32 位微控制器

MT32F005/006 系列技术手册

目录

1. 概述 7

2. 特性 9

3. 缩写 15

4. 方块图 16

5. 选型表和管脚图 17

5.1 MT32F005/006 选型表 17

5.2 管脚图 18

5.2.1 TSSOP-20 封装 18

5.2.2 QFN-20 封装 18

5.2.3 SOP-16 封装 19

5.2.4 QFN-28 封装 19

5.3 管脚描述 20

6. 功能描述 23

6.1 CPU 内核及总线架构 23

6.1.1 32-bit CPU 内核 23

6.1.2 系统总线架构 24

6.2 系统管理器 25

6.2.1 概述 25

6.2.2 系统复位 25

6.2.3 系统电源架构 26

6.2.4 系统时钟架构 26

6.2.5 系统功率控制 28

6.2.6 系统存储器映射以及调试模式下地址重映射 32

6.2.7 系统管理器控制寄存器列表 34

6.2.8 系统定时器（Systick） 79

6.2.9 系统定时器控制寄存器映射 79

6.2.10 嵌套向量中断控制器（NVIC） 82

6.2.11 系统控制块（SCB） 99

6.3 模拟比较器及运算放大器模块 105

6.3.1 概述 105
6.3.2 比较器特性 105
6.3.3 运算放大器特性 108
6.3.4 寄存器映射 109
6.4 模拟数字转换器（ADC） 124
6.4.1 概述 124
6.4.2 特性 124
6.4.3 ADC 功能描述 125
6.4.4 寄存器映射 133
6.5 Flash 内存控制器(FMC) 162
6.5.1 概述 162
6.5.2 特性 162
6.5.3 Flash 存储器组织结构 162
6.5.4 寄存器映射 163
6.6 IO 配置管理器（IOCONFIG） 175
6.6.1 概述 175
6.6.2 特性 175
6.6.3 IO 管脚结构框图 176
6.6.4 IO 管脚功能复用 176
6.6.5 IO 管脚配置寄存器映射 179
6.7 GPIO 控制器（GPIO） 183
6.7.1 概述 183
6.7.2 GPIO 寄存器映射 183
6.8 I2C 总线控制器(I2C) 196
6.8.1 概述 196
6.8.2 特性 196
6.8.3 基本配置 196
6.8.4 寄存器映射 197
6.9 串行外设接口控制器(SPI) 201
6.9.1 概述 201

6.9.2 特性 201
6.9.3 SPI 控制器框图 202
6.9.4 基本配置 202
6.9.5 SPI 功能描述 203
6.9.6 寄存器映射 206
6.10 定时器和 PWM 发生器 216
6.10.1 概述 216
6.10.2 特性 216
6.10.3 定时器结构框图 217
6.10.4 基本配置 218
6.10.5 基本操作 218
6.10.6 寄存器映射 222
6.11 MotorPWM 控制器(MCPWM) 258
6.11.1 概述 258
6.11.2 特性 258
6.11.3 MCPWM 结构框图 259
6.11.4 基本配置 259
6.11.5 基本操作 260
6.11.6 寄存器映射 272
6.12 UART 接口控制器(UART) 311
6.12.1 概述 311
6.12.2 特性 311
6.12.3 功能描述 312
6.12.4 寄存器映射 315
6.13 窗看门狗 WWDG 325
6.13.1 概述 325
6.13.2 基本特性 325
6.13.3 基本配置 325
6.13.4 功能描述 326
6.13.5 寄存器描述 327

6.14 独立看门狗 IWDG 331
6.14.1 概述 331
6.14.2 基本特性 331
6.14.3 基本配置 331
6.14.4 功能描述 332
6.14.5 寄存器描述 332
6.15 触摸按键控制器（TSC） 338
6.15.1 概述 338
6.15.2 基本特性 338
6.15.3 基本配置 338
6.15.4 功能描述 339
6.15.5 寄存器描述 340
6.16 简单 TIMER 345
6.16.1 概述 345
6.16.2 特性 345
6.16.3 基本配置 345
6.16.4 寄存器映射 345
6.17 USB 全速接口 352
6.17.1 概述 352
6.17.2 特性 352
6.17.3 基本配置 352
6.17.4 功能描述 353
6.17.5 寄存器配置 353
6.18 CRS 系统（时钟回馈系统） 369
6.18.1 概述 369
6.18.2 特性 369
6.18.3 基本配置 370
6.18.4 功能描述 370
6.18.5 寄存器配置 374
6.19 控制器局域网(CAN) 381

6.19.1 概述 381
6.19.2 特性 381
6.19.3 模块框图 382
6.19.4 基本配置 382
6.19.5 功能描述 383
6.19.6 寄存器映射 387
7. MT32F005/006 电气特性 413
8. 封装尺寸 414
8.1 TSSOP20 414
8.2 QFN20 415
8.3 QFN28 416
8.4 SOP16 416
9. 版本历史 417

1. 概述

MT32F005/006 系列 MCU 是以 Cortex-M0 为内核的 32 位微控制器， CPU 运行频率可以达到 96MHZ，工作电压 2.0V ~ 5.5V，工作温度-40℃ ~ 105℃。 在系统运行于 25K IRC 时，芯片的待机功耗（**数据全部保持**）最低可以降低到 4uA(3.3V 供电条件下)。同时芯片还可提供高达 6KV 的 ESD 特性（HBM）。

芯片内嵌 32K 字节的 Flash 存储器，以及 4K 字节的 SRAM 存储器。内部还包含许多系统级外设功能，如高速通用 I/O 端口，USB2.0 FS Dev**(仅 006 包含)**，CAN2.0B**(仅 006 包含)**，Timer，UART，SPI，I2C，PWM，Motor-PWM，ADC，模拟比较器，运算放大器**(仅 006 包含)**，触摸按键控制器，看门狗定时器，欠压检测器等。这些功能都被集成到 MT32F005/006 的芯片内部，具有高整合度、高抗干扰、高可靠性的特点。配合成熟的 Keil & IAR 调试开发软件，支持 C 语言及汇编语言，汇编指令。

芯片内部 SRAM 和 GPIO 模块支持位带操作，CPU 一条指令即可读取或改写任意 SRAM 空间中任意一个 bit 的值，而无需传统的 R-M-W 操作。也可以一条指令改变任意一个 GPIO 的输出电平而不影响其他 IO 的电平。

MT32F006 与市场同类芯片对比：(*参数摘自各芯片数据手册*)

	MT32F005/006	HC32F003/005	MM32F003	STM8S003	N76E003
内核	**32bit**	32bit	32bit	8bit	8bit
主频(Mhz)	**96**	32	24	16	16
Flash(KB)	**32**	16/32	16	8	18
SRAM(KB)	**4**	2/4	2	1	1
ADC	**14 通道 12bit@1.5Mhz**	9 通道 12bit@1Mhz	8 通道 12bit@1Mhz	5 通道 10bit@1Mhz	8 通道 12bit@1Mhz
ACMP	**2 通道**	2 通道	无	无	无
OPA	**无/2 通道**	无	无	无	无
USB	**无/2.0 FS dev**	无	无	无	无
CAN	**无/CAN2.0B**	无	无	无	无
TSC(触摸按键)	**20 通道**	无	无	无	无

GPIO	**24**	16	16	28	18
电压（V）	**2.0~5.5**	1.8~5.5	2.0~5.5	2.95~5.5	2.4~5.5
ESD	**6KV**	2KV	7KV	4KV	7KV

典型应用：

- 电机驱动，电动玩具
- 小家电，充电器，遥控器，电子烟，燃气报警器，数显表，温控器，记录仪等
- USB外设应用
- 工业CAN总线应用
- 智能交通，智慧城市，智能家居
- 火警探头，智能门锁，无线监控等智能传感器应用

2. 特性

- 内核
  - 32 位 CPU 内核，运行频率最高 96MHZ
  - 一个 24 位系统定时器
  - 支持低功耗睡眠模式
  - 单指令周期 32 位硬件乘法器
  - 嵌套向量中断控制器 NVIC 支持 32 个中断输入，每个中断有 4 个优先级
  - 支持串行调试接口（SWD），2 个观察点/4 个断点
- 宽电压工作范围：2.0V ~ 5.5V
- 128bit 唯一 ID
- 存储器
  - 32K 字节 Flash 用于存储用户程序
  - 4K 字节 SRAM 用于内部高速暂存存储器
- 时钟控制
  - 可编程的系统时钟源
  - 内部 12MHZ 高速 RC 振荡器
  - 内部 25KHZ 低功耗 RC 振荡器
  - 12MHZ 外部高速晶振输入
  - 32.768KHZ 外部晶振输入
  - PLL 支持 CPU 最高运行在 96MHZ
- I/O 端口
  - 在 TSSOP20/QFN20 管脚中最多支持 18 个通用 I/O 端口（GPIO），QFN28 封装中最多支持 24 个通用 I/O 端口（GPIO）
  - I/O 工作模式：
    - 双向模式
    - 模拟模式
    - 输入上拉、下拉、浮空模式
    - 推挽输出模式
    - 开漏、开源输出模式

- ◆ I/O 管脚可被配置为电平或边沿触发模式的中断源

- 16 位定时器/PWM（最多支持 4 路独立的 PWM 输出）
  - ◆ 一个定时器/计数器模块，包含一个可编程的 16 位预分频器以及 4 路独立的 16 位计数通道
  - ◆ 每个计数通道可执行定时器或计数器操作
  - ◆ 每个计数通道作为计数器时都可以选择向上、向下或中心对齐方式计数
  - ◆ 每个计数通道可被配置成捕获通道，可在输入信号跳变时捕捉计数通道定时器的瞬时值，捕获事件也可以产生中断
  - ◆ 每个计数通道都有一个16位匹配寄存器，允许进行以下操作：
    - ➢ 匹配时连续工作，在匹配时可选择产生中断
    - ➢ 匹配时停止定时器工作，可选择产生中断
    - ➢ 匹配时复位定时器，可选择产生中断
  - ◆ 每个计数通道都有一个与匹配寄存器相对应的外部输出，这些输出可以具有以下功能：
    - ➢ 匹配时设为低电平
    - ➢ 匹配时设为高电平
    - ➢ 匹配时翻转电平
    - ➢ 匹配时不执行任何操作
  - ◆ 对于每个计数通道，都可将对应的匹配寄存器配置为 PWM 输出
  - ◆ PWM 输出可以设置为单脉冲输出模式以及固定脉冲个数输出模式
  - ◆ 在捕获模式下可以自动测量捕获信号的高低电平占空比
  - ◆ 支持载波功能
  - ◆ 支持触发 ADC 采样
- 16 位简易定时器
  - ◆ 2 路 16 位简易定时器，可定时产生中断
  - ◆ 时钟源可选择
- 窗看门狗定时器
  - ◆ 带内部预分频的可编程 32 位计数器
  - ◆ 可编程的递减计数器

- ◆ 未在一个特定的窗口时间内喂狗会导致产生中断或复位
- ◆ 可产生早期唤醒中断

- 独立看门狗定时器
  - ◆ 由内部 25K 振荡器驱动，即使系统主时钟失效也可继续工作
  - ◆ 可编程的递减计数器
  - ◆ 计数到 0 时产生中断
- 16 位三相电机控制模块（Motor-PWM）
  - ◆ 含有三个独立的通道，每个通道包括：
    - ➢ 一个 16 位定时器/计数器（TC）
    - ➢ 一个 16 位界限寄存器（LIM）
    - ➢ 一个 16 位匹配寄存器（MAT）
    - ➢ 一个 10 位死区时间寄存器（DT）和相应的死区时间计数器
    - ➢ 一个 16 位捕获寄存器
    - ➢ 两个极性相反的已调制的输出（MCOA 和 MCOB）
    - ➢ 一个周期中断，一个脉宽中断，一个捕获中断
  - ◆ 输入引脚 MCI0-2 可触发 TC 捕获或使通道的计数值加 1，全局异常输入可强制所有通道进入“无效”状态并产生一个中断
  - ◆ 支持三相 DC 模式
  - ◆ 支持三相 AC 模式
  - ◆ 支持带霍尔传感器的三相无刷直流电机模式
  - ◆ 延后换相功能，延后时间点可灵活配置
  - ◆ 支持硬件测速功能
  - ◆ 支持硬件移相功能，通道 1 和 2 可硬件设置相对通道 0 的相位偏移
  - ◆ 支持触发 ADC 采样，可灵活配置触发 ADC 采样时间点
- TSC
  - ◆ 支持多达 20 个电容传感通道
  - ◆ 无需外部电容
  - ◆ 每个 IO 可单独设置计数器的比较阈值，提高产品的灵活性和稳定性
  - ◆ 支持硬件自动扫描模式，减少软件干预

- ◆ 采用内部参考电压，有效的抗外部电压干扰，提高触摸的稳定性

- SPI
  - ◆ 支持 SPI 主机/从机模式
  - ◆ 支持 SPI 数据格式
  - ◆ 支持全双工以及半双工数据传输（从机模式下只支持半双工）
  - ◆ 数据长度可改变（4-16bit）
  - ◆ 时钟初始相位和极性可配置
  - ◆ 独立的 SPI 时钟源
  - ◆ SPI 实际传输速率达到 24MHz
- UART
  - ◆ 可编程波特率发生器
  - ◆ 接收器和发送器支持缓冲，均带有 16bytes 的 FIFO 缓冲
  - ◆ 可编程数据字长度(5、6、7、8 位)
  - ◆ 可配置的停止位--1 或 2 个停止位
  - ◆ 支持总线 IDLE 检测，timeout 时间可配置
- I2C
  - ◆ 支持主机/从机模式
  - ◆ 主从机之间双向数据传输
  - ◆ 多主机总线支持（无中心主机）
  - ◆ 多主机同时发送数据时进行仲裁，总线上串行数据不会被损坏
  - ◆ 可编程配置的时钟可适应多样化的传输速率控制.
  - ◆ 串行时钟同步允许不同位速率的器件通过同一个串行总线通信
  - ◆ 串行时钟同步可作为一个握手机制来挂起和恢复串行传输
- CAN
  - ◆ 支持完整的 CAN 协议（CAN2.0A 和 CAN2.0B）
  - ◆ 支持 11 位和 29 位标识符多主机总线支持（无中心主机）
  - ◆ 多通信速率支持系统时钟的 1/(16~3200)，支持宽度大于 5Kbaud~1Mbaud
  - ◆ 64 字节接收 FIFO
  - ◆ 软件驱动的位速率检测（支持热插拔）

- ◆ 接收滤波
- ◆ 单次发送选择（无重发）
- ◆ 只听模式（无确认、无激活的错误标志）
- ◆ 接收自身报文（自接收请求）
- ◆ 对每一个 CAN 总线错误产生中断
- ◆ 读/写错误计数器
- ◆ 最近一次的错误代码寄存器
- ◆ 可编程错误报警限额寄存器

- USB2.0 FS DEV
  - ◆ 符合 USB2.0 全速设备的技术规范
  - ◆ 可以工作在 device 模式
  - ◆ Endpoint 0-3 双向数据传输端口
  - ◆ 512Byte 的专用数据缓存
  - ◆ 时钟恢复功能，无需外挂晶振
  - ◆ USB 所需 48MHz 时钟来自于 PLL，若要使能 USB 模块，PLL 必须工作于 96MHz 或 48MHz

- ADC
  - ◆ 12 位逐次逼近式模数转换器 ADC
  - ◆ 12bits/10bits/8bits/6bits 分辨率可调
  - ◆ 高达 1.5 Mhz 的转换速率
  - ◆ 模拟输入测量电压范围: 5mV～VDD*0.99
  - ◆ 基准电压可选择 VDD 或片内基准电压（2.0V@typical）
  - ◆ 自带失调校准以及增益误差校准
  - ◆ 最多 14 个外部通道输入
  - ◆ 两个通道可用于测量内部 OPA 的输出
  - ◆ 1 个内部温度传感器检测输入
  - ◆ 1 个内部通道用来检测 1/4 VDD 电压
  - ◆ 转换开始可由软件、内部硬件信号或外部引脚触发
  - ◆ 支持单次转换模式/连续转换模式/非连续转换模式

- ◆ 每个通道都有独立的数据结果寄存器
- ◆ 支持模拟看门狗功能

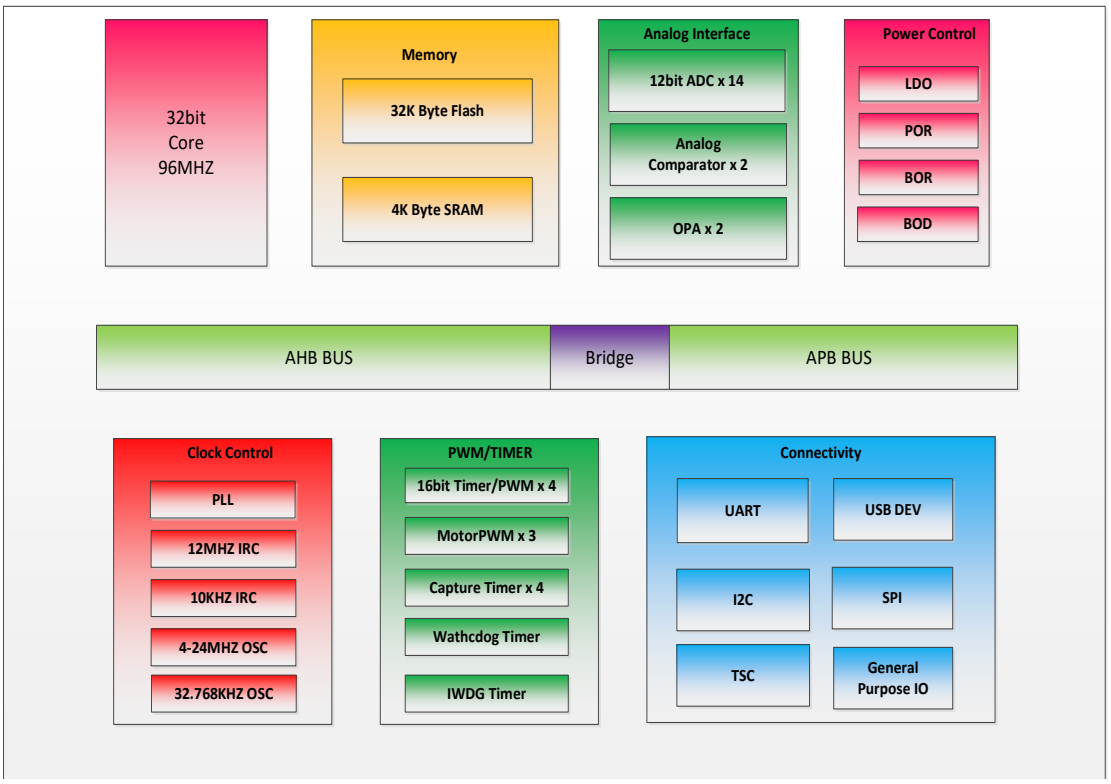
- ACMP
  - ◆ 2 组模拟比较器模块
  - ◆ 每个比较器模块包含 POS 和 NEG 两个端口
  - ◆ 每个 POS 和 NEG 都有对应的外部 IO，可接外部电压，也可选择内部参考电压，也可选择内部 OPA 的输出
  - ◆ 迟滞功能，迟滞电压 4 档可调(15mV/40mV)
  - ◆ 比较结果发生改变时可产生中断
- OPA
  - ◆ 两路运算放大器模块
  - ◆ 运算放大器的增益内部 8 档可调，可内部配置成 1/2/3/4/6/8/10 倍放大
  - ◆ 运算放大器的增益也可由外部电阻网络决定
  - ◆ 放大器的输出可以作为比较器的 POS 端
  - ◆ 放大器的输出可以接至 ADC 的输入通道
- 欠压检测（BOD）
  - ◆ 支持 4 级检测电压（4.2V/3.6V/2.6V/2.1V）
  - ◆ 可以产生中断
- 欠压复位（BOR）
  - ◆ 支持 4 级复位电压（3.7V/2.7V/2.2V/1.8V）
  - ◆ 可以产生中断或复位芯片
- 工作温度
  - ◆ -40℃ ~ 105℃
- 封装
  - ◆ 无铅封装（RoHS）
  - ◆ 20pin-TSSOP，20pin-QFN,，16pin-SOP，28pin-QFN

3. 缩写

ACMP	**Analog Comparator controller**
ADC	**Analog-digital Convertor**
APB	**Advanced Peripheral Bus**
AHB	**Advanced High-performance Bus**
BOD	**Brown-out Detect**
BOR	**Brown-out Reset**
CAN	**Controller Area Network**
CRS	**Clock Recovery System**
DAP	**Debug Access Port**
FIFO	**First-in, First-out**
FMC	**Flash memory controller**
GPIO	**General-Purpose Input/Output**
HCLK	**Clock of AHB**
I2C	**Inter-Integrated Circuit**
IAP	**In Application Program**
ICP	**In Circuit Program**
ISP	**In System Program**
IRC	**Internal RC Oscillator**
IWDG	**Independent Watchdog Timer**
LDO	**Low Dropout Regulator**
MPWM	**Motor PWM**
NVIC	**Nested Vectored Interrupt Controller**
OPA	**Operation Amplifier**
PCLK	**Clock of APB**
PLL	**Phase-Locked Loop**
PWM	**Pulse Width Modulation**
SPI	**Serial Peripheral Interface**

TSC	**Touch Sense Controller**
USART	**Universal Synchronous Asynchronous Receiver/Transmitter**
USB	**Universal Serial Bus**
WWDT	**Window Watchdog Timer**

4. 方块图



5. 选型表和管脚图

5.1 MT32F005/006 选型表

Part Number	RAM	Flash	I/O	Timer\ PWM	Motor PWM	USB2.0 DEV	CAN	SPI	I2C	TSC	UART	ADC	ACMP	OPA	Package
MT005C6PB	4KB	32KB	18	4	3	-	-	1	1	18	1	14	2	-	TSSOP20
MT005C6UB	4KB	32KB	18	4	3	-	-	1	1	18	1	14	2	-	QFN20
MT005C6WA	4KB	32KB	14	4	3	-	-	1	1	14	1	12	1	-	SOP16
MT005C6XA	4KB	32KB	24	4	3	-	-	1	1	18	1	14	2	-	QFN28
MT006C6PB	4KB	32KB	18	4	3	1	1	1	1	18	1	14	2	1	TSSOP20
MT006C6UB	4KB	32KB	18	4	3	1	1	1	1	18	1	14	2	1	QFN20
MT006C6WA	4KB	32KB	14	4	3	1	1	1	1	14	1	12	1	1	SOP16
MT006C6XA	4KB	32KB	24	4	3	1	1	1	1	18	1	14	2	2	QFN28

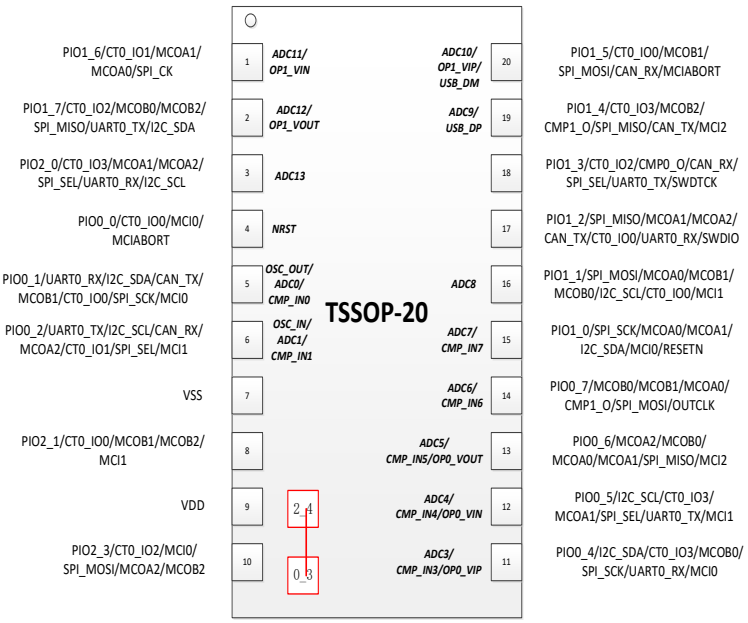
***：最大可提供 28 管脚封装，GPIO 数量扩展到 24 个**

***：QFN20，TSSOP20 封装管脚与主流 003/005 系列兼容**

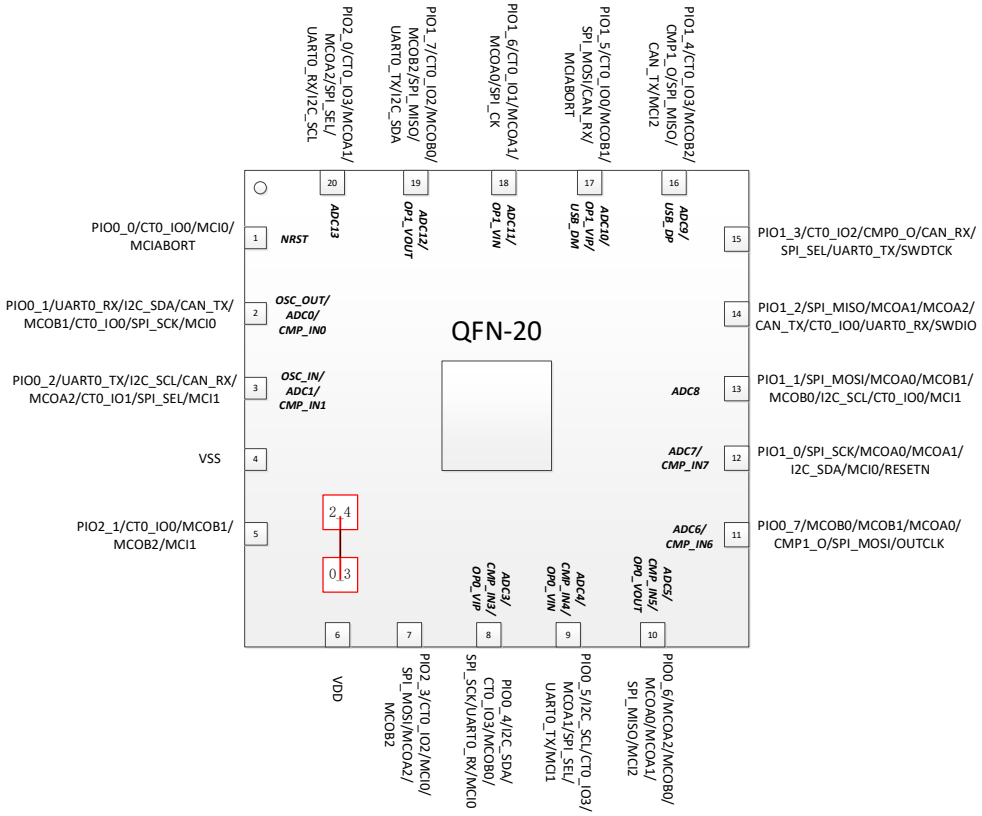
5.2 管脚图

以下管脚图中，GPIO 的外圈描述为该 IO 的数字复用功能，内圈描述为该 IO 作为数模混用 IO 时的模拟 IO 功能。

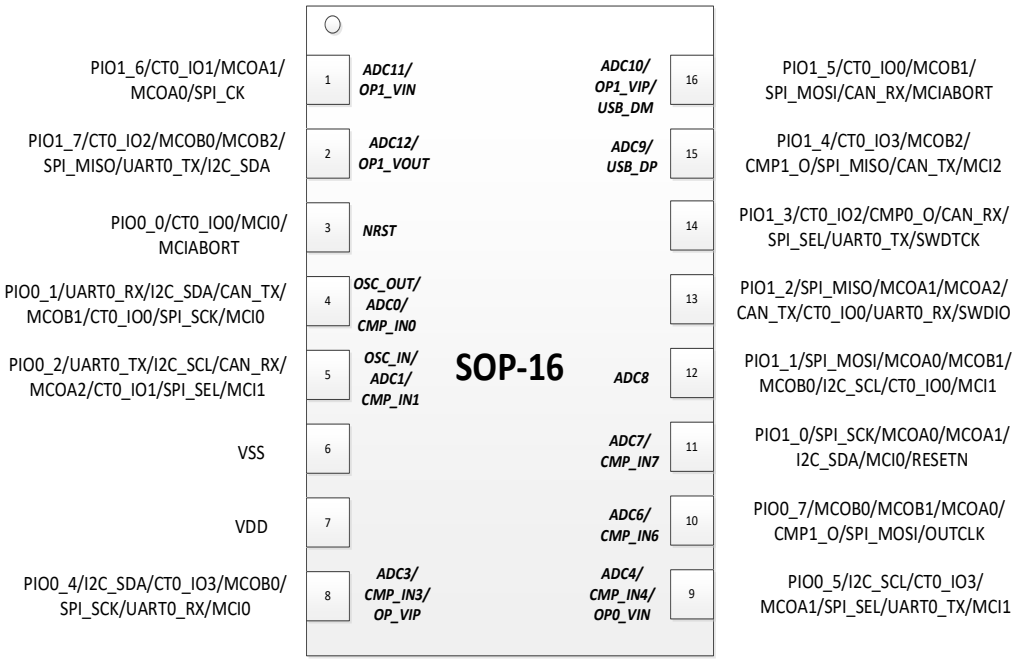
5.2.1 TSSOP-20 封装



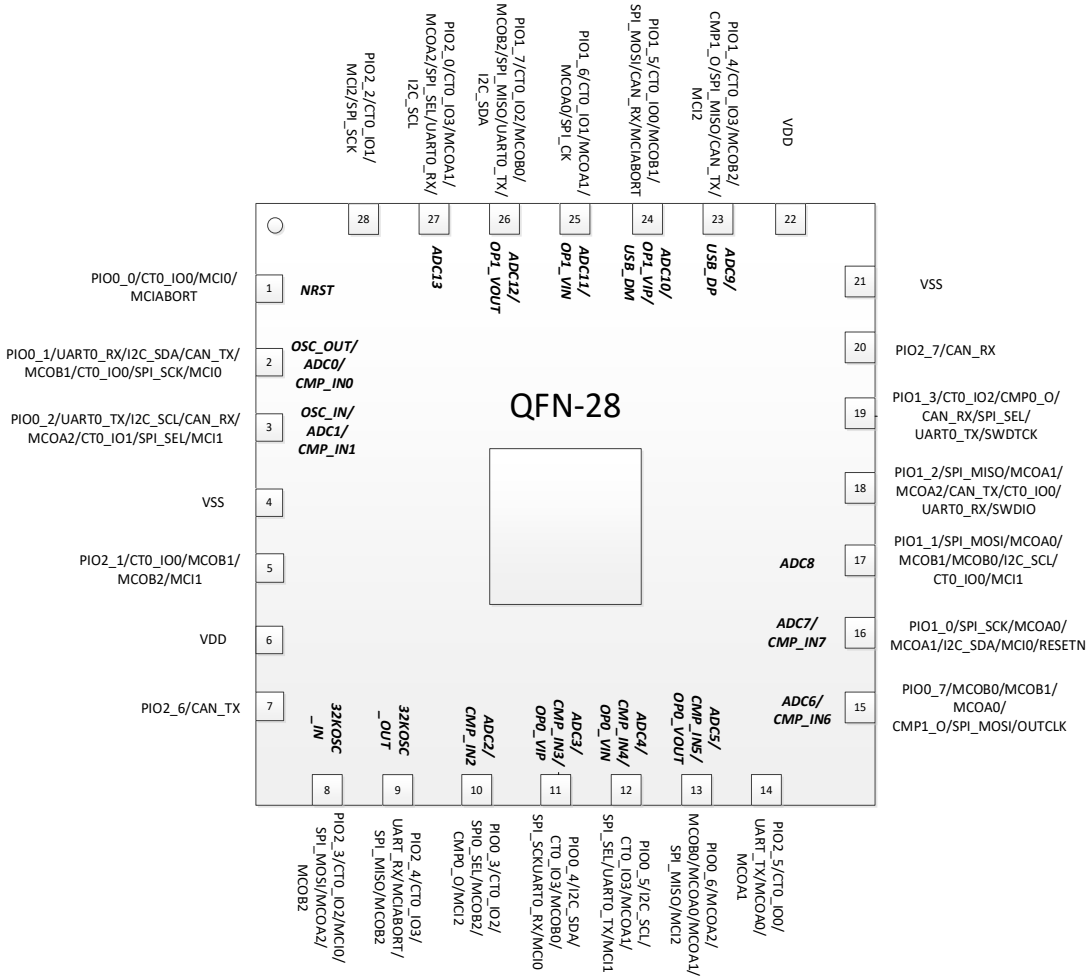
5.2.2 QFN-20 封装



5.2.3 SOP-16 封装



5.2.4 QFN-28 封装



5.3 管脚描述

管脚号				符号	数字复用功能								模拟功能
TSSOP 20	QFN 20	SOP 16	QFN 28		0	1	2	3	4	5	6	7	
	6	7	6,22	VDD									电源
7	4	6	4,21	VSS									地
4	1	3	1	PIO0_0	GPIO	CT0 IO0	MCI0					MCI ABORT	NRST 外部复位
5	2	4	2	PIO0_1	GPIO	UART0 RX	I2C SDA	CAN TX	MCO B1	CT0 IO0	SPI SCK	MCI0	OSC_OUT ADC0 CMP_IN0
6	3	5	3	PIO0_2	GPIO	UART0 TX	I2C SCL	CAN RX	MCO A2	CT0 IO1	SPI SEL	MCI1	OSC_IN ADC1 CMP_IN1
			10	PI0_3	GPIO	CT0 IO2	SPI SEL	MCO B2	CMP0 OUT			MCI2	ADC2 CMP_IN2
11	8	8	11	PIO0_4	GPIO	I2C SDA	CT0 IO3	MCO B0		SPI SCK	UART0 RX	MCI0	ADC3 CMP_IN3 OP0_VIP
12	9	9	12	PIO0_5	GPIO	I2C SCL	CT0 IO3	MCO A1		SPI SEL	UART0 TX	MCI1	ADC4 CMP_IN4 OP0_VIN
13	10		13	PIO0_6	GPIO	MCO A2	MCO B0	MCO A0	MCO A1	SPI MISO		MCI2	ADC5 CMP_IN5 OP0_OUT
14	11	10	15	PIO0_7	GPIO	MCO B0	MCO B1	MCO A0	CMP1 OUT	SPI MOSI		OUT CLK	ADC6 CMP_IN6
15	12	11	16	PIO1_0	GPIO	SPI SCK	MCO A0	MCO A1	I2C SDA		MCI0	OUT RESET	ADC7 CMP_IN7
16	13	12	17	PIO1_1	GPIO	SPI MOSI	MCO A0	MCO B1	MCO B0	I2C SCL	CT0 IO0	MCI1	ADC8 CMP_IN8

17	14	13	18	PIO1_2	GPIO	SPI MISO	MCO A1	MCO A2	CAN TX	CT0 IO0	UART0 RX	SWD IO	
18	15	14	19	PIO1_3	GPIO	CT0 IO2		CMP0 OUT	CAN RX	SPI0 SEL	UART0 TX	SWD SCK	
19	16	15	23	PIO1_4	GPIO	CT0 IO3		MCO B2	CMP1 OUT	SPI MISO	CAN TX	MCI2	ADC9 USB_DP
20	17	16	24	PIO1_5	GPIO	CT0 IO0		MCO B1	MCI1	SPI MOSI	CAN RX	MCI ABORT	ADC10 OP1_VIP USB_DM
1	18	1	25	PIO1_6	GPIO	CT0 IO1		MCO A1	MCO A0	SPI SCK			ADC11 OP1_VIN
2	19	2	26	PIO1_7	GPIO	CT0 IO2	MCO B0	MCO B2	SPI MISO	UART0 TX		I2C SDA	ADC12 OP1_OUT
3	20	3	27	PIO2_0	GPIO	CT0 IO3	MCO A1	MCO A2	SPI SEL	UART0 RX		I2C SCL	ADC13
8	5		5	PIO2_1	GPIO	CT0 IO0		MCO B1	MCO B2	MCI1			
			28	PIO2_2	GPIO	CT0 IO1		MCI2	SPI SCK				
10	7		8	PIO2_3	GPIO	CT0 IO2		MCI0	SPI MOSI	MCO A2	MCO B2		32KOSC IN
			9	PIO2_4	GPIO	CT0 IO3	UART RX	MCI ABORT	SPI MISO	MCO B2			32KOSC OUT
			14	PIO2_5	GPIO	CT0 IO0	UART TX	MCO A0	MCO A1				

			7	PIO2_6	GPIO	CAN TX							
			20	PIO2_7	GPIO	CAN RX							

注：GPIO：表示通用输入输出 IO 功能

CT0_IOx：表示 TIMER0 的 IO0/1/2/3，每组 Timer 有 4 个 IO，可配置为 PWM 输出或 Capture 输入

MCx：表示 MotorPWM 模块的管脚，共有 3 组互补 PWM 输出管脚：MCOA0/1/2, MCOB0/1/2，

三组输入捕获管脚：MCI0/1/2，一个全局输入管脚：MCIABORT

SPI：SPI 模块管脚，共有 4 个管脚：SPI_SCK，SPI_SEL，SPI_MISO，SPI_MOSI

I2C：I2C 模块管脚，共有 2 个管脚：I2C_SCL, I2C_SDA

UART：UART0 模块管脚，每个模块有 2 个管脚：UART0_TX, UART0_RX

CAN：CAN 模块管脚，每个模块有 2 个管脚：CAN_TX，CAN_RX

USB：USB 模块管脚，有两个管脚：USB_DP，USB_DM

CMPx：比较器模块管脚，有两个结果输出管脚：CMP0_out，CMP1_out，输入管脚与 ADC 复用

OPx：运放模块管脚，每个运放有三个管脚：VIP，VIN，OUT

SWDx: SWD 调试接口管脚，共有两个管脚：SWD_CLK, SWD_IO

ADCx: ADC 模块输入管脚，共有 14 个管脚：ADC0-ADC13

CMP_Inx：模拟比较器模块输入管脚，共有 8 个管脚：CMP_IN0-CMP_IN7，与 ADC0-7 复用

OUTCLK：时钟输出管脚，可以将芯片内部时钟源信号从此管脚输出

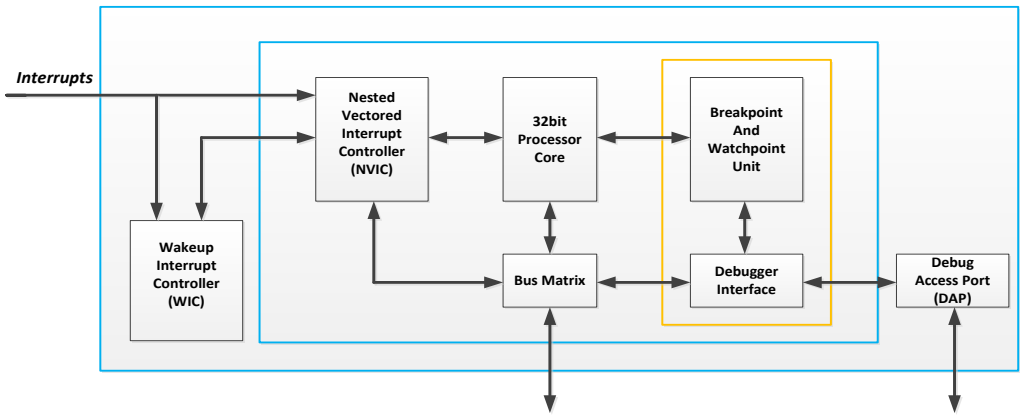
RESET： 复位输出管脚，可以将芯片内部复位源信号从此管脚输出

6. 功能描述

6.1 CPU 内核及总线架构

6.1.1 32-bit CPU 内核

MT32F005/006 采用的处理器是 32 位多级可配置的 RISC 处理器，最大主频 96MHZ。它有 AMBA AHB-Lite 接口和嵌套向量中断控制器（NVIC），具有可选的硬件调试功能。该系列处理器支持两种操作模式:Thread 模式和 Handler 模式。当有异常发生时，处理器进入 Handler 模式。异常返回只能在 Handler 模式下发生。当复位时，处理器会进入 Thread 模式，处理器也可在异常返回时进入到 Thread 模式。下图显示了处理器内核的各个功能模块。



图：CPU 内核功能框图

CPU 设备提供：

- 低门数 32bit 处理器，特性如下：
  - 最高主频 96MHZ
  - 24-bit SysTick 定时器
  - 32-bit 硬件乘法器
  - 系统接口支持小端(little-endian)数据访问
  - 具有确定性，固定延迟的中断处理能力
  - 可以丢弃和重新开始多次加载/存储和多周期乘法指令以保证快速中断处理

- 与 C 应用程序二进制接口兼容的异常模式（C-ABI）
- 使用中断等待（WFI），事件等待（WFE）指令，或者从中断返回时直接进入睡眠的 sleep-on-exit 特性可以进入低功耗休眠模式

- NVIC 特性
  - 32 个外部中断输入，每个中断具有 4 级优先级
  - 不可屏蔽中断输入（NMI）
  - 支持电平和脉冲触发中断
- 调试
  - 四个硬件断点
  - 两个观察点
  - 用于非侵入式代码的程序计数采样寄存器（PCSR）
  - 单步和向量捕获能力
- 总线接口
  - 单一 32 位的 AMBA3 AHB-Lite 系统接口，为所有的系统外设和存储器提供方便的集成
  - 支持 DAP（Debug Access Port）的单一 32 位的从机接口

6.1.2 系统总线架构

芯片内部采用多层 AHB 总线架构，AHB 总线频率与 CPU 相同，每个 Slave 模块都有一个独立的 AHB 总线。当不同的 master 访问不同的 slave 时，操作可以同时进行，最大化了系统内部的总线带宽。

6.2 系统管理器

6.2.1 概述

系统管理器包括如下功能：

- 系统复位
- 系统电源架构
- 系统时钟架构
- 系统功率控制
- 系统存储器映射
- 系统定时器（SysTick）
- 系统控制寄存器

6.2.2 系统复位

系统复位可由如下事件发起，这些复位事件标志可由 SYSRSTSTAT 寄存器读出。

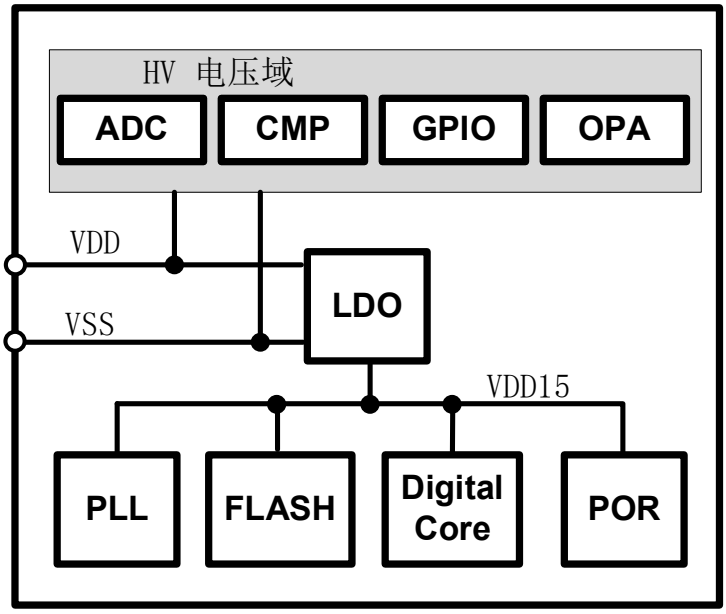
- 硬件复位
  - 上电复位
  - 复位脚（NRST）上有低电平（为增加芯片可用 IO，NRST 管脚与 PIO0_0 复用，芯片复位后该 IO 默认为 NRST 功能，必须保持高电平芯片才能正常工作，程序运行时可以通过 ANACTRL 寄存器中的 dis_nrst 位来屏蔽改 IO 的复位功能）
  - 看门狗定时溢出复位（WDT）
  - 独立看门狗复位（IWDG）
  - 低电压复位（BOR）
- 软件复位
  - MCU 复位 - SYSRESETREQ（AIRCR[2]）

6.2.3 系统电源架构

该芯片的电源架构分为两个部分：

- 来自 VDD 和 VSS 的 IO 高压电源，为模拟模块（ADC，CMP，OPA 等等）以及 GPIO 提供工作电源，包括 LDO 的输入电源也来自 VDD。
- 片内 LDO 的输出电源 VDD15,改电源负责给内核逻辑以及 PLL，Flash 等模块供电
- 内部稳压器 LDO 为 CAPLESS 架构，不需要片外稳压电容。

下图给出了 MT32F005/006 内部电源架构图：



图：芯片电源架构框图

6.2.4 系统时钟架构

时钟控制器为整个芯片提供时钟，包括系统时钟和所有外设时钟。时钟控制器还利用独立的时钟 ON/OFF 控制、时钟源选择和时钟分频器来实现功耗控制功能。

系统时钟发生器由如下 5 个时钟源组成：

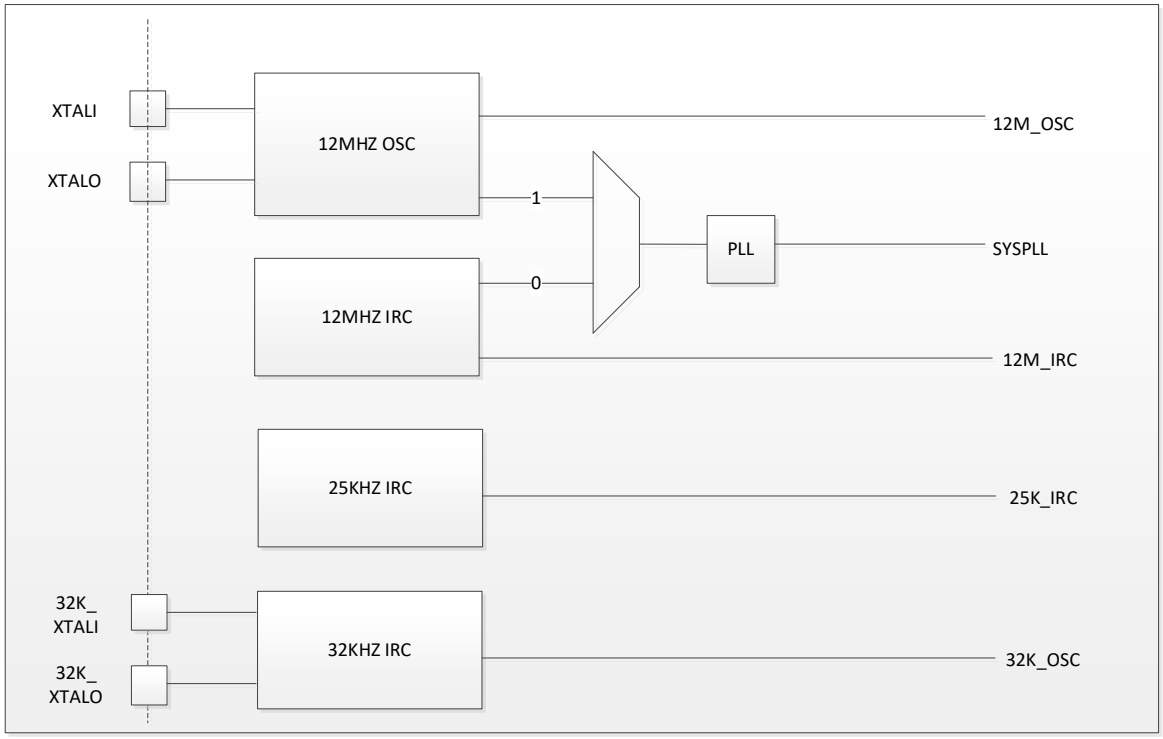
12MHZ 内部 RC 振荡器（12M_IRC）

12MHZ 外部晶振（12M_OSC）

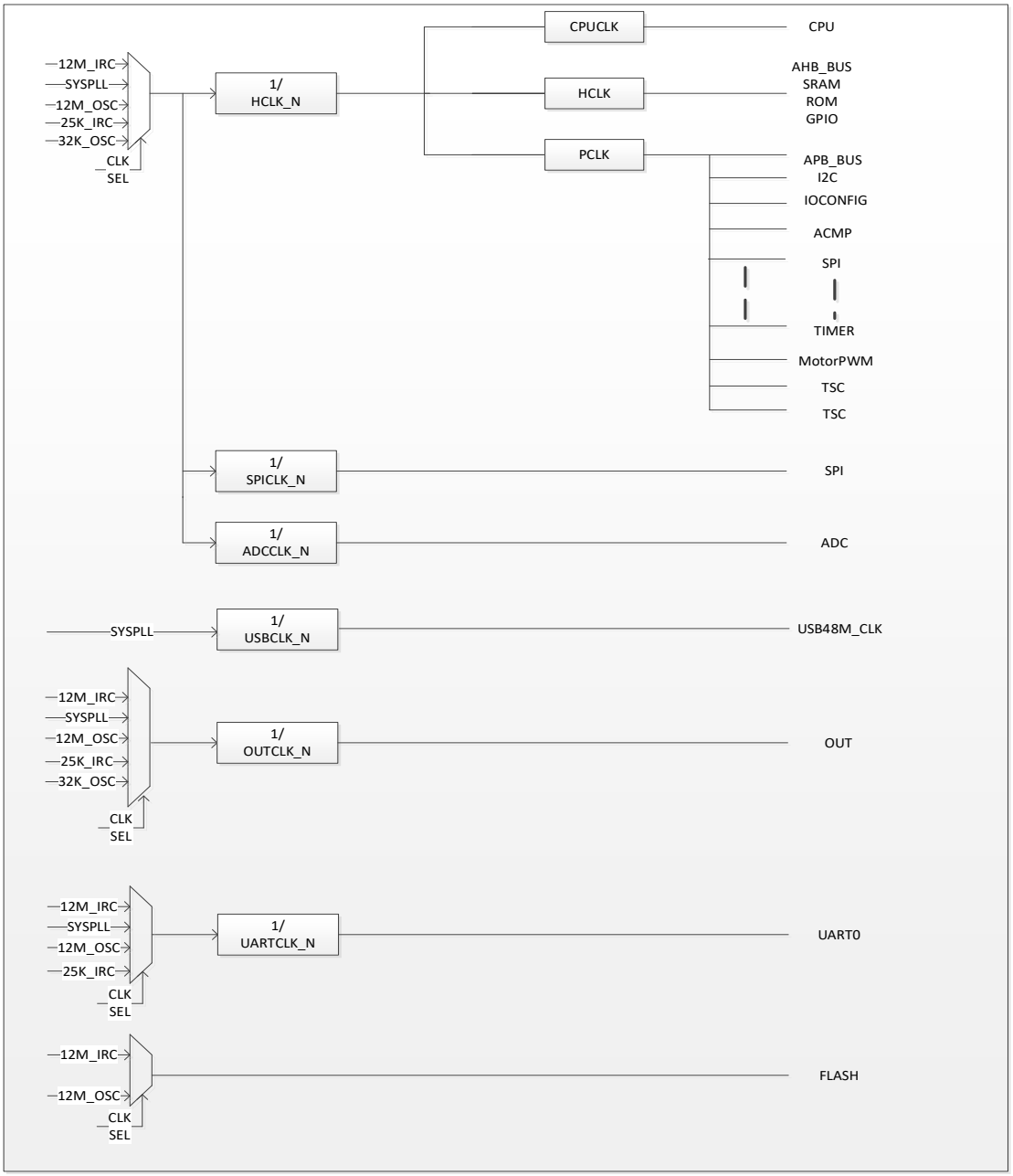
25KHZ 内部低功耗振荡器(25K_IRC)

32KHZ 外部晶振（32K_OSC）

96MHZ 的内部 SYSPLL（PLL 输入时钟源可选择 12M_IRC 或是 12M_OSC）



图：时钟发生器框图



图：系统时钟源框图

6.2.5 系统功率控制

MT32F005/006 系列 MCU 支持多种功耗控制特性，通过优化芯片中模块的电源和时钟可以降低处理器运行时的功率消耗。芯片共有三种专门的处理器节电模式：睡眠模式、深度睡眠模式、以及掉电模式。

在睡眠模式下，CPU 内核时钟会被关闭，用户可以自行选择哪些外设可以继

续运行，可以将不需要运行的外设时钟关闭。中断信号可以将 CPU 从睡眠模式中唤醒。

在深度睡眠模式下，芯片会自动关闭大部分数字模块的时钟，用户还可以选择哪些模拟模块被关闭，来减少消耗的功率。同时还可以选择进入低功耗模式来进一步降低芯片功耗。

在掉电模式下，数字内核掉电，模拟部分除 WAKEUP 模块外也都处于掉电状态，此时可以通过芯片的 6 个具有 WAKEUP 功能的 IO 以及 NRST 管脚来给芯片上电,WAKEUP 功能可被软件屏蔽。

根据实际需要，用户可以通过切换时钟源、重新配置 PLL 值和/或改变系统时钟的分频值，来控制 CPU 以及系统的运行频率。这样可以根据实际应用的需要来对功耗和处理速度进行权衡。

同时，运行时（runtime）功耗控制允许将片内各外设各自的时钟关闭，也允许关闭应用无需使用到的外设来减少功耗。

下面是芯片的几种运行模式具体的描述:

- 运行模式：

CPU 内核、总线，存储器以及外设模块的控制寄存器都由系统时钟(CPUCLK,HCLK,PCLK)来提供时钟。AHBCLKCTRL 寄存器用来控制哪个存储器和外设运行。系统时钟频率可由 AHBCLKDIV 寄存器来控制。

除了系统时钟外，被选用的外设（如 USART、SPI、WDT 等）还有自己的外设时钟，这些外设时钟都有自己的时钟分频器。外设时钟可通过对应的时钟分频寄存器来关闭。

模拟模块（PLL，IRC，ADC，OPA，BOD 以及 FLASH 等）的功耗可通过 PDRUNCFG 寄存器来分别控制。

- 睡眠模式

在睡眠模式下，CPU 内核的系统时钟会停止运行，指令执行也被挂起，直到发生复位或者中断。

执行 CPU 内核的等待中断（WFI）指令就可以进入睡眠模式，当有中断到达

内核时就会自动退出睡眠模式。

如果外设时钟在 AHBCLKCTRL 寄存器中设置为打开，那么外设功能在睡眠模式下仍然可以运行，而且可以通过触发中断来使处理器从睡眠模式中恢复。在睡眠模式下，内核、存储系统和相关控制逻辑、以及内部总线都会被关闭。

内核状态和寄存器、外设寄存器以及内部 SRAM 的值都被保留，而管脚上的逻辑电平保持不变。

- 深度睡眠模式

在深度睡眠模式下，整个芯片处于睡眠状态，系统大部分时钟被自动关闭，大部分外设时钟也被自动关闭，可以通过配置 HW_DEEPSLEEP_CTRL 寄存器来设置哪些数字模块的时钟在进入深度睡眠模式时不被自动关闭。同时那些通过 PDSLEEPCFG 寄存器被设置的模拟模块也会被关闭。通过 PDSLEEPCFG 和 PDAWAKECFG 寄存器来设置要使哪些模拟模块在进入深度睡眠模式时被自动关闭以及再从深度睡眠模式恢复时被自动打开。

在深度睡眠模式下系统中也有一小部分逻辑是需要时钟来保持运行的，因此在进入深度睡眠模式时不能够将系统时钟源全部关闭，必须至少保留一个时钟源工作（通常为 IRC25K），并且将 MAIN_CLK 的时钟源切换到这个打开的时钟上。

进入深度睡眠模式的步骤如下：

1）通过 PDSLEEPCFG 寄存器选择哪些模拟模块要在深度睡眠模式下掉电（晶振、PLL、BOD、FLASH 等）

2）通过 PDAWAKECFG 寄存器选择哪些模拟模块要在深度睡眠模式唤醒时上电。

3）写 1 到 CPU 内核的 SCR 寄存器的 SLEEPDEEP 位。

4）执行 CPU 内核的 WFI 指令。

可以通过一些模块的中断功能来在深度睡眠模式下产生中断唤醒 CPU。Watch Dog，TIMER，GPIO，USART 等模块可以用来生成中断唤醒 CPU。同时所有 GPIO 上的中断也都可以用来唤醒 CPU。

在深度睡眠模式下，处理器状态和寄存器、外设寄存器以及内部 SRAM 值都会被保留，而管脚上的逻辑电平不变。

一旦进入深度睡眠模式，系统会自动关闭除了 FCLK 外的所有系统时钟。同时用户可以通过将 AHBCLKDIV 值设大来降低 FCLK 频率,并且可以将其他外设 CLK 的 DIV 寄存器设为 0 来关闭这些外设 CLK。这样一来系统中可以只有 FCLK 在运行，大大降低了动态功耗。

深度睡眠模式的好处在于用户可以关闭时钟生成模块，如振荡器和 PLL，因此比睡眠模式节省更多的动态功耗。

在深度睡眠模式下由于系统功耗可以被降低到几百个 uA 左右，这个时候如果需要进一步降低系统功耗，可以选择进入低功耗模式，这样整个系统的功耗可以降低到几个 uA。

进入深度睡眠模式并且进入低功耗模式的步骤如下：

1）通过 PDSLEEPCFG 寄存器选择哪些模拟模块要在深度睡眠模式下掉电（晶振、PLL、BOD、FLASH 等）

2）通过 PDAWAKECFG 寄存器选择哪些模拟模块要在深度睡眠模式唤醒时上电。

3）设置 PCON 寄存器中的 LOWPOWER 位为 1

4）写 1 到 CPU 内核的 SCR 寄存器的 SLEEPDEEP 位。

5）执行 CPU 内核的 WFI 指令。

退出深度睡眠模式+低功耗模式的步骤如下：

1） CPU 接收到中断退出 deepsleep 状态

2） PDAWAKECFG 寄存器选择的模拟模块自动上电

3） 等待 PCON 寄存器中的 NORMAL 位变为 1

4） 设置 PCON 寄存器中的 LOWPOWER 位为 0

5） 根据系统需求，切换时钟源，提升系统频率

- 掉电模式

在掉电模式下，除了具有 WAKEUP 功能的管脚，整个芯片的功率和时钟都被关闭。将 WAKEUP 功能管脚拉高可将芯片从掉电模式唤醒。在掉电模式下 SRAM 的内容不会被保存。

进入深度掉电模式的步骤如下：

1).将 PCON 寄存器的 DPDEN 位置位。

2).写 1 到 CPU 内核的 SRC 寄存器的 SLEEPDEEP 位。

3).执行 WFI 指令

所有 GPIO 中具有 WAKEUP 功能的 IO 共有 6 个，分别是：PIO1_0,PIO1_1,PIO1_5,PIO1_6,PIO1_7,PIO2_0,此外 NRST 管脚也可用来唤醒芯片。在掉电模式下，这 7 个 IO 中任意一个为高电平都可以唤醒芯片，在非掉电模式下这 7 个 IO 为普通 GPIO。要使用这些 GPIO 在掉电模式的唤醒功能，必须在掉电之前打开这些 IO 对应的唤醒使能位，详见 HW_WAKEUP 寄存器。NRST 管脚始终可以唤醒芯片。

6.2.6 系统存储器映射以及调试模式下地址重映射

MT32F005/006 系列 MCU 提供 4G 字节的寻址空间。每个片上模块存储器的地址分配情况如下表所示。详细的寄存器定义和寻址空间以及编程细节将在后续的各个片上外设描述章节里描述。MT32F005/006 系列仅支持小端数据格式。

在 MT32F005/006 中存在一个地址重映射机制，以供开发者对芯片进行调试开发。芯片上电后 CPU 会从 0x00000000 的地址开始执行程序，正常模式下这段空间是分配给 Flash 存储器的 Sector0 空间，但是在地址重映射模式下，可选择将 Flash 中的任意地址开始的 256Byte 的连续空间映射到 0x00000000 地址。（见 REMAP_FLASH 寄存器）

表：片上模块地址空间分配表

地址空间	标志	模块
Flash & SRAM 空间(0x0000_0000 – 0x2000_FFFF)		
0x0000_0000 – 0x0000_7FFF 0x0800_0000 – 0x0800_7FFF	Main Flash	32K Byte
0x1FFF_F400 – 0x1FFF_F7FF	INFO Flash	1K Byte
0x1FFF_F800 – 0x1FFF_FBFF	SYS Flash	1K Byte
0x2000_0000 – 0x2000_0FFF	SRAM	4K Byte

AHB 模块空间(0x4000_0000 – 0x4008_0000)		
0x4000_0000 – 0x4000_FFFF	APB	APB 桥外设寄存器（64KB）
0x4002_0000 – 0x4004_FFFF	GPIO	GPIO(port0-2)寄存器空间（192KB）
0x4006_0000 – 0x4006_FFFF	USB	USB 模块寄存器空间
APB 模块空间(0x4000_0000 – 0x4000_FFFF)		
0x4000_0000 – 0x4000_0FFF	SYSCONFIG	SYSCONFIG 模块寄存器
0x4000_1000 – 0x4000_17FF	IOCONFIG	IOCONFIG 模块寄存器
0x4000_1800 – 0x4000_1FFF	TSC	TSC 模块寄存器
0x4000_2000 – 0x4000_27FF	UART0	UART0 模块寄存器
0x4000_3000 – 0x4000_37FF	ACMP&OPA	ACMP&OPA 模块寄存器
0x4000_4000 – 0x4000_47FF	I2C0	I2C0 模块寄存器
0x4000_5000 – 0x4000_57FF	ADC	ADC 模块寄存器
0x4000_6000 – 0x4000_6FFF	CAN	CAN 模块寄存器
0x4000_7000 – 0x4000_77FF	MPWM	MotorPWM 模块寄存器
0x4000_8000 – 0x4000_87FF	SPI0	SPI0 模块寄存器
0x4000_9000 – 0x4000_97FF	TIMER0	TIMER0 模块寄存器
0x4000_B000 – 0x4000_BFFF	FLASH	FLASH 控制器模块寄存器
0x4000_C000 – 0x4000_C7FF	WDT	WATCHDOG 模块寄存器
0x4000_C800 – 0x4000_CFFF	IWDG	Independent Watchdog 模块寄存器
0x4000_D000 – 0x4000_DFFF	STIMER0	简易 TIMER 模块寄存器
0x4000_F000 – 0x4000_FFFF	CRS	CRS 模块寄存器
系统控制空间（0xE000_E000 – 0xE000_EFFF）		
0xE000_E010 – 0xE000_E0FF	SCS	系统定时器控制寄存器
0xE000_E100 – 0xE000_ECFF	SCS	外部中断控制器控制寄存器
0xE000_ED00–0xE000_ED8F	SCS	系统控制块寄存器

6.2.7 系统管理器控制寄存器列表

寄存器	偏移量	R/W	描述	复位值
SCR Base Address： SCR_BA = 0x40000000				
PRESETCTRL	SCR_BA+0x00	R/W	外设复位控制寄存器	0x001f_ffff
AHBCLKCTRL	SCR_BA+0x10	R/W	外设时钟控制寄存器	0x0000_10a7
BOOTREMAP	SCR_BA+0x20	RO	重映射及保护状态寄存器	0x0000_0000
REMAPFLASH	SCR_BA+0x24	R/W	FLASH 地址重映射控制寄存器	0x0000_0000
12OSC_CTRL	SCR_BA+0x28	R/W	12M 时钟控制寄存器	0x0000_0080
10IRC_CTRL	SCR_BA+0x2C	R/W	25K 振荡器时钟控制寄存器	0x0000_0008
SYSPLLCTRL	SCR_BA+0x30	R/W	系统 PLL 控制寄存器	0x8000_0002
SYSPLLSTAT	SCR_BA+0x34	R	系统 PLL 状态寄存器	0x0000_0001
LDOCTRL	SCR_BA+0x38	R/W	LDO 控制寄存器	0x0000_0088
SYSRSTSTAT	SCR_BA+0x40	R/W	系统复位源标识寄存器	0x0000_0001
MAINCLKSEL	SCR_BA+0x50	R/W	系统主时钟源选择寄存器	0x0000_0000
MAINCLKUEN	SCR_BA+0x54	R/W	系统主时钟源更新寄存器	0x0000_0000
USARTCLKSEL	SCR_BA+0x58	R/W	USART 时钟源选择寄存器	0x0000_0000
USARTCLKUEN	SCR_BA+0x5C	R/W	USART 时钟源更新寄存器	0x0000_0000
FLASHCLKSEL	SCR_BA+0x60	R/W	FLASH 擦写时钟源选择寄存器	0x0000_0000
FLASHCLKUEN	SCR_BA+0x64	R/W	FLASH 擦写时钟源更新寄存器	0x0000_0000
OUTCLKSEL	SCR_BA+0x70	R/W	OUT 时钟源选择寄存器	0x0000_0000
OUTCLKUEN	SCR_BA+0x74	R/W	OUT 时钟源更新寄存器	0x0000_0000
AHBCLKDIV	SCR_BA+0x80	R/W	AHB 时钟分频寄存器	0x0000_0002
USART0CLKDIV	SCR_BA+0x90	R/W	USART0 时钟分频寄存器	0x0000_0010
SYSTICKCLKDIV	SCR_BA+0xA0	R/W	SYSTICK 时钟分频寄存器	0x0000_0010
SPI0CLKDIV	SCR_BA+0xA4	R/W	SPI0 模块时钟分频寄存器	0x0000_0010
OUTCLKDIV	SCR_BA+0xAC	R/W	OUTCLK 时钟分频寄存器	0x0000_0001
ADCANACLKDIV	SCR_BA+0xB0	R/W	ADC 采样时钟分频寄存器	0x0000_0001
USBCLKDIV	SCR_BA+0xC0	R/W	USB48MCLK 时钟分频寄存器	0x0000_0001

WAKEUPEN	SCR_BA+0xE0	R/W	WAKEUP 管脚使能寄存器	0x0000_0000
SYSTICKCAL	SCR_BA+0x120	R/W	SYSTICK 校准寄存器	0x0000_0000
BORCTRL	SCR_BA+0x130	R/W	BOR 控制寄存器	0x0000_0005
PDSLEEPCFG	SCR_BA+0x200	RW	深度睡眠模式掉电配置寄存器	0x0000_0000
PDAWAKECFG	SCR_BA+0x204	RW	深度睡眠模式唤醒配置寄存器	0x0000_0000
PDRUNCFG	SCR_BA+0x208	RW	运行时掉电配置寄存器	0x0000_01e3
DEVICEID	SCR_BA+0x220	RW	芯片 ID 寄存器	0x0005_0005
DEVICEID1	SCR_BA+0x230	RW	芯片 ID 寄存器	0x0006_0006
SYSCTRL	SCR_BA+0x230	RW	系统杂项控制寄存器	0x0000_0000
PCON	SCR_BA+0x280	RW	功耗控制寄存器	0x0000_0000
ANACTRL	SCR_BA+0x290	RW	模拟电路调试寄存器	0x0000_0000
ANACTRL1	SCR_BA+0x300	RW	模拟电路调试寄存器 1	0x0000_0000
DEEPSLEEPCTRL	SCR_BA+0x310	RW	深度睡眠模块时钟控制寄存器	0x001f_ffff

6.2.7.1 PRESETCTRL 寄存器

该寄存器用来软件复位芯片中的外设模块。设为 0 可以复位相应的模块，设为 1 则复位失效。

同时这个寄存器有三个 set/clear/toggle 影子寄存器，往 set 寄存器中任意位写 1 会将主寄存器的对应位置 1，往 clear 寄存器的任意位置 1 会将主寄存器的对应位清 0，往 toggle 寄存器的任意位写 1 会将主寄存器的对应位翻转。这三个影子寄存器的地址分为别主寄存器地址加上 4/8/C。

寄存器	偏移量	R/W	描述	复位值
PRESETCTRL	SCR_BA+0x00	R/W	外设复位控制寄存器	0x0001_ffff

31	30	29	28	27	26	25	24
保留	保留	保留	保留	保留	保留	保留	保留
23	22	21	20	19	18	17	16
保留	保留	保留	CRS	USB	CAN	STIMER	保留

15	14	13	12	11	10	9	8
ACMP	保留	TSC	IWDG	保留	TIMER0	SPI0	MPWM
7	6	5	4	3	2	1	0
WDT	IOCONFIG	ADC	I2C0	UART0	GPIO	保留	FLASH

Bits	描述	
[31：21]	保留	保留位
[20]	CRS	**CRS 模块复位** 将该位置 0 将产生一个复位信号送到 CRS 模块，用户需要再将该位置 1 才能退出复位状态 0 = 复位 CRS 模块 1 = CRS 模块正常工作
[19]	USB	**USB 模块复位** 将该位置 0 将产生一个复位信号送到 USB 模块，用户需要再将该位置 1 才能退出复位状态 0 = 复位 USB 模块 1 = USB 模块正常工作
[18]	CAN	**CAN 模块复位** 将该位置 0 将产生一个复位信号送到 CAN 模块，用户需要再将该位置 1 才能退出复位状态 0 = 复位 CAN 模块 1 = CAN 模块正常工作
[17]	STIMER	**简单 TIMER 模块复位** 将该位置 0 将产生一个复位信号送到 STIMER 模块，用户需要再将该位置 1 才能退出复位状态 0 = 复位 STIMER 模块 1 = STIMER 模块正常工作
[16]	保留	保留位
[15]	ACMP	比较器模块复位

		将该位置 0 将产生一个复位信号送到比较器模块，用户需要再将该位置 1 才能退出复位状态 0 = 复位比较器模块 1 = 比较器模块正常工作
[14]	保留	保留位
[13]	TSC	TSC 模块复位 将该位置 0 将产生一个复位信号送到 TSC 模块，用户需要再将该位置 1 才能退出复位状态 0 = 复位 TSC 模块 1 = TSC 模块正常工作
[12]	IWDG	IWDG 模块复位 将该位置 0 将产生一个复位信号送到 IWDG 模块，用户需要再将该位置 1 才能退出复位状态 0 = 复位 IWDG 模块 1 = IWDG 模块正常工作
[11]	保留	保留位
[10]	TIMER0	TIMER0 模块复位 将该位置 0 将产生一个复位信号送到 TIMER0 模块，用户需要再将该位置 1 才能退出复位状态 0 = 复位 TIMER0 模块 1 = TIMER0 模块正常工作
[9]	SPI0	SPI0 模块复位 将该位置 0 将产生一个复位信号送到 SPI0 模块，用户需要再将该位置 1 才能退出复位状态 0 = 复位 SPI0 模块 1 = SPI0 模块正常工作
[8]	MPWM	Motor PWM 模块复位 将该位置 0 将产生一个复位信号送到 Motor PWM 模块，用户需要再将该位置 1 才能退出复位状态

		0 = 复位 Motor PWM 模块 1 = Motor PWM 模块正常工作
[7]	WDT	**Watch Dog 模块复位** 将该位置 0 将产生一个复位信号送到 Watch Dog 模块，用户需要再将该位置 1 才能退出复位状态 0 = 复位 Watch Dog 模块 1 = Watch Dog 模块正常工作
[6]	IOCONFIG	**IOConfig 模块复位** 将该位置 0 将产生一个复位信号送到 IOConfig 模块，用户需要再将该位置 1 才能退出复位状态 0 = 复位 IOConfig 模块 1 = IOConfig 模块正常工作
[5]	ADC	**ADC 模块复位** 将该位置 0 将产生一个复位信号送到 ADC 模块，用户需要再将该位置 1 才能退出复位状态 0 = 复位 ADC 模块 1 = ADC 模块正常工作
[4]	I2C0	**I2C0 模块复位** 将该位置 0 将产生一个复位信号送到 I2C0 模块，用户需要再将该位置 1 才能退出复位状态 0 = 复位 I2C0 模块 1 = I2C0 模块正常工作
[3]	UART0	**UART0 模块复位** 将该位置 0 将产生一个复位信号送到 UART0 模块，用户需要再将该位置 1 才能退出复位状态 0 = 复位 UART0 模块 1 = UART0 模块正常工作
[2]	GPIO	**GPIO 模块复位** 将该位置 0 将产生一个复位信号送到 GPIO 模块，用户需要再将

		该位置 1 才能退出复位状态 0 = 复位 GPIO 模块 1 = GPIO 模块正常工作
[1]	保留	保留
[0]	FLASH	FLASH 模块复位 将该位置 0 将产生一个复位信号送到 FLASH 模块，用户需要再将该位置 1 才能退出复位状态 0 = 复位 FLASH 模块 1 = FLASH 模块正常工作

6.2.7.2 AHBCLKCTRL 寄存器

该寄存器用来使能系统和各个外设的时钟。该寄存器和 PRESETCTRL 寄存器一样具有 set/clear/toggle 影子寄存器,地址分别为主寄存器地址加上 4/8/C。

将寄存器的各个位设为 1 时将使能对应模块的时钟，设为 0 时禁止对应模块的时钟。芯片在运行时通过将不需要的功能模块的时钟关闭可以起到节省动态功耗的作用。

寄存器	偏移量	R/W	描述	复位值
AHBCLKCTRL	SCR_BA+0x10	R/W	外设复位控制寄存器	0x0000_1087

31	30	29	28	27	26	25	24
保留	保留	保留	保留	保留	保留	保留	保留
23	22	21	20	19	18	17	16
保留	保留	保留	CRS	USB	CAN	STIMER	保留
15	14	13	12	11	10	9	8
ACMP	保留	TSC	IWDG	保留	TIMER0	SPI0	MPWM
7	6	5	4	3	2	1	0
WDT	IOCONFIG	ADC	I2C0	UART0	GPIO	保留	FLASH

Bits	描述	
[31:21]	保留	保留位
[20]	CRS	**CRS 模块时钟使能** 将该位置 1 将使能 CRS 模块的时钟，用户在使用 CRS 模块之前必须首先将该位置 1。 将该位置 0 将关闭 CRS 模块的时钟，CRS 模块无法工作。 0 = 关闭 CRS 模块时钟 1 = 打开 CRS 模块时钟
[19]	USB	**USB 模块时钟使能** 将该位置 1 将使能 USB 模块的时钟，用户在使用 USB 模块之前必须首先将该位置 1。 将该位置 0 将关闭 USB 模块的时钟，USB 模块无法工作。 0 = 关闭 USB 模块时钟 1 = 打开 USB 模块时钟
[18]	CAN	**CAN 模块时钟使能** 将该位置 1 将使能 CAN 模块的时钟，用户在使用 CAN 模块之前必须首先将该位置 1。 将该位置 0 将关闭 CAN 模块的时钟，CAN 模块无法工作。 0 = 关闭 CAN 模块时钟 1 = 打开 CAN 模块时钟
[17]	STIMER	**简单 TIMER 模块时钟使能** 将该位置 1 将使能 STIMER 模块的时钟，用户在使用 STIMER 模块之前必须首先将该位置 1。 将该位置 0 将关闭 STIMER 模块的时钟，STIMER 模块无法工作。 0 = 关闭 STIMER 模块时钟 1 = 打开 STIMER 模块时钟
[16]	保留	保留位
[15]	ACMP	**比较器模块时钟使能** 将该位置 1 将使能比较器模块的时钟，用户在使用比较器模块

		之前必须首先将该位置 1。 将该位置 0 将关闭比较器模块的时钟，比较器模块无法工作。 0 = 关闭比较器模块时钟 1 = 打开比较器模块时钟
[14]	保留	保留位
[13]	TSC	TSC 模块时钟使能 将该位置 1 将使能 TSC 模块的时钟，用户在使用 TSC 模块之前必须首先将该位置 1。 将该位置 0 将关闭 TSC 模块的时钟，TSC 模块无法工作。 0 = 关闭 TSC 模块时钟 1 = 打开 TSC 模块时钟位
[12]	IWDG	IWDG 模块时钟使能 将该位置 1 将使能 IWDG 模块的时钟，用户在使用 IWDG 模块之前必须首先将该位置 1。 将该位置 0 将关闭 IWDG 模块的时钟，IWDG 模块无法工作。 0 = 关闭 IWDG 模块时钟 1 = 打开 IWDG 模块时钟
[11]	保留	保留位
[10]	TIMER0	TIMER0 模块时钟使能 将该位置 1 将使能 TIMER0 模块的时钟，用户在使用 TIMER0 模块之前必须首先将该位置 1。 将该位置 0 将关闭 TIMER0 模块的时钟，TIMER0 模块无法工作。 0 = 关闭 TIMER0 模块时钟 1 = 打开 TIMER0 模块时钟
[9]	SPI0	SPI0 模块时钟使能 将该位置 1 将使能 SPI0 模块的时钟，用户在使用 SPI0 模块之前必须首先将该位置 1。 将该位置 0 将关闭 SPI0 模块的时钟，SPI0 模块无法工作。 0 = 关闭 SPI0 模块时钟

		1 = 打开 SPI0 模块时钟
[8]	MPWM	**Motor PWM 模块时钟使能** 将该位置 1 将使能 Motor PWM 模块的时钟,用户在使用 Motor PWM 模块之前必须首先将该位置 1。 将该位置 0 将关闭 Motor PWM 模块的时钟, Motor PWM 模块无法工作。 0 = 关闭 Motor PWM 模块时钟 1 = 打开 Motor PWM 模块时钟
[7]	WDT	**WatchDog 模块时钟使能** 将该位置 1 将使能 WatchDog 模块的时钟,用户在使用 WatchDog 模块之前必须首先将该位置 1。 将该位置 0 将关闭 WatchDog 模块的时钟,WatchDog 模块无法工作。 0 = 关闭 WatchDog 模块时钟 1 = 打开 WatchDog 模块时钟
[6]	IOCONFG	**IOConfig 模块时钟使能** 将该位置 1 将使能 IOConfig 模块的时钟,用户在使用 IOConfig 模块之前必须首先将该位置 1。 将该位置 0 将关闭 IOConfig 模块的时钟,IOConfig 模块无法工作。 0 = 关闭 IOConfig 模块时钟 1 = 打开 IOConfig 模块时钟
[5]	ADC	**ADC 模块时钟使能** 将该位置 1 将使能 ADC 模块的时钟, 用户在使用 ADC 模块之前必须首先将该位置 1。 将该位置 0 将关闭 ADC 模块的时钟, ADC 模块无法工作。 0 = 关闭 ADC 模块时钟 1 = 打开 ADC 模块时钟
[4]	I2C0	**I2C0 模块时钟使能**

		将该位置 1 将使能 I2C0 模块的时钟，用户在使用 I2C0 模块之前必须首先将该位置 1。 将该位置 0 将关闭 I2C0 模块的时钟，I2C0 模块无法工作。 0 = 关闭 I2C0 模块时钟 1 = 打开 I2C0 模块时钟
[3]	UART0	**UART0 模块时钟使能** 将该位置 1 将使能 UART0 模块的时钟，用户在使用 UART0 模块之前必须首先将该位置 1。 将该位置 0 将关闭 UART0 模块的时钟，UART0 模块无法工作。 0 = 关闭 UART0 模块时钟 1 = 打开 UART0 模块时钟
[2]	GPIO	**GPIO 模块时钟使能** 将该位置 1 将使能 GPIO 模块的时钟，用户在使用 GPIO 模块之前必须首先将该位置 1。 将该位置 0 将关闭 GPIO 模块的时钟，GPIO 模块无法工作。 0 = 关闭 GPIO 模块时钟 1 = 打开 GPIO 模块时钟
[1]	RAM	**RAM 模块时钟使能** 将该位置 1 将使能 RAM 模块的时钟，用户在使用 RAM 模块之前必须首先将该位置 1。 将该位置 0 将关闭 RAM 模块的时钟，RAM 模块无法工作。 0 = 关闭 RAM 模块时钟 1 = 打开 RAM 模块时钟
[0]	FLASH	**FLASH 模块时钟使能** 将该位置 1 将使能 FLASH 模块的时钟，用户在使用 FLASH 模块之前必须首先将该位置 1。 将该位置 0 将关闭 FLASH 模块的时钟，FLASH 模块无法工作。 0 = 关闭 FLASH 模块时钟 1 = 打开 FLASH 模块时钟

6.2.7.3 BOOTREMAP 寄存器

该寄存器用来标记芯片是否进入了地址 REMAP 模式以及程序保护模式。

在正常模式下，芯片的 Main Flash 空间被映射到 0x0000000 开始的地址空间，上电后执行的是 Main Flash 中的代码。

在地址重映射模式下，可以设置将 Main Flash 的任意地址开始的 256Byte 的空间映射到芯片 0x00000000 开始的地址空间。

寄存器	偏移量	R/W	描述	复位值
BOOTREMAP	SCR_BA+0x20	RO	重映射及程序保护状态寄存器	0x0000_0000

31	30	29	28	27	26	25	24
保留							
23	22	21	20	19	18	17	16
保留							
15	14	13	12	11	10	9	8
保留							
7	6	5	4	3	2	1	0
保留				PROTECT1	PROTECT0	REMAP	保留

Bits	描述	
[31:4]	保留	保留
[3]	PROTECT1	**标记芯片是否处于 PROTECT1 保护模式** 1：表示芯片处于 protect1 保护模式，此时通过 SWD 接口访问 Main Flash 中数据的行为被禁止，除非首先对 Flash 进行 main 擦除操作。（注意:此种模式下 SWD_CLK 管脚上需外部上拉或则下拉） 0：表示芯片不处于 protect1 模式
[2]	PROTECT2	**标记芯片是否处于 PROTECT2 保护模式** 1：表示芯片处于 protect2 保护模式，此时无法通过 SWD 接口

		连接芯片，也无法通过调试模式读取 Flash 中的代码。 0：表示芯片不处于 protect2 模式 一旦设置 PROTECT2 模式，芯片将无法再通过 SWD 进行连接，需要谨慎使用
[1]	REMAP	**标记芯片是否由于设置了 REMAPFLASH 寄存器中相应位而进入 REMAP 模式** 该位为 1 表示芯片进入了地址重映射模式，触发进入地址重映射模式的原因是设置了 REMAPFLASH 寄存器。
[0]	保留	保留

6.2.7.4 REMAPFLASH 寄存器

该寄存器用来设置芯片的地址重映射功能，可以将 Main Flash 空间中任意地址开始的 256Byte 的空间映射到 0x00000000 地址开始的 256Byte 地址空间。

寄存器	偏移量	R/W	描述	复位值
REMAPFLASH	SCR_BA+0x24	R/W	芯片地址重映射控制寄存器	0x0000_0000

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

Bits	描述	
[31:16]	WRITE_PROTECT	写保护位，仿制对该寄存器进行误操作

		要设置 REMAP_ADDR 的值，需要往 WRITE_PROTECT 位写 0x5555 如果已进入 REMAP 模式，往 WRITE_PROTECT 位写 0xAAAA 可以退出 REMAP 模式，同时将 REAMP_ADDR 域清零。
[15:0]	REMAP_ADDR	可将 REMAP_ADDR 中的地址空间映射到 0x00000000 地址开始的 256Byte 地址空间。 必须写 REMAP_ADDR 位的同时，WRITE_PROTECT 位为 0x5555 时，芯片才会进如 REMAP 模式。

6.2.7.5 12OSC_CTRL 寄存器

该寄存器用来设置与内部 12M IRC 以及外部 12M OSC 相关的控制参数。

寄存器	偏移量	R/W	描述	复位值
12OSC_CTRL	SCR_BA+0x28	R/W	12M 时钟控制寄存器	0x0000_0080

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

Bits	描述	
[31:18]	保留	保留
[17]	OSC_EO	设为 1 来使能外部 12MHZ 晶体振荡器的输出端口 要使用外部 12MHZ 的晶体振荡器之前必须首先将该位置 1 0 = 禁能外部 12MHZ 晶体振荡器

		1 = 使能外部 12MHZ 晶体振荡器 注意：要想使用 12M 晶振作为系统的时钟源，还必须打开 SYSPLLCTRL 寄存器的 bit30
[16]	OSC_EI	**设为 1 来使能外部 12MHZ 晶体振荡器的输入端口** 要使用外部 12MHZ 的晶体振荡器之前必须首先将该位置 1 0 = 禁能外部 12MHZ 晶体振荡器 1 = 使能外部 12MHZ 晶体振荡器 注意：要想使用 12M 晶振作为系统的时钟源，还必须打开 SYSPLLCTRL 寄存器的 bit30
[15:8]	保留	保留
[7:0]	IRC_ADJ	**用来调节内部 12MHZ 的 IRC 的频率** 值设的越大，IRC 的输出频率就越高。***不要去改变这个参数的值***，IRC 的频率在芯片出厂时已经进行过校准

6.2.7.6 25KIRC_CTRL 寄存器

该寄存器用来设置与内部 25K IRC 相关的控制参数。

寄存器	偏移量	R/W	描述	复位值
25KIRC_CTRL	SCR_BA+0x2C	R/W	25K RC 振荡器控制寄存器	0x0000_0008

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

Bits	描述	
[31:4]	保留	保留
[3:0]	IRC_ADJ	**用来调节内部 25KHZ 的 IRC 的频率** 值设的越大，IRC 的输出频率就越高。***不要去改变这个参数的值***，IRC 的频率在芯片出厂时已经进行过校准

6.2.7.7 SYSPLLCTRL 寄存器

该寄存器用来设置系统 PLL 的相关参数。

寄存器	偏移量	R/W	描述	复位值
SYSPLLCTRL	SCR_BA+0x30	R/W	系统 PLL 控制寄存器	0x8000_0002

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

Bits	描述	
[31]	保留	保留
[30]	REF_SEL	**选择系统 PLL 以及 12M 晶振信号的输入时钟源** 0：时钟源为内部 12M IRC 1：时钟源为外部 12M OSC 注意：若是系统时钟源想要选择 12M OSC，无论使不使用 PLL，这个位都必须配置为 1
[29:3]	保留	保留

[2:0]	FREQ	**设置系统 PLL 的输出频率** Freq = 12*(N+4)　N 为 FREQ 的值

6.2.7.8 SYSPLLSTAT 寄存器

该寄存器用来反映系统 PLL 的状态。

寄存器	偏移量	R/W	描述	复位值
SYSPLLSTAT	SCR_BA+0x34	R	系统 PLL 控制寄存器	0x0000_0001

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

Bits	描述	
[31:1]	保留	保留
[0]	LOCK	**反映系统 PLL 的状态** 1：系统 PLL 锁定，可以切换时钟到 PLL 0：系统 PLL 未锁定，不可以切换时钟到 PLL

6.2.7.9 LDOCTRL 寄存器

该寄存器用来调节内部 1.5V LDO 的输出电压。

寄存器	偏移量	R/W	描述	复位值
LDOCTRL	SCR_BA+0x38	R/W	LDO 控制寄存器	0x0008_0008

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

Bits	描述	
[31:8]	保留	保留
[7:4]	LOWPOWER_LDO_ADJ	**调节内部低功耗 1.5V LDO 输出电压** 值设的越大，LOWPOWER LDO 输出电压就越高。
[3:0]	LDO_ADJ	**调节内部 1.5V 主 LDO 输出电压** 值设的越大，LDO 输出电压就越高。***不要去改变这个参数的值***，LDO 电压输出值在芯片出厂时已经进行过校准

6.2.7.10 SYSRSTSTAT 寄存器

该寄存器显示最新的复位事件源。寄存器中的任何位均可以通过写 1 来清零。POR 发生时会将该寄存器中除了 POR 位以外的所有其他位清零。但是如果另一复位源事件在 POR 结束后仍然存在，那么它的对应位会被设置成 1。

寄存器	偏移量	R/W	描述	复位值
SYSRSTSTAT	SCR_BA+0x40	R/W	**系统复位源标识寄存器**	0x0000_0001

31	30	29	28	27	26	25	24
保留							
23	22	21	20	19	18	17	16
保留							

15	14	13	12	11	10	9	8
保留							
7	6	5	4	3	2	1	0
保留		IWDGRST	SYSRST	BORRST	WDTRST	EXTRST	PORRST

Bits	描述	
[31:6]	保留	保留
[5]	IWDGRST	1：表示发生了独立看门狗复位
[4]	SYSRST	1：表示发生了系统复位
[3]	BORRST	1：表示发生了低电压检测复位
[2]	WDTRST	1：表示发生了看门狗复位
[1]	EXTRST	1：表示发生了外部按键复位
[0]	PORRST	1：表示发生了上电复位

6.2.7.11 MAINCLKSEL 寄存器

该寄存器用于选择主系统时钟，主系统时钟可以由内部 IRC，外部 OSC 或系统 PLL 提供。主系统时钟为内核、外设、存储器模块提供时钟信号。

寄存器	偏移量	R/W	描述	复位值
MAINCLKSEL	SCR_BA+0x50	R/W	系统主时钟源选择寄存器	0x0000_0000

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

保留	SEL

Bits	描述	
[31:4]	保留	保留
[3:0]	SEL	MAINCLK 时钟源选择 xx00: 内部 12M IRC 作为时钟源(xx 不等于 11) xx01：系统 PLL 作为时钟源(xx 不等于 11) xx10：外部 12M OSC 作为时钟源(xx 不等于 11) xx11：内部 25K IRC 作为时钟源(xx 不等于 11) 11xx：外部 32K OSC 作为时钟源 改变该位不会立即切换时钟源，需要等到 MAINCLKUEN 寄存器中的 ENA 位发生从 0 到 1 的变化时才会完成时钟源切换 注意：若是想要使用 12M OSC 作为时钟源，必须首先配置 SYSPLLCTRL 寄存器中的 REF_SEL 位为 1

6.2.7.12 MAINCLKUEN 寄存器

在 MAINCLKSEL 寄存器被编程后，必须先写 0 到 MAINCLKUEN 寄存器，然后再写 1 到 MAINCLKUEN 才会使 MAINCLK 时钟源的更新生效。

寄存器	偏移量	R/W	描述	复位值
MAINCLKUEN	SCR_BA+0x54	R/W	系统主时钟源更新寄存器	0x0000_0000

31	30	29	28	27	26	25	24
保留							
23	22	21	20	19	18	17	16
保留							
15	14	13	12	11	10	9	8
保留							

7	6	5	4	3	2	1	0
保留							ENA

Bits	描述	
[31:1]	保留	保留
[0]	ENA	**使能 MAINCLK 时钟源的切换** 从 0 到 1 的变化会使能 MAINCLK 时钟源的切换 要切换 MAINCLK 的时钟源，首先在 MAINCLKSEL 寄存器中配置正确的值，然后将 ENA 置 0，再将 ENA 置 1，才会完成整个时钟切换过程。

6.2.7.13 USARTCLKSEL 寄存器

该寄存器用于选择 USART 模块时钟，USART 模块时钟可以由内部 IRC，外部 OSC 或系统 PLL 提供。UART 时钟为内核、外设、存储器 UART 模块提供 PCLK 时钟信号。

寄存器	偏移量	R/W	描述	复位值
USARTCLKSEL	SCR_BA+0x58	R/W	USART 时钟源选择寄存器	0x0000_0000

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

Bits	描述	
[31:2]	保留	保留
[1:0]	SEL	USARTCLK 时钟源选择 00：内部 12M IRC 作为时钟源 01：系统 PLL 作为时钟源 10：外部 12M OSC 作为时钟源 11：内部 25K IRC 作为时钟源 改变该位不会立即切换时钟源，需要等到 USARTCLKUEN 寄存器中的 ENA 位发生从 0 到 1 的变化时才会完成时钟源切换

6.2.7.14 USARTCLKUEN 寄存器

在 USARTCLKSEL 寄存器被编程后，必须先写 0 到 USARTCLKUEN 寄存器，然后再写 1 到 USARTCLKUEN 才会使 USARTCLK 时钟源的更新生效。

寄存器	偏移量	R/W	描述	复位值
USARTCLKUEN	SCR_BA+0x5C	R/W	USART 时钟源更新寄存器	0x0000_0000

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

Bits	描述	
[31:1]	保留	保留

[0]	ENA	**使能 USARTCLK 时钟源的切换** 从 0 到 1 的变化会使能 USARTCLK 时钟源的切换 要切换 USARTCLK 的时钟源，首先在 USARTCLKSEL 寄存器中配置正确的值，然后将 ENA 置 0，再将 ENA 置 1，才会完成整个时钟切换过程。

6.2.7.15 FLASHCLKSEL 寄存器

该寄存器用于选择 FLASH 模块擦写时钟，该模块时钟可以由内部 12M IRC 或外部 12M OSC 提供。

寄存器	偏移量	R/W	描述	复位值
FLASHCLKSEL	SCR_BA+0x60	R/W	FLASH 时钟源选择寄存器	0x0000_0000

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

Bits	描述	
[31:2]	保留	保留
[1:0]	SEL	**FLASHCLK 时钟源选择** 0x：内部 12M IRC 作为时钟源 10：外部 12M OSC 作为时钟源 11：OUTCLK 作为时钟源 改变该位不会立即切换时钟源，需要等到 FLASHCLKUEN 寄存器中

		的 ENA 位发生从 0 到 1 的变化时才会完成时钟源切换

6.2.7.16 FLASHCLKUEN 寄存器

在 FLASHCLKSEL 寄存器被编程后，必须先写 0 到 FLASHCLKUEN 寄存器，然后再写 1 到 FLASHCLKUEN 才会使 FLASHCLK 时钟源的更新生效。

寄存器	偏移量	R/W	描述	复位值
FLASHCLKUEN	SCR_BA+0x64	R/W	FLASH 时钟源更新寄存器	0x0000_0000

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

Bits	描述	
[31:1]	保留	保留
[0]	ENA	**使能 FLASHCLK 时钟源的切换** 从 0 到 1 的变化会使能 FLASHCLK 时钟源的切换 要切换 FLASHCLK 的时钟源，首先在 FLASHCLKSEL 寄存器中配置正确的值，然后将 ENA 置 0，再将 ENA 置 1，才会完成整个时钟切换过程。

6.2.7.17 OUTCLKSEL 寄存器

该寄存器用于选择 OUT 管脚输出的时钟，OUT 管脚输出的时钟可以由内部 IRC，系统 PLL 或外部 OSC 提供。OUT 管脚输出的时钟为 OUT 时钟输出管脚提供参考时钟。

寄存器	偏移量	R/W	描述	复位值
OUTCLKSEL	SCR_BA+0x70	R/W	OUT 时钟源选择寄存器	0x0000_0000

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

Bits	描述	
[31:4]	保留	保留
[3:0]	SEL	OUTCLK 时钟源选择（同时作为 STIMER 模块的计数时钟源） xx00: 内部 12M IRC 作为时钟源(xx 不等于 11) xx01：系统 PLL 作为时钟源(xx 不等于 11) xx10：内部 25K IRC 作为时钟源(xx 不等于 11) xx11：外部 12M OSC 作为时钟源(xx 不等于 11) 11xx：外部 32K OSC 作为时钟源 改变该位不会立即切换时钟源，需要等到 OUTCLKUEN 寄存器中的 ENA 位发生从 0 到 1 的变化时才会完成时钟源切换

6.2.7.18 OUTCLKUEN 寄存器

在 OUTCLKSEL 寄存器被编程后，必须先写 0 到 OUTCLKUEN 寄存器，然后再写 1 到 OUTCLKUEN 才会使 OUTCLK 时钟源的更新生效。

寄存器	偏移量	R/W	描述	复位值
OUTCLKUEN	SCR_BA+0x74	R/W	OUT 时钟源更新寄存器	0x0000_0000

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

Bits	描述	
[31:1]	保留	保留
[0]	ENA	**使能 OUTCLK 时钟源的切换** 从 0 到 1 的变化会使能 OUTCLK 时钟源的切换 要切换 OUTCLK 的时钟源，首先在 OUTCLKSEL 寄存器中配置正确的值，然后将 ENA 置 0，再将 ENA 置 1，才会完成整个时钟切换过程。

6.2.7.19 SYSAHBCLKDIV 寄存器

该寄存器对主系统时钟(MAINCLK)进行分频，然后向内核、存储器以及外设提供系统时钟。可以通过设置 DIV 位为 0x0 完全关闭系统时钟。（注意：一旦设为 0 则系统时钟被关闭，同时 CPU 将无法再配置该寄存器，导致系统无法继续工作）

寄存器	偏移量	R/W	描述	复位值
SYSAHBCLKDIV	SCR_BA+0x80	R/W	AHB 时钟分频寄存器	0x0000_0002

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

Bits	描述	
[31:16]	保留	保留
[15:0]	DIV	系统 AHB 时钟的分频系数 0：关闭 AHB 时钟 1~65536：AHBCLK = MAINCLK/DIV

6.2.7.20 USART0CLKDIV 寄存器

该寄存器对 USART 模块时钟进行分频，然后向 USART0 模块提供参考时钟。可以通过设置 DIV 位为 0x0 完全关闭 USART0 时钟。

寄存器	偏移量	R/W	描述	复位值
USART0CLKDIV	SCR_BA+0x90	R/W	USART0 时钟分频寄存器	0x0000_0010

31	30	29	28	27	26	25	24
保留							
23	22	21	20	19	18	17	16

保留							
15	14	13	12	11	10	9	8
保留							
7	6	5	4	3	2	1	0
DIV							

Bits	描述	
[31:8]	保留	保留
[7:0]	DIV	USART0 时钟的分频系数 0：关闭 USART0 时钟 1~255：USART0_CLK = USARTCLK/DIV

6.2.7.21 SYSTICKCLKDIV 寄存器

该寄存器对 SYSTICKCLK 时钟进行分频，然后向 CPU 模块提供 SYSTICKCLK 时钟。可以通过设置 DIV 位为 0x0 完全关闭 SYSTICKCLK 时钟。

寄存器	偏移量	R/W	描述	复位值
SYSTICKCLKDIV	SCR_BA+0xA0	R/W	WDT 模块时钟分频寄存器	0x0000_0010

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

Bits	描述

[31:8]	保留	保留
[7:0]	DIV	SYSTICKCLK 时钟的分频系数 0：关闭 SYSTICKCLK_CLK 时钟 1~255：SYSTICKCLK = MAINCLK/DIV

6.2.7.22 SPI0CLKDIV 寄存器

该寄存器对 SPI0CLK 时钟进行分频，然后向 SPI0 模块提供 SPI0_CLK 时钟。可以通过设置 DIV 位为 0x0 完全关闭 SPI0_CLK 时钟。

寄存器	偏移量	R/W	描述	复位值
SPI0CLKDIV	SCR_BA+0xA4	R/W	SPI0 模块时钟分频寄存器	0x0000_0010

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

Bits	描述	
[31:8]	保留	保留
[7:0]	DIV	SPI0CLK 时钟的分频系数 0：关闭 SPI0_CLK 时钟 1~255：SPI0_CLK = MAINCLK/DIV

6.2.7.23 OUTCLKDIV 寄存器

该寄存器对 OUTCLK 时钟进行分频，然后向 OUT 管脚提供 OUTCLK_CLK 时钟。可以通过设置 DIV 位为 0x0 完全关闭 OUTCLK_CLK 时钟。

寄存器	偏移量	R/W	描述	复位值
OUTCLKDIV	SCR_BA+0xAC	R/W	OUTCLK 时钟分频寄存器	0x0000_0001

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

Bits	描述	
[31:8]	保留	保留
[7:0]	DIV	OUTCLK 时钟的分频系数 0：关闭 OUTCLK_CLK 时钟 1~255：OUTCLK_CLK = OUTCLK/DIV

6.2.7.24 ADCANACLKDIV 寄存器

该寄存器对 MAINCLK 时钟进行分频,然后向 ADC 模块提供 ADC 的采样时钟源。可以通过设置 DIV 位为 0x0 完全关闭 ADCANA_CLK 时钟。

寄存器	偏移量	R/W	描述	复位值
ADCANACLKDIV	SCR_BA+0xB0	R/W	ADC 采样时钟分频寄存器	0x0000_0001

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

Bits	描述	
[31:8]	保留	保留
[7:0]	DIV	ADCANACLK 时钟的分频系数 0：关闭 ADCANA_CLK 时钟 1~255：ADCANA_CLK = MAINCLK/DIV

6.2.7.25 USBCLKDIV 寄存器

该寄存器对 PLL 时钟进行分频，然后向 USB 模块提供 48M 时钟源。可以通过设置 DIV 位为 0x0 完全关闭 USBCLK 时钟。

寄存器	偏移量	R/W	描述	复位值
USBCLKDIV	SCR_BA+0xC0	R/W	USB 时钟分频寄存器	0x0000_0001

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

保留	DIV

Bits	描述	
[31:2]	保留	保留
[1:0]	DIV	USBCLK 时钟的分频系数 0：关闭 USBCLK 时钟 1~3：USBCLK = PLLCLK/DIV 要使用 USB 模块，PLL 必须配置成 48MHz 或 96MHz，同时 DIV 域配置为 1 或 2。

6.2.7.26 WAKEUPEN 寄存器

该寄存器用来使能 WAKEUP 管脚在芯片处于掉电模式下的唤醒功能。

寄存器	偏移量	R/W	描述	复位值
WAKEUPEN	SCR_BA+0xE0	R/W	WAKEUP 管脚使能寄存器	0x0000_0000

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

Bits	描述	
[31:8]	保留	保留
[7:0]	EN	设置 7 个 WAKEUP 管脚的掉电唤醒功能 0：关闭 WAKEUP 管脚的掉电唤醒功能

		1：使能 WAKEUP 管脚的掉电唤醒功能 芯片有 7 个 IO 可以在掉电模式下用来唤醒芯片，在进入掉电模式前需要将对应的 EN 位置 1 才能打开该 IO 的唤醒功能。 Bit7：控制 PIO0_0（NRST）的唤醒功能 Bit6：控制 PIO2_0 的唤醒功能 Bit5：控制 PIO1_7 的唤醒功能 Bit4：控制 PIO1_6 的唤醒功能 Bit3：控制 PIO1_5 的唤醒功能 Bit2：保留，必须为 0。 Bit1：控制 PIO1_1 的唤醒功能 Bit0：控制 PIO1_0 的唤醒功能

6.2.7.27 SYSTICKCAL 寄存器

该寄存器用来设置 CPU 的 systick 时钟的校准值。

寄存器	偏移量	R/W	描述	复位值
SYSTICKCAL	SCR_BA+0x120	R/W	SYSTICK 校准值寄存器	0x0000_0000

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

Bits	描述

[31:26]	保留	保留
[25:0]	CAL	SYSTICK 时钟的校准值

6.2.7.28 BORCTRL 寄存器

该寄存器用来控制系统低电压复位以及低电压检测的相关参数。

寄存器	偏移量	R/W	描述	复位值
BORCTRL	SCR_BA+0x130	R/W	BOR/BOD 控制寄存器	0x0000_0005

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

Bits	描述	
[31:17]	保留	保留
[16:15]	BODLEV	**设置系统低电压检测中断的阈值** 00：2.2V 产生 BOD 中断 01：2.7V 产生 BOD 中断 10：3.7V 产生 BOD 中断 11：4.3V 产生 BOD 中断 该位用来设置系统何时产生低电压中断。当系统电压低于上述设置的阈值时，会产生一个 BOD 中断给 CPU。
[2]	保留	保留

[1:0]	BORLEV	**设置系统低电压复位的阈值** 00：1.9V 产生 BOR 复位 01：2.3V 产生 BOR 复位 10：2.8V 产生 BOR 复位 11：3.8V 产生 BOR 复位 该位用来设置系统何时产生低电压复位。当系统电压低于上述设置的阈值时，会产生一个 BOR 复位来复位系统。 要使用 BOR 复位功能必须首先在 PDRUNCFG 寄存器中打开 BOR 模块电源

6.2.7.29 PDSLEEPCFG 寄存器

该寄存器的位可以由软件进行编程，指示芯片在内核进入深度睡眠模式时要将哪些模拟模块的电源关闭。进入深度睡眠模式后，PDSLEEPCFG 寄存器的值会自动加载到 PDRUNCFG 寄存器中。

寄存器	偏移量	R/W	描述	复位值
PDSLEEPCFG	SCR_BA+0x200	RW	深度睡眠模式掉电配置寄存器	0x0000_0000

31	30	29	28	27	26	25	24
保留							
23	22	21	20	19	18	17	16
保留							
15	14	13	12	11	10	9	8
保留							OSC32K
7	6	5	4	3	2	1	0
USB	保留	SYSPLL	FLASH	IRC12M	BOR	BOD	ADC

Bits	描述	
[31:9]	保留	保留

8	OSC32K	**设置芯片进入深度睡眠模式时是否需要关闭 32K 晶振电源** 0：表示不关闭电源 1：表示关闭电源
7	USB	**设置芯片进入深度睡眠模式时是否需要关闭 USB 模块电源** 0：表示不关闭电源 1：表示关闭电源
6	**保留**	**保留**
5	SYSPLL	**设置芯片进入深度睡眠模式时是否需要关闭 SYSPLL 模块电源** 0：表示不关闭电源 1：表示关闭电源
4	FLASH	**设置芯片进入深度睡眠模式时是否需要关闭 FLASH 模块电源** 0：表示不关闭电源 1：表示关闭电源
3	IRC12M	**设置芯片进入深度睡眠模式时是否需要关闭 12M IRC 模块电源** 0：表示不关闭电源 1：表示关闭电源
2	BOR	**设置芯片进入深度睡眠模式时是否需要关闭 BOR 模块电源** 0：表示不关闭电源 1：表示关闭电源
1	BOD	**设置芯片进入深度睡眠模式时是否需要关闭 BOD 模块电源** 0：表示不关闭电源 1：表示关闭电源
0	ADC	**设置芯片进入深度睡眠模式时是否需要关闭 ADC 模块电源** 0：表示不关闭电源 1：表示关闭电源

6.2.7.30 PDAWAKECFG 寄存器

该寄存器可以通过软件编程，指示芯片在从深度睡眠模式唤醒时要进入的状

态。在从深度睡眠模式推出后，PDAWAKECFG 寄存器的值会自动加载到 PDRUNCFG 寄存器中。

寄存器	偏移量	R/W	描述	复位值
PDAWAKECFG	SCR_BA+0x204	RW	深度睡眠模式唤醒配置寄存器	0x0000_0000

31	30	29	28	27	26	25	24
保留							
23	22	21	20	19	18	17	16
保留							
15	14	13	12	11	10	9	8
保留							OSC32K
7	6	5	4	3	2	1	0
USB	保留	SYSPLL	FLASH	IRC12M	BOR	BOD	ADC

Bits	描述	
[31:9]	保留	保留
8	OSC32K	设置芯片退出深度睡眠模式时是否需要关闭 32K 晶振电源 0：表示不关闭电源 1：表示关闭电源
7	USB	设置芯片退出深度睡眠模式时是否需要关闭 USB 模块电源 0：表示不关闭电源 1：表示关闭电源
6	保留	保留
5	SYSPLL	设置芯片退出深度睡眠模式时是否需要关闭 SYSPLL 模块电源 0：表示不关闭电源 1：表示关闭电源
4	FLASH	设置芯片退出深度睡眠模式时是否需要关闭 FLASH 模块电源 0：表示不关闭电源 1：表示关闭电源

3	IRC12M	**设置芯片退出深度睡眠模式时是否需要关闭 12M IRC 模块电源** 0：表示不关闭电源 1：表示关闭电源
2	BOR	**设置芯片退出深度睡眠模式时是否需要关闭 BOR 模块电源** 0：表示不关闭电源 1：表示关闭电源
1	BOD	**设置芯片退出深度睡眠模式时是否需要关闭 BOD 模块电源** 0：表示不关闭电源 1：表示关闭电源
0	ADC	**设置芯片退出深度睡眠模式时是否需要关闭 ADC 模块电源** 0：表示不关闭电源 1：表示关闭电源

6.2.7.31 PDRUNCFG 寄存器

该寄存器的位控制着各个模拟模块的供电。芯片运行时，可在随时对该寄存器进行写操作，每次写操作都会立即生效。

寄存器	偏移量	R/W	描述	复位值
PDRUNCFG	SCR_BA+0x208	RW	运行时掉电配置寄存器	0x0000_01e3

31	30	29	28	27	26	25	24
保留							
23	22	21	20	19	18	17	16
保留							
15	14	13	12	11	10	9	8
保留							OSC32K
7	6	5	4	3	2	1	0
USB	保留	SYSPLL	FLASH	IRC12M	BOR	BOD	ADC

Bits	描述	
[31:9]	保留	保留
8	OSC32K	设置芯片运行时是否需要关闭 32K 晶振电源 0：表示不关闭电源 1：表示关闭电源
7	USB	设置芯片运行时是否需要关闭 USB 模块电源 0：表示不关闭电源 1：表示关闭电源
6	保留	保留
5	SYSPLL	设置芯片运行时是否需要关闭 SYSPLL 模块电源 0：表示不关闭电源 1：表示关闭电源
4	FLASH	设置芯片运行时是否需要关闭 FLASH 模块电源 0：表示不关闭电源 1：表示关闭电源
3	IRC12M	设置芯片运行时是否需要关闭 12M IRC 模块电源 0：表示不关闭电源 1：表示关闭电源
2	BOR	设置芯片运行时是否需要关闭 BOR 模块电源 0：表示不关闭电源 1：表示关闭电源
1	BOD	设置芯片运行时是否需要关闭 BOD 模块电源 0：表示不关闭电源 1：表示关闭电源
0	ADC	设置芯片运行时是否需要关闭 ADC 模块电源 0：表示不关闭电源 1：表示关闭电源

6.2.7.32 DEVICEID 寄存器

该寄存器用来读取芯片 ID。

寄存器	偏移量	R/W	描述	复位值
DEVICEID	SCR_BA+0x220	R	芯片 ID 寄存器	0x0005_0005

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

Bits	描述	
[31:0]	ID	芯片的 ID: 0x0005_0005

6.2.7.33 DEVICEID1 寄存器

该寄存器用来读取芯片 ID。

寄存器	偏移量	R/W	描述	复位值
DEVICEID1	SCR_BA+0x230	R	芯片 ID 寄存器 1	0x0006_0006

31	30	29	28	27	26	25	24
ID							
23	22	21	20	19	18	17	16
ID							

15	14	13	12	11	10	9	8
ID							
7	6	5	4	3	2	1	0
ID							

Bits	描述	
[31:0]	ID	芯片的 ID: 0x0006_0006 或 0x0005_0005(由 Flash NVR 决定)

6.2.7.34 PCON 寄存器

功率控制寄存器选择是否进入内核控制的其中某种掉电模式（睡眠模式或深度睡眠模式）或深度掉电模式，并分别为睡眠模式或深度睡眠模式和深度掉电模式提供标志。

寄存器	偏移量	R/W	描述	复位值
PCON	SCR_BA+0x280	RW	功耗控制寄存器	0x0010_0000

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

Bits	描述	
[31:21]	保留	保留
20	NORMAL	只读位，表示是否可以退出 LOWPOWER 状态

		1：表示可以退出 LOWPOWER 状态 0：表示不可以退出 LOWPOWER 状态
[19:17]	保留	保留
16	LOWPOWER	**切换系统到 LOWPOWER 模式** 1：表示将系统切换到低功耗模式 0：表示将系统从低功耗模式切换到 Normal 模式 *注意：低功耗模式的切换要严格按照流程进行，具体见系统功率控制章节描述*
[15:9]	保留	保留
8	SLEEP	**睡眠模式标志** 读操作下： 0：表示未进入睡眠模式，芯片工作于正常模式 1：表示进入睡眠、深度睡眠或掉电模式 写操作下： 0：无影响 1：将该位清 0
[7:2]	保留	保留
1	DPDEN	**使能深度掉电模式** 0：内核执行 WFI 指令时芯片进入睡眠或深度睡眠模式 1：内核执行 WFI 指令时芯片进入深度掉电模式
0	保留	保留

6.2.7.35 ANACTRL 寄存器

ANACTRL 控制寄存器用来控制模拟电路的一些参数。

寄存器	偏移量	R/W	描述	复位值
ANACTRL	SCR_BA+0x290	RW	ANA 控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
保留							
23	22	21	20	19	18	17	16

保留			DIS_NRST	保留			
15	14	13	12	11	10	9	8
	TEMP			SMALL		LARGE	
7	6	5	4	3	2	1	0
保留		WWDG	IWDG	保留			

Bits	描述	
[31:21]	保留	保留
20	DIS_NRST	1：NRST 管脚的复位功能被屏蔽，此时该管脚做为 PI00_0 功能使用 0：NRST 管脚做为外部复位管脚使用
[19:6]	保留	保留
5	WWDG	控制 WWDG 在 SWD 调试时是否停止 0：SWD 调试时，WWDG 停止计数 1：SWD 调试时，WWDG 保持计数
4	IWDG	控制 IWDG 在 SWD 调试时是否停止 0：SWD 调试时，IWDG 停止计数 1：SWD 调试时，IWDG 保持计数
[3:0]	保留	保留位

6.2.7.36 ANACTRL1 寄存器

ANACTRL1 控制寄存器用来控制模拟电路的一些参数。

寄存器	偏移量	R/W	描述	复位值
ANACTRL1	SCR_BA+0x300	RW	ANA 控制寄存器 1	0x0000_0000

31	30	29	28	27	26	25	24
保留							
23	22	21	20	19	18	17	16

保留							
15	14	13	12	11	10	9	8
保留		UART_INVT	UART_INVR	保留		ADC_REG	512B_MODE
7	6	5	4	3	2	1	0
保留		OP1_CHOP	OP0_CHOP	ADC_CHOP	ADC_BUF	CMP2PWM	

Bits	描述	
[31:14]	保留	保留
13	UART_INVT	1：UART_TX 信号极性取反 0：UART_TX 信号极性正常
12	UART_INVR	1：UART_RX 信号极性取反 0：UART_RX 信号极性正常
[11:10]	保留	保留
9	ADC_REG	0: ADC 的校准值（bvos 和 multi）使用 Flash NVR 中预设的值 1：ADC 的校准值寄存器可进行软件配置，并使用配置值
8	512B_MODE	0: Flash 的 sector 按 1K Byte 大小组织 1：Flash 的 sector 按 512 Byte 大小组织
[7:3]	保留	保留
2	ADC_BUF	0: 禁能 ADC 的 BUF 模式 1：使能 ADC 的 BUF 模式
[1:0]	CMP2PWM	00: MPWM 模块的 abort 信号连接到 abort_i 功能 IO 上 01：MPWM 模块的 abort 信号连接到比较器 0 的输出 10：MPWM 模块的 abort 信号连接到比较器 1 的输出 11：MPWM 模块的 abort 信号连接到比较器 0 的输出和比较器输出的与逻辑上

6.2.7.37 DEEPSLEEPCTRL 寄存器

该寄存器用来控制芯片在进入 deepsleep 模式时哪些模块的时钟不被自动关闭。芯片在进入 deepsleep 模式时会自动将绝大部分模块的 pclk 关闭，若用户想要某个模块的 pclk 不被自动关闭可将对应的 bit 设为 0。

寄存器	偏移量	R/W	描述	复位值
DEEPSLEEPCTRL	SCR_BA+0x310	R/W	深度睡眠时钟控制寄存器	0x001F_FFFF

31	30	29	28	27	26	25	24
保留	保留	保留	保留	保留	保留	保留	保留
23	22	21	20	19	18	17	16
保留	保留	保留	CRS	USB	CAN	STIMER	保留
15	14	13	12	11	10	9	8
ACMP	保留	TSC	IWDG	保留	TIMER0	SPI0	MPWM
7	6	5	4	3	2	1	0
WDT	IOCONFIG	ADC	I2C0	UART0	GPIO	保留	FLASH

Bits	描述	
[31:21]	保留	保留位
[20]	CRS	0 = 深度睡眠时不关闭 CRS 模块 PCLK 时钟 1 = 深度睡眠时自动关闭 CRS 模块 PCLK 时钟
[19]	USB	0 = 深度睡眠时不关闭 USB 模块 PCLK 时钟 1 = 深度睡眠时自动关闭 USB 模块 PCLK 时钟
[18]	CAN	0 = 深度睡眠时不关闭 CAN 模块 PCLK 时钟 1 = 深度睡眠时自动关闭 CAN 模块 PCLK 时钟
[17]	STIMER	0 = 深度睡眠时不关闭 STIMER 模块 PCLK 时钟 1 = 深度睡眠时自动关闭 STIMER 模块 PCLK 时钟
[16]	保留	保留位
[15]	ACMP	0 = 深度睡眠时不关闭 ACMP 模块 PCLK 时钟 1 = 深度睡眠时自动关闭 ACMP 模块 PCLK 时钟

[14]	保留	保留位
[13]	TSC	0 = 深度睡眠时不关闭 TSC 模块 PCLK 时钟 1 = 深度睡眠时自动关闭 TSC 模块 PCLK 时钟
[12]	IWDG	0 = 深度睡眠时不关闭 IWDG 模块 PCLK 时钟 1 = 深度睡眠时自动关闭 IWDG 模块 PCLK 时钟
[11]	保留	保留位
[10]	TIMER0	0 = 深度睡眠时不关闭 TIMER0 模块 PCLK 时钟 1 = 深度睡眠时自动关闭 TIMER0 模块 PCLK 时钟
[9]	SPI0	0 = 深度睡眠时不关闭 SPI0 模块 PCLK 时钟 1 = 深度睡眠时自动关闭 SPI0 模块 PCLK 时钟
[8]	MPWM	0 = 深度睡眠时不关闭 MPWM 模块 PCLK 时钟 1 = 深度睡眠时自动关闭 MPWM 模块 PCLK 时钟
[7]	WDT	0 = 深度睡眠时不关闭 WDT 模块 PCLK 时钟 1 = 深度睡眠时自动关闭 WDT 模块 PCLK 时钟
[6]	IOCONFG	0 = 深度睡眠时不关闭 IOCONFIG 模块 PCLK 时钟 1 = 深度睡眠时自动关闭 IOCONFIG 模块 PCLK 时钟
[5]	ADC	0 = 深度睡眠时不关闭 ADC 模块 PCLK 时钟 1 = 深度睡眠时自动关闭 ADC 模块 PCLK 时钟
[4]	I2C0	0 = 深度睡眠时不关闭 I2C0 模块 PCLK 时钟 1 = 深度睡眠时自动关闭 I2C0 模块 PCLK 时钟
[3]	UART0	0 = 深度睡眠时不关闭 UART0 模块 PCLK 时钟 1 = 深度睡眠时自动关闭 UART0 模块 PCLK 时钟
[2]	GPIO	0 = 深度睡眠时不关闭 GPIO 模块 PCLK 时钟 1 = 深度睡眠时自动关闭 GPIO 模块 PCLK 时钟
[1]	RAM	0 = 深度睡眠时不关闭 RAM 模块 PCLK 时钟 1 = 深度睡眠时自动关闭 RAM 模块 PCLK 时钟
[0]	FLASH	0 = 深度睡眠时不关闭 FLASH 模块 PCLK 时钟 1 = 深度睡眠时自动关闭 FLASH 模块 PCLK 时钟

6.2.8 系统定时器（Systick）

系统内核包含一个集成的系统定时器，SysTick。 SysTick 提供一种简单的，24位写清零，向下计数，计数至0后自动装载的计数器，有一个灵活的控制机制。计数器可作为实时操作系统的节拍定时器或者作为一个简单的计数器。

使能后，系统定时器从SysTick 当前值寄存器(SYST_CVR)的值向下计数到0，并在下一个时钟边沿，重新加载SysTick重装载值寄存器(SYST_RVR)的值到SysTick当前值寄存器(SYST_CVR) ，然后随接下来的时钟递减。当计数器减到0时，标志位COUNTFLAG置位，标志位COUNTFLAG是读清0的。

复位后，SYST_CVR 的值未知。使能前，软件应该写该寄存器使其清0。这样确保定时器在使能后以SYST_RVR中的值计数，而非任意值。

若 SYST_RVR 是 0 ，在重新加载后，定时器将保持当前值 0，这种机制可以用来在不使用系统定时器的使能位的情形下禁用系统定时器。

6.2.9 系统定时器控制寄存器映射

寄存器	偏移量	R/W	描述	复位值
SYST Base Address： SYST_BA = 0xE000_E010				
SYST_CSR	SYST_BA+0x00	R/W	Systick 控制与状态寄存器	0x0000_0000
SYST_RVR	SYST_BA+0x04	R/W	Systick 重装载寄存器	0xXXXX_XXXX
SYST_CVR	SYST_BA+0x08	R/W	Systick 当前值寄存器	0xXXXX_XXXX

6.2.9.1 SysTick 控制与状态 (SYST_CSR) 寄存器

寄存器	偏移量	R/W	描述	复位值
SYST_CSR	SYST_BA+0x00	R/W	Systick 控制与状态寄存器	0x0000_0000

31	30	29	28	27	26	25	24

保留							
23	22	21	20	19	18	17	16
保留							COUNTFLAG
15	14	13	12	11	10	9	8
保留							
7	6	5	4	3	2	1	0
保留					CLKSRC	TICKINT	ENABLE

Bits	描述	
[31:17]	保留	保留
[16]	COUNTFLAG	**系统节拍器计数标志** 从上次该寄存器被读，如果定时器计数到0，则返回1。 计数由1到0时，COUNTFLAG 置位。 读该位或向当前值寄存器（SYST_CVR）写时，COUNTFLAG 被清零。
[15:3]	保留	保留
[2]	CLKSRC	**系统节拍器时钟源选择** 1：内核时钟(HCLK)用于SysTick. 0：保留
[1]	TICKINT	**系统节拍器中断使能** 1 ： 向下计数到0将导致SysTick 中断发生。通过软件寄存器写操作清SysTick ，当前值寄存器的值将不会导致SysTick发生中断。 0 ： 向下计数到0不会引起SysTick发生中断。软件可以使用COUNTFLAG来确定是否已经发生计数到0。
[0]	ENABLE	**系统节拍器计数器使能** 1 ： 计数器开始计数，并运行于周期模式. 0 ： 禁用计数器

6.2.9.2 SysTick 重装载值（SYST_RVR）寄存器

寄存器	偏移量	R/W	描述	复位值
SYST_RVR	SYST_BA+0x04	R/W	Systick 重装载寄存器	0xXXXX_XXXX

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

Bits	描述	
[31:24]	保留	保留
[23:0]	RELOAD	**系统节拍器重装载值** 当计数器达到 0 时，该值加载到当前值寄存器.

6.2.9.3 SysTick 当前值（SYST_CVR）寄存器

寄存器	偏移量	R/W	描述	复位值
SYST_CVR	SYST_BA+0x08	R/W	Systick 当前值寄存器	0xXXXX_XXXX

31	30	29	28	27	26	25	24
保留							
23	22	21	20	19	18	17	16
CURRENT[23:16]							

15	14	13	12	11	10	9	8
CURRENT [15:8]							
7	6	5	4	3	2	1	0
CURRENT [7:0]							

Bits	描述	
[31:24]	保留	保留
[23:0]	CURRENT	**系统节拍器当前值** 当前计数值，为采样时刻的计数器的值，计数器不提供读-修改-写保护功能，该寄存器为写清零的，软件写入任何值将清寄存器为 0。这些位不支持读为零(read as zero)，参见系统重装载值寄存器（SYST_RVR)。

6.2.10 嵌套向量中断控制器（NVIC）

CPU内核提供中断控制器，作为异常模式的组成部分，称之为“嵌套向量中断控制器(NVIC)”。它与处理器内核紧密联系，并具有以下特性：

- 支持嵌套和向量中断
- 自动保存和恢复上下文
- 可动态改变优先级
- 简化的精确的中断延迟

NVIC 对所有支持的异常按优先级排序并处理，所有异常在“处理模式”处理。NVIC 结构支持具有四级优先级的 32 个(IRQ[31:0])离散中断。所有的中断和大多数系统异常可以配置为不同的优先级。当中断发生时，NVIC 将比较新中断与当前中断的优先级，如果新中断优先级高于当前中断，则新中断将代替当前中断被处理。

当任何中断被响应时，中断服务程序（ISR）的起始地址从内存的向量表中取得。不需要由软件确定响应哪个中断，也不要软件跳转到相应 ISP 的起始地址。

当取得起始地址时，NVIC 将自动保存处理器状态，包括以下寄存器“PC, PSR, LR, R0~R3, R12” 的值到栈中。在 ISR 结束时, NVIC 将从栈中恢复相关寄存器的值，恢复正常操作， 因此处理器将花费更少的并且确定的时间去处理中断请求。

NVIC 支持末尾连锁“Tail Chaining” ，有效处理尾对尾中断“back-to-back interrupts”， 即无需重复保存和恢复当前状态从而减少从当前 ISR 结束切换到等待处理的 ISR 的延迟时间。NVIC 还支持晚到“Late Arrival”，可以提升同时发生的 ISR 的效率。在当前 ISR 开始执行（保存处理器状态并获取起始地址阶段）之前如果较高优先级中断请求发生，NVIC 将立即选择处理更高优先级的中断，从而提高了实时性。

6.2.10.1 异常模式和系统中断映射

下表列出了 MT031x 系列支持的异常模型。软件可以对其中一些异常以及所有中断设置 4 级优先级。最高用户可配置优先级记为“0”，最低优先级记为“3”，所有用户可配置的优先级的默认值为“0”。注意：优先级“0”在整个系统中为第 4 优先级，排在“Reset”，“NMI” 与“Hard Fault”之后。

表：异常模型

异常名称	向量号	优先级
Reset	1	-3
NMI	2	-2
Hard Fault	3	-1
保留	4~10	保留
SVCall	11	可配置
保留	12~13	保留
PendSV	14	可配置
Systick	15	可配置
Interrupt(IRQ0-IRQ31)	16~47	可配置

表：系统中断映射向量表

异常号	向量地址	中断号（Bit in Interrupt Register)	中断名	源 IP	中断描述
1	0x00	-	-	-	系统异常
NMI	0x04	-	WDT_irq	WDT 模块	WDT 模块中断
3-15	0x08-0x3C	-	-	-	系统异常
16	0x40	0	保留	保留	保留
17	0x44	1	UART0_irq	UART0 模块	UART 模块所产生的中断，具体产生条件见 UART 模块一章
18	0x48	2	I2C_irq	I2C 模块	I2C 模块所产生的中断，具体产生条件见 I2C 模块一章
19	0x4C	3	SPI0_irq	SPI0 模块	SPI0 模块所产生的中断，具体产生条件见 SPI 模块一章
20	0x50	4	TIMR0_irq	TIMER0 模块	TIMER0 模块所产生的中断，具体产生条件见 TIMER 模块一章
21	0x54	5	MPWM_irq	MPWM 模块	MPWM 模块所产生的中断，具体产生条件见 MPWM 模块一章
22	0x58	6	GPIO0_irq	GPIO 模块	GPIO 模块所产生

					的中断，具体产生条件见 GPIO 模块一章
23	0x5C	7	ADC_irq	ADC 模块	ADC 模块所产生的中断，具体产生条件见 ADC 模块一章
24	0x60	8	BOD_irq	系统管理器	低电压检测中断，具体产生条件见系统管理器一章
25	0x64	9	BOR_irq	系统管理器	低电压复位中断，具体产生条件见系统管理器一章
26	0x68	10	TSC_irq	TSC 模块	TSC 模块所产生的中断，具体产生条件见 TSC 模块一章
27	0x6C	11	CRS_irq	CRS 模块	CRS 模块所产生的中断，具体产生条件见 CRS 模块一章
28	0x70	12	CMP_irq	CMP 模块	CMP 模块所产生的中断，具体产生条件见 CMP 模块一章
29	0x74	13	USB	USB 模块	USB 模块所产生的中断，具体产生条件见 USB 模块一章
30	0x78	14	STIMER_irq	STIMER 模块	STIMER 模块所产生的中断，具体产生条件见 STIMER 模块一章
31	0x7C	15	CAN_irq	CAN 模块	CAN 模块所产生的

					中断，具体产生条件见 CAN 模块一章
32~47	0x80~0xBC	15-31	保留	保留	保留

6.2.10.2 向量表

当任何中断被响应时，处理器会自动从内存的向量表中获取中断服务程序（ISR）的起始地址，向量表的基地址固定在 0x00000000。向量表包括复位后栈指针的初始值，和所有异常处理函数的入口地址。上一节的向量号(异常号)决定了异常处理函数在向量表中的入口顺序。

向量表字偏移量	描述
0	SP_MAIN-主堆栈指针
Vector Number	异常入口指针，用向量号表示

6.2.10.3 操作描述

通过写相应中断使能设置寄存器或清使能寄存器位域，可以使能 NVIC 中断或禁用 NVIC 中断，这些寄存器通过写 1 使能和写 1 清除为零，读取这两个寄存器均返回当前相应中断的使能状态。当某一个中断被禁用时，中断声明将使该中断处于等待处理状态，然而，该中断不会被激活。如果某一个中断在被禁用时处于激活状态，该中断就保持在激活状态，直到通过复位或异常返回来清除。清使能位可以阻止相关中断被再次激活。

NVIC 中断可以使用互补的寄存器对来挂起/解除挂起以使能/禁用这些中断，这些寄存器分别为 Set-Pending 寄存器与 Clear-Pending 寄存器，这些寄存器使用写 1 使能和写 1 清除的方式，读取这两种寄存器都返回相应中断的当前挂起状态。Clear-Pending 寄存器不会对处于激活状态的中断的执行状态产生任何影响。

NVIC中断通过更新32位寄存器中的各个8位字段（每个寄存器支持4个中断）来分配中断的优先级。与NVIC相关的通用寄存器都可以通过系统控制空间的内存

区域访问，下一节将作出描述。

6.2.10.4 NVIC 控制寄存器映射

R：只读，W：只写，R/W：可读写

寄存器	偏移量	R/W	描述	复位后的值
NVIC 的基地址： **NVIC_BA = 0xE000_E100**				
NVIC_ISER	NVIC_BA+0x000	R/W	IRQ0~IRQ31 设置使能控制寄存器	0x0000_0000
NVIC_ICER	NVIC_BA+0x080	R/W	IRQ0~IRQ31 清使能控制寄存器	0x0000_0000
NVIC_ISPR	NVIC_BA+0x100	R/W	IRQ0~IRQ31 设置挂起控制寄存器	0x0000_0000
NVIC_ICPR	NVIC_BA+0x180	R/W	IRQ0~IRQ31 清挂起控制寄存器	0x0000_0000
NVIC_IPR0	NVIC_BA+0x300	R/W	IRQ0~IRQ3 优先级控制寄存器	0x0000_0000
NVIC_IPR1	NVIC_BA+0x304	R/W	IRQ4~IRQ7 优先级控制寄存器	0x0000_0000
NVIC_IPR2	NVIC_BA+0x308	R/W	IRQ8~IRQ11 优先级控制寄存器	0x0000_0000

NVIC_IPR3	NVIC_BA+0x30C	R/W	IRQ12~IRQ15 优先级控制寄存器	0x0000_0000
NVIC_IPR4	NVIC_BA+0x310	R/W	IRQ16~IRQ19 优先级控制寄存器	0x0000_0000
NVIC_IPR5	NVIC_BA+0x314	R/W	IRQ20~IRQ23 优先级控制寄存器	0x0000_0000
NVIC_IPR6	NVIC_BA+0x318	R/W	IRQ24~IRQ27 优先级控制寄存器	0x0000_0000
NVIC_IPR7	NVIC_BA+0x31C	R/W	IRQ27~IRQ31 优先级控制寄存器	0x0000_0000

6.2.10.4.1 IRQ0~IRQ31 设置使能控制寄存器（NVIC_ISER）

寄存器	偏移量	R/W	描述	复位值
NVIC_ISER	**NVIC_BA+0x000**	R/W	**IRQ0~IRQ31 设置使能控制寄存器**	**0x0000_0000**

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

Bits	描述	
[31:0]	SETENA	**中断使能寄存器** 使能1个或多个中断，每位代表从IRQ0~IRQ31的中断号(向量号：16~47) 写： 1 = 使能相关中断 0 = 无效 读： 1 = 相应中断是使能的 0 = 相应中断是禁止的 寄存器读取返回当前使能状态

6.2.10.4.2 IRQ0~IRQ31 清使能控制寄存器（NVIC_ICER）

寄存器	偏移量	R/W	描述	复位值
NVIC_ICER	**NVIC_BA+0x080**	R/W	**IRQ0~IRQ31 清使能控制寄存器**	**0x0000_0000**

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

Bits	描述	
[31:0]	CLRENA	**中断禁止寄存器** 禁用1个或多个中断，每位代表从IRQ0~IRQ31的中断号(向量号：16~47) 写： 1 = 禁用相关中断 0 = 无效 读：

		1 = 相应中断是使能的 0 = 相应中断是禁止的 寄存器读取返回当前使能状态

6.2.10.4.3 IRQ0~IRQ31 设置挂起控制寄存器（NVIC_ISPR）

寄存器	偏移量	R/W	描述	复位值
NVIC_ISPR	**NVIC_BA+0x100**	R/W	**IRQ0~IRQ31 设置挂起控制寄存器**	**0x0000_0000**

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

Bits	描述	
[31:0]	SETPEND	**设置中断挂起寄存器** 软件写1，挂起相应中断。每位代表从IRQ0~IRQ31 的中断号(向量号：16~47) 写0无效 寄存器读取返回当前挂起状态

6.2.10.4.4 IRQ0~IRQ31 清挂起控制寄存器（NVIC_ICPR）

寄存器	偏移量	R/W	描述	复位值
NVIC_ICPR	**NVIC_BA+0x180**	R/W	**IRQ0~IRQ31 清挂起控制寄存器**	**0x0000_0000**

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

Bits	描述	
[31:0]	CLRPEND	**清中断挂起寄存器** 写1清除挂起的相应中断，每位代表从IRQ0~IRQ31的中断号(向量号：16~47) 写0无效 寄存器读取返回当前挂起状态

6.2.10.4.5 IRQ0~IRQ3 中断优先级寄存器（NVIC_IPR0）

寄存器	偏移量	R/W	描述	复位值
NVIC_IPR0	**NVIC_BA+0x300**	R/W	**IRQ0~IRQ3 中断优先级寄存器**	**0x0000_0000**

31	30	29	28	27	26	25	24
PRI_3		保留					
23	22	21	20	19	18	17	16
PRI_2		保留					
15	14	13	12	11	10	9	8
PRI_1		保留					
7	6	5	4	3	2	1	0
PRI_0		保留					

Bits	描述	
[31:30]	PRI_3	IRQ3优先级 “0” 表示最高优先级 &“3” 表示最低优先级
[29:24]	保留	保留
[23:22]	PRI_2	IRQ2优先级 “0” 表示最高优先级 &“3” 表示最低优先级
[29:24]	保留	保留
[15:14]	PRI_1	IRQ1优先级 “0” 表示最高优先级 &“3” 表示最低优先级
[13:8]	保留	保留
[7:6]	PRI_0	IRQ0优先级 “0” 表示最高优先级 &“3” 表示最低优先级
[5:0]	保留	保留

6.2.10.4.6 IRQ4~IRQ7 中断优先级寄存器（NVIC_IPR1）

寄存器	偏移量	R/W	描述	复位值
NVIC_IPR1	NVIC_BA+0x304	R/W	IRQ4~IRQ7 中断优先级寄存器	0x0000_0000

31	30	29	28	27	26	25	24
PRI_7		保留					
23	22	21	20	19	18	17	16
PRI_6		保留					
15	14	13	12	11	10	9	8
PRI_5		保留					
7	6	5	4	3	2	1	0
PRI_4		保留					

Bits	描述	
[31:30]	PRI_7	IRQ7优先级 “0” 表示最高优先级 &“3” 表示最低优先级

[29:24]	保留	保留
[23:22]	PRI_6	IRQ6优先级 “0” 表示最高优先级 &“3” 表示最低优先级
[29:24]	保留	保留
[15:14]	PRI_5	IRQ5优先级 “0” 表示最高优先级 &“3” 表示最低优先级
[13:8]	保留	保留
[7:6]	PRI_4	IRQ4优先级 “0” 表示最高优先级 &“3” 表示最低优先级
[5:0]	保留	保留

6.2.10.4.7 IRQ8~IRQ11 中断优先级寄存器（NVIC_IPR2）

寄存器	偏移量	R/W	描述	复位值
NVIC_IPR2	NVIC_BA+0x308	R/W	IRQ8~IRQ11 中断优先级寄存器	0x0000_0000

31	30	29	28	27	26	25	24
PRI_11		保留					
23	22	21	20	19	18	17	16
PRI_10		保留					
15	14	13	12	11	10	9	8
PRI_9		保留					
7	6	5	4	3	2	1	0
PRI_8		保留					

Bits	描述	
[31:30]	PRI_11	IRQ11优先级 “0” 表示最高优先级 &“3” 表示最低优先级
[29:24]	保留	保留
[23:22]	PRI_10	IRQ10优先级

		“0” 表示最高优先级 &“3” 表示最低优先级
[29:24]	保留	保留
[15:14]	PRI_9	IRQ9优先级 “0” 表示最高优先级 &“3” 表示最低优先级
[13:8]	保留	保留
[7:6]	PRI_8	IRQ8优先级 “0” 表示最高优先级 &“3” 表示最低优先级
[5:0]	保留	保留

6.2.10.4.8 IRQ12~IRQ15 中断优先级寄存器（NVIC_IPR3）

寄存器	偏移量	R/W	描述	复位值
NVIC_IPR3	NVIC_BA+0x30C	R/W	IRQ12~IRQ15 中断优先级寄存器	0x0000_0000

31	30	29	28	27	26	25	24
PRI_15		保留					
23	22	21	20	19	18	17	16
PRI_14		保留					
15	14	13	12	11	10	9	8
PRI_13		保留					
7	6	5	4	3	2	1	0
PRI_12		保留					

Bits	描述	
[31:30]	PRI_15	IRQ15优先级 “0” 表示最高优先级 &“3” 表示最低优先级
[29:24]	保留	保留
[23:22]	PRI_14	IRQ14优先级 “0” 表示最高优先级 &“3” 表示最低优先级
[29:24]	保留	保留

[15:14]	PRI_13	IRQ13优先级 “0” 表示最高优先级 &“3” 表示最低优先级
[13:8]	保留	保留
[7:6]	PRI_12	IRQ12优先级 “0” 表示最高优先级 &“3” 表示最低优先级
[5:0]	保留	保留

6.2.10.4.9 IRQ16~IRQ19 中断优先级寄存器（NVIC_IPR4）

寄存器	偏移量	R/W	描述	复位值
NVIC_IPR4	NVIC_BA+0x310	R/W	IRQ16~IRQ19 中断优先级寄存器	0x0000_0000

31	30	29	28	27	26	25	24
PRI_19		保留					
23	22	21	20	19	18	17	16
PRI_18		保留					
15	14	13	12	11	10	9	8
PRI_17		保留					
7	6	5	4	3	2	1	0
PRI_16		保留					

Bits	描述	
[31:30]	PRI_19	IRQ19优先级 “0” 表示最高优先级 &“3” 表示最低优先级
[29:24]	保留	保留
[23:22]	PRI_18	IRQ18优先级 “0” 表示最高优先级 &“3” 表示最低优先级
[29:24]	保留	保留
[15:14]	PRI_17	IRQ17优先级 “0” 表示最高优先级 &“3” 表示最低优先级

[13:8]	保留	保留
[7:6]	PRI_16	IRQ16优先级 “0” 表示最高优先级 &“3” 表示最低优先级
[5:0]	保留	保留

6.2.10.4.10 IRQ20~IRQ23 中断优先级寄存器（NVIC_IPR5）

寄存器	偏移量	R/W	描述	复位值
NVIC_IPR5	NVIC_BA+0x314	R/W	IRQ20~IRQ23 中断优先级寄存器	0x0000_0000

31	30	29	28	27	26	25	24
PRI_23		保留					
23	22	21	20	19	18	17	16
PRI_22		保留					
15	14	13	12	11	10	9	8
PRI_21		保留					
7	6	5	4	3	2	1	0
PRI_20		保留					

Bits	描述	
[31:30]	PRI_23	IRQ23优先级 “0” 表示最高优先级 &“3” 表示最低优先级
[29:24]	保留	保留
[23:22]	PRI_22	IRQ22优先级 “0” 表示最高优先级 &“3” 表示最低优先级
[29:24]	保留	保留
[15:14]	PRI_21	IRQ21优先级 “0” 表示最高优先级 &“3” 表示最低优先级
[13:8]	保留	保留
[7:6]	PRI_20	IRQ20优先级

		“0” 表示最高优先级 &“3” 表示最低优先级
[5:0]	保留	保留

6.2.10.4.11 IRQ24~IRQ27 中断优先级寄存器（NVIC_IPR6）

寄存器	偏移量	R/W	描述	复位值
NVIC_IPR6	NVIC_BA+0x318	R/W	IRQ24~IRQ27 中断优先级寄存器	0x0000_0000

31	30	29	28	27	26	25	24
PRI_27		保留					
23	22	21	20	19	18	17	16
PRI_26		保留					
15	14	13	12	11	10	9	8
PRI_25		保留					
7	6	5	4	3	2	1	0
PRI_24		保留					

Bits	描述	
[31:30]	PRI_27	IRQ27优先级 “0” 表示最高优先级 &“3” 表示最低优先级
[29:24]	保留	保留
[23:22]	PRI_26	IRQ26优先级 “0” 表示最高优先级 &“3” 表示最低优先级
[29:24]	保留	保留
[15:14]	PRI_25	IRQ25优先级 “0” 表示最高优先级 &“3” 表示最低优先级
[13:8]	保留	保留
[7:6]	PRI_24	IRQ24优先级 “0” 表示最高优先级 &“3” 表示最低优先级
[5:0]	保留	保留

6.2.10.4.12 IRQ28~IRQ31 中断优先级寄存器（NVIC_IPR7）

寄存器	偏移量	R/W	描述	复位值
NVIC_IPR7	NVIC_BA+0x31C	R/W	IRQ28~IRQ31 中断优先级寄存器	0x0000_0000

31	30	29	28	27	26	25	24
PRI_31		保留					
23	22	21	20	19	18	17	16
PRI_30		保留					
15	14	13	12	11	10	9	8
PRI_29		保留					
7	6	5	4	3	2	1	0
PRI_28		保留					

Bits	描述	
[31:30]	PRI_31	IRQ31优先级 “0” 表示最高优先级 &“3” 表示最低优先级
[29:24]	保留	保留
[23:22]	PRI_30	IRQ30优先级 “0” 表示最高优先级 &“3” 表示最低优先级
[29:24]	保留	保留
[15:14]	PRI_29	IRQ29优先级 “0” 表示最高优先级 &“3” 表示最低优先级
[13:8]	保留	保留
[7:6]	PRI_28	IRQ28优先级 “0” 表示最高优先级 &“3” 表示最低优先级
[5:0]	保留	保留

6.2.11 系统控制块（SCB）

CPU 内核的状态和操作模式由系统控制寄存器控制，包括 CPUID，CPU 中断优先级和 CPU 电源管理都可以通过这些系统控制寄存器控制。

6.2.11.1 系统控制块寄存器映射

R：只读，W：只写，R/W：可读写

寄存器	偏移量	R/W	描述	复位后的值
SCB 的基地址： **SCB_BA = 0xE000_ED00**				
CPUID	SCB_BA+0x000	R	CPUID寄存器	0x410C_C200
ICSR	SCB_BA+0x004	R/W	中断控制状态寄存器	0x0000_0000
AIRCR	SCB_BA+0x00C	R/W	中断应用和复位控制寄存器	0xFA05_0000
SCR	SCB_BA+0x010	R/W	系统控制寄存器	0x0000_0000
SHPR2	SCB_BA+0x01C	R/W	系统处理函数优先级寄存器2	0x0000_0000
SHPR3	SCB_BA+0x020	R/W	系统处理函数优先级寄存器3	0x0000_0000

6.2.11.1.1 CPUID Base 寄存器（CPUID）

寄存器	偏移量	R/W	描述	复位值
CPUID	**SCB_BA+0x000**	**R**	**CPUID寄存器**	**0x410C_C200**

31	30	29	28	27	26	25	24
IMPLEMENTER[7:0]							
23	22	21	20	19	18	17	16
保留				PART[3:0]			
15	14	13	12	11	10	9	8
PARTNO[11:4]							

7	6	5	4	3	2	1	0
PARTNO[3:0]				REVISION[3:0]			

Bits	描述	
[31:24]	IMPLEMENTER	执行码
[23:20]	保留	保留
[19:16]	PART	处理器架构码
[15:4]	PARTNO	处理器型号
[3:0]	REVISION	版本号

6.2.11.1.2 中断控制状态寄存器（ICSR）

寄存器	偏移量	R/W	描述	复位值
ICSR	**SCB_BA+0x004**	**R/W**	中断控制状态寄存器	**0x0000_0000**

31	30	29	28	27	26	25	24
NMIPENDSET	保留		PENDSVSET	PENDSVCLR	PENDSTSET	PENDSTCLR	保留
23	22	21	20	19	18	17	16
ISRPREEMPT	ISRPENDING	保留				VECTPENDING[5:4]	
15	14	13	12	11	10	9	8
VECTPENDING[3:0]				保留			
7	6	5	4	3	2	1	0
保留		VECTACTIVE[5:0]					

Bits	描述	
[31]	NMIPENDSET	**触发NMI中断** 写: 0 = 无作用 1 = 改变NMI异常状态为未处理. 读:

		0 = 没有NMI 异常等待处理 1 = 有NMI异常等待处理 注：设置该位将激活 NMI，由于 NMI 是最高优先级的异常，正常情况下，只要处理器检测到写’1’到这个位，将马上进入中断处理函数，然后这个位就会被自动清成’0’。这就意味着如果在中断处理函数中读这个位返回’1，就是 NMI 中断再次被置。
[30:29]	保留	保留
[28]	PENDSVSET	**触发PendSV中断** 写: 0 = 无作用 1 = 改变PendSV异常状态为未处理. 读: 0 = 没有PendSV异常等待处理 1 = 有PendSV异常等待处理 注：触发 PendSV 中断。写”1”到这个位是唯一触发 PendSV 的方式。
[27]	PENDSVCLR	**清PendSV中断** 写： 0 = 无作用 1 = 清除PendSV 中断. 这个位是只写的，当要清除 PendSV 中断的时候，必须同时对 PENDSVSET 写”0”， PENDSVCLR 写”1”
[26]	PENDSTSET	**触发SysTick中断** 写: 1 = 触发SysTick中断. 0 = 无作用 读: 1 = 有SysTick中断等待处理 0 = 没有 SysTick 中断等待处理。
[25]	PENDSTCLR	**清除SysTick中断** 写: 1 = 清除等待处理的SysTick.中断 0 = 无作用
[24]	保留	保留
[23]	ISRPREEMPT	**中断抢占位** 如果置位，当从调试状态退出时，未处理的异常将被处理
[22]	ISRPENDING	**有中断等待处理标志位，不包括NMI和HardFault中断(只读)** 0 = 没有中断等待处理

		1 = 有中断等待处理 这个位是只读的
[21:18]	保留	保留
[17:12]	VECTPENDING	**等待处理的最高优先级异常号** 0 = 没有中断等待处理 非 0 = 等待处理的最高优先级异常号
[11:6]	保留	保留
[5:0]	VECTACTIVE	**包含当前正在处理的异常号** 0 = 线程模式 非 0: 当前正在处理的异常号

6.2.11.1.3 应用中断和复位控制寄存器（AIRCR）

寄存器	偏移量	R/W	描述	复位值
AIRCR	**SCB_BA+0x00C**	**R/W**	**中断应用和复位控制寄存器**	**0xFA05_0000**

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

Bits	描述	
[31:16]	VECTORKEY	**寄存器访问值** 写该寄存器时，该域应该是 0x05FA，否则写动作将被忽略。 VECTORKEY 域用来防止复位或者异常状态清除时意外写该寄存器
[15:3]	保留	保留

[2]	SYSRESETREQ	**系统复位请求** 该位写1，产生复位信号给芯片表示有复位请求. 该位是只写的，在复位时自动清零.
[1]	VECTCLRACTIVE	**异常活动状态清除位** 预留用于调试。写该寄存器时，用户必须将该位写0，否则行为无法预期
[0]	保留	保留

6.2.11.1.4 系统控制寄存器（SCR）

寄存器	偏移量	R/W	描述	复位值
SCR	**SCB_BA+0x010**	**R/W**	**系统控制寄存器**	**0x0000_0000**

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

Bits	描述	
[31:5]	保留	保留
[4]	SEVONPEND	**未处理位发送事件** 0 = 只有使能的中断或者事件可以唤醒处理器，禁止的中断不能唤醒 1 = 使能的事件和所有中断，包括禁止的中断，可以唤醒处理器当事件或者中断进入未处理状态时，将把CPU从WFE指令唤醒。如果处理器没有正在等待一个事件，这个事件将被注册，在下一个WFE指令时起作用. 执行SEV指令或者外部事件也可以唤醒处理器

[3]	保留	保留
[2]	SLEEPDEEP	**控制处理器进入睡眠还是深度睡眠模式** 0 = 睡眠 1 = 深度睡眠
[1]	SLEEPONEXIT	**退出时睡眠使能控制** 当从Handler模式返回Thread模式时，是否进入睡眠模式 0： 返回Thread模式时，不要进入睡眠 1：当从中断处理函数返回Thread模式时进入睡眠或者深度睡眠模式 设置这个比特为”1”，可以使能一个中断驱动的应用，避免从中断返回时进入一个空的主函数
[0]	保留	保留

6.3 模拟比较器及运算放大器模块

6.3.1 概述

MT32F005/006 系列微控制器的包含两个比较器：CMP0 和 CMP1，每个比较器有两个模拟输入端: pos 和 neg，用于比较外部输入信号的电平大小。

MT32F006 系列微控制器包含两个运算放大器。每个运算放大器有两个模拟输入端: VIP 和 VIN，一个输出端口 VOUT。放大器的增益可由内部寄存器配置，也可由外部电阻网络决定。

在使用比较器或放大器前，需要将相应的 IO 配置成模拟模式（详细说明见 GPIO 章节），且需在系统管理器模块中打开 CMP 模块的时钟使能位 AHBCLKCTRL.CMP（bit15）。

如果需要复位 CMP 模块，只需向寄存器 bit 位 PRESETCTRL. CMP (bit15) 写 0，再写 1 即可实现模块复位。

6.3.2 比较器特性

- 可以比较两个输入管脚(pos 和 neg)的电压值；
- 可以选择内部电压作为参考电压输入到比较器负端；
- 内部参考电压可灵活调节；
- 比较结果发生变化可以产生中断信号给 CPU；
- 迟滞功能，可以软件配置 15mV/40mV 迟滞选择。

6.3.2.1 外部输入选择

每个模拟比较器都有多个外部输入管脚供选择，这样 PinMux 的复用可以更加灵活，输入管脚可以通过 cmp0/1_neg_sel 和 cmp0/1_pos_sel 来选择。

6.3.2.2 参考电压选择

CMP 可以用于比较 pos 和 neg 管脚上的电压值，也可以选择内部电压作为参考电压。

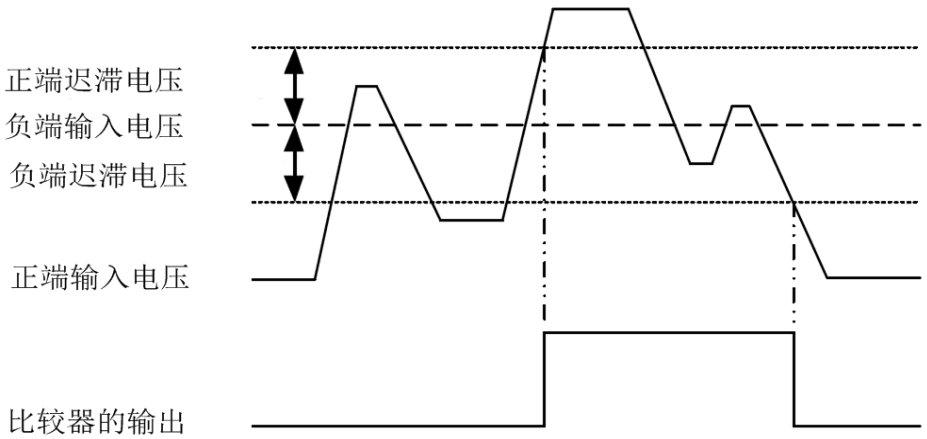
cmp0/1_neg_sel[3:0]是比较器 0/1 的负端输入选择信号，0000 选择 cmp_in0(PIO0_1)、0001 选择 cmp_in1(PIO0_2)、0010 选择 cmp_in2(PIO0_3)、0011 选择 cmp_in3(PIO0_4)、0100 选择 cmp_in4(PIO0_5)、0101 选择 cmp_in5(PIO0_6)、0110 选择 cmp_in6(PIO0_7)、0111 选择 cmp_in7(PIO1_0)、1000 选择电阻分压输出电压、1001 选择内置温度传感器输出电压、1010 选择内部基准电压 1.25V、1011 选择 ADC 的基准 REF（这个 REF 是跟随 ADC 的基准选择变的），1100 选择 LDO 的输出电压，即内核电源电压 1.5V。

cmp0/1_pos_sel[2:0]是比较器 0/1 的正端输入选择信号，正端输入源选择为，000 选择 cmp_in0(adc_in0)、001 选择 cmp_in1(adc_in1)、010 选择 cmp_in2(adc_in2)、011 选择 cmp_in3(adc_in3)、100 选择 cmp_in4(adc_in4)、101 选择 cmp_in5(adc_in5)、110 选择 cmp_in6(adc_in6)、111 选择 cmp_in7(adc_in7)。也可以选择运放的输出。

6.3.2.3 迟滞模式

当 cmpx_hysen 为 1 时，比较器加上 0mw/15mv/40mv 的迟滞功能，能有效的滤除门限电平附近微小的干扰，使输出保持稳定。

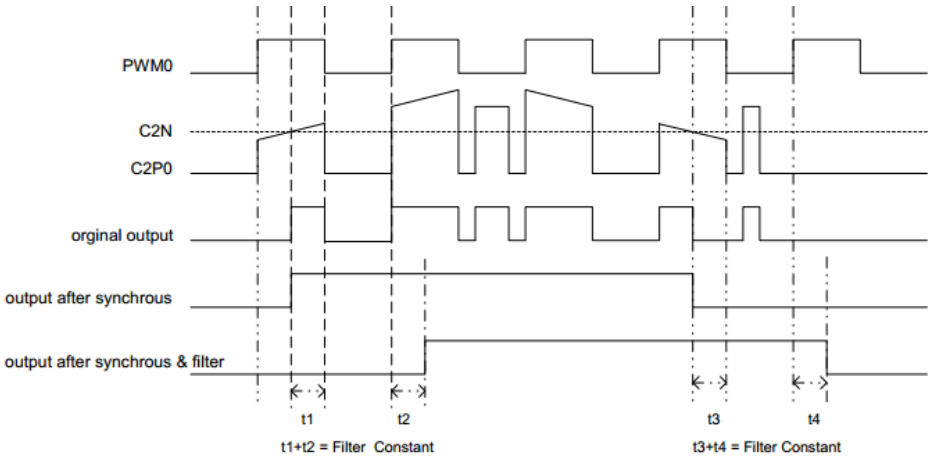
如果当前比较器输出为 0，正端输入电压超过负端输入电压的值大于迟滞电压，比较器输出才会变成 1。相似的，如果当前比较器输出为 1，负端输入电压低于正端输入电压的值大于迟滞电压，比较器输出才会变成 0。



图：CMP 迟滞电压作用示意图

6.3.2.4 比较器_PWM 用法

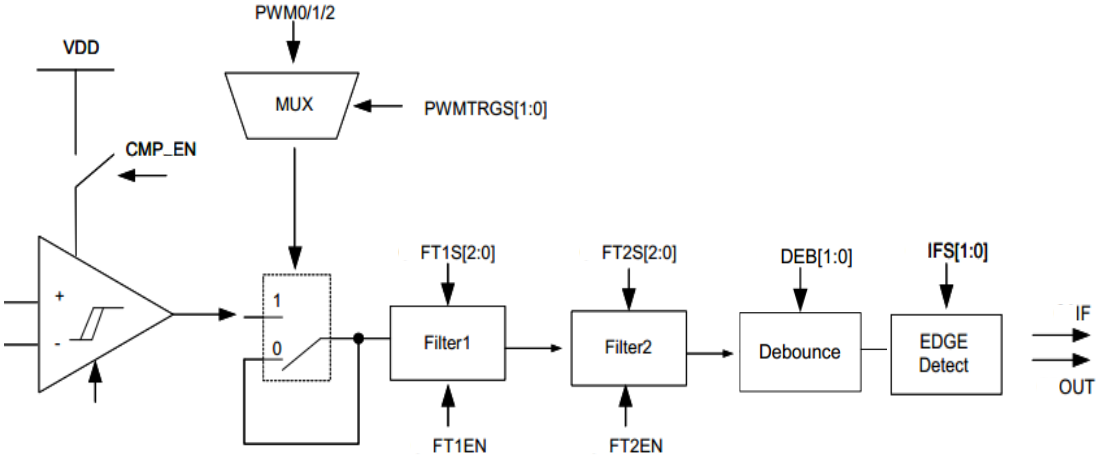
芯片内置 2 个比较器，比较器的输出可以作为 PWM 模块的故障输入检测信号，自动迅速关闭 PWM 输出，从而实现快速保护，也可以用在某些电压/电流闭环中。比较器内置有可编程的施密特窗口，同时集成滤波常数可编程的数字滤波电路，同时提供中断功能，比较适合用作需要精确比较的应用场合，比如直流无刷电机的反电动势过零点检测。比较器集成同步比较功能，可以和 PWM0/PWM1/PWM2 输出同步，如果 TRGPOL 设为 0，那么只有当 PWM 输出为有效电平期间，比较器的输出结果才被采样，给到后级滤波电路，在 PWM 输出无效期间，不采样比较器输出， 此时比较器的输出会保持， 同时滤波器也停止计数， 当 PWM 输出再次有效时， 恢复对比较器的采样和滤波计数。参见下图（PWMTRGS[1:0]= 01，TRGPOL = 0）。比较器的滤波是先经过滤波器 1 再经过滤波器 2，如果只想用其中的一种滤波，可以将另一种屏蔽，也可以将两种都屏蔽。



图：比较器 2 同步比较示意图，（C2PCHS[1:0] = 0, PWMTRGS[1:0] = 01,TRGPOL = 0）

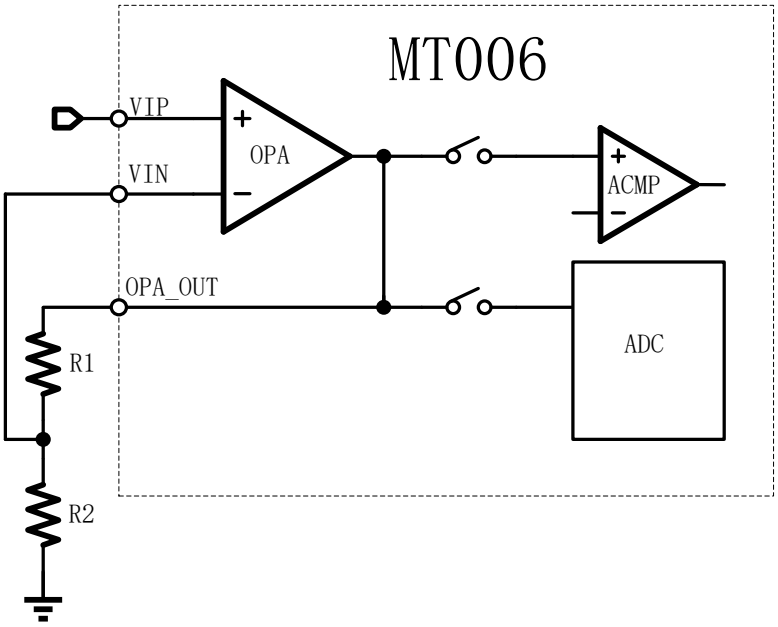
6.3.2.5 滤波功能的用法

两个比较器的滤波功能都采用三级级联的方法，如下图所示，用法见寄存器描述。

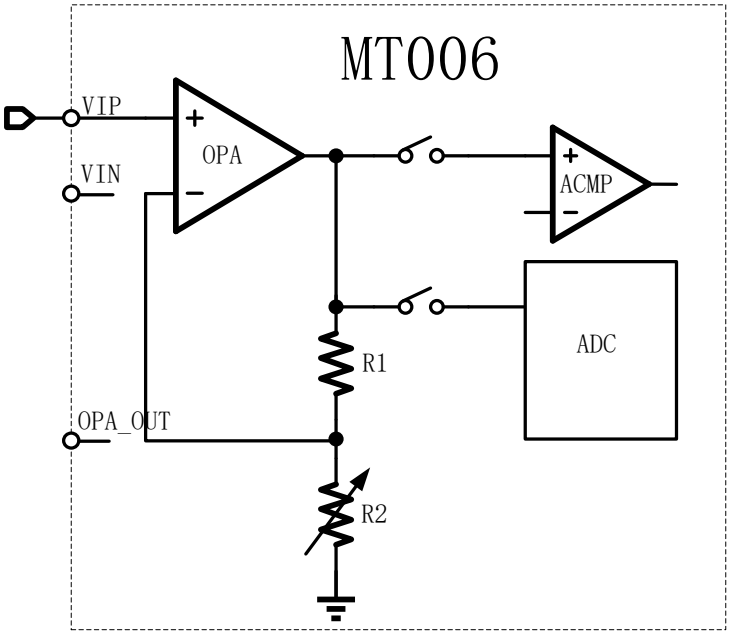


6.3.3 运算放大器特性

- 每个比较器模块包含 VIN，VIP 和 VOUT 三个端口
- 增益可由内部寄存器配置，也可由片外电阻网络决定
- 运放的输出可内部连接到比较器或者 ADC 的输入端
- 工作在内部模式时，VOUT 只能连接到比较器或 ADC，无法输出到 IO 管脚



片外电阻网络模式



片内电阻增益模式

6.3.4 寄存器映射

R：只读，W：只写，R/W：可读写

寄存器	偏移量	R/W	描述	复位后的值

ACMP 的基地址： ACMP_BA = 0x4000_5800				
CTRL0	ACMP _BA+0x00	R/W	ACMP 控制寄存器 0	0x0000_0000
CTRL1	ACMP _BA+0x10	R/W	ACMP 控制寄存器 1	0x0000_0000
CTRL2	ACMP _BA+0x20	R/W	ACMP 控制寄存器 2	0x0000_0000
STATUS	ACMP _BA+0x30	R	ACMP 状态寄存器	0x0000_0000
OP_CTRL	ACMP _BA+0x40	R/W	运算放大器 OP 控制寄存器	0x0101_0101
ADV_CTRL0	ACMP _BA+0x50	R/W	附加控制寄存器 0	0x0000_0000
ADV_CTRL1	ACMP _BA+0x60	R/W	附加控制寄存器 1	0x0000_0000
ADV_CTRL2	ACMP _BA+0x70	R/W	附加控制寄存器 2	0x0000_0000
ADV_CTRL3	ACMP _BA+0x80	R/W	附加控制寄存器 3	0x0000_0000
CMPCLKDIV	ACMP _BA+0xA0	R/W	比较器模拟时钟分频寄存器	0x0000_0010

6.3.4.1 CTRL0 寄存器

寄存器	偏移量	R/W	描述	复位值
CTRL0	ACMP_BA+0x00	R/W	ACMP 控制寄存器 0	0x0000_0000

Bits	描述	
[31:21]	保留	保留
[20]	cmp0_inv	**比较器 0 输出取反** 0：不取反 1：取反
[19:18]	保留	保留
[17:16]	cmp0_edge_sel	**选择比较器 0 输出边沿触发标志位和中断** 00：下降沿

		01：上升沿 10：双边沿
[15:14]	保留	保留
[13]	cmp0_op_ch	比较器0正端选择OPA输出的选择信号，0选择OPA0的输出，1选择OPA1
[12]	cmp0_pos_op	置1时让比较器0选择两个OPA的输出作为输入，默认0.
[11]	保留	保留
[10:8]	cmp0_pos_sel	比较器0的正端输入源选择: 000选择cmp_in0 (adc_in0) 001选择cmp_in1 (adc_in1) 010选择cmp_in2 (adc_in2) 011选择cmp_in3 (adc_in3) 100选择cmp_in4 (adc_in4) 101选择cmp_in5 (adc_in5) 110选择cmp_in6 (adc_in6) 111选择cmp_in7 (adc_in7)
[7：4]	cmp0_neg_sel	比较器0的负端输入源选择: 0000选择cmp_in0 (adc_in0) 0001选择cmp_in1 (adc_in1) 0010选择cmp_in2 (adc_in2) 0011选择cmp_in3 (adc_in3) 0100选择cmp_in4 (adc_in4) 0101选择cmp_in5 (adc_in5) 0110选择cmp_in6 (adc_in6) 0111选择cmp_in7 (adc_in7) 1000选择电阻分压输出电压 1001选择内置温度传感器输出电压 010选择内部基准电压1.25V 1011选择ADC的基准REF(这个REF是跟随ADC的基准选择变的，

		就是ADC选择的基准是什么，这边的基准就是什么） 1100选择LDO的输出电压，即内核电源电压
[3：2]	cmp0_hys_sel	比较器 0 的迟滞选择信号 10 对应 15mV 迟滞 11 对应 40mv 迟滞 00，01：保留
[1]	cmp0_ie	**比较器0中断使能** 0：比较器 0 的中断被禁能 1：比较器 0 的中断被使能
[0]	cmp0_en	**比较器 0 的使能位** 在使用比较器 0 之前，需将该 bit 位置 1 来使能比较器 0 0：比较器被禁能 1：比较器被使能

6.3.4.2 CTRL1 寄存器

寄存器	偏移量	R/W	描述	复位值
CTRL1	ACMP_BA+0x10	R/W	ACMP 控制寄存器 1	0x0000_0000

Bits	描述	
[31:21]	保留	保留
[20]	Cmp1_inv	**比较器 1 输出取反** 0：不取反 1：取反
[19:18]	保留	保留
[17:16]	Cmp1_edge_sel	**选择比较器 1 输出边沿触发标志位和中断** 00：下降沿 01：上升沿 10：双边沿

[15:14]	保留	保留
[13]	Cmp1_op_ch	比较器1正端选择OPA输出的选择信号，0选择OPA0的输出，1选择OPA1
[12]	Cmp1_pos_op	置1时让比较器1选择两个OPA的输出作为输入，默认0
[11]	保留	保留
[10:8]	Cmp1_pos_sel	比较器1的正端输入源选择： 000选择cmp_in0 (adc_in0) 001选择cmp_in1 (adc_in1) 010选择cmp_in2 (adc_in2) 011选择cmp_in3 (adc_in3) 100选择cmp_in4 (adc_in4) 101选择cmp_in5 (adc_in5) 110选择cmp_in6 (adc_in6) 111选择cmp_in7 (adc_in7)
[7：4]	Cmp1_neg_sel	比较器1的负端输入源选择位： 0000选择cmp_in0 (adc_in0) 0001选择cmp_in1 (adc_in1) 0010选择cmp_in2 (adc_in2) 0011选择cmp_in3 (adc_in3) 0100选择cmp_in4 (adc_in4) 0101选择cmp_in5 (adc_in5) 0110选择cmp_in6 (adc_in6) 0111选择cmp_in7 (adc_in7) 1000选择电阻分压输出电压 1001选择内置温度传感器输出电压 010选择内部基准电压1.25V 1011选择ADC的基准REF（这个REF是跟随ADC的基准选择变的，就是ADC选择的基准是什么，这边的基准就是什么） 1100选择LDO的输出电压，即内核电源电压

[3：2]	Cmp1_hys_sel	比较器 1 的迟滞选择信号 10 对应 15mV 迟滞 11 对应 40mv 迟滞 00，01：保留
[1]	Cmp1_ie	**比较器1中断使能** 0：比较器 1 的中断被禁能 1：比较器 1 的中断被使能
[0]	Cmp1_en	**比较器 1 的使能位** 在使用比较器 1 之前，需将该 bit 位置 1 来使能比较器 0 0：比较器被禁能 1：比较器被使能

6.3.4.3 STATUS 寄存器

寄存器	偏移量	R/W	描述	复位值
STATUS	ACMP_BA+0x30	R	ACMP 状态寄存器	0x0000_0000

31	30	29	28	27	26	25	24
保留							
23	22	21	20	19	18	17	16
保留							
15	14	13	12	11	10	9	8
保留							
7	6	5	4	3	2	1	0
保留			cmp1_co	cmp0_co	保留	cmp1_flag	cmp0_flag

Bits	描述	
[31:5]	保留	保留

[4]	cmp1_co	**比较器1的比较结果** 0：pos端电压低于neg端电压 1：pos端电压高于neg端电压
[3]	cmp0_co	**比较器0的比较结果** 0：pos端电压低于neg端电压 1：pos端电压高于neg端电压
[2]	保留	保留
[1]	cmp1_flag	**比较器 1 的中断标志位** 为 1 时表示比较器 1 的比较结果发生变化。可向该 bit 位写 1 来清除该 bit 位。 0：比较器 1 比较结果没有变化 1：比较器 1 比较结果发生变化
[0]	cmp0_flag	**比较器 0 的中断标志位** 为 1 时表示比较器 0 的比较结果发生变化。可向该 bit 位写 1 来清除该 bit 位。 0：比较器 0 比较结果没有变化 1：比较器 0 比较结果发生变化

6.3.4.4 OP_CTRL 寄存器

寄存器	偏移量	R/W	描述	复位值
OP_CTRL	ACMP_BA+0x40	R/W	运算放大器控制寄存器	0x0101_0101

Bits	描述	
[31:12]	保留	保留
[11:9]	OP1_GAIN	**运放 1 的增益选择** 000时增益由片外电阻网络决定 001对应增益1

		010对应增益2 011对应增益3 100对应增益4 101对应增益6 110对应增益8 111 对应增益 10
[8]	OP1_PWD	**运算放大器 1 power down 使能** 在使用运算放大器 1 之前，需将该 bit 位置 0 来使能运算放大器 1 0：运算放大器 1 被使能 1：运算放大器1被禁能
[7:4]	**保留**	**保留**
[3:1]	OPO_GAIN	**运放 0 的增益选择** 000时增益由片外电阻网络决定 001对应增益1 010对应增益2 011对应增益3 100对应增益4 101对应增益6 110对应增益8 111 对应增益 10
[0]	OPO_PWD	**运算放大器 0 power down 使能** 在使用运算放大器 0 之前，需将该 bit 位置 0 来使能运算放大器 0 0：运算放大器 0 被使能 1：运算放大器0被禁能

6.3.4.5 ADV_CTRL0 寄存器

寄存器	偏移量	R/W	描述	复位值
ADV_CTRL0	ACMP_BA+0x50	R/W	附加控制寄存器 0	0x0000_0000

Bits	描述	
[31:7]	保留	保留
[6]	CMP1_TRGPOL	**同步比较功能控制位** 0：仅在PWM输出有效期间比较 1：仅在PWM输出无效期间比较
[5:4]	CMP1_TRGS	**比较器 1 同步比较控制位** 00：连续比较模式，无同步比较功能 01：和PWM0输出同步 10：和PWM1输出同步 11：和PWM2输出同步
[3]	保留	保留
[2]	CMP0_TRGPOL	**同步比较功能控制位** 0：仅在PWM输出有效期间比较 1：仅在PWM输出无效期间比较
[1:0]	CMP0_TRGS	**比较器 0 同步比较控制位** 00：连续比较模式，无同步比较功能 01：和PWM0输出同步 10：和PWM1输出同步 11：和PWM2输出同步

6.3.4.6 ADV_CTRL1 寄存器

寄存器	偏移量	R/W	描述	复位值
ADV_CTRL1	ACMP_BA+0x60	R/W	附加控制寄存器 1	0x0000_0000

Bits	描述	
[31:16]	保留	保留
[15]	CMP1_FT1EN	**比较器 1 滤波器 1 使能** 0：滤波器使能 1：滤波器禁能 用法和比较器0相同
[14:12]	CMP1_FT1S	**比较器 1 滤波器 1 参数设置** 000：滤波器常数为32 001：滤波器常数为64 010：滤波器常数为128 011：滤波器常数为256 100：滤波器常数为512 101：滤波器常数为1024 110：滤波器常数为2048 111：滤波器常数为4096
[11]	CMP1_FT2EN	**比较器 1 滤波器 2 使能** 0：滤波器使能 1：滤波器禁能 用法和比较器0相同
[10:8]	CMP1_FT2S	**比较器 1 滤波器 2 参数设置** 000：滤波器常数为8 001：滤波器常数为16 010：滤波器常数为32 011：滤波器常数为64 100：滤波器常数为128 101：滤波器常数为256 110：滤波器常数为512 111：滤波器常数为1024

[7]	CMP0_FT1EN	**比较器 0 滤波器 1 使能** 0：滤波器使能 1：滤波器禁能 滤波器 1 算法说明：用系统时钟采样滤波器输入采用数字积分，如果采样结果为高电平，计数器加 1，计数器结果超出 CMP0_FT1S [2:0]设定的常数，则滤波器输出 1 同时计数器置为滤波常数×2；如果采样结果为低电平，计数器减 1，计数器结果小于滤波器常数,则滤波器输出 0 同时计数器置为 0 **注：** 1. 计数器上限设置为常数的 *2* 倍，下限为 *0*，超出上下限，不执行加减动作 2. 滤波计数器初值设定:若比较器中断设为上升沿触发，计数器初始值为 *0*；若设为下降沿触发，则滤波计数器初值设为滤波常数 *x2*。若设为双沿触发或不触发，则滤波计数器初值设为滤波常数 3. 滤波器输出稳定需要一个建立时间，大致为滤波常数设置的时间，如滤波常数设置为 256，则在 256 个系统时钟后，滤波器输出稳定，在这之前，滤波器输出不确定，中断标志 C2IF 可能会被误置 1，因此，刚打开滤波器时，需要在滤波器输出稳定后软件清除此中断标志位
[6:4]	CMP0_FT1S	**比较器 0 滤波器 1 参数设置** 000：滤波器常数为32 001：滤波器常数为64 010：滤波器常数为128 011：滤波器常数为256 100：滤波器常数为512 101：滤波器常数为1024 110：滤波器常数为2048 111：滤波器常数为4096
[3]	CMP0_FT2EN	**比较器 0 滤波器 2 使能** 0：滤波器使能 1：滤波器禁能 滤波器 2 算法说明:用系统时钟采样滤波器输入当滤波器 2 当前输出为 0 时，若输入采样为 1，计数器加 1；若输入采样为 0，则将计数器清 0；当计数器超过 CMP0_FT2S [2:0]设定的滤波常数时，滤波器输出翻转为 1，同时计数器清 0 同理，当滤波器 2 当前输出为 1 时，若输入采样为 0，计数器加 1，若输入采样为 1，则将计数器清 0，当计数器超过 CMP0_FT2S [2:0]设定的滤

		波常数时，滤波器输出翻转为 0，同时计数器清 0 **注：** 1. 计数器初始值为 0 2. 滤波器输出稳定需要一个建立时间，大致为滤波常数设置的时间，如滤波常数设置为 256，则在 256 个系统时钟后，滤波器输出稳定，在这之前，滤波器输出不确定，中断标志 C2IF 可能会被误置 1，因此，刚打开滤波器时，需要在滤波器输出稳定后软件清除此中断标志位
[2:0]	CMP0_FT2S	**比较器 0 滤波器 2 参数设置** 000：滤波器常数为8 001：滤波器常数为16 010：滤波器常数为32 011：滤波器常数为64 100：滤波器常数为128 101：滤波器常数为256 110：滤波器常数为512 111：滤波器常数为1024

6.3.4.7 ADV_CTRL2 寄存器

寄存器	偏移量	R/W	描述	复位值
ADV_CTRL2	ACMP_BA+0x70	R/W	附加控制寄存器 2	0x0000_0000

Bits	描述	
[31:24]	保留	保留
[23:16]	CMP1_DEB_NUM	**比较器 1 输出信号滤波器参数设置** 滤波时间大小为CMP1_DEB_NUM+1个cycle
[15:8]	CMP0_DEB_NUM	**比较器 0 输出信号滤波器参数设置** 滤波时间大小为CMP0_DEB_NUM+1个cycle
[7:2]	保留	保留
[1]	CMP1_DEB_EN	比较器1输出信号滤波器使能

[0]	CMP0_DEB_EN	比较器 0 输出信号滤波器使能

6.3.4.8 ADV_CTRL3 寄存器

寄存器	偏移量	R/W	描述	复位值
ADV_CTRL3	ACMP_BA+0x80	R/W	附加控制寄存器 3	0x0000_0000

Bits	描述	
[31:12]	保留	保留
[11:10]	PWM3_FLT_SEL	PWM3 输出使能选择 00：使能一直有效 01：比较器0输出的为0的时候使能有效 10：比较器1输出的为0的时候使能有效 11：保留
[9:8]	PWM2_FLT_SEL	PWM2 输出使能选择 00：使能一直有效 01：比较器0输出的为0的时候使能有效 10：比较器1输出的为0的时候使能有效 11：保留
[7:6]	PWM1_FLT_SEL	PWM1 输出使能选择 00：使能一直有效 01：比较器0输出的为0的时候使能有效 10：比较器1输出的为0的时候使能有效 11：保留
[5:4]	PWM0_FLT_SEL	PWM0 输出使能选择 00：使能一直有效 01：比较器0输出的为0的时候使能有效 10：比较器1输出的为0的时候使能有效 11：保留

[3:2]	保留	保留
[1]	CMP1_FLT_SEL	比较器 1 故障输入选择 0：比较器输出 1：外部故障管脚输入
[0]	CMP0_FLT_SEL	比较器 0 故障输入选择 0：比较器输出 1：外部故障管脚输入

6.3.4.9 ADV_CTRL4 寄存器

寄存器	偏移量	R/W	描述	复位值
ADV_CTRL4	ACMP_BA+0x90	R/W	附加控制寄存器 3	0x0000_0000

Bits	描述	
[31:14]	保留	保留
[13:8]	res_div	电阻分压选择 生成 (1/res_div)* ref_vdd的电压，ref_vdd见bit0
[7:5]	保留	保留
[4]	res_div_enable	电阻分压模块使能信号，默认 0，为 1 时开启电阻分压模块
[1]	保留	保留
[0]	ref_sel	比较器电阻分压输入的选择信号，默认 0 选择 VDD 作为电阻分压的输入源，置 1 时选择 ADC 基准作为电阻分压的输入

6.3.4.10 CMPCLKDIV 寄存器

寄存器	偏移量	R/W	描述	复位值
CMPCLKDIV	ACMP_BA+0xa0	R/W	运算放大器控制寄存器 3	0x0000_0010

Bits	描述	
[31:8]	保留	保留
[7:0]	DIV	**输出给模拟部分 12M 时钟的分频值** 模拟部分需要一个恒定的 12MHz 时钟，当系统 PLL 频率发生变化时，需要改变该寄存器值以满足 12MHz 频率要求。

6.4 模拟数字转换器（ADC）

6.4.1 概述

MT32F005/006 系列包含一个 16 通道 12 位的模拟–数字转换器(A/D 转换器). A/D 转换器支持以下几种工作模式: 单次转换模式、循环转换模式、非连续转换模式。A/D 转换可以通过软件、外部管脚或者内部硬件信号（如 PWM 等）来触发。在使用 ADC 之前，需要在系统管理器模块中打开 ADC 模块的时钟使能位 AHBCLKCTRL.ADC(bit5) 和清除 ADC 模块的 PowerDown 使能位 PDRUNCFG.ADC(bit0)。

如果需要复位 ADC 模块，只需向寄存器 bit 位 PRESETCTRL.ADC(bit5)写 0，再写 1 即可实现模块复位。

6.4.2 特性

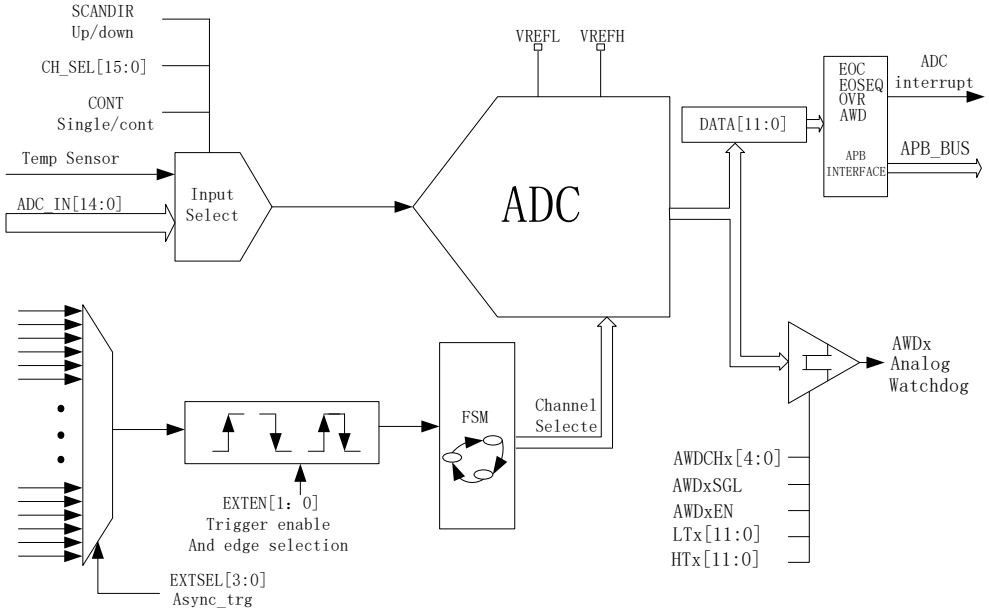
- 模拟输入电压范围: 5mV～0.99*VDD
- 12/10/8/6 位分辨率可调
- 自带失调校准以及增益误差校准
- 最多 14 路外部模拟输入通道
- 1 路 1/4 VDD 测量通道
- 1 路内部温度传感器输入通道
- 基准电压可选择 VDD 或片内基准电压 VREF（2.0V@typical）
- 最大的 ADC 时钟频率 24MHz，高达 1.5M SPS 采样速率
- 三种操作模式
  - ◆ 单次转换模式：A/D 一次完成所有使能通道上的转换
  - ◆ 循环转换模式：A/D 完成所有使能通道上的转换后，自动的从头开始新一轮相同的转换
  - ◆ 非连续转换模式：每完成一次转换，都需要软件或者硬件来触发
- A/D 转换开始条件

- ◆ 软件向 ADSTART 写 1
- ◆ 内部硬件信号触发(如 timer、MotorPWM 等)
- ◆ 外部 IO 管脚触发

- 通道 0-15 有单独的结果寄存器，用于暂存转换的结果
- 模拟看门狗功能，当采样结果大于或小于设定的值时产生中断

6.4.3 ADC 功能描述

下图为 ADC 功能模块框图:



图：ADC 模块功能框图

下表为 ADC 模拟管脚的说明：

名称	信号类型	注解
VREFH	输入，模拟参考正极	ADC 使用的高端/正极参考电压，VREFH = VDD 或 VREF
VREFL	输入，模拟参考负极	ADC 使用的低端/负极参考电压，VREFL = VSS
ADCx_IN[13:0]	模拟信号输入	14 个外部模拟输入通道
ADCx_IN[14]	1/4*VDD 信号输入	1/4 VDD 电压输入
ADC_IN[15]	内部模拟输入信号	内部温度传感器输入

6.4.3.1 ADC 时钟

ADC 模拟部分的时钟可以是时钟控制器提供的 ADC_CLK 时钟或者是HCLK(系统时钟)的分频。寄存器 CFGR2.CLKMODE[1:0]来控制时钟源选择，详细请见寄存器章节。

6.4.3.2 通道选择

ADC 一共有 16 个模拟信号输入(0~15)，通道 15 用于连接内部温度传感器,通道 14 用于连接 1/4 VDD 电压，其他 14 个通道都由芯片 GPIO 管脚输入，ADC0-6 通道对应于 PIO0_1-PIO0_7，ADC7-14 通道对应于 PIO1_0，PIO1_1，PIO1_4-PIO1-7, PIO2_0。其中通道 12 还可以选择 OP0 的输出，通道 13 可以选择 OP1 的输出。

用户可以选择单个通道的转换或者选择多个通道进行连续的扫描方式的转换。每个通道在 CHESELR 寄存器中都有独立的使能位(CHSEL0~CHSEL15)。

扫描转换的方向由 CFGR1. SCANDIR 位控制：

- SCANDIR=0:转换由通道 0 到通道 15
- SCANDIR=1:转换由通道 15 到通道 0

通道 0~7 有独立的结果寄存器，如果使能这几个通道，可以等所有通道转换完成后再一起读取各个通道的转换结果。如果选择其它通道进行转换，则需要每次转换完成后就读取转换结果，不然会造成数据的丢失。

6.4.3.3 单次转换模式

当 CFGR1.CONT 位为 0 时，使能单次转换模式；在该模式下，ADC 完成一次所有使能通道上的转换；转换可以由下面两种方式触发：

- 将 ADC_CR 寄存器中的 ADSTART 寄存器置 1
- 硬件信号触发方式

在连续转换中，每次转换完成时：

- 转换的结果被保存到 16bit 寄存器 ADC_DR 中
- 转换完成标志 EOC 会被置 1
- 如果 EOCIE 被置 1，则会产生一次中断

在连续转换完成后：

- 连续转换完成标志会被置 1
- 如果 EOSEQIE 被置 1，则会产生一次中断

最终 ADC 将会停止，除非新的触发信号产生或者 ADSTART 位被重新置 1。

6.4.3.4 循环转换模式(CONT=1)

当 ADC_CFGR1 寄存器中 CONT 位为 1 时，使能循环转换模式；在该模式下，A/D 完成所有使能通道上的转换后，自动的从头开始新一轮相同的转换。开始转换可以由下面两种方式触发：

- 将 ADC_CR 寄存器中的 ADSTART 寄存器置 1
- 硬件信号触发方式

在连续转换中，每次转换完成时：

- 转换的结果被保存到 16bit 寄存器 ADC_DR 中
- 转换完成标志 EOC 会被置 1
- 如果 EOCIE 被置 1，则会产生一次中断

在连续转换完成后：

- 连续转换完成标志会被置 1
- 如果 EOSEQIE 被置 1，则会产生一次中断

转换结束后新的转换会自动开始。

6.4.3.5 非连续转换模式

当 CFGR1. DISCEN 位置 1 时，使能非连续转换模式。该模式下，每完成一次转换，都需要软件或者硬件来触发，例如：

- DISCEN=1，通道 0，3，7，10 被使能

  —第一次触发：通道 0 进行转换，且产生一个 EOC 事件；

  —第二次触发：通道 3 进行转换，且产生一个 EOC 事件；

  —第三次触发：通道 7 进行转换，且产生一个 EOC 事件；

  —第四次触发：通道 10 进行转换，且产生一个 EOC 事件；

  —第五次触发：通道 0 进行转换，且产生一个 EOC 事件；

  —第 6 次触发：通道 3 进行转换，且产生一个 EOC 事件；

  —……

- DISCEN=0，通道 0，3，7，10 被使能

  —第一次触发：将进行一次连续的转换：通道 0—>通道 3—>通道 7—>通道 10，每次转换都会产生一个 EOC 事件，且最后一次转换还会产生一个 EOSEQ 事件。

  —每次触发将会重新开始一个完整的连续转换。

注：DISCEN 和 CONT 不能同时为 1，软件应避免这种配置。

6.4.3.6 ADSTART

软件将 ADSTART 置为 1 来启动转换。当 ADSTART 置为 1，转换将：

- 当 EXTEN=0x0 时(软件触发)，立即启动
- 当 EXTEN≠0x0 时，在下一个有效的硬件触发信号的边沿启动

软件可以通过将 ADSTART 置 0 来将 ADC 还原为 idle 模式。ADCSTART 也会被硬件自动清零：

- 在单次模式且是软件触发方式下（CONT=0，EXTEN=0，且需 EXTSEL=0）

  —在每个连续转换结束时(EOSEQ=1)

在循环转换模式下(CONT=1)，当 EOSEQ 产生时，ADSTART 不会被硬件清零，因为转换会自动重新开始。

当单次模式且是硬件触发方式，ADSTART 也不会硬件清零。

6.4.3.7 外部触发和触发信号选择

硬件信号也可以用来触发 ADC 的转换，当 EXTEN[1:0]≠0x0 时，外部信号触发方式被使能。当有转换进行时，外部触发信号将被忽略。且外部触发方式只有在 ADSTART=1 时才有效。

源	EXTEN[1:0]
硬件触发方式被禁能	00
信号的上升沿触发转换	01
信号的下降沿触发转换	10
信号的上升沿和下降沿都触发转换	11

硬件触发源可以用过 ASYNC_TRG 和 EXTSEL[3:0]来选择，硬件触发源分为两大类，一类是同步触发源，信号源与 ADC 的时钟同步，内部不需要做同步处理，所以触发开始到转换结束的延迟较小；另一类是异步触发源，信号源与 ADC 的时钟不同步，内部逻辑要做同步处理，所以触发开始到转换结束的延迟稍大。触发信号源和寄存器对应关系如下表所示；

表：ADC 触发信号源列表

名字	信号源	{async_trg,extsel[4:0]}
s_trg_source0	mcoA0（mcpwm 的输出）	6’b000000
s_trg_source1	mcoA1（mcpwm 的输出）	6’b000001
s_trg_source2	mcoA2（mcpwm 的输出）	6’b000010
s_trg_source3	mcoB0（mcpwm 的输出）	6’b000011
s_trg_source4	mcoB1（mcpwm 的输出）	6’b000100
s_trg_source5	mcoB2（mcpwm 的输出）	6’b000101
s_trg_source6	ct32b0_mat[0] （timer0 的输出）	6’b000110
s_trg_source7	ct32b0_mat[1] （timer0 的输出）	6’b000111
s_trg_source8	ct32b0_mat[2] （timer0 的输出）	6’b001000
s_trg_source9	ct32b0_mat[3] （timer0 的输出）	6’b001001
s_trg_source10	mcoA0_adc(mcpwm 通道 0 设置的 ADC 触发值)	6’b001010

s_trg_source11	mcoA1_adc(mcpwm 通道 1 设置的 ADC 触发值)	6’b001011
s_trg_source12	mcoA2_adc(mcpwm 通道 2 设置的 ADC 触发值)	6’b001100
s_trg_source13	保留	6’b001101
s_trg_source14	mcpwm_irq（mcpwm 的中断）	6’b001110
s_trg_source15	timer0_irq（timer0 的中断）	6’b001111
s_trg_source16	保留	6’b010000
s_trg_source17	cmp0_0 （比较器 0 的输出）	6’b010001
s_trg_source18	cmp1_0（比较器 1 的输出）	6’b010010
s_trg_source19	cmp_irq(比较器模块的中断)	6’b010011
as_trg_source0	gpio_data_in_0 （gpio0_0）	6’b100000
as_trg_source1	gpio_data_in_1 （gpio0_1）	6’b100001
as_trg_source2	gpio_data_in_2 （gpio0_2）	6’b100010
as_trg_source3	gpio_data_in_3 （gpio0_3）	6’b100011
as_trg_source4	gpio_data_in_4 （gpio0_4）	6’b100100
as_trg_source5	gpio_data_in_5 （gpio0_5）	6’b100101
as_trg_source6	gpio_data_in_6 （gpio0_6）	6’b100110
as_trg_source7	gpio_data_in_7 （gpio0_7）	6’b100111
as_trg_source8	gpio_data_in_8 （gpio1_0）	6’b101000
as_trg_source9	gpio_data_in_9 （gpio1_1）	6’b101001
as_trg_source10	gpio_data_in_10 （gpio1_2）	6’b101010
as_trg_source11	gpio_data_in_11 （gpio1_3）	6’b101011
as_trg_source12	gpio_data_in_12 （gpio1_4）	6’b101100
as_trg_source13	gpio_data_in_13 （gpio1_5）	6’b101101
as_trg_source14	gpio_data_in_14 （gpio1_6）	6’b101110
as_trg_source15	gpio_data_in_15 （gpio1_7）	6’b101111
as_trg_source16	gpio_data_in_16 （gpio2_0）	6’b110000

as_trg_source17	gpio_data_in_17 （gpio2_1）	6’b110001
as_trg_source18	gpio_data_in_18 （gpio2_2）	6’b110010
as_trg_source19	gpio_data_in_19 （gpio2_3）	6’b110011

6.4.3.8 数据管理

6.4.3.8.1 数据寄存器和数据对齐(ADC_CR,ALIGN)

在每次转换结束后（当 EOC=1）,转换结果被暂存到 16-bit 数据寄存器中，ADC_CFGR1 寄存器中的 ALIGN 控制位用于选择 12-bit 结果数据在 16-bit 寄存器中是右对齐模式(ALIGN=0)还是左对齐模式(ALIGN=1)。

ALIGN	RES	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0x0	0x0				DR[11:0]											
	0x1	0x00						DR[9:0]									
	0x2	0x00								DR[7:0]							
	0x3	0x00										DR[5:0]					
1	0x0	DR[11:0]												0x0			
	0x1	DR[9:0]										0x00					
	0x2	DR[7:0]								0x00							
	0x3	0x00								DR[5:0]						0x0	

6.4.3.8.2 数据溢出处理

当新的转换结束，且结果寄存器中的数据没有被 CPU 读走时(EOC=1)，数据发生溢出是，溢出标志位 OVR 被置 1。如果 OVERIE 寄存器被置 1 则将会产生一个中断信号。

当溢出发生时，ADC 会继续进行转换。ADC_CFGR1 寄存器中的 OVRMOD 位用于控制丢弃新的数据还是老的数据。

- OVRMOD=0

—当溢出发生时，新的数据将会被丢弃，老的数据会保留在数据寄存器中。当 OVR 保持为 1 是，新的转换会进行，但结果会被丢弃。

- OVRMOD=1

—数据寄存器会一直保留最新一次转换的数据，老的数据将会被丢弃。

6.4.3.9 模拟看门狗(AWDEN,AWDSGL,AWDCH,AWD_HTR/LTR,AWD)

当寄存器 ADC_CFGR1 寄存器中的 AWDEN 位为 1 时，使能模拟看门狗功能，他用来监测一个或多个通道的电平是否保持在配置的电平窗口之内。当通道的电平低于看门狗的底部阈值，或者高于顶部阈值时，看门狗状态位 AWD 将会被置 1；如果 ADC_IER 寄存器中的 AWDIE 为 1 时，将会产生一个中断。AWD 标志位由软件写 1 清零。

选择通道	AWDSGL	AWDEN
无效	X	0
所有通道	0	1
单个通道（AWDCH[4:0]选择）	1	1

6.4.3.10 温度传感器

ADC 的通道 15 用于连接芯片内部的温度传感器，当需要测试温度时，需要将温度传感器的使能位打开(ADC_CCR 寄存器的 TSEN 位)，在等待至少 10us 后再将通道 15 的使能打开即可(如果等待时间过短，可能会导致第一个采样的值不准确)。

注：芯片当前温度的计算公式为：

$$芯片温度 = \frac{1.457 - \frac{ADC_{code} * VREF}{4095}}{0.0038},$$ 其中 ADC_{code} 是 ADC 通道 15 读取的电压值，

VREF 是 ADC 采样时的参考电压源。

另外，在用 ADC 采样芯片温度时（ADC 通道 15），需要将 ADC 的采样频率设置为≤100KSPS。

6.4.3.11 ADC 中断源

下面的事件可以触发 ADC 中断：

- 任何一次转换结束(EOC 标志)
- 连续转换结束(EOSEQ 标志)
- 模拟看门狗被触发(AWD 标志)
- 溢出事件发生（OVR 标志）

每个触发源都有独立的使能位来灵活配置。

6.4.4 寄存器映射

R：只读，W：只写，R/W：可读写

寄存器	偏移量	R/W	描述	复位后的值
ADC 的基地址： **ADC_BA = 0x4000_5000**				
ISR	ADC_BA+0x000	R	ADC 中断及状态寄存器	0x0000_0000
IER	ADC_BA+0x004	R/W	ADC 中断使能寄存器	0x0000_0000
CR	ADC_BA+0x008	R/W	ADC 控制使能寄存器	0x0000_0000
CFGR1	ADC_BA+0x00C	R/W	ADC 配置寄存器 1	0x0000_0000
CFGR2	ADC_BA+0x010	R/W	ADC 配置寄存器 2	0x0000_0000
TR	ADC_BA+0x020	R/W	ADC 看门狗阈值寄存器	0x0000_0000
CHSELR	ADC_BA+0x028	R/W	ADC 通道选择寄存器	0x0000_0000
DR	ADC_BA+0x040	R	ADC 数据寄存器	0x0000_0000
CCR	ADC_BA+0x308	R/W	ADC 通用配置寄存器	0x0000_0000
ACR0	ADC_BA+0x400	R/W	ADC 附加控制寄存器	0x0000_0000

DR_CH0	ADC_BA+0x410	R	ADC 通道 0 数据寄存器	0x0000_0000
DR_CH1	ADC_BA+0x414	R	ADC 通道 1 数据寄存器	0x0000_0000
DR_CH2	ADC_BA+0x418	R	ADC 通道 2 数据寄存器	0x0000_0000
DR_CH3	ADC_BA+0x41C	R	ADC 通道 3 数据寄存器	0x0000_0000
DR_CH4	ADC_BA+0x420	R	ADC 通道 4 数据寄存器	0x0000_0000
DR_CH5	ADC_BA+0x424	R	ADC 通道 5 数据寄存器	0x0000_0000
DR_CH6	ADC_BA+0x428	R	ADC 通道 6 数据寄存器	0x0000_0000
DR_CH7	ADC_BA+0x42c	R	ADC 通道 7 数据寄存器	0x0000_0000
DR_CH8	ADC_BA+0x430	R	ADC 通道 8 数据寄存器	0x0000_0000
DR_CH9	ADC_BA+0x434	R	ADC 通道 9 数据寄存器	0x0000_0000
DR_CH10	ADC_BA+0x438	R	ADC 通道 10 数据寄存器	0x0000_0000
DR_CH11	ADC_BA+0x43C	R	ADC 通道 11 数据寄存器	0x0000_0000
DR_CH12	ADC_BA+0x440	R	ADC 通道 12 数据寄存器	0x0000_0000
DR_CH13	ADC_BA+0x444	R	ADC 通道 13 数据寄存器	0x0000_0000
DR_CH14	ADC_BA+0x448	R	ADC 通道 14 数据寄存器	0x0000_0000
DR_CH15	ADC_BA+0x44c	R	ADC 通道 15 数据寄存器	0x0000_0000
ACR1	ADC_BA+0x460	R/W	ADC 附加控制寄存器 1	0x0000_0000
ACR2	ADC_BA+0x470	R/W	ADC 附加控制寄存器 2	0x0000_0000
DELAY_CTRL	ADC_BA+0x480	R/W	ADC 延迟控制寄存器	0x0000_0075
STATUS	ADC_BA+0x490	R	ADC 状态寄存器	0x0000_0040

6.4.4.1 ISR 寄存器

寄存器	偏移量	R/W	描述	复位值
ISR	ADC_BA+0x00	R	ADC 中断及状态寄存器	0x0000_0000

31	30	29	28	27	26	25	24
保留							

23	22	21	20	19	18	17	16
保留							
15	14	13	12	11	10	9	8
保留							
7	6	5	4	3	2	1	0
AWD	保留		OVR	EOSEQ	EOC	保留	

Bits	描述	
[31:8]	保留	保留
[7]	AWD	**模拟看门狗被触发标志** 当转换的电压值超过ADC_TR.LTR和HADC_TR.HTR中的阈值时，该bit位被硬件置1。如果要将该bit位清零，软件向该bit位写1。 0：没有模拟看门狗事件发生 1：有看门狗事件发生
[6:5]	保留	保留
[4]	OVR	**溢出标志** 当有溢出事件发生时，该bit位被硬件置1。如果要将该bit位清零，软件向该bit位写1。 0：没有溢出事件发生 1：有溢出事件发生
[3]	EOSEQ	**连续转换结束标志** 当连续转换结束时，该 bit 位被硬件置 1。如果要将该 bit 位清零，软件向该 bit 位写 1。 0：连续转换没有结束 1：连续转换结束
[2]	EOC	**转换结束标志** 当一个通道的转换结束时，硬件将该 bit 位置 1。如果要将该 bit 位清零，软件向该 bit 位写 1。 0：通道转换没有结束

		1：通道转换结束
[1:0]	保留	保留

6.4.4.2 IER 寄存器

寄存器	偏移量	R/W	描述	复位值
IER	ADC_BA+0x04	R/W	ADC 中断使能寄存器	0x0000_0000

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

Bits	描述	
[31:8]	保留	保留
[7]	AWDIE	**模拟看门狗中断使能位** 软件通过 Clear/Set 该 bit 位来禁能/使能模拟看门狗中断 0：看门狗中断被禁能 1：看门狗中断被使能
[6:5]	保留	保留
[4]	OVRIE	**溢出中断使能位** 软件通过 Clear/Set 该 bit 位来禁能/使能溢出中断 0：溢出中断被禁能 1：溢出中断被使能

[3]	EOSEQIE	**连续转换结束中断使能位** 软件通过 Clear/Set 该 bit 位来禁能/使能连续转换中断 0：连续转换中断被禁能 1：连续转换中断被使能
[2]	EOCIE	**转换结束中断使能位** 软件通过 Clear/Set 该 bit 位来禁能/使能转换中断 0：转换中断被禁能 1：转换中断被使能
[1:0]	保留	保留

6.4.4.3 CR 寄存器

寄存器	偏移量	R/W	描述	复位值
CR	ADC_BA+0x08	R/W	ADC 控制使能寄存器	0x0000_0000

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

Bits	描述	
[31:3]	保留	保留
[2]	ADSTART	**ADC 开始转换命令** 软件通过将该 bit 位置 1 来开始 ADC 转换。如果在软件触发模式

		下，ADC 会立刻进行转换；当在硬件触发模式下，转换会等到触发信号发生时才开始转换。 该 bit 位被在以下情况下被硬件清零： —在单次转换模式下，选择软件触发方式（EXTEN=0x0），当 EOSEQ 标志产生时。
[1:0]	保留	保留

6.4.4.4 CFGR1 寄存器

寄存器	偏移量	R/W	描述	复位值
CFGR1	ADC_BA+0x0C	R/W	ADC 配置寄存器 1	0x0000_0000

31	30	29	28	27	26	25	24
ASYNC_TRG	AWDCH					保留	EXTSEL[4]
23	22	21	20	19	18	17	16
AWDEN	AWDSG1						DISCEN
15	14	13	12	11	10	9	8
保留	WAIT	CONT	OVRMOD	EXTEN		EXTSEL[3:2]	
7	6	5	4	3	2	1	0
EXTSEL[1:0]		ALIGN	RES		SCANDIR	保留	FLASH

Bits	描述	
[31]	ASYNC_TRG	**同步触发/异步触发信号选择** 该 bit 位于 EXTSEL[3:0]一起用于选择外部触发源 0：选择同步触发源 1：选择异步触发源
[30:26]	AWDCH	**模拟看门狗的通道选择** 该 bit 域由软件置位或清零，用于选择哪个输入通道与 AWD 进

		行比较 x0000:通道 0 的转换值被 AWD 监测; x0001:通道 1 的转换值被 AWD 监测; x0010:通道 2 的转换值被 AWD 监测; …… x1111:通道 15 的转换值被 AWD 监测;
[25]	保留	保留
[24]	EXTSEL[4]	**触发源选择 EXTSEL[4]** 该 bit 域和 ASYNC_TRG 一起用于选择外部触发源
[23]	AWDEN	**模拟看门狗使能** 该 bit 位由软件置位或清零 0：模拟看门狗被禁能 1：模拟看门狗被使能
[22]	AWDSG1	**选择模拟看门狗监控单个通道还是所有通道** 该 bit 位由软件置位或清零用于选择看门狗监测单个通道（由 AWDCH[3:0]选择）还是所有通道 0: 模拟看门狗监测所有通道 1：模拟看门狗监测单个通道（由 AWDCH[3:0]选择）
[21:17]	保留	保留
[16]	DISCEN	**非连续转换使能位** 该 bit 位由软件置位或清零用于使能/禁能非连续转换模式 0：非连续转换模式禁能 1：非连续转换模式使能
[15]	保留	保留
[14]	WAIT	**等待转换模式** 该 bit 位由软件置位或清零用于使能/禁能等待转换模式 0：等待转换模式禁能 1：等待转换模式使能，在该模式下，新的转换必须等到上一次转换的数据被读走后才能触发(即 EOC=0 时)

[13]	CONT	**选择单次转换还是连续转换** 该 bit 位由软件置位或清零用于选择单次转换模式或者循环转换模式 0：单次转换 1：循环转换
[12]	OVRMOD	**溢出时数据处理控制位** 该 bit 位由软件置位或清零用于配置发生溢出时数据的处理模式 0：溢出时，保留老的数据，新数据被丢弃 1：溢出时，老的数据被丢弃，新的数据被装入数据寄存器中
[11:10]	EXTEN	**触发信号边沿选择** 00：软件触发方式 01：上升沿触发 10：下降沿触发 11：上升沿和下降沿触发
[9：6]	EXTSEL[3:0]	**触发源选择 EXTSEL[3:0]** 该 bit 域和 ASYNC_TRG 一起用于选择外部触发源
[5]	ALIGN	**数据寄存器左右对齐选择** 0：右对齐 1：左对齐
[4:3]	RES	**转换精度选择** 该 bit 域由软件置位或清零，用于配置 ADC 的采样精度，但需要注意的时，ADC 是固定的 12bit 精度采样，在低精度时，只是舍去了部分 bit 位。所以降低转换精度并不会提高转换速率。 00：12bit 精度 01：10bit 精度 10：8bit 精度 11：6bit 精度
[2]	SCANDIR	**连续扫描转换的方向选择位**

		0：由通道 0 到通道 15 1：由通道 15 到通道 0
[1：0]	保留	保留

6.4.4.5 CFGR2 寄存器

寄存器	偏移量	R/W	描述	复位值
CFGR2	ADC_BA+0x10	R/W	ADC 配置寄存器 2	0x0000_0000

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

Bits	描述	
[31:30]	CLKMODE	**时钟模式选择** 该 bit 域用于选择模拟部分 ADC 转换的时钟源，为 00 时 ADC_CLK 是从 main_clk 分频下来的时钟，为 01 和 10 时都是 HCLK 分频出来的时钟。ADC 的采样频率是该时钟源的 1/14。 00：ADC_CLK 01：HCLK/2 10：HCLK/4 11：保留
[29:0]	保留	保留

6.4.4.6 TR 寄存器

寄存器	偏移量	R/W	描述	复位值
TR	ADC_BA+0x20	R/W	ADC 看门狗阈值寄存器	0x0000_0000

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

Bits	描述	
[31:28]	保留	保留
[27:16]	HT	**模拟看门狗顶部阈值** 该 bit 域由软件写入模拟看门狗的顶部阈值
[15:13]	保留	保留
[11:0]	LT	**模拟看门狗底部阈值** 该 bit 域由软件写入模拟看门狗的低部阈值

6.4.4.7 CHSELR 寄存器

寄存器	偏移量	R/W	描述	复位值
CHSELR	ADC_BA+0x28	R/W	ADC 通道选择寄存器	0x0000_0000

31	30	29	28	27	26	25	24

保留							
23	22	21	20	19	18	17	16
保留							
15	14	13	12	11	10	9	8
CHSEL15	CHSEL14	CHSEL13	CHSEL12	CHSEL11	CHSEL10	CHSEL9	CHSEL8
7	6	5	4	3	2	1	0
CHSEL7	CHSEL6	CHSEL5	CHSEL4	CHSEL3	CHSEL2	CHSEL1	CHSEL0

Bits	描述	
[31:16]	保留	保留
[15]	CHSEL15	**通道 15 使能位** 该 bit 置 1 时，通道 15 使能
[14]	CHSEL14	**通道 14 使能位** 该 bit 置 1 时，通道 14 使能
[13]	CHSEL13	**通道 13 使能位** 该 bit 置 1 时，通道 13 使能
[12]	CHSEL12	**通道 12 使能位** 该 bit 置 1 时，通道 12 使能
[11]	CHSEL11	**通道 11 使能位** 该 bit 置 1 时，通道 11 使能
[10]	CHSEL10	**通道 10 使能位** 该 bit 置 1 时，通道 10 使能
[9]	CHSEL9	**通道 9 使能位** 该 bit 置 1 时，通道 9 使能
[8]	CHSEL8	**通道 8 使能位** 该 bit 置 1 时，通道 8 使能
[7]	CHSEL7	**通道 7 使能位** 该 bit 置 1 时，通道 7 使能
[6]	CHSEL6	**通道 6 使能位**

		该 bit 置 1 时，通道 6 使能
[5]	CHSEL5	**通道 5 使能位** 该 bit 置 1 时，通道 5 使能
[4]	CHSEL4	**通道 4 使能位** 该 bit 置 1 时，通道 4 使能
[3]	CHSEL3	**通道 3 使能位** 该 bit 置 1 时，通道 3 使能
[2]	CHSEL2	**通道 2 使能位** 该 bit 置 1 时，通道 2 使能
[1]	CHSEL1	**通道 1 使能位** 该 bit 置 1 时，通道 1 使能
[0]	CHSEL0	**通道 0 使能位** 该 bit 置 1 时，通道 0 使能

6.4.4.8 MISR 寄存器

寄存器	偏移量	R/W	描述	复位值
MISR	ADC_BA+0x30	R	ADC 屏蔽后中断状态寄存器	0x0000_0000

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

Bits	描述	
[31:8]	保留	保留
[7]	AWD	**模拟看门狗被触发标志** 当转换的电压值超过ADC_TR.LTR和HADC_TR.HTR中的阈值时，且中断使能打开，该bit位被硬件置1。如果要将该bit位清零，软件向该bit位写1。 0：没有相应中断标志位 1：相应中断标志位被置1
[6:5]	保留	保留
[4]	OVR	**溢出标志** 当有溢出事件发生时，且中断使能打开，该bit位被硬件置1。如果要将该bit位清零，软件向该bit位写1。 0：没有相应中断标志位 1：相应中断标志位被置1
[3]	EOSEQ	**连续转换结束标志** 当连续转换结束时，且中断使能打开，该 bit 位被硬件置 1。如果要将该 bit 位清零，软件向该 bit 位写 1。 0：没有相应中断标志位 1：相应中断标志位被置 1
[2]	EOC	**转换结束标志** 当一个通道的转换结束时，且中断使能打开，硬件将该 bit 位置 1。如果要将该 bit 位清零，软件向该 bit 位写 1。 0：没有相应中断标志位 1：相应中断标志位被置 1
[1:0]	保留	保留

6.4.4.9 DR 寄存器

寄存器	偏移量	R/W	描述	复位值

DR	ADC_BA+0x40	R	ADC 数据寄存器	0x0000_0000

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

Bits	描述	
[31:16]	保留	保留
[15:0]	DATA	16bit 结果寄存器 该寄存器为只读寄存器，此寄存器用于保存最近一次 ADC 转换的结果

6.4.4.10 CCR 寄存器

寄存器	偏移量	R/W	描述	复位值
CCR	ADC_BA+0x308	R/W	ADC 通用配置寄存器	0x0000_0000

31	30	29	28	27	26	25	24
保留							
23	22	21	20	19	18	17	16
TSEN	保留	VREF	保留				
15	14	13	12	11	10	9	8
保留							

7	6	5	4	3	2	1	0
保留					OP1_CH13	OP0_CH12	保留

Bits	描述	
[31:24]	保留	保留
[23]	TSEN	**温度传感器使能位** 该 bit 位置 1 使能温度传感器。 最好在该 bit 位置 1 后至少等 10us 后再开启通道 15 进行采样。
[22]	保留	保留
[21]	VREF	**ADC 参考电压的选择** 0:选择芯片电源 VDD 1:选择芯片内部基准 VREF（2.0V） 如果选择片内基准，ADC 采样率要降到 100KHZ 以下
[20:3]	保留	保留
[2]	OP1_CH13	**0：通道 13 选择 ADC IO 通道** **1：通道 13 选择 OP1 的输出**
[1]	OP0_CH12	**0：通道 12 选择 ADC IO 通道** **1：通道 12 选择 OP0 的输出**
[0]	保留	保留

6.4.4.11 ACR0 寄存器

寄存器	偏移量	R/W	描述	复位值
ACR0	ADC_BA+0x400	R/W	ADC 附加控制寄存器 0	0x0000_0000

31	30	29	28	27	26	25	24
保留							
23	22	21	20	19	18	17	16

保留							
15	14	13	12	11	10	9	8
7	6	5	4	3	2	1	0
保留		DISCARD1		保留		DISCARD0	

Bits	描述	
[31:24]	保留	保留
[5:4]	DISCARD1	**切换通道时丢弃的个数** 该 bit 域配置通道切换时，需要丢失的数据个数。如:使能了通道 0、1、2，且 SCANDIR=0 时，在通道 0 转换完，进行通道 1 转换时，通道 1 丢弃数据的个数由该 bit 域控制；在通道 1 转换完，进行通道 2 转换时，通道 2 丢弃数据的个数也由该 bit 域控制。采用硬件触发时，需要配置为 01。 00：不丢弃 01：丢弃 1 个 10：丢弃 2 个 11：丢弃 3 个
[3:2]	保留	保留
[1:0]	DISCARD0	**第一次转换丢弃个数** 第一次转换指：使能 adstart 后的刚开始的第一次转换，或者是一次连续的转换结束后，再一次新的转换，即 EOSEQ 变换 1 后的第一次转换。采用硬件触发时，需要配置为 01。 00：不丢弃 01：丢弃 1 个 10：丢弃 2 个 11：丢弃 3 个

6.4.4.12 ACR1 寄存器

寄存器	偏移量	R/W	描述	复位值
ACR1	ADC_BA+0x460	R/W	ADC 附加控制寄存器 0	0x0000_0000

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

Bits	描述	
[31:16]	保留	保留
[15:14]	SELRES	ADC 转换精度选择 00：12bit 01：10bit 10：8bit 11：6bit
[13]	保留	保留
[12]	保留	保留
[11]	MULTICAL	1：使能斜率校准，校准值使用 ACR2 寄存器中的值 0：不使能斜率校准
[10]	保留	保留
[9]	LOADCAL	加载用户校准值 1：加载零点校准值（BVOSI）到内部寄存器 0：使用默认值

[8:7]	保留	保留
[6:0]	BVOSI	ADC 的失调校准系数，上电时从 Flash 中自动加载失调系数，也可以软件输入失调系数。

6.4.4.13 ACR2 寄存器

寄存器	偏移量	R/W	描述	复位值
ACR0	ADC_BA+0x470	R/W	ADC 附加控制寄存器 0	0x0000_0000

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

Bits	描述	
[31:12]	保留	保留
[11:0]	MULTI_SINGLE	斜率校准的乘法系数

6.4.4.14 DELAY_CTRL 寄存器

寄存器	偏移量	R/W	描述	复位值
DELAY_CTRL	ADC_BA+0x480	R/W	ADC 附加控制寄存器 0	0x0000_0005

31	30	29	28	27	26	25	24
保留							
23	22	21	20	19	18	17	16

保留							
15	14	13	12	11	10	9	8
保留							
7	6	5	4	3	2	1	0
保留				EOC			

Bits	描述	
[31:4]	保留	保留
[3:0]	EOC	控制模拟部分采样结束信号 EOC 的延迟 EOC 的值配置为 N-1，N=AHBCLK/ADCCLK 举例:当 ADCCLK=12M，AHBCLK=12M 时，EOC 配置为 0 当 ADCCLK=12M，AHBCLK=24M 时，EOC 配置为 1 当 ADCCLK=96M，AHBCLK=12M 时，EOC 配置为 7

6.4.4.15 DR_CH0R 寄存器

寄存器	偏移量	R/W	描述	复位值
DR_CH0	ADC_BA+0x410	R	ADC 通道 0 数据寄存器	0x0000_0000

31	30	29	28	27	26	25	24
保留							
23	22	21	20	19	18	17	16
保留							
15	14	13	12	11	10	9	8
DR_CH0							
7	6	5	4	3	2	1	0
DR_CH0							

Bits	描述

[31:16]	保留	保留
[15:0]	DR_CH0	通道 0 的 16bit 结果寄存器

6.4.4.16 DR_CH1R 寄存器

寄存器	偏移量	R/W	描述	复位值
DR_CH1	ADC_BA+0x414	R	ADC 通道 1 数据寄存器	0x0000_0000

31	30	29	28	27	26	25	24
保留							
23	22	21	20	19	18	17	16
保留							
15	14	13	12	11	10	9	8
DR_CH1							
7	6	5	4	3	2	1	0
DR_CH1							

Bits	描述	
[31:16]	保留	保留
[15:0]	DR_CH1	通道 1 的 16bit 结果寄存器

6.4.4.17 DR_CH2R 寄存器

寄存器	偏移量	R/W	描述	复位值
DR_CH2	ADC_BA+0x418	R	ADC 通道 2 数据寄存器	0x0000_0000

31	30	29	28	27	26	25	24
保留							
23	22	21	20	19	18	17	16
保留							

15	14	13	12	11	10	9	8
DR_CH2							
7	6	5	4	3	2	1	0
DR_CH2							

Bits	描述	
[31:16]	保留	保留
[15:0]	DR_CH2	通道 2 的 16bit 结果寄存器

6.4.4.18 DR_CH3R 寄存器

寄存器	偏移量	R/W	描述	复位值
DR_CH3	ADC_BA+0x41C	R	ADC 通道 3 数据寄存器	0x0000_0000

31	30	29	28	27	26	25	24
保留							
23	22	21	20	19	18	17	16
保留							
15	14	13	12	11	10	9	8
DR_CH3							
7	6	5	4	3	2	1	0
DR_CH3							

Bits	描述	
[31:16]	保留	保留
[15:0]	DR_CH3	通道 3 的 16bit 结果寄存器

6.4.4.19 DR_CH4R 寄存器

寄存器	偏移量	R/W	描述	复位值
DR_CH4	ADC_BA+0x420	R	ADC 通道 4 数据寄存器	0x0000_0000

31	30	29	28	27	26	25	24
保留							
23	22	21	20	19	18	17	16
保留							
15	14	13	12	11	10	9	8
DR_CH4							
7	6	5	4	3	2	1	0
DR_CH4							

Bits	描述	
[31:16]	保留	保留
[15:0]	DR_CH4	通道 4 的 16bit 结果寄存器

6.4.4.20 DR_CH5R 寄存器

寄存器	偏移量	R/W	描述	复位值
DR_CH5	ADC_BA+0x424	R	ADC 通道 5 数据寄存器	0x0000_0000

31	30	29	28	27	26	25	24
保留							
23	22	21	20	19	18	17	16
保留							
15	14	13	12	11	10	9	8

DR_CH5							
7	6	5	4	3	2	1	0
DR_CH5							

Bits	描述	
[31:16]	保留	保留
[15:0]	DR_CH5	通道 5 的 16bit 结果寄存器

6.4.4.21 DR_CH6R 寄存器

寄存器	偏移量	R/W	描述	复位值
DR_CH6	ADC_BA+0x428	R	ADC 通道 6 数据寄存器	0x0000_0000

31	30	29	28	27	26	25	24
保留							
23	22	21	20	19	18	17	16
保留							
15	14	13	12	11	10	9	8
DR_CH6							
7	6	5	4	3	2	1	0
DR_CH6							

Bits	描述	
[31:16]	保留	保留
[15:0]	DR_CH6	通道 6 的 16bit 结果寄存器

6.4.4.22 DR_CH7R 寄存器

寄存器	偏移量	R/W	描述	复位值
DR_CH7	ADC_BA+0x42C	R	ADC 通道 7 数据寄存器	0x0000_0000

31	30	29	28	27	26	25	24
保留							
23	22	21	20	19	18	17	16
保留							
15	14	13	12	11	10	9	8
DR_CH7							
7	6	5	4	3	2	1	0
DR_CH7							

Bits	描述	
[31:16]	保留	保留
[15:0]	DR_CH7	通道 7 的 16bit 结果寄存器

6.4.4.23 DR_CH8R 寄存器

寄存器	偏移量	R/W	描述	复位值
DR_CH8	ADC_BA+0x430	R	ADC 通道 8 数据寄存器	0x0000_0000

31	30	29	28	27	26	25	24
保留							
23	22	21	20	19	18	17	16
保留							
15	14	13	12	11	10	9	8

DR_CH8							
7	6	5	4	3	2	1	0
DR_CH8							

Bits	描述	
[31:16]	保留	保留
[15:0]	DR_CH8	通道 8 的 16bit 结果寄存器

6.4.4.24 DR_CH9R 寄存器

寄存器	偏移量	R/W	描述	复位值
DR_CH9	ADC_BA+0x434	R	ADC 通道 9 数据寄存器	0x0000_0000

31	30	29	28	27	26	25	24
保留							
23	22	21	20	19	18	17	16
保留							
15	14	13	12	11	10	9	8
DR_CH9							
7	6	5	4	3	2	1	0
DR_CH9							

Bits	描述	
[31:16]	保留	保留
[15:0]	DR_CH9	通道 9 的 16bit 结果寄存器

6.4.4.25 DR_CH10R 寄存器

寄存器	偏移量	R/W	描述	复位值
DR_CH10	ADC_BA+0x438	R	ADC 通道 10 数据寄存器	0x0000_0000

31	30	29	28	27	26	25	24
保留							
23	22	21	20	19	18	17	16
保留							
15	14	13	12	11	10	9	8
DR_CH10							
7	6	5	4	3	2	1	0
DR_CH10							

Bits	描述	
[31:16]	保留	保留
[15:0]	DR_CH10	通道 10 的 16bit 结果寄存器

6.4.4.26 DR_CH11R 寄存器

寄存器	偏移量	R/W	描述	复位值
DR_CH11	ADC_BA+0x43C	R	ADC 通道 11 数据寄存器	0x0000_0000

31	30	29	28	27	26	25	24
保留							
23	22	21	20	19	18	17	16
保留							
15	14	13	12	11	10	9	8

DR_CH11							
7	6	5	4	3	2	1	0
DR_CH11							

Bits	描述	
[31:16]	保留	保留
[15:0]	DR_CH11	通道 11 的 16bit 结果寄存器

6.4.4.27 DR_CH12R 寄存器

寄存器	偏移量	R/W	描述	复位值
DR_CH12	ADC_BA+0x440	R	ADC 通道 12 数据寄存器	0x0000_0000

31	30	29	28	27	26	25	24
保留							
23	22	21	20	19	18	17	16
保留							
15	14	13	12	11	10	9	8
DR_CH12							
7	6	5	4	3	2	1	0
DR_CH12							

Bits	描述	
[31:16]	保留	保留
[15:0]	DR_CH12	通道 12 的 16bit 结果寄存器

6.4.4.28 DR_CH13R 寄存器

寄存器	偏移量	R/W	描述	复位值
DR_CH13	ADC_BA+0x444	R	ADC 通道 13 数据寄存器	0x0000_0000

31	30	29	28	27	26	25	24
保留							
23	22	21	20	19	18	17	16
保留							
15	14	13	12	11	10	9	8
DR_CH13							
7	6	5	4	3	2	1	0
DR_CH13							

Bits	描述	
[31:16]	保留	保留
[15:0]	DR_CH13	通道 13 的 16bit 结果寄存器

6.4.4.29 DR_CH14R 寄存器

寄存器	偏移量	R/W	描述	复位值
DR_CH14	ADC_BA+0x448	R	ADC 通道 14 数据寄存器	0x0000_0000

31	30	29	28	27	26	25	24
保留							
23	22	21	20	19	18	17	16
保留							
15	14	13	12	11	10	9	8

DR_CH14							
7	6	5	4	3	2	1	0
DR_CH14							

Bits	描述	
[31:16]	保留	保留
[15:0]	DR_CH14	通道 14 的 16bit 结果寄存器

6.4.4.30 DR_CH15R 寄存器

寄存器	偏移量	R/W	描述	复位值
DR_CH15	ADC_BA+0x44C	R	ADC 通道 15 数据寄存器	0x0000_0000

31	30	29	28	27	26	25	24
保留							
23	22	21	20	19	18	17	16
保留							
15	14	13	12	11	10	9	8
DR_CH15							
7	6	5	4	3	2	1	0
DR_CH15							

Bits	描述	
[31:16]	保留	保留
[15:0]	DR_CH15	通道 15 的 16bit 结果寄存器

6.5 Flash 内存控制器(FMC)

6.5.1 概述

MT32F005/006 系列 MCU 具有 32K 字节的片上 Main Flash，用于存储应用程序。此外还有 1K 字节的 Info Flash，可用来存放数据，以及 512Byte 的 Sys Flash，用来存放与芯片相关的信息。

MT32F005/006 系列 MCU 中的 Flash 的数据宽度为 32 位，因此对 Flash 的读写操作都是以 32 位为单位进行的。

Flash 存储器由若干个 sector 组成。每个 Sector 的容量为 1K 字节，深度为 256，宽度为 4 字节。对于 32K 字节的 Main Flash，它由 32 个 sector 组成，对于 1K 字节的 Info Flash,它由 1 个 sector 组成。

6.5.2 特性

- 所有嵌入的 Flash 存储器支持 1K 字节的页擦除模式
- 32K 字节的程序存储空间（Main Flash）
- Flash 写操作基本单位为 32 位，Flash 块擦除基本单位为 1K 字节
- 具有 Flash 程序加密模式，可以禁止非法读取 Flash 中的内容

6.5.3 Flash 存储器组织结构

Flash 存储器具有 32K 字节的 Main Flash 以及 1K 字节的 Info Flash。它们在物理上的地址空间如下表所示：

Flash 区域	Flash 内存地址	大小(Bytes)	描述
Main	0x08000000-0x080003FF	1Kbyte	Sector0
	0x08000400-0x080007FF	1Kbyte	Sector1

	. . .	. . .	. . .
	0x08007C00-0x08007FFF	1Kbyte	Sector31
Info	0x1FFFF400-0x1FFFF7FF	1Kbyte	Info sector0
Sys	0x1FFFF800-0x1FFFF9FF	512 byte	Sys sector0

表： Flash 存储器结构

MT32F005/006系列MCU的FLASH内存由Main Flash，Info Flash及Sys Flash区域构成。Main Flash是用户应用的主要内存，用户可以写他们的应用程序到Main Flash，Info Flash可以用来存放客户特定的数据，Sys Flash区域提供几个字节来控制系统逻辑，比如Flash安全锁，启动选择，开门狗使能，时钟以及LDO校准值等。用户配置如同是上电设置的保险，上电时用户配置会自动从FLASH存储器装载到它相应的控制寄存器中去。

6.5.4 寄存器映射

R：只读，W：只写，R/W：可读写

寄存器	偏移量	R/W	描述	复位后的值
FMC 的基地址： **FMC_BA = 0x4000_B000**				
ACR	FMC_BA+0x00	R/W	Flash读访问控制寄存器	0x0000_0000
KEYR	FMC_BA+0x04	W	Flash写保护寄存器	0x0000_0000
ERASER	FMC_BA+0x08	R/W	Flash块擦除控制寄存器	0x0000_0000
SR	FMC_BA+0x0C	R/W	状态寄存器	0x0000_0000
CR	FMC_BA+0x10	R/W	Flash擦写访问控制寄存器	0x8000_0000
INFO0	FMC_BA+0x28	RO	NVR区域地址0的数据	NA
INFO1	FMC_BA+0x2C	RO	NVR区域地址1的数据	NA
INFO2	FMC_BA+0x30	RO	NVR区域地址2的数据	NA
INFO3	FMC_BA+0x34	RO	NVR区域地址3的数据	NA

ID0	FMC_BA+0x38	RO	NVR区域地址4的数据	NA
ID1	FMC_BA+0x3C	RO	NVR区域地址5的数据	NA
ID2	FMC_BA+0x40	RO	NVR区域地址6的数据	NA
ID3	FMC_BA+0x44	RO	NVR区域地址7的数据	NA

6.5.4.1 ACR 寄存器

该寄存器用来控制对 Flash 的读访问。

寄存器	偏移量	R/W	描述	复位值
ACR	**FMC_BA+0x00**	**R/W**	**Flash读访问控制寄存器**	**0x0000_0000**

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

Bits	描述	
[31:8]	保留	保留
[6:4]	DLY_NUM	调节 DLY CELL 的个数，不要随意改动这个值
[3]	保留	保留
[2:0]	LATENCY	设置 FLASH 读操作的延时值 由于 Flash 本身只能工作在 24MHZ 频率下，如果系统时钟超过 24MHZ，就要设置相应的 LATENCY 值来满足 FLASH 的时序要求 0<HCLK<=24MHZ: LATENCY = 0

		24<HCLK<=48MHZ: LATENCY = 1 48<HCLK<=72MHZ: LATENCY = 2 72<HCLK<=96MHZ: LATENCY = 3 96<HCLK<=120MHZ: LATENCY = 4 ***在改变 LATENCY 值时需要注意切换流程：*** 当要提升系统频率时，首先将 LATENCY 值设为正确值，然后再提升系统时钟

6.5.4.2 KEYR 寄存器

该寄存器用来控制对 Flash 的读访问。

寄存器	偏移量	R/W	描述	复位值
KEYR	**FMC_BA+0x04**	**W**	Flash写保护寄存器	**0x0000_0000**

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

Bits	描述	
[31:0]	KEY	**设置写保护键值** CR 寄存器上电后默认是不允许操作的，必须先向 KEYR 寄存器中写入正确的值解锁后才能配置 CR 寄存器。 解锁顺序：1）向 keyr 寄存器中写入 0x76543210 2）向 keyr 寄存器中写入 0xfedcba98

6.5.4.3 ERASER 寄存器

该寄存器用来控制对 Flash 进行 Sector 擦除时所要擦出的 Sector 号。

寄存器	偏移量	R/W	描述	复位值
ERASER	**FMC_BA+0x08**	**R/W**	Flash块擦除控制寄存器	**0x0000_0000**

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

Bits	描述	
[31:7]	保留	保留
[6:0]	NUM	设置对 Flash 进行 Sector 擦除时所要擦除的 Sector 号 0000000: 0 号 sector 0000001: 1 号 sector …… 1111111: 127 号 sector

6.5.4.4 SR 寄存器

该寄存器为 FMC 的状态寄存器。

寄存器	偏移量	R/W	描述	复位值
SR	**FMC_BA+0x0C**	**R/W**	状态寄存器	**0x0000_0000**

31	30	29	28	27	26	25	24

保留							
23	22	21	20	19	18	17	16
							BUSY
15	14	13	12	11	10	9	8
保留							
7	6	5	4	3	2	1	0
保留						EOE	EOP

Bits	描述	
[31:17]	保留	保留
[16]	BUSY	表示 FMC 的状态 1：FMC 不空闲，有操作正在进行 0：FMC 空闲
[15:2]	保留	保留
[1]	EOE	当进行 erase 操作时，读这位可以知道 erase 操作的状态 读操作： 1：ERASE 操作结束 0：ERASE 操作还在进行中 写 1 可以清零这个位
[0]	EOP	当进行 program 操作时，读这位可以知道 program 操作的状态 读操作： 1：PROGRAM 操作结束 0：PROGRAM 操作还在进行中 写 1 可以清零这个位

6.5.4.5 CR 寄存器

该寄存器为 FLASH 擦写操作的控制寄存器。

寄存器	偏移量	R/W	描述	复位值
CR	**FMC_BA+0x10**	**R/W**	Flash擦写访问控制寄存器	**0x8000_0000**

31	30	29	28	27	26	25	24
LOCK	保留					EOEIE	EOPIE
23	22	21	20	19	18	17	16
保留							START
15	14	13	12	11	10	9	8
保留							
7	6	5	4	3	2	1	0
保留				INFO	MER	SER	PG

Bits	描述	
[31]	LOCK	**表示 CR 寄存器是否处于写保护状态** 1：表示 CR 寄存器处于写保护状态，此时无法写入 CR 寄存器 0：表示 CR 寄存器处于可配置状态 为了避免对 Flash 进行误操作，CR 寄存器默认处于写保护状态，只有对 KEYR 寄存器进行正确的操作之后，CR 寄存器才会处于可写状态，此时 LOCK 位会变为 0。 可以通过对 LOCK 位重新写入 1 让 CR 寄存器进入写保护状态
[30:26]	保留	保留
[25]	EOEIE	**使能 ERASE 操作中断** 1：当 ERASE 操作结束后产生中断 0：ERASE 操作不产生中断
[24]	EOPIE	**使能 PROGRAM 操作中断** 1：当 PROGRAM 操作结束后产生中断 0：PROGRAM 操作不产生中断
[23:17]	保留	保留
[16]	START	**启动操作** 1：启动 ERASE 或 PROGRAM 操作 0：不启动 在设置好所需的 MER,SER 或 PG 位后需要将 START 设为 1 才会启动操作，START 位在当前操作结束后会自动清零
[15:8]	保留	保留
[7:4]	保留	保留位
[3]	INFO	在擦除或者写 NVR Sector 时必须首先将这位设为 1，在擦除或写操作结束时这位会自动清零

[2]	MER	**表示进行 MAIN 区域擦除** 1：表示擦除整个 MAIN 区域 0：不进行 MAIN 区域整个擦除 设为 1 进行 MAIN 区域一次性擦除操作，该操作会将 MAIN 区域整个擦除
[1]	SER	**表示进行 SECTOR 擦除** 1：表示进行 SECTOR 擦除操作 0：不进行 SECTOR 擦除操作 设为 1 时会进行 SECTOR 擦除操作，所擦除的 SECTOR 由 ERASER 寄存器、SYS、INFO 位共同决定。 SER 位和 MER 位同时只能有一个为 1
[0]	PG	**表示进行编程操作** 1：表示进行 Flash 编程操作 0：不进行 Flash 编程操作 设为 1 后，可以对一个物理地址写入一个 32 位的 WORD。 ***在对一个 SECTOR 中的 WORD 进行编程前一定要先对这个 WORD 所在的 SECTOR 进行擦除操作***

6.5.4.6 INFO0 寄存器

该寄存器为 NVR 区域地址 0 上电后读出的数据保存寄存器。

寄存器	偏移量	R/W	描述	复位值
INFO0	**FMC_BA+0x28**	**RO**	INFO0数据寄存器	**NA**

31	30	29	28	27	26	25	24
INFO0							
23	22	21	20	19	18	17	16
INFO0							
15	14	13	12	11	10	9	8
INFO0							
7	6	5	4	3	2	1	0
INFO0							

Bits	描述	
[31:0]	INFO0	保存 NVR 区域地址 0 处的数据

6.5.4.7 INFO1 寄存器

该寄存器为 NVR 区域地址 1 上电后读出的数据保存寄存器。

寄存器	偏移量	R/W	描述	复位值
INFO1	**FMC_BA+0x2C**	**RO**	INFO1数据寄存器	**NA**

31	30	29	28	27	26	25	24
INFO1							
23	22	21	20	19	18	17	16
INFO1							
15	14	13	12	11	10	9	8
INFO1							
7	6	5	4	3	2	1	0
INFO1							

Bits	描述	
[31:0]	INFO1	保存 NVR 区域地址 1 处的数据

6.5.4.8 INFO2 寄存器

该寄存器为 NVR 区域地址 3 上电后读出的数据保存寄存器。

寄存器	偏移量	R/W	描述	复位值
INFO2	**FMC_BA+0x30**	**RO**	INFO2数据寄存器	**NA**

31	30	29	28	27	26	25	24

INFO2							
23	22	21	20	19	18	17	16
INFO2							
15	14	13	12	11	10	9	8
INFO2							
7	6	5	4	3	2	1	0
INFO2							

Bits	描述	
[31:0]	INFO2	保存 NVR 区域地址 2 处的数据

6.5.4.9 INFO3 寄存器

该寄存器为 NVR 区域地址 3 上电后读出的数据保存寄存器。

寄存器	偏移量	R/W	描述	复位值
INFO3	**FMC_BA+0x34**	**RO**	INFO3数据寄存器	**NA**

31	30	29	28	27	26	25	24
INFO3							
23	22	21	20	19	18	17	16
INFO3							
15	14	13	12	11	10	9	8
INFO3							
7	6	5	4	3	2	1	0
INFO3							

Bits	描述	
[31:0]	INFO3	保存 NVR 区域地址 3 处的数据

6.5.4.10 ID0 寄存器

该寄存器为 NVR 区域地址 4 上电后读出的芯片 ID 数据寄存器。

寄存器	偏移量	R/W	描述	复位值
ID0	**FMC_BA+0x38**	**RO**	**ID0数据寄存器**	**NA**

31	30	29	28	27	26	25	24
ID0							
23	22	21	20	19	18	17	16
ID0							
15	14	13	12	11	10	9	8
ID0							
7	6	5	4	3	2	1	0
ID0							

Bits	描述	
[31:0]	ID0	保存 NVR 区域地址 4 处的数据

6.5.4.11 ID1 寄存器

该寄存器为 NVR 区域地址 5 上电后读出的芯片 ID 数据寄存器。

寄存器	偏移量	R/W	描述	复位值
ID1	**FMC_BA+0x3C**	**RO**	**ID1数据寄存器**	**NA**

31	30	29	28	27	26	25	24
ID1							
23	22	21	20	19	18	17	16
ID1							

15	14	13	12	11	10	9	8
ID1							
7	6	5	4	3	2	1	0
ID1							

Bits	描述	
[31:0]	ID1	保存 NVR 区域地址 5 处的数据

6.5.4.12 ID2 寄存器

该寄存器为 NVR 区域地址 6 上电后读出的芯片 ID 数据寄存器。

寄存器	偏移量	R/W	描述	复位值
ID2	**FMC_BA+0x40**	**RO**	ID2数据寄存器	**NA**

31	30	29	28	27	26	25	24
ID2							
23	22	21	20	19	18	17	16
ID2							
15	14	13	12	11	10	9	8
ID2							
7	6	5	4	3	2	1	0
ID2							

Bits	描述	
[31:0]	ID2	保存 NVR 区域地址 6 处的数据

6.5.4.13 ID3 寄存器

该寄存器为 NVR 区域地址 7 上电后读出的芯片 ID 数据寄存器。

寄存器	偏移量	R/W	描述	复位值
ID2	**FMC_BA+0x44**	**RO**	ID3数据寄存器	**NA**

31	30	29	28	27	26	25	24
ID3							
23	22	21	20	19	18	17	16
ID3							
15	14	13	12	11	10	9	8
ID3							
7	6	5	4	3	2	1	0
ID3							

Bits	描述	
[31:0]	ID3	保存 NVR 区域地址 7 处的数据

6.6 IO 配置管理器（IOCONFIG）

6.6.1 概述

MT32F005/006 系列 MCU 最多有 24 个通用 IO 管脚（GPIO），这些管脚和其他数字以及模拟功能模块共享。24 个管脚分为 3 租端口，PIO0-PIO2，每组端口有 8 个管脚，分别为 PIOx._0 - PIOx._7。每个管脚都是独立的，都有相应的寄存器来控制管脚工作模式。

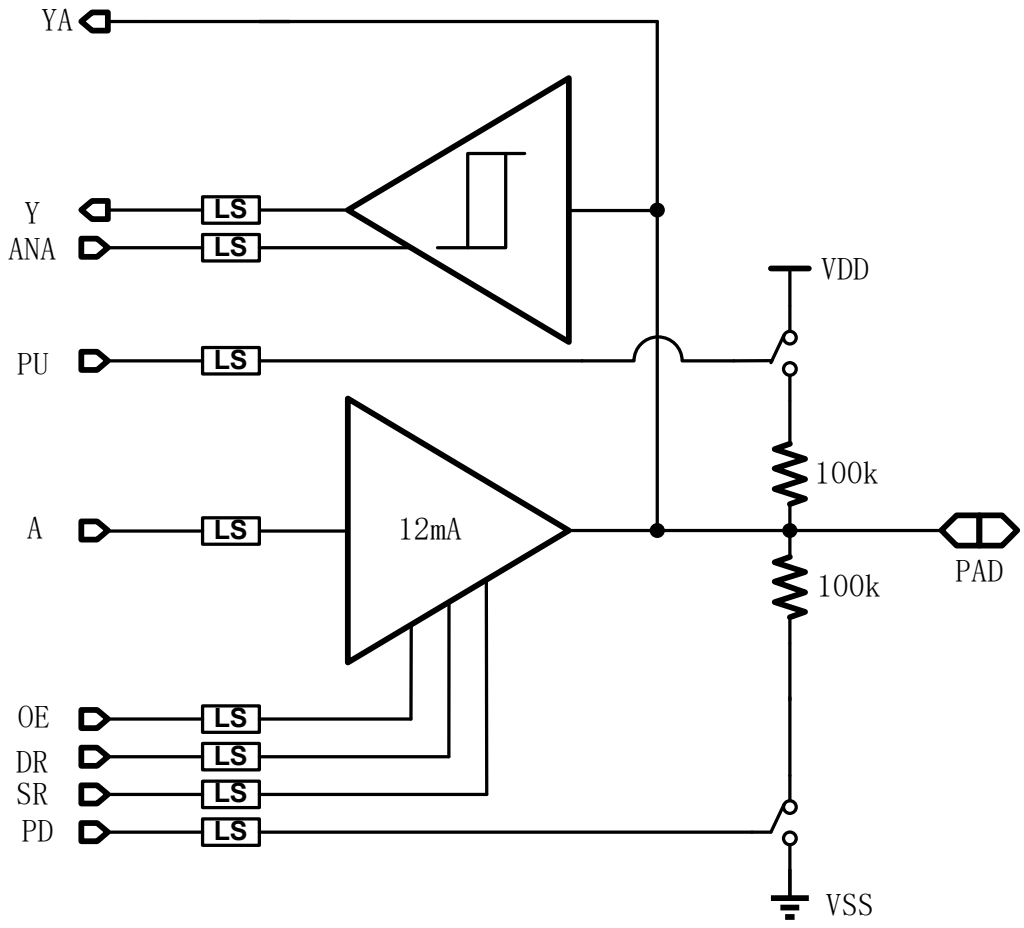
每个管脚的类型都可以由软件独立配置为输入或输出模式，还可以配置管脚的内部上下拉模式。每个管脚配有独立的上拉/下拉电阻，内部上拉/下拉电阻阻值大约为 100KΩ。同时管脚还可以被配置成数模混用模式，允许模拟信号从通用 IO 上通过。

复位后，所有的管脚默认为通用 GPIO 功能，且处于输入悬空状态。但是如下管脚除外：PIO1_2 和 PIO1_3 默认配置为 SWD 调试管脚。（芯片内部针对 SWD 管脚的复用做了硬件保护，芯片上电复位后的 100ms 内无法改变 PIO1_2 和 PIO1_3 两个管脚的功能）

6.6.2 特性

- IO 管脚功能灵活复用，可在模块功能管脚和通用管脚间切换
- IO 管脚可作为模拟管脚使用（ADC，ACMP，OPA，TSC 等）
- I/O 工作模式：
  - ◆ 双向模式
  - ◆ 模拟模式
  - ◆ 输入上拉、下拉、高阻模式
  - ◆ 推挽输出模式
  - ◆ 输出速率可配

6.6.3 IO 管脚结构框图



图：IO 管脚结构框图

6.6.4 IO 管脚功能复用

每个管脚除了作为通用 IO 功能之外还可以配置成数字功能模块管脚。下面是 IO 管脚数字功能复用图：

表：IO 管脚数字功能复用表

GPIO0					7	6	5	4	3	2	1	0
Pad_Name					PIO0_7(a)	PIO0_6(a)	PIO0_5(a)	PIO0_4(a)	PIO0_3(a)	PIO0_2(a)	PIO0_1(a)	PIO0_0
PinFunction0	MUXSEL=	0	0	0	GPIO0_7	GPIO0_6	GPIO0_5	GPIO0_4	GPIO0_3	GPIO0_2	GPIO0_1	GPIO0_0
PinFunction1	MUXSEL=	0	0	1	MCOB0(o)*	MCOA2(o)*	I2C0_SCL*	I2C0_SDA*	CT0_IO2	UART0_TXD(o)	UART0_RXD(i)	CT0_IO0*
PinFunction2	MUXSEL=	0	1	0	MCOB1(o)*	MCOB0(o)*	CT0_IO3*	CT0_IO3	SPI0_SEL*	I2C0_SCL	I2C0_SDA	MCI0(i)*
PinFunction3	MUXSEL=	0	1	1	MCOA0(o)*	MCOA0(o)	MCOA1(o)*	MCOB0(o)	MCOB2(o)	CAN_RX(i)	CAN_TX(o)	
PinFunction4	MUXSEL=	1	0	0	CMP1_0(o)	MCOA1(o)*			CMP0_0(o)	MCOA2(o)	MCOB1(o)	
PinFunction5	MUXSEL=	1	0	1	SPI0_MOSI	SPI0_MISO	SPI0_SEL	SPI0_SCK		CT0_IO1	CT0_IO0	
PinFunction6	MUXSEL=	1	1	0			UART0_TXD(o)*	UART0_RXD(i)*		SPI0_SEL*	SPI0_SCK*	
PinFunction7	MUXSEL=	1	1	1	OUTCLK(o)	MCI2(i)*	MCI1(i)*	MCI0(i)*	MCI2(i)*	MCI1(i)*	MCI0(i)*	MCIABORT(i)*

GPIO1					7	6	5	4	3	2	1	0
Pad_Name					PIO1_7(a)	PIO1_6(a)	PIO1_5(a)	PIO1_4(a)	PIO1_3	PIO1_2	PIO1_1(a)	PIO1_0(a)
PinFunction0	MUXSEL=	0	0	0	GPIO1_7	GPIO1_6	GPIO1_5	GPIO1_4	GPIO1_3	GPIO1_2	GPIO1_1	GPIO1_0
PinFunction1	MUXSEL=	0	0	1	CT0_IO2*	CT0_IO1*	CT0_IO0*	CT0_IO3*	CT0_IO2*	SPI0_MISO*	SPI0_MOSI*	SPI0_SCK*
PinFunction2	MUXSEL=	0	1	0	MCOB0(o)*					MCOA1(o)*	MCOA0(o)*	MCOA0(o)*
PinFunction3	MUXSEL=	0	1	1	MCOB2(o)*	MCOA1(o)*	MCOB1(o)*	MCOB2(o)*	CMP0_0(o)*	MCOA2(o)*	MCOB1(o)*	MCOA1(o)
PinFunction4	MUXSEL=	1	0	0	SPI0_MISO*	MCOA0(o)*	MCI1(i)*	CMP1_0(o)*	CAN_RX(i)*	CAN_TX(o)*	MCOB0(o)*	I2C0_SDA*
PinFunction5	MUXSEL=	1	0	1	UART0_TXD(o)*	SPI0_SCK*	SPI0_MOSI*	SPI0_MISO*	SPI0_SEL*	CT0_IO0*	I2C0_SCL*	
PinFunction6	MUXSEL=	1	1	0			CAN_RX(i)*	CAN_TX(o)*	UART0_TXD(o)*	UART0_RXD(i)*	CT0_IO0*	MCI0(i)
PinFunction7	MUXSEL=	1	1	1	I2C0_SDA*		MCIABORT(i)	MCI2(i)	SWD_TCK(i)	SWDIO	MCI1(i)	RESETN(o)

GPIO2					7	6	5	4	3	2	1	0
Pad_Name					PIO2_7	PIO2_6	PIO2_5	PIO2_4	PIO2_3	PIO2_2	PIO2_1	PIO2_0(a)
PinFunction0	MUXSEL=	0	0	0	GPIO2_7	GPIO2_6	GPIO2_5	GPIO2_4	GPIO2_3	GPIO2_2	GPIO2_1	GPIO2_0
PinFunction1	MUXSEL=	0	0	1	CAN_RX(i)*	CAN_TX(o)*	CT0_IO0*	CT0_IO3*	CT0_IO2*	CT0_IO1*	CT0_IO0*	CT0_IO3*
PinFunction2	MUXSEL=	0	1	0			UART0_TXD(o)*	UART0_RXD(i)*				MCOA1(o)*
PinFunction3	MUXSEL=	0	1	1			MCOA0(o)*	MCIABORT(i)*	MCI0(i)*	MCI2(i)*	MCOB1(o)*	MCOA2(o)*
PinFunction4	MUXSEL=	1	0	0			MCOA1(o)*	SPI0_MISO*	SPI0_MOSI*	SPI0_SCK*	MCOB2(o)*	SPI0_SEL*
PinFunction5	MUXSEL=	1	0	1				MCOB2(o)*	MCOA2(o)*		MCI1(i)*	UART0_RXD(i)*
PinFunction6	MUXSEL=	1	1	0					MCOB2(o)*			
PinFunction7	MUXSEL=	1	1	1								I2C0_SCL*

14个ADC通道：PIO0_1-PIO0_7 PIO1_0-PIO1_1 PIO1_4-PIO1_7 PIO2_0

从图中可以看到每个 IO 最多可以被分配 8 种功能，由 MUXSEL[2:0]来决定，其中 000 表示 GPIO 功能，其他值表示其他功能模块管脚。同时名称后面有(a)的管脚还可以被配置成数模混用 IO，可以从管脚上输入或输出模拟量。

- **单个 IO 功能复用**

以 PIO0_1 为例，它一共可以有 8 种数字 IO 功能，同时还可以被配置成模拟 IO 功能。当被配置成 GPIO 时，IO 的输入输出方向，输入输出值都由 GPIO 寄存器来配置，当被配置为功能模块管脚时，IO 的输入输出方向和输入输出值由对应的功能模块自动设置。

当 MUXSEL=000 时，作为 GPIO 使用，此时 IO 输入输出方向，输入输出值等信息由 GPIO 寄存器来控制。

当 MUXSEL=001 时，作为 UART0 模块的 RXD 管脚使用，I O 自动被 UART1 模块配置为输入状态。

当 MUXSEL=010 时，作为 I2C0 模块的 SDA 信号管脚使用，IO 自动被 I2C0 模块配置为双向状态。

当 MUXSEL=011 时，无模块使用此管脚，要避免这种配置。

当 MUXSEL=100 时，作为 MCPWM 模块的 B1 管脚使用，IO 自动被 MCPWM 模块配置为输出状态。

当 MUXSEL=101 时，作为 TIMER0 模块的 IO0 管脚使用，IO 自动被 TIMER0 模块配置为双向状态。

当 MUXSEL=110 时，作为 SPI 模块的 SCK 管脚使用，IO 自动被 SPI 模块配置为双向状态。

当 MUXSEL=111 时，作为 MCPWM 模块的 I0 管脚使用，IO 自动被 MCPWM 模块配置为输入状态。

- **同一功能模块多个功能管脚分布**

为了灵活使用各个 IO，对于同一功能模块的功能管脚会出现在多个 IO 上。这些 IO 在使用时要特别小心，在配置时不要让同一功能管脚出现在一个以上的

IO 上，以免导致信号冲突。

以 SPI0 模块为例，SPI0 模块共有 SPI0_SCK，SPI0_SEL，SPI0_MOSI 和 SPI0_MISO 四个功能管脚。可以看到每个管脚都出现在多个 IO 上，以 SPI0_SCK 为例，共出现在 PIO0_1，PIO0_4，PIO1_0，PIO1_6 和 PIO2_2 上。在使用时可以选择任意一个 IO 作为 SPI0_SCK 功能管脚，但是要注意不能同时有超过一个 IO 管脚被配置成 SCK。

PIO1_2 和 PIO1_3 两个管脚默认被配置为 SWD 功能，用户程序中如果要更改这两个 IO 的功能需要特别小心，有可能会造成芯片无法与调试器连接的情况出现。

6.6.5 IO 管脚配置寄存器映射

R：只读，W：只写，R/W：可读写

寄存器	偏移量	R/W	描述	复位后的值
IOCON 的基地址： **IOCON_BA = 0x4000_1000**				
PIO0_0	IOCON_BA+0x00	R/W	PIO0_0配置寄存器	0x0000_0080
PIO0_1	IOCON_BA+0x04	R/W	PIO0_1配置寄存器	0x0000_0080
PIO0_2	IOCON_BA+0x08	R/W	PIO0_2配置寄存器	0x0000_0080
PIO0_3	IOCON_BA+0x0C	R/W	PIO0_3配置寄存器	0x0000_0080
PIO0_4	IOCON_BA+0x10	R/W	PIO0_4配置寄存器	0x0000_0080
PIO0_5	IOCON_BA+0x14	R/W	PIO0_5配置寄存器	0x0000_0080
PIO0_6	IOCON_BA+0x18	R/W	PIO0_6配置寄存器	0x0000_0080
PIO0_7	IOCON_BA+0x1C	R/W	PIO0_7配置寄存器	0x0000_0080
PIO1_0	IOCON_BA+0x20	R/W	PIO1_0配置寄存器	0x0000_0080
PIO1_1	IOCON_BA+0x24	R/W	PIO1_1配置寄存器	0x0000_0080
PIO1_2	IOCON_BA+0x28	R/W	PIO1_2配置寄存器	0x0000_0087
PIO1_3	IOCON_BA+0x2C	R/W	PIO1_3配置寄存器	0x0000_0087

PIO1_4	IOCON_BA+0x30	R/W	PI01_4配置寄存器	0x0000_0080
PIO1_5	IOCON_BA+0x34	R/W	PI01_5配置寄存器	0x0000_0080
PIO1_6	IOCON_BA+0x38	R/W	PI01_6配置寄存器	0x0000_0080
PIO1_7	IOCON_BA+0x3C	R/W	PI01_7配置寄存器	0x0000_0080
PIO2_0	IOCON_BA+0x40	R/W	PI02_0配置寄存器	0x0000_0080
PIO2_1	IOCON_BA+0x44	R/W	PI02_1配置寄存器	0x0000_0080
PIO2_2	IOCON_BA+0x48	R/W	PI02_2配置寄存器	0x0000_0080
PIO2_3	IOCON_BA+0x4C	R/W	PI02_3配置寄存器	0x0000_0080
PIO2_4	IOCON_BA+0x50	R/W	PI02_4配置寄存器	0x0000_0080
PIO2_5	IOCON_BA+0x54	R/W	PI02_5配置寄存器	0x0000_0080
PIO2_6	IOCON_BA+0x58	R/W	PI02_6配置寄存器	0x0000_0080
PIO2_7	IOCON_BA+0x5C	R/W	PI02_7配置寄存器	0x0000_0080

6.6.5.1 PIO0_0 寄存器

该寄存器用来配置 PIO0_0 的功能模式。

寄存器	偏移量	R/W	描述	复位值
PIO0_0	**IOCON_BA+0x00**	**R/W**	PI00_0配置寄存器	**0x0000_0080**

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

Bits	描述	
[31:14]	保留	保留
[13]	OS	1：使能 IO 的开源输出功能 0：不使能
[12]	OD	1：使能 IO 的开漏输出功能 0：不使能
[11:10]	保留	保留
[9]	SR	设置 IO 的输出速率 0：fast 1：slow
[8]	保留	保留
[7]	IE	设置 IO 的输入功能是否使能 1：使能 IO 的输入功能 0：禁止 IO 的输入功能
[6]	保留	保留
[5]	ANA	设置数模混用模式 1：PIO 作为模拟功能管脚使用，可直接外接模拟信号，数字功能被禁止 0：PIO 作为正常数字功能管脚使用
[4:3]	MODE	设置 IO 内部工作模式 00：IO 内部上下拉电阻都不使能 01：IO 下拉电阻使能 10：IO 上拉电阻使能 11：保留
[2:0]	FUNC	设置 IO 的复用功能 000：GPIO 功能 001：TIMER 模块通道 0 功能

		010：MCPWM 通道 0 输入功能 011~110：保留 111：MCPWM 模块的 abort 输入功能

由于 IO 管脚较多，这里不一一描述每个 IO 配置寄存器的详细信息。每个 IO 的上电默认状态可以从对应的 IO 寄存器的复位值中查到。每个 IO 对应的管脚功能可以从 IO 管脚复用功能图中查询。

6.7 GPIO 控制器（GPIO）

6.7.1 概述

MT32F005/006 系列 MCU 最多有 24 个通用 IO 管脚,每个管脚都可以被配置成 GPIO 功能。当 IO 被配置成 GPIO 功能时，IO 的输入输出状态，输入输出值等信息由 GPIO 模块来控制。

GPIO 模块将 3 组 IO 分为一个 32 位的 BANK 来控制， BANK0 控制 PIO0-PIO2 这三组共 24 个 PIO，相关寄存器的高 8 位忽略。

可以通过 CPU 的一次写操作改变任意 IO 的输出电平而不影响其他 IO 的输出电平值。

6.7.2 GPIO 寄存器映射

R：只读，W：只写，R/W：可读写

寄存器	偏移量	R/W	描述	复位后的值
GPIO 的基地址： **BANK0_BA = 0x4002_0000**				
FILTER_CTRL	BANK0_BA+0x0100	R/W	GPIO滤波控制寄存器	0x0000_0000
BSRR0_LOW	BANK0_BA+0x4000	R/W	位设置/清除寄存器	0x0000_0000
BSRR0_HIGH	BANK0_BA+0x4004	R/W	位设置/清除寄存器	0x0000_0000
DATA0	BANK0_BA+0x0000	R/W	BANK0数据寄存器	NA
DIR0	BANK0_BA+0x8000	R/W	BANK0数据方向寄存器	0x0000_0000
IS0	BANK0_BA+0x8010	R/W	BANK0中断触发寄存器	0x0000_0000
IBE0	BANK0_BA+0x8020	R/W	BANK0中断双边沿寄存器	0x0000_0000
IEV0	BANK0_BA+0x8030	R/W	BANK0中断事件寄存器	0x0000_0000
IE0	BANK0_BA+0x8040	R/W	BANK0中断屏蔽寄存器	0x0000_0000
IRS0	BANK0_BA+0x8050	R	BANK0中断原始状态寄存器	0x0000_0000
MIS0	BANK0_BA+0x8060	R	BANK0中断屏蔽状态寄存器	0x0000_0000

MIC0	BANK0_BA+0x8070	W	BANK0中断清除寄存器	0x0000_0000
DATAMASK0	BANK0_BA+0x8080	R/W	BANK0数据屏蔽寄存器	0x0000_0000

GPIO 所有的主寄存器都有三个 set/clear/toggle 影子寄存器，往 set 寄存器中任意位写 1 会将主寄存器的对应位置 1，往 clear 寄存器的任意位置 1 会将主寄存器的对应位清 0，往 toggle 寄存器的任意位写 1 会将主寄存器的对应位翻转。这三个影子寄存器的地址分别为主寄存器地址加上 4/8/C。

6.7.2.1 FILTER_CTRL 寄存器

滤波寄存器过滤外部 GPIO 管脚电平的脉冲宽度。

寄存器	偏移量	R/W	描述	复位值
FILTER_CTRL	BANK0_BA+**0x0100**	**R/W**	**控制寄存器**	**0x0**

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

Bits	描述	
[31:25]	保留	保留
[24]	FILTER_EN	**滤波使能位** 1：使能管脚滤波； 0：禁能管脚滤波。
[23:8]	保留	保留

[7:0]	FILTER_NUM	**用于设置滤波的计数值** 计数单位为 1 个总线时钟周期,滤波计数时间为 FILTER_NUM+1。 如果滤波功能使能，只有当管脚上的电平宽度超过设定的 CLK 周期个数时，该电平才能进入 GPIO 模块内部。

6.7.2.2 BSRR0_LOW 寄存器

Bank0 位设置/清除寄存器，该寄存器控制 bank0 的低 16bit。

寄存器	偏移量	R/W	描述	复位值
BSRR0_LOW	**BANK0_BA+0x4000**	**R/W**	**位设置寄存器**	**0x0**

31	30	29	28	27	26	25	24
BSRR0_LOW_WREN[15:8]							
23	22	21	20	19	18	17	16
BSRR0_LOW_WREN[7:0]							
15	14	13	12	11	10	9	8
BSRR0_LOW_DATA[15:8]							
7	6	5	4	3	2	1	0
BSRR0_LOW_DATA[7:0]							

Bits	描述	
[31:16]	BSRR0_LOW_WREN	**用于设置 bank0 的低 16bit 的值使能位** 当 BSRR0_LOW_WREN 某些位为 1 时，BSRR0_LOW_DATA 对应位写入到 Bank0 的低 16bit 中，BSRR0_LOW_WREN 为 0 的位保持原来的值不变。
[15:0]	BSRR0_LOW_DATA	**用于设置 bank0 的低 16bit 的值**

		当 BSRR0_LOW_WREN 某些位为 1 时，BSRR0_LOW_DATA 对应位写入到 Bank0 的低 16bit 中，BSRR0_LOW_WREN 为 0 的位保持原来的值不变。

6.7.2.3 BSRR0_HIGH 寄存器

Bank0 位设置/清除寄存器，该寄存器控制 bank0 的高 16bit。

寄存器	偏移量	R/W	描述	复位值
BSRR0_HIGH	**BANK0_BA+0x4004**	**R/W**	**位设置寄存器**	**0x0**

31	30	29	28	27	26	25	24
BSRR0_ HIGH_WREN[15:8]							
23	22	21	20	19	18	17	16
BSRR0_ HIGH_WREN[7:0]							
15	14	13	12	11	10	9	8
BSRR0_ HIGH_DATA[15:8]							
7	6	5	4	3	2	1	0
BSRR0_ HIGH_DATA[7:0]							

Bits	描述	
[31:16]	BSRR0_HIGH_WREN	**用于设置 bank0 的高 16bit 的值使能位** 当 BSRR0_HIGH_WREN 某些位为 1 时，BSRR0_HIGH_DATA 对应位写入到 Bank0 的高 16bit 中，BSRR0_HIGH_WREN 为 0 的位保持原来的值不变。
[15:0]	BSRR0_HIGH_DATA	**用于设置 bank0 的高 16bit 的值** 当 BSRR0_HIGH _WREN 某些位为 1 时，BSRR0_

		HIGH_DATA 对应位写入到 Bank0 的高 16bit 中，BSRR0_HIGH_WREN 为 0 的位保持原来的值不变。

6.7.2.4 DATA0 寄存器

无论管脚是被配置为 GPIO 输入或输出，或是其它模块功能， DATA0 寄存器都保存着对应的 IO 的当前状态（高电平或低电平）。

如果引脚被配置为 GPIO 输出，则 DATA0 寄存器的当前值会被驱动至对应的管脚上。

寄存器	偏移量	R/W	描述	复位值
DATA0	**BANK0_BA+0x0000**	**R/W**	**BANK0数据寄存器**	**NA**

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

Bits	描述	
[31:0]	DATA	**反映对应 PIO 的管脚电平状态** **1：对应的 PIO 上位高电平** **0：对应的 PIO 上位低电平** 读取 DATA0 寄存器时总会返回引脚的当前逻辑电平（状态），而

		不论它的配置是什么。

6.7.2.5 DIR0 寄存器

配置对应 PIO 管脚的输入输出方向

寄存器	偏移量	R/W	描述	复位值
DIR0	**BANK0_BA+0x8000**	**R/W**	BANK0数据方向寄存器	**0x0000_0000**

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

Bits	描述	
[31:0]	DIR	**设置对应 PIO 的输入输出状态** 1：对应的 PIO 被配置为输出 0：对应的 PIO 被配置为输入

6.7.2.6 IS0 寄存器

配置对应 PIO 管脚的中断触发方式

寄存器	偏移量	R/W	描述	复位值

IS0	BANK0_BA+0x8010	R/W	BANK0中断触发寄存器	0x0000_0000

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

Bits	描述	
[31:0]	IS	**设置对应 PIO 的中断触发方式** 1：对应的 PIO 的中断被配置为电平触发 0：对应的 PIO 的中断被配置为边沿触发

6.7.2.7 IBE0 寄存器

配置对应 PIO 管脚的中断是否为双边沿触发

寄存器	偏移量	R/W	描述	复位值
IBE0	BANK0_BA+0x8020	R/W	BANK0中断双边沿寄存器	0x0000_0000

31	30	29	28	27	26	25	24
IBE[31:24]							
23	22	21	20	19	18	17	16
IBE[23:16]							
15	14	13	12	11	10	9	8

IBE[15:8]							
7	6	5	4	3	2	1	0
IBE[7:0]							

Bits	描述	
[31:0]	IBE	**设置对应 PIO 的边沿中断是否为双边沿触发** 1：对应的 PIO 的边沿中断被配置为双边沿触发 0：对应的 PIO 的边沿中断由 IEV0 寄存器控制

6.7.2.8 IEV0 寄存器

配置对应 PIO 管脚的中断触发事件。

寄存器	偏移量	R/W	描述	复位值
IEV0	**BANK0_BA+0x8030**	**R/W**	**BANK0中断事件寄存器**	**0x0000_0000**

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

Bits	描述	
[31:0]	IEV	**设置对应 PIO 的中断触发事件** 1：根据 IS0 寄存器的配置，对应的 PIO 的中断由上升沿或高电平

		触发 0：根据 IS0 寄存器的配置，对应的 PIO 的中断由下降沿或低电平触发

6.7.2.9 IE0 寄存器

设置是否允许对应的 PIO 产生各自的中断以及产生组合而成的 BANKx 中断。

寄存器	偏移量	R/W	描述	复位值
IE0	**BANK0_BA+0x8040**	**R/W**	**BANK0中断屏蔽寄存器**	**0x0000_0000**

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

Bits	描述	
[31:0]	IE	**设置对应 PIO 的中断是否要被屏蔽** 1：对应 PIO 管脚的中断不被屏蔽 0：对应 PIO 管脚的中断被屏蔽 只要 BANK0 对应的管脚中有一个产生了 PIO 中断，那么 BANK0 中断就会产生

6.7.2.10 IRS0 寄存器

反映对应 PIO 管脚上的原始中断状态（未经屏蔽前），如果为高电平则表明在触发 IE0 寄存器前的所有要求都已满足。

寄存器	偏移量	R/W	描述	复位值
IRS0	**BANK0_BA+0x8050**	**R**	BANK0中断原始状态寄存器	**0x0000_0000**

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

Bits	描述	
[31:0]	IRS	**反映对应 PIO 上的中断原始状态** 1：对应 PIO 管脚上有原始中断 0：对应 PIO 管脚上无中断

6.7.2.11 MIS0 寄存器

反映对应 PIO 管脚上的中断状态（经屏蔽后）。

寄存器	偏移量	R/W	描述	复位值
MIS0	**BANK0_BA+0x8060**	**R**	BANK0中断屏蔽状态寄存器	**0x0000_0000**

31	30	29	28	27	26	25	24

MIS[31:24]							
23	22	21	20	19	18	17	16
MIS[23:16]							
15	14	13	12	11	10	9	8
MIS[15:8]							
7	6	5	4	3	2	1	0
MIS[7:0]							

Bits	描述	
[31:0]	MIS	**反映对应 PIO 上的中断状态** 1：对应 PIO 管脚上有中断 0：对应 PIO 管脚上无中断或被屏蔽

6.7.2.12 MIC0 寄存器

该寄存器允许用软件清除在中断触发寄存器中定义为边沿触发的 PIO 上的边沿检测。该寄存器对被定义为电平触发的 PIO 无影响。

寄存器	偏移量	R/W	描述	复位值
MIC0	**BANK0_BA+0x8070**	**W**	**BANK0中断清除寄存器**	**0x0000_0000**

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

MIC[7:0]

Bits	描述	
[31:0]	MIC	**设置清除对应 PIO 管脚上的边沿检测逻辑** 1：清除对应 PIO 管脚上的边沿检测逻辑 0：无影响

6.7.2.13 DATAMASK0 寄存器

该寄存器允许在对数据寄存器进行读写操作时只对某些位起作用，而将其他位屏蔽。

写操作时，被屏蔽的位的值不发生改变，读操作时被屏蔽的位的读出值始终为 0。

寄存器	偏移量	R/W	描述	复位值
DATAMASK0	**BANK0_BA+0x8080**	**R/W**	BANK0数据屏蔽寄存器	**0x0000_0000**

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

Bits	描述	
[31:0]	MIC	**设置对 DATA 寄存器操作时要屏蔽的位**

		1：对对应 DATA 寄存器中的位的读写操作被屏蔽 0：不屏蔽

6.8 I2C 总线控制器(I2C)

6.8.1 概述

I2C 总线是两线制的串行总线，包含一个串行数据线 SDA 和一个串行时钟线 SCL。每一个挂在 I2C 总线上的设备都有唯一指定的地址，发起数据传输时，它们分为 master 和 slave 两种，Master 用于初始化一个数据传输，并向总线上提供时钟信号，而此时被 master 寻址的设备就称为 slave。

I2C 模块可以工作在 2 种模式，标准模式（最高支持 100Kb/s），快速模式（最高支持 400Kb/s）。

6.8.2 特性

- 标准 I2C 的总线接口；
- 两种速度模式：
  - ◆标准模式（最高支持 100Kb/s）
  - ◆快速模式（最高支持 400Kb/s）
- 可编程时钟能够实现 I2C 传输速率控制；
- 7- or 10-bit 地址模式；
- 主、从机之间双向数据传输；
- 串行时钟同步允许不同位速率的器件通过同一个串行总线通信；
- 串行时钟同步可作为一个握手机制来挂起和恢复串行传输；

6.8.3 基本配置

使用以下系统寄存器来打开 I2C 模块：

时钟：I2C 模块的工作时钟默认是关闭的，使用前必须在 AHBCLKCTRL（bit4）寄存器中打开对应的控制位。

复位：I2C 模块的复位默认是高，即处于正常工作状态，如果要对 I2C 模块复位，应先将 PRESET_CTRL（bit4）的对应位置为 0，然后再置为 1。

6.8.4 寄存器映射

R: 只读, W: 只写, R/W: 可读写

寄存器	偏移量	R/W	描述	复位后的值
I2C 的基地址： I2C_BA = 0x4000_4000				
ADDR	I2C_BA+0x00	R/W	Slave 地址寄存器	0x00000000
XADDR	I2C_BA+0x10	R/W	扩展 Slave 地址寄存器	0x00000000
DATA	I2C_BA+0x04	R/W	数据寄存器	0x00000000
CNTR	I2C_BA+0x08	R/W	控制寄存器	0x00000000
STAT	I2C_BA+0x0C	R	状态寄存器	0x00000000
CCR	I2C_BA+0x0C	W	时钟控制寄存器	0x00000000

● 6.8.4.1 ADDR 寄存器

Bits	描述	
[31:8]	保留	保留
[7:3]	SLA[7:2]/5’h11110	7bit 地址：高 5bit slave 地址 10bit 地址：固定写入 5’h11110
[2:1]	SLA[1:0]/SLAX[9:8]	7bit 地址：低 2bit slave 地址 10bit 地址：高 2bit slave 地址
[0]	GCE	广播地址使能

NOTE：当 I2C 控制器工作在 MASTER 模式时，该寄存器不需要配置。

6.8.4.2 XADDR 寄存器

Bits	描述	
[31:8]	保留	保留
[7:0]	SLAX[7:0]	低 8bit 扩展 slave 地址

NOTE：当 I2C 控制器工作在 MASTER 模式时，该寄存器不需要配置。

6.8.4.3 DATA 寄存器

Bits	描述	
[31:8]	保留	保留
[7:0]	DATA	该寄存器包含接收或者待发送的数据，读该寄存器会得到从 I2C 接口读入的数据。 在发送时，MSB 先发送；在接收时，第一个 bit 会存到寄存器的 MSB 位置

6.8.4.4 CNTR 寄存器

Bits	描述	
[31:8]	保留	保留
[7]	IEN	置 1 时中断使能
[6]	ENAB	置 1 时 BUS 使能
[5]	STA	置 1 时发送 START，当 START 发送完成后，改 bit 位自动清零。
[4]	STP	置 1 时发送 STOP，当 STOP 发送完成后，改 bit 位自动清零。
[3]	IFLG	中断状态标志

[2]	AAK	置 1 时发送 ACK，当 ACK 发送完成后，改 bit 位自动清零。
[1:0]	Reserved	保留

- 6.8.4.5 STAT 寄存器(read only)

Bits	描述	
[31:8]	保留	保留
[7:0]	STAT_NUM	模块状态代码，状态码对应状态请见下表

表： 状态码对应状态

Code	Status
00h	Bus error
08h	START condition transmitted
10h	Repeated START condition transmitted
18h	Address + Write bit transmitted, ACK received
20h	Address + Write bit transmitted, ACK not received
28h	Data byte transmitted in master mode, ACK received
30h	Data byte transmitted in master mode, ACK not received
38h	Arbitration lost in address or data byte
40h	Address + Read bit transmitted, ACK received
48h	Address + Read bit transmitted, ACK not received
50h	Data byte received in master mode, ACK transmitted
58h	Data byte received in master mode, not ACK transmitted
60h	Slave address + Write bit received, ACK transmitted
68h	Arbitration lost in address as master, slave address + Write bit received, ACK transmitted
70h	General Call address received, ACK transmitted
78h	Arbitration lost in address as master, General Call address received, ACK transmitted
80h	Data byte received after slave address received, ACK transmitted
88h	Data byte received after slave address received, not ACK transmitted
90h	Data byte received after General Call received, ACK transmitted
98h	Data byte received after General Call received, not ACK transmitted
A0h	STOP or repeated START condition received in slave mode
A8h	Slave address + Read bit received, ACK transmitted
B0h	Arbitration lost in address as master, slave address + Read bit received, ACK

	transmitted
B8h	Data byte transmitted in slave mode, ACK received
C0h	Data byte transmitted in slave mode, ACK not received
C8h	Last byte transmitted in slave mode, ACK received
D0h	Second Address byte + Write bit transmitted, ACK received
D8h	Second Address byte + Write bit transmitted, ACK not received

- 6.8.4.6 CCR 寄存器(write only)

Bits	描述	
[31:8]	保留	保留
[6:3]	M	时钟分配因子 M
[2:0]	N	时钟分配因子 N

两个时钟因子可以允许 Master 模式下，输出的时钟 OSCL 的频率和内部采样的时钟频率相互独立。

内部采样频率计算公式：$F_{SAMP} = F_0 = F_{CLK} / 2^N$。

输出时钟计算公式：$F_{OSCL} = F_1 / 10 = F_{CLK} / (2^N.(M + 1).10)$。

6.9 串行外设接口控制器(SPI)

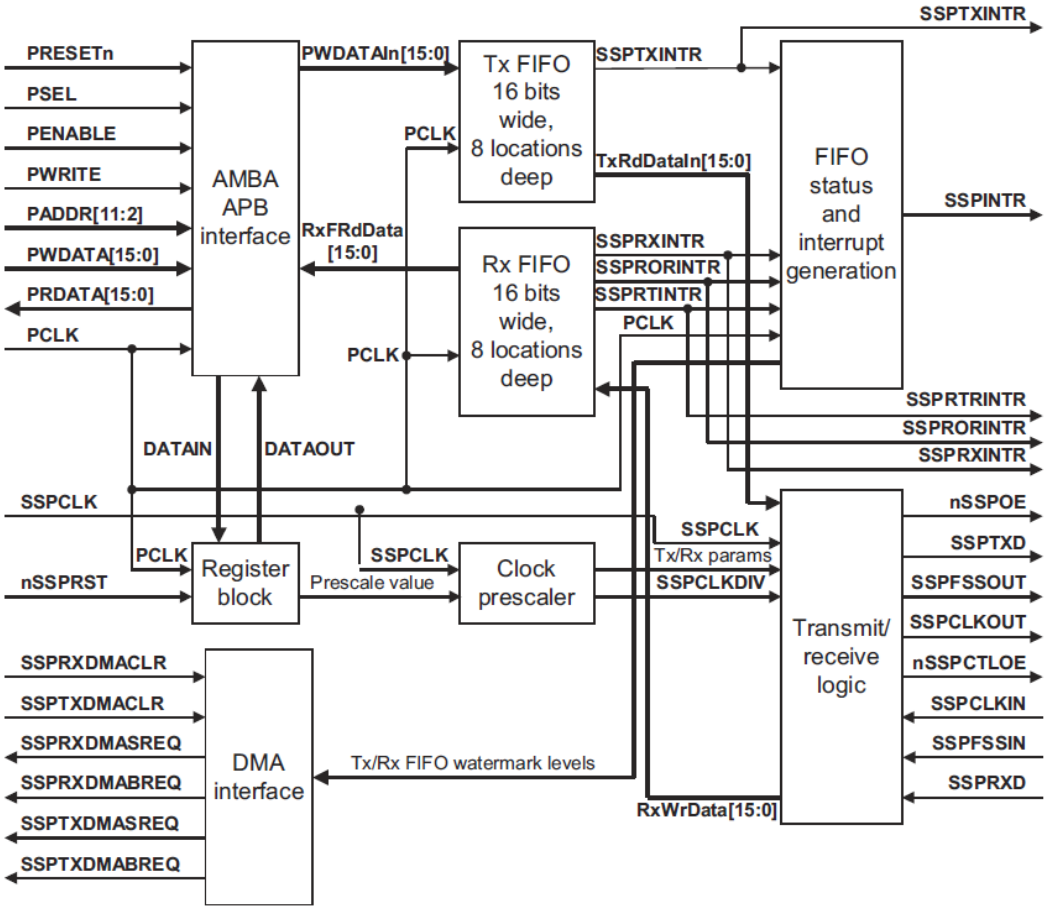
6.9.1 概述

串行外设接口(SPI)是一个工作于全双工模式下的同步串行数据通讯协议接口。MT32F005/006 系列包括 1 组 SPI 控制器，可工作于主机或从机模式，将从外设接收到的数据进行串并转换，或将要发送到外设的数据进行并串转换。

6.9.2 特性

- 支持主机或从机模式
- 支持 SPI 数据格式
- 支持全双工数据传输模式（slave 模式下仅支持半双工模式）
- 传输比特长度可配置位 4-16bit
- 提供 FIFO 缓存
- 数据传输时钟频率可配置，最大可为系统时钟的 1/2

6.9.3 SPI 控制器框图



图： SPI 模块框图

6.9.4 基本配置

有多个 IO 可以配置成 SPI 管脚，具体根据不同的封装形式有所不同，参见 6.6.4 节 IO 管脚功能复用一节，然后在 IOCONFIG 模块中将相应的管脚功能配置成 SPI 功能。

SPI 模块有两个时钟，分别为 PCLK 和 SPICLK，SPIO 模块的 PCLK 通过系统控制模块中的 AHBCLKCTRL 寄存器的 bit[9]来使能，SPICLK 由 SPIOCLKDIV 来设置，详见 6.2 系统管理器一章。其中 SPICLK 频率必须小于等于 PCLK 的频率。

6.9.5 SPI 功能描述

6.9.5.1 SPI 接口信号

SPI 接口是一个四线全双工串行协议接口，四个接口信号分别为：SPISCK（串行时钟信号），SPISSN（外设片选信号），MOSI（主机数据输出信号）和 MISO（主机数据输入信号）。

串行时钟信号的相位和极性共有四种可能的组合，由 SPH 和 SPO 寄存器位来控制。时钟相位（SPH）决定了串行传输是由外设片选信号的下降沿发起还是由串行时钟信号的第一个时钟沿发起。时钟极性（SPO）决定了串行时钟的无效状态时是高电平还是低电平。

外设片选信号低电平有效，因此在 SPI 控制器空闲或禁止时 SPISSN 信号为高电平。

SPICLK 的频率必须小于等于 PCLK。Master 模式下，SPISCK 的频率最大为 SPICLK 的一半。Slave 模式下 SPICLK 必须至少为 SPISCK 信号频率的 12 倍。

6.9.5.2 发送和接收 FIFO 缓冲

接收端和发送端都各有一个 16 位宽，深度为 8 的 FIFO。当数据帧长度设置为小于 16 位时，每个帧也占据 FIFO 的一个 entry，并且在 FIFO 中右对齐，高位自动补 0。

对于发送 FIFO，当系统往 SPI 控制器的数据寄存器(DR)中写数据时就触发一次对 FIFO 的写入行为，SPI 控制器的移位逻辑为从 FIFO 中取出数据到发送移位寄存器中。

对于接收 FIFO，让移位逻辑从移位寄存器中将一帧数据写入 FIFO 时触发一次写操作，系统对 SPI 控制器的数据寄存器(DR)的读操作触发一次 FIFO 的读操作。

6.9.5.3 传输模式

SPI 控制器可工作于下面三种模式，由 CR1 寄存器的 RXO 和 TXO 位控制。

- 全双工模式

当 TXO 和 RXO 都为 0 时，工作于全双工状态。往 TXFIFO 中写入数据后，会产生相应的 SPISCK 信号，SPI 的数据发送管脚会开始发送 TXFIFO 中的数据，同时 SPI 的数据接收管脚上的数据会被保存到 RXFIFO 中去。当 TXFIFO 为空时，SPI 模块停止发送以及接收信号。

- 只发送模式(TXO=1)

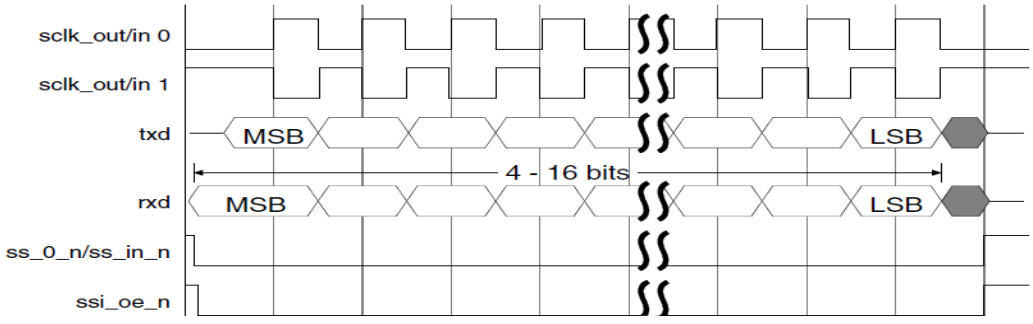
往 TXFIFO 中写入数据后会产生相应的 SPISCK 信号，SPI 的数据发送管脚会开始发送 TXFIFO 中的数据，此时 RX 信号线上的数据被忽略，不会被写到接收 FIFO 中去。当 TXFIFO 为空时，SPI 模块停止发送以及接收信号。

- 只接收模式(RXO=1)

此时无需往 TXFIFO 中写入数据，SPI 模块会在 RXO=1 时自动产生 SPISCK 信号，同时 TX 管脚上不会有信号输出，而 RX 上的数据会被接收到接收 FIFO 中去。当收到预期的数据后需要将 RXO 位置 0 以停止继续产生 SPISCK 信号。

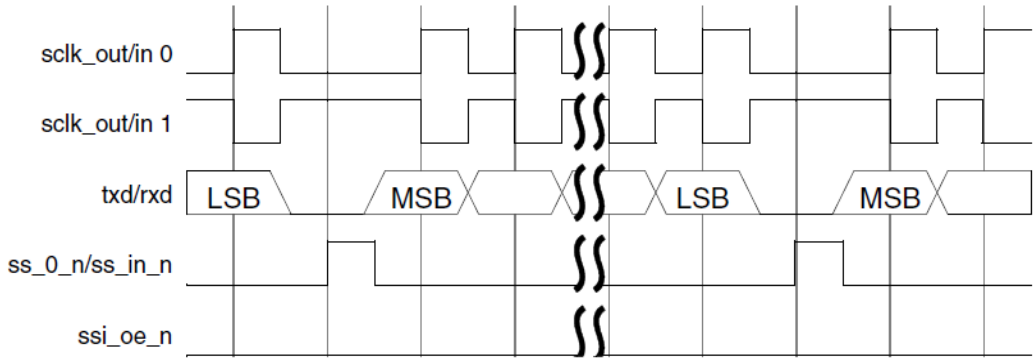
6.9.5.4 SPI 接口时序

对于 SPI 接口，时钟极性(SPO)决定了串行时钟无效状态时处于高电平还是低电平。要传输数据，两个 SPI 设备都必须有唯一的时钟相位(SPH)和时钟极性(SPO)值。数据帧长度可以从 4-16 位。



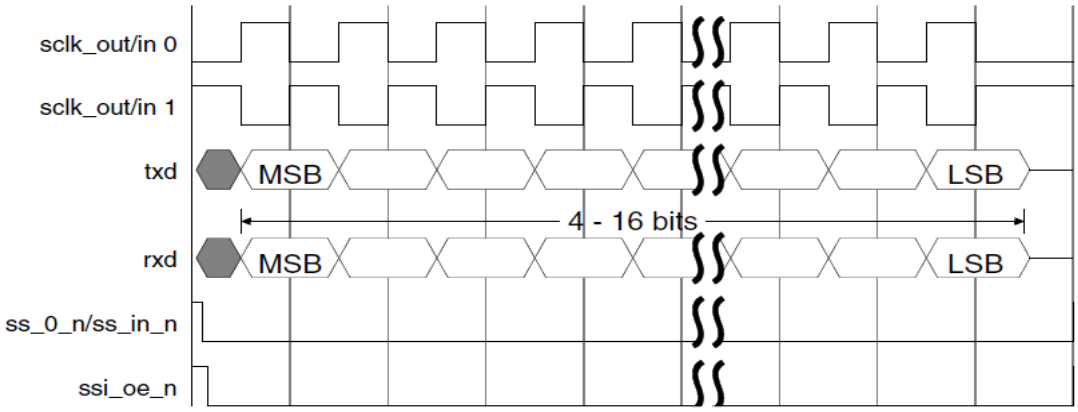
图：SPI 帧格式(SCPH=0, SCPOL=0 & 1)

由于 SCPH=0 时数据传输由 SPISSN 片选信号的下降沿开始，连续的数据帧传输需要 SPISSN 信号在数据帧之前发生翻转。

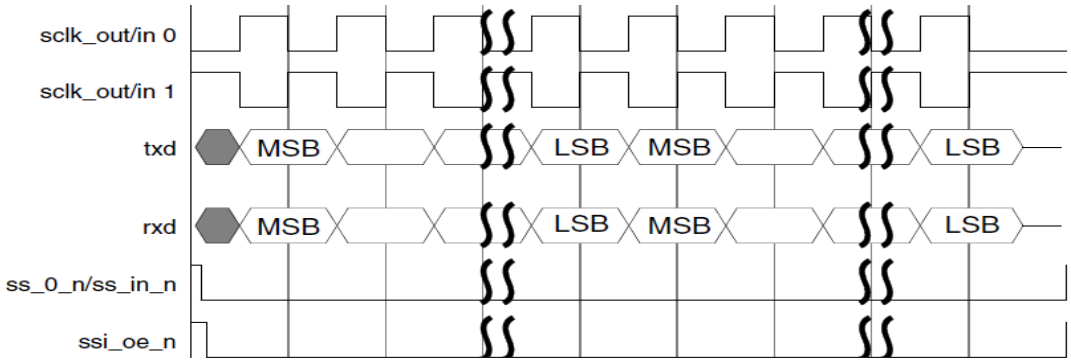


图：连续帧传输情况下 SPI 数据帧格式(SCPH=0, SCPOL=0 & 1)

当 SCPH=1 时，主从外设都是从 SPISSN 有效后的第一个串行时钟沿开始发送数据。第一个数据位在随后的时钟沿(下降沿)被捕获。主从设备都在串行时钟的上升沿传播数据。在连续数据帧传输过程中，SPISSN 可以一直保持为低直到最后一帧的最后一位被捕获。



图：SPI 帧格式(SCPH=1, SCPOL=0 & 1)



图：连续帧传输格式下 SPI 数据帧格式(SCPH=1, SCPOL=0 & 1)

6.9.6 寄存器映射

R：只读，W：只写，R/W：可读写

寄存器	偏移量	R/W	描述	复位后的值
SPI 的基地址： **SPI0_BA = 0x4000_8000**				
CR0	SPI_BA+0x00	R/W	SPI控制寄存器0	0x0000_0000
CR1	SPI_BA+0x04	R/W	SPI控制寄存器1	0x0000_0000
DR	SPI_BA+0x08	R/W	SPI数据寄存器	0x0000_0000
SR	SPI_BA+0x0C	R	SPI状态寄存器	0x0000_0043
CPSR	SPI_BA+0x10	R/W	SPI时钟预分频寄存器	0x0000_0000
IMSC	SPI_BA+0x14	R/W	SPI中断屏蔽使能寄存器	0x0000_0000
RIS	SPI_BA+0x18	R	SPI中断原始状态寄存器	0x0000_0008
MIS	SPI_BA+0x1C	R	SPI中断屏蔽状态寄存器	0x0000_0000
ICR	SPI_BA+0x20	W	SPI中断清除寄存器	0x0000_0000

6.9.6.1 CR0 寄存器

SPI 模块的控制寄存器 0，控制数据帧的频率，格式，位宽等信息。

寄存器	偏移量	R/W	描述	复位值

CR0	SPI_BA+0x00	R/W	SPI控制寄存器0	0x0000_0000

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

Bits	描述	
[31:16]	保留	保留
[15:8]	SCR	设置输出时钟波特率 Fsck= Fspiclk/(CPSDVR*(1+SCR)) CPSDVR 是一个 2-256 之间的偶数值，通过 CPSR 寄存器来配置 SCR 是一个 0-255 之间的数值
[7]	SPH	设置 SCK 的相位
[6]	SPO	设置 SCK 的极性
[5:4]	保留	保留
[3:0]	DSS	设置数据帧的位宽 0000：保留 0001：保留 0010：保留 0011: 4-bit 0100: 5-bit …….

		1111: 16-bit

6.9.6.2 CR1 寄存器

SPI 模块控制寄存器 1。

寄存器	偏移量	R/W	描述	复位值
CR1	**SPI_BA+0x04**	**R/W**	**SPI控制寄存器1**	**0x0000_0000**

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

Bits	描述	
[31:7]	保留	保留
[6]	RXO	默认为全双工模式 该位置 1 表示为 rx only 模式
[5]	TXO	默认为全双工模式 该位置 1 表示为 tx only 模式
[4]	LOCKSSN	该位置 1 时强制输出的 SSN 信号为 0
[3]	SOD	该位决定 slave 模式下是处于接受还是发送模式 0：slave 发送模式 1：slave 接收模式

[2]	MS	选择 master 模式还是 slave 模式 0：master 1： slave
[1]	SSE	1：使能 SPI 控制器的操作 0：禁止 SPI 控制器的操作
[0]	LBM	Loopback 模式 0：正常模式 1：loopback 模式

6.9.6.3 DR 寄存器

SPI 模块数据寄存器，往 DR 中写数据等于往 TXFIFO 中写数据，从 DR 中读数据等于从 RXFIFO 中读数据。

寄存器	偏移量	R/W	描述	复位值
DR	**SPI_BA+0x08**	**R/W**	SPI数据寄存器	**0x0000_0000**

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

Bits	描述	
[31:16]	保留	保留
[15:0]	DATA	读操作：读取 rx fifo 中的数据

		写操作：往 tx fifo 中写数据

6.9.6.4 SR 寄存器

SPI 模块状态寄存器，是一个只读寄存器，包含 SPI 控制器状态机以及 FIFO 的状态。

寄存器	偏移量	R/W	描述	复位值
SR	**SPI_BA+0x0C**	**R/W**	SPI状态寄存器	**0x0000_0043**

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

Bits	描述	
[31:7]	保留	保留
[6]	THE	Tx FIFO half empty 1：表示 tx fifo 半空
[5]	RHF	Rx FIFO half full 1：表示 rx fifo 半满
[4]	BSY	0：表示 SPI 控制器 IDLE 1：表示 SPI 控制器正在发送或接收数据或 tx fifo 不为空
[3]	RFF	Rx FIFO Full

		1：表示 rx fifo 满
[2]	RNE	Rx FIFO not empty 1：表示 rx fifo 不为空
[1]	TNF	Tx FIFO not full 1: 表示 tx fifo 不满
[0]	TNE	Tx FIFO empty 1：表示 tx fifo 空

6.9.6.5 CPSR 寄存器

SPI 模块时钟预分频寄存器，用来对 SPICLK 进行预分频。往这个寄存器中写入的值必须为 2-254 之间的偶数，写入值的最低位会自动置 0。

寄存器	偏移量	R/W	描述	复位值
CPSR	**SPI_BA+0x10**	**R/W**	SPI时钟预分频寄存器	**0x0000_0000**

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

Bits	描述	
[31:8]	保留	保留
[7:0]	CPSDVR	时钟预分频系数

6.9.6.6 IMSC 寄存器

IMSC 寄存器用来设置是否屏蔽各种中断，往特定的位写 1 会打开对应中断的屏蔽，使得该中断对 CPU 可见，往特定的位写 0 会屏蔽该中断。

寄存器	偏移量	R/W	描述	复位值
IMSC	**SPI_BA+0x14**	**R/W**	SPI中断屏蔽使能寄存器	**0x0000_0000**

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

Bits	描述	
[31:4]	保留	保留
[3]	TXIM	TX FIFO 中断屏蔽位 0：tx fifo half empty or less 中断被屏蔽 1：tx fifo half empty or less 中断被打开
[2]	RXIM	RX FIFO 中断屏蔽位 0：rx fifo half full or less 中断被屏蔽 1：rx fifo half full or less 中断被打开
[1]	RTIM	RX timeout 中断屏蔽位 0：rx fifo not empty 并且在 timeout 前没有来读 fifo 行为的中断被屏蔽

		1：rx fifo not empty 并且在 timeout 前没有来读 fifo 行为的中断被打开
[0]	RORIM	RX FIFO overrun 中断屏蔽位 0：RxFIFO 已满时继续写 RX FIFO 行为的中断被屏蔽 1：RxFIFO 已满时继续写 RX FIFO 行为的中断被打开

6.9.6.7 RIS 寄存器

RIS 是中断原始状态寄存器，反映了各个中断未被屏蔽前的状态，是只读寄存器，写操作不起作用。

寄存器	偏移量	R/W	描述	复位值
RIS	**SPI_BA+0x18**	**R**	SPI中断原始状态寄存器	**0x0000_0008**

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

Bits	描述	
[31:4]	保留	保留
[3]	TXRIS	反映 TXINTR 的原始状态
[2]	RXRIS	反映 RXINTR 的原始状态
[1]	RTRIS	反映 RTINTR 的原始状态

[0]	RORRIS	反映 RORINTR 的原始状态

6.9.6.8 MIS 寄存器

MIS 是中断屏蔽后状态寄存器，反映了各个中断被屏蔽后的状态，是只读寄存器，写操作不起作用。

寄存器	偏移量	R/W	描述	复位值
MIS	**SPI_BA+0x1C**	**R**	SPI中断屏蔽后状态寄存器	**0x0000_0000**

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

Bits	描述	
[31:4]	保留	保留
[3]	TXMIS	反映 TXINTR 的屏蔽后状态
[2]	RXMIS	反映 RXINTR 的屏蔽后状态
[1]	RTMIS	反映 RTINTR 的屏蔽后状态
[0]	RORMIS	反映 RORINTR 的屏蔽后状态

6.9.6.9 ICR 寄存器

ICR 寄存器用来清除中断，是一个只写寄存器，写 1 时对应的中断被清除。

寄存器	偏移量	R/W	描述	复位值
ICR	**SPI_BA+0x20**	**W**	SPI中断清除寄存器	**0x0000_0000**

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

Bits	描述	
[31:2]	保留	保留
[1]	RTIC	写 1 清除 RTINTR 中断
[0]	RORIC	写 1 清除 RORINTR 中断

6.10 定时器和 PWM 发生器

6.10.1 概述

MT32F005/006中包含1个定时器和PWM发生器模块，模块内含有四路独立的计数通道，可配置为定时器、计数器、PWM发生器功能以及外部输入捕获功能。

计数器/定时器用于对外设时钟（PCLK）周期或IO管脚上的时钟周期进行计数，可在规定的时间处选择产生中断或执行其它操作。

每个计数器/定时器还可配置成捕获输入功能，用来在输入信号跳变时捕捉定时器的瞬时值，也可以选择产生中断。捕获功能下具备硬件自动测量占空比功能，可以精确测量输入信号的频率和占空比。

在PWM模式下，可向四个匹配输出管脚提供单边沿控制的PWM输出，可设置成单脉冲输出以及固定个数的脉冲输出，还可以设置成载波输出模式。

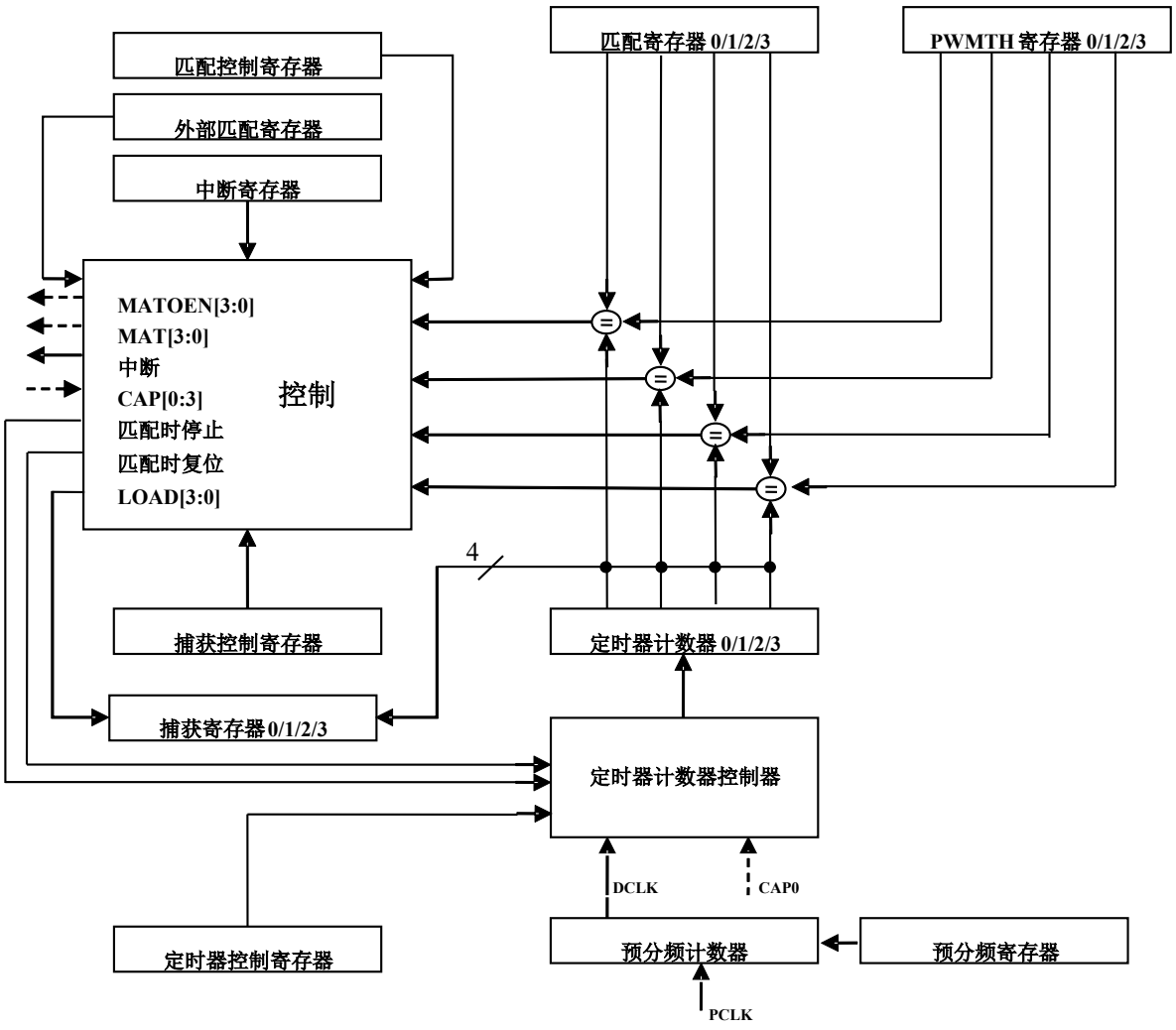
6.10.2 特性

- 四路独立的 16 位定时器/计数器，带有一个可编程的 16 位预分频器。
- 计数器或定时器操作
- 四路 16 位的捕获通道，可在对应 IO 输入信号跳变时捕获定时器的瞬时值。捕获事件可以产生中断。
- 硬件测量功能，可硬件测量输入信号的频率和占空比
- 4 个 16 位匹配寄存器，允许执行以下操作：
  - 匹配时连续工作，在匹配时可选择产生中断
  - 匹配时停止定时器的运行，可选择产生中断
  - 匹配时复位定时器，可选择产生中断
- 有 4 个与匹配寄存器对应的外部输出，这些输出具有以下功能：
  - 匹配时设为低电平
  - 匹配时设为高电平

- 匹配时翻转电平
- 匹配时不执行任何操作

- 对于各定时器，匹配寄存器可配置为 PWM 模式，允许使用多达 4 个匹配输出作为单边沿控制的 PWM 输出。
- PMW 单脉冲输出
- PWM 脉冲计数输出
- 载波功能
- 可硬件触发 ADC 采样

6.10.3 定时器结构框图



图：定时器模块框图

6.10.4 基本配置

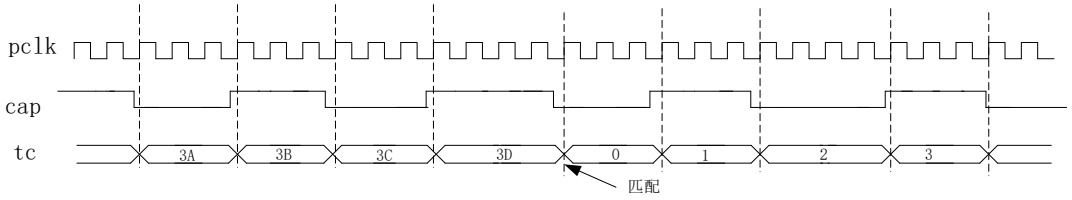
每个定时器模块有 4 个管脚，为 CT0_IO[3:0]，对应于内部的 4 路独立计数器，四个管脚根据需要可被配置成 PWM 输出功能或输入捕获功能。有多个 IO 可以配置成 TIMER 管脚，具体根据不同的封装形式有所不同，参见 6.6.4 节 IO 管脚功能复用图，然后在 IOCONFIG 模块中将相应的管脚功能配置成 TIMER 功能。

TIMER0 的模块时钟为 PCLK，通过 AHBCLKCTRL 寄存器的 bit[10]来使能。

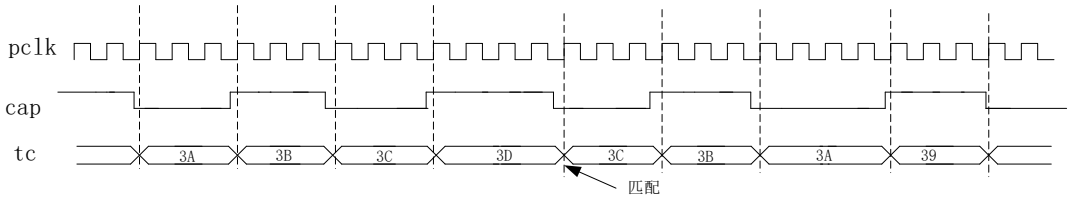
6.10.5 基本操作

6.10.5.1 计数器模式

在该模式下，TC0/1/2/3 的值在 IO0/1/2/3 输入的上升沿/下降沿/两个边沿上递增、递减或先递增后递减。通过配置 CTCR 寄存器和 DIR 寄存器即可实现相应功能。递增计数时，计数器 x 被使能后从 0 开始计数，计数到 MR0/1/2/3 后，继续从 0 开始计数。递减计数时，计数器 x 被使能后从 MRx 开始递减计数，计数到 0 后，继续从 MRx 开始递减计数。如果是中心对齐方式计数，计数器 x 被使能后从 0 开始递增计数，计数到 MRx 后递减计数，一直到 0，然后再重复计数。如果中断被使能会置位中断标志位。



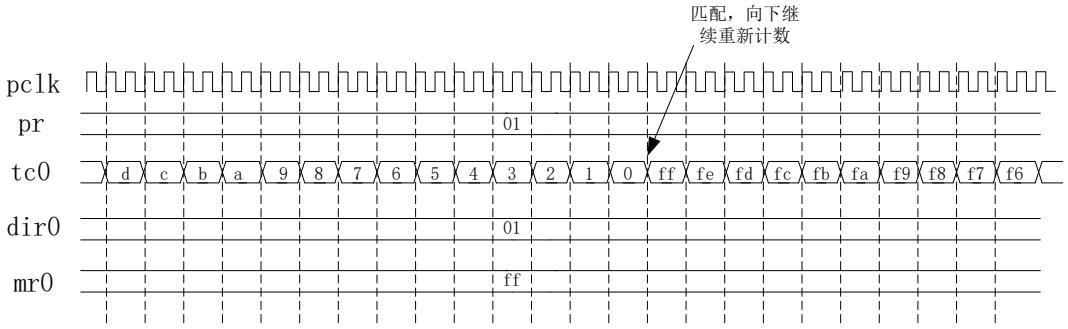
图：定时器边沿对齐递增方式计数模式波形图(MR=3D)



图：定时器中心对齐方式计数模式波形图(MR=3D)

6.10.5.2 定时器模式

在此模式下 PC 寄存器在每个 PCLK 的边沿加 1。当它达到预分频寄存器中存储的值时，TCx 计数一次， PC 将在下一个 PCLK 复位。这就使得当 PR=0 时，TCx 在每个 PCLK 时计数一次，PR=1 时，TCx 在每 2 个 PCLK 时计数一次，以此类推。计数方式与（1）相同。



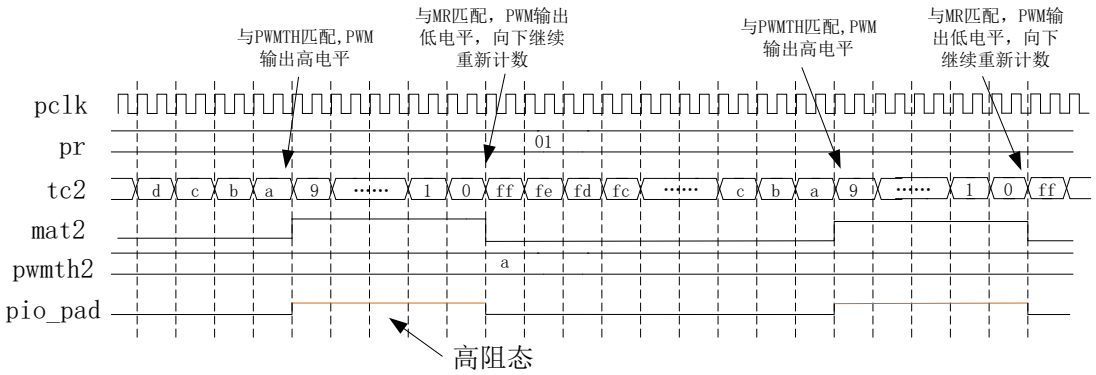
图：定时器向下计数模式波形图(MR=FF)

6.10.5.3 单边沿控制的 PWM 输出

在该操作下，分别设置各匹配输出，将 PWMTH0、PWMTH1、PWMTH2、PWMTH3 分别配置为 PWM 输出 MAT0/1/2/3 的脉冲占空比，MR0/1/2/3 来设置 PWM 的周期长度。每个 PWM 都相互独立，互不影响。例如计数器 2 被用作 PWM2 的计数器。当为递增模式时，计数器 2 递增计数，当与 PWMHT2 寄存器发生匹配时，相应的 PWM 输出置为高电平（如果 ACTIVE_HIZ 为 1，则 MAT2_OEN 为 0，则相应的 PWM 输出管脚为高阻态）；当与 MR2 匹配时，相应的 PWM 输出置为低电平（如果 INACTIVE_HIZ 为 1，则 MAT2_OEN 为 0，则相应的 PWM 输出管脚为高阻态），计数器 2 复位为 0，重新计数。当为递减模式时，计数器 2 从 MR2 递减计数，当与 PWMTH2 寄存器发生匹配时，相应的 PWM 输出置为高电平（如果 ACTIVE_HIZ 为 1，则 MAT2_OEN 为 0，则相应的 PWM 输出管脚为高阻态）；当计数到 0 时，相应的 PWM 输出置为低电平（如果 INACTIVE_HIZ 为 1，则 MAT2_OEN 为 0，则相应的 PWM 输出

管脚为高阻态），计数器 2 复位为 MR2,并重新计数。当为中心对齐方式时，计数器 2 从 0 递增计数，当与 PWMTH2 寄存器发生匹配时，相应的 PWM 输出置为高电平（如果 ACTIVE_HIZ 为 1，则 MAT2_OEN 为 0，则相应的 PWM 输出管脚为高阻态）；当递减计数，再次发生匹配时，相应的 PWM 输出置为低电平（如果 INACTIVE_HIZ 为 1，则 MAT2_OEN 为 0，则相应的 PWM 输出管脚为高阻态）。

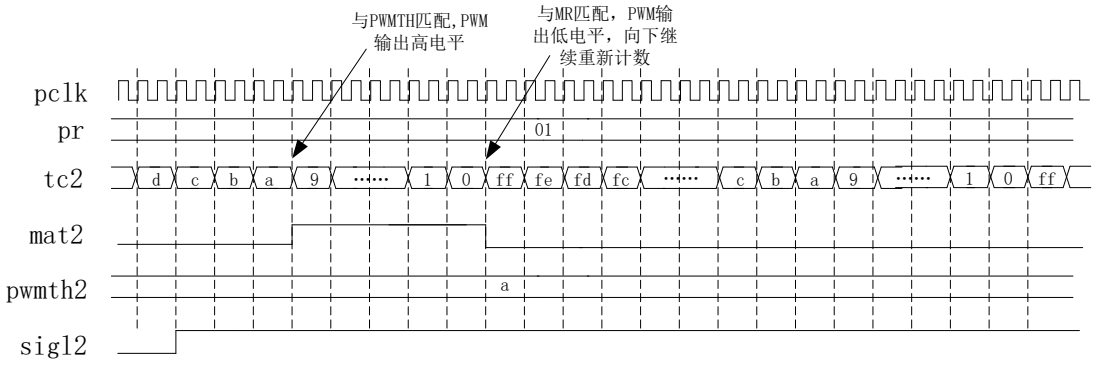
下图为 PWM 模式输出，MAT2 输出为 PWM 脉冲，但由于 ACTIVE_HIZ 为 1，则 MAT2_OEN 为 0，相应的管脚输出 PIO_PAD 为有效态为高阻态，无效态为低电平。



图：定时器递减模式下 PWM 模式波形图

6.10.5.4 单脉冲 PWM 输出

当设置为 PWM 模式时，并设置为单脉冲输出方式，相应的 PWM 输出只输出一次脉冲。

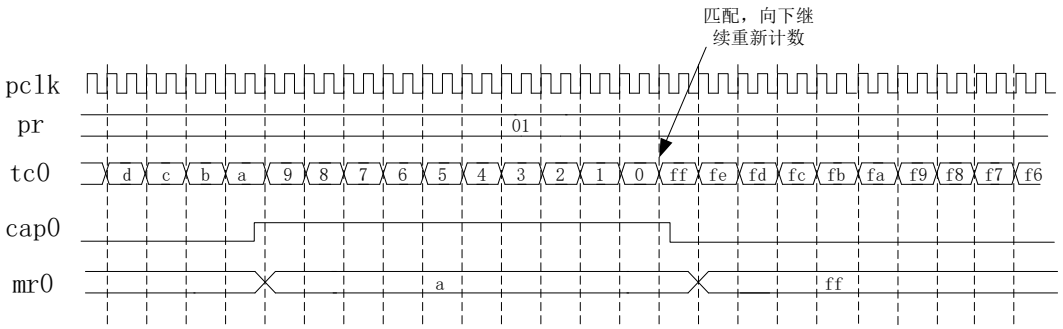


图：定时器单脉冲输出模式波形图

6.10.5.5 捕获操作

每个定时器通道在对应的输入信号跳变时捕捉定时器的瞬时值，捕获事件也可选来产生中断。在此操作下通过配置 CCR 寄存器即可实现将 TCx 的内容装入 CRx 中。

在捕获模式下，可以使能硬件测量占空比功能。硬件会自动测量输入信号的频率和占空比。

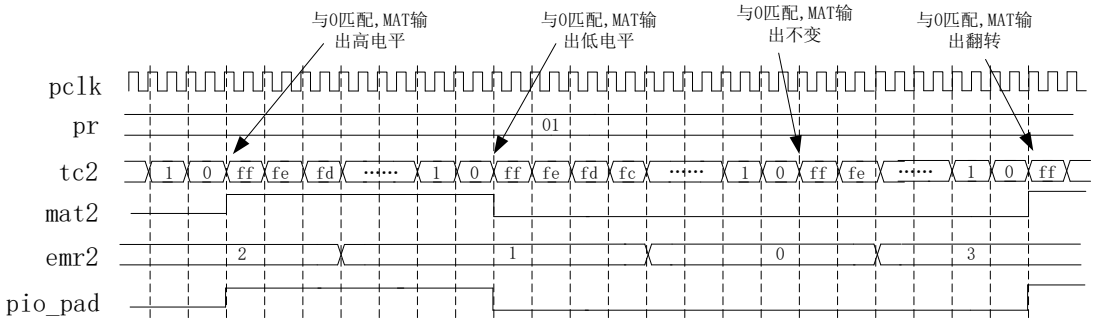


图：定时器捕获模式波形图

6.10.5.6 匹配操作

TCx 的值与 MRx 寄存器的值（递增）或 0（递减）相比较，当两个值相等时，自动触发相应操作，这些操作包括产生中断、复位定时器计数器或停止定时器，所有操作由 MCR 寄存器控制。如果发生停止操作，计数器被禁能并保持当前值，若想继续进行计数，首先得清除 MCR 寄存器中相应的停止位，然后再配置 TCR 寄存器，使能该计数器。外部匹配管脚 MAT [3:0]是与 MRO/1/2/3 相对应的外部输出。通过配置 EMR 寄存器，可实现对 MAT [3:0]的控制，能实现以下功能：匹配时输出低电平；匹配时输出高电平；匹配时翻转； 匹配时不执行任何操作。这些操作的输出不受 ACTIVE_HIZ 和 INACTIVE_HIZ 的影响。

下图所示：即使 ACTIVE_HIZ 为 1，但在匹配模式时，管脚输出不受 HIZ 的影响。



图：定时器匹配操作波形图

6.10.6 寄存器映射

R：只读，W：只写，R/W：可读写

寄存器	偏移量	R/W	描述	复位后的值
TIMER 的基地址： **TIMER0_BA = 0x4000_9000**				
IR	TIMERx_BA+0x000	R/W	中断寄存器	0x0000_0000
TCR	TIMERx_BA+0x010	R/W	定时器控制寄存器	0x0000_0000
DIR	TIMERx_BA+0x020	R/W	计数方式寄存器	0x0000_0000
TC0	TIMERx_BA+0x030	R	定时器计数器0寄存器	0x0000_0000
TC1	TIMERx_BA+0x040	R	定时器计数器1寄存器	0x0000_0000
TC2	TIMERx_BA+0x050	R	定时器计数器2寄存器	0x0000_0000
TC3	TIMERx_BA+0x060	R	定时器计数器3寄存器	0x0000_0000
PR	TIMERx_BA+0x070	R/W	预分频寄存器	0x0000_0000
PC	TIMERx_BA+0x080	R	预分频计数器寄存器	0x0000_0000
MCR	TIMERx_BA+0x090	R/W	匹配控制寄存器	0x0000_0000
MR0	TIMERx_BA+0x0A0	R/W	匹配寄存器0	0x0000_0000
MR1	TIMERx_BA+0x0B0	R/W	匹配寄存器1	0x0000_0000
MR2	TIMERx_BA+0x0C0	R/W	匹配寄存器2	0x0000_0000

MR3	TIMERx_BA+0x0D0	R/W	匹配寄存器3	0x0000_0000
CCR0	TIMERx_BA+0x0E0	R/W	捕获控制寄存器0	0x0000_0000
CCR1	TIMERx_BA+0x200	R/W	捕获控制寄存器1	0x0000_0000
CR0	TIMERx_BA+0x0F0	R	捕获寄存器0	0x0000_0000
CR1	TIMERx_BA+0x100	R	捕获寄存器1	0x0000_0000
CR2	TIMERx_BA+0x110	R	捕获寄存器2	0x0000_0000
CR3	TIMERx_BA+0x120	R	捕获寄存器3	0x0000_0000
EMR	TIMERx_BA+0x130	R/W	外部匹配寄存器	0x0000_0000
PWMTH0	TIMERx_BA+0x140	R/W	PWM0占空比阈值寄存器	0x0000_0000
PWMTH1	TIMERx_BA+0x150	R/W	PWM1占空比阈值寄存器	0x0000_0000
PWMTH2	TIMERx_BA+0x160	R/W	PWM2占空比阈值寄存器	0x0000_0000
PWMTH3	TIMERx_BA+0x170	R/W	PWM3占空比阈值寄存器	0x0000_0000
CTCR	TIMERx_BA+0x180	R/W	计数控制寄存器	0x0000_0000
PWMC	TIMERx_BA+0x190	R/W	PWM控制寄存器	0x0000_0000
PWMV0	TIMERx_BA+0x1A0	R/W	PWM0脉冲输出个数寄存器	0x0000_0000
PWMV1	TIMERx_BA+0x1B0	R/W	PWM1脉冲输出个数寄存器	0x0000_0000
PWMV2	TIMERx_BA+0x1C0	R/W	PWM2脉冲输出个数寄存器	0x0000_0000
PWMV3	TIMERx_BA+0x1D0	R/W	PWM3脉冲输出个数寄存器	0x0000_0000
PERIOD0H	TIMERx_BA+0x220	R/W	通道0信号高电平周期寄存器	0x0000_0000
PERIOD0L	TIMERx_BA+0x230	R/W	通道0信号低电平周期寄存器	0x0000_0000
PERIOD1H	TIMERx_BA+0x240	R/W	通道1信号高电平周期寄存器	0x0000_0000
PERIOD1L	TIMERx_BA+0x250	R/W	通道1信号低电平周期寄存器	0x0000_0000
PERIOD2H	TIMERx_BA+0x260	R/W	通道2信号高电平周期寄存器	0x0000_0000
PERIOD2L	TIMERx_BA+0x270	R/W	通道2信号低电平周期寄存器	0x0000_0000
PERIOD3H	TIMERx_BA+0x280	R/W	通道3信号高电平周期寄存器	0x0000_0000
PERIOD3L	TIMERx_BA+0x290	R/W	通道3信号低电平周期寄存器	0x0000_0000

6.10.6.1 IR 寄存器

该寄存器包含 4 个用于匹配中断的位及 1 个用于捕获中断的位。如果有中断产生，则 IR 中的相应位为高电平。否则，该位为低电平。向对应的 IR 位写逻辑 1 会使中断复位。写 0 无效。

寄存器	偏移量	R/W	描述	复位值
IR	**TIMER0_BA+0x000**	**R/W**	**中断寄存器**	**0x0000_0000**

31	30	29	28	27	26	25	24
保留							
23	22	21	20	19	18	17	16
保留							
15	14	13	12	11	10	9	8
保留				CR3	CR2	CR1	SING3
7	6	5	4	3	2	1	0
SING2	SING1	SING0	CR0	MR3	MR2	MR1	MR0

Bits	描述	
[31:12]	保留	保留
[11]	CR3	**捕获通道 3 事件的中断标志** 1：发生捕获通道 3 的事件中断 0：未发生捕获通道 3 的事件中断
[10]	CR2	**捕获通道 2 事件的中断标志** 1：发生捕获通道 2 的事件中断 0：未发生捕获通道 2 的事件中断
[9]	CR1	**捕获通道 1 事件的中断标志** 1：发生捕获通道 1 的事件中断

		0：未发生捕获通道 1 的事件中断
[8]	SING3	**PWM3 通道输出特定个 PWM 波形的中断标志** 1：发生中断 0：未发生中断
[7]	SING2	**PWM2 通道输出特定个 PWM 波形的中断标志** 1：发生中断 0：未发生中断
[6]	SING1	**PWM1 通道输出特定个 PWM 波形的中断标志** 1：发生中断 0：未发生中断
[5]	SING0	**PWM0 通道输出特定个 PWM 波形的中断标志** 1：发生中断 0：未发生中断
[4]	CR0	**捕获通道 0 事件的中断标志** 1：发生捕获通道 0 的事件中断 0：未发生捕获通道 0 的事件中断
[3]	MR3	**匹配通道 3 的中断标志** 1：发生匹配通道 3 的中断 0：未发生匹配通道 3 的中断
[2]	MR2	**匹配通道 2 的中断标志** 1：发生匹配通道 2 的中断 0：未发生匹配通道 2 的中断
[1]	MR1	**匹配通道 1 的中断标志** 1：发生匹配通道 1 的中断 0：未发生匹配通道 1 的中断
[0]	MR0	**匹配通道 0 的中断标志** 1：发生匹配通道 0 的中断 0：未发生匹配通道 0 的中断

6.10.6.2 TCR0 寄存器

该寄存器用于控制计数器/定时器的操作。

寄存器	偏移量	R/W	描述	复位值
TCR0	TIMER0_BA+0x010	R/W	定时器控制寄存器	0x00f0_0000

31	30	29	28	27	26	25	24
保留							
23	22	21	20	19	18	17	16
PWM3IO	PWM2IO	PWM1IO	PWM0IO	PWM3EN	PWM2EN	PWM1EN	PWM0EN
15	14	13	12	11	10	9	8
CNT3OUT		CNT2OUT		CNT1OUT		CNT0OUT	
7	6	5	4	3	2	1	0
CNT3RST	CNT2RST	CNT1RST	CNT0RST	CNT3EN	CNT2EN	CNT1EN	CNT0EN

Bits	描述	
[31:24]	保留	保留
[23]	PWM3IO	使能 CT0IO3 的输出功能，默认为 1 1：使能 CT0IO3 管脚的 PWM 输出功能 0：CT0IO3 管脚为 CAP 输入功能
[22]	PWM2IO	使能 CT0IO2 的输出功能，默认为 1 1：使能 CT0IO2 管脚的 PWM 输出功能 0：CT0IO2 管脚为 CAP 输入功能
[21]	PWM1IO	使能 CT0IO1 的输出功能，默认为 1 1：使能 CT0IO1 管脚的 PWM 输出功能 0：CT0IO1 管脚为 CAP 输入功能
[20]	PWM0IO	使能 CT0IO0 的输出功能，默认为 1

		1：使能 CT0IO0 管脚的 PWM 输出功能 0：CT0IO0 管脚为 CAP 输入功能
[19]	PWM3EN	使能 MAT3 的 PWM 功能 1：使能 MAT3 的 PWM 功能 0：禁止 MAT3 的 PWM 功能
[18]	PWM2EN	使能 MAT2 的 PWM 功能 1：使能 MAT2 的 PWM 功能 0：禁止 MAT2 的 PWM 功能
[17]	PWM1EN	使能 MAT1 的 PWM 功能 1：使能 MAT1 的 PWM 功能 0：禁止 MAT1 的 PWM 功能
[16]	PWM0EN	使能 MAT0 的 PWM 功能 1：使能 MAT0 的 PWM 功能 0：禁止 MAT0 的 PWM 功能
[15:14]	CNT3OUT	设置定时器 3 未使能时 MAT3 管脚上的输出电平 00：MAT3 的输出电平保持不变 01：MAT3 上输出 0 10：MAT3 上输出 1 11：MAT3 上输出高阻
[13:12]	CNT2OUT	设置定时器 2 未使能时 MAT2 管脚上的输出电平 00：MAT2 的输出电平保持不变 01：MAT2 上输出 0 10：MAT2 上输出 1 11：MAT2 上输出高阻
[11:10]	CNT1OUT	设置定时器 1 未使能时 MAT1 管脚上的输出电平 00：MAT1 的输出电平保持不变 01：MAT1 上输出 0 10：MAT1 上输出 1 11：MAT1 上输出高阻
[9: 8]	CNT0OUT	设置定时器 0 未使能时 MAT0 管脚上的输出电平

		00：MAT0 的输出电平保持不变 01：MAT0 上输出 0 10：MAT0 上输出 1 11：MAT0 上输出高阻
[7]	CNT3RST	**复位定时器/计数器 3** 1：设为 1 后，在下一个时钟沿定时器/计数器 3 被复位 0：释放定时器/计数器 3 的复位
[6]	CNT2RST	**复位定时器/计数器 2** 1：设为 1 后，在下一个时钟沿定时器/计数器 2 被复位 0：释放定时器/计数器 2 的复位
[5]	CNT1RST	**复位定时器/计数器 1** 1：设为 1 后，在下一个时钟沿定时器/计数器 1 被复位 0：释放定时器/计数器 1 的复位
[4]	CNT0RST	**复位定时器/计数器 0** 1：设为 1 后，在下一个时钟沿定时器/计数器 0 被复位 0：释放定时器/计数器 0 的复位
[3]	CNT3EN	**使能定时器/计数器 3** 1：使能定时器/计数器 3 0：禁止定时器/计数器 3
[2]	CNT2EN	**使能定时器/计数器 2** 1：使能定时器/计数器 2 0：禁止定时器/计数器 2
[1]	CNT1EN	**使能定时器/计数器 1** 1：使能定时器/计数器 1 0：禁止定时器/计数器 1
[0]	CNT0EN	**使能定时器/计数器 0** 1：使能定时器/计数器 0 0：禁止定时器/计数器 0

6.10.6.3 DIR0 寄存器

该寄存器包含 4 个计数器计数模式以及计数方向的控制位。决定计数器是向上、向下还是先向上再向下计数的中心对齐方式。中心对齐方式被置位后递增/递减模式被禁能。同时该寄存器还包含 4 个通道 PWM 输出的载波功能的控制位，使能一个通道的载波功能后，该通道的实际输出将会与指定的另一个通道的原始输出相与后进行输出。

寄存器	偏移量	R/W	描述	复位值
DIR0	**TIMER0_BA+0x020**	**R/W**	**计数方式寄存器**	**0x0000_0000**

31	30	29	28	27	26	25	24
CARR3		CARR2		CARR1		CARR0	
23	22	21	20	19	18	17	16
保留							
15	14	13	12	11	10	9	8
保留		DIR3		保留		DIR2	
7	6	5	4	3	2	1	0
保留		DIR1		保留		DIR0	

Bits	描述	
[31:30]	CARR3	**设置通道 3 的载波输出功能** 00：输出通道 3 的原始 PWM 波形 01：与通道 0 的原始 PWM 波形相与后输出 10：与通道 1 的原始 PWM 波形相与后输出 11：与通道 2 的原始 PWM 波形相与后输出
[29:28]	CARR2	**设置通道 2 的载波输出功能** 00：输出通道 2 的原始 PWM 波形

		01：与通道 3 的原始 PWM 波形相与后输出 10：与通道 0 的原始 PWM 波形相与后输出 11：与通道 1 的原始 PWM 波形相与后输出
[27:26]	CARR1	**设置通道 1 的载波输出功能** 00：输出通道 1 的原始 PWM 波形 01：与通道 2 的原始 PWM 波形相与后输出 10：与通道 3 的原始 PWM 波形相与后输出 11：与通道 0 的原始 PWM 波形相与后输出
[25:24]	CARR0	**设置通道 0 的载波输出功能** 00：输出通道 0 的原始 PWM 波形 01：与通道 1 的原始 PWM 波形相与后输出 10：与通道 2 的原始 PWM 波形相与后输出 11：与通道 3 的原始 PWM 波形相与后输出
[23:14]	**保留**	**保留**
[13:12]	DIR3	**设置定时器/计数器 3 的计数模式和计数方向** 00：定增计数 01：定减计数 10：中心对齐计数 11：保留
[11:10]	**保留**	**保留**
[9:8]	DIR2	**设置定时器/计数器 2 的计数模式和计数方向** 00：定增计数 01：定减计数 10：中心对齐计数 11：保留
[7:6]	**保留**	**保留**
[5:4]	DIR1	**设置定时器/计数器 1 的计数模式和计数方向** 00：定增计数

		01：定减计数 10：中心对齐计数 11：保留
[3:2]	**保留**	**保留**
[1:0]	DIR0	**设置定时器/计数器 0 的计数模式和计数方向** 00：定增计数 01：定减计数 10：中心对齐计数 11：保留

6.10.6.4 TC0/1/2/3 寄存器

当预分频计数器达到计数上限时，16 位的 TC0/1/2/3 加 1。如果 TC0/1/2/3 在到达计数器上限之前没有复位，它将一直计数到 0xFFFF 然后翻转到 0x0000。该事件不会产生中断，但是如果需要的话，可使用匹配寄存器检测溢出。

寄存器	偏移量	R/W	描述	复位值
TC0/1/2/3	**TIMER0_BA+0x030** **TIMER0_BA+0x040** **TIMER0_BA+0x050** **TIMER0_BA+0x060**	**R**	**定时器计数器0/1/2/3寄存器**	**0x0000_0000**

31	30	29	28	27	26	25	24
保留							
23	22	21	20	19	18	17	16
保留							
15	14	13	12	11	10	9	8
COUNT[15:8]							

7	6	5	4	3	2	1	0
COUNT[7:0]							

Bits	描述	
[15:0]	COUNT	反映 32 位定时器/计数器的计数值。

6.10.6.5 PR0 寄存器

16 位预分频寄存器 PRx 指定了预分频计数器的最大计数值。

寄存器	偏移量	R/W	描述	复位值
PR0	**TIMER0_BA+0x070**	**R/W**	预分频寄存器	**0x0000_0000**

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

Bits	描述	
[15:0]	COUNT	**设置预分频计数器的最大计数值** 当预分频计数器的值达到这个值时会返回 0 重新开始计数

6.10.6.6 PC 寄存器

16 位预分频计数器寄存器 PC 用某个常量来控制 PCLK 的分频，使其分频后输入到定时器计数器。这样就可以控制定时器计数器精度和定时器溢出前所能达到的最大值之间的关系。预分频计数器在每个 PCLK 周期加 1，当它达到预分频寄存器中设置的值时，定时器计数器计数一次 ，预分频计数器将在下一个 PCLK 复位。这就使得当 PR=0 时，TC0 在每个 PCLK 时加 1，PR=1 时，TC0 在每 2 个 PCLK 时计数一次 ，以此类推。

寄存器	偏移量	R/W	描述	复位值
PC	TIMER0_BA+0x080	R	分频计数器寄存器	0x0000_0000

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

Bits	描述	
[15:0]	COUNT	反映预分频计数器的当前计数值

6.10.6.7 MCR0 寄存器

该寄存器用于控制当其中一个匹配寄存器的值与定时器计数器的值匹配时，应当执行的操作。

寄存器	偏移量	R/W	描述	复位值

MCR0	TIMER0_BA+0x090	R/W	匹配控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
保留							
23	22	21	20	19	18	17	16
保留							
15	14	13	12	11	10	9	8
保留				MR3S	MR3R	MR3I	MR2S
7	6	5	4	3	2	1	0
MR2R	MR2I	MR1S	MR1R	MR1I	MR0S	MR0R	MR0I

Bits	描述	
[31:12]	保留	保留
[11]	MR3S	**使能匹配寄存器 3 的停止功能** 1：当匹配寄存器 3 与 TC3 的值匹配时，将停止 TC3，同时 TCR 寄存器的 bit3 置 0 0：禁止匹配寄存器 3 的停止功能
[10]	MR3R	**使能匹配寄存器 3 的复位功能** 1：当匹配寄存器 3 与 TC3 的值匹配时，复位 TC3 0：禁止匹配寄存器 3 的复位功能
[9]	MR3I	**使能匹配寄存器 3 的中断功能** 1：当匹配寄存器 3 与 TC3 的值匹配时产生中断 0：禁止匹配寄存器 3 的中断功能
[8]	MR2S	**使能匹配寄存器 2 的停止功能** 1：当匹配寄存器 2 与 TC2 的值匹配时，将停止 TC2，同时 TCR 寄存器的 bit2 置 0 0：禁止匹配寄存器 2 的停止功能

[7]	MR2R	**使能匹配寄存器 2 的复位功能** 1：当匹配寄存器 2 与 TC2 的值匹配时，复位 TC2 0：禁止匹配寄存器 2 的复位功能
[6]	MR2I	**使能匹配寄存器 2 的中断功能** 1：当匹配寄存器 2 与 TC2 的值匹配时产生中断 0：禁止匹配寄存器 2 的中断功能
[5]	MR1S	**使能匹配寄存器 1 的停止功能** 1：当匹配寄存器 1 与 TC1 的值匹配时，将停止 TC1，同时 TCR 寄存器的 bit1 置 0 0：禁止匹配寄存器 1 的停止功能
[4]	MR1R	**使能匹配寄存器 1 的复位功能** 1：当匹配寄存器 1 与 TC1 的值匹配时，复位 TC1 0：禁止匹配寄存器 1 的复位功能
[3]	MR1I	**使能匹配寄存器 1 的中断功能** 1：当匹配寄存器 1 与 TC1 的值匹配时产生中断 0：禁止匹配寄存器 1 的中断功能
[2]	MR0S	**使能匹配寄存器 0 的停止功能** 1：当匹配寄存器 0 与 TC0 的值匹配时，将停止 TC0，同时 TCR 寄存器的 bit0 置 0 0：禁止匹配寄存器 0 的停止功能
[1]	MR0R	**使能匹配寄存器 0 的复位功能** 1：当匹配寄存器 0 与 TC0 的值匹配时，复位 TC0 0：禁止匹配寄存器 0 的复位功能
[0]	MR0I	**使能匹配寄存器 0 的中断功能** 1：当匹配寄存器 0 与 TC0 的值匹配时产生中断 0：禁止匹配寄存器 0 的中断功能

6.10.6.8 MR0/1/2/3 寄存器

该寄存器的值不断地与定时器计数器的值相比较，当两个值相等时，自动触发相应操作。这些操作包括产生中断，复位定时器计数器或停止定时器。所有操作由 MCR 寄存器控制。

寄存器	偏移量	R/W	描述	复位值
MR0/1/2/3	TIMER0_BA+0x0A0 TIMER0_BA+0x0B0 TIMER0_BA+0x0C0 TIMER0_BA+0x0D0	R/W	匹配寄存器0/1/2/3	0x0000_0000

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

Bits	描述	
[15:0]	VALUE	设置通道 0/1/2/3 的匹配值

6.10.6.9 CCR0 寄存器

该寄存器用于控制当捕获事件发生时，是否将定时器计数器中的值装入捕获寄存器中，以及捕获事件是否产生中断。同时设置上升沿位和下降沿位是一种有

效的配置，它使两个边沿都产生捕获事件。该寄存器还可以使能硬件自动测量功能来计算输入信号的周期和占空比。

寄存器	偏移量	R/W	描述	复位值
CCR0	TIMER0_BA+0x0E0	R/W	捕获控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
保留				MEAS3	MEAS2	MEAS1	MEAS0
23	22	21	20	19	18	17	16
CNT3CAP		CNT2CAP		CNT1CAP		CNT0CAP	
15	14	13	12	11	10	9	8
CAP1FIR				保留	CAP1I	CAP1FE	CAP1RE
7	6	5	4	3	2	1	0
CAP0FIR				保留	CAP0I	CAP0FE	CAP0RE

Bits	描述	
[31:28]	保留	保留
[27]	MEAS3	1：使能通道 3 的自动测量功能 0：不使能硬件测量功能
[26]	MEAS2	1：使能通道 2 的自动测量功能 0：不使能硬件测量功能
[25]	MEAS1	1：使能通道 1 的自动测量功能 0：不使能硬件测量功能
[24]	MEAS0	1：使能通道 0 的自动测量功能 0：不使能硬件测量功能
[23:22]	CNT3_CAP	为 CNT3 计数器选择采用的输入通道 00：表示使用 CAP0 管脚 01：表示使用 CAP1 管脚

		10：表示使用 CAP2 管脚 11：表示使用 CAP3 管脚
[21:20]	CNT2_CAP	**为 CNT2 计数器选择采用的输入通道** 00：表示使用 CAP0 管脚 01：表示使用 CAP1 管脚 10：表示使用 CAP2 管脚 11：表示使用 CAP3 管脚
[19:18]	CNT1_CAP	**为 CNT1 计数器选择采用的输入通道** 00：表示使用 CAP0 管脚 01：表示使用 CAP1 管脚 10：表示使用 CAP2 管脚 11：表示使用 CAP3 管脚
[17:16]	CNT0_CAP	**为 CNT0 计数器选择采用的输入通道** 00：表示使用 CAP0 管脚 01：表示使用 CAP1 管脚 10：表示使用 CAP2 管脚 11：表示使用 CAP3 管脚
[15:12]	CAP1FIR	**使能捕获通道 1 的滤波功能** CAP1 管脚上小于 CPA1FIR 个系统周期的脉冲会被过滤掉
[11]	保留	保留
[10]	CAP1I	**使能捕获事件 1 的的中断** 1：捕获事件导致的 CR1 寄存器装载会产生中断 0：禁止产生中断
[9]	CAP1FE	**使能捕获通道 1 下降沿捕获功能** 1：捕获通道上 1 到 0 的跳变将使 TC1 的值载入 CR1 0：禁止下降沿捕获功能
[8]	CAP1RE	**使能捕获通道 1 上升沿捕获功能** 1：捕获通道上 0 到 1 的跳变将使 TC1 的值载入 CR1

		0：禁止上升沿捕获功能
[7:4]	CAP0FIR	**使能捕获通道 0 的滤波功能** CAP0 管脚上小于 CPA0FIR 个系统周期的脉冲会被过滤掉
[3]	保留	保留
[2]	CAP0I	**使能捕获事件 0 的的中断** 1：捕获事件导致的 CR0 寄存器装载会产生中断 0：禁止产生中断
[1]	CAP0FE	**使能捕获通道 0 下降沿捕获功能** 1：捕获通道上 1 到 0 的跳变将使 TC0 的值载入 CR0 0：禁止下降沿捕获功能
[0]	CAP0RE	**使能捕获通道 0 上升沿捕获功能** 1：捕获通道上 0 到 1 的跳变将使 TC0 的值载入 CR0 0：禁止上升沿捕获功能

6.10.6.10 CCR1 寄存器

该寄存器用于控制当捕获事件发生时，是否将定时器计数器中的值装入捕获寄存器中，以及捕获事件是否产生中断。同时设置上升沿位和下降沿位是一种有效的配置，它使两个边沿都产生捕获事件。

寄存器	偏移量	R/W	描述	复位值
CCR1	**TIMER0_BA+0x200**	**R/W**	**捕获控制寄存器1**	**0x0000_0000**

31	30	29	28	27	26	25	24
保留							
23	22	21	20	19	18	17	16
保留							
15	14	13	12	11	10	9	8

CAP3FIR				保留	CAP3I	CAP3FE	CAP3RE
7	6	5	4	3	2	1	0
CAP2FIR				保留	CAP2I	CAP2FE	CAP2RE

Bits	描述	
[31:16]	保留	保留
[15:12]	CAP3FIR	**使能捕获通道 3 的滤波功能** CAP3 管脚上小于 CPA3FIR 个系统周期的脉冲会被过滤掉
[11]	保留	保留位
[10]	CAP3I	**使能捕获事件 3 的的中断** 1：捕获事件导致的 CR3 寄存器装载会产生中断 0：禁止产生中断
[9]	CAP3FE	**使能捕获通道 3 下降沿捕获功能** 1：捕获通道上 1 到 0 的跳变将使 TC3 的值载入 CR3 0：禁止下降沿捕获功能
[8]	CAP3RE	**使能捕获通道 3 上升沿捕获功能** 1：捕获通道上 0 到 1 的跳变将使 TC3 的值载入 CR3 0：禁止上升沿捕获功能
[7:4]	CAP2FIR	**使能捕获通道 2 的滤波功能** CAP2 管脚上小于 CPA2FIR 个系统周期的脉冲会被过滤掉
[3]	保留	保留位
[2]	CAP2I	**使能捕获事件 2 的的中断** 1：捕获事件导致的 CR2 寄存器装载会产生中断 0：禁止产生中断
[1]	CAP2FE	**使能捕获通道 2 下降沿捕获功能** 1：捕获通道上 1 到 0 的跳变将使 TC2 的值载入 CR2 0：禁止下降沿捕获功能

[0]	CAP2RE	**使能捕获通道 2 上升沿捕获功能** 1：捕获通道上 0 到 1 的跳变将使 TC2 的值载入 CR2 0：禁止上升沿捕获功能

6.10.6.11 CR0/1/2/3 寄存器

该组寄存器与捕获管脚相关联，当捕获管脚发生特定的事件时，可将定时器计数器 0/1/2/3 的值装入相应捕获寄存器 0/1/2/3。CCR 捕获控制寄存器中的设置决定是否使能捕获功能，以及在捕获管脚的上升沿、下降沿或双边沿时是否产生捕获事件。

寄存器	偏移量	R/W	描述	复位值
CR0/1/2/3	**TIMER_BA+0x0F0** **TIMER_BA+0x100** **TIMER_BA+0x110** **TIMER_BA+0x120**	**R**	**捕获寄存器0/1/2/3**	**0x0000_0000**

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

Bits	描述	
[15:0]	VALUE	VALUE

6.10.6.12 EMR 寄存器

该寄存器提供了外部匹配管脚 MAT [3:0]的控制和状态。如果匹配输出配置为 PWM 输出，则外部匹配寄存器的功能由 PWM 规则决定。

寄存器	偏移量	R/W	描述	复位值
EMR	TIMER0_BA+0x130	R/W	外部匹配寄存器	0x0000_0000

31	30	29	28	27	26	25	24
保留							
23	22	21	20	19	18	17	16
保留							
15	14	13	12	11	10	9	8
EMO3	EMO2	EMO1	EMO0	EMC3		EMC2	
7	6	5	4	3	2	1	0
EMC1		EMC0		EM3	EM2	EM1	EM0

Bits	描述	
[31:16]	保留	保留
[15]	EMO3	设置匹配功能下 MAT3 的默认输出电平 0：默认输出 0 1：默认输出 1
[14]	EMO2	设置匹配功能下 MAT2 的默认输出电平 0：默认输出 0 1：默认输出 1
[13]	EMO1	设置匹配功能下 MAT1 的默认输出电平 0：默认输出 0 1：默认输出 1

[12]	EMOO	**设置匹配功能下 MAT0 的默认输出电平** 0：默认输出 0 1：默认输出 1
[11:10]	EMC3	**设置外部匹配控制 3** 00：不执行任何操纵 01：将对应的外部匹配位/输出设置为 0 10：将对应的外部匹配位/输出设置为 1 11：将对应的外部匹配位/输出翻转
[9:8]	EMC2	**设置外部匹配控制 2** 00：不执行任何操纵 01：将对应的外部匹配位/输出设置为 0 10：将对应的外部匹配位/输出设置为 1 11：将对应的外部匹配位/输出翻转
[7:6]	EMC1	**设置外部匹配控制 1** 00：不执行任何操纵 01：将对应的外部匹配位/输出设置为 0 10：将对应的外部匹配位/输出设置为 1 11：将对应的外部匹配位/输出翻转
[5:4]	EMC0	**设置外部匹配控制 0** 00：不执行任何操纵 01：将对应的外部匹配位/输出设置为 0 10：将对应的外部匹配位/输出设置为 1 11：将对应的外部匹配位/输出翻转
[3]	EM3	**外部匹配 3** 该位反映输出MAT3的状态，不管该输出是否连接到管脚。当TC3和MR3匹配时，定时器的输出可以翻转电平，变为低电平，变为高电平或不执行任何动作。位EMR[11:10]控制该输出的功能.
[2]	EM2	**外部匹配 2**

		该位反映输出MAT2的状态，不管该输出是否连接到管脚。当TC2和MR2匹配时，定时器的输出可以翻转电平，变为低电平，变为高电平或不执行任何动作。位EMR[9:8]控制该输出的功能.
[1]	EM1	**外部匹配 1** 该位反映输出MAT1的状态，不管该输出是否连接到管脚。当TC1和MR1匹配时，定时器的输出可以翻转电平，变为低电平，变为高电平或不执行任何动作。位EMR[7:6]控制该输出的功能.
[0]	EM0	**外部匹配 0** 该位反映输出MAT0的状态，不管该输出是否连接到管脚。当TC0和MR0匹配时，定时器的输出可以翻转电平，变为低电平，变为高电平或不执行任何动作。位EMR[5:4]控制该输出的功能.

6.10.6.13 PWMTH0/1/2/3 寄存器

该寄存器用于 PWM 模式下，与 MR0/1/2/3 共同决定 PWM 的占空比。当为向上计数模式时，计数器从 0 开始递增计数，计数到 TH0/1/2/3 时，PWM 变高，继续递增到 MR0/1/2/3 时，PWM 变低，计数器同时也被复位 0。由此可知，该 PWM 的周期为 MR0/1/2/3，高电平宽度为 MR0/1/2/3-TH0/1/2/3.当为向下计数模式时，计数器从 MR0/1/2/3 开始递减计数，计数到 TH0/1/2/3 时，PWM 变高，继续递减到 0，PWM 变低，计数器同时被复位到 MR0/1/2/3。由此可知，该 PWM 的周期为 MR0/1/2/3，高电平宽度为 TH0/1/2/3。当为中心对齐方式时，计数器从 0 开始递增计数，计数到 TH0/1/2/3 时，PWM 变高电平，继续递增到 MR0/1/2/3，开始递减计数，计数到 TH0/1/2/3 时，PWM 变低电平，继续递减至 0。由此可知，该 PWM 的周期为 2xMR0/1/2/3，高电平宽度为 2x（MR0/1/2/3-TH0/1/2/3）。

寄存器	偏移量	R/W	描述	复位值
PWMTH0/1/2/3	**TIMER0_BA+0x140** **TIMER0_BA+0x150**	**R/W**	**PWM0/1/2/3占空比阈值寄存器**	**0x0000_0000**

	TIMER0_BA+0x160 **TIMER0_BA+0x170**			

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

Bits	描述	
[15:0]	VALUE	PWM 的阈值

6.10.6.14 CTCR0 寄存器

该寄存器用于在定时器和计数器模式之间进行选择，并且在计数模式下选择计数的管脚和边沿。

工作模式选择为计数器模式时，在 PCLK 时钟的每个上升沿对 CAP 输入（由 CTCR 寄存器的 bit[3:2]选择）进行采样。在对这个 CAP 输入的连续两次采样值进行比较之后，可以识别出下面其中一种事件：上升沿、下降沿、上升/下降沿或所选 CAP 输入的电平不变。只有当发生所识别出的事件，并且该事件与 CTCR 寄存器中bit[1:0] 选择的一个事件相对应时，定时计数器寄存器的值才会加 1。

要有效地处理计数器的外部源时钟还有一些限制，因为需使用 PCLK 时钟的 2 个连续的上升沿才能确定 CAP 选择的输入上的一个边沿， 所以 CAP 输入的频率不能超过 PCLK 时钟的一半。因此，相同 CAP 输入上的高/低电平持续时间不

应少于 1/（2×PCLK）。

寄存器	偏移量	R/W	描述	复位值
CTCR0	**TIMER0_BA+0x180**	**R/W**	计数控制寄存器	**0x0000_0000**

31	30	29	28	27	26	25	24
保留							
23	22	21	20	19	18	17	16
保留							
15	14	13	12	11	10	9	8
保留							
7	6	5	4	3	2	1	0
MODE3		MODE2		MODE1		MODE0	

Bits	描述	
[31:8]	保留	保留
[7:6]	MODE3	**该字段选择TC3在何时递增** 00：定时器模式，TC3在PCLK上升沿递增 01：计数器模式，TC3在CAP输入的上升沿递增 10：计数器模式，TC3在CAP输入的下降沿递增 11：计数器模式，TC3在CAP输入的双沿递增
[5:4]	MODE2	**该字段选择TC2在何时递增** 00：定时器模式，TC2在PCLK上升沿递增 01：计数器模式，TC2在CAP输入的上升沿递增 10：计数器模式，TC2在CAP输入的下降沿递增 11：计数器模式，TC2在CAP输入的双沿递增
[3:2]	MODE1	**该字段选择TC1在何时递增** 00：定时器模式，TC1在PCLK上升沿递增

		01：计数器模式，TC1在CAP输入的上升沿递增 10：计数器模式，TC1在CAP输入的下降沿递增 11：计数器模式，TC1在CAP输入的双沿递增
[1:0]	MODE0	**该字段选择TC0在何时递增** 00：定时器模式，TC0在PCLK上升沿递增 01：计数器模式，TC0在CAP输入的上升沿递增 10：计数器模式，TC0在CAP输入的下降沿递增 11：计数器模式，TC0在CAP输入的双沿递增

6.10.6.15 PWMCR0 寄存器

使用该寄存器将匹配输出配置为 PWM 输出。可分别设置各匹配输出，以确定匹配输出是作为 PWM 输出还是作为功能受外部匹配寄存器（EMR）控制的匹配输出。还可以设置为 PWM 单脉冲输出，还是连续的 PWM 输出。

寄存器	偏移量	R/W	描述	复位值
PWMC0	**TIMER0_BA+0x190**	**R/W**	**PWM控制寄存器**	**0x0000_0000**

31	30	29	28	27	26	25	24
保留							
23	22	21	20	19	18	17	16
SING3_INT	SING2_INT	SING1_INT	SING0_INT	INV3	INV2	INV1	INV0
15	14	13	12	11	10	9	8
INA3HIZ	A3HIZ	INA2HIZ	A2HIZ	INA1HIZ	A1HIZ	INA0HIZ	A0HIZ
7	6	5	4	3	2	1	0
SING3	SING2	SING1	SING0	保留			

Bits	描述

[31:24]	保留	保留
[23]	SING3_INT	**PWM3 脉冲计数中断使能** 1: 当 PWM3 输出预设个数的 PWM 后产生中断 0：不产生中断
[22]	SING2_INT	**PWM2 脉冲计数中断使能** 1: 当 PWM2 输出预设个数的 PWM 后产生中断 0：不产生中断
[21]	SING1_INT	**PWM1 脉冲计数中断使能** 1: 当 PWM1 输出预设个数的 PWM 后产生中断 0：不产生中断
[20]	SING0_INT	**PWM0 脉冲计数中断使能** 1: 当 PWM0 输出预设个数的 PWM 后产生中断 0：不产生中断
[19]	INV3	**PWM3 输出电平取反** 1：由 bit[15:14]设置的 PWM3 的输出电平进行取反后再输出 0：PWM3 的输出电平由 bit[15:14]决定
[18]	INV2	**PWM2 输出电平取反** 1：由 bit[13:12]设置的 PWM2 的输出电平进行取反后再输出 0：PWM2 的输出电平由 bit[13:12]决定
[17]	INV1	**PWM1 输出电平取反** 1：由 bit[11:10]设置的 PWM1 的输出电平进行取反后在输出 0：PWM1 的输出电平由 bit[11:10]决定
[16]	INV0	**PWM0 输出电平取反** 1：由 bit[9:8]设置的 PWM0 的输出电平进行取反后再输出 0：PWM0 的输出电平由 bit[9:8]决定
[15]	INA3HIZ	**PWM3 无效状态高阻使能位** 1：PWM3无效状态时输出高阻 0：PWM3无效状态时输出低电平

[14]	A3HIZ	**PWM3 有效状态高阻使能位** 1：PWM3有效状态时输出高阻 0：PWM3 有效状态时输出高电平
[13]	INA2HIZ	**PWM2 无效状态高阻使能位** 1：PWM2无效状态时输出高阻 0：PWM2无效状态时输出低电平
[12]	A2HIZ	**PWM2 有效状态高阻使能位** 1：PWM2有效状态时输出高阻 0：PWM2 有效状态时输出高电平
[11]	INA1HIZ	**PWM1 无效状态高阻使能位** 1：PWM1无效状态时输出高阻 0：PWM1无效状态时输出低电平
[10]	A1HIZ	**PWM1 有效状态高阻使能位** 1：PWM1有效状态时输出高阻 0：PWM1 有效状态时输出高电平
[9]	INA0HIZ	**PWM0 无效状态高阻使能位** 1：PWM0无效状态时输出高阻 0：PWM0无效状态时输出低电平
[8]	A0HIZ	**PWM0 有效状态高阻使能位** 1：PWM0有效状态时输出高阻 0：PWM0 有效状态时输出高电平
[7]	SING3	**PWM3 脉冲计数使能** 1：使能 MAT3 的 PWM 脉冲计数输出功能 0：禁止脉冲计数输出功能
[6]	SING2	**PWM2 脉冲计数使能** 1：使能 MAT2 的 PWM 脉冲计数输出功能 0：禁止脉冲计数输出功能
[5]	SING1	**PWM1 脉冲计数使能** 1：使能 MAT1 的 PWM 脉冲计数输出功能

		0：禁止脉冲计数输出功能
[4]	SING0	PWM0 脉冲计数使能 1：使能 MAT0 的 PWM 脉冲计数输出功能 0：禁止脉冲计数输出功能
[3：0]	保留	保留位

6.10.6.16 PWMV0/1/2/3 寄存器

在 PWM0/1/2/3 脉冲计数输出功能使能时用来设置要输出的脉冲个数。

寄存器	偏移量	R/W	描述	复位值
PWMV0/1/2/3	**TIMER0_BA+0x1A0** **TIMER0_BA+0x1B0** **TIMER0_BA+0x1C0** **TIMER0_BA+0x1D0**	**R/W**	**PWM脉冲输出个数寄存器**	**0x0000_0000**

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

Bits	描述	
[15: 0]	VALUE	PWM 脉冲输出个数 个数=VALUE+1

6.10.6.17 PERIOD0_HIGH 寄存器

当使能通道 0 的输入捕获硬件自动测量功能时，该寄存器保存输入信号的高电平周期数。

寄存器	偏移量	R/W	描述	复位值
PERIOD0_HIGH	TIMER0_BA+0x220	R	通道0高电平周期数寄存器	0x0000_0000

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

Bits	描述	
[31: 0]	VALUE	当使能通道 0 的硬件自动测量功能时，会自动用系统时钟来计算输入通道 0 上信号的高电平周期数。 与对应的 PERIOD0_LOW 寄存器配合即可计算出输入信号的周期以及占空比。

6.10.6.18 PERIOD0_LOW 寄存器

当使能通道 0 的输入捕获硬件自动测量功能时，该寄存器保存输入信号的低电平周期数。

寄存器	偏移量	R/W	描述	复位值

PERIOD0_LOW	TIMER0_BA+0x230	R	通道0低电平周期数寄存器	0x0000_0000

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

Bits	描述	
[31: 0]	VALUE	当使能通道 0 的硬件自动测量功能时，会自动用系统时钟来计算输入通道 0 上信号的低电平周期数。 与对应的 PERIOD0_HIGH 寄存器配合即可计算出输入信号的周期以及占空比。

6.10.6.19 PERIOD1_HIGH 寄存器

当使能通道 1 的输入捕获硬件自动测量功能时，该寄存器保存输入信号的高电平周期数。

寄存器	偏移量	R/W	描述	复位值
PERIOD1_HIGH	TIMER0_BA+0x240	R	通道1高电平周期数寄存器	0x0000_0000

31	30	29	28	27	26	25	24
VALUE[31:24]							
23	22	21	20	19	18	17	16

VALUE[23:16]							
15	14	13	12	11	10	9	8
VALUE[15:8]							
7	6	5	4	3	2	1	0
VALUE[7:0]							

Bits	描述	
[31: 0]	VALUE	当使能通道 1 的硬件自动测量功能时，会自动用系统时钟来计算输入通道 1 上信号的高电平周期数。 与对应的 PERIOD1_LOW 寄存器配合即可计算出输入信号的周期以及占空比。

6.10.6.20 PERIOD1_LOW 寄存器

当使能通道 1 的输入捕获硬件自动测量功能时，该寄存器保存输入信号的低电平周期数。

寄存器	偏移量	R/W	描述	复位值
PERIOD1_LOW	TIMER0_BA+0x250	R	通道1低电平周期数寄存器	0x0000_0000

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

Bits	描述	
[31: 0]	VALUE	当使能通道 1 的硬件自动测量功能时，会自动用系统时钟来计算输入通道 1 上信号的低电平周期数。 与对应的 PERIOD1_HIGH 寄存器配合即可计算出输入信号的周期以及占空比。

6.10.6.21 PERIOD2_HIGH 寄存器

当使能通道 2 的输入捕获硬件自动测量功能时，该寄存器保存输入信号的高电平周期数。

寄存器	偏移量	R/W	描述	复位值
PERIOD2_HIGH	**TIMER0_BA+0x260**	**R**	**通道2高电平周期数寄存器**	**0x0000_0000**

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

Bits	描述	
[31: 0]	VALUE	当使能通道 2 的硬件自动测量功能时，会自动用系统时钟来计算输入通道 2 上信号的高电平周期数。 与对应的 PERIOD2_LOW 寄存器配合即可计算出输入信号的周期以及占空比。

6.10.6.22 PERIOD2_LOW 寄存器

当使能通道 2 的输入捕获硬件自动测量功能时，该寄存器保存输入信号的低电平周期数。

寄存器	偏移量	R/W	描述	复位值
PERIOD2_LOW	TIMER0_BA+0x270	R	通道2低电平周期数寄存器	0x0000_0000

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

Bits	描述	
[31: 0]	VALUE	当使能通道 2 的硬件自动测量功能时，会自动用系统时钟来计算输入通道 2 上信号的低电平周期数。 与对应的 PERIOD2_HIGH 寄存器配合即可计算出输入信号的周期以及占空比。

6.10.6.23 PERIOD3_HIGH 寄存器

当使能通道 3 的输入捕获硬件自动测量功能时，该寄存器保存输入信号的高电平周期数。

寄存器	偏移量	R/W	描述	复位值
PERIOD3_HIGH	TIMER0_BA+0x280	R	通道3高电平周期数寄存器	0x0000_0000

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

Bits	描述	
[31: 0]	VALUE	当使能通道 3 的硬件自动测量功能时，会自动用系统时钟来计算输入通道 3 上信号的高电平周期数。 与对应的 PERIOD3_LOW 寄存器配合即可计算出输入信号的周期以及占空比。

6.10.6.24 PERIOD3_LOW 寄存器

当使能通道 3 的输入捕获硬件自动测量功能时，该寄存器保存输入信号的低电平周期数。

寄存器	偏移量	R/W	描述	复位值
PERIOD3_LOW	TIMER0_BA+0x290	R	通道3低电平周期数寄存器	0x0000_0000

31	30	29	28	27	26	25	24
VALUE[31:24]							
23	22	21	20	19	18	17	16
VALUE[23:16]							
15	14	13	12	11	10	9	8

VALUE[15:8]							
7	6	5	4	3	2	1	0
VALUE[7:0]							

Bits	描述	
[31: 0]	VALUE	当使能通道 3 的硬件自动测量功能时，会自动用系统时钟来计算输入通道 3 上信号的低电平周期数。 与对应的 PERIOD3_HIGH 寄存器配合即可计算出输入信号的周期以及占空比。

6.11 MotorPWM 控制器(MCPWM)

6.11.1 概述

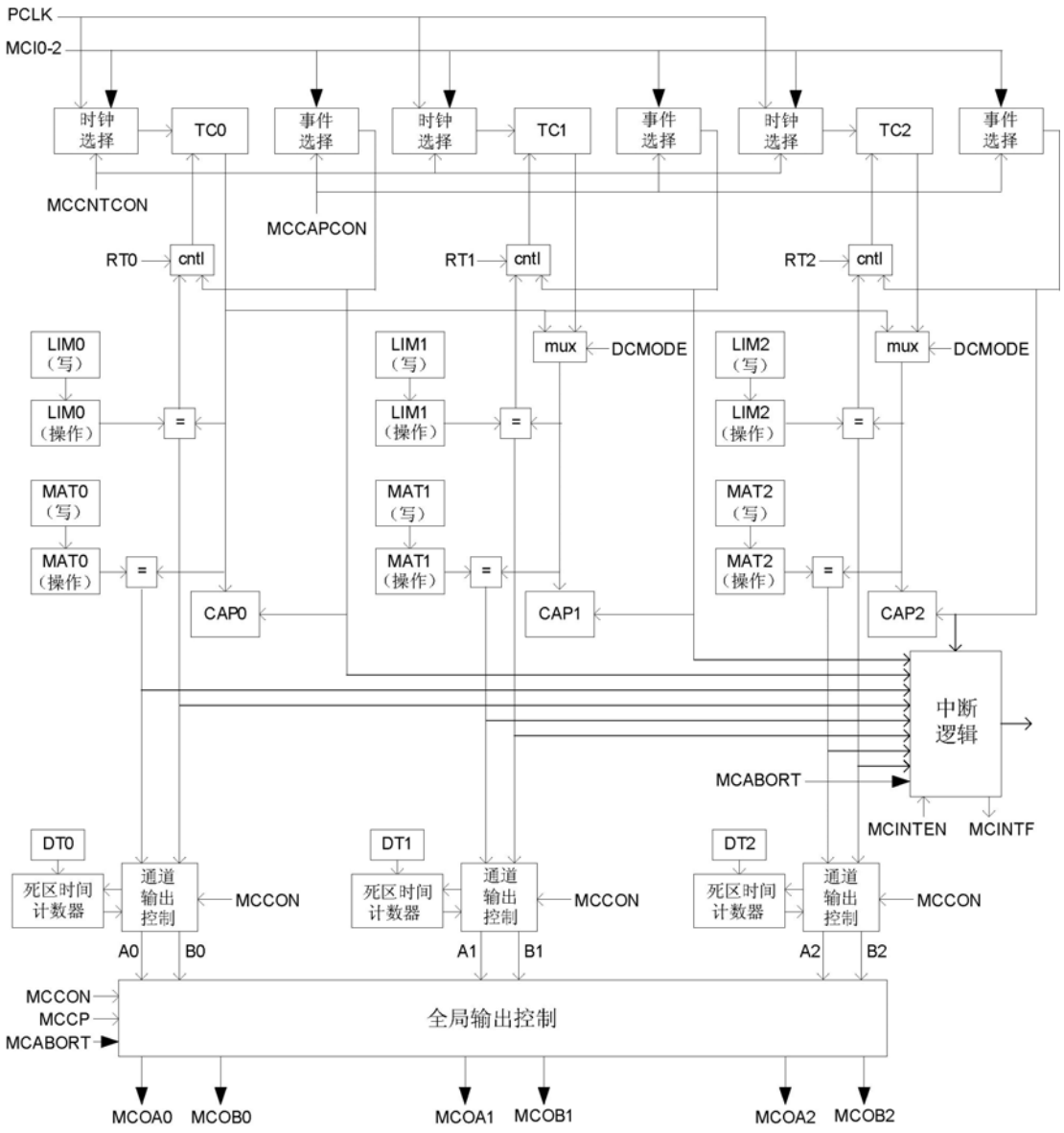
电机控制PWM（MCPWM）非常适用于三相交流AC和直流DC电机控制应用，但它还可以用于其它需要通用定时、捕获和比较的应用中。

6.11.2 特性

MCPWM 含有 3 个独立的通道，每个通道包括：

- 1 个 16 位定时器/计数器(TC)
- 1 个 16 位界限寄存器(LIM)
- 1 个 16 位匹配寄存器(MAT)
- 1 个 10 位死区时间寄存器(DT)和相应的 10 位死区时间计数器
- 1 个 16 位捕获寄存器
- 2 个极性相反的已调整的输出(MCOA 和 MCOB)
- 1 个周期中断，1 个脉宽中断和 1 个捕获中断
- 硬件触发 ADC 功能，触发时间点可配置
- 转速自动测量功能
- HALL 控制信号自动更新功能，更新时间点可配置
- 硬件移相功能

6.11.3 MCPWM 结构框图



图： MCPWM 模块结构框图

6.11.4 基本配置

MCPWM 模块有 3 个独立通道，每个通道有 2 个输出管脚和 1 个输入管脚，分别为 MCOAx，MCOBx 和 MCIx(x=0，1，2，分别表示通道 0,1,2)。此外还有

一个全局 ABORT 输入管脚，用于快速中止所有的输出。这些管脚分布在不同的 PIO 上，参见 5.3 节管脚描述一章，然后在 GPIO 模块中将相应的管脚功能配置成 MCPWM 功能。

MCPWM 的模块时钟为 PCLK，分别通过 AHBCLKCTRL 寄存器的 bit[12]来使能。

6.11.5 基本操作

6.11.5.1 脉宽调制

MCPWM的每个通道都具有两个输出A和B，它们可以驱动一对晶体管来切换两个电源导轨之间的一个已控制的点。大多数情况下两个输出极性相反，但可以使能死区时间特性（以每个通道为基础）来延时信号从“无效”到“有效”状态的跳变，这样晶体管永远都不会同时导通。更普遍的说法是，每个输出对的状态可认为是“high”、“low”、“floating”或“up”、“down”和“center-off”。

每个通道的映射从“有效”和“无效”到0V和3.3V(5V)是可编程的。

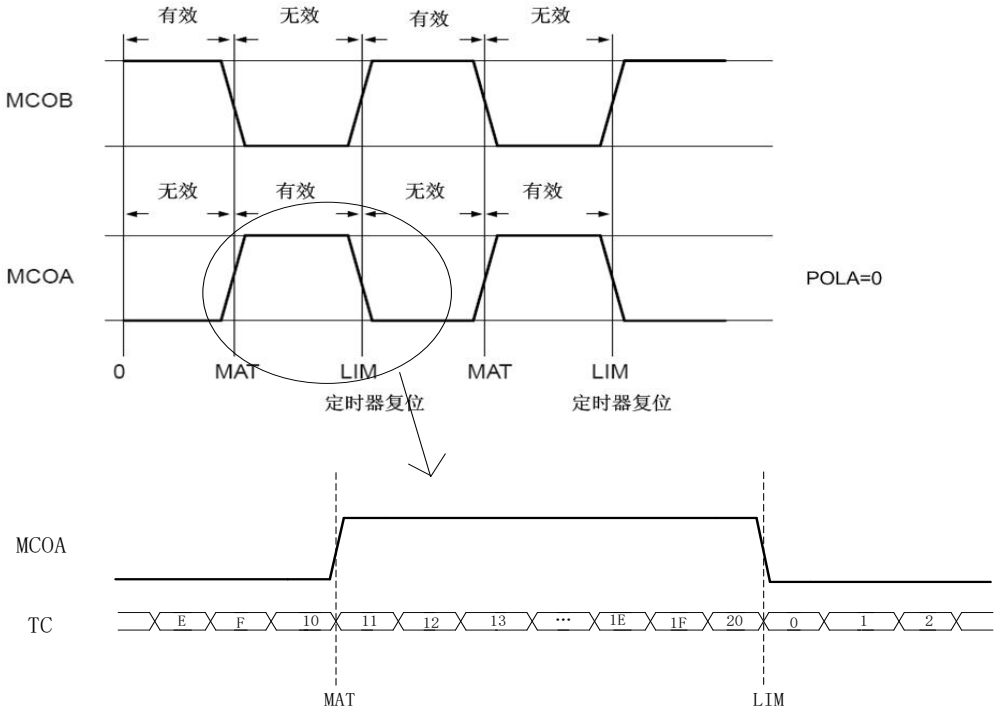
复位后，MCPWM模块的3个A和B的输出都为低电平。同时还可以通过CON寄存器的AxLVL和BxLVL位来手动设置A和B的输出电平。

MCPWM可执行边沿对齐和中心对齐的脉宽调制。

注：在定时器模式下，通道的已调制MCO输出的周期由它的界限寄存器决定，周期起始处的脉宽由它的匹配寄存器决定。你可以将界限寄存器看作是“周期寄存器”，把匹配寄存器看作是“脉宽寄存器”。

- *不带死区时间的边沿对齐 PWM*

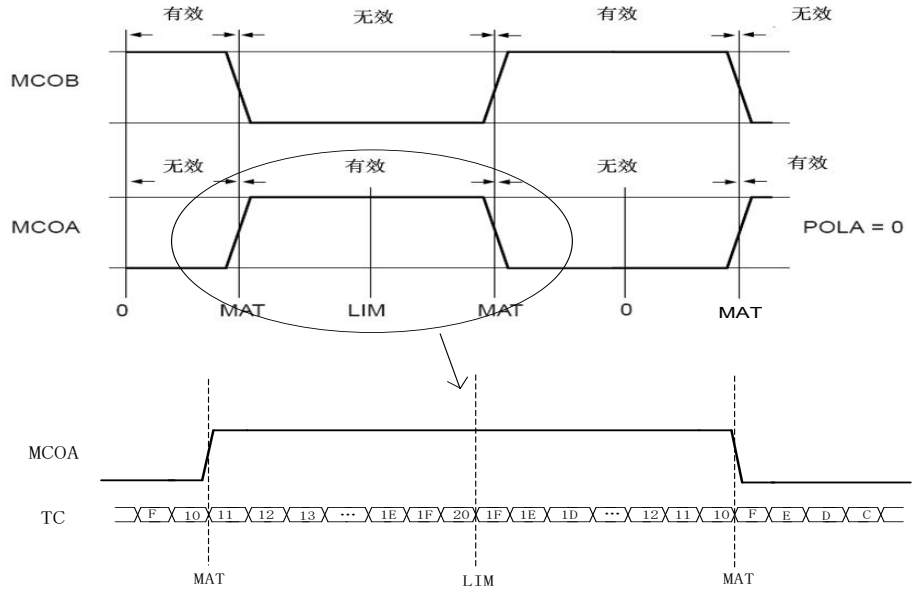
如下图所示，在该模式中定时器TC从0开始递增计数直到LIM寄存器中的值。在TC与匹配寄存器相匹配之前，MCO一直保持无效状态，匹配时，MCO状态变为有效。当TC与界限寄存器匹配时，MCO变回无效状态，TC复位且再次开始递增计数。



图：不带死区时间的边沿对齐 PWM 波形图(MAT=10,LIM=20,POLA=0)

- *不带死区时间的中心对齐 PWM*

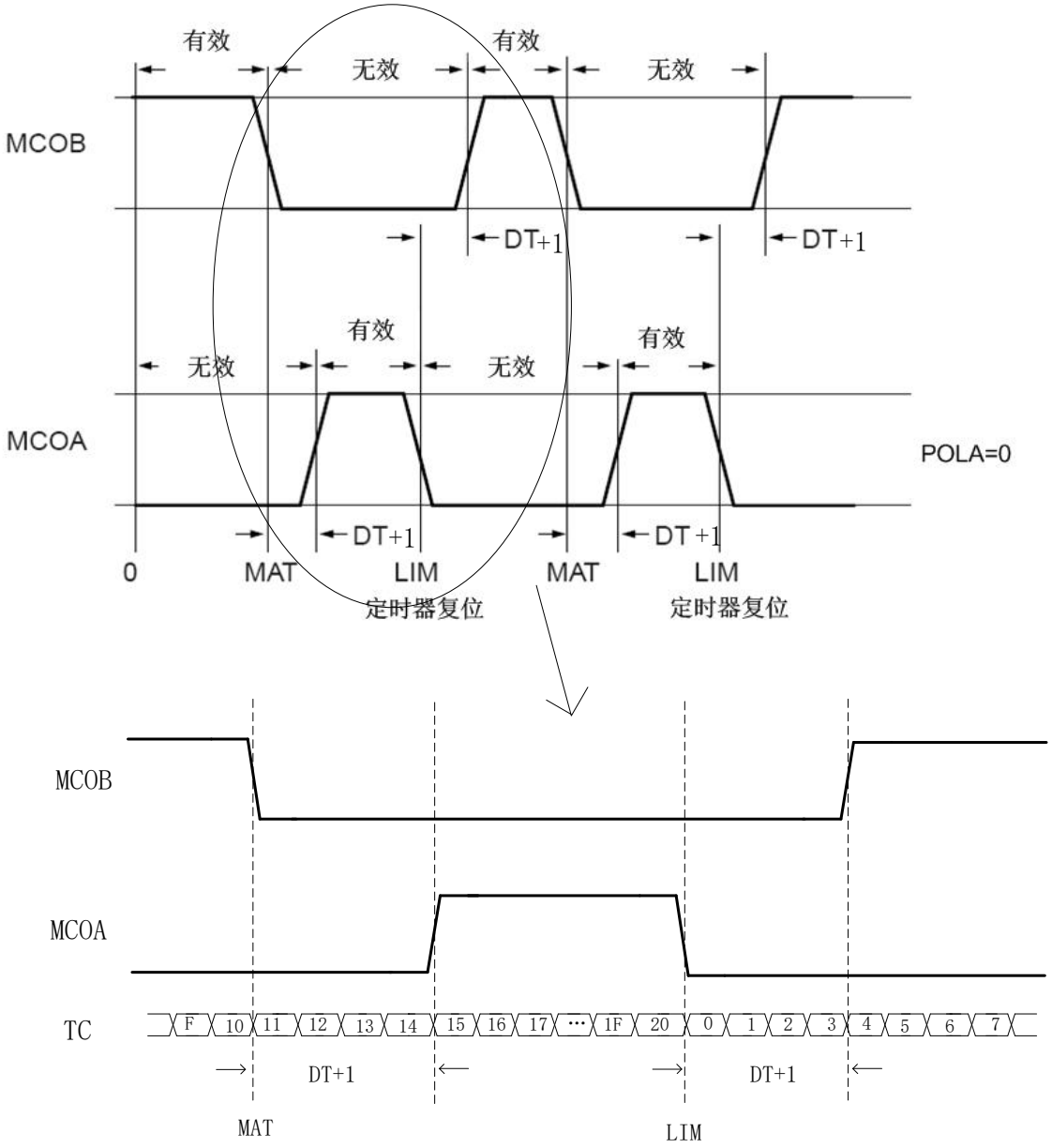
如下图所示，在该模式中定时器 TC 从 0 开始递增计数直到 LIM 寄存器中的值，然后递减计数到 0 并重复操作。当定时器递增计数时，MCO 为无效状态直至 TC 与匹配寄存器相匹配，此时 MCO 状态变为有效。当 TC 与界限寄存器匹配时，它开始递减计数。当 TC 与匹配寄存器在向下过程中匹配时，MCO 状态变为无效。



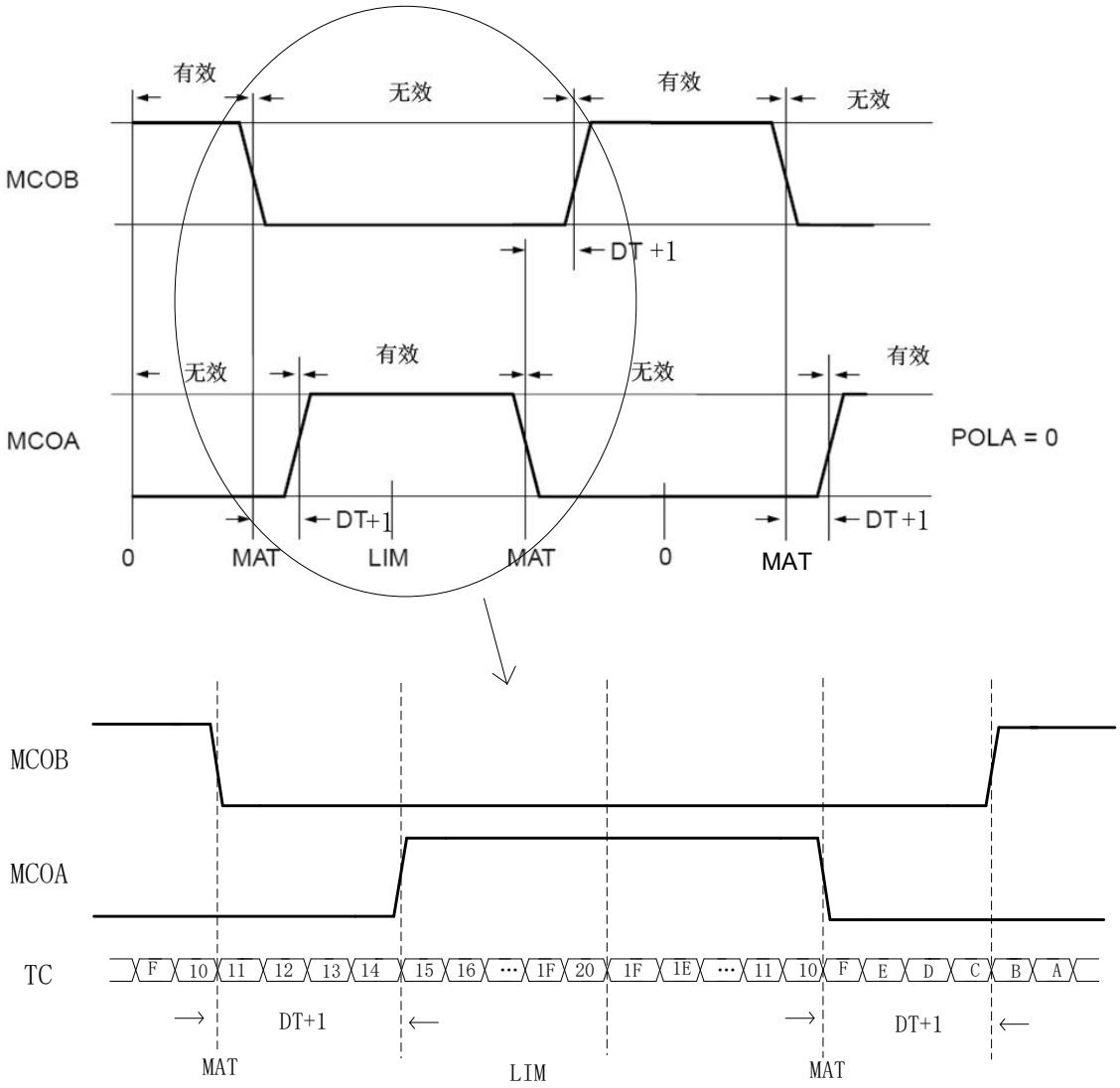
图：不带死区时间的中心对齐 PWM 的波形(MAT=10,LIM=20,POLA=0)

- **死区时间计数器**

当通道的DTE位在MCCON中置位时，死区时间计数器延时两个MCO输出的“无效到有效状态”的跳变。只要通道的A或B输出从有效状态变为无效状态，死区时间计数器就开始递增计数，从0到DT值（在MCDT寄存器）。其它输出从无效状态到有效状态的跳变被延时直至死区时间计数器到达DT值。在死区时间内，MCOA和MCOB输出电平都无效。下图1所示为带死区时间的边沿对齐模式的操作，图2所示为带死区时间的中心对齐模式的操作。



图：带死区时间的边沿对齐 PWM 的波形(MAT=10,LIM=20,DT=3,POLA=0)



图： 带死区时间的中心对齐 PWM 的波形(MAT=10,LIM=20,DT=3,POLA=0)

6.11.5.2 映射寄存器和同时更新

界限、匹配寄存器和通信格式寄存器（MCLIM、MCMAT、MCCP）作为寄存器对来执行，每个寄存器都包含一个写寄存器和一个操作寄存器。软件可写入写寄存器。操作寄存器控制每个通道的实际操作，并且在TC从0开始递增计数时将写寄存器的当前值载入到操作寄存器中。

置位MCCON寄存器的DISUP位可禁止功能寄存器的更新。如果DISUP位置位，

那么功能寄存器不更新直至软件停止通道。

如果通道在软件写入其LIM或MAT寄存器时不运行，那么功能寄存器立即更新。仅当通道被停止时，软件才能写TC寄存器。

MCPWM有一个外部输入引脚MCABORT。当该输入变低时，6个MCO输出均表现为无效状态，且在中断使能的情况下产生Abort中断。输出在无效状态下保持锁定直至ABORT中断标志被清除或Abort中断被禁止。ABORT标志在MCABORT输入变为高电平之前可能不会被清除。

6.11.5.3 捕获事件

当输入信号发生跳变时，每个PWM通道可捕获TC的瞬时值。在MCCAPCON寄存器的控制下，任意通道都可在任意或所有MCPWM输入0-2的上升沿和/或下降沿上产生捕获事件。输入上升沿或下降沿的检测与PCLK同步。

如果通道MCCAPCON寄存器的HNP位被置位以使能“噪声滤波”，那么MCI管脚上选择的边沿将启动该通道的死区时间计数器，以下描述的捕获事件操作将被延时死区时间。这种功能特别适用于执行具有霍尔（Hall）传感器的三相无刷DC电机控制。

通道上的捕获事件（可能通过 HNF 延时）会引起下列操作：

- TC 的当前值保存到捕获寄存器中(CAP)
- 如果通道的捕获事件中断被使能，那么设置捕获事件中断标志
- 如果通道的 RT 位在 MCCAPCON 寄存器中被置位，使能捕获事件上的复位，那么输入事件等效于通道的 TC 与其 LIM 寄存器相匹配。这包括复位 TC 以及在边沿对齐模式下切换 MCO 管脚，请参见小节“MCPWM 界限值寄存器 0-2”和“脉宽调制”中的描述

6.11.5.4 外部事件计数

若通道MODE位在MCCNTCON中置位，那么通道的TC将在MCO-2输入的上升沿和/或下降沿（同时被检测的）上加1。PWM的功能和捕获功能都不会受影响。

6.11.5.5 三相 DC 模式

三相DC模式可通过置位MCCON寄存器中的DCMODE位来选择。

在该模式下，内部MCOA0信号可以被连接到任意或全部的MCO输出。每个MCO输出可通过当前通信格式寄存器MCCP中的位来屏蔽。如果MCCP寄存器中的位为0，那么它的输出管脚具有输出MCOA0的无效状态的逻辑电平。断开状态的极性由POLA0位来决定。

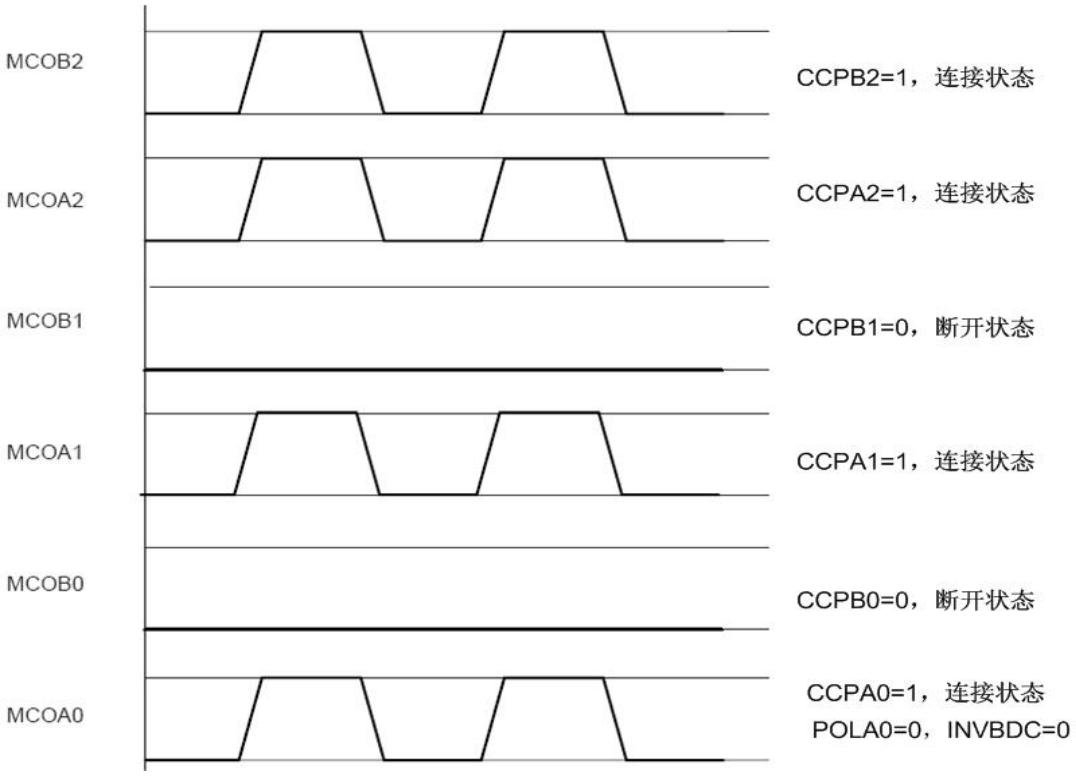
在MCCP寄存器中所有MCO输出含有1的位由内部MCOA0信号来控制。

当MCCON寄存器中的INVBDC位为1时，3个MCOB输出管脚被反相。这种特性可用来调节桥驱动器（桥驱动器的低端开关为低电平有效输入）。

MCCP寄存器作为一对映射寄存器来操作，因此有效通信模式的变化在新PWM周期的起始处出现。有关写和读这些寄存器的内容请参见章节“界限寄存器”和“映射寄存器和同时更新”。

下图所示为三相DC模式中MCO的示例波形。MCCP寄存器（对应于输出MCOB1和MCOB0）中的位1和位3为0，所以这些输出被屏蔽，处于断开状态。它们的逻辑电平由POLA0位来决定（这里，POLA0=0使得无效状态为逻辑低电平）。INVBDC位被设为0（逻辑电平没有翻转），所以B输出和A输出的极性相同。注意该模式与其它模式不同，因为其它模式的MCOB输出不是MCOA输出的反相。

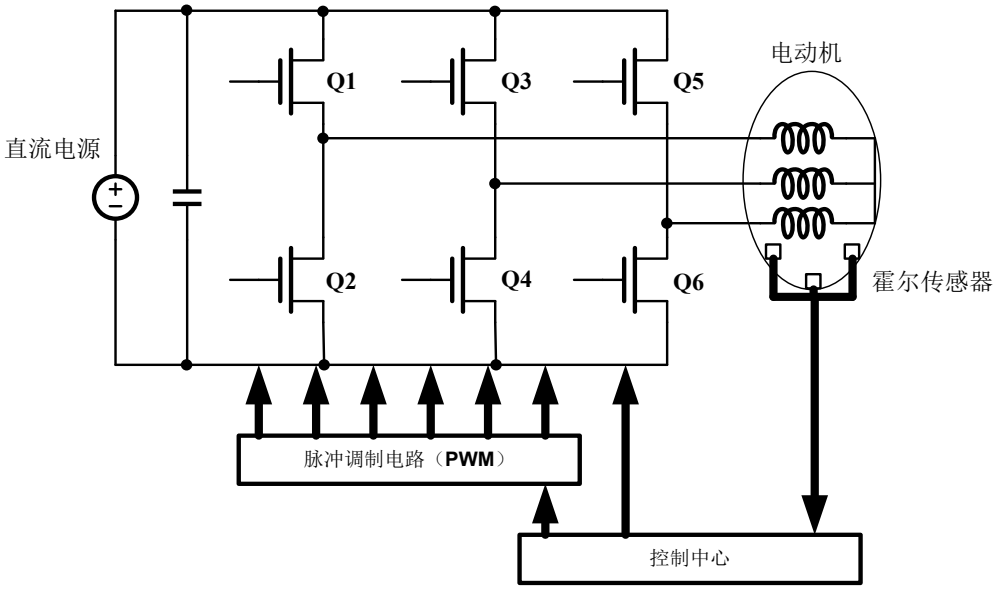
如下图所示，MCCP寄存器中的位0、2、4和5被设为1，表示MCOA1和通道2的MCOA2输出跟随MCOA0信号。



图： MCPWM 三相 DC 模式的实例波形

6.11.5.6 带霍尔传感器的三相无刷直流电机模式

无刷电动机内部在定子适当位置放置三个霍尔位置传感器，用以检测转子磁钢的位置。将位置信号转换成电信号，产生六个状态编码，控制逆变桥各功率管通断。使电动机定子绕组中的电流随着转子位置的变化按次序换相。如下图所示。当电机转动起来，控制中心根据驱动器设定的速度及加速率命令与霍尔信号变化的速度加以对比，从而决定下一组定子绕组供电电路的通断，以及通断时间长短。



图： 带霍尔传感器的三相直流电机工作原理图

在霍尔传感器模式中，MCPWM通过输入接口MCI0，MCI1，MCI2自动采样霍尔信号，为滤除霍尔信号中的干扰，利用死区时间计数器DT0实现滤波功能（HNFCAP0），任何霍尔信号的跳变将自动重载DT0并开始从0向上计数，当DT0计数到设定值时，进行霍尔信号的采集操作。从而在霍尔信号变化到采样点之间产生了一个延时，减少干扰对换相的影响。

当滤波延时完成后，会自动进行霍尔信号的采样和比较操作，如果霍尔输入值等于期望值(EXPH)，则表明发生正确的霍尔换相事件，可触发正确霍尔换相中断。如果输入霍尔信号既不等于期望值，也不等于当前值（CURH）则表明霍尔信号发生错误(如霍尔信号线断开)，同时可触发错误霍尔事件中断，通知CPU进行相应处理。

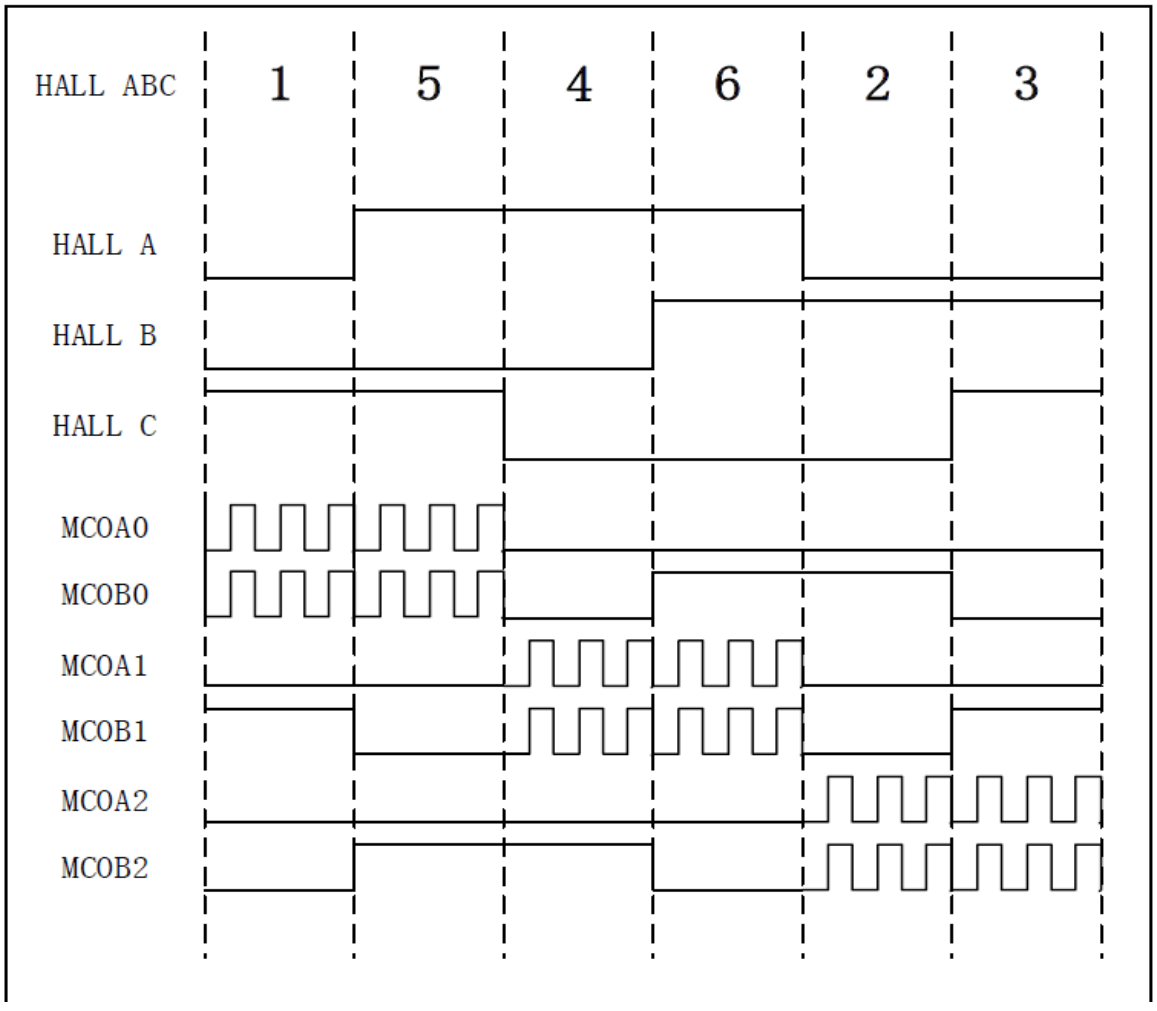
正确霍尔换相事件可以触发换相操作，自动更新霍尔映射寄存器(HALLS)。MCP用于调制6路PWM (A0 B0 A1 B1 A2 B2)，当MCPx = 1时，对应的PWM通道输出PWM，当MCPx = 0时，对应的PWM输出无效电平(由PSLx决定无效电平值)，因此对HALLS的更新即可以实现对6路PWM的调制操作。

为避免软件更新带来的换相延迟，添加映射机制，使用HALLS 映射寄存器。当正确换相事件发生时，HALL中的内容会自动更新到映射寄存器HALLS中，从而

自动改变6路PWM的输出状态。用户程序仅需在下次换相前更新寄存器MCCP和EXPH即可。 为增加灵活性，HALLS寄存器也可由软件直接配置。

在DC模式下定时器0用于产生所有输出的PWM信号，PWM与MCPx(x = 0～5)进行耦合，输出到PWM端口，产生所需的PWM。

在霍尔传感器模式中，可以自动进行霍尔信号的滤波、采样、比较、换相操作，与传统的软件扫描相比，换相准确，换相误差小，尤其对于高速直流无刷电机等对换相误差要求高的应用领域，其优势更加明显。



图： MCPWM 在 DC 模式下的输入输出状态波形图

霍尔模式DC操作流程：

1、channel0计时器模式使能，由于输入信号被用作霍尔信号，所以该模式下只能通过计时的模式产生PWM，其他通道的pwm输出都与通道0的相同。

2、输入边沿触发使能，CAPCON(0x3F)三个输入信号的所有上下边沿都作为cap0_enable事件的触发条件。

3、设置PWM参数以及死区时间。PWM占空比的大小来调节电流的大小从而调节电机转速的快慢。

4、使能DC模式，使能HALL模式，如果需要测速功能可使能测速模式。

5、在启动电机之前，通过识别霍尔信号先判断当前电机所处的位置，当前霍尔状态。根据当前状态设置EXPH和CURH状态值，配置MCP和PSL的值，决定输出状态。

6、启动电机。根据捕获的cap0_enable事件，改变EXPH的值，以及MCP和PSL的值。如上图，该电机每个霍尔传感器相隔120°，每个霍尔状态发生改变时，输出状态也跟随改变。如第一个状态，A0B0输出PWM格式，且A0B0反向并含有死区时间。A1B1A2B2为无效状态，但B1的无效值为1，其他的为0.该状态下相当于导通A0二极管和电机还有B1二极管这条通路。

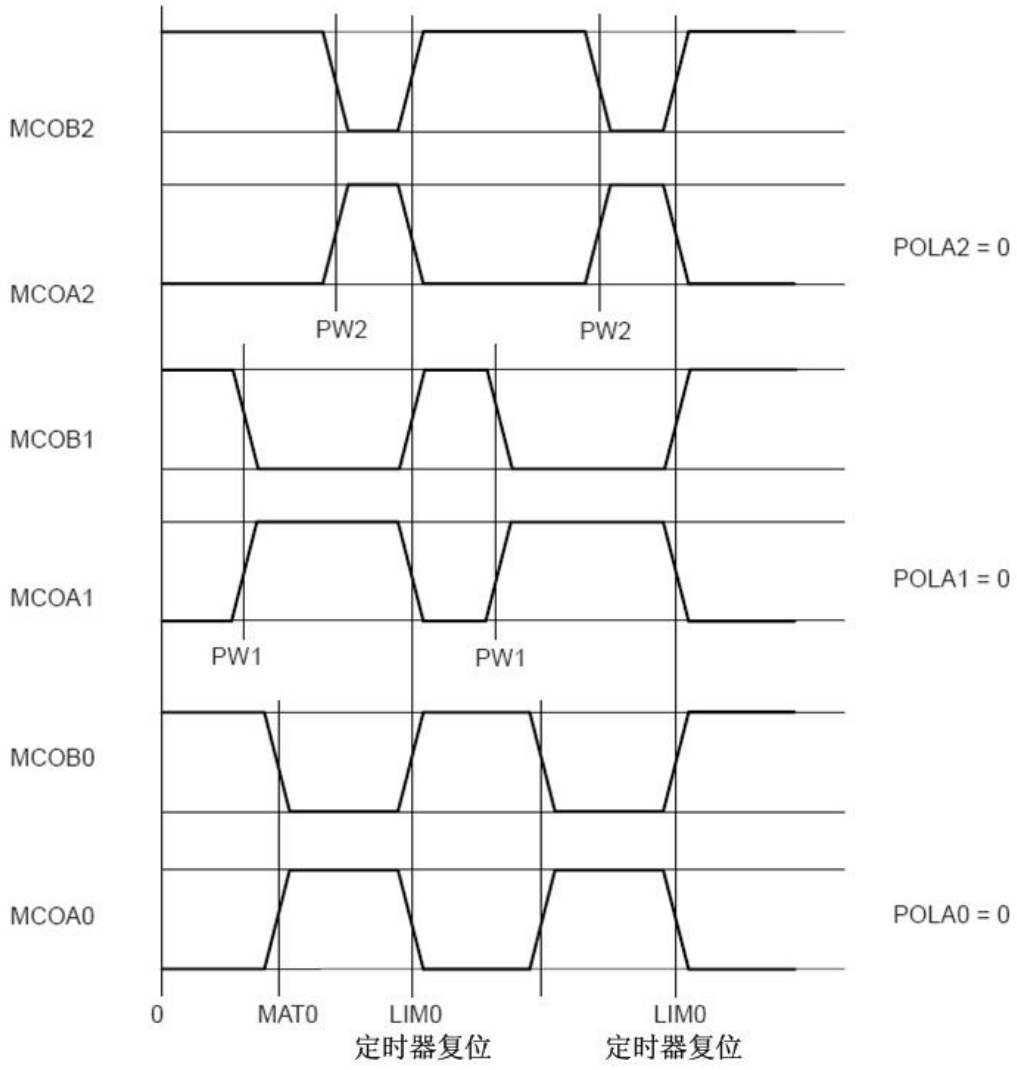
6.11.5.7 三相 AC 模式

三相AC模式可通过置位MCCON寄存器中的ACMODE位来选择。

在该模式下，通道0的TC值可用于与所有通道的MAT寄存器进行比较（不使用LIM1-2寄存器）。

每个通道通过比较它的MAT值与TC0来控制其MCO输出。

下图所示为三相AC模式下6个MCO输出的示例波形。POLA位设为0用于所有的3个通道，因此对于所有的MCO输出来说有效状态下的电平为高电平，无效状态下的电平为低电平。每个通道具有不同的LIM值，可以与MCTC0值进行比较。在这种模式下，周期值指定用于所有的3个通道，并且由MCLIM0来决定。死区时间模式被禁能。



图：MCPWM 在三相 AC 模式下的实例波形图

6.11.5.8 移相模式

移相模式可通过置位MCCON寄存器中的CHGPH位来选择。如果选择这种模式则只需要使能RUN0，RUN1和RUN2要保持为0，HALLS_DIS位也要使能。

在该模式下，通道0做为参考通道来产生要输出的波形，1和2通道管脚上波形的周期、占空比、死区时间与0通道完全相同，只是在相位上会有一定的偏移。同时在MCCON寄存器中关于CH0,1,2的CENTERx和DTEx位配置必须完全相同。

具体的偏移量以(CHx_CHGPH+1)x(PR+1)个APBCLK周期数来计算，CHx_CHGPH通过CHGPH寄存器来设定。

6.11.5.9 中断

MCPWM 有 23 个可能的中断源：

- 当任意通道的 TC 与它的匹配寄存器匹配时
- 当任意通道的 TC 与它的界限寄存器匹配时
- 当任意通道捕获其 TC 值并保存到捕获寄存器时
- 当任意通道处于中心对齐模式时,其 TC 值下降到 0,下降到 MAT 值或者上升到 MAT 时
- 当所有3个通道的输出被强制为“无效”状态时（因为ABORT管脚变为低电平）。MCABORT“中断寄存器”小节解释了怎样使能这些中断，“捕获控制寄存器”小节介绍了怎样映射MCI0-2输入的边沿到3个通道的“捕获事件”
- 4个与带霍尔传感器无刷直流电机有关的中断

6.11.6 寄存器映射

R：只读，W：只写，R/W：可读写

寄存器	偏移量	R/W	描述	复位后的值
MCPWM 的基地址： **MCPWM_BA = 0x4000_7000**				
CON	MCPWM_BA+0x000	R	控制读寄存器	0x0000_0000
CON_SET	MCPWM_BA+0x004	W	控制置位寄存器	0x0000_0000
CON_CLR	MCPWM_BA+0x008	W	控制清除寄存器	0x0000_0000
CAPCON	MCPWM_BA+0x00C	R	捕获控制读寄存器	0x0000_0000
CAPCON_SET	MCPWM_BA+0x010	W	捕获控制置位寄存器	0x0000_0000
CAPCON_CLR	MCPWM_BA+0x014	W	获控制清除寄存器	0x0000_0000
TC0	MCPWM_BA+0x018	R/W	通道0计数寄存器	0x0000_0000
TC1	MCPWM_BA+0x01C	R/W	通道1计数寄存器	0x0000_0000

TC2	MCPWM_BA+0x020	R/W	通道2计数寄存器	0x0000_0000
LIM0	MCPWM_BA+0x024	R/W	通道0界限寄存器	0xffff_ffff
LIM1	MCPWM_BA+0x028	R/W	通道1界限寄存器	0xffff_ffff
LIM2	MCPWM_BA+0x02C	R/W	通道2界限寄存器	0xffff_ffff
MAT0	MCPWM_BA+0x030	R/W	通道0匹配寄存器	0xffff_ffff
MAT1	MCPWM_BA+0x034	R/W	通道1匹配寄存器	0xffff_ffff
MAT2	MCPWM_BA+0x038	R/W	通道2匹配寄存器	0xffff_ffff
DT	MCPWM_BA+0x03C	R/W	死区时间寄存器	0x3fff_ffff
CP	MCPWM_BA+0x040	R/W	通信格式寄存器	0x0000_0000
CAP0	MCPWM_BA+0x044	R/W	通道0捕获寄存器	0x0000_0000
CAP1	MCPWM_BA+0x048	R/W	通道1捕获寄存器	0x0000_0000
CAP2	MCPWM_BA+0x04C	R/W	通道2捕获寄存器	0x0000_0000
INTEN	MCPWM_BA+0x050	R	中断使能读寄存器	0x0000_0000
INTEN_SET	MCPWM_BA+0x054	W	中断使能置位寄存器	0x0000_0000
INTEN_CLR	MCPWM_BA+0x058	W	中断使能清除寄存器	0x0000_0000
CNTCON	MCPWM_BA+0x05C	R	计数控制读寄存器	0x0000_0000
CNTCON_SET	MCPWM_BA+0x060	W	计数控制置位寄存器	0x0000_0000
CNTCON_CLR	MCPWM_BA+0x064	W	计数控制清除寄存器	0x0000_0000
INTF	MCPWM_BA+0x068	R	中断标志读寄存器	0x0000_0000
INTF_SET	MCPWM_BA+0x06C	W	中断标志置位寄存器	0x0000_0000
INTF_CLR	MCPWM_BA+0x070	W	中断标志清除寄存器	0x0000_0000
CAP_CLR	MCPWM_BA+0x074	W	通道捕获清除寄存器	0x0000_0000
HALL	MCPWM_BA+0x078	R/W	霍尔寄存器	0x0000_0000
HALLS	MCPWM_BA+0x07C	R	霍尔映射寄存器	0x0000_0000
VELCMP	MCPWM_BA+0x080	R/W	电机速度比较寄存器	0x0000_0000
VELVAL	MCPWM_BA+0x084	R	电机速度当前值寄存器	0x0000_0000
VELTH	MCPWM_BA+0x088	R/W	电机速度阈值寄存器	0x0000_0000

MCIST	MCPWM_BA+0x08C	R	输入信号状态寄存器	0x0000_0000
PR	MCPWM_BA+0x090	R/W	时钟预分频寄存器	0x0000_0000
CHGPH	MCPWM_BA+0x094	R/W	移相相位寄存器	0x0000_0000
HALLAEN	MCPWM_BA+0x098	R/W	HALL更新使能寄存器	0x0000_0000
HALLA	MCPWM_BA+0x09C	R/W	HALL更新值寄存器	0x0000_0000
ADC_TRIG0	MCPWM_BA+0x0A0	R/W	通道0触发ADC寄存器	0x0000_0000
ADC_TRIG1	MCPWM_BA+0x0A4	R/W	通道1触发ADC寄存器	0x0000_0000
ADC_TRIG2	MCPWM_BA+0x0A8	R/W	通道2触发ADC寄存器	0x0000_0000

6.11.6.1 MCCON 寄存器

CON 是只读寄存器。CON 寄存器控制着所有 PWM 通道的操作。该寄存器中的位通过写 1 到寄存器 CON_SET 和 CON_CLR 的相应位来置位和清零。

寄存器	偏移量	R/W	描述	复位值
CON	**MCPWM_BA+0x000**	**R**	**控制读寄存器**	**0x0000_0000**
CON_SET	**MCPWM_BA+0x004**	**W**	**控制置位寄存器**	**0x0000_0000**
CON_CLR	**MCPWM_BA+0x008**	**W**	**控制清除寄存器**	**0x0000_0000**

31	30	29	28	27	26	25	24
DCMODE	ACMODE	INVBDC	CHGPH	HALLS_DIS	保留	VEL	HALL
23	22	21	20	19	18	17	16
B2LVL	A2LVL	CH2EN	DISUP2	DTE2	POLA2	CENTER2	RUN2
15	14	13	12	11	10	9	8
B1LVL	A1LVL	CH1EN	DISUP1	DTE1	POLA1	CENTER1	RUN1
7	6	5	4	3	2	1	0
B0LVL	A0LVL	CH0EN	DISUP0	DTE0	POLA0	CENTER0	RUN0

Bits	描述	
[31]	DCMODE	**选择三相 DC 模式** 1：三相 DC 模式打开，PWM_A[0]输出通过 MCCCP 0：关闭三相 DC 模式
[30]	ACMODE	**选择三相 AC 模式** 1：三相 AC 模式打开,所有 PWM 通道都使用通道 0 的定时计数器和周期寄存器 0：关闭三相 AC 模式,每个 PWM 通道使用其自身的定时计数器和周期寄存器
[29]	INVB	**控制 3 个通道的 MCOB 输出的极性(在三相 DC 模式中和三线 PWM 模式下使用）** **0：**MCOB 正常输出 **1：**MCOB 取反后输出
[28]	CHGPH	**使能移相功能** **1：使能移相功能** **0：禁止移相功能**
[27]	HALLS_DIS	**禁止 HALLS 寄存器自动更新功能** 1: 禁止 HALL 寄存器自动更新 HALLS 寄存器（要使能移相功能必须禁止 HALLS 寄存器自动更新功能） 0：HALL 寄存器可以自动更新 HALLS 寄存器
[26]	**保留**	**保留**
[25]	VEL	**使能测速功能** 1：使能测速功能 0：禁用测速功能
[24]	HALL	**使能霍尔功能** 1：使能霍尔功能 0：禁用霍尔功能
[23]	B2LVL	**RUN2=0,CH2EN=1时，B2管脚上的输出值**

		1：输出1 0：输出0
[22]	A2LVL	RUN2=0,CH2EN=1时，A2管脚上的输出值 1：输出1 0：输出0
[21]	CH2EN	RUN2=0,CH2EN=1时，A2，B2管脚上的输出值由A2LVL和B2LVL来控制，用来给电机管脚一个确定的初始值。 RUN2=0,CH2EN=0时，A2，B2上输出0
[20]	DISUP2	使能/禁止通道2的寄存器更新功能 1：只要定时器在运行，功能寄存器保持原样 0：在每个PWM周期结束用写寄存器的值更新功能寄存器
[19]	DTE2	控制通道2的死区时间 1：死区时间使能 0：死区时间被禁止
[18]	POLA2	设置通道2输出管脚的极性 1：无效状态位高电平，有效状态为低电平 0：无效状态位低电平，有效状态为高电平
[17]	CENTER2	设置通道2的边沿/中心对齐PWM 1：中心对齐 0：边沿对齐
[16]	RUN2	停止/启动定时器通道2 1：启动定时器通道2 0：停止定时器通道2 当发生Abort事件，同时Abort中断被使能时，该bit会被自动清零
[15]	B1LVL	RUN1=0,CH1EN=1时，B1管脚上的输出值 1：输出1 0：输出0

[14]	A1LVL	**RUN1=0,CH1EN=1时，A1管脚上的输出值** **1：输出1** **0：输出0**
[13]	CH1EN	**RUN1=0,CH1EN=1时，A1，B1管脚上的输出值由A1LVL和B1LVL来控制，用来给电机管脚一个确定的初始值。** **RUN1=0,CH1EN=0时，A1，B1上输出0**
[12]	DISUP1	**使能/禁止通道1的寄存器更新功能** 1：只要定时器在运行，功能寄存器保持原样 0：在每个PWM周期结束用写寄存器的值更新功能寄存器
[11]	DTE1	**控制通道1的死区时间** 1：死区时间使能 0：死区时间被禁止
[10]	POLA1	**设置通道1输出管脚的极性** 1：无效状态位高电平，有效状态为低电平 0：无效状态位低电平，有效状态为高电平
[9]	CENTER1	**设置通道1的边沿/中心对齐PWM** 1：中心对齐 0：边沿对齐
[8]	RUN1	**停止/启动定时器通道1** 1：启动定时器通道1 0：停止定时器通道1 当发生Abort事件，同时Abort中断被使能时，该bit会被自动清零
[7]	B0LVL	**RUN0=0,CH0EN=1时，B0管脚上的输出值** **1：输出1** **0：输出0**
[6]	A0LVL	**RUN0=0,CH0EN=1时，A0管脚上的输出值** **1：输出1**

		0：输出0
[5]	CH0EN	**RUN0=0,CH0EN=1时，A0，B0管脚上的输出值由A0LVL和B0LVL来控制，用来给电机管脚一个确定的初始值。** **RUN0=0,CH0EN=0时，A0，B0上输出0**
[4]	DISUP0	**使能/禁止通道0的寄存器更新功能** 1：只要定时器在运行，功能寄存器保持原样 0：在每个PWM周期结束用写寄存器的值更新功能寄存器
[3]	DTE0	**控制通道0的死区时间** 1：死区时间使能 0：死区时间被禁止
[2]	POLA0	**设置通道0输出管脚的极性** 1：无效状态位高电平，有效状态为低电平 0：无效状态位低电平，有效状态为高电平
[1]	CENTER0	**设置通道0的边沿/中心对齐PWM** 1：中心对齐 0：边沿对齐
[0]	RUN0	**停止/启动定时器通道0** 1：启动定时器通道0 0：停止定时器通道0 当发生Abort事件，同时Abort中断被使能时，该bit会被自动清零

6.11.6.2 CAPCON 寄存器

CAPCON是只读寄存器。CAPCON寄存器控制着所有MCPWM输入通道的事件检测。3个CAP外部引脚中的任何一个输入都可用来触发所有3个通道上的捕获事件。该寄存器中的位通过写1到寄存器CAPCON_SET和CAPCON_CLR的相应位来置位和清

零。

寄存器	偏移量	R/W	描述	复位值
CAPCON	MCPWM_BA+0x00C	R	捕获控制读寄存器	0x0000_0000
CAPCON_SET	MCPWM_BA+0x010	W	捕获控制置位寄存器	0x0000_0000
CAPCON_CLR	MCPWM_BA+0x014	W	捕获控制清除寄存器	0x0000_0000

31	30	29	28	27	26	25	24
保留							
23	22	21	20	19	18	17	16
HNFCAP2	HNFCAP1	HNFCAP0	RT2	RT1	RT0	C2I2_FE	C2I2_RE
15	14	13	12	11	10	9	8
C2I1_FE	C2I1_RE	C2I0_FE	C2I0_RE	C1I2_FE	C1I2_RE	C1I1_FE	C1I1_RE
7	6	5	4	3	2	1	0
C1I0_FE	C1I0_RE	C0I2_FE	C0I2_RE	C0I1_FE	C0I1_RE	C0I0_FE	C0I0_RE

Bits	描述	
[31：24]	保留	保留
[23]	HNFCAP2	**设置 CAP2 的硬件噪声滤波器** 1：CAP2 通道上的捕获事件被延迟，内容参见“捕获事件”一节 0：禁用 CAP2 通道的硬件噪声滤波器
[22]	HNFCAP1	**设置 CAP1 的硬件噪声滤波器** 1：CAP1 通道上的捕获事件被延迟，内容参见“捕获事件”一节 0：禁用 CAP1 通道的硬件噪声滤波器
[21]	HNFCAP0	**设置 CAP0 的硬件噪声滤波器** 1：CAP0 通道上的捕获事件被延迟，内容参见“捕获事件”一节 0：禁用 CAP0 通道的硬件噪声滤波器
[20]	RT2	**设为 1 时 CAP2 通道上出现捕获事件会复位通道 2 的计数器，同时**

		A2 输出有效状态，B2 输出无效状态
[19]	RT1	设为 1 时 CAP1 通道上出现捕获事件会复位通道 1 的计数器，同时 A1 输出有效状态，B1 输出无效状态
[18]	RT0	设为 1 时 CAP0 通道上出现捕获事件会复位通道 0 的计数器，同时 A0 输出有效状态，B0 输出无效状态
[17]	C2I2_FE	设为 1 时，在 MCI2 的下降沿上使能通道 2 的捕获事件
[16]	C2I2_RE	设为 1 时，在 MCI2 的上升沿上使能通道 2 的捕获事件
[15]	C2I1_FE	设为 1 时，在 MCI1 的下降沿上使能通道 2 的捕获事件
[14]	C2I1_RE	设为 1 时，在 MCI1 的上升沿上使能通道 2 的捕获事件
[13]	C2I0_FE	设为 1 时，在 MCI0 的下降沿上使能通道 2 的捕获事件
[12]	C2I0_RE	设为 1 时，在 MCI0 的上升沿上使能通道 2 的捕获事件
[11]	C1I2_FE	设为 1 时，在 MCI2 的下降沿上使能通道 1 的捕获事件
[10]	C1I2_RE	设为 1 时，在 MCI2 的上升沿上使能通道 1 的捕获事件
[9]	C1I1_FE	设为 1 时，在 MCI1 的下降沿上使能通道 1 的捕获事件
[8]	C1I1_RE	设为 1 时，在 MCI1 的上升沿上使能通道 1 的捕获事件
[7]	C1I0_FE	设为 1 时，在 MCI0 的下降沿上使能通道 1 的捕获事件
[6]	C1I0_RE	设为 1 时，在 MCI0 的上升沿上使能通道 1 的捕获事件
[5]	C0I2_FE	设为 1 时，在 MCI2 的下降沿上使能通道 0 的捕获事件
[4]	C0I2_RE	设为 1 时，在 MCI2 的上升沿上使能通道 0 的捕获事件
[3]	C0I1_FE	设为 1 时，在 MCI1 的下降沿上使能通道 0 的捕获事件
[2]	C0I1_RE	设为 1 时，在 MCI1 的上升沿上使能通道 0 的捕获事件
[1]	C0I0_FE	设为 1 时，在 MCI0 的下降沿上使能通道 0 的捕获事件
[0]	C0I0_RE	设为 1 时，在 MCI0 的上升沿上使能通道 0 的捕获事件

6.11.6.3 TC0/1/2 寄存器

该寄存器包含了通道 0/1/2 的 16 位定时器的当前值。每个值在每个 PCLK 上

递增，或在 MCIO-2 引脚的边沿上递增（按照 CNTCON 寄存器选择的执行）。定时器/计数器从 0 开始递增计数直至它达到其相应的 LIM 寄存器的值为止（或通过写 CON_CLR 来停止计数）。

TC 寄存器可在任何时候读取，但仅当其通道停止时才可以被写入。

寄存器	偏移量	R/W	描述	复位值
TC0/1/2	MCPWM_BA+ 0x018/1C/20	R/W	通道0/1/2计数寄存器	0x0000_0000

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

Bits	描述	
[15：0]	VALUE	反映 32 位定时器的当前值

6.11.6.4 LIM0/1/2 寄存器

这些寄存器保存了定时器0-2的界限值。当定时器到达其相应界限值时会出现2种情况：在边沿对齐PWM模式下，TC复位，然后从0开始计数；在中心对齐PWM模式下，它开始从0向该值递增计数，然后再从该值向0递减计数。

如果通道寄存器CON中的CENTER位为0，选择边沿对齐模式，那么当TC和LIM匹配时通道的A输出从“有效”状态切换为“无效”状态。如果通道的CENTER和DTE位在CON中都为0，那么上述匹配同时将通道的B输出从“无效”状态切换为

“有效”状态。

如果通道的CENTER位为0，但DTE位为1时，那么上述匹配将触发通道死区时间计数器开始计数，当死区时间计数器溢出时，通道的B输出从“无效”状态切换为“有效”状态。

在中心对齐模式下，通道的TC和LIM寄存器之间的匹配对其A和B输出没有影响。

写界限寄存器或匹配寄存器都会将写入值装载到“写”寄存器，对于停止的通道，它还会装载到与TC比较的“操作”寄存器。对于运行通道，如果CON的“禁止更新”位为0，那么操作寄存器载入写寄存器的值，如下所示：1）在边沿对齐模式下，当TC与操作界限寄存器匹配时；2）在中心对齐模式下，当TC递减计数到0时。若通道正在运行且“禁止更新”位为1，那么操作寄存器不会从写寄存器中载入，直至软件停止通道。

读取LIM地址总是返回操作值。

寄存器	偏移量	R/W	描述	复位值
LIM0/1/2	**MCPWM_BA+ 0x024/28/2C**	**R/W**	**通道0/1/2界限寄存器**	**0xffff_ffff**

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

Bits	描述

[31:16]	保留	保留
[15：0]	LIM	设置 16 位的界限值

6.11.6.5 MAT0/1/2 寄存器

这些寄存器和上面描述的界限寄存器一样也具有“写”和“操作”寄存器。操作寄存器也可以和通道的TC比较。有关读和写界限和匹配寄存器的细节请参见“LIMO/1/2寄存器”小节。

匹配和界限寄存器控制着MCOO-2输出。要使匹配寄存器在其通道上有效，其值必须比相应界限寄存器的值小。

- 边沿对齐模式下的匹配寄存器

如果通道寄存器CON中的CENTER位为0，选择边沿对齐模式，那么在TC和MAT之间的匹配将通道的B输出从“有效”状态切换为“无效”状态。如果通道的CENTER和DTE位在MCCON中都为0，那么上述匹配同时将通道的A输出从“无效”状态切换为“有效”状态。

如果通道的CENTER位为0，但DTE位为1，那么上述匹配触发通道的死区时间计数器从0开始计数,当死区时间计数器溢出时，通道的A输出从“无效”状态切换为“有效”状态。

- 中心对齐模式下的匹配寄存器

如果通道寄存器CON中的CENTER位为1，选择中心对齐模式，那么当TC递增时TC和MAT之间的匹配将通道的B输出从“有效”状态切换为“无效”状态，而当TC递减时匹配将A输出从“有效”状态切换为“无效”状态。如果通道寄存器CON中的CENTER位为1，但DTE位为0，那么上述匹配将同时反方向切换通道的其它输出。

如果通道的CENTER和DTE位都为1，那么TC和MAT匹配会触发通道的死区时间计数器从0开始计数，当死区时间计数器溢出时，如果TC在匹配时递增计数，则通道的B输出从“无效”状态切换为“有效”状态；如果TC在匹配时递减计数，则通道的A输出从“无效”状态切换为“有效”状态。

- 0和100%占空比

要使通道的MCO输出锁定在“B有效，A无效”这个状态，只需要写入较大的值到它的匹配寄存器，且该值必须大于写入界限寄存器的值。这样匹配就不会发生。

要使通道的输出锁定在相反的状态，则需要：写0到匹配寄存器或按照上面所讲的，将比写入LIM值大的值写入MAT寄存器并对POLA位取反。

寄存器	偏移量	R/W	描述	复位值
MAT0/1/2	**MCPWM_BA+ 0x030/34/38**	**R/W**	**通道0/1/2的匹配寄存器**	**0xffff_ffff**

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

Bits	描述	
[31：16]	保留	保留
[15：0]	MAT	设置通道的匹配值

6.11.6.6 DT 寄存器

该寄存器保存通道0-2的死区时间值。如果通道寄存器CON中的DTE位为1以使能它的死区时间计数器，那么在其通道输出从“有效”状态变为“无效”状态时

计数器从0开始递增计数。当死区时间计数器到达设定值时，通道的其它输出从“无效”状态改变为“有效”状态。(延迟时间为DT+1)。

死区时间的操作特性是功率晶体管（例如：在电机控制应用中由A和B输出驱动的功率晶体管）完全断开所需的时间比导通的时间更长。如果A和B晶体管同时打开，那么浪费且破坏性的电流将通过晶体管在电源导轨之间流动。在这些应用中，用PCLK周期的数目来编程死区时间寄存器，PCLK周期数大于或等于晶体管的最大关断时间减去其最小的导通时间。

寄存器	偏移量	R/W	描述	复位值
DT	MCPWM_BA+0x03C	R/W	死区时间寄存器	0x3fff_ffff

31	30	29	28	27	26	25	24
保留		DT2[9:4]					
23	22	21	20	19	18	17	16
DT2[3:0]				DT1[9:6]			
15	14	13	12	11	10	9	8
DT1[5:0]						DT1[9:8]	
7	6	5	4	3	2	1	0
DT1[7:0]							

Bits	描述	
[31：30]	保留	保留
[29:20]	DT2	通道 2 的死区时间
[19:10]	DT1	通道 1 的死区时间
[9:0]	DT0	通道 0 的死区时间 ACMODE=1 时，该位控制所有 3 个通道的死区时间 ACMODE=0 时，DT0/1/2 分别控制 3 各通道的死区时间

6.11.6.7 CP 寄存器

该寄存器只在DC模式中使用。在该寄存器位的控制下，内部MCOA0信号将与6个输出管脚全部相连或与任意一个输出管脚连接。就像匹配和界限寄存器一样，该寄存器具有映射“写”和“可操作”寄存器。其它细节请见章节“IML0/1/2”寄存器和“映射寄存器和同时更新”。

寄存器	偏移量	R/W	描述	复位值
CP	**MCPWM_BA+0x040**	**R/W**	**通信格式寄存器**	**0x0000_0000**

31	30	29	28	27	26	25	24
保留							
23	22	21	20	19	18	17	16
保留							
15	14	13	12	11	10	9	8
保留							
7	6	5	4	3	2	1	0
保留		CPB2	CPA2	CPB1	CPA1	CPB0	CPA0

Bits	描述	
[31：6]	保留	保留
[5]	CPB2	0＝MCOB2 断开 1＝MCOB2 跟踪内部 MCOA0
[4]	CPA2	0＝MCOA2 断开 1＝MCOA2 跟踪内部 MCOA0
[3]	CPB1	0＝MCOB1 断开 1＝MCOB1 跟踪内部 MCOA0
[2]	CPA1	0＝MCOA1 断开

		1＝MCOA1 跟踪内部 MCOA0
[1]	CPB0	0＝MCOB0 断开 1＝MCOB0 跟踪内部 MCOA0
[0]	CPA0	0＝MCOA0 断开 1＝内部 MACOA0 激活

6.11.6.8 CAP0/1/2 寄存器

CAPCON寄存器允许软件选择MCPWM输入0-2的任意边沿作为每个通道的捕获事件。当通道上出现捕获事件时，该通道的当前TC值就保存到它的只读捕获寄存器中。

这些寄存器地址为只读，但下面的寄存器可通过写CAP_CLR寄存器来清零。

寄存器	偏移量	R/W	描述	复位值
CAP0/1/2	**MCPWM_BA+ 0x044/48/4C**	**R/W**	**通道0/1/2捕获寄存器**	**0x0000_0000**

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

Bits	描述	
[31:16]	保留	保留

[15：0]	CR0/1/2	出现捕获事件时通道 0,1,2 上定时器的值

6.11.6.9 INTEN 寄存器

INTEN寄存器反映了每个MCPWM中断的使能/禁能状态。该地址为只读，要使能或禁能中断，则使用INTEN_SET和INTEN_CLR寄存器地址。

寄存器	偏移量	R/W	描述	复位值
INTEN	**MCPWM_BA+0x050**	**R**	**中断使能读寄存器**	**0x0000_0000**
INTEN_SET	**MCPWM_BA+0x054**	**W**	**中断使能置位寄存器**	**0x0000_0000**
INTEN_CLR	**MCPWM_BA+0x058**	**W**	**中断使能清除寄存器**	**0x0000_0000**

31	30	29	28	27	26	25	24
保留	CTZERO2	MATF2	MATR2	保留	CTZERO1	MATF1	MATR1
23	22	21	20	19	18	17	16
保留	CTZERO0	MATF0	MATR0	VFAST	VSLOW	STERR	STCHG
15	14	13	12	11	10	9	8
保留			ABORT	保留	ICAP2	IMAT2	ILIM2
7	6	5	4	3	2	1	0
保留	ICAP1	IMAT1	ILIM1	保留	ICAP0	IMAT0	ILIM0

Bits	描述	
[31]	保留	保留
[30]	CTZERO2	CH2 中心对齐模式下，TC 值下降到 0 时触发中断 1：使能 0：禁止
[29]	MATF2	CH2 中心对齐模式下，TC 值下降到 MAT 时触发中断 1：使能 0：禁止

[28]	MATR2	CH2 中心对齐模式下，TC 值上升到 MAT 时触发中断 1：使能 0：禁止
[27]	保留	保留
[26]	CTZERO1	CH1 中心对齐模式下，TC 值下降到 0 时触发中断 1：使能 0：禁止
[25]	MATF1	CH1 中心对齐模式下，TC 值下降到 MAT 时触发中断 1：使能 0：禁止
[24]	MATR1	CH1 中心对齐模式下，TC 值上升到 MAT 时触发中断 1：使能 0：禁止
[23]	保留	保留
[22]	CTZERO0	CH0 中心对齐模式下，TC 值下降到 0 时触发中断 1：使能 0：禁止
[21]	MATF0	CH0 中心对齐模式下，TC 值下降到 MAT 时触发中断 1：使能 0：禁止
[20]	MATR0	CH0 中心对齐模式下，TC 值上升到 MAT 时触发中断 1：使能 0：禁止
[19]	VFAST	高速中断 1：使能 0：禁止
[18]	VSLOW	低速中断 1：使能

		0：禁止
[17]	STERR	**电机运转状态错误中断** 1：使能 0：禁止
[16]	STCHG	**电机运转状态改变中断** 1：使能 0：禁止
[15:13]	保留	保留
[12]	ABORT	**快速中止中断** 1：使能 0：禁止
[11]	保留	保留
[10]	ICAP2	**通道 2 的捕获中断** 1：使能 0：禁止
[9]	IMAT2	**通道 2 的匹配中断** 1：使能 0：禁止
[8]	ILIM2	**通道 2 的界限中断** 1：使能 0：禁止
[7]	保留	保留
[6]	ICAP1	**通道 1 的捕获中断** 1：使能 0：禁止
[5]	IMAT1	**通道 1 的匹配中断** 1：使能

		0：禁止
[4]	ILIM1	**通道 1 的界限中断** 1：使能 0：禁止
[3]	**保留**	**保留**
[2]	ICAP0	**通道 0 的捕获中断** 1：使能 0：禁止
[1]	IMAT0	**通道 0 的匹配中断** 1：使能 0：禁止
[0]	ILIM0	**通道 0 的界限中断** 1：使能 0：禁止

6.11.6.10 CNTCON 寄存器

CNTCON寄存器控制MCPWM通道的定时器或计数器模式以及在计数器模式时计数器是在其中一个MCI输入或全部3个MCI输入的上升沿上加1还是在下降沿上加1。若选择的是定时器模式，那么计数器随PCLK时钟递增。

该地址为只读。要想设置或清除寄存器的位，可向ICNTCON_SET和CNTCON_CLR寄存器地址写入1。

寄存器	偏移量	R/W	描述	复位值
CNTCON	**MCPWM_BA+0x05C**	**R**	**计数控制读寄存器**	**0x0000_0000**
CNTCON_SET	**MCPWM_BA+0x060**	**W**	**计数控制置位寄存器**	**0x0000_0000**
CNTCON_CLR	**MCPWM_BA+0x064**	**W**	**计数控制清除寄存器**	**0x0000_0000**

31	30	29	28	27	26	25	24
CNTR2	CNTR1	CNTR0	保留				
23	22	21	20	19	18	17	16
保留						TC2I2_FE	TC2I2_RE
15	14	13	12	11	10	9	8
TC2I1_FE	TC2I1_RE	TC2I0_FE	TC2I0_RE	TC1I2_FE	TC1I2_RE	TC1I1_FE	TC1I1_RE
7	6	5	4	3	2	1	0
TC1I0_FE	TC1I0_RE	TC0I2_FE	TC0I2_RE	TC0I1_FE	TC0I1_RE	TC0I0_FE	TC0I0_RE

Bits	描述	
[31]	CNTR2	1：通道 2 为计数器模式 0：通道 2 为定时器模式
[30]	CNTR1	1：通道 1 为计数器模式 0：通道 1 为定时器模式
[29]	CNTR0	1：通道 0 为计数器模式 0：通道 0 为定时器模式
[28:18]	保留	保留
[17]	TC2I2_FE	1：如果 CNTR2 为 1，计数器 2 在 MCI2 的下降沿上加 1 0：计数器 2 在 MCI2 的下降沿上不加 1
[16]	TC2I2_RE	1：如果 CNTR2 为 1，计数器 2 在 MCI2 的上升沿上加 1 0：计数器 2 在 MCI2 的上升沿上不加 1
[15]	TC2I1_FE	1：如果 CNTR2 为 1，计数器 2 在 MCI1 的下降沿上加 1 0：计数器 2 在 MCI1 的下降沿上不加 1
[14]	TC2I1_RE	1：如果 CNTR2 为 1，计数器 2 在 MCI1 的上升沿上加 1

		0：计数器 2 在 MCI1 的上升沿上不加 1
[13]	TC2I0_FE	1：如果 CNTR2 为 1，计数器 2 在 MCI0 的下降沿上加 1 0：计数器 2 在 MCI0 的下降沿上不加 1
[12]	TC2I0_RE	1：如果 CNTR2 为 1，计数器 2 在 MCI0 的上升沿上加 1 0：计数器 2 在 MCI0 的上升沿上不加 1
[11]	TC1I2_FE	1：如果 CNTR1 为 1，计数器 1 在 MCI2 的下降沿上加 1 0：计数器 1 在 MCI2 的下降沿上不加 1
[10]	TC1I2_RE	1：如果 CNTR1 为 1，计数器 1 在 MCI2 的上升沿上加 1 0：计数器 1 在 MCI2 的上升沿上不加 1
[9]	TC1I1_FE	1：如果 CNTR1 为 1，计数器 1 在 MCI1 的下降沿上加 1 0：计数器 1 在 MCI1 的下降沿上不加 1
[8]	TC1I1_RE	1：如果 CNTR1 为 1，计数器 1 在 MCI1 的上升沿上加 1 0：计数器 1 在 MCI1 的上升沿上不加 1
[7]	TC1I0_FE	1：如果 CNTR1 为 1，计数器 1 在 MCI0 的下降沿上加 1 0：计数器 1 在 MCI0 的下降沿上不加 1
[6]	TC1I0_RE	1：如果 CNTR1 为 1，计数器 1 在 MCI0 的上升沿上加 1 0：计数器 1 在 MCI0 的上升沿上不加 1
[5]	TC0I2_FE	1：如果 CNTR0 为 1，计数器 0 在 MCI2 的下降沿上加 1 0：计数器 0 在 MCI2 的下降沿上不加 1
[4]	TC0I2_RE	1：如果 CNTR0 为 1，计数器 0 在 MCI2 的上升沿上加 1 0：计数器 0 在 MCI2 的上升沿上不加 1
[3]	TC0I1_FE	1：如果 CNTR0 为 1，计数器 0 在 MCI1 的下降沿上加 1 0：计数器 0 在 MCI1 的下降沿上不加 1
[2]	TC0I1_RE	1：如果 CNTR0 为 1，计数器 0 在 MCI1 的上升沿上加 1 0：计数器 0 在 MCI1 的上升沿上不加 1
[1]	TC0I0_FE	1：如果 CNTR0 为 1，计数器 0 在 MCI0 的下降沿上加 1 0：计数器 0 在 MCI0 的下降沿上不加 1

[0]	TC0I0_RE	1：如果 CNTR0 为 1，计数器 0 在 MCI0 的上升沿上加 1 0：计数器 0 在 MCI0 的上升沿上不加 1

6.11.6.11 INTF 寄存器

INTF寄存器包含所有MCPWM中断标志，这些标志在相应的硬件事件出现时置位，或在写1到INTFLAG_SET寄存器地址对应的位时置位。当INTEN和该寄存器的相应位都为1时，MCPWM向中断控制器模块提交其中断请求。该寄存器地址为只读。若要设置或将INTFLAG寄存器中的位清零，可使用INTFLAG_SET和INTFLAG_CLR寄存器。

寄存器	偏移量	R/W	描述	复位值
INTF	**MCPWM_BA+0x068**	**R**	**中断标志读寄存器**	**0x0000_0000**
INTF_SET	**MCPWM_BA+0x06C**	**W**	**中断标志置位寄存器**	**0x0000_0000**
INTF_CLR	**MCPWM_BA+0x070**	**W**	**中断标志清除寄存器**	**0x0000_0000**

31	30	29	28	27	26	25	24
保留	CTZERO2	MATF2	MATR2	保留	CTZERO1	MATF1	MATR1
23	22	21	20	19	18	17	16
保留	CTZERO0	MATF0	MATR0	VFAST	VSLOW	STERR	STCHG
15	14	13	12	11	10	9	8
保留			ABORT	保留	ICAP2	IMAT2	ILIM2
7	6	5	4	3	2	1	0
保留	ICAP1	IMAT1	ILIM1	保留	ICAP0	IMAT0	ILIM0

Bits	描述	
[31]	保留	保留
[30]	CTZERO2	CH2 中心对齐模式下，TC 值下降到 0 时的中断标志位
[29]	MATF2	CH2 中心对齐模式下，TC 值下降到 MAT 时的中断标志位

[28]	MATR2	CH2 中心对齐模式下，TC 值上升到 MAT 时的中断标志位
[27]	保留	保留
[26]	CTZERO1	CH1 中心对齐模式下，TC 值下降到 0 时的中断标志位
[25]	MATF1	CH1 中心对齐模式下，TC 值下降到 MAT 时的中断标志位
[24]	MATR1	CH1 中心对齐模式下，TC 值下降到 MAT 时的中断标志位
[23]	保留	保留
[22]	CTZERO0	CH0 中心对齐模式下，TC 值下降到 0 时的中断标志位
[21]	MATF0	CH0 中心对齐模式下，TC 值下降到 MAT 时的中断标志位
[20]	MATR0	CH0 中心对齐模式下，TC 值下降到 MAT 时的中断标志位
[19]	VFAST	高速中断标志位
[18]	VSLOW	低速中断标志位
[17]	STERR	电机运转状态错误中断标志位
[16]	STCHG	电机运转状态改变中断标志位
[15:13]	保留	保留
[12]	ABORT	快速中止中断标志位
[11]	保留	保留
[10]	ICAP2	通道 2 的捕获中断标志位
[9]	IMAT2	通道 2 的匹配中断标志位
[8]	ILIM2	通道 2 的界限中断标志位
[7]	保留	保留
[6]	ICAP1	通道 1 的捕获中断标志位
[5]	IMAT1	通道 1 的匹配中断标志位
[4]	ILIM1	通道 1 的界限中断标志位
[3]	保留	保留
[2]	ICAP0	通道 0 的捕获中断标志位
[1]	IMAT0	通道 0 的匹配中断标志位
[0]	ILIM0	通道 0 的界限中断标志位

6.11.6.12 CAP_CLR 寄存器

向该寄存器的位写入1会清零相应的CAP寄存器。

寄存器	偏移量	R/W	描述	复位值
CAP_CLR	MCPWM_BA+0x074	W	通道捕获清除寄存器	0x0000_0000

31	30	29	28	27	26	25	24
保留							
23	22	21	20	19	18	17	16
保留							
15	14	13	12	11	10	9	8
保留							
7	6	5	4	3	2	1	0
保留					CLR2	CLR1	CLR0

Bits	描述	
[31:3]	保留	保留
[2]	CLR2	写 1 到这个位来清零 CAP2 寄存器，此位在置 1 的下个周期会自动清零
[1]	CLR1	写 1 到这个位来清零 CAP1 寄存器，此位在置 1 的下个周期会自动清零
[0]	CLR0	写 1 到这个位来清零 CAP0 寄存器，此位在置 1 的下个周期会自动清零

6.11.6.13 HALL 寄存器

该寄存控制MCPWM的霍尔工作模式。

寄存器	偏移量	R/W	描述	复位值
HALL	MCPWM_BA+0x078	R/W	霍尔寄存器	0x0000_0000

31	30	29	28	27	26	25	24
保留							
23	22	21	20	19	18	17	16
保留	CURH			保留	EXPH		
15	14	13	12	11	10	9	8
保留		PSLB2	PSLA2	PSLB1	PSLA1	PSLB0	PSLA0
7	6	5	4	3	2	1	0
保留		MCPB2	MCPA2	MCPB1	MCPA1	MCPB0	MCPA0

Bits	描述	
[31:23]	保留	保留
[22:20]	CURH	**霍尔信号当前状态（MCI[2:0]）** 在霍尔功能使能的情况下，该寄存器保存霍尔传感电机的霍尔信号的应有的当前状态。当接收到霍尔信号发生改变中断时，通过软件向该寄存器写入当前霍尔信号状态。
[19]	保留	保留
[18:16]	EXPH	**霍尔信号期望状态（MCI[2:0]）** 在霍尔功能使能的情况下，该寄存器保存霍尔传感电机的霍尔信号的下一个期望状态。当接收到霍尔信号发生改变中断时，通过软件向该寄存器写入下一个霍尔信号状态。
[15:14]	保留	保留
[13]	PSLB2	**CPB2 为 0 时，MCOB2 输出的无效状态值** 0=无效态为 0 1=无效态为 1

[12]	PSLA2	**CPA2 为 0 时，MCOA2 输出的无效状态值** 0=无效态为 0 1=无效态为 1
[11]	PSLB1	**CPB1 为 0 时，MCOB1 输出的无效状态值** 0=无效态为 0 1=无效态为 1
[10]	PSLA1	**CPA1 为 0 时，MCOA1 输出的无效状态值** 0=无效态为 0 1=无效态为 1
[9]	PSLB0	**CPB0 为 0 时，MCOB0 输出的无效状态值** 0=无效态为 0 1=无效态为 1
[8]	PSLA0	**CPA0 为 0 时，MCOA0 输出的无效状态值** 0=无效态为 0 1=无效态为 1
[7:6]	保留	保留
[5]	MCPB2	0＝MCOB2 断开,进入无效状态 1＝跟随 MCOB0，输出 PWM 信号
[4]	MCPA2	0＝MCOA2 断开,进入无效状态 1＝跟随 MCOA0，输出 PWM 信号
[3]	MCPB1	0＝MCOB1 断开,进入无效状态 1＝跟随 MCOB0，输出 PWM 信号
[2]	MCPA1	0＝MCOA1 断开,进入无效状态 1＝跟随 MCOA0，输出 PWM 信号
[1]	MCPB0	0＝MCOB0 断开，进入无效状态 1＝内部 MCOB0 激活，输出与 MCOA0 对应的 PWM 信号
[0]	MCPA0	0＝MCOA0 断开,进入无效状态

		1＝内部 MCOA0 激活，输出 PWM 信号

6.11.6.14 HALLS 寄存器

该寄存器为霍尔映射寄存器，当捕获到霍尔信号发生改变时，霍尔寄存器（HALL）的值赋值给映射寄存器（HALLS）。在捕获事件发生的下一个时钟，霍尔输入信号与映射寄存器EXPH和CURH做比较，如果与EXPH相等，则说明发生正确的状态切换(STCHG)；如果既不等于EXPH也不等于CURH则说明发生错误（STERR）。映射寄存器中的MCPx和PSLx被用作实际输出状态的控制位。

该寄存器也可以通过软件来手动配置。

寄存器	偏移量	R/W	描述	复位值
HALLS	**MCPWM_BA+0x07C**	**R/W**	**霍尔映射寄存器**	**0x0000_0000**

31	30	29	28	27	26	25	24
保留							
23	22	21	20	19	18	17	16
保留	CURH			保留	EXPH		
15	14	13	12	11	10	9	8
保留		PSLB2	PSLA2	PSLB1	PSLA1	PSLB0	PSLA0
7	6	5	4	3	2	1	0
保留		MCPB2	MCPA2	MCPB1	MCPA1	MCPB0	MCPA0

Bits	描述	
[31:23]	保留	保留
[22:20]	CURH	**霍尔信号当前状态（MCI[2:0]）** 在霍尔功能使能的情况下，该寄存器保存霍尔传感电机的霍尔信号的应有的当前状态。当接收到霍尔信号发生改变中断时，通过

		软件向该寄存器写入当前霍尔信号状态。
[19]	保留	保留
[18:16]	EXPH	**霍尔信号期望状态（MCI[2:0]）** 在霍尔功能使能的情况下，该寄存器保存霍尔传感电机的霍尔信号的下一个期望状态。当接收到霍尔信号发生改变中断时，通过软件向该寄存器写入下一个霍尔信号状态。
[15:14]	保留	保留
[13]	PSLB2	**CPB2 为 0 时，MCOB2 输出的无效状态值** 0=无效态为 0 1=无效态为 1
[12]	PSLA2	**CPA2 为 0 时，MCOA2 输出的无效状态值** 0=无效态为 0 1=无效态为 1
[11]	PSLB1	**CPB1 为 0 时，MCOB1 输出的无效状态值** 0=无效态为 0 1=无效态为 1
[10]	PSLA1	**CPA1 为 0 时，MCOA1 输出的无效状态值** 0=无效态为 0 1=无效态为 1
[9]	PSLB0	**CPB0 为 0 时，MCOB0 输出的无效状态值** 0=无效态为 0 1=无效态为 1
[8]	PSLA0	**CPA0 为 0 时，MCOA0 输出的无效状态值** 0=无效态为 0 1=无效态为 1
[7:6]	保留	保留
[5]	MCPB2	0＝MCOB2 断开,进入无效状态

		1＝跟随 MCOB0，输出 PWM 信号
[4]	MCPA2	0＝MCOA2 断开,进入无效状态 1＝跟随 MCOA0，输出 PWM 信号
[3]	MCPB1	0＝MCOB1 断开,进入无效状态 1＝跟随 MCOB0，输出 PWM 信号
[2]	MCPA1	0＝MCOA1 断开,进入无效状态 1＝跟随 MCOA0，输出 PWM 信号
[1]	MCPB0	0＝MCOB0 断开，进入无效状态 1＝内部 MCOB0 激活，输出与 MCOA0 对应的 PWM 信号
[0]	MCPA0	0＝MCOA0 断开,进入无效状态 1＝内部 MCOA0 激活，输出 PWM 信号

6.11.6.15 VELCMP 寄存器

寄存器存放预设的电机速度值，用于与当前实测的速度做比较。该寄存器的值表示，相邻霍尔捕获事件直接所耗费的时钟周期值。计数方法，VELCMP=(1/(vel*n))*freq，vel为转速，freq为时钟频率，n为相邻触发事件之间距离占圆周的大小，也可认为是分了多少份。比如：电机转速为100r/s，时钟频率为600KHz，霍尔传感器相邻60°角，相邻触发事件之间距离占圆周的1/6,也就是n为6.这样，VELCMP为1000.

寄存器	偏移量	R/W	描述	复位值
VELCMP	**MCPWM_BA+0x080**	**R/W**	**电机速度比较寄存器**	**0x0000_0000**

31	30	29	28	27	26	25	24
VELCMP[31:24]							
23	22	21	20	19	18	17	16
VELCMP[23:16]							

15	14	13	12	11	10	9	8
VELCMP[15:8]							
7	6	5	4	3	2	1	0
VELCMP[7:0]							

Bits	描述	
[31:0]	VELCMP	设置速度比较值

6.11.6.16 VELVAL 寄存器

该寄存器记录当前实测的速度值，也就是在相邻触发事件之间时钟周期数的大小。velcnt计数器在每一个capture0_even时从零开始计算，当下一个capture事件时，把velcnt的值赋值给VELVAL寄存器，velcnt计数器清零重新计数。

寄存器	偏移量	R/W	描述	复位值
VELVAL	MCPWM_BA+0x084	R	电机速度当前值寄存器	0x0000_0000

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

Bits	描述	
[31:0]	VELVAL	反映当前实测速度值

6.11.6.17 VELTH 寄存器

该寄存器保存可以接受的速度阈值大小。也就是说在+/-TH范围内可被认为电机运转正常。如果实测值大于CMP+TH则被认为是速度过快，可产生相应的VFAST中断，如果实测值VELVAL小于CMP-TH则被认为是速度过慢，可产生相应的VSLOW中断。

寄存器	偏移量	R/W	描述	复位值
VELTH	**MCPWM_BA+0x088**	**R/W**	**电机速度阈值寄存器**	**0x0000_0000**

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

Bits	描述	
[31:0]	VELTH	**设置速度阈值**

6.11.6.18 MCIST 寄存器

该寄存器保存MCPWM输入信号MCIO/1/2的状态值。是只读寄存器，用于判断电机当前所处的位置。

寄存器	偏移量	R/W	描述	复位值
MCIST	**MCPWM_BA+0x08C**	**R**	**输入信号状态寄存器**	**0x0000_0000**

31	30	29	28	27	26	25	24
保留							
23	22	21	20	19	18	17	16
保留							
15	14	13	12	11	10	9	8
保留							
7	6	5	4	3	2	1	0
保留					MCI2	MCI1	MCI0

Bits	描述	
[31:3]	保留	保留
[2]	MCI2	反映 **MCI2** 管脚上的状态
[1]	MCI1	反映 **MCI1** 管脚上的状态
[0]	MCI0	反映 **MCI0** 管脚上的状态

6.11.6.19 PR 寄存器

预分频寄存器，用来设置实际的技术时钟。

寄存器	偏移量	R/W	描述	复位值
PR	**MCPWM_BA+0x090**	**R/W**	时钟预分频寄存器	**0x0000_0000**

31	30	29	28	27	26	25	24
保留							
23	22	21	20	19	18	17	16
保留							
15	14	13	12	11	10	9	8

PR[15:8]							
7	6	5	4	3	2	1	0
PR[7:0]							

Bits	描述	
[31:16]	保留	保留
[15:0]	PR	**时钟预分频值** 实际计数时钟频率为：Fapbclk/(PR+1)

6.11.6.20 CHGPH 寄存器

该寄存器用来设置CH1和CH2相对于CH0的相位偏移量。

寄存器	偏移量	R/W	描述	复位值
CHGPH	**MCPWM_BA+0x094**	**R/W**	**相位偏移量寄存器**	**0x0000_0000**

31	30	29	28	27	26	25	24
CH2_PH[15:8]							
23	22	21	20	19	18	17	16
CH2_PH[7:0]							
15	14	13	12	11	10	9	8
CH1_PH[15:8]							
7	6	5	4	3	2	1	0
CH1_PH[7:0]							

Bits	描述	
[31:16]	CH2_PH	通道 **2** 相对于通道 **0** 的相位偏移（以 **APBCLK** 为单位）
[15:0]	CH1_PH	通道 **1** 相对于通道 **0** 的相位偏移（以 **APBCLK** 为单位）

6.11.6.21 HALLAEN 寄存器

HALL模式延迟换相使能寄存器，用来使能延迟换相功能。

寄存器	偏移量	R/W	描述	复位值
HALLAEN	**MCPWM_BA+0x098**	**R/W**	HALL延迟换相使能寄存器	**0x0000_0000**

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

Bits	描述	
[31:1]	保留	保留
[0]	EN	使能延迟换相功能 **1：使能 HALL 模式延迟换相功能，在 HALL 信号改变后的特定时间点（DLY_CNT 个系统时钟后）将 HALLA 寄存器的值自动更新到 HALLS 寄存器，同时 EN 位会自动清零** **0：禁止 HALL 模式延迟换相功能**

6.11.6.22 HALLA 寄存器

HALL模式延迟换相寄存器，用来配置延迟换相功能下的输出值。

寄存器	偏移量	R/W	描述	复位值
HALLA	**MCPWM_BA+0x09C**	**R/W**	HALL延迟换相配置寄存器	**0x0000_0000**

31	30	29	28	27	26	25	24
PSLB2	PSLA2	PSLB1	PSLA1	PSLB0	PSLA0	MCPB2	MCPA2
23	22	21	20	19	18	17	16
MCPB1	MCPA1	MCPB0	MCPA0	DLY_CNT[19:16]			
15	14	13	12	11	10	9	8
DLY_CNT[15:8]							
7	6	5	4	3	2	1	0
DLY_CNT[7:0]							

Bits	描述	
[31]	PSLB2	对应于 HALLS 寄存器中的 PSLB2
[30]	PSLA2	对应于 HALLS 寄存器中的 PSLA2
[29]	PSLB1	对应于 HALLS 寄存器中的 PSLB1
[28]	PSLA1	对应于 HALLS 寄存器中的 PSLA1
[27]	PSLB0	对应于 HALLS 寄存器中的 PSLB0
[26]	PSLA0	对应于 HALLS 寄存器中的 PSLA0
[25]	MCPB2	对应于 HALLS 寄存器中的 MCPB2
[24]	MCPA2	对应于 HALLS 寄存器中的 MCPA2
[23]	MCPB1	对应于 HALLS 寄存器中的 MCPB1
[22]	MCPA1	对应于 HALLS 寄存器中的 MCPA1
[21]	MCPB0	对应于 HALLS 寄存器中的 MCPB0
[20]	MCPA0	对应于 HALLS 寄存器中的 MCPA0
[19:0]	DLY_CNT	设置延迟换相的时间 当使能延迟换相功能后，**HALL** 信号发生改变后的 **DLY_CNT** 个系统时钟周期后，自动将 **HALLA** 中的 **PSLx** 和 **MCPx** 的值自动更新到 **HALLS** 寄存器的对应位。

6.11.6.23 ADC_TRIG0 寄存器

通道0触发ADC硬件采样配置寄存器，用来配置通道0触发ADC硬件采样的时间点，可选择通道0的PWM周期内的任意时间点来产生硬件触发信号。

寄存器	偏移量	R/W	描述	复位值
ADC_TRIG0	MCPWM_BA+0x0A0	R/W	通道0触发ADC采样寄存器	0x0000_0000

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

Bits	描述	
[31]	EN	1：使能通道 0 的 ADC 硬件触发功能 0：禁止通道 0 的 ADC 硬件触发功能
[30]	DIR	边沿计数模式下不起作用，中心对齐模式下用来设置计数方向 1：向下计数 0：向上计数
[29:16]	保留	保留
[15：0]	TRIG_CNT	设置产生硬件触发 ADC 信号的时间点 当通道 0 的计数器值达到 TRIG_CNT 的值时产生触发信号

6.11.6.24 ADC_TRIG1 寄存器

通道1触发ADC硬件采样配置寄存器，用来配置通道1触发ADC硬件采样的时间

点，可选择通道0的PWM周期内的任意时间点来产生硬件触发信号。

寄存器	偏移量	R/W	描述	复位值
ADC_TRIG1	MCPWM_BA+0x0A4	R/W	通道1触发ADC采样寄存器	0x0000_0000

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

Bits	描述	
[31]	EN	1：使能通道 1 的 ADC 硬件触发功能 0：禁止通道 1 的 ADC 硬件触发功能
[30]	DIR	边沿计数模式下不起作用，中心对齐模式下用来设置计数方向 1：向下计数 0：向上计数
[29:16]	保留	保留
[15：0]	TRIG_CNT	设置产生硬件触发 ADC 信号的时间点 当通道 1 的计数器值达到 TRIG_CNT 的值时产生触发信号

6.11.6.25 ADC_TRIG2 寄存器

通道2触发ADC硬件采样配置寄存器，用来配置通道2触发ADC硬件采样的时间点，可选择通道2的PWM周期内的任意时间点来产生硬件触发信号。

寄存器	偏移量	R/W	描述	复位值

ADC_TRIG2	MCPWM_BA+0x0A8	R/W	通道2触发ADC采样寄存器	0x0000_0000

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

Bits	描述	
[31]	EN	1：使能通道 2 的 ADC 硬件触发功能 0：禁止通道 2 的 ADC 硬件触发功能
[30]	DIR	边沿计数模式下不起作用，中心对齐模式下用来设置计数方向 1：向下计数 0：向上计数
[29:16]	保留	保留
[15：0]	TRIG_CNT	设置产生硬件触发 ADC 信号的时间点 当通道 2 的计数器值达到 TRIG_CNT 的值时产生触发信号

6.12 UART 接口控制器(UART)

6.12.1 概述

通用异步收发器 UART 是一个串并转换的接口模块，主要实现 CPU 与外部串行接口设备的数据，MT32F005/006 系列微控制器包含 1 个 UART 模块：UART0。

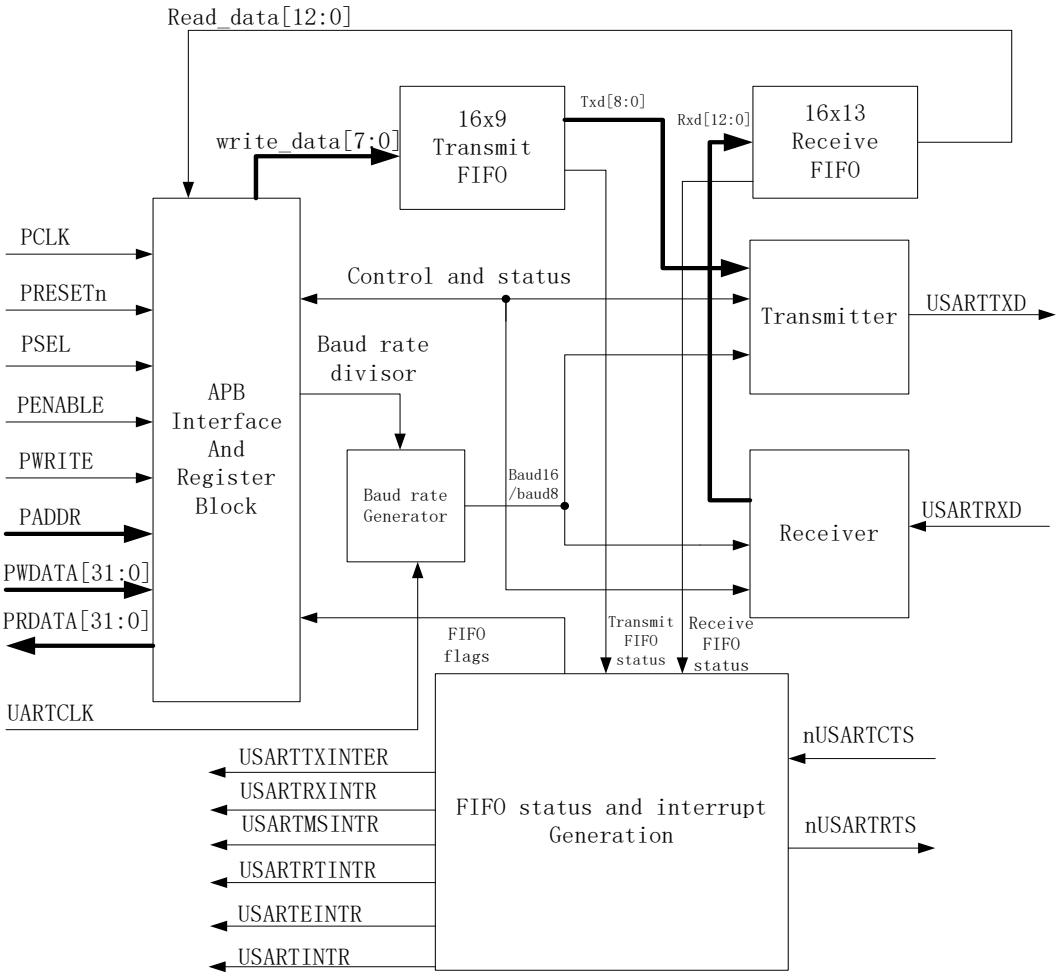
6.12.2 特性

- 全双工的，异步通信
- NRZ 标准格式
- 分数波特率发生器
- 可配置的停止位-支持 1 或 2 个停止位
- 单独的发送器和接收器使能位
- 数据接收超时中断
- 检测标志
  - 接收缓冲器满
  - 发送缓冲器空
  - 传输结束标志
- 校验控制
  - 发送校验位
  - 对接收数据进行校验
- 四个错误检测标志
  - 溢出错误
  - 噪音错误
  - 帧错误
  - 校验错误

- 总线 IDLE 检测，timeout 时间可配置
- TX/RX 极性取反模式

6.12.3 功能描述

下图为 UART 功能模块框图，APB 接口用于系统访问 UART 的状态/控制寄存器。Baud rate generator 模块用于产生 16 倍的波特率时钟，该时钟用于采样串行接口数据。接收 FIFO 和发送 FIFO 的深度都是 16，可以设置不同的触发阈值。



6.12.3.1 UART 模块时钟和复位

在使用 UART 前需在系统管理器模块中打开 UART 模块的时钟使能位 CLKCTRL.UART(bit4)，且需在 UARTCLKSEL 寄存器中选择正确的时钟源，在 UARTCLKDIV 配置非零值。如果需要复位 UART 模块，只需向寄存器 bit 位 PRESETCTRL. UART (bit4)写 0，再写 1 即可实现模块复位。

6.12.3.2 波特率计算

波特率的计算公式如下：

当选择 16 倍波特率时，波特率除数 BAUDDIV = ($F_{UARTCLK}$/ (16×波特率))

其中 $F_{UARTCLK}$ 是 UART 的参考时钟频率。

BAUDDIV 由整数值（BAUD DIVINT）和小数值（BAUD DIVFRAC）组成。

注意：

只有在完成当前字符的发送或接收后，UARTIBRD 和 UARTFBRD 的寄存器的内容才会更新。

分频的最小值为 1，最大值为 65535。

当 UARTIBRD= 65535（即是 0xFFFF），然后 UARTFBRD 必须为零。如果超过它将会导致发送或接收的中止。

示例：

当要求波特率为 230400 且 UARTCLK = 4MHz 时：
波特率除数 = (4×10^6)/ (16×230400) = 1.085
这表明整数部分 BRDI = 1 和小数部分 BRDF = 0.085。
这样，小数部分寄存器配置的值 m = integer ((0.085×64)+0.5) = 5
实际得到的波特率除数 = 1+5/64 = 1.078
实际得到的波特率 = (4×10^6) / (16×1.078) = 231911
误差 = (231911-230400) /230400 × 100 = 0.656%
使用 6-bit 的 UARTFBRD 寄存器，最大误差为 1/64×100 = 1.56%。

6.12.3.3 数据发送

发送数据的帧结构是可编程的，它由 1 个起始位、5-9 个数据位、1 个可选的奇偶校验位和 1-2 个停止位组成，这些可以在线性控制寄存器 LINECTRL 和 CTRL3 中设定。发送器还可以产生断点发送，使串行输出发送连续的逻辑 0 状态。当前发送字被完全发送出去后，这个断点信号随后发送。断点信号发送之后，继续发送数据到 TXFIFO(FIFO 禁能时则发送到 TX 保持寄存器)。

发送方式：

中断：中断情况下，将根据 TXFIFO 的触发中断完成数据的发送；

查询：查询方式和中断方式类似，是通过 CPU 不停的读取状态寄存器中的 TXFIFO 的空满状态来完成数据的发送。

6.12.3.4 数据接收

接收数据的帧格式也是可编程的。它由 1 个起始位、5-9 个数据位、1 个可选的奇偶校验位和 1-2 个停止位组成，这些可以在线性控制寄存器 LINECTRL 中设定。接收器可以检测溢出错误和帧错误，奇偶校验错误，断点出错等。

溢出错误：在旧数据被读出来之前新的数据覆盖了旧的数据

帧错误：接收数据没有有效的停止位

奇偶校验错误：校验位不满足约定的条件

断点出错：接收的数据中，有连续的逻辑 0 状态

在设置了接收超时等待时间以后，没有接收到任何数据并且接收 FIFO 非空时，将会产生一个接收超时中断。

接收方式：

中断：中断情况下，将根据接收数据读入到 RXFIFO 的深度来触发接收中断，在中断处理下完成数据的接收

查询：查询方式和中断方式类似，是通过 CPU 不停的读取状态寄存器中的

RXFIFO 的空满状态来完成数据的接收。

6.12.3.5 IDLE 检测

控制器会自动检测 uart 接口是否处于 idle 状态，如果 idle 状态持续一段时间，控制器会设置相应的标志位，如果是能了 idle 检测中断，则控制器会向 CPU 发送中断信号，具体见寄存器描述。

6.12.4 寄存器映射

R：只读，W：只写，R/W：可读写

寄存器	偏移量	R/W	描述	复位后的值
UART 的基地址： **UART0_BA = 0x4000_2000**				
UARTDR	UARTx_BA+0x00	R/W	数据寄存器	0x00000000
UARTRSR	UARTx_BA+0x04	R/W	接收状态	0x00000000
UARTTFR	UARTx_BA+0x18	R/W	标志寄存器	0x00000090
UARTIBRD	UARTx_BA+0x24	R/W	整数比特率分频寄存器	0x00000000
UARTFBRD	UARTx_BA+0x28	R/W	小数分频寄存器	0x00000000
UARTLCR_H	UARTx_BA+0x2c	R/W	线性控制寄存器，高 byte	0x00000000
UARTCR	UARTx_BA+0x30	R/W	控制寄存器	0x00000300
UARTIFLS	UARTx_BA+0x34	R/W	Fifo 中断阈值选择寄存器	0x00000012
UARTIMSC	UARTx_BA+0x38	R/W	中断 mask 位清零/置位寄存器	0x00000000
UARTRIS	UARTx_BA+0x3C	R/W	原始中断状态寄存器	0x00000000
UARTMIS	UARTx_BA+0x40	R/W	Mask 后中断状态寄存器	0x00000000
UARTICR	UARTx_BA+0x44	R/W	中断清零寄存器	0x00000000
UARTIDCR	UARTx_BA+0x50	R/W	串行线 IDLE 状态检测控制寄存器	0x00000000

6.12.4.1 UARTDR 寄存器

Bits	描述	
[31:12]	保留	保留
[11]	OE	**溢出错误标志** 只在接收时有效
[10]	BE	**break 错误标志** 只在接收时有效
[9]	PE	**奇偶校验错误标志** 只在接收时有效
[8]	FE	**帧错误标志** 只在接收时有效
[7:0]	DATA	**内部 FIFO 入口地址** 当向该寄存器写入时，8bit 数据发送到 TXFIFO 中;读该寄存器 bit 域，得到 RXFIFO 中的数据

6.12.4.2 UARTRSR 寄存器

Bits	描述	
[31:4]	保留	保留
[3]	OE	**溢出错误标志** 只在接收时有效，写 0 清零
[2]	BE	**break 错误标志** 只在接收时有效，写 0 清零
[1]	PE	**奇偶校验错误标志** 只在接收时有效，写 0 清零
[0]	FE	**帧错误标志**

		只在接收时有效，写 0 清零

6.12.4.3 UARTTFR 寄存器

Bits	描述	
[31:8]	保留	保留
[7]	TXFE	发送 FIFO 空标志；
[6]	RXFF	接受 FIFO 满标志；
[5]	TXFF	发送 FIFO 满标志；
[4]	RXFE	接受 FIFO 空标志；
[3]	BUSY	为 1 时表示 UART 正在发送数据
[2:0]	保留	保留

6.12.4.4 UARTIBRD 寄存器

Bits	描述	
[31:16]	保留	保留
[15:0]	BAUD DIVINT	波特率分频值的整数部分 该寄存器写入以后，需等到 UARTLCR_H 寄存器写操作后才 LOAD 到波特率计数器中

6.12.4.5 UARTFBRD 寄存器

Bits	描述	
[31:6]	保留	保留
[5:0]	BAUD DIVFRAC	波特率分频值的小数部分

6.12.4.6 UARTLCR_H 寄存器

Bits	描述	
[31:8]	保留	保留
[7]	SPS	**固定校验使能位** 0:禁能固定校验 1:使能固定校验 固定校验位使能，PEN=1 有效时，该位生效； 当这个寄存器中的 bit1、bit2 和 bit7 都被置 1 时，校验位固定为 0，当 bit1 和 bit7 被置 1 且 bit2 被清零，校验位固定为 1。
[6:5]	WLEN	**有效数据位字长** 00：字长 5 位 01：字长 6 位 10：字长 7 位 11：字长 8 位
[4]	FEN	**FIFO 使能位** 0:接收和发送 FIFO 禁能，深度为 1 的保持寄存器 1:接收和发送 FIFO 使能，FIFO 缓存模式
[3]	STP2	**停止位选择位** 0：一个停止位 1：两个停止位
[2]	EPS	**奇偶校验选择位** 0：PEN 有效时，奇校验 1：PEN 有效时，偶校验
[1]	PEN	**奇偶校验位使能** 0：奇偶校验位禁能 1：奇偶校验位使能

[0]	BRK	发送 break 位使能 0：无影响，正常工作 1： USART 结束输出当前字符后，继续输出低电平

6.12.4.7 UARTCR 寄存器

Bits	描述	
[31:10]	保留	保留
[9]	RXE	置 1 表示接收使能，复位值为 1
[8]	TXE	置 1 表示发送使能，复位值为 1
[7:1]	保留	保留
[0]	UARTEN	置 1 表示 UART 使能，复位值为 0

6.12.4.8 UARTIFLS 寄存器

Bits	描述	
[31:6]	保留	保留
[5:3]	RXIFLSEL	选择 RXFIFO 阈值 当数据深度等于或大于当触发阈值时，发送中断 1/8 = 0x0 深度 16 的 RXFIFO，level 多于 2 触发 1/4=0x1 深度 16 的 RXFIFO，level 多于 4 触发 1/2=0x2 深度 16 的 RXFIFO，level 多于 8 触发 3/4=0x3 深度 16 的 RXFIFO，level 多于 12 触发 7/8=0x4 深度 16 的 RXFIFO，level 多于 14 触发 INVALID5 = 0x5 保留. INVALID6 = 0x6 保留. INVALID7 = 0x7 保留.
[2:0]	TXIFLSEL	选择 TXFIFO 阈值

		当数据深度等于或少于触发阈值时，发送中断 1/8 = 0x0 深度 16 的 TXFIFO，level 少于 2 触发 1/4=0x1 深度 16 的 TXFIFO，level 少于 4 触发 1/2=0x2 深度 16 的 TXFIFO，level 少于 8 触发 3/4=0x3 深度 16 的 TXFIFO，level 少于 12 触发 7/8=0x4 深度 16 的 TXFIFO，level 少于 14 触发 INVALID5 = 0x5 保留. INVALID6 = 0x6 保留. INVALID7 = 0x7 保留.

6.12.4.9 UARTIMSC 寄存器

Bits	描述	
31:12	Reserved	Reserved
11	IDIM	**IDLE 检测中断使能** 0：IDLE 检测中断禁能 1：IDLE 检测中断使能
10	OEIM	**溢出中断使能** 0：溢出中断禁能 1：溢出中断使能
9	BEIM	**BREAK 中断使能** 0：BREAK 错误中断禁能 1：BREAK 错误中断使能
8	PEIM	**奇偶校验错误中断时能** 0：奇偶校验错误中断禁能 1：奇偶校验错误中断使能
7	FEIM	**帧错误中断使能** 0：帧错误中断禁能 1：帧错误中断使能
6	RTIM	**接收超时中断使能**

		0：接收超时中断禁能 1：接收超时中断使能
5	TXIM	**发送中断使能** 0：发送中断禁能 1：发送中断使能
4	RXIM	**接收中断使能** 0：接收中断禁能 1：接收中断使能
3:0	Reserved	Reserved

6.12.4.10 UARTRIS 寄存器

Bits	描述	
31:12	Reserved	Reserved
11	IDRIS	**Raw IDLE 检测中断状态** 0：无 IDLE 检测中断状态 1：有 IDLE 检测中断状态
10	OERIS	**Raw 溢出中断使能** 0：无溢出中断状态 1：有溢出中断状态
9	BERIS	**Raw BREAK 中断使能** 0：无 BREAK 错误中断状态 1：有 BREAK 错误中断状态
8	PERIS	**Raw 奇偶校验错误中断时能** 0：无奇偶校验错误中断状态 1：有奇偶校验错误中断状态
7	FERIS	**Raw 帧错误中断使能** 0：无帧错误中断状态 1：有帧错误中断状态
6	RTRIS	**Raw 接收超时中断使能** 0：无接收超时中断状态

		1：有接收超时中断状态
5	TXRIS	**Raw 发送中断使能** 0：无发送中断状态 1：有发送中断状态
4	RXRIS	**Raw 接收中断使能** 0：无接收中断状态 1：有接收中断状态
3:0	Reserved	Reserved

6.12.4.11 UARTMIS 寄存器

Bits	描述	
31:12	Reserved	Reserved
11	IDMIS	**IDLE 检测中断状态** 0：无 IDLE 检测中断 1：有 IDLE 检测中断
10	OEMIS	**溢出中断状态** 0：无溢出中断 1：有溢出中断
9	BEMIS	**BREAK 中断状态** 0：无 BREAK 错误中断 1：有 BREAK 错误中断
8	PEMIS	**奇偶校验错误中断状态** 0：无奇偶校验错误中断 1：有奇偶校验错误中断
7	FEMIS	**帧错误中断状态** 0：无帧错误中断 1：有帧错误中断
6	RTMIS	**接收超时中断状态** 0：无接收超时中断 1：有接收超时中断

5	TXMIS	**发送中断状态** 0：无发送中断 1：有发送中断
4	RXMIS	**接收中断状态** 0：无接收中断 1：有接收中断
3:0	Reserved	Reserved

6.12.4.12 UARTICR 寄存器

Bits	描述	
31:12	Reserved	Reserved
11	IDIC	**写 1 清除 IDLE 检测中断使能** 该 bit 硬件自动清零
10	OEIC	**写 1 清除溢出中断标志** 该 bit 硬件自动清零
9	BEICS	**写 1 清除 BREAK 中断标志** 该 bit 硬件自动清零
8	PEIC	**写 1 清除奇偶校验错误中断标志** 该 bit 硬件自动清零
7	FEIC	**写 1 清除帧错误中断标志** 该 bit 硬件自动清零
6	RTIC	**写 1 清除接收超时中断标志** 该 bit 硬件自动清零
5	TXIC	**写 1 清除发送中断标志** 该 bit 硬件自动清零
4	RXIC	**写 1 清除接收中断标志** 该 bit 硬件自动清零

3:0	Reserved	Reserved

6.12.4.13 UARTIDCR 寄存器

Bits	描述	
31:9	Reserved	Reserved
8	IDLE_E	**IDEL 检测使能位** 0：不使能 IDLE 检测 1：使能 IDLE 检测
7:0	IDLE_CNT	**设置 IDLE 检测的时间** 单位为 11 个波特率长度，当 IDLE 时间大于（11*(IDLE_CNT+1)*baud）时产生 IDLE 检测中断。

6.13 窗看门狗 WWDG

6.13.1 概述

窗口看门狗通常被用来监测，由外部干扰或不可预见的逻辑条件造成的应用程序背离正常的运行序列而产生的软件故障。除非递减计数器的值在 T6 位变成 0 前被刷新，看门狗电路在达到预置的时间周期时，会产生一个 MCU 复位。在递减计数器达到窗口寄存器数值之前，如果 7 位的递减计数器数值(在控制寄存器中)被刷新， 那么也将产生一个 MCU 复位。这表明递减计数器需要在一个有限的时间窗口中被刷新。

6.13.2 基本特性

- 可编程的自由运行递减计数器；
- 条件复位
  - 当递减计数器的值小于 0x40，(若看门狗被启动)则产生复位。
  - 当递减计数器在窗口外被重新装载，(若看门狗被启动)则产生复位。
- 如果启动了看门狗并且允许中断，当递减计数器等于 0x40 时产生早期唤醒中断(EWI)，它可以被用于重装载计数器以避免 WWDG 复位。

6.13.3 基本配置

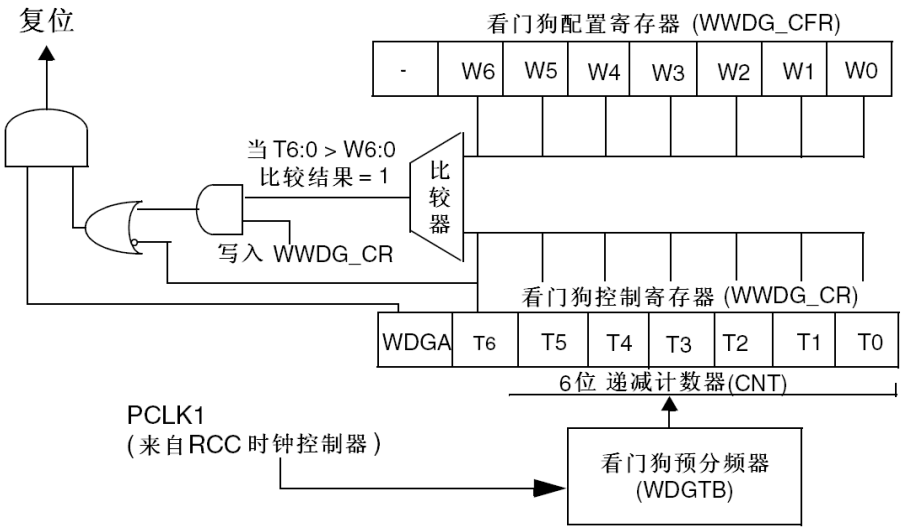
使用以下系统寄存器来打开 WWDG 模块：

1） 时钟：WWDG 模块的工作时钟默认是关闭的，使用前必须在 AHBCLKCTRL0 寄存器中打开对应的控制位（bit7）。

2） 复位：WWDG 模块的复位默认是高，即处于正常工作状态，如果要对 WWDG 模块复位，应先将 PRESET_CTRL0 的对应位（bit7）置为 0，然后再置为 1。

6.13.4 功能描述

如果看门狗被启动(WWDG_CR 寄存器中的 WDGA 位被置'1')， 并且当 7 位(T[6:0])递减计数器从 0x40 翻转到 0x3F(T6 位清零)时，则产生一个复位。如果软件在计数器值大于窗口寄存器中的数值时重新装载计数器，将产生一个复位。



图：窗看门狗框图

应用程序在正常运行过程中必须定期地写入 WWDG_CR 寄存器以防止 MCU 发生复位。只有当计数器值小于窗口寄存器的值时，才能进行写操作。储存在 WWDG_CR 寄存器中的数值必须在 0xFF 和 0xC0 之间：

● 启动看门狗 在系统复位后，看门狗总是处于关闭状态，设置 WWDG_CR 寄存器的 WDGA 位能够开启看门狗，随后它不能再被关闭，除非发生复位。

● 控制递减计数器 递减计数器处于自由运行状态，即使看门狗被禁止，递减计数器仍继续递减计数。当看门狗被启用时，T6 位必须被设置，以防止立即产生一个复位。 T[5:0]位包含了看门狗产生复位之前的计时数目；复位前的延时时间在一个最小值和一个最大值之间变化，这是因为写入 WWDG_CR 寄存器时，预分频值是未知的。

配置寄存器(WWDG_CFR) 中包含窗口的上限值：要避免产生复位，递减

计数器必须在其值小于窗口寄存器的数值并且大于 0x3F 时被重新装载，0 描述了窗口寄存器的工作过程。 另一个重装载计数器的方法是利用早期唤醒中断(EWI)。设置 WWDG_CFR 寄存器中的 WEI 位开启该中断。当递减计数器到达 0x40 时，则产生此中断，相应的中断服务程序(ISR)可以用来加载计数器以防止 WWDG 复位。在 WWDG_SR 寄存器中写'0'可以清除该中断。 注： 可以用 T6 位产生一个软件复位(设置 WDGA 位为'1'，T6 位为'0')。

6.13.5 寄存器描述

R: 只读, W: 只写, R/W: 可读写

寄存器	偏移量	R/W	描述	复位后的值
WWDG 的基地址： **WWDG_BA = 0x4000_C000**				
WWDG_CR	WWDG_BA+0x00	R/W	控制寄存器	0x7F
WWDG_CFR	WWDG _BA+0x04	R/W	配置寄存器	0x7F
WWDG_SR	WWDG _BA+0x08	R/W	状态寄存器	0x0

6.13.5.1 WWDG_CTRL 寄存器

寄存器	偏移量	R/W	描述	复位值
WWDG_CTRL	0x000	R/W	控制寄存器	**0x**7F

31	30	29	28	27	26	25	24
保留							
23	22	21	20	19	18	17	16
保留							
15	14	13	12	11	10	9	8
保留							

7	6	5	4	3	2	1	0
WDGA	T						

Bits	描述	
[31:9]	保留	保留
[8]	保留	保留
[7]	WDGA	位 7 激活位 (Activation bit) 此位由软件置'1'，但仅能由硬件在复位后清'0'。当 WDGA=1 时，看门狗可以产生复位。 0：禁止看门狗 1：启用看门狗
6:0	T	7 位计数器(MSB 至 LSB) (7-bit counter) 这些位用来存储看门狗的计数器值。每(4096x2WDGTB)个 PCLK1 周期减 1。当计数器值从 40h 变为 3Fh 时(T6 变成 0)，产生看门狗复位。

6.13.5.2 WWDG_CFR 寄存器

寄存器	偏移量	R/W	描述	复位值
WWDG_CFR	0x004	R/W	配置寄存器	**0x**7F

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

Bits	描述	
[31:10]	保留	保留
[9]	EWI	提前唤醒中断 (Early wakeup interrupt) 此位若置'1'，则当计数器值达到 40h，即产生中断。 此中断只能由硬件在复位后清除。
[8:7]	WDGTB	时基 (Timer base) 预分频器的时基可以设置如下： 00: CK 计时器时钟(PCLK 除以 16384)除以 1 01: CK 计时器时钟(PCLK 除以 16384)除以 2 10: CK 计时器时钟(PCLK 除以 16384)除以 4 11: CK 计时器时钟(PCLK 除以 16384)除以 8
[6:0]	W	7 位窗口值 (7-bit window value) 这些位包含了用来与递减计数器进行比较用的窗口值。。

6.13.5.3 WWDG_SR 寄存器

寄存器	偏移量	R/W	描述	复位值
WWDG_SR	0x008	R/W	状态寄存器	**0x0**

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

Bits	描述	
[31:1]	保留	保留
[0]	EWIF	提前唤醒中断标志 (Early wakeup interrupt flag) 当计数器值达到 40h 时，此位由硬件置’1’。它必须通过软件写’0’来清除。对此位写’1’无效。若中断未被使能，此位也会被置’1’。

6.14 独立看门狗 IWDG

6.14.1 概述

MT32F005/006 内置两个看门狗，提供了更高的安全性、时间的精确性和使用的灵活性。两个看门狗设备(独立看门狗和窗口看门狗)可用来检测和解决由软件错误引起的故障；当计数器达到给定的超时值时，触发一个中断(仅适用于窗口型看门狗)或产生系统复位。独立看门狗(IWDG)由专用的低速时钟(25K IRC)驱动，即使主时钟发生故障它也仍然有效。窗口看门狗使用系统时钟做为驱动时钟，通过可配置的时间窗口来检测应用程序非正常的过迟或过早的操作。IWDG 最适合应用于那些需要看门狗作为一个在主程序之外，能够完全独立工作，并且对时间精度要求较低的场合。WWDG 最适合那些要求看门狗在精确计时窗口起作用的应用程序。

6.14.2 基本特性

- 自由运行的递减计数器
- 时钟由独立的 25K IRC 振荡器提供(可在停止和待机模式下工作)
- 看门狗被激活后，则在计数器计数至 0x000 时产生复位

6.14.3 基本配置

使用以下系统寄存器来打开 IWDG 模块：

- 时钟：IWDG 模块的工作时钟默认是关闭的，使用前必须在 AHBCLKCTRL0 寄存器中打开对应的控制位（bit12）。
- 复位：IWDG 模块的复位默认是高，即处于正常工作状态，如果要对 IWDG 模块复位，应先将 PRESET_CTRL0 的对应位（bit12）置为 0，然后再置为 1。

6.14.4 功能描述

在键寄存器(IWDG_KR)中写入 0xCCCC，开始启用独立看门狗；此时计数器开始从其复位值 0xFFF 递减计数。当计数器计数到末尾 0x000 时，会产生一个复位信号(IWDG_RESET)。 无论何时，只要在键寄存器 IWDG_KR 中写入 0xAAAA，IWDG_RLR 中的值就会被重新加载到计数器，从而避免产生看门狗复位 。

6.14.5 寄存器描述

寄存器	偏移量	R/W	描述	复位后的值
IWDG_KR	0x000	R/W	键寄存器	**0x**0
IWDG_PR	0x004	R/W	预分频寄存器	**0x**0
IWDG_RLR	0x008	R/W	重装载寄存器	**0xFFF**
IWDG_SR	0x00C	R	状态寄存器	**0x0**
IWDG_WINR	0x010	R/W	窗口寄存器	**0xFFF**

6.14.5.1 IWDG_KR 寄存器

寄存器	偏移量	R/W	描述	复位值
IWDG_KR	0x000	R/W	键寄存器	**0x**0

31	30	29	28	27	26	25	24
保留							
23	22	21	20	19	18	17	16
保留							
15	14	13	12	11	10	9	8
KEY[15:8]							

7	6	5	4	3	2	1	0
KEY[7:0]							

Bits	描述	
[31:16]	保留	保留
[15:0]	KEY	键值(只写寄存器，读出值为 0x0000) 软件必须以一定的间隔写入 0xAAAA，否则，当计数器为 0 时，看门狗会产生复位。 写入 0x5555 表示允许访问 IWDG_PR 和 IWDG_RLR 寄存器。写入 0xCCCC，启动看门狗工作(若选择了硬件看门狗则不受此命令字限制)。

6.14.5.2 IWDG_PR 寄存器

寄存器	偏移量	R/W	描述	复位值
IWDG_PR	0x004	R/W	预分频寄存器	**0x**0

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

Bits	描述	
[31:3]	保留	保留

[2:0]	PR	预分频因子 (Prescaler divider) 这些位具有写保护设置。通过设置这些位来选择计数器时钟的预分频因子。要改变预分频因子，IWDG_SR 寄存器的 PVU 位必须为 0。 000: 预分频因子=4 100: 预分频因子=64 001: 预分频因子=8 101: 预分频因子=128 010: 预分频因子=16 110: 预分频因子=256 011: 预分频因子=32 111: 预分频因子=256 注意：对此寄存器进行读操作，将从 VDD 电压域返回预分频值。如果写操作正在进行，则读回的值可能是无效的。因此，只有当 IWDG_SR 寄存器的 PVU 位为 0 时，读出的值才有效。

6.14.5.3 IWDG_RLR 寄存器

寄存器	偏移量	R/W	描述	复位值
IWDG_RLR	0x008	R/W	重装载寄存器	**0xFFF**

31	30	29	28	27	26	25	24
保留							
23	22	21	20	19	18	17	16
保留							
15	14	13	12	11	10	9	8
保留				**RLR**			

7	6	5	4	3	2	1	0
RLR							

Bits	描述	
[31:12]	保留	保留
[11:0]	RLR	看门狗计数器重装载值 (Watchdog counter reload value) 这些位具有写保护功能。用于定义看门狗计数器的重装载值，每当向 IWDG_KR 寄存器写入 0xAAAA 时，重装载值会被传送到计数器中。随后计数器从这个值开始递减计数。看门狗超时周期可通过此重装载值和时钟预分频值来计算。 只有当 IWDG_SR 寄存器中的 RVU 位为 0 时，才能对此寄存器进行修改。 注：对此寄存器进行读操作，将从 VDD 电压域返回预分频值。如果写操作正在进行，则读回的值可能是无效的。因此，只有当 IWDG_SR 寄存器的 RVU 位为 0 时，读出的值才有效。

6.14.5.4 IWDG_SR 寄存器

寄存器	偏移量	R/W	描述	复位值
IWDG_SR	0x00C	R	状态寄存器	**0x0**

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

保留	WVU	RVU	PVU

Bits	描述	
[31:3]	保留	保留
[2]	WVU	看门狗计数器窗口值值更新 此位由硬件置'1'用来指示窗口值的更新正在进行中。当在 VDD 域中的重装载更新结束后，此位由硬件清'0'(最多需 5 个 40kHz 的 RC 周期)。窗口值只有在 RVU 位被清'0'后才可更新。
[1]	RVU	看门狗计数器重装载值更新 (Watchdog counter reload value update) 此位由硬件置'1'用来指示重装载值的更新正在进行中。当在 VDD 域中的重装载更新结束后，此位由硬件清'0'(最多需 5 个 40kHz 的 RC 周期)。重装载值只有在 RVU 位被清'0'后才可更新。
[0]	PVU	看门狗预分频值更新 (Watchdog prescaler value update) 此位由硬件置'1'用来指示预分频值的更新正在进行中。当在 VDD 域中的预分频值更新结束后，此位由硬件清'0'(最多需 5 个 40kHz 的 RC 周期)。预分频值只有在 PVU 位被清'0'后才可更新。

6.14.5.5 IWDG_WINR 寄存器

寄存器	偏移量	R/W	描述	复位值
IWDG_WINR	0x010	R/W	窗口寄存器	**0xFFF**

31	30	29	28	27	26	25	24
保留							
23	22	21	20	19	18	17	16

保留							
15	14	13	12	11	10	9	8
保留				IWDG_WINR			
7	6	5	4	3	2	1	0
IWDG_WINR							

Bits	描述	
[31:12]	保留	保留
[11:0]	IWDG_WINR	看门狗计数器窗口值。 该 bit 域包含窗口的上线值，用于和计数器做比较。

6.15 触摸按键控制器（TSC）

6.15.1 概述

MT32F005/006 系列微控制器的包含一个触摸传感控制器模块 TSC，触摸传感控制器提供了给任何应用增加电容式触摸传感功能的一个简单的解决方案。电容式触摸解决方案可以检测与电极接近的手指，从而避免了触摸按键设计中出现的电气直接接触现象。通过检测有手指(或者其它导体) 所引入的电容变化的方法被来判断是否存在触摸按键。

6.15.2 基本特性

触摸传感控制器包含下列主要功能：

- 内部 RC 时钟振荡检测原理；
- 支持多达 20 个电容传感通道；
- 全硬件管理检测，提供非常好的响应时间；
- 用户可以根据最终板级情况设置每个 IO 的触发阈值；
- 通道 I/O 可选择
- 兼容滑条，触摸键，线性和旋转的触摸传感

6.15.3 基本配置

使用以下系统寄存器来打开 TSC 模块：

- 时钟：TSC 模块的工作时钟默认是关闭的，使用前必须在 AHBCLKCTRL 寄存器中打开对应的控制位（bit13）。
- 复位：TSC 模块的复位默认是高，即处于正常工作状态，如果要对 TSC 模块复位，应先将 PRESET_CTRL 的对应位（bit13）置为 0，然后再置为 1。

6.15.4 功能描述

在需要使用 TSC 功能的产品中，上电后软件首先配置 TSC 进入扫描模式，硬件会自动扫描被使能的 IO 上的初始计数值（非触摸状态下），并将每个 IO 的初始计数值存储到 IO 计数寄存器（IOCCR0~IOCCR6）中，并在扫描完所有被使能的 IO 后自动使能中断。软件收到中断后，查询每个 IO 的初始值（非触摸状态下），并计算出一个阈值写回到对应到 IO 计数寄存器（IOCCR0~IOCCR6）中，再配置 TSC 模块退出扫描模式，进入正常轮询模式；硬件自动轮询所有被使能的通道，自动读取每个通道的计数值，并和寄存器中的阈值做对比：如果大于阈值时，硬件会自动设置状态寄存器中对应的状态位，并在所有被使能的通道轮询结束后，触发中断（如果中断被使能的话）。软件收到中断后查询状态寄存器 HW_TSC_STAT，即可得到哪些 IO 被触摸了。

6.15.4.1 扫描模式

寄存器 HW_TSC_CR.SCAN_ENABLE 被写 1 后，TSC 模块进入扫描模式， 硬件会自动扫描被使能的 IO 上的初始计数值（非触摸状态下），并讲每个 IO 的初始计数值存储到 IO 计数寄存器（IOCCR0~IOCCR6）中，并在扫描完所有被使能的 IO 后自动使能中断。

6.15.4.2 正常轮询模式

寄存器 HW_TSC_CR.CON_MODE 被写 1 后，TSC 模块进入扫描模式，硬件自动轮询所有被使能的通道，自动读取每个通道的计数值，并和寄存器中的阈值做对比：如果大于阈值时，硬件会自动设置状态寄存器中对应的状态位，并在所以被使能的通道轮询结束后，触发中断（如果中断被使能的话）。软件收到中断后查询状态寄存器 HW_TSC_STAT，即可得到哪些 IO 被触摸了。

6.15.5 寄存器描述

R：只读，W：只写，R/W：可读写

寄存器	偏移量	R/W	描述	复位后的值
TSC 的基地址： TSC_BA = 0x4000_1800				
HW_TSC_CR	TSC_BA+0x00	R/W	TSC 控制寄存器	0x0000_0000
HW_TSC_ENABLE	TSC_BA+0x04	R/W	TSC 使能寄存器	0x0000_0000
HW_TSC_IER	TSC_BA+0x08	R/W	中断使能寄存器	0x0000_0000
HW_TSC_ICR	TSC_BA+0x0C	R/W	中断清零寄存器	0x0000_0000
HW_TSC_ISR	TSC_BA+0x10	R	中断状态寄存器	0x0000_0000
HW_TSC_STAT	TSC_BA+0x14	R	触摸状态寄存器	0x0000_0000
HW_TSC_OVF_STAT	TSC_BA+0x18	R	溢出标志寄存器	0x0000_0000
HW_TSC_RANGE	TSC_BA+0x1C	R/W	检测范围寄存器	0x0000_0064
HW_TSC_IOCCR0	TSC_BA+0x30	R/W	IO 计数寄存器 0	0x0000_0000
HW_TSC_IOCCR1	TSC_BA+0x40	R/W	IO 计数寄存器 1	0x0000_0000
HW_TSC_IOCCR2	TSC_BA+0x50	R/W	IO 计数寄存器 2	0x0000_0000
HW_TSC_IOCCR3	TSC_BA+0x60	R/W	IO 计数寄存器 3	0x0000_0000
HW_TSC_IOCCR4	TSC_BA+0x70	R/W	IO 计数寄存器 4	0x0000_0000
HW_TSC_IOCCR5	TSC_BA+0x80	R/W	IO 计数寄存器 5	0x0000_0000
HW_TSC_IOCCR6	TSC_BA+0x90	R/W	IO 计数寄存器 6	0x0000_0000
HW_TSC_IOCCR7	TSC_BA+0xa0	R/W	IO 计数寄存器 7	0x0000_0000
HW_TSC_IOCCR8	TSC_BA+0xB0	R/W	IO 计数寄存器 8	0x0000_0000
HW_TSC_IOCCR9	TSC_BA+0xc0	R/W	IO 计数寄存器 9	0x0000_0000

6.15.5.1 HW_TSC_CR 寄存器

Bits	描述	
[31:28]	保留	保留

[27:24]	ANA_PREDIV	**模拟时钟分频控制** 分频系数为：ANA_PREDIV+1，该值越大，计数寄存器中的计数值越大；
[23:20]	BIAS_ADJ	**TSC 模块模拟内部偏置的调节** 主要是为了调节输出 clk 的频率，默认 0000，值越大，频率越高。默认模拟部分送出时钟大概 200KHz 左右。
[24:17]	**保留**	**保留**
[16]	ANA_ENABLE	模拟部分使能信号
[15:12]	DIG_PREDIV	**数字部分计数分频控制** 分频系数为：DIG_PREDIV+1；该值越大，计数寄存器中的计数值越小；
[11:9]	**保留**	**保留**
[8]	CON_MODE	**正常轮询模式使能位** 该 bit 使能后，硬件自动轮询所有被使能的通道，自动读取每个通道的计数值，并和寄存器中的阈值做对比。大于阈值时，硬件会自动设置对应的标志位，并在所以被使能的通道轮询完毕后，触发中断（如果中断被使能的话）。
[7:5]	**保留**	**保留**
[4]	SCAN_ENABLE	**扫描模式使能位** 使能后硬件自动扫描所有被使能的通道，并将每个通道的计数值写到寄存器中，供软件查询。所有被使能的通道扫描完毕后会出发中断（如果中断被使能的话）。
[3:2]	**保留**	**保留**
[1:0]	DISCARD	**丢弃控制寄存域** 数字部分计数器前，丢弃的模拟时钟周期数，CON_MODE 为 0 时，每个通道都会丢弃； CON_MODE 为 1 时，只有第一个通道会丢弃；

6.15.5.2 HW_TSC_ENABLE 寄存器

Bits	描述	
[31:20]	保留	保留
[24]	ENABLE	在使用 TSC 模块时，该 bit 位须置为 1
[23:20]	保留	保留
[19:0]	CHANNLE_ENABLE	20 个触摸按键只能位，从低到高分别对应 [0:7]对应 PAD0_0 ~ PAD0_7 [8:15]对应 PAD1_0 ~ PAD1_7 [16:18]对应 PAD2_0 ~ PAD2_3

6.15.5.3 HW_TSC_IER 寄存器

Bits	描述	
[31:1]	保留	保留
[0]	IER	写 1 使能中断

6.15.5.4 HW_TSC_ICR 寄存器

Bits	描述	
[31:1]	保留	保留
[0]	ICR	写 1 清除中断，该 bit 位硬件自清零

6.15.5.5 HW_TSC_ISR 寄存器

Bits	描述

[31:1]	保留	保留
[0]	ISR	中断状态寄存器

6.15.5.6 HW_TSC_STAT 寄存器

Bits	描述	
[31:24]	保留	保留
[19:0]	STAT	**触摸状态寄存器** 每个 bit 对应一个 IO，非扫描模式下，该 bit 置 1，代表对应的 IO 上有触摸动作。

6.15.5.7 HW_TSC_OVF_STAT 寄存器

Bits	描述	
[31:24]	保留	保留
[19:0]	OVF_STAT	**溢出标志寄存器** 每个 bit 对应一个 IO，非扫描模式下，该 bit 置 1，代表对应的 IO 上计数溢出。

6.15.5.8 HW_TSC_RANGE 寄存器

Bits	描述	
[31:10]	保留	保留
[9:0]	Range	**触摸有效计数范围** IO_NUM[x]配置的是计数比较的下限，（IO_NUM[x] + RANGE）是计数比较的上限，只有在此范围内，才会触发触摸中断；

6.15.5.9 HW_TSC_IOCCR0~ HW_TSC_IOCCR9 寄存器

Bits	描述	
[31]	OVF[x+1]	通道 x+1 的计数溢出状态
[30:16]	IO_NUM[x+1]	扫描模式下，IO[x+1]的对应计数值;，用户读取该 bit 域得到非触摸状态下的 IO 计数值，然后根据这个计数值算出一个触发阈值，用户将触发阈值再次写到该寄存器中。在非扫描模式下，硬件会自动检查对应 IO 的计数值，如果大于阈值，则代码有触发动作，状态被记录到 HW_TSC_STAT 寄存器中，并在所有 IO 检测完毕后触发中断。
[15]	OVF[x]	通道 X 的计数溢出状态
[14:0]	IO_NUM[x]	扫描模式下，IO[x]的对应计数值;，用户读取该 bit 域得到非触摸状态下的 IO 计数值，然后根据这个计数值算出一个触发阈值，用户将触发阈值再次写到该寄存器中。在非扫描模式下，硬件会自动检查对应 IO 的计数值，如果大于阈值，则代码有触发动作，状态被记录到 HW_TSC_STAT 寄存器中，并在所有 IO 检测完毕后触发中断。

6.16 简单 TIMER

6.16.1 概述

MT006 系列芯片包含一个精简的 TIMER 模块，内部有两路独立的 32 位倒数计时的计数器，当计数到 0 时会产生中断。

6.16.2 特性

- 2 路 32bit 计数器
- 计数器时钟源可配置
- 计数到 0 时会产生中断
- 计数到 0 或者重新使能后会自动加载倒计时计数的初始值

6.16.3 基本配置

计数器的计数时钟源来自 Sysconfig 模块的 OUTCLK，可选择不同的时钟源并进行预分频。计数时钟频率不要超过 PCLK 频率的一半。

STIMER 的模块时钟为 PCLK，通过 AHBCLKCTRL 寄存器的 bit[17]来使能。

开始计时前先将计数器 ENABLE 位置 0，配置好对应的 LOADCNT 寄存器后再将计数器的 ENABLE 位置 1，即可开始进行倒计时。

6.16.4 寄存器映射

R: 只读, W: 只写, R/W: 可读写

寄存器	偏移量	R/W	描述	复位后的值
STIMER 的基地址： **STIMER_BA = 0x4000_D000**				
LOADCNT0 LOADCNT1	STIMER_BA+0x00 STIMER_BA+0x14	R/W	计数器0/1初始值寄存器	0x0000_ffff

CURRVAL0 CURRVAL1	STIMER_BA +0x04 STIMER_BA+0x18	R	计数器0/1当前值寄存器	0x0000_0000
CTRL0 CTRL1	STIMER_BA+0x08 STIMER_BA+0x1C	R/W	计数器0/1控制寄存器	0x0000_0000
CLR_IRQ0 CLR_IRQ1	STIMER_BA+0x0C STIMER_BA+0x20	R	计数器0/1中断清除寄存器	0x0000_0000
STAT_IRQ0 STAT_IRQ1	STIMER_BA+0x10 STIMER_BA+0x24	R	计数器0/1当前中断状态寄存器	0x0000_0000
STAT_STIMER	STIMER_BA+0xA0	R	简单计数器模块中断状态寄存器	0x0000_0000
CLR_IRQ	STIMER_BA+0xA4	R	简单计数器模块中断清除寄存器	0x0000_0000
RAW_IRQ_STAT	STIMER_BA+0xA8	R	简单计数器模块原始中断状态寄存器	0x0000_0000

6.16.4.1 LOADCNT0/1 寄存器

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

Bits	描述	
[31:0]	CNT	计数器开始计数时的初始值

6.16.4.2 CURRVAL0/1 寄存器

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

Bits	描述	
[31:0]	**VAL**	**计数器的当前计数值**

6.16.4.3 CTRL0/1 寄存器

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

Bits	描述	
[31:3]	保留	保留
[2]	IRQ_MASK	0：不屏蔽计数器的中断

		1：屏蔽计数器的中断
[1]	MODE	0：Free 模式（每次重新计数时自动选择 0xFFFFFFFF 做为初始值） 1：User 模式（每次重新计数时加载 LoadCNT 寄存器的值做为初始值）
[0]	ENABLE	0：计数器不使能 1：计数器使能

6.16.4.4 CLR_IRQ0/1 寄存器

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

Bits	描述	
[31:1]	保留	保留
[0]	CLR	读这位返回 0，并清除计数器的中断

6.16.4.5 STAT_IRQ0/1 寄存器

31	30	29	28	27	26	25	24
保留							

23	22	21	20	19	18	17	16
保留							
15	14	13	12	11	10	9	8
保留							
7	6	5	4	3	2	1	0
保留							STAT

Bits	描述	
[31:1]	保留	保留
[0]	STAT	计数器的中断状态位

6.16.4.6 STAT_STIMER 寄存器

31	30	29	28	27	26	25	24
保留							
23	22	21	20	19	18	17	16
保留							
15	14	13	12	11	10	9	8
保留							
7	6	5	4	3	2	1	0
保留						STAT1	STAT0

Bits	描述	
[31:2]	保留	保留
[1]	STAT1	计数器 1 的中断状态位
[0]	STAT0	计数器 0 的中断状态位

6.16.4.7 CLR_IRQ 寄存器

31	30	29	28	27	26	25	24
保留							
23	22	21	20	19	18	17	16
保留							
15	14	13	12	11	10	9	8
保留							
7	6	5	4	3	2	1	0
保留						STAT1	STAT0

Bits	描述	
[31:2]	保留	保留
[1]	STAT1	读这个寄存器清除所有有效的中断
[0]	STAT0	读这个寄存器清除所有有效的中断

6.16.4.8 RAW_IRQ_STAT 寄存器

31	30	29	28	27	26	25	24
保留							
23	22	21	20	19	18	17	16
保留							
15	14	13	12	11	10	9	8
保留							
7	6	5	4	3	2	1	0
保留						STAT1	STAT0

Bits	描述	
[31:2]	保留	保留
[1]	STAT1	计数器 1 的未屏蔽中断状态
[0]	STAT0	计数器 0 的未屏蔽中断状态

6.17 USB 全速接口

6.17.1 概述

MT006 系列芯片包括一个 USB 2.0 FS 控制器，工作在 device 模式，主要用于数据加载或者调试应用等。

6.17.2 特性

- 可以工作在 device 模式
- 符合 USB2.0 全速设备的技术规范
- 可配置 1 到 4 个 USB 端点
- 支持和全速(full-speed)或者低速设备(low-speed)通信
- CRC(循环冗余校验)生成/校验，反向不归零(NRZI)编码/解码和位填充
- 支持 USB 挂起/恢复操作
- 每个端点 Txfifo 和 Rxfifo 大小都是独立的固定 64byte

6.17.3 基本配置

USB 模块需要两个时钟源 PCLK 和 USB48MCLK，其中 PCLK 时钟源由 Sysconfig 模块的 AHBCLKCTRL 寄存器的 bit19 控制，在使用 USB 模块之前必须先打开这一位。USB48MCLK 由 Sysconfig 模块的 USBCLKDIV 寄存器控制，该时钟的是时钟源为 SYSPLLCLK，因此要使用 USB 功能，系统 PLL 时钟必须配置为 96MHz 或者 48MHz，同时 USBCLKDIV 寄存器配置为 2 或者 1。

USB 模块的软复位由 Sysconfig 模块的 PRESETCTRL 寄存器的 bit19 控制，该寄存器置 0 可复位 USB 模块，置 1 使模块正常工作。

6.17.4 功能描述

USB 模块为 PC 主机和微控制器所实现的功能之间提供了符合 USB 规范的通信连接。USB 模块最多可用于 4 个双向端点。USB 模块同 PC 主机通信，根据 USB 规范实现令牌分组的检测，数据发送/接收的处理，和握手分组的处理。整个传输的格式由硬件完成，其中包括 CRC 的生成和校验。

当 USB 模块识别出一个有效的功能/端点的令牌分组时，(如果需要传输数据并且端点已配置)随之发生相关的数据传输。在所有的数据传输完成后，如果需要，则根据传输的方向，发送或接收适当的握手分组。

在数据传输结束时，USB 模块将触发与端点相关的中断，通过读状态寄存器和/或者利用不同的中断处理程序，微控制器可以确定：

- 哪个端点需要得到服务。
- 产生如位填充、格式、CRC、协议、缺失 ACK、缓冲区溢出/缓冲区未满等错误时，正在进行的是哪种类型的传输。

在任何不需要使用 USB 模块的时候，通过写控制寄存器总可以使 USB 模块置于低功耗模式(SUSPEND 模式)。在这种模式下，会有效降低 USB 部分功耗，同时 USB 时钟也会减慢或停止。通过对 USB 线上数据传输的检测，可以在低功耗模式下唤醒 USB 模块。

6.17.5 寄存器配置

USB 寄存器如下表所示。分为通用寄存器和索引寄存器两大部分，可通过设置 index 寄存器来访问不同 endpoint 的索引寄存器。

表：通用寄存器列表

地址	名称	说明
00	Faddr	功能地址寄存器
01	Power	电源管理寄存器

02	IntrTx1	端点 0 的中断寄存器，Tx 端点 1～3
03	IntrTx2	保留
04	IntrRx1	RX 端点 1~3 的中断寄存器
05	IntrRx2	保留
06	IntrUSB	通用 USB 中断的中断寄存器
07	IntrTx1E	中断使能寄存器，对应 IntrTx1
08	IntrTx2E	保留
09	IntrRx1E	中断使能寄存器，对应 IntrRx1
0A	IntrRx2E	保留
0B	IntrUSBE	中断使能寄存器，对应 IntrUSB
0C	Frame1	帧号 bit7:0
0D	Frame2	帧号 bit15:8
0E	Index	索引寄存器用于选择 endpoint 的状态和控制寄存器
0F	--	--

表：被索引寄存器列表(device 模式)

地址	名称	说明
10	TxMaxP	外设 Tx 端点的最大包的大小。(仅在设置索引寄存器来选择端点 1～2 时)
11	CSR0	端点 0 的控制/状态寄存器。（索引寄存器选择端点 0 时）
	TxCSR1	外设 Tx 端点的控制/状态寄存器。（索引寄存器选择端点 1～2 时）
12	TxCSR2	外设 Tx 端点的控制/状态寄存器。（索引寄存器选择端点 1～2 时）
13	RxMaxP	外设 Rx 端点的最大包大小。（仅在索引寄存器选择端点 1～2 时）
14	RxCSR1	外设 Rx 端点的控制状态寄存器。（仅在索引寄存器选择端点 1～2 时）
15	RxCSR2	外设 Rx 端点的控制状态寄存器。（仅在索引寄存器选择端点 1～2 时）
16	Count0	从外设 Rx 端点 0 的 FIF0 读取的字节数。（索引寄存器选择端点 0 时）
	RxCount1	从外设 Rx 端点 FIF0 读取的字节数（低位）。（索引寄存器选择端点 1～2 时）
17	RxCount2	从外设 Rx 端点 FIF0 读取的字节数（高位）。（索引寄存器选择端点 1～2 时）

18-1B	-	保留
1C-1F	-	保留

表：FIFO 列表

地址	名称	说明
20	FIFO0	端点 0 的 FIFO 地址
24	FIFO1	端点 1 的 FIFO 地址
28	FIFO2	端点 2 的 FIFO 地址
2C	FIFO3	端点 3 的 FIFO 地址

通用寄存器：

6.17.5.1 FADDR

Faddr 是一个 8 位的寄存器，应当使用传输时外设部分的 7 位地址来写入。当 USB 核用于外设模式（DevCtl.D2=0）时，该寄存器应当使用通过 SET_ADDRESS 指令接收到的地址写入，将用于并行的标记包内对功能地址进行解码。

注意：当USB核在外设模式时，该寄存器仅用于传输操作。

FADDR 位域说明

位	标签	读写	复位	说明
7	RSRVD	r	0x0	未使用，总是返回 0
6:0	Function Addr	rw	0x00	功能地址

6.17.5.2 POWER

Power 是一个 8 位寄存器，用于控制休眠和唤醒信号，以及 USB 核的基本操作。

POWER 位域说明

位	标签	读写	复位	说明
7	ISO Update	rw/-	0x0	CPU 设置该位之后，USE 核将等待 SOF 标识（它是在设置 TxPktRdy 之后，发送数据包之前产生）。如果在 SOF 信号之前收到 IN 信号，一个长度为 0 的数据包将被发送。 注意：此位只在外设模式下才有效。另外，该位只影响端点执行等时传输时。
6:4	RSVD	r/r	0x0	保留
3	Reset	r/rw	0x0	当 Reset 信号出现在总线时，该位被设置。 注意：在主机模式下，CPU可对该位读写；而在外设模式下，它却是只读的。
2	Resume	rw/rw	0x0	当功能处于休眠模式时，CPU 设置该位以生成唤醒信号。CPU 应该在停止唤醒信号 10ms（最大 15ms）后清除该位。在主机模式下，USB 核休眠时唤醒信号被检测到，该位自动被置位。
1	Suspend Mode	r/set	0x0	在主机模式下，CPU 设置该位进入休眠模式；在外设模式下，设置该位进入休眠模式。在读取中断寄存器，或者设置了上述的 Resume 位时，该位被清除。
0	Enable SuspendM	rw/rw	0x0	CPU 设置该位激活 SUSPENDM 输出。

6.17.5.3 INTRTX

IntrTx 是一个 8 位只读寄存器，用来指示端点 0 和 Tx 端点 1～2 的哪些中断是活动的。

注意：与端点相关的位，如果没有配置，将一直返回0。另外，当该寄存器被读取时，所有活动的中断将被清除。

IntrTx 位域说明

位	标签	读写	复位	说明
7:4	RSRVD	R	0x0	未使用，总是返回 0
3	EP3	R	0x0	Tx 端点 3 中断
2	EP2	R	0x0	Tx 端点 2 中断
1	EP1	R	0x0	Tx 端点 1 中断
0	EP0	R	0x0	端点 0 中断

6.17.5.4 INTRRX

IntrTx 是一个 8 位只读寄存器，用来指示 Rx 端点 1～2 的哪些中断是活动的。

注意：与端点相关的位，如果没有配置，将一直返回0。另外，当该寄存器被读取时，所有活动的中断将被清除。

IntrTx 位域说明

位	标签	读写	复位	说明
7:4	RSRVD	R	0x0	未使用，总是返回 0
3	EP2	R	0x0	Rx 端点 3 中断
2	EP2	R	0x0	Rx 端点 2 中断
1	EP1	R	0x0	Rx 端点 1 中断
0	EP0	R	0x0	端点 0 中断

6.17.5.5 INTRTXE

IntrTxE 是一个 8 位寄存器，它含有 IntrTx 中断的中断使能位。若某位被置 1，MC_NINT 将被告知：IntrTx 寄存器相应中断被设置；若某位被清 0，IntrTx 中断仍然会被设置，但不会通知 MC_NINT。复位时，设计包含的端点 0、Tx 端点的相关位被置 1，而其他位被清 0。

注意：与端点相关的位，如果没有配置，将一直返回0。

IntrTxE 位域说明

位	标签	读写	复位	说明
7:4	RSRVD	R	0x0	未使用，总是返回 0
3	EP2	R	0x0	Tx 端点 3 中断使能
2	EP2	R	0x0	Tx 端点 2 中断使能
1	EP1	R	0x0	Tx 端点 1 中断使能
0	EP0	R	0x0	端点 0 中断

6.17.5.6 INTRRXE

IntrRxE 是一个 8 位寄存器，它含有 IntrRx 中断的中断使能位。若某位被置 1，MC_NINT 将被告知 IntrTx 寄存器相应中断被设置；若某位被清 0，IntrRx 中断仍然会被设置，但不会通知 MC_NINT。复位时，设计包含的端点 0、Tx 端点的相关位被置 1，而其他位被清 0。

注意：与端点相关的位，如果没有配置，将一直返回0。

IntrRxE 位域说明

位	标签	读写	复位	说明
7:4	RSRVD	R	0x0	未使用，总是返回 0
3	EP2	R	0x0	Rx 端点 3 中断使能
2	EP2	R	0x0	Rx 端点 2 中断使能
1	EP1	R	0x0	Rx 端点 1 中断使能
0	EP0	R	0x0	端点 0 中断

6.17.5.7 INTRUSB

IntrUSB 是一个 8 位只读寄存器，它指明了哪些 USB 中断被激活。 当寄存器被读取时，所有的活动中断被清除。

IntrUSB 位域说明

位	标签	读写	复位	说明
7：4	保留	r	0x0	保留
3	SOF	r	0x0	SOF 中断
2	Reset/	r	0x1	Reset 中断
1	Resume	r	0x1	Resume 中断
0	Suspend	r	0x0	Suspend 中断

6.17.5.8 INTRUSBE

IntrUSBE 是一个 8 位寄存器，它为 IntrUSB 的中断提供中断使能位。

IntrUSBE 位域说明

位	标签	读写	复位	说明
7：4	保留	r	0x0	保留
3	SOF	r	0x0	SOF 中断使能
2	Reset/	r	0x1	Reset 中断使能，外设模式下
1	Resume	r	0x1	Resume 中断使能
0	Suspend	r	0x0	Suspend 中断使能

6.17.5.9 FRAME1

Frame1 和 Frame2 组成一个 16 位只读寄存器，它保存的是最后接收到的帧序号。

FRAME 位域说明

位	标签	读写	复位	说明
7:0	FRAME	r	0x0	帧序号低 8 位

6.17.5.10 FRAME2

Frame1 和 Frame2 组成一个 16 位只读寄存器，它保存的是最后接收到的帧序号。

FRAME 位域说明

位	标签	读写	复位	说明
7:0	FRAME	r	0x0	帧序号高 8 位

6.17.5.11 INDEX

每个 Tx 和 Rx 端点有自己的一组位于 100h-1FFh 之间的控制/状态寄存器。另外，还有一组位于 10h-19h 的 Tx/Rx 控制/状态寄存器。Index 是一个 4 位的寄存器，它确定了哪些端点的控制/状态寄存器被访问。

INDEX 位域说明

位	标签	读写	复位	说明
7:4	RSRVD	r	0x0	未使用
3:0	INDEX	rw	0x0	选择的端点

在访问位于 10h-19h 的端点控制/状态寄存器之前，端点号应当被写入到 Index 寄存器，以确保正确的控制/状态寄存器出现在内存映射中。

索引寄存器：

6.17.5.12 CSR0

CSR0 是一个 8 位寄存器，用于对端点 0 进行控制和状态监视。注意：该寄存器的行为依赖于 USB 核是一个外设还是主机。用户应该注意到：当寄存器被读取时，返回的值反应了附加的状态（例如：写入到寄存器的结果）。

CSR0 地址: 11h（Index 寄存器设置为 0）

device 模式下 CSR0 位域说明

位	标签	读写	复位	说明
7	ServicedSetupEnd	set	0x0	CPU 对该位置 1 是为了清除 SetupEnd 位。该位会自动清除。
6	ServicedRxPktRdy	set	0x0	CPU 对该位置 1 是为了清除 RxPktRdy 位。该位会自动清除。
5	SendStall	set	0x0	CPU 对该位置 1 是为了终止当前传输。STALL 握手信号将被送出，且该位自动被清除。
4	SetupEnd	r	0x0	控制传输结束时，DataEnd 位被设置前，该位被置位。中断将会发生，FIFO 此时被刷新。CPU 是通过将 ServicedSetupEnd 位置 1 来清除该位。
3	DataEnd	set	0x0	CPU 在以下情况下设置该位： 1. 为最后的数据包设置 TxPktRdy 时。 2. 卸载最后的数据包后，清除 RxPktRdy 时。 3. 为一个长度为 0 的数据包设置 TxPktRdy 时。 该位会自动被清 0。
2	SentStall	r/clear	0x0	当 STALL 握手信号被发送时该位被设置。CPU 应当清除该位。
1	TxPktRdy	r/set	0x0	在加载一个数据包到 FIFO 后，CPU 设置该位。当一个数据包被传送时，它会自动被清除。此时会产生中断。（允许时）
0	RxPktRdy	r	0x0	当一个接收到一个数据包时设置该位。该位被设置时中断会发生。CPU 通过设置 CPU ServicedRxPktRdy 位来清除该位。

6.17.5.13 COUNT0

Count0 是一个 7 位的只读寄存器，用来指示端点 0 FIFO 接收到的数据字节数。当 FIFO 改变时，返回值也会变化。该位只在 RxPktRdy (CSR0.D0)被设置后返回值才是合法的。

地址: 16h（Index 寄存器设置为 0）

COUNT0 位域说明

位	标签	读写	复位	说明
7	RSVRD	r	0x0	未使用，读取时返回 0。
6:0	Endpoint 0 Rx Count	r	0x0	端点 0 Rx 计数值

6.17.5.14 TXMAXP

TxMaxP 寄存器定义了在单个操作中，通过选择的 Tx 端点可以传输的最大数目。除端点 0 外，每个 Tx 端点都有一个 TxMaxP 寄存器。

地址：10h

位	标签	读写	复位	说明
7:0	Maximum Packet Size/transaction	r	0x0	选择的 frames 数目是 2^{m-1}（m 是该寄存器设置的值，合法值是 2～16）。需注意 FIFO 的大小。

6.17.5.15 TXCSR1

TxCSR1 提供了通过当前选择的 Tx 端点的传输控制和状态位。每一个配置的 Tx 端点有一个 TxCSR 寄存器（不包括端点 0）注意：寄存器的含义依赖于 USB 核是作为一个外设还是主机。用户应该注意到：当寄存器被读取时，返回的值反应了达到的状态。例如：写入到寄存器的结果。

地址：11h

device 模式下 TXCSR 位域说明

位	标签	读写	复位	说明
7	RSRVD	r	0x0	未使用，总是返回 0
6	ClrDataTog	set	0x0	CPU 对该位置 1 来将端点 data toggle 复位到 0。
5	SentStall	r/clear	0x0	当 STALL 握手信号被发送，该位置 1。FIFO 被刷新， TxPktRdy 位被清除（参看后续部分）。CPU 应当清除该位。
4	SendStall	rw	0x0	CPU 对该位置 1 来发布一个 STALL 握手信号给 IN 指令。CPU 清除该位来终止 stall 条件 。 注意：端点在用于等时传输时，该位无效。
3	FlushFIFO	set	0x0	CPU 对该位置 1 来刷新最近来自端点 Tx FIFO 的数据包。FIFO 指针被复位，TxPktRdy 位（后面）被清除，且产生了一个中断。也许与 TxPktRdy 同时设置会放弃当前加载到 FIFO 的数据包。注意：FlushFIFO 只能在 TxPktRdy 被设置后使用。其他情况下，它将导致数据被破坏。另外，如果 FIFO 是双缓冲，FlushFIFO 也许需要设置两次来完全清除 FIFO。
2	UnderRun	r/clear	0x0	当 TxPktRdy 没有设置，如果收到 IN 指令，USB 设置该位。CPU 应当清除该位。
1	FIFONotEmpty	r/clear	0x0	至少有一个数据包在 Tx FIFO 中时，USB 设置该位。
0	TxPktRdy	r/set	0x0	在加载数据包到 FIFO 后，CPU 设置该位。当

				数据包已传送，该位自动被清除。此时会产生一个中断（允许的话）。在装载双缓冲 FIFO 的第 2 个数据包时，TxPktRdy 也会自动被清除。

6.17.5.16 TXCSR2

TxCSR2 提供了通过当前选择的 Tx 端点的传输控制和状态位。每一个配置的 Tx 端点有一个 TxCSR 寄存器（不包括端点 0）注意：寄存器的含义依赖于 USB 核是作为一个外设还是主机。用户应该注意到：当寄存器被读取时，返回的值反应了达到的状态。例如：写入到寄存器的结果。

地址：12h

TXCSR2 位域说明

位	标签	读写	复位	说明
7	AutoSet	rw	0x0	如果 CPU 设置了该位，当最大数据包的大小（TxMaxP 中存储该值）数据被加载到 Tx FIFO，TxPktRdy 将自动被设置。如果小于最大数据包大小的数据被加载，TxPktRdy 不得不手动设置。 注意：对于高带宽的等时/中断端点，不应该设置此位。
6	ISO	rw	0x0	CPU 置 1 表示同步传输
5	Mode	rw	0x1	该位置 1 使端点方向为 Tx，清 0 则使之为 Rx。 注意：该位只在同样的FIFO同时用于Tx和Rx处理时才有效。

4	DMAReqEnab	rw	0x0	CPU 设置该位，使能请求 Tx 端点的 DMA。
3	FrcDataTog	rw	0x0	CPU 对该位置位，以强制端点的 data toggle 切换并且清除来自 FIFO 的数据包，而不管是否收到 ACK。它可以用于中断 Tx 端点，该端点可以用于等时端点的通信比率反馈。
2	DMAMode	rw	0x0	CPU 设置该位来选择 DMA 请求模式 1，清除该位来选择 DMA 请求模式 0。 注意：在清除前面的DMAReqEnab位之前或同一个周期内，该位不能被清除。
1：0	-	-	-	保留

6.17.5.17 RXMAXP

TxMaxP 寄存器定义了在单个操作中，通过选择的 Rx 端点可以传输的最大数目。除端点 0 外，每个 Rx 端点都有一个 RxMaxP 寄存器。

地址：13h

位	标签	读写	复位	说明
7:0	Maximum Packet Size/transaction	r	0x0	选择的 frames 数目是 2^{m-1}（m 是该寄存器设置的值，合法值是 2～16）。需注意 FIFO 的大小。

6.17.5.18 RXCSR1

RxCSR1 提供了通过当前选择的 Rx 端点的传输控制和状态位。每一个配置的 Rx 端点有一个 RxCSR1 寄存器（不包括端点 0）注意：寄存器的含义依赖于 USB 核是作为一个外设还是主机。用户应该注意到：当寄存器被读取时，返回

的值反应了达到的状态。例如：写入到寄存器的结果。

地址：14h

模式下 RXCSR1 位域说明

位	标签	读写	复位	说明
7	ClrDataTog	set	0x0	CPU 对该位置 1 来将端点 data toggle 复位到 0。
6	SentStall	r/clear	0x0	当 STALL 握手信号被发送，该位置 1。CPU 应当清除该位。
5	SendStall	rw	0x0	CPU 对该位置 1 来发布一个 STALL 握手信号给 IN 指令。CPU 应当清除该位。 注意：端点在用于等时传输时，该位无效。
4	FlushFIFO	set	0x0	CPU 对该位置 1 来刷新下一个从端点 Rx FIFO 读取的数据包。FIFO 指针被复位，RxPktRdy 位（后面）被清除。注意：FlushFIFO 只能在 RxPktRdy 被设置后使用。其他情况下，它将导致数据被破坏。另外，如果 FIFO 是双缓冲，FlushFIFO 也许需要设置两次来完全清除 FIFO。
3	DataError	r	0x0	如果数据包有一个 CRC 或者 bit-stuff 错误，RxPktRdy 被设置后，该位被置位。当 RxPktRdy 被清除，该位被清 0。 注意：该位只在端点运行在 ISO 模式下才有效。在批量模式下，它总是返回 0。
2	OverRun	r/clear	0x0	当 OUT 包未能加载到 Rx FIFO 中，该位被设置。CPU 应当清除该位。 注意：该位只在端点运行在ISO模式下才有

				效。在批量模式下，它总是返回0。
1	FIFOFULL	r/clear	0x0	当 Rx FIFO 不能加载更多包时，该位被置位。
0	RxPktRdy	r/set	0x0	在接收到数据包后，CPU 设置该位。当数据包从 Rx FIFO 卸载后，CPU 应当清除该位。该位被设置后，会产生一个中断。

6.17.5.19 RXCSR2

RxCSR2 提供了通过当前选择的 Rx 端点的传输控制和状态位。每一个配置的 Rx 端点有一个 RxCSR2 寄存器（不包括端点 0）

地址：15h

RXCSR2 位域说明

位	标签	读写	默认值	说明
7	AutoSet	rw	0x0	如果 CPU 设置了该位，当最大数据包的大小（TxMaxP 中存储该值）数据被加载到 Tx FIFO，TxPktRdy 将自动被设置。如果小于最大数据包大小的数据被加载，TxPktRdy 不得不手动设置。 注意：对于高带宽的等时/中断端点，不应该设置此位。
6	ISO	rw	0x0	CPU 设置此位，为了使 Tx 端点的等时传输有效，当使 Tx 端点的批量或中断传输有效时清除该位。
5	DMAReqEnab	rw	0x0	CPU 设置该位，使能请求 Tx 端点的 DMA。
4	DMAReqMode	rw	0x0	CPU 设置该位来选择 DMA 请求模式 1，清除该

				位来选择 DMA 请求模式 0。 注意：在清除前面的DMAReqEnab位之前或同一个周期内，该位不能被清除。
3:0	RSRVD	r	0x0	未使用，总是返回 0

6.17.5.20 RXCOUNT1

RxCount1 是一个只读寄存器，它保存了将从 Rx FIFO 顺序读取的数据包的字节数的低 8 位。如果数据包以多个批量包来传送时，给定的数目指的是合并的包。注意：当 FIFO 卸载后返回值会发生变化，返回值只在 RxPktRdy (RxCSR.D0)设置后有效。

地址: 16h

RXCOUNT1 位域说明

位	标签	读写	默认值	说明
7:0	Endpoint Rx Count	r	0x0	端点 Rx 的计数值，低 8 位

6.17.5.21 RXCOUNT2

RxCount2 是一个 3bit 只读寄存器。它保存了将从 Rx FIFO 顺序读取的数据包的字节数的高 3 位。

RXCOUNT2 位域说明

位	标签	读写	默认值	说明
2:0	Endpoint Rx Count	r	0x0	端点 Rx 的计数值，高 3 位

6.18 CRS 系统（时钟回馈系统）

6.18.1 概述

时钟回馈系统（CRS）是一种作用在内部时钟（IRC）上的一种先进的数字控制器，可以对时钟进行校准。通过比较可选同步源， CRS 可以评估振荡器的输出频率。通过测量频率误差值，CRS 可以自动校准振荡器。当然也可通过手动校准。

CRS 系统是适用于给 USB 外设提供精准的时钟。在 USB 被使用的情况下，同步信号源可来自 USB 总线上的 SOF 包，且 SOF 由 USB 主机每隔 1mS 发送而来。

同步信号同样可以由一个外部的引脚或者是直接由用户软件来提供。

6.18.2 特性

- 可选同步源（含可编程预分频器及极性选择）
  - 外部引脚
  - USB SOF 包接收
- 可以通过软件产生同步脉冲
- 可脱离 CPU，自动校准
- 手动控制选项启动收敛更快
- 带有自动误差捕获及数据加载的 16 位频率误差计数器
- 可编程有限的频率错误评估及状态报告
- 可屏蔽中断/事件
  - 期望同步（ESYNC）
  - 同步正确（SYNCOK）
  - 同步警告（SYNCWARN）
  - 同步错误或校准错误（ERR）

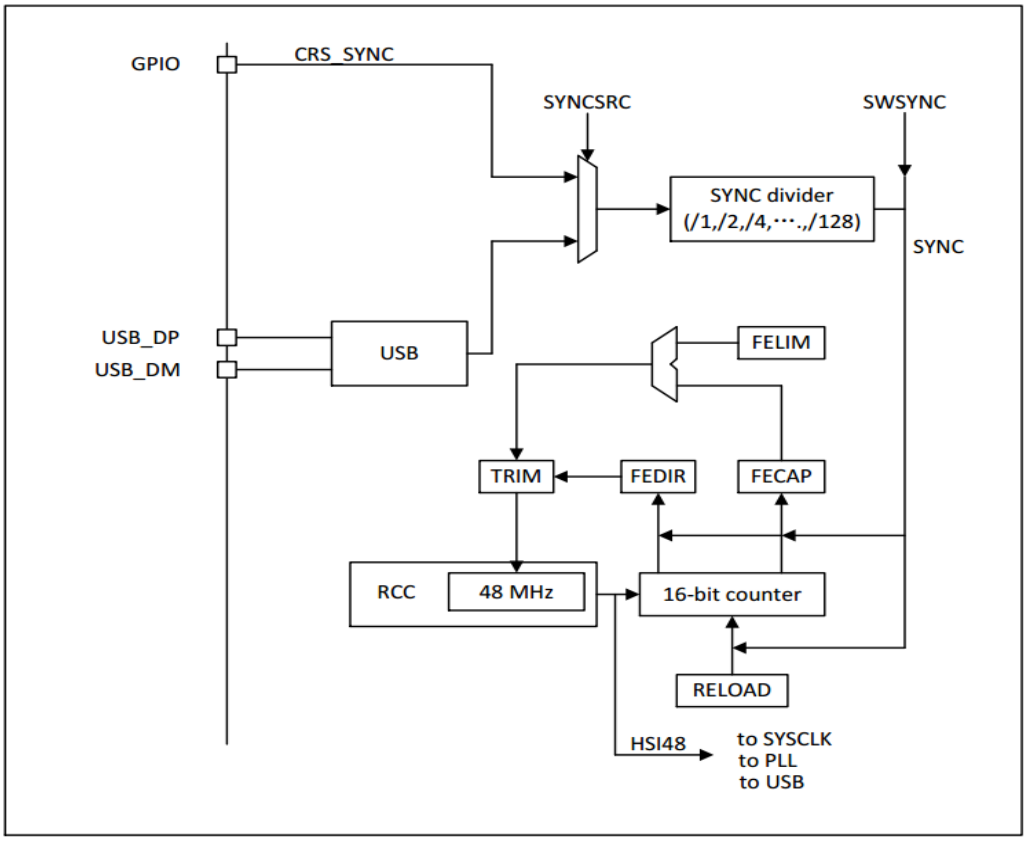
6.18.3 基本配置

CRS 模块的时钟源由 Sysconfig 模块的 AHBCLKCTRL 寄存器的 bit20 控制，在使用 CRS 模块之前必须先打开这一位。

CRS 模块的软复位由 Sysconfig 模块的 PRESETCTRL 寄存器的 bit20 控制，该寄存器置 0 可复位 CRS 模块，置 1 使模块正常工作。

6.18.4 功能描述

6.18.4.1 CRS 框图架构



6.18.4.2 同步输入

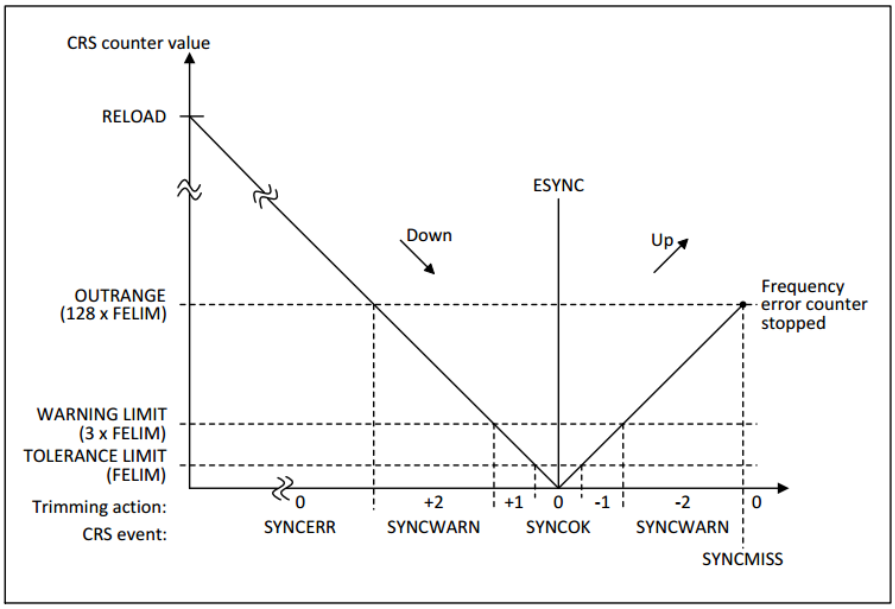
通过 CRS_CFGR 寄存器配置的 CRS 同步源可以来自外部 CRS_SYNC 引脚或 USB SOF 包。为满足同步输入源更好的稳健性，一个简单的通过 IRC12M 时钟采样的 2 级数字滤波是必不可少的，它可以过滤很多小错误。 信号源拥有一个可配置的极性，且可通过一个可编程的预分频器分频，以保证其处于一个合适的频率范围内（通常为 1KHz）。

更多 CRS 同步源配置信息，可参考及寄存器小节： CRS 配置寄存器（CRS_CFGR）。

6.18.4.3 频率误差测量

频率误差计数器是一个 16 位向上/下计数器，它在每次事件触发时，都会加载 RELOAD 值。它开始向下计数直到 0，触发 ESYNC（期望同步）事件。然后它开始向上计数到 OUTRANGE 的界限，并停下（如果没有接收到 SYNC 事件），并且产生一个 SYNCMISS 的事件。 OUTRANGE 的限制数值是由频率误差值（CRS_CFGR 寄存器的 FELIM 位）乘以 128 得到的。

当检测到同步事件，频率误差计数器的实际值及计数方向都被存储在 CRS_ISR 寄存器的 FECAP 和 FEDIR 位。在计数器向下计数阶段（到达‘0’之前）检测到同步事件，表示实际频率低于目标值（TRIM 值需递增）。反之，向上计数阶段检测到同步事件，表示实际频率高于目标值（TRIM 值需递减）。



6.18.4.4 频率误差计算及自动校准

频率误差评估是通过比较实测值和各个限制值之间的关系：

- 容忍界限（TOLERANCE LIMIT）：直接取自 CRS_CFGR 寄存器的 FELIM 位
- 警告界限（WARNING LIMIT）：定义为 3*FELIM 值
- 溢出边界（ERROR LIMIT）：定义为 128*FELIM 值

比较结果通常被用于生成状态标志，及控制自动校准（如果 CRS_CR 寄存器内的 AUTOTRIM 位被使能）：

- 当频率误差低于容忍界限，表明实际校准值位于理想范围内，不需要修正。
- 同步正确标志位（SYNCOK）被置位
- 自动校准（AUTOTRIM）模式下，不需要修正 TRIM 值
- 当频率误差低于警告界限且大于或等于容忍界限，表明需要些许校准，调节器只需调整 1 档就能达到理想的 TRIM 值。
- 同步正确标志位（SYNCOK）被置位
- 自动校准（AUTOTRIM）模式下 TRIM 数值每次调节一个档位
- 当误差计数大于或等于警告界限，但小于错误界限，表明需要较大校准并且下个同步周期将不会达到理想的 TRIM 值。
- 同步警告标志位（SYNCWARN）被置位
- 自动校准（AUTOTRIM）模式下 TRIM 数值每次调节二个档位
- 当误差计数大于或等于错误界限，表明频率误差超出可校准范围。当同步信号未清除或丢失时，也将发生上述情况（如： USB SOF 包丢失） 。
- 同步错误（SYNCERR）或同步丢失（SYNCMISS）标志位被置位
- 自动校准（AUTOTRIM）模式下 TRIM 数值不变

注意：如果实际 TRIM 值已接近边界，继续（自动）校准将导致 TRIM 值溢出，此时 TRIM 值将被锁定在边界，且校准溢出（TRIMOVF）标志位被置位。

自动校准（AUTOTRIM）模式下（配置 CRS_CR 寄存器 AUTOTRIM 位），TRIM 位仅由硬件配置且只读。

6.18.4.5 CRS 初始化及配置

加载（RELOAD）值

RELOAD 值得选定是根据目标频率和分频后的同步源频率比较得出的。RELOAD 值每次会减少一从而希望在零值得时候达到期望的频率。公式如下：

$$RELOAD = (f_{TARGET} / f_{SYNC}) - 1$$

重置后的 RELOAD 值一个匹配目标的 12MHz 的时钟和 1KHz 的同步信号频率（来自 USB 的 SOF 信号）。

FELIM 值

FELIM 的选择需要将 IRC 的特性以及它的典型的调节档位相结合。理想的数值对应于一半的调整步骤，表示为一系列 IRC 振荡器的计数。公式如下：

$$FELIM = (f_{TARGET} / f_{SYNC}) * STEP[\%] / 100\% / 2$$

为了获得更好的修正反馈，应该总是捕获到修正值最近的整数，如果不希望频繁的调整修正值，可以通过适当的增加 FELIM 数值重置的 FELIM 应该匹配（F_{TARGET}/F_{SYNC}） = 12000。

注意：一个错误的 RELOAD 和 FELIM 配置会导致一个不稳定的递减响应，并且没有硬件的保护。理想的运作模式需要适当的 RELOAD 值的设置（根据同步源的频率设置）而且 RELOAD 的值也是大于 FELIM 值的 128 倍。

6.18.4.6 CRS 中断

中断事件	事件标志	使能控制位	清除控制位
期望同步	ESYNCF	ESYNCIE	ESYNCIE
同步正确	SYNCOKF	SYNCOKIE	SYNCOK
同步警告	SYNCWARNF	SYNCWARNIE	SYNCWARNC
同步错误或校准错误（TRIMOVF, SYNCMISS, SYNCERR）	ERRF	ERRIE	ERRC

6.18.5 寄存器配置

6.18.5.1 CRS 控制寄存器（CRS_CR）

寄存器	偏移量	R/W	描述	复位值
CRS_CR	CRS_BA+0x00	R/W	**控制寄存器 0**	0x0000_8000

Bits	描述	
[31:16]	保留	保留
[15:8]	TRIM	**IRC校准值** 这些位给用户提供了可编写的修正值，用户可以根据一些可能影响到IRC频率的一些因数（电压、温度）来对TRIM值进行改写。默认值为0x80，这个值是修改挡位的中间值，更大的TRIM值意味着更高的输出频率 当AUTOTRIMEN被设置，这个域由硬件控制，且为只读。
[7]	SWSYNC	**产生软件同步行为** 这位由软件设置，产生一个软件SYNC事件，由硬件自动清除。 0：无操作 1：软件SYNC行为被触发（生成软件SYNC事件）

[6]	AUTOTRIMEN	**自动校准使能** 这个为根据测量两次同步事件之间的频率误差来计算 TRIM 值。如果 AUTOTRIMEN 位被使能，TRIM 位将被锁定为只读，TRIM 值每次可由硬件调整一个或者两个挡位（取决于测量频率误差值） 0：禁止自动校准，TRIM 位可以被用户修改 1：使能自动校准，TRIM 位只读且由硬件控制
[5]	CEN	**自动校准使能** 0：频率误差计数器禁止 1：频率误差计数器使能 当被设置，CRS_CFGR寄存器被写保护且不能被改写
[4]	保留	保留
[3]	ESYNCIE	**Expected SYNC 中断使能** 0：Expected SYNC（ESYNCF）中断禁止 1：Expected SYNC（ESYNCF）中断使能
[2]	ERRIE	**同步及修正错误中断使能** 0：同步及修正错误（ERRF）中断禁止 1：同步及修正错误（ERRF）中断使能
[1]	SYNCWARNIE	**SYNC 警告中断使能** 0：SYNC 警告(SYNCWARNF)中断禁止 1：SYNC 警告(SYNCWARNF)中断使能
[0]	SYNCOKIE	**SYNC 行为 OK 中断使能** 0：SYNC 行为 OK(SYNCOKF)中断禁止 1：SYNC 行为 OK(SYNCOKF)中断使能

6.18.5.2 CRS 控制寄存器（CRS_CFGR）

寄存器	偏移量	R/W	描述	复位值
CRS_CFGR	CRS_BA+0x04	R/W	控制寄存器 0	0x2022_2EE0

Bits	描述	
[31]	SYNCPOL	**SYNC 极性选择** 0：SYNC 上升沿有效 1：SYNC 下降沿有效
[30]	保留	保留
[29:28]	SYNCSRC	**SYNC信号源选择** 00：GPIO选为SYNC的信号源 01：保留 10：USB SOF选为SYNC的信号源（默认） 11：保留
[27]	保留	保留
[26:24]	SYNCDIV	**SYNC 分频** 这些位由软件设置清除，用来控制SYNC信号的分频系数。 000：SYNC 不分频 001：SYNC 2 分频 010: SYNC 4 分频 011：SYNC 8 分频 100：SYNC 16 分频 101：SYNC 32 分频 110: SYNC 64 分频 111：SYNC 128 分频
[23:16]	FELIM	**频率误差限制** FELIM 包含的数值是用来计算捕获的频率误差数值，频率误

		差数值存放在 CRS_ISR 寄存器的 FECAP 域
[15:0]	RELOAD	**计数器加载的数值** RELOAD 的值在每次同步行为后都会重新加载到频率差寄存器中

6.18.5.3 CRS 中断状态寄存器（CRS_ISR）

寄存器	偏移量	R/W	描述	复位值
CRS_ISR	CRS_BA+0x08	R/W	**控制寄存器 0**	0x0000_0000

Bits	描述	
[31:16]	FECAP	**频率误差捕获** FECAP 是频率误差计数器的值，在每次的同步行为结束后被锁定。
[15]	FEDIR	**频率错误方向** FEDIR 是频率误差计数器的计数方向在每次的同步行为后被锁定，表明了实际频率是高于还是低于目标频率 0：向下技术，实际频率低于目标频率 1：向上计数，实际频率高于目标频率
[14:11]	保留	保留
[10]	TRIMOVF	**修正上溢出或者下溢出** 当自动修正要上溢出或者下溢出的时候，这个标志位被硬件设置。如果 CRS_CR 寄存器的 ERRIE 位被置位，中断产生。通过软件设置 CRS_ICR 寄存器的 ERRC 位来清除这一位。 0：没有修正错误信号 1：产生修正错误信号
[9]	SYNCMISS	SYNC 丢失

		当频率误差计数器达到FELIM的128倍且没有任何SYNC时间被触发，意味着一个SYNC脉冲丢失或者是频率误差太大（内部频率太高）以至于不能通过调整TRIM值来调节，需要采取其它的措施，SYNCERR 被硬件置位。在这种情况下，频率误差计数器停止工作（等待下一个 SYNC）。如果 CRS_CR 寄存器的 ERRIR 位被置位，中断产生。通过软件设置 CRS_ICR 寄存器的 ERRC 位清除。
[8]	SYNCERR	**SYNC 错误** 当同步脉冲输入时，在触发 ESYNC 行为之前，测量到的频率误差大于或者等于 FELIM 的 128 倍，SYNCERR 被硬件置位。这说明频率误差太大（内部频率太低）以至于不能通过调整 TRIM 值来调节，需要采取其它的措施。如果 CRS_CR 寄存器的 ERRIR 位被置位，中断产生。通过软件设置 CRS_ICR 寄存器的 ERRC 位清除 0：没有 SYNC 错误信号 1：产生 SYNC 错误信号
[7：4]	保留	保留
[3]	ESYNCF	**期望 SYNC 标志** 当标志频率计数器到达 0 值得时候，这个标志被硬件置位，如果 CRS_CR 寄存器的 ESYNCIE 位被置位，中断产生。通过软件设置 CRS_ICR 及粗气的 ESYNCC 位清除。
[2]	ERRF	**错误标志** 当误差频率计数器到达 0 值得时候，这个标志位被硬件置位。如果 CRS_CR 寄存器得 ESYNCIE 位被置位，中断产生。通过软件设置 CRS_ICR 寄存器得 ESYNCC 位清除 0：没有期望的 SYNC 信号 1：产生期望的 SYNC 信号
[1]	SYNCWARNF	SYNC 警告标志

		当测量到的频率误差大于或者等于 FELIM 的 3 倍，但小于 FELIM 的 128 倍，这个标志位被硬件置位。这意味着两档甚至更多的挡位才能修正频率误差。如果 CRS_CR 寄存器的 SYNCWARNIE 位被置位，中断产生。通过软件设置 CRS_ICR 寄存器的 SYNCWARNC 位来清除这一位。 0：没有 SYNC 警告信号 1：产生 SYNC 警告信号
[0]	SYNCOKF	**SYNC 行为 OK 标志** 当测量到的频率误差小于 FELIM 的 3 倍，这个标志位被硬件置位。这意味着不需要有校准的行为，或者一档的调节就足够修正频率误差。如果 CRS_CR 寄存器的 SYNCOKIE 位被置位，中断产生。通过软件设置 CRS_ICR 寄存器的 SYNCOKC 位来清除这一位。 0：没有 SYNC 行为 OK 信号 1：产生 SYNC 行为 OK 信号

6.18.5.4 CRS 中断状态寄存器（CRS_ISR）

寄存器	偏移量	R/W	描述	复位值
CRS_ISR	CRS_BA+0x0C	R/W	**控制寄存器 0**	0x0000_0000

Bits	描述	
[31：4]	**保留**	**保留**
[3]	ESYNCC	**Expected SYNC 清除标志** 写 1 清除 CRS_ISR 寄存器的 ESYNCF 标志
[2]	ERRC	**错误清除标志** 写 1 清除 CRS_ISR 寄存器的 SYNCERRF 标志

[1]	SYNCWARNC	**SYNC 警告清除标志** 写 1 清除 CRS_ISR 寄存器的 SYNCWARNF 标志
[0]	SYNCOKC	**SYNC 行为 OK 清除标志** 写 1 清除 CRS_ISR 寄存器的 SYNCOKF 标志

6.19 控制器局域网(CAN)

6.19.1 概述

控制局域网（CAN）是串行数据通信的一种高性能通信协议。CAN控制器提供了一个完整的CAN协议（遵循CAN规范V2.0B）实现方案。

CAN模块可以同时支持多个CAN总线，允许器件在各种CAN总线应用中用作网关、交换机或路由器。

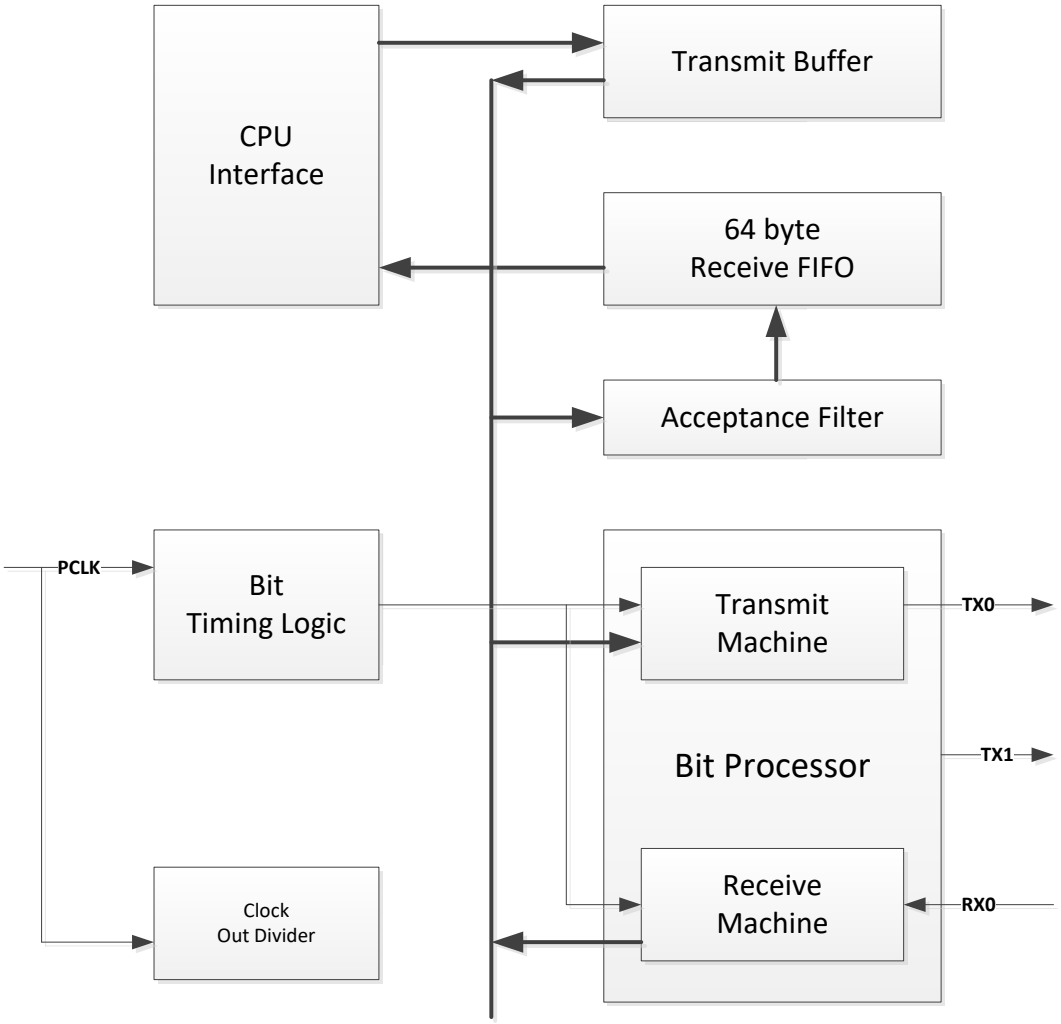
CAN 控制器模块处于微控制器和标准 CAN 收发器之间。CAN 收发器负责将 CAN 控制器生成的数字信号转化为适合总线传输（差分输出）的信号。CAN 总线通常采用双绞线，状态为两线的差分电压。

6.19.2 特性

- 支持CAN协议2.0版本A和B部分
- 多主机结构,带有无破坏性的仲裁
- 为高优先级报文确保了等待时间
- 传输速率可编程(高达 1Mbit/s)
- 多播和广播报文模式
- 数据长度：0-8 字节
- 非归零(NRZ)编码/译码，带有位填充
- 支持 11 位和 29 位的标识符
- 验收滤波
- 可编程的错误报警界限和可读写的错误寄存器
- 仲裁丢失捕获和错误代码捕获(带有详细的位位置)
- 软件驱动的比特率检测(提供热插拔支持)
- 单次触发的发送(不会重复发送)

- 只听模式(无应答，无活动错误标志)
- 自身报文接收(自接收请求)

6.19.3 模块框图



图：CAN 模块结构框图

6.19.4 基本配置

有多个 IO 可以配置成 CAN 管脚，具体根据不同的封装形式有所不同，参见 6.6.4 管脚描述一章，然后在 GPIO 模块中将相应的管脚功能配置成 CAN 功能。

CAN 的模块时钟为 PCLK，通过 AHBCLKCTRL 寄存器的 bit[18]来使能。

6.19.5 功能描述

6.19.5.1 工作模式

CAN 控制器有两种主要工作模式：操作模式和复位模式。在操作模式下，数据可以被接收和发送。在复位模式下，总线时序参数、消息验收滤波可以被设置，接收和发送错误计数器，错误报警界限参数也可以被改变。

当执行了硬件复位或是软件设置了模式寄存器(MOD)的 Reset Mode 位(MOD.0)时会进入复位模式，当 MOD.0 位被清零后控制器返回操作模式。

CAN 控制器还支持只听模式和自测试模式，可以在操作模式或复位模式下通过设置模式寄存器来进入这两种模式。

在只听模式下，控制器只会接收数据，不会发送任何数据。同时 CAN 控制器还强制进入被动错误状态(error passive)。这种模式允许进行软件驱动下的比特率侦测行为，因此使得控制器可以支持 CAN 设备的热插拔。

在自测试模式下，CAN 控制器使用自接收特性来发送和接收数据，而不需要寻求远程节点的应答信号。

6.19.5.2 数据发送

要发送的数据被写到 CAN 控制器的发送缓冲中去，数据可以以标准帧或者扩展帧的格式存在。发送缓冲将 CAN 模块地址空间 0x10-0x1C 的 13 个字节空间组合起来，为一个最多可以包含 8 个字节的消息帧预留了空间。

在往发送缓冲中写入要发送的数据之前，必须通过配置命令寄存器(CMR)来发起一个发送请求(CMR.0=1)，或者在自测试模式下配置命令寄存器来发起一个自接收请求(CMR.4=1)。一旦发送开始，发送状态位(SR.5)变为 1，同时发送请

求位(CMR.0)被清零。

要发送的位序列一次出现在 TX0 管脚上。如果在消息发送过程中出现总线仲裁丢失或是出现发送错误，CAN 控制器会自动重试发送消息。

每个帧后面都会跟着发送一个 15 位的 CRC 码。

消息的发送可以通过设置发送中止命令(CMR.1=1)来中止，只有在消息尚未发送时才可以通过这个命令来中止消息的发送。如果消息已经开始发送，那么无法中止发送过程。

6.19.5.3 数据接收

接收到的数据首先要经过验收滤波器的过滤，然后才会写到接收 FIFO 中去。验收滤波器只会让那些标识符与验收滤波器寄存器中所设置的标识符匹配的消息通过。

当数据被写到接收 FIFO 中去的时候，状态寄存器中的接收状态位(SR.4)变为 1。一旦数据开始被接收到，接收缓冲状态位(SR.0)变为 1 并且产生接收中断(如果中断被使能的话)。

接收 FIFO 有 64 字节深度，允许放入最多五个完整的扩展消息帧。FIFO 是以循环模式来使用的，当 FIFO 中没有足够空间来放置接收到的数据时，状态寄存器中的数据溢出状态位(SR.1)被置 1，同时接收到的数据帧被丢弃，数据溢出中断也会产生(如果使能的话)。

接收 FIFO 中的数据通过一个和接收缓冲相关联的 13 字节的窗口来读取，这个窗口的地址位于 CAN 控制器地址空间的 0x10-0x1C，和发送缓冲占据同样的地址空间。同样，这个空间也足够容纳一个最多具有 8 字节数据的消息。

由于每个消息都是从接收 FIFO 中读出，主机需要设置释放接收缓冲命令(CMR.2=1)来将窗口从 FIFO 中的当前位置释放。如果在接收 FIFO 中有另一个消息在等待被读取，那么它将立刻可以从接收缓冲中读取。如果接收 FIFO 中没有待读取的消息，那么接收中断位和接收缓冲状态位将被清除。

6.19.5.4 自接收

CAN 控制器允许在发送消息给另一个节点的同时，自身也接收这个消息。

对当前发送的消息进行自接收的行为是通过配置自接收请求命令(CMR.4)来实现的。CAN 控制器会自动产生正确操作所需的发送和接收中断。

6.19.5.5 验收滤波

在一个 CAN 网络中，所有的节点都会接收到所有在总线上发送的消息。

为了使一个节点能够忽略与它不相关的消息，CAN 控制器通过对接收到的数据使用一个 4 字节的验收滤波器来预过滤接收数据。只有与滤波器中设定的标识符相同的消息才能通过滤波器进入到接收 FIFO 中。

通常来说，消息过滤是基于完整的标识符的，这个标识符根据消息是标准帧格式还是扩展帧格式可以为 11 或 29 位长度。在 CAN 控制器中可以更进一步通过设立可选的屏蔽位来忽略标识符中的某些位来使得属于某些组的消息可以整个通过验收滤波进入接收 FIFO 中。

滤波器通过 4 个 8 位的验收码寄存器(记录需要匹配的标识符)和 4 个 8 位的验收屏蔽寄存器(可以设置忽略标识符中的哪些位)来实现。这些寄存器组成一个 32 位的滤波器来过滤接收消息的前 4 个字节，或者是组成两组独立的 16 位滤波器来过滤接收消息的前两个字节。当作为两组滤波器来过滤时，消息的标识符至少需要匹配其中的一组才能通过滤波器。

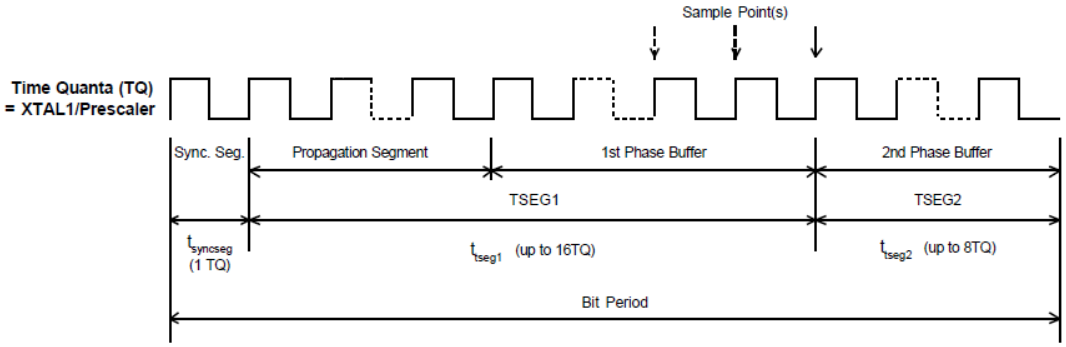
6.19.5.6 波特率

CAN 总线上的波特率通过指定每个数据位周期的长度所占据的时间单元数(TQ)来表示，TQ 表示为若干个 CAN 模块的输入时钟 PCLK 的周期。在一个数据位周期中何时进行数据采样也是由若干个 TQ 来表示。

CAN 控制器允许位周期被定义为 3-25 个 TQ，然而实际使用的位周期需要遵循 BOSCH 所定义的 8-25 个 TQ 的标准。

CAN 控制器使用的位周期和采样点通过三个总线时序寄存器 BTR0，BTR1 和 BTR2 来设置。

CAN 规范规定一个位周期有一个同步段，一个延时段和两个相位缓冲组成。BTR1 寄存器的作用在于：由下图可以看出位周期由同步段加上 TSEG1 和 TSEG2 组成，TSEG1 等于延时段加上第一个相位缓冲，TSEG2 等于第二个相位缓冲。



图：CAN 总线位周期结构图

6.19.5.7 总线仲裁

任何时候，具有最低标识符的 CAN 节点控制 CAN 总线。一个 CAN 节点在失去对总线的控制时必须撤销对总线的控制，并且在 CAN 总线空闲之前不要试图去再去控制总线。

6.19.5.8 错误处理

CAN 控制器包含两个错误计数器：一个用于接收错误计数(RXERR)，一个用于发送错误计数(TXERR)，当发生错误时它们都会自动递加。与此同时，错误的类型和在消息帧中的位置也会被捕获到错误码捕获寄存器中。正确的发送和接收行为也会导致两个错误计数器自动递减。

CAN 控制器还有一个错误报警界限寄存器(EWL)，这个寄存器用来设置当

RXERR 或 TXERR 寄存器中的值达到多少时会产生报警。EWL 的默认值为 96，当 RXERR 或 TXERR 中的值超过 96，状态寄存器中的错误状态位(SR.6)被置 1，同时产生一个错误报警中断(如果使能的话)。

如果任何一个计数器超过 127，CAN 控制器就会进入被动错误状态并发起一个被动错误中断(如果使能的话)。

当 TXERR 超过 255 时，总线状态位(SR.7)会被置 1，CAN 控制器进入复位模式，同时产生一个错误报警中断(如果使能的话)。CAN 控制器会停留在复位模式知道 CPU 将 MOD.0 清零。更进一步，当 CAN 控制器返回操作模式后，会等待 CAN 总线上出现 128 个总线空闲序列后才上线。

6.19.5.9 睡眠模式

当 CAN 总线上没有活动，也没有中断挂起的时候，可以让控制器进入睡眠模式来降低功耗。

通过设置模式寄存器中的睡眠模式位(MOD.4)到 1 来进入睡眠模式。

任何下列事件会唤醒 CAN 控制器:

- 将睡眠模式位设为 0
- CAN 总线 RX0 信号线上有活动

6.19.6 寄存器映射

R: 只读, W: 只写, R/W: 可读写

寄存器	偏移量	R/W	描述	复位后的值
CAN 的基地址： **CAN_BA = 0x4000_6000**				
MODR	CAN_BA+0x00	R/W	模式寄存器	0x0000_0001
CMR	CAN_BA+0x01	W	命令寄存器	0x0000_0000

SR	CAN_BA+0x02	R	状态寄存器	0x0000_003C
IR	CAN_BA+0x03	R	中断寄存器	0x0000_0000
IER	CAN_BA+0x04	R/W	中断使能寄存器	0x0000_0000
BTR0	CAN_BA+0x05	R/W	总线时序寄存器0	0x0000_0000
BTR1	CAN_BA+0x06	R/W	总线时序寄存器1	0x0000_0000
BTR2	CAN_BA+0x07	R/W	总线时序寄存器2	0x0000_0000
OCR	CAN_BA+0x08	R/W	输出控制寄存器	0x0000_0000
ALCR	CAN_BA+0x0B	R	仲裁丢失捕获寄存器	0x0000_0000
ECCR	CAN_BA+0x0C	R	错误码捕获寄存器	0x0000_0000
EWLR	CAN_BA+0x0D	R/W	错误报警界限寄存器	0x0000_0060
RXERR	CAN_BA+0x0E	R/W	接收错误计数寄存器	0x0000_0000
TXERR	CAN_BA+0x0F	R/W	发送错误计数寄存器	0x0000_0000
TRANSMIT BUFFER	CAN_BA+ 0x10-0x1C	W	发送缓冲寄存器	0x0000_0000
RECEIVE WINDOW	CAN_BA+ 0x10-0x1C	R	接收窗口寄存器	0x0000_0000
ACR 0-3	CAN_BA+ 0x10-0x13	R/W	验收码寄存器0-3	0x0000_0000
AMR 0-3	CAN_BA+ 0x14-0x17	R/W	验收码屏蔽寄存器0-3	0x0000_0000
RMCR	CAN_BA+0x1D	R	接收消息计数寄存器	0x0000_0000
RBSA	CAN_BA+0x1E	R/W	接收缓冲器开始地址寄存器	0x0000_0000
CDR	CAN_BA+0x1F	R/W	时钟预分频寄存器	0x0000_0000
RECEIVE FIFO	CAN_BA+ 0x20-0x5F	R/W	接收FIFO	0x0000_0000
TRANSMIT BUFFER	CAN_BA+ 0x60-0x6C	R	发送缓冲	0xXXXX_XXXX

6.19.6.1 MODR 寄存器

该寄存器用来设置 CAN 控制器的行为。

寄存器	偏移量	R/W	描述	复位值
MODR	**CAN_BA+0x00**	**R/W**	**模式寄存器**	**0x0000_0001**

7	6	5	4	3	2	1	0
保留			SM	AFM	STM	LOM	RM

Bits	描述	
[7:5]	保留	保留
[4]	SM	**设置控制器进入睡眠模式** 1：进入睡眠模式 0：唤醒 这个位只有在操作模式下才能进行写操作。 写 1 时，如果 CAN 总线上没有有效活动，也没有中断挂起，则控制器进入睡眠模式，如果写 1 时总线上有活动或有中断挂起，则控制器执行唤醒流程。 写 0 时，如果控制器处于睡眠模式，则执行唤醒流程
[3]	AFM	**设置验收过滤模式** 1：单重过滤，接收数据过滤采用 4 字节过滤器 0：双重过滤，接收数据过滤采用两个短的过滤器
[2]	STM	**设置自测试模式** 1：使能自测试模式 0：正常操作模式 在自测试模式下，可以在总线上没有其它节点的情况下通过自接收请求命令来完成一个完整的节点测试。CAN 控制器可以在没有

		收到响应的情况下成功执行发送任务。
[1]	LOM	**设置只听模式** 1：使能只听模式 0：正常操作模式 在只听模式下，CAN 控制器不会发送任何响应到 CAN 总线上去
[0]	RM	**设置复位模式** 1：使能复位模式 0：正常操作模式 在该位由 0 变 1 时，任何正在发送或接收的消息被中断，控制器进入复位模式

6.19.6.2 CMR 寄存器

设置该寄存器中的一个或多个位来发起 CAN 控制器传输层的操作。

寄存器	偏移量	R/W	描述	复位值
CMR	**CAN_BA+0x01**	**W**	**命令寄存器**	**0x0000_0000**

7	6	5	4	3	2	1	0
保留			SRR	CDO	RRB	AT	TR

Bits	描述	
[7:5]	保留	保留
[4]	SRR	**自接收请求命令** 当一个消息要被同时发送和接收时将这个位设为 1。 同时将 AT 和 SRR 设为 1 会使用自接收特性触发一个单次发送的消息，在出错或仲裁丢失的情况下不会重新发送。

[3]	CDO	**清除数据超载命令** 将这位置 1 可以清除触发数据超载状态位(SR.1)为 1 的数据超载条件。 由于数据超载状态位仍然为 1，因此这个命令不会引起额外的数据超载中断
[2]	RRB	**释放接收缓冲器命令** 设为 1 来释放接收缓冲器。 在读出接收缓冲器中的数据后，CPU 可以通过执行此操作来释放接收缓冲器空间，这会使得另外的消息可用，若没有另外的消息则接收中断位被置位。
[1]	AT	**中止发送命令** 设为 1 来取消下一个尚未被执行的发送请求。 当这个被置 1 时，已经在发送的消息不会被停止。通过检查发送结束状态位来判断最初的消息已经被发送成功或被中止了，应该在发送缓冲器状态位被置 1 或发送中断产生后执行检查操作。
[0]	TR	**发送请求命令** 当要发送一个消息时将这位置 1。 对于之前命令中的发送请求，无法通过将 TR 设为 0 来取消，只有通过将 AT 设为 1 来取消。 同时将 AT 和 TR 设为 1 会触发一个单次发送的消息，在出错或仲裁丢失的情况下不会重新发送。 同时将 AT 和 SRR 设为 1，SRR 会被忽略。

6.19.6.3 SR 寄存器

该寄存器中反映了 CAN 控制器的状态。

寄存器	偏移量	R/W	描述	复位值
SR	**CAN_BA+0x02**	**R**	**状态寄存器**	**0x0000_003C**

7	6	5	4	3	2	1	0
BS	ES	TS	RS	TCS	TBS	DOS	RBS

Bits	描述	
[7]	BS	**总线状态** 1：CAN 控制器处于总线关闭状态，不在处理总线活动 0：CAN 控制器在处理总线活动
[6]	ES	**错误状态** 1：发送和接收错误计数器中的一个或两个已经到达错误报警界限寄存器中(EWL)设置的界限 0：两个错误计数器的值都处于界限值以下
[5]	TS	**发送状态** 1：CAN 控制器正在发送一个消息 0：没有消息在发送
[4]	RS	**接收状态** 1：CAN 控制器正在接收一个消息 0：没有消息在接收 TS 和 RS 都为 0 时，CAN 总线处于空闲。

		TS 和 RS 都为 1 时，CAN 控制器等待进入下一次空闲。 在硬件复位后，一旦检测到总线空闲序列(11 个连续的隐形位)就会进入空闲状态。 在总线关闭事件发生后，必须检测到 128 个总线空闲序列才会重新进入空闲状态。
[3]	TCS	**发送完成状态** 1：上一个被请求的发送成功完成 0：上一个被请求的发送没有完成
[2]	TBS	**发送缓冲器状态** 1：发送缓冲器被释放。CPU 可以往发送缓冲器中写入消息。 0：发送缓冲器被锁定。由于有一个消息正处于发送过程中或处于等待发送过程中，CPU 无法访问发送缓冲器。
[1]	DOS	**数据超载状态** 1：数据超载。由于接收 FIFO 中没有足够的空间来容纳收到的消息而导致消息丢失。 0：自从上一次清除数据超载命令执行后没有发生数据超载。 **只有完整的消息被收到时才会报告超载，如果消息没有完成不会报告超载。**
[0]	RBS	**接收缓冲器状态** 1：接收缓冲器满。一个或多个完整的消息等待被通过接收缓冲器来从接收 FIFO 中读取。 0：接收缓冲器空。当前没有有效的消息等待被读。

6.19.6.4 IR 寄存器

该寄存器中的位指出了 CAN 控制器中断的中断源。当其中一个或多个位被置 1 时,CAN 控制器向 CPU 发送一个中断。当 CPU 读这个寄存器后除了接收中断位以外的其他位被自动复位。

寄存器	偏移量	R/W	描述	复位值
IR	**CAN_BA+0x03**	**R**	**中断寄存器**	**0x0000_0000**

7	6	5	4	3	2	1	0
BEI	ALI	EPI	WUI	DOI	EI	TI	RI

Bits	描述	
[7]	BEI	**总线错误中断** 如果 BEIE(IER.7)位使能，CAN 控制器检测到 CAN 总线上有错误，就会将这位设为 1
[6]	ALI	**仲裁丢失中断** 如果 ALIE(IER.6)位使能，CAN 控制器在尝试发送时丢失总线总裁就会将这位设为 1
[5]	EPI	**被动错误中断** 如果 EPIE(IER.5)位使能，CAN 控制器在由被动错误状态重新进入主动错误状态时或至少一个错误计数器的值超过协议所规定的 127 时将这个位设为 1
[4]	WUI	**唤醒中断** 如果 WUIE(IER.4)位使能，当 CAN 控制器处于睡眠模式时检测到总线活动时将这个位设为 1。

		当 CPU 试图将 CAN 控制器置为睡眠模式时，如果 CAN 控制器仍在处理总线活动或是有中断挂起，此时也会将这个位设为 1。
[3]	DOI	**数据溢出中断** 如果 DOIE(IER.3)位使能，当数据溢出状态位(SR.1)从 0 变到 1 时将这个位设为 1。
[2]	EI	**错误报警中断** 如果 EIE(IER.2)位使能，当总线状态位(SR.7)或错误状态位(SR.6)中任意一个位每次发生改变时将这个位设为 1。
[1]	TI	**发送中断** 如果 TIE(IER.1)位使能，当发送缓冲状态位(SR.2)从 0 变到 1 时将这个位设为 1。
[0]	RI	**接收中断** 如果 RIE(IER.0)位使能，当接收缓冲中有一个或多个有效消息时将这位设为 1。 当释放接收缓冲命令(CMR.2)被执行时，如果接收缓冲总没有消息需要被读走，那么将这个位清 0。

6.19.6.5 IER 寄存器

该寄存器用来设置哪些事件可以触发一个 CPU 中断。

寄存器	偏移量	R/W	描述	复位值
IER	**CAN_BA+0x04**	**R/W**	**中断使能寄存器**	**0x0000_0000**

7	6	5	4	3	2	1	0
BEIE	ALIE	EPIE	WUIE	DOIE	EIE	TIE	RIE

Bits	描述	
[7]	BEIE	**总线错误中断使能** 1：使能总线错误中断 0：禁止总线错误中断
[6]	ALIE	**仲裁丢失中断使能** 1：使能仲裁丢失中断 0：禁止仲裁丢失中断
[5]	EPIE	**被动错误中断使能** 1：使能被动错误中断 0：禁止被动错误中断
[4]	WUIE	**唤醒中断使能** 1：使能唤醒中断 0：禁止唤醒中断
[3]	DOIE	**数据溢出中断使能** 1：使能数据溢出中断 0：禁止数据溢出中断
[2]	EIE	**错误报警中断使能** 1：使能错误报警中断 0：禁止错误报警中断
[1]	TIE	**发送中断使能** 1：使能发送中断 0：禁止发送中断
[0]	RI	**接收中断使能** 1：使能接收中断

		0：禁止接收中断

6.19.6.6 BTR0 寄存器

BTR 寄存器用来控制如何从 APB 时钟获得所需的各种 CAN 时序。BTR 寄存器只能在复位模式下编程，操作模式下是只读的。

其中 BTR0 定义了同步跳变宽度(SJW),BTR0 还和 BTR2 一起定义了波特率预分频系数。

寄存器	偏移量	R/W	描述	复位值
BTR0	**CAN_BA+0x05**	**R/W**	**总线时序寄存器0**	**0x0000_0000**

7	6	5	4	3	2	1	0
SJW[1:0]		BRP[5:0]					

Bits	描述	
[7:6]	SJW	**设置同步跳变宽度** 定义了在每个位周期中可以延长或缩短多少个时间单元(TQ)的上限，其值为 SJW 的值加 1。
[5:0]	BRP[5:0]	**设置波特率预分频系数** 和 BTR2 寄存器一起定义了时钟预分频系数，描述见 BTR2 寄存器

6.19.6.7 BTR1 寄存器

BTR 寄存器用来控制如何从 APB 时钟获得所需的各种 CAN 时序。BTR 寄存器只能在复位模式下编程，操作模式下是只读的。

其中 BTR1 定义了位周期的长度，采样点的位置以及每个采样点处要采样的次数。

寄存器	偏移量	R/W	描述	复位值
BTR1	**CAN_BA+0x06**	**R/W**	**总线时序寄存器1**	**0x0000_0000**

7	6	5	4	3	2	1	0
SAM	TSEG2			TSEG1			

Bits	描述	
[7]	SAM	**设置采样次数** 1：总线被采样三次(建议用于低到中速总线过滤总线线路干扰的情况) 0：总线被采样一次(建议用于高速总线情况)
[6:4]	TSEG2	从采样点到下个指定同步点的延时是(TSEG2的值+1)个时间单元。规定的位周期的时间是（TSEG2的值+TSEG1的值+3）个时间单元
[3:0]	TSEG1	从指定同步到采样点的延时是（TSEG1值+1）个时间单元 T_{SYNC} = 1xTQ T_{SEG1} = TQ x (8 x TSEG1.3 + 4 x TSEG1.2 + 2 x TSEG1.1 + TSEG1.0 + 1) T_{SEG2} = TQ x (4 x TSEG2.2 + 2 x TSEG2.1 + TSEG2.0 + 1) 理论上，可以通过寄存器将位周期时间设置为3~25TQ，但是，实际所用的位周期需要满足CAN协议中规定的8~25TQ

6.19.6.8 BTR2 寄存器

BTR 寄存器用来控制如何从 APB 时钟获得所需的各种 CAN 时序。BTR 寄存器只能在复位模式下编程，操作模式下是只读的。

其中 BTR2 和 BTR0 一起定义了波特率预分频系数。

寄存器	偏移量	R/W	描述	复位值
BTR2	**CAN_BA+0x07**	**R/W**	**总线时序寄存器2**	**0x0000_0000**

7	6	5	4	3	2	1	0
保留				BRP[9:6]			

Bits	描述	
[7:4]	保留	保留
[3:0]	BRP[9:6]	**设置波特率预分频系数** 和 BTR0 寄存器一起定义了时钟预分频系数 BRP[9:0]。 这个预分频系数定义了一个时间单元 TQ 有多少个 PCLK 组成。 $TQ = 2 \times T_{PCLK} \times (512 \times BRP.9 + 256 \times BRP.8 + 128 \times BRP.7 + 64 \times BRP.6 + 32 \times BRP.5 + 16 \times BRP.4 + 8 \times BRP.3 + 4 \times BRP.2 + 2 \times BRP.1 + BRP.0 + 1)$

6.19.6.9 OCR 寄存器

OCR 寄存器用来控制 CAN 模块 TX1 管脚上的输出。这个寄存器对 CAN 模块的操作无影响。

寄存器	偏移量	R/W	描述	复位值
CR	CAN_BA+0x08	R/W	输出控制寄存器	0x0000_0000

6.19.6.10 ALCR 寄存器

ALCR 寄存器用来记录仲裁丢失位置。总线仲裁丢失后，产生一个仲裁丢失中断（若使能），并且串行位数据序列处理的当前位置被记录到仲裁丢失捕捉寄存器中。该寄存器的内容一直保持到软件对其读操作，之后捕捉机制恢复正常。

寄存器	偏移量	R/W	描述	复位值
ALCR	CAN_BA+0x0B	R	仲裁丢失捕获寄存器	0x0000_0000

7	6	5	4	3	2	1	0
保留			ALC				

Bits	描述	
[7:5]	保留	保留
[4:0]	ALC	记录仲裁丢失的位置 00000：仲裁丢失位置在标识符第 1 位(ID.28) 00001：仲裁丢失位置在标识符第 2 位(ID.27) 00010：仲裁丢失位置在标识符第 3 位(ID.26) 00011：仲裁丢失位置在标识符第 4 位(ID.25) 00100：仲裁丢失位置在标识符第 5 位(ID.24) 00101：仲裁丢失位置在标识符第 6 位(ID.23) 00110：仲裁丢失位置在标识符第 7 位(ID.22) 00111：仲裁丢失位置在标识符第 8 位(ID.21)

		01000：仲裁丢失位置在标识符第 9 位(ID.20) 01001：仲裁丢失位置在标识符第 10 位(ID.19) 01010：仲裁丢失位置在标识符第 11 位(ID.18) 01011：仲裁丢失位置在 SRTR 位 01100：仲裁丢失位置在 IDE 位 01101：仲裁丢失位置在标识符第 12 位(ID.17) 01110：仲裁丢失位置在标识符第 13 位(ID.16) 01111：仲裁丢失位置在标识符第 14 位(ID.15) 10000：仲裁丢失位置在标识符第 15 位(ID.14) 10001：仲裁丢失位置在标识符第 16 位(ID.13) 10010：仲裁丢失位置在标识符第 17 位(ID.12) 10011：仲裁丢失位置在标识符第 18 位(ID.11) 10100：仲裁丢失位置在标识符第 19 位(ID.10) 10101：仲裁丢失位置在标识符第 20 位(ID.9) 10110：仲裁丢失位置在标识符第 21 位(ID.8) 10111：仲裁丢失位置在标识符第 22 位(ID.7) 11000：仲裁丢失位置在标识符第 23 位(ID.6) 11001：仲裁丢失位置在标识符第 24 位(ID.5) 11010：仲裁丢失位置在标识符第 25 位(ID.4) 11011：仲裁丢失位置在标识符第 26 位(ID.3) 11100：仲裁丢失位置在标识符第 27 位(ID.2) 11101：仲裁丢失位置在标识符第 28 位(ID.1) 11110：仲裁丢失位置在标识符第 29 位(ID.0) 11111：仲裁丢失位置在 RTR 位 位 13-31 只针对扩展帧格式

		SRTR 位表示标准帧格式里的 RTR 位

6.19.6.11 ECCR 寄存器

ECCR 寄存器用来记录总线的错误类型和错误位置。

总线发生错误时，产生一个总线错误中断（若使能），并且当前数据位的位置信息被记录到错误码捕捉寄存器中。该寄存器的内容一直保持到软件对其读操作，之后错误捕捉机制恢复正常。

寄存器	偏移量	R/W	描述	复位值
ECCR	**CAN_BA+0x0C**	**R**	**错误码捕获寄存器**	**0x0000_0000**

7	6	5	4	3	2	1	0
ERRCODE			DIR	SEGCODE			

Bits	描述	
[7:5]	ERRCODE	**错误类型** 00：位错误 01：格式错误 10：填充错误 11：其它错误
[4]	DIR	**错误方向** 1：接收时发生错误 0：发送时发生错误
[3:0]	SEGCODE	**错误段**

		00011：帧起始处 00010：ID.28 到 ID.21 00110：ID.20 到 ID.18 00100：SRTR 位 00101：IDE 位 00111：ID.17 到 ID.13 01111：ID.12 到 ID.5 01110：ID.4 到 ID.0 01100：RTR 位 01101：保留位 1 01001：保留位 0 01011：数据长度码 01010：数据域 01000：CRC 序列 11000：CRC 界定符 11001：应答 11011：应答界定符 11010：帧结尾 10010：间歇场 10001：主动错误标志 10110：被动错误标志 10011：冗余显性位 10111：错误界定符 11100：过载标志

6.19.6.12 EWLR 寄存器

EWLR 寄存器用来定义何时产生错误报警中断的界限(如果使能的话)。

该寄存器只能在复位模式下写入，工作模式下只读。还应该注意，复位模式下对寄存器做的的更改只有在重新进入工作模式后才会有效。

寄存器	偏移量	R/W	描述	复位值
EWLR	**CAN_BA+0x0D**	**R/W**	**错误报警界限寄存器**	**0x0000_0060**

7	6	5	4	3	2	1	0
EWL							

Bits	描述	
[7:0]	EWL	**设置产生错误报警中断的界限** **在 CAN 模块工作过程中，TXERR 和 RXERR 寄存器中的值与这个值进行比较，如果有一个值与这个值匹配，SR 寄存器的 ES 状态位被设置。**

6.19.6.13 RXERRR 寄存器

RXERR 寄存器用来定义何时产生错误报警中断的界限(如果使能的话)。

该寄存器记录接收错误计数器的当前计数值。这个计数器在接收数据流有错时递增，当接收消息无错时递减。RXERR 和 TXERR 一起可以对 CAN 总线上的传输质量进行表征。

寄存器	偏移量	R/W	描述	复位值
RXERR	**CAN_BA+0x0E**	**R/W**	**接收错误计数寄存器**	**0x0000_0000**

7	6	5	4	3	2	1	0
VALUE							

Bits	描述	
[7:0]	VALUE	**记录接收错误计数器的当前值** 当这个值达到 EWLR 寄存器中设置的值时，将会产生一个错误报警中断(如果使能的话)。但是如果之前已经由 TXERR 产生了这个中断，那 RXERR 不再产生中断。 当这个值超过 127 时，设备进入被动错误状态（除非 TXERR 已经导致进入被动错误状态），触发一个主动错误事件并产生主动错误中断(如果使能的话)。

6.19.6.14 TXERRR 寄存器

TXERR 寄存器用来定义何时产生错误报警中断的界限(如果使能的话)。

该寄存器记录发送错误计数器的当前计数值。这个计数器在发送有错时递增，当发送消息无错时递减。RXERR 和 TXERR 一起可以对 CAN 总线上的传输质量进行表征。

寄存器	偏移量	R/W	描述	复位值
TXERR	**CAN_BA+0x0F**	**R/W**	**发送错误计数寄存器**	**0x0000_0000**

7	6	5	4	3	2	1	0
VALUE							

Bits	描述	
[7:0]	VALUE	**记录发送错误计数器的当前值** 当这个值达到 EWLR 寄存器中设置的值时，将会产生一个错误报警中断(如果使能的话)。但是如果之前已经由 RXERR 产生了这个中断，那 TXERR 不再产生中断。 当这个值超过 127 时，设备进入被动错误状态（除非 RXERR 已经导致进入被动错误状态），触发一个主动错误事件并产生主动错误中断(如果使能的话)。 当这个值超过 255 时，设备进入掉线状态并自动进入复位模式，并产生一个错误报警中断(如果使能的话)。

6.19.6.15 TRANSMIT BUFFER 寄存器

TRANSMIT BUFFER 寄存器有 13 个字节的长度，可以容纳一个最多包含 8 字节数据的发送消息。对这个 BUFFER 的读访问使用 0x60-0x6C 的地址空间。

寄存器	偏移量	R/W	描述	复位值
TRANSMIT BUFFER	**CAN_BA+ 0x10-0x1C**	**W**	**发送缓冲寄存器**	**0x0000_0000**

6.19.6.15.1 发送缓冲器布局

发送缓冲器被分为描述符和数据场。描述符的第一个字节包含帧信息，描述了帧格式（SFF or EFF），远程帧或数据帧标识，及数据长度。然后是标准帧格式的两个字节的标识符或者扩展帧格式的四个字节的标识符。数据场最多包含 8 个字节数据。

标准帧格式(SFF)		扩展帧格式(EFF)	
CAN 地址	**场**	**CAN 地址**	**场**
10h	帧信息	10h	帧信息
11h	标识符 1	11h	标识符 1
12h	标识符 2	12h	标识符 2
13h	数据 1	13h	标识符 3
14h	数据 2	14h	标识符 4
15h	数据 3	15h	数据 1
16h	数据 4	16h	数据 2
17h	数据 5	17h	数据 3
18h	数据 6	18h	数据 4
19h	数据 7	19h	数据 5
1Ah	数据 8	1Ah	数据 6
1Bh	(unused)	1Bh	数据 7
1Ch	(unused)	1Ch	数据 8

6.19.6.15.2 发送缓冲器场分布

发送缓冲器描述符场的位分布见下表，上面的为标准帧，下面的为扩展帧。描述符的具体差别在下面对应部分介绍。

- 发送格式(SFF)

地址	BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
10h	FF	RTR	$X_{(1)}$	$X_{(1)}$	DLC.3	DLC.2	DLC.1	DLC.0
11h	ID.28	ID.27	ID.26	ID.25	ID.24	ID.23	ID.22	ID.21
12h	ID.20	ID.19	ID.18	$X_{(2)}$	$X_{(1)}$	$X_{(1)}$	$X_{(1)}$	$X_{(1)}$

- 发送格式(EFF)

地址	BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
10h	FF	RTR	$X_{(1)}$	$X_{(1)}$	DLC.3	DLC.2	DLC.1	DLC.0

11h	ID.28	ID.27	ID.26	ID.25	ID.24	ID.23	ID.22	ID.21
12h	ID.20	ID.19	ID.18	ID.17	ID.16	ID.15	ID.14	ID.13
13h	ID.12	ID.11	ID.10	ID.9	ID.8	ID.7	ID.6	ID.5
14h	ID.4	ID.3	ID.2	ID.1	ID.0	$X_{(2)}$	$X_{(1)}$	$X_{(1)}$

(1)：任意值，但推荐为 0
(2)：任意值，但推荐为 1

FF(Frame Format)，选择待发送报文的帧格式类型。“1”表示扩展帧，“0”表示标准帧。

RTR(Remote Transmission Request)，用来表示待发送报文时数据帧还是远程帧。“1”表示远程帧（即,向其他节点请求数据），“0”表示数据帧。

DLC(Data Length Code)，用来指定待发送报文中的数据字节数。在一个数据帧中可发送的数据字节数最多为 8，因此，若 DLC[3:0]的值大于 8，会被自动截断为 8。

ID(Identifier)，可以视为报文的名字，接收器用其来对报文进行验收滤波。同时，标识符也决定了发送器访问总线的优先级。标识符越小，其对应优先级越高。

数据场中的数据长度应当同 DLC 中指定的字节数保持一致。地址为 19h(SFF)或 21h(EFF)的数据字节的最高位(MSB)首先发送。

6.19.6.16 RECEIVE BUFFER 寄存器

RECEIVE BUFFER 寄存器有 13 个字节的长度，为 CPU 访问 RXFIFO 提供了一个窗口。可以容纳一个最多包含 8 字节数据的接收消息。这个 BUFFER 与 TX BUFFER 具有同样的地址空间。

寄存器	偏移量	R/W	描述	复位值
RECEIVE BUFFER	**CAN_BA+ 0x10-0x1C**	**W**	**发送缓冲寄存器**	**0x0000_0000**

6.19.6.16.1 接收缓冲器布局

接收缓冲器被分为描述符和数据场。描述符的第一个字节包含帧信息，描述了帧格式（SFF or EFF），远程帧或数据帧标识，及数据长度。然后是标准帧格式的两个字节的标识符或者扩展帧格式的四个字节的标识符。数据场最多包含 8 个字节数据。

标准帧格式(SFF)		扩展帧格式(EFF)	
CAN 地址	场	CAN 地址	场
10h	帧信息	10h	帧信息
11h	标识符 1	11h	标识符 1
12h	标识符 2	12h	标识符 2
13h	数据 1	13h	标识符 3
14h	数据 2	14h	标识符 4
15h	数据 3	15h	数据 1
16h	数据 4	16h	数据 2
17h	数据 5	17h	数据 3
18h	数据 6	18h	数据 4
19h	数据 7	19h	数据 5
1Ah	数据 8	1Ah	数据 6
1Bh	(unused)	1Bh	数据 7
1Ch	(unused)	1Ch	数据 8

6.19.6.16.2 接收缓冲器场分布

接收缓冲器描述符场的位分布见下表，上面的为标准帧，下面的为扩展帧。描述符的具体差别在下面对应部分介绍。

- **发送格式(SFF)**

地址	BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
10h	FF	RTR	0	0	DLC.3	DLC.2	DLC.1	DLC.0

11h	ID.28	ID.27	ID.26	ID.25	ID.24	ID.23	ID.22	ID.21
12h	ID.20	ID.19	ID.18	RTR	0	0	0	0

- 发送格式(SFF)

地址	BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
10h	FF	RTR	0	0	DLC.3	DLC.2	DLC.1	DLC.0
11h	ID.28	ID.27	ID.26	ID.25	ID.24	ID.23	ID.22	ID.21
12h	ID.20	ID.19	ID.18	ID.17	ID.16	ID.15	ID.14	ID.13
13h	ID.12	ID.11	ID.10	ID.9	ID.8	ID.7	ID.6	ID.5
14h	ID.4	ID.3	ID.2	ID.1	ID.0	RTR	0	0

注：接收的帧信息字节(地址 10h)里的数据长度代码表示发送器所发送的数据字节数，可能会大于 8 个字节。但是，可以接受的最大数据长度为 8 字节。

6.19.6.17 ACR 寄存器

这 4 个 8 位寄存器提供验收滤波的模式，接收器用这些寄存器的内容及 AMR0~3 提供的掩码对接收到的报文进行验收滤波。

模式的设置方式取决于验收码寄存器处于单滤波模式还是双滤波模式，以及报文帧为标准帧还是扩展帧。

这几个寄存器只能在复位模式下读写访问。

寄存器	偏移量	R/W	描述	复位值
ACR 0-3	CAN_BA+ 0x10-0x13	R/W	验收码寄存器0-3	0x0000_0000

6.19.6.18 AMR 寄存器

这 4 个 8 位寄存器提供验收滤波的掩码模式。“0”代表下一个报文的标识符的对应位需要与验收码寄存器的对应位匹配；“1”代表不用考虑。

寄存器的工作方式取决于滤波器处于单滤波器状态还是双滤波器状态，以及

报文帧为标准帧还是扩展帧。

这几个寄存器只能在复位模式下读写访问。

寄存器	偏移量	R/W	描述	复位值
AMR 0-3	CAN_BA+ 0x14-0x17	R/W	验收码屏蔽寄存器0-3	0x0000_0000

6.19.6.19 RMCR 寄存器

该寄存器用来记录当前 RxFIFO 中的可读报文数。它在每一个接收事件完成后加 1，或者在接收缓冲释放命令后减 1。

该寄存器在复位模式和工作模式下均为只读。

寄存器	偏移量	R/W	描述	复位值
RMCR	CAN_BA+0x1D	R	接收消息计数寄存器	0x0000_0000

7	6	5	4	3	2	1	0
保留			RMC				

Bits	描述	
[7:5]	保留	保留
[4:0]	RMC	记录接收到的消息的数目

6.19.6.20 RBSA 寄存器

该寄存器记录了 RxFIFO 读指针（0~63）的当前指向位置。0 对应地址 20h；63 对应地址 5Fh。

该寄存器可以通过硬件复位清零，但软件复位对其无影响（同样对 RxFIFO

的内容也无影响）。但是，软件复位将 RxFIFO 的读指针设为同写指针一致。因此，软件复位后到达接收缓冲器访问的消息会被下一个写入 RxFIFO 的消息所覆盖。

寄存器	偏移量	R/W	描述	复位值
RBSA	**CAN_BA+0x1E**	**R/W**	**接收缓冲器开始地址寄存器**	**RBSA**

7	6	5	4	3	2	1	0
保留			RBSA				

Bits	描述	
[7:5]	保留	保留
[4:0]	RBSA	记录了 RxFIFO 读指针（0~63）的当前指向位置

6.19.6.21 CDR 寄存器

该寄存器用来控制 CLKOUT 信号。对 CAN 控制器的操作无影响。

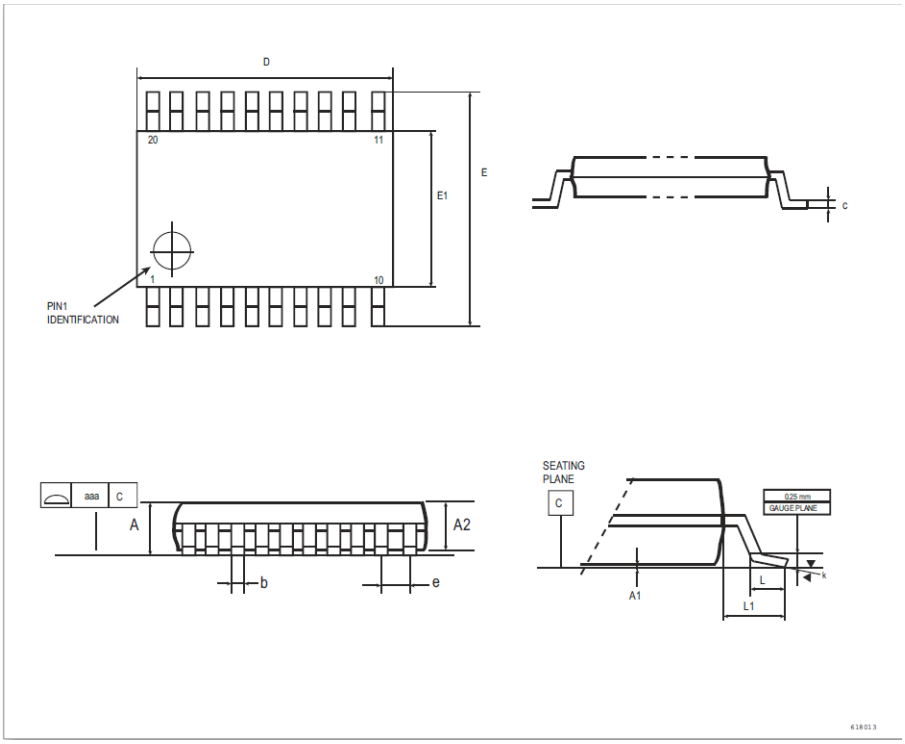
寄存器	偏移量	R/W	描述	复位值
CDR	**CAN_BA+0x1F**	**R/W**	**时钟预分频寄存器**	**0x0000_0000**

7. MT32F005/006 电气特性

参见 MT32F005/006 芯片规格书

8. 封装尺寸

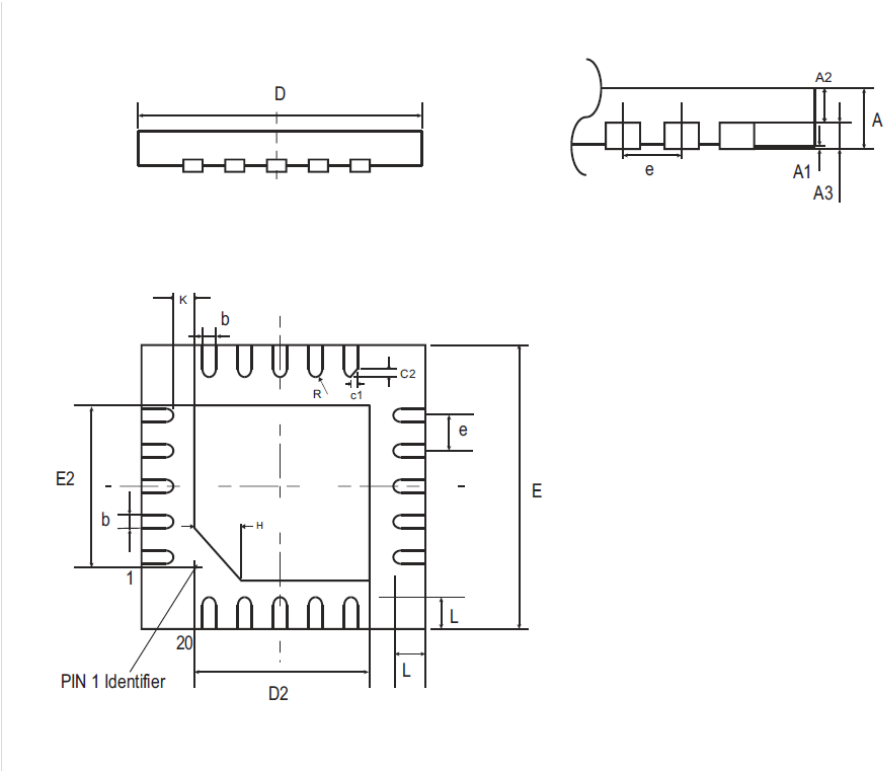
8.1 TSSOP20



标号	毫米		
	最小值	典型值	最大值
A			1.20
A1	0.05		0.15
A2	0.08	1.00	1.05
b	0.19		0.30
c	0.09		0.20
D	6.40	6.50	6.60
E	6.25	6.40	6.55
E1	4.30	4.40	4.50
e		0.65	
L	0.45	0.60	0.75

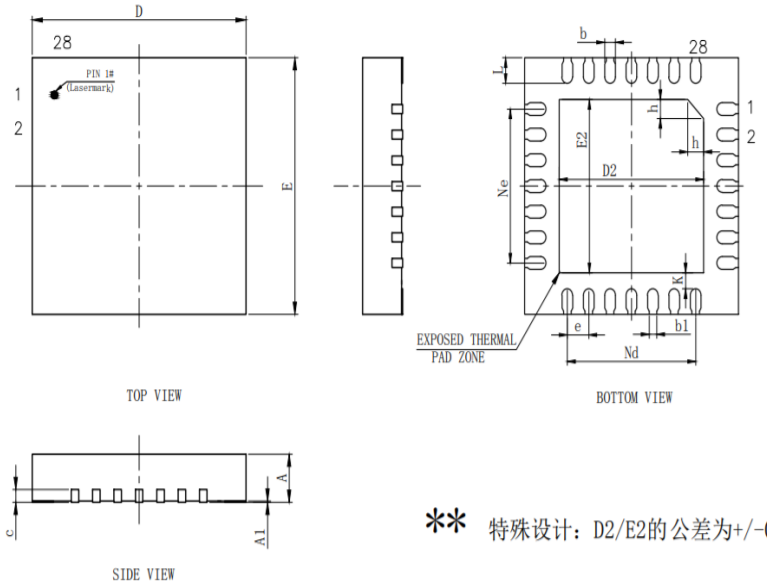
标号	毫米		
	最小值	典型值	最大值
L1		1.00	
N	引脚数目 = 20		

8.2 QFN20



标号	毫米		
	最小值	典型值	最大值
A	0.70	0.75	0.80
A1	0.00	0.02	0.05
A2	0.50	0.55	0.60
A3		0.20REF	
b	0.15	0.20	0.25
D	2.90	3.00	3.10
E	2.90	3.00	3.10
D2	1.40	1.50	1.60
E2	1.40	1.50	1.60
e	0.30	0.40	0.50
H		0.35REF	
K		0.35REF	
L	0.35	0.40	0.45
R	0.085		
C1		0.07	
C2		0.07	
N	引脚数目 = 20		

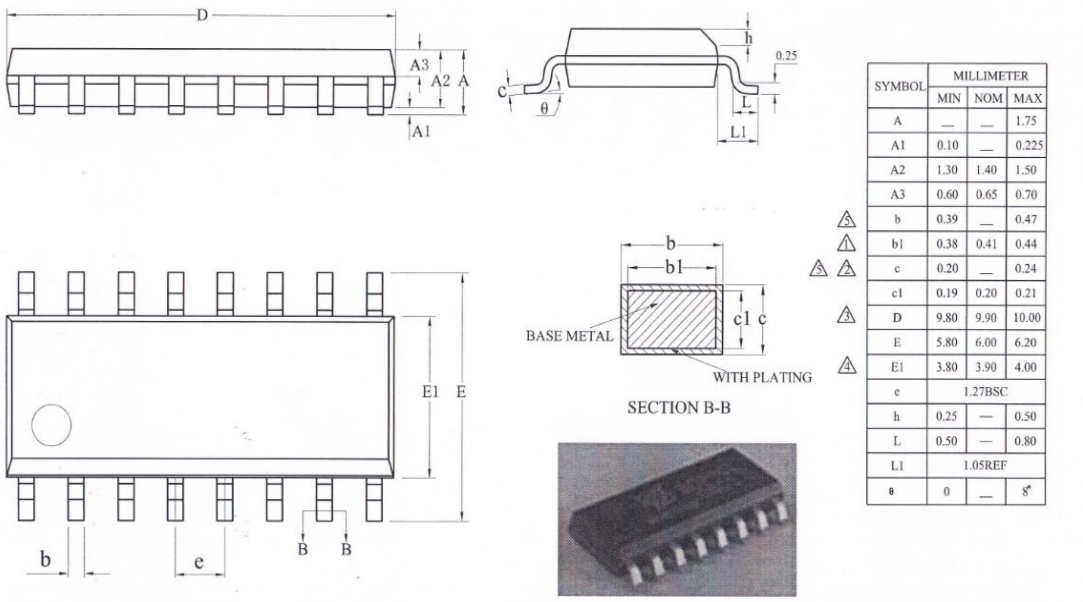
8.3 QFN28



SYMBOL	MILLIMETER		
	MIN	NOM	MAX
A	0.70	0.75	0.80
A1	0	0.02	0.05
b	0.15	0.20	0.25
b1	0.14REF		
c	0.20REF		
D	3.90	4.00	4.10
D2	2.65	2.70	2.75
e	0.40BSC		
Ne	2.40BSC		
Nd	2.40BSC		
E	3.90	4.00	4.10
E2	2.65	2.70	2.75
L	0.35	0.40	0.45
h	0.25	0.30	0.35
K	0.25REF		

** 特殊设计：D2/E2的公差为+/-0.05

8.4 SOP16



SYMBOL	MILLIMETER		
	MIN	NOM	MAX
A	—	—	1.75
A1	0.10	—	0.225
A2	1.30	1.40	1.50
A3	0.60	0.65	0.70
5 b	0.39	—	0.47
1 b1	0.38	0.41	0.44
5 4 c	0.20	—	0.24
c1	0.19	0.20	0.21
3 D	9.80	9.90	10.00
E	5.80	6.00	6.20
4 E1	3.80	3.90	4.00
e	1.27BSC		
h	0.25	—	0.50
L	0.50	—	0.80
L1	1.05REF		
θ	0	—	8°

9. 版本历史

版本	日期	页	描述
V1.0	2020 年 12 月		正式版第一版
V1.1	2021 年 1 月		添加 SRAM，GPIO 位带功能描述 添加 SWD 调试 IO 功能复用硬件保护描述
V1.2	2021 年 1 月		修改 HW_WAKEUPEN 寄存器描述
V1.3	2021 年 3 月		修改 IOCONFIG 的 OS，OD 控制位位置描述错误 修改芯片选型表