UM

HPM5300

基于 RISC-V 内核的 32 位高性能微控制器

Rev 0.3

HPM5300 系列高性能微控制器用户手册

目录

1 产品概述

1.1 系统框图

本产品的系统框图如图 1。

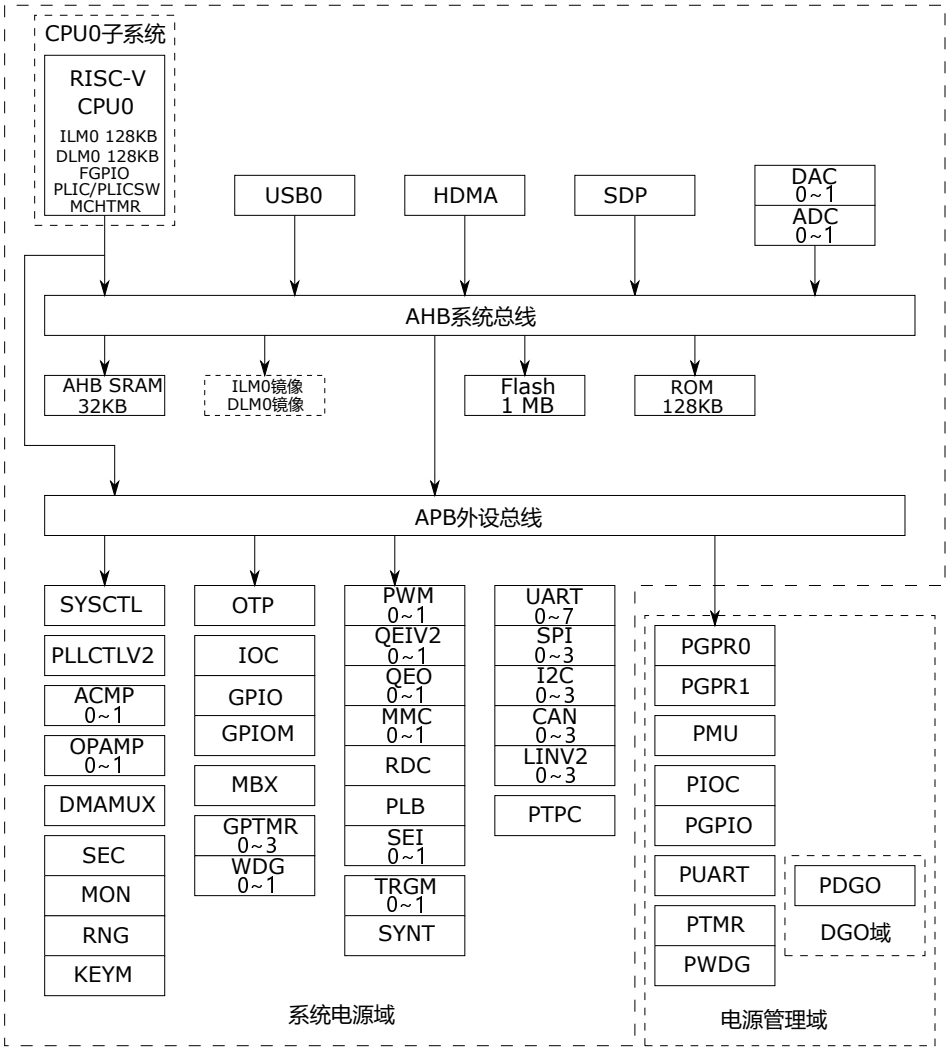


图 1: 系统架构框图

表 1总结了图 1中所有外设简称的释义。

简称	描述
CPU0 子系统	包含 RISC-V CPU0 及其本地存储器和私有外设的子系统
HART	硬件线程（Hardware Thread），RISC-V 规范定义一个可以包含完整 RISC-V 体系架构，并可以独立执行指令的单元为 HART。本手册中，HART 等同与 RISC-V 内核。
ILM	指令本地存储器（Instruction Local Memory）
DLM	数据本地存储器（Data Local Memory）
FGPIO	快速 GPIO 控制器（Fast General Purpose Input Output）
USB	通用串行总线（Universal Serial Bus）

简称	描述
SDP	安全数据处理器（Secure Data Processor）
HDMA	AHB 外设总线 DMA 控制器（AHB DMA）
AHB SRAM	AHB 总线 SRAM
EXIP	在线解密模块（Encrypted Execution-In-Place）
ADC	模数转换器（Analog-to-Digital Convertor）
DAC	数模转换器（Digital-to-Analog Convertor）
SYSCTL	系统控制模块（System Control）
PLLCTL	锁相环控制器（PLL Controller）
ACMP	模拟比较器（Analog Comparator）
MBX	信箱（Mailbox）
DMAMUX	DMA 请求路由器
IOC	IO 控制器（Input Output Controllor）
PIOC	电源管理域 IO 控制器
GPIO	通用输入输出控制器（General Purpose Input Output）
PGPIO	电源管理域 GPIO 控制器
GPIOM	GPIO 管理器（GPIO Manager）
OTP	一次性可编程存储（One Time Program）
PWM	PWM 定时器（Pulse Width Modulation）
QEIv2	正交编码器输入（Quadrature Encoder Input）
QEO	正交编码器输出（Quadrature Encoder Output）
SEI	串行编码器接口（Serial Encoder Interface）
MMC	运动管理控制器（Motion Management Controller）
RDC	旋转变压器接口（Resolver Decoder）
PLB	可编程逻辑单元（Programmable Logic Block）
TRGM	互联管理器（Trigger Manager）
SYNT	同步定时器（Sync Timer）
GPTMR	通用定时器（General Purpose Timer）
PTMR	电源管理域内的通用定时器
EWDG	看门狗（Watchdog）
PWDG	电源管理域内的看门狗
UART	通用异步收发器（Universal Asynchronous Receiver and Transmitter）
PUART	电源管理域内的通用异步收发器
SPI	串行外设接口（Serial Peripheral Interface）
I2C	集成电路总线（Inter-Integrated Circuit）
CAN	控制器局域网（Control Area Network）
LINv2	局域互联网络（Local Interconnect Network）
PTPC	精确时间协议模块（Precise Time Protocol）
RNG	随机数发生器（Random Number Generator）
KEYM	密钥管理器（Key Manager）
PGPR0/1	电源管理域的通用寄存器 0/1

简称	描述
PCFG	电源管理域配置模块
PDGO	电源管理域开关机模块
SEC	安全管理器
MON	安全监视器
系统电源域	本手册中，系统电源域专指由 VDD_SOC 供电的逻辑和存储电路
电源管理域	本手册中，电源管理域专指由 VPMC 供电的逻辑和存储电路

表 1: 外设简称总结

1.2 特性总结

本章节介绍本产品的主要特性。

1.2.1 内核与系统

32 位 RISC-V 处理器，处理器特性如下:

- RV32-IMAFDCPB 指令集
  - 整数指令集
  - 乘法指令集
  - 原子指令集
  - 单精度浮点数指令集
  - 双精度浮点数指令集
  - 压缩指令集
  - DSP 单元，支持 SIMD 和 DSP 指令，兼容 RV32-P 扩展指令集
  - 位运算指令集
- 性能可达 3.57 CoreMark / MHz
- 特权模式支持 Machine 模式和 User 模式
- 支持 16 个物理内存保护（Physical Memory Protection PMP）区域
- 支持 16KB L1 指令缓存和 16KB L1 数据缓存
- 支持 128 KB 指令本地存储器 ILM 和 128 KB 数据本地存储器 DLM

处理器配备 1 个平台中断控制器 PLIC，用于管理 RISC-V 的外部中断

- 支持多个中断源
- 支持 8 级可编程中断优先级
- 中断嵌套扩展和中断向量扩展

处理器内核配备 1 个软件中断控制器 PLICSW，管理 RISC-V 的软件中断

- 生成 RISC-V 软件中断

处理器内核配备 1 个机器定时器 MCHTMR，管理 RISC-V 的定时器中断

- 生成 RISC-V 定时器中断

DMA 控制器：

- HDMA，支持 32 个通道，用于在外设寄存器和存储器之间进行低延迟的数据搬移，也可以用于存储器之间的数据搬移

- 支持 DMA 请求路由分配到任意 DMA 控制器

包括 1 个邮箱 MBX，支持处理器不同进程间的通信：

- 支持独立的信息收发接口
- 支持生成中断

1.2.2 内部存储器

内部存储器包括：

- 288 KB 的片上 SRAM
  - ILM0，RISC-V CPU0 的指令本地存储器，128KB
  - DLM0，RISC-V CPU0 的数据本地存储器，128KB
  - AHB SRAM，32KB，适用于 HDMA 的低延时访问
- 通用寄存器
  - 电源管理域通用寄存器 PGPR，2 组各 64 字节 (共 128 字节)，可以在系统电源域掉电时保存数据
  - DGO 通用寄存器 DGO_GPR，容量 16 字节，可以在系统电源域，电源管理域掉电时保存数据
- 内部只读存储器 ROM，容量 128KB，ROM 存放本产品的启动代码，闪存加载（Flashloader）和部分外设驱动程序
- 一次性可编程存储器 OTP，4096 位，可用于存放芯片的部分出厂信息，用户密钥和安全配置，启动配置等数据

1.2.3 电源管理

本产品集成了完整的电源管理系统：

- 多个片上电源
  - DCDC 电压转换器，提供 0.9~1.3V 输出，为系统电源域的电路供电，可调节 DCDC 输出，以支持动态电压频率调整 DVFS，来优化运行时的功耗
  - LDOPMC，典型值 1.1V 输出的线性稳压器，为电源管理域的电路供电
  - LDOOTP，典型值 2.5V 输出的线性稳压器，为 OTP 供电，仅可在烧写 OTP 时打开
- 运行模式和低功耗模式：等待模式、停止模式、休眠模式和关机模式
- 芯片集成上电复位电路
- 芯片集成低压检测电路

1.2.4 时钟

本产品时钟管理系统支持多个时钟源和时钟低功耗管理：

- 外部时钟源：
  - 24MHz 片上振荡器，OSC24M，支持 24MHz 晶体，也支持通过引脚从外部输入 24MHz 有源时钟
- 内部时钟源：
  - 内部 RC 振荡器，RC24M，频率 24MHz，允许配置内部 RC 振荡器作为 PLL 的候补时钟源
  - 内部 32KHz RC 振荡器，RC32K
- 2 个锁相环 PLL，支持小数分频，支持展频
- 支持低功耗管理，支持自动时钟门控

1.2.5 复位

全局复位，可以复位整个芯片，包括电源管理域和系统电源域，复位源有：

- RESETN 引脚复位（RESETN）

系统电源域复位可以复位系统电源域，复位源有：

- VPMC 引脚的低压复位（VPMC Brownout）
- 调试复位（DEBUG RST）
- 看门狗复位（WDOGx RST）
- 软件复位（SW RST）

1.2.6 启动

BootROM 为该芯片上电后执行的第一段程序, 它支持如下功能:

- 从串行 NOR FLASH 启动
- UART/USB 启动
- 在系统编程 (ISP)
- 安全启动
- 低功耗唤醒
- 多种 ROM API

1.2.7 外部存储器

外部存储器接口包括：

- 1 个串行总线控制器 XPI，可以连接片外的各种 SPI 串行存储设备，也可以连接支持串行总线的器件，每个 XPI：
  - 支持 1/2/4 位数据模式，支持 2 个 CS 片选信号
  - 支持 SDR 和 DDR，最高支持 166MHz
  - 支持 Quad-SPI 和 Octal-SPI 的串行 NOR Flash
  - 支持串行 NAND Flash
  - 支持 HyperBus，HyperRAM 和 HyperFlash
  - 支持 Quad/Oct SPI PSRAM

1.2.8 运动控制系统

运动控制系统包括：

- 2 组电机控制系统，每组电机控制系统配备有：
  - 2 个 8 通道 PWM 定时器 PWM，PWM 调制精度达 3.0ns，支持产生互补 PWM 输出，死区插入和故障保护
  - 2 个正交编码器输入 QEIv2
  - 2 个正交编码器输出 QEO
  - 2 个运动管理控制器 MMC
  - 1 个旋转变压器解码 RDC
  - 1 个可编程逻辑单元 PLB
  - 2 个串行编码器接口 SEI

  - 1 个互联管理器 TRGM
- 各模块支持通过互联管理器 TRGM 与电机控制系统内部或外部的模块交互
- 1 个同步定时器，用于同步

1.2.9 定时器

定时器包括：

- 5 组 32 位通用定时器，其中一组 (PTMR) 位于电源管理域，支持低功耗唤醒，每组通用定时器包括 4 个 32 位计数器
- 3 个看门狗，其中一个 (PWDG) 位于电源管理域

1.2.10 通讯外设

支持丰富的通讯外设，包括：

- 9 个通用异步收发器 UART，其中 1 个 (PUART) 位于电源管理域，支持低功耗唤醒
- 4 个串行外设接口 SPI
- 4 个集成电路总线 I2C，支持标准（100kbps），快速（400kbps）和快速 +（1 Mbps）
- 4 个控制器局域网 CAN，支持 CAN_FD
  - 支持 CAN 2.0B 标准，1Mbps
  - 支持 CAN FD，8 Mbps
  - 支持时间戳
- 1 个精确时间协议模块 PTPC，PTPC 支持 2 组时间戳模块，每组包含 64 位计数器，连接到 CAN 模块，CAN 模块可以随时从端口读取时间戳信息
- 4 个局域互联网络 LIN
- 1 个 USB OTG 控制器，集成 1 个高速 USB-PHY
  - 符合 *Universal Serial Bus Specification Rev. 2.0*

1.2.11 模拟外设

模拟外设包括：

- 2 个 16 位模拟数字转换器 ADC
  - 16 位逐次逼近型 ADC
  - 支持 16 个输入通道
  - 2M 采样率，4M 采样率（转换精度设置为 12 位）
- 2 个高速比较器
  - 工作电压 3.0 ~ 3.6V，支持轨到轨输入
  - 内置 8 位 DAC
- 2 个数模转换器 DAC
  - 12 位精度，1MSPS，支持输出缓存
- 2 个运算放大器 OPAMP
  - 支持 PGA 模式

1.2.12 输入输出

- 提供 PA~PY 共 8 组最多 56 个 GPIO 功能复用引脚
- IO 支持 3V 电压模式，分组供电

- IO 支持开漏控制、内部上下拉、驱动能力调节，内置施密特触发器
- GPIO 控制器
  - 支持读取任意 IO 的输入或者控制 IO 的输出
  - 支持 IO 输入触发中断
- 快速 GPIO 控制器 FGPIO，作为处理器私有的 IO 快速访问接口
- 提供一个 GPIO 管理器，管理各 GPIO 控制器的 IO 控制权限
- 电源管理域专属 IO PYxx 拥有专属 GPIO 控制器和 IO 配置模块，支持低功耗模式下状态保持

1.2.13 信息安全系统

信息安全模块包含：

- 安全数据处理器 SDP，为片上加解密算法引擎：
  - 支持 AES-128/256，SM4，支持 ECB 模式和 CBC 模式
  - 支持 SHA-1/SHA-256，SM3
- 在线解密模块 EXIP：
  - 与串行总线控制器 XPI 紧密耦合，支持外部 NOR Flash 在线解密
  - AES-128 CTR 模式，零等待周期解密
  - 支持 RFC3394 的密钥解封，通过密钥加密密钥 KEK 保护数据加密密钥 DEK
- 密钥管理器 KEYM：
  - 支持通过独立的数据通路从 OTP 的密钥区载入密钥
  - 支持密钥混淆
  - 支持从真随机数发生器 RNG 载入随机密钥
  - 支持生成 Session Key
  - 支持独立的数据通路将密钥传送到安全数据处理器 SDP
- OTP 中的密钥区，支持存放并保护；
  - SDP，EXIP 的相关密钥
  - 安全启动的相关密钥
  - 安全调试相关密钥
  - 产品生命周期配置
- 真随机数发生器 RNG：
  - 3 个独立熵源为内部模拟噪声源
- 安全管理器 SEC：
  - 监测产品生命周期
  - 配置系统安全状态，
  - 制定安全规则并监测安全规则违反的事件
  - 关联监视器 MON，监测 VPMC 供电和时钟 OSC24M
- 基于 BOOT ROM 的安全启动机制，支持加密启动，支持可信的执行环境

1.2.14 系统调试

系统调试模块包括：

- 支持 JTAG 接口
  - 支持 *RISC-V External Debug Support V0.13* 规范
  - 支持 IEEE1149.1

  - 访问 RISC-V 内核寄存器和 CSR，访问存储器
- 调试端口锁定功能
  - 开放模式，调试功能开放
  - 锁定模式，调试功能关闭，可以通过调试密钥解锁
  - 关闭模式，调试功能关闭

1.3 文档约定

1.3.1 寄存器相关缩写词列表

本手册的寄存器说明中采用了如下缩写词

缩写	描述
ro（只读）	软件只能读该位。
rc（只读，读清零）	软件只能读该位，读后清零。
wo（只写）	软件只能写该位。
w1c（写 1 清零）	软件可以通过写入 1 将该位清零。写入 0 对该位的值无影响。
w1s（写 1 置 1）	软件可以通过写入 1 将该位置 1。写入 0 对该位的值无影响。
wc（写清零）	软件可以通过写入任意值将该位清零。
ws（写置 1）	软件可以通过写入任意值将该位置 1。
rw（读/写）	软件可以读写该位。
rw1c（可读/写 1 清零）	软件可以读该位，也可以通过写入 1 将该位清零。写入 0 对该位的值无影响。
rw1s（可读/写 1 置 1）	软件可以读该位，也可以通过写入 1 将该位置 1。写入 0 对该位的值无影响。
rwc（可读/写清零）	软件可以读该位，也可以通过写入任意值将该位清零。
rws（可读/写置 1）	软件可以读该位，也可以通过写入任意值将该位置 1。

表 2: 寄存器描述缩写词列表

2 处理器内核

本产品集成了高性能 RISC-V 处理器作为 CPU，符合以下 RISC-V 规范：

- ***The RISC-V Instruction Set Manual, Volume I: User-Level ISA, Version 2.2***
- ***The RISC-V Instruction Set Manual Volume II: Privileged Architecture Version 1.11***
- ***The RISC-V Debug Specification, Version 0.13***

2.1 中央处理器

RISC-V32 位高性能嵌入式处理器，支持以下指令集：

- RISC-V, RV32I：基础整数指令集
- RISC-V, M 扩展：乘法和除法指令集
- RISC-V, A 扩展：原子指令集
- RISC-V, F 扩展：单精度浮点数指令集
- RISC-V, D 扩展：双精度浮点数指令集
- RISC-V, C 扩展：压缩指令集
- RISC-V, B 扩展：位操作指令集

同时也支持扩展指令集：

- RISC-V, P 扩展：SIMD 和 DSP 扩展指令集

其他特性：

- 动态分支预测
- 处理器性能监视器
- 非对齐的存储器访问

2.2 总线和存储器接口

RISC-V 处理器配置为支持以下总线接口:

- 总线主接口，处理器以之访问片上的内存和其他资源。
- 总线从接口，片上其他的总线主设备可以以之访问处理器的数据和指令本地存储器
- 指令本地存储器接口，处理器以之访问高速指令本地存储器
- 数据本地存储器接口，处理器以之访问高速数据本地存储器

2.3 TRAP

按照 RISC-V 规范，由异常或者中断引起的处理器指令执行控制流程转换称为 trap，其中异常由处理器自身的指令执行引发，而中断是由处理器内外部的中断源生成。当 trap 发生时，处理器会中断当前的指令执行流程，关闭中断，保存需要的内核通用寄存器到堆栈，随后执行相应的 trap 服务程序。

有关 RISC-V 处理器 TRAP 的细节，请用户查询章 4。

本产品所有支持生成中断的外设，及其中断向量表，请查阅节 4.4。

2.4 机器定时器 MCHTMR

机器定时器是个 64 位的定时器，符合 RISC-V 规范，这个定时器以固定的时钟运行，提供固定的时间基准，并能够产生机器定时器中断 (Machine Timer Interrupt)。机器定时器 MCHTMR，包含 64 位的计数器 mtime 和 64

位的比较器 mtimecmpx，当计数器 mtime >= mtimecmpx 时，产生中断。

2.5 硬件性能监视器 (Hardware Performance Monitor)

硬件性能监视器符合 RISC-V 规范，包含：

- mcycle / mcycleh CSR，64 位的计数器，统计特定时刻以来处理器运行的时钟周期数。
- minstret / minstreth CSR，64 位的计数器，统计特定时刻以来处理器执行完成（instruction retired）的指令数目。
- hpmcounter3~6 / hpmcounterh3~6 CSR，4 个 64 位的事件计数器，通过配置 CSR mhpmevent3~6，可以配置事件计数器统计特定时刻以来：
  - 特定指令执行的数量，指令可以是 LOAD，STORE，乘法等整数指令，浮点数相关指令，跳转，返回等程序流控指令
  - 处理器本地存储器 ILM 和 DLM 的访问次数
  - 缓存相关的事件数目：如指令或数据缓存命中，MISS 次数等
  - 分支预测模块预测失败次数等

2.6 特权模式

本产品符合 The RISC-V Instruction Set Manual Volume II: Privileged Architecture Version 1.11 规范。

- 支持机器模式，也称为 Machine Mode，M-mode，M 模式
- 支持用户模式，也称为 User Mode，U-mode，U 模式

支持物理存储保护 Physical Memory Protection(PMP)。支持 16 个 region。

2.7 物理内存保护（Physical Memory Protection）

RISC-V 规范定义，RISC-V 处理器可以支持 PMP (Physical Memory Protection) 模块。PMP 支持对存储器映射的指定地址区间提供读，写和执行代码保护。PMP 的读，写，代码执行检查针对 CPU 的特权模式，即允许用户通过配置 PMP，向用户模式 User Mode 下的软件授权对指定地址区间的读，写或者代码执行权限。

用户通过配置 PMP，可以撤回机器模式 Machine Mode 下执行的软件，对指定地址区间的读，写或者代码执行权限。

本产品的 RISC-V 处理器，支持 16 个 PMP 入口 entry，可以通过 4 个 PMP CFG CSR 和 16 个 PMP ADDR CSR，配置这 16 个 PMP 入口。

用户可以通过 PMP 的 ADDR CSR 配置指定内存区域的基地址和长度。

用户可以配置 PMP 的 CFG CSR，指定当 CPU 处于用户模式 User Mode 时，对指定的地址区间，可读，可写，或者可执行代码。

用户可以锁定 PMP CSR 的配置。一旦锁定，即使 CPU 在机器模式 Machine Mode 下，在下次复位前，也不能再修改 PMP 的相关 CSR。同时，一旦锁定，PMP 的配置不再只针对用户模式 User Mode 生效，对机器模式 Machine Mode 也有效。

2.8 相关文档

RISC-V 开源指令集的相关信息，请访问 RISC-V 主页 https://riscv.org/。

RISC-V 指令集相关规范，请访问 https://riscv.org/technical/specifications/。

本产品 RISC-V CPU 的 DSP 扩展指令相关信息，请访问 http://www.andestech.com/en/products-solutions/product-documentation/ 并下载《AndeStar V5 DSP ISA Extension Specification》。

3 RISC-V 处理器的控制状态寄存器

本章节列举了本产品 RISC-V 处理器内核的控制状态寄存器组（control and status register），简称为 CSR。

CSR 属于 RISC-V CPU 私有的寄存器，RISC-V 规范定义了一系列标准的 CSR，处理器厂家也可以在 CPU 中实现 CSR 支持非标准的功能。

CSR 主要功能有：

- 包含处理器固有信息相关：例如处理器的厂商信息，架构信息，核心数等；
- 中断相关：例如中断开关以及中断入口等信息；
- 中断响应相关：例如中断原因，中断返回地址等信息；
- 存储器保护相关：设置不同地址空间的存储器的访问属性，例如可读可写可执行等等；
- 性能统计相关和调试接口相关

RISC-V 指令集定义了一系列访问 CSR 的指令，用户可以通过这些指令来操作 CSR 寄存器，如：

```
csrr rd, csr  //Read CSR
csrw csr, rs  //Write CSR
```

用户可以查询 RISC-V 相关规范以了解更多的相关信息。

3.1 控制状态寄存器 CSR 说明

控制状态寄存器 CSR 列表如下，注意列表中的地址偏移，代表 RISC-V CPU 的 CSR 的编码空间地址：

地址偏移	名称	描述	类型	复位值
0x000	USTATUS	用户状态寄存器	standard read/write	0x00000000
0x004	UIE	用户中断使能	standard read/write	0x00000000
0x005	UTVEC	用户 trap 向量基地址	standard read/write	0x00000000
0x040	USCRATCH	用户暂存登记表	standard read/write	0x00000000
0x041	UEPC	用户异常程序计数器	standard read/write	0x00000000
0x042	UCAUSE	用户原因寄存器	standard read/write	0x00000000
0x043	UTVAL	用户 trap 值	standard read/write	0x00000000
0x044	UIP	用户中断未决	standard read/write	0x00000000
0x300	MSTATUS	机器模式状态	standard read/write	0x00001800
0x301	MISA	机器模式指令集架构寄存器	standard read/write	-
0x304	MIE	机器模式中断使能	standard read/write	0x00000000
0x305	MTVEC	机器模式 trap 向量基地址	standard read/write	0x00000000
0x306	MCOUNTEREN	机器计数器使能	standard read/write	0x00000000
0x320	MCOUNTINHIBIT	机器模式禁止计数	non-standard read/write	0x00000000
0x323	MHPMEVENT3	机器模式性能监控事件选择器	standard read/write	0x00000000
0x324	MHPMEVENT4	机器模式性能监控事件选择器	standard read/write	0x00000000

地址偏移	名称	描述	类型	复位值
0x325	MHPMEVENT5	机器模式性能监控事件选择器	standard read/write	0x00000000
0x326	MHPMEVENT6	机器模式性能监控事件选择器	standard read/write	0x00000000
0x340	MSCRATCH	机器模式擦写寄存器	standard read/write	0x00000000
0x341	MEPC	机器模式异常程序计数器	standard read/write	0x00000000
0x342	MCAUSE	机器模式异常原因寄存器	standard read/write	0x00000000
0x343	MTVAL	机器模式 trap 值寄存器	standard read/write	0x00000000
0x344	MIP	机器模式中断未决	standard read/write	0x00000000
0x3A0	PMPCFG0	PMP 配置寄存器	standard read/write	0x00000000
0x3A1	PMPCFG1	PMP 配置寄存器	standard read/write	0x00000000
0x3A2	PMPCFG2	PMP 配置寄存器	standard read/write	0x00000000
0x3A3	PMPCFG3	PMP 配置寄存器	standard read/write	0x00000000
0x3B0	PMPADDR[PMPADDR0]	PMP 地址寄存器	standard read/write	-
0x3B1	PMPADDR[PMPADDR1]	PMP 地址寄存器	standard read/write	-
0x3B2	PMPADDR[PMPADDR2]	PMP 地址寄存器	standard read/write	-
0x3B3	PMPADDR[PMPADDR3]	PMP 地址寄存器	standard read/write	-
0x3B4	PMPADDR[PMPADDR4]	PMP 地址寄存器	standard read/write	-
0x3B5	PMPADDR[PMPADDR5]	PMP 地址寄存器	standard read/write	-
0x3B6	PMPADDR[PMPADDR6]	PMP 地址寄存器	standard read/write	-
0x3B7	PMPADDR[PMPADDR7]	PMP 地址寄存器	standard read/write	-
0x3B8	PMPADDR[PMPADDR8]	PMP 地址寄存器	standard read/write	-
0x3B9	PMPADDR[PMPADDR9]	PMP 地址寄存器	standard read/write	-
0x3BA	PMPADDR[PMPADDR10]	PMP 地址寄存器	standard read/write	-
0x3BB	PMPADDR[PMPADDR11]	PMP 地址寄存器	standard read/write	-
0x3BC	PMPADDR[PMPADDR12]	PMP 地址寄存器	standard read/write	-
0x3BD	PMPADDR[PMPADDR13]	PMP 地址寄存器	standard read/write	-
0x3BE	PMPADDR[PMPADDR14]	PMP 地址寄存器	standard read/write	-
0x3BF	PMPADDR[PMPADDR15]	PMP 地址寄存器	standard read/write	-
0x7A0	TSELECT	trigger 选择	standard read/write	0x00000000
0x7A1	TDATA1	trigger 数据 1	standard read/write	0x20000000
0x7A1	MCONTROL	匹配控制	standard read/write	0x22400000
0x7A1	ICOUNT	指令计数	standard read/write	0x20000400
0x7A1	ITRIGGER	中断触发	standard read/write	0x20000000
0x7A1	ETRIGGER	异常 trigger	standard read/write	0x20000000
0x7A2	TDATA2	trigger 数据 2	standard read/write	0x00000000
0x7A3	TDATA3	triggr 数据 3	standard read/write	0x00000000
0x7A3	TEXTRA	额外触发	standard read/write	0x00000000
0x7A4	TINFO	trigger 信息	standard read/write	-
0x7A5	TCONTROL	trigger 控制	standard read/write	0x00000000

地址偏移	名称	描述	类型	复位值
0x7A8	MCONTEXT	机器模式上下文	standard read/write	0x00000000
0x7AA	SCONTEXT	特权模式上下文	standard read/write	0x00000000
0x7B0	DCSR	调试控制和状态寄存器	debug-mode-only	0x40000603
0x7B1	DPC	调试程序计数器	debug-mode-only	0x00000000
0x7B2	DSCRATCH0	调试暂存寄存器 0	debug-mode-only	0x00000000
0x7B3	DSCRATCH1	调试暂存寄存器 1	debug-mode-only	0x00000000
0x7C0	MILMB	指令本地存储器基址寄存器	non-standard read/write	-
0x7C1	MDLMB	数据本地存储器基址寄存器	non-standard read/write	-
0x7C2	MECC_CODE	ECC 代码寄存器	non-standard read/write	0x00000001
0x7C3	MNVEC	NMI 向量基地址寄存器	non-standard read/write	-
0x7C4	MXSTATUS	机器模式扩展状态	non-standard read/write	0x00000000
0x7C5	MPFT_CTL	性能节流控制寄存器	non-standard read/write	0x00000000
0x7C6	MHSP_CTL	机器模式硬件堆栈保护控制	non-standard read/write	0x00000000
0x7C7	MSP_BOUND	机器模式 SP 绑定寄存器	non-standard read/write	0xFFFFFFFF
0x7C8	MSP_BASE	机器模式 SP 基址寄存器	non-standard read/write	0xFFFFFFFF
0x7C9	MDCAUSE	机器模式详细 trap 原因	non-standard read/write	0x00000000
0x7CA	MCACHE_CTL	缓存控制寄存器	non-standard read/write	0x00000800
0x7CB	MCCTLBEGINADDR	机器模式 CCTL 起始地址	non-standard read/write	0x00000000
0x7CC	MCCTLCOMMAND	机器模式 CCTL 命令	non-standard read/write	0x00000000
0x7CD	MCCTLDATA	机器模式 CCTL 数据	non-standard read/write	0x00000000
0x7CE	MCOUNTERWEN	机器计数器写使能	non-standard read/write	0x00000000
0x7CF	MCOUNTERINTEN	机器计数器中断使能	non-standard read/write	0x00000000
0x7D0	MMISC_CTL	机器模式杂项控制寄存器	non-standard read/write	0x00000048

地址偏移	名称	描述	类型	复位值
0x7D1	MCOUNTERMASK_M	机器模式的机器计数器掩码	non-standard read/write	0x00000000
0x7D2	MCOUNTERMASK_S	特权模式的机器计数器掩码	non-standard read/write	0x00000000
0x7D3	MCOUNTERMASK_U	用户模式的机器计数器掩码	non-standard read/write	0x00000000
0x7D4	MCOUNTEROVF	机器计数器溢出状态	non-standard read/write	0x00000000
0x7E0	DEXC2DBG	异常重定向寄存器	non-standard read/write	0x00000000
0x7E1	DDCAUSE	调试异常详细原因	non-standard read/write	0x00000000
0x800	UITB	指令表基地址寄存器	non-standard read/write	0x00000000
0x801	UCODE	代码寄存器	non-standard read/write	0x00000000
0x809	UDCAUSE	用户详细的 trap 原因	non-standard read/write	0x00000000
0x80B	UCCTLBEGINADDR	用户 CCTL 起始地址	non-standard read/write	0x00000000
0x80C	UCCTLCOMMAND	用户 CCTL 命令	non-standard read/write	0x00000000
0xB00	MCYCLE	机器模式循环计数器	standard read/write	0x00000000
0xB02	MINSTRET	机器模式指令报废计数器	standard read/write	0x00000000
0xB03	MHPMCOUNTER3	机器模式性能监控计数器	standard read/write	0x00000000
0xB04	MHPMCOUNTER4	机器模式性能监控计数器	standard read/write	0x00000000
0xB05	MHPMCOUNTER5	机器模式性能监控计数器	standard read/write	0x00000000
0xB06	MHPMCOUNTER6	机器模式性能监控计数器	standard read/write	0x00000000
0xB80	MCYCLEH	机器模式循环计数器	standard read/write	0x00000000
0xB82	MINSTRETH	机器模式指令报废计数器	standard read/write	0x00000000
0xB83	MHPMCOUNTER3H	机器模式性能监控计数器	standard read/write	0x00000000
0xB84	MHPMCOUNTER4H	机器模式性能监控计数器	standard read/write	0x00000000
0xB85	MHPMCOUNTER5H	机器模式性能监控计数器	standard read/write	0x00000000
0xB86	MHPMCOUNTER6H	机器模式性能监控计数器	standard read/write	0x00000000
0xC00	CYCLE	CYCLE 寄存器		0x00000000
0xC80	CYCLEH	CYCLE 高 32 位寄存器		0x00000000
0xF11	MVENDORID	机器模式供应商 ID 寄存器	standard read only	0x0000031E
0xF12	MARCHID	机器模式架构 ID 寄存器	standard read only	0x00000025
0xF13	MIMPID	机器模式实现 ID 寄存器	standard read only	-
0xF14	MHARTID	HartID 寄存器	standard read only	0x00000000

地址偏移	名称	描述	类型	复位值

表 3: CSR 寄存器列表

3.2 控制状态寄存器 CSR 详细信息

控制状态寄存器 CSR 的详细说明如下：

3.2.1 USTATUS (0x0) (standard read/write)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																											UPIE	RSVD			UIE
N/A																											RW	N/A			RW
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	x	x	x	0

USTATUS [31:0]

位域	名称	描述
4	UPIE	UPIE 在异常之前保存 UIE 位的值。
0	UIE	U 模式中断使能位。 0：禁用 1：启用

USTATUS 位域

3.2.2 UIE (0x4) (standard read/write)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																							UEIE	RSVD			UTIE	RSVD			USIE
N/A																							RW	N/A			RW	N/A			RW
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	x	x	x	0	x	x	x	0

UIE [31:0]

位域	名称	描述
8	UEIE	U 模式外部中断使能位 0：禁用 1：启用
4	UTIE	U 模式定时器中断使能位。 0：禁用 1：启用

位域	名称	描述
0	USIE	U 模式软件中断使能位。 0：禁用 1：启用

UIE 位域

3.2.3 UTVEC (0x5) (standard read/write)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BASE_31_2																														RSVD	
RW																														N/A	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	x	x

UTVEC [31:0]

位域	名称	描述
31-2	BASE_31_2	中断和异常处理程序的基址。当 PLIC 处于矢量模式时，请参阅上面的描述以了解对齐要求。

UTVEC 位域

3.2.4 USCRATCH (0x40) (standard read/write)

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

USCRATCH [31:0]

位域	名称	描述
31-0	USCRATCH	暂存寄存器存储。

USCRATCH 位域

3.2.5 UEPC (0x41) (standard read/write)

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

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RW																															N/A
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	x

UEPC [31:0]

位域	名称	描述
31-1	EPC	异常程序计数器。

UEPC 位域

3.2.6 UCAUSE (0x42) (standard read/write)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
INTERRUPT	RSVD																					EXCEPTION_CODE									
RW	N/A																					RW									
0	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0

UCAUSE [31:0]

位域	名称	描述
31	INTERRUPT	中断

位域	名称	描述
9-0	EXCEPTION_CODE	异常代码。 当中断为 1 时： 0：用户软件中断 4：用户定时器中断 8：用户外部中断 当中断为 0 时： 0：指令地址未对齐 1：指令访问错误 2：非法指令 3：断点 4：加载地址未对齐 5：负载访问故障 6：Store/AMO 地址未对齐 7：Store/AMO 访问故障 8：U-mode 环境调用 9-11：保留 12：指令缺页 13：加载页面错误 14：保留 15: Store/AMO 页面错误 32：堆栈溢出异常 33：堆栈下溢异常 40-47：保留

UCAUSE 位域

3.2.7 UTVAL (0x43) (standard read/write)

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

UTVAL [31:0]

位域	名称	描述
31-0	UTVAL	软件 trap 处理的异常特定信息。

UTVAL 位域

3.2.8 UIP (0x44) (standard read/write)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																							UEIP	RSVD			UTIP	RSVD			USIP
N/A																							RW	N/A			RW	N/A			RW
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	x	x	x	0	x	x	x	0

UIP [31:0]

位域	名称	描述
8	UEIP	U 模式外部中断挂起位。 0：未挂起 1：待定
4	UTIP	U 模式定时器中断挂起位。 0：未挂起 1：待定
0	USIP	U 模式软件中断挂起位。 0：未挂起 1：待定

UIP 位域

3.2.9 MSTATUS (0x300) (standard read/write)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SD	RSVD											MXR	RSVD	MPRV	XS		FS		MPP		RSVD			MPIE	RSVD		UPIE	MIE	RSVD		UIE
RO	N/A											RW	N/A	RW	RO		RW		RW		N/A			RW	N/A		RW	RW	N/A		RW
0	x	x	x	x	x	x	x	x	x	x	x	0	x	0	0	0	0	0	1	1	x	x	x	0	x	x	0	0	x	x	0

MSTATUS [31:0]

位域	名称	描述
31	SD	SD 总结了 FS 字段或 XS 字段是否脏。
19	MXR	MXR 控制只执行页是否可读。当基于页的虚拟内存无效时它没有作用 0：只执行页面不可读 1：只执行页面是可读的
17	MPRV	当 MPRV 位被设置时，加载和存储的内存访问权限由 MPP 字段指定。当 U 模式不可用时，该字段被硬连线为 0。

位域	名称	描述
16-15	XS	XS 保存 ACE 指令的架构状态（ACE 寄存器）的状态。如果没有配置 ACE 扩展，这个字段的值为零。这个字段主要由软件管理。处理器硬件在两个方面辅助状态管理： XS 关闭时会触发非法指令异常。 当 XS 不是 Off 时，随着 ACE 指令的执行，XS 更新为 Dirty 状态。更改此字段的设置对 ACE 状态的内容没有影响。特别是，将 XS 设置为 Off 不会破坏状态，将 XS 设置为 Initial 也不会清除内容。 0：关 1：初始 2：清洁 3：脏
14-13	FS	FS 保存着浮点单元架构状态的状态，包括 fcsr CSR 和 f0 -f31 浮点数据寄存器。如果处理器没有 FPU，则该字段的值为零且只读。该字段主要由软件管理，在硬件电路方面，处理器可以在两方面进行辅助管理： 当 FS 关闭时，尝试访问 fcsr 或任何 f 寄存器会引发非法指令异常。 当 FS 为 Initial 或 Clean 时，通过执行更新 fcsr 或任何 f 寄存器的任何指令，FS 将更新为 Dirty 状态。更改此字段的设置对浮点寄存器状态的内容没有影响。特别是，将 FS 设置为 Off 不会破坏状态，将 FS 设置为 Initial 也不会清除内容。 0：关 1：初始 2：干净 3：脏
12-11	MPP	MPP 持有 trap 之前的特权模式。特权模式的编码在表 5 中描述。当 U 模式不可用时，该字段被硬连接到 3。
7	MPIE	MPIE 在 trap 之前保存 MIE 位的值。
4	UPIE	UPIE 在 trap 之前保存 UIE 位的值。
3	MIE	M 模式中断使能位。 0：禁用 1：启用
0	UIE	U 模式中断使能位。 0：禁用 1：启用

MSTATUS 位域

3.2.10 MISA (0x301) (standard read/write)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BASE		RSVD				Z	Y	X	W	V	U	T	S	R	Q	P	O	N	M	L	K	J	I	H	G	F	E	D	C	B	A
RO		N/A				RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
0	1	x	x	x	x	0	0	1	0	0	*	0	*	0	0	0	0	*	1	0	0	0	1	0	0	*	0	*	1	0	*

MISA [31:0]

位域	名称	描述
31-30	BASE	本机基本整数 ISA 的通用寄存器宽度。 0：保留 1:32 2:64 3:128
25	Z	保留
24	Y	保留
23	X	存在非标准扩展
22	W	保留
21	V	暂为 Vector 扩展保留
20	U	用户模式实现 0：机器 1：机器 + 用户/机器 + 特权 + 用户
19	T	暂时保留用于事务性内存扩展
18	S	包含了特权模式 0：机器/机器 + 用户 1：机器 + 特权 + 用户
17	R	保留
16	Q	四精度浮点扩展
15	P	暂时保留用于 Packed-SIMD 扩展
14	O	保留
13	N	支持用户级中断 0：否 1：是
12	M	整数乘法/除法扩展
11	L	暂时保留用于十进制浮点扩展
10	K	保留
9	J	暂时保留用于动态翻译语言扩展
8	I	RV32I/64I/128I 基础 ISA
7	H	保留
6	G	存在额外的标准扩展

位域	名称	描述
5	F	单精度浮点扩展 0：无 1：双精度 + 单精度/单精度
4	E	RV32E 基础 ISA
3	D	双精度浮点扩展 0：单精度/无 1：双精度 + 单精度
2	C	压缩扩展
1	B	暂保留用于位操作扩展
0	A	原子扩展 0：否 1：是

MISA 位域

3.2.11 MIE (0x304) (standard read/write)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD													PMOVI	BWEI	IMECCI	RSVD				MEIE	RSVD		UEIE	MTIE	RSVD		UTIE	MSIE	RSVD		USIE
N/A													RW	RW	RW	N/A				RW	N/A		RW	RW	N/A		RW	RW	N/A		RW
x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	x	x	x	x	0	x	x	0	0	x	x	0	0	x	x	0

MIE [31:0]

位域	名称	描述
18	PMOVI	性能监视器溢出本地中断使能位 0：禁用 1：启用
17	BWEI	总线读/写事务错误本地中断使能位。处理器可能会收到加载/存储指令或缓存写回的总线错误。 0：禁用 1：启用
16	IMECCI	不精确的 ECC 错误本地中断使能位。处理器可能会在从端口访问或缓存写回时收到不精确的 ECC 错误。 0：禁用 1：启用
11	MEIE	M 模式外部中断使能位 0：禁用 1：启用

位域	名称	描述
8	UEIE	U 模式外部中断使能位 0：禁用 1：启用
7	MTIE	M 模式定时器中断使能位。 0：禁用 1：启用
4	UTIE	U 模式定时器中断使能位。 0：禁用 1：启用
3	MSIE	M 模式软件中断使能位 0：禁用 1：启用
0	USIE	U 模式软件中断使能位。 0：禁用 1：启用

MIE 位域

3.2.12 MTVEC (0x305) (standard read/write)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BASE_31_2																														RSVD	
RW																														N/A	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	x	x

MTVEC [31:0]

位域	名称	描述
31-2	BASE_31_2	中断和异常处理程序的基址。当 PLIC 处于矢量模式时，请参阅上面的描述以了解对齐要求

MTVEC 位域

3.2.13 MCOUNTEREN (0x306) (standard read/write)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																									HPM6	HPM5	HPM4	HPM3	IR	TM	CY
N/A																									RW	RW	RW	RW	RW	RW	RW
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0

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

MCOUNTEREN [31:0]

位域	名称	描述
6	HPM6	见寄存器说明
5	HPM5	见寄存器说明
4	HPM4	见寄存器说明
3	HPM3	见寄存器说明
2	IR	见寄存器说明
1	TM	见寄存器说明
0	CY	见寄存器说明

MCOUNTEREN 位域

3.2.14 MCOUNTINHIBIT (0x320) (non-standard read/write)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																									HPM6	HPM5	HPM4	HPM3	IR	TM	CY
N/A																									RW	RW	RW	RW	RW	RW	RW
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0

MCOUNTINHIBIT [31:0]

位域	名称	描述
6	HPM6	请参阅寄存器说明。
5	HPM5	请参阅寄存器说明。
4	HPM4	请参阅寄存器说明。
3	HPM3	请参阅寄存器说明。
2	IR	请参阅寄存器说明。
1	TM	请参阅寄存器说明。
0	CY	请参阅寄存器说明。

MCOUNTINHIBIT 位域

3.2.15 MHPMEVENT3 (0x323) (standard read/write)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																							SEL					TYPE			
N/A																							RW					RW			
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0

MHPMEVENT3 [31:0]

位域	名称	描述
8-4	SEL	参见事件选择器表
3-0	TYPE	参见事件选择器表

MHPMEVENT3 位域

TYPE	SEL	事件名称	解释
0	1	Cycle count	Number of elapsed processor clock cycles
0	2	Retired instruction count	Number of retired instructions
0	3	Integer load instruction count	Number of retired load instructions (including LR).
0	4	Integer store instruction count	Number of retired store instructions (including SC).
0	5	Atomic instruction count	Number of retired atomic instructions (not including LR and SC).
0	6	System instruction count	Number of retired SYSTEM instructions(instructions with major opcode equal to 0b1110011).
0	7	Integer computational instruction count	Number of retired integer computational instructions.
0	8	Conditional branch instruction count	Number of retired conditional branch instructions.
0	9	Taken conditional branch instruction count	Number of retired conditional branch instructions that are taken.
0	10	JAL instruction count	Number of retired JAL instructions.
0	11	JALR instruction count	Number of retired JALR instructions.This event selector also counts the events monitored by the return instruction count event selector defined in the next row.
0	12	Return instruction count	Number of retired return instructions. Return instructions are JALR instructions with zero immediate offset and the following operands: • (rd != x1/x5) and (rs1 == x1/x5) • rd == x1 and rs1 == x5 • rd == x5 and rs1 == x1
0	13	Control transfer instruction count	Number of retired unconditional jumps (JAL and JALR) and conditional branch instructions.
0	14	EXEC.IT instruction count	Number of retired EXEC.IT instructions.
0	15	Integer multiplication instruction count	Number of retired integer multiplication instructions.
0	16	Integer division instruction count	Number of retired integer division/remainder instructions.
0	17	Floating-point load instruction count	Number of retired floating-point load instructions.

TYPE	SEL	事件名称	解释
0	18	Floating-point store instruction count	Number of retired floating-point store instructions.
0	19	Floating-point addition instruction count	Number of retired floating-point addition/subtraction instructions.
0	20	Floating-point multiplication instruction count	Number of retired floating-point multiplication instructions.
0	21	Floating-point fused multiply-add instruction count	Number of retired floating-point fused multiply-add/subtraction instructions (FMADD, FMSUB, FNMSUB, FNMADD).
0	22	Floating-point division or square-root instruction count	Number of retired floating-point division/square-root instructions.
0	23	Other floating-point instruction count	Number of retired floating-point instructions not counted by the previous floating-point instruction event selectors
0	24	Integer multiplication and add/sub instruction count	Number of retired integer multiplication and add/sub instructions.
0	25	Retired operation count	Number of retired operations. a floating-point multiply-add instruction is counted as 2 operations. other instructions are counted as 1 operation
1	0	ILM access	Number of ILM transfers, including speculative instruction fetch, load/store accesses, ECC repair and slave port accesses.
1	1	DLM access	Number of DLM transfers, including speculative load/store accesses, ECC repair and slave port accesses.
1	2	I-Cache access	Number of completed I-Cache fetch access.
1	3	I-Cache miss	Number of I-Cache fetch miss.
1	4	D-Cache access	Number of completed D-Cache load-and-store access. Misaligned load/store accesses might increase this counter by either one or two, depending on access sizes and alignments. Only misaligned accesses crossing two cache lines are guaranteed to result in increment of two.
1	5	D-Cache miss	The event counts the number of D-Cache load-and-store miss. Misaligned load/store accesses might increase this counter by either zero, one or two, depending on access sizes, alignments and whether the accessed lines are in D-Cache.
1	6	D-Cache load access	Number of completed D-Cache load access. See the D-Cache access count event selector for the handling of misaligned load accesses.

TYPE	SEL	事件名称	解释
1	7	D-Cache load miss	Number of D-Cache load miss. See the D-Cache miss count event selector for the handling of misaligned load accesses.
1	8	D-Cache store access	Number of completed D-Cache store access. See the D-Cache access count event selector for the handling of misaligned load accesses.
1	9	D-Cache store miss	Number of D-Cache store miss. See the D-Cache miss count event selector for the handling of misaligned load accesses.
1	10	D-Cache writeback	Number of D-Cache writeback.
1	11	Cycles waiting for I-Cache fill data	Number of cycles waiting for the return of the critical word of I-Cache misses from the system bus. This event selector does not monitor accesses to I/O regions or accesses to cacheable regions when I-Cache is turned off.
1	12	Cycles waiting for D-Cache fill data	Number of cycles waiting for the return of the critical word of D-Cache misses from the system bus. This event selector does not monitor accesses to I/O regions or accesses to cacheable regions when D-Cache is turned off.
1	13	Uncached fetch data access from bus	Number of accesses of uncached instruction data returning from the system bus. This event selector monitors accesses to I/O regions or accesses to cacheable regions when I-Cache is not configured or off.
1	14	Uncached load data access from bus	Number of accesses of uncached load data returning from the system bus. This event selector monitors accesses to I/O regions or accesses to cacheable regions when D-Cache is not configured or off.
1	15	Cycles waiting for uncached fetch data from bus	Number of cycles waiting for the instruction data to return from the system bus. This event selector monitors accesses to I/O regions or accesses to cacheable regions when I-Cache is not configured or off.
1	16	Cycles waiting for uncached load data from bus	Number of cycles waiting for the load data to return from the system bus. This event selector monitors accesses to I/O regions or accesses to cacheable regions when D-Cache is not configured or off.
1	23	Hardware prefetch bus access	This event counts the bus accesses generated by the hardware data prefetcher.

TYPE	SEL	事件名称	解释
2	0	Misprediction of conditional branches (direction)	Number of misprediction of committed conditional branches.
2	1	Misprediction of taken conditional branches (direction)	Number of misprediction of committed taken conditional branches.
2	2	Misprediction of targets of Return instructions	Number of misprediction of committed Return instruction.

表 4: Event Selectors

3.2.16 MHPMEVENT4 (0x324) (standard read/write)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																							SEL					TYPE			
N/A																							RW					RW			
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0

MHPMEVENT4 [31:0]

位域	名称	描述
8-4	SEL	参见事件选择器表
3-0	TYPE	参见事件选择器表

MHPMEVENT4 位域

3.2.17 MHPMEVENT5 (0x325) (standard read/write)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																							SEL					TYPE			
N/A																							RW					RW			
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0

MHPMEVENT5 [31:0]

位域	名称	描述
8-4	SEL	参见事件选择器表
3-0	TYPE	参见事件选择器表

MHPMEVENT5 位域

3.2.18 MHPMEVENT6 (0x326) (standard read/write)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																							SEL					TYPE			
N/A																							RW					RW			
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0

MHPMEVENT6 [31:0]

位域	名称	描述
8-4	SEL	参见事件选择器表
3-0	TYPE	参见事件选择器表

MHPMEVENT6 位域

3.2.19 MSCRATCH (0x340) (standard read/write)

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

MSCRATCH [31:0]

位域	名称	描述
31-0	MSCRATCH	暂存寄存器存储。

MSCRATCH 位域

3.2.20 MEPC (0x341) (standard read/write)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EPC																															RSVD
RW																															N/A
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	x

MEPC [31:0]

位域	名称	描述
31-1	EPC	异常程序计数器。

MEPC 位域

3.2.21 MCAUSE (0x342) (standard read/write)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
INTERRUPT	RSVD																			EXCEPTION_CODE											
RW	N/A																			RW											
0	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0

MCAUSE [31:0]

位域	名称	描述
31	INTERRUPT	中断
11-0	EXCEPTION_CODE	异常代码当中断为 1 时，该值表示： 0：用户软件中断 1：特权软件中断 3：机器软件中断 4：用户定时器中断 7：机器定时器中断 8：用户外部中断 11：机器外部中断 16：不精确的 ECC 错误中断（从端口访问、D-Cache 驱逐和非阻塞加载/存储）（M 模式） 17：总线读/写事务错误中断（M-mode） 18：性能监视器溢出中断（M-mode） 当中断位为 0 时，该值表示： 0：指令地址未对齐 1：指令访问错误 2：非法指令 3：断点 4：加载地址未对齐 5：负载访问故障 6：Store/AMO 地址未对齐 7：Store/AMO 访问故障 8：U-mode 环境调用 11：M 模式的环境调用 32：堆栈溢出异常 33：堆栈下溢异常 40-47：保留

MCAUSE 位域

3.2.22 MTVAL (0x343) (standard read/write)

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

MTVAL [31:0]

位域	名称	描述
31-0	MTVAL	软件 trap 处理的异常特定信息。

MTVAL 位域

3.2.23 MIP (0x344) (standard read/write)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD													PMOVI	BWEI	IMECCI	RSVD				MEIP	RSVD	SEIP	UEIP	MTIP	RSVD	STIP	UTIP	MSIP	RSVD	SSIP	USIP
N/A													RW	RW	RW	N/A				RW	N/A	RW	RW	RW	N/A	RW	RW	RW	N/A	RW	RW
x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	x	x	x	x	0	x	0	0	0	x	0	0	0	x	0	0

MIP [31:0]

位域	名称	描述
18	PMOVI	性能监视器溢出本地中断挂起位。 0：未挂起 1：挂起
17	BWEI	总线读/写事务错误本地中断挂起位。处理器可能会收到加载/存储指令或缓存写回的总线错误。 0：未挂起 1：挂起
16	IMECCI	不精确的 ECC 错误本地中断使能位。处理器可能会在从端口访问或缓存写回时收到不精确的 ECC 错误。 0：未挂起 1：挂起
11	MEIP	M 模式外部中断挂起位。 0：未挂起 1：挂起
9	SEIP	S 模式外部中断挂起位。 0：未挂起 1：挂起

位域	名称	描述
8	UEIP	U 模式外部中断挂起位。 0：未挂起 1：挂起
7	MTIP	M 模式定时器中断挂起位。 0：未挂起 1：挂起
5	STIP	S 模式定时器中断挂起位。 0：未挂起 1：挂起
4	UTIP	U 模式定时器中断挂起位 0：未挂起 1：挂起
3	MSIP	M 模式软件中断挂起位。 0：未挂起 1：挂起
1	SSIP	S 模式软件中断挂起位。 0：未挂起 1：挂起
0	USIP	U 模式软件中断挂起位。 0：未挂起 1：挂起

MIP 位域

3.2.24 PMPCFG0 (0x3A0) (standard read/write)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PMP3CFG								PMP2CFG								PMP1CFG								PMP0CFG							
RW								RW								RW								RW							
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

PMPCFG0 [31:0]

位域	名称	描述
31-24	PMP3CFG	见 PMPCFG 表
23-16	PMP2CFG	见 PMPCFG 表
15-8	PMP1CFG	见 PMPCFG 表
7-0	PMP0CFG	见 PMPCFG 表

PMPCFG0 位域

3.2.25 PMPCFG1 (0x3A1) (standard read/write)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PMP7CFG								PMP6CFG								PMP5CFG								PMP4CFG							
RW								RW								RW								RW							
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

PMPCFG1 [31:0]

位域	名称	描述
31-24	PMP7CFG	见 PMPCFG 表
23-16	PMP6CFG	见 PMPCFG 表
15-8	PMP5CFG	见 PMPCFG 表
7-0	PMP4CFG	见 PMPCFG 表

PMPCFG1 位域

3.2.26 PMPCFG2 (0x3A2) (standard read/write)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PMP11CFG								PMP10CFG								PMP9CFG								PMP8CFG							
RW								RW								RW								RW							
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

PMPCFG2 [31:0]

位域	名称	描述
31-24	PMP11CFG	见 PMPCFG 表
23-16	PMP10CFG	见 PMPCFG 表
15-8	PMP9CFG	见 PMPCFG 表
7-0	PMP8CFG	见 PMPCFG 表

PMPCFG2 位域

3.2.27 PMPCFG3 (0x3A3) (standard read/write)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PMP15CFG								PMP14CFG								PMP13CFG								PMP12CFG							
RW								RW								RW								RW							

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

PMPCFG3 [31:0]

位域	名称	描述
31-24	PMP15CFG	见 PMPCFG 表
23-16	PMP14CFG	见 PMPCFG 表
15-8	PMP13CFG	见 PMPCFG 表
7-0	PMP12CFG	见 PMPCFG 表

PMPCFG3 位域

7	6	5	4	3	2	1	0
L	RSVD		A		X	W	R
W1S*	N/A		RW		RW	RW	RW
0	x	x	0	0	0	0	0

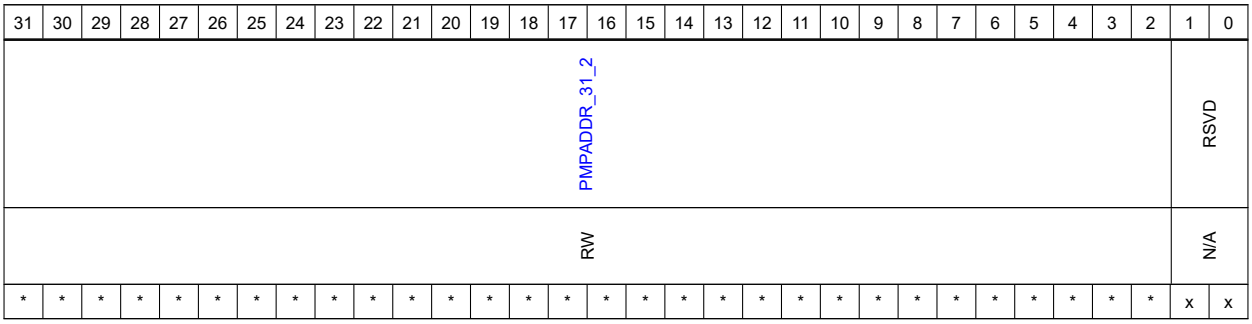
表 5: PMPCFG [7:0]

位域	名称	描述
7	L	Write lock and permission enforcement bit for Machine mode 0:Machine mode writes to PMP entry registers are allowed. R/W/X permissions apply to S and U modes. 1:For PMP entry i, writes to PMPiCFG and PMPADDRi are ignored. Additionally, if PMPiCFG.A is set to TOR, writes to pmpaddri-1 are ignored as well. As for permission enforcement, R/W/X permissions apply to all modes. This bit can only be cleared to 0 with a system reset.
4-3	A	Address matching mode. 0:OFF: Null region. 1:TOR: Top of range. For PMP entry 0, it matches any address A < pmpaddr0. For PMP entry i, it matches any address A such that pmpaddri > A >= pmpaddri-1. But the 4-byte range is not supported. 2:Reserved. 3:NAPOT: Naturally aligned power-of-2 region. This mode makes use of the low-order bits of the associated address register to encode the size of the range. The minimal size of NAPOT regions must be 8 bytes.
2	X	Instruction execution control. 0:Instruction execution is not allowed 1:Instruction execution is allowed.

位域	名称	描述
1	W	Write access control. 0:Write accesses are not allowed. 1:Write accesses are allowed.
0	R	Read access control. 0:Read accesses are not allowed. 1:Read accesses are allowed.

表 6: PMPCFG 位域

3.2.28 PMPADDR[PMPADDR0] (0x3B0 + 0x1 * n) (standard read/write)

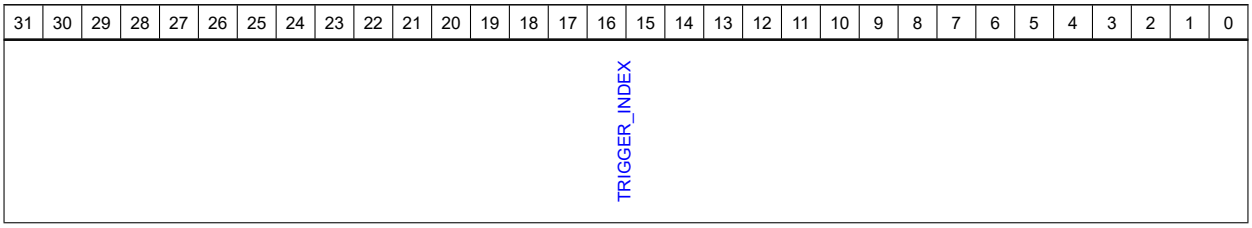


PMPADDR [31:0]

位域	名称	描述
31-2	PMPADDR_31_2	寄存器内容：匹配大小（字节） aaaa. . . aaa0 8 aaaa. . . aa01 16 aaaa. . . a011 32 . . . . . aa01. . . 1111 2^{XLEN} a011. . . 1111 2^{XLEN+1} 0111. . . 1111 2^{XLEN+2} 1111. . . 1111 2^{XLEN+3*1}

PMPADDR 位域

3.2.29 TSELECT (0x7A0) (standard read/write)



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

TSELECT [31:0]

位域	名称	描述
31-0	TRIGGER_INDEX	该寄存器确定哪个 trigger 可通过其他 trigger 寄存器访问。

TSELECT 位域

3.2.30 TDATA1 (0x7A1) (standard read/write)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TYPE				DMODE	DATA																										
RW				RW	RW																										
0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

TDATA1 [31:0]

位域	名称	描述
31-28	TYPE	表示 trigger 类型。 0：选择的 trigger 无效。 2：选择的 trigger 是地址/数据匹配 trigger。 3：选择的 trigger 是指令计数 trigger 4：选择的 trigger 为中断 trigger。 5：选择的 trigger 是异常 trigger。
27	DMODE	设置此字段以指示调试模式使用的 trigger。 0：Debug-mode 和 M-mode 都可以写当前选中的 trigger 寄存器。 1：只有调试模式可以写入当前选择的 trigger 寄存器。忽略来自 M 模式的写入。
26-0	DATA	特定 trigger 的数据

TDATA1 位域

3.2.31 MCONTROL (0x7A1) (standard read/write)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TYPE				DMODE	MASKMAX						RSVD					ACTION				CHAIN	MATCH				M	RSVD		U	EXECUTE	STORE	LOAD
RW				RW	RO						N/A					RW				RW	RW				RW	N/A		RW	RW	RW	RW
0	0	1	0	0	0	1	0	0	1	0	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	x	x	0	0	0	0

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

MCONTROL [31:0]

位域	名称	描述
31-28	TYPE	表示 trigger 类型。 0：选择的 trigger 无效。 2：选择的 trigger 是地址/数据匹配 trigger。
27	DMODE	设置此字段以指示调试模式使用 trigger。 0：Debug-mode 和 M-mode 都可以写当前选中的 trigger 寄存器 1：只有调试模式可以写入当前选择的 trigger 寄存器。忽略来自 M 模式的写入。
26-21	MASKMAX	表示硬件支持的最大自然对齐范围为 2^12 字节。
15-12	ACTION	设置此字段以选择此 trigger 匹配时会发生什么。 0：引发断点异常 1：进入调试模式。（仅当 DMODE 为 1 时才支持。）
11	CHAIN	设置此字段以启用 trigger 链。 0：当此 trigger 匹配时，执行配置的动作。 1：当这个 trigger 不匹配时，它阻止下一个索引的 trigger 匹配。 如果 trigger 数为 2，则该字段在触发器 1 上硬连线为 0 (tselect = 1)。 如果 trigger 数量为 4，则该字段是硬连线的 在 trigger 3 上变为 0 (tselect = 3)。 如果 trigger 数为 8，则该字段在 trigger 3 和 trigger 7 上硬连线为 0（tselect = 3 或 7）。
10-7	MATCH	设置该字段选择匹配方案。0：当值等于 tdata2 时匹配。1：当值的前 M 位与 tdata2 的前 M 位匹配时匹配。M 是 31 减去最低有效位的索引 tdata2 中包含 0。 2：当值大于（无符号）或等于 tdata2 时匹配。 3：当值小于（无符号）tdata2 时匹配
6	M	设置此字段以在 M 模式下启用此 trigger。
3	U	设置此字段以在 U 模式下启用此 trigger。
2	EXECUTE	设置此字段以启用此 trigger 来比较指令的虚拟地址。
1	STORE	设置此字段以启用此 trigger 来比较 store 的虚拟地址。
0	LOAD	设置此字段以启用此 trigger 以比较负载的虚拟地址。

MCONTROL 位域

3.2.32 ICOUNT (0x7A1) (standard read/write)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TYPE				DMODE	RSVD																COUNT	M	RSVD		U	ACTION						
RW				RW	N/A																RO	RW	N/A		RW	RW						
0	0	1	0	0	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	1	0	x	x	0	0	0	0	0	0	0

ICOUNT [31:0]

位域	名称	描述
31-28	TYPE	选择的 trigger 是指令计数 trigger。
27	DMODE	设置此字段以指示调试模式使用 trigger。 0：Debug-mode 和 M-mode 都可以写当前选中的 trigger 寄存器。 1：只有调试模式可以写入当前选择的 trigger 寄存器。忽略来自 M 模式的写入。
10	COUNT	此字段硬连线为 1 以支持单步执行
9	M	设置此字段以在 M 模式下启用此 trigger。
6	U	设置此字段以在 U 模式下启用此 trigger。
5-0	ACTION	设置此字段以选择此 trigger 匹配时会发生什么。 0：引发断点异常 1：进入调试模式。（仅当 DMODE 为 1 时才支持。）

ICOUNT 位域

3.2.33 ITRIGGER (0x7A1) (standard read/write)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TYPE				DMODE	RSVD																	M	RSVD		U	ACTION						
RW				RW	N/A																	RW	N/A		RW	RW						
0	0	1	0	0	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	x	x	0	0	0	0	0	0	0

ITRIGGER [31:0]

位域	名称	描述
31-28	TYPE	所选 trigger 为中断 trigger。
27	DMODE	设置此字段以指示调试模式使用 trigger。 0：Debug-mode 和 M-mode 都可以写当前选中的 trigger 寄存器。 1：只有调试模式可以写入当前选择的 trigger 寄存器。忽略来自 M 模式的写入。
9	M	设置此字段以在 M 模式下启用此 trigger。
6	U	设置此字段以在 U 模式下启用此 trigger。

位域	名称	描述
5-0	ACTION	设置此字段以选择此 trigger 匹配时会发生什么。 0：引发断点异常。 1：进入调试模式。（仅当 DMODE 为 1 时才支持。）

ITRIGGER 位域

3.2.34 ETRIGGER (0x7A1) (standard read/write)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TYPE				DMODE	RSVD																NMI	M	RSVD		U	ACTION						
RW				RW	N/A																RW	RW	N/A		RW	RW						
0	0	1	0	0	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	x	x	0	0	0	0	0	0	0

ETRIGGER [31:0]

位域	名称	描述
31-28	TYPE	选择的 trigger 是异常 trigger。
27	DMODE	设置此字段以指示调试模式使用 trigger。 0：Debug-mode 和 M-mode 都可以写当前选中的 trigger 寄存器。 1：只有调试模式可以写入当前选择的 trigger 寄存器。忽略来自 M 模式的写入。
10	NMI	设置此字段以在不可屏蔽中断中启用此 trigger，而不管 s、u 和 m 的值如何。
9	M	设置此字段以在 M 模式下启用此 trigger。
6	U	设置此字段以在 U 模式下启用此 trigger。
5-0	ACTION	设置此字段以选择此 trigger 匹配时会发生什么。 0：引发断点异常 1：进入调试模式。（仅当 DMODE 为 1 时才支持。）

ETRIGGER 位域

3.2.35 TDATA2 (0x7A2) (standard read/write)

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

TDATA2 [31:0]

位域	名称	描述
31-0	DATA	该寄存器提供对 tselect 寄存器选择的当前所选 trigger 寄存器的 tdata2 寄存器的访问，并且它保存特定于 trigger 的数据。

TDATA2 位域

3.2.36 TDATA3 (0x7A3) (standard read/write)

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

TDATA3 [31:0]

位域	名称	描述
31-0	DATA	该寄存器提供对 tselect 寄存器选择的当前所选触发寄存器的 tdata3 寄存器的访问，并且它保存特定于触发器的数据。

TDATA3 位域

3.2.37 TEXTRA (0x7A3) (standard read/write)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MVALUE						MSELECT	RSVD														SVALUE										SSELECT	
RW						RW	N/A														RW										RW	
0	0	0	0	0	0	0	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0

TEXTRA [31:0]

位域	名称	描述
31-26	MVALUE	与 MSELECT 一起使用的数据。
25	MSELECT	0：忽略 MVALUE。 1：此 trigger 仅在 mcontext 的低位等于 MVALUE 时匹配。
10-2	SVALUE	与 SSELECT 一起使用的数据。
1-0	SSELECT	0: 忽略 MVALUE 1：此 trigger 仅在 scontext 的低位等于 SVALUE 时匹配 2：此 trigger 仅在 satp.ASID 等于 SVALUE 时才匹配。

TEXTRA 位域

3.2.38 TINFO (0x7A4) (standard read/write)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																INFO															
N/A																RO															
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*

TINFO [31:0]

位域	名称	描述
15-0	INFO	tdata1 中每种可能类型的一位。位 N 对应于类型 N。如果设置了该位，那么当前选择的 trigger 支持类型。如果当前选择的 trigger 不存在，则该字段包含 1。 0：当该位被设置时，该 tselect 没有触发 1：保留并硬连线为 0。 2：设置该位时，选择的 trigger 支持地址/数据匹配 trigger 类型 3：当该位置位时，选择的 trigger 支持指令计数 trigger 类型。 4：当该位被设置时，选择的 trigger 支持中断 trigger 类型 5：当该位被设置时，选择的 trigger 支持异常 trigger 类型 15：当该位被设置时，选定的 trigger 存在（因此枚举不应终止），但当前不可用。 其他：保留以备将来使用。

TINFO 位域

3.2.39 TCONTROL (0x7A5) (standard read/write)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																								MPTE	RSVD			MTE	RSVD		
N/A																								RW	N/A			RW	N/A		
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	x	x	x	0	x	x	x

TCONTROL [31:0]

位域	名称	描述
7	MPTE	M 模式先前 trigger 启用字段。当陷入 M 模式时，MPTE 被设置为 MTE 的值。
3	MTE	M-mode trigger enable field。当陷入 M-mode 的异常时，MTE 设置为 0。当执行 MRET 指令时，MTE 设置为 MPTE 的值。 0：当 hart 处于 M 模式时，trigger 不匹配/trigger。 1：当 hart 处于 M 模式时，trigger 会匹配/开火。

TCONTROL 位域

3.2.40 MCONTEXT (0x7A8) (standard read/write)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																										MCONTEXT					
N/A																										RW					
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0

MCONTEXT [31:0]

位域	名称	描述
5-0	MCONTEXT	机器模式软件可以将上下文编号写入该寄存器，可用于设置仅在该特定上下文中 trigger 的 trigger。

MCONTEXT 位域

3.2.41 SCONTEXT (0x7AA) (standard read/write)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																											SCONTEXT					
N/A																											RW					
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0

SCONTEXT [31:0]

位域	名称	描述
8-0	SCONTEXT	机器模式软件可以将上下文编号写入该寄存器，可用于设置仅在该特定上下文中 trigger 的 trigger。

SCONTEXT 位域

3.2.42 DCSR (0x7B0) (debug-mode-only)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
XDEBUGVER				RSVD												EBREAKM	RSVD		EBREAKU	STEPIE	STOPCOUNT	STOPTIME	CAUSE			RSVD	MPRVEN	NMIP	STEP	PRV	
RO				N/A												RW	N/A		RW	RW	RW	RW	RO			N/A	RW	RO	RW	RW	
0	1	0	0	x	x	x	x	x	x	x	x	x	x	x	x	0	x	x	0	0	1	1	0	0	0	x	0	0	0	1	1

DCSR [31:0]

位域	名称	描述
31-28	XDEBUGVER	外部调试器的版本。0 表示不存在外部调试器，4 表示外部调试器符合 RISC-V External Debug Support (TD003) V0.13
15	EBREAKM	该位控制机器模式下 EBREAK 指令的行为 0：生成常规断点异常 1：进入调试模式
12	EBREAKU	该位控制 EBREAK 指令在用户/应用程序模式下的行为 0：生成常规断点异常 1：进入调试模式
11	STEPIE	该位控制单步执行时是否启用中断 0：单步执行时禁止中断 1：允许单步中断
10	STOPCOUNT	该位控制是否在调试模式下停止性能计数器。 0：在调试模式下不停止计数器 1：在调试模式下停止计数器
9	STOPTIME	此位控制定时器在调试模式下是否停止。如果处理器处于调试模式并且此位已设置，则处理器仅将其停止时间输出引脚驱动为 1。需要集成工作以使平台中的定时器观察此引脚是否真正停止他们。 0：在调试模式下不停止定时器 1：在调试模式下停止定时器
8-6	CAUSE	进入调试模式的原因。当有多种原因进入调试模式时，确定 CAUSE 值的优先级将是：触发模块 >EBREAK>halt-on-reset>halt 请求 > 单步。Halt 请求是发出的请求通过外部调试器 0：保留 1：EBREAK 2：trigger 模块 3：hart 请求 4：单步 5：Halt-on-reset 6-7：保留
4	MPRVEN	该位控制 mstatus.MPRV 在调试模式下是否生效。 0: mstatus 中的 MPRV 在调试模式下被忽略。 1：mstatus 中的 MPRV 在 Debug 模式下生效。
3	NMIP	设置此位后，hart 将有一个不可屏蔽中断 (NMI) 待处理。由于 NMI 可以指示硬件错误情况，因此一旦设置了此位，就可能无法进行可靠的调试。

位域	名称	描述
2	STEP	该位控制非 DebugMode 指令执行是否处于单步模式。设置时，hart 在单条指令执行后返回 Debug Mode。如果由于异常指令没有完成，hart 将立即进入 Debug 执行异常处理程序之前的模式，并设置了适当的异常寄存器。 0：单步模式关闭 1：单步模式开启
1-0	PRV	进入调试模式时 hart 运行的权限级别。外部调试器可以修改此值以在退出调试模式时更改 hart 的权限级别。 0：用户/应用 1：特权 2：预留 3：机器

DCSR 位域

3.2.43 DPC (0x7B1) (debug-mode-only)

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

DPC [31:0]

位域	名称	描述
31-0	DPC	调试程序计数器。位 0 硬连线为 0。

DPC 位域

3.2.44 DSCRATCH0 (0x7B2) (debug-mode-only)

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

DSCRATCH0 [31:0]

位域	名称	描述
31-0	DSCRATCH	保留供调试模块使用的暂存寄存器。

位域	名称	描述

DSCRATCH0 位域

3.2.45 DSCRATCH1 (0x7B3) (debug-mode-only)

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

DSCRATCH1 [31:0]

位域	名称	描述
31-0	DSCRATCH	保留供调试模块使用的暂存寄存器。

DSCRATCH1 位域

3.2.46 MILMB (0x7C0) (non-standard read/write)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IBPA																						RSVD						RWECC	ECCEN		IEN
RO																						N/A						RW	RW		RO
*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	x	x	x	x	x	x	0	0	0	1

MILMB [31:0]

位域	名称	描述
31-10	IBPA	ILM 的基本物理地址。它必须是 ILM 大小的整数倍
3	RWECC	控制 ILM RAM 的 ECC 代码的诊断访问。设置后，加载/存储到 ILM 将 ECC 代码读/写到 mecc_code 寄存器。可以设置此位以注入 ECC 错误以测试 ECC 处理程序。 0：禁用 ECC 代码的诊断访问 1：启用 ECC 代码的诊断访问
2-1	ECCEN	奇偶校验/ECC 使能控制： 0：禁用奇偶校验/ECC 1：预留 2：仅对不可纠正的奇偶校验/ECC 错误生成异常 3：对任何类型的奇偶校验/ECC 错误生成异常

位域	名称	描述
0	IEN	ILM 使能控制： 0：ILM 被禁用 1：ILM 已启用

MILMB 位域

3.2.47 MDLMB (0x7C1) (non-standard read/write)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DBPA																						RSVD						RWECC	ECCEN		DEN
RO																						N/A						RW	RW		RO
*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	x	x	x	x	x	x	0	0	0	1

MDLMB [31:0]

位域	名称	描述
31-10	DBPA	DLM 的基本物理地址。它必须是 DLM 大小的整数倍
3	RWECC	控制 DLM RAM 的 ECC 代码的诊断访问。设置后，加载/存储到 DLM 将 ECC 代码读/写到 mecc_code 寄存器。可以设置此位以注入 ECC 错误以测试 ECC 处理程序。 0：禁用 ECC 代码的诊断访问 1：启用 ECC 代码的诊断访问
2-1	ECCEN	奇偶校验/ECC 使能控制： 0：禁用奇偶校验/ECC 1：预留 2：仅对不可纠正的奇偶校验/ECC 错误生成异常 3：对任何类型的奇偶校验/ECC 错误生成异常
0	DEN	DLM 使能控制： 0：DLM 被禁用 1：DLM 已启用

MDLMB 位域

3.2.48 MECC_CODE (0x7C2) (non-standard read/write)

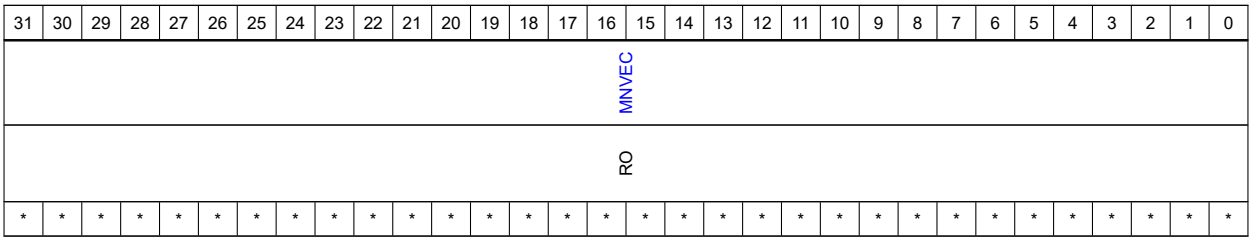
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD									INSN	RAMID				P	C	RSVD									CODE							
N/A									RO	RO				RO	RO	N/A									RW							
x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	1

MECC_CODE [31:0]

位域	名称	描述
22	INSN	指示奇偶校验/ECC 错误是由取指令还是数据访问引起的。 0：数据访问 1：指令获取
21-18	RAMID	导致奇偶校验/ECC 错误的 RAM 的 ID。该位在奇偶校验/ECC 错误异常时更新。 0-1: 保留 2：I-Cache 的 Tag RAM 3：I-Cache 的数据 RAM 4：D-Cache 的 Tag RAM 5：D-Cache 的数据 RAM 6：TLB 的标签 RAM 7：TLB 的数据 RAM 8: ILM 9: DLM 10-15：保留
17	P	精确错误。此位在奇偶校验/ECC 错误异常时更新。 0：不精确的错误 1：精确误差
16	C	可纠正错误。此位在奇偶校验/ECC 错误异常时更新。 0：不可纠正的错误 1：可纠正错误
6-0	CODE	该字段记录 ECC 错误异常的 ECC 值。当启用 ECC 代码的诊断访问时（milmb.RWECC 或 mdlmb.RWECC 为 1），此字段还用于读/写 ECC 代码。

MECC_CODE 位域

3.2.49 MNVEC (0x7C3) (non-standard read/write)



MNVEC [31:0]

位域	名称	描述
31-0	MNVEC	NMI 处理程序的基址。它的值是 reset_vector 的零扩展值。

MNVEC 位域

3.2.50 MXSTATUS (0x7C4) (non-standard read/write)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																										PDME	DME	PIME	IME	PPFT_EN	PFT_EN
N/A																										RW	RW	RW	RW	RW	RW
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0

MXSTATUS [31:0]

位域	名称	描述
5	PDME	用于在进入 trap 时保存之前的 DME 状态。如果不支持数据缓存和数据本地内存，则此字段硬连线为 0。
4	DME	机器数据错误标志。它表示数据缓存或数据本地内存 (DLM) 发生异常。设置此位时，加载/存储访问将绕过 D-Cache。处理完机器错误后，异常处理程序应清除该位。
3	PIME	用于在进入中断时保存以前的 IME 状态。如果不支持指令缓存和 ilm，则为 0
2	IME	指示计算机错误标志。它表示指令缓存发生异常或指令本地内存异常（ILM）。当为 1 时所有取指令将绕过指令缓存。异常处理程序应在计算机错误出现后清除此位处理。
1	PPFT_EN	当 mcause 是不精确异常（以中断的形式）时，PM 字段记录了导致不精确异常的指令的特权模式。PM 字段编码 定义如下： 0：用户模式 1：特权模式 2：保留 3：机器模式
0	PFT_EN	启用性能节流。启用节流后，处理器将在 mpft_ctl.T_LEVEL 中指定的性能级别执行指令。进入 trap 时： PPFT_EN <= PFT_EN; PFT_EN <= mpft_ctl.FAST_INT ? 0 :PFT_EN; 在执行 MRET 指令时： PFT_EN <= PPFT_EN; 如果不支持 PowerBrake 功能，则此字段硬连线为 0。

MXSTATUS 位域

3.2.51 MPFT_CTL (0x7C5) (non-standard read/write)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																							FAST_INT	T_LEVEL				RSVD			
N/A																							RW	RW				N/A			
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	x	x	x	x

MPFT_CTL [31:0]

位域	名称	描述
8	FAST_INT	快速中断响应。如果设置了这个字段，当处理器进入中断处理程序时，mxstatus.PFT_EN 将被自动清除。
7-4	T_LEVEL	节流级别。处理器在节流级别 0 时性能最高，在节流级别最低时节流级别 15 的性能。 0：Level 0（最高性能） 1-14:1-14 级 15：Level 15（最低性能）

MPFT_CTL 位域

3.2.52 MHSP_CTL (0x7C6) (non-standard read/write)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																										M	S	U	SCHM	UDF_EN	OVF_EN
N/A																										RW	RW	RW	RW	RW	RW
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0

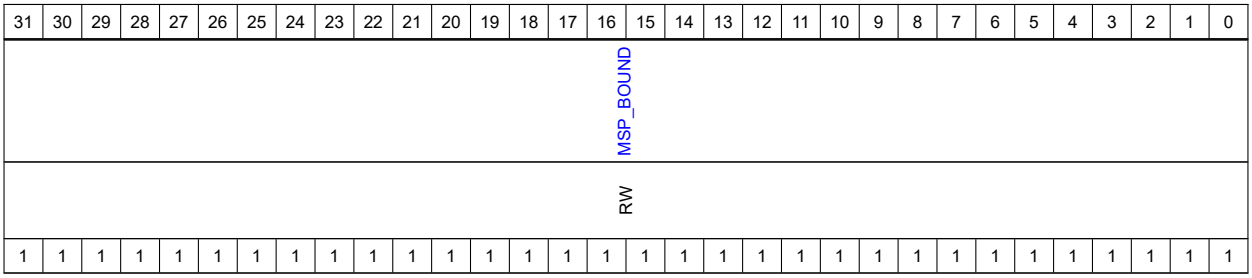
MHSP_CTL [31:0]

位域	名称	描述
5	M	在机器模式下启用 SP 保护和记录机制 0：该机制在机器模式下被禁用。 1：该机制在机器模式下启用。
4	S	在特权模式下启用 SP 保护和记录机制 0：特权模式下该机制被禁用 1：该机制在特权模式下启用
3	U	在用户模式下启用 SP 保护和记录机制 0：该机制在用户模式下被禁用 1：该机制在用户模式下启用。
2	SCHM	选择堆栈保护和记录机制的运行方案 0：堆栈上溢/下溢检测 1：栈顶记录

位域	名称	描述
1	UDF_EN	堆栈下溢保护机制使能位。当发生堆栈保护（上溢或下溢）异常时，该位将被硬件自动清零。 0：堆栈下溢保护禁用 1：堆栈下溢保护使能。
0	OVF_EN	堆栈溢出保护和记录机制使能位。当发生堆栈保护（上溢或下溢）异常时，该位将被硬件自动清零。 0：禁用堆栈溢出保护和记录机制。 1：启用堆栈溢出保护和记录机制。

MHSP_CTL 位域

3.2.53 MSP_BOUND (0x7C7) (non-standard read/write)

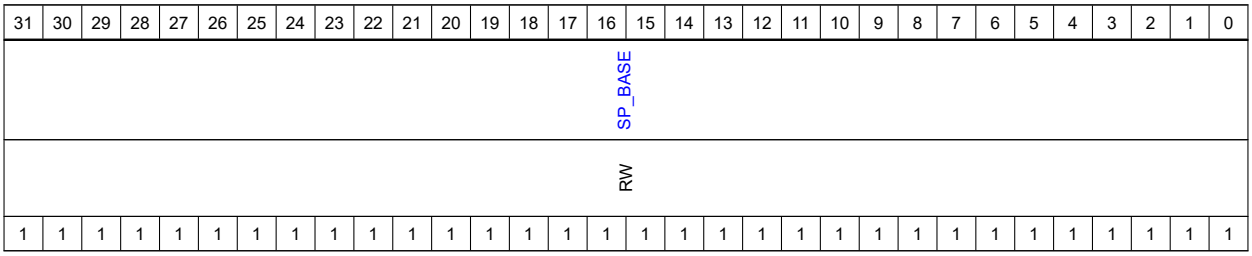


MSP_BOUND [31:0]

位域	名称	描述
31-0	MSP_BOUND	机器 SP 绑定

MSP_BOUND 位域

3.2.54 MSP_BASE (0x7C8) (non-standard read/write)



MSP_BASE [31:0]

位域	名称	描述
31-0	SP_BASE	机器 SP base

MSP_BASE 位域

3.2.55 MDCAUSE (0x7C9) (non-standard read/write)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																													MDCAUSE		
N/A																													RW		
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0

MDCAUSE [31:0]

位域	名称	描述
2-0	MDCAUSE	该寄存器进一步消除了 mcause 寄存器中记录的陷阱的原因。 精确异常的 MDCAUSE 值： 当 mcause == 1（指令访问错误）时： 0：保留；1：ECC/奇偶校验错误；2：PMP 指令访问冲突；3：总线错误； 当 mcause == 2（非法指令）时： 0：请解析 mtval CSR；1：FP 禁用异常；2：ACE 禁用异常 当 mcause == 5 (Load access fault): 0：保留；1：ECC/奇偶校验错误；2：PMP 加载访问冲突；3：总线错误；4：地址错位；6：PMA 属性不一致； 当 mcause == 7（存储访问错误）时： 0：保留；1：ECC/奇偶校验错误；2：PMP 加载访问冲突；3：总线错误；4：地址错位；6：PMA 属性不一致；

MDCAUSE 位域

3.2.56 MCACHE_CTL (0x7CA) (non-standard read/write)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																				IC_FIRST_WORD	RSVD		CCTL_SUEN	DC_RWECC	IC_RWECC	DC_ECCEN		IC_ECCEN		DC_EN	IC_EN
N/A																				RO	N/A		RW	RW	RW	RW		RW		RW	RW
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	1	x	x	0	0	0	0	0	0	0	0	0

MCACHE_CTL [31:0]

位域	名称	描述
11	IC_FIRST_WORD	缓存未命中分配填充策略 0：缓存行数据先返回关键（双）字 1：缓存线数据先返回最低地址（双）字”

位域	名称	描述
8	CCTL_SUEN	为特权模式和用户模式软件启用位以访问 ucctlbeginaddr 和 ucctlcommand CSR 0：在 S/U 模式下禁止 ucctlbeginaddr 和 ucctlcommand 访问 1：在 S/U 模式下启用 ucctlbeginaddr 和 ucctlcommand 访问
7	DC_RWECC	控制数据缓存 RAM 的 ECC 代码的诊断访问。设置它以启用 CCTL 操作以访问 ECC 代码。此位可以设置为注入 ECC 错误以测试 ECC 处理程序 0：禁用 ECC 代码的诊断访问 1：启用 ECC 代码的诊断访问
6	IC_RWECC	控制指令缓存 RAM 的 ECC 代码的诊断访问。设置它以启用 CCTL 操作以访问 ECC 代码。可以设置该位以注入 ECC 错误以测试 ECC 处理程序。 0：禁用 ECC 代码的诊断访问 1：启用 ECC 代码的诊断访问
5-4	DC_ECCEN	数据缓存奇偶校验/ECC 错误检查启用控制。 0：禁用奇偶校验/ECC 1：预留 2：仅对不可纠正的奇偶校验/ECC 错误生成异常 3：对任何类型的奇偶校验/ECC 错误生成异常
3-2	IC_ECCEN	指令缓存奇偶校验/ECc 错误检查启用控制 0：禁用奇偶校验/ECC 1：预留 2：仅对不可纠正的奇偶校验/ECC 错误生成异常 3：对任何类型的奇偶校验/ECC 错误生成异常
1	DC_EN	控制是否启用数据缓存。 0:D-Cache 被禁用 1:D-Cache 已启用
0	IC_EN	控制是否启用指令缓存。 0：I-Cache 被禁用 1：I-Cache 已启用

MCACHE_CTL 位域

3.2.57 MCCTLBEGINADDR (0x7CB) (non-standard read/write)

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

MCCTLBEGINADDR [31:0]

位域	名称	描述
31-0	VA	该寄存器保存 CCTL 操作所需的地址信息

MCCTLBEGINADDR 位域

3.2.58 MCCTLCOMMAND (0x7CC) (non-standard read/write)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																											VA				
N/A																											RW				
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0

MCCTLCOMMAND [31:0]

位域	名称	描述
4-0	VA	参见 CCTL 命令定义表

MCCTLCOMMAND 位域

值 (DEC)	值 (BIN)	命令	类型
0	0b00_000	L1D_VA_INVAL	VA
1	0b00_001	L1D_VA_WB	VA
2	0b00_010	L1D_VA_WBINVAL	VA
3	0b00_011	L1D_VA_LOCK	VA
4	0b00_100	L1D_VA_UNLOCK	VA
6	0b00_110	L1D_WBINVAL_ALL	-
7	0b00_111	L1D_WB_ALL	-
8	0b01_000	L1I_VA_INVAL	VA
11	0b01_011	L1I_VA_LOCK	VA
12	0b01_100	L1I_VA_UNLOCK	VA
16	0b10_000	L1D_IX_INVAL	Index
17	0b10_001	L1D_IX_WB	Index
18	0b10_010	L1D_IX_WBINVAL	Index
19	0b10_011	L1D_IX_RTAG	Index
20	0b10_100	L1D_IX_RDATA	Index
21	0b10_101	L1D_IX_WTAG	Index
22	0b10_110	L1D_IX_WDATA	Index
23	0b10_111	L1D_INVAL_ALL	-
24	0b11_000	L1I_IX_INVAL	Index
27	0b11_011	L1I_IX_RTAG	Index
28	0b11_100	L1I_IX_RDATA	Index
29	0b11_101	L1I_IX_WTAG	Index

值 (DEC)	值 (BIN)	命令	类型
30	0b11_110	L1I_IX_WDATA	Index

表 7: CCTL Command Definition

3.2.59 MCCTLDATA (0x7CD) (non-standard read/write)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																											VA				
N/A																											RW				
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0

MCCTLDATA [31:0]

位域	名称	描述
4-0	VA	查看访问 mcctldata 表的 CCTL 命令

MCCTLDATA 位域

值 (DEC)	值 (BIN)	命令	类型
3	0b00_011	L1D_VA_LOCK	VA
11	0b01_011	L1I_VA_LOCK	VA
19	0b10_011	L1D_IX_RTAG	Index
20	0b10_100	L1D_IX_RDATA	Index
21	0b10_101	L1D_IX_WTAG	Index
22	0b10_110	L1D_IX_WDATA	Index
27	0b11_011	L1I_IX_RTAG	Index
28	0b11_100	L1I_IX_RDATA	Index
29	0b11_101	L1I_IX_WTAG	Index
30	0b11_110	L1I_IX_WDATA	Index

表 8: CCTL Commands Which Access mcctldata

3.2.60 MCOUNTERWEN (0x7CE) (non-standard read/write)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																									HPM6	HPM5	HPM4	HPM3	IR	RSVD	CY
N/A																									RW	RW	RW	RW	RW	N/A	RW
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	x	0

MCOUNTERWEN [31:0]

位域	名称	描述
6	HPM6	见寄存器说明
5	HPM5	见寄存器说明
4	HPM4	见寄存器说明
3	HPM3	见寄存器说明
2	IR	见寄存器说明
0	CY	见寄存器说明

MCOUNTERWEN 位域

3.2.61 MCOUNTERINTEN (0x7CF) (non-standard read/write)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																									HPM6	HPM5	HPM4	HPM3	IR	RSVD	CY
N/A																									RW	RW	RW	RW	RW	N/A	RW
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	x	0

MCOUNTERINTEN [31:0]

位域	名称	描述
6	HPM6	见寄存器说明
5	HPM5	见寄存器说明
4	HPM4	见寄存器说明
3	HPM3	见寄存器说明
2	IR	见寄存器说明
0	CY	见寄存器说明

MCOUNTERINTEN 位域

3.2.62 MMISC_CTL (0x7D0) (non-standard read/write)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																									MSA_UNA	RSVD		BRPE	RVCOMPM	VEC_PLIC	RSVD
N/A																									RW	N/A		RW	RW	RW	N/A
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	1	x	x	1	0	0	x

MMISC_CTL [31:0]

位域	名称	描述
6	MSA_UNA	该字段控制加载/存储指令是否可以访问未对齐的内存位置而不会产生地址未对齐的异常。 支持指令：LW/LH/LHU/SW/SH 0：未对齐访问产生地址未对齐异常。 1：未对齐访问产生地址未对齐异常。
3	BRPE	分支预测使能位。该位控制所有分支预测结构。 0：禁用 1：启用 如果不支持分支预测结构，则该位硬连线为 0。
2	RVCOMPM	RISC-V 兼容模式使能位。如果打开兼容模式，所有特定指令都成为保留指令 0：禁用 1：启用
1	VEC_PLIC	选择 PLIC 的操作模式： 0：普通模式 1：矢量模式 请注意，该位和 NCEPLIC100 中功能使能寄存器的向量模式使能位 (VECTORED) 都应打开，以便向量中断支持正常工作。如果不支持向量 PLIC 功能，则该位硬连线为 0。

MMISC_CTL 位域

3.2.63 MCOUNTERMASK_M (0x7D1) (non-standard read/write)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																									HPM6	HPM5	HPM4	HPM3	IR	RSVD	CY
N/A																									RW	RW	RW	RW	RW	N/A	RW
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	x	0

MCOUNTERMASK_M [31:0]

位域	名称	描述
6	HPM6	见寄存器说明
5	HPM5	见寄存器说明
4	HPM4	见寄存器说明
3	HPM3	见寄存器说明
2	IR	见寄存器说明
0	CY	见寄存器说明

MCOUNTERMASK_M 位域

3.2.64 MCOUNTERMASK_S (0x7D2) (non-standard read/write)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																									HPM6	HPM5	HPM4	HPM3	IR	RSVD	CY
N/A																									RW	RW	RW	RW	RW	N/A	RW
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	x	0

MCOUNTERMASK_S [31:0]

位域	名称	描述
6	HPM6	见寄存器说明
5	HPM5	见寄存器说明
4	HPM4	见寄存器说明
3	HPM3	见寄存器说明
2	IR	见寄存器说明
0	CY	见寄存器说明

MCOUNTERMASK_S 位域

3.2.65 MCOUNTERMASK_U (0x7D3) (non-standard read/write)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																									HPM6	HPM5	HPM4	HPM3	IR	RSVD	CY
N/A																									RW	RW	RW	RW	RW	N/A	RW
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	x	0

MCOUNTERMASK_U [31:0]

位域	名称	描述
6	HPM6	见寄存器说明
5	HPM5	见寄存器说明
4	HPM4	见寄存器说明
3	HPM3	见寄存器说明
2	IR	见寄存器说明
0	CY	见寄存器说明

MCOUNTERMASK_U 位域

3.2.66 MCOUNTEROVF (0x7D4) (non-standard read/write)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																									HPM6	HPM5	HPM4	HPM3	IR	RSVD	CY
N/A																									RW	RW	RW	RW	RW	N/A	RW
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	x	0

MCOUNTEROVF [31:0]

位域	名称	描述
6	HPM6	见寄存器说明
5	HPM5	见寄存器说明
4	HPM4	见寄存器说明
3	HPM3	见寄存器说明
2	IR	见寄存器说明
0	CY	见寄存器说明

MCOUNTEROVF 位域

3.2.67 DEXC2DBG (0x7E0) (non-standard read/write)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD												PMOV	RSVD			BWE	SLPECC	ACE	HSP	MEC	RSVD		UEC	SAF	SAM	LAF	LAM	NMI	II	IAF	IAM
N/A												RW	N/A			RW	RW	RW	RW	RW	N/A		RW	RW	RW	RW	RW	RW	RW	RW	RW
x	x	x	x	x	x	x	x	x	x	x	x	0	x	x	x	0	0	0	0	0	x	x	0	0	0	0	0	0	0	0	0

DEXC2DBG [31:0]

位域	名称	描述
19	PMOV	指示是否重定向性能计数器溢出中断进入调试模式 0：不重定向 1：重定向
15	BWE	指示是否将 Bus-write Transaction Error 本地中断重定向到进入调试模式 0：不重定向 1：重定向
14	SLPECC	指示是否将本地内存从端口 ECC Error 本地中断重定向到 Debug Mode 0：不重定向 1：重定向

位域	名称	描述
13	ACE	指示 ACE 相关的异常是否被重定向进入调试模式。这个位只有在 mmsc_cfg.ACE 被设置时才存在 0：不重定向 1：重定向
12	HSP	指示堆栈保护异常是否被重定向以进入调试模式。仅当设置 mmsc_cfg.HSP 时才存在此位。 0：不重定向 1：重定向
11	MEC	指示 M-mode 环境调用异常是否重定向进入调试模式 0：不重定向 1：重定向
8	UEC	指示是否将 U 模式环境调用异常重定向到进入调试模式。 0：不重定向 1：重定向
7	SAF	指示是否将 Store Access Fault 异常重定向到进入调试模式。 0：不重定向 1：重定向
6	SAM	指示是否将 Store Access Misaligned 异常重定向到进入调试模式。 0：不重定向 1：重定向
5	LAF	指示负载访问故障异常是否被重定向到进入调试模式。 0：不重定向 1：重定向
4	LAM	指示是否将 Load Access Misaligned 异常重定向到进入调试模式 0：不重定向 1：重定向
3	NMI	表示不可屏蔽中断 0：不重定向 1：重定向
2	II	指示非法指令异常是否被重定向以进入调试模式。 0：不重定向 1：重定向
1	IAF	指示指令访问错误异常是否重定向进入调试模式 0：不重定向 1：重定向
0	IAM	指示指令访问未对齐异常是否被重定向以进入调试模式。 0：不重定向 1：重定向

DEXC2DBG 位域

3.2.68 DDCAUSE (0x7E1) (non-standard read/write)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																SUBTYPE								MAINTYPE							
N/A																RO								RO							
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

DDCAUSE [31:0]

位域	名称	描述
15-8	SUBTYPE	主要类型的子类型。 下表列出了 DCSR.CAUSE==1 和 DDCAUSE.MAINTYPE==3 的子类型。 0：非法指令 1：特权指令 2：不存在的 CSR 3：特权 CSR 访问 4：只读 CSR 更新
7-0	MAINTYPE	重定向到调试模式的原因。 0：软件断点（EBREAK） 1：指令访问未对齐（IAM） 2：指令访问错误（IAF） 3：非法指令（II） 4：不可屏蔽中断（NMI） 5：负载访问未对齐（LAM） 6：负载访问故障（LAF） 7：存储访问未对齐（SAM） 8：存储访问故障（SAF） 9：U-mode 环境调用（UEC） 12：M 模式环境调用（MEC） 16：不精确的 ECC 错误 17; 总线写事务错误 18：性能计数器溢出 19 -31：保留 32：堆栈溢出异常 33：堆栈下溢异常 34: ACE 禁用异常 35-39：保留 40-47:ACE 异常 >=48：保留

位域	名称	描述

DDCAUSE 位域

3.2.69 UITB (0x800) (non-standard read/write)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDR																														RSVD	HW
RW																														N/A	RO
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	x	0

UITB [31:0]

位域	名称	描述
31-2	ADDR	CoDense 指令表的基地址。如果 uitb.HW == 1，则保留此字段。
0	HW	该位指定 CoDense 指令表是否是硬连线的。 0：指令表位于内存中。uitb.ADDR 应该在使用 CoDense 指令之前初始化为指向表。 1：指令表是硬连线的。在使用 CoDense 指令之前不需要初始化 uitb.ADDR。

UITB 位域

3.2.70 UCODE (0x801) (non-standard read/write)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																															OV
N/A																															RW
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0

UCODE [31:0]

位域	名称	描述
0	OV	溢出标志。由 DSP 指令设置，结果饱和。 0：不产生饱和结果 1：产生饱和结果

UCODE 位域

3.2.71 UDCAUSE (0x809) (non-standard read/write)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																													UDCAUSE		
N/A																													RW		
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0

UDCAUSE [31:0]

位域	名称	描述
2-0	UDCAUSE	该寄存器进一步消除了 ucause 寄存器中记录的异常的原因。有关详细信息，请参阅下面的列表。 UDCAUSE 用于精确异常的值： 当 ucause == 1（指令访问错误） 0：保留 1：ECC/奇偶校验错误 2：PMP 指令访问冲突 3：总线错误 4：当 ucause == 2 (非法指令) 0：请解析 utval CSR 1：FP 禁用异常 2：ACE 禁用异常 当 ucause == 5 (Load access fault) 0：保留 1：ECC/奇偶校验错误 2：PMP 加载访问冲突 3：总线错误 4：地址错位 当 ucause == 7 (Store access fault) 0：保留 1：ECC/奇偶校验错误 2：PMP 存储访问冲突 3：总线错误 4：地址错位

UDCAUSE 位域

3.2.72 UCCTLBEGINADDR (0x80B) (non-standard read/write)

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

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

UCCTLBEGINADDR [31:0]

位域	名称	描述
31-0	VA	它是 mcctlbeginaddr 寄存器的别名，只有当 mcache_ctl.CCTL_SUEN 为 1 时，特权模式和用户模式软件才能访问它。否则将触发非法指令异常。

UCCTLBEGINADDR 位域

3.2.73 UCCTLCOMMAND (0x80C) (non-standard read/write)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																											VA					
N/A																											RW					
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0

UCCTLCOMMAND [31:0]

位域	名称	描述
4-0	VA	参见用户 CCTL 命令定义表

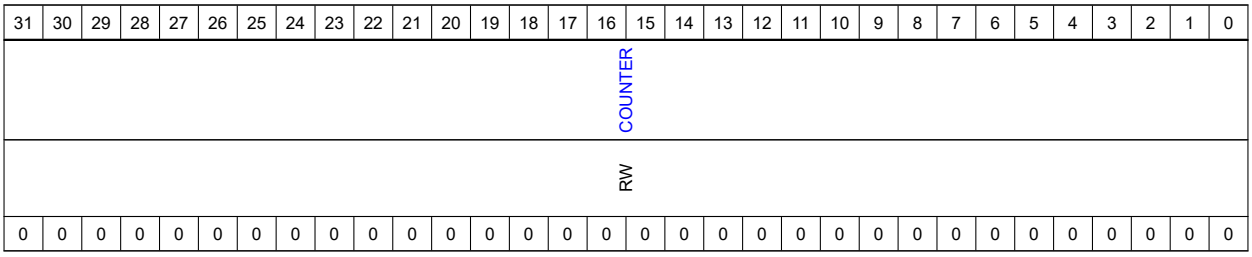
UCCTLCOMMAND 位域

值 (DEC)	值 (BIN)	命令	类型	用户模式允许
0	0b00_000	L1D_VA_INVAL	VA	1
1	0b00_001	L1D_VA_WB	VA	1
2	0b00_010	L1D_VA_WBINVAL	VA	1
3	0b00_011	L1D_VA_LOCK	VA	0
4	0b00_100	L1D_VA_UNLOCK	VA	0
6	0b00_110	L1D_WBINVAL_ALL	-	0
7	0b00_111	L1D_WB_ALL	-	0
8	0b01_000	L1I_VA_INVAL	VA	1
11	0b01_011	L1I_VA_LOCK	VA	0
12	0b01_100	L1I_VA_UNLOCK	VA	0
16	0b10_000	L1D_IX_INVAL	Index	0
17	0b10_001	L1D_IX_WB	Index	0
18	0b10_010	L1D_IX_WBINVAL	Index	0
19	0b10_011	L1D_IX_RTAG	Index	0
20	0b10_100	L1D_IX_RDATA	Index	0
21	0b10_101	L1D_IX_WTAG	Index	0
22	0b10_110	L1D_IX_WDATA	Index	0

值 (DEC)	值 (BIN)	命令	类型	用户模式允许
23	0b10_111	L1D_INVAL_ALL	-	0
24	0b11_000	L1I_IX_INVAL	Index	0
27	0b11_011	L1I_IX_RTAG	Index	0
28	0b11_100	L1I_IX_RDATA	Index	0
29	0b11_101	L1I_IX_WTAG	Index	0
30	0b11_110	L1I_IX_WDATA	Index	0

表 9: CCTL Command Definition

3.2.74 MCYCLE (0xB00) (standard read/write)

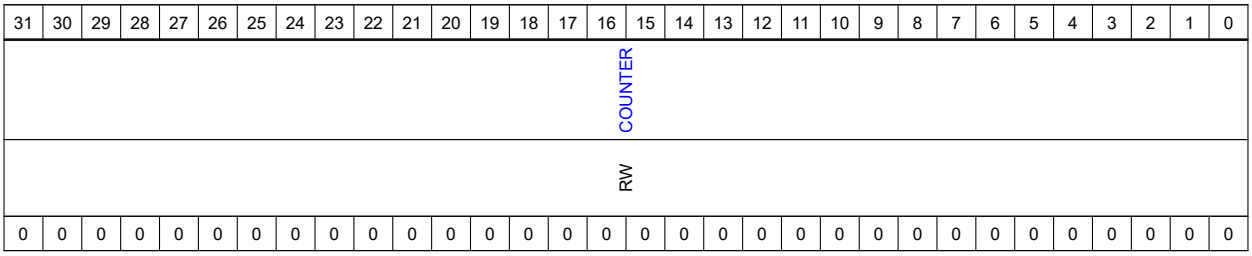


MCYCLE [31:0]

位域	名称	描述
31-0	COUNTER	机器周期计数器的低 32 位

MCYCLE 位域

3.2.75 MINSTRET (0xB02) (standard read/write)

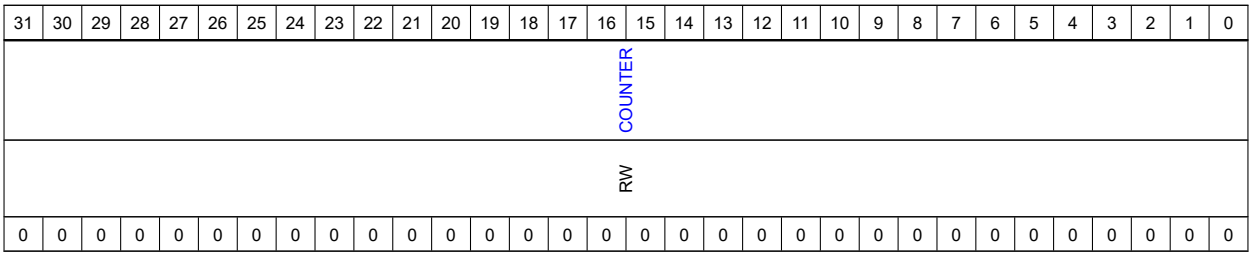


MINSTRET [31:0]

位域	名称	描述
31-0	COUNTER	机器指令引退计数器的低 32 位

MINSTRET 位域

3.2.76 MHPMCOUNTER3 (0xB03) (standard read/write)

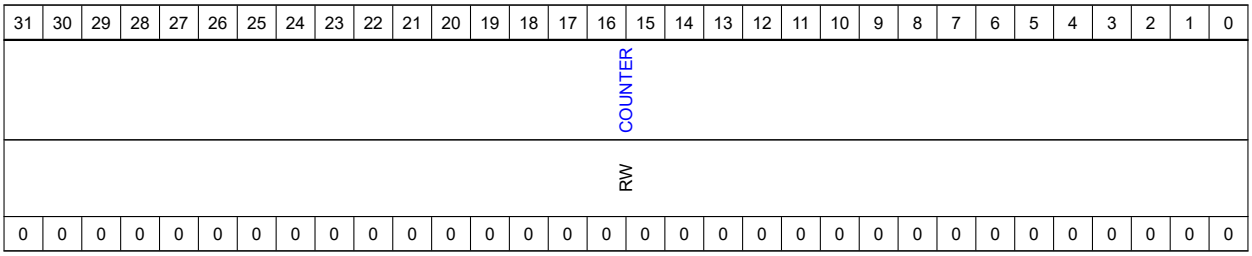


MHPMCOUNTER3 [31:0]

位域	名称	描述
31-0	COUNTER	计算 mhpmevent3 选择的事件数

MHPMCOUNTER3 位域

3.2.77 MHPMCOUNTER4 (0xB04) (standard read/write)

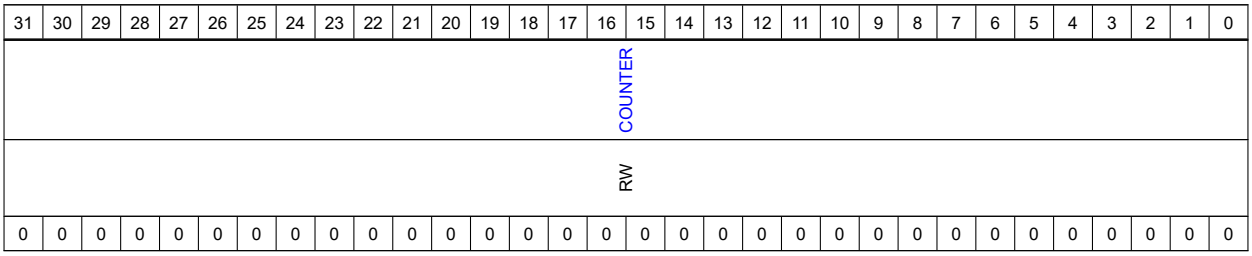


MHPMCOUNTER4 [31:0]

位域	名称	描述
31-0	COUNTER	计算 mhpmevent4 选择的事件数

MHPMCOUNTER4 位域

3.2.78 MHPMCOUNTER5 (0xB05) (standard read/write)

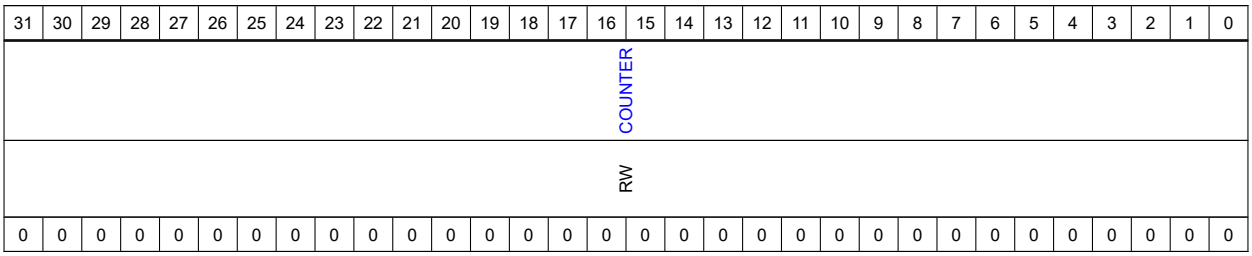


MHPMCOUNTER5 [31:0]

位域	名称	描述
31-0	COUNTER	计算 mhpmevent5 选择的事件数

MHPMCOUNTER5 位域

3.2.79 MHPMCOUNTER6 (0xB06) (standard read/write)

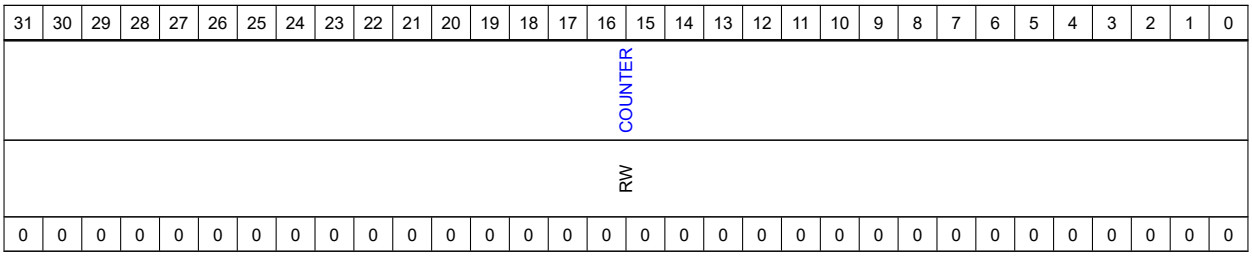


MHPMCOUNTER6 [31:0]

位域	名称	描述
31-0	COUNTER	计算 mhpmevent6 选择的事件数

MHPMCOUNTER6 位域

3.2.80 MCYCLEH (0xB80) (standard read/write)

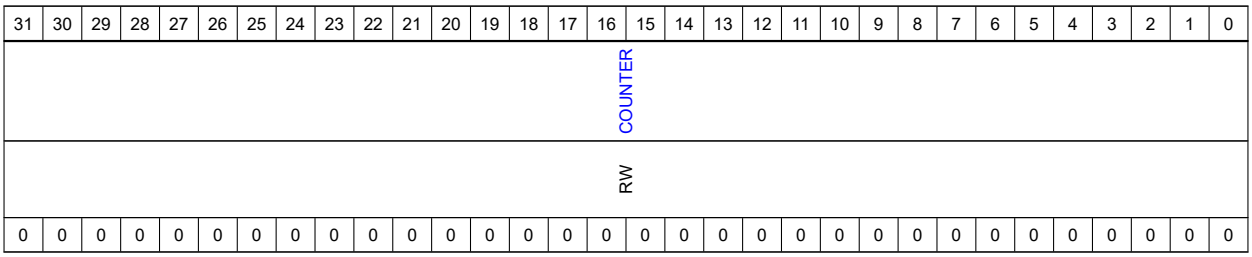


MCYCLEH [31:0]

位域	名称	描述
31-0	COUNTER	机器周期计数器的高 32 位

MCYCLEH 位域

3.2.81 MINSTRETH (0xB82) (standard read/write)

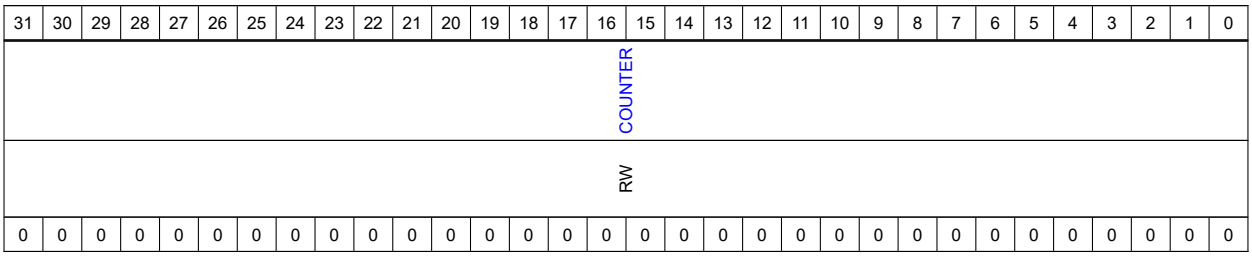


MINSTRETH [31:0]

位域	名称	描述
31-0	COUNTER	机器指令引退计数器的高 32 位

MINSTRETH 位域

3.2.82 MHPMCOUNTER3H (0xB83) (standard read/write)

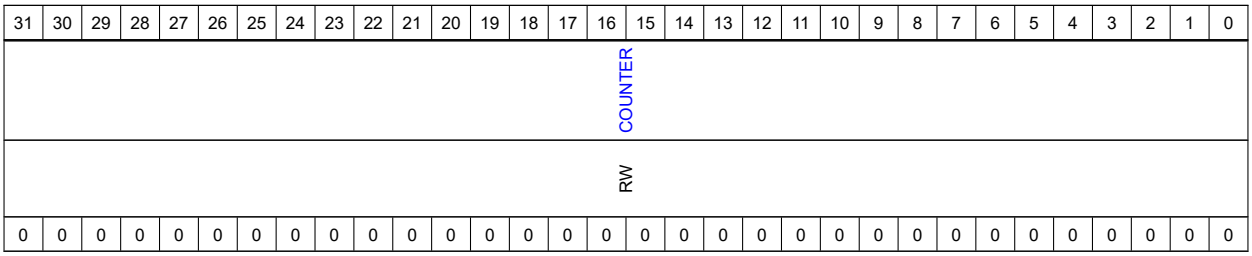


MHPMCOUNTER3H [31:0]

位域	名称	描述
31-0	COUNTER	计算 mhpmevent3 选择的事件数

MHPMCOUNTER3H 位域

3.2.83 MHPMCOUNTER4H (0xB84) (standard read/write)

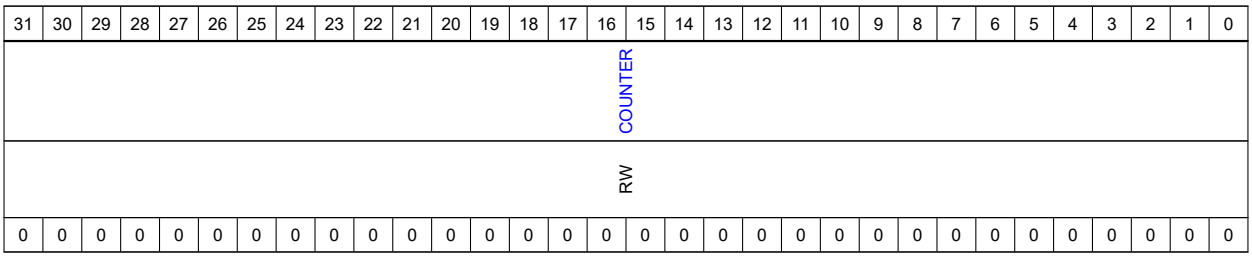


MHPMCOUNTER4H [31:0]

位域	名称	描述
31-0	COUNTER	计算 mhpmevent4 选择的事件数

MHPMCOUNTER4H 位域

3.2.84 MHPMCOUNTER5H (0xB85) (standard read/write)

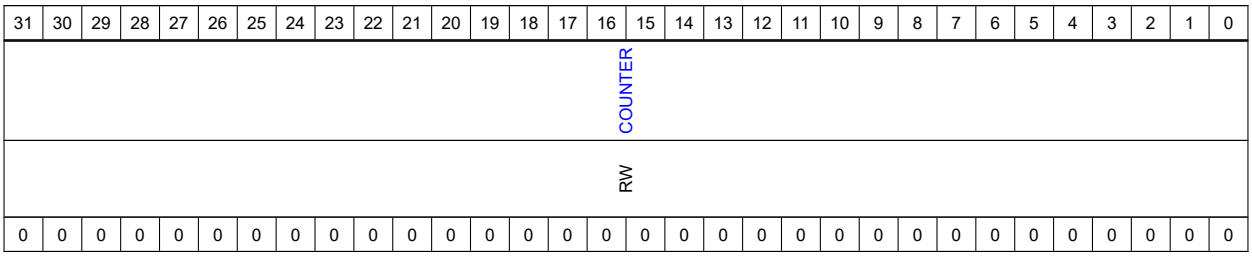


MHPMCOUNTER5H [31:0]

位域	名称	描述
31-0	COUNTER	计算 mhpmevent5 选择的事件数

MHPMCOUNTER5H 位域

3.2.85 MHPMCOUNTER6H (0xB86) (standard read/write)

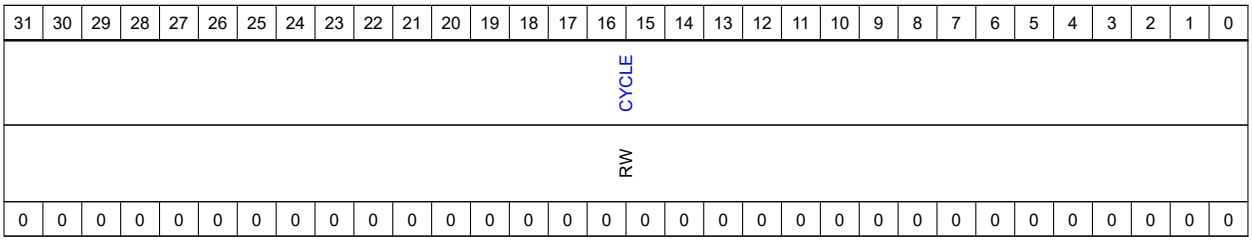


MHPMCOUNTER6H [31:0]

位域	名称	描述
31-0	COUNTER	计算 mhpmevent6 选择的事件数

MHPMCOUNTER6H 位域

3.2.86 CYCLE (0xC00) ()

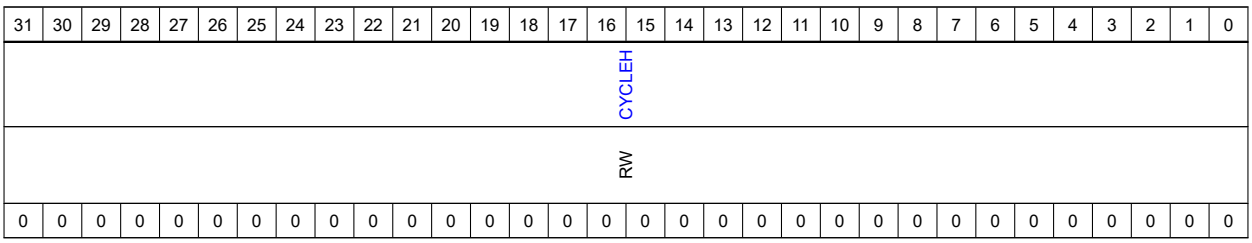


CYCLE [31:0]

位域	名称	描述
31-0	CYCLE	CYCLE 计数器

CYCLE 位域

3.2.87 CYCLEH (0xC80) ()



CYCLEH [31:0]

位域	名称	描述
31-0	CYCLEH	CYCLE 计数器高 32 位

CYCLEH 位域

3.2.88 MVENDORID (0xF11) (standard read only)

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

MVENDORID [31:0]

位域	名称	描述
31-0	MVENDORID	制造商标识

MVENDORID 位域

3.2.89 MARCHID (0xF12) (standard read only)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD	CPU_ID																														
N/A	RO																														
x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	1

MARCHID [31:0]

位域	名称	描述
30-0	CPU_ID	CPU 标识

MARCHID 位域

3.2.90 MIMPID (0xF13) (standard read only)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MAJOR																								MINOR				EXTENSION			
RO																								RO				RO			
*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*

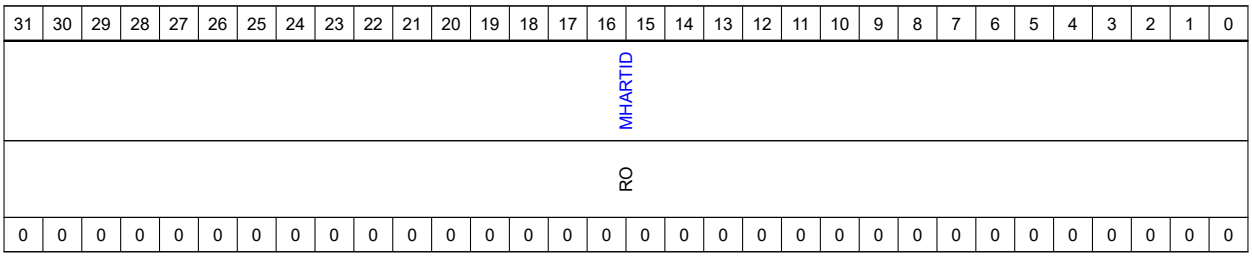
MIMPID [31:0]

位域	名称	描述
31-8	MAJOR	主要修订版本
7-4	MINOR	次要修订版本

位域	名称	描述
3-0	EXTENSION	修订扩展版本

MIMPID 位域

3.2.91 MHARTID (0xF14) (standard read only)



MHARTID [31:0]

位域	名称	描述
31-0	MHARTID	Hart ID

MHARTID 位域

4 TRAP 处理器的异常和中断

本章节介绍 RISC-V 处理器的 TRAP，即处理器的异常和中断。

本产品所有支持生成中断的外设，及其中断向量表，请查阅节 4.4。

4.1 RISC-V 处理器 TRAP 概述

本产品的采用高性能 RISC-V 处理器作为中央处理单元，处理器符合 RISC-V 规范，RISC-V 把所有由处理器本身或者外设引发的程序执行流转换称为 trap。当 trap 发生时，处理器会中断执行当前程序，转而响应 trap 的服务程序 (handler)。

按照 RISC-V 规范，trap 分为两种：异常 (exception) 和中断 (interrupt)。异常指的是由处理器本身由于指令执行异常引发的 TRAP。中断，是指由各类外设对处理器发出的请求，要求处理器响应。

中断可以分为本地中断和外部中断。本地中断主要由处理器自身或其私有设备，如软件中断、定时器中断等，可以通过访问处理器的 CSR(控制和状态寄存器 control and status registers) 管理本地中断。本产品绝大部分外设产生的各种中断由平台中断控制器 PLIC 管理，PLIC 的中断请求输出连接到 RISC-V 处理器的机器外部中断 (Machine External Interrupt)。

平台中断控制器 PLIC 管理所有片上外设中断。RISC-V 的规范定义了 PLIC 的所有功能。本产品上的 PLIC 兼容 RISC-V 规范，并支持增强功能：中断嵌套扩展和中断向量扩展。

4.1.1 异常

RISC-V 处理器支持以下异常：

- 指令地址不对齐 Instruction address misaligned
  - 指令跳转到非对齐的地址时
- 取指出错 Instruction access fault
  - 取指时总线错误
  - 取指时 PMP 错误
  - 取指时 PMA 错误
- 非法指令 Illegal instruction
  - 执行非法指令
  - 访问非法 CSR
- 断点 Breakpoint
- 数据载入 Load 地址不对齐 Load address misaligned
- 数据载入 Load 出错 Load access fault
  - Load 指令载入数据时总线错误
  - Load 指令载入数据时 PMP 错误
  - Load 指令载入数据时 PMA 错误
- Store 或 AMO 指令地址不对齐 Store/AMO address misaligned
- Store 或 AMO 指令出错 Store/AMO access fault
  - Store 指令存储数据时总线错误
  - Store 指令存储数据时 PMP 错误
  - Store 指令存储数据时 PMA 错误
- 用户模式 Environment 调用 Environment call from U-mode

- 堆栈上溢出 Stack overflow exception
- 堆栈下溢出 Stack underflow exception

4.1.2 中断

RISC-V 处理器支持的中断可以分为本地中断和外部中断：

本地中断由处理器自身的设备生成，包括：

- 机器模式 M-Mode 软件中断 software interrupt，由软件中断控制器 PLICSW 生成
- 机器模式 M-Mode 定时器中断 timer interrupt，由机器定时器 MCHTMR 生成
- 机器模式的总线访问错误 Bus read/write transaction error interrupt (M-mode)
- 机器模式的性能监视器溢出 Performance monitor overflow interrupt (M-mode)

外部中断指机器外部中断 (Machine external interrupt)。外部中断由平台中断控制器 PLIC 管理。PLIC 会对各中断按优先级仲裁，并把符合条件的中断发送到处理器。

4.1.3 平台中断控制器 PLIC

本产品的平台中断控制器 (Platform-Level Interrupt Controller) 管理片上的所有外设产生的中断。

PLIC 具有以下特性：

- PLIC 将中断输出到处理器的外部中断接口 (external interrupt)
- 支持 8 级优先级。
- 支持可配置的中断边沿触发或者电平触发
- 支持中断嵌套，允许高优先级的中断打断正在执行的低优先级中断
- 支持中断向量，中断向量模式可以简化 RISC-V 规范定义的中断的声明程序，减少中断响应延时

4.1.4 软件中断控制器 PLICSW

本产品的 RISC-V 处理器处理器的机器软件中断 (machine software interrupt) 由软件中断控制器 PLICSW 管理，PLICSW 的编程模型与 PLIC 类似，但是只包含一个中断源，用户可以通过设置 PLICSW 的中断 pending 位来生成机器软件中断。

4.1.5 机器定时器 MCHTMR

RISC-V 规范定义了机器定时器。机器定时器要求能以某个固定时钟频率运行，并生成定时器中断。本产品 RISC-V 处理器处理器的机器定时器中断 (machine timer interrupt) 由机器定时器 MCH_TMR 生成。机器定时器支持 64 位的计数器和 64 位的比较器。当计数器计数达到或者超过比较器值时，生成机器定时器中断 Machine timer interrupt。

4.2 TRAP 和处理器特权模式

本产品采用的 RISC-V 处理器处理器支持两种 RISC-V 规范定义的特权模式：机器模式 Machine Mode 和用户模式 User Mode。

RISC-V 处理器运行时，总是处于某一种特权模式。其中机器模式是权限最高的特权模式，用户模式的权限最低。特权模式可以用来对软件和分配给软件的片上资源进行隔离。HART 运行在较低权限的模式时，不允许访问那些高权限模式独享的资源 (如某一段内存，某一个外设等)。用户可以因此建立可信的执行环境，保护敏感信息或资源，使其免受非法访问。

RISC-V 规范规定，所有 TRAP 发生时，处理器模式都会转换到机器模式，来处理 TRAP 服务程序。注意，即使一些中断是为低权限模式设计的，但是当这些中断发生时，处理器仍然会进入机器模式，来处理它们的中断服务程序。

本产品上，PLIC 支持 1 个中断请求输出，连接到 RISC-V 处理器的机器外部中断 (Machine External Interrupt)。在中断发生后，处理器都会在机器模式下响应中断。

本产品上，PLIC 的中断请求输出，连接到 RISC-V 处理器的机器外部中断 (Machine External Interrupt)，用户可以在用户模式下运行软件，不过中断或异常发生时，处理器特权模式会切换到机器模式。

如果用户并不要求通过处理器的特权模式区分软件，并赋予软件不同的资源访问权限。这时可以只使用 PLIC 的机器外部中断请求输出。这样常规的程序和中断服务响应程序都运行在机器模式下。

4.3 中断嵌套

中断嵌套扩展是指本产品允许处理器暂停低优先级的中断服务程序，转而响应高优先级的中断服务程序。本产品的平台中断控制器 PLIC(Platform Level Interrupt Controlled) 在 RISC-V 规范定义的 PLIC 基础上增加这一扩展功能，使得处理器能够更好的满足实时系统应用的要求。

用户可以通过把 PLIC 的 Feature Enable Register 寄存器的 PREEMPT 位置 1，来打开中断嵌套。在处理器响应中断服务程序时，如果 mstatus CSR 的 MIE 位置 1，优先级更高的中断就可以打断当前中断服务程序。在处理器响应高优先级中断之后，会继续处理被打断的中断服务程序。

在中断服务程序中，PLIC 不会响应相同或者更低优先级的中断。

中断嵌套的具体实现过程为：

- 当中断发生时 (interrupt claim)，PLIC 会保存当前的优先级阈值寄存器 Priority Threshold Register，把新到中断的优先级赋给优先级阈值寄存器 Priority Threshold Register。
- 在中断结束时 (interrupt completion)，PLIC 会把优先级阈值寄存器 Priority Threshold Register 恢复为前值。

在本产品上，PLIC 管理几乎所有的外设中断，并将中断请求输出到处理器的机器外部中断上。因此，PLIC 的中断优先级设置只对 PLIC 管理的外设中断有效。

当软件在中断服务程序中，把 mstatus CSR 的 MIE 位置 1 后，PLIC 允许中断优先级更高的中断作为新的机器外部中断，打断当前中断。除此以外，处理器的异常和其他本地中断，如软件中断或定时器中断，也可以打断当前中断。这些异常和其他中断的优先级是不受 PLIC 的优先级设置影响的。

4.4 中断向量

当 trap 发生时，处理器会跳转到 mtvec CSR(控制和状态寄存器 control and status registers) 指向的地址，执行中断服务程序。软件可以通过 mcause CSR 判断 trap 的来源，是中断还是异常。如果发生的 trap 是外部中断 (机器外部中断)，软件需要通过 PLIC 的 interrupt pending register 进一步判定中断的来源。

中断向量扩展是 PLIC 的一项增强特性，可以减少本产品外设的中断响应延时。中断向量扩展是指当 trap 由外部中断触发时，处理器会从中断向量表中读取中断服务程序的地址指针，并赋予给处理器的程序计数器 (program counter PC)。这样处理器会直接跳转到对应的中断服务程序，软件无需通过先读取 mcause CSR，再读取 PLIC 的 claim/complete 寄存器来判定中断的源头。

用户可以通过把 PLIC 的 Feature Enable Register 寄存器的 VECTORED 位置 1，并且把 MMISC CSR 的 VEC_PLIC 位置 1，来打开中断向量模式。

RISC-V 规范规定，默认状态下，所有的 trap 都由机器模式处理。这时，中断向量表的基地址存储在处理器的 mtvec(机器 trap 向量基地址寄存器)CSR 中。

对于机器外部中断 (由 PLIC 的 Target 0 输出，引发 mip CSR 的 MEIP 置 1)：

- 基地址即 mtvec，序号为 N 的中断向量地址为：基地址 + 4 x N。
- 基地址为 mtvec + 1024，序号为 N 的中断向量地址为：基地址 + 4 x N
- 序号为 N 的中断向量地址为：基地址 + 4 x N

本产品的中断向量分配如下：

IRQ 编号	IRQ 名称	描述
1	GPIO0_A	
2	GPIO0_B	
3	GPIO0_X	
4	GPIO0_Y	
5	GPTMR0	
6	GPTMR1	
7	GPTMR2	
8	GPTMR3	
9	LIN0	
10	LIN1	
11	LIN2	
12	LIN3	
13	UART0	
14	UART1	
15	UART2	
16	UART3	
17	UART4	
18	UART5	
19	UART6	
20	UART7	
21	I2C0	
22	I2C1	
23	I2C2	
24	I2C3	
25	SPI0	
26	SPI1	
27	SPI2	
28	SPI3	
29	TSNS	
30	MBX0A	
31	MBX0B	
32	WDG0	

IRQ 编号	IRQ 名称	描述
33	WDG1	
34	HDMA	
35	CAN0	
36	CAN1	
37	CAN2	
38	CAN3	
39	PTPC	
40	PWM0	
41	QEI0	
42	SEI0	
43	MMC0	
44	TRGMUX0	
45	PWM1	
46	QEI1	
47	SEI1	
48	MMC1	
49	TRGMUX1	
50	RDC	
51	USB0	
52	XPI0	
53	SDP	
54	PSEC	
55	SECMON	
56	RNG	
57	FUSE	
58	ADC0	
59	ADC1	
60	DAC0	
61	DAC1	
62	ACMP_0	
63	ACMP_1	
64	SYSCTL	
65	PGPIO	
66	PTMR	
67	PUART	
68	PWDG	
69	BROWNOUT	
70	PAD_WAKEUP	
71	DEBUG0	
72	DEBUG1	

IRQ 编号	IRQ 名称	描述

表 10: IRQ 列表

4.5 中断响应流程

常规的中断服务程序流程，以机器模式（M-Mode）为例，处理器硬件会采取以下动作：

- 更新 mepc CSR 的值为当前 PC(program counter)
- 更新 mstatus CSR 的值：
  - 更新 MPP 位域值为当前特权模式
  - 更新 MPIE 的值为 MIE
  - MIE 位置 0
- 更新 mcause CSR 的值，包含当前中断源信息
- 处理器特权模式变更为机器模式 (machine mode)
- 根据 mtvec CSR 的值更新 PC(program counter)
  - 如果不支持中断向量，mtvec CSR 的值赋给 PC
  - 如果支持中断向量，基地址 + 4 * N 的值赋给 PC，这里 N 是 PLIC 的中断 ID 号码

。

此后，即开始执行中断服务程序的软件，此时软件应该：

- 保存必要的处理器整数寄存器，压入堆栈
- 如果不支持中断向量，或者发生的 TRAP 是异常或者本地中断，要保存 mcause CSR
- 如果希望支持中断嵌套，应当：
  - 保存 mepc CSR
  - 保存 mstatus CSR
  - 根据需要保存其他必要的 CSR，如浮点数/DSP 扩展相关 CSR 等
  - 把 mstatus CSR 的 MIE 位置 1，这时允许产生新的中断
- 开始执行中断服务程序
  - 如果不支持中断向量，或者发生的 TRAP 是异常或者本地中断，需要根据 mcause 判断 TRAP 产生的原因，并执行相应的处理程序。如果是由 PLIC 发送到处理器的外部中断，需要读取 PLIC 的中断声明/完成寄存器 (Claim/Compplete register) 来完成中断声明，并获得中断 ID 号码来确定中断源，之后再执行相应的中断服务程序。
  - 如果支持中断向量，直接读取相应外设的中断标识位，判断中断原因，执行中断服务程序

在完成 TRAP 的对应服务程序之后，处理流程如下：

- 读取 PLIC 的中断声明/完成寄存器 (Claim/Compplete register) 来表示中断完成。
- 如果是允许嵌套的中断，应当：
  - 把 mstatus CSR 的 MIE 位置 0，不再响应新的中断
  - 插入一条 FENCE 指令，保证之前的读取 PLIC 中断声明/完成寄存器操作完成
  - 恢复 mepc CSR
  - 恢复 mstatus CSR
  - 恢复之前保存的其他 CSR，如浮点数/DSP 扩展相关 CSR 等
- 软件恢复之前压入堆栈的处理器整数寄存器
- 软件执行 MRET 指令，从机器模式返回。

一旦执行 MRET 指令，处理器硬件会执行以下操作：

- 把 mepc CSR 的值恢复到 PC(program counter)
- 恢复 mstatus CSR 的值：
  - 根据 MPP 位域值恢复处理器的特权模式
  - 把 MPIE 的值恢复到 MIE

5 机器定时器 MCHTMR

本章节介绍机器定时器 MCHTMR 的功能和特性。

5.1 特性总结

本章节介绍机器定时器 MCHTMR 的主要特性：

- 64 位实时计数器
- 64 位计数比较器
- 支持 32 位和 64 位寄存器访问
- 支持向 RISC-V 核心发送机器定时器中断

MCHTMR 的框图如图 2。

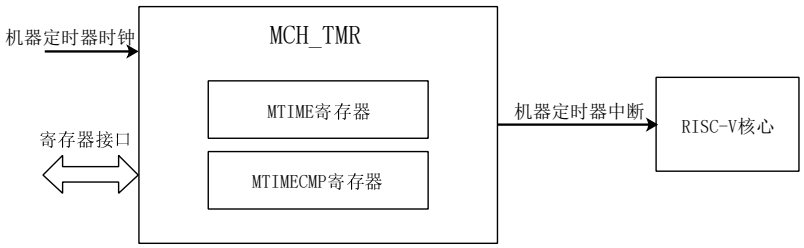


图 2: MCHTMR 框图

5.2 功能描述

本章节描述机器定时器 MCHTMR 的功能。

5.2.1 计时与定时功能

MCHTMR 内置一个 64 位计数器，机器定时器时钟每翻转一次计数器的值增加 1。读取 MTIME 寄存器可获取当前计数值。

定时功能通过设置 MTIMECMP 寄存器实现，当计数器的值大于或等于 MTIMECMP 寄存器时，MCHTMR 向 RISC-V 核心发出机器定时器中断。

当机器定时器中断有效时，对 MTIMECMP 寄存器的写操作会清除该中断。

5.2.2 计时器重置

内部计数器从 1 开始计时，改写 MTIME 寄存器能够将计数器重置为特定的值，修改 MTIME 寄存器需遵循一定的规则：

MTIME 寄存器分为 MTIME[63:32] 和 MTIME[31:0] 两个 32 位寄存器，修改 MTIME 寄存器时必须以 32 位方式分别对两个 32 位寄存器进行操作。

- 如果将设置 MTIME[31:29] 的目标值为 7，则：
    1. 设置 MTIME[31:0] 为 0
    2. 设置 MTIME[63:32] 目标值
    3. 设置 MTIME[31：0] 目标值
- 如果将设置 MTIME[31:29] 的目标值不为 7，则：

1. 设置 MTIME[31：0] 目标值
2. 设置 MTIME[63:32] 目标值

5.3 MCHTMR 寄存器说明

MCHTMR 的寄存器列表如下：

MCHTMR base address: 0xE6000000

地址偏移	名称	描述	复位值
0x0000	MTIME	机器计时器	0x0000000000020210
0x0008	MTIMECMP	机器定时比较器	0x0000000000020210

表 11: MCHTMR 寄存器列表

5.4 MCHTMR 寄存器详细信息

5.4.1 MTIME (0x0)

63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
MTIME																															
RW																															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	0	1	0	0	0	0

MTIME [63:32]

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

MTIME [31:0]

位域	名称	描述
63-0	MTIME	计时数值

MTIME 位域

5.4.2 MTIMECMP (0x8)

63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
MTIMECMP																															

63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
RW																															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	0	1	0	0	0	0

MTIMECMP [63:32]

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

MTIMECMP [31:0]

位域	名称	描述
63-0	MTIMECMP	设置定时器计数值，当机器定时器中断有效时，对该寄存器进行写操作可清除机器定时器中断

MTIMECMP 位域

6 平台级中断控制器 PLIC

本章节介绍平台级中断控制器 PLIC 的功能和特性。

6.1 特性总结

本章节介绍平台级中断控制器 PLIC 的主要特性：

- 可配置中断触发方式
- 支持向量式中断
- 支持中断优先级抢占

6.2 功能描述

本章节描述平台级中断控制器 PLIC 的功能。

PLIC 汇集各中断源的中断请求进入中断网关，根据配置完成优先级处理后将中断请求转发给 RISC-V 核心。

中断配置包括使能和优先级等。注意软件不应在有中断未处理的情况下更改中断配置。

6.2.1 向量式中断扩展

RISC-V 标准要求 PLIC 在收到外部中断时能够通知 RISC-V 核心,RISC-V 核心读取 PLIC 的 CLAIM_COMPLETE 寄存器获取中断 ID，该读操作称为中断响应 (claim)。RISC-V 核心获取中断 ID 后执行相应中断的处理程序，再将该 ID 写回 PLIC 的 CLAIM_COMPLETE 寄存器，该写操作称为中断完成 (complete)。

本产品的 PLIC 还支持向量 (vector) 中断扩展，即 PLIC 通知 RISC-V 核心中断请求的同时，会将中断 ID 一同发出，RISC-V 核心以硬件方式向 PLIC 发送中断响应，软件无需读取 PLIC 来获取中断 ID 和执行中断响应，以加快中断处理速度。

通过配置寄存器 FEATURE_ENABLE.VECTORED 来使能向量中断扩展。

6.2.2 中断优先级抢占

PLIC 支持中断抢占优先级，在 PRIORITY 寄存器中可配置每个中断 ID 的优先级，优先级有效值为 1~7。当优先级抢占使能时，若 RISC-V 核心已对某中断进行了中断响应但尚未中断完成，此时如果有新的优先级更高的中断出现，PLIC 会再次向 RISC-V 核心发送新的中断请求。

将寄存器 FEATURE_ENABLE.PREEMPT 设置为 1 可使能优先级抢占功能。

在中断处理程序中通过使能全局中断寄存器 mstatus.MIE，来允许高优先级的中断抢占当前正在执行的中断处理程序。

在非向量模式下，RISC-V 核心应在保存完上下文并执行完中断响应之后使能中断抢占。由于中断响应和中断完成都是对 device 区域执行 loadstore 指令，它们会被自动保证顺序，所以不需要额外禁止全局中断，也不需要在发出中断完成操作后插入 FENCE 指令。

建议的非向量模式中断处理程序如下：

1. 保存寄存器和 CSR 到 stack
2. 向 PLIC 发送中断响应操作
3. 使能全局中断 (mstatus.MIE)
4. 处理中断
5. 向 PLIC 发送中断完成操作

6. 恢复寄存器和 CSR
7. 从中断处理程序返回

在向量模式下，RISC-V 核心在保存完上下文之后即可使能中断抢占。为了避免和下一个中断响应出现冲突，在执行中断完成前需关闭全局中断使能。另外在退出中断处理程序前还应执行一次 FENCE io 指令以保证中断完成操作已到达 PLIC。

建议的向量模式中断处理程序如下：

1. 保存寄存器和 CSR 到 stack
2. 使能全局中断 (mstatus.MIE)
3. 处理中断
4. 禁止全局中断 (mstatus.MIE)
5. 向 PLIC 发送中断完成操作
6. 恢复寄存器和 CSR
7. 执行 FENCE io 指令确保中断完成操作已被送达 PLIC
8. 从中断处理程序返回

寄存器 THRESHOLD 可设置中断优先级阈值，只有大于该阈值的中断才会被通知 RISC-V 核心。当中断已响应但未完成时，THRESHOLD 寄存器会实时更新为当前正在处理的中断的优先级，用以屏蔽更低优先级的中断而放行更高优先级的中断抢占。

6.3 PLIC 寄存器说明

PLIC 的寄存器列表如下：

PLIC base address: 0xE4000000

地址偏移	名称	描述	复位值
0x0000	FEATURE	特性使能寄存器	0x00000000
0x0004	PRIORITY[PRIORITY1]	中断优先级寄存器	0x00000001
0x0008	PRIORITY[PRIORITY2]	中断优先级寄存器	0x00000001
0x000C	PRIORITY[PRIORITY3]	中断优先级寄存器	0x00000001
0x0010	PRIORITY[PRIORITY4]	中断优先级寄存器	0x00000001
0x0014	PRIORITY[PRIORITY5]	中断优先级寄存器	0x00000001
0x0018	PRIORITY[PRIORITY6]	中断优先级寄存器	0x00000001
0x001C	PRIORITY[PRIORITY7]	中断优先级寄存器	0x00000001
0x0020	PRIORITY[PRIORITY8]	中断优先级寄存器	0x00000001
0x0024	PRIORITY[PRIORITY9]	中断优先级寄存器	0x00000001
0x0028	PRIORITY[PRIORITY10]	中断优先级寄存器	0x00000001
0x002C	PRIORITY[PRIORITY11]	中断优先级寄存器	0x00000001
0x0030	PRIORITY[PRIORITY12]	中断优先级寄存器	0x00000001
0x0034	PRIORITY[PRIORITY13]	中断优先级寄存器	0x00000001
0x0038	PRIORITY[PRIORITY14]	中断优先级寄存器	0x00000001
0x003C	PRIORITY[PRIORITY15]	中断优先级寄存器	0x00000001
0x0040	PRIORITY[PRIORITY16]	中断优先级寄存器	0x00000001
0x0044	PRIORITY[PRIORITY17]	中断优先级寄存器	0x00000001

地址偏移	名称	描述	复位值
0x0048	PRIORITY[PRIORITY18]	中断优先级寄存器	0x00000001
0x004C	PRIORITY[PRIORITY19]	中断优先级寄存器	0x00000001
0x0050	PRIORITY[PRIORITY20]	中断优先级寄存器	0x00000001
0x0054	PRIORITY[PRIORITY21]	中断优先级寄存器	0x00000001
0x0058	PRIORITY[PRIORITY22]	中断优先级寄存器	0x00000001
0x005C	PRIORITY[PRIORITY23]	中断优先级寄存器	0x00000001
0x0060	PRIORITY[PRIORITY24]	中断优先级寄存器	0x00000001
0x0064	PRIORITY[PRIORITY25]	中断优先级寄存器	0x00000001
0x0068	PRIORITY[PRIORITY26]	中断优先级寄存器	0x00000001
0x006C	PRIORITY[PRIORITY27]	中断优先级寄存器	0x00000001
0x0070	PRIORITY[PRIORITY28]	中断优先级寄存器	0x00000001
0x0074	PRIORITY[PRIORITY29]	中断优先级寄存器	0x00000001
0x0078	PRIORITY[PRIORITY30]	中断优先级寄存器	0x00000001
0x007C	PRIORITY[PRIORITY31]	中断优先级寄存器	0x00000001
0x0080	PRIORITY[PRIORITY32]	中断优先级寄存器	0x00000001
0x0084	PRIORITY[PRIORITY33]	中断优先级寄存器	0x00000001
0x0088	PRIORITY[PRIORITY34]	中断优先级寄存器	0x00000001
0x008C	PRIORITY[PRIORITY35]	中断优先级寄存器	0x00000001
0x0090	PRIORITY[PRIORITY36]	中断优先级寄存器	0x00000001
0x0094	PRIORITY[PRIORITY37]	中断优先级寄存器	0x00000001
0x0098	PRIORITY[PRIORITY38]	中断优先级寄存器	0x00000001
0x009C	PRIORITY[PRIORITY39]	中断优先级寄存器	0x00000001
0x00A0	PRIORITY[PRIORITY40]	中断优先级寄存器	0x00000001
0x00A4	PRIORITY[PRIORITY41]	中断优先级寄存器	0x00000001
0x00A8	PRIORITY[PRIORITY42]	中断优先级寄存器	0x00000001
0x00AC	PRIORITY[PRIORITY43]	中断优先级寄存器	0x00000001
0x00B0	PRIORITY[PRIORITY44]	中断优先级寄存器	0x00000001
0x00B4	PRIORITY[PRIORITY45]	中断优先级寄存器	0x00000001
0x00B8	PRIORITY[PRIORITY46]	中断优先级寄存器	0x00000001
0x00BC	PRIORITY[PRIORITY47]	中断优先级寄存器	0x00000001
0x00C0	PRIORITY[PRIORITY48]	中断优先级寄存器	0x00000001
0x00C4	PRIORITY[PRIORITY49]	中断优先级寄存器	0x00000001
0x00C8	PRIORITY[PRIORITY50]	中断优先级寄存器	0x00000001
0x00CC	PRIORITY[PRIORITY51]	中断优先级寄存器	0x00000001
0x00D0	PRIORITY[PRIORITY52]	中断优先级寄存器	0x00000001
0x00D4	PRIORITY[PRIORITY53]	中断优先级寄存器	0x00000001
0x00D8	PRIORITY[PRIORITY54]	中断优先级寄存器	0x00000001
0x00DC	PRIORITY[PRIORITY55]	中断优先级寄存器	0x00000001
0x00E0	PRIORITY[PRIORITY56]	中断优先级寄存器	0x00000001
0x00E4	PRIORITY[PRIORITY57]	中断优先级寄存器	0x00000001

地址偏移	名称	描述	复位值
0x00E8	PRIORITY[PRIORITY58]	中断优先级寄存器	0x00000001
0x00EC	PRIORITY[PRIORITY59]	中断优先级寄存器	0x00000001
0x00F0	PRIORITY[PRIORITY60]	中断优先级寄存器	0x00000001
0x00F4	PRIORITY[PRIORITY61]	中断优先级寄存器	0x00000001
0x00F8	PRIORITY[PRIORITY62]	中断优先级寄存器	0x00000001
0x00FC	PRIORITY[PRIORITY63]	中断优先级寄存器	0x00000001
0x0100	PRIORITY[PRIORITY64]	中断优先级寄存器	0x00000001
0x0104	PRIORITY[PRIORITY65]	中断优先级寄存器	0x00000001
0x0108	PRIORITY[PRIORITY66]	中断优先级寄存器	0x00000001
0x010C	PRIORITY[PRIORITY67]	中断优先级寄存器	0x00000001
0x0110	PRIORITY[PRIORITY68]	中断优先级寄存器	0x00000001
0x0114	PRIORITY[PRIORITY69]	中断优先级寄存器	0x00000001
0x0118	PRIORITY[PRIORITY70]	中断优先级寄存器	0x00000001
0x011C	PRIORITY[PRIORITY71]	中断优先级寄存器	0x00000001
0x0120	PRIORITY[PRIORITY72]	中断优先级寄存器	0x00000001
0x0124	PRIORITY[PRIORITY73]	中断优先级寄存器	0x00000001
0x0128	PRIORITY[PRIORITY74]	中断优先级寄存器	0x00000001
0x012C	PRIORITY[PRIORITY75]	中断优先级寄存器	0x00000001
0x0130	PRIORITY[PRIORITY76]	中断优先级寄存器	0x00000001
0x0134	PRIORITY[PRIORITY77]	中断优先级寄存器	0x00000001
0x0138	PRIORITY[PRIORITY78]	中断优先级寄存器	0x00000001
0x013C	PRIORITY[PRIORITY79]	中断优先级寄存器	0x00000001
0x0140	PRIORITY[PRIORITY80]	中断优先级寄存器	0x00000001
0x0144	PRIORITY[PRIORITY81]	中断优先级寄存器	0x00000001
0x0148	PRIORITY[PRIORITY82]	中断优先级寄存器	0x00000001
0x014C	PRIORITY[PRIORITY83]	中断优先级寄存器	0x00000001
0x0150	PRIORITY[PRIORITY84]	中断优先级寄存器	0x00000001
0x0154	PRIORITY[PRIORITY85]	中断优先级寄存器	0x00000001
0x0158	PRIORITY[PRIORITY86]	中断优先级寄存器	0x00000001
0x015C	PRIORITY[PRIORITY87]	中断优先级寄存器	0x00000001
0x0160	PRIORITY[PRIORITY88]	中断优先级寄存器	0x00000001
0x0164	PRIORITY[PRIORITY89]	中断优先级寄存器	0x00000001
0x0168	PRIORITY[PRIORITY90]	中断优先级寄存器	0x00000001
0x016C	PRIORITY[PRIORITY91]	中断优先级寄存器	0x00000001
0x0170	PRIORITY[PRIORITY92]	中断优先级寄存器	0x00000001
0x0174	PRIORITY[PRIORITY93]	中断优先级寄存器	0x00000001
0x0178	PRIORITY[PRIORITY94]	中断优先级寄存器	0x00000001
0x017C	PRIORITY[PRIORITY95]	中断优先级寄存器	0x00000001
0x0180	PRIORITY[PRIORITY96]	中断优先级寄存器	0x00000001
0x0184	PRIORITY[PRIORITY97]	中断优先级寄存器	0x00000001

地址偏移	名称	描述	复位值
0x0188	PRIORITY[PRIORITY98]	中断优先级寄存器	0x00000001
0x018C	PRIORITY[PRIORITY99]	中断优先级寄存器	0x00000001
0x0190	PRIORITY[PRIORITY100]	中断优先级寄存器	0x00000001
0x0194	PRIORITY[PRIORITY101]	中断优先级寄存器	0x00000001
0x0198	PRIORITY[PRIORITY102]	中断优先级寄存器	0x00000001
0x019C	PRIORITY[PRIORITY103]	中断优先级寄存器	0x00000001
0x01A0	PRIORITY[PRIORITY104]	中断优先级寄存器	0x00000001
0x01A4	PRIORITY[PRIORITY105]	中断优先级寄存器	0x00000001
0x01A8	PRIORITY[PRIORITY106]	中断优先级寄存器	0x00000001
0x01AC	PRIORITY[PRIORITY107]	中断优先级寄存器	0x00000001
0x01B0	PRIORITY[PRIORITY108]	中断优先级寄存器	0x00000001
0x01B4	PRIORITY[PRIORITY109]	中断优先级寄存器	0x00000001
0x01B8	PRIORITY[PRIORITY110]	中断优先级寄存器	0x00000001
0x01BC	PRIORITY[PRIORITY111]	中断优先级寄存器	0x00000001
0x01C0	PRIORITY[PRIORITY112]	中断优先级寄存器	0x00000001
0x01C4	PRIORITY[PRIORITY113]	中断优先级寄存器	0x00000001
0x01C8	PRIORITY[PRIORITY114]	中断优先级寄存器	0x00000001
0x01CC	PRIORITY[PRIORITY115]	中断优先级寄存器	0x00000001
0x01D0	PRIORITY[PRIORITY116]	中断优先级寄存器	0x00000001
0x01D4	PRIORITY[PRIORITY117]	中断优先级寄存器	0x00000001
0x01D8	PRIORITY[PRIORITY118]	中断优先级寄存器	0x00000001
0x01DC	PRIORITY[PRIORITY119]	中断优先级寄存器	0x00000001
0x01E0	PRIORITY[PRIORITY120]	中断优先级寄存器	0x00000001
0x01E4	PRIORITY[PRIORITY121]	中断优先级寄存器	0x00000001
0x01E8	PRIORITY[PRIORITY122]	中断优先级寄存器	0x00000001
0x01EC	PRIORITY[PRIORITY123]	中断优先级寄存器	0x00000001
0x01F0	PRIORITY[PRIORITY124]	中断优先级寄存器	0x00000001
0x01F4	PRIORITY[PRIORITY125]	中断优先级寄存器	0x00000001
0x01F8	PRIORITY[PRIORITY126]	中断优先级寄存器	0x00000001
0x01FC	PRIORITY[PRIORITY127]	中断优先级寄存器	0x00000001
0x1000	PENDING[PENDING0]	待处理中断列表	0x00000000
0x1004	PENDING[PENDING1]	待处理中断列表	0x00000000
0x1008	PENDING[PENDING2]	待处理中断列表	0x00000000
0x100C	PENDING[PENDING3]	待处理中断列表	0x00000000
0x1080	TRIGGER[TRIGGER0]	中断触发方式配置	0x00000000
0x1084	TRIGGER[TRIGGER1]	中断触发方式配置	0x00000000
0x1088	TRIGGER[TRIGGER2]	中断触发方式配置	0x00000000
0x108C	TRIGGER[TRIGGER3]	中断触发方式配置	0x00000000
0x1100	NUMBER	PLIC 属性查询	-
0x1104	INFO	PLIC 属性查询	-

地址偏移	名称	描述	复位值
0x2000	TARGETINT[TARGET0][INTEN][INTEN0]	机器中断使能	0x00000000
0x2004	TARGETINT[TARGET0][INTEN][INTEN1]	机器中断使能	0x00000000
0x2008	TARGETINT[TARGET0][INTEN][INTEN2]	机器中断使能	0x00000000
0x200C	TARGETINT[TARGET0][INTEN][INTEN3]	机器中断使能	0x00000000
0x200000	TARGETCONFIG[TARGET0][THRESHOLD]	机器中断优先级阈值	0x00000000
0x200004	TARGETCONFIG[TARGET0][CLAIM]	中断响应和完成	0x00000000
0x200400	TARGETCONFIG[TARGET0][PPS]	中断抢占状态	0x00000000

表 12: PLIC 寄存器列表

6.4 PLIC 寄存器详细信息

PLIC 的寄存器详细说明如下：

6.4.1 FEATURE (0x0)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																														VECTORED	PREEMPT
N/A																														RW	RW
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0

FEATURE [31:0]

位域	名称	描述
1	VECTORED	向量式中断使能 0：禁止 1：使能
0	PREEMPT	中断优先级抢占使能 0：禁止 1：使能

FEATURE 位域

6.4.2 PRIORITY (0x4 + 0x4 * n)

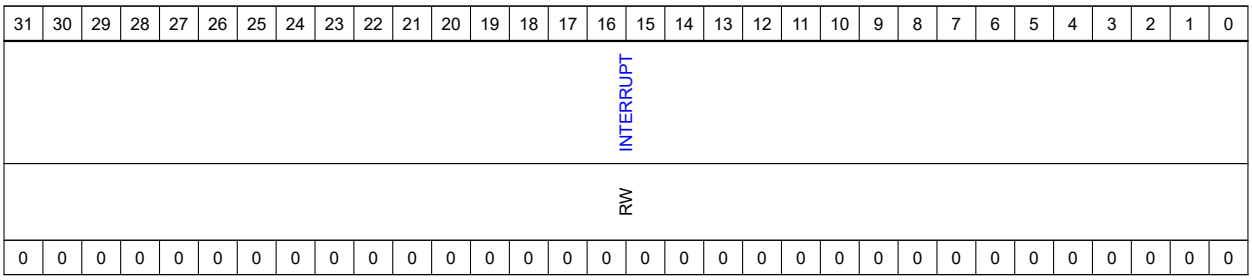
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PRIORITY																															
RW																															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

PRIORITY [31:0]

位域	名称	描述
31-0	PRIORITY	中断源优先级，有效值为 0 到 7。 0：禁止该中断 1-7：设置中断优先级，数字越大优先级越高

PRIORITY 位域

6.4.3 PENDING (0x1000 + 0x4 * n)

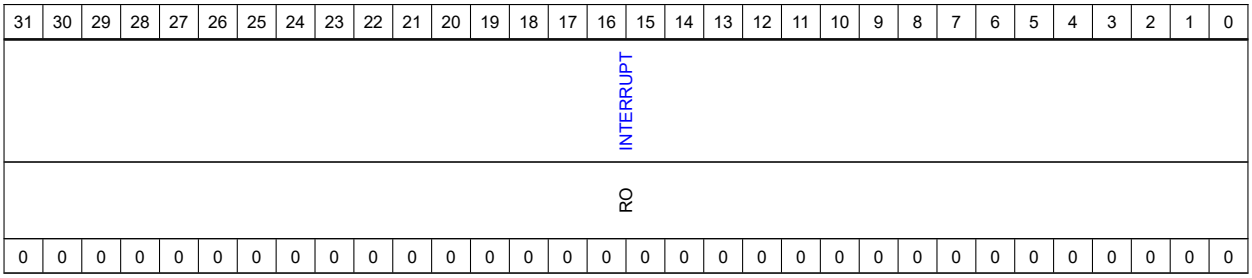


PENDING [31:0]

位域	名称	描述
31-0	INTERRUPT	待处理的中断列表，每个 bit 代表一个中断源

PENDING 位域

6.4.4 TRIGGER (0x1080 + 0x4 * n)

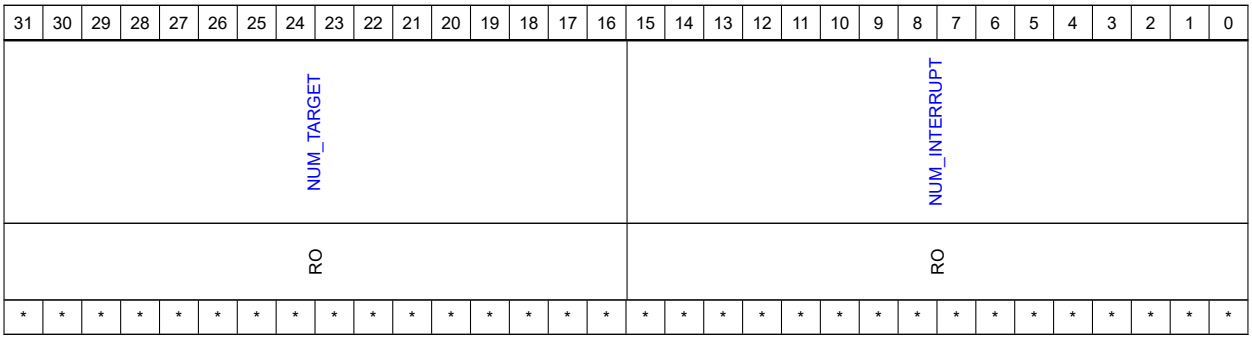


TRIGGER [31:0]

位域	名称	描述
31-0	INTERRUPT	设置中断的触发方式，每个 bit 代表一个中断源。 0：电平触发 1：边沿触发

TRIGGER 位域

6.4.5 NUMBER (0x1100)

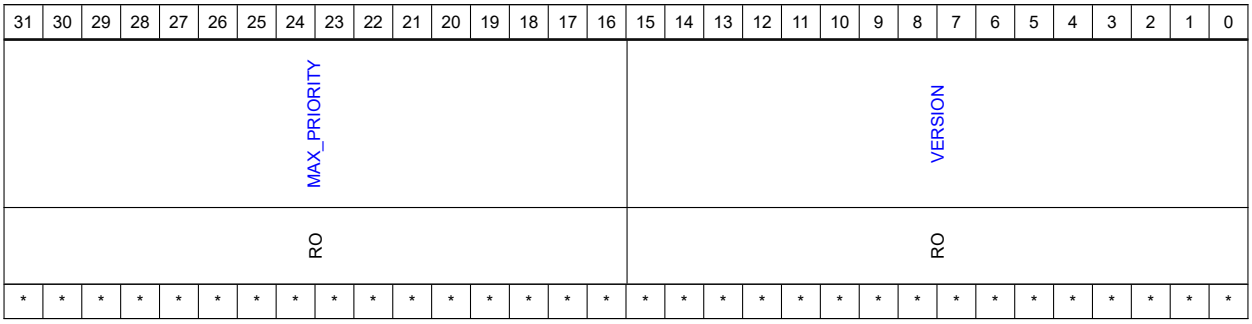


NUMBER [31:0]

位域	名称	描述
31-16	NUM_TARGET	支持的 target 数量
15-0	NUM_INTERRUPT	支持的中断源数量

NUMBER 位域

6.4.6 INFO (0x1104)

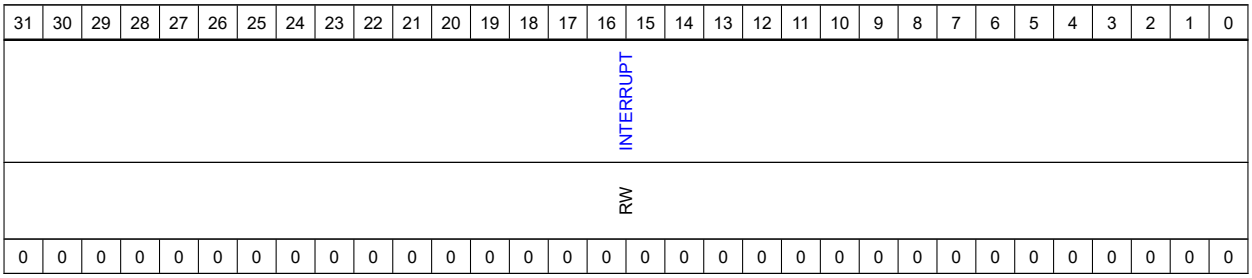


INFO [31:0]

位域	名称	描述
31-16	MAX_PRIORITY	支持的最大优先级
15-0	VERSION	IP 版本

INFO 位域

6.4.7 TARGETINT[INTEN] (0x2000 + 0x80 * n + 0x4 * m)

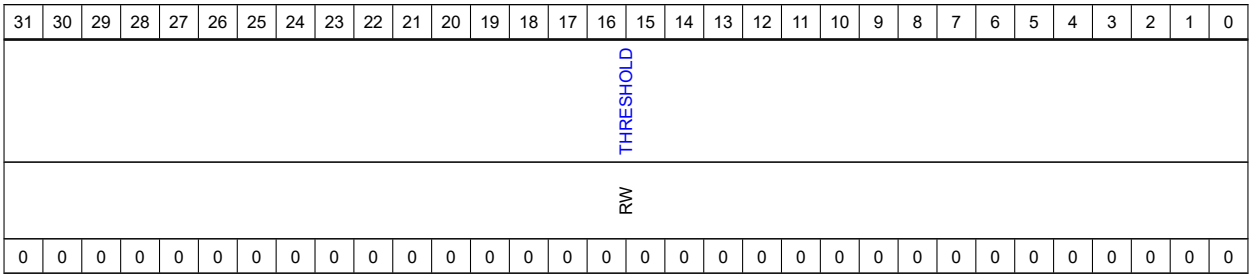


TARGETINT[INTEN] [31:0]

位域	名称	描述
31-0	INTERRUPT	中断使能，每个 bit 代表一个中断源。如果所在寄存器属于 TARGET0，则将中断作为机器中断使能，如果寄存器属于 TARGET1，则将中断作为特权中断使能

TARGETINT[INTEN] 位域

6.4.8 TARGETCONFIG[THRESHOLD] (0x200000 + 0x1000 * n)

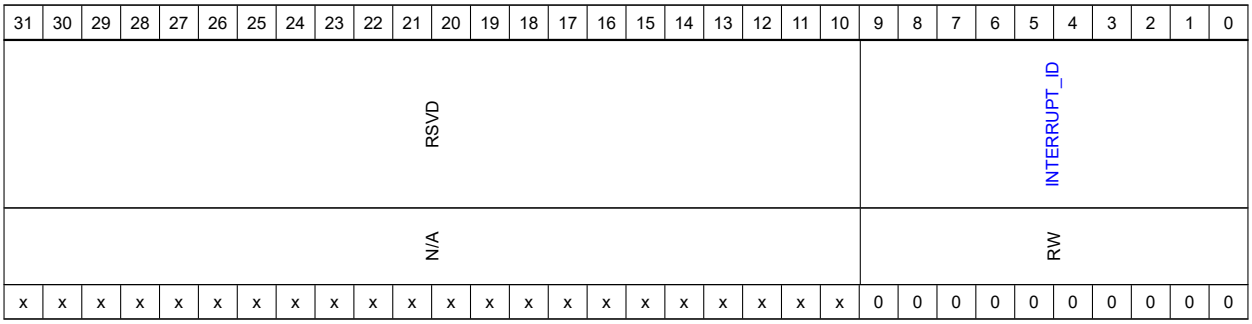


TARGETCONFIG[THRESHOLD] [31:0]

位域	名称	描述
31-0	THRESHOLD	中断优先级阈值

TARGETCONFIG[THRESHOLD] 位域

6.4.9 TARGETCONFIG[CLAIM] (0x200004 + 0x1000 * n)

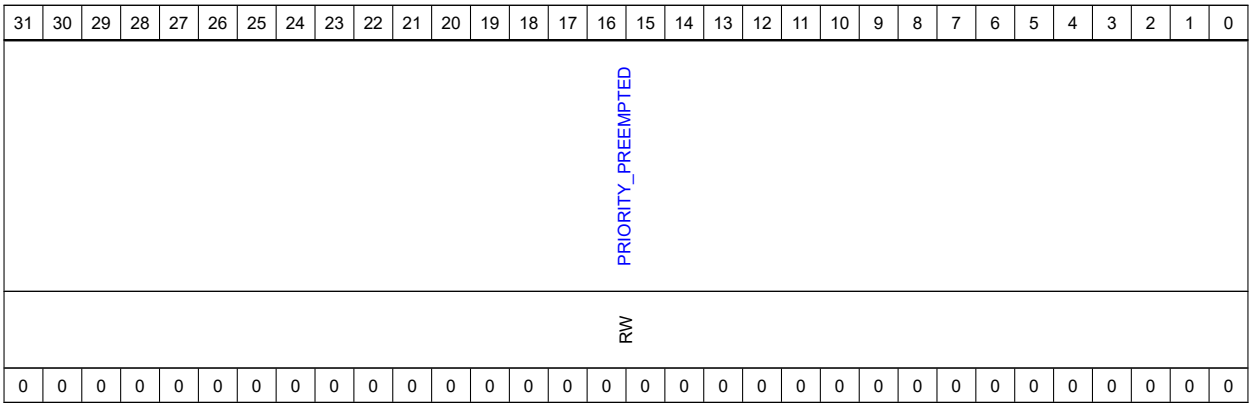


TARGETCONFIG[CLAIM] [31:0]

位域	名称	描述
9-0	INTERRUPT_ID	读该寄存器可获得待处理的中断号，并实现中断响应操作。将中断号写入该寄存器可实现中断完成操作。

TARGETCONFIG[CLAIM] 位域

6.4.10 TARGETCONFIG[PPS] (0x200400 + 0x1000 * n)



TARGETCONFIG[PPS] [31:0]

位域	名称	描述
31-0	PRIORITY_PREEMPTED	每一位代表一个中断优先级，置 1 表示该中断优先级正被更高优先级的中断抢占。

TARGETCONFIG[PPS] 位域

7 平台级软件中断控制器 PLICSW

本章节介绍平台级软件中断控制器 PLICSW 的功能和特性。

7.1 特性总结

本章节介绍平台级中断控制器 PLIC 的主要特性：

- 支持通过软件编程产生中断

7.2 功能描述

本章节描述平台级中断控制器 PLIC 的功能。

RISC-V 核心向寄存器 PENDING 写 1 可触发软件中断，RISC-V 核心收到中断后需对 CLAIM 寄存器进行读操作以完成中断响应（claim），在中断处理完成后需对 CLAIM 寄存器进行写操作以执行中断完成操作。

7.3 PLICSW 寄存器说明

PLIC 的寄存器列表如下：

PLICSW base address: 0xE6400000

地址偏移	名称	描述	复位值
0x1000	PENDING	软件中断触发寄存器	0x00000000
0x2000	INTEN	软件中断使能	0x00000000
0x200004	CLAIM	软件中断响应和完成	0x00000000

表 13: PLIC_SW 寄存器列表

7.3.1 PLICSW 寄存器详细信息

PLIC 的寄存器详细说明如下：

7.3.2 PENDING (0x1000)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																														INTERRUPT	RSVD
N/A																														RW	N/A
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	x

PENDING [31:0]

位域	名称	描述
1	INTERRUPT	对该寄存器位写 1 可触发软件中断

PENDING 位域

7.3.3 INTEN (0x2000)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																															INTERRUPT
N/A																															RW
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0

INTEN [31:0]

位域	名称	描述
0	INTERRUPT	软件中断使能

INTEN 位域

7.3.4 CLAIM (0x200004)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																															INTERRUPT_ID
N/A																															RW
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0

CLAIM [31:0]

位域	名称	描述
0	INTERRUPT_ID	读该寄存器可实现中断响应操作。将 0x1 写入该寄存器可实现中断完成操作。

CLAIM 位域

8 电源管理

本章节介绍电源管理系统。

8.1 电源系统结构

本产品在电源结构上划分了 2 个主要的电源域，如图 3所示：

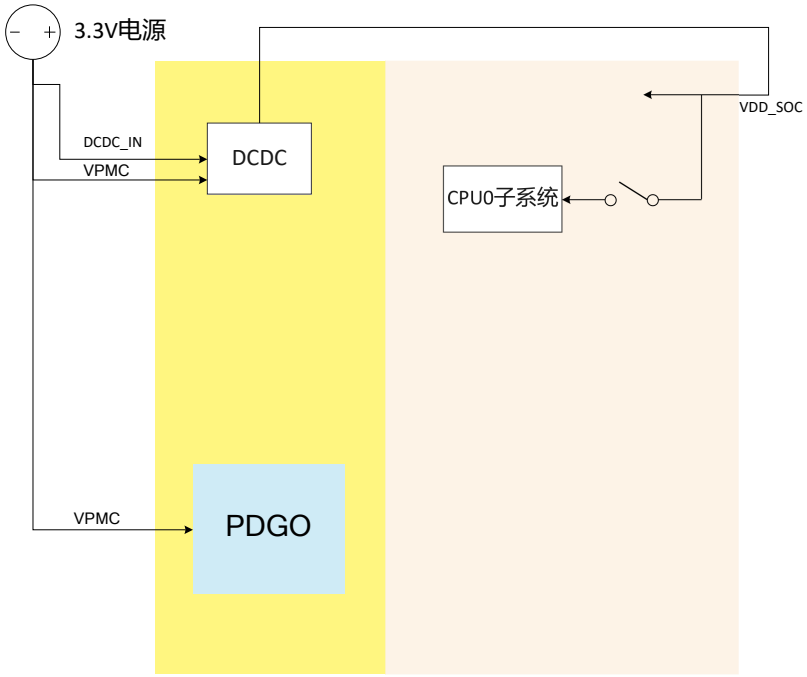


图 3: 电源系统结构框图

2 个电源域包括：

- 系统电源域中包含了芯片中大部分的功能模块，其中有 1 个可独立开关电源的子系统 (CPU0)，以及若干其他功能模块
- 电源管理域中有一个 DCDC 可用于为系统电源域供电，以及数个能够唤醒系统的功能模块，能够在系统电源域关闭的状态下保持低功耗运行。电源管理域里的 PDGO 模块，能够在电源管理域其它模块关闭的状态下以极低功耗运行，并保存必要的数据

各电源域中的主要功能模块可在图 1中查看。

典型应用场景下芯片使用单一 3.3V 电源供电即可工作。

8.2 电源供电系统

芯片内置 2 路 LDO 线性电源和 1 路 DCDC 开关电源，以支持单一 3.3V 供电应用。

电源供电系统框图如图 4：

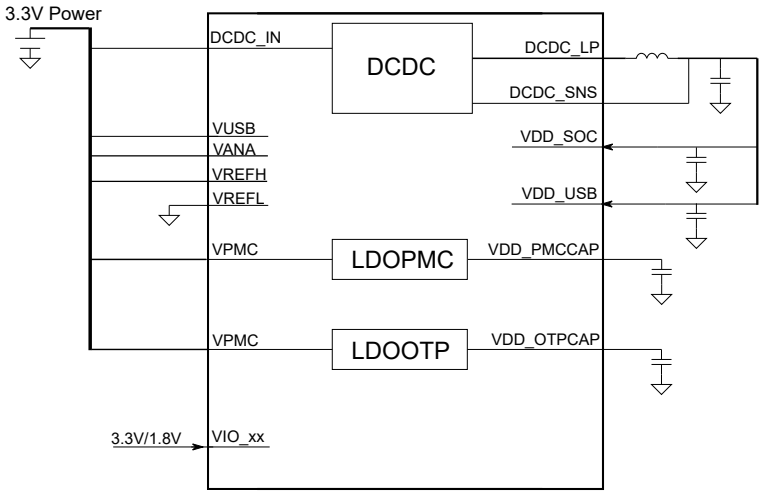


图 4: 电源供电系统框图

8.2.1 开关电源 DCDC

DCDC 位于电源管理域，将 3.3V 的外部电源转化为典型值为 1.1V 的供电，能够为系统电源域的电路提供电源。

DCDC 输出电压可调节，且支持动态电压频率调整。

DCDC_IN 是 DCDC 的电源输入引脚。DCDC_LP 是 DCDC 的电源输出引脚，应连接一个 2.2~10uH 的电感。DCDC_SNS 是 DCDC 的反馈引脚，应连接到 DCDC 的电源输出。

当用户选择使用外部的电源替代片上的 DCDC 为系统电源域供电时，建议把 DCDC 的相关引脚 DCDC_IN、DCDC_LP 和 DCDC_SNS 通过 10k 欧姆电阻接地。

8.2.2 电源管理域线性稳压器 LDOPMC

LDOPMC 位于电源管理域，将 3.3V 外部电源转化为典型值为 1.1V 的供电，为电源管理域里除了 PDGO 模块以外的电路供电。PDGO 模块使用 3.3V 的 VPMC 供电。

LDOPMC 以 VPMC 引脚作为电源输入，需要在 VDD_PMCCAP 引脚上接一个 4.7uF 去耦电容。

8.2.3 OTP 线性稳压器 LDOOTP

LDOOTP 位于电源管理域，将 3.3V 外部电源转化为典型值为 2.5V 的供电，为 OTP 烧写电路提供电源。

LDOOTP 仅在 OTP 烧写操作时才应被使能，其他时间应保持关闭。

LDOOTP 以 VPMC 引脚作为电源输入，需要在 VDD_OTPCAP 引脚上接一个 4.7uF 去耦电容。

8.3 IO 供电

芯片的 GPIO 引脚被划为了多个组 (Bank)，每个 IO 组有独立的 IO 供电引脚 VIO_B*，GPIO 可支持 3.3V 电压。

各 IO 所使用的 IO 供电引脚可在章 20中查找。

电源管理域 GPIO PY* 固定使用 VPMC 供电，因此只能工作在 3V 模式。

8.4 电源引脚说明

电源引脚	电压范围	说明
VPMC	3.0V ~ 3.6V	LDOPMC 电源输入，LDOOTP 电源输入，为电源管理域模拟电路和数字电路 PDG 模块供电
VDD_PMCCAP	-	LDOPMC 的去耦电容连接引脚
VANA	3.0V ~ 3.6V	为 PLL，ADC，ACMP 模拟电路供电
VREFH	2.4V ~ VANA	ADC 高位参考电压 ACMP 内置 DAC 高位参考电压 DAC 高位参考电压
DCDC_IN	3.0V ~ 3.6V	DCDC 电源输入
DCDC_LP	-	DCDC 电源输出，可为系统电源域数字电路供电
DCDC_SNS	-	DCDC 电源反馈引脚，应与 DCDC_LP 连接在一起
VDD_SOC	0.925V ~ 1.26V	系统电源域数字电路供电引脚
VUSB	3.0V ~ 3.6V	USB 模拟电路供电，本产品上与 VIO_B00 短接。
VDD_USB	0.925V ~ 1.26V	USB 数字电路供电，通常与 VDD_SOC 短接
VIO_B00 ~ VIO_B01	3.3V	IO 供电引脚
DCDC_GND	-	DCDC 接地引脚
VSS	-	接地引脚
VREFL	-	ADC 低位参考电压，需接地

表 14: 电源和接地引脚说明

8.5 低功耗概览

本产品支持多种功耗模式，它们的功耗水平从高到低和唤醒时间从短到长依次为：

- 运行模式 (RUN)
- 等待模式 (WAIT)
- 停止模式 (STOP)
- 休眠模式 (STANBY)
- 关机模式 (SHUTDOWN)

运行模式下 CPU 正常执行指令，所有必要的功能模块正常工作。可关闭不需要的功能模块，CPU 时钟频率和供电电压可用软件调节。

等待模式下 CPU 核心时钟因 WFI 指令触发而关闭，其他功能模块保持运行模式下的状态，出现中断时 CPU 能够立即恢复运行并处理中断。

停止模式由 CPU 的 WFI 指令触发，通过预先配置，系统电源域内各子系统和模块，包括 CPU 自身的电源能够在 SYSCTL 模块的控制下灵活关闭或保持。

休眠模式下整个系统电源域都处于复位或掉电状态，该模式可以由 CPU 的 WFI 指令触发，也可以通过软件操作触发。

关机模式下系统电源域和电源管理域里除 PDGO 模块以外的部分都处于复位或掉电状态，仅保留 PDGO 工作，该模式可以通过软件操作触发。

特别说明:本产品的 CPU 本地存储器 ILM 和 DLM 支持 memory retention 功能,当系统电源域有电(VDD_SOC 有电),CPU0 子系统掉电(VDD_CPU 掉电)时,CPU 本地存储器 ILM 和 DLM 的 memory core 保持有电状态,存储的内容能够保留下来,方便未来的上电恢复。该功能无需配置,也不是一种低功耗模式,上述模式中只要满足系统电源域有电,CPU0 子系统掉电就可自动实现该功能。

8.6 电源管理功能相关 IO

电源管理域里的 PDGO 模块是芯片电源管理系统的基础,其 IO 能被用作电源管理的相关功能:

引脚	默认功能	说明
RESETN	RESETN	全局复位输入引脚,可以复位整个系统,也可以用于关机模式的退出,详见小节 11.4.2
WAKEUP	WAKEUP	电源唤醒输入引脚,可用于关机模式和其他低功耗模式的退出,详见小节 11.4.2

表 15: 电源管理相关 IO

9 系统复位

本章节介绍本产品的系统复位。

9.1 复位概览

复位系统支持对芯片的不同范围进行复位，如图 5所示：

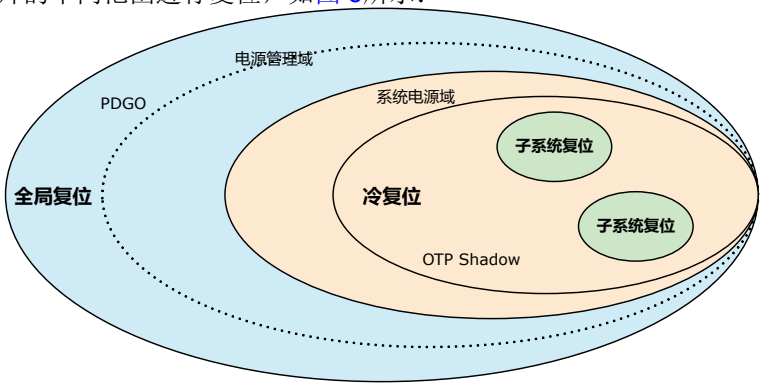


图 5: 复位系统

图中各种复位的范围与电源域结构有如下关系：

- 电源管理域里的 PDGO 模块会导致整个芯片的复位，即全局复位
- 电源管理域里除 PDGO 以外部分的复位不会影响 PDGO 模块，但会连带系统电源域进行复位
- 系统电源域的复位不会影响电源管理域，但会将各子系统进行复位

9.2 全局复位

全局复位会复位整个芯片，有两种情况会引起全局复位发生：

- VPMC 电源管脚的上电
- RESETN 管脚被外部电路拉低

RESETN 引脚是专用 IO。

RESETN 引脚默认产生的是全局复位，通过配置 BDGO 模块能够修改其复位范围，使电源管理域里的 PDGO 的状态不受影响，详见**??**。

VPMC 引脚的上电会使整个芯片复位。

9.3 系统电源域的复位

系统电源域的复位：

- 冷复位 (Cold Reset)：系统电源域保留 OTP 的影子寄存器的值，不复位 DEBUG 相关电路。

系统电源域的复位有多个复位源，包括：

- VPMC 供电电压低于约 2.7V
- VPMC 供电电压低于约 2.5V，此复位会同时复位 PDGO 区域
- 调试复位 (Debug Reset)：由外部 debugger 发出的复位请求
- 安全违例 (Security Violation)
- 看门狗超时
- 软件复位

除了由以上复位源触发的复位外，软件还能够通过控制 SYSCTL 模块单独复位某个子系统，详见 SYSCTL 相关章节。

10 时钟系统

本章节介绍本产品的时钟系统。

10.1 时钟系统概述

时钟系统主要由时钟源和功能时钟组成，其结构如图 图 6 所示。

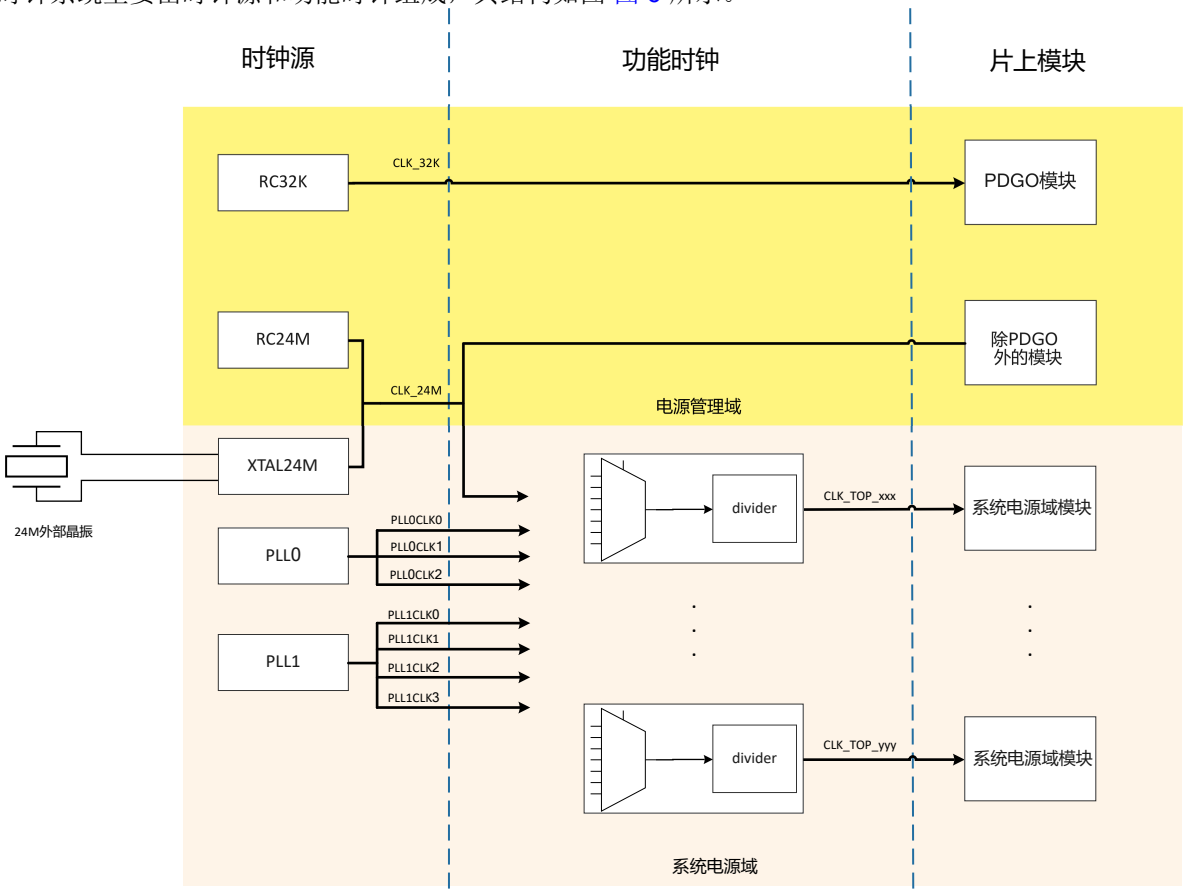


图 6: 时钟系统框图

时钟源包括外部晶振 XTAL、内部 RC 振荡器和锁相环 PLL 等，时钟源能够产生各种不同频率，不同精度的时钟。

功能时钟 CLK_TOP 是对时钟源进行选择和分频后的时钟。

10.2 时钟源

10.2.1 32KHz 时钟源 CLK_32K

产生 CLK_32K 时钟的模块有：

- RC32K - 内部 32KHz RC 振荡器

CLK_32K 直接为电源管理域的 PDGO 模块提供时钟。

10.2.2 24MHz 时钟源 CLK_24M

产生 24MHz 时钟源的模块有：

- XTAL24M - 24MHz 高速振荡器，支持 24MHz 外部晶振，也支持通过引脚从外部输入 24 MHz 有源时钟
- RC24M - 内部 24MHz RC 振荡器

XTAL24M 和 RC24M 共同构成 CLK_24M 时钟源。当 XTAL24M 使能并频率稳定时，CLK_24M 来自 XTAL24M 的输出时钟，当 XTAL24M 关闭或尚未稳定时，CLK_24M 来自 RC24M 的输出时钟。两个时钟之间的切换自动进行。

XTAL24M 默认会自动使能，其在低功耗模式下的开关控制请参考章 13。

CLK_24M 直接为电源管理域的模块提供时钟。

10.2.3 锁相环 PLL

产品内置 2 个 PLL 用以生成系统所需要的各种频率的时钟，其中 PLL0 有 3 路，PLL1 有 4 路独立分频的时钟输出，所以 PLL 时钟源共有 7 个。

PLL 默认使用 XTAL24M 时钟作为参考时钟。各 PLL 已预先设置了默认工作频率，如表 16所示：

PLL 的频率和分频系数的修改方法请参考章 14。

10.3 功能时钟

功能时钟能够根据配置在多个时钟源中做出选择并进行分频，之后为系统电源域的各模块提供运行时钟。

系统中有 8 个时钟源，所以每个功能时钟有一个 8 选 1 的多路选择器进行时钟源选择，选择值与时钟源的对应关系如表 16所示。

选择值	时钟源	默认频率	说明
0	CLK_24M	24MHz	24MHz 时钟源
1	PLL0CLK0	720MHz	默认作为处理器的时钟源
2	PLL0CLK1	600MHz	
3	PLL0CLK2	400MHz	
4	PLL1CLK0	800MHz	
5	PLL1CLK1	666MHz	
6	PLL1CLK2	500MHz	
7	PLL1CLK3	266MHz	

表 16: 时钟源选择及默认频率

每个功能时钟可单独设置分频系数，对选定的时钟源进行分频，分频系数范围为从 1 到 256 的任意整数。

为便于使用，各功能时钟已预先设置了时钟源选择和分频系数，具体信息见表 17。时钟源选择和分频系数的修改方式请参考节 13.2。

功能时钟名称	默认时钟源	默认分频系数	默认频率	应用
CLK_TOP_HART0	PLL0CLK0	2	360MHz	CPU0 核心/ ILM0/DLM0/FGPIO0
CLK_TOP_MCHTMR0	CLK24M	1	24MHz	MCHTMR0 计时

功能时钟名称	默认时钟源	默认分频系数	默认频率	应用
CLK_TOP_AHB	CLK_TOP_HART0	2	180MHz	外设总线/HDMA/AHB SRAM/ 电机系统/ACMP/GPIO
CLK_TOP_CAN0	PLL1CLK0	10	80MHz	CAN0 接口
CLK_TOP_CAN1	PLL1CLK0	10	80MHz	CAN1 接口
CLK_TOP_CAN2	PLL1CLK0	10	80MHz	CAN2 接口
CLK_TOP_CAN3	PLL1CLK0	10	80MHz	CAN3 接口
CLK_TOP_LIN0	PLL1CLK0	10	80MHz	LIN0 接口
CLK_TOP_LIN1	PLL1CLK0	10	80MHz	LIN1 接口
CLK_TOP_LIN2	PLL1CLK0	10	80MHz	LIN2 接口
CLK_TOP_LIN3	PLL1CLK0	10	80MHz	LIN3 接口
CLK_TOP_GPTMR0	PLL1CLK0	8	100MHz	GPTMR0 计时
CLK_TOP_GPTMR1	PLL1CLK0	8	100MHz	GPTMR1 计时
CLK_TOP_GPTMR2	PLL1CLK0	8	100MHz	GPTMR2 计时
CLK_TOP_GPTMR3	PLL1CLK0	8	100MHz	GPTMR3 计时
CLK_TOP_I2C0	PLL1CLK0	10	80MHz	I2C0 接口
CLK_TOP_I2C1	PLL1CLK0	10	80MHz	I2C1 接口
CLK_TOP_I2C2	PLL1CLK0	10	80MHz	I2C2 接口
CLK_TOP_I2C3	PLL1CLK0	10	80MHz	I2C3 接口
CLK_TOP_SPI0	PLL1CLK0	10	80MHz	SPI0 接口
CLK_TOP_SPI1	PLL1CLK0	10	80MHz	SPI1 接口
CLK_TOP_SPI2	PLL1CLK0	10	80MHz	SPI2 接口
CLK_TOP_SPI3	PLL1CLK0	10	80MHz	SPI3 接口
CLK_TOP_UART0	PLL1CLK0	10	80MHz	UART0 接口
CLK_TOP_UART1	PLL1CLK0	10	80MHz	UART1 接口
CLK_TOP_UART2	PLL1CLK0	10	80MHz	UART2 接口
CLK_TOP_UART3	PLL1CLK0	10	80MHz	UART3 接口
CLK_TOP_UART4	PLL1CLK0	10	80MHz	UART4 接口
CLK_TOP_UART5	PLL1CLK0	10	80MHz	UART5 接口
CLK_TOP_UART6	PLL1CLK0	10	80MHz	UART6 接口
CLK_TOP_UART7	PLL1CLK0	10	80MHz	UART7 接口
CLK_TOP_XPI0	PLL1CLK1	2	333MHz*	XPI0 接口
CLK_TOP_ANA0	PLL1CLK0	4	200MHz	ADC0 备选时钟
CLK_TOP_ANA1	PLL1CLK0	4	200MHz	ADC1 备选时钟
CLK_TOP_ANA2	PLL1CLK0	4	200MHz	DAC0 备选时钟
CLK_TOP_ANA3	PLL1CLK0	4	200MHz	DAC1 备选时钟
CLK_TOP_REF0	PLL1CLK2	10	50MHz	REF0 备选时钟
CLK_TOP_REF1	PLL1CLK2	10	50MHz	REF1 备选时钟
CLK_TOP_ADC0	CLK_TOP_AHB		180MHz	ADC0
CLK_TOP_ADC1	CLK_TOP_AHB		180MHz	ADC1

功能时钟名称	默认时钟源	默认分频系数	默认频率	应用
CLK_TOP_DAC0	CLK_TOP_AHB		180MHz	DAC0
CLK_TOP_DAC1	CLK_TOP_AHB		180MHz	DAC1

表 17: 功能时钟汇总

* 实际频率与启动配置有关。

注意，ADC 和 DAC 的功能时钟采用两级多路选择结构，用以支持任意个 ADC 或任意个 DAC 同步工作或异步工作。

ADC 和 DAC 的功能时钟结构如图 7 所示。

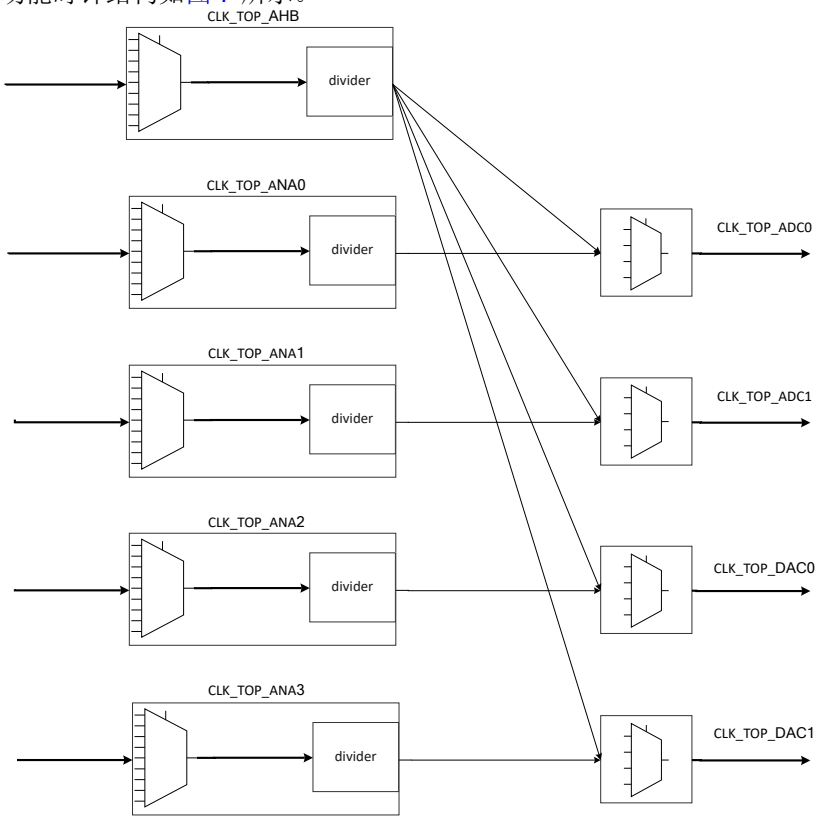


图 7: ADC 功能时钟结构

CLK_TOP_ADCx 和 CLK_TOP_DACx 上的多路选择器所选定的前一级功能时钟见表 18。

选择值	**CLK_TOP_ADC0 时钟选择**
0	CLK_TOP_AHB
1	CLK_TOP_ANA0
选择值	**CLK_TOP_ADC1 时钟选择**
0	CLK_TOP_AHB
1	CLK_TOP_ANA1
选择值	**CLK_TOP_DAC0 时钟选择**
0	CLK_TOP_AHB

选择值	CLK_TOP_ADC0 时钟选择
1	CLK_TOP_ANA2
选择值	**CLK_TOP_DAC1 时钟选择**
0	CLK_TOP_AHB
1	CLK_TOP_ANA3

表 18: 两级功能时钟选择

CLK_TOP_ADCx 和 CLK_TOP_DACx 默认依次选择 CLK_TOP_AHB。

该结构使任意个 ADC 或 DAC 能够以相同的时钟工作或者各自工作在独立的时钟频率。

10.4 直接使用时钟源的模块

系统中一部分模块没有使用功能时钟而是直接连接时钟源，其中有电源管理域的全部模块，也包括系统电源域的部分计时模块，在表 19列出。

模块	时钟源
电源管理域模块	
PDGO	CLK_32K
PTMR	CLK_24M
PWDG	CLK_24M 和 CLK_32K
PUART	CLK_24M
PGPIO	CLK_24M
系统电源域模块	
WDG0~WDG1	CLK_32K 和 CLK_TOP_AHB
USB0	CLK_24M

表 19: 模块时钟列表

11 电源管理域开关机模块 PDGO

本章节描述了电源管理域中的开关机模块 PDGO 的主要功能和使用。

11.1 主要特性

PDGO 由 3.3V 的 VPMC 供电，主要功能包括：

- 32KHz 时钟源管理
- 全局复位引脚 RESETN 控制
- 电源管理域开关机控制

11.2 CLK_32K 时钟源管理

电源管理域中有一个 IRC32K 的时钟振荡器模块，系统在电源管理域上电后，PDGO 会使用 IRC32K 时钟做为 CLK_32K 时钟源进行工作。

可通过 fuse 进行 IRC32K 的精度校准，用户也可以通过设置 DGO_RC32K_CFG 寄存器对 IRC32K 的精度进行校准。

11.3 全局复位引脚 RESETN 控制

全局复位引脚 RESETN 为专用引脚，用户可以通过把 DGO_CTR0 寄存器的 RETENTION 位置 1，这样，引脚复位就只复位系统电源域和电源管理域中除 PDGO 以外的部分，不影响 PDGO 的状态。

另外，全局复位引脚也可以作为退出关机模式的唤醒源，详见 小节 11.4.2。

11.4 电源管理域开关机控制

PDGO 能够控制关机模式的进入和退出。关机模式下整个系统电源域，以及电源管理域里除 PDGO 模块以外的部分都处于复位或掉电状态，仅保留 PDGO 工作。改模式下系统具有极低功耗。

11.4.1 关机模式的进入

进入关机模式的方式为软件操作触发，向 DGO_TURNOFF 寄存器的 COUNTER 位写入一个非零计数值，则计数器每个 24M 时钟周期减一，直到计数器为 1 时，系统进入关机模式。进入关机模式后，该寄存器会被复位。

11.4.2 关机模式的退出

退出关机模式的方式有四种：

- 软件倒计时单次唤醒
- 软件倒计时多次唤醒
- 唤醒引脚 WAKEUP
- 复位引脚 RESETN

软件倒计时单次唤醒：用户需要在关机操作之前，把 DGO_CTRL1 寄存器的 AUTO_SYS_WAKEUP 位置 0，并把 DGO_CTRL3 寄存器的 WAKEUP_COUNTER 位写一个非零计数值。计数器从 WAKEUP_COUNTER 为非零值的时刻开始每个 32KHz 时钟周期减一，直到计数器为 2 时，产生一个唤醒事件，同时硬件自动清空 DGO_CTRL3 寄存器的 WAKEUP_COUNTER 位。

需要注意，唤醒倒计时并不是从关机之后才开始倒计时，所以用户需要控制好关机倒计时和唤醒倒计时的时

间差。

软件倒计时多次唤醒：用户需要在关机操作之前，把 DGO_CTRL1 寄存器的 AUTO_SYS_WAKEUP 位置 1，并把 DGO_CTRL3 寄存器的 WAKEUP_COUNTER 位写一个非零计数值。计数器从 WAKEUP_COUNTER 为非零值的时刻开始每个 32KHz 时钟周期减一，直到计数器为 2 时，产生一个唤醒事件。计数器为 0 时，重新从 WAKEUP_COUNTER 开始每个 32KHz 时钟周期减一，直到计数器为 2 时，再次产生一个唤醒事件。循环往复，直到用户手动清零 DGO_CTRL1 寄存器的 AUTO_SYS_WAKEUP 位，停止产生唤醒事件，同时硬件自动清空 DGO_CTRL3 寄存器的 WAKEUP_COUNTER 位。

唤醒引脚 WAKEUP：当系统处于关机模式时，唤醒引脚 WAKEUP 上检测到有效的高电平，可以产生一个唤醒事件，退出关机模式。

复位引脚 RESETN：当系统处于关机模式时，唤醒引脚 RESETN 上检测到有效的低电平，可以产生一个唤醒事件，退出关机模式。

注意：只有当系统进入关机模式后，唤醒事件才会生效。

11.5 PDGO 寄存器说明

PDGO 的寄存器列表如下：

PDGO base address: 0xF4134000

地址偏移	名称	描述	复位值
0x0000	DGO_TURNOFF	关机控制	0x00000000
0x0004	DGO_RC32K_CFG	RC32K 时钟	0x00000000
0x0600	DGO_GPR00	通用寄存器 0	0x00000000
0x0604	DGO_GPR01	通用寄存器 1	0x00000000
0x0608	DGO_GPR02	通用寄存器 2	0x00000000
0x060C	DGO_GPR03	通用寄存器 3	0x00000000
0x0700	DGO_CTR0	控制寄存器 0	0x00000000
0x0704	DGO_CTR1	控制寄存器 1	0x00000000
0x0708	DGO_CTR2	控制寄存器 2	0x00000000
0x070C	DGO_CTR3	控制寄存器 3	0x00000000
0x0710	DGO_CTR4	控制寄存器 4	0x00000000

表 20: PDGO 寄存器列表

11.6 PDGO 寄存器详细信息

PDGO 的寄存器详细说明如下：

11.6.1 DGO_TURNOFF (0x0)

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

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

DGO_TURNOFF [31:0]

位域	名称	描述
31-0	COUNTER	关机计数器，计数器每一个 24m 时钟周期减一，计数到 0 停止，关机发生在计数器值是 1 的时刻。进入关机模式后，该位被清 0

DGO_TURNOFF 位域

11.6.2 DGO_RC32K_CFG (0x4)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IRC_TRIMMED	RSVD							CAPEX7_TRIM	CAPEX6_TRIM	RSVD													CAP_TRIM								
RW	N/A							RW	RW	N/A													RW								
0	x	x	x	x	x	x	x	0	0	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0

DGO_RC32K_CFG [31:0]

位域	名称	描述
31	IRC_TRIMMED	已校准，此位在熔丝校准后自动置位，并停止加载熔丝值，写 0 清 0: 32K 未校准 1: 32K 已校准
23	CAPEX7_TRIM	32K 校准值，冗余位 7
22	CAPEX6_TRIM	32K 校准值，冗余位 6
8-0	CAP_TRIM	32K 校准值

DGO_RC32K_CFG 位域

11.6.3 DGO_GPR00 (0x600)

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

DGO_GPR00 [31:0]

位域	名称	描述
31-0	GPR	通用寄存器 0

位域	名称	描述

DGO_GPR00 位域

11.6.4 DGO_GPR01 (0x604)

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

DGO_GPR01 [31:0]

位域	名称	描述
31-0	GPR	通用寄存器 1

DGO_GPR01 位域

11.6.5 DGO_GPR02 (0x608)

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

DGO_GPR02 [31:0]

位域	名称	描述
31-0	GPR	通用寄存器 2

DGO_GPR02 位域

11.6.6 DGO_GPR03 (0x60C)

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

DGO_GPR03 [31:0]

位域	名称	描述
31-0	GPR	通用寄存器 3

DGO_GPR03 位域

11.6.7 DGO_CTR0 (0x700)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD															RETENTION	RSVD															
N/A															RW	N/A															
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x

DGO_CTR0 [31:0]

位域	名称	描述
16	RETENTION	DGO 寄存器状态保持

DGO_CTR0 位域

11.6.8 DGO_CTR1 (0x704)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AOTO_SYS_WAKEUP	RSVD														WAKEUP_EN	RSVD															PIN_WAKEUP_STATUS
RW	N/A														RW	N/A															RO
0	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0

DGO_CTR1 [31:0]

位域	名称	描述
31	AOTO_SYS_WAKEUP	软件唤醒：0: 单次唤醒；1：持续唤醒
16	WAKEUP_EN	允许唤醒按键或软件唤醒
0	PIN_WAKEUP_STATUS	唤醒按键状态

DGO_CTR1 位域

11.6.9 DGO_CTR2 (0x708)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD							RESETN_PULLUP_DISABLE	RSVD							WAKEUP_PULLDN_DISABLE	RSVD															
N/A							RW	N/A							RW	N/A															
x	x	x	x	x	x	x	0	x	x	x	x	x	x	x	0	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x

DGO_CTR2 [31:0]

位域	名称	描述
24	RESETN_PULLUP_DISABLE	关闭复位按键上拉，保持之前的状态
16	WAKEUP_PULLDN_DISABLE	关闭唤醒按键下拉，保持之前的状态

DGO_CTR2 位域

11.6.10 DGO_CTR3 (0x70C)

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

DGO_CTR3 [31:0]

位域	名称	描述
31-0	WAKEUP_COUNTER	软件唤醒计数，计数器每一个 32K 时钟周期减一，计数到 0 即停止，唤醒发生在计数器值是 2 的时。单次唤醒模式，唤醒发生后，该位会自动清 0。持续唤醒模式，该位在持续唤醒模式关闭后自动清 0

DGO_CTR3 位域

11.6.11 DGO_CTR4 (0x710)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																														BANDGAP_LESS_POWER	BANDGAP_LP_MODE
N/A																														RW	RW
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0

DGO_CTR4 [31:0]

位域	名称	描述
1	BANDGAP_LESS_POWER	基准源省电模式 0: 正常模式 1: 省电模式
0	BANDGAP_LP_MODE	基准源低功耗模式 0: 正常模式 1: 低功耗模式

DGO_CTR4 位域

12 电源管理域配置模块 PCFG

本章节描述了电源域管理器 PCFG 的主要特性。

12.1 特性总结

PCFG 是电源管理域内的电源、时钟和模拟类外设的集成控制模块。用户可以通过 PCFG 实现：

- 线性稳压器 LDOPMC 控制
- 线性稳压器 LDOOTP 控制
- 开关电源 DCDC 配置
- 电源管理域门控时钟管理

12.2 功能描述

本章节描述 PCFG 的功能。

12.2.1 线性稳压器 LDOPMC 配置

线性稳压器 LDOPMC 是电源管理域除 PDGO 以外的电源。默认使能, 支持软件强制关闭或关机模式下硬件自动关闭。

12.2.2 线性稳压器 LDOOTP 配置

线性稳压器 LDOOTP 是片上一次性可编程存储器 OTP 的电源。对 OTP 进行烧写前，必须先将 LDO2P5 寄存器的 ENABLE 位使能，并等待 READY 位有效。在 OTP 烧写结束后必须清除 ENABLE 位以关闭该 LDO。

12.2.3 开关电源 DCDC 配置

开关电源 DCDC 可以用来为系统电源域提供电源。DCDC 的输出电压通过 DCDC_MODE 寄存器的 VOLT 位设置，默认电压为 1.15V。

DCDC 默认开启，当芯片使用外部电源直接为 VDD_SOC 供电时，应将 DCDC_MODE 寄存器的 MODE 位设为 0 以关闭 DCDC。

当系统进入休眠模式 (详见节 15.4) 时,DCDC 也可随之关闭或者进入低功耗状态,相关控制在 POWER_TRAP 寄存器的 TRAP 和 RETENTION，其组合关系如表 21 所示：

TRAP 位	RETENTION	设置
0	-	系统休眠时，DCDC 工作状态不变
1	0	系统休眠时，DCDC 关闭
1	1	系统休眠时，DCDC 进入低功耗模式

表 21: DCDC 低功耗状态控制

DCDC 进入低功耗模式时，输出电压在 DCDC_LPMODE 寄存器的 STBY_VOLT 中设置。可以根据应用设置一个较低的电压，例如使用系统电源域的存储器内容保持电压值。

DCDC 支持获取当前电流强度，在开启电流测量功能后，可从 DCDC_CURRENT 寄存器的 LEVEL 读取。

12.2.4 电源管理域模块的时钟门控

用户可以通过 SCG_CTRL 寄存器打开或关闭电源管理域部分模块的时钟。

- PGPIO
- PIOC
- PTMR
- PWDG
- PUART

12.2.5 系统电源域唤醒

电源管理域中的唤醒源有：

- WAKEUP 输入引脚
- PTMR 中断
- PUART 中断
- PWDG 中断
- PGPIO 中断

可通过 PCFG 模块的 WAKE_MASK 寄存器允许以上唤醒源唤醒系统，而唤醒源状态可在 WAKE_CAUSE 寄存器中读取和清零。

12.2.6 电源管理域复位控制模块 PPOR

PPOR 管理系统电源域的复位。

用户可以通过 RESET_ENABLE 寄存器来打开或者关闭这些复位。

用户可以通过 RESET_FLAG 寄存器来查看最近一次复位的来源。

RESET_COLD 寄存器用来控制每个复位源产生冷复位或者温复位。

用户可以通过 SOFTWARE_RESET 的 COUNTER 位域配置软件复位倒计时计数器。用户可以任意配置 COUNTER，当这个计数器倒计时到 2 时，生成软件复位。对 COUNTER 写入 0 可以关闭软件复位，并取消正在进行中的软件复位倒计时。

12.3 PCFG 寄存器说明

PCFG 的寄存器列表如下：

PCFG base address: 0xF4104000

地址偏移	名称	描述	复位值
0x0000	BANDGAP	基准源	0x00101010
0x0004	LDO1P1	1.1V 稳压源	0x0001044C
0x0008	LDO2P5	2.5V 稳压源	0x000009C4
0x0010	DCDC_MODE	DCDC 工作模式	0x0001047E
0x0014	DCDC_LPMODE	DCDC 低功耗设置	0x00000384
0x0018	DCDC_PROT	DCDC 保护	0x00000010
0x001C	DCDC_CURRENT	DCDC 电流估计	0x00000000
0x0020	DCDC_ADVMODE	DCDC 高级设置	0x03120040

地址偏移	名称	描述	复位值
0x0024	DCDC_ADVPARAM	DCDC 高级参数	0x00006E1C
0x0028	DCDC_MISC	DCDC 杂项设置	0x00070100
0x002C	DCDC_DEBUG	DCDC 调试设置	0x00005DBF
0x0030	DCDC_START_TIME	DCDC 启动时间	0x0001193F
0x0034	DCDC_RESUME_TIME	DCDC 恢复时间	0x00008C9F
0x0040	POWER_TRAP	DCDC 低功耗设置	0x00000000
0x0044	WAKE_CAUSE	唤醒原因	0x00000000
0x0048	WAKE_MASK	唤醒掩蔽	0x00000000
0x004C	SCG_CTRL	保持外设时钟开关	0xFFFFFFFF
0x0060	RC24M	RC24M 设置	0x00000310
0x0064	RC24M_TRACK	RC24M 跟踪设置	0x00000000
0x0068	TRACK_TARGET	RC24M 跟踪频率	0x00000000
0x006C	STATUS	RC24M 跟踪状态	0x00000000

表 22: PCFG 寄存器列表

12.4 PCFG 寄存器详细信息

PCFG 的寄存器详细说明如下：

12.4.1 BANDGAP (0x0)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VBG_TRIMMED	RSVD										VBG_1P0_TRIM					RSVD			VBG_P65_TRIM					RSVD			VBG_P50_TRIM				
RW	N/A										RW					N/A			RW					N/A			RW				
0	x	x	x	x	x	x	x	x	x	x	1	0	0	0	0	x	x	x	1	0	0	0	0	x	x	x	1	0	0	0	0

BANDGAP [31:0]

位域	名称	描述
31	VBG_TRIMMED	已校准，此位在熔丝校准后自动置位，并停止加载熔丝值，写 0 清 0: 基准源未校准 1: 基准源已校准
20-16	VBG_1P0_TRIM	1.0V 基准校准值
12-8	VBG_P65_TRIM	0.65V 基准校准值
4-0	VBG_P50_TRIM	0.50 基准校准值

位域	名称	描述

BANDGAP 位域

12.4.2 LDO1P1 (0x4)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD															ENABLE	RSVD				VOLT											
N/A															RW	N/A				RW											
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	1	x	x	x	x	0	1	0	0	0	1	0	0	1	1	0	0

LDO1P1 [31:0]

位域	名称	描述
16	ENABLE	稳压电源开关 0: 关闭 1: 打开
11-0	VOLT	输出电压，单位毫伏。允许输入范围 700mV-1320mV。硬件自动选择不低于目标值的最小电压。

LDO1P1 位域

12.4.3 LDO2P5 (0x8)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD			READY	RSVD											ENABLE	RSVD				VOLT											
N/A			RO	N/A											RW	N/A				RW											
x	x	x	0	x	x	x	x	x	x	x	x	x	x	x	0	x	x	x	x	1	0	0	1	1	1	0	0	0	1	0	0

LDO2P5 [31:0]

位域	名称	描述
28	READY	2.5V 电源以稳定 0：关闭或尚未稳定 1：已稳定
16	ENABLE	稳压电源开关 0: 关闭 1: 打开
11-0	VOLT	输出电压，单位毫伏。允许输入范围 2125mV-2900mV。硬件自动选择不低于目标值的最小电压。

LDO2P5 位域

12.4.4 DCDC_MODE (0x10)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD			READY	RSVD									MODE			RSVD				VOLT											
N/A			RO	N/A									RW			N/A				RW											
x	x	x	0	x	x	x	x	x	x	x	x	x	0	0	1	x	x	x	x	0	1	0	0	0	1	1	1	1	1	1	0

DCDC_MODE [31:0]

位域	名称	描述
28	READY	DCDC 稳定标志 0: DCDC 尚未稳定 1: DCDC 已稳定
18-16	MODE	DCDC 工作模式 XX0: 关闭 001: 基本模式 011: 通用模式 111: 专家模式
11-0	VOLT	输出电压，单位毫伏。允许输入范围 600mV-1375mV。硬件自动选择不低于目标值的最小电压。

DCDC_MODE 位域

12.4.5 DCDC_LPMODE (0x14)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																				STBY_VOLT											
N/A																				RW											
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	1	1	1	0	0	0	0	1	0	0

DCDC_LPMODE [31:0]

位域	名称	描述
11-0	STBY_VOLT	低功耗模式输出电压，单位毫伏。允许输入范围 600mV-1375mV。硬件自动选择不低于目标值的最小电压。

DCDC_LPMODE 位域

12.4.6 DCDC_PROT (0x18)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD			ILIMIT_LP	RSVD			OVERLOAD_LP	RSVD							POWER_LOSS_FLAG	DISABLE_OVERVOLTAGE	RSVD						OVERVOLT_FLAG	DISABLE_SHORT	RSVD		SHORT_CURRENT	RSVD			SHORT_FLAG
N/A			RW	N/A			RO	N/A							RO	RW	N/A						RO	RW	N/A		RW	N/A			RO
x	x	x	0	x	x	x	0	x	x	x	x	x	x	x	0	0	x	x	x	x	x	x	0	0	x	x	1	x	x	x	0

DCDC_PROT [31:0]

位域	名称	描述
28	ILIMIT_LP	低功耗模式负载电流 0:250mA 1:200mA
24	OVERLOAD_LP	低功耗模式下过流 0: 电流低于设定值 1: 电流超过设定值
16	POWER_LOSS_FLAG	失电标志 0: 电源输入正常 1: 电源输入过低
15	DISABLE_OVERVOLTAGE	禁止输出过压保护 0: 允许保护, DCDC 过压后自动关闭 1: 禁止保护, DCDC 过压后正常工作
8	OVERVOLT_FLAG	输出过压标志 0: 输出正常 1: 输出过高
7	DISABLE_SHORT	禁止输出短路保护 0: 允许保护，DCDC 短路后自动关闭 1: 禁止保护，DCDC 短路后保持工作
4	SHORT_CURRENT	短路电流 0: 2.0A, 1: 1.3A
0	SHORT_FLAG	短路标志 0: 电流低于短路电流 1: 检测到短路

DCDC_PROT 位域

12.4.7 DCDC_CURRENT (0x1C)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																ESTI_EN	RSVD						VALID	RSVD			LEVEL				
N/A																RW	N/A						RO	N/A			RO				
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	x	x	x	x	x	x	0	x	x	x	0	0	0	0	0

DCDC_CURRENT [31:0]

位域	名称	描述
15	ESTI_EN	使能电流估计
8	VALID	电流估计有效 0: 无估计数据 1: 估计值有效
4-0	LEVEL	DCDC 估计电流, 电流值位 LEVEL * 50mA

DCDC_CURRENT 位域

12.4.8 DCDC_ADVMODE (0x20)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD					EN_RCSCALE			RSVD		DC_C		DC_R				RSVD									EN_FF_DET	EN_FF_LOOP	EN_AUTOLP	EN_DCM_EXIT	EN_SKIP	EN_IDLE	EN_DCM
N/A					RW			N/A		RW		RW				N/A									RW	RW	RW	RW	RW	RW	RW
x	x	x	x	x	0	1	1	x	x	0	1	0	0	1	0	x	x	x	x	x	x	x	x	x	1	0	0	0	0	0	0

DCDC_ADVMODE [31:0]

位域	名称	描述
26-24	EN_RCSCALE	Enable RC scale
21-20	DC_C	Loop C number
19-16	DC_R	Loop R number
6	EN_FF_DET	enable feed forward detect 0: feed forward detect is disabled 1: feed forward detect is enabled
5	EN_FF_LOOP	enable feed forward loop 0: feed forward loop is disabled 1: feed forward loop is enabled
4	EN_AUTOLP	enable auto enter low power mode 0: do not enter low power mode 1: enter low power mode if current is detected low

位域	名称	描述
3	EN_DCM_EXIT	avoid over voltage 0: stay in DCM mode when voltage excess 1: change to CCM mode when voltage excess
2	EN_SKIP	enable skip on narrow pulse 0: do not skip narrow pulse 1: skip narrow pulse
1	EN_IDLE	enable skip when voltage is higher than threshold 0: do not skip 1: skip if voltage is excess
0	EN_DCM	DCM mode 0: CCM mode 1: DCM mode

DCDC_ADVMODE 位域

12.4.9 DCDC_ADVPARAM (0x24)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																	MIN_DUT							RSVD	MAX_DUT						
N/A																	RW							N/A	RW						
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	1	1	0	1	1	1	0	x	0	0	1	1	1	0	0

DCDC_ADVPARAM [31:0]

位域	名称	描述
14-8	MIN_DUT	minimum duty cycle
6-0	MAX_DUT	maximum duty cycle

DCDC_ADVPARAM 位域

12.4.10 DCDC_MISC (0x28)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD			EN_HYST	RSVD		HYST_SIGN	HYST_THRS	RSVD			RC_SCALE	RSVD	DC_FF			RSVD						OL_THRE		RSVD			OL_HYST	RSVD	DELAY	CLK_SEL	EN_STEP
N/A			RW	N/A		RW	RW	N/A			RW	N/A	RW			N/A						RW		N/A			RW	N/A	RW	RW	RW
x	x	x	0	x	x	0	0	x	x	x	0	x	1	1	1	x	x	x	x	x	x	0	1	x	x	x	0	x	0	0	0

DCDC_MISC [31:0]

位域	名称	描述
28	EN_HYST	hysteres enable
25	HYST_SIGN	hysteres sign
24	HYST_THRS	hysteres threshold
20	RC_SCALE	Loop RC scale threshold
18-16	DC_FF	Loop feed forward number
9-8	OL_THRE	overload for threshold for lod power mode
4	OL_HYST	current hysteres range 0: 12.5mV 1: 25mV
2	DELAY	enable delay 0: delay disabled, 1: delay enabled
1	CLK_SEL	clock selection 0: select DCDC internal oscillator 1: select RC24M oscillator
0	EN_STEP	enable stepping in voltage change 0: stepping disabled, 1: steping enabled

DCDC_MISC 位域

12.4.11 DCDC_DEBUG (0x2C)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD												UPDATE_TIME																			
N/A												RW																			
x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	1	0	1	1	1	0	1	1	0	1	1	1	1	1	1

DCDC_DEBUG [31:0]

位域	名称	描述
19-0	UPDATE_TIME	DCDC 电压调整时间，以 24M 时钟计数，缺省值 1mS

DCDC_DEBUG 位域

12.4.12 DCDC_START_TIME (0x30)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD												START_TIME																			
N/A												RW																			
x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	1	0	0	0	1	1	0	0	1	0	0	1	1	1	1	1	1

DCDC_START_TIME [31:0]

位域	名称	描述
19-0	START_TIME	DCDC 电压启动时间，以 24M 时钟计数，缺省值 3mS

DCDC_START_TIME 位域

12.4.13 DCDC_RESUME_TIME (0x34)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD												RESUME_TIME																			
N/A												RW																			
x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	1	0	0	0	1	1	0	0	1	0	0	1	1	1	1	1

DCDC_RESUME_TIME [31:0]

位域	名称	描述
19-0	RESUME_TIME	DCDC 低功耗恢复时间，以 24M 时钟计数，缺省值 1.5mS

DCDC_RESUME_TIME 位域

12.4.14 POWER_TRAP (0x40)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TRIGGERED	RSVD														RETENTION	RSVD															TRAP
RW	N/A														RW	N/A															RW
0	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0

POWER_TRAP [31:0]

位域	名称	描述
31	TRIGGERED	DCDC 进入低功耗，若 DCDC 计入低功耗状态，此位会置位，此位写 1 清零 0：DCDC 未进入过低功耗模式 1: DCDC 进入过低功耗模式
16	RETENTION	DCDC 进入低功耗模式，降低电压保持 SOCSRAM 的内容 0: 关闭 DCDC 1: 降低电压
0	TRAP	允许 DCDC 在低功耗模式下关闭或降低电压，该位在 DCDC 关闭或降低电压后自动清零 0: DCDC 在低功耗模式下保持工作状态 1: DCDC 在低功耗模式下关闭或降低电压

POWER_TRAP 位域

12.4.15 WAKE_CAUSE (0x44)

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

WAKE_CAUSE [31:0]

位域	名称	描述
31-0	CAUSE	可从 DCDC 低功耗模式唤醒的唤醒状态，每位代表 1 个唤醒源，写 1 清零 0: 唤醒源未唤醒过系统 1: 唤醒源唤醒过系统 bit 0: SOC 请求 bit 7: UART 中断 bit 8: TMR 中断 bit 9: WDG 中断 bit10: PGPIO 中断 bit31: 唤醒按键

WAKE_CAUSE 位域

12.4.16 WAKE_MASK (0x48)

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

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

WAKE_MASK [31:0]

位域	名称	描述
31-0	MASK	禁止唤醒源唤醒系统 0: 允许事件唤醒系统 1: 禁止事件唤醒系统 bit 0: SOC 请求 bit 7: UART 中断 bit 8: TMR 中断 bit 9: WDG 中断 bit10: PGPIO 中断 bit31: 唤醒按键

WAKE_MASK 位域

12.4.17 SCG_CTRL (0x4C)

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

SCG_CTRL [31:0]

位域	名称	描述
31-0	SCG	电源管理域外设时钟管理，每两位代表一个外设 00,01: reserved 10: 保持关闭 11: 保持运行 bit6-7:GPIO bit8-9:IO 管理 bit10-11: TMR bit12-13:WDG bit14-15:UART

SCG_CTRL 位域

12.4.18 RC24M (0x60)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RC_TRIMMED	RSVD																				TRIM_C			RSVD			TRIM_F				
RW	N/A																				RW			N/A			RW				
0	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	1	1	x	x	x	1	0	0	0	0

RC24M [31:0]

位域	名称	描述
31	RC_TRIMMED	RC24M 校准标志，若熔丝有校准值则该位在校准后置 1，写 0 清零该位 0: RC24M 未校准 1: RC24M 已校准
10-8	TRIM_C	RC24M 细调, 数值越大频率越高
4-0	TRIM_F	RC24M 粗调, 数值越大频率越高

RC24M 位域

12.4.19 RC24M_TRACK (0x64)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD															SEL24M	RSVD											RETURN	RSVD			TRACK
N/A															RW	N/A											RW	N/A			RW
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	x	x	x	x	x	x	x	x	x	x	x	0	x	x	x	0

RC24M_TRACK [31:0]

位域	名称	描述
16	SEL24M	选择外部参考 0: 32K OSC 1: 24M OSC
4	RETURN	外部时钟源停止时行为 0: 保持 1: 返回缺省值
0	TRACK	自动跟踪 0: RC24M 自由振荡 1: 跟踪外部时钟源

RC24M_TRACK 位域

12.4.20 TRACK_TARGET (0x68)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PRE_DIV																TARGET															
RW																RW															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

TRACK_TARGET [31:0]

位域	名称	描述
31-16	PRE_DIV	参考频率预分频数
15-0	TARGET	目标频率，预分频倍数

TRACK_TARGET 位域

12.4.21 STATUS (0x6C)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD											SEL32K	RSVD			SEL24M	EN_TRIM	RSVD				TRIM_C			RSVD			TRIM_F			
N/A											RO	N/A			RO	RO	N/A				RO			N/A			RO			
x	x	x	x	x	x	x	x	x	x	x	0	x	x	x	0	0	x	x	x	x	0	0	0	x	x	x	0	0	0	0	0

STATUS [31:0]

位域	名称	描述
20	SEL32K	参考 OSC 32K 0: 否 1: 是
16	SEL24M	参考 OSC 24M 0: 否 1: 是
15	EN_TRIM	缺省值可用 0: 不可用 1: 可用
10-8	TRIM_C	缺省粗调值
4-0	TRIM_F	缺省细调值

STATUS 位域

12.5 PPOR 寄存器说明

PPOR 的寄存器列表如下：

PPOR base address: 0xF4100000

地址偏移	名称	描述	复位值
0x0000	RESET_FLAG	复位源标志	0x00000000
0x0004	RESET_STATUS	复位源状态	0x00000000
0x0008	RESET_HOLD	复位保持	0x00000000
0x000C	RESET_ENABLE	复位使能	0xFFFFFFFF
0x0010	RESET_TYPE	复位类型	0x00000000
0x001C	SOFTWARE_RESET	软复位计数器	0x00000000

表 23: PPOR 寄存器列表

12.6 PPOR 寄存器详细信息

PPOR 的寄存器详细说明如下：

12.6.1 RESET_FLAG (0x0)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FLAG																															
W1C																															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

RESET_FLAG [31:0]

位域	名称	描述
31-0	FLAG	发生过硬复位的复位标志，写 1 清 0: 电压过低 4: 调试复位 16: WDG0 17: WDG1 24: PWDG 31: 软件复位

RESET_FLAG 位域

12.6.2 RESET_STATUS (0x4)

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

RESET_STATUS [31:0]

位域	名称	描述
31-0	STATUS	复位源当前状态 0: 电压过低 4: 调试复位 16: WDG0 17: WDG1 24: PWDG 31: 软件复位

RESET_STATUS 位域

12.6.3 RESET_HOLD (0x8)

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

RESET_HOLD [31:0]

位域	名称	描述
31-0	HOLD	复位保持，此位置 1 的时候，若复位源保持有效则 SOC 保持复位状态，否则 SOC 复位自动释放? 0: 电压过低 4: 调试复位 16: WDG0 17: WDG1 24: PWDG 31: 软件复位

RESET_HOLD 位域

12.6.4 RESET_ENABLE (0xC)

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

RESET_ENABLE [31:0]

位域	名称	描述
31-0	ENABLE	允许复位源复位 SOC 0: 电压过低 4: 调试复位 16: WDG0 17: WDG1 24: PWDG 31: 软件复位

RESET_ENABLE 位域

12.6.5 RESET_TYPE (0x10)

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

RESET_TYPE [31:0]

位域	名称	描述
31-0	TYPE	冷热复位选择，0：冷复位，除了 debug/fuse/ioc 以外的所有配置均被清除，1：热复位，系统和引脚设置均被被保留，只复位某些 subsystem 0: 电压过低 4: 调试复位 16: WDG0 17: WDG1 24: PWDG 31: 软件复位

RESET_TYPE 位域

12.6.6 SOFTWARE_RESET (0x1C)

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

SOFTWARE_RESET [31:0]

位域	名称	描述
31-0	COUNTER	计数器在 24M 时钟下倒计数到 0 时停止，计数器计至 2 时产生复位，写 0 可撤销中断

SOFTWARE_RESET 位域

13 系统控制模块 SYSCTL

本章节介绍系统控制模块 SYSCTL 的主要功能及使用。

13.1 特性总结

系统控制模块 SYSCTL 是系统电源域的管理模块。它的主要功能有：

- 功能时钟配置
- 处理器启动管理
- 系统电源域资源管理
- 低功耗管理
- 时钟测量

13.2 功能时钟配置

功能时钟的配置在 SYSCTL 模块中实现，用户可通过修改 CLOCK_CPU[x] 和 CLOCK[x] 寄存器来配置功能时钟。其中 MUX 位用来选择时钟源，时钟源列表见表 16。其中 DIV 位用来设置分频系数，寄存器可配置的有效值为 0~255，实际分频系数为寄存器值 +1，如设置 DIV 位为 3，则会将选择的时钟源进行 4 分频后输出。

LOC_BUSY 位为 0 时标志该功能时钟在当前配置下已能够使用。

分频系数和时钟源选择均支持运行中 (on-the-fly) 修改。

13.3 CPU 启动管理

CPU0 是唯一的核，因此系统总是从 CPU0 启动。

13.3.1 启动管理

CPU0 的启动支持低功耗唤醒后的程序快速跳转，详见节 19.5。在该流程中 SYSCTL 提供以下支持：

- CPU[CPU0][LP] 寄存器的 RESET/SLEEP/WAKE 位用来标识事件记录
- CPU[CPU0][GPR0] 寄存器用来存放程序快速跳转入口
- CPU[CPU0][GPRx] 寄存器用来实现快速跳转的程序校验

13.4 系统电源域资源管理

系统控制模块 SYSCTL 管理整个系统电源域的时钟、电源开关和复位。

本节将对系统电源域中各种资源的开关控制机制作详细的介绍,编程模型的简单说明可直接跳转到小节 13.4.6查看。

13.4.1 资源节点

在系统电源域中，各种能够被开启或关闭的节点称为资源节点 (resource)，包括各功能模块、功能时钟、子系统电源和复位、时钟源等。

RESOURCE[x] 寄存器用来配置资源节点，MODE 位用来配置资源节点的控制模式，软件可直接修改 MODE 位来直接控制某资源的使能和关闭。

对于功能模块，将其 RESOURCE[*] 寄存器的 MODE 位设为 1 即可使能该模块，设为 2 则关闭该模块。

对于 CPU 子系统，将其 RESOURCE[POW_*] 寄存器的 mode 位设为 1 即打开该子系统的电源开关，设为

2 则关闭电源开关。

对于 CPU 和 SOC 子系统，将其 RESOURCE[RST_*] 寄存器的 mode 位设为 2 则将该子系统保持复位，设为 1 则退出复位。

对于功能时钟和时钟源，将其 RESOURCE[CLK_TOP_*] 和 RESOURCE[CLK_SRC_*] 寄存器的 MODE 位设为 1 即使能该时钟，设为 2 则关闭该时钟。

LOC_BUSY 位为 0 时标志该资源节点已能够使用。

在实际应用中，资源节点的控制模式通常配置为默认值 0 即自动模式，其具体的工作机制详见下一小节。

13.4.2 资源节点的链式结构

资源节点之间存在依赖关系，如功能模块的正常工作依赖功能时钟的正确输出，功能时钟又依赖 PLL 等时钟源的频率稳定。这种依赖关系构成了一种链式结构，在链的下游是终端的各个功能模块，中游是功能时钟、电源和复位等直接服务功能模块的资源，上游主要是各时钟源。

图 8以 CONN 子系统中的 ENET0 和 SDXC0 模块为例描述了资源节点的链式关系。

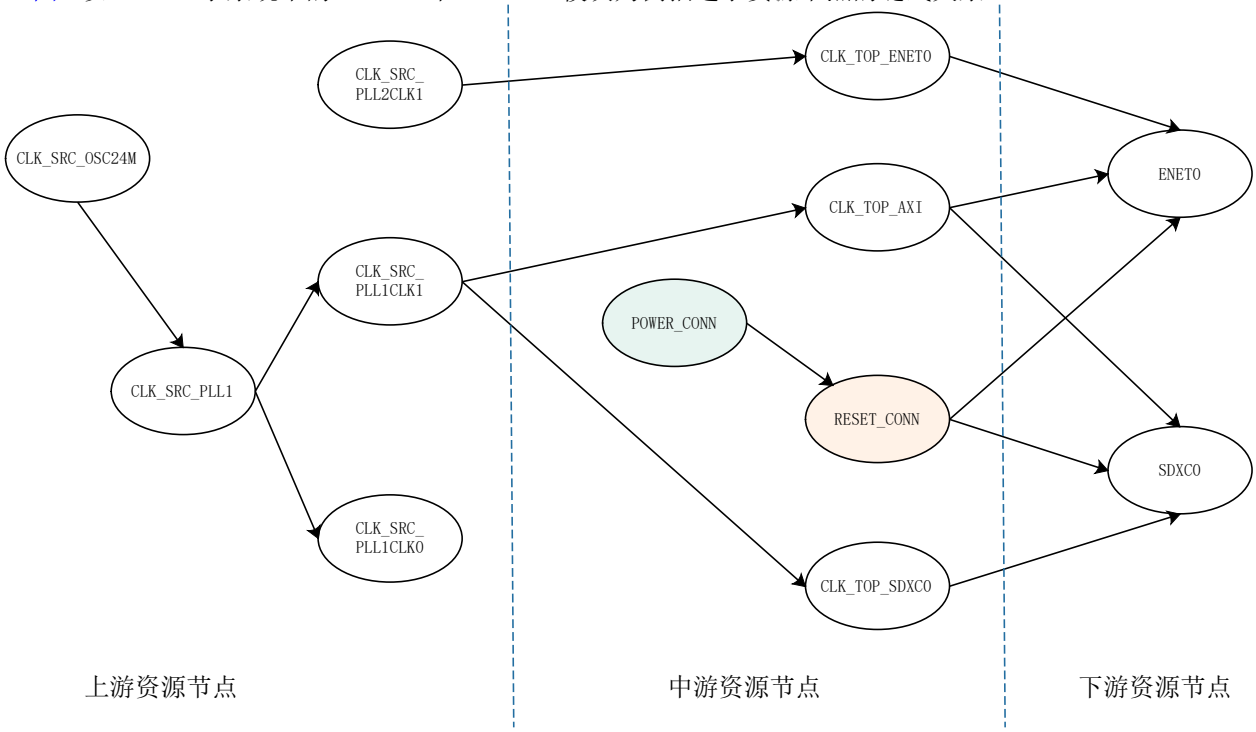


图 8: 资源节点的链式关系

*仅用于阐述结构原理，不表明产品一定存在 ENET0/SDCX0/CONN 子系统

下游的资源节点 ENET0 的正常工作依赖于 CLK_TOP_ENET0 提供接口时钟，依赖于 CLK_TOP_AXI 提供总线时钟，还需要其所在的 CONN 子系统处于非复位状态且电源开关打开。

中游的资源节点 CLK_TOP_ENET0 在默认配置下依赖时钟源 PLL2CLK1，CLK_TOP_AXI 在默认配置下依赖时钟源 PLL1CLK1，另外 CONN 子系统的复位退出依赖 CONN 子系统的电源开关打开。

上游的资源节点 PLL1CLK1 时钟源依赖 PLL1 使能并频率稳定，PLL1 又依赖 XTAL24M 提供有效的参考时钟。

所有的资源节点都由 SYSCTL 模块统一管理，资源节点之间的依赖关系已在硬件上自动实现，用户无需关注具体的依赖关系。在资源节点链上，当下游的某个节点需要开启时，它会自动请求自己依赖的所有中上游的资源

节点开启。如果中游和上游相关资源节点的 RESOURCE[] 寄存器中 MODE 位为 0 即处于自动模式时，该节点会在收到请求后自动开启，并同时要求更上游的资源节点开启。而资源节点默认就是工作在自动模式。

以 ENET0 模块为例，在所有中游和上游资源节点都工作于默认的自动模式时，将 RESOURCE[ENET0] 寄存器的 MODE 位修改为 1 即是要求使能 ENET0 模块，RESOURCE[ENET0] 寄存器的 LOC_BUSY 位会被置 1，同时通过链式传导，所有相关的资源节点都会被自动使能，直到 XTAL24M 震荡稳定，PLL1 时钟锁定，CLK_TOP_AXI 和 CLK_TOP_ENET0 时钟使能，CONN 子系统电源开启且复位结束时，RESOURCE[ENET0] 寄存器的 LOC_BUSY 位会被置 0，表示 ENET0 模块已准备就绪可以使用。

13.4.3 资源节点的自动关闭机制

链式结构上的资源节点能够根据请求自动开启，也能够在一定的条件下自动关闭。

当一个资源节点关闭后，它会向自己依赖的资源节点发送关闭请求，当一个资源节点连接的下游的所有节点都发送了关闭请求时，该节点才被允许关闭。

假设图 8中已显示了相关资源节点完整的依赖关系，在所有中游和上游资源节点都工作于默认的自动模式时，若将 RESOURCE[ENET0] 寄存器的 MODE 位修改为 2，即要求关闭 ENET0 模块，首先 ENET0 的模块时钟会被关闭，之后会请求关闭其依赖的所有资源节点。CLK_TOP_ENET0 由于只服务 ENET0 这一个下游资源节点，在收到其关闭请求后会执行关闭操作并继续向其依赖的上游资源节点 PLL2CLK1 发出关闭请求。

CLK_TOP_AXI 也会收到 ENET0 的关闭请求，但由于其下游同时连接的 SDXC0 仍处于工作状态，所以 CLK_TOP_AXI 不被允许关闭，RESET_CONN 资源节点也是如此。

13.4.4 资源节点保持开启

资源节点的自动关闭机制在允许的情况下能够一直将关闭请求自动传导到每条链的最上游如 XTAL24M 和 PLL 等，这些资源的重新开启会消耗一定的时间，如果用户关闭请求到达某个资源节点即停止传播，只需将该节点的 RESOURCE[] 寄存器的 MODE 位设为 1 即可将其强制保持使能。

常用的场景有：保持 PLL 不被关闭或 XTAL24M 一直工作，以及子系统的电源保持。

对于有独立电源开关的子系统，在低功耗模式下，存在以下几种可能的工作状态：

- 时钟保持：保持子系统内个别或全部模块的资源节点处于使能状态，使这些模块能够随时开始工作，这需要将相应的 RESOURCE[*] 寄存器的 MODE 位设为 1
- 寄存器保持：允许子系统内全部的模块资源节点关闭，但不对子系统进行复位，且保留电源，使得子系统内的所有寄存器和逻辑电路状态得以保存，这需要设置子系统的 RESOURCE[RST_*] 和 RESOURCE[POW_*] 寄存器的 MODE 位为 1
- 存储器保持：允许子系统进入复位状态，但保留电源，这样能够保留子系统中存储器内的数据，这需要设置子系统 RESOURCE[POW_*] 寄存器的 MODE 位为 1
- 子系统电源关闭

子系统的状态设置还有另外一种方式实现，详见小节 13.5.2。

13.4.5 功能模块与处理器的连接

中游和上游资源节点工作在自动模式时，会根据其下游节点的请求自动执行开启和关闭，而下游的功能模块资源节点工作在自动模式时，则能够根据处理器的状态执行开启和关闭。

SYSCTL 提供了一个连接组 (link group 0) 用来将功能模块与处理器连接起来，group0 提供给 CPU0 使用。

功能模块与 group 的连接在 GROUP[*] 寄存器中设置，寄存器每一个 bit 对应一个功能模块，将 bit 置 1 表示

将该模块与该 group 连接绑定。表 24列出了 GROUP 寄存器的 bit 位与功能模块的对应关系。

GROUPx[0] 寄存器		GROUPx[1] 寄存器	
bit	**资源节点**	**bit**	**资源节点**
0	AHB/APB 外设总线	0	UART7
1	ILM/DLM	1	WDG0
2	MCHTMR	2	WDG1
3	ROM	3	MBX
4	CAN0	4	TSNS
5	CAN1	5	CRC
6	CAN2	6	ADC0
7	CAN3	7	ADC1
8	PTPC	8	DAC0
9	LIN0	9	DAC1
10	LIN1	10	ACMP
11	LIN2	11	OPA0
12	LIN3	12	OPA1
13	GPTMR0	13	MOTOR
14	GPTMR1	14	RNG
15	GPTMR2	15	SDP
16	GPTMR3	16	KEYM
17	I2C0	17	GPIO
18	I2C1	18	HDMA
19	I2C2	19	XPI
20	I2C3	20	USB
21	SPI0	21	
22	SPI1	22	
23	SPI2	23	
24	SPI3	24	
25	UART0	25	
26	UART1	26	
27	UART2	27	
28	UART3	28	
29	UART4	29	
30	UART5	30	
31	UART6	31	

表 24: 功能模块连接位表

为保证基本系统能够工作，SYSCTL 已对部分资源节点做了默认配置，如将 CPU0 默认连接绑定至 group0，将资源节点 AHB/APB 外设总线和 ROM 默认连接到 group0。

CPU 与 group 的连接通过 AFFILIATE[x] 寄存器配置，而 AFFILIATE[CPU0] 寄存器的 LINK 位默认值为 0b1，即已将 CPU0 连接到了 group0。

当 CPU 与功能模块资源节点连接至同一个 group，且资源节点工作于自动模式时，这些资源节点就会随着 CPU 的状态变化而自动执行开启或关闭，如当 CPU 通过修改 GROUP 寄存器将一个资源节点连接至 CPU 自身所绑定的 group 时，该节点的功能模块会自动使能，模块依赖的一系列资源都会连带的自动开启，这也是系统推荐使用的最便捷的功能模块使能方式。

相应的，当 CPU 进入休眠时，其连接的各资源节点也会自动关闭。详细的 CPU 休眠后的行为管理请参考节 13.5。

引入 group 和 CPU 的连接功能后，图 8可以扩展为图 9的形式，构成完整的链式结构。

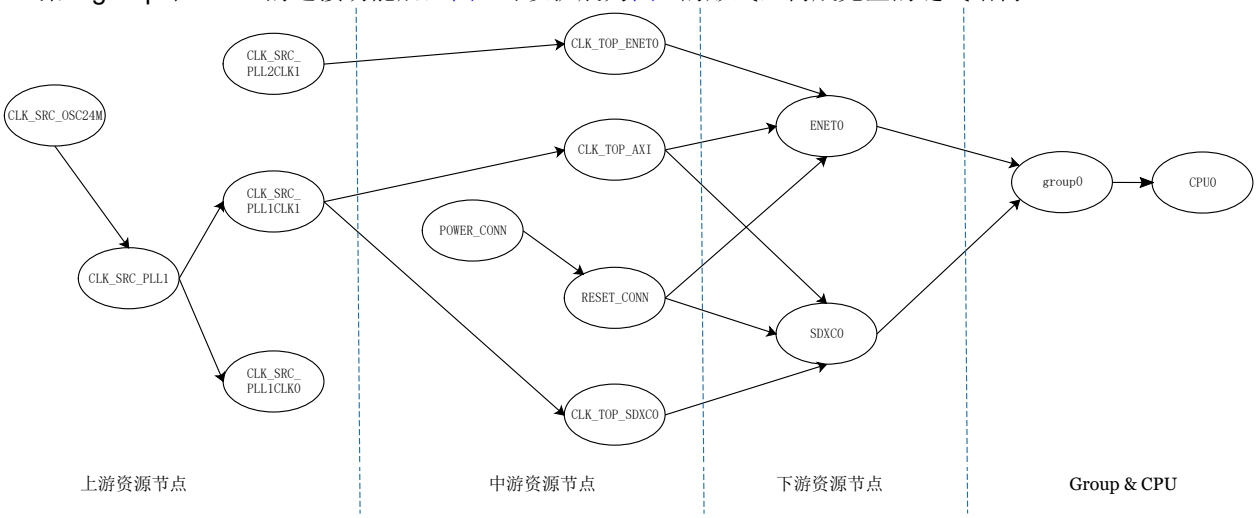


图 9: 资源节点与 group/CPU 的链式关系

* 仅用于阐述结构原理，不表明产品一定存在 ENET0/SDCX0/CONN 子系统

图中将 ENET0 和 SDXC0 连接到了 group0，表示 CPU0 需要使用 ENET0 和 SDXC0 这两个模块，其也形成了资源节点链的依赖关系。

当 CPU0 进入休眠后，会向 ENET0 和 SDXC0 都发出关闭请求，ENET0 和 SDXC0 会执行关闭并将关闭请求传导至链的上游。

13.4.6 资源节点的使用

系统电源域中所有的功能模块都是资源节点，他们在表 24被列出。

除了 AHB/APB 外设总线以及 ROM 等模块默认已使能外，大部分功能模块都默认处于关闭状态。

如果 CPU0 上的软件需要使用某个功能模块，只需将 GROUP0 寄存器中该模块对应的 bit 置 1，再等待该模块对应的 RESOURCE[*] 寄存器的 GLB_BUSY 位为 0 即可，所有必要的资源会自动打开。

13.5 低功耗管理

13.5.1 处理器的低功耗模式及唤醒

执行 WFI 指令是处理器从运行模式向低功耗模式切换的标志。进入低功耗模式后系统的行为由 CPU[CPUx][LP] 寄存器的 MODE 位设置。

- MODE 位为 b00 - 等待模式 (WAIT Mode)：只关闭处理器核心时钟
- MODE 位为 b01 - 停止模式 (STOP Mode)：关闭处理器核心时钟，且在资源节点链上向上游传导关闭请求，允许关闭更多的资源节点
- MODE 位为 b10 - 运行模式 (RUN Mode)：保留处理器核心时钟持续运行

- MODE 位为 b11 - 无效设置

处理器只有进入停止模式时才会触发 SYSCTL 执行资源节点链上的关闭操作。

从停止模式中唤醒一般由中断触发,哪些中断源能够唤醒系统是在 SYSCTL 的 CPU[CPUx][WAKEUP_ENABLE] 寄存器中配置的，将 bit 置 1 则对应的中断源能够将该 CPU 唤醒。

CPU[CPUx][WAKEUP_ENABLE] 寄存器的位分配与产品中断向量表所定义的中断号一致。

系统唤醒以 CPU 退出休眠并能够处理中断为目标，因而当能够唤醒 CPU 的中断出现时，SYSCTL 模块会自动按顺序将 CPU 所在的资源节点链上的所有节点全部打开。

13.5.2 低功耗模式下资源节点的状态保持

对于有独立电源开关的子系统，如果其电源和复位的资源节点工作在自动模式，处理器的休眠有可能使其电源关闭而导致子系统内的寄存器和存储器信息全部丢失。

对于 XTAL24M 和 PLL 等资源节点，如果工作在自动模式，处理器的休眠有可能使其自动关闭，导致系统唤醒时需花费大量的时间等待它们频率稳定。

要单独控制这些资源节点在低功耗模式下的行为，除了直接修改其 RESOURCE[] 寄存器将 MODE 从自动模式改为使能模式外，更推荐使用 RETENTION[CPUx] 寄存器来控制部分中上游资源节点的行为。

RETENTION 寄存器中每一个 bit 对应一个资源节点，将 bit 置 1 后能够控制该资源忽略下游的关闭请求而保持开启。

SYSCTL 提供了 1 个 RETENTION 寄存器，分配给 CPU0 使用，当 CPU0 上的软件希望某些资源节点在低功耗模式下保持开启时，只需在 RETENTION 寄存器中做出修改，将对应 bit 置 1。

对于一个可控的资源节点，只要 RETENTION 寄存器要求其保持开启，它就会在低功耗模式下一直保持工作状态。

注意，RETENTION 配置与资源节点链和 group 连接都无关，可以理解为加在部分中上游资源节点上的强制使能开关。

RETENTION 寄存器中具体的 bit 分配如表 25所示：

bit	保持开启的资源节点
0	系统电源域供电
1	系统电源域复位
2	CPU0 子系统供电
3	CPU0 子系统复位
4	XTAL24M
5	PLL0
6	PLL1

表 25: RETENTION 寄存器位表

13.6 时钟测量模块

时钟测量模块允许用户选择片上的时钟进行测量，并将当前的结果保存在结果寄存器里。

SYSCTL 中有 4 个时钟测量单元 SLICE0~3，用户可以通过 MONITOR[SLICEx][CONTROL] 寄存器的 SELECTION 位选择需要测量的时钟，SELECTION 值对应的测量时钟见表 26。

通过 MONITOR[SLICEx][CONTROL] 寄存器的 REFERENCE 位选择测量的基准是 XTAL32K 还是 XTAL24M，通过 MONITOR[SLICEx][CONTROL] 寄存器的 ACCURAY 位选择精度是 1KHz 或是 1Hz。

SELECTION	测量时钟选择
1	clk_32k
2	clk_rc24m
3	clk_xtal24m
4	clk_usb0_phy
20	clk_24m
21	pll0clk0
22	pll0clk1
23	pll0clk2
24	pll1clk0
25	pll1clk1
26	pll1clk2
27	pll1clk3
128	clk_top_cpu0
129	clk_top_mct0
130	clk_top_can0
131	clk_top_can1
132	clk_top_can2
133	clk_top_can3
134	clk_top_lin0
135	clk_top_lin1
136	clk_top_lin2
137	clk_top_lin3
138	clk_top_gptmr0
139	clk_top_gptmr1
140	clk_top_gptmr2
141	clk_top_gptmr3
142	clk_top_i2c0
143	clk_top_i2c1
144	clk_top_i2c2
145	clk_top_i2c3
146	clk_top_spi0
147	clk_top_spi1
148	clk_top_spi2
149	clk_top_spi3
150	clk_top_uart0
151	clk_top_uart1
152	clk_top_uart2
153	clk_top_uart3

SELECTION	测量时钟选择
154	clk_top_uart4
155	clk_top_uart5
156	clk_top_uart6
157	clk_top_uart7
158	clk_top_xpi0
159	clk_top_ana0
160	clk_top_ana1
161	clk_top_ana2
162	clk_top_ana3
163	clk_top_ref0
164	clk_top_ref1

表 26: 测量时钟选择表

配置完成后对 MONITOR[SLICEx][CONTROL] 寄存器的 START 位写 1 可启动时钟测量,等待 MONITOR[SLICEx][CONTROL] 寄存器的 VALID 标志位置 1 后，用户可以从 MONITOR[SLICEx][CURRENT] 寄存器里读取时钟的频率信息。

注意：时钟测量功能只适用于 400MHz 以下的时钟频率。

13.7 SYSCTL 寄存器说明

SYSCTL 的寄存器列表如下：

SYSCTL base address: 0xF4000000

地址偏移	名称	描述	复位值
0x0000	RESOURCE[CPU0]	cpu0_core 资源寄存器	0x00000000
0x0004	RESOURCE[CPX0]	cpu0 平台资源寄存器	0x00000000
0x0054	RESOURCE[POW_CPU0]	cpu0 电源资源寄存器	0x00000000
0x0058	RESOURCE[RST_SOC]	soc 复位资源寄存器	0x00000000
0x005C	RESOURCE[RST_CPU0]	cpu0 复位资源寄存器	0x00000000
0x0080	RESOURCE[CLK_SRC_XTAL]	24MHz 晶体资源寄存器	0x00000000
0x0084	RESOURCE[CLK_SRC_PLL0]	pll0 资源寄存器	0x00000000
0x0088	RESOURCE[CLK_SRC_CLK0_PLL0]	clk0_pll0 资源寄存器	0x00000000
0x008C	RESOURCE[CLK_SRC_CLK1_PLL0]	clk1_pll0 资源寄存器	0x00000000
0x0090	RESOURCE[CLK_SRC_CLK2_PLL0]	clk2_pll0 资源寄存器	0x00000000
0x0094	RESOURCE[CLK_SRC_PLL1]	pll1 资源寄存器	0x00000000
0x0098	RESOURCE[CLK_SRC_CLK0_PLL1]	clk0_pll1 资源寄存器	0x00000000
0x009C	RESOURCE[CLK_SRC_CLK1_PLL1]	clk1_pll1 资源寄存器	0x00000000
0x00A0	RESOURCE[CLK_SRC_CLK2_PLL1]	clk2_pll1 资源寄存器	0x00000000
0x00A4	RESOURCE[CLK_SRC_CLK3_PLL1]	clk3_pll1 资源寄存器	0x00000000
0x00A8	RESOURCE[CLK_SRC_PLL0_REF]	pll0 ref clock 资源寄存器	0x00000000
0x00AC	RESOURCE[CLK_SRC_PLL1_REF]	pll1 ref clock 资源寄存器	0x00000000

地址偏移	名称	描述	复位值
0x0100	RESOURCE[CLK_TOP_CPU0]	cpu0 功能时钟资源寄存器	0x00000000
0x0104	RESOURCE[CLK_TOP_MCT0]	mct0 功能时钟资源寄存器	0x00000000
0x0108	RESOURCE[CLK_TOP_CAN0]	can0 功能时钟资源寄存器	0x00000000
0x010C	RESOURCE[CLK_TOP_CAN1]	can1 功能时钟资源寄存器	0x00000000
0x0110	RESOURCE[CLK_TOP_CAN2]	can2 功能时钟资源寄存器	0x00000000
0x0114	RESOURCE[CLK_TOP_CAN3]	can3 功能时钟资源寄存器	0x00000000
0x0118	RESOURCE[CLK_TOP_LIN0]	lin0 功能时钟资源寄存器	0x00000000
0x011C	RESOURCE[CLK_TOP_LIN1]	lin1 功能时钟资源寄存器	0x00000000
0x0120	RESOURCE[CLK_TOP_LIN2]	lin2 功能时钟资源寄存器	0x00000000
0x0124	RESOURCE[CLK_TOP_LIN3]	lin3 功能时钟资源寄存器	0x00000000
0x0128	RESOURCE[CLK_TOP_TMR0]	tmr0 功能时钟资源寄存器	0x00000000
0x012C	RESOURCE[CLK_TOP_TMR1]	tmr1 功能时钟资源寄存器	0x00000000
0x0130	RESOURCE[CLK_TOP_TMR2]	tmr2 功能时钟资源寄存器	0x00000000
0x0134	RESOURCE[CLK_TOP_TMR3]	tmr3 功能时钟资源寄存器	0x00000000
0x0138	RESOURCE[CLK_TOP_I2C0]	i2c0 功能时钟资源寄存器	0x00000000
0x013C	RESOURCE[CLK_TOP_I2C1]	i2c1 功能时钟资源寄存器	0x00000000
0x0140	RESOURCE[CLK_TOP_I2C2]	i2c2 功能时钟资源寄存器	0x00000000
0x0144	RESOURCE[CLK_TOP_I2C3]	i2c3 功能时钟资源寄存器	0x00000000
0x0148	RESOURCE[CLK_TOP_SPI0]	spi0 功能时钟资源寄存器	0x00000000
0x014C	RESOURCE[CLK_TOP_SPI1]	spi1 功能时钟资源寄存器	0x00000000
0x0150	RESOURCE[CLK_TOP_SPI2]	spi2 功能时钟资源寄存器	0x00000000
0x0154	RESOURCE[CLK_TOP_SPI3]	spi3 功能时钟资源寄存器	0x00000000
0x0158	RESOURCE[CLK_TOP_URT0]	uart0 功能时钟资源寄存器	0x00000000
0x015C	RESOURCE[CLK_TOP_URT1]	uart1 功能时钟资源寄存器	0x00000000
0x0160	RESOURCE[CLK_TOP_URT2]	uart2 功能时钟资源寄存器	0x00000000
0x0164	RESOURCE[CLK_TOP_URT3]	uart3 功能时钟资源寄存器	0x00000000
0x0168	RESOURCE[CLK_TOP_URT4]	uart4 功能时钟资源寄存器	0x00000000
0x016C	RESOURCE[CLK_TOP_URT5]	uart5 功能时钟资源寄存器	0x00000000
0x0170	RESOURCE[CLK_TOP_URT6]	uart6 功能时钟资源寄存器	0x00000000
0x0174	RESOURCE[CLK_TOP_URT7]	uart7 功能时钟资源寄存器	0x00000000
0x0178	RESOURCE[CLK_TOP_XPI0]	xpi0 功能时钟资源寄存器	0x00000000
0x017C	RESOURCE[CLK_TOP_ANA0]	ana0 功能时钟资源寄存器	0x00000000
0x0180	RESOURCE[CLK_TOP_ANA1]	ana1 功能时钟资源寄存器	0x00000000
0x0184	RESOURCE[CLK_TOP_ANA2]	ana2 功能时钟资源寄存器	0x00000000
0x0188	RESOURCE[CLK_TOP_ANA3]	ana3 功能时钟资源寄存器	0x00000000
0x018C	RESOURCE[CLK_TOP_REF0]	ref0 功能时钟资源寄存器	0x00000000
0x0190	RESOURCE[CLK_TOP_REF1]	ref1 功能时钟资源寄存器	0x00000000
0x0194	RESOURCE[CLK_TOP_ADC0]	adc0 功能时钟资源寄存器	0x00000000
0x0198	RESOURCE[CLK_TOP_ADC1]	adc1 功能时钟资源寄存器	0x00000000
0x019C	RESOURCE[CLK_TOP_DAC0]	dac0 功能时钟资源寄存器	0x00000000

地址偏移	名称	描述	复位值
0x01A0	RESOURCE[CLK_TOP_DAC1]	dac1 功能时钟资源寄存器	0x00000000
0x0400	RESOURCE[AHB0]	soc bus 资源寄存器	0x00000000
0x0404	RESOURCE[LMM0]	ILM0/DLM0 资源寄存器	0x00000000
0x0408	RESOURCE[MCT0]	mchtmr0 资源寄存器	0x00000000
0x040C	RESOURCE[ROM0]	rom 资源寄存器	0x00000000
0x0410	RESOURCE[CAN0]	can0 资源寄存器	0x00000000
0x0414	RESOURCE[CAN1]	can1 资源寄存器	0x00000000
0x0418	RESOURCE[CAN2]	can2 资源寄存器	0x00000000
0x041C	RESOURCE[CAN3]	can3 资源寄存器	0x00000000
0x0420	RESOURCE[PTPC]	ptpc 资源寄存器	0x00000000
0x0424	RESOURCE[LIN0]	lin0 资源寄存器	0x00000000
0x0428	RESOURCE[LIN1]	lin1 资源寄存器	0x00000000
0x042C	RESOURCE[LIN2]	lin2 资源寄存器	0x00000000
0x0430	RESOURCE[LIN3]	lin3 资源寄存器	0x00000000
0x0434	RESOURCE[TMR0]	tmr0 资源寄存器	0x00000000
0x0438	RESOURCE[TMR1]	tmr1 资源寄存器	0x00000000
0x043C	RESOURCE[TMR2]	tmr2 资源寄存器	0x00000000
0x0440	RESOURCE[TMR3]	tmr3 资源寄存器	0x00000000
0x0444	RESOURCE[I2C0]	i2c0 资源寄存器	0x00000000
0x0448	RESOURCE[I2C1]	i2c1 资源寄存器	0x00000000
0x044C	RESOURCE[I2C2]	i2c2 资源寄存器	0x00000000
0x0450	RESOURCE[I2C3]	i2c3 资源寄存器	0x00000000
0x0454	RESOURCE[SPI0]	spi0 资源寄存器	0x00000000
0x0458	RESOURCE[SPI1]	spi1 资源寄存器	0x00000000
0x045C	RESOURCE[SPI2]	spi2 资源寄存器	0x00000000
0x0460	RESOURCE[SPI3]	spi3 资源寄存器	0x00000000
0x0464	RESOURCE[URT0]	uart0 资源寄存器	0x00000000
0x0468	RESOURCE[URT1]	uart1 资源寄存器	0x00000000
0x046C	RESOURCE[URT2]	uart2 资源寄存器	0x00000000
0x0470	RESOURCE[URT3]	uart3 资源寄存器	0x00000000
0x0474	RESOURCE[URT4]	uart4 资源寄存器	0x00000000
0x0478	RESOURCE[URT5]	uart5 资源寄存器	0x00000000
0x047C	RESOURCE[URT6]	uart6 资源寄存器	0x00000000
0x0480	RESOURCE[URT7]	uart7 资源寄存器	0x00000000
0x0484	RESOURCE[WDG0]	wdg0 资源寄存器	0x00000000
0x0488	RESOURCE[WDG1]	wdg1 资源寄存器	0x00000000
0x048C	RESOURCE[MBX0]	mbx0 资源寄存器	0x00000000
0x0490	RESOURCE[TSNS]	tsns 资源寄存器	0x00000000
0x0494	RESOURCE[CRC0]	crc 资源寄存器	0x00000000
0x0498	RESOURCE[ADC0]	adc0 资源寄存器	0x00000000

地址偏移	名称	描述	复位值
0x049C	RESOURCE[ADC1]	adc1 资源寄存器	0x00000000
0x04A0	RESOURCE[DAC0]	dac0 资源寄存器	0x00000000
0x04A4	RESOURCE[DAC1]	dac1 资源寄存器	0x00000000
0x04A8	RESOURCE[ACMP]	acmp 资源寄存器	0x00000000
0x04AC	RESOURCE[OPA0]	opamp0 资源寄存器	0x00000000
0x04B0	RESOURCE[OPA1]	opamp1 资源寄存器	0x00000000
0x04B4	RESOURCE[MOT0]	motor 资源寄存器	0x00000000
0x04B8	RESOURCE[RNG0]	rng 资源寄存器	0x00000000
0x04BC	RESOURCE[SDP0]	sdp 资源寄存器	0x00000000
0x04C0	RESOURCE[KMAN]	key_manager 资源寄存器	0x00000000
0x04C4	RESOURCE[GPIO]	gpio 资源寄存器	0x00000000
0x04C8	RESOURCE[HDMA]	hdma 资源寄存器	0x00000000
0x04CC	RESOURCE[XPI0]	xpi0 资源寄存器	0x00000000
0x04D0	RESOURCE[USB0]	usb 资源寄存器	0x00000000
0x04D4	RESOURCE[REF0]	ref0 资源寄存器	0x00000000
0x04D8	RESOURCE[REF1]	ref1 资源寄存器	0x00000000
0x0800	GROUP0[LINK0][VALUE]	资源组 0 分组控制寄存器 &struct_array_index	0x00000000
0x0804	GROUP0[LINK0][SET]	资源组 0 分组控制寄存器 &struct_array_index 置位	0x00000000
0x0808	GROUP0[LINK0][CLEAR]	资源组 0 分组控制寄存器 &struct_array_index 清零	0x00000000
0x080C	GROUP0[LINK0][TOGGLE]	资源组 0 分组控制寄存器 &struct_array_index 翻转	0x00000000
0x0810	GROUP0[LINK1][VALUE]	资源组 0 分组控制寄存器 &struct_array_index	0x00000000
0x0814	GROUP0[LINK1][SET]	资源组 0 分组控制寄存器 &struct_array_index 置位	0x00000000
0x0818	GROUP0[LINK1][CLEAR]	资源组 0 分组控制寄存器 &struct_array_index 清零	0x00000000
0x081C	GROUP0[LINK1][TOGGLE]	资源组 0 分组控制寄存器 &struct_array_index 翻转	0x00000000
0x0820	GROUP0[LINK2][VALUE]	资源组 0 分组控制寄存器 &struct_array_index	0x00000000
0x0824	GROUP0[LINK2][SET]	资源组 0 分组控制寄存器 &struct_array_index 置位	0x00000000
0x0828	GROUP0[LINK2][CLEAR]	资源组 0 分组控制寄存器 &struct_array_index 清零	0x00000000
0x082C	GROUP0[LINK2][TOGGLE]	资源组 0 分组控制寄存器 &struct_array_index 翻转	0x00000000

地址偏移	名称	描述	复位值
0x0830	GROUP0[LINK3][VALUE]	资源组 0 分组控制寄存器 &struct_array_index	0x00000000
0x0834	GROUP0[LINK3][SET]	资源组 0 分组控制寄存器 &struct_array_index 置位	0x00000000
0x0838	GROUP0[LINK3][CLEAR]	资源组 0 分组控制寄存器 &struct_array_index 清零	0x00000000
0x083C	GROUP0[LINK3][TOGGLE]	资源组 0 分组控制寄存器 &struct_array_index 翻转	0x00000000
0x0900	AFFILIATE[CPU0][VALUE]	CPU0 资源组控制寄存器	0x00000000
0x0904	AFFILIATE[CPU0][SET]	CPU0 资源组控制寄存器置位	0x00000000
0x0908	AFFILIATE[CPU0][CLEAR]	CPU0 资源组控制寄存器清零	0x00000000
0x090C	AFFILIATE[CPU0][TOGGLE]	CPU0 资源组控制寄存器翻转	0x00000000
0x0920	RETENTION[CPU0][VALUE]	CPU0 唤醒保持寄存器	0x00000000
0x0924	RETENTION[CPU0][SET]	CPU0 唤醒保持寄存器置位	0x00000000
0x0928	RETENTION[CPU0][CLEAR]	CPU0 唤醒保持寄存器清零	0x00000000
0x092C	RETENTION[CPU0][TOGGLE]	CPU0 唤醒保持寄存器翻转	0x00000000
0x1000	POWER[CPU0][STATUS]	CPU0 电源开关控制寄存器	0x80000000
0x1004	POWER[CPU0][LF_WAIT]	CPU0 电源开关低扇出等待	0x000000FF
0x100C	POWER[CPU0][OFF_WAIT]	CPU0 电源关闭等待	0x0000000F
0x1010	POWER[CPU0][RET_WAIT]	CPU0 memory 内容保持模式跳转等待	0x0000000F
0x1400	RESET[SOC][CONTROL]	SOC 复位控制寄存器	0x80000000
0x1404	RESET[SOC][CONFIG]	SOC 复位配置寄存器	0x00402003
0x140C	RESET[SOC][COUNTER]	SOC 复位计数器	0x00000000
0x1410	RESET[CPU0][CONTROL]	CPU0 复位控制寄存器	0x80000000
0x1414	RESET[CPU0][CONFIG]	CPU0 复位配置寄存器	0x00402003
0x141C	RESET[CPU0][COUNTER]	CPU0 复位计数器	0x00000000
0x1800	CLOCK_CPU[CLK_TOP_CPU0]	cpu0 功能时钟设置寄存器	0x00000000
0x1804	CLOCK[CLK_TOP_MCT0]	mct0 功能时钟设置寄存器	0x00000000
0x1808	CLOCK[CLK_TOP_CAN0]	can0 功能时钟设置寄存器	0x00000000
0x180C	CLOCK[CLK_TOP_CAN1]	can1 功能时钟设置寄存器	0x00000000
0x1810	CLOCK[CLK_TOP_CAN2]	can2 功能时钟设置寄存器	0x00000000
0x1814	CLOCK[CLK_TOP_CAN3]	can3 功能时钟设置寄存器	0x00000000
0x1818	CLOCK[CLK_TOP_LIN0]	lin0 功能时钟设置寄存器	0x00000000
0x181C	CLOCK[CLK_TOP_LIN1]	lin1 功能时钟设置寄存器	0x00000000
0x1820	CLOCK[CLK_TOP_LIN2]	lin2 功能时钟设置寄存器	0x00000000
0x1824	CLOCK[CLK_TOP_LIN3]	lin3 功能时钟设置寄存器	0x00000000
0x1828	CLOCK[CLK_TOP_TMR0]	tmr0 功能时钟设置寄存器	0x00000000
0x182C	CLOCK[CLK_TOP_TMR1]	tmr1 功能时钟设置寄存器	0x00000000
0x1830	CLOCK[CLK_TOP_TMR2]	tmr2 功能时钟设置寄存器	0x00000000

地址偏移	名称	描述	复位值
0x1834	CLOCK[CLK_TOP_TMR3]	tmr3 功能时钟设置寄存器	0x00000000
0x1838	CLOCK[CLK_TOP_I2C0]	i2c0 功能时钟设置寄存器	0x00000000
0x183C	CLOCK[CLK_TOP_I2C1]	i2c1 功能时钟设置寄存器	0x00000000
0x1840	CLOCK[CLK_TOP_I2C2]	i2c2 功能时钟设置寄存器	0x00000000
0x1844	CLOCK[CLK_TOP_I2C3]	i2c3 功能时钟设置寄存器	0x00000000
0x1848	CLOCK[CLK_TOP_SPI0]	spi0 功能时钟设置寄存器	0x00000000
0x184C	CLOCK[CLK_TOP_SPI1]	spi1 功能时钟设置寄存器	0x00000000
0x1850	CLOCK[CLK_TOP_SPI2]	spi2 功能时钟设置寄存器	0x00000000
0x1854	CLOCK[CLK_TOP_SPI3]	spi3 功能时钟设置寄存器	0x00000000
0x1858	CLOCK[CLK_TOP_URT0]	uart0 功能时钟设置寄存器	0x00000000
0x185C	CLOCK[CLK_TOP_URT1]	uart1 功能时钟设置寄存器	0x00000000
0x1860	CLOCK[CLK_TOP_URT2]	uart2 功能时钟设置寄存器	0x00000000
0x1864	CLOCK[CLK_TOP_URT3]	uart3 功能时钟设置寄存器	0x00000000
0x1868	CLOCK[CLK_TOP_URT4]	uart4 功能时钟设置寄存器	0x00000000
0x186C	CLOCK[CLK_TOP_URT5]	uart5 功能时钟设置寄存器	0x00000000
0x1870	CLOCK[CLK_TOP_URT6]	uart6 功能时钟设置寄存器	0x00000000
0x1874	CLOCK[CLK_TOP_URT7]	uart7 功能时钟设置寄存器	0x00000000
0x1878	CLOCK[CLK_TOP_XPI0]	xpi0 功能时钟设置寄存器	0x00000000
0x187C	CLOCK[CLK_TOP_ANA0]	ana0 功能时钟设置寄存器	0x00000000
0x1880	CLOCK[CLK_TOP_ANA1]	ana1 功能时钟设置寄存器	0x00000000
0x1884	CLOCK[CLK_TOP_ANA2]	ana2 功能时钟设置寄存器	0x00000000
0x1888	CLOCK[CLK_TOP_ANA3]	ana3 功能时钟设置寄存器	0x00000000
0x188C	CLOCK[CLK_TOP_REF0]	ref0 功能时钟设置寄存器	0x00000000
0x1890	CLOCK[CLK_TOP_REF1]	ref1 功能时钟设置寄存器	0x00000000
0x1C00	ADCCLK[CLK_TOP_ADC0]	adc0 功能时钟设置寄存器	0x00000000
0x1C04	ADCCLK[CLK_TOP_ADC1]	adc1 功能时钟设置寄存器	0x00000000
0x1C08	DACCLK[CLK_TOP_DAC0]	dac0 功能时钟设置寄存器	0x00000000
0x1C0C	DACCLK[CLK_TOP_DAC1]	dac1 功能时钟设置寄存器	0x00000000
0x2000	GLOBAL00	时钟模式寄存器	0x00000000
0x2400	MONITOR[SLICE0][CONTROL]	时钟观测控制寄存器 SLICE0	0x00000000
0x2404	MONITOR[SLICE0][CURRENT]	时钟频率 SLICE0	0x00000000
0x2408	MONITOR[SLICE0][LOW_LIMIT]	时钟频率下限 SLICE0	0xFFFFFFFF
0x240C	MONITOR[SLICE0][HIGH_LIMIT]	时钟频率上限 SLICE0	0x00000000
0x2420	MONITOR[SLICE1][CONTROL]	时钟观测控制寄存器 SLICE1	0x00000000
0x2424	MONITOR[SLICE1][CURRENT]	时钟频率 SLICE1	0x00000000
0x2428	MONITOR[SLICE1][LOW_LIMIT]	时钟频率下限 SLICE1	0xFFFFFFFF
0x242C	MONITOR[SLICE1][HIGH_LIMIT]	时钟频率上限 SLICE1	0x00000000
0x2440	MONITOR[SLICE2][CONTROL]	时钟观测控制寄存器 SLICE2	0x00000000

地址偏移	名称	描述	复位值
0x2444	MONITOR[SLICE2][CURRENT]	时钟频率 SLICE2	0x00000000
0x2448	MONITOR[SLICE2][LOW_LIMIT]	时钟频率下限 SLICE2	0xFFFFFFFF
0x244C	MONITOR[SLICE2][HIGH_LIMIT]	时钟频率上限 SLICE2	0x00000000
0x2460	MONITOR[SLICE3][CONTROL]	时钟观测控制寄存器 SLICE3	0x00000000
0x2464	MONITOR[SLICE3][CURRENT]	时钟频率 SLICE3	0x00000000
0x2468	MONITOR[SLICE3][LOW_LIMIT]	时钟频率下限 SLICE3	0xFFFFFFFF
0x246C	MONITOR[SLICE3][HIGH_LIMIT]	时钟频率上限 SLICE3	0x00000000
0x2800	CPU[CPU0][LP]	CPU0 低功耗控制器	0x00001000
0x2804	CPU[CPU0][LOCK]	CPU0 通用寄存器锁定控制寄存器	0x00000000
0x2808	CPU[CPU0][GPR][GPR0]	CPU0 通用寄存器 0	0x00000000
0x280C	CPU[CPU0][GPR][GPR1]	CPU0 通用寄存器 1	0x00000000
0x2810	CPU[CPU0][GPR][GPR2]	CPU0 通用寄存器 2	0x00000000
0x2814	CPU[CPU0][GPR][GPR3]	CPU0 通用寄存器 3	0x00000000
0x2818	CPU[CPU0][GPR][GPR4]	CPU0 通用寄存器 4	0x00000000
0x281C	CPU[CPU0][GPR][GPR5]	CPU0 通用寄存器 5	0x00000000
0x2820	CPU[CPU0][GPR][GPR6]	CPU0 通用寄存器 6	0x00000000
0x2824	CPU[CPU0][GPR][GPR7]	CPU0 通用寄存器 7	0x00000000
0x2828	CPU[CPU0][GPR][GPR8]	CPU0 通用寄存器 8	0x00000000
0x282C	CPU[CPU0][GPR][GPR9]	CPU0 通用寄存器 9	0x00000000
0x2830	CPU[CPU0][GPR][GPR10]	CPU0 通用寄存器 10	0x00000000
0x2834	CPU[CPU0][GPR][GPR11]	CPU0 通用寄存器 11	0x00000000
0x2838	CPU[CPU0][GPR][GPR12]	CPU0 通用寄存器 12	0x00000000
0x283C	CPU[CPU0][GPR][GPR13]	CPU0 通用寄存器 13	0x00000000
0x2840	CPU[CPU0][WAKEUP_STATUS][STATUS0]	唤醒 CPU0 的 IRQ 状态	0x00000000
0x2844	CPU[CPU0][WAKEUP_STATUS][STATUS1]	唤醒 CPU0 的 IRQ 状态	0x00000000
0x2848	CPU[CPU0][WAKEUP_STATUS][STATUS2]	唤醒 CPU0 的 IRQ 状态	0x00000000
0x284C	CPU[CPU0][WAKEUP_STATUS][STATUS3]	唤醒 CPU0 的 IRQ 状态	0x00000000
0x2880	CPU[CPU0][WAKEUP_ENABLE][ENABLE0]	唤醒 CPU0 的 IRQ 使能	0x00000000
0x2884	CPU[CPU0][WAKEUP_ENABLE][ENABLE1]	唤醒 CPU0 的 IRQ 使能	0x00000000
0x2888	CPU[CPU0][WAKEUP_ENABLE][ENABLE2]	唤醒 CPU0 的 IRQ 使能	0x00000000
0x288C	CPU[CPU0][WAKEUP_ENABLE][ENABLE3]	唤醒 CPU0 的 IRQ 使能	0x00000000

表 27: SYSCTL 寄存器列表

注意：本产品 ROM 的启动代码会配置本模块，因此软件读取到的寄存器值与本章节描述的复位值有可能不一致。

13.8 寄存器详细信息

SYSCTL 的寄存器详细说明如下：

13.8.1 RESOURCE (0x0 + 0x4 * n)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GLB_BUSY	LOC_BUSY	RSVD																												MODE	
RO	RO	N/A																												RW	
0	0	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0

RESOURCE [31:0]

位域	名称	描述
31	GLB_BUSY	全局忙标志位 0: 所有资源的状态均已稳定 1: 系统里至少有一个资源正在打开或关闭
30	LOC_BUSY	当前资源忙标志位 0: 当前资源的状态已稳定 1: 当前资源正在打开或关闭
1-0	MODE	资源管理模式 00：根据需求自动打开或关闭（推荐） 01：强制打开 10：强制关闭 11：保留

RESOURCE 位域

13.8.2 GROUP0[VALUE] (0x800 + 0x10 * n)

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

GROUP0[VALUE] [31:0]

位域	名称	描述
31-0	LINK	组内外设选择寄存器，每一位代表一个外设，编号从 AHBP(0x400) 开始 0：外设不属于该分组 1：外设属于该分组

GROUP0[VALUE] 位域

13.8.3 GROUP0[SET] (0x804 + 0x10 * n)

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

GROUP0[SET] [31:0]

位域	名称	描述
31-0	LINK	组内外设选择寄存器，每一位代表一个外设，编号从 AHBP(0x400) 开始。 读取值和控制寄存器相同，写入值如下 0：不影响分组 1：将外设加入该分组

GROUP0[SET] 位域

13.8.4 GROUP0[CLEAR] (0x808 + 0x10 * n)

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

GROUP0[CLEAR] [31:0]

位域	名称	描述
31-0	LINK	组内外设选择寄存器，每一位代表一个外设，编号从 AHBP(0x400) 开始。 读取值和控制寄存器相同，写入值如下 0：不影响分组 1：将外设从该分组删除

GROUP0[CLEAR] 位域

13.8.5 GROUP0[TOGGLE] (0x80C + 0x10 * n)

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

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

GROUP0[TOGGLE] [31:0]

位域	名称	描述
31-0	LINK	组内外设选择寄存器，每一位代表一个外设，编号从 AHBP(0x400) 开始。 读取值和控制寄存器相同，写入值如下 0：不影响分组 1：若外设属于该组删除外设，若不属于则加入该分组

GROUP0[TOGGLE] 位域

13.8.6 AFFILIATE[VALUE] (0x900 + 0x10 * n)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																												LINK			
N/A																												RW			
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0

AFFILIATE[VALUE] [31:0]

位域	名称	描述
3-0	LINK	CPU0 使用的资源组，每一位代表一个分组 0：分组不属于 CPU0 1：分组属于 CPU0

AFFILIATE[VALUE] 位域

13.8.7 AFFILIATE[SET] (0x904 + 0x10 * n)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																												LINK			
N/A																												RW			
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0

AFFILIATE[SET] [31:0]

位域	名称	描述
3-0	LINK	CPU0 使用的资源组，每一位代表一个分组。读取值和 CPU0 资源组控制寄存器相同，写入值如下 0：不影响 CPU0 1：将分组加入 CPU0

AFFILIATE[SET] 位域

13.8.8 AFFILIATE[CLEAR] (0x908 + 0x10 * n)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																												LINK			
N/A																												RW			
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0

AFFILIATE[CLEAR] [31:0]

位域	名称	描述
3-0	LINK	CPU0 使用的资源组，每一位代表一个分组。读取值和 CPU0 资源组控制寄存器相同，写入值如下 0：不影响 CPU0 1：将分组 CPU0 删除

AFFILIATE[CLEAR] 位域

13.8.9 AFFILIATE[TOGGLE] (0x90C + 0x10 * n)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																												LINK			
N/A																												RW			
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0

AFFILIATE[TOGGLE] [31:0]

位域	名称	描述
3-0	LINK	CPU0 使用的资源组，每一位代表一个分组。读取值和 CPU0 资源组控制寄存器相同，写入值如下 0：不影响 CPU0 1：若分组属于 CPU0 删除分组，若不属于则加入 CPU0

AFFILIATE[TOGGLE] 位域

13.8.10 RETENTION[VALUE] (0x920 + 0x10 * n)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																	LINK														
N/A																	RW													
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

RETENTION[VALUE] [31:0]

位域	名称	描述
14-0	LINK	在 CPU0 进入停止模式的时候，保持资源工作，每一位代表一个资源 位 0: soc 电源和存储器内容 位 1: soc 外设和寄存器设置 位 2: cpu0 电源和存储器内容 位 3: cpu0 外设和寄存器设置 位 4: xtal 晶体 位 5: pll0 锁定 位 6: pll1 锁定

RETENTION[VALUE] 位域

13.8.11 RETENTION[SET] (0x924 + 0x10 * n)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																	LINK														
N/A																	RW													
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

RETENTION[SET] [31:0]

位域	名称	描述
14-0	LINK	在 CPU0 进入停止模式的时候，保持资源工作，每一位代表一个资源，读值和唤醒保持寄存器 0 相同, 写入值如下 0：没有影响 1：保持

RETENTION[SET] 位域

13.8.12 RETENTION[CLEAR] (0x928 + 0x10 * n)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																	LINK														
N/A																	RW														
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

RETENTION[CLEAR] [31:0]

位域	名称	描述
14-0	LINK	在 CPU0 进入停止模式的时候，保持资源工作，每一位代表一个资源，读值和唤醒保持寄存器 0 相同, 写入值如下 0：没有影响 1：不保持

RETENTION[CLEAR] 位域

13.8.13 RETENTION[TOGGLE] (0x92C + 0x10 * n)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																	LINK														
N/A																	RW														
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

RETENTION[TOGGLE] [31:0]

位域	名称	描述
14-0	LINK	在 CPU0 进入停止模式的时候，保持资源工作，每一位代表一个资源，读值和唤醒保持寄存器 0 相同, 写入值如下 0：没有影响 1：翻转设置

RETENTION[TOGGLE] 位域

13.8.14 POWER[STATUS] (0x1000 + 0x14 * n)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FLAG	FLAG_WAKE	RSVD												MEM_RET_N	MEM_RET_P	RSVD			LF_DISABLE	RSVD			LF_ACK	RSVD							
RW	RW	N/A												RO	RO	N/A			RO	N/A			RO	N/A							
1	0	x	x	x	x	x	x	x	x	x	x	x	x	0	0	x	x	x	0	x	x	x	0	x	x	x	x	x	x	x	x

POWER[STATUS] [31:0]

位域	名称	描述
31	FLAG	上电标志，指示在发生过上电事件，此标志位写 1 会清除标志 0：未发生上电 1：发生过上电
30	FLAG_WAKE	掉电唤醒标志，指示在首次上电后又发生过上电事件，此标志位写 1 会清除标志 0：未发生上电 1：发生过上电
17	MEM_RET_N	memory 信息保持模式控制信号 0：memory 进入保持模式 1：memory 退出保持模式
16	MEM_RET_P	memory 信息保持模式控制信号 0：memory 退出保持模式 1：memory 进入保持模式
12	LF_DISABLE	低扇出电源开关控制信号 0：电源开关断路 1：电源开关接通
8	LF_ACK	低扇出电源开关控制反馈信号 0：电源开关断路 1：电源开关接通

POWER[STATUS] 位域

13.8.15 POWER[LF_WAIT] (0x1004 + 0x14 * n)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD												WAIT																			
N/A												RW																			
x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1

POWER[LF_WAIT] [31:0]

位域	名称	描述
19-0	WAIT	电源开关开启时间，以 24M 时钟为单位，缺省值为 255 0: 0 个周期 1: 1 个周期 . . .

POWER[LF_WAIT] 位域

13.8.16 POWER[OFF_WAIT] (0x100C + 0x14 * n)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD												WAIT																			
N/A												RW																			
x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1

POWER[OFF_WAIT] [31:0]

位域	名称	描述
19-0	WAIT	电源开关关闭时间，以 24M 时钟为单位，缺省值为 15 0: 0 个周期 1: 1 个周期 . . .

POWER[OFF_WAIT] 位域

13.8.17 POWER[RET_WAIT] (0x1010 + 0x14 * n)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD												WAIT																			
N/A												RW																			
x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1

POWER[RET_WAIT] [31:0]

位域	名称	描述
19-0	WAIT	memory 信息保持模式跳转时间，以 24M 时钟为单位，缺省值为 15 0: 0 个周期 1: 1 个周期 . . .

POWER[RET_WAIT] 位域

13.8.18 RESET[CONTROL] (0x1400 + 0x10 * n)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FLAG	FLAG_WAKE	RSVD																									HOLD	RSVD			RESET
RW	RW	N/A																									RW	N/A			RW
1	0	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	x	x	x	0

RESET[CONTROL] [31:0]

位域	名称	描述
31	FLAG	复位标志，指示发生过复位，此标志可通过写入 1 来清零 0: 未经历复位过程 1: 已经历复位过程
30	FLAG_WAKE	复位标志，指示发生过唤醒复位，此标志可通过写入 1 来清零 0: 未经历复位过程 1: 已经历复位过程
4	HOLD	软件复位时，复位是否自动释放 0: 软件复位时，自动释放 1: 软件复位时，复位保持，等待软件释放
0	RESET	软件复位 0: 复位已释放 1：复位有效

RESET[CONTROL] 位域

13.8.19 RESET[CONFIG] (0x1404 + 0x10 * n)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD								PRE_WAIT								RSTCLK_NUM								POST_WAIT							
N/A								RW								RW								RW							
x	x	x	x	x	x	x	x	0	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	1

RESET[CONFIG] [31:0]

位域	名称	描述
23-16	PRE_WAIT	复位前等待周期，以 24M 周期为单位 0：0 个周期 1：1 个周期 . . .
15-8	RSTCLK_NUM	复位时钟数目，必须是偶数 0: 0 周期 1: 0 周期 2: 2 周期 3: 2 周期 . . .

位域	名称	描述
7-0	POST_WAIT	复位后等待周期，以 24M 周期为单位 0：0 个周期 1：1 个周期 . . .

RESET[CONFIG] 位域

13.8.20 RESET[COUNTER] (0x140C + 0x10 * n)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD												COUNTER																			
N/A												RW																			
x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

RESET[COUNTER] [31:0]

位域	名称	描述
19-0	COUNTER	复位倒计数器，计数器值为 1 的时候触发复位，写 0 可取消复位。 计数器以 24M 时钟运行 0: 不复位 1: 1 立即复位 2：等待 1 个周期 . . .

RESET[COUNTER] 位域

13.8.21 CLOCK_CPU (0x1800 + 0x4 * n)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GLB_BUSY	LOC_BUSY	RSVD	PRESERVE	RSVD								SUB0_DIV				RSVD					MUX			DIV							
RO	RO	N/A	RW	N/A								RW				N/A					RW			RW							
0	0	x	0	x	x	x	x	x	x	x	x	0	0	0	0	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0

CLOCK_CPU [31:0]

位域	名称	描述
31	GLB_BUSY	全局时钟忙标志位 0：所有功能时钟均处于稳定工作或关闭状态 1：至少有一个功能时钟正在改变设置

位域	名称	描述
30	LOC_BUSY	当前时钟忙标志位 0: 当前时钟处于稳定工作或关闭状态 1: 当前时钟正在改变设置
28	PRESERVE	全局时钟设定使能 0: 可以选择全局时钟设定 1: 不选择任何全局时钟设定
19-16	SUB0_DIV	ahb 总线分频数，该总线时钟由 cpu 时钟分频得到 0：除以 1 1：除以 2 . . .
10-8	MUX	时钟选择 0:osc0_clk0 1:pll0_clk0 2:pll0_clk1 3:pll0_clk2 4:pll1_clk0 5:pll1_clk1 6:pll1_clk2 7:pll1_clk3
7-0	DIV	分频数 0: 除以 1 1: 除以 2 2: 除以 3 . . . 255: 除以 256

CLOCK_CPU 位域

13.8.22 CLOCK (0x1804 + 0x4 * n)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GLB_BUSY	LOC_BUSY	RSVD	PRESERVE	RSVD																	MUX			DIV							
RO	RO	N/A	RW	N/A																	RW			RW							
0	0	x	0	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0

CLOCK [31:0]

位域	名称	描述
31	GLB_BUSY	全局时钟忙标志位 0：所有功能时钟均处于稳定工作或关闭状态 1：至少有一个功能时钟正在改变设置
30	LOC_BUSY	当前时钟忙标志位 0: 当前时钟处于稳定工作或关闭状态 1: 当前时钟正在改变设置
28	PRESERVE	全局时钟设定使能 0: 可以选择全局时钟设定 1: 不选择任何全局时钟设定
10-8	MUX	时钟选择 0:osc0_clk0 1:pll0_clk0 2:pll0_clk1 3:pll0_clk2 4:pll1_clk0 5:pll1_clk1 6:pll1_clk2 7:pll1_clk3
7-0	DIV	分频数 0: 除以 1 1: 除以 2 2: 除以 3 . . . 255: 除以 256

CLOCK 位域

13.8.23 ADCCLK (0x1C00 + 0x4 * n)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GLB_BUSY	LOC_BUSY	RSVD	PRESERVE	RSVD																			MUX	RSVD							
RO	RO	N/A	RW	N/A																			RW	N/A							
0	0	x	0	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	x	x	x	x	x	x	x	x

ADCCLK [31:0]

位域	名称	描述
31	GLB_BUSY	全局时钟忙标志位 0：所有功能时钟均处于稳定工作或关闭状态 1：至少有一个功能时钟正在改变设置

位域	名称	描述
30	LOC_BUSY	当前时钟忙标志位 0: 当前时钟处于稳定工作或关闭状态 1: 当前时钟正在改变设置
28	PRESERVE	全局时钟设定使能 0: 可以选择全局时钟设定 1: 不选择任何全局时钟设定
8	MUX	时钟选择 0: ahb0 clock N 1: ana clock

ADCCLK 位域

13.8.24 DACCLK (0x1C08 + 0x4 * n)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GLB_BUSY	LOC_BUSY	RSVD	PRESERVE	RSVD																			MUX	RSVD							
RO	RO	N/A	RW	N/A																			RW	N/A							
0	0	x	0	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	x	x	x	x	x	x	x	x

DACCLK [31:0]

位域	名称	描述
31	GLB_BUSY	全局时钟忙标志位 0：所有功能时钟均处于稳定工作或关闭状态 1：至少有一个功能时钟正在改变设置
30	LOC_BUSY	当前时钟忙标志位 0: 当前时钟处于稳定工作或关闭状态 1: 当前时钟正在改变设置
28	PRESERVE	全局时钟设定使能 0: 可以选择全局时钟设定 1: 不选择任何全局时钟设定
8	MUX	时钟选择 0: ahb0 clock N 1: ana clock

DACCLK 位域

13.8.25 GLOBAL00 (0x2000)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																								MUX							
N/A																								RW							
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0

GLOBAL00 [31:0]

位域	名称	描述
7-0	MUX	全局时钟设置，每个位代表一个设置，设置在该位从 0 变成 1 的时候发生。如需二次设置，需写 0 后再次写入 1 0 位：24M 时钟 1 位：推荐设置 2 位：测试用 3 位：测试用 4 位：测试用 5 位：测试用 6 位：测试用 7 位：测试用

GLOBAL00 位域

13.8.26 MONITOR[CONTROL] (0x2400 + 0x20 * n)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VALID	RSVD			DIV_BUSY	RSVD		OUTEN	DIV								HIGH	LOW	RSVD	START	RSVD	MODE	ACCURACY	REFERENCE	SELECTION							
RW	N/A			RO	N/A		RW	RW								RW	RW	N/A	RW	N/A	RW	RW	RW	RW							
0	x	x	x	0	x	x	0	0	0	0	0	0	0	0	0	0	0	x	0	x	0	0	0	0	0	0	0	0	0	0	0

MONITOR[CONTROL] [31:0]

位域	名称	描述
31	VALID	结果有效 0：结果不可用 1：结果可用
27	DIV_BUSY	0：分频器稳定工作 1：分频器正在改变设置
24	OUTEN	输出使能 0：禁止输出 1：允许输出

位域	名称	描述
23-16	DIV	输出分频器 0：除以 1 1：除以 2 . . .
15	HIGH	监测频率高于上限 0：频率不高于上限 1：频率高于上限
14	LOW	监测频率低于下限 0：频率不低于下限 1：频率低于下限
12	START	开始测量 0：禁止测量 1：开始测量
10	MODE	模式选择 0：比较模式，测量结果与 min 和 max 寄存器的数值作比较 1：记录模式，出现过的最大最小值记录在 min 和 max 寄存器里面
9	ACCURACY	测量精度 0：精确到 1KHz 1：精确到 1Hz
8	REFERENCE	参考时钟选择 0：32KHz 1：24MHz
7-0	SELECTION	时钟测量选择

MONITOR[CONTROL] 位域

13.8.27 MONITOR[CURRENT] (0x2404 + 0x20 * n)

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

MONITOR[CURRENT] [31:0]

位域	名称	描述
31-0	FREQUENCY	测量结果，以 Hz 为单位

MONITOR[CURRENT] 位域

13.8.28 MONITOR[LOW_LIMIT] (0x2408 + 0x20 * n)

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

MONITOR[LOW_LIMIT] [31:0]

位域	名称	描述
31-0	FREQUENCY	时钟频率下限，以 Hz 为单位

MONITOR[LOW_LIMIT] 位域

13.8.29 MONITOR[HIGH_LIMIT] (0x240C + 0x20 * n)

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

MONITOR[HIGH_LIMIT] [31:0]

位域	名称	描述
31-0	FREQUENCY	时钟频率上限，以 Hz 为单位

MONITOR[HIGH_LIMIT] 位域

13.8.30 CPU[LP] (0x2800 + 0x400 * n)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WAKE_CNT								RSVD							HALT	RSVD		WAKE	EXEC	RSVD	WAKE_FLAG	SLEEP_FLAG	RESET_FLAG	RSVD							MODE
RW								N/A							RW	N/A		RO	RO	N/A	RW	RW	RW	N/A							RW
0	0	0	0	0	0	0	0	x	x	x	x	x	x	x	0	x	x	0	1	x	0	0	0	x	x	x	x	x	x	0	0

CPU[LP] [31:0]

位域	名称	描述
31-24	WAKE_CNT	CPU 唤醒计数器，累计到 255 之后将不再增加，写 0 可清零该计数器 0：未经历过唤醒 1：唤醒过 1 次 2：唤醒过 2 次 . . .
16	HALT	CPU 停止运行 0：CPU 允许运行 1：CPU 启动时或 WFI 之后则不再唤醒
13	WAKE	唤醒状态 0：无唤醒事件 1：有唤醒时间
12	EXEC	运行状态 0：睡眠 1：运行
10	WAKE_FLAG	唤醒标志，写 1 清 0：CPU 未经历过唤醒 1：CPU 经历过唤醒
9	SLEEP_FLAG	睡眠标志，写 1 清 0：CPU 未经历过睡眠 1：CPU 经历过睡眠
8	RESET_FLAG	复位标志，写 1 清 0：CPU 未经历过复位 1：CPU 经历过复位
1-0	MODE	低功耗模式，设置 CPUWFI 之后的行为 00：等待，WFI 后仅关闭核心时钟 01：停止，WFI 后触发系统的低功耗过程 10：运行，WFI 后保持不变 11：保留

CPU[LP] 位域

13.8.31 CPU[LOCK] (0x2804 + 0x400 * n)

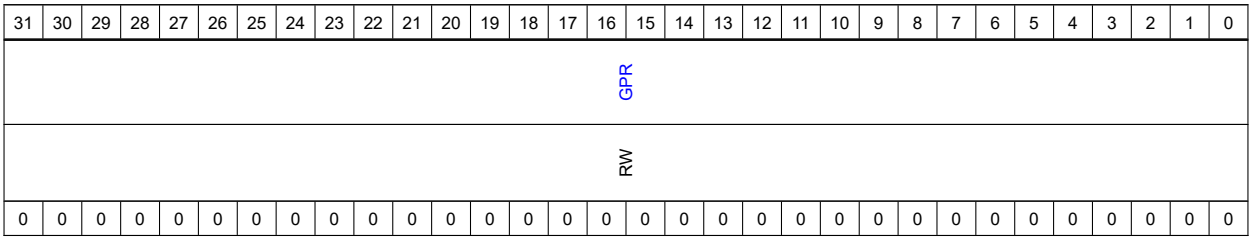
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																GPR														LOCK	RSVD
N/A																RW														RW	N/A
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	x

CPU[LOCK] [31:0]

位域	名称	描述
15-2	GPR	锁定寄存器，每一位对应一个 GPR 0：未锁定 1：锁定
1	LOCK	锁定 LOCK 寄存器 0：未锁定 1：锁定

CPU[LOCK] 位域

13.8.32 CPU[GPR] (0x2808 + 0x400 * n + 0x4 * m)

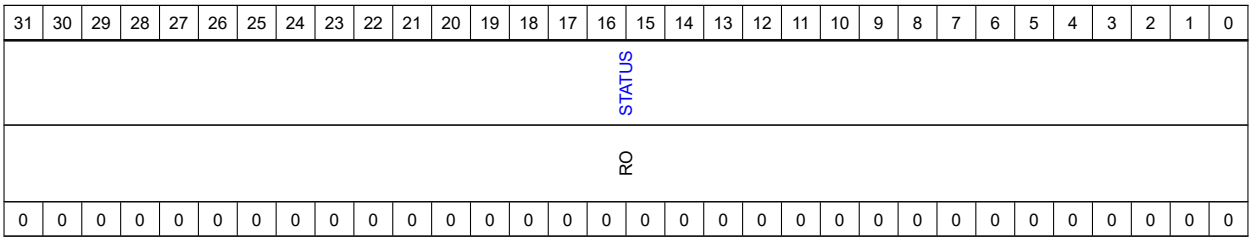


CPU[GPR] [31:0]

位域	名称	描述
31-0	GPR	通用寄存器，主要用于休眠和唤醒控制

CPU[GPR] 位域

13.8.33 CPU[WAKEUP_STATUS] (0x2840 + 0x400 * n + 0x4 * m)

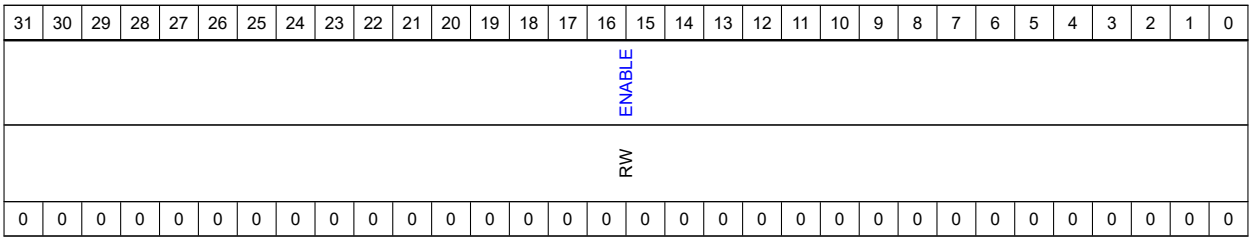


CPU[WAKEUP_STATUS] [31:0]

位域	名称	描述
31-0	STATUS	中断状态

CPU[WAKEUP_STATUS] 位域

13.8.34 CPU[WAKEUP_ENABLE] (0x2880 + 0x400 * n + 0x4 * m)



CPU[WAKEUP_ENABLE] [31:0]

位域	名称	描述
31-0	ENABLE	唤醒使能寄存器，设置 1 则允许该中断唤醒 CPU

CPU[WAKEUP_ENABLE] 位域

14 锁相环控制器 PLLCTL

本章节描述了锁相环控制器 PLLCTL 的主要功能与特性。

14.1 特性总结

本产品包含一个锁相环控制器 PLLCTL，用于配置片上的 PLL。主要特性如下：

- XTAL 振荡器配置
- 片上 PLL 工作模式配置
  - 配置 PLL 参考时钟
  - 配置 PLL 工作频率
  - 配置 PLL 扩频功能
- 配置输出分频
- 支持硬件自动控制
- 支持运行时修改频率

14.2 功能描述

本章节描述了锁相环控制器 PLLCTL 的主要功能。

14.2.1 XTAL 振荡器

XTAL 振荡器可以读取 XTAL 的工作状态，用户可以查询 XTAL 寄存器的 ENABLE 标志位，置 1 表示作为参考时钟源的晶振打开，查询 XTAL 寄存器的 RESPONSE 标志位，置 1 时，表示参考时钟源状态稳定。可以设置 XTAL 启动所需的时间，该时间的默认值为 3ms，这一设置影响关闭 XTAL 低功耗模式的进出时间。

14.2.2 PLL 参考时钟设置

锁相环参考时钟可以通过 REFSEL 进行选择：

- 0：选择 XTAL
- 1：选择 RC24M

该选择可以在 PLL 工作时修改，工作时修改会引起 PLL 输出时钟频率的短暂波动。

14.2.3 PLL 工作频率配置

锁相环压控振荡器输出频率如下：

$$F_{vco} = F_{ref} \times (MFI + (MFN \div MFD))$$

MFD 的缺省值为 240000000，MFN 上的数字 1 代表 0.1Hz。

PLL 寄存器的 MFI 和 MFN 字段支持运行时修改，修改后的可通过读取 BUSY 标志判断 PLL 是否在新的设置下稳定工作。PLL 在切换到新设置时不会停止时钟输出。MFD 不支持运行时修改，若修改，则在 PLL 关闭再打开后生效。

14.2.4 PLL 扩谱模式

PLL 的扩谱模式可以通过 SPREAD 使能：

- 0：禁止扩谱
- 1：使能扩谱

扩谱可以在 PLL 工作时使能或禁止。PLL 的扩谱为向下扩谱，当 PLL 的扩谱使能时，VCO 振荡频率在扩谱上下限之间扫频，每一个 24M 时钟周期增加或减少

$$F_{ref} \times (STEP \div MFD)$$

。

VCO 振荡频率上限为：

$$F_{vco} = F_{ref} \times (MFI + (MFN \div MFD))$$

VCO 振荡频率下限为：

$$F_{vco} = F_{ref} \times (MFI + ((MFN - STOP) \div MFD))$$

PLL 工作在扩谱模式下，STEP 和 STOP 不支持工作时改变设置。若此时修改，新设置将在禁止扩谱后再次使能，或者关闭 PLL 再次打开后生效。

14.2.5 PLL 输出分频器

VCO 的时钟经过分频后，输出给时钟模块产生系统时钟。该分频器为小数分频器，分频系数以 0.2 步进。

$$F_{out} = F_{vco} \div (1 + 0.2 \times DIV)$$

DIV 支持工作时修改工作频率。

14.3 PLLCTL 寄存器

14.3.1 寄存器说明

PLLCTLv2 的寄存器列表如下：

PLLCTLV2 base address: 0xF40C0000

地址偏移	名称	描述	复位值
0x0000	XTAL	晶体振荡器配置和状态寄存器	0x0001FFFF
0x0080	PLL[PLL0][MFI]	锁相环 0 倍频数寄存器	0x00000010
0x0084	PLL[PLL0][MFN]	锁相环 0 分子寄存器	0x09896800
0x0088	PLL[PLL0][MFD]	锁相环 0 分母寄存器	0x0E4E1C00
0x008C	PLL[PLL0][SS_STEP]	锁相环 0 扩频补偿寄存器	0x00000000
0x0090	PLL[PLL0][SS_STOP]	锁相环 0 扩频范围寄存器	0x00000000
0x0094	PLL[PLL0][CONFIG]	锁相环 0 配置寄存器	0x00000000
0x0098	PLL[PLL0][LOCKTIME]	锁相环 0 锁定时间寄存器	0x000009C4
0x009C	PLL[PLL0][STEPTIME]	锁相环 0 步进时间寄存器	0x000009C4
0x00A0	PLL[PLL0][ADVANCED]	锁相环 0 高级配置寄存器	0x00000000
0x00C0	PLL[PLL0][DIV][DIV0]	锁相环 0 分频输出 0 配置寄存器	0x00000000

地址偏移	名称	描述	复位值
0x00C4	PLL[PLL0][DIV][DIV1]	锁相环 0 分频输出 1 配置寄存器	0x00000000
0x00C8	PLL[PLL0][DIV][DIV2]	锁相环 0 分频输出 2 配置寄存器	0x00000000
0x0100	PLL[PLL1][MFI]	锁相环 1 倍频数寄存器	0x00000018
0x0104	PLL[PLL1][MFN]	锁相环 1 分子寄存器	0x00000000
0x0108	PLL[PLL1][MFD]	锁相环 1 分母寄存器	0x0E4E1C00
0x010C	PLL[PLL1][SS_STEP]	锁相环 1 扩频补偿寄存器	0x00000000
0x0110	PLL[PLL1][SS_STOP]	锁相环 1 扩频范围寄存器	0x00000000
0x0114	PLL[PLL1][CONFIG]	锁相环 1 配置寄存器	0x00000000
0x0118	PLL[PLL1][LOCKTIME]	锁相环 1 锁定时间寄存器	0x000009C4
0x011C	PLL[PLL1][STEPTIME]	锁相环 1 步进时间寄存器	0x000009C4
0x0120	PLL[PLL1][ADVANCED]	锁相环 1 高级配置寄存器	0x000009C4
0x0140	PLL[PLL1][DIV][DIV0]	锁相环 1 分频输出 0 配置寄存器	0x00000001
0x0144	PLL[PLL1][DIV][DIV1]	锁相环 1 分频输出 1 配置寄存器	0x00000004
0x0180	PLL[PLL2][MFI]	锁相环 2 倍频数寄存器	0x0000001E
0x0184	PLL[PLL2][MFN]	锁相环 2 分子寄存器	0x00182B80
0x0188	PLL[PLL2][MFD]	锁相环 2 分母寄存器	0x0E4E1C00
0x018C	PLL[PLL2][SS_STEP]	锁相环 2 扩频补偿寄存器	0x00000000
0x0190	PLL[PLL2][SS_STOP]	锁相环 2 扩频范围寄存器	0x00000000
0x0194	PLL[PLL2][CONFIG]	锁相环 2 配置寄存器	0x00000000
0x0198	PLL[PLL2][LOCKTIME]	锁相环 2 锁定时间寄存器	0x000009C4
0x019C	PLL[PLL2][STEPTIME]	锁相环 2 步进时间寄存器	0x000009C4
0x01A0	PLL[PLL2][ADVANCED]	锁相环 2 高级配置寄存器	0x000009C4
0x01C0	PLL[PLL2][DIV][DIV0]	锁相环 2 分频输出 0 配置寄存器	0x00000002
0x01C4	PLL[PLL2][DIV][DIV1]	锁相环 2 分频输出 1 配置寄存器	0x00000003

表 28: PLLCTLV2 寄存器列表

PLLCTLv2 的寄存器详细说明如下：

14.3.2 XTAL (0x0)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BUSY	RSVD	RESPONSE	ENABLE	RSVD								RAMP_TIME																			
RO	N/A	RO	RO	N/A								RW																			
0	x	0	0	x	x	x	x	x	x	x	x	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

XTAL [31:0]

位域	名称	描述
31	BUSY	忙标志 0: 晶体振荡器正常工作或处于关闭状态 1: 晶体振荡器正在启动或正在停止
29	RESPONSE	振荡器状态 0: 振荡器未稳定 1: 振荡器已稳定
28	ENABLE	Crystal oscillator enable status 0: Oscillator is off 1: Oscillator is on
19-0	RAMP_TIME	晶体振荡器的启动时间，以内部 RC24M 时钟周期计数 0: 0 周期 1: 1 周期 2: 2 周期 1048575: 1048575 周期

XTAL 位域

14.3.3 PLL[MFI] (0x80 + 0x80 * n)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BUSY	RSVD	RESPONSE	ENABLE	RSVD																					MFI						
RO	N/A	RO	RO	N/A																					RW						
0	x	0	0	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	1	0	0	0	0

PLL[MFI] [31:0]

位域	名称	描述
31	BUSY	忙标志 0: 锁相环正常工作或处于关闭状态 1: 锁相环正在启动或正在停止
29	RESPONSE	锁相环状态 0: 锁相环未稳定 1: 锁相环已稳定
28	ENABLE	锁相环控制状态 0: 锁相环打开 1: 锁相环关闭

位域	名称	描述
6-0	MFI	锁相环反馈分频器数值，有效范围是 13-42, f=fref*(mfi + mfn/mfd)。支持运行时修改。 0-15: 无效 16: 分频数为 16 17: 分频数为 17 . . . 42: 分频数为 42 43~: 无效

PLL[MFI] 位域

14.3.4 PLL[MFN] (0x84 + 0x80 * n)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																MFN															
N/A																RW															
x	x	0	0	1	0	0	1	1	0	0	0	1	0	0	1	0	1	1	0	1	0	0	0	0	0	0	0	0	0	0	0

PLL[MFN] [31:0]

位域	名称	描述
29-0	MFN	分数分频的分子,f=fref*(mfi + mfn/mfd)。支持运行时修改。

PLL[MFN] 位域

14.3.5 PLL[MFD] (0x88 + 0x80 * n)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																MFD															
N/A																RW															
x	x	0	0	1	1	1	0	0	1	0	0	1	1	1	0	0	0	0	1	1	1	0	0	0	0	0	0	0	0	0	0

PLL[MFD] [31:0]

位域	名称	描述
29-0	MFD	分数分频的分母,f=fref*(mfi + mfn/mfd)。不支持运行时修改，如果运行时被修改，新设置在下次开启时生效。

PLL[MFD] 位域

14.3.6 PLL[SS_STEP] (0x8C + 0x80 * n)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																STEP															
N/A																RW															
x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

PLL[SS_STEP] [31:0]

位域	名称	描述
29-0	STEP	扩频调制的步长 当锁相环开启扩频，不支持运行时修改。如果运行时被修改，新设置在下次开启时生效。

PLL[SS_STEP] 位域

14.3.7 PLL[SS_STOP] (0x90 + 0x80 * n)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																STOP															
N/A																RW															
x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

PLL[SS_STOP] [31:0]

位域	名称	描述
29-0	STOP	扩频调制的范围 当锁相环开启扩频，不支持运行时修改。如果运行时被修改，新设置在下次开启时生效。

PLL[SS_STOP] 位域

14.3.8 PLL[CONFIG] (0x94 + 0x80 * n)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																							SPREAD	RSVD							REFSEL
N/A																							RW	N/A							RW
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	x	x	x	x	x	x	x	0

PLL[CONFIG] [31:0]

位域	名称	描述
8	SPREAD	开启扩频功能。支持运行时修改。

位域	名称	描述
0	REFSEL	选择参考时钟。支持运行时修改，但是应用必须考虑到可能发生的频率以及抖动的改变。若 MFN 一同改变，应用应确保切换参考时钟时，锁相环处于锁定状态。 0: 晶体振荡器 1: 阻容振荡器

PLL[CONFIG] 位域

14.3.9 PLL[LOCKTIME] (0x98 + 0x80 * n)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																LOCKTIME															
N/A																RW															
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	1	0	0	1	1	1	0	0	0	1	0	0

PLL[LOCKTIME] [31:0]

位域	名称	描述
15-0	LOCKTIME	锁定时间，以 24M 时钟计数，缺省值为 2500。如果锁相环启动过程中 MFI 发生了改变，锁定时间会延长。

PLL[LOCKTIME] 位域

14.3.10 PLL[STEPTIME] (0x9C + 0x80 * n)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																STEPTIME															
N/A																RW															
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	1	0	0	1	1	1	0	0	0	1	0	0

PLL[STEPTIME] [31:0]

位域	名称	描述
15-0	STEPTIME	运行时修改频率时的步进间隔，以 24M 时钟计数，缺省值为 2500。

PLL[STEPTIME] 位域

14.3.11 PLL[ADVANCED] (0xA0 + 0x80 * n)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD			SLOW	RSVD			DITHER	RSVD																							
N/A			RW	N/A			RW	N/A																							
x	x	x	0	x	x	x	0	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x

PLL[ADVANCED] [31:0]

位域	名称	描述
28	SLOW	禁止锁定加速，使用慢速锁定模式。此模式对干扰有更强的免疫能力。软件需同时修改锁定时间，步进时间不需要修改。 0: 加速锁定，锁定时间为 100us 1: 慢速锁定，锁定时间为 400us
24	DITHER	开启抖动功能

PLL[ADVANCED] 位域

14.3.12 PLL[DIV] (0xC0 + 0x80 * n + 0x4 * m)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BUSY	RSVD	RESPONSE	ENABLE	RSVD																						DIV					
RO	N/A	RO	RO	N/A																						RW					
0	x	0	0	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0

PLL[DIV] [31:0]

位域	名称	描述
31	BUSY	忙标志 0: 分频器正常工作或处于关闭状态 1: 分频器正在启动或正在停止
29	RESPONSE	分频器状态 0: 分频器未稳定 1: 分频器已稳定
28	ENABLE	分频器控制状态 0: 分频器打开 1: 分频器关闭
5-0	DIV	分频因子，分频数为 DIV/5 + 1 0: 除以 1 1: 除以 1.2 2: 除以 1.4 . . . 63: 除以 13.6

位域	名称	描述

PLL[DIV] 位域

15 低功耗管理

本章节介绍系统级低功耗的管理和应用。

15.1 运行模式 RUN

处理器正常运行程序时处于运行模式，运行模式下系统支持的降低功耗的方式有：

- 正确设置 SYSCTL 的 GROUPx 寄存器，仅使能必要的功能模块，则不使用的模块会自动关闭以节省功耗
- 通过配置 PLL 的 FBDIV、POSTDIV 和 SYSCTL 中功能时钟的分频系数，适当降低 CPU 和功能模块的时钟频率
- 如果系统电源域使用 DCDC 供电，可根据需要，配置 PCFG 模块来适当降低 DCDC 输出电压

15.2 等待模式 WAIT

如果 SYSCTL 寄存器 CPU[CPUx][LP] 的 MODE 位为 b00，则 CPU 执行 WFI 指令后即进入等待模式，CPU 核心时钟会被关闭。但在收到中断时 CPU 能够立即唤醒进行处理。

15.3 停止模式 STOP

如果 SYSCTL 寄存器 CPU[CPUx][LP] 的 MODE 位为 b01，则 CPU 执行 WFI 指令后即进入停止模式。停止模式下 SYSCTL 模块将根据配置关闭系统电源域内的部分资源节点。

与等待模式相比，停止模式允许关闭更多的资源，且提供灵活的配置选项，用户需根据功耗和唤醒时间的要求在 SYSCTL 模块中进行合理的配置。

考虑最简单的应用场景 1：系统只使用 CPU0，仅通过 SYSCTL 的 GROUP0 寄存器连接使能了必要的功能模块，将 RETENTION[CPU0] 寄存器修改为全 0。那么首先，不使用的资源节点在运行模式下已经自动关闭。在 CPU 执行 WFI 进入停止模式后，CPU0 的时钟、CPU0 子系统的电源、CPU0 使用的各功能模块、功能时钟、其他子系统电源、PLL 和 24M 晶振等都会依次关闭，仅保留 SYSCTL 的功能以等待唤醒。

应用场景 2：系统只使用 CPU0，如果要求 CPU0 进入停止模式后自身的寄存器状态不丢失即 CPU0 子系统不掉电、不复位，且要求 24M 晶振和 PLL 不关闭，以此实现较快的唤醒速度，则需要设置 SYSCTL 的 RETENTION[CPU0] 寄存器，将 CPU0 寄存器、24M 晶振和 PLL 对应的位置 1。

应用场景 3：系统只使用 CPU0，如果要求 CPU0 进入停止模式后某个功能模块如 GPTMR0 保持运行，而该模块不在 RETENTION 寄存器的管理范围，则需设置 SYSCTL 的 RESOURCE[GPTMR0] 寄存器，将 MODE 位设为 1 使其一直保持开启。

注意，RETENTION[CPU0] 寄存器的默认值是保留 CPU0 子系统和系统电源域的电源。

中断能够将系统从停止模式唤醒，这些中断可以来自系统电源域中仍然保持工作的功能模块，也可以来自电源管理域，需确保期望唤醒的中断源已在 SYSCTL 的 CPU[CPUx][WAKEUP_ENABLE][ENABLEx] 寄存器中被使能。

总之，配置停止模式可能涉及的步骤如下：

- 配置 SYSCTL 寄存器 CPU[CPUx][LP] 的 MODE 位为 b01
- 配置 SYSCTL 寄存器 RETENTION[x]，使部分资源节点保持开启
- 配置 SYSCTL 寄存器 RESOURCE[x]，使更多的资源节点保持开启
- 配置 SYSCTL 寄存器 CPU[CPUx][WAKEUP_ENABLE][ENABLEx]，指定唤醒中断源

- 配置用来产生唤醒中断的模块，设定中断触发条件并使能中断
- 执行 WFI 指令

15.4 休眠模式 STANDBY

休眠模式下整个系统电源域不再运行，其可能在复位状态或掉电状态。如果 SYSCTL 的配置中 RETENTION 寄存器没有任何需要保持运行的资源节点，RESOURCE 寄存器中也没有任何强制开启的资源节点，那么当全部 CPU 进入停止模式后整个系统电源域便不再有运行需求，SYSCTL 模块在关闭了全部的资源节点后，会向电源管理域发出关闭系统电源域的请求。请求发生后，首先系统电源域会被冷复位，之后 PCFG 模块会根据 POWER_TRAP 寄存器的设置执行操作：

如果使用 DCDC 为系统电源域供电，则 DCDC 会根据表 21进行状态切换：正常运行、关闭或进入低功耗模式，其中 DCDC 在低功耗模式下能够配置为提供较低的输出电压。

进入休眠模式后系统只能被电源管理域内模块的中断唤醒，唤醒源详见小节 12.2.5。

另电源管理域中的不使用的模块可以通过设置 PCFG 的 SCG_CTRL 寄存器来关闭其时钟，从而在休眠模式下进一步降低系统功耗，详见小节 12.2.4。

总之，配置休眠模式可能涉及的步骤如下：：

- 配置 SYSCTL 寄存器 CPU[CPUx][LP] 的 MODE 位为 b01
- 配置 SYSCTL 寄存器 RETENTION[x] 寄存器为全 0
- 配置所有的 SYSCTL 寄存器 RESOURCE[x] 寄存器中 MODE 位都不为 1
- 配置 PCFG 模块 POWER_TRAP 寄存器的 TRIGGERED 位和 RETENTION 位，以设置系统电源域电源的保持、关闭或降低电压
- 如果使用外部电源为系统电源域供电且使用引脚控制其行为，则需要在 BIOC 中配置 PZ10 或 PZ11 的功能选择
- 配置 PCFG 模块 WAKE_MASK 寄存器以使能唤醒源
- 配置电源管理域中唤醒源模块的中断触发条件并使能中断
- 执行 WFI 指令

15.5 关机模式 SHUTDOWN

关机模式下仅电源管理域的 PDGO 模块能够正常工作，电源管理域除 PDGO 外的其他模块和系统电源域都处于复位或掉电状态。

进入关机模式方式为软件写寄存器：

- 向倒计时关机寄存器 DGO_TURNOFF 写入一个非零倒计时值，在倒计时结束后，系统进入关机模式

根据芯片的外部电源供电结构考虑以下场景：

场景 1：单一 3.3V 供电，且使用 DCDC 为系统电源域供电。则进入关机模式时，电源管理域除 DGO 外的模块进入复位状态，DCDC 关闭从而系统电源域掉电。

场景 2：单一 3.3V 供电，使用外部电源为系统电源域供电。则进入关机模式时，电源管理域除 DGO 外的模块和系统电源域都进入复位状态。

关机模式下系统只能从 PDGO 模块唤醒，退出关机模式的方式详见 小节 11.4.2。

15.6 CPU 本地存储器 memory retention

本产品的 CPU 本地存储器 ILM 和 DLM 支持 memory retention 功能，当系统电源域有电（VDD_SOC 有电），CPU0 子系统掉电（VDD_CPU 掉电）时，CPU 本地存储器 ILM 和 DLM 的 memory core 保持有电状态，存储的内容能够保留下来，方便未来的上电恢复。该功能无需配置，也不是一种低功耗模式，上述模式中只要满足系统电源域有电，CPU0 子系统掉电就可自动实现该功能。

15.7 低功耗总结

图 10描述了各低功耗模式的控制流程和作用范围。

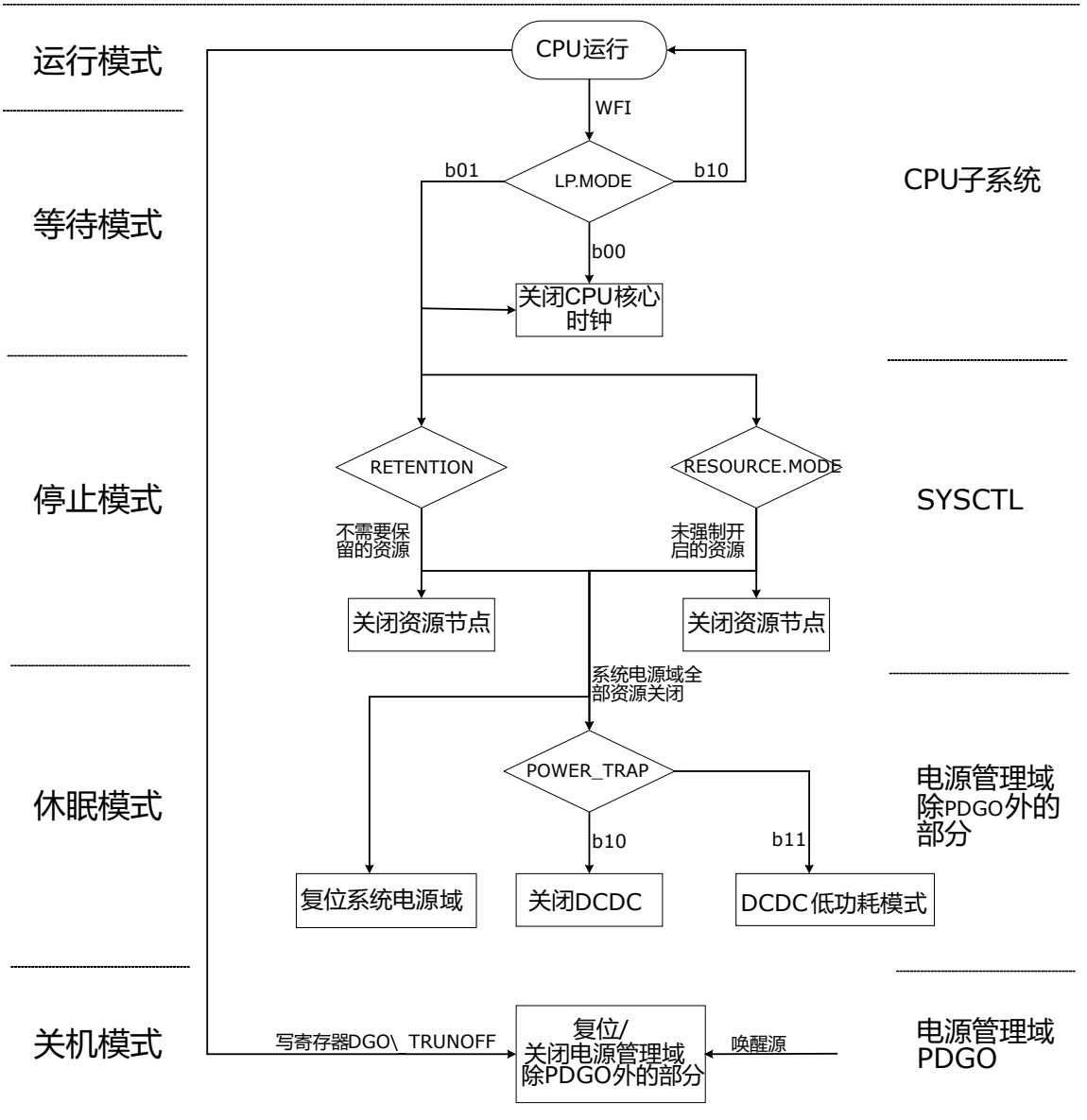


图 10: 低功耗模式流程图

16 系统内存映射

本章节介绍本产品的系统内存映射。

16.1 系统内存映射 System Memory Map

本产品的系统内存映射表如下：

起始地址	结束地址	地址空间	名称	描述
0x00000000	0x0001FFFF	128 KBytes	CPU0_ILM_SLV	指令本地存储器
0x00060000	0x0007FFFF	128 KBytes	CPU0_ILM_SLV_ALIAS	
0x00080000	0x0009FFFF	128 KBytes	CPU0_DLM_SLV	数据本地存储器
0x000A0000	0x000BFFFF	128 KBytes	CPU0_DLM_SLV_ALIAS	
0x000C0000	0x000C00FF	256 Bytes	FGPIO	快速 GPIO 控制器
0x01040000	0x0105FFFF	128 KBytes	CPU0_ILM_SLV_ALS	CPU ILM 镜像
0x01060000	0x0107FFFF	128 KBytes	CPU0_DLM_SLV_ALS	CPU DLM 镜像
0x20000000	0x2001FFFF	128 KBytes	ROM	只读存储器 ROM
0x30000000	0x300FFFFF	1 MBytes	DM	DEBUG 调试模块
0x80000000	0x8FEFFFFF	255 MBytes	XPI0	串行总线控制器 XPI 存储空间
0xE4000000	0xE43FFFFF	4 MBytes	PLIC	平台中断控制器 PLIC
0xE6000000	0xE60FFFFF	1 MBytes	MCHTMR	机器定时器
0xE6400000	0xE67FFFFF	4 MBytes	PLICSW	平台软件中断控制器
0xF0000000	0xF0003FFF	16 KBytes	GPTMR0	定时器 0
0xF0004000	0xF0007FFF	16 KBytes	GPTMR1	定时器 1
0xF0008000	0xF000BFFF	16 KBytes	GPTMR2	定时器 2
0xF000C000	0xF000FFFF	16 KBytes	GPTMR3	定时器 3
0xF0020000	0xF0023FFF	16 KBytes	LIN0	串行通讯网络 LIN0
0xF0024000	0xF0027FFF	16 KBytes	LIN1	串行通讯网络 LIN1
0xF0028000	0xF002BFFF	16 KBytes	LIN2	串行通讯网络 LIN2
0xF002C000	0xF002FFFF	16 KBytes	LIN3	串行通讯网络 LIN3
0xF0040000	0xF0043FFF	16 KBytes	UART0	通用异步收发器 UART0
0xF0044000	0xF0047FFF	16 KBytes	UART1	通用异步收发器 UART1
0xF0048000	0xF004BFFF	16 KBytes	UART2	通用异步收发器 UART2
0xF004C000	0xF004FFFF	16 KBytes	UART3	通用异步收发器 UART3
0xF0050000	0xF0053FFF	16 KBytes	UART4	通用异步收发器 UART4
0xF0054000	0xF0057FFF	16 KBytes	UART5	通用异步收发器 UART5
0xF0058000	0xF005BFFF	16 KBytes	UART6	通用异步收发器 UART6
0xF005C000	0xF005FFFF	16 KBytes	UART7	通用异步收发器 UART7
0xF0060000	0xF0063FFF	16 KBytes	I2C0	集成电路总线 I2C0

起始地址	结束地址	地址空间	名称	描述
0xF0064000	0xF0067FFF	16 KBytes	I2C1	集成电路总线 I2C1
0xF0068000	0xF006BFFF	16 KBytes	I2C2	集成电路总线 I2C2
0xF006C000	0xF006FFFF	16 KBytes	I2C3	集成电路总线 I2C3
0xF0070000	0xF0073FFF	16 KBytes	SPI0	串行外设总线 SPI0
0xF0074000	0xF0077FFF	16 KBytes	SPI1	串行外设总线 SPI1
0xF0078000	0xF007BFFF	16 KBytes	SPI2	串行外设总线 SPI2
0xF007C000	0xF007FFFF	16 KBytes	SPI3	串行外设总线 SPI3
0xF0080000	0xF0083FFF	16 KBytes	CRC	循环冗余校验 CRC
0xF0090000	0xF0093FFF	16 KBytes	TSNS	温度传感器 TEMPERATURE SENSER
0xF00A0000	0xF00A3FFF	16 KBytes	MBX0A	信箱 MBX0A
0xF00A4000	0xF00A7FFF	16 KBytes	MBX0B	信箱 MBX0B
0xF00B0000	0xF00B3FFF	16 KBytes	WDG0	看门狗 WDG0
0xF00B4000	0xF00B7FFF	16 KBytes	WDG1	看门狗 WDG1
0xF00C0000	0xF00C3FFF	16 KBytes	MISC	杂类控制寄存器
0xF00C4000	0xF00C7FFF	16 KBytes	DMAMUX	DMA 请求路由器 DMAMUX
0xF00C8000	0xF00CBFFF	16 KBytes	HDMA	DMA 控制寄存器 HDMA
0xF00D0000	0xF00D3FFF	16 KBytes	GPIO0	GPIO 控制器 GPIO0
0xF00D8000	0xF00DBFFF	16 KBytes	GPIOM	GPIO 管理器
0xF0280000	0xF0283FFF	16 KBytes	MCAN0	控制器局域网 CAN0
0xF0284000	0xF0287FFF	16 KBytes	MCAN1	控制器局域网 CAN1
0xF0288000	0xF028BFFF	16 KBytes	MCAN2	控制器局域网 CAN2
0xF028C000	0xF028FFFF	16 KBytes	MCAN3	控制器局域网 CAN3
0xF02FC000	0xF02FFFFF	16 KBytes	PTPC	高精度时间同步协议控制器 PTPC
0xF0300000	0xF0303FFF	16 KBytes	QEI0	正交编码器输入接口 QEI0
0xF0304000	0xF0307FFF	16 KBytes	QEI1	正交编码器输入接口 QEI1
0xF0308000	0xF030BFFF	16 KBytes	QEO0	正交编码器输出接口 QEI0
0xF030C000	0xF030FFFF	16 KBytes	QEO1	正交编码器输出接口 QEI1
0xF0310000	0xF0313FFF	16 KBytes	MMC0	运动管理控制器 MMC0
0xF0314000	0xF0317FFF	16 KBytes	MMC1	运动管理控制器 MMC1
0xF0318000	0xF031BFFF	16 KBytes	PWM0	PWM 定时器 PWM0
0xF031C000	0xF031FFFF	16 KBytes	PWM1	PWM 定时器 PWM1
0xF0320000	0xF0323FFF	16 KBytes	RDC	旋转编码器接口 RDC

起始地址	结束地址	地址空间	名称	描述
0xF0324000	0xF0327FFF	16 KBytes	PLB	可编程逻辑阵列 PLB
0xF0328000	0xF032BFFF	16 KBytes	SYNT	同步定时器 SYNT
0xF032C000	0xF032FFFF	16 KBytes	SEI	穿行编码器接口 SEI
0xF033C000	0xF033FFFF	16 KBytes	TRGM0	电机通用寄存器 GPR
0xF0400000	0xF0407FFF	32 KBytes	HRAM	AHB SRAM
0xF3000000	0xF3003FFF	16 KBytes	XPI0	串行总线控制器 XPI
0xF300C000	0xF300FFFF	16 KBytes	USB0	通用串行总线 USB
0xF3014000	0xF3017FFF	16 KBytes	ROMC	只读存储器控制 ROMC
0xF3040000	0xF3043FFF	16 KBytes	SDP	安全数据处理器 SDP
0xF3044000	0xF3047FFF	16 KBytes	SEC	安全管理器
0xF3048000	0xF304BFFF	16 KBytes	MON	安全监视器
0xF304C000	0xF304FFFF	16 KBytes	RNG	随机数发生器 RNG
0xF3050000	0xF3053FFF	16 KBytes	OTP	熔丝 FUSE
0xF3054000	0xF3057FFF	16 KBytes	KEYM	密钥管理器 KMAN
0xF3080000	0xF3083FFF	16 KBytes	ADC0	模数转换器 ADC0
0xF3084000	0xF3087FFF	16 KBytes	ADC1	模数转换器 ADC1
0xF3090000	0xF3093FFF	16 KBytes	DAC0	数模转换器 DAC0
0xF3094000	0xF3097FFF	16 KBytes	DAC1	数模转换器 DAC1
0xF30A0000	0xF30A3FFF	16 KBytes	OPAMP0	运算放大器 OPAMP0
0xF30A4000	0xF30A7FFF	16 KBytes	OPAMP1	运算放大器 OPAMP1
0xF30B0000	0xF30B3FFF	16 KBytes	ACMP	模拟比较器 ACMP
0xF4000000	0xF403FFFF	256 KBytes	SYSCTL	系统控制模块
0xF4040000	0xF407FFFF	256 KBytes	IOC	IO 控制器
0xF40C0000	0xF40FFFFF	256 KBytes	PLLCTLV2	锁相环控制器
0xF4100000	0xF4103FFF	16 KBytes	PPOR	电源管理域复位模块
0xF4104000	0xF4107FFF	16 KBytes	PCFG	电源管理域配置寄存器
0xF4110000	0xF4113FFF	16 KBytes	PGPR0	电源管理域通用寄存器 0
0xF4114000	0xF4117FFF	16 KBytes	PGPR1	电源管理域通用寄存器 1
0xF4118000	0xF411BFFF	16 KBytes	PIOC	电源管理域 IO 控制器
0xF411C000	0xF411FFFF	16 KBytes	PGPIO	电源管理域 GPIO 控制器
0xF4120000	0xF4123FFF	16 KBytes	PTMR	电源管理域定时器
0xF4124000	0xF4127FFF	16 KBytes	PUART	电源管理域通用异步收发器
0xF4128000	0xF412BFFF	16 KBytes	PWDG	电源管理域看门狗
0xF4134000	0xF4137FFF	16 KBytes	PDGO	电源管理域 DGO 寄存器

表 29: 地址空间分配

17 OTP 映射表

本产品的 OTP 映射表如下：

字	名称	位置	描述	类型
0	HARD_LOCK	[0:31]	熔丝值锁定	安全
1	LIFECYCLE_B	[28:31]	生命周期	安全
1	TCU_DISABLE	[19:19]	禁止测试	安全
1	DEBUG_DISABLE	[17:17]	禁止 CPU 调试	安全
1	JTAG_DISABLE	[16:16]	禁止 JTAG 端口	安全
1	PUK_REVOKE	[8:15]	公钥销毁	安全
1	LIFECYCLE_A	[0:3]	生命周期	安全
2	EXIP0_RESTRICT	[0:0]	限制 EXIP0 寄存器访问	安全
3	SW_VER	[0:31]	软件版本	安全
8	DIE_TRACE	[0:127]	芯片识别数据	识别
12	DEBUG_KEY	[0:127]	调试密钥	密钥
21	TSNS_BASE	[0:11]	温度传感器基准	通用
21	TSNS_SLOPE	[12:23]	温度传感器斜率	通用
64	CHIP_ID	[0:31]	芯片型号代码	通用
67	USB_VID	[0:15]	USB 厂商 ID	通用
67	USB_PID	[16:31]	USB 设备 ID	通用
80	PUBLIC_KEY_HASH	[0:255]	公钥哈希	识别
88	UUID	[0:127]	唯一识别号	识别
96	EXIP0_KEY	[0:255]	EXIP 密钥 0	密钥
112	MASTER_KEY	[0:255]	对称算法根密钥	密钥

表 30: OTP 列表

本产品上，用户可以自行烧写 OTP Word 69~79，用于存放应用数据。

18 总线

本章节介绍本产品的总线结构。

18.1 总线结构概述

本产品的总线系统由多个系统总线矩阵构成。片上的所有模块都通过这些系统总线矩阵连接起来，使得总线主设备，如处理器，DMA 等可以访问总线从设备，如内存，外设等。由于各个主从设备连接到不同总线矩阵的不同位置，不同主设备对不同从设备的访问效率也会有所不同。

本产品包括以下系统总线矩阵：

- AHB 系统总线，这是位于中心位置的总线，系统上大部分主设备要通过它来访问片上资源;
- APB 外设总线，通过它可以访问片上外设的寄存器;

18.2 总线配置

本章节介绍各个总线的连接和配置情况。

18.2.1 AHB 系统总线

AHB 系统总线是位于中心位置的总线矩阵。

连接到它的主设备有:

- RISC-V CPU
- HDMA
- Debug 调试模块（DM）
- USB
- XPISLV
- 加解密模块
- ADCxDACx

通过它可以访问的从设备有：

- 片上只读存储器 ROM
- XPI 控制器，和与它配套的在线解密模块
- 通用外设寄存器接口
- CAN 总线寄存器接口和电机控制外设寄存器接口
- HRAM
- 安全模块寄存器接口，模拟外设寄存器接口，USB, XPI，EXIP 寄存器等
- 电源管理域外设寄存器接口，系统控制模块寄存器接口, 锁相环寄存器等
- ILM 和 ILM 的镜像, DLM 和 DLM 的镜像等

18.2.2 APB 外设总线

APB 外设总线连接片上大部分外设。RISC-V 处理器和 DMA 通过它来访问外设寄存器。

19 BootROM

本章详细介绍了 BootROM 相关的特性及应用。

19.1 BootROM 概述

BootROM 为该芯片上电后执行的第一段程序, 它支持如下功能:

- 从Serial NOR FLASH 启动
  - 支持代码在 FLASH 上原地执行代码或者复制到内部 RAM 执行
  - 支持基于 AES-CTR 模式的原地解密执行 (EXIP)
- 串行启动
  - 从 UART0 或 USB0 启动, 支持流式下载代码到系统 RAM 中执行
- 在系统编程 (ISP)
  - 支持 UART0 和 USB0(USB-HID 协议)
  - 支持对连接在 XPI0 的 NOR FLASH 进行编程
  - 支持对片上 OTP 进行编程
- 安全启动
  - 支持基于 ECDSA-P256 + SHA256 的安全启动
  - 运行基于 AES-CBC + SHA256 的安全加密启动
  - 支持基于 SM2+SM3 的安全启动
  - 支持基于 SM4-CBC + SM3 的安全加密启动
- 低功耗唤醒
  - 支持 CPU 低功耗掉电模式下通过 BootROM 快速唤醒
- Debug 口权限管理
- 丰富的 ROM API

19.2 启动流程

本章介绍 BootROM 启动相关的流程。

BootROM 支持如下启动模式：

- 主启动模 (XPI NOR 启动模式)。
- 串行启动模式 (UART, USB-HID)
- 在系统编程模式 (In-System-Programming)

19.2.1 启动流程图

本节展示 BootROM 的启动流程图。

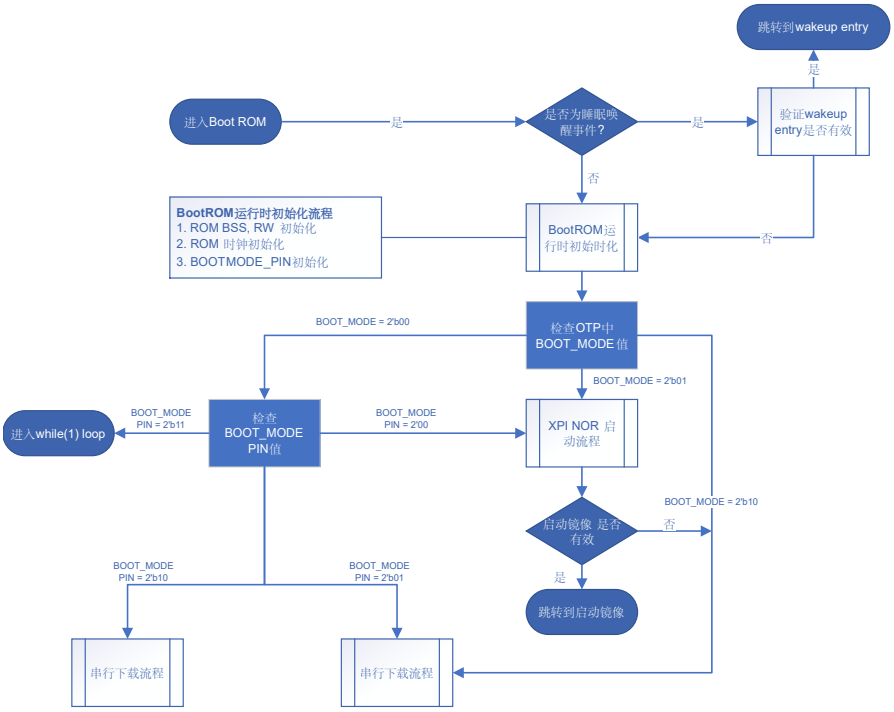


图 11: BootROM 启动流程图

19.2.2 XPI NOR 启动

19.2.2.1 支持的 Serial NOR 设备

Boot ROM 支持从主流的 Serial NOR FLASH 启动，包括但不限于:

- QSPI NOR FLASH (旺宏、华邦、兆易创新、ISSI 等）

19.2.2.2 FLASH 的接入方式

BootROM 通过以下典型的连接方式与 4 线 FLASH 连接。

QSPI FLASH 通过 CA 连接

SoC 通过 XPI 的 CA 端口与 4 线 FLASH 连接。市面上大部分的 4 线 FLASH 都不提供 DQS 引脚，这种情况下，XPIn.CA_DQS 应悬空，该引脚不能用于其他目的 (若XPI 信息小节有特别声明则按该声明执行)。

图 12: XPI QSPI NOR FLASH CA 端口连接

FLASH 连接方式的指定

BootROM 根据以下 OTP 字段来决定相应的 FLASH 连接：

- 根据 XPI_PIN_GROUP 选择默认的 XPI 引脚分组

详细的 XPI 的引脚分配见XPI 信息小节。

FLASH 配置信息的指定

BootROM 根据以下 OTP 字段来决定相应的 FLASH 配置信息：

- 根据 XPI_NOR_CFG_SRC 决定默认的 FLASH 配置信息的来源：Serial NOR FLASH 或 OTP
- 根据 XPI_DEFAULT_READ 决定上电后默认访问 FLASH 的命令
- 根据 ENCRYPT_XIP 决定是否启动加密原地执行功能

当 FLASH 配置信息来自 OTP 时:

- 根据 XPI_FREQ_OPTION 决定 FLASH 读操作的时钟频率
- 根据 PROBE_TYPE 决定 FLASH 的类型
- 根据 XPI_DRIVE_STRENGTH 决定 XPI IO 的驱动强度
- 根据 DUMMY_CYCLE 决定 read 时 dummy cycle 的值

当 FLASH 配置信息来自 FLASH 时，根据 XPI_DEFAULT_READ 决定使用的 FLASH 读命令，并从 FLASH 的 0x400 地址读取 FLASH 配置信息。

注:FLASH 配置信息的详细描述见XPI NOR 配置选项

19.2.2.3 XPI NOR 启动流程

当用户选择从 XPI NOR 启动后，设备上电后会按图 (13) 所示流程来执行 XPI NOR 启动。

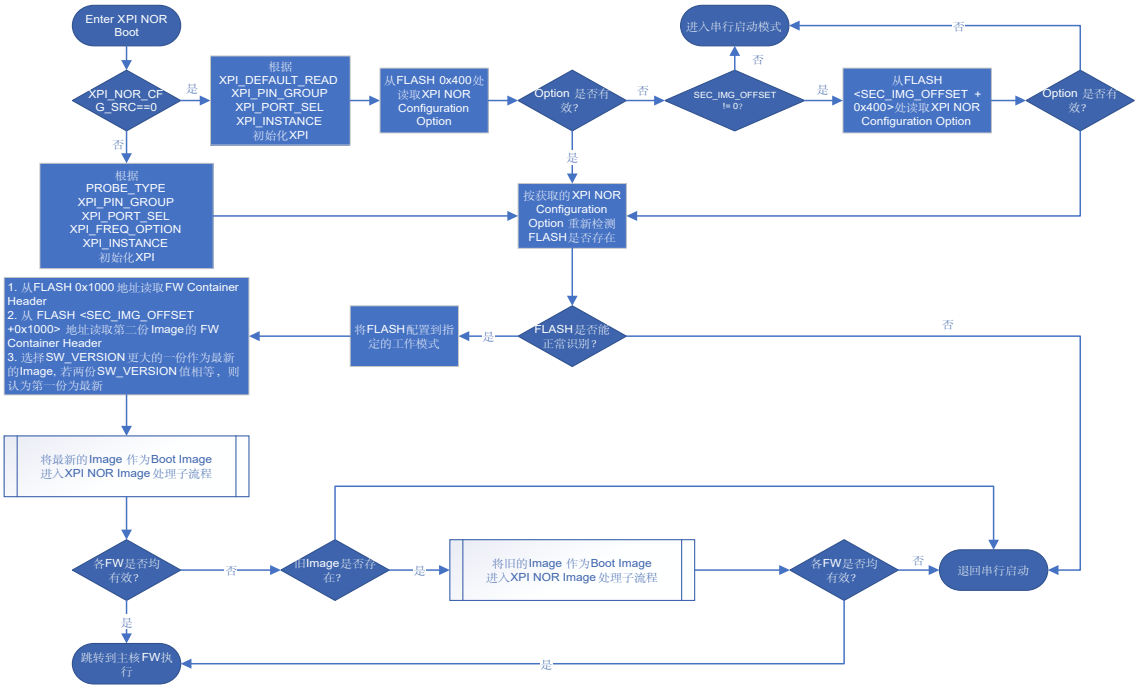


图 13: XPI NOR 启动流程

19.2.2.4 XPI NOR 启动镜像布局

XPI NOR 的启动镜像在 FLASH 中的布局如图 (14) 所示：

图 14: XPI NOR 启动镜像布局

19.2.2.5 原地解密执行

BootROM 支持通过 XPI NOR 的原地解密执行。当 OTP 中的 ENCRYPT_XIP 字段被置 1 后。BootROM 在 XPI NOR 启动中会强制打开 EXIP，并尝试用 EXIP0_KEK(XPI_INSTANCE 值为 0）/EXPI1_KEK(XPI_INSTANCE 值为 1）解密 EXIP Blob, 当 EXIP Blob 解密无误后，ROM 会根据 EXIP BLOB 中的信息配置对应的解密信息 (解密区域、初始向量等)。当 EXIP 配置完成后。ROM 即可执行原地解密执行操作。

注意: 启动镜像的头部不能被加密，否则 BootROM 无法启动该镜像

原地解密执行分两个步骤:

1. 用 OTP 中的 $EXIP\_KEK_n$ 来解密加密的 EXIP BLOB。EXIP BLOB 采用RFC3394标准来加密。
2. 用 EXIP BLOB 中的信息来配置 EXIP 模块, 准备好原地执行解密环境。

19.2.3 在系统编程 (In-System-Programming))

ROM 在 ISP 模式下，支持如下接口：

- 串口（UART）
- USB（USB-HID）

ROM 在 ISP 模式下，支持如下功能：

- 查询运行时信息 (query-rte)
- 配置运行时信息 (config-rte)
- 配置存储 (configure-memory)
- 写存储设备 (write-memory)
- 读存储设备 (read-memory)
- 加载镜像 (load-image)
- 擦除存储设备 (erase)
- 重启设备 (reset)
- 产生固件 Blob(gen-fwblob)

19.2.3.1 命令协议

BootROM 支持如下命令：

- 查询运行时环境 (query-rte)
- 配置运行时环境 (config-rte)
- 配置存储设备 (configure-memory)
- 写存储设备 (write-memory)
- 读存储设备 (read-memory)
- 加载镜像 (load-image)
- 擦除存储设备 (erase)
- 复位 SoC (reset)
- 产生固件 Blob(gen-fwblob)

通信过程

在 ROM 的通信协议下，每个命令均包含 4 个阶段。如下图所示。

1. 第一阶段：主机端命令和数据发送阶段。在该阶段，若命令只需要传输简单的参数，则该命令数据发送只包含一个数据包（如 reset/erase)；若命令包含数据传输，则数据发送包含一个混合包（命令 + 数据）以及若干包纯数据包。
2. 第二阶段：从机端发送 ACK/NAK/Abort 阶段。对于第一阶段的每个数据包，从机端都需要给于对应的确认。当数据处理没有异常时，回复 ACK；当数据检验错误时，回复 NAK；当数据处理出错时，回复 Abort。
3. 第三阶段：从机端发送响应阶段。当第一阶段的数据传输完成或者 Device 回复 Abort 后。通信转入第三阶段。在该阶段，如果命令只需要从机回复简单的状态，则数据包只有一响应包（如 reset/erase)；若从机需要回复数据，则从机会先回复命令响应帧，然后回复剩余需要发送的数据。在每一帧发送结束后，从机都需要等主机的确认帧，当收到主机发送的 Abort 后，通信结束。
4. 第四阶段：主机发送对从端响应的确认。当主机收到从端的响应包时，需要回复以 ACK/NAK/Abort。当

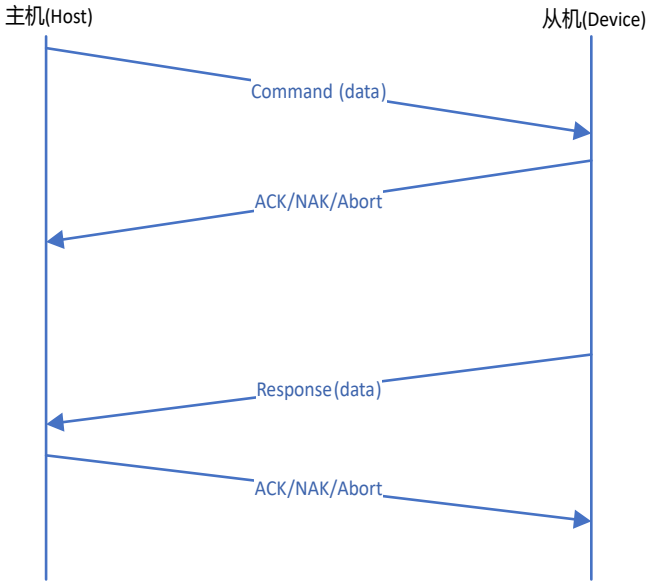


图 15: 串行通信协议

数据包无异常时，回复 ACK；当数据包检验出错时，回复 NAK，当从端发送了超出预期的数据时，回复 Abort。

命令数据结构

每个命令均统一包含由 4 字节组成的命令头, 详细定义如下表所示：

偏移 (字节)	字段	描述
0	命令标识符	1 - 查询运行时环境 (query-rte) 2 - 配置运行时环境 (config-rte) 3 - 配置存储 (configure-memory) 4 - 写存储 (write-memory) 5 - 读存储 (read-memory) 6 - 加载镜像 (load-image) 7 - 擦除 (erase) 8 - 复位 (reset) 9 - 生成固件 Blob(gen-fwblob)
1	命令参数个数	-
2	命令类型	0 - 命令 + 数据 1 - 仅数据 2 - 仅响应
3	保留	-

表 31: 命令数据结构

查询运行时环境 (query-rte)

该命令可用于查询如下状态：

- ROM 参数
- 活动的通信接口参数
- 上次启动的状态
- Memory 的属性

详细的命令数据结构如下表所示:

当 ROM 收到该命令后，若参数合法，ROM 则会回复相应的响应包。响应包的参数部分根据不同的命令而不同。

ROM 参数响应包

<table>
<tr><th>偏移 (字节)</th><th>宽度 (字节)</th><th>字段</th><th>描述</th></tr>
<tr><td>0</td><td>1</td><td>命令标识</td><td>1 - query-rte</td></tr>
<tr><td>1</td><td>1</td><td>命令参数个数</td><td>5</td></tr>
<tr><td>2</td><td>1</td><td>命令类型</td><td>2 - 仅响应数据</td></tr>
<tr><td>3</td><td>1</td><td>保留</td><td>-</td></tr>
<tr><td>4</td><td>4</td><td>命令状态</td><td>0 - 成功</td></tr>
<tr><td>8</td><td>4</td><td>ROM 版本</td><td><table><tr><td>Bit[31:24]</td><td>ASCII: V(0x56)</td></tr><tr><td>Bit[23:16]</td><td>主版本号：0x01</td></tr><tr><td>Bit[15:8]</td><td>次版本号:0x00</td></tr><tr><td>Bit[7:0]</td><td>问题修复:0x00</td></tr></table></td></tr>
<tr><td>12</td><td>4</td><td>SoC 版本</td><td>与 ROM 版本号定义一致</td></tr>
<tr><td>16</td><td>4</td><td>杂项</td><td>Bit[7:0] - 生命周期
Bit[31:8] - 保留</td></tr>
<tr><td>20</td><td>4</td><td>ROM 功能信息</td><td><table><tr><td>Bit[31:27]</td><td>保留</td></tr><tr><td>Bit[26:26]</td><td>是否支持 SM4_CCM</td></tr><tr><td>Bit[25:25]</td><td>是否支持 AES_CCM(256 位密钥)</td></tr><tr><td>Bit[24:24]</td><td>是否支持 AES_CCM(128 位密钥)</td></tr><tr><td>Bit[23:20]</td><td>保留</td></tr><tr><td>Bit[19:19]</td><td>是否支持 SM2</td></tr><tr><td>Bit[18:18]</td><td>是否支持 ECDSA P521</td></tr><tr><td>Bit[17:17]</td><td>是否支持 ECDSA P384</td></tr><tr><td>Bit[16:16]</td><td>是否支持 ECDSA P256</td></tr><tr><td>Bit[15:12]</td><td>保留</td></tr><tr><td>Bit[11:11]</td><td>是否支持设备配置块</td></tr><tr><td>Bit[10:10]</td><td>是否支持命令容器</td></tr><tr><td>Bit[9:9]</td><td>是否支持加密启动</td></tr><tr><td>Bit[8:8]</td><td>是否支持安全启动</td></tr><tr><td>Bit[7:4]</td><td>保留</td></tr><tr><td>Bit[3:3]</td><td>是否支持在系统编程 (ISP)</td></tr><tr><td>Bit[2:2]</td><td>是否支持串行启动</td></tr><tr><td>Bit[1:1]</td><td>是否支持恢复启动</td></tr><tr><td>Bit[0:0]</td><td>是否支持主启动</td></tr></table></td></tr>
</table>

表 33: ROM 参数响应包

活动外设信息包

偏移 (字节)	宽度 (字节)	字段	描述
0	1	命令标识	1 - query-rte
1	1	命令参数个数	如数参数不合法，该值为 0，否则为”word” 字节 +1
2	1	命令类型	2 - 仅响应数据
3	1	保留	-
4	4	命令状态	0 - 成功 2 - 无效参数
8	4	通用信息	Bit[7:0] word 数（以 32-bit 为单位） Bit[31:8] - 保留
12	4	外设掩码	0x01 - UART 0x10 - USB-HID
16	4	速率	对于 UART，该字段表示波特率；对于 USB-HID，该字段无意义
20	8	保留位	-
28	4	最大包长	最大数据包长度 (不包括包头和 CRC 校验位)

表 34: 活动外设信息包

上次 ROM 启动状态包

偏移 (字节)	宽度 (字节)	字段	描述
0	1	命令标识	1 - query-rte
1	1	命令参数个数	如数参数不合法，该值为 0，否则为”word” 字节 +1
2	1	命令类型	2 - 仅响应数据
3	1	保留	-
4	4	命令状态	0 - 成功
8	4	启动参数	启动参数 Bit[31:16] 保留位 Bit[15:8] 启动状态的字数 (以 32-bit 为单位) Bit[7:0] 启动模式

表 35: 上次 ROM 启动状态

存储设备属性包

偏移 (字节)	宽度 (字节)	字段	描述
0	1	命令标识	1 - query-rte
1	1	命令参数个数	如数参数不合法，该值为 0，否则为”word” 字节 +1
2	1	命令类型	2 - 仅响应数据

偏移 (字节)	宽度 (字节)	字段	描述
3	1	保留	-
4	4	命令状态	0 - 成功
8	4	通用信息	Bit[7:0] 存储设备属性字数 (单位为 32-bit) Bit[15:8] 存储大小的单位：0 - 1 字节, 1 - 2 字节 2 - 4 字节 其它 - 保留位
12	8	存储设备标识	TBD
16	4	起始地址	存储设备起始地址
20	4	结束地址	存储设备结束地址
24	4	存储设备容量	存储设备的容量
28	4	Page 大小（可选）	仅对 FLASH 设备有效
32	4	Sector 大小 (可选)	仅对 FLASH 设备有效
32	4	Block 大小 (可选)	仅对 FLASH 设备有效

表 36: 存储设备属性响应包

配置运行时环境 (configure-rte)

该命令可以配置活动外设的参数，当前仅 UART 的波特率支持被配置。详细的命令数据结构如下：

偏移 (字节)	宽度 (字节)	字段	描述
0	1	命令标识	2 - configure-rte
1	1	命令参数个数	4
2	1	命令类型	0 - 命令 + 数据
3	1	保留	-
4	4	运行时环境标识	1 - 活动外设信息
8	4	通用信息	Bit[7:0] 参数字数 (单位为 32-bit) 其它 - 保留位
12	4	外设掩码	1 - UART
16	4	速率	UART 波特率

表 37: 配置运行时环境 (configure-rte)

该命令的响应帧数据结构如下：

偏移 (字节)	宽度 (字节)	字段	描述
0	1	命令标识	2 - configure-rte

偏移 (字节)	宽度 (字节)	字段	描述
1	1	命令参数个数	1
2	1	命令类型	2 - 仅响应数据
3	1	保留	-
4	4	命令状态	0 - 成功 2 - 无效参数

表 38: 配置运行时环境响应包

配置存储 (configure-memory)

该命令用于通过指定地址存储的存储设备配置信息来使能对外挂存储设备的支持，如使能对 Serial NOR FLASH 的支持。该命令的详细数据结构如下：

偏移 (字节)	宽度 (字节)	字段	描述
0	1	命令标识	3 - configure-memory
1	1	命令参数个数	2
2	1	命令类型	0 - 命令 + 数据
3	1	保留	-
4	4	存储设备标识	0x10000 - 挂载在 XPI0 上的 NOR FLASH 0x10001 - 挂载在 XPI1 上的 NOR FLASH
8	4	存储配置块地址	存储配置块的地址 (仅支持从片上 RAM 中获取)

表 39: 配置存储 (configure-memory)

该命令的响应数据帧的数据结构如下：

偏移 (字节)	宽度 (字节)	字段	描述
0	1	命令标识	3 - configure-memory
1	1	命令参数个数	1
2	1	命令类型	2 - 仅响应数据
3	1	保留	-
4	4	命令状态	0 - 成功 2 - 无效参数

表 40: configure-memory 响应帧数据结构

写存储设备 (write-memory)

该命令用于向 on-chip SRAM、片上的 OTP 以及外挂的 XPI NOR 写入数据。该命令可由单一数据包或者多包数据组成。其中第一包的数据结构定义如下:

偏移 (字节)	宽度 (字节)	字段	描述
0	1	命令标识	4 - write-memory
1	1	命令参数个数	3
2	1	命令类型	0 - 命令 + 数据
3	1	保留	-
4	4	要写入的地址	要写入的设备地址
8	4	长度	要写入的数据长度
12	4	内存设备标识	0 - 片上 RAM 0x10000 - 挂载在 XPI0 上的 NOR FLASH 0x10001 - 挂载在 XPI1 上的 NOR FLASH 0x20000 - OTP
16	可变长度	数据	要写入到存储设备的数据

表 41: 配置存储 (write-memory) - 命令 + 数据

第二包到最后一包的数据结构如下：

偏移 (字节)	宽度 (字节)	字段	描述
0	1	命令标识	4 - write-memory
1	1	命令参数个数	0
2	1	命令类型	1- 仅数据
3	1	保留	-
4	可变长度	数据	要写入到存储设备的数据

表 42: 配置存储 (write-memory) - 仅数据

当数据可以被一包发送完时，只需要第一类数据包，当数据需要多包发送时，需要第二类数据包。数据包中的 Data 部分的长度通过各类通信接口中收到的数据包长度获取。

如果在 write-memory 执行过程中出错，从机将中止传输，并在响应包中通知主机错误代码。

当 write-memory 成功执行完成，并在响应包中向主机报告成功代码。

Write-memory 的响应包数据结构定义如下：

偏移 (字节)	宽度 (字节)	字段	描述
0	1	命令标识	4 - configure-memory
1	1	命令参数个数	1
2	1	命令类型	2 - 仅响应数据
3	1	保留	-
4	4	命令状态	0 - 成功 其它 - 错误代码

偏移 (字节)	宽度 (字节)	字段	描述

表 43: write-memory 响应帧数据结构

读存储 (read-memory)

该命令用于从 on-chip SRAM、片上的 OTP 以及外挂的 XPI NOR 读数据。该命令由一条命令包与多条响应包组成。

命令包的数据结构如下：

偏移 (字节)	宽度 (字节)	字段	描述
0	1	命令标识	5 - read-memory
1	1	命令参数个数	3
2	1	命令类型	0 - 命令 + 数据
3	1	保留	-
4	4	要读取的地址	要读取的设备地址
8	4	长度	要读取的数据长度
12	4	内存设备标识	0 - 片上 RAM 0x10000 - 挂载在 XPI0 上的 NOR FLASH 0x10001 - 挂载在 XPI1 上的 NOR FLASH 0x20000 - OTP

表 44: read-memory 命令数据桢

当从机收到该命令后，会先进行参数检查，若参数检查出错，会直接在命令的第二阶段回复主机 Abort，然后在响应包中告知错误代码，在等到主机发出的第四阶段的确认后，该次命令结束。若参数检查通过，从机在第二阶段回复主机 ACK, 然后从机会先在第一个响应包中通知主机参数检查成功，并在后续将请求的数据依次通过数据包发给主机。

从机回复的第一类响应帧如下：

偏移 (字节)	宽度 (字节)	字段	描述
0	1	命令标识	5 - read-memory
1	1	命令参数个数	1
2	1	命令类型	2 - 仅响应数据
3	1	保留	-
4	4	命令状态	0 - 成功 其它 - 错误代码

表 45: read-memory 响应帧数据结构

从机的纯数据包帧如下：

偏移 (字节)	宽度 (字节)	字段	描述
0	1	命令标识	5 - read-memory
1	1	命令参数个数	0
2	1	命令类型	1 - 仅数据
3	1	保留	-
4	可变长度	数据	从存储设备中读取的数据

表 46: read-memory 响应帧数据结构

加载镜像 (load-image)

该命令用于加载并执行可执行镜像，或加载并解析命令容器。该命令由纯数据包组成，一个命令由若干个数据包组成。

其数据包结构如下：

偏移 (字节)	宽度 (字节)	字段	描述
0	1	命令标识	6 - load-image
1	1	命令参数个数	0
2	1	命令类型	1- 仅数据
3	1	保留	-
4	可变长度	数据	镜像的数据

表 47: load-image 数据帧结构

当数据发送完成后，从机会回复一包响应帧，其数据结构如下：

偏移 (字节)	宽度 (字节)	字段	描述
0	1	命令标识	6 - load-image
1	1	命令参数个数	1
2	1	命令类型	2 - 仅响应数据
3	1	保留	-
4	4	命令状态	0 - 成功 其它 - 错误代码

表 48: load-image 响应帧数据结构

擦除存储设备 (erase)

该命令可用于擦除外挂的 XPI NOR 的部分或者全部的 sector, 其数据结构如下:

偏移 (字节)	宽度 (字节)	字段	描述
0	1	命令标识	7 - erase

偏移 (字节)	宽度 (字节)	字段	描述
1	1	命令参数个数	2 - chip erase 4-sector erase
2	1	命令类型	1- 仅数据
3	1	保留	-
4	4	擦除类型	0 - sector erase 1 - chip erase
8	4	存储设备标识	0x10000 - 挂载在 XPI0 上的 NOR FLASH 0x10001 - 挂载在 XPI1 上的 NOR FLASH
12	4	擦除地址	仅当擦除类型为 sector erase 时有效
16	4	擦除长度	仅当擦除类型为 sector erase 时有效

表 49: erase 数据帧结构

命令的响应帧如下：

偏移 (字节)	宽度 (字节)	字段	描述
0	1	命令标识	7 - erase
1	1	命令参数个数	1
2	1	命令类型	2 - 仅响应数据
3	1	保留	-
4	4	命令状态	0 - 成功 其它 - 错误代码

表 50: erase 响应帧数据结构

复位 (reset)

该命令可复位 SoC。其数据结构如下：

偏移 (字节)	宽度 (字节)	字段	描述
0	1	命令标识	8 - reset
1	1	命令参数个数	1
2	1	命令类型	1- 仅数据
3	1	保留	-
4	4	复位类型	在该 SoC 中本字段为保留位

表 51: reset 数据帧结构

该命令的响应帧如下：

偏移 (字节)	宽度 (字节)	字段	描述
0	1	命令标识	8 - reset

偏移 (字节)	宽度 (字节)	字段	描述
1	1	命令参数个数	1
2	1	命令类型	2 - 仅响应数据
4	4	命令状态	0 - 成功 其它 - 错误代码

表 52: reset 响应帧数据结构

生成固件 Blob(gen-fwblob)

该命令能生成加密镜像所需的固件 BLOB。

偏移 (字节)	宽度 (字节)	字段	描述
0	1	命令标识	9 - gen-fwblob
1	1	命令参数个数	1 + 密钥的字节数/4 (从密钥参数后开始计)
2	1	命令类型	1- 仅数据
3	1	保留	-
4	4	密钥参数	密钥参数 **Byte 0 - alg (算法)** 0x33 - AES, 0x55 - SM4 **Byte 1 - mode (模式)** 0x11 - CBC-MAC **Byte 2 - key_src (对称密钥来源)** 1 - FMK , 2 - ZMK, 3 - OTP KEY0, 4 - OTP KEY1 **Byte 3 - key_size(密钥长度)** SM4/AES-128 : 16 AES-256 : 32
8	4	Reserved	-
12	16/32	对称密钥	用于加密固件的密钥

表 53: 生成固件 Blob(gen-fwblob) 数据帧结构

当从机收到该命令后，会先进行参数检查，若参数检查出错，会直接在命令的第二阶段回复主机 Abort，然后在响应包中告知错误代码，在等到主机发出的第四阶段的确认后，该次命令结束。若参数检查通过，从机在第二阶段回复主机 ACK, 然后从机会先在第一个响应包中通知主机参数检查成功，并在后续将请求的数据依次通过数据包发给主机。

从机回复的第一类响应帧如下：

偏移 (字节)	宽度 (字节)	字段	描述
0	1	命令标识	9 - gen-fwblob
1	1	命令参数个数	1

偏移 (字节)	宽度 (字节)	字段	描述
2	1	命令类型	2 - 仅响应数据
3	1	保留	-
4	4	命令状态	0 - 成功 其它 - 错误代码

表 54: gen-fwblob 响应帧数据结构

从机的纯数据包帧如下：

偏移 (字节)	宽度 (字节)	字段	描述
0	1	命令标识	9 - gen-fwblob
1	1	命令参数个数	0
2	1	命令类型	1 - 仅数据
3	1	保留	-
4	可变长度	数据	返回的完整的固件 Blob 数据

表 55: gen-fwblob 回复的数据帧

19.2.4 串口通信

ROM 默认支持的串口通信参数为 115200，8-N-1，可通过 configure-rte 命令切换波特率。

串口通信的协议数据包分别为两种：

- 载荷包
- 确认包

19.2.4.1 载荷包 (Payload Packet)

载荷帧由包头、包类型、载荷以及 CRC 校验 4 部分组成，详细数据包结构如下：

偏移 (字节)	宽度 (字节)	字段	描述
0	1	起始字节	必须为 0x5A
1	1	数据包类型	0xA5 - 载荷包
2	2	载荷长度	载荷的长度。不能超过 512 字节
4	<Payload_Len>	数据载荷	数据载荷，用于承载命令协议中的数据
4 + <Payload_Len>	2	CRC 检验值	整个数据包前 <4+Payload_Len> 个字节的 CRC 检验值

表 56: 载荷包数据结构

CRC 的计算范围为起始字节到数据载荷的最后一个字节。

CRC 的算法是 XMODEM 的一个变种。其 C 语言实现如下：

```
uint16_t crc16_update(const uint8_t *src, uint32_t lengthInBytes)
{
```

```
    uint32_t crc = 0;
    uint32_t j;
    for (j = 0; j < lengthInBytes; ++j)
    {
        uint32_t i;
        uint32_t byte = src[j];
        crc ^= byte << 8;
        for (i = 0; i < 8; ++i)
        {
            uint32_t temp = crc << 1;
            if (crc & 0x8000)
            {
                temp ^= 0x1021;
            }
            crc = temp;
        }
    }
    return (uint16_t)crc;
}
```

19.2.4.2 确认包 (ACK Packet)

确认包用于实现载荷包的 ACK, NAK 或者中止 (Abort)。详细的数据包结构如下:

偏移 (字节)	宽度 (字节)	字段	描述
0	1	起始字节	必须为 0x5A
1	1	数据包类型	0xA1 - ACK 0xA2 - NAK 0xA3 - Abort

表 57: 确认包

每个载荷包的发出方都需要等待接收方的确认包后才能进行后续的数据发送。

数据包的载荷部分可携带上节介绍的各个命令或者数据。

19.2.5 USB-HID 通信

BootROM 支持通过 USB-HID 来进行通信，实现诸如 FLASH 烧写、EFUSE 烧写、加载启动镜像等功能。

19.2.5.1 USB-HID 描述符信息

BootROM 中 USB 的描述符配置如下表所示:

描述符标识	内容/值
VID	0x34b7

描述符标识	内容/值
PID	0x0005
厂商描述符	HPMICRO Semiconductor Co., Ltd
产品描述符	Boot ROM Open(出厂的设置) Boot ROM Closed (量产的设置) Boot ROM Return(返厂的设置)
HID 包长	接 High-Speed HUB 时：516 字节 接 Full-Speed HUB 时:64 字节
HID 中断传输间隔	接 High-Speed HUB 时: 125us 接 Full-Speed HUB 时：1ms

表 58: USB 描述符

19.2.5.2 USB 端点信息

BootROM 中使用了如下三个端点:

- Control(端点 0) - 用于获取设备描述符
- OUT (端点 1) - 用于 Host 向 Device 发送数据
- IN (端点 2) - 用于 Device 向 Host 发送数据

19.2.5.3 USB-HID 通信协议

USB-HID 通信协议和 UART 类似，均包含载荷包和确认包。

USB-HID 载荷包

USB-HID 的载荷包包含四个字段：包头、包类型、载荷长度以及数据载荷。详细的的数据结构如下所示：

偏移 (字节)	宽度 (字节)	字段	描述
0	1	起始字节	Host 发往 Device 为 0x1, Device 发往 Host 为 0x2
1	1	数据包类型	0xA5 - 载荷包
2	2	载荷长度	载荷的长度。不能超过 512 字节
4	<Payload_Len>	数据载荷	数据载荷，用于承载命令协议中的数据

表 59: USB-HID 载荷包数据结构

USB-HID 确认包

USB-HID 的载荷包三个字段：包头、包类型、载荷长度。详细的确认包的数据结构如下所示：

偏移 (字节)	宽度 (字节)	字段	描述
0	1	起始字节	Host 发往 Device 为 0x1, Device 发往 Host 为 0x2
1	1	数据包类型	0xA1 - ACK 0xA2 - NAK 0xA3 - Abort
2	2	载荷长度	固定为 0

偏移 (字节)	宽度 (字节)	字段	描述

表 60: USB-HID 确认包数据结构

19.3 启动镜像 (Boot Image)

该设备支持的启动镜像由固件容器 (FW Container) 及固件 (Firmware) 两部分构成。其中 FW Container 由 FW Container Header、FW Info Table、Device Configuration Block 和 Signature Block 构成。启动镜像的布局如图所示。

图 16: 启动镜像布局

19.3.1 固件容器头 (FW Container Header)

固件容器头的详细数据结构如下所示：

表 61: 固件容器头

偏移 (字节)	字段	宽度 (字节)	描述
0x00	tag	1	启动镜像标记, 必须为 0xBF
0x01	version	1	启动镜像格式版本 bit[7:4] - 主版本 bit[3:0] - 次版本 本 SoC 上该值必须为 0x10

表 61: 固件容器头

<table>
<tr><th>偏移
(字节)</th><th>字段</th><th>宽度
(字节)</th><th>描述</th></tr>
<tr><td>0x02</td><td>length</td><td>2</td><td>固件头的长度
包含固件容器头、固件信息表、设备配置块和签名块</td></tr>
<tr><td>0x04</td><td>flags</td><td>4</td><td>相应的标记
<table>
<tr><td>Bit[3:0]</td><td>SRK 的集合索引</td></tr>
<tr><td>Bit[7:4]</td><td>SRK 的索引</td></tr>
<tr><td>Bit[11:8]</td><td>SRK_REVOKE_MASK
每个 bit 代表撤销相应位置的 SRK，当该字节为非 0 时，BootROM 会尝试按该字段的值撤销相应的 SRK，并将 OTP 中的 PUK_REVOKE 烧写成相同的值</td></tr>
<tr><td>Bit[19:16]</td><td>签名类型
1 - ECDSA P256
4 - SM3</td></tr>
<tr><td>Bit[23:20]</td><td>固件 BLOB 位置
0 - BLOB 在签名块中
1 - BLOB 在存储介质固件的位置</td></tr>
<tr><td>Bit[27:24]</td><td>生命周期
0 - 生命周期不作改动
其它 - BootROM 会将生命周期切换成该字段的值</td></tr>
<tr><td>Bit[31:28]</td><td>保留</td></tr>
</table></td></tr>
<tr><td>0x08</td><td>sw_version</td><td>2</td><td>软件版本号，定义了当前 Image 的版本，当存储介质中有多份 Image 时，ROM 根据该字段来判断哪一份是较新的 Image</td></tr>
<tr><td>0x0A</td><td>fuse_version</td><td>1</td><td>这个字段用于 Image 的防回滚, 该字段的值不得小于 OTP 中 SW_VER 字段的值
注:SW_VER 中 bit1 的个数代表版本号 (如,0xffff 代表版本号为 15)</td></tr>
<tr><td>0x0B</td><td>number_of_fw</td><td>1</td><td>指定当前 Image 里包含的固件数，当前 SoC 最多支持两份</td></tr>
<tr><td>0x0C</td><td>device_config_block_offset</td><td>2</td><td>设备配置块相对于固件容器头的偏移，设备配置块是可选的，若当前 Image 中没有该块，此字段须为 0</td></tr>
<tr><td>0x0E</td><td>signature_block_offset</td><td>2</td><td>签名块相对于固件容器头的偏移，若签名块不存在，此字段须为 0</td></tr>
</table>

表 61: 固件容器头

19.3.2 固件信息表 (FW Info Table)

固件信息表主要提供固件执行地址、长度、在存储介质中相对固件容器头的偏移、固件哈希值等。其详细数据结构如下所示：

<table>
<tr><th>偏移
(字节)</th><th>字段</th><th>宽度
(字节)</th><th>描述</th></tr>
<tr><td>0x00</td><td>offset</td><td>4</td><td>固件相对于固件容器头的偏移</td></tr>
<tr><td>0x04</td><td>size</td><td>4</td><td>固件的大小</td></tr>
<tr><td>0x08</td><td>flags</td><td>4</td><td>相应的标记<table><tr><td>Bit[3:0]</td><td>fw_type - 固件类型
0 - 可执行镜像
1 - 命令容器</td></tr><tr><td>Bit[7:4]</td><td>core_id - CPU ID 号
0 - CPU0
1 - CPU1</td></tr><tr><td>Bit[11:8]</td><td>hash_type - 哈希类型
0 - 无
1 - SHA256
4 - SM3</td></tr><tr><td>Bit[15:12]</td><td>is_encrypted - 是否加密
0 - 未加密 1 - 固件已加密</td></tr><tr><td>其它</td><td>保留</td></tr></table></td></tr>
<tr><td>0x0C</td><td>reserved</td><td>4</td><td>保留给未来使用</td></tr>
<tr><td>0x10</td><td>load_addr</td><td>4</td><td>固件加载地址</td></tr>
<tr><td>0x14</td><td>reserved</td><td>4</td><td>保留给未来使用</td></tr>
<tr><td>0x18</td><td>entry_point</td><td>4</td><td>固件入口</td></tr>
<tr><td>0x1C</td><td>reserved</td><td>4</td><td>保留给未来使用</td></tr>
<tr><td>0x20</td><td>hash</td><td>64</td><td>固件哈希值，目前只使用 32 字节。
当固件类型为可执行文件时，该字段是整个 FW 的哈希值。
当固件类型为命令容器时，该字段是命令容器头包的哈希值。当固件为加密固件时，该字段的前 128-bit 作为初始向量</td></tr>
<tr><td>0x60</td><td>reserved</td><td>32</td><td>保留给未来使用</td></tr>
</table>

表 62: 固件容器头

19.3.3 设备配置信息块 (Device Configuration Info Block)

设备配置块可支持如下 4 类配置信息：

- XPI NOR 配置信息
- 唤醒入口检查信息
- OTP UNLOCK 信息

设备配置信息块由以下部分组成：

- 设备配置信息块头 (Device Configuration Block Header)
- 设备配置信息
  - 设备配置信息头
  - 设备配置信息内容

设备配置信息头的数据结构如下：

Bit[31:16]	Bit[15:8]	Bit[7:0]
配置信息块的长度	版本号，当前为 0x10	配置信息类型 0xC0 - 总设备配置信息

表 63: 配置信息块头 (Configure Info Block Header)

每个配置信息有各自的头，头的定义如下:

Bit[31:16]	Bit[15:8]	Bit[7:0]
配置信息块的长度	参数 对 XPI NOR, XPI RAM 表示 XPI 的实例号	配置信息类型 0xC1 - XPI NOR 配置信息 0xC2 - XPI RAM 配置信息 0xC3 - SDRAM 配置信息 0xC4 - 通用寄存器信息 0xC5 - 唤醒验证配置信息 0xC6 - OTP 解锁信息

表 64: 配置信息头 (Configure Info Header)

紧随配置信息头其后的是相应的配置信息，各配置信息的详细定义如后续小节所示。

19.3.3.1 XPI NOR 配置信息

XPI NOR 配置信息分两类：一类为精简的 XPI NOR 配置信息，另一类为完整的 XPI NOR 配置块。

XPI NOR 配置选项

偏移 (字节)	字段	描述
0x00	Header	Bit[31:12] - tag: 标记，必须为 0xfcf90 Bit[11:4] - 保留 Bit[3:0] - words: 配置选项字数 (不包括 Header 本身)

偏移 (字节)	字段	描述
0x04	Option0	**Bit[31:28] - 探测类型** 0 - SFDP SDR, 1 - SFDP DDR 2 - 1-4-4 Read 0xEB, 3 - 1-2-2 Read 0xBB 4 - HyperBus 1V8, 5 - HyperBus 3V0 6 - OctaBus DDR, 8 - Xccela DDR 10 - EcoXiP DDR **Bit[27:24] - 上电复位后发送命令的数据引脚数** 0 - SPI, 1 - DPI, 2 - QPI, 3 - OPI **Bit[23:20] - FLASH 配置完后发送命令的数据引脚数** 0 - SPI, 1 - DPI, 2 - QPI, 3 - OPI **Bit[19:16] - 1-4-4 模式使能的序列** 0 - 不需要或者自动 1 - QE bit is at bit 6 in Status Register1 2 - QE bit is at bit1 in Status Register2 3 - QE bit is at bit7 in Status Register2 4 - QE bit is at bit1 in Status Register2 and should be programmed by 0x31 **Bit[15:8] - dummy cycles** 0 - 自动探测/默认值 其它 - 用户指定的值，对于 DTR 读，该值需要是实际数据手册上的 *dummycycle*×2 **Bit[7:4] - 杂项** 0 - 无 1 - SPI 模式 2 - 内部回环模式 3 - 强制外部 DQS **Bit[3:0] - 频率选项** 0 - 不改变现有配置 Others - SoC 相应的值

偏移 (字节)	字段	描述
0x08	Option1	该字段在“words”字段值大于 1 时有效 Bit[31:20] - reserved **Bit[19:16] - IO 电压** 0 - IO is 3V, 1 - IO is 1.8V **Bit[15:12] -引脚分组选择** 0 - 第一组, 1 - 第二组 **Bit[11:8] - 连接方式选择** 0 - PORTA_CS0, 1 - PORB_CS0, 2 - PORTA_CS0 +PORTB_CS0, 3 - PORTA_CS0+PORTA_CS1, 4 - PORTB_CS0+PORTB_CS1 **Bit[7:0] - IO 驱动强度** 0 - 默认值 (最大驱动强度) 其它 - 详见 (TBD)
0x0C	Option2	该字段在“words”大于 2 时有效 Bit[31:16] - reserved **Bit[15:12] - 擦除 sector 命令选项** 0 - Erase 4KB, 1 - Erase 32KB, 2 - Erase 64KB, 3 - Erase 256KB **Bit[11:8] - Sector 大小选项** 0 - 4KB, 1 - 32KB, 2 - 64KB, 3 - 256KB **Bit[7:0] - FLASH 大小选项** 0 - 4MB, 1 - 8MB, 2 - 16MB

表 65: XPI NOR 配置选项 (XPI NOR Configuration Option)

XPI NOR 配置块

XPI NOR 配置块的数据结构如下表所示：

偏移 (字节)	字段	宽度 (字节)	描述
0x000	tag	4	配置块标记，固定为 0x524f4e58
0x004	reserved	4	-
0x008	rxclk_src	1	采样时钟的来源 0 - 内部回环 1 - DQS 内部回环 3-外部 DQS 输入
0x009	clk_freq	1	时钟频率选项，详见19.8.2.1
0x00A	drive_strength	1	驱动强度 0 - 默认值 (最大驱动强度) 其它 - 详见 (TBD)

偏移 (字节)	字段	宽度 (字节)	描述
0x00B	column_addr_width	1	列地址的宽度 仅对 HyperFLASH 有效，其他 FLASH 需设为 0
0x00C	rxclk_src_for_init	1	FLASH 初始化时 rxclk_src 的值
0x00D	config_in_progress	1	指示当前 FLASH 是否正在初始化过程中 1 - 正在初始化，其它 - 未在初始化
0x00E	reserved	2	-
0x010	port_info	16	- 连接的端口信息 该字段分 4 组，分别代表 CA_CS0, CA_CS1,CB_CS0, CB_CS1。 每组的 4 个字节的定义如下: Byte 0 - Enable: 使能位，代表该连接被使能 Byte 1 - Group: 引脚分组: 0 代表第一组，1 代表第二组 Byte 2, Byte 3 - 保留给未来使用
0x020	device_info	0x50	详细定义见67
0x070	instruction_set	0x90	标准的 XPI NOR 指令集，分 9 组，每组 16 个字节代表一个完整的指令序列, 分组信息如下: 0 - FLASH Read 序列 1 - FLASH Write 序列 2 - FLASH Read Status 序列 3 - FLASH Read Status XPI 序列 (QPI/OPI 模式) 4 - FLASH Write Enable 序列 5 - FLASH Write Enable XPI 序列 (QPI/OPI 模式) 6 - FLASH Erase Sector 序列 7 - FLASH Erase Block 序列 8 - FLASH Erase Chip 序列 **注：对于 HyperFLASH, 上述定义并不成立，BootROM 支持专用的 FLASH 操作**

表 66: XPI NOR 配置块

XPI NOR 设备信息表的数据结构如下所示:

偏移 (字节)	字段	宽度 (字节)	描述
0x00	size_in_kbytes	4	FLASH 大小 (以 KB 为单位)
0x04	page_size	2	Page 大小
0x06	sector_size_kbytes	2	Sector 大小 (以 KB 为单位)
0x08	block_size_kbytes	2	Block 大小 (以 KB 为单位)
0x0A	busy_offset	1	Busy 位的偏移 (在 FLASH 的 Status 寄存器中)
0x0B	busy_polarity	1	Busy 位的极性: 0 - 1 代表忙 1 - 0 代表忙

偏移 (字节)	字段	宽度 (字节)	描述
0x0C	data_pads	1	数据引脚的个数 0 - 1 线, 1 - 2 线，2 - 4 线, 3 - 8 线
0x0D	en_ddr_mode	1	使能 DDR/DTR 模式
0x0E	clk_freq_for_dev_cfg	1	FLASH 配置时使用的时钟频率 时钟频率定义见19.8.2.1
0x0F	working_mode_por	1	FLASH 上电复位后的工作模式 0 - Extended SPI 1 - XPI(DPI/QPI/OPI) 2 - HyperBUS
0x10	working_mode	1	FLASH 初始化后的工作模式 0 - Extended SPI 1 - XPI(DPI/QPI/OPI) 2 - HyperBUS
0x11	en_diff_clk	1	使能差分时钟 仅 1.8V 的 HyperFLASH 需要将该选项置为 1，其他 FLASH 需置为 0
0x12	data_valid_time	1	-
0x13	en_half_clk_for_ non_read_cmd	1	对非读命令使能降低一半的时钟频率
0x14	clk_freq_for_ non_read_cmd		设置非读命令的时钟频率 时钟频率定义见19.8.2.1
0x15	reserved	3	-
0x18	cs_hold_time	1	设置 CS Hold 的时钟周期数
0x19	cs_setup_time	1	设置 CS Setup 的时钟周期数
0x1A	cs_interval	1	两次命令之间的间隔时钟周期数
0x1B	en_dev_mode_cfg	1	使能设备模式配置
0x1C	flash_state_ctx	4	FLASH 状态上下文 上下文定义见 (TBD)
0x20	mode_cfg_list	4	包含两组，每组 2 个字节，每组的数据结构如下: Byte 0 - 配置命令类型 Byte 1 - 参数大小
0x24	mode_cfg_param	8	包含 2 组，第组 4 个字节，与上述列表里的元素一一对应
0x2C	reserved	4	-
0x30	cfg_instr_seq	32	包含 2 组，每组 16 个字节为一个命令序列，与上述列表里的元素一一对应 详细的命令序列定义见 (TBD)

表 67: XPI NOR 配置块

19.3.3.2 唤醒验证信息

唤醒校验信息提供了唤醒模式下，用来检验唤醒程序完整性和合法性的相关参数，包括唤醒程序入口地址、唤

醒程序代码起始地址，长度以及相应的 SHA256 哈希值。该信息在安全启动使能的条件下，所有的信息均由 ROM 负责签名验证。

详细的数据结构如下：

偏移 (字节)	字段	宽度 (字节)	描述
0x00	Start	4	唤醒代码起始地址
0x04	wake-up entry	4	唤醒代码入口地址 注意：该地址必须在 start 和 length 指定的地址范围内
0x08	Length	4	唤醒代码的长度
0x0C	Reserved	4	-
0x10	wake-up entry HASH	32	start 和 Length 指定范围内代码的哈希值

表 68: 唤醒验证信息 (Wake-up Check Info)

19.3.3.3 OTP 解锁信息

OTP 解锁操作的数据结构如下：

Bit[31:24]	Bit[23:16]	Bit[15:8]	Bit[7:0]
Tag，必须为 0x55	Reserved	Reserved	解锁掩码 当 Lifecycle 为 Closed 或者更高时，BootROM 会默认禁用相应 OTP 字段的写权限。当 BootROM 检测到如下位在 OTP 解锁信息中被置 1 后，会解锁相应的写权限 Bit 0 - HARD_LOCK Bit 1 - LIFECYCLE,PUBKEY_REVOKE 等 Bit 2 - Boot 配置信息部分 Bit 3 - OTP KEY0 (用于 Encrypt boot) Bit 4 - OTP KEY1 (用于 Encrypt boot)

表 69: OTP 解锁信息 (OTP Unlock Info)

19.3.4 签名块 (Signature Block)

完整的签名块提供如下信息:

- 签名块的头 (Signature Block Header)
- 超级根证书表 (SRK Table) - 该部分包含根密钥表头 SRK Table Header 以及 4 份根密钥表项 (SRK0 - SRK3)
- 签名 (Signature)
- 证书 (Certificate) - 该部分是可选的
- 固件 Blob(FW Blob)

签名块中各个部分在目标存储中的布局如图所示：

19.3.4.1 签名块头部 (Signature Block Header)

签名块头部包含如下信息:

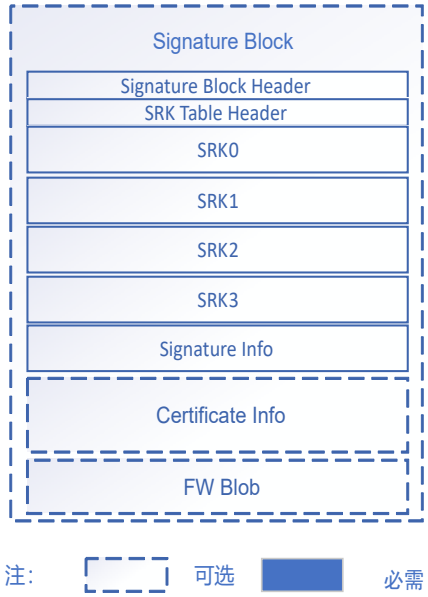


图 17: 签名块中各部分的布局

- 签名块的长度
- 签名块的版本号
- 签名的标识
- 根密钥表 (SRK Table) 的相对偏移
- 证书的相对偏移
- 签名的的相对偏移
- FW Blob 的相对偏移（若 FW Blob 在 Container 内）

详细的数据结构如下表所示：

<table>
<tr><th>偏移
(字节)</th><th>字段</th><th>宽度
(字节)</th><th>描述</th></tr>
<tr><td>0x00</td><td>Header</td><td>4</td><td>签名块 Header
<table><tr><td>Bit[31:16]</td><td>Length - 整个签名块的长度</td></tr><tr><td>Bit [15:8]</td><td>Version - 签名块的版本，本 SoC 固定为 0x10</td></tr><tr><td>Bit [7:0]</td><td>Tag - 签名块标签，固定为 0xD0</td></tr></table></td></tr>
<tr><td>0x04</td><td>Flags</td><td>4</td><td>签名块的标记
Bit0 Key 标识 1 - OEM Key
其它 - 保留位</td></tr>
</table>

<table>
<tr><th>偏移 (字节)</th><th>字段</th><th>宽度 (字节)</th><th>描述</th></tr>
<tr><td>0x08</td><td>Member_Offset</td><td>8</td><td>签名中各个部分的相对偏移
<table>
<tr><td>Bit[15:0]</td><td>srk_table_offset
SRK 表对于签名块首地址的偏移</td></tr>
<tr><td>Bit[31:16]</td><td>signature_offset
签名相对于签名块首地址的偏移</td></tr>
<tr><td>Bit[47:32]</td><td>certificate_offset(可选)
证书相对于签名块首地址的偏移
仅在二级证书被使用的情况下有效</td></tr>
<tr><td>Bit[63:48]</td><td>blob_offset(可选)
固件 BLOB 相对于签名块首地址的偏移
仅当加密固件存在且固件 BLOB 在签名块内里的情况下有效</td></tr>
</table></td></tr>
</table>

表 70: 签名块头 (Signature Block Header)

注意: 为保证 CPU 访问签名块中各个部分的效率，SRK 表，签名，证书，Blob 的起始地址均需按 8 字节对齐

19.3.5 根密钥表 (SRK Table)

SRK Table 由根密钥表头 (SRK Table Header) 以及根密钥表项 (SRK Item) 组成。

19.3.5.1 根密钥表头 (SRK Table Header)

根密钥表头由 4 个字节组成：

Bit[31:16]	Bit[15:8]	Bit[7:0]
SRK 表长 (小端模式存储)	SRK 表版本，当前为 0x10	SRK 表标记, 固定为 0xD1

表 71: 根密钥表头 (SRK Table Header)

19.3.5.2 根密钥表项 (SRK Table Item)

根密钥表项中包含公钥的相关信息，详细数据结构如下：

偏移 (字节)	字段	宽度 (字节)	描述
0x00	Header	4	Bit[31:16] 根密钥表项的长度 (小端模式存储) Bit[15:8] 密钥类型, 0x21 - 公钥 标记 - 必须为 0xE1
0x04	Flags	4	Bit[31-16] -保留 Bit[15:8] 曲线类型: 1 - ECDSA P256, 4 - SM2 Bit[7:0] 哈希类型: 1 - SHA256, 4 - SM3

偏移 (字节)	字段	宽度 (字节)	描述
0x08	Key Info	4	Bit[7:0] - <X Length> : 大数 X 的长度 Bit[23:16] - <Y Length> : 大数 Y 的长度 其他 - 保留位
0x0C	Big Number X	<X Length>	大数 X
0x0C + <X Length>	Big Number Y	<Y Length>	大数 Y

表 72: 根密钥表项 (SRK Table Item)

19.3.6 签名信息 (Signature Info)

签名信息的数据结构如下所示:

偏移 (字节)	字段	宽度 (字节)	描述
0x00	Header	4	Bit[31:16] - <Sig_Len>: 签名信息的长度 (小端模式) Bit[15:8] - 签名信息的版本，当前固定为 0x10 Bit[7:0] - 标记，固定为 0xD2
0x04 0x08	reserved Signature Data	4 <Sig_Len>	保留给未来使用 签名数据

表 73: 签名信息 (Signature Info)

19.3.7 证书 (Certificate)

证书的数据结构如下所示：

偏移 (字节)	字段	宽度 (字节)	描述
0x00	Header	4	Bit[31:16] - <Sig_Len>: 证书的长度 (小端模式) Bit[15:8] - 证书的版本，当前固定为 0x10 Bit[7:0] - 标记，固定为 0xD3
0x04	Offset	4	签名信息相对证书头部的偏移
0x08	SRK Table Item	<SRK_Table_Item_Len>	密钥表项, 详细信息见 (19.3.5.2)
0x08 + <SRK_Table_Item_Len>	Signature Info	<Signature_Info_Len>	签名信息, 详细信息见 (19.3.6)

表 74: 证书头部 (Certificate Header)

19.3.8 固件 BLOB(FW BLOB)

固件 Blob 是一种封装 FW 解密密钥的一种加密打包格式，其定义如下所示：

<table>
<tr><th>偏移 (字节)</th><th>字段</th><th>宽度 (字节)</th><th>描述</th></tr>
<tr><td>0x00</td><td>Header</td><td>4</td><td>Bit[31:16] - <FWBLOB_Len>: 固件 BLOB 长度 (小端模式)
Bit[15:8] - 版本，当前固定为 0x10
Bit[7:0] - 标记，固定为 0xD4</td></tr>
<tr><td>0x04</td><td>Flags</td><td>4</td><td>标记
Byte 0 - Key Source
对称密钥来源: 01 - FMK, 2 - ZMK, 3 - OTP KEY0, 4 - OTP KEY1
Byte 1 - Key Length
密钥长度: 0x10 - 128bit, 0x20 - 256bit
Byte 2 - Encryption Algorithm
加密算法：0x33 - AES, 0x55 - SM4
Byte 3 - Mode
加密模式：0x11-CBC-MAC</td></tr>
<tr><td>0x08</td><td>IV</td><td>16</td><td>初始向量
- 用于解密加密的 Blob 密钥
- 用于解密封装的 KEK</td></tr>
<tr><td>0x18</td><td>Blob 内容</td><td><Blob_Len></td><td>Blob 内容
<table>
<tr><th>偏移</th><th>长度 (字节)</th><th>字段</th><th>描述</th></tr>
<tr><td>0x00</td><td>0x10 /0x20</td><td>Encrypted Blob Key</td><td>加密的 Blob 密钥
该密钥用于解压加密打包的 DEK(DEK 用于加密的固件解密)</td></tr>
<tr><td>0x10 / 0x20</td><td>0x10 / 0x20</td><td>Wrapped DEK</td><td>加密打包的 DEK</td></tr>
<tr><td>0x20 /0x40</td><td>16</td><td>MAC</td><td>信息认证码</td></tr>
</table></td></tr>
</table>

表 75: 固件 Blob(FW Blob)

KEK 由 AES_CCM 或者 SM4_CCM 算法来封装，其中：

- AAD 为空
- NONCE 为 IV 字段的前 13 字节
- MAC 的长度为 16 字节

BootROM 根据 Key source 指定的 Key 以及选用的算法（AES/SM4) 来从 Encrypted Blob Key 中提取 Blob Key, 并从 Blob 中提取 DEK，随后，结合 FW 里的 IV 以及 DEK 来解密固件，并对解密后的固件开展哈希值校验。

19.3.9 命令容器 (Command Container)

命令容器是一种对 ROM 支持的若干命令的打包封装，该格式支持加密 (AES-CBC/SM4-CBC) 或者明文两种形式，用户可根据自己系统的安全性要求选择对应的打包方式。

命令容器数据包按块打包，块的长度为变长，由前一个块的末尾的 next_block_info 字段来指示下一个块的长

度。第一个块的固定为命令容器头，长度固定。最后一个块不包含 next_block_info。

偏移 (字节)	字段	宽度 (字节)	描述
0x00	Payload	<Data_Bytes>	数据载荷部分
<Data_Bytes>	next_block_info	16	下一个块的信息 Byte 0-1：下一个块的大小 Byte 2-3：是否为最后一个块 Byte 4-15: 保留
<Data_Bytes> + 16	next_block_hash	16	下一个块的哈希值。哈希值的长度根据62中的 hash_type 来决定

表 76: 命令容器 (Command Container)

19.3.9.1 命令容器头 (Command Container Header)

偏移 (字节)	字段	宽度 (字节)	描述
0x00	Tag	4	命令容器标记：0x33764343(ASCII:CCv1)
4	Version	4	命令容器格式的版本，当前为 0x56010000(V1.0.0)
8	Flags	4	保留给未来使用
12	Reserved	48	保留给未来使用
64	Description	48	命令容器的简单描述，如 (xpi nor boot image)

表 77: 命令容器头部 (Command Container Header)

19.3.9.2 命令容器支持的命令

命令容器支持如下的命令：

- configure-memory
- write-memory
- erase
- reset

Configure-memory 的数据块结构如下：

偏移 (字节)	字段	宽度 (字节)	描述
0x00	Command Tag	1	命令标识 - 0x03
0x01	Reserved	1	保留
0x02	Command Misc.	1	命令杂项 - 固定为 0
0x03	Reserved	1	保留
0x4	Memory Identifier	4	存储设备标识 0x10000 - XPI0 0x10001 - XPI1

偏移 (字节)	字段	宽度 (字节)	描述
0x08	Reserved	4	保留
12	Configuration Block Data Bytes	4	配置块数据字节
16	Configuration Block Data	<Configuration Block Data Bytes>	配置块数据 当数据长度不足 16 字节时，需按 16 字节对齐补 0 填充

表 78: 配置存储设备数据块

Write-memory 数据块结构如下：

偏移 (字节)	字段	宽度 (字节)	描述
0x00	Command Tag	1	命令标识 - 0x04
0x01	Reserved	1	保留
0x02	Command Misc.	1	命令杂项 - 固定为 0
0x03	Reserved	1	保留
0x4	Memory Identifier	4	存储设备标识 0x10000 - XPI0 0x10001 - XPI1 0x20000 - OTP
0x08	Start	4	起始地址
12	Data Bytes	4	要写入的字节数
16	Data Block	<Data Bytes>	要写入的数据 (最多 256 字节) 当数据长度不足 16 字节时，需按 16 字节对齐补 0 填充

表 79: 写存储设备数据块

Erase 数据块结构如下：

偏移 (字节)	字段	宽度 (字节)	描述
0x00	Command Tag	1	命令标识 - 0x07
0x01	Reserved	1	保留
0x02	Command Misc.	1	命令杂项: 0x0 - Sector Erase 0x1 - Chip Erase
0x03	Reserved	1	保留
0x4	Memory Identifier	4	存储设备标识 0x10000 - XPI0 0x10001 - XPI1

偏移 (字节)	字段	宽度 (字节)	描述
0x08	Start	4	起始地址
12	Length	4	要擦除的字节数 (Chip Erase 时忽略)

表 80: 擦除存储设备数据块

Reset 数据块结构如下：

偏移 (字节)	字段	宽度 (字节)	描述
0x00	Command Tag	1	命令标识 - 0x08
0x01	Reserved	1	保留
0x02	Command Misc.	1	命令杂项，固定为 0
0x03	Reserved	1	保留
0x4	Reset Type	4	复位类型，在该 SoC 中为保留位
0x08	Reserved	8	保留位

表 81: 复位数据块

19.3.9.3 加密命令容器 (Encrypted Command Container)

命令容器支持按 AES-CBC/SM4-CBC 模式加密。其中 IV 为当前块的哈希值的前 128 位。命令容器的加密密钥被打包在 FW Blob 中。命令容器头的 IV 为固件信息表中哈希值的前 128 位。

19.4 安全启动 (Secure Boot)

19.4.1 安全启动简介

本设备支持安全启动。

安全启动支持基于公共密钥基础设施 (PKI) 派生出来的数字证书的签名和认证。

本设备的安全启动支持如下两种证书签名和认证模式：

1. 根据 CA 生成的一级证书的直接签名认证，其中一级证书最多支持 4 个。
2. 根据一级证书签发的二级证书的签名认证，采取信任链的方式来逐级认证，其中其中一级证书最多支持 4 个，二级证书最多支持 1 个。

本设备支持 ECDSA P256 和 SM2 签名认证。

为简化整个安全启动的流程，减少不相关的格式转换和解析，BootROM 对如下部分采用了固定的编码：

- 公钥编码 - 详见根密钥表小节
- 数字签名编码 - 详见签名信息小节
- 证书 - 详见证书小节

19.4.2 安全启动流程

为缩短整个安全启动的时间，安全启动的设计模型如下：

1. 只对 FW Container 头部进行验签，其中被签名/验签的数据如下图所示:

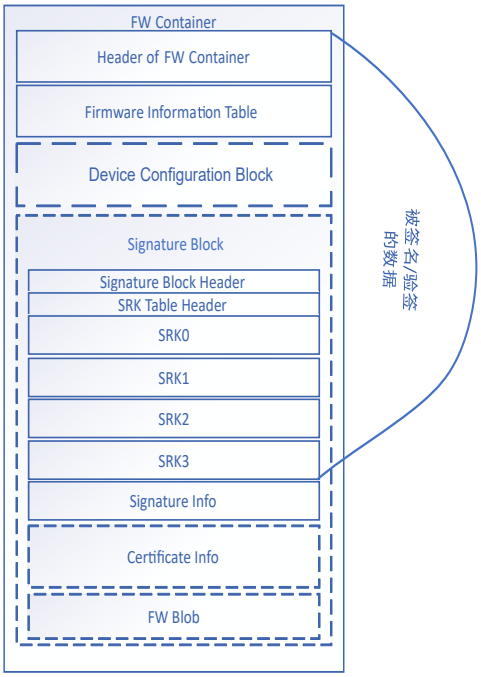


图 18: 要被签名的固件头部分

2. 对 FW 采取比对哈希值的方式，其中 FW 的哈希值包含在固件信息表中，如上图所示。所以哈希值本身已被验签，这种设计保证了签名的唯一性。
3. 当 FW 头部验签通过后，BootROM 解析固件表，根据该表里指定的检验方式分别处理各个 FW 的验证。本设备支持两种验证方法:
    - 基于 SHA256 或 SM3 的明文哈希值校验
    - 基于 AES-CCM 或者 SM4-CCM 的解密校验
4. 当 FW Container 头检验和各固件校验都通过后，则整个启动镜像安全认证通过，BootROM 支持按需启动或者处理相应的固件。如果上述任何环节检验失败，则认为启动镜像不合法。

19.4.3 固件容器验签

对于支持一级证书的安全启动，其流程如下：

1. BootROM 解析 FW Container，获取签名块，并从签名块中提取 SRK 表, 并计算 SRK 表的哈希值
2. BootROM 从 OTP 中获取 SRK 表的哈希值（根据 Container 头中的 SRK_SET 来判断 SRK 表哈希值来源），并与上一步中计算出来的哈希值进行比对，如果匹配，则代表 SRK 为有效值
3. 计算从 FW Container 头到 SRK 表结尾的数据的哈希值作为被签名的数据
4. 用 FW Container 头部指定的 SRK 来对签名部分进行验签，如果验签通过，则代表 FW Container 头合法，并未被篡改。

对于支持二级证书的安全启动，其流程如下：

1. BootROM 解析 FW Container，获取签名块，并从签名块中提取 SRK 表, 并计算 SRK 表的哈希值
2. BootROM 从 OTP 中的 SRK_HASH 字段获取 SRK 表的哈希值（根据 Container 头中的 SRK_SET 来判断 SRK 表哈希值来源），并与上一步中计算出来的哈希值进行比对，如果匹配，则代表 SRK 为有效值
3. 计算证书块中证书的哈希值作为二级证书的验签数据

4. 用 FW Container 头部指定的 SRK 来对签名部分进行验签，如果验签通过，则代表二级证书合法，否则安全启动失败
5. 计算从 FW Container 头到 SRK 表结尾的数据的哈希值作为被签名的数据
6. 用二级证书对从上一步的数据进行验签。如果验签通过，则代表 FW Container 头合法，并未被篡改

19.4.4 固件的验签

当 FW Container 验签通过后，若设备配置块存在，BootROM 会根据设备配置块的参数完成设备的配置，然后进入到固件信息的处理环节。ROM 支持两种固件的验签方式：

- 基于明文的哈希校验，BootROM 会根据固件信息表里指定的地址和长度来完成 FW 哈希值的计算，并将结果与固件信息表中的结果进行比对。
  当 FW Container 的验签公钥类型为 ECDSA P256 时，固件的哈希值基于 SHA256 算法
  当 FW Container 的验签公钥类型为 SM2 时，固件的哈希值基于 SM3 算法或 SHA256 算法
- 基于 CBC（AES-CBC/SM4-CBC) 解密以及哈希检验。BootROM 采取加解密边校验的方式按固件信息表里指定的加载地址和长度进行处理。
  当 FW Container 的验签公钥类型为 ECDSA P256 时，固件采用 AES-CBC + SHA256 进行解密并验证哈希值。
  当 FW Container 的验签公钥类型为 SM2 时，固件采用 SM4-CBC +SM3 / AES-CBC+SHA256 进行解密并验证哈希值。

注意：SM3 和 SM4-CBC 基于软件实现，性能上弱于基于硬件的 SHA256 和 AES-CBC。

19.4.5 公钥的撤销

随着产品使用时间的增长，当前使用的公钥可能会面临过期或者私钥泄露等风险，当这种情况发生时，当前使用的公钥需要被撤销。BootROM 不提供对公钥的自动撤销，用户程序需要按需去撤销相应的公钥。BootROM 提供了两种撤销公钥的方式。

- 通过 FW Container 撤销。
  将固件容器头中的 SRK_REVOKE 中对应当前密钥的位置 1，并将新镜像用新的私钥签名，将新镜像烧入存储介质或者通过串行启动模式加载。当新镜像加载完成后，BootROM 会自动完成指定公钥的撤销。
- 通过用户程序撤销。
  在镜像中使用设备配置信息块，并解锁 PUK_REVOKE 的写权限。并将新镜像用新的私钥签名，将新镜像烧入存储介质或者通过串行启动模式加载。当新镜像被成功引导后，用户可通过 OTP 驱动来写 PUK_REVOKE 实现公钥的撤销。

19.5 低功耗唤醒

设备在特定的低功耗模式下，CPU 内核会掉电，只有部分必要的模块处于上电状态，在特定的事件下（如按下唤醒键) CPU 内核可被唤醒并重新上电运行 BootROM 代码，并经由 BootROM 代码实现后续的唤醒支持。ROM 支持如下模式的低功耗唤醒选项：

- 快速跳转：直接跳转到用户指定的唤醒地址执行，只做基本的地址范围合法性校验
- 强校验跳转：按用户指定的范围做完整性检验 (SHA256)
- 禁用低功耗唤醒：强制冷启动，忽略低功耗唤醒选项

19.5.1 快速跳转

用户需要在 SYSCTL.CPU0_GPR0 处填入合法的唤醒地址，并配置好正确的唤醒源。若地址不合法，BootROM 会进行完整的启动流程。

示例代码如下：

```
HPM_SYSCTL->CPU0_LP |= SYSCTL_CPU0_LP_STOP_MASK;
HPM_SYSCTL->RETENTION[0].VALUE = 0;
for (uint32_t i = 0; i < 4; i++) {
    HPM_SYSCTL->CPU0_WAKEUP_ENABLE[i] = ~0;
}
HPM_PMIC_WDG->WREN = 0x5AA5UL;
HPM_PMIC_WDG->CTRL = WDT_CTRL_INTTIME(2) | WDT_CTRL_RSTEN(0) | WDT_CTRL_INTEN(1) \
| WDT_CTRL_CLKSEL(1) | WDT_CTRL_EN(1);
HPM_SYSCTL->CPU0_GPR0 = (uint32_t)&lowpower_wakeup_test;
__asm("wfi");
```

19.5.2 强检验跳转

当 FORCE_WAKEUP_ENTRY_CHK OTP 位被置上后，BootROM 会强制执行唤醒代码的完整性检查，只有完整性检验通过后，BootROM 才会执行唤醒代码，否则 BootROM 会执行完整的启动流程。除快速跳转配置外，用户还需要将程序的起始地址，长度填入 SYSCTL.CPU0_PARAM[0], SYSCTL.CPU0_PARAM[1] 并将以及整个 wake-up 代码的 SHA256 哈希值填入 SYSCTL.CPU0_DATA[0] - SYSCTL.CPU0_DATA[7]。其中 SYSCTL.CPU_GPR0 的地址入口必须在 CPU0_GPR1 和 CPU0_GPR2 指定的地址范围内。

当 CPU0 被唤醒后，ROM 会根据 SYSCTL.CPU0_PARAM[0] 以及 SYSCTL.CPU0_PARAM[1] 的地址范围来计算 SHA256 哈希值并与 SYSCTL.CPU0_DATA[0] - SYSCTL.CPU0_DATA[7] 中的值比对，当有当结果匹配时，BootROM 才会跳转到唤醒地址入口。

为保证安全性，用户可通过 SYSCTL.CPU0_LOCK 寄存器锁定上述寄存器。

开启方式：

- 在用户代码上按上述描述来配置
- 在设备配置信息块中配置，详细信息见唤醒验证信息小节。当用户需要 BootROM 确保唤醒代码也被安全启动认证时，推荐使用该方式。在该模式下，BootROM 会在跳转到用户程序前锁定上述寄存器。

19.5.3 禁用低功耗唤醒

当 FORCE_COLD_BOOT OTP 位被置上后，BootROM 会忽略唤醒请求，执行完整的安全启动流程。

19.6 Debug 接口权限管理

SoC 的调试端口的访问受 SoC 硬件和 ROM 共同管理。

- 当设备处于 CREATE 模式时，Debug 口默认处于开启状态，BootROM 的执行可以随时被打断接管。
- 当设备出厂后，Debug 口默认处于禁止状态，Debug 端口控制由 BootROM 接管。当 SoC 离开 BootROM 时，BootROM 会开启 DEBUG 端口的访问权限。
  - 在 ISP boot 模式下，当 BootROM 检测到 DEBUG 请求后，会打开 DEBUG 端口访问权限并进入 while(1) 循环。

– 在有正常启动镜像的情况下，BootROM 会在跳转到 APP 之前打开 DEBUG 端口访问权限并继续执行用户应用程序。

19.7 ROM API

本节详细介绍该 SoC 支持的所有 API 接口。

19.7.1 简介

BootROM 导出了常用的 FLASH 操作 API 接口、安全相关的 API 接口。用户程序可直接调用这些接口进行快速的 FLASH 和安全相关的编程，以节省开发和验证的时间。API 以模块为单位来分类，以下为当前 SoC 所支持的 API 大类：

- run_bootloader 接口
- XPI NOR 驱动接口
- XPI 底层驱动接口
- OTP API 接口
- 安全启动 API 接口
- SDP API 接口
- EXIP API 接口

本 SoC 中，BootROM 提供的 API 以位于 0x2001_FF00 的 API 根列表为基础来组织。其中 API 根的数据结构如下:

```
typedef struct {
    const uint32_t version;                                  /*!< offset: 0x00 */
    const char *copyright;                                   /*!< offset: 0x04 */
    hpm_stat_t (*const run_bootloader)(void *arg);           /*!< offset: 0x08 */
    const otp_driver_interface_t *otp_driver_if;             /*!< offset: 0x0c */
    const xpi_driver_interface_t *xpi_driver_if;             /*!< offset: 0x10 */
    const xpi_nor_driver_interface_t *xpi_nor_driver_if;     /*!< offset: 0x14 */
    const xpi_ram_driver_interface_t *xpi_ram_driver_if;     /*!< offset: 0x18 */
    const sdp_driver_interface_t *sdp_driver_if;             /*!< offset: 0x1c */
    const sec_boot_api_interface_t *sec_boot_api_if;         /*!< offset: 0x20 */
} bootloader_api_table_t;

#define ROM_API_TABLE_ROOT  (const bootloader_api_table_t*)0x2001FF00U
```

19.7.2 run_bootloader API

用户程序可通过调用该 API 跳转到指定的启动模式 (如进入 ISP 模式或选择启动另一个镜像)。其原型定义为：

```
typedef struct {
    uint32_t index:8;
    uint32_t peripheral:8;
    uint32_t src:8;
    uint32_t tag:8;
```

```
}api_boot_arg_t;

#define API_BOOT_TAG  (0xEBU)
#define API_BOOT_SRC_FUSE (0U)
#define API_BOOT_SRC_PRIMARY (1U)
#define API_BOOT_SRC_SERIAL_BOOT (2U)
#define API_BOOT_SRC_ISP (3U)
#define API_BOOT_SRC_MAX    (API_BOOT_SRC_ISP)
#define API_BOOT_PERIPH_AUTO (0U)
#define API_BOOT_PERIPH_UART (1U)
#define API_BOOT_PERIPH_USBHID (2U)
#define API_BOOT_PERIPH_MASK (API_BOOT_PERIPH_USBHID)

void run_bootloader(void *arg);
```

19.7.2.1 示例:

用户在应用程序中想进入在系统编程模式，可通过如下方式进入:

```
api_boot_arg_t boot_arg = {.index = 0,
            .peripheral = API_BOOT_PERIPH_AUTO,
            .src = API_BOOT_SRC_ISP,
            .tag = API_BOOT_TAG};
ROM_API_TABLE_ROOT->run_bootloader(&boot_arg);
```

19.7.3 OTP API

OTP API 提供了 OTP 的读和写操作。其原型定义如下：

```
typedef struct {
    uint32_t version;
    void (*init)(void);
    uint32_t reserved;
    uint32_t (*read_from_shadow)(uint32_t addr);
    uint32_t (*read_from_ip)(uint32_t addr)
    hpm_stat_t (*program)(uint32_t addr, const uint32_t *src, uint32_t num_of_words);
    hpm_stat_t (*reload)(efuse_region_t region);
    hpm_stat_t (*lock_otp)(uint32_t addr, otp_lock_option_t lock_option);
    hpm_stat_t (*lock_shadow)(uint32_t addr, otp_lock_option_t lock_option);
    hpm_stat_t (*set_configurable_region)(uint32_t start, uint32_t num_of_words);
    hpm_stat_t (*write_shadow_register)(uint32_t addr, uint32_t data);
} otp_driver_interface_t;
```

19.7.3.1 init

该 API 用于初始化 OTP 驱动，在调用 OTP 的读写操作前，确认该 API 已被调用过。

19.7.3.2 read_from_shadow

该 API 用于从 OTP Shadow 寄存器中读取指定的 OTP 字。

19.7.3.3 read_from_ip

该 API 用于用 IP 读的方式从 OTP 中读取指定的 OTP 字。

19.7.3.4 program

该 API 用于向指定的 OTP 地址范围内写入数据。该 API 的参数定义如下：

- addr - 指定的写入地址
- src - 要写入的数据的指针
- num_of_words - 要写入的字数 (单位 32-bit)

19.7.3.5 reload

该 API 用于重回加载指定区域的 shadow 寄存器的值。其中参数的定义如下：

```
typedef enum {
    efuse_region0_mask = 1U,   /*!< Address range: [0, 7]   */
    efuse_region1_mask = 2U,   /*!< Address range: [8, 15]  */
    efuse_region2_mask = 4U,   /*!< Address range: [16, 127] */
    efuse_region3_mask = 8U,   /*!< Address range: user defined */
}otp_region_t;
```

19.7.3.6 lock_otp

该 API 可锁定指定 OTP 字的相应权限。其参数定义如下所示：

```
    typedef enum {
        efuse_no_lock = 0,
        efuse_read_only = 1,
        efuse_permanent_no_lock = 2,
        efuse_disable_access = 3,
        efuse_lock_option_max = efuse_disable_access,
    }otp_lock_option_t;
```

19.7.3.7 set_configuratable_region

该 API 用于设置用户指定的可配置区域。

19.7.3.8 write_shadow_register

该 API 可直接向指定的 shadow 寄存器写入指定的值。

19.7.3.9 示例

如用户想读取 OTP 的第 26 个字。可通过如下两种方式来获取: 方式一: 通过 Shadow 寄存器获取:

```
uint32_t fuse_word = ROM_API_TABLE_ROOT->otp_driver_if->read_from_shadow(26);
```

方式二：通过 IP 读来获取:

```
uint32_t fuse_word = ROM_API_TABLE_ROOT->otp_driver_if->read_from_ip(26);
```

19.7.4 XPI 底层驱动 API

XPI 底层 API 提供了 XPI 模块特定功能的配置以及数据的传输，其 API 接口原型定义如下所示::

```
typedef struct {
    uint32_t version;
    hpm_stat_t (*get_default_config)(xpi_config_t *xpi_config);
    hpm_stat_t (*get_default_device_config)(xpi_device_config_t *dev_config);
    hpm_stat_t (*init)(XPI_Type *base, xpi_config_t *xpi_config);
    hpm_stat_t (*config_ahb_buffer)(XPI_Type *base, xpi_ahb_buffer_cfg_t *ahb_buf_cfg);
    hpm_stat_t (*config_device)(XPI_Type *base, xpi_device_config_t *dev_cfg, 4
                xpi_port_t port);
    hpm_stat_t (*update_instr_table)(XPI_Type *base, const uint32_t *inst_base,
                uint32_t seq_idx, uint32_t num);
    hpm_stat_t (*transfer_blocking)(XPI_Type *base, xpi_xfer_ctx_t *xfer);
    void (*software_reset)(XPI_Type *base);
    bool (*is_idle)(XPI_Type *base);
    void (*update_dllcr)(XPI_Type *base, uint32_t serial_root_clk_freq,
                uint32_t data_valid_time, xpi_port_t port, uint32_t dly_target);

    hpm_stat_t (*get_abs_apb_xfer_addr)(XPI_Type *base, xpi_xfer_port_t port,
                uint32_t in_addr, uint32_t *out_addr);
    hpm_stat_t (*iomux_init)(XPI_Type *base, const xpi_io_config_t *io_cfg);
    hpm_stat_t (*clock_init)(XPI_Type *base, const xpi_clk_config_t *clk_cfg);
} xpi_driver_interface_t;
```

19.7.4.1 get_default_config

本 API 用于获取 XPI 默认的配置参数。xpi_config_t 定义详见 SDK。

示例:

```
xpi_config_t xpi_config;
hpm_stat_t status = ROM_API_TABLE_ROOT->xpi_driver_if->get_default_config(&xpi_config);
```

19.7.4.2 get_default_device_config

本 API 用于获取设备的默认配置参数。xpi_device_config_t 定义见 SDK。

示例:

```
xpi_device_config_t device_config;
status = ROM_API_TABLE_ROOT->xpi_driver_if->get_default_default_config(&device_config);
```

19.7.4.3 init

根据 xpi_config 参数配置 XPI 控制器.

示例:

```
status = ROM_API_TABLE_ROOT->xpi_driver_if->init(HPM_XPI0, &xpi_config);
```

19.7.4.4 config_ahb_buffer

配置 AHB Buffer 的信息。xpi_ahb_buffer_cfg_t 的定义如下:

```
typedef struct
{
    struct
    {
        uint8_t priority;           /* Offset: 0x00 */
        uint8_t master_idx;         /* Offset: 0x01 */
        uint8_t buf_size_in_dword;  /* Offset: 0x02 */
        bool enable_prefetch;       /* Offset: 0x03 */
    }entry[XPI_AHB_BUF_CR_COUNT];
}xpi_ahb_buffer_cfg_t;
```

示例:

```
xpi_ahb_buffer_cfg_t ahb_cfg;
for(uint32_t i=0; i<XPI_AHB_BUF_CR_COUNT; i++) {
    ahb_cfg.entry[i].enable_prefetch = false;
}
ahb_cfg.entry[0].priority = 0;
ahb_cfg.entry[0].master_idx = 0;
ahb_cfg.entry[0].buf_size_in_dword = 64;
ahb_cfg.entry[0].enable_prefetch = true;
status = ROM_API_TABLE_ROOT->xpi_driver_if->config_ahb_buffer(HPM_XPI0, &ahb_cfg);
```

19.7.4.5 config_device

将 Device 信息配置到 XPI 控制器。

示例:

```
status = ROM_API_TABLE_ROOT->xpi_driver_if->config_device(HPM_XPI0, &device_config);
```

19.7.4.6 update_instr_table

将指令表更新到 XPI 控制器。

示例:

```
uint32_t lut[4] = {XPI_INSTR_SEQ(XPI_PHASE_CMD_SDR, XPI_1PAD, 0x03,
                                 XPI_PHASE_RADDR_SDR, XPI_1PAD,0x18),
                   XPI_INSTR_SEQ(XPI_PHASE_READ_SDR, XPI_1PAD, 0x04,
                                 XPI_PHASE_STOP, XPI_1PAD, 0x00),
                   0, 0};
status = ROM_API_TABLE_ROOT->xpi_driver_if->update_instr_table(HPM_XPI0,
                                                               &lut[0], 0, 1);
```

19.7.4.7 transfer_blocking

完成阻塞式的传输 (读/写/发命令)。示例：

```
uint32_t buffer[128];
xpi_xfer_ctx_t xfer = {0};
xfer.cmd_type = xpi_apb_xfer_type_read;
xfer.addr = 0x1000;
xfer.seq_idx = 0;
xfer.seq_num = 1;
xfer.port = xpi_xfer_port_auto;
xfer.buf = &buffer[0];
xfer.xfer_size = sizeof(buffer);
status = ROM_API_TABLE_ROOT->xpi_driver_if->transfer_blocking(HPM_XPI0, &xfer);
```

19.7.4.8 software_reset

对 XPI 模块进行软复位。示例：

```
ROM_API_TABLE_ROOT->xpi_driver_if->software_reset(HPM_XPI0);
```

19.7.4.9 is_idle

判断 XPI 模块是否空闲。示例：

```
bool is_idle = ROM_API_TABLE_ROOT->xpi_driver_if->is_idle(HPM_XPI0);
```

19.7.4.10 update_dllcr

更新延时线 (delay line) 设置。示例：

```
uint32_t serial_root_clk_freq = 133000000UL;
uint8_t data_valid_time = 100;
xpi_port_t port = xpi_port_a1;
uint8_t delay_target = 8;
```

```
ROM_API_TABLE_ROOT->xpi_driver_if->update_dllcr(HPM_XPI0, serial_root_clk_freq,
                                                data_valid_time, port, delay_target);
```

19.7.5 XPI NOR API

XPI NOR API 提供了对市面上大多数 FLASH 设备的支持，可以方便的进行 FLASH 的读写操作，其 API 接口原型定义如下所示:

```
typedef struct {
    uint32_t version;
    hpm_stat_t (*get_config)(XPI_Type *base,
               xpi_nor_config_t *nor_cfg,
               xpi_nor_config_option_t *cfg_option);
    hpm_stat_t (*init)(XPI_Type *base, xpi_nor_config_t *nor_config);
    hpm_stat_t (*enable_write)(XPI_Type *base,
               xpi_xfer_port_t port,
               const xpi_nor_config_t *nor_config, uint32_t addr);
    hpm_stat_t (*get_status)(XPI_Type *base, xpi_xfer_port_t port,
               const xpi_nor_config_t *nor_config, uint32_t addr,
               uint16_t *out_status);
    hpm_stat_t (*wait_busy)(XPI_Type *base, xpi_xfer_port_t port,
               const xpi_nor_config_t *nor_config, uint32_t addr);
    hpm_stat_t (*erase)(XPI_Type *base, xpi_xfer_port_t port,
               const xpi_nor_config_t *nor_config, uint32_t start,
               uint32_t length);
    hpm_stat_t (*erase_chip)(XPI_Type *base, xpi_xfer_port_t port,
               const xpi_nor_config_t *nor_config);
    hpm_stat_t (*erase_sector)(XPI_Type *base, xpi_xfer_port_t port,
               const xpi_nor_config_t *nor_config, uint32_t addr);
    hpm_stat_t (*erase_block)(XPI_Type *base, xpi_xfer_port_t port,
               const xpi_nor_config_t *nor_config, uint32_t addr);
    hpm_stat_t (*program)(XPI_Type *base, xpi_xfer_port_t port,
               const xpi_nor_config_t *nor_config, const uint32_t *src,
               uint32_t dst_addr, uint32_t length);
    hpm_stat_t (*read)(XPI_Type *base, xpi_xfer_port_t port,
               const xpi_nor_config_t *nor_config, uint32_t *dst,
               uint32_t start, uint32_t length);
    hpm_stat_t (*page_program_nonblocking)(XPI_Type *base, xpi_xfer_port_t port,
               const xpi_nor_config_t *nor_config, const uint32_t *src,
               uint32_t dst_addr, uint32_t length);
    hpm_stat_t (*erase_sector_nonblocking)(XPI_Type *base, xpi_xfer_port_t port,
               const xpi_nor_config_t *nor_config, uint32_t addr);
    hpm_stat_t (*erase_block_nonblocking)(XPI_Type *base, xpi_xfer_port_t port,
               const xpi_nor_config_t *nor_config, uint32_t addr);
```

```
    hpm_stat_t (*erase_chip_nonblocking)(XPI_Type *base, xpi_xfer_port_t port,
                const xpi_nor_config_t *nor_config);
    hpm_stat_t (*restore_spi_protocol)(XPI_Type *base, xpi_xfer_port_t port,
                xpi_nor_config_t *config, flash_run_context_t *run_ctx);
    hpm_stat_t (*write_persistent)(const uint32_t data);
    hpm_stat_t (*read_persistent)(uint32_t *data);
    hpm_stat_t (*auto_config)(XPI_Type *base, xpi_nor_config_t *nor_cfg,
                xpi_nor_config_option_t *cfg_option);
    hpm_stat_t (*get_property)(XPI_Type *base, xpi_nor_config_t *nor_cfg,
                uint32_t property_id, uint32_t *value);
    hpm_stat_t (*set_property)(XPI_Type *base, xpi_nor_config_t *nor_cfg,
                uint32_t property_id, uint32_t value);
} xpi_nor_driver_interface_t;
```

19.7.5.1 get_config

根据 FLASH 配置选项来获取 FLASH 的配置。该 API 支持市面上大多数的 4 线和 8 线串行 NOR FLASH。详细的定义见XPI NOR 配置信息。

示例：

```
xpi_nor_config_option_t flash_option = {0};
xpi_nor_config_t flash_config;
flash_option.header.U = 0xfcf90001;
flash_option.option0.U = 0x00000007;

hpm_stat_t status = ROM_API_TABLE_ROOT->xpi_nor_driver_if->get_config(HPM_XPI0,
                                        &flash_config, &flash_option);
```

19.7.5.2 init

根据 FLASH 配置初始化 XPI。示例：

```
status = ROM_API_TABLE_ROOT->xpi_nor_driver_if->init(HPM_XPI0, &flash_config);
```

19.7.5.3 enable_write

使能 FLASH 写操作。示例：

```
status = ROM_API_TABLE_ROOT->xpi_nor_driver_if->enable_write(HPM_XPI0,
                                    xpi_xfer_port_auto, &flash_config);
```

19.7.5.4 get_status

获取 FLASH 当前的状态。示例：

```
uint16_t flash_status;
status = ROM_API_TABLE_ROOT->xpi_nor_driver_if->get_status(HPM_XPI0,
```

```
                xpi_xfer_port_auto , &flash_config , 0, &flash_status );
```

19.7.5.5 wait_busy

FLASH 忙等待操作。API 在 FLASH 忙时会一直阻塞式等待。示例：

```
status = ROM_API_TABLE_ROOT->xpi_nor_driver_if->wait_busy(HPM_XPI0,
            xpi_xfer_port_auto , &flash_config , 0);
```

19.7.5.6 erase

阻塞式擦除指定的 FLASH 区间，该命令依次执行 FLASH 擦除和忙等待操作。示例：

```
status = ROM_API_TABLE_ROOT->xpi_nor_driver_if->erase(HPM_XPI0,
                xpi_xfer_port_auto , &flash_config , 0, 0x2000 );
```

19.7.5.7 erase_chip

阻塞式擦除整块 FLASH。该命令依次执行 FLASH 擦除和忙等待操作。示例：

```
status = ROM_API_TABLE_ROOT->xpi_nor_driver_if->erase_chip(HPM_XPI0,
                xpi_xfer_port_auto , &flash_config );
```

19.7.5.8 erase_sector

阻塞式擦除指定的 FLASH 扇区。该命令依次执行 FLASH 擦除和忙等待操作。示例：

```
status = ROM_API_TABLE_ROOT->xpi_nor_driver_if->erase_sector(HPM_XPI0,
                    xpi_xfer_port_auto , &flash_config , 0x4000 );
```

19.7.5.9 erase_block

阻塞式擦除指定的 FLASH 块。该命令依次执行 FLASH 擦除和忙等待操作。示例：

```
status = ROM_API_TABLE_ROOT->xpi_nor_driver_if->erase_block(HPM_XPI0,
                    xpi_xfer_port_auto , &flash_config , 0x4000 );
```

19.7.5.10 program

阻塞式烧写数据到指定的 FLASH 区域。该命令依次执行 FFLASH 写操作和忙等待操作示例：

```
uint32_t buffer[64];
uint8_t *buf_8 = (uint8_t*)&buffer[0];
for(uint32_t i=0; i<256; i++) {
    buf_8[i] = (uint8_t)i & 0xFF;
}
status = ROM_API_TABLE_ROOT->xpi_nor_driver_if->program(HPM_XPI0,
        xpi_xfer_port_auto , &flash_config , buffer , 0, sizeof(buffer));
```

19.7.5.11 read

阻塞式读 FLASH 指定的区域。示例：

```
uint32_t buffer[64];
status = ROM_API_TABLE_ROOT->xpi_nor_driver_if->read(HPM_XPI0,
                xpi_xfer_port_auto, &flash_config, buffer, 0, sizeof(buffer));
```

19.7.5.12 page_program_nonblocking

非阻塞式的写数据到指定 FLASH 页。该 API 只负责完成 PAGE Program 命令的发送，用户程序需要完成后续的忙等待操作。示例：

```
uint32_t buffer[64];
uint8_t *buf_8 = (uint8_t*)&buffer[0];
for(uint32_t i=0; i<256; i++) {
    buf_8[i] = (uint8_t)i & 0xFF;
}
status = ROM_API_TABLE_ROOT->xpi_nor_driver_if->page_program_nonblocking(HPM_XPI0,
            xpi_xfer_port_auto, &flash_config, buffer, 0, sizeof(buffer));
```

19.7.5.13 erase_sector_nonblocking

非阻塞式的擦除指定的 FLASH 扇区，该 API 只负责完成 Sector Erase 命令的发送，用户程序需要完成后续的忙等待操作。示例：

```
status = ROM_API_TABLE_ROOT->xpi_nor_driver_if->erase_sector_nonblocking(HPM_XPI0,
            xpi_xfer_port_auto, &flash_config, 0x4000);
```

19.7.5.14 erase_block_nonblocking

非阻塞式的擦除指定的 FLASH 块, 该 API 只负责完成 Block Erase 命令的发送，用户程序需要完成后续的忙等待操作。示例：

```
status = ROM_API_TABLE_ROOT->xpi_nor_driver_if->erase_block_nonblocking(HPM_XPI0,
            xpi_xfer_port_auto, &flash_config, 0x4000);
```

19.7.5.15 erase_chip_nonblocking

非阻塞式的擦除整片 FLASH 空间, 该 API 只负责完成 Chip Erase 命令的发送，用户程序需要完成后续的忙等待操作。示例：

```
status = ROM_API_TABLE_ROOT->xpi_nor_driver_if->erase_chip_nonblocking(HPM_XPI0,
            xpi_xfer_port_auto, &flash_config, 0x4000);
```

19.7.6 安全启动 API

安全启动 API 提供了对兼容 BootROM 启动镜像的数据的安全认证支持。其 API 接口原型定义如下所示：

```
typedef struct {
    uint32_t version;
    secure_status_t (*init)(rom_core_ctx_t *ctx, uint32_t core_id);
    secure_status_t (*auth_img_hdr)(rom_core_ctx_t *ctx,
                    const boot_image_hdr_t *img_hdr,
                    const uint8_t *blob,
                    const uint8_t *srk_hash);
    secure_status_t (*auth_fw)(rom_core_ctx_t *ctx,
                    const firmware_info_t *fw_info);
    secure_status_t (*hdl_img)(rom_core_ctx_t *ctx,
                    const image_buf_t *img_hdr_buf,
                    const boot_interface_t *boot_if,
                    uint32_t img_offset);
} sec_boot_api_interface_t;
```

详细的结构体定义见 SDK 中的示例。

19.7.7 SDP API

详细的结构体定义见 SDK 中的示例

19.7.8 EXIP API

详细的结构体定义见 SDK 中的 hpm_romapi.h

19.8 SoC 专有信息

19.8.1 OTP 映射表

OTP 字索引	位偏移	字段宽度(位)	字段	描述
1	0	4	LIFE_CYCLE_A	Life Cycle(生命周期)
	8	8	PUBK_REVOKE	Public Key REVOKE(公钥撤销) 低 4 位有效， bit_n 代表撤销第 n 个公钥
	28	4	LIFE_CYCLE_B	Duplicate Life Cycle(重复的生命周期) 实际的生命周期值为 LIFE_CYCLE_A|LIFE_CYCLE_B
3	0	32	SW_VER	软件最低版本号
24	0	4	XPI_FREQ_OPTION	XPI 频率选项 详见XPI 频率选项
	5	1	XPI_PIN_GROUP	XPI 引脚分组 0 - 第一组 1 - 第二组

OTP 字索引	位偏移	字段宽度(位)	字段	描述
	8	4	PROBE_TYPE	Serial NOR 探测类型 0 - 基于 SFDP (SDR 模式) 1 - 基于 SFDP (DDR/DTR 模式) 2 - 1-4-4 读 (命令 0xEB, 地址 24 位) 3 - 1-2-2 读 (命令 0xBB, 地址 24 位) 其它-保留
	12	1	ENCRYPT_XIP	加密原地执行使能 0 - 不启用加密原地执行 1 - 启用加密原地执行
	13	1	XPI_NOR_CFG_SRC	选择 XPI NOR 配置选项的位置 0 - 从 FLASH 偏移 0x400 处寻找 XPI NOR CFG OPTION 1 - 从 OTP 中寻找 XPI NOR CFG OPTION
	14	2	XPI_DEFAULT_READ	XPI 默认读模式 0 - 1-1-1 读，命令为 0x03,24 位地址 其他 - 保留
	16	4	BOOT_MODE	启动模式选择 0 - 启动模式由 BOOT_MODE 引脚决定 0 - 从 XPI NOR 启动 1 - 从 UART/USB-HID 启动 2 - 在系统编程模式 3 - 保留 1 - 从 XPI NOR 启动 2 - 从 UART/USB-HID 启动 其它 - 保留
	20	4	DRIVE_STRENGTH	IO 驱动强度选择 0 - BootROM 默认配置 其它-见 IOC 中的 PAD_CTL 中的 DS 字段定义
	24	7	DUMMY_CYCLE	Serial NOR FLASH DUMMY CYCLE 值 0 - DUMM CYCLE 值由 ROM 自动检测/使用 ROM 预置的值 其它 - 使用该字段定义的 DUMMY CYCLE 值
25	0	16	SEC_IMG_OFFSET	第二份启动镜像的地址偏移 0 - 第二份启动镜像并不存在 其它 - SEC_IMG_OFFSET×256KB
	16	16	MAX_IMG_LEN	启动镜像的最大长度 0 - SEC_IMG_OFFSET×256KB 其它 - MAX_IMG_LEN×256KB

OTP 字索引	位偏移	字段宽度 (位)	字段	描述
26	0	2	FORCE_COLD_BOOT	强制冷启动 0 - 支持低功耗唤醒 其它 - 强制冷启动
	2	2	FORCE_WAKEUP_ENTRY_CHK	强制检查唤醒入口的完整性校验 0 - 无需检查唤醒入口的完整性 1 - 强制检测唤醒入口的完整性 (基于 SHA256)
	4	1	HIGH_SPEED_BOOT	高速启动模式使能 0 - 普通启动模式 (CPU CORE 频率约为 325MHz) 1 - 高速启动模式 (CPU CORE 频率约为 650MHz)
	5	27	保留	-
27	0	32	保留	-
80	0	32	SRK_HASH[31:0]	SRK_HASH 在 OTP 里以小端形式存储 若 SRK_HASH 以字节的形式表示为 (b0, b1, b2, b3, ..., b30, b31), 则在 OTP 的存储依次为: 0xb3b2b1b0, 0xb7b6b5b4 ..., 0xb31b30b29b28
81	0	32	SRK_HASH[63:32]	
82	0	32	SRK_HASH[95:64]	
83	0	32	SRK_HASH[127:96]	
84	0	32	SRK_HASH[159:128]	
85	0	32	SRK_HASH[191:160]	
86	0	32	SRK_HASH[223:192]	
87	0	32	SRK_HASH[255:224]	
96	0	32	EXIP0_KEK[31:0]	KEK 在 OTP 里以大端方式存储。 若 KEK 以字节的形式表示为 (b0, b1, b2, ..., b14, b15) 则在 OTP 里的存储表示为 0xb15b14b13b12, ..., 0xb3b2b1b0
97	0	32	EXIP0_KEK[63:32]	
98	0	32	EXIP0_KEK[95:64]	
99	0	32	EXIP0_KEK[127:96]	
100	0	32	OTP_KEK0[31:0]	-
101	0	32	OTP_KEK0[63:32]	
102	0	32	OTP_KEK0[95:64]	
103	0	32	OTP_KEK0[127:96]	

表 82: BootROM 相关的 OTP 映射表

19.8.2 BootROM 支持的外设信息

19.8.2.1 XPI 信息

XPI0 引脚组

引脚分组	引脚	功能	描述
0	PA31	**XPI0.CA_CS0**	当 FLASH 设备支持 SPI 模式时，XPI 默认通过**加粗**的引脚来访问 FLASH
	PA24	XPI0.CA_CS1	
	PA27	**XPI0.CA_SCLK**	
	PA28	**XPI0.CA_D[0]**	
	PA30	**XPI0.CA_D[1]**	
	PA29	XPI0.CA_D[2]	
	PA26	XPI0.CA_D[3]	
	PA25	XPI0.CA_DQS	
1	PX02	XPI0.CA_CS0	-
	PX06	XPI0.CA_SCLK	
	PX05	XPI0.CA_D[0]	
	PX01	XPI0.CA_D[1]	
	PX00	XPI0.CA_D[2]	
	PX07	XPI0.CA_D[3]	
	PX03	XPI0.CA_DQS	

表 83: XPI0 引脚信息

注意:

- 当 XPI0 CA 这组引脚连接了不支持 DQS 引脚的 FLASH 时，CA_DQS 引脚可用于其他用途
- 第二组的 XPI 引脚只有 SiP 封装的芯片才有效

XPI 频率选项

BootROM 支持的 XPI NOR 频率如下：

- 1 - 31MHz (注：SDR 模式下，该值为 29.6MHz, DTR/DDR 模式下，该值为 28.6MHz)
- 2 - 50MHz
- 3 - 66MHz
- 4 - 80MHz
- 5 - 102MHz
- 6 - 120MHz (注：SDR 模式下，该值为 114MHz, DTR/DDR 模式下，该值为 125MHz)
- 7 - 133MHz
- 8 - 166MHz

注意: 上述频率均基于 SoC 默认的 PRESET 时钟分频而来，若对应的时钟源（如 PLL1CLK1) 的频率发生改变，上述频率会发生改变，用户需事先确认对 PLL 时钟的改变 XPI 的频率产生影响。

19.8.2.2 UART 引脚

芯片引脚	功能	功能描述
PA00	UART0.TXD	UART0 引脚
PA01	UART0.RXD	

表 84: UART0 引脚

19.8.2.3 USB 引脚

芯片引脚	功能	功能描述
USB0.DP	USB_DP	USB0 引脚
USB0.DM	USB_DM	

表 85: USB0 引脚

注意: 上述 USB 引脚仅对 LQFP100 封装有效

19.8.2.4 启动模式 (BOOT_MODE) 引脚

芯片引脚	功能	功能描述
PA02	BOOT_MODE[0]	启动模式引脚
PA03	BOOT_MODE[1]	

表 86: BOOT_MODE 引脚

19.8.3 BootROM 预占用的寄存器信息

寄存器	功能	描述/备注
SYSCTL.CPU0.GPR0	CPU0 唤醒入口	CPU0 的唤醒程序入口

SYSCTL.CPU0.PARAM0	CPU0 唤醒代码起始地址	CPU0 唤醒程序的起始地址（用于完整性检验）
SYSCTL.CPU0.PARAM1	CPU0 唤醒代码的长度	CPU0 唤醒程序的长度（用于完整性检验）
SYSCTL.CPU0_DATA0 - SYSCTL.CPU0_DATA7	CPU0 唤醒代码哈希值	在 FORCE_WAKEUP_ENTRY_CHK 被置 1 后，BootROM 需要根据 SYSCTL.CPU0_PARAM0 和 SYSCTL.CPU0_PARAM1 指定的地址范围做 SHA256 计算，并与 PMIC.GPR4-PMIC.GPR11 中的值作比较，只有当结果匹配后，才允许跳转，否则执行完整的启动流程
SYSCTL.CPU0_LOCK	上述寄存器的锁定寄存器	-
PMIC.GPR0	run_bootloader 参数	详见ROM API 简介
PMIC.GPR1	XPI NOR FLASH 状态上下文	保存 XPI NOR 运行的上下文，详细定义见
PMIC.GPR2	ROM 特定标记	-

表 87: BootROM 占用的寄存器

19.8.4 通用寄存器配置支持的寄存器范围

以下模块支持在通用寄存器信息中配置：

- SYSCTL
- IOC
- XPI0

19.8.5 BootROM 内存映射表

地址范围	功能
0x2000_0000 - 0x2000_1FFFF	BootROM 代码及数据
0x0009_7000 - 0x0009_FFFF	BSS,RW,Stack 等

表 88: BootROM 内存映射

19.8.6 BootROM 生命周期 (Lifecycle)

编码	生命周期	描述
4’b0000	CREATE	该模式下 BootROM 不控制 DEBUG 接口
4’b0001	NONSEC	该模式下 DEBUG 口受 BootROM 管控，BootROM 支持引导非签名的和签名的启动镜像
4’b0011	SECURE	该模式下 DEBUG 口受 BootROM 管控，BootROM 仅支持引导签名的启动镜像
4’b0111	RETURN	该模式下芯片的安全相关的信息被禁用，BootROM 可引导非签名的启动镜像

4’b1011	NONRET	该模式和封闭/部署模式下的表现一致，唯一不同的是 OEM 厂商在该模式下只能切到废弃模式
4’b1111	SCRIBE	作废模式，BootROM 在该模式下无法执行正常的启动操作

表 89: BootROM 生命周期编码

20 引脚配置及功能 PINMUX

20.1 IO 功能分配

本产品引脚配置及功能如下：

封装			PIN 名称	数字功能	模拟功能
LQFP_100	LQFP_64	QFN_48			
25	16	12	PA00	GPIO_A_00(ALT0) GPTMR1_COMP_0(ALT1) UART0_TXD(ALT2) LIN0_TXD(ALT6) CAN0_TXD(ALT7) PWM0_FAULT_0(ALT16) PWM1_P_0(ALT17) TRGM0_P_00(ALT18) PWM1_FAULT_0(ALT19) SYSCTL_CLK_OBS_0(ALT 24)	-
24	15	11	PA01	GPIO_A_01(ALT0) GPTMR1_CAPT_0(ALT1) UART0_RXD(ALT2) LIN0_RXD(ALT6) CAN0_RXD(ALT7) PWM0_FAULT_1(ALT16) PWM1_P_1(ALT17) TRGM0_P_01(ALT18) ACMP_COMP_0(ALT19) SYSCTL_CLK_OBS_1(ALT 24)	-

封装			PIN 名称	数字功能	模拟功能
LQFP_100	LQFP_64	QFN_48			
19	11	8	PA02	GPIO_A_02(ALT0) GPTMR1_COMP_1(ALT1) UART0_DE(ALT2) UART0_RTS(ALT3) I2C0_SCL(ALT4) LIN0_TREN(ALT6) CAN0_STBY(ALT7) ACMP_COMP_0(ALT16) PWM1_P_2(ALT17) TRGM0_P_02(ALT18) ACMP_COMP_1(ALT19) QEI1_F(ALT20) SYSCTL_CLK_OBS_2(ALT24)	-
18	10	7	PA03	GPIO_A_03(ALT0) GPTMR1_CAPT_1(ALT1) UART0_CTS(ALT3) I2C0_SDA(ALT4) SPI3_CS_3(ALT5) CAN1_STBY(ALT7) ACMP_COMP_1(ALT16) PWM1_P_3(ALT17) TRGM0_P_03(ALT18) PWM1_FAULT_1(ALT19) QEI1_H1(ALT20) SYSCTL_CLK_OBS_3(ALT24)	-
2	64	1	PA04	GPIO_A_04(ALT0) UART1_CTS(ALT3) SPI0_CS_0(ALT5) CAN1_RXD(ALT7) PWM0_P_0(ALT16) PWM1_P_4(ALT17) TRGM0_P_04(ALT18) RDC0_EXC_P(ALT19) QEI1_A(ALT20) QEO1_A(ALT21) SEI1_DE(ALT22) JTAG_TDO(ALT24)	-

封装			PIN 名称	数字功能	模拟功能
LQFP_100	LQFP_64	QFN_48			
1	63	48	PA05	GPIO_A_05(ALT0) GPTMR1_COMP_2(ALT1) UART1_DE(ALT2) UART1_RTS(ALT3) SPI0_SCLK(ALT5) LIN1_TREN(ALT6) CAN1_TXD(ALT7) PWM0_P_1(ALT16) PWM1_P_5(ALT17) TRGM0_P_05(ALT18) RDC0_EXC_N(ALT19) QEI1_B(ALT20) QEO1_B(ALT21) SEI1_CK(ALT22) JTAG_TDI(ALT24)	-
100	62	47	PA06	GPIO_A_06(ALT0) GPTMR0_CAPT_0(ALT1) UART1_RXD(ALT2) I2C1_SDA(ALT4) SPI0_MISO(ALT5) LIN1_TXD(ALT6) PWM0_P_2(ALT16) PWM1_P_6(ALT17) TRGM0_P_06(ALT18) QEI1_Z(ALT20) QEO1_Z(ALT21) SEI1_TX(ALT22) JTAG_TCK(ALT24)	-

封装			PIN 名称	数字功能	模拟功能
LQFP_100	LQFP_64	QFN_48			
99	61	46	PA07	GPIO_A_07(ALT0) GPTMR0_COMP_0(ALT1) UART1_TXD(ALT2) I2C1_SCL(ALT4) SPI0_MOSI(ALT5) LIN1_RXD(ALT6) PWM0_P_3(ALT16) PWM1_P_7(ALT17) TRGM0_P_07(ALT18) QEI1_H0(ALT20) SEI1_RX(ALT22) JTAG_TMS(ALT24)	-
3	1	2	PA08	GPIO_A_08(ALT0) GPTMR0_COMP_1(ALT1) UART2_TXD(ALT2) I2C2_SCL(ALT4) SPI3_CS_2(ALT5) CAN2_TXD(ALT7) PWM0_P_4(ALT16) PWM0_FAULT_0(ALT18) JTAG_TRST(ALT24)	-
4	2	3	PA09	GPIO_A_09(ALT0) GPTMR0_CAPT_1(ALT1) UART2_RXD(ALT2) I2C2_SDA(ALT4) SPI3_CS_1(ALT5) CAN2_RXD(ALT7) PWM0_P_5(ALT16) PWM0_FAULT_1(ALT18) SOC_REF0(ALT24)	-

封装			PIN 名称	数字功能	模拟功能
LQFP_100	LQFP_64	QFN_48			
5	3	4	PA10	GPIO_A_10(ALT0) GPTMR0_COMP_2(ALT1) UART2_DE(ALT2) UART2_RTS(ALT3) SPI3_CS_0(ALT5) LIN2_RXD(ALT6) CAN2_STBY(ALT7) PWM0_P_6(ALT16) PWM1_FAULT_0(ALT17) ACMP_COMP_0(ALT18) QEI1_A(ALT20) QEO0_A(ALT21) SEI1_DE(ALT22)	-
6	4	-	PA11	GPIO_A_11(ALT0) UART2_CTS(ALT3) SPI3_SCLK(ALT5) LIN2_TXD(ALT6) PWM0_P_7(ALT16) PWM1_FAULT_1(ALT17) ACMP_COMP_1(ALT18) QEI1_B(ALT20) QEO0_B(ALT21) SEI1_CK(ALT22) WDG0_RST(ALT24)	-
11	8	-	PA12	GPIO_A_12(ALT0) UART3_CTS(ALT3) I2C3_SDA(ALT4) SPI3_MISO(ALT5) LIN2_TREN(ALT6) PWM0_P_0(ALT16) PWM1_FAULT_0(ALT17) PWM0_FAULT_0(ALT18) RDC0_EXC_P(ALT19) QEI1_Z(ALT20) QEO0_Z(ALT21) SEI1_TX(ALT22) SYSCTL_CLK_OBS_3(ALT24)	-

封装			PIN 名称	数字功能	模拟功能
LQFP_100	LQFP_64	QFN_48			
12	9	-	PA13	GPIO_A_13(ALT0) GPTMR1_COMP_3(ALT1) UART3_DE(ALT2) UART3_RTS(ALT3) I2C3_SCL(ALT4) SPI3_MOSI(ALT5) LIN3_TREN(ALT6) CAN3_STBY(ALT7) PWM0_P_1(ALT16) PWM1_FAULT_1(ALT17) PWM0_FAULT_1(ALT18) RDC0_EXC_N(ALT19) QEI1_H0(ALT20) SEI1_RX(ALT22) SYSCTL_CLK_OBS_2(ALT24)	-
13	-	-	PA14	GPIO_A_14(ALT0) UART3_RXD(ALT2) SPI3_DAT2(ALT5) LIN3_RXD(ALT6) CAN3_RXD(ALT7) PWM0_P_2(ALT16) ACMP_COMP_0(ALT18) QEI1_H1(ALT20) WDG1_RST(ALT24)	-
14	-	-	PA15	GPIO_A_15(ALT0) GPTMR0_COMP_3(ALT1) UART3_TXD(ALT2) SPI3_DAT3(ALT5) LIN3_TXD(ALT6) CAN3_TXD(ALT7) PWM0_P_3(ALT16) ACMP_COMP_1(ALT18) QEI1_F(ALT20) SOC_REF0(ALT24)	-

封装			PIN 名称	数字功能	模拟功能
LQFP_100	LQFP_64	QFN_48			
98	-	-	PA16	GPIO_A_16(ALT0) GPTMR3_COMP_0(ALT1) UART4_TXD(ALT2) LIN0_TXD(ALT6) CAN0_TXD(ALT7) PWM0_P_4(ALT16) PWM1_P_0(ALT17) TRGM0_P_04(ALT18) QEO0_A(ALT21) SEI1_DE(ALT22)	-
97	-	-	PA17	GPIO_A_17(ALT0) GPTMR3_CAPT_0(ALT1) UART4_RXD(ALT2) LIN0_RXD(ALT6) CAN0_RXD(ALT7) PWM0_P_5(ALT16) PWM1_P_1(ALT17) TRGM0_P_05(ALT18) QEO0_B(ALT21) SEI1_CK(ALT22)	-
93	-	-	PA18	GPIO_A_18(ALT0) GPTMR3_COMP_1(ALT1) UART4_DE(ALT2) UART4_RTS(ALT3) I2C0_SCL(ALT4) LIN0_TREN(ALT6) CAN0_STBY(ALT7) PWM0_P_6(ALT16) PWM1_P_2(ALT17) TRGM0_P_06(ALT18) QEO0_Z(ALT21) SEI1_TX(ALT22)	-

封装			PIN 名称	数字功能	模拟功能
LQFP_100	LQFP_64	QFN_48			
92	-	-	PA19	GPIO_A_19(ALT0) GPTMR3_CAPT_1(ALT1) UART4_CTS(ALT3) I2C0_SDA(ALT4) SPI1_CS_3(ALT5) CAN1_STBY(ALT7) PWM0_P_7(ALT16) PWM1_P_3(ALT17) TRGM0_P_07(ALT18) SEI1_RX(ALT22)	-
91	-	-	PA20	GPIO_A_20(ALT0) UART5_CTS(ALT3) SPI2_CS_0(ALT5) CAN1_RXD(ALT7) PWM0_FAULT_0(ALT16) PWM1_P_4(ALT17) TRGM0_P_00(ALT18) QEI0_A(ALT20) QEO0_A(ALT21) SEI0_DE(ALT22)	-
90	-	-	PA21	GPIO_A_21(ALT0) GPTMR3_COMP_2(ALT1) UART5_DE(ALT2) UART5_RTS(ALT3) SPI2_SCLK(ALT5) LIN1_TREN(ALT6) CAN1_TXD(ALT7) PWM0_FAULT_1(ALT16) PWM1_P_5(ALT17) TRGM0_P_01(ALT18) QEI0_B(ALT20) QEO0_B(ALT21) SEI0_CK(ALT22)	-

封装			PIN 名称	数字功能	模拟功能
LQFP_100	LQFP_64	QFN_48			
89	-	-	PA22	GPIO_A_22(ALT0) GPTMR2_CAPT_0(ALT1) UART5_RXD(ALT2) I2C1_SDA(ALT4) SPI2_MISO(ALT5) LIN1_TXD(ALT6) PWM1_P_6(ALT17) TRGM0_P_02(ALT18) PWM1_FAULT_0(ALT19) QEI0_Z(ALT20) QEO0_Z(ALT21) SEI0_TX(ALT22)	-
88	-	-	PA23	GPIO_A_23(ALT0) GPTMR2_COMP_0(ALT1) UART5_TXD(ALT2) I2C1_SCL(ALT4) SPI2_MOSI(ALT5) LIN1_RXD(ALT6) PWM1_P_7(ALT17) TRGM0_P_03(ALT18) PWM1_FAULT_1(ALT19) QEI0_H0(ALT20) SEI0_RX(ALT22)	-
87	57	44	PA24	GPIO_A_24(ALT0) GPTMR2_COMP_1(ALT1) UART6_TXD(ALT2) I2C2_SCL(ALT4) SPI1_CS_2(ALT5) CAN2_TXD(ALT7) XPI0_CA_CS1(ALT14) PWM0_P_0(ALT16) PWM1_P_0(ALT17) TRGM0_P_00(ALT18) QEI0_H1(ALT20)	-

封装			PIN 名称	数字功能	模拟功能
LQFP_100	LQFP_64	QFN_48			
86	56	43	PA25	GPIO_A_25(ALT0) GPTMR2_CAPT_1(ALT1) UART6_RXD(ALT2) I2C2_SDA(ALT4) SPI1_CS_1(ALT5) CAN2_RXD(ALT7) XPI0_CA_DQS(ALT14) PWM0_P_1(ALT16) PWM1_P_1(ALT17) TRGM0_P_01(ALT18) QEI0_F(ALT20)	-
82	55	42	PA26	GPIO_A_26(ALT0) GPTMR2_COMP_2(ALT1) UART6_DE(ALT2) UART6_RTS(ALT3) SPI1_CS_0(ALT5) LIN2_RXD(ALT6) CAN2_STBY(ALT7) XPI0_CA_D_3(ALT14) PWM0_P_2(ALT16) PWM1_P_2(ALT17) TRGM0_P_02(ALT18) QEI0_A(ALT20) QEO0_A(ALT21) SEI0_DE(ALT22) SYSCTL_CLK_OBS_0(ALT24)	-

封装			PIN 名称	数字功能	模拟功能
LQFP_100	LQFP_64	QFN_48			
81	54	41	PA27	GPIO_A_27(ALT0) UART6_CTS(ALT3) SPI1_SCLK(ALT5) LIN2_TXD(ALT6) XPI0_CA_SCLK(ALT14) PWM0_P_3(ALT16) PWM1_P_3(ALT17) TRGM0_P_03(ALT18) QEI0_B(ALT20) QEO0_B(ALT21) SEI0_CK(ALT22) SYSCTL_CLK_OBS_1(ALT24)	-
80	53	40	PA28	GPIO_A_28(ALT0) UART7_CTS(ALT3) I2C3_SDA(ALT4) SPI1_MISO(ALT5) LIN2_TREN(ALT6) XPI0_CA_D_0(ALT14) PWM0_P_4(ALT16) PWM1_P_4(ALT17) TRGM0_P_04(ALT18) RDC0_EXC_P(ALT19) QEI0_Z(ALT20) QEO0_Z(ALT21) SEI0_TX(ALT22) SYSCTL_CLK_OBS_2(ALT24)	-

封装			PIN 名称	数字功能	模拟功能
LQFP_100	LQFP_64	QFN_48			
79	52	39	PA29	GPIO_A_29(ALT0) GPTMR3_COMP_3(ALT1) UART7_DE(ALT2) UART7_RTS(ALT3) I2C3_SCL(ALT4) SPI1_MOSI(ALT5) LIN3_TREN(ALT6) CAN3_STBY(ALT7) XPI0_CA_D_2(ALT14) PWM0_P_5(ALT16) PWM1_P_5(ALT17) TRGM0_P_05(ALT18) RDC0_EXC_N(ALT19) QEI0_H0(ALT20) SEI0_RX(ALT22) SYSCTL_CLK_OBS_3(ALT24) USB0_OC(ALT25)	-
78	51	38	PA30	GPIO_A_30(ALT0) UART7_RXD(ALT2) SPI1_DAT2(ALT5) LIN3_RXD(ALT6) CAN3_RXD(ALT7) XPI0_CA_D_1(ALT14) PWM0_P_6(ALT16) PWM1_P_6(ALT17) TRGM0_P_06(ALT18) QEI0_H1(ALT20) SOC_REF0(ALT24) USB0_PWR(ALT25)	-

封装			PIN 名称	数字功能	模拟功能
LQFP_100	LQFP_64	QFN_48			
77	50	37	PA31	GPIO_A_31(ALT0) GPTMR2_COMP_3(ALT1) UART7_TXD(ALT2) SPI1_DAT3(ALT5) LIN3_TXD(ALT6) CAN3_TXD(ALT7) XPI0_CA_CS0(ALT14) PWM0_P_7(ALT16) PWM1_P_7(ALT17) TRGM0_P_07(ALT18) QEI0_F(ALT20) USB0_ID(ALT25)	-
63	41	-	PB00	GPIO_B_00(ALT0) GPTMR1_COMP_0(ALT1) UART0_TXD(ALT2) LIN0_TXD(ALT6) CAN0_TXD(ALT7) PWM0_P_0(ALT16) PWM1_FAULT_0(ALT17) TRGM0_P_04(ALT18) ACMP_COMP_0(ALT19)	ADC0_IN15 ADC1_IN15 ACMP_CMP0_INN7 OPA0_OUT
62	40	-	PB01	GPIO_B_01(ALT0) GPTMR1_CAPT_0(ALT1) UART0_RXD(ALT2) LIN0_RXD(ALT6) CAN0_RXD(ALT7) PWM0_P_1(ALT16) PWM1_FAULT_1(ALT17) TRGM0_P_05(ALT18) ACMP_COMP_1(ALT19)	ADC0_IN14 ADC1_IN14 ACMP_CMP1_INN7 OPA1_OUT

封装			PIN 名称	数字功能	模拟功能
LQFP_100	LQFP_64	QFN_48			
61	-	-	PB02	GPIO_B_02(ALT0) GPTMR1_COMP_1(ALT1) UART0_DE(ALT2) UART0_RTS(ALT3) I2C0_SCL(ALT4) LIN0_TREN(ALT6) CAN0_STBY(ALT7) PWM0_P_2(ALT16) ACMP_COMP_1(ALT17) TRGM0_P_06(ALT18) PWM0_FAULT_0(ALT19)	ADC0_IN12 ADC1_IN12 ACMP_CMP0_INP7 OPA0_EXT
60	-	-	PB03	GPIO_B_03(ALT0) GPTMR1_CAPT_1(ALT1) UART0_CTS(ALT3) I2C0_SDA(ALT4) SPI2_CS_3(ALT5) CAN1_STBY(ALT7) PWM0_P_3(ALT16) ACMP_COMP_0(ALT17) TRGM0_P_07(ALT18) PWM0_FAULT_1(ALT19)	ADC0_IN8 ADC1_IN8 ACMP_CMP1_INP7 OPA1_EXT
59	-	-	PB04	GPIO_B_04(ALT0) UART1_CTS(ALT3) SPI3_CS_0(ALT5) CAN1_RXD(ALT7) PWM0_P_4(ALT16) PWM1_P_0(ALT17) TRGM0_P_00(ALT18) QEI1_A(ALT20) QEO1_A(ALT21) SEI0_DE(ALT22)	ADC0_IN0 ADC1_IN0 ACMP_CMP0_INN5 ACMP_CMP1_INN5 OPA0_INP0

封装			PIN 名称	数字功能	模拟功能
LQFP_100	LQFP_64	QFN_48			
58	-	-	PB05	GPIO_B_05(ALT0) GPTMR1_COMP_2(ALT1) UART1_DE(ALT2) UART1_RTS(ALT3) SPI3_SCLK(ALT5) LIN1_TREN(ALT6) CAN1_TXD(ALT7) PWM0_P_5(ALT16) PWM1_P_1(ALT17) TRGM0_P_01(ALT18) QEI1_B(ALT20) QEO1_B(ALT21) SEI0_CK(ALT22)	ADC0_IN13 ADC1_IN13 ACMP_CMP0_INN3 ACMP_CMP1_INN3 OPA0_INN0
54	-	-	PB06	GPIO_B_06(ALT0) GPTMR0_CAPT_0(ALT1) UART1_RXD(ALT2) I2C1_SDA(ALT4) SPI3_MISO(ALT5) LIN1_TXD(ALT6) PWM0_P_6(ALT16) PWM1_P_2(ALT17) TRGM0_P_02(ALT18) RDC0_EXC_P(ALT19) QEI1_Z(ALT20) QEO1_Z(ALT21) SEI0_TX(ALT22)	ADC0_IN9 ADC1_IN9 ACMP_CMP0_INP5 ACMP_CMP1_INP5 OPA0_INP1
53	-	-	PB07	GPIO_B_07(ALT0) GPTMR0_COMP_0(ALT1) UART1_TXD(ALT2) I2C1_SCL(ALT4) SPI3_MOSI(ALT5) LIN1_RXD(ALT6) PWM0_P_7(ALT16) PWM1_P_3(ALT17) TRGM0_P_03(ALT18) RDC0_EXC_N(ALT19) QEI1_H0(ALT20) SEI0_RX(ALT22)	ADC0_IN10 ADC1_IN10 ACMP_CMP0_INP3 ACMP_CMP1_INP3 OPA0_INN1

封装			PIN 名称	数字功能	模拟功能
LQFP_100	LQFP_64	QFN_48			
52	36	27	PB08	GPIO_B_08(ALT0) GPTMR0_COMP_1(ALT1) UART2_TXD(ALT2) I2C2_SCL(ALT4) SPI2_CS_2(ALT5) CAN2_TXD(ALT7) ACMP_COMP_0(ALT16) PWM1_P_4(ALT17) QEI1_H1(ALT20) QEO1_A(ALT21) SEI1_DE(ALT22) USB0_ID(ALT25)	ADC0_IN11 ADC1_IN11 DAC0_OUT ACMP_CMP0_INN6 ACMP_CMP1_INN6 OPA0_INP2 OPA1_INP2
51	35	26	PB09	GPIO_B_09(ALT0) GPTMR0_CAPT_1(ALT1) UART2_RXD(ALT2) I2C2_SDA(ALT4) SPI2_CS_1(ALT5) CAN2_RXD(ALT7) ACMP_COMP_1(ALT16) PWM1_P_5(ALT17) QEI1_F(ALT20) QEO1_B(ALT21) SEI1_CK(ALT22) USB0_OC(ALT25)	ADC0_IN1 ADC1_IN1 DAC1_OUT ACMP_CMP0_INP6 ACMP_CMP1_INP6 OPA0_INN2 OPA1_INN2
50	34	25	PB10	GPIO_B_10(ALT0) GPTMR0_COMP_2(ALT1) UART2_DE(ALT2) UART2_RTS(ALT3) SPI2_CS_0(ALT5) LIN2_RXD(ALT6) CAN2_STBY(ALT7) ACMP_COMP_0(ALT16) PWM1_P_6(ALT17) QEI0_H1(ALT20) QEO1_Z(ALT21) SEI1_TX(ALT22) USB0_PWR(ALT25)	ADC0_IN2 ADC1_IN2 ACMP_CMP0_INP4 ACMP_CMP1_INP4 OPA0_INP3 OPA1_INP3

封装			PIN 名称	数字功能	模拟功能
LQFP_100	LQFP_64	QFN_48			
49	33	24	PB11	GPIO_B_11(ALT0) UART2_CTS(ALT3) SPI2_SCLK(ALT5) LIN2_TXD(ALT6) ACMP_COMP_1(ALT16) PWM1_P_7(ALT17) QEI0_F(ALT20) SEI1_RX(ALT22)	ADC0_IN3 ADC1_IN3 ACMP_CMP0_INN4 ACMP_CMP1_INN4 OPA0_INN3 OPA1_INN3
48	32	23	PB12	GPIO_B_12(ALT0) UART3_CTS(ALT3) I2C3_SDA(ALT4) SPI2_MISO(ALT5) LIN2_TREN(ALT6) PWM1_FAULT_0(ALT16) PWM1_P_0(ALT17) TRGM0_P_00(ALT18) QEI0_A(ALT20) QEO1_A(ALT21) SEI0_DE(ALT22)	ADC0_IN4 ADC1_IN4 ACMP_CMP0_INN2 ACMP_CMP1_INN2 OPA1_INP0
47	31	22	PB13	GPIO_B_13(ALT0) GPTMR1_COMP_3(ALT1) UART3_DE(ALT2) UART3_RTS(ALT3) I2C3_SCL(ALT4) SPI2_MOSI(ALT5) LIN3_TREN(ALT6) CAN3_STBY(ALT7) PWM1_FAULT_1(ALT16) PWM1_P_1(ALT17) TRGM0_P_01(ALT18) QEI0_B(ALT20) QEO1_B(ALT21) SEI0_CK(ALT22)	ADC0_IN5 ADC1_IN5 ACMP_CMP0_INP2 ACMP_CMP1_INP2 OPA1_INN0

封装			PIN 名称	数字功能	模拟功能
LQFP_100	LQFP_64	QFN_48			
46	30	21	PB14	GPIO_B_14(ALT0) UART3_RXD(ALT2) SPI2_DAT2(ALT5) LIN3_RXD(ALT6) CAN3_RXD(ALT7) PWM0_FAULT_0(ALT16) PWM1_P_2(ALT17) TRGM0_P_02(ALT18) RDC0_EXC_P(ALT19) QEI0_Z(ALT20) QEO1_Z(ALT21) SEI0_TX(ALT22)	ADC0_IN6 ADC1_IN6 ACMP_CMP0_INN1 ACMP_CMP1_INN1 OPA1_INP1
45	29	20	PB15	GPIO_B_15(ALT0) GPTMR0_COMP_3(ALT1) UART3_TXD(ALT2) SPI2_DAT3(ALT5) LIN3_TXD(ALT6) CAN3_TXD(ALT7) PWM0_FAULT_1(ALT16) PWM1_P_3(ALT17) TRGM0_P_03(ALT18) RDC0_EXC_N(ALT19) QEI0_H0(ALT20) SEI0_RX(ALT22)	ADC0_IN7 ADC1_IN7 ACMP_CMP0_INP1 ACMP_CMP1_INP1 OPA1_INN1
41	28	19	PY00	GPIO_Y_00(ALT0) GPTMR3_COMP_0(ALT1) UART0_TXD(ALT2) LIN2_TXD(ALT6) CAN2_TXD(ALT7) PWM0_P_0(ALT16) PWM1_P_4(ALT17) PWM0_FAULT_0(ALT18) USB0_ID(ALT25)	-

封装			PIN 名称	数字功能	模拟功能
LQFP_100	LQFP_64	QFN_48			
40	27	18	PY01	GPIO_Y_01(ALT0) GPTMR3_CAPT_0(ALT1) UART0_RXD(ALT2) LIN2_RXD(ALT6) CAN2_RXD(ALT7) PWM0_P_1(ALT16) PWM1_P_5(ALT17) PWM0_FAULT_1(ALT18) WDG0_RST(ALT24) USB0_OC(ALT25)	-
39	26	-	PY02	GPIO_Y_02(ALT0) GPTMR3_COMP_1(ALT1) UART0_DE(ALT2) UART0_RTS(ALT3) I2C2_SCL(ALT4) LIN2_TREN(ALT6) CAN2_STBY(ALT7) PWM0_P_2(ALT16) PWM1_P_6(ALT17) ACMP_COMP_0(ALT18) PWM1_FAULT_0(ALT19) WDG1_RST(ALT24) USB0_PWR(ALT25)	-
38	25	-	PY03	GPIO_Y_03(ALT0) GPTMR3_CAPT_1(ALT1) UART0_CTS(ALT3) I2C2_SDA(ALT4) CAN3_STBY(ALT7) PWM0_P_3(ALT16) PWM1_P_7(ALT17) ACMP_COMP_1(ALT18) PWM1_FAULT_1(ALT19)	-
37	-	-	PY04	GPIO_Y_04(ALT0) UART1_CTS(ALT3) SPI2_CS_0(ALT5) CAN3_RXD(ALT7) PWM0_P_4(ALT16) TRGM0_P_04(ALT18)	-

封装			PIN 名称	数字功能	模拟功能
LQFP_100	LQFP_64	QFN_48			
36	-	-	PY05	GPIO_Y_05(ALT0) GPTMR3_COMP_2(ALT1) UART1_DE(ALT2) UART1_RTS(ALT3) SPI2_SCLK(ALT5) LIN3_TREN(ALT6) CAN3_TXD(ALT7) PWM0_P_5(ALT16) TRGM0_P_05(ALT18) WDG0_RST(ALT24)	-
35	-	-	PY06	GPIO_Y_06(ALT0) GPTMR2_CAPT_0(ALT1) UART1_RXD(ALT2) I2C3_SDA(ALT4) SPI2_MISO(ALT5) LIN3_TXD(ALT6) PWM0_P_6(ALT16) TRGM0_P_06(ALT18) WDG1_RST(ALT24)	-
34	-	-	PY07	GPIO_Y_07(ALT0) GPTMR2_COMP_0(ALT1) UART1_TXD(ALT2) I2C3_SCL(ALT4) SPI2_MOSI(ALT5) LIN3_RXD(ALT6) PWM0_P_7(ALT16) TRGM0_P_07(ALT18)	-
74	48	35	XTALI	-	-
75	49	36	XTALO	-	-
73	-	-	USB_DM	-	-
72	-	-	USB_DP	-	-
26	17	13	RESETN	-	-
27	18	14	WAKEUP	-	-
68	-	-	USBVBUS	-	-
30	21	-	DCDC_IN	-	-
28	19	-	DCDC_GND	-	-
29	20	15	DCDC_LP	-	-
21	-	-	DCDC_SNS	-	-

封装			PIN 名称	数字功能	模拟功能
LQFP_100	LQFP_64	QFN_48			
7,17,20,44,57,69,83,94	5,12,39,45,58	5,9,28,33,45	VDD_SOC	-	-
31	22	16	VPMC	-	-
23	14	10	VDD_OTPCAP	-	-
33	24	17	VDD_PMCCAP	-	-
71	-	-	VPLL	-	-
67	44	32	VANA	-	-
9,15,85,96	7,47,60	6,34	VIO_B00	-	-
42,55	37	29	VIO_B01	-	-
65	43	31	VREFH	-	-
64	42	30	VREFL	-	-
8,10,16,22,32,43,56,66,70,76,84,95	6,13,23,38,46,59	-	VSS	-	-

表 90: SOC IOMUX

20.2 电源管理域 IO 功能分配

本产品电源管理域 IO 引脚配置及功能如下：

封装			PIN 名称	数字功能
LQFP_100	LQFP_64	QFN_48		
41	28	19	PY00	PGPIO_Y_00(ALT0) PUART_TXD(ALT1) PTMR_COMP_0(ALT2) SOC_GPIO_Y_00(ALT3)
40	27	18	PY01	PGPIO_Y_01(ALT0) PUART_RXD(ALT1) PTMR_COMP_1(ALT2) SOC_GPIO_Y_01(ALT3)

封装			PIN 名称	数字功能
LQFP _100	LQFP _64	QFN_ 48		
39	26	-	PY02	PGPIO_Y_02(ALT0) PUART_RTS(ALT1) PTMR_COMP_2(ALT2) SOC_GPIO_Y_02(ALT3)
38	25	-	PY03	PGPIO_Y_03(ALT0) PUART_CTS(ALT1) PTMR_COMP_3(ALT2) SOC_GPIO_Y_03(ALT3)
37	-	-	PY04	PGPIO_Y_04(ALT0) PTMR_COMP_0(ALT2) SOC_GPIO_Y_04(ALT3)
36	-	-	PY05	PGPIO_Y_05(ALT0) PWDG_RST(ALT1) PTMR_CAPT_0(ALT2) SOC_GPIO_Y_05(ALT3)
35	-	-	PY06	PGPIO_Y_06(ALT0) PTMR_COMP_1(ALT2) SOC_GPIO_Y_06(ALT3)
34	-	-	PY07	PGPIO_Y_07(ALT0) PTMR_CAPT_1(ALT2) SOC_GPIO_Y_07(ALT3)

表 91: PMIC IOMUX

20.3 系统电源域外设管脚分配

本产品系统电源域的外设，不同模块的引脚分配总结如下：

ACMP 信号名称	引脚
CMP0_INN1	PB14
CMP0_INN2	PB12
CMP0_INN3	PB05
CMP0_INN4	PB11
CMP0_INN5	PB04
CMP0_INN6	PB08
CMP0_INN7	PB00
CMP0_INP1	PB15
CMP0_INP2	PB13
CMP0_INP3	PB07
CMP0_INP4	PB10

ACMP 信号名称	引脚
CMP0_INP5	PB06
CMP0_INP6	PB09
CMP0_INP7	PB02
CMP1_INN1	PB14
CMP1_INN2	PB12
CMP1_INN3	PB05
CMP1_INN4	PB11
CMP1_INN5	PB04
CMP1_INN6	PB08
CMP1_INN7	PB01
CMP1_INP1	PB15
CMP1_INP2	PB13
CMP1_INP3	PB07
CMP1_INP4	PB10
CMP1_INP5	PB06
CMP1_INP6	PB09
CMP1_INP7	PB03
COMP_0	PA01 PA02 PA10 PA14 PB00 PB03 PB08 PB10 PY02
COMP_1	PA02 PA03 PA11 PA15 PB01 PB02 PB09 PB11 PY03

表 92: ACMP 信号引脚

ADC0 信号名称	引脚
IN0	PB04
IN1	PB09

ADC0 信号名称	引脚
IN2	PB10
IN3	PB11
IN4	PB12
IN5	PB13
IN6	PB14
IN7	PB15
IN8	PB03
IN9	PB06
IN10	PB07
IN11	PB08
IN12	PB02
IN13	PB05
IN14	PB01
IN15	PB00

表 93: ADC0 信号引脚

ADC1 信号名称	引脚
IN0	PB04
IN1	PB09
IN2	PB10
IN3	PB11
IN4	PB12
IN5	PB13
IN6	PB14
IN7	PB15
IN8	PB03
IN9	PB06
IN10	PB07
IN11	PB08
IN12	PB02
IN13	PB05
IN14	PB01
IN15	PB00

表 94: ADC1 信号引脚

CAN0 信号名称	引脚
RXD	PA01 PA17 PB01 PX01

CAN0 信号名称	引脚
STBY	PA02 PA18 PB02 PX02
TXD	PA00 PA16 PB00 PX00

表 95: CAN0 信号引脚

CAN1 信号名称	引脚
RXD	PA04 PA20 PB04 PX04
STBY	PA03 PA19 PB03 PX03
TXD	PA05 PA21 PB05 PX05

表 96: CAN1 信号引脚

CAN2 信号名称	引脚
RXD	PA09 PA25 PB09 PY01
STBY	PA10 PA26 PB10 PY02
TXD	PA08 PA24 PB08 PY00

表 97: CAN2 信号引脚

CAN3 信号名称	引脚
RXD	PA14 PA30 PB14 PY04
STBY	PA13 PA29 PB13 PY03
TXD	PA15 PA31 PB15 PY05

表 98: CAN3 信号引脚

DAC0 信号名称	引脚
OUT	PB08

表 99: DAC0 信号引脚

DAC1 信号名称	引脚
OUT	PB09

表 100: DAC1 信号引脚

GPIO 信号名称	引脚
A_00	PA00
A_01	PA01
A_02	PA02
A_03	PA03
A_04	PA04
A_05	PA05
A_06	PA06
A_07	PA07
A_08	PA08
A_09	PA09
A_10	PA10
A_11	PA11
A_12	PA12
A_13	PA13
A_14	PA14
A_15	PA15

GPIO 信号名称	引脚
A_16	PA16
A_17	PA17
A_18	PA18
A_19	PA19
A_20	PA20
A_21	PA21
A_22	PA22
A_23	PA23
A_24	PA24
A_25	PA25
A_26	PA26
A_27	PA27
A_28	PA28
A_29	PA29
A_30	PA30
A_31	PA31
B_00	PB00
B_01	PB01
B_02	PB02
B_03	PB03
B_04	PB04
B_05	PB05
B_06	PB06
B_07	PB07
B_08	PB08
B_09	PB09
B_10	PB10
B_11	PB11
B_12	PB12
B_13	PB13
B_14	PB14
B_15	PB15
X_00	PX00
X_01	PX01
X_02	PX02
X_03	PX03
X_04	PX04
X_05	PX05
X_06	PX06
X_07	PX07

GPIO 信号名称	引脚
Y_00	PY00
Y_01	PY01
Y_02	PY02
Y_03	PY03
Y_04	PY04
Y_05	PY05
Y_06	PY06
Y_07	PY07

表 101: GPIO 信号引脚

GPTMR0 信号名称	引脚
CAPT_0	PA06 PB06
CAPT_1	PA09 PB09
COMP_0	PA07 PB07
COMP_1	PA08 PB08
COMP_2	PA10 PB10
COMP_3	PA15 PB15

表 102: GPTMR0 信号引脚

GPTMR1 信号名称	引脚
CAPT_0	PA01 PB01
CAPT_1	PA03 PB03
COMP_0	PA00 PB00
COMP_1	PA02 PB02
COMP_2	PA05 PB05
COMP_3	PA13 PB13

表 103: GPTMR1 信号引脚

GPTMR2 信号名称	引脚
CAPT_0	PA22 PX01 PY06
CAPT_1	PA25 PX03
COMP_0	PA23 PX00 PY07
COMP_1	PA24 PX02
COMP_2	PA26 PX05
COMP_3	PA31

表 104: GPTMR2 信号引脚

GPTMR3 信号名称	引脚
CAPT_0	PA17 PX06 PY01
CAPT_1	PA19 PY03
COMP_0	PA16 PX07 PY00
COMP_1	PA18 PY02
COMP_2	PA21 PY05
COMP_3	PA29

表 105: GPTMR3 信号引脚

I2C0 信号名称	引脚
SCL	PA02 PA18 PB02 PX02
SDA	PA03 PA19 PB03 PX03

I2C0 信号名称	引脚

表 106: I2C0 信号引脚

I2C1 信号名称	引脚
SCL	PA07 PA23 PB07 PX07
SDA	PA06 PA22 PB06 PX06

表 107: I2C1 信号引脚

I2C2 信号名称	引脚
SCL	PA08 PA24 PB08 PY02
SDA	PA09 PA25 PB09 PY03

表 108: I2C2 信号引脚

I2C3 信号名称	引脚
SCL	PA13 PA29 PB13 PY07
SDA	PA12 PA28 PB12 PY06

表 109: I2C3 信号引脚

JTAG 信号名称	引脚
TCK	PA06
TDI	PA05
TDO	PA04

JTAG 信号名称	引脚
TMS	PA07
TRST	PA08

表 110: JTAG 信号引脚

LIN0 信号名称	引脚
RXD	PA01 PA17 PB01 PX01
TREN	PA02 PA18 PB02 PX02
TXD	PA00 PA16 PB00 PX00

表 111: LIN0 信号引脚

LIN1 信号名称	引脚
RXD	PA07 PA23 PB07 PX07
TREN	PA05 PA21 PB05 PX05
TXD	PA06 PA22 PB06 PX06

表 112: LIN1 信号引脚

LIN2 信号名称	引脚
RXD	PA10 PA26 PB10 PY01
TREN	PA12

LIN2 信号名称	引脚
	PA28 PB12 PY02
TXD	PA11 PA27 PB11 PY00

表 113: LIN2 信号引脚

LIN3 信号名称	引脚
RXD	PA14 PA30 PB14 PY07
TREN	PA13 PA29 PB13 PY05
TXD	PA15 PA31 PB15 PY06

表 114: LIN3 信号引脚

OPA0 信号名称	引脚
EXT	PB02
INN0	PB05
INN1	PB07
INN2	PB09
INN3	PB11
INP0	PB04
INP1	PB06
INP2	PB08
INP3	PB10
OUT	PB00

表 115: OPA0 信号引脚

OPA1 信号名称	引脚
EXT	PB03

OPA1 信号名称	引脚
INN0	PB13
INN1	PB15
INN2	PB09
INN3	PB11
INP0	PB12
INP1	PB14
INP2	PB08
INP3	PB10
OUT	PB01

表 116: OPA1 信号引脚

PWM0 信号名称	引脚
FAULT_0	PA00 PA08 PA12 PA20 PB02 PB14 PY00
FAULT_1	PA01 PA09 PA13 PA21 PB03 PB15 PY01
P_0	PA04 PA12 PA24 PB00 PY00
P_1	PA05 PA13 PA25 PB01 PY01
P_2	PA06 PA14 PA26 PB02 PY02

PWM0 信号名称	引脚
P_3	PA07 PA15 PA27 PB03 PY03
P_4	PA08 PA16 PA28 PB04 PY04
P_5	PA09 PA17 PA29 PB05 PY05
P_6	PA10 PA18 PA30 PB06 PY06
P_7	PA11 PA19 PA31 PB07 PY07

表 117: PWM0 信号引脚

PWM1 信号名称	引脚
FAULT_0	PA00 PA10 PA12 PA22 PB00 PB12 PY02
FAULT_1	PA03 PA11 PA13 PA23 PB01 PB13

PWM1 信号名称	引脚
	PY03
P_0	PA00 PA16 PA24 PB04 PB12
P_1	PA01 PA17 PA25 PB05 PB13
P_2	PA02 PA18 PA26 PB06 PB14
P_3	PA03 PA19 PA27 PB07 PB15
P_4	PA04 PA20 PA28 PB08 PY00
P_5	PA05 PA21 PA29 PB09 PY01
P_6	PA06 PA22 PA30 PB10 PY02
P_7	PA07 PA23 PA31 PB11 PY03

PWM1 信号名称	引脚

表 118: PWM1 信号引脚

QEI0 信号名称	引脚
A	PA20 PA26 PB12
B	PA21 PA27 PB13
F	PA25 PA31 PB11
H0	PA23 PA29 PB15
H1	PA24 PA30 PB10
Z	PA22 PA28 PB14

表 119: QEI0 信号引脚

QEI1 信号名称	引脚
A	PA04 PA10 PB04
B	PA05 PA11 PB05
F	PA02 PA15 PB09
H0	PA07 PA13 PB07
H1	PA03 PA14 PB08
Z	PA06

QEI1 信号名称	引脚
	PA12 PB06

表 120: QEI1 信号引脚

QEO0 信号名称	引脚
A	PA10 PA16 PA20 PA26
B	PA11 PA17 PA21 PA27
Z	PA12 PA18 PA22 PA28

表 121: QEO0 信号引脚

QEO1 信号名称	引脚
A	PA04 PB04 PB08 PB12
B	PA05 PB05 PB09 PB13
Z	PA06 PB06 PB10 PB14

表 122: QEO1 信号引脚

RDC0 信号名称	引脚
EXC_N	PA05 PA13 PA29 PB07 PB15

RDC0 信号名称	引脚
EXC_P	PA04 PA12 PA28 PB06 PB14

表 123: RDC0 信号引脚

SEI0 信号名称	引脚
CK	PA21 PA27 PB05 PB13
DE	PA20 PA26 PB04 PB12
RX	PA23 PA29 PB07 PB15
TX	PA22 PA28 PB06 PB14

表 124: SEI0 信号引脚

SEI1 信号名称	引脚
CK	PA05 PA11 PA17 PB09
DE	PA04 PA10 PA16 PB08
RX	PA07 PA13 PA19 PB11
TX	PA06

SEI1 信号名称	引脚
	PA12 PA18 PB10

表 125: SEI1 信号引脚

SOC 信号名称	引脚
REF0	PA09 PA15 PA30

表 126: SOC 信号引脚

SPI0 信号名称	引脚
CS_0	PA04
MISO	PA06
MOSI	PA07
SCLK	PA05

表 127: SPI0 信号引脚

SPI1 信号名称	引脚
CS_0	PA26 PX04
CS_1	PA25
CS_2	PA24
CS_3	PA19
DAT2	PA30
DAT3	PA31
MISO	PA28 PX06
MOSI	PA29 PX07
SCLK	PA27 PX05

表 128: SPI1 信号引脚

SPI2 信号名称	引脚
CS_0	PA20 PB10 PY04
CS_1	PB09

SPI2 信号名称	引脚
CS_2	PB08
CS_3	PB03
DAT2	PB14
DAT3	PB15
MISO	PA22 PB12 PY06
MOSI	PA23 PB13 PY07
SCLK	PA21 PB11 PY05

表 129: SPI2 信号引脚

SPI3 信号名称	引脚
CS_0	PA10 PB04
CS_1	PA09
CS_2	PA08
CS_3	PA03
DAT2	PA14
DAT3	PA15
MISO	PA12 PB06
MOSI	PA13 PB07
SCLK	PA11 PB05

表 130: SPI3 信号引脚

SYSCTL 信号名称	引脚
CLK_OBS_0	PA00 PA26
CLK_OBS_1	PA01 PA27
CLK_OBS_2	PA02 PA13 PA28
CLK_OBS_3	PA03

SYSCTL 信号名称	引脚
	PA12 PA29

表 131: SYSCTL 信号引脚

TRGM0 信号名称	引脚
P_00	PA00 PA20 PA24 PB04 PB12
P_01	PA01 PA21 PA25 PB05 PB13
P_02	PA02 PA22 PA26 PB06 PB14
P_03	PA03 PA23 PA27 PB07 PB15
P_04	PA04 PA16 PA28 PB00 PY04
P_05	PA05 PA17 PA29 PB01 PY05
P_06	PA06 PA18 PA30 PB02 PY06
P_07	PA07

TRGM0 信号名称	引脚
	PA19 PA31 PB03 PY07

表 132: TRGM0 信号引脚

UART0 信号名称	引脚
CTS	PA03 PB03 PY03
DE	PA02 PB02 PY02
RTS	PA02 PB02 PY02
RXD	PA01 PB01 PY01
TXD	PA00 PB00 PY00

表 133: UART0 信号引脚

UART1 信号名称	引脚
CTS	PA04 PB04 PY04
DE	PA05 PB05 PY05
RTS	PA05 PB05 PY05
RXD	PA06 PB06 PY06
TXD	PA07 PB07 PY07

UART1 信号名称	引脚

表 134: UART1 信号引脚

UART2 信号名称	引脚
CTS	PA11 PB11
DE	PA10 PB10
RTS	PA10 PB10
RXD	PA09 PB09
TXD	PA08 PB08

表 135: UART2 信号引脚

UART3 信号名称	引脚
CTS	PA12 PB12
DE	PA13 PB13
RTS	PA13 PB13
RXD	PA14 PB14
TXD	PA15 PB15

表 136: UART3 信号引脚

UART4 信号名称	引脚
CTS	PA19 PX03
DE	PA18 PX02
RTS	PA18 PX02
RXD	PA17 PX01
TXD	PA16 PX00

UART4 信号名称	引脚

表 137: UART4 信号引脚

UART5 信号名称	引脚
CTS	PA20 PX04
DE	PA21 PX05
RTS	PA21 PX05
RXD	PA22 PX06
TXD	PA23 PX07

表 138: UART5 信号引脚

UART6 信号名称	引脚
CTS	PA27
DE	PA26
RTS	PA26
RXD	PA25
TXD	PA24

表 139: UART6 信号引脚

UART7 信号名称	引脚
CTS	PA28
DE	PA29
RTS	PA29
RXD	PA30
TXD	PA31

表 140: UART7 信号引脚

USB0 信号名称	引脚
ID	PA31 PB08 PY00
OC	PA29 PB09 PY01
PWR	PA30

USB0 信号名称	引脚
	PB10 PY02

表 141: USB0 信号引脚

WDG0 信号名称	引脚
RST	PA11 PY01 PY05

表 142: WDG0 信号引脚

WDG1 信号名称	引脚
RST	PA14 PY02 PY06

表 143: WDG1 信号引脚

XPI0 信号名称	引脚
CA_CS0	PA31 PX02
CA_CS1	PA24 PX04
CA_DQS	PA25 PX03
CA_D_0	PA28 PX05
CA_D_1	PA30 PX01
CA_D_2	PA29 PX00
CA_D_3	PA26 PX07
CA_SCLK	PA27 PX06

表 144: XPI0 信号引脚

20.4 电源管理域外设管脚分配

本产品电源管理域的外设，不同模块的引脚分配总结如下：

PGPIO 信号名称	引脚
Y_00	PY00
Y_01	PY01
Y_02	PY02
Y_03	PY03
Y_04	PY04
Y_05	PY05
Y_06	PY06
Y_07	PY07

表 145: PGPIO 信号引脚

PTMR 信号名称	引脚
CAPT_0	PY05
CAPT_1	PY07
COMP_0	PY00 PY04
COMP_1	PY01 PY06
COMP_2	PY02
COMP_3	PY03

表 146: PTMR 信号引脚

PUART 信号名称	引脚
CTS	PY03
RTS	PY02
RXD	PY01
TXD	PY00

表 147: PUART 信号引脚

PWDG 信号名称	引脚
RST	PY05

表 148: PWDG 信号引脚

SOC 信号名称	引脚
GPIO_Y_00	PY00
GPIO_Y_01	PY01
GPIO_Y_02	PY02
GPIO_Y_03	PY03
GPIO_Y_04	PY04
GPIO_Y_05	PY05

SOC 信号名称	引脚
GPIO_Y_06	PY06
GPIO_Y_07	PY07

表 149: SOC 信号引脚

20.5 特殊功能引脚

芯片默认是通过 BOOT_MODE[1:0]=[PA03:PA02] 引脚选择三种不同的启动模式，启动配置如表 150。其他特殊引脚配置如表 151。

启动模式选择引脚		启动模式	说明
BOOT_MODE1	BOOT_MODE0		
0	0	XPI NOR 启动	从连接在 XPI0 上的串行 NOR FLASH 启动
0	1	在系统编程（ISP)/串行启动	从 UART0/USB0 上烧写固件，OTP, 或从 UART0/USB0 上启动
1	0	在系统编程（ISP)/串行启动	从 UART0/USB0 上烧写固件，OTP, 或从 UART0/USB0 上启动
1	1	保留模式	保留模式

表 150: 启动配置表

引脚名称	描述	建议用法
XTALI	24MHz 时钟输入	接 24MHz 晶体或有源时钟
XTALO	24MHz 时钟输出	接 24MHz 晶体或悬空

表 151: 特殊功能引脚配置

注意：本产品的 LQFP64、QFN48 封装上：

- USB_DP 与 PA24 复用，当用作 USB 功能时，应当配置 PA24 为模拟功能（PAD[FUNC_CTL].ANALOG 位置 1）
- USB_DM 与 PA25 复用，当用作 USB 功能时，应当配置 PA25 为模拟功能（PAD[FUNC_CTL].ANALOG 位置 1）
- 在系统编程（ISP）模式下，仅支持从 UART0 烧写固件，或从 UART0 启动

20.6 IO 复位状态

表 152总结了本产品所有 IO 在系统复位后的状态：

名称	复位后状态
PA04	高阻
PA05	输入内部上拉
PA06	输入内部下拉
PA07	输入内部上拉
PA08	输入内部上拉
其余 IO	输入状态保持器

表 152: IO 复位状态表

21 输入输出模块概述

本章节介绍了本产品的输入输出 IO 相关模块。本产品的输入输出相关控制模块包含通用 IO 控制模块 IOC，电源管理域 IO 控制模块 PIOC。GPIO 控制器，快速 GPIO 控制，GPIO 管理器以及电源管理域 GPIO 控制器 PGPIO。

21.1 IO 控制器

IO 控制器模块包括通用 IO 控制器 IOC，电源管理域 IO 控制器 PIOC。

通用 IO 控制器 IOC 可以控制通用 IO（PA，PB，PX）。

电源管理域 IO 控制器 PIOC 可以控制电源管理域 IO（PY）。它的功能和通用 IOC 一致，可以配置电源管理域 IO 的基本属性以及外设功能。

PIOC 可以把电源管理域 IO（PY）中的一个或者多个 IO 映射到系统电源域。之后，这些 IO 就可以由 IOC 控制。

IO 控制器支持对任意 IO 进行配置，如开漏控制，内部上下拉控制，施密特触发器，压摆率，驱动能力等。还可以配置每一个 IO 的外设复用功能映射，模拟输入和 IO 状态监测功能。通用 IO 的外设复用功能由 IOC 配置，如图 19。

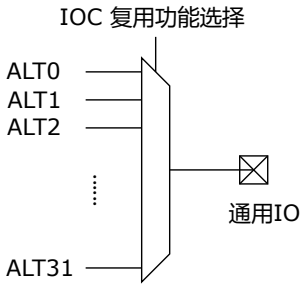


图 19: 通用 IO 外设复用功能选择

电源管理域 IO（PY）的外设复用功能由 PIOC 和 IOC 配置，如图 20。

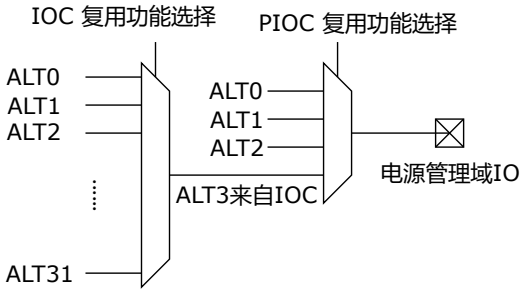


图 20: 电源管理域 IO 外设复用功能选择

当电源管理域 IO（PY）的外设复用功能由 PIOC 设置为 ALT3 时，IOC 针对该 IO 的配置生效，而 PIOC 的配置不再生效。

21.2 GPIO 控制器

GPIO 控制器包括：GPIO 控制器（GPIO0），快速 GPIO 控制器（FGPIO0），电源管理域 GPIO 控制器（PGPIO）。

GPIO0 和 FGPIO0 可以控制通用 IO（PA，PB，PX）。对任一 IO，由 GPIO 管理器 GPIOM 配置决定具体哪个控制器生效。

电源管理域 GPIO 控制器 PGPIO 可以控制电源管理域 IO（PY）。

PIOC 可以把电源管理域 IO（PY）中的一个或者多个 IO 映射到系统电源域。之后，这些 IO 就可以由 GPIO0 或 FGPIO0 控制。

GPIO 控制器 GPIO0 和快速 GPIO 控制器 FGPIO0，可以控制片上的通用 IO（PA，PB，PX）。

通用 IO 的 GPIO 控制如图 21。

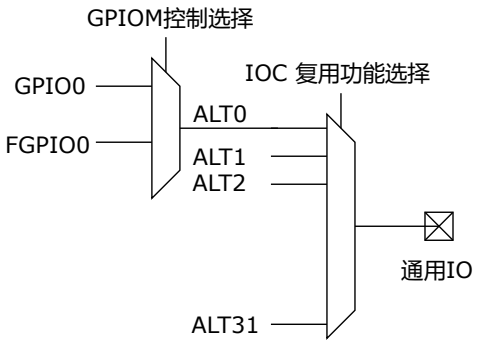


图 21: 通用 IO GPIO 控制选择

经过适当的 PIOC 和 IOC 配置，2 个 GPIO 控制器 GPIO0，GPIO1 和 2 个快速 GPIO 控制器 FGPIO0，FGPIO1 可以控制电源管理域 IO（PY）。

电源管理域 IO 的 GPIO 控制如图 22。

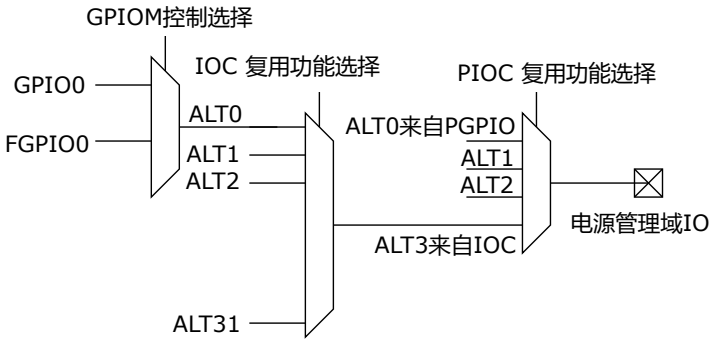


图 22: 电源管理域 IO GPIO 控制选择

快速 GPIO 控制器 (FGPIO0) 是处理器 CPU0 的私有外设，只能由 RISC-V 处理器访问。处理器支持以零等待周期访问自己的快速 GPIO 控制器。

GPIO 控制器与快速 GPIO 控制器功能基本相同，可以按照 IO 端口 Port 读取输入，配置 IO 的方向（输入或者输出），设置 IO 输出，或者同时把一个或者多个 IO 输出设置高，设置低或者翻转。

GPIO 控制器支持配置和生成 GPIO 中断。快速 GPIO 控制器不支持中断。

电源管理域 GPIO 控制器 PGPIO 是电源管理域 IO（PY）的 GPIO 控制器，和 GPIO 控制器一样具有读取 IO 输入和配置 IO 输出的功能。此外 PGPIO 作为电源管理域外设，它能在系统电源域掉电时保持工作，PGPIO 中断能在系统电源域掉电时把系统唤醒。

21.3 GPIO 管理器 GPIOM

GPIO 管理器用来管理所有的 IO。它的主要功能是配置 IO 的管理权限。GPIO 管理器为每一个 IO 选择快速 GPIO 控制器和 GPIO 控制器中的一个，作为这个 IO 的控制器。

GPIO 管理器支持配置任意 IO 输入的可见度，即快速 GPIO 控制器和 GPIO 控制器的输入寄存器可否读取到 IO 的输入。

复位后，GPIO 管理器默认 GPIO0 控制所有的通用 IO。所有的 GPIO 控制器和快速 GPIO 控制器都可以读取到 IO 的输入。

22 IO 控制器 IOC，PIOC，BIOC

本章节描述了 IO 控制器 IOC 的主要特性和功能。

22.1 特性总结

IO 控制器 IOC 的主要特性如下：

- 外设复用功能映射
- 输出回送控制 (loopback)
- 模拟功能配置
- 压摆率配置
- 开漏设置
- 施密特触发器
- 上下拉配置
- 驱动能力配置

22.2 功能描述

本章节描述 IO 控制器 IOC 的功能。

22.2.1 IO 基本配置

IO 控制器 IOC 可以用来配置 IO 的基本属性，这些属性包括 IO 的开漏选择，施密特触发器开关，压摆率内部上下拉电阻，以及驱动强度。用户可以通过 IOC_X_PAD_CTL 寄存器，配置任意 IO 的基本属性。

本产品的 IO 支持 3.3V 和 1.8V 两种工作电压，其中电池域的 IO 只工作在 3.3V。

IO 的开漏选择是指，用户可以把 IO 配置成开漏输出（open drain）。如果 IO 配置成开漏输出，那么输出低电平时，正常输出；输出高电平时，IO 不会驱动高电平，而是输出高阻，需要用户配置外部上拉电阻。

IO 的施密特触发器是指，用户可以打开 IO 的输入施密特触发器，即打开输入滞回（hysteresis），增加抗干扰能力。

IO 还可以打开内部的上下拉电阻，即可以配置成内部上拉，也可以配置成下拉。

IO 支持配置压摆率。

IO 支持配置驱动能力。

22.2.2 IO 外设功能配置

用户可以通过 IOC 的 IOC_X_FUNC_CTL 寄存器配置 IO 的外设功能，包括输出回送功能，模拟功能和外设功能映射。

用户打开 IO 的输出回送功能后，即可在输入端读取到输出信号。

用户打开 IO 的模拟功能后，这个 IO 就可以用作模拟外设的引脚，如 ADC，ACMP 等。

用户可以通过 IO 的外设功能复选器，选择映射到这个 IO 上的外设功能。有关 IO 和外设功能的映射表，请查阅 PINMUX 相关章节。

22.3 IOC 寄存器

22.3.1 寄存器说明

IOC 的寄存器列表如下：

IOC base address: 0xF4040000

PIOC base address: 0xF4118000

地址偏移	名称	描述	复位值
0x0000	PAD[PA00][FUNC_CTL]	ALT SELECT	0x00000000
0x0004	PAD[PA00][PAD_CTL]	PAD SETTINGS	0x01010056
0x0008	PAD[PA01][FUNC_CTL]	ALT SELECT	0x00000000
0x000C	PAD[PA01][PAD_CTL]	PAD SETTINGS	0x01010056
0x0010	PAD[PA02][FUNC_CTL]	ALT SELECT	0x00000000
0x0014	PAD[PA02][PAD_CTL]	PAD SETTINGS	0x01010056
0x0018	PAD[PA03][FUNC_CTL]	ALT SELECT	0x00000000
0x001C	PAD[PA03][PAD_CTL]	PAD SETTINGS	0x01010056
0x0020	PAD[PA04][FUNC_CTL]	ALT SELECT	0x00000000
0x0024	PAD[PA04][PAD_CTL]	PAD SETTINGS	0x01010056
0x0028	PAD[PA05][FUNC_CTL]	ALT SELECT	0x00000000
0x002C	PAD[PA05][PAD_CTL]	PAD SETTINGS	0x01010056
0x0030	PAD[PA06][FUNC_CTL]	ALT SELECT	0x00000000
0x0034	PAD[PA06][PAD_CTL]	PAD SETTINGS	0x01010056
0x0038	PAD[PA07][FUNC_CTL]	ALT SELECT	0x00000000
0x003C	PAD[PA07][PAD_CTL]	PAD SETTINGS	0x01010056
0x0040	PAD[PA08][FUNC_CTL]	ALT SELECT	0x00000000
0x0044	PAD[PA08][PAD_CTL]	PAD SETTINGS	0x01010056
0x0048	PAD[PA09][FUNC_CTL]	ALT SELECT	0x00000000
0x004C	PAD[PA09][PAD_CTL]	PAD SETTINGS	0x01010056
0x0050	PAD[PA10][FUNC_CTL]	ALT SELECT	0x00000000
0x0054	PAD[PA10][PAD_CTL]	PAD SETTINGS	0x01010056
0x0058	PAD[PA11][FUNC_CTL]	ALT SELECT	0x00000000
0x005C	PAD[PA11][PAD_CTL]	PAD SETTINGS	0x01010056
0x0060	PAD[PA12][FUNC_CTL]	ALT SELECT	0x00000000
0x0064	PAD[PA12][PAD_CTL]	PAD SETTINGS	0x01010056
0x0068	PAD[PA13][FUNC_CTL]	ALT SELECT	0x00000000
0x006C	PAD[PA13][PAD_CTL]	PAD SETTINGS	0x01010056
0x0070	PAD[PA14][FUNC_CTL]	ALT SELECT	0x00000000
0x0074	PAD[PA14][PAD_CTL]	PAD SETTINGS	0x01010056
0x0078	PAD[PA15][FUNC_CTL]	ALT SELECT	0x00000000
0x007C	PAD[PA15][PAD_CTL]	PAD SETTINGS	0x01010056
0x0080	PAD[PA16][FUNC_CTL]	ALT SELECT	0x00000000
0x0084	PAD[PA16][PAD_CTL]	PAD SETTINGS	0x01010056
0x0088	PAD[PA17][FUNC_CTL]	ALT SELECT	0x00000000

地址偏移	名称	描述	复位值
0x008C	PAD[PA17][PAD_CTL]	PAD SETTINGS	0x01010056
0x0090	PAD[PA18][FUNC_CTL]	ALT SELECT	0x00000000
0x0094	PAD[PA18][PAD_CTL]	PAD SETTINGS	0x01010056
0x0098	PAD[PA19][FUNC_CTL]	ALT SELECT	0x00000000
0x009C	PAD[PA19][PAD_CTL]	PAD SETTINGS	0x01010056
0x00A0	PAD[PA20][FUNC_CTL]	ALT SELECT	0x00000000
0x00A4	PAD[PA20][PAD_CTL]	PAD SETTINGS	0x01010056
0x00A8	PAD[PA21][FUNC_CTL]	ALT SELECT	0x00000000
0x00AC	PAD[PA21][PAD_CTL]	PAD SETTINGS	0x01010056
0x00B0	PAD[PA22][FUNC_CTL]	ALT SELECT	0x00000000
0x00B4	PAD[PA22][PAD_CTL]	PAD SETTINGS	0x01010056
0x00B8	PAD[PA23][FUNC_CTL]	ALT SELECT	0x00000000
0x00BC	PAD[PA23][PAD_CTL]	PAD SETTINGS	0x01010056
0x00C0	PAD[PA24][FUNC_CTL]	ALT SELECT	0x00000000
0x00C4	PAD[PA24][PAD_CTL]	PAD SETTINGS	0x01010056
0x00C8	PAD[PA25][FUNC_CTL]	ALT SELECT	0x00000000
0x00CC	PAD[PA25][PAD_CTL]	PAD SETTINGS	0x01010056
0x00D0	PAD[PA26][FUNC_CTL]	ALT SELECT	0x00000000
0x00D4	PAD[PA26][PAD_CTL]	PAD SETTINGS	0x01010056
0x00D8	PAD[PA27][FUNC_CTL]	ALT SELECT	0x00000000
0x00DC	PAD[PA27][PAD_CTL]	PAD SETTINGS	0x01010056
0x00E0	PAD[PA28][FUNC_CTL]	ALT SELECT	0x00000000
0x00E4	PAD[PA28][PAD_CTL]	PAD SETTINGS	0x01010056
0x00E8	PAD[PA29][FUNC_CTL]	ALT SELECT	0x00000000
0x00EC	PAD[PA29][PAD_CTL]	PAD SETTINGS	0x01010056
0x00F0	PAD[PA30][FUNC_CTL]	ALT SELECT	0x00000000
0x00F4	PAD[PA30][PAD_CTL]	PAD SETTINGS	0x01010056
0x00F8	PAD[PA31][FUNC_CTL]	ALT SELECT	0x00000000
0x00FC	PAD[PA31][PAD_CTL]	PAD SETTINGS	0x01010056
0x0100	PAD[PB00][FUNC_CTL]	ALT SELECT	0x00000000
0x0104	PAD[PB00][PAD_CTL]	PAD SETTINGS	0x01010056
0x0108	PAD[PB01][FUNC_CTL]	ALT SELECT	0x00000000
0x010C	PAD[PB01][PAD_CTL]	PAD SETTINGS	0x01010056
0x0110	PAD[PB02][FUNC_CTL]	ALT SELECT	0x00000000
0x0114	PAD[PB02][PAD_CTL]	PAD SETTINGS	0x01010056
0x0118	PAD[PB03][FUNC_CTL]	ALT SELECT	0x00000000
0x011C	PAD[PB03][PAD_CTL]	PAD SETTINGS	0x01010056
0x0120	PAD[PB04][FUNC_CTL]	ALT SELECT	0x00000000
0x0124	PAD[PB04][PAD_CTL]	PAD SETTINGS	0x01010056
0x0128	PAD[PB05][FUNC_CTL]	ALT SELECT	0x00000000

地址偏移	名称	描述	复位值
0x012C	PAD[PB05][PAD_CTL]	PAD SETTINGS	0x01010056
0x0130	PAD[PB06][FUNC_CTL]	ALT SELECT	0x00000000
0x0134	PAD[PB06][PAD_CTL]	PAD SETTINGS	0x01010056
0x0138	PAD[PB07][FUNC_CTL]	ALT SELECT	0x00000000
0x013C	PAD[PB07][PAD_CTL]	PAD SETTINGS	0x01010056
0x0140	PAD[PB08][FUNC_CTL]	ALT SELECT	0x00000000
0x0144	PAD[PB08][PAD_CTL]	PAD SETTINGS	0x01010056
0x0148	PAD[PB09][FUNC_CTL]	ALT SELECT	0x00000000
0x014C	PAD[PB09][PAD_CTL]	PAD SETTINGS	0x01010056
0x0150	PAD[PB10][FUNC_CTL]	ALT SELECT	0x00000000
0x0154	PAD[PB10][PAD_CTL]	PAD SETTINGS	0x01010056
0x0158	PAD[PB11][FUNC_CTL]	ALT SELECT	0x00000000
0x015C	PAD[PB11][PAD_CTL]	PAD SETTINGS	0x01010056
0x0160	PAD[PB12][FUNC_CTL]	ALT SELECT	0x00000000
0x0164	PAD[PB12][PAD_CTL]	PAD SETTINGS	0x01010056
0x0168	PAD[PB13][FUNC_CTL]	ALT SELECT	0x00000000
0x016C	PAD[PB13][PAD_CTL]	PAD SETTINGS	0x01010056
0x0170	PAD[PB14][FUNC_CTL]	ALT SELECT	0x00000000
0x0174	PAD[PB14][PAD_CTL]	PAD SETTINGS	0x01010056
0x0178	PAD[PB15][FUNC_CTL]	ALT SELECT	0x00000000
0x017C	PAD[PB15][PAD_CTL]	PAD SETTINGS	0x01010056
0x0D00	PAD[PX00][FUNC_CTL]	ALT SELECT	0x00000000
0x0D04	PAD[PX00][PAD_CTL]	PAD SETTINGS	0x01010056
0x0D04	PAD[PX01][FUNC_CTL]	ALT SELECT	0x00000000
0x0D08	PAD[PX01][PAD_CTL]	PAD SETTINGS	0x01010056
0x0D08	PAD[PX02][FUNC_CTL]	ALT SELECT	0x00000000
0x0D0C	PAD[PX02][PAD_CTL]	PAD SETTINGS	0x01010056
0x0D0C	PAD[PX03][FUNC_CTL]	ALT SELECT	0x00000000
0x0D10	PAD[PX03][PAD_CTL]	PAD SETTINGS	0x01010056
0x0D10	PAD[PX04][FUNC_CTL]	ALT SELECT	0x00000000
0x0D14	PAD[PX04][PAD_CTL]	PAD SETTINGS	0x01010056
0x0D14	PAD[PX05][FUNC_CTL]	ALT SELECT	0x00000000
0x0D18	PAD[PX05][PAD_CTL]	PAD SETTINGS	0x01010056
0x0D18	PAD[PX06][FUNC_CTL]	ALT SELECT	0x00000000
0x0D1C	PAD[PX06][PAD_CTL]	PAD SETTINGS	0x01010056
0x0D1C	PAD[PX07][FUNC_CTL]	ALT SELECT	0x00000000
0x0D20	PAD[PX07][PAD_CTL]	PAD SETTINGS	0x01010056
0x0E00	PAD[PY00][FUNC_CTL]	ALT SELECT	0x00000000
0x0E04	PAD[PY00][PAD_CTL]	PAD SETTINGS	0x01010056
0x0E08	PAD[PY01][FUNC_CTL]	ALT SELECT	0x00000000

地址偏移	名称	描述	复位值
0x0E0C	PAD[PY01][PAD_CTL]	PAD SETTINGS	0x01010056
0x0E10	PAD[PY02][FUNC_CTL]	ALT SELECT	0x00000000
0x0E14	PAD[PY02][PAD_CTL]	PAD SETTINGS	0x01010056
0x0E18	PAD[PY03][FUNC_CTL]	ALT SELECT	0x00000000
0x0E1C	PAD[PY03][PAD_CTL]	PAD SETTINGS	0x01010056
0x0E20	PAD[PY04][FUNC_CTL]	ALT SELECT	0x00000000
0x0E24	PAD[PY04][PAD_CTL]	PAD SETTINGS	0x01010056
0x0E28	PAD[PY05][FUNC_CTL]	ALT SELECT	0x00000000
0x0E2C	PAD[PY05][PAD_CTL]	PAD SETTINGS	0x01010056
0x0E30	PAD[PY06][FUNC_CTL]	ALT SELECT	0x00000000
0x0E34	PAD[PY06][PAD_CTL]	PAD SETTINGS	0x01010056
0x0E38	PAD[PY07][FUNC_CTL]	ALT SELECT	0x00000000
0x0E3C	PAD[PY07][PAD_CTL]	PAD SETTINGS	0x01010056

表 153: IOC 寄存器列表

IOC 的寄存器详细说明如下：

22.3.2 PAD[FUNC_CTL] (0x0 + 0x8 * n)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD															LOOP_BACK	RSVD							ANALOG	RSVD			ALT_SELECT				
N/A															RW	N/A							RW	N/A			RW				
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	x	x	x	x	x	x	x	0	x	x	x	0	0	0	0	0

PAD[FUNC_CTL] [31:0]

位域	名称	描述
16	LOOP_BACK	force input on 0: disable 1: enable
8	ANALOG	select analog pin in pad 0: disable 1: enable
4-0	ALT_SELECT	alt select 0: ALT0 1: ALT1 ... 31:ALT31

PAD[FUNC_CTL] 位域

22.3.3 PAD[PAD_CTL] (0x4 + 0x8 * n)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD							HYS	RSVD		PRS		RSVD	PS	PE	KE	RSVD							OD	RSVD	SR	SPD		RSVD	DS		
N/A							RW	N/A		RW		N/A	RW	RW	RW	N/A							RW	N/A	RW	RW		N/A	RW		
x	x	x	x	x	x	x	1	x	x	0	0	x	0	0	1	x	x	x	x	x	x	x	0	x	1	0	1	x	1	1	0

PAD[PAD_CTL] [31:0]

位域	名称	描述
24	HYS	schmitt trigger enable 0: disable 1: enable
21-20	PRS	select pull up/down internal resistance strength: For pull down, only have 100 Kohm resistance For pull up: 00: 100 KOhm 01: 47 KOhm 10: 22 KOhm 11: 22 KOhm
18	PS	pull select 0: pull down 1: pull up
17	PE	pull enable 0: pull disable 1: pull enable
16	KE	keeper capability enable 0: keeper disable 1: keeper enable
8	OD	open drain 0: open drain disable 1: open drain enable
6	SR	slew rate 0: Slow slew rate 1: Fast slew rate

位域	名称	描述
5-4	SPD	additional 2-bit slew rate to select IO cell operation frequency range with reduced switching noise 00: Slow frequency slew rate(50Mhz) 01: Medium frequency slew rate(100 Mhz) 10: Fast frequency slew rate(150 Mhz) 11: Max frequency slew rate(200Mhz)
2-0	DS	drive strength 1.8V Mode: 000: 260 Ohm 001: 260 Ohm 010: 130 Ohm 011: 88 Ohm 100: 65 Ohm 101: 52 Ohm 110: 43 Ohm 111: 37 Ohm 3.3V Mode: 000: 157 Ohm 001: 157 Ohm 010: 78 Ohm 011: 53 Ohm 100: 39 Ohm 101: 32 Ohm 110: 26 Ohm 111: 23 Ohm

PAD[PAD_CTL] 位域

23 GPIO 控制器

本章节描述了 GPIO 控制器的主要特性和功能。GPIO 控制器包括：GPIO 控制器 GPIO0，快速 GPIO 控制器 FGPIO0，电源管理域 GPIO 控制器（PGPIO）。

23.1 特性总结

本章节介绍 GPIO 控制器的主要特性：

- 配置 IO 作为输入或者输出
- 读取 IO 输入的状态
- 设置 IO 的输出
- 原子化操作设置 IO 输出高，输出低，翻转

GPIO，PGPIO 支持配置 GPIO 中断，FGPIO0 不支持生成中断。

23.2 功能描述

23.2.1 GPIO 控制

GPIO 控制器支持 OE 寄存器，每一个 IO 都有对应的 DIRECTION 控制位。用户把该位置 1 就可以把对应的 IO 配置为 GPIO 输出，反之该 IO 即为 GPIO 输入。

用户可以通过 GPIO 控制器的 DO 寄存器配置 GPIO 的输出。

GPIO 输出支持原子化操作寄存器：

- 输出高寄存器 SET，把此寄存器里对应位置 1，会把对应 IO 输出置高；置 0 则 IO 输出不变。
- 输出低寄存器 CLEAR，把此寄存器里对应位置 1，会把对应 IO 输出置低；置 0 则 IO 输出不变。
- 翻转寄存器 TOGGLE，把此寄存器里对应位置 1，会把对应 IO 输出翻转；置 0 则 IO 输出不变。

用户可以通过 GPIO 控制器的 DI 寄存器读取 IO 的电平状态。

注意，DI 寄存器可以实现 IO 监听。即无论 GPIO 配置为输入还是输出，或者对应的 IO 控制器 IOC 是否将 IO 功能映射为 GPIO，用户总能从 DI 读取到 IO 的状态。

23.2.2 GPIO 中断

用户可以通过 GPIO 控制器的 IE 寄存器打开 GPIO 中断，IE 寄存器内的对应位置 1 就可以使能对应 IO 的中断。

用户可以通过 GPIO 控制器的 TP 寄存器来指定中断的类型，对应位置 1，表示中断由边沿触发，对应位置 0，表示中断由电平触发。

用户可以通过 GPIO 控制器的 PL 寄存器来指定中断的极性，对应位置 1，表示中断由下降沿或者低电平触发，对应位置 0，表示中断由上升沿或高电平触发。

GPIO 控制器支持在检测到上升沿或下降沿生成中断，也支持双沿或单沿触发模式。

GPIO 支持生成异步中断，异步中断允许在系统时钟异常时生成中断。

用户可以通过 GPIO 控制器的 IF 寄存器来查询中断的状态，对应标志位置 1，表示对应 IO 有中断待处理。对标志位写 1，可以清除这个标志位。

23.3 GPIO 寄存器列表

GPIO 的寄存器列表如下：

FGPIO base address: 0x000C0000

GPIO0 base address: 0xF00D0000

PGPIO base address: 0xF411C000

地址偏移	名称	描述	复位值
0x0000	DI[GPIOA][VALUE]	gpioa 状态寄存器	0x00000000
0x0010	DI[GPIOB][VALUE]	gpiob 状态寄存器	0x00000000
0x00D0	DI[GPIOX][VALUE]	gpiox 状态寄存器	0x00000000
0x00E0	DI[GPIOY][VALUE]	gpioy 状态寄存器	0x00000000
0x0100	DO[GPIOA][VALUE]	gpioa 输出寄存器	0x00000000
0x0104	DO[GPIOA][SET]	gpioa 输出设置寄存器	0x00000000
0x0108	DO[GPIOA][CLEAR]	gpioa 输出清除寄存器	0x00000000
0x010C	DO[GPIOA][TOGGLE]	gpioa 输出翻转寄存器	0x00000000
0x0110	DO[GPIOB][VALUE]	gpiob 输出寄存器	0x00000000
0x0114	DO[GPIOB][SET]	gpiob 输出设置寄存器	0x00000000
0x0118	DO[GPIOB][CLEAR]	gpiob 输出清除寄存器	0x00000000
0x011C	DO[GPIOB][TOGGLE]	gpiob 输出翻转寄存器	0x00000000
0x01D0	DO[GPIOX][VALUE]	gpiox 输出寄存器	0x00000000
0x01D4	DO[GPIOX][SET]	gpiox 输出设置寄存器	0x00000000
0x01D8	DO[GPIOX][CLEAR]	gpiox 输出清除寄存器	0x00000000
0x01DC	DO[GPIOX][TOGGLE]	gpiox 输出翻转寄存器	0x00000000
0x01E0	DO[GPIOY][VALUE]	gpioy 输出寄存器	0x00000000
0x01E4	DO[GPIOY][SET]	gpioy 输出设置寄存器	0x00000000
0x01E8	DO[GPIOY][CLEAR]	gpioy 输出清除寄存器	0x00000000
0x01EC	DO[GPIOY][TOGGLE]	gpioy 输出翻转寄存器	0x00000000
0x0200	OE[GPIOA][VALUE]	gpioa 方向控制寄存器	0x00000000
0x0204	OE[GPIOA][SET]	gpioa 方向控制设置寄存器	0x00000000
0x0208	OE[GPIOA][CLEAR]	gpioa 方向控制清除寄存器	0x00000000
0x020C	OE[GPIOA][TOGGLE]	gpioa 方向控制翻转寄存器	0x00000000
0x0210	OE[GPIOB][VALUE]	gpiob 方向控制寄存器	0x00000000
0x0214	OE[GPIOB][SET]	gpiob 方向控制设置寄存器	0x00000000
0x0218	OE[GPIOB][CLEAR]	gpiob 方向控制清除寄存器	0x00000000
0x021C	OE[GPIOB][TOGGLE]	gpiob 方向控制翻转寄存器	0x00000000
0x02D0	OE[GPIOX][VALUE]	gpiox 方向控制寄存器	0x00000000
0x02D4	OE[GPIOX][SET]	gpiox 方向控制设置寄存器	0x00000000
0x02D8	OE[GPIOX][CLEAR]	gpiox 方向控制清除寄存器	0x00000000
0x02DC	OE[GPIOX][TOGGLE]	gpiox 方向控制翻转寄存器	0x00000000
0x02E0	OE[GPIOY][VALUE]	gpioy 方向控制寄存器	0x00000000
0x02E4	OE[GPIOY][SET]	gpioy 方向控制设置寄存器	0x00000000

地址偏移	名称	描述	复位值
0x02E8	OE[GPIOY][CLEAR]	gpioy 方向控制清除寄存器	0x00000000
0x02EC	OE[GPIOY][TOGGLE]	gpioy 方向控制翻转寄存器	0x00000000
0x0300	IF[GPIOA][VALUE]	gpioa 中断标志	0x00000000
0x0310	IF[GPIOB][VALUE]	gpiob 中断标志	0x00000000
0x03D0	IF[GPIOX][VALUE]	gpiox 中断标志	0x00000000
0x03E0	IF[GPIOY][VALUE]	gpioy 中断标志	0x00000000
0x0400	IE[GPIOA][VALUE]	gpioa 中断使能	0x00000000
0x0404	IE[GPIOA][SET]	gpioa 中断使能设置	0x00000000
0x0408	IE[GPIOA][CLEAR]	gpioa 中断使能清除	0x00000000
0x040C	IE[GPIOA][TOGGLE]	gpioa 中断使能翻转	0x00000000
0x0410	IE[GPIOB][VALUE]	gpiob 中断使能	0x00000000
0x0414	IE[GPIOB][SET]	gpiob 中断使能设置	0x00000000
0x0418	IE[GPIOB][CLEAR]	gpiob 中断使能清除	0x00000000
0x041C	IE[GPIOB][TOGGLE]	gpiob 中断使能翻转	0x00000000
0x04D0	IE[GPIOX][VALUE]	gpiox 中断使能	0x00000000
0x04D4	IE[GPIOX][SET]	gpiox 中断使能设置	0x00000000
0x04D8	IE[GPIOX][CLEAR]	gpiox 中断使能清除	0x00000000
0x04DC	IE[GPIOX][TOGGLE]	gpiox 中断使能翻转	0x00000000
0x04E0	IE[GPIOY][VALUE]	gpioy 中断使能	0x00000000
0x04E4	IE[GPIOY][SET]	gpioy 中断使能设置	0x00000000
0x04E8	IE[GPIOY][CLEAR]	gpioy 中断使能清除	0x00000000
0x04EC	IE[GPIOY][TOGGLE]	gpioy 中断使能翻转	0x00000000
0x0500	PL[GPIOA][VALUE]	gpioa 中断极性	0x00000000
0x0504	PL[GPIOA][SET]	gpioa 中断极设置	0x00000000
0x0508	PL[GPIOA][CLEAR]	gpioa 中断极性清除	0x00000000
0x050C	PL[GPIOA][TOGGLE]	gpioa 中断极性翻转	0x00000000
0x0510	PL[GPIOB][VALUE]	gpiob 中断极性	0x00000000
0x0514	PL[GPIOB][SET]	gpiob 中断极设置	0x00000000
0x0518	PL[GPIOB][CLEAR]	gpiob 中断极性清除	0x00000000
0x051C	PL[GPIOB][TOGGLE]	gpiob 中断极性翻转	0x00000000
0x05D0	PL[GPIOX][VALUE]	gpiox 中断极性	0x00000000
0x05D4	PL[GPIOX][SET]	gpiox 中断极设置	0x00000000
0x05D8	PL[GPIOX][CLEAR]	gpiox 中断极性清除	0x00000000
0x05DC	PL[GPIOX][TOGGLE]	gpiox 中断极性翻转	0x00000000
0x05E0	PL[GPIOY][VALUE]	gpioy 中断极性	0x00000000
0x05E4	PL[GPIOY][SET]	gpioy 中断极设置	0x00000000
0x05E8	PL[GPIOY][CLEAR]	gpioy 中断极性清除	0x00000000
0x05EC	PL[GPIOY][TOGGLE]	gpioy 中断极性翻转	0x00000000
0x0600	TP[GPIOA][VALUE]	gpioa 中断类型	0x00000000
0x0604	TP[GPIOA][SET]	gpioa 中断类型设置	0x00000000

地址偏移	名称	描述	复位值
0x0608	TP[GPIOA][CLEAR]	gpioa 中断类型清除	0x00000000
0x060C	TP[GPIOA][TOGGLE]	gpioa 中断类型翻转	0x00000000
0x0610	TP[GPIOB][VALUE]	gpiob 中断类型	0x00000000
0x0614	TP[GPIOB][SET]	gpiob 中断类型设置	0x00000000
0x0618	TP[GPIOB][CLEAR]	gpiob 中断类型清除	0x00000000
0x061C	TP[GPIOB][TOGGLE]	gpiob 中断类型翻转	0x00000000
0x06D0	TP[GPIOX][VALUE]	gpiox 中断类型	0x00000000
0x06D4	TP[GPIOX][SET]	gpiox 中断类型设置	0x00000000
0x06D8	TP[GPIOX][CLEAR]	gpiox 中断类型清除	0x00000000
0x06DC	TP[GPIOX][TOGGLE]	gpiox 中断类型翻转	0x00000000
0x06E0	TP[GPIOY][VALUE]	gpioy 中断类型	0x00000000
0x06E4	TP[GPIOY][SET]	gpioy 中断类型设置	0x00000000
0x06E8	TP[GPIOY][CLEAR]	gpioy 中断类型清除	0x00000000
0x06EC	TP[GPIOY][TOGGLE]	gpioy 中断类型翻转	0x00000000
0x0700	AS[GPIOA][VALUE]	gpioa 异步中断	0x00000000
0x0704	AS[GPIOA][SET]	gpioa 异步中断设置	0x00000000
0x0708	AS[GPIOA][CLEAR]	gpioa 异步中断清除	0x00000000
0x070C	AS[GPIOA][TOGGLE]	gpioa 异步中断翻转	0x00000000
0x0710	AS[GPIOB][VALUE]	gpiob 异步中断	0x00000000
0x0714	AS[GPIOB][SET]	gpiob 异步中断设置	0x00000000
0x0718	AS[GPIOB][CLEAR]	gpiob 异步中断清除	0x00000000
0x071C	AS[GPIOB][TOGGLE]	gpiob 异步中断翻转	0x00000000
0x07D0	AS[GPIOX][VALUE]	gpiox 异步中断	0x00000000
0x07D4	AS[GPIOX][SET]	gpiox 异步中断设置	0x00000000
0x07D8	AS[GPIOX][CLEAR]	gpiox 异步中断清除	0x00000000
0x07DC	AS[GPIOX][TOGGLE]	gpiox 异步中断翻转	0x00000000
0x07E0	AS[GPIOY][VALUE]	gpioy 异步中断	0x00000000
0x07E4	AS[GPIOY][SET]	gpioy 异步中断设置	0x00000000
0x07E8	AS[GPIOY][CLEAR]	gpioy 异步中断清除	0x00000000
0x07EC	AS[GPIOY][TOGGLE]	gpioy 异步中断翻转	0x00000000
0x0800	PD[GPIOA][VALUE]	gpioa 双沿触发中断	0x00000000
0x0804	PD[GPIOA][SET]	gpioa 双沿触发中断设置	0x00000000
0x0808	PD[GPIOA][CLEAR]	gpioa 双沿触发中断清楚	0x00000000
0x080C	PD[GPIOA][TOGGLE]	gpioa 双沿触发中断翻转	0x00000000
0x0810	PD[GPIOB][VALUE]	gpiob 双沿触发中断	0x00000000
0x0814	PD[GPIOB][SET]	gpiob 双沿触发中断设置	0x00000000
0x0818	PD[GPIOB][CLEAR]	gpiob 双沿触发中断清楚	0x00000000
0x081C	PD[GPIOB][TOGGLE]	gpiob 双沿触发中断翻转	0x00000000
0x08D0	PD[GPIOX][VALUE]	gpiox 双沿触发中断	0x00000000
0x08D4	PD[GPIOX][SET]	gpiox 双沿触发中断设置	0x00000000

地址偏移	名称	描述	复位值
0x08D8	PD[GPIOX][CLEAR]	gpiox 双沿触发中断清楚	0x00000000
0x08DC	PD[GPIOX][TOGGLE]	gpiox 双沿触发中断翻转	0x00000000
0x08E0	PD[GPIOY][VALUE]	gpioy 双沿触发中断	0x00000000
0x08E4	PD[GPIOY][SET]	gpioy 双沿触发中断设置	0x00000000
0x08E8	PD[GPIOY][CLEAR]	gpioy 双沿触发中断清楚	0x00000000
0x08EC	PD[GPIOY][TOGGLE]	gpioy 双沿触发中断翻转	0x00000000

表 154: GPIO 寄存器列表

23.4 GPIO 寄存器描述

GPIO 的寄存器详细说明如下：

23.4.1 DI[VALUE] (0x0 + 0x10 * n)

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

DI[VALUE] [31:0]

位域	名称	描述
31-0	INPUT	GPIO 输入值，每一位代表一个引脚 0: 引脚上为低电平 1: 引脚上为高电平

DI[VALUE] 位域

23.4.2 DO[VALUE] (0x100 + 0x10 * n)

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

DO[VALUE] [31:0]

位域	名称	描述
31-0	OUTPUT	GPIO 输出值，每一位代表一个引脚 0: 引脚输出低电平 1: 引脚输出高电平

DO[VALUE] 位域

23.4.3 DO[SET] (0x104 + 0x10 * n)

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

DO[SET] [31:0]

位域	名称	描述
31-0	OUTPUT	GPIO 输出值置位，每一位代表一个引脚，写 1 则置位，写 0 没有影响 0: 引脚输出不变 1: 引脚输出高电平

DO[SET] 位域

23.4.4 DO[CLEAR] (0x108 + 0x10 * n)

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

DO[CLEAR] [31:0]

位域	名称	描述
31-0	OUTPUT	GPIO 输出值清零，每一位代表一个引脚，写 1 则清零，写 0 没有影响 0: 引脚输出不变 1: 引脚输出低电平

DO[CLEAR] 位域

23.4.5 DO[TOGGLE] (0x10C + 0x10 * n)

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

DO[TOGGLE] [31:0]

位域	名称	描述
31-0	OUTPUT	GPIO 输出值翻转，每一位代表一个引脚，写 1 则翻转，写 0 没有影响 0: 引脚输出不变 1: 引脚输出翻转

DO[TOGGLE] 位域

23.4.6 OE[VALUE] (0x200 + 0x10 * n)

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

OE[VALUE] [31:0]

位域	名称	描述
31-0	DIRECTION	GPIO 方向，每一位代表一个引脚 0: 输入 1: 输出

OE[VALUE] 位域

23.4.7 OE[SET] (0x204 + 0x10 * n)

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

OE[SET] [31:0]

位域	名称	描述
31-0	DIRECTION	GPIO 方向置位，每一位代表一个引脚 0: OE 不变 1: OE 置 1

OE[SET] 位域

23.4.8 OE[CLEAR] (0x208 + 0x10 * n)

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

OE[CLEAR] [31:0]

位域	名称	描述
31-0	DIRECTION	GPIO 方向清零，每一位代表一个引脚 0: OE 不变 1: OE 置 0

OE[CLEAR] 位域

23.4.9 OE[TOGGLE] (0x20C + 0x10 * n)

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

OE[TOGGLE] [31:0]

位域	名称	描述
31-0	DIRECTION	GPIO 方向翻转，每一位代表一个引脚 0: OE 不变 1: OE 翻转

OE[TOGGLE] 位域

23.4.10 IF[VALUE] (0x300 + 0x10 * n)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IRQ_FLAG																															
W1C																															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

IF[VALUE] [31:0]

位域	名称	描述
31-0	IRQ_FLAG	GPIO 中断标志，每一位代表一个引脚，写 1 清零，写 0 无影响 0: 没有产生中断 1: 产生中断

IF[VALUE] 位域

23.4.11 IE[VALUE] (0x400 + 0x10 * n)

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

IE[VALUE] [31:0]

位域	名称	描述
31-0	IRQ_EN	GPIO 中断使能，每一位代表一个引脚 0: 禁止中断 1: 使能中断

IE[VALUE] 位域

23.4.12 IE[SET] (0x404 + 0x10 * n)

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

IE[SET] [31:0]

位域	名称	描述
31-0	IRQ_EN	GPIO 中断使能置位，每一位代表一个引脚 0: 禁止中断 1: 使能中断

IE[SET] 位域

23.4.13 IE[CLEAR] (0x408 + 0x10 * n)

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

IE[CLEAR] [31:0]

位域	名称	描述
31-0	IRQ_EN	GPIO 中断使能清零，每一位代表一个引脚 0: 禁止中断 1: 使能中断

IE[CLEAR] 位域

23.4.14 IE[TOGGLE] (0x40C + 0x10 * n)

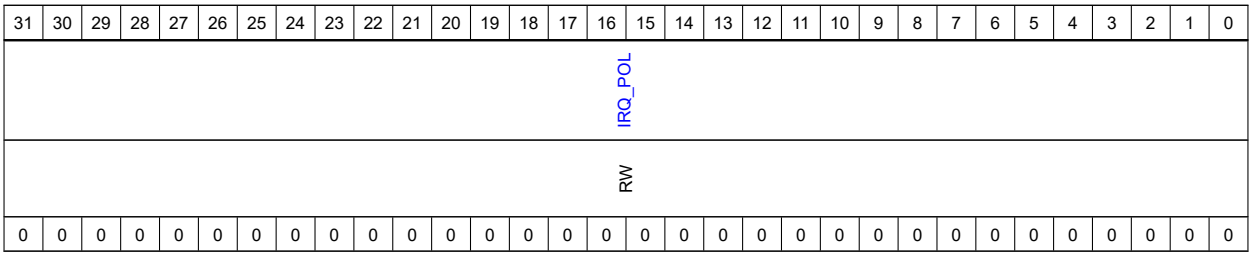
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IRQ_EN																															
RW																															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

IE[TOGGLE] [31:0]

位域	名称	描述
31-0	IRQ_EN	GPIO 中断使能翻转，每一位代表一个引脚 0: 禁止中断 1: 使能中断

IE[TOGGLE] 位域

23.4.15 PL[VALUE] (0x500 + 0x10 * n)

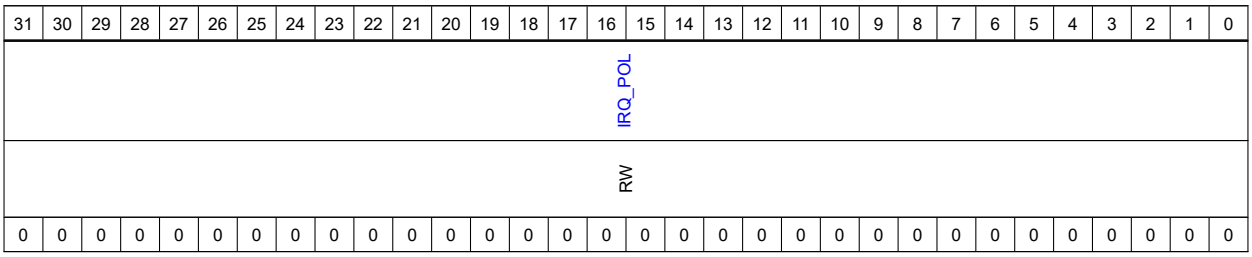


PL[VALUE] [31:0]

位域	名称	描述
31-0	IRQ_POL	GPIO 中断极性，每一位代表一个引脚 0: 中断在高电平或上升沿产生 1: 中断在低电平或下降沿产生

PL[VALUE] 位域

23.4.16 PL[SET] (0x504 + 0x10 * n)

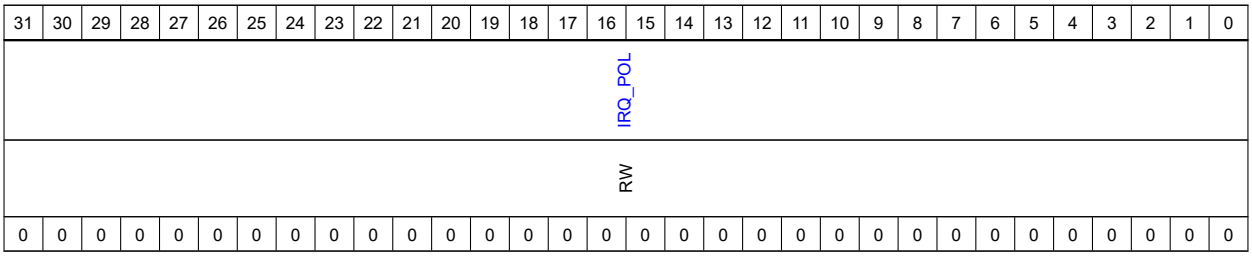


PL[SET] [31:0]

位域	名称	描述
31-0	IRQ_POL	GPIO 中断极性置位，每一位代表一个引脚 0: 中断在高电平或上升沿产生 1: 中断在低电平或下降沿产生

PL[SET] 位域

23.4.17 PL[CLEAR] (0x508 + 0x10 * n)



PL[CLEAR] [31:0]

位域	名称	描述
31-0	IRQ_POL	GPIO 中断极性清零，每一位代表一个引脚 0: 中断在高电平或上升沿产生 1: 中断在低电平或下降沿产生

PL[CLEAR] 位域

23.4.18 PL[TOGGLE] (0x50C + 0x10 * n)

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

PL[TOGGLE] [31:0]

位域	名称	描述
31-0	IRQ_POL	GPIO 中断极性翻转，每一位代表一个引脚 0: 中断在高电平或上升沿产生 1: 中断在低电平或下降沿产生

PL[TOGGLE] 位域

23.4.19 TP[VALUE] (0x600 + 0x10 * n)

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

TP[VALUE] [31:0]

位域	名称	描述
31-0	IRQ_TYPE	GPIO 中断类型，每一位代表一个引脚 0: 中断以电平方式产生 1: 中断以边沿方式产生

TP[VALUE] 位域

23.4.20 TP[SET] (0x604 + 0x10 * n)

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

TP[SET] [31:0]

位域	名称	描述
31-0	IRQ_TYPE	GPIO 中断类型置位，每一位代表一个引脚 0: 中断以电平方式产生 1: 中断以边沿方式产生

TP[SET] 位域

23.4.21 TP[CLEAR] (0x608 + 0x10 * n)

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

TP[CLEAR] [31:0]

位域	名称	描述
31-0	IRQ_TYPE	GPIO 中断类型清零，每一位代表一个引脚 0: 中断以电平方式产生 1: 中断以边沿方式产生

TP[CLEAR] 位域

23.4.22 TP[TOGGLE] (0x60C + 0x10 * n)

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

TP[TOGGLE] [31:0]

位域	名称	描述
31-0	IRQ_TYPE	GPIO 中断类型翻转，每一位代表一个引脚 0: 中断以电平方式产生 1: 中断以边沿方式产生

TP[TOGGLE] 位域

23.4.23 AS[VALUE] (0x700 + 0x10 * n)

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

AS[VALUE] [31:0]

位域	名称	描述
31-0	IRQ_ASYNC	GPIO 以异步方式产生中断，每一位代表一个引脚。 0: 使用系统时钟产生中断 1: 利用组合逻辑产生中断 注：异步方式可以在没有时钟的条件下产生中断，但中断信号的产生对环境干扰较敏感。

AS[VALUE] 位域

23.4.24 AS[SET] (0x704 + 0x10 * n)

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

AS[SET] [31:0]

位域	名称	描述
31-0	IRQ_ASYNC	GPIO 以异步方式产生中断置位，每一位代表一个引脚。 0: 使用系统时钟产生中断 1: 利用组合逻辑产生中断 注：异步方式可以在没有时钟的条件下产生中断，但中断信号的产生对环境干扰较敏感。

位域	名称	描述

AS[SET] 位域

23.4.25 AS[CLEAR] (0x708 + 0x10 * n)

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

AS[CLEAR] [31:0]

位域	名称	描述
31-0	IRQ_ASYNC	GPIO 以异步方式产生中断清零，每一位代表一个引脚。 0: 使用系统时钟产生中断 1: 利用组合逻辑产生中断 注：异步方式可以在没有时钟的条件下产生中断，但中断信号的产生对环境干扰较敏感。

AS[CLEAR] 位域

23.4.26 AS[TOGGLE] (0x70C + 0x10 * n)

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

AS[TOGGLE] [31:0]

位域	名称	描述
31-0	IRQ_ASYNC	GPIO 以异步方式产生中断翻转，每一位代表一个引脚。 0: 使用系统时钟产生中断 1: 利用组合逻辑产生中断 注：异步方式可以在没有时钟的条件下产生中断，但中断信号的产生对环境干扰较敏感。

AS[TOGGLE] 位域

23.4.27 PD[VALUE] (0x800 + 0x10 * n)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																															IRQ_DUAL
N/A																															RW
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0

PD[VALUE] [31:0]

位域	名称	描述
0	IRQ_DUAL	双沿 (上升沿和下降沿) 触发中断 0：单沿触发 (具体上升还是下降沿由 [PL] 决定) 1：双沿触发

PD[VALUE] 位域

23.4.28 PD[SET] (0x804 + 0x10 * n)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																															IRQ_DUAL
N/A																															RW
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0

PD[SET] [31:0]

位域	名称	描述
0	IRQ_DUAL	双沿 (上升沿和下降沿) 触发中断-> 设置 0：单沿触发 (具体上升还是下降沿由 [PL] 决定) 1：双沿触发

PD[SET] 位域

23.4.29 PD[CLEAR] (0x808 + 0x10 * n)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																															IRQ_DUAL
N/A																															RW
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0

PD[CLEAR] [31:0]

位域	名称	描述
0	IRQ_DUAL	双沿 (上升沿和下降沿) 触发中断-> 清除 0：保持之前的单沿或双沿配置 1：单沿触发

PD[CLEAR] 位域

23.4.30 PD[TOGGLE] (0x80C + 0x10 * n)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																															IRQ_DUAL
N/A																															RW
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0

PD[TOGGLE] [31:0]

位域	名称	描述
0	IRQ_DUAL	双沿 (上升沿和下降沿) 触发中断-> 翻转 0：保持之前的单沿或双沿配置 1：翻转之前的单沿或双沿配置，比如，之前是单沿，该 bit 为 1 时，则变为双沿，之前时双沿触发的，该 bit 为 1 时，则变为单沿。

PD[TOGGLE] 位域

24 GPIO 管理器 GPIOM

本章节描述了 GPIO 管理器 GPIOM 的主要特性和功能。GPIO 管理器的主要功能是为任一 IO 指定 GPIO 配置生效的模块，每个 IO 都可以单独地从 GPIO 控制器（GPIO0）和快速 GPIO 控制器（FGPIO0）中指定作为其控制器。

24.1 特性总结

本章节介绍 GPIO 管理器 GPIOM 的主要特性：

- 分配 IO 给指定的 GPIO 控制器
- 配置 IO 输入是否对特定 GPIO 控制器可见
- GPIOM 寄存器访问控制
- GPIOM 寄存器锁定

24.2 功能描述

本章节介绍 GPIO 管理器 GPIOM 的功能。

24.2.1 GPIO 分配

本产品支持多个 GPIO 控制器，其目的是为了方便多核，多任务间实现 GPIO 资源隔离。GPIO 管理器的主要功能是管理所有的 IO，每一个 IO 都可以单独地指定由某一个 GPIO 控制器控制，其他 GPIO 控制器无法控制 GPIO。

用户可以通过 GPIOxASSIGNy 寄存器实现管理 Px 的 y 引脚。该寄存器的 SELECT 位域可以选择这个 IO 收到哪个 GPIO 控制器控制：

- 0’b0，GPIO0
- 0’b1，FGPIO0

GPIOxASSIGNy 寄存器的 HIDE 位域，由 2 个 HIDE 位组成，每一位置 1，即表示这个 IO 的输入在对应的 GPIO 控制器 DI 寄存器内不可见，即 GPIO 控制器无法读取到这个 IO 的输入。

- HIDE[0]，置 1 表示 IO 对 GPIO0 不可见
- HIDE[1]，置 1 表示 IO 对 FGPIO0 不可见

24.2.2 访问控制

GPIOxASSIGNy 寄存器的 LOCK 位，可以用来锁定这个 IO 的对应寄存器。一旦置 1，这个寄存器的配置直到下次复位前都不能再更改。

GPIOxASSIGNy 寄存器的 NON_SEC 位，可以按照系统安全状态进行访问权限控制。一旦置 1，寄存器只能在安全状态下访问。请访问系统安全的相关章节获取系统安全状态的详细信息。

24.3 GPIOM 寄存器列表

GPIOM 的寄存器列表如下：

GPIOM base address: 0xF00D8000

地址偏移	名称	描述	复位值
0x0000	ASSIGN[GPIOA][PIN][PIN00]	GPIOA 管理寄存器	0x00000000
0x0004	ASSIGN[GPIOA][PIN][PIN01]	GPIOA 管理寄存器	0x00000000
0x0008	ASSIGN[GPIOA][PIN][PIN02]	GPIOA 管理寄存器	0x00000000
0x000C	ASSIGN[GPIOA][PIN][PIN03]	GPIOA 管理寄存器	0x00000000
0x0010	ASSIGN[GPIOA][PIN][PIN04]	GPIOA 管理寄存器	0x00000000
0x0014	ASSIGN[GPIOA][PIN][PIN05]	GPIOA 管理寄存器	0x00000000
0x0018	ASSIGN[GPIOA][PIN][PIN06]	GPIOA 管理寄存器	0x00000000
0x001C	ASSIGN[GPIOA][PIN][PIN07]	GPIOA 管理寄存器	0x00000000
0x0020	ASSIGN[GPIOA][PIN][PIN08]	GPIOA 管理寄存器	0x00000000
0x0024	ASSIGN[GPIOA][PIN][PIN09]	GPIOA 管理寄存器	0x00000000
0x0028	ASSIGN[GPIOA][PIN][PIN10]	GPIOA 管理寄存器	0x00000000
0x002C	ASSIGN[GPIOA][PIN][PIN11]	GPIOA 管理寄存器	0x00000000
0x0030	ASSIGN[GPIOA][PIN][PIN12]	GPIOA 管理寄存器	0x00000000
0x0034	ASSIGN[GPIOA][PIN][PIN13]	GPIOA 管理寄存器	0x00000000
0x0038	ASSIGN[GPIOA][PIN][PIN14]	GPIOA 管理寄存器	0x00000000
0x003C	ASSIGN[GPIOA][PIN][PIN15]	GPIOA 管理寄存器	0x00000000
0x0040	ASSIGN[GPIOA][PIN][PIN16]	GPIOA 管理寄存器	0x00000000
0x0044	ASSIGN[GPIOA][PIN][PIN17]	GPIOA 管理寄存器	0x00000000
0x0048	ASSIGN[GPIOA][PIN][PIN18]	GPIOA 管理寄存器	0x00000000
0x004C	ASSIGN[GPIOA][PIN][PIN19]	GPIOA 管理寄存器	0x00000000
0x0050	ASSIGN[GPIOA][PIN][PIN20]	GPIOA 管理寄存器	0x00000000
0x0054	ASSIGN[GPIOA][PIN][PIN21]	GPIOA 管理寄存器	0x00000000
0x0058	ASSIGN[GPIOA][PIN][PIN22]	GPIOA 管理寄存器	0x00000000
0x005C	ASSIGN[GPIOA][PIN][PIN23]	GPIOA 管理寄存器	0x00000000
0x0060	ASSIGN[GPIOA][PIN][PIN24]	GPIOA 管理寄存器	0x00000000
0x0064	ASSIGN[GPIOA][PIN][PIN25]	GPIOA 管理寄存器	0x00000000
0x0068	ASSIGN[GPIOA][PIN][PIN26]	GPIOA 管理寄存器	0x00000000
0x006C	ASSIGN[GPIOA][PIN][PIN27]	GPIOA 管理寄存器	0x00000000
0x0070	ASSIGN[GPIOA][PIN][PIN28]	GPIOA 管理寄存器	0x00000000
0x0074	ASSIGN[GPIOA][PIN][PIN29]	GPIOA 管理寄存器	0x00000000
0x0078	ASSIGN[GPIOA][PIN][PIN30]	GPIOA 管理寄存器	0x00000000
0x007C	ASSIGN[GPIOA][PIN][PIN31]	GPIOA 管理寄存器	0x00000000
0x0080	ASSIGN[GPIOB][PIN][PIN00]	GPIOB 管理寄存器	0x00000000
0x0084	ASSIGN[GPIOB][PIN][PIN01]	GPIOB 管理寄存器	0x00000000
0x0088	ASSIGN[GPIOB][PIN][PIN02]	GPIOB 管理寄存器	0x00000000
0x008C	ASSIGN[GPIOB][PIN][PIN03]	GPIOB 管理寄存器	0x00000000
0x0090	ASSIGN[GPIOB][PIN][PIN04]	GPIOB 管理寄存器	0x00000000
0x0094	ASSIGN[GPIOB][PIN][PIN05]	GPIOB 管理寄存器	0x00000000
0x0098	ASSIGN[GPIOB][PIN][PIN06]	GPIOB 管理寄存器	0x00000000
0x009C	ASSIGN[GPIOB][PIN][PIN07]	GPIOB 管理寄存器	0x00000000

地址偏移	名称	描述	复位值
0x00A0	ASSIGN[GPIOB][PIN][PIN08]	GPIOB 管理寄存器	0x00000000
0x00A4	ASSIGN[GPIOB][PIN][PIN09]	GPIOB 管理寄存器	0x00000000
0x00A8	ASSIGN[GPIOB][PIN][PIN10]	GPIOB 管理寄存器	0x00000000
0x00AC	ASSIGN[GPIOB][PIN][PIN11]	GPIOB 管理寄存器	0x00000000
0x00B0	ASSIGN[GPIOB][PIN][PIN12]	GPIOB 管理寄存器	0x00000000
0x00B4	ASSIGN[GPIOB][PIN][PIN13]	GPIOB 管理寄存器	0x00000000
0x00B8	ASSIGN[GPIOB][PIN][PIN14]	GPIOB 管理寄存器	0x00000000
0x00BC	ASSIGN[GPIOB][PIN][PIN15]	GPIOB 管理寄存器	0x00000000
0x00C0	ASSIGN[GPIOB][PIN][PIN16]	GPIOB 管理寄存器	0x00000000
0x00C4	ASSIGN[GPIOB][PIN][PIN17]	GPIOB 管理寄存器	0x00000000
0x00C8	ASSIGN[GPIOB][PIN][PIN18]	GPIOB 管理寄存器	0x00000000
0x00CC	ASSIGN[GPIOB][PIN][PIN19]	GPIOB 管理寄存器	0x00000000
0x00D0	ASSIGN[GPIOB][PIN][PIN20]	GPIOB 管理寄存器	0x00000000
0x00D4	ASSIGN[GPIOB][PIN][PIN21]	GPIOB 管理寄存器	0x00000000
0x00D8	ASSIGN[GPIOB][PIN][PIN22]	GPIOB 管理寄存器	0x00000000
0x00DC	ASSIGN[GPIOB][PIN][PIN23]	GPIOB 管理寄存器	0x00000000
0x00E0	ASSIGN[GPIOB][PIN][PIN24]	GPIOB 管理寄存器	0x00000000
0x00E4	ASSIGN[GPIOB][PIN][PIN25]	GPIOB 管理寄存器	0x00000000
0x00E8	ASSIGN[GPIOB][PIN][PIN26]	GPIOB 管理寄存器	0x00000000
0x00EC	ASSIGN[GPIOB][PIN][PIN27]	GPIOB 管理寄存器	0x00000000
0x00F0	ASSIGN[GPIOB][PIN][PIN28]	GPIOB 管理寄存器	0x00000000
0x00F4	ASSIGN[GPIOB][PIN][PIN29]	GPIOB 管理寄存器	0x00000000
0x00F8	ASSIGN[GPIOB][PIN][PIN30]	GPIOB 管理寄存器	0x00000000
0x00FC	ASSIGN[GPIOB][PIN][PIN31]	GPIOB 管理寄存器	0x00000000
0x0680	ASSIGN[GPIOX][PIN][PIN00]	GPIOX 管理寄存器	0x00000000
0x0684	ASSIGN[GPIOX][PIN][PIN01]	GPIOX 管理寄存器	0x00000000
0x0688	ASSIGN[GPIOX][PIN][PIN02]	GPIOX 管理寄存器	0x00000000
0x068C	ASSIGN[GPIOX][PIN][PIN03]	GPIOX 管理寄存器	0x00000000
0x0690	ASSIGN[GPIOX][PIN][PIN04]	GPIOX 管理寄存器	0x00000000
0x0694	ASSIGN[GPIOX][PIN][PIN05]	GPIOX 管理寄存器	0x00000000
0x0698	ASSIGN[GPIOX][PIN][PIN06]	GPIOX 管理寄存器	0x00000000
0x069C	ASSIGN[GPIOX][PIN][PIN07]	GPIOX 管理寄存器	0x00000000
0x0700	ASSIGN[GPIOY][PIN][PIN00]	GPIOY 管理寄存器	0x00000000
0x0704	ASSIGN[GPIOY][PIN][PIN01]	GPIOY 管理寄存器	0x00000000
0x0708	ASSIGN[GPIOY][PIN][PIN02]	GPIOY 管理寄存器	0x00000000
0x070C	ASSIGN[GPIOY][PIN][PIN03]	GPIOY 管理寄存器	0x00000000
0x0710	ASSIGN[GPIOY][PIN][PIN04]	GPIOY 管理寄存器	0x00000000
0x0714	ASSIGN[GPIOY][PIN][PIN05]	GPIOY 管理寄存器	0x00000000
0x0718	ASSIGN[GPIOY][PIN][PIN06]	GPIOY 管理寄存器	0x00000000
0x071C	ASSIGN[GPIOY][PIN][PIN07]	GPIOY 管理寄存器	0x00000000

地址偏移	名称	描述	复位值

表 155: GPIOM 寄存器列表

24.4 GPIOM 寄存器描述

GPIOM 的寄存器详细说明如下：

24.4.1 ASSIGN[PIN] (0x0 + 0x80 * n + 0x4 * m)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LOCK	RSVD																			HIDE				RSVD						SELECT	
RW	N/A																			RW				N/A						RW	
0	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	x	x	x	x	x	x	0	0

ASSIGN[PIN] [31:0]

位域	名称	描述
31	LOCK	锁定该寄存器的设置，锁定后无法解锁，锁定只能由复位清零 0: 寄存器字段可以修改 1: 寄存器字段不可修改
11-8	HIDE	引脚状态对 GPIO 可见 0 位: 1, GPIO0 不可见; 0: GPIO0 可见 1 位: 1, CPU0 快速 GPIO0 不可见; 0: CPU0 快速 GPIO0 可见
1-0	SELECT	选择引脚控制来源 0: GPIO0; 2: CPU0 快速 GPIO

ASSIGN[PIN] 位域

25 存储器概述

本章节介绍了本产品的内部存储器和外部存储外设相关接口。

25.1 内部 SRAM

本产品集成了总共 288 KB 内部 RAM，其中包括处理器的本地存储器和通用内存。

25.1.1 本地存储器 Local Memory

本地存储器 LM 包括指令本地存储器 ILM 和数据本地存储器 DLM，是指可以与内核工作在相同时钟频率，可以供内核实现零等待周期访问的内存。本产品上 RISC-V CPU0 配置有 256 KB 的本地存储器。本地存储器 LM 可被处理器直接访问，也能通过处理器的总线从接口被片上的其他主设备访问，其对应系统地址映射表上的 LM 和 LM_SLV。

ILD 和 DLM 的分配如下：

- ILM0，RISC-V CPU0 的指令本地存储器，128KB;
- DLM0，RISC-V CPU0 的数据本地存储器，128KB;

25.1.2 通用内存

本产品支持通用内存如下：

- AHB SRAM，32KB，位于 AHB 总线;

25.2 通用寄存器

本产品包含一定容量的通用寄存器 GPR(General Purpose Register)，可供用户存放数据供特殊用途。

- 电源管理域通用寄存器 PGPR，容量 128 字节，位于电源管理域，可以在系统电源域掉电时保存数据。并且不受系统电源域复位影响。
- DGO 域通用寄存器 DGO_GPR，容量 16 字节，位于 DGO 域，可以在系统电源域，电源管理域掉电时保存数据。

25.3 串行总线控制器 XPI

串行总线控制器 XPI 支持连接各类可串行访问的外部存储器，也可以与一些支持串行总线的器件互联。

XPI 支持的外部存储器包括串行 NOR Flash，串行 NAND Flash，串行 PSRAM 等，也支持 HyperBus 器件，如 HyperRAM，HyperFlash。

XPI0 的 CA 端口连接片内 1MB NOR Flash 时，支持代码在线执行 XIP。并可以通过在线解密模块 EXIP 实现代码和数据的在线解密，即数据和代码可以以密文的形式存放在片内 NOR Flash 中。

25.4 只读存储器 ROM

本产品支持 128KB 内部只读存储器 ROM。

ROM 存放本产品的启动代码，Flashloader 和部分外设驱动程序。

25.5 一次性可编程存储器 OTP

本产品支持 4096 bit 熔丝作为片上一次性可编程存储器 OTP。

OTP 用于存储以下信息：

- 产品的独特序列号 Unique ID
- 产品的启动配置信息
- 产品的安全配置信息
- 一部分安全密钥。
- 产品出厂时存放的片上模拟模块校正数据
- 供用户使用的通用数据

一次性可编程存储器 OTP 的管理由 OTP 控制器实现。用户可以通过 OTP 控制器实现对 OTP 的读和烧写操作。此外，OTP 控制器还负责实现 OTP 的读保护和写保护。

本产品上，用户在烧写 OTP 熔丝阵列时，需要打开 LDOOTP，在烧写完成后，必须关闭 LDOOTP。

26 OTP 和 OTP 控制器

本章节介绍片上一次性可编程存储器 OTP 和 OTP 控制器的主要功能和特点。

26.1 特性总结

片上一次性可编程存储器 OTP 的特性如下：

- OTP 基于熔丝阵列实现，熔丝阵列内的每个位可以烧写一次，烧写之后不可更改
- OTP 熔丝阵列包含 128 个字 (word)，每个字包含 32 位 (bit) 熔丝，容量共 4096 位
- OTP 字按功能和特性分为以下不同类型：
  - 识别（IDENTITY）：芯片的出厂信息，UUID 等信息归为识别类型，此类信息由原厂烧写，安全启动的公钥 HASH 也归为识别类型，可由客户烧写
  - 安全（SECURE）：芯片的生命周期，OTP 硬件锁定位等 OTP 字归为安全类型
  - 密钥（SECRET）：各类密钥，如 Debug Key，EXIP 的 KEK，密钥管理器的 FMK 等归为密钥类型
  - 通用（GENERAL）：用户可自由烧录通用数据，归为通用类型。保留给 BOOT ROM 使用的控制 OTP 字，也归为通用类型
- 支持 32 位 OTP 硬件锁定位，每个锁定位对应 4 个 OTP 字，结合 OTP 控制器的影子锁定位和熔丝锁定位，实现 OTP 读和烧写保护

26.1.1 OTP 控制器特性总结

本产品的一次性可编程存储器 OTP 管理由 OTP 控制器实现。用户可以通过 OTP 控制器实现对 OTP 的读和烧写操作。此外，OTP 控制器还负责实现 OTP 的读保护和写保护。

OTP 控制器在系统地址映射上分为 2 部分：

- 本体部分（内存映射表上命名为 OTP）
- 影子部分（内存映射表上命名为 OTPSHW）

OTP 控制器的特性，以及本体部分（OTP）和影子部分（OTPSHW）的不同配置如下：

- 支持 128 个影子寄存器，对应 OTP 的 128 个字
  - 本体部分（OTP）包含影子寄存器 0~15，对应 OTP 字 0~15
  - 影子部分（OTPSHW）包含影子寄存器 16~127，对应 OTP 字 16~127
- 影子锁定寄存器，OTP 每个字对应 2 位锁定位，影子锁定寄存器可配置影子寄存器的读写保护
  - 本体部分（OTP）包含影子锁定寄存器 0，对应 OTP 字 0~15
  - 影子部分（OTPSHW）包含影子锁定寄存器 1~7，对应 OTP 字 16~127
- 支持 128 个熔丝寄存器，对应 OTP 的 128 个字，熔丝寄存器是 OTP 的直接读写接口
  - 本体部分（OTP）包含熔丝寄存器 0~127，对应 OTP 字 0~127
  - 影子部分（OTPSHW）不支持熔丝寄存器
- 熔丝锁定寄存器，OTP 每个字对应 2 个锁定位，熔丝锁定寄存器可配置熔丝阵列的读写保护
  - 本体部分（OTP）包含熔丝锁定寄存器 0~7，对应 OTP 字 0~127
  - 影子部分（OTPSHW）不支持熔丝锁定寄存器
- 支持 OTP 操作引擎，支持读，写 OTP，及 OTP 影子寄存器重载，支持生成中断
  - 本体部分（OTP）的操作引擎支持 OTP 字 0~127
  - 影子部分（OTPSHW）不支持操作引擎

26.2 OTP 控制器功能描述

26.2.1 OTP 影子寄存器

OTP 控制器包含 128 个影子寄存器，每个影子寄存器对应 OTP 的一个字（Word）。OTP 控制器会在芯片从复位中释放的时候，把 OTP 的熔丝阵列的全部数据载入影子寄存器。

用户也可以通过 OTP 操作引擎，手动重载特定的 OTP 字到对应的影子寄存器，细节请查阅节 26.4。

如果某个影子寄存器没有配置读，写保护的话，用户可以任意读取并修改影子寄存器的值。注意，一些保存在 OTP 中的系统配置相关控制位，实际上是载入影子寄存器后生效的。因此，修改 OTP 影子寄存器可以在不烧录 OTP 的情况下，直接修改对应配置。比如，把 OTP 影子寄存器的 DEBUG_DISABLE 位置 1，可以立即关闭调试端口。

OTP 影子寄存器包含一套影子锁定寄存器。锁定寄存器用来配置影子寄存器的读保护，写保护以及是否允许影子寄存器从 OTP 的熔丝阵列重载对应字的值。

每个 OTP 影子寄存器（32 位）对应 2 位锁定控制位。锁定位 LOCK 位域定义如下：

- 2'b00，不锁定，不限制用户读写影子寄存器
- 2'b01，单重锁定，用户可以读影子寄存器，但不允许写影子寄存器，并且不允许再修改锁定位 LOCK 本身
- 2'b10，不锁定，不限制用户读写影子寄存器，并且不允许再修改锁定位 LOCK 本身
- 2'b11，双重锁定，用户即不允许读也不允许写影子寄存器，并且不允许再修改锁定位 LOCK 本身

对于 OTP 的不同类型字对应的影子寄存器，实际的读保护，写，重载保护效果，同时受影子锁定寄存器和 OTP 硬件锁定位影响。具体效果，请查阅节 26.5。

26.3 OTP 熔丝寄存器

OTP 控制器包含 128 个熔丝寄存器，每个熔丝寄存器对应 OTP 的一个字（Word）。熔丝寄存器实质上是 OTP 熔丝阵列的读写接口。对熔丝寄存器读操作，可以绕过 OTP 影子寄存器直接读取到 OTP 熔丝阵列对应字的值。

为了防止用户误操作，熔丝寄存器包含一个解锁寄存器 UNLOCK，对解锁寄存器写入 0x4F50454E（对应 ASCII 字符串“OPEN”），可以解锁熔丝寄存器的写保护。解锁以后，对熔丝寄存器写操作，可以直接启动对 OTP 对应字的烧录。如果用户对解锁寄存器写入 0x4F50454E 以外的任意值，会直接激活熔丝寄存器的写保护。

OTP 熔丝寄存器包含一套熔丝锁定寄存器。锁定寄存器可以配置熔丝阵列对应字的读保护和烧写保护。

每个 OTP 字（32 位）对应 2 位锁定控制位。锁定位 LOCK 位域定义如下：

- 2'b00，不锁定，不限制用户读写熔丝寄存器
- 2'b01，单重锁定，用户可以读熔丝，但不允许烧写熔丝，并且不允许再修改锁定位 LOCK 本身
- 2'b10，不锁定，不限制用户读写熔丝寄存器，并且不允许再修改锁定位 LOCK 本身
- 2'b11，双重锁定，用户即不允许读也不允许烧写熔丝，并且不允许再修改锁定位 LOCK 本身

对于 OTP 熔丝阵列内的不同类型的字，实际的读保护，烧写保护效果，同时受熔丝锁定寄存器和 OTP 硬件锁定位影响。具体效果，请查阅节 26.5。

26.4 OTP 操作引擎

OTP 控制器支持通过读写熔丝寄存器，直接读写 OTP 的熔丝阵列。不过，如果处理器直接读写熔丝寄存器的话，由于熔丝阵列本身的读写延时较长，可能会阻塞处理器工作。因此，OTP 控制器支持一套操作引擎，用户可以通过操作引擎实现 OTP 熔丝阵列的读，烧写，还可以把 OTP 熔丝阵列重载入对应的影子寄存器。

通过操作引擎读取 OTP 步骤如下:

- 对 ADDR 寄存器写入目标字的子地址，即 0 代表 Word 0，1 代表 Word 1，以此类推。
- 对 CMD 寄存器写入 0x52454144（对应 ASCII 字符串“READ”）
- 等待 INT_FLAG[READ] 标志位置 1
- 从 DATA 寄存器读取目标字的值

通过操作引擎烧录 OTP 步骤如下：

- 对 ADDR 寄存器写入目标字的子地址，即 0 代表 Word 0，1 代表 Word 1，以此类推。
- 对 DATA 寄存器写入目标字的烧录值
- 对 CMD 寄存器写入 0x424C4F57（对应 ASCII 字符串“BLOW”）
- 等待 INT_FLAG[WRITE] 标志位置 1

用户可以通过操作引擎重载 OTP 影子寄存器，操作引擎支持 4 个重载区域：

- REGION[LOAD_REGION0]，重载 OTP Word 0 ~ Word7 至 SHAWD0 ~ SHADOW7
- REGION[LOAD_REGION1]，重载 OTP Word 8 ~ Word15 至 SHAWD7 ~ SHADOW15
- REGION[LOAD_REGION2]，重载 OTP Word 16 ~ Word127 至 SHAWD16 ~ SHADOW127
- REGION[LOAD_REGION3]，用户自定义 OTP Word 的起始和结尾字地址

用户对 LOAD_REQ 寄存器的位 0~3 置 1，即可启动 LOAD_REGION0~3 的重载。

注意，在启动REGION[LOAD_REGION3]之前，用户需要配置REGION[LOAD_REGION3]寄存器的 START位域和REGION[LOAD_REGION3]寄存器的 STOP位域，来指定一段连续的 OTP 字。START位域定义了需要重载的第一个 OTP 字地址，STOP位域定义了重载的截止字地址，即重载 OTP Word (START) ~ OTP Word (STOP -1) 至 SHAWD (START) ~ SHADOW(STOP -1)。

用户可以通过LOAD_COMP寄存器，查询重载进程，对应的重载区域重载完成时，标志位置 1。

注意，OTP 操作引擎对 OTP 的读，写和重载，会受到 OTP 硬件锁定位，影子寄存器锁定位和熔丝寄存器锁定位的限制。如果对应的 OTP 字，或者 OTP 影子寄存器被读或写保护，那么 OTP 操作引擎的读，写，重载操作就不会成功。

26.5 OTP 读写保护

OTP 控制器支持对 OTP 内不同类型的字配置不同的读，写保护策略。OTP 熔丝阵列，熔丝寄存器，和影子寄存器的实际读写保护效果，同时受到 OTP 硬件锁定位，影子寄存器锁定位和熔丝寄存器锁定位的影响。

OTP 熔丝阵列的字 0，即 32 位硬件锁定位 HARD_LOCK[31:0]，烧写 HARD_LOCK 可以对 OTP 的指定区域加以读或写保护。把 HARD_LOCK 中的若干位烧写为 1，就对对应的 OTP 区域加以硬件锁定。

注意，OTP 支持 32 位硬件锁定位 HARD_LOCK[31:0]，每位对应 OTP 的 4 个字。OTP 熔丝寄存器和影子寄存器的锁定位，每 2 位对应 OTP 的一个字或一个 32 位影子寄存器。

全部 OTP 数据按字可分为以下 4 个类型，不同类型的 OTP 字，设置对应的硬件锁定位后，读保护或写保护的效果有所不同：

- 识别（IDENTITY）
- 安全（SECURE）

- 密钥（SECRET）
- 通用（GENERAL）

用户可以查阅 OTP 映射表，了解 OTP 每个字的类型归属。

总结 OTP 的读写保护策略如下：

- 影子寄存器锁定位的效果
  - 影子寄存器的锁定控制位设置为不锁定时，影子寄存器可以自由读写，或者重载 OTP 阵列的值到影子寄存器
  - 影子寄存器的锁定控制位设置为单重锁定时，影子寄存器被写保护，不允许重载 OTP 阵列的值到影子寄存器
  - 影子寄存器的锁定控制位设置为双重锁定时，影子寄存器被读保护和写保护，不允许重载 OTP 阵列的值到影子寄存器
- 熔丝寄存器锁定位的效果
  - 熔丝寄存器的锁定控制位设置为不锁定时，熔丝寄存器可以自由读写
  - 熔丝寄存器的锁定控制位设置为单重锁定时，熔丝阵列被写保护，即不允许通过写熔丝寄存器或者 OTP 操作引擎烧写熔丝阵列
  - 熔丝寄存器的锁定控制位设置为双重锁定时，熔丝阵列被读保护和写保护，即不允许通过熔丝寄存器或者 OTP 操作引擎读取或者烧写熔丝阵列
- OTP 硬件锁定位的效果
  - 识别类型的 OTP 字，当其对应的硬件锁定位为 1 时，只施加写保护，即允许通过熔丝寄存器或者 OTP 操作引擎读取该 OTP 字，也允许读取该 OTP 字对应的影子寄存器。但是不允许通过熔丝寄存器或者 OTP 操作引擎烧写该 OTP 字，也不允许写或者重载该 OTP 字对应的影子寄存器。
  - 安全类型的 OTP 字，当其对应的硬件锁定位为 1 时，只施加写保护，即允许通过熔丝寄存器或者 OTP 操作引擎读取该 OTP 字，也允许读取该 OTP 字对应的影子寄存器。但是不允许通过熔丝寄存器或者 OTP 操作引擎烧写该 OTP 字，也不允许写或者重载该 OTP 字对应的影子寄存器。注意，安全类型 OTP 字中，对应产品生命周期的影子寄存器为例外，允许用户写该影子寄存器，使当前生命周期前进（把产品生命周期的位由 0 置 1），但不允许生命周期回退（把产品生命周期的位由 1 清 0）
  - 密钥类型的 OTP 字，当其对应的硬件锁定位为 1 时，同时施加读保护和写保护，即不允许通过熔丝寄存器或者 OTP 操作引擎读取该 OTP 字，也不允许读取该 OTP 字对应的影子寄存器。也不允许通过熔丝寄存器或者 OTP 操作引擎烧写该 OTP 字，也不允许写或者重载该 OTP 字对应的影子寄存器。
  - 通用类型的 OTP 字，当其对应的硬件锁定位为 1 时，只对熔丝阵列施加写保护，即允许通过熔丝寄存器或者 OTP 操作引擎读取该 OTP 字，也允许读取该 OTP 字对应的影子寄存器。不允许通过熔丝寄存器或者 OTP 操作引擎烧写该 OTP 字，但是允许写或者重载该 OTP 字对应的影子寄存器。
- 对于 OTP 阵列中某个字，如果硬件锁定位和熔丝锁定位配置的读保护或者烧写保护冲突，则保护总是生效。同样的，对于某个 OTP 影子寄存器，如果硬件锁定位和影子锁定位配置的读保护或者写保护冲突，则保护总是生效。

以下表格汇总了 OTP 不同类型字的读写保护效果：

识别类型	硬件锁定	熔丝单重锁定	熔丝双重锁定	影子单重锁定	影子双重锁定
读熔丝	Y	Y	N	Y	Y
写熔丝	N	N	N	Y	Y
影子寄存器重载	Y	Y	Y	N	N
读影子寄存器	Y	Y	Y	Y	N
写影子寄存器	N	Y	Y	N	N

表 156: 识别类型 OTP 字读写保护总结

安全类型	硬件锁定	熔丝单重锁定	熔丝双重锁定	影子单重锁定	影子双重锁定
读熔丝	Y	Y	N	Y	Y
写熔丝	N	N	N	Y	Y
影子寄存器重载	Y	Y	Y	N	N
读影子寄存器	Y	Y	Y	Y	N
写影子寄存器	N	Y	Y	N	N

表 157: 安全类型 OTP 字读写保护总结

密钥类型	硬件锁定	熔丝单重锁定	熔丝双重锁定	影子单重锁定	影子双重锁定
读熔丝	N	Y	N	Y	Y
写熔丝	N	N	N	Y	Y
影子寄存器重载	Y	Y	Y	N	N
读影子寄存器	N	Y	Y	Y	N
写影子寄存器	N	Y	Y	N	N

表 158: 密钥类型 OTP 字读写保护总结

通用类型	硬件锁定	熔丝单重锁定	熔丝双重锁定	影子单重锁定	影子双重锁定
读熔丝	Y	Y	N	Y	Y
写熔丝	N	N	N	Y	Y
影子寄存器重载	Y	Y	Y	N	N
读影子寄存器	Y	Y	Y	Y	N
写影子寄存器	Y	Y	Y	N	N

表 159: 通用类型 OTP 字读写保护总结

26.6 OTP 控制器寄存器

OTP 的寄存器列表如下：

OTP base address: 0xF3050000

地址偏移	名称	描述	复位值
0x0000	SHADOW[SHADOW000]	熔丝加载寄存器	0x00000000
0x0004	SHADOW[SHADOW001]	熔丝加载寄存器	0x00000000
0x0008	SHADOW[SHADOW002]	熔丝加载寄存器	0x00000000
0x000C	SHADOW[SHADOW003]	熔丝加载寄存器	0x00000000

地址偏移	名称	描述	复位值
0x0010	SHADOW[SHADOW004]	熔丝加载寄存器	0x00000000
0x0014	SHADOW[SHADOW005]	熔丝加载寄存器	0x00000000
0x0018	SHADOW[SHADOW006]	熔丝加载寄存器	0x00000000
0x001C	SHADOW[SHADOW007]	熔丝加载寄存器	0x00000000
0x0020	SHADOW[SHADOW008]	熔丝加载寄存器	0x00000000
0x0024	SHADOW[SHADOW009]	熔丝加载寄存器	0x00000000
0x0028	SHADOW[SHADOW010]	熔丝加载寄存器	0x00000000
0x002C	SHADOW[SHADOW011]	熔丝加载寄存器	0x00000000
0x0030	SHADOW[SHADOW012]	熔丝加载寄存器	0x00000000
0x0034	SHADOW[SHADOW013]	熔丝加载寄存器	0x00000000
0x0038	SHADOW[SHADOW014]	熔丝加载寄存器	0x00000000
0x003C	SHADOW[SHADOW015]	熔丝加载寄存器	0x00000000
0x0040	SHADOW[SHADOW016]	熔丝加载寄存器	0x00000000
0x0044	SHADOW[SHADOW017]	熔丝加载寄存器	0x00000000
0x0048	SHADOW[SHADOW018]	熔丝加载寄存器	0x00000000
0x004C	SHADOW[SHADOW019]	熔丝加载寄存器	0x00000000
0x0050	SHADOW[SHADOW020]	熔丝加载寄存器	0x00000000
0x0054	SHADOW[SHADOW021]	熔丝加载寄存器	0x00000000
0x0058	SHADOW[SHADOW022]	熔丝加载寄存器	0x00000000
0x005C	SHADOW[SHADOW023]	熔丝加载寄存器	0x00000000
0x0060	SHADOW[SHADOW024]	熔丝加载寄存器	0x00000000
0x0064	SHADOW[SHADOW025]	熔丝加载寄存器	0x00000000
0x0068	SHADOW[SHADOW026]	熔丝加载寄存器	0x00000000
0x006C	SHADOW[SHADOW027]	熔丝加载寄存器	0x00000000
0x0070	SHADOW[SHADOW028]	熔丝加载寄存器	0x00000000
0x0074	SHADOW[SHADOW029]	熔丝加载寄存器	0x00000000
0x0078	SHADOW[SHADOW030]	熔丝加载寄存器	0x00000000
0x007C	SHADOW[SHADOW031]	熔丝加载寄存器	0x00000000
0x0080	SHADOW[SHADOW032]	熔丝加载寄存器	0x00000000
0x0084	SHADOW[SHADOW033]	熔丝加载寄存器	0x00000000
0x0088	SHADOW[SHADOW034]	熔丝加载寄存器	0x00000000
0x008C	SHADOW[SHADOW035]	熔丝加载寄存器	0x00000000
0x0090	SHADOW[SHADOW036]	熔丝加载寄存器	0x00000000
0x0094	SHADOW[SHADOW037]	熔丝加载寄存器	0x00000000
0x0098	SHADOW[SHADOW038]	熔丝加载寄存器	0x00000000
0x009C	SHADOW[SHADOW039]	熔丝加载寄存器	0x00000000
0x00A0	SHADOW[SHADOW040]	熔丝加载寄存器	0x00000000
0x00A4	SHADOW[SHADOW041]	熔丝加载寄存器	0x00000000
0x00A8	SHADOW[SHADOW042]	熔丝加载寄存器	0x00000000
0x00AC	SHADOW[SHADOW043]	熔丝加载寄存器	0x00000000

地址偏移	名称	描述	复位值
0x00B0	SHADOW[SHADOW044]	熔丝加载寄存器	0x00000000
0x00B4	SHADOW[SHADOW045]	熔丝加载寄存器	0x00000000
0x00B8	SHADOW[SHADOW046]	熔丝加载寄存器	0x00000000
0x00BC	SHADOW[SHADOW047]	熔丝加载寄存器	0x00000000
0x00C0	SHADOW[SHADOW048]	熔丝加载寄存器	0x00000000
0x00C4	SHADOW[SHADOW049]	熔丝加载寄存器	0x00000000
0x00C8	SHADOW[SHADOW050]	熔丝加载寄存器	0x00000000
0x00CC	SHADOW[SHADOW051]	熔丝加载寄存器	0x00000000
0x00D0	SHADOW[SHADOW052]	熔丝加载寄存器	0x00000000
0x00D4	SHADOW[SHADOW053]	熔丝加载寄存器	0x00000000
0x00D8	SHADOW[SHADOW054]	熔丝加载寄存器	0x00000000
0x00DC	SHADOW[SHADOW055]	熔丝加载寄存器	0x00000000
0x00E0	SHADOW[SHADOW056]	熔丝加载寄存器	0x00000000
0x00E4	SHADOW[SHADOW057]	熔丝加载寄存器	0x00000000
0x00E8	SHADOW[SHADOW058]	熔丝加载寄存器	0x00000000
0x00EC	SHADOW[SHADOW059]	熔丝加载寄存器	0x00000000
0x00F0	SHADOW[SHADOW060]	熔丝加载寄存器	0x00000000
0x00F4	SHADOW[SHADOW061]	熔丝加载寄存器	0x00000000
0x00F8	SHADOW[SHADOW062]	熔丝加载寄存器	0x00000000
0x00FC	SHADOW[SHADOW063]	熔丝加载寄存器	0x00000000
0x0100	SHADOW[SHADOW064]	熔丝加载寄存器	0x00000000
0x0104	SHADOW[SHADOW065]	熔丝加载寄存器	0x00000000
0x0108	SHADOW[SHADOW066]	熔丝加载寄存器	0x00000000
0x010C	SHADOW[SHADOW067]	熔丝加载寄存器	0x00000000
0x0110	SHADOW[SHADOW068]	熔丝加载寄存器	0x00000000
0x0114	SHADOW[SHADOW069]	熔丝加载寄存器	0x00000000
0x0118	SHADOW[SHADOW070]	熔丝加载寄存器	0x00000000
0x011C	SHADOW[SHADOW071]	熔丝加载寄存器	0x00000000
0x0120	SHADOW[SHADOW072]	熔丝加载寄存器	0x00000000
0x0124	SHADOW[SHADOW073]	熔丝加载寄存器	0x00000000
0x0128	SHADOW[SHADOW074]	熔丝加载寄存器	0x00000000
0x012C	SHADOW[SHADOW075]	熔丝加载寄存器	0x00000000
0x0130	SHADOW[SHADOW076]	熔丝加载寄存器	0x00000000
0x0134	SHADOW[SHADOW077]	熔丝加载寄存器	0x00000000
0x0138	SHADOW[SHADOW078]	熔丝加载寄存器	0x00000000
0x013C	SHADOW[SHADOW079]	熔丝加载寄存器	0x00000000
0x0140	SHADOW[SHADOW080]	熔丝加载寄存器	0x00000000
0x0144	SHADOW[SHADOW081]	熔丝加载寄存器	0x00000000
0x0148	SHADOW[SHADOW082]	熔丝加载寄存器	0x00000000
0x014C	SHADOW[SHADOW083]	熔丝加载寄存器	0x00000000

地址偏移	名称	描述	复位值
0x0150	SHADOW[SHADOW084]	熔丝加载寄存器	0x00000000
0x0154	SHADOW[SHADOW085]	熔丝加载寄存器	0x00000000
0x0158	SHADOW[SHADOW086]	熔丝加载寄存器	0x00000000
0x015C	SHADOW[SHADOW087]	熔丝加载寄存器	0x00000000
0x0160	SHADOW[SHADOW088]	熔丝加载寄存器	0x00000000
0x0164	SHADOW[SHADOW089]	熔丝加载寄存器	0x00000000
0x0168	SHADOW[SHADOW090]	熔丝加载寄存器	0x00000000
0x016C	SHADOW[SHADOW091]	熔丝加载寄存器	0x00000000
0x0170	SHADOW[SHADOW092]	熔丝加载寄存器	0x00000000
0x0174	SHADOW[SHADOW093]	熔丝加载寄存器	0x00000000
0x0178	SHADOW[SHADOW094]	熔丝加载寄存器	0x00000000
0x017C	SHADOW[SHADOW095]	熔丝加载寄存器	0x00000000
0x0180	SHADOW[SHADOW096]	熔丝加载寄存器	0x00000000
0x0184	SHADOW[SHADOW097]	熔丝加载寄存器	0x00000000
0x0188	SHADOW[SHADOW098]	熔丝加载寄存器	0x00000000
0x018C	SHADOW[SHADOW099]	熔丝加载寄存器	0x00000000
0x0190	SHADOW[SHADOW100]	熔丝加载寄存器	0x00000000
0x0194	SHADOW[SHADOW101]	熔丝加载寄存器	0x00000000
0x0198	SHADOW[SHADOW102]	熔丝加载寄存器	0x00000000
0x019C	SHADOW[SHADOW103]	熔丝加载寄存器	0x00000000
0x01A0	SHADOW[SHADOW104]	熔丝加载寄存器	0x00000000
0x01A4	SHADOW[SHADOW105]	熔丝加载寄存器	0x00000000
0x01A8	SHADOW[SHADOW106]	熔丝加载寄存器	0x00000000
0x01AC	SHADOW[SHADOW107]	熔丝加载寄存器	0x00000000
0x01B0	SHADOW[SHADOW108]	熔丝加载寄存器	0x00000000
0x01B4	SHADOW[SHADOW109]	熔丝加载寄存器	0x00000000
0x01B8	SHADOW[SHADOW110]	熔丝加载寄存器	0x00000000
0x01BC	SHADOW[SHADOW111]	熔丝加载寄存器	0x00000000
0x01C0	SHADOW[SHADOW112]	熔丝加载寄存器	0x00000000
0x01C4	SHADOW[SHADOW113]	熔丝加载寄存器	0x00000000
0x01C8	SHADOW[SHADOW114]	熔丝加载寄存器	0x00000000
0x01CC	SHADOW[SHADOW115]	熔丝加载寄存器	0x00000000
0x01D0	SHADOW[SHADOW116]	熔丝加载寄存器	0x00000000
0x01D4	SHADOW[SHADOW117]	熔丝加载寄存器	0x00000000
0x01D8	SHADOW[SHADOW118]	熔丝加载寄存器	0x00000000
0x01DC	SHADOW[SHADOW119]	熔丝加载寄存器	0x00000000
0x01E0	SHADOW[SHADOW120]	熔丝加载寄存器	0x00000000
0x01E4	SHADOW[SHADOW121]	熔丝加载寄存器	0x00000000
0x01E8	SHADOW[SHADOW122]	熔丝加载寄存器	0x00000000
0x01EC	SHADOW[SHADOW123]	熔丝加载寄存器	0x00000000

地址偏移	名称	描述	复位值
0x01F0	SHADOW[SHADOW124]	熔丝加载寄存器	0x00000000
0x01F4	SHADOW[SHADOW125]	熔丝加载寄存器	0x00000000
0x01F8	SHADOW[SHADOW126]	熔丝加载寄存器	0x00000000
0x01FC	SHADOW[SHADOW127]	熔丝加载寄存器	0x00000000
0x0200	SHADOW_LOCK[LOCK00]	加载寄存器锁定	0x00000000
0x0204	SHADOW_LOCK[LOCK01]	加载寄存器锁定	0x00000000
0x0208	SHADOW_LOCK[LOCK02]	加载寄存器锁定	0x00000000
0x020C	SHADOW_LOCK[LOCK03]	加载寄存器锁定	0x00000000
0x0210	SHADOW_LOCK[LOCK04]	加载寄存器锁定	0x00000000
0x0214	SHADOW_LOCK[LOCK05]	加载寄存器锁定	0x00000000
0x0218	SHADOW_LOCK[LOCK06]	加载寄存器锁定	0x00000000
0x021C	SHADOW_LOCK[LOCK07]	加载寄存器锁定	0x00000000
0x0400	FUSE[FUSE000]	熔丝	0x00000000
0x0404	FUSE[FUSE001]	熔丝	0x00000000
0x0408	FUSE[FUSE002]	熔丝	0x00000000
0x040C	FUSE[FUSE003]	熔丝	0x00000000
0x0410	FUSE[FUSE004]	熔丝	0x00000000
0x0414	FUSE[FUSE005]	熔丝	0x00000000
0x0418	FUSE[FUSE006]	熔丝	0x00000000
0x041C	FUSE[FUSE007]	熔丝	0x00000000
0x0420	FUSE[FUSE008]	熔丝	0x00000000
0x0424	FUSE[FUSE009]	熔丝	0x00000000
0x0428	FUSE[FUSE010]	熔丝	0x00000000
0x042C	FUSE[FUSE011]	熔丝	0x00000000
0x0430	FUSE[FUSE012]	熔丝	0x00000000
0x0434	FUSE[FUSE013]	熔丝	0x00000000
0x0438	FUSE[FUSE014]	熔丝	0x00000000
0x043C	FUSE[FUSE015]	熔丝	0x00000000
0x0440	FUSE[FUSE016]	熔丝	0x00000000
0x0444	FUSE[FUSE017]	熔丝	0x00000000
0x0448	FUSE[FUSE018]	熔丝	0x00000000
0x044C	FUSE[FUSE019]	熔丝	0x00000000
0x0450	FUSE[FUSE020]	熔丝	0x00000000
0x0454	FUSE[FUSE021]	熔丝	0x00000000
0x0458	FUSE[FUSE022]	熔丝	0x00000000
0x045C	FUSE[FUSE023]	熔丝	0x00000000
0x0460	FUSE[FUSE024]	熔丝	0x00000000
0x0464	FUSE[FUSE025]	熔丝	0x00000000
0x0468	FUSE[FUSE026]	熔丝	0x00000000
0x046C	FUSE[FUSE027]	熔丝	0x00000000

地址偏移	名称	描述	复位值
0x0470	FUSE[FUSE028]	熔丝	0x00000000
0x0474	FUSE[FUSE029]	熔丝	0x00000000
0x0478	FUSE[FUSE030]	熔丝	0x00000000
0x047C	FUSE[FUSE031]	熔丝	0x00000000
0x0480	FUSE[FUSE032]	熔丝	0x00000000
0x0484	FUSE[FUSE033]	熔丝	0x00000000
0x0488	FUSE[FUSE034]	熔丝	0x00000000
0x048C	FUSE[FUSE035]	熔丝	0x00000000
0x0490	FUSE[FUSE036]	熔丝	0x00000000
0x0494	FUSE[FUSE037]	熔丝	0x00000000
0x0498	FUSE[FUSE038]	熔丝	0x00000000
0x049C	FUSE[FUSE039]	熔丝	0x00000000
0x04A0	FUSE[FUSE040]	熔丝	0x00000000
0x04A4	FUSE[FUSE041]	熔丝	0x00000000
0x04A8	FUSE[FUSE042]	熔丝	0x00000000
0x04AC	FUSE[FUSE043]	熔丝	0x00000000
0x04B0	FUSE[FUSE044]	熔丝	0x00000000
0x04B4	FUSE[FUSE045]	熔丝	0x00000000
0x04B8	FUSE[FUSE046]	熔丝	0x00000000
0x04BC	FUSE[FUSE047]	熔丝	0x00000000
0x04C0	FUSE[FUSE048]	熔丝	0x00000000
0x04C4	FUSE[FUSE049]	熔丝	0x00000000
0x04C8	FUSE[FUSE050]	熔丝	0x00000000
0x04CC	FUSE[FUSE051]	熔丝	0x00000000
0x04D0	FUSE[FUSE052]	熔丝	0x00000000
0x04D4	FUSE[FUSE053]	熔丝	0x00000000
0x04D8	FUSE[FUSE054]	熔丝	0x00000000
0x04DC	FUSE[FUSE055]	熔丝	0x00000000
0x04E0	FUSE[FUSE056]	熔丝	0x00000000
0x04E4	FUSE[FUSE057]	熔丝	0x00000000
0x04E8	FUSE[FUSE058]	熔丝	0x00000000
0x04EC	FUSE[FUSE059]	熔丝	0x00000000
0x04F0	FUSE[FUSE060]	熔丝	0x00000000
0x04F4	FUSE[FUSE061]	熔丝	0x00000000
0x04F8	FUSE[FUSE062]	熔丝	0x00000000
0x04FC	FUSE[FUSE063]	熔丝	0x00000000
0x0500	FUSE[FUSE064]	熔丝	0x00000000
0x0504	FUSE[FUSE065]	熔丝	0x00000000
0x0508	FUSE[FUSE066]	熔丝	0x00000000
0x050C	FUSE[FUSE067]	熔丝	0x00000000

地址偏移	名称	描述	复位值
0x0510	FUSE[FUSE068]	熔丝	0x00000000
0x0514	FUSE[FUSE069]	熔丝	0x00000000
0x0518	FUSE[FUSE070]	熔丝	0x00000000
0x051C	FUSE[FUSE071]	熔丝	0x00000000
0x0520	FUSE[FUSE072]	熔丝	0x00000000
0x0524	FUSE[FUSE073]	熔丝	0x00000000
0x0528	FUSE[FUSE074]	熔丝	0x00000000
0x052C	FUSE[FUSE075]	熔丝	0x00000000
0x0530	FUSE[FUSE076]	熔丝	0x00000000
0x0534	FUSE[FUSE077]	熔丝	0x00000000
0x0538	FUSE[FUSE078]	熔丝	0x00000000
0x053C	FUSE[FUSE079]	熔丝	0x00000000
0x0540	FUSE[FUSE080]	熔丝	0x00000000
0x0544	FUSE[FUSE081]	熔丝	0x00000000
0x0548	FUSE[FUSE082]	熔丝	0x00000000
0x054C	FUSE[FUSE083]	熔丝	0x00000000
0x0550	FUSE[FUSE084]	熔丝	0x00000000
0x0554	FUSE[FUSE085]	熔丝	0x00000000
0x0558	FUSE[FUSE086]	熔丝	0x00000000
0x055C	FUSE[FUSE087]	熔丝	0x00000000
0x0560	FUSE[FUSE088]	熔丝	0x00000000
0x0564	FUSE[FUSE089]	熔丝	0x00000000
0x0568	FUSE[FUSE090]	熔丝	0x00000000
0x056C	FUSE[FUSE091]	熔丝	0x00000000
0x0570	FUSE[FUSE092]	熔丝	0x00000000
0x0574	FUSE[FUSE093]	熔丝	0x00000000
0x0578	FUSE[FUSE094]	熔丝	0x00000000
0x057C	FUSE[FUSE095]	熔丝	0x00000000
0x0580	FUSE[FUSE096]	熔丝	0x00000000
0x0584	FUSE[FUSE097]	熔丝	0x00000000
0x0588	FUSE[FUSE098]	熔丝	0x00000000
0x058C	FUSE[FUSE099]	熔丝	0x00000000
0x0590	FUSE[FUSE100]	熔丝	0x00000000
0x0594	FUSE[FUSE101]	熔丝	0x00000000
0x0598	FUSE[FUSE102]	熔丝	0x00000000
0x059C	FUSE[FUSE103]	熔丝	0x00000000
0x05A0	FUSE[FUSE104]	熔丝	0x00000000
0x05A4	FUSE[FUSE105]	熔丝	0x00000000
0x05A8	FUSE[FUSE106]	熔丝	0x00000000
0x05AC	FUSE[FUSE107]	熔丝	0x00000000

地址偏移	名称	描述	复位值
0x05B0	FUSE[FUSE108]	熔丝	0x00000000
0x05B4	FUSE[FUSE109]	熔丝	0x00000000
0x05B8	FUSE[FUSE110]	熔丝	0x00000000
0x05BC	FUSE[FUSE111]	熔丝	0x00000000
0x05C0	FUSE[FUSE112]	熔丝	0x00000000
0x05C4	FUSE[FUSE113]	熔丝	0x00000000
0x05C8	FUSE[FUSE114]	熔丝	0x00000000
0x05CC	FUSE[FUSE115]	熔丝	0x00000000
0x05D0	FUSE[FUSE116]	熔丝	0x00000000
0x05D4	FUSE[FUSE117]	熔丝	0x00000000
0x05D8	FUSE[FUSE118]	熔丝	0x00000000
0x05DC	FUSE[FUSE119]	熔丝	0x00000000
0x05E0	FUSE[FUSE120]	熔丝	0x00000000
0x05E4	FUSE[FUSE121]	熔丝	0x00000000
0x05E8	FUSE[FUSE122]	熔丝	0x00000000
0x05EC	FUSE[FUSE123]	熔丝	0x00000000
0x05F0	FUSE[FUSE124]	熔丝	0x00000000
0x05F4	FUSE[FUSE125]	熔丝	0x00000000
0x05F8	FUSE[FUSE126]	熔丝	0x00000000
0x05FC	FUSE[FUSE127]	熔丝	0x00000000
0x0600	FUSE_LOCK[LOCK00]	Fuse lock	0x00000000
0x0604	FUSE_LOCK[LOCK01]	Fuse lock	0x00000000
0x0608	FUSE_LOCK[LOCK02]	Fuse lock	0x00000000
0x060C	FUSE_LOCK[LOCK03]	Fuse lock	0x00000000
0x0610	FUSE_LOCK[LOCK04]	Fuse lock	0x00000000
0x0614	FUSE_LOCK[LOCK05]	Fuse lock	0x00000000
0x0618	FUSE_LOCK[LOCK06]	Fuse lock	0x00000000
0x061C	FUSE_LOCK[LOCK07]	Fuse lock	0x00000000
0x0800	UNLOCK	UNLOCK	0x00000000
0x0804	DATA	DATA	0x00000000
0x0808	ADDR	ADDR	0x00000000
0x080C	CMD	CMD	0x00000000
0x0A00	LOAD_REQ	LOAD Request	0x00000007
0x0A04	LOAD_COMP	LOAD complete	0x00000007
0x0A20	REGION[LOAD_REGION0]	LOAD region	0x00000800
0x0A24	REGION[LOAD_REGION1]	LOAD region	0x00001008
0x0A28	REGION[LOAD_REGION2]	LOAD region	0x00000010
0x0A2C	REGION[LOAD_REGION3]	LOAD region	0x00000000
0x0C00	INT_FLAG	interrupt flag	0x00000000
0x0C04	INT_EN	interrupt enable	0x00000000

地址偏移	名称	描述	复位值

表 160: OTP 寄存器列表

26.7 OTP 控制器寄存器详细信息

OTP 的寄存器详细说明如下：

26.7.1 SHADOW (0x0 + 0x4 * n)

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

SHADOW [31:0]

位域	名称	描述
31-0	SHADOW	熔丝加载寄存器 电源域 0-15 有效，数字域 16-128 有效

SHADOW 位域

26.7.2 SHADOW_LOCK (0x200 + 0x4 * n)

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

SHADOW_LOCK [31:0]

位域	名称	描述
31-0	LOCK	加载寄存器锁定，每两位控制一个 32 位字，只有该字段为 0 的时候可以写入，不同类型的熔丝的锁定行为具有差异 00: 未锁定 01: 锁定 10: 不锁定，禁止锁定 11: 双重锁定

SHADOW_LOCK 位域

26.7.3 FUSE (0x400 + 0x4 * n)

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

FUSE [31:0]

位域	名称	描述
31-0	FUSE	熔丝，仅电源域有效 写操作可读出熔丝值 写操作写入熔丝 (写入前需并开启 2.5V 电源，并对熔丝解锁)

FUSE 位域

26.7.4 FUSE_LOCK (0x600 + 0x4 * n)

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

FUSE_LOCK [31:0]

位域	名称	描述
31-0	LOCK	熔丝锁定，每两位控制一个 32 位字，只有该字段为 0 的时候可以写入，不同类型的熔丝的锁定行为具有差异 00: 未锁定 01: 锁定 10: 不锁定，禁止锁定 11: 双重锁定

FUSE_LOCK 位域

26.7.5 UNLOCK (0x800)

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

UNLOCK [31:0]

位域	名称	描述
31-0	UNLOCK	解锁熔丝写入功能 写入"OPEN" 可解锁，写入其他值将锁定熔丝 写入熔丝前需打开 2.5V 电源，24M OSC

UNLOCK 位域

26.7.6 DATA (0x804)

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

DATA [31:0]

位域	名称	描述
31-0	DATA	非阻塞工作模式下的读写数据

DATA 位域

26.7.7 ADDR (0x808)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																									ADDR						
N/A																									RW						
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0

ADDR [31:0]

位域	名称	描述
6-0	ADDR	读写的 32 位字地址

ADDR 位域

26.7.8 CMD (0x80C)

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

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CMD [31:0]																															

位域	名称	描述
31-0	CMD	熔丝命令 ”BLOW” 将 DATA 寄存器写入位于 ADDR 的字 ”READ” 将位于 ADDR 的字读出到 DATA 寄存器

CMD 位域

26.7.9 LOAD_REQ (0xA00)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																												REQUEST			
N/A																												RW			
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	1	1	1

LOAD_REQ [31:0]

位域	名称	描述
3-0	REQUEST	4 个区域的加载请求 bit0: 区域 0 bit1: 区域 1 bit2: 区域 2 bit3: 区域 3

LOAD_REQ 位域

26.7.10 LOAD_COMP (0xA04)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																												COMPLETE			
N/A																												RW			
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	1	1	1

LOAD_COMP [31:0]

位域	名称	描述
3-0	COMPLETE	4 个区域的加载完成标志，写 1 清零 bit0: 区域 0 bit1: 区域 1 bit2: 区域 2 bit3: 区域 3

LOAD_COMP 位域

26.7.11 REGION (0xA20 + 0x4 * n)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																	STOP							RSVD	START						
N/A																	RW							N/A	RW						
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	x	0	0	0	0	0	0	0

REGION [31:0]

位域	名称	描述
14-8	STOP	加载区域的末尾地址，末尾地址不会被加载
6-0	START	加载区域的开始地址，开始地址会被加载

REGION 位域

26.7.12 INT_FLAG (0xC00)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																													WRITE	READ	LOAD
N/A																													RW	RW	RW
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0

INT_FLAG [31:0]

位域	名称	描述
2	WRITE	熔丝写入完成标志 0: 未写入或写入未完成 1: 写入完成
1	READ	熔丝读取完成标志 0: 未读取或读取未完成 1: 读取完成

位域	名称	描述
0	LOAD	熔丝加载完成标志 0: 未加载或未完成 1: 加载完成

INT_FLAG 位域

26.7.13 INT_EN (0xC04)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																													WRITE	READ	LOAD
N/A																													RW	RW	RW
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0

INT_EN [31:0]

位域	名称	描述
2	WRITE	熔丝写入完成中断使能 0: 禁止 1: 使能
1	READ	熔丝读取完成中断使能 0: 禁止 1: 使能
0	LOAD	熔丝加载完成中断使能 0: 禁止 1: 使能

INT_EN 位域

27 DMA，DMAMUX 和信箱概述

本章节介绍了 DMA 控制器，DMAMUX 和通讯信箱 MBX。

27.1 DMA 控制器

本产品包括 1 个 DMA 控制器。

- HDMA，作为主设备连接到 AHB 外设总线。它可以实现实时的外设寄存器与内存，内存之间的数据搬移。

HDMA 有 32 个 DMA 请求输入信号，这些 DMA 的请求源来自 DMA 请求路由器（DMAMUX）。

27.2 DMA 请求路由器

DMA 请求路由器 (DMAMUX)。将来自各个外设模块的 DMA 请求分配到 32 输出信号，作为 HDMA 的 DMA 传输请求源。

用户可以通过配置 DMAMUX 寄存器，把来自特定外设的 DMA 请求，连接到 HDMA 的 32 个通道。

DMAMUX 的输入具体分配如下：

DMA MUX 编号	DMA MUX SOURCE 名称	描述
0	GPTMR0_0	
1	GPTMR0_1	
2	GPTMR0_2	
3	GPTMR0_3	
4	GPTMR1_0	
5	GPTMR1_1	
6	GPTMR1_2	
7	GPTMR1_3	
8	GPTMR2_0	
9	GPTMR2_1	
10	GPTMR2_2	
11	GPTMR2_3	
12	GPTMR3_0	
13	GPTMR3_1	
14	GPTMR3_2	
15	GPTMR3_3	
16	LIN0	
17	LIN1	
18	LIN2	
19	LIN3	
20	UART0_RX	
21	UART0_TX	
22	UART1_RX	
23	UART1_TX	
24	UART2_RX	

DMA MUX 编号	DMA MUX SOURCE 名称	描述
25	UART2_TX	
26	UART3_RX	
27	UART3_TX	
28	UART4_RX	
29	UART4_TX	
30	UART5_RX	
31	UART5_TX	
32	UART6_RX	
33	UART6_TX	
34	UART7_RX	
35	UART7_TX	
36	I2C0	
37	I2C1	
38	I2C2	
39	I2C3	
40	SPI0_RX	
41	SPI0_TX	
42	SPI1_RX	
43	SPI1_TX	
44	SPI2_RX	
45	SPI2_TX	
46	SPI3_RX	
47	SPI3_TX	
48	CAN0	
49	CAN1	
50	CAN2	
51	CAN3	
52	MOT_0	
53	MOT_1	
54	MOT_2	
55	MOT_3	
56	MOT_4	
57	MOT_5	
58	MOT_6	
59	MOT_7	
60	XPI0_RX	
61	XPI0_TX	
62	DAC0	
63	DAC1	
64	ACMP0	

DMA MUX 编号	DMA MUX SOURCE 名称	描述
65	ACMP1	

表 161: DMA MUX 列表

27.3 通讯信箱 MBX

本产品支持 1 个通讯信箱 MBX，用于进程间通信。处理器可以利用 MBX 互相发送数据，MBX 支持生成中断。

28 DMA 控制器

本章节介绍 DMA 控制器的功能和特性。

28.1 概述

DMA 控制器的主要特性有：

- 支持 32 个可配置的通道
- 通道支持 2 级优先级配置
- 相同优先级通道使用 Round-Robin 仲裁
- 支持链式连接多个 DMA 任务

28.2 DMA 架构框图

本模块的结构图如图 23所示：

DMA 模块中的 dma_apb_csr 为控制寄存器模块，dma_ahb_ahbmst 模块实现了 ahb master 接口，DMA_ahb_engine

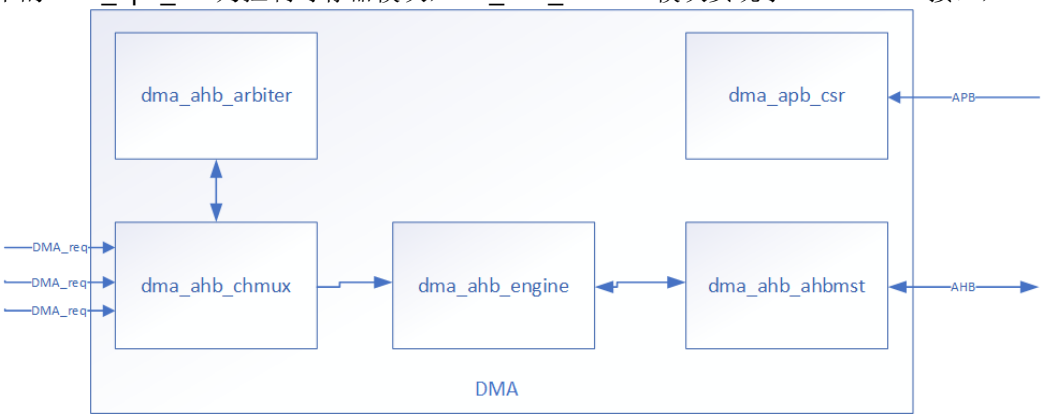


图 23: DMA 结构框图

模块实现了 DMA 操作引擎状态机，dma_ahb_arbiter 和 dma_ahb_chmux 模块实现多路 DMA 请求的仲裁与选择。

28.3 管脚说明

表 162: DMA 管脚说明

管脚	方向	功能描述
dma_req[31:0]	输入	输入的 DMA 请求信号
dma_ack[31:0]	输出	输出的 DMA 应答信号

28.4 功能说明

DMA 控制器支持 32 个通道，每个通道都可独立配置数据传输的参数。用户可以同时使用多个 DMA 通道，DMA 控制器会交替执行不同通道的数据传输任务，而不是等待一个通道传输完成后再响应下一个通道的传输请求。也就是说 DMA 控制器可能分多次完成单个通道的数据传输，在这个通道全部任务完成后，将状态寄存器内的相应标志位置位。

某个 DMA 通道的数据传输可能因为数据传输错误而中断，也可以由软件中止。此种情况下，DMA 控制器会

停止传输数据，并将状态寄存器内的相应标志位置位。

28.4.1 DMA 硬件握手

DMA 控制器支持 32 对请求-响应的握手信号，请求信号来自功能模块并经由 DMAMUX 路由至 DMA 控制器，DMA 控制器根据配置使用某个通道执行将该请求，完成长度为 SrcBurstSize*SrcWidth 的数据传输后发送响应信号给 DMAMUX，完成硬件握手。

28.4.2 DMA 链式传输

DMA 支持链式传输，可以在处理器不介入的情况下，连续完成多个不同配置的传输任务。

要想使用 DMA 的链式传输功能,用户需要预先在内存中构建一系列相互链接的 DMA 任务描述符列表。列表的第一个元素就是 DMA 通道的控制寄存器，其余的任务描述符存放在内存中。DMA 任务描述符中的 ChnLLPointer 控制字就是 DMA 通道 n 下一个任务描述符的指针。如果 ChnLLPointer 值非零，DMA 控制器会在完成当前任务描述符的相应任务后，从 ChnLLPointer 指向地址取下一个任务描述符。如果 ChnLLPointer 的值为 0，表示此任务描述符为列表中最后一个，链条中止。

表 163列举了任务描述符的格式，描述符的存储地址需要 8 字节对齐，并且描述符本身不要跨越 4K 的边界。

表 163: DMA 链式任务描述符

名称	地址偏移	描述
Ctrl	0x00	通道控制，格式同 CTRL 寄存器
TranSize	0x04	总传输长度，格式同 TRANSIZE 寄存器
SrcAddrL	0x08	源地址 L，格式同 SRCADDRL 寄存器
SrcAddrH	0x0C	源地址 H，格式同 SRCADDRH 寄存器
DstAddrL	0x10	目标地址 L，格式同 DSTADDRL 寄存器
DstAddrH	0x14	目标地址 H，格式同 DSTADDRH 寄存器
LLPointerL	0x18	链表指针 L，格式同 LLPOINTERL 寄存器
LLPointerH	0x1C	链表指针 H，格式同 LLPOINTERH 寄存器

28.4.3 数据顺序

DMA 控制器支持 3 种地址模式：递增模式，递减模式和不变模式。

- 递增模式是指，在 DMA 每次访问源或者目标地址后，DMA 下次访问此次源或者目标之后的地址
- 递减模式是指，在 DMA 每次访问源或者目标地址后，DMA 下次访问此次源或者目标之前的地址
- 不变模式是指，在 DMA 每次访问源或者目标地址后，DMA 下次访问的源或者目标地址固定不变

在源和目标地址的模式配置相同时，DMA 会保持源与目标区数据的字节顺序不变。但是如果源与目标的地址配置相反，那么源与目标地址间数据的字节顺序也会相反。其中不变模式字节顺序与递增模式相同。

28.5 DMA 寄存器

DMA 的寄存器列表如下：

HDMA base address: 0xF00C8000

地址偏移	名称	描述	复位值
0x0004	IDMISC	ID Misc	0x00000000
0x0010	DMACFG	配置查询寄存器	0x00000000

地址偏移	名称	描述	复位值
0x0014	DMACTRL	软件复位寄存器	0x00000000
0x0018	CHABORT	通道中止寄存器	0x00000000
0x0024	INTHALFSTS	传输完成一半中断状态寄存器	0x00000000
0x0028	INTTCSTS	传输完成状态中断寄存器	0x00000000
0x002C	INTABORTSTS	中止状态中断寄存器	0x00000000
0x0030	INTERRSTS	错误状态中断寄存器	0x00000000
0x0034	CHEN	通道使能状态寄存器	0x00000000
0x0040	CHCTRL[CH0][CTRL]	通道 ch0 控制寄存器	0x00000000
0x0044	CHCTRL[CH0][TRANSIZE]	通道 ch0 传输数据量	0x00000000
0x0048	CHCTRL[CH0][SRCADDR]	通道 ch0 源数据低位地址	0x00000000
0x004C	CHCTRL[CH0][CHANREQCTRL]	通道 ch0 数据请求选择控制寄存器	0x00000000
0x0050	CHCTRL[CH0][DSTADDR]	通道 ch0 目标数据低位地址	0x00000000
0x0058	CHCTRL[CH0][LLPOINTER]	通道 ch0 链式传输指针低位地址	0x00000000
0x0060	CHCTRL[CH1][CTRL]	通道 ch1 控制寄存器	0x00000000
0x0064	CHCTRL[CH1][TRANSIZE]	通道 ch1 传输数据量	0x00000000
0x0068	CHCTRL[CH1][SRCADDR]	通道 ch1 源数据低位地址	0x00000000
0x006C	CHCTRL[CH1][CHANREQCTRL]	通道 ch1 数据请求选择控制寄存器	0x00000000
0x0070	CHCTRL[CH1][DSTADDR]	通道 ch1 目标数据低位地址	0x00000000
0x0078	CHCTRL[CH1][LLPOINTER]	通道 ch1 链式传输指针低位地址	0x00000000
0x0080	CHCTRL[CH2][CTRL]	通道 ch2 控制寄存器	0x00000000
0x0084	CHCTRL[CH2][TRANSIZE]	通道 ch2 传输数据量	0x00000000
0x0088	CHCTRL[CH2][SRCADDR]	通道 ch2 源数据低位地址	0x00000000
0x008C	CHCTRL[CH2][CHANREQCTRL]	通道 ch2 数据请求选择控制寄存器	0x00000000
0x0090	CHCTRL[CH2][DSTADDR]	通道 ch2 目标数据低位地址	0x00000000
0x0098	CHCTRL[CH2][LLPOINTER]	通道 ch2 链式传输指针低位地址	0x00000000
0x00A0	CHCTRL[CH3][CTRL]	通道 ch3 控制寄存器	0x00000000
0x00A4	CHCTRL[CH3][TRANSIZE]	通道 ch3 传输数据量	0x00000000
0x00A8	CHCTRL[CH3][SRCADDR]	通道 ch3 源数据低位地址	0x00000000
0x00AC	CHCTRL[CH3][CHANREQCTRL]	通道 ch3 数据请求选择控制寄存器	0x00000000
0x00B0	CHCTRL[CH3][DSTADDR]	通道 ch3 目标数据低位地址	0x00000000
0x00B8	CHCTRL[CH3][LLPOINTER]	通道 ch3 链式传输指针低位地址	0x00000000
0x00C0	CHCTRL[CH4][CTRL]	通道 ch4 控制寄存器	0x00000000
0x00C4	CHCTRL[CH4][TRANSIZE]	通道 ch4 传输数据量	0x00000000
0x00C8	CHCTRL[CH4][SRCADDR]	通道 ch4 源数据低位地址	0x00000000
0x00CC	CHCTRL[CH4][CHANREQCTRL]	通道 ch4 数据请求选择控制寄存器	0x00000000

地址偏移	名称	描述	复位值
0x00D0	CHCTRL[CH4][DSTADDR]	通道 ch4 目标数据低位地址	0x00000000
0x00D8	CHCTRL[CH4][LLPOINTER]	通道 ch4 链式传输指针低位地址	0x00000000
0x00E0	CHCTRL[CH5][CTRL]	通道 ch5 控制寄存器	0x00000000
0x00E4	CHCTRL[CH5][TRANSIZE]	通道 ch5 传输数据量	0x00000000
0x00E8	CHCTRL[CH5][SRCADDR]	通道 ch5 源数据低位地址	0x00000000
0x00EC	CHCTRL[CH5][CHANREQCTRL]	通道 ch5 数据请求选择控制寄存器	0x00000000
0x00F0	CHCTRL[CH5][DSTADDR]	通道 ch5 目标数据低位地址	0x00000000
0x00F8	CHCTRL[CH5][LLPOINTER]	通道 ch5 链式传输指针低位地址	0x00000000
0x0100	CHCTRL[CH6][CTRL]	通道 ch6 控制寄存器	0x00000000
0x0104	CHCTRL[CH6][TRANSIZE]	通道 ch6 传输数据量	0x00000000
0x0108	CHCTRL[CH6][SRCADDR]	通道 ch6 源数据低位地址	0x00000000
0x010C	CHCTRL[CH6][CHANREQCTRL]	通道 ch6 数据请求选择控制寄存器	0x00000000
0x0110	CHCTRL[CH6][DSTADDR]	通道 ch6 目标数据低位地址	0x00000000
0x0118	CHCTRL[CH6][LLPOINTER]	通道 ch6 链式传输指针低位地址	0x00000000
0x0120	CHCTRL[CH7][CTRL]	通道 ch7 控制寄存器	0x00000000
0x0124	CHCTRL[CH7][TRANSIZE]	通道 ch7 传输数据量	0x00000000
0x0128	CHCTRL[CH7][SRCADDR]	通道 ch7 源数据低位地址	0x00000000
0x012C	CHCTRL[CH7][CHANREQCTRL]	通道 ch7 数据请求选择控制寄存器	0x00000000
0x0130	CHCTRL[CH7][DSTADDR]	通道 ch7 目标数据低位地址	0x00000000
0x0138	CHCTRL[CH7][LLPOINTER]	通道 ch7 链式传输指针低位地址	0x00000000
0x0140	CHCTRL[CH8][CTRL]	通道 ch8 控制寄存器	0x00000000
0x0144	CHCTRL[CH8][TRANSIZE]	通道 ch8 传输数据量	0x00000000
0x0148	CHCTRL[CH8][SRCADDR]	通道 ch8 源数据低位地址	0x00000000
0x014C	CHCTRL[CH8][CHANREQCTRL]	通道 ch8 数据请求选择控制寄存器	0x00000000
0x0150	CHCTRL[CH8][DSTADDR]	通道 ch8 目标数据低位地址	0x00000000
0x0158	CHCTRL[CH8][LLPOINTER]	通道 ch8 链式传输指针低位地址	0x00000000
0x0160	CHCTRL[CH9][CTRL]	通道 ch9 控制寄存器	0x00000000
0x0164	CHCTRL[CH9][TRANSIZE]	通道 ch9 传输数据量	0x00000000
0x0168	CHCTRL[CH9][SRCADDR]	通道 ch9 源数据低位地址	0x00000000
0x016C	CHCTRL[CH9][CHANREQCTRL]	通道 ch9 数据请求选择控制寄存器	0x00000000
0x0170	CHCTRL[CH9][DSTADDR]	通道 ch9 目标数据低位地址	0x00000000
0x0178	CHCTRL[CH9][LLPOINTER]	通道 ch9 链式传输指针低位地址	0x00000000
0x0180	CHCTRL[CH10][CTRL]	通道 ch10 控制寄存器	0x00000000
0x0184	CHCTRL[CH10][TRANSIZE]	通道 ch10 传输数据量	0x00000000
0x0188	CHCTRL[CH10][SRCADDR]	通道 ch10 源数据低位地址	0x00000000

地址偏移	名称	描述	复位值
0x018C	CHCTRL[CH10][CHANREQCTRL]	通道 ch10 数据请求选择控制寄存器	0x00000000
0x0190	CHCTRL[CH10][DSTADDR]	通道 ch10 目标数据低位地址	0x00000000
0x0198	CHCTRL[CH10][LLPOINTER]	通道 ch10 链式传输指针低位地址	0x00000000
0x01A0	CHCTRL[CH11][CTRL]	通道 ch11 控制寄存器	0x00000000
0x01A4	CHCTRL[CH11][TRANSIZE]	通道 ch11 传输数据量	0x00000000
0x01A8	CHCTRL[CH11][SRCADDR]	通道 ch11 源数据低位地址	0x00000000
0x01AC	CHCTRL[CH11][CHANREQCTRL]	通道 ch11 数据请求选择控制寄存器	0x00000000
0x01B0	CHCTRL[CH11][DSTADDR]	通道 ch11 目标数据低位地址	0x00000000
0x01B8	CHCTRL[CH11][LLPOINTER]	通道 ch11 链式传输指针低位地址	0x00000000
0x01C0	CHCTRL[CH12][CTRL]	通道 ch12 控制寄存器	0x00000000
0x01C4	CHCTRL[CH12][TRANSIZE]	通道 ch12 传输数据量	0x00000000
0x01C8	CHCTRL[CH12][SRCADDR]	通道 ch12 源数据低位地址	0x00000000
0x01CC	CHCTRL[CH12][CHANREQCTRL]	通道 ch12 数据请求选择控制寄存器	0x00000000
0x01D0	CHCTRL[CH12][DSTADDR]	通道 ch12 目标数据低位地址	0x00000000
0x01D8	CHCTRL[CH12][LLPOINTER]	通道 ch12 链式传输指针低位地址	0x00000000
0x01E0	CHCTRL[CH13][CTRL]	通道 ch13 控制寄存器	0x00000000
0x01E4	CHCTRL[CH13][TRANSIZE]	通道 ch13 传输数据量	0x00000000
0x01E8	CHCTRL[CH13][SRCADDR]	通道 ch13 源数据低位地址	0x00000000
0x01EC	CHCTRL[CH13][CHANREQCTRL]	通道 ch13 数据请求选择控制寄存器	0x00000000
0x01F0	CHCTRL[CH13][DSTADDR]	通道 ch13 目标数据低位地址	0x00000000
0x01F8	CHCTRL[CH13][LLPOINTER]	通道 ch13 链式传输指针低位地址	0x00000000
0x0200	CHCTRL[CH14][CTRL]	通道 ch14 控制寄存器	0x00000000
0x0204	CHCTRL[CH14][TRANSIZE]	通道 ch14 传输数据量	0x00000000
0x0208	CHCTRL[CH14][SRCADDR]	通道 ch14 源数据低位地址	0x00000000
0x020C	CHCTRL[CH14][CHANREQCTRL]	通道 ch14 数据请求选择控制寄存器	0x00000000
0x0210	CHCTRL[CH14][DSTADDR]	通道 ch14 目标数据低位地址	0x00000000
0x0218	CHCTRL[CH14][LLPOINTER]	通道 ch14 链式传输指针低位地址	0x00000000
0x0220	CHCTRL[CH15][CTRL]	通道 ch15 控制寄存器	0x00000000
0x0224	CHCTRL[CH15][TRANSIZE]	通道 ch15 传输数据量	0x00000000
0x0228	CHCTRL[CH15][SRCADDR]	通道 ch15 源数据低位地址	0x00000000

地址偏移	名称	描述	复位值
0x022C	CHCTRL[CH15][CHANREQCTRL]	通道 ch15 数据请求选择控制寄存器	0x00000000
0x0230	CHCTRL[CH15][DSTADDR]	通道 ch15 目标数据低位地址	0x00000000
0x0238	CHCTRL[CH15][LLPOINTER]	通道 ch15 链式传输指针低位地址	0x00000000
0x0240	CHCTRL[CH16][CTRL]	通道 ch16 控制寄存器	0x00000000
0x0244	CHCTRL[CH16][TRANSIZE]	通道 ch16 传输数据量	0x00000000
0x0248	CHCTRL[CH16][SRCADDR]	通道 ch16 源数据低位地址	0x00000000
0x024C	CHCTRL[CH16][CHANREQCTRL]	通道 ch16 数据请求选择控制寄存器	0x00000000
0x0250	CHCTRL[CH16][DSTADDR]	通道 ch16 目标数据低位地址	0x00000000
0x0258	CHCTRL[CH16][LLPOINTER]	通道 ch16 链式传输指针低位地址	0x00000000
0x0260	CHCTRL[CH17][CTRL]	通道 ch17 控制寄存器	0x00000000
0x0264	CHCTRL[CH17][TRANSIZE]	通道 ch17 传输数据量	0x00000000
0x0268	CHCTRL[CH17][SRCADDR]	通道 ch17 源数据低位地址	0x00000000
0x026C	CHCTRL[CH17][CHANREQCTRL]	通道 ch17 数据请求选择控制寄存器	0x00000000
0x0270	CHCTRL[CH17][DSTADDR]	通道 ch17 目标数据低位地址	0x00000000
0x0278	CHCTRL[CH17][LLPOINTER]	通道 ch17 链式传输指针低位地址	0x00000000
0x0280	CHCTRL[CH18][CTRL]	通道 ch18 控制寄存器	0x00000000
0x0284	CHCTRL[CH18][TRANSIZE]	通道 ch18 传输数据量	0x00000000
0x0288	CHCTRL[CH18][SRCADDR]	通道 ch18 源数据低位地址	0x00000000
0x028C	CHCTRL[CH18][CHANREQCTRL]	通道 ch18 数据请求选择控制寄存器	0x00000000
0x0290	CHCTRL[CH18][DSTADDR]	通道 ch18 目标数据低位地址	0x00000000
0x0298	CHCTRL[CH18][LLPOINTER]	通道 ch18 链式传输指针低位地址	0x00000000
0x02A0	CHCTRL[CH19][CTRL]	通道 ch19 控制寄存器	0x00000000
0x02A4	CHCTRL[CH19][TRANSIZE]	通道 ch19 传输数据量	0x00000000
0x02A8	CHCTRL[CH19][SRCADDR]	通道 ch19 源数据低位地址	0x00000000
0x02AC	CHCTRL[CH19][CHANREQCTRL]	通道 ch19 数据请求选择控制寄存器	0x00000000
0x02B0	CHCTRL[CH19][DSTADDR]	通道 ch19 目标数据低位地址	0x00000000
0x02B8	CHCTRL[CH19][LLPOINTER]	通道 ch19 链式传输指针低位地址	0x00000000
0x02C0	CHCTRL[CH20][CTRL]	通道 ch20 控制寄存器	0x00000000
0x02C4	CHCTRL[CH20][TRANSIZE]	通道 ch20 传输数据量	0x00000000
0x02C8	CHCTRL[CH20][SRCADDR]	通道 ch20 源数据低位地址	0x00000000

地址偏移	名称	描述	复位值
0x02CC	CHCTRL[CH20][CHANREQCTRL]	通道 ch20 数据请求选择控制寄存器	0x00000000
0x02D0	CHCTRL[CH20][DSTADDR]	通道 ch20 目标数据低位地址	0x00000000
0x02D8	CHCTRL[CH20][LLPOINTER]	通道 ch20 链式传输指针低位地址	0x00000000
0x02E0	CHCTRL[CH21][CTRL]	通道 ch21 控制寄存器	0x00000000
0x02E4	CHCTRL[CH21][TRANSIZE]	通道 ch21 传输数据量	0x00000000
0x02E8	CHCTRL[CH21][SRCADDR]	通道 ch21 源数据低位地址	0x00000000
0x02EC	CHCTRL[CH21][CHANREQCTRL]	通道 ch21 数据请求选择控制寄存器	0x00000000
0x02F0	CHCTRL[CH21][DSTADDR]	通道 ch21 目标数据低位地址	0x00000000
0x02F8	CHCTRL[CH21][LLPOINTER]	通道 ch21 链式传输指针低位地址	0x00000000
0x0300	CHCTRL[CH22][CTRL]	通道 ch22 控制寄存器	0x00000000
0x0304	CHCTRL[CH22][TRANSIZE]	通道 ch22 传输数据量	0x00000000
0x0308	CHCTRL[CH22][SRCADDR]	通道 ch22 源数据低位地址	0x00000000
0x030C	CHCTRL[CH22][CHANREQCTRL]	通道 ch22 数据请求选择控制寄存器	0x00000000
0x0310	CHCTRL[CH22][DSTADDR]	通道 ch22 目标数据低位地址	0x00000000
0x0318	CHCTRL[CH22][LLPOINTER]	通道 ch22 链式传输指针低位地址	0x00000000
0x0320	CHCTRL[CH23][CTRL]	通道 ch23 控制寄存器	0x00000000
0x0324	CHCTRL[CH23][TRANSIZE]	通道 ch23 传输数据量	0x00000000
0x0328	CHCTRL[CH23][SRCADDR]	通道 ch23 源数据低位地址	0x00000000
0x032C	CHCTRL[CH23][CHANREQCTRL]	通道 ch23 数据请求选择控制寄存器	0x00000000
0x0330	CHCTRL[CH23][DSTADDR]	通道 ch23 目标数据低位地址	0x00000000
0x0338	CHCTRL[CH23][LLPOINTER]	通道 ch23 链式传输指针低位地址	0x00000000
0x0340	CHCTRL[CH24][CTRL]	通道 ch24 控制寄存器	0x00000000
0x0344	CHCTRL[CH24][TRANSIZE]	通道 ch24 传输数据量	0x00000000
0x0348	CHCTRL[CH24][SRCADDR]	通道 ch24 源数据低位地址	0x00000000
0x034C	CHCTRL[CH24][CHANREQCTRL]	通道 ch24 数据请求选择控制寄存器	0x00000000
0x0350	CHCTRL[CH24][DSTADDR]	通道 ch24 目标数据低位地址	0x00000000
0x0358	CHCTRL[CH24][LLPOINTER]	通道 ch24 链式传输指针低位地址	0x00000000
0x0360	CHCTRL[CH25][CTRL]	通道 ch25 控制寄存器	0x00000000
0x0364	CHCTRL[CH25][TRANSIZE]	通道 ch25 传输数据量	0x00000000
0x0368	CHCTRL[CH25][SRCADDR]	通道 ch25 源数据低位地址	0x00000000

地址偏移	名称	描述	复位值
0x036C	CHCTRL[CH25][CHANREQCTRL]	通道 ch25 数据请求选择控制寄存器	0x00000000
0x0370	CHCTRL[CH25][DSTADDR]	通道 ch25 目标数据低位地址	0x00000000
0x0378	CHCTRL[CH25][LLPOINTER]	通道 ch25 链式传输指针低位地址	0x00000000
0x0380	CHCTRL[CH26][CTRL]	通道 ch26 控制寄存器	0x00000000
0x0384	CHCTRL[CH26][TRANSIZE]	通道 ch26 传输数据量	0x00000000
0x0388	CHCTRL[CH26][SRCADDR]	通道 ch26 源数据低位地址	0x00000000
0x038C	CHCTRL[CH26][CHANREQCTRL]	通道 ch26 数据请求选择控制寄存器	0x00000000
0x0390	CHCTRL[CH26][DSTADDR]	通道 ch26 目标数据低位地址	0x00000000
0x0398	CHCTRL[CH26][LLPOINTER]	通道 ch26 链式传输指针低位地址	0x00000000
0x03A0	CHCTRL[CH27][CTRL]	通道 ch27 控制寄存器	0x00000000
0x03A4	CHCTRL[CH27][TRANSIZE]	通道 ch27 传输数据量	0x00000000
0x03A8	CHCTRL[CH27][SRCADDR]	通道 ch27 源数据低位地址	0x00000000
0x03AC	CHCTRL[CH27][CHANREQCTRL]	通道 ch27 数据请求选择控制寄存器	0x00000000
0x03B0	CHCTRL[CH27][DSTADDR]	通道 ch27 目标数据低位地址	0x00000000
0x03B8	CHCTRL[CH27][LLPOINTER]	通道 ch27 链式传输指针低位地址	0x00000000
0x03C0	CHCTRL[CH28][CTRL]	通道 ch28 控制寄存器	0x00000000
0x03C4	CHCTRL[CH28][TRANSIZE]	通道 ch28 传输数据量	0x00000000
0x03C8	CHCTRL[CH28][SRCADDR]	通道 ch28 源数据低位地址	0x00000000
0x03CC	CHCTRL[CH28][CHANREQCTRL]	通道 ch28 数据请求选择控制寄存器	0x00000000
0x03D0	CHCTRL[CH28][DSTADDR]	通道 ch28 目标数据低位地址	0x00000000
0x03D8	CHCTRL[CH28][LLPOINTER]	通道 ch28 链式传输指针低位地址	0x00000000
0x03E0	CHCTRL[CH29][CTRL]	通道 ch29 控制寄存器	0x00000000
0x03E4	CHCTRL[CH29][TRANSIZE]	通道 ch29 传输数据量	0x00000000
0x03E8	CHCTRL[CH29][SRCADDR]	通道 ch29 源数据低位地址	0x00000000
0x03EC	CHCTRL[CH29][CHANREQCTRL]	通道 ch29 数据请求选择控制寄存器	0x00000000
0x03F0	CHCTRL[CH29][DSTADDR]	通道 ch29 目标数据低位地址	0x00000000
0x03F8	CHCTRL[CH29][LLPOINTER]	通道 ch29 链式传输指针低位地址	0x00000000
0x0400	CHCTRL[CH30][CTRL]	通道 ch30 控制寄存器	0x00000000
0x0404	CHCTRL[CH30][TRANSIZE]	通道 ch30 传输数据量	0x00000000
0x0408	CHCTRL[CH30][SRCADDR]	通道 ch30 源数据低位地址	0x00000000

地址偏移	名称	描述	复位值
0x040C	CHCTRL[CH30][CHANREQCTRL]	通道 ch30 数据请求选择控制寄存器	0x00000000
0x0410	CHCTRL[CH30][DSTADDR]	通道 ch30 目标数据低位地址	0x00000000
0x0418	CHCTRL[CH30][LLPOINTER]	通道 ch30 链式传输指针低位地址	0x00000000
0x0420	CHCTRL[CH31][CTRL]	通道 ch31 控制寄存器	0x00000000
0x0424	CHCTRL[CH31][TRANSIZE]	通道 ch31 传输数据量	0x00000000
0x0428	CHCTRL[CH31][SRCADDR]	通道 ch31 源数据低位地址	0x00000000
0x042C	CHCTRL[CH31][CHANREQCTRL]	通道 ch31 数据请求选择控制寄存器	0x00000000
0x0430	CHCTRL[CH31][DSTADDR]	通道 ch31 目标数据低位地址	0x00000000
0x0438	CHCTRL[CH31][LLPOINTER]	通道 ch31 链式传输指针低位地址	0x00000000

表 164: DMAV2 寄存器列表

28.6 DMA 寄存器详细信息

DMA 的寄存器详细说明如下：

28.6.1 IDMISC (0x4)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																DMASTATE			CURCHAN					RSVD							
N/A																RO			RO					N/A							
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	x	x	x	x	x	x	x	x

IDMISC [31:0]

位域	名称	描述
15-13	DMASTATE	DMA 状态机 0 表示状态空闲 其他值表示非空闲
12-8	CURCHAN	当前正在工作的通道

IDMISC 位域

28.6.2 DMACFG (0x10)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CHAINXFR	REQSYNC	RSVD				DATAWIDTH		ADDRWIDTH							CORENUM	BUSNUM	REQNUM					FIFODEPTH						CHANNELNUM			
RO	RO	N/A				RO		RO							RO	RO	RO					RO						RO			
0	0	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

DMACFG [31:0]

位域	名称	描述
31	CHAINXFR	0：不支持链式传输 1：支持链式传输
30	REQSYNC	DMA 请求是否被同步
25-24	DATAWIDTH	AXI 总线位宽
23-17	ADDRWIDTH	AXI 地址位宽
16	CORENUM	DMA 核心数量
15	BUSNUM	AXI 总线数量
14-10	REQNUM	请求/响应支持数量
9-4	FIFODEPTH	FIFO 深度
3-0	CHANNELNUM	通道数量

DMACFG 位域

28.6.3 DMACTRL (0x14)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																															RESET
N/A																															WO
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0

DMACTRL [31:0]

位域	名称	描述
0	RESET	软件复位，将该位置 1 则复位 DMA 控制器并禁止所有通道

DMACTRL 位域

28.6.4 CHABORT (0x18)

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

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

CHABORT [31:0]

位域	名称	描述
31-0	CHABORT	每个通道对应一位，写 1 表示将该通道的传输中止，仅对已使能的通道有效。

CHABORT 位域

28.6.5 INTHALFSTS (0x24)

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

INTHALFSTS [31:0]

位域	名称	描述
31-0	STS	传输完成一半中断状态，每一位对应一个通道。

INTHALFSTS 位域

28.6.6 INTTCSTS (0x28)

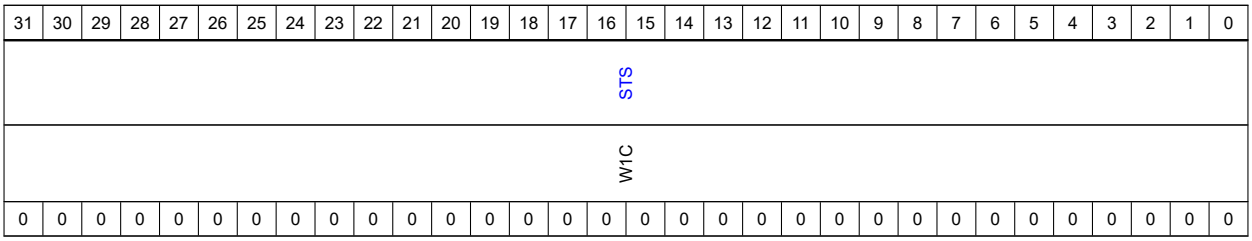
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
STS																															
W1C																															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

INTTCSTS [31:0]

位域	名称	描述
31-0	STS	通道传输结束标志，每一位对应一个通道。

INTTCSTS 位域

28.6.7 INTABORTSTS (0x2C)

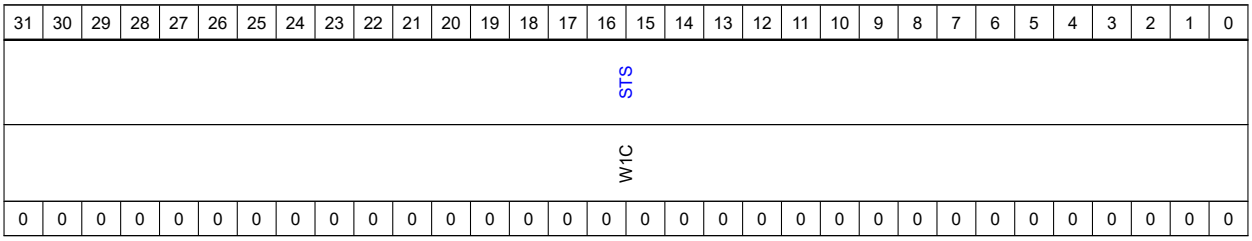


INTABORTSTS [31:0]

位域	名称	描述
31-0	STS	通道传输中止标志，每一位对应一个通道。

INTABORTSTS 位域

28.6.8 INTERRSTS (0x30)

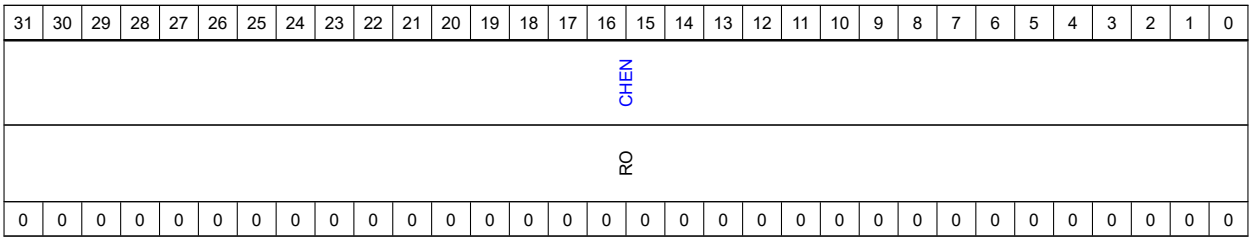


INTERRSTS [31:0]

位域	名称	描述
31-0	STS	通道传输错误标志，每一位对应一个通道。 错误事件包括： - 总线错误 - 非对齐地址 - 非对齐位宽 - 保留设置

INTERRSTS 位域

28.6.9 CHEN (0x34)



CHEN [31:0]

位域	名称	描述
31-0	CHEN	通道使能状态查询，每一位对应一个通道。

CHEN 位域

28.6.10 CHCTRL[CTRL] (0x40 + 0x20 * n)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
INFINITELOOP	HANDSHAKEOPT	PRIORITY	BURSTOPT	SRCBURSTSIZE				SRCWIDTH			DSTWIDTH			SRCMODE	DSTMODE	SRCADDRCTRL		DSTADDRCTRL		RSVD							INTHALFCNTMASK	INTABTMASK	INTERRMASK	INTTCMASK	ENABLE
RW	RW	RW	RW	RW				RW			RW			RW	RW	RW		RW		N/A							RW	RW	RW	RW	RW
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	x	x	x	x	x	x	x	0	0	0	0	0

CHCTRL[CTRL] [31:0]

位域	名称	描述
31	INFINITELOOP	设置当前配置为无限循环
30	HANDSHAKEOPT	0: 一个请求传输一个 burst 1: 一个传输 ch_tts 中定义的所有数据的请求
29	PRIORITY	通道优先级： 0：低优先级 1：高优先级
28	BURSTOPT	设置改变 burst 数量的定义
27-24	SRCBURSTSIZE	源数据 burst 数量，burst 传输的字节数等于（SrcBurstSize * SrcWidth）。 当 BURSTOPT 为 0 时： 0x0: 1 transfer 0x1: 2 transfers 0x2: 4 transfers 0x3: 8 transfers 0x4: 16 transfers 0x5: 32 transfers 0x6: 64 transfers 0x7: 128 transfers 0x8: 256 transfers 0x9:512 transfers 0xa: 1024 transfers 0xb - 0xf: 非法值 当 BURSTOPT 为 1 时，burst 数为本寄存器加 1（0：1transfer；0xf: 16 transfer）

位域	名称	描述
23-21	SRCWIDTH	源数据传输位宽： 0x0: 8 位 0x1: 16 位 0x2: 32 位 0x3: 64 位 0x4 - 0x7: 非法值 对于 XDMA，允许设置的最大值为 0x3，对于 HDMA，允许设置的最大值为 0x2
20-18	DSTWIDTH	目标数据传输位宽： 0x0: 8 位 0x1: 16 位 0x2: 32 位 0x3: 64 位 0x4 - 0x7: 非法值 对于 XDMA，允许设置的最大值为 0x3，对于 HDMA，允许设置的最大值为 0x2
17	SRCMODE	源数据 DMA 握手模式： 0：普通模式 1：握手模式 普通模式：软件设置 Enable 位使能及开始 DMA 传输； 握手模式：软件设置 Enable 位使能 DMA，通过 DMAMUX 选择请求信号开始 DMA 传输
16	DSTMODE	目标数据 DMA 握手模式： 0：普通模式 1：握手模式 源数据/目标数据握手模式的区别在于何时响应 DMA 传输请求， 源数据握手模式会在读完源数据后响应； 目标数据握手模式会在写完数据后响应。 注意：不能同时设置 SrcMode 和 DstMode，不然结果未知。
15-14	SRCADDRCTRL	源数据地址访问控制： 0x0：地址递增 0x1：地址递减 0x2：固定地址 0x3：非法设置
13-12	DSTADDRCTRL	目标数据地址访问控制： 0x0：地址递增 0x1：地址递减 0x2：固定地址 0x3：非法设置

位域	名称	描述
4	INTHALFCNTMASK	传输完成一半中断屏蔽： 0：允许数据传输完成一半时发出中断 1：禁止数据传输完成一半时发出中断
3	INTABTMASK	数据中止中断屏蔽： 0：允许数据中止时发出中断 1：禁止数据中止时发出中断
2	INTERRMASK	数据错误中断屏蔽： 0：允许数据错误时发出中断 1：禁止数据错误时发出中断
1	INTTCMASK	数据传输完成中断屏蔽： 0：允许数据传输完成时发出中断 1：禁止数据传输完成时发出中断
0	ENABLE	通道使能

CHCTRL[CTRL] 位域

28.6.11 CHCTRL[TRANSIZE] (0x44 + 0x20 * n)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD				TRANSIZE																											
N/A				RW																											
x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

CHCTRL[TRANSIZE] [31:0]

位域	名称	描述
27-0	TRANSIZE	源数据的数据量，总传输数据量为（TranSize * SrcWidth）

CHCTRL[TRANSIZE] 位域

28.6.12 CHCTRL[SRCADDR] (0x48 + 0x20 * n)

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

CHCTRL[SRCADDR] [31:0]

位域	名称	描述
31-0	SRCADDRL	源数据地址的低 32 位

CHCTRL[SRCADDR] 位域

28.6.13 CHCTRL[CHANREQCTRL] (0x4C + 0x20 * n)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD			SRCREQSEL					RSVD			DSTREQSEL					RSVD															
N/A			RW					N/A			RW					N/A															
x	x	x	0	0	0	0	0	x	x	x	0	0	0	0	0	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x

CHCTRL[CHANREQCTRL] [31:0]

位域	名称	描述
28-24	SRCREQSEL	源数据 DMA 请求选择
20-16	DSTREQSEL	目标数据 DMA 请求选择

CHCTRL[CHANREQCTRL] 位域

28.6.14 CHCTRL[DSTADDR] (0x50 + 0x20 * n)

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

CHCTRL[DSTADDR] [31:0]

位域	名称	描述
31-0	DSTADDRL	目标数据地址的低 32 位

CHCTRL[DSTADDR] 位域

28.6.15 CHCTRL[LLPOINTER] (0x58 + 0x20 * n)

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

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RW																													N/A		
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	x	x	x

CHCTRL[LLPOINTER] [31:0]

位域	名称	描述
31-3	LLPOINTERL	链式传输的描述符低位地址，地址为 64 位对齐。

CHCTRL[LLPOINTER] 位域

28.7 DMA 配置使用说明

以启动某次 DMA 握手模式传输为例，所需要的步骤为：

1. 配置对应通道的寄存器 transize, 总数据量为 transize x srcwidth；
2. 配置对应通道的寄存器 srcaddr，指向源数据的起始地址；
3. 配置对应通道的寄存器 chanreqctl，包括源数据 DMA 请求选择 srcreqsel 或目的数据 DMA 请求选择 dstreqsel；
4. 配置对应通道的寄存器 dstaddr，指向目标数据的起始地址；
5. 如果是非链式传输，则配置 llpointer 为 0x0，否则配置为下一链式数据结构的起始地址；
6. 配置对应通道的寄存器 ctrl，包括优先级 priority 字段、源突发长度 srcburstsize 字段、源数据位宽 srcwidth、目的数据位宽 dstwidth 字段、通道使能字段 enable；
7. 配置 DMAMUX，将 srcreqsel 或者 dstreqsel 路由至对应的通道;
8. 读取的寄存器 inttcsts，判断对应通道的传输是否完成；

29 DMA 请求路由器 DMAMUX

本章节介绍 DMA 请求路由器 DMAMUX 的功能和特性。

29.1 概述

本章节介绍 DMA 请求路由器 DMAMUX 的主要特性：

- 支持多达 128 个外设 DMA 请求
- 支持 32 个输出通道

本产品所有支持生成 DMA 请求的外设，及其在 DMA 请求路由器 DMAMUX 上的分配详情，请查阅节 27.2。

29.2 DMAMUX 架构图

本模块的结构图如图 24。

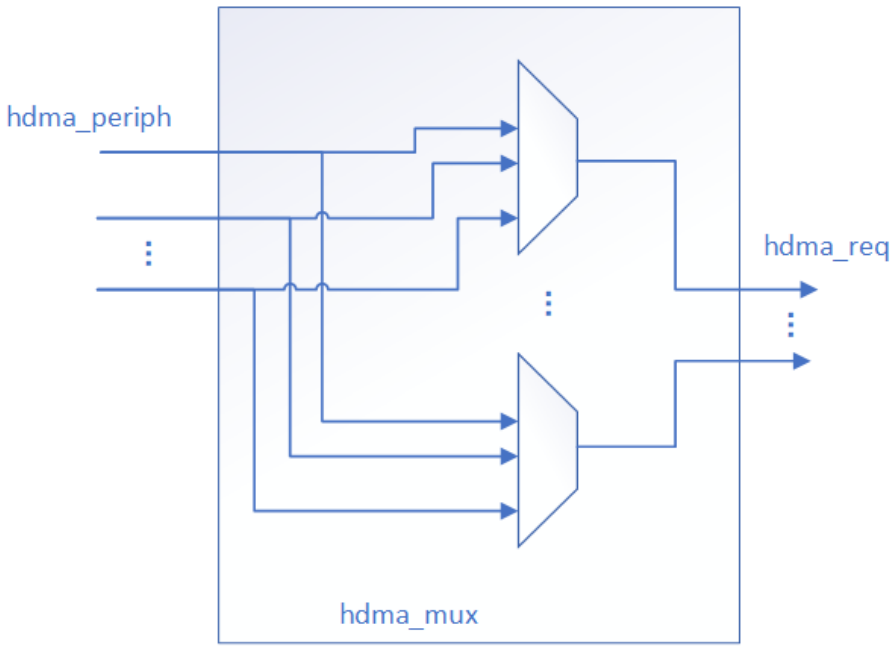


图 24: DMAMUX 结构框图

29.3 管脚说明

表 165: DMAMUX 管脚说明

管脚	方向	功能描述
hdma_req[31:0]	输出	输出到 DMA 的 DMA 请求信号
hdma_ack[31:0]	输入	输入的 DMA 应答信号
hdma_req_periph[127:0]	输入	输入的各外设 dma 请求信号
hdma_ack_periph[127:0]	输出	输出到各外设 dma 应答信号

29.4 功能说明

DMA 请求路由器的功能是实现外设 DMA 请求对 DMA 控制器各个通道的灵活动态映射，使得任意外设可以发送 DMA 请求给任意 DMA 控制器的任意通道。

通过 DMAMUX 的各个通道的配置寄存器 CHCFGx 配置 DMAMUX 的步骤如下：

- 通过MUXCFGx的SOURCE位选择外设的 DMA 请求
- MUXCFGx的ENABLE位置 1，打开 DMAMUX 的通道 x

29.5 DMAMUX 寄存器

DMAMUX 的寄存器列表如下：

DMAMUX base address: 0xF00C4000

地址偏移	名称	描述	复位值
0x0000	MUXCFG[HDMA_MUX0]	HDMA 端口 0 配置寄存器	0x00000000
0x0004	MUXCFG[HDMA_MUX1]	HDMA 端口 1 配置寄存器	0x00000000

地址偏移	名称	描述	复位值
0x0008	MUXCFG[HDMA_MUX2]	HDMA 端口 2 配置寄存器	0x00000000
0x000C	MUXCFG[HDMA_MUX3]	HDMA 端口 3 配置寄存器	0x00000000
0x0010	MUXCFG[HDMA_MUX4]	HDMA 端口 4 配置寄存器	0x00000000
0x0014	MUXCFG[HDMA_MUX5]	HDMA 端口 5 配置寄存器	0x00000000
0x0018	MUXCFG[HDMA_MUX6]	HDMA 端口 6 配置寄存器	0x00000000
0x001C	MUXCFG[HDMA_MUX7]	HDMA 端口 7 配置寄存器	0x00000000
0x0020	MUXCFG[HDMA_MUX8]	HDMA 端口 8 配置寄存器	0x00000000
0x0024	MUXCFG[HDMA_MUX9]	HDMA 端口 9 配置寄存器	0x00000000
0x0028	MUXCFG[HDMA_MUX10]	HDMA 端口 10 配置寄存器	0x00000000
0x002C	MUXCFG[HDMA_MUX11]	HDMA 端口 11 配置寄存器	0x00000000
0x0030	MUXCFG[HDMA_MUX12]	HDMA 端口 12 配置寄存器	0x00000000
0x0034	MUXCFG[HDMA_MUX13]	HDMA 端口 13 配置寄存器	0x00000000
0x0038	MUXCFG[HDMA_MUX14]	HDMA 端口 14 配置寄存器	0x00000000
0x003C	MUXCFG[HDMA_MUX15]	HDMA 端口 15 配置寄存器	0x00000000
0x0040	MUXCFG[HDMA_MUX16]	HDMA 端口 16 配置寄存器	0x00000000
0x0044	MUXCFG[HDMA_MUX17]	HDMA 端口 17 配置寄存器	0x00000000
0x0048	MUXCFG[HDMA_MUX18]	HDMA 端口 18 配置寄存器	0x00000000
0x004C	MUXCFG[HDMA_MUX19]	HDMA 端口 19 配置寄存器	0x00000000
0x0050	MUXCFG[HDMA_MUX20]	HDMA 端口 20 配置寄存器	0x00000000
0x0054	MUXCFG[HDMA_MUX21]	HDMA 端口 21 配置寄存器	0x00000000
0x0058	MUXCFG[HDMA_MUX22]	HDMA 端口 22 配置寄存器	0x00000000
0x005C	MUXCFG[HDMA_MUX23]	HDMA 端口 23 配置寄存器	0x00000000
0x0060	MUXCFG[HDMA_MUX24]	HDMA 端口 24 配置寄存器	0x00000000
0x0064	MUXCFG[HDMA_MUX25]	HDMA 端口 25 配置寄存器	0x00000000
0x0068	MUXCFG[HDMA_MUX26]	HDMA 端口 26 配置寄存器	0x00000000
0x006C	MUXCFG[HDMA_MUX27]	HDMA 端口 27 配置寄存器	0x00000000
0x0070	MUXCFG[HDMA_MUX28]	HDMA 端口 28 配置寄存器	0x00000000
0x0074	MUXCFG[HDMA_MUX29]	HDMA 端口 29 配置寄存器	0x00000000
0x0078	MUXCFG[HDMA_MUX30]	HDMA 端口 30 配置寄存器	0x00000000
0x007C	MUXCFG[HDMA_MUX31]	HDMA 端口 31 配置寄存器	0x00000000

表 166: DMAMUX 寄存器列表

29.6 DMAMUX 寄存器详细信息

DMAMUX 的寄存器详细说明如下：

29.6.1 MUXCFG (0x0 + 0x4 * n)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ENABLE	RSVD																								SOURCE						

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RW	N/A																								RW						
0	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0

MUXCFG [31:0]

位域	名称	描述
31	ENABLE	使能 DMA 端口
6-0	SOURCE	将选定的 DMA 请求路由到 DMA 端口

MUXCFG 位域

29.7 DMAMUX 配置使用说明

配置 DMAMUX 所需要的步骤为：

1. 配置对应通道的寄存器 MUXCFG.SOURCE 字段, 选择 128 个外设 DMA 请求的哪一个；
2. 同时配置对应通道的寄存器 MUXCFG.ENABLE 为 1；

30 通信信箱 MBX

本章节介绍通信信箱 MBX 的功能和特性。

30.1 特性总结

通信信箱 MBX 支持进行处理器核间通信，以及处理器的进程间通信。主要特性如下：

- 支持 2 个寄存器访问接口
- 每个接口支持 TX FIFO 和 RX FIFO
- 支持标志位反映 TX FIFO 和 RX FIFO 状态
- 支持生成中断

30.2 功能描述

本章节描述通信信箱 MBX 的功能。

30.2.1 MBX 寄存器访问接口

通信信箱 MBX 有 2 套寄存器访问接口，接口 A 和接口 B。A 和 B 接口都具有一套 TX FIFO 寄存器、RX FIFO 寄存器、控制寄存器和状态寄存器。

用户从 A 接口的发送端 TX 发送的数据，可以在 B 接口的接收端 RX 接收到。同理，A 接口的接收端 RX 可以接收到 B 接口发送端 TX 发送的数据。

30.2.2 FIFO 管理

MBX 的发送端 TX FIFO 深度为 4 个字 (4×32 位)，TX FIFO 的写入接口为 TXREG 寄存器。MBX 支持提示 TX FIFO 的各类状态，即 TX FIFO 是否为空，TX FIFO 是否已满，以及 TX FIFO 中空的字数目。

MBX 的接收端 RX FIFO 深度为 4 个字 (4×32 位)。RX FIFO 的读取接口为 RXREG 寄存器。MBX 支持提示 RX FIFO 的各类状态，即 RX FIFO 是否为空，RX FIFO 是否已满，以及 RX FIFO 中有效的字数目。

30.2.3 中断

MBX 支持生成以下中断：

- TX FIFO 为空时，
- TX FIFO 为满时，
- RX FIFO 非空时，
- RX FIFO 已满时

30.3 MBX 寄存器

MBX 的寄存器列表如下：

MBX0A base address: 0xF00A0000

MBX0B base address: 0xF00A4000

地址偏移	名称	描述	复位值
0x0000	CR	控制寄存器	0x00000000
0x0004	SR	状态寄存器	0x000000E2

地址偏移	名称	描述	复位值
0x0008	TXREG	消息字发送寄存器	0x00000000
0x000C	RXREG	消息字接收寄存器	0x00000000
0x0010	TXWRD[TXFIFO0]	消息队列发送 FIFO	0x00000000
0x0020	RXWRD[RXFIFO0]	消息队列接收 FIFO	0x00000000

表 167: MBX 寄存器列表

30.4 MBX 寄存器详细信息

MBX 的寄存器详细说明如下：

30.4.1 CR (0x0)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TXRESET	RSVD															BARCTL		RSVD					BEIE	TFMAIE	TFMEIE	RFMAIE	RFMFIE	RSVD		TWMEIE	RWMVIE
RW	N/A															RW		N/A					RW	RW	RW	RW	RW	N/A		RW	RW
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

CR [31:0]

位域	名称	描述
31	TXRESET	复位发送消息字和发送消息队列。 写 1 复位。
15-14	BARCTL	总线访问响应控制： 0x0：不产生任何错误响应 0x1：以下情况会产生总线访问错误响应 - (1) 访问无效地址，(2) 向只读寄存器发出写操作，(3) 当发送消息队列 FIFO 已满或发送消息字未被读取时对其进行写操作，(4) 当接收消息队列 FIFO 为空或接收消息字无效时对其进行读操作。 0x2：保留值 0x3：保留值 本寄存器配置不影响以下结果：对已满的发送消息队列的写操作不会生效，对未被读取的发送消息字的写操作会覆盖之前的消息字，对空的接收消息队列或无效的接收消息字的读操作，会返回上次读取的数据或复位值。
8	BEIE	总线访问错误中断使能，其对应的错误状态包括 SR[13:8] 的各个位域。
7	TFMAIE	发送消息队列非空中断使能
6	TFMEIE	发送消息队列空中断使能
5	RFMAIE	接收消息队列非空中断使能
4	RFMFIE	接收消息队列满中断使能

位域	名称	描述
1	TWMEIE	发送消息字已被读取中断使能
0	RWMVIE	接收消息字有效中断使能

CR 位域

30.4.2 SR (0x4)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD								RFVC				TFEC				RSVD		ERRRE	EWTRF	ERRFE	EWTFF	EAIVA	EW2RO	TFMA	TFME	RFMA	RFMF	RSVD		TWME	RWMV
N/A								RO				RO				N/A		W1C	W1C	W1C	W1C	W1C	W1C	RW	RW	RO	RO	N/A		RO	RO
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	x	x	0	0	0	0	0	0	1	1	1	0	0	0	1	0

SR [31:0]

位域	名称	描述
23-20	RFVC	接收消息队列中的有效消息数量
19-16	TFEC	发送消息队列中的剩余空间数量
13	ERRRE	访问错误状态标志：在接收消息字无效时对其进行读取
12	EWTRF	访问错误状态标志：在发送消息字未被读取时对其进行写入
11	ERRFE	访问错误状态标志：在接收消息队列为空时对其进行读取
10	EWTFF	访问错误状态标志：在发送消息队列满时对其进行写入
9	EAIVA	访问错误状态标志：访问了不存在的寄存器地址
8	EW2RO	访问错误状态标志：对接收消息字和接收消息队列进行了写操作
7	TFMA	状态标志：发送消息队列未满
6	TFME	状态标志：发送消息队列空
5	RFMA	状态标志：接收消息队列非空 在相关寄存器使能时能够触发中断
4	RFMF	状态标志：发送消息队列非空 在相关寄存器使能时能够触发中断
1	TWME	状态标志：发送消息字为空（已被读取） 在相关寄存器使能时能够触发中断
0	RWMV	状态标志：接收消息字有效 在相关寄存器使能时能够触发中断

SR 位域

30.4.3 TXREG (0x8)

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

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

TXREG [31:0]

位域	名称	描述
31-0	TXREG	发送消息字

TXREG 位域

30.4.4 RXREG (0xC)

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

RXREG [31:0]

位域	名称	描述
31-0	RXREG	接收消息字

RXREG 位域

30.4.5 TXWRD (0x10 + 0x4 * n)

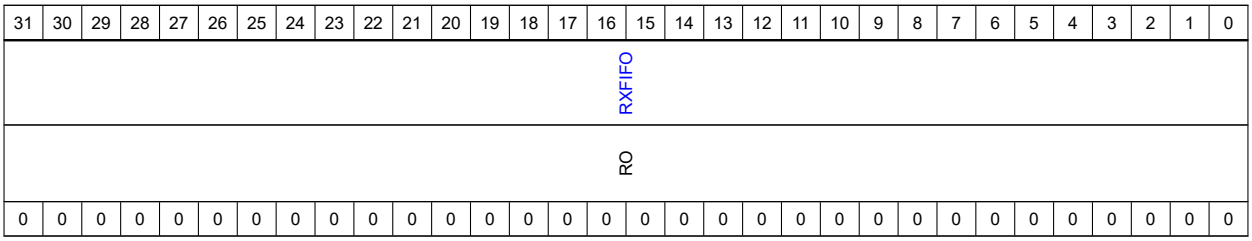
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TXFIFO																															
WO																															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

TXWRD [31:0]

位域	名称	描述
31-0	TXFIFO	发送消息队列 FIFO 写入寄存器，FIFO 深度为 4，FIFO 位宽为 32bit，向该寄存器的写操作会向 FIFO 中 push 一个 word。

TXWRD 位域

30.4.6 RXWRD (0x20 + 0x4 * n)



RXWRD [31:0]

位域	名称	描述
31-0	RXFIFO	接收消息队列 FIFO 读出寄存器，FIFO 深度为 4，FIFO 位宽为 32bit，向该寄存器的读操作会从 FIFO 中 pop 一个 word。

RXWRD 位域

31 循环冗余检测 CRC

本章节介绍循环冗余检测 CRC 的功能和特性。

31.1 特性总结

循环冗余检测 CRC 的主要特性如下：

- 支持 8 个通道独立计算
- 支持可编程 CRC 算法（可支持所有最高次幂小于等于 32 的 CRC 算法）
- 支持按字节、双字节、全字传输
- 支持常见的 15 种 CRC 算法快速配置

31.2 功能描述

循环冗余检测 CRC 模块做为系统 APB 总线上的从设备，用户可以通过总线配置，输入码流和读取 CRC 校验码。CRC 模块可以根据用户自定义的 CRC 算法，向 CHN[n][DATA] 寄存器输入待检验的码流，即可从 CHN[n][result] 得到其对应的 CRC 校验码，CRC 模块提供了 8 个独立计算的通道，每个通道均可以独立配置其 CRC 算法并独立完成计算。输入的码流支持按字节、双字节和全字传输，同时还支持码流配置成按字节或按位翻转次序。每个通道都支持常见的 15 种 CRC 算法的快速预配置。

CRC 使用前需进行 CRC 算法的配置，可以通过详细配置各个算法寄存器或配置预配置寄存器来快速完成。支持码流断续传输，码流断续传输和连续传输的计算结果一致。CHN[n][result] 内的计算结果始终为当前已累计传输码流的校验码。

31.3 CRC 算法配置

- CHN[n][POLY]: CRC 算法多项式的表达式，例如：CRC-32 对应的多项式表达为 0x04C11DB7。若 CRC 多项式低于 32 位时，此配置寄存器高位写 0 即可。
- CHN[n][INIT_DATA]: CRC 初始值。
- CHN[n][XOROUT]: CRC 输出值的异或值，即 CRC 计算结果与此参数异或后得到最终的校验码。
- CHN[n][MISC_SETTING]: CRC 的杂项配置，包含：POLY_WIDTH(CRC 校验码的长度)、REV_IN(是否将码流按位翻转)、REV_OUT(是否将输出结果按位翻转)、BYTE_REV(是否将输入码流按字节翻转)。
- CHN[n][PRE_SET]: CRC 快速预配置寄存器。预配置寄存器对应的 CRC 算法优先度高于上述详细算法参数寄存器的配置，即当用户配置了预配置寄存器后，之前若有已配置的算法参数将会被预配置的对应参数覆盖。

31.4 CRC 寄存器

CRC 的寄存器列表如下：

CRC base address: 0xF0080000

地址偏移	名称	描述	复位值
0x0000	CHN[0][PRE_SET]	0 crc 预配置寄存器	0x00000000
0x0004	CHN[0][CLR]	chn0 复位 CRC 通道配置和结果	0x00000000
0x0008	CHN[0][POLY]	chn0 多项式表达式	0x00000000
0x000C	CHN[0][INIT_DATA]	chn0 CRC 初始值	0x00000000

地址偏移	名称	描述	复位值
0x0010	CHN[0][XOROUT]	chn0 CRC 输出异或值	0x00000000
0x0014	CHN[0][MISC_SETTING]	chn0 CRC 的杂项配置	0x00000000
0x0018	CHN[0][DATA]	chn0 CRC 输入字节流	0x00000000
0x001C	CHN[0][RESULT]	chn0 CRC 结果	0x00000000
0x0040	CHN[1][PRE_SET]	1 crc 预配置寄存器	0x00000000
0x0044	CHN[1][CLR]	chn1 复位 CRC 通道配置和结果	0x00000000
0x0048	CHN[1][POLY]	chn1 多项式表达式	0x00000000
0x004C	CHN[1][INIT_DATA]	chn1 CRC 初始值	0x00000000
0x0050	CHN[1][XOROUT]	chn1 CRC 输出异或值	0x00000000
0x0054	CHN[1][MISC_SETTING]	chn1 CRC 的杂项配置	0x00000000
0x0058	CHN[1][DATA]	chn1 CRC 输入字节流	0x00000000
0x005C	CHN[1][RESULT]	chn1 CRC 结果	0x00000000
0x0080	CHN[2][PRE_SET]	2 crc 预配置寄存器	0x00000000
0x0084	CHN[2][CLR]	chn2 复位 CRC 通道配置和结果	0x00000000
0x0088	CHN[2][POLY]	chn2 多项式表达式	0x00000000
0x008C	CHN[2][INIT_DATA]	chn2 CRC 初始值	0x00000000
0x0090	CHN[2][XOROUT]	chn2 CRC 输出异或值	0x00000000
0x0094	CHN[2][MISC_SETTING]	chn2 CRC 的杂项配置	0x00000000
0x0098	CHN[2][DATA]	chn2 CRC 输入字节流	0x00000000
0x009C	CHN[2][RESULT]	chn2 CRC 结果	0x00000000
0x00C0	CHN[3][PRE_SET]	3 crc 预配置寄存器	0x00000000
0x00C4	CHN[3][CLR]	chn3 复位 CRC 通道配置和结果	0x00000000
0x00C8	CHN[3][POLY]	chn3 多项式表达式	0x00000000
0x00CC	CHN[3][INIT_DATA]	chn3 CRC 初始值	0x00000000
0x00D0	CHN[3][XOROUT]	chn3 CRC 输出异或值	0x00000000
0x00D4	CHN[3][MISC_SETTING]	chn3 CRC 的杂项配置	0x00000000
0x00D8	CHN[3][DATA]	chn3 CRC 输入字节流	0x00000000
0x00DC	CHN[3][RESULT]	chn3 CRC 结果	0x00000000
0x0100	CHN[4][PRE_SET]	4 crc 预配置寄存器	0x00000000
0x0104	CHN[4][CLR]	chn4 复位 CRC 通道配置和结果	0x00000000
0x0108	CHN[4][POLY]	chn4 多项式表达式	0x00000000
0x010C	CHN[4][INIT_DATA]	chn4 CRC 初始值	0x00000000
0x0110	CHN[4][XOROUT]	chn4 CRC 输出异或值	0x00000000
0x0114	CHN[4][MISC_SETTING]	chn4 CRC 的杂项配置	0x00000000
0x0118	CHN[4][DATA]	chn4 CRC 输入字节流	0x00000000
0x011C	CHN[4][RESULT]	chn4 CRC 结果	0x00000000
0x0140	CHN[5][PRE_SET]	5 crc 预配置寄存器	0x00000000
0x0144	CHN[5][CLR]	chn5 复位 CRC 通道配置和结果	0x00000000
0x0148	CHN[5][POLY]	chn5 多项式表达式	0x00000000
0x014C	CHN[5][INIT_DATA]	chn5 CRC 初始值	0x00000000

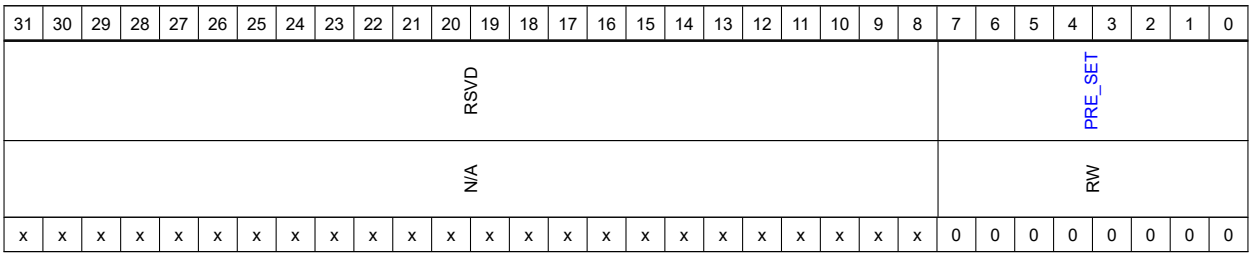
地址偏移	名称	描述	复位值
0x0150	CHN[5][XOROUT]	chn5 CRC 输出异或值	0x00000000
0x0154	CHN[5][MISC_SETTING]	chn5 CRC 的杂项配置	0x00000000
0x0158	CHN[5][DATA]	chn5 CRC 输入字节流	0x00000000
0x015C	CHN[5][RESULT]	chn5 CRC 结果	0x00000000
0x0180	CHN[6][PRE_SET]	6 crc 预配置寄存器	0x00000000
0x0184	CHN[6][CLR]	chn6 复位 CRC 通道配置和结果	0x00000000
0x0188	CHN[6][POLY]	chn6 多项式表达式	0x00000000
0x018C	CHN[6][INIT_DATA]	chn6 CRC 初始值	0x00000000
0x0190	CHN[6][XOROUT]	chn6 CRC 输出异或值	0x00000000
0x0194	CHN[6][MISC_SETTING]	chn6 CRC 的杂项配置	0x00000000
0x0198	CHN[6][DATA]	chn6 CRC 输入字节流	0x00000000
0x019C	CHN[6][RESULT]	chn6 CRC 结果	0x00000000
0x01C0	CHN[7][PRE_SET]	7 crc 预配置寄存器	0x00000000
0x01C4	CHN[7][CLR]	chn7 复位 CRC 通道配置和结果	0x00000000
0x01C8	CHN[7][POLY]	chn7 多项式表达式	0x00000000
0x01CC	CHN[7][INIT_DATA]	chn7 CRC 初始值	0x00000000
0x01D0	CHN[7][XOROUT]	chn7 CRC 输出异或值	0x00000000
0x01D4	CHN[7][MISC_SETTING]	chn7 CRC 的杂项配置	0x00000000
0x01D8	CHN[7][DATA]	chn7 CRC 输入字节流	0x00000000
0x01DC	CHN[7][RESULT]	chn7 CRC 结果	0x00000000

表 168: CRC 寄存器列表

31.5 CRC 寄存器详细信息

CRC 的寄存器详细说明如下：

31.5.1 CHN[PRE_SET] (0x0 + 0x40 * n)



CHN[PRE_SET] [31:0]

位域	名称	描述
7-0	PRE_SET	0: 不使能预设值 1: CRC32 2: CRC32-AUTOSAR 3: CRC16-CCITT 4: CRC16-XMODEM 5: CRC16-MODBUS 1: CRC32 2: CRC32-autosar 3: CRC16-ccitt 4: CRC16-xmodem 5: CRC16-modbus 6: crc16_dnp 7: crc16_x25 8: crc16_usb 9: crc16_maxim 10: crc16_ibm 11: crc8_maxim 12: crc8_rohc 13: crc8_itu 14: crc8 15: crc5_usb

CHN[PRE_SET] 位域

31.5.2 CHN[CLR] (0x4 + 0x40 * n)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																															CLR
N/A																															RW
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0

CHN[CLR] [31:0]

位域	名称	描述
0	CLR	写 1 复位 crc 通道配置和结果， 读操作一直读到 0

CHN[CLR] 位域

31.5.3 CHN[POLY] (0x8 + 0x40 * n)

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

CHN[POLY] [31:0]

位域	名称	描述
31-0	POLY	CRC 配置多项式表达式，比如配置 CRC32 时，需写入 0x4C11DB7

CHN[POLY] 位域

31.5.4 CHN[INIT_DATA] (0xC + 0x40 * n)

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

CHN[INIT_DATA] [31:0]

位域	名称	描述
31-0	INIT_DATA	CRC 的初始值

CHN[INIT_DATA] 位域

31.5.5 CHN[XOROUT] (0x10 + 0x40 * n)

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

CHN[XOROUT] [31:0]

位域	名称	描述
31-0	XOROUT	CRC 的结果会和本寄存器进行异或

CHN[XOROUT] 位域

31.5.6 CHN[MISC_SETTING] (0x14 + 0x40 * n)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD							BYTE_REV	RSVD							REV_OUT	RSVD							REV_IN	RSVD		POLY_WIDTH					
N/A							RW	N/A							RW	N/A							RW	N/A		RW					
x	x	x	x	x	x	x	0	x	x	x	x	x	x	x	0	x	x	x	x	x	x	x	0	x	x	0	0	0	0	0	0

CHN[MISC_SETTING] [31:0]

位域	名称	描述
24	BYTE_REV	0: 不反转输入字节次序 1: 输入字节流按字节次序反转
16	REV_OUT	0: 不反转输出结果次序 1: 输出结果按位反转
8	REV_IN	0: 不反转输入字节流的位次序 1: 输入字节流按位反转
5-0	POLY_WIDTH	crc 长度

CHN[MISC_SETTING] 位域

31.5.7 CHN[DATA] (0x18 + 0x40 * n)

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

CHN[DATA] [31:0]

位域	名称	描述
31-0	DATA	待计算 CRC 的数据

CHN[DATA] 位域

31.5.8 CHN[RESULT] (0x1C + 0x40 * n)

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

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CHN[RESULT] [31:0]																															

位域	名称	描述
31-0	RESULT	CRC 结果

CHN[RESULT] 位域

32 运动控制系统概述

本章节介绍了本产品的运动控制系统。本产品的运动控制系统包括 2 个 PWM 定时器、2 个正交编码器接口、2 个正交编码器输出、2 个运动管理控制器、2 个串行编码器控制器、1 个旋变解调器和 1 个互联管理器。可以和片上的模拟外设如模数转换器 ADC，比较器 ACMP 等外设配合使用。本产品包括一个全局定时器，为运动控制系统提供时间戳。

所有运动控制单元和同步定时器均工作在外部总线时钟 (CLK_TOP_AHB), 以实现最小延时。

本产品的电机控制系统如图 25。

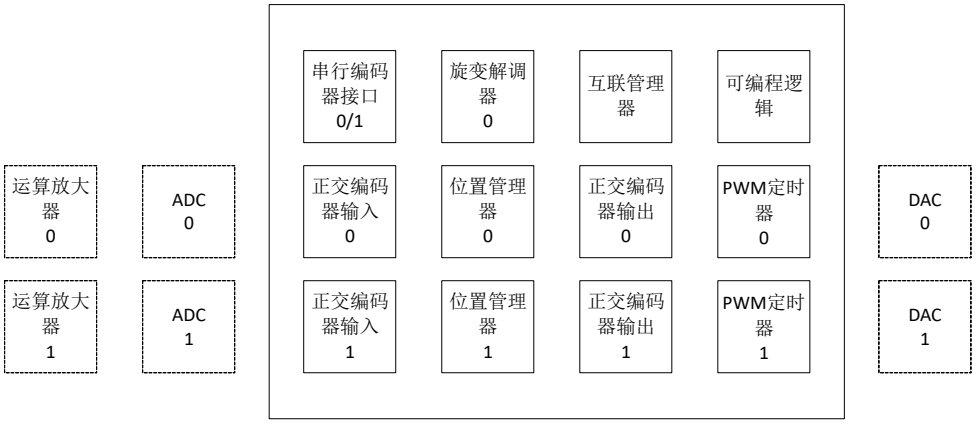


图 25: 电机系统框图

32.1 PWM 定时器 PWM

本产品支持 2 个 PWM 定时器。单个 PWM 支持高达 28 位计数器，支持 24 个通道，其中通道 0~7 用于生成 8 路独立或者 4 对互补 PWM 输出，支持死区控制和故障保护。其余 16 路通道支持输出比较，可以通过互联管理器输出到 IO，或者其他片上模块。PWM 定时器支持输入捕获。

本产品上，单个 PWM 定时器信号连接如下：

- 通道 0~7 输出经过 PWM 控制逻辑，连接到 IO 上
- 通道 8~23 输出连接到电机控制单元内的互联管理器
- 输入捕获 0~7 来自 IO
- 输入捕获 8~23 来自电机控制单元内的互联管理器
- 计数器同步触发输入 SYNCI 来自电机控制单元内的互联管理器
- 强制输出的使能输入 FRCI 来自电机控制单元内的互联管理器
- 强制输出影子寄存器的生效的触发输入 SHSYNCI 来自电机控制单元内的互联管理器
- 外部故障保护输入 FAULTE0~1 来自 IO，可在无时钟状态实现故障保护
- 内部故障保护输入 FAULTI0~3 来自电机控制单元内的互联管理器，需在电机系统时钟正常工作时实现故障保护

各个 PWM 模块信号与电机控制单元内的互联管理器内信号连接请查阅节 32.8。

32.2 正交编码器接口 QEI

本产品支持 2 个正交编码器接口 QEI。QEI 支持外部的各种类型的正交编码器，QEI 还支持数字霍尔信号输入，用以感应电机的位置信息，并提供电机的转向，转速信息。

本产品上，单个正交编码器接口 QEI 的信号连接如下：

- A/U 相输入 APH，来自 IO
- B/V 相输入 BPH，来自 IO
- Z/W 相输入 ZPH，来自 IO
- H 信号输入 HPH，也称为 HOME，来自 IO
- F 信号输入 FPH，也称为 FAULT，来自 IO
- 位置输出，输出到位置转发器
- 正弦 ADC 输入，来自 ADC 转发器
- 余弦 ADC 输入，来自 ADC 转发器
- 快照输入 SNAPI，来自电机控制单元内的互联管理器
- 触发输出 TRGO，连接到电机控制单元内的互联管理器

正交编码器接口 QEI 的各个信号支持通过电机控制单元内互联管理器灵活地分配，可以连接到 TRGM 的各个 IO，也可以连接到片上的其他外设。具体信号连接请查阅节 32.8。

32.3 正交编码器输出 QEO

本产品支持 2 个正交编码器输出 QEO。QEO 支持向外部发送正交编码信号。

本产品上，单个正交编码器输出 QEO 的信号连接如下：

- A/U 相输出，通过 IO 输出
- B/V 相输出，通过 IO 输出
- Z/W 相输出，通过 IO 输出
- 位置输入，来自位置转发器
- A 相 DAC 输出，输出到 DAC 转发器
- B 相 DAC 输出，输出到 DAC 转发器
- C 相 DAC 输出，输出到 DAC 转发器
- 换向信号输出，步进电机和直流无刷电机换向，连接到电机控制单元内互联管理器

正交编码器输出 QEO 的各个信号支持通过电机控制单元内互联管理器灵活地分配，可以连接到 TRGM 的各个 IO，也可以连接到片上的其他外设。具体信号连接请查阅节 32.8。

32.4 串行编码器接口 SEI

本产品支持 2 个串行编码器接口 SEI。SEI 支持同步和异步的编码器，可以用作主机和编码器。

本产品上，串行编码器接口 SEI 的信号连接如下：

- CK 双向时钟，来自 IO
- TX 输出，通过 IO 输出
- RX 输入，来自 IO
- DE 输出使能，通过 IO 输出
- 位置输入，来自位置转发器
- 位置输出，输出到位置转发器
- 触发输入，来自电机控制单元内的互联管理器
- 触发输出，连接到电机控制单元内的互联管理器

串行编码器接口 SEI 的各个信号支持通过电机控制单元内互联管理器灵活地分配，可以连接到 TRGM 的各个 IO，

也可以连接到片上的其他外设。具体信号连接请查阅节 32.8。

32.5 位置管理器 MMC

本产品支持 2 个位置管理器 MMC。MMC 支持运动信号的产生、分析、补偿和预测输出。

本产品上，位置管理器 MMC 的信号连接如下：

- 位置输入，来自位置转发器
- 位置输出 0，输出到位置转发器
- 位置输出 1，输出到位置转发器
- 触发输入，来自电机控制单元内的互联管理器
- 触发输出，连接到电机控制单元内的互联管理器

位置管理器 MMC 的各个信号支持通过电机控制单元内互联管理器灵活地分配，可以连接到 TRGM 的各个 IO，也可以连接到片上的其他外设。具体信号连接请查阅节 32.8。

32.6 旋变解调器 RDC

本产品支持 1 个旋变解调器 RDC。RDC 支持输出旋变励磁信号，并对 ADC 采样后的正交信号进行解调。

本产品上，旋变解调器 RDC 的信号连接如下：

- PWM 励磁信号输出，通过 IO 输出
- 正弦励磁信号输出，输出到 DAC 转发器
- 正弦 ADC 输入，来自 ADC 转发器
- 余弦 ADC 输入，来自 ADC 转发器
- 正弦 ADC 输出，输出到 ADC 转发器
- 余弦 ADC 输出，输出到 ADC 转发器
- 触发输入，来自电机控制单元内的互联管理器
- 触发输出，连接到电机控制单元内的互联管理器

旋变解调器 RDC 的各个信号支持通过电机控制单元内互联管理器灵活地分配，可以连接到 TRGM 的各个 IO，也可以连接到片上的其他外设。具体信号连接请查阅节 32.8。

32.7 可编程逻辑单元 PLB

本产品支持 1 个可编程逻辑单元 PLB。PLB 支持组合逻辑运算和计数运算。

本产品上，可编程逻辑单元 PLB 的信号连接如下：

- 触发输入，来自电机控制单元内的互联管理器
- 触发输出，连接到电机控制单元内的互联管理器

可编程逻辑单元 PLB 的各个信号支持通过电机控制单元内互联管理器灵活地分配，可以连接到 TRGM 的各个 IO，也可以连接到片上的其他外设。具体信号连接请查阅节 32.8。

32.8 互联管理器 TRGM

本产品的互联管理器 TRGM 支持电机控制单元内外各个设备的信号间互通互联，可以把片上各个外设整合起来，实现外设间相互同步，相互配合。

互联管理器支持多个输入，输入来自于 IO，电机控制单元内外的各个外设，以及其他电机控制单元的互联管

理器。各个互联管理器的输入信号，请查阅小节 32.8.1。

互联管理器支持多个输出，用户可以从互联管理器的众多输入信号中选择一个，连接到任意输出。各个互联管理器的输出信号分配，请查阅小节 32.8.2。

注意：将电机系统内部信号连接到电机系统外时，需要考虑时钟频率和信号有效长度。

以下几个信号只有一个电机系统时钟周期，这些信号可以在电机系统内部任意连接，不建议出电机系统：

正交解码器 n 触发输出；霍尔传感器 n 接口触发输出；同步定时器通道 n。

互联管理器支持管理电机控制单元内外设的 DMA 请求，本产品上，电机控制单元内设备的 DMA 请求不直接连接到 DMAMUX，而是通过 TRGM 转接。各个 TRGM 管理的 DMA 请求，请查阅小节 32.8.3。

互联管理器支持多个数字滤波器，可以对 TRGM 的特定输入信号进行滤波。各个 TRGM 中配置有数字滤波器的输入信号，请查阅小节 32.8.4。

32.8.1 互联管理器输入分配

本章节介绍互联管理器输入信号分配。

互联管理器的输入信号分配：

TRGM_INPUT MUX 编号	TRGM_INPUT MUX SOURCE 名称	描述
0	VSS	低电平
1	VDD	高电平
2	DEBUG_FLAG	调试模式进入标志位
3	USB0_SOF	USB0 桢起始
4	PTPC_CMP0	精确时间协议模块 PTPC 输出比较 0
5	PTPC_CMP1	精确时间协议模块 PTPC 输出比较 1
6	CMP0_OUT	比较器 0 输出
7	CMP1_OUT	比较器 1 输出
8	GPTMR0_OUT2	通用定时器 0 通道 2
9	GPTMR0_OUT3	通用定时器 0 通道 3
10	GPTMR1_OUT2	通用定时器 1 通道 2
11	GPTMR1_OUT3	通用定时器 1 通道 3
12	GPTMR2_OUT2	通用定时器 2 通道 2
13	GPTMR2_OUT3	通用定时器 2 通道 3
14	GPTMR3_OUT2	通用定时器 2 通道 2
15	GPTMR3_OUT3	通用定时器 3 通道 3
16	TRGM0_P0	互联管理器 0 输入 0（来自 IO）
17	TRGM0_P1	互联管理器 0 输入 1（来自 IO）
18	TRGM0_P2	互联管理器 0 输入 2（来自 IO）
19	TRGM0_P3	互联管理器 0 输入 3（来自 IO）
20	TRGM0_P4	互联管理器 0 输入 4（来自 IO）
21	TRGM0_P5	互联管理器 0 输入 5（来自 IO）
22	TRGM0_P6	互联管理器 0 输入 6（来自 IO）
23	TRGM0_P7	互联管理器 0 输入 7（来自 IO）

TRGM_INPUT MUX 编号	TRGM_INPUT MUX SOURCE 名称	描述
24	SYNT0_CH0	同步定时器 0 通道 0
25	SYNT0_CH1	同步定时器 0 通道 1
26	SYNT0_CH2	同步定时器 0 通道 2
27	SYNT0_CH3	同步定时器 0 通道 3
28	MMC0_TRGO_0	
29	MMC0_TRGO_1	
30	MMC1_TRGO_0	
31	MMC1_TRGO_1	
32	QEO0_TRGO_0	
33	QEO0_TRGO_1	
34	QEO0_TRGO_2	
35	QEO0_TRGO_3	
36	QEO0_TRGO_4	
37	QEO0_TRGO_5	
38	QEO0_TRGO_6	
39	QEO0_TRGO_7	
40	QEO1_TRGO_0	
41	QEO1_TRGO_1	
42	QEO1_TRGO_2	
43	QEO1_TRGO_3	
44	QEO1_TRGO_4	
45	QEO1_TRGO_5	
46	QEO1_TRGO_6	
47	QEO1_TRGO_7	
48	PWM0_CH8REF	PWM 定时器 0 通道 8 参考输出
49	PWM0_CH9REF	PWM 定时器 0 通道 9 参考输出
50	PWM0_CH10REF	PWM 定时器 0 通道 10 参考输出
51	PWM0_CH11REF	PWM 定时器 0 通道 11 参考输出
52	PWM0_CH12REF	PWM 定时器 0 通道 12 参考输出
53	PWM0_CH13REF	PWM 定时器 0 通道 13 参考输出
54	PWM0_CH14REF	PWM 定时器 0 通道 14 参考输出
55	PWM0_CH15REF	PWM 定时器 0 通道 15 参考输出
56	PWM1_CH8REF	PWM 定时器 1 通道 8 参考输出
57	PWM1_CH9REF	PWM 定时器 1 通道 9 参考输出
58	PWM1_CH10REF	PWM 定时器 1 通道 10 参考输出
59	PWM1_CH11REF	PWM 定时器 1 通道 11 参考输出
60	PWM1_CH12REF	PWM 定时器 1 通道 12 参考输出
61	PWM1_CH13REF	PWM 定时器 1 通道 13 参考输出
62	PWM1_CH14REF	PWM 定时器 1 通道 14 参考输出

TRGM_INPUT MUX 编号	TRGM_INPUT MUX SOURCE 名称	描述
63	PWM1_CH15REF	PWM 定时器 1 通道 15 参考输出
64	PLB_OUT00	PLB 模块输出 0
65	PLB_OUT01	
66	PLB_OUT02	
67	PLB_OUT03	
68	PLB_OUT04	
69	PLB_OUT05	
70	PLB_OUT06	
71	PLB_OUT07	
72	PLB_OUT08	
73	PLB_OUT09	
74	PLB_OUT10	
75	PLB_OUT11	
76	PLB_OUT12	
77	PLB_OUT13	
78	PLB_OUT14	
79	PLB_OUT15	
80	PLB_OUT16	
81	PLB_OUT17	
82	PLB_OUT18	
83	PLB_OUT19	
84	PLB_OUT20	
85	PLB_OUT21	
86	PLB_OUT22	
87	PLB_OUT23	
88	PLB_OUT24	
89	PLB_OUT25	
90	PLB_OUT26	
91	PLB_OUT27	
92	PLB_OUT28	
93	PLB_OUT29	
94	PLB_OUT30	
95	PLB_OUT31	
96	RDC_TRGO_0	
97	RDC_TRGO_1	
98	QEI1_TRGO	正交解码器 1 触发输出
99	QEI0_TRGO	正交解码器 0 触发输出
100	SEI_TRGO_0	
101	SEI_TRGO_1	

TRGM_INPUT MUX 编号	TRGM_INPUT MUX SOURCE 名称	描述
102	SEI_TRGO_2	
103	SEI_TRGO_3	
104	SEI_TRGO_4	
105	SEI_TRGO_5	
106	SEI_TRGO_6	
107	SEI_TRGO_7	
108	PWM0_FAULT0	PWM0 fault 输入 0（来自 IO）
109	PWM0_FAULT1	PWM0 fault 输入 1（来自 IO）
110	PWM1_FAULT0	PWM1 fault 输入 0（来自 IO）
111	PWM1_FAULT1	PWM1 fault 输入 1（来自 IO）
112	PWM0_CAPIN0	PWM0 输入 0（来自 IO）
113	PWM0_CAPIN1	
114	PWM0_CAPIN2	
115	PWM0_CAPIN3	
116	PWM0_CAPIN4	
117	PWM0_CAPIN5	
118	PWM0_CAPIN6	
119	PWM0_CAPIN7	
120	PWM1_CAPIN0	PWM1 输入 0（来自 IO）
121	PWM1_CAPIN1	
122	PWM1_CAPIN2	
123	PWM1_CAPIN3	
124	PWM1_CAPIN4	
125	PWM1_CAPIN5	
126	PWM1_CAPIN6	
127	PWM1_CAPIN7	

表 169: TRGM_INPUT MUX 列表

32.8.2 互联管理器输出分配

本章节介绍互联管理器输出信号分配。

互联管理器的输出信号分配：

TRGM_OUTPUT MUX 编号	TRGM_OUTPUT MUX SOURCE 名称	描述
0	MOT2OPAMP0_0	放大器 0 输入触发 0
1	MOT2OPAMP0_1	放大器 0 输入触发 1
2	MOT2OPAMP0_2	放大器 0 输入触发 2
3	MOT2OPAMP0_3	放大器 0 输入触发 3
4	MOT2OPAMP0_4	放大器 0 输入触发 4

TRGM_OUTPUT MUX 编号	TRGM_OUTPUT MUX SOURCE 名称	描述
5	MOT2OPAMP0_5	放大器 0 输入触发 5
6	MOT2OPAMP0_6	放大器 0 输入触发 6
7	MOT2OPAMP0_7	放大器 0 输入触发 7
8	MOT2OPAMP1_0	放大器 1 输入触发 0
9	MOT2OPAMP1_1	放大器 1 输入触发 1
10	MOT2OPAMP1_2	放大器 1 输入触发 2
11	MOT2OPAMP1_3	放大器 1 输入触发 3
12	MOT2OPAMP1_4	放大器 1 输入触发 4
13	MOT2OPAMP1_5	放大器 1 输入触发 5
14	MOT2OPAMP1_6	放大器 1 输入触发 6
15	MOT2OPAMP1_7	放大器 1 输入触发 6
16	GPTMR0_IN2	通用定时器 0 通道 2 输入
17	GPTMR0_IN3	通用定时器 0 通道 3 输入
18	GPTMR0_SYNCI	通用定时器 0 计数器同步输入
19	GPTMR1_IN2	通用定时器 1 通道 2 输入
20	GPTMR1_IN3	通用定时器 1 通道 3 输入
21	GPTMR1_SYNCI	通用定时器 1 计数器同步输入
22	GPTMR2_IN2	通用定时器 2 通道 2 输入；此位同时作为 DAC0 阶梯模式启动触发 1
23	GPTMR2_IN3	通用定时器 2 通道 3 输入；此位同时作为 DAC0 阶梯模式启动触发 2
24	GPTMR2_SYNCI	通用定时器 2 计数器同步输入；此位同时作为 DAC0 阶梯模式启动触发 3
25	GPTMR3_IN2	通用定时器 3 通道 2 输入；此位同时作为 DAC1 阶梯模式启动触发 1
26	GPTMR3_IN3	通用定时器 3 通道 3 输入；此位同时作为 DAC1 阶梯模式启动触发 2
27	GPTMR3_SYNCI	通用定时器 3 计数器同步输入；此位同时作为 DAC1 阶梯模式启动触发 3
28	CMP0_WIN	比较器 0 窗口模式输入
29	CMP1_WIN	比较器 1 窗口模式输入
30	DAC0_BUFTRG	DAC0 缓存模式启动触发；此位同时作为 DAC0 阶梯模式启动触发 0
31	DAC1_BUFTRG	DAC1 缓存模式启动触发；此位同时作为 DAC1 阶梯模式启动触发 0
32	ADC0_STRGI	ADC0 的序列转换触发输入
33	ADC1_STRGI	ADC1 的序列转换触发输入
34	ADCX_PTRGI0A	ADC0, 1 的抢占转换触发输入 0A
35	ADCX_PTRGI0B	ADC0, 1 的抢占转换触发输入 0B

TRGM_OUTPUT MUX 编号	TRGM_OUTPUT MUX SOURCE 名称	描述
36	ADCX_PTRGI0C	ADC0, 1 的抢占转换触发输入 0C
37	ADCX_PTRGI1A	ADC0, 1 的抢占转换触发输入 1A
38	ADCX_PTRGI1B	ADC0, 1 的抢占转换触发输入 1B
39	ADCX_PTRGI1C	ADC0, 1 的抢占转换触发输入 1C
40	ADCX_PTRGI2A	ADC0, 1 的抢占转换触发输入 2A
41	ADCX_PTRGI2B	ADC0, 1 的抢占转换触发输入 2B
42	ADCX_PTRGI2C	ADC0, 1 的抢占转换触发输入 2C
43	ADCX_PTRGI3A	ADC0, 1 的抢占转换触发输入 3A
44	ADCX_PTRGI3B	ADC0, 1 的抢占转换触发输入 3B
45	ADCX_PTRGI3C	ADC0, 1 的抢占转换触发输入 3C
46	CAN_PTPC0_CAP	精确时间协议模块 PTPC 输入捕获 0,
47	CAN_PTPC1_CAP	精确时间协议模块 PTPC 输入捕获 1
48	QEO0_TRIG_IN0	QEO0 触发输入 0
49	QEO0_TRIG_IN1	QEO0 触发输入 1
50	QEO1_TRIG_IN0	QEO1 触发输入 0
51	QEO1_TRIG_IN1	QEO1 触发输入 1
52	SEI_TRIG_IN0	SEI 触发输入 0
53	SEI_TRIG_IN1	SEI 触发输入 1
54	SEI_TRIG_IN2	SEI 触发输入 2
55	SEI_TRIG_IN3	SEI 触发输入 3
56	SEI_TRIG_IN4	SEI 触发输入 4
57	SEI_TRIG_IN5	SEI 触发输入 5
58	SEI_TRIG_IN6	SEI 触发输入 6
59	SEI_TRIG_IN7	SEI 触发输入 7
60	MMC0_TRIG_IN0	MMC0 触发输入 0
61	MMC0_TRIG_IN1	MMC0 触发输入 1
62	MMC1_TRIG_IN0	MMC1 触发输入 0
63	MMC1_TRIG_IN1	MMC1 触发输入 1
64	PLB_IN_00	PLB 模块输入 00
65	PLB_IN_01	PLB 模块输入 01
66	PLB_IN_02	PLB 模块输入 02
67	PLB_IN_03	PLB 模块输入 03
68	PLB_IN_04	PLB 模块输入 04
69	PLB_IN_05	PLB 模块输入 05
70	PLB_IN_06	PLB 模块输入 06
71	PLB_IN_07	PLB 模块输入 07
72	PLB_IN_08	PLB 模块输入 08
73	PLB_IN_09	PLB 模块输入 09
74	PLB_IN_10	PLB 模块输入 10

TRGM_OUTPUT MUX 编号	TRGM_OUTPUT MUX SOURCE 名称	描述
75	PLB_IN_11	PLB 模块输入 11
76	PLB_IN_12	PLB 模块输入 12
77	PLB_IN_13	PLB 模块输入 13
78	PLB_IN_14	PLB 模块输入 14
79	PLB_IN_15	PLB 模块输入 15
80	PLB_IN_16	PLB 模块输入 16
81	PLB_IN_17	PLB 模块输入 17
82	PLB_IN_18	PLB 模块输入 18
83	PLB_IN_19	PLB 模块输入 19
84	PLB_IN_20	PLB 模块输入 20
85	PLB_IN_21	PLB 模块输入 21
86	PLB_IN_22	PLB 模块输入 22
87	PLB_IN_23	PLB 模块输入 23
88	PLB_IN_24	PLB 模块输入 24
89	PLB_IN_25	PLB 模块输入 25
90	PLB_IN_26	PLB 模块输入 26
91	PLB_IN_27	PLB 模块输入 27
92	PLB_IN_28	PLB 模块输入 28
93	PLB_IN_29	PLB 模块输入 29
94	PLB_IN_30	PLB 模块输入 30
95	PLB_IN_31	PLB 模块输入 31
96	MOT_GPIO0	互联管理器输出 0（输出至 IO）
97	MOT_GPIO1	互联管理器输出 1（输出至 IO）
98	MOT_GPIO2	互联管理器输出 2（输出至 IO）
99	MOT_GPIO3	互联管理器输出 3（输出至 IO）
100	MOT_GPIO4	互联管理器输出 4（输出至 IO）
101	MOT_GPIO5	互联管理器输出 5（输出至 IO）
102	MOT_GPIO6	互联管理器输出 6（输出至 IO）
103	MOT_GPIO7	互联管理器输出 7（输出至 IO）
104	PWM_IN8	PWM 定时器 x 输入捕获 8
105	PWM_IN9	PWM 定时器 x 输入捕获 9
106	PWM_IN10	PWM 定时器 x 输入捕获 10
107	PWM_IN11	PWM 定时器 x 输入捕获 11
108	PWM_IN12	PWM 定时器 x 输入捕获 12
109	PWM_IN13	PWM 定时器 x 输入捕获 13
110	PWM_IN14	PWM 定时器 x 输入捕获 14
111	PWM_IN15	PWM 定时器 x 输入捕获 15
112	PWM0_FRCI	PWM 定时器 0 强制输出控制输入

TRGM_OUTPUT MUX 编号	TRGM_OUTPUT MUX SOURCE 名称	描述
113	PWM0_FRCSYNCI	PWM 定时器 0 强制输出控制同步生效输入
114	PWM0_SYNCI	PWM 定时器 0 计数器同步触发输入
115	PWM0_SHRLDSYNCI	PWM 定时器 0 影子寄存器生效的触发输入
116	PWM0_FAULTI0	PWM 定时器 0 故障保护输入 0
117	PWM0_FAULTI1	PWM 定时器 0 故障保护输入 1
118	PWM1_FRCI	PWM 定时器 1 强制输出控制输入
119	PWM1_FRCSYNCI	PWM 定时器 1 强制输出控制同步生效输入
120	PWM1_SYNCI	PWM 定时器 1 计数器同步触发输入
121	PWM1_SHRLDSYNCI	PWM 定时器 1 影子寄存器生效的触发输入
122	PWM1_FAULTI0	PWM 定时器 1 故障保护输入 0
123	PWM1_FAULTI1	PWM 定时器 1 故障保护输入 1
124	RDC_TRIG_IN0	RDC 触发输入 0
125	RDC_TRIG_IN1	RDC 触发输入 1
126	SYNCTIMER_TRIG	同步计时器输入触发
127	QEI0_TRIG_IN	QEI0 触发输入
128	QEI1_TRIG_IN	QEI1 触发输入
129	QEI0_PAUSE	QEI0 暂停输入
130	QEI1_PAUSE	QEI0 暂停输入
131	UART_TRIG0	UART0/1/2/3 触发信号
132	UART_TRIG1	UART4/5/6/7 触发信号
133	TRGM_IRQ0	互联管理器中断信号 0
134	TRGM_IRQ1	互联管理器中断信号 1
135	TRGM_DMA0	互联管理器 DMA 请求 0
136	TRGM_DMA1	互联管理器 DMA 请求 1

表 170: TRGM_OUTPUT MUX 列表

32.8.3 互联管理器 DMA 请求

本章节介绍互联管理器 DMA 请求分配。TRGM 支持 4 个 DMA 请求输出，用户可以配置 TRGM，从多个 DMA 请求输入中，选择 4 个连接到 DMAMUX。

互联管理器的 DMA 请求分配：

TRGM_DMA MUX 编号	TRGM_DMA MUX SOURCE 名称	描述
0	PWM0_CMP0	PWM 定时器 0 比较器 0 的输入捕获或者输出比较匹配
1	PWM0_CMP1	PWM 定时器 0 比较器 1 的输入捕获或者输出比较匹配
2	PWM0_CMP2	PWM 定时器 0 比较器 2 的输入捕获或者输出比较匹配
3	PWM0_CMP3	PWM 定时器 0 比较器 3 的输入捕获或者输出比较匹配
4	PWM0_CMP4	PWM 定时器 0 比较器 4 的输入捕获或者输出比较匹配
5	PWM0_CMP5	PWM 定时器 0 比较器 5 的输入捕获或者输出比较匹配
6	PWM0_CMP6	PWM 定时器 0 比较器 6 的输入捕获或者输出比较匹配
7	PWM0_CMP7	PWM 定时器 0 比较器 7 的输入捕获或者输出比较匹配
8	PWM0_CMP8	PWM 定时器 0 比较器 8 的输入捕获或者输出比较匹配
9	PWM0_CMP9	PWM 定时器 0 比较器 9 的输入捕获或者输出比较匹配
10	PWM0_CMP10	PWM 定时器 0 比较器 10 的输入捕获或者输出比较匹配
11	PWM0_CMP11	PWM 定时器 0 比较器 11 的输入捕获或者输出比较匹配
12	PWM0_CMP12	PWM 定时器 0 比较器 12 的输入捕获或者输出比较匹配
13	PWM0_CMP13	PWM 定时器 0 比较器 13 的输入捕获或者输出比较匹配
14	PWM0_CMP14	PWM 定时器 0 比较器 14 的输入捕获或者输出比较匹配
15	PWM0_CMP15	PWM 定时器 0 比较器 15 的输入捕获或者输出比较匹配
16	PWM0_CMP16	PWM 定时器 0 比较器 16 的输入捕获或者输出比较匹配
17	PWM0_CMP17	PWM 定时器 0 比较器 17 的输入捕获或者输出比较匹配
18	PWM0_CMP18	PWM 定时器 0 比较器 18 的输入捕获或者输出比较匹配
19	PWM0_CMP19	PWM 定时器 0 比较器 19 的输入捕获或者输出比较匹配

TRGM_DMA MUX 编号	TRGM_DMA MUX SOURCE 名称	描述
20	PWM0_CMP20	PWM 定时器 0 比较器 20 的输入捕获或者输出比较匹配
21	PWM0_CMP21	PWM 定时器 0 比较器 21 的输入捕获或者输出比较匹配
22	PWM0_CMP22	PWM 定时器 0 比较器 22 的输入捕获或者输出比较匹配
23	PWM0_CMP23	PWM 定时器 0 比较器 23 的输入捕获或者输出比较匹配
24	PWM0_RLD	PWM 定时器 0 计数器重载
25	PWM0_HALFRLD	PWM 定时器 0 半周期器重载
26	PWM0_XRLD	PWM 定时器 0 扩展计数器重载
27	PWM1_CMP0	PWM 定时器 1 比较器 0 的输入捕获或者输出比较匹配
28	PWM1_CMP1	PWM 定时器 1 比较器 1 的输入捕获或者输出比较匹配
29	PWM1_CMP2	PWM 定时器 1 比较器 2 的输入捕获或者输出比较匹配
30	PWM1_CMP3	PWM 定时器 1 比较器 3 的输入捕获或者输出比较匹配
31	PWM1_CMP4	PWM 定时器 1 比较器 4 的输入捕获或者输出比较匹配
32	PWM1_CMP5	PWM 定时器 1 比较器 5 的输入捕获或者输出比较匹配
33	PWM1_CMP6	PWM 定时器 1 比较器 6 的输入捕获或者输出比较匹配
34	PWM1_CMP7	PWM 定时器 1 比较器 7 的输入捕获或者输出比较匹配
35	PWM1_CMP8	PWM 定时器 1 比较器 8 的输入捕获或者输出比较匹配
36	PWM1_CMP9	PWM 定时器 1 比较器 9 的输入捕获或者输出比较匹配
37	PWM1_CMP10	PWM 定时器 1 比较器 10 的输入捕获或者输出比较匹配
38	PWM1_CMP11	PWM 定时器 1 比较器 11 的输入捕获或者输出比较匹配
39	PWM1_CMP12	PWM 定时器 1 比较器 12 的输入捕获或者输出比较匹配
40	PWM1_CMP13	PWM 定时器 1 比较器 13 的输入捕获或者输出比较匹配

TRGM_DMA MUX 编号	TRGM_DMA MUX SOURCE 名称	描述
41	PWM1_CMP14	PWM 定时器 1 比较器 14 的输入捕获或者输出比较匹配
42	PWM1_CMP15	PWM 定时器 1 比较器 15 的输入捕获或者输出比较匹配
43	PWM1_CMP16	PWM 定时器 1 比较器 16 的输入捕获或者输出比较匹配
44	PWM1_CMP17	PWM 定时器 1 比较器 17 的输入捕获或者输出比较匹配
45	PWM1_CMP18	PWM 定时器 1 比较器 18 的输入捕获或者输出比较匹配
46	PWM1_CMP19	PWM 定时器 1 比较器 19 的输入捕获或者输出比较匹配
47	PWM1_CMP20	PWM 定时器 1 比较器 20 的输入捕获或者输出比较匹配
48	PWM1_CMP21	PWM 定时器 1 比较器 21 的输入捕获或者输出比较匹配
49	PWM1_CMP22	PWM 定时器 1 比较器 22 的输入捕获或者输出比较匹配
50	PWM1_CMP23	PWM 定时器 1 比较器 23 的输入捕获或者输出比较匹配
51	PWM1_RLD	PWM 定时器 1 计数器重载
52	PWM1_HALFRLD	PWM 定时器 1 半周期器重载
53	PWM1_XRLD	PWM 定时器 1 扩展计数器重载
54	QEI0	正交解码器 0 的 DMA 请求
55	QEI1	正交解码器 1 的 DMA 请求
56	MMC0	MMC0 的 DMA 请求
57	MMC1	MMC1 的 DMA 请求
58	SEI0	SEI 的 DMA 请求 0
59	SEI1	SEI 的 DMA 请求 1
60	TRGM0	TRGM 的 DMA 请求 0，来自于互联管理器输出 135
61	TRGM1	TRGM 的 DMA 请求 1，来自于互联管理器输出 136

表 171: TRGM_DMA MUX 列表

32.8.4 互联管理器数字滤波器

本章节介绍互联管理器数字滤波器和输入信号对应情况。

互联管理器的数字滤波器分配：

TRGM_FILTER MUX 编号	TRGM_FILTER MUX SOURCE 名称	描述
0	PWM0_IN0	PWM 定时器 0 输入捕获 0
1	PWM0_IN1	PWM 定时器 0 输入捕获 1
2	PWM0_IN2	PWM 定时器 0 输入捕获 2
3	PWM0_IN3	PWM 定时器 0 输入捕获 3
4	PWM0_IN4	PWM 定时器 0 输入捕获 4
5	PWM0_IN5	PWM 定时器 0 输入捕获 5
6	PWM0_IN6	PWM 定时器 0 输入捕获 6
7	PWM0_IN7	PWM 定时器 0 输入捕获 7
8	PWM1_IN0	PWM 定时器 1 输入捕获 0
9	PWM1_IN1	PWM 定时器 1 输入捕获 1
10	PWM1_IN2	PWM 定时器 1 输入捕获 2
11	PWM1_IN3	PWM 定时器 1 输入捕获 3
12	PWM1_IN4	PWM 定时器 1 输入捕获 4
13	PWM1_IN5	PWM 定时器 1 输入捕获 5
14	PWM1_IN6	PWM 定时器 1 输入捕获 6
15	PWM1_IN7	PWM 定时器 1 输入捕获 7
16	TRGM_IN0	互联管理器输入 0
17	TRGM_IN1	互联管理器输入 1
18	TRGM_IN2	互联管理器输入 2
19	TRGM_IN3	互联管理器输入 3
20	TRGM_IN4	互联管理器输入 4
21	TRGM_IN5	互联管理器输入 5
22	TRGM_IN6	互联管理器输入 6
23	TRGM_IN7	互联管理器输入 7
24	PWM0_FAULT0	PWM 定时器 0 输入 fault0
25	PWM0_FAULT1	PWM 定时器 0 输入 fault1
26	PWM1_FAULT0	PWM 定时器 1 输入 fault0
27	PWM1_FAULT1	PWM 定时器 1 输入 fault1

表 172: TRGM_FILTER MUX 列表

32.9 同步定时器 SYNT

本产品支持 1 个同步定时器 SYNT，SYNT 支持 32 位计数器，和 4 个输出比较通道，输出信号连接到本产品上的 2 个电机控制单元 TRGM 的输入。用作各个电机控制单元内外设的同步。

同步定时器 SYNCT 的输出信号连接，请查阅小节 32.8.1。

33 PWM 定时器 PWM

本章节介绍本产品 PWM 定时器的主要功能和特性。

33.1 特性总结

本章节介绍 PWM 控制器的主要特性：

- 28 (24 +4) 位分辨率计数器，支持向上计数模式
- 支持计数器同步
- 多达 24 个比较器，支持用作输出比较，或者输入捕获
- 多达 24 个通道，其中通道 0~7 可用于 PWM 输出
  - 支持 8 路独立或者 4 对互补 PWM 输出
  - 互补 PWM 支持死区插入，支持独立配置双侧死区宽度
  - 支持把 PWM 输出强制设置为指定状态
  - 支持故障保护输入，在出错时（如故障保护输入时），单独配置每个 PWM 输出通道的状态
- 支持为每个输出通道灵活地分配数目不等的比较器，灵活控制输出信号，生成例如边沿对齐 PWM、左右不对称的中央对齐 PWM 以及更复杂的输出信号
- 支持生成各类 DMA 请求和中断请求
- 部分寄存器配有影子寄存器，支持灵活的寄存器新值更新/生效时机

PWM 的框图如图 26。

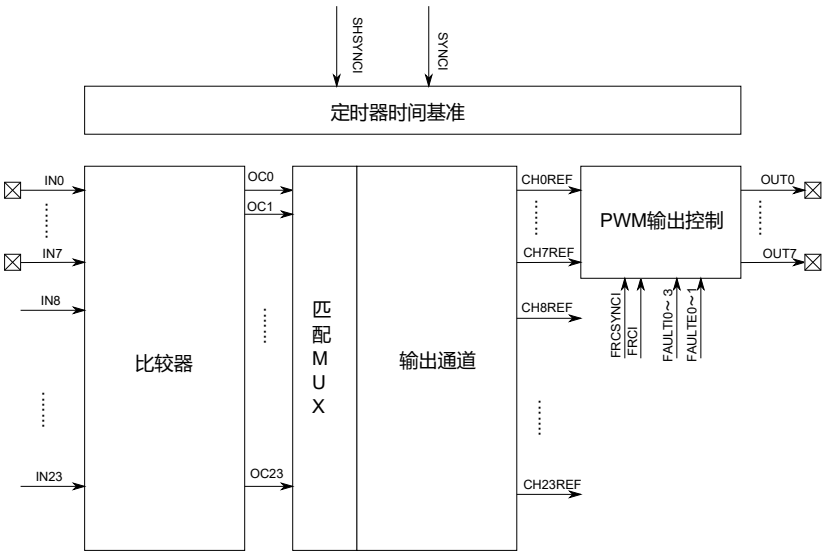


图 26: PWM 定时器框图

33.2 功能描述

本章节描述 PWM 定时器各个子模块的功能。

33.2.1 定时器时间基准

定时器时间基准模块的作用是决定 PWM 定时器运行的时间和周期。

它包含以下几部分：

- 计数器，计数器包括计数器和扩展计数两部分，计数器 24 位，扩展 4 位。可以合并成 28 位计数器使用
- 起始寄存器，可以设置计数器起始值和扩展起始值
- 重载寄存器，可以设置计数器重载值和扩展重载值

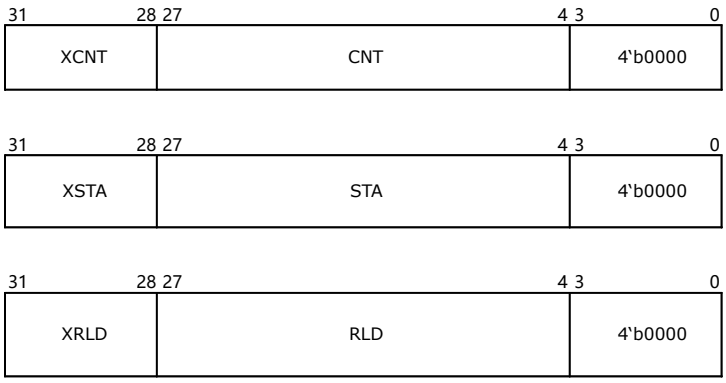


图 27: PWM 的时间基准模块

计数器是一个由 24 位计数器（CNT）和 4 位扩展计数器（XCNT）组成，可以扩展为 28 位计数器。

起始寄存器STA可以设置 24 位计数器的起始值（STA），以及 4 位扩展计数器的起始值（XSTA）。

重载寄存器RLD可以设置 24 位计数器的重载值（RLD），以及 4 位扩展计数器的重载值（XRLD）。

计数器，起始寄存器和重载寄存器分别占用 32 位的寄存器空间，其中低 4 位为 0。

计时器支持向上计数。

计数器使能 (GCR[CEN] 位置 1) 以后，总是从起始值开始计数，当计时器的值（CNT）计数至重载值（RLD）后，重载标志RLDF位置 1，此时，扩展计数器值 +1，计数器的值恢复到起始值（STA）。当扩展计数器值（XCNT）也计数到扩展重载值（XRLD）后，重载标志XRLDF位置 1，扩展计数器恢复到起始值。

如果把计数器起始值设置为 0，重载值设置为 x，实际定时器的计时周期为 x+1 个时钟周期。如果把计数器起始值 STA 设置为 24'h000000，重载值 RLD 设置为 24'hFFFFFF, 扩展起始值 XSTA 设置为 4'h0，扩展重载值 XRLD 设置为 4'hF。效果等同于把计数器由 24 位扩展为 28 位。

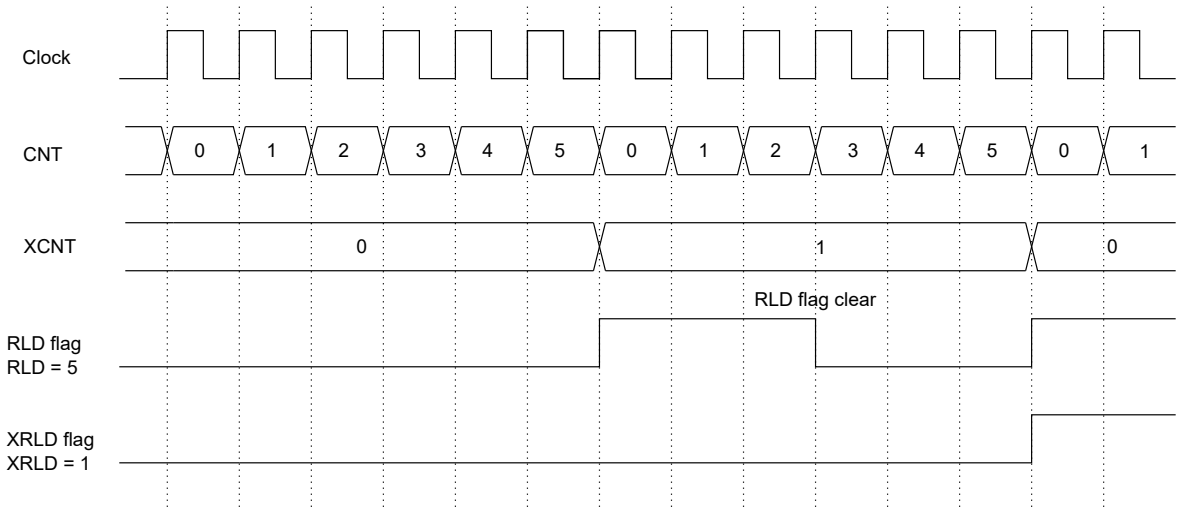


图 28: PWM 计数器计数与重载、扩展重载标志位置位示意图

用户配置完成STA和RLD寄存器后，将GCR[CEN] 位置 1，计数器 CNT 开始计数。

用户也可以利用 PWM 定时器的同步触发输入（SYNCI）来同步计数器开始计数的时机。用户可以配置GCR [RLDSYNCEN] 和 GCR [XRLDSYNCEN] 位来选择是否打开外部同步。GCR [RLDSYNCEN] 位置 1 时，同步触发输入（SYNCI）同步触发输入可以重置计数器到起始值（CNT = STA），重载标志位RLDF置 1。如果GCR [RLDSYNCEN] 位置 1，计数器的扩展位重置到起始值的扩展位（XCNT = XSTA），扩展重载标志位 XRLDF 也会置 1。

多电机协同工作时，可以使用同步定时器 SYNT，产生同步脉冲，让多个 PWM 同时开始计数，或者间隔指定的时间开始计数。

计数器模块还支持一个半重载标志位HALFRLDF，当计数器计数达到（RLD - STA）一半时（注意，不包括 XRLD 和 XSTA），该位置 1。

控制模块管理如下标志位

- RLDF：当计数器（不包括扩展计数位）计数至重载寄存器的值时该位置 1，或者由 SYNCI 将计数器重置时，该位也置 1
- XRLDF：当扩展计数器扩展计数位计数至扩展重载计数位时，该位置 1，或者由 SYNCI 将计数器重置时，该位也置 1
- HALFRLDF：当计数器计数至起始值和重载值中间时，该位置 1

33.2.2 PWM 生成

PWM 生成需要配合使用比较器和通道，主要功能是以通道为单位，利用比较器组合生成输出参考信号。

如下图所示，PWM 生成模块包括 24 个比较器和 24 个输出通道。

其中通道 0 到通道 7 是 PWM 输出通道，通道输出 CH0REF ~ CH7REF 连接到 PWM 控制逻辑，如互补控制，死区生成，故障保护等模块。最终输出信号 OUT0~OUT7 到芯片的管脚上。

通道 8 及之后的通道是通用输出通道，这些通道的输出参考信号不经过 PWM 控制逻辑，也可以输出信号供内外部使用。

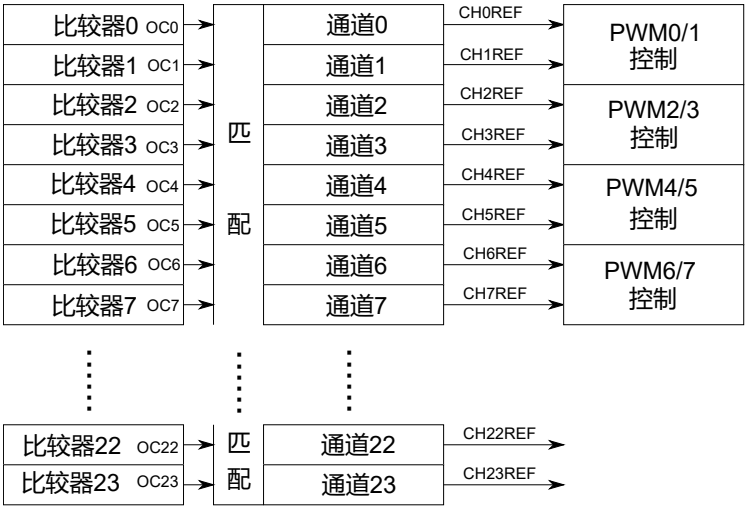


图 29: PWM 定时器的 PWM 生成模块

PWM 定时器的 24 个比较器，当它们用于 PWM 生成时，需要配置为输出比较模式（把 CMPCFGx[CMPMODE] 位置 0）。此时，当计数器的值等于比较寄存器的值时，产生匹配事件。注意比较器和计时器一样，包含 24 位比较位（CMP）和 4 位扩展比较位（XCMP）。可以设置CMPCFGx[XCNTCMPEN] 位，选择 4 位扩展位是否参与比较。

- 当XCNTCMPEN[3:0] 位都置 0 时，4 位扩展比较位不参与比较。24 位计数器计数达到 24 位比较位，即 CNT == CMPx，产生匹配事件
- 当XCNTCMPEN[3:0] 位中某一位或者某几位置 1 时，对应的扩展比较位会和计时器的对应扩展计数位比较。比如XCNTCMPEN[3:0] = 4'b1111 时，4 位扩展计数位达到 4 位扩展比较位，并且 24 位计数器计数达到 24 位比较位，即 XCNT == XCMP && CNT == CMP，产生匹配事件。

比较器寄存器还包含 4 位为小数比较位（CMPHLF和CMPJIT），设置这些位可以使得比较寄存器生成匹配事件的精度小于一个 PWM 定时器时钟周期。CMPHLF是半周期比较位，由 PWM 定时器时钟的下边沿实现，该位置 1，可以使匹配事件的生成时间延后 1/2 时钟周期。CMPJIT是抖动比较位，由 4，8，16 个周期边沿抖动，平均后实现精度达 1/4，1/8 和 1/16 时钟周期的延时效果。

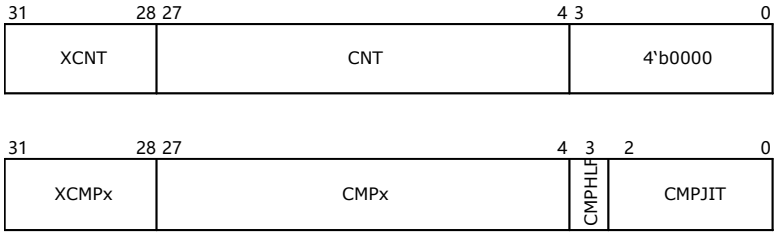


图 30: PWM 比较器对应计数器位域图

用户可以把一个或者多个连续的比较器分配给某个通道,实现灵活复杂的输出。用户通过设置 CHCFGx [CMPSELBEG] 位，选择分配给通道 x 的比较寄存器起始序号。通过设置 CHCFGx [CMPSELEND] 位，选择分配给通道 x 的比较寄存器末尾序号。

一个比较器可以同时分配给多个通道。

例如，以下是某个应用的配置，有 3 个 pwm 输出和两个控制信号输出：

设置 CHCFG[0][CMPSELBEG] = 5'b00000，CHCFG[0][CMPSELEND] = 5'b00010，表示把比较寄存器 0~2 分配给通道 0。

设置 CHCFG[1][CMPSELBEG] = 5'b00011，CHCFG[1][CMPSELEND] = 5'b00011，表示把比较寄存器 3 分配给通道 1。

设置 CHCFG[5][CMPSELBEG] = 5'b00100，CHCFG[5][CMPSELEND] = 5'b00111，表示把比较寄存器 4~7 分配给通道 5。

设置 CHCFG[8][CMPSELBEG] = 5'b00100，CHCFG[8][CMPSELEND] = 5'b00101，表示把比较寄存器 4~5 分配给通道 8。

设置 CHCFG[9][CMPSELBEG] = 5'b00111，CHCFG[9][CMPSELEND] = 5'b00111，表示把比较寄存器 7 分配给通道 9。

当计数器 CNT 计数达到比较器 CMPx 配置的CMP或者XCMP时，产生匹配事件，此时 OCx 输出置逻辑 1。当计数器 CNT 值到达重载寄存器，发生重载事件，输出重置逻辑 0。

如果比较器值 CMPx 等于重载值 RLD，OCx 会保持输出逻辑 1。因此设置 CMPx = RLD 可用来生成 100% 占空比的 PWM 输出。

当比较器值大于重载值，即 CMPx > RLD，由于计数器值 CNT 始终达不到 CMPx，比较器输出 OCx 会保持逻辑 0。

注意: 比较器的输出 OCx 为 PWM 信号，PWM 的占空比由 CMPx 和 RLD 共同决定，设置 CMPx == RLD 可以得到占空比 100% 的 PWM，设置 CMPx > RLD 可以得到占空比 0% 的 PWM。

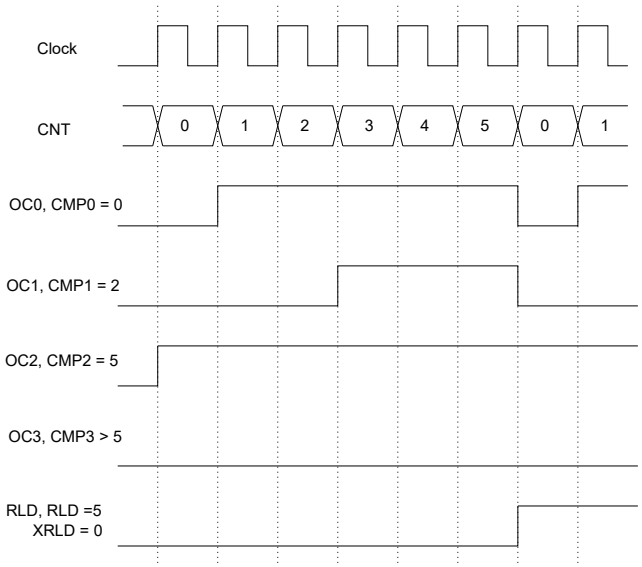


图 31: PWM 计数器计数与重载、扩展重载标志位置位示意图

通道 x 的输出参考信号 CHxREF，由分配给它的全部比较器输出 OC_{BEG} ~OC_{END} 异或后得到。即

$$CHxREF = OC_{BEG} \oplus OC_{BEG+1} \oplus OC_{BEG+2} \oplus \cdots \oplus OC_{END}$$

33.2.3 PWM 生成举例

本章节给出几个通过配置比较器，结合输出通道生成特定 PWM 输出的例子。

以下是在通道 0（CH0REF）和通道 1（CH1REF）输出上生成边沿对齐 PWM 的例子。

分配比较器 0 到通道 0，即 CHCFG0[CMPSELBEG] = 0，CHCFG0[CMPSELEND] = 0

分配比较器 1 到通道 1，即 CHCFG1[CMPSELBEG] = 1，CHCFG1[CMPSELEND] = 1

设置比较器 0~1 为电平输出模式。设置比较器 0~1 的值 $STA < CMP0 < CMP1 < RLD$。以此可以得到 CH0REF，CH1REF 输出如图 32：

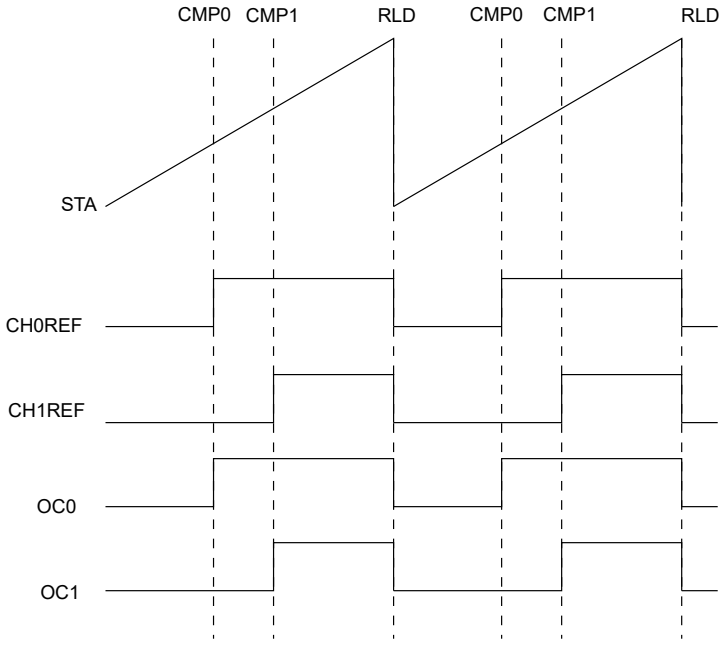


图 32: 边沿对齐 PWM 生成示例图

以下是在通道 0（CH0REF）和通道 1（CH1REF）输出上生成中心对齐相移 PWM 的例子。

分配比较器 0，1 到通道 0，即 CHCFG0[CMPSELBEG] = 0，CHCFG0[CMPSELEND] = 1

分配比较器 2，3 到通道 1，即 CHCFG1[CMPSELBEG] = 2，CHCFG1[CMPSELEND] = 3

设置比较器 0~3 为电平输出模式。

设置比较器 0~3 的值 $STA < CMP2 < CMP0 < CMP3 < CMP1 < RLD$。以此可以得到 CH0REF，CH1REF 输出如图 33：

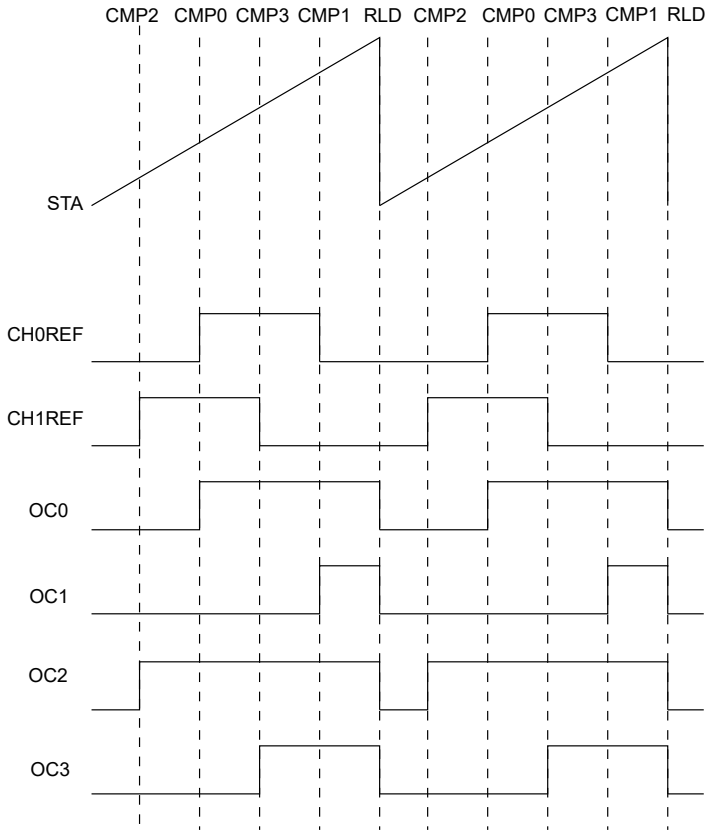


图 33: 中心对齐相移 PWM 生成示例图

以下是在通道 0（CH0REF）输出上生成双翻转 PWM 的例子。

分配比较器 0 ~ 比较器 3 到通道 0，即 CHCFG0[CMPSELBEG] = 0，CHCFG0[CMPSELEND] = 3。

设置比较器 0~3 为电平输出模式。

设置比较器 0~3 的值 $STA < CMP0 < CMP1 < CMP2 < CMP3 < RLD$。

以此可以得到 CH0REF 输出如图 34：

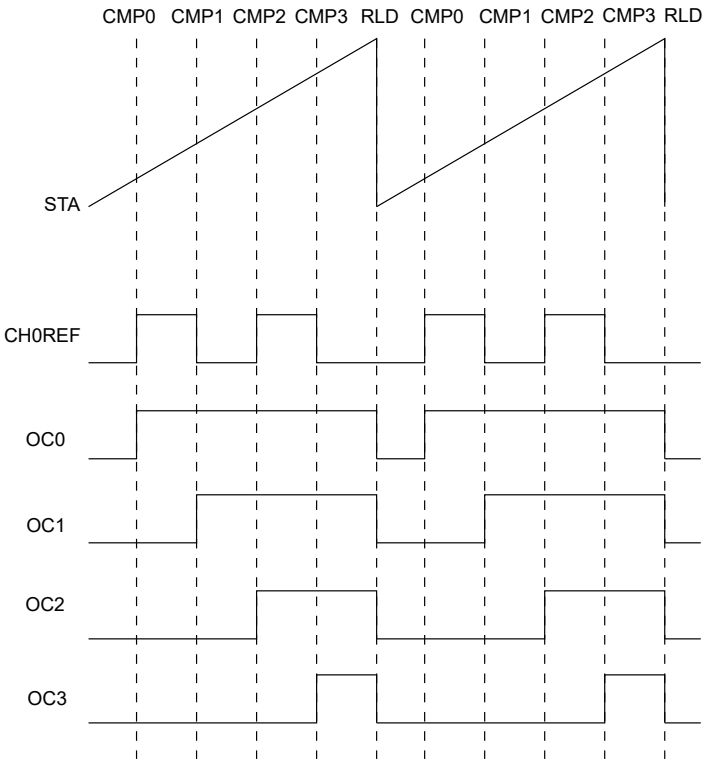


图 34: 双翻转 PWM 生成示例图

33.2.4 PWM 输出控制概述

PWM 输出通道（通道 0~7）的参考信号（CH0REF~CH7REF）经过后续的互补控制，死区插入，取反控制，强制输出，故障保护后，形成输出信号（OUT0~OUT7）到 IO。这些 PWM 控制逻辑可以通过 PWMCFG[x] 和 CHCFG[x] 寄存器内的控制位配置。

PWM 控制逻辑，以相邻的一对 PWM 输出为例，如图 35所示。

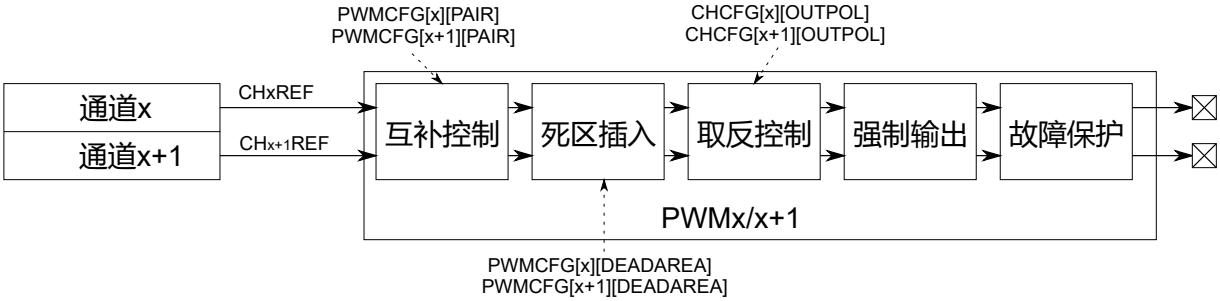


图 35: PWM 输出控制示例图

33.2.5 PWM 互补控制

PWM 定时器支持生成成对的互补 PWM 输出。当把寄存器 PWMCFGx[PAIR] 位置 1 时，可以把 PWM0 和 PWM1，PWM2 和 PWM3，PWM4 和 PWM5，PWM6 和 PWM7 设置为互补的 PWM 输出。注意，一定把成对的 PWMCFGx[PAIR] 寄存器位都置 1，互补输出才会生效。一旦设置成 PWM 互补输出，偶数序号的 PWM 通道配置会生效，奇数序号的 PWM 通道输出为偶数通道的输出取反。

比如，当 PWM0 和 PWM1 配置为互补输出时，PWM0 输出（OUT0）为通道 0 的输出参考信号（CH0REF），

PWM1（OUT1）则是 PWM0 取反。

33.2.6 死区控制

PWM 定时器配置成输出一对互补的 PWM 输出时，用户可以通过死区控制模块，在成对的输出参考信号的翻转之间插入一定的延时，避免受 PWM 输出控制的外部开关同时导通。

PWM 定时器允许用户在 2 路互补的参考信号上插入的不同的延时。这样用户可以根据片外开关器件的特性，优化死区时间。

用户可以通过配置 PWMCFGx[DEADAREA] 位来配置死区的长度，必须同时配置成对的通道 x 和通道 x+1 寄存器才能正确地生成死区。

例如，当把 PWM 输出 0 和输出 1 配置成互补的 PWM 输出时，需要配置：PWMCFG0[DEADAREA] 可以把通道 0 参考信号的每一个上升沿推迟若干个时钟周期。PWMCFG1[DEADAREA] 可以把通道 1 参考信号的每一个上升沿推迟若干个时钟周期。

注意，PWMCFGx[DEADAREA] 配置死区长度的单位是 0.5 个 PWM 定时器时钟周期。死区支持的最小长度是 1 个时钟周期。
即：PWMCFGx[DEADAREA] = 20'h00003，表示死区长度为 1.5 个时钟周期
PWMCFGx[DEADAREA] = 20'h000A0，表示死区长度为 80 个时钟周期

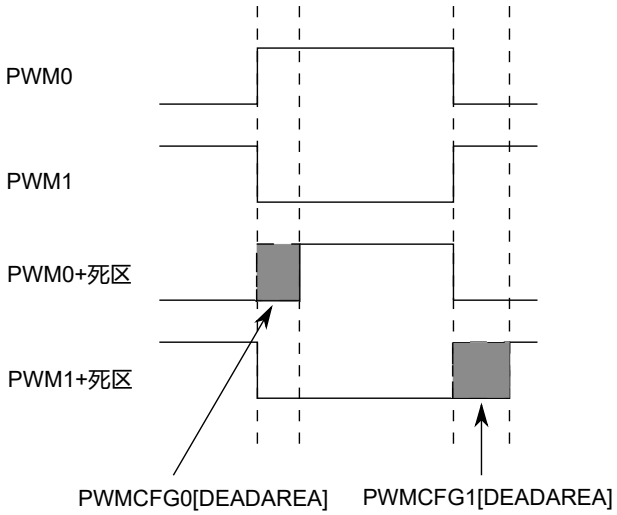


图 36: PWM 死区控制示意图

33.2.7 输出取反

外部器件打开以及关断所需要的可以是高电平也可以是低电平。为了适应各种外部器件，PWM 定时器的全部通道输出可以通过设置 CHCFGx[OP] 位，来配置输出极性：

- 置 0 时，输出为 CHxREF 不变
- 置 1 时，输出为 CHxREF 取反

33.2.8 强制输出控制

强制输出控制模块可以作用于通道 0~7 的参考信号，允许用户把参考信号配置成指定的状态。此模块可以方便生成驱动 BLDC 电机需要的 6 步换相 PWM 输出。

通过配置 PWMCFGx[FRCSRCSEL] 位，可以自由选择 PWM 输出 0~7 强制调试是由硬件触发还是由软件触

发

- 该位置 0 时，表示 PWM 输出 x 强制输出由外部输入 FRCI 控制，可以通过 GCR[FRCPOL] 位选择 FRCI 的有效极性，GCR[FRCPOL] = 1'b0，表示 FRCI 逻辑 1 有效，否则逻辑 0 有效。
- 该位置 1 时，并且软件将 GCR[SWFRC] 位置 1 时，强制输出有效。

通过配置寄存器 GCR[FRCTIME]，用户可以选择输出生效的时机。

- 2'b00，即时生效
- 2'b01，强制输出在使能后，计数器重载标志位 RLD 置 1 时生效
- 2'b10，强制输出在使能后，同步输入触发（FRCSYNCI）上捕获到上升沿时生效
- 2'b11，强制输出不生效

注意，GCR[FRCTIME] 同样控制强制输出失效的时机。即强制输出关闭后，输出失效的时机也由其控制。

通过配置寄存器 FRCMD[MODEx]，可以配置强制输出生效时，通道 0~7 输出的状态：

- 2'b00，强制输出逻辑 0
- 2'b01，强制输出逻辑 1
- 2'b10，关闭输出，引脚变为高阻 HiZ
- 2'b11，不进行强制输出，即输出保持通道 x 的参考信号 CHxREF 不变

注意，FRCMD 寄存器配备有影子寄存器，直接写 FRCMD 寄存器并不一定会立即生效。如果用户希望改变某个 PWM 输出的强制输出状态，比如从强制输出逻辑 0 改为输出逻辑 1，其生效时间由PWMCFG[x] [FRCSHDWUPT] 决定，具体请参考小节 33.2.12。

33.2.9 故障保护

PWM 定时器允许用户在紧急情况下实时快速地切断 PWN 输出信号。

PWM 定时器支持内外部共 6 个故障保护输入信号，其中 4 个内部故障输入信号（FAULTI0~FAULTI3），和 2 个外部故障输入信号（FAULTE0，FAULTE1）。

PWM 定时器外部故障输入信号（FAULTE0，FAULTE1），在 PWM 定时器时钟丢失的情况下，也可以发挥作用。用户可以设置 GCR[FAULTExPOL] 位来选择外部故障输入信号是逻辑 1 生效，还是逻辑 0 生效。

- 1'b0，FAULTEx 逻辑 1 时生效
- 1'b1，FAULTEx 逻辑 0 时生效

用户可以通过配置 GCR[FAULTExEN] 位，打开或者关闭 PWM 定时器的外部故障输入：

- 1'b0，关闭故障保护，PWM 输出不响应外部故障输入信号 x（FAULTEx）
- 1'b1，使能故障保护，按照 GCR[FAULTExPOL] 位的设置，在故障输入信号（FAULTIx）为逻辑 1 或者逻辑 0 时生效

用户可以通过配置 GCR[FAULTIxEN] 位，打开或者关闭 PWM 定时器的内部故障输入。

- 1'b0，关闭故障保护，PWM 输出不响应内部故障输入信号 x（FAULTIx）
- 1'b1，使能故障保护，内部故障输入在逻辑 1 时生效

在故障输入生效时，PWM 定时器可以把通道 0~ 通道 7 的输出（OUT0 ~ OUT7）强制到某个安全的既定状态。

用户可以选择打开一个或者多个乃至全部故障输入信号。在打开的故障输入中，任意一个生效时，故障保护即有效。

通过配置 PWMCFGx[FAULTMODE]，PWM0~7 输出都可以独立配置为：

- 2'b0x，当故障发生时，关闭输出，引脚变为高阻 HiZ
- 2'b10，当故障发生时，输出变为逻辑 0
- 2'b11，当故障发生时，输出变为逻辑 1

一旦故障发生，那么在系统状态变化之前，PWM 输出都不会恢复。

用户可以通过设置 PWMCFGx[FAULTRECTIME]，选择 PWM 恢复的时机：

- 2'b00，故障恢复后，立即输出恢复。
- 2'b01，故障恢复后，计数器重载标志位 RLD 置 1 时输出恢复。
- 2'b10，故障恢复后，计数器与某一个 CMP 发生匹配后输出恢复，可以通过 GCR [FAULTRECHWSEL] 位从 24 个比较器选择其一，作为故障恢复时机。
- 2'b11，故障恢复后，软件将 GCR[FAULTCLR] 位置 1 后输出恢复

33.2.10 Debug 模式支持

PWM 定时器允许用户进入调试模式时，断开 PWM 输出，以此来保护外部功率器件。

可以设置 GCR[DEBUGFAULT] 位，该位置 1 时，一旦芯片进入调试模式（Debug 模式），PWM 输出状态会如同故障保护一样，强制到指定的状态。管脚的具体状态如同故障保护，由 PWMCFGx[FAULTMODE] 位的设置决定。

33.2.11 输入捕获模块

PWM 定时器的各个比较器，也可以用作输入捕获模式。

注意，一旦某个比较器配置为输入捕获模式，它就不能再被分配给通道，用作输出。

用户可以配置 CMPCFGx[CMPMODE] 位来选择比较器的工作模式：

- 1'b0，输出比较模式，比较器可以被分配给通道，用作输出
- 1'b1，输入捕获模式，此时，比较器用来捕获输入信号的翻转，并在翻转时保存计数器的值（CNT 和 XCNT）。

把比较器配置成输入捕获模式后，用户可以从寄存器 CAPPOSx 寄存器读取到 CMPx 在信号（INx）上升沿捕获到的计数器值；在 CAPNEGx 寄存器读取到 CMPx 在下降沿捕获到的计数器值。CAPPOSx 和 CAPNEGx 寄存器即会保存计数器寄存器的 24 位计数器（CNT）值，也会保存 4 位扩展计数器（XCNT）值。

注意，CAPPOSx 寄存器和 CAPNEGx 寄存器的值在捕获到新的边沿时会刷新，之前捕获的值会丢失。

33.2.12 影子寄存器

PWM 定时器的部分寄存器支持影子寄存器（shadow register）。影子寄存器的作用是为定时器的部分关键寄存器提供读写接口。在处理器访问寄存器的时候，实质上改变的是它的影子寄存器，新值并不马上生效。只有在用户指定的时刻，才把影子寄存器的值更新到寄存器，并以此改变 PWM 定时器的工作状态。

PWM 定时器的以下三组寄存器支持影子寄存器：

- 计数器的起始寄存器 STA（包括 STA/XSTA）和重载寄存器 RLD（包括 RLD/XRLD）
- 比较器 CMPx，当其在用作输出比较时
- 控制寄存器中的 FRCMD[FRCMD] 位，即通道 0~7（PWM 通道）的强制输出控制模式位

以上这些寄存器和寄存器位，用户对它们进行写操作的时候，是不会即刻生效的。需要等到某个指定的时刻，统一生效。

对于计数器的起始寄存器 STA 和计数器的重载寄存器 RLD，它们的影子寄存器值（包括 STA，XSTA，RLD，XRLD）生效时间可以通过 SHCR[CNTSHDWUPT] 位设置：

- 2'b00，由软件把 SHLK [SHLK] 位置 1 后生效
- 2'b01，实时生效，在寄存器写之后，一个周期内生效
- 2'b10，定时器的某一个 CMP 发生匹配后生效，用户可以通过 SHCR [CNTSHDWSEL] 从比较器 0~23 中选择一个，匹配可以是该比较器的输出比较，也可以是输入捕获。用户可以选择把选中的比较器 CMPx 的值设为与 RLD 或 xRLD 相等，达到一个完整的 PWM 周期后更新影子寄存器的目的。
- 2'b11，影子寄存器重载触发输入 SHRLDSYNCI 上捕获到上升沿时

PWM 定时器的比较器 CMPx，它们的影子寄存器值生效时刻可以通过 CMPCFGx[CMPSHDWUPT] 位设置：

- 2'b00，由软件把 SHLK [SHLK] 位置 1 后生效
- 2'b01，实时生效，在寄存器写之后，一个周期内生效
- 2'b10，定时器的某一个 CMP 发生匹配后生效，用户可以通过 GCR [CMPSHDWSEL] 从比较器 0~23 中选择一个，匹配可以是该比较器的输出比较，也可以是输入捕获。用户可以选择把选中的比较器 CMPx 的值设为与 RLD 或 xRLD 相等，达到一个完整的 PWM 周期后更新影子寄存器的目的。
- 2'b11，影子寄存器重载触发输入 SHRLDSYNCI 上捕获到上升沿时

注意，在 PWM 定时器工作过程中，如果通过更新比较器 CMPx 来实时改变 PWM 输出波形占空比，推荐用户将 CMPx 的 CMPCFGx[CMPSHDWUPT] 位配置为 2'b10 或者 2'b11，即由硬件指定 CMPx 的影子寄存器的生效时刻。避免软件在不恰当的时机更改配置，导致输出波形异常。

通道 0~7（PWM 通道）的强制输出模式位的影子寄存器位生效时间，可以通过 PWMCFGx[FRCSHUPT] 位设置：

- 2'b00，由软件把 SHLK[SHLK] 位置 1 后生效
- 2'b01，实时生效，在寄存器写之后，一个周期内生效
- 2'b10，定时器的某一个 CMP 发生匹配后生效，用户可以通过 SHCR [FRCSHDWSEL] 从比较器 0~23 中选择一个，匹配可以是该比较器的输出比较，也可以是输入捕获。用户可以选择把选中的比较器 CMPx 的值设为与 RLD 或 xRLD 相等，达到一个完整的 PWM 周期后更新影子寄存器的目的。
- 2'b11，影子寄存器重载触发输入 SHRLDSYNCI 上捕获到上升沿时

此外，影子寄存器支持写保护，用户可以把 SHCR[SHLKEN] 位置 1，之后把 SHclk[SHLK] 位也置 1。这样，软件对任意影子寄存器的写操作都无效。

用户可以通过对 UNLK 寄存器写入神奇代码 0xB0382607 来解锁影子寄存器。解锁之后，可以恢复对所有影子寄存器的写操作。

用户在更新多个影子寄存器的过程中，有可能软件或 DMA 只更新了部分影子寄存器，影子寄存器的生效时刻已到，导致出错。用户可以通过以下操作步骤避免这种情况：

- 把 SHCR[SHLKEN] 位置 1，之后把 SHLK[SHLK] 位也置 1，打开影子寄存器写保护
- 对 UNLK 寄存器写入神奇代码 0xB0382607 来解锁影子寄存器的写保护
- 更新所有需要更新的影子寄存器
- 再把 SHCR[SHLK] 位置 1，此时，重新锁定影子寄存器写保护。
- 之后在下一个影子寄存器生效事件发生时，新的寄存器值生效。注意，如果把生效时机控制位设为 2'b00，即由软件把 SHLK[SHLK] 位置 1 后生效，那么用户要再把 SHLK[SHLK] 位置 1 一次，使影子寄存器新值生效。

33.2.13 中断和 DMA

PWM 定时器支持生成以下中断：

- RLD 当重载事件产生时，即计数器计数达到重载寄存器值，或者由同步触发输入 SYNCI 引发计数器重载
- XRLD：当扩展计数器扩展计数位计数至扩展重载计数位时，该位置 1，或者由 SYNCI 将计数器重置时
- HALFRLD：当计数器计数至起始寄存器到重载寄存器的中间时
- 当比较器设置为输出比较，发生匹配事件时
- 当比较器设置为输入捕获，并在输入通道捕获到指定边沿跳变时
- 故障保护生效时

PWM 定时器支持生成以下 DMA 请求：

- RLD 当重载事件产生时，即计数器计数达到重载寄存器值，或者由同步触发输入 SYNCI 引发计数器重载
- XRLD：当扩展计数器扩展计数位计数至扩展重载计数位时，该位置 1，或者由 SYNCI 将计数器重置时
- HALFRLD：当计数器计数至起始寄存器到重载寄存器的中间时
- 当比较器设置为输出比较，发生匹配事件时
- 当比较器设置为输入捕获，并在输入通道捕获到指定边沿跳变时

33.3 PWM 寄存器

PWM 的寄存器列表如下：

PWM0 base address: 0xF0318000

PWM1 base address: 0xF031C000

地址偏移	名称	描述	复位值
0x0000	UNLK	影子寄存器解锁寄存器	0x00000000
0x0004	STA	计数器起始值寄存器	0x00000000
0x0008	RLD	计数器重载值寄存器	0x00000000
0x000C	CMP[0]	比较器寄存器	0x00000000
0x0010	CMP[1]	比较器寄存器	0x00000000
0x0014	CMP[2]	比较器寄存器	0x00000000
0x0018	CMP[3]	比较器寄存器	0x00000000
0x001C	CMP[4]	比较器寄存器	0x00000000
0x0020	CMP[5]	比较器寄存器	0x00000000
0x0024	CMP[6]	比较器寄存器	0x00000000
0x0028	CMP[7]	比较器寄存器	0x00000000
0x002C	CMP[8]	比较器寄存器	0x00000000
0x0030	CMP[9]	比较器寄存器	0x00000000
0x0034	CMP[10]	比较器寄存器	0x00000000
0x0038	CMP[11]	比较器寄存器	0x00000000
0x003C	CMP[12]	比较器寄存器	0x00000000
0x0040	CMP[13]	比较器寄存器	0x00000000
0x0044	CMP[14]	比较器寄存器	0x00000000

地址偏移	名称	描述	复位值
0x0048	CMP[15]	比较器寄存器	0x00000000
0x004C	CMP[16]	比较器寄存器	0x00000000
0x0050	CMP[17]	比较器寄存器	0x00000000
0x0054	CMP[18]	比较器寄存器	0x00000000
0x0058	CMP[19]	比较器寄存器	0x00000000
0x005C	CMP[20]	比较器寄存器	0x00000000
0x0060	CMP[21]	比较器寄存器	0x00000000
0x0064	CMP[22]	比较器寄存器	0x00000000
0x0068	CMP[23]	比较器寄存器	0x00000000
0x0078	FRCMD	强制输出模式寄存器	0x00000000
0x007C	SHLK	影子寄存器锁定寄存器	0x00000000
0x0080	CHCFG[0]	输出通道配置寄存器	0x00000000
0x0084	CHCFG[1]	输出通道配置寄存器	0x00000000
0x0088	CHCFG[2]	输出通道配置寄存器	0x00000000
0x008C	CHCFG[3]	输出通道配置寄存器	0x00000000
0x0090	CHCFG[4]	输出通道配置寄存器	0x00000000
0x0094	CHCFG[5]	输出通道配置寄存器	0x00000000
0x0098	CHCFG[6]	输出通道配置寄存器	0x00000000
0x009C	CHCFG[7]	输出通道配置寄存器	0x00000000
0x00A0	CHCFG[8]	输出通道配置寄存器	0x00000000
0x00A4	CHCFG[9]	输出通道配置寄存器	0x00000000
0x00A8	CHCFG[10]	输出通道配置寄存器	0x00000000
0x00AC	CHCFG[11]	输出通道配置寄存器	0x00000000
0x00B0	CHCFG[12]	输出通道配置寄存器	0x00000000
0x00B4	CHCFG[13]	输出通道配置寄存器	0x00000000
0x00B8	CHCFG[14]	输出通道配置寄存器	0x00000000
0x00BC	CHCFG[15]	输出通道配置寄存器	0x00000000
0x00C0	CHCFG[16]	输出通道配置寄存器	0x00000000
0x00C4	CHCFG[17]	输出通道配置寄存器	0x00000000
0x00C8	CHCFG[18]	输出通道配置寄存器	0x00000000
0x00CC	CHCFG[19]	输出通道配置寄存器	0x00000000
0x00D0	CHCFG[20]	输出通道配置寄存器	0x00000000
0x00D4	CHCFG[21]	输出通道配置寄存器	0x00000000
0x00D8	CHCFG[22]	输出通道配置寄存器	0x00000000
0x00DC	CHCFG[23]	输出通道配置寄存器	0x00000000
0x00F0	GCR	全局控制寄存器	0x00000000
0x00F4	SHCR	影子寄存器控制寄存器	0x00000000
0x0100	CAPPOS[0]	上升沿捕获寄存器	0x00000000
0x0104	CAPPOS[1]	上升沿捕获寄存器	0x00000000
0x0108	CAPPOS[2]	上升沿捕获寄存器	0x00000000

地址偏移	名称	描述	复位值
0x010C	CAPPOS[3]	上升沿捕获寄存器	0x00000000
0x0110	CAPPOS[4]	上升沿捕获寄存器	0x00000000
0x0114	CAPPOS[5]	上升沿捕获寄存器	0x00000000
0x0118	CAPPOS[6]	上升沿捕获寄存器	0x00000000
0x011C	CAPPOS[7]	上升沿捕获寄存器	0x00000000
0x0120	CAPPOS[8]	上升沿捕获寄存器	0x00000000
0x0124	CAPPOS[9]	上升沿捕获寄存器	0x00000000
0x0128	CAPPOS[10]	上升沿捕获寄存器	0x00000000
0x012C	CAPPOS[11]	上升沿捕获寄存器	0x00000000
0x0130	CAPPOS[12]	上升沿捕获寄存器	0x00000000
0x0134	CAPPOS[13]	上升沿捕获寄存器	0x00000000
0x0138	CAPPOS[14]	上升沿捕获寄存器	0x00000000
0x013C	CAPPOS[15]	上升沿捕获寄存器	0x00000000
0x0140	CAPPOS[16]	上升沿捕获寄存器	0x00000000
0x0144	CAPPOS[17]	上升沿捕获寄存器	0x00000000
0x0148	CAPPOS[18]	上升沿捕获寄存器	0x00000000
0x014C	CAPPOS[19]	上升沿捕获寄存器	0x00000000
0x0150	CAPPOS[20]	上升沿捕获寄存器	0x00000000
0x0154	CAPPOS[21]	上升沿捕获寄存器	0x00000000
0x0158	CAPPOS[22]	上升沿捕获寄存器	0x00000000
0x015C	CAPPOS[23]	上升沿捕获寄存器	0x00000000
0x0170	CNT	计数器	0x00000000
0x0180	CAPNEG[0]	下降沿捕获寄存器	0x00000000
0x0184	CAPNEG[1]	下降沿捕获寄存器	0x00000000
0x0188	CAPNEG[2]	下降沿捕获寄存器	0x00000000
0x018C	CAPNEG[3]	下降沿捕获寄存器	0x00000000
0x0190	CAPNEG[4]	下降沿捕获寄存器	0x00000000
0x0194	CAPNEG[5]	下降沿捕获寄存器	0x00000000
0x0198	CAPNEG[6]	下降沿捕获寄存器	0x00000000
0x019C	CAPNEG[7]	下降沿捕获寄存器	0x00000000
0x01A0	CAPNEG[8]	下降沿捕获寄存器	0x00000000
0x01A4	CAPNEG[9]	下降沿捕获寄存器	0x00000000
0x01A8	CAPNEG[10]	下降沿捕获寄存器	0x00000000
0x01AC	CAPNEG[11]	下降沿捕获寄存器	0x00000000
0x01B0	CAPNEG[12]	下降沿捕获寄存器	0x00000000
0x01B4	CAPNEG[13]	下降沿捕获寄存器	0x00000000
0x01B8	CAPNEG[14]	下降沿捕获寄存器	0x00000000
0x01BC	CAPNEG[15]	下降沿捕获寄存器	0x00000000
0x01C0	CAPNEG[16]	下降沿捕获寄存器	0x00000000
0x01C4	CAPNEG[17]	下降沿捕获寄存器	0x00000000

地址偏移	名称	描述	复位值
0x01C8	CAPNEG[18]	下降沿捕获寄存器	0x00000000
0x01CC	CAPNEG[19]	下降沿捕获寄存器	0x00000000
0x01D0	CAPNEG[20]	下降沿捕获寄存器	0x00000000
0x01D4	CAPNEG[21]	下降沿捕获寄存器	0x00000000
0x01D8	CAPNEG[22]	下降沿捕获寄存器	0x00000000
0x01DC	CAPNEG[23]	下降沿捕获寄存器	0x00000000
0x01F0	CNTCOPY	计数器拷贝	0x00000000
0x0200	PWMCFG[0]	PWM 通道配置寄存器	0x00000000
0x0204	PWMCFG[1]	PWM 通道配置寄存器	0x00000000
0x0208	PWMCFG[2]	PWM 通道配置寄存器	0x00000000
0x020C	PWMCFG[3]	PWM 通道配置寄存器	0x00000000
0x0210	PWMCFG[4]	PWM 通道配置寄存器	0x00000000
0x0214	PWMCFG[5]	PWM 通道配置寄存器	0x00000000
0x0218	PWMCFG[6]	PWM 通道配置寄存器	0x00000000
0x021C	PWMCFG[7]	PWM 通道配置寄存器	0x00000000
0x0220	SR	状态寄存器	0x00000000
0x0224	IRQEN	中断请求使能寄存器	0x00000000
0x022C	DMAEN	DMA 请求使能寄存器	0x00000000
0x0230	CMPCFG[CMPCFG0]	比较器配置寄存器	0x00000000
0x0234	CMPCFG[1]	比较器配置寄存器	0x00000000
0x0238	CMPCFG[2]	比较器配置寄存器	0x00000000
0x023C	CMPCFG[3]	比较器配置寄存器	0x00000000
0x0240	CMPCFG[4]	比较器配置寄存器	0x00000000
0x0244	CMPCFG[5]	比较器配置寄存器	0x00000000
0x0248	CMPCFG[6]	比较器配置寄存器	0x00000000
0x024C	CMPCFG[7]	比较器配置寄存器	0x00000000
0x0250	CMPCFG[8]	比较器配置寄存器	0x00000000
0x0254	CMPCFG[9]	比较器配置寄存器	0x00000000
0x0258	CMPCFG[10]	比较器配置寄存器	0x00000000
0x025C	CMPCFG[11]	比较器配置寄存器	0x00000000
0x0260	CMPCFG[12]	比较器配置寄存器	0x00000000
0x0264	CMPCFG[13]	比较器配置寄存器	0x00000000
0x0268	CMPCFG[14]	比较器配置寄存器	0x00000000
0x026C	CMPCFG[15]	比较器配置寄存器	0x00000000
0x0270	CMPCFG[16]	比较器配置寄存器	0x00000000
0x0274	CMPCFG[17]	比较器配置寄存器	0x00000000
0x0278	CMPCFG[18]	比较器配置寄存器	0x00000000
0x027C	CMPCFG[19]	比较器配置寄存器	0x00000000
0x0280	CMPCFG[20]	比较器配置寄存器	0x00000000
0x0284	CMPCFG[21]	比较器配置寄存器	0x00000000

地址偏移	名称	描述	复位值
0x0288	CMPCFG[22]	比较器配置寄存器	0x00000000
0x028C	CMPCFG[23]	比较器配置寄存器	0x00000000

表 173: PWM 寄存器列表

33.4 PWM 寄存器详细信息

PWM 的寄存器详细说明如下：

33.4.1 UNLK (0x0)

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

UNLK [31:0]

位域	名称	描述
31-0	SHUNLK	对此位域写入 0xB0382607 可以解锁地址偏移从 0x04 到 0x78 的寄存器的影子寄存器。 解锁后，允许写入这些寄存器的影子寄存器。

UNLK 位域

33.4.2 STA (0x4)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
XSTA				STA																								RSVD			
RW				RW																								N/A			
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

STA [31:0]

位域	名称	描述
31-28	XSTA	PWM 定时器的计数器起始位扩展值 当计数器计数至 XRLD 时，计数器值扩展值 XCNT 会重载到 XSTA

位域	名称	描述
27-4	STA	PWM 定时器的计数器起始位 当计数器计数至 RLD 时，计数器值 CNT 会重载到 STA 当计数器重载或者软件写入 SHUNLK 位时，STA 的影子寄存器值会生效

STA 位域

33.4.3 RLD (0x8)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
XRLD				RLD																								RSVD			
RW				RW																								N/A			
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

RLD [31:0]

位域	名称	描述
31-28	XRLD	PWM 定时器的计数器重载位扩展值 当计数器计数至 XRLD 时，计数器值扩展值 XCNT 会重载到 XSTA
27-4	RLD	PWM 定时器的计数器重载值 当计数器计数至 RLD 时，计数器值 CNT 会重载到 STA 当计数器重载或者软件写入 SHUNLK 位时，RLD 的影子寄存器值会生效

RLD 位域

33.4.4 CMP (0xC + 0x4 * m)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
XCMP				CMP																								CMPHLF	CMPJIT		
RW				RW																								RW	RW		
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

CMP [31:0]

位域	名称	描述
31-28	XCMP	比较器值扩展位

位域	名称	描述
27-4	CMP	比较器值 比较器的输出 OCx 默认为 0，在计数器计数到 CMP 时置 1，计数器重载时再清 0
3	CMPHLF	比较器半周期比较位，此位置 1 可以使比较器的精度达到 1/2 时钟周期
2-0	CMPJIT	比较器值抖动位

CMP 位域

33.4.5 FRCMD (0x78)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																FRCMD															
N/A																RW															
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

FRCMD [31:0]

位域	名称	描述
15-0	FRCMD	PWM 输出通道的强制输出模式控制位 00：强制输出 0 01：强制输出 1 10：强制输出高阻 11：不强制输出

FRCMD 位域

33.4.6 SHLK (0x7C)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SHLK	RSVD																														
RW	N/A																														
0	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x

SHLK [31:0]

位域	名称	描述
31	SHLK	影子寄存器锁定位 写入 1 锁定所有的影子寄存器，锁定后不允许写入影子寄存器

SHLK 位域

33.4.7 CHCFG (0x80 + 0x4 * n)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD			CMPSELEND					RSVD			CMPSELBEG					RSVD														OUTPOL	RSVD
N/A			RW					N/A			RW					N/A														RW	N/A
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0

CHCFG [31:0]

位域	名称	描述
28-24	CMPSELEND	比较器选择结尾位 指定分配给输出通道的若干个比较器的结尾序号
20-16	CMPSELBEG	比较器选择起始位 指定分配给输出通道的若干个比较器的起始序号
1	OUTPOL	输出极性 置 1 可以使通道输出取反

CHCFG 位域

33.4.8 GCR (0xF0)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FAULTI3EN	FAULTI2EN	FAULTI1EN	FAULTI0EN	DEBUGFAULT	FRCPOL	RSVD	HWSHDWEDG	CMPSHDWSEL					FAULTRECEDG	FAULTRECHWSEL					FAULTE1EN	FAULTE0EN	FAULTEXPOL		RLDSYNCEN	CEN	FAULTCLR	XRLDSYNCEN	RSVD	TIMERRESET	FRCTIME		SWFRC
RW	RW	RW	RW	RW	RW	N/A	RW	RW					RW	RW					RW	RW	RW		RW	RW	RW	RW	N/A	RW	WO		RW
0	0	0	0	0	0	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	x	0	0	0	0

GCR [31:0]

位域	名称	描述
31	FAULTI3EN	1：使能内部故障输入信号 3（FAULTI3）
30	FAULTI2EN	1：使能内部故障输入信号 2（FAULTI2）
29	FAULTI1EN	1：使能内部故障输入信号 1（FAULTI1）
28	FAULTI0EN	1：使能内部故障输入信号 0（FAULTI0）
27	DEBUGFAULT	1：使能调试模式保护
26	FRCPOL	强制输出硬件控制信号 FRCI 的极性 1：FRCI 高有效 0：FRCI 低有效

位域	名称	描述
24	HWSHDWEDG	比较器影子寄存器比较器生效时机边沿选择 当影子寄存器生效的时机设置为硬件事件（某一个比较器）时，并且该比较器配置为输入捕获模式时，此位控制比较器输入信号的有效边沿： 1：下降沿 0：上升沿
23-19	CMPSHDWSEL	比较器影子寄存器硬件事件生效比较器选择：当 CMPSHDUPDT 设置为 10 时， 此位选择比较器之一，以选中的比较器匹配事件，作为比较器影子寄存器生效的时机。
18	FAULTRECEDG	故障恢复比较器生效时机边沿选择 当故障恢复的时机设置为硬件事件（某一个比较器）时， 并且该比较器配置为输入捕获模式时，此位控制比较器输入信号的有效边沿： 1：下降沿 0：上升沿
17-13	FAULTRECHWSEL	故障恢复比较器选择位 选择比较器之一作为故障恢复的时机。当选中的比较器发生匹配事件时，恢复 PWM 输出。 匹配事件可以是比较器的输出比较或者输入捕获。
12	FAULTE1EN	1：使能外部故障输入信号 1（FAULTE1）
11	FAULTE0EN	1：使能外部故障输入信号 0（FAULTE0）
10-9	FAULTEXPOL	外部故障输入有效极性 1：低电平有效 0：高电平有效
8	RLDSYNCEN	1：使能通过 SYNCI 信号触发 PWM 定时器计数器重载
7	CEN	1：使能 PWM 定时器的计数器 0：关闭 PWM 定时器的计数器
6	FAULTCLR	故障清除位 如果 FAULTRECTIME 设置为 11，对该位写 1，PWM 输出会恢复。 注意，用户只能在故障条件清除之后，新的故障条件发生前对该位写 1
5	XRLDSYNCEN	1：使能通过 SYNCI 信号触发 PWM 定时器计数器和扩展值一同重载
3	TIMERRESET	1：复位计数器（包括 24 位主计时器和 4 位扩展计时器） 硬件自动清零

位域	名称	描述
2-1	FRCTIME	强制输出时机控制位 00：强制输出立即生效 01：强制输出在计数器下一次重载时生效 10：强制输出在 FRCSYNCI 上捕捉到上升沿时生效 11：不生效
0	SWFRC	软件强制输出位，当 FRCSRCSEL 置 0 时 1：使能强制输出 0：关闭强制输出

GCR 位域

33.4.9 SHCR (0xF4)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																			FRCSHDWSEL					CNTSHDWSEL						CNTSHDWUPT		SHLKEN	
N/A																			RW					RW						RW		RW	
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0

SHCR [31:0]

位域	名称	描述
12-8	FRCSHDWSEL	FRCMD 影子寄存器硬件事件生效比较器选择： 当 FRCSHDUPDT 设置为 10 时，此位选择比较器之一，以选中的比较器匹配事件，作为 FRCMD 影子寄存器生效的时机。
7-3	CNTSHDWSEL	RLD，STA 影子寄存器硬件事件生效比较器选择： 当 CNTSHDUPDT 设置为 10 时，此位选择比较器之一，以选中的比较器匹配事件，作为 RLD，STA 影子寄存器生效的时机。
2-1	CNTSHDWUPT	RLD，STA 影子寄存器生效时机选择： 00：由软件把 SHLK [SHLK] 位置 1 后生效 01：实时生效，在寄存器写之后，一个周期内生效 10：定时器的某一个 CMP 发生匹配后生效，用户可以通过 SHCR [CNTSHDWSEL] 从比较器 0 到 23 中选择一个，匹配可以是该比较器的输出比较，也可以是输入捕获。 11：影子寄存器重载触发输入 SHRLDSYNCI 上捕获到上升沿时生效
0	SHLKEN	1：使能影子寄存器锁定，允许锁定影子寄存器 0：关闭影子寄存器锁定，不能锁定影子寄存器

SHCR 位域

33.4.10 CAPPOS (0x100 + 0x4 * n)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CAPPOS																												RSVD			
RO																												N/A			
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	x	x	x	x

CAPPOS [31:0]

位域	名称	描述
31-4	CAPPOS	在输入信号上升沿时，捕获到的计数器值

CAPPOS 位域

33.4.11 CNT (0x170)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
XCNT				CNT																								RSVD			
RO				RO																								N/A			
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

CNT [31:0]

位域	名称	描述
31-28	XCNT	计数器扩展位
27-4	CNT	计数器

CNT 位域

33.4.12 CAPNEG (0x180 + 0x4 * n)

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

CAPNEG [31:0]

位域	名称	描述
31-0	CAPNEG	在输入信号下降沿时，捕获到的计数器值;

CAPNEG 位域

33.4.13 CNTCOPY (0x1F0)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
XCNT				CNT																								RSVD			
RO				RO																								N/A			
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

CNTCOPY [31:0]

位域	名称	描述
31-28	XCNT	计数器扩展位
27-4	CNT	计数器

CNTCOPY 位域

33.4.14 PWMCFG (0x200 + 0x4 * n)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD			OEN	FRCSHDWUPT		FAULTMODE		FAULTRECTIME		FRCSRCSEL	PAIR	DEADAREA																			
N/A			RW	RW		RW		RW		RW	RW	RW																			
x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

PWMCFG [31:0]

位域	名称	描述
28	OEN	PWM 输出使能 1：打开 PWM 输出 0：关闭 PWM 输出
27-26	FRCSHDWUPT	FRCMD 影子寄存器生效时机选择： 00：由软件把 SHLK [SHLK] 位置 1 后生效 01：实时生效，在寄存器写之后，一个周期内生效 10：定时器的某一个 CMP 发生匹配后生效，用户可以通过 SHCR [FRCSHDWSEL] 从比较器 0□23 中选择一个，匹配可以是该比较器的输出比较，也可以是输入捕获。 11：影子寄存器重载触发输入 SHRLDSYNCI 上捕获到上升沿时生效

位域	名称	描述
25-24	FAULTMODE	故障保护模式 00：发生故障时，PWM 输出强制为低电平 01：发生故障时，PWM 输出强制为高电平 1x：发生故障时，PWM 输出强制为高阻
23-22	FAULTRECTIME	故障恢复输出时机选择： 00：故障条件消除后立即恢复输出 01：故障条件消除后，在计数器重载 RLD 后恢复输出 10：定时器的某一个 CMP 发生匹配后生效，用户可以通过 FAULTRECHWSEL 从比较器 0□23 中选择一个， 匹配可以是该比较器的输出比较，也可以是输入捕获。 11：故障条件消除后，软件将 FAULTCLR 位置 1 后，恢复输出
21	FRCSRCSEL	强制输出选择位 0：硬件强制输出，由 FRCI 控制强制输出开关 1：软件强制输出，SWFRC 位控制强制输出开关
20	PAIR	PWM 互补输出位 0：PWM 输出位独立模式 1：PWM 输出位互补模式
19-0	DEADAREA	死区长度位，设置死区长度，单位为 0.5 时钟周期。死区长度最小为 1 个时钟周期。 用户配置此位域前，必须将 PAIR 位置 1

PWMCFG 位域

33.4.15 SR (0x220)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD				FAULTF	XRLDF	HALFRLDF	RLDF	CMPFX																							
N/A				W1C	W1C	W1C	W1C	W1C																							
x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

SR [31:0]

位域	名称	描述
27	FAULTF	故障标志位
26	XRLDF	扩展重载标志位
25	HALFRLDF	半重载标志位
24	RLDF	重载标志位
23-0	CMPFX	比较器匹配事件标志位

SR 位域

33.4.16 IRQEN (0x224)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD				FAULTIRQE	XRLDIRQE	HALFRLDIRQE	RLDIRQE	CMPIRQEX																							
N/A				RW	RW	RW	RW	RW																							
x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

IRQEN [31:0]

位域	名称	描述
27	FAULTIRQE	故障中断请求使能位
26	XRLDIRQE	扩展重载中断请求使能位
25	HALFRLDIRQE	半重载中断请求使能位
24	RLDIRQE	重载中断请求使能位
23-0	CMPIRQEX	比较器匹配事件中断请求使能位

IRQEN 位域

33.4.17 DMAEN (0x22C)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD				FAULTEN	XRLDEN	HALFRLDEN	RLDEN	CMPENX																							
N/A				RW	RW	RW	RW	RW																							
x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

DMAEN [31:0]

位域	名称	描述
27	FAULTEN	故障 DMA 请求使能位
26	XRLDEN	扩展重载 DMA 请求使能位
25	HALFRLDEN	半重载 DMA 请求使能位
24	RLDEN	重载 DMA 请求使能位
23-0	CMPENX	比较器匹配事件 DMA 请求使能位

DMAEN 位域

33.4.18 CMPCFG (0x230 + 0x4 * n)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																								XCNTCMPEN				CMPSHDWUPT		CMPMODE	RSVD
N/A																								RW				RW		RW	N/A
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0

CMPCFG [31:0]

位域	名称	描述
7-4	XCNTCMPEN	比较器扩展比较值使能位
3-2	CMPSHDWUPT	比较器影子寄存器生效时机选择： 00：由软件把 SHLK [SHLK] 位置 1 后生效 01：实时生效，在寄存器写之后，一个周期内生效 10：定时器的某一个 CMP 发生匹配后生效，用户可以通过 CMPSHDWSEL 从比较器 0□23 中选择一个， 匹配可以是该比较器的输出比较，也可以是输入捕获。 11：影子寄存器重载触发输入 SHRLDSYNCI 上捕获到上升沿时生效
1	CMPMODE	比较器模式选择 0：输出比较模式 1：输入捕获模式

CMPCFG 位域

34 互联管理器 TRGM

在本产品上，多个外设的输入输出信号可以相互连接，使得多个外设可以相互配合使用。互联管理器可以配置这些连接。

互联管理器支持若干输入输出（IO），可以把这些 IO 分配给连接到互联管理器的外设，用作它们的输入或者输出。

本产品上，互联管理器的输入信号分配，请查阅小节 32.8.1。

本产品上，互联管理器的输出信号分配，请查阅小节 32.8.2。

互联管理器输入配有数字滤波器，可以对一些信号进行滤波。

本产品上，互联管理器的数字滤波器分配，请查阅小节 32.8.4。

互联管理器也管理外设的 DMA 请求，把选中的 DMA 请求发送到 DMA 控制器。

本产品上，互联管理器的 DMA 请求信号分配，请查阅小节 32.8.3。

34.1 特性总结

本章节介绍互联管理器 TRGM（Trigger Manager）的主要特性：

- 多路复选器（MUX）阵列
- 支持多个输入和多个输出
- 每个输出都可以单独配置，从众多输入中选择
- 输入信号数字滤波器
- 输出信号极性取反
- 信号边沿到脉冲转换
- DMA 请求管理，管理 PWMT，QDEC 和 HALL 的 DMA 请求

34.2 功能描述

本章节描述互联管理器 TRGM 的功能。

34.2.1 管理器信号输入输出复选器

用户通过配置互联管理器寄存器,可以把 TRGM 的任意输入信号连接到互相管理器的输出,配置 TRGOCFGx[OCFG] 位来从 TRGM 的输入中选择一个，作为 OUTx 的输出。

34.2.2 输出配置

对于互联管理器的输出，在选择了信号输入后，用户还可以通过配置 TRGOCFGx[OPOL] 位，对输出取反。TRGOCFGx[OPOL] 位置 0 时，输出信号不变，置 1 时，输出信号取反。

此外，互联管理器的输出配置支持对信号进行边沿到脉冲的转换。TRGOCFGx[REDG2PEN] 位置 1 时，输入信号的每一个上升沿，都会输出宽度为一个互联管理器时钟周期长度的脉冲。

TRGOCFGx[FEDG2PEN] 位置 1 时，输入信号的每一个下降沿，都会输出宽度为一个互联管理器时钟周期长度的脉冲。

34.2.3 数字滤波器

互联管理器 TRGM 为部分输入信号提供数字滤波器，用户可以通过 FILTCFGx[MODE] 寄存器滤波器的工作模式：

- 3'b000，旁路模式，数字滤波器关闭
- 3'b100，滤刺模式，滤波器输入翻转后，输出也会立即翻转，之后会在一定时间内无视滤波器的输入。这个模式下，滤波器输出会紧随输入，同时会避免输出信号出现毛刺。
- 3'b101，延时滤波器，滤波器输入翻转后需要保持一定时间，滤波器输出才会翻转
- 3'b110，滤峰模式，滤波器输入置逻辑 1 后，需要保持一定时间，滤波器输出才会置逻辑 1，而滤波器输入置 0，滤波器输出会立即置 0，这个模式的目的是滤除不够长的输入波峰。
- 3'b111，滤谷模式，滤波器输入置逻辑 0 后，需要保持一定时间，滤波器输出才会置逻辑 0，而滤波器输入置 1，滤波器输出会立即置 1，这个模式的目的是滤除不够长的输入波谷。

用户可以通过 FILTCFGx[FILTLEN] 寄存器位配置滤波器的滤波宽度。单位是互联管理器的时钟周期。

用户可以通过把 FILTCFGx[OUTINV] 位置 1 来将滤波器的输出取反。

34.2.4 DMA 请求管理

互联管理器支持管理与它互联的部分外设的 DMA 请求，这些外设的 DMA 请求通过互联管理器转发到 DMA-MUX。

互联管理器支持 4 个 DMA 请求，用户可以配置 DMACFG0~ DMACFG3 寄存器，从下表中选择输出到 DMA 控制器的 DMA 请求。

34.3 TRGM 寄存器列表

TRGM 的寄存器列表如下：

TRGM0 base address: 0xF033C000

地址偏移	名称	描述	复位值
0x0000	FILTCFG[PWM0_IN0]	滤波器配置寄存器	0x00000000
0x0004	FILTCFG[PWM0_IN1]	滤波器配置寄存器	0x00000000
0x0008	FILTCFG[PWM0_IN2]	滤波器配置寄存器	0x00000000
0x000C	FILTCFG[PWM0_IN3]	滤波器配置寄存器	0x00000000
0x0010	FILTCFG[PWM0_IN4]	滤波器配置寄存器	0x00000000
0x0014	FILTCFG[PWM0_IN5]	滤波器配置寄存器	0x00000000
0x0018	FILTCFG[PWM0_IN6]	滤波器配置寄存器	0x00000000
0x001C	FILTCFG[PWM0_IN7]	滤波器配置寄存器	0x00000000
0x0020	FILTCFG[PWM1_IN0]	滤波器配置寄存器	0x00000000
0x0024	FILTCFG[PWM1_IN1]	滤波器配置寄存器	0x00000000
0x0028	FILTCFG[PWM1_IN2]	滤波器配置寄存器	0x00000000
0x002C	FILTCFG[PWM1_IN3]	滤波器配置寄存器	0x00000000
0x0030	FILTCFG[PWM1_IN4]	滤波器配置寄存器	0x00000000
0x0034	FILTCFG[PWM1_IN5]	滤波器配置寄存器	0x00000000
0x0038	FILTCFG[PWM1_IN6]	滤波器配置寄存器	0x00000000
0x003C	FILTCFG[PWM1_IN7]	滤波器配置寄存器	0x00000000

地址偏移	名称	描述	复位值
0x0040	FILTCFG[MOTO_GPIO_IN0]	滤波器配置寄存器	0x00000000
0x0044	FILTCFG[MOTO_GPIO_IN1]	滤波器配置寄存器	0x00000000
0x0048	FILTCFG[MOTO_GPIO_IN2]	滤波器配置寄存器	0x00000000
0x004C	FILTCFG[MOTO_GPIO_IN3]	滤波器配置寄存器	0x00000000
0x0050	FILTCFG[MOTO_GPIO_IN4]	滤波器配置寄存器	0x00000000
0x0054	FILTCFG[MOTO_GPIO_IN5]	滤波器配置寄存器	0x00000000
0x0058	FILTCFG[MOTO_GPIO_IN6]	滤波器配置寄存器	0x00000000
0x005C	FILTCFG[MOTO_GPIO_IN7]	滤波器配置寄存器	0x00000000
0x0060	FILTCFG[PWM0_FAULT0]	滤波器配置寄存器	0x00000000
0x0064	FILTCFG[PWM0_FAULT1]	滤波器配置寄存器	0x00000000
0x0068	FILTCFG[PWM1_FAULT0]	滤波器配置寄存器	0x00000000
0x006C	FILTCFG[PWM1_FAULT1]	滤波器配置寄存器	0x00000000
0x0100	TRGOCFG[MOT2OPAMP0_0]	TRGM 输出配置寄存器	0x00000000
0x0104	TRGOCFG[MOT2OPAMP0_1]	TRGM 输出配置寄存器	0x00000000
0x0108	TRGOCFG[MOT2OPAMP0_2]	TRGM 输出配置寄存器	0x00000000
0x010C	TRGOCFG[MOT2OPAMP0_3]	TRGM 输出配置寄存器	0x00000000
0x0110	TRGOCFG[MOT2OPAMP0_4]	TRGM 输出配置寄存器	0x00000000
0x0114	TRGOCFG[MOT2OPAMP0_5]	TRGM 输出配置寄存器	0x00000000
0x0118	TRGOCFG[MOT2OPAMP0_6]	TRGM 输出配置寄存器	0x00000000
0x011C	TRGOCFG[MOT2OPAMP0_7]	TRGM 输出配置寄存器	0x00000000
0x0120	TRGOCFG[MOT2OPAMP1_0]	TRGM 输出配置寄存器	0x00000000
0x0124	TRGOCFG[MOT2OPAMP1_1]	TRGM 输出配置寄存器	0x00000000
0x0128	TRGOCFG[MOT2OPAMP1_2]	TRGM 输出配置寄存器	0x00000000
0x012C	TRGOCFG[MOT2OPAMP1_3]	TRGM 输出配置寄存器	0x00000000
0x0130	TRGOCFG[MOT2OPAMP1_4]	TRGM 输出配置寄存器	0x00000000
0x0134	TRGOCFG[MOT2OPAMP1_5]	TRGM 输出配置寄存器	0x00000000
0x0138	TRGOCFG[MOT2OPAMP1_6]	TRGM 输出配置寄存器	0x00000000
0x013C	TRGOCFG[MOT2OPAMP1_7]	TRGM 输出配置寄存器	0x00000000
0x0140	TRGOCFG[GPTMR0_IN2]	TRGM 输出配置寄存器	0x00000000
0x0144	TRGOCFG[GPTMR0_IN3]	TRGM 输出配置寄存器	0x00000000
0x0148	TRGOCFG[GPTMR0_SYNCI]	TRGM 输出配置寄存器	0x00000000
0x014C	TRGOCFG[GPTMR1_IN2]	TRGM 输出配置寄存器	0x00000000
0x0150	TRGOCFG[GPTMR1_IN3]	TRGM 输出配置寄存器	0x00000000
0x0154	TRGOCFG[GPTMR1_SYNCI]	TRGM 输出配置寄存器	0x00000000
0x0158	TRGOCFG[GPTMR2_IN2]	TRGM 输出配置寄存器	0x00000000
0x015C	TRGOCFG[GPTMR2_IN3]	TRGM 输出配置寄存器	0x00000000
0x0160	TRGOCFG[GPTMR2_SYNCI]	TRGM 输出配置寄存器	0x00000000
0x0164	TRGOCFG[GPTMR3_IN2]	TRGM 输出配置寄存器	0x00000000
0x0168	TRGOCFG[GPTMR3_IN3]	TRGM 输出配置寄存器	0x00000000
0x016C	TRGOCFG[GPTMR3_SYNCI]	TRGM 输出配置寄存器	0x00000000

地址偏移	名称	描述	复位值
0x0170	TRGOCFG[CMP0_WIN]	TRGM 输出配置寄存器	0x00000000
0x0174	TRGOCFG[CMP1_WIN]	TRGM 输出配置寄存器	0x00000000
0x0178	TRGOCFG[DAC0_BUFTRG]	TRGM 输出配置寄存器	0x00000000
0x017C	TRGOCFG[DAC1_BUFTRG]	TRGM 输出配置寄存器	0x00000000
0x0180	TRGOCFG[ADC0_STRGI]	TRGM 输出配置寄存器	0x00000000
0x0184	TRGOCFG[ADC1_STRGI]	TRGM 输出配置寄存器	0x00000000
0x0188	TRGOCFG[ADCX_PTRGI0A]	TRGM 输出配置寄存器	0x00000000
0x018C	TRGOCFG[ADCX_PTRGI0B]	TRGM 输出配置寄存器	0x00000000
0x0190	TRGOCFG[ADCX_PTRGI0C]	TRGM 输出配置寄存器	0x00000000
0x0194	TRGOCFG[ADCX_PTRGI1A]	TRGM 输出配置寄存器	0x00000000
0x0198	TRGOCFG[ADCX_PTRGI1B]	TRGM 输出配置寄存器	0x00000000
0x019C	TRGOCFG[ADCX_PTRGI1C]	TRGM 输出配置寄存器	0x00000000
0x01A0	TRGOCFG[ADCX_PTRGI2A]	TRGM 输出配置寄存器	0x00000000
0x01A4	TRGOCFG[ADCX_PTRGI2B]	TRGM 输出配置寄存器	0x00000000
0x01A8	TRGOCFG[ADCX_PTRGI2C]	TRGM 输出配置寄存器	0x00000000
0x01AC	TRGOCFG[ADCX_PTRGI3A]	TRGM 输出配置寄存器	0x00000000
0x01B0	TRGOCFG[ADCX_PTRGI3B]	TRGM 输出配置寄存器	0x00000000
0x01B4	TRGOCFG[ADCX_PTRGI3C]	TRGM 输出配置寄存器	0x00000000
0x01B8	TRGOCFG[CAN_PTPC0_CAP]	TRGM 输出配置寄存器	0x00000000
0x01BC	TRGOCFG[CAN_PTPC1_CAP]	TRGM 输出配置寄存器	0x00000000
0x01C0	TRGOCFG[QEO0_TRIG_IN0]	TRGM 输出配置寄存器	0x00000000
0x01C4	TRGOCFG[QEO0_TRIG_IN1]	TRGM 输出配置寄存器	0x00000000
0x01C8	TRGOCFG[QEO1_TRIG_IN0]	TRGM 输出配置寄存器	0x00000000
0x01CC	TRGOCFG[QEO1_TRIG_IN1]	TRGM 输出配置寄存器	0x00000000
0x01D0	TRGOCFG[SEI_TRIG_IN0]	TRGM 输出配置寄存器	0x00000000
0x01D4	TRGOCFG[SEI_TRIG_IN1]	TRGM 输出配置寄存器	0x00000000
0x01D8	TRGOCFG[SEI_TRIG_IN2]	TRGM 输出配置寄存器	0x00000000
0x01DC	TRGOCFG[SEI_TRIG_IN3]	TRGM 输出配置寄存器	0x00000000
0x01E0	TRGOCFG[SEI_TRIG_IN4]	TRGM 输出配置寄存器	0x00000000
0x01E4	TRGOCFG[SEI_TRIG_IN5]	TRGM 输出配置寄存器	0x00000000
0x01E8	TRGOCFG[SEI_TRIG_IN6]	TRGM 输出配置寄存器	0x00000000
0x01EC	TRGOCFG[SEI_TRIG_IN7]	TRGM 输出配置寄存器	0x00000000
0x01F0	TRGOCFG[MMC0_TRIG_IN0]	TRGM 输出配置寄存器	0x00000000
0x01F4	TRGOCFG[MMC0_TRIG_IN1]	TRGM 输出配置寄存器	0x00000000
0x01F8	TRGOCFG[MMC1_TRIG_IN0]	TRGM 输出配置寄存器	0x00000000
0x01FC	TRGOCFG[MMC1_TRIG_IN1]	TRGM 输出配置寄存器	0x00000000
0x0200	TRGOCFG[PLB_IN_00]	TRGM 输出配置寄存器	0x00000000
0x0204	TRGOCFG[PLB_IN_01]	TRGM 输出配置寄存器	0x00000000
0x0208	TRGOCFG[PLB_IN_02]	TRGM 输出配置寄存器	0x00000000
0x020C	TRGOCFG[PLB_IN_03]	TRGM 输出配置寄存器	0x00000000

地址偏移	名称	描述	复位值
0x0210	TRGOCFG[PLB_IN_04]	TRGM 输出配置寄存器	0x00000000
0x0214	TRGOCFG[PLB_IN_05]	TRGM 输出配置寄存器	0x00000000
0x0218	TRGOCFG[PLB_IN_06]	TRGM 输出配置寄存器	0x00000000
0x021C	TRGOCFG[PLB_IN_07]	TRGM 输出配置寄存器	0x00000000
0x0220	TRGOCFG[PLB_IN_08]	TRGM 输出配置寄存器	0x00000000
0x0224	TRGOCFG[PLB_IN_09]	TRGM 输出配置寄存器	0x00000000
0x0228	TRGOCFG[PLB_IN_10]	TRGM 输出配置寄存器	0x00000000
0x022C	TRGOCFG[PLB_IN_11]	TRGM 输出配置寄存器	0x00000000
0x0230	TRGOCFG[PLB_IN_12]	TRGM 输出配置寄存器	0x00000000
0x0234	TRGOCFG[PLB_IN_13]	TRGM 输出配置寄存器	0x00000000
0x0238	TRGOCFG[PLB_IN_14]	TRGM 输出配置寄存器	0x00000000
0x023C	TRGOCFG[PLB_IN_15]	TRGM 输出配置寄存器	0x00000000
0x0240	TRGOCFG[PLB_IN_16]	TRGM 输出配置寄存器	0x00000000
0x0244	TRGOCFG[PLB_IN_17]	TRGM 输出配置寄存器	0x00000000
0x0248	TRGOCFG[PLB_IN_18]	TRGM 输出配置寄存器	0x00000000
0x024C	TRGOCFG[PLB_IN_19]	TRGM 输出配置寄存器	0x00000000
0x0250	TRGOCFG[PLB_IN_20]	TRGM 输出配置寄存器	0x00000000
0x0254	TRGOCFG[PLB_IN_21]	TRGM 输出配置寄存器	0x00000000
0x0258	TRGOCFG[PLB_IN_22]	TRGM 输出配置寄存器	0x00000000
0x025C	TRGOCFG[PLB_IN_23]	TRGM 输出配置寄存器	0x00000000
0x0260	TRGOCFG[PLB_IN_24]	TRGM 输出配置寄存器	0x00000000
0x0264	TRGOCFG[PLB_IN_25]	TRGM 输出配置寄存器	0x00000000
0x0268	TRGOCFG[PLB_IN_26]	TRGM 输出配置寄存器	0x00000000
0x026C	TRGOCFG[PLB_IN_27]	TRGM 输出配置寄存器	0x00000000
0x0270	TRGOCFG[PLB_IN_28]	TRGM 输出配置寄存器	0x00000000
0x0274	TRGOCFG[PLB_IN_29]	TRGM 输出配置寄存器	0x00000000
0x0278	TRGOCFG[PLB_IN_30]	TRGM 输出配置寄存器	0x00000000
0x027C	TRGOCFG[PLB_IN_31]	TRGM 输出配置寄存器	0x00000000
0x0280	TRGOCFG[MOT_GPIO0]	TRGM 输出配置寄存器	0x00000000
0x0284	TRGOCFG[MOT_GPIO1]	TRGM 输出配置寄存器	0x00000000
0x0288	TRGOCFG[MOT_GPIO2]	TRGM 输出配置寄存器	0x00000000
0x028C	TRGOCFG[MOT_GPIO3]	TRGM 输出配置寄存器	0x00000000
0x0290	TRGOCFG[MOT_GPIO4]	TRGM 输出配置寄存器	0x00000000
0x0294	TRGOCFG[MOT_GPIO5]	TRGM 输出配置寄存器	0x00000000
0x0298	TRGOCFG[MOT_GPIO6]	TRGM 输出配置寄存器	0x00000000
0x029C	TRGOCFG[MOT_GPIO7]	TRGM 输出配置寄存器	0x00000000
0x02A0	TRGOCFG[PWM_IN8]	TRGM 输出配置寄存器	0x00000000
0x02A4	TRGOCFG[PWM_IN9]	TRGM 输出配置寄存器	0x00000000
0x02A8	TRGOCFG[PWM_IN10]	TRGM 输出配置寄存器	0x00000000
0x02AC	TRGOCFG[PWM_IN11]	TRGM 输出配置寄存器	0x00000000

地址偏移	名称	描述	复位值
0x02B0	TRGOCFG[PWM_IN12]	TRGM 输出配置寄存器	0x00000000
0x02B4	TRGOCFG[PWM_IN13]	TRGM 输出配置寄存器	0x00000000
0x02B8	TRGOCFG[PWM_IN14]	TRGM 输出配置寄存器	0x00000000
0x02BC	TRGOCFG[PWM_IN15]	TRGM 输出配置寄存器	0x00000000
0x02C0	TRGOCFG[PWM0_FRCI]	TRGM 输出配置寄存器	0x00000000
0x02C4	TRGOCFG[PWM0_FRCSYNCI]	TRGM 输出配置寄存器	0x00000000
0x02C8	TRGOCFG[PWM0_SYNCI]	TRGM 输出配置寄存器	0x00000000
0x02CC	TRGOCFG[PWM0_SHRLDSYNCI]	TRGM 输出配置寄存器	0x00000000
0x02D0	TRGOCFG[PWM0_FAULTI0]	TRGM 输出配置寄存器	0x00000000
0x02D4	TRGOCFG[PWM0_FAULTI1]	TRGM 输出配置寄存器	0x00000000
0x02D8	TRGOCFG[PWM1_FRCI]	TRGM 输出配置寄存器	0x00000000
0x02DC	TRGOCFG[PWM1_FRCSYNCI]	TRGM 输出配置寄存器	0x00000000
0x02E0	TRGOCFG[PWM1_SYNCI]	TRGM 输出配置寄存器	0x00000000
0x02E4	TRGOCFG[PWM1_SHRLDSYNCI]	TRGM 输出配置寄存器	0x00000000
0x02E8	TRGOCFG[PWM1_FAULTI0]	TRGM 输出配置寄存器	0x00000000
0x02EC	TRGOCFG[PWM1_FAULTI1]	TRGM 输出配置寄存器	0x00000000
0x02F0	TRGOCFG[RDC_TRIG_IN0]	TRGM 输出配置寄存器	0x00000000
0x02F4	TRGOCFG[RDC_TRIG_IN1]	TRGM 输出配置寄存器	0x00000000
0x02F8	TRGOCFG[SYNCTIMER_TRIG]	TRGM 输出配置寄存器	0x00000000
0x02FC	TRGOCFG[QEI0_TRIG_IN]	TRGM 输出配置寄存器	0x00000000
0x0300	TRGOCFG[QEI1_TRIG_IN]	TRGM 输出配置寄存器	0x00000000
0x0304	TRGOCFG[QEI0_PAUSE]	TRGM 输出配置寄存器	0x00000000
0x0308	TRGOCFG[QEI1_PAUSE]	TRGM 输出配置寄存器	0x00000000
0x030C	TRGOCFG[UART_TRIG0]	TRGM 输出配置寄存器	0x00000000
0x0310	TRGOCFG[UART_TRIG1]	TRGM 输出配置寄存器	0x00000000
0x0314	TRGOCFG[TRGM_IRQ0]	TRGM 输出配置寄存器	0x00000000
0x0318	TRGOCFG[TRGM_IRQ1]	TRGM 输出配置寄存器	0x00000000
0x031C	TRGOCFG[TRGM_DMA0]	TRGM 输出配置寄存器	0x00000000
0x0320	TRGOCFG[TRGM_DMA1]	TRGM 输出配置寄存器	0x00000000
0x0400	DMACFG[DMACFG0]	DMA 请求配置寄存器	0x00000000
0x0404	DMACFG[DMACFG1]	DMA 请求配置寄存器	0x00000000
0x0408	DMACFG[DMACFG2]	DMA 请求配置寄存器	0x00000000
0x040C	DMACFG[DMACFG3]	DMA 请求配置寄存器	0x00000000
0x0410	DMACFG[DMACFG4]	DMA 请求配置寄存器	0x00000000
0x0414	DMACFG[DMACFG5]	DMA 请求配置寄存器	0x00000000
0x0418	DMACFG[DMACFG6]	DMA 请求配置寄存器	0x00000000
0x041C	DMACFG[DMACFG7]	DMA 请求配置寄存器	0x00000000
0x0500	GCR	通用控制寄存器	0x00000000
0x0510	ADC_MATRIX_SEL	ADC 矩阵选择寄存器	0x00000000
0x0514	DAC_MATRIX_SEL	DAC 矩阵选择寄存器	0x00000000

地址偏移	名称	描述	复位值
0x0518	POS_MATRIX_SEL0	位置矩阵选择寄存器 0	0x00000000
0x0518	POS_MATRIX_SEL1	位置矩阵选择寄存器 1	0x00000000
0x0600	TRGM_IN[0]		0x00000000
0x0604	TRGM_IN[1]		0x00000000
0x0608	TRGM_IN[2]		0x00000000
0x060C	TRGM_IN[3]		0x00000000
0x0620	TRGM_OUT[0]		0x00000000
0x0624	TRGM_OUT[1]		0x00000000
0x0628	TRGM_OUT[2]		0x00000000
0x062C	TRGM_OUT[3]		0x00000000
0x0630	TRGM_OUT[4]		0x00000000

表 174: TRGM 寄存器列表

34.4 TRGM 寄存器描述

TRGM 的寄存器详细说明如下：

34.4.1 FILTCFG (0x0 + 0x4 * n)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD															OUTINV	MODE			SYNCEN	FILTLEN											
N/A															RW	RW			RW	RW											
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

FILTCFG [31:0]

位域	名称	描述
16	OUTINV	1：滤波器输出取反 0：滤波器输出不变
15-13	MODE	此位域定义了滤波器的模式 000：旁路模式 100：滤刺模式 101：延时滤波器 110：滤峰模式 111：滤谷模式
12	SYNCEN	1：将滤波器输入信号于 TRGM 时钟同步 0：不进行同步（有可能会导致时序问题）
11-0	FILTLEN	此位域设置滤波器的长度，单位是 TRGM 时钟周期

FILTCFG 位域

34.4.2 TRGOCFG (0x100 + 0x4 * n)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																				OUTINV	FEDG2PEN	REDG2PEN	RSVD		TRIGOSEL							
N/A																				RW	RW	RW	N/A		RW							
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	x	x	0	0	0	0	0	0	0

TRGOCFG [31:0]

位域	名称	描述
11	OUTINV	1：TRGM 输出取反 0：TRGM 输出不变
10	FEDG2PEN	1: 将输入信号的下降沿转化为输出一个脉冲
9	REDG2PEN	1: 将输入信号的上升沿转化为输出一个脉冲
6-0	TRIGOSEL	此位域从 TRGM 输入中选择一个作为 TRGM 输出

TRGOCFG 位域

34.4.3 DMACFG (0x400 + 0x4 * n)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMAMUX_EN	RSVD																									DMASRCSEL					
RW	N/A																									RW					
0	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0

DMACFG [31:0]

位域	名称	描述
31	DMAMUX_EN	DMA 使能
5-0	DMASRCSEL	此位域选择 DMA 请求之一，作为 TRGM 的 DMA 请求

DMACFG 位域

34.4.4 GCR (0x500)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																								TRGOPEN							
N/A																								RW							
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0

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

GCR [31:0]

位域	名称	描述
7-0	TRGOPEN	此位域使能 TRGM 输出到 IO

GCR 位域

34.4.5 ADC_MATRIX_SEL (0x510)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
QEI1_ADC1_SEL								QEI1_ADC0_SEL								QEI0_ADC1_SEL								QEI0_ADC0_SEL							
RW								RW								RW								RW							
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ADC_MATRIX_SEL [31:0]

位域	名称	描述
31-24	QEI1_ADC1_SEL	0-adc0; 1-adc1; 2-rdc_adc0; 3-rdc_adc1; bit7 is used to invert adc_value; others reserved
23-16	QEI1_ADC0_SEL	
15-8	QEI0_ADC1_SEL	
7-0	QEI0_ADC0_SEL	

ADC_MATRIX_SEL 位域

34.4.6 DAC_MATRIX_SEL (0x514)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DAC1_DAC_SEL								DAC0_DAC_SEL								ACMP1_DAC_SEL								ACMP0_DAC_SEL							
RW								RW								RW								RW							
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

DAC_MATRIX_SEL [31:0]

位域	名称	描述
31-24	DAC1_DAC_SEL	0-qeo0_dac0; 1-qeo0_dac1; 2-qeo0_dac2; 3-qeo1_dac0; 4-qeo1_dac1; 5-qeo1_dac2; 6-rdc_dac0; 7-rdc_dac1; bit7 is used to invert dac_value; others reserved
23-16	DAC0_DAC_SEL	
15-8	ACMP1_DAC_SEL	
7-0	ACMP0_DAC_SEL	

DAC_MATRIX_SEL 位域

34.4.7 POS_MATRIX_SEL0 (0x518)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MMC1_POSIN_SEL								MMC0_POSIN_SEL								SEI_POSIN1_SEL								SEI_POSIN0_SEL							
RW								RW								RW								RW							
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

POS_MATRIX_SEL0 [31:0]

位域	名称	描述
31-24	MMC1_POSIN_SEL	0-sei_pos_out0; 1-sei_pos_out1; 2-qei0_pos; 3-qei1_pos; 4-mmc0_pos_out0; 5-mmc0_pos_out1; 6-mmc1_pos_out0; 7-mmc1_pos_out1; bit7 is used to invert position value; others reserved
23-16	MMC0_POSIN_SEL	
15-8	SEI_POSIN1_SEL	
7-0	SEI_POSIN0_SEL	

POS_MATRIX_SEL0 位域

34.4.8 POS_MATRIX_SEL1 (0x518)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																QEO1_POS_SEL								QEO0_POS_SEL							
N/A																RW								RW							
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

POS_MATRIX_SEL1 [31:0]

位域	名称	描述
15-8	QEO1_POS_SEL	
7-0	QEO0_POS_SEL	

POS_MATRIX_SEL1 位域

34.4.9 TRGM_IN (0x600 + 0x4 * n)

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

TRGM_IN [31:0]

位域	名称	描述
31-0	TRGM_IN	mmc1_trig_out[1:0], mmc0_trig_out[1:0],sync_pulse[3:0],moto_gpio_in_sync[7:0],//31:16 gtmr3_to_motor_sync[1:0],gtmr2_to_motor_sync[1:0],gtmr1_to_motor_sync[1:0],g //15:8 acmp_out_sync[1:0],can2mot_event_sync[1:0],usb0_sof_tog_sync,pwm_debug,1 //7:0

TRGM_IN 位域

34.4.10 TRGM_OUT (0x620 + 0x4 * n)

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

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

TRGM_OUT [31:0]

位域	名称	描述

31-0	TRGM_OUT	motor_to_opamp0[7:0] = trig_mux_out[7:0];
		motor_to_opamp1[7:0] = trig_mux_out[15:8];
		motor_to_gtmr0_capt[1:0] = trig_mux_out[17:16];
		motor_to_gtmr0_sync = trig_mux_out[18];
		motor_to_gtmr1_capt[1:0] = trig_mux_out[20:19];
		motor_to_gtmr1_sync = trig_mux_out[21];
		motor_to_gtmr2_capt[1:0] = trig_mux_out[23:22];
		motor_to_gtmr2_sync = trig_mux_out[24];
		motor_to_gtmr3_capt[1:0] = trig_mux_out[26:25];
		motor_to_gtmr3_sync = trig_mux_out[27];
		acmp_window[1:0] = trig_mux_out[29:28];
		dac0_buf_trigger = trig_mux_out[30];
		dac1_buf_trigger = trig_mux_out[31];
		dac0_step_trigger[3:0] = {trig_mux_out[24:22],trig_mux_out[30]};//use same buf_trig, and gtmr2
		dac1_step_trigger[3:0] = {trig_mux_out[27:25],trig_mux_out[31]}; //use same buf_trig, and gtmr3

TRGM_OUT 位域

35 同步定时器 SYNT

本章节介绍同步定时器 SYNT 的主要功能和特性。

35.1 特性总结

本章节介绍同步定时器 SYNT 的主要特性：

- 32 位计数器和重载寄存器
- 支持 4 通道，每个通道支持一个比较寄存器
- 支持输出同步事件

35.2 功能描述

同步定时器有一个 32 位的计数器和 32 位的重载寄存器。

用户可以把 GCR[CEN] 位置 1 来使能计数器，使能以后，计数器总是从 0 开始计数，当计时器的值到达重载寄存器 RLD 值时，计数器恢复到 0 开始继续计数。

用户也可以通过 GCR[CRST] 位来把计数器复位到 0。

同步定时器支持 4 个通道，每个通道配置有 32 位比较寄存器 CMPx。

当计数器计数达到比较寄存器值时，会输出长度为一个同步定时器时钟脉冲的比较事件。这个事件可以通过互联管理器连接到电机系统的其他模块，比如 PWM,QEI,HAL 等，用作同步。

例如，通过连接到 PWM 的同步触发输入（SYNCI），可让 4 个电机系统精确到同一时钟周期开始运转，也可精确调节 4 个电机系统的相位差。

注意：同步定时器设计用于片上电机系统间的同步，工作时钟与电机系统同步，不能用于其他时钟域的外设。

同步定时器不支持生成中断或者 DMA 请求。

35.3 SYNT 寄存器列表

SYNT 的寄存器列表如下：

SYNT base address: 0xF0328000

地址偏移	名称	描述	复位值
0x0000	GCR	全局控制寄存器	0x00000000
0x0004	RLD	计数器重载寄存器	0x00000000
0x0008	TIMESTAMP_NEW	时间戳更新值	0x00000000
0x000C	CNT	计数器	0x00000000
0x0010	TIMESTAMP_SAV	时间戳触发保存值	0x00000000
0x0014	TIMESTAMP_CUR	时间戳当前值	0x00000000
0x0020	CMP[0]	比较器	0x00000000
0x0024	CMP[1]	比较器	0x00000000
0x0028	CMP[2]	比较器	0x00000000
0x002C	CMP[3]	比较器	0x00000000

表 175: SYNT 寄存器列表

35.4 SYNT 寄存器详细信息

SYNT 的寄存器详细说明如下：

35.4.1 GCR (0x0)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TIMESTAMP_INC_NEW	TIMESTAMP_DEC_NEW	TIMESTAMP_SET_NEW	TIMESTAMP_RESET	RSVD																						TIMESTAMP_DEBUG_EN	TIMESTAMP_ENABLE	RSVD	COUNTER_DEBUG_EN	CRST	CEN
WO	WO	WO	WO	N/A																						RW	RW	N/A	RW	RW	RW
0	0	0	0	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	x	0	0	0

GCR [31:0]

位域	名称	描述
31	TIMESTAMP_INC_NEW	设 1 在时间戳当前值上加上更新值
30	TIMESTAMP_DEC_NEW	设 1 在时间戳当前值上减去更新值
29	TIMESTAMP_SET_NEW	设 1 将时间戳值改为更新值
28	TIMESTAMP_RESET	复位时间戳
5	TIMESTAMP_DEBUG_EN	使能调试模式下停止时间戳
4	TIMESTAMP_ENABLE	时间戳使能
2	COUNTER_DEBUG_EN	使能调试模式下停止计数器
1	CRST	1: 复位计数器 软件需要清零此位以退出复位状态
0	CEN	1: 使能计数器

GCR 位域

35.4.2 RLD (0x4)

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

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

RLD [31:0]

位域	名称	描述
31-0	RLD	计数器的重载值

RLD 位域

35.4.3 TIMESTAMP_NEW (0x8)

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

TIMESTAMP_NEW [31:0]

位域	名称	描述
31-0	VALUE	时间戳更新值

TIMESTAMP_NEW 位域

35.4.4 CNT (0xC)

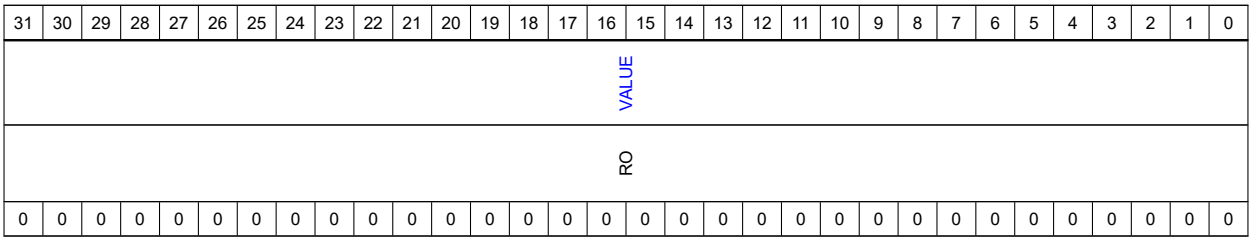
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CNT																															
RO																															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

CNT [31:0]

位域	名称	描述
31-0	CNT	计数器

CNT 位域

35.4.5 TIMESTAMP_SAV (0x10)

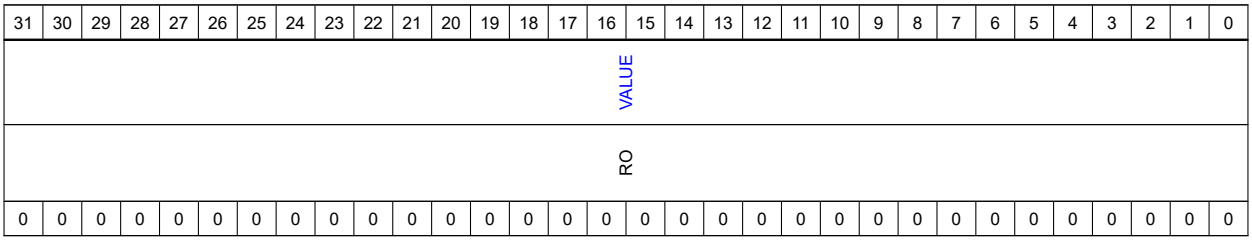


TIMESTAMP_SAV [31:0]

位域	名称	描述
31-0	VALUE	检测到外部触发信号时，将时间戳值存到该寄存器中

TIMESTAMP_SAV 位域

35.4.6 TIMESTAMP_CUR (0x14)

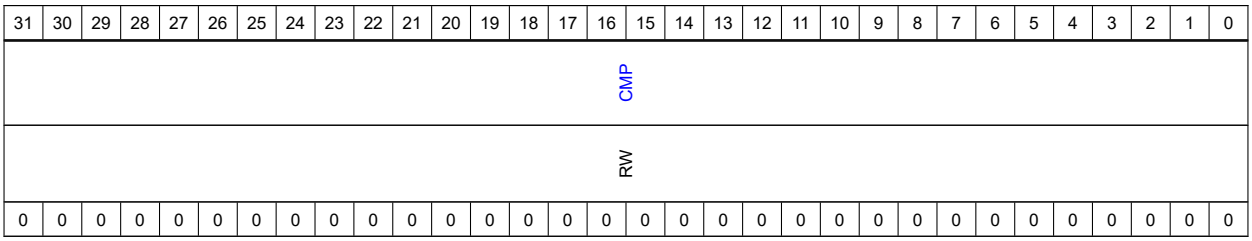


TIMESTAMP_CUR [31:0]

位域	名称	描述
31-0	VALUE	时间戳当前值

TIMESTAMP_CUR 位域

35.4.7 CMP (0x20 + 0x4 * n)



CMP [31:0]

位域	名称	描述
31-0	CMP	比较器值，当计数器计数到达比较器值时，输出长度为一个时钟脉冲的比较事件

CMP 位域

36 正交编码器接口 QEIV2

根据输入信号，测速和产生位置信息。

36.1 特性总结

本章节介绍正交编码器接口 QEIV2 的主要特性：

- 支持单相、两相 (正交/上下/方向脉冲)、三相（120° 相差）方波输入;
- 支持索引信号 INDEX/Z，可产生中断，重置线数计数器，改变圈数计数器
- 支持两个归位信号 HOME，仅产生中断
- 支持单相、两相 (正交) 正弦波输入（通过 ADC 线束）
- 支持正弦波支持幅度/中心值和夹角修正
- 支持线数支持到 32 位
- 支持两个预设的位置触发中断，可配置指定的圈数，线数，方向，位置
- 支持软件触发锁存，事件触发锁存，支持两个外部触发锁存（包括圈数，线数，方向，角度，位置）
- 支持两个基于脉冲数的测速计数，测速结果保存两个，记录测速时间
- 支持两个时长的测速计数，测速结果保存两个，记录测速时间
- 支持将位置转换成二进制位置，通过位置线束输出到位置管理器（包含位置获取时间）
- 允许软件修改变位置，线数计数器，圈数计数器（仅支持绝对值方式），允许增量和绝对值方式
- 支持看门狗，支持堵转检测
- 所有中断信号，也可配置产生 dma 请求，触发锁存，或触发输出信号

36.2 架构图

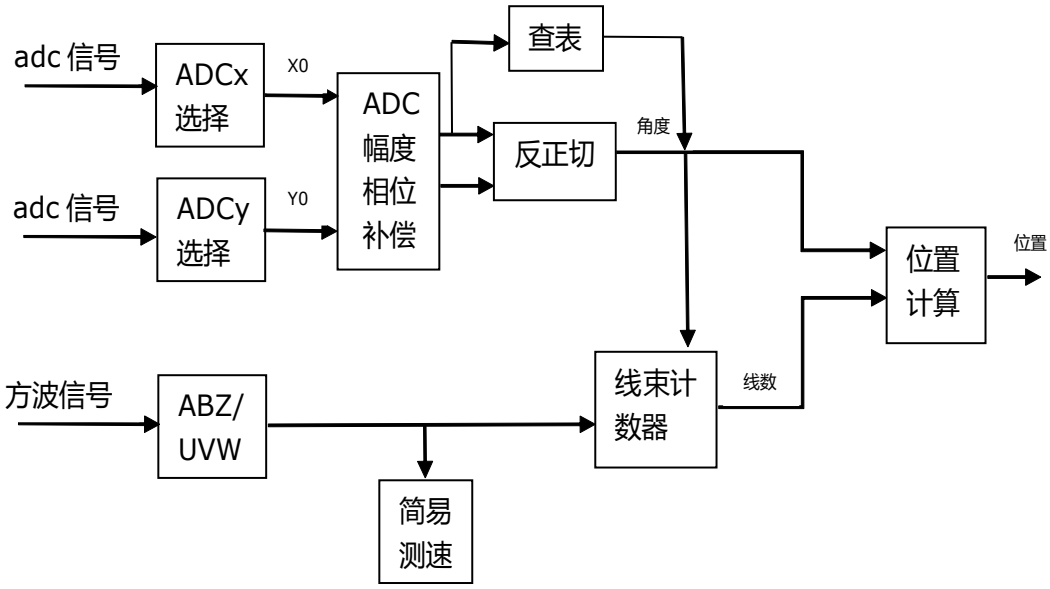


图 37: QEI 架构图

36.3 管脚说明

表 176: QEI 管脚说明

管脚	方向	功能描述
QEIn[1] A	输入	A 相信号，也可作为 UVW 的 U 相信号。
QEIn_B	输入	B 相信号，也可作为 UVW 的 V 相信号。
QEIn_Z	输入	Z 相信号，也可作为 UVW 的 W 相信号。
QEIn_F	输入	Fault 信号，有效时可以切断位置信息输出, 可产生中断。
QEIn_H0	输入	Home0 信号，可产生中断。
QEIn_H1	输入	Home1 信号，可产生中断。

[1] n=0,1

36.4 功能描述

36.4.1 位置信息输出

支持双路弦波输入，单路弦波输入，或方波输入。

弦波输入时，信号来自于芯片自带的 adc。

双路弦波输入，经过幅度调整以及相位调整后，输出两路正交信号，由反正切电路计算得出当前角度, 0 对应于 0 度，0x100000000 对应于 360 度。

单路弦波输入，经幅度调整后查表得出当前角度。

方波输入时，由软件配置每个输入信号电平或边沿对应的角度（比如 abz 信号，每个 a/b 信号沿对应 90 度，uvw 信号，每个信号沿对应 60 度）

硬件维护一个线数计数器，每当计算出的角度跨越 360 度时加一（实际算法是，前一个角度在第四象限，下一个角度在第一象限时加一），达到最大值后清零。根据当前的线数计数器，角度值，以及最大线数，可以计算得出当前位置值：

$$\text{position} = \frac{\text{phase\_cnt} * 2^{32} + \text{angle}}{\text{phase\_total}} \quad (1)$$

线数 (phase_cnt) 相当于整数部分，角度 (angle) 相当于小数部分, phase_total 为最大线数。

例如，最大线数 128，则软件需要配置 phase_param=0x02000000，如果当前线数计数器为 16，当前角度为 90 度（0x40000000）时，当前位置为 16*0x02000000 + 0x40000000/128 = 0x20800000

36.4.2 弦波配置指南

36.4.2.1 双路弦波

对于双路弦波输入，软件需要配置 adcx_cfg0(地址 0x200) 和 adcy_cfg0(地址 0x210), 指定 x 路（也就是 cos 信号）和 y 路（sin 信号）分别来自于哪个 adc 的哪个通道 (两路可以来自于同一个 adc，但是这样两路信号会有时间差，计算出的角度会不准，固定速度时可以由后续的相位偏差补偿，变化速度时就无法补偿了)。

adc 输出的信号是无符号的电压值，需要软件设置 0 电压点 (adcx_cfg2 和 adcy_cfg2，地址 0x208 和 0x218)，将无符号值转化为有符号值，同时可以微调电压幅度及方向（正负 0.5 到正负 2 之间，大幅度的调整需要外部电路完成）。

有时候输入的双路弦波信号不是完全正交的,可能由一定的相位偏差,软件可以配置 adc 参数寄存器 (adcx_cfg1

和 adcy_cfg1，地址 0x204 和 0x214), 进行相位调整。

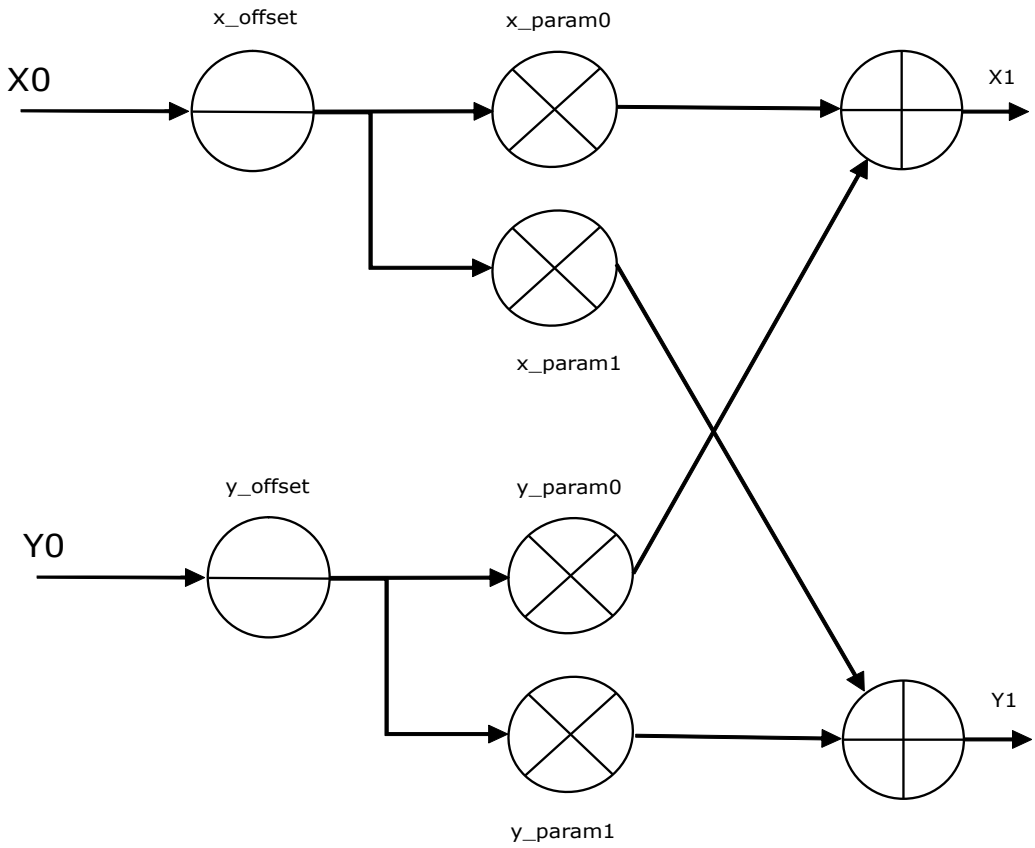


图 38: ADC 幅度相位补偿

如果正弦波信号相位偏移为 B，即输入为 Y0=sin(A+B), X0=cosA，需要配置参数得到 sinA 和 cosA：

sinA = sin((A+B)-B)

= sin(A+B)*cosB-cos(A+B)*sinB

= sin(A+B)*cosB-(cosA*cosB*sinB-sinA*sinB*sinB)

sinA*(1-(sinB)^2) $= sin(A+B)*cosB-cosA*cosB*sinB$

sinA*(cosB)^2 $= sin(A+B)*cosB-cosA*cosB*sinB$

sinA = sin(A+B)/cosB + cosA*(-tanB)

因此，对于 adcx_cfg1 和 adcy_cfg1 的配置如下：

x_param0 = 1(0x4000)

x_param1 = -tanB

y_param0 = 0

y_param1 = 1/cosB

这样可以在 X1 保持不变为 cosA，Y1 可以得到 sinA

36.4.2.2 单路弦波

对于单路正弦波输入，由 X1 信号查表得出角度，后续和双弦波一样，计算得出位置信息

36.4.3 简易测速

QEI 模块对于方波信号，提供两种测速方法，一种基于脉冲数量，一种基于时长（时钟数），每种各有两个计数器可单独基于不同的参数测量。

基于脉冲数量

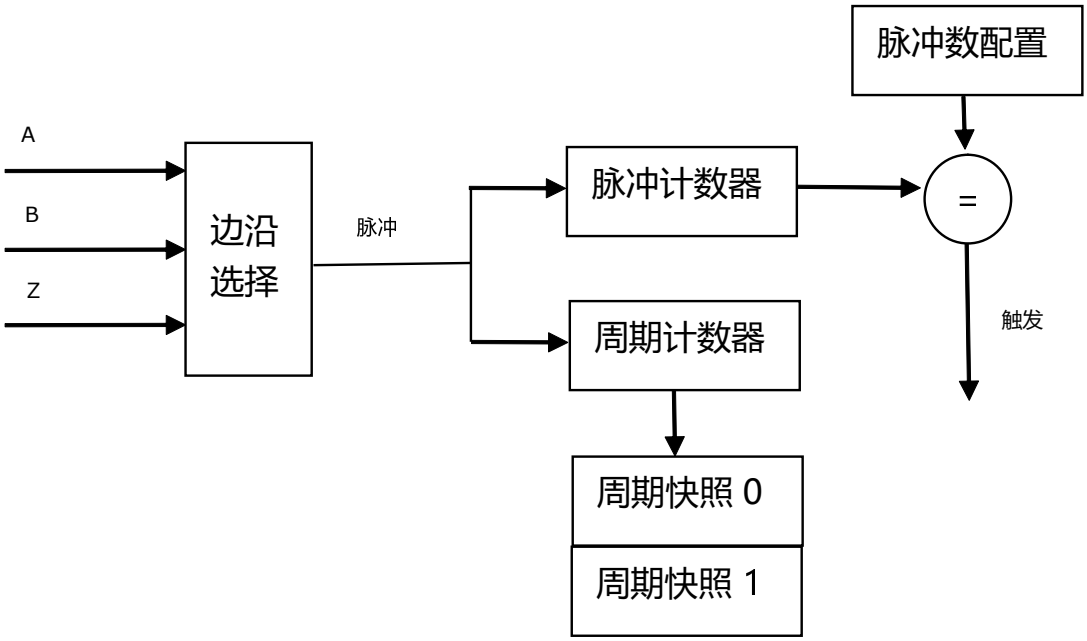


图 39: 脉冲测速

从输入的方波信号选择所需的边沿产生脉冲（可从 A,B,Z 信号中，选择一个或多个的上升沿或下降沿，或上升下降沿，不支持类似于 A 上升沿 B 下降沿）。

可配置两个不同的脉冲数（图中只画出了一组电路），当检测到输入信号的上升沿或下降沿时，脉冲计数器开始工作，每个边沿加一，同时周期计数也会开始工作，每个时钟周期加一。

当脉冲计数器等于配置的脉冲数时，产生触发信号，将当前的周期计数器值存入周期快照 0，同时将之前的周期快照 0 存入周期快照 1，周期计数器和脉冲计数器会清零重新开始工作。

软件可以在两个快照寄存器中读到最近两次的测速结果。

基于时长

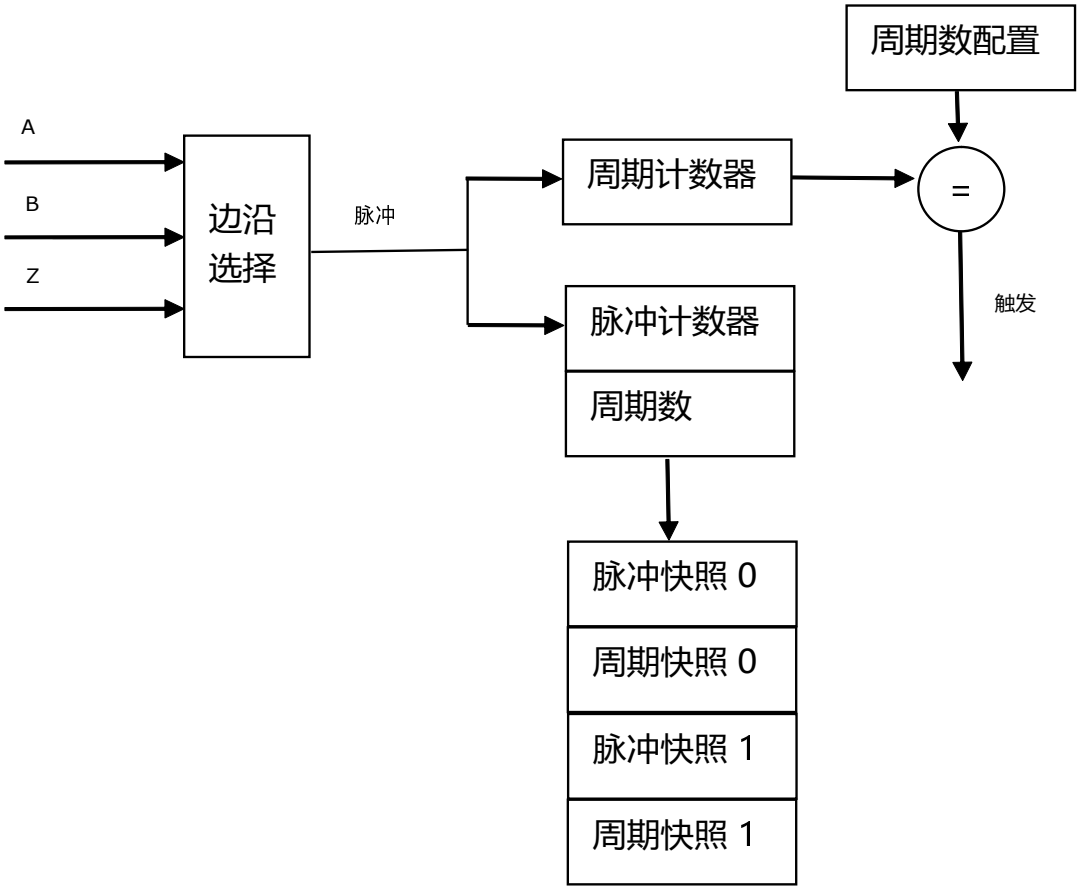


图 40: 时长测速

同样检测到脉冲信号后，脉冲计数器和周期计数器开始工作，每次脉冲计数器加一时，会将当前的周期计数器值存到周期数中。

当周期计数器等于配置的周期数时，产生触发信号，将当前的脉冲计数器值和周期数存入快照 0，同时将之前的快照 0 存入快照 1，周期计数器和脉冲计数器会清零重新开始工作。

软件可以在两个快照寄存器中读到最近两次的测速结果。

36.5 QEIV2 寄存器列表

QEIV2 的寄存器列表如下：

QEI0 base address: 0xF0300000

QEI1 base address: 0xF0304000

地址偏移	名称	描述	复位值
0x0000	CR	控制寄存器	0x00000000
0x0004	PHCFG	相位配置寄存器	0x00000000
0x0008	WDGCFG	看门狗配置寄存器	0x00000000
0x000C	PHIDX	相位索引寄存器	0x00000000
0x0010	TRGOEN	触发输出使能寄存器	0x00000000
0x0014	READEN	读取事件使能寄存器	0x00000000

地址偏移	名称	描述	复位值
0x0018	ZCMP	Z 相比较器	0x00000000
0x001C	PHCMP	AB 相比较器	0x00000000
0x0020	SPDCMP	转速比较器	0x00000000
0x0024	DMAEN	DMA 请求使能寄存器	0x00000000
0x0028	SR	状态寄存器	0x00000000
0x002C	IRQEN	中断请求使能寄存器	0x00000000
0x0030	COUNT[CURRENT][Z]	Z 相计数器	0x00000000
0x0034	COUNT[CURRENT][PH]	AB 相计数器	0x00000000
0x0038	COUNT[CURRENT][SPD]	转速计数器	0x00000000
0x003C	COUNT[CURRENT][TMR]	定时器计数器	0x00000000
0x0040	COUNT[READ][Z]	Z 相读取寄存器	0x00000000
0x0044	COUNT[READ][PH]	AB 相读取寄存器	0x00000000
0x0048	COUNT[READ][SPD]	转速读取寄存器	0x00000000
0x004C	COUNT[READ][TMR]	定时器读取寄存器	0x00000000
0x0050	COUNT[SNAP0][Z]	Z 相快照寄存器 0	0x00000000
0x0054	COUNT[SNAP0][PH]	AB 相快照寄存器 0	0x00000000
0x0058	COUNT[SNAP0][SPD]	转速快照寄存器 0	0x00000000
0x005C	COUNT[SNAP0][TMR]	定时器快照寄存器 0	0x00000000
0x0060	COUNT[SNAP1][Z]	Z 相快照寄存器 1	0x00000000
0x0064	COUNT[SNAP1][PH]	AB 相快照寄存器 1	0x00000000
0x0068	COUNT[SNAP1][SPD]	转速快照寄存器 1	0x00000000
0x006C	COUNT[SNAP1][TMR]	定时器快照寄存器 1	0x00000000
0x0080	ZCMP2	Z 相比较器 2	0x00000000
0x0084	PHCMP2	AB 相比较器 2	0x00000000
0x0088	SPDCMP2	转速比较器 2	0x00000000
0x008C	MATCH_CFG	位置比较器配置	0x00000000
0x0090	FILT_CFG[FILT_CFG_A]	A 相信号滤波配置	0x00000000
0x0094	FILT_CFG[FILT_CFG_B]	A 相信号滤波配置	0x00000000
0x0098	FILT_CFG[FILT_CFG_Z]	A 相信号滤波配置	0x00000000
0x009C	FILT_CFG[FILT_CFG_H]	A 相信号滤波配置	0x00000000
0x00A0	FILT_CFG[FILT_CFG_H2]	A 相信号滤波配置	0x00000000
0x00A4	FILT_CFG[FILT_CFG_F]	A 相信号滤波配置	0x00000000
0x0100	QEI_CFG	QEI 配置寄存器	0x00000000
0x0110	PULSE0_NUM	脉冲数 0	0x00000000
0x0114	PULSE1_NUM	脉冲数 1	0x00000000
0x0118	CYCLE0_CNT	周期计数器 0	0x00000000
0x011C	CYCLE0PULSE_CNT	周期 0 脉冲计数器	0x00000000
0x0120	CYCLE1_CNT	周期计数器 1	0x00000000
0x0124	CYCLE1PULSE_CNT	周期 1 脉冲计数器	0x00000000
0x0128	CYCLE0_SNAP0	周期 0 快照 0	0x00000000

地址偏移	名称	描述	复位值
0x012C	CYCLE0_SNAP1	周期 0 快照 1	0x00000000
0x0130	CYCLE1_SNAP0	周期 1 快照 0	0x00000000
0x0134	CYCLE1_SNAP1	周期 1 快照 1	0x00000000
0x0140	CYCLE0_NUM	周期数 0	0x00000000
0x0144	CYCLE1_NUM	周期数 1	0x00000000
0x0148	PULSE0_CNT	脉冲计数器 0	0x00000000
0x014C	PULSE0CYCLE_CNT	脉冲 0 周期计数器	0x00000000
0x0150	PULSE1_CNT	脉冲计数器 1	0x00000000
0x0154	PULSE1CYCLE_CNT	脉冲 1 周期计数器	0x00000000
0x0158	PULSE0_SNAP0	脉冲 0 快照 0	0x00000000
0x015C	PULSE0CYCLE_SNAP0	脉冲 0 周期快照 0	0x00000000
0x0160	PULSE0_SNAP1	脉冲 0 快照 1	0x00000000
0x0164	PULSE0CYCLE_SNAP1	脉冲 0 周期快照 1	0x00000000
0x0168	PULSE1_SNAP0	脉冲 1 快照 0	0x00000000
0x016C	PULSE1CYCLE_SNAP0	脉冲 1 周期快照 0	0x00000000
0x0170	PULSE1_SNAP1	脉冲 1 快照 1	0x00000000
0x0174	PULSE1CYCLE_SNAP1	脉冲 1 周期快照 1	0x00000000
0x0200	ADCX_CFG0	X 通道 adc 配置寄存器 0	0x00000000
0x0204	ADCX_CFG1	X 通道 adc 配置寄存器 1	0x00000000
0x0208	ADCX_CFG2	X 通道 adc 配置寄存器 2	0x00000000
0x0210	ADCY_CFG0	Y 通道 adc 配置寄存器 0	0x00000000
0x0214	ADCY_CFG1	Y 通道 adc 配置寄存器 1	0x00000000
0x0218	ADCY_CFG2	Y 通道 adc 配置寄存器 2	0x00000000
0x0220	CAL_CFG	计算配置	0x00000000
0x0230	PHASE_PARAM	相位计算参数	0x00000000
0x0234	ANGLE_ADJ	角度调整	0x00000000
0x0238	POS_THRESHOLD	位置变化门限	0x00000000
0x0240	UVW_POS[UVW_POS0]	UVW 位置 0 角度	0x00000000
0x0244	UVW_POS[UVW_POS1]	UVW 位置 0 角度	0x00000000
0x0248	UVW_POS[UVW_POS2]	UVW 位置 0 角度	0x00000000
0x024C	UVW_POS[UVW_POS3]	UVW 位置 0 角度	0x00000000
0x0250	UVW_POS[UVW_POS4]	UVW 位置 0 角度	0x00000000
0x0254	UVW_POS[UVW_POS5]	UVW 位置 0 角度	0x00000000
0x0258	UVW_POS_CFG[UVW_POS0_CFG]	UVW 位置 0 配置	0x00000000
0x025C	UVW_POS_CFG[UVW_POS1_CFG]	UVW 位置 0 配置	0x00000000
0x0260	UVW_POS_CFG[UVW_POS2_CFG]	UVW 位置 0 配置	0x00000000
0x0264	UVW_POS_CFG[UVW_POS3_CFG]	UVW 位置 0 配置	0x00000000
0x0268	UVW_POS_CFG[UVW_POS4_CFG]	UVW 位置 0 配置	0x00000000
0x026C	UVW_POS_CFG[UVW_POS5_CFG]	UVW 位置 0 配置	0x00000000
0x0280	PHASE_CNT	AB 相计数器	0x00000000

地址偏移	名称	描述	复位值
0x0284	PHASE_UPDATE	AB 相计数器更新	0x00000000
0x0288	POSITION	位置	0x00000000
0x028C	POSITION_UPDATE	位置改变	0x00000000
0x0290	ANGLE	角度	0x00000000
0x0294	POS_TIMEOUT	位置超时配置	0x00000000

表 177: QEIV2 寄存器列表

36.6 QEIV2 寄存器详细信息

QEIV2 的寄存器详细说明如下：

36.6.1 CR (0x0)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
READ	RSVD								ZCNTCFG	PHCALIZ	Z_ONLY_EN	H2FDIR0	H2FDIR1	H2RDIR0	H2RDIR1	PAUSEPOS	PAUSESPD	PAUSEPH	PAUSEZ	HFDIR0	HFDIR1	HRDIR0	HRDIR1	RSVD	FAULTPOS	SNAPEN	RSTCNT	RD_SEL	ENCTYP		
WO	N/A								RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	N/A	RW	RW	RW	RW	RW		
0	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	x	0	0	0	0	0	0	0

CR [31:0]

位域	名称	描述
31	READ	1：生成软件读取事件，把 PHCNT，ZCNT，SPDCNT 和 TMRCNT 这些寄存器的值，载入对应的读取寄存器
22	ZCNTCFG	1：当 PHCNT 递增计数到达 PHMAX 时，ZCNT 加 1，当 PHCNT 递减计数过 0 时，ZCNT 减 1 0：当 Z 相输入有效时，ZCNT 按照电机旋转反向加 1 或减 1
21	PHCALIZ	1：当 Z 相输入有效时，重置 PHCNT 为 PHIDX(ABZ 数字信号）
20	Z_ONLY_EN	1: 仅用 Z 相信号重置 PHCNT 为 PHIDX（用于模拟 AB 弦波信号，且需要设置 phcaliz）
19	H2FDIR0	1：HOMEF2 标志位在 H2 相输入下降沿，并且电机正向旋转（DIR==0）时置 1
18	H2FDIR1	1：HOMEF2 标志位在 H2 相输入下降沿，并且电机反向选择（DIR==1）时置 1
17	H2RDIR0	1：HOMEF2 标志位在 H2 相输入上升沿，并且电机正向旋转（DIR==0）时置 1
16	H2RDIR1	1：HOMEF2 标志位在 H2 相输入上升沿，并且电机反向选择（DIR==1）时置 1
15	PAUSEPOS	1：当 PAUSE 输入有效时，停止输出位置有效信号
14	PAUSESPD	1：当 PAUSE 输入有效时，暂定 SPDCNT
13	PAUSEPH	1：当 PAUSE 输入有效时，暂定 PHCNT

位域	名称	描述
12	PAUSEZ	1：当 PAUSE 输入有效时，暂定 ZCNT
11	HFDIR0	1：HOMEF 标志位在 H 相输入下降沿，并且电机正向旋转（DIR==0）时置 1
10	HFDIR1	1：HOMEF 标志位在 H 相输入下降沿，并且电机反向选择（DIR==1）时置 1
9	HRDIR0	1：HOMEF 标志位在 H 相输入上升沿，并且电机正向旋转（DIR==0）时置 1
8	HRDIR1	1：HOMEF 标志位在 H 相输入上升沿，并且电机反向选择（DIR==1）时置 1
6	FAULTPOS	1：当 FAULT 输入有效时，停止输出位置有效信号
5	SNAPEN	1：打开快照功能，当 SNAPI 输入有效时，把 PHCNT，ZCNT，SPDCNT 和 TMRCNT 这些寄存器的值，载入对应的快照寄存器
4	RSTCNT	1：将 ZCNT，TMRCNT 和 SPDCNT 重置为 0，将 PHCNT 重置为 PHIDX 的值
3	RD_SEL	读计数器选择，影响转速和定时器计数器内容（包括当前值，读取值，两个快照值） 0：转速和定时器值 1：转速计数器用于位置值；定时器计数器用于角度值
2-0	ENCTYP	解码器模式控制位： 000：ABZ 模式 001：PD 模式 010：UD 模式 011：UVW 模式 100：单相数字模式 101：单项正弦波 110：正交正余弦波 111：保留

CR 位域

36.6.2 PHCFG (0x4)

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

PHCFG [31:0]

位域	名称	描述
31-0	PHMAX	PHCNT 的计数上限，当 PHCNT 计数递增到达 PHMAX 时，会回滚为 0；当 PHCNT 计数递减至 0 时，会回滚为 PHMAX

PHCFG 位域

36.6.3 WDGCFG (0x8)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WDGEN		WDOG_CFG		WDGTO																											
RW		RW		RW																											
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

WDGCFG [31:0]

位域	名称	描述
31	WDGEN	1：使能看门狗计数器
30-28	WDOG_CFG	定义多少个相位计数器变化清零看门狗计数器，防止堵转的发生。 0：每次相位计数器变化会清零看门狗； 2：相位计数器变化超过 2 会清零看门狗； ..... 7：相位计数器变化超过 7 会清零看门狗；
27-0	WDGTO	看门狗计数器的超时值

WDGCFG 位域

36.6.4 PHIDX (0xC)

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

PHIDX [31:0]

位域	名称	描述
31-0	PHIDX	PHCNT 的重置值，如果 PHCALIZ 置 1，当 Z 相输入有效时，PHCNT 重置为 PHIDX

PHIDX 位域

36.6.5 TRGOEN (0x10)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WDGFEN	HOMEFEN	POSCMPFEN	ZPHFEN	ZMISSFEN	WIDTHTMFEN	POS2CMPFEN	DIRCHGFEN	CYCLE0FEN	CYCLE1FEN	PULSE0FEN	PULSE1FEN	HOME2FEN	FAULTFEN	RSVD																	
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	N/A																	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x

TRGOEN [31:0]

位域	名称	描述
31	WDGFEN	WDGF 标志位触发输出 TRGO 使能位
30	HOMEFEN	HOMEF 标志位触发输出 TRGO 使能位
29	POSCMPFEN	POSCMPF 标志位触发输出 TRGO 使能位
28	ZPHFEN	ZPHF 标志位触发输出 TRGO 使能位
27	ZMISSFEN	Z 相缺失标志位触发输出 TRGO 使能位
26	WIDTHTMFEN	信号宽度超长标志位触发输出 TRGO 使能位
25	POS2CMPFEN	POS2CMPF 标志位触发输出 TRGO 使能位
24	DIRCHGFEN	方向改变标志位触发输出 TRGO 使能位
23	CYCLE0FEN	周期计数器 0 标志位触发输出 TRGO 使能位
22	CYCLE1FEN	周期计数器 0 标志位触发输出 TRGO 使能位
21	PULSE0FEN	脉冲计数器 0 标志位触发输出 TRGO 使能位
20	PULSE1FEN	脉冲计数器 1 标志位触发输出 TRGO 使能位
19	HOME2FEN	HOME2F 标志位触发输出 TRGO 使能位
18	FAULTFEN	FAULT 标志位触发输出 TRGO 使能位

TRGOEN 位域

36.6.6 READEN (0x14)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WDGFEN	HOMEFEN	POSCMPFEN	ZPHFEN	ZMISSFEN	WIDTHTMFEN	POS2CMPFEN	DIRCHGFEN	CYCLE0FEN	CYCLE1FEN	PULSE0FEN	PULSE1FEN	HOME2FEN	FAULTFEN	RSVD																	
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	N/A																	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x

READEN [31:0]

位域	名称	描述
31	WDGFEN	WDGF 标志位生成读取事件使能位
30	HOMEFEN	HOMEF 标志位生成读取事件使能位

位域	名称	描述
29	POSCMPFEN	POSCMPF 标志位生成读取事件使能位
28	ZPHFEN	ZPHF 标志位生成读取事件使能位
27	ZMISSFEN	Z 相缺失标志位生成读取事件使能位
26	WIDTHTMFEN	信号宽度超长标志位生成读取事件使能位
25	POS2CMPFEN	POS2CMPF 标志位生成读取事件使能位
24	DIRCHGFEN	方向改变标志位生成读取事件使能位
23	CYCLE0FEN	周期计数器 0 标志位生成读取事件使能位
22	CYCLE1FEN	周期计数器 0 标志位生成读取事件使能位
21	PULSE0FEN	脉冲计数器 0 标志位生成读取事件使能位
20	PULSE1FEN	脉冲计数器 1 标志位生成读取事件使能位
19	HOME2FEN	HOME2F 标志位生成读取事件使能位
18	FAULTFEN	FAULT 标志位生成读取事件使能位

READEN 位域

36.6.7 ZCMP (0x18)

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

ZCMP [31:0]

位域	名称	描述
31-0	ZCMP	ZCNT 计时器位置匹配的比较值，当 ZCNT==ZCMP 时，匹配才成立

ZCMP 位域

36.6.8 PHCMP (0x1C)

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

PHCMP [31:0]

位域	名称	描述
31-0	PHCMP	PHCNT 计时器位置匹配的比较值，当 PHCNT==PHCMP 时，匹配才成立

PHCMP 位域

36.6.9 SPDCMP (0x20)

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

SPDCMP [31:0]

位域	名称	描述
31-0	SPDCMP	SPHCNT 计时器位置匹配的比较值，当 SPDCNT==SPDCMP 时，匹配才成立

SPDCMP 位域

36.6.10 DMAEN (0x24)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WDGFEN	HOMEFEN	POSCMPFEN	ZPHFEN	ZMISSFEN	WIDTHTMFEN	POS2CMPFEN	DIRCHGFEN	CYCLE0FEN	CYCLE1FEN	PULSE0FEN	PULSE1FEN	HOME2FEN	FAULTFEN	RSVD																	
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	N/A																	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x

DMAEN [31:0]

位域	名称	描述
31	WDGFEN	WDGF 标志位生成 DMA 请求使能位
30	HOMEFEN	HOMEF 标志位生成 DMA 请求使能位
29	POSCMPFEN	POSCMPF 标志位生成 DMA 请求使能位
28	ZPHFEN	ZPHF 标志位生成 DMA 请求使能位
27	ZMISSFEN	Z 相缺失标志位生成 DMA 请求使能位
26	WIDTHTMFEN	信号宽度超长标志位生成 DMA 请求使能位
25	POS2CMPFEN	POS2CMPF 标志位生成 DMA 请求使能位
24	DIRCHGFEN	方向改变标志位生成 DMA 请求使能位
23	CYCLE0FEN	周期计数器 0 标志位生成 DMA 请求使能位
22	CYCLE1FEN	周期计数器 0 标志位生成 DMA 请求使能位

位域	名称	描述
21	PULSE0FEN	脉冲计数器 0 标志位生成 DMA 请求使能位
20	PULSE1FEN	脉冲计数器 1 标志位生成 DMA 请求使能位
19	HOME2FEN	HOME2F 标志位生成 DMA 请求使能位
18	FAULTFEN	FAULT 标志位生成 DMA 请求使能位

DMAEN 位域

36.6.11 SR (0x28)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WDGF	HOMEF	POSCMPF	ZPHF	ZMISSF	WIDTHTMF	POS2CMPF	DIRCHGF	CYCLE0F	CYCLE1F	PULSE0F	PULSE1F	HOME2F	FAULTF	RSVD																	
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	N/A																	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x

SR [31:0]

位域	名称	描述
31	WDGF	看门狗超时标志位
30	HOMEF	HOMEF 标志位
29	POSCMPF	位置匹配标志位
28	ZPHF	ZPH 标志位
27	ZMISSF	Z 相缺失标志位
26	WIDTHTMF	信号宽度超长标志位
25	POS2CMPF	POS2CMPF 标志位
24	DIRCHGF	方向改变标志位
23	CYCLE0F	周期计数器 0 标志位
22	CYCLE1F	周期计数器 0 标志位
21	PULSE0F	脉冲计数器 0 标志位
20	PULSE1F	脉冲计数器 1 标志位
19	HOME2F	HOME2F 标志位
18	FAULTF	FAULT 标志位

SR 位域

36.6.12 IRQEN (0x2C)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WDGIE	HOMEIE	POSCMPIE	ZPHIE	ZMISSE	WIDTHTME	POS2CMPE	DIRCHGE	CYCLE0E	CYCLE1E	PULSE0E	PULSE1E	HOME2E	FAULTE	RSVD																	

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	N/A																	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x

IRQEN [31:0]

位域	名称	描述
31	WDGIE	WDGF 标志位生成中断请求使能位
30	HOMEIE	HOMEF 标志位生成中断请求使能位
29	POSCMPIE	POSCMPF 标志位生成中断请求使能位
28	ZPHIE	ZPHF 标志位生成中断请求使能位
27	ZMISSE	Z 相缺失标志位生成中断请求使能位
26	WIDTHTME	信号宽度超长标志位生成中断请求使能位
25	POS2CMPE	POS2CMPF 标志位生成中断请求使能位
24	DIRCHGE	方向改变标志位生成中断请求使能位
23	CYCLE0E	周期计数器 0 标志位生成中断请求使能位
22	CYCLE1E	周期计数器 0 标志位生成中断请求使能位
21	PULSE0E	脉冲计数器 0 标志位生成中断请求使能位
20	PULSE1E	脉冲计数器 1 标志位生成中断请求使能位
19	HOME2E	HOME2F 标志位生成中断请求使能位
18	FAULTE	FAULT 标志位生成中断请求使能位

IRQEN 位域

36.6.13 COUNT[Z] (0x30 + 0x10 * n)

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

COUNT[Z] [31:0]

位域	名称	描述
31-0	ZCNT	Z 相计数器 ZCNT

COUNT[Z] 位域

36.6.14 COUNT[PH] (0x34 + 0x10 * n)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD	DIR	RSVD			ASTAT	BSTAT	RSVD				PHCNT																				

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
N/A	RO	N/A			RO	RO	N/A				RO																				
x	0	x	x	x	0	0	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

COUNT[PH] [31:0]

位域	名称	描述
30	DIR	旋转反向 0：电机正向旋转 1：电机反向旋转
26	ASTAT	A 相输入信号状态 1：A 相输入为逻辑 1 0：A 相输入为逻辑 0
25	BSTAT	B 相输入信号状态 1：B 相输入为逻辑 1 0：B 相输入为逻辑 0
20-0	PHCNT	相位计数器 PHCNT

COUNT[PH] 位域

36.6.15 COUNT[SPD] (0x38 + 0x10 * n)

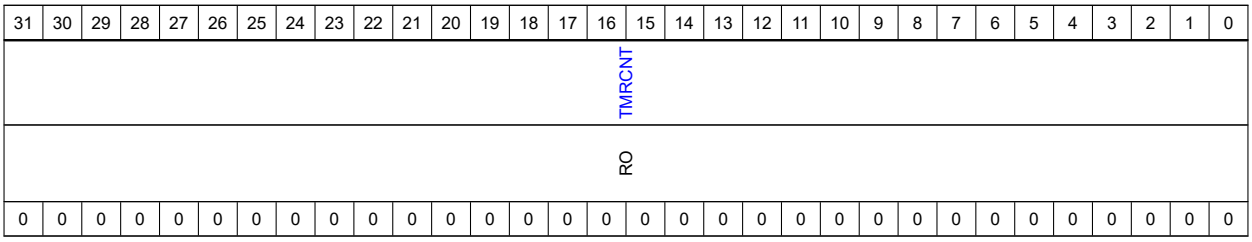
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DIR	ASTAT	BSTAT	RSVD	SPDCNT																											
RO	RO	RW	N/A	RO																											
0	0	0	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

COUNT[SPD] [31:0]

位域	名称	描述
31	DIR	旋转反向 0：电机正向旋转 1：电机反向旋转
30	ASTAT	A 相输入信号状态 1：A 相输入为逻辑 10：A 相输入为逻辑 0
29	BSTAT	B 相输入信号状态 1：B 相输入为逻辑 10：B 相输入为逻辑 0
27-0	SPDCNT	转速计数器 SPDCNT

COUNT[SPD] 位域

36.6.16 COUNT[TMR] (0x3C + 0x10 * n)

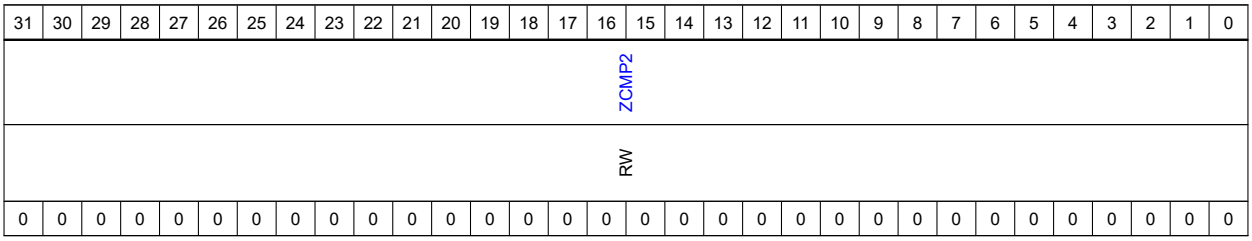


COUNT[TMR] [31:0]

位域	名称	描述
31-0	TMRCNT	定时器计数器 TMRCNT

COUNT[TMR] 位域

36.6.17 ZCMP2 (0x80)

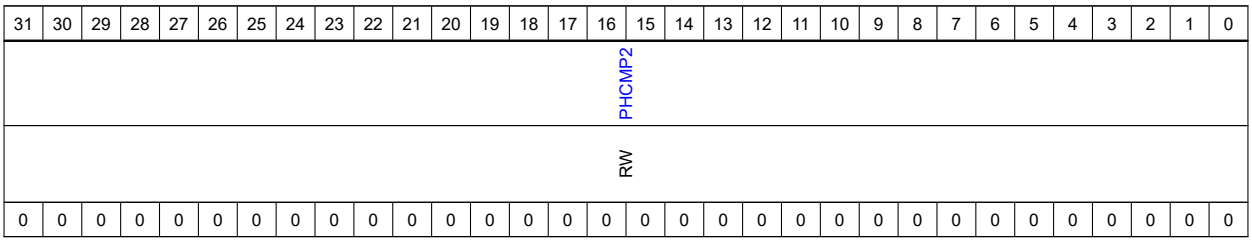


ZCMP2 [31:0]

位域	名称	描述
31-0	ZCMP2	ZCNT 计时器位置匹配的比较值，当 ZCNT==ZCMP2 时，匹配才成立

ZCMP2 位域

36.6.18 PHCMP2 (0x84)



PHCMP2 [31:0]

位域	名称	描述
31-0	PHCMP2	PHCNT 计时器位置匹配的比较值，当 PHCNT==PHCMP2 时，匹配才成立

PHCMP2 位域

36.6.19 SPDCMP2 (0x88)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SPDCMP2																															
RW																															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

SPDCMP2 [31:0]

位域	名称	描述
31-0	SPDCMP2	SPHCNT 计时器位置匹配的比较值，当 SPDCNT==SPDCMP2 时，匹配才成立

SPDCMP2 位域

36.6.20 MATCH_CFG (0x8C)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ZCMPDIS	DIRCMPDIS	DIRCMP	SPDCMPDIS	PHASE_MATCH_DIS	POS_MATCH_DIR	POS_MATCH_OPT	RSVD									ZCMP2DIS	DIRCMP2DIS	DIRCMP2	SPDCMP2DIS	PHASE_MATCH_DIS2	POS_MATCH2_DIR	POS_MATCH2_OPT	RSVD								
RW	RW	RW	RW	RW	RW	RW	N/A									RW	RW	RW	RW	RW	RW	RW	N/A								
0	0	0	0	0	0	0	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	x	x	x	x	x	x	x	x	x

MATCH_CFG [31:0]

位域	名称	描述
31	ZCMPDIS	1：位置匹配时，忽略 ZCMP
30	DIRCMPDIS	1：位置匹配时，忽略电机旋转方向 DIR
29	DIRCMP	配置匹配时的旋转方向比较 1：位置匹配时，电机需反向旋转（DIR==1） 0：位置匹配时，电机需正向旋转（DIR==0）
28	SPDCMPDIS	1：位置匹配时，忽略 SPDCMP
27	PHASE_MATCH_DIS	1：位置匹配时，忽略相位计数器
26	POS_MATCH_DIR	1：position 从小到大 0：position 从大到小
25	POS_MATCH_OPT	1：使用 position 值作位置匹配时，由 pos_match_dir 指定方向 0：使用 position 值作位置匹配时，忽略方向
15	ZCMP2DIS	1：位置匹配时，忽略 ZCMP2

位域	名称	描述
14	DIRCMP2DIS	1：位置匹配时，忽略电机旋转方向 DIR2
13	DIRCMP2	配置匹配时的旋转方向比较 1：位置匹配时，电机需反向旋转（DIR==1） 0：位置匹配时，电机需正向旋转（DIR==0）
12	SPDCMP2DIS	1：位置匹配时，忽略 SPDCMP2
11	PHASE_MATCH_DIS2	1：位置匹配时，忽略相位计数器 2
10	POS_MATCH2_DIR	1：position 从小到大 0：position 从大到小
9	POS_MATCH2_OPT	1：使用 position 值作位置匹配时，由 pos_match_dir 指定方向 0：使用 position 值作位置匹配时，忽略方向

MATCH_CFG 位域

36.6.21 FILT_CFG (0x90 + 0x4 * n)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD															OUTINV	MODE			SYNCEN	FILTLEN											
N/A															RW	RW			RW	RW											
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

FILT_CFG [31:0]

位域	名称	描述
16	OUTINV	1：滤波器输出取反 0：滤波器输出不变
15-13	MODE	此位域定义了滤波器的模式 000：旁路模式 100：滤刺模式 101：延时滤波器 110：滤峰模式 111：滤谷模式
12	SYNCEN	1：将滤波器输入信号于 TRGM 时钟同步 0：不进行同步（有可能会导致时序问题）
11-0	FILTLEN	此位域设置滤波器的长度，11:9 是三位移位值，最大 7；8:0 为滤波器长度基数。 设置 0x100：256 周期； 设置 0x580：0x180«2, 也就是 0x600 周期； 设置 0x900：0x100«4, 也就是 0x1000 周期； 设置 0xFFF：0x1FF«7, 也就是 0xFF80 周期；（大约 320us@200MHz）

位域	名称	描述

FILT_CFG 位域

36.6.22 QEI_CFG (0x100)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																			SPEED_DIR_CHG_EN	RSVD						UVW_POS_OPT0	NEGEDGE_EN	POSIDGE_EN	SIGZ_EN	SIGB_EN	SIGA_EN
N/A																			RW	N/A						RW	RW	RW	RW	RW	RW
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	x	x	x	x	x	x	0	0	0	0	0	0

QEI_CFG [31:0]

位域	名称	描述
12	SPEED_DIR_CHG_EN	置位使能改变方向时，清零所有的脉冲、周期计数器
5	UVW_POS_OPT0	0：在信号边沿输出精确位置信息（MMC 使用） 1：在信号边沿输出下一个位置周期的位置（QEO 使用）
4	NEGEDGE_EN	下降沿使能，用于选择需要的边沿作为周期/脉冲计数器清零或递增信号
3	POSIDGE_EN	上升沿使能，用于选择需要的边沿作为周期/脉冲计数器清零或递增信号
2	SIGZ_EN	Z 相信号使能（也用于 hal 的 W 信号）
1	SIGB_EN	B 相信号使能（也用于 hal 的 V 信号）
0	SIGA_EN	A 相信号使能（也用于 hal 的 U 信号）

QEI_CFG 位域

36.6.23 PULSE0_NUM (0x110)

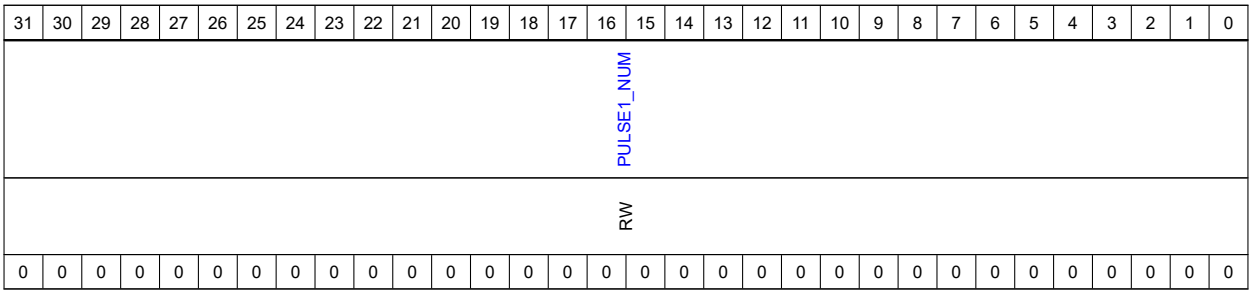
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PULSE0_NUM																															
RW																															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

PULSE0_NUM [31:0]

位域	名称	描述
31-0	PULSE0_NUM	用于速度检测，会统计达到 pulse0_num 的脉冲数所需的时钟周期数，可以在 cycle0_cnt 中读到

PULSE0_NUM 位域

36.6.24 PULSE1_NUM (0x114)

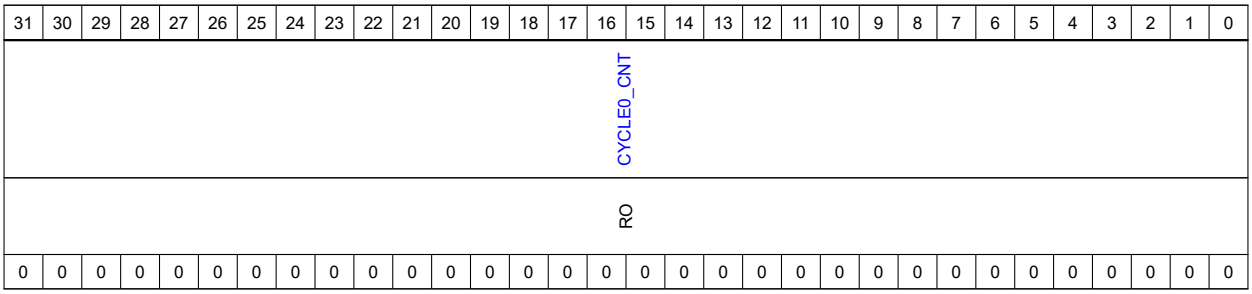


PULSE1_NUM [31:0]

位域	名称	描述
31-0	PULSE1_NUM	用于速度检测，会统计达到 pulse1_num 的脉冲数所需的时钟周期数，可以在 cycle1_cnt 中读到

PULSE1_NUM 位域

36.6.25 CYCLE0_CNT (0x118)

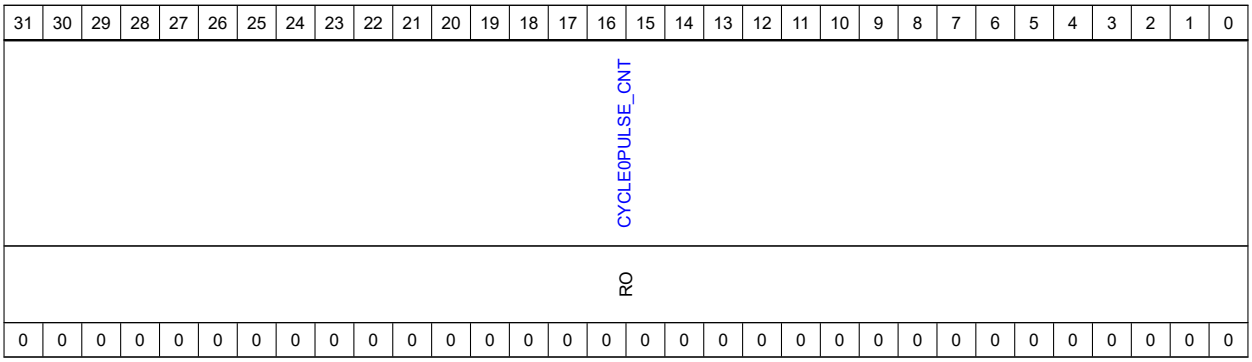


CYCLE0_CNT [31:0]

位域	名称	描述
31-0	CYCLE0_CNT	周期计数器 0 当前值

CYCLE0_CNT 位域

36.6.26 CYCLE0PULSE_CNT (0x11C)

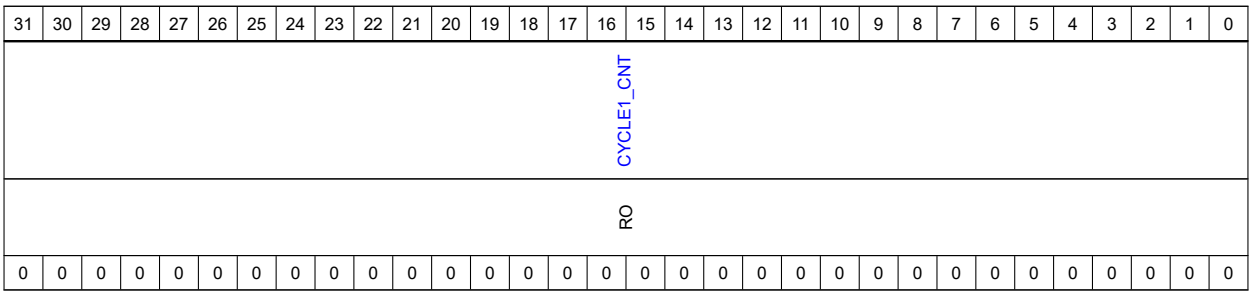


CYCLE0PULSE_CNT [31:0]

位域	名称	描述
31-0	CYCLE0PULSE_CNT	周期计数器 0 对应的脉冲计数器当前值

CYCLE0PULSE_CNT 位域

36.6.27 CYCLE1_CNT (0x120)

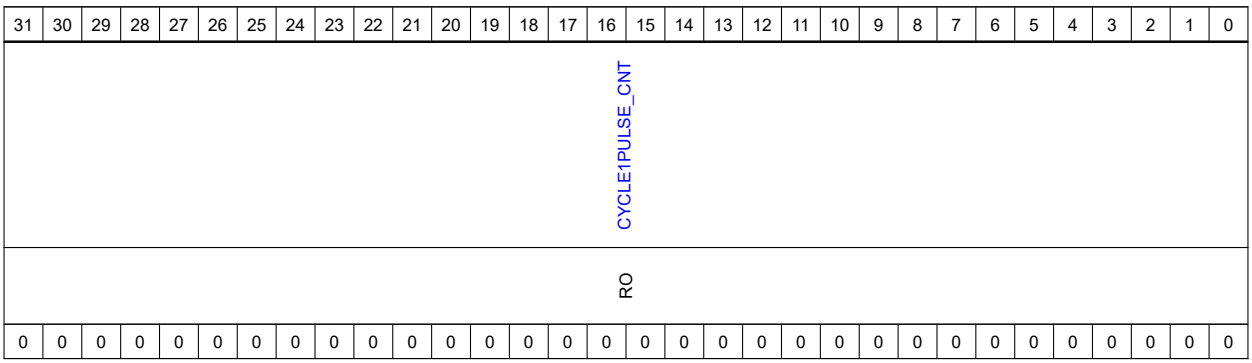


CYCLE1_CNT [31:0]

位域	名称	描述
31-0	CYCLE1_CNT	周期计数器 1 当前值

CYCLE1_CNT 位域

36.6.28 CYCLE1PULSE_CNT (0x124)

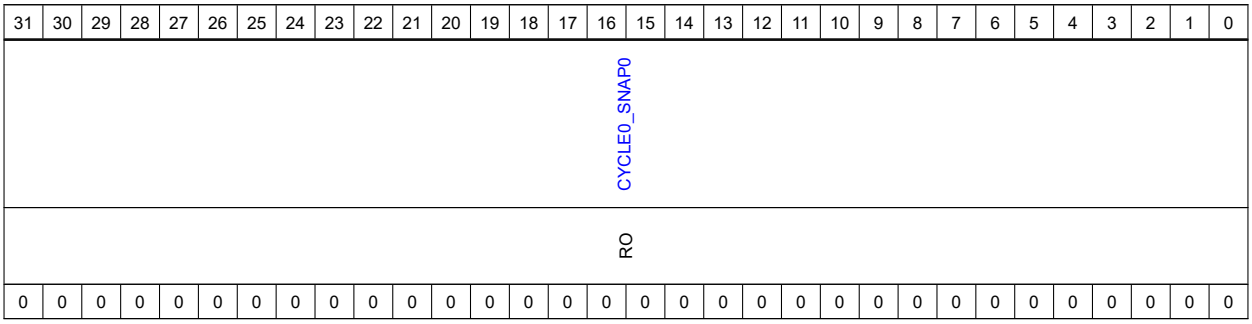


CYCLE1PULSE_CNT [31:0]

位域	名称	描述
31-0	CYCLE1PULSE_CNT	周期计数器 1 对应的脉冲计数器当前值

CYCLE1PULSE_CNT 位域

36.6.29 CYCLE0_SNAP0 (0x128)

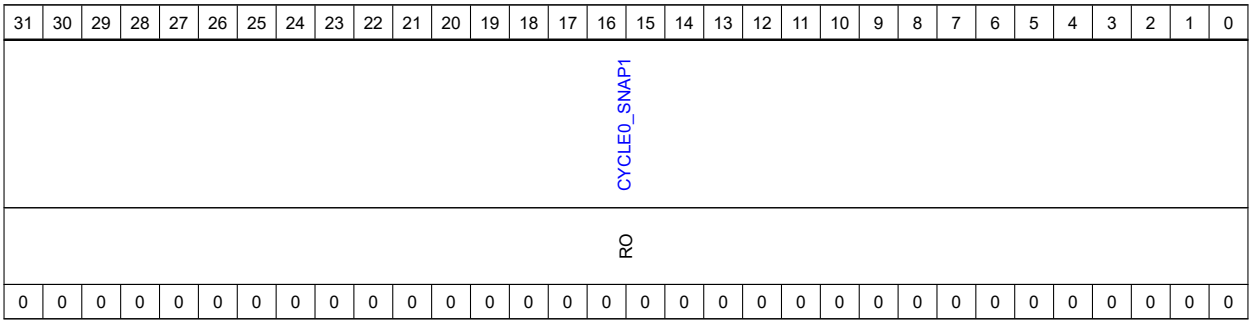


CYCLE0_SNAP0 [31:0]

位域	名称	描述
31-0	CYCLE0_SNAP0	周期计数器 0 快照 0，脉冲计数器计到 pulse0_num 时，会将周期计数器 0 的值存到当前寄存器

CYCLE0_SNAP0 位域

36.6.30 CYCLE0_SNAP1 (0x12C)

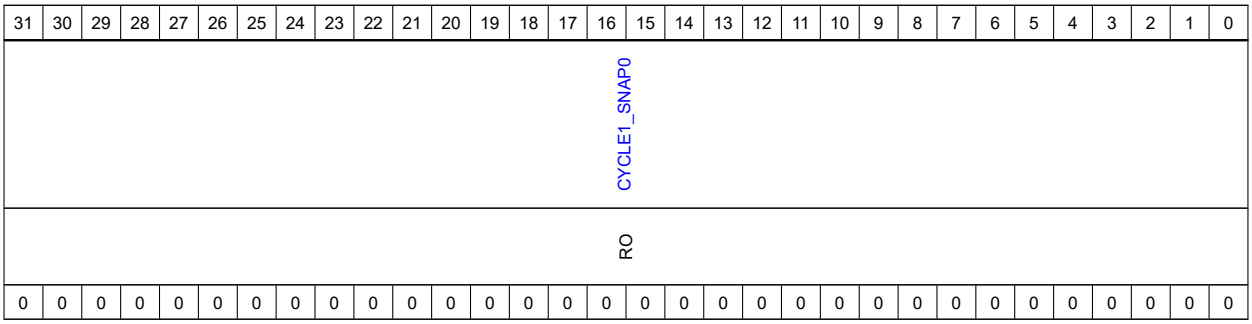


CYCLE0_SNAP1 [31:0]

位域	名称	描述
31-0	CYCLE0_SNAP1	周期计数器 0 快照 1，脉冲计数器计到 pulse0_num 时，会将 cycle0_snap0 的值存到当前寄存器

CYCLE0_SNAP1 位域

36.6.31 CYCLE1_SNAP0 (0x130)

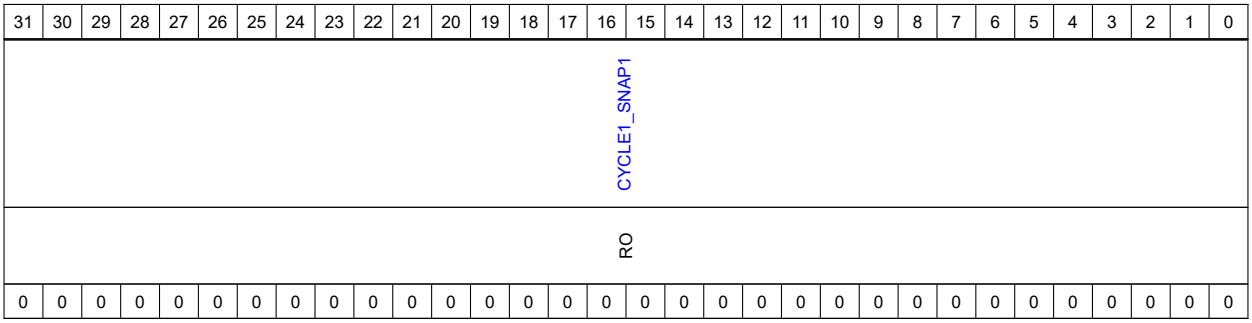


CYCLE1_SNAP0 [31:0]

位域	名称	描述
31-0	CYCLE1_SNAP0	周期计数器 1 快照 0，脉冲计数器计到 pulse1_num 时，会将周期计数器 1 的值存到当前寄存器

CYCLE1_SNAP0 位域

36.6.32 CYCLE1_SNAP1 (0x134)

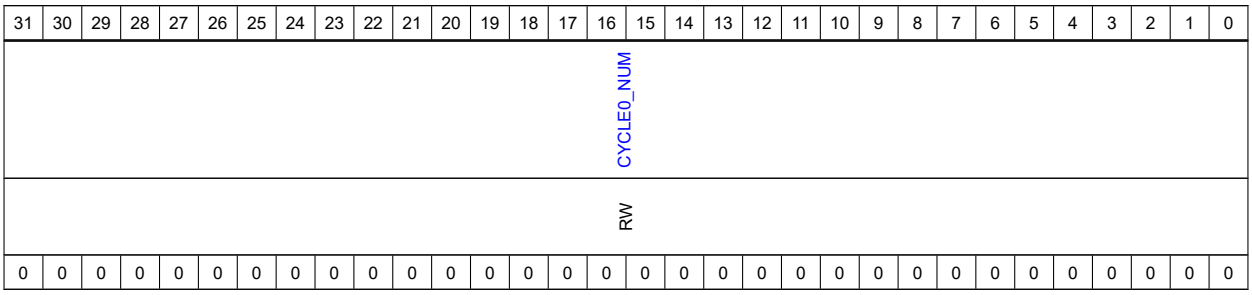


CYCLE1_SNAP1 [31:0]

位域	名称	描述
31-0	CYCLE1_SNAP1	周期计数器 1 快照 1，脉冲计数器计到 pulse1_num 时，会将 cycle1_snap0 的值存到当前寄存器

CYCLE1_SNAP1 位域

36.6.33 CYCLE0_NUM (0x140)



CYCLE0_NUM [31:0]

位域	名称	描述
31-0	CYCLE0_NUM	用于速度检测，会统计达到 cycle0_num 的周期数所检测到的脉冲数， 可以在 pulse0_cnt 中读到, 实际周期数可在 pulse0cycle_cnt 中读到

CYCLE0_NUM 位域

36.6.34 CYCLE1_NUM (0x144)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CYCLE1_NUM																															
RW																															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

CYCLE1_NUM [31:0]

位域	名称	描述
31-0	CYCLE1_NUM	用于速度检测，会统计达到 cycle1_num 的周期数所检测到的脉冲数， 可以在 pulse1_cnt 中读到, 实际周期数可在 pulse1cycle_cnt 中读到

CYCLE1_NUM 位域

36.6.35 PULSE0_CNT (0x148)

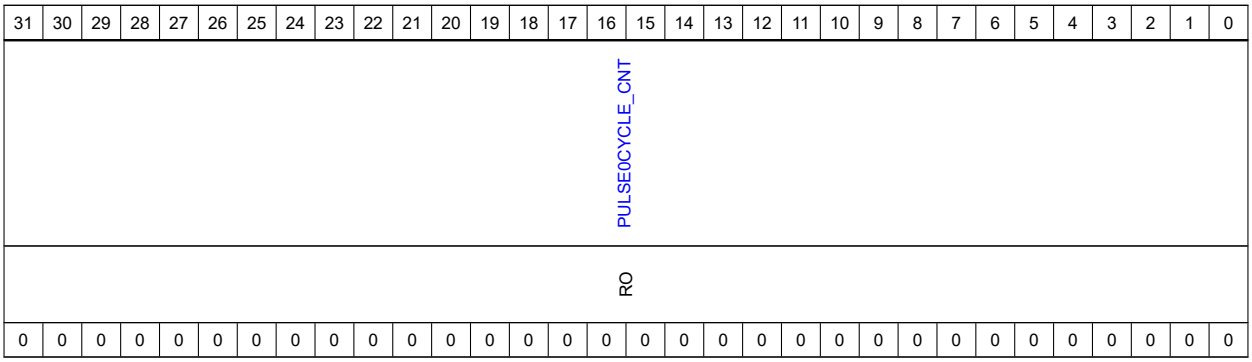
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PULSE0_CNT																															
RO																															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

PULSE0_CNT [31:0]

位域	名称	描述
31-0	PULSE0_CNT	脉冲计数器 0 当前值

PULSE0_CNT 位域

36.6.36 PULSE0CYCLE_CNT (0x14C)

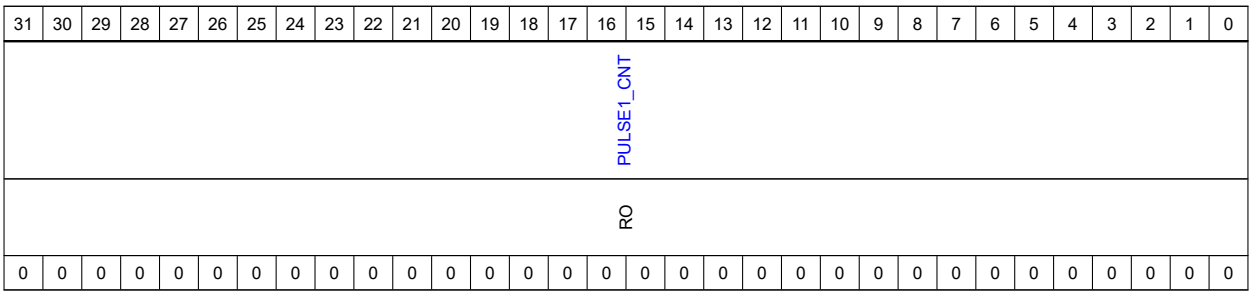


PULSE0CYCLE_CNT [31:0]

位域	名称	描述
31-0	PULSE0CYCLE_CNT	对应脉冲计数器 0 的脉冲数所需周期计数器当前值

PULSE0CYCLE_CNT 位域

36.6.37 PULSE1_CNT (0x150)

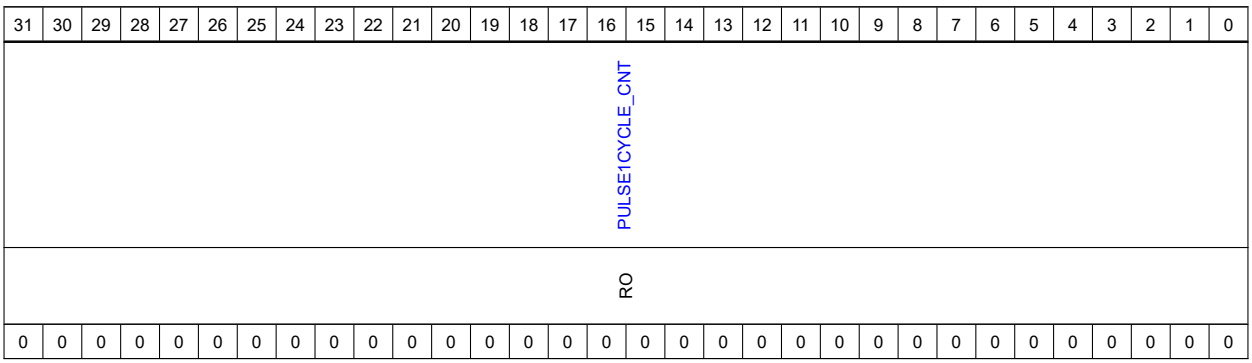


PULSE1_CNT [31:0]

位域	名称	描述
31-0	PULSE1_CNT	脉冲计数器 1 当前值

PULSE1_CNT 位域

36.6.38 PULSE1CYCLE_CNT (0x154)

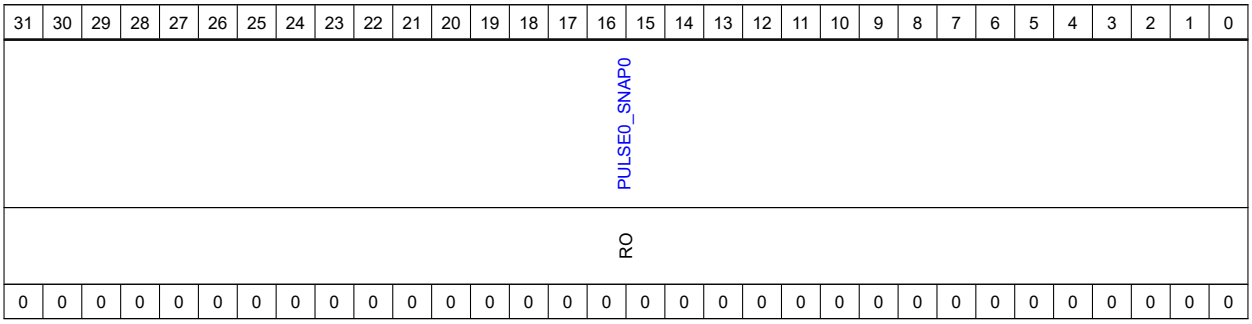


PULSE1CYCLE_CNT [31:0]

位域	名称	描述
31-0	PULSE1CYCLE_CNT	对应脉冲计数器 1 的脉冲数所需周期计数器当前值

PULSE1CYCLE_CNT 位域

36.6.39 PULSE0_SNAP0 (0x158)

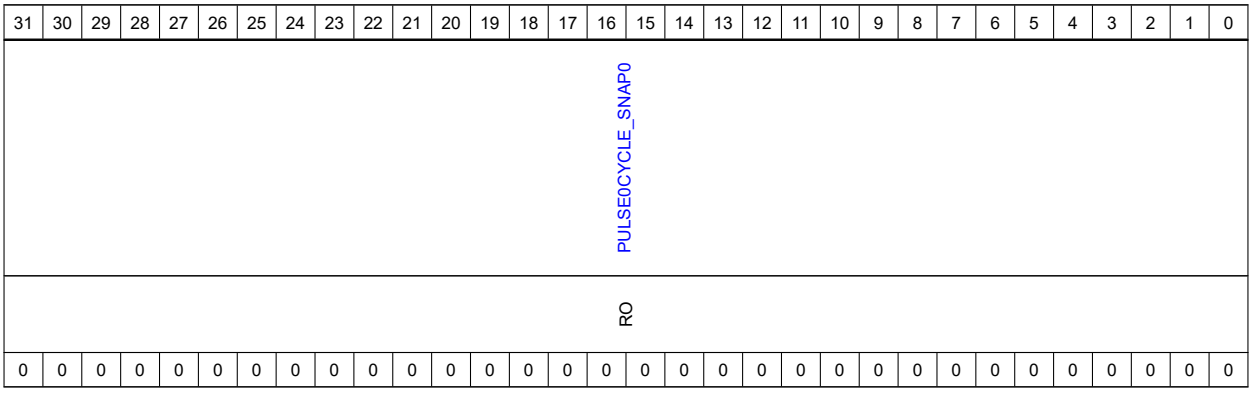


PULSE0_SNAP0 [31:0]

位域	名称	描述
31-0	PULSE0_SNAP0	脉冲计数器 0 快照 0，周期计数器计到 cycle0_num 时，会将脉冲计数器 0 的值存到当前寄存器

PULSE0_SNAP0 位域

36.6.40 PULSE0CYCLE_SNAP0 (0x15C)

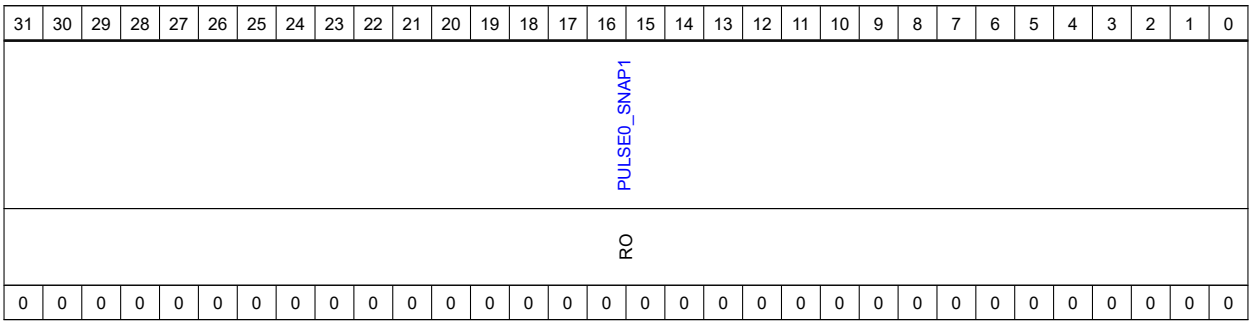


PULSE0CYCLE_SNAP0 [31:0]

位域	名称	描述
31-0	PULSE0CYCLE_SNAP0	脉冲计数器 0 对应的周期快照 0，周期计数器计到 cycle0_num 时，会将对应脉冲计数器 0 的脉冲数所需周期计数器值存到当前寄存器

PULSE0CYCLE_SNAP0 位域

36.6.41 PULSE0_SNAP1 (0x160)

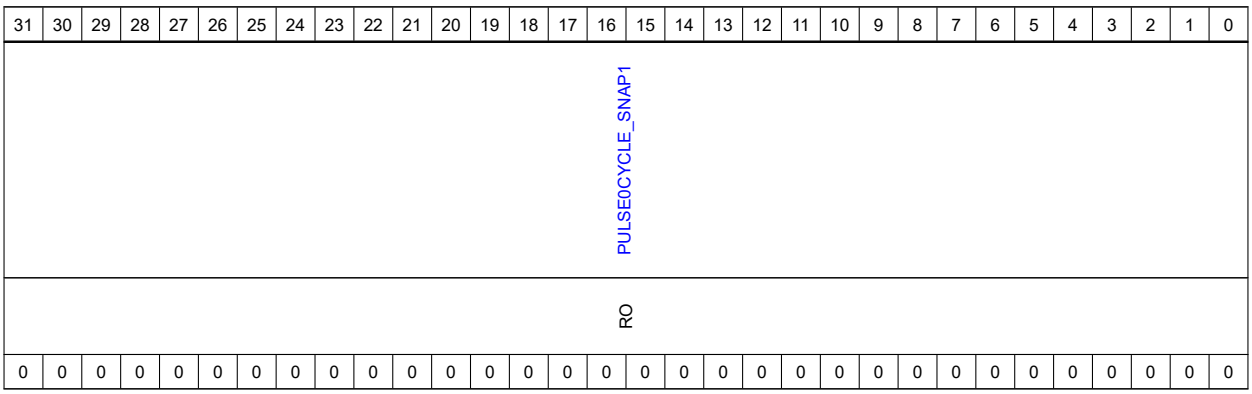


PULSE0_SNAP1 [31:0]

位域	名称	描述
31-0	PULSE0_SNAP1	脉冲计数器 0 快照 1，周期计数器计到 cycle0_num 时，会将 pulse0_snap0 存到当前寄存器

PULSE0_SNAP1 位域

36.6.42 PULSE0CYCLE_SNAP1 (0x164)

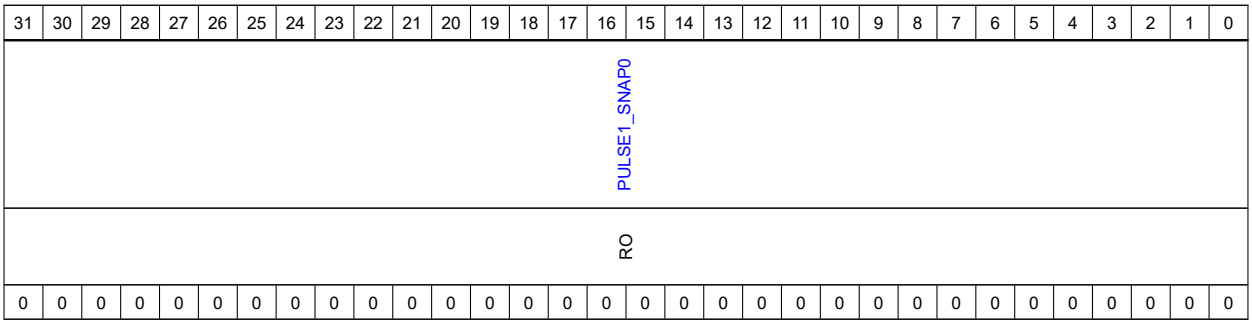


PULSE0CYCLE_SNAP1 [31:0]

位域	名称	描述
31-0	PULSE0CYCLE_SNAP1	脉冲计数器 0 对应的周期快照 0，周期计数器计到 cycle0_num 时，会将 pulse0cycle_snap0 存到当前寄存器

PULSE0CYCLE_SNAP1 位域

36.6.43 PULSE1_SNAP0 (0x168)

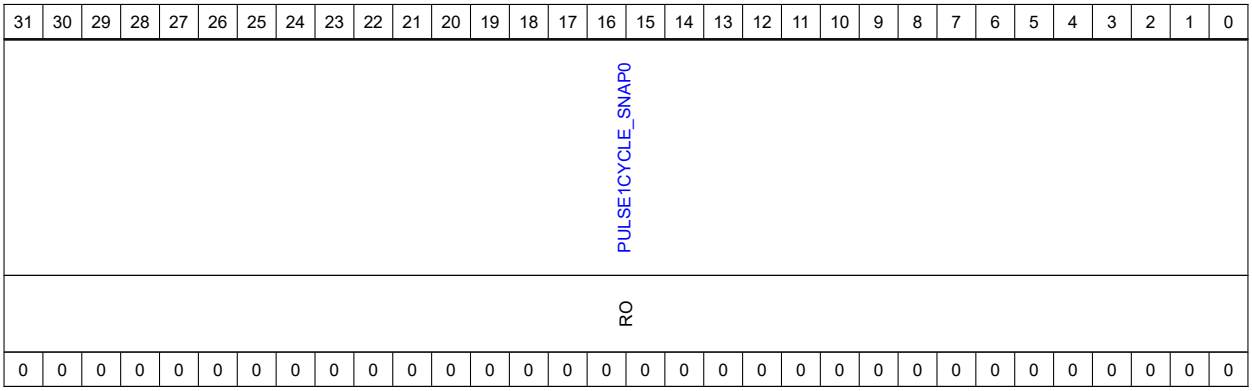


PULSE1_SNAP0 [31:0]

位域	名称	描述
31-0	PULSE1_SNAP0	脉冲计数器 1 快照 0，周期计数器计到 cycle1_num 时，会将脉冲计数器 1 的值存到当前寄存器

PULSE1_SNAP0 位域

36.6.44 PULSE1CYCLE_SNAP0 (0x16C)

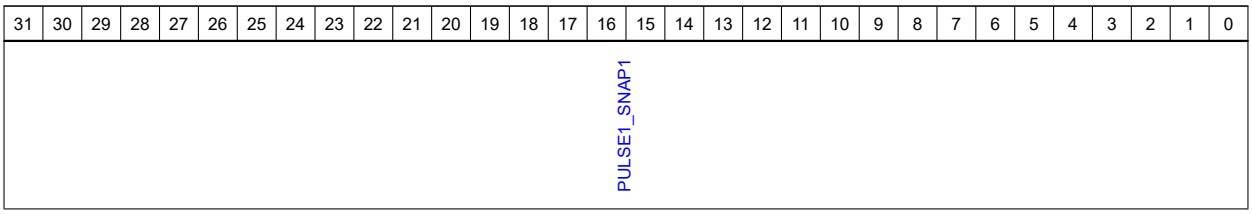


PULSE1CYCLE_SNAP0 [31:0]

位域	名称	描述
31-0	PULSE1CYCLE_SNAP0	脉冲计数器 1 对应的周期快照 0，周期计数器计到 cycle1_num 时，会将对应脉冲计数器 1 的脉冲数所需周期计数器值存到当前寄存器

PULSE1CYCLE_SNAP0 位域

36.6.45 PULSE1_SNAP1 (0x170)



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

PULSE1_SNAP1 [31:0]

位域	名称	描述
31-0	PULSE1_SNAP1	脉冲计数器 1 快照 1，周期计数器计到 cycle1_num 时，会将 pulse1_snap0 存到当前寄存器

PULSE1_SNAP1 位域

36.6.46 PULSE1CYCLE_SNAP1 (0x174)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PULSE1CYCLE_SNAP1																															
RO																															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

PULSE1CYCLE_SNAP1 [31:0]

位域	名称	描述
31-0	PULSE1CYCLE_SNAP1	脉冲计数器 1 对应的周期快照 0，周期计数器计到 cycle1_num 时，会将 pulse1cycle_snap0 存到当前寄存器

PULSE1CYCLE_SNAP1 位域

36.6.47 ADCX_CFG0 (0x200)

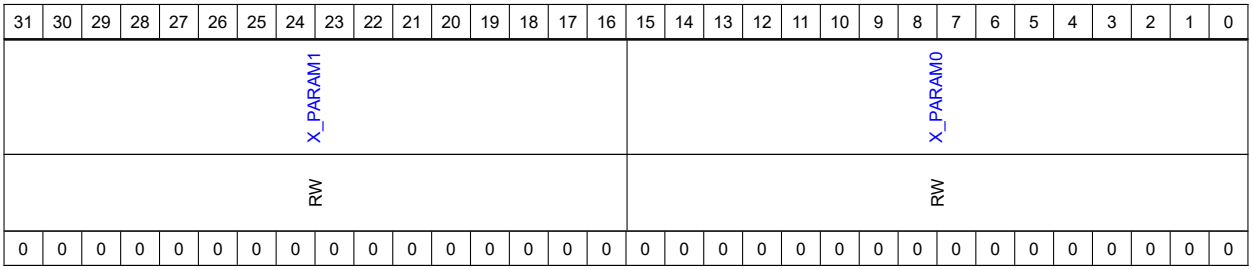
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																							X_ADCSEL	X_ADC_ENABLE	RSVD		X_CHAN				
N/A																							RW	RW	N/A		RW				
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	x	x	0	0	0	0	0

ADCX_CFG0 [31:0]

位域	名称	描述
8	X_ADCSEL	配置 X 通道数据来自于哪个 ADC。设 0 来自于 ADC0；设 1 来自于 ADC1； 一般 X 通道为 COS 信号
7	X_ADC_ENABLE	X 通道使能
4-0	X_CHAN	X 通道来自于 ADC 的通道数

ADCX_CFG0 位域

36.6.48 ADCX_CFG1 (0x204)

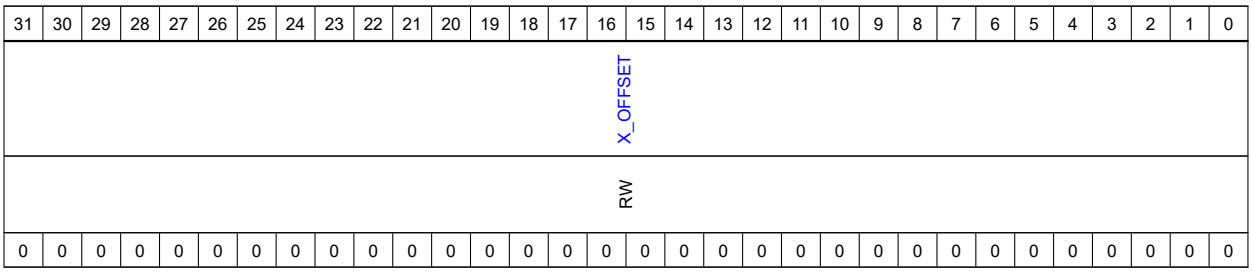


ADCX_CFG1 [31:0]

位域	名称	描述
31-16	X_PARAM1	X 参数 1，有符号数，用于对 Y 数据的 X 分量微调。默认值为 0
15-0	X_PARAM0	X 参数 0，有符号数，用于对 X 数据的大小微调。默认值 0x4000 为 1；0x8000 为-2.0000；0x7FFF 为 1.9999

ADCX_CFG1 位域

36.6.49 ADCX_CFG2 (0x208)



ADCX_CFG2 [31:0]

位域	名称	描述
31-0	X_OFFSET	X 偏移量，用于将 adc 数据转换为有符号数 最终输入到反正切模块的 X 通道数据为： (adc_xvalue-x_offset)*x_param0 + (adc_yvalue-y_offset)*y_param0

ADCX_CFG2 位域

36.6.50 ADCY_CFG0 (0x210)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																							Y_ADCSEL	Y_ADC_ENABLE	RSVD		Y_CHAN				
N/A																							RW	RW	N/A		RW				
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	x	x	0	0	0	0	0

ADCY_CFG0 [31:0]

位域	名称	描述
8	Y_ADCSEL	配置 Y 通道数据来自于哪个 ADC。设 0 来自于 ADC0；设 1 来自于 ADC1； 一般 y 通道为 SIN 信号
7	Y_ADC_ENABLE	Y 通道使能
4-0	Y_CHAN	Y 通道来自于 ADC 的通道数

ADCY_CFG0 位域

36.6.51 ADCY_CFG1 (0x214)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Y_PARAM1																Y_PARAM0															
RW																RW															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ADCY_CFG1 [31:0]

位域	名称	描述
31-16	Y_PARAM1	Y 参数 1，有符号数，用于对 Y 数据的大小微调。默认值 0x4000 为 1；0x8000 为-2.0000；0x7FFF 为 1.9999
15-0	Y_PARAM0	Y 参数 0，有符号数，用于对 X 数据的 Y 分量微调。默认值为 0

ADCY_CFG1 位域

36.6.52 ADCY_CFG2 (0x218)

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

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

ADCY_CFG2 [31:0]

位域	名称	描述
31-0	Y_OFFSET	Y 偏移量，用于将 adc 数据转换为有符号数 最终输入到反正切模块的 Y 通道数据为： (adc_xvalue-x_offset)*x_param1 + (adc_yvalue-y_offset)*y_param1

ADCY_CFG2 位域

36.6.53 CAL_CFG (0x220)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD								XY_DELAY																							
N/A								RW																							
x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

CAL_CFG [31:0]

位域	名称	描述
23-0	XY_DELAY	配置 X，Y 通道超时值，包括以下等待情况： 有 adc 转换信号后等待 adc 数据有效信号； 有一个通道的转换信号等待另一个通道的转换信号； 有一个通道的 adc 数据有效信号，等待另一个通道的数据有效信号； 默认 0x100，1.25us@200MHz；最大约 80ms@200MHz

CAL_CFG 位域

36.6.54 PHASE_PARAM (0x230)

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

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

PHASE_PARAM [31:0]

位域	名称	描述
31-0	PHASE_PARAM	每个相位对应的归一化 32 位位置值；软件需要配置为 2^32/(phmax+1) 比如电机一共 128 相位，每个相位对应于 1/128 位置，该寄存器需要配置为 0x02000000； 如果只有一个相位，该寄存器配置为 0xFFFFFFFF；

PHASE_PARAM 位域

36.6.55 ANGLE_ADJ (0x234)

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

ANGLE_ADJ [31:0]

位域	名称	描述
31-0	ANGLE_ADJ	角度调整值，有符号数。

ANGLE_ADJ 位域

36.6.56 POS_THRESHOLD (0x238)

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

POS_THRESHOLD [31:0]

位域	名称	描述
31-0	POS_THRESHOLD	位置变化门限，当两次位置变化超过这个值，会认为是无效位置，不输出位置有效信号

POS_THRESHOLD 位域

36.6.57 UVW_POS (0x240 + 0x4 * n)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
UVW_POS0																															
RW																															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

UVW_POS [31:0]

位域	名称	描述
31-0	UVW_POS0	UVW 信号位置 0 对应的角度

UVW_POS 位域

36.6.58 UVW_POS_CFG (0x258 + 0x4 * n)

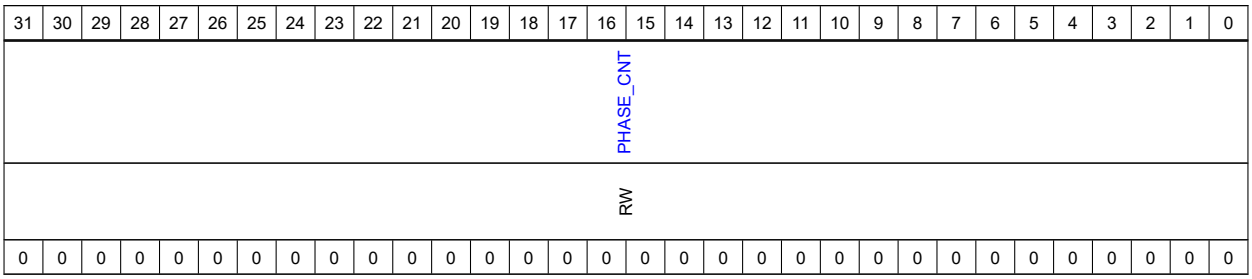
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																									POS_EN	U_POS_SEL		V_POS_SEL		W_POS_SEL	
N/A																									RW	RW		RW		RW	
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0

UVW_POS_CFG [31:0]

位域	名称	描述
6	POS_EN	置位使能以下位置选择信号输出位置信息
5-4	U_POS_SEL	U 信号位置选择： uvw_pos_opt0 为 0 时，00b-低电平；01b-高电平；10b-边沿信号；11b-无用 uvw_pos_opt0 为 1 时，00b-低电平；11b-高电平；其他无效。
3-2	V_POS_SEL	V 信号位置选择： uvw_pos_opt0 为 0 时，00b-低电平；01b-高电平；10b-边沿信号；11b-无用 uvw_pos_opt0 为 1 时，00b-低电平；11b-高电平；其他无效。
1-0	W_POS_SEL	W 信号位置选择： uvw_pos_opt0 为 0 时，00b-低电平；01b-高电平；10b-边沿信号；11b-无用 uvw_pos_opt0 为 1 时，00b-低电平；11b-高电平；其他无效。

UVW_POS_CFG 位域

36.6.59 PHASE_CNT (0x280)

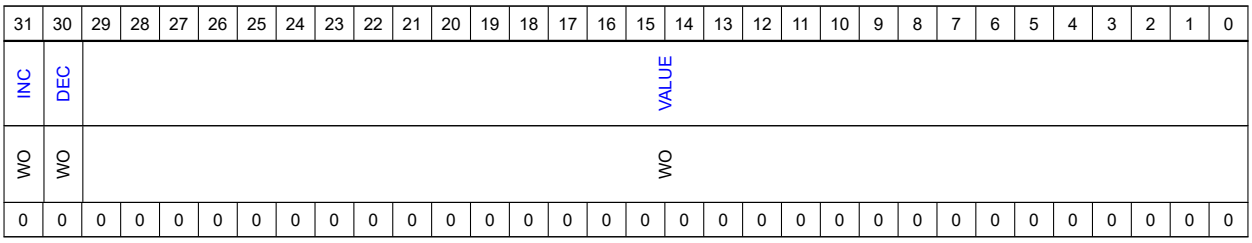


PHASE_CNT [31:0]

位域	名称	描述
31-0	PHASE_CNT	32 位 AB 相计数器，软件可改写；

PHASE_CNT 位域

36.6.60 PHASE_UPDATE (0x284)

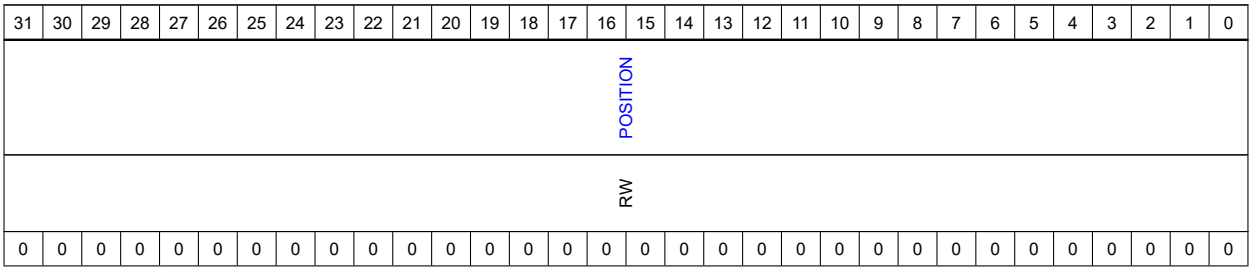


PHASE_UPDATE [31:0]

位域	名称	描述
31	INC	写 1 时，会将当前写操作的位 29 到位 0，加到当前 AB 相计数器上
30	DEC	写 1 时，会将当前写操作的位 29 到位 0，从当前 AB 相计数器上减去。 同时设置 inc 和 dec，会做加操作。
29-0	VALUE	AB 相计数器改变值，只有在 inc 或 dec 置位的同一次 32 位写操作中有效

PHASE_UPDATE 位域

36.6.61 POSITION (0x288)



31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
POSITION [31:0]																															
RW																															
0																															

POSITION [31:0]

位域	名称	描述
31-0	POSITION	位置寄存器，软件可改写；

POSITION 位域

36.6.62 POSITION_UPDATE (0x28C)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
INC	DEC	VALUE																													
WO	WO	WO																													
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

POSITION_UPDATE [31:0]

位域	名称	描述
31	INC	写 1 时，会将当前写操作的位 29 到位 0，加到当前位置上
30	DEC	写 1 时，会将当前写操作的位 29 到位 0，从当前位置上减去。 同时设置 inc 和 dec，会做加操作。
29-0	VALUE	位置改变值，只有在 inc 或 dec 置位的同一次 32 位写操作中有效

POSITION_UPDATE 位域

36.6.63 ANGLE (0x290)

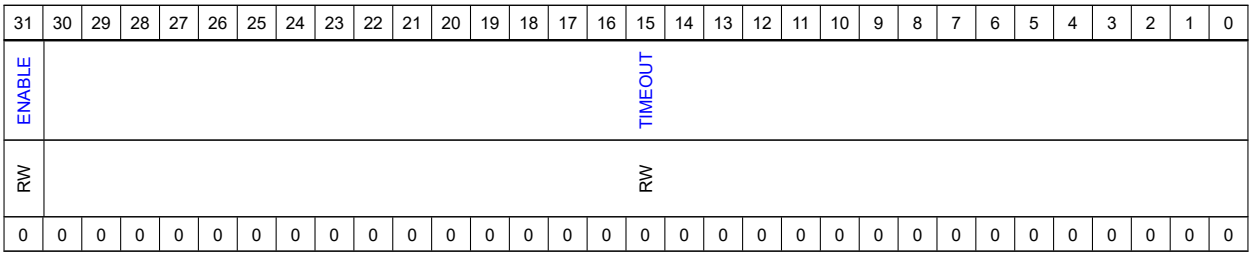
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ANGLE																															
RO																															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ANGLE [31:0]

位域	名称	描述
31-0	ANGLE	角度寄存器，只读。

ANGLE 位域

36.6.64 POS_TIMEOUT (0x294)



POS_TIMEOUT [31:0]

位域	名称	描述
31	ENABLE	位置超时使能，打开时，会在超时后再发送一次之前的位置信息
30-0	TIMEOUT	超时值

POS_TIMEOUT 位域

37 正交编码器输出 QEO

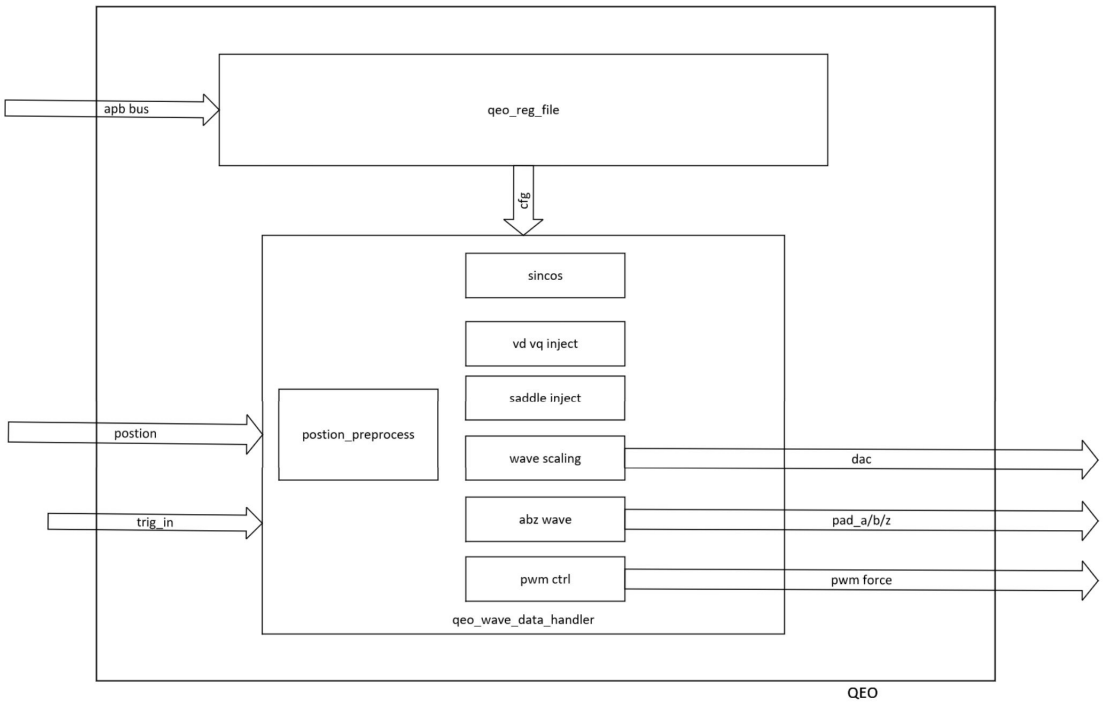
本章节介绍正交编码器输出 QEO 的主要功能和特性。

37.1 特性总结

本章节介绍正交编码器输出 QEO 的主要特性：

- 从系统位置管理器获取位置
- 支持位置补偿
- 支持线数转换
- 支持输出单相，两相（正交，上下，方向脉冲），三相数字脉冲输出
- 支持 INDEX/Z 信号输出
- 可配置最大和最小脉冲宽度
- 支持从 DAC 输出模拟信号
- 支持三相直流无刷，和两相步进控制控制 PWM 输出
- 支持三相直流无刷，和两相步进直接输出
- 支持正弦波/馒头波/马鞍波/锯齿波
- 支持信号放大/偏置等调整
- 超限信号指定占空比

37.2 架构图



37.3 功能描述

37.3.1 位置预处理

QEO 位置来源有两类:

1）系统位置管理器；

2）软件通过写寄存器 POSTION_SOFTWARE 注入位置。

输入 QEO 的位置有效位数为 32 位，该数从 0x00000000 过渡到 0xFFFFFFFF 对应位置过渡 1 个线，对应弧度为 2π, 对应角度为 360°。

每个 QEO 的有效位置输入均会先进行预处理，其包含两个步骤：

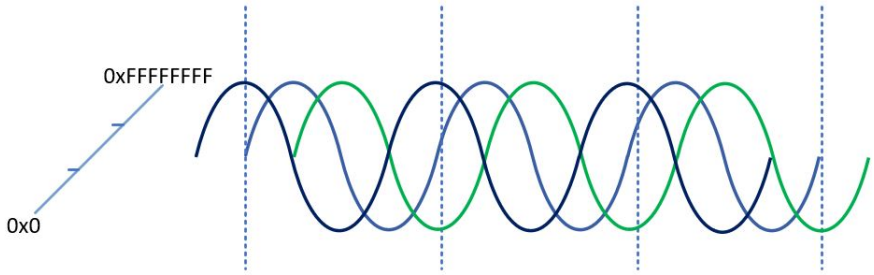
1) 线数转换；

2）位置补偿。

线数转换是指，上游位置输入从 0x00000000 过渡到 0xFFFFFFFF 本来是对应 1 线，通过线数转换模块后，上游输入的位置从 0x00000000 过渡到 0xFFFFFFFF 对应为 n 线，n 为 WAVE/ABZ/PWM[RESOLUTION] 寄存器的值。

位置补偿是指对进行线数转换后的位置进行相位偏移。

弦波、ABZ 以及 PWM force 的线数转换和位置补偿均有独立的配置寄存器。



上图以输出弦波为例，线数转换设置为 3，3 个通道设置为各差 120°，当输入从 0x0 过渡到 0xFFFFFFFF 时，输出的弦波会输出三个周期的弦波。

37.3.2 弦波输出

弦波支持 4 种输出模式：

1) 余弦波；2）馒头波；3）马鞍波；4）锯齿波。

每个输出模式均配备 3 个输出通道，可通过 WAVE[PHASH_SHIFT][WAVEn] 独立配置相位偏移。

37.3.2.1 弦波的 VD/VQ 控制

弦波的 3 个输出通道，均可以通过软件配置 WAVE[VD_VQ_INJECT] 寄存器来控制各个弦波对应的 VD/VQ 值。

$$V = \cos(angle) * VD - \sin(angle) * VQ \quad (2)$$

V 代表经过反 park 变化的输出值，angle 为 QEO 的输入位置经预处理后对应的弧度值。

37.3.2.2 弦波输出中的马鞍波模式

当选择输出马鞍波时，马鞍波的模式有两种，软件可以通过配置 WAVE[MODE].SADDLE_TYPE 来选择：

1) 标准马鞍波。仅当三个输出通道相位相差为 120° 时才能正常输出。

2) 采用三角波叠加成马鞍波，此模式对三个输出通道的相位差没有要求。

37.3.2.3 弦波的缩放和限幅

弦波输出的每个通道均支持独立的缩放和限幅。缩放倍数由 WAVE[AMPLITUDE] 寄存器控制。缩放倍数可以配置为 n * (1/4096), n 可以为 0-65535 直接任意整数，n 来自于寄存器 WAVE[AMPLITUDE][15:0]。

缩放后的输出部分可以由 WAVE[MID_POINT][WAVEn] 进行中心点调整。超过弦波输出量程的部分可以通过配置 WAVE[MODE][WAVEn_ABOVE_MAX_LIMIT] 和 WAVE[MODE][WAVEn_BELOW_MIN_LIMIT] 完成对应的限幅。量程内也支持限幅控制，具体请参考 WAVE[LIMIT][WAVEn][MAX] 和 WAVE[LIMIT][WAVEn][MIN]。

37.3.2.4 弦波的死区补偿

弦波输出的各个通道,经过限幅后还可以进行防死区补偿。可以通过配置 WAVE[DEADZONE_SHIFT][WAVEn] 来进行补偿。

37.3.3 ABZ 输出

ABZ 为数字脉冲输出，它支持：

1) 带自有位置的 A/B 两相正交输出；

2) 高速脉冲 Pulse&Revise 输出；

3) 上下 up&down 输出；

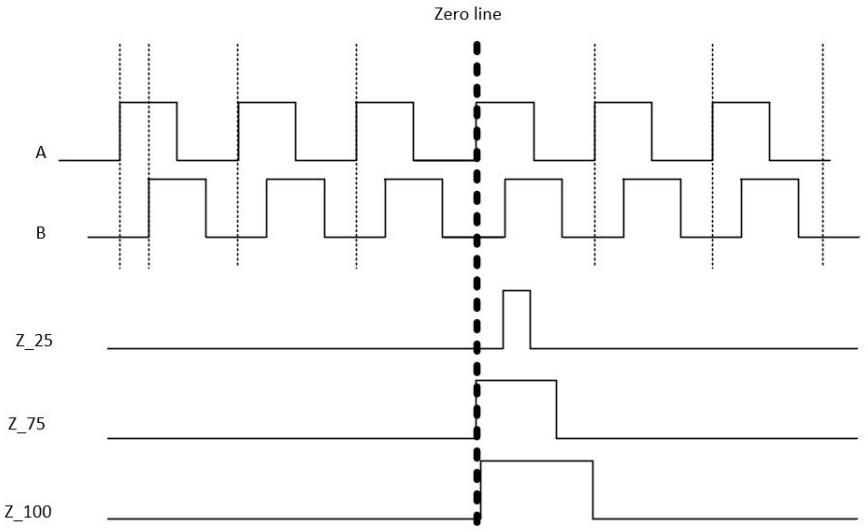
4) 不带自有位置的 A/B 两相正交输出。

37.3.3.1 带自有位置的 A/B 两相正交输出

QEO 的 ABZ 输出模块内置了 ABZ 内部位置管理器。系统复位后，内部位置归零。当 QEO 接收到新的输入位置后，内部位置会按 ABZ[LINE_WIDTH] 设置的周期数为最小周期去追踪输入位置，当内部位置和输入位置一致后，停止追踪。最小周期对应 A、B 输出的 1/4 线对应的最小周期数。

软件可以通过配置 ABZ[POSTION_SYNC], 将 QEO 接收到的下一笔有效位置做为内部位置管理器的位置，此功能可以用来配置启动时刻的 ABZ 起始位置，防止 ABZ 误追踪。

当 ABZ 选择为带自有位置的 A/B 两相正交输出时，A、B、Z 的输出就是由内部位置值来决定的。



上图中 Z_25、Z_75、Z_100 为 Z 相的三种输出模式，由 ABZ[MODE].Z_TYPE 寄存器控制。

37.3.3.2 高速脉冲 Pulse&Revise 输出

高速脉冲输出模式也是采用 ABZ 内部位置.

A 相输出时钟 pulse，当内部位置正向追踪输入位置时，B 相维持输出高电平；当内部位置反向追踪时，B 相维持输出低电平。

37.3.3.3 上下 up&down 输出

上下 up&down 输出模式，同样是采用 ABZ 内部位置。

当内部位置正向追踪输入位置时，A 相输出时钟 pulse；当内部位置反向追踪时，B 相输出时钟 pulse。

37.3.3.4 不带自有位置的 A/B 两相正交输出

此模式下，A/B/Z 的输出由 QEO 实时输入位置决定。最小周期和看门狗配置对此模式均无效。

37.3.3.5 内部位置管理器的看门狗

为防止系统长时间没有位置更新时,用户无法判断系统是否处于正常模式,软件可以配置 ABZ[MODE].EN_WDOG 和 ABZ[WDOG_WIDTH] 来使能看门狗功能，使能看门狗功能后，若 QEO 输入位置在设定的周期数内无变化时，内部位置会自动正向步进 1/4 线，再反向步进 1/4 线，或者先反向步进 1/4 线，再正向步进 1/4 线。

37.3.4 PWM FORCE 输出

QEO 提供了 8 组 PWM FORCE 控制信号。每组 PWM FORCE 对应两 bit 控制信号。PWM FORCE 控制信号是通过输入位置来查找 PWM 换相表来确定的。

输入位置来自 QEO 的位置预处理模块，软件可以通过配置 PWM[MODE].PHASE_NUM 来确定换相数。最大换相数为 12。然后将换相表填入 PWM [PHASE_TABLE] [POSEDGEn] 或 PWM [PHASE_TABLE] [NEGEDGEn]，n 的个数取决于 PWM[MODE].PHASE_NUM 配置的换相数。采用 PWM [PHASE_TABLE] [POSEDGEn] 或 PWM [PHASE_TABLE] [NEGEDGEn] 由 PWM[MODE].REVISE_UP_DN 寄存器决定。

本模块还提供了安全模式的 PWM_FORCE 查找表。该查找表由 PWM[MODE].PWMn_SAFETY 寄存器决定。支持两种进入安全模式的的方法：

1) 通过 TRIG_MUX 配置 QEO.TRIGGER_IN[1]，当来源为高时进入安全模式。

2) 通过配置 PWM[MODE].PWM_ENTER_SAFETY_MODE 进入安全模式。

37.4 QEO 寄存器列表

QEO 的寄存器列表如下：

QEO0 base address: 0xF0308000

QEO1 base address: 0xF030C000

地址偏移	名称	描述	复位值
0x0000	WAVE[MODE]	弦波输出模式	0x00000000
0x0004	WAVE[RESOLUTION]	wave0/1/2 的线数	0x00000000
0x0008	WAVE[PHASE_SHIFT][WAVE0]	wave0 的相位偏移	0x00000000
0x000C	WAVE[PHASE_SHIFT][WAVE1]	wave1 的相位偏移	0x00000000
0x0010	WAVE[PHASE_SHIFT][WAVE2]	wave2 的相位偏移	0x00000000
0x0014	WAVE[VD_VQ_INJECT][WAVE0]	wave0 vd vq 注入值	0x00000000
0x0018	WAVE[VD_VQ_INJECT][WAVE1]	wave1 vd vq 注入值	0x00000000
0x001C	WAVE[VD_VQ_INJECT][WAVE2]	wave2 vd vq 注入值	0x00000000
0x0020	WAVE[VD_VQ_LOAD]	加载 wave0/1/2 vd vq 注入值	0x00000000
0x0024	WAVE[AMPLITUDE][WAVE0]	wave0 幅值放大倍数	0x00000000
0x0028	WAVE[AMPLITUDE][WAVE1]	wave1 幅值放大倍数	0x00000000
0x002C	WAVE[AMPLITUDE][WAVE2]	wave2 幅值放大倍数	0x00000000
0x0030	WAVE[MID_POINT][WAVE0]	wave0 输出位置的中心点偏移	0x00000000
0x0034	WAVE[MID_POINT][WAVE1]	wave1 输出位置的中心点偏移	0x00000000
0x0038	WAVE[MID_POINT][WAVE2]	wave2 输出位置的中心点偏移	0x00000000
0x003C	WAVE[LIMIT][WAVE0][MIN]	wave0 低区限幅值	0x00000000
0x0040	WAVE[LIMIT][WAVE0][MAX]	wave0 高区限幅值	0xFFFFFFFF
0x0044	WAVE[LIMIT][WAVE1][MIN]	wave1 低区限幅值	0x00000000
0x0048	WAVE[LIMIT][WAVE1][MAX]	wave1 高区限幅值	0xFFFFFFFF
0x004C	WAVE[LIMIT][WAVE2][MIN]	wave2 低区限幅值	0x00000000
0x0050	WAVE[LIMIT][WAVE2][MAX]	wave2 高区限幅值	0xFFFFFFFF
0x0054	WAVE[DEADZONE_SHIFT][WAVE0]	wave0 死区偏移值	0x00000000
0x0058	WAVE[DEADZONE_SHIFT][WAVE1]	wave1 死区偏移值	0x00000000
0x005C	WAVE[DEADZONE_SHIFT][WAVE2]	wave2 死区偏移值	0x00000000
0x0060	ABZ[MODE]	wave_a/b/z 输出模式	0x00000000
0x0064	ABZ[RESOLUTION]	wave_a/b/z 的线数	0x00000000
0x0068	ABZ[PHASE_SHIFT][A]	wave_a 的相位偏移	0x00000000

地址偏移	名称	描述	复位值
0x006C	ABZ[PHASE_SHIFT][B]	wave_b 的相位偏移	0x00000000
0x0070	ABZ[PHASE_SHIFT][Z]	wave_z 的相位偏移	0x00000000
0x0074	ABZ[LINE_WIDTH]	两相正交输出 1/4 周期长度	0x00000000
0x0078	ABZ[WDOG_WIDTH]	qeo 输出的看门狗等待时间	0x00000000
0x007C	ABZ[POSTION_SYNC]	同步 abz 内部位置	0x00000000
0x0080	PWM[MODE]	pwm 模式	0x00000000
0x0084	PWM[RESOLUTION]	pwm force 输出的线数	0x00000000
0x0088	PWM[PHASE_SHIFT][A]	pwm_a 的相位偏移	0x00000000
0x008C	PWM[PHASE_SHIFT][B]	pwm_b 的相位偏移	0x00000000
0x0090	PWM[PHASE_SHIFT][C]	pwm_c 的相位偏移	0x00000000
0x0094	PWM[PHASE_SHIFT][D]	pwm_d 的相位偏移	0x00000000
0x0098	PWM[PHASE_TABLE][POSEDGE0]	pwm 正向换相表 0	0x00000000
0x009C	PWM[PHASE_TABLE][POSEDGE1]	pwm 正向换相表 1	0x00000000
0x00A0	PWM[PHASE_TABLE][POSEDGE2]	pwm 正向换相表 2	0x00000000
0x00A4	PWM[PHASE_TABLE][POSEDGE3]	pwm 正向换相表 3	0x00000000
0x00A8	PWM[PHASE_TABLE][POSEDGE4]	pwm 正向换相表 4	0x00000000
0x00AC	PWM[PHASE_TABLE][POSEDGE5]	pwm 正向换相表 5	0x00000000
0x00B0	PWM[PHASE_TABLE][POSEDGE6]	pwm 正向换相表 6	0x00000000
0x00B4	PWM[PHASE_TABLE][POSEDGE7]	pwm 正向换相表 7	0x00000000
0x00B8	PWM[PHASE_TABLE][POSEDGE8]	pwm 正向换相表 8	0x00000000
0x00BC	PWM[PHASE_TABLE][POSEDGE9]	pwm 正向换相表 9	0x00000000
0x00C0	PWM[PHASE_TABLE][POSEDGE10]	pwm 正向换相表 10	0x00000000
0x00C4	PWM[PHASE_TABLE][POSEDGE11]	pwm 正向换相表 11	0x00000000
0x00C8	PWM[PHASE_TABLE][NEGEDGE0]	pwm 反向换相表 0	0x00000000
0x00CC	PWM[PHASE_TABLE][NEGEDGE1]	pwm 反向换相表 1	0x00000000
0x00D0	PWM[PHASE_TABLE][NEGEDGE2]	pwm 反向换相表 2	0x00000000
0x00D4	PWM[PHASE_TABLE][NEGEDGE3]	pwm 反向换相表 3	0x00000000
0x00D8	PWM[PHASE_TABLE][NEGEDGE4]	pwm 反向换相表 4	0x00000000
0x00DC	PWM[PHASE_TABLE][NEGEDGE5]	pwm 反向换相表 5	0x00000000
0x00E0	PWM[PHASE_TABLE][NEGEDGE6]	pwm 反向换相表 6	0x00000000
0x00E4	PWM[PHASE_TABLE][NEGEDGE7]	pwm 反向换相表 7	0x00000000
0x00E8	PWM[PHASE_TABLE][NEGEDGE8]	pwm 反向换相表 8	0x00000000
0x00EC	PWM[PHASE_TABLE][NEGEDGE9]	pwm 反向换相表 9	0x00000000
0x00F0	PWM[PHASE_TABLE][NEGEDGE10]	pwm 反向换相表 10	0x00000000
0x00F4	PWM[PHASE_TABLE][NEGEDGE11]	pwm 反向换相表 11	0x00000000
0x00F8	POSTION_SOFTWARE	软件注入的位置	0x00000000
0x00FC	POSTION_SEL	选择软件注入的位置	0x00000000
0x0100	STATUS	状态寄存器	0x00000000
0x0104	DEBUG0	debug 状态寄存器 0	0x00000000
0x0108	DEBUG1	debug 状态寄存器 1	0x00000000

地址偏移	名称	描述	复位值
0x010C	DEBUG2	debug 状态寄存器 2	0x00000000
0x0110	DEBUG3	debug 状态寄存器 3	0x00000000

表 178: QEO 寄存器列表

37.5 QEO 寄存器详细信息

QEO 的寄存器详细说明如下：

37.5.1 WAVE[MODE] (0x0)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WAVE2_ABOVE_MAX_LIMIT		WAVE2_HIGH_AREA1_LIMIT	WAVE2_HIGH_AREA0_LIMIT	WAVE2_LOW_AREA1_LIMIT	WAVE2_LOW_AREA0_LIMIT	WAVE2_BELOW_MIN_LIMIT		WAVE1_ABOVE_MAX_LIMIT		WAVE1_HIGH_AREA1_LIMIT	WAVE1_HIGH_AREA0_LIMIT	WAVE1_LOW_AREA1_LIMIT	WAVE1_LOW_AREA0_LIMIT	WAVE1_BELOW_MIN_LIMIT		WAVE0_ABOVE_MAX_LIMIT		WAVE0_HIGH_AREA1_LIMIT	WAVE0_HIGH_AREA0_LIMIT	WAVE0_LOW_AREA1_LIMIT	WAVE0_LOW_AREA0_LIMIT	WAVE0_BELOW_MIN_LIMIT		SADDLE_TYPE	EN_WAVE2_VD_VQ_INJECT	EN_WAVE1_VD_VQ_INJECT	EN_WAVE0_VD_VQ_INJECT	RSVD		WAVES_OUTPUT_TYPE	
RW		RW	RW	RW	RW	RW		RW		RW	RW	RW	RW	RW		RW		RW	RW	RW	RW	RW		RW	RW	RW	RW	N/A		RW	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	x	x	0	0

WAVE[MODE] [31:0]

位域	名称	描述
31-30	WAVE2_ABOVE_MAX_LIMIT	wave2 缩放后大于满量程的部分的限幅模式。 0: 输出 0xffff. 1: 输出 0x0. 2: 输出 level_max_limit2.level0_max_limit 寄存器里的值
29	WAVE2_HIGH_AREA1_LIMIT	wave2 高区 1 的限幅模式. 0: 输出 0xffff. 1: 输出 level_max_limit2.level0_max_limit 寄存器里的值
28	WAVE2_HIGH_AREA0_LIMIT	wave2 高区 0 的限幅模式. 0: 输出 0xffff. 1: 输出 level_max_limit2.level0_max_limit 寄存器里的值
27	WAVE2_LOW_AREA1_LIMIT	wave2 低区 1 的限幅模式。 0: 输出 0. 1: 输出 level_min_limit2.level1_min_limit 寄存器里的值。
26	WAVE2_LOW_AREA0_LIMIT	wave2 低区 0 的限幅模式。 0: 输出 0. 1: 输出 level_min_limit2.level1_min_limit 寄存器里的值。

位域	名称	描述
25-24	WAVE2_BELOW_MIN_LIMIT	wave2 缩放后小于 0 的部分的限幅模式。 0: 输出 0. 1: 输出 0xffff. 2: 输出 level_min_limit2.level1_min_limit 寄存器里的值
23-22	WAVE1_ABOVE_MAX_LIMIT	wave1 缩放后大于满量程的部分的限幅模式。 0: 输出 0xffff. 1: 输出 0x0. 2: 输出 level_max_limit1.level0_max_limit 寄存器里的值
21	WAVE1_HIGH_AREA1_LIMIT	wave1 高区 1 的限幅模式. 0: 输出 0xffff. 1: 输出 level_max_limit1.level0_max_limit 寄存器里的值
20	WAVE1_HIGH_AREA0_LIMIT	wave1 高区 0 的限幅模式. 0: 输出 0xffff. 1: 输出 level_max_limit1.level0_max_limit 寄存器里的值
19	WAVE1_LOW_AREA1_LIMIT	wave1 低区 1 的限幅模式。 0: 输出 0. 1: 输出 level_min_limit1.level1_min_limit 寄存器里的值。
18	WAVE1_LOW_AREA0_LIMIT	wave1 低区 0 的限幅模式。 0: 输出 0. 1: 输出 level_min_limit1.level1_min_limit 寄存器里的值。
17-16	WAVE1_BELOW_MIN_LIMIT	wave1 缩放后小于 0 的部分的限幅模式。 0: 输出 0. 1: 输出 0xffff. 2: 输出 level_min_limit1.level1_min_limit 寄存器里的值
15-14	WAVE0_ABOVE_MAX_LIMIT	wave0 缩放后大于满量程的部分的限幅模式。 0: 输出 0xffff. 1: 输出 0x0. 2: 输出 level_max_limit0.level0_max_limit 寄存器里的值
13	WAVE0_HIGH_AREA1_LIMIT	wave0 高区 1 的限幅模式. 0: 输出 0xffff. 1: 输出 level_max_limit0.level0_max_limit 寄存器里的值
12	WAVE0_HIGH_AREA0_LIMIT	wave0 高区 0 的限幅模式. 0: 输出 0xffff. 1: 输出 level_max_limit0.level0_max_limit 寄存器里的值
11	WAVE0_LOW_AREA1_LIMIT	wave0 低区 1 的限幅模式。 0: 输出 0. 1: 输出 level_min_limit0.level1_min_limit 寄存器里的值。
10	WAVE0_LOW_AREA0_LIMIT	wave0 低区 0 的限幅模式。 0: 输出 0. 1: 输出 level_min_limit0.level1_min_limit 寄存器里的值。

位域	名称	描述
9-8	WAVE0_BELOW_MIN_LIMIT	wave0 缩放后小于 0 的部分的限幅模式。 0: 输出 0. 1: 输出 0xffff. 2: 输出 level_min_limit0.level1_min_limit 寄存器里的值
7	SADDLE_TYPE	马鞍波类型： 0：标准马鞍波，仅当三相相差配置为 120° 弦波时适用。 1：三角波叠加型马鞍波，对三相相差没有任何要求。
6	EN_WAVE2_VD_VQ_INJECT	wave2 VdVq 注入使能。 0：不使能。 1：使能。
5	EN_WAVE1_VD_VQ_INJECT	wave1 VdVq 注入使能。 0：不使能。 1：使能。
4	EN_WAVE0_VD_VQ_INJECT	wave0 VdVq 注入使能。 0：不使能。 1：使能。
1-0	WAVES_OUTPUT_TYPE	弦波输出模式。 0：余弦波。 1：马鞍波。 2：馒头波。 3: 锯齿波。

WAVE[MODE] 位域

37.5.2 WAVE[RESOLUTION] (0x4)

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

WAVE[RESOLUTION] [31:0]

位域	名称	描述
31-0	LINES	wave0/1/2 对应的线数。

WAVE[RESOLUTION] 位域

37.5.3 WAVE[PHASE_SHIFT] (0x8 + 0x4 * m)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																VAL															
N/A																RW															
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

WAVE[PHASE_SHIFT] [31:0]

位域	名称	描述
15-0	VAL	wave0 相位偏移值，缺省是 0x0，写入其他非零值会使得该相位向左偏移（cfg_value/2^16）周期，比如若希望该相位提前 1/4 周期，或者说提前 90°，或者说提前 Pi/2, cfg_value 设置为 0x4000。若希望该相位推迟 1/4 周期，即提前 3/4 周期，即 cfg_value 设置为 0xC000。

WAVE[PHASE_SHIFT] 位域

37.5.4 WAVE[VD_VQ_INJECT] (0x14 + 0x4 * m)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VQ_VAL																VD_VAL															
RW																RW															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

WAVE[VD_VQ_INJECT] [31:0]

位域	名称	描述
31-16	VQ_VAL	Vq 注入值为有符号数
15-0	VD_VAL	Vd 注入值为有符号数

WAVE[VD_VQ_INJECT] 位域

37.5.5 WAVE[VD_VQ_LOAD] (0x20)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																															LOAD
N/A																															WO
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0

WAVE[VD_VQ_LOAD] [31:0]

位域	名称	描述
0	LOAD	加载 wave0/1/2 vd vq 注入值。该 bit 只能读到 0。 0. 即使 vq_vd_wave 寄存器的值发生变化，wave0/1/2 vd vq 注入值也保持不变。 1: 在同一时刻加载 wave0/1/2 vd vq 注入值。

WAVE[VD_VQ_LOAD] 位域

37.5.6 WAVE[AMPLITUDE] (0x24 + 0x4 * m)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD															EN_SCAL	AMP_VAL															
N/A															RW	RW															
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

WAVE[AMPLITUDE] [31:0]

位域	名称	描述
16	EN_SCAL	wave 幅值缩放使能，0：不使能；1：使能。
15-0	AMP_VAL	wave 幅值缩放倍数。bit15-12 为整数倍数，bit11-0 为小数部分。

WAVE[AMPLITUDE] 位域

37.5.7 WAVE[MID_POINT] (0x30 + 0x4 * m)

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

WAVE[MID_POINT] [31:0]

位域	名称	描述
31-0	VAL	wave0 输出中心点值偏移值，该 32bit 采用有符号数，bit31 为符号位，bit30-27 为整数部分，bit26-0 为小数部分。正数会让波形向上偏移，负数会让波形向下偏移。其 1 倍对应满量程的一半。

WAVE[MID_POINT] 位域

37.5.8 WAVE[LIMIT][MIN] (0x3C + 0x8 * m)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LIMIT1																LIMIT0															
RW																RW															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

WAVE[LIMIT][MIN] [31:0]

位域	名称	描述
31-16	LIMIT1	低区限幅 level1
15-0	LIMIT0	低区限幅 level0

WAVE[LIMIT][MIN] 位域

37.5.9 WAVE[LIMIT][MAX] (0x40 + 0x8 * m)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LIMIT1																LIMIT0															
RW																RW															
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

WAVE[LIMIT][MAX] [31:0]

位域	名称	描述
31-16	LIMIT1	高区限幅 level1
15-0	LIMIT0	高区限幅 level0

WAVE[LIMIT][MAX] 位域

37.5.10 WAVE[DEADZONE_SHIFT] (0x54 + 0x4 * m)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																VAL															
N/A																RW															
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

WAVE[DEADZONE_SHIFT] [31:0]

位域	名称	描述
15-0	VAL	wave0 死区偏移值为有符号数

WAVE[DEADZONE_SHIFT] 位域

37.5.11 ABZ[MODE] (0x60)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD			REVERSE_EDGE_TYPE	RSVD			EN_WDOG	RSVD			Z_POLARITY	RSVD			B_POLARITY	RSVD			A_POLARITY	RSVD		Z_TYPE		RSVD		B_TYPE		RSVD		A_TYPE	
N/A			RW	N/A			RW	N/A			RW	N/A			RW	N/A			RW	N/A		RW		N/A		RW		N/A		RW	
x	x	x	0	x	x	x	0	x	x	x	0	x	x	x	0	x	x	x	0	x	x	0	0	x	x	0	0	x	x	0	0

ABZ[MODE] [31:0]

位域	名称	描述
28	REVERSE_EDGE_TYPE	高速脉冲计数的方向指示信号的切换方向的时间点选择。 0：指示信号切换方向发生在脉冲时钟的下降沿和上升沿的中间点，距离时钟信号的上升或下降沿间隔大于等于 line_width 给出的系统时钟周期个数。 1：指示信号切换方向发生在脉冲时钟的下降沿。
24	EN_WDOG	使能 abz 看门狗功能。 0：不使能。 1：使能。
20	Z_POLARITY	wave_z 输出极性翻转。 0：不翻转，正常输出。 1：翻转正在输出。
16	B_POLARITY	wave_b 输出极性翻转。 0：不翻转，正常输出。 1：翻转正在输出。
12	A_POLARITY	wave_a 输出极性翻转。 0：不翻转，正常输出。 1：翻转正在输出。
9-8	Z_TYPE	wave_z 输出类型： 0：做为零点脉冲输出，在 wave_a 和 wave_b 同时为高的时候为高，高电平持续 25% 的周期。 1：做为零点脉冲输出，高电平持续 75% 的周期。 2：做为零点脉冲输出，高电平持续 100% 的周期。 3：做为第三相输出。
5-4	B_TYPE	wave_b 输出类型： 0：两相正交输出 wave_b。 1：高速脉冲计数方向指示电平。 2：高速脉冲计数反方向计数脉冲。 3：三相输出 wave_b。

位域	名称	描述
1-0	A_TYPE	wave_b 输出类型： 0：两相正交输出 wave_a。 1：高速脉冲计数时钟脉冲。 2：高速脉冲计数正方向计数脉冲。 3：三相输出 wave_a。

ABZ[MODE] 位域

37.5.12 ABZ[RESOLUTION] (0x64)

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

ABZ[RESOLUTION] [31:0]

位域	名称	描述
31-0	LINES	wave_a/b/z 对应的线数。

ABZ[RESOLUTION] 位域

37.5.13 ABZ[PHASE_SHIFT] (0x68 + 0x4 * m)

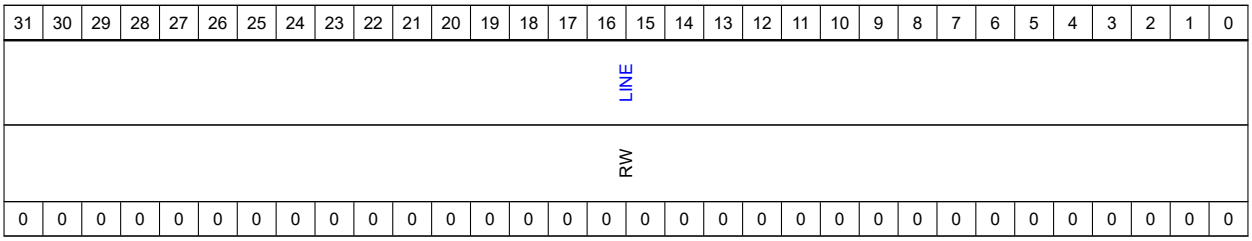
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																VAL															
N/A																RW															
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ABZ[PHASE_SHIFT] [31:0]

位域	名称	描述
15-0	VAL	wave_a 相位偏移值，缺省是 0x0，写入其他非零值会使得该相位向左偏移（cfg_value/2^16）周期，比如若希望该相位提前 1/4 周期，或者说提前 90°，或者说提前 Pi/2, cfg_value 设置为 0x4000。若希望该相位推迟 1/4 周期，即提前 3/4 周期，即 cfg_value 设置为 0xC000。

ABZ[PHASE_SHIFT] 位域

37.5.14 ABZ[LINE_WIDTH] (0x74)

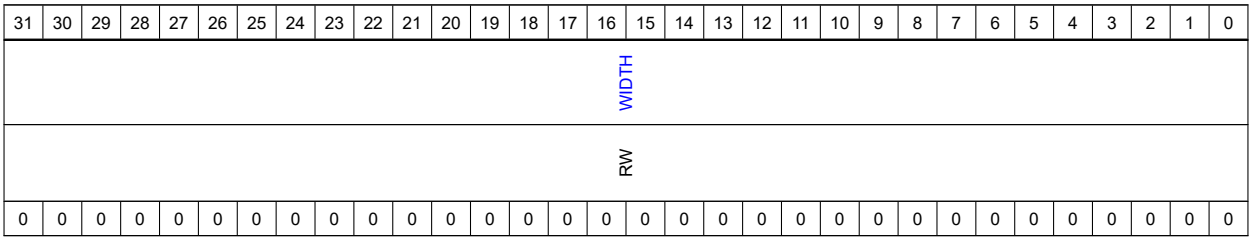


ABZ[LINE_WIDTH] [31:0]

位域	名称	描述
31-0	LINE	wave_a 输出两相正交波形时，1/4 周期对应的系统时钟个数。

ABZ[LINE_WIDTH] 位域

37.5.15 ABZ[WDOG_WIDTH] (0x78)

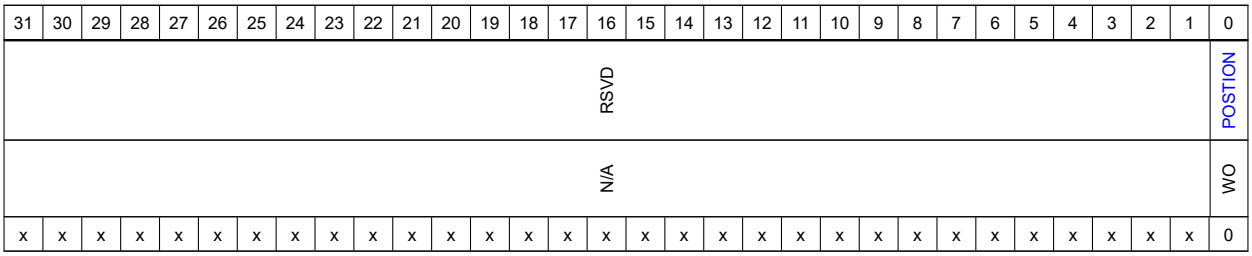


ABZ[WDOG_WIDTH] [31:0]

位域	名称	描述
31-0	WIDTH	两相正交输出时，当无有效位置输入的时间超过 wdog_width 对应的系统时钟长度后，会步进或步退 1/4 线，已提示外界系统输出仍在掌控中，消除死机顾虑。

ABZ[WDOG_WIDTH] 位域

37.5.16 ABZ[POSTION_SYNC] (0x7C)



ABZ[POSTION_SYNC] [31:0]

位域	名称	描述
0	POSTION	是否将下一笔有效的位置直接加载到 abz 内部位置管理器。只同步线数的整数部分。读该 bit 永远是 0. 1. 同步。 2. 不同步。

ABZ[POSTION_SYNC] 位域

37.5.17 PWM[MODE] (0x80)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PWM7_SAFETY		PWM6_SAFETY		PWM5_SAFETY		PWM4_SAFETY		PWM3_SAFETY		PWM2_SAFETY		PWM1_SAFETY		PWM0_SAFETY		RSVD						PWM_ENTER_SAFETY_MODE	PWM_SAFETY_BYPASS	RSVD			REVISE_UP_DN	PHASE_NUM			
RW		RW		RW		RW		RW		RW		RW		RW		N/A						RW	RW	N/A			RW	RW			
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	x	x	x	x	x	x	0	0	x	x	x	0	0	0	0	0

PWM[MODE] [31:0]

位域	名称	描述
31-30	PWM7_SAFETY	PWM 输出通道安全保护控制位 00：PWM 正常输出 01：PWM 正常输出 10：强制输出 0 11：强制输出 0
29-28	PWM6_SAFETY	PWM 输出通道安全保护控制位 00：PWM 正常输出 01：PWM 正常输出 10：强制输出 0 11：强制输出 0
27-26	PWM5_SAFETY	PWM 输出通道安全保护控制位 00：PWM 正常输出 01：PWM 正常输出 10：强制输出 0 11：强制输出 0

位域	名称	描述
25-24	PWM4_SAFETY	PWM 输出通道安全保护控制位 00：PWM 正常输出 01：PWM 正常输出 10：强制输出 0 11：强制输出 0
23-22	PWM3_SAFETY	PWM 输出通道安全保护控制位 00：PWM 正常输出 01：PWM 正常输出 10：强制输出 0 11：强制输出 0
21-20	PWM2_SAFETY	PWM 输出通道安全保护控制位 00：PWM 正常输出 01：PWM 正常输出 10：强制输出 0 11：强制输出 0
19-18	PWM1_SAFETY	PWM 输出通道安全保护控制位 00：PWM 正常输出 01：PWM 正常输出 10：强制输出 0 11：强制输出 0
17-16	PWM0_SAFETY	PWM 输出通道安全保护控制位 00：PWM 正常输出 01：PWM 正常输出 10：强制输出 0 11：强制输出 0
9	PWM_ENTER_SAFETY_MODE	是否进入 PWM 保护模式。 0：不进入。 1：进入。
8	PWM_SAFETY_BYPASS	是否屏蔽 trigger in 触发的 PWM safety 模式。 0：不屏蔽。 1：屏蔽。
4	REVISE_UP_DN	是否采用反向查找表输出。 0：正向。 1：反向。
3-0	PHASE_NUM	pwm force 的换相拍数。

PWM[MODE] 位域

37.5.18 PWM[RESOLUTION] (0x84)

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

PWM[RESOLUTION] [31:0]

位域	名称	描述
31-0	LINES	pwm force 输出的线数

PWM[RESOLUTION] 位域

37.5.19 PWM[PHASE_SHIFT] (0x88 + 0x4 * m)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																VAL															
N/A																RW															
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

PWM[PHASE_SHIFT] [31:0]

位域	名称	描述
15-0	VAL	pwm_a 相位偏移值，缺省是 0x0，写入其他非零值会使得该相位向左偏移（cfg_value/2^16）周期，比如若希望该相位提前 1/4 周期，或者说提前 90°，或者说提前 Pi/2, cfg_value 设置为 0x4000。若希望该相位推迟 1/4 周期，即提前 3/4 周期，即 cfg_value 设置为 0xC000。

PWM[PHASE_SHIFT] 位域

37.5.20 PWM[PHASE_TABLE] (0x98 + 0x4 * m)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																PWM7		PWM6		PWM5		PWM4		PWM3		PWM2		PWM1		PWM0	
N/A																RW		RW		RW		RW		RW		RW		RW		RW	
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

PWM[PHASE_TABLE] [31:0]

位域	名称	描述
15-14	PWM7	PWM 输出通道的强制输出模式控制位 00：PWM 正常输出 01：PWM 正常输出 10：强制输出 0 11：强制输出 0
13-12	PWM6	PWM 输出通道的强制输出模式控制位 00：PWM 正常输出 01：PWM 正常输出 10：强制输出 0 11：强制输出 0
11-10	PWM5	PWM 输出通道的强制输出模式控制位 00：PWM 正常输出 01：PWM 正常输出 10：强制输出 0 11：强制输出 0
9-8	PWM4	PWM 输出通道的强制输出模式控制位 00：PWM 正常输出 01：PWM 正常输出 10：强制输出 0 11：强制输出 0
7-6	PWM3	PWM 输出通道的强制输出模式控制位 00：PWM 正常输出 01：PWM 正常输出 10：强制输出 0 11：强制输出 0
5-4	PWM2	PWM 输出通道的强制输出模式控制位 00：PWM 正常输出 01：PWM 正常输出 10：强制输出 0 11：强制输出 0
3-2	PWM1	PWM 输出通道的强制输出模式控制位 00：PWM 正常输出 01：PWM 正常输出 10：强制输出 0 11：强制输出 0
1-0	PWM0	PWM 输出通道的强制输出模式控制位 00：PWM 正常输出 01：PWM 正常输出 10：强制输出 0 11：强制输出 0

PWM[PHASE_TABLE] 位域

37.5.21 POSTION_SOFTWARE (0xF8)

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

POSTION_SOFTWARE [31:0]

位域	名称	描述
31-0	POSTION_SOFTWAVE	软件注入的位置

POSTION_SOFTWARE 位域

37.5.22 POSTION_SEL (0xFC)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																															POSTION_SEL
N/A																															RW
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0

POSTION_SEL [31:0]

位域	名称	描述
0	POSTION_SEL	使能软件注入的位置。 0：不使能。 1：使能。

POSTION_SEL 位域

37.5.23 STATUS (0x100)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PWM_FOURCE																RSVD															PWM_SAFETY
RO																N/A															RO
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0

STATUS [31:0]

位域	名称	描述
31-16	PWM_FOURCE	qeo_pwm_force[15:0] 输出观察
0	PWM_SAFETY	PWM ctrl 是否进入 fault 模式

STATUS 位域

37.5.24 DEBUG0 (0x104)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WAVE1																WAVE0															
RO																RO															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

DEBUG0 [31:0]

位域	名称	描述
31-16	WAVE1	wave1 输出观察
15-0	WAVE0	wave0 输出观察

DEBUG0 位域

37.5.25 DEBUG1 (0x108)

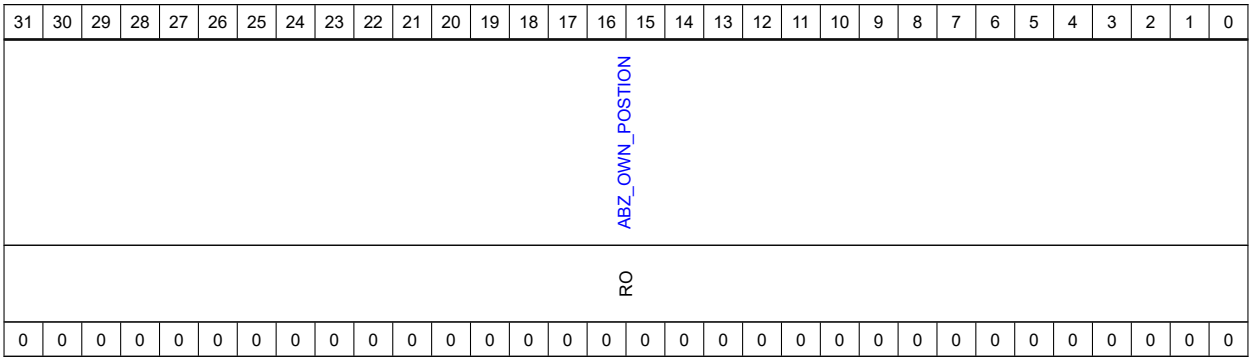
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD			QEO_FINISH	RSVD			WAVE_Z	RSVD			WAVE_B	RSVD			WAVE_A	WAVE2															
N/A			RO	N/A			RO	N/A			RO	N/A			RO	RO															
x	x	x	0	x	x	x	0	x	x	x	0	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

DEBUG1 [31:0]

位域	名称	描述
28	QEO_FINISH	qeo_finish, 为高时代表 qeo 已完成当前位置的计算。
24	WAVE_Z	wave_z 输出观察
20	WAVE_B	wave_b 输出观察
16	WAVE_A	wave_a 输出观察
15-0	WAVE2	wave2 输出观察

DEBUG1 位域

37.5.26 DEBUG2 (0x10C)

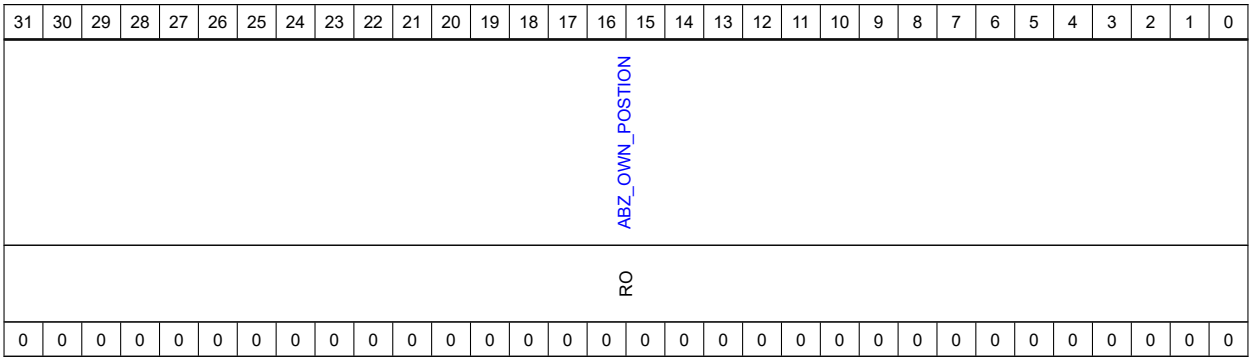


DEBUG2 [31:0]

位域	名称	描述
31-0	ABZ_OWN_POSTION	abz 内部位置观察，bit31:2 为当前整数线，bit1：0 为小数部分。

DEBUG2 位域

37.5.27 DEBUG3 (0x110)



DEBUG3 [31:0]

位域	名称	描述
31-0	ABZ_OWN_POSTION	abz 内部位置观察，只观察当前整线数

位域	名称	描述

DEBUG3 位域

38 运动处理单元 MMC

38.1 概述

本章节介绍 MMC 的主要功能和特性。

MMC 又称运动处理单元。它接收来自其他模块的转子位置和时间戳，然后通过内在的算法来预测并补偿在未来时刻的转子位置并输出，同时给出速度和加速度估计。目前支持的特性如下表 179：

功能	主要特性
初始化时	· 配置好初始位置、速度、加速度。 · 配置好初始位置、速度、加速度的增量。
已经运行时	· 动态配置初始位置、速度、加速度。 · 动态配置初始位置、速度、加速度的增量。
开环模式	· 给定初始位置、速度、加速度，然后随着时间不断演进。这时只有预测模块在工作。
触发模式	· 可配置位置、时间、和外部事件做触发。可以支持动态改变初始位置、速度、加速度，以及动态改变位置、速度、加速度的增量。目前外部事件仅做上升沿检测。
中断	· 支持配置完成中断、触发中断和异常中断。

表 179: MMC 主要特性

38.2 MMC 架构图

本模块的结构图如图 41。其中，圈内位置和圈数，统称位置。

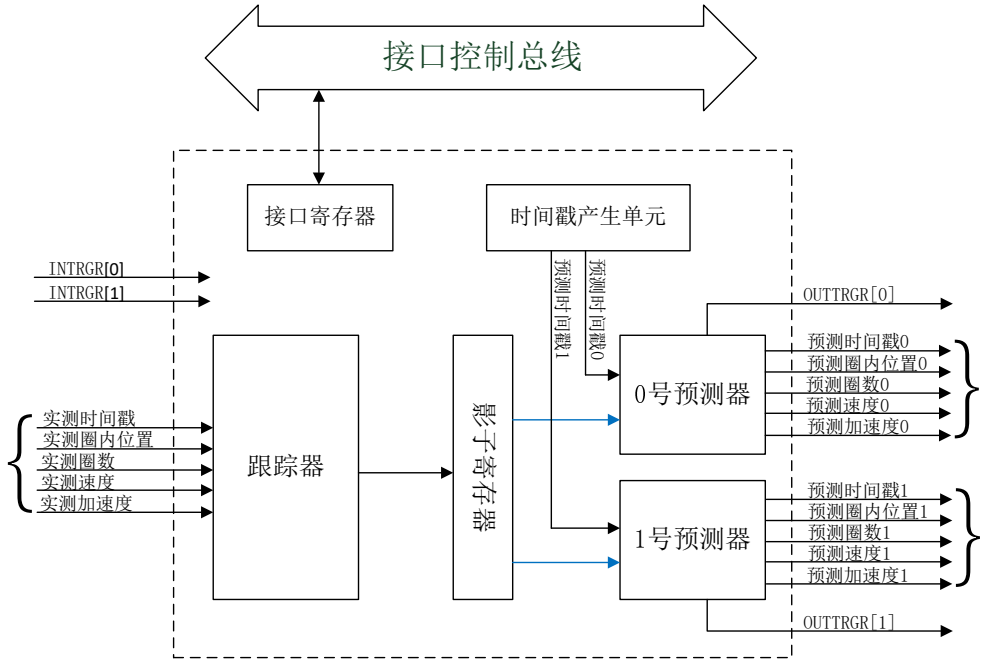


图 41: MMC 结构框图

38.3 功能说明

38.3.1 MMC 一般说明

其主要数据流图是：实测数据送入到跟踪器里，训练跟踪器里的相关参数。每来一次实测数据，训练一次。每次训练完后，这些训练出来的参数被送至影子寄存器里，后面 2 个预测器会根据各自预测的时间戳，以及最新的影子寄存器里的内容，来给出预测时刻的位置、速度、加速度等信息。

38.3.2 预测器的预测时间戳的产生

每个预测器都有独立的预测时间戳的产生方式。下面列出了可能的几种选择。图中，

- 黑色时间 t_n 是要预测的时间，
- 红色时间 t'_n 是运算开始的时间，
- 黄色时间 t_{tn} 是触发信号到达的时间，
- 蓝色曲线箭头是由寄存器 BR[n][BR_TRG_PERIOD] 控制，
- 红色曲线箭头是由寄存器 BR[n][BR_TIMEOFF] 控制，
- 黑色箭头是由寄存器 BR[n][BR_TRG_F_TIME] 控制，
- 黄色箭头是触发信号到达时间，
- 蓝色小框是 MMC 计算所要花费的时间

当 BR [n] [BR_CTRL [PRED_MODE]] = 0 时，且 BR [n] [BR_CTRL [F_TRG_TYPE]] = 0, BR [n] [BR_CTRL [NF_TRG_TYPE]] = 0, 会产生周期性的预测。其时间戳产生模式如图 42。这幅图里表示的是：第一个要预测的时间 t_0 是配置在 BR [n] [BR_TRG_F_TIME] 里的，但这个预测数据是在 t'_0 时间开始算的（其中，t'_0=t_0-BR [n] [BR_TIMEOFF]）；下一个要预测的时间是 t_1 = t_0+BR [n] [BR_TRG_PERIOD]；后面的计算以此类推。

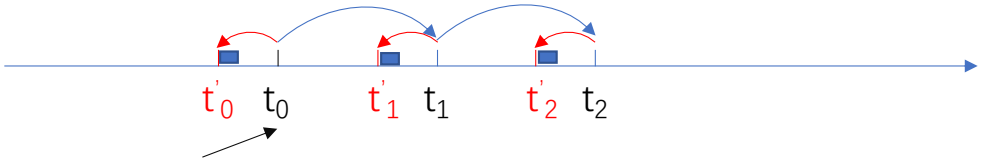


图 42: MMC 时间戳产生方式 0

当 BR [n] [BR_CTRL [PRED_MODE]] = 0 时，且 BR [n] [BR_CTRL [F_TRG_TYPE]] = 1, BR [n] [BR_CTRL [NF_TRG_TYPE]]=1, 会产生依据触发信号的预测。其时间戳产生模式如图 43。这幅图里表示的是：第一个要预测的时间 t_1 是触发时间 t_{t1}+BR [n] [BR_TRG_PERIOD]，但这个预测数据是在 t'_1 时间开始算的（其中，t'_1=t_1-BR [n] [BR_TIMEOFF]）；下一个要预测的时间是 t_2 由触发时间 t_{t2} 决定；后面的计算以此类推。

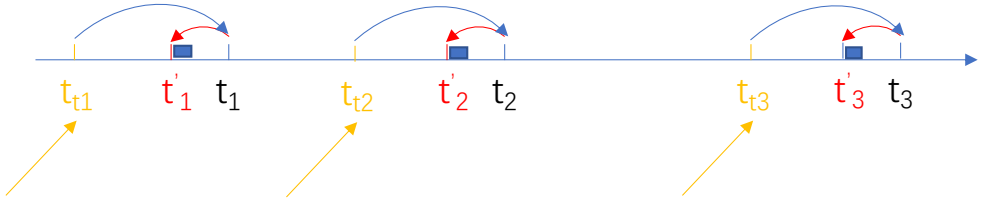


图 43: MMC 时间戳产生方式 1

当 BR [n] [BR_CTRL [PRED_MODE]] = 1 时，会产生连续不断的预测。

- 当 BR [n] [BR_CTRL [F_TRG_TYPE]] = 0 时, 第一个要预测的时间 t_0 是配置在 BR [n] [BR _TRG _F _TIME] 里的，但这个预测数据是在 t'_0 时间开始算的（其中，t'_0=t_0-BR [n] [BR_TIMEOFF]）；下一个要预测的时间是 t_1= 计算结束后的时间 +BR [n] [BR_TRG_PERIOD]；后面的计算以此类推。

- 当 BR [n] [BR_CTRL [F_TRG_TYPE]] = 1 时，第一个要预测的时间 t_1 是触发时间 t_{t1} + BR [n] [BR _TRG _PERIOD]，但这个预测数据是在 t_1' 时间开始算的（其中，$t_1'=t_1$-BR [n] [BR_TIMEOFF]）；下一个要预测的时间是 t_2= 计算结束后的时间 +BR [n] [BR_TRG_PERIOD]；后面的计算以此类推。

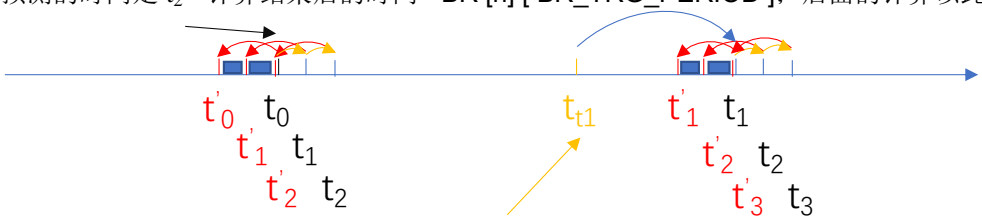


图 44: MMC 时间戳产生方式 2

上面展示的都是有多次预测的模式。当 BR [n] [BR_CTRL [PRED_MODE]] = 2 时，除了只会产生一次预测外，别的方面类似。

38.3.3 MMC 数据流图

其具体数据流图如图 45。

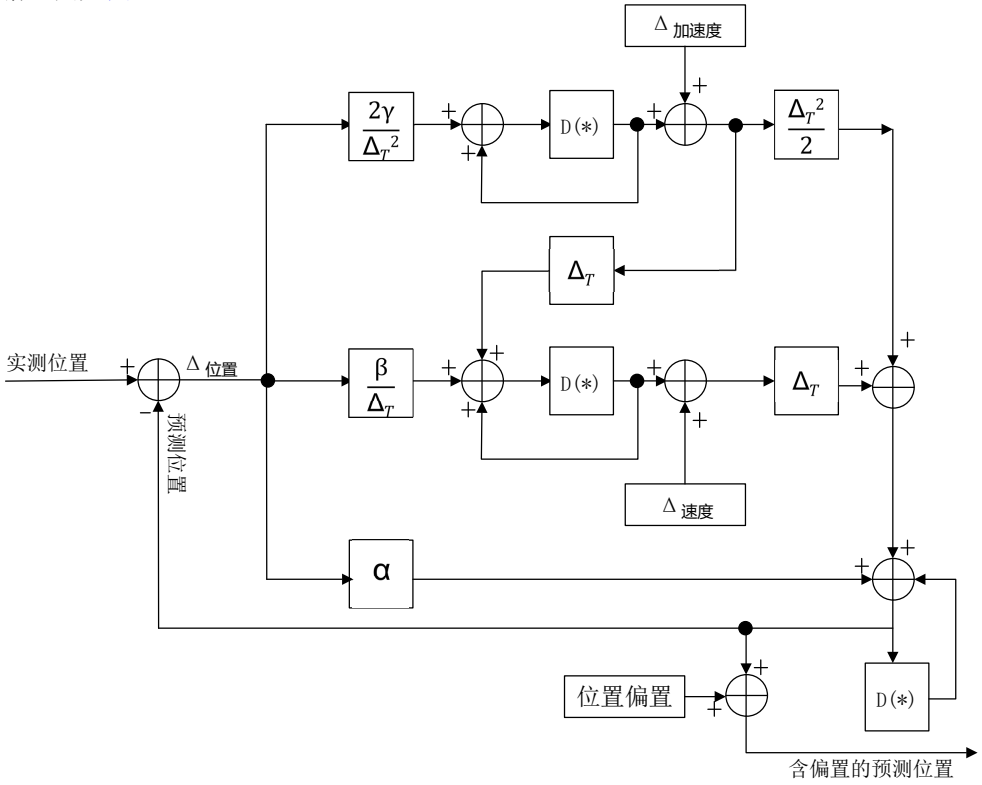


图 45: MMC 结构框图

这幅图是跟踪器的图，输入为“实测位置”和“Δ_T”；预测器和跟踪器比较类似，只是预测器则只需要输入“Δ_T”，然后算出“预测位置”和“含偏置的预测位置”，别的相关参数都从跟踪器通过影子寄存器拷贝而来。其中，α 又被称为 P 系数；β 又被称为 I 系数；$\gamma/2$ 又被称为 A 系数。

38.3.4 MMC 预测输出

注意：本节给出了预测输出的配置及其影响。为明显区别各个配置的不同，图中的效果仅是示意图，是有所夸张的，是来帮助理解的，并不是真实的数据。

预测输出，可以通过如下两种方式来产生：

38.3.4.1 当 CR[ADJOP]=0 时的预测输出

当 CR[ADJOP]=0 时的 MMC 预测效果举例如图 46。图中，横轴代表时间，纵轴代表位置，红线代表真实的位置，黄线代表预测出的位置。横轴上的标记 Tn 代表观测到的实测位置的时间戳，可见时间戳不是均匀的。

跟踪和预测都是连续的；每次当一个新的实测位置到来时，会改变跟踪到的参数，同时会影响预测值向实测位置靠拢。

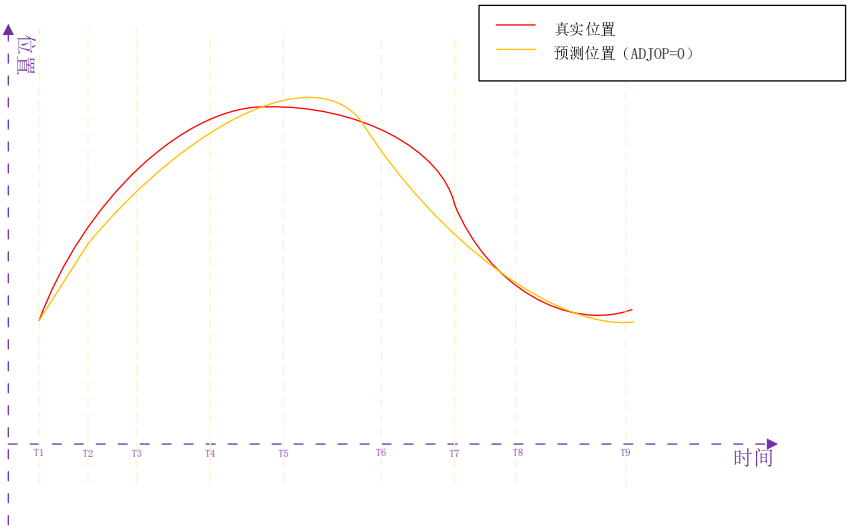


图 46: MMC 预测波形，当 CR[ADJOP]=0

38.3.4.2 当 CR[ADJOP]=1 时的预测输出

当 CR[ADJOP]=1 时的 MMC 预测效果举例如图 47。图中，黑线代表预测出的位置。

这时，每次一个新的实测位置到来后，在改变跟踪到的参数的同时，会影响预测值从实测位置开始，利用跟踪到的参数进行预测。

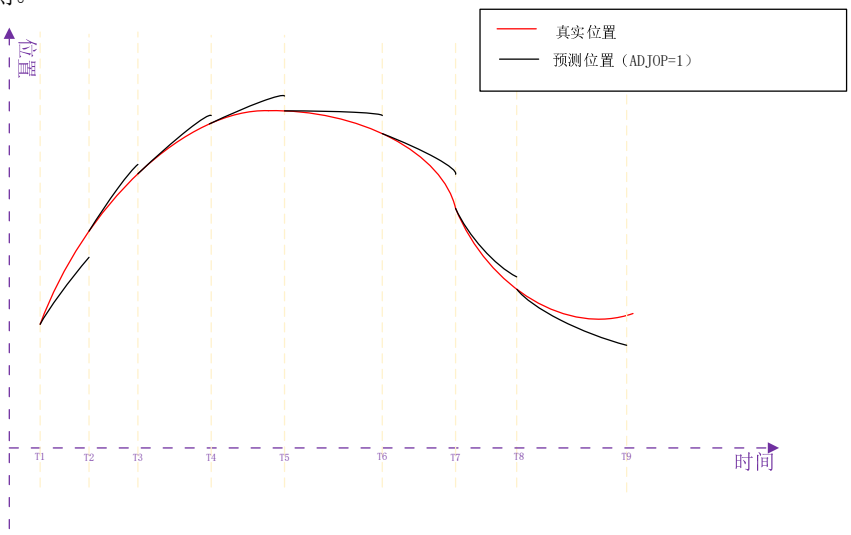


图 47: MMC 预测波形，当 CR[ADJOP]=1

当 CR[ADJOP]=1 时，还有一个失步再调整的机制。失步门限需要配置寄存器 OOSYNC_THETA_THR 到一

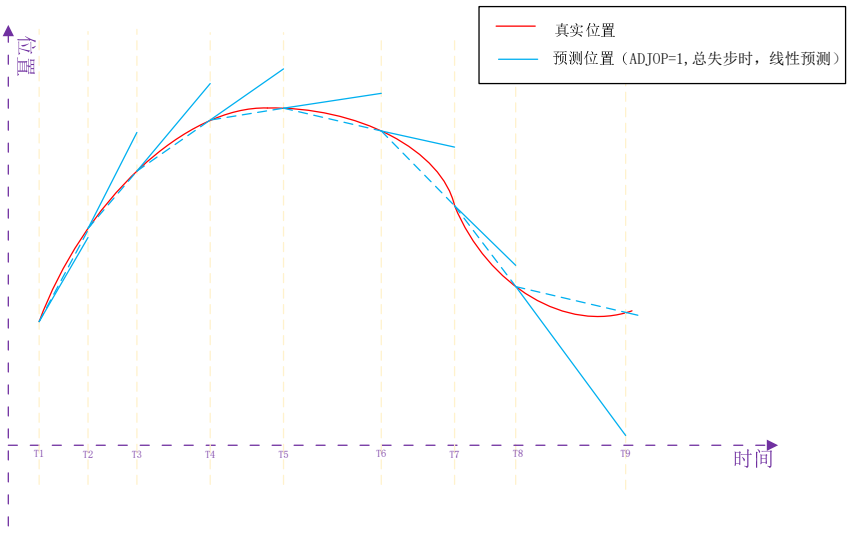


图 48: MMC 预测波形，当 CR[ADJOP]=1，且总是失步时的预测结果

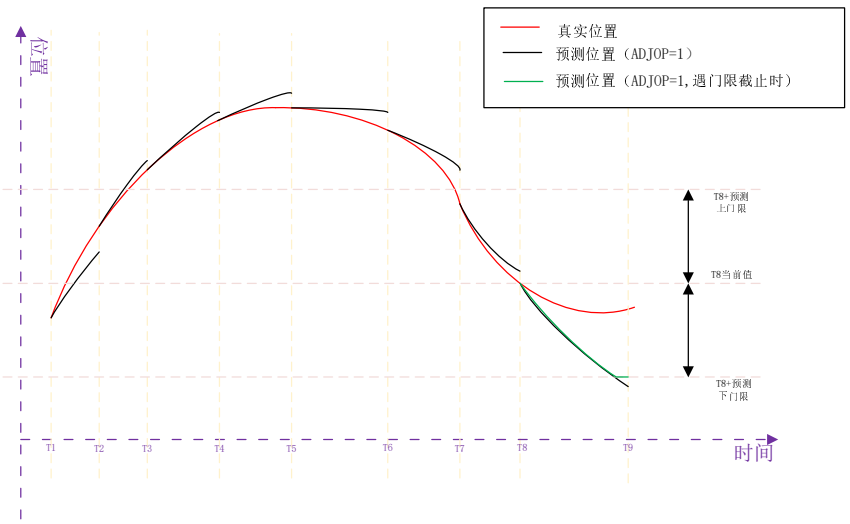


图 49: MMC 预测波形，当 CR[ADJOP]=1，且 T8 时刻后遇到门限截止时的预测结果

个恰当的数值。当图 45里的 $\triangle_{位置}$ 的绝对值大于 OOSYNC_THETA_THR 时，认为当前跟踪到的数据已经离实际值比较大，就会强制对跟踪到的系数进行重新初始化，即：把预估的加速度清为 0，预估的速度设为

$$\frac{(新实测位置\text{-}上一次的实测位置)}{(新实测时间\text{-}上一次的实测时间)}$$

。此时 MMC 预测效果举例如图 48。图中，蓝实线代表预测出的位置，蓝虚线代表估计速度的依据，可见蓝实线其实是蓝虚线的延长。

当 CR[ADJOP]=1 时，还有一个上下门限的机制。上下门限需要配置寄存器 DISCRETECFG1 [INV_POSMAX] 到一个恰当的数值。当图 45里的（预测位置-上一次的实测位置）的绝对值大于 DISCRETECFG1 [INV_POSMAX] 时，认为当前预测的数据已经离实际值比较大，就会强制对预测到的数据锁死在这个门限上。此时 MMC 预测效果举例如图 49。图中，仅标出了在时间 T8 之后出现这种触及门限的情况。如果有门限的限制，预测值按照黑线的方式进行；如果没有门限的限制，预测值按照绿线的方式进行。图中为清楚起见，将黑线和绿线分开画了，其实，它们在 T8 时刻过后，一开始是相等的，然后分叉了。

38.3.5 开环和闭环

开环模式，指的是，在某个特定速度，和加速度的情况下，从某个特定的位置开始，不断随着时间的演进，从而不断地给出新的预测值。开环模式仅需要预测模块就可以实现。

闭环模式，指的是，根据输入的时间和位置，本模块去估计速度和加速度，然后预测，在比较短的时间里，电机到达的新位置，究竟是多少。这个新位置可以用来产生更精确的电机驱动波形。闭环模式需要跟踪模块和预测模块同时工作。

具体配置方法，见小节 38.6.6。

38.3.5.1 从开环切换到闭环

这是用来模拟和驱动电机从静止到运动的一种方法。可以通过设置一定的加速度的方式，产生一系列的假设位置，让后面的电路模块按照这些位置，去强行拖动电机；同时跟踪模块也在不断地评估现在的位置、速度、和加速度。

一般来说，当跟踪模块跟踪到一定的速度之后，就可以退出开环模式，进入到闭环模式，用跟踪到的位置、速度、和加速度等信息来驱动电机了。

38.3.6 初始化位置，速度，加速度

如果在 MMC 模块运行前，可以知道相关的位置，速度和加速度，则可以事先将其配上，减少收敛时间。具体配置方法，见小节 38.6.2。

38.3.7 初始化位置、速度、加速度的增量 (即：位置偏置，$\triangle_{速度}$，$\triangle_{加速度}$)

由于跟踪到参数收敛总是需要一定的时间，这在某些情况下，可能希望更快的收敛速度。而在实际驱动的时候，总是会知道是想加速还是减速，甚至某些时候，还可以预测到速度和/或加速度的增量的值。这时，就可以根据情况，预先将速度和加速度的增量配置上，这样，会让系统收敛的更快。所以，速度、加速度的增量一般配置在跟踪器里。

有时候电机的电角度和物理角度之间存在一个小的固定偏差，这个时候，就可以将位置增量配置成这个偏差，从而输出更准确的（带偏置的）位置。所以，位置配置一般配置在预测器里。

具体配置方法，见小节 38.6.3。

38.3.8 触发机制

触发机制主要用来设定什么时候来设定位置、速度、加速度、位置增量、速度增量、加速度增量等。可以按照时间、位置、速度、外来触发信号这几种方式来触发。

具体配置方法，见小节 38.6.5。

38.3.9 内部数据监控

有如下寄存器，可以用来监控内部的一些信息。

- 跟踪器的状态
  - 估计数据中的时间戳显示在 ESTM_TIM 寄存器里。
    为读取这个数据，需要先设CR[SHADOW_RD_REQ]=1，然后等CR[SHADOW_RD_REQ]=0后可读取。
  - 估计数据中的位置显示在 ESTM_POS、ESTM_REV 寄存器里。
    为读取这个数据，需要先设CR[SHADOW_RD_REQ]=1，然后等CR[SHADOW_RD_REQ]=0后可读取。
  - 估计数据中的速度显示在 ESTM_SPEED 寄存器里。
    为读取这个数据，需要先设CR[SHADOW_RD_REQ]=1，然后等CR[SHADOW_RD_REQ]=0后可读取。
  - 估计数据中的加速度显示在 ESTM_ACCEL 寄存器里。
    为读取这个数据，需要先设CR[SHADOW_RD_REQ]=1，然后等CR[SHADOW_RD_REQ]=0后可读取。
  - 估计数据中的加速度显示在 CUR_PCOEF 寄存器里。
    为读取这个数据，需要先设CR[SHADOW_RD_REQ]=1，然后等CR[SHADOW_RD_REQ]=0后可读取。
  - 估计数据中的加速度显示在 CUR_ICOEF 寄存器里。
    为读取这个数据，需要先设CR[SHADOW_RD_REQ]=1，然后等CR[SHADOW_RD_REQ]=0后可读取。
  - 估计数据中的加速度显示在 CUR_ACOEF 寄存器里。
    为读取这个数据，需要先设CR[SHADOW_RD_REQ]=1，然后等CR[SHADOW_RD_REQ]=0后可读取。
  - 跟踪器最新的输入实时地显示在 BK0_TIMESTAMP、BK0_POSITION、BK0_REVOLUTION、BK0_SPEED、BK0_ACCELERATOR。
  - 跟踪器次新的输入实时地显示在 BK1_TIMESTAMP、BK1_POSITION、BK1_REVOLUTION、BK1_SPEED、BK1_ACCELERATOR。
- 预测器的状态
  - 从跟踪模块来的时间戳显示在 BR[n][BR_CUR_POS_TIME] 寄存器里。直接读取，无影子寄存器。
  - 从跟踪模块来的位置显示在 BR[n][BR_CUR_POS]、BR[n][BR_CUR_REV] 寄存器里。直接读取，无影子寄存器。
  - 从跟踪模块来的速度显示在 BR[n][BR_CUR_SPEED] 寄存器里。直接读取，无影子寄存器。
  - 从跟踪模块来的加速度显示在 BR[n][BR_CUR_ACCEL] 寄存器里。直接读取，无影子寄存器。

38.3.10 单重及多重 P、I、A 参数

闭环时，环路的收敛速度取决于 P、I、A 参数。过小的参数会导致收敛过慢，乃至于频频失步；过大的参数则在很快收敛的同时，可能会有更大的噪声和过冲。单重参数具体配置方法，见小节 38.6.4。如果其参数值增加，其影响效果一般如下表 180。

表 180: P、I、A 参数的影响

响应	上升时间	过冲	稳定时间	稳态误差
P	下降	上升	不明确	下降
I	下降	上升	上升	消除
A	下降	上升	不明确	不明确

因为很难将这些参数配置成万用参数，所以依据不同收敛情况来自动选择不同的系数会更好。所以本模块也支持多重 P、I、A 参数配置，具体见小节 38.6.7。

38.3.11 中断

本模块支持配置完成中断、触发中断和异常中断。相关中断源的具体说明如表 181所示。

中断事件	事件标志	使能控制位
跟踪器失步	STA[OOSYNC]	INT_EN[OOSYNC_IE]
跟踪器速度触发	STA[SPEED_TRG_VALID]	INT_EN[SPEED_TRG_VLD_IE]
跟踪器位置触发	STA[POS_TRG_VALID]	INT_EN[POS_TRG_VLD_IE]
跟踪器系数初始化命令结束	STA[INI_COEFS_CMD_DONE]	INT_EN[INI_COEFS_CMD_DONE_IE]
跟踪器位置增量初始化命令结束	STA[INI_DELTA_POS_REQ_CMD_DONE]	INT_EN[INI_DELTA_POS_REQ_CMD_DONE_IE]
跟踪器位置初始化命令结束	STA[INI_POS_REQ_CMD_DONE]	INT_EN[INI_POS_REQ_CMD_DONE_IE]
0 号预测器速度触发	BR[0][BR_ST[SPEED_TRG_VLD]]	BR[0][BR_CTRL[SPEED_TRG_VLD_IE]]
0 号预测器位置触发	BR[0][BR_ST[POS_TRG_VLD]]	BR[0][BR_CTRL[POS_TRG_VLD_IE]]
0 号预测器位置增量初始化命令结束	BR[0][BR_ST[INI_DELTA_POS_DONE]]	BR[0][BR_CTRL[INI_DELTA_POS_DONE_IE]]
0 号预测器位置初始化命令结束	STA[INI_BR0_POS_REQ_CMD_DONE]	INT_EN[INI_BR0_POS_REQ_CMD_DONE_IE]
1 号预测器速度触发	BR[1][BR_ST[SPEED_TRG_VLD]]	BR[1][BR_CTRL[SPEED_TRG_VLD_IE]]
1 号预测器位置触发	BR[1][BR_ST[POS_TRG_VLD]]	BR[1][BR_CTRL[POS_TRG_VLD_IE]]
1 号预测器位置增量初始化命令结束	BR[1][BR_ST[INI_DELTA_POS_DONE]]	BR[1][BR_CTRL[INI_DELTA_POS_DONE_IE]]

中断事件	事件标志	使能控制位
1 号预测器位置初始化命令结束	STA[INI_BR1_POS_REQ_CMD_DONE]	INT_EN[INI_BR1_POS_REQ_CMD_DONE_IE]

表 181: MMC 中断请求

38.4 MMC 寄存器列表

MMC 的寄存器列表如下：

MMC0 base address: 0xF0310000

MMC1 base address: 0xF0314000

地址偏移	名称	描述	复位值
0x0000	CR	控制寄存器	0x00000000
0x0004	STA	状态寄存器	0x00000020
0x0008	INT_EN	中断使能寄存器	0x00000000
0x000C	SYSCLK_FREQ	系统时钟频率寄存器	0x00000000
0x0010	SYSCLK_PERIOD	系统时钟周期寄存器	0x00000000
0x0014	OOSYNC_THETA_THR	位置失步门限寄存器	0x00000000
0x0018	DISCRETECFG0	离散模式 0 号配置寄存器	0x00000000
0x001C	DISCRETECFG1	离散模式 1 号配置寄存器	0x00000000
0x0020	CONTCFG0	连续模式 0 号配置寄存器	0x00000000
0x0024	INI_POS_TIME	位置初始化的时间戳寄存器	0x00000000
0x0028	INI_POS	位置初始化的位置寄存器	0x00000000
0x002C	INI_REV	位置初始化的转数寄存器	0x00000000
0x0030	INI_SPEED	位置初始化的速度寄存器	0x00000000
0x0034	INI_ACCEL	位置初始化的加速度寄存器	0x00000000
0x0038	INI_COEF_TIME	系数初始化的时间戳寄存器	0x00000000
0x003C	INI_PCOEF	系数初始化的 P 系数寄存器	0x00000000
0x0040	INI_ICOEF	系数初始化的 I 系数寄存器	0x00000000
0x0044	INI_ACOEF	系数初始化的 A 系数寄存器	0x00000000
0x0048	ESTM_TIM	内部估计的时间戳寄存器	0x00000000
0x004C	ESTM_POS	内部估计的位置寄存器	0x00000000
0x0050	ESTM_REV	内部估计的转数寄存器	0x00000000
0x0054	ESTM_SPEED	内部估计的速度寄存器	0x00000000
0x0058	ESTM_ACCEL	内部估计的加速度寄存器	0x00000000
0x005C	CUR_PCOEF	内部估计的 P 系数寄存器	0x00000000
0x0060	CUR_ICOEF	内部估计的 I 系数寄存器	0x00000000
0x0064	CUR_ACOEF	内部估计的 A 系数寄存器	0x00000000
0x0068	INI_DELTA_POS_TIME	增量位置初始化的时间戳寄存器	0x00000000
0x006C	INI_DELTA_POS	增量位置初始化的增量位置寄存器	0x00000000

地址偏移	名称	描述	复位值
0x0070	INI_DELTA_REV	增量位置初始化的增量转数寄存器	0x00000000
0x0074	INI_DELTA_SPEED	增量位置初始化的增量速度寄存器	0x00000000
0x0078	INI_DELTA_ACCEL	增量位置初始化的增量加速度寄存器	0x00000000
0x0080	POS_TRG_CFG	跟踪器位置触发控制寄存器	0x00000000
0x0084	POS_TRG_POS_THR	跟踪器位置门限寄存器	0x00000000
0x0088	POS_TRG_REV_THR	跟踪器转数门限寄存器	0x00000000
0x008C	SPEED_TRG_CFG	跟踪器速度触发控制寄存器	0x00000000
0x0090	SPEED_TRG_THR	跟踪器速度门限寄存器	0x00000000
0x00A0	COEF_TRG_CFG[0][ERR_THR]	跟踪器系数触发控制器 0	0x00000000
0x00A4	COEF_TRG_CFG[0][P]	跟踪器系数触发控制器 0 系数 P	0x00000000
0x00A8	COEF_TRG_CFG[0][I]	跟踪器系数触发控制器 0 系数 I	0x00000000
0x00AC	COEF_TRG_CFG[0][A]	跟踪器系数触发控制器 0 系数 A	0x00000000
0x00B0	COEF_TRG_CFG[0][TIME]	跟踪器系数触发控制器 0 时间	0x00000000
0x00B4	COEF_TRG_CFG[1][ERR_THR]	跟踪器系数触发控制器 1	0x00000000
0x00B8	COEF_TRG_CFG[1][P]	跟踪器系数触发控制器 1 系数 P	0x00000000
0x00BC	COEF_TRG_CFG[1][I]	跟踪器系数触发控制器 1 系数 I	0x00000000
0x00C0	COEF_TRG_CFG[1][A]	跟踪器系数触发控制器 1 系数 A	0x00000000
0x00C4	COEF_TRG_CFG[1][TIME]	跟踪器系数触发控制器 1 时间	0x00000000
0x00C8	COEF_TRG_CFG[2][ERR_THR]	跟踪器系数触发控制器 2	0x00000000
0x00CC	COEF_TRG_CFG[2][P]	跟踪器系数触发控制器 2 系数 P	0x00000000
0x00D0	COEF_TRG_CFG[2][I]	跟踪器系数触发控制器 2 系数 I	0x00000000
0x00D4	COEF_TRG_CFG[2][A]	跟踪器系数触发控制器 2 系数 A	0x00000000
0x00D8	COEF_TRG_CFG[2][TIME]	跟踪器系数触发控制器 2 时间	0x00000000
0x0100	BR[0][BR_CTRL]	预测器控制寄存器	0x00000000
0x0104	BR[0][BR_TIMEOFF]	预测器时间偏置寄存器	0x00000000
0x0108	BR[0][BR_TRG_PERIOD]	预测器触发周期寄存器	0x00000000
0x010C	BR[0][BR_TRG_F_TIME]	预测器第一触发时间寄存器	0x00000000
0x0110	BR[0][BR_ST]	预测器状态寄存器	0x00000000
0x0140	BR[0][BR_TRG_POS_CFG]	预测器的位置触发控制	0x00000000
0x0144	BR[0][BR_TRG_POS_THR]	预测器的位置门限	0x00000000
0x0148	BR[0][BR_TRG_REV_THR]	预测器的转数门限	0x00000000
0x014C	BR[0][BR_TRG_SPEED_CFG]	预测器的速度触发控制	0x00000000
0x0150	BR[0][BR_TRG_SPEED_THR]	预测器的速度门限	0x00000000
0x01C0	BR[0][BR_INI_POS_TIME]	开环模式下的位置初始化的时间戳寄存器	0x00000000
0x01C4	BR[0][BR_INI_POS]	开环模式下的位置初始化的位置寄存器	0x00000000

地址偏移	名称	描述	复位值
0x01C8	BR[0][BR_INI_REV]	开环模式下的位置初始化的转数寄存器	0x00000000
0x01CC	BR[0][BR_INI_SPEED]	开环模式下的位置初始化的速度寄存器	0x00000000
0x01D0	BR[0][BR_INI_ACCEL]	开环模式下的位置初始化的加速度寄存器	0x00000000
0x01D4	BR[0][BR_INI_DELTA_POS_TIME]	预测模式下的增量位置初始化的时间戳寄存器	0x00000000
0x01D8	BR[0][BR_INI_DELTA_POS]	预测模式下的增量位置初始化的位置增量寄存器	0x00000000
0x01DC	BR[0][BR_INI_DELTA_REV]	预测模式下的增量位置初始化的转数增量寄存器	0x00000000
0x01E0	BR[0][BR_INI_DELTA_SPEED]	预测模式下的增量位置初始化的速度增量寄存器	0x00000000
0x01E4	BR[0][BR_INI_DELTA_ACCEL]	预测模式下的增量位置初始化的加速度增量寄存器	0x00000000
0x01EC	BR[0][BR_CUR_POS_TIME]	预测器输出的时间戳监视寄存器	0x00000000
0x01F0	BR[0][BR_CUR_POS]	预测器输出的位置监视寄存器	0x00000000
0x01F4	BR[0][BR_CUR_REV]	预测器输出的转数监视寄存器	0x00000000
0x01F8	BR[0][BR_CUR_SPEED]	预测器输出的速度监视寄存器	0x00000000
0x01FC	BR[0][BR_CUR_ACCEL]	预测器输出的加速度监视寄存器	0x00000000
0x0200	BR[1][BR_CTRL]	预测器控制寄存器	0x00000000
0x0204	BR[1][BR_TIMEOFF]	预测器时间偏置寄存器	0x00000000
0x0208	BR[1][BR_TRG_PERIOD]	预测器触发周期寄存器	0x00000000
0x020C	BR[1][BR_TRG_F_TIME]	预测器第一触发时间寄存器	0x00000000
0x0210	BR[1][BR_ST]	预测器状态寄存器	0x00000000
0x0240	BR[1][BR_TRG_POS_CFG]	预测器的位置触发控制	0x00000000
0x0244	BR[1][BR_TRG_POS_THR]	预测器的位置门限	0x00000000
0x0248	BR[1][BR_TRG_REV_THR]	预测器的转数门限	0x00000000
0x024C	BR[1][BR_TRG_SPEED_CFG]	预测器的速度触发控制	0x00000000
0x0250	BR[1][BR_TRG_SPEED_THR]	预测器的速度门限	0x00000000
0x02C0	BR[1][BR_INI_POS_TIME]	开环模式下的位置初始化的时间戳寄存器	0x00000000
0x02C4	BR[1][BR_INI_POS]	开环模式下的位置初始化的位置寄存器	0x00000000
0x02C8	BR[1][BR_INI_REV]	开环模式下的位置初始化的转数寄存器	0x00000000
0x02CC	BR[1][BR_INI_SPEED]	开环模式下的位置初始化的速度寄存器	0x00000000
0x02D0	BR[1][BR_INI_ACCEL]	开环模式下的位置初始化的加速度寄存器	0x00000000

地址偏移	名称	描述	复位值
0x02D4	BR[1][BR_INI_DELTA_POS_TIME]	预测模式下的增量位置初始化的时间戳寄存器	0x00000000
0x02D8	BR[1][BR_INI_DELTA_POS]	预测模式下的增量位置初始化的位置增量寄存器	0x00000000
0x02DC	BR[1][BR_INI_DELTA_REV]	预测模式下的增量位置初始化的转数增量寄存器	0x00000000
0x02E0	BR[1][BR_INI_DELTA_SPEED]	预测模式下的增量位置初始化的速度增量寄存器	0x00000000
0x02E4	BR[1][BR_INI_DELTA_ACCEL]	预测模式下的增量位置初始化的加速度增量寄存器	0x00000000
0x02EC	BR[1][BR_CUR_POS_TIME]	预测器输出的时间戳监视寄存器	0x00000000
0x02F0	BR[1][BR_CUR_POS]	预测器输出的位置监视寄存器	0x00000000
0x02F4	BR[1][BR_CUR_REV]	预测器输出的转数监视寄存器	0x00000000
0x02F8	BR[1][BR_CUR_SPEED]	预测器输出的速度监视寄存器	0x00000000
0x02FC	BR[1][BR_CUR_ACCEL]	预测器输出的加速度监视寄存器	0x00000000
0x0300	BK0_TIMESTAMP	跟踪器的最近接收到的时间戳监视寄存器	0x00000000
0x0304	BK0_POSITION	跟踪器的最近接收到的位置监视寄存器	0x00000000
0x0308	BK0_REVOLUTION	跟踪器的最近接收到的转数监视寄存器	0x00000000
0x030C	BK0_SPEED	跟踪器的最近接收到的速度监视寄存器	0x00000000
0x0310	BK0_ACCELERATOR	跟踪器的最近接收到的加速度监视寄存器	0x00000000
0x0320	BK1_TIMESTAMP	跟踪器的接收到的历史时间戳监视寄存器	0x00000000
0x0324	BK1_POSITION	跟踪器的接收到的历史位置监视寄存器	0x00000000
0x0328	BK1_REVOLUTION	跟踪器的接收到的历史转数监视寄存器	0x00000000
0x032C	BK1_SPEED	跟踪器的接收到的历史速度监视寄存器	0x00000000
0x0330	BK1_ACCELERATOR	跟踪器的接收到的历史加速度监视寄存器	0x00000000

表 182: MMC 寄存器列表

38.5 MMC 寄存器详细信息

MMC 的寄存器详细说明如下：

38.5.1 CR (0x0)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SFTRST	RSVD	INI_BR0_POS_REQ	INI_BR1_POS_REQ	FRCACCELZERO	MS_COEF_EN	INI_DELTA_POS_TRG_TYPE			INI_POS_TRG_TYPE			INI_DELTA_POS_CMD_MSK				INI_DELTA_POS_REQ	OPEN_LOOP_MODE	POS_TYPE	INI_POS_CMD_MSK				INI_POS_REQ	INI_COEFS_CMD_MSK			INI_COEFS_CMD	SHADOW_RD_REQ	ADJOP	DISCRETETRC	MOD_EN
RW	N/A	RW	RW	RW	RW	RW			RW			RW				RW	RW	RW	RW				RW	RW			RW	RW	RW	RW	RW
0	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

CR [31:0]

位域	名称	描述
31	SFTRST	软件复位，高有效。写入 1 时，所有内部逻辑将被重置。 0b-无操作 1b-所有 MMC 内部寄存器被强制进入复位状态。接口寄存器不受影响。
29	INI_BR0_POS_REQ	0 号预测器位置初始化命令。仅在开环模式下用。
28	INI_BR1_POS_REQ	1 号预测器位置初始化命令。仅在开环模式下用。
27	FRCACCELZERO	1：在计算中强制加速度估计值为 0 0：在计算中不强制加速度估计值为 0
26	MS_COEF_EN	1：使能多种系数自动选择。涉及到 COEF_TRG_CFG[n] (n=0,1,2) 中数据的应用 0：不使能多种系数自动选择。
25-23	INI_DELTA_POS_TRG_TYPE	增量位置初始化的触发方法选择： 0: 使用配置中的时间戳来触发 1: 使用模块输入 INTRGR[0] 信号上升沿来触发 2: 使用模块输入 INTRGR[1] 信号上升沿来触发 3: 使用模块输出 OUTTRGR[0] 信号上升沿来触发 4: 使用模块输出 OUTTRGR[0] 信号上升沿来触发 5: 使用跟踪器自己的位置触发信号来触发 6: 使用跟踪器自己的速度触发信号来触发 其他：无功能

位域	名称	描述
22-20	INI_POS_TRG_TYPE	位置初始化的触发方法选择： 0: 使用配置中的时间戳来触发 1: 使用模块输入 INTRGR[0] 信号上升沿来触发 2: 使用模块输入 INTRGR[1] 信号上升沿来触发 3: 使用模块输出 OUTTRGR[0] 信号上升沿来触发 4: 使用模块输出 OUTTRGR[0] 信号上升沿来触发 5: 使用跟踪器自己的位置触发信号来触发 6: 使用跟踪器自己的速度触发信号来触发 其他：无功能
19-16	INI_DELTA_POS_CMD_MSK	增量位置初始化的掩码： 1: 此数值要更新 0: 此数值不更新 bit 3: 对加速度增量的选择 bit 2: 对速度增量的选择 bit 1: 对转数增量的选择 bit 0: 对位置增量的选择
15	INI_DELTA_POS_REQ	1: 重新加载增量位置命令。自动清零。 0:
14	OPEN_LOOP_MODE	1: 开环模式 0: 闭环模式
13	POS_TYPE	在读取估计位置 ESTM_POS 寄存器时 1: 转数为 16 比特，位置为 16 比特，合为一个 32 比特 0: 转数为 32 比特，位置为 32 比特
12-9	INI_POS_CMD_MSK	位置初始化的掩码： 1: 改变 0: 不改变 bit 3: 加速度选择 bit 2: 速度选择 bit 1: 转数选择 bit 0: 位置选择
8	INI_POS_REQ	1: 重新加载位置命令。自动清零。 0:
7-5	INI_COEFS_CMD_MSK	1: 改变 0: 不改变 bit 2: A 系数选择 bit 1: I 系数选择 bit 0: P 系数选择
4	INI_COEFS_CMD	1: 重新加载系数命令。自动清零。 0:
3	SHADOW_RD_REQ	1: 追踪器内部参数影像化读取允许。自动清零 0:

位域	名称	描述
2	ADJOP	1: 采用新输入数据为新跟踪数据的起点，并且在跟踪时有边界控制最大误差 0: 不采用新输入数据为新跟踪数据的起点，无边界控制最大误差
1	DISCRETETRC	1: 离散位置输入 0: 连续位置输入
0	MOD_EN	模块使能

CR 位域

38.5.2 STA (0x4)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ERR_ID				RSVD																	SPEED_TRG_VALID	POS_TRG_VALID	INI_DELTA_POS_REQ_CMD_DONE	INI_BR0_POS_REQ_CMD_DONE	INI_BR1_POS_REQ_CMD_DONE	IDLE	OOSYNC	RSVD	INI_POS_REQ_CMD_DONE	INI_COEFS_CMD_DONE	SHADOW_RD_DONE
RO				N/A																	W1C	W1C	W1C	W1C	W1C	RO	W1C	N/A	W1C	W1C	RO
0	0	0	0	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	1	0	x	0	0	0

STA [31:0]

位域	名称	描述
31-28	ERR_ID	跟踪器的 ERR_ID
10	SPEED_TRG_VALID	1：速度触发事件已发生。
9	POS_TRG_VALID	1：位置触发事件已发生。
8	INI_DELTA_POS_REQ_CMD_DONE	1：重新加载增量位置事件已完成
7	INI_BR0_POS_REQ_CMD_DONE	1：重新加载 0 号预测器位置事件已完成
6	INI_BR1_POS_REQ_CMD_DONE	1：重新加载 1 号预测器位置事件已完成
5	IDLE	跟踪器处在空闲状态
4	OOSYNC	跟踪器失步。写一清零。
2	INI_POS_REQ_CMD_DONE	重新加载位置命令完成。写一清零。
1	INI_COEFS_CMD_DONE	重新加载系数命令完成。写一清零。

位域	名称	描述
0	SHADOW_RD_DONE	追踪器内部参数影像化完成。被 CR[SHADOW_RD_REQ] 置为 1 的操作清零

STA 位域

38.5.3 INT_EN (0x8)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																					SPEED_TRG_VLD_IE	POS_TRG_VLD_IE	INI_DELTA_POS_REQ_CMD_DONE_IE	INI_BR0_POS_REQ_CMD_DONE_IE	INI_BR1_POS_REQ_CMD_DONE_IE	RSVD	OOSYNC_IE	RSVD	INI_POS_REQ_CMD_DONE_IE	INI_COEFS_CMD_DONE_IE	SHADOW_RD_DONE_IE
N/A																					RW	RW	RW	RW	RW	N/A	RW	N/A	RW	RW	RW
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	x	0	x	0	0	0

INT_EN [31:0]

位域	名称	描述
10	SPEED_TRG_VLD_IE	1：速度触发事件中断允许。
9	POS_TRG_VLD_IE	1：位置触发事件中断允许。
8	INI_DELTA_POS_REQ_CMD_DONE_IE	1：重新加载增量位置事件中断允许
7	INI_BR0_POS_REQ_CMD_DONE_IE	1：重新加载 0 号预测器位置事件中断允许
6	INI_BR1_POS_REQ_CMD_DONE_IE	1：重新加载 1 号预测器位置事件中断允许
4	OOSYNC_IE	OOSYNC 中断允许
2	INI_POS_REQ_CMD_DONE_IE	INI_POS_REQ_CMD_DONE 中断允许
1	INI_COEFS_CMD_DONE_IE	INI_COEFS_CMD_DONE 中断允许
0	SHADOW_RD_DONE_IE	SHADOW_RD_DONE 中断允许

位域	名称	描述

INT_EN 位域

38.5.4 SYSCLK_FREQ (0xC)

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

SYSCLK_FREQ [31:0]

位域	名称	描述
31-0	VAL	系统时钟主频, ufix<32, 0>

SYSCLK_FREQ 位域

38.5.5 SYSCLK_PERIOD (0x10)

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

SYSCLK_PERIOD [31:0]

位域	名称	描述
31-0	VAL	系统时钟周期，数值为时钟周期 *(2^24)*(2^20) 后取整，ufix<32, 0>

SYSCLK_PERIOD 位域

38.5.6 OOSYNC_THETA_THR (0x14)

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

OOSYNC_THETA_THR [31:0]

位域	名称	描述
31-0	VAL	失步判决门限，ufix<32, 32>

OOSYNC_THETA_THR 位域

38.5.7 DISCRETECFG0 (0x18)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD												POSMAX																			
N/A												RW																			
x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

DISCRETECFG0 [31:0]

位域	名称	描述
19-0	POSMAX	离散输入中的最大值，为线数-1。ufix<32，0>

DISCRETECFG0 位域

38.5.8 DISCRETECFG1 (0x1C)

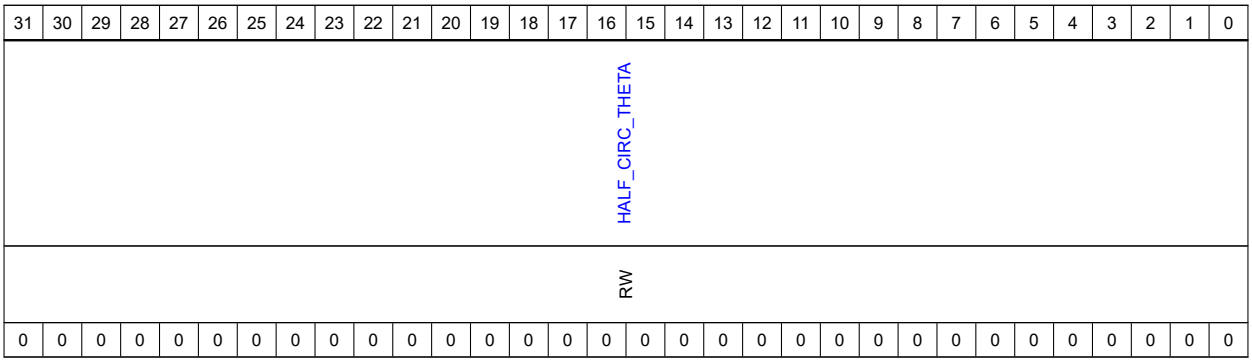
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
INV_POSMAX																															
RW																															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

DISCRETECFG1 [31:0]

位域	名称	描述
31-0	INV_POSMAX	离散模式：线数的倒数, ufix<32, 0> 连续模式：基于上一个位置的跟踪上下界限, ufix<32, 32>

DISCRETECFG1 位域

38.5.9 CONTCFG0 (0x20)

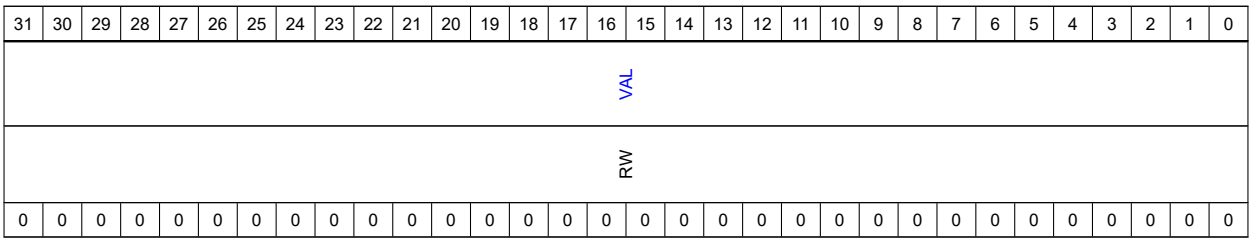


CONTCFG0 [31:0]

位域	名称	描述
31-0	HALF_CIRC_THETA	连续模式下的正反转判决门限，ufix<32，32>

CONTCFG0 位域

38.5.10 INI_POS_TIME (0x24)

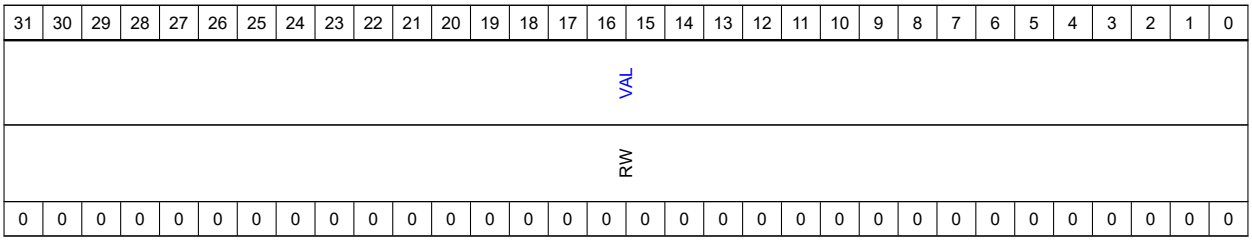


INI_POS_TIME [31:0]

位域	名称	描述
31-0	VAL	标志什么时间做改变。 0：立即改变

INI_POS_TIME 位域

38.5.11 INI_POS (0x28)



INI_POS [31:0]

位域	名称	描述
31-0	VAL	数值； 连续模式：ufix<32，32>

INI_POS 位域

38.5.12 INI_REV (0x2C)

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

INI_REV [31:0]

位域	名称	描述
31-0	VAL	数值； 连续模式：ufix<32, 0>

INI_REV 位域

38.5.13 INI_SPEED (0x30)

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

INI_SPEED [31:0]

位域	名称	描述
31-0	VAL	数值; 连续模式：fix<32, 19>

INI_SPEED 位域

38.5.14 INI_ACCEL (0x34)

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

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

INI_ACCEL [31:0]

位域	名称	描述
31-0	VAL	数值; 连续模式下：fix<32, 19>

INI_ACCEL 位域

38.5.15 INI_COEF_TIME (0x38)

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

INI_COEF_TIME [31:0]

位域	名称	描述
31-0	VAL	标志什么时间做改变。 0：立即改变

INI_COEF_TIME 位域

38.5.16 INI_PCOEF (0x3C)

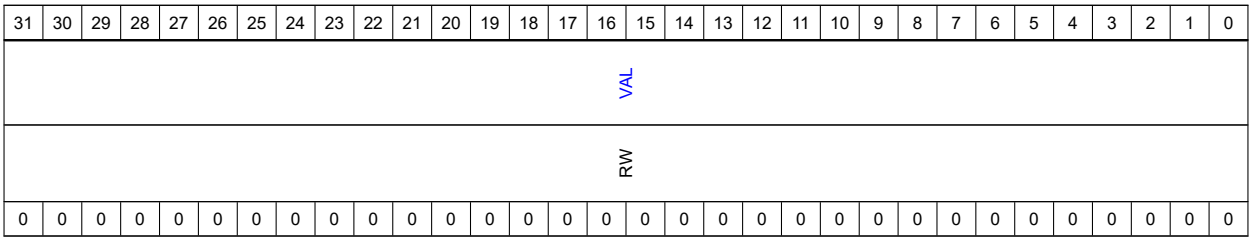
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VAL																															
RW																															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

INI_PCOEF [31:0]

位域	名称	描述
31-0	VAL	数值，fix<32, 15>

INI_PCOEF 位域

38.5.17 INI_ICOEF (0x40)

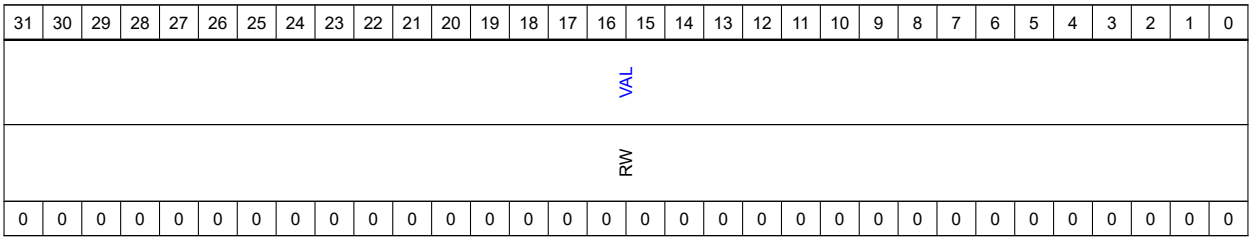


INI_ICOEF [31:0]

位域	名称	描述
31-0	VAL	数值，fix<32, 21>

INI_ICOEF 位域

38.5.18 INI_ACOEF (0x44)

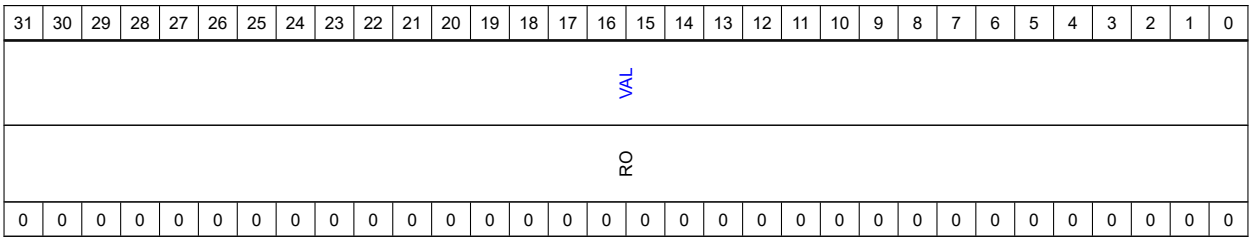


INI_ACOEF [31:0]

位域	名称	描述
31-0	VAL	数值，fix<32, 19>

INI_ACOEF 位域

38.5.19 ESTM_TIM (0x48)



ESTM_TIM [31:0]

位域	名称	描述
31-0	VAL	数值

ESTM_TIM 位域

38.5.20 ESTM_POS (0x4C)

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

ESTM_POS [31:0]

位域	名称	描述
31-0	VAL	数值

ESTM_POS 位域

38.5.21 ESTM_REV (0x50)

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

ESTM_REV [31:0]

位域	名称	描述
31-0	VAL	数值

ESTM_REV 位域

38.5.22 ESTM_SPEED (0x54)

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

ESTM_SPEED [31:0]

位域	名称	描述
31-0	VAL	数值

ESTM_SPEED 位域

38.5.23 ESTM_ACCEL (0x58)

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

ESTM_ACCEL [31:0]

位域	名称	描述
31-0	VAL	数值

ESTM_ACCEL 位域

38.5.24 CUR_PCOEF (0x5C)

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

CUR_PCOEF [31:0]

位域	名称	描述
31-0	VAL	数值

CUR_PCOEF 位域

38.5.25 CUR_ICOEF (0x60)

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

CUR_ICOEF [31:0]

位域	名称	描述
31-0	VAL	数值

CUR_ICOEF 位域

38.5.26 CUR_ACOEF (0x64)

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

CUR_ACOEF [31:0]

位域	名称	描述
31-0	VAL	数值

CUR_ACOEF 位域

38.5.27 INI_DELTA_POS_TIME (0x68)

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

INI_DELTA_POS_TIME [31:0]

位域	名称	描述
31-0	VAL	标志什么时间做改变。 0：立即改变

INI_DELTA_POS_TIME 位域

38.5.28 INI_DELTA_POS (0x6C)

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

INI_DELTA_POS [31:0]

位域	名称	描述
31-0	VAL	数值; 连续模式：ufix<32, 32>

INI_DELTA_POS 位域

38.5.29 INI_DELTA_REV (0x70)

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

INI_DELTA_REV [31:0]

位域	名称	描述
31-0	VAL	数值 连续模式: fix<32, 0>

INI_DELTA_REV 位域

38.5.30 INI_DELTA_SPEED (0x74)

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

INI_DELTA_SPEED [31:0]

位域	名称	描述
31-0	VAL	数值； 连续模式：fix <32, 19>

INI_DELTA_SPEED 位域

38.5.31 INI_DELTA_ACCEL (0x78)

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

INI_DELTA_ACCEL [31:0]

位域	名称	描述
31-0	VAL	数值 连续模式: fix<32, 19>

INI_DELTA_ACCEL 位域

38.5.32 POS_TRG_CFG (0x80)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																														EDGE	EN
N/A																														RW	RW
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0

POS_TRG_CFG [31:0]

位域	名称	描述
1	EDGE	0: 上升沿（即位置增长超过门限时）, 1: 下降沿，（即位置下降低于门限时）
0	EN	1-该触发规则有效; 0-该触发规则无效

POS_TRG_CFG 位域

38.5.33 POS_TRG_POS_THR (0x84)

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

POS_TRG_POS_THR [31:0]

位域	名称	描述
31-0	VAL	位置门限的数值 ufix<32, 32>

POS_TRG_POS_THR 位域

38.5.34 POS_TRG_REV_THR (0x88)

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

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

POS_TRG_REV_THR [31:0]

位域	名称	描述
31-0	VAL	转数门限的数值 fix<32, 0>

POS_TRG_REV_THR 位域

38.5.35 SPEED_TRG_CFG (0x8C)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																													COMP_TYPE	EDGE	EN
N/A																													RW	RW	RW
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0

SPEED_TRG_CFG [31:0]

位域	名称	描述
2	COMP_TYPE	1: 用绝对值来做比较. 0: 用补码表示的正负数值做比较
1	EDGE	0: 上升沿（即速度增长超过门限时）, 1: 下降沿，（即速度下降低于门限时）
0	EN	1-该触发规则有效; 0-该触发规则无效

SPEED_TRG_CFG 位域

38.5.36 SPEED_TRG_THR (0x90)

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

SPEED_TRG_THR [31:0]

位域	名称	描述
31-0	VAL	速度门限的数值 连续模式: fix<32, 19>

位域	名称	描述

SPEED_TRG_THR 位域

38.5.37 COEF_TRG_CFG[ERR_THR] (0xA0 + 0x14 * n)

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

COEF_TRG_CFG[ERR_THR] [31:0]

位域	名称	描述
31-0	VAL	ErrThr0: 误差门限 0，当跟踪误差的绝对值 >= 此数值时，选择下面的 P0/I0/A0 系数 注意：ErrThr0>ErrThr1>ErrThr2 ufix<31，28>

COEF_TRG_CFG[ERR_THR] 位域

38.5.38 COEF_TRG_CFG[P] (0xA4 + 0x14 * n)

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

COEF_TRG_CFG[P] [31:0]

位域	名称	描述
31-0	VAL	系数 P0, fix<32, 15>

COEF_TRG_CFG[P] 位域

38.5.39 COEF_TRG_CFG[I] (0xA8 + 0x14 * n)

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

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

COEF_TRG_CFG[I] [31:0]

位域	名称	描述
31-0	VAL	系数 I0，fix<32，21>

COEF_TRG_CFG[I] 位域

38.5.40 COEF_TRG_CFG[A] (0xAC + 0x14 * n)

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

COEF_TRG_CFG[A] [31:0]

位域	名称	描述
31-0	VAL	系数 A0, fix<32，19>

COEF_TRG_CFG[A] 位域

38.5.41 COEF_TRG_CFG[TIME] (0xB0 + 0x14 * n)

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

COEF_TRG_CFG[TIME] [31:0]

位域	名称	描述
31-0	VAL	CoefTime0: 停留在第 0 套系数值的最少时间（以一个输入采样为单位 1）。为真实停留的采样数-1。当此停留时间用完，如果误差绝对值仍旧大于 ErrThr0，继续选择第 0 套系数；否则选择下一套系数（第 1 套系数）。ufix<32,0>

COEF_TRG_CFG[TIME] 位域

38.5.42 BR[BR_CTRL] (0x100 + 0x100 * n)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD	SPEED_TRG_VALID_IE	POS_TRG_VALID_IE	RSVD			INI_POS_TRG_TYPE			RSVD	INI_POS_CMD_MSK				RSVD	INI_DELTA_POS_TRG_TYPE			INI_DELTA_POS_DONE_IE	INI_DELTA_POS_CMD_MSK				INI_DELTA_POS_REQ	OPEN_LOOP_MODE	RSVD	PRED_MODE		RSVD	NF_TRG_TYPE	F_TRG_TYPE	BR_EN
N/A	RW	RW	N/A			RW			N/A	RW				N/A	RW			RW	RW				RW	RW	N/A	RW		N/A	RW	RW	RW
x	0	0	x	x	x	0	0	0	x	0	0	0	0	x	0	0	0	0	0	0	0	0	0	0	x	0	0	x	0	0	0

BR[BR_CTRL] [31:0]

位域	名称	描述
30	SPEED_TRG_VALID_IE	1：速度触发事件中断允许。
29	POS_TRG_VALID_IE	1：位置触发事件中断允许。
25-23	INI_POS_TRG_TYPE	位置初始化的触发方法选择： 0: 使用配置中的时间戳来触发 1: 使用模块输入 INTRGR[0] 信号上升沿来触发 2: 使用模块输入 INTRGR[1] 信号上升沿来触发 3: 使用模块输出 OUTTRGR[0] 信号上升沿来触发 4: 使用模块输出 OUTTRGR[1] 信号上升沿来触发 5: 使用跟踪器自己的位置触发信号来触发 6: 使用跟踪器自己的速度触发信号来触发 其他：无功能
21-18	INI_POS_CMD_MSK	位置初始化的掩码： 1: 改变 0: 不改变 bit 3: 加速度选择 bit 2: 速度选择 bit 1: 转数选择 bit 0: 位置选择
16-14	INI_DELTA_POS_TRG_TYPE	增量位置初始化的触发方法选择： 0: 使用配置中的时间戳来触发 1: 使用模块输入 INTRGR[0] 信号上升沿来触发 2: 使用模块输入 INTRGR[1] 信号上升沿来触发 3: 使用模块输出 OUTTRGR[0] 信号上升沿来触发 4: 使用模块输出 OUTTRGR[1] 信号上升沿来触发 5: 使用跟踪器自己的位置触发信号来触发 6: 使用跟踪器自己的速度触发信号来触发 其他：无功能

位域	名称	描述
13	INI_DELTA_POS_DONE_IE	1. INI_DELTA_POS_DONE 中断允许
12-9	INI_DELTA_POS_CMD_MSK	增量位置初始化的掩码： 1: 此数值要更新 0: 此数值不更新 bit 3: 对加速度增量的选择 bit 2: 对速度增量的选择 bit 1: 对转数增量的选择 bit 0: 对位置增量的选择
8	INI_DELTA_POS_REQ	1: 重新加载增量位置命令。自动清零。 0:
7	OPEN_LOOP_MODE	1: 开环模式 0: 闭环模式
5-4	PRED_MODE	1: 自动无周期连续预测 0: 基于周期性预测 2: 只预测一次
2	NF_TRG_TYPE	1. 非首次预测由外部触发产生 0. 非首次预测由内部时钟触发产生
1	F_TRG_TYPE	1. 首次预测由外部触发产生 0. 首次预测由内部时钟触发产生
0	BR_EN	预测器使能

BR[BR_CTRL] 位域

38.5.43 BR[BR_TIMEOFF] (0x104 + 0x100 * n)

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

BR[BR_TIMEOFF] [31:0]

位域	名称	描述
31-0	VAL	ufix<32，0>, 预测器的预测提前量

BR[BR_TIMEOFF] 位域

38.5.44 BR[BR_TRG_PERIOD] (0x108 + 0x100 * n)

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

BR[BR_TRG_PERIOD] [31:0]

位域	名称	描述
31-0	VAL	ufix<32，0>, 预测器的周期

BR[BR_TRG_PERIOD] 位域

38.5.45 BR[BR_TRG_F_TIME] (0x10C + 0x100 * n)

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

BR[BR_TRG_F_TIME] [31:0]

位域	名称	描述
31-0	VAL	ufix<32，0>, 预测器的的第一次使能

BR[BR_TRG_F_TIME] 位域

38.5.46 BR[BR_ST] (0x110 + 0x100 * n)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																					OPEN_LOOP_ST	SPEED_TRG_VLD	POS_TRG_VLD	RSVD	INI_DELTA_POS_DONE	IDLE	RSVD	ERR_ID			
N/A																					RO	W1C	W1C	N/A	W1C	RO	N/A	RO			
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	x	0	1	x	0	0	0	0

BR[BR_ST] [31:0]

位域	名称	描述
10	OPEN_LOOP_ST	1：预测器处于开环状态下 0：预测器处于闭环状态下
9	SPEED_TRG_VLD	1：速度触发事件已发生 0：速度触发事件未发生
8	POS_TRG_VLD	1：位置触发事件已发生 0：位置触发事件未发生
6	INI_DELTA_POS_DONE	1: 增量位置初始化命令已完成 0: 增量位置初始化命令未完成
5	IDLE	1：预测器处在空闲状态 0：预测器处在计算中
3-0	ERR_ID	预测器 ERR_ID

BR[BR_ST] 位域

38.5.47 BR[BR_TRG_POS_CFG] (0x140 + 0x100 * n)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																														EDGE	EN
N/A																														RW	RW
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0

BR[BR_TRG_POS_CFG] [31:0]

位域	名称	描述
1	EDGE	0: 上升沿（即位置增长超过门限时）, 1: 下降沿，（即位置下降低于门限时）
0	EN	1-该触发规则有效; 0-该触发规则无效

BR[BR_TRG_POS_CFG] 位域

38.5.48 BR[BR_TRG_POS_THR] (0x144 + 0x100 * n)

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

BR[BR_TRG_POS_THR] [31:0]

位域	名称	描述
31-0	VAL	位置门限的数值 ufix<32, 32>

BR[BR_TRG_POS_THR] 位域

38.5.49 BR[BR_TRG_REV_THR] (0x148 + 0x100 * n)

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

BR[BR_TRG_REV_THR] [31:0]

位域	名称	描述
31-0	VAL	转数门限的数值 ufix<32, 0>

BR[BR_TRG_REV_THR] 位域

38.5.50 BR[BR_TRG_SPEED_CFG] (0x14C + 0x100 * n)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																													COMP_TYPE	EDGE_SEL	EN
N/A																													RW	RW	RW
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0

BR[BR_TRG_SPEED_CFG] [31:0]

位域	名称	描述
2	COMP_TYPE	1: 用绝对值来做比较. 0: 用补码表示的正负数值做比较
1	EDGE_SEL	0: 上升沿（即速度增长超过门限时）, 1: 下降沿，（即速度下降低于门限时）
0	EN	1-该触发规则有效; 0-该触发规则无效

BR[BR_TRG_SPEED_CFG] 位域

38.5.51 BR[BR_TRG_SPEED_THR] (0x150 + 0x100 * n)

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

BR[BR_TRG_SPEED_THR] [31:0]

位域	名称	描述
31-0	VAL	速度门限的数值 连续模式: fix<32, 19>

BR[BR_TRG_SPEED_THR] 位域

38.5.52 BR[BR_INI_POS_TIME] (0x1C0 + 0x100 * n)

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

BR[BR_INI_POS_TIME] [31:0]

位域	名称	描述
31-0	VAL	标志什么时间做改变。 0：立即改变

BR[BR_INI_POS_TIME] 位域

38.5.53 BR[BR_INI_POS] (0x1C4 + 0x100 * n)

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

BR[BR_INI_POS] [31:0]

位域	名称	描述
31-0	VAL	数值 ufix<32, 32>

BR[BR_INI_POS] 位域

38.5.54 BR[BR_INI_REV] (0x1C8 + 0x100 * n)

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

BR[BR_INI_REV] [31:0]

位域	名称	描述
31-0	VAL	数值 ufix<32, 0>

BR[BR_INI_REV] 位域

38.5.55 BR[BR_INI_SPEED] (0x1CC + 0x100 * n)

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

BR[BR_INI_SPEED] [31:0]

位域	名称	描述
31-0	VAL	数值 连续模式: fix<32, 19>

BR[BR_INI_SPEED] 位域

38.5.56 BR[BR_INI_ACCEL] (0x1D0 + 0x100 * n)

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

BR[BR_INI_ACCEL] [31:0]

位域	名称	描述
31-0	VAL	数值 连续模式: fix<32, 19>

BR[BR_INI_ACCEL] 位域

38.5.57 BR[BR_INI_DELTA_POS_TIME] (0x1D4 + 0x100 * n)

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

BR[BR_INI_DELTA_POS_TIME] [31:0]

位域	名称	描述
31-0	VAL	标志什么时间做改变。 0：立即改变

BR[BR_INI_DELTA_POS_TIME] 位域

38.5.58 BR[BR_INI_DELTA_POS] (0x1D8 + 0x100 * n)

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

BR[BR_INI_DELTA_POS] [31:0]

位域	名称	描述
31-0	VAL	数值 连续模式: ufix<32, 32>

BR[BR_INI_DELTA_POS] 位域

38.5.59 BR[BR_INI_DELTA_REV] (0x1DC + 0x100 * n)

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

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

BR[BR_INI_DELTA_REV] [31:0]

位域	名称	描述
31-0	VAL	数值 连续模式: fix<32, 0>

BR[BR_INI_DELTA_REV] 位域

38.5.60 BR[BR_INI_DELTA_SPEED] (0x1E0 + 0x100 * n)

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

BR[BR_INI_DELTA_SPEED] [31:0]

位域	名称	描述
31-0	VAL	数值 连续模式: fix<32, 19>

BR[BR_INI_DELTA_SPEED] 位域

38.5.61 BR[BR_INI_DELTA_ACCEL] (0x1E4 + 0x100 * n)

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

BR[BR_INI_DELTA_ACCEL] [31:0]

位域	名称	描述
31-0	VAL	数值 连续模式：fix<32, 19>

BR[BR_INI_DELTA_ACCEL] 位域

38.5.62 BR[BR_CUR_POS_TIME] (0x1EC + 0x100 * n)

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

BR[BR_CUR_POS_TIME] [31:0]

位域	名称	描述
31-0	VAL	数值

BR[BR_CUR_POS_TIME] 位域

38.5.63 BR[BR_CUR_POS] (0x1F0 + 0x100 * n)

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

BR[BR_CUR_POS] [31:0]

位域	名称	描述
31-0	VAL	数值

BR[BR_CUR_POS] 位域

38.5.64 BR[BR_CUR_REV] (0x1F4 + 0x100 * n)

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

BR[BR_CUR_REV] [31:0]

位域	名称	描述
31-0	VAL	数值

BR[BR_CUR_REV] 位域

38.5.65 BR[BR_CUR_SPEED] (0x1F8 + 0x100 * n)

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

BR[BR_CUR_SPEED] [31:0]

位域	名称	描述
31-0	VAL	数值

BR[BR_CUR_SPEED] 位域

38.5.66 BR[BR_CUR_ACCEL] (0x1FC + 0x100 * n)

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

BR[BR_CUR_ACCEL] [31:0]

位域	名称	描述
31-0	VAL	数值

BR[BR_CUR_ACCEL] 位域

38.5.67 BK0_TIMESTAMP (0x300)

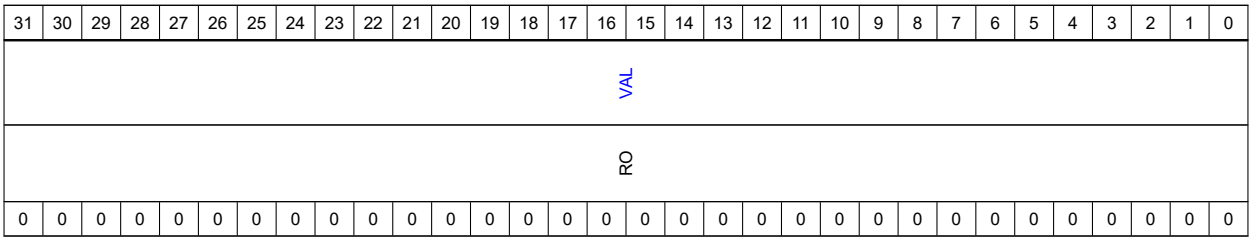
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VAL																															
RO																															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

BK0_TIMESTAMP [31:0]

位域	名称	描述
31-0	VAL	数值

BK0_TIMESTAMP 位域

38.5.68 BK0_POSITION (0x304)

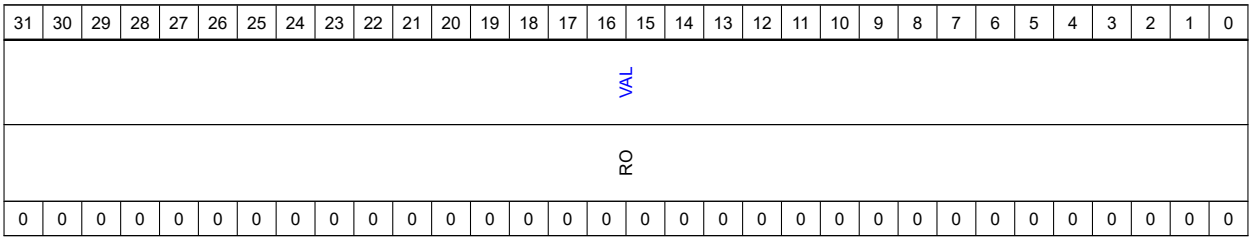


BK0_POSITION [31:0]

位域	名称	描述
31-0	VAL	数值

BK0_POSITION 位域

38.5.69 BK0_REVOLUTION (0x308)

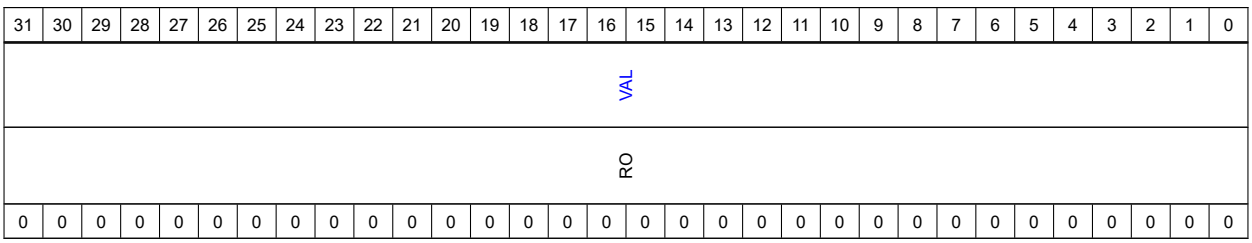


BK0_REVOLUTION [31:0]

位域	名称	描述
31-0	VAL	数值

BK0_REVOLUTION 位域

38.5.70 BK0_SPEED (0x30C)



BK0_SPEED [31:0]

位域	名称	描述
31-0	VAL	数值

BK0_SPEED 位域

38.5.71 BK0_ACCELERATOR (0x310)

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

BK0_ACCELERATOR [31:0]

位域	名称	描述
31-0	VAL	数值

BK0_ACCELERATOR 位域

38.5.72 BK1_TIMESTAMP (0x320)

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

BK1_TIMESTAMP [31:0]

位域	名称	描述
31-0	VAL	数值

BK1_TIMESTAMP 位域

38.5.73 BK1_POSITION (0x324)

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

BK1_POSITION [31:0]

位域	名称	描述
31-0	VAL	数值

BK1_POSITION 位域

38.5.74 BK1_REVOLUTION (0x328)

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

BK1_REVOLUTION [31:0]

位域	名称	描述
31-0	VAL	数值

BK1_REVOLUTION 位域

38.5.75 BK1_SPEED (0x32C)

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

BK1_SPEED [31:0]

位域	名称	描述
31-0	VAL	数值

BK1_SPEED 位域

38.5.76 BK1_ACCELERATOR (0x330)

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

BK1_ACCELERATOR [31:0]

位域	名称	描述
31-0	VAL	数值

BK1_ACCELERATOR 位域

38.6 MMC 配置使用简单说明

38.6.1 软件复位

将 CR[SFTRST] 置为 1 后复位，再将 CR[SFTRST] 置为 0 后解除复位。

38.6.2 配置位置、速度、加速度

有多种触发方式，先以时间触发方式举例。

对跟踪器的相关配置各步骤如下：

1. 将寄存器 INI_POS_TIME 配成要对位置等参量进行更新的时刻。
2. 将寄存器 INI_POS 、INI_REV 、INI_SPEED 、INI_ACCEL 配成要更改成的数据。不需要更改的，可以不配置。
3. 按照需要对寄存器 CR [INI_POS_CMD_MSK] 配置。
4. 按照所选择的触发类型对寄存器 CR [INI_POS_TRG_TYPE] 配置为 0。
5. 将寄存器 CR [INI_POS_REQ] 配成 1。

这样，在指定时间上，就可以对跟踪器的位置等参量做出修改了。

对 0 号预测器的相关配置各步骤如下：

1. 将寄存器 BR [0] [BR_INI_POS_TIME] 配成要对位置等参量进行更新的时刻。
2. 将寄存器 BR [0] [BR_INI_POS] 、BR [0] [BR_INI_REV] 、BR [0] [BR_INI_SPEED] 、BR [0] [BR_INI_ACCEL] 配成要更改成的数据。不需要更改的，可以不配置。
3. 按照需要对寄存器 BR [0] [BR_CTRL [INI_POS_CMD_MSK]] 配置。
4. 按照所选择的触发类型对寄存器 BR [0] [BR_CTRL [INI_POS_TRG_TYPE]] 配置为 0。
5. 将寄存器 CR [INI_BR0_POS_REQ] 配成 1。

这样，在指定时间上，就可以对 0 号预测器的位置等参量做出修改了。

对 1 号预测器的相关配置各步骤如下：

1. 将寄存器 BR [1] [BR_INI_POS_TIME] 配成要对位置等参量进行更新的时刻。
2. 将寄存器 BR [1] [BR_INI_POS] 、BR [1] [BR_INI_REV] 、BR [1] [BR_INI_SPEED] 、BR [1] [BR_INI_ACCEL] 配成要更改成的数据。不需要更改的，可以不配置。
3. 按照需要对寄存器 BR [1] [BR_CTRL [INI_POS_CMD_MSK]] 配置。
4. 按照所选择的触发类型对寄存器 BR [1] [BR_CTRL [INI_POS_TRG_TYPE]] 配置为 0。
5. 将寄存器 CR [INI_BR1_POS_REQ] 配成 1。

这样，在指定时间上，就可以对 1 号预测器的位置等参量做出修改了。

38.6.3 配置位置、速度、加速度的增量

有多种触发方式，先以时间触发方式举例。

对跟踪器的相关配置各步骤如下：

1. 将寄存器 INI_DELTA_POS_TIME 配成要对位置等参量进行更新的时刻。
2. 将寄存器 INI_DELTA_POS 、INI_DELTA_REV 、INI_DELTA_SPEED 、INI_DELTA_ACCEL 配成要更改成的数据。不需要更改的，可以不配置。
3. 按照需要对寄存器 CR [INI_DELTA_POS_CMD_MSK] 配置。
4. 按照所选择的触发类型对寄存器 CR [INI_DELTA_POS_TRG_TYPE] 配置为 0。
5. 将寄存器 CR [INI_DELTA_POS_REQ] 配成 1。

这样，在指定时间上，就可以对跟踪器的位置增量等参量做出修改了。

对 0 号预测器的相关配置各步骤如下：

1. 将寄存器 BR [0] [BR_INI_DELTA_POS_TIME] 配成要对位置等参量进行更新的时刻。
2. 将寄存器 BR [0] [BR_INI_DELTA_POS] 、BR [0] [BR_INI_DELTA_REV] 、BR [0] [BR_INI_DELTA_SPEED] 、BR [0] [BR_INI_DELTA_ACCEL] 配成要更改成的数据。不需要更改的，可以不配置。
3. 按照需要对寄存器 BR [0] [BR_CTRL [INI_DELTA_POS_CMD_MSK]] 配置。
4. 按照所选择的触发类型对寄存器 BR [0] [BR_CTRL [INI_DELTA_POS_TRG_TYPE]] 配置为 0。
5. 将寄存器 BR [0] [BR_CTRL [INI_DELTA_POS_REQ]] 配成 1。

这样，在指定时间上，就可以对 0 号预测器的位置增量等参量做出修改了。

对 1 号预测器的相关配置各步骤如下：

1. 将寄存器 BR [1] [BR_INI_DELTA_POS_TIME] 配成要对位置等参量进行更新的时刻。
2. 将寄存器 BR [1] [BR_INI_DELTA_POS] 、BR [1] [BR_INI_DELTA_REV] 、BR [1] [BR_INI_DELTA_SPEED] 、BR [1] [BR_INI_DELTA_ACCEL] 配成要更改成的数据。不需要更改的，可以不配置。
3. 按照需要对寄存器 BR [1] [BR_CTRL [INI_DELTA_POS_CMD_MSK]] 配置。
4. 按照所选择的触发类型对寄存器 BR [1] [BR_CTRL [INI_DELTA_POS_TRG_TYPE]] 配置为 0。
5. 将寄存器 BR [1] [BR_CTRL [INI_DELTA_POS_REQ]] 配成 1。

这样，在指定时间上，就可以对 1 号预测器的位置增量等参量做出修改了。

38.6.4 配置 P、I、A 参数

各步骤如下：

1. 将寄存器 INI_COEF_TIME 配成要对 P、I、A 参数进行更新的时刻。
2. 将寄存器 INI_PCOEF 、INI_ICOEF 、INI_ACOEF 配成要更改成的数据。不需要更改的，可以不配置。
3. 按照需要对寄存器 CR [INI_COEFS_CMD_MSK] 配置。
4. 将寄存器 CR [INI_COEFS_REQ] 配成 1。

这样，在指定时间上，就可以对跟踪器的 P、I、A 参数做出修改了。

38.6.5 配置触发机制

本模块支持多种触发机制，如用时间、或位置、或速度、或输入 INTRGR[0]、或输入 INTRGR[1]、或输出 OUTTRGR[0]、或输出 OUTTRGR[1] 来触发。

以对 1 号预测器的速度触发来配置 1 号预测器的位置等参数来举例。

1. 将寄存器 BR [1] [BR_INI_POS] 、BR [1] [BR_INI_REV] 、BR [1] [BR_INI_SPEED] 、BR [1] [BR_INI_ACCEL] 配成要更改成的数据。不需要更改的，可以不配置。
2. 按照需要对寄存器 BR [1] [BR_CTRL [INI_POS_CMD_MSK]] 配置。
3. 将寄存器 BR [1] [BR_TRG_SPEED_THR] 配置成速度门限。
4. 按照需要对寄存器 BR [1] [BR_TRG_SPEED_CFG] 配置。
5. 按照所选择的触发类型对寄存器 BR [1] [BR_CTRL [INI_DELTA_POS_TRG_TYPE]] 配置为 6。
6. 将寄存器 CR [INI_BR1_POS_REQ] 配成 1。

这样，在 1 号预测器内部观测到的速度满足速度门限时，就可以对 1 号预测器的位置等参量做出修改了。

再以对 0 号预测器的位置触发来配置 0 号预测器的位置等参数来举例。

1. 将寄存器 BR [0] [BR_INI_POS] 、BR [0] [BR_INI_REV] 、BR [0] [BR_INI_SPEED] 、BR [0] [BR_INI_ACCEL] 配成要更改成的数据。不需要更改的，可以不配置。
2. 按照需要对寄存器 BR [0] [BR_CTRL [INI_POS_CMD_MSK]] 配置。
3. 将寄存器 BR [0] [BR_TRG_POS_THR] , BR [0] [BR_TRG_REV_THR] 配置成位置门限。
4. 按照需要对寄存器 BR [0] [BR_TRG_POS_CFG] 配置。
5. 按照所选择的触发类型对寄存器 BR [0] [BR_CTRL [INI_DELTA_POS_TRG_TYPE]] 配置为 5。
6. 将寄存器 CR [INI_BR0_POS_REQ] 配成 1。

这样，在 0 号预测器内部观测到的位置满足位置门限时，就可以对 0 号预测器的位置等参量做出修改了。

触发激励可以是 OUTTRGR[0]、或 OUTTRGR[1]。

OUTTRGR[n] 是一个节拍的信号，只能由第 n 号预测器产生，产生方式是 =（预测器自身的位置触发信号的有效边沿 | 速度触发信号的有效边沿）。

38.6.6 配置开环转闭环

38.6.6.1 从开环到闭环

将第 n 号预测器的寄存器 BR [n] [BR_CTRL [OPEN_LOOP_MODE]] 清为 0，就是选择了闭环模式。而该预测器真正进入闭环模式，要在跟踪器已经对第一个输入的采样点计算过后。

38.6.6.2 从闭环到开环

将第 n 号预测器的寄存器 BR [n] [BR_CTRL [OPEN_LOOP_MODE]] 置为 1，就是选择了开环模式。不过之前需要先停止该预测器，然后将相应的位置和位置增量等相关寄存器配好。之后重新使能该预测器。

38.6.7 配置多重 P、I、A 参数

为了能在收敛速度和收敛精度之间做更灵活的折中，本模块也支持 4 重 P、I、A 参数的设置。其中，

- 第 0 号参数系列，收敛速度较快，但收敛误差较大，应用于跟踪误差最大的情况；
- 第 1 号参数系列，收敛速度较 0 号参数系列慢，但收敛误差比 0 号参数系列小，应用于跟踪误差次大的情况；
- 第 2 号参数系列，收敛速度较 1 号参数系列慢，但收敛误差比 1 号参数系列小，应用于跟踪误差更小一些的情况；
- 第 3 号参数系列，收敛速度最慢，但收敛误差也最好，应用于跟踪误差最小的情况。

对于 n=0,1,2 的第 n 号参数系列，其误差门限为寄存器 COEF_TRG_CFG [n] [ERR_THR] ，至少停留的采样点数为寄存器 COEF_TRG_CFG [n] [TIME] ，采用的参数为 COEF_TRG_CFG [n] [P] 、COEF_TRG_CFG [n] [I] 、COEF_TRG_CFG [n] [A] 。随着 n 的增加，COEF_TRG_CFG [n] [ERR_THR] 越来越小。

对于 n=3 的第 3 号参数系列，没有误差门限，也没有至少停留的采样点数，采用的参数就是 INI_PCOEF 、INI_ICOEF 、INI_ACOEF。

这样，就可以在跟踪误差比较大时，自动选择收敛速度快但精度差的参数；然后依据收敛状况，自动地逐步选择更小收敛误差的参数系列。

配置时，先要配置好上述相关寄存器，然后配置 CR[MS_COEF_EN]=1。这样就可以使能多重 P、I、A 参数的自动切换及跟踪了。

38.6.8 MMC 使能流程

1. 将寄存器 SYSCLK_FREQ 配成所需值。比方说，200MHz 的主频，200e+6 Hz = 0x0BEBC200 Hz，则配置 SYSCLK_FREQ = 0xBEBC200。
2. 将寄存器 SYSCLK_PERIOD 配成所需值。比方说，200MHz 的主频，对应时钟周期为 5e-9 秒，round(5e-9*（2^{24}）* （2^{20}）) = 0x15799，则配置 SYSCLK_PERIOD = 0x15799。
3. 将寄存器 OOSYNC_THETA_THR 配成所需值。比方说，如果能接受最大（0.5/270）圆周的跟踪误差为不失步的跟踪误差，则该寄存器配置成（round (0.5 / 270 * 2^{32}) = ）0x795CEB。
4. 将寄存器 DISCRETECFG0 配置为 0。
5. 将寄存器 DISCRETECFG1 配成所需值。比方说，如果能接受最大（1/270）圆周的跟踪误差为相邻 2 次跟踪值之间的最大误差（即预测值将以这个误差值为限，超限的预测值会锁死在这个误差值上），则该寄存器配置成（round (1 / 270 * 2^{32}) = ）0xF2B9D6。
6. 将寄存器 CONTCFG0 配置为 0x0100_0000。则，如果本次的外界输入的在一个圆周里的相位比上一次的外界输入的相位在一个圆周里的相位（按照无符号数计算）大于 220 度角，则认为相角在反转；否则，即是正转。
7. 如果需要配置位置寄存器，则按照小节 38.6.2（除了最后的 CR [INI_POS_REQ] 比特位暂时不配），对位置寄存器进行配置。如果不配置这一步，则本模块会自动从外界输入中提取相关信息。
8. 按照小节 38.6.4（除了最后的 CR [INI_COEFS_REQ] 比特位暂时不配），对 P、I、A 系数寄存器进行配置
9. 将寄存器 CR [INI_POS_REQ]（如果配置了位置寄存器）, CR [INI_COEFS_REQ] 同一个配置命令里配为 1。
10. 反复读取寄存器 STA [INI_COEFS_CMD_DONE] 直至其为 1。然后向这位写 1 清成 0
11. 按照所选择的触发类型对寄存器 BR [0] [BR_CTRL [INI_DELTA_POS_TRG_TYPE]] 、和/或 BR [1] [BR_CTRL [INI_DELTA_POS_TRG_TYPE]] 进行配置，并置 BR [0] [BR_CTRL [BR_EN]] = 1、和/或 BR [1] [BR_CTRL [BR_EN]] = 1 来使能相应预测器模块。
12. 将寄存器 CR [MOD_EN] 配成 1，则本模块就开始工作了。

38.7 MMC 使用注意事项

有如下注意事项：

- 预测器产生的时间戳，只有在跟踪器产生了至少一次跟踪事件后，才会有效。

39 可编程逻辑单元 PLB

本章节介绍可编程逻辑单元 PLB 的主要功能和特性。

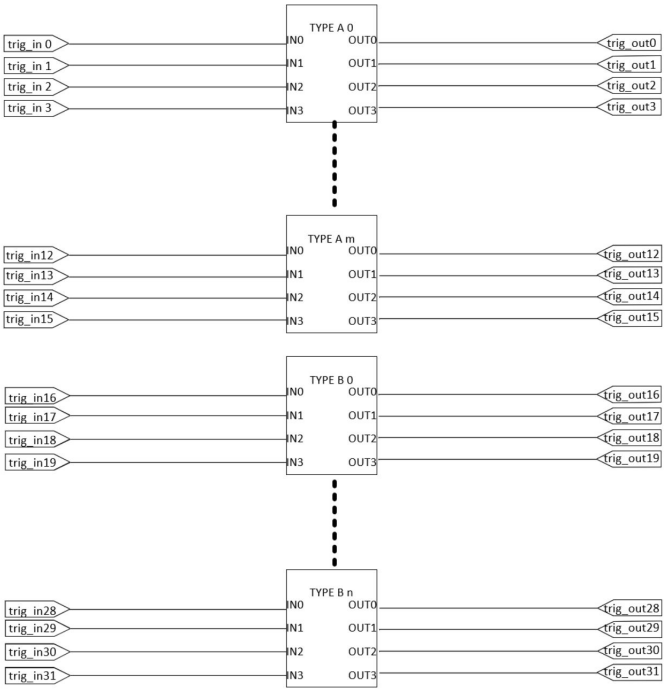
39.1 特性总结

本章节介绍可编程逻辑单元 PLB 的主要特性：

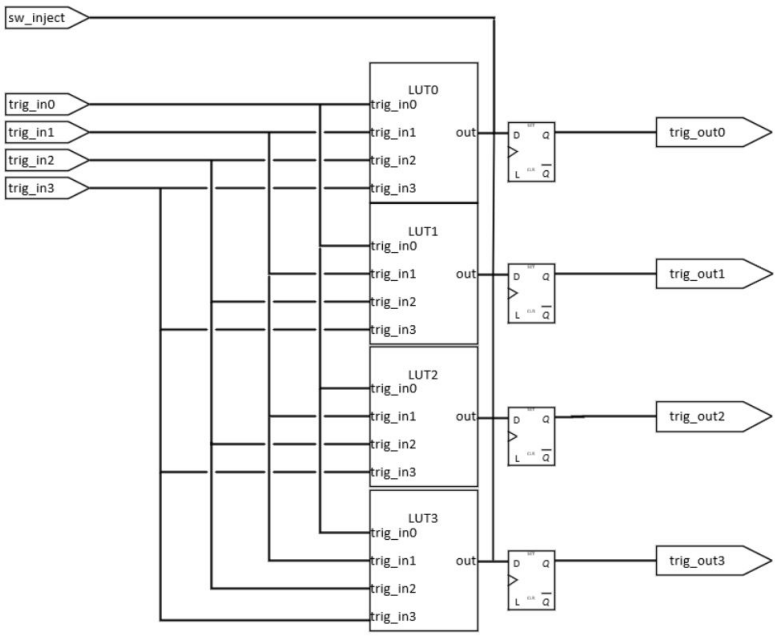
- 两种可编程类型
- TYPE_A 可实现组合逻辑 + 时序逻辑编程
- TYPE_B 可实现计数、位移、滤波等复杂操作

39.2 架构图

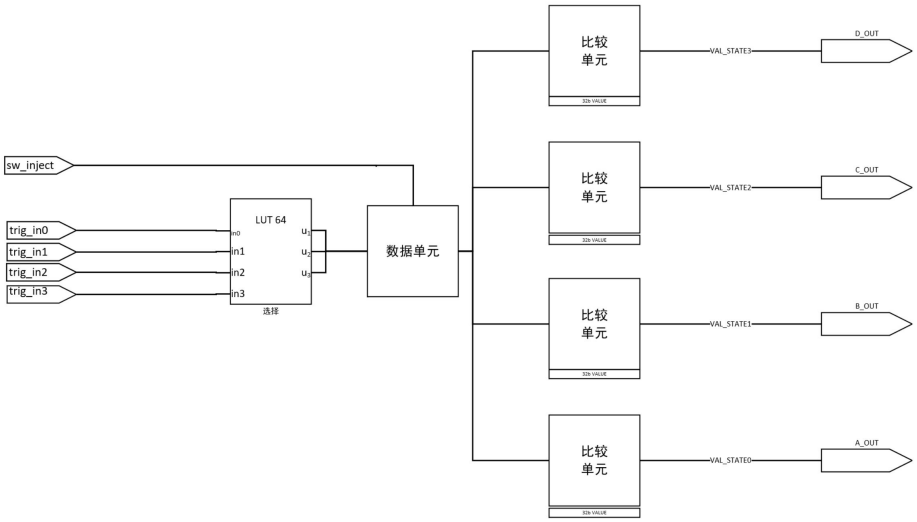
PLB 的输入输出均连接在 TRIG_MUX 中。整体结构如下图所示：



TYPE A 结构如下图所示：



TYPE B 结构如下图所示：



39.3 功能描述

39.3.1 TYEP A 功能描述

PLB 内置 4 个 TYPE A 通道，每个 TYPE A 通道有 4 个 trig_in 输入和 4 个 trig_out 输出。每个 trig_out 输出对应一个查找表单元。

这 4 个查找表均采用 4bit trig_in 做为索引，查找表的内容可以视为 4bit trig_in 的真值表。这 4 个查找表使用同样的 trig_in，不同 TYPE A 模块的 trig_in 顺序增加，比如 TYPE A 0 使用 trig_in[3:0], TYPE A 1 使用 trig_in[7:4]。

查找表的输出会用寄存器输出到 trig_out 上。

软件可以配置 TYPE_A[SW_INJECT] 直接注入到 trig_out 上，该注入值会保持一个系统时钟周期，之后将继续由 trig_in 和查找表决定 trig_out 的值。

39.3.2 TYEP B 功能描述

PLB 内置 4 个 TYPE B 通道，每个 TYPE B 通道有 4 个 trig_in 输入和 4 个 trig_out 输出。

每个 TYPE B 通道有一个 64bit 的查找表，该查找表由 TYPE_B[LUT][0] 和 TYPE_B[LUT][1] 拼接为一个 64bit 的双字，lut0 占 bit31：0，lut1 占 bit63：32，然后将 64bit 看成 16 个 4bit 组成 slice 单元，slice0 对应 bit3：0，slice1 对应 bit7：4，依此类推，slice15 对应 bit63：60，每个 type B 通道分配了 4 个 trig in，这 4bit 的 trig in 将做为 slice 的索引，这样每个通道的会产生对应 4bit 的操作选择，该操作选择会决定 type b 数据单元的操作。

0：数据单元维持上周期数值。

1：数据单元会把本通道的 cmp[0] 寄存器的数值做为下一周期的数值。

2：数据单元会把本通道的 cmp[1] 寄存器的数值做为下一周期的数值。

3：数据单元会把本通道的 cmp[2] 寄存器的数值做为下一周期的数值。

4：MODE.OPT_SEL=0 时，数据单元下一周期数值为当前值 +1；MODE.OPT_SEL=1 时，数据单元下一周期数值为当前值左移 1 位，低位补 0.

5：MODE.OPT_SEL=0 时，数据单元下一周期数值为当前值 +2；MODE.OPT_SEL=1 时，数据单元下一周期数值为当前值左移 1 位，低位补 1.

6：MODE.OPT_SEL=0 时，数据单元下一周期数值为当前值-1；MODE.OPT_SEL=1 时，数据单元下一周期数值为当前值右移 1 位，高位补 0.

7：MODE.OPT_SEL=0 时，数据单元下一周期数值为当前值-2；MODE.OPT_SEL=1 时，数据单元下一周期数值为当前值右移 1 位，高位补 1.

其他值：数据单元数值保持不变。

每个 TYPE B 通道中有 4 个比较单元，每个比较单元的输出对应相应的 trig_out。

比较单元的运算模式由 TYPE_B[MODE].OUTn_SEL 决定。

0：0

1：1

2：数据单元数据是否大于 TYPE_B[CMP] 的数值，大于则输出 1，不大于则输出 0.

3：数据单元数据是否小于 TYPE_B[CMP] 的数值，小于则输出 1，不小于则输出 0.

4：数据单元数据是否等于 TYPE_B[CMP] 的数值，等于则输出 1，不等于则输出 0.

5：数据单元数据是否不等于 TYPE_B[CMP] 的数值，不等于则输出 1，等于则输出 0.

6：数据单元数据是否大于等于 TYPE_B[CMP] 的数值，大于等于则输出 1，小于则输出 0.

7：数据单元数据是否小于等于 TYPE_B[CMP] 的数值，小于等于则输出 1，大于则输出 0.

8：0

9：1

10：把数据单元数据中 TYPE_B[CMP] 对应 bit 为 1 的 bit 位进行与操作。

11：把数据单元数据中 TYPE_B[CMP] 对应 bit 为 1 的 bit 位进行或操作。

12：把数据单元数据中 TYPE_B[CMP] 对应 bit 为 1 的 bit 位进行异或操作。

13：把数据单元数据中 TYPE_B[CMP] 对应 bit 为 1 的 bit 位进行与非操作。

14：把数据单元数据中 TYPE_B[CMP] 对应 bit 为 1 的 bit 位进行或非操作。

15：把数据单元数据中 TYPE_B[CMP] 对应 bit 为 1 的 bit 位进行同或操作。

39.4 PLB 寄存器列表

PLB 的寄存器列表如下：

PLB base address: 0xF0324000

地址偏移	名称	描述	复位值
0x0000	TYPE_A[0][LOOKUP_TABLE][0]	TYPE A 通道 0 查找表 0	0x00000000
0x0004	TYPE_A[0][LOOKUP_TABLE][1]	TYPE A 通道 0 查找表 1	0x00000000
0x0008	TYPE_A[0][LOOKUP_TABLE][2]	TYPE A 通道 0 查找表 2	0x00000000
0x000C	TYPE_A[0][LOOKUP_TABLE][3]	TYPE A 通道 0 查找表 3	0x00000000
0x0010	TYPE_A[0][SW_INJECT]	TYPE A 通道 0 软件注入	0x00000000
0x0020	TYPE_A[1][LOOKUP_TABLE][0]	TYPE A 通道 1 查找表 0	0x00000000
0x0024	TYPE_A[1][LOOKUP_TABLE][1]	TYPE A 通道 1 查找表 1	0x00000000
0x0028	TYPE_A[1][LOOKUP_TABLE][2]	TYPE A 通道 1 查找表 2	0x00000000
0x002C	TYPE_A[1][LOOKUP_TABLE][3]	TYPE A 通道 1 查找表 3	0x00000000
0x0030	TYPE_A[1][SW_INJECT]	TYPE A 通道 1 软件注入	0x00000000
0x0040	TYPE_A[2][LOOKUP_TABLE][0]	TYPE A 通道 2 查找表 0	0x00000000
0x0044	TYPE_A[2][LOOKUP_TABLE][1]	TYPE A 通道 2 查找表 1	0x00000000
0x0048	TYPE_A[2][LOOKUP_TABLE][2]	TYPE A 通道 2 查找表 2	0x00000000
0x004C	TYPE_A[2][LOOKUP_TABLE][3]	TYPE A 通道 2 查找表 3	0x00000000
0x0050	TYPE_A[2][SW_INJECT]	TYPE A 通道 2 软件注入	0x00000000
0x0060	TYPE_A[3][LOOKUP_TABLE][0]	TYPE A 通道 3 查找表 0	0x00000000
0x0064	TYPE_A[3][LOOKUP_TABLE][1]	TYPE A 通道 3 查找表 1	0x00000000
0x0068	TYPE_A[3][LOOKUP_TABLE][2]	TYPE A 通道 3 查找表 2	0x00000000
0x006C	TYPE_A[3][LOOKUP_TABLE][3]	TYPE A 通道 3 查找表 3	0x00000000
0x0070	TYPE_A[3][SW_INJECT]	TYPE A 通道 3 软件注入	0x00000000
0x0400	TYPE_B[0][LUT][0]	TYPE B 通道 0 查找表 0	0x00000000
0x0404	TYPE_B[0][LUT][1]	TYPE B 通道 0 查找表 1	0x00000000
0x0408	TYPE_B[0][CMP][0]	TYPE B 通道 0 比较单元的 cmp0	0x00000000
0x040C	TYPE_B[0][CMP][1]	TYPE B 通道 0 比较单元的 cmp1	0x00000000
0x0410	TYPE_B[0][CMP][2]	TYPE B 通道 0 比较单元的 cmp2	0x00000000
0x0414	TYPE_B[0][CMP][3]	TYPE B 通道 0 比较单元的 cmp3	0x00000000
0x0418	TYPE_B[0][MODE]	TYPE B 通道 0 的模式控制寄存器	0x00000000
0x041C	TYPE_B[0][SW_INJECT]	TYPE B 通道 0 的数据单元软件注入值	0x00000000

地址偏移	名称	描述	复位值
0x0420	TYPE_B[1][LUT][0]	TYPE B 通道 1 查找表 0	0x00000000
0x0424	TYPE_B[1][LUT][1]	TYPE B 通道 1 查找表 1	0x00000000
0x0428	TYPE_B[1][CMP][0]	TYPE B 通道 1 比较单元的 cmp0	0x00000000
0x042C	TYPE_B[1][CMP][1]	TYPE B 通道 1 比较单元的 cmp1	0x00000000
0x0430	TYPE_B[1][CMP][2]	TYPE B 通道 1 比较单元的 cmp2	0x00000000
0x0434	TYPE_B[1][CMP][3]	TYPE B 通道 1 比较单元的 cmp3	0x00000000
0x0438	TYPE_B[1][MODE]	TYPE B 通道 1 的模式控制寄存器	0x00000000
0x043C	TYPE_B[1][SW_INJECT]	TYPE B 通道 1 的数据单元软件注入值	0x00000000
0x0440	TYPE_B[2][LUT][0]	TYPE B 通道 2 查找表 0	0x00000000
0x0444	TYPE_B[2][LUT][1]	TYPE B 通道 2 查找表 1	0x00000000
0x0448	TYPE_B[2][CMP][0]	TYPE B 通道 2 比较单元的 cmp0	0x00000000
0x044C	TYPE_B[2][CMP][1]	TYPE B 通道 2 比较单元的 cmp1	0x00000000
0x0450	TYPE_B[2][CMP][2]	TYPE B 通道 2 比较单元的 cmp2	0x00000000
0x0454	TYPE_B[2][CMP][3]	TYPE B 通道 2 比较单元的 cmp3	0x00000000
0x0458	TYPE_B[2][MODE]	TYPE B 通道 2 的模式控制寄存器	0x00000000
0x045C	TYPE_B[2][SW_INJECT]	TYPE B 通道 2 的数据单元软件注入值	0x00000000
0x0460	TYPE_B[3][LUT][0]	TYPE B 通道 3 查找表 0	0x00000000
0x0464	TYPE_B[3][LUT][1]	TYPE B 通道 3 查找表 1	0x00000000
0x0468	TYPE_B[3][CMP][0]	TYPE B 通道 3 比较单元的 cmp0	0x00000000
0x046C	TYPE_B[3][CMP][1]	TYPE B 通道 3 比较单元的 cmp1	0x00000000
0x0470	TYPE_B[3][CMP][2]	TYPE B 通道 3 比较单元的 cmp2	0x00000000
0x0474	TYPE_B[3][CMP][3]	TYPE B 通道 3 比较单元的 cmp3	0x00000000
0x0478	TYPE_B[3][MODE]	TYPE B 通道 3 的模式控制寄存器	0x00000000

地址偏移	名称	描述	复位值
0x047C	TYPE_B[3][SW_INJECT]	TYPE B 通道 3 的数据单元软件注入值	0x00000000

表 183: PLB 寄存器列表

39.5 PLB 寄存器详细信息

PLB 的寄存器详细说明如下：

39.5.1 TYPE_A[LOOKUP_TABLE] (0x0 + 0x20 * n + 0x4 * m)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																LOOKUP_TABLE															
N/A																RW															
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

TYPE_A[LOOKUP_TABLE] [31:0]

位域	名称	描述
15-0	LOOKUP_TABLE	查找表采用 4bit trig_in 做为索引，查找表的内容可以视为 4bit trig_in 的真值表。每个 TYPEA 通道中有 4 个查找表，这 4 个查找表使用同样的 trig_in，不同通道的 trig_in 顺序增加，比如通道 0 使用 trig_in[3:0], 通道 1 使用 trig_in[7:4]。

TYPE_A[LOOKUP_TABLE] 位域

39.5.2 TYPE_A[SW_INJECT] (0x10 + 0x20 * n)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																												SW_INJECT			
N/A																												RW			
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0

TYPE_A[SW_INJECT] [31:0]

位域	名称	描述
3-0	SW_INJECT	软件可以通过写此寄存器向对应 TYPE A 通道注入一个 cycle 的值。一个 cycle 后，该通道的输出值将继续由 trig_in 和查找表决定。

TYPE_A[SW_INJECT] 位域

39.5.3 TYPE_B[LUT] (0x400 + 0x20 * n + 0x4 * m)

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

TYPE_B[LUT] [31:0]

位域	名称	描述
31-0	LOOKUP_TABLE	lut0 和 lut1 会拼接为一个 64bit 的双字，lut0 占 bit31：0，lut1 占 bit63：32，然后将 64bit 看成 16 个 4bit 组成 slice 单元，slice0 对应 bit3：0，slice1 对应 bit7：4，依此类推，slice15 对应 bit63：60，每个 type B 通道分配了 4 个 trig in，这 4bit 的 trig in 将做为 slice 的索引，这样每个通道的会产生对应 4bit 的操作选择，该操作选择会决定 type b 数据单元的操作. 0：数据单元维持上周期数值。 1：数据单元会把本通道的 cmp[0] 寄存器的数值做为下一周期的数值。 2：数据单元会把本通道的 cmp[1] 寄存器的数值做为下一周期的数值。 3：数据单元会把本通道的 cmp[2] 寄存器的数值做为下一周期的数值。 4：mode.opt_sel=0 时，数据单元下一周期数值为当前值 +1；mode.opt_sel=1 时，数据单元下一周期数值为当前值左移 1 位，低位补 0. 5：mode.opt_sel=0 时，数据单元下一周期数值为当前值 +2；mode.opt_sel=1 时，数据单元下一周期数值为当前值左移 1 位，低位补 1. 6：mode.opt_sel=0 时，数据单元下一周期数值为当前值-1；mode.opt_sel=1 时，数据单元下一周期数值为当前值右移 1 位，高位补 0. 7：mode.opt_sel=0 时，数据单元下一周期数值为当前值-2；mode.opt_sel=1 时，数据单元下一周期数值为当前值右移 1 位，高位补 1. 其他值：数据单元数值保持不变。

TYPE_B[LUT] 位域

39.5.4 TYPE_B[CMP] (0x408 + 0x20 * n + 0x4 * m)

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

TYPE_B[CMP] [31:0]

位域	名称	描述
31-0	CMP_VALUE	cmp 数值，用于数据单元的数据更新和比较单元的运算。

位域	名称	描述

TYPE_B[CMP] 位域

39.5.5 TYPE_B[MODE] (0x418 + 0x20 * n)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD															OPT_SEL	OUT3_SEL				OUT2_SEL				OUT1_SEL				OUT0_SEL			
N/A															RW	RW				RW				RW				RW			
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

TYPE_B[MODE] [31:0]

位域	名称	描述
16	OPT_SEL	数据单元的数值更新选择，详见 TYPE B lut 寄存器介绍。
15-12	OUT3_SEL	本通道 trig out3 对应的输出模式： 0：0 1：1 2：数据单元数据是否大于 cmp3 的数值，大于则输出 1，不大于则输出 0. 3：数据单元数据是否小于 cmp3 的数值，小于则输出 1，不小于则输出 0. 4：数据单元数据是否等于 cmp3 的数值，等于则输出 1，不等于则输出 0. 5：数据单元数据是否不等于 cmp3 的数值，不等于则输出 1，等于则输出 0. 6：数据单元数据是否大于等于 cmp3 的数值，大于等于则输出 1，小于则输出 0. 7：数据单元数据是否小于等于 cmp3 的数值，小于等于则输出 1，大于则输出 0. 8：0 9：1 10：把数据单元数据中 cmp3 对应 bit 为 1 的 bit 位进行与操作。 11：把数据单元数据中 cmp3 对应 bit 为 1 的 bit 位进行或操作。 12：把数据单元数据中 cmp3 对应 bit 为 1 的 bit 位进行异或操作。 13：把数据单元数据中 cmp3 对应 bit 为 1 的 bit 位进行与非操作。 14：把数据单元数据中 cmp3 对应 bit 为 1 的 bit 位进行或非操作。 15：把数据单元数据中 cmp3 对应 bit 为 1 的 bit 位进行同或操作。

位域	名称	描述
11-8	OUT2_SEL	本通道 trig out2 对应的输出模式： 0：0 1：1 2：数据单元数据是否大于 cmp2 的数值，大于则输出 1，不大于则输出 0. 3：数据单元数据是否小于 cmp2 的数值，小于则输出 1，不小于则输出 0. 4：数据单元数据是否等于 cmp2 的数值，等于则输出 1，不等于则输出 0. 5：数据单元数据是否不等于 cmp2 的数值，不等于则输出 1，等于则输出 0. 6：数据单元数据是否大于等于 cmp2 的数值，大于等于则输出 1，小于则输出 0. 7：数据单元数据是否小于等于 cmp2 的数值，小于等于则输出 1，大于则输出 0. 8：0 9：1 10：把数据单元数据中 cmp2 对应 bit 为 1 的 bit 位进行与操作。 11：把数据单元数据中 cmp2 对应 bit 为 1 的 bit 位进行或操作。 12：把数据单元数据中 cmp2 对应 bit 为 1 的 bit 位进行异或操作。 13：把数据单元数据中 cmp2 对应 bit 为 1 的 bit 位进行与非操作。 14：把数据单元数据中 cmp2 对应 bit 为 1 的 bit 位进行或非操作。 15：把数据单元数据中 cmp2 对应 bit 为 1 的 bit 位进行同或操作。

位域	名称	描述
7-4	OUT1_SEL	本通道 trig out1 对应的输出模式： 0：0 1：1 2：数据单元数据是否大于 cmp1 的数值，大于则输出 1，不大于则输出 0. 3：数据单元数据是否小于 cmp1 的数值，小于则输出 1，不小于则输出 0. 4：数据单元数据是否等于 cmp1 的数值，等于则输出 1，不等于则输出 0. 5：数据单元数据是否不等于 cmp1 的数值，不等于则输出 1，等于则输出 0. 6：数据单元数据是否大于等于 cmp1 的数值，大于等于则输出 1，小于则输出 0. 7：数据单元数据是否小于等于 cmp1 的数值，小于等于则输出 1，大于则输出 0. 8：0 9：1 10：把数据单元数据中 cmp1 对应 bit 为 1 的 bit 位进行与操作。 11：把数据单元数据中 cmp1 对应 bit 为 1 的 bit 位进行或操作。 12：把数据单元数据中 cmp1 对应 bit 为 1 的 bit 位进行异或操作。 13：把数据单元数据中 cmp1 对应 bit 为 1 的 bit 位进行与非操作。 14：把数据单元数据中 cmp1 对应 bit 为 1 的 bit 位进行或非操作。 15：把数据单元数据中 cmp1 对应 bit 为 1 的 bit 位进行同或操作。

位域	名称	描述
3-0	OUT0_SEL	本通道 trig out0 对应的输出模式： 0：0 1：1 2：数据单元数据是否大于 cmp0 的数值，大于则输出 1，不大于则输出 0. 3：数据单元数据是否小于 cmp0 的数值，小于则输出 1，不小于则输出 0. 4：数据单元数据是否等于 cmp0 的数值，等于则输出 1，不等于则输出 0. 5：数据单元数据是否不等于 cmp0 的数值，不等于则输出 1，等于则输出 0. 6：数据单元数据是否大于等于 cmp0 的数值，大于等于则输出 1，小于则输出 0. 7：数据单元数据是否小于等于 cmp0 的数值，小于等于则输出 1，大于则输出 0. 8：0 9：1 10：把数据单元数据中 cmp0 对应 bit 为 1 的 bit 位进行与操作。 11：把数据单元数据中 cmp0 对应 bit 为 1 的 bit 位进行或操作。 12：把数据单元数据中 cmp0 对应 bit 为 1 的 bit 位进行异或操作。 13：把数据单元数据中 cmp0 对应 bit 为 1 的 bit 位进行与非操作。 14：把数据单元数据中 cmp0 对应 bit 为 1 的 bit 位进行或非操作。 15：把数据单元数据中 cmp0 对应 bit 为 1 的 bit 位进行同或操作。

TYPE_B[MODE] 位域

39.5.6 TYPE_B[SW_INJECT] (0x41C + 0x20 * n)

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

TYPE_B[SW_INJECT] [31:0]

位域	名称	描述
31-0	SOFTWARE_INJECT	当写下此寄存器值时，数据单元内的数据会在下一个周期更新为此寄存器值。下下周期时，继续恢复到之前的更新机制。

位域	名称	描述

TYPE_B[SW_INJECT] 位域

40 旋转编码器接口 RDC

本章节介绍旋转编码器接口 RDC 的主要功能和特性。

40.1 概述

RDC 模块可以生产励磁信号，并对旋转变压器的感应信号进行累加预处理。目前支持的特性如下：

功能	主要特性
励磁信号产生	· 单个 DAC 输出 · 两路差分 DAC 输出 · PWM 输出，NMOS+PMOS · PWM 输出，PMOS+NMOS · 支持幅度和偏置调整 · 支持死区配置
ADC 采样控制	· 一个 ADC 交替采样 · 两个 ADC 同步采样 · 连续采样 · 每个励磁信号周期对齐高点采样一次 · 对齐 PWM 低电平中点采样一次 · 对齐 PWM 高电平中点采样一次 · 对齐 PWM 高/低电平中点采样一次 · ADC 输入的最大值和最小值记录
整流控制	· 参考点基于励磁信号 0/90/180/270 相位 · 参考点支持外部同步信号输入 · ADC 采样数据的过 0 点延时测量 · ADC 采样数据的过 0 点抖动过滤
位置预处理	· 允许设置累加周期数 · 允许累加周期数的动态配置 · 允许 ADC 采样结果向 QEI 的透传 · 允许 ADC 数据累加结果的软件读取 · 累加结果最大值记录 · 累加结果最小值记录 · 累加模值超限告警

表 184: RDC 主要特性

40.2 RDC 架构图

本模块的结构图如图 50。

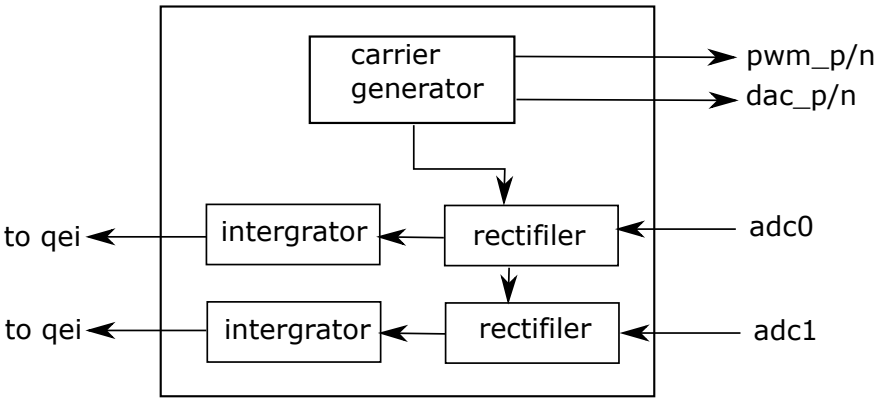


图 50: RDC 结构框图

RDC 模块中的 carrier generator 为励磁信号发生器，可以产生频率为 10k-20kHz 的正弦波，正弦波可以以两路 PWM 信号的形式输出，也可以通过 DAC 的转换为模拟信号后输出。rectifiler 模块可实现从 adc 采样数据的整流，整流后的数据在 integrator 模块中根据用户的配置周期性的将数据进行累加，累加的结果可理解为电机旋转运动引起的对励磁信号幅度调制的包络，该结果可输出到后级的 QEI 模块中再进行进一步的处理。

40.3 管脚说明

表 185: rdc 管脚说明

管脚	方向	功能描述
pwm_p	输出	输出 pwm 信号
pwm_n	输出	输出 pwm 互补的信号
dac_p	输出	输出到 DAC 的信号
dac_n	输出	输出到 DAC 的信号
adc_0	输入	ADC 转换后的数据
adc_1	输入	ADC 转换后的数据

40.4 功能说明

40.4.1 励磁生成

RDC 中内置一个三角函数表，其深度为 256，可根据用户配置生成一定频率和一定点数的正弦波。假设正弦波的频率为 f_sin，工作时钟为 f_soc，则正弦波的频率符合如下公式：

$$f\_sin = \frac{f\_soc}{a * 2^b * 4}$$

其中，a 为寄存器 exc_timming.smp_rate 的值，b 为寄存器 exc_timming.smp_num 的值。

励磁信号的触发有两种方式,分别是软件触发方式和硬件触发方式。对于软件触发方式,软件将寄存器 rdc_ctl.exc_start 为 1 即可触发励磁信号的产生，该寄存器写 1 后可自动回 0。对于硬件触发方式，励磁信号的触发受到 trig_mux 模块输出信号的控制，当该信号发生跳变时，在经过一定的延时后就可以触发励磁信号的产生，该延时受到寄存器 exc_sync_dly.delay 的控制，利用这个延时可以调整励磁信号的相位。

励磁信号可以通过 DAC 直接以模拟信号的形式输出，也可以通过 PWM 接口以数字信号的形式输出，数字

信号经过外部的 power stage 电路整形后就可恢复为正弦波信号。

当励磁信号通过 DAC 输出时，正弦波的摆幅和偏置值都是可以调整的，如下式所示：

$$v\_dac = \left(\frac{\sin\theta * 4095 * 4096 * m}{2^d} + s\right) * 256$$

其中,m 为寄存器 exc_scaling.amp_man 的值;d 为寄存器 exc_scaling.ampl_exp 的值;s 为寄存器 exc_offset.ampl_offset 的值；v_dac 为输出到 DAC 的数据，DAC 根据该数据将数字信号转换为模拟电压。

励磁信号通过 DAC 输出时，还可以有一个反相的信号，该信号的幅值和偏置是与第一路 DAC 的输出相同的，指示相位是相反的，如下式所示：

$$v\_dac = \left(\frac{-\sin\theta * 4095 * 4096 * m}{2^d} + s\right) * 256$$

当励磁信号通过 PWM 输出时，利用 PWM 波在一个 PWM 周期内的占空比来模拟三角函数的值，默认情况下一个 PWM 的周期与 DAC 输出的两个点之间的时间差是一致的，PWM 的周期也可以变得更宽，如下式所示：

$$f\_pwm = \frac{f\_soc}{a * (p + 1)}$$

其中，p 为寄存器 exc_timming.pwm_prd 的值。

因 PWM 输出波形的占空比对应着三角函数的数值，所以其摆幅和偏置值也是可以调整的，如下式所示：

$$d\_pwm = \frac{\sin\theta * pm}{2^p d} + \frac{ps}{a}$$

其中，pm 为寄存器 pwm_scaling.ampl_man 的值；pd 为寄存器 pwm_scaling.ampl_offset 的值；ps 为寄存器 pwm_offset.ampl_offset 的值。

PWM 的输出信号包括 PWM_P 和 PWM_N，PWM_N 是 PWM_P 的反相输出。另外 PWM 的输出时支持死区配置的，默认情况下死区是关闭的，可通过寄存器 pwm_dz 寄存器配置死区的功能。

总体来说，使能输出励磁的步骤如下：

1. 根据系统要求，配置励磁信号时序控制寄存器 exc_timming 为合适的值；
2. 根据系统要求，配置励磁信号幅值控制寄存器 exc_scaling、pwm_scaling、exc_offset 和 pwm_offset 为合适的值；
3. 配置 rdc_ctl.exc_en 为 1；
4. 配置软件或硬件触发方式，硬件触发时配置 exc_sync_dly 寄存器，等待触发发生后，励磁信号就可以输出了；

伴随着励磁信号的输出，RDC 模块还可以生成相应的 trigger 输出信号，用于指导 ADC 采样或用于其他事件。RDC 支持的 trigger 输出包括：

- 在 PWM 高电平的中间输出，并支持寄存器 trig_out1_cfg 控制的前后 offset 调节；

- 在 PWM 低电平的中间输出，并支持寄存器 trig_out1_cfg 控制的前后 offset 调节；
- 在 sin 函数最大值输出；
- 在 sin 函数最小值输出；

以上四种输出格式，支持任何形式的组合。

40.4.2 延时测量

励磁信号经过旋转编码器感应后，分两路信号经过 ADC 完成模数转换后输入到 RDC 模块，RDC 模块可辅助完成励磁信号与感应信号的相位延时的测量，两路信号的延时测量可独立进行，测量的基本原理如下图所示：

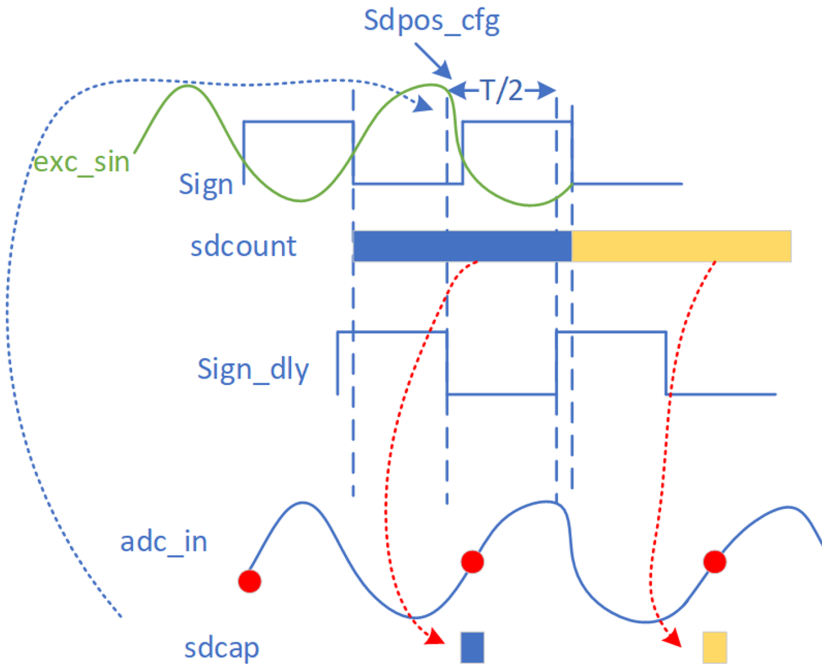


图 51: 延时测量示意图

图中，

- 绿色的 exc_sin 表示输出的励磁信号；
- sign 表示励磁信号的符号位；
- sdcount 为 RDC 内部 32bit 计数器；
- adc_in 为从 ADC 转换得到的数据，也是 sin 函数；
- sdcap 为 RDC 内部 32bit 寄存器 rise_delay_i/q，用来保存 adc_in 过 0 点时 adcount 的值；
- sign_dly 为 RDC 内部生成的延时后的整流信号；

延时测量的过程为：

1. 配置 rdc_ctl.exc_en 和 rdc_ctl.acc_en 为 1，rdc_ctl.rectify_sel 为 0，选择励磁信号作为同步源；
2. 读取寄存器 max_i 和 max_q 的值，获取 ADC 输入的最大值 v_max 和最小值 v_min；
3. 计算 ADC 输入信号的 offset，并将计算结果配置到 thrs_i 和 thrs_q 中，计算公式如下式所示：

$$v\_thrs = \frac{v\_max - v\_min}{2} + v\_min - 0x800000$$

4. 根据需要配置 edg_det_ctl.filter 的值，用来启用采样过 0 点的滤抖功能；
5. 多次读取寄存器 rise_delay_i/q 的值并求平均，该值就对应着 sdcap 的具体数值；
6. 如果延时测量结果超过半个 sin 函数的时钟周期，则需要配置 rdc_ctl.rectify_sel 为 2，并重复以上测量过程；

延时测量完成后，需要将测量得到的延时值配置到寄存器 sync_delay_i/q 中，同时还需要将 sin 函数的周期配置到寄存器 exc_period 中。RDC 内部会重新生成延时后的整流信号，整流信号生成的方法是：

- 当 sdcount 的值与 sync_delay_i/q 的值相等时，整流信号 sign_dly 由 1 变为 0；
- 当 sdcount 的值与 sync_delay_i/q 的值再加上半个 sin 函数时钟周期相等时，整流信号 sign_dly 由 0 变 1；

值得注意的是，RDC 两路 ADC 输入通道的延时测量是可以独立进行的，两路延时的测量结果也是不同的，在实际使用中，应尽量将两路延时的配置值配置为接近的值。

40.4.3 载波累加

RDC 输出到 QEI 的数据是按照一定时间间隔周期性累加后的数据，其累加过程可以用如下计算式表示：

$$acc = \sum \left(\frac{v\_adc}{256} - 0x800000 \right) * sign\_dly$$

式中，acc 为输出到 QEI 的累加结果，v_adc 为 ADC 模数转换得到的值，sign_dly 为 RDC 内部生成的整流信号。

按照一定时间间隔周期性累加的意思是说，从 ADC 的输入数据，每隔一个或多个 sin 函数周期就输出一次累加结果，这个时间的控制是由寄存器 rdc_ctl.acc_len 控制的。一般情况下，当电机转速较高时，需要控制这个时间间隔小一点，以实现对电机位置的快速探测；当电机转速较低时，可以控制这个时间间隔大一点，以实现更准确的位置探测。

RDC 假设 ADC 输入数据是 32 位的无符号数，实际累加完成后结果会变成 40bit 的有符号数，而实际从 RDC 输出的累加结果是 32bit 的，寄存器 acc_scaling.acc_shift 可控制累加的结果是如何输出的。

RDC 输出的相邻累加结果的间隔是比较长的，为了 RDC 和 QEI 间的配合，RDC 在下一累加结果输出前需要给 QEI 一个采样时刻的指示，该指示信号可以在一个累加周期的开始处、中间处或结束处给出，由寄存器 rdc_ctl.ts_sel 控制。

RDC 模块还具备模值异常检测功能，检测的基本原理是先计算两路累加结果的模值，再探测该模值是否在给定的范围内。计算模值 m 的公式如下式所示：

$$m = \left| max(acc\_i, acc\_q) + \frac{min(acc\_i, acc\_q)}{2} \right|$$

式中，acc_i 表示第一路 ADC 输入数据的累计值；acc_q 表示第二路 ADC 输入数据的累加值。

模值的合法范围由寄存器 amp_max 和寄存器 amp_min 控制。

40.4.4 中断

RDC 内部支持的中断源包括如下几种：

- 两路累加结果更新中断；

- 两路 ADC 输入信号上升过 0 点中断；
- 两路 ADC 输入信号下降过 0 点中断；
- 两路延时后整流信号 sign_dly 上升沿采样 ADC 输入值的中断；
- 两路延时后整流信号 sign_dly 下降沿采样 ADC 输入值的中断；
- 两路累加值上溢中断；
- 两路累加值下溢中断；
- 累加结果的模值上溢中断；
- 累加结果的模值下溢中断；

40.5 RDC 寄存器列表

RDC 的寄存器列表如下：

RDC base address: 0xF0320000

地址偏移	名称	描述	复位值
0x0000	RDC_CTL	rdc 控制寄存器	0x00000000
0x0004	ACC_I	i_channel 累加结果	0x00000000
0x0008	ACC_Q	q_channel 累加结果	0x00000000
0x000C	IN_CTL	输入通道选择	0x00000000
0x0010	OUT_CTL	输出通道选择	0x00000000
0x0034	EXC_TIMMING	励磁信号时间参数设置	0x000400C8
0x0038	EXC_SCALING	励磁信号幅度设置	0x00000011
0x003C	EXC_OFFSET	励磁信号偏置设置	0x00800000
0x0040	PWM_SCALING	励磁信号幅度设置	0x00000111
0x0044	PWM_OFFSET	励磁信号偏置设置	0x00000064
0x0048	TRIG_OUT0_CFG	trigger0 输出控制	0x00100019
0x004C	TRIG_OUT1_CFG	trigger1 输出控制	0x0010004B
0x0050	PWM_DZ	pwm 死区控制	0x00000000
0x0054	SYNC_OUT_CTRL	励磁信号同步控制	0x00000000
0x0058	EXC_SYNC_DLY	trigger 输入延时控制，单位为工作时钟 cycle	0x01000001
0x0070	MAX_I	i_channel 输入最大值	0x00000000
0x0074	MIN_I	i_channel 输入最小值	0x00000000
0x0078	MAX_Q	q_channel 输入最大值	0x00000000
0x007C	MIN_Q	q_channel 输入最小值	0x00000000
0x0080	THRS_I	i_channel 沿检测偏置设置	0x00000000
0x0084	THRS_Q	q_channel 边沿检测的偏置设置	0x00000000
0x0088	EDG_DET_CTL	边沿检测控制	0x00000080
0x008C	ACC_SCALING	累加结果的控制	0x00000000
0x0090	EXC_PERIOD	励磁信号周期	0x00001770
0x00A0	SYNC_DELAY_I	同步信号延时设置	0x00000008
0x00A8	RISE_DELAY_I	励磁同步信号与 i_channel 输入上升沿之间的延时值	0x00000000

地址偏移	名称	描述	复位值
0x00AC	FALL_DELAY_I	从励磁同步信号到 i_channel 输入数据下降沿之间的延时值	0x00000000
0x00B0	SAMPLE_RISE_I	延时后的整流信号上升沿采样的输入值	0x00000000
0x00B4	SAMPLE_FALL_I	延时后的整流信号下降沿采样的输入值	0x00000000
0x00B8	ACC_CNT_I	累加数量	0x00000000
0x00BC	SIGN_CNT_I	整流信号与输入符号失配计数	0x00000000
0x00C0	SYNC_DELAY_Q	同步信号延时设置	0x00000008
0x00C8	RISE_DELAY_Q	励磁同步信号与 q_channel 输入上升沿之间的延时值	0x00000000
0x00CC	FALL_DELAY_Q	从励磁同步信号到 q_channel 输入数据下降沿之间的延时值	0x00000000
0x00D0	SAMPLE_RISE_Q	延时后的整流信号上升沿采样的输入值	0x00000000
0x00D4	SAMPLE_FALL_Q	延时后的整流信号下降沿采样的输入值	0x00000000
0x00D8	ACC_CNT_Q	累加数量	0x00000000
0x00DC	SIGN_CNT_Q	整流信号与输入符号失配计数	0x00000000
0x00E0	AMP_MAX	累加幅值最大值	0x01000000
0x00E4	AMP_MIN	累加幅值的最小值	0x00400000
0x00E8	INT_EN	中断 mask 控制	0x00000000
0x00EC	ADC_INT_STATE	中断状态	0x00000000

表 186: RDC 寄存器列表

40.6 RDC 寄存器详细信息

RDC 的寄存器详细说明如下：

40.6.1 RDC_CTL (0x0)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD										TS_SEL		ACC_LEN								RSVD					RECTIFY_SEL			RSVD	ACC_EN	EXC_START	EXC_EN
N/A										RW		RW								N/A					RW			N/A	RW	RW1C	RW
x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	x	x	x	x	x	0	0	0	x	0	0	0

RDC_CTL [31:0]

位域	名称	描述
21-20	TS_SEL	累加结果时戳点控制 0: end of accumulation 1: start of accumulation 2: center of accumulation
19-12	ACC_LEN	累加周期数，支持动态修改 0：1 cycle 1：2 cycles … 255: 256 cycles
6-4	RECTIFY_SEL	选择内部整流信号的参考相移 0: 0 phase of internal exciting signal 1: 90 phase of internal exciting signal 2: 180 phase of internal exciting signal 3: 270 phase of internal exciting signal 4: use value on external pin 5: use invert value on external pin
2	ACC_EN	RDC 累加使能控制 0: disable 1: enable
1	EXC_START	写 1 触发励磁信号的输出，回读固定为 0 0: no effect 1: start excite signal
0	EXC_EN	励磁使能控制 0: disable 1: enable

RDC_CTL 位域

40.6.2 ACC_I (0x4)

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

ACC_I [31:0]

位域	名称	描述
31-0	ACC	i_channel 累加结果, 有符号数

ACC_I 位域

40.6.3 ACC_Q (0x8)

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

ACC_Q [31:0]

位域	名称	描述
31-0	ACC	q_channel 累加结果, 有符号数

ACC_Q 位域

40.6.4 IN_CTL (0xC)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD											PORT_Q_SEL	RSVD			CH_Q_SEL					RSVD			PORT_I_SEL	RSVD			CH_I_SEL				
N/A											RW	N/A			RW					N/A			RW	N/A			RW				
x	x	x	x	x	x	x	x	x	x	x	1	x	x	x	0	0	0	0	0	x	x	x	0	x	x	x	0	0	0	0	0

IN_CTL [31:0]

位域	名称	描述
20	PORT_Q_SEL	q_channel 输入端口选择 0:sel port0 1:sel port1
16-12	CH_Q_SEL	q_channel 输入通道选择 0: channel 0 selected 1: channel 1 selected … 31: channel 31 selected
8	PORT_I_SEL	i_channel 输入端口选择 0:sel port0 1:sel port1
4-0	CH_I_SEL	i_channel 输入通道选择 0: channel 0 selected 1: channel 1 selected … 31: channel 31 selected

IN_CTL 位域

40.6.5 OUT_CTL (0x10)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																			CH_Q_SEL					RSVD			CH_I_SEL				
N/A																			RW					N/A			RW				
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	x	x	x	0	0	0	0	0

OUT_CTL [31:0]

位域	名称	描述
12-8	CH_Q_SEL	q_channel 的输出通道
4-0	CH_I_SEL	i_channel 的输出通道

OUT_CTL 位域

40.6.6 EXC_TIMMING (0x34)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD							SWAP	PWM_PRD				SMP_NUM				SMP_RATE																
N/A							RW	RW				RW				RW																
x	x	x	x	x	x	x	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	1	1	0	0	1	0	0	0

EXC_TIMMING [31:0]

位域	名称	描述
24	SWAP	将 PWM 或 DAC 交叉输出 0: disable swap 1: swap output
23-20	PWM_PRD	Pwm 周期设置，单位是多少个采样点 0：1 sample period 1: 2 sample period ... 15: 16 sample period
19-16	SMP_NUM	每个励磁周期的点数 0: 4 point 1: 8 point … 8: 1024 point

位域	名称	描述
15-0	SMP_RATE	励磁信号采样点周期，单位为工作时钟 cycle 数 0: not allowed 1: 1 cycle 2: 2 cycles … 65535 : 65535 cycles

EXC_TIMMING 位域

40.6.7 EXC_SCALING (0x38)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																								AMP_EXP				AMP_MAN			
N/A																								RW				RW			
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	1	0	0	0	1

EXC_SCALING [31:0]

位域	名称	描述
7-4	AMP_EXP	励磁信号摆幅设置, amplitude = [table value] x man / 2^exp
3-0	AMP_MAN	励磁信号摆幅设置, amplitude = [table value] x man / 2^exp

EXC_SCALING 位域

40.6.8 EXC_OFFSET (0x3C)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD								AMP_OFFSET																							
N/A								RW																							
x	x	x	x	x	x	x	x	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

EXC_OFFSET [31:0]

位域	名称	描述
23-0	AMP_OFFSET	励磁信号偏置值

EXC_OFFSET 位域

40.6.9 PWM_SCALING (0x40)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																		N_POL	P_POL	RSVD			DITHER	AMP_EXP				AMP_MAN			
N/A																		RW	RW	N/A			RW	RW				RW			
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	x	x	x	1	0	0	0	1	0	0	0	1

PWM_SCALING [31:0]

位域	名称	描述
13	N_POL	exc_n 信号输出极性控制 0: high active 1: low active
12	P_POL	exc_p 信号输出极性控制 0: high active 1: low active
8	DITHER	pwm 小数位是否累加 0: disable 1: enable
7-4	AMP_EXP	励磁信号摆幅设置, amplitude = [table value] x man / 2^exp
3-0	AMP_MAN	励磁信号摆幅设置, amplitude = [table value] x man / 2^exp

PWM_SCALING 位域

40.6.10 PWM_OFFSET (0x44)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD								AMP_OFFSET																							
N/A								RW																							
x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	1	0	0

PWM_OFFSET [31:0]

位域	名称	描述
23-0	AMP_OFFSET	励磁信号偏置设置

PWM_OFFSET 位域

40.6.11 TRIG_OUT0_CFG (0x48)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD											ENABLE	LEAD_TIM																			
N/A											RW	RW																			
x	x	x	x	x	x	x	x	x	x	x	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	1

TRIG_OUT0_CFG [31:0]

位域	名称	描述
20	ENABLE	使能 trigger0 输出 0: disable 1: enable
19-0	LEAD_TIM	PWM 低电平中间的事件 trigger0 时刻控制，单位是工作时钟 cycle，有符号数 … 2: 2 cycle befor center of low level 1: 1 cycle before center of low level 0: center of low level -1: 1cycle after center of low level -2: 2cycle after center of low level

TRIG_OUT0_CFG 位域

40.6.12 TRIG_OUT1_CFG (0x4C)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD											ENABLE	LEAD_TIM																			
N/A											RW	RW																			
x	x	x	x	x	x	x	x	x	x	x	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	1	1

TRIG_OUT1_CFG [31:0]

位域	名称	描述
20	ENABLE	使能 trigger1 输出 0: disable 1: enable

位域	名称	描述
19-0	LEAD_TIM	PWM 高电平中间的事件 trigger1 时刻控制，单位是工作时钟 cycle，有符号数 … 2: 2 cycle befor center of hight level 1: 1 cycle before center of hight level 0: center of hight level -1: 1cycle after center of hight level -2: 2cycle after center of hight level

TRIG_OUT1_CFG 位域

40.6.13 PWM_DZ (0x50)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																DZ_N								DZ_P							
N/A																RW								RW							
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

PWM_DZ [31:0]

位域	名称	描述
15-8	DZ_N	Exc_n 死区控制，单位为工作时钟 cycle 0: no dead zone 1: 1 cycle dead zone 2: 2 cycle dead zone …
7-0	DZ_P	Exc_p 死区控制，单位为工作时钟 cycle 0: no dead zone 1: 1 cycle dead zone 2: 2 cycle dead zone …

PWM_DZ 位域

40.6.14 SYNC_OUT_CTRL (0x54)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PWM_OUT_DLY																RSVD										MIN2TRIG_EN	MAX2TRIG_EN	RSVD			SYNC_OUT_SEL

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RO																N/A										RW	RW	N/A		RW	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	x	x	x	x	x	x	x	x	x	x	0	0	x	x	0	0

SYNC_OUT_CTRL [31:0]

位域	名称	描述
31-16	PWM_OUT_DLY	从延时后的励磁信号触发信号，到第一个 pwm 信号输出的延时，单位是工作时钟 cycle 1: 1 cycle 2: 2 cycle ...
5	MIN2TRIG_EN	在励磁信号的最低点产生一次 trigger 输出 1: enable 0: disable
4	MAX2TRIG_EN	在励磁信号的最高点产生一次 trigger 输出 1: enable 0: disable
1-0	SYNC_OUT_SEL	输出同步信号相位选择 0: 0 phase of internal exciting signal 1: 90 phase of internal exciting signal 2: 180 phase of internal exciting signal 3: 270 phase of internal exciting signal

SYNC_OUT_CTRL 位域

40.6.15 EXC_SYNC_DLY (0x58)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD							DISABLE	DELAY																							
N/A							RW	RW																							
x	x	x	x	x	x	x	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

EXC_SYNC_DLY [31:0]

位域	名称	描述
24	DISABLE	关闭由硬件 trigger 输入触发励磁信号产生的功能 0: enable 1: disable

位域	名称	描述
23-0	DELAY	从 trigger 输入到触发励磁信号开始输出的延时值，单位为工作时钟 cycle 数 0: 1 cycle 1: 2 cycle … 0xffffff: 2^24 cycle

EXC_SYNC_DLY 位域

40.6.16 MAX_I (0x70)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MAX																								RSVD							VALID
RWC																								N/A							RWC
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	x	x	x	x	x	x	x	0

MAX_I [31:0]

位域	名称	描述
31-8	MAX	i_channel 输入最大值，写清
0	VALID	最大值有效指示，写清 0: max value is not valid 1: max value is valid

MAX_I 位域

40.6.17 MIN_I (0x74)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MIN																								RSVD							VALID
RWC																								N/A							RWC
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	x	x	x	x	x	x	x	0

MIN_I [31:0]

位域	名称	描述
31-8	MIN	i_channel 输入最小值，写清
0	VALID	最小值有效指示，写清 0: min value is not valid 1: min value is valid

MIN_I 位域

40.6.18 MAX_Q (0x78)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MAX																								RSVD							VALID
RWC																								N/A							RWC
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	x	x	x	x	x	x	x	0

MAX_Q [31:0]

位域	名称	描述
31-8	MAX	q_channel 输入最大值，写清
0	VALID	最大值有效指示，写清 0: max value is not valid 1: max value is valid

MAX_Q 位域

40.6.19 MIN_Q (0x7C)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MIN																								RSVD							VALID
RWC																								N/A							RWC
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	x	x	x	x	x	x	x	0

MIN_Q [31:0]

位域	名称	描述
31-8	MIN	q_channel 输入最小值，写清
0	VALID	最小值有效指示，写清 0: min value is not valid 1: min value is valid

MIN_Q 位域

40.6.20 THRS_I (0x80)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
THRS																								RSVD							
RW																								N/A							

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

THRS_I [31:0]

位域	名称	描述
31-8	THRS	i_channel 边沿检测的偏置设置，有符号数 … 2: the offset is 0x800000+2 1: the offset is 0x800000+1 0: the offset is 0x800000 -1: the offset is 0x800000-1 -2: the offset is 0x800000-2 …

THRS_I 位域

40.6.21 THRS_Q (0x84)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
THRS																								RSVD							
RW																								N/A							
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	x	x	x	x	x	x	x	x

THRS_Q [31:0]

位域	名称	描述
31-8	THRS	q_channel 沿检测的偏置配置，有符号数 … 2: the offset is 0x800000+2 1: the offset is 0x800000+1 0: the offset is 0x800000 -1: the offset is 0x800000-1 -2: the offset is 0x800000-2 …

THRS_Q 位域

40.6.22 EDG_DET_CTL (0x88)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																						HOLD						RSVD	FILTER		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
N/A																						RW						N/A	RW		
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	1	0	0	0	x	0	0	0

EDG_DET_CTL [31:0]

位域	名称	描述
9-4	HOLD	两个沿检测允许的最小间距 0:1 sample 1:2 sample 2:3 samples … 63:64 samples
2-0	FILTER	沿检测时需要连续为正数或负数的数量 0: 1 1: 2 … 7: 8

EDG_DET_CTL 位域

40.6.23 ACC_SCALING (0x8C)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																							TOXIC_LK	RSVD				ACC_SHIFT			
N/A																							RW	N/A				RW			
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	x	x	x	x	0	0	0	0

ACC_SCALING [31:0]

位域	名称	描述
8	TOXIC_LK	异常累加结果移除控制 1: enable 0: disable

位域	名称	描述
3-0	ACC_SHIFT	累加结果的移位控制，有符号数 0: {acc[39],acc[38:8]} 1: {acc[39],acc[37:7]} 2: {acc[39],acc[36:6]} … 7: {acc[39],acc[31:1]} 8: {acc[39],acc[30:0]} 9: acc/2^9 10: acc/2^10 … 15:acc/2^15

ACC_SCALING 位域

40.6.24 EXC_PERIOD (0x90)

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

EXC_PERIOD [31:0]

位域	名称	描述
31-0	EXC_PERIOD	励磁信号周期设置，单位为工作时钟 cycle 数 0: invalid value 1:1 cycle 2:2 cycles …

EXC_PERIOD 位域

40.6.25 SYNC_DELAY_I (0xA0)

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

SYNC_DELAY_I [31:0]

位域	名称	描述
31-0	DELAY	同步信号延时设置，单位为工作时钟 cycle 数，该值应小于 exc_period.exc_period 值的一半 0: invalid value 1: 1 cycles 2: 2 cycles ...

SYNC_DELAY_I 位域

40.6.26 RISE_DELAY_I (0xA8)

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

RISE_DELAY_I [31:0]

位域	名称	描述
31-0	RISE_DELAY	i_channel 输入上升沿的延时值，单位为工作时钟 cycle 0: 1 cycle 1: 2 cycles ...

RISE_DELAY_I 位域

40.6.27 FALL_DELAY_I (0xAC)

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

FALL_DELAY_I [31:0]

位域	名称	描述
31-0	FALL_DELAY	i_channel 输入数据下降沿的延时值，单位为工作时钟 cycle 0: 1 cycle 1: 2 cycles ...

FALL_DELAY_I 位域

40.6.28 SAMPLE_RISE_I (0xB0)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VALUE																								RSVD							
RO																								N/A							
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	x	x	x	x	x	x	x	x

SAMPLE_RISE_I [31:0]

位域	名称	描述
31-8	VALUE	延时后的整流信号上升沿采样到的输入值

SAMPLE_RISE_I 位域

40.6.29 SAMPLE_FALL_I (0xB4)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VALUE																								RSVD							
RO																								N/A							
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	x	x	x	x	x	x	x	x

SAMPLE_FALL_I [31:0]

位域	名称	描述
31-8	VALUE	延时后的整流信号下降沿采样的输入值

SAMPLE_FALL_I 位域

40.6.30 ACC_CNT_I (0xB8)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CNT_NEG																CNT_POS															
RO																RO															

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

ACC_CNT_I [31:0]

位域	名称	描述
31-16	CNT_NEG	整流信号低电平时对应的采样数量 1: 1 2: 2 …
15-0	CNT_POS	整流信号高电平时对应的采样个数 1: 1 2: 2 …

ACC_CNT_I 位域

40.6.31 SIGN_CNT_I (0xBC)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CNT_NEG																CNT_POS															
RO																RO															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

SIGN_CNT_I [31:0]

位域	名称	描述
31-16	CNT_NEG	整流信号为负数时，采样值为正数的个数统计
15-0	CNT_POS	整流信号为正数时，采样值为负数的个数统计

SIGN_CNT_I 位域

40.6.32 SYNC_DELAY_Q (0xC0)

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

SYNC_DELAY_Q [31:0]

位域	名称	描述
31-0	DELAY	同步信号延时设置，单位为工作时钟 cycle 数，该值应小于 exc_period.exc_period 值的一半 0: invalid value 1: 1 cycles 2: 2 cycles ...

SYNC_DELAY_Q 位域

40.6.33 RISE_DELAY_Q (0xC8)

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

RISE_DELAY_Q [31:0]

位域	名称	描述
31-0	RISE_DELAY	q_channel 输入上升沿的延时值，单位为工作时钟 cycle 0: 1 cycle 1: 2 cycles ...

RISE_DELAY_Q 位域

40.6.34 FALL_DELAY_Q (0xCC)

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

FALL_DELAY_Q [31:0]

位域	名称	描述
31-0	FALL_DELAY	q_channel 输入数据下降沿的延时值，单位为工作时钟 cycle 0: 1 cycle 1: 2 cycles ...

FALL_DELAY_Q 位域

40.6.35 SAMPLE_RISE_Q (0xD0)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VALUE																								RSVD							
RO																								N/A							
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	x	x	x	x	x	x	x	x

SAMPLE_RISE_Q [31:0]

位域	名称	描述
31-8	VALUE	延时后的整流信号上升沿采样到的输入值

SAMPLE_RISE_Q 位域

40.6.36 SAMPLE_FALL_Q (0xD4)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VALUE																								RSVD							
RO																								N/A							
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	x	x	x	x	x	x	x	x

SAMPLE_FALL_Q [31:0]

位域	名称	描述
31-8	VALUE	延时后的整流信号下降沿采样的输入值

SAMPLE_FALL_Q 位域

40.6.37 ACC_CNT_Q (0xD8)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CNT_NEG																CNT_POS															
RO																RO															

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

ACC_CNT_Q [31:0]

位域	名称	描述
31-16	CNT_NEG	整流信号低电平时对应的采样数量 1: 1 2: 2 …
15-0	CNT_POS	整流信号高电平时对应的采样个数 1: 1 2: 2 …

ACC_CNT_Q 位域

40.6.38 SIGN_CNT_Q (0xDC)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CNT_NEG																CNT_POS															
RO																RO															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

SIGN_CNT_Q [31:0]

位域	名称	描述
31-16	CNT_NEG	整流信号为负数时，采样值为正数的个数统计
15-0	CNT_POS	整流信号为正数时，采样值为负数的个数统计

SIGN_CNT_Q 位域

40.6.39 AMP_MAX (0xE0)

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

AMP_MAX [31:0]

位域	名称	描述
31-0	MAX	累加幅值的最大值

AMP_MAX 位域

40.6.40 AMP_MIN (0xE4)

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

AMP_MIN [31:0]

位域	名称	描述
31-0	MIN	累加幅值的最小值

AMP_MIN 位域

40.6.41 INT_EN (0xE8)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
INT_EN	RSVD															ACC_VLD_I_EN	ACC_VLD_Q_EN	RISING_DELAY_I_EN	FALLING_DELAY_I_EN	RISING_DELAY_Q_EN	FALLING_DELAY_Q_EN	SAMPLE_RISING_I_EN	SAMPLE_FALLING_I_EN	SAMPLE_RISING_Q_EN	SAMPLE_FALLING_Q_EN	ACC_VLD_I_OVH_EN	ACC_VLD_Q_OVH_EN	ACC_VLD_I_OVL_EN	ACC_VLD_Q_OVL_EN	ACC_AMP_OVH_EN	ACC_AMP_OVL_EN
RW	N/A															RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
0	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

INT_EN [31:0]

位域	名称	描述
31	INT_EN	使能中断输出
15	ACC_VLD_I_EN	i_channel 累加结果有效中断使能
14	ACC_VLD_Q_EN	q_channel 累加结果有效中断使能
13	RISING_DELAY_I_EN	i_channel 延时后整流信号上升沿中断使能
12	FALLING_DELAY_I_EN	i_channel 延时后整流信号下降沿中断使能
11	RISING_DELAY_Q_EN	q_channel 延时后整流信号上升沿中断使能

位域	名称	描述
10	FALLING_DELAY_Q_EN	q_channel 延时后整流信号下降沿中断使能
9	SAMPLE_RISING_I_EN	i_channel 输入上升沿中断使能
8	SAMPLE_FALLING_I_EN	i_channel 输入下降沿中断使能
7	SAMPLE_RISING_Q_EN	q_channel 输入上升沿中断使能
6	SAMPLE_FALLING_Q_EN	q_channel 输入下降沿中断使能
5	ACC_VLD_I_OVH_EN	i_channel 累加结果上溢出中断使能
4	ACC_VLD_Q_OVH_EN	q_channel 累加结果上溢出中断使能
3	ACC_VLD_I_OVL_EN	i_channel 累加结果下溢出中断使能
2	ACC_VLD_Q_OVL_EN	q_channel 累加结果下溢出中断使能
1	ACC_AMP_OVH_EN	累加结果幅值上溢出中断使能
0	ACC_AMP_OVL_EN	累加结果幅值下溢出中断使能

INT_EN 位域

40.6.42 ADC_INT_STATE (0xEC)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																ACC_VLD_I_STA	ACC_VLD_Q_STA	RISING_DELAY_I_STA	FALLING_DELAY_I_STA	RISING_DELAY_Q_STA	FALLING_DELAY_Q_STA	SAMPLE_RISING_I_STA	SAMPLE_FALLING_I_STA	SAMPLE_RISING_Q_STA	SAMPLE_FALLING_Q_STA	ACC_VLD_I_OVH_STA	ACC_VLD_Q_OVH_STA	ACC_VLD_I_OVL_STA	ACC_VLD_Q_OVL_STA	ACC_AMP_OVH_STA	ACC_AMP_OVL_STA
N/A																W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ADC_INT_STATE [31:0]

位域	名称	描述
15	ACC_VLD_I_STA	i_channel 累加结果有效中断状态
14	ACC_VLD_Q_STA	q_channel 累加结果有效中断状态

位域	名称	描述
13	RISING_DELAY_I_STA	i_channel 延时后整流信号上升沿中断状态
12	FALLING_DELAY_I_STA	i_channel 延时后整流信号下降沿中断状态
11	RISING_DELAY_Q_STA	q_channel 延时后整流信号上升沿中断状态
10	FALLING_DELAY_Q_STA	q_channel 延时后整流信号下降沿中断状态
9	SAMPLE_RISING_I_STA	i_channel 输入上升沿中断状态
8	SAMPLE_FALLING_I_STA	i_channel 输入下降沿中断状态
7	SAMPLE_RISING_Q_STA	q_channel 输入上升沿中断状态
6	SAMPLE_FALLING_Q_STA	q_channel 输入下降沿中断状态
5	ACC_VLD_I_OVH_STA	i_channel 累加结果上溢出中断状态
4	ACC_VLD_Q_OVH_STA	q_channel 累加结果上溢出中断状态
3	ACC_VLD_I_OVL_STA	i_channel 累加结果下溢出中断状态
2	ACC_VLD_Q_OVL_STA	q_channel 累加结果下溢出中断状态
1	ACC_AMP_OVH_STA	累加结果幅值上溢出中断状态
0	ACC_AMP_OVL_STA	累加结果幅值下溢出中断状态

ADC_INT_STATE 位域

40.7 RDC 配置使用说明

40.7.1 励磁生成配置说明

励磁信号的输出支持 DAC 输出和 PWM 输出，其步骤如下：

1. 配置励磁信号的周期，配置寄存器 exc_timming.smp_rate 和 exc_timming.smp_num 为需要的值；
2. 如果励磁信号是 DAC 的输出方式，配置寄存器 exc_scaling.amp_man 和寄存器 exc_scaling.ampl_exp 和 exc_offset.ampl_offset 为合适的值；
3. 如果励磁信号是 PWM 的输出方式，配置寄存器 pwm_scaling.ampl_man 和 pwm_scaling.ampl_exp 和 pwm_offset.ampl_offset 为合适的值；如果需要将上一 PWM 周期的占空比小数位累加到下一 PWM 周期，则还需要配置寄存器 pwm_scaling.dither 的值为 1；

4. 配置寄存器 rdc_ctl.exc_en 为 1，允许励磁信号的产生；
5. 如果是通过软件方式触发励磁信号，则写寄存器 rdc_ctl.exc_start 为 1，即可触发励磁信号的输出；
6. 如果是通过硬件事件触发励磁信号，则需要配置寄存器 exc_sync_dly.dly 为合适的值，控制从硬件事件到励磁信号产生的延时，再配置 exc_sync_dly.disable 为 0，当硬件事件触发时，励磁信号就可以产生了。

40.7.2 延时测量配置说明

延时测量有多种方法，下面描述一下如何采用 RDC 模块辅助的方法进行从励磁信号产生到感应信号接收的相位延时的测量步骤：

1. 配置 rdc_ctl.exc_en 和 rdc_ctl.acc_en 为 1，配置 rdc_ctl.rectify_sel 为 0，选择励磁信号生成器作为同步源；
2. 配置 in_ctl 为合适的值，选择合适的 ADC 输入和通道号；
3. 读取 max_i/q 的值，获得 ADC 输入的最大值和最小值；经多次计算中间值，并求平均，将结果配置到 thrs_i/q 中；
4. 如果需要启动滤抖功能，配置 edg_det_ctl.filter 为合适的值；
5. 读取 rise_delay_i/q 的值，并多次求平均，则这个值就是相位延时值，当这个值超过励磁信号周期的一半时，需要将寄存器 rdc_ctl.rectify_sel 配置为 2，并重复以上步骤；
6. 将相位延时值配置到寄存器 sync_delay_i/q 中，同时将励磁信号的周期配置到寄存器 exc_period 中。

‘

40.7.3 载波累加配置说明

载波累加的配置方法为：

1. 配置 rdc_ctl.acc_en 为 1，配置 RDC_ctl.acc_len 为需要的值；
2. 配置 in_ctl 为合适的值，选择争取的 ADC 输入源；
3. 等待中断产生，读取 acc_i/q 的值，获取累加结果；

41 串行编码器接口 SEI

本章节介绍 SEI 的主要功能和特性。

41.1 特性总结

本章节介绍 SEI 的主要特性：

- 2 个 SEI 控制器
- 9 个数据寄存器组
- 最高支持 64 条指令
- 支持同步通信和异步通信
- 支持主机模式和从机模式（作为编码器）
- 支持 RS-485 及 RS-422 接口
- 每个 SEI 控制器支持共 3 种触发方式
  - 外部触发，包括触发输入及触发输出各 7 路
  - 周期性触发
  - 软件触发
- 支持精确控制 SAMPLE 或 UPDATE 位置信息与时间戳的时机
- 支持命令匹配及指令跳转
- 支持自动 CRC 校验
- 支持自动奇偶校验
- 支持 WatchDog
- 支持超时 TIMEOUT 及收发 CDM/CDS

SEI 的框图如图 52。

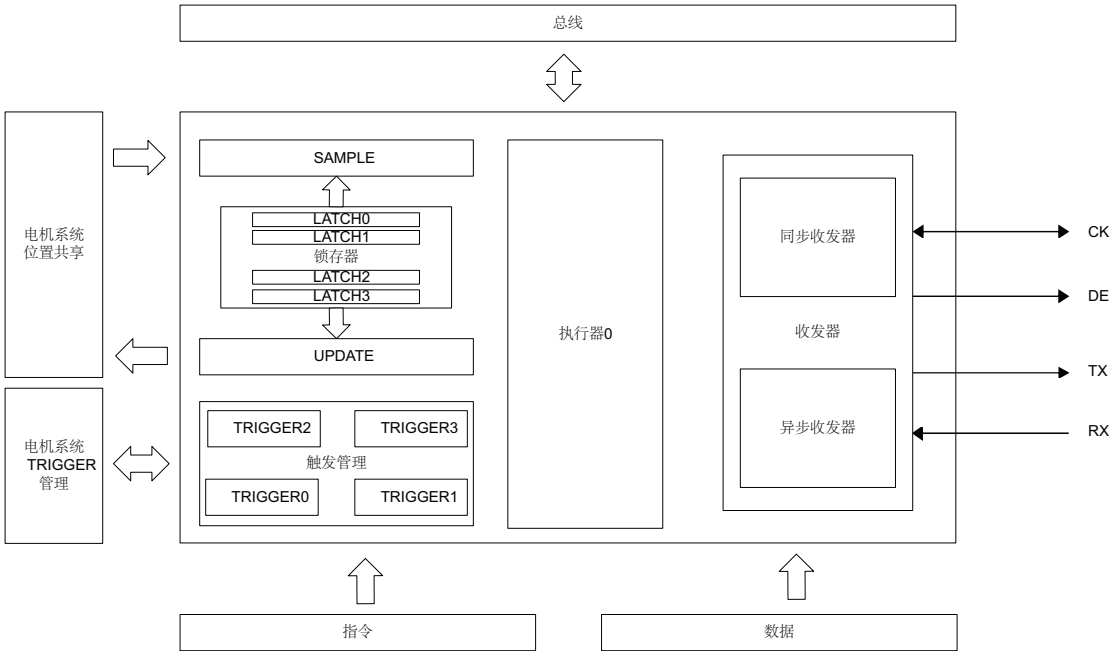


图 52: SEI 结构框图

41.2 功能描述

41.2.1 数据发送与接收

每组 SEI 有 DE、CK、TX 和 RX 一共 4 个引脚，支持 RS-485 和 RS-422 接口。

在作为主机时，SEI 的主要用途是从外部获取位置信息（如转子的位置、圈数、转速及电机工作状态等），而在作为从机时，SEI 的主要用途是将电机系统内获取的位置信息发送至外部，此时 SEI 充当编码器的作用。

41.2.1.1 异步模式

当收发器工作在异步模式下时，配置 CTRL[n][XCVR][BAUD_CFG] 寄存器中的 BAUD_DIV 位域可以设置收发数据的波特率，并通过设置 CTRL[n][XCVR][DATA_CFG] 寄存器中的 RXD_POINT 和 TXD_POINT 位域调整数据收发的时间点。

此外，还需要根据具体的协议要求，配置 CTRL[n][XCVR][PIN] 寄存器设置 DE、TX 和 RX 引脚的状态，控制空闲时数据线和时钟线的线上状态。收发时的数据长度通过 CTRL[n][XCVR][TYPE_CFG] 寄存器中的 DATA_LEN 位域设置，最高支持 32 位，收发器会根据此处的设置自动在通信过程中插入起始位和停止位、奇偶校验位。通过 CTRL[n][XCVR][TYPE_CFG] 寄存器中的 PAR_EN 和 PAR_POL 选择是否允许奇偶校验并选择奇校验还是偶校验。

41.2.1.2 同步模式

在同步模式下，根据需要选择收发器工作在主机模式还是从机模式。当需要从外部的编码器中获取位置信息时，将 SEI 配置为主机模式，即可将 SEI 作为上位机；当需要将电机系统内部的位置信息发送出去时，将 SEI 配置为从机模式，此时 SEI 作为编码器。

通过配置 CTRL[n][XCVR][BAUD_CFG] 寄存器中的 BAUD_DIV 位域，可以设置收发数据的时钟频率，并通过配置 CTRL[n][XCVR][CLK_CFG] 寄存器设置时钟跳变点 CK0_POINT 和 CK1_POINT。配置 CTRL[n][XCVR][DATA_CFG] 寄存器中的 RXD_POINT 和 TXD_POINT 位域，可以调整数据收发的时间点。此外，还需要根据具体的协议要求，配置 CTRL[n][XCVR][PIN] 寄存器设置 CK、DE、TX 和 RX 引脚的状态，控制空闲时数据线和时钟线的线上状态。

41.2.2 指令

SEI 一共有 8 种不同的操作，根据指令寄存器的配置执行数据收发的相关操作，最高支持 64 条指令。每条指令由操作、时钟、CRC 寄存器、DATA 寄存器、长度字段组成，用户根据应用场景，按照时序先后顺序，依次配置各条指令，通过多条指令组成满足目标通信协议的时序。各个字段的说明如下：

- 操作字段：表明此条指令所执行的操作。操作字段的各种操作类型说明如下：

操作码	操作	功能描述
000	停止/等待	收发器停止一段时间或结束运行，用于产生掩饰或等待时间
001	跳转	用于指令跳转： 可根据命令匹配的结果，跳转至对应的指令开始执行； 可直接跳转至 POINTER_WDOG 中的指令开始执行； 可直接跳转至 POINTER_INIT 中的指令开始执行；
010	时限内发送	发送一定数量的数据位，无时间限制
011	无时限发送	发送一定数量的数据位，有时间限制

操作码	操作	功能描述
100	时限内等待	等待数据线上出现期望的电平，用于主从机之间同步，有时间限制
101	无时限等待	等待数据线上出现期望的电平，用于主从机之间同步，无时间限制
110	时限内接收	接收一定数量的数据位，有时间限制
111	无时限接收	接收一定数量的数据位，无时间限制

表 187: 数据寄存器组的用途及描述

- 时钟字段：配置此条指令执行时的时钟线状态。
- CRC 寄存器字段：用于指定用于计算 CRC 的数据寄存器。
- DATA 寄存器字段：用于此条指令收发数据时所需要使用的数据寄存器。
- 操作数字段：此条指令收发数据的长度，或指令执行占用的时长。

41.2.3 数据

SEI 中共有 9 组数据寄存器，其中忽略数据寄存器有 1 组，通用数据寄存器有 8 组，各组数据寄存器之间相互独立。此外，SEI0 与 SEI1 各有一组命令寄存器。

序号	用途	功能描述
DATA[0]	忽略数据寄存器	用于接收通信过程中无意义的数据或用户不关心的数据，使用该寄存器接收到的数据将会被舍弃，用户无法再对数据进行其他处理。
CMD	命令寄存器	用于接收或发送通信中的命令。
DATA[9:2]	通用数据寄存器	工作在数据模式时，用于接收或发送数据； 工作在检查模式时，用于检查比对接收到的数据与期望数据； 工作在 CRC 模式时，用于计算 CRC 或 CHECKSUM；

表 188: 数据寄存器组的用途及描述

除此之外，SEI 指令中的 DAT 字段还有常数 0 和常数 1 可供作为接收或发送的数据内容。使用时可以根据需求，灵活的配置发送一定长度的 0 或 1 到数据线上，或等待直到接收到数据线上一定长度的 0 或 1。

需要注意的是，当同一条使用了命令寄存器的指令分别由 SEI0 和 SEI1 执行时，会自动使用 SEI0 和 SEI1 各自内部的命令寄存器，无需再进行其他配置。

数据寄存器工作模式

- 数据模式
  工作在数据模式时，可通过 DATA[n][MODE] 寄存器配置该数据寄存器用于收发数据时的数据格式，包括数据长度、字序、位序及是否为有符号数等。同时，还需要在 DATA[n][IDX] 寄存器中配置数据的填充方式及填充顺序。配置完成后，需要通过将该数据寄存器的复位指针进行置位，以更新该数据寄存器中的初始指针。
- 检查模式
  工作在检查模式时，软件提前将期望值写入 DATA[n][GOLD] 寄存器中，其他配置与工作在数据模式时类似。当使用该数据寄存器接收数据时，会将收到的数据与期望数据进行逐位检查比对，若不一致则会产生传输错误。

- CRC 模式
  工作在 CRC 模式时，将 CRC 多项式与 CRC 初始值分别填入 DATA[n][CRCPOLY] 寄存器与 DATA[n][CRCINIT] 寄存器中，并根据需要在 DATA[n][MODE] 寄存器中配置 CRC 长度、移位模式及是否启用 CRC 反相。通过配置操作指令，可以将 CRC 的计算结果作为数据发送，也可以将 CRC 的计算结果作为期望值，与接收到的 CRC 进行检查比对。
  需要注意的是，在 SAMPLE 或 UPDATE 发生时，数据寄存器中 CRC 的计算结果会被清除，自动重置为初始值。

命令寄存器与命令匹配

命令寄存器的基本功能与配置方法和通用数据寄存器一致，但命令寄存器可以将接收到的命令与命令表中预填的命令进行匹配，并根据匹配结果跳转至对应的指令执行后续的操作。

接收到 CMD 后，使用跳转命令，开始命令匹配，并跳转至匹配结果。SEI 会根据匹配条件，自动从 CMD_TABLE[0] 到 CMD_TABLE[6] 逐个比较，若均未匹配到，则会无条件跳转至最后一个，即 CMD_TABLE[7] 的 PTR[n] 指令。其中匹配条件为：

- CMD_TABLE[n][MSK]: 选择需要进行比较的 Bit
- CMD_TABLE[n][MIN]: 接收到的 CMD 是否小于此寄存器的值
- CMD_TABLE[n][MAX]：接收到的 CMD 是否大于此寄存器的值

若满足匹配条件，则跳转至该 CMD_TABLE 的 PTR[n] 指令开始执行，此处的 n 由该条跳转指令的 OPR[3:0] 字段选择。

41.2.4 触发管理

SEI 支持程序指令直接执行，也支持通过触发信号来控制程序指令的执行。SEI 支持外部的触发、周期性触发和软件触发共 3 种触发方式，其中外部的触发为 TRIGGER[0] 和 TRIGGER[1] ，周期性触发为 TRIGGER[2]，软件触发为 TRIGGER[3] 。

SEI 内部有 4 个 TRIGGER 命令寄存器，用户可以预先将命令写入触发命令寄存器，当触发产生时，硬件会自动将该触发方式所对应的触发命令寄存器中的命令填入命令寄存器，并在执行发送命令寄存器的指令时将该命令发出。此外，当触发产生时，触发产生的时间会被记录，用户可以通过读取对应的寄存器来获取各种触发生成时刻的时间戳，从而估算传输用时。

41.2.4.1 外部触发

- 触发输入
  每个 SEI 支持 8 个通道的外部触发信号输入，用户可以从 8 个通道中选择任意输入信号作为 TRIGGER[0] 和 TRIGGER[1] 的触发信号。外部触发信号可以通过配置 TRIGM 选择其来源，通常来自电机系统中的其他模块，也可以通过 TRIGM 的 GPIO 来自芯片外部，从而触发 SEI 开始执行指令。
- 触发输出
  每个 SEI 支持 8 个通道的触发信号输出，8 个通道的触发信号来自 SEI 内部的 4 个锁存器 LATCH[3:0]，用户可以通过 CTRL[n][TRG][OUT_CFG] 寄存器将对应的锁存器的输出信号输出到 TRGM，从而输出到电机系统中的其他模块或芯片外部。

41.2.4.2 周期性触发

SEI 内部有一个 32 位的计数器用于实现程序指令的周期性触发，通过配置触发周期，可以实现周期性的执行程序指令。用户可以从相关寄存器中查看周期性触发的状态及触发次数。周期性触发可以选择直接触发或等待触发源后再开始触发：

- 若配置为直接触发，则使能周期性触发后会立即触发第一次执行，并重新开始计时，在到达触发周期后生成新的触发信号，并开始第二次执行，如此往复。
- 若配置为等待触发源再生成触发，则使能周期性触发后，不会立即开始第一次执行与计时，需要等待外部触发源到来 SEI 才会开始周期性触发。根据需要，还可以选择是否将外部触发源与周期性触发信号进行同步。若同步，则每次外部触发源到来时都会产生触发，随后周期性触发。

需要注意的是，在其他所有配置完成以后才可以进行使能周期性触发这一操作，避免 SEI 未配置完成便开始工作。且配置的触发周期需要大于程序指令执行单次的用时，否则新的触发信号会中断当前正在进行的通信。

41.2.4.3 软件触发

软件可通过对 CTRL[n][TRG][SW] 寄存器进行置位来实现触发 SEI 直接开始执行指令。软件触发的指令仅执行一次，若需要执行多次或重复执行，可以在指令中添加跳转指令，在执行到该跳转指令后程序可跳转至特定的指令开始执行。

41.2.5 位置信息的 SAMPLE 及 UPDATE

通过 SAMPLE 及 UPDATE 功能，SEI 既可以实现从外部带通信协议的编码器中获取位置信息，也可以将电机系统内部其他模块产生的位置信息通过通信协议向外部发送。

41.2.5.1 LATCH 状态机

SEI 的 SAMPLE 及 UPDATE 的实现主要基于状态机的跳转。SEI 内部共有 4 个 LATCH 状态机可供选取用于控制 SAMPLE 或 UPDATE，其中每个 LATCH 状态机均有 4 个状态，这 4 个状态只能单向流转，即状态 0 到状态 1，状态 1 到状态 2，状态 2 到状态 3，状态 3 到状态 0。通过配置寄存器，可以将通信过程中的特定时间点作为状态跳转的条件。可选取作为状态机跳转的条件如下：

- 收发超时，如状态为 1、状态为 0、上升时
- 发送数据线 TX 的状态，如状态为 0、状态为 1、上升、下降时
- 主机模式下，时钟线状态，如状态为 0、状态为 1、上升、下降时
- 指针，如指针匹配、指针不匹配、指针开始、指针结束时

实际配置时，并非状态机的每一个状态都需要设置跳转条件，当不需要 4 个状态时，中间状态可以配置为无条件跳转。通过配置 LATCH 的延时，可以延迟 SAMPLE 或 UPDATE 的发生时机。

41.2.5.2 SAMPLE 功能

在作为主机时，SAMPLE 功能主要用于对时间戳的采样，为每一次通信获取的位置信息提供时间戳，以便电机系统内部的其他模块对其进行后续分析处理，如 MMC 的预测与跟随。而在作为从机时，SAMPLE 功能则是将电机系统其他模块产生的位置信息及时间戳采样至指定的数据寄存器，用于向外发送。SAMPLE 时间点的选取，可参考相关协议。

根据应用场景的需求，可以位置、圈数、速度及加速度中的任意一种或多种数据作为 SAMPLE 发生时采样的数据，只需指定用于收发该数据的数据寄存器并使能对应的控制位即可。此外，还可以配置 SAMPLE 发生时的采

样窗口及采样方式。

41.2.5.3 UPDATE 功能

在作为主机时，UPDATE 功能主要用于传输完成后，将获取到的位置信息与时间戳更新至电机系统内部，供其他模块使用。

根据应用场景的需求，可以位置、圈数、速度及加速度中的任意一种或多种数据作为 UPDATE 发生时更新的数据，只需指定用于收发该数据的数据寄存器并使能对应的控制位即可。并可配置 UPDATE 发生时是采用接收到的数值还是内部软件填写的覆盖值。此外，当传输出错时，还可以选择是否仍将数据通过 UPDATE 传递给下一级。

41.2.6 中断和 DMA

SEI 支持生成以下中断：

- TRG_ERR[3:0]：当 TRIGGER[0]~TRIGGER[3] 发生错误时；
- TRIGER[3:0]：当 TRIGGER[0]~TRIGGER[3] 生成触发信号时；
- LATCH[3:0]：当 LATCH[0]~LATCH[3] 状态跳转时；
- SMP_ERR：当采样位置信息发生错误时；
- TIMEOUT：当通信过程中的超时发生时；
- TRX_ERR：当传输过程中发生传输错误时，包括 CRC 校验结果错误、数据检查比对错误等；
- INSTR0/1：当指令 0 或指令 1 的开始或结束时，需通过配置 CTRL[n][IRQ][INSTR0] 或 CTRL[n][IRQ][INSTR1] 寄存器指定任意一条指令为指令 0 或指令 1，故可在执行到任意一条特定的指令时触发中断；
- PTR0/1：当指针 0 或指令 1 的开始或结束时，需通过配置 CTRL[n][IRQ][POINTER0] 或 CTRL[n][IRQ][POINTER1] 指定任意一条指针为指针 0 或指针 1，故可在执行到任意一条特定的指针时触发中断；
- WDOG：当设置的看门狗触发时；
- EXECPT：当异常事件发生时；
- STALL：当收发器的通信停止时；

SEI 支持生成以下 DMA 请求：

- TRG_ERR[3:0]：当 TRIGGER[0]~TRIGGER[3] 发生错误时；
- TRIGER[3:0]：当 TRIGGER[0]~TRIGGER[3] 生成触发信号时；
- LATCH[3:0]：当 LATCH[0]~LATCH[3] 状态跳转时；
- SMP_ERR：当采样位置信息发生错误时；
- TIMEOUT：当通信过程中的超时发生时；
- TRX_ERR：当传输过程中发生传输错误时，包括 CRC 校验结果错误、数据检查比对错误等；
- INSTR0/1：当指令 0 或指令 1 的开始或结束时，需通过配置 CTRL[n][IRQ][INSTR0] 或 CTRL[n][IRQ][INSTR1] 寄存器指定任意一条指令为指令 0/1，故可在执行到任意一条特定的指令时生成 DMA 请求；
- PTR0/1：当指针 0 或指令 1 的开始或结束时，需通过配置 CTRL[n][IRQ][POINTER0] 或 CTRL[n][IRQ][POINTER1] 指定任意一条指针为指针 0/1，故可在执行到任意一条特定的指针时生成 DMA 请求；
- WDOG：当设置的看门狗触发时；
- EXECPT：当异常事件发生时；
- STALL：当收发器的通信停止时；

41.2.7 协议支持

SEI 可以通过编程实现多种通信协议，如 Tamagawa 编码器协议、HIPERFACE® 、Nikon 编码器协议、SSI、BiSS-C 及 EnDat2.1/2.2 协议等，用户还可以通过自定义编程支持更多协议。尽管可支持的协议非常丰富，但 SEI 整体的配置流程与思路基本一致，以 Tamagawa 编码器的通信协议为例，当 SEI 作为上位机通过 DataID 3 命令周期性地去读取编码器的数据时，配置流程如下：

- 配置控制寄存器
  - 配置初始程序指针位置及数据寄存器地址偏移；
  - 配置 SEI 工作模式及异步收发器的波特率及空闲时的线上状态；
  - 配置周期性触发；
  - 将 Data ID 3 的命令填入周期性触发命令寄存器 CTRL[0][TRG_TABLE][CMD][2]；
- 配置操作指令
  - 使用命令寄存器发送 8 bits 的读位置命令，即发送控制域（CF, Control Filed）；
  - 使用数据寄存器 2 接收 8 bits 的读位置命令 CF 数据，同时将接收到的数据传给数据寄存器 7 计算 CRC；
  - 使用数据寄存器 3 接收 8 bits 的状态标志域（SF, Status Filed）数据，使用数据寄存器 7 计算 CRC；
  - 使用数据寄存器 4 接收数据域（DF, Data Filed）中 24 bits 的位置信息（ABS0~ABS2, Absolute data in one revolution），使用数据寄存器 7 计算 CRC；
  - 使用数据寄存器 0 接收数据域中 8 bits 的编码器 ID（ENID, Encoder ID），使用数据寄存器 7 计算 CRC；
  - 使用数据寄存器 5 接收数据域中 24 bits 的圈数信息（ABM0~ABM2, Multi-turn data），使用数据寄存器 7 计算 CRC；
  - 使用数据寄存器 6 接收数据域中 8 bits 的报错信息（ALMC, Encoder error），使用数据寄存器 7 计算 CRC；
  - 使用数据寄存器 7 接收 CRC 校验位（CRC, CRC Field）数据，并进行 CRC 检查；
- 配置需要使用的数据寄存器
  - 配置命令寄存器对应的数据模式寄存器（CTRL[0][CMD][MODE]）及数据索引寄存器（CTRL[0][CMD][IDX]）。其中 MODE 配置为数据模式。
  - 配置数据寄存器 2~6 对应的数据模式寄存器（DATA[n][MODE]）及数据索引寄存器（DATA[n][IDX]），其中 MODE 配置为数据模式；
  - 配置数据寄存器 7 为 CRC 模式，配置 CRC 多项式、初始值等；
- 使能周期性触发
  - 使能 ENGINE 及周期性触发，指令开始执行。

41.2.7.1 异步通信协议

Tamagawa 指令配置参考：

INSTR	OP 字段	CK 字段	CRC 字段	DAT 字段	LEN 字段	寄存器配置	配置说明
0	无时限发送	任意	不计算 CRC	命令寄存器	8 bits	0x0C000107	使用命令寄存器发送 CF 命令

INSTR	OP 字段	CK 字段	CRC 字段	DAT 字段	LEN 字段	寄存器配置	配置说明
1	无时限接收	任意	数据寄存器 7	数据寄存器 2	8 bits	0x1C070207	使用数据寄存器 2 接收 CF 数据，并用数据寄存器 7 计算 CRC
2	无时限接收	任意	数据寄存器 7	数据寄存器 3	8 bits	0x1C070307	使用数据寄存器 3 接收 SF，并计算 CRC
3	无时限接收	任意	数据寄存器 7	数据寄存器 4	24 bits	0x1C070417	使用数据寄存器 4 接收 ABS0~ABS2，并计算 CRC
4	无时限接收	任意	数据寄存器 7	数据寄存器 0	8 bits	0x1C070007	使用数据寄存器 0 接收 ENID，并计算 CRC
5	无时限接收	任意	数据寄存器 7	数据寄存器 5	24 bits	0x1C070517	使用数据寄存器 5 接收 ABM0~ABM2，并计算 CRC
6	无时限接收	任意	数据寄存器 7	数据寄存器 6	8 bits	0x1C070607	使用数据寄存器 6 接收 ALMC，并计算 CRC
7	无时限接收	任意	不计算 CRC	数据寄存器 7	8 bits	0x1C000707	使用数据寄存器 7 接收比对 CRC

表 189: 作为 Tamagawa 上位机时 SEI 指令参考

HIPERFACE® 指令配置参考（以 42H 命令为例）：

INSTR	OP 字段	CK 字段	CRC 字段	DAT 字段	LEN 字段	寄存器配置	配置说明
0	无时限发送	任意	数据寄存器 7	数据寄存器 2	8 bits	0x0C070207	使用数据寄存器 2 发送 ADDRESS 数据，并用数据寄存器 7 计算校验码
1	无时限发送	任意	数据寄存器 7	命令寄存器	8 bits	0x0C070107	使用命令寄存器发送 COMMAND 命令，并计算校验码
2	无时限发送	任意	不计算 CRC	数据寄存器 7	8 bits	0x0C000707	发送数据寄存器 7 中的 CHECKSUM 校验码
3	无时限接收	任意	数据寄存器 7	数据寄存器 3	8 bits	0x1C070307	使用数据寄存器 3 接收编码器发回的 ADDRESS 数据，并用数据寄存器 7 计算校验码

INSTR	OP 字段	CK 字段	CRC 字段	DAT 字段	LEN 字段	寄存器配置	配置说明
4	无时限接收	任意	数据寄存器 7	数据寄存器 4	8 bits	0x1C070407	使用数据寄存器 4 接收编码器发回的 COMMAND 数据，并计算校验码
5	无时限接收	任意	数据寄存器 7	数据寄存器 5	8 bits	0x1C070507	使用数据寄存器 5 接收位置数据中的 Position[31:24]，并计算校验码
6	无时限接收	任意	数据寄存器 7	数据寄存器 5	8 bits	0x1C070507	使用数据寄存器 5 接收位置数据中的 Position[23:16]，并计算校验码
7	无时限接收	任意	数据寄存器 7	数据寄存器 5	8 bits	0x1C070507	使用数据寄存器 5 接收位置数据中的 Position[15:8]，并计算校验码
8	无时限接收	任意	数据寄存器 7	数据寄存器 5	8 bits	0x1C070507	使用数据寄存器 5 接收位置数据中的 Position[7:0]，并计算校验码
9	无时限接收	任意	不计算 CRC	数据寄存器 7	8 bits	0x1C000707	使用数据寄存器 7 接收比对 CHECKSUM

表 190: 作为 HIPERFACE® 上位机时 SEI 指令参考

Nikon 指令配置参考（以 CDF0 命令为例）：

INSTR	OP 字段	CK 字段	CRC 字段	DAT 字段	LEN 字段	寄存器配置	配置说明
0	无时限发送	任意	数据寄存器 5	命令寄存器	13 bits	0x0C05010C	发送命令寄存器中 CDF0 的 Sink Code, Frame Code, EA 及 CC 部分的数据，并使用数据寄存器 5 计算 CRC
1	无时限发送	任意	不计算 CRC	数据寄存器 5	3 bits	0x0C000602	发送命令寄存器中的 CRC 计算结果

INSTR	OP 字段	CK 字段	CRC 字段	DAT 字段	LEN 字段	寄存器配置	配置说明
2	无时限接收	任意	数据寄存器 5	数据寄存器 2	16bits	0x1C05020F	使用数据寄存器 2 接收 IF（Information Field），并使用数据寄存器 5 计算 CRC
3	无时限接收	任意	数据寄存器 5	数据寄存器 3	16bits	0x1C05030F	使用数据寄存器 3 接收 DF0（IData Field），并计算 CRC
4	无时限接收	任意	数据寄存器 5	数据寄存器 3	16bits	0x1C05030F	使用数据寄存器 3 接收 DF1，并计算 CRC
5	无时限接收	任意	数据寄存器 5	数据寄存器 4	8bits	0x1C050407	使用数据寄存器 2 接收 DF2 中非 CRC 部分的数据，并计算 CRC
6	无时限接收	任意	不计算 CRC	数据寄存器 5	8bits	0x1C000507	使用数据寄存器 5 接收 DF2 中 CRC 部分的数据，并比对 CRC

表 191: 作为 Nikon 上位机时 SEI 指令参考

41.2.7.2 同步通信协议

SSI 指令配置参考 (以 POSITAL 的 SSI 协议描述为例)：

INSTR	OP 字段	CK 字段	CRC 字段	DAT 字段	LEN 字段	寄存器配置	配置说明
0	停顿	时钟线为高	不计算 CRC	忽略数据	2 bits	0x03000001	拉高时钟线等待两个时钟周期
1	无时限接收	时钟线上升-下降	不计算 CRC	常数 1	1 bits	0x1D001F00	发出时钟并在第一个下降沿接收数据线上空闲状态时 1bit 的 1
2	无时限接收	时钟线上升-下降	不计算 CRC	数据寄存器 3	1 bit	0x1D000300	使用数据寄存器 3 接收 1bit 的 1
3	无时限接收	时钟线上升-下降	不计算 CRC	数据寄存器 8	12 bits	0x1D00080B	使用数据寄存器 8 接收 12 bits 的圈数信息（MT, Multi TurnValue）
4	无时限接收	时钟线上升-下降	不计算 CRC	数据寄存器 9	12 bits	0x1D00090B	使用数据寄存器 9 接收 12 bits 的位置信息（ST, Single Turn-Value）

INSTR	OP 字段	CK 字段	CRC 字段	DAT 字段	LEN 字段	寄存器配置	配置说明
5	无时限等待	时钟线为高	不计算 CRC	常数 1	1 bit	0x17001F00	拉高时钟线，等待直到收到 1bit 的 1，等待 WatchDog 产生 Time-out

表 192: 主机模式

INSTR	OP 字段	CK 字段	CRC 字段	DAT 字段	LEN 字段	寄存器配置	配置说明
0	无时限发送	时钟线下降-上升	不计算 CRC	数据寄存器 6	1 bit	0x0E000600	发送 1bit 的 1
1	无时限发送	时钟线下降-上升	不计算 CRC	数据寄存器 6	12 bits	0x0E00040B	使用数据寄存器 6 发送 12 bits 的圈数信息（MT, Multi TurnValue）
2	无时限发送	时钟线下降-上升	不计算 CRC	数据寄存器 5	12 bits	0x0E00050B	使用数据寄存器 5 发送 12 bits 的位置信息（ST, Single Turn-Value）
3	无时限发送	时钟线下降-上升	不计算 CRC	常数 0	2 bits	0x0E001E00	发送 2bits 的 0 作为 Timeout 信号拉低数据线直到 WatchDog 产生 Timeout

表 193: 从机模式

BiSS-C 指令配置参考：

INSTR	OP 字段	CK 字段	CRC 字段	DAT 字段	LEN 字段	寄存器配置	配置说明
0	停顿	时钟线为高	不计算 CRC	忽略数据	2 bits	0x03000001	拉高时钟线等待两个时钟周期
1	无时限等待	时钟线上升-下降	不计算 CRC	常数 1	1 bit	0x15001F00	等待直至收到 1bit 的 1
2	无时限等待	时钟线上升-下降	不计算 CRC	常数 1	1 bit	0x15001F00	等待直至收到 1 bit 的 1 作为 START 位
3	无时限接收	时钟线上升-下降	不计算 CRC	数据寄存器 4	1 bit	0x1D000400	使用数据寄存器 4 接收 1 bit 的 CDS

INSTR	OP 字段	CK 字段	CRC 字段	DAT 字段	LEN 字段	寄存器配置	配置说明
4	无时限接收	时钟线上升-下降	数据寄存器 7	数据寄存器 8	12 bits	0x1D07080B	使用数据寄存器 8 接收 12 bits 的位置信息（MT, Multiturnposition），并使用数据寄存器 7 计算 CRC
5	无时限接收	时钟线上升-下降	数据寄存器 7	数据寄存器 9	12 bits	0x1D07090B	使用数据寄存器 5 接收 12 bits 的位置信息（ST, Singleturnposition），并计算 CRC
6	无时限接收	时钟线上升-下降	数据寄存器 7	常数 0	1 bit	0x1D071E00	接收 Error status bit，并计算 CRC
7	无时限接收	时钟线上升-下降	数据寄存器 7	常数 1	1 bit	0x1D071F00	接收 Warning status bit，并计算 CRC
8	无时限接收	时钟线上升-下降	不计算 CRC	数据寄存器 7	6 bits	0x1D000705	使用数据寄存器 7 接收比对 CRC

表 194: 主机模式

INSTR	OP 字段	CK 字段	CRC 字段	DAT 字段	LEN 字段	寄存器配置	配置说明
0	无时限发送	时钟线下降-上升	不计算 CRC	常数 1	1 bit	0x0E001F00	发送 1bit 的 1
1	无时限发送	时钟线下降-上升	不计算 CRC	数据寄存器 3	5 bits	0x0E000304	使用数据寄存器 3 发送 Header
2	无时限发送	时钟线下降-上升	数据寄存器 6	数据寄存器 4	12 bits	0x0E06040B	使用数据寄存器 4 发送 12 bits 的位置信息（MT, Multiturnposition），并使用数据寄存器 6 计算 CRC
3	无时限发送	时钟线下降-上升	数据寄存器 6	数据寄存器 5	12 bits	0x0E06050B	使用数据寄存器 5 发送 12 bits 的位置信息（ST, Singleturnposition），并计算 CRC
4	无时限发送	时钟线下降-上升	数据寄存器 6	数据寄存器 3	1 bit	0x0E060300	使用数据寄存器 3 发送 Error status bit，并计算 CRC
5	无时限发送	时钟线下降-上升	数据寄存器 6	数据寄存器 3	1 bit	0x0E060300	使用数据寄存器 3 发送 Warning status bit，并计算 CRC
6	无时限发送	时钟线下降-上升	不计算 CRC	数据寄存器 6	6 bits	0x0E000605	发送数据寄存器 6 中 CRC 的计算结果

INSTR	OP 字段	CK 字段	CRC 字段	DAT 字段	LEN 字段	寄存器配置	配置说明
7	无时限发送	时钟线为高	不计算 CRC	常数 0	2 bits	0x0F001E01	发送 2bits 的 0 作为 Timeout 信号拉低数据线直到 WatchDog 产生 Timeout

表 195: 从机模式

EnDat 指令配置参考 (以 110001 命令为例)：

INSTR	OP 字段	CK 字段	CRC 字段	DAT 字段	LEN 字段	寄存器配置	配置说明
0	无时限等待	时钟线为低	不计算 CRC	常数 1	1bit	0x14001F00	拉低时钟线，直到收到 1bit 1
1	无时限发送	时钟线下降-上升	不计算 CRC	常数 1	2 bits	0x0E001F01	发送 1bit 的 1
2	无时限发送	时钟线下降-上升	不计算 CRC	命令寄存器	6 bits	0x0E000105	使用命令寄存器发送 CMD10
3	无时限发送	时钟线下降-上升	不计算 CRC	数据寄存器 2	8 bits	0x0E000207	使用数据寄存器 2 发送 ADDRESS
4	无时限发送	时钟线下降-上升	不计算 CRC	数据寄存器 3	16 bits	0x0E00030F	使用数据寄存器 3 发送 16bits 数据内容
5	无时限发送	时钟线下降-上升	不计算 CRC	常数 1	1 bit	0x0E001F00	发送 1bit 的 1
6	无时限等待	时钟线下降-上升	不计算 CRC	常数 0	1 bit	0x16001E00	等待直至收到 1bit 的 0
7	无时限等待	时钟线下降-上升	不计算 CRC	常数 1	1 bit	0x16001F00	等待直至收到 Start bit
8	时限内接收	时钟线下降-上升	数据寄存器 7	数据寄存器 4	8 bits	0x1A070407	使用数据寄存器 4 接收 ADDRESS，并使用数据寄存器 7 计算 CRC
9	时限内接收	时钟线下降-上升	数据寄存器 7	数据寄存器 5	16 bits	0x1A07050F	使用数据寄存器接收 16bits 数据内容，并计算 CRC
10	时限内接收	时钟线下降-上升	不计算 CRC	数据寄存器 7	5 bits	0x1A000704	使用数据寄存器 7 接收 CRC 并比对
11	时限内接收	时钟线下降-上升	不计算 CRC	忽略数据	1 bit	0x1A000000	接收 1bit 的 1

INSTR	OP 字段	CK 字段	CRC 字段	DAT 字段	LEN 字段	寄存器配置	配置说明
12	无时限等待	时钟线为高	不计算 CRC	常数 0	1 bit	0x17001E00	拉高时钟线等待直至收到 1bit 的 0，等待 WatchDog 触发 Timeout

表 196: 主机模式

INSTR	OP 字段	CK 字段	CRC 字段	DAT 字段	LEN 字段	寄存器配置	配置说明
0	时限内接收	时钟线上升-下降	不计算 CRC	忽略数据	2 bits	0x19000001	接收 2 bit 的数据
1	时限内接收	时钟线上升-下降	不计算 CRC	命令寄存器	6 bits	0x19000105	使用命令寄存器接收命令
2	时限内接收	时钟线上升-下降	不计算 CRC	数据寄存器 4	8 bits	0x19000407	使用数据寄存器 4 接收 8 bits 的数据内容
3	时限内接收	时钟线上升-下降	不计算 CRC	数据寄存器 5	16 bits	0x1900050F	使用数据寄存器 5 接收 16 bits 的数据内容
4	时限内接收	时钟线上升-下降	不计算 CRC	忽略数据	1 bit	0x19000000	接收 1 bit 的数据
5	无时限发送	时钟线上升-下降	不计算 CRC	常数 0	1 bit	0x0D001E00	发送 1 bit 的常数 0
6	无时限发送	时钟线下降-上升	不计算 CRC	常数 0	3 bit	0x0E001E02	发送 3 bits 的常数 0
7	无时限发送	时钟线下降-上升	不计算 CRC	常数 1	1 bits	0x0E001F00	发送 1 bit 的常数 1
8	无时限发送	时钟线下降-上升	数据寄存器 6	数据寄存器 7	8 bits	0x0E060707	使用数据寄存器 7 发送 ADDRESS 数据，并使用数据寄存器 6 计算 CRC
9	无时限发送	时钟线下降-上升	数据寄存器 6	数据寄存器 8	16 bits	0x0E06080F	使用数据寄存器 8 发送 16bits 数据内容，并计算 CRC
10	无时限发送	时钟线下降-上升	不计算 CRC	数据寄存器 6	5 bits	0x0E000604	发送数据寄存器 6 中 CRC 的计算结果
11	无时限发送	时钟线下降-上升	不计算 CRC	常数 1	2 bits	0x0E001F01	发送 2 bits 的 1，等待 WatchDog 触发 Time-out

表 197: 从机模式

41.3 SEI 寄存器列表

SEI 的寄存器列表如下：

SEI base address: 0xF032C000

地址偏移	名称	描述	复位值
0x0000	CTRL[0][ENGINE][CTRL]	执行单元控制寄存器	0x00000000
0x0004	CTRL[0][ENGINE][PTR_CFG]	程序指针配置寄存器	0x00000000
0x0008	CTRL[0][ENGINE][WDG_CFG]	看门口狗设置	0x00000000
0x0010	CTRL[0][ENGINE][EXE_STA]	程序执行状态	0x00000000
0x0014	CTRL[0][ENGINE][EXE_PTR]	程序指针	0x00000000
0x0018	CTRL[0][ENGINE][EXE_INST]	当前指令	0x00000000
0x001C	CTRL[0][ENGINE][WDG_STA]	看门狗状态	0x00000000
0x0020	CTRL[0][XCVR][CTRL]	传输控制寄存器	0x00000000
0x0024	CTRL[0][XCVR][TYPE_CFG]	收发器配置寄存器	0x00000000
0x0028	CTRL[0][XCVR][BAUD_CFG]	波特率	0x00000000
0x002C	CTRL[0][XCVR][DATA_CFG]	数据时序	0x00000000
0x0030	CTRL[0][XCVR][CLK_CFG]	时钟时序	0x00000000
0x0038	CTRL[0][XCVR][PIN]	引脚状态	0x00000000
0x003C	CTRL[0][XCVR][STATE]	异步状态机状态	0x00000000
0x0040	CTRL[0][TRG][IN_CFG]	触发输入配置	0x00000000
0x0044	CTRL[0][TRG][SW]	软件触发	0x00000000
0x0048	CTRL[0][TRG][PRD_CFG]	周期性触发配置	0x00000000
0x004C	CTRL[0][TRG][PRD]	触发周期	0x00000000
0x0050	CTRL[0][TRG][OUT_CFG]	触发输出配置	0x00000000
0x0060	CTRL[0][TRG][PRD_STS]	周期性触发状态	0x00000000
0x0064	CTRL[0][TRG][PRD_CNT]	周期性触发计数器	0x00000000
0x0080	CTRL[0][TRG_TABLE][CMD][0]	触发命令	0x00000000
0x0084	CTRL[0][TRG_TABLE][CMD][1]	触发命令	0x00000000
0x0088	CTRL[0][TRG_TABLE][CMD][2]	触发命令	0x00000000
0x008C	CTRL[0][TRG_TABLE][CMD][3]	触发命令	0x00000000
0x00A0	CTRL[0][TRG_TABLE][TIME][0]	触发时间	0x00000000
0x00A4	CTRL[0][TRG_TABLE][TIME][1]	触发时间	0x00000000
0x00A8	CTRL[0][TRG_TABLE][TIME][2]	触发时间	0x00000000
0x00AC	CTRL[0][TRG_TABLE][TIME][3]	触发时间	0x00000000
0x00C0	CTRL[0][CMD][MODE]	命令设置	0x00000000
0x00C4	CTRL[0][CMD][IDX]	命令索引	0x00000000
0x00C8	CTRL[0][CMD][GOLD]	命令比较值	0x00000000
0x00CC	CTRL[0][CMD][CRCINIT]	命令初始 CRC	0x00000000
0x00D0	CTRL[0][CMD][CRCPOLY]	命令 CRC 多项式	0x00000000
0x00E0	CTRL[0][CMD][CMD]	命令	0x00000000
0x00E4	CTRL[0][CMD][SET]	命令置位寄存器	0x00000000

地址偏移	名称	描述	复位值
0x00E8	CTRL[0][CMD][CLR]	命令清零寄存器	0x00000000
0x00EC	CTRL[0][CMD][INV]	命令取反寄存器	0x00000000
0x00F0	CTRL[0][CMD][IN]	输入命令	0x00000000
0x00F4	CTRL[0][CMD][OUT]	输出命令	0x00000000
0x00F8	CTRL[0][CMD][STS]	命令传输状态	0x00000000
0x0100	CTRL[0][CMD_TABLE][0][MIN]	命令匹配最小值	0x00000000
0x0104	CTRL[0][CMD_TABLE][0][MAX]	命令匹配最大值	0x00000000
0x0108	CTRL[0][CMD_TABLE][0][MSK]	命令匹配比较位	0x00000000
0x0110	CTRL[0][CMD_TABLE][0][PTA]	命令指针 0-3	0x00000000
0x0114	CTRL[0][CMD_TABLE][0][PTB]	命令指针 4-7	0x00000000
0x0120	CTRL[0][CMD_TABLE][1][MIN]	命令匹配最小值	0x00000000
0x0124	CTRL[0][CMD_TABLE][1][MAX]	命令匹配最大值	0x00000000
0x0128	CTRL[0][CMD_TABLE][1][MSK]	命令匹配比较位	0x00000000
0x0130	CTRL[0][CMD_TABLE][1][PTA]	命令指针 0-3	0x00000000
0x0134	CTRL[0][CMD_TABLE][1][PTB]	命令指针 4-7	0x00000000
0x0140	CTRL[0][CMD_TABLE][2][MIN]	命令匹配最小值	0x00000000
0x0144	CTRL[0][CMD_TABLE][2][MAX]	命令匹配最大值	0x00000000
0x0148	CTRL[0][CMD_TABLE][2][MSK]	命令匹配比较位	0x00000000
0x0150	CTRL[0][CMD_TABLE][2][PTA]	命令指针 0-3	0x00000000
0x0154	CTRL[0][CMD_TABLE][2][PTB]	命令指针 4-7	0x00000000
0x0160	CTRL[0][CMD_TABLE][3][MIN]	命令匹配最小值	0x00000000
0x0164	CTRL[0][CMD_TABLE][3][MAX]	命令匹配最大值	0x00000000
0x0168	CTRL[0][CMD_TABLE][3][MSK]	命令匹配比较位	0x00000000
0x0170	CTRL[0][CMD_TABLE][3][PTA]	命令指针 0-3	0x00000000
0x0174	CTRL[0][CMD_TABLE][3][PTB]	命令指针 4-7	0x00000000
0x0180	CTRL[0][CMD_TABLE][4][MIN]	命令匹配最小值	0x00000000
0x0184	CTRL[0][CMD_TABLE][4][MAX]	命令匹配最大值	0x00000000
0x0188	CTRL[0][CMD_TABLE][4][MSK]	命令匹配比较位	0x00000000
0x0190	CTRL[0][CMD_TABLE][4][PTA]	命令指针 0-3	0x00000000
0x0194	CTRL[0][CMD_TABLE][4][PTB]	命令指针 4-7	0x00000000
0x01A0	CTRL[0][CMD_TABLE][5][MIN]	命令匹配最小值	0x00000000
0x01A4	CTRL[0][CMD_TABLE][5][MAX]	命令匹配最大值	0x00000000
0x01A8	CTRL[0][CMD_TABLE][5][MSK]	命令匹配比较位	0x00000000
0x01B0	CTRL[0][CMD_TABLE][5][PTA]	命令指针 0-3	0x00000000
0x01B4	CTRL[0][CMD_TABLE][5][PTB]	命令指针 4-7	0x00000000
0x01C0	CTRL[0][CMD_TABLE][6][MIN]	命令匹配最小值	0x00000000
0x01C4	CTRL[0][CMD_TABLE][6][MAX]	命令匹配最大值	0x00000000
0x01C8	CTRL[0][CMD_TABLE][6][MSK]	命令匹配比较位	0x00000000
0x01D0	CTRL[0][CMD_TABLE][6][PTA]	命令指针 0-3	0x00000000
0x01D4	CTRL[0][CMD_TABLE][6][PTB]	命令指针 4-7	0x00000000

地址偏移	名称	描述	复位值
0x01E0	CTRL[0][CMD_TABLE][7][MIN]	命令匹配最小值	0x00000000
0x01E4	CTRL[0][CMD_TABLE][7][MAX]	命令匹配最大值	0x00000000
0x01E8	CTRL[0][CMD_TABLE][7][MSK]	命令匹配比较位	0x00000000
0x01F0	CTRL[0][CMD_TABLE][7][PTA]	命令指针 0-3	0x00000000
0x01F4	CTRL[0][CMD_TABLE][7][PTB]	命令指针 4-7	0x00000000
0x0200	CTRL[0][LATCH][0][TRAN][0_1]	锁存状态 0 到 1 转移条件	0x00000000
0x0204	CTRL[0][LATCH][0][TRAN][1_2]	锁存状态 1 到 2 转移条件	0x00000000
0x0208	CTRL[0][LATCH][0][TRAN][2_3]	锁存状态 2 到 3 转移条件	0x00000000
0x020C	CTRL[0][LATCH][0][TRAN][3_0]	锁存状态 3 到 0 转移条件	0x00000000
0x0210	CTRL[0][LATCH][0][CFG]	锁存配置	0x00000000
0x0218	CTRL[0][LATCH][0][TIME]	锁存发生时间	0x00000000
0x021C	CTRL[0][LATCH][0][STS]	锁存状态	0x00000000
0x0220	CTRL[0][LATCH][1][TRAN][0_1]	锁存状态 0 到 1 转移条件	0x00000000
0x0224	CTRL[0][LATCH][1][TRAN][1_2]	锁存状态 1 到 2 转移条件	0x00000000
0x0228	CTRL[0][LATCH][1][TRAN][2_3]	锁存状态 2 到 3 转移条件	0x00000000
0x022C	CTRL[0][LATCH][1][TRAN][3_0]	锁存状态 3 到 0 转移条件	0x00000000
0x0230	CTRL[0][LATCH][1][CFG]	锁存配置	0x00000000
0x0238	CTRL[0][LATCH][1][TIME]	锁存发生时间	0x00000000
0x023C	CTRL[0][LATCH][1][STS]	锁存状态	0x00000000
0x0240	CTRL[0][LATCH][2][TRAN][0_1]	锁存状态 0 到 1 转移条件	0x00000000
0x0244	CTRL[0][LATCH][2][TRAN][1_2]	锁存状态 1 到 2 转移条件	0x00000000
0x0248	CTRL[0][LATCH][2][TRAN][2_3]	锁存状态 2 到 3 转移条件	0x00000000
0x024C	CTRL[0][LATCH][2][TRAN][3_0]	锁存状态 3 到 0 转移条件	0x00000000
0x0250	CTRL[0][LATCH][2][CFG]	锁存配置	0x00000000
0x0258	CTRL[0][LATCH][2][TIME]	锁存发生时间	0x00000000
0x025C	CTRL[0][LATCH][2][STS]	锁存状态	0x00000000
0x0260	CTRL[0][LATCH][3][TRAN][0_1]	锁存状态 0 到 1 转移条件	0x00000000
0x0264	CTRL[0][LATCH][3][TRAN][1_2]	锁存状态 1 到 2 转移条件	0x00000000
0x0268	CTRL[0][LATCH][3][TRAN][2_3]	锁存状态 2 到 3 转移条件	0x00000000
0x026C	CTRL[0][LATCH][3][TRAN][3_0]	锁存状态 3 到 0 转移条件	0x00000000
0x0270	CTRL[0][LATCH][3][CFG]	锁存配置	0x00000000
0x0278	CTRL[0][LATCH][3][TIME]	锁存发生时间	0x00000000
0x027C	CTRL[0][LATCH][3][STS]	锁存状态	0x00000000
0x0280	CTRL[0][POS][SMP_EN]	采样选择	0x00000000
0x0284	CTRL[0][POS][SMP_CFG]	采样配置	0x00000000
0x0288	CTRL[0][POS][SMP_DAT]	采样数据	0x00000000
0x0290	CTRL[0][POS][SMP_POS]	位置覆盖值	0x00000000
0x0294	CTRL[0][POS][SMP_REV]	圈数覆盖值	0x00000000
0x0298	CTRL[0][POS][SMP_SPD]	速度覆盖值	0x00000000
0x029C	CTRL[0][POS][SMP_ACC]	加速度覆盖值	0x00000000

地址偏移	名称	描述	复位值
0x02A0	CTRL[0][POS][UPD_EN]	更新选择	0x00000000
0x02A4	CTRL[0][POS][UPD_CFG]	更新配置	0x00000000
0x02A8	CTRL[0][POS][UPD_DAT]	采样数据	0x00000000
0x02AC	CTRL[0][POS][UPD_TIME]	时间覆盖值	0x00000000
0x02B0	CTRL[0][POS][UPD_POS]	位置覆盖值	0x00000000
0x02B4	CTRL[0][POS][UPD_REV]	圈数覆盖值	0x00000000
0x02B8	CTRL[0][POS][UPD_SPD]	速度覆盖值	0x00000000
0x02BC	CTRL[0][POS][UPD_ACC]	加速度覆盖值	0x00000000
0x02C0	CTRL[0][POS][SMP_VAL]	采样有效	0x00000000
0x02C4	CTRL[0][POS][SMP_STS]	采样状态	0x00000000
0x02CC	CTRL[0][POS][TIME_IN]	输入时间	0x00000000
0x02D0	CTRL[0][POS][POS_IN]	输入位置	0x00000000
0x02D4	CTRL[0][POS][REV_IN]	输入圈数	0x00000000
0x02D8	CTRL[0][POS][SPD_IN]	输入速度	0x00000000
0x02DC	CTRL[0][POS][ACC_IN]	输入加速度	0x00000000
0x02E4	CTRL[0][POS][UPD_STS]	更新状态	0x00000000
0x0300	CTRL[0][IRQ][INT_EN]	中断使能	0x00000000
0x0304	CTRL[0][IRQ][INT_FLAG]	中断标志	0x00000000
0x0308	CTRL[0][IRQ][INT_STS]	中断状态	0x00000000
0x0310	CTRL[0][IRQ][POINTER0]	匹配指针 0	0x00000000
0x0314	CTRL[0][IRQ][POINTER1]	匹配指针 1	0x00000000
0x0318	CTRL[0][IRQ][INSTR0]	匹配指令 0	0x00000000
0x031C	CTRL[0][IRQ][INSTR1]	匹配指令 1	0x00000000
0x0400	CTRL[1][ENGINE][CTRL]	执行单元控制寄存器	0x00000000
0x0404	CTRL[1][ENGINE][PTR_CFG]	程序指针配置寄存器	0x00000000
0x0408	CTRL[1][ENGINE][WDG_CFG]	看门口狗设置	0x00000000
0x0410	CTRL[1][ENGINE][EXE_STA]	程序执行状态	0x00000000
0x0414	CTRL[1][ENGINE][EXE_PTR]	程序指针	0x00000000
0x0418	CTRL[1][ENGINE][EXE_INST]	当前指令	0x00000000
0x041C	CTRL[1][ENGINE][WDG_STA]	看门狗状态	0x00000000
0x0400	CTRL[1][XCVR][CTRL]	传输控制寄存器	0x00000000
0x0404	CTRL[1][XCVR][TYPE_CFG]	收发器配置寄存器	0x00000000
0x0408	CTRL[1][XCVR][BAUD_CFG]	波特率	0x00000000
0x040C	CTRL[1][XCVR][DATA_CFG]	数据时序	0x00000000
0x0410	CTRL[1][XCVR][CLK_CFG]	时钟时序	0x00000000
0x0418	CTRL[1][XCVR][PIN]	引脚状态	0x00000000
0x041C	CTRL[1][XCVR][STATE]	异步状态机状态	0x00000000
0x0400	CTRL[1][TRG][IN_CFG]	触发输入配置	0x00000000
0x0404	CTRL[1][TRG][SW]	软件触发	0x00000000
0x0408	CTRL[1][TRG][PRD_CFG]	周期性触发配置	0x00000000

地址偏移	名称	描述	复位值
0x040C	CTRL[1][TRG][PRD]	触发周期	0x00000000
0x0410	CTRL[1][TRG][OUT_CFG]	触发输出配置	0x00000000
0x0420	CTRL[1][TRG][PRD_STS]	周期性触发状态	0x00000000
0x0424	CTRL[1][TRG][PRD_CNT]	周期性触发计数器	0x00000000
0x0400	CTRL[1][TRG_TABLE][CMD][0]	触发命令	0x00000000
0x0404	CTRL[1][TRG_TABLE][CMD][1]	触发命令	0x00000000
0x0408	CTRL[1][TRG_TABLE][CMD][2]	触发命令	0x00000000
0x040C	CTRL[1][TRG_TABLE][CMD][3]	触发命令	0x00000000
0x0420	CTRL[1][TRG_TABLE][TIME][0]	触发时间	0x00000000
0x0424	CTRL[1][TRG_TABLE][TIME][1]	触发时间	0x00000000
0x0428	CTRL[1][TRG_TABLE][TIME][2]	触发时间	0x00000000
0x042C	CTRL[1][TRG_TABLE][TIME][3]	触发时间	0x00000000
0x0400	CTRL[1][CMD][MODE]	命令设置	0x00000000
0x0404	CTRL[1][CMD][IDX]	命令索引	0x00000000
0x0408	CTRL[1][CMD][GOLD]	命令比较值	0x00000000
0x040C	CTRL[1][CMD][CRCINIT]	命令初始 CRC	0x00000000
0x0410	CTRL[1][CMD][CRCPOLY]	命令 CRC 多项式	0x00000000
0x0420	CTRL[1][CMD][CMD]	命令	0x00000000
0x0424	CTRL[1][CMD][SET]	命令置位寄存器	0x00000000
0x0428	CTRL[1][CMD][CLR]	命令清零寄存器	0x00000000
0x042C	CTRL[1][CMD][INV]	命令取反寄存器	0x00000000
0x0430	CTRL[1][CMD][IN]	输入命令	0x00000000
0x0434	CTRL[1][CMD][OUT]	输出命令	0x00000000
0x0438	CTRL[1][CMD][STS]	命令传输状态	0x00000000
0x0400	CTRL[1][CMD_TABLE][0][MIN]	命令匹配最小值	0x00000000
0x0404	CTRL[1][CMD_TABLE][0][MAX]	命令匹配最大值	0x00000000
0x0408	CTRL[1][CMD_TABLE][0][MSK]	命令匹配比较位	0x00000000
0x0410	CTRL[1][CMD_TABLE][0][PTA]	命令指针 0-3	0x00000000
0x0414	CTRL[1][CMD_TABLE][0][PTB]	命令指针 4-7	0x00000000
0x0420	CTRL[1][CMD_TABLE][1][MIN]	命令匹配最小值	0x00000000
0x0424	CTRL[1][CMD_TABLE][1][MAX]	命令匹配最大值	0x00000000
0x0428	CTRL[1][CMD_TABLE][1][MSK]	命令匹配比较位	0x00000000
0x0430	CTRL[1][CMD_TABLE][1][PTA]	命令指针 0-3	0x00000000
0x0434	CTRL[1][CMD_TABLE][1][PTB]	命令指针 4-7	0x00000000
0x0440	CTRL[1][CMD_TABLE][2][MIN]	命令匹配最小值	0x00000000
0x0444	CTRL[1][CMD_TABLE][2][MAX]	命令匹配最大值	0x00000000
0x0448	CTRL[1][CMD_TABLE][2][MSK]	命令匹配比较位	0x00000000
0x0450	CTRL[1][CMD_TABLE][2][PTA]	命令指针 0-3	0x00000000
0x0454	CTRL[1][CMD_TABLE][2][PTB]	命令指针 4-7	0x00000000
0x0460	CTRL[1][CMD_TABLE][3][MIN]	命令匹配最小值	0x00000000

地址偏移	名称	描述	复位值
0x0464	CTRL[1][CMD_TABLE][3][MAX]	命令匹配最大值	0x00000000
0x0468	CTRL[1][CMD_TABLE][3][MSK]	命令匹配比较位	0x00000000
0x0470	CTRL[1][CMD_TABLE][3][PTA]	命令指针 0-3	0x00000000
0x0474	CTRL[1][CMD_TABLE][3][PTB]	命令指针 4-7	0x00000000
0x0480	CTRL[1][CMD_TABLE][4][MIN]	命令匹配最小值	0x00000000
0x0484	CTRL[1][CMD_TABLE][4][MAX]	命令匹配最大值	0x00000000
0x0488	CTRL[1][CMD_TABLE][4][MSK]	命令匹配比较位	0x00000000
0x0490	CTRL[1][CMD_TABLE][4][PTA]	命令指针 0-3	0x00000000
0x0494	CTRL[1][CMD_TABLE][4][PTB]	命令指针 4-7	0x00000000
0x04A0	CTRL[1][CMD_TABLE][5][MIN]	命令匹配最小值	0x00000000
0x04A4	CTRL[1][CMD_TABLE][5][MAX]	命令匹配最大值	0x00000000
0x04A8	CTRL[1][CMD_TABLE][5][MSK]	命令匹配比较位	0x00000000
0x04B0	CTRL[1][CMD_TABLE][5][PTA]	命令指针 0-3	0x00000000
0x04B4	CTRL[1][CMD_TABLE][5][PTB]	命令指针 4-7	0x00000000
0x04C0	CTRL[1][CMD_TABLE][6][MIN]	命令匹配最小值	0x00000000
0x04C4	CTRL[1][CMD_TABLE][6][MAX]	命令匹配最大值	0x00000000
0x04C8	CTRL[1][CMD_TABLE][6][MSK]	命令匹配比较位	0x00000000
0x04D0	CTRL[1][CMD_TABLE][6][PTA]	命令指针 0-3	0x00000000
0x04D4	CTRL[1][CMD_TABLE][6][PTB]	命令指针 4-7	0x00000000
0x04E0	CTRL[1][CMD_TABLE][7][MIN]	命令匹配最小值	0x00000000
0x04E4	CTRL[1][CMD_TABLE][7][MAX]	命令匹配最大值	0x00000000
0x04E8	CTRL[1][CMD_TABLE][7][MSK]	命令匹配比较位	0x00000000
0x04F0	CTRL[1][CMD_TABLE][7][PTA]	命令指针 0-3	0x00000000
0x04F4	CTRL[1][CMD_TABLE][7][PTB]	命令指针 4-7	0x00000000
0x0400	CTRL[1][LATCH][0][TRAN][0_1]	锁存状态 0 到 1 转移条件	0x00000000
0x0404	CTRL[1][LATCH][0][TRAN][1_2]	锁存状态 1 到 2 转移条件	0x00000000
0x0408	CTRL[1][LATCH][0][TRAN][2_3]	锁存状态 2 到 3 转移条件	0x00000000
0x040C	CTRL[1][LATCH][0][TRAN][3_0]	锁存状态 3 到 0 转移条件	0x00000000
0x0410	CTRL[1][LATCH][0][CFG]	锁存配置	0x00000000
0x0418	CTRL[1][LATCH][0][TIME]	锁存发生时间	0x00000000
0x041C	CTRL[1][LATCH][0][STS]	锁存状态	0x00000000
0x0420	CTRL[1][LATCH][1][TRAN][0_1]	锁存状态 0 到 1 转移条件	0x00000000
0x0424	CTRL[1][LATCH][1][TRAN][1_2]	锁存状态 1 到 2 转移条件	0x00000000
0x0428	CTRL[1][LATCH][1][TRAN][2_3]	锁存状态 2 到 3 转移条件	0x00000000
0x042C	CTRL[1][LATCH][1][TRAN][3_0]	锁存状态 3 到 0 转移条件	0x00000000
0x0430	CTRL[1][LATCH][1][CFG]	锁存配置	0x00000000
0x0438	CTRL[1][LATCH][1][TIME]	锁存发生时间	0x00000000
0x043C	CTRL[1][LATCH][1][STS]	锁存状态	0x00000000
0x0440	CTRL[1][LATCH][2][TRAN][0_1]	锁存状态 0 到 1 转移条件	0x00000000
0x0444	CTRL[1][LATCH][2][TRAN][1_2]	锁存状态 1 到 2 转移条件	0x00000000

地址偏移	名称	描述	复位值
0x0448	CTRL[1][LATCH][2][TRAN][2_3]	锁存状态 2 到 3 转移条件	0x00000000
0x044C	CTRL[1][LATCH][2][TRAN][3_0]	锁存状态 3 到 0 转移条件	0x00000000
0x0450	CTRL[1][LATCH][2][CFG]	锁存配置	0x00000000
0x0458	CTRL[1][LATCH][2][TIME]	锁存发生时间	0x00000000
0x045C	CTRL[1][LATCH][2][STS]	锁存状态	0x00000000
0x0460	CTRL[1][LATCH][3][TRAN][0_1]	锁存状态 0 到 1 转移条件	0x00000000
0x0464	CTRL[1][LATCH][3][TRAN][1_2]	锁存状态 1 到 2 转移条件	0x00000000
0x0468	CTRL[1][LATCH][3][TRAN][2_3]	锁存状态 2 到 3 转移条件	0x00000000
0x046C	CTRL[1][LATCH][3][TRAN][3_0]	锁存状态 3 到 0 转移条件	0x00000000
0x0470	CTRL[1][LATCH][3][CFG]	锁存配置	0x00000000
0x0478	CTRL[1][LATCH][3][TIME]	锁存发生时间	0x00000000
0x047C	CTRL[1][LATCH][3][STS]	锁存状态	0x00000000
0x0400	CTRL[1][POS][SMP_EN]	采样选择	0x00000000
0x0404	CTRL[1][POS][SMP_CFG]	采样配置	0x00000000
0x0408	CTRL[1][POS][SMP_DAT]	采样数据	0x00000000
0x0410	CTRL[1][POS][SMP_POS]	位置覆盖值	0x00000000
0x0414	CTRL[1][POS][SMP_REV]	圈数覆盖值	0x00000000
0x0418	CTRL[1][POS][SMP_SPD]	速度覆盖值	0x00000000
0x041C	CTRL[1][POS][SMP_ACC]	加速度覆盖值	0x00000000
0x0420	CTRL[1][POS][UPD_EN]	更新选择	0x00000000
0x0424	CTRL[1][POS][UPD_CFG]	更新配置	0x00000000
0x0428	CTRL[1][POS][UPD_DAT]	采样数据	0x00000000
0x042C	CTRL[1][POS][UPD_TIME]	时间覆盖值	0x00000000
0x0430	CTRL[1][POS][UPD_POS]	位置覆盖值	0x00000000
0x0434	CTRL[1][POS][UPD_REV]	圈数覆盖值	0x00000000
0x0438	CTRL[1][POS][UPD_SPD]	速度覆盖值	0x00000000
0x043C	CTRL[1][POS][UPD_ACC]	加速度覆盖值	0x00000000
0x0440	CTRL[1][POS][SMP_VAL]	采样有效	0x00000000
0x0444	CTRL[1][POS][SMP_STS]	采样状态	0x00000000
0x044C	CTRL[1][POS][TIME_IN]	输入时间	0x00000000
0x0450	CTRL[1][POS][POS_IN]	输入位置	0x00000000
0x0454	CTRL[1][POS][REV_IN]	输入圈数	0x00000000
0x0458	CTRL[1][POS][SPD_IN]	输入速度	0x00000000
0x045C	CTRL[1][POS][ACC_IN]	输入加速度	0x00000000
0x0464	CTRL[1][POS][UPD_STS]	更新状态	0x00000000
0x0400	CTRL[1][IRQ][INT_EN]	中断使能	0x00000000
0x0404	CTRL[1][IRQ][INT_FLAG]	中断标志	0x00000000
0x0408	CTRL[1][IRQ][INT_STS]	中断状态	0x00000000
0x0410	CTRL[1][IRQ][POINTER0]	匹配指针 0	0x00000000
0x0414	CTRL[1][IRQ][POINTER1]	匹配指针 1	0x00000000

地址偏移	名称	描述	复位值
0x0418	CTRL[1][IRQ][INSTR0]	匹配指令 0	0x00000000
0x041C	CTRL[1][IRQ][INSTR1]	匹配指令 1	0x00000000
0x3400	INSTR[0]	指令	0x00000000
0x3404	INSTR[1]	指令	0x00000000
0x3408	INSTR[2]	指令	0x00000000
0x340C	INSTR[3]	指令	0x00000000
0x3410	INSTR[4]	指令	0x00000000
0x3414	INSTR[5]	指令	0x00000000
0x3418	INSTR[6]	指令	0x00000000
0x341C	INSTR[7]	指令	0x00000000
0x3420	INSTR[8]	指令	0x00000000
0x3424	INSTR[9]	指令	0x00000000
0x3428	INSTR[10]	指令	0x00000000
0x342C	INSTR[11]	指令	0x00000000
0x3430	INSTR[12]	指令	0x00000000
0x3434	INSTR[13]	指令	0x00000000
0x3438	INSTR[14]	指令	0x00000000
0x343C	INSTR[15]	指令	0x00000000
0x3440	INSTR[16]	指令	0x00000000
0x3444	INSTR[17]	指令	0x00000000
0x3448	INSTR[18]	指令	0x00000000
0x344C	INSTR[19]	指令	0x00000000
0x3450	INSTR[20]	指令	0x00000000
0x3454	INSTR[21]	指令	0x00000000
0x3458	INSTR[22]	指令	0x00000000
0x345C	INSTR[23]	指令	0x00000000
0x3460	INSTR[24]	指令	0x00000000
0x3464	INSTR[25]	指令	0x00000000
0x3468	INSTR[26]	指令	0x00000000
0x346C	INSTR[27]	指令	0x00000000
0x3470	INSTR[28]	指令	0x00000000
0x3474	INSTR[29]	指令	0x00000000
0x3478	INSTR[30]	指令	0x00000000
0x347C	INSTR[31]	指令	0x00000000
0x3480	INSTR[32]	指令	0x00000000
0x3484	INSTR[33]	指令	0x00000000
0x3488	INSTR[34]	指令	0x00000000
0x348C	INSTR[35]	指令	0x00000000
0x3490	INSTR[36]	指令	0x00000000
0x3494	INSTR[37]	指令	0x00000000

地址偏移	名称	描述	复位值
0x3498	INSTR[38]	指令	0x00000000
0x349C	INSTR[39]	指令	0x00000000
0x34A0	INSTR[40]	指令	0x00000000
0x34A4	INSTR[41]	指令	0x00000000
0x34A8	INSTR[42]	指令	0x00000000
0x34AC	INSTR[43]	指令	0x00000000
0x34B0	INSTR[44]	指令	0x00000000
0x34B4	INSTR[45]	指令	0x00000000
0x34B8	INSTR[46]	指令	0x00000000
0x34BC	INSTR[47]	指令	0x00000000
0x34C0	INSTR[48]	指令	0x00000000
0x34C4	INSTR[49]	指令	0x00000000
0x34C8	INSTR[50]	指令	0x00000000
0x34CC	INSTR[51]	指令	0x00000000
0x34D0	INSTR[52]	指令	0x00000000
0x34D4	INSTR[53]	指令	0x00000000
0x34D8	INSTR[54]	指令	0x00000000
0x34DC	INSTR[55]	指令	0x00000000
0x34E0	INSTR[56]	指令	0x00000000
0x34E4	INSTR[57]	指令	0x00000000
0x34E8	INSTR[58]	指令	0x00000000
0x34EC	INSTR[59]	指令	0x00000000
0x34F0	INSTR[60]	指令	0x00000000
0x34F4	INSTR[61]	指令	0x00000000
0x34F8	INSTR[62]	指令	0x00000000
0x34FC	INSTR[63]	指令	0x00000000
0x3800	DAT[0][MODE]	数据模式	0x00000000
0x3804	DAT[0][IDX]	数据索引	0x00000000
0x3808	DAT[0][GOLD]	数据比较值	0x00000000
0x380C	DAT[0][CRCINIT]	数据初始 CRC	0x00000000
0x3810	DAT[0][CRCPOLY]	数据 CRC 多项式	0x00000000
0x3820	DAT[0][DATA]	数据	0x00000000
0x3824	DAT[0][SET]	数据置位寄存器	0x00000000
0x3828	DAT[0][CLR]	数据清零寄存器	0x00000000
0x382C	DAT[0][INV]	数据取反寄存器	0x00000000
0x3830	DAT[0][IN]	输入数据	0x00000000
0x3834	DAT[0][OUT]	输出数据	0x00000000
0x3838	DAT[0][STS]	数据传输状态	0x00000000
0x3840	DAT[1][MODE]	数据模式	0x00000000
0x3844	DAT[1][IDX]	数据索引	0x00000000

地址偏移	名称	描述	复位值
0x3848	DAT[1][GOLD]	数据比较值	0x00000000
0x384C	DAT[1][CRCINIT]	数据初始 CRC	0x00000000
0x3850	DAT[1][CRCPOLY]	数据 CRC 多项式	0x00000000
0x3860	DAT[1][DATA]	数据	0x00000000
0x3864	DAT[1][SET]	数据置位寄存器	0x00000000
0x3868	DAT[1][CLR]	数据清零寄存器	0x00000000
0x386C	DAT[1][INV]	数据取反寄存器	0x00000000
0x3870	DAT[1][IN]	输入数据	0x00000000
0x3874	DAT[1][OUT]	输出数据	0x00000000
0x3878	DAT[1][STS]	数据传输状态	0x00000000
0x3880	DAT[2][MODE]	数据模式	0x00000000
0x3884	DAT[2][IDX]	数据索引	0x00000000
0x3888	DAT[2][GOLD]	数据比较值	0x00000000
0x388C	DAT[2][CRCINIT]	数据初始 CRC	0x00000000
0x3890	DAT[2][CRCPOLY]	数据 CRC 多项式	0x00000000
0x38A0	DAT[2][DATA]	数据	0x00000000
0x38A4	DAT[2][SET]	数据置位寄存器	0x00000000
0x38A8	DAT[2][CLR]	数据清零寄存器	0x00000000
0x38AC	DAT[2][INV]	数据取反寄存器	0x00000000
0x38B0	DAT[2][IN]	输入数据	0x00000000
0x38B4	DAT[2][OUT]	输出数据	0x00000000
0x38B8	DAT[2][STS]	数据传输状态	0x00000000
0x38C0	DAT[3][MODE]	数据模式	0x00000000
0x38C4	DAT[3][IDX]	数据索引	0x00000000
0x38C8	DAT[3][GOLD]	数据比较值	0x00000000
0x38CC	DAT[3][CRCINIT]	数据初始 CRC	0x00000000
0x38D0	DAT[3][CRCPOLY]	数据 CRC 多项式	0x00000000
0x38E0	DAT[3][DATA]	数据	0x00000000
0x38E4	DAT[3][SET]	数据置位寄存器	0x00000000
0x38E8	DAT[3][CLR]	数据清零寄存器	0x00000000
0x38EC	DAT[3][INV]	数据取反寄存器	0x00000000
0x38F0	DAT[3][IN]	输入数据	0x00000000
0x38F4	DAT[3][OUT]	输出数据	0x00000000
0x38F8	DAT[3][STS]	数据传输状态	0x00000000
0x3900	DAT[4][MODE]	数据模式	0x00000000
0x3904	DAT[4][IDX]	数据索引	0x00000000
0x3908	DAT[4][GOLD]	数据比较值	0x00000000
0x390C	DAT[4][CRCINIT]	数据初始 CRC	0x00000000
0x3910	DAT[4][CRCPOLY]	数据 CRC 多项式	0x00000000
0x3920	DAT[4][DATA]	数据	0x00000000

地址偏移	名称	描述	复位值
0x3924	DAT[4][SET]	数据置位寄存器	0x00000000
0x3928	DAT[4][CLR]	数据清零寄存器	0x00000000
0x392C	DAT[4][INV]	数据取反寄存器	0x00000000
0x3930	DAT[4][IN]	输入数据	0x00000000
0x3934	DAT[4][OUT]	输出数据	0x00000000
0x3938	DAT[4][STS]	数据传输状态	0x00000000
0x3940	DAT[5][MODE]	数据模式	0x00000000
0x3944	DAT[5][IDX]	数据索引	0x00000000
0x3948	DAT[5][GOLD]	数据比较值	0x00000000
0x394C	DAT[5][CRCINIT]	数据初始 CRC	0x00000000
0x3950	DAT[5][CRCPOLY]	数据 CRC 多项式	0x00000000
0x3960	DAT[5][DATA]	数据	0x00000000
0x3964	DAT[5][SET]	数据置位寄存器	0x00000000
0x3968	DAT[5][CLR]	数据清零寄存器	0x00000000
0x396C	DAT[5][INV]	数据取反寄存器	0x00000000
0x3970	DAT[5][IN]	输入数据	0x00000000
0x3974	DAT[5][OUT]	输出数据	0x00000000
0x3978	DAT[5][STS]	数据传输状态	0x00000000
0x3980	DAT[6][MODE]	数据模式	0x00000000
0x3984	DAT[6][IDX]	数据索引	0x00000000
0x3988	DAT[6][GOLD]	数据比较值	0x00000000
0x398C	DAT[6][CRCINIT]	数据初始 CRC	0x00000000
0x3990	DAT[6][CRCPOLY]	数据 CRC 多项式	0x00000000
0x39A0	DAT[6][DATA]	数据	0x00000000
0x39A4	DAT[6][SET]	数据置位寄存器	0x00000000
0x39A8	DAT[6][CLR]	数据清零寄存器	0x00000000
0x39AC	DAT[6][INV]	数据取反寄存器	0x00000000
0x39B0	DAT[6][IN]	输入数据	0x00000000
0x39B4	DAT[6][OUT]	输出数据	0x00000000
0x39B8	DAT[6][STS]	数据传输状态	0x00000000
0x39C0	DAT[7][MODE]	数据模式	0x00000000
0x39C4	DAT[7][IDX]	数据索引	0x00000000
0x39C8	DAT[7][GOLD]	数据比较值	0x00000000
0x39CC	DAT[7][CRCINIT]	数据初始 CRC	0x00000000
0x39D0	DAT[7][CRCPOLY]	数据 CRC 多项式	0x00000000
0x39E0	DAT[7][DATA]	数据	0x00000000
0x39E4	DAT[7][SET]	数据置位寄存器	0x00000000
0x39E8	DAT[7][CLR]	数据清零寄存器	0x00000000
0x39EC	DAT[7][INV]	数据取反寄存器	0x00000000
0x39F0	DAT[7][IN]	输入数据	0x00000000

地址偏移	名称	描述	复位值
0x39F4	DAT[7][OUT]	输出数据	0x00000000
0x39F8	DAT[7][STS]	数据传输状态	0x00000000
0x3A00	DAT[8][MODE]	数据模式	0x00000000
0x3A04	DAT[8][IDX]	数据索引	0x00000000
0x3A08	DAT[8][GOLD]	数据比较值	0x00000000
0x3A0C	DAT[8][CRCINIT]	数据初始 CRC	0x00000000
0x3A10	DAT[8][CRCPOLY]	数据 CRC 多项式	0x00000000
0x3A20	DAT[8][DATA]	数据	0x00000000
0x3A24	DAT[8][SET]	数据置位寄存器	0x00000000
0x3A28	DAT[8][CLR]	数据清零寄存器	0x00000000
0x3A2C	DAT[8][INV]	数据取反寄存器	0x00000000
0x3A30	DAT[8][IN]	输入数据	0x00000000
0x3A34	DAT[8][OUT]	输出数据	0x00000000
0x3A38	DAT[8][STS]	数据传输状态	0x00000000
0x3A40	DAT[9][MODE]	数据模式	0x00000000
0x3A44	DAT[9][IDX]	数据索引	0x00000000
0x3A48	DAT[9][GOLD]	数据比较值	0x00000000
0x3A4C	DAT[9][CRCINIT]	数据初始 CRC	0x00000000
0x3A50	DAT[9][CRCPOLY]	数据 CRC 多项式	0x00000000
0x3A60	DAT[9][DATA]	数据	0x00000000
0x3A64	DAT[9][SET]	数据置位寄存器	0x00000000
0x3A68	DAT[9][CLR]	数据清零寄存器	0x00000000
0x3A6C	DAT[9][INV]	数据取反寄存器	0x00000000
0x3A70	DAT[9][IN]	输入数据	0x00000000
0x3A74	DAT[9][OUT]	输出数据	0x00000000
0x3A78	DAT[9][STS]	数据传输状态	0x00000000

表 198: SEI 寄存器列表

41.4 SEI 寄存器详细信息

SEI 的寄存器详细说明如下：

41.4.1 CTRL[ENGINE][CTRL] (0x0 + 0x400 * n)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD							WATCH	RSVD							ARMING	RSVD							EXCEPT	RSVD			REWIND	RSVD			ENABLE
N/A							RW	N/A							RW	N/A							RW	N/A			RW	N/A			RW
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

CTRL[ENGINE][CTRL] [31:0]

位域	名称	描述
24	WATCH	允许指令看门狗 0：指令看门狗关闭 1：指令看门狗开启
16	ARMING	第一次执行时，等待触发 0：直接执行程序 1：等待触发再执行程序
8	EXCEPT	将超时解释为异常 0：超时发生后执行下一条指令 1：超时发生后跳转到超时处理向量
4	REWIND	重新执行 0：保持执行 1：清除执行状态，并跳转到初始向量
0	ENABLE	允许执行 0：复位执行状态，禁止执行 1：开始执行

CTRL[ENGINE][CTRL] 位域

41.4.2 CTRL[ENGINE][PTR_CFG] (0x4 + 0x400 * n)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD			DAT_CDM					RSVD			DAT_BASE					POINTER_WDOG								POINTER_INIT							
N/A			RW					N/A			RW					RW								RW							
0	0	0	0	0	0	0	0	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

CTRL[ENGINE][PTR_CFG] [31:0]

位域	名称	描述
28-24	DAT_CDM	BiSS-C 从机模式 CDM 位存储寄存器选择 0：忽略数据 1：命令寄存器 2：数据寄存器 2 3：数据寄存器 3 ... 29：数据寄存器 29 30：常数 0 31：常数 1

位域	名称	描述
20-16	DAT_BASE	数据偏置，寄存器寻址中增加的偏移量，若偏移后的编号超过 32，则会绕回 0 地址 0：数据寄存器无偏移 1：数据寄存器偏移 1 ... 31：数据寄存器偏移 31
15-8	POINTER_WDOG	指令超时后，程序开始执行的指令指针。超时时间为 WDOG_TIME。
7-0	POINTER_INIT	初始向量，程序开始执行的指令指针

CTRL[ENGINE][PTR_CFG] 位域

41.4.3 CTRL[ENGINE][WDG_CFG] (0x8 + 0x400 * n)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																WDOG_TIME															
N/A																RW															
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

CTRL[ENGINE][WDG_CFG] [31:0]

位域	名称	描述
15-0	WDOG_TIME	指令超时时间，以位时钟计数

CTRL[ENGINE][WDG_CFG] 位域

41.4.4 CTRL[ENGINE][EXE_STA] (0x10 + 0x400 * n)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD											TRIGERED	RSVD			ARMED	RSVD							EXPIRE	RSVD							STALL
N/A											RO	N/A			RO	N/A							RO	N/A							RO
x	x	x	x	x	x	x	x	x	x	x	0	x	x	x	0	x	x	x	x	x	x	x	0	x	x	x	x	x	x	x	0

CTRL[ENGINE][EXE_STA] [31:0]

位域	名称	描述
20	TRIGERED	程序执行已被触发 0：程序执行未被触发 1：程序执行已被触发

位域	名称	描述
16	ARMED	等待触发执行 0：不处于等待 1：等待触发
8	EXPIRE	执行发生超时 0：未发生超时 1：执行超时
0	STALL	执行结束 0：程序正在执行 1：程序已结束

CTRL[ENGINE][EXE_STA] 位域

41.4.5 CTRL[ENGINE][EXE_PTR] (0x14 + 0x400 * n)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD			HALT_CNT					RSVD			BIT_CNT					RSVD								POINTER							
N/A			RO					N/A			RO					N/A								RO							
x	x	x	0	0	0	0	0	x	x	x	0	0	0	0	0	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0

CTRL[ENGINE][EXE_PTR] [31:0]

位域	名称	描述
28-24	HALT_CNT	停止位计数
20-16	BIT_CNT	数据传输位计数
7-0	POINTER	当前程序指针

CTRL[ENGINE][EXE_PTR] 位域

41.4.6 CTRL[ENGINE][EXE_INST] (0x18 + 0x400 * n)

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

CTRL[ENGINE][EXE_INST] [31:0]

位域	名称	描述
31-0	INST	当前指令

CTRL[ENGINE][EXE_INST] 位域

41.4.7 CTRL[ENGINE][WDG_STA] (0x1C + 0x400 * n)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																WDOG_CNT															
N/A																RO															
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

CTRL[ENGINE][WDG_STA] [31:0]

位域	名称	描述
15-0	WDOG_CNT	当前看门狗计数器数值

CTRL[ENGINE][WDG_STA] 位域

41.4.8 CTRL[XCVR][CTRL] (0x20 + 0x400 * n)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																			TRISMP	RSVD			PAR_CLR	RSVD			RESTART	RSVD		MODE	
N/A																			RW	N/A			WC	N/A			WC	N/A		RW	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

CTRL[XCVR][CTRL] [31:0]

位域	名称	描述
12	TRISMP	三重采样 0：数据采样仅传输一次 1：数据传输三次，三取二
8	PAR_CLR	清除奇偶校验错，此位自动清零 0：无操作 1：清除奇偶校验错
4	RESTART	复位收发器开始，此位自动清零 0：无操作 1：复位收发器
1-0	MODE	收发器模式 0：同步主模式 1：同步从模式 2：异步模式 3：异步模式

位域	名称	描述

CTRL[XCVR][CTRL] 位域

41.4.9 CTRL[XCVR][TYPE_CFG] (0x24 + 0x400 * n)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WAIT_LEN								RSVD			DATA_LEN					RSVD						PAR_POL	PAR_EN	RSVD				DA_IDLEZ	CK_IDLEZ	DA_IDLEV	CK_IDLEV
RW								N/A			RW					N/A						RW	RW	N/A				RW	RW	RW	RW
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

CTRL[XCVR][TYPE_CFG] [31:0]

位域	名称	描述
31-24	WAIT_LEN	等待位长度 0：0 位 1：1 位 ... 255：255 位
20-16	DATA_LEN	数据长度 0：1 位 1：2 位 ... 31：32 位
9	PAR_POL	奇偶校验极性 0：偶校验 1：奇校验
8	PAR_EN	允许奇偶校验 0：禁止奇偶校验 1：允许奇偶校验
3	DA_IDLEZ	空闲态数据线驱动状态 0：输出 1：高阻
2	CK_IDLEZ	空闲态时钟线驱动状态 0：输出 1：高阻
1	DA_IDLEV	空闲态数据线数据状态 0：数据 0 1：数据 1

位域	名称	描述
0	CK_IDLEV	空闲态时钟线数据状态 0：数据 0 1：数据 1

CTRL[XCVR][TYPE_CFG] 位域

41.4.10 CTRL[XCVR][BAUD_CFG] (0x28 + 0x400 * n)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SYNC_POINT																BAUD_DIV															
RW																RW															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

CTRL[XCVR][BAUD_CFG] [31:0]

位域	名称	描述
31-16	SYNC_POINT	波特率同步范围，最小可能位时间
15-0	BAUD_DIV	波特率分频器

CTRL[XCVR][BAUD_CFG] 位域

41.4.11 CTRL[XCVR][DATA_CFG] (0x2C + 0x400 * n)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TXD_POINT																RXD_POINT															
RW																RW															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

CTRL[XCVR][DATA_CFG] [31:0]

位域	名称	描述
31-16	TXD_POINT	数据发送点
15-0	RXD_POINT	数据接收点

CTRL[XCVR][DATA_CFG] 位域

41.4.12 CTRL[XCVR][CLK_CFG] (0x30 + 0x400 * n)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CK1_POINT																CK0_POINT															
RW																RW															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

CTRL[XCVR][CLK_CFG] [31:0]

位域	名称	描述
31-16	CK1_POINT	时钟跳变点 1
15-0	CK0_POINT	时钟跳变点 0

CTRL[XCVR][CLK_CFG] 位域

41.4.13 CTRL[XCVR][PIN] (0x38 + 0x400 * n)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD					OE_CK	DI_CK	DO_CK	RSVD					OE_RX	DI_RX	DO_RX	RSVD					OE_DE	DI_DE	DO_DE	RSVD					OE_TX	DI_TX	DO_TX
N/A					RO	RO	RO	N/A					RO	RO	RO	N/A					RO	RO	RO	N/A					RO	RO	RO
x	x	x	x	x	0	0	0	x	x	x	x	x	0	0	0	x	x	x	x	x	0	0	0	x	x	x	x	x	0	0	0

CTRL[XCVR][PIN] [31:0]

位域	名称	描述
26	OE_CK	CK 引脚驱动状态 0：输入 1：输出
25	DI_CK	CK 引脚状态 0：数据 0 1：数据 1
24	DO_CK	CK 引脚输出状态 0：数据 0 1：数据 1
18	OE_RX	RX 引脚驱动状态 0：输入 1：输出
17	DI_RX	RX 引脚状态 0：数据 0 1：数据 1
16	DO_RX	RX 引脚输出状态 0：数据 0 1：数据 1

位域	名称	描述
10	OE_DE	DE 引脚驱动状态 0：输入 1：输出
9	DI_DE	DE 引脚状态 0：数据 0 1：数据 1
8	DO_DE	DE 引脚输出状态 0：数据 0 1：数据 1
2	OE_TX	TX 引脚驱动状态 0：输入 1：输出
1	DI_TX	TX 引脚状态 0：数据 0 1：数据 1
0	DO_TX	TX 引脚输出状态 0：数据 0 1：数据 1

CTRL[XCVR][PIN] 位域

41.4.14 CTRL[XCVR][STATE] (0x3C + 0x400 * n)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD					RECV_STATE			RSVD					SEND_STATE			RSVD															
N/A					RO			N/A					RO			N/A															
x	x	x	x	x	0	0	0	x	x	x	x	x	0	0	0	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x

CTRL[XCVR][STATE] [31:0]

位域	名称	描述
26-24	RECV_STATE	异步接收状态机
18-16	SEND_STATE	异步发送状态机

CTRL[XCVR][STATE] 位域

41.4.15 CTRL[TRG][IN_CFG] (0x40 + 0x400 * n)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD								PRD_EN	RSVD				SYNC_SEL			IN1_EN	RSVD				IN1_SEL			IN0_EN	RSVD				IN0_SEL		
N/A								RW	N/A				RW			RW	N/A				RW			RW	N/A				RW		
x	x	x	x	x	x	x	x	0	x	x	x	x	0	0	0	0	x	x	x	x	0	0	0	0	x	x	x	x	0	0	0

CTRL[TRG][IN_CFG] [31:0]

位域	名称	描述
23	PRD_EN	允许周期性触发（触发 2） 0：禁止周期性触发 1：允许周期性触发
18-16	SYNC_SEL	周期性触发（触发 2）同步选择 0：触发输入 0 1：触发输入 1 ... 7：触发输入 7
15	IN1_EN	允许触发 1 0：禁止触发 1 1：允许触发 1
10-8	IN1_SEL	触发 1 选择 0：触发输入 0 1：触发输入 1 ... 7：触发输入 7
7	IN0_EN	允许触发 0 0：禁止触发 1 1：允许触发 1
2-0	IN0_SEL	触发 0 选择 0：触发输入 0 1：触发输入 1 ... 7：触发输入 7

CTRL[TRG][IN_CFG] 位域

41.4.16 CTRL[TRG][SW] (0x44 + 0x400 * n)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																															SOFT
N/A																															WC

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

CTRL[TRG][SW] [31:0]

位域	名称	描述
0	SOFT	软件触发（触发 3），此位自动清零 0：无触发 1：软件触发

CTRL[TRG][SW] 位域

41.4.17 CTRL[TRG][PRD_CFG] (0x48 + 0x400 * n)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD															ARMING	RSVD															SYNC
N/A															RW	N/A															RW
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0

CTRL[TRG][PRD_CFG] [31:0]

位域	名称	描述
16	ARMING	等待触发源，触发后，周期性触发 0：直接触发 1：等待触发源再生成触发
0	SYNC	同步 0：不与触发源同步 1：触发源到来时，产生触发，随后周期性触发

CTRL[TRG][PRD_CFG] 位域

41.4.18 CTRL[TRG][PRD] (0x4C + 0x400 * n)

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

CTRL[TRG][PRD] [31:0]

位域	名称	描述
31-0	PERIOD	触发周期

位域	名称	描述

CTRL[TRG][PRD] 位域

41.4.19 CTRL[TRG][OUT_CFG] (0x50 + 0x400 * n)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OUT3_EN	RSVD				OUT3_SEL			OUT2_EN	RSVD				OUT2_SEL			OUT1_EN	RSVD				OUT1_SEL			OUT0_EN	RSVD				OUT0_SEL		
RW	N/A				RW			RW	N/A				RW			RW	N/A				RW			RW	N/A				RW		
0	x	x	x	x	0	0	0	0	x	x	x	x	0	0	0	0	x	x	x	x	0	0	0	0	x	x	x	x	0	0	0

CTRL[TRG][OUT_CFG] [31:0]

位域	名称	描述
31	OUT3_EN	允许触发 3 0：禁止触发 3 1：允许触发 3
26-24	OUT3_SEL	触发 3 选择 0：触发输出 0 1：触发输出 1 ... 7：触发输出 7
23	OUT2_EN	允许触发 2 0：禁止触发 2 1：允许触发 2
18-16	OUT2_SEL	触发 2 选择 0：触发输出 0 1：触发输出 1 ... 7：触发输出 7
15	OUT1_EN	允许触发 1 0：禁止触发 1 1：允许触发 1
10-8	OUT1_SEL	触发 1 选择 0：触发输出 0 1：触发输出 1 ... 7：触发输出 7
7	OUT0_EN	允许触发 0 0：禁止触发 1 1：允许触发 1

位域	名称	描述
2-0	OUT0_SEL	触发 0 选择 0：触发输出 0 1：触发输出 1 ... 7：触发输出 7

CTRL[TRG][OUT_CFG] 位域

41.4.20 CTRL[TRG][PRD_STS] (0x60 + 0x400 * n)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD											TRIGERED	RSVD			ARMED	RSVD															
N/A											RO	N/A			RO	N/A															
x	x	x	x	x	x	x	x	x	x	x	0	x	x	x	0	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x

CTRL[TRG][PRD_STS] [31:0]

位域	名称	描述
20	TRIGERED	已触发 0：未被触发 1：已被触发
16	ARMED	等待触发 0：不处于等待 1：等待触发

CTRL[TRG][PRD_STS] 位域

41.4.21 CTRL[TRG][PRD_CNT] (0x64 + 0x400 * n)

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

CTRL[TRG][PRD_CNT] [31:0]

位域	名称	描述
31-0	PERIOD_CNT	周期触发计数器

CTRL[TRG][PRD_CNT] 位域

41.4.22 CTRL[TRG_TABLE][CMD] (0x80 + 0x400 * n + 0x4 * x)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CMD_TRIGGER0																															
RW																															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

CTRL[TRG_TABLE][CMD] [31:0]

位域	名称	描述
31-0	CMD_TRIGGER0	触发命令

CTRL[TRG_TABLE][CMD] 位域

41.4.23 CTRL[TRG_TABLE][TIME] (0xA0 + 0x400 * n + 0x4 * x)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TRIGGER0_TIME																															
RO																															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

CTRL[TRG_TABLE][TIME] [31:0]

位域	名称	描述
31-0	TRIGGER0_TIME	触发时间

CTRL[TRG_TABLE][TIME] 位域

41.4.24 CTRL[CMD][MODE] (0xC0 + 0x400 * n)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD			RSVD								WLEN					RSVD		RSVD		WORDER	BORDER	SIGNED	REWIND	RSVD							MODE
N/A			N/A								RW					N/A		N/A		RW	RW	RW	WC	N/A							RW
0	0	0	0	0	0	0	0	0	0	0	0	0	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0				

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

CTRL[CMD][MODE] [31:0]

位域	名称	描述
20-16	WLEN	字长度 0：1 位 1：2 位 ... 31：32 位
11	WORDER	字序 0：和位序相同 1：和位序相反
10	BORDER	位序 0：从低到高 1：从高到低
9	SIGNED	有符号数，有符号时，最高位会自动扩展 0：无符号 1：有符号
8	REWIND	复位位指针，该位会被自动清除。
1-0	MODE	模式 0：数据模式 1：检查模式 2：CRC 模式

CTRL[CMD][MODE] 位域

41.4.25 CTRL[CMD][IDX] (0xC4 + 0x400 * n)

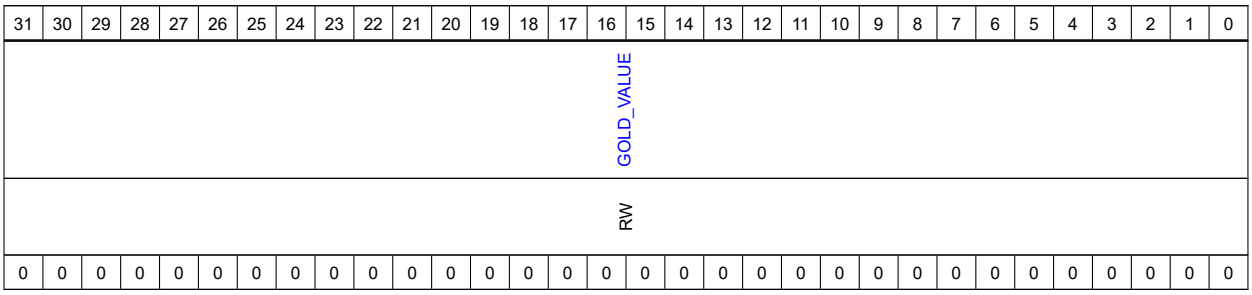
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD			LAST_BIT					RSVD			FIRST_BIT					RSVD			MAX_BIT					RSVD			MIN_BIT				
N/A			RW					N/A			RW					N/A			RW					N/A			RW				
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

CTRL[CMD][IDX] [31:0]

位域	名称	描述
28-24	LAST_BIT	末个填充位
20-16	FIRST_BIT	首个填充位
12-8	MAX_BIT	最高位
4-0	MIN_BIT	最低位

CTRL[CMD][IDX] 位域

41.4.26 CTRL[CMD][GOLD] (0xC8 + 0x400 * n)

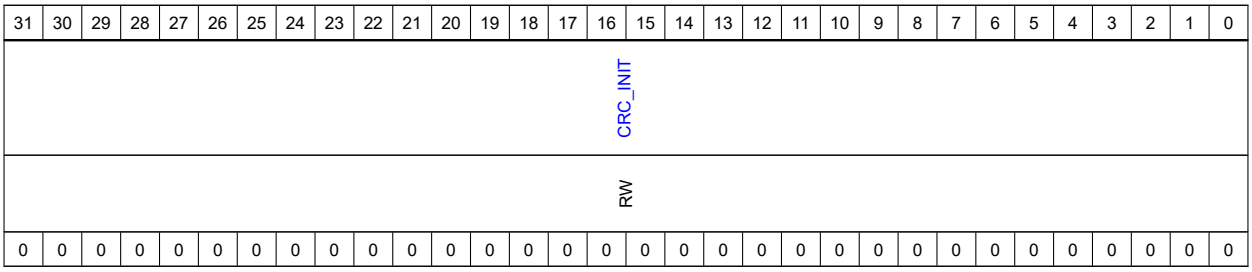


CTRL[CMD][GOLD] [31:0]

位域	名称	描述
31-0	GOLD_VALUE	期望值

CTRL[CMD][GOLD] 位域

41.4.27 CTRL[CMD][CRCINIT] (0xCC + 0x400 * n)

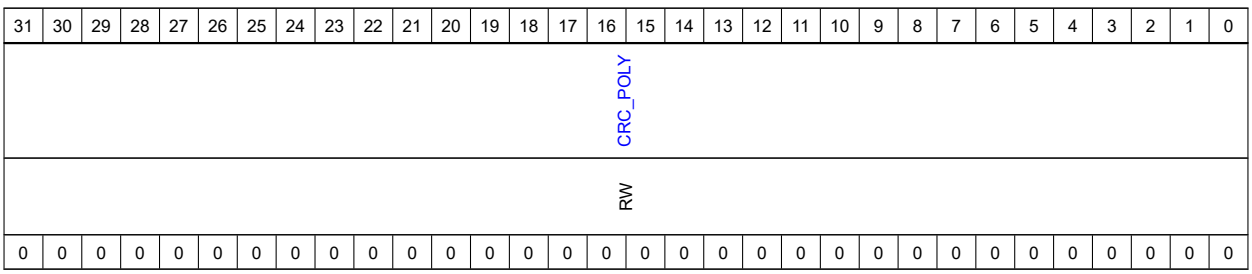


CTRL[CMD][CRCINIT] [31:0]

位域	名称	描述
31-0	CRC_INIT	CRC 初始值

CTRL[CMD][CRCINIT] 位域

41.4.28 CTRL[CMD][CRCPOLY] (0xD0 + 0x400 * n)



CTRL[CMD][CRCPOLY] [31:0]

位域	名称	描述
31-0	CRC_POLY	CRC 多项式

CTRL[CMD][CRCPOLY] 位域

41.4.29 CTRL[CMD][CMD] (0xE0 + 0x400 * n)

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

CTRL[CMD][CMD] [31:0]

位域	名称	描述
31-0	DATA	数据

CTRL[CMD][CMD] 位域

41.4.30 CTRL[CMD][SET] (0xE4 + 0x400 * n)

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

CTRL[CMD][SET] [31:0]

位域	名称	描述
31-0	DATA_SET	数据置位

CTRL[CMD][SET] 位域

41.4.31 CTRL[CMD][CLR] (0xE8 + 0x400 * n)

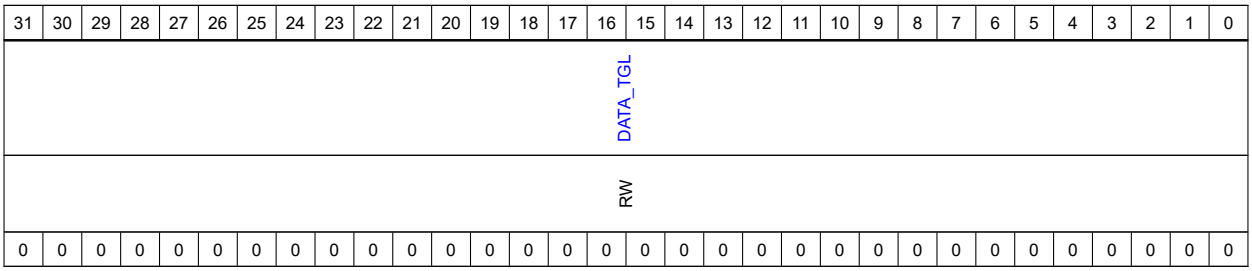
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA_CLR																															
RW																															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

CTRL[CMD][CLR] [31:0]

位域	名称	描述
31-0	DATA_CLR	数据清零

CTRL[CMD][CLR] 位域

41.4.32 CTRL[CMD][INV] (0xEC + 0x400 * n)

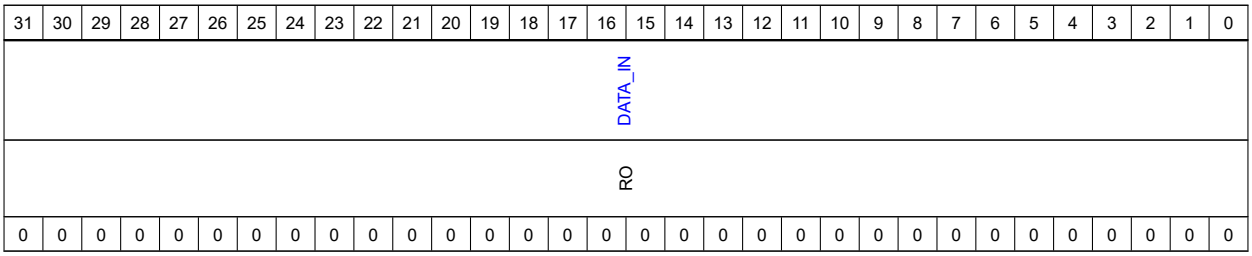


CTRL[CMD][INV] [31:0]

位域	名称	描述
31-0	DATA_TGL	数据翻转

CTRL[CMD][INV] 位域

41.4.33 CTRL[CMD][IN] (0xF0 + 0x400 * n)

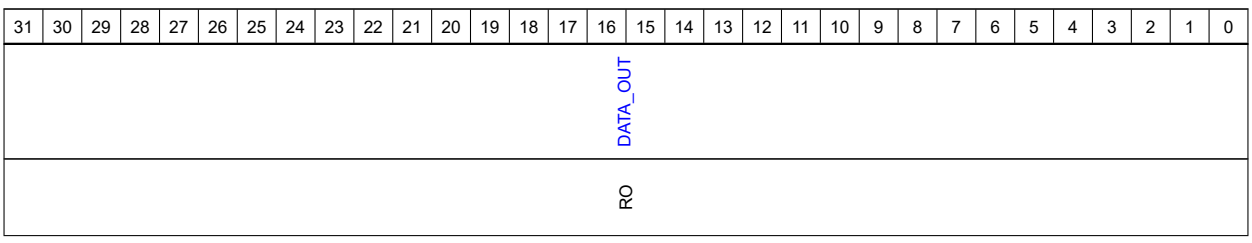


CTRL[CMD][IN] [31:0]

位域	名称	描述
31-0	DATA_IN	输入命令

CTRL[CMD][IN] 位域

41.4.34 CTRL[CMD][OUT] (0xF4 + 0x400 * n)



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

CTRL[CMD][OUT] [31:0]

位域	名称	描述
31-0	DATA_OUT	输出命令

CTRL[CMD][OUT] 位域

41.4.35 CTRL[CMD][STS] (0xF8 + 0x400 * n)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD			CRC_IDX					RSVD			WORD_IDX					RSVD			WORD_CNT					RSVD				BIT_IDX			
N/A			RO					N/A			RO					N/A			RO					N/A				RO			
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

CTRL[CMD][STS] [31:0]

位域	名称	描述
28-24	CRC_IDX	CRC 指针
20-16	WORD_IDX	字指针
12-8	WORD_CNT	字计数
4-0	BIT_IDX	位指针

CTRL[CMD][STS] 位域

41.4.36 CTRL[CMD_TABLE][MIN] (0x100 + 0x400 * n + 0x20 * m)

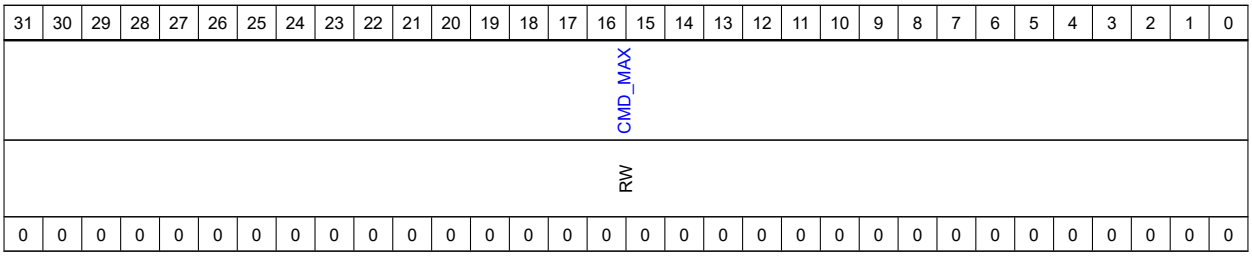
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CMD_MIN																															
RW																															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

CTRL[CMD_TABLE][MIN] [31:0]

位域	名称	描述
31-0	CMD_MIN	命令最小值

CTRL[CMD_TABLE][MIN] 位域

41.4.37 CTRL[CMD_TABLE][MAX] (0x104 + 0x400 * n + 0x20 * m)

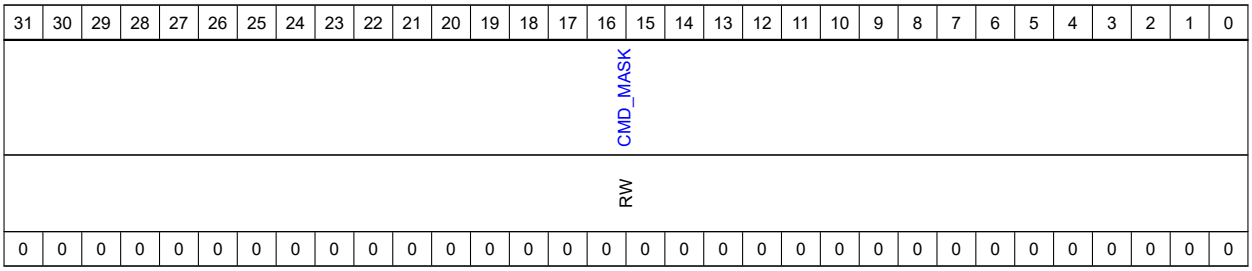


CTRL[CMD_TABLE][MAX] [31:0]

位域	名称	描述
31-0	CMD_MAX	命令最大值

CTRL[CMD_TABLE][MAX] 位域

41.4.38 CTRL[CMD_TABLE][MSK] (0x108 + 0x400 * n + 0x20 * m)

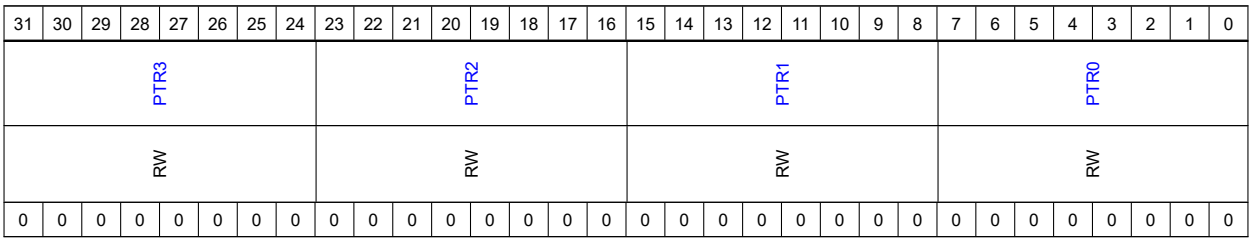


CTRL[CMD_TABLE][MSK] [31:0]

位域	名称	描述
31-0	CMD_MASK	命令比较位

CTRL[CMD_TABLE][MSK] 位域

41.4.39 CTRL[CMD_TABLE][PTA] (0x110 + 0x400 * n + 0x20 * m)



CTRL[CMD_TABLE][PTA] [31:0]

位域	名称	描述
31-24	PTR3	指针 3
23-16	PTR2	指针 2
15-8	PTR1	指针 1
7-0	PTR0	指针 0

位域	名称	描述

CTRL[CMD_TABLE][PTA] 位域

41.4.40 CTRL[CMD_TABLE][PTB] (0x114 + 0x400 * n + 0x20 * m)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PTR7								PTR6								PTR5								PTR4							
RW								RW								RW								RW							
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

CTRL[CMD_TABLE][PTB] [31:0]

位域	名称	描述
31-24	PTR7	指针 7
23-16	PTR6	指针 6
15-8	PTR5	指针 5
7-0	PTR4	指针 4

CTRL[CMD_TABLE][PTB] 位域

41.4.41 CTRL[LATCH][TRAN] (0x200 + 0x400 * n + 0x20 * m + 0x4 * x)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
POINTER								RSVD						CFG_TM		RSVD		CFG_TXD		CFG_CLK		CFG_PTR		RSVD			OV_TM	RSVD	OV_TXD	OV_CLK	OV_PTR
RW								N/A						RW		N/A		RW		RW		RW		N/A			RW	N/A	RW	RW	RW
0	0	0	0	0	0	0	0	0	0	0	0	x	x	0	0	0	0	0	0	0	0	0	0	x	0	0	0				

CTRL[LATCH][TRAN] [31:0]

位域	名称	描述
31-24	POINTER	指针
17-16	CFG_TM	收发超时 0：状态为 1 1：状态为 0 2：上升 3：下降

位域	名称	描述
13-12	CFG_TXD	发送数据状态 0：状态为 1 1：状态为 0 2：上升 3：下降
11-10	CFG_CLK	时钟线状态（仅主机模式时可用） 0：状态为 1 1：状态为 0 2：上升 3：下降
9-8	CFG_PTR	指针 0：匹配 1：不匹配 2：开始 3：结束
4	OV_TM	0：检查超时 1：不检查超时
2	OV_TXD	0：检查发送数据线 1：不检查发送数据线
1	OV_CLK	0：检查时钟线 1：不检查时钟线
0	OV_PTR	0：检查指针 1：不检查指针

CTRL[LATCH][TRAN] 位域

41.4.42 CTRL[LATCH][CFG] (0x210 + 0x400 * n + 0x20 * m)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN	RSVD				SELECT			RSVD								DELAY																
RW	N/A				RW			N/A								RW															
0	x	x	x	x	0	0	0	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

CTRL[LATCH][CFG] [31:0]

位域	名称	描述
31	EN	允许锁存 0：禁止 1：允许

位域	名称	描述
26-24	SELECT	选择输出 0：状态 0-状态 1 1：状态 1-状态 2 2：状态 2-状态 3 3：状态 3-状态 0
15-0	DELAY	状态跳转延时，以系统时钟计数

CTRL[LATCH][CFG] 位域

41.4.43 CTRL[LATCH][TIME] (0x218 + 0x400 * n + 0x20 * m)

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

CTRL[LATCH][TIME] [31:0]

位域	名称	描述
31-0	LAT_TIME	锁存时间

CTRL[LATCH][TIME] 位域

41.4.44 CTRL[LATCH][STS] (0x21C + 0x400 * n + 0x20 * m)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD					STATE			RSVD								LAT_CNT															
N/A					RO			N/A								RO															
x	x	x	x	x	0	0	0	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

CTRL[LATCH][STS] [31:0]

位域	名称	描述
26-24	STATE	内部状态
15-0	LAT_CNT	锁存计数器

CTRL[LATCH][STS] 位域

41.4.45 CTRL[POS][SMP_EN] (0x280 + 0x400 * n)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ACC_EN	RSVD		ACC_SEL					SPD_EN	RSVD		SPD_SEL					REV_EN	RSVD		REV_SEL					POS_EN	RSVD		POS_SEL				
RW	N/A		RW					RW	N/A		RW					RW	N/A		RW					RW	N/A		RW				
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

CTRL[POS][SMP_EN] [31:0]

位域	名称	描述
31	ACC_EN	包含加速度
28-24	ACC_SEL	加速度寄存器选择
23	SPD_EN	包含速度
20-16	SPD_SEL	速度寄存器选择
15	REV_EN	包含圈数
12-8	REV_SEL	圈数选择
7	POS_EN	包含位置
4-0	POS_SEL	位置选择

CTRL[POS][SMP_EN] 位域

41.4.46 CTRL[POS][SMP_CFG] (0x284 + 0x400 * n)

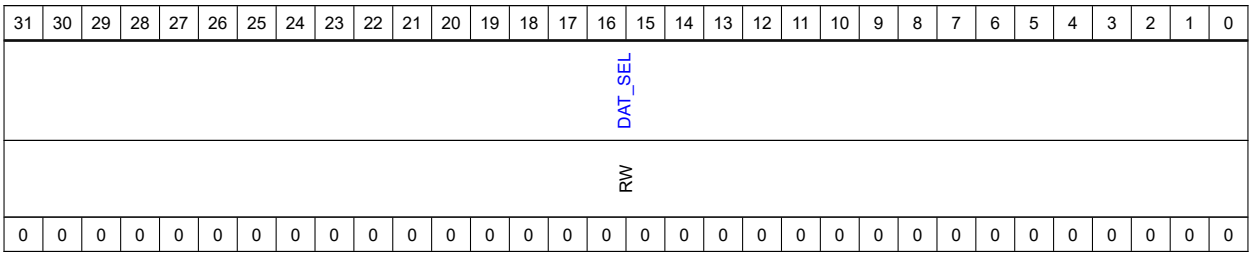
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD							ONCE	RSVD						LAT_SEL		WINDOW																
N/A							RW	N/A						RW		RW																
x	x	x	x	x	x	x	0	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

CTRL[POS][SMP_CFG] [31:0]

位域	名称	描述
24	ONCE	单次采样 0：采样窗口保持有效 1：采样窗口在采样成功后关闭
17-16	LAT_SEL	锁存选择 0：锁存 0 1：锁存 1 2：锁存 2 3：锁存 3
15-0	WINDOW	采样窗口，以时钟周期为单位的采样窗口时间

CTRL[POS][SMP_CFG] 位域

41.4.47 CTRL[POS][SMP_DAT] (0x288 + 0x400 * n)

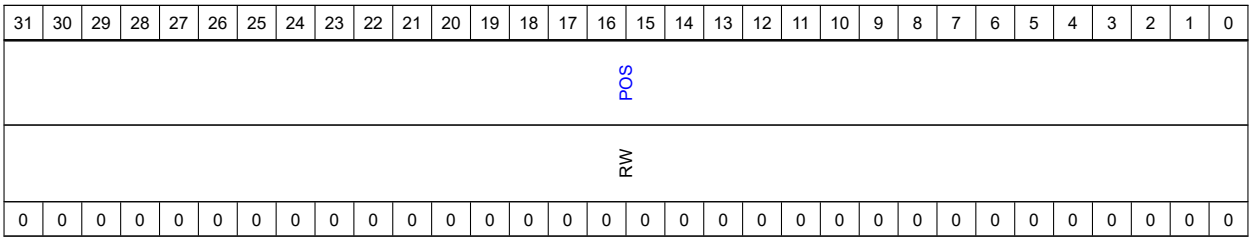


CTRL[POS][SMP_DAT] [31:0]

位域	名称	描述
31-0	DAT_SEL	采样寄存器选择，每一位代表一个数据寄存器

CTRL[POS][SMP_DAT] 位域

41.4.48 CTRL[POS][SMP_POS] (0x290 + 0x400 * n)

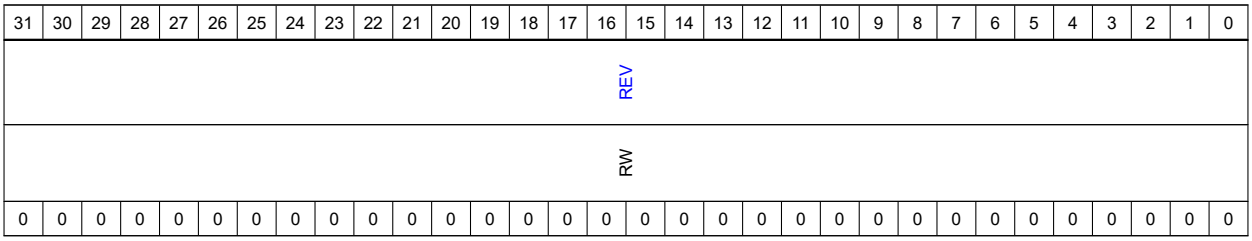


CTRL[POS][SMP_POS] [31:0]

位域	名称	描述
31-0	POS	位置覆盖值

CTRL[POS][SMP_POS] 位域

41.4.49 CTRL[POS][SMP_REV] (0x294 + 0x400 * n)



CTRL[POS][SMP_REV] [31:0]

位域	名称	描述
31-0	REV	圈数覆盖值

CTRL[POS][SMP_REV] 位域

41.4.50 CTRL[POS][SMP_SPD] (0x298 + 0x400 * n)

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

CTRL[POS][SMP_SPD] [31:0]

位域	名称	描述
31-0	SPD	速度覆盖值

CTRL[POS][SMP_SPD] 位域

41.4.51 CTRL[POS][SMP_ACC] (0x29C + 0x400 * n)

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

CTRL[POS][SMP_ACC] [31:0]

位域	名称	描述
31-0	ACC	加速度覆盖值

CTRL[POS][SMP_ACC] 位域

41.4.52 CTRL[POS][UPD_EN] (0x2A0 + 0x400 * n)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ACC_EN	RSVD			ACC_SEL				SPD_EN	RSVD			SPD_SEL				REV_EN	RSVD			REV_SEL				POS_EN	RSVD			POS_SEL			
RW	N/A			RW				RW	N/A			RW				RW	N/A			RW				RW	N/A			RW			
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

CTRL[POS][UPD_EN] [31:0]

位域	名称	描述
31	ACC_EN	包含加速度
28-24	ACC_SEL	加速度寄存器选择
23	SPD_EN	包含速度

位域	名称	描述
20-16	SPD_SEL	速度寄存器选择
15	REV_EN	包含圈数
12-8	REV_SEL	圈数选择
7	POS_EN	包含位置
4-0	POS_SEL	位置选择

CTRL[POS][UPD_EN] 位域

41.4.53 CTRL[POS][UPD_CFG] (0x2A4 + 0x400 * n)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OVRD	RSVD						ONERR	RSVD						LAT_SEL		RSVD																
RW	N/A						RW	N/A						RW		N/A																
0	x	x	x	x	x	x	0	x	x	x	x	x	x	0	0	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x

CTRL[POS][UPD_CFG] [31:0]

位域	名称	描述
31	OVRD	使用覆盖值 0：使用接收到的数值 1：使用内部覆盖值
24	ONERR	出错时更新，传输出错时，仍然把接收到的数值传递给下级 0：出错时不更新 1：出错时仍然更新
17-16	LAT_SEL	锁存选择 0：锁存 0 1：锁存 1 2：锁存 2 3：锁存 3

CTRL[POS][UPD_CFG] 位域

41.4.54 CTRL[POS][UPD_DAT] (0x2A8 + 0x400 * n)

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

CTRL[POS][UPD_DAT] [31:0]

位域	名称	描述
31-0	DAT_SEL	更新寄存器选择，每一位代表一个数据寄存器

CTRL[POS][UPD_DAT] 位域

41.4.55 CTRL[POS][UPD_TIME] (0x2AC + 0x400 * n)

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

CTRL[POS][UPD_TIME] [31:0]

位域	名称	描述
31-0	TIME	时间覆盖值

CTRL[POS][UPD_TIME] 位域

41.4.56 CTRL[POS][UPD_POS] (0x2B0 + 0x400 * n)

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

CTRL[POS][UPD_POS] [31:0]

位域	名称	描述
31-0	POS	位置覆盖值

CTRL[POS][UPD_POS] 位域

41.4.57 CTRL[POS][UPD_REV] (0x2B4 + 0x400 * n)

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

CTRL[POS][UPD_REV] [31:0]

位域	名称	描述
31-0	REV	圈数覆盖值

CTRL[POS][UPD_REV] 位域

41.4.58 CTRL[POS][UPD_SPD] (0x2B8 + 0x400 * n)

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

CTRL[POS][UPD_SPD] [31:0]

位域	名称	描述
31-0	SPD	速度覆盖值

CTRL[POS][UPD_SPD] 位域

41.4.59 CTRL[POS][UPD_ACC] (0x2BC + 0x400 * n)

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

CTRL[POS][UPD_ACC] [31:0]

位域	名称	描述
31-0	ACC	加速度覆盖值

CTRL[POS][UPD_ACC] 位域

41.4.60 CTRL[POS][SMP_VAL] (0x2C0 + 0x400 * n)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ACC	RSVD							SPD	RSVD							REV	RSVD							POS	RSVD						
RO	N/A							RO	N/A							RO	N/A							RO	N/A						
0	x	x	x	x	x	x	x	0	x	x	x	x	x	x	x	0	x	x	x	x	x	x	x	0	x	x	x	x	x	x	x

CTRL[POS][SMP_VAL] [31:0]

位域	名称	描述
31	ACC	包含加速度
23	SPD	包含速度
15	REV	包含圈数
7	POS	包含位置

CTRL[POS][SMP_VAL] 位域

41.4.61 CTRL[POS][SMP_STS] (0x2C4 + 0x400 * n)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD							OCCUR	RSVD								WIN_CNT															
N/A							RO	N/A								RO															
x	x	x	x	x	x	x	0	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

CTRL[POS][SMP_STS] [31:0]

位域	名称	描述
24	OCCUR	已采样 0：未采样到有效数据 1：采样到有效数据
15-0	WIN_CNT	采样窗口计数器

CTRL[POS][SMP_STS] 位域

41.4.62 CTRL[POS][TIME_IN] (0x2CC + 0x400 * n)

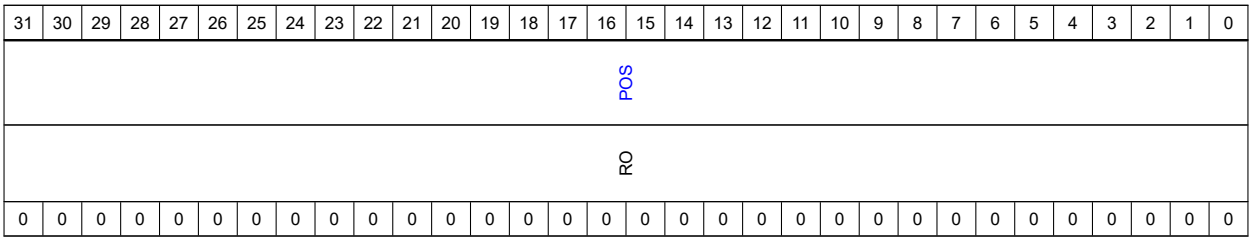
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TIME																															
RO																															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

CTRL[POS][TIME_IN] [31:0]

位域	名称	描述
31-0	TIME	输入时间

CTRL[POS][TIME_IN] 位域

41.4.63 CTRL[POS][POS_IN] (0x2D0 + 0x400 * n)

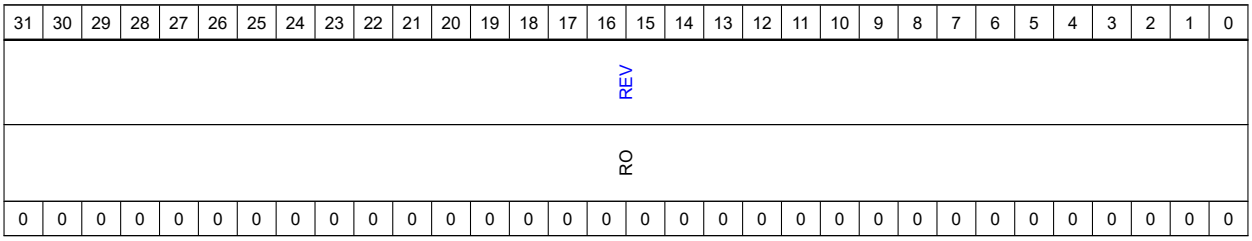


CTRL[POS][POS_IN] [31:0]

位域	名称	描述
31-0	POS	输入位置

CTRL[POS][POS_IN] 位域

41.4.64 CTRL[POS][REV_IN] (0x2D4 + 0x400 * n)

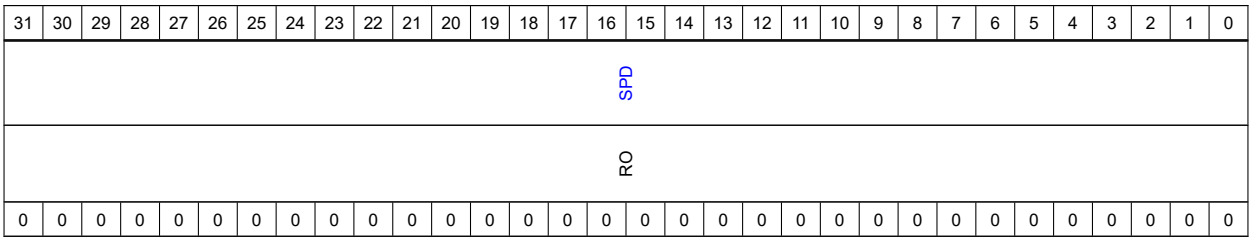


CTRL[POS][REV_IN] [31:0]

位域	名称	描述
31-0	REV	输入圈数

CTRL[POS][REV_IN] 位域

41.4.65 CTRL[POS][SPD_IN] (0x2D8 + 0x400 * n)



CTRL[POS][SPD_IN] [31:0]

位域	名称	描述
31-0	SPD	输入速度

CTRL[POS][SPD_IN] 位域

41.4.66 CTRL[POS][ACC_IN] (0x2DC + 0x400 * n)

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

CTRL[POS][ACC_IN] [31:0]

位域	名称	描述
31-0	ACC	输入加速度

CTRL[POS][ACC_IN] 位域

41.4.67 CTRL[POS][UPD_STS] (0x2E4 + 0x400 * n)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD							UPD_ERR	RSVD																							
N/A							RO	N/A																							
x	x	x	x	x	x	x	0	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x

CTRL[POS][UPD_STS] [31:0]

位域	名称	描述
24	UPD_ERR	更新错误 0：更新数据正常 1：更新数据传输中发生错误

CTRL[POS][UPD_STS] 位域

41.4.68 CTRL[IRQ][INT_EN] (0x300 + 0x400 * n)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TRG_ERR3	TRG_ERR2	TRG_ERR1	TRG_ERR0	TRIGER3	TRIGER2	TRIGER1	TRIGER0	RSVD			SMP_ERR	LATCH3	LATCH2	LATCH1	LATCH0	RSVD		TIMEOUT	TRX_ERR	INSTR1_END	INSTR0_END	PTR1_END	PTR0_END	INSTR1_ST	INSTR0_ST	PTR1_ST	PTR0_ST	RSVD	WDOG	EXECPT	STALL
RW	RW	RW	RW	RW	RW	RW	RW	N/A			RW	RW	RW	RW	RW	N/A		RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	N/A	RW	RW	RW
0	0	0	0	0	0	0	0	x	x	x	0	0	0	0	0	x	x	0	0	0	0	0	0	0	0	0	0	x	0	0	0

CTRL[IRQ][INT_EN] [31:0]

位域	名称	描述
31	TRG_ERR3	触发错误 3
30	TRG_ERR2	触发错误 2

位域	名称	描述
29	TRG_ERR1	触发错误 1
28	TRG_ERR0	触发错误 0
27	TRIGER3	触发 3
26	TRIGER2	触发 2
25	TRIGER1	触发 1
24	TRIGER0	触发 0
20	SMP_ERR	采样错误
19	LATCH3	锁存 3
18	LATCH2	锁存 2
17	LATCH1	锁存 1
16	LATCH0	锁存 0
13	TIMEOUT	超时
12	TRX_ERR	传输错
11	INSTR1_END	指令 1 结束
10	INSTR0_END	指令 0 结束
9	PTR1_END	指针 1 结束
8	PTR0_END	指针 0 结束
7	INSTR1_ST	指令 1 开始
6	INSTR0_ST	指令 0 开始
5	PTR1_ST	指针 1 开始
4	PTR0_ST	指针 0 开始
2	WDOG	看门狗
1	EXECPT	异常
0	STALL	停止

CTRL[IRQ][INT_EN] 位域

41.4.69 CTRL[IRQ][INT_FLAG] (0x304 + 0x400 * n)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TRG_ERR3	TRG_ERR2	TRG_ERR1	TRG_ERR0	TRIGER3	TRIGER2	TRIGER1	TRIGER0	RSVD			SMP_ERR	LATCH3	LATCH2	LATCH1	LATCH0	RSVD		TIMEOUT	TRX_ERR	INSTR1_END	INSTR0_END	PTR1_END	PTR0_END	INSTR1_ST	INSTR0_ST	PTR1_ST	PTR0_ST	RSVD	WDOG	EXECPT	STALL
W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	N/A			W1C	W1C	W1C	W1C	W1C	N/A		W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	N/A	W1C	W1C	W1C
0	0	0	0	0	0	0	0	x	x	x	0	0	0	0	0	x	x	0	0	0	0	0	0	0	0	0	0	x	0	0	0

CTRL[IRQ][INT_FLAG] [31:0]

位域	名称	描述
31	TRG_ERR3	触发错误 3
30	TRG_ERR2	触发错误 2

位域	名称	描述
29	TRG_ERR1	触发错误 1
28	TRG_ERR0	触发错误 0
27	TRIGER3	触发 3
26	TRIGER2	触发 2
25	TRIGER1	触发 1
24	TRIGER0	触发 0
20	SMP_ERR	采样错误
19	LATCH3	锁存 3
18	LATCH2	锁存 2
17	LATCH1	锁存 1
16	LATCH0	锁存 0
13	TIMEOUT	超时
12	TRX_ERR	传输错
11	INSTR1_END	指令 1 结束
10	INSTR0_END	指令 0 结束
9	PTR1_END	指针 1 结束
8	PTR0_END	指针 0 结束
7	INSTR1_ST	指令 1 开始
6	INSTR0_ST	指令 0 开始
5	PTR1_ST	指针 1 开始
4	PTR0_ST	指针 0 开始
2	WDOG	看门狗
1	EXECPT	异常
0	STALL	停止

CTRL[IRQ][INT_FLAG] 位域

41.4.70 CTRL[IRQ][INT_STS] (0x308 + 0x400 * n)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TRG_ERR3	TRG_ERR2	TRG_ERR1	TRG_ERR0	TRIGER3	TRIGER2	TRIGER1	TRIGER0	RSVD			SMP_ERR	LATCH3	LATCH2	LATCH1	LATCH0	RSVD		TIMEOUT	TRX_ERR	INSTR1_END	INSTR0_END	PTR1_END	PTR0_END	INSTR1_ST	INSTR0_ST	PTR1_ST	PTR0_ST	RSVD	WDOG	EXECPT	STALL
RO	RO	RO	RO	RO	RO	RO	RO	N/A			RO	RO	RO	RO	RO	N/A		RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	N/A	RO	RO	RO
0	0	0	0	0	0	0	0	x	x	x	0	0	0	0	0	x	x	0	0	0	0	0	0	0	0	0	0	x	0	0	0

CTRL[IRQ][INT_STS] [31:0]

位域	名称	描述
31	TRG_ERR3	触发错误 3
30	TRG_ERR2	触发错误 2

位域	名称	描述
29	TRG_ERR1	触发错误 1
28	TRG_ERR0	触发错误 0
27	TRIGER3	触发 3
26	TRIGER2	触发 2
25	TRIGER1	触发 1
24	TRIGER0	触发 0
20	SMP_ERR	采样错误
19	LATCH3	锁存 3
18	LATCH2	锁存 2
17	LATCH1	锁存 1
16	LATCH0	锁存 0
13	TIMEOUT	超时
12	TRX_ERR	传输错
11	INSTR1_END	指令 1 结束
10	INSTR0_END	指令 0 结束
9	PTR1_END	指针 1 结束
8	PTR0_END	指针 0 结束
7	INSTR1_ST	指令 1 开始
6	INSTR0_ST	指令 0 开始
5	PTR1_ST	指针 1 开始
4	PTR0_ST	指针 0 开始
2	WDOG	看门狗
1	EXECPT	异常
0	STALL	停止

CTRL[IRQ][INT_STS] 位域

41.4.71 CTRL[IRQ][POINTER0] (0x310 + 0x400 * n)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																								POINTER							
N/A																								RW							
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0

CTRL[IRQ][POINTER0] [31:0]

位域	名称	描述
7-0	POINTER	匹配指针 0

CTRL[IRQ][POINTER0] 位域

41.4.72 CTRL[IRQ][POINTER1] (0x314 + 0x400 * n)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																								POINTER							
N/A																								RW							
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0

CTRL[IRQ][POINTER1] [31:0]

位域	名称	描述
7-0	POINTER	匹配指针 1

CTRL[IRQ][POINTER1] 位域

41.4.73 CTRL[IRQ][INSTR0] (0x318 + 0x400 * n)

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

CTRL[IRQ][INSTR0] [31:0]

位域	名称	描述
31-0	INSTR	匹配指令 0

CTRL[IRQ][INSTR0] 位域

41.4.74 CTRL[IRQ][INSTR1] (0x31C + 0x400 * n)

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

CTRL[IRQ][INSTR1] [31:0]

位域	名称	描述
31-0	INSTR	匹配指令 1

CTRL[IRQ][INSTR1] 位域

41.4.75 INSTR (0x3400 + 0x4 * n)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD			OP			CK		RSVD			CRC					RSVD			DAT					RSVD			OPR				
N/A			RW			RW		N/A			RW					N/A			RW					N/A			RW				
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

INSTR [31:0]

位域	名称	描述
28-26	OP	操作 0：停顿 1：跳转 2：时限内发送 3：无时限发送 4：时限内等待 5：无时限等待 6：时限内接收 7：无时限接收
25-24	CK	时钟 0：时钟线为低 1：时钟线上升-下降 2：时钟线下降-上升 3：时钟线为高
20-16	CRC	CRC 寄存器 0：不计算 CRC 1：不可用 2：数据寄存器 2 3：数据寄存器 3 ... 29：数据寄存器 29 30：常数 0，发送时为 0，接收时等待接收到 0 31：常数 1，发送时为 1，接收时等待接收到 1
12-8	DAT	DATA 寄存器 0：忽略数据 1：命令寄存器 2：数据寄存器 2 3：数据寄存器 3 ... 29：数据寄存器 29 30：常数 0，发送时为 0，接收时等待接收到 0 31：常数 1，发送时为 1，接收时等待接收到 1

位域	名称	描述
4-0	OPR	[1] 当 OP 为 0 时，此区域为停顿时间，以波特率为时间单位，0 表示无限长。 当 OP 为 1 时，此区域为指令表的指针： OPR[4]=1 时，OPR[3:0] 的值为 CMD_TABLE 的指针； OPR[4]=0 时，OPR[3:0]=0 为 INIT_POINTER； OPR[4]=0 时，OPR[3:0]=1 为 WDG_POINTER。 当 OP 为 2~7 时，此区域为数据长度： 0：1 位 1：2 位 ... 31：32 位

INSTR 位域

41.4.76 DAT[MODE] (0x3800 + 0x40 * n)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD			CRC_LEN					RSVD			WLEN					RSVD		CRC_SHIFT	CRC_INV	WORDER	BORDER	SIGNED	REWIND	RSVD						MODE	
N/A			RW					N/A			RW					N/A		RW	RW	RW	RW	RW	RW	N/A						RW	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

DAT[MODE] [31:0]

位域	名称	描述
28-24	CRC_LEN	CRC 长度 0：1 位 1：2 位 ... 31：32 位
20-16	WLEN	字长度 0：1 位 1：2 位 ... 31：32 位
13	CRC_SHIFT	移位模式, 此模式用于校验重复码 0：CRC 模式 1：重复码模式
12	CRC_INV	CRC 反相 0：使用 CRC 1：使用 CRC 的反码

位域	名称	描述
11	WORDER	字序 0：和位序相同 1：和位序相反
10	BORDER	位序 0：从低到高 1：从高到低
9	SIGNED	有符号数，有符号时，最高位会自动扩展 0：无符号数 1：符号数
8	REWIND	复位位指针，该位会被自动清除。
1-0	MODE	模式 0：数据模式 1：检查模式 2：CRC 模式

DAT[MODE] 位域

41.4.77 DAT[IDX] (0x3804 + 0x40 * n)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD			LAST_BIT					RSVD			FIRST_BIT					RSVD			MAX_BIT					RSVD				MIN_BIT				
N/A			RW					N/A			RW					N/A			RW					N/A				RW				
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

DAT[IDX] [31:0]

位域	名称	描述
28-24	LAST_BIT	末个填充位
20-16	FIRST_BIT	首个填充位
12-8	MAX_BIT	最高位
4-0	MIN_BIT	最低位

DAT[IDX] 位域

41.4.78 DAT[GOLD] (0x3808 + 0x40 * n)

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

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

DAT[GOLD] [31:0]

位域	名称	描述
31-0	GOLD_VALUE	期望值

DAT[GOLD] 位域

41.4.79 DAT[CRCINIT] (0x380C + 0x40 * n)

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

DAT[CRCINIT] [31:0]

位域	名称	描述
31-0	CRC_INIT	CRC 初始值

DAT[CRCINIT] 位域

41.4.80 DAT[CRCPOLY] (0x3810 + 0x40 * n)

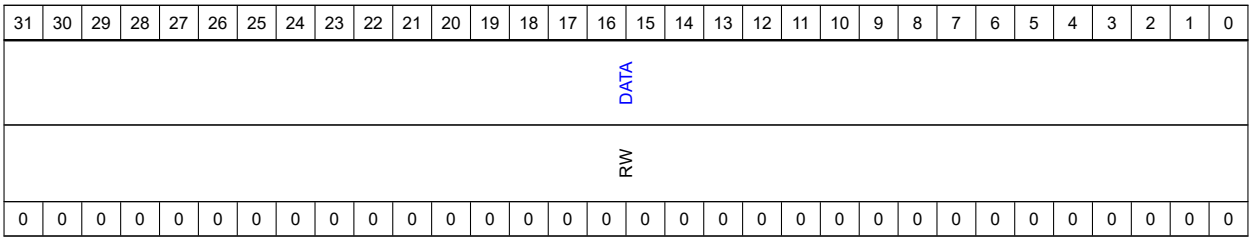
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CRC_POLY																															
RW																															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

DAT[CRCPOLY] [31:0]

位域	名称	描述
31-0	CRC_POLY	CRC 多项式

DAT[CRCPOLY] 位域

41.4.81 DAT[DATA] (0x3820 + 0x40 * n)

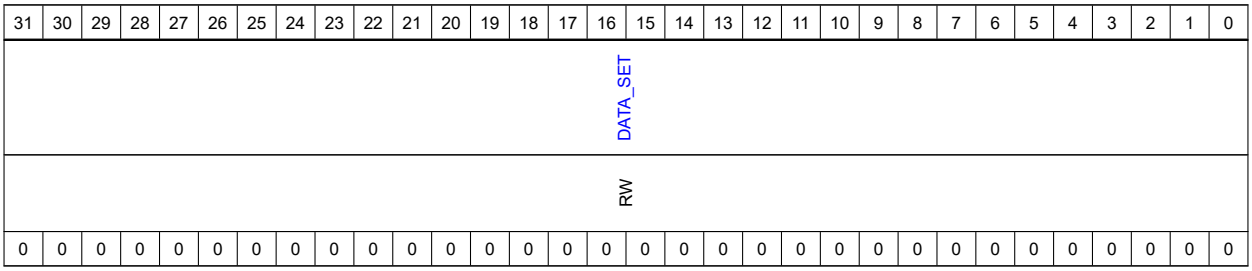


DAT[DATA] [31:0]

位域	名称	描述
31-0	DATA	数据

DAT[DATA] 位域

41.4.82 DAT[SET] (0x3824 + 0x40 * n)

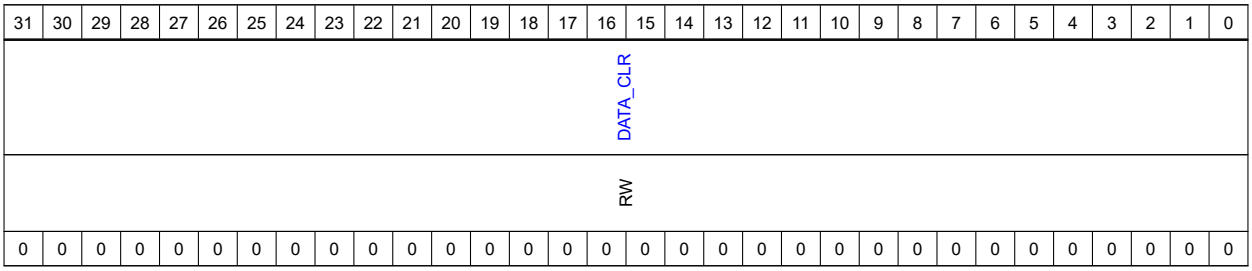


DAT[SET] [31:0]

位域	名称	描述
31-0	DATA_SET	数据置位

DAT[SET] 位域

41.4.83 DAT[CLR] (0x3828 + 0x40 * n)



DAT[CLR] [31:0]

位域	名称	描述
31-0	DATA_CLR	数据清零

DAT[CLR] 位域

41.4.84 DAT[INV] (0x382C + 0x40 * n)

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

DAT[INV] [31:0]

位域	名称	描述
31-0	DATA_INV	数据翻转

DAT[INV] 位域

41.4.85 DAT[IN] (0x3830 + 0x40 * n)

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

DAT[IN] [31:0]

位域	名称	描述
31-0	DATA_IN	输入数据

DAT[IN] 位域

41.4.86 DAT[OUT] (0x3834 + 0x40 * n)

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

DAT[OUT] [31:0]

位域	名称	描述
31-0	DATA_OUT	输出数据

DAT[OUT] 位域

41.4.87 DAT[STS] (0x3838 + 0x40 * n)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD			CRC_IDX					RSVD			WORD_IDX					RSVD			WORD_CNT					RSVD			BIT_IDX				
N/A			RO					N/A			RO					N/A			RO					N/A			RO				
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

DAT[STS] [31:0]

位域	名称	描述
28-24	CRC_IDX	CRC 指针
20-16	WORD_IDX	字指针
12-8	WORD_CNT	字计数
4-0	BIT_IDX	位指针

DAT[STS] 位域

42 定时器概述

本章节介绍了本产品的定时器。本产品上的定时器包括通用定时器，看门狗定时器。

42.1 通用定时器 GPTMR, PTMR

本产品支持 5 个通用定时器，每个通用定时器支持 4 个通道，每个通道支持 32 位计数器，重载寄存器和一个输入捕获/输出比较通道，支持通用计时，输入捕获，输出比较，PWM 生成，以及产生中断和 DMA 请求。

- 4 个定时器位于系统电源域称为通用定时器 GPTMR0~3
- 1 个定时器位于电源管理域，称为电源管理域定时器 PTMR。

本产品上，GPTMR0~3 的 4 个输入捕获/输出比较通道中，除了连接到 IO 上之外，通道 2 和 3 连接到电机控制单元的各个互联管理器上。GPTMR0~3 的计数器同步输入 SYNCI，来自于互联管理器。

GPTMR0~3 的内部信号互联细节，请查阅节 32.8。

本产品上，PTMR 的 4 个输入捕获/输出比较通道连接到电源管理域 IO(GPIO 端口 Y)。PTMR 可以在系统电源域掉电时保持工作。PTMR 中断可以把系统从低功耗模式下唤醒。PTMR 不支持生成 DMA 请求。

42.2 看门狗定时器 WDG, PWDG

本产品支持 3 个看门狗定时器，看门狗定时器可以防止系统因为软件或者硬件的错误而锁死。看门狗定时器支持在软件喂狗超时生成系统复位。

其中 2 个位于系统电源域称为 WDG0~1，WDGx 的时钟为 CLK_TOP_AHB，看门狗计数器外部时钟来自 CLK_32K。

1 个位于电源管理域，称为 PWDG。PWDG 可以在系统电源域掉电时保持工作。PWDG 的时钟为 CLK_24M，看门狗计数器外部时钟为 CLK_32K。PWDG 支持从电源管理域 IO 上输出复位信号。

43 定时器 TMR

本章节介绍定时器 TMR 的主要功能和特性。

43.1 特性总结

本章节介绍定时器 TMR 的主要特性：

- 定时器由 4 通道组成
- 单个通道配置一个 32 位分辨率计数器，支持向上模式
- 各个通道的计数器支持同步
- 单个通道支持可配置的 32 位重载寄存器
- 通道可同时用于输入捕获以及输出比较
- 单个通道支持 2 个输出比较器
- 支持捕获上升沿和/或者下降沿，支持测量 PWM 周期和占空比
- 支持生成 DMA 和中断请求

定时器的框图如图 53。

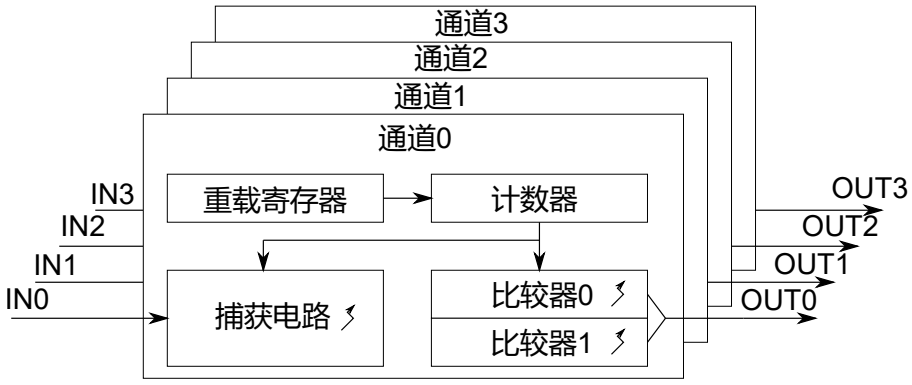


图 53: 定时器框图

43.2 功能描述

本章节描述定时器各个子模块的功能。

43.2.1 定时器时间基准

定时器每个通道的时间基准模块的作用是决定定时器运行需要时间和周期。

它包含以下几部分：

- 32 位计数器
- 32 位的重载寄存器

计时器支持向上计数。

用户可以设置 CHxCR [TEN] 位使能计数器，使能以后，计数器总是从 0 开始计数，当计时器的值到达重载值时，重载标志位置 1. 计数器恢复到 0 开始继续计数。

用户可以通过 CHxCR [CNTRST] 位随时复位计数器。把 GCR[CNTRST] 位置 1 可以使计数器归 0。需要再把 GCR[CNTRST] 位置 0，计数器才会重新开始计数。通过 GCR[CNTRST] 位重置计数器的同时，也会重置所有

的输入捕获值寄存器，同时把输出比较的输出重置到 CHxCR[CMPINIT] 位定义的起始电平。

计数器寄存器为只读寄存器，用户可以通过以下步骤将计数器 CHxCNT 的值更新到期望值：

- 把期望值写入寄存器 CHxCNTUPTVAL
- 把 CHxCR [CNTUPT] 位置 1，将 CHxCNTNEW 的值载入计数器 CHxCNT

用户也可以利用定时器的同步触发输入（SYNCI）来同步计数器开始计数的时机。定时器支持软件同步和硬件同步等同步机制：

- 把 CHxCR [SWSYNCIEN] 位置 1 后，软件同步可以通过写入 GCR[SWSYNCTx] 位触发，每个通道都有独立的软件触发位。用户同时把多个通道的软件触发位置 1，即可实现这些通道的计时器同步开始计时。
- 硬件同步信号来自定时器外部，即 SYNCI，可以通过把 CHxCR [SYNCIREN] 或者 CHxCR [SYNCIFEN] 位置 1，来选择同步发生在 SYNCI 的上升沿还是下降沿。
- 此外，对于通道序号大于 0 的通道，用户可以选择把 CHxCR [SYNCFLW] 位置 1，这样，该通道计数器会随着它之前通道的计数器同步。比如，通道 1 的计数器会与通道 0 的计数器同时同步。

如果通过同步触发输入重置计数器到 0，重载标志位 RLDF 也会置 1。

43.2.2 输出比较

通用定时器的通道支持输出比较功能。

使用通道输出比较功能配置流程如下：

- 通过 CHxCR [CMPINIT] 位配置输出的起始状态，该位置 0 时，表示计数器开始计数时，输出起始状态为逻辑 0，反之，则计数器开始计数时输出起始状态为逻辑 1。此外，当计数器计重载时（CNT == RLD），输出也会重置到起始状态。
- 配置通道的两个比较器，CMP0 和 CMP1。每当计数器达到任一比较器值时，通道输出会翻转。
  如果配置 CMP0 和 CMP1 相等，当计数器达到比较器值时，输出无变化。
- 通过设置 CHxCR [CMPEN] 位为 1，打开输出比较功能
- 通过设置 CHxCR [CEN] 位为 1 使能计数器

用户如果需要输出边沿对齐 PWM，可以通过 CMP0 设置 PWM 的占空比，设置 CMP1 值等于重载寄存器值（CMP1 = RLD）。

用户如果需要输出中心对齐 PWM，可以设置 $0 < CMP0 < CMP1 < RLD$。这样，PWM 周期即为 RLD 值，占空比为（CMP1 -CMP0）。

输出比较支持快速输出逻辑 0 和逻辑 1，即把 CMP0 配置为 32'hFFFFFFFF 时，通道输出始终为逻辑 0。把 CMP1 配置为 32'hFFFFFFFF，且 CMP0 不为全 1 时，通道输出始终为逻辑 1。

注意：CMP0 和 CMP1 的初始值均为 32'hFFFFFFFF，用户使用比较输出功能时需配置两个值。

43.2.3 输入捕获

通用定时器的通道支持输入捕获。

用户可以配置 CHxCR[CAPMODE] 位来选择不同的捕获模式：

- 3'b000，关闭输入捕获
- 3'b001，捕获上升沿，每当输入发生从逻辑 0 变成逻辑 1 时，将当前的计数器值保存到 CAPPOS 寄存器，标志位 CHxCAPF 置 1
- 3'b010，捕获下降沿，每当输入发生从逻辑 1 变成逻辑 0 时，将当前的计数器值保存到 CAPNEG 寄存

器，标志位 CHxCAPF 置 1

- 3'b011，捕获上升和下降沿，每当在输入发生翻转时，将当前的计数器值保存到 CAPPOS 和 CAPNEG 寄存器，标志位 CHxCAPF 置 1
- 3'b100，PWM 测量模式，此时定时器会自动测量 PWM 周期和占空比。定时器会把输入信号两个上升沿之间的计数值差（即 PWM 周期）保存在 CAPPRD 寄存器，把信号上升沿和下降沿之间的计数值差（即 PWM 占空比）保存在 CAPDTY 寄存器。CAPVAL 寄存器仍然保存最近一次信号上升沿时计数器的值。在此模式下，捕获到输入信号上升沿时，标志位 CHxCAPF 置 1

注意，CAPPOS 寄存器、CAPNEG 寄存器、CAPPRD 寄存器和 CAPDTY 寄存器的值，根据不同设置，在捕获到新的边沿时会刷新，之前捕获的值会丢失。

注意，在 PWM 测量模式时，用户应当注意计数器 CNT 的值，在相邻的信号边沿跳转之间发生计数器重载，计数器值归 0 的话，CAPPRD 寄存器和 CAPDTY 寄存器的值会不准确。

43.2.4 中断和 DMA

定时器支持生成以下中断：

- CHxRLDF 标志位置 1，即计数器计数达到重载寄存器值，或者由同步触发输入 SYNCI 引发计数器重载
- CHxCMPxF 标志位置 1，即输出比较功能打开时，计数器计数达到 CMP0 或者 CMP1 值时
- CHxCAPF 标志位置 1 时，即输入捕获功能打开时，在输入通道捕获到指定边沿跳变时

通过设置 CHxIRQEN 寄存器相应位，可以打开或者关闭以上中断请求。

用户可以通过读取 CHxSR 寄存器查询所有的标志位，SR 寄存器相应标志位写 1 可以清除标志位。

定时器支持生成以下 DMA 请求：

- CHxRLDF 标志位置 1，即计数器计数达到重载寄存器值，或者由同步触发输入 SYNCI 引发计数器重载
- CHxCMPxF 标志位置 1，即输出比较功能打开时，计数器计数达到 CMP0 或者 CMP1 值时
- CHxCAPF 标志位置 1 时，即输入捕获功能打开时，在输入通道捕获到指定边沿跳变时

通过设置 CHxCR[DMASEL] 寄存器位，可以选择从以上请求中选择一个作为 DMA 请求输出。

通过设置 CHxCR [DMAEN] 寄存器位，可以打开或者关闭 DMA 请求。

43.2.5 调试模式支持

通用定时器支持调试功能，即在内核进入调试模式（Debug）时，暂停计数器计数。用户把 CHxCR[DBGPAUSE] 位置 1 时，此功能即生效。

43.3 定时器寄存器

43.3.1 寄存器说明

定时器的寄存器列表如下：

GPTMR0 base address: 0xF0000000

GPTMR1 base address: 0xF0004000

GPTMR2 base address: 0xF0008000

GPTMR3 base address: 0xF000C000

PTMR base address: 0xF4120000

地址偏移	名称	描述	复位值
0x0000	CHANNEL[CH0][CR]	控制寄存器	0x00000000
0x0004	CHANNEL[CH0][CMP][CMP0]	比较寄存器 0	0xFFFFFFFF
0x0008	CHANNEL[CH0][CMP][CMP1]	比较寄存器 1	0xFFFFFFFF
0x000C	CHANNEL[CH0][RLD]	重载寄存器	0xFFFFFFFF
0x0010	CHANNEL[CH0][CNTUPTVAL]	计数器更新值寄存器	0x00000000
0x0020	CHANNEL[CH0][CAPPOS]	上升沿捕获寄存器	0x00000000
0x0024	CHANNEL[CH0][CAPNEG]	下降沿捕获寄存器	0x00000000
0x0028	CHANNEL[CH0][CAPPRD]	PWM 周期测量寄存器	0x00000000
0x002C	CHANNEL[CH0][CAPDTY]	PWM 占空比测量寄存器	0x00000000
0x0030	CHANNEL[CH0][CNT]	计数器	0x00000000
0x0040	CHANNEL[CH1][CR]	控制寄存器	0x00000000
0x0044	CHANNEL[CH1][CMP][CMP0]	比较寄存器 0	0xFFFFFFFF
0x0048	CHANNEL[CH1][CMP][CMP1]	比较寄存器 1	0xFFFFFFFF
0x004C	CHANNEL[CH1][RLD]	重载寄存器	0xFFFFFFFF
0x0050	CHANNEL[CH1][CNTUPTVAL]	计数器更新值寄存器	0x00000000
0x0060	CHANNEL[CH1][CAPPOS]	上升沿捕获寄存器	0x00000000
0x0064	CHANNEL[CH1][CAPNEG]	下降沿捕获寄存器	0x00000000
0x0068	CHANNEL[CH1][CAPPRD]	PWM 周期测量寄存器	0x00000000
0x006C	CHANNEL[CH1][CAPDTY]	PWM 占空比测量寄存器	0x00000000
0x0070	CHANNEL[CH1][CNT]	计数器	0x00000000
0x0080	CHANNEL[CH2][CR]	控制寄存器	0x00000000
0x0084	CHANNEL[CH2][CMP][CMP0]	比较寄存器 0	0xFFFFFFFF
0x0088	CHANNEL[CH2][CMP][CMP1]	比较寄存器 1	0xFFFFFFFF
0x008C	CHANNEL[CH2][RLD]	重载寄存器	0xFFFFFFFF
0x0090	CHANNEL[CH2][CNTUPTVAL]	计数器更新值寄存器	0x00000000
0x00A0	CHANNEL[CH2][CAPPOS]	上升沿捕获寄存器	0x00000000
0x00A4	CHANNEL[CH2][CAPNEG]	下降沿捕获寄存器	0x00000000
0x00A8	CHANNEL[CH2][CAPPRD]	PWM 周期测量寄存器	0x00000000
0x00AC	CHANNEL[CH2][CAPDTY]	PWM 占空比测量寄存器	0x00000000
0x00B0	CHANNEL[CH2][CNT]	计数器	0x00000000
0x00C0	CHANNEL[CH3][CR]	控制寄存器	0x00000000

地址偏移	名称	描述	复位值
0x00C4	CHANNEL[CH3][CMP][CMP0]	比较寄存器 0	0xFFFFFFFF
0x00C8	CHANNEL[CH3][CMP][CMP1]	比较寄存器 1	0xFFFFFFFF
0x00CC	CHANNEL[CH3][RLD]	重载寄存器	0xFFFFFFFF
0x00D0	CHANNEL[CH3][CNTUPTVAL]	计数器更新值寄存器	0x00000000
0x00E0	CHANNEL[CH3][CAPPOS]	上升沿捕获寄存器	0x00000000
0x00E4	CHANNEL[CH3][CAPNEG]	下降沿捕获寄存器	0x00000000
0x00E8	CHANNEL[CH3][CAPPRD]	PWM 周期测量寄存器	0x00000000
0x00EC	CHANNEL[CH3][CAPDTY]	PWM 占空比测量寄存器	0x00000000
0x00F0	CHANNEL[CH3][CNT]	计数器	0x00000000
0x0200	SR	状态寄存器	0x00000000
0x0204	IRQEN	中断使能寄存器	0x00000000
0x0208	GCR	全局控制寄存器	0x00000000

表 199: GPTMR 寄存器列表

定时器的寄存器详细说明如下：

43.3.2 CHANNEL[CR] (0x0 + 0x40 * n)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CNTUPT	RSVD													RSVD			CNTRST	SYNCFLW	SYNCIFEN	SYNCIREN	CEN	CMPINIT	CMPEN	DMASEL		DMAEN	SWSYNCIEN	DBGPAUSE	CAPMODE		
WO	N/A													N/A			RW	RW	RW	RW	RW	RW	RW	RW		RW	RW	RW	RW		
0	0	0	0	0	0	0	0	0	0	0	0	0	0	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

CHANNEL[CR] [31:0]

位域	名称	描述
31	CNTUPT	1: 将计数器值更新为 CNTUPTVAL 寄存器的值，该位置 1 后会在 1 个周期后自动清 0
14	CNTRST	1: 复位计数器 用户需要清零此位，不然计时器始终处于复位状态
13	SYNCFLW	1: 该通道计数器会随着上一通道计数器的一起重载。 该位对定时器通道 0 不适用。
12	SYNCIFEN	1: 定时器同步输入 SYNCI 下降沿有效
11	SYNCIREN	1: 定时器同步输入 SYNCI 上升沿有效
10	CEN	1: 使能计数器

位域	名称	描述
9	CMPINIT	输出比较初始极性 1: 通道输出初始极性为逻辑高 0: 通道输出初始极性为逻辑低 用户需要在把 CMPEN 位置 1 前设置此位
8	CMPEN	1: 使能此通道的输出比较功能
7-6	DMASEL	DMA 请求选择位： 00: 比较器 0 标志位 CMP0 置 1 时生产 DMA 请求 01: 比较器 0 标志位 CMP1 置 1 时生产 DMA 请求 10: 输入捕获标志位 CAPF 置 1 时生产 DMA 请求 11: 重载标志位 RLD 置 1 时生产 DMA 请求
5	DMAEN	1: 使能 DMA
4	SWSYNCIEN	1: 使能软件计数器同步，该位置 1 时，计数器会在 SWSYNCT 位置 1 时重载
3	DBGPAUSE	1: 调试模式下计数器会暂停
2-0	CAPMODE	输入捕获模式配置位 100: PWM 测量模式，定时器会测量输入 PWM 的周期和占空比 011: 捕获上升沿和下降沿 010: 捕获下降沿 001: 捕获上升沿 000: 关闭输入捕获

CHANNEL[CR] 位域

43.3.3 CHANNEL[CMP] (0x4 + 0x40 * n + 0x4 * m)

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

CHANNEL[CMP] [31:0]

位域	名称	描述
31-0	CMP	比较器值 0 全 1 时 (0xffffffff) 比较器输出始终为 0. 初始值为全 1，用户使用比较功能前需要配置此位和比较寄存器 1

CHANNEL[CMP] 位域

43.3.4 CHANNEL[RLD] (0xC + 0x40 * n)

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

CHANNEL[RLD] [31:0]

位域	名称	描述
31-0	RLD	重载值

CHANNEL[RLD] 位域

43.3.5 CHANNEL[CNTUPTVAL] (0x10 + 0x40 * n)

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

CHANNEL[CNTUPTVAL] [31:0]

位域	名称	描述
31-0	CNTUPTVAL	计数器更新值，当 CR 寄存器的 CNTUPT 位置 1 时，计数器更新为此值

CHANNEL[CNTUPTVAL] 位域

43.3.6 CHANNEL[CAPPOS] (0x20 + 0x40 * n)

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

CHANNEL[CAPPOS] [31:0]

位域	名称	描述
31-0	CAPPOS	输入信号上升沿捕获到的计数器值

CHANNEL[CAPPOS] 位域

43.3.7 CHANNEL[CAPNEG] (0x24 + 0x40 * n)

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

CHANNEL[CAPNEG] [31:0]

位域	名称	描述
31-0	CAPNEG	输入信号下降沿捕获到的计数器值

CHANNEL[CAPNEG] 位域

43.3.8 CHANNEL[CAPPRD] (0x28 + 0x40 * n)

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

CHANNEL[CAPPRD] [31:0]

位域	名称	描述
31-0	CAPPRD	PWM 测量模式下，测量的 PWM 周期长度

CHANNEL[CAPPRD] 位域

43.3.9 CHANNEL[CAPDTY] (0x2C + 0x40 * n)

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

CHANNEL[CAPDTY] [31:0]

位域	名称	描述
31-0	MEAS_HIGH	PWM 测量模式下，测量的 PWM 占空比长度

CHANNEL[CAPDTY] 位域

43.3.10 CHANNEL[CNT] (0x30 + 0x40 * n)

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

CHANNEL[CNT] [31:0]

位域	名称	描述
31-0	COUNTER	32 位计数器值

CHANNEL[CNT] 位域

43.3.11 SR (0x200)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																CH3CMP1F	CH3CMP0F	CH3CAPF	CH3RLDF	CH2CMP1F	CH2CMP0F	CH2CAPF	CH2RLDF	CH1CMP1F	CH1CMP0F	CH1CAPF	CH1RLDF	CH0CMP1F	CH0CMP0F	CH0CAPF	CH0RLDF
N/A																W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

SR [31:0]

位域	名称	描述
15	CH3CMP1F	通道 3 比较器 1 标志位
14	CH3CMP0F	通道 3 比较器 0 标志位
13	CH3CAPF	通道 3 输入捕获标志位
12	CH3RLDF	通道 3 重载标志位
11	CH2CMP1F	通道 2 比较器 1 标志位
10	CH2CMP0F	通道 2 比较器 0 标志位
9	CH2CAPF	通道 2 输入捕获标志位
8	CH2RLDF	通道 2 重载标志位
7	CH1CMP1F	通道 1 比较器 1 标志位
6	CH1CMP0F	通道 1 比较器 0 标志位
5	CH1CAPF	通道 1 输入捕获标志位
4	CH1RLDF	通道 1 重载标志位
3	CH0CMP1F	通道 0 比较器 1 标志位
2	CH0CMP0F	通道 0 比较器 0 标志位
1	CH0CAPF	通道 0 输入捕获标志位
0	CH0RLDF	通道 0 重载标志位

位域	名称	描述

SR 位域

43.3.12 IRQEN (0x204)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																CH3CMP1EN	CH3CMP0EN	CH3CAPEN	CH3RLDEN	CH2CMP1EN	CH2CMP0EN	CH2CAPEN	CH2RLDEN	CH1CMP1EN	CH1CMP0EN	CH1CAPEN	CH1RLDEN	CH0CMP1EN	CH0CMP0EN	CH0CAPEN	CH0RLDEN
N/A																RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

IRQEN [31:0]

位域	名称	描述
15	CH3CMP1EN	通道 3 比较器 1 标志位中断使能位
14	CH3CMP0EN	通道 3 比较器 0 标志位中断使能位
13	CH3CAPEN	通道 3 输入捕获标志位中断使能位
12	CH3RLDEN	通道 3 重载标志位中断使能位
11	CH2CMP1EN	通道 2 比较器 1 标志位中断使能位
10	CH2CMP0EN	通道 2 比较器 0 标志位中断使能位
9	CH2CAPEN	通道 2 输入捕获标志位中断使能位
8	CH2RLDEN	通道 2 重载标志位中断使能位
7	CH1CMP1EN	通道 1 比较器 1 标志位中断使能位
6	CH1CMP0EN	通道 1 比较器 0 标志位中断使能位
5	CH1CAPEN	通道 1 输入捕获标志位中断使能位
4	CH1RLDEN	通道 1 重载标志位中断使能位
3	CH0CMP1EN	通道 0 比较器 1 标志位中断使能位
2	CH0CMP0EN	通道 0 比较器 0 标志位中断使能位
1	CH0CAPEN	通道 0 输入捕获标志位中断使能位
0	CH0RLDEN	通道 0 重载标志位中断使能位

IRQEN 位域

43.3.13 GCR (0x208)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																													SWSYNCT		
N/A																													RW		
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0

GCR [31:0]

位域	名称	描述
3-0	SWSYNCT	计数器软件同步位，每一位对应一个通道

GCR 位域

44 看门狗 EWDG

本章节介绍看门狗 EWDG 的功能和特性。

44.1 特性总结

看门狗 EWDG 的主要特性有：

- 支持在特定时间窗口期喂狗
- 支持喂狗前需要解锁保护，可以配置成多种解锁方式
- 支持关键控制寄存器锁定，更新关键寄存器之前需要解锁
- 看门狗工作时钟源可配置，在总线时钟停止时仍可以在 32K 时钟下工作
- 关键控制寄存器自带奇偶校验功能
- 在芯片进入 debug 模式下可以将看门狗配置成自动停止或继续工作
- 在芯片进入低功耗模式下可以将看门狗配置成自动停止或继续工作

44.2 EWDG 架构图

看门狗的框架图如图 54。

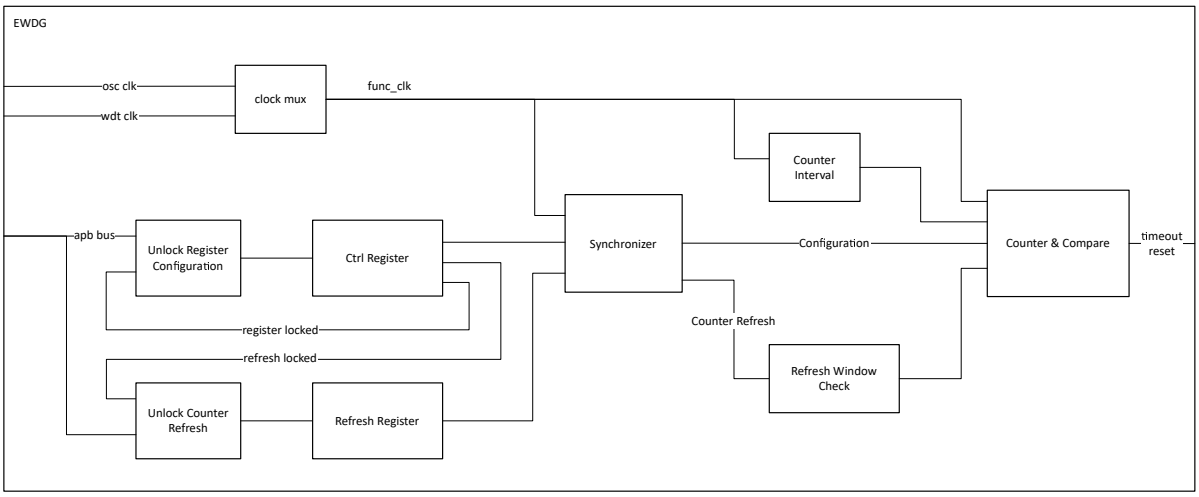


图 54: 看门狗框图

44.3 功能描述

44.3.1 功能概述

EWDG 内置 16bit 计时器，经过若干个工作时钟后 (数值可配置，最大为 32）计时器的数值增加，如果在指定的时间内没有执行喂狗，计时器的值增加大于复位数值寄存器规定的值后，就会触发复位操作。喂狗操作为向喂狗寄存器写入特定字符：32'h5A45_524F

处于安全考虑，EWDG 在更改关键寄存器和执行喂狗操作之前可以分别要求解锁，若未解锁或者解锁失败会产生违例，触发相应中断或复位。另外，支持窗口模式，喂狗操作只在特定时间窗口内才允许执行，在其他时间喂狗会产生违例，触发中断或复位。

44.3.2 时钟选择

EWDG 有三个时钟输入，分别是总线配置时钟，外部工作时钟 1 和外部工作时钟 2。总线工作时钟用来配置 EWDG 的寄存器和喂狗操作，外部工作时钟 1 和外部工作时钟 2 作为 EWDG 的主要工作时钟，可以由寄存器配置来选择。

> 一般情况下，SoC 会选择将外部工作时钟 1 连接到总线配置时钟上，以实现 EWDG 快速响应目的。将外部工作时钟 2 连接到 32K 时钟，以实现喂狗间隔时间更长和总线时钟停止时 EWDG 仍能工作的目的。

EWDG 的计时器可以根据配置在经过若干个工作时钟后递增，在收到合法的喂狗操作后计时器清零 0，在计数器到达预先配置的超时数值后产生复位操作。

EWDG 的超时时间计算公式为：

$$T_{timeout} = T_{clk\_period} * (1 << counter\_interval) * counter\_rst\_value$$

44.3.3 寄存器配置解锁和奇偶校验要求

EWDG 的控制寄存器 0 和配置寄存器 1 有奇偶校验检查。每次在写入配置寄存器 0 和配置寄存器 1 的时候要在对应寄存器的 BIT31 填入校验位，保证寄存器中 1 的个数（包含校验位）始终为偶数，否则寄存器配置失败，并产生中断或者复位操作

在寄存器保护使能打开后，一旦通过配置 EWDG 使能寄存器让 EWDG 开始工作，则开启寄存器保护功能，以后在更新控制寄存器 0，配置寄存器 1，EWDG 复位数值等寄存器之前需要先执行解锁操作。处于保护状态下的寄存器只有在正确解锁后才能更改，未正确解锁情况下更新寄存器配置会造成寄存器更新失败以及触发中断或者复位操作。

在寄存器保护功能生效后，通过给配置保护解锁寄存器写入正确的密码来解锁寄存器更新操作。密码在寄存器保护使能打开之前写入到配置保护解锁寄存器。

> 注意：一旦控制寄存器 0 的寄存器保护使能位被打开，则配置保护解锁寄存器就不允许再更改，所以该寄存器的更改要在控制寄存器 0 中的寄存器保护使能位被打开之前配置

寄存器配置保护被解锁后，可以通过软件来更改 EWDG 的控制寄存器 0/1 和复位数值等寄存器，EWDG 会在以下几种情况下自动重新打开寄存器配置保护功能：

- 若 EWDG 当前处于停止状态，则在 EWDG 重新开始工作后自动进入保护状态
- 若 EWDG 当前处于工作状态，则在经过若干总线时钟周期（周期数可配置）后自动进入保护状态

44.3.4 解锁喂狗操作

EWDG 支持喂狗保护功能。若该功能打开，则每次在喂狗之前需要先解锁喂狗操作，否则当前喂狗失败，并触发相应的中断或复位功能。

44.3.4.1 喂狗保护使能

若 EWDG 的控制寄存器 0 中的喂狗保护使能位被配置，则当 EWDG 开始工作后，进入喂狗保护模式

44.3.4.2 解锁喂狗操作

当 EWDG 喂狗保护生效后，在喂狗之前需要先执行解锁操作。解锁的操作方式可以配置成以下几种方式中的一种：

- 固定配置密码：在初始阶段向喂狗保护解锁寄存器中写入初始密码，之后向该寄存器中写入相同的配置密码来解锁
- 循环左移密码：在初始阶段向喂狗保护解锁寄存器中写入初始密码，之后向该寄存器中写入密码的循环左移结果来解锁，此后密码自动更新为本次循环左移结果
- LSFR 密码：在初始阶段向喂狗保护解锁寄存器中写入初始密码，之后向该寄存器中写入按照 $X^{15}+1$ 的公式写入 LSFR 结果来解锁，并且密码更新为本次 LSFR 结果
- 固定数值解锁：不需要配置初始密码，在解锁阶段向喂狗保护解锁寄存器中写入固定数值 16'h55AA 来解锁

注意：喂狗保护解锁寄存器只能在控制寄存器 0 中的喂狗保护使能位被使能之前配置，一旦喂狗保护使能位被打开，则喂狗保护解锁寄存器的值将不会被更改

每次喂狗保护被解锁后，EWDG 会在以下情况自动重新打开保护功能：

- 喂狗操作已经完成，则立刻重新打开保护功能，下次喂狗前需要重新解锁
- 解锁后在特定时间内（时间可以配置）没有完成解锁操作，则当前解锁时间超时，会重新进入喂狗保护状态，同时产生喂狗违例标志

44.3.5 窗口模式

EWDG 可以配置成窗口模式。在窗口模式被打开后，EWDG 只允许在计时器特定的窗口内执行喂狗操作，不在窗口时间内喂狗会导致喂狗失败并触发相应的中断或复位操作。

窗口的下限可以配置成看门狗复位数值寄存器的 4/8，5/8，6/8，7/8。窗口的上限值是在将复位数值寄存器十六等分之后，在下限的基础之上加上若干个复位数值寄存器的十六分之一。

$$window\_lower = reset\_counter\_value * 1/2 + win\_lower\_reg * 1/8$$

$$window\_upper = window\_lower + reset\_counter\_value/16 * win\_upper\_reg$$

注意：若未配置上限寄存器，或者上限超过复位数值寄存器的数值，则复位数值寄存器的值将作为喂狗的上限，在超时复位之前完成喂狗操作即可

44.3.6 Debug 模式下工作状态

EWDG 默认在芯片处于 debug 模式下暂停工作，在芯片退出 debug 模式后继续工作。可以通过寄存器配置 EWDG 在 debug 模式下仍正常工作。

44.3.7 低功耗模式下工作状态

EWDG 默认在芯片处于低功耗模式下暂停工作，在芯片退出低功耗模式后继续工作。可以通过寄存器配置 EWDG 在低功耗模式下仍正常工作。

44.4 EWDG 寄存器列表

EWDG 的寄存器列表如下：

WDG0 base address: 0xF00B0000

WDG1 base address: 0xF00B4000

PWDG base address: 0xF4128000

地址偏移	名称	描述	复位值
0x0000	CTRL0	控制寄存器 0 注意：寄存器的配置有奇偶校验检查，值为 1 的 bit 位数（包括校验位）之和应该为偶数	0x00000000
0x0004	CTRL1	控制寄存器 1 注意：寄存器的配置有奇偶校验检查，值为 1 的 bit 位数（包括校验位）之和应该为偶数	0x00000000
0x000C	OT_RST_VAL	超时复位数值寄存器	0x00000000
0x0010	WDT_REFRESH_REG	喂狗寄存器	0x00000000
0x0014	WDT_STATUS	状态寄存器	0x00000000
0x0018	CFG_PROT	配置保护解锁寄存器 注意：该寄存器需要在控制寄存器 0 的寄存器配置保护功能使能位配置前写入	0x00000000
0x001C	REF_PROT	喂狗保护解锁寄存器 注意：该寄存器需要在控制寄存器 0 的喂狗保护模式使能位配置前写入	0x00000000
0x0020	WDT_EN	看门狗使能寄存器	0x00000000
0x0024	REF_TIME	喂狗限定时间寄存器	0x00000000

表 200: EWDG 寄存器列表

44.5 EWDG 寄存器描述

EWDG 的寄存器详细说明如下：

44.5.1 CTRL0 (0x0)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD		CLK_SEL	RSVD	DIV_VALUE			WIN_EN	WIN_LOWER		CFG_LOCK	RSVD			OT_SELF_CLEAR	RSVD	REF_OT_REQ	WIN_UPPER			RSVD						REF_LOCK	REF_UNLOCK_MEC		EN_DBG	EN_LP	
N/A		RW	N/A	RW			RW	RW		RW	N/A			RW	N/A	RW	RW			N/A						RW	RW		RW	RW	
x	x	0	x	0	0	0	0	0	0	0	x	x	x	0	x	0	0	0	0	x	x	x	x	x	x	0	0	0	0	0	0

CTRL0 [31:0]

位域	名称	描述
29	CLK_SEL	工作时钟选择 0：工作时钟 1（一般为总线时钟） 1：工作时钟 2（一般为 OSC 32K）
27-25	DIV_VALUE	计时器计时间隔，计时器每隔（1 « 计时器间隔）个工作时钟后计数加 1。 注意：该寄存器最大值为 5，即每隔 32 个工作时候计时器加 1。
24	WIN_EN	窗口模式使能
23-22	WIN_LOWER	窗口模式下，喂狗时间窗口下限 00：下限是计数器超时时间的 4/8 01：下限是计数器超时时间的 5/8 10：下限是计数器超时时间的 6/8 11：下限是计数器超时时间的 7/8
21	CFG_LOCK	寄存器配置保护功能使能 一旦寄存器配置保护功能生效，则控制寄存器 0，控制寄存器 1 以及复位数值寄存器需要在解锁后才能更新
17	OT_SELF_CLEAR	超时复位自释放使能。 当产生超时复位时，可以通过配置该寄存器 bit 让看门狗在 32 个工作时钟后释放驱动 reset 信号
15	REF_OT_REQ	喂狗解锁后超时检查使能 该寄存器被配置后，喂狗操作需要在喂狗解锁后特定的时间内完成，否则本次喂狗解锁功能失效。
14-12	WIN_UPPER	喂狗窗口上限寄存器。窗口模式下，喂狗时间窗口上限。上限时间为： 上限 = 下限 + 复位数值寄存器 / 16 * 喂狗窗口上限寄存器 当上限寄存器的值为 0 或超过复位数值寄存器规定的值时，则喂狗上限为复为数值寄存器
5	REF_LOCK	喂狗保护模式使能 当该寄存器被使能后，每次喂狗前需要先解锁保护功能
4-3	REF_UNLOCK_MEC	解锁喂狗保护模式方法： 00 固定配置密码：在初始阶段向喂狗保护解锁寄存器中写入初始密码，之后向该寄存器中写入相同的配置密码来解锁 01 循环左移密码：在初始阶段向喂狗保护解锁寄存器中写入初始密码，之后向该寄存器中写入密码的循环左移结果来解锁，并且密码更新为当此循环左移结果 10 固定数值解锁：不需要配置初始密码，在解锁阶段向喂狗保护解锁寄存器中写入固定数值 16'h55AA 来解锁 11 LSFR 密码：在初始阶段向喂狗保护解锁寄存器中写入初始密码，之后向该寄存器中写入按照 X^15 + 1 的公式写入 LSFR 结果来解锁，并且密码更新为当此 LSFR 结果
2	EN_DBG	

位域	名称	描述
1-0	EN_LP	EWDOG 在低功耗模式下使能 0：EWDOG 在低功耗模式下停止工作，退出低功耗模式后恢复正常工作 1：EWDOG 在低功耗模式下继续工作

CTRL0 位域

44.5.2 CTRL1 (0x4)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD								REF_FAIL_RST_EN	REF_FAIL_INT_EN	UNL_REF_FAIL_RST_EN	UNL_REF_FAIL_INT_EN	RSVD		OT_RST_EN	RSVD									CTL_VIO_RST_EN	CTL_VIO_INT_EN	UNL_CTL_FAIL_RST_EN	UNL_CTL_FAIL_INT_EN	PARITY_FAIL_RST_EN	PARITY_FAIL_INT_EN	RSVD	
N/A								RW	RW	RW	RW	N/A		RW	N/A									RW	RW	RW	RW	RW	RW	N/A	
x	x	x	x	x	x	x	x	0	0	0	0	x	x	0	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	x	x

CTRL1 [31:0]

位域	名称	描述
23	REF_FAIL_RST_EN	喂狗操作违例复位使能 当下列喂狗违例行为发生时，EWDOG 会触发复位信号： 1）当窗口模式打开后，没有在指定的窗口内喂狗 2）当喂狗保护模式打开后，没有在喂狗前解锁 3）当喂狗解锁超时功能打开后，没有在解锁后的规定时间内完成喂狗操作 4）喂狗时写入的喂狗指令数值不正确（向喂狗寄存器写入除 32'h5A45_524F 外其他数值）
22	REF_FAIL_INT_EN	喂狗操作违例中断使能 当喂狗违例行为发生时，EWDOG 会触发中断信号：
21	UNL_REF_FAIL_RST_EN	喂狗保护解锁失败复位使能
20	UNL_REF_FAIL_INT_EN	喂狗保护解锁失败中断使能
17	OT_RST_EN	看门狗超时复位使能 当看门狗未在超时复位之前喂狗，则超时后会产生复位信号
7	CTL_VIO_RST_EN	寄存器配置保护违例复位使能。 当寄存器配置保护功能使能后，在更新配置寄存器，复位数值寄存器前未解锁保护功能，则会触发复位信号

位域	名称	描述
6	CTL_VIO_INT_EN	寄存器配置保护违例中断使能。 当寄存器配置保护功能使能后，在更新配置寄存器，复位数值寄存器前未解锁保护功能，则会触发中断信号
5	UNL_CTL_FAIL_RST_EN	寄存器配置保护解锁失败复位使能 当寄存器配置保护功能使能后，解锁配置功能失败后会触发复位操作
4	UNL_CTL_FAIL_INT_EN	寄存器配置保护解锁失败复位使能 当寄存器配置保护功能使能后，解锁配置功能失败后会触发中断操作
3	PARITY_FAIL_RST_EN	配置寄存器奇偶校验失败复位使能 在更新配置寄存器 0 和配置寄存器 1 时，如果发生奇偶校验失败会触发复位信号
2	PARITY_FAIL_INT_EN	配置寄存器奇偶校验失败中断使能 在更新配置寄存器 0 和配置寄存器 1 时，如果发生奇偶校验失败会触发中断信号

CTRL1 位域

44.5.3 OT_RST_VAL (0xC)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																OT_RST_VAL															
N/A																RW															
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

OT_RST_VAL [31:0]

位域	名称	描述
15-0	OT_RST_VAL	当计数器超过超时复位数值寄存器规定的值时，可以产生复位信号

OT_RST_VAL 位域

44.5.4 WDT_REFRESH_REG (0x10)

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

WDT_REFRESH_REG [31:0]

位域	名称	描述
31-0	WDT_REFRESH_REG	喂狗寄存器。在向该寄存器成功写入 32'h5A45_524F 数值后会执行喂狗操作

WDT_REFRESH_REG 位域

44.5.5 WDT_STATUS (0x14)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																									PARITY_ERROR	OT_RST	RSVD	CTL_UNL_FAIL	CTL_VIO	REF_UNL_FAIL	REF_VIO
N/A																									RW	RO	N/A	RW	RW	RW	RW
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	x	0	0	0	0

WDT_STATUS [31:0]

位域	名称	描述
6	PARITY_ERROR	是否发生奇偶校验错误 0：未发生奇偶校验错误 1：发生奇偶校验错误，W1C
5	OT_RST	计时器是否已经到达超时复位数值 该寄存器通过喂狗或者复位操作清 0
3	CTL_UNL_FAIL	寄存器配置保护解锁失败标志，W1C
2	CTL_VIO	寄存器配置保护违例标志，W1C
1	REF_UNL_FAIL	喂狗保护解锁失败标志，W1C
0	REF_VIO	喂狗操作失败标志，W1C

WDT_STATUS 位域

44.5.6 CFG_PROT (0x18)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD												UPD_OT_TIME				UPD_PSD															
N/A												RW				RW															
x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

CFG_PROT [31:0]

位域	名称	描述
19-16	UPD_OT_TIME	配置更新时间寄存器。寄存器配置保护解锁后，寄存器的更新限定时间。 解锁后，寄存器需要在（128 * 2 ^ 配置更新时间寄存器）个总线时钟内完成寄存器更新操作
15-0	UPD_PSD	配置保护解锁密码。16bit 密码，用于解锁配置保护功能

CFG_PROT 位域

44.5.7 REF_PROT (0x1C)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																REF_UNL_PSD															
N/A																RW															
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

REF_PROT [31:0]

位域	名称	描述
15-0	REF_UNL_PSD	喂狗保护解锁密码。16bit 初始密码，用于解锁喂狗操作

REF_PROT 位域

44.5.8 WDT_EN (0x20)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																															WDOG_EN
N/A																															RW
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0

WDT_EN [31:0]

位域	名称	描述
0	WDOG_EN	看门狗使能 0：停止看门狗 1：使能看门狗

WDT_EN 位域

44.5.9 REF_TIME (0x24)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																REFRESH_PERIOD															
N/A																RW															
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

REF_TIME [31:0]

位域	名称	描述
15-0	REFRESH_PERIOD	喂狗解锁后的喂狗限定时间 在喂狗保护功能解锁后，需要在该寄存器规定的总线时钟内完成喂狗操作

REF_TIME 位域

44.6 看门狗配置使用说明

44.6.1 配置参考：窗口工作模式，喂狗保护使能

要求 EWDG 时钟选择工作时钟 2（OSC32K），工作在窗口模式，在计时器的 4/8 到 7/8 的阶段允许喂狗。同时喂狗前需要先解锁，解锁的密码为循环左移。

参考配置为：

- 配置喂狗保护寄存器（REF_PROT），写入初始密码 PSD。如果期望在解锁后的 512 个总线时钟内完成喂狗操作，则配置喂狗限定时间寄存器（REF_TIME）为 512。
- 配置控制寄存器的喂狗保护模式解锁方法（REF_UNLOCK_MEC）值为 01，选择循环左移方式。配置喂狗保护模式使能（REF_LOCK）。
- 配置控制寄存器 0 的 CLK_SEL 为 1，选择 OSC 32K。
- 配置控制寄存器 0 的 WIN_EN 打开窗口模式，下限 WIN_LOWER 的值是 00（选择计时器超时时间的 4/8），上限在下限的基础上增加超时时间的 3/8，则配置 WIN_UPPER 的值为 6（6 / 16 + 下限值）。
- 若期待 EWDG 在超时后产生一个复位信号，配置控制寄存器 1 的超时复位使能（OT_RST_EN）。若希望未按规定要求喂狗产生一个中断，则配置控制寄存器 1 的喂狗操作违例中断使能（REF_FAIL_INT_EN）和喂狗保护解锁失败中断使能（UNL_REF_FAIL_INT_EN）。这样在喂狗失败或者喂狗解锁失败会产生中断操作

- 注意在更新配置寄存器 0 和配置寄存器 1 时要保证奇偶校验通过，数值为 1 的位数是偶数。
- 配置使能寄存器 WDT_EN，将 1 写入 WDOG_EN，看门狗开始工作
- 当计时器在允许喂狗的窗口期准备喂狗时，首先解锁喂狗保护。将喂狗保护寄存器（REF_PROT）里的初始密码循环左移的结果写入到 REF_PROT 寄存器中解锁喂狗操作，同时喂狗保护寄存器里的密码更新到了初始密码循环左移的结果。
- 在 512 个总线时钟内向喂狗寄存器（WDT_REFRESH_REG）写入保留字 32'h5A45524F 完成喂狗操作。

44.6.2 配置参考：寄存器配置锁定和解锁

在配置寄存器已经锁定后希望更新配置寄存器 1，将解锁失败的行为由中断改成复位。

- 将配置保护寄存器的配置更新时间（UPD_OD_TIME）配置成 2，希望在配置解锁后的 512 个周期（$128*2^2$）内完成配置。将初始密码 PSD 写入 UPD_PSD。
- 将控制寄存器 0 的寄存器保护使能（CFG_LOCK）打开。
- EWDG 使能后（使能寄存器写 1），配置保护功能打开，如果未经解锁去更新控制寄存器和复位数值寄存器位产生违例
- 希望解锁时，先将密码写入配置保护寄存器解锁配置保护。写入使能寄存器位 0x0 停止 EWDG（推荐，防止在临界状态下产生意外情况）。然后在 512 个系统时钟周期内更新控制寄存器 1 将喂狗操作违例复位使能（REF_FAIL_RST_EN）打开，配置使能寄存器 0x01 重新启动 EWDG，同时配置保护功能在 EWDG 工作后会自动重新开启。

45 通讯外设概述

本章节介绍了本产品的通讯外设。

45.1 通用异步收发器 UART，PUART

本产品支持 9 个通用异步收发器，支持与外部设备实现用于双向异步通信。

8 个通用异步收发器位于系统电源域，称为 UART0~7。

1 个通用异步收发器位于电源管理域，称为 PUART。PUART 支持在系统电源域掉电时保持工作，产生的中断可用于低功耗唤醒。PUART 不支持生成 DMA 请求。

45.2 串行外设总线 SPI

本产品支持 4 个串行外设总线 SPI，支持与外部设备实现全双工，同步的通信。SPI 支持经典单向 2 路数据线模式，即 MISO，MOSI，也支持双向 2 路，4 路数据线模式，即 Dual-SPI 和 Quad-SPI。

45.3 集成电路总线 I2C

本产品支持 4 个集成电路总线 I2C。I2C 支持标准模式最大 100Kbps，快速模式最大 400Kbps 和快速 + 模式最大 1Mbps。

45.4 控制器局域网 MCAN

本产品支持 4 个控制器局域网 MCAN0~4，MCAN 遵循 CAN 总线协议 2.0A 和 2.0B 协议，并支持 CAN-FD 协议。

本产品上，MCAN 支持时间触发 CAN 通信，MCAN 可以通过精确时间协议模块 PTPC 的时间戳输出端口直接载入时间戳信息。

45.5 局域互联网络 LINV2

本产品支持 4 个局域互联网络 LINV2。

45.6 精确时间协议模块 PTPC

本产品支持 1 个精确时间协议模块 PTPC。PTPC 支持 2 套时间戳，可以生成秒和纳秒为分辨率的精确时间戳信息供网络不同节点间同步时钟。

PTPC 支持 2 个时间戳输出端口，分别连接到 CAN0~1，CAN 模块可以从这些端口直接载入时间戳信息。

PTPC 支持 2 个时间戳输入捕获信号 CAP0 和 CAP1，支持根据输入捕获信号捕获当前的时间戳。CAP0 和 CAP1 来自互联管理器 TRGM0~1。PTPC 支持在时间戳匹配时，输出 2 个时间戳输出比较 CMP0 和 CMP1。CMP0 和 CMP1 输出到互联管理器 TRGM0~1。具体连接请查阅节 32.8。

45.7 通用串行总线 USB

本产品支持 1 个通用串行总线 USB，USB 集成 HS PHY，支持 USB2.0 低速，全速，和高速传输。USB 支持 Host，Device 和 OTG 模式。

本产品上，USB 控制器支持输出 USB 的 SOF (桢起始 Start of Frame) 信号，可以通过互联管理器 TRGM 连

接到片上的其他模块。

46 通用异步收发器 UART

本章节介绍通用异步收发器 UART 的功能和特性。

46.1 特性总结

本章节介绍通用异步收发器 UART 的主要特性：

- 支持 5~9 位数据长度
- 可配置停止位：1 位，1.5 位或者 2 位
- 可配置奇偶校验位：奇校验，偶校验，粘校验位
- 支持 DMA 数据传输
- 支持可配置波特率，支持独立的波特率生成时钟
- 支持硬件流控
- 支持奇偶校验错误，数据 FIFO 溢出等错误检测
- 16 字节的 TXFIFO 和 RXFIFO
- 支持各类中断

UART 的框图如图 55。

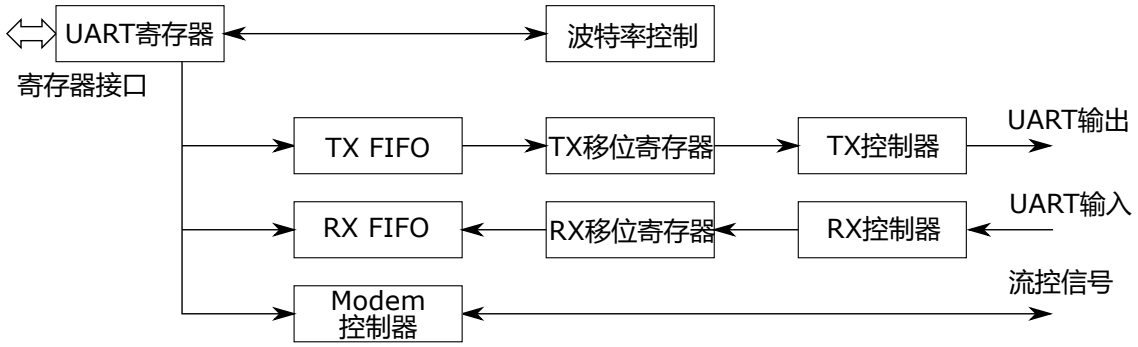


图 55: UART 框图

46.2 功能描述

本章节描述通用异步收发器 UART 的功能。

46.2.1 UART 发送

UART 发送端由 TX FIFO 和 TX 移位寄存器和 TX 控制器组成。

TX FIFO 包含待发送的数据，并将数据传送到 TX 移位寄存器。

TX 移位寄存器为并行-串行转换器，把发送数据转换成串行的比特流。

TX 控制器在发送一个数据时，会生成一个 START 位，一个可配置的奇偶校验位 PARITY 位，和一个长度可配的停止位。用户可以LCR寄存器 (Line Control Register) 来配置奇偶效验位和停止位。

TXFIFO 的写入口是THR寄存器 (Transmitter Holding Register)。用户需要把FCR寄存器 (FIFO Control Register) 的 FIFOE 位置 1，来打开 TXFIFO。

46.2.2 UART 接收

UART 接收端由 RX FIFO 和 RX 移位寄存器和 RX 控制器组成。

RX 控制器使用波特率控制模块生成的过采样时钟，对输入的每一位进行采样，并把收到的每一位移入 RX 移位寄存器。

RX 移位寄存器对数据进行串行-并行转换，并把数据存入 RX FIFO。

RX FIFO 的读入口是RBR寄存器 (Receiver Buffer Register)，用户需要把FCR寄存器 (FIFO Control Register) 的 FIFOE 位置 1，来打开 RXFIFO。

RX 控制器支持检测接收数据过程中的错误，如奇偶校验位错误，停止位错误，RX FIFO 溢出等。

46.2.3 波特率控制

波特率控制模块对 UART 时钟分频，得到波特率时钟，分频器为 16 位长，分别位于 2 个寄存器中，每个寄存器存放 8 位分频值。分频值的 MSB 位于DLM寄存器 (Divisor Latch MSB)，而 LSB 位于DLL寄存器 (Divisor Latch LSB)。

UART 时钟与波特率的比率就是过采样率，过采样率OSC存放在OSCR寄存器 (Over Sample Control Register)。OSC默认值为 16。

分频值的公式如 (46.2.3)：

$$Divisor = \frac{f_{uart\_clock}}{Baudrate \times OSR}$$

RX 控制器利用过采样时钟对输入数据进行采样。假设过采样率OSC为 16 当检测到输入信号第一个下降沿时 (起始 START 位)，计数器从 1 开始计数直到 16，在计数到 8 时，RX 控制器对输入数据采样。计数器在计数到 16 后，会复位到 1，以此采样下一位数据，循环往复，直到停止 STOP 位。TX 控制器同样利用过采样时钟来生成输出数据流。

46.2.4 Modem 控制器

Modem 控制器提供 Modem 控制功能，也支持自动流控，可以进一步降低软件控制开销。

UART 的发送接收流控通过 RTS / CTS 握手实现。流控可以避免因为数据传输速率超过数据消化速率而引起的 overrun 错误。

UART 支持自动流控，包括 auto-RTS 和 auto-CTS 功能，前者用于数据接收，后者用于数据发送。

打开 auto-RTS 功能时，UART 的 RTS 输出需要连接到对方 UART 的 CTS 信号。

- N 是 RX FIFO 的触发阈值
- Bn 是 RX FIFO 接收到的字节数目
- 当 $B_n \geq N$ 时，RTS 置低
- 当 $B_n < N$ 时，RTS 会自动置高，提示对方 UART 可以发送数据

打开 auto-CTS 功能时，UART 的 CTS 输入连接到对方 UART 的 RTS 信号。UART 只有在 CTS 检测到输入低时，才会发送数据。CTS 必须在当前传输数据的停止 STOP 位发出前置高，这样 UART 才能不发出下一个数据。

46.2.5 Loopback 模式

UART 支持 Loopback 模式。

Loopback 模式打开时，UART 的 TX 连接到 RX，RTS 连接到 CTS。通过THR寄存器发出的数据可以从RBR寄存器接收到。

46.2.6 DMA

UART 支持发送 DMA 请求和接收 DMA 请求。

用户通过FCR寄存器配置 RX FIFO 的触发阈值RFIFOT，当 RX FIFO 接收的数据超过阈值时，生成 DMA 接收请求。

用户通过FCR寄存器配置 TX FIFO 的触发阈值TFIFOT，当 TX FIFO 的待发送数据少于阈值时，生成 DMA 发送请求。

46.3 寄存器说明

UART 的寄存器列表如下：

UART0 base address: 0xF0040000

UART1 base address: 0xF0044000

UART2 base address: 0xF0048000

UART3 base address: 0xF004C000

UART4 base address: 0xF0050000

UART5 base address: 0xF0054000

UART6 base address: 0xF0058000

UART7 base address: 0xF005C000

PUART base address: 0xF4124000

地址偏移	名称	描述	复位值
0x0004	IDLE_CFG	空闲配置寄存器	0x00000000
0x0008	ADDR_CFG	地址匹配寄存器	0x00000000
0x000C	IIR2	中断识别寄存器 2	0x00000001
0x0010	CFG	配置查询寄存器	0x00000000
0x0014	OSCR	过采样控制寄存器	0x00000010
0x0018	FCRR	FIFO 控制寄存器配置	0x00000000
0x001C	MOTO_CFG	电机系统控制寄存器	0x00000000
0x0020	RBR	接收缓冲寄存器（当 DLAB=0 时）	0x00000000
0x0020	THR	发送缓冲寄存器（当 DLAB=0 时）	0x00000000
0x0020	DLL	分频参数低位寄存器（当 DLAB=1 时）	0x00000001
0x0024	IER	中断使能寄存器（当 DLAB=0 时）	0x00000000
0x0024	DLM	分频参数高位寄存器（当 DLAB=1 时）	0x00000000

地址偏移	名称	描述	复位值
0x0028	IIR	中断 ID 寄存器	0x00000001
0x0028	FCR	FIFO 控制寄存器	0x00000000
0x002C	LCR	传输控制寄存器	0x00000000
0x0030	MCR	流量控制寄存器	0x00000000
0x0034	LSR	传输状态寄存器	0x00000000
0x0038	MSR	流控状态寄存器	0x00000000

表 201: UART 寄存器列表

46.4 寄存器详细信息

UART 的寄存器详细说明如下：

46.4.1 IDLE_CFG (0x4)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD						TX_IDLE_COND	TX_IDLE_EN	TX_IDLE_THR								RSVD				RXEN	RSVD	RX_IDLE_COND	RX_IDLE_EN	RX_IDLE_THR							
N/A						RW	RW	RW								N/A				RW	N/A	RW	RW	RW							
x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	x	x	x	x	0	x	0	0	0	0	0	0	0	0	0	0

IDLE_CFG [31:0]

位域	名称	描述
25	TX_IDLE_COND	发送空闲检测条件 0 - RX 引脚为高电平时认为是空闲 1 - UART 状态机为空闲时认为是空闲
24	TX_IDLE_EN	使能 UART 发送空闲检测 0 - 禁用 1 - 使能
23-16	TX_IDLE_THR	串口发送空闲检测的阈值（以 bit 为单位）
11	RXEN	UART 接收使能 0 - 将输入信号强制为高电平，防止 RX 引脚跳动导致的错误接收 1 - 正常工作，输入信号反应 RX 引脚的状态 软件需要在配置 pinmux 后将此位置位
9	RX_IDLE_COND	接收空闲检测条件 0 - RX 引脚为高电平时认为是空闲 1 - UART 状态机为空闲时认为是空闲

位域	名称	描述
8	RX_IDLE_EN	使能 UART 接收空闲检测 0 - 禁用 1 - 使能 使用地址匹配功能时需打开
7-0	RX_IDLE_THR	串口接收空闲检测的阈值（以 bit 为单位）

IDLE_CFG 位域

46.4.2 ADDR_CFG (0x8)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD											TXEN_9BIT	RXEN_ADDR_MSB	RXEN_9BIT	A1_EN	A0_EN	ADDR1								ADDR0							
N/A											RW	RW	RW	RW	RW	RW								RW							
x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ADDR_CFG [31:0]

位域	名称	描述
20	TXEN_9BIT	设置 9bit 发送模式，将第一个字符 MSB 设置作为地址标志，其他字符设置为 0。
19	RXEN_ADDR_MSB	在接收中设置使用 MSB 作为地址标志（实际上，该操作在软件正确设置地址 0/地址 1 中的 MSB 时完成） 清零：使用第一个字符作为地址 仅在使能地址匹配功能时使用
18	RXEN_9BIT	设置使用 9bit 接收模式，仅当 rxen_addr_msb 置 1 时有效
17	A1_EN	使能地址 1 与第一个字符比较 如果 a1_en 或 a0_en, 地址不匹配，则不会接收数据 如果 ~a1_en 或 ~a0_en, 会正常接收所有数据 注意：如果使能了地址匹配功能应当设置 idle_timeout_en
16	A0_EN	使能地址 0 与第一个字符比较
15-8	ADDR1	地址 1 在 9bit 模式，这是一个完整的地址字节 对于其他模式（8/7/6/5bit），MSB 应当被设置为地址标志 如果希望地址 0 在 8 位模式下被匹配，应当设置 addr1 为 0x80
7-0	ADDR0	地址 0

ADDR_CFG 位域

46.4.3 IIR2 (0xC)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RXIDLE_FLAG	TXIDLE_FLAG	ADDR_MATCH	ADDR_MATCH_IDLE	DATA_LOST	RSVD																			FIFOED		RSVD		INTRID			
W1C	W1C	W1C	W1C	W1C	N/A																			RO		N/A		RO			
0	0	0	0	0	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	x	x	0	0	0	1

IIR2 [31:0]

位域	名称	描述
31	RXIDLE_FLAG	UART 接收空闲标志位，在 rxd 引脚为低电平且 rx idle timeout 后产生，写 1 清 0 0 - UART 忙 1 - UART 空闲
30	TXIDLE_FLAG	UART 发送空闲标志位，在 txd 引脚为低电平且 tx idle timeout 后产生，写 1 清 0 0 - UART 忙 1 - UART 空闲
29	ADDR_MATCH	地址匹配标志，在地址匹配功能使能且地址匹配时产生，写 1 清 0 注意：在使能 DMA 时，地址字节此时会被 DMA 搬走，用户可以等到随后的地址匹配空闲中断查看全部的数据（包括地址）
28	ADDR_MATCH_IDLE	地址匹配和接收空闲中断，在地址匹配事件发生后接收总线空闲时产生，写 1 清 0
27	DATA_LOST	数据丢失标志位，在地址匹配前数据丢失时产生，写 1 清 0 当地址匹配事件未发生时，它不会出现
7-6	FIFOED	当 FIFO 被使能（FIFOE 位为 1）时，FIFOED 的值为 0x3

位域	名称	描述
3-0	INTRID	中断 ID 6: 接收状态 当发生 overrun 错误，奇偶校验错误，帧错误或 line break 时产生 读寄存器 LSR 清零 4: 接收数据有效 如果 FIFOE 不使能，表示接收到一个数据在 RBR 中 如果 FIFOE 使能，表示接收到的数据达到 RFIFOT 触发水平 通过读接收缓存寄存器 RBR 直到剩余数据小于设置的触发水平 清零 0xC: 字符超时 当 FIFOE 被使能，没有字符从接收 FIFO 中被移出或移入，并且 在最后四个字符时间内接收 FIFO 中至少有一个字符 读接收寄存器 RBR 直到 FIFO 为控清零 2: 发送缓存寄存器空 如果 FIFOE 不使能，表示 1 字节的 THR 为空 如果 FIFOE 使能，表示整个 16 字节的发送 FIFO 为空 向 THR 写入数据清零或读取 IIR 清零该位 0: modem 状态 当 modem 状态寄存器 MSR 的 bit[3:0] 的值不是 0 时置位，指示 下列事件中的一种发生： Clear To Send (CTS), Data Set Ready (DSR), Ring Indicator (RI), or Data Carrier Detect (DCD) 读 MSR 清零该位

IIR2 位域

46.4.4 CFG (0x10)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																														FIFOSIZE	
N/A																														RO	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

CFG [31:0]

位域	名称	描述
1-0	FIFOSIZE	RXFIFO 和 TXFIFO 深度： 0：16 字节 1：32 字节 2：64 字节 3：128 字节

CFG 位域

46.4.5 OSCR (0x14)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																											OSC				
N/A																											RW				
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	1	0	0	0	0

OSCR [31:0]

位域	名称	描述
4-0	OSC	过采样率设置： OSC=0：过采样率为 32 OSC<=8：过采样率为 8 OSC>8：过采样率为 OSC 的值 OSC 设置值必须是偶数，若写入了奇数则会被自动转换为一个偶数值

OSCR 位域

46.4.6 FCRR (0x18)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD								FIFOT4EN	RSVD			TFIFOT4				RSVD				RFIFOT4				RFIFOT		TFIFOT		DMAE	TFIFORST	RFIFORST	FIFOE
N/A								RW	N/A			RW				N/A				RW				RW		RW		RW	WO	WO	RW
x	x	x	x	x	x	x	x	0	x	x	x	0	0	0	0	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0

FCRR [31:0]

位域	名称	描述
23	FIFOT4EN	1：使用 4bit 宽度的 fifo 门限（TFIFOT4 和 RFIFOT4） 0：使用 2bit 位宽的 fifo 门限（TFIFOT 和 RFIFOT）

位域	名称	描述
19-16	TFIFOT4	发生 fifo 门限 0-1 字节 ... 0xF - 16 字节 当 fifo 中的数据量小于门限值时产生 tx_dma_req
11-8	RFIFOT4	接收 fifo 门限 0-1 字节 ... 0xF - 16 字节 当 fifo 中的数据量达到门限值时产生 rx_dma_req 同时会产生 rxdata 中断如果该中断使能
7-6	RFIFOT	接收 FIFO trigger level
5-4	TFIFOT	发送 FIFO trigger level
3	DMAE	DMA 使能 0-不使能 1-使能
2	TFIFORST	发送 FIFO 复位，写 1 会清除发送 FIFO 中的所有数据并复位指针，该位自动清零
1	RFIFORST	接收 FIFO 复位，写 1 会清除接收 FIFO 中的所有数据并复位指针，该位自动清零
0	FIFOE	FIFO 使能。 写 1 会使能发送 FIFO 和接收 FIFO，该位发生翻转时会复位所有的 FIFO

FCRR 位域

46.4.7 MOTO_CFG (0x1C)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SWTRG	RSVD															TXSTP_BITS								HWTRG_EN	TRG_MODE	TRG_CLR_RFIFO	TXSTOP_INSERT	RSVD			
WO	N/A															RW								RW	RW	RW	RW	N/A			
0	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	x	x	x	x

MOTO_CFG [31:0]

位域	名称	描述
31	SWTRG	软件触发。不能同时使用软件和硬件触发，否则结果状态未知 硬件自动清零

位域	名称	描述
15-8	TXSTP_BITS	发送停止位长度 当 TXSTOP_INSERT 使能时，设置的停止位长度会被加到每一个字节 0-1bit 0xff-256bits
7	HWTRG_EN	设置硬件触发（由电机系统触发）
6	TRG_MODE	设置触发模式 软件需要先将需要的数据写入发送缓冲中，此时 UART 将不会开始传输 用户需要发送触发信号（硬件或软件），UART 将发送所有的数据直到发送 FIFO 为空 注意：硬件触发信号通过 trgmux 的脉冲信号
5	TRG_CLR_RFIFO	设置在发送触发 (硬件或软件) 时, 清除接收 fifo，以避免接收不期望的数据在接收 fifo
4	TXSTOP_INSERT	使能插入停止位在每个发送字节直到发送 fifo 为空 注意：当该位被设置时，此时不支持 1.5/2 功能，LCR.STB 应当被设置为 0

MOTO_CFG 位域

46.4.8 RBR (0x20)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																								RBR							
N/A																								RO							
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0

RBR [31:0]

位域	名称	描述
7-0	RBR	接收数据读取

RBR 位域

46.4.9 THR (0x20)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																								THR							
N/A																								WO							
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
THR [31:0]																															

位域	名称	描述
7-0	THR	发送数据写入

THR 位域

46.4.10 DLL (0x20)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																								DLL							
N/A																								RW							
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	1
DLL [31:0]																															

位域	名称	描述
7-0	DLL	当 DLAB 为 1 时，该寄存器配置分频参数的低 8 位

DLL 位域

46.4.11 IER (0x24)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ERXIDLE	ETXIDLE	EADDRM	EADDRM_IDLE	EDATLOST	RSVD																							EMSI	ELSI	ETHEI	ERBI
RW	RW	RW	RW	RW	N/A																							RW	RW	RW	RW
0	0	0	0	0	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0
IER [31:0]																															

位域	名称	描述
31	ERXIDLE	接收空闲中断使能: 0 - 当检测到 UART 空闲时，不触发中断 1 - 当检测到 UART 空闲时，触发空闲中断
30	ETXIDLE	使能发送空闲中断
29	EADDRM	使能地址匹配中断
28	EADDRM_IDLE	使能地址匹配并接收空闲中断
27	EDATLOST	使能数据丢失中断

位域	名称	描述
3	EMSI	流控中断使能： 当自动流控功能关闭时，CTS 状态的变化会触发中断。 当自动流控使能时，流控中断不会被使能，CTS 状态用来自动控制发送行为。
2	ELSI	接收状态中断使能
1	ETHEI	发送状态中断使能
0	ERBI	接收数据有效中断使能 character 超时中断使能

IER 位域

46.4.12 DLM (0x24)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																								DLM							
N/A																								RW							
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0

DLM [31:0]

位域	名称	描述
7-0	DLM	当 DLAB 为 1 时，该寄存器配置分频参数的高 8 位

DLM 位域

46.4.13 IIR (0x28)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RXIDLE_FLAG	RSVD																							FIFOED		RSVD		INTRID			
W1C	N/A																							RO		N/A		RO			
0	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	x	x	0	0	0	1

IIR [31:0]

位域	名称	描述
31	RXIDLE_FLAG	UART 空闲标志位 0 - UART 忙 1 - UART 空闲 注意：写 1 清此位时，注意不要改写 FCR 寄存器，因为 FCR 寄存器和 IIR 寄存器是同一个地址。 建议用 IIR2 操作中断
7-6	FIFOED	当 FIFO 被使能（FIFOE 位为 1）时，FIFOED 的值为 0x3
3-0	INTRID	中断 ID 6: 接收状态 当发生 overrun 错误，奇偶校验错误，帧错误或 line break 时产生 读寄存器 LSR 清零 4: 接收数据有效 如果 FIFOE 不使能，表示接收到一个数据在 RBR 中 如果 FIFOE 使能，表示接收到的数据达到 RFIFOT 触发水平 通过读接收缓存寄存器 RBR 直到剩余数据小于设置的触发水平 清零 0xC: 字符超时 当 FIFOE 被使能，没有字符从接收 FIFO 中被移出或移入，并且在最后四个字符时间内接收 FIFO 中至少有一个字符 读接收寄存器 RBR 直到 FIFO 为控清零 2: 发送缓存寄存器空 如果 FIFOE 不使能，表示 1 字节的 THR 为空 如果 FIFOE 使能，表示整个 16 字节的发送 FIFO 为空 向 THR 写入数据清零或读取 IIR 清零该位 0: modem 状态 当 modem 状态寄存器 MSR 的 bit[3:0] 的值不是 0 时置位，指示下列事件中的一种发生： Clear To Send (CTS), Data Set Ready (DSR), Ring Indicator (RI), or Data Carrier Detect (DCD) 读 MSR 清零该位

IIR 位域

46.4.14 FCR (0x28)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																								RFIFOT		TFIFOT		DMAE	TFIFORST	RFIFORST	FIFOE
N/A																								WO		WO		WO	WO	WO	WO
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0

FCR [31:0]

位域	名称	描述
7-6	RFIFOT	接收 FIFO trigger level
5-4	TFIFOT	发送 FIFO trigger level
3	DMAE	DMA 使能
2	TFIFORST	发送 FIFO 复位，写 1 会清除发送 FIFO 中的所有数据并复位指针，该位自动清零
1	RFIFORST	接收 FIFO 复位，写 1 会清除接收 FIFO 中的所有数据并复位指针，该位自动清零
0	FIFOE	FIFO 使能。 写 1 会使能发送 FIFO 和接收 FIFO，该位发生翻转时会复位所有的 FIFO

FCR 位域

46.4.15 LCR (0x2C)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																								DLAB	BC	SPS	EPS	PEN	STB	WLS	
N/A																								RW	RW	RW	RW	RW	RW	RW	
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0

LCR [31:0]

位域	名称	描述
7	DLAB	分频参数访问控制位，为 1 时访问相关寄存器会定向到分频参数控制
6	BC	break 控制
5	SPS	固定奇偶校验位： 1：parity bit 会固定为 0 或 1，根据 EPS 位设定 0：parity bit 不固定
4	EPS	奇偶校验选择： 1：偶校验（data 和 parity bits 中有偶数个 1） 0：奇校验
3	PEN	奇偶校验使能，使能后在发送数据时一个校验位会被放置在第一个 STOP 之前，在接收数据时会进行校验位检查
2	STB	STOP 位数量： 0：1 位 1：STOP 位的数量基于 WLS 的设置， WLS 为 0 时，STOP 为 1.5 位 WLS 为 1，2，3 时，STOP 为 2 位

位域	名称	描述
1-0	WLS	word 长度设置 0：5 位 1：6 位 2：7 位 3：8 位

LCR 位域

46.4.16 MCR (0x30)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																										AFE	LOOP	RSVD		RTS	RSVD
N/A																										RW	RW	N/A		RW	N/A
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	x	x	0	x

MCR [31:0]

位域	名称	描述
5	AFE	自动流控使能： 0：关闭 1：当 RTS 为 0 时，仅 CTS 做自动流控，当 RTS 为 1 时，CTS 和 RTS 自动流控都使能
4	LOOP	loopback 使能
1	RTS	请求 RTS 流控：设为 1 时，RTS 信号输出 0

MCR 位域

46.4.17 LSR (0x34)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RXIDLE	TXIDLE	RSVD									RFIFO_NUM					RSVD			TFIFO_NUM					ERRF	TEMT	THRE	LBREAK	FE	PE	OE	DR
RO	RO	N/A									RO					N/A			RO					RO	RO	RO	RO	RO	RO	RO	RO
0	0	x	x	x	x	x	x	x	x	x	0	0	0	0	0	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0

LSR [31:0]

位域	名称	描述
31	RXIDLE	接收空闲标志
30	TXIDLE	发送空闲标志
20-16	RFIFO_NUM	接收 FIFO 中的数据数量

位域	名称	描述
12-8	TFIFO_NUM	发送 FIFO 中的数据数量
7	ERRF	接收 FIFO 错误： 在 FIFO 模式下，当出现了奇偶校验错误，帧错误或者传输打断时该位会置 1。 该位读清零。
6	TEMT	发送 FIFO 空
5	THRE	发送 FIFO 空，该位为 1 时，如果对应中断使能，则会触发中断
4	LBREAK	传输打断。 当 RXD 信号拉低持续超过一个 full-word 时该位会被置 1。 一个 full-word 包括 START，data，parity 和 STOP 位。 该位读清零。
3	FE	帧错误。 当接收的 STOP 位为低时该位置 1。 该位读清零。
2	PE	奇偶校验错误。 该位读清零。
1	OE	接收数据过载。
0	DR	接收数据就绪。 存在有效的接收数据时该位置 1。 在所有接收数据被读取后该位清零。

LSR 位域

46.4.18 MSR (0x38)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																											CTS	RSVD			DCTS
N/A																											RO	N/A			RC
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	x	x	x	0

MSR [31:0]

位域	名称	描述
4	CTS	CTS 信号状态： 0：CTS 输入为高电平 1：CTS 输入为低电平
0	DCTS	该位置 1 表示在上次该位被读取之后，CTS 输入信号曾发生了翻转。

MSR 位域

47 串行外设总线 SPI

串行外设总线 SPI 目前支持的特性如下：

功能	主要特性
通信方式	· 支持 1，2，4 线工作模式 · 支持主机模式或者从机模式 · 主机模式最高支持 80MHz · 从机模式最高支持 20MHz
时钟特性	· SPI 时钟相位，极性可配置 · CS 与 SCLK 接口时序可配置 · 时钟源与 SCLK 频率比可配置
数据格式	· MSB 先行和 LSB 先行可配置 · 命令固定 8bit，可任意定义 · 地址长度 8，16，24，32bit 可配置 · 数据长度可配置，支持单次超过 512 个数 · 支持最多 4 个 cs
数据缓冲	· TX 有 8 字深，32 位宽的 FIFO 缓冲区域 · RX 有 8 字深，32 位宽的 FIFO 缓冲区域
中断	· 从机命令中断，从机模式下收到属命令时产生中断 · SPI 输出结束时产生中断 · TXFIFO 有效数据小于等于设置阈值时产生中断 · RXFIFO 有效数据小于等于设置阈值时产生中断 · TXFIFO 欠载时产生中断 · RXFIFO 溢出时产生中断

表 202: I2S 主要特性

47.1 SPI 架构图

SPI 的框图如图 56。

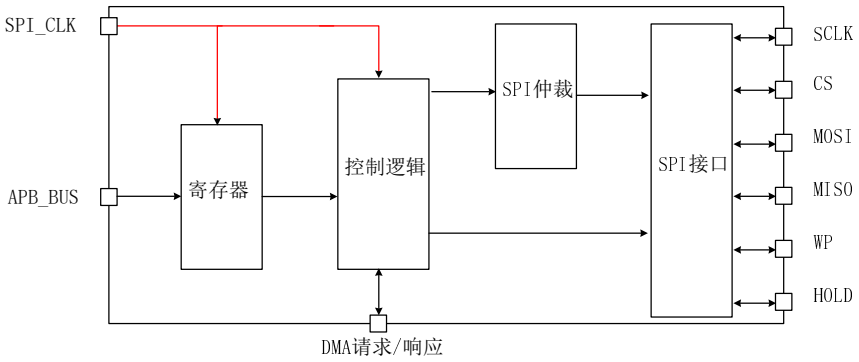


图 56: SPI 框图

47.2 管脚说明

管脚	方向	功能描述
SCLK	输入输出	时钟
CS	输入输出	片选
MOSI	输入输出	MOSI /data[0]
MISO	输入输出	MISO /data[1]
WP	输入输出	WP /data[2]
HOLD	输入输出	HOLD /data[3]

表 203: SPI 管脚说明

47.3 功能描述

47.3.1 一般说明

串行外设总线 SPI 支持主机模式和从机模式。主机模式下可以控制多种外设，从机模式下可以接受主机的请求完成数据交换。

SPI 引脚功能

- SCLK：时钟信号，主机输出，从机输入。
- CS：片选信号，主机输出，从机输入。
- MOSI：
  - 主机输出，从机输入，数据来自主机；
  - 2 线，4 线模式时，作为 data[0], 方向输入输出。
- MISO：
  - 主机输入，从机输出，数据来自从机；
  - 2 线，4 线模式时，作为 data[1], 方向输入输出。
- WP：
  - WP 端口；
  - 2 线，4 线模式时，作为 data[2], 方向输入输出。
- HOLD
  - HOLD 端口
  - 2 线，4 线模式时，作为 data[3], 方向输入输出。

47.3.2 主机模式

SPI 在主机模式下控制发起传输。SPI 传输的格式和接口时序可以通过寄存器编程。

SPI 传输包括命令、地址和数据字段，SPI 控制器提供专用的寄存器用来存储这些字段。

数据寄存器在接受和发送时是共用的。

数据传输可以由寄存器访问发起，也可以由 DMA 发起。

SPI 控制器提供发送和接受缓冲阈值中断，用以减轻流控的软件负担。

控制器还可以在传输结束时产生中断。

除支持标准的 SPI 传输以外，控制器允许软件直接控制 SPI 接口上的信号。这一功能可以用来实现接口上的

特殊时序。

主机模式下，数据传输格式：cmd+addr+data ；如图 57。

- cmd:8bit，用户根据使用自行定义。
- addr:8，16，24，32bit 可配置。
- data: 长度根据应用配置，支持多种传输模式，传输顺序查看寄存器 TRANSCTRL.TRANSMODE。

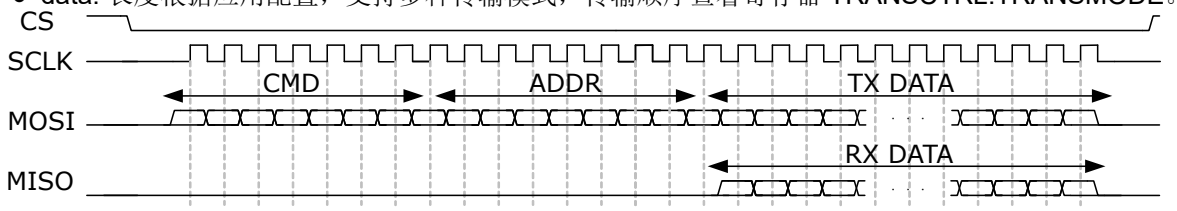


图 57: SPI 主机模式传输

47.3.3 从机模式

从机模式下，控制器接收接口上的信号，把信号解释为命令、填充和数据。命令和填充字段固定为 8 位，数据段的长度由接收到的命令和寄存器的设置决定。

从机模式下，内部配置有以下命令可以直接使用，该模式下还可以支持用户在寄存器中自定义的任意 8bit 命令。

命令名称	byte1(命令)	byte2(填充)	数据
1 线读取控制器状态	0x05	dummy	32bit 状态
2 线读取控制器状态	0x15	dummy	32bit 状态
4 线读取控制器状态	0x25	dummy	32bit 状态
1 线读取数据	0x0b	dummy	数据
2 线读取数据	0x0c	dummy	数据
4 线读取数据	0x0e	dummy	数据
1 线写入数据	0x51	dummy	数据
2 线写入数据	0x52	dummy	数据
4 线写入数据	0x54	dummy	数据
自定义命令	任意 8bit	dummy	自定义

表 204: 从机模式支持的命令

对于读取状态的命令，控制器返回状态寄存器的值。

对于数据读写指令，传输由命令-填充-数据组成。

对于用户自定义指令，数据段格式由寄存器配置。例如，传输模式被设置成（填充，写入），填充周期为 (dummyCnt+1)*((DataLen+1)/IO 宽度)，填充阶段字段被丢弃，之后的数据保存到数据寄存器

从机模式下，数据传输格式：cmd+dummy+data；如图 58。

- cmd:8bit，内部已配置命令或用户自行定义。
- dummy:8bit。
- data: 长度根据应用配置，支持多种传输模式，传输顺序查看寄存器 TRANSCTRL.TRANSMODE。

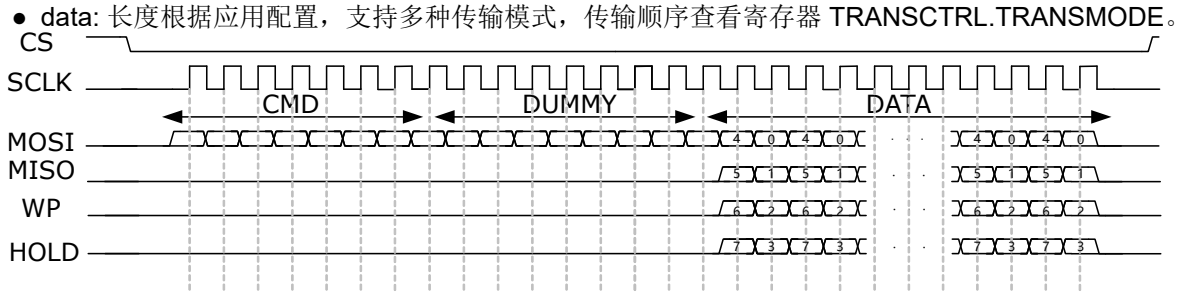


图 58: SPI 从机模式传输

47.3.4 2 线模式

2 线模式下把 MISO 和 MOSI 都用作双向信号从而获得 2 倍的带宽。

2 线模式下提供两种传输方式。一种方式地址和数据均使用 2 线传输。另一种方式下，只有数据使用 2 线传输。传输方式可以通过寄存器配置。

47.3.5 4 线模式

4 线模式把 MOSI、MISO、WP 和 HOLD 均作为双向数据信号使用，获得 4 倍带宽。

4 线模式下提供两种传输方式。一种方式地址和数据均使用 4 线传输。另一种方式下，只有数据使用 4 线传输。传输方式可以通过寄存器配置。

47.3.6 中断

SPI 支持生成多种中断，这些中断状态在中断状态寄存器 INTRST(0x3c) 相应标志位里。

这些中断共用一根中断线，所以需要读取相应状态位和中断允许位来决定中断原因。

47.3.7 DMA

主机模式使用 DMA 写：

- 读取 CONFIG 寄存器 (0x7c)，获得 RX/TX FIFO 的深度；
- 读取 SPI 的状态寄存器 (0x34)，等待 SPI 空闲；
- 配置 TRANSFMT 寄存器 (0x10)，配置 addrlen，datalen，datamerge，其他值复位值即可；
- 配置 TRANSCTRL 寄存器 (0x20)，配置 cmden=1，addren=1，transmode=1(仅写)，wrTranCnt，其他值复位值即可；
- 配置 CTRL 寄存器 (0x30)，使能 DMA，复位 TX FIFO，根据 FIFO 配置设置触发阈值，eg: 一半 TX FIFO 深度；
- 配置 INTREN 寄存器 (0x38)，使能 EndIntEn 中断；
- 配置 TIMING 寄存器 (0x40)，配置接口时许；
- 使用 DMA 从其他地方搬数到 DATA 寄存器 (0x2C)；
- 配置 ADDR 寄存器 (0x28)；
- 配置 CMD 寄存器 (0x24), 触发 SPI 的传输；命令取决于 SPI 设备；
- 等待 INTRST 寄存器 (0x3C) 中断位；

- 配置 INTRST 寄存器 (0x3C) 清除中断；

主机模式使用 DMA 读：

- 读取 CONFIG 寄存器 (0x7c)，获得 RX/TX FIFO 的深度；
- 读取 SPI 的状态寄存器 (0x34)，等待 SPI 空闲；
- 配置 TRANSFMT 寄存器 (0x10)，配置 addrlen，datalen，datamerge，其他值复位值即可；
- 配置 TRANSCTRL 寄存器 (0x20)，配置 cmden=1，addren=1，transmode=2(仅读)，RdTranCnt，其他值复位值即可；
- 配置 CTRL 寄存器 (0x30)，使能 DMA，复位 RX FIFO，根据 FIFO 配置设置触发阈值，eg: 一半 RX FIFO 深度；
- 配置 INTREN 寄存器 (0x38)，使能 EndIntEn 中断；
- 配置 TIMING 寄存器 (0x40)，配置接口时许；
- 使用 DMA 从 DATA 寄存器 (0x2C) 搬数到其他地方；
- 配置 ADDR 寄存器 (0x28)；
- 配置 CMD 寄存器 (0x24), 触发 SPI 的传输；命令取决于 SPI 设备；
- 等待 INTRST 寄存器 (0x3C) 中断位；
- 配置 INTRST 寄存器 (0x3C) 清除中断；

47.4 SPI 寄存器列表

SPI0 base address: 0xF0070000

SPI1 base address: 0xF0074000

SPI2 base address: 0xF0078000

SPI3 base address: 0xF007C000

地址偏移	名称	描述	复位值
0x0004	WR_TRANS_CNT	写入数据长度	0x00000000
0x0008	RD_TRANS_CNT	读取数据长度	0x00000000
0x0010	TRANSFMT	数据格式	0x00020780
0x0014	DIRECTIO	直接管脚控制	0x00003100
0x0020	TRANSCTRL	传输控制	0x00000000
0x0024	CMD	命令寄存器	0x00000000
0x0028	ADDR	地址寄存器	0x00000000
0x002C	DATA	数据寄存器	0x00000000
0x0030	CTRL	控制寄存器	0x00000000
0x0034	STATUS	状态寄存器	0x00000000
0x0038	INTREN	中断使能寄存器	0x00000000
0x003C	INTRST	中断状态	0x00000000
0x0040	TIMING	接口时序	0x00000000
0x0060	SLVST	从机状态	0x00000000
0x0064	SLVDATACNT	从机数据计数	0x00000000
0x0068	SLVDATAWCNT		0x00000000
0x006C	SLVDATARCNT		0x00000000

地址偏移	名称	描述	复位值
0x007C	CONFIG	配置寄存器	0x00004311

表 205: SPI 寄存器列表

47.5 SPI 寄存器描述

47.5.1 WR_TRANS_CNT (0x4)

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

WR_TRANS_CNT [31:0]

位域	名称	描述
31-0	WRTRANCNT	写入数据长度。 WrTranCnt 表示 SPI 写入数据长度。实际传输长度为（WrTranCnt+1）。 WrTranCnt 仅在 TransMode 为 0、1、3、4、5、6 或 8 时生效。 数据单位的大小（位宽度）由传输格式寄存器的 DataLen 字段定义。 对于传输模式 0，WrTranCnt 必须等于 RdTranCnt。

WR_TRANS_CNT 位域

47.5.2 RD_TRANS_CNT (0x8)

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

RD_TRANS_CNT [31:0]

位域	名称	描述
31-0	RDTRANCNT	读取数据长度，RdTranCnt 表示从 SPI 总线接收的数据长度。实际接收长度为（RdTranCnt+1）。 RdTransCnt 仅在 TransMode 为 0、2、3、4、5、6 或 9 时生效。 数据单位由传输格式寄存器的 DataLen 字段定义。 对于传输模式 0，WrTranCnt 必须等于 RdTranCnt。

RD_TRANS_CNT 位域

47.5.3 TRANSFMT (0x10)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD														ADDRLEN		RSVD			DATALEN					DATAMERGE	RSVD		MOSIBIDIR	LSB	SLVMODE	CPOL	CPHA
N/A														RW		N/A			RW					RW	N/A		RW	RW	RW	RW	RW
0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	x	x	x	0	0	1	1	1	1	x	x	0	0	0	0	0

TRANSFMT [31:0]

位域	名称	描述
17-16	ADDRLEN	地址长度： 0x0:8 位， 0x1:16 位， 0x2:24 位， 0x3:32 位。
12-8	DATALEN	数据单位长度 实际位数为（DataLen+1）
7	DATAMERGE	允许数据合并模式，该模式在写入时自动拆分数据，在读取时合并数据。 此位仅在 DataLen=0x7 时生效。在数据合并模式下，对数据寄存器的每次写入将传输写入数据的所有四个字节；从数据寄存器读取的每个数据将检索四个字节的接收数据，作为单个字数据。 当禁用数据合并模式时，只有数据寄存器的最小（DataLen+1）有效位对读/写操作有效；不会执行自动数据拆分/合并。
4	MOSIBIDIR	常规模式下的双向 MOSI： 0x0:MOSI 是单向信号， 0x1:MOSI 是双向信号。
3	LSB	传输顺序： 0x0：高位先行， 0x1：低位先行。

位域	名称	描述
2	SLVMODE	主/从模式选择： 0x0：主模式， 0x1：从模式。
1	CPOL	SPI 时钟极性： 0x0: 空闲态为低电平， 0x1: 空闲态为高电平。
0	CPHA	SPI 时钟相位： 0x0：奇数 SCLK 边沿采样， 0x1：偶数 SCLK 边沿采样。

TRANSFMT 位域

47.5.4 DIRECTIO (0x14)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD							DIRECTIOEN	RSVD		HOLD_OE	WP_OE	MISO_OE	MOSI_OE	SCLK_OE	CS_OE	RSVD		HOLD_O	WP_O	MISO_O	MOSI_O	SCLK_O	CS_O	RSVD		HOLD_I	WP_I	MISO_I	MOSI_I	SCLK_I	CS_I
N/A							RW	N/A		RW	RW	RW	RW	RW	RW	N/A		RW	RW	RW	RW	RW	RW	N/A		RO	RO	RO	RO	RO	RO
x	x	x	x	x	x	x	0	x	x	0	0	0	0	0	0	x	x	1	1	0	0	0	1	x	x	0	0	0	0	0	0

DIRECTIO [31:0]

位域	名称	描述
24	DIRECTIOEN	启用直接引脚控制： 0x0：禁用， 0x1：启用。
21	HOLD_OE	允许 HOLD 引脚的输出。
20	WP_OE	允许 WP 引脚的输出。
19	MISO_OE	允许 MISO 引脚的输出。
18	MOSI_OE	允许 MOSI 引脚的输出。
17	SCLK_OE	允许 SCLK 引脚的输出。
16	CS_OE	允许 CS 引脚的输出。
13	HOLD_O	HOLD 引脚的输出值。
12	WP_O	WP 引脚的输出值。
11	MISO_O	SPI MISO 引脚输出值。
10	MOSI_O	SPI MOSI 引脚输出值。
9	SCLK_O	SCLK 引脚输出值。
8	CS_O	CS 引脚输出值。
5	HOLD_I	HOLD 引脚的状态。
4	WP_I	WP 引脚状态。
3	MISO_I	MISO 引脚状态。

位域	名称	描述
2	MOSI_I	MOSI 引脚状态。
1	SCLK_I	SCLK 引脚状态。
0	CS_I	CS 引脚状态。

DIRECTIO 位域

47.5.5 TRANSCTRL (0x20)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SLVDATAONLY	CMDEN	ADDREN	ADDRFMT	TRANSMODE				DUALQUAD		TOKENEN	WRTRANCNT									TOKENVALUE	DUMMYCNT		RDTRANCNT								
RW	RW	RW	RW	RW				RW		RW	RW									RW	RW		RW								
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

TRANSCTRL [31:0]

位域	名称	描述
31	SLVDATAONLY	纯数据模式（仅从模式）： 0x0：禁用纯数据模式， 0x1：启用纯数据模式。 注：此模式仅在常规模式下有效，因此 MOSIBiDir、DualQuad 和 TransMode 应设置为 0。
30	CMDEN	启用命令段（仅主模式）： 0x0：禁用命令段， 0x1: 启用命令段。
29	ADDREN	启用地址段（仅主模式）： 0x0：禁用地址段， 0x1: 启用地址段。
28	ADDRFMT	SPI 地址段格式（仅主模式）： 0x0：常规（1 位）模式， 0x1：数据段（2 线/4 线）相同。

位域	名称	描述
27-24	TRANSMODE	传输模式，传输顺序可以是 0x0：同时读写， 0x1：仅写， 0x2：只读， 0x3：写，读， 0x4：读、写， 0x5：写、填充、读， 0x6：读、填充、写， 0x7：无数据（必须在主模式下启用 CmdEn 或 AddrEn）， 0x8：填充，写， 0x9：填充，读， 0xa~0xf: 保留。
23-22	DUALQUAD	SPI 数据段格式： 0x0：1 线模式， 0x1：2 线模式， 0x2：4 线模式， 0x3：保留。
21	TOKENEN	启用令牌（仅主模式），在 SPI 读取地址段后的 1 字节的令牌。应在 TokenValue 中选择特殊令牌的值。 0x0：禁用令牌， 0x1：启用令牌。
20-12	WRTRANCNT	写入数据长度。 WrTranCnt 表示 SPI 写入数据长度。实际传输长度为（WrTranCnt+1）。 WrTranCnt 仅在 TransMode 为 0、1、3、4、5、6 或 8 时生效。数据单位的大小（位宽度）由传输格式寄存器的 DataLen 字段定义。 对于传输模式 0，WrTranCnt 必须等于 RdTranCnt。
11	TOKENVALUE	令牌值（仅主模式），SPI 读取地址段后的令牌。 0x0：令牌值 =0x00， 0x1：令牌值 =0x69。
10-9	DUMMYCNT	填充数据长度。实际填充长度为（DummyCnt+1）。 SPI 接口上的填充周期数为（DummyCnt+1）*（（DataLen+1）/SPI IO 宽度），在填充数据段，数据引脚被置于高阻抗中。 DummyCnt 仅用于 TransMode 5、6、8 和 9，其具有填充数据段。
8-0	RDTRANCNT	读取数据长度，RdTranCnt 表示从 SPI 总线接收的数据长度。实际接收长度为（RdTranCnt+1）。 RdTransCnt 仅在 TransMode 为 0、2、3、4、5、6 或 9 时生效。数据单位由传输格式寄存器的 DataLen 字段定义。 对于传输模式 0，WrTranCnt 必须等于 RdTranCnt。

TRANSCTRL 位域

47.5.6 CMD (0x24)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																								CMD							
N/A																								RW							
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0

CMD [31:0]

位域	名称	描述
7-0	CMD	SPI 命令。

CMD 位域

47.5.7 ADDR (0x28)

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

ADDR [31:0]

位域	名称	描述
31-0	ADDR	SPI 地址（仅主模式）。

ADDR 位域

47.5.8 DATA (0x2C)

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

DATA [31:0]

位域	名称	描述
31-0	DATA	对于写操作，数据传输到 TX FIFO。低字节先行。如果 TX FIFO 满且状态寄存器的 SPIActive 位为 1，总线被阻塞并插入等待状态。 对于读操作，数据从 RX FIFO 读取。低字节先行。如果 RX FIFO 空且状态寄存器的 SPIActive 位为 1，总线被阻塞并插入等待状态。 FIFO 将 SPI 传输和软件的速度的缓冲。当 TX FIFO 为空时，SPI 传输将暂停，直到更多数据写入 TX FIFO；当 RX FIFO 满时，SPI 传输将暂停，直到 RX FIFO 中有更多空间。 如果写入 TX FIFO 的数据多于写入传输数量（WrTranCnt），剩余数据保留在 TX FIFO 中，下次传输或 TX FIFO 复位清除。

DATA 位域

47.5.9 CTRL (0x30)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD				CS_EN				TXTHRES								RXTHRES								RSVD			TXDMAEN	RXDMAEN	TXFIFORST	RXFIFORST	SPIRST
N/A				RW				RW								RW								N/A			RW	RW	RW	RW	RW
x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	x	x	x	0	0	0	0	0

CTRL [31:0]

位域	名称	描述
27-24	CS_EN	
23-16	TXTHRES	TXFIFO 阈值，TX 数据小于或等于 TX FIFO 阈值时，发出 TXFIFO 中断或 DMA 请求。
15-8	RXTHRES	RXFIFO 阈值，RX 数据大于或等于 RX FIFO 阈值时，发出 RXFIFO 中断或 DMA 请求。
4	TXDMAEN	TX DMA 启用。
3	RXDMAEN	RX DMA 启用。
2	TXFIFORST	发送 FIFO 复位，写 1 复位，复位后自动清零。
1	RXFIFORST	接收 FIFO 复位，写 1 复位，复位后自动清零。
0	SPIRST	SPI 复位，写 1 复位，复位后自动清零。

CTRL 位域

47.5.10 STATUS (0x34)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD		TXNUM_7_6		RSVD		RXNUM_7_6		TXFULL	TXEMPTY	TXNUM_5_0						RXFULL	RXEMPTY	RXNUM_5_0						RSVD							SPIACTIVE
N/A		RO		N/A		RO		RO	RO	RO						RO	RO	RO						N/A							RO
x	x	0	0	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	x	x	x	x	x	x	x	0

STATUS [31:0]

位域	名称	描述
29-28	TXNUM_7_6	发送 FIFO 中的有效数据。
25-24	RXNUM_7_6	接收 FIFO 中的有效数据。
23	TXFULL	发送 FIFO 满标志。
22	TXEMPTY	发送 FIFO 空标志。
21-16	TXNUM_5_0	发送 FIFO 中的有效数据。
15	RXFULL	接收 FIFO 满标志。
14	RXEMPTY	接收 FIFO 空标志。
13-8	RXNUM_5_0	接收 FIFO 中的有效数据。
0	SPIACTIVE	SPI 寄存器编程正在进行中。 在主模式下，SPIActive 在写入 SPI 命令寄存器后变为 1，在传输完成后变为 0。 在从属模式下，SPI CS 信号被断言后，SPIActive 变为 1；SPI CS 信号被解除断言后，SPIActive 变为 0。 注意，由于时钟同步，当相应条件发生时，SPIActive 最多可能需要两个 spi_ 时钟周期才能改变。 注意：当使用直接 IO 控制或内存映射接口时，该位保持为 0。

STATUS 位域

47.5.11 INTREN (0x38)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																										SLVCMDEN	ENDINTEN	TXFIFOINTEN	RXFIFOINTEN	TXFIFOURINTEN	RXFIFOORINTEN
N/A																										RW	RW	RW	RW	RW	RW
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0

INTREN [31:0]

位域	名称	描述
5	SLVCMDEN	启用从机命令中断，在从模式下收到属命令时产生中断（仅从模式）。
4	ENDINTEN	启用 SPI 传输结束中断。SPI 传输结束时是否产生中断（在从模式下，读取结束不产生中断）。
3	TXFIFOINTEN	启用发送 FIFO 阈值中断。有效数据小于或等于 TX FIFO 阈值时是否产生中断。
2	RXFIFOINTEN	启用接收 FIFO 阈值中断。有效数据大于或等于 RX FIFO 阈值时是否产生中断。
1	TXFIFOURINTEN	启用发送 FIFO 欠载中断。发送 FIFO 数据耗尽时是否产生中断（仅从模式）。
0	RXFIFOORINTEN	启用接收 FIFO 溢出中断。接收 FIFO 溢出时是否产生中断（仅从模式）。

INTREN 位域

47.5.12 INTRST (0x3C)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																										SLVCMDINT	ENDINT	TXFIFOINT	RXFIFOINT	TXFIFOURINT	RXFIFOORINT
N/A																										W1C	W1C	W1C	W1C	W1C	W1C
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0

INTRST [31:0]

位域	名称	描述
5	SLVCMDINT	从机命令中断（仅从模式）。
4	ENDINT	SPI 传输中断结束。
3	TXFIFOINT	TX FIFO 阈值中断。
2	RXFIFOINT	RX FIFO 阈值中断。
1	TXFIFOURINT	TX FIFO 欠载中断（仅从模式）。
0	RXFIFOORINT	RX FIFO 溢出中断（仅从模式）。

INTRST 位域

47.5.13 TIMING (0x40)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																		CS2SCLK		CSHT				SCLK_DIV							

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
N/A																		RW		RW				RW							
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0

TIMING [31:0]

位域	名称	描述
13-12	CS2SCLK	CS 有效到 SCLK 边缘最短时间。SCLK_ 周期 *（CS2SCLK+1）/2。
11-8	CSHT	CS 高电平的最短时间。SCLK_ 周期 *（CSHT+1）/2。
7-0	SCLK_DIV	时钟源和 SPI 接口 SCLK 之间的时钟频率比。F(SCLK)=F(SPI)/((SCLK_DIV+1)*2)。 当 SCLK_DIV 是 0xff 时，SCLK 频率应与 SPI 时钟源同频。

TIMING 位域

47.5.14 SLVST (0x60)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD													UNDERRUN	OVERRUN	READY	USR_STATUS															
N/A													W1C	RW	RW	RW															
x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

SLVST [31:0]

位域	名称	描述
18	UNDERRUN	上次传输中发生数据不足。
17	OVERRUN	上次传输中发生数据溢出。
16	READY	SPI 准备就绪。除从机模式下读取状态，该位都会在传输结束后清零。
15-0	USR_STATUS	自定义的状态标志。

SLVST 位域

47.5.15 SLVDATACNT (0x64)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD						WCNT										RSVD						RCNT									
N/A						RO										N/A						RO									
x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0

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

SLVDATACNT [31:0]

位域	名称	描述
25-16	WCNT	从机发送数据计数。
9-0	RCNT	从机接收数据计数。

SLVDATACNT 位域

47.5.16 SLVDATAWCNT (0x68)

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

SLVDATAWCNT [31:0]

位域	名称	描述
31-0	VAL	

SLVDATAWCNT 位域

47.5.17 SLVDATARCNT (0x6C)

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

SLVDATARCNT [31:0]

位域	名称	描述
31-0	VAL	

SLVDATARCNT 位域

47.5.18 CONFIG (0x7C)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																	SLAVE	RSVD				QUADSPI	DUALSPI	TXFIFOSIZE				RXFIFOSIZE			
N/A																	RO	N/A				RO	RO	RO				RO			
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	1	x	x	x	x	1	1	0	0	0	1	0	0	0	1

CONFIG [31:0]

位域	名称	描述
14	SLAVE	支持从机模式。
9	QUADSPI	支持 4 线模式。
8	DUALSPI	支持 2 线模式。
7-4	TXFIFOSIZE	发送 FIFO 的深度： 0x0:2 个字， 0x1:4 个字， 0x2:8 个字， 0x3:16 个字， 0x4:32 个字， 0x5:64 个字， 0x6:128 字。
3-0	RXFIFOSIZE	接收 FIFO 的深度： 0x0:2 个字， 0x1:4 个字， 0x2:8 个字， 0x3:16 个字， 0x4:32 个字， 0x5:64 个字， 0x6:128 字。

CONFIG 位域

48 集成电路总线 I2C

本章节介绍 I2C 的功能和特性。

48.1 特性总结

本章节介绍 I2C 的主要特性：

- 支持标准模式 (100Kb/s)，快速模式 (400Kb/s) 和快速模式 +(1Mb/s)
- 可配置主从模式
- 支持 7 位和 10 位地址模式
- 支持广播呼叫地址 (general call address)
- 自动时钟延展 (clock stretching)
- 可配置的时钟/数据时序
- 支持直接内存访问 (DMA)
- 4 字节 FIFO

I2C 的框图如图 59。

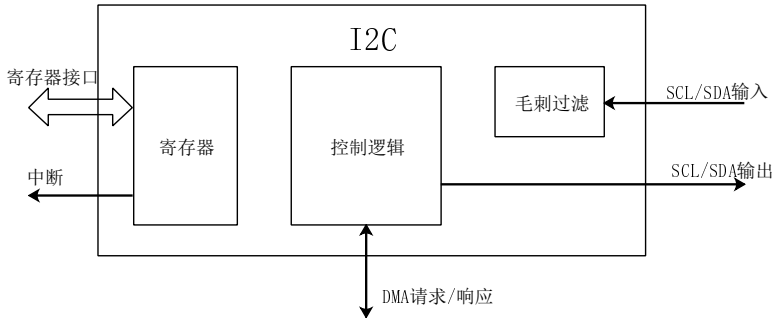


图 59: I2C 框图

48.2 功能描述

本章节描述 I2C 的功能。

48.2.1 主要功能

通过 SETUP 寄存器的 MASTER 位可配置 I2C 工作在主机模式或者从机模式。作为 I2C 主机，该控制器能够高效的发起传输。每个传输由 4 个阶段组成：起始，地址，数据和结束。在起始阶段会产生 START 操作，在地址阶段发送地址，在数据阶段 1 个或多个数据字节被传送，在结束阶段产生 STOP 操作。每个阶段都能够独立控制是否执行。

作为 I2C 从机，当 I2C 传输的地址与寄存器 ADDR 的地址匹配时，该控制器被选定，可配置 INTEN 寄存器的 ADDRHIT 位使控制器产生中断来通知软件准备后续的操作。如果收到广播呼叫地址，控制器会响应 ACK 并将 STATUS 寄存器的 GENCALL 位置 1。

当软件没有准备好下一个字节的发送数据或者接收数据时 FIFO 已满，控制会自动延展 I2C 总线时钟来暂停总线传输。主机模式和从模式都支持自动时钟延展。

控制器默认使能了自动响应，即除了最后一个字节外每接收一个字节数据都会自动发出 ACK，软件可通过使能 ByteReceive 中断来禁止自动响应功能，在每个字节接收完毕后决定是否发送 ACK 响应。

48.2.2 时序配置

I2C 时序参数在 SETUP 寄存器中配置，所有参数的单位都是 I2C 模块功能时钟 (CLK_TOP_I2Cx) 的时钟周期数。由于这些可配参数的位数有限，当功能时钟频率非常高时，时序参数的位宽可能无法满足配置要求，因此控制器提供了 TPM 寄存器用来设置时序参数乘数，该乘数能够扩展时序参数的范围。TPM 寄存器的修改应该在 I2C 总线空闲且 I2C 控制器关闭 (寄存器 SETUP 的 IICEN 位为 0) 时进行。

以下以 CLK_TOP_I2C 频率 40MHz 且 TPM 等于 0 为例来说明时序参数配置。

毛刺过滤：在 SCL 和 SDA 输入信号上可被过滤掉的脉冲宽度，由 SETUP 寄存器的 T_SP 位定义，计算公式如 (3):

$$PulseWidth = T\_SP \times Clk\_Period \times (TPM + 1) \tag{3}$$

对于快速模式和快速模式 +，要求 50ns 以下的毛刺必须被过滤掉，计算可得 T_SP 应为 2.

数据建立时间 (Data Setup Time)：数据建立时间定义了在 SCL 上升沿之前 SDA 应该保持稳定的时间，由 SETUP 寄存器的 T_SUDAT 位设置，计算公式如 (4):

$$SetupTime = (2 \times Clk\_Period) + (2 + T\_SP + T\_SUDAT) \times Clk\_Period \times (TPM + 1) \tag{4}$$

对于标准模式，数据建立时间要求为 250ns 时，可得 T_SUDAT 应设为 4。

数据保持时间 (Data Hold Time)：数据保持时间定义了 SCL 下降沿之后应保持稳定的时间，由 SETUP 寄存器的 T_HDDAT 位设置，计算公式如 (5):

$$HoldTime = (2 \times Clk\_Period) + (2 + T\_SP + T\_HDDAT) \times Clk\_Period \times (TPM + 1) \tag{5}$$

对于标准模式，数据保持时间要求为 300ns 时，可得 T_HDDAT 应设为 6。

总线时钟频率：I2C 总线时钟频率由 SETUP 寄存器的 T_SCLHI 位和 T_SCLRATIO 位定义，计算公式如 (6)和(7)

$$SCL\ HighPeriod = (2 \times Clk\_Period) + (2 + T\_SP + T\_SCLHI) \times Clk\_Period \times (TPM + 1) \tag{6}$$

$$SCL\ LowPeriod = (2 \times Clk\_Period) + (2 + T\_SP + T\_SCLHI \times T_S CLRATIO) \times Clk\_Period \times (TPM + 1) \tag{7}$$

48.2.3 主机模式

不使用 DMA 的数据发送：

1. 设置时序参数
2. 配置主机模式 MASTER=1 并使能控制器 IICEN=1
3. 设置数据长度 DATACNT，传输方向 DIR 和传输阶段选择 PHASE_START/ADDR/DATA/STOP
4. 配置目标从机地址 ADDRESS
5. 设置 CMPL 使传输完成时产生中断，设置 FIFOEMPTY 使 FIFO 空时产生中断
6. 写 0x1 到 COMMAND 寄存器来启动传输
7. 等待中断：若 FIFO 为空则通过写 DATA 寄存器来存入数据，若所有数据已存入 FIFO 则关闭 FIFO 空中断，否则重复步骤 7；若传输完成则检查 ADDRHIT 确保目标从机正确接收了传输，CMPL 寄存器写 1 清除中断，跳转到步骤 8。
8. 关闭所有中断，检查 DATACNT 以确保所有数据传输完成

不使用 DMA 的数据接收：

1. 设置时序参数

2. 配置主机模式 MASTER=1 并使能控制器 IICEN=1
3. 设置数据长度 DATACNT，传输方向 DIR 和传输阶段选择 PHASE_START/ADDR/DATA/STOP
4. 配置目标从机地址 ADDRESS
5. 设置 CMPL 使传输完成时产生中断，设置 FIFOFULL 使 FIFO 满时产生中断
6. 写 0x1 到 COMMAND 寄存器来启动传输
7. 等待中断：若 FIFO 为满则通过从 DATA 寄存器来读出数据，若所有数据已存入 FIFO 则关闭 FIFO 空中断，否则重复步骤 7；若传输完成则检查 ADDRHIT 确保目标从机正确接收了传输，CMPL 寄存器写 1 清除中断，跳转到步骤 8。
8. 关闭所有中断，检查 DATACNT 以确保所有数据传输完成

使用 DMA 的数据发送：

1. 设置时序参数
2. 配置主机模式 MASTER=1，使能控制器 IICEN=1，使能 DMA 功能 DMAEn=1
3. 配置 DMA 控制器 IP
4. 设置数据长度 DATACNT，传输方向 DIR 和传输阶段选择 PHASE_START/ADDR/DATA/STOP
5. 配置目标从机地址 ADDRESS
6. 设置 CMPL 使传输完成时产生中断
7. 写 0x1 到 COMMAND 寄存器来启动传输
8. 等待中断，检查 DATACNT 以确保所有数据传输完成

48.2.4 从机模式

不使用 DMA 的数据传输：

1. 设置从机地址
2. 设置时序参数
3. 配置从机模式 MASTER=0 并使能控制器 IICEN=1
4. 使能地址命中中断 ADDRHIT 和传输完成中断 CMPL
5. 等待地址命中中断：读取传输方向 DIR，查看 GENCALL 确认是否是广播呼叫。若 DIR 方向为从机接收，则使能 FIFO 满中断并跳转到步骤 6；若 DIR 方向为从机发送，则使能 FIFO 空中断并跳转到步骤 7
6. 从机接收：等待 FIFO 满中断，读取数据，直到收到传输完成中断
7. 从机发送：等待 FIFO 空中断，写入数据，直到收到传输完成中断
8. 检查 DATACNT 寄存器确认传输数据量，清除传输完成中断

使用 DMA 的数据传输：

1. 设置从机地址
2. 设置时序参数
3. 配置从机模式 MASTER=0 并使能控制器 IICEN=1
4. 使能地址命中中断 ADDRHIT 和传输完成中断 CMPL
5. 等待地址命中中断：读取传输方向 DIR，如果不是期望的传输方向，则跳转到不使用 DMA 的数据传输流程。查看 GENCALL 确认是否是广播呼叫
6. 从机接收：等待 FIFO 满中断，读取数据，直到收到传输完成中断
7. 从机发送：等待 FIFO 空中断，写入数据，直到收到传输完成中断
8. 检查 DATACNT 寄存器确认传输数据量，清除传输完成中断

48.3 I2C 寄存器

48.3.1 寄存器说明

I2C 的寄存器列表如下：

I2C0 base address: 0xF0060000

I2C1 base address: 0xF0064000

I2C2 base address: 0xF0068000

I2C3 base address: 0xF006C000

地址偏移	名称	描述	复位值
0x0010	CFG	配置查询寄存器	0x00000001
0x0014	INTEN	中断使能寄存器	0x00000000
0x0018	STATUS	状态寄存器	0x00000001
0x001C	ADDR		0x00000000
0x0020	DATA	数据寄存器	0x00000000
0x0024	CTRL	控制寄存器	0x00905E00
0x0028	CMD	指令寄存器	0x00000000
0x002C	SETUP	设置寄存器	0x05252100
0x0030	TPM	时序参数乘数	0x00000000

表 206: I2C 寄存器列表

48.3.2 寄存器详细信息

I2C 的寄存器详细说明如下：

48.3.3 CFG (0x10)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																														FIFOSIZE	
N/A																														RO	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

CFG [31:0]

位域	名称	描述
1-0	FIFOSIZE	FIFO 深度： 0：2 字节 1：4 字节 2：8 字节 3：16 字节

位域	名称	描述

CFG 位域

48.3.4 INTEN (0x14)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																						CMPL	BYTERECV	BYTETRANS	START	STOP	ARBLOSE	ADDRHIT	FIFOHALF	FIFOFULL	FIFOEMPTY
N/A																						RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

INTEN [31:0]

位域	名称	描述
9	CMPL	传输完成中断使能： 主机模式：一个 transaction 被正确发出且没有失去总线仲裁 从机模式：发送给本控制器的一个 transaction 传输完成
8	BYTERECV	字节接收中断使能： 当收到一个字节的数据后发出中断，该位设为 1 时会关闭自动 ACK 响应，软件需要控制 ACK 或 NACK 的发出。
7	BYTETRANS	字节发送中断使能： 当完成一个字节的发送后产生中断。
6	START	START 中断使能： 当探测到 START 或 repeated START 时产生中断。
5	STOP	STOP 中断使能： 当探测到 STOP 时产生中断。
4	ARBLOSE	总线仲裁丢失中断使能： 主机模式：当控制器失去总线仲裁时产生中断 从机模式：无效
3	ADDRHIT	地址命中中断使能： 主机模式：收到目标从机的 ACK 响应时产生中断 从机模式：地址匹配时产生中断
2	FIFOHALF	FIFO 半满/半空中断： 工作在数据接收状态时，当 FIFO 已被填充一半时产生中断。 工作在数据发送状态时，当 FIFO 空出一半空间时产生中断。 数据的发送和接收状态由传输的方向决定。
1	FIFOFULL	FIFO 满中断使能
0	FIFOEMPTY	FIFO 空中断使能

INTEN 位域

48.3.5 STATUS (0x18)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																	LINESDA	LINESCL	GENCALL	BUSBUSY	ACK	CMPL	BYTERECV	BYTETRANS	START	STOP	ARBLOSE	ADDRHIT	FIFOHALF	FIFOFULL	FIFOEMPTY
N/A																	RO	RO	RO	RO	RO	W1C	W1C	W1C	W1C	W1C	W1C	W1C	RO	RO	RO
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

STATUS [31:0]

位域	名称	描述
14	LINESDA	SDA 信号状态： 1：高电平 0：低电平
13	LINESCL	SCL 信号状态： 1：高电平 0：低电平
12	GENCALL	广播呼叫标志： 1：当前 transaction 为广播呼叫 0：非广播呼叫
11	BUSBUSY	总线繁忙标志： 总线上出现了 START 但尚未出现 STOP 时为繁忙状态，该位置 1。
10	ACK	记录上一次传输的 ACK 状态： 1：ACK 0：NACK
9	CMPL	传输完成： 主机模式：一个 transaction 被正确发出且没有失去总线仲裁 从机模式：发送给本控制器的一个 transaction 传输完成 该状态位必须被清除，否则下一个 transaction 会被阻塞。
8	BYTERECV	一个字节数据已被接收
7	BYTETRANS	一个字节数据已被发送
6	START	START 或 repeated START 已被发送或接收
5	STOP	STOP 已被发送或接收
4	ARBLOSE	控制器已失去总线仲裁
3	ADDRHIT	主机模式：表示从机已发出响应 从机模式：表示 transaction 以本控制器为目标从机（包括广播呼叫）
2	FIFOHALF	数据发送状态下 FIFO 半空
1	FIFOFULL	FIFO 满
0	FIFOEMPTY	FIFO 空

STATUS 位域

48.3.6 ADDR (0x1C)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																						ADDR									
N/A																						RW									
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ADDR [31:0]

位域	名称	描述
9-0	ADDR	从机地址： 对于 7 位地址模式，该寄存器的低 7 位有效

ADDR 位域

48.3.7 DATA (0x20)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																								DATA							
N/A																								RW							
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

DATA [31:0]

位域	名称	描述
7-0	DATA	对该寄存器执行写操作会对 FIFO 进行 push，对该寄存器的读操作会对 FIFO 进行 pop

DATA 位域

48.3.8 CTRL (0x24)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATACNT_HIGH								RESET_LEN				RSVD					RESET_HOLD_SCKIN	RESET_ON	PHASE_START	PHASE_ADDR	PHASE_DATA	PHASE_STOP	DIR	DATACNT							
RW								RW				N/A					RW	RW	RW	RW	RW	RW	RW	RW							

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

CTRL [31:0]

位域	名称	描述
31-24	DATACNT_HIGH	byte 数量： 主机模式下：需要发送或接收的字节数，0 表示最大字节数。每次 byte 传输后该寄存器值减 1。 从机模式下：该寄存器的功能与 DMA 模式相关： 如果 DMA 关闭，该寄存器显示已向主机发送或从主机接收的字节数，当地址匹配时清零，每字节的数据传输后加 1。 如果 DMA 使能，该寄存器表示需要向主机发送或从主机接收的字节数，它不会清零且每字节的数据传输后减 1。
23-20	RESET_LEN	复位时钟个数，默认为 9。时钟高电平和低电平宽度由控制寄存器中 T_SCLHi 决定，50% 占空比
14	RESET_HOLD_SCKIN	设 1，会在复位期间将输入内部的 SCL 拉高，以免影响内部逻辑。
13	RESET_ON	设 1 开始复位，会在 SCL 上发送若干个时钟信号，时钟个数由 reset_len 寄存器决定。 复位完成后硬件自动清除此寄存器，软件不能通过写零清零。
12	PHASE_START	发送 START 使能，仅主机模式。
11	PHASE_ADDR	发送地址使能，仅主机模式。
10	PHASE_DATA	发送数据使能，仅主机模式。
9	PHASE_STOP	发送 STOP 使能，仅主机模式。
8	DIR	传输方向： 主机模式下： 0：发送 1：接收 从机模式下： 0：接收 1：发送
7-0	DATACNT	byte 数量： 主机模式下：需要发送或接收的字节数，0 表示最大字节数。每次 byte 传输后该寄存器值减 1。 从机模式下：该寄存器的功能与 DMA 模式相关： 如果 DMA 关闭，该寄存器显示已向主机发送或从主机接收的字节数，当地址匹配时清零，每字节的数据传输后加 1。 如果 DMA 使能，该寄存器表示需要向主机发送或从主机接收的字节数，它不会清零且每字节的数据传输后减 1。

CTRL 位域

48.3.9 CMD (0x28)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																													CMD		
N/A																													RW		
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

CMD [31:0]

位域	名称	描述
2-0	CMD	定义需要执行的操作： 0x0：无操作 0x1：发送一个数据 transaction，仅主机模式 0x2：发送一个 ACK 响应 0x3：发送一个 NACK 响应 0x4：清除 FIFO 0x5：复位该控制器，包括中断使能寄存器 当通过向该寄存器写 1 来发送 transaction 时，寄存器保持 0x1 的值直到传输完成。

CMD 位域

48.3.10 SETUP (0x2C)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD			T_SUDAT					T_SP			T_HDDAT					RSVD		T_SCLRADIO	T_SCLHI									DMAEN	MASTER	ADDRESSING	IICEN
N/A			RW					RW			RW					N/A		RW	RW									RW	RW	RW	RW
0	0	0	0	0	1	0	1	0	0	1	0	0	1	0	1	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	0

SETUP [31:0]

位域	名称	描述
28-24	T_SUDAT	定义 SCL 上升沿之前 SDA 的建立时间
23-21	T_SP	定义毛刺过滤的脉冲宽度
20-16	T_HDDAT	定义 SCL 下降沿之后 SDA 的保持时间
13	T_SCLRADIO	定义 SCL 占空比： 0：50% 占空比 1：低电平的持续时间约为高电平的 2 倍 仅主机模式有效。
12-4	T_SCLHI	定义 SCL 高电平时间，仅主机模式有效

位域	名称	描述
3	DMAEN	DMA 使能
2	MASTER	配置主从模式： 1：主机模式 0：从机模式
1	ADDRESSING	I2C 地址模式： 1：10 位地址模式 0：7 位地址模式
0	IICEN	使能 I2C 控制器

SETUP 位域

48.3.11 TPM (0x30)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																											TPM					
N/A																											RW					
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

TPM [31:0]

位域	名称	描述
4-0	TPM	时序参数乘数

TPM 位域

49 控制器局域网 MCAN

本章节介绍控制器局域网 MCAN 的功能和特性。

49.1 概述

控制器局域网 MCAN 的特性如下：

- 支持 CAN 2.0B 协议，支持多达 8 字节的数据载荷, 数据速率可达 1Mbit/s;
- 支持 CAN FD 协议，支持多达 64 字节的数据载荷, 数据速率可达 8Mbit/s;
- 支持 1~1/16 的波特率预分频，配合系统 CAN 时钟分频器，可灵活配置波特率 ;
- 多个 MCAN 模块共享 HRAM（AHB SRAM，32KB）作为消息存储器
  - CPU 可以通过 AHB 访问 HRAM，不同 MCAN 模块通过配置消息 RAM 为不同地址（低 16 位地址即可），共享同一块 HRAM
  - 可灵活配置成不同的发送，接收缓冲器，过滤器，事件缓冲器。
- 可编程 ID CODE 位以及 MASK 位 ;
- PTB/STB 均支持支持单次发送模式 ;
- 支持静默模式 ;
- 支持回环模式 ;
- 支持待机模式 ;
- 支持捕捉传输的错误种类以及定位仲裁失败位置 ;
- 可编程的错误警告值 ;
- 符合 ISO 11898-1:2015
- 支持 AUTOSAR
- 支持 SAE J1939
- 支持调试
- 支持 DMA
- 根据 CiA 603 支持硬件时间戳
  - 16 组 32 位时间戳，或 8 组 64 位时间戳
  - TSU 可以使用自己的时间戳，也可以使用外部时间戳
  - 可以输出时间戳供系统其他 mcan 模块使用

49.2 CAN 架构图

本模块的结构图如图 60。各个主要单元的功能如下，

发送处理单元：控制着从消息存储到 can 核的发送传输，可以设置成最大有 32 个发送缓存，

接收处理单元：控制着哪些消息匹配可以接收，可以设置成最大有 64 个接收缓存。

can 核：can 核控制着发送接收移位寄存器，处理所有的 ISO 11898-1：2015 协议的功能，支持 11 位和 29 位标识符。

中断控制：处理产生以及清除中断。

消息存储接口以及消息存储：储存发送以及接收的消息。

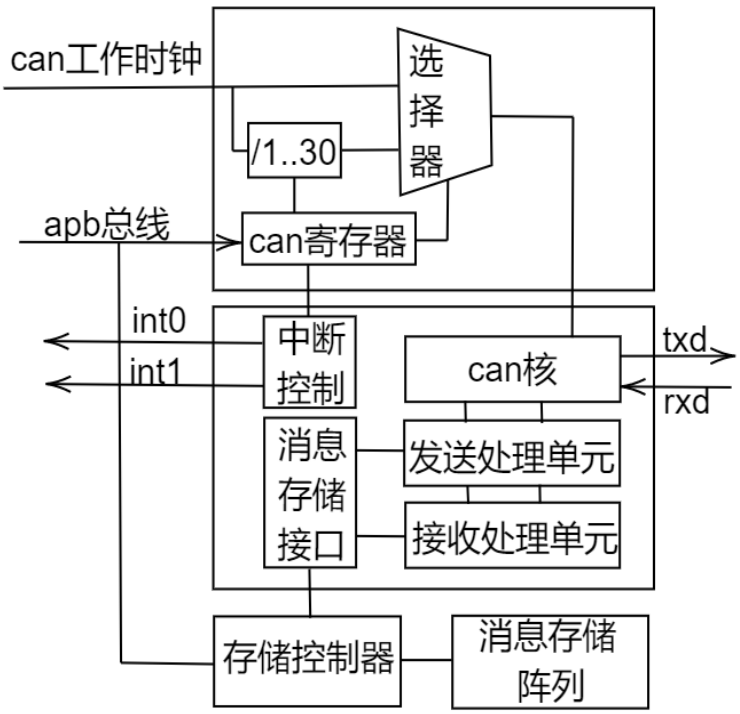


图 60: CAN 结构框图

49.3 管脚说明

表 207: CAN 管脚说明

管脚	方向	功能描述
CANn[1]_TXD	输出	CAN 总线数据输出
CANn_RXD	输入	CAN 总线数据输入
CANn_STANDBY	输出	CAN 控制器处于待机模式

[1] n=0,..,3

49.4 功能说明

49.4.1 CAN 一般说明

CAN 引脚功能

- CANn_TXD：CAN 总线输出数据。
- CANn_RXD：CAN 总线输入数据。
- CANn_STANDBY：当此信号为高时，表示 CAN 控制器处于待机模式，此时可以关掉 can 的时钟省电。

49.4.2 位时间

位时间指的是 CAN 总线的每个位的周期 T=1/波特率，由三部分组成如图图 61所示，

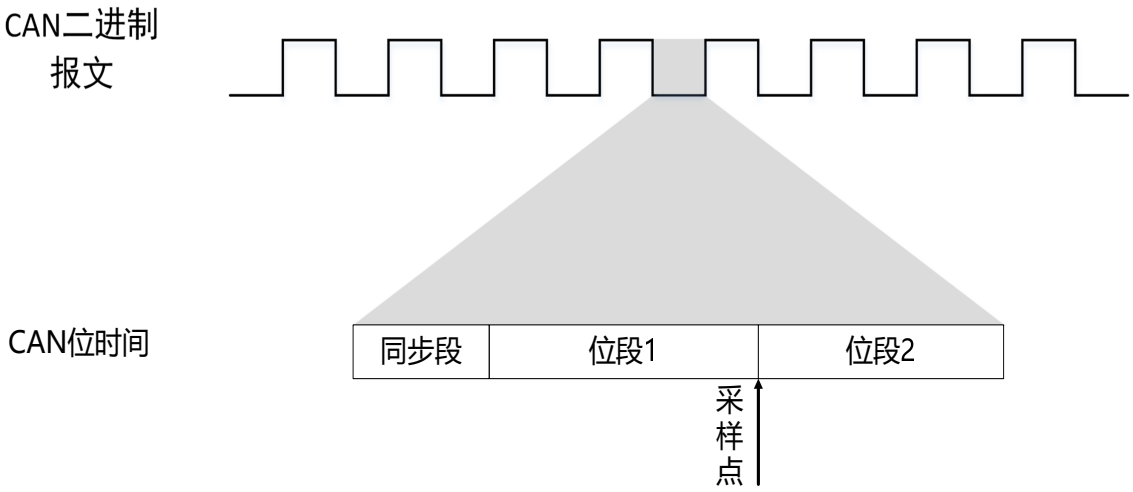


图 61: CAN 位时间

同步段：指的是在数值发生改变直到稳定的时候所需要的时间长度，在这里是固定的一个时间元。

位段 1：定义了采样的点，包括 can 标准里面定义的传播段和相位补偿段 1，它的周期是可以配置的，最大值是 16 个时间元，最小 1 个时间元。

位段 2：定义了传输点的位置，这个位段指的是 can 标准里的相位补偿段 2，它的周期是可以配置的，最大值是 8 个时间元，最小值是 1 个时间元。

重同步补偿宽度: 重同步补偿宽度 (SJW) 定义了延长或者缩短位段，它的值是可以配置的，最大为 4 个时间元，最小为 1 个时间元。

因此，各个段的时间计算如下：

位时间 tq = (CAN_DBTP.DBRP[4:0] + 1) * can_tq_clk_period

同步段时间 = 1 * tq

位段 1 时间 = tq * (CAN_DBTP.DTSEG1[4:0] + 1)

位段 2 时间 = tq * (CAN_DBTP.DTSEG2[3:0] + 1)

49.4.3 正常工作模式

当 CAN 初始化完成之后便可以把 cccr 寄存器的 init 位设置位 0，此时 CAN 控制器就会准备进行通信。当从 can 总线接收到的消息地址匹配 CAN 控制器设置的规则时，接收到的数据存到接收缓存或者接收的 FIFO0/1。放在发送缓存或者发送 FIFO0/1 里的数据在配完寄存器 TXBAR 也会自动的准备去发送到 CAN 总线。

49.4.4 CAN FD 工作模式

相对于普通模式，fd 的模式有两点不同，一是对数据部分的长度作了很大的扩充，DLC 最大支持 64 个字节。二是 can fd 帧的控制部分，数据部分以及校验部分处于更高的比特率。

can fd 可以通过设置 cccr 的 FDOE 位来使能，当使能 can fd 的时候，可以发送和接收 FD 帧，也可以发送和接收普通的帧。当发送缓存或者接收缓存里的 FDF 位是 0 的时候，can 控制器会把此时的帧解析成普通的帧，当 FDF 位为 1 的时候并且使能了 FDOE,can 控制器会解析成 FD 的帧。在实际使用的时候，系统启动后，所有的节点都在传输普通的帧格式，只有确定系统里的所有的节点都支持 CAN FD 的帧格式，那么所有的节点要一起转换成 FD 的操作。

CAN FD 的格式中，DLC 的含义和普通的 CAN 格式有些不同，DLC 的值是 0 到 8 的时候含义和普通的是一样的，9 到 15 的值的含义如表 208所示：

表 208: DLC 含义

DLC 的值	9	10	11	12	13	14	15
代表的字节数	12	16	20	24	32	48	64

49.4.5 静默模式

此模式可以通过设置 CCCR 的 MON 位为 1 来使能，在静默模式下（ISO11898-1:2015），can 控制器只收数据帧不发送数据帧，这种模式的好处是可以用来在不影响 can 总线的情况下观测 can 总线的数据流，其结构如图 62所示。

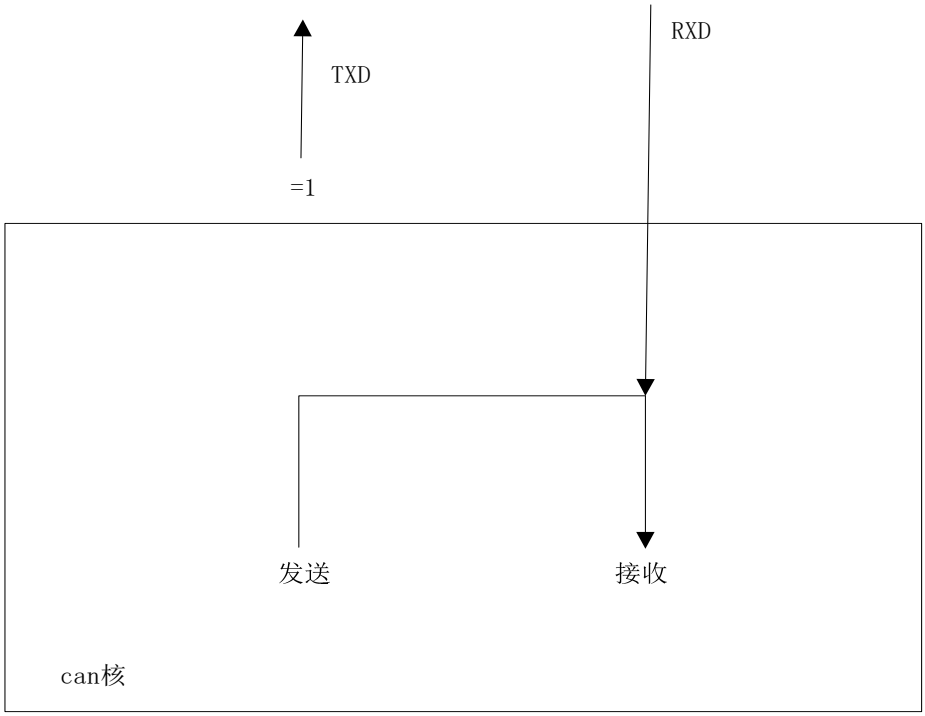


图 62: CAN 静默模式

49.4.6 待机模式

此模式可以通过设置 CCCR 的 CSR 位为 1 来使能，当设置了 CCCR 的 CSR 位为 1，can 控制器会等待所有的发送缓存里的任务完成，然后 CCCR 的 INIT 位被自动设置位 1 以防止用户增添新的任务，当成功进入待机模式后 CCCR 的 CSA 会变为 1，然后可以配置寄存器让 STANDBY 的输出为高，最后关掉输入的时钟来省电。当想退出待机模式的时候，首先要把 can 模块的时钟打开，然后写 CCCR 的 CSR 位为 0，最后软件在清 CCCR 的 INIT 位就可以唤醒。

49.4.7 外部环回模式

此模式可以通过设置 TEST 寄存器的 LBCK 位来使能，设置 TEST 寄存器前，需要设置 CCCR 的 TEST 位为 1，在此环回模式下，can 把输出的数据帧直接放到输入的端口并且把数据帧放到接收缓存或者接收 FIFO 中，并且输入的 RXD 和外面的端口断开，如图 63中可以看出外部环回的发送端口和接收端口的关系。外部环回用来硬件的自检。

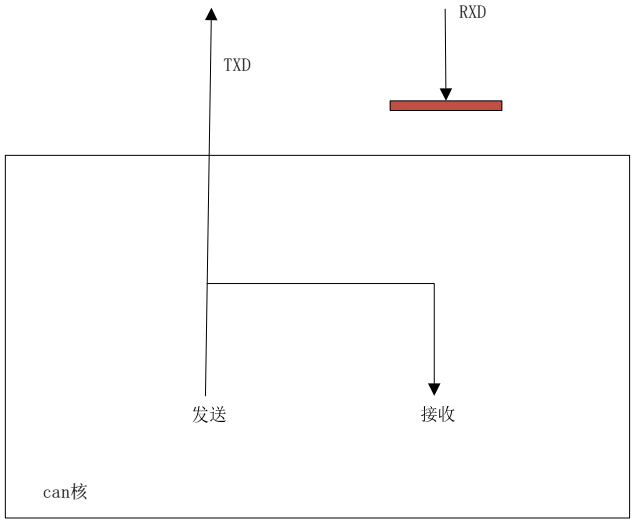


图 63: CAN 外部环回

49.4.8 内部环回模式

此模式可以通过设置 TEST 寄存器的 LBCK 位以及 CCCR 寄存器的 MON 位来使能，设置 TEST 寄存器前，需要设置 CCCR 的 TEST 位为 1，这个模式可以用来热自检，可以在不影响整个正在运行的 can 系统的情况下对 can 进行自检。在这种情况下，can 的 TXD 被置为了 1，RXD 和外面断开。如图 64所示。

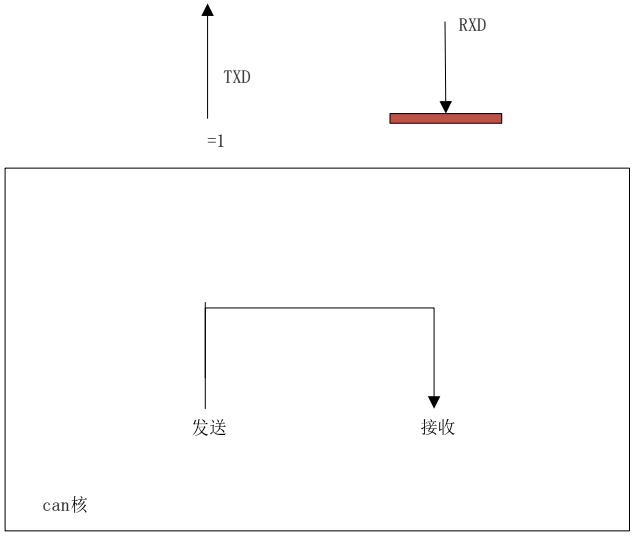


图 64: CAN 内部环回

49.4.9 时间戳的产生

can 控制器内部提供了 4 个 64 位的回环计数器，当 can 使用的时候可以通过寄存器 GLB_CTRL 的 tsu_tbin_sel 来选择时间的来源。时间管理单元（TSU）里可以储存 8 个 64 位的时间或者 16 个 32 位的时间，使用之前，需要先配置寄存器 CCCR 的 UTSU 位为 1，当使用时间戳接收消息的时候，在接收一帧的结尾的时候，会触发时间管理单元去记录当前选择的计数器的值，并且根据时间戳指针 (TSP) 的值，把计数器的值储存到对应的时间戳记录寄存器中，同时时间戳指针增加 1。并且把时间戳指针（TSP）的值放到接收缓存格式的 R1B 中的 RXTSP。

49.4.10 接收缓存数据格式

接收缓存的数据格式如图 65所示，其中

R1A: 当不使用时间管理单元 (TSU) 的时候 (CCCR 的 UTSU 位为 0)，这个时候 R1A 的 RXTS[15:0] 保存了 can 控制器内部自己的 16 位的时间戳。

R1B: 当使用时间管理单元 TSU 的时候 (CCCR 的 UTSU 位为 1)，并且设置了过滤器里的 SSYNC/ESYNC 位，这时候 R1B 的 TSC 位为 1 表示时间戳捕获成功，R1B 的 RXTSP 保存了 TSU 的时间戳寄存器编号，对应编号的时间戳寄存器保存了 32 位或者 64 位的时间戳。

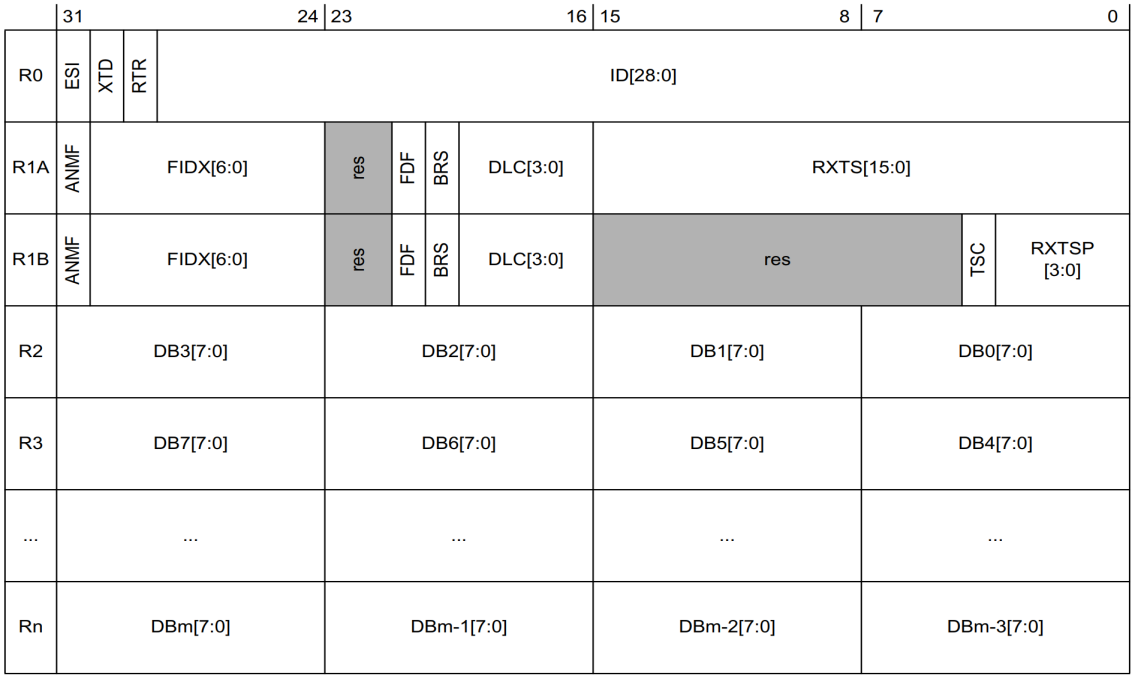


图 65: CAN 接收缓存数据格式

- R0 第 31 位 ESI：错误状态指示。
  - 0: 传输节点错误
  - 1: 传输节点无错误
- R0 第 30 位 XTD：扩展标识符。
  - 0: 11 位扩展标识符
  - 1: 29 位扩展标识符
- R0 第 29 位 RTR：远程传输请求。
  - 0: 接收到的是数据帧
  - 1: 接收到的是远程帧
- R0 第 28 到 0 位 ID：标识符。
  - 标准标识符有效位是 ID[28:18] 扩展标识符有效位为 ID[28:0]
- R1 第 31 位 ANMF：接受不匹配帧。
  - 0: 接收的帧匹配接收过滤器
  - 1: 接收的帧不匹配接收过滤器
- R1 第 30 位到 24 位 FIDX[6:0]：过滤器的索引。
  - 范围为 0 到 127，表示匹配的接受过滤器的索引

- R1 第 21 位 FDF：FD 的帧格式。
  - 0: 表示为传统的 can 帧格式
  - 1: 表示为 FD 帧格式
- R1 第 20 位 BRS：比特率切换。
  - 0: 表示接收的帧没有比特率切换
  - 1: 表示接收的帧有比特率切换
- R1 第 19 位到 16 位 DLC[3:0]：数据长度编码。
  - 0 到 8: 表示接收的帧的数据长度为 0 到 8 个字节
  - 9 到 15: 普通的帧的话表示接收的帧有 8 个字节，FD 帧的话分别表示 12/16/20/24/32/48/64 个字节
- R1A 第 15 位到 0 位 RXTS[15:0]：接收时间戳。
  - 帧接收开始时捕获的时间戳计数器值。
- R1B 第 4 位 TSC：时间戳捕获。
  - 0: 表示没有时间戳的捕获
  - 1: 表示有时间戳的捕获，且时间戳寄存器的编号在 R1B 的 RXTSP 的位置
- R1B 第 3 位到第 0 位 RXTSP[3:0]：接收时间戳编号。
  - 表示接收时的时间戳存到了哪个时间戳寄存器，也就是时间戳寄存器的编号。
- R2 第 31 位到 24 位 DB3[7:0]：数据字节 3。
- R2 第 23 位到 16 位 DB2[7:0]：数据字节 2。
- R2 第 15 位到 8 位 DB1[7:0]：数据字节 1。
- R2 第 7 位到 0 位 DB0[7:0]：数据字节 0。
- R3 第 31 位到 24 位 DB3[7:0]：数据字节 7。
- R3 第 23 位到 16 位 DB2[7:0]：数据字节 6。
- R3 第 15 位到 8 位 DB1[7:0]：数据字节 5。
- R3 第 7 位到 0 位 DB0[7:0]：数据字节 4。
- ... ... : ...
- Rn 第 31 位到 24 位 DB3[7:0]：数据字节 m。
- Rn 第 23 位到 16 位 DB2[7:0]：数据字节 m-1。
- Rn 第 15 位到 8 位 DB1[7:0]：数据字节 m-2。
- Rn 第 7 位到 0 位 DB0[7:0]：数据字节 m-3。

49.4.11 发送缓存数据格式

发送缓存的数据格式如图 66所示

- T0 第 31 位 ESI：错误状态指示。
  - 0: 传输节点错误
  - 1: 传输节点无错误
- T0 第 30 位 XTD：扩展标识符。
  - 0: 11 位扩展标识符
  - 1: 29 位扩展标识符
- T0 第 29 位 RTR：远程传输请求。
  - 0: 接收到的是数据帧
  - 1: 接收到的是远程帧

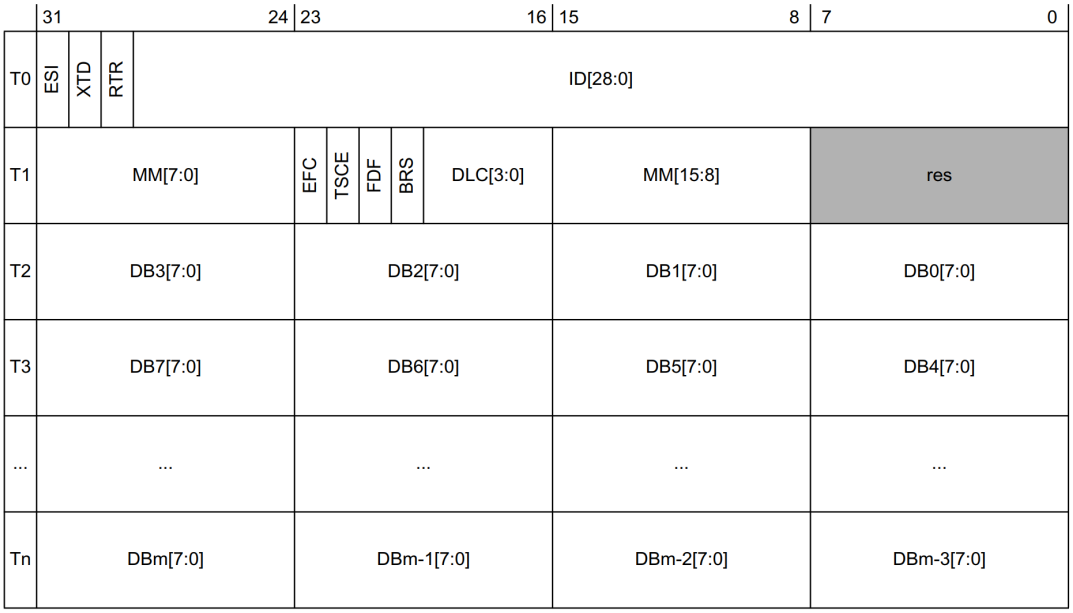


图 66: CAN 发送缓存数据格式

- T0 第 28 到 0 位 ID：标识符。
  - 标准标识符有效位是 ID[28:18] 扩展标识符有效位为 ID[28:0]
- T1 第 31 位到 24 位 MM[7:0]：消息标记低 8 位。
  - 在配置发送缓存数据的时候由 cpu 写入，然后会复制到发送时间 FIFO 中进行发送消息的识别判断
- T1 第 30 位到 24 位 FIDX[6:0]：过滤器的索引。
  - 范围为 0 到 127，表示匹配的接受过滤器的索引
- T1 第 23 位 EFC：事件 FIFO 控制。
  - 0: 不保存发送的行为事件
  - 1: 保存发送行为事件
- T1 第 22 位 BTSCE: TSU 的时间戳捕获使能，只有 CCCR 的寄存器的 UTSU 的位置起来的时候才有效
  - 0: 时间戳捕获不使能
  - 1: 时间戳捕获使能
- T1 第 21 位 FDF：FD 帧格式。
  - 0: 表示发送的帧为普通帧格式
  - 1: 表示发送的帧为 CAN FD 帧格式
- T1 第 20 位 BRS：比特率切换。
  - 0: 表示接收的帧没有比特率切换
  - 1: 表示接收的帧有比特率切换
- T1 第 19 位到 16 位 DLC[3:0]：数据长度编码。
  - 0 到 8: 表示接收的帧的数据长度为 0 到 8 个字节
  - 9 到 15: 普通的帧的话表示接收的帧有 8 个字节，FD 帧的话分别表示 12/16/20/24/32/48/64 个字节
- T1 第 15 位到 8 位 MM[15:8]：消息标记高 8 位。

  - 在配置发送缓存数据的时候由 cpu 写入，然后会复制到发送时间 FIFO 中进行发送消息的识别判断。

- T2 第 31 位到 24 位 DB3[7:0]：数据字节 3。
- T2 第 23 位到 16 位 DB2[7:0]：数据字节 2。
- T2 第 15 位到 8 位 DB1[7:0]：数据字节 1。
- T2 第 7 位到 0 位 DB0[7:0]：数据字节 0。
- T3 第 31 位到 24 位 DB3[7:0]：数据字节 7。
- T3 第 23 位到 16 位 DB2[7:0]：数据字节 6。
- T3 第 15 位到 8 位 DB1[7:0]：数据字节 5。
- T3 第 7 位到 0 位 DB0[7:0]：数据字节 4。
- ... ... : ...
- Tn 第 31 位到 24 位 DB3[7:0]：数据字节 m。
- Tn 第 23 位到 16 位 DB2[7:0]：数据字节 m-1。
- Tn 第 15 位到 8 位 DB1[7:0]：数据字节 m-2。
- Tn 第 7 位到 0 位 DB0[7:0]：数据字节 m-3。

49.4.12 发送事件数据格式

发送事件的数据保存了有关传输的信息，cpu 可以安照发送的先后顺序读取发送事件 FIFO, 发送事件 FIFO 的状态可以通过寄存器 TXEFS 来读取到。FIFO 里数据的含义如图 67所示。其中

E1A: 当不使用时间管理单元 (TSU) 的时候 (CCCR 的 UTSU 位为 0) 并且 CCCR 寄存器的 WMM 为 0，这个时候 E1A 的 TXTS[15:0] 保存了 can 控制器内部自己的 16 位的时间戳。

E1B: 当使用时间管理单元 TSU 的时候 (CCCR 的 UTSU 位为 1) 或者打开了 16 位的消息标记位 (CCCR 的 WMM 配置为 1)，这时候 E1B 的 TSC 位为 1 表示时间戳捕获成功，T1B 的 TXTSP 保存了 TSU 的时间戳寄存器编号，对应编号的时间戳寄存器保存了 32 位或者 64 位的时间戳。

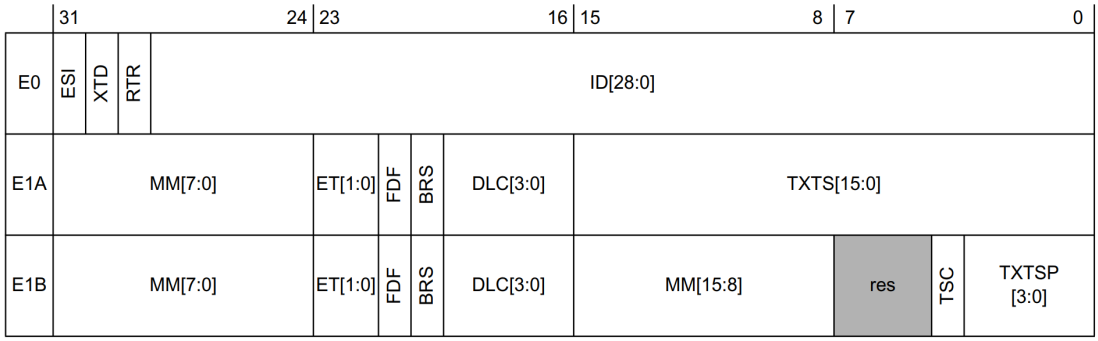


图 67: CAN 发送事件数据格式

- E0 第 31 位 ESI：错误状态指示。
  - 0: 传输节点错误
  - 1: 传输节点无错误
- E0 第 30 位 XTD：扩展标识符。
  - 0: 11 位扩展标识符
  - 1: 29 位扩展标识符
- E0 第 29 位 RTR：远程传输请求。
  - 0: 接收到的是数据帧

  - 1: 接收到的是远程帧
- E0 第 28 到 0 位 ID：标识符。
  - 标准标识符有效位是 ID[28:18] 扩展标识符有效位为 ID[28:0]
- E1 第 31 位到 24 位 MM[7:0]：过滤器的索引。
  - 范围为 0 到 127，表示匹配的接受过滤器的索引
- E1 第 23 位到 22 位 ET[1:0]：事件的类型。
  - 00: 保留
  - 01: 发送事件
  - 10: 取消后再发送
  - 11: 保留
- E1 第 21 位 FDF：FD 的帧格式。
  - 0: 表示为传统的 can 帧格式
  - 1: 表示为 FD 帧格式
- E1 第 20 位 BRS：比特率切换。
  - 0: 表示接收的帧没有比特率切换
  - 1: 表示接收的帧有比特率切换
- E1 第 19 位到 16 位 DLC[3:0]：数据长度编码。
  - 0 到 8: 表示接收的帧的数据长度为 0 到 8 个字节
  - 9 到 15: 普通的帧的话表示接收的帧有 8 个字节，FD 帧的话分别表示 12/16/20/24/32/48/64 个字节
- E1A 第 15 位到 0 位 TXTS[15:0]：发送时间戳。
  - 帧发送开始时捕获的时间戳计数器值。
- E1B 第 15 位到 8 位 MM[15:8]：消息标记高 8 位。
  - 在配置发送缓存数据的时候由 cpu 写入，然后会复制到发送时间 FIFO 中进行发送消息的识别判断。
- E1B 第 4 位 TSC：时间戳捕获。
  - 0: 表示没有时间戳的捕获
  - 1: 表示有时间戳的捕获，且时间戳寄存器的编号在 R1B 的 RXTSP 的位置
- E1B 第 3 位到第 0 位 TXTSP[3:0]：发送时间戳编号。
  - 表示发送时的时间戳存到了哪个时间戳寄存器，也就是时间戳寄存器的编号。

49.4.13 标准消息标识过滤

最多可以位 11 位的标识符配置 128 个过滤器，当访问过滤器的时候，它的地址是过滤器的起始地址寄存器 SIDFC 的值加上过滤器的索引偏移。过滤器的内容含义如如图 68

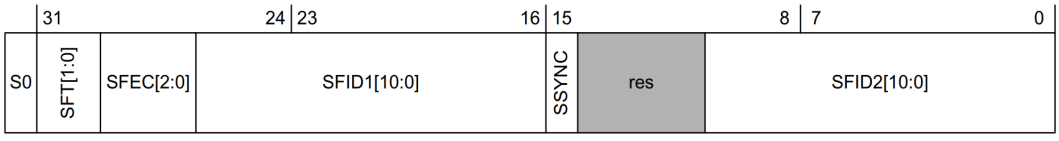


图 68: CAN 标准消息标识过滤

- S0 第 31 位到 30 位 SFT[1:0]：标准过滤器类型。
  - 00: 标识符范围过滤器，从 SFID1 开始，到 SFID2 截至，其中 SFID2>=SFID1
  - 01: 标识符值过滤器，等于 SFID1 或者 SFID2

  - 10: 传统掩码过滤器，SFID1 为过滤器的值，SFID2 为对应的掩码
  - 11: 不使能过滤器
- S0 第 29 位到 27 位 SFEC[2:0]：标准过滤器的配置。
  - 000: 不使能过滤器
  - 001: 过滤器匹配后把数据存储到接收 FIFO 0 中
  - 010: 过滤器匹配后把数据存储到接收 FIFO 1 中
  - 011: 过滤器匹配后不接收数据
  - 100: 过滤器匹配后仅仅设置优先级，不存储
  - 101: 过滤器匹配后设置优先级并且把数据存储到接收 FIFO 0 中
  - 110: 过滤器匹配后设置优先级并且把数据存储到接收 FIFO 1 中
  - 111: 过滤器匹配后把数据存储到接收缓存（Rx buffer）中
- S0 第 26 到 16 位 SFID1[10:0]：标准标识符过滤器 1。
  - 第一个标准标识符过滤器，用来在不同的情况下的过滤
- S0 第 15 位 SSYNC：标准的同步消息。只有当 CCCR 的 UTSU 的位为 1 时有效，触发时间管理单元 (TSU) 来记录时间戳
  - 0：不使能触发时间管理单元记录时间戳
  - 1：不使能触发时间管理单元记录时间戳
- S0 第 10 到 0 位 SFID2[10:0]：标准标识符过滤器 2。并且根据不同的 SFEC 的值有不同的含义。
  当 SFEC 的值为从 000 到 110 时，代表了标准过滤器的第二个部分。
  当 SFEC 的值为 111 时，
  - SFID2[10:9] 定义了接收到的消息存放到缓存还是被是为调试消息。
    * 00：存储到接收缓存（Rx buffer）
    * 01：调试消息 A
    * 01：调试消息 B
    * 01：调试消息 C
  - SFID2[8:6] 定义了用于控制扩展接口上的滤波器事件引脚
  - SFID2[5:0] 定义了接收的数据存到储存相对起始地址的偏移

49.4.14 扩展消息标识过滤

最多可以为 29 位的标识符配置 64 个过滤器，当访问过滤器的时候，它的地址是过滤器的起始地址寄存器 XIDFC 的值加上两倍的过滤器的索引偏移。过滤器的内容含义如如图 69

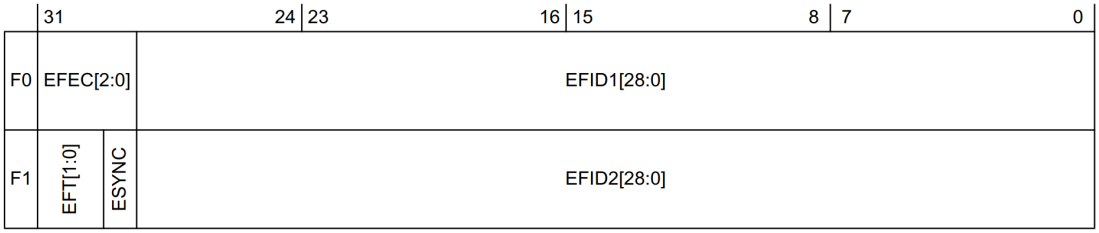


图 69: CAN 扩展消息标识过滤

- F0 第 31 位到 29 位 EFEC[2:0]：标准过滤器的配置。
  - 000: 不使能过滤器
  - 001: 过滤器匹配后把数据存储到接收 FIFO 0 中
  - 010: 过滤器匹配后把数据存储到接收 FIFO 1 中

  - 011: 过滤器匹配后不接收数据
  - 100: 过滤器匹配后仅仅设置优先级，不存储
  - 101: 过滤器匹配后设置优先级并且把数据存储到接收 FIFO 0 中
  - 110: 过滤器匹配后设置优先级并且把数据存储到接收 FIFO 1 中
  - 111: 过滤器匹配后把数据存储到接收缓存（Rx buffer）中
- F0 第 28 到 0 位 EFID1[28:0]：扩展标识符过滤器 1。
  - 第一个扩展标识符过滤器，用来在不同的情况下的过滤
- F1 第 31 位到 30 位 EFT[1:0]：扩展过滤器类型。
  - 00: 标识符范围过滤器，从 EFID1 开始，到 EFID2 截至，其中 EFID2>=EFID1
  - 01: 标识符值过滤器，等于 EFID1 或者 EFID2
  - 10: 传统掩码过滤器，EFID1 为过滤器的值，EFID2 为对应的掩码
  - 11: 不使能过滤器
- F1 第 29 位 ESYNC：扩展的同步消息。只有当 CCCR 的 UTSU 的位为 1 时有效，触发时间管理单元 (TSU) 来记录时间戳
  - 0：不使能触发时间管理单元记录时间戳
  - 1：不使能触发时间管理单元记录时间戳
- F1 第 28 到 0 位 EFID2[28:0]：扩展标识符过滤器 2。并且根据不同的 SFEC 的值有不同的含义。
  当 EFEC 的值为从 000 到 110 时，代表了标准过滤器的第二个部分。
  当 EFEC 的值为 111 时，
  - SFID2[10:9] 定义了接收到的消息存放到缓存还是被是为调试消息。
    * 00：存储到接收缓存（Rx buffer）
    * 01：调试消息 A
    * 01：调试消息 B
    * 01：调试消息 C
  - SFID2[8:6] 定义了用于控制扩展接口上的滤波器事件引脚
  - SFID2[5:0] 定义了接收的数据存到储存相对起始地址的偏移

49.4.15 消息存储器示意图

消息储存的宽度为 32 位，每个部分最多的空间如图 70所示，各个部分的先后顺序也没有严格的先后顺序。

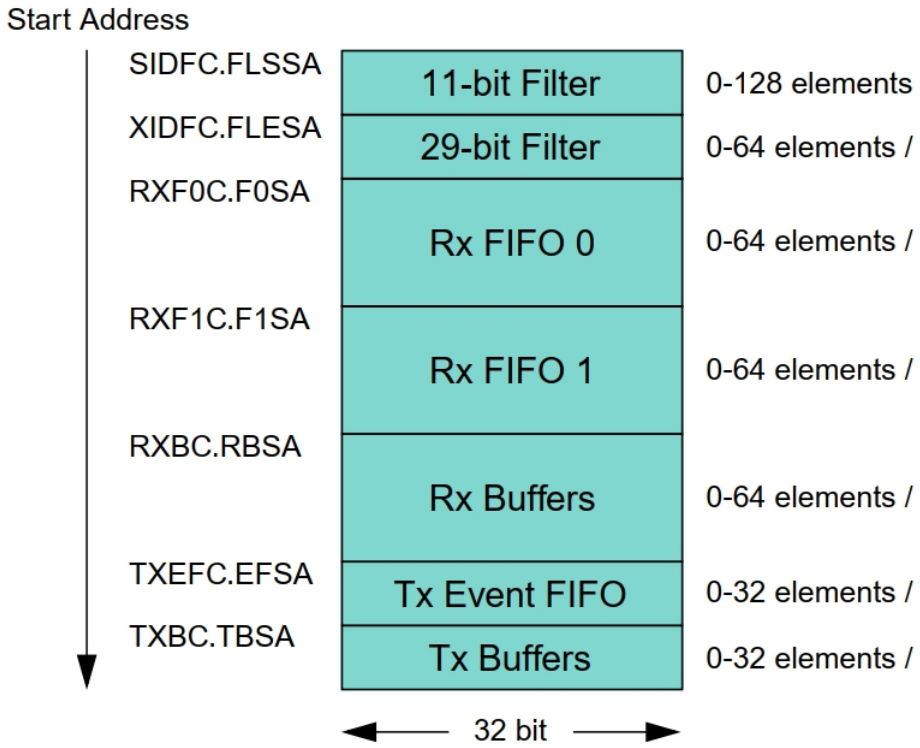


图 70: 消息存储器示意图

49.5 MCAN 寄存器

49.5.1 MCAN 寄存器说明

MCAN 的寄存器列表如下：

MCAN0 base address: 0xF0280000

MCAN1 base address: 0xF0284000

MCAN2 base address: 0xF0288000

MCAN3 base address: 0xF028C000

地址偏移	名称	描述	复位值
0x0004	ENDN	字节序寄存器	0x87654321
0x000C	DBTP	数据位定时和预分频器寄存器	0x00000A33
0x0010	TEST	测试寄存器	0x00000000
0x0014	RWD	ram 看门狗寄存器	0x00000000
0x0018	CCCR	CC 控制寄存器	0x00000001
0x001C	NBTP	标称位定时和预分频寄存器	0x06000A03
0x0020	TSCC	时间戳计数器配置寄存器	0x00000000
0x0024	TSCV	时间戳计数器值寄存器	0x00000000

地址偏移	名称	描述	复位值
0x0028	TOCC	超时计数器配置寄存器	0xFFFF0000
0x002C	TOCV	超时计数器值寄存器	0x0000FFFF
0x0040	ECR	错误计数器寄存器	0x00000000
0x0044	PSR	协议状态寄存器	0x00000707
0x0048	TDCR	发送延迟补偿寄存器	0x00000000
0x0050	IR	中断寄存器	0x00000000
0x0054	IE	中断使能寄存器	0x00000000
0x0058	ILS	中断线选择寄存器	0x00000000
0x005C	ILE	中断线使能寄存器	0x00000000
0x0080	GFC	全局过滤器配置寄存器	0x00000000
0x0084	SIDFC	标准 ID 过滤器配置寄存器	0x00000000
0x0088	XIDFC	扩展 ID 过滤器配置寄存器	0x00000000
0x0090	XIDAM	扩展 ID 和掩码寄存器	0x1FFFFFFF
0x0094	HPMS	高优先级消息状态寄存器	0x00000000
0x0098	NDAT1	新数据 1 寄存器	0x00000000
0x009C	NDAT2	新数据 2 寄存器	0x00000000
0x00A0	RXF0C	接收 FIFO 0 配置寄存器	0x00000000
0x00A4	RXF0S	接收 FIFO 0 状态寄存器	0x00000000
0x00A8	RXF0A	接收 FIFO 0 确认寄存器	0x00000000
0x00AC	RXBC	接收缓冲区配置寄存器	0x00000000
0x00B0	RXF1C	接收 FIFO 1 配置寄存器	0x00000000
0x00B4	RXF1S	接收 FIFO 1 状态寄存器	0x00000000
0x00B8	RXF1A	接收 FIFO 1 确认寄存器	0x00000000
0x00BC	RXESC	接收缓冲区元素大小配置寄存器	0x00000000
0x00C0	TXBC	发送缓冲区配置寄存器	0x00000000
0x00C4	TXFQS	发送 FIFO/队列状态寄存器	0x00000000
0x00C8	TXESC	发送缓冲区元素大小配置寄存器	0x00000000
0x00CC	TXBRP	发送缓冲区请求挂起寄存器	0x00000000
0x00D0	TXBAR	发送缓冲区添加请求寄存器	0x00000000
0x00D4	TXBCR	发送缓冲区取消请求寄存器	0x00000000
0x00D8	TXBTO	发送缓冲区发送已发生寄存器	0x00000000
0x00DC	TXBCF	发送缓冲区取消完成寄存器	0x00000000
0x00E0	TXBTIE	发送缓冲区发送中断使能寄存器	0x00000000
0x00E4	TXBCIE	发送缓冲区取消完成中断使能寄存器	0x00000000
0x00F0	TXEFC	发送事件 FIFO 配置寄存器	0x00000000
0x00F4	TXEFS	发送事件 FIFO 状态寄存器	0x00000000

地址偏移	名称	描述	复位值
0x00F8	TXEFA	发送事件 FIFO 确认寄存器	0x00000000
0x0200	TS_SEL[TS_SEL0]	时间戳 0-15	0x00000000
0x0204	TS_SEL[TS_SEL1]	时间戳 0-15	0x00000000
0x0208	TS_SEL[TS_SEL2]	时间戳 0-15	0x00000000
0x020C	TS_SEL[TS_SEL3]	时间戳 0-15	0x00000000
0x0210	TS_SEL[TS_SEL4]	时间戳 0-15	0x00000000
0x0214	TS_SEL[TS_SEL5]	时间戳 0-15	0x00000000
0x0218	TS_SEL[TS_SEL6]	时间戳 0-15	0x00000000
0x021C	TS_SEL[TS_SEL7]	时间戳 0-15	0x00000000
0x0220	TS_SEL[TS_SEL8]	时间戳 0-15	0x00000000
0x0224	TS_SEL[TS_SEL9]	时间戳 0-15	0x00000000
0x0228	TS_SEL[TS_SEL10]	时间戳 0-15	0x00000000
0x022C	TS_SEL[TS_SEL11]	时间戳 0-15	0x00000000
0x0230	TS_SEL[TS_SEL12]	时间戳 0-15	0x00000000
0x0234	TS_SEL[TS_SEL13]	时间戳 0-15	0x00000000
0x0238	TS_SEL[TS_SEL14]	时间戳 0-15	0x00000000
0x023C	TS_SEL[TS_SEL15]	时间戳 0-15	0x00000000
0x0240	CREL	内核发布信息寄存器	0x00000000
0x0244	TSCFG	时间戳配置寄存器	0x00000000
0x0248	TSS1	时间戳状态寄存器 1	0x00000000
0x024C	TSS2	时间戳状态寄存器 2	0x00000000
0x0250	ATB	实际时间基准寄存器	0x00000000
0x0254	ATBH	实际时间基准高位寄存器	0x00000000
0x0400	GLB_CTL	全局控制寄存器	0x00000000
0x0404	GLB_STATUS	全局状态寄存器	0x00000000

表 209: MCAN 寄存器列表

49.5.2 寄存器详细信息

MCAN 的寄存器详细说明如下：

49.5.3 ENDN (0x4)

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

ENDN [31:0]

位域	名称	描述
31-0	EVT	字节序测试值 字节序测试值为 0x8765 4321。

ENDN 位域

49.5.4 DBTP (0xC)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD								TDC	RSVD		DBRP					RSVD			DTSEG1					DTSEG2				DSJW			
N/A								RW	N/A		RW					N/A			RW					RW				RW			
x	x	x	x	x	x	x	x	0	x	x	0	0	0	0	0	x	x	x	0	1	0	1	0	0	0	1	1	0	0	1	1

DBTP [31:0]

位域	名称	描述
23	TDC	发送延时补偿使能 0：禁止收发器延迟补偿 1：使能收发器延迟补偿
20-16	DBRP	数据比特率预分频器 生成位时间片时将振荡器频率除以的值。位时间为该时间片的倍数。比特率预分频器的有效值为 0 到 31。当 TDC=1 时，范围限制为 0，1。硬件将该值解析为编程值加 1。
12-8	DTSEG1	数据有效到采样点的时间 有效值为 1 到 31。实际上，硬件会将该值解析为编程值加 1。
7-4	DTSEG2	采样点后数据有效时间 有效值为 1 到 15。实际上，硬件会将该值解析为编程值加 1。
3-0	DSJW	同步跳转宽度 有效值为 0 到 15。实际上，硬件会将该值解析为编程值加 1。

DBTP 位域

49.5.5 TEST (0x10)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD										SVAL	TXBNS					RSVD		PVAL	TXBNP					RX	TX		LBCK	RSVD			
N/A										R	R					N/A		R	R					R	RW		RW	N/A			
x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	x	x	0	0	0	0	0	0	0	0	0	0	x	x	x	x

TEST [31:0]

位域	名称	描述
21	SVAL	TXBNS 有效指示 0：TXBNS 的值无效 1：TXBNS 的值有效
20-16	TXBNS	已启动的发送缓存区号 最后开始传输的消息的缓冲区号。SVAL 置“1”时有效。取值范围为 0 ~ 31。
13	PVAL	TXBNP 有效指示 0：TXBNP 的值无效 1：TXBNP 的值有效
12-8	TXBNP	待发送消息的缓冲区号 准备传输的消息缓冲区号。PVAL 置 1 时有效。取值范围为 0 ~ 31。
7	RX	can 总线电平 监视发送引脚 pin m_can_rx 的实际值 0：CAN 总线为显性（m_can_rx = “0”） 1：CAN 总线为隐性（m_can_rx = “1”）
6-5	TX	can 发送控制模式寄存器 00：复位值，m_can_tx 由 CAN 内核控制，会在 CAN 位时间结束时更新 01：可在引脚 m_can_tx 监控采样点 10：引脚 m_can_tx 上为显性（“0”）电平 11：引脚 m_can_tx 上为隐性（“1”）电平
4	LBCK	环回模式控制寄存器 0：复位值，禁止环回模式 1：使能环回模式

TEST 位域

49.5.6 RWD (0x14)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																WDV								WDC							
N/A																R								RW							
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

RWD [31:0]

位域	名称	描述
15-8	WDV	计时器实际值 实际消息 RAM 看门狗计数器值。

位域	名称	描述
7-0	WDC	看门狗配置 消息 RAM 看门狗计数器的起始值。如果使用复位值“00”，计数器会禁止。

RWD 位域

49.5.7 CCCR (0x18)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																NISO	TXP	EFBI	PXHD	WMM	UTSU	BRSE	FDOE	TEST	DAR	MON	CSR	CSA	ASM	CCE	INIT
N/A																RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	R	RW	RW	RW
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

CCCR [31:0]

位域	名称	描述
15	NISO	非 ISO 操作 如果此位置 1，FDCAN 会使用 Bosch CAN FD 规范 V1.0 规定的 CAN FD 帧格式。 0：符合 ISO11898-1 规定的 CAN FD 帧格式 1：符合 Bosch CAN FD 规范 V1.0 规定的 CAN FD 帧格式 注：当通用参数 iso_only_g 在硬件合成中被设置为“1”时，该位将被保留，并被读取为“0”。M_CAN 总是按照 ISO 11898-1:2015 的 CAN FD 帧格式操作。
14	TXP	帧间暂停使能寄存器 如果此位置“1”，成功发送帧后，FDCAN 会先暂停两个 CAN 位时间，然后再开始进行下一次发送。 0：禁止 1：使能
13	EFBI	总线同步期间的边沿过滤 0：禁止边沿过滤 1：需要两个连续显性 tq 才能检测硬同步边沿
12	PXHD	协议异常处理禁止 0：使能协议异常处理 1：禁止协议异常处理
11	WMM	宽消息标记 允许使用 16 位宽消息标记。当使用 16 位宽消息标记 (WMM = “1”) 时，对发送事件 FIFO 禁用 16 位内部时间戳。 0：使用 8 位消息标记 1：使用 16 位消息标记, 替换发送事件 FIFO 中的 16 位时间戳

位域	名称	描述
10	UTSU	使用时间戳单元 当 UTSU 置“1”时，无论 WMM 的值如何，也将启用 16 位宽消息标记。 0：内部时间戳 1：TSU 外部时间戳 注：当通用参数 connected_tsu_g = “0”时，没有 TSU 连接到 M_CAN。在这种情况下，UTSU 位通过合成被固定为 0。
9	BRSE	比特率切换使能 0：禁止发送时进行比特率切换 1：使能发送时进行比特率切换 注意: 当禁用 CAN FD 操作 FDOE = “0”时，将忽略 BRSE 值。
8	FDOE	FD 模式使能 0：禁止 FD 操作 1：使能 FD 操作
7	TEST	测试模式使能 0：正常操作，寄存器 TEST 保存复位值 1：测试模式，使能对寄存器 TEST 的写访问
6	DAR	禁止自动重发送 0：使能自动重发送未成功发送的消息 1：禁止自动重发送
5	MON	总线监控模式 仅当 CCE 和 INIT 均置“1”时，才能通过软件将 MON 位置 1。此位可随时通过主机 复位。 0：禁止总线监控模式 1：使能总线监控模式
4	CSR	时钟停止请求 0：未请求时钟停止 1：已请求时钟停止。如果请求时钟停止，则在所有挂起的传输请求均已完成且 CAN 总线达到空闲状态之后，INIT 会先置“1”，然后 CSA 会置“1”。
3	CSA	时钟停止确认 0：未确认时钟停止 1：可通过停止 APB 时钟和内核时钟将 FDCAN 设为掉电状态
2	ASM	ASM 受限工作模式 仅当 CCE 和 INIT 均置“1”时，才能通过软件将 ASM 位置“1”。此位可随时通过软件复位。如果 FDCAN 连接到 CAN 时钟校准单元，只要校准未完成，ASM 位就会由硬件置 1。 0：正常 CAN 操作 1：受限工作模式激活

位域	名称	描述
1	CCE	配置更改使能 0：CPU 对受保护的配置寄存器没有写访问权限 1：CPU 对受保护的配置寄存器有写访问权限（CCCR.INIT = “1”时）
0	INIT	初始化 0：正常工作 1：启动初始化

CCCR 位域

49.5.8 NBTP (0x1C)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NSJW							NBRP									NTSEG1								RSVD	NTSEG2						
RW							RW									RW								N/A	RW						
0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	x	0	0	0	0	0	1	1

NBTP [31:0]

位域	名称	描述
31-25	NSJW	标称 (重新) 同步跳转宽度 有效值为 1 到 127。实际上，硬件会将该值解析为编程值加 1。
24-16	NBRP	标称比特率预分频 生成位时间片时将振荡器频率除以的值。位时间为该时间片的倍数。有效值为 0 到 511。 实际上，硬件会将该值解析为编程值加 1。
15-8	NTSEG1	采样点之前的标称时间段 有效值为 1 到 255。实际上，硬件会将该值解析为编程值加 1。
6-0	NTSEG2	采样点之后的标称时间段 有效值为 1 到 127。实际上，硬件会将该值解析为编程值加 1。

NBTP 位域

49.5.9 TSCC (0x20)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD												TCP				RSVD														TSS	
N/A												RW				N/A														RW	
x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0

TSCC [31:0]

位域	名称	描述
19-16	TCP	时间戳计数器预分频器 将时间戳和超时计数器时间单位配置为 CAN 位时间的倍数 [1…16]。 实际上，硬件会将该值解析为编程值加 1。
1-0	TSS	时间戳选择 00= 时间戳计数器永远为 0x0000 01= 时间戳计数器通过 TCP 递增 10= 使用外部时间戳计数器 11= 和 “00” 一样

TSCC 位域

49.5.10 TSCV (0x24)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																TSC															
N/A																RC															
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

TSCV [31:0]

位域	名称	描述
15-0	TSC	时间戳计数器 内部/外部时间戳计数器值是在（接收和发送）帧开始时捕获的。 当 TSCC[TSS] = “01” 时，时间戳计数器会以 CAN 为时间的倍数 [1…16] 递增，具体取决于 TSCC[TCP] 的配置。 回卷操作会将中断标志 IR[TSW] 置 “1”。写访问会将计数器复位为 “0”。当 TSCC.TSS = “10” 时，TSC 会反映外部时间戳计数器值。写访问没有影响。

TSCV 位域

49.5.11 TOCC (0x28)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TOP																RSVD													TOS		RP
RW																N/A													RW		RW
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0

TOCC [31:0]

位域	名称	描述
31-16	TOP	超时时长 超时计数器（递减计数器）的起始值。配置超时时长。
2-1	TOS	超时选择 在连续模式下工作时，对 TOCV 进行写访问会将计数器预设为由 TOCC[TOP] 配置的值并继续递减计数。超时计数器由其中一个 FIFO 控制时，空 FIFO 会将计数器预设为由 TOCC[TOP] 配置的值。存储了第一个 FIFO 元素后，会开始递减计数。 00：连续工作 01：超时由发送事件 FIFO 控制 10：超时由接收 FIFO 0 控制 11：超时由接收 FIFO 1 控制
0	RP	使能超时计数器 0：禁止超时计数器 1：使能超时计数器

TOCC 位域

49.5.12 TOCV (0x2C)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																TOC															
N/A																RC															
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

TOCV [31:0]

位域	名称	描述
15-0	TOC	超时计数器值 超时计数器会以 CAN 位时间的倍数 [1…16] 递减，具体取决于 TSCC.TCP 的配置。当计数器递减至 0 时，中断标志 IR.TOO 会置“1”，超时计数器会停止计数。开始和复位/重启条件是通过 TOCC.TOS 配置的。 注：字节访问：当 TOCC.TOS = “00”时, 写其中一个寄存器字节 3/2/1/0 将会预设 Timeout Counter 寄存器。

TOCV 位域

49.5.13 ECR (0x40)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD								CEL								RP	REC							TEC							
N/A								X								R	R							R							
x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ECR [31:0]

位域	名称	描述
23-16	CEL	CAN 错误记录 每当 CAN 协议错误导致发送错误计数器或接收错误计数器递增时，计数器都会递增。对 CEL 进行读访问时，计数器会复位。计数器值达到 0xFF 时，会停止计数；TEC 或 REC 下一次递增时，中断标志 IR[ELO] 会置“1”。 访问类型为 RX：进行读访问时复位。 注：字节访问：读字节 2 将重置 CEL，读字节 3/1/0 没有影响。
15	RP	接收错误被动 0：接收错误计数器低于错误被动级别 128 1：接收错误计数器已达到错误被动级别 128
14-8	REC	接收错误计数器 接收错误计数器的实际状态，取值范围为 0 到 127。
7-0	TEC	发送错误计数器 发送错误计数器的实际状态，取值范围为 0 到 255。 注：如果 CCCR.ASM 置“1”，则检测到 CAN 协议错误时，CAN 协议控制器不会使 TEC 和 REC 递增，但 CEL 仍会递增。

ECR 位域

49.5.14 PSR (0x44)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD									TDCV							RSVD	PXE	RFDF	RBRS	RESI	DLEC			BO	EW	EP	ACT		LEC		
N/A									R							N/A	X	X	X	X	S			R	R	R	R		S		
x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	x	0	0	0	0	1	1	1	0	0	0	0	0	1	1	1

PSR [31:0]

位域	名称	描述
22-16	TDCV	发送器延迟补偿值 第二采样点的位置，由从 m_can_tx 到 m_can_rx 之间测得的延迟与 TDCR.TDCO 之和定义。在数据阶段，SSP 位置为已发送位起点与第二采样点之间的最小时间片 (mtq) 数。有效值为 0 到 127 个 mtq。
14	PXE	协议异常事件 0：自上次读访问起未发生协议异常事件 1：已发生协议异常事件 注：字节访问：读字节 0 将重置 PXE，读字节 3/2/1 没有影响。
13	RFDF	已接收 FDCAN 消息 此位的设置与验收过滤无关。 0：由于此位是通过 CPU 复位的，因此尚未接收到 FDCAN 消息 1：已接收到 EDL 标志置“1”的 FDCAN 格式的消息 访问类型为 RX：进行读访问时复位。 注：字节访问：读字节 0 将重置 RFDF，读字节 3/2/1 没有影响。
12	RBRS	上次接收的 FDCAN 消息的 BRS 标志 此位与 RFDF 一起设置，与验收过滤无关。 0：上次接收的 FDCAN 消息的 BRS 标志未置“1” 1：上次接收的 FDCAN 消息的 BRS 标志已置“1” 注：字节访问：读字节 0 将重置 RBRS，读字节 3/2/1 没有影响。
11	RESI	上次接收的 FDCAN 消息的 ESI 标志 此位与 RFDF 一起设置，与验收过滤无关。 0：上次接收的 FDCAN 消息的 ESI 标志未置“1” 1：上次接收的 FDCAN 消息的 ESI 标志已置“1” 注：字节访问：读字节 0 将重置 RESI，读字节 3/2/1 没有影响。
10-8	DLEC	数据上一错误代码 在 BRS 标志置“1”的 FDCAN 格式帧数据阶段发生的上一错误类型。代码与 LEC 相同。当 BRS 标志置“1”的 FDCAN 格式帧已无错传输（接收或发送）时，此位域将被清零。 访问类型为 RS：进行读访问时置“1”。 注：字节访问：读字节 0 将置位 DLEC 为”111”，读字节 3/2/1 没有影响。
7	BO	Bus_Off 状态 0：FDCAN 未处于 Bus_Off 状态 1：FDCAN 处于 Bus_Off 状态
6	EW	警告状态 0：两个错误计数器的值均小于 Error_Warning 限值 96 1：至少有一个错误计数器已达到 Error_Warning 限值 96

位域	名称	描述
5	EP	错误被动 0：FDCAN 处于 Error_Active 状态。此位通常会参与总线通信，并会在检测到错误后发送 主动错误标志 1：FDCAN 处于 Error_Passive 状态
4-3	ACT	活动 监控模块的 CAN 通信状态。 00：同步中：节点在 CAN 通信时同步 01：空闲：节点既不是接收器，也不是发送器 10：接收器：节点作为接收器工作 11：发送器：节点作为发送器工作 注：ACT 被协议异常事件设置为“00”。

位域	名称	描述
2-0	LEC	上一错误代码 LEC 指示 CAN 总线上发生的上一错误的类型。消息已无错传输后（接收或发送），此位域将被清零。 000：无错：自消息成功接收或发送而将 LEC 复位后，未发生任何错误。 001：内容错误：在已接收的消息中，有一部分连续出现 5 个以上的相等位，这种情况是不允许发生的。 010：格式错误：已接收帧的固定格式部分的格式不正确。 011：AckError：由 FDCAN 发送的消息未被另一节点确认。 100：Bit1Error：在消息发送过程中（仲裁字段例外），器件希望发送隐性电平（位逻辑值为“1”），但受监控的总线值为显性。 101：Bit0Error：在消息发送过程中（或者确认位，或者主动错误标志，或者过载标志），器件希望发送显性电平（数据或标识符位逻辑值“0”），但受监控的总线值为隐 性。在 Bus_Off 恢复期间，每次监控到 11 个隐性位组成的序列时，此状态都会置“1”。这样，CPU 便可监控 Bus_Off 恢复序列的进行（指示总线并未保持显性或持续受到干扰）。 110：CRCError：已接收消息的 CRC 校验和不正确。传入消息的 CRC 与通过已接收数据计算出的 CRC 不匹配。 111：NoChange：如果对协议状态寄存器进行读访问，则会将 LEC 重新初始化为“7”。如果 LEC 显示的值为“7”，则自 CPU 上一次对协议状态寄存器进行读访问起，未检测到任何 CAN 总线事件。访问类型为 RS：进行读访问时置“1”。 注：如果采用 FDCAN 格式的帧已到达 BRS 标志置 1 的数据阶段，下一 CAN 事件（错误或有效帧）将显示在 FLEC 中，而不显示在 LEC 中。FDCAN CRC 序列中的固定内容位中的错误将显示为格式错误，而不是内容错误 注：Bus_Off 恢复序列（请参见 CAN 规范第 2.0 版或 ISO11898-1）不能通过将 CCCR[INIT] 置 1 或复位来缩短。如果器件进入 Bus_Off 状态，则会将其自身的 CCCR.INIT 置 1，从而会停止所有总线活动。 一旦 CCCR[INIT] 由 CPU 清零，器件随后将等待总线空闲状态出现 129 次（129 × 11 个连续隐性位），然后才会恢复正常操作。Bus_Off 恢复序列结束时，错误管理计数器将复位。 在 CCCR[INIT] 复位后的等待时间内，每次监控到由 11 个隐性位组成的序列时，Bit0 错误代码都会写入 PSR[LEC] 中，从而使 CPU 能够准备好检查 CAN 总线保持显性还是持续受到干扰，并能够监控 Bus_Off 恢复序列。ECR[REC] 用于对这些序列进行计数。 注：字节访问：读字节 0 将置位 LEC 为”111”，读字节 3/2/1 没有影响。

位域	名称	描述

PSR 位域

49.5.15 TDCR (0x48)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																	TDCO							RSVD	TDCF						
N/A																	RW							N/A	RW						
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	x	0	0	0	0	0	0	0

TDCR [31:0]

位域	名称	描述
14-8	TDCO	发送器延迟补偿偏移 定义从 FDCAN_TX 到 FDCAN_RX 测得的延迟与第二采样点之间距离的偏移值。有效值为 0 到 127 个 mtq。
6-0	TDCF	发送器延迟补偿过滤器窗口长度 定义 SSP 位置的最小值，要测量发送器延迟，会忽略 m_can_rx 上会导致 SSP 位置提前的显性边沿。

TDCR 位域

49.5.16 IR (0x50)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD		ARA	PED	PEA	WDI	BO	EW	EP	ELO	BEU	BEC	DRX	TOO	MRAF	TSW	TEFL	TEFF	TEFW	TEFN	TFE	TCF	TC	HPM	RF1L	RF1F	RF1W	RF1N	RF0L	RF0F	RF0W	RF0N
N/A		RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

IR [31:0]

位域	名称	描述
29	ARA	访问保留地址 0：未对保留地址进行访问 1：已对保留地址进行访问
28	PED	数据阶段的协议错误 0：数据阶段没有协议错误 1：检测到数据阶段有协议错误（PSR.DLEC 不是 0、7）
27	PEA	仲裁阶段中的协议错误 0：仲裁阶段中没有协议错误 1：检测到仲裁阶段有协议错误（PSR.LEC 不是 0、7）

位域	名称	描述
26	WDI	看门狗中断 0：未发生消息 RAM 看门狗事件 1：因 READY 缺失而发生消息 RAM 看门狗事件
25	BO	Bus_Off 状态 0：Bus_Off 状态未更改 1：Bus_Off 状态已更改
24	EW	警告状态 0：Error_Warning 状态未更改 1：Error_Warning 状态已更改
23	EP	错误被动 0：Error_Passive 状态未更改 1：Error_Passive 状态已更改
22	ELO	错误记录溢出 0：CAN 错误记录计数器未溢出 1：CAN 错误记录计数器已溢出
21	BEU	位错误未校正 检测到 RAM 位错误并未校正。由连接到 Message RAM 的可选外部奇偶校验/ ECC 逻辑生成的输入信号 m_can_aeim_berr[1] 控制。未校正的消息 RAM 比特错误将 CCCR.INIT 置“1”。这样做是为了避免传输损坏的数据。 0：从消息 RAM 读取时没有检测到比特错误 1：检测到比特错误，未校正 (例如奇偶校验逻辑)
20	BEC	错位校正 检测到 RAM 位错误并已校正。由连接到 Message RAM 的可选外部奇偶校验/ ECC 逻辑生成的输入信号 m_can_aeim_berr[0] 控制。 0：从消息 RAM 读取时没有检测到比特错误 1：检测到比特错误，未校正 (例如 ECC)
19	DRX	消息存储到专用接收缓冲区 接收到的消息已存储到专用接收缓冲区后，此标志会置“1”。 0：未更新接收缓冲区 1：至少有一条已接收消息存储到接收缓冲区中
18	TOO	发生超时 0：无超时 1：达到超时

位域	名称	描述
17	MRAF	消息 RAM 访问失败 此标志在以下情况下会置“1”： 1. 当接收处理单元未在已接收之后的消息的仲裁字段之前完成对已接收消息的验收过滤或存储时。在这种情况下，验收过滤或消息存储会中止，接收处理单元会开始处理之后的消息。 2. 当接收处理单元无法向消息 RAM 写入消息时。在这种情况下，会中止消息存储。 在这两种情况下，FIFO 放入索引不会更新，专用接收缓冲区的新数据标志也不会置“1”。下一条消息存储到此位置时，部分存储的消息会被覆盖。 当发送处理单元无法及时从消息 RAM 中读取数据时，此标志也会置“1”。在这种情况下，消息发送会中止。如果发送处理单元访问失败，M_TTCAN 会切换到受限工作模式（请参见受限工作模式）。要退出受限工作模式，主机 CPU 必须复位 CCCR.ASM。 0：未发生消息 RAM 访问失败 1：发生消息 RAM 访问失败
16	TSW	时间戳回卷 0：时间戳未回卷 1：时间戳已回卷
15	TEFL	发送事件 FIFO 元素丢失 0：发送事件 FIFO 元素未丢失 1：发送事件 FIFO 元素已丢失，尝试向大小为零的发送事件 FIFO 进行写入时也会置“1”
14	TEFF	发送事件 FIFO 已满 0：发送事件 FIFO 未满 1：发送事件 FIFO 已满
13	TEFW	达到发送事件 FIFO 水印 0：发送事件 FIFO 填充级别低于水印 1：发送事件 FIFO 填充级别达到水印
12	TEFN	发送事件 FIFO 新条目 0：发送事件 FIFO 未更改 1：发送处理单元写入发送事件 FIFO 元素
11	TFE	发送 FIFO 为空 0：发送 FIFO 非空 1：发送 FIFO 为空
10	TCF	发送取消完成 0：发送取消未完成 1：发送取消已完成
9	TC	发送完成 0：发送未完成 1：发送已完成

位域	名称	描述
8	HPM	高优先级消息 0：未接收到任何高优先级消息 1：已接收到高优先级消息
7	RF1L	接收 FIFO 1 消息丢失 0：接收 FIFO 1 消息未丢失 1：接收 FIFO 1 消息已丢失，尝试向大小为零的接收 FIFO 1 进行写入后也会置“1”
6	RF1F	接收 FIFO 1 已满 0：接收 FIFO 1 未满 1：接收 FIFO 1 已满
5	RF1W	达到接收 FIFO 1 水印 0：接收 FIFO 1 填充级别低于水印 1：接收 FIFO 1 填充级别达到水印
4	RF1N	接收 FIFO 1 新消息 0：未向接收 FIFO 1 写入新消息 1：已向接收 FIFO 1 写入新消息
3	RF0L	接收 FIFO 0 消息丢失 0：接收 FIFO 0 消息未丢失 1：接收 FIFO 0 消息已丢失，尝试向大小为零的接收 FIFO 0 进行写入后也会置“1”
2	RF0F	接收 FIFO 0 已满 0：接收 FIFO 0 未满 1：接收 FIFO 0 已满
1	RF0W	达到接收 FIFO 0 水印 0：接收 FIFO 0 填充级别低于水印 1：接收 FIFO 0 填充级别达到水印
0	RF0N	接收 FIFO 0 新消息 0：未向接收 FIFO 0 写入新消息 1：已向接收 FIFO 0 写入新消息

IR 位域

49.5.17 IE (0x54)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD		ARAE	PEDE	PEAE	WDIE	BOE	EWE	EPE	ELOE	BEUE	BECE	DRXE	TOOE	MRAFE	TSWE	TEFLE	TEFFE	TEFWE	TEFNE	TFEE	TCFE	TCE	HPME	RF1LE	RF1FE	RF1WE	RF1NE	RF0LE	RF0FE	RF0WE	RF0NE
N/A		RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

IE [31:0]

位域	名称	描述
29	ARAE	访问保留地址使能
28	PEDE	数据阶段的协议错误使能
27	PEAE	仲裁阶段中的协议错误使能
26	WDIE	看门狗中断使能 0：禁止中断 1：使能中断
25	BOE	Bus_Off 状态 0：禁止中断 1：使能中断
24	EWE	警告状态中断使能 0：禁止中断 1：使能中断
23	EPE	错误被动中断使能 0：禁止中断 1：使能中断
22	ELOE	错误记录溢出中断使能 0：禁止中断 1：使能中断
21	BEUE	位错误未更正中断使能 0：禁止中断 1：使能中断
20	BECE	位错误已更正中断使能 0：禁止中断 1：使能中断
19	DRXE	消息存储到专用接收缓冲区中断使能 0：禁止中断 1：使能中断
18	TOOE	超时已发生中断使能 0：禁止中断 1：使能中断
17	MRAFE	消息 RAM 访问失败中断使能 0：禁止中断 1：使能中断
16	TSWE	时间戳回卷中断使能 0：禁止中断 1：使能中断
15	TEFLE	发送事件 FIFO 元素丢失中断使能 0：禁止中断 1：使能中断

位域	名称	描述
14	TEFFE	发送事件 FIFO 已满中断使能 0：禁止中断 1：使能中断
13	TEFWE	达到发送事件 FIFO 水印中断使能 0：禁止中断 1：使能中断
12	TEFNE	发送事件 FIFO 新条目中断使能 0：禁止中断 1：使能中断
11	TFEE	发送 FIFO 为空中断使能 0：禁止中断 1：使能中断
10	TCFE	发送取消完成中断使能 0：禁止中断 1：使能中断
9	TCE	发送完成中断使能 0：禁止中断 1：使能中断
8	HPME	高优先级消息中断使能 0：禁止中断 1：使能中断
7	RF1LE	接收 FIFO 1 消息丢失中断使能 0：禁止中断 1：使能中断
6	RF1FE	接收 FIFO 1 已满中断使能 0：禁止中断 1：使能中断
5	RF1WE	达到接收 FIFO 1 水印中断使能 0：禁止中断 1：使能中断
4	RF1NE	接收 FIFO 1 新消息中断使能 0：禁止中断 1：使能中断
3	RF0LE	接收 FIFO 0 消息丢失中断使能 0：禁止中断 1：使能中断
2	RF0FE	接收 FIFO 0 已满中断使能 0：禁止中断 1：使能中断

位域	名称	描述
1	RF0WE	达到接收 FIFO 0 水印中断使能 0：禁止中断 1：使能中断
0	RF0NE	接收 FIFO 0 新消息中断使能 0：禁止中断 1：使能中断

IE 位域

49.5.18 ILS (0x58)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD		ARAL	PEDL	PEAL	WDIL	BOL	EWL	EPL	ELOL	BEUL	BECL	DRXL	TOOL	MRAFL	TSWL	TEFLL	TEFFL	TEFWL	TEFNL	TFEL	TCFL	TCL	HPML	RF1LL	RF1FL	RF1WL	RF1NL	RF0LL	RF0FL	RF0WL	RF0NL
N/A		RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ILS [31:0]

位域	名称	描述
29	ARAL	访问保留地址线
28	PEDL	数据阶段的协议错误线
27	PEAL	仲裁阶段的协议错误线
26	WDIL	看门狗中断线
25	BOL	Bus_Off 状态中断线
24	EWL	警告状态中断线
23	EPL	错误被动中断线
22	ELOL	错误记录溢出中断线
21	BEUL	位错误未更正中断线
20	BECL	位错误已更正中断线
19	DRXL	消息存储到专用接收缓冲区中断线
18	TOOL	超时已发生中断线
17	MRAFL	消息 RAM 访问失败中断线
16	TSWL	时间戳回卷中断线
15	TEFLL	发送事件 FIFO 元素丢失中断线
14	TEFFL	发送事件 FIFO 已满中断线
13	TEFWL	达到发送事件 FIFO 水印中断线
12	TEFNL	发送事件 FIFO 新条目中断线
11	TFEL	发送 FIFO 为空中断线
10	TCFL	发送取消完成中断线
9	TCL	发送完成中断线
8	HPML	高优先级消息中断线

位域	名称	描述
7	RF1LL	接收 FIFO 1 消息丢失中断线
6	RF1FL	接收 FIFO 1 已满中断线
5	RF1WL	达到接收 FIFO 1 水印中断线
4	RF1NL	接收 FIFO 1 新消息中断线
3	RF0LL	接收 FIFO 0 消息丢失中断线
2	RF0FL	接收 FIFO 0 已满中断线
1	RF0WL	达到接收 FIFO 0 水印中断线
0	RF0NL	接收 FIFO 0 新消息中断线

ILS 位域

49.5.19 ILE (0x5C)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																														EINT1	EINT0
N/A																														RW	RW
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0

ILE [31:0]

位域	名称	描述
1	EINT1	使能中断线 1 0：禁止中断线 fdcan_intr0_it 1：使能中断线 fdcan_intr0_it
0	EINT0	使能中断线 0 0：禁止中断线 fdcan_intr1_it 1：使能中断线 fdcan_intr1_it

ILE 位域

49.5.20 GFC (0x80)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																										ANFS		ANFE		RRFS	RRFE
N/A																										RW		RW		RW	RW
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0

GFC [31:0]

位域	名称	描述
5-4	ANFS	接受非匹配标准帧 定义了如何处理与过滤器列表的任何元素都不匹配的 ID 为 11 位的已接收消息。 00：在接收 FIFO 0 中接受 01：在接收 FIFO 1 中接受 10：拒绝 11：拒绝
3-2	ANFE	接受非匹配扩展帧 定义了如何处理与过滤器列表的任何元素都不匹配的 ID 为 29 位的已接收消息。 00：在接收 FIFO 0 中接受 01：在接收 FIFO 1 中接受 10：拒绝 11：拒绝
1	RRFS	拒绝标准远程帧 0：过滤采用 11 位标准 ID 的远程帧 1：拒绝所有采用 11 位标准 ID 的远程帧
0	RRFE	拒绝扩展远程帧 0：过滤采用 29 位标准 ID 的远程帧 1：拒绝所有采用 29 位标准 ID 的远程帧

GFC 位域

49.5.21 SIDFC (0x84)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD								LSS								FLSSA														RSVD	
N/A								RW								RW														N/A	
x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	x	x

SIDFC [31:0]

位域	名称	描述
23-16	LSS	标准列表大小 0：无标准消息 ID 过滤器 1-128：标准消息 ID 过滤器元素数量 >128：大于 128 的值会被解析为 128。
15-2	FLSSA	标准过滤器列表起始地址 标准消息 ID 过滤器列表的起始地址（32 位字地址，请参见 XXXX：标准消息 ID 过滤器元素）。

SIDFC 位域

49.5.22 XIDFC (0x88)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD									LSE							FLESA														RSVD	
N/A									RW							RW														N/A	
x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	x	x

XIDFC [31:0]

位域	名称	描述
22-16	LSE	扩展列表大小 0：无标准消息 ID 过滤器 1-64：标准消息 ID 过滤器元素数量 >64：大于 64 的值会被解析为 64。
15-2	FLESA	扩展过滤器列表起始地址 标准消息 ID 过滤器列表的起始地址（32 位字地址，请参见 XXXX：扩展消息 ID 过滤器元素）。

XIDFC 位域

49.5.23 XIDAM (0x90)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD			EIDM																												
N/A			RW																												
x	x	x	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

XIDAM [31:0]

位域	名称	描述
28-0	EIDM	扩展 ID 掩码 要对扩展帧进行验收过滤，会将扩展 ID 与掩码与已接收帧的消息 ID 进行与运算。用于屏蔽 SAE J1939 中的 29 位 ID。如果所有位的复位值均设为“1”，则掩码无效。

XIDAM 位域

49.5.24 HPMS (0x94)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																FLST	FIDX							MSI		BIDX					
N/A																R	R							R		R					
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

HPMS [31:0]

位域	名称	描述
15	FLST	过滤器列表 0：标准过滤器列表 1：扩展过滤器列表
14-8	FIDX	过滤器索引 匹配过滤器元素的索引。范围为 0 到 SIDFC[LSS] - 1 或 XIDFC[LSE] - 1。
7-6	MSI	消息存储指示符 00：未选择 FIFO 01：FIFO 上溢 10：消息存储在 FIFO 0 中 11：消息存储在 FIFO 1 中
5-0	BIDX	缓冲区索引 消息存储到的接收 FIFO 元素的索引。仅当 MSI[1] = “1” 时有效。

HPMS 位域

49.5.25 NDAT1 (0x98)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ND1																															
RW																															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

NDAT1 [31:0]

位域	名称	描述
31-0	ND1	新数据 [31:0] 该寄存器保存接收缓冲区 0 到 31 的新数据标志。当相应的接收缓冲区通过已接收的帧进行更新时，这些标志会置 1。在主机将标志清零之前，标志保持置 “1” 状态。通过向对应位位置写入 “1” 将标志清零。写入 “0” 不会起任何作用。硬复位会将寄存器清零。 0：接收缓冲区未更新 1：接收缓冲区通过新消息更新

位域	名称	描述

NDAT1 位域

49.5.26 NDAT2 (0x9C)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ND2																															
RW																															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

NDAT2 [31:0]

位域	名称	描述
31-0	ND2	新数据 [63:32] 该寄存器保存接收缓冲区 32 到 63 的新数据标志。当相应的接收缓冲区通过已接收的帧进 行更新时，这些标志会置 1。在主机将标志清零之前，标志保持置 1 状态。通过向对应位位置写入“1”将标志清零。写入“0”不会起任何作用。硬复位会将寄存器清零。 0：接收缓冲区未更新 1：接收缓冲区通过新消息更新

NDAT2 位域

49.5.27 RXF0C (0xA0)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
F0OM	F0WM							RSVD	F0S							F0SA														RSVD	
RW	RW							N/A	RW							RW														N/A	
0	0	0	0	0	0	0	0	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	x	x

RXF0C [31:0]

位域	名称	描述
31	F0OM	FIFO 0 工作模式 FIFO 0 可工作在阻止模式或覆盖模式下。 0：FIFO 0 处于阻止模式 1：FIFO 0 处于覆盖模式

位域	名称	描述
30-24	F0WM	FIFO 0 水印 0：禁止水印中断 1-64：接收 FIFO 0 水印中断 (IR[RF0W]) 的级别 >64：禁止水印中断
22-16	F0S	接收 FIFO 0 大小 0：无接收 FIFO 0 1-64：接收 FIFO 0 元素数 >64：大于 64 的值会被解析为 64 接收 FIFO 0 元素的索引为 0 到 F0S-1。
15-2	F0SA	接收 FIFO 0 起始地址 消息 RAM 中接收 FIFO 0 的起始地址（32 位字地址，请参见 XXXX：消息 RAM 配置）。

RXF0C 位域

49.5.28 RXF0S (0xA4)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD						RF0L	F0F	RSVD		F0PI						RSVD		F0GI						RSVD	F0FL						
N/A						R	R	N/A		R						N/A		R						N/A	R						
x	x	x	x	x	x	0	0	x	x	0	0	0	0	0	0	x	x	0	0	0	0	0	0	x	0	0	0	0	0	0	0

RXF0S [31:0]

位域	名称	描述
25	RF0L	接收 FIFO 0 消息丢失 此位是中断标志 IR[RF0L] 的副本。当 IR[RF0L] 复位时，此位也会复位。 0：接收 FIFO 0 消息未丢失 1：接收 FIFO 0 消息已丢失，尝试向大小为零的接收 FIFO 0 进行写入后也会置“1”。 注：当 RXF0C.F0OM=’1’ 时重写最旧消息将不会置位该标志位。
24	F0F	接收 FIFO 0 已满 0：接收 FIFO 0 未满 1：接收 FIFO 0 已满
21-16	F0PI	接收 FIFO 0 放入索引 接收 FIFO 0 写入索引指针，范围为 0 到 63。
13-8	F0GI	接收 FIFO 0 获取索引 接收 FIFO 0 读取索引指针，范围为 0 到 63。
6-0	F0FL	接收 FIFO 0 填充级别 接收 FIFO 0 中存储的元素数，范围为 0 到 64。

位域	名称	描述

RXF0S 位域

49.5.29 RXF0A (0xA8)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																										F0AI					
N/A																										RW					
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0

RXF0A [31:0]

位域	名称	描述
5-0	F0AI	接收 FIFO 0 确认索引 主机从接收 FIFO 0 读取消息或消息序列后，必须将从接收 FIFO 0 读取的最后一个元素的缓冲区索引写入 F0AI 中。此操作会将接收 FIFO 0 获取索引 RXF0S[F0GI] 设为 F0AI + 1，并会更新 FIFO 0 填充级别 RXF0S[F0FL]。

RXF0A 位域

49.5.30 RXBC (0xAC)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																RBSA														RSVD	
N/A																RW														N/A	
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	x	x

RXBC [31:0]

位域	名称	描述
15-2	RBSA	接收缓冲区起始地址 配置消息 RAM 中接收缓冲区部分的起始地址（32 位字地址）。此外，也用于引用调试消息 A、B、C。

RXBC 位域

49.5.31 RXF1C (0xB0)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
F1OM	F1WM							RSVD	F1S							F1SA														RSVD	
RW	RW							N/A	RW							RW														N/A	
0	0	0	0	0	0	0	0	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	x	x

RXF1C [31:0]

位域	名称	描述
31	F1OM	FIFO 1 工作模式 FIFO 1 可工作在阻止模式或覆盖模式下。 0：FIFO 1 处于阻止模式 1：FIFO 1 处于覆盖模式
30-24	F1WM	FIFO 1 水印 0：禁止水印中断 1-64：接收 FIFO 1 水印中断 (IR[RF1W]) 的级别 >64：禁止水印中断
22-16	F1S	接收 FIFO 1 大小 0：无接收 FIFO 1 1-64：接收 FIFO 1 元素数 >64：大于 64 的值会被解析为 64 接收 FIFO 1 元素的索引为 0 到 F1S - 1。
15-2	F1SA	接收 FIFO 1 起始地址 消息 RAM 中接收 FIFO 1 的起始地址（32 位字地址，请参见 XXXX：消息 RAM 配置）。

RXF1C 位域

49.5.32 RXF1S (0xB4)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMS	RSVD					RF1L	F1F	RSVD		F1PI						RSVD		F1GI						RSVD	F1FL						
R	N/A					R	R	N/A		R						N/A		R						N/A	R						
0	0	x	x	x	x	0	0	x	x	0	0	0	0	0	0	x	x	0	0	0	0	0	0	x	0	0	0	0	0	0	0

RXF1S [31:0]

位域	名称	描述
31-30	DMS	调试消息状态 00：空闲状态，等待接收调试消息，DMA 请求清零 01：已接收调试消息 A 10：已接收调试消息 A、B 11：已接收调试消息 A、B、C，DMA 请求置“1”
25	RF1L	接收 FIFO 1 消息丢失 此位是中断标志 IR[RF1L] 的副本。当 IR[RF1L] 复位时，此位也会复位。 0：接收 FIFO 1 消息未丢失 1：接收 FIFO 1 消息已丢失，尝试向大小为零的接收 FIFO 1 进行写入后也会置“1”。 注：当 RXF1C.F1OM=’1’ 时重写最旧消息将不会置位该标志位。
24	F1F	接收 FIFO 1 已满 0：接收 FIFO 1 未满 1：接收 FIFO 1 已满
21-16	F1PI	接收 FIFO 1 放入索引 接收 FIFO 1 写入索引指针，范围为 0 到 63。
13-8	F1GI	接收 FIFO 1 获取索引 接收 FIFO 1 读取索引指针，范围为 0 到 63。
6-0	F1FL	接收 FIFO 1 填充级别 接收 FIFO 1 中存储的元素数，范围为 0 到 64。

RXF1S 位域

49.5.33 RXF1A (0xB8)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																										F1AI					
N/A																										RW					
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0

RXF1A [31:0]

位域	名称	描述
5-0	F1AI	接收 FIFO 1 确认索引 主机从接收 FIFO 1 读取消息或消息序列后，必须将从接收 FIFO 1 读取的最后一个元素的缓冲区索引写入 F1AI 中。此操作会将接收 FIFO 1 获取索引 RXF1S[F1GI] 设为 F1AI + 1，并会更新 FIFO 1 填充级别 RXF1S[F1FL]。

RXF1A 位域

49.5.34 RXESC (0xBC)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																					RBDS			RSVD	F1DS			RSVD	F0DS		
N/A																					RW			N/A	RW			N/A	RW		
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	x	0	0	0	x	0	0	0

RXESC [31:0]

位域	名称	描述
10-8	RBDS	接收缓冲区数据字段大小 000：8 字节数据字段 001：12 字节数据字段 010：16 字节数据字段 011：20 字节数据字段 100：24 字节数据字段 101：32 字节数据字段 110：48 字节数据字段 111：64 字节数据字段
6-4	F1DS	接收 FIFO 1 数据字段大小 000：8 字节数据字段 001：12 字节数据字段 010：16 字节数据字段 011：20 字节数据字段 100：24 字节数据字段 101：32 字节数据字段 110：48 字节数据字段 111：64 字节数据字段
2-0	F0DS	接收 FIFO 0 数据字段大小 000：8 字节数据字段 001：12 字节数据字段 010：16 字节数据字段 011：20 字节数据字段 100：24 字节数据字段 101：32 字节数据字段 110：48 字节数据字段 111：64 字节数据字段 注：注意: 如果接受的 CAN 帧的数据字段大小超过了匹配 Rx Buffer 或 Rx FIFO 配置的数据字段大小，只有 RXESC 配置的字节数被存储到 Rx Buffer 响应中。Rx FIFO 元素。帧的其余数据字段将被忽略。

RXESC 位域

49.5.35 TXBC (0xC0)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD	TFQM	TFQS						RSVD		NDTB						TBSA														RSVD	
N/A	RW	RW						N/A		RW						RW														N/A	
x	0	0	0	0	0	0	0	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	x	x

TXBC [31:0]

位域	名称	描述
30	TFQM	发送 FIFO/队列模式 0：发送 FIFO 操作 1：发送队列操作
29-24	TFQS	发送 FIFO/队列大小 0：无发送 FIFO/队列 1-32：用于发送 FIFO/队列的发送缓冲区数 >32：大于 32 的值会被解析为 32。
21-16	NDTB	专用发送缓冲区数 1-32：专用发送缓冲区数 >32：大于 32 的值会被解析为 32。
15-2	TBSA	发送缓冲区起始地址 消息 RAM 中发送缓冲区部分的起始地址（32 位字地址，请参见图 xxx）。 注:TFQS 和 NDTB 之和不能大于 32。没有检查错误配置。消息 RAM 中的发送缓冲区部分从专用的发送缓冲区开始。

TXBC 位域

49.5.36 TXFQS (0xC4)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD										TFQF	TFQPI					RSVD			TFGI					RSVD		TFFL					
N/A										R	R					N/A			R					N/A		R					
x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	x	x	x	0	0	0	0	0	x	x	0	0	0	0	0	0

TXFQS [31:0]

位域	名称	描述
21	TFQF	发送 FIFO/队列已满 0：发送 FIFO/队列未满 1：发送 FIFO/队列已满
20-16	TFQPI	发送 FIFO/队列放入索引 发送 FIFO/队列写入索引指针，范围为 0 到 31。
12-8	TFGI	发送 FIFO 获取索引 发送 FIFO 读取索引指针，范围为 0 到 31。如果已配置发送队列操作（TXBC.TFQM = “1”），则读为零
5-0	TFFL	发送 FIFO 空闲级别 从 TFGI 开始的连续空闲发送 FIFO 元素数，范围为 0 到 32。如果已配置发送队列操作（TXBC[TFQM] = “1”），则读为零 注：如果是专用发送缓冲区与发送 FIFO 或发送队列相结合的混合配置，放入和获取索引指示从第一个专用发送缓冲区开始的发送缓冲区数。例如：对于 12 个专用发送缓冲区与包含 20 个缓冲区的发送 FIFO 相结合的混合配置，放入索引 15 会指向发送 FIFO 的第四个缓冲区。

TXFQS 位域

49.5.37 TXESC (0xC8)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																													TBDS		
N/A																													RW		
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0

TXESC [31:0]

位域	名称	描述
2-0	TBDS	发送缓冲区数据字段大小 000：8 字节数据字段 001：12 字节数据字段 010：16 字节数据字段 011：20 字节数据字段 100：24 字节数据字段 101：32 字节数据字段 110：48 字节数据字段 111：64 字节数据字段

TXESC 位域

49.5.38 TXBRP (0xCC)

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

TXBRP [31:0]

位域	名称	描述
31-0	TRP	发送请求挂起 每个发送缓冲区都有自己的发送请求挂起位。这些位通过寄存器 TXBAR 置“1”。请求的发 送已完成或已通过 TXBCR 取消后，这些位会复位。TXBRP 位仅会为通过 TXBC 配置的发送缓冲区置“1”。TXBRP 位置 1 后，会启动发送扫描（请参见过滤调试消息）以检查优先级最高（消息 ID 最小的发送缓冲区）的挂起发送 请求。取消请求会使寄存器 TXBRP 的相应发送请求挂起位复位。如果请求取消时发送已开始进行，则无论发送是否成功，都会在发送结束时执行复位操作。相应的 TXBRP 位复位后，取消请求位也会立即复位。请求取消后，会在下列情况下通过 TXBCF 指示完成取消 -与相应的 TXBTO 位一起成功发送后 -取消时发送尚未开始 -发送因仲裁丢失而中止 -帧发送过程中出错 在 DAR 模式下，所有发送在发送失败后都会自动取消。对于所有未成功的发送，相应的 TXBCF 位会置“1”。 0：无发送请求挂起 1：发送请求挂起 注：如果 TXBRP 位在正在进行发送扫描时置“1”，则进行该特殊发送扫描期间不会考虑此位。如果请求取消此类发送缓冲区，则该添加请求会立即取消，相应的 TXBRP 位会复位。

TXBRP 位域

49.5.39 TXBAR (0xD0)

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

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

TXBAR [31:0]

位域	名称	描述
31-0	AR	添加请求 每个发送缓冲区都有自己的添加请求位。写入“1”会将相应的添加请求位置“1”；写入“0”没有影响。这样，主机通过对 TXBAR 进行一次写操作便可为多个发送缓冲区设置发送请求。仅会为通过 TXBC 配置的发送缓冲区设置 TXBAR 位。如果没有运行发送扫描，这些位会立即复位，否则在发送扫描过程完成之前，这些位保持置“1”状态。 0：未添加发送请求 1：已添加发送请求 注：如果添加请求应用到具有挂起发送请求的发送缓冲区（相应的 TXBRP 位已置 1），则会忽略请求。

TXBAR 位域

49.5.40 TXBCR (0xD4)

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

TXBCR [31:0]

位域	名称	描述
31-0	CR	取消请求 每个发送缓冲区都有自己的取消请求位。写入“1”会将相应的取消请求位置“1”；写入“0”没有影响。这样，主机通过对 TXBCR 进行一次写操作便可为多个发送缓冲区设置取消请求。仅会为通过 TXBC 配置的发送缓冲区设置 TXBCR 位。在相应的 TXBRP 位复位之前，这些位保持置“1”状态。 0：无取消挂起 1：取消挂起

TXBCR 位域

49.5.41 TXBTO (0xD8)

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

TXBTO [31:0]

位域	名称	描述
31-0	TO	发送已发生 每个发送缓冲区都有自己的发送已发生位。当相应的 TXBRP 位在发送成功后清零时，这些位会置“1”。通过向寄存器 TXBAR 的相应位写入“1”请求进行新发送时，这些位会复位。 0：未进行发送 1：已进行发送

TXBTO 位域

49.5.42 TXBCF (0xDC)

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

TXBCF [31:0]

位域	名称	描述
31-0	CF	取消完成 每个发送缓冲区都有自己的取消完成位。当相应的 TXBRP 位在通过 TXBCR 请求取消后清零时，这些位会置“1”。如果相应的 TXBRP 位在取消时未置“1”，CF 会立即置“1”。通过向寄存器 TXBAR 的相应位写入“1”请求进行新发送时，这些位会复位。 0：无发送缓冲区取消 1：发送缓冲区取消完成

TXBCF 位域

49.5.43 TXBTIE (0xE0)

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

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

TXBTIE [31:0]

位域	名称	描述
31-0	TIE	发送中断使能 每个发送缓冲区都有自己的发送中断使能位。 0：禁止发送中断 1：使能发送中断

TXBTIE 位域

49.5.44 TXBCIE (0xE4)

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

TXBCIE [31:0]

位域	名称	描述
31-0	CFIE	取消完成中断使能 每个发送缓冲区都有自己的取消完成中断使能位。 0：禁止取消完成中断 1：使能取消完成中断

TXBCIE 位域

49.5.45 TXEFC (0xF0)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD		EFWM						RSVD		EFS						EFSA														RSVD	
N/A		RW						N/A		RW						RW														N/A	
x	x	0	0	0	0	0	0	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	x	x

TXEFC [31:0]

位域	名称	描述
29-24	EFWM	事件 FIFO 水印 0：禁止水印中断 1-32：发送事件 FIFO 水印中断 (IR[TEFW]) 的级别 >32：禁止水印中断
21-16	EFS	事件 FIFO 大小 0：禁止发送事件 FIFO 1-32：发送事件 FIFO 元素数 >32：大于 32 的值会被解析为 32 发送事件 FIFO 元素的索引为 0 到 EFS-1。
15-2	EFSA	事件 FIFO 起始地址 消息 RAM 中发送事件 FIFO 的起始地址（32 位字地址，请参见 xxxx：消息 RAM 配置）。

TXEFC 位域

49.5.46 TXEFS (0xF4)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD						TEFL	EFF	RSVD			EFPI					RSVD			EFGI					RSVD		EFFL					
N/A						R	R	N/A			R					N/A			R					N/A		R					
x	x	x	x	x	x	0	0	x	x	x	0	0	0	0	0	x	x	x	0	0	0	0	0	x	x	0	0	0	0	0	0

TXEFS [31:0]

位域	名称	描述
25	TEFL	发送事件 FIFO 元素丢失 此位是中断标志 IR[TEFL] 的副本。当 IR[TEFL] 复位时，此位也会复位。 0：没有发送事件 FIFO 元素丢失 1：发送事件 FIFO 元素已丢失，尝试向大小为零的发送事件 FIFO 进行写入时也会置“1”。
24	EFF	事件 FIFO 已满 0：发送事件 FIFO 未满 1：发送事件 FIFO 已满
20-16	EFPI	事件 FIFO 放入索引 发送事件 FIFO 写入索引指针，范围为 0 到 31。
12-8	EFGI	事件 FIFO 获取索引 发送事件 FIFO 读取索引指针，范围为 0 到 31。
5-0	EFFL	事件 FIFO 填充级别 发送事件 FIFO 中存储的元素数，范围为 0 到 31。

TXEFS 位域

49.5.47 TXEFA (0xF8)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																											EFAI					
N/A																											RW					
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0

TXEFA [31:0]

位域	名称	描述
4-0	EFAI	事件 FIFO 确认索引 主机从发送事件 FIFO 读取元素或元素序列后，必须将从发送事件 FIFO 读取的最后一个元素的索引写入 EFAI 中。此操作会将发送事件 FIFO 获取索引 TXEFS[EFGI] 设为 EFAI + 1，并会更新 FIFO 0 填充级别 TXEFS[EFFL]。

TXEFA 位域

49.5.48 TS_SEL (0x200 + 0x4 * n)

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

TS_SEL [31:0]

位域	名称	描述
31-0	TS	时间戳字。 默认可以保存最多 16 个 32 位的时间戳。 如果设置为使用 64 位模式，则可以保存最多 8 个 64 位时间戳

TS_SEL 位域

49.5.49 CREL (0x240)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
REL				STEP				SUBSTEP				YEAR				MON								DAY							
R				R				R				R				R								R							

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

CREL [31:0]

位域	名称	描述
31-28	REL	发布版本寄存器 一位，BCD 编码。
27-24	STEP	发布版本阶段 一位，BCD 编码。
23-20	SUBSTEP	发布版本子阶段寄存器 一位，BCD 编码。
19-16	YEAR	时间戳年 一位，BCD 编码。
15-8	MON	时间戳月 两位，BCD 编码。
7-0	DAY	时间戳日 两位，BCD 编码。

CREL 位域

49.5.50 TSCFG (0x244)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																TBPRE								RSVD				EN64	SCP	TBCS	TSUE
N/A																RW								N/A				RW	RW	RW	RW
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	x	x	x	x	0	0	0	0

TSCFG [31:0]

位域	名称	描述
15-8	TBPRE	时间基准预分频 0x00-0xFF 振荡器频率以生成时基计数器时钟除以的值。取值范围为 0-255。硬件将该值解析为编程值加 1。仅影响 TSU 内部时基。当合成不使用内部时基时，TBPRE[7:0] 固定为 0x00，则不使用时间戳预分频。
3	EN64	设置为使用 64 位时间戳。 启用时，tsu 最多可以保存 8 个不同的时间戳，TS(k) 和 TS(k+1) 用于一个 64 位时间戳，k 为 0~7。 TSP 可以用来选择不同的一个。

位域	名称	描述
2	SCP	选择捕捉位置 0: 在 EOF 捕获时间戳 1: 在 SOF 捕获时间戳
1	TBCS	时间基准计数器选择 当合成不使用内部时基时，TBCS 固定为‘1’。 0: 从内部时基计数器捕获的时间戳值，TB[31:0] 是内部时间基准计数器 1: 从输入 tsu_tbin[31:0] 捕获的时间戳值，TB[31:0] 是 tsu_tbin[31:0]
0	TSUE	时间戳单元使能 0: 时间戳单元禁止 1: 时间戳单元使能

TSCFG 位域

49.5.51 TSS1 (0x248)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TSL																TSN															
R																R															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

TSS1 [31:0]

位域	名称	描述
31-16	TSL	时间戳丢失 每个时间戳寄存器 (TS0-TS15) 被分配一个比特。这些位是在相关时间戳寄存器中存储的时间戳在读取之前被覆盖时设置的。 读取时间戳寄存器将重置相关的位。
15-0	TSN	新时间戳 每个时间戳寄存器 (TS0-TS15) 被分配一个比特。这些位是在一个时间戳被存储在相关时间戳寄存器时置位的。读取一个时间戳寄存器将会复位相关的位。

TSS1 位域

49.5.52 TSS2 (0x24C)

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

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
N/A																												R			
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0

TSS2 [31:0]

位域	名称	描述
3-0	TSP	时间戳指针 每次捕获一个时间戳，时间戳指针都加 1。从它的最大值 (3,7 或 15 开始，取决于 number_ts_g)，它被递增到 0。 它的值也输出到 m_can_tsp[3:0]。

TSS2 位域

49.5.53 ATB (0x250)

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

ATB [31:0]

位域	名称	描述
31-0	TB	用于时间戳生成 31-0 的时间基准

ATB 位域

49.5.54 ATBH (0x254)

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

ATBH [31:0]

位域	名称	描述
31-0	TBH	用于时间戳生成 63-32 的时间基准

ATBH 位域

49.5.55 GLB_CTL (0x400)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
M_CAN_STBY	STBY_CLR_EN	STBY_POL	RSVD																											TSU_TBIN_SEL	
RW	RW	RW	N/A																											RW	
0	0	0	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0

GLB_CTL [31:0]

位域	名称	描述
31	M_CAN_STBY	m_can 进入 standby 控制位
30	STBY_CLR_EN	m_can standby 清除控制 0: 由软件控制清除 m_can_stby bit[bit31] 1: 当 rxdata 为 0 时硬件自动清除 standby
29	STBY_POL	standby 极性选择
1-0	TSU_TBIN_SEL	外部时间戳选择。每个 CAN 模块有 4 个外部时间戳（可以通过 PTPC 寄存器配置 4 个时间戳来源，可以是来自于 ptpc，也可以来自去其他 CAN 模块） 当 TSCFG.TBCS 位置 1 时，本寄存器用于从 4 个外部时间戳中选择一个作为本地时间戳。

GLB_CTL 位域

49.5.56 GLB_STATUS (0x404)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																												M_CAN_INT1	M_CAN_INT0	RSVD	
N/A																												R	R	N/A	
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	x	x

GLB_STATUS [31:0]

位域	名称	描述
3	M_CAN_INT1	m_can 中断状态位 1
2	M_CAN_INT0	m_can 中断状态位 0

GLB_STATUS 位域

49.6 CAN 配置使用简单说明

49.6.1 软件初始化

CAN 软件可以通过设置 CCCR 寄存器 init 位进行初始化。当 cccr 寄存器的 init 位设置为 1 时，can 总线停止接收和发送数据，输出的 TXD 一直为 1，此时和 cccr 的 cce 位可以一起控制寄存器的读写权限，当有读写权限的时候软件可以进行寄存器的初始化。需要注意的是 cce 这位只有在 INIT 位是 1 的时候才能进行配置. 当 INIT 清 0 的时候，CCE 也会自动的清 0. 下面的寄存器当 CCE 置位的时候会被复位：

- HPMS: 高优先级消息状态
- RXF0S: 接收 FIFO0 状态
- RXF1S: 接收 FIFO1 状态
- TXFQS: 发送 FIFO/队列状态
- TXBRP: 发送请求等待
- TXBTO: 发送传输发生
- TXBCF: 发送传输取消成功
- TXEFS: 发送事件 FIFO 状态

并且下面的两个寄存器只有当 CCE 清 0 之后才能写：

- TXBAR: 发送缓存添加发送请求
- TXBCR: 发送缓存取消发送请求

49.6.2 接收处理

can 控制器提供两种形式的接收过滤器，一种是用标准的标识符，另一种是用扩展的标识符。这些过滤器可以分配给接收缓存或者接收 FIFO0/1, 对于接收过滤器，每个过滤器都从元素 0 开始直到遇到匹配的结束，遇到匹配的元素之后的元素便不在查询匹配。接收过滤器主要的功能为：

- 每个过滤器可以配置为:
  - 一个范围的范围过滤器, 过滤器匹配消息的标识符在寄存器 SF1ID/SF2ID 或者 EF1ID/EF2ID 的一个或两个标识符
  - 筛选一个或两个特定的标识符，此时配置的寄存器 SF1ID=SF2ID, 或者 EF1ID=EF2ID
  - 经典的掩码形式，经典的位掩码过滤使用接收的标识符与位掩码，SF1ID/EF1ID 用作消息过滤器，而 SF2ID/EF2ID 用作过滤掩码。
- 每个过滤器均可配置为匹配后是接收数据或拒绝接收数据
- 每个过滤器可以单独的启用和禁用

根据过滤器的配置（SFEC/EFEC），匹配之后会触发下面的操作之一

- 将接收到的帧放到 FIFO0 或者 FIFO1
- 将接收到的帧放到接收缓存
- 拒绝接收数据帧
- 设置高优先级的消息中断标志寄存器 IR 的 HPM 位
- 设置高优先级的消息中断标志寄存器 IR 的 HPM 位并且将接收到的帧放到 FIFO0 或者 FIFO1

接收过滤器在接收到完整的标识符之后开始进行筛选，筛选完成后，如果是匹配的标识符，那么就将收到的数据帧以 32 位的格式放到接收缓存或者 FIFO 中。如果不接收或者接收到的数据帧检测出来有错误（例如 CRC 校验错误），那么就会丢弃此数据

49.6.3 发送处理

发送处理发送缓存的发送 FIFO 的以及发送队列的发送请求，最多可以设置 32 个发送缓存区用来发送数据，并且每个发送缓存可以单独的配置是普通的 can 总线模式还是 can fd 模式。下表写了具体的配置。

表 210: 帧发送的配置情况

寄存器 CCCR		发送缓存		帧发送
BRSE	FDOE	FDF	BRS	
忽略	0	忽略	忽略	普通 can 格式传输
0	1	0	忽略	普通 can 格式传输
0	1	1	忽略	can FD 格式传输无比特率切换
1	1	0	忽略	普通 can 格式传输
1	1	1	0	can FD 格式传输无比特率切换
1	1	1	1	can FD 格式传输且比特率切换

当更新寄存器 TXBRP 的时候会自动扫描发送缓存的数据，发送最高优先级的发送请求。当传输完成或者软件增加或者取消发送任务时硬件自动更新寄存器 TXBRP 的值。

49.6.3.1 发送缓存

每个发送缓存都有一个可配置的消息标识符，如果有多个发送缓存配置了相同的消息标识符，缓存表编号小的发送缓存先发送。当更新完发送的数据部分后，通过配置寄存器 TXBRP 可以增加发送请求，此时，发送请求在内部和发送的 FIFO 或者发送的队列根据消息的标识符进行优先级的仲裁，优先级高的先发送。

49.6.3.2 发送 FIFO

发送 FIFO 通过配置寄存器 TXBC 的 TFQM 位为 0 来实现，当将一个消息添加到发送 FIFO, 通过对寄存器 TXBAR 对应的位写 1 来使能发送操作，发送 FIFO 里每发送一个消息，对应的寄存器 TXFQS 的 TFGI 就会加 1，直到发送 FIFO 空为止。

49.6.3.3 发送队列

发送队列通过配置寄存器 TXBC 的 TFQM 位为 1 来实现，发送队列里消息标识符低的有较高的优先级，新的消息要放到 TXFQS 的 TFQPI 所对应的发送缓存里。

49.6.3.4 发送取消

can 控制器支持取消发送操作，这个功能对于网关应用程序和汽车开放系统架构很适用，如果想取消一个发送请求，只需要将寄存器 TXBCR 对应发送请求的位置 1. 取下发送请求的功能只支持在发送缓存或者发送队列里的，对于发送 FIFO 里的，暂时不支持取消发送请求。成功的取消掉发送请求之后，寄存器 TXBCF 对应的位将置 1.

50 精确时间协议模块 PTPC

本章节介绍精确时间协议模块 PTPC 的主要功能和特性。

50.1 特性总结

本章节介绍精确时间协议模块 PTPC 的主要特性：

- 支持 2 套时间戳
- 时间戳支持 63 位计数器
- 计数器分为 32 位秒计数器和 31 位纳秒计数器
- 支持输入捕获时间戳
- 支持时间戳匹配时，产生中断
- 支持从时间戳读取接口发送数据到 CAN 模块

50.2 功能描述

本章节描述精确时间协议模块 PTPC 的功能。

50.2.1 与 can 的关系结构图

精确时间协议模块 PTPC 与 can 模块的关系如图 71所示。

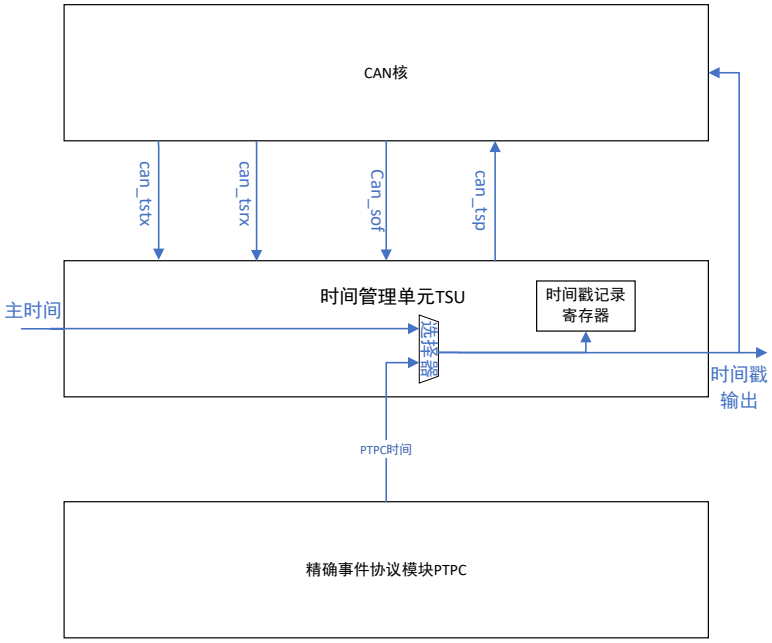


图 71: ptpc 与 can 的关系

50.2.2 时间戳模块

精确时间协议模块 PTPC 包含 2 个时间戳模块，每个时间戳模块包含一个 63 位的计数器。

计数器分为 32 位的秒计数器 TIMEH 和 31 位的纳秒计数器 TIMEL

TIMEL 在每次更新时，不像传统计数器那样单纯加 1，而是加上 CTRL1[SS_INCR] 的值：

$$TIMEL = TIMEL + SS\_INCR$$

假设 PTPC 的时钟为 100 MHz，每个时钟周期为 10 ns。此时，需要将 CTRL1 [SS_INCR] 配置为 8'd10。即 TIMEL 每次以 10 纳秒为单位累加。

计数器分为 31 位的 TIMEL 和 32 位的 TIMEH，每当 TIMEL 溢出时，TIMEH 累加。TIMEL 的溢出点支持以下 2 种：

- 计数器溢出，计数器低 31 位 TIMEL，当计数至 32'h7FFFFFFF 时溢出。TIMEL 溢出时，TIMEH 加 1
- 秒计数溢出，计数器低 31 位 TIMEL，当计数至 1000000000，即 32'h3B9ACA00 时溢出。TIMEL 溢出时，TIMEH 加 1。TIMEL 以纳秒为单位，TIMEH 以秒为单位计数。

用户把 CTRL0 [SUBSEC_DIGITAL_ROLLOVER] 置 0 时，配置计数器为计数器模式。置 1 时，配置计数器为秒计数溢出。

推荐用户将此位置 1，这样读取时间戳计数器，可以直观地得到 TIMEH 秒又 TIMEL 纳秒的真实时间计时。

在使用 PTPC 前，需要先完成 PTPC 初始化，初始化时间戳计数器 TIMEL，TIMEH 的步骤如下，：

- 把 CTRL0[TIMER_ENABLE] 位置 1
- 在 TS_UPDTH 和 TS_UPDTL 寄存器写入目标值。注意，如果 TIMEL 设置位秒计数溢出，TS_UPDTL 的值不要超过 1000000000，即 32'h3B9ACA00
- 把 CTRL0[INIT_TIMER] 位置 1

PTPC 开始计时之后，用户可以根据需要更新计时器，PTPC 更新时间戳 TIMEL，TIMEH 的步骤如下：

- 在 TS_UPDTH 和 TS_UPDTL 寄存器写入目标值。
- 配置 TS_UPDTL[ADD_SUB] 位，置 1 时，TIMEL/TIMEH 会在原值上加上 TS_UPDTL 和 TS_UPDTH，置 1 时，TIMEL/TIMEH 会在原值上减去 TS_UPDTL 和 TS_UPDTH
- 把 CTRL0[UPDATE_TIMER] 位置 1

PTPC 支持通过配置 ADDEND 寄存器微调其时间戳计数器的计时快慢。

TIMEL 作为纳秒计数器，在每个 PTPC 时钟周期累加 CTRL1 [SS_INCR]，当 PTPC 时钟为 100 MHz（即周期为 10 ns）时，应当配置 SS_INCR 为 10，即 TIMEL 每个 PTPC 时钟周期累加 10。当时钟同步软件基于 PTPC 计时器进行时钟同步时，有可能发现本地时间和网络的时钟 Master 之间的误差是由 PTPC 时钟的误差引起的。这时可以通过配置 ADDEND 寄存器实现等效的微调。

微调时，有个累加器，每个 ptpc 时钟加 ADDEND，累加器溢出 TIMEL 加 SS_INCR。所以如果使用微调模式，建议将 SS_INCR 设置成 20，ADDEND=0x80000000 时和原来一样，也就是两个时钟周期加 20。

50.2.3 时间戳捕获和比较

PTPC 支持输入捕获功能。每个时间戳支持一个捕获输入信号，可以配置 CAP0 或者 CAP1 信号的有效边沿，在 CAP0 或者 CAP1 上升沿或者下降沿时，将时间戳 TIMEL 和 TIMEH 的值，捕获存入寄存器 CAPT_SNAPL 和 CAPT_SNAPH。捕获发生时，捕获标志位会置 1。如果在中断使能寄存器内把相应的中断使能位置 1，就会生成时间戳捕获中断请求。

PTPC 支持时间戳匹配功能，用户将目标时间戳值写入 TARH 和 TARL 寄存器，并置位 CTRL0[COMP_EN]，当时间戳计数器 TIMEH 和 TIMEL 大于或等于 TARH 和 TARL 时，发生匹配事件, 匹配事件发生后，COMP_EN 会自动清零。

此时，匹配标志位会置 1，如果在中断使能寄存器内把相应的中断使能位置 1，就会生成时间戳匹配中断请

求。

PTPC 支持生成时间戳匹配输出比较，在时间戳匹配时：输出 CMP0 或 CMP1 到系统的其他模块。

50.2.4 时间戳输出端口

PTPC 支持 4 个时间戳输出端口，系统上的其他模块可以从此端口直接载入 TIMEL 和 TIMEH 的值。用户可以通过 TIME_SEL 寄存器直接配置端口 x 上时间戳信息的来源。相应的选择位置 0 时，输出模块 0 的时间戳。选择位置 1 时，输出模块 1 的时间戳。

50.3 PTPC 寄存器列表

PTPC 的寄存器列表如下：

PTPC base address: 0xF02FC000

地址偏移	名称	描述	复位值
0x0000	PTPC[0][CTRL0]	控制寄存器 0	0x00000000
0x0004	PTPC[0][CTRL1]	控制寄存器 1	0x00000000
0x0008	PTPC[0][TIMEH]	时间戳的高位	0x00000000
0x000C	PTPC[0][TIMEL]	时间戳的低位	0x00000000
0x0010	PTPC[0][TS_UPDTH]	更新高位时间戳	0x00000000
0x0014	PTPC[0][TS_UPDTL]	更新低位时间戳	0x00000000
0x0018	PTPC[0][ADDEND]	加数计数器更新值	0x00000000
0x001C	PTPC[0][TARH]	纳秒比较器目标值高位	0x00000000
0x0020	PTPC[0][TARL]	纳秒比较器目标值低位	0x00000000
0x002C	PTPC[0][PPS_CTRL]	秒脉冲周期控制	0x00000000
0x0030	PTPC[0][CAPT_SNAPH]	捕获值高位	0x00000000
0x0034	PTPC[0][CAPT_SNAPL]	捕获值低位	0x00000000
0x1000	PTPC[1][CTRL0]	控制寄存器 0	0x00000000
0x1004	PTPC[1][CTRL1]	控制寄存器 1	0x00000000
0x1008	PTPC[1][TIMEH]	时间戳的高位	0x00000000
0x100C	PTPC[1][TIMEL]	时间戳的低位	0x00000000
0x1010	PTPC[1][TS_UPDTH]	更新高位时间戳	0x00000000
0x1014	PTPC[1][TS_UPDTL]	更新低位时间戳	0x00000000
0x1018	PTPC[1][ADDEND]	加数计数器更新值	0x00000000
0x101C	PTPC[1][TARH]	纳秒比较器目标值高位	0x00000000
0x1020	PTPC[1][TARL]	纳秒比较器目标值低位	0x00000000
0x102C	PTPC[1][PPS_CTRL]	秒脉冲周期控制	0x00000000
0x1030	PTPC[1][CAPT_SNAPH]	捕获值高位	0x00000000
0x1034	PTPC[1][CAPT_SNAPL]	捕获值低位	0x00000000
0x2000	TIME_SEL		0x00000000
0x2004	INT_STS	中断状态	0x00000000
0x2008	INT_EN	中断使能	0x00000000
0x3000	PTPC_CAN_TS_SEL	时间戳选择	0x00000000

地址偏移	名称	描述	复位值

表 211: PTPC 寄存器列表

50.4 PTPC 寄存器描述

PTPC 的寄存器详细说明如下：

50.4.1 PTPC[CTRL0] (0x0 + 0x1000 * n)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																						SUBSEC_DIGITAL_ROLLOVER	CAPT_SNAP_KEEP	CAPT_SNAP_POS_EN	CAPT_SNAP_NEG_EN	RSVD	COMP_EN	UPDATE_TIMER	INIT_TIMER	FINE_COARSE_SEL	TIMER_ENABLE
N/A																						RW	RW	RW	RW	N/A	RW	WO	WO	RW	RW
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0

PTPC[CTRL0] [31:0]

位域	名称	描述
9	SUBSEC_DIGITAL_ROLLOVER	纳秒计数器模式 1-数字模式，溢出时间 1000000000/0x3B9ACA00，分辨率为 1 纳秒； 0-二进制，溢出时间 0x7FFFFFFF，分辨率为大约 0.466 纳秒
8	CAPT_SNAP_KEEP	保持捕获值 1：PTPC 会在捕获事件后保持捕获值，直到软件读取 capt_snapl 寄存器。此模式下，软件需要先读 capt_snaph 寄存器，避免错误结果。 0：PTPC 会在每次捕获事件后更新捕获值，软件需要在合适的时间点读取捕获值。
7	CAPT_SNAP_POS_EN	上升沿捕获使能 1 ：使能上升沿捕获 0：禁止上升沿捕获
6	CAPT_SNAP_NEG_EN	下降沿捕获使能 1 ：使能下降沿捕获 0：禁止下降沿捕获
4	COMP_EN	比较器使能。 置 1 使能比较器，PTPC 会在比较条件满足后清零此位

位域	名称	描述
3	UPDATE_TIMER	计时器更新 置 1 会在当前计时器上加或减 ts_updateh/ts_updatel 的时间。硬件在完成更新后自动清零该位
2	INIT_TIMER	计时器初始化 置 1 会在将计时器初始化为 ts_updateh/ts_updatel 的时间。硬件在完成更新后自动清零该位
1	FINE_COARSE_SEL	精细初略更新选择 0：精细更新，每次加数计数器溢出时，纳秒计数器增加 ss_incr 1：粗略更新，纳秒计数器每个时钟增加 ss_incr
0	TIMER_ENABLE	计数器使能 置 1 计数器工作

PTPC[CTRL0] 位域

50.4.2 PTPC[CTRL1] (0x4 + 0x1000 * n)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																								SS_INCR							
N/A																								RW							
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0

PTPC[CTRL1] [31:0]

位域	名称	描述
7-0	SS_INCR	纳秒计数器更新值。 例如 50MHz 时钟，每周期为 20 纳秒， 数字模式下，设为 0x14（每个时钟周期增加 20 纳秒）； 二进制模式下，设为 0x2B（20/0.466）；

PTPC[CTRL1] 位域

50.4.3 PTPC[TIMEH] (0x8 + 0x1000 * n)

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

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

PTPC[TIMEH] [31:0]

位域	名称	描述
31-0	TIMESTAMP_HIGH	当前秒计时器值

PTPC[TIMEH] 位域

50.4.4 PTPC[TIMEL] (0xC + 0x1000 * n)

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

PTPC[TIMEL] [31:0]

位域	名称	描述
31-0	TIMESTAMP_LOW	当前纳秒计时器值。 数值模式下精度为 1 纳秒； 二进制模式下精度为 0.466 纳秒

PTPC[TIMEL] 位域

50.4.5 PTPC[TS_UPDTH] (0x10 + 0x1000 * n)

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

PTPC[TS_UPDTH] [31:0]

位域	名称	描述
31-0	SEC_UPDATE	秒计时器更新值。 初始化时，用于将该值更新到秒计时器中； 更新时，从当前秒计时器加或减该值（纳秒计时器可能产生进位或借位）

PTPC[TS_UPDTH] 位域

50.4.6 PTPC[TS_UPDTL] (0x14 + 0x1000 * n)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADD_SUB	NS_UPDATE																														
RW	RW																														
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

PTPC[TS_UPDTL] [31:0]

位域	名称	描述
31	ADD_SUB	计时器更新模式。 1 表示减；0 表示加； 仅用于计时器更新。
30-0	NS_UPDATE	纳秒计时器更新值。 初始化时，用于将该值更新到纳秒计时器中； 更新时，从当前纳秒计时器加或减该值，可能会产生对秒计时器的进位或借位。

PTPC[TS_UPDTL] 位域

50.4.7 PTPC[ADDEND] (0x18 + 0x1000 * n)

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

PTPC[ADDEND] [31:0]

位域	名称	描述
31-0	ADDEND	加数计数器更新值。 仅用于精细更新模式。 每个时钟周期 32 位的加数计数器会累加该值，当加数计数器溢出后，纳秒计时器会增加 ss_incr

PTPC[ADDEND] 位域

50.4.8 PTPC[TARH] (0x1C + 0x1000 * n)

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

PTPC[TARH] [31:0]

位域	名称	描述
31-0	TARGET_TIME_HIGH	秒比较器目标值

PTPC[TARH] 位域

50.4.9 PTPC[TARL] (0x20 + 0x1000 * n)

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

PTPC[TARL] [31:0]

位域	名称	描述
31-0	TARGET_TIME_LOW	纳秒比较器目标值

PTPC[TARL] 位域

50.4.10 PTPC[PPS_CTRL] (0x2C + 0x1000 * n)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																												PPS_CTRL			
N/A																												RW			
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0

PTPC[PPS_CTRL] [31:0]

位域	名称	描述
3-0	PPS_CTRL	秒脉冲周期控制 为 0 时，PPS 输出为每秒一个脉冲（宽度为一个时钟周期）。 对于其他值，PPS 输出成为以下频率的生成时钟： 0001：二进制模式为 2 Hz，数字模式为 1 Hz。 0010：二进制模式为 4 Hz，数字模式为 2 Hz。 0011：二进制模式为 8 Hz，数字模式为 4 Hz。 0100: 二进制模式为 16 Hz，数字模式为 8 Hz。 .. 1111：二进制模式为 32.768 KHz，数字模式为 16.384KHz。 在二进制翻转模式下，PPS 输出是一个占空比 50% 的时钟。 在数字翻转模式下，PPS 输出频率为平均值 数字实际时钟的频率不同每秒同步一次。例如： 当 PPSCTRL=0001 时，PPS（1 Hz）的低周期为 537 ms 高周期 463ms； 当 PPSCTRL=0010 时，PPS（2 Hz）为以下序列： -一个 50% 占空比的 537 毫秒周期的时钟 -第二个时钟周期为 463 毫秒（低 268 毫秒，高 195 毫秒） 当 PPSCTRL=0011 时，PPS（4 Hz）为以下序列： -三个 50% 占空比的 268 毫秒周期的时钟 -第四个时钟周期为 195 毫秒（低 134 毫秒，高 61 毫秒） 这种行为是由于数字模式中，溢出位的非线性导致的（不是计时器全 1 时溢出）

PTPC[PPS_CTRL] 位域

50.4.11 PTPC[CAPT_SNAPH] (0x30 + 0x1000 * n)

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

PTPC[CAPT_SNAPH] [31:0]

位域	名称	描述
31-0	CAPT_SNAP_HIGH	在输入的捕获信号上升沿或者下降沿，将当前秒计数器值存放与该寄存器中。 捕获信号来自于电机系统的互联管理器

PTPC[CAPT_SNAPH] 位域

50.4.12 PTPC[CAPT_SNAPL] (0x34 + 0x1000 * n)

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

PTPC[CAPT_SNAPL] [31:0]

位域	名称	描述
31-0	CAPT_SNAP_LOW	在输入的捕获信号上升沿或者下降沿，将当前纳秒计数器值存放与该寄存器中。

PTPC[CAPT_SNAPL] 位域

50.4.13 TIME_SEL (0x2000)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																												CAN3_TIME_SEL	CAN2_TIME_SEL	CAN1_TIME_SEL	CAN0_TIME_SEL

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
N/A																												RW	RW	RW	RW
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0

TIME_SEL [31:0]

位域	名称	描述
3	CAN3_TIME_SEL	
2	CAN2_TIME_SEL	
1	CAN1_TIME_SEL	
0	CAN0_TIME_SEL	设 1 选择 PTPC1 作为 CAN 下的系统时钟； 设 0 选择 PTPTC0 作为 CANx 的系统时钟。

TIME_SEL 位域

50.4.14 INT_STS (0x2004)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD													COMP_INT_STS1	CAPTURE_INT_STS1	PPS_INT_STS1	RSVD													COMP_INT_STS0	CAPTURE_INT_STS0	PPS_INT_STS0
N/A													W1C	W1C	W1C	N/A													W1C	W1C	W1C
x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0

INT_STS [31:0]

位域	名称	描述
18	COMP_INT_STS1	PTPC1 比较器中断状态位
17	CAPTURE_INT_STS1	PTPC1 捕获中断状态位
16	PPS_INT_STS1	PTPC1 秒脉冲（pps）信号中断状态位，会在 pps 信号的上升沿和下降沿都产生中断
2	COMP_INT_STS0	PTPC0 比较器中断状态位
1	CAPTURE_INT_STS0	PTPC0 捕获中断状态位
0	PPS_INT_STS0	PTPC0 秒脉冲（pps）信号中断状态位，会在 pps 信号的上升沿和下降沿都产生中断

INT_STS 位域

50.4.15 INT_EN (0x2008)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD													COMP_INT_STS1	CAPTURE_INT_STS1	PPS_INT_STS1	RSVD													COMP_INT_STS0	CAPTURE_INT_STS0	PPS_INT_STS0
N/A													RW	RW	RW	N/A													RW	RW	RW
x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0

INT_EN [31:0]

位域	名称	描述
18	COMP_INT_STS1	PTPC1 比较器中断使能位
17	CAPTURE_INT_STS1	PTPC1 捕获中断使能位
16	PPS_INT_STS1	PTPC1 秒脉冲（pps）信号中断使能位
2	COMP_INT_STS0	PTPC0 比较器中断使能位
1	CAPTURE_INT_STS0	PTPC0 捕获中断使能位
0	PPS_INT_STS0	PTPC0 秒脉冲（pps）信号中断使能位

INT_EN 位域

50.4.16 PTPC_CAN_TS_SEL (0x3000)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TSU_TBIN3_SEL						TSU_TBIN2_SEL						TSU_TBIN1_SEL						TSU_TBIN0_SEL						RSVD							
RW						RW						RW						RW						N/A							
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	x	x	x	x	x	x	x	x

PTPC_CAN_TS_SEL [31:0]

位域	名称	描述
31-26	TSU_TBIN3_SEL	CANx 输入时间戳 3 选择，每个 CAN 模块有同样的 4 个输入时间戳，CAN 可以配置内部寄存器选择其中一个。 本寄存器配置输入时间戳 3 的来源。 0x20 选择 PTPC0；0x21 选择 PTPC1； 0x00 选择 CAN0 输出时间戳；0x01 选择 CAN1 输出时间戳； 0x02 选择 CAN2 输出时间戳；0x03 选择 CAN3 输出时间戳；
25-20	TSU_TBIN2_SEL	CANx 输入时间戳 2 选择

位域	名称	描述
19-14	TSU_TBIN1_SEL	CANx 输入时间戳 1 选择
13-8	TSU_TBIN0_SEL	CANx 输入时间戳 0 选择

PTPC_CAN_TS_SEL 位域

51 局域互联网络 LIN

本章节介绍局域互联网络 LIN 的功能和特性。

51.1 概述

本章节列举了局域互联网络 LIN 的特性总结：

- 支持 LIN 1.3 规范，当作为 master 使用时支持 LIN 2.2 规范；
- 支持 master 功能和 slave 功能，支持 master 和 slave 功能的动态切换；
- master 模式时，可工作在 1Kbps 至 20Kbps 数据速率下；
- slave 模式时，支持总线波特率的动态自动测量；
- 数据缓存高达 8bytes；
- 支持 bit error 的检测与告警；
- 支持 checksum 和 enhence checksum 的生成与检测；
- 支持 sleep 模式的进入与退出；
- 支持多种 timeout error 的探测与告警；

51.2 LIN 结构图

本模块的结构图如图图 72：

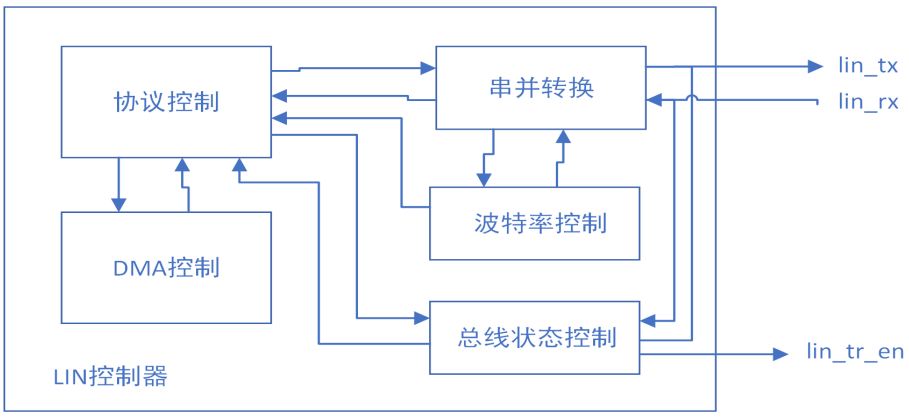


图 72: LIN 结构框图

图中，串并转换模块可以实现 8bit 内部数据和 1bit 总线串行数据的转换；波特率控制模块在主模式时，从系统工作时钟分频得到 LIN 总线波特率时钟，在从模式时，可动态测量 LIN 总线波特率；总线状态控制模块可实现 LIN 总线在 wake up 和 sleep 状态的管理；协议控制模块可实现 LIN 数据帧的成帧控制与校验位的生成；DMA 控制模块可在 slave 模式下，协助 DMA 引擎实现数据的自动收发。

51.3 管脚说明

51.4 功能说明

51.4.1 LIN 总线与数据帧

LIN 总线是一种一主多从的串行总线，可以以数据帧的形式实现 LIN 总线上各节点之间的数据通信。LIN 总线控制器提供了符合 LIN 标准协议的串行数据通信功能，可作为 master 节点也可作为 slave 节点使用，软件可通

表 212: lin 管脚说明

管脚	方向	功能描述
lin_tx	输出	LIN 总线输出信号
lin_rx	输入	LIN 总线输入信号
lin_tr_en	输出	LIN 总线使能信号

过 LIN 总线控制器实现 LIN 数据帧的生成、发送与接收等功能。

一个完整的 LIN 数据帧包括 header 和 response 两个部分，其中 header 部分必须由 LIN 的 master 节点发出，response 部分可以由 LIN 的 master 发送，也可以由 LIN 的 slave 发送。LIN 数据帧的 header 部分包含三个区域，分别是 break field、sync field 和 protected identifier，其中，break field 区域由一个长度超过 11bit 的低电平组成，用来标识一个数据帧的开始；sync field 的内容固定为 0x55，slave 节点可以通过 sync field 字段测量出当前 LIN 总线的波特率；protected identifier 字段包含 6bit 的 ID 值和 2bit 的校验位，LIN 消息通过 ID 值来区分本数据帧的业务类型；response 字段由 0 至 8byte 的数据和 1 byte 的 checksum 字段组成，checksum 指示传统校验模式和增强校验模式。

51.4.2 波特率控制

LIN 总线的波特率是由 LIN 控制器工作时钟分频得到的，LIN 控制器工作时钟默认为 80MHz，当 LIN 控制器工作在主模式时，LIN 总线波特率和工作时钟的关系如下式所示：

$$f\_bit = \frac{f\_clock}{2^{prescal+1} \cdot bt\_div \cdot (bt\_mul + 1)}$$

其中,f_bit 表示 LIN 总线的波特率;f_clock 表示 LIN 控制器的工作时钟;prescal 是寄存器 timing_control.prescl 的值；bt_div 是寄存器 timing_control.bt_div 的值；bt_mul 是寄存器 timing_control.bt_mul 的值。

当 LIN 工作在 master 模式时，首先需要通过下式确定 bt_mul 的值，如果计算结果为小数，丢去小数位，向下取整：

$$bt\_mul = \frac{20Kbit/s}{f\_bit} - 1$$

然后再确定 prescal 的值，如下式所示，计算结果也是向下取整：

$$prescal = \ln\left(\frac{f\_clock}{(bt\_mul + 1) \cdot f\_bit \cdot 200}\right) \cdot \frac{1}{\ln(2)} - 1$$

最后再确定 bt_div 的值，如下式所示，计算结果也是向下取整：

$$bt\_div = \frac{f\_clock}{2^{(prescal+1)} \cdot (bt\_mul + 1) \cdot f\_bit}$$

当 LIN 工作在 slave 模式时，因 slave 节点的波特率是自动测量出来的，因此只需要配置 prescal 和 bt_div 的值即可，如下式所示：

$$prescl = \ln\left(\frac{f\_clock}{20KHz \cdot 200}\right) \cdot \frac{1}{\ln(2)} - 1$$

$$bt\_div = \frac{f\_clock}{2^{(prescal+1)}} \cdot 20KHz$$

51.4.3 master 工作模式

当 LIN 控制器工作在 master 模式时，LIN 控制器可以发送帧头启动 LIN 总线数据传输，LIN 数据帧中的数据可以是 master 发给 slave 节点，也可是是 slave 节点发送到 master 的，其工作过程为：

1. 配置 ID 值到寄存器 data_len_id.ID 中；
2. 配置数据长度到寄存器 data_len_id.data_len 中;配置适当的 checksum 模式到寄存器 data_len_id.enh_check 中；
3. 配置数据传输的方向到寄存器 control_status.transmit 中，可以是发送数据，也可以接受数据；如果是发送数据，则需要将待发送的数据配置到寄存器 data_byte0 7 中；
4. 通过将寄存器 control_status.start_req 写 1，启动数据帧的生成与发送，并等待中断的发生；
5. 收到中断后，读取寄存器 control_status，检查其中的 error、wakeup、complete 的状态，正常情况下 complete 应为 1，error 为 0；

51.4.4 slave 工作模式

当 LIN 控制器工作在 slave 模式时，LIN 控制器每当接受到一个 LIN 数据帧的有效帧头后就产生一次中断，之后 LIN 控制器可在软件的控制下对 master 进行应答，完成一次 LIN 数据帧的完整交互，其工作过程为：

1. 收到中断后，读取寄存器 control_status 的值；
2. 检查其中 data_req 寄存器的值是否为 1，若是，则根据 ID 的值，决定 transmit 的值和 data_len 的值，如果 transmit 为 1，则将待发送的数据写到寄存器 data_byte0 7 中；然后将寄存器 data_ack 置 1，使得 LIN 控制器完成数据帧的传输；
3. 检查其中 error、timeout、bus idle timeout、wakeup 寄存器是否为 1，若为 1，则通过 reset_int 和 reset_error 寄存器清掉错误；
4. 检查其中的 complete 寄存器是否为 1，若为 1，则表示一次 LIN 数据帧的收发完成了；

51.4.5 sleep 和 wakeup 工作模式

为了降低系统功耗，LIN 协议定义了 sleep 模式，软件可通过配置寄存器 control_status 寄存器使 LIN 控制器进入 sleep 模式。master 模式和 slave 模式都可以进入 sleep 模式，一般的，当 LIN 总线在超过 4s 以上无任何数据帧收发时，LIN 控制器会产生中断，此时软件可将 LIN 控制器设置为 sleep 模式。在 sleep 模式，管脚 lin_tr_en 会置低电平，关闭 LIN 总线的输出驱动。

通过发送 wakeup 信号可以使 LIN 总线退出 sleep 模式，wakeup 信号是一段超过 250us 的低电平。当 LIN 控制器工作在 master 模式时，软件写寄存器 control_status.wakeup_req 触发 wakeup 信号的产生，当 wakeup 信号发送完成后，LIN master 节点会产生中断。当 LIN 控制器工作在 slave 模式时，软件也可以通过写寄存器 control_status.wakeup_req 触发 wakeup 信号的产生，当 wakeup 信号发送完成后，LIN slave 节点并不会产生中断，但此时 slave 会计算从 wakeup 信号发出，到接收到 LIN 数据帧头的时间，如果这个时间超过 150ms，则 slave 节点会产生中断。

51.4.6 错误与超时

LIN 控制器指示的错误检测有：

- bit_error：表示发送一个 bit 的电平与从总线上检测到的电平不一致，或者该接受到 stop 位置不为高电平；
- chk_error：表示接收到的数据帧的 checksum 校验出错；
- parity_error：表示接收到的数据帧帧头中 ID 字段的奇偶校验出错；
- break_err：表示接收到的 break 信号长度过段，不满足 11 个 bit 的长度要求；

值得注意的是，对于 slave 模式 checksum 的生成与检测，LIN 控制器在一帧的传输过程中是不可以修改的。比如，如果 slave 在接收帧头阶段 enh_check 的值设置为 0，slave 完成帧头的接受并产生了中断，软件在收到中断后不能再修改 enh_check 的值了。这也就意味着，LIN 控制器在 slave 模式工作时，如果需要切换 checksum 的校验模式，需要在 LIN 总线的 master 节点发送帧头前，就将 LIN 总线的 slave 节点的 checksum 校验模式修改完成。

LIN 控制器支持的超时错误有：

- bus_idle_timeout：表示 LIN 总线空闲时间超过 4s；
- timeout：在 master 模式，表示一次数据帧传输的时间超过理想情况的 1.4 倍；在 slave 模式时，表示在收到 LIN 数据帧帧头后，软件未及时下发针对该数据帧的处理意见；在 slave 模式时，还可以表示从发送完成一次 wakeup 信号，到一次有效的数据帧头的时间超过 150ms；

51.4.7 DMA 工作模式

当 LIN 控制器工作在 slave 模式下，如果有一种特定类型的数据访问需要频繁发生，则可以通过 DMA 的方式进行，避免该业务类型对 CPU 的频繁中断。根据数据传输方向的不同，对于 DMA 的操作过程可能会出现两种情况：LIN master 向 slave 发送数据和 LIN master 向 slave 获取数据。

当 LIN master 向 slave 发送数据时，其工作过程如下：

1. 配置寄存器 dma_control.dma_req_id 的值为需要 DMA 参与的业务类型；
2. 配置寄存器 dma_control.dma_req_id_type 为 0，表示 LIN slave 需要接受数据；
3. 配置寄存器 dma_control.dma_req_len 为需要的值，表示 LIN 数据帧的数据字段长度；
4. 配置寄存器 dma_control.dma_req_enh_chk 为合适的值，用来指示 LIN 数据帧的 checksum 校验模式；
5. 在系统内存中开辟一块内存，用来保存 LIN 数据帧中的数据，配置 DMA 引擎，使得 dstaddr 指向该内存，srcaddr 指向 LIN 控制器的 byte0 寄存器；
6. 配置寄存器 dma_control.dma_req_enable 为 1，当 LIN 控制器收到一个数据帧，且数据帧帧头中 ID 字段与 dma_control.dma_req_id 相同时，则在该数据帧传输完成后，会自动触发 DMA 的操作，将 LIN 数据帧中的数据保存到系统内存中；

当 LIN master 向 slave 索要数据时，其工作过程如下：

1. 配置寄存器 dma_control.dma_req_id 的值为需要 DMA 参与的业务类型；
2. 配置寄存器 dma_control.dma_req_id_type 为 1，表示 LIN slave 需要接受数据；
3. 在系统内存中开辟一块内存，用来保存 LIN 数据帧中的待发送的数据，并在其后紧跟寄存器 data_len_id 和寄存器 control_status 需要配置的值，并配置 DMA 引擎，使得 srcaddr 指向该内存，dstaddr 指向 LIN 控制器的 byte0 寄存器；
4. 配置寄存器 dma_control.dma_req_enable 为 1，当 LIN 控制器收到一个数据帧，且数据帧帧头中 ID 字段与 dma_control.dma_req_id 相同时，则在该数据帧的帧头时会自动触发 DMA 的操作，DMA 操作完

成后，LIN 控制器再完成 LIN 数据帧的 data 字段和 checksum 字段的传输；

51.5 LIN 寄存器

51.5.1 LIN 寄存器说明

LINV2 的寄存器列表如下：

LIN0 base address: 0xF0020000

LIN1 base address: 0xF0024000

LIN2 base address: 0xF0028000

LIN3 base address: 0xF002C000

地址偏移	名称	描述	复位值
0x0000	DATA[DATA0]	payload 数据	0x00000000
0x0004	DATA[DATA1]	payload 数据	0x00000000
0x0000	DATA_BYTE[DATA_BYTE0]		0x00
0x0001	DATA_BYTE[DATA_BYTE1]		0x00
0x0002	DATA_BYTE[DATA_BYTE2]		0x00
0x0003	DATA_BYTE[DATA_BYTE3]		0x00
0x0004	DATA_BYTE[DATA_BYTE4]		0x00
0x0005	DATA_BYTE[DATA_BYTE5]		0x00
0x0006	DATA_BYTE[DATA_BYTE6]		0x00
0x0007	DATA_BYTE[DATA_BYTE7]		0x00
0x0008	DATA_LEN_ID	数据长度与 ID 寄存器	0x00000000
0x000C	CONTROL_STATUS	控制与状态寄存器	0x00000000
0x0010	TIMING_CONTROL	时钟控制寄存器	0x00400000
0x0014	DMA_CONTROL	dma 控制寄存器	0x00000000

表 213: LINV2 寄存器列表

51.5.2 寄存器详细信息

LINV2 的寄存器详细说明如下：

51.5.3 DATA (0x0 + 0x4 * m)

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

DATA [31:0]

位域	名称	描述
31-0	DATA	payload 数据

DATA 位域

51.5.4 DATA_BYTE (0x0 + 0x1 * m)

7	6	5	4	3	2	1	0
DATA_BYTE							
RW							
0	0	0	0	0	0	0	0

DATA_BYTE [7:0]

位域	名称	描述
7-0	DATA_BYTE	payload 数据

DATA_BYTE 位域

51.5.5 DATA_LEN_ID (0x8)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD								CHECKSUM								ID_PARITY		ID						ENH_CHECK	RSVD			DATA_LEN			
N/A								RO								RO		RW						RW	N/A			RW			
x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	x	x	x	0	0	0	0

DATA_LEN_ID [31:0]

位域	名称	描述
23-16	CHECKSUM	checksum 结果， 当 control_status.transmit = 1 时，为待发送的 checksum 结果； 当 control_status.transmit = 0 时，为接收到的 checksum 结果；
15-14	ID_PARITY	ID 校验位， 当 master 模式时，此为待发送的 ID 校验位， 当 slave 时，此为接收到的 ID 校验位。
13-8	ID	ID 寄存器
7	ENH_CHECK	1: 增强校验模式 0：普通校验模式
3-0	DATA_LEN	payload 数据长度寄存器，0~8。当配置为全 1 时，有 ID[5:4] 译码得到：00-2 01-2 10-4 11-8

DATA_LEN_ID 位域

51.5.6 CONTROL_STATUS (0xC)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD										BREAK_ERR_DIS	BREAK_ERR	PARITY_ERROR	TIME_OUT	CHK_ERROR	BIT_ERROR	LIN_ACTIVE	BUS_IDLE_TIMEOUT	ABORTED	DATA_REQ	INT	ERROR	WAKEUP	COMPLETE	STOP	SLEEP	TRANSMIT	DATA_ACK	RESET_INT	RESET_ERROR	WAKEUP_REQ	START_REQ
N/A										RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	WO	RW	RW	RW	WO	WO	RW	RW
x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

CONTROL_STATUS [31:0]

位域	名称	描述
21	BREAK_ERR_DIS	关闭 break_err 寄存器的功能； 0：enable 1：disable
20	BREAK_ERR	slave 模式收到 break error 指示； 1：break 时间过短错误； 0：无错误；
19	PARITY_ERROR	slave 模式下，ID 字段奇偶校验错
18	TIME_OUT	超时错误。Master 模式时，当收不到 slave 的响应帧时置 1。Slave 模式时，在接收数据第一个 byte 结束之前，未收到软件关于该数据请求的相关配置（如 data_ack 或 stop），认为超时错误。在 slave 模式时，当发送一个 timeout 信号，但在 150ms 内未检测到 sync 响应，认为超时错误。
17	CHK_ERROR	数据校验和错
16	BIT_ERROR	总线上出现了 bit 错误
15	LIN_ACTIVE	1：LIN 总线正传输数据 0：LIN 总线处于 IDLE
14	BUS_IDLE_TIMEOUT	slave 模式下，当 sleep 寄存器为 0，且总线在 4s 内没有被激活，置 1。Reset_error 为 1 时清零
13	ABORTED	slave 模式下，当数据域段产生错误或传输超时时，置 1
12	DATA_REQ	slave 模式下，当收到一个有效的帧头并请求中断时，置 1
11	INT	当检测到中断时置 1，当 reset_int 为 1 时清零
10	ERROR	当检测到错误时置 1，当 reset_error 为 1 时清零
9	WAKEUP	当发送一次 wakeup 信号或接收一次 wakeup 信号时置 1，当 reset_error 为 1 时清零。
8	COMPLETE	当一次传输成功完成后置 1，当下一次传输开始时清 0
7	STOP	如果软件认为读取的 ID 不可用，则将该寄存器置 1，slave 不会回复响应帧。硬件自动回 0（aborted）.

位域	名称	描述
6	SLEEP	该 bit 可决定 LINbus 是否处于 Sleep 状态。当发送或收到 sleep mode 帧，或总线空闲超时发生，或作为 master 时发送的 wakeup 请求没有得到回应并超时时，该 bit 置 1 。当检测到总线上出现 wakeup 信号时，该 bit 由硬件清 0.
5	TRANSMIT	1 表示数据需要发送 0 表示数据需要接收
4	DATA_ACK	只在 slave 模式时有效，当处理完数据请求中断后，可将该寄存器写 1，来发送或接收应答帧。
3	RESET_INT	软件写 1 清除中断寄存器
2	RESET_ERROR	软件写 1 可清除状态寄存器中的 error 状态和 error 寄存器，还可清除 timeout、complete、wakeup 寄存器。
1	WAKEUP_REQ	软件写 1 退出 sleep 模式，并在总线上发出 wakeup 信号，硬件自动回 0
0	START_REQ	只在 master 模式下有效，置 1 启动数据帧的传输，当检测到错误或数据传输完成后自动清零

CONTROL_STATUS 位域

51.5.7 TIMING_CONTROL (0x10)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD		WAKE_LEN			BRK_LEN			RSVD	LINBUSDISABLE	LIN_INITIAL	MASTER_MODE	BUS_INACTIVE_TIME		WUP_REPEAT_TIME		PRESCL		BT_MUL					BT_DIV								
N/A		RW			RW			N/A	RW	RW	RW	RW		RW		RW		RW					RW								
x	x	0	0	0	0	0	0	x	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

TIMING_CONTROL [31:0]

位域	名称	描述
29-27	WAKE_LEN	weakup 时间长度配置， 0: 250us 1：500us 2: 750us … 7: 2000us

位域	名称	描述
26-24	BRK_LEN	break 时间长度配置，单位为总线发送单个 bit 的时间，只对 master 有效 0: 13bit 的 break 1：15bit 的 break 2：17bit 的 break ... 7: 27bit 的 break
22	LINBUSDISABLE	1：lin 总线的 rx 不使能，rx 按常 1 处理
21	LIN_INITIAL	1: lin 控制器初始化
20	MASTER_MODE	1：master 模式
19-18	BUS_INACTIVE_TIME	slave 模式下有效。lin 总线空闲超时寄存器，00-4s 01-6s 10-8s 11-10s
17-16	WUP_REPEAT_TIME	slave 模式下有效。wakeup 重发时间寄存器，00-180ms 01-200ms 10-220ms 11-240ms
15-14	PRESCL	prescl 寄存器
13-9	BT_MUL	bt_mul 寄存器
8-0	BT_DIV	bt_div 寄存器

TIMING_CONTROL 位域

51.5.8 DMA_CONTROL (0x14)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																			DMA_REQ_ENH_CHK	DMA_REQ_LEN				DMA_REQ_ID_TYPE	DMA_REQ_ID						DMA_REQ_ENABLE
N/A																			RW	RW				RW	RW						RW
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0

DMA_CONTROL [31:0]

位域	名称	描述
12	DMA_REQ_ENH_CHK	dma 操作时，payload 数据校验方式
11-8	DMA_REQ_LEN	dma 操作的 payload 数据长度
7	DMA_REQ_ID_TYPE	1：数据发送 0：数据接收
6-1	DMA_REQ_ID	dma_req_id 寄存器
0	DMA_REQ_ENABLE	slave 模式下有效，当收到数据请求帧头的 ID 与 dma_req_id 相等时，可触发 dma 操作

位域	名称	描述

DMA_CONTROL 位域

51.6 LIN 配置使用说明

51.6.1 master 模式配置使用说明

当 LIN 控制器工作于 master 模式时，如果是 master 往 slave 发送数据的话，其工作过程如下：

1. 根据 LIN 总线的波特率，配置三个分频寄存器的值，包括 timing_control.bt_div、timing_control.bt_mul 和 timing_control.prescl；
2. 配置 LIN 数据帧所需要的帧头信息，主要是 data_len_id 寄存器中的信息；
3. 配置 LIN 数据帧所需要的数据，主要是 data_byte0 data_byte7 寄存器；
4. 启动 LIN 数据帧的生成与发送，主要是 control_status.start_req 写 1，control_status.transmit 写 1；
5. 等待中断产生后，读取寄存器 control_status，判断 control_status.complete 是否为 1，是否有其他 error 指示。

当 LIN 控制器工作于 master 模式时，如果是 slave 往 master 发送数据的话，其工作过程如下：

1. 根据 LIN 总线的波特率，配置三个分频寄存器的值，包括 timing_control.bt_div、timing_control.bt_mul 和 timing_control.prescl；
2. 配置 LIN 数据帧所需要的帧头信息，主要是 data_len_id 寄存器中的信息；
3. 启动 LIN 数据帧的生成与发送，主要是 control_status.start_req 写 1，control_status.transmit 写 1；
4. 等待中断产生后，读取寄存器 control_status，判断 control_status.complete 是否为 1，是否有其他 error 指示。
5. 如果没有错误，通过寄存器 data_byte0 data_byte7 获取数据帧中的数据信息。

51.6.2 slave 模式配置使用说明

当 LIN 控制器工作于 slave 模式时，如果是 master 往 slave 发送数据的话，其工作过程如下：

1. 根据 LIN 总线的波特率，配置三个分频寄存器的值，包括 timing_control.bt_div、timing_control.bt_mul 和 timing_control.prescl；
2. 等待中断的产生，读取寄存器 control_status，判断其 data_req 是否为 1，是否有 error 产生，如果没有 error，从寄存器 data_len_id 中获取接收帧头中的 ID 值；
3. 配置 control_status.data_ack 为 1，control_status.transmit 为 0；
4. 等待中断的产生，读取寄存器 control_status，判断其 complete 是否为 1，若为 1 表示数据帧接收完成；
5. 读取 data_byte0 7, 获取 LIN 数据帧中的数据；

当 LIN 控制器工作于 slave 模式时，如果是 slave 往 master 发送数据的话，其工作过程如下：

1. 根据 LIN 总线的波特率，配置三个分频寄存器的值，包括 timing_control.bt_div、timing_control.bt_mul 和 timing_control.prescl；
2. 等待中断的产生，读取寄存器 control_status，判断其 data_req 是否为 1，是否有 error 产生，如果没有 error，从寄存器 data_len_id 中获取接收帧头中的 ID 值；
3. 配置 control_status.data_ack 为 1,control_status.transmit 为 1,将待发送的数据写入寄存器 data_byte0 7 中；
4. 等待中断的产生，读取寄存器 control_status，判断其 complete 是否为 1，若为 1 表示数据帧接收完成；

51.6.3 sleep 模式配置使用说明

LIN 控制器内部有一个 LIN 总线 IDLE 超时功能，一般情况下在产生了 IDLE 超时后，软件可以控制 LIN 控制器进入 sleep 模式。进入 sleep 模式只需要配置寄存器 control_status.sleep 为 1 就可以了。如果 LIN 控制器成功进入了 sleep 模式，回读 control_status.sleep 也会是 1.

如果需要主动退出 sleep 模式时，当 LIN 工作在 master 模式时，其步骤为：

1. 写寄存器 control_status.wakeup_req 为 1；
2. 等待中断产生后，读取 control_status.wakeup 是否为 1，表示 wakeup 信号成功发出；
3. 写寄存器 control_status.reset_int 为 1 清除中断。

如果需要主动退出 sleep 模式时，当 LIN 工作在 slave 模式时，其步骤为：

1. 写寄存器 control_status.wakeup_req 为 1；
2. 等待中断产生后，读取 control_status.time_out 是否为 0，control_status.data_req 是否为 1，表示 slave 发出的 wakeup 信号成功唤醒了 LIN 总线的 master 节点；
3. 如果 control_status.time_out 为 1，表明 slave 发出的 wakeup 信号唤醒失败，此时可能需要软件重新下发 wakeup 信号。

52 通用串行总线 USB

52.1 功能简介

提供 USB 通信解决方案, 完全兼容 USB2.0 协议

- 支持 OTG(可配置成主机 host 模式, 也可配置成设备 device 模式)
- 主机模式支持高速 (480Mbps), 全速 (12Mbps) 和低速 (1.5Mbps)
- 设备模式支持高速 (480Mbps), 全速 (12Mbps)
- 设备模式支持 8 个双向端点 (8IN + 8OUT endpoint)
- 内置片上高速 usbphy(支持全速和低速)
- 内置 DMA, 通过描述符，无需 cpu 参与即可完成 usb 与系统存储器之间的数据交互
- 主机模式软件结构完全兼容 EHCI1.0(Enhanced Host Controller Interface Specification for Universal Serial Bus) 协议
- 低功耗模式，可以关掉片上所有 PLL 的情况下, 通过片上 osc 时钟完成唤醒

52.2 管脚说明

表 214: USB 管脚说明

管脚	方向	功能描述
USB_DP	双向	差分信号正端。
USB_DN	双向	差分信号负端。

52.3 工作流程

52.3.1 usbphy 初始化

退出 reset 状态：OTG_CTRL0.OTG_UTMI_RESRET_SW=0

退出 suspend 状态：OTG_CTRL0.OTG_UTMI_SUSPENDM_SW=1

检测 usbphy 时钟：PHY_STATUS.UTMI_CLK_VALID=1

等待 usbphy 时钟稳定：while(PHY_STATUS.UTMI_CLK_VALID==1)

52.3.2 配置工作模式

系统上电后 USB 模块处于 OTG 模式, 如需改变模式，先进行 usb reset(USBCMD.RST=1).

用户可以根据需求配置成主机或者设备模式.

也可以保持 OTG 模式, 根据外部连接情况进行配置.

如果是 micro-AB 接口, 保持 ID 为高 (通过 gpio 配置上拉), 保持 VBUS 为低

- 检测到 ID 变低, 说明有设备接入 (也可能有 otg 线插入)，设置成主机模式 (USBMODE.CM=3)
- 检测到 VBUS 变高，说明有主机接入，设置成设备模式 (USBMODE.CM=2)

如果是 typeC 接口, 请参考 typeC 协议设置主机或设备模式

52.3.3 主机初始化流程

打开 VBUS(一般通过 gpio 控制板子上的供电芯片).

打开端口：PORTSC1.PP=1

等待设备连接，在收到端口变化中断 (USBSTS.PCI) 后, 检查端口连接状态 (PORTSC1.CCS),1 表示有设备连接

检查 linestate：PHY_STATUS.LINE_STATE

为 2，表示接入的是低速设备，需要使用串行模式 (设置 PORTSC1.STS=1)

为 1，表示接入的是全速或者高速设备，使用并行模式 (设置 PORTSC1.STS=0)

启动 USB：USBCMD.RS=1

reset 设备：PORTSC1.PR=1

等待 bus reset 结束：while(PORTSC1.PR==1), 这个时间大概在 55ms 左右，期间会收到端口变化的中断 (USBSTS.PCI)

确定设备连接速度：PORTSC1.PSPD

枚举设备 (具体流程参考 EHCI 协议)

52.3.4 设备初始化流程

设置并行模式 (设备模式不支持低速, 可以全部工作在并行模式) ：PORTSC1.STS=0

准备好端点 0 的接收描述符, 并将其地址写入 ENDPTLISTADDR.

等待 VBUS(OTGSC.AVV) 为 1.

启动 USB：USBCMD.RS=1

等待主机发送 bus reset(USBSTS.URI)

等待端口变化中断 USBSTS.PCI.

确定主机连接速度：PORTSC1.PSPD

等待主机发起枚举并响应.

52.4 数据结构

52.4.1 主机数据结构

为方便软件开发，主机数据结构完全兼容 EHCI1.0 协议，具体请参考协议.

这里简单介绍一下 BULK/CTRL 传输使用的异步队列, 所有图片来源于 EHCI 协议

异步队列是一个循环链表，如图 73所示，链表中每一项 (也叫 qhead) 对应一个端点 (EP, endpoint), 一个主机可以连接多个设备，一个设备可以有多个端点.

硬件会轮询这个链表，看是否有需要传输的数据. 其中一个 qhead 会标记 H 为 1，表示链表头，当硬件两次轮询到链表头之间，没有数据传输时，会停止轮询。软件在初始化时需将链表头的地址写入 ASYNCLISTADDR 寄存器，硬件会在轮询过程中实时更新 ASYNCLISTADDR 为当前正在执行的 qhead 地址每个 qhead 可以链接 0 到多个 qTD，qTD 用于传输数据，包含数据包地址，传输状态，以及下一个 qTD 地址。

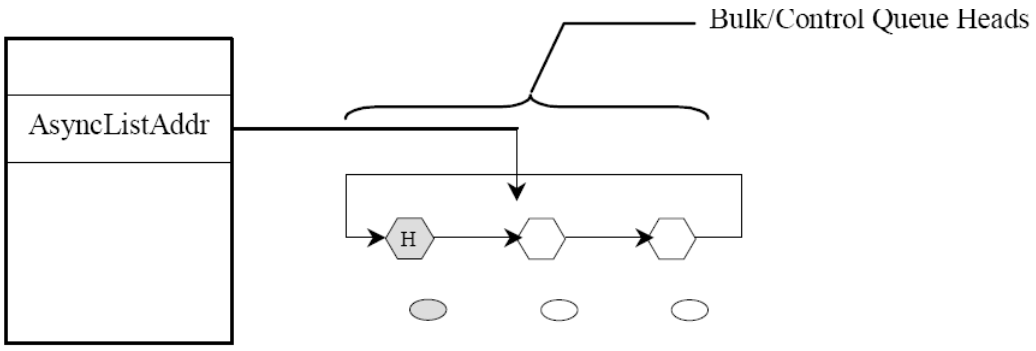


图 73: 异步队列示意图

qhead 结构如图 74所示,

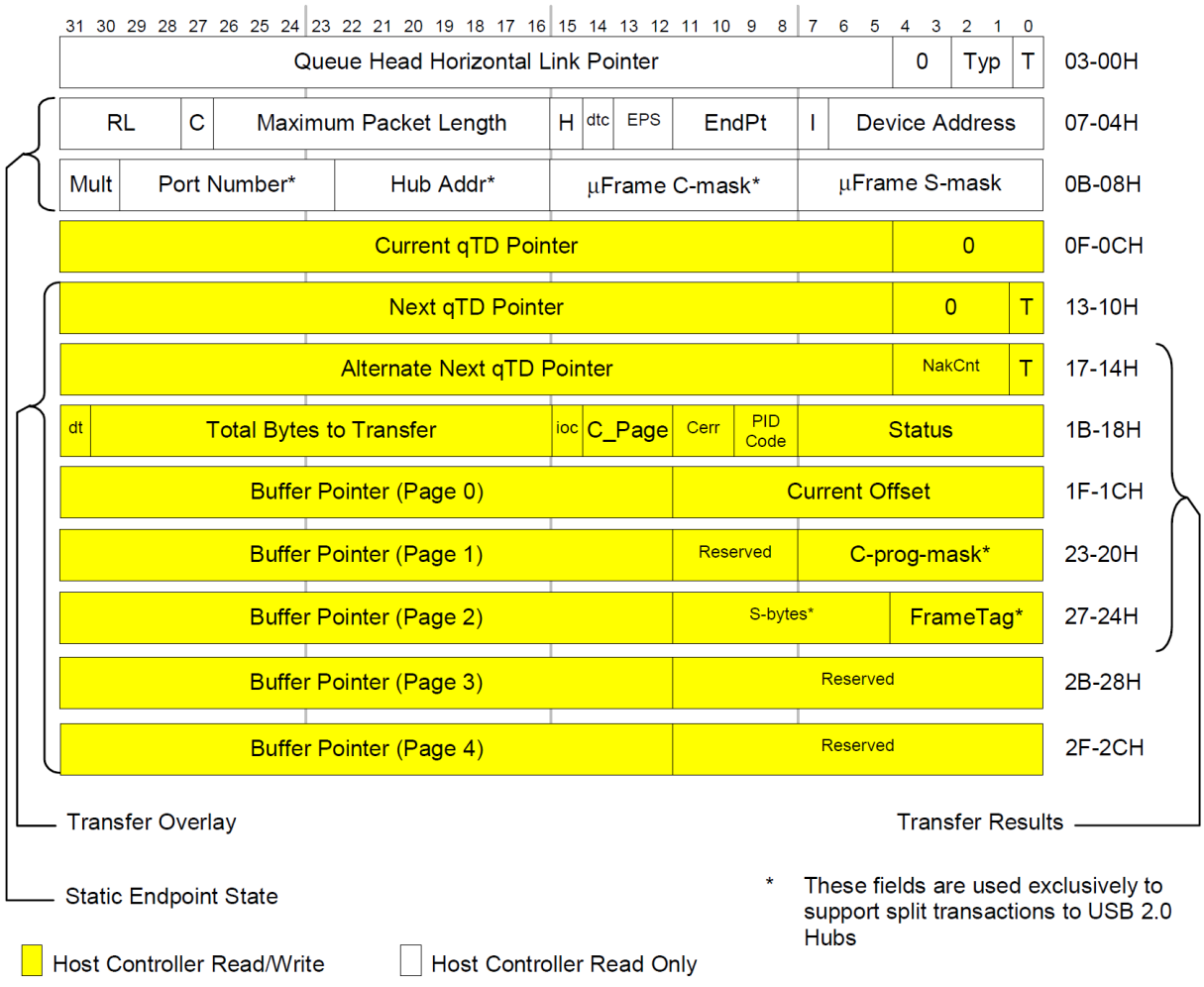


图 74: qhead 示意图

简单介绍下 BULK/CTRL 传输需要用到的部分，未介绍的详见 EHCI 协议。

qhead 的字节 0x10 及之后部分，也叫 Transfer Overlay，硬件会把当前传输的 qTD 写到 qhead 的这部分空间，在当前 qTD 传输完成前，在 qhead 里更新状态，传输完成后回写到 qTD

字节	位	描述
0x00	31:5	Queue Head Horizontal Link Pointer 表示下一个 qhead 地址，qhead 数据结构必须 32byte 对齐，如果当前应用中只有一个设备的一个端点，这个地址可以指向 qhead 本身。
0x00	2:1	Typ 是 qhead 类型，BULK 和 CTRL 传输需要设成 01。
0x00	0	T 是 Terminate 的简称，设 1 表示当前地址无效，对于 BULK/CTRL 的 qhead，Queue Head Horizontal Link Pointer 的 T 需要为 0。
0x04	31:28	RL 设 0
0x04	27	C 表示 CTRL 端点，当 qhead.EPS 为高速设备且当前端点是 CTRL 端点时，需要设 1，其他情况设 0
0x04	26:16	Maximum Packet Length, 当前端点最大数据包长度，主机枚举过程中可以在设备获得，未获得之前可以设成 8
0x04	15	H 表示当前 qehad 是链表中第一个，整个 qhead 链表中有且仅有一个 qehad 需要设 H=1
0x04	14	dtc，data toggle control，设 1 表示用 qtd 的 DT 位作为当前传输的 data toggle 位；0 表示由硬件维护 data toggle；在 CTRL 传输时，data toggle 有些特殊需要用到 (详见 usb2 协议 8.5.3)
0x04	13:12	EPS, 表示端点速度，全速 00，低速 01，高速 10，11 为 reserved
0x04	11:8	EndPt, 表示当前 qhead 对应的端点号
0x04	7	I 是同步队列需要用的，对于 BULK/CTRL 传输设 0
0x04	6:0	DeviceAddress 表示当前 qhead 对应的设备地址
0x08	31:0	用于同步传输以及 hub 相关，简单应用时设 0
0x0C	31:5	Current qTD Pointer, 硬件用于维护当前传输的 qTD 地址，软件初始化时设 0
0x10	31:5	Next qTD Pointer，表示下一个 qTD 地址
0x10	0	T 是 Terminate 的简称，设 0 表示当前地址有效, 软件初始化 qhead 时，如果没有需要传输的数据，需要设 T 为 1
0x14	31:0	简单应用设 1 即可 (或者 T=1，其他部分随意, 比如 0xbadbadff)
0x18	31	dt，data toggle，当 qhead.dtc 为 1 时，硬件用这一位作为当前传输的起始 data toggle，简单应用时仅用于 CTRL 传输
0x18	30:16	Total Bytes Transfer，当前 dTD 需要传输的数据长度，最大支持 20Kbyte，建议最大 16Kbyte。 对于 OUT，软件初始化为需要发送的数据，硬件在每传输一个数据包 (长度为 Maximum Packet Length, 最后一个数据包除外) 后递减，当传输完成后应当为 0； 对于 IN，软件初始化为接收空间的大小，硬件在每传输一个数据包后更新这个值，当为 0 或者收到 short packet(数据包长度小于 Maximum Packet Length) 后当前 qTD 传输完成，结果可能不为 0
0x18	15	ioc，Interrup On Complete，设 1 表示当前 qTD 传输完成后置位 USBSTS.UI，用于产生中断，对于多 qTD 的传输，软件可以仅置位最后一个 qTD 的 ioc 位，这样可以减少中断次数提高传输效率

字节	位	描述
0x18	14:12	CPage，数据 buffer 的指针，软件需要初始化为 0，硬件会实时更新，表示当前数据在哪个 buffer(有效值为 0 到 4，对应 qTD 中 Buffer Pointer Page 0 到 4)
0x18	11:10	Cerr，传输错误计数器，软件需初始化为 3，每次传输错误会减一，到 0 后当前 qTD 会结束并置位 USBSTS.UEI, 用于产生出错中断
0x18	9:8	PID Code，当前 qtd 传输的 PID，有效值为：OUT-00，IN-01，SETUP-10
0x18	7:0	Status，用于表示当前 qTD 状态，软件需要初始化成 0x80. bit7：Active，用于表示当前 qTD 是否有效，硬件仅在 Active=1 时会认为有效而进行传输，当传输完成后会将 Active 清零； bit6：Halt，当硬件碰到严重错误 (babble，Cerr 计数到 0，收到 STALL) 时，硬件将 Halt 置 1，同时将 Active 清零； bit5：Data Buffer Error，当发生 overrun 或者 underrun 时会置 1； bit4：Babble，硬件检测到 babble(USB2 协议,8.7.4) 情况时置 1； bit3：XactErr，表示发生传输错误，具体错误类型详见 EHCI 协议 4.15.1.1； bit2:0 在简单应用时可忽略。
0x1C	31:12	Buffer Pointer (Page 0)，数据包起始地址，由 buffer0 和 current offset 指定，硬件在传输完 buffer0 的数据后，会切换地址到 buffer1 开始传输，5 个 buffer 可以是连续地址也可以不连续，只需要 4Kbyte 对齐即可
0x1C	11:0	Current Offset，软件初始化为数据包在 buffer0 的起始偏移量，硬件传输过程中会实时更新为当前 buffer(由 cpage 指定) 的偏移量，表示已经传输了多少数据
0x20 至 0x2F	31:12	qhead 余下部分，简单应用时只需配置数据 buffer pointer 地址即可 (page1,2,3,4)

表 215: qhead 结构

qtd 结构如图 75所示, 内容和 qhead overlay 部分相同，软件在需要传输数据时，需要初始化好一个或多个 qtd，将第一个 qtd 地址写到 qhead.Next qTD Pointer 即可。

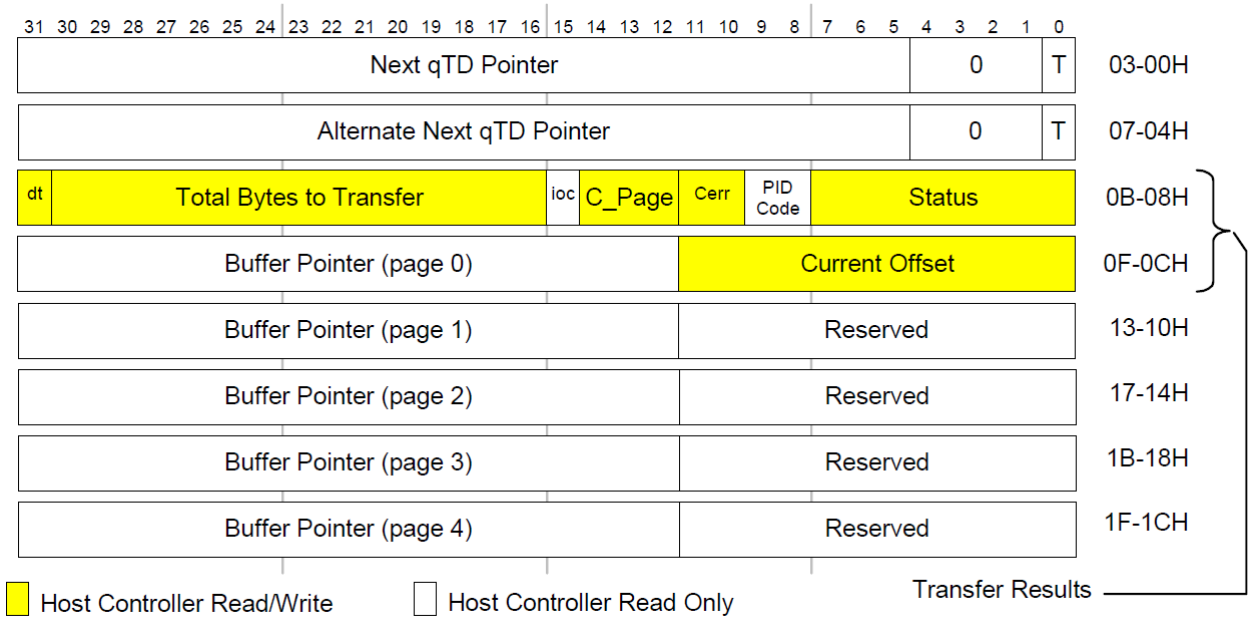


图 75: qtd 示意图

52.4.2 设备数据结构

设备数据结构指软件在存储空间中分配的一段空间，由以下几个部分组成：

- dqhlist，存储所有 dqh，每个端点对应两个 dqh(IN 和 OUT)，每个 dqh 占用 64-byte，dqhlist 的起始地址必须 2Kbyte 对齐。
- dqh，用于配置该端点参数，包含最大数据包长度，当前以及下一个传输描述符地址，同时会保存当前传输描述符内容，如图 77所示。详细信息如**??**所示。
- dtd，传输描述符，用于传输数据，保护数据长度，起始地址，状态等信息, 如图 78所示

如图 76所示，寄存器的 ENDPTLISTADDR，指向 dqhlist 的起始地址。软件需要在启动 USB(USBCMD.RS=1) 前准备好至少端点 0 的接收 dqh(EP0 的 OUT dqh，位于 dqhlist 起始地址)，用于接收主机发送的枚举信息中第一个 SETUP 数据包，之后根据接收到的数据，准备相应端点的 dqh 以及 dtd。

这里以接收枚举的第一个 SETUP 数据包为例，简单介绍软件流程：

- 给 dqhlist 准备一块 2Kbyte 对齐的空间，起始地址写入 ENDPTLISTADDR，大小为设备所需最大端点号乘 128byte。比如一个 mass storage 设备，除端点 0 外还需要端点 1 作为 IN，端点 2 和端点 3 作为 OUT，那么 dqhlist 需要 4*128=512byte; 极端情况某设备仅需要端点 7 作为 IN，那么 dqhlist 需要 8*128=1Kbyte；
- 准备端点 0 的接收 dqh(地址位于 dqhlist+0x00)，Maximum Packet Length 设置成本设备支持的最大数据包长度，一般低速为 8，全速或高速为 64；ios 设 1 用于收到 SETUP 后产生中断；T=1；Status=0；其他部分随意；
- 准备端点 0 的发送 dqh(地址位于 dqhlist+0x40)，Maximum Packet Length 设置成本设备支持的最大数据包长度，Status=0, 其他部分随意；
- 设置 USBINTR.UI=1, 等待中断；
- 收到 UI 中断后，SETUP 数据包的 8byte 内容会存放在端点 0 接收 dqh 的 Setup Buffer 中，根据 SETUP 内容准备响应；

- 假设收到的是 GET DESC，需要准备设备信息返回给主机，软件需要准备一个 IN dtd 返回数据，以及一个 0byte 的 OUT dtd 作为 Status 阶段的结束。
- 对于 IN dtd，T=1；TotalBytes 为设备信息字节数；Status=0x80；数据起始地址写入 buffer0 以及 Current Offset，如需要 (设备信息的内容跨 4K 地址) 可以准备 buffer1，其他部分为 0；
- 对于 OUT dtd，T=1，TotalBytes=0，Status=0x80，ioc=1，其他为 0；
- 将 IN dtd 地址写入端点 0 IN dqh 的 Next dTD Pointer 并清零 T，设置 ENDPTPRIME 的 bit16，等待 bit16 被硬件自动清零 (此时硬件会从 IN0 dqh 中读取相应内容，将需要发送的数据读入内部存储空间，结束后会清零 ENDPTPRIME 的对应 bit); 以上内容称为 PRIME EP0 IN。
- 同样做法，PRIME EP0 OUT
- 等待中断；如果正常完成，应该是收到 EP0 的 OUT 中断，因为以上流程中设置了 OUT dtd 的 ioc，没有设置 IN 的 ioc。

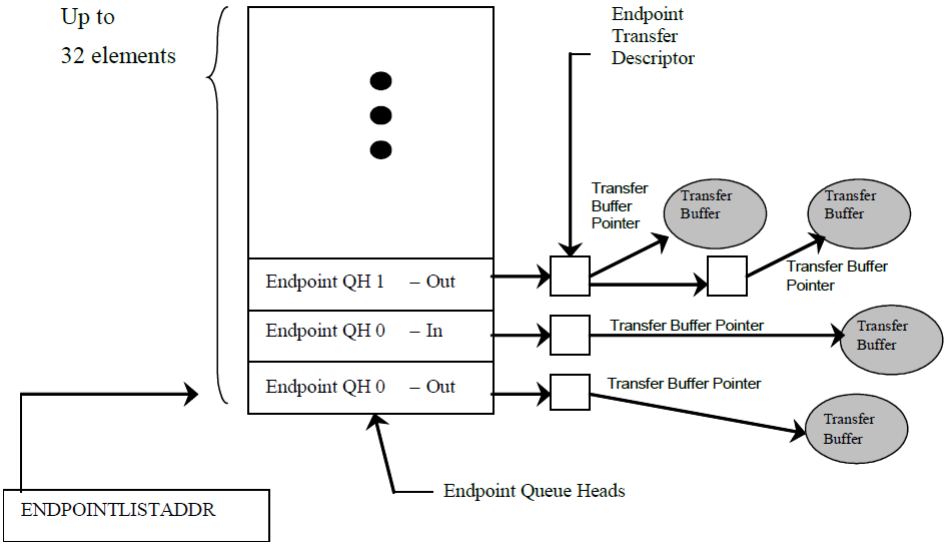


图 76: dqhlist 示意图

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00	word
mult		zlt	0		max_packet_size											ios	0															03-00h
cur_dtd_addr																												0				07-04h
nxt_dtd_addr																												0			T	0B-08h
0	total_bytes															ioc	c_page				multo	0		status								0F-0Ch
buffer(page 0)																					current_offset											13-10h
buffer(page 1)																					reserved											17-14h
buffer(page 2)																					reserved											1B-18h
buffer(page 3)																					reserved											1F-1Ch
buffer(page 4)																					reserved											23-20h
reserved																																27-24h
setup buffer bytes 3..0																																2B-28h
setup buffer bytes 7..4																																2F-2Ch

图 77: dqh 示意图

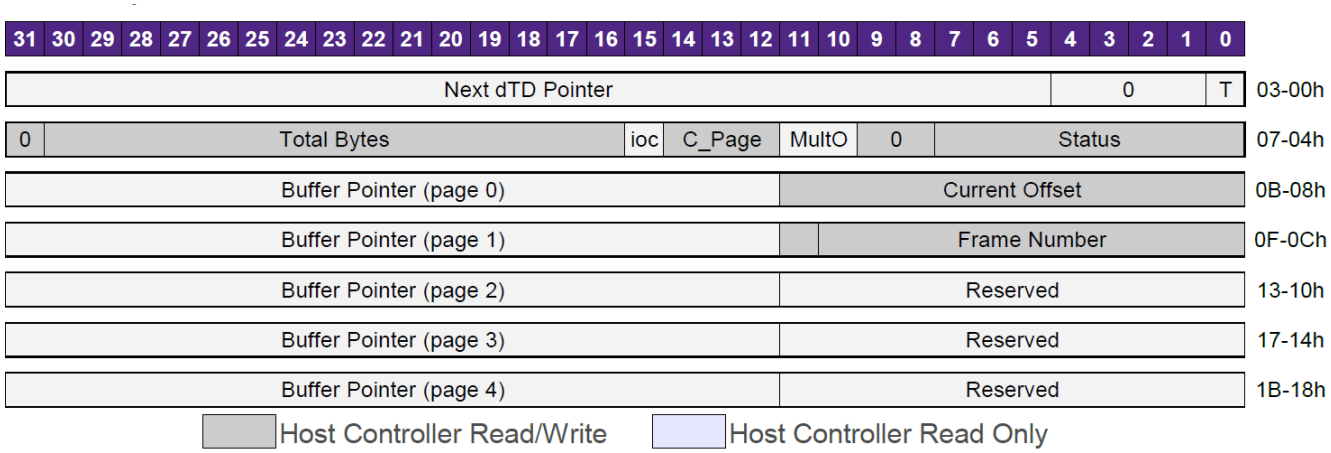


图 78: dtd 示意图

类似于主机 qhea 的 overlay 部分，设备 dtd 也会被硬件全部复制到 dqh 中再进行传输，所以 dqh 中一部分 (0x08 至 0x23) 和 dtd 完全一致。

设备数据结构 dqh 详解如下：

字节	位	描述
0x00	31:30	mult, ISO 端点，必须设置成 1/2/3，表示每个 dTD 可以传输多少数据包 非 ISO 端点，必须设置为 0。
0x00	29	zlt, 设 1 表示，当传输长度是最大数据包长度的整数倍时，最后加一个 0 长度数据包作为传输结束标志。仅用于非 ISO 端点
0x00	26：16	max_packet_size，端点的最大数据包长度。
0x00	15	ios, interrupt on setup, 设 1 表示，当 CTRL 端点收到 SETUP 数据包后会将 USBSTS.UI 置位.CTRL 端点的 dqh 需要设置此位
0x04	31:5	cur_dtd_addr, 当前 dqh 中正在传输的 dtd 物理地址，32 字节对齐。端点 prime 时，或者当一个 dtd 传输完成后，硬件会把 nxt_dtd_addr 复制到当前位置
0x08	31:5	nxt_dtd_addr, 下一个将要处理的 dtd 物理地址，32 字节对齐
0x08	0	T(Terminate), 表示 nxt_dtd_addr 地址是否有效,0 表示有效,1 表示当前 dtd 是最后一个，队列中没有下一个有效 dtd
0x0C	30:16	total_bytes, 表示当前 dtd 需要传输的长度。 对于 IN，表示需要传给主机的数据包长度； 对于 OUT，表示可以从主机接收的长度，实际接收的长度可能小于这个值.
0x0C	15	ioc，Interrup On Complete，设 1 表示，当前 dtd 传输完成后，会将 USBSTS.UI 置位
0x0C	14:12	CPage，当前 buffer，软件需要初始化成 0，硬件会在传输过程中实时更新
0x0C	11:10	multo，用于 dtd 覆盖 dqh 的 mult 部分. 例如，dqh.mult=3, max_packet_size=,512. 当需要传输 1023byte 数据时，软件需要将 dtd.multo 设成 2，这样会传输 2 个数据包：512+511； 如果软件不设置 multo(为 0 的话)，会传输 3 个数据包：512+511+0.

字节	位	描述
0x0C	7:0	Status，用于表示当前 dtd 状态，软件需要初始化成 0x80. bit7: Active, 表示当前 dtd 有效, 软件需要在准备 dtd 的时候将这个位设 1, 硬件在完成传输后会将这个位清零. Bit6: Halted. 表示当前端点发生处于 Halt 状态 bit5: data buffer error. 表示 buffer underrun(IN) 或者 overrun(OUT) bit3: transaction error. 表示数据传输有错误 bit2:0 在简单应用时可忽略。
0x10	31:12	Buffer Pointer (Page 0)，每个 dtd 有 5 个 buffer, 每个 4Kbyte, 可以是连续的物理地址 (buffer[n]=buffer[n-1]+0x1000, 也可以是离散的地址. Buffer page0 的起始地址由 current_offset 指定，其他 buffer 必须 4Kbyte 对齐. 最大可以传输 20Kbyte 数据, 如果起始地址不是 4Kbyte 对齐，建议不要超过 16Kbyte
0x10	11:0	Current Offset，软件初始化为数据包在 buffer0 的起始偏移量，硬件传输过程中会实时更新为当前 buffer(由 cpage 指定) 的偏移量，表示已经传输了多少数据
0x14 至 0x23	31:12	数据 buffer pointer 地址 (page1,2,3,4)
0x28 至 0x2F	31：0	setup buffer，dqh 的最后 8 字节，用于存放 SETUP 数据包，当收到 SETUP 包后，会将 8 字节 SETUP 数据存放在端点 0 的 OUT dqh 中

表 216: dqh 结构

52.5 USB 寄存器列表

USB0 base address: 0xF300C000

地址偏移	名称	描述	复位值
0x0080	GPTIMER0LD	General Purpose Timer #0 Load Register	0x00000000
0x0084	GPTIMER0CTRL	General Purpose Timer #0 Controller Register	0x00000000
0x0088	GPTIMER1LD	General Purpose Timer #1 Load Register	0x00000000
0x008C	GPTIMER1CTRL	General Purpose Timer #1 Controller Register	0x00000000
0x0090	SBUSCFG	System Bus Config Register	0x00000000
0x0140	USBCMD	USB Command Register	0x00080000
0x0144	USBSTS	USB Status Register	0x00000000
0x0148	USBINTR	Interrupt Enable Register	0x00000000
0x014C	FRINDEX	USB Frame Index Register	0x00000000
0x0154	DEVICEADDR	Device Address Register	0x00000000
0x0154	PERIODICLISTBASE	Frame List Base Address Register	0x00000000
0x0158	ASYNCLISTADDR	Next Asynch. Address Register	0x00000000
0x0158	ENDPTLISTADDR	Endpoint List Address Register	0x00000000

地址偏移	名称	描述	复位值
0x0160	BURSTSIZE	Programmable Burst Size Register	0x00000000
0x0164	TXFILLTUNING	TX FIFO Fill Tuning Register	0x00000000
0x0178	ENDPTNAK	Endpoint NAK Register	0x00000000
0x017C	ENDPTNAKEN	Endpoint NAK Enable Register	0x00000000
0x0184	PORTSC1	Port Status & Control	0x00000000
0x01A4	OTGSC	On-The-Go Status & control Register	0x00000000
0x01A8	USBMODE	USB Device Mode Register	0x00000000
0x01AC	ENDPTSETUPSTAT	Endpoint Setup Status Register	0x00000000
0x01B0	ENDPTPRIME	Endpoint Prime Register	0x00000000
0x01B4	ENDPTFLUSH	Endpoint Flush Register	0x00000000
0x01B8	ENDPTSTAT	Endpoint Status Register	0x00000000
0x01BC	ENDPTCOMPLETE	Endpoint Complete Register	0x00000000
0x01C0	ENDPTCTRL[ENDPTCTRL0]	Endpoint Control0 Register... Endpoint Control7 Register	0x00000000
0x01C4	ENDPTCTRL[ENDPTCTRL1]	Endpoint Control0 Register... Endpoint Control7 Register	0x00000000
0x01C8	ENDPTCTRL[ENDPTCTRL2]	Endpoint Control0 Register... Endpoint Control7 Register	0x00000000
0x01CC	ENDPTCTRL[ENDPTCTRL3]	Endpoint Control0 Register... Endpoint Control7 Register	0x00000000
0x01D0	ENDPTCTRL[ENDPTCTRL4]	Endpoint Control0 Register... Endpoint Control7 Register	0x00000000
0x01D4	ENDPTCTRL[ENDPTCTRL5]	Endpoint Control0 Register... Endpoint Control7 Register	0x00000000
0x01D8	ENDPTCTRL[ENDPTCTRL6]	Endpoint Control0 Register... Endpoint Control7 Register	0x00000000
0x01DC	ENDPTCTRL[ENDPTCTRL7]	Endpoint Control0 Register... Endpoint Control7 Register	0x00000000
0x0200	OTG_CTRL0		0x00000000
0x0210	PHY_CTRL0		0x00000000
0x0214	PHY_CTRL1		0x00000000
0x0220	TOP_STATUS		0x00000000
0x0224	PHY_STATUS		0x00000000

表 217: USB 寄存器列表

52.6 USB 寄存器描述

52.6.1 GPTIMER0LD (0x80)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD								GPTLD																							
N/A								RW																							
x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

GPTIMER0LD [31:0]

位域	名称	描述
23-0	GPTLD	通用定时器 0 重置值 当 GPTRST 位设置为“1b”时，这些位字段加载到 GPTCNT 位。 该值表示计时器持续时间的时间（微秒减 1）。 示例：对于一毫秒计时器，加载 1000-1=999 或 0x0003E7。 注：最大值为 0xFFFFFF 或 16.777215 秒。

GPTIMER0LD 位域

52.6.2 GPTIMER0CTRL (0x84)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GPTRUN	GPTRST	RSVD					GPTMODE	GPTCNT																							
RW	WO	N/A					RW	RO																							
0	0	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

GPTIMER0CTRL [31:0]

位域	名称	描述
31	GPTRUN	通用定时器 0 启动 0：计时器停止（保持原值） 1：计时器工作（倒计时） 注：GPTCNT 不会因为此位的设置或清除而变化
30	GPTRST	通用定时器 0 重置 0：无操作 1：将 GPTLD 的值加载到计时器 GPTCNT 中
24	GPTMODE	通用定时器 0 模式 0：单次模式，计时器将倒计时至零，生成中断，并停止； 1：重复模式，计时器将倒计时至零，生成中断并自动重新加载 GPTLD 位的值，继续倒计时。
23-0	GPTCNT	通用定时器 0 时间 此位表示当前通用定时器的值。

GPTIMER0CTRL 位域

52.6.3 GPTIMER1LD (0x88)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD								GPTLD																							
N/A								RW																							
x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

GPTIMER1LD [31:0]

位域	名称	描述
23-0	GPTLD	通用定时器 1 重置值

GPTIMER1LD 位域

52.6.4 GPTIMER1CTRL (0x8C)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GPTRUN	GPTRST	RSVD					GPTMODE	GPTCNT																							
RW	WO	N/A					RW	RO																							
0	0	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

GPTIMER1CTRL [31:0]

位域	名称	描述
31	GPTRUN	通用定时器 1 启动
30	GPTRST	通用定时器 1 重置
24	GPTMODE	通用定时器 1 模式
23-0	GPTCNT	通用定时器 1 时间

GPTIMER1CTRL 位域

52.6.5 SBUSCFG (0x90)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																													AHBBRST		
N/A																													RW		
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0

SBUSCFG [31:0]

位域	名称	描述
2-0	AHBBRST	AHB 主接口突发配置 这些位控制 AHB 主传输类型序列（或优先级）。 000-仅限未指定长度的增量突发，长度由 BURSTSIZE 寄存器决定 001-INCR4 突发，然后单次传输 010-INCR8 突发，INCR4 突发，然后单次传输 011-INCR16 突发、INCR8 突发、INCR4 突发，然后是单次传输 100-保留，不使用 101-增量突发，然后是未指定长度的增量突发 110-INCR8 突发、INCR4 突发，然后是未指定长度的增量突发 111-INCR16 突发、INCR8 突发、INCR4 突发，然后是未指定长度的增量突发

SBUSCFG 位域

52.6.6 USBCMD (0x140)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD								ITC								FS_2	ATDTW	SUTW	RSVD	ASPE	RSVD	ASP		RSVD	IAA	ASE	PSE	FS_1		RST	RS
N/A								RW								RW	RW	RW	N/A	RW	N/A	RW		N/A	RW	RW	RW	RW		RW	RW
x	x	x	x	x	x	x	x	0	0	0	0	1	0	0	0	0	0	0	x	0	x	0	0	x	0	0	0	0	0	0	0

USBCMD [31:0]

位域	名称	描述
23-16	ITC	中断阈值控制 系统软件使用此字段设置主机/设备控制器发出中断的最大速率。 ITC 包含以微帧为单位测量的最大中断间隔。有效值如下所示 00000000-立即（无阈值） 00000001-1 微帧 00000010-2 微帧 00000100-4 微帧 00001000-8 微帧 00010000-16 微帧 00100000-32 微帧 01000000-64 微帧

位域	名称	描述
15	FS_2	帧列表大小。[仅主机模式] 此字段指定帧列表的大小，用于控制帧索引寄存器中的哪些位应用于帧列表当前索引。 注：该字段由 USBCMD 位 15、3 和 2 组成。 0b000-1024 个元素（4096 字节）默认值 0b001-512 个元素（2048 字节） 0b010-256 个元素（1024 字节） 0b011-128 个元素（512 字节） 0b100-64 个元素（256 字节） 0b101-32 个元素（128 字节） 0b110-16 个元素（64 字节） 0b111-8 个元素（32 字节）
14	ATDTW	添加 dTD TripWire。[仅限设备模式] 此位用作标志位，以确保将新 dTD 正确添加到活动（已启动）端点的链接列表中。该位由软件设置和清除。 当状态机是危险区域时，硬件也会清除该位，在该危险区域中，向预处理端点添加 dTD 可能无法识别。
13	SUTW	设置 TripWire。[仅限设备模式] 此位用作标志位，以确保 DCD 从 QH 中提取 8 字节的设置数据有效负载而不会损坏。 如果设置锁定模式关闭（USB 核心寄存器 USBMODE 中的 SLOM 位，请参阅 USBMODE）， 则当 DCD 从 QH 复制上一个设置数据包的设置数据有效负载时，新的设置数据到达时存在危险。该位由软件设置和清除。 当检测到危险时，硬件也会清除该位。
11	ASPE	异步计划驻车模式启用。 软件使用此位启用或禁用驻车模式。当该位为 1 时，启用驻车模式。当该位为零时，驻车模式被禁用。
9-8	ASP	异步计划驻车模式计数。 包含在继续遍历异步调度之前，允许主机控制器从异步调度上的高速队列头执行的连续事务数的计数。有效值为 1 至 3。 当驻车模式启用为 1 时，软件不得将零写入该位，因为这将导致未定义的行为。

位域	名称	描述
6	IAA	异步前进门铃中断。 此位被软件用作门铃，以告知主机控制器在下次推进异步计划时发出中断。软件必须对此位写入 1 才能按门铃。 当主机控制器退出所有适当的缓存调度状态时，它会在 USBSTS 寄存器中设置异步前进状态位上的中断。 如果 USBINTR 寄存器中的同步提前启用中断位为 1，则主机控制器将在下一个中断阈值断言中断。 主机控制器将 USBSTS 寄存器中的中断同步高级状态位设置为 1 后，将该位设置为 0。 当异步计划处于非活动状态时，软件不应向该位写入 1。这样做将产生未定义的结果。 此位仅在主机模式下使用。选择设备模式时，将 1 写入该位将产生未定义的结果。
5	ASE	异步计划启用。 此位控制主机控制器是否跳过异步计划的处理。 只有主机控制器使用此位。 0-不要处理异步计划。 1-使用 ASYNCLISTADDR 寄存器访问异步计划。
4	PSE	定期计划启用-读/写。默认值为 0b。 此位控制主机控制器是否跳过处理定期计划。 只有主机控制器使用此位。 0-不处理定期计划 1-使用 PERIODICLISTBASE 寄存器访问定期计划。
3-2	FS_1	
1	RST	控制器复位。 软件使用此位重置控制器。复位过程完成后，主机/设备控制器将该位设置为零。软件无法通过向该寄存器写入零来提前终止重置过程。 主机操作模式： 当软件将一写入该位时，控制器将其内部管道、计时器、计数器、状态机等重置为初始值。USB 上当前正在进行的任何事务都将立即终止。USB 重置不会在下游端口上驱动。 当 USBSTS 寄存器中的 HCHARTED 位为零时，软件不应将该位设置为 1。尝试重置活动运行的主机控制器将导致未定义的行为。 设备操作模式： 当软件将一写入该位时，控制器将其内部管道、计时器、计数器、状态机等重置为初始值。当设备处于连接状态时，不建议将 1 写入该位，因为对连接主机的影响尚未定义。 为了确保在启动设备控制器重置之前设备未处于连接状态，应刷新所有已启动的端点，并将 USBCMD 运行/停止位设置为 0。

位域	名称	描述
0	RS	运行/停止。 1= 运行。0= 停止。 主机操作模式： 当设置为“1b”时，控制器继续执行计划。只要该位设置为 1，控制器就会继续执行。当该位设置为 0 时，主机控制器在 USB 上完成当前事务，然后停止。 状态寄存器中的 HC 暂停位指示控制器何时完成事务并进入停止状态。 除非控制器处于暂停状态（即 USBSTS 寄存器中的 HCHARTED 为 1），否则软件不应向该字段写入 1。 设备操作模式： 将 1 写入该位将导致控制器启用 D+ 上拉并启动连接事件。 此控制位不直接连接到上拉启用，因为在转换到高速模式时上拉将被禁用。在控制器正确初始化之前，软件应使用此位防止附加事件。将 0 写入此将导致分离事件。

USBCMD 位域

52.6.7 USBSTS (0x144)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD						TI1	TI0	RSVD				UPI	UAI	RSVD	NAKI	AS	PS	RCL	HCH	RSVD			SLI	SRI	URI	AAI	SEI	FRI	PCI	UEI	UI
N/A						RWC	RWC	N/A				RWC	RWC	N/A	RO	RO	RO	RO	RO	N/A			RWC	RWC	RWC	RWC	RWC	RWC	RWC	RWC	RWC
x	x	x	x	x	x	0	0	x	x	x	x	0	0	x	0	0	0	0	0	x	x	x	0	0	0	0	0	0	0	0	0

USBSTS [31:0]

位域	名称	描述
25	TI1	通用定时器 1 中断标志位 当定时器 1 倒计时到 0 后置位；
24	TI0	通用定时器 0 中断标志位 当定时器倒计时 0 到 0 后置位；
19	UPI	USB 主机周期队列传输完成中断标志位 当周期队列中的描述符传输完成时，且该标识符（TD）设置了完成中断（IOC）位，主机控制器将置位此位。 当检测到短数据包且该数据包处于周期队列时，主机也会设置该位。短数据包是指接收到的实际字节数小于预期的字节数。 设备控制器不使用该位，该位始终为零

位域	名称	描述
18	UAI	USB 主机异步队列传输完成中断标志位 当异步队列中的描述符传输完成时，且该标识符（TD）设置了完成中断（IOC）位，主机控制器将置位此位。 当检测到短数据包且该数据包处于异步计划时，主机也会设置该位。短数据包是指接收到的实际字节数小于预期的字节数。 设备控制器不使用该位，该位始终为零
16	NAKI	NAK 中断标志位 当设备某个端点回复 NAK，且 ENDPTNAKEN 中对应位为 1，则会置位此位。 当 ENDPTNAK 中所有位清零，此位自动清零。 仅用于设备模式。
15	AS	异步计划状态 此位报告异步计划的当前实际状态。为零时，表示异步计划状态为禁用，如果为一，则状态为启用。 当软件转换 USBCMD 寄存器中的异步计划启用位时，主机控制器无需立即禁用或启用异步计划。 当此位和异步计划启用位的值相同时，异步计划将启用（1）或禁用（0）。 仅在主机操作模式下使用。
14	PS	周期计划状态 此位报告周期计划的当前实际状态。为零时，表示周期计划被禁用，为一，则状态为启用。 当软件转换 USBCMD 寄存器中的周期计划启用位时，主机控制器无需立即禁用或启用周期计划。 当该位和周期计划启用位的值相同时，周期计划被启用（1）或禁用（0）。 仅在主机操作模式下使用。
13	RCL	回收 这是一个只读状态位，用于检测空的异步计划。 仅在主机操作模式下使用。
12	HCH	主机停止。 当运行/停止位为 1 时，该位为 0。由于运行/停止位被软件或控制器硬件设置为 0（例如，内部错误），控制器在停止执行后将该位设置为 1。 仅在主机操作模式下使用。
8	SLI	设备控制器挂起中断标志位 当控制器从活动状态进入暂停状态时，该位将设置为 1。设备控制器在从挂起状态退出时清除位。 仅在设备操作模式下使用。

位域	名称	描述
7	SRI	收到 SOF 中断标志位 当设备控制器检测到（微）帧开始时，该位将设置为 1。当 SOF 非常晚时，设备控制器将自动设置此位，以指示预期 SOF。 因此，该位在设备 FS 模式下大约每 1ms 设置一次，在 HS 模式下每 125ms 设置一次，并与接收到的实际 SOF 同步。 由于设备控制器在连接前已初始化为 FS，因此在连接和啁啾的前奏中，该位将以 1ms 的间隔设置。 在主机模式下，该位将每 125us 设置一次，并可由主机控制器驱动程序用作时基。
6	URI	已接收 USB 重置中断标志位 当设备控制器检测到 USB 重置并进入默认状态时，该位将设置为 1。 仅在设备操作模式下使用。
5	AAI	异步前进中断标志位 系统软件可通过在 USBCMD 寄存器中异步前进门铃位的中断中写入一个中断，强制主机控制器在下次主机控制器推进异步计划时发出中断。此状态位表示该中断源的断言。 仅在主机操作模式下使用。
4	SEI	系统错误中断标志位 当 DMA 访问系统总线，收到错误回复时置位。 一般发生于访问了不该访问的地址，由于描述符或寄存器配置错误导致
3	FRI	帧列表滚动中断标志位 当帧列表索引从其最大值滚动到零时，主机控制器将该位设置为 1。发生滚动的确切值取决于帧列表的大小。 例如如果帧列表大小（在 USBCMD 寄存器的帧列表大小字段中编程）为 1024，则每次 FRINDEX[13] 切换时，帧索引寄存器都会滚动。 类似地，如果大小为 512，则主机控制器每次切换 FHINDEX[12] 时将该位设置为 1。 仅在主机操作模式下使用。
2	PCI	端口更改中断标志位 当在任何端口上发生连接状态、端口启用/禁用更改或强制端口恢复位作为挂起端口上 J-K 转换的结果而设置时，主机控制器将该位设置为 1。 当端口控制器进入全速或高速操作状态时，设备控制器将该位设置为 1。 当端口控制器由于复位或挂起事件退出完全或高速操作状态时，通知机制分别为 USB 复位接收位和 DCSuspend 位。

位域	名称	描述
1	UEI	USB 错误中断标志位 当 USB 事务的完成导致错误情况时，该位由主机/设备控制器设置。如果发生错误中断的 TD 也设置了完整中断（IOC）位，则该位与 USBINT 位一起设置。
0	UI	USB 传输中断标志位 当 USB 传输完成，且其传输描述符（TD）设置了完成中断（IOC）位。 当检测到短数据包时，主机控制器也会设置该位。短数据包是指接收到的实际字节数小于预期的字节数。

USBSTS 位域

52.6.8 USBINTR (0x148)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD						TIE1	TIE0	RSVD				UPIE	UAIE	RSVD	NAKE	RSVD							SLE	SRE	URE	AAE	SEE	FRE	PCE	UEE	UE
N/A						RW	RW	N/A				RW	RW	N/A	RO	N/A							RW	RW	RW	RW	RW	RW	RW	RWC	RW
x	x	x	x	x	x	0	0	x	x	x	x	0	0	x	0	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0

USBINTR [31:0]

位域	名称	描述
25	TIE1	通用定时器 1 中断使能
24	TIE0	通用定时器 0 中断使能
19	UPIE	USB 主机周期队列传输完成中断使能
18	UAIE	USB 主机异步队列传输完成中断使能
16	NAKE	NAK 中断使能 当此位置位且 USBSTS 中的 NAKI 置位，则产生中断
8	SLE	设备控制器挂起中断使能 仅用于主机模式
7	SRE	收到 SOF 中断使能
6	URE	收到 USB 重置中断使能
5	AAE	异步前进中断使能
4	SEE	系统错误中断使能
3	FRE	帧列表滚动中断使能
2	PCE	端口更改中断使能
1	UEE	USB 错误中断使能
0	UE	USB 传输中断使能

USBINTR 位域

52.6.9 FRINDEX (0x14C)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																		FRINDEX													
N/A																		RW													
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0

FRINDEX [31:0]

位域	名称	描述
13-0	FRINDEX	帧索引。 该寄存器中的值在每个时间帧（微帧）结束时递增。位 [N:3] 用于帧列表当前索引。这意味着在移动到下一个索引之前，帧列表的每个位置都会被访问 8 次（帧或微帧）。 以下说明了在主机模式下使用时，基于 USBCMD 寄存器中帧列表大小字段的值 N 的值。 USBCMD[帧列表大小] 数字元素 N 在设备模式下，该值是最后发送的帧的当前帧编号。它不用作索引。 在任一模式下，位 2:0 表示当前微帧。 下面的位字段值描述表示为（帧列表大小）数字元素 N。 00000000000000 - (1024) 12 00000000000001 - (512) 11 00000000000010 - (256) 10 00000000000011 - (128) 9 00000000000100 - (64) 8 00000000000101 - (32) 7 00000000000110 - (16) 6 00000000000111 - (8) 5

FRINDEX 位域

52.6.10 DEVICEADDR (0x154)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
USBADR							USBADRA	RSVD																							
RW							RW	N/A																							
0	0	0	0	0	0	0	0	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x

DEVICEADDR [31:0]

位域	名称	描述
31-25	USBADR	设备地址 此字段表示 USB 设备地址，仅用于设备模式
24	USBADRA	设备地址提前。 当该位为“0”时，对 USBADR 的任何写入都是瞬时的。当该位同时或在写入 USBADR 之前写入“1”时，对 USBADR 字段的写入被暂存并保存在隐藏寄存器中。 在端点 0 上发生 IN 并确认后，将从保留寄存器加载 USBADR。在以下情况下，硬件将自动清除该位： 1）中的已确认为终结点 0。（USBADR 从暂存寄存器更新）。 2）对终结点 0 执行出/设置。（USBADR 未更新）。 3）发生设备重置（USBADR 重置为 0）。 注：在 SET_ADDRESS 描述符的状态阶段之后，DCD 有 2 毫秒的时间对 USBADR 字段进行编程。当 DCD 无法在设置地址状态阶段 2 毫秒内写入设备地址时，该机制将确保满足该规范。 如果 DCD 在 SET_ADDRESS 数据阶段之后（在状态阶段开始之前）以 USBADRA=1 写入 USBADR，则 USBADR 将在正确的时间被更新，以满足 2ms USB 要求。

DEVICEADDR 位域

52.6.11 PERIODICLISTBASE (0x154)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BASEADR																				RSVD											
RW																				N/A											
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	x	x	x	x	x	x	x	x	x	x	x	x

PERIODICLISTBASE [31:0]

位域	名称	描述
31-12	BASEADR	该 20 位寄存器包含系统内存中周期序列表的起始地址的高 20 位，低 12 位必须为 0。软件需在主机控制器开始执行调度之前初始化此寄存器。 此物理内存指针引用的内存结构为 4 KB 对齐。该寄存器的内容与帧索引寄存器（FRINDEX）相结合，以使主机控制器能够按顺序单步执行周期帧列表。 仅用于主机模式

PERIODICLISTBASE 位域

52.6.12 ASYNCLISTADDR (0x158)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ASYBASE																											RSVD					
RW																											N/A					
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	x	x	x	x	x

ASYNCLISTADDR [31:0]

位域	名称	描述
31-5	ASYBASE	此寄存器包含主机要执行的下一个异步队列头的地址高 27 位，低 5 位必须为 0（32 字节对齐）。 仅用于主机模式

ASYNCLISTADDR 位域

52.6.13 ENDPTLISTADDR (0x158)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EPBASE																					RSVD										
RW																					N/A										
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	x	x	x	x	x	x	x	x	x	x	x

ENDPTLISTADDR [31:0]

位域	名称	描述
31-11	EPBASE	端点列表基地址 此地址必须 2KB 对齐，对应于最多 16 个 dQH

ENDPTLISTADDR 位域

52.6.14 BURSTSIZE (0x160)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																TXPBURST								RXPBURST							
N/A																RW								RW							
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

BURSTSIZE [31:0]

位域	名称	描述
15-8	TXPBURST	发送可配置的突发长度 当 AHBBRST 配置为仅限未指定长度的增量突发时，发送时 DMA 访问内存的突发长度由此寄存器配置
7-0	RXPBURST	接收可配置的突发长度 当 AHBBRST 配置为仅限未指定长度的增量突发时，接收时 DMA 访问内存的突发长度由此寄存器配置

BURSTSIZE 位域

52.6.15 TXFILLTUNING (0x164)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD										TXFIFOTHRES						RSVD			TXSCHHEALTH					RSVD	TXSCHOH						
N/A										RW						N/A			RWC					N/A	RW						
x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	x	x	x	0	0	0	0	0	x	0	0	0	0	0	0	0

TXFILLTUNING [31:0]

位域	名称	描述
21-16	TXFIFOTHRES	FIFO 突发阈值。 该寄存器控制在数据包开始传输到总线之前，在主机模式下发送到 TX 延迟 FIFO 的数据突发数。最小值为 2，该值应尽可能低，以最大限度地提高 USB 性能。 在延迟不可预测和/或带宽不足的系统中，可以使用更高的值，因为从延迟 FIFO 传输到 USB 的数据发生在从系统内存补充之前，FIFO 可能运行不足。 如果设置了 USBMODE 寄存器中的 SDIS 位，则忽略此值。
12-8	TXSCHHEALTH	调度程序运行状况计数器。 当主机控制器在下一帧开始前发送数据包的时间不足之前，未能将 TX 延迟 FIFO 填充到 TXFITOTHRES 编程的水平时，该寄存器会增加。 此健康计数器测量发生这种情况的次数，以便为选择合适的 TXSCHOH 提供反馈。写入此寄存器将清除计数器，此计数器最大值为 31。

位域	名称	描述
6-0	TXSCHOH	调度程序开销 该寄存器向上述计划时间估计器添加一个额外的固定偏移量，称为 Tff。作为近似值， 为该寄存器选择的值应将 TXSCHHEALTH 中捕获的退避事件数量限制在高利用率总线中小于每秒 10 个。不需要为此寄存器选择过高的值，因为它会不必要地降低 USB 利用率。 当设备以高速模式连接时，该寄存器中表示的时间单位为 1.267us。当设备以低速/全速模式连接时，该寄存器中表示的时间单位为 6.333us。

TXFILLTUNING 位域

52.6.16 ENDPTNAK (0x178)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD								EPTN								RSVD								EPRN							
N/A								RWC								N/A								RWC							
x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0

ENDPTNAK [31:0]

位域	名称	描述
23-16	EPTN	发送端点 NAK 状态位 当设备某端点收到主机的 IN 后回复 NAK，则此位置位。 每个端点对应一位。 软件写 1 清零该位
7-0	EPRN	接收端点 NAK 状态位 当设备某端点收到主机的 OUT 或 PING 后回复 NAK，则此位置位。 每个端点对应一位。 软件写 1 清零该位

ENDPTNAK 位域

52.6.17 ENDPTNAKEN (0x17C)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD								EPTNE								RSVD								EPRNE							
N/A								RW								N/A								RW							
x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ENDPTNAKEN [31:0]																															

位域	名称	描述
23-16	EPTNE	发送端点 NAK 中断使能 当某发送端点回复 NAK，且对应的该位置位，则置位 USBSTS 中 NAK 中断状态位。 每个端点对应一位
7-0	EPRNE	接收端点 NAK 中断使能 当某接收端点回复 NAK，且对应的该位置位，则置位 USBSTS 中 NAK 中断状态位。 每个端点对应一位

ENDPTNAKEN 位域

52.6.18 PORTSC1 (0x184)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD		STS	PTW	PSPD		RSVD	PFSC	PHCD	WKOC	WKDC	WKCN	PTC				RSVD			PP	LS		HSP	PR	SUSP	FPR	OCC	OCA	PEC	PE	CSC	CCS
N/A		RW	RW	RO		N/A	RW	RW	RW	RW	RW	RW				N/A			RW	RO		RO	RW	RW	RW	RW	RO	RWC	RWC	RWC	RWC
x	x	0	0	0	0	x	0	0	0	0	0	0	0	0	0	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0

PORTSC1 [31:0]

位域	名称	描述
29	STS	串行收发器选择 1：选择串行接口引擎 0：选择并行接口信号 当该位设置为“1b”时，将使用串行接口引擎而不是并行接口信号。
28	PTW	并行收发器接口宽度 0：选择 8 位 UTMI 端口（60MHz） 1：选择 16 位 UTMI 端口（30MHz） 选择串行收发器时此位无效
27-26	PSPD	端口速度 此寄存器反应当前正在工作的端口速度 00b ： 全速 01b ： 低速 10b ： 高速 11b ： 无效

位域	名称	描述
24	PFSC	强制全速连接 当此位被设置时，端口强制工作在全速状态。 作为主机，会在 USB 复位的速度协商过程中，不输出 ChirpKJ 序列； 作为设备，会在 USB 复位的速度协商过程中，不输出 ChirpK。 0 ：常规工作 1 ：强制全速连接
23	PHCD	PHY 低功耗模式 设置此位，会拉低输出到 PHY 的 suspendm 信号，让 PHY 进入低功耗模式。 主机模式时，由软件决定何时进入低功耗模式； 设备模式时，当收到 SLI 中断后，可以由软件设置此位进入低功耗模式。 当检测到唤醒信号时，硬件自动清掉此位。
22	WKOC	过流唤醒使能 当 USB 处于低功耗模式，当检测发生过载时，设置此位会产生唤醒事件 仅用于主机模式
21	WKDC	断开连接唤醒使能 当 USB 处于低功耗模式，当检测设备断开连接时，设置此位会产生唤醒事件 仅用于主机模式
20	WKCN	断开连接唤醒使能 当 USB 处于低功耗模式，当检测到设备连接时，设置此位会产生唤醒事件 仅用于主机模式

位域	名称	描述
19-16	PTC	端口测试控制-读/写。默认值 =0000b。 有关使用这些测试模式的操作模型，请参阅端口测试模式；有关每个测试模式的详细信息，请参阅 USB 规范版本 2.0 第 7 章。 强制启用 FS 和强制启用 LS 是 EHCI 规范中指定的测试模式支持的扩展。将 PTC 字段写入任意 FORCE_ENABLE_{HS/FS/LS} 值将强制端口以所选速度进入已连接和已启用状态。将 PTC 字段写回 TEST_MODE_DISABLE 将允许端口状态机从此点正常运行。 注意：设备不支持低速操作。 除零以外的任何其他值表示端口正在测试模式下运行。 0000-测试模式禁用 0001-J 状态 0010-K 状态 0011-SE0（主机）/NAK（设备） 0100-数据包 0101-强制启用高速 0110-强制启用全速 0111-强制启用低速 1000-1111-保留
12	PP	端口电源 主机控制器需要端口电源控制开关。该位表示开关的当前设置（0= 关闭，1= 打开）。当端口电源不可用时（即 PP 等于 0），该端口不起作用，不会报告连接、分离等。 当在通电端口上检测到过流情况时，主机控制器会将 PP 位从 1 转换为零（从端口断电）。 仅用于主机模式，设备模式时此位无意义。
11-10	LS	总线状态 表示当前 USB 总线上的状态，bit11 表示 D+ 状态，bit10 表示 D-状态。 00 ： SE0 01 ： K 状态 10 ： J 状态 11 ： 未定义 注意，以上状态在低速和全速时可用；当 USB 工作于高速状态时，以上定义需需参考 UTMI 协议
9	HSP	高速端口 当此位置 1 时，表示当前主机或设备连接成高速模式。 此位是 PSPD(bit27,bit26) 的冗余，用户也可从 PSPD 中获取当前信息。

位域	名称	描述
8	PR	端口复位 在主机模式下：该位可读写。 1：端口处于复位状态。 0：端口未处于重置状态 当软件将该位置 1 时，主机将根据 USB 规范版本 2.0 中定义，开始总线复位序列。复位顺序完成后，该位将自动变为零。 此行为与 EHCI 不同，EHCI 要求主机控制器驱动程序在驱动程序中定时重置持续时间后将此位设置为零。 在设备模式下：该位为只读状态位。USB 总线的设备复位也在 USBSTS 寄存器中指示。
7	SUSP	挂起-读/写或只读。默认值 =0b。 1= 端口处于挂起状态。0= 端口未处于挂起状态。 在主机模式下：读/写。 此寄存器的端口启用位（PE，bit2）和挂起位（SUSP，bit7）定义端口状态，如下所示： 端口状态 0x：禁用 10：启用 11：挂起 处于挂起状态时，此端口的下行数据传播被阻止，端口复位除外。如果在将该位写入 1 时正在进行数据传输，则会在当前传输结束后进入挂起状态。在挂起状态下，端口对恢复检测敏感会检测恢复（resume）信号。请注意，在端口挂起之前，位状态不会改变，如果 USB 上当前正在进行事务，则挂起端口可能会有延迟。 当软件将 Force Port Resume 位设置为零时，主机控制器将无条件地将该位设置为零。主机控制器忽略对此位的零写入。 如果主机软件在端口未启用时将该位设置为 1（即，端口启用位 PE 为零），则结果未定义。 如果在主机模式下端口电源（PP）为零，则此字段为零。 在设备模式下：只读。 在设备模式下，该位为只读状态位。

位域	名称	描述
6	FPR	强制端口恢复。1= 在端口上检测到/驱动恢复。0= 在端口上未检测到恢复（K 状态）。 在主机模式下： 软件将该位设置为 1，以驱动恢复信号。如果在端口处于挂起状态时检测到 J-to-K 转换，主机控制器将此位设置为 1。 当此位由于检测到 J-to-K 转换而转换为 1 时，USBSTS 寄存器中的端口更改检测位也设置为 1。 恢复序列完成后，该位将自动变为零。 此行为与 EHCI 不同，EHCI 要求主机控制器驱动程序在驱动程序中计时恢复持续时间后将此位设置为零。 请注意，当主机控制器拥有该端口时，恢复序列遵循 USB 规范修订版 2.0 中定义的序列。 只要该位保持为 1，就在端口上驱动恢复信令（全速“K”）。该位将保持为 1，直到端口切换到高速空闲。 写入零无效，因为端口控制器将计时恢复操作，并在恢复时序结束，端口状态切换到 HS 或 FS idle 时清零该位。 如果在主机模式下端口电源（PP）为零，则此字段为零。 此位与 EHCI 不兼容。 在设备模式下： 设备处于暂停状态 5 毫秒或更长时间后，软件必须将该位设置为 1，以在清除之前的恢复信号。 如果在端口处于挂起状态时检测到 J-to-K 转换，则设备控制器将此位设置为 1。 当设备恢复正常运行时，该位将被清除。 此外，当由于检测到 K-to-J 转换而清除该位时，USBSTS 寄存器中的端口更改检测位也设置为 1。
5	OCC	过流状态改变 当硬件检测到过流时会置位此位 软件需要写 1 清除该位。
4	OCA	过流状态 此位表示当前过流状态 0：端口当前没有过流发生 1：端口处于过流状态 当过流结束时，此位会自动清零
3	PEC	端口启用变化 1：端口启用状态发生过变化（从禁止到启用，或者从启用到禁止） 0：端口启用状态无变化 软件写 1 清零该位 如果在主机模式下端口电源（PP）为零，则此字段为零。 设备模式端口永远启用，所以该位一直为 0.

位域	名称	描述
2	PE	端口启用/禁用。1= 启用。0= 禁用。 在主机模式下： 端口只能由主机控制器作为重置和启用的一部分启用。软件无法通过写 1 来启用端口。 端口可以由故障条件（断开事件或其他故障条件）或主机软件禁用（软件可以通过写 0 禁用端口）。 请注意，在端口状态实际更改之前，位状态不会更改。由于总线传输，禁用或启用端口时可能会有延迟。 当端口被禁用时，（0b）数据的下游传播被阻止，重置除外。 如果在主机模式下端口功率（PORTSC1）为零，则此字段为零。 在设备模式下： 设备端口始终处于启用状态，因此此位始终为“1b”。
1	CSC	连接状态更改。1= 当前连接状态更改。0= 没有变化。 在主机模式下： 表示端口的当前连接状态发生了更改。主机控制器会在端口设备连接状态有更改时设置此位，即使系统软件尚未清除现有连接状态更改。 例如，在系统软件清除更改的条件之前，插入状态更改两次，集线器硬件将“设置”一个已设置的位（即，该位将保持设置）。软件通过向其写入 1 来清除该位。 如果在主机模式下端口电源（PP）为零，则此字段为零。 在设备模式下： 此位在设备控制器模式下未定义。
0	CCS	当前连接状态。 在主机模式下： 1= 端口上存在设备。0= 不存在任何设备。默认值 =0。此值反映端口的当前状态，可能与导致设置连接状态更改位（位 1）的事件不直接对应。 如果在主机模式下端口电源（PP）为零，则此字段为零。 在设备模式下： 1 表示设备已成功连接，并且正在高速或全速运行，如该寄存器中的高速端口位所示。 0 表示设备未成功连接或被软件向 USBCMD 寄存器中的运行位写入零而强制断开连接。

PORTSC1 位域

52.6.19 OTGSC (0x1A4)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD					ASVIE	AVVIE	IDIE	RSVD					ASVIS	AVVIS	IDIS	RSVD					ASV	AVV	ID	RSVD		IDPU	RSVD			VC	VD

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
N/A					RW	RW	RW	N/A					RWC	RWC	RWC	N/A					RO	RO	RO	N/A		RW	N/A			RW	RW
x	x	x	x	x	0	0	0	x	x	x	x	x	0	0	0	x	x	x	x	x	0	0	0	x	x	0	x	x	x	0	0

OTGSC [31:0]

位域	名称	描述
26	ASVIE	session valid 变化中断使能
25	AVVIE	vbus valid 变化中断使能
24	IDIE	ID 变化中断使能
18	ASVIS	session valid 中断状态 软件写 1 清零该位
17	AVVIS	vbus valid 中断状态。 注意，当 ID 为 1 时，不会产生 vbus 变化中断，用户可以使用 session valid 中断替代 软件写 1 清零该位
16	IDIS	ID 变化中断状态 软件写 1 清零该位
10	ASV	session valid 状态位 表示 VBUS 高于 session valid 阈值 注意：此位通过去抖电路，与实际 vbus 状态会有 1 到 2 毫秒的延迟，用户可以在 PHY_STATUS 寄存器中读到实时状态
9	AVV	vbus valid 状态位 表示 VBUS 高于 vbus valid 阈值 注意：此位通过去抖电路，与实际 vbus 状态会有 1 到 2 毫秒的延迟，用户可以在 PHY_STATUS 寄存器中读到实时状态
8	ID	ID 状态位 注意：此位通过去抖电路，与实际 ID 状态会有 1 到 2 毫秒的延迟，用户可以在 PHY_STATUS 寄存器中读到实时状态
5	IDPU	ID 上拉使能 设 1 上拉 ID
1	VC	VBUS 充电
0	VD	VBUS 放电

OTGSC 位域

52.6.20 USBMODE (0x1A8)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																											SDIS	SLOM	ES	CM	
N/A																											RW	RW	RW	RW	
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0

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

USBMODE [31:0]

位域	名称	描述
4	SDIS	流禁用模式。 设备模式：设置为“1”将禁用低带宽系统的 RX 和 TX 上的双重启动。 此模式确保当 RX 和 TX 缓冲区足以包含整个数据包时，禁用标准双缓冲方案以防止带宽受限系统中的超限/欠载。 注意：在高速模式下，当流禁用激活时，所有接收到的数据包都会通过 NYET 握手进行响应。 主机模式：设置为“1”可确保在 RX 和 TX 缓冲区足以容纳整个数据包的低带宽系统中，消除延迟 FIFO 的超限/不足。启用 stream disable 还可以确保在将数据包发送到 USB 之前将发送数据预取到内部缓存。 注意：使用此功能会限制整体 USB 性能，建议仅在系统带宽不足时使用。
3	SLOM	SETUP 锁定模式 仅用于设备模式，控制 SETUP 传输的锁定机制 0：SETUP 锁定打开 0：SETUP 锁定关闭，软件需要使用 USBCMD 寄存器中的 SUTW
2	ES	Endian 选择。 此位可以更改传输缓冲区的字节对齐方式，以匹配主机微处理器。微处理器接口中的位字段和数据结构不受该位值的影响，因为它们基于 32 位字。 0-小端 [默认值] 1-大端
1-0	CM	控制器模式 默认为空闲状态。 在控制器复位后，仅能写一次。 软件必须在配置此寄存器前，先在 USBCMD.RST 位写 1 复位整个控制器。 00：空闲 01：保留 10：设备模式 11：主机模式

USBMODE 位域

52.6.21 ENDPTSETUPSTAT (0x1AC)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																								ENDPTSETUPSTAT							
N/A																								RWC							
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0

ENDPTSETUPSTAT [31:0]

位域	名称	描述
7-0	ENDPTSETUPSTAT	SETUP 端点状态 对于接收到的每个 SETUP 传输，该寄存器中的对应位设置为 1。软件从队列头读取 8 字节 SETUP 数据后，必须通过向相应位写入 1 来清除或确认 SETUP 传输 此寄存器仅在设备模式下使用。

ENDPTSETUPSTAT 位域

52.6.22 ENDPTPRIME (0x1B0)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD								PETB								RSVD								PERB							
N/A								RWS								N/A								RWS							
x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0

ENDPTPRIME [31:0]

位域	名称	描述
23-16	PETB	端点发送缓冲区预取 对于每个端点使用相应的位请求为发送传输准备缓冲区，以便响应 USB 主机的传输请求。 当向端点队列头发布新的传输描述符时，软件应向相应位写入 1，硬件自动使用该位开始解析队列头中的新传输描述符，并准备传输缓冲区。 当相关端点成功启动时，硬件清除此位。 USB 协议中，传输数据是由主机发起，设备并不知道何时主机会发起传输数据要求（IN），因此当软件准备好发送数据后，需要通过此位，通知设备控制器，将数据从系统内存中搬到 USB 发送缓冲区，在搬完之前（或者发送缓冲区满之前），设备会对主机的请求（IN）回复 NAK。

位域	名称	描述
7-0	PERB	端点接收缓冲区准备 每当向端点队列头发布新的传输描述符时，软件应向相应位写入 1，硬件自动使用该位开始解析队列头中的新传输描述符，并准备接收缓冲区。 当相关端点成功启动时，硬件清除此位。 跟发送不同，对于接收数据，只要接收缓冲区有空间，就可以接收数据，所以设置此位，硬件仅需解析传输符即可。 在软件设置此位，硬件解析完传输符之前，设备会对主机的请求（OUT 或 PING）回复 NAK。

ENDPTPRIME 位域

52.6.23 ENDPTFLUSH (0x1B4)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD								FETB								RSVD								FERB							
N/A								RWS								N/A								RWS							
x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0

ENDPTFLUSH [31:0]

位域	名称	描述
23-16	FETB	刷新端点发送缓冲区 将一写入该寄存器中的一位会导致相关端点清除对应的发送缓冲区。 如果一个关联端点的数据包正在进行中，则该传输将继续，直到完成。 在端点刷新操作成功后，硬件清除此寄存器。
7-0	FERB	刷新端点接收缓冲区 将一写入该寄存器中的一位会导致相关端点清除对应的接收缓冲区。 如果一个关联端点的数据包正在进行中，则该传输将继续，直到完成。 在端点刷新操作成功后，硬件清除此寄存器。

ENDPTFLUSH 位域

52.6.24 ENDPTSTAT (0x1B8)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD								ETBR								RSVD								ERBR							

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
N/A								RO								N/A								RO							
x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0

ENDPTSTAT [31:0]

位域	名称	描述
23-16	ETBR	端点发送缓冲区就绪。 每个端点的一位表示各自端点发送缓冲区的状态。 硬件将该位设置为 1，作为从 ENDPTPRIME 寄存器中相应位接收命令的响应。 在 ENDPTPRIME 寄存器中设置一个位和端点指示就绪之间始终存在延迟。 此延迟时间根据当前 USB 通信量和 ENDPRIME 寄存器中设置的位数而变化。 USB 复位、USB DMA 系统或 ENDPTFLUSH 寄存器可清除缓冲区就绪状态。 注意：当 dTD 失效且 dQH 更新时，在硬件端点重新启动操作期间，硬件会立即清除这些位。
7-0	ERBR	端点接收缓冲区就绪。 每个端点的一位表示各自端点接收缓冲区的状态。 硬件将该位设置为 1，作为从 ENDPTPRIME 寄存器中相应位接收命令的响应。 在 ENDPTPRIME 寄存器中设置一个位和端点指示就绪之间始终存在延迟。 此延迟时间根据当前 USB 通信量和 ENDPRIME 寄存器中设置的位数而变化。 USB 复位、USB DMA 系统或 ENDPTFLUSH 寄存器可清除缓冲区就绪状态。 注意：当 dTD 失效且 dQH 更新时，在硬件端点重新启动操作期间，硬件会立即清除这些位。

ENDPTSTAT 位域

52.6.25 ENDPTCOMPLETE (0x1BC)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD								ETCE								RSVD								ERCE							
N/A								RWC								N/A								RWC							
x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0

ENDPTCOMPLETE [31:0]

位域	名称	描述
23-16	ETCE	端点发送完成状态位 每个位表示发生了发送事件（IN），软件应读取相应的端点队列以确定端点状态。 如果在传输描述符中设置了相应的 IOC 位，则该位与 USBINT 同时设置。 写入一将清除该寄存器中的相应位。
7-0	ERCE	端点接收完成状态位 每个位表示发生了接收事件（OUT/SETUP），软件应读取相应的端点队列以确定端点状态。 如果在传输描述符中设置了相应的 IOC 位，则该位与 USBINT 同时设置。 写入一将清除该寄存器中的相应位。

ENDPTCOMPLETE 位域

52.6.26 ENDPTCTRL (0x1C0 + 0x4 * n)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD								TXE	TXR	RSVD		TXT		RSVD	TXS	RSVD								RXE	RXR	RSVD		RXT		RSVD	RXS
N/A								RW	WS	N/A		RW		N/A	RW	N/A								RW	WS	N/A		RW		N/A	RW
x	x	x	x	x	x	x	x	0	0	x	x	0	0	x	0	x	x	x	x	x	x	x	x	0	0	x	x	0	0	x	0

ENDPTCTRL [31:0]

位域	名称	描述
23	TXE	发送端点使能 0：禁止 1：使能 端点仅在被配置（指 USB 枚举协议中的配置端点）后才能使能
22	TXR	发送数据 toggle 重置 写 1 重置发送数据 toggle 位，从 DATA0 开始发送。 当收到对该端点的配置事件时（通过枚举，从主机收到相应配置请求），软件必须对相应位写 1 以同步主机和设备之间的数据 PID（DATA0 或者 DATA1）

位域	名称	描述
19-18	TXT	发送端点类型 00：CTRL 01：ISO 10：BULK 11：INT 注意：当该发送端点被设置为非 CTRL（默认值）时，必须将对应的接收端点也设置成非 CTRL 类型，因为 CTRL 端点必须成对存在
16	TXS	发送端点暂停 0：端点正常 1：端点暂停 如果此端点配置为控制端点，则在收到设置请求时（SETUP 包），此位将自动清除，并且此位将继续由硬件清除，直到清除关联的 ENDPTSETUPSTAT 位。 软件可以将 1 写入该位，以强制端点向主机返回暂停握手，直到此位被软件清除或自动清除，如上所述，用于控制端点。 注意：[仅限控制端点类型]：在 ENDPTSETUPSTAT 开始清除和硬件继续清除该位之间有轻微延迟（最多 50 个时钟）。 在大多数系统中，设备软件不太可能观察到这种延迟。但是，如果在向其写入 1 后观察到该位未设置，则遵循以下步骤： 通过检查相关的 endptsetupstat 位，持续写入此暂停位，直到设置完毕或接收到新设置为止。
7	RXE	接收端点使能 0：禁止 1：使能
6	RXR	接收数据 toggle 重置 写 1 重置发送数据 toggle 位，从 DATA0 开始接收。 当收到对该端点的配置事件时（通过枚举，从主机收到相应配置请求），软件必须对相应位写 1 以同步主机和设备之间的数据 PID（DATA0 或者 DATA1）
3-2	RXT	接收端点类型 00：CTRL 01：ISO 10：BULK 11：INT 注意：当该接收端点被设置为非 CTRL（默认值）时，必须将对应的发送端点也设置成非 CTRL 类型，因为 CTRL 端点必须成对存在

位域	名称	描述
0	RXS	接收端点暂停 0：端点正常 1：端点暂停 如果此端点配置为控制端点，则在收到设置请求时（SETUP 包），此位将自动清除，并且此位将继续由硬件清除，直到清除关联的 ENDPTSETUPSTAT 位。 软件可以将 1 写入该位，以强制端点向主机返回暂停握手，直到此位被软件清除或自动清除，如上所述，用于控制端点。 注意：[仅限控制端点类型]：在 ENDPTSETUPSTAT 开始清除和硬件继续清除该位之间有轻微延迟（最多 50 个时钟）。 在大多数系统中，设备软件不太可能观察到这种延迟。但是，如果在向其写入 1 后观察到该位未设置，则遵循以下步骤： 通过检查相关的 endptsetupstat 位，持续写入此暂停位，直到设置完毕或接收到新设置为止。

ENDPTCTRL 位域

52.6.27 OTG_CTRL0 (0x200)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD						OTG_WKDPDMCHG_EN	RSVD					AUTORESUME_EN	RSVD	OTG_VBUS_WAKEUP_EN	OTG_ID_WAKEUP_EN	RSVD		OTG_VBUS_SOURCE_SEL	OTG_UTMI_SUSPENDM_SW	OTG_UTMI_RESET_SW	OTG_WAKEUP_INT_ENABLE	OTG_POWER_MASK	OTG_OVER_CUR_POL	OTG_OVER_CUR_DIS	RSVD		SER_MODE_SUSPEND_EN	RSVD			
N/A						RW	N/A					RW	N/A	RW	RW	N/A		RW	RW	RW	RW	RW	RW	RW	N/A		RW	N/A			
x	x	x	x	x	x	0	x	x	x	x	x	0	x	0	0	x	x	0	0	0	0	0	0	0	x	x	0	x	x	x	x

OTG_CTRL0 [31:0]

位域	名称	描述
25	OTG_WKDPDMCHG_EN	DP/DM 变化唤醒使能 当 USB 进入低功耗模式（suspend），设置此位，或 VBUS 当前为高，会自动唤醒 USB，并在使能的情况下（otg_wakeup_int_enable 为 1）产生唤醒中断。 此位仅用于设备模式。 主机模式时，由 PROTSC1 中的 WKDNS/WKCN 使能 DP，DM 唤醒

位域	名称	描述
19	AUTORESUME_EN	自动 resume 信号使能 1：使能 0：禁止 仅用于主机模式
17	OTG_VBUS_WAKEUP_EN	VBUS 变化唤醒使能 1：使能 0：禁止
16	OTG_ID_WAKEUP_EN	ID 变化唤醒使能 1：使能 0：禁止
13	OTG_VBUS_SOURCE_SEL	VBUS 唤醒来源 0：使用 vbus valid 作为唤醒源 1：使用 session valid 作为唤醒源 这两个信号区别在于阈值不同，vbus valid 会比较高，在某些应用中，用户可能 vbus 不足 5V，这种情况可以使用此位选择 session valid
12	OTG_UTMI_SUSPENDM_SW	PHY suspendm 软件控制 默认为 0，PHY 处于低功耗模式 软件需要在初始化时置位此位。 注意：在 suspendm 置 1 和 reset 清零之间，需要保证至少 1 微秒时间间隔
11	OTG_UTMI_RESET_SW	PHY 复位软件控制 默认为 1，PHY 处于复位状态。 软件需要在初始化时清零此位。 注意：在置位 suspendm 后至少等待 1 微秒才能清零此位
10	OTG_WAKEUP_INT_ENABLE	唤醒中断使能 当 usb 处于低功耗模式（PORTSC1.PHCD 为 1）时，系统可以关闭所有 PLL 和外部晶振，保留内部 osc 作为唤醒时钟。 当 usb 收到唤醒事件且相应事件使能时，如果此位置位，会产生唤醒中断。 唤醒中断状态位在 TOP_STATUS.wakeup_int_status，软件需要清零此位清除唤醒中断。 建议流程是，在进入低功耗模式后置位此位，收到唤醒中断后清零此位。
9	OTG_POWER_MASK	VBUS 控制极性 主机模式时，可从引脚输出 USBx_PWR（来源时 PORTSC1.PP)，用于控制外部 VBUS 开关， 此位可用于控制外部 VBUS 电路的极性，默认为 0，低有效。 如果外部 VBUS 控制开关时高有效，可以置位此位。 用户也可不适应 USBx_PWR，而使用其他 gpio 作为 VBUS 控制开关。

位域	名称	描述
8	OTG_OVER_CUR_POL	过流保护极性 主机模式时，可以从引脚 USBx_OC 引入过流状态（一般与来自于 VBUS 控制电路），此位定义过流状态极性。 0：高表示有过流发生 1：低表示有过流发生
7	OTG_OVER_CUR_DIS	过流禁止 主机模式时，用户可能在电路上没有过流状态指示，可以设置此位避免因为过流保护极性设置错误导致的错误过流状态。 0：过流状态正常工作 1：过流状态禁止，永远不会产生过流
4	SER_MODE_SUSPEND_EN	此位需置 1

OTG_CTRL0 位域

52.6.28 PHY_CTRL0 (0x210)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD						GPIO_ID_SEL_N	RSVD										ID_DIG_OVERRIDE	SESS_VALID_OVERRIDE	VBUS_VALID_OVERRIDE	RSVD									ID_DIG_OVERRIDE_EN	SESS_VALID_OVERRIDE_EN	VBUS_VALID_OVERRIDE_EN
N/A						RW	N/A										RW	RW	RW	N/A									RW	RW	RW
x	x	x	x	x	x	0	x	x	x	x	x	x	x	x	x	x	0	0	0	x	x	x	x	x	x	x	x	x	0	0	0

PHY_CTRL0 [31:0]

位域	名称	描述
25	GPIO_ID_SEL_N	ID 选择 PHY 有 USB ID 引脚，在某些封装，为了节约引脚，可能会不把 PHY 引脚接到芯片外部，此时可以使用某个 GPIO 作为 USBID（具体参见 PINMUX） 0：使用 PHY ID 1：使用 gpio ID
14	ID_DIG_OVERRIDE	ID 信号强制改写 当 id_dig_override_en 为 1 时，可以通过此位强制输入 ID 值。适用于因为外部电路原因，无法通过引脚得知当前 ID 状态，软件可以通过把通过其他渠道得到的 ID 状态写入此位

位域	名称	描述
13	SESS_VALID_OVERRIDE	sess valid 信号强制改写 适用于因为外部电路原因，无法通过 PHY 得知当前 vbus 状态，软件可以通过把通过其他渠道得到的 phy 状态写入此位，设备模式时，必须 vbus 有效（vbus valid 或者 sess valid 为高），设备控制器才能正常工作
12	VBUS_VALID_OVERRIDE	vbus valid 信号强制改写
2	ID_DIG_OVERRIDE_EN	ID 信号强制改写使能
1	SESS_VALID_OVERRIDE_EN	sess valid 信号强制改写使能
0	VBUS_VALID_OVERRIDE_EN	vbus valid 信号强制改写使能

PHY_CTRL0 位域

52.6.29 PHY_CTRL1 (0x214)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD											UTMI_CFG_RST_N	RSVD																		UTMI_OTG_SUSPENDM	RSVD
N/A											RW	N/A																		RW	N/A
x	x	x	x	x	x	x	x	x	x	x	0	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	x

PHY_CTRL1 [31:0]

位域	名称	描述
20	UTMI_CFG_RST_N	PHY 配置电路复位 低有效，默认为 0，PHY 配置电路处于复位状态，软件在初始化 PHY 时需要设置此位
1	UTMI_OTG_SUSPENDM	

PHY_CTRL1 位域

52.6.30 TOP_STATUS (0x220)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WAKEUP_INT_STATUS	RSVD																														
RW	N/A																														
0	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x

TOP_STATUS [31:0]

位域	名称	描述
31	WAKEUP_INT_STATUS	唤醒中断状态位 0：无唤醒中断 1：有唤醒中断 注意：此位不是写 1 清零，软件需要清零唤醒中断使能清零此位。

TOP_STATUS 位域

52.6.31 PHY_STATUS (0x224)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
UTMI_CLK_VALID	RSVD																							LINE_STATE		HOST_DISCONNECT	ID_DIG	RSVD	UTMI_SESS_VALID	RSVD	VBUS_VALID
RW	N/A																							RW		RW	RW	N/A	RW	N/A	RW
0	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	x	0	x	0

PHY_STATUS [31:0]

位域	名称	描述
31	UTMI_CLK_VALID	PHY 时钟有效状态位 此位用于软件判断当前 PHY 是否输出时钟。 使用流程如下： 写 1 清零此位； 读此位，为 0 表示 PHY 无时钟输出，1 表示有时钟输出。
7-6	LINE_STATE	USB 总线状态 低速全速时，bit7 为 DM 状态；bit6 为 DP 状态 高速时 PHY 仅输出是否有数据变化，00 表示无数据；01 表示有数据

位域	名称	描述
5	HOST_DISCONNECT	断开连接指示 仅用于主机模式，有高速设备连接时。 当高速设备被断开连接，主机无法像低速全速那样，通过 USB 中线状态变化检测到，只能通过在 SOF 帧末尾检测电平来判断，此位为 1 表示检测到高速设备被断开连接
4	ID_DIG	PHY ID 状态
2	UTMI_SESS_VALID	session valid 状态
0	VBUS_VALID	vbus valiud 状态

PHY_STATUS 位域

53 模拟外设概述

本章节介绍了本产品的模拟外设。

53.1 16 位模拟数字转换器 ADC16

本产品支持 3 个 16 位模拟数字转换器 ADC16。ADC16 是一种采用逐次逼近方式的模拟数字转换器，可以转换来自外部引脚、以及芯片内部的模拟信号，ADC16 的转换精度设置为 16 位时，最大采样率 2MSPS，当转换精度设置为 12 位时，最大采样率 4MSPS。

本产品上，二个 16 位 ADC 称为 ADC0~1。

本产品上，二个 16 位 ADC 的参考电压输入是 VREFH 和 VREFL。

注意：本产品上，ADC16 相关寄存器 (ADC16_ 开头的寄存器，偏移量大于等于 0x1400)，需要配置如下寄存器才能正常写入：

- 设置 CONV_CFG1 寄存器 CLK_DIVIDER=1；
- 设置 ANA_CTRL0 寄存器 ADC_CLK_ON=1;

53.1.1 ADC0 输入通道分配

16 位模拟数字转换器 ADC0 支持输入通道 ina0~15，全部来自 IO.

ADC0 输入通道与 IO 的具体连接，请参考章 20。

53.1.2 ADC1 输入通道分配

16 位模拟数字转换器 ADC1 支持输入通道 ina0~15，全部来自 IO。

ADC1 输入通道与 IO 的具体连接，请参考章 20。

53.1.3 ADC 转换触发信号连接

本产品上 16 位模拟数字转换器 ADC16 支持 2 种队列转换，序列转换和抢占转换：

- 序列转换长度最长可达 16，支持软件或者硬件触发，硬件触发信号 STRIGI，来自互联管理器 TRGM。片上的其他模块可以以此触发 ADC 序列转换
- 抢占转换长度最长可达 4，每个 ADC 支持 4 组，每组 3 个，共 12 个抢占转换队列。每个抢占队列支持软件或者硬件触发，12 个硬件触发信号来自互联管理器，片上的其他模块可以以此触发 ADC 抢占转换
  - PTRGI0A，PTRGI0B，PTRGI0C 来自互联管理器 TRGM
  - PTRGI1A，PTRGI1B，PTRGI1C 来自互联管理器 TRGM
  - PTRGI2A，PTRGI2B，PTRGI2C 来自互联管理器 TRGM
  - PTRGI3A，PTRGI3B，PTRGI3C 来自互联管理器 TRGM
  - 片上各个 ADC 的相同序号抢占转换触发序号短接，因此，同一个触发信号可以同时触发片上所有 ADC 的相同序号的抢占转换序列。

ADC16 的触发信号的具体连接请查阅节 32.8。

注意：在接收触发信号之前，ADC 必须先初始化，不然可能导致工作异常。

53.2 比较器 ACMP

本产品支持 2 个模拟比较器 ACMP。ACMP 可以对两个模拟电压输入 (正端电压 INP) 和负端电压 (INN) 进行比较，并输出比较结果。ACMP 支持内部 8 位数字模拟转换器 DAC，支持比较 2 个外部模拟信号，或者比较外部模拟信号与内部 DAC 生成的参考信号。

本产品上，ACMP 内部 DAC 的参考电压来自 VREFH。

本产品上，ACMP 支持 8 路正端电压 INP 输入通道和 8 路负端电压 INN 输入通道。INP0 和 INN0 来自 ACMP 内置 DAC 的输出。其余输入通道来自 IO。本产品上，各个 ACMP 输入通道与 IO 的具体连接，请参考章 20。

本产品上，2 个 ACMP 的输出连接到 IO，也连接到电机系统的互联管理器。具体连接请参考**??**和小节 32.8.1。

53.3 温度传感器

本产品上，集成了一个温度传感器，可以用于测量芯片内部的温度。

53.4 数字模拟转换器 DAC

本产品上，集成了 2 个 12 位数字模拟转换器 DAC。可以将数字信号，转换为模拟信号输出。

53.5 运算放大器 OPAMP

本产品上，集成了 2 个运算放大器 OPAMP。

54 16 位模数转换器 ADC16

本章节介绍 16 位模数转换器 ADC16 的主要功能和特性:

54.1 特性总结

本章节介绍 16 位模数转换器 ADC16 的主要特性：

- 16 位逐次逼近型 ADC
- 最大 2MHz 采样率
- 支持单端输入
- 独立的 ADC 转换时钟
- 支持任意配置的 AD 转换分辨率
- 可配置采样周期数
- 内置 DMA 可直接把 ADC 转换结果写入内存
- 支持读取转换模式
  - 读取结果寄存器直接触发转换
- 支持周期转换模式
  - 内置定时器按周期进行转换
  - 支持硬件阈值比较，对超出范围的转换结果报警
- 支持序列转换模式
  - 可由软件或硬件触发
  - 序列最长可达 16
  - 支持单次转换或者连续转换
  - 支持序列循环
- 支持抢占转换模式
  - 可由软件或硬件触发
  - 连续转换序列最长可达 4
- 支持生成各类中断

54.2 功能描述

本章节描述 16 位模数转换器 ADC16 的功能。

54.2.1 ADC 时钟

ADC 时钟包括 ADC 控制器时钟和 ADC 转换时钟。

其中 ADC 转换时钟是 ADC 逐次逼近模拟转换节拍控制的时钟。

由 ADC 控制器时钟经过预分频得到，ADC 控制器时钟由系统控制模块配置，默认为系统总线时钟 200MHz(与电机系统时钟同步)。

用户可以通过设置 CONVCFG1[CONVCLKDIV] 寄存器位，配置 ADC 转换时钟相对于 ADC 控制器时钟的预分频的值。CONVCFG1[CONVCLKDIV] 值为 0 时，代表预分频值为/1；值为 1 时，预分频值为/2，最大为/16。

ADC16 转换时钟最大频率为 50MHz。

54.2.2 ADC 输入通道配置

ADC 的每一个输入通道都可以通过 SAMPLE_CFGx 寄存器来单独配置。

用户可以通过 SAMPLE_CFGx [SAMPLE_CLOCK_NUMBER] 和 [SAMPLE_CLOCK_NUMBER_SHIFT] 位来设置通道的采样时间。

采样时间为 $SAMPLE\_CLOCK\_NUMBER \times 2^{SAMPLE\_CLOCK\_NUMBER\_SHIFT}$ 个时钟周期。采样时间最短为 1 个时钟周期。

此外，用户需要把 CONV_CFG1[CONVERT_CLOCK_NUMBER] 配置成 21，即转换时间为 21 个时钟周期，来达到 16 位转换精度。

如用户不需要 16 位精度，但希望更快得到结果，可以通过缩短转换时间来完成。

用户需同时改变 CONV_CFG1[CONVERT_CLOCK_NUMBER] 和 ADC16_CONFIG1[COV_END_CNT]。

COV_END_CNT = 21-CONVERT_CLOCK_NUMBER; CONVERT_CLOCK_NUMBER=21 时 ADC16 工作在完全的 16 位模式；

CONVERT_CLOCK_NUMBER=14 时 ADC16 大约工作在 12 位模式；

CONVERT_CLOCK_NUMBER=11 时 ADC16 大约工作在 10 位模式；

CONVERT_CLOCK_NUMBER= 9 时 ADC16 大约工作在 8 位模式；

其他数值也可工作，用户可根据精度和速度需求自由选择。

每个通道的转换时间是采样时间和转换时间的和。

54.2.3 读取转换模式

用户可以通过读取转换模式直观的读取 ADC 某个输入通道的转换结果。

适用于用户需要尽快知道某个通道的结果，在此之前无其他任务可做的情况。

当用户读取 BUS_RESULTx 寄存器时，就会触发一次对 ADC 输入 x 的转换。待转换完成以后，直接返回结果。

根据不同的 ADC 通道配置，返回转换结果的用时也会不同。

在 ADC 完成转换之前，ADC 会阻塞相关外设总线的访问，而由转换引起的阻塞时间可能会比较长。用户可以把 BUF_CFG0 [WAITDIS] 位置 1 来关闭总线阻塞，这时读 BUS_RESULTx 寄存器会直接返回上一次转换的结果。当 BUS_RESULTx [VALID] 位置 1 时，提示 ADC 转换完成，此时 BUS_RESULTx 寄存器中保存了最近一次的转换结果。

54.2.4 周期转换模式

用户可以通过周期转换模式要求 ADC 的一个或者多个输入通道进行周期性的转换。

适用于有定时转换的需求，或监控某个通道是否超过阈值。

用户可以通过 PRD_CFGx [PRD_CNT] 和 [PRESCALE] 寄存器位设置周期的长度，周期的长度具体为 $PRD\_CNT \times 2^{PRESCALE}$，每个 ADC 时钟周期，计数器都会-1，计数器计数到 0 时，即开始转换，同时计数器重载。

注意，把 PRD_CFGx [PRD_CNT] 位清 0 的话，表示关闭该输入的周期转换。

周期转换的结果保存在 PRD_RESULTx 寄存器里，软件能从 PRD_RESULTx 里读取到输入通道 x 的最近一次转换结果。

注意，PRD_RESULTx 总是保存 ADC 输入通道 x 最近一次的转换结果。PRD_RESULTx 不仅保存周期转换的结果，也保存其他转换模式的转换结果。

用户可以通过设置 PRD_THSHD_CFGx[THSHDH] 和 PRD_THSHD_CFGx[THSHDL] 对输入通道 x 的转换结果进行监测。一旦转换结果超出范围 (Result 大于 THSHDHx 或者 Result 小于 THSHDLx)，即会报警，相应的标志位 INT_STS[ADWDGx] 位会置 1，如果相应中断使能，还会产生中断。

注意，ADC 转换结果检测不限于周期转换模式，而是对所有读取模式都有效。如果设置的门限不正确，比如，把 THSHDHx 设为 0，那么每当通道转换结束，都会触发报警。

如果有多个通道同时计时到期需要开始转换，那么序号较小的输入通道会先转换。

54.2.5 序列转换模式

ADC 支持序列转换模式，在此模式下，ADC 可以按照预先编程好的顺序，对指定的输入通道逐一转换。

适用于对某一或者某几个通道，进行连续转换，数据量比较大的情况，比如测试信噪比或 ENOB。

用户可以通过依次配置 SEQ_QUEx 寄存器来指定序列的转换目标是哪个输入通道。SEQ_QUE0 [CHAN_NUM] 位域可以配置序列转换触发之后，第一次 AD 转换的通道序号；SEQ_QUE1 [CHNUM] 位域配置第二次 AD 转换的通道序号，以此类推，SEQ_QUE15[CHNUM] 位配置第 16 次 AD 转换的通道序号。

此外，用户需要配置 SEQ_CFG0 [SEQ_LEN] 位，来指定转换序列的长度，ADC 的转换序列最长可达 16。

序列开始转换可以通过软件或者硬件触发:

- 用户通过软件，先把 SEQ_CFG0 [SW_TRIG_EN] 位置 1，再把 SEQ_CFG0 [SW_TRIG] 位置 1，即触发序列转换。
- SEQ_CFG0 [HW_TRIG_EN] 位置 1 以后，在 STRGI 上捕获到上升沿，即触发序列转换

一经触发，ADC 将按照顺序，根据 SEQ_QUE0 的配置开始转换输入通道，转换完成后，标志位 INT_STS [SEQ_CVC]（转换完成，conversion complete）就会置 1。此时:

- 如果 SEQ_CFG0 [CONT_EN] 位置 1，ADC 会自动根据 SEQ_QUE1 的配置开始下一次 AD 转换，直到达到指定的序列长度 SEQ_QUEn。
- 如果 SEQ_CFG0 [CONT_EN] 位清 0，ADC 等到下一次软件或者硬件触发，才会开始下一个通道的转换。

当 ADC 完成序列的全部转换（长度由 SEQ_CFG0 [SEQ_LEN] 位设置）后，标志位 INT_STS[SEQ_CMPT]（序列转换完成，Sequence Complete）位会置 1。此时:

- 如果 SEQ_CFG0 [RESTART_EN] 位置 1,SEQ_CFG0 [CONT_EN] 位置 1,ADC 会自动根据 SEQ_QUEx 的配置，从 SEQ_QUE0 开始连续依次转换
- 如果 SEQ_CFG0 [RESTART_EN] 位置 0，SEQ_CFG0 [CONT_EN] 位置 1，ADC 在下一次软件或者硬件触发后，会连续转换直到整个序列完成
- 如果 SEQ_CFG0 [CONT_EN] 位置 0，无论 SEQ_CFG0 [RESTART_EN] 位置 1 还是置 0，ADC 在下一次软件或者硬件触发后，才会重新根据 SEQ_QUEx 的配置开始转换。

如果在 ADC 序列转换过程中，接收到新的序列转换触发信号，ADC 会忽略这个触发。如果是软件触发序列转换，此时，INT_STS [SEQ_SW_CFLCT] 位会置 1。如果是由 STRGI 引发的硬件触发序列转换，INT_STS [SEQ_HW_CFLCT] 位会置 1。

54.2.6 序列转换模式的 DMA

ADC 序列转换模式支持内置 DMA，可以直接把转换结果写入内存中用户指定的缓冲区。用户可以通过寄存器 SEQ_DMA_ADDR 配置目标地址，通过 SEQ_DMA_CFG [BUF_LEN] 设置数据区域长度。这样为 ADC 的序列转换结果配置了一个循环缓冲区。即：

第一次 AD 转换完成，DMA 会把转换结果写入 SEQ_DMA_ADDR，下一次 AD 转换的结果会写入 SEQ_DMA_ADDR + 4。之后每一次转换完成都会使指针地址持续累加，如果地址超出 SEQ_DMA_ADDR + BUF_LEN，那么 ADC 会重新向 SEQ_DMA_ADDR 指向的内存地址单元写入下一次 AD 转换的结果。

注意，DMA 每次会向内存写入 32 位（4 字节）数据，其中 16 位的 AD 转换结果位于 [15:0]。位 [31:16] 还包括了周期，序列编号，AD 输入通道序号等标志数据，方便软件解读 AD 转换的信息。

图 79为序列转换模式，DMA 写入内存的数据格式。

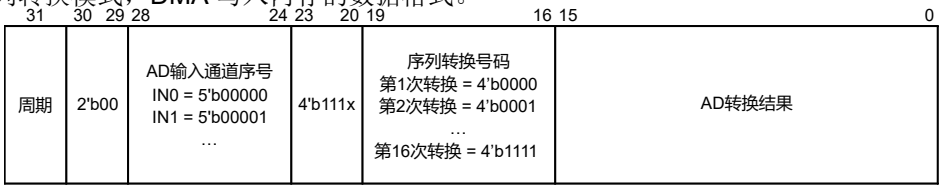


图 79: 序列转换模式数据格式

用户可以利用其中的周期（Cycle）位 [31] 来辅助判断 AD 转换结果的实时性。当 DMA 在内存的缓冲区中第一次写入转换结果时，该位会置 1。当写满整个缓冲区后，DMA 指针循环指向起始位置，这一轮该位会置 0。当 DMA 又一次写满缓冲区，指针第三次回到起始位置，这一轮周期（Cycle）位又会置 1。

为防止 DMA 覆盖掉缓冲区中没有被及时读取的数据，用户可以把 SEQ_DMA_CFG [STOP_EN] 位置 1，并设置 SEQ_DMA_CFG [STOP_POS] 位，以此标记最近一次软件读取转换结果的位置。DMA 在指针到达 SEQ_DMA_ADDR + STOP_POS 后，就不再将数据写入缓冲区。这次及之后的 AD 转换的结果会丢失，标志位 INT_STS [SEQ_DMAABT] 会置 1，如果打开对应中断，那么 ADC 会生成中断报警。

用户可以通过对 SEQ_DMA_CFG [DMA_RST] 位置 1 再清 0，复位整个转换序列。复位之后，ADC 在下一次软件或者硬件触发后，重新根据 SEQ_QUE0 的配置，开始序列转换。该位也会重置 DMA 的指针到 SEQ_DMA_ADDR 寄存器包含的地址。周期（Cycle）位也会重置到 1。

建议用户把 SEQ_DMA_CFG [DMA_RST] 位置 1 后，软件把内存中的缓冲区初始化清 0，之后再把 SEQ_DMA_CFG [DMA_RST] 位清 0。

54.2.7 抢占转换模式

ADC 支持抢占转换模式，抢占转换是优先级最高的转换模式。

适用于电机控制系统，需要在指定时间对指定通道进行转换的实时性要求比较高的情况。

ADC 支持 12 组抢占转换序列，触发信号分别来自于芯片的片上互联模块，共 12 个抢占触发源。

用户可以通过寄存器 CONFIGx，分别配置这些抢占转换序列。

抢占转换通过硬件触发：

- 在 PTRGIxA，PTRGIxB 或者 PTRGIxC 上捕获到上升沿，即触发抢占序列 xA，xB 或 xC

用户首先需要通过 CONFIGx 寄存器的 [TRIG_LEN] 位配置抢占转换的序列长度，序列长度最多为 4，即每个触发最多可以触发一次长度为 4 的转换。

用户可以通过 CONFIGx 寄存器配置抢占转换的转换顺序，配置寄存器的 [CHAN0]，[CHAN1]，[CHAN2]，

[CHAN3] 位域依次可以配置触发后的第 1，2，3，4 次转换的 AD 输入通道号码。

抢占转换一旦开始，就会根据 [TRIG_LEN] 的配置连续转换完成整个抢占序列。抢占序列的第 x 次转换完成后，如果 CONFIGx 寄存器的 [INTENx] 位置 1，INT_STS [TRIG_CMPT] 标志位会置 1，如果相应的中断控制位也置 1，ADC 就会生成中断。

如果在 ADC 抢占转换过程中，接收到新的抢占转换触发信号，ADC 不会响应新的触发。但是会根据触发来源是软件还是硬件，将 INT_STS [TRIG_SW_CFLICT] 标志位或者 INT_STS[TRIG_HW_CFLICT] 标志位置 1，表示发生了抢占转换触发冲突，如果对应的中断控制位置 1，ADC 会产生中断报警。

如果几个不同的抢占转换同时触发，ADC 会按以下次序优先响应：

组号较小的抢占转换，如 PTRGI0A 优先于 PTRGI1A，同组之间，xA 优先于 xB，xB 优先于 xC。如 PTRGI0A 优先于 PTRGI0B。同时 INT_STS[PTCHWCFLICT] 标志位置 1。

54.2.8 抢占转换模式的 DMA

ADC 抢占转换模式支持内置 DMA，可以直接把转换结果写入内存中用户指定的缓冲区。

用户可以通过寄存器 TRG_DMA_ADDR 配置目标地址，以此地址为基地址，ADC 会占用一块 192 字节的区域作为抢占转换结果的缓冲区。缓冲区分配如表 218：

序号	地址	描述
0	TRG_DMA_ADDR + 0x0	PTRGI0A 触发的第一次 AD 转换结果与附加信息
1	TRG_DMA_ADDR + 0x4	PTRGI0A 触发的第二次 AD 转换结果与附加信息
2	TRG_DMA_ADDR + 0x8	PTRGI0A 触发的第三次 AD 转换结果与附加信息
3	TRG_DMA_ADDR + 0xC	PTRGI0A 触发的第四次 AD 转换结果与附加信息
4	TRG_DMA_ADDR + 0x10	PTRGI0B 触发的第一次 AD 转换结果与附加信息
5	TRG_DMA_ADDR + 0x14	PTRGI0B 触发的第二次 AD 转换结果与附加信息
6	TRG_DMA_ADDR + 0x18	PTRGI0B 触发的第三次 AD 转换结果与附加信息
7	TRG_DMA_ADDR + 0x1C	PTRGI0B 触发的第四次 AD 转换结果与附加信息
8	TRG_DMA_ADDR + 0x20	PTRGI0C 触发的第一次 AD 转换结果与附加信息
9	TRG_DMA_ADDR + 0x24	PTRGI0C 触发的第二次 AD 转换结果与附加信息
10	TRG_DMA_ADDR + 0x28	PTRGI0C 触发的第三次 AD 转换结果与附加信息
11	TRG_DMA_ADDR + 0x2C	PTRGI0C 触发的第四次 AD 转换结果与附加信息
12	TRG_DMA_ADDR + 0x30	PTRGI1A 触发的第一次 AD 转换结果与附加信息
13	TRG_DMA_ADDR + 0x34	PTRGI1A 触发的第二次 AD 转换结果与附加信息
14	TRG_DMA_ADDR + 0x38	PTRGI1A 触发的第三次 AD 转换结果与附加信息
15	TRG_DMA_ADDR + 0x3C	PTRGI1A 触发的第四次 AD 转换结果与附加信息
16	TRG_DMA_ADDR + 0x40	PTRGI1B 触发的第一次 AD 转换结果与附加信息
17	TRG_DMA_ADDR + 0x44	PTRGI1B 触发的第二次 AD 转换结果与附加信息
18	TRG_DMA_ADDR + 0x48	PTRGI1B 触发的第三次 AD 转换结果与附加信息
19	TRG_DMA_ADDR + 0x4C	PTRGI1B 触发的第四次 AD 转换结果与附加信息
20	TRG_DMA_ADDR + 0x50	PTRGI1C 触发的第一次 AD 转换结果与附加信息
21	TRG_DMA_ADDR + 0x54	PTRGI1C 触发的第二次 AD 转换结果与附加信息
22	TRG_DMA_ADDR + 0x58	PTRGI1C 触发的第三次 AD 转换结果与附加信息

序号	地址	描述
23	TRG_DMA_ADDR + 0x5C	PTRGI1C 触发的第四次 AD 转换结果与附加信息
24	TRG_DMA_ADDR + 0x60	PTRGI2A 触发的第一次 AD 转换结果与附加信息
25	TRG_DMA_ADDR + 0x64	PTRGI2A 触发的第二次 AD 转换结果与附加信息
26	TRG_DMA_ADDR + 0x68	PTRGI2A 触发的第三次 AD 转换结果与附加信息
27	TRG_DMA_ADDR + 0x6C	PTRGI2A 触发的第四次 AD 转换结果与附加信息
28	TRG_DMA_ADDR + 0x70	PTRGI2B 触发的第一次 AD 转换结果与附加信息
29	TRG_DMA_ADDR + 0x74	PTRGI2B 触发的第二次 AD 转换结果与附加信息
30	TRG_DMA_ADDR + 0x78	PTRGI2B 触发的第三次 AD 转换结果与附加信息
31	TRG_DMA_ADDR + 0x7C	PTRGI2B 触发的第四次 AD 转换结果与附加信息
32	TRG_DMA_ADDR + 0x80	PTRGI2C 触发的第一次 AD 转换结果与附加信息
33	TRG_DMA_ADDR + 0x84	PTRGI2C 触发的第二次 AD 转换结果与附加信息
34	TRG_DMA_ADDR + 0x88	PTRGI2C 触发的第三次 AD 转换结果与附加信息
35	TRG_DMA_ADDR + 0x8C	PTRGI2C 触发的第四次 AD 转换结果与附加信息
36	TRG_DMA_ADDR + 0x90	PTRGI3A 触发的第一次 AD 转换结果与附加信息
37	TRG_DMA_ADDR + 0x94	PTRGI3A 触发的第二次 AD 转换结果与附加信息
38	TRG_DMA_ADDR + 0x98	PTRGI3A 触发的第三次 AD 转换结果与附加信息
39	TRG_DMA_ADDR + 0x9C	PTRGI3A 触发的第四次 AD 转换结果与附加信息
40	TRG_DMA_ADDR + 0xA0	PTRGI3B 触发的第一次 AD 转换结果与附加信息
41	TRG_DMA_ADDR + 0xA4	PTRGI3B 触发的第二次 AD 转换结果与附加信息
42	TRG_DMA_ADDR + 0xA8	PTRGI3B 触发的第三次 AD 转换结果与附加信息
43	TRG_DMA_ADDR + 0xAC	PTRGI3B 触发的第四次 AD 转换结果与附加信息
44	TRG_DMA_ADDR + 0xB0	PTRGI3C 触发的第一次 AD 转换结果与附加信息
45	TRG_DMA_ADDR + 0xB4	PTRGI3C 触发的第二次 AD 转换结果与附加信息
46	TRG_DMA_ADDR + 0xB8	PTRGI3C 触发的第三次 AD 转换结果与附加信息
47	TRG_DMA_ADDR + 0xBC	PTRGI3C 触发的第四次 AD 转换结果与附加信息

表 218: ADC 抢占转换 DMA 数据缓冲区分配

抢占转换触发后，按照触发来源以及在抢占转换序列中的序号，DMA 会把转换结果写入缓冲区对应的位置。

注意，DMA 每次会向内存写入 32 位（4 字节）数据，其中 16 位的 AD 转换结果位于 [15:0]。位 [31:16] 还包括了周期，序列编号，AD 输入通道序号等标志数据，方便软件解读 AD 转换的信息。

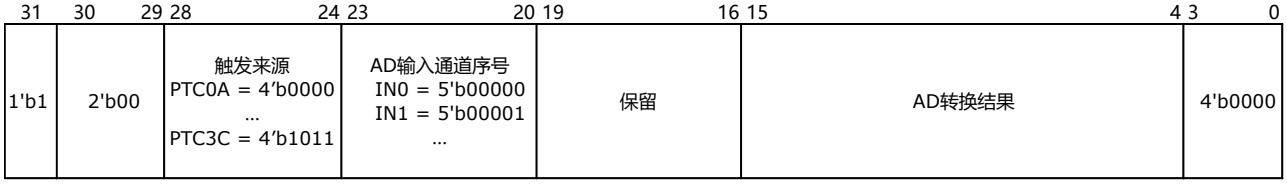


图 80: 抢占转换模式数据格式

位 31 用于和软件的交互，软件可以在初始化缓冲区时将此位清零，在收到转换完成中断后读取结果时，需检查此位为高，确保 ADC 已经将新的结果写入缓冲区。

54.2.9 ADC 中断

ADC 支持以下中断：

- 在序列转换模式下，序列中某一次 AD 转换完成，INT_STS [SEQ_CVC] 位置 1
- 在序列转换模式下，整个序列转换完成，INT_STS[SEQ_CMPT] 位置 1
- ADC 序列转换进行中，软件触发序列转换，INT_STS[SEQ_SW_CFLCT] 位置 1
- ADC 序列转换进行中，硬件触发序列转换，INT_STS[SEQ_HW_CFLCT] 位置 1
- 在序列转换模式下，SEQ_DMA_CFG[STOP_EN] 位置 1，并设置 SEQ_DMA_CFG [STOP_POS] 位，DMA 在指针到达 SEQ_DMA_ADDR + STOP_POS 后，停止将数据写入缓冲区，标志位 INT_STS[SEQ_DMAABT] 置 1
- 抢占转换模式下，抢占序列的第 x 次转换完成后，并且 CONFIGx 寄存器的 [INTENx] 位置 1，INT_STS[TRIG_CMPT] 位置 1
- ADC 序列抢占转换进行中，软件触发抢占转换，INT_STS[TRIG_SW_CFLICT] 位置 1
- ADC 序列抢占转换进行中，又收到抢占触发信号，或者多个抢占触发信号同时到达，INT_STS[TRIG_HW_CFLICT] 位置 1
- ADC 读取转换进行中，并且已将 BUF_CFG0 [WAITDIS] 位置 1，软件读取另一个 BUS_RESULT 寄存器，INT_STS [READ_CFLCT] 位置 1
- 当输入通道 x 的 AD 转换结果超出 THSHDx 寄存器设置的范围，即会报警，INT_STS [WDOG] 位置 1
- 序列转换和抢占转换的 DMA FIFO 溢出时，INT_STS[DMA_FIFO_FULL] 位置 1
- 序列转换和抢占转换的 DMA 对存储器写入 ADC 转换结果出错时，INT_STS [AHB_ERR] 位置 1

用户可以对 INT_STS 寄存器内的各个标志位写入 1 来把该标志位清 0。

ADC 可以通过 INT_EN 寄存器管理这些中断，如果对应的中断使能位置 1，那么标志位置 1 时，即表示产生了中断。

54.3 ADC16 寄存器列表

ADC16 的寄存器列表如下：

ADC0 base address: 0xF3080000

ADC1 base address: 0xF3084000

地址偏移	名称	描述	复位值
0x0000	CONFIG[TRG0A]	抢占模式配置寄存器	0x00000000
0x0004	CONFIG[TRG0B]	抢占模式配置寄存器	0x00000000
0x0008	CONFIG[TRG0C]	抢占模式配置寄存器	0x00000000
0x000C	CONFIG[TRG1A]	抢占模式配置寄存器	0x00000000
0x0010	CONFIG[TRG1B]	抢占模式配置寄存器	0x00000000
0x0014	CONFIG[TRG1C]	抢占模式配置寄存器	0x00000000
0x0018	CONFIG[TRG2A]	抢占模式配置寄存器	0x00000000
0x001C	CONFIG[TRG2B]	抢占模式配置寄存器	0x00000000
0x0020	CONFIG[TRG2C]	抢占模式配置寄存器	0x00000000
0x0024	CONFIG[TRG3A]	抢占模式配置寄存器	0x00000000
0x0028	CONFIG[TRG3B]	抢占模式配置寄存器	0x00000000
0x002C	CONFIG[TRG3C]	抢占模式配置寄存器	0x00000000

地址偏移	名称	描述	复位值
0x0030	TRG_DMA_ADDR	抢占模式 DMA 地址寄存器	0x00000000
0x0034	TRG_SW_STA		0x00000000
0x0400	BUS_RESULT[CHN0]	读取模式结果寄存器	0x00000000
0x0404	BUS_RESULT[CHN1]	读取模式结果寄存器	0x00000000
0x0408	BUS_RESULT[CHN2]	读取模式结果寄存器	0x00000000
0x040C	BUS_RESULT[CHN3]	读取模式结果寄存器	0x00000000
0x0410	BUS_RESULT[CHN4]	读取模式结果寄存器	0x00000000
0x0414	BUS_RESULT[CHN5]	读取模式结果寄存器	0x00000000
0x0418	BUS_RESULT[CHN6]	读取模式结果寄存器	0x00000000
0x041C	BUS_RESULT[CHN7]	读取模式结果寄存器	0x00000000
0x0420	BUS_RESULT[CHN8]	读取模式结果寄存器	0x00000000
0x0424	BUS_RESULT[CHN9]	读取模式结果寄存器	0x00000000
0x0428	BUS_RESULT[CHN10]	读取模式结果寄存器	0x00000000
0x042C	BUS_RESULT[CHN11]	读取模式结果寄存器	0x00000000
0x0430	BUS_RESULT[CHN12]	读取模式结果寄存器	0x00000000
0x0434	BUS_RESULT[CHN13]	读取模式结果寄存器	0x00000000
0x0438	BUS_RESULT[CHN14]	读取模式结果寄存器	0x00000000
0x043C	BUS_RESULT[CHN15]	读取模式结果寄存器	0x00000000
0x0500	BUF_CFG0	读取模式配置寄存器 0	0x00000000
0x0800	SEQ_CFG0	序列模式配置寄存器 0	0x00000000
0x0804	SEQ_DMA_ADDR	序列模式 DMA 地址寄存器	0x00000000
0x0808	SEQ_WR_ADDR	序列模式 DMA 指针寄存器	0x00000000
0x080C	SEQ_DMA_CFG	序列模式 DMA 配置寄存器	0x00000000
0x0810	SEQ_QUE[CFG0]	序列模式队列配置寄存器	0x00000000
0x0814	SEQ_QUE[CFG1]	序列模式队列配置寄存器	0x00000000
0x0818	SEQ_QUE[CFG2]	序列模式队列配置寄存器	0x00000000
0x081C	SEQ_QUE[CFG3]	序列模式队列配置寄存器	0x00000000
0x0820	SEQ_QUE[CFG4]	序列模式队列配置寄存器	0x00000000
0x0824	SEQ_QUE[CFG5]	序列模式队列配置寄存器	0x00000000
0x0828	SEQ_QUE[CFG6]	序列模式队列配置寄存器	0x00000000
0x082C	SEQ_QUE[CFG7]	序列模式队列配置寄存器	0x00000000
0x0830	SEQ_QUE[CFG8]	序列模式队列配置寄存器	0x00000000
0x0834	SEQ_QUE[CFG9]	序列模式队列配置寄存器	0x00000000
0x0838	SEQ_QUE[CFG10]	序列模式队列配置寄存器	0x00000000
0x083C	SEQ_QUE[CFG11]	序列模式队列配置寄存器	0x00000000
0x0840	SEQ_QUE[CFG12]	序列模式队列配置寄存器	0x00000000
0x0844	SEQ_QUE[CFG13]	序列模式队列配置寄存器	0x00000000
0x0848	SEQ_QUE[CFG14]	序列模式队列配置寄存器	0x00000000
0x084C	SEQ_QUE[CFG15]	序列模式队列配置寄存器	0x00000000
0x0850	SEQ_HIGH_CFG	序列模式扩展配置寄存器	0x00000000

地址偏移	名称	描述	复位值
0x0C00	PRD_CFG[CHN0][PRD_CFG]	周期模式配置寄存器	0x00000000
0x0C04	PRD_CFG[CHN0][PRD_THSHD_CFG]	转换结果监测配置寄存器	0x00000000
0x0C08	PRD_CFG[CHN0][PRD_RESULT]	周期模式结果寄存器	0x00000000
0x0C10	PRD_CFG[CHN1][PRD_CFG]	周期模式配置寄存器	0x00000000
0x0C14	PRD_CFG[CHN1][PRD_THSHD_CFG]	转换结果监测配置寄存器	0x00000000
0x0C18	PRD_CFG[CHN1][PRD_RESULT]	周期模式结果寄存器	0x00000000
0x0C20	PRD_CFG[CHN2][PRD_CFG]	周期模式配置寄存器	0x00000000
0x0C24	PRD_CFG[CHN2][PRD_THSHD_CFG]	转换结果监测配置寄存器	0x00000000
0x0C28	PRD_CFG[CHN2][PRD_RESULT]	周期模式结果寄存器	0x00000000
0x0C30	PRD_CFG[CHN3][PRD_CFG]	周期模式配置寄存器	0x00000000
0x0C34	PRD_CFG[CHN3][PRD_THSHD_CFG]	转换结果监测配置寄存器	0x00000000
0x0C38	PRD_CFG[CHN3][PRD_RESULT]	周期模式结果寄存器	0x00000000
0x0C40	PRD_CFG[CHN4][PRD_CFG]	周期模式配置寄存器	0x00000000
0x0C44	PRD_CFG[CHN4][PRD_THSHD_CFG]	转换结果监测配置寄存器	0x00000000
0x0C48	PRD_CFG[CHN4][PRD_RESULT]	周期模式结果寄存器	0x00000000
0x0C50	PRD_CFG[CHN5][PRD_CFG]	周期模式配置寄存器	0x00000000
0x0C54	PRD_CFG[CHN5][PRD_THSHD_CFG]	转换结果监测配置寄存器	0x00000000
0x0C58	PRD_CFG[CHN5][PRD_RESULT]	周期模式结果寄存器	0x00000000
0x0C60	PRD_CFG[CHN6][PRD_CFG]	周期模式配置寄存器	0x00000000
0x0C64	PRD_CFG[CHN6][PRD_THSHD_CFG]	转换结果监测配置寄存器	0x00000000
0x0C68	PRD_CFG[CHN6][PRD_RESULT]	周期模式结果寄存器	0x00000000
0x0C70	PRD_CFG[CHN7][PRD_CFG]	周期模式配置寄存器	0x00000000
0x0C74	PRD_CFG[CHN7][PRD_THSHD_CFG]	转换结果监测配置寄存器	0x00000000
0x0C78	PRD_CFG[CHN7][PRD_RESULT]	周期模式结果寄存器	0x00000000
0x0C80	PRD_CFG[CHN8][PRD_CFG]	周期模式配置寄存器	0x00000000
0x0C84	PRD_CFG[CHN8][PRD_THSHD_CFG]	转换结果监测配置寄存器	0x00000000
0x0C88	PRD_CFG[CHN8][PRD_RESULT]	周期模式结果寄存器	0x00000000
0x0C90	PRD_CFG[CHN9][PRD_CFG]	周期模式配置寄存器	0x00000000
0x0C94	PRD_CFG[CHN9][PRD_THSHD_CFG]	转换结果监测配置寄存器	0x00000000
0x0C98	PRD_CFG[CHN9][PRD_RESULT]	周期模式结果寄存器	0x00000000
0x0CA0	PRD_CFG[CHN10][PRD_CFG]	周期模式配置寄存器	0x00000000
0x0CA4	PRD_CFG[CHN10][PRD_THSHD_CFG]	转换结果监测配置寄存器	0x00000000
0x0CA8	PRD_CFG[CHN10][PRD_RESULT]	周期模式结果寄存器	0x00000000
0x0CB0	PRD_CFG[CHN11][PRD_CFG]	周期模式配置寄存器	0x00000000
0x0CB4	PRD_CFG[CHN11][PRD_THSHD_CFG]	转换结果监测配置寄存器	0x00000000
0x0CB8	PRD_CFG[CHN11][PRD_RESULT]	周期模式结果寄存器	0x00000000
0x0CC0	PRD_CFG[CHN12][PRD_CFG]	周期模式配置寄存器	0x00000000
0x0CC4	PRD_CFG[CHN12][PRD_THSHD_CFG]	转换结果监测配置寄存器	0x00000000
0x0CC8	PRD_CFG[CHN12][PRD_RESULT]	周期模式结果寄存器	0x00000000
0x0CD0	PRD_CFG[CHN13][PRD_CFG]	周期模式配置寄存器	0x00000000

地址偏移	名称	描述	复位值
0x0CD4	PRD_CFG[CHN13][PRD_THSHD_CFG]	转换结果监测配置寄存器	0x00000000
0x0CD8	PRD_CFG[CHN13][PRD_RESULT]	周期模式结果寄存器	0x00000000
0x0CE0	PRD_CFG[CHN14][PRD_CFG]	周期模式配置寄存器	0x00000000
0x0CE4	PRD_CFG[CHN14][PRD_THSHD_CFG]	转换结果监测配置寄存器	0x00000000
0x0CE8	PRD_CFG[CHN14][PRD_RESULT]	周期模式结果寄存器	0x00000000
0x0CF0	PRD_CFG[CHN15][PRD_CFG]	周期模式配置寄存器	0x00000000
0x0CF4	PRD_CFG[CHN15][PRD_THSHD_CFG]	转换结果监测配置寄存器	0x00000000
0x0CF8	PRD_CFG[CHN15][PRD_RESULT]	周期模式结果寄存器	0x00000000
0x1000	SAMPLE_CFG[CHN0]	通道采样配置寄存器	0x00000000
0x1004	SAMPLE_CFG[CHN1]	通道采样配置寄存器	0x00000000
0x1008	SAMPLE_CFG[CHN2]	通道采样配置寄存器	0x00000000
0x100C	SAMPLE_CFG[CHN3]	通道采样配置寄存器	0x00000000
0x1010	SAMPLE_CFG[CHN4]	通道采样配置寄存器	0x00000000
0x1014	SAMPLE_CFG[CHN5]	通道采样配置寄存器	0x00000000
0x1018	SAMPLE_CFG[CHN6]	通道采样配置寄存器	0x00000000
0x101C	SAMPLE_CFG[CHN7]	通道采样配置寄存器	0x00000000
0x1020	SAMPLE_CFG[CHN8]	通道采样配置寄存器	0x00000000
0x1024	SAMPLE_CFG[CHN9]	通道采样配置寄存器	0x00000000
0x1028	SAMPLE_CFG[CHN10]	通道采样配置寄存器	0x00000000
0x102C	SAMPLE_CFG[CHN11]	通道采样配置寄存器	0x00000000
0x1030	SAMPLE_CFG[CHN12]	通道采样配置寄存器	0x00000000
0x1034	SAMPLE_CFG[CHN13]	通道采样配置寄存器	0x00000000
0x1038	SAMPLE_CFG[CHN14]	通道采样配置寄存器	0x00000000
0x103C	SAMPLE_CFG[CHN15]	通道采样配置寄存器	0x00000000
0x1104	CONV_CFG1	转换配置寄存器 1	0x00000000
0x1108	ADC_CFG0	ADC 配置寄存器 0	0x00000000
0x1110	INT_STS	状态寄存器	0x00000000
0x1114	INT_EN	中断使能寄存器	0x00000000
0x1200	ANA_CTRL0	模拟控制寄存器 0	0x00000000
0x1210	ANA_STATUS	模拟状态寄存器	0x00000000
0x1400	ADC16_PARAMS[ADC16_PARA00]	ADC16 参数寄存器	0x0000
0x1402	ADC16_PARAMS[ADC16_PARA01]	ADC16 参数寄存器	0x0000
0x1404	ADC16_PARAMS[ADC16_PARA02]	ADC16 参数寄存器	0x0000
0x1406	ADC16_PARAMS[ADC16_PARA03]	ADC16 参数寄存器	0x0000
0x1408	ADC16_PARAMS[ADC16_PARA04]	ADC16 参数寄存器	0x0000
0x140A	ADC16_PARAMS[ADC16_PARA05]	ADC16 参数寄存器	0x0000
0x140C	ADC16_PARAMS[ADC16_PARA06]	ADC16 参数寄存器	0x0000
0x140E	ADC16_PARAMS[ADC16_PARA07]	ADC16 参数寄存器	0x0000
0x1410	ADC16_PARAMS[ADC16_PARA08]	ADC16 参数寄存器	0x0000
0x1412	ADC16_PARAMS[ADC16_PARA09]	ADC16 参数寄存器	0x0000

地址偏移	名称	描述	复位值
0x1414	ADC16_PARAMS[ADC16_PARA10]	ADC16 参数寄存器	0x0000
0x1416	ADC16_PARAMS[ADC16_PARA11]	ADC16 参数寄存器	0x0000
0x1418	ADC16_PARAMS[ADC16_PARA12]	ADC16 参数寄存器	0x0000
0x141A	ADC16_PARAMS[ADC16_PARA13]	ADC16 参数寄存器	0x0000
0x141C	ADC16_PARAMS[ADC16_PARA14]	ADC16 参数寄存器	0x0000
0x141E	ADC16_PARAMS[ADC16_PARA15]	ADC16 参数寄存器	0x0000
0x1420	ADC16_PARAMS[ADC16_PARA16]	ADC16 参数寄存器	0x0000
0x1422	ADC16_PARAMS[ADC16_PARA17]	ADC16 参数寄存器	0x0000
0x1424	ADC16_PARAMS[ADC16_PARA18]	ADC16 参数寄存器	0x0000
0x1426	ADC16_PARAMS[ADC16_PARA19]	ADC16 参数寄存器	0x0000
0x1428	ADC16_PARAMS[ADC16_PARA20]	ADC16 参数寄存器	0x0000
0x142A	ADC16_PARAMS[ADC16_PARA21]	ADC16 参数寄存器	0x0000
0x142C	ADC16_PARAMS[ADC16_PARA22]	ADC16 参数寄存器	0x0000
0x142E	ADC16_PARAMS[ADC16_PARA23]	ADC16 参数寄存器	0x0000
0x1430	ADC16_PARAMS[ADC16_PARA24]	ADC16 参数寄存器	0x0000
0x1432	ADC16_PARAMS[ADC16_PARA25]	ADC16 参数寄存器	0x0000
0x1434	ADC16_PARAMS[ADC16_PARA26]	ADC16 参数寄存器	0x0000
0x1436	ADC16_PARAMS[ADC16_PARA27]	ADC16 参数寄存器	0x0000
0x1438	ADC16_PARAMS[ADC16_PARA28]	ADC16 参数寄存器	0x0000
0x143A	ADC16_PARAMS[ADC16_PARA29]	ADC16 参数寄存器	0x0000
0x143C	ADC16_PARAMS[ADC16_PARA30]	ADC16 参数寄存器	0x0000
0x143E	ADC16_PARAMS[ADC16_PARA31]	ADC16 参数寄存器	0x0000
0x1440	ADC16_PARAMS[ADC16_PARA32]	ADC16 参数寄存器	0x0000
0x1442	ADC16_PARAMS[ADC16_PARA33]	ADC16 参数寄存器	0x0000
0x1444	ADC16_CONFIG0	ADC16 配置寄存器 0	0x00000000
0x1460	ADC16_CONFIG1		0x00000000

表 219: ADC16 寄存器列表

54.4 ADC16 寄存器描述

ADC16 的寄存器详细说明如下：

54.4.1 CONFIG (0x0 + 0x4 * n)

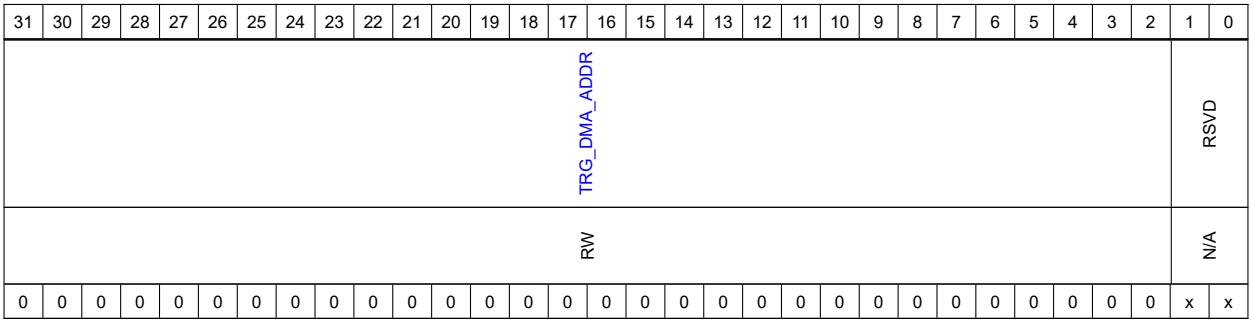
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TRIG_LEN		INTEN3	CHAN3					RSVD		INTEN2	CHAN2					RSVD		INTEN1	CHAN1					RSVD	QUEUE_EN	INTEN0	CHAN0				
WO		RW	RW					N/A		RW	RW					N/A		RW	RW					N/A	RW	RW	RW				
0	0	0	0	0	0	0	0	x	x	0	0	0	0	0	0	x	x	0	0	0	0	0	0	x	0	0	0	0	0	0	0

CONFIG [31:0]

位域	名称	描述
31-30	TRIG_LEN	抢占转换队列长度，0 表示队列包含 1 个转换，3 表示队列包含 4 个转换
29	INTEN3	第 4 次转换结束后中断使能，置 1 时，此次转换结束后 TRIG_COMPT 标志位置 1
28-24	CHAN3	第 4 次转换的通道号码
21	INTEN2	第 3 次转换结束后中断使能，置 1 时，此次转换结束后 TRIG_COMPT 标志位置 1
20-16	CHAN2	第 3 次转换的通道号码
13	INTEN1	第 2 次转换结束后中断使能，置 1 时，此次转换结束后 TRIG_COMPT 标志位置 1
12-8	CHAN1	第 2 次转换的通道号码
6	QUEUE_EN	抢占转换队列使能控制
5	INTEN0	第 1 次转换结束后中断使能，置 1 时，此次转换结束后 TRIG_COMPT 标志位置 1
4-0	CHAN0	第 1 次转换的通道号码

CONFIG 位域

54.4.2 TRG_DMA_ADDR (0x30)



TRG_DMA_ADDR [31:0]

位域	名称	描述
31-2	TRG_DMA_ADDR	抢占转换的 DMA 目标地址，抢占转换结束后，ADC 内置的 DMA 会将转换结果存放入该位域指向的一块长度为 192 字节的缓冲区。

TRG_DMA_ADDR 位域

54.4.3 TRG_SW_STA (0x34)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																											TRG_SW_STA	TRIG_SW_INDEX			
N/A																											RW	RW			
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0

TRG_SW_STA [31:0]

位域	名称	描述
4	TRG_SW_STA	
3-0	TRIG_SW_INDEX	

TRG_SW_STA 位域

54.4.4 BUS_RESULT (0x400 + 0x4 * n)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD															VALID	CHAN_RESULT															
N/A															RO	RO															
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

BUS_RESULT [31:0]

位域	名称	描述
16	VALID	转换结果有效提示位。如果 WAIT_DIS 位置 1，该位在读取转换结束后置 1，并在软件从结果寄存器读取转换结果后清 0 该位为 0 时，对读取转换结果寄存器的读操作会触发一次 AD 转换如果 AD 转换过程中读取其他通道的读取转换结果寄存器，不会触发 AD 转换，只会返回旧的转换结果，并且 read_cflict 标志位会置 1
15-0	CHAN_RESULT	读此位域会触发一次对应通道的 AD 转换 如果 WAIT_DIS 位置 1，读此位域会返回前一次 AD 转换结果，直到 VALID 位置 1 后，才能读到最近一次的转换结果如果 WAIT_DIS 位置 0，读此位域会阻塞总线，直到 AD 完成转换之后返回转换结果

BUS_RESULT 位域

54.4.5 BUF_CFG0 (0x500)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																														BUS_MODE_EN	WAIT_DIS
N/A																														RW	RW
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0

BUF_CFG0 [31:0]

位域	名称	描述
1	BUS_MODE_EN	读取模式使能
0	WAIT_DIS	该位置 1 可以防止读取模式阻塞总线

BUF_CFG0 位域

54.4.6 SEQ_CFG0 (0x800)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CYCLE	RSVD																			SEQ_LEN				RSVD			RESTART_EN	CONT_EN	SW_TRIG	SW_TRIG_EN	HW_TRIG_EN
RO	N/A																			RW				N/A			RW	RW	WO	RW	RW
0	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	x	x	x	0	0	0	0	0

SEQ_CFG0 [31:0]

位域	名称	描述
31	CYCLE	序列模式 DMA 写内存的周期提示位，该位在每次 DMA 写内存后会翻转
11-8	SEQ_LEN	序列转换的队列长度，0 表示包含 1 次转换，0xF 表示包含 16 次转换
4	RESTART_EN	自动重新转换位，当此位与 CONT_EN 位都置 1 时，仅需 1 次触发，ADC 硬件会持续地循环转换整个序列
3	CONT_EN	连续转换位 1：ADC 在收到 1 次序列转换触发后，就会连续转换完整个序列 0：ADC 在收到 1 次序列转换触发后，转换序列中的一个通道
2	SW_TRIG	序列转换软件触发位
1	SW_TRIG_EN	序列转换软件触发使能位
0	HW_TRIG_EN	序列转换硬件触发使能位

SEQ_CFG0 位域

54.4.7 SEQ_DMA_ADDR (0x804)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TAR_ADDR																														RSVD	
RW																														N/A	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	x	x

SEQ_DMA_ADDR [31:0]

位域	名称	描述
31-2	TAR_ADDR	序列转换 DMA 的目标地址

SEQ_DMA_ADDR 位域

54.4.8 SEQ_WR_ADDR (0x808)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD								SEQ_WR_POINTER																							
N/A								RO																							
x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

SEQ_WR_ADDR [31:0]

位域	名称	描述
23-0	SEQ_WR_POINTER	序列抓换 DMA 的写指针，每次 DMA 把转换结果写入内存后，硬件会自动更新该位 下一次写入的地址为 DDR + 4 * SEQ_WR_POINTER

SEQ_WR_ADDR 位域

54.4.9 SEQ_DMA_CFG (0x80C)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD				STOP_POS												RSVD		DMA_RST	STOP_EN	BUF_LEN											
N/A				RW												N/A		RW	RW	RW											
x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0

SEQ_DMA_CFG [31:0]

位域	名称	描述
27-16	STOP_POS	DMA 停止位置位，如果 STOP_EN 位置 1，序列模式的 DMA 在写入到该位指定的位置后，就不再继续将结果写入内存。直到软件更新此位，指定新的停止写入位置。
13	DMA_RST	重置序列转换 DMA，置 1 后，序列转换 DMA 不再将转换结果写入内存，会把写内存指针重置到 TAR_ADDR，并把 cycle 位重置为 1
12	STOP_EN	该位置 1 后，序列模式 DMA 会在指针指向特定位置后，不再往内存写入转换结果
11-0	BUF_LEN	内存中分配给序列转换 DMA 的缓冲区长度。DMA 会在写满缓冲区后，重新从缓冲区头部写入新的转换结果 0 表示缓冲区大小为 4 Byte，0xFFF 表示缓冲区大小为 16 KB

SEQ_DMA_CFG 位域

54.4.10 SEQ_QUE (0x810 + 0x4 * n)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																										SEQ_INT_EN	CHAN_NUM_4_0				
N/A																										RW	RW				
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0

SEQ_QUE [31:0]

位域	名称	描述
5	SEQ_INT_EN	本次转换完成后中断使能，置 1 后，此次转换完成后 SEQ_CVC 标志位置 1
4-0	CHAN_NUM_4_0	本次转换的通道序号

SEQ_QUE 位域

54.4.11 SEQ_HIGH_CFG (0x850)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD								STOP_POS_HIGH												BUF_LEN_HIGH											
N/A								RW												RW											
x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

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

SEQ_HIGH_CFG [31:0]

位域	名称	描述
23-12	STOP_POS_HIGH	
11-0	BUF_LEN_HIGH	

SEQ_HIGH_CFG 位域

54.4.12 PRD_CFG[PRD_CFG] (0xC00 + 0x10 * n)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																			PRESCALE					PRD							
N/A																			RW					RW							
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0

PRD_CFG[PRD_CFG] [31:0]

位域	名称	描述
12-8	PRESCALE	周期模式周期长度的预分频 0: 1* PRD 1：2* PRD 2：4* PRD ..... 15：32K* PRD ..... 31: 2G* PRD PRD 为分频前的 adc 时钟，默认为 AHB 时钟
7-0	PRD	周期模式的周期长度，0 表示关闭周期模式 单位由 prescale 决定

PRD_CFG[PRD_CFG] 位域

54.4.13 PRD_CFG[PRD_THSHD_CFG] (0xC04 + 0x10 * n)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
THSHDH																THSHDL															
RW																RW															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

PRD_CFG[PRD_THSHD_CFG] [31:0]

位域	名称	描述
31-16	THSHDH	监测上限，当转换结果超过上限时，WDOG 标志位置 1
15-0	THSHDL	监测下限，当转换结果低于下限时，WDOG 标志位置 1

PRD_CFG[PRD_THSHD_CFG] 位域

54.4.14 PRD_CFG[PRD_RESULT] (0xC08 + 0x10 * n)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																CHAN_RESULT															
N/A																RO															
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

PRD_CFG[PRD_RESULT] [31:0]

位域	名称	描述
15-0	CHAN_RESULT	周期模式转换结果，周期模式下，该位会周期性得更新 AD 转换结果，如果对应 AD 通道由其他转换模式触发，转换结果也会存放到该位域

PRD_CFG[PRD_RESULT] 位域

54.4.15 SAMPLE_CFG (0x1000 + 0x4 * n)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																				SAMPLE_CLOCK_NUMBER_SHIFT			SAMPLE_CLOCK_NUMBER									
N/A																				RW			RW								
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0

SAMPLE_CFG [31:0]

位域	名称	描述
11-9	SAMPLE_CLOCK_NUMBER_SHIFT	采样长度左移位数 1：sample_clock_number«1 .....
8-0	SAMPLE_CLOCK_NUMBER	采样长度，以 ADC 时钟为单位

SAMPLE_CFG 位域

54.4.16 CONV_CFG1 (0x1104)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																							CONVERT_CLOCK_NUMBER					CLOCK_DIVIDER			
N/A																							RW					RW			
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0

CONV_CFG1 [31:0]

位域	名称	描述
8-4	CONVERT_CLOCK_NUMBER	AD 转换时间长度，单位为 ADC 时钟周期 当 ADC 转换精度为 16 位时，应设为 21，对应于 21 个时钟周期 用户可以设置更小的值，以达到更快的转换速度，但是精度会下降，设置其他值时需同时配置 adc16_config1.con_end_cnt
3-0	CLOCK_DIVIDER	ADC 时钟分频位，ADC 时钟与总线时钟比率 0：1:1 1：1:2 2：1:3 ... 15: 1:16

CONV_CFG1 位域

54.4.17 ADC_CFG0 (0x1108)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SEL_SYNC_AHB	RSVD	ADC_AHB_EN	RSVD	CONVERT_DURATION																RSVD											PORT3_REALTIME
RW	N/A	RW	N/A	RW																N/A											RW
0	x	0	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	x	x	x	x	x	x	x	x	x	x	x	0

ADC_CFG0 [31:0]

位域	名称	描述
31	SEL_SYNC_AHB	1: ADC 时钟与总线时钟同步，此位置 1 时，ADC 时钟必须与总线时钟相同
29	ADC_AHB_EN	1：允许序列转换和抢占转换的 DMA 把转换结果写入内存
27-12	CONVERT_DURATION	
0	PORT3_REALTIME	

ADC_CFG0 位域

54.4.18 INT_STS (0x1110)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TRIG_CMPT	TRIG_SW_CFLCT	TRIG_HW_CFLCT	READ_CFLCT	SEQ_SW_CFLCT	SEQ_HW_CFLCT	SEQ_DMAABT	SEQ_CMPT	SEQ_CVC	DMA_FIFO_FULL	AHB_ERR	RSVD							WDOG														
W1C	W1C	RW	W1C	W1C	RW	W1C	W1C	W1C	RW	RW	N/A							W1C													
0	0	0	0	0	0	0	0	0	0	0	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0

INT_STS [31:0]

位域	名称	描述
31	TRIG_CMPT	抢占转换完成标志位
30	TRIG_SW_CFLCT	软件触发抢占转换冲突标志位
29	TRIG_HW_CFLCT	硬件触发抢占转换冲突标志位
28	READ_CFLCT	读取转换冲突位，如果 WAIT_DIS 位置 1，在 AD 转换进行中，软件读取其他通道的读取结果寄存器后，该位置 1
27	SEQ_SW_CFLCT	软件触发序列转换冲突标志位
26	SEQ_HW_CFLCT	硬件触发序列转换冲突标志位

位域	名称	描述
25	SEQ_DMAABT	序列模式 DMA 放弃标志位，当 STOP_EN 位置 1，并且 DMA 写入到 STOP_POS 指定的停止位置后，该位置 1
24	SEQ_CMPT	序列模式队列完成标志位
23	SEQ_CVC	序列模式单次转换完成标志位，对应的 SEQ_INT_EN 位置 1 后，才会置 1
22	DMA_FIFO_FULL	序列模式或抢占模式 DMA 内部 FIFO 满标志位，DMA 的目标存储器写入速率不足时可能置 1
21	AHB_ERR	序列模式或抢占模式 DMA 错误标志位，如目标地址为非法地址时可能置 1
13-0	WDOG	AD 转换结果监测看门狗标志位

INT_STS 位域

54.4.19 INT_EN (0x1114)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TRIG_CMPT	TRIG_SW_CFLCT	TRIG_HW_CFLCT	READ_CFLCT	SEQ_SW_CFLCT	SEQ_HW_CFLCT	SEQ_DMAABT	SEQ_CMPT	SEQ_CVC	DMA_FIFO_FULL	AHB_ERR	RSVD							WDOG													
W1C	W1C	RW	W1C	W1C	RW	W1C	W1C	W1C	RW	RW	N/A							W1C													
0	0	0	0	0	0	0	0	0	0	0	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0

INT_EN [31:0]

位域	名称	描述
31	TRIG_CMPT	抢占转换完成标志位中断使能位
30	TRIG_SW_CFLCT	软件触发抢占转换冲突标志位中断使能位
29	TRIG_HW_CFLCT	硬件触发抢占转换冲突标志位中断使能位
28	READ_CFLCT	读取转换冲突位中断使能位
27	SEQ_SW_CFLCT	软件触发序列转换冲突标志位中断使能位
26	SEQ_HW_CFLCT	硬件触发序列转换冲突标志位中断使能位
25	SEQ_DMAABT	序列模式 DMA 放弃标志位中断使能位
24	SEQ_CMPT	序列模式队列完成标志位中断使能位
23	SEQ_CVC	序列模式单次转换完成标志位中断使能位
22	DMA_FIFO_FULL	序列模式或抢占模式 DMA 内部 FIFO 满标志位中断使能位
21	AHB_ERR	序列模式或抢占模式 DMA 错误标志位中断使能位
13-0	WDOG	AD 转换结果监测看门狗标志位中断使能位

INT_EN 位域

54.4.20 ANA_CTRL0 (0x1200)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MOTO_EN	RSVD																		ADC_CLK_ON	RSVD										STARTCAL	RSVD	
RW	N/A																		RW	N/A										RW	N/A	
0	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	x	x	x	x	x	x	x	x	x	0	x	x

ANA_CTRL0 [31:0]

位域	名称	描述
31	MOTO_EN	
12	ADC_CLK_ON	在写入任意 ADC16_* 寄存器前，该位必须置 1
2	STARTCAL	1：开始校正

ANA_CTRL0 位域

54.4.21 ANA_STATUS (0x1210)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																								CALON	RSVD						
N/A																								RW	N/A						
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	x	x	x	x	x	x	x

ANA_STATUS [31:0]

位域	名称	描述
7	CALON	校正提示位 1：ADC16 校正进行中

ANA_STATUS 位域

54.4.22 ADC16_PARAMS (0x1400 + 0x2 * n)

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PARAM_VAL															
RW															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ADC16_PARAMS [15:0]

位域	名称	描述
15-0	PARAM_VAL	ADC16 校正参数

ADC16_PARAMS 位域

54.4.23 ADC16_CONFIG0 (0x1444)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD							REG_EN	BANDGAP_EN	CAL_AVG_CFG			RSVD					PREEMPT_EN	CONV_PARAM													
N/A							RW	RW	RW			N/A					RW	RW													
x	x	x	x	x	x	x	0	0	0	0	0	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ADC16_CONFIG0 [31:0]

位域	名称	描述
24	REG_EN	1：打开稳压器
23	BANDGAP_EN	1：打开 Bandgap
22-20	CAL_AVG_CFG	用于校正结果平均 0：1 轮 1：2 轮 2：4 轮 ..... 5：32 轮 其他：保留
14	PREEMPT_EN	1：使能抢占转换抢占功能，抢占转换可以打断进行中的其他模式转换过程
13-0	CONV_PARAM	AD 转换参数

ADC16_CONFIG0 位域

54.4.24 ADC16_CONFIG1 (0x1460)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																			COV_END_CNT					RSVD							
N/A																			RW					N/A							
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	x	x	x	x	x	x	x	x

ADC16_CONFIG1 [31:0]

位域	名称	描述
12-8	COV_END_CNT	AD 转换结束计数器，当使用完全 16 位精度时，此位为默认 1；如果减小转换时间，除需将 conv_cfg1.convert_clock_number 配置为小于 21 的值外，还需配置此位： conv_end_cnt=21-convert_clock_number+1

ADC16_CONFIG1 位域

55 模拟比较器 ACMP

本章节介绍模拟比较器 ACMP 的功能和特性。

55.1 特性总结

模拟比较器 ACMP 的主要特性如下：

- 3.0 至 3.6V 工作
- 支持轨到轨输入
- 可编程滞回
- 支持窗口模式输出
- 支持对输出进行数字滤波
- 支持生成中断
- 支持 DMA
- 内置 8bit DAC

55.2 功能描述

本章节描述模拟比较器 ACMP 的功能。

比较器可以用来比较 PIN 和 MIN 这两个模拟输入信号，当 PIN 的电压高于 MIN 电压时，模拟器输出逻辑 1。反之，当 PIN 的电压低于于 MIN 电压时，模拟器输出逻辑 0。

55.2.1 ACMP 输入配置

用户可以通过配置比较器的 CFG[PINSEL] 位，选择 PIN 的输入来源：

- INN0，固定连接到 DAC_OUT
- INN1
- INN2
- INN3
- INN4
- INN5
- INN6
- INN7

用户可以通过配置比较器的 CFG[MINSEL] 位，选择 MIN 的输入来源：

- INP0，固定连接到 DAC_OUT
- INP1
- INP2
- INP3
- INP4
- INP5
- INP6
- INP7

用户可以通过配置 CFG[HYST] 位来设置比较器的滞回 (hysteresis)。比较器支持 4 个不同水平的滞回。

55.2.2 ACMP 输出控制

ACMP 的输出 OUT 反映了 2 个输入之间电压大小关系。用户也可以通过比较器控制逻辑对输出进行一定的干预。

用户可以通过 CFG[OPOL] 位，对输出取反。

用户可以对比较器输出进行数字滤波，当 CFG[FLTBYPS] 置 0 时，数字滤波器有效，置 1 时，数字滤波器无效。

当数字滤波器有效时，用户配置 CFG[FLTMODE] 可以指定滤波器的工作模式：

- 3'b000，旁路模式，数字滤波器关闭
- 3'b100，滤刺模式，滤波器输入翻转后，输出也会立即翻转，之后会在一定时间内无视滤波器的输入。这个模式下，滤波器输出会紧随输入，同时会避免输出信号出现毛刺。
- 3'b101，延时滤波器，滤波器输入翻转后需要保持一定时间，滤波器输出才会翻转
- 3'b110，滤峰模式，滤波器输入置逻辑 1 后，需要保持一定时间，滤波器输出才会置逻辑 1，而滤波器输入置 0，滤波器输出会立即置 0，这个模式的目的是滤除不够长的输入波峰。
- 3'b111，滤谷模式，滤波器输入置逻辑 0 后，需要保持一定时间，滤波器输出才会置逻辑 0，而滤波器输入置 1，滤波器输出会立即置 1，这个模式的目的是滤除不够长的输入波谷。

用户可以通过 CFG[FLTLEN] 位，配置滤波器的长度。

用户可以通过 CFG[WINEN] 位，打开输出的窗口模式。窗口模式下，即 CFG [WINEN] 位置 1 时，在 ACMP 的 WIN 输入为逻辑 0 时，比较器始终输出逻辑 0，在 WIN 输入为逻辑 1 时，比较器输出正常。

55.2.3 ACMP 工作模式

ACMP 支持高性能模式和常规模式这两种工作模式。用户将 CFG[HPMODE] 位置 1，即可将 ACMP 设置为高性能模式。在高性能模式下，ACMP 的输出延时比常规模式更小，但是功耗更高。

55.2.4 中断和 DMA

ACMP 可以在输出的上升沿,下降沿,或者双边沿时产生中断,用户可以配置 IRQEN[REDGEN] 和 IRQEN[FEDGEN] 位来打开或者关闭相应的中断请求。

同样的，CMP 可以在输出的上升沿，下降沿，或者双边沿时产生 DMA 请求，用户可以配置 DMAEN[REDGEN] 和 IRQEN[FEDGEN] 位来打开或者关闭相应的 DMA 请求。

55.3 ACMP 寄存器列表

ACMP 的寄存器列表如下：

ACMP base address: 0xF30B0000

地址偏移	名称	描述	复位值
0x0000	CHANNEL[CHN0][CFG]	配置寄存器	0x00000000
0x0004	CHANNEL[CHN0][DACCFG]	DAC 配置寄存器	0x00000000
0x0010	CHANNEL[CHN0][SR]	状态寄存器	0x00000000
0x0014	CHANNEL[CHN0][IRQEN]	中断请求使能寄存器	0x00000000
0x0018	CHANNEL[CHN0][DMAEN]	DMA 请求使能寄存器	0x00000000
0x0020	CHANNEL[CHN1][CFG]	配置寄存器	0x00000000

地址偏移	名称	描述	复位值
0x0024	CHANNEL[CHN1][DACCFG]	DAC 配置寄存器	0x00000000
0x0030	CHANNEL[CHN1][SR]	状态寄存器	0x00000000
0x0034	CHANNEL[CHN1][IRQEN]	中断请求使能寄存器	0x00000000
0x0038	CHANNEL[CHN1][DMAEN]	DMA 请求使能寄存器	0x00000000
0x0040	CHANNEL[CHN2][CFG]	配置寄存器	0x00000000
0x0044	CHANNEL[CHN2][DACCFG]	DAC 配置寄存器	0x00000000
0x0050	CHANNEL[CHN2][SR]	状态寄存器	0x00000000
0x0054	CHANNEL[CHN2][IRQEN]	中断请求使能寄存器	0x00000000
0x0058	CHANNEL[CHN2][DMAEN]	DMA 请求使能寄存器	0x00000000
0x0060	CHANNEL[CHN3][CFG]	配置寄存器	0x00000000
0x0064	CHANNEL[CHN3][DACCFG]	DAC 配置寄存器	0x00000000
0x0070	CHANNEL[CHN3][SR]	状态寄存器	0x00000000
0x0074	CHANNEL[CHN3][IRQEN]	中断请求使能寄存器	0x00000000
0x0078	CHANNEL[CHN3][DMAEN]	DMA 请求使能寄存器	0x00000000

表 220: ACMP 寄存器列表

55.4 ACMP 寄存器描述

ACMP 的寄存器详细说明如下：

55.4.1 CHANNEL[CFG] (0x0 + 0x20 * n)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
HYST		DACEN	HPMODE	CMPEN	MINSEL			RSVD	PINSEL			CMPOEN	FLTBYPS	WINEN	OPOL	FLTMODE			SYNCEN	FLTLEN											
RW		RW	RW	RW	RW			N/A	RW			RW	RW	RW	RW	RW			RW	RW											
0	0	0	0	0	0	0	0	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

CHANNEL[CFG] [31:0]

位域	名称	描述
31-30	HYST	比较器滞回设置位 00: 滞回值 0 01: 滞回值 1 10: 滞回值 2 11: 滞回值 3
29	DACEN	DAC 使能位 0: DAC 关闭 1: DAC 使能

位域	名称	描述
28	HPMODE	高性能使能位 0: 高性能模式关闭 1: 高性能模式打开
27	CMPEN	比较器使能位 0: 比较器关闭 1: 比较器打开
26-24	MINSEL	MIN 选择 从比较器输入信号和 DAC 输出中选择一个作为 MIN
22-20	PINSEL	PIN 选择 从比较器输入信号和 DAC 输出中选择一个作为 PIN
19	CMPOEN	比较器输出使能位 0: 比较器输出关闭 1: 比较器输出打开
18	FLTBYPS	比较器输出滤波器旁路 0: 比较器输出会经过数字滤波 1: 比较器输出不经过数字滤波
17	WINEN	比较器窗口模式使能位 0: 比较器窗口模式关闭 1: 比较器窗口模式使能
16	OPOL	比较器输出取反位 0: 比较器输出不变 1: 比较器输出取反
15-13	FLTMODE	此位域定义了比较器输出滤波器的模式 000：旁路模式 100：滤刺模式 101：延时滤波器 110：滤峰模式 111：滤谷模式
12	SYNCEN	比较器输出时钟同步使能位 0: 比较器输出不与比较器时钟同步 1: 比较器输出与比较器时钟同步
11-0	FLTLEN	比较器输出滤波器的长度，单位是时钟周期

CHANNEL[CFG] 位域

55.4.2 CHANNEL[DACCFG] (0x4 + 0x20 * n)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																								DACCFG							
N/A																								RW							

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

CHANNEL[DACCFG] [31:0]

位域	名称	描述
7-0	DACCFG	8 位 DAC 输出值

CHANNEL[DACCFG] 位域

55.4.3 CHANNEL[SR] (0x10 + 0x20 * n)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																														FEDGF	REDGF
N/A																														RW	RW
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0

CHANNEL[SR] [31:0]

位域	名称	描述
1	FEDGF	比较器输出下降沿标志位，该位写 1 清 0
0	REDGF	比较器输出上升沿标志位，该位写 1 清 0

CHANNEL[SR] 位域

55.4.4 CHANNEL[IRQEN] (0x14 + 0x20 * n)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																														FEDGEN	REDGEN
N/A																														RW	RW
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0

CHANNEL[IRQEN] [31:0]

位域	名称	描述
1	FEDGEN	比较器输出下降沿中断请求使能位
0	REDGEN	比较器输出上升沿中断请求使能位

CHANNEL[IRQEN] 位域

55.4.5 CHANNEL[DMAEN] (0x18 + 0x20 * n)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																														FEDGEN	REDGEN
N/A																														RW	RW
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0

CHANNEL[DMAEN] [31:0]

位域	名称	描述
1	FEDGEN	比较器输出下降沿 DMA 请求使能位
0	REDGEN	比较器输出上升沿 DMA 请求使能位

CHANNEL[DMAEN] 位域

56 数模转换器 DAC

本章节介绍数模转换器 DAC 的功能和特性。

56.1 特性总结

数模转换器 DAC 的主要特性如下：

- 最大 1MHz，12bit
- 两个时钟域：AHB，DAC
- 内置 16 位分频器，可以将 DAC 时钟分频至小于等于 1MHz 的数据输出频率。
- 支持直接模式，阶梯模式和内存模式
- 内置 DMA
- 支持中断，DMA 请求

56.2 功能描述

本章节描述数模转换器 DAC 的功能。

56.2.1 直接模式

用户通过写 12 位寄存器，直接输出所需电压。

56.2.2 阶梯模式

DAC 输出电压从起始值到终止值。

- 可以递增或递减；
- 每次阶梯值可配；
- 到达终止值后，可配置停在终止值，或者回到起始值继续；
- 支持 4 组阶梯配置，每组可以独立配置起止值，阶梯值，循环模式，递增递减等；

56.2.3 内存模式

DAC 内置 DMA 可以从指定存储空间中读出一段数据，从 DAC 输出；

DMA 按 32 位从存储空间读出数据，每笔 32 位数据可以配置成一个或两个 DAC 输出数据；

用户可以配置两个存储空间，每个最大 1M 字节，支持最多 512K 个点，DAC 控制器读完一个后自动从另一个读取数据。

56.2.4 触发控制

阶梯模式和内存模式，都可以由软件或硬件触发使能，使能后根据配置的 DAC 输出频率输出数据；

- 软件触发：软件寄存器触发；
- 硬件触发：通过配置互联管理器产生硬件触发；

56.2.5 中断，DMA 请求

内存模式时，支持在每个存储空间结束时发出中断或者 DMA 请求。

56.2.6 时钟控制

DAC 输入有两个时钟，APB 和 DAC 时钟，APB 时钟用于 CPU 读写配置寄存器，以及内部 DMA 在内存模式时通过 AHB 读数据；

DAC 时钟用于模拟电路控制，以及输出数据频率控制；

用户需要配置 dac_cfg1.div_cfg，配置 DAC 输出数据频率；

模拟电路需要一个时钟锁存 DAC 输出数据，直接模式时，用户需要设置 dac_cfg1.ana_clk_en 来打开此时钟；阶梯模式和内存模式时无需配置。

56.3 DAC 寄存器

DAC 的寄存器列表如下：

DAC0 base address: 0xF3090000

DAC1 base address: 0xF3094000

地址偏移	名称	描述	复位值
0x0000	CFG0	配置寄存器 0	0x00000000
0x0004	CFG1	配置寄存器 1	0x00010000
0x0008	CFG2	配置寄存器 2	0x00000000
0x0010	STEP_CFG[STEP0]	Step0 寄存器	0x00000000
0x0014	STEP_CFG[STEP1]	Step0 寄存器	0x00000000
0x0018	STEP_CFG[STEP2]	Step0 寄存器	0x00000000
0x001C	STEP_CFG[STEP3]	Step0 寄存器	0x00000000
0x0020	BUF_ADDR[BUF0]	Buffer 地址寄存器 0	0x00000000
0x0024	BUF_ADDR[BUF1]	Buffer 地址寄存器 1	0x00000000
0x0028	BUF_LENGTH	Buffer 长度寄存器	0x00000000
0x0030	IRQ_STS	状态寄存器	0x00000000
0x0034	IRQ_EN	中断使能寄存器	0x00000000
0x0038	DMA_EN	DMA 使能寄存器	0x00000000
0x0040	ANA_CFG0	模拟配置寄存器 0	0x00000030
0x0044	CFG0_BAK	配置寄存器 0 备份寄存器	0x00000000
0x0048	STATUS0	Status0 寄存器	0x00000000

表 221: DAC 寄存器列表

56.4 DAC 寄存器详细信息

DAC 的寄存器详细说明如下：

56.4.1 CFG0 (0x0)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD				SW_DAC_DATA												RSVD						DMA_AHB_EN	SYNC_MODE	TRIG_MODE	HW_TRIG_EN	DAC_MODE		BUF_DATA_MODE	HBURST_CFG		
N/A				WO												N/A						WO	WO	WO	WO	WO		WO	WO		
x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0

CFG0 [31:0]

位域	名称	描述
27-16	SW_DAC_DATA	直接模式下使用的 DAC 数据（dac_mode==2’b10）
9	DMA_AHB_EN	设置为启用内部 DMA，如果 FIFO 中有足够的空间，它将读取一个突发。在设置此位之前，用户应配置正确的缓冲区起始地址和长度。 仅在缓冲区模式下使用。
8	SYNC_MODE	1: 同步 clk_dac 和 clk_ahb 在同步模式下, 所有硬件触发信号都是脉冲信号，可以获得更快的响应; 0: 异步 clk_dac 和 clk_ahb 所有硬件触发信号应为电平信号，并应超过一个 DAC 时钟周期，用于获得准确的输出频率（无法从 AHB 时钟分频得到的频率）
7	TRIG_MODE	0：单模， 一个触发脉冲将一个 12 位数据发送到 DAC 模拟模块; 1：连续模式， 如果设置了触发信号（SW 或 HW），如果 FIFO 不为空，DAC 将发送数据，如果触发信号清除，DAC 将停止发送数据。
6	HW_TRIG_EN	设置为使用来自 trigger_mux 的触发信号，在单模下，用户应将其配置为脉冲，在连续模式下，配置为电平
5-4	DAC_MODE	00：直接模式，DAC 输出固定配置数据（来自 sw_dac_data） 01：步进模式，DAC 输出从 start_point 到终点，具有配置的步进，可以升步或降步 10：缓冲模式，从缓冲器读取数据，然后输出到模拟模块，如果本地 FIFO 中有足够的空间，内部 DMA 将加载下一个突发， 11: 触发模式，外部触发 DAC 输出 注：hpm63xx, hpm62xx 系列不支持触发模式
3	BUF_DATA_MODE	缓冲区模式的数据结构， 0：每 32 位数据包含 2 个点，b11：0 对应第一个点，b27：16 对应第二个点. 1：每 32 位数据包含 1 个点，b11：0 对应第一个点

位域	名称	描述
2-0	HBURST_CFG	AC 仅支持以下固定突发 000-单; 011-INCR4; 101：INCR8 其他: 保留

CFG0 位域

56.4.2 CFG1 (0x4)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD													ANA_CLK_EN	ANA_DIV_CFG		DIV_CFG															
N/A													RW	RW		RW															
x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

CFG1 [31:0]

位域	名称	描述
18	ANA_CLK_EN	设置为启用模拟时钟（除以 ana_div_cfg） 直接模式和触发模式需要设置此位
17-16	ANA_DIV_CFG	
15-0	DIV_CFG	步进模式和缓冲模式,: DAC 输出速率分频比，应配置为小于 1MHz 的数据速率。 直接模式和触发模式： DAC 数据更新速率，应配置小于 1MHz 的数据速率。 注： 直接模式和触发模式的设置在 hpm63xx，hpm62xx 系列不支持。

CFG1 位域

56.4.3 CFG2 (0x8)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																								DMA_RST1	DMA_RST0	FIFO_CLR	BUF_SW_TRIG	STEP_SW_TRIG3	STEP_SW_TRIG2	STEP_SW_TRIG1	STEP_SW_TRIG0
N/A																								WO	WO	WO	RW	RW	RW	RW	RW
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0

CFG2 [31:0]

位域	名称	描述
7	DMA_RST1	重置 dma 读取指针到 buf1_start_addr; 如果同时设置 dma_rst0 与 dma_rst1，将设置为 buf0_start_addr 当用户使用 dma_rst*，可以设置 fifo_clr 位.
6	DMA_RST0	重置 dma 读取指向 buf0_start_addr
5	FIFO_CLR	清除 FIFO 内容（将读/写指针都设置为 0）
4	BUF_SW_TRIG	软件触发缓冲模式，硬件自动清除
3	STEP_SW_TRIG3	步进模式软件触发 3，硬件自动清除
2	STEP_SW_TRIG2	步进模式软件触发 2，硬件自动清除
1	STEP_SW_TRIG1	步进模式软件触发 1，硬件自动清除
0	STEP_SW_TRIG0	步进模式软件触发 0，硬件自动清除

CFG2 位域

56.4.4 STEP_CFG (0x10 + 0x4 * n)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD		ROUND_MODE	UP_DOWN	END_POINT												STEP_NUM				START_POINT											
N/A		RW	RW	RW												RW				RW											
x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

STEP_CFG [31:0]

位域	名称	描述
29	ROUND_MODE	步进模式工作方式： 0：在终点处停止; 1：重新加载起点，再次步进
28	UP_DOWN	0 表示向上，1 表示向下
27-16	END_POINT	结束点
15-12	STEP_NUM	在每个 DAC 时钟周期内，输出数据变化的 step_num。例如：如果 step_num=3，则输出数据序列为 0，3，6，9... 注意：如果 step_num 不是 1，用户应确保可以访问 end_point 如果 step_num 为 0，则输出数据将始终位于起点
11-0	START_POINT	起始点

STEP_CFG 位域

56.4.5 BUF_ADDR (0x20 + 0x4 * n)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BUF_START_ADDR																														RSVD	BUF_STOP
RW																														N/A	RW
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	x	0

BUF_ADDR [31:0]

位域	名称	描述
31-2	BUF_START_ADDR	缓冲区起始地址，应为 4 字节对齐 AHB 突发不能跨越 1K 字节边界，用户应配置地址/长度/突发以避免此类问题。
0	BUF_STOP	设置此位硬件会在缓冲区 0 末尾停止读取数据

BUF_ADDR 位域

56.4.6 BUF_LENGTH (0x28)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BUF1_LEN																BUF0_LEN															
RW																RW															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

BUF_LENGTH [31:0]

位域	名称	描述
31-16	BUF1_LEN	缓冲区 1 长度，单位为 32 位（4 字节），一个缓冲区最大 256K 字节
15-0	BUF0_LEN	缓冲区 0 长度，单位为 32 位（4 字节），一个缓冲区最大 256K 字节

BUF_LENGTH 位域

56.4.7 IRQ_STS (0x30)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																											STEP_CMPT	AHB_ERROR	FIFO_EMPTY	BUF1_CMPT	BUF0_CMPT
N/A																											W1C	W1C	W1C	W1C	W1C

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

IRQ_STS [31:0]

位域	名称	描述
4	STEP_CMPT	步进模式结束中断状态位，写 1 清零
3	AHB_ERROR	如果 hresp==2'b01（错误）则设置此中断位，写 1 清零
2	FIFO_EMPTY	缓冲模式下，缓冲区空中断状态位，写 1 清零
1	BUF1_CMPT	缓冲模式下，缓冲区 1 完成中断状态位，写 1 清零
0	BUF0_CMPT	缓冲模式下，缓冲区 0 完成中断状态位，写 1 清零

IRQ_STS 位域

56.4.8 IRQ_EN (0x34)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																											STEP_CMPT	AHB_ERROR	FIFO_EMPTY	BUF1_CMPT	BUF0_CMPT
N/A																											RW	RW	RW	RW	RW
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0

IRQ_EN [31:0]

位域	名称	描述
4	STEP_CMPT	标志位中断请求使能
3	AHB_ERROR	标志位中断请求使能
2	FIFO_EMPTY	标志位中断请求使能
1	BUF1_CMPT	标志位中断请求使能
0	BUF0_CMPT	标志位中断请求使能

IRQ_EN 位域

56.4.9 DMA_EN (0x38)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																											STEP_CMPT	RSVD		BUF1_CMPT	BUF0_CMPT
N/A																											RW	N/A		RW	RW
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	x	x	0	0

DMA_EN [31:0]

位域	名称	描述
4	STEP_CMPT	步进模式 DMA 使能位
1	BUF1_CMPT	缓冲区 1 完成 DMA 使能位
0	BUF0_CMPT	缓冲区 0 完成 DMA 使能位

DMA_EN 位域

56.4.10 ANA_CFG0 (0x40)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																							DAC12BIT_LP_MODE	DAC_CONFIG				CALI_DELTA_V_CFG		BYPASS_CALI_GM	DAC12BIT_EN
N/A																							RW	RW				RW		RW	RW
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	1	1	0	0	0	0

ANA_CFG0 [31:0]

位域	名称	描述
8	DAC12BIT_LP_MODE	
7-4	DAC_CONFIG	
3-2	CALI_DELTA_V_CFG	
1	BYPASS_CALI_GM	
0	DAC12BIT_EN	

ANA_CFG0 位域

56.4.11 CFG0_BAK (0x44)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD				SW_DAC_DATA												RSVD						DMA_AHB_EN	SYNC_MODE	TRIG_MODE	HW_TRIG_EN	DAC_MODE		BUF_DATA_MODE	HBURST_CFG		
N/A				RW												N/A						RW	RW	RW	RW	RW		RW	RW		
x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0

CFG0_BAK [31:0]

位域	名称	描述
27-16	SW_DAC_DATA	直接模式下使用的 DAC 数据（dac_mode==2'b10）
9	DMA_AHB_EN	设置为启用内部 DMA，如果 FIFO 中有足够的空间，它将读取一个突发。在设置此位之前，用户应配置正确的缓冲区起始地址和长度。 仅在缓冲区模式下使用。
8	SYNC_MODE	1: 同步 clk_dac 和 clk_ahb 在同步模式下, 所有硬件触发信号都是脉冲信号，可以获得更快的响应; 0: 异步 clk_dac 和 clk_ahb 所有硬件触发信号应为电平信号，并应超过一个 DAC 时钟周期，用于获得准确的输出频率（无法从 AHB 时钟分频得到的频率）
7	TRIG_MODE	0：单模， 一个触发脉冲将一个 12 位数据发送到 DAC 模拟模块; 1：连续模式， 如果设置了触发信号（SW 或 HW），如果 FIFO 不为空，DAC 将发送数据，如果触发信号清除，DAC 将停止发送数据。
6	HW_TRIG_EN	设置为使用来自 trigger_mux 的触发信号，在单模下，用户应将其配置为脉冲，在连续模式下，配置为电平
5-4	DAC_MODE	00：直接模式，DAC 输出固定配置数据（来自 sw_dac_data） 01：步进模式，DAC 输出从 start_point 到终点，具有配置的步进，可以升步或降步 10：缓冲模式，从缓冲器读取数据，然后输出到模拟模块，如果本地 FIFO 中有足够的空间，内部 DMA 将加载下一个突发，
3	BUF_DATA_MODE	缓冲区模式的数据结构， 0：每 32 位数据包含 2 个点，b11：0 对应第一个点，b27：16 对应第二个点. 1：每 32 位数据包含 1 个点，b11：0 对应第一个点
2-0	HBURST_CFG	AC 仅支持以下固定突发 000-单; 011-INCR4; 101：INCR8 其他: 保留

CFG0_BAK 位域

56.4.12 STATUS0 (0x48)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD								CUR_BUF_OFFSET																CUR_BUF_INDEX	RSVD						
N/A								RW																RW	N/A						

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

STATUS0 [31:0]

位域	名称	描述
23-8	CUR_BUF_OFFSET	当前 AHB 读取索引。 注意：根据 DAC 数据速率（div_cfg）和系统总线延迟，此指数与实际 DAC 输出之间有延迟
7	CUR_BUF_INDEX	0 表示缓冲区 0，1 表示缓冲区 1

STATUS0 位域

57 温度传感器 TSNS

本章节介绍温度传感器 TSNS 的功能和特性。

57.1 特性总结

本产品包含温度传感器，可以测量管芯温度。主要特性如下：

- 单次测量
- 连续测量
- 多次测量取平均
- 记录温度上下限
- 温度范围比较
- 温度超范围时产生中断
- 温度超范围时复位芯片

57.2 功能描述

TSNS 通过内置 ADC 对内置温敏二极管进行转换，得到管芯温度。软件需要使能其偏置电路才能进行温度采集。

温度采集过程，由 ADC 对内部温度传感器进行多测采样，并求平均以获得较高的精度。平均次数由 AVERAGE 字段设置，最高支持 128 次平均。

AGE 寄存器，温度传感器模块提供上一次温度采集距离现在时间，该寄存器提供的时间以 24M 时钟周期计数。

温度传感器内部记录出现过的最高和最低温度。可通过 TMAX 和 TMIN 读取，可通过 RECORD_MAX_CLR 和 RECORD_MIN_CLR 清除，并重新记录。

57.2.1 温度采集模式

转换可以由软件触发也可以定时连续采集。

在单次采集模式下，软件通过写入 TRIGGER 进行温度测量，测量完成后 VALID 置 1，表示温度数值有效。

连续模式通过 CONTINOUS 字段设置。在连续模式下，用户通过设置 VALIDITY 字段设置温度采样的时间间隔，时间间隔以 24M 时钟计数。

57.2.2 温度比较功能

温度传感器具有温度超范围产生中断和复位的功能，该功能可通过设置 COMPARE_MIN_EN 和 COMPARE_MAX_EN 允许高温和低温比较。

中断和复位的温度范围独立设置。

中断比较的温度设置在 UPPER_LIM_IRQ 和 LOWER_LIM_IRQ 寄存器中。

复位比较的温度设置在 UPPER_LIM_RST 和 LOWER_LIM_RST 寄存器中。

当测量到的管芯温度超出设置的范围时，温度传感器模块产生标志位。若中断使能位 IRQ_EN 置 1 则产生中断。若复位使能位 RST_EN 置 1 则产生复位。

57.3 TSNS 寄存器

TSNS 的寄存器列表如下：

TSNS base address: 0xF4104000

地址偏移	名称	描述	复位值
0x0000	T	当前温度	0x00000000
0x0004	TMAX	最高温度	0xFF800000
0x0008	TMIN	最低温度	0x007FFFFF
0x000C	AGE	采集时间	0x00000000
0x0010	STATUS	状态寄存器	0x00000000
0x0014	CONFIG	配置寄存器	0x00600300
0x0018	VALIDITY	采样间隔	0x016E3600
0x001C	FLAG	标志位	0x00000000
0x0020	UPPER_LIM_IRQ	中断高温	0x00000000
0x0024	LOWER_LIM_IRQ	中断低温	0x00000000
0x0028	UPPER_LIM_RST	复位高温	0x00000000
0x002C	LOWER_LIM_RST	复位低温	0x00000000
0x0030	ASYNC	异步模式配置	0x00000000
0x0038	ADVAN	高级配置	0x00000000

表 222: TSNS 寄存器列表

57.4 TSNS 寄存器详细信息

TSNS 的寄存器详细说明如下：

57.4.1 T (0x0)

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

T [31:0]

位域	名称	描述
31-0	T	摄氏温度，24 位整数，8 位小数

T 位域

57.4.2 TMAX (0x4)

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

TMAX [31:0]

位域	名称	描述
31-0	T	出现过的最高温度

TMAX 位域

57.4.3 TMIN (0x8)

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

TMIN [31:0]

位域	名称	描述
31-0	T	出现过的最低温度

TMIN 位域

57.4.4 AGE (0xC)

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

AGE [31:0]

位域	名称	描述
31-0	AGE	上次采集时间，上次采集到当前时间的 24M 时钟周期数

AGE 位域

57.4.5 STATUS (0x10)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VALID	RSVD																													TRIGGER	
RO	N/A																													W1C	
0	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0

STATUS [31:0]

位域	名称	描述
31	VALID	温度寄存器数据是否有效 0: 温度尚未采集，温度寄存器中的数据无效 1: 温度已采集，温度寄存器中的数据有效
0	TRIGGER	在触发模式下，软件出发温度采集。若处于非触发模式或采集正在进行，写入将被忽略

STATUS 位域

57.4.6 CONFIG (0x14)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IRQ_EN	RST_EN	RSVD				COMPARE_MIN_EN	COMPARE_MAX_EN	SPEED								RSVD					AVERAGE			RSVD			CONTINUOUS	RSVD		ASYNC	ENABLE
RW	RW	N/A				RW	RW	RW								N/A					RW			N/A			RW	N/A		RW	RW
0	0	x	x	x	x	0	0	0	0	0	0	0	0	0	0	x	x	x	x	x	0	0	0	x	x	x	0	x	x	0	0

CONFIG [31:0]

位域	名称	描述
31	IRQ_EN	使能中断
30	RST_EN	使能复位
25	COMPARE_MIN_EN	允许低温比较
24	COMPARE_MAX_EN	允许高温比较
23-16	SPEED	温度采集步长 24M 时钟周期数，有效范围是 24-255, 缺省值 96 24: 24 周期 25: 25 周期 26: 26 周期 ... 255: 255 周期

位域	名称	描述
10-8	AVERAGE	平均次数，缺省值为 3 0: 不做平均 1: 平均两次 2: 平均四次 . . . 7: 平均 128 次
4	CONTINUOUS	采样模式 0: 触发模式，温度传感器在软件触发后采样温度 1: 连续模式，温度传感器周期性采样温度
1	ASYNC	异步模式, 该模式下无需时钟，但是只能比较高温或低温 0: 正常模式 1: 异步模式
0	ENABLE	使能温度传感器，关闭温度传感器可以降低 SOC 功耗 0: 关闭温度传感器 1: 打开温度传感器

CONFIG 位域

57.4.7 VALIDITY (0x18)

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

VALIDITY [31:0]

位域	名称	描述
31-0	VALIDITY	连续采样时采样间隔 24M 时钟周期数

VALIDITY 位域

57.4.8 FLAG (0x1C)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD										RECORD_MIN_CLR	RECORD_MAX_CLR	RSVD		UNDER_TEMP	OVER_TEMP	RSVD															IRQ
N/A										RW	RW	N/A		RW	RW	N/A															RW

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

FLAG [31:0]

位域	名称	描述
21	RECORD_MIN_CLR	写入 1，清除最低温度
20	RECORD_MAX_CLR	写入 1，清除最高温度
17	UNDER_TEMP	写入 1，清除低温警告
16	OVER_TEMP	写入 1，清高低温警告
0	IRQ	写入 1，清除中断标志

FLAG 位域

57.4.9 UPPER_LIM_IRQ (0x20)

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

UPPER_LIM_IRQ [31:0]

位域	名称	描述
31-0	T	最高温度

UPPER_LIM_IRQ 位域

57.4.10 LOWER_LIM_IRQ (0x24)

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

LOWER_LIM_IRQ [31:0]

位域	名称	描述
31-0	T	最低温度

LOWER_LIM_IRQ 位域

57.4.11 UPPER_LIM_RST (0x28)

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

UPPER_LIM_RST [31:0]

位域	名称	描述
31-0	T	最高温度

UPPER_LIM_RST 位域

57.4.12 LOWER_LIM_RST (0x2C)

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

LOWER_LIM_RST [31:0]

位域	名称	描述
31-0	T	最低温度

LOWER_LIM_RST 位域

57.4.13 ASYNC (0x30)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD							ASYNC_TYPE	RSVD							POLARITY	RSVD					VALUE										
N/A							RW	N/A							RW	N/A					RW										
x	x	x	x	x	x	x	0	x	x	x	x	x	x	x	0	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0

ASYNC [31:0]

位域	名称	描述
24	ASYNC_TYPE	比较类型 0: 高温比较 1: 低温比较
16	POLARITY	内部比较器极性
10-0	VALUE	比较结果

ASYNC 位域

57.4.14 ADVAN (0x38)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD						ASYNC_IRQ	ACTIVE_IRQ	RSVD							SAMPLING	RSVD														NEG_ONLY	POS_ONLY
N/A						RO	RO	N/A							RO	N/A														RW	RW
x	x	x	x	x	x	0	0	x	x	x	x	x	x	x	0	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0

ADVAN [31:0]

位域	名称	描述
25	ASYNC_IRQ	异步中断状态
24	ACTIVE_IRQ	中断状态
16	SAMPLING	采样种
1	NEG_ONLY	仅使用负极性
0	POS_ONLY	仅使用正极性

ADVAN 位域

58 运算放大器 OPAMP

58.1 概述

运算放大器是用于放大输入差分信号，获得相应输出电压的模拟模块，其可扩展 MCU 的模拟应用范围。

- 可以通过配置内部反馈网络和利用外部元器件实现电压跟随、正向放大、反向放大、自定义等功能。
- 可以实现轨到轨的输入输出范围，运放开环增益可在轨到轨输入范围内保持相对一致。
- 可适应宽范围的容性负载（≤50pF）和阻性负载（≥1KOhm），并可根据不同的容性负载优化运放响应速度。
- 基准电压可以通过内部基准电路提供、外部激励源提供、DAC 输出电压提供。
- 输出可以连接 ADC 的输入端口。
- 具有四通道正输入端和四通道负输入端。
- 支持软件直接配置工作模式，或者硬件触发选择预设的工作模式。
- 一组直接配置参数寄存器和八组预设参数寄存器。

58.2 管脚说明

表 223: OPAMP 管脚说明

管脚	方向	功能描述
OPAn_OUT	输出	放大器输出。
OPAn_EXT	输入	辅助功能。
OPAn_INP0	输入	差分输入正端。
OPAn_INN0	输入	差分输入负端。
OPAn_INP1	输入	差分输入 1 正端。
OPAn_INN1	输入	差分输入 1 负端。
OPAn_INP2	输入	差分输入 2 正端。
OPAn_INN2	输入	差分输入 2 负端。
OPAn_INP3	输入	差分输入 3 正端。
OPAn_INN3	输入	差分输入 3 负端。

n=0,1

58.3 功能说明

58.3.1 跟随模式

Vout=Vin

Vin 是输入电压，Vout 是输出电压。

可选择管脚 OPAn_INP0/1/2/3 作为输入端，管脚 OPAn_OUT 是输出端。

58.3.2 正向放大模式

Vout=(Vin-Vref)*Av+Vref

Av=2,4,8,16,32,64,128

Vin 是输入电压，Vref 是基准电压，Vout 是输出电压，Av 是放大倍数。

□ 利用内部电阻反馈网络，内部 GND 作为基准电压。

可选择管脚 OPAn_INP0/1/2/3 作为输入端，管脚 OPAn_OUT 是输出端。

□ 利用内部电阻反馈网络和外部基准电压。

可选择管脚 OPAn_INP0/1/2/3 作为输入端,可选择管脚 OPAn_INN0/1/2/3 连接外部基准电压,管脚 OPAn_OUT 是输出端。

□ 利用内部电阻反馈网络，内部 GND 作为基准电压，以及外部滤波电容。

可选择管脚 OPAn_INP0/1/2/3 作为输入端，管脚 OPAn_INN0/1/2/3 和 OPAn_OUT 之间连接滤波电容，管脚 OPAn_OUT 是输出端。

□ 利用内部电阻反馈网络和外部基准电压，以及外部滤波电容。

可选择管脚 OPAn_INP0/1/2/3 作为输入端,可选择管脚 OPAn_INN0/1/2/3 连接外部基准电压,管脚 OPAn_EXT 和 OPAn_OUT 之间连接滤波电容，管脚 OPAn_OUT 是输出端。

□ 利用内部电阻反馈网络，DAC 输出作为基准电压。

可选择管脚 OPAn_INP0/1/2/3 作为输入端，通过相应配置选通 DAC 输出作为基准电压，管脚 OPAn_OUT 是输出端。

58.3.3 反向放大模式

Vout=(Vin-Vref)*Av+Vref

Av=-2,-4,-8,-16,-32,-64,-128

Vin 是输入电压，Vref 是基准电压，Vout 是输出电压，Av 是放大倍数。

□ 利用内部电阻反馈网络和内部基准电压。

可选择管脚 OPAn_INN0/1/2/3 作为输入端，管脚 OPAn_OUT 是输出端。

□ 利用内部电阻反馈网络和外部基准电压。

可选择管脚 OPAn_INN0/1/2/3 作为输入端,可选择管脚 OPAn_INP0/1/2/3 连接外部基准电压,管脚 OPAn_OUT 是输出端。

□ 利用内部电阻反馈网络和外部基准电压，以及外部滤波电容。

可选择管脚 OPAn_INN0/1/2/3 作为输入端,可选择管脚 OPAn_INP0/1/2/3 连接外部基准电压,管脚 OPAn_EXT 和 OPAn_OUT 之间连接滤波电容，管脚 OPAn_OUT 是输出端。

□ 利用内部电阻反馈网络，DAC 输出作为基准电压。

可选择管脚 OPAn_INN0/1/2/3 作为输入端，通过相应配置选通 DAC 输出作为基准电压，管脚 OPAn_OUT 是输出端。

58.3.4 自定义模式

自由搭建运算放大器外部电路，实现多种功能。

可选择管脚 OPAn_INP0/1/2/3 作为运放正输入端，可选择管脚 OPAn_INN0/1/2/3 作为运放负输入端，管脚 OPAn_OUT 是输出端。

58.4 功能表

功能表中的第一行缩写除 ISO 为电压 ISO_LV_VDD33 外，其余均来自寄存器，可来自直接配置参数，也可来自触发选择的预设参数：

ISO	EN	BP	VW	PS	MS	P7	P2	IS	C3	C2	X1	X0	功能
3.3	x	x	x	x	x	x	x	x	x	x	x	x	isolation
0	0	x	x	x	x	x	x	x	x	x	x	x	关闭 OPAMP。
0	1	x	x	0	x	x	x	x	x	x	x	x	OPAMP 正输入端选通管脚 OPAn_INP0。
0	1	x	x	1	x	x	x	x	x	x	x	x	OPAMP 正输入端选通管脚 OPAn_INP1。
0	1	x	x	2	x	x	x	x	x	x	x	x	OPAMP 正输入端选通管脚 OPAn_INP2。
0	1	x	x	3	x	x	x	x	x	x	x	x	OPAMP 正输入端选通管脚 OPAn_INP3。
0	1	x	x	4	x	x	x	x	x	x	x	x	OPAMP 正输入端选通内部基准电压 0.25*Vsupply。
0	1	x	x	5	x	x	x	x	x	x	x	x	OPAMP 正输入端选通内部基准电压 0.5*Vsupply。
0	1	x	x	6	x	x	x	x	x	x	x	x	OPAMP 正输入端选通内部基准电压 0.75*Vsupply。
0	1	x	x	7	x	x	x	x	x	x	x	x	OPAMP 正输入端悬空。
0	1	x	x	x	0	x	x	x	x	x	x	x	OPAMP 负输入端选通管脚 OPAn_INN0。
0	1	x	x	x	1	x	x	x	x	x	x	x	OPAMP 负输入端选通管脚 OPAn_INN1。
0	1	x	x	x	2	x	x	x	x	x	x	x	OPAMP 负输入端选通管脚 OPAn_INN2。
0	1	x	x	x	3	x	x	x	x	x	x	x	OPAMP 负输入端选通管脚 OPAn_INN3。
0	1	x	x	x	4	x	x	x	x	x	x	x	OPAMP 负输入悬空。
0	1	x	x	x	x	x	x	x	x	6	x	x	选择 5 组补偿电容。
0	1	x	x	x	x	x	x	x	x	7	x	x	选择 7 组补偿电容。
0	1	x	x	x	x	x	x	x	x	0	x	x	选择 9 组补偿电容。
0	1	x	x	x	x	x	x	x	x	1	x	x	选择 11 组补偿电容。
0	1	x	x	x	x	x	x	x	x	2	x	x	选择 14 组补偿电容。
0	1	x	x	x	x	x	x	x	x	3	x	x	选择 18 组补偿电容。
0	1	x	x	x	x	x	x	x	x	4	x	x	选择 23 组补偿电容。
0	1	x	x	x	x	x	x	x	x	5	x	x	选择 30 组补偿电容。
0	1	x	x	x	x	x	x	x	x	0	x	x	(举例) 低负载：Cload≤5pF，Rload≥1Kohm or No Rload。(选择 9 组补偿电容)。
0	1	x	x	x	x	x	x	x	x	3	x	x	(举例) 高负载：Cload≤50pF，Rload≥1Kohm or No Rload。(选择 18 组补偿电容)。
0	1	x	x	x	x	x	0	x	x	x	x	x	选择 2 倍放大倍数。
0	1	x	x	x	x	x	1	x	x	x	x	x	选择 4 倍放大倍数。
0	1	x	x	x	x	x	2	x	x	x	x	x	选择 8 倍放大倍数。
0	1	x	x	x	x	x	3	x	x	x	x	x	选择 16 倍放大倍数。
0	1	x	x	x	x	x	4	x	x	x	x	x	选择 32 倍放大倍数。
0	1	x	x	x	x	x	5	x	x	x	x	x	选择 64 倍放大倍数。
0	1	x	x	x	x	x	6	x	x	x	x	x	选择 128 倍放大倍数。
0	1	x	x	x	x	x	7	x	x	x	x	x	选择 128 倍放大倍数。

ISO	EN	BP	VW	PS	MS	P7	P2	IS	C3	C2	X1	X0	功能
0	1	1	0	<4	4	6	x	0	1	0	0	0	功能：跟随模式。可选择管脚 OPAn_INP0/1/2/3 作为输入端，管脚 OPAn_OUT 是输出端。
0	1	0	0	5	<4	8	x	0	0	x	0	0	功能：反向放大模式（利用内部电阻反馈网络和内部基准电压）。可选择管脚 OPAn_INN0/1/2/3 作为输入端，管脚 OPAn_OUT 是输出端。
0	1	0	0	<4	<4	8	x	0	0	x	0	0	功能：反向放大模式（利用内部电阻反馈网络和外部基准电压）。可选择管脚 OPAn_INN0/1/2/3 作为输入端，可选择管脚 OPAn_INP0/1/2/3 连接外部基准电压，管脚 OPAn_OUT 是输出端。
0	1	0	0	<4	<4	18	x	0	1	x	0	0	功能：反向放大模式（利用内部电阻反馈网络和外部基准电压，以及外部滤波电容）。可选择管脚 OPAn_INN0/1/2/3 作为输入端，可选择管脚 OPAn_INP0/1/2/3 连接外部基准电压，管脚 OPAn_EXT 和 OPAn_OUT 之间连接滤波电容，管脚 OPAn_OUT 是输出端。
0	1	0	0	2	<4	8	x	0	0	x	0	0	功能：反向放大模式（利用内部电阻反馈网络，DAC 输出作为基准电压）。可选择管脚 OPAn_INN0/1/2/3 作为输入端，通过相应配置选通 DAC 输出作为基准电压，管脚 OPAn_OUT 是输出端。
0	1	0	0	<4	4	1	x	0	0	x	0	0	功能：正向放大模式（利用内部电阻反馈网络，内部 GND 作为基准电压）。可选择管脚 OPAn_INP0/1/2/3 作为输入端，管脚 OPAn_OUT 是输出端。
0	1	0	0	<4	<4	9	x	0	0	x	0	0	功能：正向放大模式（利用内部电阻反馈网络和外部基准电压）。可选择管脚 OPAn_INP0/1/2/3 作为输入端，可选择管脚 OPAn_INN0/1/2/3 连接外部基准电压，管脚 OPAn_OUT 是输出端。
0	1	0	0	<4	<4	11	x	0	1	x	0	0	功能：正向放大模式（利用内部电阻反馈网络，内部 GND 作为基准电压，以及外部滤波电容）。可选择管脚 OPAn_INP0/1/2/3 作为输入端，管脚 OPAn_INN0/1/2/3 和 OPAn_OUT 之间连接滤波电容，管脚 OPAn_OUT 是输出端。
0	1	0	0	<4	<4	19	x	0	1	x	0	0	功能：正向放大模式（利用内部电阻反馈网络和外部基准电压，以及外部滤波电容）。可选择管脚 OPAn_INP0/1/2/3 作为输入端，可选择管脚 OPAn_INN0/1/2/3 连接外部基准电压，管脚 OPAn_EXT 和 OPAn_OUT 之间连接滤波电容，管脚 OPAn_OUT 是输出端。

ISO	EN	BP	VW	PS	MS	P7	P2	IS	C3	C2	X1	X0	功能
0	1	0	0	<4	2	9	x	0	0	x	0	0	功能：正向放大模式（利用内部电阻反馈网络，DAC 输出作为基准电压）。可选择管脚 OPAn_INP0/1/2/3 作为输入端，通过相应配置选通 DAC 输出作为基准电压，管脚 OPAn_OUT 是输出端。
0	1	1	0	<4	<4	4	x	0	1	x	0	0	功能：自定义模式（自由搭建运算放大器外部电路，实现多种功能）。可选择管脚 OPAn_INP0/1/2/3 作为运放正输入端，可选择管脚 OPAn_INN0/1/2/3 作为运放负输入端，管脚 OPAn_OUT 是输出端。

表 225: 功能表缩写

缩写	参数名	位数
BP	VBYPASS	1
EN	EN_LV	1
VW	VSWITCH_SEL	3
PS	VIP_SEL	3
MS	VIM_SEL	3
P7	PGA_SEL[7:3]	5
P2	PGA_SEL[2:0]	3
IS	ISEL	2
C3	CSEL[3]	1
C2	CSEL[2:0]	3
X1	OPAOUT_SEL[1]	1
X0	OPAOUT_SEL[1]	1

58.5 OPAMP 寄存器

OPAMP 的寄存器列表如下：

OPAMP0 base address: 0xF30A0000

OPAMP1 base address: 0xF30A4000

地址偏移	名称	描述	复位值
0x0000	CTRL0	控制寄存器	0x00000000

表 226: OPAMP 寄存器列表

58.6 OPAMP 寄存器详细信息

OPAMP 的寄存器详细说明如下：

58.6.1 CTRL0 (0x0)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD					EN_LV	OPAOUT_SEL		VSWITCH_SEL			ISEL		VIM_SEL			GPA_SEL								CSEL				VBYPASS	VIP_SEL		
N/A					RW	RW		RW			RW		RW			RW								RW				RW	RW		
x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

CTRL0 [31:0]

位域	名称	描述
26	EN_LV	enable OPAMP。 设 0 时，除了主电阻串 short R 的 switch 和 Miller cap switch 和 I33 可以被选通外，其余 switch 全部关断。 所有与外部 pad 的连接全部断开（高阻）。opamp_core 的 inp、inm、vout 均为高阻。 opamp_core 的 inp、inm、vctrl、ntail、ptail short 在一起。
25-24	OPAOUT_SEL	00: opamp_core_vout 仅连接 VOUT(pad)。 01: opamp_core_vout 仅连接 VOUT_ADC 10: opamp_core_vout 既连接 VOUT(pad)，又连接 VOUT_ADC。 11: opamp_core_vout 与 VOUT(pad) 和 VOUT_ADC 都断开连接。
23-21	VSWITCH_SEL	100：vswitch=0.3 *vsupply (0.99V @vsupply=3.3V) 101：vswitch=0.35*vsupply (1.155V@vsupply=3.3V) 110：vswitch=0.4 *vsupply (1.32V @vsupply=3.3V) 111：vswitch=0.45*vsupply (1.485V@vsupply=3.3V) 000：vswitch=0.5 *vsupply (1.65V @vsupply=3.3V) 001：vswitch=0.55*vsupply (1.815V@vsupply=3.3V) 010：vswitch=0.6 *vsupply (1.98V @vsupply=3.3V) 011：vswitch=0.65*vsupply (2.145V@vsupply=3.3V)
20-19	ISEL	11：0.667 倍 iref。 00：1 倍 iref。 01：1.333 倍 iref。 10：2 倍 iref。
18-16	VIM_SEL	000：opamp_core 的 inm 选通外部 pad VIM0。 001：opamp_core 的 inm 选通外部 pad VIM1（内部已 cut）。 010：opamp_core 的 inm 选通外部 pad VIM2（内部已 cut）。 011：opamp_core 的 inm 选通外部 pad VIM_DAC（内部已 cut）。其实应该选通内部 DAC 输出。 1xx ：opamp_core 的 inm floating。

位域	名称	描述
15-8	GPA_SEL	2:0 为主电阻串选择： 000-1R; 001-3R; 010-7R; 011-15R; 100-31R; 101-63R; 11x-127R; 7:3 为 function mode, 具体看 OPAMP_function_table
7-4	CSEL	110 Miller cap: 5*0.36pF 111 Miller cap: 6*0.36pF 000 Miller cap: 7*0.36pF 001 Miller cap: 8*0.36pF 010 Miller cap: 10*0.36pF 011 Miller cap: 13*0.36pF 100 Miller cap: 15*0.36pF 101 Miller cap: 18*0.36pF
3	VBYPASS	使主电阻串与 VSSA 的通路断开，无论 PGA_SEL_LV<6> 如何设置。
2-0	VIP_SEL	000：opamp_core 的 inp 选通外部 pad VIP0。 001：opamp_core 的 inp 选通外部 pad VIP1（内部已 cut）。 010：opamp_core 的 inp 选通外部 pad VIP2（内部已 cut）。 011：opamp_core 的 inp 选通外部 pad VIP_DAC（内部已 cut）。其实应该选通内部 DAC 输出。 100：opamp_core 的 inp 选通内部 reference=0.25*vsupply。 101：opamp_core 的 inp 选通内部 reference=0.5*vsupply。 110：opamp_core 的 inp 选通内部 reference=0.75*vsupply。 111：opamp_core 的 inp floating。

CTRL0 位域

59 信息安全模块概述

本章节介绍了本产品的信息安全模块。

本产品的信息安全模块主要划分为以下几类：

- 加解密引擎
  - 安全数据处理器 SDP
  - 在线解密模块 EXIP
- 加解密引擎的密钥管理
  - 密钥管理器 KEYM
  - OTP 中的密钥区
- 真随机数发生器
- 系统安全状态监视器
  - 基于产品生命周期管理
  - 监视 JTAG 调试接口
  - 监视处理器进入 Debug 模式
- BOOT ROM 安全启动机制，保证安全，可信的加载用户程序镜像

59.1 安全数据处理器 SDP

本产品包含一个安全数据处理器 SDP，支持加解密运算 AES-128/256，并支持 AES 的 ECB，CBC 加密模式，支持国密 SM4 算法。SDP 支持哈希运算 SHA-1/256，支持 CRC32 以及国密 SM3 算法。

SDP 支持从密钥管理器 KEY 载入密钥。本产品上，SDP 的以下密钥来自密钥管理器 KEYM。

- MK[0:255]，来自密钥管理器的 MK 密钥输出
- SK0[0:255]，来自密钥管理器的 SK0 密钥输出
- SK1[0:255]，来自密钥管理器的 SK1 密钥输出
- SK2[0:255]，来自密钥管理器的 SK2 密钥输出
- SK3[0:255]，来自密钥管理器的 SK3 密钥输出

59.2 在线解密引擎 EXIP

本产品包括 1 个在线解密引擎 EXIP0，EXIP0 与 XPI0 紧密耦合。用户可以使用 AES-128 CTR 模式按照规范对外部 NOR Flash 存储数据加密。当本产品通过 XPI 连接到加密的外部 NOR Flash 时，EXIP 零等待周期实时在线解密，把加密后的代码还原成明文，供处理器直接在线执行。

EXIP 支持利用 OTP 中存储的 EXIP KEK(Key Encryption Key 加密密钥的密钥) 从 Key Blob 中还原 DEK(Data Encryption Key 加密数据的密钥)。实现对 DEK 的保护。

有关本产品密钥管理情况，请查阅节 59.4。

59.3 密钥管理器

本产品支持密钥管理器 KEYM，密钥管理器支持通过独立的数据通路从 OTP 的密钥区载入密钥，并进行混淆处理后，通过独立的数据通路将密钥传送到安全数据处理器 SDP。实现不依赖处理器参与的密钥加载，保证不向任何软件暴露密钥。

本产品上，密钥管理器支持从以下模块载入密钥信息：

- SMKRNG[0:255]，来自随机数发生器 RNG
- FMK[0:255]，来自 OTP 的密钥区

本产品上，KEYM 通过独立数据通路向 SDP 发送密钥信息：

- MK[0:255]
- SK0[0:255]
- SK1[0:255]
- SK2[0:255]
- SK3[0:255]

有关本产品密钥管理情况，请查阅节 59.4。

59.4 密钥管理总结

本产品上，密钥管理总结如图 81所示：

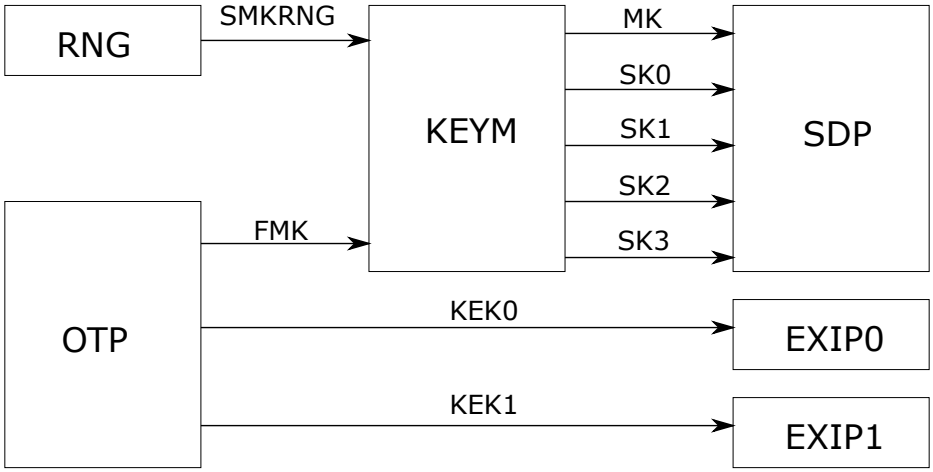


图 81: 密钥管理总结

59.5 一次性可编程存储 OTP

本产品的 OTP 上支持保存各类安全功能有关的数据：

- 128 位调试密钥 Debug Key，本产品支持 JTAG 口锁定，在锁定后，调试器需要提供调试密码，解锁 JTAG 接口，进行调试。
- 128 位 UUID，芯片的唯一标识
- 安全启动的公钥 HASH，为公钥的 256 位 SHA-256 哈希值
- 用于 EXIP 解密包含 DEK 的 Key Blob 的 256 位 KEK，EXIP0 KEK
- 密钥管理器的 256 位 FMK

有关这些安全相关数据的具体信息，请查阅 OTP 及其他安全模块的相关章节。

59.6 真随机数发生器 RNG

本产品包含一个真随机数发生器。熵源为内部模拟噪声源，对片上环境如温度，电压，时钟非常敏感，产生 512 位熵 entropy。RNG 支持错误检测。

59.7 安全管理器 PSEC

本产品包含一个安全管理器 PSEC。

安全管理器 PSEC 的主要功能是根据芯片当前的生命周期，配置芯片安全状态。

安全管理器 PSEC 可以制定安全规则，并检测各类安全事件。当有安全违例事件发生时，采取不同的措施，来保护用户的敏感信息，如密钥等。

安全管理器支持监视如下事件：

- OSC24M 时钟故障
- CPU0 进入调试模式
- VPMC 电源毛刺或者 OSC24M 时钟毛刺，来自电源域监视器 PMON
- VPMC 欠压（Brownout）
- JTAG 端口连接
- JTAG TCK 引脚活动

当安全事件发生，芯片安全状态会转换为 FAIL，会锁定密钥管理器 KEYM 的各个密钥数据通路，并锁定 OTP 的密钥和敏感数据区域及其影子寄存器为不可读。

59.8 安全监视器 PMON

本产品包含一个安全监视器 PMON。安全监视器的作用是监测电源域的电源和时钟。安全监视器在检测到 VPMC 上异常的电源毛刺，或者 OSC24M 上的时钟毛刺，或者 OSC24M 停止工作时会向安全管理器 PSEC 报警。

59.9 BOOT ROM

本产品集成了 BOOT ROM，系统复位后总是最先执行 BOOT ROM 的代码。由只读存储器的物理特性，保证了 BOOT ROM 代码几乎不可能篡改。由此构建了最初的可信安全执行环境。

BOOT ROM 支持如下安全功能：

- 通过 EXIP 和 XPI 直接从加密的串行 NOR Flash 执行代码。
- 安全启动，在跳转到用户软件镜像前可以先验证镜像的来源合法可信。支持 ECDSA 或者 SM2 的签名验证
- 支持加载加密的软件镜像，支持 AES 或者 SM4 加密算法
- Flash loader 支持烧录加密的固件

有关 BOOT ROM 支持的安全功能细节，请查阅 ROM 相关章节。

60 安全数据处理器 SDP

本章节描述了安全数据处理器 SDP 的主要特性和功能。

60.1 特性总结

安全数据处理器 SDP 的主要特性如下：

- 支持 AES-128/256，支持 ECB，CBC 模式
- 支持 SM4
- 支持 SHA-1/256
- 支持 CRC-32
- 支持 SM3
- 内置 DMA，支持数据拷贝，块拷贝和数据充填
- 支持从密钥管理器加载受保护的密钥，或者使用用户写入 SDP 密钥存储器的密钥
- 密钥寄存器可读保护

60.2 功能描述

数据安全处理器 SDP 是一个可以实现数据加解密，计算数据哈希值的安全运算引擎。SDP 作为总线主设备，可以自主访问芯片上的存储外设；SDP 实现自主读取命令描述符，读取待处理数据，写回处理后数据。

60.2.1 命令描述符

数据安全处理器 SDP 工作由命令描述符控制，用户可以在内存中编写命令描述符。SDP 从内存中自主读取命令描述符，按照描述符的配置执行任务，并反馈结果。

SDP 可以通过命令描述符执行的任务总结如下：

- AES-128/256 ECB 模式，对指定长度的明文加密，或者密文解密
- AES-128/256 CBC 模式，对指定长度的明文加密，或者密文解密
- SHA-1/256，对指定长度的数据计算其 Hash 值
- CRC-32，对指定长度的数据计算 CRC32 值
- 将指定长度的数据从源地址拷贝至目标地址
- 对目标地址充填指定长度、指定样式的数据

用户可以通过 CMDPTR 寄存器，指定命令描述符在内存中的地址。

命令描述符的格式如表 227：

地址	名称	描述
CMDPTR + 0x00	NXTCMD	下一条命令描述符指针
CMDPTR + 0x04	PKTCTL	命令控制字，32 位
CMDPTR + 0x08	PKTSRC	源数据地址，32 位
CMDPTR + 0x0C	PKTDST	目标数据地址，32 位
CMDPTR + 0x10	BUFSIZE	数据长度（字节数），32 位
CMDPTR + 0x14	RESERVED	保留
CMDPTR + 0x18	RESERVED	保留

地址	名称	描述
CMDPTR + 0x1C	RESERVED	保留

表 227: SDP 命令描述符格式

NXTCMD 是指向下一条命令描述符，如果 PKTCTL[3]，CHAIN 位置 1，SDP 在执行完当前命令后，继续执行 NXTCMD 指向的命令描述符。

PKTCTL 是命令描述符的控制字，配置 SDP 的任务，描述如下：

- PKTCTL[31:24]，PKTTAG，数据包标签，用户可以为每个数据包打上不同的标签；
- PKTCTL[6]，CIPHIV，使用 AES-CBC 模式时置 1，提示 AES 载入初始向量（Initial Vector）；
- PKTCTL[5]，HASFNL，置 1 提示 HASH 引擎，这是待处理数据的结尾；
- PKTCTL[4]，HASINI，置 1 提示 HASH 引擎，这是待处理数据的起始；
- PKTCTL[3]，CHAIN，置 1 提示 SDP 在执行完当前命令后，从 NXTCMD 取下一条命令描述符；
- PKTCTL[2]，DCRSEMA，置 1 提示 SDP 在执行完当前命令后，将 PKTCNT [CNTVAL] 减 1，当 PKTCNT [CNTVAL] = 0 时，SDP 会停止工作；
- PKTCTL[1]，PKTINT，置 1 时，SDP 在执行完当前命令后，会生成中断请求；

PKTSRC，源数据地址，即 SDP 待处理数据的读取地址，PKTSRC 应当指向待加密的明文，待解密的密文，或者需要通过 HASH 函数压缩的数据。当 SDP 执行数据拷贝时，PKTSRC 指向待拷贝的数据。

PKTDST，目标数据地址，即 SDP 处理后数据的存放地址。PKTDST 应当指向加密后的密文，解密后的明文。当 SDP 执行数据拷贝时，PKTDST 指向拷贝的目标地址。

BUFSIZE，数据的长度（字节数），即 SDP 加密明文的长度，解密密文的长度，需要计算 Hash 值或者 CRC 值的数据块长度，或者需要数据拷贝的数据块长度。

用户在使用 SDP 运算前，需要

- 按需求配置 SDPCR 和 MODCTRL 寄存器
- 编写命令描述符，注意
  - PKTCTL[2]，DCRSEMA 置 1，提示命令执行完成后，PKTCNT [CNTVAL] -1
  - 如果希望执行多条命令，最后一条命令外的命令描述符 PKTCTL[3],CHAIN 位置 1，并把 NXTCMD 填入下一条命令的地址
  - 如果希望命令执行完成后生成中断请求，把命令描述符 PKTCTL[1]，PKTINT 置 1
- 在 CMDPTR 寄存器写入命令描述符地址
- 对 PKTCNT [CNTINCR] 写入需要执行的命令描述符数量，写入 PKTCNT [CNTINCR] 会更新 PKTCNT [CNTVAL] = PKTCNT [CNTVAL] + PKTCNT [CNTINCR]，PKTCNT [CNTVAL] 代表了需要执行的命令数据
  - 如果只需要执行一条命令，写 1
  - 如果希望连续执行 n 条命令，写入 n

60.2.2 AES 加解密引擎

SDP 支持 AES 加解密引擎，AES 为 Advanced Encyrption Standard 的简称，由美国国家标准与技术研究院（NIST）于 2001 年 11 月 26 日发布于 FIPS PUB 197。用户可以自行查阅 AES 的细节。

用户把 SDPCR[CIPHEN] 位置 1，即打开 AES 加解密引擎。

AES 引擎支持 AES-128 和 AES-256，即 128 位和 256 位的 AES 密钥长度。

用户可以通过 MODCTRL[AESALG] 位配置 AES 的密钥长度：

- 4'b0000，AES-128，即密钥长度为 128 位
- 4'b0001，AES-256，即密钥长度为 256 位

AES 引擎支持加密，即通过密钥将明文加密成为密文。AES 引擎支持解密，即通过密钥将密文恢复为明文。用户可以通过 MODCTRL [AESDIR] 位配置 AES 引擎进行的是加密运算还是解密运算：

- MODCTRL [AESDIR] = 1'b0，AES 引擎执行加密运算
- MODCTRL [AESDIR] = 1'b1，AES 引擎执行解密运算

AES 引擎支持 ECB 工作模式和 CBC 工作模式。

ECB 模式称为电子密码本模式（Electronic codebook），是最基本的 AES 加密模式，加密前把全部数据根据数据块大小（AES 数据块长度为 16 字节，即 128 位）分成若干块，之后将每块使用相同的密钥单独加密，解密同理。通常不建议用户使用 ECB 模式加密长度超过一个 AES 数据块的数据。

CBC 模式称为密码分组链接模式（Cipher-block chaining），CBC 模式对于每个待加密的数据块在加密前会先与前一个数据块的密文异或然后再用加密器加密。第一个明文块与一个叫初始化向量（Initial Vector）的数据块异或。解密的过程相似，第一个密文块使用密钥解密后，与初始化向量数据块异或恢复出第一个明文块，之后的密文块解密后与前一个数据块的密文异或恢复出明文。

用户通过 MODCTRL[AESMOD] 位配置 AES 的工作模式：

- MODCTRL[AESMOD] = 4'b0000，AES 工作模式为 ECB 模式
- MODCTRL[AESMOD] = 4'b0001，AES 工作模式为 CBC 模式

用户使用 AES-CBC 模式，需要把第一个命令描述符的 PKTCTL[6]，CIPHIV 位置 1，提示 AES 引擎载入初始化向量（Initial Vector），同时，用户需要把初始化向量（Initial Vector）写入 CIPHIV0 ~ CIPHIV3 寄存器。

60.2.3 AES 密钥配置

AES 作为公开发布的标准算法，其加解密的数据处理步骤是公开可知的。AES 算法的安全性不依赖于算法本身，而是取决于密钥的安全性。

SDP 的 AES 引擎采取了一系列设计，保护 AES 加解密的密钥不泄露。

AES 引擎的密钥有以下选项：

- 从 SDP 内部存储器 KEYRAM 载入密钥，KEYRAM 容量 256 字节，可以存储 16 个 128 位的 AES-128 密钥或者 8 个 256 位的 AES-256 密钥
- 从密钥管理器 KEYM 的专用密钥通路载入密钥，可用密钥有 MK，SK0 ~ SK3

SDP 的内部存储器 KEYRAM 为只可写，不可读存储器。用户可以通过 KEYADDR 和 KEYDAT 寄存器来初始化 SDP 的内部存储器 KEYRAM，方法如下：

- 通过写 KEYADDR [INDEX] 位域来指定 KEYRAM 内部地址索引。
  - KEYADDR [INDEX] = 0，表示指向 KEYRAM 第 1 个 128 位密钥
  - KEYADDR [INDEX] = 1，表示指向 KEYRAM 第 2 个 128 位密钥
  - ……
  - KEYADDR [INDEX] = 15，表示指向 KEYRAM 第 16 个 128 位密钥
- 通过 KEYDAT，写入密钥，KEYDAT 为 32 位寄存器
  - 第 1 次写 KEYDAT，写入密钥 [0:31]

- 第 2 次写 KEYDAT，写入密钥 [32:63]
- 第 3 次写 KEYDAT，写入密钥 [64:95]
- 第 4 次写 KEYDAT，写入密钥 [96:127]

显然，如果用户希望使用 256 位的 AES-256 密钥，需要先将 KEYADDR [INDEX] 取 0 ~ 15 之间的偶数，按以上步骤写入密钥的前 128 位。之后将现有 KEYADDR [INDEX] + 1，再写入 4 次 KEYDAT 寄存器，来写入密钥的后 128 位，密钥 [128:255]

密钥初始化完成后，用户可以通过 MODCTRL[AESKS] 位域来选择密钥：

当 AES 引擎设置为 AES-128 时：

- MODCTRL[AESKS] = 6'h00，选择 KEYRAM 第 1 个 128 位密钥，即 KEYRAM [0:127]
- MODCTRL[AESKS] = 6'h01，选择 KEYRAM 第 2 个 128 位密钥，即 KEYRAM [128:255]
- ……
- MODCTRL[AESKS] = 6'h0E，选择 KEYRAM 第 15 个 128 位密钥，即 KEYRAM [1792:1919]
- MODCTRL[AESKS] = 6'h0F，选择 KEYRAM 第 16 个 128 位密钥，即 KEYRAM [1920:2047]
- MODCTRL[AESKS] = 6'h20，选择来自 KEYM 的 SK0[0:127]
- MODCTRL[AESKS] = 6'h21，选择来自 KEYM 的 SK0[128:255]
- MODCTRL[AESKS] = 6'h22，选择来自 KEYM 的 SK1[0:127]
- MODCTRL[AESKS] = 6'h23，选择来自 KEYM 的 SK1[128:255]
- MODCTRL[AESKS] = 6'h24，选择来自 KEYM 的 SK2[0:127]
- MODCTRL[AESKS] = 6'h25，选择来自 KEYM 的 SK2[128:255]
- MODCTRL[AESKS] = 6'h26，选择来自 KEYM 的 SK3[0:127]
- MODCTRL[AESKS] = 6'h27，选择来自 KEYM 的 SK3[128:255]
- MODCTRL[AESKS] = 6'h2E，选择来自 KEYM 的 MK[0:127]
- MODCTRL[AESKS] = 6'h2F，选择来自 KEYM 的 MK[128:255]

当 AES 引擎设置为 AES-256 时：

- MODCTRL[AESKS] = 6'h00，选择 KEYRAM 第 1 个 256 位密钥，即 KEYRAM [0:255]
- MODCTRL[AESKS] = 6'h01，无效
- ……
- MODCTRL[AESKS] = 6'h0E，选择 KEYRAM 第 8 个 256 位密钥，即 KEYRAM [1792:2048]
- MODCTRL[AESKS] = 6'h0F，无效
- MODCTRL[AESKS] = 6'h20，选择来自 KEYM 的 SK0[0:255]
- MODCTRL[AESKS] = 6'h21，无效
- MODCTRL[AESKS] = 6'h22，选择来自 KEYM 的 SK1[0:255]
- MODCTRL[AESKS] = 6'h23，无效
- MODCTRL[AESKS] = 6'h24，选择来自 KEYM 的 SK2[0:255]
- MODCTRL[AESKS] = 6'h25，无效
- MODCTRL[AESKS] = 6'h26，选择来自 KEYM 的 SK3[0:255]
- MODCTRL[AESKS] = 6'h27，无效
- MODCTRL[AESKS] = 6'h2E，选择来自 KEYM 的 MK[0:255]
- MODCTRL[AESKS] = 6'h2F，无效

60.2.4 SM4 加解密引擎

SDP 支持国密 SM4 分组加解密算法。符合国际标准 ISO/IEC 18033-3：2010/AMD1：2021《信息技术安全技术加密算法第 3 部分：分组密码补篇 1：SM4》。

60.2.5 HASH 模块

SDP 支持一个 HASH 模块。

HASH，一般翻译做散列、杂凑，或音译为哈希，是把任意长度的输入数据，通过 HASH 算法变换成固定长度的输出，该输出就是 HASH 值，也译作散列值或者杂凑值。这种转换是一种压缩映射，也就是，HASH 值的空间通常远小于输入的空间，不同的输入可能会散列成相同的输出，所以理论上不可能从散列值来确定唯一的输入值。简单的说就是一种将任意长度的消息压缩到某一固定长度的消息摘要 (Digest) 的函数，因此，通常也把一段信息的 HASH 值，称为这段信息的摘要 (Digest)。

尽管理论上，一段数据和它的 HASH 值不存在一一对应关系，即总是存在不同的数据，它们的 HASH 值是相同的。对于不同的数据得到相同的 HASH 值，称为碰撞。但是，密码学理论认为，现代的 HASH 算法具备以下特征：

- 从 HASH 值不能反向推导出原始数据，即单向性
- 对输入数据非常敏感，原始数据最微小的修改，会导致 HASH 值大不相同
- 碰撞发生的概率虽然理论上不为 0，但数学上，认为碰撞的概率近似于 0。

因此，HASH 算法得出的数据摘要，或者说 HASH 值，可以用来校验数据的完整性，并在信息安全领域上派生出许多应用。

SDP 的 HASH 模块支持 SHA-1，SHA-256 和 SM3。

SHA-1（Secure Hash Algorithm 1）由美国国家安全局设计，并由美国国家标准技术研究所（NIST）发布为联邦数据处理标准（FIPS）。SHA-1 支持长度不超过 $(2^{64} - 1)$ 位的输入数据，可以生成 160 位（20 字节）HASH 值。

SHA-2（Secure Hash Algorithm 2），由美国国家安全局研发，由美国国家标准与技术研究院（NIST）在 2001 年发布。是 SHA-1 的后继算法。SHA-256 是 SHA-2 的算法之一，支持长度不超过 $(2^{64} - 1)$ 位的输入数据，可以生成 256 位（32 字节）HASH 值。

SM3 密码杂凑算法是由中国国家密码管理局发布的中国商用密码杂凑算法标准，适用于商用密码应用中的数字签名和验证。

用户把 SDPCR[HASHEN] 位置 1，即打开 HASH 模块。

用户可以通过 MODCTRL[HASALG] 位，选择 HASH 算法：

- MODCTRL[HASALG] = 4’b0000，选择 SHA-1
- MODCTRL[HASALG] = 4’b0010，选择 SHA-256

用户在使用 HASH 模块时，必须按需要配置命令描述符的 PKTCTL 字

- 如果此命令描述符处理的数据，包括 HASH 算法输入数据的结尾，要把 PKTCTL[5]，HASFNL 置 1
- 如果此命令描述符处理的数据，包括 HASH 算法输入数据的起始，PKTCTL[4]，HASINI 置 1
- 如果对一段内存内连续的数据执行 HASH 算法，用户可以把 PKTCTL[5] 和 PKTCTL[4] 都置 1，用户利用一条命令描述符就可以计算一段连续的数据的 HASH 值。

60.2.6 数据拷贝和数据充填

SDP 支持内存区段的数据拷贝和数据充填功能。

用户把 SDPCR[MCPEN] 位置 1，即打开数据拷贝功能。这样，SDP 功能与 DMA 类似，通过配置命令描述符，可以把任意长度的数据从源地址指定的内存区段，拷贝到目标地址指定的内存区段。

用户把 SDPCR[CONFEN] 位置 1，即打开数据充填功能。这时，用户配置命令描述符，把需要充填的数据写入源地址字 PKTSRC，SDP 会把目标地址字 PKTDST 起，长度为 BUFSIZE 的区域全部填写 PKTSRC 的值。

60.2.7 数据重排序

SDP 支持对从源地址读取到输入数据，写入目标地址的输出数据，以及密钥进行数据重排序，以适应不同系统的数据排序要求。

MODCTRL[DINSWP] 位域，配置了输入数据的重排序方式：

- 2'b00，输入 32 位数据字节从高位到低位为 [Byte3, Byte2, Byte1, Byte0]，保持不变
- 2'b01，输入 32 位数据字节从高位到低位为 [Byte3, Byte2, Byte1, Byte0]，重新排列为 [Byte2, Byte3, Byte0, Byte1]
- 2'b10，输入 32 位数据字节从高位到低位为 [Byte3, Byte2, Byte1, Byte0]，重新排列为 [Byte1, Byte0, Byte3, Byte2]
- 2'b11，输入 32 位数据字节从高位到低位为 [Byte3, Byte2, Byte1, Byte0]，重新排列为 [Byte0, Byte1, Byte2, Byte3]

MODCTRL[DOUTSWP] 位域，配置了输出数据的重排序方式：

- 2'b00，输出 32 位数据字节从高位到低位为 [Byte3, Byte2, Byte1, Byte0]，保持不变
- 2'b01，输出 32 位数据字节从高位到低位为 [Byte3, Byte2, Byte1, Byte0]，重新排列为 [Byte2, Byte3, Byte0, Byte1]
- 2'b10，输出 32 位数据字节从高位到低位为 [Byte3, Byte2, Byte1, Byte0]，重新排列为 [Byte1, Byte0, Byte3, Byte2]
- 2'b11，输出 32 位数据字节从高位到低位为 [Byte3, Byte2, Byte1, Byte0]，重新排列为 [Byte0, Byte1, Byte2, Byte3]

MODCTRL[KEYSWP] 位域，配置了密钥数据的重排序方式：

- 2'b00，密钥 32 位数据字节从高位到低位为 [Byte3, Byte2, Byte1, Byte0]，保持不变
- 2'b01，密钥 32 位数据字节从高位到低位为 [Byte3, Byte2, Byte1, Byte0]，重新排列为 [Byte2, Byte3, Byte0, Byte1]
- 2'b10，密钥 32 位数据字节从高位到低位为 [Byte3, Byte2, Byte1, Byte0]，重新排列为 [Byte1, Byte0, Byte3, Byte2]
- 2'b11，密钥 32 位数据字节从高位到低位为 [Byte3, Byte2, Byte1, Byte0]，重新排列为 [Byte0, Byte1, Byte2, Byte3]

60.3 SDP 寄存器

60.3.1 寄存器说明

SDP 的寄存器列表如下：

SDP base address: 0xF3040000

地址偏移	名称	描述	复位值
0x0000	SDPCR	SDP 控制寄存器	0x30000000
0x0004	MODCTRL	模式控制寄存器	0x00000000
0x0008	PKTCNT	数据包计数寄存器	0x00000000
0x000C	STA	状态寄存器	0x00000000
0x0010	KEYADDR	密钥地址寄存器	0x00000040
0x0014	KEYDAT	密钥数据	0x00000030
0x0018	CIPHIV[CIPHIV0]	密码初始化向量 0	0x00000000
0x001C	CIPHIV[CIPHIV1]	密码初始化向量 1	0x00000000
0x0020	CIPHIV[CIPHIV2]	密码初始化向量 2	0x00000000
0x0024	CIPHIV[CIPHIV3]	密码初始化向量 3	0x00000000
0x0028	HASWRD[HASWRD0]	哈希数据字 0	0x00000030
0x002C	HASWRD[HASWRD1]	哈希数据字 1	0x00000030
0x0030	HASWRD[HASWRD2]	哈希数据字 2	0x00000030
0x0034	HASWRD[HASWRD3]	哈希数据字 3	0x00000030
0x0038	HASWRD[HASWRD4]	哈希数据字 4	0x00000030
0x003C	HASWRD[HASWRD5]	哈希数据字 5	0x00000030
0x0040	HASWRD[HASWRD6]	哈希数据字 6	0x00000030
0x0044	HASWRD[HASWRD7]	哈希数据字 7	0x00000030
0x0048	CMDPTR	命令字指针	0x00000000
0x004C	NPKTPTR	下一个数据包地址指针	0x00000000
0x0050	PKTCTL	数据包控制寄存器	0x00000000
0x0054	PKTSRC	数据包在内存中源地址	0x00000000
0x0058	PKTDST	数据包处理完后内存中的目标地址	0x00000000
0x005C	PKTBUF	数据包在内存中的大小。	0x00000000

表 228: SDP 寄存器列表

SDP 的寄存器详细说明如下：

60.3.2 SDPCR (0x0)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SFTRST	CLKGAT	CIPDIS	HASDIS	RSVD				CIPHEN	HASHEN	MCPEN	CONFEN	DCRPDI	RSVD	TSTPKT0IRQ	RSVD								RDSCEN	RSVD							INTEN
RW	RW	RO	RO	N/A				RW	RW	RW	RW	RW	N/A	RW	N/A								RW	N/A							RW
0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	x	x	x	x	x	x	x	x	0	x	x	x	x	x	x	x	0

SDPCR [31:0]

位域	名称	描述
31	SFTRST	软复位控制位。 此寄存器位写“1”，然后再写“0”，来软复位 SDP 模块。
30	CLKGAT	SDP 的主要电路的门控时钟控制位。 此为寄存器写“ 1 ”，会关闭 SDP 内部绝大部分逻辑电路的时钟。
29	CIPDIS	加解密硬件禁止位，此位为只读寄存器位；此位为“1”时，加解密的功能不可用；此位为“0”时，加解密的功能可用。
28	HASDIS	哈希硬件禁止位，此位为只读寄存器位；此位为“1”时，哈希的功能不可用；此位为“0”时，哈希的功能可用。
23	CIPHEN	加解密使能位。 此位置“1”时，加解密的功能可用； 此位置“0”时，加解密的功能不可用。
22	HASHEN	哈希软件使能位。 此位置“1”时，哈希的功能可用； 此位置“0”时，哈希的功能不可用。
21	MCPEN	内存数据拷贝功能使能位； 此位置“1”时，内存拷贝的功能可用； 此位置“0”时，内存拷贝的功能不可用。
20	CONFEN	内存数据填充的使能位。 此位置“1”时，数据填充内存的功能可用； 此位置“0”时，常数填充内存功能不可用。
19	DCRPDI	AES 解密禁止位。 写“1”禁止解密功能。 此位仅能通过硬复位清零。
17	TSTPKT0IRQ	保留位。
8	RDSCEN	寄存器存储包描述符使能位。 当设置“1”时，第一个数据包描述符位于寄存器中（CMDPTR，NPKTPTR，...） 当设置“0”时，第一个数据包描述符位于内存中（由 CMDPTR 指向）。
0	INTEN	中断使能，由软件控制。 当置“1”时，SDP 中断使能。 当置“0”时，SDP 中断禁用。

SDPCR 位域

60.3.3 MODCTRL (0x4)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AESALG				AESMOD				AESKS						RSVD	AESDIR	HASALG				CRCEN	HASCHK	HASOUT	RSVD	RSVD		DINSWP		DOUTSWP		KEYSWP	
RW				RW				RW						N/A	RW	RW				RW	RW	RW	N/A	N/A		RW		RW		RW	

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

MODCTRL [31:0]

位域	名称	描述
31-28	AESALG	AES 算法选择。 0x0 = AES 128； 0x1 = AES 256； 0x8 = SM4； 其它值，保留。
27-24	AESMOD	AES 模式选择。 0x0 = ECB； 0x1 = CBC； 其它，保留。
23-18	AESKS	AES 密钥选择。用于选择存储在 SDP 的 16x128 密钥内存中的 AES 密钥，或从 KMAN 中选择密钥。详情如下： 0x00~0x0F：选择密钥内存中的 128 位密钥，0x00~0x0F 为密钥的地址；当地址为偶数时，AES256 模式下将使用来自该地址的 128 位和下一个地址的 128 位密钥作为 256 位 AES 密钥。当地址为奇数时，仅对 AES128 有效。 0x20～0x27：来自密钥管理器 KMAN 的密钥 SK(session key): -0x20，AES128 使用 kman_sk0[127:0] 作为密钥；AES256 使用 kman_sk0[255:0] 作为密钥。 -0x21，AES128 使用 kman_sk0[255:128] 作为密钥；对 AES256 无效。 -0x22，AES128 使用 kman_sk1[127:0] 作为密钥；AES256 使用 kman_sk1[255:0] 作为密钥。 -0x23，AES128 使用 kman_sk1[255:128] 作为密钥；对 AES256 无效。 -0x24，AES128 使用 kman_sk2[127:0] 作为密钥；AES256 使用 kman_sk2[255:0] 作为密钥。 -0x25，AES128 使用 kman_sk2[255:128] 作为密钥；对 AES256 无效。 -0x26，AES128 使用 kman_sk3[127:0] 作为密钥；AES256 使用 kman_sk3[255:0] 作为密钥。 -0x27，AES128 使用 kman_sk3[255:128] 作为密钥；对 AES256 无效。 -0x2E：AES128 使用 kman_mk[127:0] 作为密钥；AES256 使用 kman_mk[255:0] 作为密钥。 -0x2F：AES128 使用 kman_mk[255:128] 作为密钥；对 AES256 无效。 其它值，保留.

位域	名称	描述
16	AESDIR	AES 方向 0x0，AES 加密。 0x1，AES 解密。
15-12	HASALG	HASH 算法选择。 0x0 SHA1 算法 0x1 CRC32 算法 0x2 SHA256 算法 0x3 SM3 算法
11	CRCEN	CRC 使能位。 0x0，禁用 CRC。 0x1，启用 CRC。
10	HASCHK	哈希检查使能位。 0 x0，未启用哈希检查，HASHRSLT0-7 存储哈希计算的结果。 0 x1，启用哈希校验，SDP 计算所得哈希结果会与 HASHRSLT 0-7 寄存器里的预设值进行比较； 对于 SHA1，将使用 HASHRSLT0-3 字，而 HASH 256 将使用 HASH0-7 字。
9	HASOUT	当启用哈希时，该位选择 AES 引擎的输入或输出数据，进行哈希运算。 0x0, AES 输入数据来进行哈希计算 0x1, AES 输出数据来进行哈希计算
5-4	DINSWP	决定 SDP 是否对输入数据进行字节交换，变成大端数据； 当所有位都被设置时，数据被假定为大端格式。
3-2	DOUTSWP	决定 SDP 是否对输出数据进行字节交换，变成大端数据；； 当所有位都被设置时，数据被假定为大端格式
1-0	KEYSWP	决定 SDP 是否对密钥进行字节交换，变成大端数据；； 当所有位都被设置时，数据被假定为大端格式

MODCTRL 位域

60.3.4 PKTCNT (0x8)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD	RSVD	RSVD						CNTVAL								RSVD								CNTINCR							
N/A	N/A	N/A						RO								N/A								RW							
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

PKTCNT [31:0]

位域	名称	描述
23-16	CNTVAL	只读计数器；显示数据包计数器的当前值。
7-0	CNTINCR	写入此处的数值将加到数据包计数器。

PKTCNT 位域

60.3.5 STA (0xC)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TAG								IRQ	RSVD		CHN1PKT0	AESBSY	HASBSY	PKTCNT0	PKTDON	RSVD										ERRSET	ERRPKT	ERRSRC	ERRDST	ERRHAS	ERRCHAIN
RO								W1C	N/A		W1C	RO	RO	W1C	W1C	N/A										W1C	W1C	W1C	W1C	W1C	W1C
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

STA [31:0]

位域	名称	描述
31-24	TAG	正在处理的数据包的标签。
23	IRQ	中断请求；当错误发生时，数据包处理完成时，数据包计数器达到零时，都会有中断请求发生。
20	CHN1PKT0	当前数据报的描述字中，“CHAIN”位为“1”，指示有后续的数据包；但是此时数据包计数器已经到达“0”，正等待等待新的缓冲区数据包。
19	AESBSY	AES 加解密引擎正忙。
18	HASBSY	哈希处理忙状态。
17	PKTCNT0	包计数器现在已经计数到“0”。
16	PKTDON	数据包处理完成后，当数据包控制字中的“PKTINT”位设置位“1”时，将触发此中断。
5	ERRSET	工作模式设置错误。
4	ERRPKT	数据包头访问错误，或状态更新错误。
3	ERRSRC	源缓冲区访问错误
2	ERRDST	目标缓冲区访问错误
1	ERRHAS	哈希结果检查错误
0	ERRCHAIN	当前数据包的 CHAIN 位 = 0，但数据包计数器不为“0”，触发发生缓冲区链错误。

STA 位域

60.3.6 KEYADDR (0x10)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD								INDEX								RSVD														SUBWRD	
N/A								RW								N/A														RW	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0

KEYADDR [31:0]

位域	名称	描述
23-16	INDEX	键索引指针（0－15）。在 SDP 中，有一个 16x128 的密钥缓存区，可以存储 16 个 AES128 密钥或 8 个 AES 256 密钥；该索引用于寻址 16 个 128 位密钥地址。
1-0	SUBWRD	密钥的子字指针。有效值为 0-3。每次写入密钥数据寄存器后，该字段会加 1。

KEYADDR 位域

60.3.7 KEYDAT (0x14)

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

KEYDAT [31:0]

位域	名称	描述
31-0	KEYDAT	密钥寄存器。该寄存器提供对密钥索引寄存器指定的密钥/密钥子字的写访问。 写入此位置的数值，会更新位于子字地址处的密钥的选定子字；写操作还会触发 SUBWRD 寄存器递增到密钥中的下一个更高的字。

KEYDAT 位域

60.3.8 CIPHIV (0x18 + 0x4 * n)

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

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CIPHIV [31:0]																															
RW																															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位域	名称	描述
31-0	CIPHIV	AES 加解密的初始化向量。

CIPHIV 位域

60.3.9 HASWRD (0x28 + 0x4 * n)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
HASWRD [31:0]																															
RW																															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0

位域	名称	描述
31-0	HASWRD	哈希数据字 - HASH 结果位；如果启用了哈希结果检查，此寄存器存储预期的哈希结果；当未启用哈希检查时，哈希引擎将在这里存储最终的哈希结果 [31:0]。 如果启用了 CRC 模式，如果启用了 CRC 校验，则此工作存储 CRC 预期结果；如果未启用 CRC 校验，此寄存器存储最终计算出的 CRC 结果。

HASWRD 位域

60.3.10 CMDPTR (0x48)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CMDPTR [31:0]																															
RW																															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位域	名称	描述
31-0	CMDPTR	当前命令地址寄存器指向内存中的要执行（或当前正在执行）的描述符地址。

CMDPTR 位域

60.3.11 NPKTPTR (0x4C)

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

NPKTPTR [31:0]

位域	名称	描述
31-0	NPKTPTR	下一个数据包地址指针。

NPKTPTR 位域

60.3.12 PKTCTL (0x50)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PKTTAG								RSVD																	CIPHIV	HASFNL	HASINI	CHAIN	DCRSEMA	PKTINT	RSVD
RW								N/A																	RW	RW	RW	RW	RW	RW	N/A
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

PKTCTL [31:0]

位域	名称	描述
31-24	PKTTAG	包标签。
6	CIPHIV	使用 AES-CBC 模式时置 1，提示 AES 装载初始向量。
5	HASFNL	是否是哈希终止包
4	HASINI	是否是哈希初始包
3	CHAIN	指示是否有下一个数据包；也就是下一个命令指针寄存器是否必须被加载。
2	DCRSEMA	在当前操作结束时，信号量是否必须递减操作。当信号量达到零值时，将不再进行进一步的数据包处理。
1	PKTINT	此位的设置，决定是否必须在数据包完成后发出中断。

PKTCTL 位域

60.3.13 PKTSRC (0x54)

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

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

PKTSRC [31:0]

位域	名称	描述
31-0	PKTSRC	数据包内存源地址。

PKTSRC 位域

60.3.14 PKTDST (0x58)

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

PKTDST [31:0]

位域	名称	描述
31-0	PKTDST	数据包内存目标地址。

PKTDST 位域

60.3.15 PKTBUF (0x5C)

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

PKTBUF [31:0]

位域	名称	描述
31-0	PKTBUF	

PKTBUF 位域

61 在线解密引擎 EXIP

本章节描述了在线解密引擎 EXIP 的主要特性和功能。

61.1 特性总结

在线解密引擎 EXIP 可以实现对加密的外部 NOR Flash 实时解密，EXIP 的主要特性有：

- 支持 AES-128 CTR 模式解密
- 支持 4 个区段，每个区段可以使用不同的密钥加密
- 支持通过 Key Blob 封装数据加密密钥等敏感信息（符合 RFC3394）
- 支持硬件解封 Key Blob（符合 RFC3394）
- 支持从 OTP 读取用于解封 Key Blob 的 KEK(加密密钥的密钥)

61.2 功能描述

在本产品上，用户可以通过 XPI 控制器连接多种不同品牌，不同容量的串行 NOR Flash。XPI 控制器支持将外部存储器直接映射到系统的地址空间上。因此处理器、DMA 等主设备可以像访问片上内存一样，直接访问外部存储器，载入数据或者执行代码。这种不需要把外部存储器数据先复制到内存，再执行的访问方式，称为在线执行 (Execution In Place)。在线解密引擎 EXIP 与 XPI 控制器紧密耦合，支持外部串行 NOR Flash 在线执行的同时，对数据实时解密。因此，只要用户通过 AES-128 CTR 算法，对外部 NOR Flash 上的数据和代码加密，就可以实现对数据和代码的保护。

61.2.1 EXIP 区段，密钥，计数器

EXIP 支持 4 个区段，每个区段都可以支持独立的密钥和计数器。

每个区段可以定义区段的起始地址和结尾地址，起始地址和结尾地址之间的区域，就是加密区域。EXIP 与 XPI 控制器紧密耦合，总线主设备（处理器或 DMA 等）通过 XPI 发起对外部存储器访问时，EXIP 会监控访问的系统地址。如果该地址位于加密区域内，就会执行解密操作。

RGNx_DSCR_W0 [START] 存放区段 x 的起始地址，RGNx_DSCR_W1 [END] 存放区段 x 的结尾地址。EXIP 要求加密区域长度的单位为 1KB，起始地址必须 1KB 对齐，假设加密区域长度为 x KB，结尾地址应为 START + x*1024 -1。因此 START 只取高 22 位，低 10 位应当为 0。而 END 只取高 22 位，低 10 位应当为 1。

EXIP 支持 AES-128 CTR 模式解密，密钥长度为 128 位。密钥存放在寄存器 RGNx_KEY0，RGNx_KEY1，RGNx_KEY2，RGNx_KEY3。

计数器长度与 AES 数据块长度相同，也为 128 位。AES CTR 模式要求使用相同密钥加密的数据，每一个数据块对应的计数器值不能重复。EXIP 对每一个 128 位（16 字节）的数据块，其计数器，由数据块的 32 位系统地址和 64 位的 NONCE 组成，其中 NONCE 存放在 RGNx_CTR0 和 RGNx_CTR1 中。计数器组成方式如下：

- Counter[127:96] = CTR0[31:0]
- Counter[95:64] = CTR1[31:0]
- Counter[63:32] = CTR0[31:0] XOR CTR1[31:0]
- Counter[31:0] = Sysaddr[31:0] AND 0xFFFFFFF0

其中：

- CTR0 为 64 位长 NONCE 的低 32 位
- CTR1 为 64 为长 NONCE 的高 32 位

- SysAddr 为密文块的系统映射地址。密文块长度为 128 位 / 16 字节，一个密文块对应一个计数器值，因此计数器值只取地址的高 28 位，地址低 4 位置 0。

61.2.2 EXIP 的密钥封装和密钥解封

EXIP 使用 RFC3394 定义的密钥封装和密钥解封算法，保护 EXIP 解密用的密钥和相关敏感数据。为了便于区分，把 EXIP 用于密钥封装/解封的密钥称为 KEK(Key-Encryption Key 密钥加密密钥)。把 EXIP 用户解密外部存储器上数据/代码的密钥，称为 DEK(Data Encryption Key)。

EXIP 的每一个区段，需要封装的 DEK 和相关数据有 5 个 64 位数据块，共 40 字节的数据。总结如下：

地址偏移	描述
0x00	DEKx[31:0]，即 RGNx_KEY0 存放的部分密钥
0x04	DEKx[63:32]，即 RGNx_KEY1 存放的部分密钥
0x08	DEKx[95:64]，即 RGNx_KEY2 存放的部分密钥
0x0C	DEKx[127:96]，即 RGNx_KEY3 存放的部分密钥
0x10	NONCE[31:0]，即 RGNx_CTR0 存放的部分 NONCE
0x14	NONCE[63:32]，即 RGNx_CTR1 存放的部分 NONCE
0x18	START，即 RGNx_RGN_SA[31:10]，加密区域起始地址, 其他位置 0
0x1C	END，即 RGNx_RGN_EA[31:10]，加密区域结尾地址 RO，即 RGNx_RGN_EA[2]，置 1 时，RGNx 的配置寄存器读保护 DECEN，即 RGNx_RGN_EA[1]，置 1 时，EXIP 对此区段执行解密 VALID，即 RGNx_RGN_EA[0]，置 1 时，区段 x 有效 其他位置 1
0x20	充填数据，全部置 0
0x24	充填数据，全部置 0

表 229: EXIP Key Wrap 数据汇总

这些数据经过密钥封装 (Key Wrap) 过程后，得到 6 个 64 位密文块，共 48 字节。

当 CFG 寄存器的 BE 位和 KBPE 位置 1 后，EXIP 会启动硬件的密钥解封过程。EXIP 会通过 XPI 控制器，直接从外部存储器的起始地址取密文块，密文块在外部存储器的位置如下：

地址偏移	描述
0x00	区段 0，RGN0 的 DEK，NONCE，区域地址信息的密文包
0x40	区段 1，RGN1 的 DEK，NONCE，区域地址信息的密文包
0x80	区段 2，RGN2 的 DEK，NONCE，区域地址信息的密文包
0xC0	区段 3，RGN3 的 DEK，NONCE，区域地址信息的密文包

表 230: EXIP Key Wrap 数据汇总

EXIP 一旦开始解封密文包，会自动执行 4 个区段密文包的解封过程 (Key Unwrap)。解封完成后，会将恢复的 DEK,NONCE,SART,END 等信息载入 RGNx 的对应寄存器。如果解封后得到的初始值与定义的 IV (0xA6A6A6A6A6A6A6A6) 不一致，就会提示错误，RGNx_DESC_W1[0] VALID 位会置 1，同时 STA[KBERR] 标志位置 1。

用户如果实际使用的区段小于 4 个，也需要按照密钥封装的步骤生成全部 4 个区段的密文包，并烧录在 Flash 的对应地址，把不用的区段起始地址和结尾地址都填 0 即可。

EXIP 用于密码解封的 KEK 从片上 OTP 的专用数据通路直接加载，KEK 一旦烧录到 OTP 指定位置，并配置恰当的读写保护后，就对软件不再可见。有关 OTP 的烧录和读写保护配置，用户可以查阅 OTP 的相关章节。

本产品上，EXIP 的打开和密钥解封由 BOOT ROM 管理，OTP 中的 ENCRYPT_XIP 字段被置 1 后。ROM 在 XPI NOR Flash 启动中会强制打开 EXIP。

总结，用户使用 EXIP 实现在线解密执行：

- 生成 KEK，并利用 KEK 对 DEK，NONCE，START，END 等数据进行密钥封装 (KeyWrap)，生成密文包，注意要生成全部 4 个 EXIP 区段的密文包
- 把密文包烧录到外部串行 NOR Flash 的指定地址
- 按照 DEK，NONCE 对镜像加密，加密后烧录到外部串行 NOR Flash
- 将 KEK 烧录到 OTP 的指定位置，并配置读写保护
- 烧录 OTP 中的 ENCRYPT_XIP 字段为 1
- 复位后，芯片即支持从加密后的外部串行 NOR Flash 启动

61.3 附录

61.3.1 RFC3394 简介

EXIP 支持 AES-128 CTR 模式解密外部 NOR Flash 上加密的数据和代码。显然，这些数据及代码的安全，取决于 EXIP 解密用的密钥的安全。EXIP 支持一种称为 Key Wrap 的密钥封装算法，来保护 EXIP 解密用的密钥和其他相关敏感数据。

EXIP 使用的 Key Wrap 算法符合 RFC3394 “Advanced Encryption Standard (AES) Key Wrap Algorithm”。RFC3394 定义了一种称为密钥封装 (Key wrap) 的算法，算法基于 AES 标准，可以对任意长度的数据进行加密保护，并提供数据的完整性检验。RFC3394 把用来封装受保护数据的密钥称为 KEK(Key-encryption Key)，即密钥加密密钥。KEK 可以是 AES 标准支持的密钥，AES-128，AES-192，AES-256。EXIP 使用的 KEK 为 128 位长。

RFC3394 定义，受保护的数据，可以以 64 位为单位分成若干个数据块。数据块个数最少为 2 个，最大个数没有限制。n 个明文数据块经过密钥封装的过程后，得到 n+1 个密文数据块。而 n+1 个密文数据块，经过密钥解封过程，可能还原成 n 个明文数据块。

引用 RFC3394 的密钥封装 Key Wrap 步骤如下：

输入: 明文, n个64位数据块 {P1, P2, ..., Pn}以及密钥K (即KEK).
输出: 密文, (n+1)个64位密文块{C0, C1, ..., Cn}.

步骤1，初始化
设置A为初始值IV
For i = 1 to n
  R[i] = P[i]

步骤2，计算中间值
For j = 0 to 5
  For i=1 to n
    B = AES(K, A | R[i])
    A = MSB(64, B) ^ t 其中t = (n*j)+i
    R[i] = LSB(64, B)

步骤3，输出结果
C[0] = A
For i = 1 to n
  C[i] = R[i]

引用 RFC3394 的密钥解封 Key Unwrap 步骤如下：

输入: 密文, (n+1)个64位密文块{C0, C1, ..., Cn}, 以及密钥 K (即KEK).
输出: 明文, n个64位数据块{P0, P1, K, Pn}.

步骤1，初始化
A = C[0]
For i = 1 to n
  R[i] = C[i]

步骤2，计算中间值
For j = 5 to 0
  For i = n to 1
    B = AES-1(K, (A ^ t) | R[i]) 其中t = n*j+i
    A = MSB(64, B)
    R[i] = LSB(64, B)

步骤3，输出结果
如果 A与初始值IV相符，则
For i = 1 to n
  P[i] = R[i]
否则表示出错

以上密钥封装和解封步骤中使用的表达式说明如下：

AES(K, W) 使用AES算法以密钥K对数据W加密
AES-1(K, W) 使用AES算法以密钥K对数据W解密
MSB(j, W) 返回数据W的j个MSB，即最高的j位
LSB(j, W) 返回数据W的j个LSB，即最低的j位
B1 \^ B2 把数据B1和B2按位异或
B1 | B2 把数据B1和B2拼接
K 即KEK，密钥加密密钥

n 64位数据块的数目

s 密钥封装的步骤数目 s = 6n

P[i] 第i个64位明文数据块

C[i] 第i个64位密文数据块

A 64位寄存器，存放完整性检验数据

R[i] 一组64位的寄存器阵列，其中i = 0, 1, ……, n

IV 封装时的64位的初始值,解封时用于与A比较,EXIP使用RFC3394定义的默认初始值:0xA6A6A6A6A6A6A6A6

62 随机数发生器 RNG

本章节介绍随机数发生器 RNG 的主要特性和功能。

62.1 特性总结

RNG 的特性如下：

- 多个熵源的 256 位硬件真随机发生器（TRNG）来生成种子（(SEED)
- 符合 NIST 标准的 160 位硬件伪随机数发生器（PRNG），伪随机数的种子是 256 位；
- 支持数字签名标准（DSS）中定义的密钥生成算法；
- 支持软件不可见的随机数传递接口；

62.2 功能描述

本章节描述随机数发生器 RNG 的功能。

62.2.1 RNG 初始化

RNG 模块在使用前需要初始化，初始化流程如下：

1. CMD 寄存器的 CLRERR 位置 1，清除所有的错误和中断
2. CMD 寄存器的 GENSD 位置 1，开始生成种子（SEED）
3. 等待 STA 寄存器的 FSTSDDN 标志位置 1，代表 SEED 就绪

62.2.2 RNG 生成 SEED

RNG 在初始化完成后，即可以开始生成 SEED。

RNG 支持自动生成 SEED，也支持在软件触发的 SEED 生成。当 CTRL 寄存器的 AUTRSD 位置 1 时，RNG 可以自动重新生成 SEED。如果 AUTRSD 位置 0，用户需要把 CMD 寄存器的 GENSD 位置 1，来指示 RNG 生成一个新 SEED。当 SEED 生成完毕时，STA 寄存器的 NEWSDDN 标致位会被置 1。

每当存储随机数的 FIFO（五个字共 160 位）为空时，RNG 会基于种子快速生成 160 位随机数填充到 FIFO；当前的种子被用来生成一定数量的的随机数后，RNG 会通过状态寄存器来标识需要生成新的种子，这时，如果自动种子生成模式（AUTRSD 为 1）被使能，会自动生成种子；否则，需要软件设置 CMD.GENSD 位来触发新的种子的生成。

可以通过 FO2B 寄存器读取 FIFO 里存储器的随机数。可以通过 STA 寄存器的 FRNNU 标志位，判断当前 FIFO 存有的随机数字数。

RNG 支持随机数直接输出端口，通过硬件直接传给其它模块，这时随机数对软件不可见；随机数传递步骤如下：

1. 通过 STA 寄存器的 FLVL 标志位，判断 SEED FIFO 内存在有效数据
2. 软件依次读取寄存器 R2SK[0] ~R2SK[7]，当读取 R2SK[7] 完成后，SEED 端口上的 256 位的 SEED 数据有效。

注意，软件读寄存器 R2SK[0] ~R2SK[7]，返回值总是 0。真实有效的随机数会从输出端口输出给其他模块。

62.2.3 RNG 自测试

RNG 支持通过自测试，确保相关电路工作正常。

用户把 CMD 寄存器的 SLFCHK 位置 1，RNG 模块即开始自测试流程。自测结束时，STA 寄存器的 SLFCHKDN 标志位会置 1，如果自测出现问题，SLFCHNPF 标志位会置 1。

62.2.4 中断

RNG 支持生成以下中断请求：

- RNG 生成 SEED 完毕或者自测完毕时
- RNG 发生错误时

CTRL 寄存器的 MIRQDN 位和 MIRQERR 位清 0 时，支持生成相应中断。

62.3 RNG 寄存器列表

RNG 的寄存器列表如下：

RNG base address: 0xF304C000

地址偏移	名称	描述	复位值
0x0000	CMD	命令寄存器	0x00000000
0x0004	CTRL	控制寄存器	0x00000000
0x0008	STA	状态寄存器	0x00000000
0x000C	ERR	错去指示寄存器	0x00000000
0x0010	FO2B	随机数到总线寄存器	0x00000000
0x0020	R2SK[FO2S0]	随机数到 KMAN/SDP 寄存器	0x00000000
0x0024	R2SK[FO2S1]	随机数到 KMAN/SDP 寄存器	0x00000000
0x0028	R2SK[FO2S2]	随机数到 KMAN/SDP 寄存器	0x00000000
0x002C	R2SK[FO2S3]	随机数到 KMAN/SDP 寄存器	0x00000000
0x0030	R2SK[FO2S4]	随机数到 KMAN/SDP 寄存器	0x00000000
0x0034	R2SK[FO2S5]	随机数到 KMAN/SDP 寄存器	0x00000000
0x0038	R2SK[FO2S6]	随机数到 KMAN/SDP 寄存器	0x00000000
0x003C	R2SK[FO2S7]	随机数到 KMAN/SDP 寄存器	0x00000000

表 231: RNG 寄存器列表

62.3.1 RNG 寄存器描述

RNG 的寄存器详细说明如下：

62.3.2 CMD (0x0)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																									SFTRST	CLRERR	CLRINT	RSVD		GENSD	SLFCHK
N/A																									RW	RW	RW	N/A		RW	RW
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

CMD [31:0]

位域	名称	描述
6	SFTRST	软复位控制位, 该位是自清零的。 0 不执行软件复位 1 执行软件复位
5	CLRERR	错误清除位，清除 ESR 寄存器中的错误和 RNG 中断，该位是自清零的。 0 不清除错误和对应的中断 1 清除错误和对应的中断
4	CLRINT	中断清除位，如果不存在错误，则清除 RNG 中断，该位是自清零的。 0 不清除中断 1 清除中断
1	GENSD	生成种子命令位，写“1”触发种子的生成；种子生成结束后此位自动清零。
0	SLFCHK	自检命令为，写“1”触发电路自检；自检结束后，此位自动清零。

CMD 位域

62.3.3 CTRL (0x4)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																									MIRQERR	MIRQDN	AUTRSD	RSVD		FUFMOD	
N/A																									RW	RW	RW	N/A		RW	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

CTRL [31:0]

位域	名称	描述
6	MIRQERR	中断屏蔽位，屏蔽因为内部错误引起的的中断请求
5	MIRQDN	中断屏蔽位，屏蔽种子产生和自检完成时产生的中断。这些作业的状态可以通过以下方式查看： • 读取 STA 并查看种子生成完成和自检完成位（STA[SDN, STDN]） • 查看正 CMD 寄存器里的生成种子或自检位 (CMD[GS,ST])，如果这些位仍为“1”，表明操作仍在进行
4	AUTRSD	种子自动生成控制位；当此位为高时，需要种子生成时，会自动触发种子生成。

位域	名称	描述
1-0	FUFMOD	FIFO 下溢响应模式 0X 返回全零并设置 ESR[FUFE] 10 返回全零，并产生总线传输错误 11 返回全零，产生中断（覆盖 CTRL[MASKERR]）

CTRL 位域

62.3.4 STA (0x8)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD								SCPF			RSVD				FUNCERR	FSIZE				FRNNU				RSVD	NSDDN	FSDDN	SCDN	RSDREQ	IDLE	BUSY	RSVD
N/A								RO			N/A				RO	RO				RO				N/A	RO	RO	RO	RO	RO	RO	N/A
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

STA [31:0]

位域	名称	描述
23-21	SCPF	自检结果位：0x1, 自检失败；0x0, 自检成功
16	FUNCERR	错误标识位，当为“1”时，标识有功能错误
15-12	FSIZE	存储随机数的 FIFO 的大小，以字为单位。
11-8	FRNNU	存储随机数的 FIFO 内当前随机数的个数。
6	NSDDN	新种子生成完成标识位
5	FSDDN	第一个种子生成完成标识位
4	SCDN	自检结束标志位，通过硬件复位或启动新的自检来清除。 通过设置 CMD[ST] 启动。 0 自检未完成 1 上次重启的自检完成
3	RSDREQ	需要重新产生种子。 当此位置高时，表示需要重新产生 RNG 的种子。种子的产生需要设置 CMD[GS] 来完成，或者 如果设置了 CTRL[ARS]，则自动执行。
2	IDLE	保留位
1	BUSY	忙标志位；当为 1 时，表示 RNG 引擎正忙于种子产生或随机数生成、自检等。

STA 位域

62.3.5 ERR (0xC)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD								RSVD								RSVD								RSVD		FUFE	RSVD	SCKERR	RSVD	RSVD	RSVD
N/A								N/A								N/A								N/A		RO	N/A	RO	N/A	N/A	N/A
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	x	x	0	0	0	0	0	0

ERR [31:0]

位域	名称	描述
5	FUFE	FIFO 访问错误（下溢）
3	SCKERR	自检错误指示位；只能通过硬件，或软件复位向 CMD[CE] 写入 1 清零。 0 自检成功。 1 自检失败。

ERR 位域

62.3.6 FO2B (0x10)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FO2B																															
RO																															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

FO2B [31:0]

位域	名称	描述
31-0	FO2B	软件读取得随机数 FIFO 数据。

FO2B 位域

62.3.7 R2SK (0x20 + 0x4 * n)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FO2S0																															
RO																															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

R2SK [31:0]

位域	名称	描述
31-0	FO2S0	软件读此此寄存器，将会得到“0”，实际得随机数 FIFO 会输出到 KMAN，可作为 SDP AES 的密钥。请确保 STA 的 FLVL 位在读此地址时不为“0”，否则 SDP 将无法从随机数 FIFO 读到数据。

R2SK 位域

63 密钥管理器 KEYM

本章节描述了密钥管理器 KEYM 的主要特性和功能。密钥管理器的主要功能是为加解密引擎输送密钥。实现安全密钥管理，保证密钥不暴露给任何软件。

63.1 特性总结

密钥管理器 KEYM 的主要特性：

- 从其他模块载入密钥
- 支持对输入密钥加扰
- 向加密引擎输送密钥
- 支持软件密钥 SMK
- 支持生成多个 SK（session key）
- 支持由多个密钥合成一个密钥

63.2 功能描述

本章节描述密钥管理器 KEYM 的的功能。

63.2.1 FMK 密钥

FMK 密钥是个 256 位的密钥。FMK 由密钥管理器从 OTP 传输而来。每一颗芯片具备唯一的 OTP 根密钥。

用户使用 FMK 可以选择是使用具有两个加扰版本，原始密钥不可直接使用。

注意，为实现系统不同安全状态下的密钥隔离，密钥管理器支持在 SEC 和 NSC 这两个不同状态下，单独地配置密钥是否加扰，并且不同模式下使用不同的加扰算法。即相同的原始数据，加扰后的输出也不相同。请查阅系统安全管理器的相关章节获取安全状态的详细信息。

SEC_KEY_SEL[FMK_SEL] 位置 0，表示当系统安全状态为 SEC 时，使用加扰后 FMK；否则使用 FMK 的另一加扰版本。

NSC_KEY_SEL[FMK_SEL] 位置 0，表示当系统安全状态为 NSC 时，使用加扰后 FMK；否则使用 FMK 的另一加扰版本。

63.2.2 SMK 密钥

SMK 密钥是个 256 位的密钥。SMK 可以由软件写入 SFK0 SFK7 寄存器。

SMK 支持 SMKRNG 端口直接载入，通过配置 KEYM_RNG 寄存器可以配置载入选项：

- RNG[RNG_XOR] = 0’b0，SMK(KEYM_SFKx) 直接填入 RNG 生成的随机数
- RNG[RNG_XOR] = 0’b1，SMK(KEYM_SFKx) 更新为原值与 RNG 生成的随机数按位异或

用户将 KEYM_RNG[BLOCK_RNG_XOR] 位置 1，可以锁定 KEYM_RNG[RNG_XOR] 位的值，不能再更改。

密钥管理器支持对存放 SMK 密钥的寄存器 SFK0 SFK7 设置读保护，将 READ_CONTROL [BLOCK_SMK_READ] 位置 1，则软件不能再读取 SFK0 SFK7。

用户使用 SMK 可以选择是使用原始数据，或者加扰。

注意，为实现系统不同安全状态下的密钥隔离，密钥管理器支持在 SEC 和 NSC 这两个不同状态下，单独地配置密钥是否加扰，并且不同模式下使用不同的加扰算法。即相同的原始数据，加扰后的输出也不相同。请查阅系统安全管理器的相关章节获取安全状态的详细信息。

SEC_KEY_SEL[SMK_SEL] 位置 1，表示当系统安全状态为 SEC 时，使用加扰后 SMK；否则使用 SMK 的原始数据。

NSC_KEY_SEL[SMK_SEL] 位置 1，表示当系统安全状态为 NSC 时，使用加扰后 SMK；否则使用 SMK 的原始数据，若安全模式设置为使用加扰版本，则非安全模式固定为加扰版本。

63.2.3 MK 密钥

MK 可以从 FMK，SMK 中选择其一，也可以由它们中的多个或者全部合成。通过密钥合成，可以把根密钥分散管理，降低风险。

注意，为实现系统不同安全状态下的密钥隔离，密钥管理器支持在 SEC 和 NSC 这两个不同状态下，单独地配置 MK 合成。

MK 合成的具体配置方法为：

- SEC_KEY_SEL 寄存器的 KEY_SEL[0] 置 1，表示当系统安全状态为 SEC 时，FMK 参与密钥合成，否则输出密钥与 FMK 无关
- SEC_KEY_SEL 寄存器的 KEY_SEL[2] 置 1，表示当系统安全状态为 SEC 时，SMK 参与密钥合成，否则输出密钥与 SMK 无关
- NSC_KEY_SEL 寄存器的 KEY_SEL[0] 置 1，表示当系统安全状态为 NSC 时，FMK 参与密钥合成，否则输出密钥与 FMK 无关
- NSC_KEY_SEL 寄存器的 KEY_SEL[2] 置 1，表示当系统安全状态为 NSC 时，SMK 参与密钥合成，否则输出密钥与 SMK 无关

假设用户将 SEC_KEY_SEL 寄存器的 KEY_SEL[0] 置 1，而 KEY_SEL[2] 置 0，那么在安全状态为 SEC 时，密钥管理器会将 FMK 作为 MK 输出。

若两个合成因子都为 0，或选择的因子有任意一个输入无效值，则输出全零密钥。可以通过将加密数据与全零密钥的密文相比对的方式检测密钥是否设置成功。

63.2.4 SK 密钥

SK 密钥又称为 session key，密钥管理器支持输出 SK0，SK1，SK2，SK3 到系统上的加解密引擎。

注意，为实现系统不同安全状态下的密钥隔离，密钥管理器支持在 SEC 和 NSC 这两个不同状态下，生成不同的 SK。

SEC_KEY_SEL[SK_VAL] 位置 1，表示当系统安全状态为 SEC 时，SK 生成有效；否则使用 SKx 的输出为全 0。

NSC_KEY_SEL[SK_VAL] 位置 1，表示当系统安全状态为 NSC 时，SK 生成有效；否则使用 SKx 的输出为全 0。

密钥管理器的 SK 密钥生成通过密钥生成模块实现。密钥生成模块的输入是 SMK，而每一颗芯片具有不同的密钥产生逻辑，不同的 SKx 之间，密钥生成模块逻辑也不相同。

因此，在同一芯片上，当 SMK = a 时，生成 SK0 = SK0a，SK1 = SK1a，SK2 = SK2a，SK3 = SK3a。只要记录下 a 的值，再次设置 SMK = a，可以还原 SK0a，SK1a，SK2a，SK3a。但是在另一芯片上，输入 SMK = a 则不能得到相同的 SK。

注意：如使用 session Key，则不应在 MK 中引入 SMK 作为合成因子。

63.3 KEYM 寄存器列表

KEYM 的寄存器列表如下：

KEYM base address: 0xF3054000

地址偏移	名称	描述	复位值
0x0000	SOFTMKEY[SFK0]	软密钥	0x00000000
0x0004	SOFTMKEY[SFK1]	软密钥	0x00000000
0x0008	SOFTMKEY[SFK2]	软密钥	0x00000000
0x000C	SOFTMKEY[SFK3]	软密钥	0x00000000
0x0010	SOFTMKEY[SFK4]	软密钥	0x00000000
0x0014	SOFTMKEY[SFK5]	软密钥	0x00000000
0x0018	SOFTMKEY[SFK6]	软密钥	0x00000000
0x001C	SOFTMKEY[SFK7]	软密钥	0x00000000
0x0020	SOFTPKEY[SPK0]	私钥	0x00000000
0x0024	SOFTPKEY[SPK1]	私钥	0x00000000
0x0028	SOFTPKEY[SPK2]	私钥	0x00000000
0x002C	SOFTPKEY[SPK3]	私钥	0x00000000
0x0030	SOFTPKEY[SPK4]	私钥	0x00000000
0x0034	SOFTPKEY[SPK5]	私钥	0x00000000
0x0038	SOFTPKEY[SPK6]	私钥	0x00000000
0x003C	SOFTPKEY[SPK7]	私钥	0x00000000
0x0040	SEC_KEY_CTL	安全密钥产生	0x00000000
0x0044	NSC_KEY_CTL	非安全密钥产生	0x00000000
0x0048	RNG	随机数发生器接口	0x00000000
0x004C	READ_CONTROL	密钥读取控制	0x00000000

表 232: KEYM 寄存器列表

63.4 寄存器描述

KEYM 的寄存器详细说明如下：

63.4.1 SOFTMKEY (0x0 + 0x4 * n)

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

SOFTMKEY [31:0]

位域	名称	描述
31-0	KEY	密钥有 4 种加扰版本，加扰方式在同一颗芯片上可重复，不同芯片上不可重现。该组寄存器必须按从 0-7 的顺序写入，否则输出全 0.

SOFTMKEY 位域

63.4.2 SOFTPKEY (0x20 + 0x4 * n)

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

SOFTPKEY [31:0]

位域	名称	描述
31-0	KEY	密钥由熔丝私钥，软件密钥，SRK 及系统安全状态派生出来。此密钥只可读取一次，再次读取则得到全 0。

SOFTPKEY 位域

63.4.3 SEC_KEY_CTL (0x40)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LOCK_SEC_CTL	RSVD														SK_VAL	RSVD			SMK_SEL	RSVD			ZMK_SEL	RSVD			FMK_SEL	RSVD	KEY_SEL		
RW	N/A														RO	N/A			RW	N/A			RW	N/A			RW	N/A	RW		
0	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	x	x	x	0	x	x	x	0	x	x	x	0	x	0	0	0

SEC_KEY_CTL [31:0]

位域	名称	描述
31	LOCK_SEC_CTL	禁止更改密钥设置
16	SK_VAL	会话密钥有效 0: 会话密钥未设置或不可用 1: 会话密钥有效
12	SMK_SEL	软密钥选择 0: 使用原始密钥 1: 使用加扰版本

位域	名称	描述
8	ZMK_SEL	电池域密钥选择（如有电池域） 0: 使用加扰版本 1: 使用原始密钥
4	FMK_SEL	熔丝对称密钥选择 0: 使用加扰版本 1: 使用另一加扰版本
2-0	KEY_SEL	安全状态对称密钥合成选择 bit0: 熔丝密钥, 0: 不参与, 1: 参与 bit1: 电池密钥, 0: 不参与, 1: 参与（如有电池域） bit2: 软密钥 key 0: 不参与, 1: 参与

SEC_KEY_CTL 位域

63.4.4 NSC_KEY_CTL (0x44)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LOCK_NSC_CTL	RSVD														SK_VAL	RSVD			SMK_SEL	RSVD			ZMK_SEL	RSVD			FMK_SEL	RSVD	KEY_SEL		
RW	N/A														RO	N/A			RW	N/A			RW	N/A			RW	N/A	RW		
0	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	x	x	x	0	x	x	x	0	x	x	x	0	x	0	0	0

NSC_KEY_CTL [31:0]

位域	名称	描述
31	LOCK_NSC_CTL	禁止更改密钥设置
16	SK_VAL	会话密钥有效 0: 会话密钥未设置或不可用 1: 会话密钥有效
12	SMK_SEL	软密钥选择 0: 使用原始密钥 1: 使用加扰版本
8	ZMK_SEL	电池域密钥选择（如有电池域） 0: 使用加扰版本 1: 使用原始密钥
4	FMK_SEL	熔丝对称密钥选择 0: 使用加扰版本 1: 使用另一加扰版本

位域	名称	描述
2-0	KEY_SEL	安全状态对称密钥合成选择 bit0: 熔丝密钥, 0: 不参与, 1: 参与 bit1: 电池密钥, 0: 不参与, 1: 参与（如有电池域） bit2: 软密钥 key 0: 不参与, 1: 参与

NSC_KEY_CTL 位域

63.4.5 RNG (0x48)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD															BLOCK_RNG_XOR	RSVD															RNG_XOR
N/A															RW	N/A															RW
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0

RNG [31:0]

位域	名称	描述
16	BLOCK_RNG_XOR	禁止修改 RNG_XOR 位，一旦置 1 不可清零，直到下次系统复位 0: RNG_XOR 可修改 1: RNG_XOR 锁定
0	RNG_XOR	选择如何接收随机数 0: 替换软密钥 1: 和原有密钥异或

RNG 位域

63.4.6 READ_CONTROL (0x4C)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD															BLOCK_PK_READ	RSVD															BLOCK_SMK_READ
N/A															RW	N/A															RW
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0

READ_CONTROL [31:0]

位域	名称	描述
16	BLOCK_PK_READ	禁止读取私钥，一旦置 1 不可清零，直到下次系统复位 0: 私钥可读取 1: 私钥不可读取
0	BLOCK_SMK_READ	禁止读取软密钥，一旦置 1 不可清零，直到下次系统复位 0: 密钥可读取 1: 密钥不可读取

READ_CONTROL 位域

64 安全管理器 PSEC

本章节描述安全管理器 PSEC 的特性和功能。

64.1 特性总结

安全管理器 PSEC 的主要特性：

- 管理芯片的安全生命周期
- 管理芯片安全状态
- 检测芯片安全事件
- 发生异常时，采取措施保护敏感信息

64.2 功能描述

本章节描述安全管理器 PSEC 的功能。

64.2.1 芯片生命周期

根据芯片生命周期的不同阶段，对芯片内部信息的读写进行管理。安全生命周期 LIFECYCLE 有 4 位，在 OTP 中冗余设置，分为 A、B 两份设置。

芯片生命周期的 4 位，描述如下：

- LIFECYCLE = 4’b0000，CREATE，保留。
- LIFECYCLE = 4’b0001，NONSEC，芯片经过测试并向用户交付的状态，用户在此状态进行开发。
  - 芯片调试端口，JTAG 和 Debug 端口开放。
  - 测试功能开放。
  - BOOT ROM 的不执行安全启动，不验证用户代码镜像的签名。
  - 密钥管理器及其关联的几个模块使用 NONSEC 密钥。
- LIFECYCLE = 4’b0011，SECURE，用户将敏感信息和代码注入系统，并对系统进行保护，防止功能遭到破坏。
  - 芯片调试端口，JTAG 和 Debug 端口关闭，调试鉴权成功后开放调试。
  - 测试功能禁止。
  - BOOT ROM 执行行安全启动，验证镜像的签名。如果验证失败，则不会运行镜像。
  - 密钥管理器及其关联的几个模块使用 SEC 密钥。
- LIFECYCLE = 4’b0111，RETURN。该状态在保证用户信息不泄露的前提下重新允许测试功能。
  - 芯片调试端口，JTAG 和 Debug 端口重新打开。
  - 测试功能开放。
  - BOOT ROM 不执行安全启动，不验证用户代码镜像的签名。
  - 不再从 OTP 载入密钥。
- LIFECYCLE = 4’b1011，NONRET。禁止返厂测试，此状态的安全保护功能同安全状态，但是永久性禁止测试功能。
  - 在 SECURE 生命周期的基础上，芯片不再允许配置位返厂模式。
- LIFECYCLE = 4’b1111，SCRIBE。芯片废弃，销毁敏感信息，芯片在此状态下无法正常使用。
  - 密钥管理器及其关联的几个模块不能再从 OTP 载入密钥，电池域的密钥也会擦除。
  - ROM 不再引导系统。

注意：如果用户通过烧录 OTP 的 DEBUG_DISABLE，JTAG_DISABLE 位来强制关闭调试端口，JTAG 端口。那么无论芯片处于哪个生命周期，都无法再重新打开这些端口。

除了读取 OTP，用户可以从系统安全监视器的 LIFECYCLE 寄存器读取芯片的当前生命周期状态：

- LIFECYCLE[0]，标志位置 1 表示未知生命周期。
- LIFECYCLE[1]，标志位置 1 表示生命周期为 CREATE。保留。
- LIFECYCLE[2]，标志位置 1 表示生命周期为 NONSEC。芯片开放。
- LIFECYCLE[3]，标志位置 1 表示生命周期为 SECURE。芯片安全。
- LIFECYCLE[4]，标志位置 1 表示生命周期为 RETURN。芯片返厂。
- LIFECYCLE[5]，标志位置 1 表示生命周期为 NONRET。芯片禁止返厂。
- LIFECYCLE[6]，标志位置 1 表示生命周期为 SCRIBE。芯片废弃。
- LIFECYCLE[7]，标志位置 1 表示生命周期为 DEBATE。冗余状态发生矛盾。

64.2.2 安全状态管理

电源管理域安全管理器可以配置并监测芯片的安全状态。支持以下安全状态：

- INSPECT,用以对系统安全相关组件自检,该状态下密钥固定为 1F1E1D1C1B1A19181716151413121110 0F0E0D0C
- NONSEC，此状态主要保护 HPM 已安装密钥和识别号等安全基础安全信息。
- SECURE，此状态主要保护客户安装的密钥等资产。
- FAIL，该状态会禁止硬件加解密引擎。

当芯片从复位中释放时，安全状态最先处于 INSPECT 状态，BOOT ROM 会根据 OTP 中芯片的 LIFECYCLE 位设置，配置芯片为 NONSEC 或者 SECURE 状态。

- 如果当前芯片的生命周期为 NONSEC，那么设置安全状态为 NONSEC；
- 如果当前芯片的生命周期为 SEUCRE，那么设置安全状态为 SECURE；
- 如果当前芯片的生命周期为 RETURN，那么设置安全状态为 NONSEC；
- 如果当前芯片的生命周期为 NONRET，那么设置安全状态为 SECURE；
- 如果当前芯片的生命周期为其他状态，那么设置安全状态为 FAIL；

片上的密钥模块，如密钥管理器，会根据芯片安全状态，提供 NONSEC 或者 SECURE 两套密钥。而在其他的安全状态下，密钥管理器会封锁密钥。

用户可以通过 SECURE_STATE 寄存器查询及配置安全状态。

安全管理器监视系统上若干安全相关的事件，有关具体事件列表，请查阅系统安全配置的相关章节。

注意，安全管理器在 SECURE 状态和 NONSEC 状态下都可以监视并响应安全事件，并在安全事件发生时，将安全状态置为 FAIL。

用户可以通过 VIOLATION_CONFIG 寄存器配置监视器是否响应检测的安全事件：

- NSC_VIO_CFG 位域包含各个安全事件的响应使能位，置 0 表示对应的安全事件在 NONSEC 状态下，不视为破坏安全规则，置 1 表示对应的安全事件在 NONSEC 状态下，视为破坏安全规则。安全状态会因之转为 FAIL。
- LOCK_NSC 位为 NONSEC 配置锁定位，置 1 即锁定 NSC_VIO_CFG 位域
- SEC_VIO_CFG 位域包含各个安全事件的响应使能位，置 0 表示对应的安全事件在 SECURE 状态下，不视为破坏安全规则，置 1 表示对应的安全事件在 SECURE 状态下，视为破坏安全规则。安全状态会因之转为 FAIL。
- LOCK_SEC 位为 SECURE 配置锁定位，置 1 即锁定 SEC_VIO_CFG 位域

用户可以通过 EVENT 寄存器的各个标志位，查询是否检测到安全事件。

当安全状态为 FAIL 时，PSEC 会禁止敏感信息的访问，并且禁止加解密引擎。

64.2.3 安全事件通报

可以通过 ESCALATE_CONFIG 寄存器配置安全监视器是否向电池域（如有电池域）安全管理器通报安全事件，允许电池域内部的安全管理器作出相应保护。

- NSC_VIO_CFG 位域包含各个安全事件的通报使能，置 1 时，电池域处于 NONSEC 模式下，响应该安全事件。
- SEC_VIO_CFG 位域包含各个安全事件的通报使能，置 1 时，电池域处于 SECURE 模式下，响应该安全事件。

用户可以通过 EVENT 寄存器的各个标志位，查询是否发生安全事件通报。

64.3 PSEC 寄存器列表

PSEC 的寄存器列表如下：

SEC base address: 0xF3044000

地址偏移	名称	描述	复位值
0x0000	SECURE_STATE	安全状态	0x00000000
0x0004	SECURE_STATE_CONFIG	安全状态机配置	0x00000000
0x0008	VIOLATION_CONFIG	安全违例配置	0x00000000
0x000C	ESCALATE_CONFIG	安全事件通报配置	0x00000000
0x0010	EVENT	事件通报状态	0x00000000
0x0014	LIFECYCLE	生命周期	0x00000000

表 233: PSEC 寄存器列表

64.4 PSEC 寄存器描述

PSEC 的寄存器详细说明如下：

64.4.1 SECURE_STATE (0x0)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD														ALLOW_NSC	ALLOW_SEC	RSVD								PMIC_FAIL	PMIC_NSC	PMIC_SEC	PMIC_INS	RSVD			
N/A														RO	RO	N/A								RW	RW	RW	RW	N/A			
x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	x	x	x	x	x	x	x	x	0	0	0	0	x	x	x	x

SECURE_STATE [31:0]

位域	名称	描述
17	ALLOW_NSC	允许非安全状态 0: 系统安全状态不可进入或停留在非安全状态，此时从 INSPECT 跳转到非安全状态的请求会使状态机进入失败状态 1: 系统安全状态允许进入或停留在非安全状态
16	ALLOW_SEC	允许非安全状态 0: 系统安全状态不可进入或停留在安全状态，此时从 INSPECT 跳转到安全状态的请求会使状态机进入失败状态 1: 系统安全状态允许进入或停留在安全状态
7	PMIC_FAIL	状态机状态，写 1 请求进入失效状态 0: 状态机不处于失败状态 1: 状态机处于失败状态
6	PMIC_NSC	状态机状态 0: 状态机不处于非安全状态 1: 状态机处于非安全状态
5	PMIC_SEC	状态机状态 0: 状态机不处于安全状态 1: 状态机处于安全状态
4	PMIC_INS	状态机状态 0: 状态机不处于检查状态 1: 状态机处于检查状态

SECURE_STATE 位域

64.4.2 SECURE_STATE_CONFIG (0x4)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																												LOCK	RSVD		ALLOW_RESTART
N/A																												RW	N/A		RW
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	x	x	0

SECURE_STATE_CONFIG [31:0]

位域	名称	描述
3	LOCK	状态机允许恢复锁定，锁定后锁定位会一同锁定，直至下一次复位 0: 未锁定，允许配置修改 1: 锁定，配置不可更改

位域	名称	描述
0	ALLOW_RESTART	允许状态机从 FAIL 回到 INSPECT 0: 不允许，只可以通过复位恢复状态机 1: 允许，软件可以把状态机从 FAIL 切换至 INSPECT

SECURE_STATE_CONFIG 位域

64.4.3 VIOLATION_CONFIG (0x8)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LOCK_NSC	NSC_VIO_CFG															LOCK_SEC	SEC_VIO_CFG														
RW	RW															RW	RW														
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

VIOLATION_CONFIG [31:0]

位域	名称	描述
31	LOCK_NSC	事件非安全状态违例配置锁定，锁定后锁定位会一同锁定，直至下一次复位 0: 未锁定，允许配置修改 1: 锁定，配置不可更改
30-16	NSC_VIO_CFG	配置事件是否作为非安全状态事件向电池域违例，每一位代表一个事件 0: 事件不向电池域违例 1: 事件向电池域违例
15	LOCK_SEC	事件安全状态违例配置锁定，锁定后锁定位会一同锁定，直至下一次复位 0: 未锁定，允许配置修改 1: 锁定，配置不可更改
14-0	SEC_VIO_CFG	配置事件是否作为安全状态事件向电池域违例，每一位代表一个事件 0: 事件不向电池域违例 1: 事件向电池域违例

VIOLATION_CONFIG 位域

64.4.4 ESCALATE_CONFIG (0xC)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LOCK_NSC	NSC_VIO_CFG															LOCK_SEC	SEC_VIO_CFG														
RW	RW															RW	RW														
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ESCALATE_CONFIG [31:0]

位域	名称	描述
31	LOCK_NSC	事件非安全状态通报配置锁定，锁定后锁定位会一同锁定，直至下一次复位 0: 未锁定，允许配置修改 1: 锁定，配置不可更改
30-16	NSC_VIO_CFG	配置事件是否作为非安全状态事件向电池域通报，每一位代表一个事件 0: 事件不向电池域通报 1: 事件向电池域通报
15	LOCK_SEC	事件安全状态通报配置锁定，锁定后锁定位会一同锁定，直至下一次复位 0: 未锁定，允许配置修改 1: 锁定，配置不可更改
14-0	SEC_VIO_CFG	配置事件是否作为安全状态事件向电池域通报，每一位代表一个事件 0: 事件不向电池域通报 1: 事件向电池域通报

ESCALATE_CONFIG 位域

64.4.5 EVENT (0x10)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EVENT																RSVD												PMIC_ESC_NSC	PMIC_ESC_SEC	RSVD	
RO																N/A												RO	RO	N/A	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	x	x	x	x	x	x	x	x	x	x	x	x	0	0	x	x

EVENT [31:0]

位域	名称	描述
31-16	EVENT	事件状态，每一位代表一个事件，1 表示发生该事件 9 位：TCK 出现翻转 8 位：JTAG 出现操作 7 位：CPU1 正在进入或处于调试状态 6 位：CPU0 正在进入或处于调试状态 5 位：电源域看门狗复位 4 位：电源域电压过低 3 位：毛刺监测 1 检出毛刺 2 位：毛刺监测 0 检出毛刺 1 位：时钟监测 1 检出异常 0 位：时钟监测 0 检出异常
3	PMIC_ESC_NSC	非安全状态事件通报 0：无事件通报 1：事件通报给电池域
2	PMIC_ESC_SEC	安全状态事件通报 0：无事件通报 1：事件通报给电池域

EVENT 位域

64.4.6 LIFECYCLE (0x14)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																								LIFECYCLE							
N/A																								RO							
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0

LIFECYCLE [31:0]

位域	名称	描述
7-0	LIFECYCLE	生命周期 7 位: 矛盾, 6 位: 销毁, 5 位: 禁止返厂, 4 位: 返厂, 3 位: 安全, 2 位: 非安全, 1 位: 生产, 0 位: 未知

LIFECYCLE 位域

65 安全监视器 PMON

本章节描述了安全监视器 PMON 的主要特性和功能。

65.1 特性总结

安全监视器 PMON 是安全管理器的一部分，它可以用来配置供电和时钟监视电路，它的主要特性：

- 2 个毛刺检测电路
- 2 个时钟故障检测电路
- 支持产生中断

65.2 功能描述

本章节描述安全监视器 PMON 的功能。

65.2.1 芯片生命周期

安全监视器包含电源和时钟故障检测电路，检测电路的作用是监测电源和时钟上意外注入的毛刺，并通知安全管理器。

安全监视器包含 2 个独立的电源毛刺检测模块和 2 个独立的时钟故障检测模块。

- 毛刺检测模块检测 VPMC 引脚上的电源输入以及 OSC24M 的时钟噪声
- 时钟故障检测模块检测外部时钟 OSC24M 或内部 RC24M 振荡器

MONITOR.GLITCH0 和 MONITOR.GLITCH1 分别对应 2 个电源毛刺检测模块，通过各自的 CONTROL 寄存器进行配置：

- CONTROL[ENABLE] 位置 0 时，检测模块关闭，置 1 时检测模块进行监测。
- CONTROL[ACTIVE] 位置 0 时，设置为被动检测，置 1 时，设置为主动检测

MONITOR.CLOKC0 和 MONITOR.CLOKC1 分别对应 2 个时钟故障检测模块，通过各自的 CONTROL 寄存器进行配置：

- CONTROL[ENABLE] 位置 0 时，检测模块关闭，置 1 时检测模块进行监测。

各个检测模块可以独立地检测电源毛刺或者时钟故障，并在发现异常时，作为安全事件通知安全管理器 PSEC，PSEC 会根据配置予以响应。

65.2.2 中断

安全监视器也可以配置在检测到电源毛刺或者时钟故障时生成中断请求。

IRQ_ENABLE 寄存器包含了相应的中断使能位。

IRQ_FLAG 寄存器可以查询中断的标志位。

65.3 PMON 寄存器列表

PMON 的寄存器列表如下：

SEC_MON base address: 0xF3048000

地址偏移	名称	描述	复位值
0x0000	MONITOR[GLITCH0][CONTROL]	时钟和电源毛刺检测器 0 设置	0x00000000
0x0004	MONITOR[GLITCH0][STATUS]	时钟和电源毛刺检测器 0 状态	0x00000000
0x0008	MONITOR[GLITCH1][CONTROL]	时钟和电源毛刺检测器 1 设置	0x00000000
0x000C	MONITOR[GLITCH1][STATUS]	时钟和电源毛刺检测器 1 状态	0x00000000
0x0010	MONITOR[CLOCK0][CONTROL]	时钟停止检测器 0 设置	0x00000000
0x0014	MONITOR[CLOCK0][STATUS]	时钟停止检测器 0 状态	0x00000000
0x0018	MONITOR[CLOCK1][CONTROL]	时钟停止检测器 1 设置	0x00000000
0x001C	MONITOR[CLOCK1][STATUS]	时钟停止检测器 1 状态	0x00000000
0x0040	IRQ_FLAG		0x00000000
0x0044	IRQ_ENABLE		0x00000000

表 234: PMON 寄存器列表

65.4 PMON 寄存器描述

PMON 的寄存器详细说明如下：

65.4.1 MONITOR[CONTROL] (0x0 + 0x8 * n)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																											ACTIVE	RSVD				ENABLE
N/A																											RW	N/A				RW
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	x	x	x	0

MONITOR[CONTROL] [31:0]

位域	名称	描述
4	ACTIVE	检测器工作模式选择. 0: 被动模式 1: 主动模式
0	ENABLE	使能检测器 0: 检测器关闭 1: 检测器运行

MONITOR[CONTROL] 位域

65.4.2 MONITOR[STATUS] (0x4 + 0x8 * n)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																															FLAG
N/A																															RW

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

MONITOR[STATUS] [31:0]

位域	名称	描述
0	FLAG	检测器状态 0: 正常 1: 异常

MONITOR[STATUS] 位域

65.4.3 IRQ_FLAG (0x40)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																												FLAG			
N/A																												RW			
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0

IRQ_FLAG [31:0]

位域	名称	描述
3-0	FLAG	中断标志，写 1 清 0: 为发生中断 1: 发生中断

IRQ_FLAG 位域

65.4.4 IRQ_ENABLE (0x44)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																												ENABLE			
N/A																												RW			
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0

IRQ_ENABLE [31:0]

位域	名称	描述
3-0	ENABLE	中断使能，每一位控制一个检测器 0: 禁止中断 1: 使能中断

IRQ_ENABLE 位域

66 系统调试概述

本章节介绍了本产品的调式模块。

Debug 调试系统符合 The RISC-V Debug Specification, Version 0.13 规范。Debug 调试系统包括 JTAG 接口转换模块（DTM）和调试模块（DM）2 部分。DTM 通过标准 JTAG 接口对接外部调试器，可以把 JTAG 上收到的调试指令转换成对 DM 模块的读写访问。调试模块 DM 集成了调试功能，可以暂停或者恢复 CPU 的运行，产生复位，以及访问片上资源。

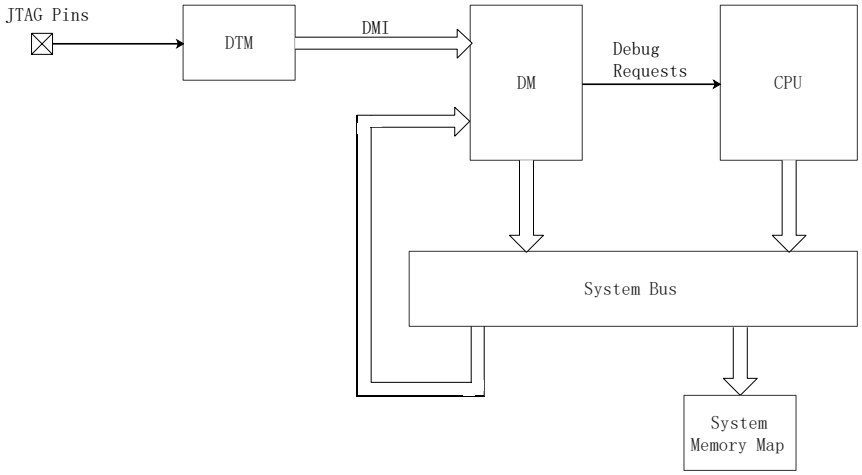


图 82: 调试系统框图

Debug 系统特性如下：

- 支持 4 线 JTAG 接口，符合 IEEE Std 1149.1
- 支持片外调试器通过 JTAG 接口转换模块连接 Debug 调试模块
- 支持处理器通过系统地址映射访问 Debug 调试模块
- 支持 2 个硬件断点
- 支持单步调试
- 支持 System Bus Access，调试接口可以作为总线主设备，可以不打断内核运行直接访问片上资源
- 支持 Debug 安全机制，可以开放，关闭或者锁定 Debug 调试接口，当 Debug 被锁定时，用户需要通过安全码解锁。

本产品的 JTAG ID 为 0x1000563d。

67 调试传输模块 DTM

调试传输模块 (DTM)

该模块符合 RISCV0.13 版的定义，支持 IEEE1149.1 所定义的 JTAG 接口。指令长度为 5 位。

67.1 DTM 指令

INSTRUCTION 列表如下：

值	名称	描述	数据宽度
0x00	USERCODE	芯片识别号	32
0x01	IDCODE	芯片识别号	32
0x08	CLAMP	CLAMP 指令	404
0x09	EXTEST	EXTEST 指令	404
0x0A	SAMPLE/PRELOAD	SAMPLE/PRELOAD 指令	404
0x0B	HIGHZ	HIGHZ 指令	1
0x0F	AUTHEN	调试认证指令	32
0x10	DTMCS	DTM 控制寄存器	32
0x11	DMI	DMI 调试接口	41
0x1F	BYPASS	BYPASS 指令	1

表 235: DTM 指令列表

67.2 DTM 指令描述

INSTRUCTION 的详细说明如下：

67.2.1 USERCODE (0x0)

位域	名称	访问	描述
31-0	USERCODE	RO	芯片型号识别号

USERCODE 位域

67.2.2 IDCODE (0x1)

位域	名称	访问	描述
31-0	IDCODE	RO	用于调试系统识别号

IDCODE 位域

67.2.3 CLAMP (0x8)

位域	名称	访问	描述
403-0	BOUNDRY	RO	参阅 IEEE1149.1

位域	名称	访问	描述

CLAMP 位域

67.2.4 EXTEST (0x9)

位域	名称	访问	描述
403-0	BOUNDRY	RO	参阅 IEEE1149.1

EXTEST 位域

67.2.5 SAMPLE/PRELOAD (0xA)

位域	名称	访问	描述
403-0	BOUNDRY	RO	参阅 IEEE1149.1

SAMPLE/PRELOAD 位域

67.2.6 HIGHZ (0xB)

位域	名称	访问	描述
0	BYPASS	RO	参阅 IEEE1149.1

HIGHZ 位域

67.2.7 AUTHEN (0xF)

位域	名称	访问	描述
31-0	AUTH	RW	调试认证指令，调试认证需要进行 8 次数据交换。前 4 次将 128 位管芯跟踪数据传输到芯片外面，后四次输入 128 位调试密钥。

AUTHEN 位域

67.2.8 DTMCS (0x10)

位域	名称	访问	描述
17	DMIHARDRESET	W1	写 1 会复位 DTM，清除之前的访问状态。通常，只有在当前访问不可能完成的情况下（例如：访问过程中发生了复位）才使用这一位。
16	DMIRESET	W1	写 1 清除错误标志，使 DTM 重试或结束一个访问。
14-12	IDLE	RO	为调试器提供信息，通知调试器 DMI 扫描后在 RunTest/Idle 中需停留的周期数，以减少繁忙的的情况。

位域	名称	访问	描述
11-10	DMISTAT	RO	DMI 接口状态 0: 无错误 1: 保留 2: 操作数错误 3: DMI 忙时发出下一个访问
9-4	ABITS	RO	DMI 接口地址线宽度，读出值为 7
3-0	VERSION	RO	DTM 版本号。当前读出值为 0x1，表示符合 RISC-V External Debug Support (TD003) V0.13.

DTMCS 位域

67.2.9 DMI (0x11)

位域	名称	访问	描述
40-34	ADDRESS	RW	DMI 的访问地址。
33-2	DATA	RW	通过 DMI 写入或从 DMI 读取的数据。
1-0	OP	RW	写入： 0：忽略数据和地址。（无操作） 1：从地址读取。（读） 2：将数据写入地址。（写） 3：保留 读取： 0：上一操作成功完成。 1：保留 2：上一操作失败。 3：DMI 请求发生重叠。

DMI 位域

67.2.10 BYPASS (0x1F)

位域	名称	访问	描述
0	BYPASS	RO	参阅 IEEE1149.1

BYPASS 位域

68 调试模块 DM

该模块符合 RISCV0.13 版的定义，具有 16x32bit 的程序缓冲器 (program buffer)。

调试模块内的寄存器有两种访问方式，CPU 通过系统总线访问或调试器通过 DTM 访问。两种访问方式具有不同的内存映射。

68.1 系统内存映射

地址偏移	名称	描述
0x0000 - 0x007F	DEBUGROM	Debug ROM
0x0080 - 0x00BF	PROGBUF	Program Buffer 0-15
0x00C0 - 0x00CF	DATA	Abstract Data 0-3
0x00D0 - 0x01FF	RESERVED	Reserved for internal use

表 236: DM 系统内存映射

68.2 DMI 内存映射

地址偏移	名称	描述	复位值
0x0004	DATA[DATA0]	抽象数据 0	0x00000000
0x0005	DATA[DATA1]	抽象数据 1	0x00000000
0x0006	DATA[DATA2]	抽象数据 2	0x00000000
0x0007	DATA[DATA3]	抽象数据 3	0x00000000
0x0010	DMCONTROL	调试模块控制	0x00000000
0x0011	DMSTATUS	调试模块状态	0x004000A2
0x0012	HARTINFO	内核信息	0x002140C0
0x0013	HALTSUM	Halt Summary	0x00000000
0x0014	HAWINDOWSEL	内核阵列窗口选择	0x00000000
0x0015	HAWINDOW	内核阵列窗口	0x00000000
0x0016	ABSTRACTCS	抽象控制和状态	0x08000004
0x0017	COMMAND	抽象命令	0x00000000
0x0018	ABSTRACTAUTO	抽象命令自动执行	0x00000000
0x0020	PROGBUF[PROGBUF0]	程序缓冲 0	0x00000000
0x0021	PROGBUF[PROGBUF1]	程序缓冲 1	0x00000000
0x0022	PROGBUF[PROGBUF2]	程序缓冲 2	0x00000000
0x0023	PROGBUF[PROGBUF3]	程序缓冲 3	0x00000000
0x0024	PROGBUF[PROGBUF4]	程序缓冲 4	0x00000000
0x0025	PROGBUF[PROGBUF5]	程序缓冲 5	0x00000000
0x0026	PROGBUF[PROGBUF6]	程序缓冲 6	0x00000000
0x0027	PROGBUF[PROGBUF7]	程序缓冲 7	0x00000000
0x0038	SBCS	系统总线访问控制和状态	0x20040407
0x0039	SBADDRESS[SBADDRESS0]	系统总线地址 0	0x00000000
0x003A	SBADDRESS[SBADDRESS1]	系统总线地址 1	0x00000000

地址偏移	名称	描述	复位值
0x003B	SBADDRESS[SBADDRESS2]	系统总线地址 2	0x00000000
0x003C	SBDATA[SBDATA0]	系统总线数据 0	0x00000000
0x003D	SBDATA[SBDATA1]	系统总线数据 1	0x00000000
0x003E	SBDATA[SBDATA2]	系统总线数据 2	0x00000000
0x003F	SBDATA[SBDATA3]	系统总线数据 3	0x00000000

表 237: DM 寄存器列表

68.3 DM 寄存器描述

68.3.1 DATA (0x4 + 0x1 * n)

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

DATA [31:0]

位域	名称	描述
31-0	DATA	从 DMI 接口和系统总线均可访问，用于外部调试器和处理器之间的数据交换。

DATA 位域

68.3.2 DMCONTROL (0x10)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
HALTREQ	RESUMEREQ	RSVD	ACKHAVERESET	RSVD	HASEL	HARTSEL										RSVD												SETRESETHALTREQ	CLRRESETHALTREQ	NDMRESET	DMACTIVE
WO	WO	N/A	RW	N/A	RW	RW										N/A												W1S	W1C	RW	RW
0	0	x	0	x	0	0	0	0	0	0	0	0	0	0	0	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0

DMCONTROL [31:0]

位域	名称	描述
31	HALTREQ	请求选中核心进入调试模式。写 1 时，若选中核心正在运行则进入调试模式，已处于调试模式的内核不受影响。若请求恢复位为 0，则写 0 对处于调试模式的内核没有影响。根据正在写入的 hartsel 和 hasel 的新值选择 Harts。
30	RESUMEREQ	选中核心恢复运行。写 1 时，被选中的核心若处于调试状态则恢复运行。若同时写入调试请求，则忽略恢复请求位。若同时写入 hartsel 和 hasel，则使用新的选择。
28	ACKHAVERESET	写 1 将选中核心的 havereset 位。若同时写入 hartsel 和 hasel，则使用新的选择。
26	HASEL	选择当前选定 HART 的定义。
25-16	HARTSEL	选择要调试的核心。
3	SETRESETHALTREQ	若 clrresethaltreq 为 0，则置位所有选中核心的复位调试请求。置位后，选中的内核在下一次复位释放时进入调试状态。该位不自动清零，需写入 clrresethaltreq 清零。若同时写入 hartsel 和 hasel，则使用新的选择。
2	CLRRESETHALTREQ	清除所有选中的核心的复位调试请求。若同时写入 hartsel 和 hasel，则使用新的选择。
1	NDMRESET	调试模块以外的系统复位控制。
0	DMACTIVE	调试模块的复位控制。

DMCONTROL 位域

68.3.3 DMSTATUS (0x11)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD									IMPEBREAK	RSVD		ALLHAVERESET	ANYHAVERESET	ALLRESUMEACK	ANYRESUMEACK	ALLNONEXISTENT	ANYNONEXISTENT	ALLUNAVAIL	ANYUNAVAIL	ALLRUNNING	ANYRUNNING	ALLHALTED	ANYHALTED	AUTHENTICATED	AUTHBUSY	HASRESETHALTREQ	DEVTREEVALID	VERSION			
N/A									RO	N/A		RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO			
x	x	x	x	x	x	x	x	x	1	x	x	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	0	0	1	0

DMSTATUS [31:0]

位域	名称	描述
22	IMPEBREAK	指示在程序缓冲区后保留字中是否有 EBREAK 指令。
19	ALLHAVERESET	指示全部选中的核心已复位单位确认。
18	ANYHAVERESET	指示有选中的核心已复位单位确认。
17	ALLRESUMEACK	指示全部选中的核心恢复运行。
16	ANYRESUMEACK	指示有选中的核心恢复运行。

位域	名称	描述
15	ALLNONEXISTENT	指示全部选中的核心不存在。
14	ANYNONEXISTENT	指示有选中的核心不存在。
13	ALLUNAVAIL	指示全部选中的核心不可用。
12	ANYUNAVAIL	指示有选中的核心不可用。
11	ALLRUNNING	指示全部选中的核心正在运行。
10	ANYRUNNING	指示有选中的核心正在运行。
9	ALLHALTED	指示全部选中的核心处于调试模式。
8	ANYHALTED	指示有选中的核心处于调试模式。
7	AUTHENTICATED	指示验证模块检查已通过。
6	AUTHBUSY	指示验证模块处于繁忙状态。
5	HASRESETHALTREQ	是否支持复位调试功能。
4	DEVTREEVALID	设备树寄存器中的信息是否保存设备树的地址。
3-0	VERSION	RISC-V 外部调试支持的版本。0x2 表示当前实现的版本为 0.13。

DMSTATUS 位域

68.3.4 HARTINFO (0x12)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD								NSCRATCH				RSVD			DATAACCESS	DATASIZE				DATAADDR												
N/A								RO				N/A			RO	RO				RO											
x	x	x	x	x	x	x	x	0	0	1	0	x	x	x	1	0	1	0	0	0	0	0	0	1	1	0	0	0	0	0	0

HARTINFO [31:0]

位域	名称	描述
23-20	NSCRATCH	核心中调试寄存器数量
16	DATAACCESS	访问数据寄存器的方法。该字段的值为 0x1，表示调试模式下数据寄存器通过在地址访问。 0：数据寄存器通过 CSR 寄存器访问。 1：数据寄存器通过地址访问。每个寄存器占 4 个字节。
15-12	DATASIZE	数据寄存器个数。
11-0	DATAADDR	总线访问时数据寄存器偏移量，有符号数。调试模式下用作 load/store 指令的偏移。

HARTINFO 位域

68.3.5 HALTSUM (0x13)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																															HALT
N/A																															RW
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0

HALTSUM [31:0]

位域	名称	描述
0	HALT	位 0 包含 hart 0-hart 31 的 32 个暂停位的逻辑 OR。

HALTSUM 位域

68.3.6 HAWINDOWSEL (0x14)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD																	HAWINDOWSEL														
N/A																	RW														
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

HAWINDOWSEL [31:0]

位域	名称	描述
14-0	HAWINDOWSEL	该寄存器选择在 hawindow 中可访问 hart 阵列掩码寄存器的 32 位部分。

HAWINDOWSEL 位域

68.3.7 HAWINDOW (0x15)

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

HAWINDOW [31:0]

位域	名称	描述
31-0	HAWINDOW	该寄存器提供对 hart 阵列掩码寄存器 32 位部分的 R/W 访问。窗户的位置由 hawindowsel 确定。也就是说，位 0 表示 hart（hawindowsel*32），而位 31 表示 hart（hawindowsel*32+31）。

HAWINDOW 位域

68.3.8 ABSTRACTCS (0x16)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD			PROGBUFSIZE					RSVD											BUSY	RSVD	CMDERR			RSVD			DATACOUNT				
N/A			RO					N/A											RO	N/A	RW1C			N/A			RO				
x	x	x	0	1	0	0	0	x	x	x	x	x	x	x	x	x	x	x	0	x	0	0	0	x	x	x	0	0	1	0	0

ABSTRACTCS [31:0]

位域	名称	描述
28-24	PROGBUFSIZE	程序缓冲区的大小，以 32 位字为单位。
12	BUSY	指示正在执行抽象命令。
10-8	CMDERR	抽象命令的错误代码。置 1 后即保持，直到写入 1 时清零。清零之前，不执行抽象命令。 0: 无：无错误 1：忙：抽象命令执行时写入了 command、abstractcs、AbstractAuto 或读写了指令缓冲或数据寄存器。 2: 不支持：不支持的抽象命令。 3: 异常：执行时发生异常。 4：停止/恢复：核心状态错误，无法执行抽象命令。
4-0	DATACOUNT	数据寄存器的数量。

ABSTRACTCS 位域

68.3.9 COMMAND (0x17)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CMDTYPE								CONTROL																							
WO								WO																							
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

COMMAND [31:0]

位域	名称	描述
31-24	CMDTYPE	抽象命令类型。
23-0	CONTROL	该字段在执行不同的抽象命令时有不同的功能。

COMMAND 位域

68.3.10 ABSTRACTAUTO (0x18)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSVD								AUTOEXECPROGBUF								RSVD												AUTOEXECDATA			
N/A								RW								N/A												RW			
x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0

ABSTRACTAUTO [31:0]

位域	名称	描述
23-16	AUTOEXECPROGBUF	当字段中的位为 1 时，对应指令缓冲读写访问将触发抽象命令的执行。
3-0	AUTOEXECDATA	当字段中的位为 1 时，对应数据字的读写访问将触发抽象命令的执行。

ABSTRACTAUTO 位域

68.3.11 PROGBUF (0x20 + 0x1 * n)

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

PROGBUF [31:0]

位域	名称	描述
31-0	DATA	程序缓冲区。程序缓冲区 0-7 可读写，8-15 的值为 0x00100073（EBREAK 指令）。

PROGBUF 位域

68.3.12 SBCS (0x38)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SBVERSION			RSVD						SBBUSYERROR	SBBUSY	SBREADONADDR	SBACCESS			SBAUTOINCREMENT	SBREADONDATA	SBERROR			SBASIZE								SBACCESS128	SBACCESS64	SBACCESS32	SBACCESS16	SBACCESS8
RO			N/A						RW1C	RO	RW	RW			RW	RW	RW1C			RO								RO	RO	RO	RO	RO
0	0	1	x	x	x	x	x	x	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	1	1

SBCS [31:0]

位域	名称	描述
31-29	SBVERSION	支持的系统总线访问版本。目前为 1，对应 RISC-V 外部调试规范 v0.13。
22	SBBUSYERROR	指示总线空闲之前发出了另一个访问命令。
21	SBBUSY	系统总线忙。
20	SBREADONADDR	写入后自动读取。
19-17	SBACCESS	访问大小。 0:8 位 1:16 位 2:32 位 3:64 位 4:128 位
16	SBAUTOINCREMENT	每次系统总线访问后，内部地址的自动增加量。
15	SBREADONDATA	从 sbdata0 读取自动触发系统总线读取（地址可递增）。
14-12	SBERROR	指示系统总线访问错误代码。写入 1 清零。 0：无错误 1：超时 2：地址不正确 3：数据对齐 4：不支持的访问大小 7：其他
11-5	SBASIZE	系统总线地址的地址宽度。
4	SBACCESS128	表示是否支持 128 位系统总线数据访问。
3	SBACCESS64	表示是否支持 64 位系统总线数据访问。
2	SBACCESS32	表示是否支持 32 位系统总线数据访问。
1	SBACCESS16	表示是否支持 16 位系统总线数据访问。
0	SBACCESS8	表示是否支持 8 位系统总线数据访问。

SBCS 位域

68.3.13 SBADDRESS (0x39 + 0x1 * n)

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

SBADDRESS [31:0]

位域	名称	描述
31-0	SBADDRESS	系统总线访问地址

SBADDRESS 位域

68.3.14 SBDATA (0x3C + 0x1 * n)

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

SBDATA [31:0]

位域	名称	描述
31-0	SBDATA	系统总线访问数据

SBDATA 位域

69 版本信息

日期	版本	描述
Rev0.0	2023/05/28	内部版 Rev 0.0 发布。
Rev0.1	2023/06/26	内部版 Rev 0.1 发布。 增加运动控制系统各模块功能描述。 更新功能综述模块描述。
Rev0.2	2023/07/28	内部版 Rev 0.2 发布。 更新产品简介描述，更新系统框图。 增加 SYNT，TRGM 章节。 更新 CPU 内核描述。 更正各模块综述章节相关功能描述。 修正各章节寄存器描述。
Rev0.3	2023/08/07	内部版 Rev 0.3 发布。 增补 QEO 章节寄存器信息。

表 238: 版本信息

70 免责声明

上海先楫半导体科技有限公司（以下简称：“先楫”）保留随时更改、更正、增强、修改先楫半导体产品和/或本文档的权利，恕不另行通知。用户可在先楫官方网站 https://www.hpmicro.com 获取最新相关信息。

本声明中的信息取代并替换先前版本中声明的信息。

表格目录

图片目录