译文

TMPM364F10FG

本资料是为了参考的目的由原始文档翻译而来。

使用本资料时，请务必确认原始文档关联的最新信息，并遵守其相关指示。

原本：”TMPM364F10FG” 2013-05-31

翻译日:2014-03-10

TOSHIBA CORPORATION
Semiconductor & Storage Products Company

译文

TOSHIBA

32位RISC微控制器

TX03系列

TMPM364F10FG

东芝株式会社

半导体&存储产品公司

译文

引言:有关本规范下SFR (特殊功能寄存器)描述的说明

SFR (特殊功能寄存器)为一款外围电路 (IP)控制寄存器。

该IP的SFR地址见内存地图的章节中所述叙述，SFR的详细说明见各IP的章节中所述。

本规范中采用的SFR的定义依据以下规则。

a. 各IP的SFR表举例

- IP各章节中所述SFR表描述了寄存器名称，地址与简要说明。
- 所有寄存器均有一32-位的唯一地址，这些寄存器的地址定义如下，一些例外除外: “基址+ (唯一)地址“

基址=0x0000_0000

寄存器名称		地址 (基址+)
控制寄存器	SAMCR	0x0004
		0x000C

注: **SAMCR寄存器地址为32位，即地址0x0000_0004 (基址 (0x00000000)+唯一地址 (0x0004))。**

注: **以上显示的寄存器为出于解释性，而非示范性目的而列举的示例。本微控制器中不存在该寄存器。**

b. SFR (寄存器)

- 每个寄存器基本上包含一个32-位寄存器 (有些例外)。
- 每个寄存器对位，比特符号，类型，复位后的初值以及功能进行了描述。

1.2.2 SAMCR (控制寄存器)

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

位	比特符号	类型	功能
31-10	−	R	“0“ 可读取。
9-7	MODE[2:0]	R/W	工作模式设定 000：采样模式 0 001：采样模式 1 010：采样模式 2 011：采样模式 3 以上设定除外的设定:保留
6-0	TDATA[6: 0]	W	传输数据

注: **分为三个类型，显示如下。**

R / W	读取写入
R	读取
W	写入

c. 数据描述

SFR描述中使用的符号含意显示如下。

- x:通道编号/端口
- n, m:位编号

d. 寄存器描述

寄存器按显示如下进行描述。

- 寄存器名称<比特符号>
  示例: SAMCR<MODE>= “000“ 或 SAMCR <MODE [2:0]>= “000“
  <MODE[2:0]>表示比特符号模式中位2 ~ 位0 (3位宽度)。
- 寄存器名称[位]
  例如:SAMCR[9:7]= “000“
  表示寄存器SAMCR (32位宽度)的位9 ~ 位7。

修订记录

日期	版本	备注
2011/4/8	试行版1	试行版的第一版
2011/6/20	1	第一版
2013/5/31	2	目录修订

译文

目 录

8. 输入 / 输出端口

9. DMA 控制器 (DMAC)

10. 静态存储控制器

15. 消费性电子产品控制 (CEC)

16. CAN 控制器 (CAN)

TMPM364F10FG

TMPM364FFG为一款带有ARM Cortex-M3微处理器内核的32-位RISC微处理器系列。

产品名称	ROM (FLASH)	RAM	封装
TMPM364F10FG	1024K字节	64K字节	P-LQFP144-2020-0.50E

TMPM364F10FG的特点如下:

1.1 特点

1. ARM Cortex-M3微处理器内核
   a. 通过使用Thumb-2指令集，提高编码效率。
      - 新的16-位Thumb指令集适用于已改进的程序流程
      - 新的32-位Thumb指令集适用于已改善的性能
      - 新的Thumb混合型16- /32-位指令集可生成更快速，更有效的编码。

   b. 既获得了高性能又实现了低功耗。

      [性能高]
      - 既获得了高性能又实现了低功耗。
      - 取决于被除数与除数，分值在2与12周期之间

      [功耗低]
      - 使用低功耗库优化了设计
      - 停止微控制器内核工作的待机功能

   c. 适于实时控制的高速中断响应
      - 可中断的长指令
      - 由硬件自动处理入栈

2. 单片级程序存储器与数据内存
   - 片上RAM : 64K字节
   - 片上 Flash ROM : 1024K字节

3. 静态存储器控制器 (SMC)
   - 达16M字节存取区 (程序 / 数据)
   - 外部数据总线 (分离总线/多路传输总线): 16-位数据总线宽度
   - 片选 / 等待控制器 : 4 通道

4. DMA控制器 (DMAC): 2通道

   转输可支持单片存储器 / 外围I/O / 外存储器

5. 16-位计时器/事件计数器 (TMRB) : 16 通道
   - 16-位区间计时器模式
   - 16-位事件计数器模式
   - 16-位PPG输出 (4通道计时器可同步启动)
   - 输入捕捉功能

6. 看门狗计时器 (WDT) : 1 通道
   看门狗计时器生成一个复位或一个不可屏蔽中断 (NMI)

7. 串口通道 (SIO/UART): 12 通道
   UART模式或输入/输出接口均可选择 (配备有4Byte FIFO)

8. 串行总线接口 (I2C/SIO): 5 通道
   无论是I2C总线模式还是同步8-位SIO模式均可选用。

9. 同步串行端口 (SSP):1 通道
   - 包括SPI的通信协议: 3种类型 (SPI/SSI/Microwire)
   - 波特率:主机模式: 16Mbps (最大值)，从机模式:5.3Mbps (最大值)

10. USB2.0全速主机:1 通道
    - 符合通用串行总线规范2.0修订版
    - 符合1.0a版本的OpenHCI
    - 控制/批量/中断/等时模式
    - 全速12Mbps (低速不提供。)

11. CAN 2.0: 1 通道.
    - 支持2.0B版本Active
    - 32个邮箱
    - 最大传输速率:1Mbps

12. CEC功能 (CEC) : 1 通道
    每1字节传输与接收

13. 遥控信号预处理程序 (RMC) : 2 通道
    一次可接收多达72位数据

14. 10-位 AD转换器 (ADC) : 16 通道
    - 采用16-位计时器启动
    - 固定通道 / 扫描模式
    - 单个 / 重复模式
    - AD监控2 通道
    - (@fsys = 40 MHz时)转换时间1.15 μsec

15. 触键唤醒 (KWUP) : 4 通道
    动态上拉

16. 实时时钟 (RTC) : 1通道
    - 时钟 (时，分与秒)
    - 日历 (月，周，日与闰年)
    - 报警 (报警输出)
    - 报警中断

17. BACKUP 模块 (BUPMD)
    关闭除具体部分之外的电源即可实现低功耗。
    - 备份RAM : 8KB
    - 端口保持 (设置了备份模式时保持端口状态)
    - CEC功能
    - 遥控信号预处理程序功能
    - 触键唤醒功能
    - 实时时钟功能

18. 中断源
    - 内部84系数:优先级别设置可超过7级。
      (看门狗计时器中断除外)
    - 外部14系数:优先级别设置可超过7级。

19. 不可屏蔽中断 (NMI)
    不可屏蔽中断 (NMI)由 看门狗计时器或 $\overline{\text{NMI}}$ 引脚生成。

20. 输入/输出端口 (PORT) : 118 引脚
    I/O 引脚 : 102 引脚
    输入 引脚 : 16 引脚

21. 低功耗模式
    IDLE2，IDLE1，SLEEP，STOP，SLOW，BACKUP (BACKUP SLEEP，BACKUP STOP)

22. 时钟生成器 (CG)
    - 片上PLL (可以选择四元组或八元组。)
    - 时钟换档功能: 高速时钟可分为1/1，1/2，1/4或1/8。

23. 字节排序
    低端在前格式

24. 调试接口
    SWD / SWV / TRACE (DATA 4位)

25. 最大工作频率 : 64MHz (使用USB时48MHz。)

26. 工作电压范围
    2.7V ~ 3.6V (采用片上调节器)
    3.0V ~ 3.6V (使用USB时)

27. 温度范围
    - -40 ~ 85度 (闪存写入 / 擦除除外)
    - 0 ~ 70度 (闪存写入 / 擦除期间)

28. 封装

    P-LQFP144-2020-0.50E (20mm × 20mm,0.5 mm间距)

1.2 方块图

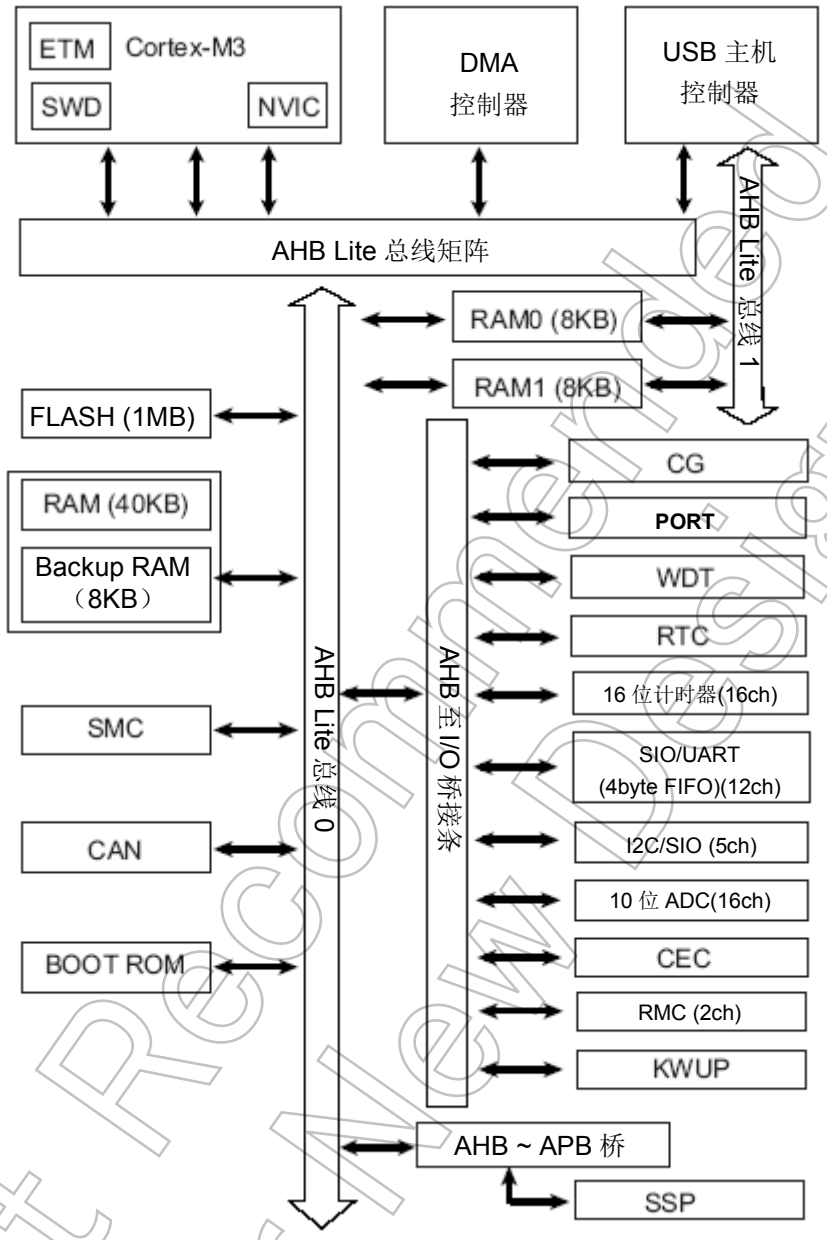


图1-1 TMPM364F10FG方块图

1.3 引脚布局图 (顶视图)

图1-2显示TMPM364F10FG的引脚布局图。

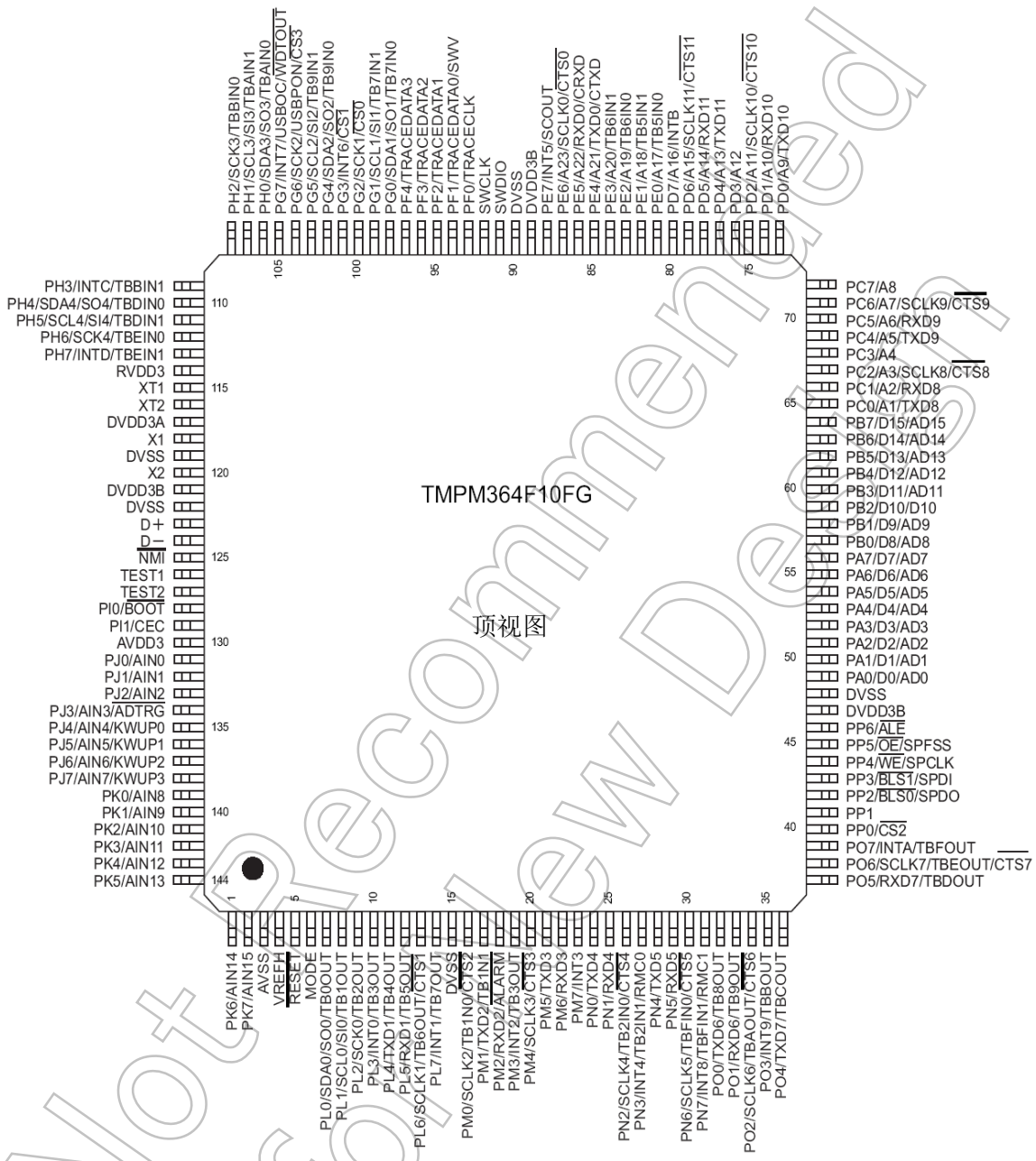


图1-2 引脚布局图 (LQFP144)

1.4 引脚名称与功能

表1-1 TMPM364F10FG通过引脚或端口的排序程序输入与输出引脚。

表1-1 通过引脚排序的引脚名称与功能 (1/10)

类型	引脚编号	引脚名称	输入/输出	功能
功能	1	PK6 AIN14	输入 输入	输入端口 模拟输入
功能	2	PK7 AIN15	输入 输入	输入端口 模拟输入
PS	3	AVSS	-	AD转换器GND引脚 (注)即使不使用AD转换器，AVSS也必须连接到GND。
PS	4	VREFH	-	AD转换器电源引脚 (注)即使不使用AD转换器，VREFH也必须连接到电源。
功能	5	$\overline{\text{RESET}}$	输入	复位输入引脚 (注)带有上拉与噪声滤波器 (约30 ns (典型值))
控制	6	MODE	输入	模式引脚 (注意)模式引脚必须连接GND。
功能	7	PL0 SDA0/SO0 TB0OUT	I/O I/O 输出	I/O端口 I2C模式数据 / SIO模式下的数据 16位计时器/事件计数器输出
功能	8	PL1 SCL0/SI0 TB1OUT	I/O I/O 输出	I/O端口 I2C模式时钟/SIO模式下的时钟 16-位计时器 / 事件计数器输出
功能	9	PL2 SCK0 TB2OUT	I/O I/O 输出	I/O端口 SIO模式时钟 16-位计时器 / 事件计数器输出
功能	10	PL3 INT0 TB3OUT	I/O 输入 输出	I/O端口 外部中断引脚 16-位计时器 / 事件计数器输出
功能	11	PL4 TXD1 TB4OUT	I/O 输出 输出	I/O端口 发送串行数据的串行通道 16-位计时器 / 事件计数器输出
功能	12	PL5 RXD1 TB5OUT	I/O 输入 输出	I/O端口 接收串行数据的串行通道 16-位计时器 / 事件计数器输出
功能	13	PL6 SCLK1 TB6OUT $\overline{\text{CTS1}}$	I/O I/O 输出 输入	I/O端口 串行通道时钟引脚 16-位计时器 / 事件计数器输出 串行通道信号交换输入引脚
功能	14	PL7 INT1 TB7OUT	I/O 输入 输出	I/O 端口 外部中断引脚 16-位计时器 / 事件计数器输出
PS	15	DVSS	-	GND引脚

表1-1 通过引脚排序的引脚名称与功能 (2/10)

类型	引脚编号	引脚名称	输入/输出	功能
功能	16	PM0	I/O	I/O端口
		SCLK2	I/O	串行通道时钟引脚
		TB1IN0	输入	输入16-位计时器 / 事件计数器捕捉触发器
		$\overline{\text{CTS2}}$	输入	串行通道信号交换输入引脚
功能	17	PM1	I/O	I/O端口
		TXD2	输出	发送串行数据的串行通道
		TB1IN1	输入	输入16-位计时器 / 事件计数器捕捉触发器
功能	18	PM2	I/O	I/O端口
		RXD2	输入	接收串行数据的串行通道
		$\overline{\text{ALARM}}$	输出	报警输出
功能	19	PM3	I/O	I/O端口
		INT2	输入	外部中断引脚
		TB3OUT	输出	16-位计时器 / 事件计数器输出
功能	20	PM4	I/O	I/O端口
		SCLK3	I/O	串行通道时钟引脚
		$\overline{\text{CTS3}}$	输入	串行通道信号交换输入引脚
功能	21	PM5	I/O	I/O端口
		TXD3	输出	发送串行数据的串行通道
功能	22	PM6	I/O	I/O端口
		RXD3	输入	接收串行数据的串行通道
功能	23	PM7	I/O	I/O端口
		INT3	输入	外部中断引脚
功能	24	PN0	I/O	I/O端口
		TXD4	输出	发送串行数据的串行通道
功能	25	PN1	I/O	I/O端口
		RXD4	输入	接收串行数据的串行通道
功能	26	PN2	I/O	I/O端口
		SCLK4	I/O	串行通道时钟引脚
		TB2IN0	输入	输入16-位计时器 / 事件计数器捕捉触发器
		$\overline{\text{CTS4}}$	输入	串行通道信号交换输入引脚
功能	27	PN3	I/O	I/O端口
		INT4	输入	外部中断引脚
		TB2IN1	输入	输入16-位计时器 / 事件计数器捕捉触发器
		RMC0	输入	输入信号 ~ 遥控器
功能	28	PN4	I/O	I/O端口
		TXD5	输出	发送串行数据的串行通道
功能	29	PN5	I/O	I/O端口
		RXD5	输入	接收串行数据的串行通道
功能	30	PN6	I/O	I/O端口
		SCLK5	I/O	串行通道时钟引脚
		TBFIN0	输入	输入16-位计时器 / 事件计数器捕捉触发器
		$\overline{\text{CTS5}}$	输入	串行通道信号交换输入引脚

表1-1 通过引脚排序的引脚名称与功能 (3/10).

类型	引脚编号	引脚名称	输入/输出	功能
功能	31	PN7	I/O	I/O端口
		INT8	输入	外部中断引脚
		TBFIN1	输入	输入16-位计时器 / 事件计数器捕捉触发器
		RMC1	输入	输入信号到遥控器
功能	32	PO0	I/O	I/O端口
		TXD6	输出	发送串行数据的串行通道
		TB8OUT	输出	16-位计时器/事件计数器输出
功能	33	PO1	I/O	I/O端口
		RXD6	输入	接收串行数据的串行通道
		TB9OUT	输出	16-位计时器 /事 件计数器输出
功能	34	PO2	I/O	I/O端口
		SCLK6	I/O	串行通道时钟引脚
		TBAOUT	输出	16-位计时器 / 事件计数器输出
		$\overline{\text{CTS6}}$	输入	串行通道信号交换输入引脚
功能	35	PO3	I/O	I/O端口
		INT9	输入	外部中断引脚
		TBBOUT	输出	16-位计时器 / 事件计数器输出
功能	36	PO4	I/O	I/O端口
		TXD7	输出	发送串行数据的串行通道
		TBCOUT	输出	16-位计时器 / 事件计数器输出
功能	37	PO5	I/O	I/O端口
		RXD7	输入	接收串行数据的串行通道
		TBDOUT	输出	16-位计时器/事件计数器输出
功能	38	PO6	I/O	I/O端口
		SCLK7	I/O	串行通道时钟引脚
		TBEOUT	输出	16-位计时器 / 事件计数器输出
		$\overline{\text{CTS7}}$	输入	串行通道信号交换输入引脚
功能	39	PO7	I/O	I/O端口
		INTA	输入	外部中断引脚
		TBFOUT	输出	16-位计时器 / 事件计数器输出
功能	40	PP0	I/O	I/O端口
		$\overline{\text{CS2}}$	输出	芯片选择引脚
功能	41	PP1	I/O	I/O端口
功能	42	PP2	I/O	I/O端口
		$\overline{\text{BLS0}}$	输出	字节通道引脚
		SPDO	输出	SSP数据输出引脚
功能	43	PP3	I/O	I/O端口
		$\overline{\text{BLS1}}$	输出	字节通道引脚
		SPDI	输入	SSP数据输入引脚
功能	44	PP4	I/O	I/O端口
		$\overline{\text{WE}}$	输出	写入选通引脚
		SPCLK	I/O	SSP 时钟引脚

表1-1 通过引脚排序的引脚名称与功能 (4/10)

类型	引脚编号	引脚名称	输入 / 输出	功能
功能	45	PP5 $\overline{OE}$ SPFSS	I/O 输出 I/O	I/O端口 输出启用引脚 SSP帧 / 从机选择引脚
功能	46	PP6 $\overline{ALE}$	I/O 输出	I/O端口 地址锁存器启用引脚
PS	47	DVDD3B	-	电源引脚
PS	48	DVSS	-	GND引脚
功能	49	PA0 D0 AD0	I/O I/O I/O	I/O端口 数据总线 地址与数据总线
功能	50	PA1 D1 AD1	I/O I/O I/O	I/O端口 数据总线 地址与数据总线
功能	51	PA2 D2 AD2	I/O I/O I/O	I/O端口 数据总线 地址与数据总线
功能	52	PA3 D3 AD3	I/O I/O I/O	I/O端口 数据总线 地址与数据总线
功能	53	PA4 D4 AD4	I/O I/O I/O	I/O端口 数据总线 地址与数据总线
功能	54	PA5 D5 AD5	I/O I/O I/O	I/O端口 数据总线 地址与数据总线
功能	55	PA6 D6 AD6	I/O I/O I/O	I/O端口 数据总线 地址与数据总线
功能	56	PA7 D7 AD7	I/O I/O I/O	I/O端口 数据总线 地址与数据总线
功能	57	PB0 D8 AD8	I/O I/O I/O	I/O端口 数据总线 地址与数据总线
功能	58	PB1 D9 AD9	I/O I/O I/O	I/O端口 数据总线 地址与数据总线
功能	59	PB2 D10 AD10	I/O I/O I/O	I/O端口 数据总线 地址与数据总线

表1-1 通过引脚排序的引脚名称与功能 (5/10)

类型	引脚编号	引脚名称	输入/输出	功能
功能	60	PB3	I/O	I/O端口
		D11	I/O	数据总线
		AD11	I/O	地址与数据总线
功能	61	PB4	I/O	I/O端口
		D12	I/O	数据总线
		AD12	I/O	地址与数据总线
功能	62	PB5	I/O	I/O端口
		D13	I/O	数据总线
		AD13	I/O	地址与数据总线
功能	63	PB6	I/O	I/O端口
		D14	I/O	数据总线
		AD14	I/O	地址与数据总线
功能	64	PB7	I/O	I/O端口
		D15	I/O	数据总线
		AD15	I/O	地址与数据总线
功能	65	PC0	I/O	I/O端口
		A1	输出	地址总线
		TXD8	输出	发送串行数据的串行通道
功能	66	PC1	I/O	I/O端口
		A2	输出	地址总线
		RXD8	输入	接收串行数据的串行通道
功能	67	PC2	I/O	I/O端口
		A3	输出	地址总线
		SCLK8	I/O	串行通道时钟引脚
		$\overline{\text{CTS8}}$	输入	串行通道信号交换输入引脚
功能	68	PC3	I/O	I/O端口
		A4	输出	地址总线
功能	69	PC4	I/O	I/O端口
		A5	输出	地址总线
		TXD9	输出	发送串行数据的串行通道
功能	70	PC5	I/O	I/O端口
		A6	输出	地址总线
		RXD9	输入	接收串行数据的串行通道
功能	71	PC6	I/O	I/O端口
		A7	输出	地址总线
		SCLK9	I/O	串行通道时钟引脚
		$\overline{\text{CTS9}}$	输入	串行通道信号交换输入引脚
功能	72	PC7	I/O	I/O端口
		A8	输出	地址总线
功能	73	PD0	I/O	I/O端口
		A9	输出	地址总线
		TXD10	输出	发送串行数据的串行通道

表1-1 通过引脚排序的引脚名称与功能 (6/10)

类型	引脚编号	引脚名称	输入/输出	功能
功能	74	PD1	I/O	I/O端口
		A10	输出	地址总线
		RXD10	输入	接收串行数据的串行通道
功能	75	PD2	I/O	I/O端口
		A11	输出	地址总线
		SCLK10	I/O	串行通道时钟引脚
		$\overline{\text{CTS10}}$	输入	串行通道信号交换输入引脚
功能	76	PD3	I/O	I/O端口
		A12	输出	地址总线
功能	77	PD4	I/O	I/O端口
		A13	输出	地址总线
		TXD11	输出	发送串行数据的串行通道
功能	78	PD5	I/O	I/O端口
		A14	输出	地址总线
		RXD11	输入	接收串行数据的串行通道
功能	79	PD6	I/O	I/O端口
		A15	输出	地址总线
		SCLK11	I/O	串行通道时钟引脚
		$\overline{\text{CTS11}}$	输入	串行通道信号交换输入引脚
功能	80	PD7	I/O	I/O端口
		A16	输出	地址总线
		INTB	输入	外部中断引脚
功能	81	PE0	I/O	I/O端口
		A17	输出	地址总线
		TB5IN0	输入	输入16-位计时器 / 事件计数器捕捉触发器
功能	82	PE1	I/O	I/O端口
		A18	输出	地址总线
		TB5IN1	输入	输入16-位计时器 / 事件计数器捕捉触发器
功能	83	PE2	I/O	I/O端口
		A19	输出	地址总线
		TB6IN0	输入	输入16-位计时器 / 事件计数器捕捉触发器
功能	84	PE3	I/O	I/O端口
		A20	输出	地址总线
		TB6IN1	输入	输入16-位计时器 / 事件计数器捕捉触发器
功能	85	PE4	I/O	I/O端口
		A21	输出	地址总线
		TXD0	输出	发送串行数据的串行通道
		CTXD	输出	发送数据的CAN
功能	86	PE5	I/O	I/O端口
		A22	输出	地址总线
		RXD0	输入	接收串行数据的串行通道
		CRXD	输入	接收数据的CAN

表1-1 通过引脚排序的引脚名称与功能 (7/10)

类型	引脚编号	引脚名称	输入/输出	功能
功能	87	PE6	I/O	I/O端口
		A23	输出	地址总线
		SCLK0	I/O	串行通道时钟引脚
		$\overline{\text{CTS0}}$	输入	串行通道信号交换输入引脚
功能	88	PE7	I/O	I/O端口
		INT5	输入	外部中断引脚
		SCOUT	输出	内时钟输出引脚
PS	89	DVDD3B	-	电源引脚
PS	90	DVSS	-	接地引脚
调试	91	SWDIO	I/O	调试引脚
调试	92	SWCLK	I/O	调试引脚
功能/调试	93	PF0	I/O	I/O端口
		TRACECLK	输出	调试引脚
功能/调试	94	PF1	I/O	I/O端口
		TRACEDATA0	输出	调试引脚
		SWV	输出	调试引脚
功能/调试	95	PF2	I/O	I/O端口
		TRACEDATA1	输出	调试引脚
功能/调试	96	PF3	I/O	I/O端口
		TRACEDATA2	输出	调试引脚
功能/调试	97	PF4	I/O	I/O端口
		TRACEDATA3	输出	调试引脚
功能	98	PG0	I/O	I/O端口
		SDA1/SO1	I/O	I2C模式数据/SIO模式下的数据
		TB7IN0	输入	输入16-位计时器 / 事件计数器捕捉触发器
功能	99	PG1	I/O	I/O端口
		SCL1/SI1	I/O	I2C模式时钟/SIO模式下的数据
		TB7IN1	输入	输入16-位计时器 / 事件计数器捕捉触发器
功能	100	PG2	I/O	I/O端口
		SCK1	I/O	SIO模式时钟
		CS0	输出	芯片选择引脚
功能	101	PG3	I/O	I/O端口
		INT6	输入	外部中断引脚
		$\overline{\text{CS1}}$	输出	芯片选择引脚
功能	102	PG4	I/O	I/O端口
		SDA2/SO2	I/O	I2C模式数据 / SIO模式下的数据
		TB9IN0	输入	输入16-位计时器 / 事件计数器捕捉触发器
功能	103	PG5	I/O	I/O端口
		SCL2/SI2	I/O	I2C模式时钟 / SIO模式下的数据
		TB9IN1	输入	输入16-位计时器 / 事件计数器捕捉触发器

表1-1 通过引脚排序的引脚名称与功能 (8/10)

类型	引脚编号	引脚名称	输入/输出	功能
功能	104	PG6	I/O	I/O端口
		SCK2	I/O	SIO模式时钟
		USBPON	输出	USB连接到总线检测 (VBus检测)
		$\overline{CS3}$	输出	芯片选择引脚
功能	105	PG7	I/O	I/O端口
		INT7	输入	外部中断引脚
		$\overline{USBOC}$	输入	USB过电流
		$\overline{WDTOUT}$	输出	看门狗计时器输出引脚
功能	106	PH0	I/O	I/O端口
		SDA3/SO3	I/O	I2C模式数据/SIO模式下的数据
		TBAIN0	输入	输入16-位计时器 / 事件计数器捕捉触发器
功能	107	PH1	I/O	I/O端口
		SCL3/SI3	I/O	I2C模式时钟/SIO模式下的数据
		TBAIN1	输入	输入16位计时器/事件计数器捕捉触发器
功能	108	PH2	I/O	I/O端口
		SCK3	I/O	如果串行总线接口以SIO模式工作，则输入与输出时钟。
		TBBIN0	输入	输入16-位计时器 / 事件计数器捕捉触发器
功能	109	PH3	I/O	I/O端口
		INTC	输入	外部中断引脚
		TBBIN1	输入	输入16-位计时器 / 事件计数器捕捉触发器
功能	110	PH4	I/O	I/O端口
		SDA4/SO4	I/O	I2C模式数据/SIO模式下的数据
		TBDIN0	输入	输入16-位计时器 / 事件计数器捕捉触发器
功能	111	PH5	I/O	I/O端口
		SCL4/SI4	I/O	I2C模式时钟/SIO模式下的数据
		TBDIN1	输入	输入16-位计时器 / 事件计数器捕捉触发器
功能	112	PH6	I/O	I/O端口
		SCK4	I/O	SIO模式时钟
		TBEIN0	输入	输入16-位计时器 / 事件计数器捕捉触发器
功能	113	PH7	I/O	I/O端口
		INTD	输入	外部中断引脚
		TBEIN1	输入	输入16-位计时器 / 事件计数器捕捉触发器
PS	114	RVDD3	-	电源引脚
时钟	115	XT1	输入	连接到低速振荡器。
时钟	116	XT2	输出	连接到低速振荡器。
PS	117	DVDD3A	-	电源引脚
时钟	118	X1	输入	连接到高速振荡器。
PS	119	DVSS	-	GND引脚
时钟	120	X2	输出	连接到高速振荡器。
PS	121	DVDD3B	-	电源引脚

表1-1 通过引脚排序的引脚名称与功能 (9/10)

类型	引脚编号	引脚名称	输入/输出	功能
PS	122	DVSS	-	GND引脚
功能	123	D+	I/O	USB数据 正
功能	124	D-	I/O	USB数据 负
控制	125	$\overline{\text{NMI}}$	输入	不可屏蔽中断 (注)带有噪声滤波器 (约 30ns (典型值))
控制	126	TEST1	-	测试引脚 (注)测试引脚必须保持开路。
控制	127	TEST2	-	测试引脚 (注)测试引脚必须保持开路。
功能	128	PI0 $\overline{\text{BOOT}}$	I/O 输入	I/O端口 设置引导启动引脚 $\overline{\text{RESET}}$ 引脚的上升沿取样“低“，则TMPM364F10FG进入单引导启动模式。
功能	129	PI1 CEC	I/O I/O	I/O端口 CEC 引脚 (注) N通道开路漏极端口
PS	130	AVDD3	-	模拟数字转换器电源引脚 (注) 即使不使用模拟数字转换器，AVDD3也必须连接到电源。
功能	131	PJ0 AIN0	输入 输入	输入端口 模拟输入
功能	132	PJ1 AIN1	输入 输入	输入端口 模拟输入
功能	133	PJ2 AIN2	输入 输入	输入端口 模拟输入
功能	134	PJ3 AIN3 $\overline{\text{ADTRG}}$	输入 输入 输入	输入端口 模拟输入 AD转换器的外触发输入
功能	135	PJ4 AIN4 KWUP0	输入 输入 输入	输入端口 模拟输入 触键唤醒引脚
功能	136	PJ5 AIN5 KWUP1	输入 输入 输入	输入端口 模拟输入 触键唤醒引脚
功能	137	PJ6 AIN6 KWUP2	输入 输入 输入	输入端口 模拟输入 触键唤醒引脚
功能	138	PJ7 AIN7 KWUP3	输入 输入 输入	输入端口 模拟输入 触键唤醒引脚
功能	139	PK0 AIN8	输入 输入	输入端口 模拟输入

表1-1 通过引脚排序的引脚名称与功能 (10/10)

类型	引脚编号	引脚名称	输入/输出	功能
功能	140	PK1 AIN9	输入 输入	输入端口 模拟输入
功能	141	PK2 AIN10	输入 输入	输入端口 模拟输入
功能	142	PK3 AIN11	输入 输入	输入端口 模拟输入
功能	143	PK4 AIN12	输入 输入	输入端口 模拟输入
功能	144	PK5 AIN13	输入 输入	输入端口 模拟输入

1.5 引脚编号与电源引脚

表1-2 引脚编号与电源

电源	电压范围	引脚编号	引脚名称
DVDD3B	2.7V ~ 3.6V (使用USB时: 3.0 ~ 3.6V)	47,89,121	PA,PB,PC,PD,PE,PF,PG,PH,PI,PL,PM PN,PO,PP,XT1,XT2, $\overline{\text{RESET}}$, $\overline{\text{NMI}}$,MODE
DVDD3A		117	X1,X2
AVDD3		130	PJ,PK
RVDD3		114	-

2. 处理器内核

TX03系列具有高性能的32位处理器内核 (ARM Cortex-M3处理器内核)。有关本处理器内核的工作信息，请参阅ARM有限公司出版的 “Cortex-M3 技术参考手册。本章只叙述了TX03系列未在那些文件中阐明的各个功能。

2.1 处理器内核信息

下表显示了TMPM364F10FG中处理器内核的修订情况。

请参阅CPU内核与结构的详细信息，参阅以下URL中的ARM手册 “Cortex-M系列处理器“:

http: //infocenter.arm.com/help/index.jsp

产品名称	内核修订
TMPM364F10FG	r2p0

2.2 可配置选件

Cortex-M3 内核具有可选用的程序块。修订版r2p0的可选用程序块为ETM™，MPU与WIC。下表显示了TMPM364F10FG中的可配置选用件。

可配置选件	执行
FPB	两个文字比较器 六个指令比较器
DWT	四个比较器
ITM	有
MPU	无
ETM	有
AHB-AP	有
AHB迹线宏单元接口	无
TPIU	有
WIC	无
调试端口	串行线

2.3 异常/ 中断

异常与中断在下节中叙述。

2.3.1 中断输入编号

中断输入编号可从Cortex-M3内核中的1～240范围内任选确定。

TMPM364F10FG具有98个中断输入。中断输入的编号反映在NVIC寄存器的<INTLINESNUM [4:0]>位中。本产品中，如果读取<INTLINESNUM[4:0]>位，则读出0×01。

2.3.2 优先级中断位的编号

Cortex-M3内核可从3 位 ～8 位中任选配置中断位优先级的编号。

TMPM364f10具有3个优先级中断位。优先级中断位的编号用于给中断优先寄存器与系统处理程序优先寄存器中的优先级赋值。

2.3.3 SysTick

Cortex-M3内核有一个能生成SysTick异常的SysTick计时器。

有关SysTick异常的详细情况，参阅异常中的 “SysTick “一节以及NVIC寄存器中的SysTick寄存器。

2.3.4 SYSRESETREQ.

在设置了应用中断<SYSRESETREQ>位与复位控制寄存器时，Cortex-M3内核即输出SYSRESETREQ信号。

在SYSRESETREQ信号被输出时，TMPM364F10FG则提供相同的操作。

注:在慢速模式下，不采用由<SYSRESETREQ>进行的复位操作。

2.3.5 LOCKUP

不能挽回的异常生成时，Cortex-M3内核则输出锁位信号，显示软件中内藏了严重错误。

TMPM364F10FG并不使用该信号。要从锁位状态返回，必须使用不可屏蔽中断 (NMI)或进行复位。

2.3.6 辅助故障状态寄存器

Cortex-M3内核配备有辅助故障状态寄存器，以便给软件提供额外的系统故障信息。

然而，TMPM364F10FG并没有定义此功能。如果读取辅助故障状态寄存器，则始终读出 “0×0000_0000“。

2.4 事件

Cortex-M3内核具有事件输出信号与事件输入信号。事件输出信号通过SEV指令执行来输出。当输入某个事件得后，内核即从由WFE指令所引起的低功耗模式返回。

TMPM364F10FG并不使用事件输出信号与事件输入信号。请勿使用SEV指令与WFE指令。

2.5 电源管理

Cortex-M3内核配备有功率调节系统，该系统使用正在睡眠信号与深睡眠信号。深睡眠信号在系统控制寄存器的<SLEEPDEEP>位设置时输出。

这些信号在以下环境中输出:

-中断等待 (WFI)指令执行

-事件等待 (WFE)指令执行

-中断服务程序 (ISR)在系统控制寄存器的<SLEEPONEXIT>位设置后退出时的时间

TMPM364F10FG并不使用SLEEPDEEP信号，这样就不必设置<SLEEPDEEP>位。而且，事件信号也不使用，请不要使用WFE指令。

电源管理的详细情况参阅 “时钟/模式控制“一章。

2.6 独占通道

在Cortex-M3内核中，DCode总线系统支持独占通道。但TMPM364F10FG并不使用此功能。

3. 调试界面

3.1 规格概述

该TMPM364F10FG含有串行线调试端口 (SW-DP)装置，用作与调试工具的接口，以及嵌入跟踪宏单元™ (ETM)装置，用于指令跟踪输出。跟踪数据通过单片式跟踪端口接口装置 (TPIU)输出到专用引脚 (TRACEDATA[3:0])。

SW-DP，ETM与TPIU的详细情况参阅 “Cortex-M3技术参考手册“。

3.2 SW-DP

SW-DP支持双-引脚串行线调试端口 (SWCLK，SWDIO)。

3.3 ETM

ETM支持四个数据信号引脚 (TRACEDATA[3:0])，一个时钟信号引脚TRACECLK)以及来自SWV的跟踪输出。

3.4 引脚功能

调试接口引脚还可用作通用端口 (PF0 ~ PF4)。

表3-1 SW-DP，ETM调试功能

SW-DP 引脚名称	端口名称	SW调试功能	
		I / O	注释
SWDIO	-	I / O	串行线数据输入/输出 (始终上拉)
SWCLK	-	输入	串行线时钟 (始终下拉)
TRACECLK	PF0	输出	跟踪时钟输出
TRACEDATA0 / SWV	PF1	输出	跟踪数据输出0 / 串行线查看器输出
TRACEDATA1	PF2	输出	跟踪数据输出1
TRACEDATA2	PF3	输出	跟踪数据输出2
TRACEDATA3	PF4	输出	跟踪数据输出3

复位后，PF0 ~ PF4引脚被配置为通用端口。调试接口引脚的各个功能需要根据需要进行编程。.

表3-2概括了调试接口引脚功能以及复位之后的相关端口设定。

表3-2 调试接口引脚功能与复位之后的相关端口设定

端口名称 (位名称)	调试功能	相关端口复位之后的设定值				
		功能 (PxFR)	输入 (PxIE)	输出 (PxCR)	上拉 (PxPUP)	下拉 (PxPDN)
PF0	TRACECLK	0	0	0	0	-
PF1	TRACEDATA0 / SWV	0	0	0	0	-
PF2	TRACEDATA1	0	0	0	0	-
PF3	TRACEDATA2	0	0	0	0	-
PF4	TRACEDATA3	0	0	0	0	-

-: 不指定

3.5 停机模式中的外围功能

在Cortex-M3内核进入停机模式时，看门狗计时器 (WDT)自动停止。其它外围功能则继续工作。

3.6 复位向量中止

$\overline{\text{RESET}}$ 引脚引起的复位有效时，TMPM330FDFG/FYFG/FWFG被禁止采用调试工具传输。在使用复位向量设定停止时，则在复位之后设置以下程序;从调试工具设置中止点，然后设置应用中断与复位控制寄存器的<SYSRESETREQ>位，以再次进行复位。

注:不要在慢速模式中采用<SYSRESETREQ>复位。

3.7 与调试工具连接

对于与调试工具连接，见生产厂商建议。

调试接口引脚包含上拉电阻器与下拉电阻器。在调试接口引脚与外部上拉或下拉连接时，请注意输入电平。

4. 内存地图

4.1 内存地图

TMPM364F10FG的内存地图基于ARM Cortex-M3处理器内核内存地图。

内部ROM分别映射到Cortex-M3内核内存的编码，内部RAM映射到SRAM区，特殊功能寄存器 (SFR)则映射到周围区。

特殊功能寄存器 (SFR)指明外围功能的I/O端口与控制寄存器。SRAM与SFR区则全都包括在位带区内。

CPU寄存器区为处理器内核的内部寄存器区域。

有关各个区的更多信息，见 “Cortex-M3技术参考手册“。

注意进入到注明为 “故障“的各个区域，如果内存故障启用，则引起内存故障，如果内存故障禁用，则引起硬故障。勿进入供应商特定的区域。

4.1.1 TMPM36410FG的内存地图

图4-1显示TMPM364F10FG的内存地图。

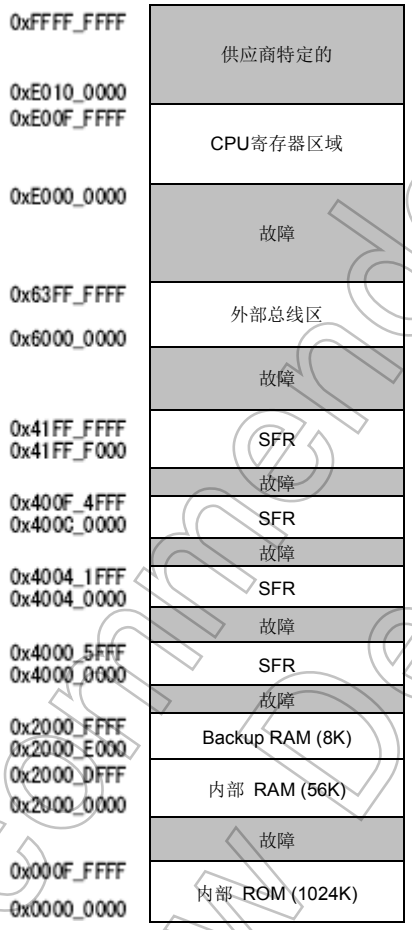


图4-1 内存地图

4.2 SFR区详情

该区段包含SFR中的地址表，这些地址 (0×4000_0000 ~ 0×4000_5FFF，0×4004_0000 ~ 0×4004_1FFF，0×400C_0000～0×400F_4FFF，0×41FF_F000～0×41FF_FFFF)均分配给外围功能。

禁止进入表4-1中的保留区。 关于SFR区，读取表4-1中没有描述的区即得出未定义值。写入这些区则置之不理。

表4-1 SFR区详情

开始地址	结束地址	外围	保留
0×4000_0000	0×4000_0FFF	DMAC	0×4000_0028 ~ 0×4000_002F 0×4000_0034 ~ 0×4000_0037 0×4000_0500 ~ 0×4000_050F 0×4000_0FE0 ~ 0×4000_0FFF
0×4000_1000	0×4000_1FFF	SMC	0×4000_1000 ~ 0×4000_1003 0×4000_1008 ~ 0×4000_100F 0×4000_1020 ~ 0×4000_1023 0×4000_1200 ~ 0×4000_1207 0×4000_1E00 ~ 0×4000_1E0B 0×4000_1FE0 ~ 0×4000_1FFF
0×4000_2000	0×4000_2FFF	CAN	
0×4000_3000	0×4000_3FFF	USB	0×4000_3058 ~ 0×4000_305F
0×4000_4000	0×4000_5FFF	保留	
0×4004_0000	0×4004_0FFF	SSP	0×4004_0028 ~ 0×4004_0FFF
0×4004_1000	0×4004_1FFF	保留	
0×400C_0000	0×400C_FFFF	端口	
0×400D_0000	0×400D_FFFF	计时器 B (16ch)	
0×400E_0000	0×400E_04FF	I2C/SIO (5ch)	
0×400E_1000	0×400E_1BFF	SIO/UART (12ch)	
0×400E_2000	0×400E_203F	CEC	
0×400E_3000	0×400E_31FF	RMC (2ch)	
0×400F_0000	0×400F_005B	ADC (16ch)	0×400F_001C ~ 0×400F_001F 0×400F_0024 ~ 0×400F_002F
0×400F_1000	0×400F_108F	KWUP	0×400F_1010 ~ 0×400F_107F
0×400F_2000	0×400F_2007	WDT	
0×400F_3000	0×400F_300F	RTC	0×400F_300D
0×400F_4000	0×400F_4037	CG	0×400F_402E ~ 0×400F_402F 0×400F_4036 ~ 0×4000_4FFF
0×41FF_F000	0×41FF_F03F	FLASH	0×41FF_F000 ~ 0×41FF_F007 0×41FF_F014 ~ 0×41FF_F017 0×41FF_F018 ~ 0×41FF_F01B 0×41FF_F024 ~ 0×41FF_F02C 0×41FF_F033 ~ 0×41FF_F037
0×41FF_F040	0×41FF_F057	保留	
0×41FF_F058	0×41FF_F05B	RAMWAIT	
0×41FF_F060	0×41FF_F093	保留	
0×41FF_F0A0	0×41FF_F0BB	保留	
0×41FF_F100	0×41FF_F103	SMCMOD	

5. 复位

TMPM364F10FG有三个复位来源:外部复位引脚（$\overline{\text{RESET}}$）, 看门狗计时器 (WDT)与应用中断与复位控制寄存器中的设定<SYSRESETREQ>。

源于WDT的复位，见WDT章节。

源于<SYSRESETREQ>复位，见 “Cortex-M3 技术参考手册“。

注:不要在慢速模式中采用<SYSRESETREQ>复位。

5.1 冷复位

上电顺序必须考虑内部调节器与振荡器达到稳定的时间。在TMPM364F10FG中，内部调节器要求至少700 μs时间达到稳定。

要获得稳定振荡所需要的时间因系统而异。冷复位时，外部复位引脚必须保持足够长的 “低“持续时间，使得内部调节器与振荡器达到稳定。

在外部复位（$\overline{\text{RESET}}$）信号释放之后，内部复位信号需再保持400μs的时间去证实。

图5-1显示上电顺序。

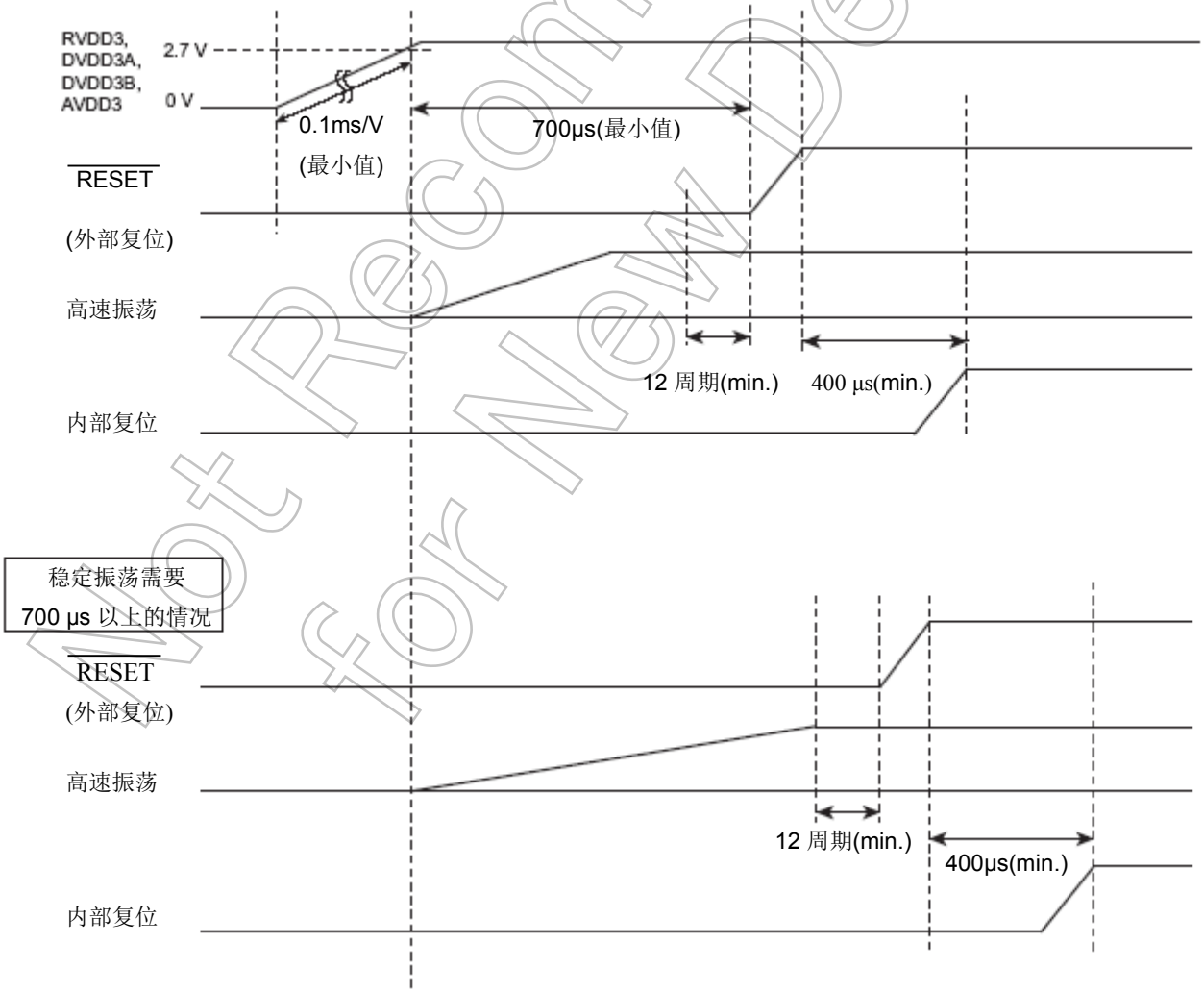


图5-1 冷复位顺序

注1: 电源必须以0.1ms/V或更低的速度上升 (从0V ~ 2.7V)。

注2: $\overline{\text{RESET}}$ 引脚被固定到 “低“的同时打开电源。在所有电源均稳定在工作电压范围之内时释放复位。

5.2 热复位

5.2.1 复位期限

作为一项前提条件，要确保供电电压在工作范围之内，内部高频振荡器在提供稳定振荡。

要将TMPM364F10FG复位，则维护 $\overline{\text{RESET}}$ 信号 (低电平激活) 12个系统时钟的最小持续时间。

在外部复位 ($\overline{\text{RESET}}$)信号释放之后，内部复位信号需再保持400μs的时间去证实。

5.3 复位后

热复位使Cortex-M3处理器内核的大部分系统控制寄存器与内部功能寄存器初始化。

该处理器内核的系统调试元件 (FPB，DWT，ITM)寄存器，时钟生成器的CGRSTFLG寄存器与FCSECBIT寄存器均仅经过冷复位进行初始化。

在复位之后，锁相环倍增电路不处于活动状态，如果需要，必须在CGPLLSEL寄存器中进行激活启用。

在复位异常处理完成时，该程序分支转移到复位中断服务程序。

注:复位操作可能改变内部RAM的状态。

6. 时钟/模式控制

6.1 特点

时钟/模式控制模块可选择时钟齿轮，预分频器时钟及PLL时钟倍频电路和振荡器的预热。

还有一通过模式转换来降低功耗的低功耗模式。

本章对如何控制时钟操作模式和模式转换进行了说明。

时钟/模式控制模块具有以下功能:

- 控制系统时钟
- 控制预分频器时钟
- 控制PLL倍频电路
- 控制预热定时器

除NORMAL模式外，TMPM364F10FG还可根据其使用条件工作低功耗模式来降低功耗。

6.2 寄存器

6.2.1 寄存器列表

下表列出了与时钟齿轮有关的寄存器和地址。

基址 = 0×400F_4000

寄存器名称		地址 (基址+)
系统控制寄存器	CGSYSCR	0×0000
振荡控制寄存器	CGOSCCR	0×0004
待机控制寄存器	CGSTBYCR	0×0008
PLL选择寄存器	CGPLLSEL	0×000C
系统时钟选择寄存器	CGCKSEL	0×0010

6.2.2 CGSYSCR (系统控制寄存器)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	USBHRES	-	-	FCSTOP	-	-	SCOSEL	
复位后	0	0	0	0	0	0	0	1
	15	14	13	12	11	10	9	8
比特符号	-	-	FPSEL1	FPSEL0	-	PRCK		
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	-	-	-	-	-	GEAR		
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-24	–	R	读作 “0“。
23	USBHRES	R/W	USB 主机控制器复位 0: 解除复位 1: 复位 可用于对 USB 主机控制器进行复位。
22-21	–	R	读作 “0“。
20	FCSTOP	R/W	停止AD转换器时钟的工作 0: 已操作 1: 已停止 可停止AD转换器时钟。复位后，操作AD转换器时钟。 当此位设置到 “1“时，应确认AD转换器操作是否完成或停止。
19-18	–	R	读作 “0“。
17-16	SCOSEL[1:0]	R/W	SCOUT 输出 00: fs 01: fsys/2 10: fsys 11: φT0 自SCOUT引脚实现规定的时钟输出。
15-14	–	R	读作 “0“。
13	FPSEL1	R/W	选择φT0 源时钟 0: 通过 <FPSEL0> 选定时钟 1: fs
12	FPSEL0	R/W	选择 fperiph 源时钟 0: fgear 1: fc
11	–	R	读作 “0“。
10- 8	PRCK[2:0]	R/W	预分频器时钟 000: fperiph 100: fperiph/16 001:fperiph/2 101: fperiph/32 010: fperiph/4 110:保留 011: fperiph/8 111:保留 将预分频时钟指定到外围I/O。
7-3	–	R	读作 “0“。
2-0	GEAR[2:0]	R/W	高速齿轮时钟 (fgear) 000:fc 100: fc/2 001:保留 101: fc/4 010:保留 110: fc/8 011:保留 111:保留

6.2.3 CGOSCCR (振荡控制寄存器)

	31	30	29	28	27	26	25	24
比特符号	WUPT							
复位后	1	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	WUPT							
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	WUPTL		-	-	-	-	XEN	XTEN
复位后	0	0	0	0	0	0	1	1
	7	6	5	4	3	2	1	0
比特符号	-	-	-	-	WUPSEL	PLLON	WUEF	WUEON
复位后	0	0	1	1	0	0	0	0

位	比特符号	类型	功能
31-20	WUPT[11:0]	R/W	规定预热计时器的计时。 高速预热计数器数值 低速预热计数器数值 (上12位) 高速预热计数器为16-位计数器，低速为18-位计数器。两种计数器的下4位被屏蔽。高速上12位，低速下4位与预热计数器进行比较。
19-16	−	R	读作 “0“。
15-14	WUPTL[1:0]	R/W	规定预热计时器的计时。 低速预热计数器数值 (下2位)
13-12	−	R/W	写作 “0“。
11-10	−	R	读作 “0“。
9	XTEN	R/W	低-速振荡器 0: 停止 1: 振荡
8	XEN	R/W	高-速振荡器 0: 停止 1: 振荡
7-4	−	R/W	写作 “0011“。
3	WUPSEL	R/W	预热计数器 0: 高速 (fosc) 1: 低速 (fs) 规定振荡器进行预热。 通过规定的振荡器生成的时钟用于预计时器计数。
2	PLLON	R/W	PLL工作 0: 停止 1: 振荡
1	WUEF	R	预热定时器 (WUP)状态 0: 预热完成。 1: 预热工作。 促使监控预热定时器状态。
0	WUEON	W	预热定时器的工作 0: 不指定 1: 启动预热。 促使启动预热定时器。 读作 “0“。

注1: 预热时间，见 “6.3.4“预热功能。

注2: 设定 PLL 倍增数值后，需要保持CGOSCCR <PLLON> = “0“ (PLL停止)在100 μs以上作为PLL初始化稳定时间。

6.2.4 CGSTBYCR (待机控制寄存器)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	PTKEEP	DRVE
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	RXTEN	RXEN
复位后	0	0	0	0	0	0	0	1
	7	6	5	4	3	2	1	0
比特符号	-	-	-	-	-	STBY		
复位后	0	0	0	0	0	0	1	1

位	比特符号	类型	功能
31-18	−	R	读作 “0“。
17	PTKEEP	R/W	在备份模式下I/O端口控制。 0:输出输出锁存器中的内容。 1:当 CGSTBYCR<PTKEEP>转换从 “0“ ~ “1“时保持地块状态不变。
16	DRVE	R/W	“STOP “模式下引脚状态 (注) 0: 非激活 1: 激活
15-10	−	R	读作 “0“。
9	RXTEN	R/W	在解除 “停止“模式后工作低速振荡器。 0: 停止 1: 振荡
8	RXEN	R/W	释放STOP模式后工作高速振荡器。 0: 停止 1: 振荡
7-3	−	R	读作 “0“。
2-0	STBY[2:0]	R/W	低功耗模式 000: 保留 001: 停止 010: 休眠 011: IDLE2 100: 保留 101: BACKUP STOP 110: BACKUP SLEEP 111: IDLE1

注1: 当 CGSTBYCR<PTKEEP>转换从 “0“ 到 “1“时保持其状态的I/O端口为除I，J， L，M及N以外的所有端口。有关释放 CGSTBYCR<PTKEEP>，见 “BACKUP模块“说明。

注2: 上述表中保留的数值不得进行设置。

6.2.5 CGPLLSEL (PLL选择寄存器)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	RS				-	IS		C2S
复位后	0	1	1	1	0	0	1	0
	7	6	5	4	3	2	1	0
比特符号	ND					-	-	PLLSEL
复位后	0	0	0	1	1	1	1	0

位	比特符号	类型	功能
31-16	−	R	读作 “0“。
15-12	RS[3:0]	R/W	PLL倍增时钟 0111: 4倍 1010: 8倍 其它:保留
11	−	R	读作 “0“。
10-9	IS[1:0]	R/W	PLL倍增时钟 00: 8 倍 01: 4 倍 其它:保留
8	C2S	R/W	PLL倍增时钟 0: 4 倍 1: 8 倍
7-3	ND[4:0]	R/W	PLL倍增时钟 00011: 4 倍 00111: 8 倍 其它:保留
2-1	−	R/W	写入 “1 “。
0	PLLSEL	R/W	使用PLL功能 0: f_{osc} 1: f_{PLL} 可用于指定启用或禁用通过PLL倍增时钟。 fosc 可在重设后自动进行设定。使用PLL功能必须进行复位。

注1: 选择表6-1 中所显示的PLL倍增数值。

注2: 选择当 CGOSCCR<PLLON> = “0“ (PLL 停止)时的PLL倍增数值。

注3: 在设定好PLL倍增数值以后，作为PLL的初始化稳定时间， 必须使 CGOSCCR <PLLON> = “0“ (PLL 停止)这一状态的保持时间超过 100 μs。

6.2.6 CGCKSEL (系统时钟选择寄存器)

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

位	比特符号	类型	功能
31-2	–	R	读作 “0“。
1	SYSCK	R/W	系统时钟 (fsys) 0: f_{gear} 1: f_S 用于指定系统时钟。 修改 <SYSCK> 前，f_{gear} 和 f_S的频率必须稳定。
0	SYSCKFLG	R	系统时钟的状态 0: f_{gear} 1: f_S 显示系统时钟状态。 通过 <SYSCK> 对振荡器进行切换使生成时滞完成。 当 <SYSCLKFLG> 读取到 <SYSCK> 中规定的振荡器的输出时，表示切换完成。

6.3 时钟控制

6.3.1 时钟类型

各时钟定义:

f_{osc}	:自X1，X2 引脚输出的时钟
f_S	:自XT1，XT2 引脚(低速时钟)输出的时钟
f_{PLL}	:由PLL生成的四倍或八倍增频时钟
f_C	:由CGPLLSEL<PLLSEL> (高速时钟) 规定的时钟
f_{gear}	:由CGSYSCR<GEAR[2:0]> (齿轮时钟)规定的时钟
f_{sys}	:由CGCKSEL<SYSCK> (系统时钟) 规定的时钟
f_{periph}	:由CGSYSCR<FPSEL0>规定的时钟
φT0	:由 CGSYSCR<FPSEL1> (预分频器时钟)规定的时钟

齿轮时钟 fgear 以及预分频器时钟 φT0 可以采用以下方式进行分频设置。

齿轮时钟	:fc，fc/2，fc/4，fc/8；
预分频器时钟	:fs，fperiph，fperiph/2，fperiph/4，fperiph/8，fperiph/16，fperiph/32

6.3.2 复位后的初始值

复位操作会对时钟进行初始化设置。

高速振荡器	: 正在振荡
低速振荡器	: 正在振荡
PLL (锁相环电路)	: 停止
高速时钟齿轮	: f_c (无分频)

复位操作会引起除低速时钟 (fs)以外的所有时钟配置与f_{osc}相同。

$f_C = f_{osc}$

$f_{sys} = f_{osc}$

$\varphi T0 = f_{osc}$

6.3.3 时钟系统示意图

图 6-1 显示时钟系统的示意图。

用箭头表示的输入时钟选择器在复位后设置为默认值。

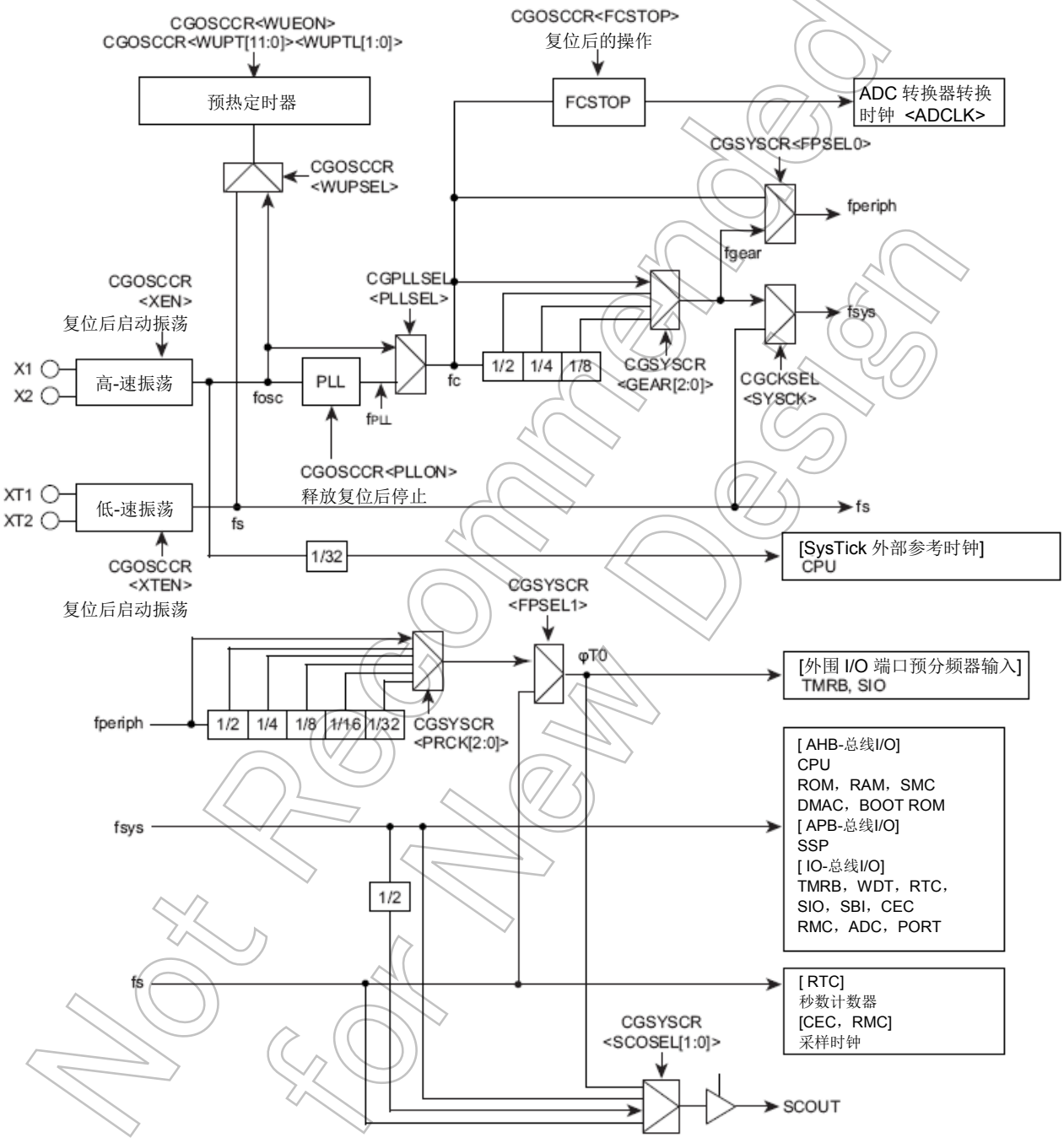


图 6-1 时钟方块图

6.3.4 预热功能

预热功能确保了振荡器和带预热定时器的PLL的稳定时间。

详情见 “6.6.8 预热” 说明。

本章描述了如何使用预热功能。

1. 规定了计数时钟

   规定了CGOSCCR<WUPSEL> 中预热计数器的计数时钟。

2. 规定了预热计数器值数

   预热时间用舍弃4位的方法通过以下公式计算，设置到<WUPT [11:0]><WUPTL[1:0]> 的位上

$$\text{预热周期数} = \frac{\text{预热时间}}{\text{预热时钟周期}}$$

3. 启动预热功能并确认预热是否完成

   当将CGOSCCR<WUEON> 设为 “1”时，预热启动一个计数。 采用 CGOSCCR <WUEF> 确认预热的启动和完成情况。<WUEF>=1 表示正在进行预热，<WUEF>=0 则表示预热完成。

   变更时钟时，可在CGCKSEL<SYSCKFLG> 中监控当前的系统时钟。

以下介绍了预热设置和示例。

1. 在从SLOW模式转换到NORMAL模式时，利用8MHz高频振荡器将预热时间设置为5ms。

   预热计数器的值数如下所示。

$$\frac{\text{预热时间}}{\text{预热时钟周期}} = \frac{5\text{ms}}{1/8\text{MHz}} = 40{,}000\text{周期} = 0\times9C40$$

   高频预热计数器为16位计数器，其下4位忽略不计。因此，将0×9C40，0×9C4中的上12位设置到CGOSCCR<WUPT[11:0]>。

<示例> 从SLOW模式转换 ~ NORMAL模式

```
CGOSCCR<WUPT[11:0]> = "0×9C4"   : 预热时间设置
C 读取 CGOSCCR<WUPT[11:0]>      : 确认预热时间反映
CGOSCCR<XEN>="1"                : 启用高速振荡 (fosc)
CGOSCCR<WUEON>="1"              : 启用预热计数 (WUP)
C 读取 CGOSCCR<WUEF>            : 等待 "0" (WUP 结束)
CGOSCCR<SYSCK>="0 "             : 系统时钟变为高速 (fgear)
C 读取 CGOSCCR<SYSCKFLG>        : 等待 "0" (当前时钟为fgear)
CGOSCCR<XTEN>="0"               : 禁用低速振荡 (fs) (在双时钟模式下无此要求。)
```

2. 自NORMAL模式转换到SLOW模式，采用32.768kHz低频振荡器将预热时间设置为1s。

   预热计数器的值数如下所示。

$$\frac{\text{预热时间}}{\text{预热时钟周期}} = \frac{1\text{s}}{1/32.768\text{kHz}} = 32{,}768\ \text{周期} = 0\times8000$$

   低频预热计数器为18位，其下4位忽略不计。

   因此，将0×8000，0×0800的上14位设置到CGOSCCR<WUPT[11:0]><WUPTL[1:0]>。

<示例> 自NORMAL模式转换到SLOW模式

CGOSCCR<WUPT[11:0]> =“0×200“ ：预热时间设置 (上 12 位)

CGOSCCR<WUPTL[1:0]> =“00“ ：预热时间设置 (下 2 位)

C 读取 CGOSCCR<WUPT><WUPTL> ：检查预热时间设置

CGOSCCR<XTEN>=“1“ ：启用低速振荡 (f_S)

CGOSCCR<WUPSEL>=“1“ ：选择XT1用于预热时钟

CGOSCCR<WUEON>=“1“ ：启用预热计数 (WUP)

C 读取 CGOSCCR<WUEF> ：等待 “0“ (WUP 结束)

CGOSCCR<SYSCK>= “1“ ：系统时钟变为低速 (f_S)

C 读取 CGOSCCR<SYSCKFLG> ：等待 “1“ (当前时钟为f_S)

CGOSCCR<XEN>=“0“ ：禁用高速振荡 (f_C)
(在双时钟模式下无此要求。)

注 1: 在使用外部时钟时，不要求稳定预热时间。

注 2: 预热定时器根据振荡器时钟操作，所以当振荡频率发生波动时，则有可能导致预热定时器出现错误。因此，预热时间应当取近似时间。

注 3: 将预热计数值数设定到 OSCCR<WUPT><WUPTL> 后，直到确认反映数值，接着变更为通过WFI指令的待机模式。

注 4: 在切换系统时钟时，应确保通过读取CGCKSEL<SYSCKFLG>来完成切换。

6.3.5 时钟倍频电路 (PLL)

这一电路主要用于输出相当于高速振荡器输出时钟 (f_{osc})四倍 / 八倍频率的 f_{PLL} 时钟。为此，即使在输入频率较低的情况下，内部时钟也可以将其转变为高速频率。

6.3.5.1 如何配置PLL功能

复位后PLL禁用。

启用PLL，在CGOSCCR<PLLON>为 “0”时，将 CGPLLSEL<RS[3:0]><IS[1:0]><C2S><ND[4:0]>设为倍增数值。接着，PLL初始化100μs后将 <PLLON>设置为 “1“。经过200μs锁定时间后，将 CGPLLSEL<PLLSEL> 设置为 “1“，f_{PLL}自fosc乘以 4 或 8后使用。

PLL初始化需要一定的时间，所以应确保借助预热功能或其它方法。

对于4或8倍增数值，仅允许以下设置。

乘数	<RS[3:0]>	<IS[1:0]>	<C2S>	<ND[4:0]>
4	0111	00	0	0_0011
8	1010	01	1	0_0111

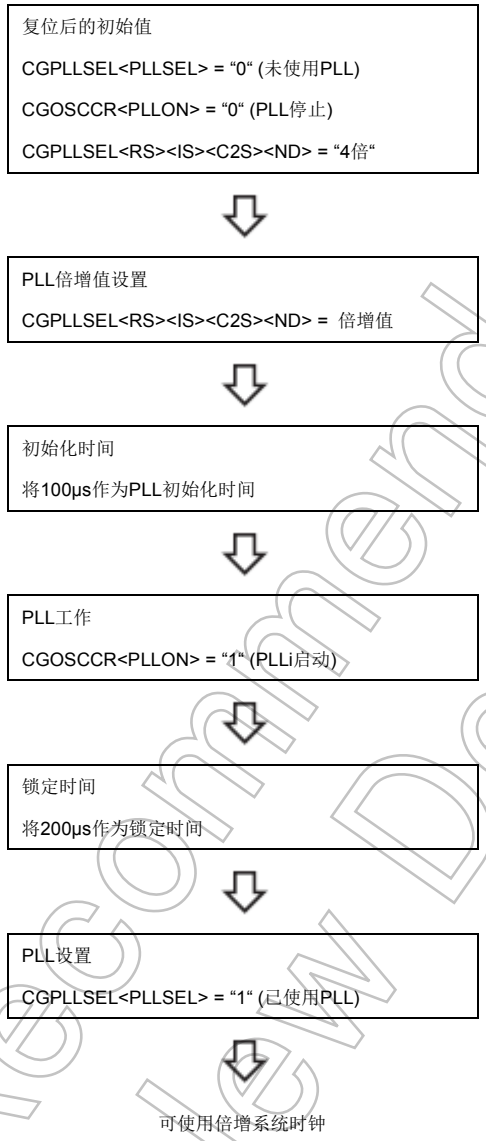
6.3.5.2 变换PLL倍增

当倍增变化时，首先将 “0“设置到CGPLLSEL<PLLSEL>，其次读取 CGPLLSEL<PLLSEL>，检查是否使用了倍增时钟的设置。

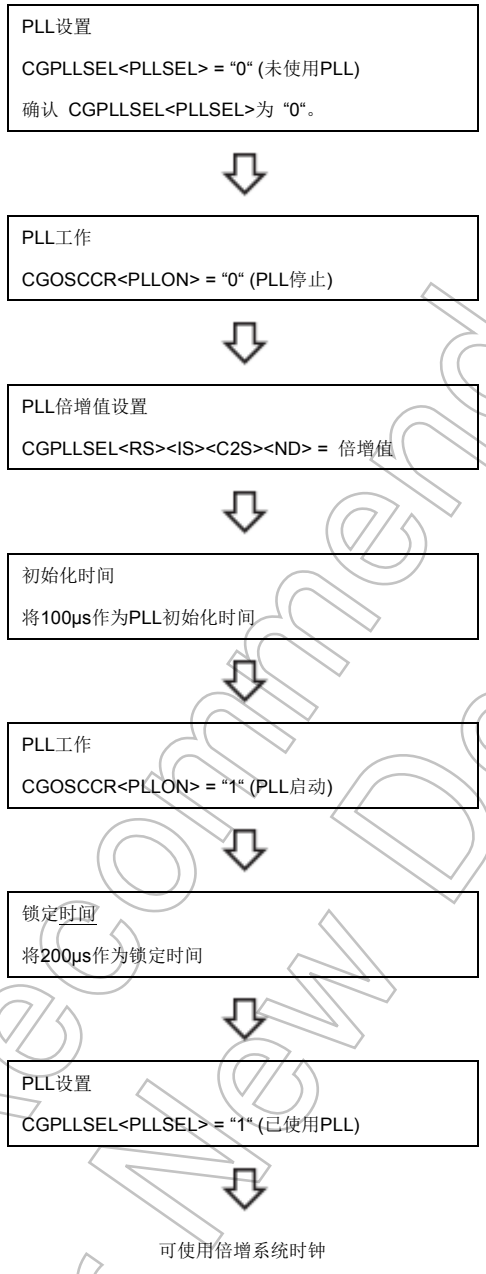
(CGPLLSEL<PLLSEL>= “0“)。再接着将 “0“设置到 <PLLON>。

将CGPLLSEL<RS[3:0]><IS[1:0]><C2S><ND[4:0]> 修改为倍增数值。PLL初始化100μs后，将 <PLLON> 设置为 “1“。200μs锁定时间后，将 CGPLLSEL<PLLSEL>设置为 “1“。

6.3.5.3 启动PLL顺序



6.3.5.4 倍增值更改顺序



6.3.6 系统时钟

TMPM364F10FG带有两种可选择系统时钟: 低-速时钟或高-速时钟。

6.3.6.1 高-速时钟

高-速时钟通过倍增来使用。

源时钟		频率	使用PLL
高-速振荡	振荡器	8 ~ 16MHz	未使用，4或 8倍增
	外部时钟	8 ~ 16MHz	

注:关于高速振荡PLL倍增和频率，见表6-1。

时钟除以CGSYSCR<GEAR[2:0]> 所得结果用于系统时钟。CGSYSCR<GEAR[2:0]> 可在操作时进行修改，但时钟的变换需要一定时间。

依据PLL倍增和时钟齿轮设定进行的操作频率设置举例入如下。

表 6-1 依据PLL倍增和时钟齿轮设置进行的操作频率举例 (单位 : MHz)

输入频率 X1, X2	PLL 倍增	最小操作频率	最大操作频率	复位后 (PLL = OFF, CG = 1/1	时钟齿轮 (CG) PLL = @ ON				时钟齿轮 (CG) PLL = @ OFF			
					1/1	1/2	1/4	1/8	1/1	1/2	1/4	1/8
8	4	1	32	8	32	16	8	4	8	4	2	1
9			36	9	36	18	9	4.5	9	4.5	2.25	1.13
10			40	10	40	20	10	5	10	5	2.5	1.25
12			48	12	48	24	12	6	12	6	3	1.5
13.5			54	13.5	54	27	13.5	6.75	13.5	6.75	3.37	1.69
16.0			64	16.0	64	32	16	8	16	8	4	2
8	8	1	64	8	64	32	16	8	8	4	2	1

注1:使用USB时，操作频率必须为48MHz。

注2:使用ADC时，通过设置 ADCLK<ADCLK>，ADCLK应等于或小于40MHz。

6.3.6.2 低-速时钟

自 XT1，XT2输入的频率如下。

表 6-2 低速频率范围

输入频率范围	最小操作频率	最大操作频率
30 ~ 34 (kHz)	30 kHz	34 kHz

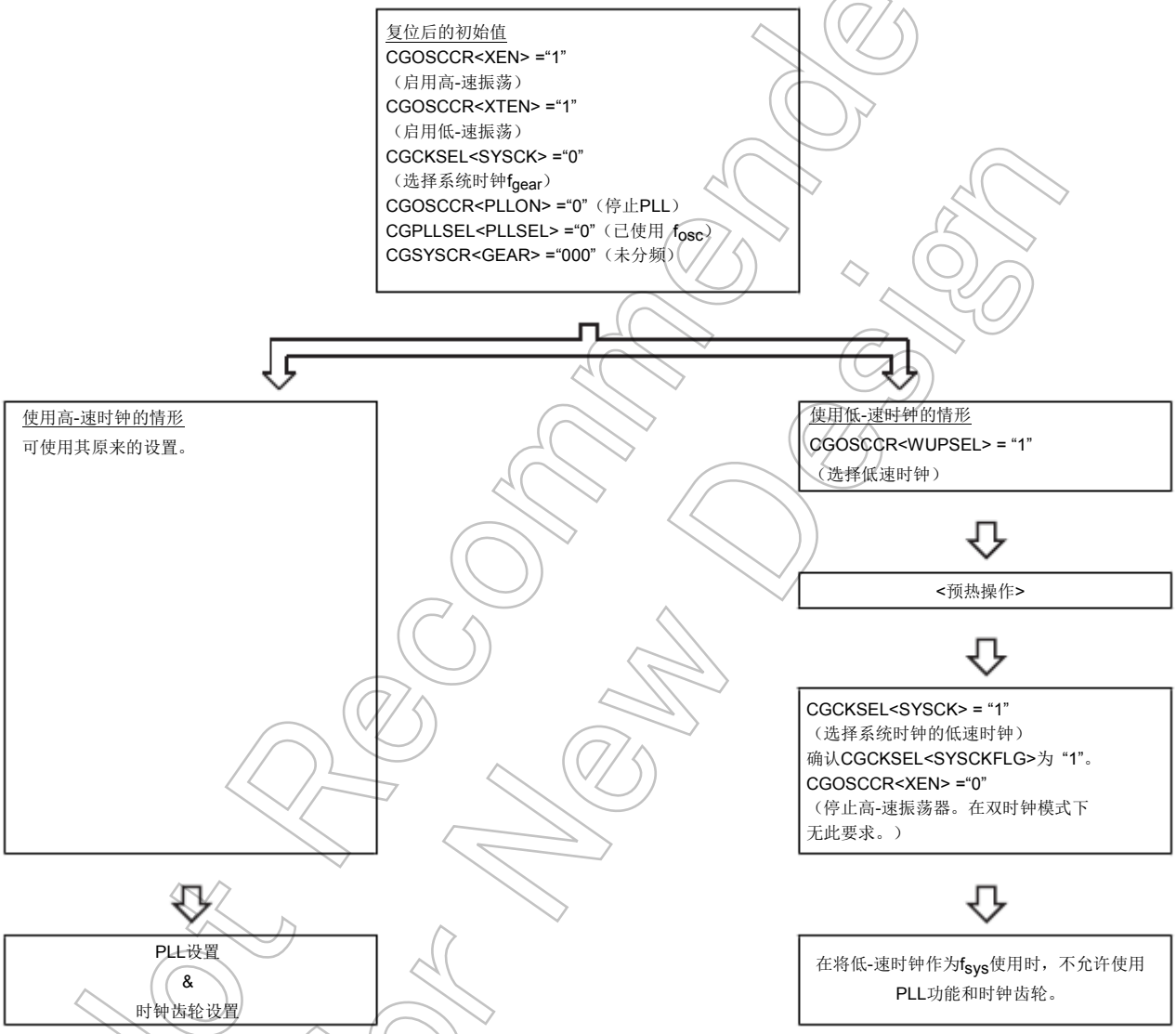
注:CEC采用f_S作为取样时钟。使用 CEC时， f_S必须在32.768kHz±4%范围内。

6.3.6.3 设置系统时钟

选择选择系统时钟通过CGOSCCR和CGCKSEL来使用。选定系统时钟后，将PLL设置为CGPLLSEL和CGOSCCR，将时钟齿轮设置为CGSYSCR。

如何设置系统时钟如下。

如何设置系统时钟



6.3.7 预分频器时钟控制

外围I/O端口具有一可用于对时钟进行分频的预分频器。对将时钟φT0输入到各预分频器时，可将CGSYSCR<FPSEL0> 中规定的 “f_{periph}“时钟按照CGSYSCR<PRCK[2:0]>中的设置进行分频。控制器复位后，f_{periph} /1选作φT0。

注: 使用时钟齿轮时，应当确保所设置的时间使预分频器自各外围功能的输出φTn慢于f_{sys} (φTn f_{sys})。切勿在定时器的计数器或其它外围功能操作时切换时钟齿轮。

6.3.8 系统时钟引脚输出功能

TMPM364F10FG能自某一引脚输出系统时钟。SCOUT引脚可输出低速时钟f_S，系统时钟f_{sys}和f_{sys}/2，以及外围I/O φT0预分频输入时钟。

注1: 不保证通过SCOUT输出的系统时钟和内部时钟之间的相差 (AC时序)

注2: 当f_{sys}自SCOUT引脚输出时，SCOUT引脚在紧接着变更时钟齿轮后输出某些异常波形。当出现异常波形对系统产生影响时，在变更时钟齿轮时应禁用SCOUT引脚输出。

通过设置CGSYSCR<SCOSEL[1:0]> 选择输出时钟。

设置为SCOUT引脚时，见 “输入/输出端口 “说明。

表 6-3 给出了当SCOUT 引脚设置到SCOUT输出时不同模式下的引脚状态

表 6-3 不同模式下SCOUT输出状态

模式 / SCOUT选择 CGSYSCR	NORMAL	SLOW	低功耗模式		
			IDLE2, 1	SLEEP	STOP/BACKUP
<SCOSEL[1:0]> = “00“	输出fs时钟				
<SCOSEL[1:0]> = “01“	输出 f_{sys}/2 时钟				
<SCOSEL[1:0]> = “10“	输出 f_{sys} 时钟				
<SCOSEL[1:0]> = “11“	输出 φT0 时钟	固定为 “0“ 或 “1“			

6.4 模式和模式转换

6.4.1 模式转换

对于系统时钟，NORMAL模式与SLOW模式分别使用高-速时钟和低-速时钟。

IDLE2/1，SLEEP，STOP以及BACKUP可作为低功耗模式使用，通过关闭处理器内核操作降低功耗。

当未使用低速时钟时，不能使用SLOW，SLEEP以及BACKUP模式。

TMPM364F10FG 具有 “BACKUP“模式。该模式能通过关闭除特殊功能外所有功能的主电源来降低功耗。

图 6-2 给出了模式转换的示意图。

异常退出时休眠的详细说明，见 “Cortex-M3 技术参考手册“。

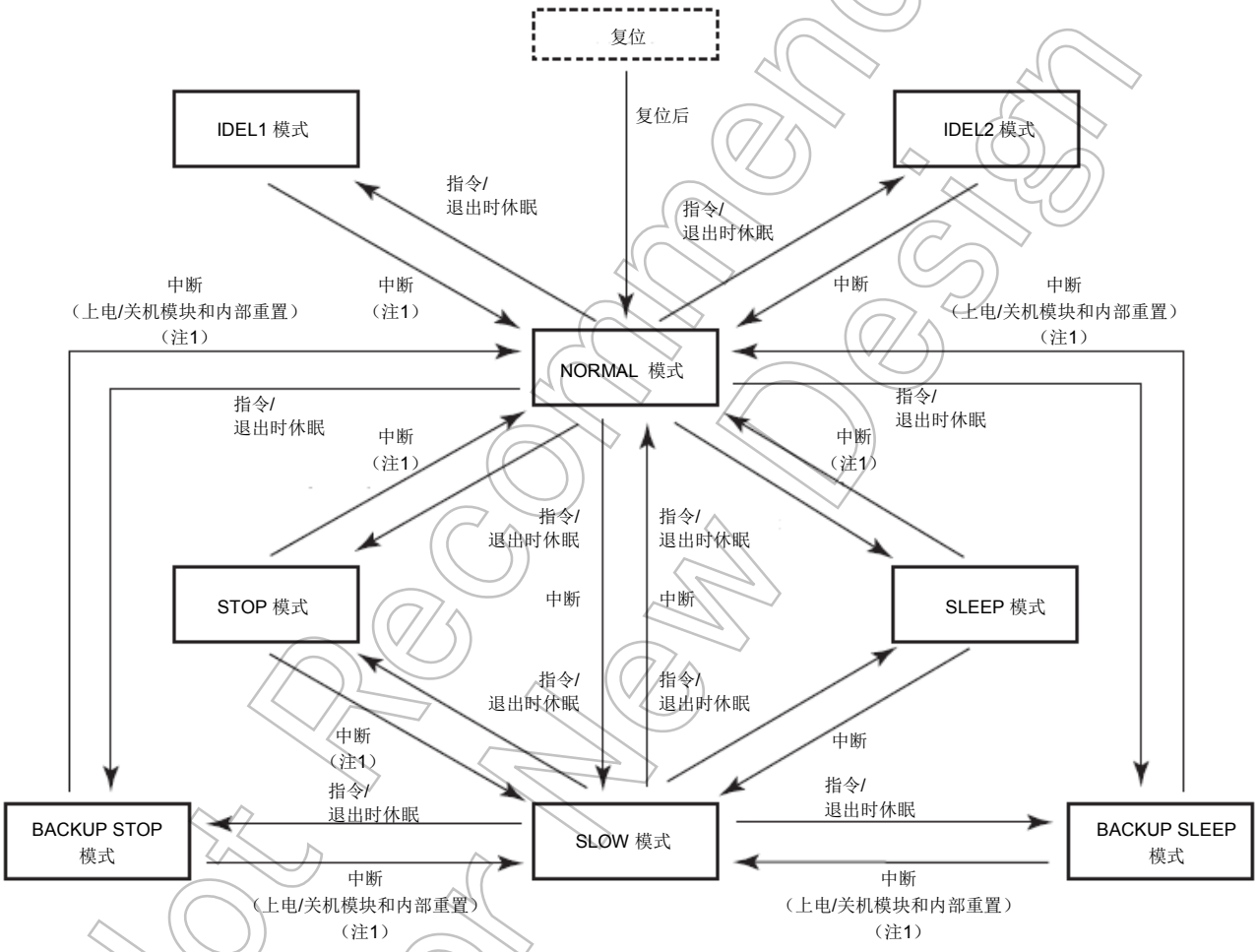


图 6-2 模式转换示意图

注1: 需要预热。在变更到STOP，SLEEP及BACKUP模式前，预热时间必须设置在NORMAL或SLOW模式下。关于预热时间的详细说明，见 “6.6.8 预热功能“说明。

注2: 未使用低速时钟时，不能使用SLOW，SLEEP及BACKUP SLEEP模式。

注3: 自SLOW模式转换到IDLE1模式和IDLE2模式不适用。

6.5 工作模式

NORMAL模式和SLOW模式适用。各模式的特点见后面章节说明。

6.5.1 NORMAL模式

此模式通过使用高-速时钟操作CPU内核和外围硬件。
复位后此模式切换到NORMAL模式。也可使用低-速时钟。

6.5.2 SLOW模式

此模式通过在高-速时钟停止时采用低-速时钟操作CPU内核和外围硬件。与NORMAL模式相比，SLOW模式降低了功耗。

此模式允许操作某些外围功能。

可操作的外围功能见表6-6。

注: 在SLOW模式下，不得采用Cortex-M3 NVIC寄存器的应用中断和复位控制寄存器<SYSRESETREQ>进行复位操作。

6.6 低功耗模式

TMPM364F10FG低功耗模式:IDLE1，IDLE2，SLEEP，STOP和BACKUP。切换到低功耗模式时，将该模式在系统控制寄存器 CGSTBYCR<STBY [2:0]> 中加以规定，并执行WFI (等待中断)指令。在这种情形下，进行复位 (BACKUP模式除外)或生成中断来释放该模式。通过提前中断请求设置进行释放。详细说明见 “异常“说明。

注 1: TMPM364F10FG没有任何释放低功耗模式的事件。不允许通过执行WFE (等待事件)指令转换到低功耗模式。

注 2: TMPM364F10FG不支持通过Cortex-M3内核中带SLEEPDEEP位元进行配置的低功耗模式。因此应严禁擅自对系统控制寄不允许设置系统控制寄存器的 <SLEEPDEEP> 位元。

注 3: 勿通过复位释放BACKUP模式。

各模式特点说明如下。

6.6.1 IDLE 模式 (LDLE 2，IDLE1)

在此模式下，CPU停止。各外围功能在IDLE模式下用于启用或禁用操作的控制寄存器均有1位。当启用IDLE模式时，在IDLE模式下操作的外围功能不能停止操作，且始终保持操作状态。

下述外围功能在IDLE模式下可以启动或禁用。设置的详细说明，见有关外围功能章节说明。

- 16-位定时器/事件计数器 (TMRB)
- 串行通道 (SIO/UART)
- 串行总线接口 (I2C/SIO)
- AD转换器 (ADC)
- 看门狗定时器 (WDT)

注:注意在IDLE模式下操作时，看门狗定时器功能计数器不能通过CPU清除。

6.6.1.1 IDLE2 模式

在IDLE2模式下，CPU停止。

与NORMAL模式相比，除了功耗降低外，其操作频率范围和外围性能与NORMAL模式相当。

6.6.1.2 IDLE1 模式

IDLE1模式在实现低功耗方面高于IDLE2。

IDLE1模式的使用条件见 “表6-6 各模式下的操作状态“，操作频率必须按照: f_{sys}=1MHz (f_{osc}=8MHz，PLL倍频器电路停止，时钟齿轮 1/8)设置。

6.6.2 SLEEP模式

SLEEP模式下，外部低速振荡器，RTC，CEC，RMC 以及键唤醒可操作。

通过释放SLEEP模式，设备返回SLEEP模式的前一模式后启动操作。

6.6.3 STOP模式

STOP模式下，所有内部电路，包括内部振荡器都被停止。

通过释放STOP模式，设备返回STOP模式的前一模式后启动操作。

STOP模式可通过设置CGSTBYCR<DRVE>选择引脚的状态。表 6-4 显示STOP模式下的引脚状态。

表 6-4 STOP下的引脚状态

	引脚名称	I/O	<DRVE> = 0	<DRVE> = 1
非端口	X1，XT1	仅输入	×	×
	X2，XT2	仅输出	“高“电平输出	“高“电平输出
	$\overline{\text{RESET}}$, $\overline{\text{NMI}}$, MODE	仅输入	o	o
端口	PL3，PL7，PM3，PM7，PN3，PE7，PG3，PG7，PN7，PO3，PO7，PD7，PH3，PH7 (当作为中断引脚PxFRn<PxmFn>=1 且输入已启用PxIE<PxmIE>=1) (注)	输入	o	o
		输出	×	取决于 PxCR[m]
	PJ4，PJ5，PJ6，PJ7 (当作为KWUP引脚PxFRn<PxmFn>=1时且输入已启用PxIE<PxmIE>=1) (注)	输入	o	o
		输出	×	取决于 PxCR[m]
	PF0，PF1，PF2，PF3，PF4 (当作为跟踪数据输出引脚 xFRn<PxmFn>=1 时) (注)	输入	×	取决于 PxIE[m]
		输出	取决于 (PxCR[m])	
	PA7 ~ PA0，PB7 ~ PB0，PC7 ~ PC0，PD7 ~ PD0，PE6 ~ PE0，PP6 ~ PP2，PP0 (当作为外部总线引脚PxFRn<PxmFn>=1 时。仅启用数据总线引脚和启用输入PxIE <PxmIE>=1) (注)	输入	o	o
		输出	取决于 (PxCR[m])	
	其它端口引脚	输入	×	取决于 PxIE[m]
		输出	×	取决于 PxCR[m]

O：已启用输入或输出。

×：已禁用输入或输出。

注:x：端口编号 / m：相应的位 / n: 功能寄存器编号

6.6.4 BACKUP模式 (BACKUP STOP，BACKUP SLEEP)

BACKUP模式通过切断内部电源稳压器将功耗降到最低。更多的详细说明，见BACKUP模式说明。

6.6.5 低功耗模式的设置

低功耗模式通过 CGSTBYCR <STBY[2:0]> 的设置加以规定。

表 6-5 中显示了这一模式在 <STBY[2:0]> 中的设置。

表 6-5 低功耗模式的设置

模式	CGSTBYCR <STBY[2:0]>
STOP	001
SLEEP	010
IDLE2	011
BACKUP STOP	101
BACKUP SLEEP	110
IDLE1	111

注:严禁采用除上述以外的设置。

6.6.6 各模式下的操作状态

表 6-6 显示了各模式下的操作状态。

表 6-6:各模式下的操作状态

模块	NORMAL	SLOW	IDLE2	IDLE1 (注 1)	SLEEP	STOP	BACKUP SLEEP	BACKUP STOP
处理器内核	o	o	−	−	−	−	×	×
DMAC	o	−	o	−	−	−	×	×
SMC	o	−	o	−	−	−	×	×
I/O端口	o	o	o (注 6)	o (注 6)	o (注 6)	o (注 2)	Δ (注 3)	Δ (注2 / 3)
ADC	o	# (注 5)	Δ	# (注 5)	− (注 5)	− (注 5)	×	×
SIO/UART	o	#	Δ	#	−	−	×	×
I2C/SIO	o	#	Δ	#	−	−	×	×
TMRB	o	o	Δ	#	−	−	×	×
WDT	o	#	Δ	#	−	−	×	×
SSP	o	#	−	−	−	−	×	×
CAN	o	#	−	−	−	−	×	×
USB-HOST	o	# (注 8)	− (注 8)	− (注 8)	− (注 8)	− (注 8)	× (注 8)	× (注 8)
KWUP	o	o	o (注 4)	o (注 4)	o	o (注 4)	Δ	Δ (注 4)
CEC	o	o	Δ	Δ	o	−	Δ	−
RMC	o	o	Δ	Δ	o	−	Δ	−
RTC	o	o	o	o	o	−	Δ	−
CG	o	o	o	o	o	o	o	o
PLL	o	*	Δ	#	#	#	#, ×	#, ×
高-速振荡器 (fc)	o	Δ	o	o	−	−	−	−
低-速振荡器 (fs)	o	o	・	・	o	−	o	−
主RAM	o	o	o	o	o	o	×	×
BACKUP RAM (注2)	o	o	o	o	o	o	o	o

o：操作

−：设置模式后时钟自动停止。(注7)

Δ：通过软件选择操作 / 停止。

#：设置模式前，必须通过软件停止这些模块。

×：完成设置模式转换后，模块自动掉电。

*：完成设置模式转换后，必须通过软件关闭这些模块。

・：进入设置模式前，可通过软件选择操作 / 停止。

注1: 在IDLE1和SLOW模式下，表 6-6 中显示的特殊外围功能必须停止。在IDLE1模式下，TMPM364F10FG的工作频率必须达到最大频率，即f_{sys} = 1MHz (f_{osc}=8MHz，PLL停止，时钟齿轮，1/8)。

注2: 此项取决于CGSTBYCR<DRVE>。

注3: 此项取决于CGSTBYCR<PTKEEP>。

注4: 当低速振荡器被迫停止或自动停止时，仅静态上拉有效。

注5: 完成设置模式转换前，将ADMO1D<VREFON>为 ”0”。

注6: 转换前端口状态保持低功耗。

注7: 完成设置模式转换后，该模块自带时钟自动停止。因此，确认各模式停止后转换设置模式。

注8: 完成设置模式转换前，须做SUSPEND 。

6.6.7 释放低功耗模式

除BACKUP模式外，低功耗模式可通过中断请求，非可屏蔽中断 (NMI)或复位进行释放。

可采用的释放源由所选低功耗模式决定。

详细说明见表 6-7。

表 6-7 各模式的释放源

低功耗模式			IDLE2	IDLE1 (注1)	SLEEP	STOP	BACKUP SLEEP (注 2)	BACKUP STOP (注 2)
解除来源	中断	INT 0 ~ 4, 8 (注 3)						
		INT 5 ~ 7, 9 ~ D (注 3)	o	o	o	o	o	o
		INTRTC	o	o	o	o	×	×
		INTTB 0 ~ F	o	o	o	×	o	×
		INTCAP 10 ~ 20, 50 ~ 70, 90 ~ B0, D0	o	×	×	×	×	×
		~ F0	o	×	×	×	×	×
		INTCAP 11 ~ 21, 51 ~ 71, 91 ~ B1, D1	o	×	×	×	×	×
		~ F1	o	×	×	×	×	×
		INTRX0 ~ B, INTTX 0 ~ B	o	×	×	×	×	×
		INTSBI0 ~ 4	o	o	o	×	o (注5)	×
		INTCECRX, INTCECTX	o	o	o	×	o	×
		INTRMCRX0, 1	o	×	×	×	×	×
		INTAD/INTADHP/INTADM0, 1	o	o	o	o	o	o
		INTKWUP						
	SysTick 中断		o	o	×	×	×	×
	NMI (INTWDT)		o	×	×	×	×	×
	NMI ($\overline{\text{NMI}}$ 引脚)		o	×	o	o	×	×
	RESET ($\overline{\text{RESET}}$ 引脚)		o	o	o	o	×	×

o：释放此模式后启动中断。 (复位初始化了LSI)

×：不可用

注1: 有关预热时间，见 “6.6.8 预热“说明。

注2: 释放BACKUP模式后，系除BACKUP模块外对电路进行初始化。

注3: 利用电平模式中断释放低功耗模式，保持该电平直到中断启动。如在此之前更改电平，将导致中断处理无法正常启动。

注4: 如需切换到低功耗模式，将CPU设置为禁止释放除释放源以外的所有中断。否则，有可能在未规定的条件下释放当前模式，从而唤醒系统。

注5: INTCECRX (CEC接收中断)为BACKUP SLEEP模式下唤醒的源触发器。而在BACKUP SLEEP模式下INTCECTX (CEC传输中断)却不是源触发器。

注6: 从NORMAL模式切换到IDLE1模式时，要求预热时间超过100μs。否则，当自IDLE1模式返回时，MCU内部系统时间不会回复。

通过中断请求进行释放

通过某一中断是否低功耗模式时，CPU必须预先设置以检测是否中断。除在CPU中设置外，须设置时钟发生器以检测所使用的中断是否是否了SLEEP模式和STOP模式。

通过非-可屏蔽中断 (NMI)释放

NMI有两种形式:WDT中断 (INTWDT)和NMI引脚。INTWDT仅限于在IDLE2模式中使用。除BACKUP和IDLE1模式外，NMI引脚可用于释放所有低功耗模式。

通过复位进行释放

除BACKUP模式外，任何低功耗模式都可通过 $\overline{\text{RESET}}$ 引脚复位进行释放。之后，模式切换到NORMAL模式，所有寄存器按其正常复位方式进行初始化。

注意通过复位自STOP的释放不诱发自动预热。应保持信号有效直到振荡器操作稳定。

通过 SysTick 中断进行释放

SysTick中断仅限于在IDLE2模式中使用。

注: 勿通过复位释放BACKUP模式。

有关 “中断“的详细情况，见 “异常“说明。

6.6.8 预热

模式的转换可能要求进行预热，以便于内部振荡器保持稳定振荡。

在将模式从STOP切换到NORMAL / SLOW，或自IDLE1 / SLEEP到NORMAL模式时，预热计数器自动激活。接着，经过预热配置时间后，系统时钟输出启动。

在执行进入STOP / IDLE1 / SLEEP模式指令之前，必须在CGOSCCR<WUPSEL>中选择用于预热的振荡器，且在CGOSCCR<WUPT[11:0]><WUPTL[1:0]> 中设置预热时间。

自NORMAL转换到SLOW / SLEEP时，要求进行预热，以便于低-速时钟禁用的情况下使内部振荡器稳定。启用低-速时钟，然后通过软件激活预热。

当禁用高速时钟时自SLOW转换到NORMAL，启用高速时钟再激活预热。

表 6-8 显示了各模式转换是否需要进行预热。

表 6-8 模式转换预热设置

模式转换	预热设置
NORMAL → IDLE2, 1	未要求
NORMAL → SLEEP	未要求 (注 1)
NORMAL → SLOW	未要求 (注 1)
NORMAL → STOP	未要求
NORMAL → BACKUP SLEEP	未要求 (注 1)
NORMAL → BACKUP STOP	未要求
SLOW → NORMAL	未要求 (注 2)
SLOW → SLEEP	未要求
SLOW → STOP	未要求
SLOW → BACKUP SLEEP	未要求
SLOW→ BACKUP STOP	未要求
IDLE2 → NORMAL	未要求
IDLE1 → NORMAL	自动预热 (注3) 高-速振荡器 : 大于或等于100μs
SLEEP → NORMAL	自动预 高-速振荡器 : 预热时间的设定数值
SLEEP → SLOW	未要求
STOP → NORMAL	自动预热 高-速振荡器 : 预热时间的设定数值
STOP→ SLOW	自动预热 低-速振荡器 : 预热时间的设定数值
BACKUP SLEEP → NORMAL	自动预热 (注3) 高-速振荡器 : 大于或等于500μs
BACKUP STOP → NORMAL	自动预热 (注3) 高-速振荡器 : 大于或等于500μs
BACKUP SLEEP→ SLOW	自动预热 (注 3) 低-速振荡器 : 大于或等于2.5ms
BACKUP STOP → SLOW	自动预热 (注 3) 低-速振荡器 : 大于或等于2.5ms

注 1: 当禁用低-速时钟时，启用低速时钟再通过软件激活预热。

注 2: 当禁用高-速时钟处时，启用高速时钟再通过软件激活预热。

注 3: 预热时间的设定数值不得小于规定值。

注 4: 通过复位返回到标准模式并不会诱发自动预热。 应保持信号有效直到振荡器操作稳定为止。

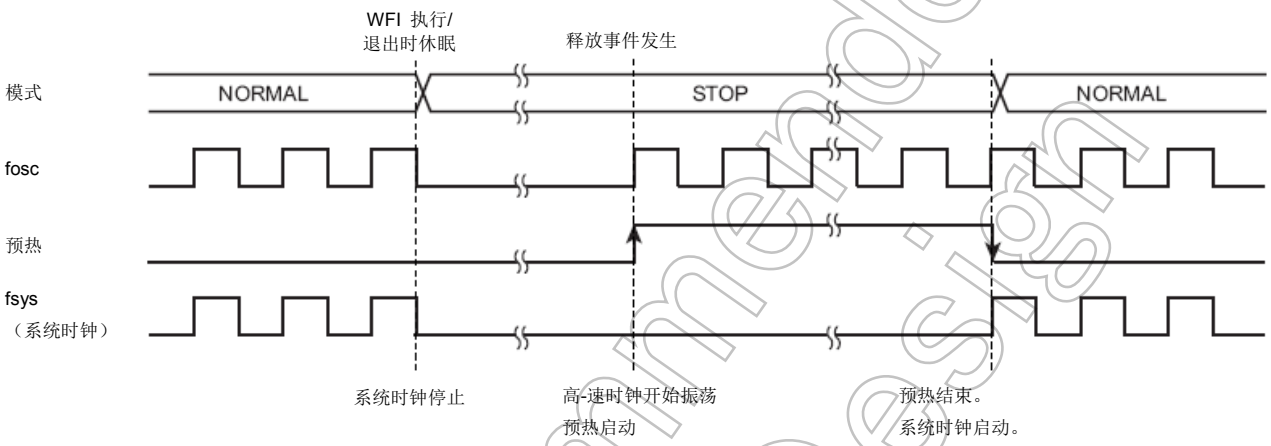
6.6.9 模式转换下的时钟操作

模式转换时的时钟操作见 6.6.9.1 ~ 6.6.9.4说明。

6.6.9.1 工作模式转换: NORMAL → STOP → NORMAL

自STOP模式返回NORMAL模式时，预热自动激活进入STOP模式前必须设置预热。

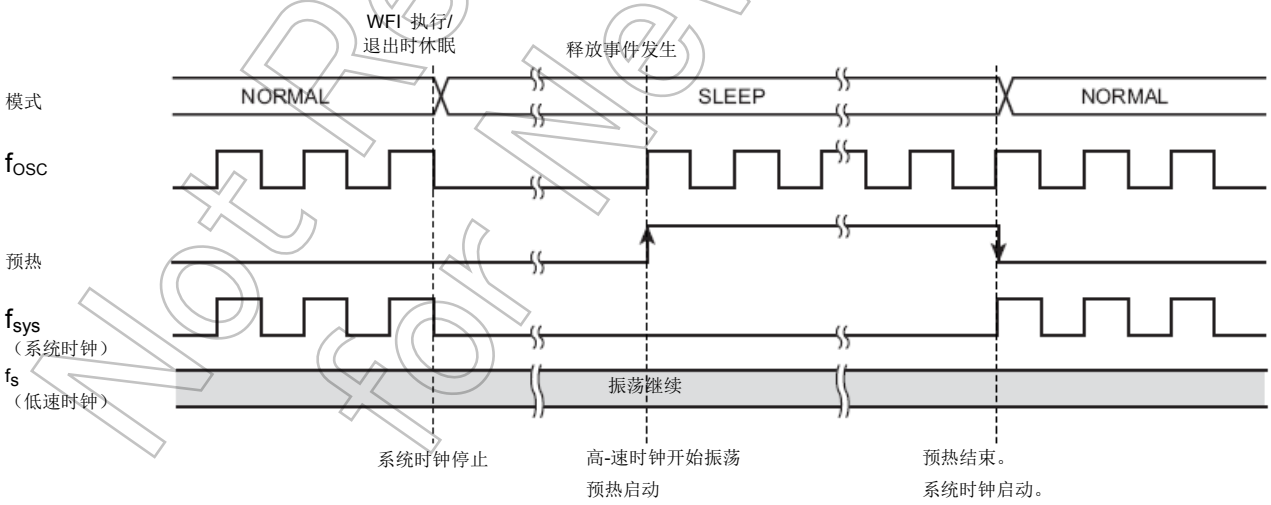
通过复位返回NORMAL模式不会诱发自动预热。应保持复位信号有效直到振荡器操作稳定。



6.6.9.2 工作模式转换 : NORMAL → SLEEP → NORMAL

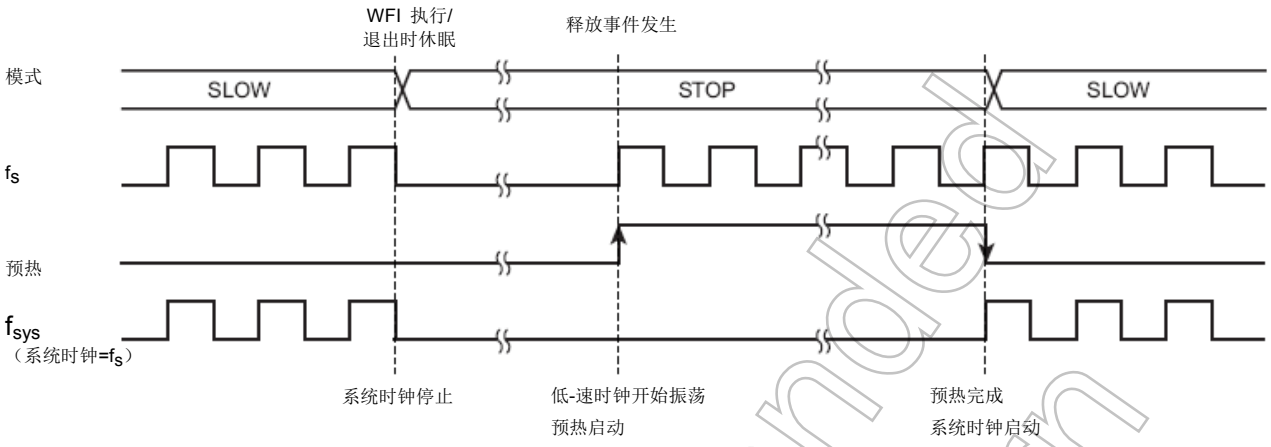
自SLEEP模式返回NORMAL模式时，预热自动激活。进入SLEEP模式前，必须设置预热时间。

通过复位返回NORMAL模式不会诱发自动预热。应保持复位信号有效直到操作稳定。



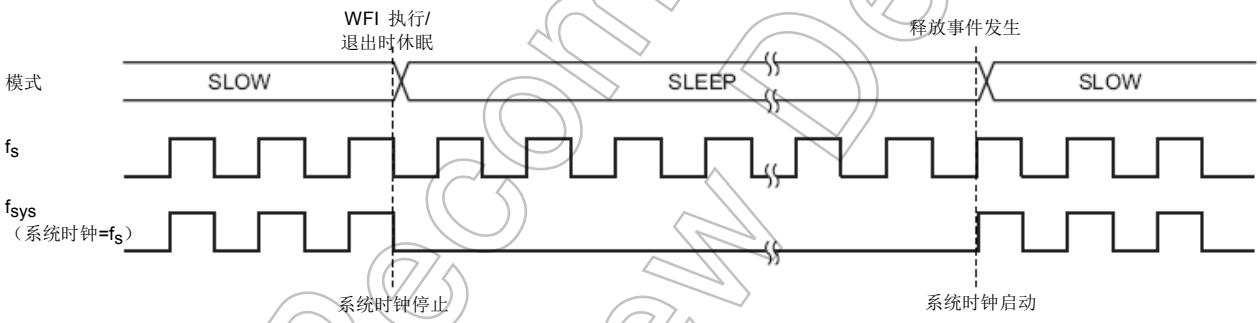
6.6.9.3 工作模式转换:SLOW → STOP → SLOW

预热自动激活。进入STOP模式前必须设置预热。



6.6.9.4 工作模式转换:SLOW → SLEEP → SLOW

在SLEEP模式下，低速时钟持续振荡。无需做预热设置。



7. 异常

本章描述异常的特征，类型及处理方式。

异常与CPU内核有密切的关系。需要时请参阅“Cortex-M3技术参考手册“。

7.1 综述

异常与CPU内核有密切的关系。需要时请参阅 “Cortex-M3技术参考手册“。

异常情况有两种类型:一种是在发生一些错误情况或执行产生异常的指令时产生的异常；另一种是由硬件产生的异常，例如从外部引脚或周边设备发出中断请求信号。

所有异常根据各自的优先级，由CPU中的嵌套向量中断控制器 (NVIC)进行处理。当出现异常时，CPU将当前异常状态存入堆栈，并跳转到相应的中断服务程序 (ISR)。中断服务程序工作完成后，存入堆栈的信息将会自动恢复。

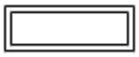
7.1.1 异常类型

Cortex-M3中存在以下异常类型。

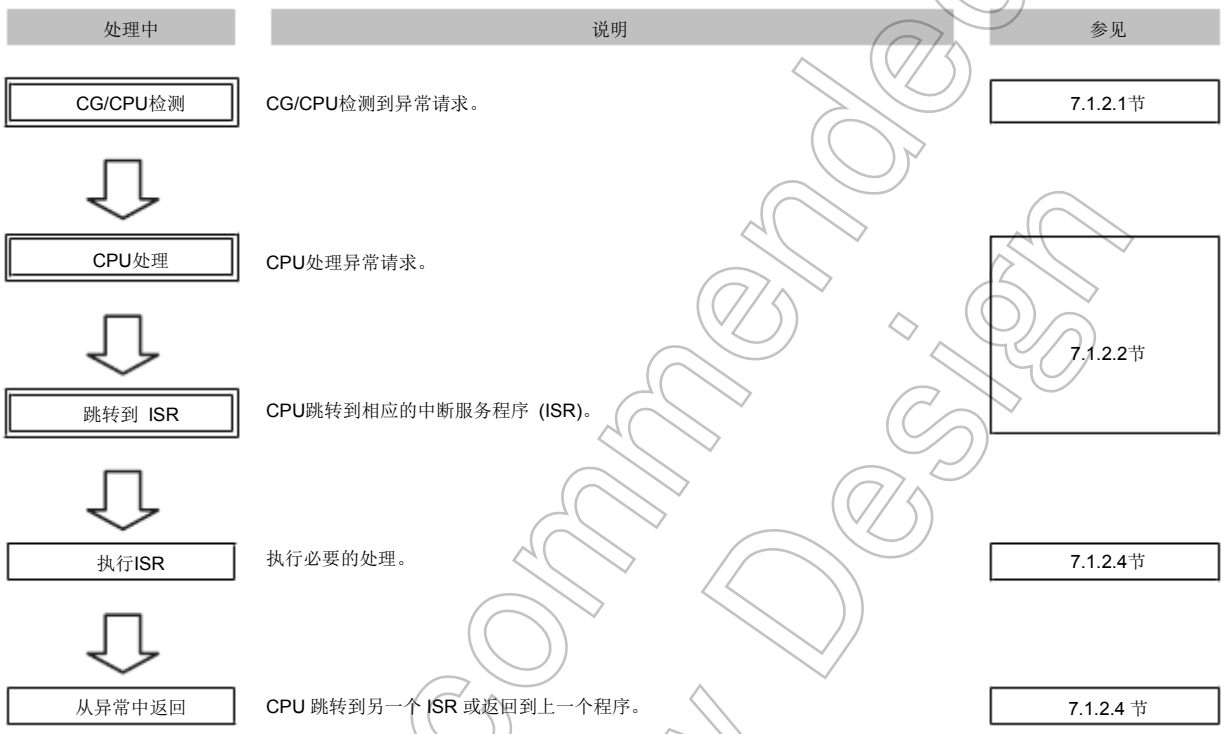
对于每种异常的详细描述，请参阅“Cortex-M3技术参考手册“。

- 复位
- 非-屏蔽中断 (NMI)
- 硬故障
- 内存管理
- 总线故障
- 使用故障
- SVCall (管理程序调用)
- 调试监测程序
- PendSV
- SysTick
- 外部中断

7.1.2 处理流程图

下面说明如何处理一个异常/中断。在以下的说明中，  表示硬件处理。  表示软件处理。

每个步骤将在本章后续部分介绍。



7.1.2.1 异常请求与检测

(1) 异常发生

异常来源包括CPU执行的指令，内存访问，和外部中断引脚或周边设备发出的中断请求。

CPU执行一条导致异常的指令，或执行指令期间出现错误情况时，会发生异常。

永不执行(XN)区域的指令取出，或故障区域的访问冲突，也会导致异常发生。

中断请求由外部中断引脚或周边设备发出，对于用于释放待机模式的中断，必需在时钟发生器中做相应设置。

有关详情, 请参见 “7.5 中断“说明。

(2) 异常检测

如果同时发生多个异常，CPU首先对优先级最高的进行处理。

表 7-1 显示了异常的优先级。 “可配置“表示可为该异常分配优先级。可以启用或禁用内存管理，总线故障和使用故障等异常。如果发生禁用异常，将其作为硬故障处理.

表7-1 异常类型和优先级

编号.	异常类型	优先级	说明
1	复位	-3 (最高)	复位引脚，WDT 或 SYSRETREQ
2	非-屏蔽中断	-2	$\overline{\text{NMI}}$ 引脚或WDT
3	硬故障	-1	因正处理一个更高优先级的故障或因被禁用而无法激活的故障
4	内存管理	可配置	内存保护单元 (MPU) 异常 (注1) 永不执行 (XN) 区域的指令取出
5	总线故障	可配置	对内存映射硬故障区域的访问冲突
6	使用故障	可配置	未定义的指令执行或与指令执行相关的其他故障
7 ~ 10	保留	-	
11	SVCall	可配置	使用SVC指令的系统服务调用
12	调试监测程序	可配置	CPU未故障时的调试监测程序
13	保留	-	
14	PendSV	可配置	可挂起的系统服务请求
15	SysTick	可配置	来自SysTick通知
16 ~	外部中断	可配置	外部中断引脚或周边设备 (注2)

注 1: **该产品不包含MPU。**

注 2: **各产品的外部中断有不同的来源和数目。有关详情，请查阅“7.5.1.5 中断源列表“。**

(3) 优先级设置

- 优先等级

在系统处理程序优先级寄存器中，外部中断优先级设置为中断优先级寄存器，而其他异常设置为<PRI_n>位。

可对<PRI_n>的配置进行修改，根据产品的不同，设置优先级所需要的位数从3位 ～8位不等。因此，根据产品不同，可设定的优先级值的范围是不同。

在8-位配置的情况下，优先级范围可设置为0～255。最高优先级为 "0"。如果存在具有相同优先级的多个元素，数字越小，优先级越高。

注: **<PRI_n>位被定义为此产品中的3-位配置。**

- 优先级分组

优先级可被分为多组。通过设定应用中断和复位控制寄存器的<PRIGROUP>，<PRI_n>可分为占先式优先级和次优先级。

将一个优先级与占先式优先级相比较。如果该优先级与占先式优先级相同，则将其与次优先级相比较。如果该优先级与次优先级相同，异常值越小，优先级越高。

表 7-2 显示了优先级组设置。表中占先式优先级与次优先级为<PRI_n>被定义为8-位配置情况下的数量。

表 7-2 优先级分组设置

<PRIGROUP[2:0]> 设置	<PRI_n[7:0]>		占先式 优先级数量	次优先级 数量
	占先式域	次优先级域		
000	[7:1]	[0]	128	2
001	[7:2]	[1:0]	64	4
010	[7:3]	[2:0]	32	8
011	[7:4]	[3:0]	16	16
100	[7:5]	[4:0]	8	32
101	[7:6]	[5:0]	4	64
110	[7]	[6:0]	2	128
111	无	[7:0]	1	256

注: **如果<PRI_n> 的配置低于8位，则低位为"0"。例如，在3-位配置的情况下，优先级设定为 <PRI_n[7:5]> 和 <PRI_n[4:0] >为"00000"。**

7.1.2.2 异常处理和分支转移到中断服务程序 (预清空)

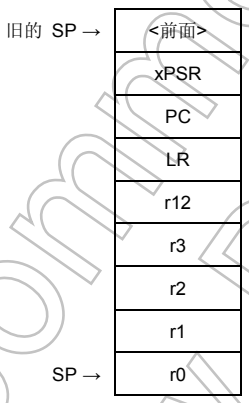
当异常发生时，CPU暂停当前正在执行的进程，并跳转到中断服务程序。这就是所谓的"占先"。

(1) 堆叠

当CPU检测到异常时，它按以下顺序将下列八个寄存器的内容推送到堆栈中:

- 程序计数器 (PC)
- 程序状态寄存器 (xPSR)
- r0 ~ r3
- r12
- 链路寄存器 (LR)

完成压栈后，SP会递减八个字。下面显示了寄存器内容已写入后堆栈的状态。



(2) 读取一个ISR

该CPU启动指令以使用储存到寄存器的数据获取中断处理。

为每个异常准备一个包含ISR顶部地址的向量表。复位后，该向量表指向代码区的地址0×0000_0000。通过设置向量表偏移寄存器，可以将向量表放置在代码或SRAM空间的任意地址。

向量表也应包含主堆叠的初始值。

(3) 迟到

在对上一个异常执行ISR之前，如果CPU检测到更高优先级的异常，CPU首先处理更高优先级的异常。这就是所谓的"迟到"。

迟到导致CPU获取一个新的向量地址，以跳转到相应的ISR，但CPU不重新将寄存器内容推送到堆栈。

(4) 向量表

向量表按如下所示进行配置。

必须始终对前四个字进行设置 (堆栈顶部地址，复位ISR地址， NMI ISR地址和硬故障ISR地址)。必要时为其他异常设置ISR地址。

偏移量	例外	目录	设置
0x00	复位	主堆栈的初始值	要求
0x04	复位	ISR 地址	要求
0x08	非-屏蔽中断	ISR 地址	要求
0x0C	硬故障	ISR 地址	要求
0x10	内存管理	ISR 地址	可选
0x14	总线故障	ISR 地址	可选
0x18	使用故障	ISR 地址	可选
0x1C ~ 0x28	保留		
0x2C	SVCall	ISR 地址	可选
0x30	调试监测器	ISR 地址	可选
0x34	保留		
0x38	PendSV	ISR 地址	可选
0x3C	SysTick	ISR 地址	可选
0x40	外部中断	ISR 地址	可选

7.1.2.3 执行 ISR

ISR为相应异常进行必要的处理。ISRs必须由用户编写。

ISR可能需要包括用于清除中断请求的代码，以便返回到正常的程序执行时，相同中断不会再次发生。

有关中断处理的详细信息，请参阅”7.5 中断”说明。

在当前异常执行ISR期间，若发生一个更高优先级的异常，CPU将放弃当前正在执行的ISR，并服务于新检测到的异常。

7.1.2.4 异常出口

(1) 从ISR返回后执行

当从ISR返回时，CPU将执行下列操作之一:

・末尾连锁

如果存在挂起异常且没有堆叠异常，或挂起异常比所有堆叠异常优先级高，CPU返回到挂起异常的ISR。

在这种情况下，当退出一个ISR并进入另一个时，CPU跳过对8个寄存器的读操作和对8个寄存器的写操作。这就是所谓的"末尾连锁"。

・返回到最后一个堆叠的ISR

如果没有挂起异常，或者最高优先级的堆叠异常比最高优先级的挂起异常优先级高，CPU返回到最后一个堆叠的ISR。

・返回到上一个程序

如果没有挂起或堆叠异常，CPU返回到上一个程序。

(2) 异常退出序列

当从ISR返回时，CPU执行以下操作:

・读取8个寄存器

从堆栈读取8个寄存器 (PC, xPSR，r0 ~ r3，r12和LR) 并调整SP。

・加载当前激活的中断号

从堆叠xPSR加载当前激活的中断号。 CPU据此跟踪应返回哪个中断。

・选择SP

如果返回一个异常 (处理程序模式)，SP为SP_main。如果返回Thread模式，SP可能为SP_main或SP_process。

7.2 复位异常

复位异常从以下三个来源产生。

使用时钟发生器的复位标志 (CGRSTFLG) 寄存器来识别复位源。

・外部复位引脚

当外部复位引脚由"低"变为"高"时，发生复位异常。

・WDT复位异常

看门狗定时器 (WDT)有复位生成功能。有关详情，请参阅WDT一章.

・SYSRESETREQ复位异常

复位可以通过设置NVIC应用中断和复位控制寄存器中的SYSRESETREQ位来生成。

注: **不要在慢模式中采用<SYSRESETREQ>复位。**

7.3 非-屏蔽中断 (NMI)

非屏蔽中断由以下两个来源产生。

使用时钟发生器的NMI标志寄存器 (CGNMIFLG)来识别非屏蔽中断源。

・外部 $\overline{\text{NMI}}$ 引脚

当外部 $\overline{\text{NMI}}$ 引脚由 "高"变为 "低"时，发生非屏蔽中断。

・WDT非屏蔽中断

看门狗定时器 (WDT)有非屏蔽中断生成功能。有关详情，请参阅WDT一章。

7.4 SysTick

SysTick使用CPU的系统计时器提供中断功能。

在设置SysTick重新加载值寄存器值并启用SysTick控制和状态寄存器的SysTick功能时，计数器加载重新加载值寄存器中设置的值并开始倒计时。当计数器达到 "0"时，发生SysTick异常。可能正挂起异常并可使用标志以了解计时器何时到达 "0"。

SysTick校准值寄存器保持用于使用系统计时器计时10 ms的重新加载值。因计数时钟频率随产品而变化，SysTick校验值寄存器中设置的值也随产品而变化。

注:在本产品中，32 f_{osc}用作外部参考时钟。

7.5 中断

本章描述了中断的路径，来源以及所需的设定。

CPU通过中断源发出的中断信号来报告中断请求。

设置中断优先权，使中断请求具有最高的优先权。

CPU通过时钟发生器收到清除待机模式的中断请求。因此，时钟发生器的设置一定要恰当。

7.5.1 中断源

7.5.1.1 中断路径

图7-1给出了一个中断请求路径。

把不用来解除待机模式的外围函数的中断直接输入到CPU内 (路径1)。

把用于解除待机模式 (路径2)的外围函数中断和外部中断引脚的中断 (路径3)输入时钟发生器，并通过解除待机模式的逻辑电路输入CPU中 (路径4和5)。

如果外部中断引脚中断不用于解除待机模式，则直接被输入CPU中，而不是通过解除待机模式的逻辑(路径6)。

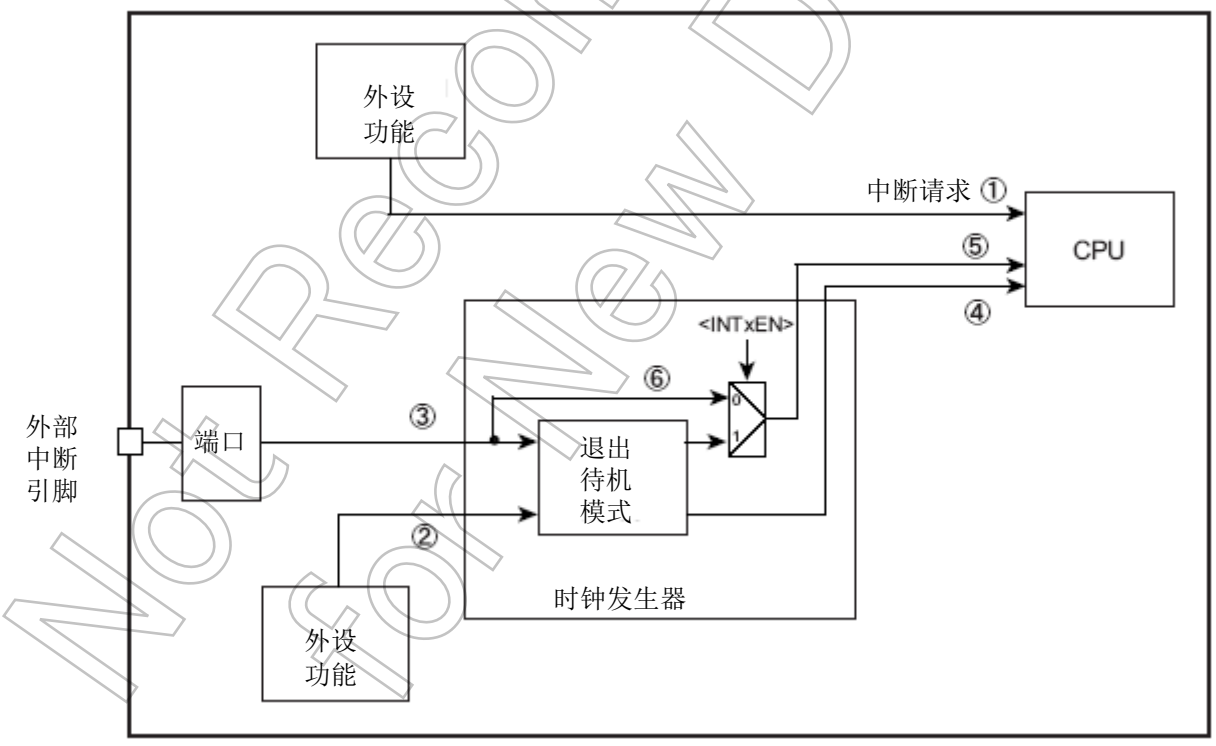


图7-1 中断路径

7.5.1.2 生成

中断请求从外部引脚或外围函数这类中断源发出，或者通过设置NVIC中断挂起寄存器发出。

- 来自外部引脚

  设置端口控制寄存器使外部引脚发挥中断功能引脚的作用。

- 来自外设功能

  设置外设功能以致其可以输出中断请求。

  详情请见各外设功能章节。

- 通过设置中断挂起寄存器 (强制挂起)

  设置中断挂起寄存器的相关位可发出中断请求。

7.5.1.3 传输

除非用于退出待机模式，否则外部引脚和外设功能产生的中断信号直接发送到CPU。

中断源处用于清除待机模式的中断请求通过时钟发生器传入CPU中。对于这些中断源，时钟发生器必须提前做恰当设置。不用于退出待机模式的外部中断源在不设置时钟发生器时也可使用。

7.5.1.4 使用外部中断引脚时的注意事项

如果使用外部中断，请注意以下情况，避免产生意外中断。

如果输入禁用 (PxIE<PxmIE>= “0“)，从外部中断引脚输入的即为“高“。同样，如果外部中断不作为解除待机模式的触发器 (表7-1 路径6)，外部引脚输入信号直接被送到CPU。因为CPU识别“高“作为一个中断，所以，如果CPU启用相应中断，并且输入禁用时，就会产生中断。

外部中断不作为待机触发器使用时，设置中断引脚输入为 “低“并启用。然后，启用CPU上的中断。

7.5.1.5 中断源表

图7-3 显示出了中断源的列表。

图7-3 中断源列表

编号	中断源		有效电平 (解除待机和中断)	CG中断模式 控制寄存器
0	INT0	中断引脚 0	可选择	CGIMCGA
1	INT1	中断引脚 1		
2	INT2	中断引脚 2		
3	INT3	中断引脚 3		
4	INT4	中断引脚 4		CGIMCGB
5	INT5	中断引脚 5		
6	INT6	中断引脚 6		
7	INT7	中断引脚 7		
8	INT8	中断引脚 8		CGIMCGC
9	INT9	中断引脚 9		
10	INTA	中断引脚 A		
11	INTB	中断引脚 B		
12	INTC	中断引脚 C		CGIMCGD
13	INTD	中断引脚 D		
14	保留	-	-	-
15	保留	-		
16	INTRX0	串行通道接收信号 0 中断		
17	INTTX0	串行通道传输信号 0 中断		
18	INTRX1	串行通道接收信号 1 中断		
19	INTTX1	串行通道传输信号 1 中断		
20	INTRX2	串行通道接收信号 2 中断		
21	INTTX2	串行通道传输信号 2 中断		
22	INTRX3	串行通道接收信号 3 中断		
23	INTTX3	串行通道传输信号 3 中断		
24	INTRX4	串行通道接收信号 4 中断		
25	INTTX4	串行通道传输信号 4 中断		
26	INTSBI0	串行总线接口 0 中断		
27	INTSBI1	串行总线接口 1 中断		
28	INTCECRX	CEC接收中断	上升沿	CGIMCGE
29	INTCECTX	CEC传输中断		
30	INTRMCRX0	遥控信号接收 0 中断		
31	INTRMCRX1	遥控信号接收 1 中断		
32	INTRTC	RTC中断	下降沿	CGIMCGF
33	INTKWUP	接通唤醒中断	高位	
34	INTSBI2	串行总线接口 2 中断		
35	INTSBI3	串行总线接口 3 中断		
36	INTSBI4	串行总线接口 4 中断		
37	INTADHP	具有最高优先级的AD转换完成中断。		
38	INTADM0	AD转换监控功能0中断		
39	INTADM1	AD转换监控功能1中断		
40	INTTB0	16-位定时器 / 事件计数器 0 匹配检测中断		
41	INTTB1	16-位定时器 / 事件计数器 1 匹配检测中断		

图7-3 中断源列表

编号	中断源		有效电平 (解除待机和中断)	CG中断模式 控制寄存器
42	INTTB2	16-位定时器 / 事件计数器 2 匹配检测中断		
43	INTTB3	16-位定时器 / 事件计数器 3 匹配检测中断		
44	INTTB4	16-位定时器 / 事件计数器 4 匹配检测中断		
45	INTTB5	16-位定时器 / 事件计数器 5 匹配检测中断		
46	INTTB6	16-位定时器 / 事件计数器 6 匹配检测中断		
47	INTTB7	16-位定时器 / 事件计数器 7 匹配检测中断		
48	INTTB8	16-位定时器 / 事件计数器 8 匹配检测中断		
49	INTTB9	16-位定时器 / 事件计数器 9 匹配检测中断		
50	INTTBA	16-位定时器 / 事件计数器 A 匹配检测中断		
51	INTTBB	16-位定时器 / 事件计数器 B 匹配检测中断		
52	INTTBC	16-位定时器 / 事件计数器 C 匹配检测中断		
53	INTTBD	16-位定时器 / 事件计数器 D 匹配检测中断		
54	INTTBE	16-位定时器 / 事件计数器 E 匹配检测中断		
55	INTTBF	16-位定时器 / 事件计数器 F 匹配检测中断		
56	INTUSB	USB中断		
57	INTCANGB	CAN状态中断		
58	INTAD	AD转换完成中断		
59	INTSSPO	SSP中断		
60	INTRX5	串行通道 5 接收中断		
61	INTTX5	串行通道 5 传输中断		
62	INTRX6	串行通道 6 接收中断		
63	INTTX6	串行通道 6 传输中断		
64	INTRX7	串行通道 7 接收中断		
65	INTTX7	串行通道 7 传输中断		
66	INTRX8	串行通道 8 接收中断		
67	INTTX8	串行通道 8 传输中断		
68	INTRX9	串行通道 9 接收中断		
69	INTTX9	串行通道 9 传输中断		
70	INTRX10	串行通道 10 接收中断		
71	INTTX10	串行通道 10 传输中断		
72	INTRX11	串行通道 11 接收		
73	INTTX11	串行通道 11 传输		
74	INTCAP10	16-位定时器 / 事件计数器1输入捕获中断 0		
75	INTCAP11	16-位定时器 / 事件计数器1输入捕获中断 1		
76	INTCAP20	16-位定时器 / 事件计数器2输入捕获中断 0		

图7-3 中断源列表

编号	中断源		有效电平 (解除待机和中断)	CG中断模式控制寄存器
77	INTCAP21	16-位定时器 / 事件计数器 2 输入捕获中断 1		
78	INTCANRX	CAN接收中断		
79	INTCANTX	CAN传输中断		
80	INTCAP50	16-位定时器 / 事件计数器 5 输入捕获中断 0		
81	INTCAP51	16-位定时器 / 事件计数器 5 输入捕获中断 1		
82	INTCAP60	16-位定时器 / 事件计数器 6 输入捕获中断 0		
83	INTCAP61	16-位定时器 / 事件计数器 6 输入捕获中断 1		
84	INTCAP70	16-位定时器 / 事件计数器 7 输入捕获中断 0		
85	INTCAP71	16-位定时器 / 事件计数器 7 输入捕获中断 1		
86	INTCAP90	16-位定时器 / 事件计数器 9 输入捕获中断 0		
87	INTCAP91	16-位定时器 / 事件计数器 9 输入捕获中断 1		
88	INTCAPA0	16-位定时器 / 事件计数器 A 输入捕获中断 0		
89	INTCAPA1	16-位定时器 / 事件计数器 A 输入捕获中断 1		
90	INTCAPB0	16-位定时器 / 事件计数器 B 输入捕获中断 0		
91	INTCAPB1	16-位定时器 / 事件计数器 B 输入捕获中断 1		
92	INTCAPD0	16-位定时器 / 事件计数器 D 输入捕获中断 0		
93	INTCAPD1	16-位定时器 / 事件计数器 D 输入捕获中断 1		
94	INTCAPE0	16-位定时器 / 事件计数器 E 输入捕获中断 0		
95	INTCAPE1	16-位定时器 / 事件计数器 E 输入捕获中断 1		
96	INTCAPF0	16-位定时器 / 事件计数器 F 输入捕获中断 0		
97	INTCAPF1	16-位定时器 / 事件计数器 F 输入捕获中断 1		
98	INTDMACERR	DMA错误状态中断		
99	INTDMACTC0	DMA终端计数状态中断		

7.5.1.6 激活电平

有效电平意味着中断源信号触发中断时的变化。CPU把“高“电平中断信号识别为中断。直接从外围函数发送到CPU的中断信号输出为“高“时预示产生了中断请求。

时钟发生器设置有效电平以产生可以作为解除待机模式的中断。外围函数的中断请求设为由上升沿或下降沿引发。中断引脚的中断请求设为电平敏感 (“高“或“低“)或边沿触发 (上升或下降)。

如果中断源用于清除待机模式，需要设置相关的时钟发生器。启用CGIMCGx<INTxEN>位，设定CGIMCGx<EMCGx>位内有效电平。必须设置图7-3所示各外围函数的中断请求的有效电平。

时钟发生器检测到的中断请求被CPU识别为“高“电平信号。

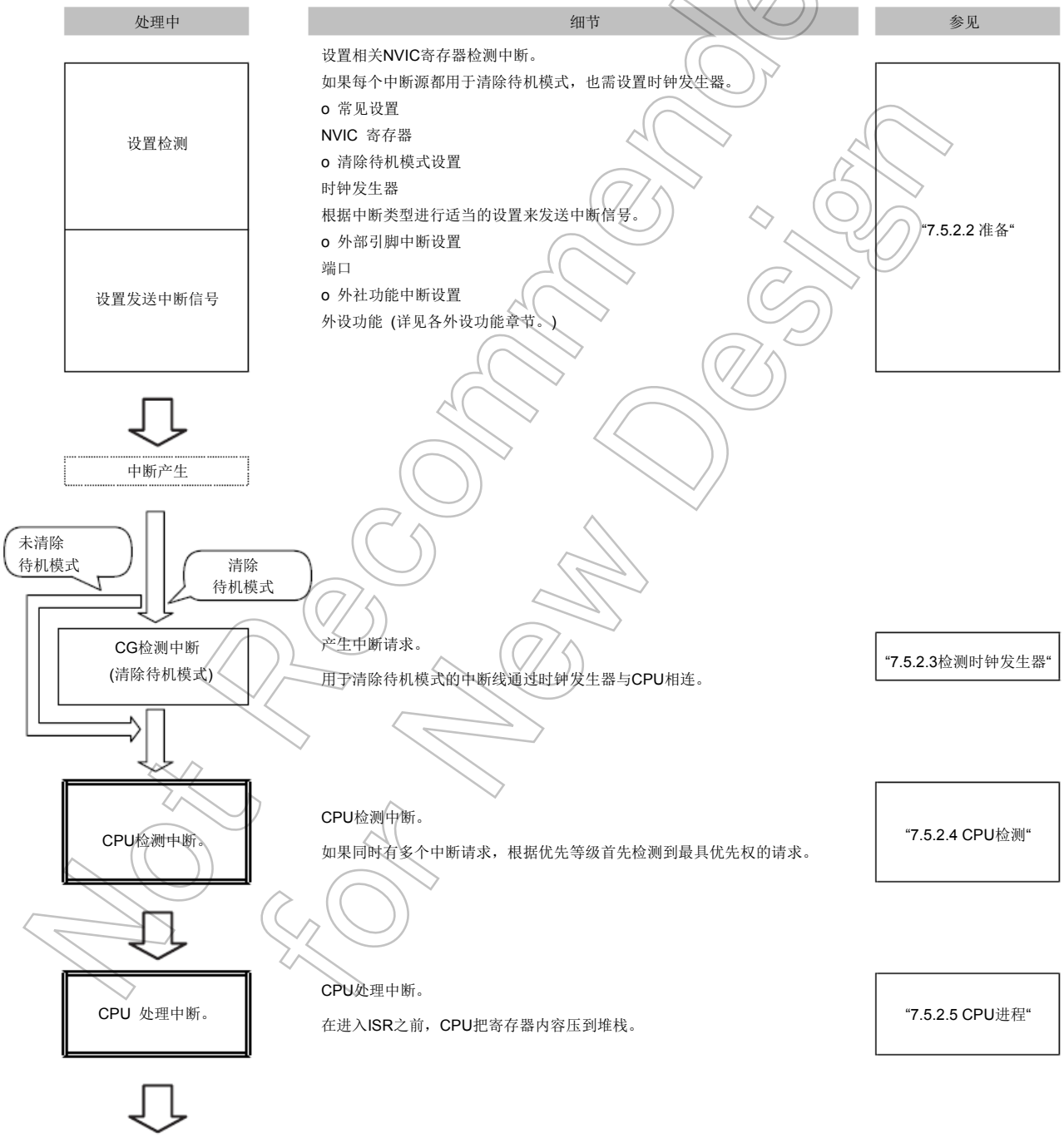
注:**对于CEC接收 / 传送，遥控信号接收和即时时钟中断，设置<INTxEN> 位到 “1“ 并设定有效电平，即使它们不用清除待机模式。**

7.5.2 中断处理

7.5.2.1 流程图

下面显示了如何处理一个中断。

下面说明如何处理一个异常/中断。在以下的说明中，[ ] 表示硬件处理。[ ] 表示软件处理。



处理中	细节	参见
ISR执行	ISR程序。 如需要清除中断源。	“7.5.2.6 中断服务路由 (ISR) “
⇩		
返回到之前的程序	设置为返回到之前的 ISR 程序。	

7.5.2.2 准备

准备中断时，需要注意配置的等级以避免出现任何意外中断。

发起中断或改变其配置必须基本按照以下顺序执行。CPU禁用中断。配置离CPU最远的路径。然后CPU启用中断。

设置时钟发生器时，必须遵循以下顺序以防产生任何意外中断。第一，设置前提。第二，清除与时钟发生器中断相关的数据，然后启用中断。

以下部分以中断处理的等级列出，描述了怎么设置。

1. CPU禁用中断
2. CPU寄存器设置
3. 预置 (1) (外部引脚中断)
4. 预置 (2) (外设功能中断)
5. 预置 (3) (中断挂起寄存器)
6. 配置时钟发生器
7. CPU启用中断

(1) CPU禁用中断

为使CPU不接受任何中断，在PRIMASK寄存器的相应位中写入 “1“。所有中断和异常可被屏蔽，非可屏蔽中断和严重故障不可被屏蔽。

使用 “MSR“指令来设置寄存器。

中断屏蔽寄存器		
PRIMASK	←	“1“ (中断禁用)

注1: PRIMASK寄存器不能被用户访问级别修改。

注2: PRIMASK寄存器设置 “1“出现的错误称为严重故障。

(2) CPU寄存器设置

可以通过写入NVIC寄存器的中断优先级寄存器里的<PRI_n>区域来分配优先等级。

为每个中断源提供8位用于分配从0～255的优先级，但是每个产品的位数目不同。优先级0是最高优先级。如果多个中断源有相同的优先级，最小位数的中断源优先级最高。

可以使用应用程序中断和复位控制寄存器内的<PRIGROUP>分配优先权。

NVIC寄存器		
<PRI_n>	←	“优先权“
<PRIGROUP>	←	“分配优先权 “ (需要时可配置。)

注: “n“指相应的异常或中断。

本产品使用3位分配一个优先级。

(3) 预置 (1) (外部引脚中断)

设置相应引脚的端口功能寄存器为 “1“。PxFRn[m]的设置可使引脚作为功能引脚使用。PxIE[m] 的设置可使引脚作为输入端口使用。

端口寄存器		
PxFRn<PxmFn>	←	“1“
PxIE<PxmIE>	←	“1“

注:x: 端口数量/m: 对应位 / n: 功能寄存器数In 模式不同于STOP 模式。不管PxFR 怎样设置，均设置PxIE为启用输入使相应中断输入可用。注意不要启用不使用的中断。而且注意 “7.5.1.4使用外部中断引脚预告警“的描述。

(4) 预置 (2) (外设功能中断)

使用的外设功能不同设置不同。详见各外设功能章节。

(5) 预置 (3) (中断挂起寄存器)

使用中断挂起寄存器产生中断，设置该寄存器的相应位为 “1“。

NVIC寄存器		
中断挂起寄存器 [m]	←	“1“

注:m: 相应位

(6) 设置时钟发生器

对于用于退出待机模式的中断源，需要设置有效电平并启用时钟发生器的CGIMCG寄存器内的中断。CGIMCG寄存器可设置每个中断源。

启用一个中断之前，清除已持有的相应中断请求。这可以避免意外中断。为了清除相应中断请求，在用于CGICRCG寄存器的中断中写入相应的值。各个值请见 “7.6.3.7CGICRCG (CG中断请求清除寄存器) “。

如果外部引脚中断请求不用于退出待机模式，可以不设置时钟发生器。然而, “高“脉冲或“高“电平信号必须输入，便于CPU可以检测为中断请求。 而且注意“7.5.1.4使用外部中断引脚预告警“的描述。

时钟发生器寄存器		
CGIMCGn<EMCGm>	←	激活电平
CGICRCG<ICRCG>	←	所用中断的相应值
CGIMCGn<INTmEN>	←	“1“ (启用中断)

注:n.寄存器号 / m: 分配到中断源的数量

(7) CPU启用中断

CPU启用中断如下所示。

清除中断“清除挂起“寄存器的中断。启用中断使能寄存器的预期中断。寄存器的每个位被分配到单独的中断源。

在中断清除延迟寄存器相应位中写入“1“清除挂起的中断。在中断使能寄存器相应位中写入“1“启用预期中断。

设置中断延迟寄存器产生中断，如果延迟中断被清除则触发中断的因素就会丢失。因此，这个操作没有必要。

最后，PRIMASK寄存器被清零。

NVIC寄存器		
中断清除-挂起寄存器[m]	←	“1“
中断设置-挂起寄存器 [m]	←	“1“
中断屏蔽寄存器		
PRIMASK	←	“0“

注 1: **m： 相应位**

注 2 : **PRIMASK寄存器不能被用户访问级别修改。**

7.5.2.3 通过时钟生成器检测

如果中断源用于退出待机模式，根据时钟发生器指定的有效电平检测到中断请求，不为CPU识别。

一旦检测到边缘触发的中断请求则保留在时钟发生器内。必须保留对电平敏感的请求到其被检测到，否则当信号电平从有效改变为无效时，这个中断请求将会停止。

当时钟发生器检测到中断请求时，它持续发送“高“电平中断信号到CPU，直到中断请求在CG中断请求清除寄存器 (CGICRCG)内被清除为止。如果没有清除中断请求便退出待机模式，正常操作时将会检测到相同的中断。一定要清除ISR内的每一个中断。

7.5.2.4 通过CPU检测

CPU检测最具优先权的中断请求。

7.5.2.5 CPU处理

检测中断时，CPU把PC，PSR，r0-r3，r12和LR的内容压到堆栈，然后进入ISR。

7.5.2.6 中断服务程序 (ISR)

ISR需要根据应用程序进行特定的编程。本节描述了服务程序编程的推荐和如何清除中断源。

(1) ISR入栈

通常ISR把寄存器内容压入堆栈内并按需处理中断。Cortex-M3核自动把PC, PSR, r0-r3, r12和LR的内容压入堆栈。不需要额外的编程。

如果需要可入栈其他寄存器的内容。

即使执行ISR，具有较高优先权的中断和类似NMI的异常也会被接受。我们建议您将可能被重写的通用寄存器的内容入栈。

(2) 清除中断源

如果中断源用于清除待机模式，每个中断请求必须用CG中断请求清除寄存器(CGICRCG)。

如果中断源设为电平敏感，中断请求会持续存在，除非在中断源处清除。因此，必须清除中断源。自动清除中断源清除了时钟发生器发出的中断请求信号。

如果中断设置为边缘敏感，通过设置CGICRCG寄存器的相应值来清除中断请求。如果有效边再次出现，将会检测到新的中断请求。

7.6 异常 / 中断相关寄存器

本章描述的CPU的寄存器和时钟发生器以及它们各自的地址如下所示。

7.6.1 寄存器列表

NVIC寄存器 基址 = 0×E000_E000

寄存器名称	地址
SysTick控制和状态寄存器	0×0010
SysTick重载值寄存器	0×0014
SysTick当前值寄存器	0×0018
SysTick校准值寄存器	0×001C
中断设置-启动寄存器 1	0×0100
中断设置-启动寄存器 2	0×0104
中断设置-启动寄存器 3	0×0108
中断设置-启动寄存器 4	0×010C
中断清除-启动寄存器 1	0×0180
中断清除-启动寄存器 2	0×0184
中断清除-启动寄存器 3	0×0188
中断清除-启动寄存器 4	0×018C
中断设置-挂起寄存器 1	0×0200
中断设置-挂起寄存器 2	0×0204
中断设置-挂起寄存器 3	0×0208
中断设置-挂起寄存器 4	0×020C
中断清除-挂起寄存器 1	0×0280
中断清除-挂起寄存器 2	0×0284
中断清除-挂起寄存器 3	0×0288
中断清除-挂起寄存器 4	0×028C
中断优先寄存器	0×0400 ~ 0×0460
矢量表偏移寄存器	0×0D08
应用程序中断和复位控制寄存器	0×0D0C
系统处理程序优先级寄存器	0×0D18, 0×0D1C, 0×0D20
系统处理程序控制和状态寄存器	0×0D24

时钟发生器器寄存器 基址 = 0×400F_4000

寄存器名称		地址
CG中断请求清除寄存器	CGICRCG	0×0014
NMI标记寄存器	CGNMIFLG	0×0018
复位标记寄存器	CGRSTFLG	0×001C
CG中断模式控制寄存器 A	CGIMCGA	0×0020
CG中断模式控制寄存器 B	CGIMCGB	0×0024
CG中断模式控制寄存器 C	CGIMCGC	0×0028
CG中断模式控制寄存器 D	CGIMCGD	0×002C
CG中断模式控制寄存器 E	CGIMCGE	0×0030
CG中断模式控制寄存器 F	CGIMCGF	0×0034
保留	-	0×0038
保留	-	0x003C

注:禁止访问 “保留“区域。

7.6.2 NVIC寄存器

7.6.2.1 SysTick控制与状态寄存器

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	COUNTFLAG
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	-	-	-	-	-	CLKSOURCE	TICKINT	ENABLE
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-17	–	R	读作 0。
16	COUNTFLAG	R/W	0: 计时器读数不为 0 1: 计时器读数为 0 如果自从上次读数之后计时器读数为 “0“，则返回 “1“。 清除SysTick控制和状态寄存器的任何部分的读数。
15-3	–	R	读作 0。
2	CLKSOURCE	R/W	0: 外部参考时钟 (fosc/32) (注) 1: CPU主频 (fsys)
1	TICKINT	R/W	0: 不挂起SysTick 1: 挂起SysTick
0	ENABLE	R/W	0: 禁用 1: 启用 如果设置了 “1“，它会重新加载重载值寄存器的值，并开始工作。

注: 在本产品中，32 f_{osc}用作外部参考时钟。

7.6.2.2 SysTick重新加载值寄存器

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	RELOAD							
复位后	未定义							
	15	14	13	12	11	10	9	8
比特符号	RELOAD							
复位后	未定义							
	7	6	5	4	3	2	1	0
比特符号	RELOAD							
复位后	未定义							

位	比特符号	类型	功能
31-24	−	R	读作 0,
23-0	RELOAD	R/W	重载值 当计时器到 “0“时，设置一个值存入SysTick当前值寄存器。

7.6.2.3 SysTick当前值寄存器

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	CURRENT							
复位后	未定义							
	15	14	13	12	11	10	9	8
比特符号	CURRENT							
复位后	未定义							
	7	6	5	4	3	2	1	0
比特符号	CURRENT							
复位后	未定义							

位	比特符号	类型	功能
31-24	−	R	读作 0。
23-0	CURRENT	R/W	[读] 当前SysTick计时器值 [写] 清除 把写入寄存器的任何值均清除为 0。 清除寄存器的同时也清除SysTick控制和状态寄存器的<COUNTFLAG>位。

7.6.2.4 SysTick校准值寄存器

	31	30	29	28	27	26	25	24
比特符号	NOREF	SKEW	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	TENMS							
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	TENMS							
复位后	0	0	0	0	1	0	0	1
	7	6	5	4	3	2	1	0
比特符号	TENMS							
复位后	1	1	0	0	0	1	0	0

位	比特符号	类型	功能
31	NOREF	R	0: 提供参考时钟 1: 无参考时钟
30	SKEW	R	0: 校准值10 ms。 1: 校准值不是10 ms。
29-24	–	R	读作 0。
23-0	TENMS	R	校准值 重载数值用于外部参考时钟的 10 ms定时(0x9C4) (备注)。

注:在多发的情况下，请使用<TENMS>-1.

7.6.2.5 中断设置-启用寄存器 1

	31	30	29	28	27	26	25	24
比特符号	SETENA (中断31)	SETENA (中断30)	SETENA (中断29)	SETENA (中断28)	SETENA (中断27)	SETENA (中断26)	SETENA (中断25)	SETENA (中断24)
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	SETENA (中断23)	SETENA (中断22)	SETENA (中断21)	SETENA (中断20)	SETENA (中断19)	SETENA (中断18)	SETENA (中断17)	SETENA (中断16)
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	SETENA (中断13)	SETENA (中断12)	SETENA (中断11)	SETENA (中断10)	SETENA (中断9)	SETENA (中断8)
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	SETENA (中断7)	SETENA (中断6)	SETENA (中断5)	SETENA (中断4)	SETENA (中断3)	SETENA (中断2)	SETENA (中断1)	SETENA (中断0)
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-16	SETENA	R/W	中断号[31:16] [写] 1: 启用 [读] 0: 禁用 1: 启用 每个位均与特定中断号相对应。 在此寄存器的位中写入 “1“启用相应中断。 写入 “0“无反应。 读这些位可以看到相应中断启用或禁用的状态。
15-14	–	R/W	写作 0。
13-0	SETENA	R/W	中断号[13:0] [写] 1: 启用 [读] 0: 禁用 1: 启用 每个位均与特定中断号相对应。 在此寄存器的位中写入 “1“启用相应中断。 写入 “0“无反应。 读这些位可以看到相应中断启用或禁用的状态。

注:关于中断和中断号的描述见 “7.5.1.5中断源列表“。

7.6.2.6 中断设置-启用寄存器 2

	31	30	29	28	27	26	25	24
比特符号	SETENA (中断 63)	SETENA (中断62)	SETENA (中断61)	SETENA (中断60)	SETENA (中断59)	SETENA (中断58)	SETENA (中断57)	SETENA (中断56)
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	SETENA (中断55)	SETENA (中断54)	SETENA (中断53)	SETENA (中断52)	SETENA (中断51)	SETENA (中断50)	SETENA (中断49)	SETENA (中断48)
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	SETENA (中断47)	SETENA (中断46)	SETENA (中断45)	SETENA (中断44)	SETENA (中断43)	SETENA (中断42)	SETENA (中断41)	SETENA (中断40)
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	SETENA (中断39)	SETENA (中断38)	SETENA (中断37)	SETENA (中断36)	SETENA (中断35)	SETENA (中断34)	SETENA (中断33)	SETENA (中断32)
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-0	SETENA	R/W	中断号[63:32] [写] 1: 启用 [读] 0: 禁用 1: 启用 每个位均与特定中断号相对应。 在此寄存器的位中写入 “1“启用相应中断。 写入 “0“无反应。 读这些位可以看到相应中断启用或禁用的状态。

注:关于中断和中断号的描述见 “7.5.1.5中断源列表“。

7.6.2.7 中断设置启用寄存器 3

	31	30	29	28	27	26	25	24
比特符号	SETENA (中断95)	SETENA (中断94)	SETENA (中断93)	SETENA (中断92)	SETENA (中断91)	SETENA (中断90)	SETENA (中断89)	SETENA (中断88)
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	SETENA (中断87)	SETENA (中断86)	SETENA (中断85)	SETENA (中断84)	SETENA (中断83)	SETENA (中断82)	SETENA (中断81)	SETENA (中断80)
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	SETENA (中断79)	SETENA (中断78)	SETENA (中断77)	SETENA (中断76)	SETENA (中断75)	SETENA (中断74)	SETENA (中断73)	SETENA (中断72)
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	SETENA (中断71)	SETENA (中断70)	SETENA (中断69)	SETENA (中断68)	SETENA (中断67)	SETENA (中断66)	SETENA (中断65)	SETENA (中断64)
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-0	SETENA	R/W	中断号[95:64] [写] 1: 启用 [读] 0: 禁用 1: 启用 每个位均与特定中断号相对应。 在此寄存器的位中写入 “1“启用相应中断。 写入 “0“无反应。 读这些位可以看到相应中断启用或禁用的状态。

注:关于中断和中断号的描述见 “7.5.1.5 中断源列表“。

7.6.2.8 中断设置启用寄存器 4

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	-	-	-	-	SETENA (中断99)	SETENA (中断98)	SETENA (中断97)	SETENA (中断96)
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-4	−	R	读作 0,
3-0	SETENA	R/W	中断号[99:96] [写] 1: 启用 [读] 0: 禁用 1: 启用 每个位均与特定中断号相对应。 在此寄存器的位中写入 “1“启用相应中断。写入 “0“无反应。 读这些位可以看到相应中断启用或禁用的状态。

注:关于中断和中断号的描述见 “7.5.1.5 中断源列表“。

7.6.2.9 中断清除-启用寄存器 1

	31	30	29	28	27	26	25	24
比特符号	CLRENA (中断31)	CLRENA (中断30)	CLRENA (中断29)	CLRENA (中断28)	CLRENA (中断27)	CLRENA (中断26)	CLRENA (中断25)	CLRENA (中断24)
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	CLRENA (中断23)	CLRENA (中断22)	CLRENA (中断21)	CLRENA (中断20)	CLRENA (中断19)	CLRENA (中断18)	CLRENA (中断17)	CLRENA (中断16)
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	CLRENA (中断13)	CLRENA (中断12)	CLRENA (中断11)	CLRENA (中断10)	CLRENA (中断9)	CLRENA (中断8)
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	CLRENA (中断7)	CLRENA (中断6)	CLRENA (中断5)	CLRENA (中断4)	CLRENA (中断3)	CLRENA (中断2)	CLRENA (中断1)	CLRENA (中断0)
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-16	CLRENA	R/W	中断号[31:16] [写] 1: 禁用 [读] 0: 禁用 1: 启用 每个位均与特定中断号相对应。它可执行中断并检查中断是否被禁用。 在此寄存器中的一个位元写入 “1“禁用相应的中断。写入 “0“无反应。 读这些位可以看到相应中断启用或禁用的状态。
15-14	−	R/W	写做0。
13-0	CLRENA	R/W	中断号[13:0] [写] 1: 禁用 [读] 0: 禁用 1: 启用 每个位均与特定中断号相对应。它可执行中断并检查中断是否被禁用。 在此寄存器中的一个位元写入 “1“禁用相应的中断。写入 “0“无反应。 读这些位可以看到相应中断启用或禁用的状态。

注:关于中断和中断号的描述见 “7.5.1.5 中断源列表“。

7.6.2.10 中断清除启用寄存器 2

	31	30	29	28	27	26	25	24
比特符号	CLRENA (中断63)	CLRENA (中断62)	CLRENA (中断61)	CLRENA (中断60)	CLRENA (中断59)	CLRENA (中断58)	CLRENA (中断57)	CLRENA (中断56)
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	CLRENA (中断55)	CLRENA (中断54)	CLRENA (中断53)	CLRENA (中断52)	CLRENA (中断51)	CLRENA (中断50)	CLRENA (中断49)	CLRENA (中断48)
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	CLRENA (中断47)	CLRENA (中断46)	CLRENA (中断45)	CLRENA (中断44)	CLRENA (中断43)	CLRENA (中断42)	CLRENA (中断41)	CLRENA (中断40)
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	CLRENA (中断39)	CLRENA (中断38)	CLRENA (中断37)	CLRENA (中断36)	CLRENA (中断35)	CLRENA (中断34)	CLRENA (中断33)	CLRENA (中断32)
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-0	CLRENA	R/W	中断号[63:32] [写] 1: 禁用 [读] 0: 禁用 1: 启用 每个位均与特定中断号相对应。它可执行中断并检查中断是否被禁用。 在此寄存器中的一个位元写入 “1“禁用相应的中断。写入 “0“无反应。 读这些位可以看到相应中断启用或禁用的状态。

注:关于中断和中断号的描述见 “7.5.1.5中断源列表“。

7.6.2.11 中断清除-启用寄存器 3

	31	30	29	28	27	26	25	24
比特符号	CLRENA (中断95)	CLRENA (中断94)	CLRENA (中断93)	CLRENA (中断92)	CLRENA (中断91)	CLRENA (中断90)	CLRENA (中断89)	CLRENA (中断88)
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	CLRENA (中断87)	CLRENA (中断86)	CLRENA (中断85)	CLRENA (中断84)	CLRENA (中断83)	CLRENA (中断82)	CLRENA (中断81)	CLRENA (中断80)
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	CLRENA (中断79)	CLRENA (中断78)	CLRENA (中断77)	CLRENA (中断76)	CLRENA (中断75)	CLRENA (中断74)	CLRENA (中断73)	CLRENA (中断72)
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	CLRENA (中断71)	CLRENA (中断70)	CLRENA (中断69)	CLRENA (中断68)	CLRENA (中断67)	CLRENA (中断66)	CLRENA (中断65)	CLRENA (中断64)
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-0	CLRENA	R/W	中断号[95:64] [写] 1: 禁用 [读] 0: 禁用 1: 启用 每个位均与特定中断号相对应。它可执行中断并检查中断是否被禁用。 在此寄存器中的一个位元写入 “1“禁用相应的中断。写入 “0“无反应。 读这些位可以看到相应中断启用或禁用的状态。

注:关于中断和中断号的描述见 “7.5.1.5 中断源列表“。

7.6.2.12 中断清除-启用寄存器 4

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	-	-	-	-	CLRENA (中断99)	CLRENA (中断98)	CLRENA (中断97)	CLRENA (中断96)
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-4	−	R	读作0,
3-0	CLRENA	R/W	中断号[99:96] [写] 1:禁用 [读] 0:禁用 1:启用 每个位均与特定中断号相对应。它可执行中断并检查中断是否被禁用。 在此寄存器中的一个位元写入“1“禁用相应的中断。写入 “0“无反应。 读这些位可以看到相应中断启用或禁用的状态。

注:关于中断和中断号的描述见 “7.5.1.5中断源列表“。

7.6.2.13 中断设置-挂起寄存器 1

	31	30	29	28	27	26	25	24
比特符号	SETPEND (中断31)	SETPEND (中断30)	SETPEND (中断29)	SETPEND (中断28)	SETPEND (中断27)	SETPEND (中断26)	SETPEND (中断25)	SETPEND (中断24)
复位后	未定义	未定义	未定义	未定义	未定义	未定义	未定义	未定义
	23	22	21	20	19	18	17	16
比特符号	SETPEND (中断23)	SETPEND (中断22)	SETPEND (中断21)	SETPEND (中断20)	SETPEND (中断19)	SETPEND (中断18)	SETPEND (中断17)	SETPEND (中断16)
复位后	未定义	未定义	未定义	未定义	未定义	未定义	未定义	未定义
	15	14	13	12	11	10	9	8
比特符号	-	-	SETPEND (中断13)	SETPEND (中断12)	SETPEND (中断11)	SETPEND (中断10)	SETPEND (中断9)	SETPEND (中断8)
复位后	未定义	未定义	未定义	未定义	未定义	未定义	未定义	未定义
	7	6	5	4	3	2	1	0
比特符号	SETPEND (中断7)	SETPEND (中断6)	SETPEND (中断5)	SETPEND (中断4)	SETPEND (中断3)	SETPEND (中断2)	SETPEND (中断1)	SETPEND (中断0)
复位后	未定义	未定义	未定义	未定义	未定义	未定义	未定义	未定义

位	比特符号	类型	功能
31-16	SETPEND	R/W	中断号[31:16] [写] 1: 挂起 [读] 0: 未挂起 1: 挂起中 对应规定的数的每一个位可迫使中断进入挂起状态，并确定哪些中断正在挂起。 将在此寄存器中的一个位元写入 “1“将对应的中断挂起。但是写入 “1“并不会影响已经挂起或禁用的某个中断。 写入 “0“摄无反应。 读入该位将对应的中断返回当前状态。 在中断清除-挂起寄存器中的一个对应位写入 “1“，清除此寄存器中的该位元。
15-14	–	R/W	写做0。
13-0	SETPEND	R/W	中断号[13:0] [写] 1: 挂起 [读] 0: 未挂起 1: 挂起中 对应规定的数的每一个位可迫使中断进入挂起状态，并确定哪些中断正在挂起。 将在此寄存器中的一个位元写入 “1“将对应的中断挂起。但是写入 “1“并不会影响已经挂起或禁用的某个中断。 写入 “0“无反应。 读入该位将对应的中断返回当前状态。 在中断清除-挂起寄存器中的一个对应位写入 “1“，清除此寄存器中的该位元。

注:关于中断和中断号的描述见 “7.5.1.5 中断源列表“。

7.6.2.14 中断设置-挂起寄存器 2

	31	30	29	28	27	26	25	24
比特符号	SETPEND (中断63)	SETPEND (中断62)	SETPEND (中断61)	SETPEND (中断60)	SETPEND (中断59)	SETPEND (中断58)	SETPEND (中断57)	SETPEND (中断56)
复位后	未定义	未定义	未定义	未定义	未定义	未定义	未定义	未定义
	23	22	21	20	19	18	17	16
比特符号	SETPEND (中断55)	SETPEND (中断54)	SETPEND (中断53)	SETPEND (中断52)	SETPEND (中断51)	SETPEND (中断50)	SETPEND (中断49)	SETPEND (中断48)
复位后	未定义	未定义	未定义	未定义	未定义	未定义	未定义	未定义
	15	14	13	12	11	10	9	8
比特符号	SETPEND (中断47)	SETPEND (中断46)	SETPEND (中断45)	SETPEND (中断44)	SETPEND (中断43)	SETPEND (中断42)	SETPEND (中断41)	SETPEND (中断40)
复位后	未定义	未定义	未定义	未定义	未定义	未定义	未定义	未定义
	7	6	5	4	3	2	1	0
比特符号	SETPEND (中断39)	SETPEND (中断38)	SETPEND (中断37)	SETPEND (中断36)	SETPEND (中断35)	SETPEND (中断34)	SETPEND (中断33)	SETPEND (中断32)
复位后	未定义	未定义	未定义	未定义	未定义	未定义	未定义	未定义

位	比特符号	类型	功能
31-0	SETPEND	R/W	中断号[63:32] [写] 1: 挂起 [读] 0: 未挂起 1: 挂起中 对应规定的数的每一个位可迫使中断进入挂起状态，并确定哪些中断正在挂起。 将在此寄存器中的一个位元写入 “1“将对应的中断挂起。但是写入 “1“并不会影响已经挂起或禁用的某个中断。 写入 “0“无反应。 读入该位将对应的中断返回当前状态。 在中断清除-挂起寄存器中的一个对应位写入 “1“，清除此寄存器中的该位元。

注:关于中断和中断号的描述见 “7.5.1.5 中断源列表“。

7.6.2.15 中断设置-挂起寄存器 3

	31	30	29	28	27	26	25	24
比特符号	SETPEND (中断95)	SETPEND (中断94)	SETPEND (中断93)	SETPEND (中断92)	SETPEND (中断91)	SETPEND (中断90)	SETPEND (中断89)	SETPEND (中断88)
复位后	未定义	未定义	未定义	未定义	未定义	未定义	未定义	未定义
	23	22	21	20	19	18	17	16
比特符号	SETPEND (中断87)	SETPEND (中断86)	SETPEND (中断85)	SETPEND (中断84)	SETPEND (中断83)	SETPEND (中断82)	SETPEND (中断81)	SETPEND (中断80)
复位后	未定义	未定义	未定义	未定义	未定义	未定义	未定义	未定义
	15	14	13	12	11	10	9	8
比特符号	SETPEND (中断79)	SETPEND (中断78)	SETPEND (中断77)	SETPEND (中断76)	SETPEND (中断75)	SETPEND (中断74)	SETPEND (中断73)	SETPEND (中断72)
复位后	未定义	未定义	未定义	未定义	未定义	未定义	未定义	未定义
	7	6	5	4	3	2	1	0
比特符号	SETPEND (中断71)	SETPEND (中断70)	SETPEND (中断69)	SETPEND (中断68)	SETPEND (中断67)	SETPEND (中断66)	SETPEND (中断65)	SETPEND (中断64)
复位后	未定义	未定义	未定义	未定义	未定义	未定义	未定义	未定义

位	比特符号	类型	功能
31-0	SETPEND	R/W	中断号[95:64] [写] 1: 挂起 [读] 0: 未挂起 1: 挂起中 对应规定的数的每一个位可迫使中断进入挂起状态，并确定哪些中断正在挂起。 将在此寄存器中的一个位元写入 “1“将对应的中断挂起。但是写入 “1“并不会影响已经挂起或禁用的某个中断。 写入 “0“无反应。 读入该位将对应的中断返回当前状态。 在中断清除-挂起寄存器中的一个对应位写入 “1“，清除此寄存器中的该位元。

注: 关于中断和中断号的描述见 “7.5.1.5中断源列表“。

7.6.2.16 中断设置-挂起寄存器 4

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	-	-	-	-	SETPEND (中断99)	SETPEND (中断98)	SETPEND (中断97)	SETPEND (中断96)
复位后	0	0	0	0	未定义	未定义	未定义	未定义

位	比特符号	类型	功能
31-4	−	R	读作 0,
3-0	SETPEND	R/W	中断号[99:96] [写] 1: 挂起 [读] 0: 未挂起 1: 挂起中 对应规定的数的每一个位可迫使中断进入挂起状态，并确定哪些中断正在挂起。 将在此寄存器中的一个位元写入 “1“将对应的中断挂起。但是写入 “1“并不会影响已经挂起或禁用的某个中断。 写入 “0“无反应。 读入该位将对应的中断返回当前状态。 在中断清除-挂起寄存器中的一个对应位写入 “1“，清除此寄存器中的该位元。

注: 关于中断和中断号的描述见 “7.5.1.5 中断源列表“。

7.6.2.17 中断清除-挂起寄存器 1

	31	30	29	28	27	26	25	24
比特符号	CLRPEND (中断31)	CLRPEND (中断30)	CLRPEND (中断29)	CLRPEND (中断28)	CLRPEND (中断27)	CLRPEND (中断26)	CLRPEND (中断25)	CLRPEND (中断24)
复位后	未定义	未定义	未定义	未定义	未定义	未定义	未定义	未定义
	23	22	21	20	19	18	17	16
比特符号	CLRPEND (中断23)	CLRPEND (中断22)	CLRPEND (中断21)	CLRPEND (中断20)	CLRPEND (中断19)	CLRPEND (中断18)	CLRPEND (中断17)	CLRPEND (中断16)
复位后	未定义	未定义	未定义	未定义	未定义	未定义	未定义	未定义
	15	14	13	12	11	10	9	8
比特符号	-	-	CLRPEND (中断13)	CLRPEND (中断12)	CLRPEND (中断11)	CLRPEND (中断10)	CLRPEND (中断9)	CLRPEND (中断8)
复位后	未定义	未定义	未定义	未定义	未定义	未定义	未定义	未定义
	7	6	5	4	3	2	1	0
比特符号	CLRPEND (中断7)	CLRPEND (中断6)	CLRPEND (中断5)	CLRPEND (中断4)	CLRPEND (中断3)	CLRPEND (中断2)	CLRPEND (中断1)	CLRPEND (中断0)
复位后	未定义	未定义	未定义	未定义	未定义	未定义	未定义	未定义

位	比特符号	类型	功能
31-16	CLRPEND	R/W	中断号[31:16] [写] 1: 清除挂起中断 [读] 0: 未挂起 1: 挂起 对应规定的数的每一个位可迫使中断进入挂起状态，并确定哪些中断正在挂起。 在此寄存器中写 “1“将会清除对应的挂起中断。然而，写入 “1“对于已操作的中断无效。 写入 “0“无反应。 读入该位将对应的中断返回当前状态。
15-14	–	R/W	写作0。
13-0	CLRPEND	R/W	中断号[13:0] [写] 1: 清除挂起中断 [读] 0: 未挂起 1: 挂起 对应规定的数的每一个位可迫使中断进入挂起状态，并确定哪些中断正在挂起。 在此寄存器中写 “1“将会清除对应的挂起中断。然而，写入 “1“对于已操作的中断无效。写入 “0“无反应。 读入该位将对应的中断返回当前状态。

注: 关于中断和中断号的描述见 “7.5.1.5 中断源列表“。

7.6.2.18 中断清除-挂起寄存器 2

	31	30	29	28	27	26	25	24
比特符号	CLRPEND (中断63)	CLRPEND (中断62)	CLRPEND (中断61)	CLRPEND (中断60)	CLRPEND (中断59)	CLRPEND (中断58)	CLRPEND (中断57)	CLRPEND (中断56)
复位后	未定义	未定义	未定义	未定义	未定义	未定义	未定义	未定义
	23	22	21	20	19	18	17	16
比特符号	CLRPEND (中断55)	CLRPEND (中断54)	CLRPEND (中断53)	CLRPEND (中断52)	CLRPEND (中断51)	CLRPEND (中断50)	CLRPEND (中断49)	CLRPEND (中断48)
复位后	未定义	未定义	未定义	未定义	未定义	未定义	未定义	未定义
	15	14	13	12	11	10	9	8
比特符号	CLRPEND (中断47)	CLRPEND (中断46)	CLRPEND (中断45)	CLRPEND (中断44)	CLRPEND (中断43)	CLRPEND (中断42)	CLRPEND (中断41)	CLRPEND (中断40)
复位后	未定义	未定义	未定义	未定义	未定义	未定义	未定义	未定义
	7	6	5	4	3	2	1	0
比特符号	CLRPEND (中断39)	CLRPEND (中断38)	CLRPEND (中断37)	CLRPEND (中断36)	CLRPEND (中断35)	CLRPEND (中断34)	CLRPEND (中断33)	CLRPEND (中断32)
复位后	未定义	未定义	未定义	未定义	未定义	未定义	未定义	未定义

位	比特符号	类型	功能
31-0	CLRPEND	R/W	中断号[63:32] [写] 1: 清除挂起中断 [读] 0: 未挂起 1: 挂起 对应规定的数的每一个位可迫使中断进入挂起状态，并确定哪些中断正在挂起。 在此寄存器中写“1“将会清除对应的挂起中断。然而，写入“1“对于已操作的中断无效。写入“0“无反应。 读入该位将对应的中断返回当前状态。

注: 关于中断和中断号的描述见“7.5.1.5 中断源列表“。

7.6.2.19 中断清除-挂起寄存器 3

	31	30	29	28	27	26	25	24
比特符号	CLRPEND (中断95)	CLRPEND (中断94)	CLRPEND (中断93)	CLRPEND (中断92)	CLRPEND (中断91)	CLRPEND (中断90)	CLRPEND (中断89)	CLRPEND (中断88)
复位后	未定义	未定义	未定义	未定义	未定义	未定义	未定义	未定义
	23	22	21	20	19	18	17	16
比特符号	CLRPEND (中断87)	CLRPEND (中断86)	CLRPEND (中断85)	CLRPEND (中断84)	CLRPEND (中断83)	CLRPEND (中断82)	CLRPEND (中断81)	CLRPEND (中断80)
复位后	未定义	未定义	未定义	未定义	未定义	未定义	未定义	未定义
	15	14	13	12	11	10	9	8
比特符号	CLRPEND (中断79)	CLRPEND (中断78)	CLRPEND (中断77)	CLRPEND (中断76)	CLRPEND (中断75)	CLRPEND (中断74)	CLRPEND (中断73)	CLRPEND (中断72)
复位后	未定义	未定义	未定义	未定义	未定义	未定义	未定义	未定义
	7	6	5	4	3	2	1	0
比特符号	CLRPEND (中断71)	CLRPEND (中断70)	CLRPEND (中断69)	CLRPEND (中断68)	CLRPEND (中断67)	CLRPEND (中断66)	CLRPEND (中断65)	CLRPEND (中断64)
复位后	未定义	未定义	未定义	未定义	未定义	未定义	未定义	未定义

位	比特符号	类型	功能
31-0	CLRPEND	R/W	中断号[95:64] [写] 1: 清除挂起中断 [读] 0: 未挂起 1: 挂起 对应规定的数的每一个位可迫使中断进入挂起状态，并确定哪些中断正在挂起。 在此寄存器中写 “1“将会清除对应的挂起中断。然而，写入 “1“对于已操作的中断无效。写入 “0“无反应。 读入该位将对应的中断返回当前状态。

注: 关于中断和中断号的描述见 “7.5.1.5 中断源列表“。

7.6.2.20 中断清除-挂起寄存器 4

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	-	-	-	-	CLRPEND (中断99)	CLRPEND (中断98)	CLRPEND (中断97)	CLRPEND (中断96)
复位后	0	0	0	0	未定义	未定义	未定义	未定义

位	比特符号	类型	功能
31-4	−	R	读作 0,
3-0	CLRPEND	R/W	中断号[99:96] [写] 1: 清除挂起中断 [读] 0: 未挂起 1: 挂起 对应规定的数的每一个位可迫使中断进入挂起状态，并确定哪些中断正在挂起。 在此寄存器中写 “1“将会清除对应的挂起中断。然而，写入 “1“对于已操作的中断无效。写入 “0“无反应。 读入该位将对应的中断返回当前状态。

注: 关于中断和中断号的描述见 “7.5.1.5 中断源列表“。

7.6.2.21 中断优先级寄存器

每个中断中断优先寄存器的带八位元。

显示与中断数对应的中断优先寄存器地址。

	31 24	23 16	15 8	7 0
0xE000_E400	PRI_3	PRI_2	PRI_1	PRI_0
0xE000_E404	PRI_7	PRI_6	PRI_5	PRI_4
0xE000_E408	PRI_11	PRI_10	PRI_9	PRI_8
0xE000_E40C	-	-	PRI_13	PRI_12
0xE000_E410	PRI_19	PRI_18	PRI_17	PRI_16
0xE000_E414	PRI_23	PRI_22	PRI_21	PRI_20
0xE000_E418	PRI_27	PRI_26	PRI_25	PRI_24
0xE000_E41C	PRI_31	PRI_30	PRI_29	PRI_28
0xE000_E420	PRI_35	PRI_34	PRI_33	PRI_32
0xE000_E424	PRI_39	PRI_38	PRI_37	PRI_36
0xE000_E428	PRI_43	PRI_42	PRI_41	PRI_40
0xE000_E42C	PRI_47	PRI_46	PRI_45	PRI_44
0xE000_E430	PRI_51	PRI_50	PRI_49	PRI_48
0xE000_E434	PRI_55	PRI_54	PRI_53	PRI_52
0xE000_E438	PRI_59	PRI_58	PRI_57	PRI_56
0xE000_E43C	PRI_63	PRI_62	PRI_61	PRI_60
0xE000_E440	PRI_67	PRI_66	PRI_65	PRI_64
0xE000_E444	PRI_71	PRI_70	PRI_69	PRI_68
0xE000_E448	PRI_75	PRI_74	PRI_73	PRI_72
0xE000_E44C	PRI_79	PRI_78	PRI_77	PRI_76
0xE000_E450	PRI_83	PRI_82	PRI_81	PRI_80
0xE000_E454	PRI_87	PRI_86	PRI_85	PRI_84
0xE000_E458	PRI_91	PRI_90	PRI_89	PRI_88
0xE000_E45C	PRI_95	PRI_94	PRI_93	PRI_92
0xE000_E460	PRI_99	PRI_98	PRI_97	PRI_96

用于分配优先权的位数目因产品不同而不同。此产品使用三位数分配一个优先权。

以下显示针对中断数目0～3的中断优先寄存器的域。此中断优先寄存器对所有其它中断数目具有同等的域。未使用的位在读取时归 “0”，写入未使用的位无效。

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

位	比特符号	类型	功能
31-29	PRI_3	R/W	中断数 3 的优先权
28-24	–	R	读作 0。
23-21	PRI_2	R/W	中断数 2 的优先权
20-16	–	R	读作0。
15-13	PRI_1	R/W	中断数 1 的优先权
12-8	–	R	读作 0。
7-5	PRI_0	R/W	中断数 0 的优先权
4-0	–	R	读作 0。

7.6.2.22 向量表偏移寄存器

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

位	比特符号	类型	功能
31-30	−	R	读作 0,
29	TBLBASE	R/W	表基址 矢量表位于 0: 代码空间 1: SRAM空间
28-7	TBLOFF	R/W	补偿值 自TBLBASE中规定的空间顶端设置补偿值。 补偿值必须依据表中异常数量进行对齐。即可用最小32个字节的数列，用于多达16个中断。更多的中断须通过算到后面的两个来调整数列。
6-0	−	R	读作 0,

7.6.2.23 应用中断与复位控制寄存器

	31	30	29	28	27	26	25	24
比特符号	VECTKEY/VECTKEYSTAT							
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	VECTKEY/VECTKEYSTAT							
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	ENDIANESS	-	-	-	-	PRIGROUP		
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	-	-	-	-	-	SYSRESET REQ	VECTCLR ACTIVE	VECTRESET
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-16	VECTKEY (写)/ VECTKEYSTAT (读)	R/W	寄存器键 [写] 写入此寄存器要求<VECTKEY>域中的0×5FA。 [读] 读作 0×FA05。
15	ENDIANESS	R/W	位顺序: (注1) 1: 大端位顺序 0: 小端位顺序
14-11	−	R	读作 0,
10-8	PRIGROUP	R/W	中断优先权分组 000: 七位占先优先权，一位次优先权 001: 六位占先优先权，二位次优先权 010: 五位占先优先权，三位次优先权 011: 四位占先优先权，四位次优先权 100: 三位占先优先权，五位次优先权 101: 二位占先优先权，七位次优先权 110: 一位占先优先权，七位次优先权 111: 无占先优先权，八位次优先权 该位的配置把中断优先寄存器<PRI_n>分割为占先优先权和次优先权。
7-3	−	R	读作 0,
2	SYSRESET REQ	R/W	系统复位请求 1=CPU 输出SYSRESETREQ信号。 (注释2)
1	VECTCLR ACTIVE	R/W	清零激活向量位 1: 清零所有激活NMI，故障，及中断的状态信息。 0: 不清零。 此位自行清零。 应用程序负责重新启动堆叠。
0	VECTRESET	R/W	系统复位位 1: 复位系统 0: 不复位系统。 复位系统，当通过设置 “1“调试元件 (FPB，DWT及ITM)且此位也清零时。

注 1: **小端顺序为本产品的默认内存格式。**

注 2: **输出SYSRESETREQ (系统复位请求)，产品进行热复位。<SYSRESETEQ>通过热复位清零。**

7.6.2.24 系统处理程序优先级寄存器

每个异常都带有系统处理优先寄存的八位。

以下显示与每个异常对应的系统处理优先寄存器地址。

	31 24	23 16	15 8	7 0
0xE000_ED18	PRI_7	PRI_6 (使用故障)	PRI_5 (总线故障)	PRI_4 (储存器管理)
0xE000_ED1C	PRI_11 (SVCall)	PRI_10	PRI_9	PRI_8
0xE000_ED20	PRI_15 (SysTick)	PRI_14 (PendSV)	PRI_13	PRI_12 (调试监测器)

用于分配优先权的位数目因产品不同而不同。此产品使用三位数分配一个优先权。

以下是用于储存器管理，总线故障以及使用故障的系统处理有限寄存器的域。未使用的位在读取时归 “0“，写入未使用的位无效。

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

位	比特符号	类型	功能
31-29	PRI_7	R/W	保留
28-24	-	R	读作 0,
23-21	PRI_6	R/W	使用故障优先权
20-16	-	R	读作 0,
15-13	PRI_5	R/W	总线故障优先权
12-8	-	R	读作 0,
7-5	PRI_4	R/W	储存器管理优先仅
4-0	-	R	读作 0,

7.6.2.25 系统处理程序控制与状态寄存器

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	USGFAULT ENA	BUSFAULT ENA	MEMFAULT ENA
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	SVCALL PENDED	BUSFAULT PENDED	MEMFAULT PENDED	USGFAULT PENDED	SYSTICKACT	PENDSVACT	-	MONITOR ACT
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2。	1	0
比特符号	SVCALLACT	-	-	-	USGFAULT ACT	-	BUSFAULT ACT	MEMFAULT ACT
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-19	–	R	读作 0,
18	USGFAULT ENA	R/W	使用故障 0: 禁用 1: 启用
17	BUSFAULT TENA	R/W	总线故障 0: 禁用 1: 启用
16	MEMFAULT ENA	R/W	内存管理 0: 禁用 1: 启用
15	SVCALL PENDED	R/W	SVCall 0: 未挂起 1: 挂起
14	BUSFAULT PENDED	R/W	总线故障 0: 未挂起 1: 挂起
13	MEMFAULT PENDED	R/W	内存管理 0: 未挂起 1: 挂起
12	USGFAULT PENDED	R/W	使用故障 0: 未挂起 1: 挂起
11	SYSTICKACT	R/W	SysTick 0: 未激活 1: 激活
10	PENDSVACT	R/W	PendSV 0: 未激活 1: 激活
9	–	R	读作 0,
8	MONITORACT	R/W	调试监测器 0: 未激活 1: 激活
7	SVCALLACT	R/W	SVCall 0: 未激活 1: 激活码
6-4	–	R	读作 0,

位	比特符号	类型	功能
3	USGFAULT ACT	R/W	使用故障 0: 未激活 1: 激活
2	−	R	读作 0,
1	BUSFAULT ACT	R/W	总线故障 0: 未激活 1: 激活
0	MEMFAULT ACT	R/W	储存器管理 0: 未激活 1: 激活

注: 激活位元清零或设置时必须高度小心，因为这些位元的清除或激活后堆栈内容不能修复。

7.6.3 时钟发生器寄存器

7.6.3.1 CGIMCGA (CG中断模式控制寄存器 A)

	31	30	29	28	27	26	25	24
比特符号	-	EMCG3			EMST3		-	INT3EN
复位后	0	0	1	0	0	0	未定义	0
	23	22	21	20	19	18	17	16
比特符号	-	EMCG2			EMST2		-	INT2EN
复位后	0	0	1	0	0	0	未定义	0
	15	14	13	12	11	10	9	8
比特符号	-	EMCG1			EMST1		-	INT1EN
复位后	0	0	1	0	0	0	未定义	0
	7	6	5)	4	3	2	1	0
比特符号	-	EMCG0			EMST0		-	INT0EN
复位后	0	0	1	0	0	0	未定义	0

位	比特符号	类型	功能
31	−	R	读作0 ,
30-28	EMCG3[2:0]	R/W	INT3 待机清除请求的激活电平设定。 (101 ~ 111: 禁止设定) 000: “低“电平 001: “高“电平 010: 下降沿 011: 上升沿 100: 双沿
27-26	EMST3[1:0]	R	INT3 待机清除请求的激活电平。 00: − 01: 上升沿 10: 下降沿 11: 双沿
25	−	R	以未定义读取。
24	INT3EN	R/W	INT3 清除输入 0: 禁用 1: 启用
23	−	R	读作0,
22-20	EMCG2[2:0]	R/W	INT2 待机清除请求的激活电平设定。 (101 ~ 111: 禁止设定) 000: “低“电平 001: “高“电平 010: 下降沿 011: 上升沿 100: 双沿
19-18	EMST2[1:0]	R	INT2 待机清除请求的激活电平。 00: − 01: 上升沿 10: 下降沿 11: 双沿
17	−	R	以未定义读取。
16	INT2EN	R/W	INT2 清除输入 0: 禁用 1: 启用
15	−	R	读作 0,

位	比特符号	类型	功能
14-12	EMCG1[2: 0]	R/W	INT1 待机清除请求的激活电平设定。 (101 ~ 111: 禁止设定) 000: “低“电平 001: “高“电平 010: 下降沿 011: 上升沿 100: 双沿
11-10	EMST1[1:0]	R	INT1 待机清除请求的激活电平。 00: − 01: 上升沿 10: 下降沿 11: 双沿
9	−	R	以未定义读取。
8	INT1EN	R/W	INT1 清除输入 0: 禁用 1: 启用
7	−	R	读作 0,
6-4	EMCG0[2:0]	R/W	INT0 待机清除请求的激活电平设定。 (101 ~ 111: 禁止设定) 000: “低“电平 001: “高“电平 010: 下降沿 011: 上升沿 100: 双沿
3-2	EMST0[1:0]	R	INT0 待机清除请求的激活电平。 00: − 01: 上升沿 10: 下降沿 11: 双沿
1	−	R	以未定义读取。
0	INT0EN	R/W	INT0 清除输入 0: 禁用 1: 启用

注1: 只有当<EMCGx[2:0]>针对上升和下降沿都设置为 “100“时<EMSTx>才有效。待机复位所使用的激活电平可参照<EMSTx>进行检查。当中断通过CGICRCG寄存器清除时，<EMSTx>也清除。

注2: 先规定边沿位再规定<INTxEN>位。禁止同时设定它们。

7.6.3.2 CGIMCGB (CG中断模式控制寄存器 B)

	31	30	29	28	27	26	25	24
比特符号	-	EMCG7			EMST7		-	INT7EN
复位后	0	0	1	0	0	0	未定义	0
	23	22	21	20	19	18	17	16
比特符号	-	EMCG6			EMST6		-	INT6EN
复位后	0	0	1	0	0	0	未定义	0
	15	14	13	12	11	10	9	8
比特符号	-	EMCG5			EMST5		-	INT5EN
复位后	0	0	1	0	0	0	未定义	0
	7	6	5	4	3	2	1	0
比特符号	-	EMCG4			EMST4		-	INT4EN
复位后	0	0	1	0	0	0	未定义	0

位	比特符号	类型	功能
31	−	R	读作 0,
30-28	EMCG7[2:0]	R/W	INT7 待机清除请求的激活电平设定。 (101 ~ 111: 禁止设定) 000: “低“电平 001: “高“电平 010: 下降沿 011: 上升沿 100: 双沿
27-26	EMST7[1:0]	R	INT7 待机清除请求的激活电平。 00: − 01: 上升沿 10: 下降沿 11: 双沿
25	−	R	以未定义读取。
24	INT7EN	R/W	INT7归零输入 0: 禁用 1: 启用
23	−	R	读作 0,
22-20	EMCG6[2:0]	R/W	INT6 待机清除请求的激活电平设定。 (101 ~ 111: 禁止设定) 000: “低“电平 001: “高“电平 010: 下降沿 011: 上升沿 100: 双沿
19-18	EMST6[1:0]	R	INT6 待机清除请求的激活电平。 00: − 01: 上升沿 10: 下降沿 11: 双沿
17	−	R	以未定义读取。
16	INT6EN	R/W	INT6 清除输入 0: 禁用 1: 启用
15	−	R	读作 0,
14-12	EMCG5[2:0]	R/W	INT5 待机清除请求的激活电平设定。 (101 ~ 111: 禁止设定) 000: “低“电平 001: “高“电平 010: 下降沿 011: 上升沿 100: 双沿

位	比特符号	类型	功能
11-10	EMST5[1:0]	R	INT5 待机清除请求的激活电平 00: − 01: 上升沿 10: 下降沿 11: 双沿
9	−	R	以未定义读取。
8	INT5EN	R/W	INT5 清除输入 0: 禁用 1: 启用
7	−	R	读作 0,
6-4	EMCG4[2:0]	R/W	INT4 待机清除请求的激活电平设定。 (101 ~ 111: 禁止设定) 000: “低“电平 001: “高“电平 010: 下降沿 011: 上升沿 100: 双沿
3-2	EMST4[1:0]	R	INT4 待机清除请求的激活电平 00: − 01: 上升沿 10: 下降沿 11: 双沿
1	−	R	以未定义读取。
0	INT4EN	R/W	INT4 清除输入 0: 禁用 1: 启用

注1: 只有当<EMCG×[2:0]>针对上升和下降沿都设置为 “100“时<EMST×>才有效。待机复位所使用的激活电平可参照<EMST×>进行检查。当中断通过CGICRCG寄存器清零时，<EMSTx>也清零。

注2: 先规定边沿位再规定<INTxEN>位。禁止对它们进行同时设定。

7.6.3.3 CGIMCGC (CG中断模式控制寄存器C)

	31	30	29	28	27	26	25	24
比特符号	-	EMCGB			EMSTB		-	INTBEN
复位后	0	0	1	0	0	0	未定义	0
	23	22	21	20	19	18	17	16
比特符号	-	EMCGA			EMSTA		-	INTAEN
复位后	0	0	1	0	0	0	未定义	0
	15	14	13	12	11	10	9	8
比特符号	-	EMCG9			EMST9		-	INT9EN
复位后	0	0	1	0	0	0	未定义	0
	7	6	5	4	3	2	1	0
比特符号	-	EMCG8			EMST8		-	INT8EN
复位后	0	0	1	0	0	0	未定义	0

位	比特符号	类型	功能
31	−	R	读作 0,
30-28	EMCGB[2:0]	R/W	INTB 待机清除请求的激活电平设定。 (101 ~ 111: 禁止设定) 000: “低“电平 001: “高“电平 010: 下降沿 011: 上升沿 100: 双沿
27-26	EMSTB[1:0]	R	INTB 待机清除请求的激活电平 00: − 01: 上升沿 10: 下降沿 11: 双沿
25	−	R	以未定义读取。
24	INTBEN	R/W	INTB 清除输入 0:禁用 1:启用
23	−	R	读作 0,
22-20	EMCGA[2:0]	R/W	对INTA 待机清除请求的激活电平设定。 (101 ~ 111:禁止设定) 000: “低“电平 001: “高“电平 010: 下降沿 011: 上升沿 100: 双沿
19-18	EMSTA[1:0]	R	INTA 待机清除请求的激活电平 00: − 01: 上升沿 10: 下降沿 11: 双沿
17	−	R	以未定义读取。
16	INTAEN	R/W	INTA 清除输入 0: 禁用 1: 启用
15	−	R	读作 0,
14-12	EMCG9[2:0]	R/W	INT9 待机清除请求的激活激电平设定。 (101 ~ 111: 禁止设定) 000: “低“电平 001: “高“电平 010: 下降沿 011: 上升沿 100: 双沿

位	比特符号	类型	功能
11-10	EMST9[1:0]	R	INT9 待机清除请求的激活电平 00: − 01: 上升沿 10: 下降沿 11: 双沿
9	−	R	以未定义读取。
8	INT9EN	R/W	INT9 清除输入 0: 禁用 1: 启用
7	−	R	读作 0,
6-4	EMCG8[2:0]	R/W	对INT8 待机清除请求的激活电平设定。 (101 ~ 111: 禁止设定) 000: “低“电平 001: “高“电平 010: 下降沿 011: 上升沿 100: 双沿
3-2	EMST8[1:0]	R	INT8 待机清除请求的激活电平 00: − 01: 上升沿 10: 下降沿 11: 双沿
1	−	R	以未定义读取。
0	INT8EN	R/W	INT8 清除输入 0: 禁用 1: 启用

注1: 只有当<EMCGx[2:0]>针对上升和下降沿都设置为 “100“时<EMSTx>才有效。待机复位所使用的激活电平可参照<EMSTx>进行检查。当中断通过CGICRCG寄存器清除时，<EMSTx>也清除。

注2: 先规定边沿位再规定<INTxEN>位。禁止对它们进行同时设定。

7.6.3.4 CGIMCGD (CG中断模式控制寄存器 D)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	1	0	0	0	未定义	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	1	0	0	0	未定义	0
	15	14	13	12	11	10	9	8
比特符号	-	EMCGD			EMSTD		-	INTDEN
复位后	0	0	1	0	0	0	未定义	0
	7	6	5	4	3	2	1	0
比特符号	-	EMCGC			EMSTC		-	INTCEN
复位后	0	0	1	0	0	0	未定义	0

位	比特符号	类型	功能
31	−	R	读作 0,
30-28	−	R/W	写入可选值。
27-26	−	R	读作 0,
25	−	R	以未定义读入。
24	−	R/W	写作 0。
23	−	R	读作0,
22-20	−	R/W	写入可选值。
19-18	−	R	读作 0,
17	−	R	以未定义读入。
16	−	R/W	写作 0。
15	−	R	读作 0,
14-12	EMCGD[2:0]	R/W	INTD 待机清除请求的激活电平设定。 (101 ~ 111: 禁止设定) 000: “低“电平 001: “高“电平 010: 下降沿 011: 上升沿 100: 双沿
11-10	EMSTD[1:0]	R	INTD 待机清除请求的激活电平 00: − 01: 上升沿 10: 下降沿 11: 双沿
9	−	R	以未定义读取。
8	INTDEN	R/W	INTD 清除輸入 0: 禁用 1: 启用
7	−	R	读作0,
6-4	EMCGC[2:0]	R/W	INTC 待机清除请求的激活电平设定。 (101 ~ 111: 禁止设定) 000: “低“电平 001: “高“电平 010: 下降沿 011: 上升沿 100: 双沿
3-2	EMSTC[1:0]	R	INTC 待机清除请求的激活电平 00: − 01: 上升沿 10: 下降沿 11: 双沿
1	−	R	以未定义读取。

位	比特符号	类型	功能
0	INTCEN	R/W	INTC 清除输入 0: 禁用 1: 启用

注 1: 只有当<EMCGx[2:0]>针对上升和下降沿都设置为 “100“时<EMSTx>才有效。待机复位所使用的激活电平可参照<EMSTx>进行检查。当中断通过CGICRCG寄存器清零时，<EMSTx>也清除。

注 2: 先规定边沿位再规定<INTxEN>位。禁止对它们进行同时设定。

7.6.3.5 CGIMCGE (CG中断模式控制寄存器 E)

	31	30	29	28	27	26	25	24
比特符号	-	EMCGJ			EMSTJ		-	INTJEN
复位后	0	0	1	0	0	0	未定义	0
	23	22	21	20	19	18	17	16
比特符号	-	EMCGI			EMSTI		-	INTIEN
复位后	0	0	1	0		0	未定义	0
	15	14	13	12	11	10	9	8
比特符号	-	EMCGH			EMSTH		-	INTHEN
复位后	0	0	1	0	0	0	未定义	0
	7	6	5	4	3	2	1	0
比特符号	-	EMCGG			EMSTG		-	INTGEN
复位后	0	0	1	0	0	0	未定义	0

位	比特符号	类型	功能
31	−	R	读作 0,
30-28	EMCGJ[2:0]	R/W	INTRMCRX1待机清除请求的激活电平设定。 设定为如下。 011: 上升沿
27-26	EMSTJ[1:0]	R	INTRMCRX1待机清除请求的R激活电平。 00: − 01: 上升沿 10: 下降沿 11: 双沿
25	−	R	以未定义读入。
24	INTJEN	R/W	INTRMCRX1 清除输入 0: 禁用 1:启用
23	−	R	读作0,
22-20	EMCGI[2:0]	R/W	INTRMCRX0 待机清除请求的激活电平设定。 设定为如下。 011: 上升沿
19-18	EMSTI[1:0]	R	INTRMCRX0 待机清除请求的R激活电平。 00: − 01: 上升沿 10: 下降沿 11: 双沿
17	−	R	以未定义读入。
16	INTIEN	R/W	INTRMCRX0 清除输入 0:禁用 1:启用
15	−	R	读作0,
14-12	EMCGH[2:0]	R/W	对INTCECTX 待机清除请求的激活电平设定。 设定为如下。 011: 上升沿
11-10	EMSTH[1:0]	R	INTCECTX 待机清除请求的激活电平。 00: − 01: 上升沿 10: 下降沿 11: 双沿
9	−	R	以未定义读入。
8	INTHEN	R/W	INTCECTX 清除输入 0: 禁用 1: 启用

位	比特符号	类型	功能
7	−	R	读作 0,
6-4	EMCGG[2:0]	R/W	INTCECRX 待机清除请求的激活电平设定。 设定为如下。 011: 上升沿
3-2	EMSTG[1:0]	R	INTCECRX 待机清除请求的激活电平。 00: − 01: 上升沿 10: 下降沿 11: 双沿
1	−	R	以未定义读入。
0	INTGEN	R/W	INTCECRX 清除输入 0: 禁用 1: 启用

注1: 只有当<EMCGx[2:0]>针对上升和下降沿都设置为 “100“时<EMSTx>才有效。待机复位所使用的激活电平可参照<EMSTx>进行检查。当中断通过CGICRCG寄存器清除时，<EMSTx>也清除。

注2: 先规定边沿位再规定<INTxEN>位。禁止对它们进行同时设定。

7.6.3.6 CGIMCGF (CG中断模式控制寄存器 F)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	EMCGL			EMSTL		-	INTLEN
复位后	0	0	1	0	0	0	未定义	0
	7	6	5	4	3	2	1	0
比特符号	-	EMCGK			EMSTK		-	INTKEN
复位后	0	0	1	0	0	0	未定义	0

位	比特符号	类型	功能
31-15	−	R	读作 0,
14-12	EMCGL[2:0]	R/W	INTKWUP 待机清除请求的激活电平设定。 设定为如下。 001: “H“电平
11-10	EMSTL[1:0]	R	INTKWUP 待机清除请求的激活电平 00: − 01: 上升沿 10: 下降沿 11: 双沿
9	−	R	以未定义读入。
8	INTLEN	R/W	INTKWUP 清除输入 0: 禁用 1: 启用
7	−	R	读作 0,
6-4	EMCGK[2:0]	R/W	INTRTC 待机清除请求的激活电平设定 设定为如下。 010: 下降沿
3-2	EMSTK[1:0]	R	INTRTC 待机清除请求的激活电平 00: − 01: 上升沿 10: 下降沿 11: 双沿
	−	R	以未定义读入。
0	INTKEN	R/W	INTRTC 清除输入 0: 禁用 1: 启用

注 1: 只有当<EMCGx[2:0]>针对上升和下降沿都设置为 “100“时<EMSTx>才有效。待机复位所使用的激活电平可参照<EMSTx>进行检查。当中断通过CGICRCG寄存器清除时，<EMSTx>也清除。

注 2: 先规定边沿位再规定<INTxEN>位。禁止对它们进行同时设定。

7.6.3.7 CGICRCG (CG中断请求清除寄存器)

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

位	比特符号	类型	功能		
31-5	−	R	读作 0,		
4-0	ICRCG[4:0]	W	清除中断请求。		
			0_0000: INT0	0_1000: INT8	1_0000: INTCECRX
			0_0001: INT1	0_1001: INT9	1_0001: INTCECTX
			0_0010: INT2	0_1010: INTA	1_0010: INTRMCRX0
			0_0011: INT3	0_1011: INTB	1_0011: INTRMCRX1
			0_0100: INT4	0_1100: INTC	1_0100: INTRTC
			0_0101: INT5	0_1101: INTD	1_0101: INTKWUP
			0_0110: INT6	0_1110: 保留	1_0110 ~ 1_1111: 保留
			0_0111: INT7	0_1111: 保留	
			读作 0。		

7.6.3.8 CGNMIFLG (NMI 标志寄存器)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2。	1	0
比特符号	-	-	-	-	-	-	NMIFLG1	NMIFLG0
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-2	−	R	读作 0,
1	NMIFLG1	R	NMI 源的产生标记 0: 不适用 1: 产生于 $\overline{\text{NMI}}$ 引脚。
0	NMIFLG0	R	NMI 源的 产生标记 0: 不适用 1: 自WDT生成

注:<NMIFLG>当被读取时清 “0“。

7.6.3.9 CGRSTFLG (复位标志寄存器)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
引脚复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
引脚复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
引脚复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	-	-	-	SYSRSTF	BUPRSTF	WDTRSTF	PINRSTF	PONRSTF
引脚复位后	0	0	0	0	0	0	1	1

位	比特符号	类型	功能
31-5	−	R	读作 0,
4	SYSRSTF	R/W	调试复位标志 (注1) 0: 写入 “0“ 1: 自SYSRESETREQ复位
3	BUPRSTF	R/W	BACKUP 复位标志 0: 写入 “0“ 1: 自BACKUP模式释放复位
2	WDTRSTF	R/W	WDT 复位标志 0: 写入 “0“ 1: 自WDT复位
1	PINRSTF	R/W	$\overline{\text{RESET}}$ 引脚标志 0: 写入 “0“ 1: 自 $\overline{\text{RESET}}$ 引脚复位
0	PONRSTF	R/W	通电标志 0: 写入 “0“ 1: 自通电复位复位

注 1: 该标志表示一通过CPU的NVIC的应用中断SYSRESETREQ位元复位控制寄存器生成的复位。

注 2: 此寄存器产品带有上电复位电路，并且只能通过上电复位才能初始化。因此，在上电后初始复位状态下即将 “1“设置到被设置到<PONRSTF>位。注意该位不会通过第二次后后续复位动作进行设置，寄存器不会自动清零。写入 “0“寄存器清零。

8. 输入 / 输出端口

8.1 端口功能

8.1.1 功能列表

TMPM364F10FG 有 118 个端口。除端口功能外，这些端口还可作为 I/O 引脚用于外围功能。

表 8-1 端口功能表。

表 8-1 端口功能列表

端口	引脚	输入/输出	可编程上拉下拉	史密特输入	噪声过滤器	可编程开漏	功能引脚
端口 A							
	PA0	I/O	上拉	-	-	o	D0 / AD0
	PA1	I/O	上拉	-	-	o	D1 / AD1
	PA2	I/O	上拉	-	-	o	D2 / AD2
	PA3	I/O	上拉	-	-	o	D3 / AD3
	PA4	I/O	上拉	-	-	o	D4 / AD4
	PA5	I/O	上拉	-	-	o	D5 / AD5
	PA6	I/O	上拉	-	-	o	D6 / AD6
	PA7	I/O	上拉	-	-	o	D7 / AD7
端口 B							
	PB0	I/O	上拉	-	-	o	D8 / AD8
	PB1	I/O	上拉	-	-	o	D9 / AD9
	PB2	I/O	上拉	-	-	o	D10 / AD10
	PB3	I/O	上拉	-	-	o	D11 / AD11
	PB4	I/O	上拉	-	-	o	D12 / AD12
	PB5	I/O	上拉	-	-	o	D13 / AD13
	PB6	I/O	上拉	-	-	o	D14 / AD14
	PB7	I/O	上拉	-	-	o	D15 / AD15
端口 C							
	PC0	I/O	上拉	o	-	o	A1，TXD8
	PC1	I/O	上拉	o	-	o	A2，RXD8
	PC2	I/O	上拉	o	-	o	A3，SCLK8，$\overline{\text{CTS8}}$
	PC3	I/O	上拉	o	-	o	A4
	PC4	I/O	上拉	o	-	o	A5，TXD9
	PC5	I/O	上拉	o	-	o	A6，RXD9
	PC6	I/O	上拉	o	-	o	A7，SCLK9，$\overline{\text{CTS9}}$
	PC7	I/O	上拉	o	-	o	A8
端口 D							
	PD0	I/O	上拉	o	-	o	A9，TXD10
	PD1	I/O	上拉	o	-	o	A10，RXD10
	PD2	I/O	上拉	o	-	o	A11，SCLK10，$\overline{\text{CTS10}}$
	PD3	I/O	上拉	o	-	o	A12
	PD4	I/O	上拉	o	-	o	A13，TXD11

表 8-1 端口功能列表

端口	引脚	输入/输出	可编程上拉下拉	史密特输入	噪声过滤器	可编程开漏	功能引脚
	PD5	I/O	上拉	o	–	o	A14，RXD11
	PD6	I/O	上拉	o	–	o	A15，SCLK11，$\overline{\text{CTS11}}$
	PD7	I/O	上拉	o	o	o	A16，INTB
端口 E							
	PE0	I/O	上拉	o	–	o	A17，TB5IN0
	PE1	I/O	上拉	o	–	o	A18，TB5IN1
	PE2	I/O	上拉	o	–	o	A19，TB6IN0
	PE3	I/O	上拉	o	–	o	A20，TB6IN1
	PE4	I/O	上拉	o	–	o	A21，TXD0，CTXD
	PE5	I/O	上拉	o	–	o	A22，RXD0，CRXD
	PE6	I/O	上拉	o	–	o	A23，SCLK0，$\overline{\text{CTS0}}$
	PE7	I/O	上拉	o	o	o	INT5，SCOUT
端口 F							
	PF0	I/O	上拉	o	–	o	TRACECLK
	PF1	I/O	上拉	o	–	o	TRACEDATA0，SWV
	PF2	I/O	上拉	o	–	o	TRACEDATA1
	PF3	I/O	上拉	o	–	o	TRACEDATA2
	PF4	I/O	上拉	o	–	o	TRACEDATA3
端口 G							
	PG0	I/O	上拉	o	–	o	SDA1/ SO1，TB7IN0
	PG1	I/O	上拉	o	–	o	SCL1/ SI1，TB7IN1
	PG2	I/O	上拉	o	–	o	SCK1，$\overline{\text{CS0}}$
	PG3	I/O	上拉	o	o	o	INT6，$\overline{\text{CS1}}$
	PG4	I/O	上拉	o	–	o	SDA2/ SO2，TB9IN0
	PG5	I/O	上拉	o	–	o	SCL2/ SI2，TB9IN1
	PG6	I/O	上拉	o	–	o	SCK2，$\overline{\text{CS3}}$，USBPON
	PG7	I/O	上拉	o	o	o	INT7，$\overline{\text{WDTOUT}}$，$\overline{\text{USBOC}}$
端口 H							
	PH0	I/O	上拉	o	–	o	SDA3/ SO3，TBAIN0
	PH1	I/O	上拉	o	–	o	SCL3/ SI3，TBAIN1
	PH2	I/O	上拉	o	–	o	SCK3，TBBIN0
	PH3	I/O	上拉	o	o	o	INTC，TBBIN1
	PH4	I/O	上拉	o	–	o	SDA4/ SO4，TBDIN0
	PH5	I/O	上拉	o	–	o	SCL4/ SI4，TBDIN1
	PH6	I/O	上拉	o	–	o	SCK4，TBEIN0
	PH7	I/O	上拉	o	o	o	INTD，TBEIN1
端口 I							
	PI0	I/O	上拉	o	–	o	$\overline{\text{BOOT}}$
	PI1	I/O	–	o	–	o (注 3)	CEC
端口 J							
	PJ0	输入	上拉	o	–	–	AIN0
	PJ1	输入	上拉	o	–	–	AIN1
	PJ2	输入	上拉	o	–	–	AIN2
	PJ3	输入	上拉	o	–	–	AIN3，$\overline{\text{ADTRG}}$

表 8-1 端口功能列表

端口	引脚	输入/输出	可编程上拉下拉	史密特输入	噪声过滤器	可编程开漏	功能引脚
	PJ4	输入	上拉	o	o	-	AIN4，KWUP0
	PJ5	输入	上拉	o	o	-	AIN5，KWUP1
	PJ6	输入	上拉	o	o	-	AIN6，KWUP2
	PJ7	输入	上拉	o	o	-	AIN7，KWUP3
端口 K							
	PK0	输入	上拉	o	-	-	AIN8
	PK1	输入	上拉	o	-	-	AIN9
	PK2	输入	上拉	o	-	-	AIN10
	PK3	输入	上拉	o	-	-	AIN11
	PK4	输入	上拉	o	-	-	AIN12
	PK5	输入	上拉	o	-	-	AIN13
	PK6	输入	上拉	o	-	-	AIN14
	PK7	输入	上拉	o	-	-	AIN15
端口 L							
	PL0	I/O	上拉	o	-	o	SDA0/ SO0，TB0OUT
	PL1	I/O	上拉	o	-	o	SCL0/ SI0，TB1OUT
	PL2	I/O	上拉	o	-	o	SCK0，TB2OUT
	PL3	I/O	上拉	o	o	o	INT0，TB3OUT
	PL4	I/O	上拉	o	-	o	TXD1，TB4OUT
	PL5	I/O	上拉	o	-	o	RXD1，TB5OUT
	PL6	I/O	上拉	o	-	o	SCLK1，TB6OUT，$\overline{\text{CTS1}}$
	PL7	I/O	上拉	o	o	o	INT1，TB7OUT
端口 M							
	PM0	I/O	上拉	o	-	o	SCLK2，TB1IN0，$\overline{\text{CTS2}}$
	PM1	I/O	上拉	o	-	o	TXD2，TB1IN1
	PM2	I/O	上拉	o	-	o	RXD2，$\overline{\text{ALARM}}$
	PM3	I/O	上拉	o	o	o	INT2，TB3OUT
	PM4	I/O	上拉	o	-	o	SCLK3，$\overline{\text{CTS3}}$
	PM5	I/O	上拉	o	-	o	TXD3
	PM6	I/O	上拉	o	-	o	RXD3
	PM7	I/O	上拉	o	o	o	INT3
端口 N							
	PN0	I/O	上拉	o	-	o	TXD4
	PN1	I/O	上拉	o	-	o	RXD4
	PN2	I/O	上拉	o	-	o	SCLK4，TB2IN0，$\overline{\text{CTS4}}$
	PN3	I/O	上拉	o	o	o	INT4，TB2IN1，RMC0
	PN4	I/O	上拉	o	-	o	TXD5
	PN5	I/O	上拉	o	-	o	RXD5
	PN6	I/O	上拉	o	-	o	SCLK5，TBFIN0，$\overline{\text{CTS5}}$
	PN7	I/O	上拉	o	o	o	INT8，TBFIN1，RMC1
端口 O							
	PO0	I/O	上拉	o	-	o	TXD6，TB8OUT
	PO1	I/O	上拉	o	-	o	RXD6，TB9OUT
	PO2	I/O	上拉	o	-	o	SCLK6，TBAOUT，$\overline{\text{CTS6}}$

表 8-1 端口功能列表

端口	引脚	输入/输出	可编程上拉下拉	史密特输入	噪声过滤器	可编程开漏	功能引脚
	PO3	I/O	上拉	o	o	o	INT9，TBBOUT
	PO4	I/O	上拉	o	-	o	TXD7，TBCOUT
	PO5	I/O	上拉	o	-	o	RXD7，TBDOUT
	PO6	I/O	上拉	o	-	o	SCLK7，TBEOUT，$\overline{\text{CTS7}}$
	PO7	I/O	上拉	o	o	o	INTA，TBFOUT
端口 P							
	PP0	I/O	上拉	o	-	o	$\overline{\text{CS2}}$
	PP1	I/O	上拉	-	-	o	-
	PP2	I/O	上拉	o	-	o	$\overline{\text{BLS0}}$，SPDO
	PP3	I/O	上拉	-	-	o	$\overline{\text{BLS1}}$，SPDI
	PP4	I/O	上拉	-	-	o	$\overline{\text{WE}}$，SPCLK
	PP5	I/O	上拉	-	-	o	$\overline{\text{OE}}$，SPFSS
	PP6	I/O	上拉	o	-	o	$\overline{\text{ALE}}$

o: 存在

-: 不存在

注 1: 典型条件下噪声过滤器的噪声消除范围大约为 30 ns。

注 2: 端口总是被上拉，尽管有 PIPUP。

注 3: N-ch 开漏端口

8.1.2 端口寄存器概括

使用端口时，需要对以下寄存器进行配置。

- PxDATA: 端口×数据寄存器

  来读取 / 写入端口数据。

- PxCR: 端口×输出控制寄存器

  控制输出。

  使用控制输入时，P×IE 需要进行配置。

- P×FRn: 端口×功能寄存器 n

  设置功能。

  配置功能可通过设置 “1”来激活。

- P×OD: 端口×开漏控制寄存器

  控制可编程的开漏。

  可编程开漏通过设定 P×OD 实现伪开漏界面技术的功能。

  当 P×OD 设定为 “1“时，输出缓冲器失效，实现伪开漏界面技术。

- P×PUP: 端口×上拉控制寄存器

  控制可编程上拉

- P×PDN: 端口×下拉控制寄存器

  控制可编程下拉。

- P×IE: 端口×输入控制寄存器

  控制输入。

  为避开直通电流，默认设置禁止输入。

8.1.3 STOP模式下端口状态

STOP 模式下的输入和输出通过 CGSTBYCR<DRVE>启用/禁用。

当 P×IE 或 P×CR 通过<DRVE>=1 激活时，STOP 模式下的输入或输出分别被启用。当<DRVE>=0 时，STOP 模式下的输入和输出被禁用，即使 P×IE 或 P×CR 启用，某些端口也不会被启用。

表 8-2 STOP 模式下引脚状态。

表 8-2 STOP 模式下端口状态

	引脚名称	I/O	<DRVE> = 0	<DRVE> = 1
不含端口	X1，XT1	仅输入	×	×
	X2，XT2	仅输出	“高“电平输出	“高“电平输出
	$\overline{\text{RESET}}$，$\overline{\text{NMI}}$，MODE	仅输入	O	O
端口	PL3，PL7，PM3，PM7，PN3，PE7，PG3，PG7，PN7，PO3，PO7，PD7，PH3，PH7 (当用于中断 (P×FRn<P×mFn>=1) 及输入被启用 (P×IE<P×mIE>=1)时) (注)	输入	O	O
		输出	×	取决于 P×CR[m]
	PJ4，PJ5，PJ6，PJ7 (当用于 KWUP (P×FRn<P×mFn>=1) 及输入被激活 (P×IE<P×mIE>=1)时) (注)	输入	O	O
		输出	×	取决于 P×CR[m]
	PF0， PF1， PF2， PF3， PF4(当用于跟踪数据输出 (P×FRn<P×mFn>=1)时) (注)	输入	×	取决于 P×CR[m]
		输出	取决于 P×CR[m]	
	PA7-PA0，PB7-PB0，PC7-PC0，PD7-PD0，PE6-PE0，PP6-PP2，PP0 (当用于外部总线 (P×FRn<P×mFn>=1)及输入针对数据总线被启用 (P×IE<P×mIE>=1)时) (注)	输入	O	O
		输出	取决于 P×CR[m]	
	其它端口	输入	×	取决于 P×CR[m]
		输出	×	取决于 P×CR[m]

o：输入或输出启用

×：输入或输出禁用

注: “×“表示一端口数字，“m“表示对应位，“n“表示功能寄存器号。

8.2 端口功能

本章描述了端口寄存器详细情况。

本章仅描述 “电路类型“读取电路的配置。详细的电路图见“8.3 端口方块图“。

8.2.1 端口 A (PA0 ~ PA7)

端口 A 是一个多用途，8-位输入 / 输出端口。该端口的输入 / 输出位为单位加以规定。除多功能输入/输出外，端口 A 还起到外部总线接口的作用。

复位将端口 A 的所有位元初始化为多功能端口，输入，输出和上拉均禁用。

当此端口作为外部总线使用时，PACR，PAFR1 和 PAIE 必须设置到 “1“。

端口 A 具有一个类型功能寄存器。如果端口 A 用作多用途端口，则将寄存器的相应位设为 “0“。如果端口 A 用作其它端口使用，则将功能寄存器对应位设为 “1“。

8.2.1.1 端口 A 电路类型

	7	6	5	4	3	2	1	0
类型	T1	T1	T1	T1	T1	T1	T1	T1

8.2.1.2 端口 A 寄存器

基址= 0×400C_0000

寄存器名称		地址 (基址+)
端口 A 数据寄存器	PADATA	0×0000
端口 A 输出控制寄存器	PACR	0×0004
端口 A 功能寄存器 1	PAFR1	0×0008
端口 A 开漏控制寄存器	PAOD	0×0028
端口 A 上拉控制寄存器	PAPUP	0×002C
端口 A 输入控制寄存器	PAIE	0×0038

8.2.1.3 PADATA (端口 A 数据寄存器)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	PA7	PA6	PA5	PA4	PA3	PA2	PA1	PA0
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-8	−	R	读作 “0“。
7-0	PA7 ~ PA0	R/W	端口 A 数据寄存器

8.2.1.4 PACR (端口 A 输出控制寄存器)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2。	1	0
比特符号	PA7C	PA6C	PA5C	PA4C	PA3C	PA2C	PA1C	PA0C
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-8	−	R	读作 “0“。
7-0	PA7C ~ PA0C	R/W	输出 0: 禁用 1: 启用

8.2.1.5 PAFR1 (端口 A 功能寄存器 1)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	PA7F1	PA6F1	PA5F1	PA4F1	PA3F1	PA2F1	PA1F1	PA0F1
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-8	–	R	读作 “0“。
7	PA7F1	R/W	0: 端口 1: D7，AD7
6	PA6F1	R/W	0:端口 1: D6，AD6
5	PA5F1	R/W	0: 端口 1: D5，AD5
4	PA4F1	R/W	0: 端口 1: D4，AD4
3	PA3F1	R/W	0: 端口 1: D3，AD3
2	PA2F1	R/W	0: 端口 1: D2，AD2
1	PA1F1	R/W	0: 端口 1: D1，AD1
0	PA0F1	R/W	0: 端口 1: D0，AD0

8.2.1.6 PAOD (端口A开漏控制寄存器)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	PA7OD	PA6OD	PA5OD	PA4OD	PA3OD	PA2OD	PA1OD	PA0OD
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-8	−	R	读作 “0“。
7-0	PA7OD ~ PA0OD	R/W	0: CMOS 1: 开漏

8.2.1.7 PAPUP (端口 A 上拉控制寄存器)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	PA7UP	PA6UP	PA5UP	PA4UP	PA3UP	PA2UP	PA1UP	PA0UP
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-8	−	R	读作 “0“。
7-0	PA7UP ~ PA0UP	R/W	上拉 0: 禁用 1: 启用

8.2.1.8 PAIE (端口 A 输入控制寄存器)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	PA7IE	PA6IE	PA5IE	PA4IE	PA3IE	PA2IE	PA1IE	PA0IE
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-8	−	R	读作 “0“。
7-0	PA7IE ~ PA0IE	R/W	输入 0: 禁用 1: 启用

8.2.2 端口 B (PB0 ~ PB7)

端口 B 是一个多用途，8-位输入 / 输出端口。该端口的输入 / 输出位为单位加以规定。除多功能输入 / 输出外，端口 B 还起到外部总线接口的作用。

复位将端口 B 的所有位初始化为多功能端口，输入，输出和上拉均禁用。

端口 B 具有多功能寄存器。如果端口 B 用作多用途端口，则将寄存器的相应位设为 “0“。如果端口 B 用作其它端口使用，则将功能寄存器的相应位设为 “1“。

8.2.2.1 端口B电路类型

	7	6	5	4	3	2	1	0
类型	T1	T1	T1	T1	T1	T1	T1	T1

8.2.2.2 端口B寄存器

基址= 0×400C_0100

寄存器名称		地址 (基址+)
端口 B 数据寄存器	PBDATA	0×0000
端口 B 输出控制寄存器	PBCR	0×0004
端口 B 功能寄存器 1	PBFR1	0×0008
端口 B 开漏控制寄存器	PBOD	0×0028
端口 B 上拉控制寄存器	PBPUP	0×002C
端口 B 输入控制寄存器	PBIE	0×0038

8.2.2.3 PBDATA (端口 B 数据寄存器)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	PB7	PB6	PB5	PB4	PB3	PB2	PB1	PB0
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-8	−	R	读作 “0“。
7-0	PB7 ~ PB0	R/W	端口 B 数据寄存器

8.2.2.4 PBCR (端口 B 输出控制寄存器)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	PB7C	PB6C	PB5C	PB4C	PB3C	PB2C	PB1C	PB0C
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-8	−	R	读作 “0“。
7-0	PB7C ~ PB0C	R/W	输出 0: 禁用 1: 启用

8.2.2.5 PBFR1 (端口B功能寄存器)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	PB7F1	PB6F1	PB5F1	PB4F1	PB3F1	PB2F1	PB1F1	PB0F1
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-8	–	R	读作 “0“。
7	PB7F1	R/W	0: 端口 1:D15，AD15
6	PB6F1	R/W	0: 端口 1:D14，AD14
5	PB5F1	R/W	0: 端口 1:D13，AD13
4	PB4F1	R/W	0: 端口 1:D12，AD12
3	PB3F1	R/W	0: 端口 1:D11，AD11
2	PB2F1	R/W	0: 端口 1:D10，AD10
1	PB1F1	R/W	0: 端口 1:D9，AD9
0	PB0F1	R/W	0: 端口 1:D8，AD8

8.2.2.6 PBOD (端口 B 开漏控制寄存器)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	PB7OD	PB6OD	PB5OD	PB4OD	PB3OD	PB2OD	PB1OD	PB0OD
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-8	−	R	读作 “0“。
7-0	PB7OD ~ PB0OD	R/W	0: CMOS 1: 开漏

8.2.2.7 PBPUP (端口 B 上拉控制寄存器)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	PB7UP	PB6UP	PB5UP	PB4UP	PB3UP	PB2UP	PB1UP	PB0UP
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-8	−	R	读作 “0“。
7-0	PB7UP ~ PB0UP	R/W	上拉 0: 禁用 1: 启用

8.2.2.8 PBIE (端口 B 输入控制寄存器)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	PB7IE	PB6IE	PB5IE	PB4IE	PB3IE	PB2IE	PB1IE	PB0IE
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-8	−	R	读作 “0“。
7-0	PB7IE ~ PB0IE	R/W	输入 0: 禁用 1: 启用

8.2.3 端口C (PC0 ~ PC7)

端口 C 是一个多用途，8-位输入/输出端口。该端口的输入 / 输出位为单位加以规定。除多功能输入 / 输出外，端口 C 还起到外部总线接口，串行通道，以及串行总线接口的作用。

复位将端口 C 的所有位初始化为多功能端口，输入，输出和上拉均禁用。

端口 C 有三种类型功能寄存器。如果端口 C 用作多用途端口，则将三种寄存器的相应位设为 "0"。如果端口C用作其它端口使用,则将功能寄存器的相应位设为 "1" 。不能将某些功能寄存器同时设为 "1"。

8.2.3.1 端口 C 电路类型

	7	6	5	4	3	2	1	0
类型	T5	T4	T3	T2	T5	T4	T3	T2

8.2.3.2 端口 C 寄存器

基址 = 0×400C_0200

寄存器名称		地址 (基址+)
端口 C 数据寄存器	PCDATA	0×0000
端口 C 输出控制寄存器	PCCR	0×0004
端口 C 功能寄存器 1	PCFR1	0×0008
端口 C 功能寄存器 2	PCFR2	0×000C
端口 C 功能寄存器 3	PCFR3	0×0010
端口 C 开漏控制寄存器	PCOD	0×0028
端口 C 上拉控制寄存器	PCPUP	0×002C
端口 C 输入控制寄存器	PCIE	0×0038

8.2.3.3 PCDATA (端口C数据寄存器)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	PC7	PC6	PC5	PC4	PC3	PC2	PC1	PC0
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-8	−	R	读作 “0“。
7-0	PC7 ~ PC0	R/W	端口 C 数据寄存器

8.2.3.4 PCCR (端口 C 输出控制寄存器)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	PC7C	PC6C	PC5C	PC4C	PC3C	PC2C	PC1C	PC0C
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-8	−	R	读作 “0“。
7-0	PC7C ~ PC0C	R/W	输出 0: 禁用 1: 启用

8.2.3.5 PCFR1 (端口 C 功能寄存器 1)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	PC7F1	PC6F1	PC5F1	PC4F1	PC3F1	PC2F1	PC1F1	PC0F1
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-8	−	R	读作 “0“。
7	PC7F1	R/W	0: 端口 1: A8
6	PC6F1	R/W	0: 端口 1: A7
5	PC5F1	R/W	0: 端口 1: A6
4	PC4F1	R/W	0: 端口 1: A5
3	PC3F1	R/W	0: 端口 1: A4
2	PC2F1	R/W	0: 端口 1: A3
1	PC1F1	R/W	0: 端口 1: A2
0	PC0F1	R/W	0: 端口 1: A1

8.2.3.6 PCFR2 (端口 C 功能寄存器 2)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	-	PC6F2	PC5F2	PC4F2	-	PC2F2	PC1F2	PC0F2
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-7	−	R	读作 “0“。
6	PC6F2	R/W	0: 端口 1: SCLK9
5	PC5F2	R/W	0: 端口 1: RXD9
4	PC4F2	R/W	0: 端口 1: TXD9
3	−	R	读作 “0“。
2	PC2F2	R/W	0: 端口 1: SCLK8
1	PC1F2	R/W	0: 端口 1: RXD8
0	PC0F2	R/W	0: 端口 1: TXD8

8.2.3.7 PCFR3 (端口 C 功能寄存器 3)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	-	PC6F3	-	-	-	PC2F3	-	-
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-7	−	R	读作 “0“。
6	PC6F3	R/W	0: 端口 1: $\overline{CTS9}$
5-3	−	R	读作 “0“。
2	PC2F3	R/W	0: 端口 1: $\overline{CTS8}$
1-0	−	R	读作 “0“。

8.2.3.8 PCOD (端口 C 开漏控制寄存器)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	PC7OD	PC6OD	PC5OD	PC4OD	PC3OD	PC2OD	PC1OD	PC0OD
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-8	−	R	读作 “0“。
7-0	PC7OD ~ PC0OD	R/W	0 : CMOS 1 : 开漏

8.2.3.9 PCPUP (端口 C 上拉控制寄存器)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	PC7UP	PC6UP	PC5UP	PC4UP	PC3UP	PC2UP	PC1UP	PC0UP
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-8	−	R	读作 “0“。
7-0	PC7UP ~ PC0UP	R/W	上拉 0: 禁用 1: 启用

8.2.3.10 PCIE (端口 C 输入控制寄存器)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	PC7IE	PC6IE	PC5IE	PC4IE	PC3IE	PC2IE	PC1IE	PC0IE
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-8	−	R	读作 “0“。
7-0	PC7IE ~ PC0IE	R/W	输入 0: 禁用 1: 启用

8.2.4 端口 D (PD0 ~ PD7)

端口 D 是一个多用途，8-位输入/输出端口。该端口的输入 / 输出位为单位加以规定。除多功能输入 / 输出外，端口 D 还起到外部总线接口，串行通道和外部信号中断的功能。

复位将端口 D 的所有位初始化多功能端口，输入，输出和上拉均禁用。

端口 D 有三种类型功能寄存器。如果端口 D 用作多用途端口，则将三种寄存器的相应位设为 “0“ 。如果端口 D 用作其它端口使用，则将功能寄存器的相应位设为 “1“。不能将某些些功能寄存器同时设为 “1“。

8.2.4.1 端口D电路类型

	7	6	5	4	3	2	1	0
类型	T6	T4	T3	T2	T5	T4	T3	T2

8.2.4.2 端口寄存器

基址 = 0×400C_0300

寄存器名称		地址 (基址+)
端口 D 数据寄存器	PDDATA	0×0000
端口 D 输出控制寄存器	PDCR	0×0004
端口 D 功能寄存器 1	PDFR1	0×0008
端口 D 功能寄存器 2	PDFR2	0×000C
端口 D 功能寄存器 3	PDFR3	0×0010
端口 D 开漏控制寄存器	PDOD	0×0028
端口 D 上拉控制寄存器	PDPUP	0×002C
端口 D 输入控制寄存器	PDIE	0×0038

8.2.4.3 PDDATA (端口 D 数据寄存器)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	PD7	PD6	PD5	PD4	PD3	PD2	PD1	PD0
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-8	−	R	读作 “0“。
7-0	PD7 ~ PD0	R/W	端口 D 数据寄存器

8.2.4.4 PDCR (端口 D 输出控制寄存器)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	PD7C	PD6C	PD5C	PD4C	PD3C	PD2C	PD1C	PD0C
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-8	−	R	读作 “0“。
7-0	PD7C ~ PD0C	R/W	输出 0: 禁用 1: 启用

8.2.4.5 PDFR1 (端口 D 功能寄存器 1)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	PD7F1	PD6F1	PD5F1	PD4F1	PD3F1	PD2F1	PD1F1	PD0F1
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-8	−	R	读作 “0“。
7	PD7F1	R/W	0: 端口 1: A16
6	PD6F1	R/W	0: 端口 1: A15
5	PD5F1	R/W	0: 端口 1: A14
4	PD4F1	R/W	0: 端口 1: A13
3	PD3F1	R/W	0: 端口 1: A12
2	PD2F1	R/W	0: 端口 1: A11
1	PD1F1	R/W	0: 端口 1: A10
0	PD0F1	R/W	0: 端口 1: A9

8.2.4.6 PDFR2 (端口 D 功能寄存器 2)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	PD7F2	PD6F2	PD5F2	PD4F2	-	PD2F2	PD1F2	PD0F2
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-8	−	R	读作 “0“。
7	PD7F2	R/W	0: 端口 1: INTB
6	PD6F2	R/W	0: 端口 1: SCLK11
5	PD5F2	R/W	0: 端口 1: RXD11
4	PD4F2	R/W	0: 端口 1: TXD11
3	−	R	读作 “0“。
2	PD2F2	R/W	0: 端口 1: SCLK10
1	PD1F2	R/W	0: 端口 1: RXD10
0	PD0F2	R/W	0 :端口 1: TXD10

8.2.4.7 PDFR3 (端口 D 功能寄存器 3)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	-	PD6F3	-	-	-	PD2F3	-	-
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-7	−	R	读作 “0“。
6	PD6F3	R/W	0: 端口 1: $\overline{\text{CTS11}}$
5-3	−	R	读作 “0“。
2	PD2F3	R/W	0: 端口 1: $\overline{\text{CTS10}}$
1-0	−	R	读作 “0“。

8.2.4.8 PDOD (端口 D 开漏控制寄存器)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	PD7OD	PD6OD	PD5OD	PD4OD	PD3OD	PD2OD	PD1OD	PD0OD
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-8	−	R	读作 “0“。
7-0	PD7OD ~ PD0OD	R/W	0: CMOS 1: 开漏

8.2.4.9 PDPUP (端口 D 上拉控制寄存器)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	PD7UP	PD6UP	PD5UP	PD4UP	PD3UP	PD2UP	PD1UP	PD0UP
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-8	−	R	读作 “0“。
7-0	PD7UP ~ PD0UP	R/W	上拉 0: 禁用 1: 启用

8.2.4.10 PDIE (端口 D 输入控制寄存器)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	PD7IE	PD6IE	PD5IE	PD4IE	PD3IE	PD2IE	PD1IE	PD0IE
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-8	−	R	读作 “0“。
7-0	PD7IE ~ PD0IE	R/W	输入 0: 禁用 1: 启用

8.2.5 端口 E (PE0-PE7)

端口 E 是一个多用途，8-位输入 / 输出端口。该端口的输入 / 输出位为单位加以规定。除多功能输入 / 输出外，端口 E 还起到外部总线接口，串行通道，外部信号中断，16-位计时器/计数器，CAN 控制器功能以及时钟输出的功能。

复位将端口 E 的所有位初始化为多功能端口，输入，输出和上拉均禁用。

端口 E 有三种类型功能寄存器。如果端口 E 用作多用途端口，则将三种寄存器的相应位设为 "0" 。如果端口 E 用作其它端口使用，则将功能寄存器的相应位设为 "1" 。不能将某些功能寄存器同时设为 "1" 。

注:在非 STOP 模式中，如果输入能在 P×IE 中激活，无论 P×FR 寄存器如何设置，则中断输入被激活。为设备编程时，应确定将未使用的中断禁用。

8.2.5.1 端口 E 电路类型

	7	6	5	4	3	2	1	0
类型	T6	T4	T37	T36	T3	T3	T3	T3

8.2.5.2 端口 E 寄存器

基址 = 0×400C_0400

寄存器名称		地址 (基址+)
端口 E 数据寄存器	PEDATA	0×0000
端口 E 输出控制寄存器	PECR	0×0004
端口 E 功能寄存器 1	PEFR1	0×0008
端口 E 功能寄存器 2	PEFR2	0×000C
端口 E 功能寄存器 3	PEFR3	0×0010
端口 E 开漏控制寄存器	PEOD	0×0028
端口 E 上拉控制寄存器	PEPUP	0×002C
端口 E 输入控制寄存器	PEIE	0×0038

8.2.5.3 PEDATA (端口 E 数据寄存器)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	PE7	PE6	PE5	PE4	PE3	PE2	PE1	PE0
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-8	−	R	读入为 “0“。
7-0	PE ~ PE0	R/W	端口 E 数据寄存器。

8.2.5.4 PECR (端口 E 输出控制寄存器)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	PE7C	PE6C	PE5C	PE4C	PE3C	PE2C	PE1C	PE0C
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-8	−	R	读作 “0“。
7-0	PE7C ~ PE0C	R/W	输出 0: 禁用 1: 启用

8.2.5.5 PEFR1 (端口 E 功能寄存器 1)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	-	PE6F1	PE5F1	PE4F1	PE3F1	PE2F1	PE1F1	PE0F1
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-8	−	R	读作 “0“。
7	−	R/W	写入 “0“。
6	PE6F1	R/W	0: 端口 1: A23
5	PE5F1	R/W	0: 端口 1: A22
4	PE4F1	R/W	0: 端口 1: A21
3	PE3F1	R/W	0: 端口 1: A20
2	PE2F1	R/W	0: 端口 1: A19
1	PE1F1	R/W	0: 端口 1: A18
0	PE0F1	R/W	0: 端口 1: A17

8.2.5.6 PEFR 2 (端口 E 功能寄存器 2)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	PE7F2	PE6F2	PE5F2	PE4F2	PE3F2	PE2F2	PE1F2	PE0F2
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-8	−	R	读作 “0“。
7	PE7F2	R/W	0: 端口 1: INT5
6	PE6F2	R/W	0: 端口 1: SCLK0
5	PE5F2	R/W	0: 端口 1: RXD0
4	PE4F2	R/W	0: 端口 1: TXD0
3	PE3F2	R/W	0: 端口 1: TB6IN1
2	PE2F2	R/W	0: 端口 1: TB6IN0
1	PE1F2	R/W	0: 端口 1: TB5IN1
0	PE0F2	R/W	0: 端口 1: TB5IN0

8.2.5.7 PEFR3 (端口 E 功能寄存器 3)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	PE7F3	PE6F3	PE5F3	PE4F3	-	-	-	-
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-8	−	R	读作 “0“。
7	PE7F3	R/W	0 : 端口 1 : SCOUT
6	PE6F3	R/W	0 : 端口 1: $\overline{CTS0}$
5	PE5F3	R/W	0 : 端口 1 : CRXD
4	PE4F3	R/W	0 : 端口 1 : CTXD
3 ~ 0	−	R	读作 “0“。

8.2.5.8 PEOD (端口E开漏控制寄存器)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	PE7OD	PE6OD	PE5OD	PE4OD	PE3OD	PE2OD	PE1OD	PE0OD
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-8	−	R	读作 “0“。
7-0	PE7OD ~ PE0OD	R/W	0 : CMOS 1 : 开漏

8.2.5.9 PEPUP (端口 E 上拉控制寄存器)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	PE7UP	PE6UP	PE5UP	PE4UP	PE3UP	PE2UP	PE1UP	PE0UP
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-8	−	R	读作 “0“。
7-0	PE7UP ~ PE0UP	R/W	上拉 0：禁用 1：启用

8.2.5.10 PEIE (端口 E 输入控制寄存器)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	PE7IE	PE6IE	PE5IE	PE4IE	PE3IE	PE2IE	PE1IE	PE0IE
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-8	−	R	读作 “0“。
7-0	PE7IE ~ PE0IE	R/W	输入 0：禁用 1：启用

8.2.6 端口 F (PF0 ~ PF4)

端口 F 是一个多用途，5-位输入 / 输出端口。该端口的输入 / 输出位为单位加以规定。除多功能输入 / 输出外，端口 F 还起到调试界面的功能。

复位将端口 F 的所有位初始化为多功能端口，输入，输出和上拉均禁用。

端口 D 有一种类型功能寄存器。如果端口 F 用作多用途端口，则将三种寄存器的相应位设为 "0"。如果端口 F 用作其它端口使用，则将功能寄存器的相应位设为 "1"。

8.2.6.1 端口F电路类型

	7	6	5	4	3	2	1	0
类型	-	-	-	T7	T7	T7	T7	T7

8.2.6.2 端口 F 寄存器

基址 = 0×400C_0500

寄存器名称		地址 (基址+)
端口 F 数据寄存器	PFDATA	0×0000
端口 F 输出控制寄存器	PFCR	0×0004
端口 F 功能寄存器 1	PFFR1	0×0008
端口 F 开漏控制寄存器	PFOD	0×0028
端口 F 上拉控制寄存器	PFPUP	0×002C
端口 F 输入控制寄存器	PFIE	0×0038

8.2.6.3 PFDATA (端口 F 数据寄存器)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	-	-	-	PF4	PF3	PF2	PF1	PF0
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-5	−	R	读作 “0“。
4-0	PF4 ~ PF0	R/W	端口 F 数据寄存器

8.2.6.4 PFCR (端口 F 输出控制寄存器)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	-	-	-	PF4C	PF3C	PF2C	PF1C	PF0C
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-5	−	R	读作 “0“。
4-0	PF4C ~ PF0C	R/W	输出 0：禁用 1：启用

8.2.6.5 PFFR1 (端口 F 功能寄存器 1)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	-	-	-	PF4F1	PF3F1	PF2F1	PF1F1	PF0F1
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-5	−	R	读作 “0“。
4	PF4F1	R/W	0：端口 1: TRACEDATA3
3	PF3F1	R/W	0：端口 1 : TRACEDATA2
2	PF2F1	R/W	0：端口 1 : TRACEDATA1
1	PF1F1	R/W	0：端口 1 : TRACEDATA0 / SWV
0	PF0F1	R/W	0：端口 1 : TRACECLK

8.2.6.6 PFOD (端口 F 开漏控制寄存器)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	-	-	-	PF4OD	PF3OD	PF2OD	PF1OD	PF0OD
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-5	−	R	读作 “0“。
4-0	PF4OD ~ PF0OD	R/W	0 : CMOS 1 : 开漏

8.2.6.7 PFPUP (端口 F 上拉控制寄存器)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	-	-	-	PF4UP	PF3UP	PF2UP	PF1UP	PF0UP
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-5	−	R	读作 “0“。
4-0	PF4UP ~ PF0UP	R/W	上拉 0 : 禁用 1 : 启用

8.2.6.8 PFIE (端口 F 输入控制寄存器)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	-	-	-	PF4IE	PF3IE	PF2IE	PF1IE	PF0IE
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-5	−	R	读作 “0“。
4-0	PF4IE ~ PF0IE	R/W	输入 0：禁用 1：启用

8.2.7 端口G (PG0 ~ PG7)

端口 D 是一个多用途，8-位输入/输出端口。该端口的输入 / 输出位为单位加以规定。除多功能输入 / 输出外，端口 G 还起到串行总线接口，外部信号中断，16-位计时器/事件计数器，外部总线接口，USB 主控制器功能以及看门狗计时器输出的功能。

复位将端口 G 的所有位初始化为多功能端口，输入，输出和上拉均禁用。

端口 G 有三种类型功能寄存器。如果端口 G 用作多用途端口时，则将三种寄存器的相应位设为 "0"。如果端口 G 用作其它端口使用，则将功能寄存器的相应位设为 "1"。不能将三种功能寄存器同时设为 "1"。

若使用外部中断输入来释放 STOP 停止模式，在 PGFR1 中选择此功能并在 PGIE 寄存器中激活输入。即使在 STOP 模式下，时钟/模式控制板块中的 CGSTBYCR<DRVE> 位被设置为停止驱动引脚，这些设置将激活中断输入。

注:在非 STOP 模式中，如果输入能在 P×IE 中激活，无论 P×FR 寄存器如何设置，则中断输入被激活。为设备编程时，应确定将未使用的中断禁用。

8.2.7.1 端口 G 电路类型

	7	6	5	4	3	2	1	0
类型	T39	T38	T8	T8	T10	T9	T8	T8

8.2.7.2 端口 G 寄存器

基址 = 0×400C_0600

寄存器名称		地址 (基址+)
端口 G 数据寄存器	PGDATA	0×0000
端口 G 输出控制寄存器	PGCR	0×0004
端口 G 功能寄存器 1	PGFR1	0×0008
端口 G 功能寄存器 2	PGFR2	0×000C
端口 G 功能寄存器 3	PGFR3	0×0010
端口 G 开漏控制寄存器	PGOD	0×0028
端口 G 上拉控制寄存器	PGPUP	0×002C
端口 G 输入控制寄存器	PGIE	0×0038

8.2.7.3 PGDATA (端口 G 数据寄存器)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	PG7	PG6	PG5	PG4	PG3	PG2	PG1	PG0
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-8	−	R	读作 “0“。
7-0	PG7 ~ PG0	R/W	端口 G 数据寄存器

8.2.7.4 PGCR (端口 G 输出控制寄存器)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	PG7C	PG6C	PG5C	PG4C	PG3C	PG2C	PG1C	PG0C
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-8	−	R	读作 “0“。
7-0	PG7C ~ PG0C	R/W	输出 0：禁用 1：启用

8.2.7.5 PGFR1 (端口 G 功能寄存器 1)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	PG7F1	PG6F1	PG5F1	PG4F1	PG3F1	PG2F1	PG1F1	PG0F1
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-8	−	R	读作 “0“。
7	PG7F1	R/W	0 : 端口 1 : INT7
6	PG6F1	R/W	0 : 端口 1:SCK2
5	PG5F1	R/W	0 : 端口 1 : SCL2 / SI2
4	PG4F1	R/W	0 : 端口 1 : SDA2 / SO2
3	PG3F1	R/W	0 : 端口 1 : INT6
2	PG2F1	R/W	0: 端口 1 : SCK1
1	PG1F1	R/W	0 : 端口 1 : SCL1 / SI1
0	PG0F1	R/W	0 : 端口 1 : SDA1 / SO1

8.2.7.6 PGFR2 (端口G功能寄存器2)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	PG7F2	PG6F2	PG5F2	PG4F2	-	-	PG1F2	PG0F2
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-8	−	R	读作 “0“。
7	PG7F2	R/W	0：端口 1：$\overline{\text{USBOC}}$
6	PG6F2	R/W	0：端口 1：USBPON
5	PG5F2	R/W	0：端口 1：TB9IN1
4	PG4F2	R/W	0：端口 1：TB9IN0
3-2	−	R	读作 “0“。
1	PG1F2	R/W	0：端口 1：TB7IN1
0	PG0F2	R/W	0：端口 1：TB7IN0

8.2.7.7 PGFR3 (端口G功能寄存器3)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	PG7F3	PG6F3	-	-	PG3F3	PG2F3	-	-
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-8	−	R	读作 “0“。
7	PG7F3	R/W	0：端口 1：$\overline{\text{WDTOUT}}$
6	PG6F3	R/W	0：端口 1：$\overline{\text{CS3}}$
5-4	−	R	读作 “0“。
3	PG3F3	R/W	0：端口 1：$\overline{\text{CS1}}$
2	PG2F3	R/W	0：端口 1：$\overline{\text{CS0}}$
1 ~ 0	−	R	读作 “0“。

8.2.7.8 PGOD (端口 G 开漏控制寄存器)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	PG7OD	PG6OD	PG5OD	PG4OD	PG3OD	PG2OD	PG1OD	PG0OD
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-8	−	R	读作 “0“。
7-0	PG7OD ~ PG0OD	R/W	0 : CMOS 1 : 开漏

8.2.7.9 PGPUP (端口 G 上拉控制寄存器)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	PG7UP	PG6UP	PG5UP	PG4UP	PG3UP	PG2UP	PG1UP	PG0UP
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-8	−	R	读作 “0“。
7-0	PG7UP ~ PG0UP	R/W	上拉 0 : 禁用 1 : 启用

8.2.7.10 PGIE (端口 G 输入控制寄存器)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	PG7IE	PG6IE	PG5IE	PG4IE	PG3IE	PG2IE	PG1IE	PG0IE
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-8	−	R	读作 “0“。
7-0	PG7IE ~ PG0IE	R/W	输入 0：禁用 1：启用

8.2.8 端口 H (PH0 ~ PH7)

端口 H 是一个多用途，8-位输入 / 输出端口。该端口的输入 / 输出位为单位加以规定。除多功能输入 / 输出外，端口 H 还起到串行总线接口，外部信号中断功能以及 16-位计时器 / 事件计数器的功能。

复位将端口 H 的所有位初始化多功能端口，输入，输出和上拉均禁用。

端口 H 有两种类型功能寄存器。如果端口 H 用作多用途端口，则将这两种寄存器的相应位设为 "0" 。如果端口 H 用作作其它端口使用，则将功能寄存器的相应位设为 "1" 。不能将某些功能寄存器同时设为 "1"。

注:在非 STOP 模式中，如果输入能在 P×IE 中激活，无论 PHFR1 寄存器如何设置，则中断输入被激活。为设备编程时，应确定将未使用的中断禁用。

8.2.8.1 端口 H 电路类型

	7	6	5	4	3	2	1	0
类型	T13	T12	T12	T12	T13	T12	T12	T12

8.2.8.2 端口 H 寄存器

基址 = 0×400C_0700

寄存器名称		地址 (基址+)
端口 H 数据寄存器	PHDATA	0×0000
端口 H 输出控制寄存器	PHCR	0×0004
端口 H 功能寄存器 1	PHFR1	0×0008
端口 H 功能寄存器 2	PHFR2	0×000C
端口 H 开漏控制寄存器	PHOD	0×0028
端口 H 上拉控制寄存器	PHPUP	0×002C
端口 B 输入控制寄存器	PHIE	0×0038

8.2.8.3 PHDATA (端口 H 数据寄存器)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	PH7	PH6	PH5	PH4	PH3	PH2	PH1	PH0
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-8	−	R	读作 “0“。
7-0	PH7 ~ PH0	R/W	端口 H 数据寄存器

8.2.8.4 PHCR (端口 H 输出控制寄存器)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	PH7C	PH6C	PH5C	PH4C	PH3C	PH2C	PH1C	PH0C
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-8	−	R	读作 “0“。
7-0	PH7C ~ PH0C	R/W	输出 0：禁用 1：启用

8.2.8.5 PHFR1 (端口 H 功能寄存器 1)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	PH7F1	PH6F1	PH5F1	PH4F1	PH3F1	PH2F1	PH1F1	PH0F1
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-8	−	R	读作 “0“。
7	PH7F1	R/W	0：端口 1 : INTD
6	PH6F1	R/W	0：端口 1 : SCK4
5	PH5F1	R/W	0：端口 1 : SCL4 / SI4
4	PH4F1	R/W	0：端口 1 : SDA4 / SO4
3	PH3F1	R/W	0：端口 1 : INTC
2	PH2F1	R/W	0：端口 1 : SCK3
1	PH1F1	R/W	0：端口 1 : SCL3 / SI3
0	PH0F1	R/W	0：端口 1 : SDA3 / SO3

8.2.8.6 PHFR2 (端口 H 功能寄存器 2)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	PH7F2	PH6F2	PH5F2	PH4F2	PH3F2	PH2F2	PH1F2	PH0F2
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-8	−	R	读作 “0“。
7	PH7F2	R/W	0：端口 1 : TBEIN1
6	PH6F2	R/W	0：端口 1 : TBEIN0
5	PH5F2	R/W	0：端口 1 : TBDIN1
4	PH4F2	R/W	0：端口 1 : TBDIN0
3	PH3F2	R/W	0：端口 1 : TBBIN1
2	PH2F2	R/W	0：端口 1 : TBBIN0
1	PH1F2	R/W	0：端口 1 : TBAIN1
0	PH0F2	R/W	0：端口 1 : TBAIN0

8.2.8.7 PHOD (端口 H 开漏控制寄存器)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	PH7OD	PH6OD	PH5OD	PH4OD	PH3OD	PH2OD	PH1OD	PH0OD
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-8	−	R	读作 “0“。
7-0	PH7OD ~ PH0OD	R/W	0 : CMOS 1 : 开漏

8.2.8.8 PHPUP (端口 H 上拉控制寄存器)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	PH7UP	PH6UP	PH5UP	PH4UP	PH3UP	PH2UP	PH1UP	PH0UP
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-8	−	R	读作 “0“。
7-0	PH7UP ~ PH0UP	R/W	上拉 0 : 禁用 1 : 启用

8.2.8.9 PHIE (端口 H 输入控制寄存器)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	PH7IE	PH6IE	PH5IE	PH4IE	PH3IE	PH2IE	PH1IE	PH0IE
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-8	−	R	读作 “0“。
7-0	PH7IE ~ PH0IE	R/W	输入 0：禁用 1：启用

8.2.9 端口 I (PI0 ~ PI1)

端口 I 是一个多用途，2-位输入 / 输出端口。该端口的输入 / 输出位为单位加以规定。除多功能端口外，端口 I 还起到 CEC 输入和操作模式设定的作用。

复位将 PI0 初始化为多功能端口，输入，输出和上拉均禁用。复位将 PI1，PI2 和 PI3 初始化为多功能端口，输入，输出均禁用。

PI1 是 N-ch 开漏端口。

端口 I 有一种多功能寄存器。如果端口 I 用作多用途端口，则将三种寄存器的相应位设为 "0"。如果端口 H 用作其它端口使用，则将功能寄存器的相应位设为 "1"。

复位引脚处在"低"位之时，PI0 ($\overline{\text{BOOT}}$)的输入和上拉将被激活。在$\overline{\text{RESET}}$引脚的上升沿，如果 PI0 ($\overline{\text{BOOT}}$)是"高"，则 TMPM364F10FG 将进入单晶片模式并将从片装闪速存储器 (Flash ROM)启动。如果 PI0 ($\overline{\text{BOOT}}$)是"低"，TMPM364F10FG 则将进入单启动模式并从片装 BOOTROM 启动单启动模式详情，见 "闪存操作 "。

8.2.9.1 端口I电路类型

	7	6	5	4	3	2	1	0
类型	–	–	–	–	–	–	T15	T14

8.2.9.2 端口I寄存器

基址 = 0×400C_0800

寄存器名称		地址 (基址+)
端口 I 数据寄存器	PIDATA	0×0000
端口 I 输出控制寄存器	PICR	0×0004
端口 I 功能寄存器 1	PIFR1	0×0008
端口 I 开漏控制寄存器	PIOD	0×0028
端口 I 上拉控制寄存器	PIPUP	0×002C
端口 I 输入控制寄存器	PIIE	0×0038

8.2.9.3 PIDATA (端口 I 数据寄存器)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	-	-	-	-	-	-	PI1	PI0
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-4	−	R	读作 “0“。
3-2	−	R/W	写入 “0“。
1-0	PI1 ~ PI0	R/W	端口 I 数据寄存器

8.2.9.4 PICR (端口 I 输出控制寄存器)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	-	-	-	-	-	-	PI1C	PI0C
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-4	−	R	读作 “0“。
3-2	−	R/W	写入 “0“。
1-0	PI1C-PI0C	R/W	输出 0：禁用 1：启用

8.2.9.5 PIFR1 (端口 I 功能寄存器 1)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	-	-	-	-	-	-	PI1F1	-
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-4	−	R	读作 “0“。
3-2	−	R/W	写入 “0“。
1	PI1F1	R/W	0 : 端口 1 : CEC
0	−	R/W	写入 “0“。

8.2.9.6 PIOD (端口 I 开漏控制寄存器)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	-	-	-	-	-	-	-	PI0OD
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-4	−	R	读作 “0“。
3-2	−	R/W	写入 “0“。
1	−	R	读作 “0“。
0	PI0OD	R/W	0 : CMOS 1 : 开漏

8.2.9.7 PIPUP (端口 I 上拉控制寄存器)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	-	-	-	-	-	-	-	PI0UP
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-1	−	R	读作 “0“。
0	PI0UP	R/W	上拉 0 : 禁用 1 : 启用

8.2.9.8 PIIE (端口 I 输入控制寄存器)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	-	-	-	-	-	-	PI1IE	PI0IE
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-4	−	R	读作 “0“。
3-2	−	R/W	写入 “0“。
1-0	PI1IE-PI0IE	R/W	输入 0：禁用 1：启用

8.2.10 端口 J (PJ0 ~ PJ7)

端口 J 是一个 8-位输入端口。除多功能输入外，端口 J 还起到 AD 整流器和接通唤醒的功能。

复位将端口 J 的所有位初始化为多功能，输入和上拉均禁用。

端口 J 有一种多功能寄存器。如果将端口 J 用作多用途端口，则将三个寄存器的相应位设为 “0“。如果将端口 J 用作其它端口使用，则将功能寄存器的相应位设为 “1“。

注:除非将所有端口 J 的位作为模拟输入引脚来使用，否则转化准确性将降低。理应验证其是否对系统不造成影响。

8.2.10.1 端口J电路类型

	7	6	5	4	3	2	1	0
类型	T19	T19	T19	T19	T18	T17	T17	T17

8.2.10.2 端口J寄存器

基址 = 0×400C_0900

寄存器名称		地址 (基址+)
端口 J 数据寄存器	PJDATA	0×0000
端口 J 功能寄存器 2	PJFR2	0×000C
端口 J 上拉控制寄存器	PJPUP	0×002C
端口 J 输入控制寄存器	PJIE	0×0038

8.2.10.3 PJDATA (端口 J 数据寄存器)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	PJ7	PJ6	PJ5	PJ4	PJ3	PJ2	PJ1	PJ0
复位后	1	1	1	1	1	1	1	1

位	比特符号	类型	功能
31-8	−	R	读作 “0“。
7-0	PJ7 ~ PJ0	R	端口 J 数据寄存器

8.2.10.4 PJFR2 (端口 J 功能寄存器 1)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	PJ7F2	PJ6F2	PJ5F2	PJ4F2	PJ3F2	-	-	-
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-8	−	R	读作 “0“。
7	PJ7F2	R/W	0：端口 1 : KWUP3
6	PJ6F2	R/W	0：端口 1 : KWUP2
5	PJ5F2	R/W	0：端口 1 : KWUP1
4	PJ4F2	R/W	0：端口 1 : KWUP0
3	PJ3F2	R/W	0：端口 1：$\overline{\text{ADTRG}}$
2-0	−	R	读作 “0“。

8.2.10.5 PJPUP (端口 J 上拉控制寄存器)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	PJ7UP	PJ6UP	PJ5UP	PJ4UP	PJ3UP	PJ2UP	PJ1UP	PJ0UP
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-8	−	R	读作 “0“。
7-0	PJ7UP ~ PJ0UP	R/W	上拉 0：禁用 1：启用

8.2.10.6 PJIE (端口 J 输入控制寄存器)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	PJ7IE	PJ6IE	PJ5IE	PJ4IE	PJ3IE	PJ2IE	PJ1IE	PJ0IE
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-8	−	R	读作 “0“。
7-0	PJ7IE ~ PJ0IE	R/W	输入 0：禁用 1：启用

8.2.11 端口 K (PK0 ~ PK7)

端口 K 是一个 8-位输入端口。除多功能输入外，端口 K 还操作 AD 整流器。

复位将端口 K 的所有位初始化为多功能端口，输入和上拉均禁用。

注:除非将所有端口 K 的位作为模拟输入引脚来使用，否则转化准确性将降低。理应验证其是否对系统不造成影响。

8.2.11.1 端口 K 电路类型

	7	6	5	4	3	2	1	0
类型	T17	T17	T17	T17	T17	T17	T17	T17

8.2.11.2 端口 K 寄存器

基址 =0×400C_0A00

寄存器名称		地址 (基址+)
端口 K 数据寄存器	PKDATA	0×0000
端口 K 上拉控制寄存器	PKPUP	0×002C
端口 K 输入控制寄存器	PKIE	0×0038

8.2.11.3 PKDATA (端口 K 数据寄存器)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	PK7	PK6	PK5	PK4	PK3	PK2	PK1	PK0
复位后	1	1	1	1	1	1	1	1

位	比特符号	类型	功能
31-8	−	R	读作 “0“。
7-0	PK7 ~ PK0	R	端口 K 数据寄存器

8.2.11.4 PKPUP (端口K上拉控制寄存器)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	PK7UP	PK6UP	PK5UP	PK4UP	PK3UP	PK2UP	PK1UP	PK0UP
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-8	−	R	读作 “0“。
7-0	PK7UP ~ PK0UP	R/W	上拉 0:禁用 1:启用

8.2.11.5 PKIE (端口 K 输入控制寄存器)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	PK7IE	PK6IE	PK5IE	PK4IE	PK3IE	PK2IE	PK1IE	PK0IE
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-8	−	R	读作 “0“。
7-0	PK7IE ~ PK0IE	R/W	输入 0：禁用 1：启用

8.2.12 端口 L (PL0 ~ PL7)

端口 L 是一个多用途，8-位输入/输出端口。该端口的输入 / 输出位为单位加以规定。除多功能输入 / 输出外，端口 L 还起到串行通道，串行总线接口，外部信号中断输入以及 16-位计时器 / 事件计数器的功能。

复位将端口 L 的所有位初始化为多功能端口，输入，输出和上拉均禁用。

端口 L 有三种类型功能寄存器。如果端口 L 用作多用途端口，则将三种寄存器的相应位设为 “0“。如果端口 L 用作其它端口使用，将功能寄存器的相应位设为 “1“。不能将某些功能寄存器同时设为 “1“。

注:在非 STOP 模式中，如果输入能在 P×IE 中激活，无论 P×FR 寄存器如何设置，则中断输入被激活。为设备编程时，应确定将未使用的中断禁用。

8.2.12.1 端口 L 电路类型

	7	6	5	4	3	2	1	0
类型	T21	T24	T23	T22	T21	T20	T20	T20

8.2.12.2 端口 L 寄存器

基址 = 0×400C_0B00

寄存器名称		地址 (基址+)
端口 L 数据寄存器	PLDATA	0×0000
端口 L 输出控制寄存器	PLCR	0×0004
端口 L 功能寄存器 1	PLFR1	0×0008
端口 L 功能寄存器 2	PLFR2	0×000C
端口 L 功能寄存器 3	PLFR3	0×0010
端口 L 开漏控制寄存器	PLOD	0×0028
端口 L 上拉控制寄存器	PLPUP	0×002C
端口 L 输入控制寄存器	PLIE	0×0038

8.2.12.3 PLDATA (端口 L 数据寄存器)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	PL7	PL6	PL5	PL4	PL3	PL2	PL1	PL0
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-8	−	R	读作 “0“。
7-0	PL7 ~ PL0	R/W	端口 L 数据寄存器

8.2.12.4 PLCR (端口L输出控制寄存器)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	PL7C	PL6C	PL5C	PL4C	PL3C	PL2C	PL1C	PL0C
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-8	−	R	读作 “0“。
7-0	PL7C ~ PL0C	R/W	输出 0：禁用 1：启用

8.2.12.5 PLFR1 (端口 K 功能寄存器 1)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	PL7F1	PL6F1	PL5F1	PL4F1	PL3F1	PL2F1	PL1F1	PL0F1
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-8	−	R	读作 “0“。
7	PL7F1	R/W	0 : 端口 1 : INT1
6	PL6F1	R/W	0: 端口 1 : SCLK1
5	PL5F1	R/W	0 : 端口 1 : RXD1
4	PL4F1	R/W	0 : 端口 1 : TXD1
3	PL3F1	R/W	0:端口 1:INT0
2	PL2F1	R/W	0 : 端口 1 : SCK0
1	PL1F1	R/W	0 : 端口 1 : SCL0 / SI0
0	PL0F1	R/W	0 : 端口 1 : SDA0 / SO0

8.2.12.6 PLFR2 (端口 L 功能寄存器 2)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	PL7F2	PL6F2	PL5F2	PL4F2	PL3F2	PL2F2	PL1F2	PL0F2
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-8	−	R	读作 “0“。
7	PL7F2	R/W	0：端口 1 : TB7OUT
6	PL6F2	R/W	0：端口 1 : TB6OUT
5	PL5F2	R/W	0：端口 1 : TB5OUT
4	PL4F2	R/W	0：端口 1 : TB4OUT
3	PL3F2	R/W	0：端口 1 :TB3OUT
2	PL2F2	R/W	0：端口 1 :TB2OUT
1	PL1F2	R/W	0：端口 1 : TB1OUT
0	PL0F2	R/W	0：端口 1 : TB0OUT

8.2.12.7 PLFR3 (端口 L 功能寄存器 3)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	-	PL6F3	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-7	−	R	读作 “0“。
6	PL6F3	R/W	0：端口 1：$\overline{CTS1}$
5-4	−	R/W	写入 “0“。
3-0	−	R	读作 “0“。

8.2.12.8 PLOD (端口 L 开漏控制寄存器)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	PL7OD	PL6OD	PL5OD	PL4OD	PL3OD	PL2OD	PL1OD	PL0OD
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-8	−	R	读作 “0“。
7-0	PL7OD ~ PL0OD	R/W	0 : CMOS 1：开漏

8.2.12.9 PLPUP (端口 L 上拉控制寄存器)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	PL7UP	PL6UP	PL5UP	PL4UP	PL3UP	PL2UP	PL1UP	PL0UP
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-8	−	R	读作 “0“。
7-0	PL7UP ~ PL0UP	R/W	上拉 0：禁用 1：启用

8.2.12.10 PLIE (端口 L 输入控制寄存器)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	PL7IE	PL6IE	PL5IE	PL4IE	PL3IE	PL2IE	PL1IE	PL0IE
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-8	−	R	读作 “0“。
7-0	PL7IE ~ PL0IE	R/W	输入 0：禁用 1：启用

8.2.13 端口 M (PM0 ~ PM7)

端口 M 是一个多用途，8 位输入/输出端口。该端口的输入 / 输出位为单位加以规定。除多功能输入 / 输出外，端口 M 还起到串口通道，外部信号阻断输入，16 位计时器 / 事件计数器与告警输出的功能。

复位将端口 M 的所有位初始化多功能端口，输入，输出和上拉均禁用。

端口 M 有三种类型功能寄存器。如果端口 M 用作多用途端口，则将三种寄存器的相应位设为 “0“。如果端口 M 用作其它端口使用，则将功能寄存器的相应位设为 “1“。不能将某些功能寄存器同时设为 “1“。

注:在非 STOP 模式中，如果输入能在 P×IE 中激活，无论 P×FR 寄存器如何设置，则中断输入被激活。为设备编程时，应确定将未使用的中断禁用。

8.2.13.1 端口电路类型

	7	6	5	4	3	2	1	0
类型	T30	T29	T28	T27	T21	T23	T26	T25

8.2.13.2 端口 M 寄存器

基址 = 0×400C_0C00

寄存器名称		地址 (基址+)
端口 M 数据寄存器	PMDATA	0×0000
端口 M 输出控制寄存器	PMCR	0×0004
端口 M 功能寄存器 1	PMFR1	0×0008
端口 M 功能寄存器 2	PMFR2	0×000C
端口 M 功能寄存器 3	PMFR3	0×0010
端口 M 开漏控制寄存器	PMOD	0×0028
端口 M 上拉控制寄存器	PMPUP	0×002C
端口 M 输入控制寄存器	PMIE	0×0038

8.2.13.3 PMDATA (端口 M 数据寄存器)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	PM7	PM6	PM5	PM4	PM3	PM2	PM1	PM0
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-8	−	R	读作 “0“。
7-0	PM7 ~ PM0	R/W	端口 M 数据寄存器

8.2.13.4 PMCR (端口M输出控制寄存器)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	PM7C	PM6C	PM5C	PM4C	PM3C	PM2C	PM1C	PM0C
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-8	−	R	读作 “0“。
7-0	PM7C ~ PM0C	R/W	输出 0：禁用 1：启用

8.2.13.5 PMFR1 (端口M功能寄存器1)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	PM7F1	PM6F1	PM5F1	PM4F1	PM3F1	PM2F1	PM1F1	PM0F1
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-8	−	R	读作 “0“。
7	PM7F1	R/W	0 : 端口 1 : INT3
6	PM6F1	R/W	0 : 端口 1 : RXD3
5	PM5F1	R/W	0 : 端口 1 : TXD3
4	PM4F1	R/W	0 : 端口 1 : SCLK3
3	PM3F1	R/W	0 : 端口 1 : INT2
2	PM2F1	R/W	0 : 端口 1 : RXD2
1	PM1F1	R/W	0 : 端口 1 : TXD2
0	PM0F1	R/W	0 : 端口 1 : SCLK2

8.2.13.6 PMFR2 (端口 M 功能寄存器 2)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	-	-	-	-	PM3F2	PM2F2	PM1F2	PM0F2
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-4	−	R	读作 “0“。
3	PM3F2	R/W	0：端口 1：TB3OUT
2	PM2F2	R/W	0：端口 1：$\overline{\text{ALARM}}$
1	PM1F2	R/W	0：端口 1：TB1IN1
0	PM0F2	R/W	0：端口 1：TB1IN0

8.2.13.7 PMFR3 (端口 M 功能寄存器 3)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	-	-	-	PM4F3	-	-	-	PM0F3
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-5	−	R	读作 “0“。
4	PM4F3	R/W	0：端口 1：$\overline{CTS3}$
3-1	−	R	读作 “0“。
0	PM0F3	R/W	0：端口 1：$\overline{CTS2}$

8.2.13.8 PMOD (端口 M 开漏控制寄存器)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	PM7OD	PM6OD	PM5OD	PM4OD	PM3OD	PM2OD	PM1OD	PM0OD
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-8	−	R	读作 “0“。
7-0	PM7OD ~ PM0OD	R/W	0 : CMOS 1：开漏

8.2.13.9 PMPUP (端口 M 上拉控制寄存器)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	PM7UP	PM6UP	PM5UP	PM4UP	PM3UP	PM2UP	PM1UP	PM0UP
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-8	−	R	读作 “0“。
7-0	PM7UP ~ PM0UP	R/W	上拉 0：禁用 1：启用

8.2.13.10 PMIE (端口 M 输入控制寄存器)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	PM7IE	PM6IE	PM5IE	PM4IE	PM3IE	PM2IE	PM1IE	PM0IE
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-8	−	R	读作 “0“。
7-0	PM7IE ~ PM0IE	R/W	输入 0:禁用 1:启用

8.2.14 端口 N (PN0 ~ PN7)

端口 N 是一个多用途，8 位输入 / 输出端口。该端口的输入 / 输出位为单位加以规定。除多功能输入 / 输出外，端口 N 还起到串口通道，外部信号阻断输入，16 位定时器 / 事件计数器及远程控制信号预处理器输入的功能。

复位将端口 N 的所有位初始化为多功能端口，输入，输出和上拉均禁用。

端口 N 有三种类型功能寄存器。如果端口 N 用作多用途端口，则将三种寄存器的相应位设为 "0"。如果端口 N 用作其它端口使用，则将功能寄存器的相应位设为 "1"。不能将些功能寄存器同时设为 "1"。

注:在非 STOP 模式中，如果输入能在 P×IE 中激活，无论 P×FR 寄存器如何设置，则中断输入被激活。为设备编程时，应确定将未使用的中断禁用。

8.2.14.1 端口 N 电路类型

	7	6	5	4	3	2	1	0
类型	T30	T25	T29	T28	T30	T25	T29	T28

8.2.14.2 端口 N 寄存器

基址 = 0×400C_0D00

寄存器名称		地址 (基址+)
端口 N 数据寄存器	PNDATA	0×0000
端 N 口输出控制寄存器	PNCR	0×0004
端口 N 功能寄存器 1	PNFR1	0×0008
端口 N 功能寄存器 2	PNFR2	0×000C
端口 N 功能寄存器 3	PNFR3	0×0010
端口 N 开漏控制寄存器	PNOD	0×0028
端口 N 上拉控制寄存器	PNPUP	0×002C
端口 N 输入控制寄存器	PNIE	0×0038

8.2.14.3 PNDATA (端口 N 数据寄存器)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	PN7	PN6	PN5	PN4	PN3	PN2	PN1	PN0
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-8	−	R	读作 “0“。
7-0	PN7 ~ PN0	R/W	端口 N 数据寄存器

8.2.14.4 PNCR (端口 N 输出控制寄存器)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	PN7C	PN6C	PN5C	PN4C	PN3C	PN2C	PN1C	PN0C
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-8	−	R	读作 “0“。
7-0	PN7C ~ PN0C	R/W	输出 0：禁用 1：启用

8.2.14.5 PNFR1 (端口 N 功能寄存器 1)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	PN7F1	PN6F1	PN5F1	PN4F1	PN3F1	PN2F1	PN1F1	PN0F1
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-8	−	R	读作 “0“。
7	PN7F1	R/W	0：端口 1 : INT8
6	PN6F1	R/W	0：端口 1 : SCLK5
5	PN5F1	R/W	0：端口 1 : RXD5
4	PN4F1	R/W	0：端口 1 : TXD5
3	PN3F1	R/W	0：端口 1:INT4
2	PN2F1	R/W	0：端口 1 : SCLK4
1	PN1F1	R/W	0：端口 1 : RXD4
0	PN0F1	R/W	0：端口 1 : TXD4

8.2.14.6 PNFR2 (端口 N 功能寄存器 2)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	PN7F2	PN6F2	-	-	PN3F2	PN2F2	-	-
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-8	−	R	读作 “0“。
7	PN7F2	R/W	0 : 端口 1 : TBFIN1
6	PN6F2	R/W	0 : 端口 1 : TBFIN0
5-4	−	R	读作 “0“。
3	PN3F2	R/W	0 : 端口 1 : TB2IN1
2	PN2F2	R/W	0 : 端口 1 : TB2IN0
1-0	−	R	读作 “0“。

8.2.14.7 PNFR3 (端口 N 功能寄存器 3)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	PN7F3	PN6F3	-	-	PN3F3	PN2F3	-	-
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-8	−	R	读作 “0“。
7	PN7F3	R/W	0：端口 1：RMC1
6	PN6F3	R/W	0：端口 1：$\overline{\text{CTS5}}$
5-4	−	R	读作 “0“。
3	PN3F3	R/W	0：端口 1：RMC0
2	PN2F3	R/W	0：端口 1：$\overline{\text{CTS4}}$
1 ~ 0	−	R	读作 “0“。

8.2.14.8 PNOD (端口 N 开漏控制寄存器)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	PN7OD	PN6OD	PN5OD	PN4OD	PN3OD	PN2OD	PN1OD	PN0OD
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-8	−	R	读作 “0“。
7-0	PN7OD ~ PN0OD	R/W	0 : CMOS 1 : 开漏

8.2.14.9 PNPUP (端口 N 上拉控制寄存器)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	PN7UP	PN6UP	PN5UP	PN4UP	PN3UP	PN2UP	PN1UP	PN0UP
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-8	−	R	读作 “0“。
7-0	PN7UP ~ PN0UP	R/W	上拉 0 : 禁用 1 : 启用

8.2.14.10 PNIE (端口 N 输入控制寄存器)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	PN7IE	PN6IE	PN5IE	PN4IE	PN3IE	PN2IE	PN1IE	PN0IE
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-8	−	R	读作 “0“。
7-0	PN7IE ~ PN0IE	R/W	输入 0：禁用 1：启用

8.2.15 端口 O (PO0 ~ PO7)

端口 O 是一个多用途，8-位输入 / 输出端口。该端口的输入 / 输出位为单位加以规定。除多功能输入 / 输出外，端口 O 还起到串口通道功能 (UART/SIO)，外部信号阻断输入，16 位计时器 / 事件计数器输出的功能。

复位将端口 O 的所有位初始化为多功能端口，输入，输出和上拉均禁用。

端口 O 有三种类型功能寄存器。如果端口 O 用作多用途端口，则将三种寄存器的相应位设为 "0"。如果端口 O 用作其它端口使用，则将功能寄存器的相应位设为 "1"。不能将某些功能寄存器同时设为 "1"。

注:在非 STOP 模式中，如果输入能在 P×IE 中激活，无论 POFR1 寄存器如何设置，则中断输入被激活。为设备编程时，应确定将未使用的中断禁用。

8.2.15.1 端口 O 电路类型

	7	6	5	4	3	2	1	0
类型	T21	T24	T23	T22	T21	T24	T23	T22

8.2.15.2 端口 O 寄存器

基址 = 0×400C_0E00

寄存器名称		地址 (基址+)
端口 O 数据寄存器	PODATA	0×0000
端口 O 输出控制寄存器	POCR	0×0004
端口 O 功能寄存器 1	POFR1	0×0008
端口 O 功能寄存器 2	POFR2	0×000C
端口 O 功能寄存器 3	POFR3	0×0010
端口 O 开漏控制寄存器	POOD	0×0028
端口 O 上拉控制寄存器	POPUP	0×002C
端口 O 输入控制寄存器	POIE	0×0038

8.2.15.3 PODATA (端口 O 数据寄存器)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	PO7	PO6	PO5	PO4	PO3	PO2	PO1	PO0
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-8	−	R	读作 “0“。
7-0	PO7 ~ PO0	R/W	端口 O 数据寄存器

8.2.15.4 POCR (端口 O 输出控制寄存器)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	PO7C	PO6C	PO5C	PO4C	PO3C	PO2C	PO1C	PO0C
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-8	−	R	读作 “0“。
7-0	PO7C ~ PO0C	R/W	输出 0：禁用 1：启用

8.2.15.5 POFR1 (端口 O 功能寄存器 1)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	PO7F1	PO6F1	PO5F1	PO4F1	PO3F1	PO2F1	PO1F1	PO0F1
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-8	−	R	读作 “0“。
7	PO7F1	R/W	0 : 端口 1 : INTA
6	PO6F1	R/W	0 : 端口 1 : SCLK7
5	PO5F1	R/W	0 : 端口 1 : RXD7
4	PO4F1	R/W	0 : 端口 1 : TXD7
3	PO3F1	R/W	0 : 端口 1 : INT9
2	PO2F1	R/W	0 : 端口 1 : SCLK6
1	PO1F1	R/W	0 : 端口 1 : RXD6
0	PO0F1	R/W	0 : 端口 1 : TXD6

8.2.15.6 POFR2 (端口 O 功能寄存器 2)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	PO7F2	PO6F2	PO5F2	PO4F2	PO3F2	PO2F2	PO1F2	PO0F2
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-8	−	R	读作 “0“。
7	PO7F2	R/W	0 : 端口 1 : TBFOUT
6	PO6F2	R/W	0 : 端口 1 : TBEOUT
5	PO5F2	R/W	0 : 端口 1 : TBDOUT
4	PO4F2	R/W	0 : 端口 1 : TBCOUT
3	PO3F2	R/W	0 : 端口 1 : TBBOUT
2	PO2F2	R/W	0 : 端口 1 : TBAOUT
1	PO1F2	R/W	0 :端口 1 : TB9OUT
0	PO0F2	R/W	0 : 端口 1 : TB8OUT

8.2.15.7 POFR3 (端口 O 功能寄存器 3)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	-	PO6F3	-	-	-	PO2F3	-	-
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-7	−	R	读作 “0“。
6	PO6F3	R/W	0:端口 1: $\overline{CTS7}$
5-3	−	R	读作 “0“。
2	PO2F3	R/W	0:端口 1: $\overline{CTS6}$
1-0	−	R	读作 “0“。

8.2.15.8 POOD (端口 O 开漏控制寄存器)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	PO7OD	PO6OD	PO5OD	PO4OD	PO3OD	PO2OD	PO1OD	PO0OD
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-8	−	R	读作 “0“。
7-0	PO7OD ~ PO0OD	R/W	0 : CMOS 1 : 开漏

8.2.15.9 POPUP (端口 O 上拉控制寄存器)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	PO7UP	PO6UP	PO5UP	PO4UP	PO3UP	PO2UP	PO1UP	PO0UP
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-8	−	R	读作 “0“。
7-0	PO7UP ~ PO0UP	R/W	上拉 0：禁用 1：启用

8.2.15.10 POIE (端口O输入控制寄存器)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	PO7IE	PO6IE	PO5IE	PO4IE	PO3IE	PO2IE	PO1IE	PO0IE
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-8	−	R	读作 “0“。
7-0	PO7IE ~ PO0IE	R/W	输入 0：禁用 1：启用

8.2.16 端口 P (PP0 ~ PP6)

端口 P 是一个多用途，7-位输入 / 输出端口。该端口的输入 / 输出位为单位加以规定。除多功能输入 / 输出外，端口 P 还执行外部总线接口和 SSP 功能。

复位将端口 P 的所有位初始化为多功能端口，输入，输出和上拉均禁用。

端口 P 有两种类型功能寄存器。如果端口 P 用作多用途端口，则将两种寄存器的相应位设为 “0“。如果端口 P 用作其它端口使用，则将功能寄存器的相应位设为 “1“。不能将两种功能寄存器同时设为 “1“。

8.2.16.1 端口 P 电路类型

	7	6	5	4	3	2	1	0
类型	-	T5	T35	T34	T33	T32	T31	T5

8.2.16.2 端口 P 寄存器

基址 = 0×400C_0F00

寄存器名称		地址 (基址+)
端 P 口数据寄存器	PPDATA	0×0000
端口 P 输出控制寄存器	PPCR	0×0004
端口 P 功能寄存器 1	PPFR1	0×0008
端口 P 功能寄存器 2	PPFR2	0×000C
端口 P 开漏控制寄存器	PPOD	0×0028
端口 P 上拉控制寄存器	PPPUP	0×002C
端口 P 输入控制寄存器	PPIE	0×0038

8.2.16.3 PPDATA (端口 P 数据寄存器)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	-	PP6	PP5	PP4	PP3	PP2	PP1	PP0
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-7	−	R	读作 “0“。
6-0	PP6 ~ PP0	R/W	端口 P 数据寄存器

8.2.16.4 PPCR (端口P输出控制寄存器)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	-	PP6C	PP5C	PP4C	PP3C	PP2C	PP1C	PP0C
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-7	−	R	读作 “0“。
6-0	PP6C ~ PP0C	R/W	输出 0：禁用 1：启用

8.2.16.5 PPFR1 (端口 P 功能寄存器 1)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	-	PP6F1	PP5F1	PP4F1	PP3F1	PP2F1	PP1F1	PP0F1
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-7	–	R	读作 “0“。
6	PP6F1	R/W	0：端口 1：$\overline{ALE}$
5	PP5F1	R/W	0：端口 1：$\overline{OE}$
4	PP4F1	R/W	0：端口 1：$\overline{WE}$
3	PP3F1	R/W	0：端口 1：$\overline{BLS1}$
2	PP2F1	R/W	0：端口 1：$\overline{BLS0}$
1	PP1F1	R/W	写入 “0“。
0	PP0F1	R/W	0：端口 1：$\overline{CS2}$

8.2.16.6 PPFR2 (端口 P 功能寄存器 2)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	-	-	PP5F2	PP4F2	PP3F2	PP2F2	-	-
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-6	−	R	读作 “0“。
5	PP5F2	R/W	0 : 端口 1 : SPFSS
4	PP4F2	R/W	0 : 端口 1 : SPCLK
3	PP3F2	R/W	0 : 端口 1 : SPDI
2	PP2F2	R/W	0 : 端口 1 : SPDO
1-0	−	R	读作 “0“。

8.2.16.7 PPOD (端口 P 开漏控制寄存器)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	-	PP6OD	PP5OD	PP4OD	PP3OD	PP2OD	PP1OD	PP0OD
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-7	−	R	读作 “0“。
6-0	PP6OD ~ PP0OD	R/W	0 : CMOS 1 : 开漏

8.2.16.8 PPPUP (端口 P 上拉控制寄存器)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	-	PP6UP	PP5UP	PP4UP	PP3UP	PP2UP	PP1UP	PP0UP
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-7	−	R	读作 “0“。
6-0	PP6UP ~ PP0UP	R/W	上拉 0 : 禁用 1 : 启用

8.2.16.9 PPIE (端口 P 输入控制寄存器)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	-	PP6IE	PP5IE	PP4IE	PP3IE	PP2IE	PP1IE	PP0IE
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-7	−	R	读作 “0“。
6-0	PP6IE ~ PP0IE	R/W	输入 0：禁用 1：启用

8.3 端口方块图

8.3.1 端口类型

端口分类如下。各端口类型方块图，见后续页所述。

点划线为"端口方块图"所述等效电路的组成部分。

表 8-3 功能列表

类型	输入/输出	功能 1	功能 2	功能 3	模拟	上拉	可编程开漏	注
T1	I/O	I/O	–	–	–	R	o	
T2	I/O	输出	输出	–	–	R	o	
T3	I/O	输出	输入	–	–	R	o	
T4	I/O	输出	I/O	输入	–	R	o	
T5	I/O	输出	–	–	–	R	o	
T6	I/O	输入 (中断)	输出	–	–	R	o	
T7	I/O	输出	–	–	–	R	o	
T8	I/O	I/O	输入	–	–	R	o	
T9	I/O	I/O	–	输入	–	R	o	
T10	I/O	输入 (中断)	–	输入	–	R	o	
T11	I/O	输入 (中断)	–	输出	–	R	o	
T12	I/O	I/O	输入	–	–	R	o	
T13	I/O	输入 (中断)	输入	–	–	R	o	
T14	I/O	–	–	–	–	R	o	$\overline{\text{BOOT}}$输入将在重新设置期间启动。
T15	I/O	输入	–	–	–	–	–	Nch 开漏端口
T16	I/O	输入 (中断)	–	–	–	NoR	o	
T17	输入	–	–	–	o	R	–	
T18	输入	输入	–	–	o	R	–	
T19	输入	输入	–	–	o	R	–	
T20	I/O	I/O	输出	–	–	R	o	
T21	I/O	输入	输出	–	–	R	o	
T22	I/O	输出	输出	–	–	R	o	
T23	I/O	输入	输出	–	–	R	o	
T24	I/O	I/O	输出	输入	–	R	o	
T25	I/O	I/O	输入	输入	–	R	o	
T26	I/O	输出	输入	–	–	R	o	
T27	I/O	I/O	–	输出	–	R	o	
T28	I/O	输出	–	–	–	R	o	
T29	I/O	输入	–	–	–	R	o	
T30	I/O	输入 (中断)	–	–	–	R	o	
T31	I/O	–	–	–	–	R	o	
T32	I/O	输出	输出	–	–	R	o	
T33	I/O	输出	输入	–	–	R	o	
T34	I/O	输出	I/O	–	–	R	o	
T35	I/O	输出	–	–	–	R	o	

中断 : 中断输入
-: 不存在
o: 存在

R: 复位期间强制关闭
NoR: 不受复位影响

表 8-3 功能列表

类型	输入/输出	功能 1	功能 2	功能 3	模拟	上拉	可编程开漏	注
T36	I/O	输出	输出	输出	−	R	O	
T37	I/O	输出	输入	输入	−	R	O	
T38	I/O	I/O	输出	输出	−	R	O	
T39	I/O	输入	输入	输出	−	R	O	

中断 : 中断输入

-: 不存在

o: 存在

R: 复位期间强制关闭

NoR: 不受复位影响

8.3.2 T1型

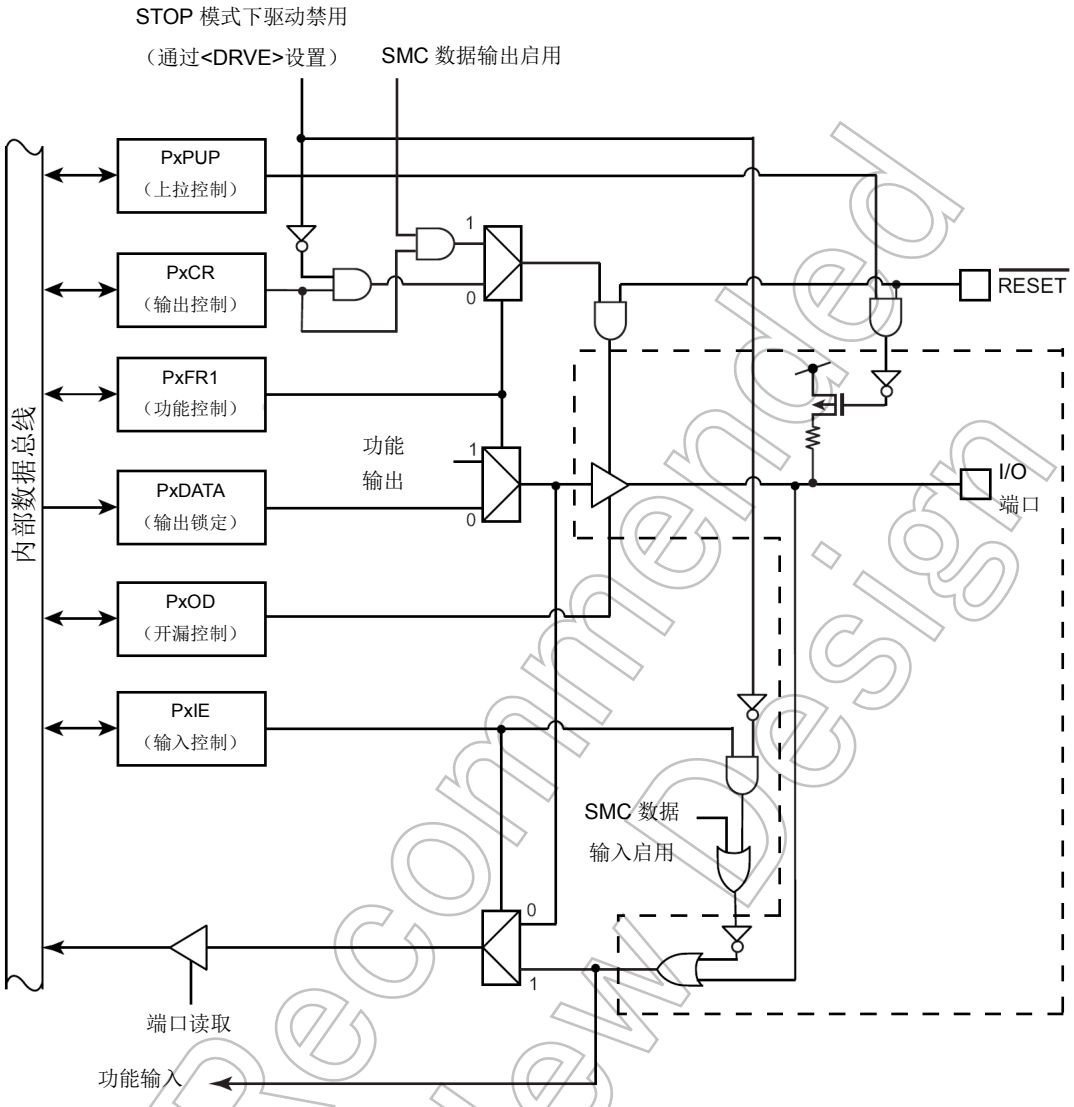


图 8-1 T1 型端口

8.3.3 T2型端口

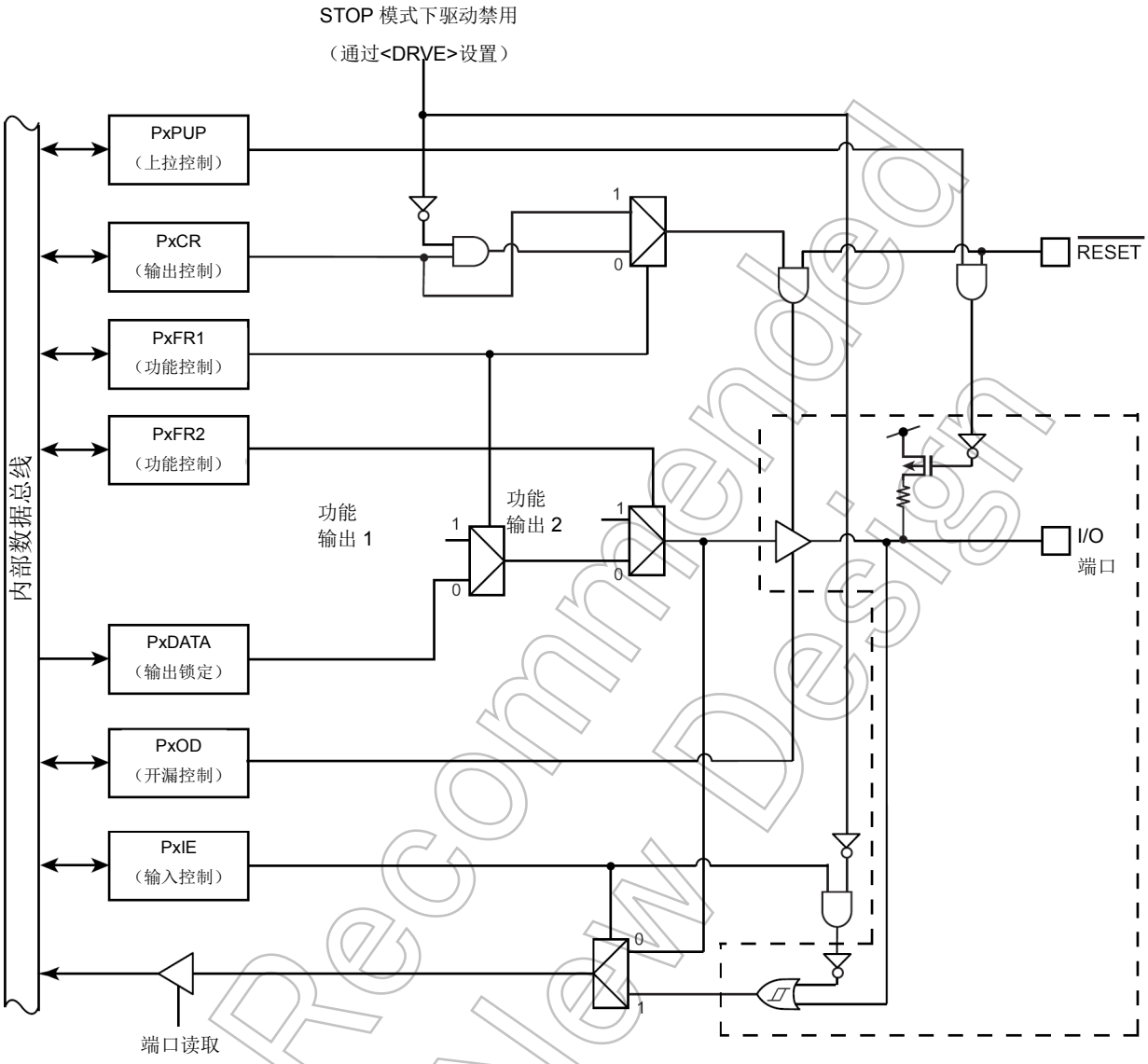


图 8-2 T2 型端口

8.3.4 T3型

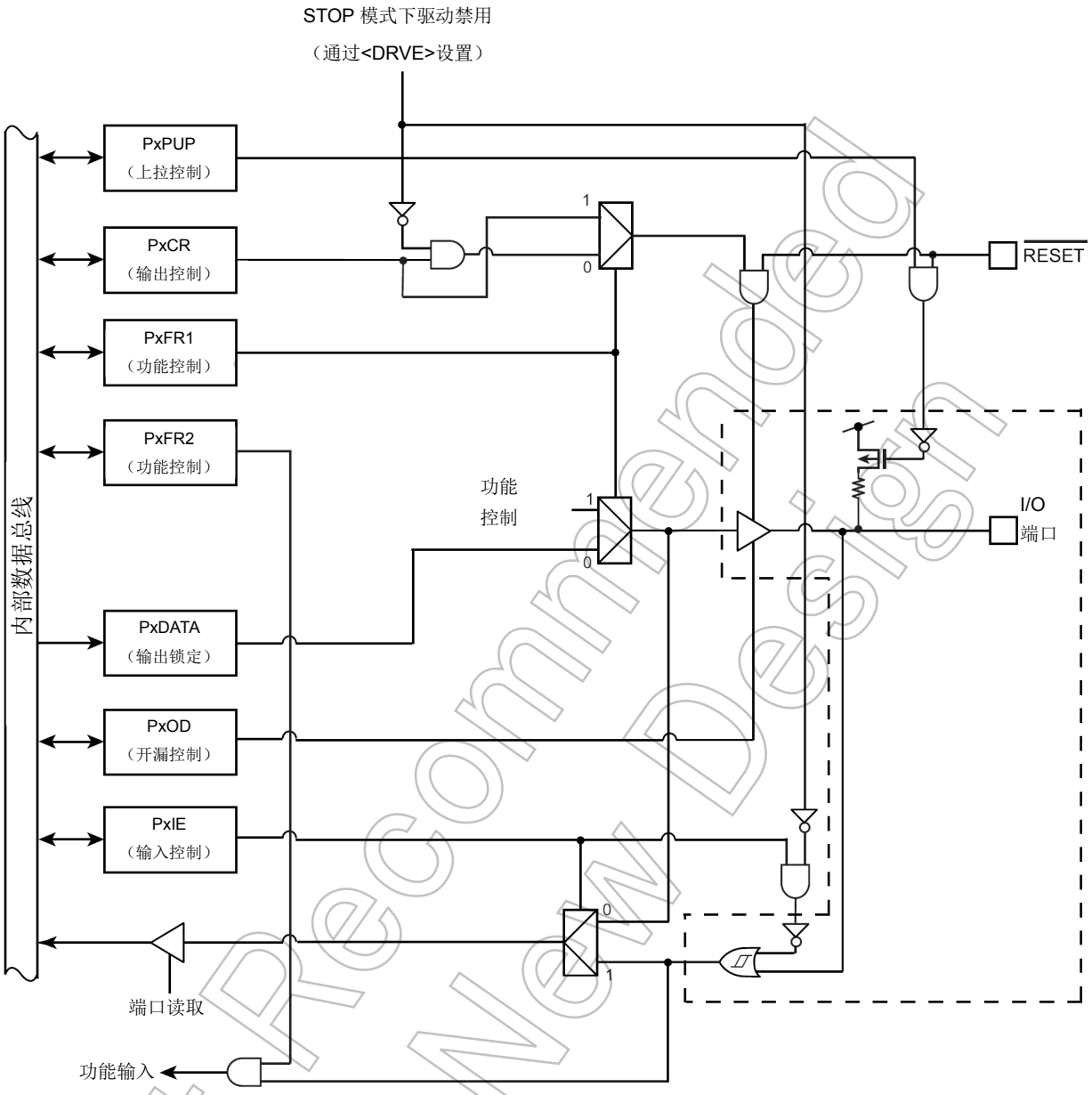


图 8-3 T3 型端口

8.3.5 T4型

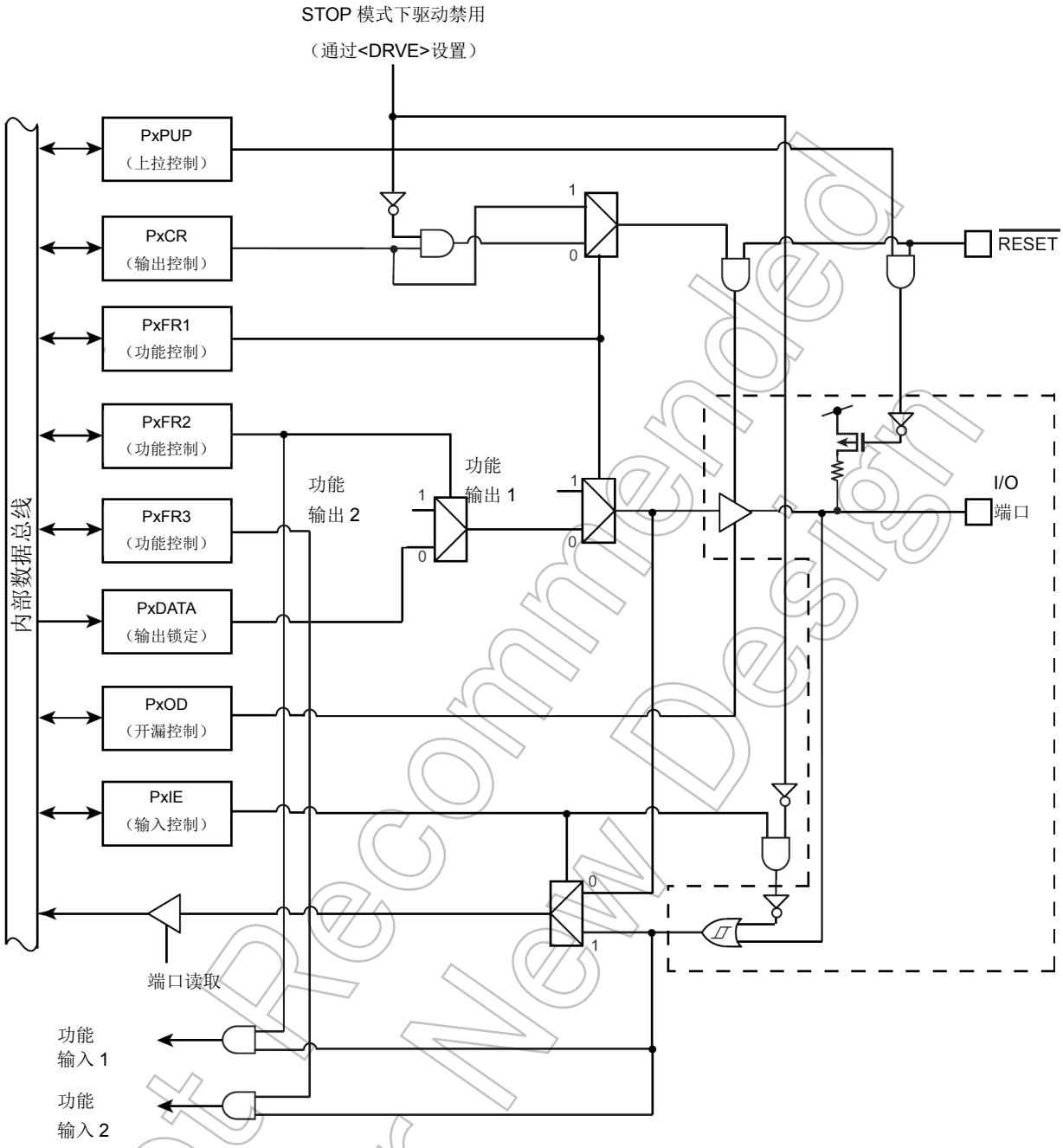


图 8-4 T4 型端口

8.3.6 T5型

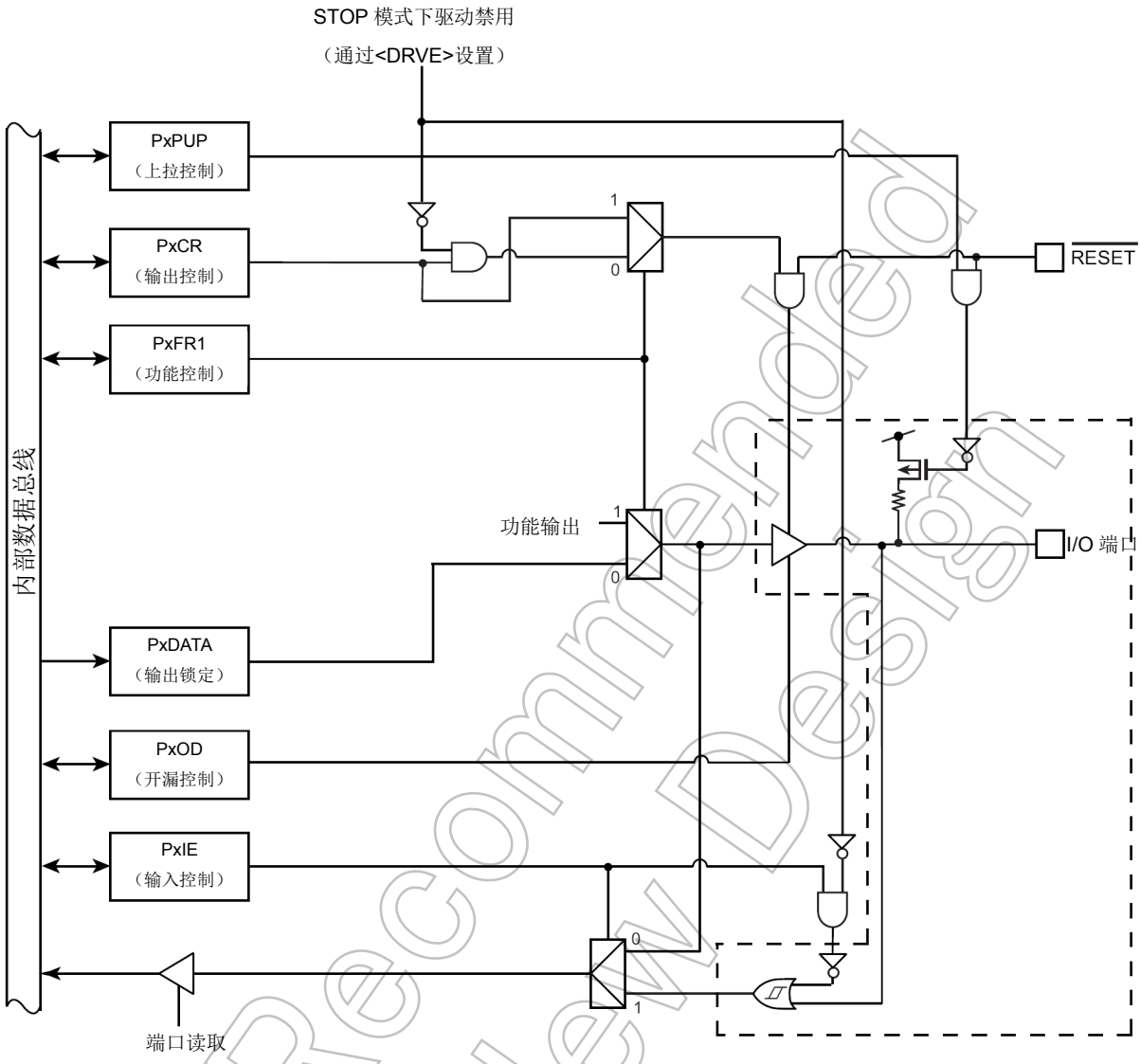


图 8-5 T5 型端口

8.3.7 T6型

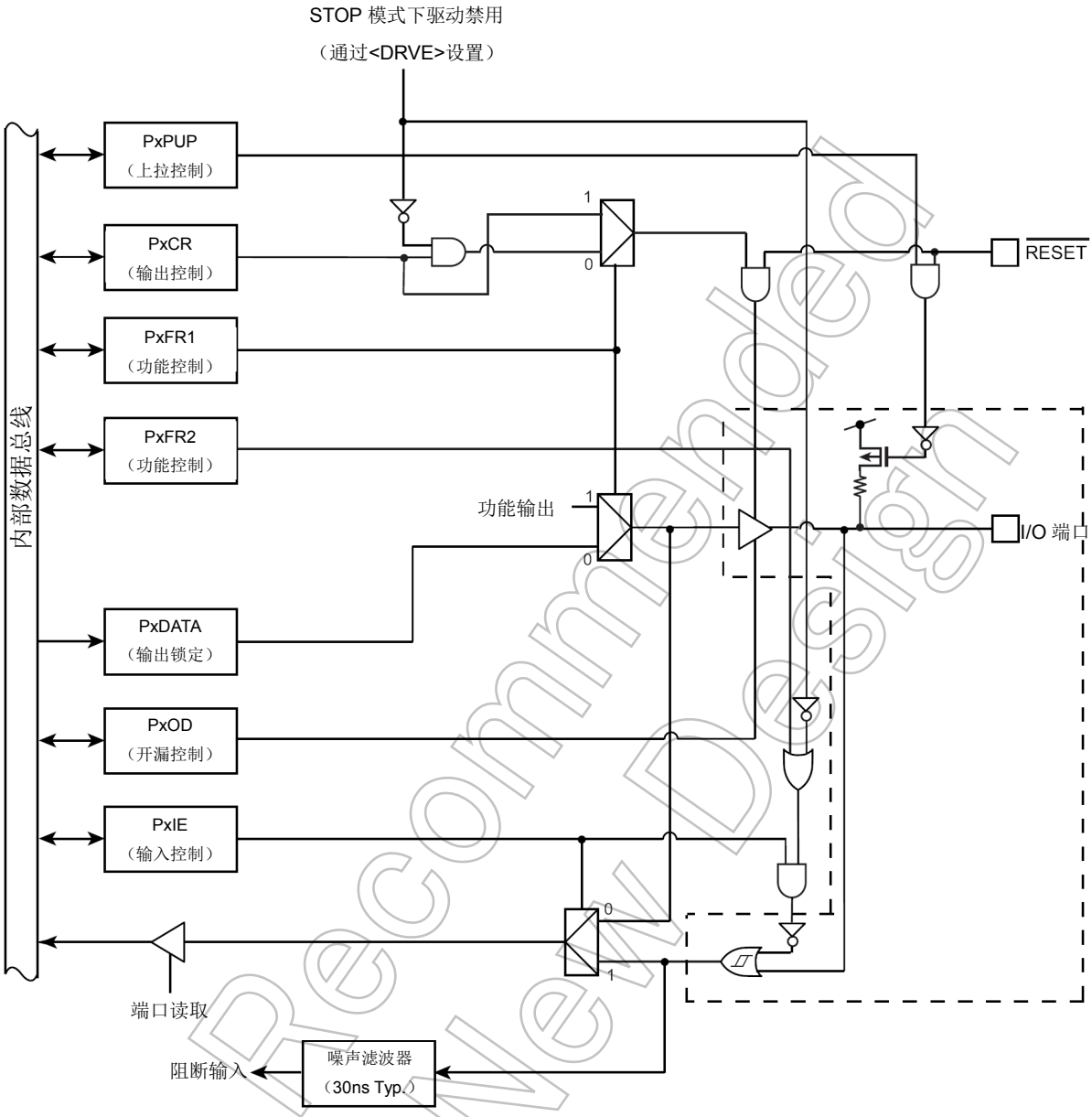


图 8-6 T6 型端口

8.3.8 T7型

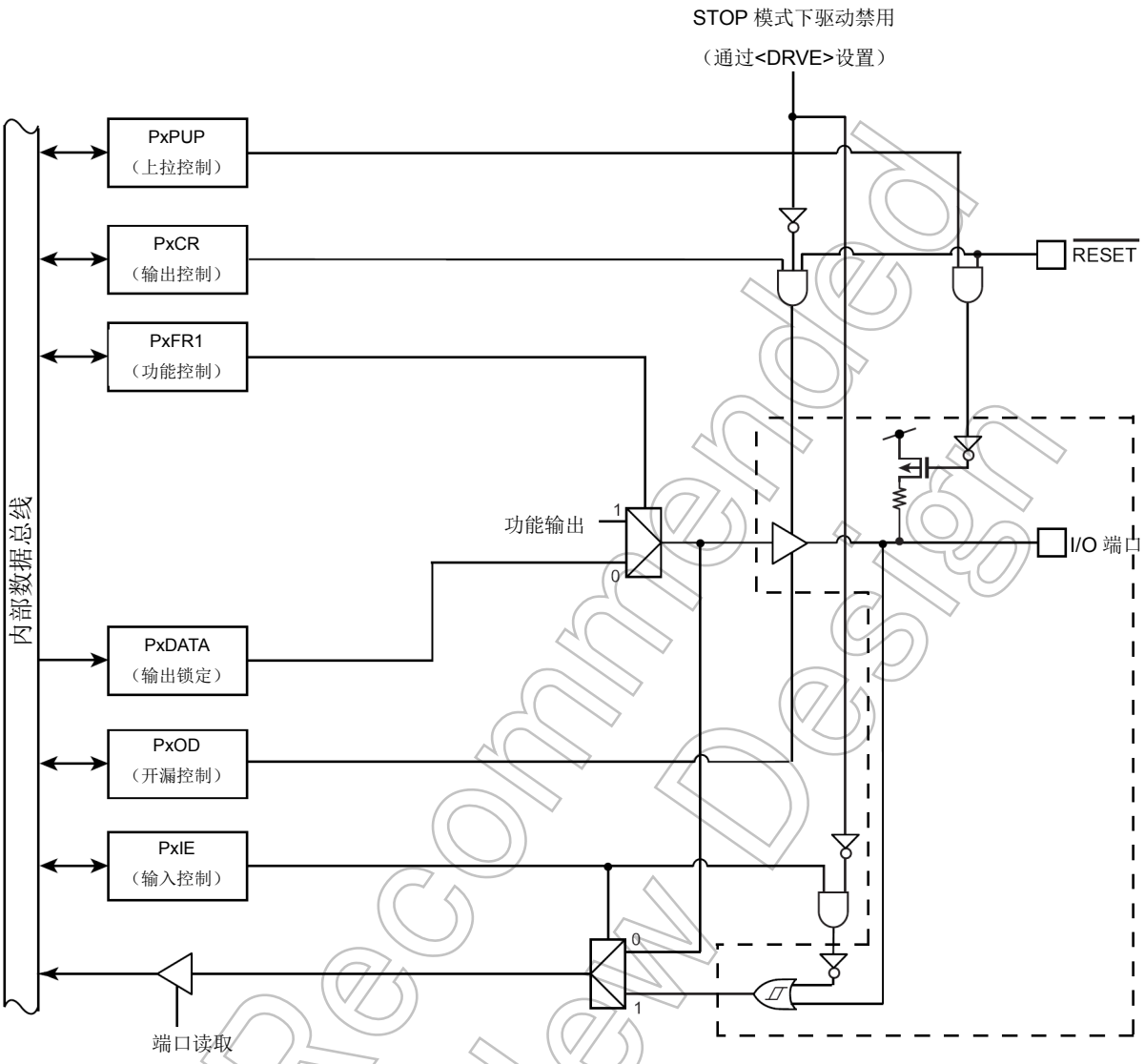


图 8-7 T7 型端口

8.3.9 T8型

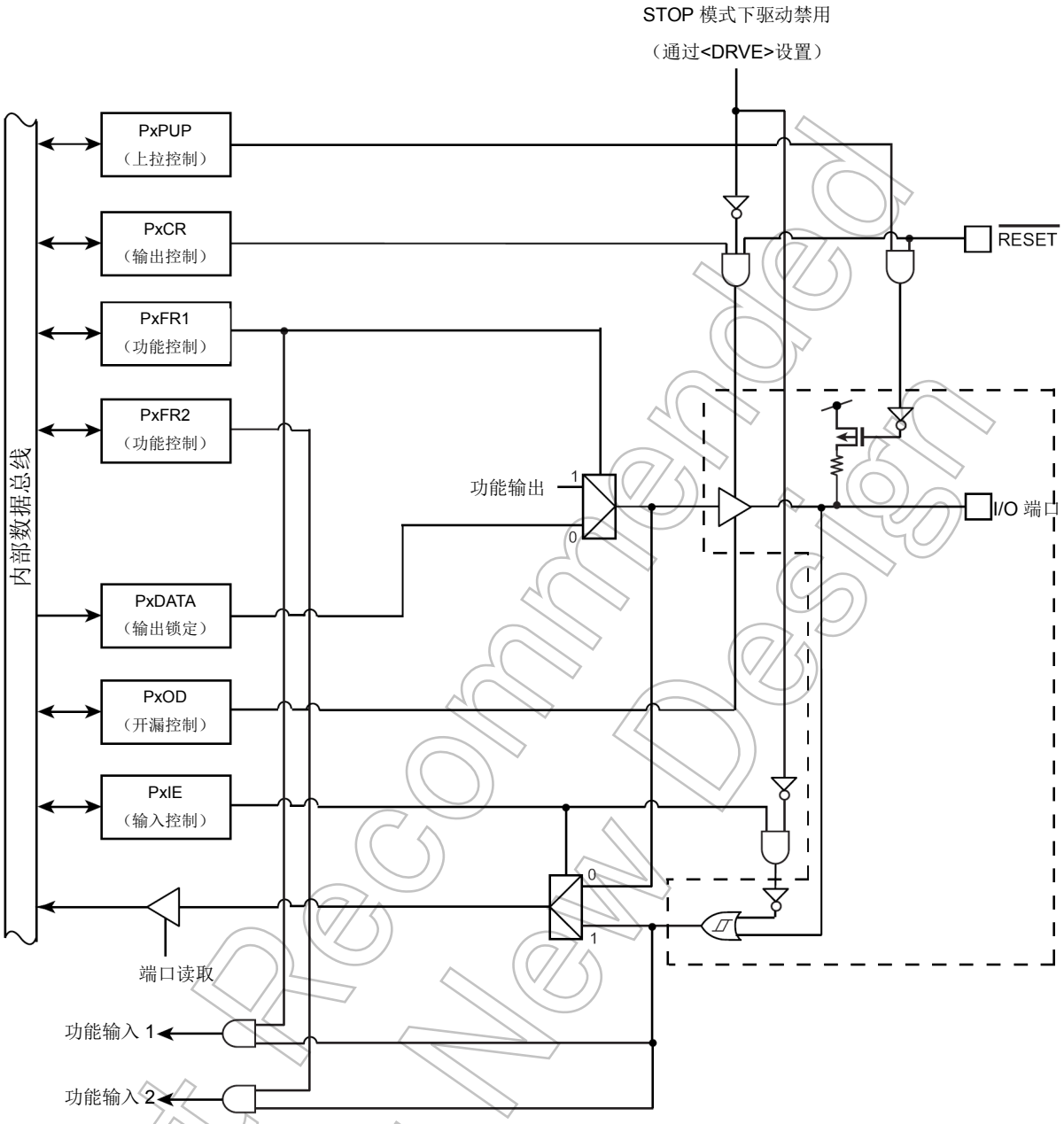


图 8-8 T8 型端口

8.3.10 T9型

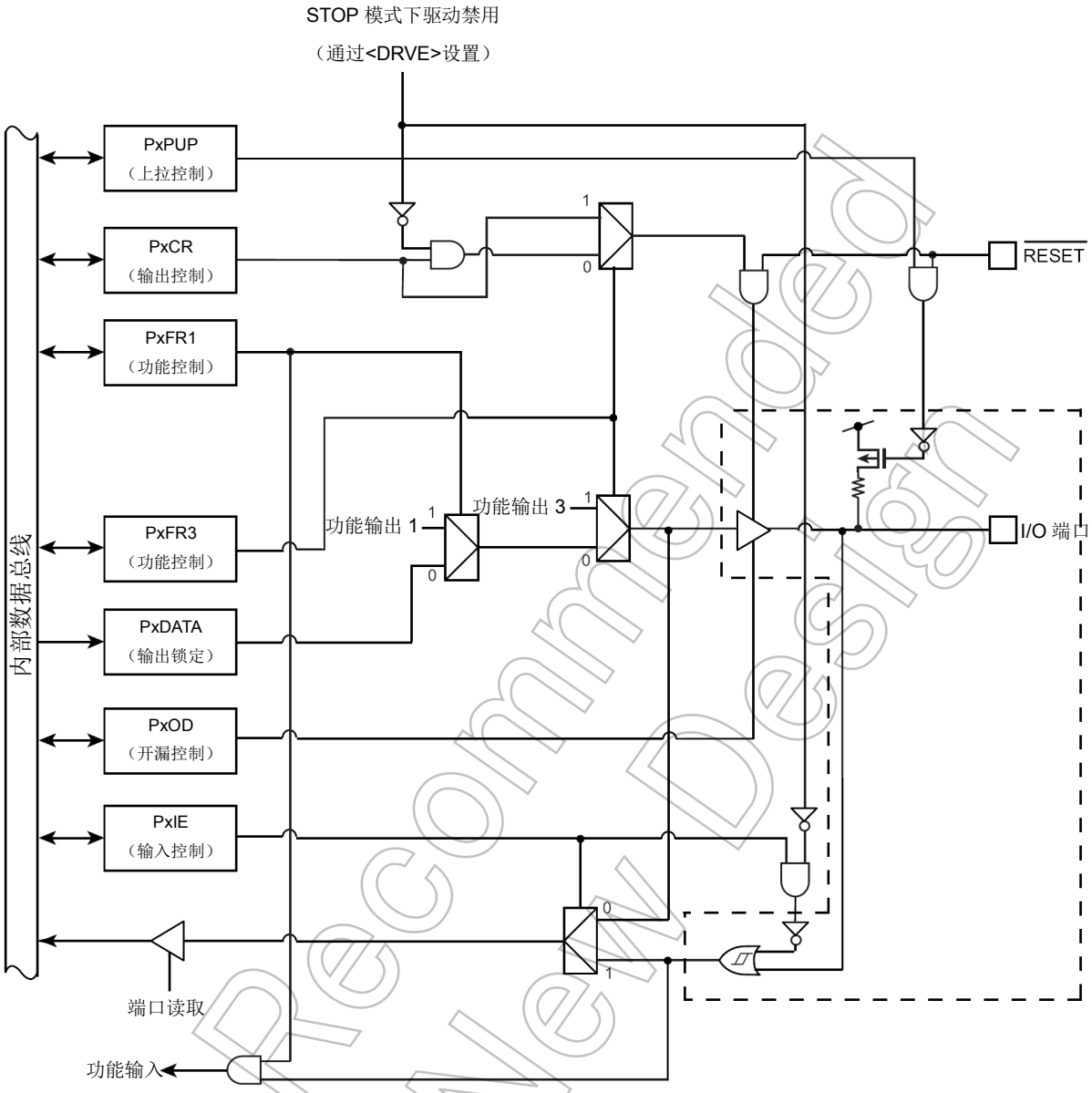


图 8-9 T9 型端口

8.3.11 T10型

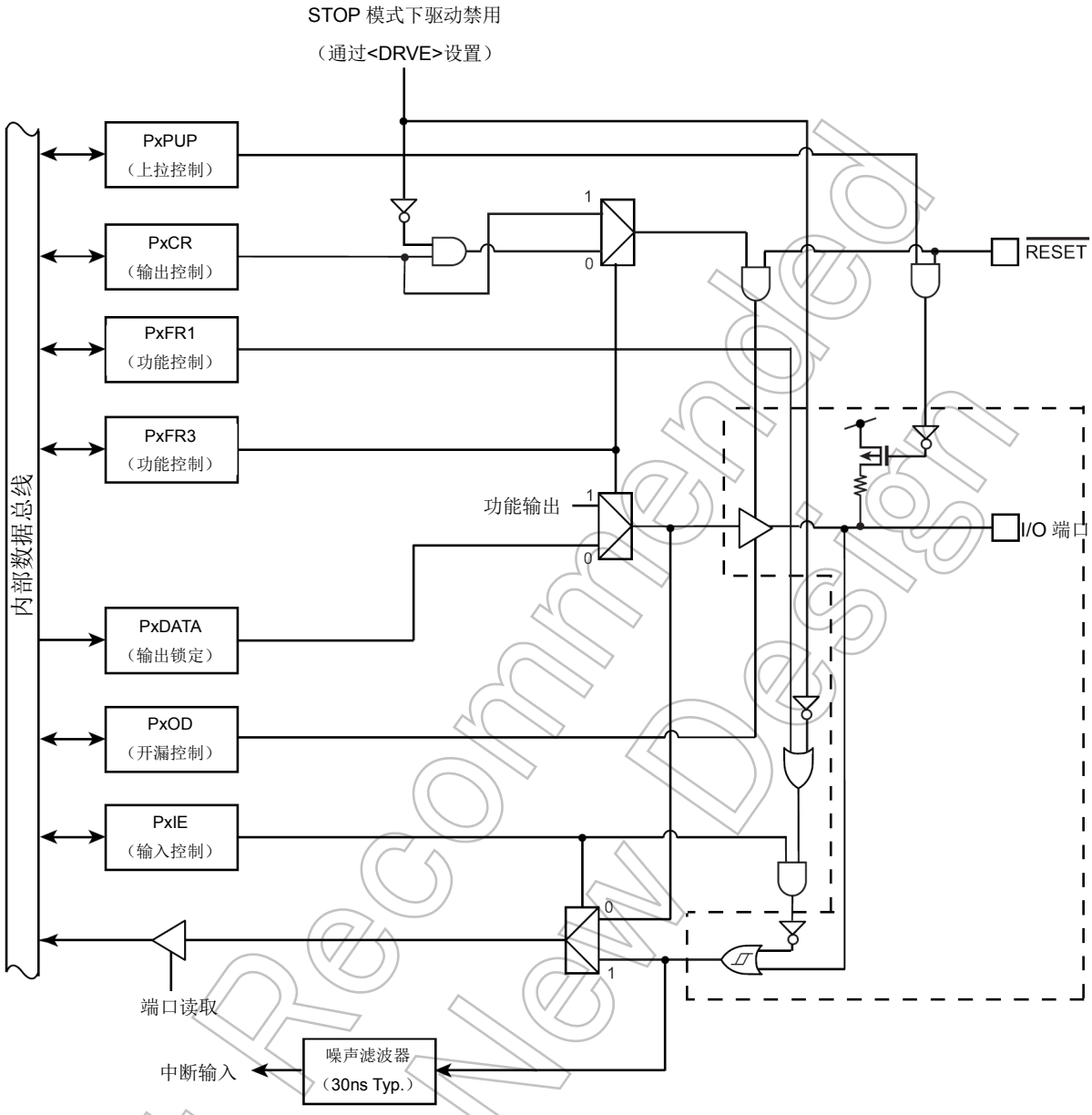


图 8-10 T10 型端口

8.3.12 T11型

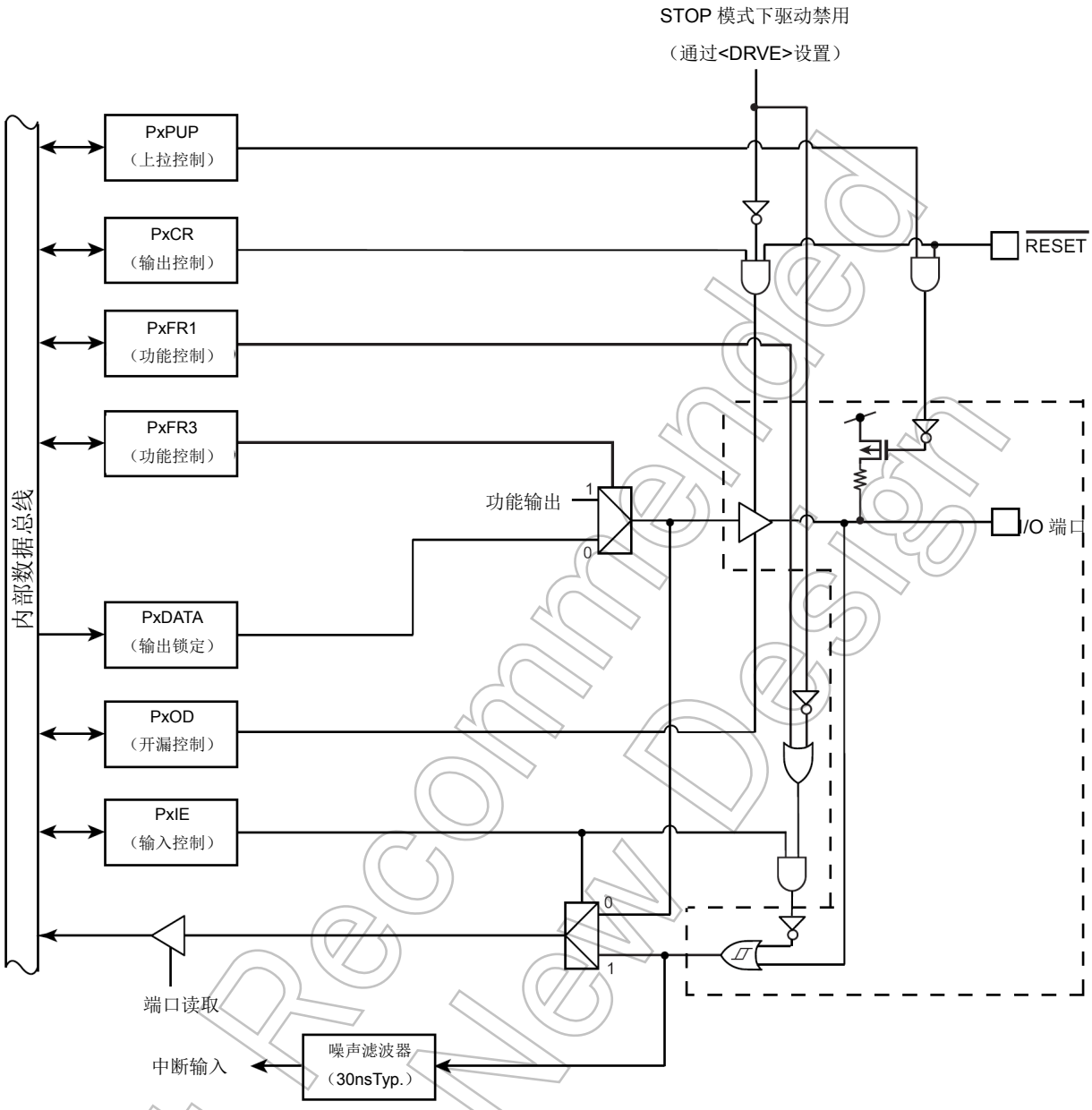


图 8-11 T11 型端口

8.3.13 T12型

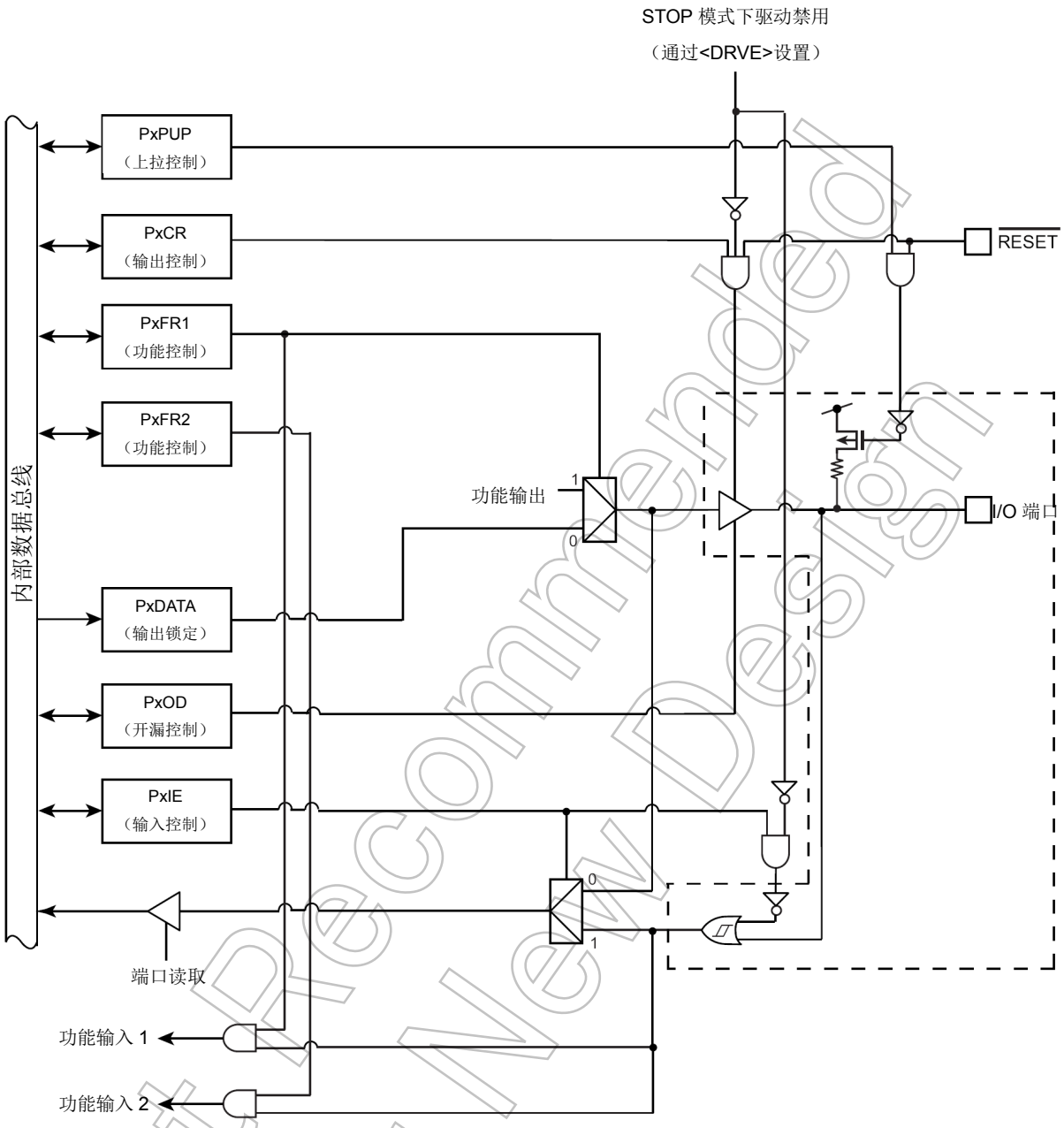


图 8-12 T12 型端口

8.3.14 T13型

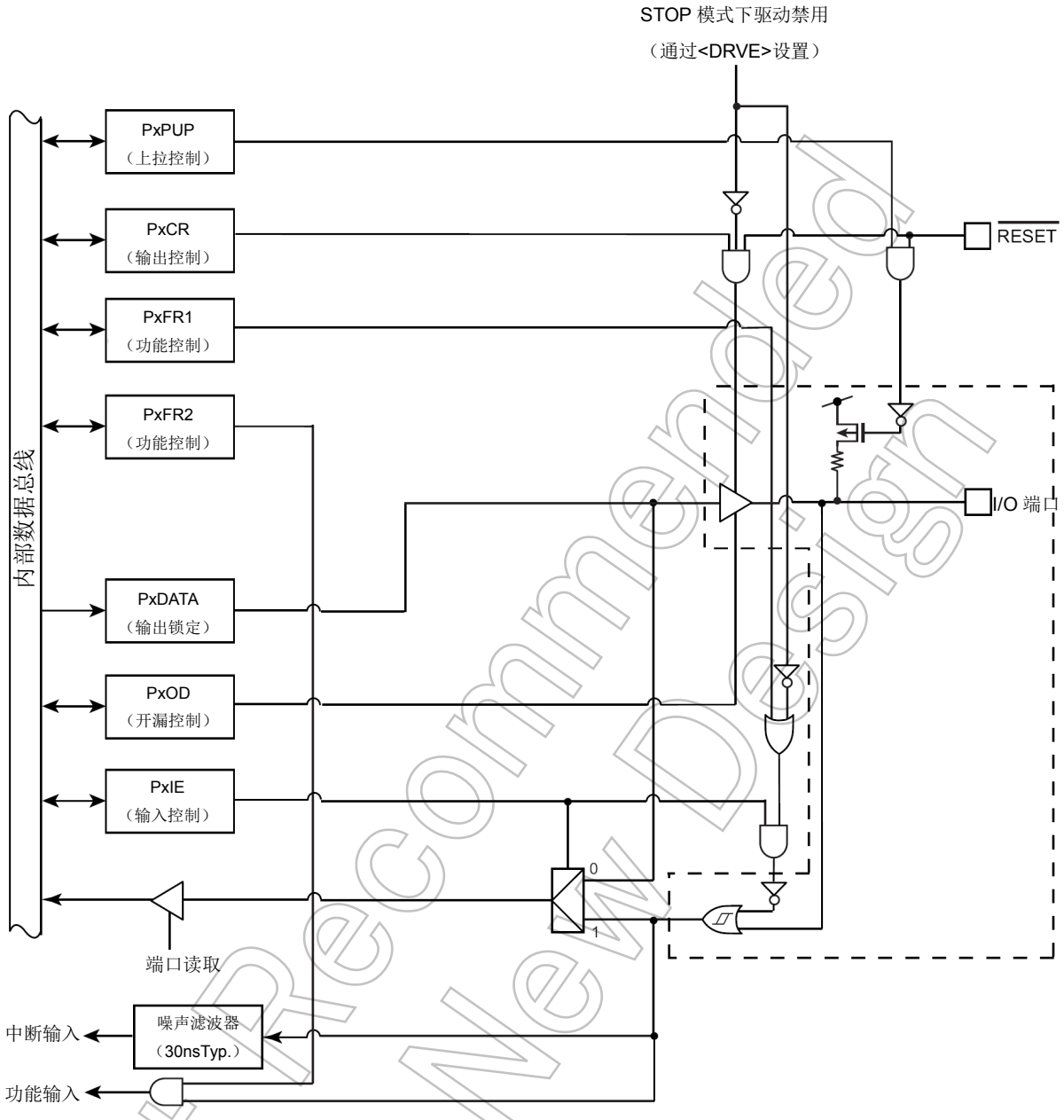


图 8-13 T13 型端口

8.3.15 T14型

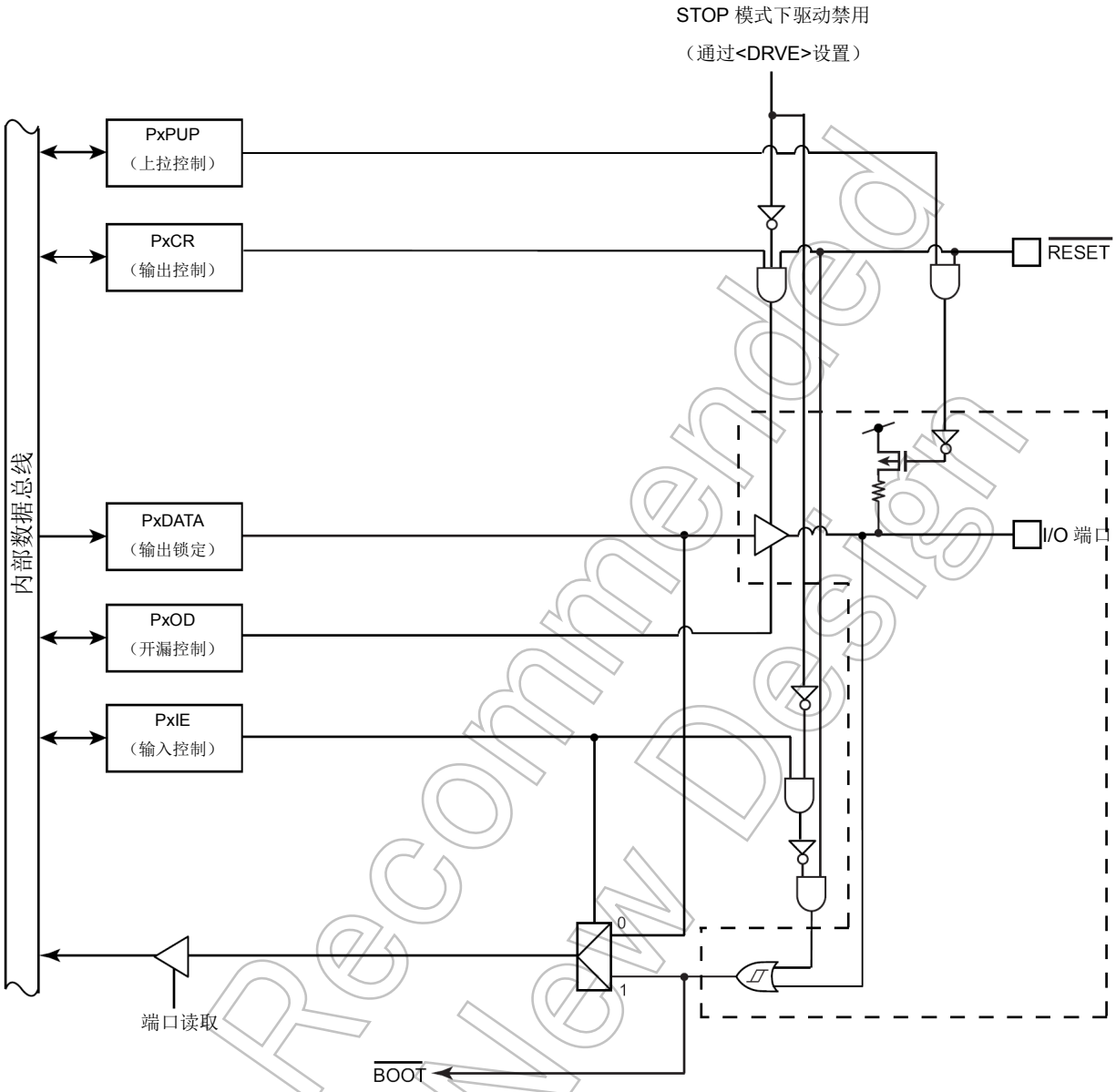


图 8-14 T14 型端口

8.3.16 T15型

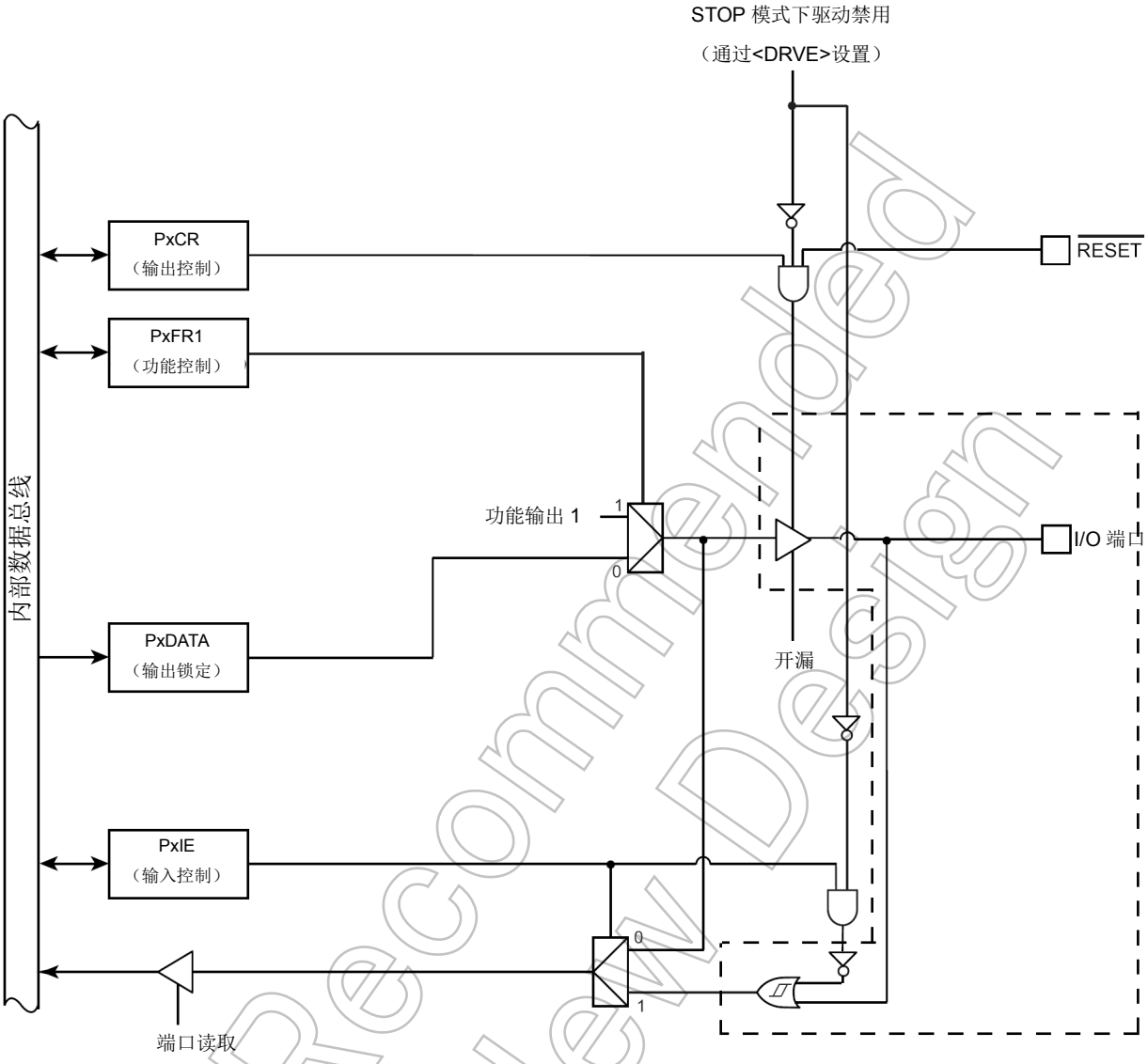


图 8-15 T15 型端口

8.3.17 T16型

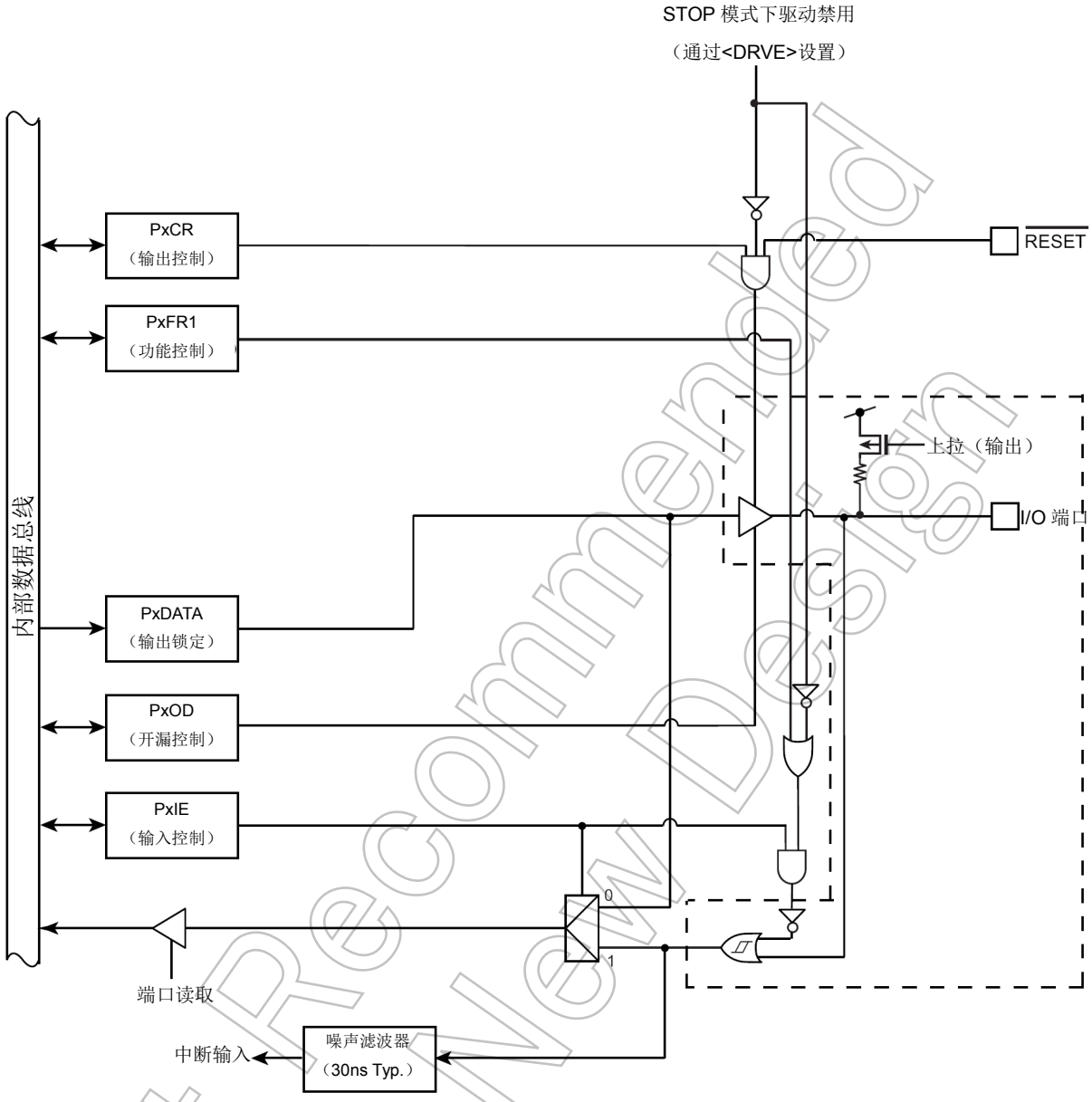


图 8-16 T16 型端口

8.3.18 T17型

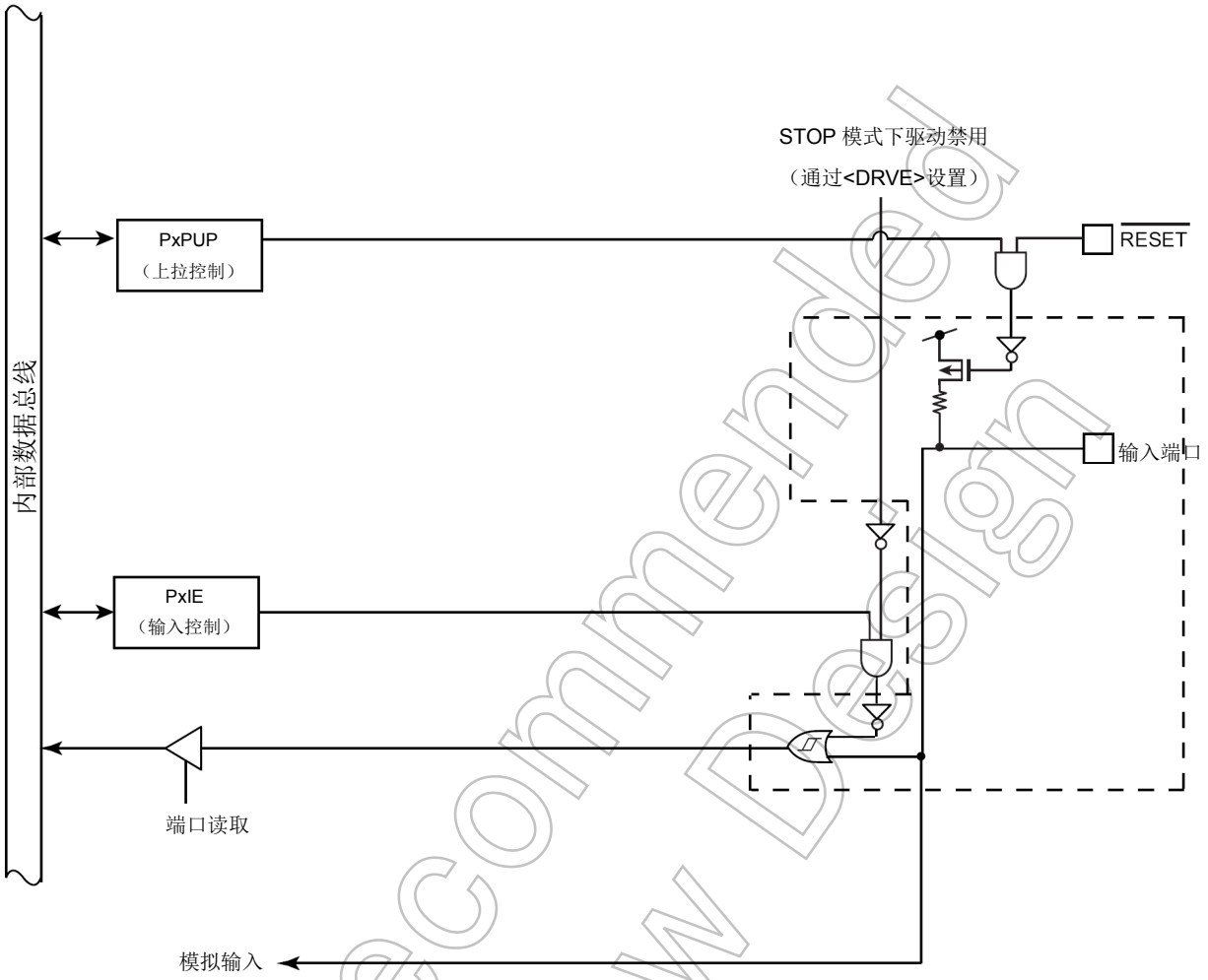


图 8-17 T17 型端口

8.3.19 T18型

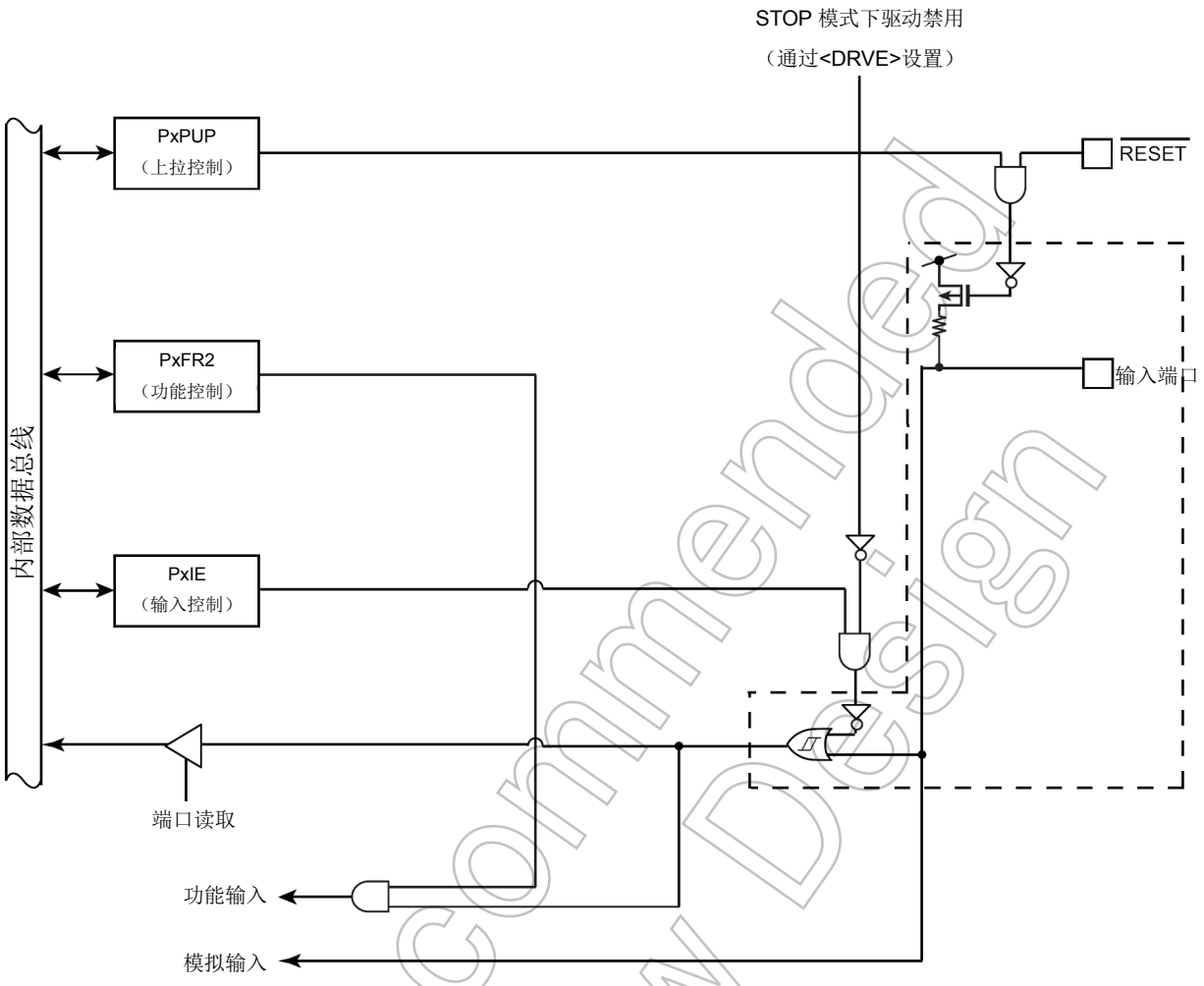


图 8-18 T18 型端口

8.3.20 T19型

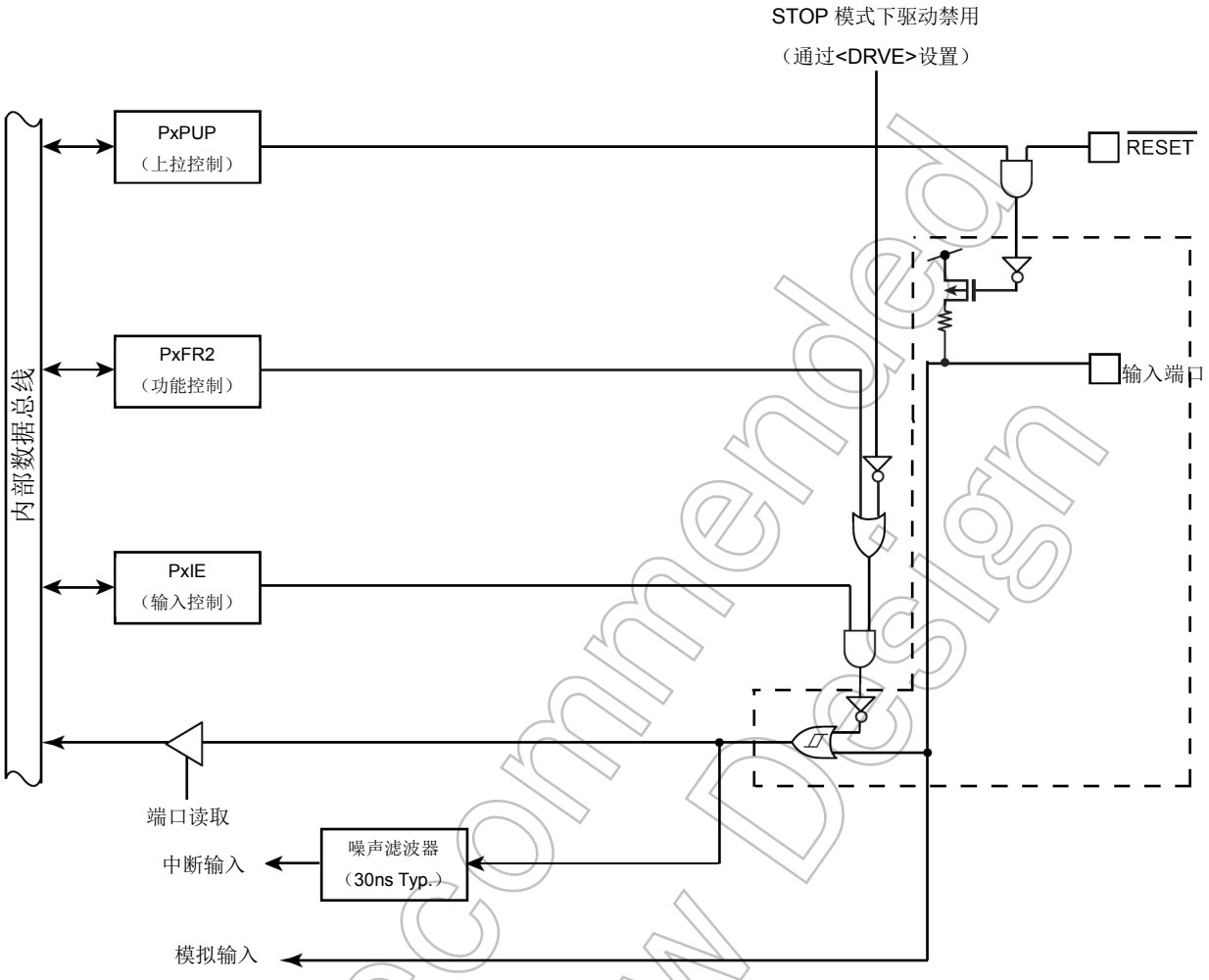


图 8-19 T19 型端口

8.3.21 T20型

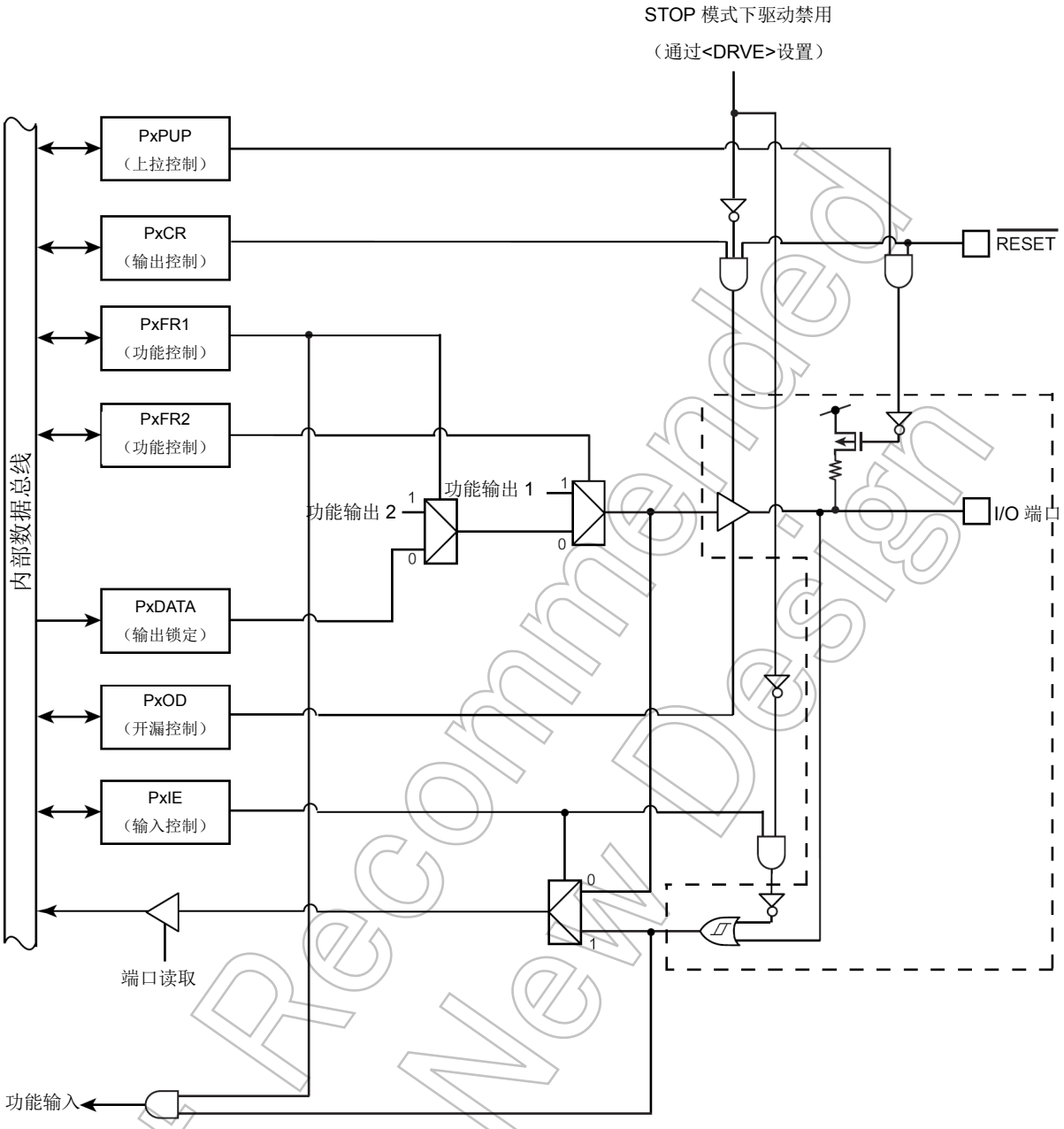


图 8-20 T20 型端口

8.3.22 T21型

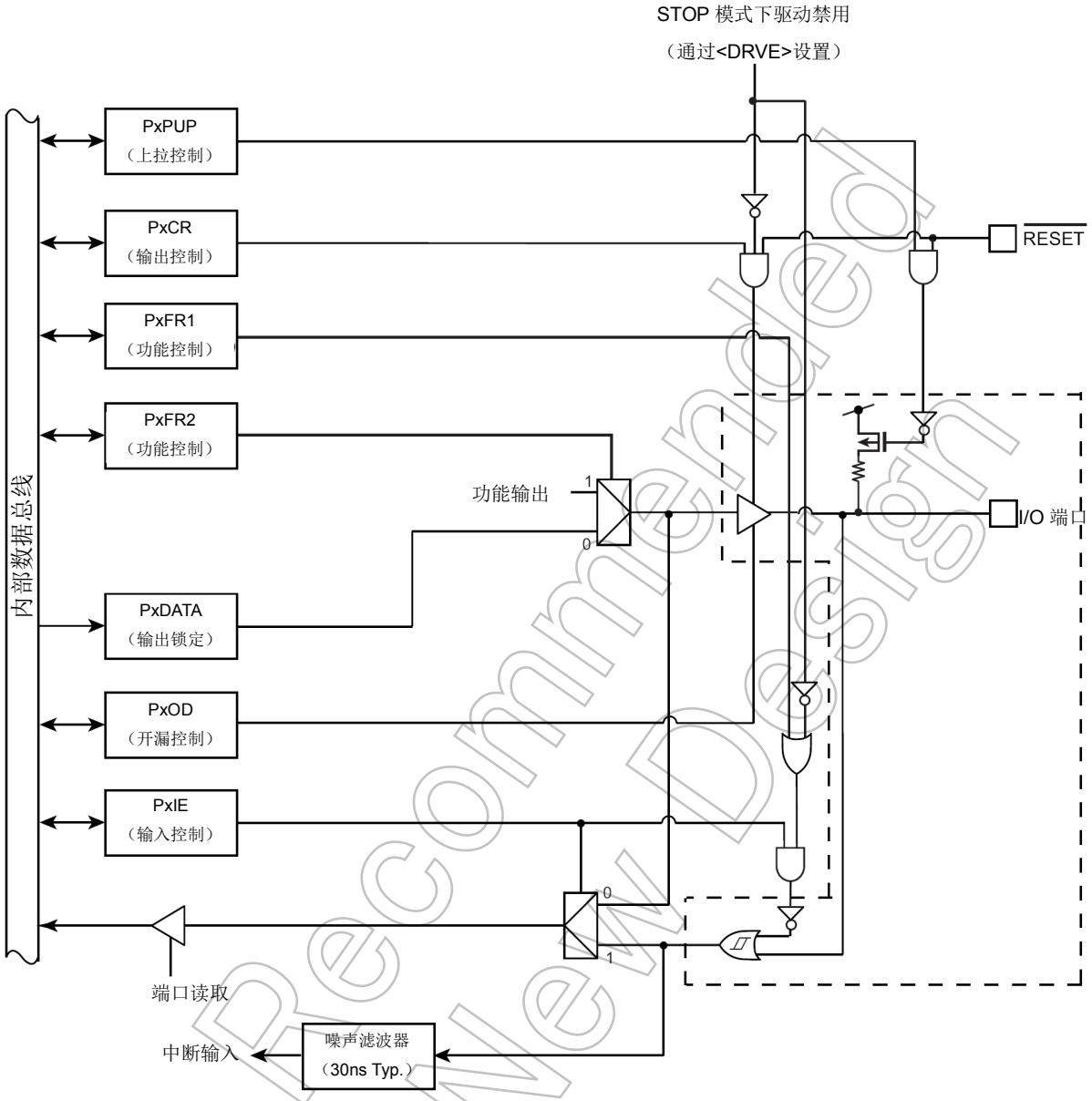


图 8-21 T21 型端口

8.3.23 T22型

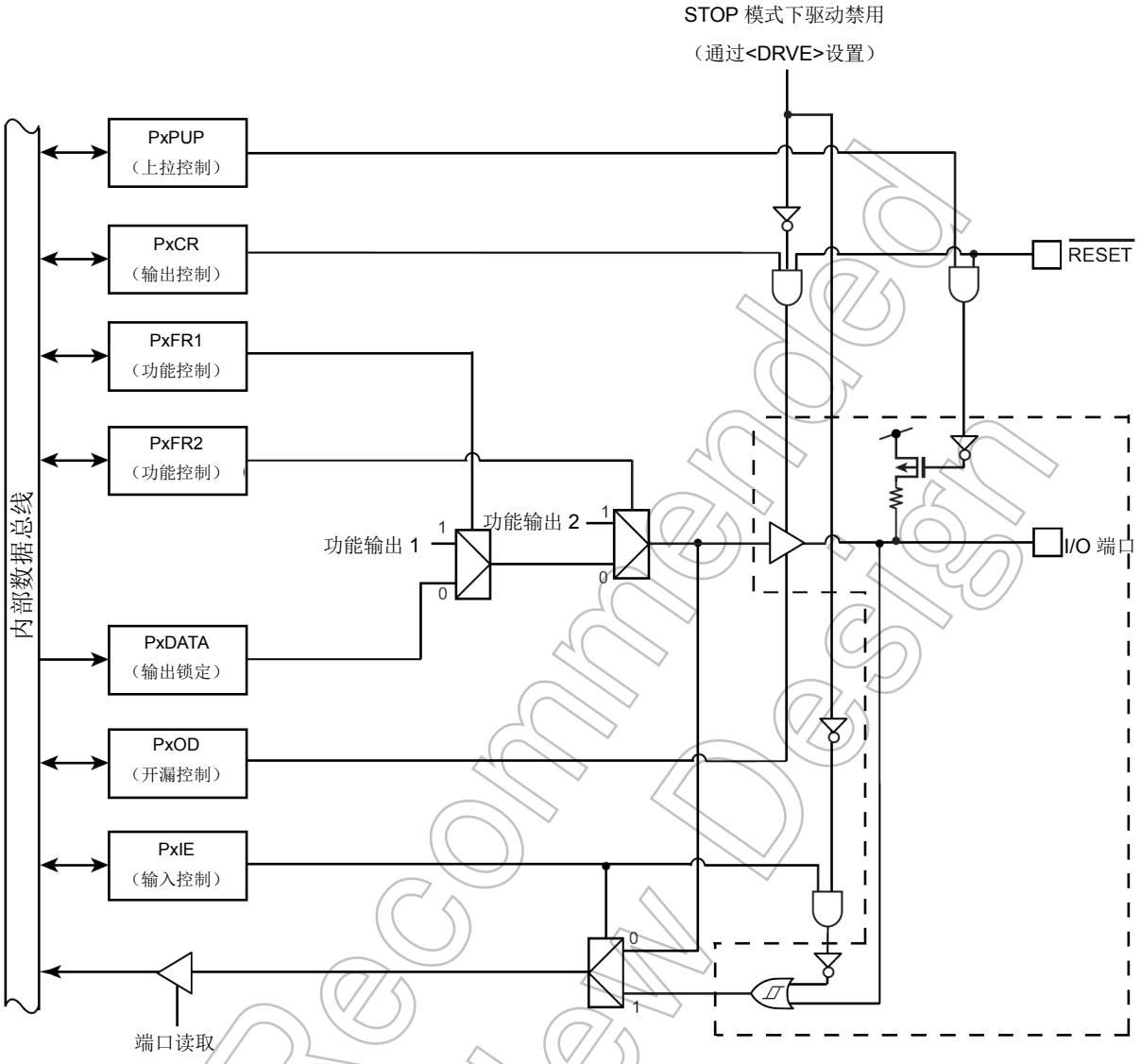


图 8-22 T22 型端口

8.3.24 T23型

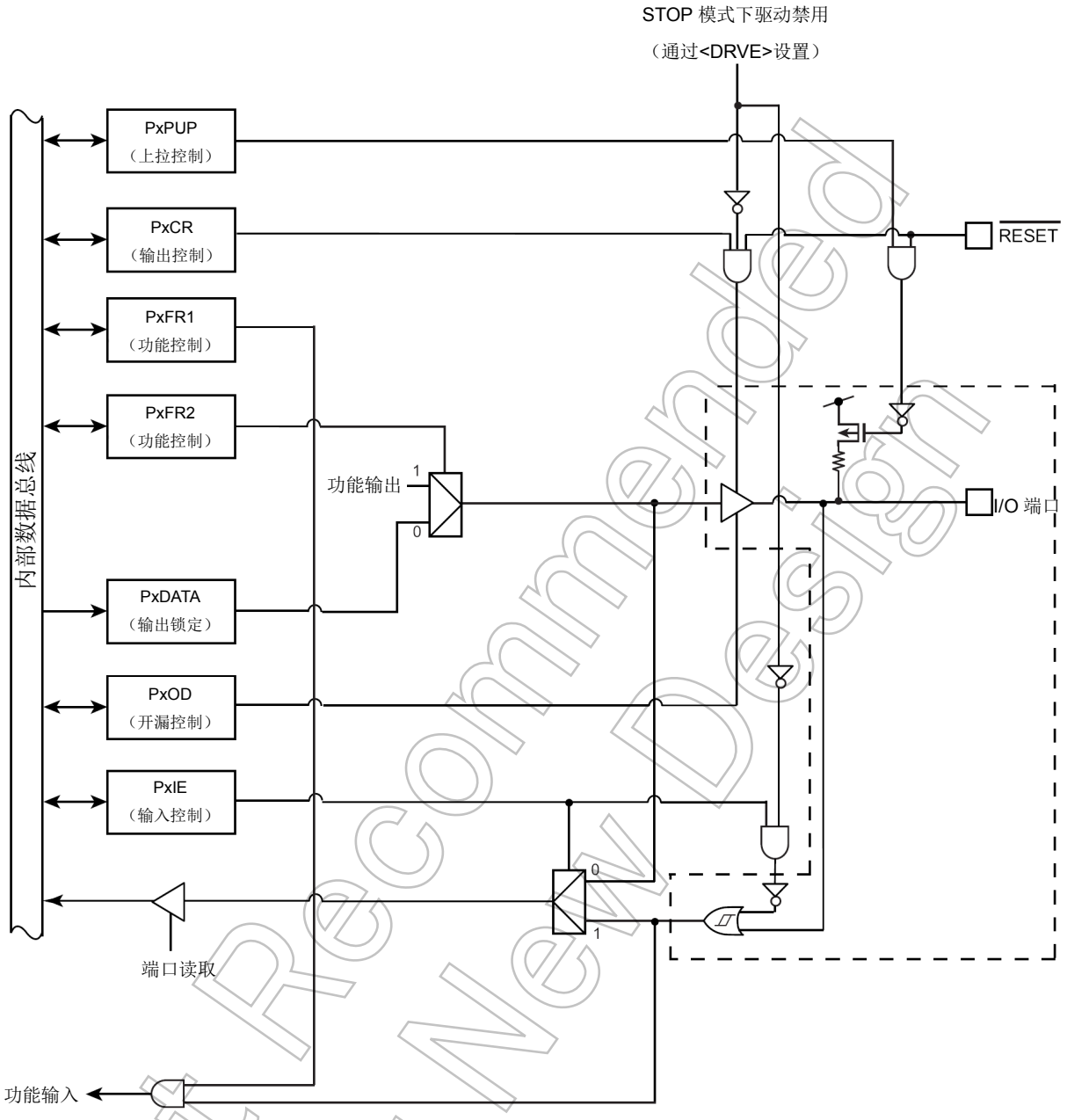


图 8-23 T23 型端口

8.3.25 T24型

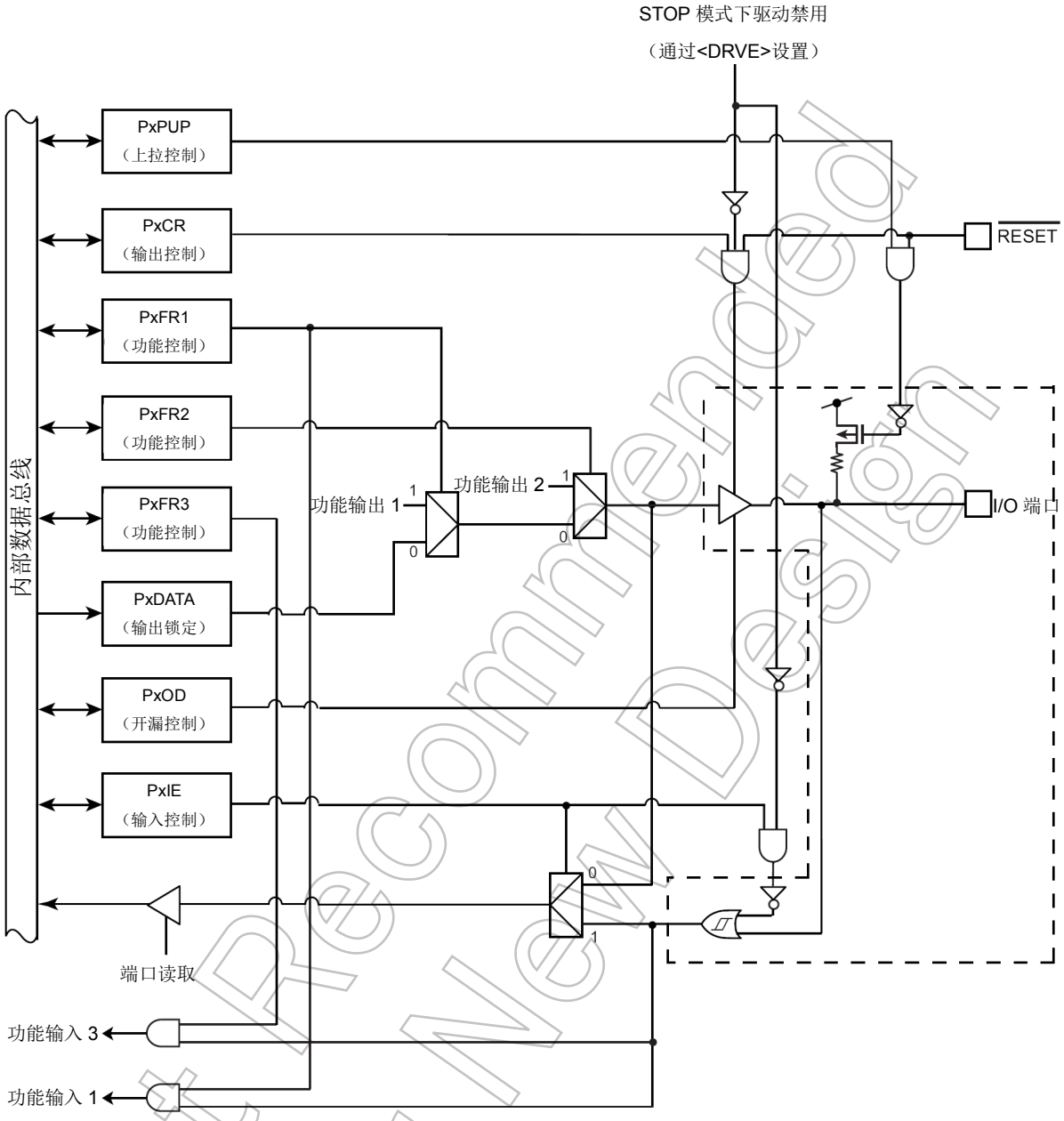


图 8-24 T24 型端口

8.3.26 T25型

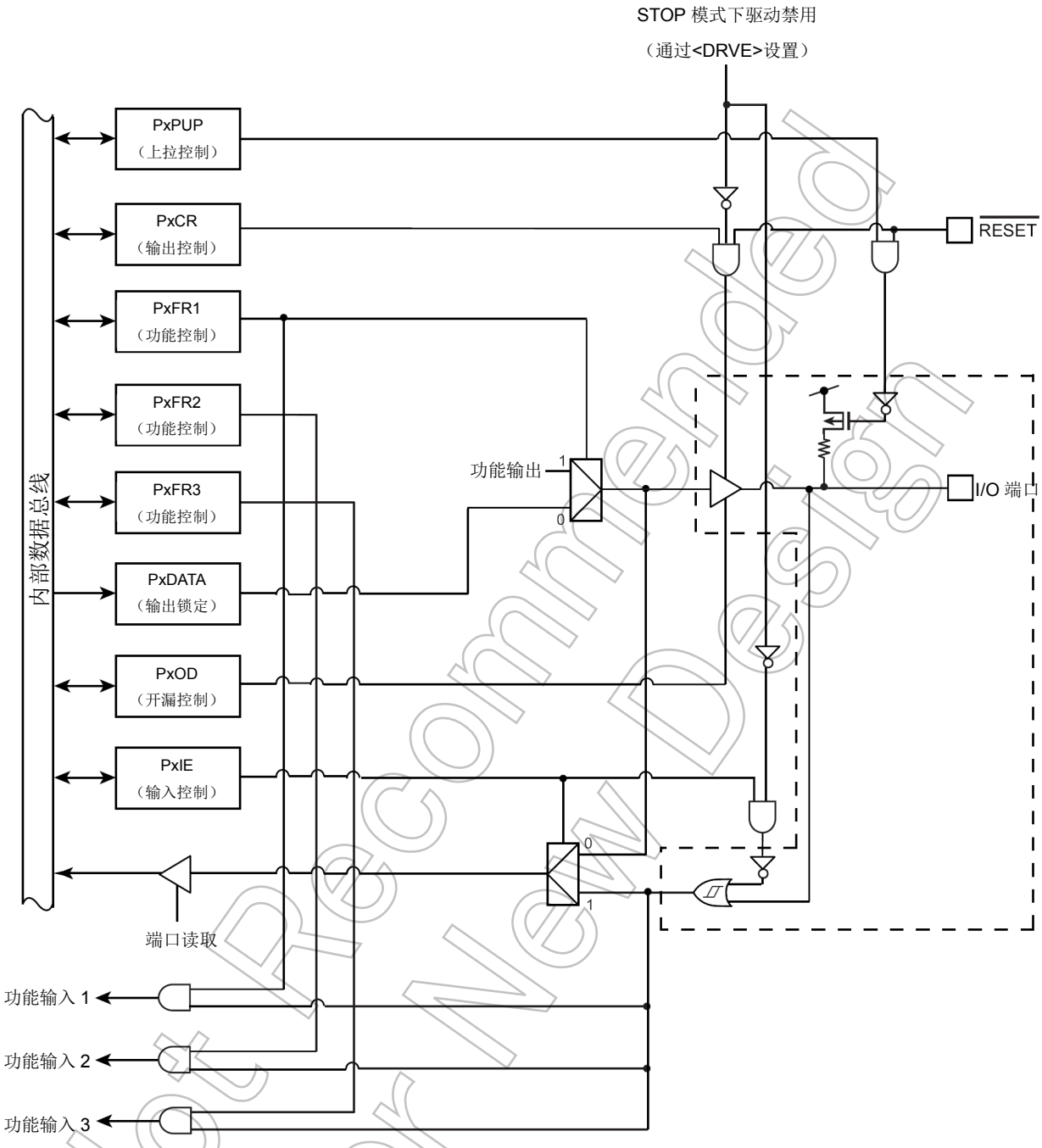


图 8-25 T25 型端口

8.3.27 T26型

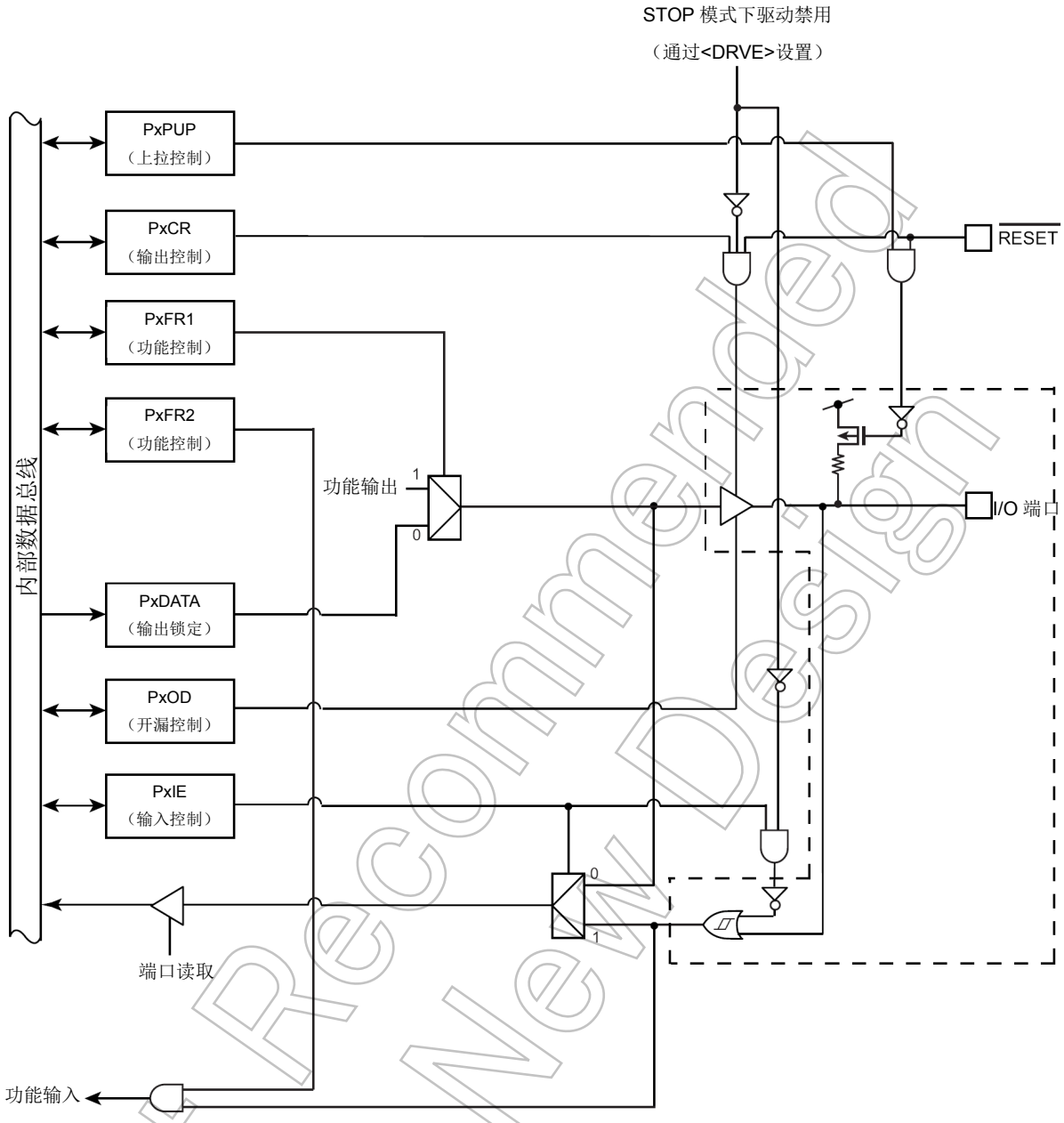


图 8-26 T26 型端口

8.3.28 T27型

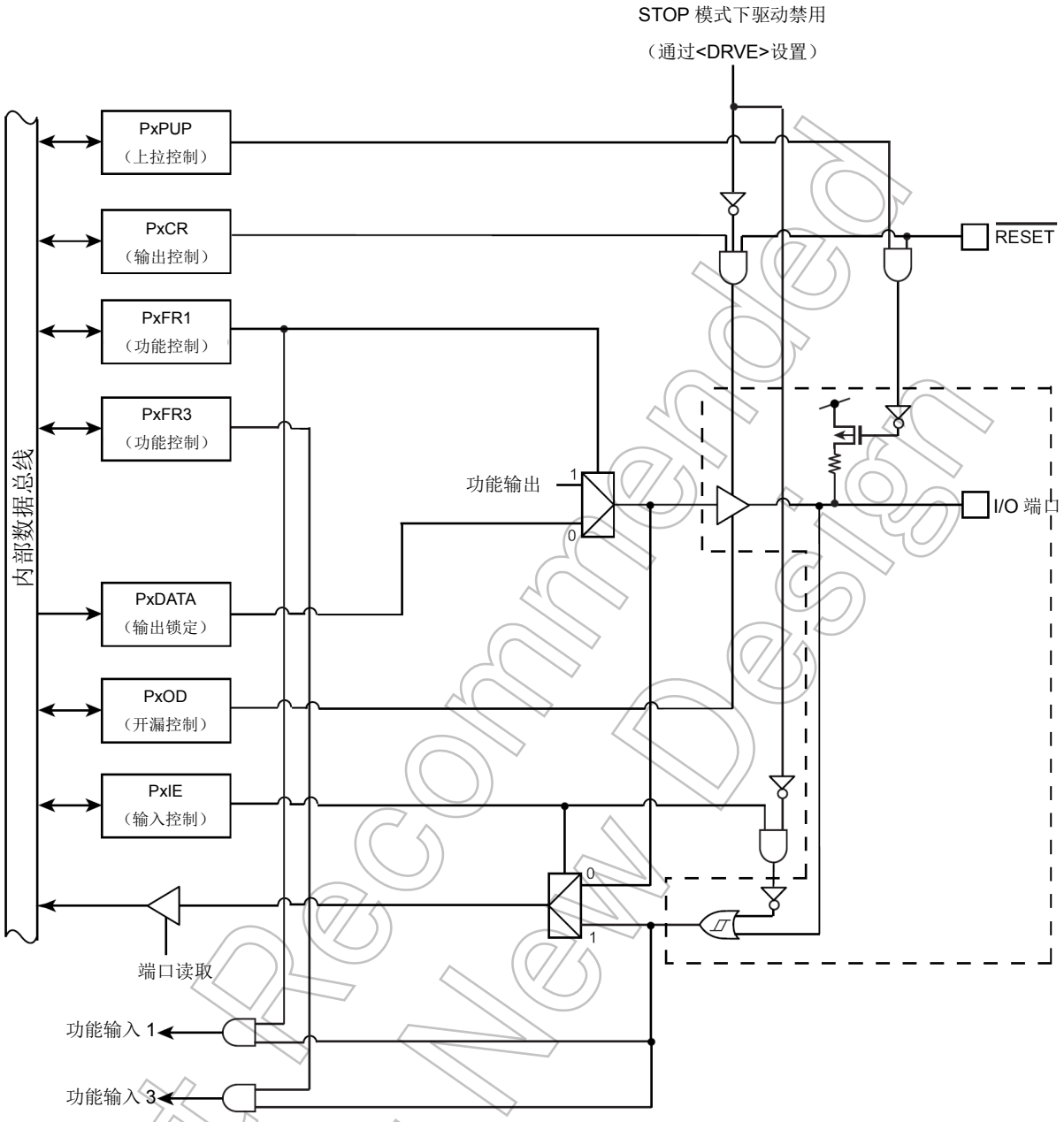


图 8-27 T27 型端口

8.3.29 T28型

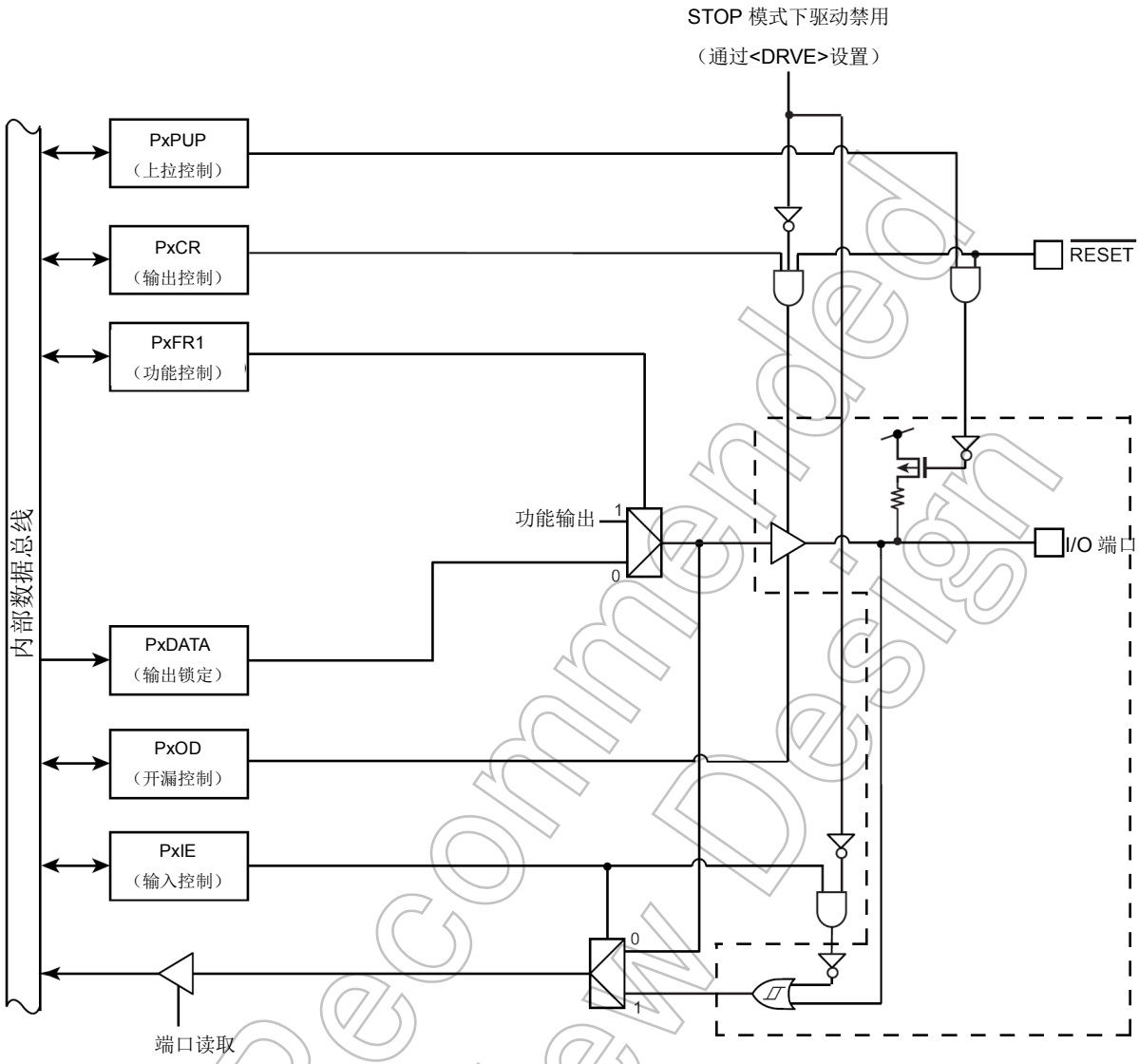


图 8-28 T28 型端口

8.3.30 T29型

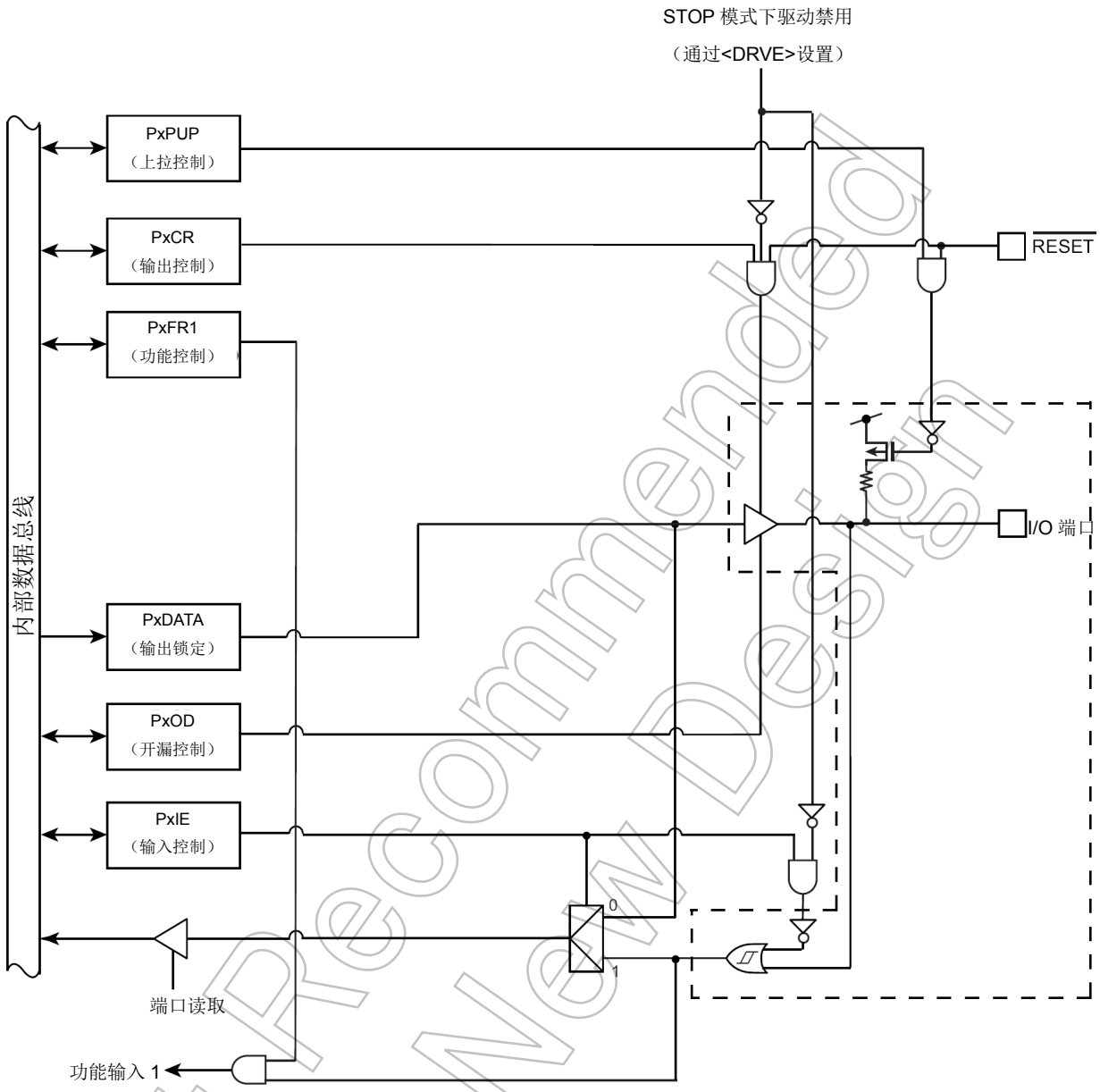


图 8-29 T29 型端口

8.3.31 T30型

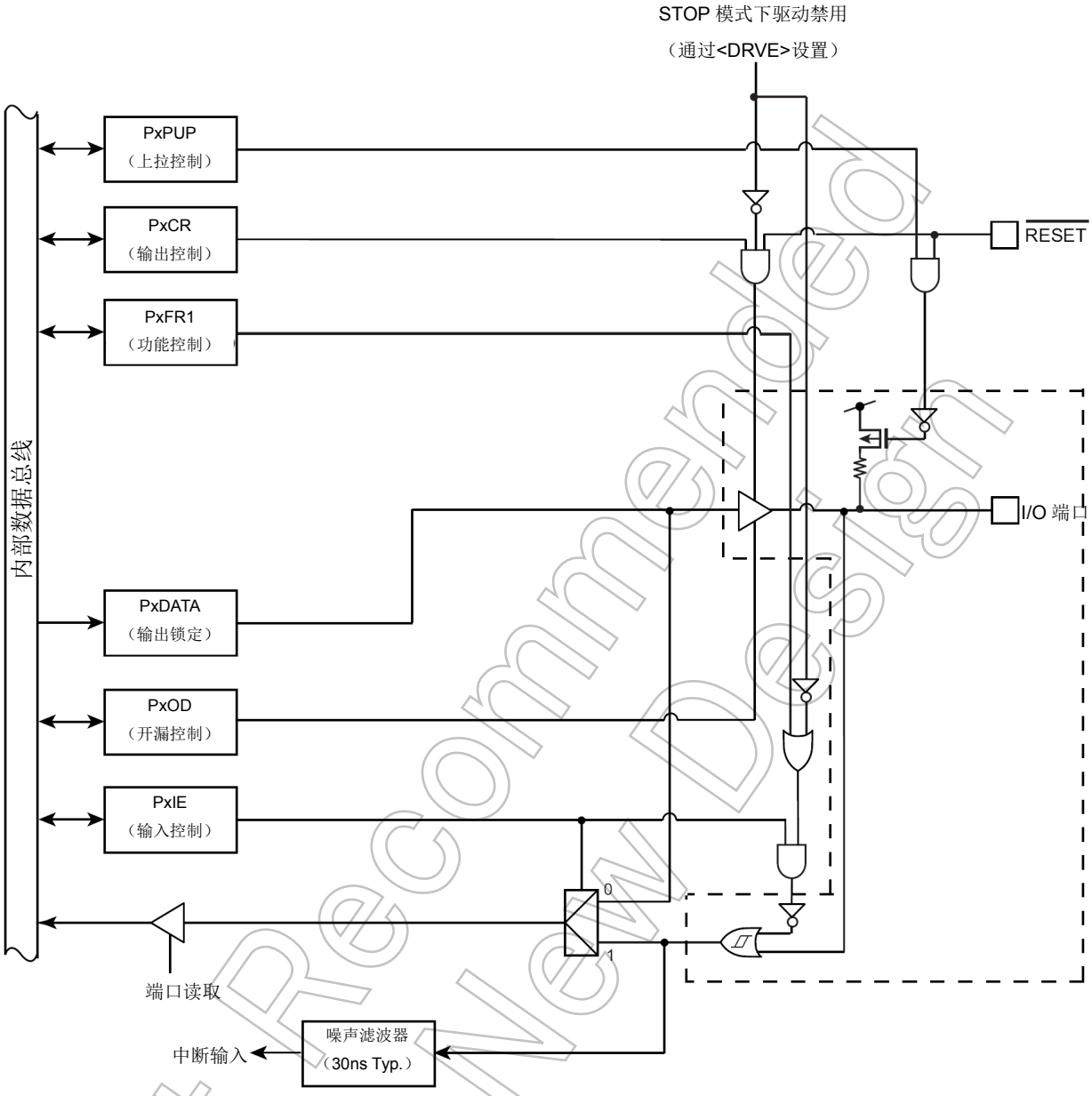


图 8-30 T30 型端口

8.3.32 T31型

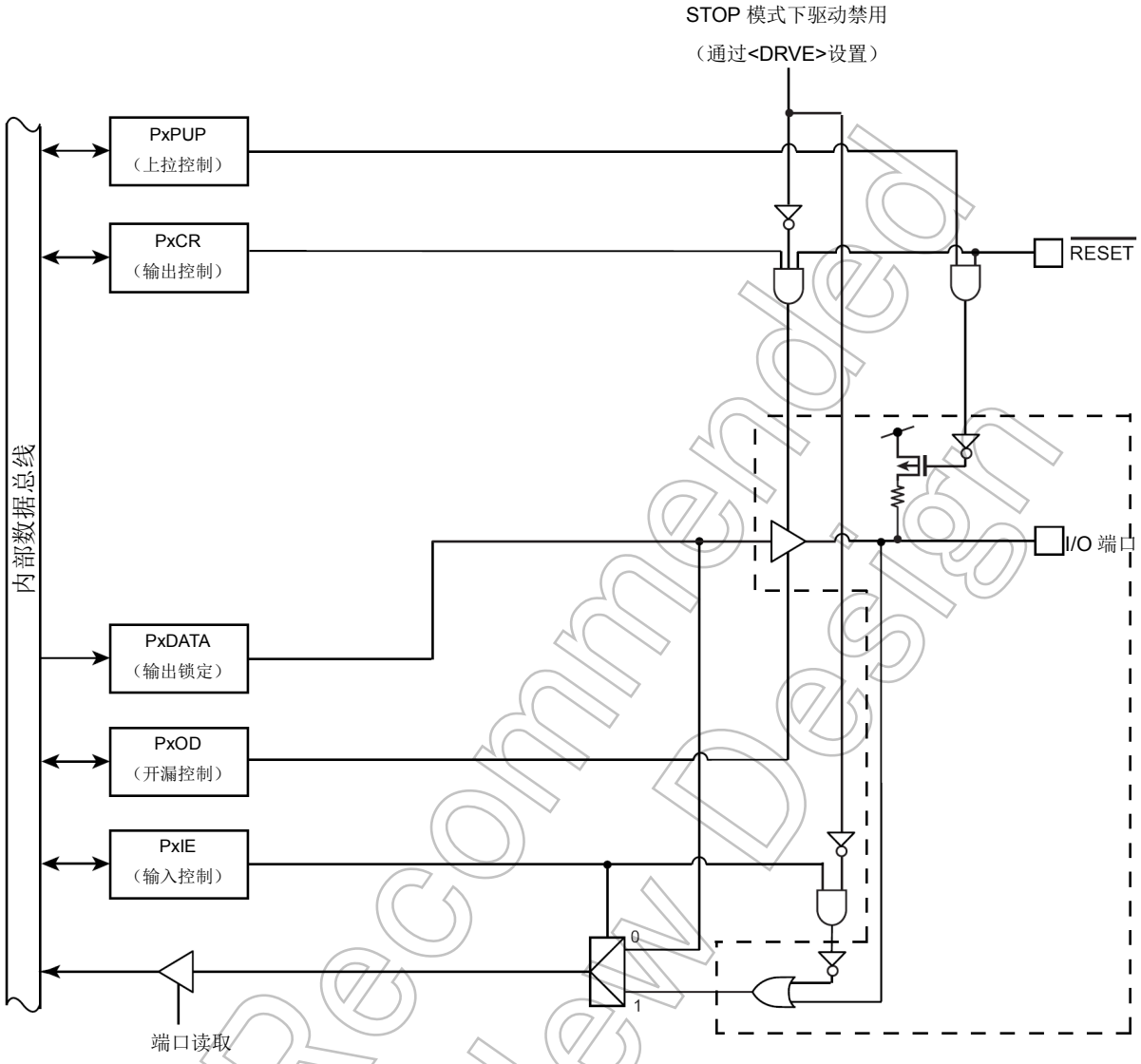


图 8-31 T31 型端口

8.3.33 T32型

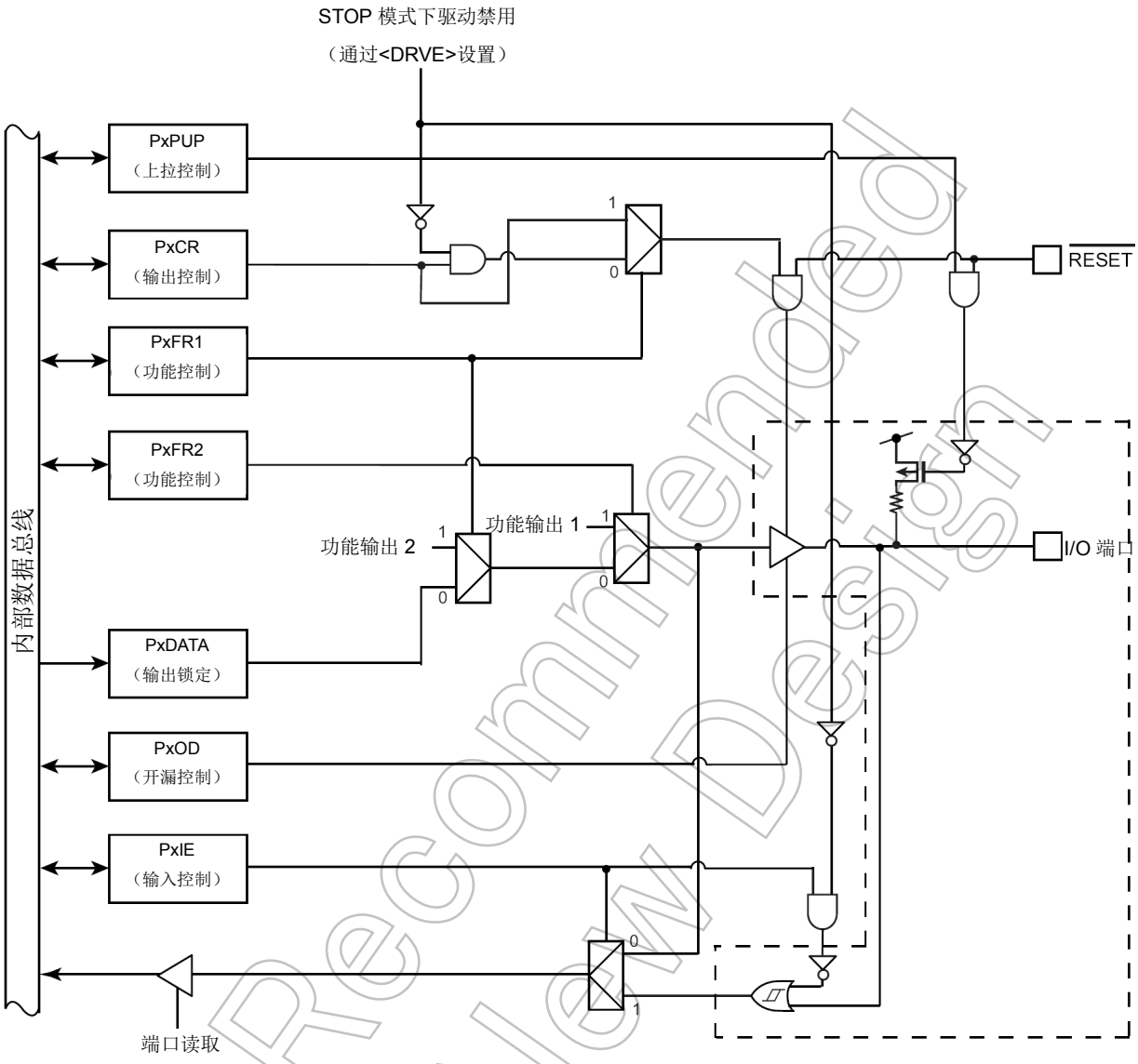


图 8-32 T32 型端口

8.3.34 T33型

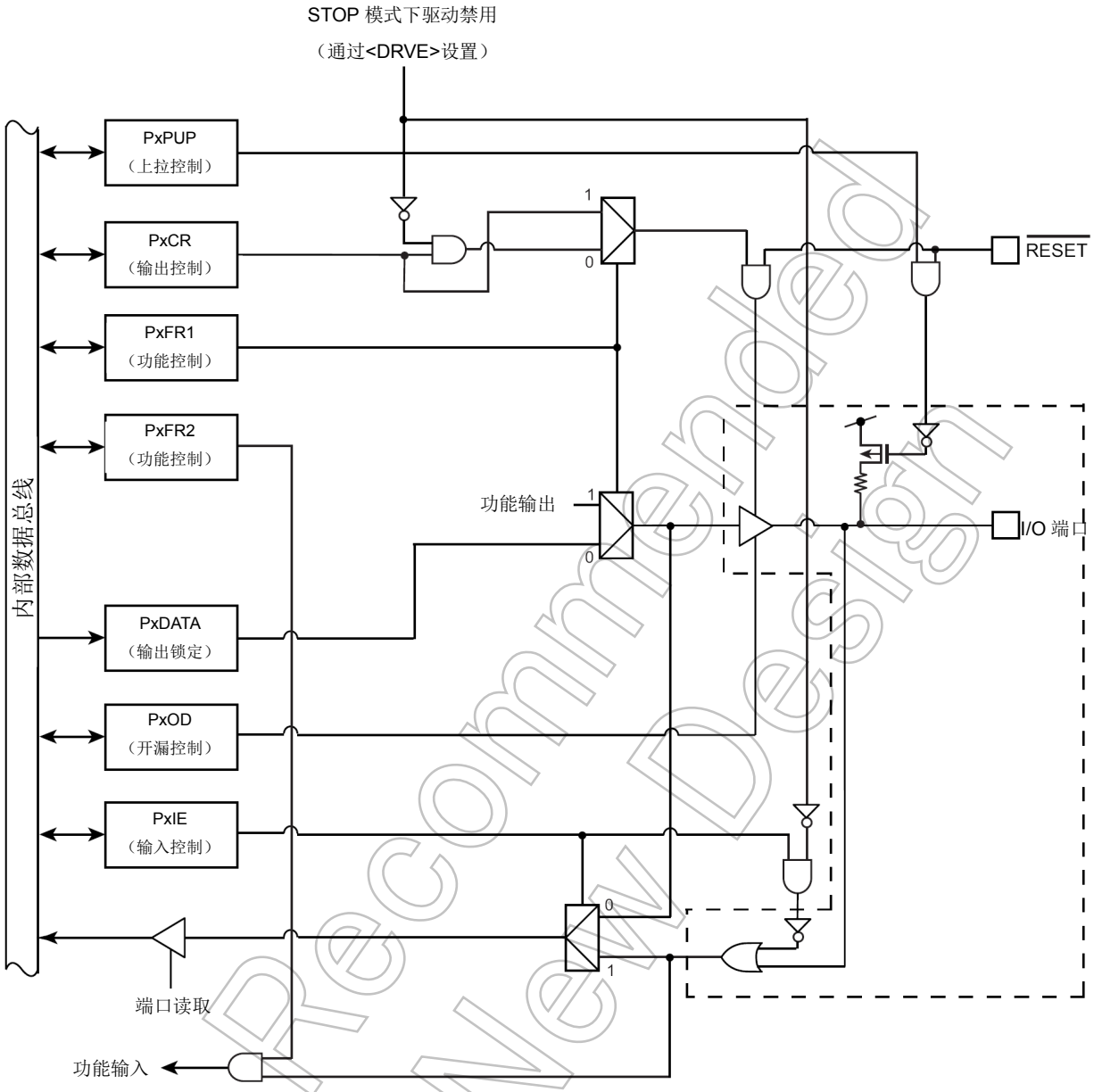


图 8-33 T33 型端口

8.3.35 T34型

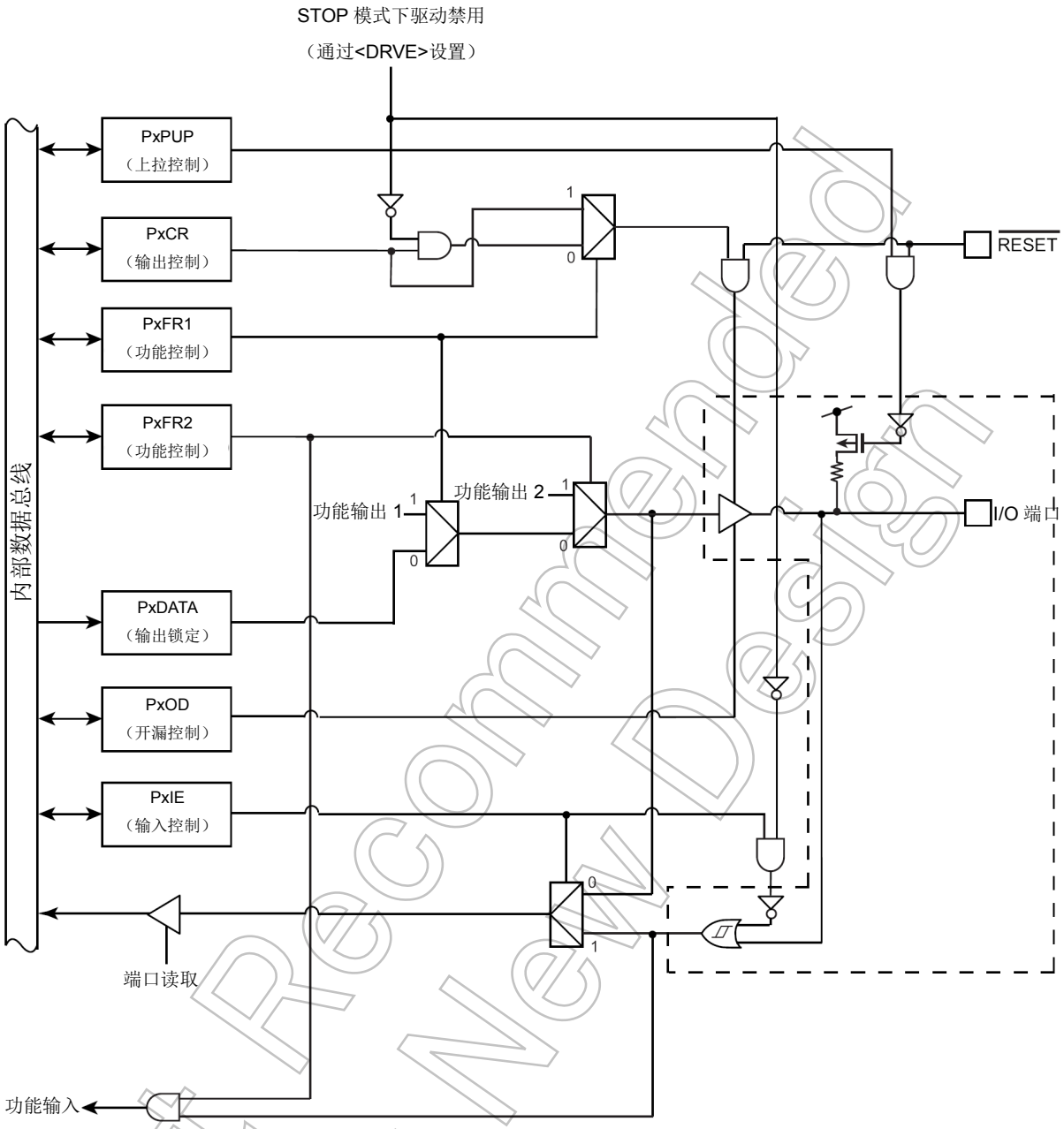


图 8-34 T34 型端口

8.3.36 T35型

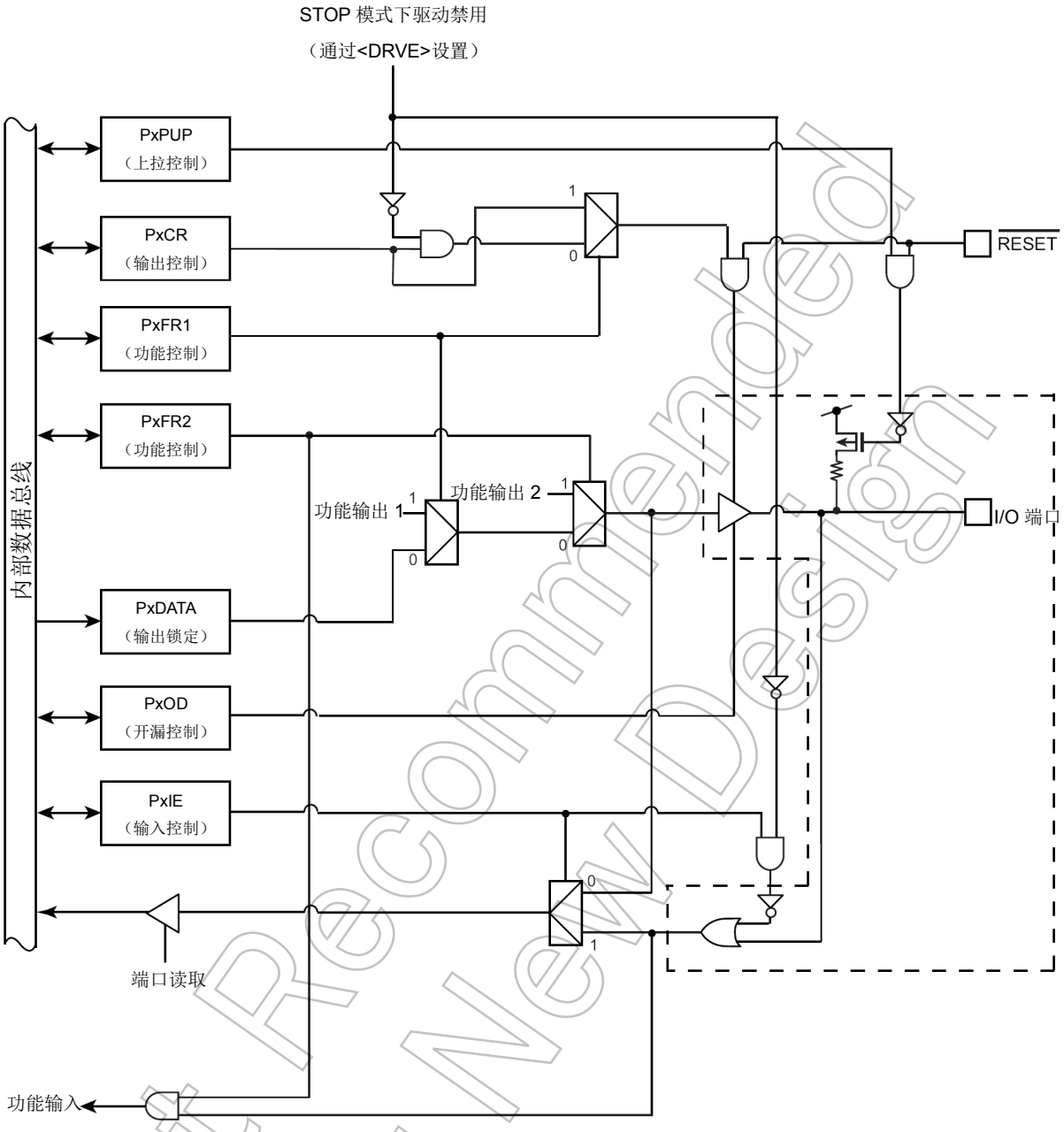


图 8-35 T35 型端口

8.3.37 T36型

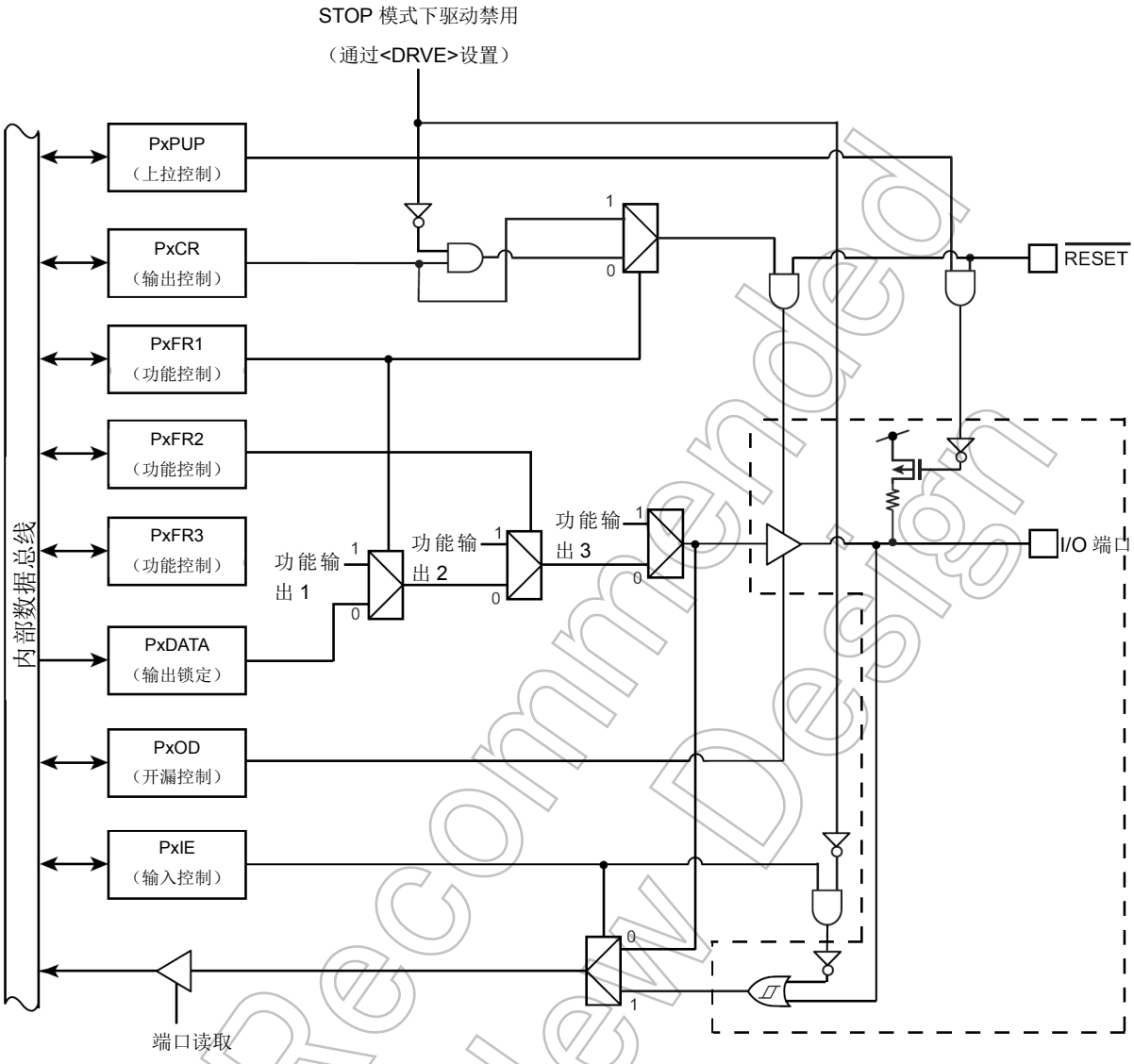


图 8-36 T36 型端口

8.3.38 T37型

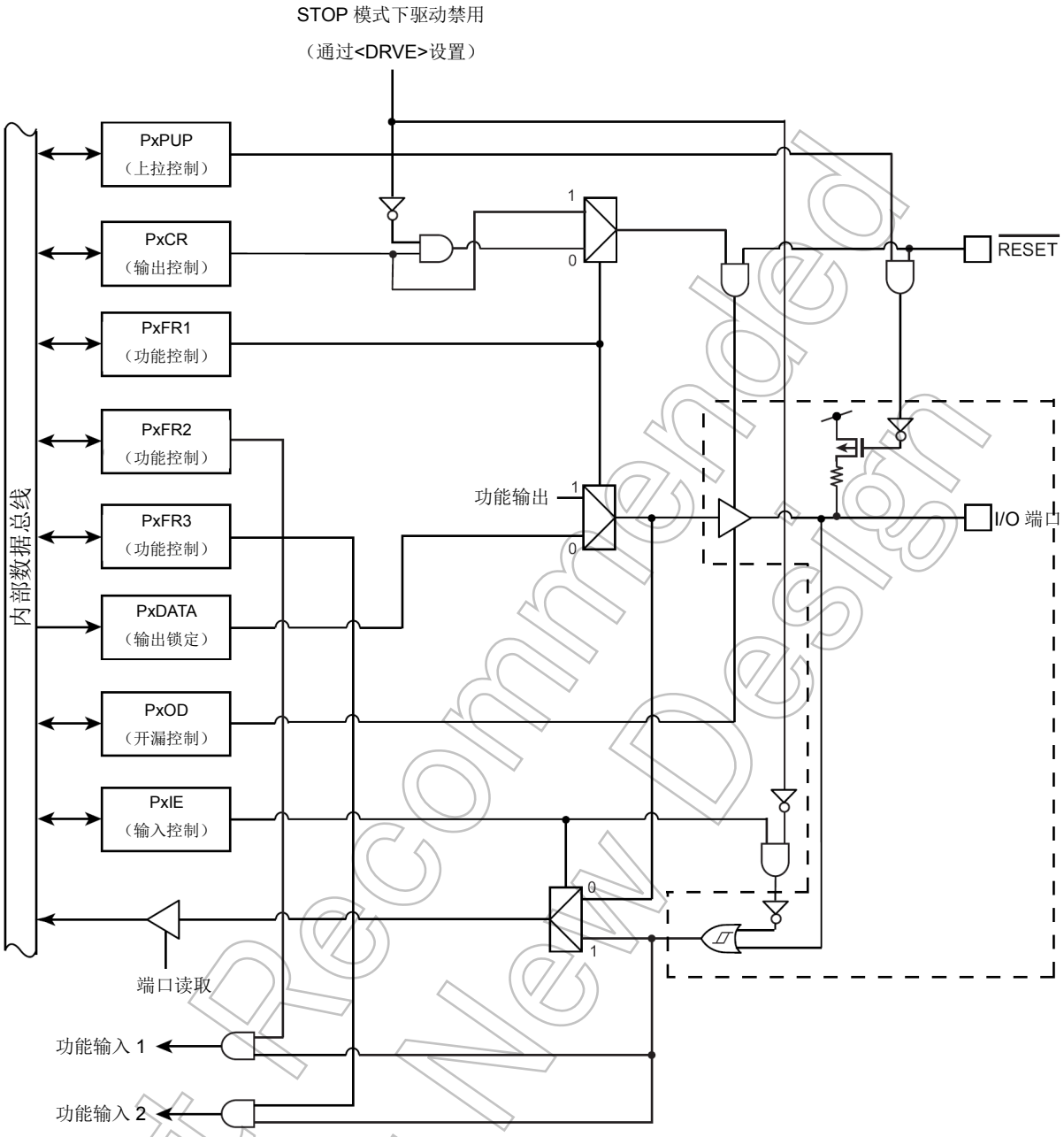


图 8-37 T37 型端口

8.3.39 T38型

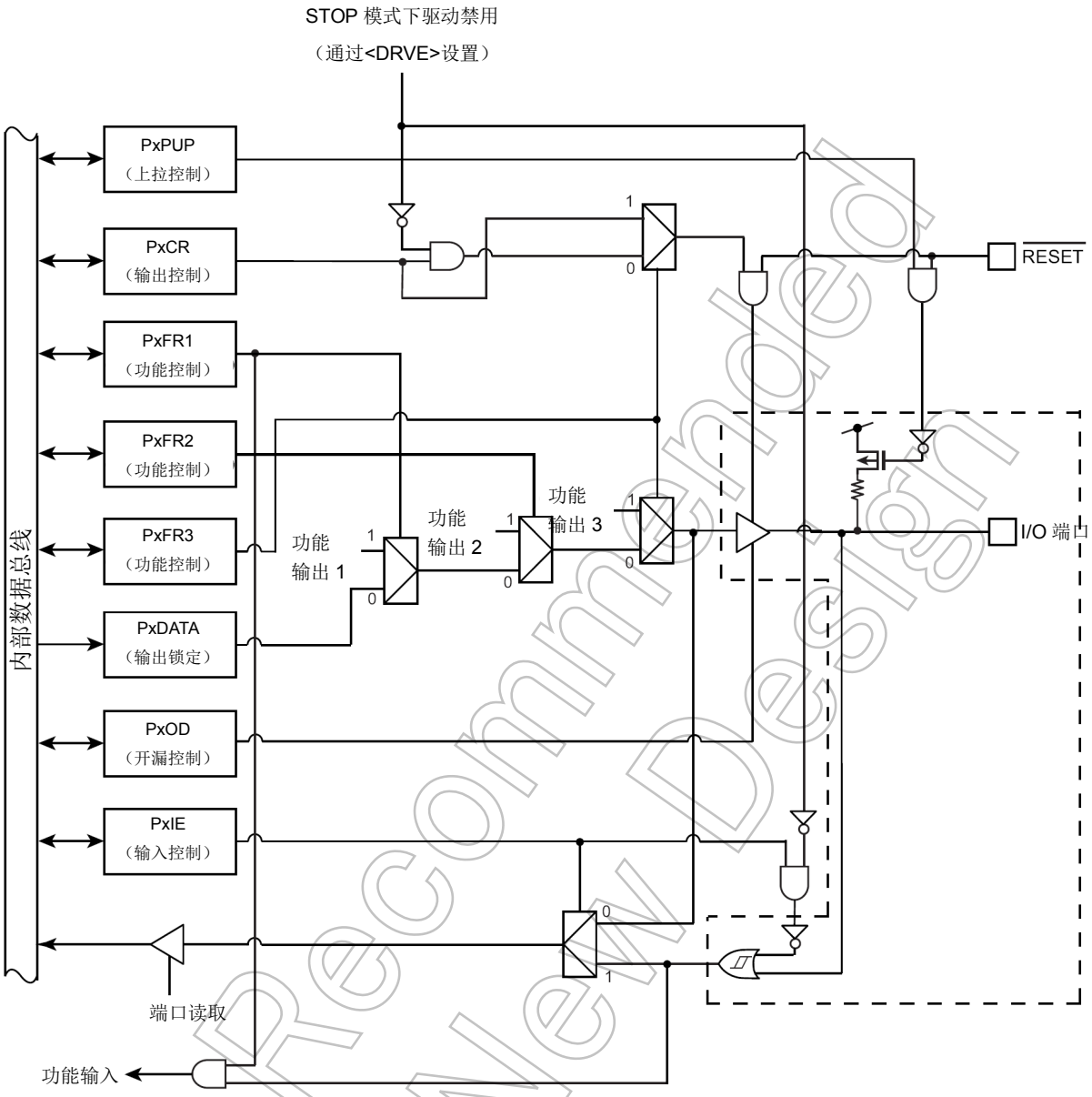


图 8-38 T38 型端口

8.3.40 T39型

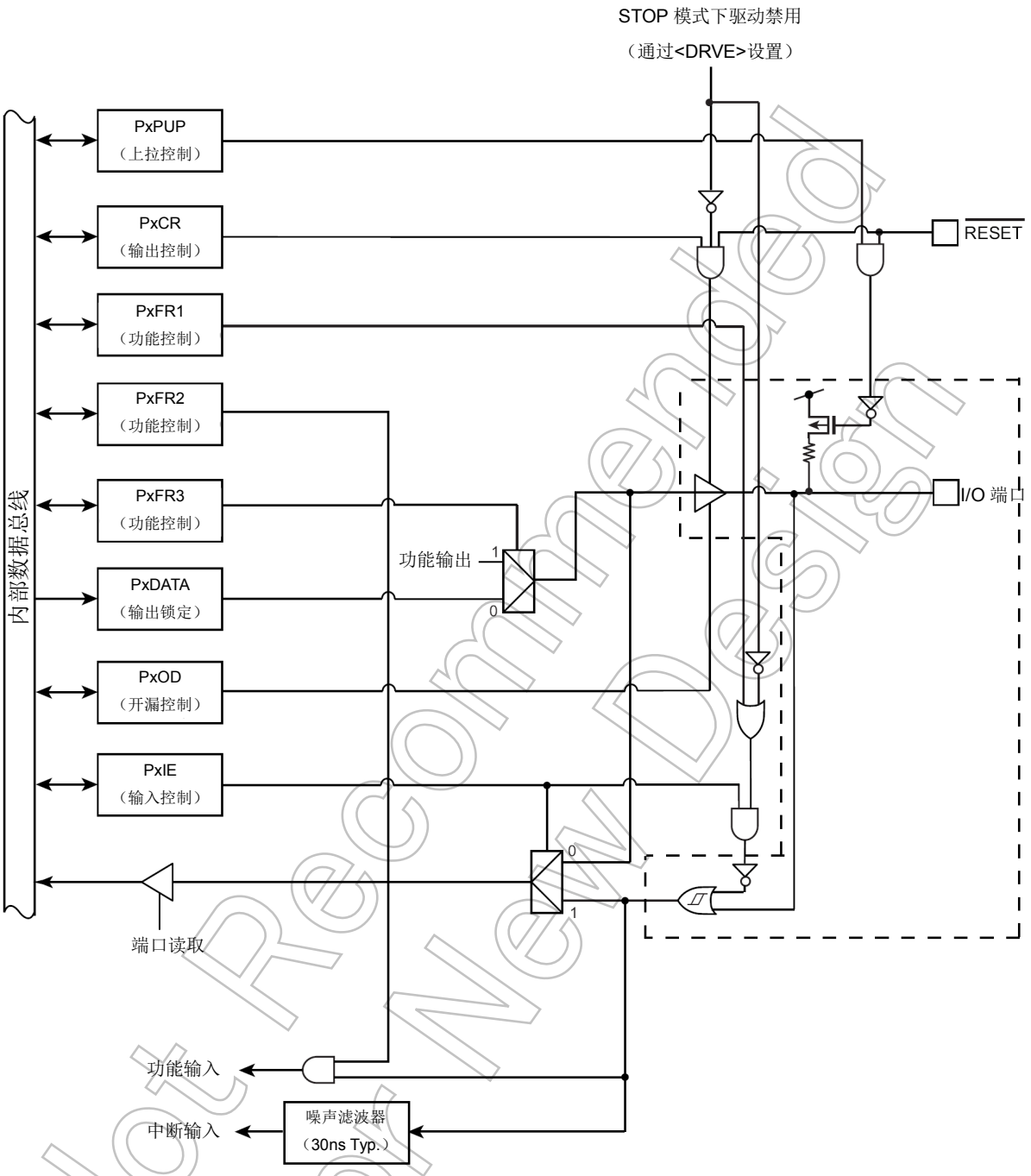


图 8-39 T39 型端口

8.4 附件 (端口设置列表)

下表为各功能寄存器的设置

对于所有寄存器设置，“复位后“区域无 o 的端口初始化设置到 “0“。

“×“位的设置可任意规定。

8.4.1 端口 A 设置

表 8-4 端口设置列表 (端口 A)

引脚	端口类型	功能	复位后	PACR	PAFR1	PAOD	PAPUP	PAIE
PA0	T1	输入端口		0	0	×	×	1
		输出端口		1	0	×	×	0
		数据 (I/O) / 地址数据 (I/O)		1	1	×	×	1
PA1	T1	输入端口		0	0	×	×	1
		输出端口		1	0	×	×	0
		数据 (I/O) / 地址数据 (I/O)		1	1	×	×	1
PA2	T1	输入端口		0	0	×	×	1
		输出端口		1	0	×	×	0
		数据 (I/O) / 地址数据 (I/O)		1	1	×	×	1
PA3	T1	输入端口		0	0	×	×	1
		输出端口		1	0	×	×	0
		数据 (I/O) / 地址数据 (I/O)		1	1	×	×	1
PA4	T1	输入端口		0	0	×	×	1
		输出端口		1	0	×	×	0
		数据 (I/O) / 地址数据 (I/O)		1	1	×	×	1
PA5	T1	输入端口		0	0	×	×	1
		输出端口		1	0	×	×	0
		数据 (I/O) / 地址数据 (I/O)		1	1	×	×	1
PA6	T1	输入端口		0	0	×	×	1
		输出端口		1	0	×	×	0
		数据 (I/O) / 地址数据 (I/O)		1	1	×	×	1
PA7	T1	输入端口		0	0	×	×	1
		输出端口		1	0	×	×	0
		数据 (I/O) / 地址数据 (I/O)		1	1	×	×	1

8.4.2 端口 B 设置

表 8-5 端口设置列表 (端口 B)

引脚	端口类型	功能	复位后	PBCR	PBFR1	PBOD	PBPUP	PBIE
PB0	T1	输入端口		0	0	×	×	1
		输出端口		1	0	×	×	0
		数据 (I/O) / 地址数据 (I/O)		1	1	×	×	1
PB1	T1	输入端口		0	0	×	×	1
		输出端口		1	0	×	×	0
		数据 (I/O) / 地址数据 (I/O)		1	1	×	×	1
PB2	T1	输入端口		0	0	×	×	1
		输出端口		1	0	×	×	0
		数据 (I/O) / 地址数据 (I/O)		1	1	×	×	1
PB3	T1	输入端口		0	0	×	×	1
		输出端口		1	0	×	×	0
		数据 (I/O) / 地址数据 (I/O)		1	1	×	×	1
PB4	T1	输入端口		0	0	×	×	1
		输出端口		1	0	×	×	0
		数据 (I/O) / 地址数据 (I/O)		1	1	×	×	1
PB5	T1	输入端口		0	0	×	×	1
		输出端口		1	0	×	×	0
		数据 (I/O) / 地址数据 (I/O)		1	1	×	×	1
PB6	T1	输入端口		0	0	×	×	1
		输出端口		1	0	×	×	0
		数据 (I/O) / 地址数据 (I/O)		1	1	×	×	1
PB7	T1	输入端口		0	0	×	×	1
		输出端口		1	0	×	×	0
		数据 (I/O) / 地址数据 (I/O)		1	1	×	×	1

8.4.3. 端口 C 设置

表 8-6 端口设置列表 (端口 C)

引脚	端口类型	功能	复位后	PCCR	PCFR1	PCFR2	PCFR3	PCOD	PCPUP	PCIE
PC0	T2	输入端口		0	0	0	0	×	×	1
		输出端口		1	0	0	0	×	×	0
		地址 (输出)		1	1	0	0	×	×	0
		TXD8 (输出)		1	0	1	0	×	×	0
PC1	T3	输入端口		0	0	0	0	×	×	1
		输出端口		1	0	0	0	×	×	0
		地址 (输出)		1	1	0	0	×	×	0
		RXD8 (输入)		0	0	1	0	×	×	1
PC2	T4	输入端口		0	0	0	0	×	×	1
		输出端口		1	0	0	0	×	×	0
		地址 (输出)		1	1	0	0	×	×	0
		SCLK8 (输入)		0	0	1	0	×	×	1
		SCLK8 (输出)		1	0	1	0	×	×	0
		$\overline{\text{CTS8}}$ (输入)		0	0	0	1	×	×	1
PC3	T5	输入端口		0	0	0	0	×	×	1
		输出端口		1	0	0	0	×	×	0
		地址 (输出)		1	1	0	0	×	×	0
PC4	T2	输入端口		0	0	0	0	×	×	1
		输出端口		1	0	0	0	×	×	0
		地址 (输出)		1	1	0	0	×	×	0
		TXD9 (输出)		1	0	1	0	×	×	0
PC5	T3	输入端口		0	0	0	0	×	×	1
		输出端口		1	0	0	0	×	×	0
		地址 (输出)		1	1	0	0	×	×	0
		RXD9 (输入)		0	0	1	0	×	×	1
PC6	T4	输入端口		0	0	0	0	×	×	1
		输出端口		1	0	0	0	×	×	0
		地址 (输出)		1	1	0	0	×	×	0
		SCLK9 (输入)		0	0	1	0	×	×	1
		SCLK9 (输出)		1	0	1	0	×	×	0
		$\overline{\text{CTS9}}$ (输入)		0	0	0	1	×	×	1
PC7	T5	输入端口		0	0	0	0	×	×	1
		输出端口		1	0	0	0	×	×	0
		地址 (输出)		1	1	0	0	×	×	0

8.4.4 端口 D 设置

表 8-7 端口设置列表 (端口 D)

引脚	端口类型	功能	复位后	PDCR	PDFR1	PDFR2	PDFR3	PDOD	PDPUP	PDIE
PD0	T2	输入端口		0	0	0	0	×	×	1
		输出端口		1	0	0	0	×	×	0
		地址 (输出)		1	1	0	0	×	×	0
		TXD10 (输出)		1	0	1	0	×	×	0
PD1	T3	输入端口		0	0	0	0	×	×	1
		输出端口		1	0	0	0	×	×	0
		地址 (输出)		1	1	0	0	×	×	0
		RXD10 (输入)		0	0	1	0	×	×	1
PD2	T4	输入端口		0	0	0	0	×	×	1
		输出端口		1	0	0	0	×	×	0
		地址 (输出)		1	1	0	0	×	×	0
		SCLK10 (输入)		0	0	1	0	×	×	1
		SCLK10 (输出)		1	0	1	0	×	×	0
		$\overline{\text{CTS10}}$ (输入)		0	0	0	1	×	×	1
PD3	T5	输入端口		0	0	0	0	×	×	1
		输出端口		1	0	0	0	×	×	0
		地址 (输出)		1	1	0	0	×	×	0
PD4	T2	输入端口		0	0	0	0	×	×	1
		输出端口		1	0	0	0	×	×	0
		地址 (输出)		1	1	0	0	×	×	0
		TXD11 (输出)		1	0	1	0	×	×	0
PD5	T3	输入端口		0	0	0	0	×	×	1
		输出端口		1	0	0	0	×	×	0
		地址 (输出)		1	1	0	0	×	×	0
		RXD11 (输入)		0	0	1	0	×	×	1
PD6	T4	输入端口		0	0	0	0	×	×	1
		输出端口		1	0	0	0	×	×	0
		地址 (输出)		1	1	0	0	×	×	0
		SCLK11 (输入)		0	0	1	0	×	×	1
		SCLK11 (输出)		1	0	1	0	×	×	0
		$\overline{\text{CTS11}}$ (输入)		0	0	0	1	×	×	1
PD7	T6	输入端口		0	0	0	0	×	×	1
		输出端口		1	0	0	0	×	×	0
		地址 (输出)		1	1	0	0	×	×	0
		INTB (输入)		0	0	1	0	×	×	1

8.4.5 端口 E 设置

表 8-8 端口设置列表 (端口 E)

引脚	端口类型	功能	复位后	PECR	PEFR1	PEFR2	PEFR3	PEOD	PEPUP	PEIE
PE0	T3	输入端口		0	0	0	0	×	×	1
		输出端口		1	0	0	0	×	×	0
		地址 (输出)		1	1	0	0	×	×	0
		TB5IN0 (输入)		0	0	1	0	×	×	1
PE1	T3	输入端口		0	0	0	0	×	×	1
		输出端口		1	0	0	0	×	×	0
		地址 (输出)		1	1	0	0	×	×	0
		TB5IN1 (输入)		0	0	1	0	×	×	1
PE2	T3	输入端口		0	0	0	0	×	×	1
		输出端口		1	0	0	0	×	×	0
		地址 (输出)		1	1	0	0	×	×	0
		TB6IN0 (输入)		0	0	1	0	×	×	1
PE3	T3	输入端口		0	0	0	0	×	×	1
		输出端口		1	0	0	0	×	×	0
		地址 (输出)		1	1	0	0	×	×	0
		TB6IN1 (输入)		0	0	1	0	×	×	1
PE4	T36	输入端口		0	0	0	0	×	×	1
		输出端口		1	0	0	0	×	×	0
		地址 (输出)		1	1	0	0	×	×	0
		TXD0 (输出)		1	0	1	0	×	×	0
		CTXD (输出)		1	0	0	1	×	×	0
PE5	T37	输入端口		0	0	0	0	×	×	1
		输出端口		1	0	0	0	×	×	0
		地址 (输出)		1	1	0	0	×	×	0
		RXD0 (输入)		0	0	1	0	×	×	1
		CRXD (输入)		0	0	0	1	×	×	1
PE6	T4	输入端口		0	0	0	0	×	×	1
		输出端口		1	0	0	0	×	×	0
		地址 (输出)		1	1	0	0	×	×	0
		SCLK0 (输入)		0	0	1	0	×	×	1
		SCLK0 (输出)		1	0	1	0	×	×	0
		$\overline{\text{CTS0}}$ (输入)		0	0	0	1	×	×	1
PE7	T6	输入端口		0	0	0	0	×	×	1
		输出端口		1	0	0	0	×	×	0
		INT5 (输入)		0	0	1	0	×	×	1
		SCOUT (输出)		1	0	0	1	×	×	0

8.4.6 端口 F 设置

表 8-9 端口设置列表 (端口 F)

引脚	端口类型	功能	复位后	PFCR	PFFR1	PFOD	PFPUP	PFIE
PF0	T7	输入端口		0	0	×	×	1
		输出端口		1	0	×	×	0
		TRACECLK (输出)		1	1	×	×	0
PF1	T7	输入端口		0	0	×	×	1
		输出端口		1	0	×	×	0
		TRACEDATA0/SWV (输出)		1	1	×	×	0
PF2	T7	输入端口		0	0	×	×	1
		输出端口		1	0	×	×	0
		TRACEDATA1 (输出)		1	1	×	×	0
PF3	T7	输入端口		0	0	×	×	1
		输出端口		1	0	×	×	0
		TRACEDATA2 (输出)		1	1	×	×	0
PF4	T7	输入端口		0	0	×	×	1
		输出端口		1	0	×	×	0
		TRACEDATA3 (输出)		1	1	×	×	0

8.4.7 端口 G 设置

表 8-10 端口设置列表 (端口 G)

引脚	端口类型	功能	复位后	PGCR	PGFR1	PGFR2	PGFR3	PGOD	PGPUP	PGIE
PG0	T8	输入端口		0	0	0	0	×	×	1
		输出端口		1	0	0	0	×	×	0
		SO1 (输出)		1	1	0	0	×	×	0
		SDA1 (I/O)		1	1	0	0	1	×	1
		TB7IN0 (输入)		0	0	1	0	×	×	1
PG1	T8	输入端口		0	0	0	0	×	×	1
		输出端口		1	0	0	0	×	×	0
		SI1 (输入)		0	1	0	0	×	×	1
		SCL1 (I/O)		1	1	0	0	1	×	1
		TB7IN1 (输入)		0	0	1	0	×	×	1
PG2	T9	输入端口		0	0	0	0	×	×	1
		输出端口		1	0	0	0	×	×	0
		SCK1 (输入)		0	1	0	0	×	×	1
		SCK1 (输出)		1	1	0	0	×	×	0
		$\overline{\text{CTS0}}$ (输入)		0	0	0	1	×	×	1
PG3	T10	输入端口		0	0	0	0	×	×	1
		输出端口		1	0	0	0	×	×	0
		INT6 (输入)		0	1	0	0	×	×	1
		$\overline{\text{CTS1}}$ (输入)		0	0	0	1	×	×	1
PG4	T8	输入端口		0	0	0	0	×	×	1
		输出端口		1	0	0	0	×	×	0
		SO2 (输出)		1	1	0	0	×	×	0
		SDA2 (I/O)		1	1	0	0	1	×	1
		TB9IN0 (输入)		0	0	1	0	×	×	1
PG5	T8	输入端口		0	0	0	0	×	×	1
		输出端口		1	0	0	0	×	×	0
		SI2 (输入)		0	1	0	0	×	×	1
		SCL2 (I/O)		1	1	0	0	1	×	1
		TB9IN1 (输入)		0	0	1	0	×	×	1
PG6	T38	输入端口		0	0	0	0	×	×	1
		输出端口		1	0	0	0	×	×	0
		SCK2 (输入)		0	1	0	0	×	×	1
		SCK2 (输出)		1	1	0	0	×	×	0
		USBPON (输出)		1	0	1	0	×	×	0
		$\overline{\text{CTS3}}$ (输入)		0	0	0	1	×	×	1
PG7	T39	输入端口		0	0	0	0	×	×	1
		输出端口		1	0	0	0	×	×	0
		INT7 (输入)		0	1	0	0	×	×	1
		$\overline{\text{USBOC}}$ (输入)		0	0	1	0	×	×	1
		$\overline{\text{WDTOUT}}$ (输出)		1	0	0	1	×	×	0

8.4.8 端口 H 设置

表 8-11 端口设置列表 (端口 H)

引脚	端口类型	功能	复位后	PHCR	PHFR1	PHFR2	PHOD	PHPUP	PHIE
PH0	T12	输入端口		0	0	0	×	×	1
		输出端口		1	0	0	×	×	0
		SO3 (输出)		1	1	0	×	×	0
		SDA3 (I/O)		1	1	0	1	×	1
		TBAIN0 (输入)		0	0	1	×	×	1
PH1	T12	输入端口		0	0	0	×	×	1
		输出端口		1	0	0	×	×	0
		SI3 (输入)		0	1	0	×	×	1
		SCL3 (I/O)		1	1	0	1	×	1
		TBAIN1 (输入)		0	0	1	×	×	1
PH2	T12	输入端口		0	0	0	×	×	1
		输出端口		1	0	0	×	×	0
		SCK3 (输入)		0	1	0	×	×	1
		SCK3 (输出)		1	1	0	×	×	0
		TBBIN0 (输入)		0	0	1	×	×	1
PH3	T13	输入端口		0	0	0	×	×	1
		输出端口		1	0	0	×	×	0
		INT6 (输入)		0	1	0	×	×	1
		TBBIN1 (输入)		0	0	1	×	×	1
PH4	T12	输入端口		0	0	0	×	×	1
		输出端口		1	0	0	×	×	0
		SO4 (输出)		1	1	0	×	×	0
		SDA4 (I/O)		1	1	0	1	×	1
		TBDIN0 (输入)		0	0	1	×	×	1
PH5	T12	输入端口		0	0	0	×	×	1
		输出端口		1	0	0	×	×	0
		SI4 (输入)		0	1	0	×	×	1
		SCL4 (I/O)		1	1	0	1	×	1
		TBDIN1 (输入)		0	0	1	×	×	1
PH6	T12	输入端口		0	0	0	×	×	1
		输出端口		1	0	0	×	×	0
		SCK2 (输入)		0	1	0	×	×	1
		SCK2 (输出)		1	1	0	×	×	0
		TBEIN0 (输入)		0	0	1	×	×	1
PH7	T13	输入端口		0	0	0	×	×	1
		输出端口		1	0	0	×	×	0
		INT7 (输入)		0	1	0	×	×	1
		TBEIN1 (输入)		0	0	1	×	×	1

8.4.9 端口I设置

表 8-12 端口设置列表 (端口 I)

引脚	端口类型	功能	复位后	PICR	PIFR1	PIOD	PIPUP	PIIE
PI0	T14	输入端口		0	0	×	×	1
		输出端口		1	0	×	×	0
PI1	T15	输入端口		0	0	×	×	1
		输出端口		1	0	×	×	0
		CEC (输入)		0	1	×	×	1

注:PI0 输入和上拉启用，并在复位处于 “低“状态时作为$\overline{\text{BOOT}}$输入引脚。

8.4.10 端口 J 设置

表 8-13 端口设置列表 (端口 J)

引脚	端口类型	功能	复位后	PJFR2	PJPUP	PJIE
PJ0	T17	输入端口		0	×	1
		模拟输入		0	0	0
PJ1	T17	输入端口		0	×	1
		模拟输入		0	0	0
PJ2	T17	输入端口		0	×	1
		模拟输入		0	0	0
PJ3	T18	输入端口		0	×	1
		模拟输入		0	0	0
		$\overline{\text{ADTRG}}$ (输入)		1	×	1
PJ4	T19	输入端口		0	×	1
		模拟输入		0	0	0
		KWUP0 (输入)		1	×	1
PJ5	T19	输入端口		0	×	1
		模拟输入		0	0	0
		KWUP1 (输入)		1	×	1
PJ6	T19	输入端口		0	×	1
		模拟输入		0	0	0
		KWUP2 (输入)		1	×	1
PJ7	T19	输入端口		0	×	1
		模拟输入		0	0	0
		KWUP3 (输入)		1	×	1

8.4.11 端口 K 设置

表 8-14 端口设置列表 (端口 K)

引脚	端口类型	功能	复位后	PKPUP	PKIE
PK0	T17	输入端口		×	1
		模拟输入		0	0
PK1	T17	输入端口		×	1
		模拟输入		0	0
PK2	T17	输入端口		×	1
		模拟输入		0	0
PK3	T17	输入端口		×	1
		模拟输入		0	0
PK4	T17	输入端口		×	1
		模拟输入		0	0
PK5	T17	输入端口		×	1
		模拟输入		0	0
PK6	T17	输入端口		×	1
		模拟输入		0	0
PK7	T17	输入端口		×	1
		模拟输入		0	0

8.4.12 端口 L 设置

表 8-15 端口设置列表 (端口 L)

引脚	端口类型	功能	复位后	PLCR	PLFR1	PLFR2	PLFR3	PLOD	PLPUP	PLIE
PL0	T20	输入端口		0	0	0	0	×	×	1
		输出端口		1	0	0	0	×	×	0
		SO0 (输出)		1	1	0	0	×	×	0
		SDA0 (I/O)		1	1	0	0	1	×	1
		TB0OUT (输出)		1	0	1	0	×	×	0
PL1	T20	输入端口		0	0	0	0	×	×	1
		输出端口		1	0	0	0	×	×	0
		SI0 (输入)		0	1	0	0	×	×	1
		SCL0 (I/O)		1	1	0	0	1	×	1
		TB1OUT (输出)		1	0	1	0	×	×	0
PL2	T20	输入端口		0	0	0	0	×	×	1
		输出端口		1	0	0	0	×	×	0
		SCK0 (输入)		0	1	0	0	×	×	1
		SCK0 (输出)		1	1	0	0	×	×	0
		TB2OUT (输出)		1	0	1	0	×	×	0
PL3	T21	输入端口		0	0	0	0	×	×	1
		输出端口		1	0	0	0	×	×	0
		INT0 (输入)		0	1	0	0	×	×	1
		TB3OUT (输出)		1	0	1	0	×	×	0
PL4	T22	输入端口		0	0	0	0	×	×	1
		输出端口		1	0	0	0	×	×	0
		TXD1 (输出)		1	1	0	0	×	×	0
		TB4OUT (输出)		1	0	1	0	×	×	0
PL5	T23	输入端口		0	0	0	0	×	×	1
		输出端口		1	0	0	0	×	×	0
		RXD1 (输入)		0	1	0	0	×	×	1
		TB5OUT (输出)		1	0	1	0	×	×	0
PL6	T24	输入端口		0	0	0	0	×	×	1
		输出端口		1	0	0	0	×	×	0
		SCLK1 (输入)		0	1	0	0	×	×	1
		SCLK1 (输出)		1	1	0	0	×	×	0
		TB6OUT (输出)		1	0	1	0	×	×	0
		$\overline{CTS1}$ (输入)		0	0	0	1	×	×	1
PL7	T21	输入端口		0	0	0	0	×	×	1
		输出端口		1	0	0	0	×	×	0
		INT1 (输入)		0	1	0	0	×	×	1
		TB7OUT (输出)		1	0	1	0	×	×	0

8.4.13 端口 M 设置

表 8-16 端口设置列表 (端口 M)

引脚	端口类型	功能	复位后	PMCR	PMFR1	PMFR2	PMFR3	PMOD	PMPUP	PMIE
PM0	T25	输入端口		0	0	0	0	×	×	1
		输出端口		1	0	0	0	×	×	0
		SCLK2 (输入)		0	1	0	0	×	×	1
		SCLK2 (输出)		1	1	0	0	×	×	0
		TB1IN0 (输入)		0	0	1	0	×	×	1
		$\overline{\text{CTS2}}$ (输入)		0	0	0	1	×	×	1
PM1	T26	输入端口		0	0	0	0	×	×	1
		输出端口		1	0	0	0	×	×	0
		TXD2 (输出)		1	1	0	0	×	×	0
		TB1IN1 (输入)		0	0	1	0	×	×	1
PM2	T23	输入端口		0	0	0	0	×	×	1
		输出端口		1	0	0	0	×	×	0
		RXD2 (输入)		0	1	0	0	×	×	1
		$\overline{\text{ALARM}}$ (输出)		1	0	1	0	×	×	0
PM3	T21	输入端口		0	0	0	0	×	×	1
		输出端口		1	0	0	0	×	×	0
		INT2 (输入)		0	1	0	0	×	×	1
		TB3OUT (输出)		1	0	1	0	×	×	0
PM4	T27	输入端口		0	0	0	0	×	×	1
		输出端口		1	0	0	0	×	×	0
		SCLK3 (输入)		0	1	0	0	×	×	1
		SCLK3 (输出)		1	1	0	0	×	×	0
		$\overline{\text{CTS3}}$ (输入)		0	0	0	1	×	×	1
PM5	T28	输入端口		0	0	0	0	×	×	1
		输出端口		1	0	0	0	×	×	0
		TXD3 (输出)		1	1	0	0	×	×	0
PM6	T29	输入端口		0	0	0	0	×	×	1
		输出端口		1	0	0	0	×	×	0
		RXD3 (输入)		0	1	0	0	×	×	1
PM7	T30	输入端口		0	0	0	0	×	×	1
		输出端口		1	0	0	0	×	×	0
		INT3 (输入)		0	1	0	0	×	×	1

8.4.14 端口 N 设置

表 8-17 端口设置列表 (端口 N)

引脚	端口类型	功能	复位后	PNCR	PNFR1	PNFR2	PNFR3	PNOD	PNPUP	PNIE
PN0	T28	输入端口		0	0	0	0	×	×	1
		输出端口		1	0	0	0	×	×	0
		TXD4 (输出)		1	1	0	0	×	×	0
PN1	T29	输入端口		0	0	0	0	×	×	1
		输出端口		1	0	0	0	×	×	0
		RXD2 (输入)		0	1	0	0	×	×	1
PN2	T25	输入端口		0	0	0	0	×	×	1
		输出端口		1	0	0	0	×	×	0
		SCLK4 (输入)		0	1	0	0	×	×	1
		SCLK4 (输出)		1	1	0	0	×	×	0
		TB2IN0 (输入)		0	0	1	0	×	×	1
		$\overline{CTS4}$ (输入)		0	0	0	1	×	×	1
PN3	T30	输入端口		0	0	0	0	×	×	1
		输出端口		1	0	0	0	×	×	0
		INT4 (输入)		0	1	0	0	×	×	1
		TB2IN1 (输入)		0	0	1	0	×	×	1
		RMC0 (输入)		0	0	0	1	×	×	1
PN4	T28	输入端口		0	0	0	0	×	×	1
		输出端口		1	0	0	0	×	×	0
		TXD5 (输出)		1	1	0	0	×	×	0
PN5	T29	输入端口		0	0	0	0	×	×	1
		输出端口		1	0	0	0	×	×	0
		RXD5 (输入)		0	1	0	0	×	×	1
PN6	T25	输入端口		0	0	0	0	×	×	1
		输出端口		1	0	0	0	×	×	0
		SCLK5 (输入)		0	1	0	0	×	×	1
		SCLK5 (输出)		1	1	0	0	×	×	0
		TBFIN0 (输入)		0	0	1	0	×	×	1
		$\overline{CTS5}$ (输入)		0	0	0	1	×	×	1
PN7	T30	输入端口		0	0	0	0	×	×	1
		输出端口		1	0	0	0	×	×	0
		INT8 (输入)		0	1	0	0	×	×	1
		TBFIN1 (输入)		0	0	1	0	×	×	1
		RMC1 (输入)		0	0	0	1	×	×	1

8.4.15 端口 O 设置

表 8-18 端口设置列表 (端口 O)

引脚	端口类型	功能	复位后	POCR	POFR1	POFR2	POFR3	POOD	POPUP	POIE
PO0	T22	输入端口		0	0	0	0	×	×	1
		输出端口		1	0	0	0	×	×	0
		TXD6 (输出)		1	1	0	0	×	×	0
		TB8OUT (输出)		1	0	1	0	×	×	0
PO1	T23	输入端口		0	0	0	0	×	×	1
		输出端口		1	0	0	0	×	×	0
		RXD6 (输入)		0	1	0	0	×	×	1
		TB9OUT (输出)		1	0	1	0	×	×	0
PO2	T24	输入端口		0	0	0	0	×	×	1
		输出端口		1	0	0	0	×	×	0
		SCLK6 (输入)		0	1	0	0	×	×	1
		SCLK6 (输出)		1	1	0	0	×	×	0
		TBAOUT (输出)		1	0	1	0	×	×	0
		$\overline{\text{CTS6}}$ (输入)		0	0	0	1	×	×	1
PO3	T21	输入端口		0	0	0	0	×	×	1
		输出端口		1	0	0	0	×	×	0
		INT9 (输入)		0	1	0	0	×	×	1
		TBBOUT (输出)		1	0	1	0	×	×	0
PO4	T22	输入端口		0	0	0	0	×	×	1
		输出端口		1	0	0	0	×	×	0
		TXD7 (输出)		1	1	0	0	×	×	0
		TBCOUT (输出)		1	0	1	0	×	×	0
PO5	T23	输入端口		0	0	0	0	×	×	1
		输出端口		1	0	0	0	×	×	0
		RXD7 (输入)		0	1	0	0	×	×	1
		TBDOUT (输出)		1	0	1	0	×	×	0
PO6	T24	输入端口		0	0	0	0	×	×	1
		输出端口		1	0	0	0	×	×	0
		SCLK7 (输入)		0	1	0	0	×	×	1
		SCLK7 (输出)		1	1	0	0	×	×	0
		TBEOUT (输出)		1	0	1	0	×	×	0
		$\overline{\text{CTS7}}$ (输入)		0	0	0	1	×	×	1
PO7	T21	输入端口		0	0	0	0	×	×	1
		输出端口		1	0	0	0	×	×	0
		INTA (输入)		0	1	0	0	×	×	1
		TBFOUT (输出)		1	0	1	0	×	×	0

8.4.16 端口 P 设置

表 8-19 端口设置列表 (端口 P)

引脚	端口类型	功能	复位后	PPCR	PPFR1	PPFR2	PPOD	PPPUP	PPIE
PP0	T5	输入端口		0	0	0	×	×	1
		输出端口		1	0	0	×	×	0
		$\overline{CS2}$ (输出)		1	1	0	×	×	0
PP1	T31	输入端口		0	0	0	×	×	1
		输出端口		1	0	0	×	×	0
PP2	T32	输入端口		0	0	0	×	×	1
		输出端口		1	0	0	×	×	0
		$\overline{BLS0}$ (输出)		1	1	0	×	×	0
		SPDO (输出)		1	0	1	×	×	0
PP3	T33	输入端口		0	0	0	×	×	1
		输出端口		1	0	0	×	×	0
		$\overline{BLS1}$ (输出)		1	1	0	×	×	0
		SPDI (输入)		0	0	1	×	×	1
PP4	T34	输入端口		0	0	0	×	×	1
		输出端口		1	0	0	×	×	0
		$\overline{WE}$ (输出)		1	1	0	×	×	0
		SPCLK (输入)		0	0	1	×	×	1
		SPCLK (输出)		1	0	1	×	×	0
PP5	T35	输入端口		0	0	0	×	×	1
		输出端口		1	0	0	×	×	0
		$\overline{OE}$ (输出)		1	1	0	×	×	0
		SPFSS (输入)		0	0	1	×	×	1
		SPFSS (输出)		1	0	1	×	×	0
PP6	T5	输入端口		0	0	0	×	×	1
		输出端口		1	0	0	×	×	0
		$\overline{ALE}$ (输出)		1	1	0	×	×	0

9. DMA控制器 (DMAC)

9.1 概述

下表列出了其主要功能。

表 9-1 DMA 控制器功能 (1 件)

项目	功能		描述
通道数目	2ch		-
DMA 请求数目	16		-
DMA 启动触发器	硬件启动		外设电路 DMA 请求
	软件启动		DMAC×SoftBReq 寄存器一个写入启动。
总线主控	32 位 × 1 (AHB)		-
优先级	高 : CH0 低 : CH1		固定
FIFO	4 字 × 2ch (1 字 = 32 位)		-
总线宽度	8/16/32 位		传输源和传输目标分别设置。
突发量	1/4/8/16/32/64/128/256		-
传输数量	直到 4095		-
地址	传输源地址	增量 非增量	可以规定传输源和传输目标是应否有增量。 (不支持地址隐藏。)
	传输目标地址	增量 非增量	
字节储存序	支持小字节序		-
传输类型	外围-存储器 存储器-外围设备 存储器-存储器 外围设备-外围设备		当选择 “存储器到存储器“时，不支持 DMA 启动硬件启动。 详情见 DMAC×Cn 配置。 当选择 “外设到外设“时，可将特定外设分配成源或目标。外设分配，见 “9.4.1 外设到外设传输支持的外设功能。 TMPM364F10FG 不支持外设到外设。
中断功能	传输结束中断 (INTDMAC×TC) 错误中断 (INTDMAC×ERR)		-
特殊功能	散射/收集功能		-

9.2 DMA 传输类型

表 9-2 DMA 传输类型

<table>
<tr><th>编号</th><th>DMA 传输
类型</th><th>电路
生成 DMA
请求</th><th>DMA 请求类型</th><th>描述</th></tr>
<tr><td>1</td><td>存储器到
外设</td><td>外设
(目标)</td><td>突发请求</td><td>1 字传输时，将 DMA 控制器的释放突发量设置到“1“。</td></tr>
<tr><td>2</td><td>外设到
存储器</td><td>外设
(源)</td><td>突发请求 /
单一请求</td><td>当传输数据量不是突发量的整倍数时，突发请求和单一请求都可使用。
当传输数据量超出于突发量时，忽略单一请求而使用突发传输。
当小于突发量时，使用单一传输。</td></tr>
<tr><td>3</td><td>存储器到
存储器</td><td>DMAC</td><td>无</td><td>没有 DMAC 请求时启用 DMCA 启动数据传输。
(选择存储器到存储器模式时，将设置 DMAC×CnConfiguration 置)<E>
到“1“)
当所有传输数据传输完毕或当 DMAC 通道禁用时，DMAC 停止。</td></tr>
<tr><td rowspan="2">4</td><td rowspan="2">外设到
外设</td><td>外设
(源)</td><td>突发请求 /
单一请求</td><td rowspan="2"><table><tr><th>传输大小</th><th>源</th><th>目标</th></tr><tr><td>(1)突发量整数倍</td><td>突发请求</td><td>突发请求</td></tr><tr><td>(2)突发量的非整数倍</td><td>突发请求 /
单一请求</td><td>-</td></tr></table></td></tr>
<tr><td>外设
(目标)</td><td>突发请求</td></tr>
</table>

注: 当大量的数据在存储器-存储器下传输时，建议使用低优先通道。使用低优先通道时，在低优先通道正在传输的时候，高优先通道可以启动。当使用高优先通道时，在高优先通道正在传输时，低优先通道不能启动。

9.3 方块图

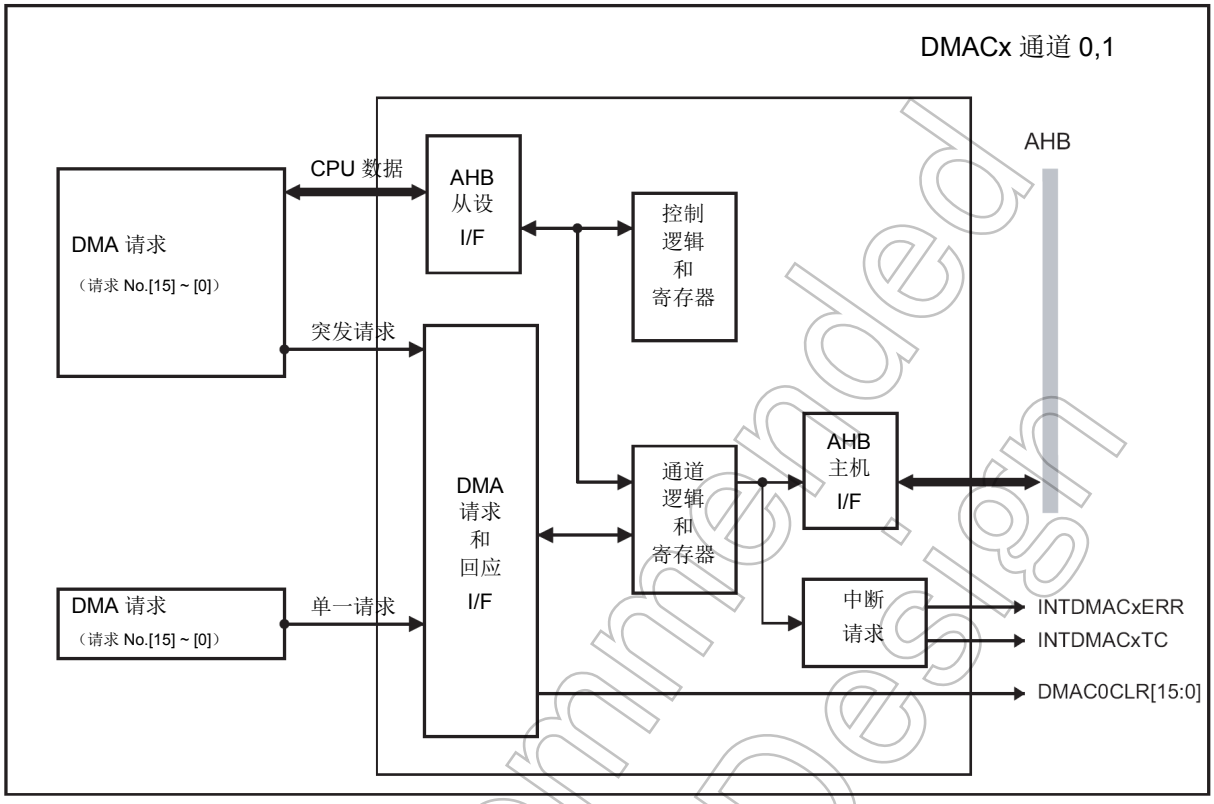


图 9-1 DMAC 方块图

9.4 TMPM364F10FG产品信息

9.4.1 外设-外设传输支持的外围设备外设功能

外设到外设传输支持的外设功能 (寄存器)如下所示。

TMPM364F10FG 不支持外设到外设传输。

9.4.2 DMA请求

与各 DMA 请求号相对应的 DMA 请求如下。

表 9-3 DMA 请求因素

DMA 请求号	对应的外设	
	ch0，ch1	
	突发请求	单一请求
0	SIO0/UART0 传输 / 接收	-
1	SIO1/UART1 传输 / 接收	-
2	SIO2/UART2 传 输 / 接收	-
3	SIO3/UART3 传输 / 接收	-
4	SIO4/UART4 传输 / 接收	-
5	-	-
6	-	-
7	-	-
8	-	-
9	-	-
10	-	-
11	-	-
12	SSP 传输	-
13	SSP 接收	SSP 接收
14	-	-
15	常规 AD 转换结束	-

9.4.3 中断请求

传输完成中断	错误中断
INTDMACTC	INTDMACERR

9.4.4 寄存器基址

基址
0×4000_0000

9.5 寄存器描述

9.5.1 DMAC寄存器列表

各寄存器的功能和地址如下。

寄存器名称		地址 (基址+)
DMAC 中断状态寄存器	DMAC×IntStaus	0×0000
DMAC 中断终端计数状态寄存器	DMAC×IntTCStatus	0×0004
DMAC 中断终端计数清除寄存器	DMAC×IntTCClear	0×0008
DMAC 错误中断状态寄存器	DMAC×IntErrorStatus	0×000C
DMAC 中断错误清除寄存器	DMAC×IntErrClr	0×0010
DMAC 原始中断终端计数状态寄存器	DMAC×RawIntTCStatus	0×0014
DMAC 原始错误中断状态寄存器	DMAC×RawIntErrorStatus	0×0018
DMAC 通道启用寄存器	DMAC×EnbldChns	0×001C
DMAC 软件突发请求寄存器	DMAC×SoftBReq	0×0020
DMAC 软件单一请求寄存器	DMAC×SoftSReq	0×0024
保留	-	0×0028
保留	-	0×002C
DMAC 配置寄存器	DMAC×Configuration	0×0030
保留	-	0×0034
DMAC 通道 0 源地址寄存器	DMAC×C0SrcAddr	0×0100
DMAC 通道 0 目的地址寄存器	DMAC×C0DestAddr	0×0104
DMAC 通道 0 链表件寄存器	DMAC×C0LLI	0×0108
DMAC 通道 0 控制寄存器	DMAC×C0Control	0×010C
DMAC 通道 0 配置寄存器	DMAC×C0Configuration	0×0110
DMAC 通道 1 源头地址寄存器	DMAC×C1SrcAddr	0×0120
DMAC 通道 1 目的地址寄存器	DMAC×C1DestAddr	0×0124
DMAC 通道 1 链表件寄存器	DMAC×C1LLI	0×0128
DMAC 通道 1 控制寄存器	DMAC×C1Control	0×012C
DMAC 通道 1 配置寄存器	DMAC×C1Configuration	0×0130

注 1: 通过使用字 (32 位)读取和写入访问寄存器。

注 2: 禁止访问 “保留“区域。

注 3: 对于为各通道准备的寄存器，如果通道结构相同，单元号用 “×“表示，通道号用 “n“表示。

注 4: 当寄存器通过各通道分配写入后而没有通过各通道分配读取时，在两个指令之间插入一个机器循环，或将没有通过各通道分配的寄存器读取两次。

9.5.2 DMAC×IntStatus (DMAC中断状态寄存器)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	未定义	未定义	未定义	未定义	未定义	未定义	未定义	未定义
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	未定义	未定义	未定义	未定义	未定义	未定义	未定义	未定义
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	未定义	未定义	未定义	未定义	未定义	未定义	未定义	未定义
	7	6	5	4.	3	2	1	0
比特符号	-	-	-	-	-	-	IntStatus1	IntStatus0
复位后	未定义	未定义	未定义	未定义	未定义	未定义	0	0

位	比特符号	类型	功能
31-2	-	-	写为零
1	IntStatus1	R	DMAC 通道 1 传输终止中断状态 0：中断未请求 1：中断请求 在穿过传输终止中断启用了寄存器以及错误中断启用寄存器后，DMAC 中断产生的状态。当发生测试错误或当计数器完成计数后，发出中断请求。
0	IntStatus0	R	DMAC 通道 0 中断生成状态 0：中断未请求 1：中断请求 在穿过传输终端中断启用了寄存器以及错误中断启用寄存器后，DMAC 中断产生的状态。当发生测试错误或当计数器完成计数后，发出中断请求。

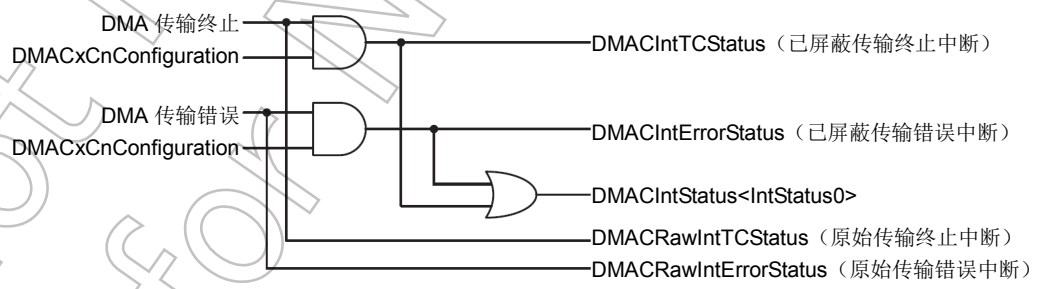


图 9-2 中断关联的方块图

9.5.3 DMAC×IntTCStatus (DMAC中断终端计数状态寄存器)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	未定义	未定义	未定义	未定义	未定义	未定义	未定义	未定义
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	未定义	未定义	未定义	未定义	未定义	未定义	未定义	未定义
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	未定义	未定义	未定义	未定义	未定义	未定义	未定义	未定义
	7	6	5	4.	3	2	1	0
比特符号	-	-	-	-	-	-	IntTCStatus1	IntTCStatus0
复位后	未定义	未定义	未定义	未定义	未定义	未定义	0	0

位	比特符号	类型	功能
31-2	-	-	写作零
1	IntTCStatus1	R	DMAC 通道 1 传输终止中断状态 0：中断未请求 1：中断请求 后启用传输终止中断生成状态。
0	IntTCStatus0	R	DMAC 通道 0 传输终止中断状态 0：中断未请求 1：中断请求 后-启用传输终止中断生成状态。

9.5.4 DMAC×IntTCClear (DMAC中断终端计数清零寄存器)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	未定义	未定义	未定义	未定义	未定义	未定义	未定义	未定义
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	未定义	未定义	未定义	未定义	未定义	未定义	未定义	未定义
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	未定义	未定义	未定义	未定义	未定义	未定义	未定义	未定义
	7	6	5	4	3	2	1	0
比特符号	-	-	-	-	-	-	IntTCClear1	IntTCClear0
复位后	未定义	未定义	未定义	未定义	未定义	未定义	0	0

位	比特符号	类型	功能
31-2	-	-	写作零
1	IntTCClear1	W	DMAC 通道 1 传输终端中断清除。 0：无操作 1：清除 当写入“1“时，DMAC×IntTCStatus<IntTCStatus1>将被清除。
0	IntTCClear0	W	DMAC 通道 0 传输终端中断清除。 0：无操作 1：清除 当写入“1“时，DMAC×IntTCStatus<IntTCStatus0>将被清除。

9.5.5 DMAC×IntErrorStatus (DMAC中断错误状态寄存器)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	未定义	未定义	未定义	未定义	未定义	未定义	未定义	未定义
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	未定义	未定义	未定义	未定义	未定义	未定义	未定义	未定义
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	未定义	未定义	未定义	未定义	未定义	未定义	未定义	未定义
	7	6	5	4	3	2	1	0
比特符号	-	-	-	-	-	-	IntErrStatus1	IIntErrStatus0
复位后	未定义	未定义	未定义	未定义	未定义	未定义	0	0

位	比特符号	类型	功能
31-2	-	-	写作零
1	IntErrStatus1	R	DMAC 通道 1 错误中断生成状态。 0：中断未请求 1：中断请求 启用后显示错误中断状态。
0	IntErrStatus0	R	DMAC 通道 0 错误中断生成状态。 0：中断未请求 1：中断请求 启用后显示错误中断状态。

9.5.6 DMAC×IntErrClr (DMAC中断错误清除寄存器)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	未定义	未定义	未定义	未定义	未定义	未定义	未定义	未定义
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	未定义	未定义	未定义	未定义	未定义	未定义	未定义	未定义
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	未定义	未定义	未定义	未定义	未定义	未定义	未定义	未定义
	7	6	5	4	3	2	1	0
比特符号	-	-	-	-	-	-	IntErrClr1	IntErrClr0
复位后	未定义	未定义	未定义	未定义	未定义	未定义	0	0

位	比特符号	类型	功能
31-2	-	-	写作零
1	IntErrClr1	W	DMAC 通道 1 传输终端中断清除。 0：无操作 1：清除 当 “1“被写入时，DMAC×IntErrorStatus<IntErrStatus1> 将被清除。
0	IntErrClr0	W	DMAC 通道 0 传输终端中断清除。 0：无操作 1：清除 当写入 “1“时，DMAC×IntErrorStatus<IntErrStatus0>将被清除。

9.5.7 DMAC×RawIntTCStatus (DMAC中断终端计数状态寄存器)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	未定义	未定义	未定义	未定义	未定义	未定义	未定义	未定义
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	未定义	未定义	未定义	未定义	未定义	未定义	未定义	未定义
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	未定义	未定义	未定义	未定义	未定义	未定义	未定义	未定义
	7	6	5	4	3	2	1	0
比特符号	-	-	-	-	-	-	RawIntTCS1	RawIntTCS0
复位后	未定义	未定义	未定义	未定义	未定义	未定义	0	0

位	比特符号	类型	功能
31-2	-	-	写作零
1	RawIntTCS1	R	DMAC 通道 1 提前-启用传输终止中断生成状态 0：中断未请求 1：中断请求
0	RawIntTCS0	R	DMAC 通道 0 提前-启用传输终止中断生成状态 0：中断未请求 1：中断请求

9.5.8 DMAC×RawIntErrorStatus (DMAC原始错误中断状态寄存器)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	未定义	未定义	未定义	未定义	未定义	未定义	未定义	未定义
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	未定义	未定义	未定义	未定义	未定义	未定义	未定义	未定义
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	未定义	未定义	未定义	未定义	未定义	未定义	未定义	未定义
	7	6	5	4	3	2	1	0
比特符号	-	-	-	-	-	-	RawIntErrS1	RawIntErrS0
复位后	未定义	未定义	未定义	未定义	未定义	未定义	0	0

位	比特符号	类型	功能
31-2	-	-	写作零
1	RawIntErrS1	R	DMAC 通道 1 前-启用错误中断状态。 0：中断未请求 1：中断请求
0	RawIntErrS0	R	DMAC 通道 0 前-启用错误中断状态。 0：中断未请求 1：中断请求

9.5.9 DMAC×EnbldChns (DMAC启用通道寄存器)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	未定义	未定义	未定义	未定义	未定义	未定义	未定义	未定义
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	未定义	未定义	未定义	未定义	未定义	未定义	未定义	未定义
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	未定义	未定义	未定义	未定义	未定义	未定义	未定义	未定义
	7	6	5	4	3	2	1	0
比特符号	-	-	-	-	-	-	EnabledCH1	EnabledCH0
复位后	未定义	未定义	未定义	未定义	未定义	未定义	0	0

位	比特符号	类型	功能
31-2	-	-	写作零
1	EnabledCH1	R	DMA 通道 1 启用状态。 0：禁用 1：启用 在完成了对 DMAC×CnControl 寄存器(值变为零)的所有次数的传输后，此标记被清除。
0	EnabledCH0	R	DMA 通道 0 启用状态。 0：禁用 1：启用 在完成了对 DMAC×CnControl 寄存器(值变为零)的所有次数的传输后，此标记被清除。

9.5.10 DMAC×SoftBReq (DMAC软件突发请求寄存器)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	未定义	未定义	未定义	未定义	未定义	未定义	未定义	未定义
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	未定义	未定义	未定义	未定义	未定义	未定义	未定义	未定义
	15	14	13	12	11	10	9	8
比特符号	SoftBReq15	SoftBReq14	SoftBReq13	SoftBReq12	SoftBReq11	SoftBReq10	SoftBReq9	SoftBReq8
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	SoftBReq7	SoftBReq6	SoftBReq5	SoftBReq4	SoftBReq3	SoftBReq2	SoftBReq1	SoftBReq0
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-16	-	-	写作零
15	SoftBReq15	R/W	由软件 (Request No. [15])发出的 DMA 突发请求 读取 : 0 : 停止 DMA 突发传输 1 : 运行 DMA 突发传输 写入 :0 : 无效 1 : DMA 突发请求
14	SoftBReq14	R/W	由软件 (Request No. [14])发出的 DMA 突发请求 读取 : 0 : 停止 DMA 突发传输 1 : 运行 DMA 突发传输 写入 : 0 : 无效 1 : DMA 突发请求
13	SoftBReq13	R/W	由软件 (Request No. [13])发出的 DMA 突发请求 读取 : 0 : 停止 DMA 突发传输 1 : 运行 DMA 突发传输 写入 : 0 : 无效 1 : DMA 突发请求
12	SoftBReq12	R/W	由软件 (Request No. [12])发出的 DMA 突发请求 读取 : 0 : 停止 DMA 突发传输 1 : 运行 DMA 突发传输 写入 : 0 : 无效 1 :DMA 突发请求
11	SoftBReq11	R/W	由软件 (Request No. [11])发出的 DMA 突发请求 读取:0:停止 DMA 突发传输 1:运行 DMA 突发传输 写入:0:无效 1:DMA 突发请求
10	SoftBReq10	R/W	由软件 (Request No. [10])发出的 DMA 突发请求 读取 : 0 : 停止 DMA 突发传输 1 : 运行 DMA 突发传输 写入 : 0: 无效 1 : DMA 突发请求
9	SoftBReq9	R/W	由软件 (Request No. [9])发出的 DMA 突发请求 读取 : 0 : 停止 DMA 突发传输 1 : 运行 DMA 突发传输 写入 :0 : 无效 1 : DMA 突发请求

位	比特符号	类型	功能
8	SoftBReq8	R/W	由软件 (Request No. [8])发出的 DMA 突发请求 读取 : 0 : 停止 DMA 突发传输 1 : 运行 DMA 突发传输 写入 : 0 : 无效 1 : DMA 突发请求
7	SoftBReq7	R/W	由软件 (Request No. [7])发出的 DMA 突发请求 读取 : 0 : 停止 DMA 突发传输 1 : 运行 DMA 突发传输 写入 : 0 : 无效 1 : DMA 突发请求
6	SoftBReq6	R/W	由软件 (Request No. [6])发出的 DMA 突发请求 读取 : 0 : 停止 DMA 突发传输 1 : 运行 DMA 突发传输 写入 :0 : 无效 1 : DMA 突发请求
5	SoftBReq5	R/W	由软件 (Request No. [5])发出的 DMA 突发请求 读取 : 0 : 停止 DMA 突发传输 1 : 运行 DMA 突发传输 写入 : 0 : 无效 1 : DMA 突发请求
4	SoftBReq4	R/W	由软件 (Request No. [4])发出的 DMA 突发请求 读取 : 0 : 停止 DMA 突发传输 1 : 运行 DMA 突发传输 写入 : 0 : 无效 1 : DMA 突发请求
3	SoftBReq3	R/W	由软件 (Request No. [3])发出的 DMA 突发请求 读取 : 0 : 停止 DMA 突发传输 1 : 运行 DMA 突发传输 写入 : 0 : 无效 1 : DMA 突发请求
2	SoftBReq2	R/W	由软件 (Request No. [2])发出的 DMA 突发请求 读取 : 0 : 停止 DMA 突发传输 1 : 运行 DMA 突发传输 写入: 0 : 无效 1 : DMA 突发请求
1	SoftBReq1	R/W	由软件 (Request No. [1])发出的 DMA 突发请求 读取:0:停止 DMA 突发传输 1:运行 DMA 突发传输 写入:0:无效 1:DMA 突发请求
0	SoftBReq0	R/W	由软件 (Request No. [0])发出的 DMA 突发请求 读取 : 0 停止 DMA 突发传输 1 : 运行 DMA 突发传输 写入 : 0 无效 1 : DMA 突发请求

注 1: 不要同时用软件和硬件执行 DMA 请求。

注 2: 参看 “9.4.2 DMA 请求“以获取 DMA 请求号。对没有突发请求的 DMA 请求号所对应的位清 “0“。

9.5.11 DMAC×SoftSReq (DMAC软件单一请求寄存器)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	未定义	未定义	未定义	未定义	未定义	未定义	未定义	未定义
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	未定义	未定义	未定义	未定义	未定义	未定义	未定义	未定义
	15	14	13	12	11	10	9	8
比特符号	SoftSReq15	SoftSReq14	SoftSReq13	SoftSReq12	SoftSReq11	SoftSReq10	SoftSReq9	SoftSReq8
复位后	0	0	0	0	0	0	0	0
	7	6	5	4.	3	2	1	0
比特符号	SoftSReq7	SoftSReq6	SoftSReq5	SoftSReq4	SoftSReq3	SoftSReq2	SoftSReq1	SoftSReq0
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-16	-	-	写作零
15	SoftSReq15	R/W	由软件 (Request No. [15])发出的 DMA 单一请求 读取： 0：停止 DMA 单一传输 1：运行 DMA 单一传输 写入：0：无效 1：DMA 单一请求
14	SoftSReq14	R/W	由软件 (Request No. [14])发出的 DMA 单一请求 读取：0：停止 DMA 单一传输 1：运行 DMA 单一传输 写入：0：无效 1：DMA 单一请求
13	SoftSReq13	R/W	由软件 (Request No. [13])发出的 DMA 单一请求 读取：0：停止 DMA 单一传输 1：运行 DMA 单一传输 写入: 0：无效 1：DMA 单一请求
12	SoftSReq12	R/W	由软件 (Request No. [12])发出的 DMA 单一请求 读取：0：停止 DMA 单一传输 1：运行 DMA 单一传输 写入：0：无效 1：DMA 单一请求
11	SoftSReq11	R/W	由软件 (Request No. [11])发出的 DMA 单一请求 读取：0：停止 DMA 单一传输 1：运行 DMA 单一传输 写入：0：无效 1：DMA 单一请求
10	SoftSReq10	R/W	由软件 (Request No. [10])发出的 DMA 单一请求 读取：0：停止 DMA 单一传输 1：运行 DMA 单一传输 写入：0:无效 1：DMA 单一请求
9	SoftSReq9	R/W	由软件 (Request No. [9])发出的 DMA 单一请求 读取：0：停止 DMA 单一传输 1：运行 DMA 单一传输 写入：0：无效 1：DMA 单一请求
8	SoftSReq8	R/W	由软件 (Request No. [8])发出的 DMA 单一请求 读取：0:停止 DMA 单一传输 1：运行 DMA 单一传输 写入：0：无效 1：DMA 单一请求

位	比特符号	类型	功能
7	SoftSReq7	R/W	由软件 (Request No. [7])发出的 DMA 单一请求 读取 : 0 : 停止 DMA 单一传输 1 : 运行 DMA 单一传输 写入 : 0 : 无效 1 : DMA 单一请求
6	SoftSReq6	R/W	由软件 (Request No. [6])发出的 DMA 单一请求 读取:0:停止 DMA 单一传输 1 : 运行 DMA 单一传输 写入 : 0 : 无效 1 : DMA 单一请求
5	SoftSReq5	R/W	由软件 (Request No. [5])发出的 DMA 单一请求 读取 : 0 : 停止 DMA 单一传输 1 : 运行 DMA 单一传输 写入 : 0 : 无效 1 : DMA 单一请求
4	SoftSReq4	R/W	由软件 (Request No. [4])发出的 DMA 单一请求 读取 : 0 : 停止 DMA 单一传输 1 : 运行 DMA 单一传输 写入 : 0 : 无效 1 : DMA 单一请求
3	SoftSReq3	R/W	由软件 (Request No. [3])发出的 DMA 单一请求 读取 : 0 : 停止 DMA 单一传输 1 : 运行 DMA 单一传输 写入 : 0 : 无效 1 : DMA 单一请求
2	SoftSReq2	R/W	由软件 (Request No. [2])发出的 DMA 单一请求 读取 : 0 : 停止 DMA 单一传输 1 : 运行 DMA 单一传输 写入 : 0 : 无效 1 : DMA 单一请求
1	SoftSReq1	R/W	由软件 (Request No. [1])发出的 DMA 单一请求 读取 : 0 : 停止 DMA 单一传输 1 : 运行 DMA 单一传输 写入 : 0 : 无效 1 : DMA 单一请求
0	SoftSReq0	R/W	由软件 (Request No. [0])发出的 DMA 单一请求 读取 : 0 : 停止 DMA 单一传输 1 : 运行 DMA 单一传输 写入 : 0 : 无效 1 : DMA 单一请求

注 1 : 不要同时用软件和硬件执行 DMA 请求。

注 2 : 参看 “9.4.2 DMA 请求 “以获取 DMA 请求号。将没有突发请求的 DMA 请求号所对应的位清 “0“。

9.5.12 DMAC×Configuration (DMAC配置寄存器)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	未定义	未定义	未定义	未定义	未定义	未定义	未定义	未定义
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	未定义	未定义	未定义	未定义	未定义	未定义	未定义	未定义
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	未定义	未定义	未定义	未定义	未定义	未定义	未定义	未定义
	7	6	5	4	3	2	1	0
比特符号	-	-	-	-	-	-	M	E
复位后	未定义	未定义	未定义	未定义	未定义	未定义	0	0

位	比特符号	类型	功能
31-2	-	-	写作零
1	M	R/W	写作零
0	E	R/W	DMA 电路控制 0：停止 1：操作 当电路停止时，DMA 电路寄存器不能读取或写入。当操作 DMA 时，总是设置<E>= “1“。

9.5.13 DMAC×CnSrcAddr (DMAC通道×源地址寄存器)

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

位	比特符号	类型	功能
31-0	SrcAddr[31:0]	R/W	设置 DMA 传输源地址。 设置前应确保源地址和位宽度的确认。 以下为源地址位宽度设置限制。

源地址位宽度 DMAC×CnControl<Swidth[2:0]>	最低有效地址设置
000：字节 (8 位)	无限制
001：半字 (16 位)	以 2 的倍数设置， (0×0,0×02,0×4,0×06,0×8,0×A,0×C…)
010：字 (32 位)	以 4 的倍数设置，(0×0,0×4,0×8,0×C…)

因为启用通道 “n“ (DMAC×CnConfiguration<E>= “1“)会更新写入寄存器数据，因此要在启用通道之前设置 DMAC×CnSrcAddr。

当 DMA 运行时，DMAC×CnSrcAddr 寄存器中的数值会循序改变，因而读取值是不固定。

而且在传输时不要更新 DMAC×CnSrcAddr。若要更改 DMAC×CnSrcAddr，更改前要确保禁用通道 “n “ (DMAC×CnConfiguration<E>= “0“)。

9.5.14 DMAC×CnDestAddr (DMAC通道×目标地址寄存器)

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

位	比特符号	类型	功能
31-0	DestAddr[31:0]	R/W	设置 DMA 传输目标地址。 设置前应确保目标地址和位宽度确认。 以下为目标地址位宽度设置限制。

目标地址位宽度 DMAC×CControl<Dwidth[2:0]>	最低有效地址设置
000：字节 (8 位)	无限制
001：半字 (16 位)	以 2 的倍数设置， (0×0,0×02,0×4,0×06,0×8,0×A,0×C…)
010：字 (32 位)	以 4 的倍数设置， (0×0,0×4,0×8,0×C…)

传输时勿更新 DMAC×CnDestAddr。若要更改 DMAC×CnDestAddr，在更改前应确保禁用通道 “n“ (DMAC×CnConfiguration<E>= “0“)。

9.5.15 DMAC×CnLLI (DMAC通道×链表件寄存器)

	31	30	29	28	27	26	25	24
比特符号	LLI							
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	LLI							
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	LLI							
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	LLI						-	-
复位后	0	0	0	0	0	0	未定义	未定义

位	比特符号	类型	功能
31-2	LLI[29:0]	R/W	为下一传输信息设置第一地址。 设置值应小于 0×FFFF_FFF0。 当<LLI> = 0，LLI 是最后一环时。在 DMA 传输结束后，DMA 通道禁用。
1-0	-	R/W	写作零

注:参看 “9.6 特殊功能“以了解<LLI>的详细操作。

9.5.16 DMAC×CnControl (DMAC通道n控制寄存器)

	31	30	29	28	27	26	25	24
比特符号	I	-	-	-	DI	SI	-	-
复位后	0	未定义	未定义	未定义	0	0	未定义	未定义
	23	22	21	20	19	18	17	16
比特符号	Dwidth			Swidth			DBSize	
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	DBSize	SBSize			TransferSize			
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	TransferSize							
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31	I	R/W	启用传输中断寄存器。 0：禁用 1：启用 传输终端中断通过设置 <I>="1"，和 DMAC×CnConfiguration<ITC>= "1"生成。当散射/收集功能用于最后的传输 DMAC 设置流，且通过将此位设置到启用只有在最后传输时才使传输终端启用生成。要在常规传输中产生中断，将此位设置为 "1"并转为启用模式。
30-28	-	W	写作零
27	DI	R/W	增量传输目标地址。 0：不增量 1：增量
26	SI	R/W	增加传输目标地址。 0：不增量 1: 增量
25-24	-	W	写作零
23-21	Dwidth[2:0]	R/W	传输目标的位宽度 000：字节 (8 位) 001：半字 (16 位) 010：字 (32 位) 其它：保留 参看表 9-4 获取设置值。
20-18	Swidth[2:0]	R/W	传输源的位宽度 000: 字节 (8 位) 001: 半字 (16 位) 010：字组 (32 位) 其它: 保留 参看表 9-4 获取设置值。
17-15	DBSize[2:0]	R/W	传输目标突发量: (注 1) 000: 1 节拍　100: 32 节拍 001: 4 节拍　101: 64 节拍 010: 8 节拍　110: 128 节拍 011: 16 节拍　111: 256 节拍 参看表 9-4 获取设置值。

位	比特符号	类型	功能
14-12	SBSize[2:0]	R/W	传输源突发量: (注 1) 000: 1 节拍 100:32 节拍 001: 4 节拍 101:64 节拍 010: 8 节拍 110:128 节拍 011: 16 节拍 111:256 节拍 参看表 9-4 获取设置值。
11-0	TransferSize [11:0]	R/W	设置传输总数。 由传输源的位宽度 (4 字节 / 2 字节 / 1 字节)指定的每一位宽度所对应的传输数据的量将被设置入 <TransderSize[11:0]>。因为突发量会显示每一内部 DMA 请求对应的传输数据量，故传输数据的量从未改变，即使当传输源位宽度和传输的总数未改变时有任何突发量被设置。 <TransferSize[11:0]> 的数值由 DMA 传输渐减到 0。 当被读取时，数据数的值没有传输。 传输总数作为传输源的位宽度<Swidth>。 例如: 当<Swidth>= “000“ (8 位)时，传输数目以字节为单位表示。 当<Swidth>= “001“ (16 位)时，传输数目以半字为单位表示。 当<Swidth>= “010“ (32 位)时，传输数目以字为单位表示。

注:DBsize (传输目标突发量)和 SBsize 设置的突发量与 AHB 总线 HBURST 没有关联。

表 9-4 如何确定 <Dwidth[2:0]>，<Swidth[2:0]>，<DBSize[2:0]>，<SBSize[2:0]>值

<Dwidth[2:0]> / <Swidth[2:0]>	设置数字来满足下述公式: 传输源位宽度 × 传输总数 = 传输目标位宽度 × N (N：整数)
	(ex.1) 传输源的位宽度:8 位，传输目标的位宽度:32 位 ，传输总数目:25 次 8 位 × 25 次 = 200 位 (25 字节) N = 200 ÷ 32 = 6.25 字 由于 6.25 不是整数，以上设置无效。 如果传输源位宽度小于传输目标位宽度，应小心设置传输总数目。
	(ex.2) 传输源的位宽度:32 位 (位)，传输目标的位宽度:16 位 (位)，传输总数目:13 次 32 位 × 13 次 = 416 位 (13 字节) N = 416 ÷ 16 = 26 半_字 由于 26 是整数，以上设置有效。
<DBSize[2:0]> / <SBSize[2:0]>	当进行 “外设到存储器“ 或 “存储器到外设“的传输时，外设电路会产生 DMA 请求信号表示已准备就绪。此信号将触发数据传输的执行。 (在“存储器到存储器“的情况下，只使用软件启动) 设置突发量以确定每个 DMA 请求信号对应的数据传输量。此寄存器连同 FIFO 缓冲器一起使用，FIFO 缓冲器可包含多种数据。

9.5.17 DMAC×CnConfiguration (DMAC 通道 n 配置寄存器)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	未定义	未定义	未定义	未定义	未定义	未定义	未定义	未定义
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	Halt	Active	Lock
复位后	未定义	未定义	未定义	未定义	未定义	0	0	0
	15	14	13	12	11	10	9	8
比特符号	ITC	IE	FlowCntrl			-	DestPeripheral	
复位后	0	0	0	0	0	未定义	0	0
	7	6	5	4	3	2	1	0
比特符号	DestPeripheral		-	SrcPeripheral				E
复位后	0	0	未定义	0	0	0	0	0

位	比特符号	类型	功能
31-19	-	W	写作零
18	Halt	R/W	控制器接收 DMA 请求 0：接收 DMA 请求 1：忽略 DMA 请求
17	Active	R	表示通道 FIFO 中是否有数据。 0 : FIFO 中无数据 1 : FIFO 中有数据
16	Lock	R/W	设置锁定传输 (非-分割传输) 0：禁用锁定传输 1：启用锁定传输 当锁定传输被激发时，所有规定的突发传输无需释放总线都将被连续执行，详见 “9.6 特殊功能“。
15	ITC	R/W	传输终端中断启用寄存器。 0：禁用中断 1：启用中断
14	IE	R/W	错误中断启用寄存器 0：禁用中断 1：启用中断
13-11	FlowCntrl [2:0]		R/W 设置传输模式 <FlowCntrl[2:0]> 设置值 / 传输方法 000: 存储器到存储器 (注) 001: 存储器到外设 010: 外设到存储器 011 ~ 111: 保留
10	-	W	写作零
9-6	DestPeripheral [3:0]	R/W	设置传输目标外设 (注 2) 见 “9.4.2 DMA 请求“。 当某一存储器是传输目标时，该设置被忽略。
5	-	W	写作零
4-1	SrcPeripheral [3:0]	R/W	设置传输源外设 (注 2) 见 “9.4.2 DMA 请求“。 当某一存储器是传输源，该设置被忽略。

位	比特符号	类型	功能
0	E	R/W	通道启用 0：禁用 1：启用 此位可启用/禁用通道。 (当选择 “存储器到存储器“模式时，此位将用作启动位) 当由 DMAC×CnControl <TransferSize>规定的传输数据量完成后，相应的<E>将被自动清 “0“。 传输中的禁用通道将失去 FIFO 中的数据。重启前将所有通道初始化。 暂停传输时，使用<HALT>来停止 DMA 请求，并轮询此数据直到 <Active>变为 “0“，然后禁用带有<E> 位的通道。

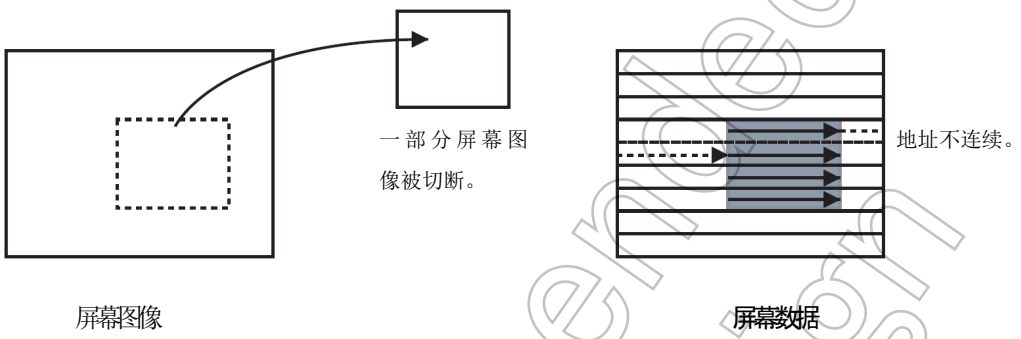
注 1: 当选择 “存储器到存储器 “时，用于 DMA 启动的硬件启动不被支持。启动传输写入 “1“ <E>。

注 2: 当 DMAC×ENableChns<EnabledCH×>启用，且相应的 DMAC×CnConfiguration<Halt>设置到 “1“时，在通道启用位 (E:bit0)被清 “0“后再把它们写入 (若不这样，则当写入它们时若出现从设备错误，该错误在复位后再出现。从设备错误，当传输宽度和地址不相符时，此错误出现。

9.6 特殊功能

9.6.1 散射/收集功能

当移动图像数据的一部分并将其传输时，图像数据不能当做连续数据来处理，地址将根据特殊规则出现大的改变。由于 DMA 只能通过使用连续地址来传输数据，必须在地址改变的位置进行所需的设置。



散射/收集功能可连续操作 DMA 设置 (传输源地址，目标地址，传输数量，以及传输总线宽度)，通过在每次设置的 DMA 执行数目完成后对它们进行加载，该 DMA 执行预置了相应“链表“，CPU 不需要控制操作。

在 DMAC×CnLLI 寄存器设置 “1“将启用/禁用操作。

可与链表进行设置的项目依照以下 4 个字进行配置:

1. DMAC×CnSrcAddr
2. DMAC×CnDestAddr
3. DMAC×CnLLI
4. DMAC×CnControl

它们可以带中断操作使用。

中断取决于 DMAC×CnControl 寄存器的计数终端中断启用位，可在每一 LLI 终端生成。当此位被使用时，即使当传输使用 LLI 来执行分支等操作的情况下，也可增加一个条件。清除中断，应控制 DMAC×IntTCClear 寄存器的适当位。

9.6.2 链表操作

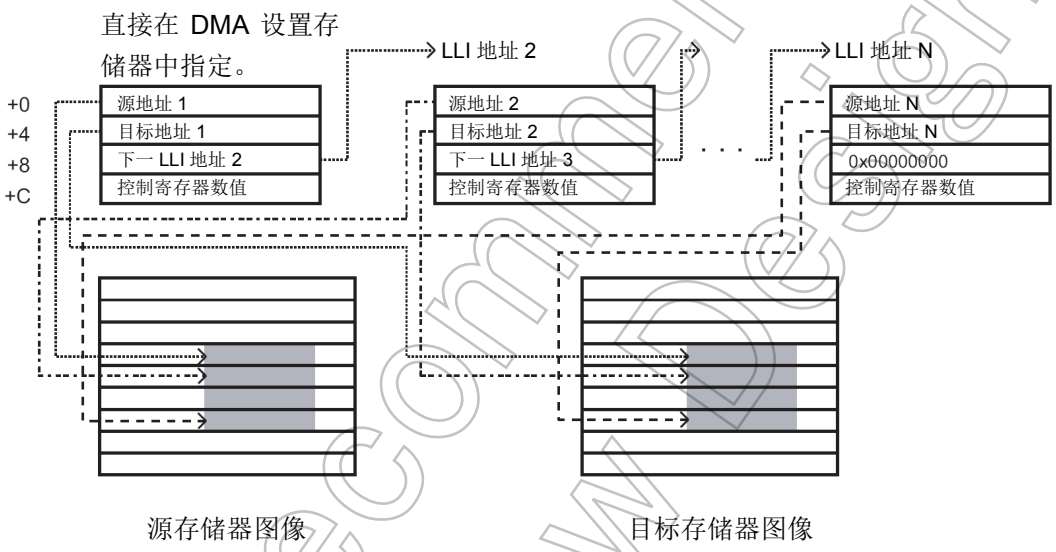
操作/散射/收集功能时，应先创建一套链表来定义传输源和源数据区域。

各设置称作 LLI (链表)。

每一个 LLI 控制一个数据块的传输。各 LLI 表示常规 DMA 设置和控制连续数据的传输。每次每一个 DMA 传输完毕后，下一 LLI 设置将被加载，以继续 DMA 操作 (菊花链)。

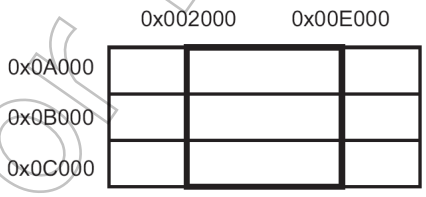
设置示例:

1. 第一个 DMA 传输设置应直接在 DMA 寄存器中设置。
2. 第二个以及随后的 DMA 传输设置应写入“下一 LLI 地址 X“的存储器地址中。
3. 到 N'th DMA 传输停止时，设置 “下一 LLI 地址 X“为 0×0000_0000。



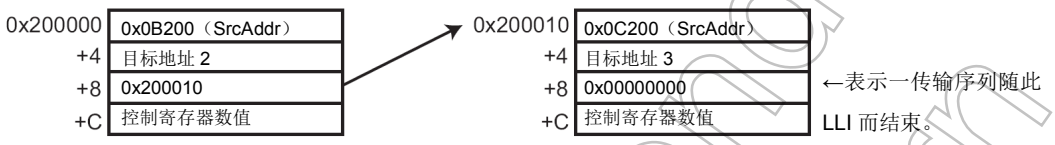
源存储器图像　　目标存储器图像

当在封闭的方框内传输数据时



	设置存储器	设置参数
+0	DMAC×CnSrcAddr:	:0×0A200
+4	DMAC×CnDestAddr:	:目标地址 1
+8	DMAC×CnLLI:	:0×200000
+C	DMAC×CnControl:	:设置突发传输的数量和传输数量，等等。

链表



10. 静态存储控制器

TMPM364F10FG 包含带异步访问静态存储 (NOR 型闪存和 SRAM)控制器。

注 1: 在确认外部存储器访问完成后，执行 WFI 指令。

注 2: 外部存储器不能用作 FIFO，因为某一外部总线的读取周期内的虚拟读取周期可能发生。

10.1 功能概述

功能概述见下表 10-1。

表 10-1 概述静态存储控制器的功能

项目	
支持的存储器类型 和总线连接	异步访问存储器 (NOR 闪存，SRAM，等)支持多元总线和独立总线。
数据总线宽度	16 位数据总线宽度
存储器地图	支持 64MB 访问区域，且将其分成四个 CS 信号和区域。 CS0 : 0×6000_0000 ~ 0×60FF_FFFF (16MB) CS1 : 0×6100_0000 ~ 0×61FF_FFFF (16MB) CS2 : 0×6200_0000 ~ 0×62FF_FFFF (16MB) CS3 : 0×6300_0000 ~ 0×63FF_FFFF (16MB)
定时调整	可通过寄存器控制 AC 定时。
时钟 (SMCCLK)	f_{sys} / 2
外部控制信号	独立总线 : D0 ~ D15，A1 ~ A23，$\overline{OE}$，$\overline{WE}$，$\overline{CS0}$ ~ $\overline{CS3}$，$\overline{BLS0}$，$\overline{BLS1}$ 多元总线 : AD0 ~ AD15，A17 ~ A23，$\overline{OE}$，$\overline{WE}$，$\overline{ALE}$，$\overline{CS0}$ ~ $\overline{CS3}$，$\overline{BLS0}$，$\overline{BLS1}$

10.2 方块图

SMC 方块图显示如下。

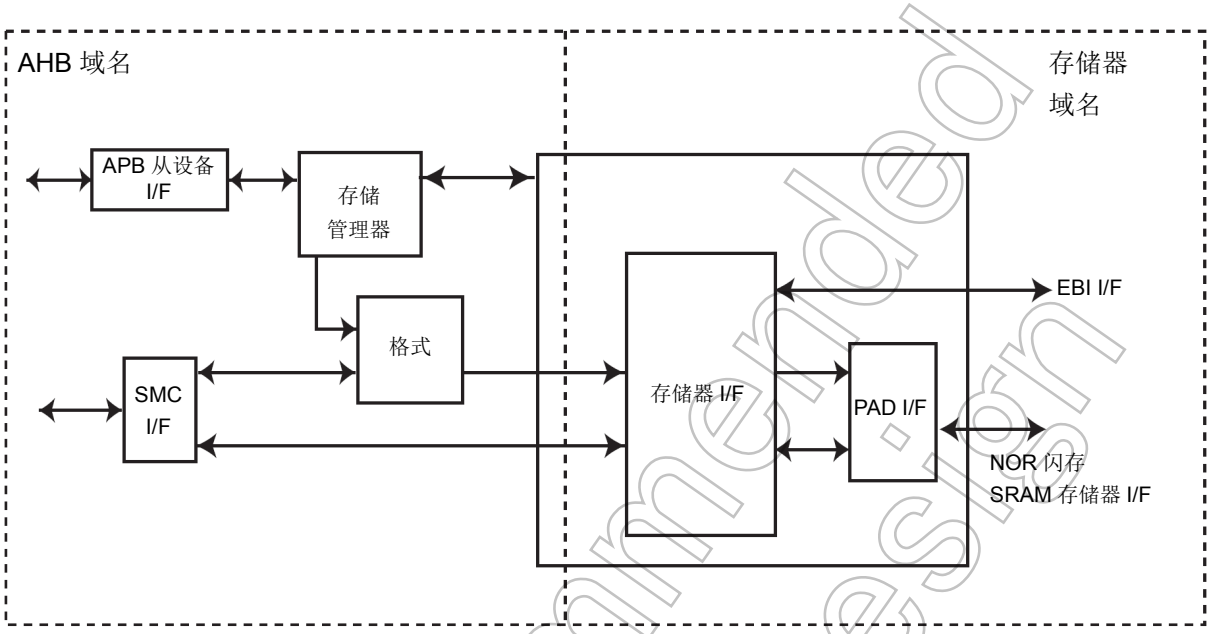


图 10-1 SMC 方块图

10.3 寄存器描述

10.3.1 SFR列表

以下列出 SFRs。

基址= 0×4000_1000

寄存器名称		地址 (基址+)
保留	-	0×0000
SMC 存储器接口配置寄存器	smc_memif_cfg	0×0004
保留	-	0×0008
保留	-	0×000C
SMC 直接命令寄存器	smc_direct_cmd	0×0010
SMC 周期设置寄存器	smc_set_cycles	0×0014
SMC 设置输入形式寄存器	smc_set_opmode	0×0018
保留	-	0×0020
SMC SRAM 周期寄存器<0>	smc_sram_cycles0_0	0×0100
SMC 输入形式寄存器<0>	smc_opmode0_0	0×0104
SMC SRAM 周期寄存器<1>	smc_sram_cycles0_1	0×0120
SMC 输入形式寄存器<1>	smc_opmode0_1	0×0124
SMC SRAM 周期寄存器<2>	smc_sram_cycles0_2	0×0140
SMC 输入形式寄存器<2>	smc_opmode0_2	0×0144
SMC SRAM 周期寄存器<3>	smc_sram_cycles0_3	0×0160
SMC 输入形式寄存器<3>	smc_opmode0_3	0×0164
保留	-	0×0200 ~ 0×0204, 0×0E00 ~ 0×0E08, 0×0FE0 ~ 0×0FFC

基址= 0×41FF_F100

寄存器名称		地址 (基址+)
SMC 模式寄存器	SMCMDMODE	0×0000

注 1: **通过使用字读取和字写入来访问寄存器。**

注 2: **勿访问保留地址。**

10.3.2 SMCMDMODE (模式寄存器)

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

位	比特符号	类型	功能
31-8	−	R	读作 “0“。
7-1	−	R/W	写作 “0“。
0	IFSMCMUXMD	R/W	SMC 存储器总线模式设置 0:独立总线模式 1:多元总线模式

注 1: 当 SMC 操作时不要更改<IFSMCMUXMD>

10.3.3 smc_memif_cfg (SMC存储器接口配置寄存器)

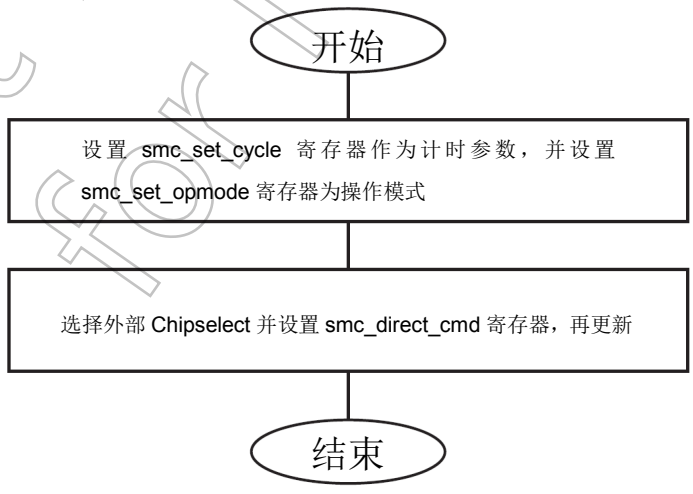
	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	未定义	未定义	未定义	未定义	未定义	未定义	未定义	未定义
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	未定义	未定义	未定义	未定义	未定义	未定义	未定义	未定义
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	未定义	未定义	未定义	未定义	未定义	未定义	未定义	未定义
	7	6	5	4	3	2	1	0
比特符号	-	-	memory_width		memory_chips		memory_type	
复位后	未定义	未定义	0	1	1	1	0	1

位	比特符号	类型	功能
31-6	−	R	以未定义读入。
5-4	memory_width [1:0]	R	最大外部 SMC 存储器总线宽度 01 : 16 位 其他 ： 忽略
3-2	memory_chips [1:0]	R	支持的存储器 CS (芯片选择)数目 00 :1 芯片 01 : 2 芯片 10 : 3 芯片 11 : 4 芯片
1-0	memory_type [1:0]	R	支持的存储器类型:SRAM 当<IFSMCMUXMD>为 “0“，读作 “01“。 当<IFSMCMUXMD>为 “1“，读作 “11“。 其他 ： 忽略

10.3.4 smc_direct_cmd (SMC直接命令寄存器)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	chip_select	
复位后	未定义	未定义	未定义	未定义	未定义	未定义	未定义	未定义
	23	22	21	20	19	18	17	16
比特符号	chip_select	cmd_type		-	-	-	-	
复位后	未定义	未定义	未定义	未定义	未定义	未定义	未定义	未定义
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	未定义	未定义	未定义	未定义	未定义	未定义	未定义	未定义
	7	6	5	4	3	2	1	0
比特符号	-	-	-	-	-	-	-	-
复位后	未定义	未定义	未定义	未定义	未定义	未定义	未定义	未定义

位	比特符号	类型	功能
31-26	–	W	写作 “0“。
25-23	chip_select[2:0]	W	CS 选择 000:CS0 001:CS1 010:CS2 011:CS3 100 ~ 111：设置禁止 选择目标芯片选择终端
22-21	cmd_type[1:0]	W	更新 set_opmode 寄存器以及 set-cycles 寄存器的数值 10：更新寄存器 其他：设置禁止
20-0	–	W	写作 “0“。



10.3.5 smc_set_cycles (SMC设置周期寄存器)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	未定义	未定义	未定义	未定义	未定义	未定义	未定义	未定义
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	Set_t5			Set_t4
复位后	未定义	未定义	未定义	未定义	未定义	未定义	未定义	未定义
	15	14	13	12	11	10	9	8
比特符号	Set_t4		Set_t3			Set_t2		
复位后	未定义	未定义	未定义	未定义	未定义	未定义	未定义	未定义
	7	6	5	4	3	2	1	0
比特符号	Set_t1				Set_t0			
复位后	未定义	未定义	未定义	未定义	未定义	未定义	未定义	未定义

位	比特符号	类型	功能
31-20	–	W	写作 “0“。
19-17	Set_t5[2:0]	W	设置 t_{TR} 的值 000： 设置禁止 001 ~ 111 : SMCCLK × 1 时钟 ~ SMCCLK × 7 时钟
16-14	Set_t4[2 :0]	W	设置 t_{PC} 的值 000： 设置禁止 001-111 : SMCCLK × 1 时钟 ~ SMCCLK × 7 时钟 多元总线模式下不支持 PAGE 访问。t_{PC} 只在独立总线模式下有效。
13-11	Set_t3[2 :0]	W	设置 t_{WP} 的值 (注) 000： 设置禁止 001 ~ 111 : SMCCLK × 1 时钟 ~ SMCCLK × 7 时钟 在多元模式下，针对<Set_t3>数值写入脉冲宽度 (t_{WP})增加一个时钟脉冲。
10-8	Set_t2[2:0]	W	设置 t_{CEOE} 的值 (注) 000： 设置禁止 001 ~ 111 : SMCCLK × 1 时钟 ~ SMCCLK × 7 时钟
7-4	Set_t1[3:0]	W	设置 t_{WC} 的值 (注) 0000： 设置禁止 0011 ~ 1111 : SMCCLK × 3 时钟 ~ SMCCLK × 15 时钟
3-0	Set_t0[3:0]	W	设置 t_{RC} 值 (注) 0000： 设置禁止 0010 ~ 1111 : SMCCLK × 2 时钟 ~ SMCCLK × 15 时钟

此寄存器用于调整静态存储器的访问周期，应设置成能满足所使用存储器的 A.C.规格。调整基时钟为 SMCCLK : $f_{sys}/2$.

要使 SMC 设置周期寄存器的设置生效，一定要执行 smc_direct_cmd 寄存器上的更新寄存器指令。

注:需要保持以下关系。

<Set_t3>,<Set_t1>	独立总线模式 :(t_{WP} + SMCCLK × 2 时钟)≤ t_{WC} 多元总线模式 :(t_{WP} + SMCCLK × 3 时钟)≤ t_{WC}
<Set_t2>,<Set_t0>	独立总线模式 :(t_{CEOE} + SMCCLK × 1 时钟)≤ t_{RC} 多元总线模式 :(t_{CEOE} + SMCCLK × 1 时钟)≤ t_{RC}

10.3.6 smc_set_opmode (SMC设置输入模式寄存器)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	未定义	未定义	未定义	未定义	未定义	未定义	未定义	未定义
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	未定义	未定义	未定义	未定义	未定义	未定义	未定义	未定义
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	set_adv	-	-	-
复位后	未定义	未定义	未定义	未定义	未定义	未定义	未定义	未定义
	7	6	5	4	3	2	1	0
比特符号	-	-	set_rd_bl			-	set_mw	
复位后	未定义	未定义	未定义	未定义	未定义	未定义	未定义	未定义

位	比特符号	类型	功能
31-12	−	W	写作 “0“。
11	set_adv	W	ALE 信号 0:不使用地址锁存启用（$\overline{ALE}$），（在独立总线模式下选用）。 1:使用地址锁存启用（$\overline{ALE}$），（在多元总线模式下选用）。
10-6	−	W	写作 “0“。
5-3	set_rd_bl[2:0]	W	读取数据的脉冲串长度的设置位。 000 : 1 节拍 001 : 4 节拍 其他:保留
2	−	W	写作 “0“。
1-0	set_mw[1:0]	W	存储器数据总线宽度设置值保持寄存器: 01 : 16 位 其他:保留 数据总线宽度的设置位。

要使 SMC 设置输入方式寄存器的设置生效，应执行 smc_direct_cmd 寄存器上的更新寄存器指令。

10.3.7 smc_sram_cycles0_0 (SMC SRAM周期寄存器0 <0>)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	未定义	未定义	未定义	未定义	未定义	未定义	未定义	未定义
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	t_tr			t_pc
复位后	未定义	未定义	未定义	未定义	0	0	1	0
	15	14	13	12	11	10	9	8
比特符号	t_pc		t_wp			t_ceoe		
复位后	1	0	1	1	0	0	1	1
	7	6	5	4	3	2	1	0
比特符号	t_wc				t_rc			
复位后	1	1	0	0	1	1	0	0

位	比特符号	类型	功能
31-20	−	W	写作 “0“。
19-17	t_tr[2:0]	R	扭转周期时间 001 ~ 111:SMCCLK × 1 时钟 ~ SMCCLK × 7 时钟
16-14	t_pc[2:0]	R	PAGE 周期时间 (注) 001 ~ 111 : SMCCLK × 1 时钟 ~ SMCCLK × 7 时钟
13-11	t_wp[2:0]	R	$\overline{WE}$脉冲周期时间 001 ~ 111 : SMCCLK × 1 时钟 ~ SMCCLK × 7 时钟
10-8	t_ceoe[2:0]	R	到$\overline{OE}$判断提示的延迟周期时间 001 ~ 111 : SMCCLK × 1 时钟 ~ SMCCLK × 7 时钟
7-4	t_wc[3:0]	R	写入周期时间 0011 ~ 1111 : SMCCLK× 3 时钟 ~ SMCCLK × 15 时钟
3-0	t_rc[3:0]	R	读取周期时间 0010 ~ 1111 : SMCCLK × 2 时钟 ~ SMCCLK × 15 时钟

注:多元总线模式下不支持页访问。t_{PC} 只在独立总线模式下有效。

10.3.8 smc_sram_cycles0_1 (SMC SRAM周期寄存器0 <1>)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	未定义	未定义	未定义	未定义	未定义	未定义	未定义	未定义
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	t_tr			t_pc
复位后	未定义	未定义	未定义	未定义	0	0	1	0
	15	14	13	12	11	10	9	8
比特符号	t_pc		t_wp			t_ceoe		
复位后	1	0	1	1	0	0	1	1
	7	6	5	4	3	2	1	0
比特符号	t_wc				t_rc			
复位后	1	1	0	0	1	1	0	0

位	比特符号	类型	功能
31-20	–	W	写作 “0“。
19-17	t_tr[2:0]	R	扭转周期时间 001 ~ 111 : SMCCLK × 1 时钟 ~ SMCCLK × 7 时钟
16-14	t_pc[2:0]	R	PAGE 周期时间 (注) 001 ~ 111 : SMCCLK × 1 时钟 ~ SMCCLK × 7 时钟
13-11	t_wp[2:0]	R	$\overline{WE}$脉冲周期时间 001 ~ 111 : SMCCLK × 1 时钟 ~ SMCCLK × 7 时钟
10-8	t_ceoe[2:0]	R	到$\overline{OE}$判断提示的延迟周期时间 001 ~ 111 : SMCCLK × 1 时钟 ~ SMCCLK × 7 时钟
7-4	t_wc[3:0]	R	写入周期时间 0011 ~ 1111 : SMCCLK × 3 时钟 ~ SMCCLK × 15 时钟
3-0	t_rc[3:0]	R	读取周期时间 0010 ~ 1111 : SMCCLK × 2 时钟 ~ SMCCLK × 15 时钟

注:多元总线模式下不支持页访问。t_{PC} 只在独立总线模式下有效。

10.3.9 smc_sram_cycles0_2 (SMC SRAM周期寄存器0 <2>)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	未定义	未定义	未定义	未定义	未定义	未定义	未定义	未定义
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	t_tr			t_pc
复位后	未定义	未定义	未定义	未定义	0	0	1	0
	15	14	13	12	11	10	9	8
比特符号	t_pc		t_wp			t_ceoe		
复位后	1	0	1	1	0	0	1	1
	7	6	5	4	3	2	1	0
比特符号	t_wc				t_rc			
复位后	1	1	0	0	1	1	0	0

位	比特符号	类型	功能
31-20	–	W	写作 “0“。
19-17	t_tr[2:0]	R	扭转周期时间 001 ~ 111:SMCCLK × 1 时钟 ~ SMCCLK × 7 时钟
16-14	t_pc[2:0]	R	PAGE 周期时间 (注释) 001 ~ 111:SMCCLK × 1 时钟 ~ SMCCLK × 7 时钟
13-11	t_wp[2:0]	R	$\overline{WE}$脉冲周期时间 001 ~ 111:SMCCLK × 1 时钟 ~ SMCCLK × 7 时钟
10-8	t_ceoe[2:0]	R	到$\overline{OE}$判断提示的延迟周期时间 001 ~ 111:SMCCLK × 1 时钟 ~ SMCCLK × 7 时钟
7-4	t_wc[3:0]	R	写入周期时间 0011 ~ 1111:SMCCLK × 3 时钟 ~ SMCCLK × 15 时钟
3-0	t_rc[3:0]	R	读取周期时间 0010 ~ 1111:SMCCLK × 2 时钟 ~ SMCCLK × 15 时钟

注:多元总线模式下不支持页访问。t_{PC}只在独立总线模式下有效。

10.3.10 smc_sram_cycles0_3 (SMC SRAM周期寄存器0 <3>)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	未定义	未定义	未定义	未定义	未定义	未定义	未定义	未定义
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	t_tr			t_pc
复位后	未定义	未定义	未定义	未定义	0	0	1	0
	15	14	13	12	11	10	9	8
比特符号	t_pc		t_wp			t_ceoe		
复位后	1	0	1	1	0	0	1	1
	7	6	5	4	3	2	1	0
比特符号	t_wc				t_rc			
复位后	1	1	0	0	1	1	0	0

位	比特符号	类型	功能
31-20	–	W	写作 “0 “。
19-17	t_tr[2:0]	R	扭转周期时间 001 ~ 111 : SMCCLK × 1 时钟 ~ SMCCLK × 7 时钟
16-14	t_pc[2:0]	R	PAGE 周期时间 (注) 001 ~ 111 : SMCCLK × 1 时钟 ~ SMCCLK × 7 时钟
13-11	t_wp[2:0]	R	$\overline{WE}$脉冲周期时间 001 ~ 111 : SMCCLK × 1 时钟 ~ SMCCLK × 7 时钟
10-8	t_ceoe[2:0]	R	到$\overline{OE}$判断提示的延迟周期时间 001 ~ 111 : SMCCLK × 1 时钟 ~ SMCCLK × 7 时钟
7-4	t_wc[3:0]	R	写入周期时间 0011 ~ 1111 : SMCCLK × 3 时钟 ~ SMCCLK × 15 时钟
3-0	t_rc[3:0]	R	读取周期时间 0010 ~ 1111 : SMCCLK × 2 时钟 ~ SMCCLK × 15 时钟

注:多元总线模式下不支持页访问。t_{PC}只在独立总线模式下有效。

10.3.11 smc_opmode0_0 (SMC 输入模式寄存器 0<0>)

	31	30	29	28	27	26	25	24
比特符号	address_match							
复位后	0	1	1	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	1	1	1	1	1	1	1	1
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	adv	-	-	-
复位后	未定义	未定义	未定义	未定义	1	未定义	未定义	未定义
	7	6	5	4	3	2	1	0
比特符号	-	-	rd_bl			-	mw	
复位后	未定义	未定义	0	0	0	未定义	1	0

位	比特符号	类型	功能
31-24	address_match [7:0]	R	CS0 区域的启动地址 读作 “0×60“。
23-16	−	R	读作 “0×FF“。
15-12	−	R	以未定义读入。
11	adv	R	地址锁存启用信号 0：地址锁存启用信号 ($\overline{ALE}$)未使用 1：使用地址锁存启用信号 ($\overline{ALE}$)
10-6	−	R	以未定义读入。
5-3	rd_bl[2:0]	R	数据读取的脉冲串长度 000：1 节拍 001：4 节拍 010 ~ 111：忽略
2	−	R	以未定义读入。
1-0	mw[1:0]	R	CS0 的数据总线宽度 01:16 位 其他：忽略

注:出设置 CS 区域外，不得访问外部存储器区域。

10.3.12 smc_opmode0_1 (SMC输入模式寄存器0<1>)

	31	30	29	28	27	26	25	24
比特符号	address_match							
复位后	0	1	1	0	0	0	0	1
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	1	1	1	1	1	1	1	1
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	adv	-	-	-
复位后	未定义	未定义	未定义	未定义	1	未定义	未定义	未定义
	7	6	5	4	3	2	1	0
比特符号	-	-	rd_bl			-	mw	
复位后	未定义	未定义	0	0	0	未定义	1	0

位	比特符号	类型	功能
31-24	address_match [7:0]	R	CS1 区域的启动地址 读作 “0×61“。
23-16	−	R	读作“0×FF“。
15-12	−	R	以未定义读入。
11	adv	R	地址锁存启用信号 0：地址锁存启用信号 ($\overline{ALE}$)未使用 1：使用地址锁存启用信号 ($\overline{ALE}$)
10-6	−	R	以未定义读入。
5-3	rd_bl[2:0]	R	数据读取的脉冲串长度 000：1 节拍 001：4 节拍 010 ~ 111:忽略
2	−	R	以未定义读入。
1-0	mw[1:0]	R	CS1 的数据总线宽度 01:16 位 其他:忽略

注:出设置 CS 区域外，不得访问外部存储器区域。

10.3.13 smc_opmode0_2 (SMC输入模式寄存器0<2>)

	31	30	29	28	27	26	25	24
比特符号	address_match							
复位后	0	1	1	0	0	0	1	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	1	1	1	1	1	1	1	1
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	adv	-	-	-
复位后	未定义	未定义	未定义	未定义	1	未定义	未定义	未定义
	7	6	5	4.	3	2	1	0
比特符号	-	-	rd_bl			-	mw	
复位后	未定义	未定义	0	0	0	未定义	1	0

位	比特符号	类型	功能
31-24	address_match [7:0]	R	CS2 区域的启动地址 读作“0×62“。
23-16	−	R	读作“0×FF“。
15-12	−	R	以未定义读入。
11	adv	R	地址锁存启用信号 0：地址锁存启用信号 ($\overline{ALE}$)未使用 1：使用地址锁存启用信号 ($\overline{ALE}$)
10-6	−	R	以未定义读入。
5-3	rd_bl[2:0]	R	数据读取的脉冲串长度 000 :1 节拍 001 : 4 节拍 010 ~ 111:忽略
2	−	R	以未定义读入。
1-0	mw[1:0]	R	CS2 的数据总线宽度 01 : 16 位 其它： 忽略

注:出设置 CS 区域外，不得访问外部存储器区域。

10.3.14 smc_opmode0_3 (SMC输入模式寄存器0<3>)

	31	30	29	28	27	26	25	24
比特符号	address_match							
复位后	0	1	1	0	0	0	1	1
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	1	1	1	1	1	1	1	1
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	adv	-	-	-
复位后	未定义	未定义	未定义	未定义	1	未定义	未定义	未定义
	7	6	5	4	3	2	1	0
比特符号	-	-	rd_bl			-	mw	
复位后	未定义	未定义	0	0	0	未定义	1	0

位	比特符号	类型	功能
31-24	address_match [7:0]	R	CS3 区域的启动地址 读作 “0×63“。
23-16	−	R	读作 “0×FF“。
15-12	−	R	以未定义读入。
11	adv	R	地址锁存启用信号 0：地址锁存启用信号 ($\overline{ALE}$)未使用 1：使用地址锁存启用信号 ($\overline{ALE}$)
10-6	−	R	以未定义读入。
5-3	rd_bl[2:0]	R	数据读取的脉冲串长度 000：1 节拍 001：4 节拍 010 ~ 111:忽略
2	−	R	以未定义读入。
1-0	mw[1:0]	R	CS3 的数据总线宽度 01：16 位 其他:忽略

注:出设置 CS 区域外，不得访问外部存储器区域。

10.4 外部总线周期

10.4.1 独立总线模式

10.4.1.1 t_{RC} / t_{CEOE}设置示例

t_{RC} =3, t_{CEOE} =1 (smc_set_cycles = 0×0002B1C3)

		t_{TR}	t_{PC}	t_{wp}	t_{CEOE}	t_{WC}	t_{RC}
smc_set_cycles	31-20	19-17	16-14	13-11	10-8	7-4	3-0
	-	Set_t5[2:0]	Set_t4[2:0]	Set_t3[2:0]	Set_t2[2:0]	Set_t1[3:0]	Set_t0[3:0]
设置值	0	001 (1)	010 (2)	110 (6)	001 (1)	1100 (C)	0011 (3)

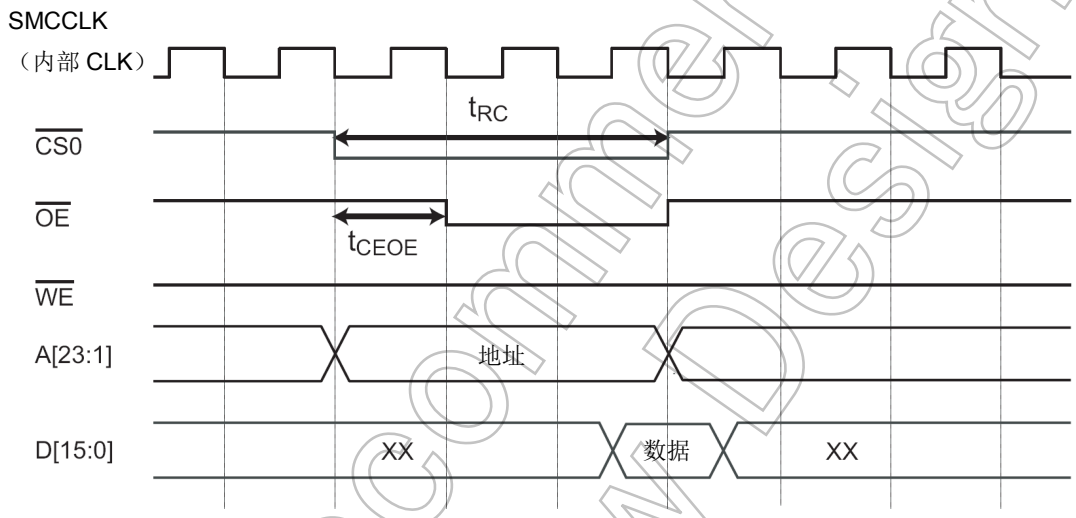


图 10-2 异步读取

10.4.1.2 t_{WC} / t_{WP}设置示例

t_{WC} =4, t_{WP} =2 (smc_set_cycles = 0×0002934C)

		t_{TR}	t_{PC}	t_{wp}	t_{CEOE}	t_{WC}	t_{RC}
smc_set_cycles	31-20	19-17	16-14	13-11	10-8	7-4	3-0
	-	Set_t5[2:0]	Set_t4[2:0]	Set_t3[2:0]	Set_t2[2:0]	Set_t1[3:0]	Set_t0[3:0]
设置	0	001 (1)	010 (2)	010 (2)	011 (3)	0100 (4)	1100 (C)

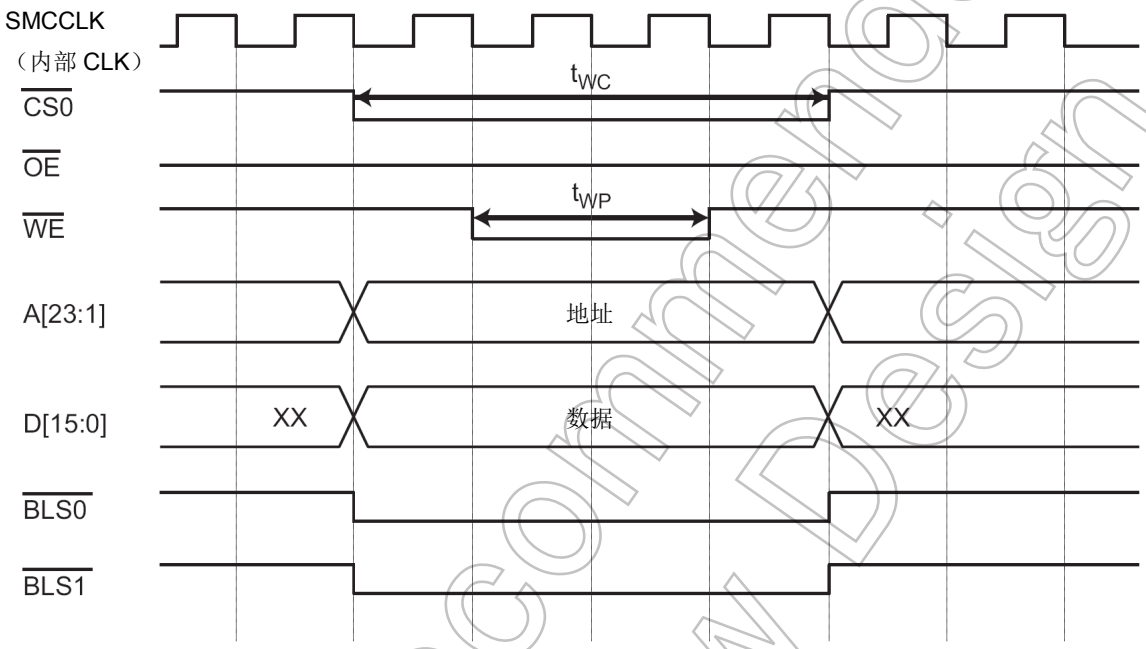


图 10-3 异步写入

10.4.1.3 t_{RC} / t_{CEOE} / t_{PC}设置示例

t_{RC} =3, t_{CEOE} =2, t_{PC} =1 (smc_set_cycles = 0×000272C3)

		t_{TR}	t_{PC}	t_{wp}	t_{CEOE}	t_{WC}	t_{RC}
smc_set_cycles	31-20	19-17	16-14	13-11	10-8	7-4	3-0
	-	Set_t5[2:0]	Set_t4[2:0]	Set_t3[2:0]	Set_t2[2:0]	Set_t1[3:0]	Set_t0[3:0]
设置值	0	001 (1)	001 (1)	110 (6)	010 (2)	1100 (C)	0011 (3)

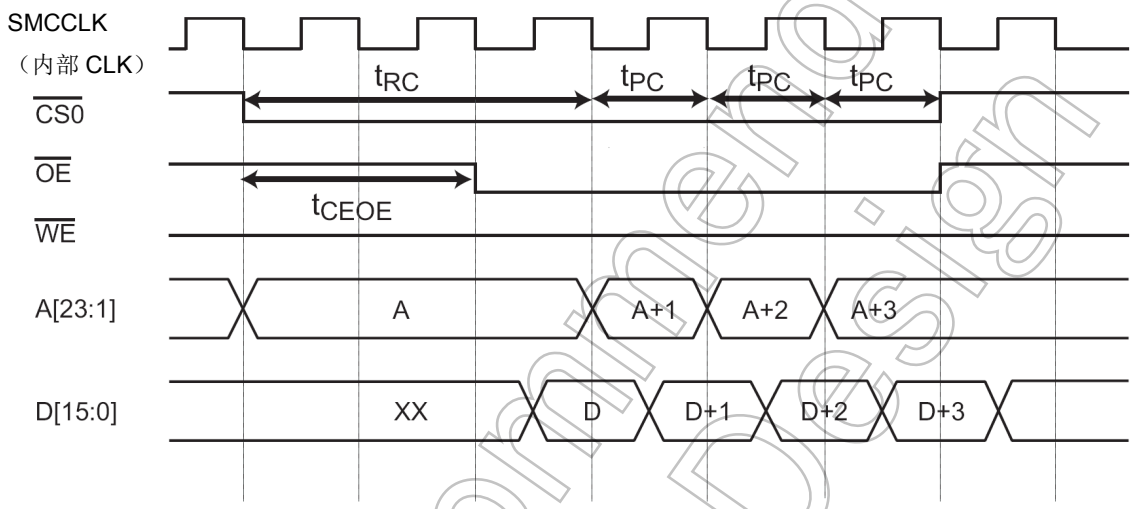


图 10-4 异步页读取

10.4.1.4 t_{TR}设置示例

t_{TR} =1 (smc_set_cycles = 0×00029143)

		t_{TR}	t_{PC}	t_{wp}	t_{CEOE}	t_{WC}	t_{RC}
smc_set_cycles	31-20	19-17	16-14	13-11	10-8	7-4	3-0
	-	Set_t5[2:0]	Set_t4[2:0]	Set_t3[2:0]	Set_t2[2:0]	Set_t1[3:0]	Set_t0[3:0]
设置值	0	001 (1)	010 (2)	010 (2)	001 (1)	0100 (4)	0011 (3)

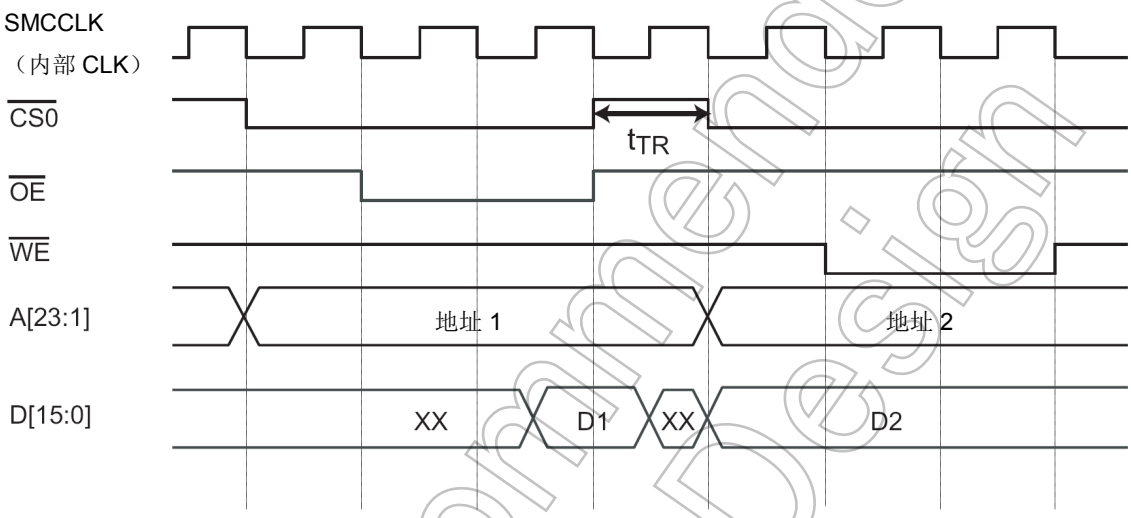


图 10-5 异步读取和异步写入

10.4.2 多元模式

10.4.2.1 t_{RC}/t_{CEOE}设置示例

t_{RC} =4, t_{CEOE} =1 (smc_set_cycles = 0×0002B1C4)

		t_{TR}	t_{PC}	t_{wp}	t_{CEOE}	t_{WC}	t_{RC}
smc_set_cycles	31-20	19-17	16-14	13-11	10-8	7-4	3-0
	-	Set_t5[2:0]	Set_t4[2:0]	Set_t3[2:0]	Set_t2[2:0]	Set_t1[3:0]	Set_t0[3:0]
设置值	0	001 (1)	010 (2)	110 (6)	001 (1)	1100 (C)	0100 (4)

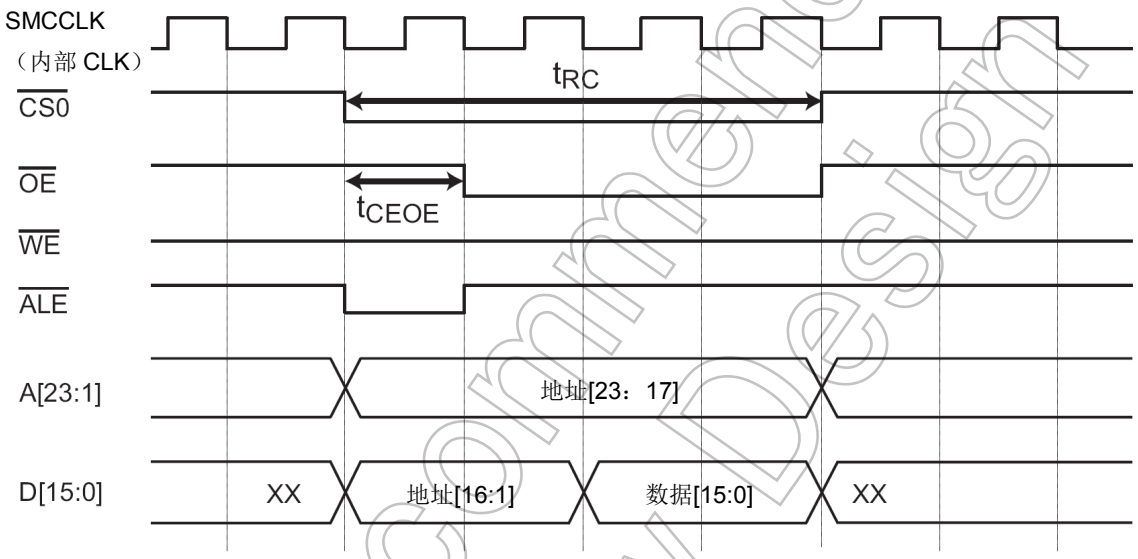


图 10-6 异步读取

10.4.2.2 t_{WC} / t_{WP}设置示例

t_{WC} =5, t_{WP} =1 (smc_set_cycles = 0×00028B5C)

		t_{TR}	t_{PC}	t_{wp}	t_{CEOE}	t_{WC}	t_{RC}
smc_set_cycles	31-20	19-17	16-14	13-11	10-8	7-4	3-0
	-	Set_t5[2:0]	Set_t4[2:0]	Set_t3[2:0]	Set_t2[2:0]	Set_t1[3:0]	Set_t0[3:0]
设置值	0	001 (1)	010 (2)	001 (1)	011 (3)	0101 (5)	1100 (C)

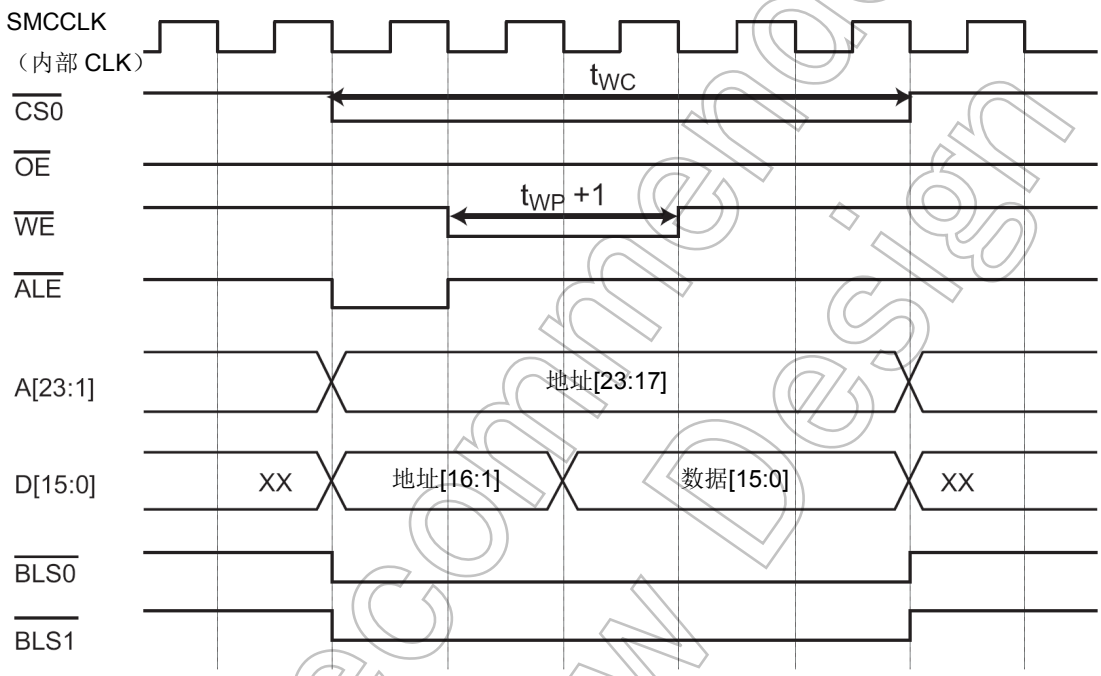


图 10-7 异步写入

10.4.2.3 t_{TR}设置示例

t_{TR} =1 (smc_set_cycles = 0×00029144)

		t_{TR}	t_{PC}	t_{wp}	t_{CEOE}	t_{WC}	t_{RC}
smc_set_cycles	31-20	19-17	16-14	13-11	10-8	7-4	3-0
	-	Set_t5[2:0]	Set_t4[2:0]	Set_t3[2:0]	Set_t2[2:0]	Set_t1[3:0]	Set_t0[3:0]
设置值	0	001 (1)	010 (2)	010 (2)	001 (1)	0100 (4)	0100 (4)

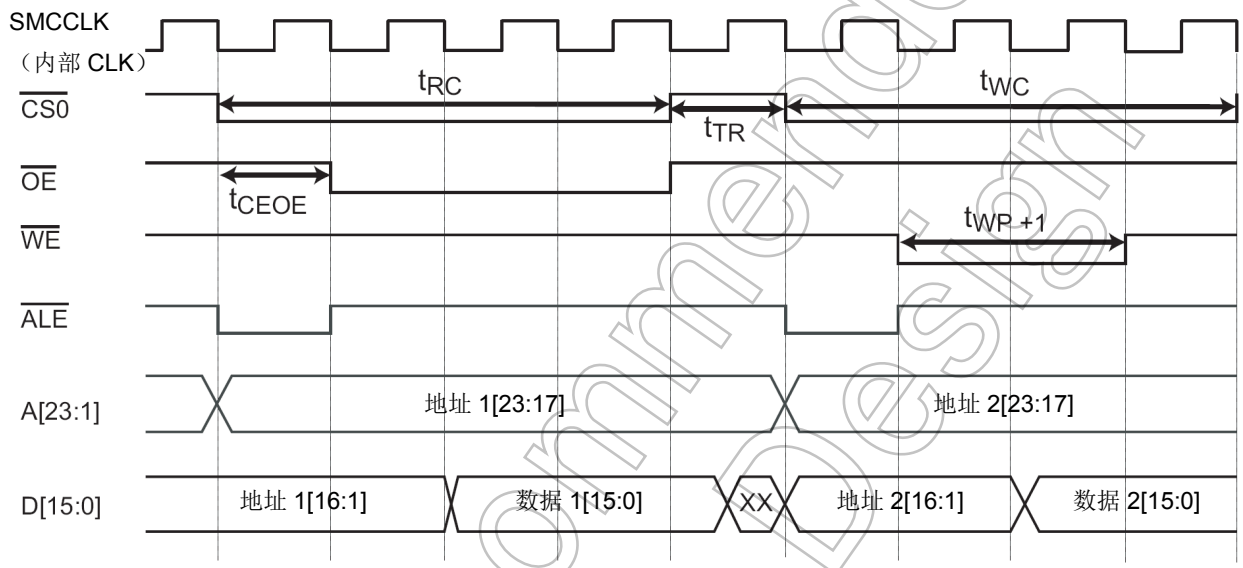


图 10-8 异步读取和异步写入

10.5 外部存储器连接举例

以下的图例为外部 16 位 NOR-Flash 和 16 位 SRAM 的连接示例。

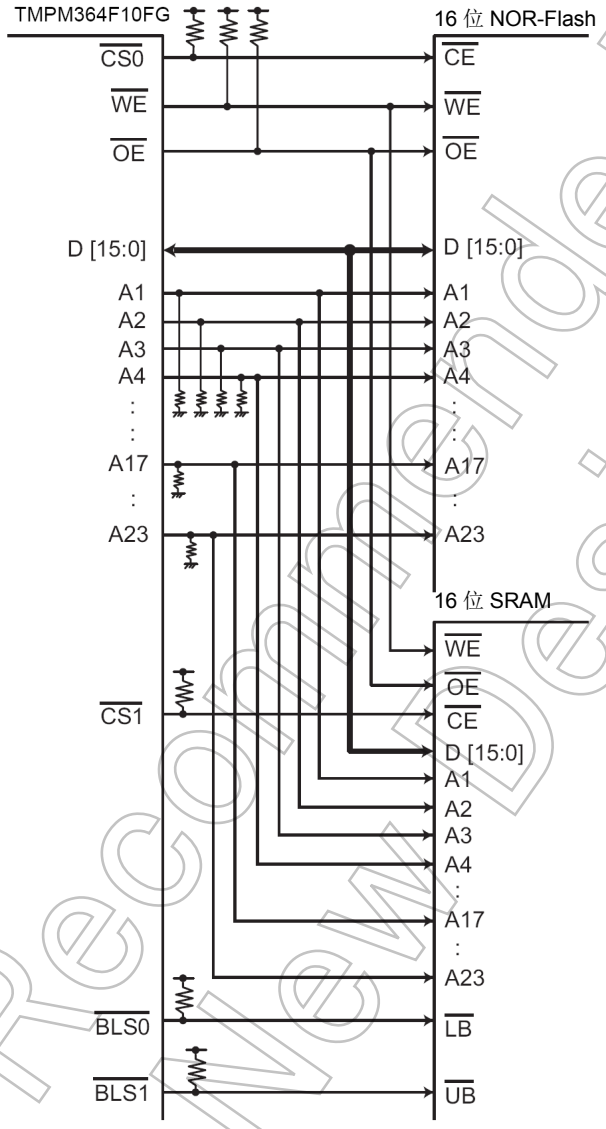


图 10-9 外部 16 位 SRAM 和 16 位 NOR-Flash 的连接举例 (独立模式)

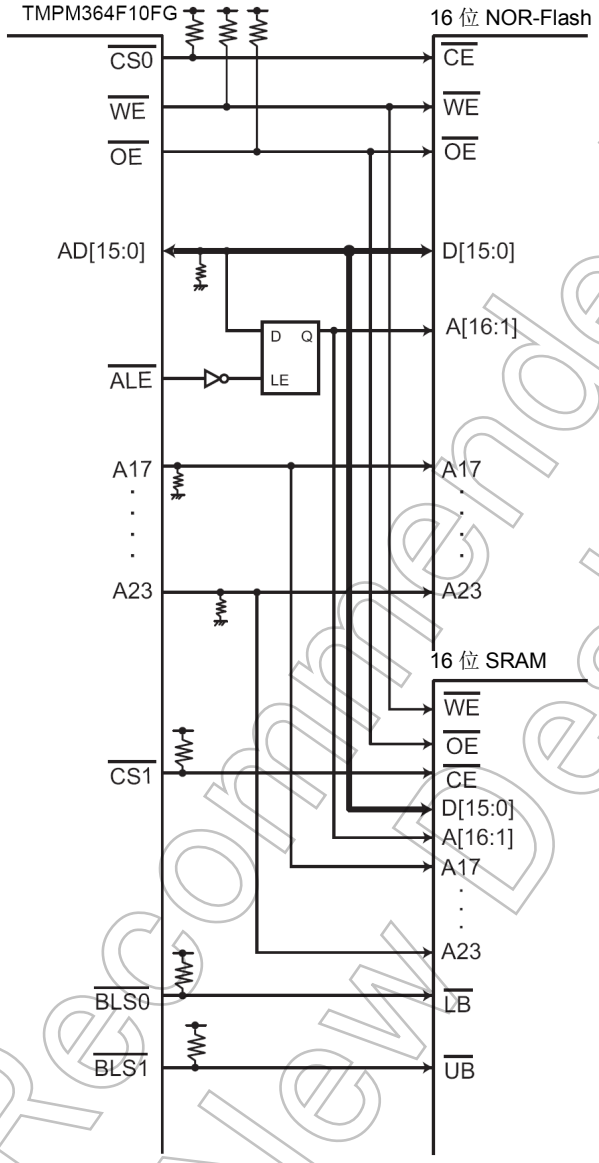


图 10-10 外部 16 位 SRAM 和 16 位 NOR-Flash 的连接举例 (多元模式)

11. 16-位计时器 / 事件计数器 (TMRB)

11.1 概述

TMRB 在以下四个操作模式下操作:

- 16-位区间计时器模式
- 16-位事件计数器模式
- 16-位可编程脉冲生成模式 (PPG)
- 计时器同步方式

对捕获功能的使用使 TMRB 操作以下三个测量。

- 一个通过外部触发器的触发脉冲输出
- 频率测量
- 脉冲宽度测量
- 时差测量

在后文中对此部分的说明， “×“表示一个通道号。

11.2 规格差异

TMPM364F10FG 包含 TMRB 的 16-通道

各通道单独起作用，除表 11-1 中所述规格差异外，通道的操作方式都相同。

有些通道可将捕获触发器和同步启动触发器置于其他通道上。

1. TMRB 0，4，8 和 C 的触发输出可用作其他通道的捕获触发器。
   - TB0OUT → 可通过 TMRB5 到 TMRB7 获取
   - TB4OUT → 可通过 TMRB1 到 TMRB3 获取
   - TB8OUT → 可通过 TMRBD 到 TMRBF 获取
   - TBCOUT → 可通过 TMRB9 到 TMRBB 获取
2. 定时器同步方式的启动触发器 (带 TBxRUN)
   - TMRB0 → 可通过 TMRB0 到 TMRB3 同步启动
   - TMRB4 → 可通过 TMRB4 到 TMRB7 同步启动
   - TMRB8 → 可通过 TMRB8 到 TMRBB 同步启动
   - TMRBC → 可通过 TMRBC 到 TMRBF 同步启动

表 11-1 TMRB 模块规格的差异

规格 通道	外部引脚		计时器之间的触发功能		中断	
	外部时钟 / 捕获触发器输入引脚	计时触发器输出引脚	捕获触发器	同步启动触发器通道	捕获中断	TMRB 中断
	信号	信号				
TMRB0	-	TB0OUT	-	TMRB0	-	INTTB0
TMRB1	TB1IN0 TB1IN1	TB1OUT	TB4OUT	TMRB0	INTCAP10 INTCAP11	INTTB1
TMRB2	TB2IN0 TB2IN1	TB2OUT	TB4OUT	TMRB0	INTCAP20 INTCAP21	INTTB2
TMRB3	- (连接到 SCLK3)	TB3OUT	TB4OUT	TMRB0	-	INTTB3
TMRB4	-	TB4OUT	-	TMRB4	-	INTTB4
TMRB5	TB5IN0 TB5IN1	TB5OUT (连接到 SIO0~ SIO3)	TB0OUT	TMRB4	INTCAP50 INTCAP51	INTTB5
TMRB6	TB6IN0 TB6IN1	TB6OUT (连接到 SIO4 ~ SIO7)	TB0OUT	TMRB4	INTCAP60 INTCAP61	INTTB6
TMRB7	TB7IN0 TB7IN1	TB7OUT	TB0OUT	TMRB4	INTCAP70 INTCAP71	INTTB7
TMRB8	-	TB8OUT	-	TMRB8	-	INTTB8
TMRB9	TB9IN0 TB9IN1	TB9OUT (连接到 SIO8 ~ SIO11)	TBCOUT	TMRB8	INTCAP90 INTCAP91	INTTB9
TMRBA	TBAIN0 TBAIN1	TBAOUT (连接到 CEC)	TBCOUT	TMRB8	INTCAPA0 INTCAPA1	INTTBA
TMRBB	TBBIN0 TBBIN1	TBBOUT (连接到 RMC)	TBCOUT	TMRB8	INTCAPB0 INTCAPB1	INTTBB
TMRBC	-	TBCOUT	-	TMRBC	-	INTTBC
TMRBD	TBDIN0 TBDIN1	TBDOUT	TB8OUT	TMRBC	INTCAPD0 INTCAPD1	INTTBD
TMRBE	TBEIN0 TBEIN1	TBEOUT	TB8OUT	TMRBC	INTCAPE0 INTCAPE1	INTTBE
TMRBF	TBFIN0 TBFIN1	TBFOUT	TB8OUT	TMRBC	INTCAPF0 INTCAPF1	INTTBF

11.3 配置

每个通道都包含一个 16-位上升计数器，两个 16-位计时寄存器 (双缓冲)，两个 16-位捕获寄存器，两个比较仪，一个捕获输入控制器，一个即时触发器和与其相连的控制电路。计时操作模式和计时触发器由一个寄存器控制。

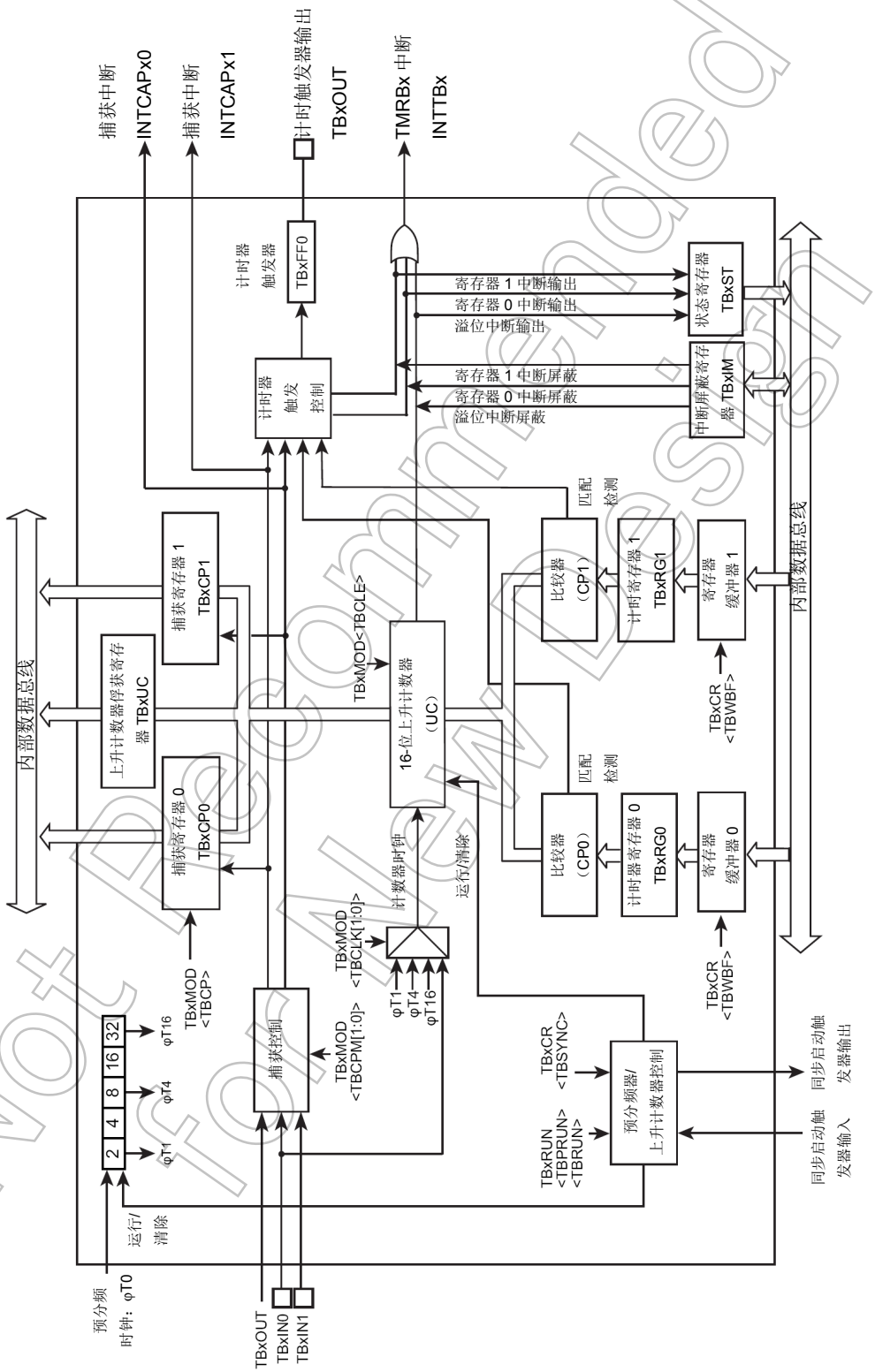


图 11-1 TMRBx 方块图 (x= 0 ~ 2，4 ~ F)

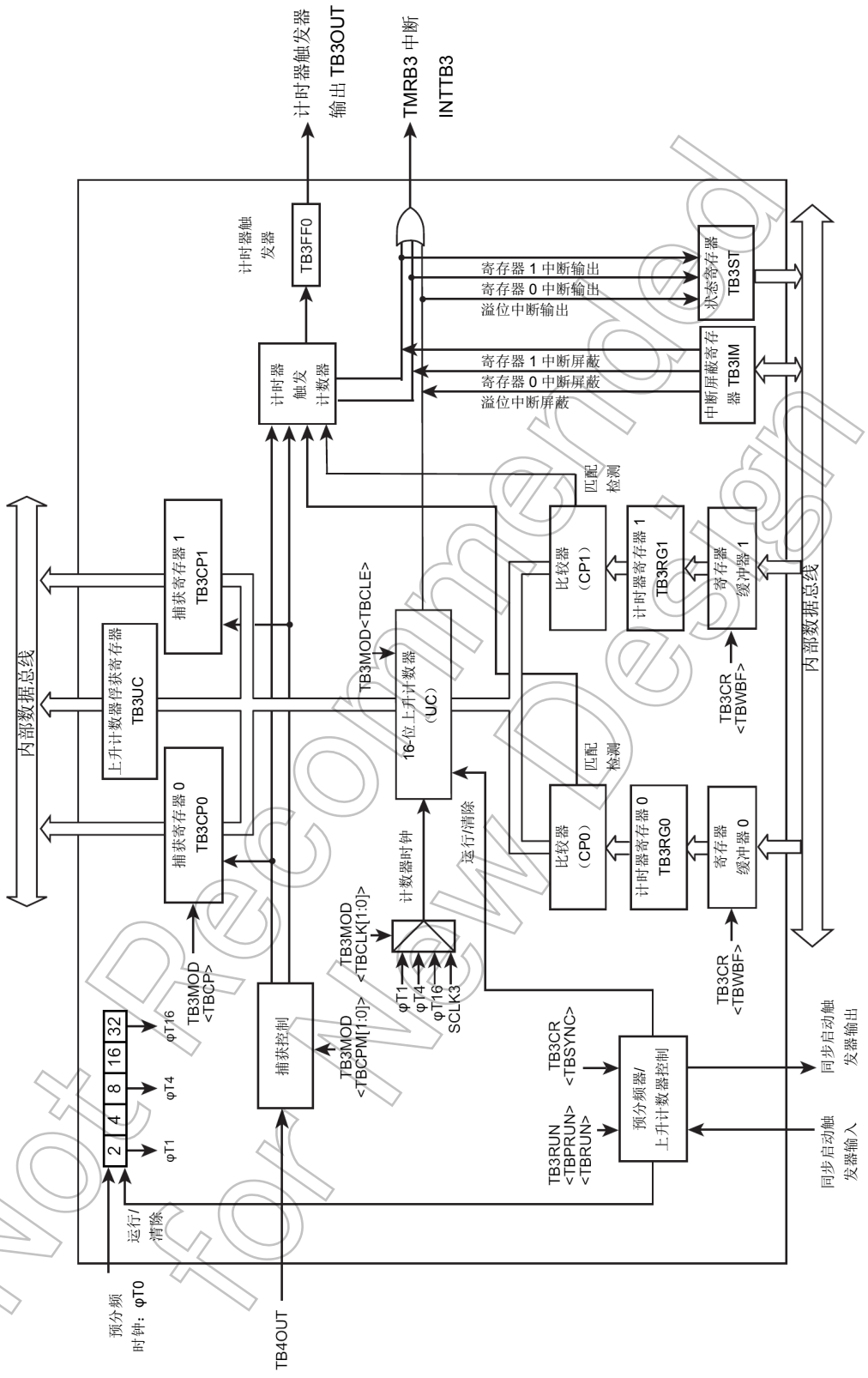


图 11-2 TMRBx 方块图 (x= 3)

11.4 寄存器

11.4.1 对应于通道的寄存器列表

以下表格显示各通道对应的寄存器名称和地址。

通道 x	基址
通道 0	0x400D_0000
通道 1	0x400D_0100
通道 2	0x400D_0200
通道 3	0x400D_0300
通道 4	0x400D_0400
通道 5	0x400D_0500
通道 6	0x400D_0600
通道 7	0x400D_0700
通道 8	0x400D_0800
通道 9	0x400D_0900
通道 A	0x400D_0A00
通道 B	0x400D_0B00
通道 C	0x400D_0C00
通道 D	0x400D_0D00
通道 E	0x400D_0E00
通道 F	0x400D_0F00

寄存器名称 (x=0 ~ F)		地址 (基址+)
启用寄存器	TBxEN	0x0000
RUN 寄存器	TBxRUN	0x0004
控制寄存器	TBxCR	0x0008
模式寄存器	TBxMOD	0x000C
触发控制寄存器	TBxFFCR	0x0010
状态寄存器	TBxST	0x0014
中断屏蔽寄存器	TBxIM	0x0018
上升计数器捕获寄存器	TBxUC	0x001C
计时器寄存器 0	TBxRG0	0x0020
计时器寄存器 1	TBxRG1	0x0024
捕获寄存器 0	TBxCP0	0x0028
捕获寄存器 1	TBxCP1	0x002C

11.4.2 TBxEN (启用寄存器)

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

位	比特符号	类型	功能
31-8	−	R	读作 “0“。
7	TBEN	R/W	TMRBx 操作 0：禁用 1：启用 规定 TMRB 操作。当操作被禁用时，TMRB 模块中的寄存器没有时钟，以减少能量消耗。 (此操作禁用对其他寄存器的读取和写入，TBxEN 寄存器除外。) 使用 TMRB 时，在 TMRB 模块中为每个寄存器编程前要启用 TMRB 操作 (设置为 “1“)。当执行 TMRB 操作后禁用时，每个寄存器中的设置将被保持。
6-0	−	R	读作 “0“。

11.4.3 TBxRUN (RUN寄存器)

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

位	比特符号	类型	功能
31-3	−	R	读作 “0“。
2	TBPRUN	R/W	预分频器操作 0：停止&清除 1：计数
1	−	R	读作 “0“。
0	TBRUN	R/W	计数运作 0：停止 & 清除 1：计数

注:当计数器停止 (<TBRUN>= “0“)且 TBxUC<TBUC[15:0]>被读取时，在计数器操作中被捕获的值将被读取。

11.4.4 TBxCR (控制寄存器)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	TBWBF	-	TBSYNC	-	I2TB	-	-	-
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-8	−	R	读作 “0“。
7	TBWBF	R/W	双缓冲 0:禁用 1:启用
6	−	R/W	写作 “0“。
5	TBSYNC	R/W	同步模式切换 0：单个 (通道单元) 1：同步
4	−	R	读作 “0“。
3	I2TB	R/W	在 IDLE 模式下操作 0：停止 1：操作
2	−	R	读作 “0“。
1-0	−	R/W	写作 “0“。

注:操作 TMRB 时勿改变 TBxCR。

11.4.5 TBxMOD (模式寄存器)

x=0 ~ 2, 4 ~ F

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	-	-	TBCP	TBCPM		TBCLE	TBCLK	
复位后	0	0	1	0	0	0	0	0

位	比特符号	类型	功能
31-7	−	R	读作 “0“。
6	−	R/W	写作 “0“。
5	TBCP	W	通过软件捕获控制 0: 通过软件捕获 1: 忽略 写入 “0“时，捕获寄存器 0 (TBxCP0)得出计数值。 读作 “1“。
4-3	TBCPM[1:0]	R/W	捕获计时 00: 禁用 01: TBxIN0↑ TBxIN1↑ 在 TBxIN0 引脚输入上升时，将计数值计入捕获寄存器 0 (TBxCP0)。 在 TBxIN1 引脚输入上升时，将计数值计入捕获寄存器 1 (TBxCP1)。 10: TBxIN0↑ TBxIN0↓ 在 TBxIN0 引脚输入上升时，将计数值计入捕获寄存器 0 (TBxCP0)。 在 TBxIN0 引脚输入下降时，将计数值计入捕获寄存器 1 (TBxCP1)。 11: TBxOUT↑ TBxOUT↓ 在 16 位计时匹配输出 (TBxOUT)的上升时，将计数值计入捕获寄存器 0 (TBxCP0)，而且在 TBxOUT 下降时将计数值计入捕获寄存器 1 (TBxCP1)。 (TMRB1 ~ 3: TB4OUT，TMRB5 ~ 7: TB0OUT，TMRB9 ~ B: TBCOUT，TMRBD ~ F: TB8OUT)
2	TBCLE	R/W	上升计数器控制 0: 上升计数器禁用清除。 1: 上升计数器启用清除。 对上升计数器进行清除和控制。 当 “0“被写入时，禁用清除上升计数器。若 “1“被写入，当匹配计时器寄存器 1 (TBxRG1)时清除上升计数器。
1-0	TBCLK[1:0]	R/W	选择 TMRBx 源时钟。 00: TBxIN0 引脚输入 01:φT1 10:φT4 11:φT16

注:TMRB0, 3, 4, 8 和 A 没有 TBxIN0 输入和 TBxIN1 输入。

x=3

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

位	比特符号	类型	功能
31-7	−	R	读作 “0“。
6	−	R/W	写作 “0“。
5	−	W	写入数值为 “1“。
4-3	−	R/W	以 “00“写入。
2	TBCLE	R/W	上升计数器控制 0: 上升计数器禁用清除。 1: 上升计数器启用清除。 对上升计数器进行清除和控制。 当 “0“被写入时，禁用清除上升计数器。若 “1“被写入，当匹配计时器寄存器 1 (TBxRG1)时清除上升计数器。
1-0	TBCLK[1:0]	R/W	选择 TMRBx 源时钟。 00: SCLK3 引脚输入 01 :φT1 10: φT4 11: φT16

11.4.6 TBxFFCR (触发控制寄存器)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	-	-	TBC1T1	TBC0T1	TBE1T1	TBE0T1	TBFF0C	
复位后	1	1	0	0	0	0	1	1

位	比特符号	类型	功能
31-8	−	R	读作 “0“。
7-6	−	R	读作 “1“。
5	TBC1T1	R/W	当上升计数器的值被计入 TBxCP1 时，TBxFF0 反向触发。 0 :禁用触发 1: 启用触发 通过设置 “1“，当上升计数器值被计入捕获寄存器 1 (TBxCP1)时，计时反向触发。
4.	TBC0T1	R/W	当上升计数器的值被计入 TBxCP01 时，TBxFF0 反向触发。 0 :禁用触发 1: 启用触发 通过设置 “1“，当上升计数器值被计入捕获寄存器 0 (TBxCP0)时，计时反向触发。
3	TBE1T1	R/W	当上升计数器的值与 TBxRG1 匹配时，TBxFF0 反向触发。 0: 禁用触发 1: 启用触发 通过设置 “1“，当上升计数器值与计时寄存器 1 (TBxRG1)匹配时，计时触发反向。
2	TBE0T1	R/W	当上升计数器的值与 TBxRG0 匹配时，TBxFF0 反向触发。 0: 禁用触发 1: 启用触发 通过设置 “1“，则当一上升计数器值与计时寄存器 0 (TBxRG0)匹配时，计时反向触发。
1-0	TBFF0C[1:0]	R/W	TBxFF0 控制 00:逆转 将 TBxFF0 的值逆转 (使用软件进行逆转)。 01: 设置 设置 TBxFF0 到 “1“。 10:清除 将 TBxFF0 清 “0“。 11: 忽略 * 总是读作 “11“。

注:操作 TMRB 时勿改变 TBxFFCR。

11.4.7 TBxST (状态寄存器)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	-	-	-	-	-	INTTBOF	INTTB1	INTTB0
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-3	−	R	读作 “0“。
2	INTTBOF	R	溢出标志 0: 无溢出发生 1: 溢出发生 当一上升计数器有溢出时，设置 “1“。
1	INTTB1	R	匹配标志 (TBxRG1) 0: 匹配未检测 1: 检测带 TBxRG1 的匹配 当检测到计时寄存器 1 (TBxRG1)的匹配时，设置 “1“。
0	INTTB0	R	匹配标志 (TBxRG0) 0: 未检测到匹配 1: 检测到 TBxRG0 的匹配 当检测到计时寄存器 0 (TBxRG0)的匹配时，设置 “1“。

注 1: 只有那些未被 TBxIM 屏蔽的因素会向 CPU 输出中断请求。即使屏蔽设置被完成，标志也进行设置。

注 2: 通过读取 TBxST 寄存器，该标志被清除。要清除此标志，应读取 TBxST 寄存器。

11.4.8 TBxIM (中断屏蔽寄存器)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	-	-	-	-	-	TBIMOF	TBIM1	TBIM0
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-3	−	R	读作 “0“。
2	TBIMOF	R/W	溢位中断屏蔽 0: 禁用 1: 启用 设置上升计数器溢出中断来启用或禁用。
1	TBIM1	R/W	匹配中断屏蔽 (TBxRG1) 0: 禁用 1: 启用 用计时器寄存器 1 (TBxRG1)来设置匹配中断屏蔽，来启用或禁用。
0	TBIM0	R/W	匹配中断屏蔽 (TBxRG0) 0: 禁用 1: 启用 通过计时器寄存器 0 (TBxRG0)设置匹配中断屏蔽，进行启用或禁用。

注:即使 TBxIM 设置完成，TBxST 也进行设置。

11.4.9 TBxUC (上升计数器捕获寄存器)

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

位	比特符号	类型	功能
31-16	−	R	读作 “0“。
15-0	TBUC[15:0]	R	通过读出上升计数器来捕获数值。 如果 TBxUC 被读取时，可捕获当前上升计数器数值。

注:当操作计数器且读取 TBxUC 时，上升计数器的数值被捕获并读取。

11.4.10 TBxRG0 (计时器寄存器 0)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	TBRG0							
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	TBRG0							
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-16	–	R	读作 “0“。
15-0	TBRG0[15:0]	R/W	比较上升计数器时设置数值。

11.4.11 TBxRG1 (计时器寄存器 1)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	TBRG1							
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	TBRG1							
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-16	–	R	读作 “0“。
15-0	TBRG1[15:0]	R/W	比较上升计数器时设置数值。

11.4.12 TBxCP0 (捕获寄存器 0)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	TBCP0							
复位后	未定义	未定义	未定义	未定义	未定义	未定义	未定义	未定义
	7	6	5	4	3	2	1	0
比特符号	TBCP0							
复位后	未定义	未定义	未定义	未定义	未定义	未定义	未定义	未定义

位	比特符号	类型	功能
31-16	–	R	读作 “0“。
15-0	TBCP0[15:0]	R	从上升计数器捕获的一数值被读取。

11.4.13 TBxCP1 (捕获寄存器 1)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	TBCP1							
复位后	未定义	未定义	未定义	未定义	未定义	未定义	未定义	未定义
	7	6	5	4	3	2	1	0
比特符号	TBCP1							
复位后	未定义	未定义	未定义	未定义	未定义	未定义	未定义	未定义

位	比特符号	类型	功能
31-16	–	R	读作 “0“。
15-0	TBCP1[15:0]	R	从上升计数器捕获的一数值被读取。

11.5 各电路操作说明

除规格不同外，各通道都以相同的方式操作，如下表 11-1 所示。

11.5.1 预分频器

有一个 4-位预分频器用于生产上升计数器 UC 源时钟。

预分频器计入时钟 φT0 为 fs，fperiph/1，fperiph/2，fperiph/4，fperiph/8，fperiph/16 或 fperiph/32，由 CG 中的 CGSYSCR<FPSEL1>选定。外设时钟，fperiph，要么为 fgear，由 CG 中的 CGSYSCR<FPSEL0>来选定的时钟；要么为 fc，被钟齿轮分割之前的一个时钟。

预定标志的操作和停止将由 TBxRUN<TBPRUN>设置，写入 “1“则启动计数，写入 “0“则清除及停止计数。下表显示预定标志输出时钟分辨率。

表 11-2 预分频标志输出时钟分辨率 (fc = 64MHz, fs = 32.768kHz)

选择 φT0 CGSYSCR <FPSEL1>	选择外设时钟 CGSYSCR <FPSEL0>	选择齿轮时钟 CGSYSCR <GEAR[2:0]>	选择预分频时钟 CGSYSCR <PRCK[2:0]>	预分频输出时钟功能		
				φT1	φT4	φT16
0	0 (fgear)	000 (fc)	000 (fperiph/1)	$fc/2^1$ (0.0312 μs)	$fc/2^3$ (0.125 μs)	$fc/2^5$ (0.5 μs)
			001 (fperiph/2)	$fc/2^2$ (0.0625 μs)	$fc/2^4$ (0.25 μs)	$fc/2^6$ (1.0 μs)
			010 (fperiph/4)	$fc/2^3$ (0.125 μs)	$fc/2^5$ (0.5 μs)	$fc/2^7$ (2.0 μs)
			011 (fperiph/8)	$fc/2^4$ (0.25 μs)	$fc/2^6$ (1.0 μs)	$fc/2^8$ (4.0 μs)
			100 (fperiph/16)	$fc/2^5$ (0.5 μs)	$fc/2^7$ (2.0 μs)	$fc/2^9$ (8.0 μs)
			101 (fperiph/32)	$fc/2^6$ (1.0 μs)	$fc/2^8$ (4.0 μs)	$fc/2^{10}$ (16.0 μs)
		100 (fc/2)	000 (fperiph/1)	$fc/2^2$ (0.0625 μs)	$fc/2^4$ (0.25 μs)	$fc/2^6$ (1.0 μs)
			001 (fperiph/2)	$fc/2^3$ (0.125 μs)	$fc/2^5$ (0.5 μs)	$fc/2^7$ (2.0 μs)
			010 (fperiph/4)	$fc/2^4$ (0.25 μs)	$fc/2^6$ (1.0 μs)	$fc/2^8$ (4.0 μs)
			011 (fperiph/8)	$fc/2^5$ (0.5 μs)	$fc/2^7$ (2.0 μs)	$fc/2^9$ (8.0 μs)
			100 (fperiph/16)	$fc/2^6$ (1.0 μs)	$fc/2^8$ (4.0 μs)	$fc/2^{10}$ (16.0 μs)
			101 (fperiph/32)	$fc/2^7$ (2.0 μs)	$fc/2^9$ (8.0 μs)	$fc/2^{11}$ (32.0 μs)
		101 (fc/4)	000 (fperiph/1)	$fc/2^3$ (0.125 μs)	$fc/2^5$ (0.5 μs)	$fc/2^7$ (2.0 μs)
			001 (fperiph/2)	$fc/2^4$ (0.25 μs)	$fc/2^6$ (1.0 μs)	$fc/2^8$ (4.0 μs)
			010 (fperiph/4)	$fc/2^5$ (0.5 μs)	$fc/2^7$ (2.0 μs)	$fc/2^9$ (8.0 μs)
			011 (fperiph/8)	$fc/2^6$ (1.0 μs)	$fc/2^8$ (4.0 μs)	$fc/2^{10}$ (16.0 μs)
			100 (fperiph/16)	$fc/2^7$ (2.0 μs)	$fc/2^9$ (8.0 μs)	$fc/2^{11}$ (32.0 μs)
			101 (fperiph/32)	$fc/2^8$ (4.0 μs)	$fc/2^{10}$ (16.0 μs)	$fc/2^{12}$ (64.0 μs)
		110 (fc/8)	000 (fperiph/1)	$fc/2^4$ (0.25 μs)	$fc/2^6$ (1.0 μs)	$fc/2^8$ (4.0 μs)
			001 (fperiph/2)	$fc/2^5$ (0.5 μs)	$fc/2^7$ (2.0 μs)	$fc/2^9$ (8.0 μs)
			010 (fperiph/4)	$fc/2^6$ (1.0 μs)	$fc/2^8$ (4.0 μs)	$fc/2^{10}$ (16.0 μs)
			011 (fperiph/8)	$fc/2^7$ (2.0 μs)	$fc/2^9$ (8.0 μs)	$fc/2^{11}$ (32.0 μs)
			100 (fperiph/16)	$fc/2^8$ (4.0 μs)	$fc/2^{10}$ (16.0 μs)	$fc/2^{12}$ (64.0 μs)
			101 (fperiph/32)	$fc/2^9$ (8.0 μs)	$fc/2^{11}$ (32.0 μs)	$fc/2^{13}$ (128.8 μs)

表 11-2 预分频标志输出时钟分辨率 (fc = 64MHz, fs = 32.768kHz)

选择 φT0 CGSYSCR <FPSEL1>	选择外设时钟 CGSYSCR <FPSEL0>	选择齿轮时钟 CGSYSCR <GEAR[2:0]>	选择预分频标志时钟 CGSYSCR <PRCK[2:0]>	预定标志输出时钟功能		
				φT1	φT4	φT16
0	1 (fc)	000 (fc)	000 (fperiph/1)	$fc/2^1$ (0.0312 μs)	$fc/2^3$ (0.125 μs)	$fc/2^5$ (0.5 μs)
			001 (fperiph/2)	$fc/2^2$ (0.0625 μs)	$fc/2^4$ (0.25 μs)	$fc/2^6$ (1.0 μs)
			010 (fperiph/4)	$fc/2^3$ (0.125 μs)	$fc/2^5$ (0.5 μs)	$fc/2^7$ (2.0 μs)
			011 (fperiph/8)	$fc/2^4$ (0.25 μs)	$fc/2^6$ (1.0 μs)	$fc/2^8$ (4.0 μs)
			100 (fperiph/16)	$fc/2^5$ (0.5 μs)	$fc/2^7$ (2.0 μs)	$fc/2^9$ (8.0 μs)
			101 (fperiph/32)	$fc/2^6$ (1.0 μs)	$fc/2^8$ (4.0 μs)	$fc/2^{10}$ (16.0 μs)
		100 (fc/2)	000 (fperiph/1)	–	$fc/2^3$ (0.125 μs)	$fc/2^5$ (0.5 μs)
			001 (fperiph/2)	$fc/2^2$ (0.0625 μs)	$fc/2^4$ (0.25 μs)	$fc/2^6$ (1.0 μs)
			010 (fperiph/4)	$fc/2^3$ (0.125 μs)	$fc/2^5$ (0.5 μs)	$fc/2^7$ (2.0 μs)
			011 (fperiph/8)	$fc/2^4$ (0.25 μs)	$fc/2^6$ (1.0 μs)	$fc/2^8$ (4.0 μs)
			100 (fperiph/16)	$fc/2^5$ (0.5 μs)	$fc/2^7$ (2.0 μs)	$fc/2^9$ (8.0 μs)
			101 (fperiph/32)	$fc/2^6$ (1.0 μs)	$fc/2^8$ (4.0 μs)	$fc/2^{10}$ (16.0 μs)
		101 (fc/4)	000 (fperiph/1)	–	$fc/2^3$ (0.125 μs)	$fc/2^5$ (0.5 μs)
			001 (fperiph/2)	–	$fc/2^4$ (0.25 μs)	$fc/2^6$ (1.0 μs)
			010 (fperiph/4)	$fc/2^3$ (0.125 μs)	$fc/2^5$ (0.5 μs)	$fc/2^7$ (2.0 μs)
			011 (fperiph/8)	$fc/2^4$ (0.25 μs)	$fc/2^6$ (1.0 μs)	$fc/2^8$ (4.0 μs)
			100 (fperiph/16)	$fc/2^5$ (0.5 μs)	$fc/2^7$ (2.0 μs)	$fc/2^9$ (8.0 μs)
			101 (fperiph/32)	$fc/2^6$ (1.0 μs)	$fc/2^8$ (4.0 μs)	$fc/2^{10}$ (16.0 μs)
		110 (fc/8)	000 (fperiph/1)	–	–	$fc/2^5$ (0.5 μs)
			001 (fperiph/2)	–	$fc/2^4$ (0.25 μs)	$fc/2^6$ (1.0 μs)
			010 (fperiph/4)	–	$fc/2^5$ (0.5 μs)	$fc/2^7$ (2.0 μs)
			011 (fperiph/8)	$fc/2^4$ (0.25 μs)	$fc/2^6$ (1.0 μs)	$fc/2^8$ (4.0 μs)
			100 (fperiph/16)	$fc/2^5$ (0.5 μs)	$fc/2^7$ (2.0 μs)	$fc/2^9$ (8.0 μs)
			101 (fperiph/32)	$fc/2^6$ (1.0 μs)	$fc/2^8$ (4.0 μs)	$fc/2^{10}$ (16.0 μs)
1	*	*	*	fs/2 (61μs)	$fs/2^3$ (244 μs)	$fc/2^5$ (977 μs)

注 1: 须选择预分频标志输出时钟 φTn，以满足 φTn < fsys (故 φTn 慢于 fsys)。

注 2: 当计时器在操作时，勿改变时钟齿轮。

注 3: "."表示设置被禁止。"*" 表示忽略。

表 11-3 预分频输出时钟分辨率 (fc = 48MHz, fs = 32.768kHz)

选择 φT0 CGSYSCR <FPSEL1>	选择外设时钟 CGSYSCR <FPSEL0>	选择齿轮时钟 CGSYSCR <GEAR[2:0]>	选择预分频时钟 CGSYSCR <PRCK[2:0]>	预定标志输出时钟功能		
				φT1	φT4	φT16
0	0 (fgear)	000 (fc)	000 (fperiph/1)	$fc/2^1$ (0.04 μs)	$fc/2^3$ (0.17 μs)	$fc/2^5$ (0.66 μs)
			001 (fperiph/2)	$fc/2^2$ (0.08 μs)	$fc/2^4$ (0.33 μs)	$fc/2^6$ (1.33 μs)
			010 (fperiph/4)	$fc/2^3$ (0.17 μs)	$fc/2^5$ (0.66 μs)	$fc/2^7$ (2.67 μs)
			011 (fperiph/8)	$fc/2^4$ (0.33 μs)	$fc/2^6$ (1.33 μs)	$fc/2^8$ (5.33 μs)
			100 (fperiph/16)	$fc/2^5$ (0.66 μs)	$fc/2^7$ (2.67 μs)	$fc/2^9$ (10.67 μs)
			101 (fperiph/32)	$fc/2^6$ (1.33 μs)	$fc/2^8$ (5.33 μs)	$fc/2^{10}$ (21.33 μs)
		100 (fc/2)	000 (fperiph/1)	$fc/2^2$ (0.08 μs)	$fc/2^4$ (0.33 μs)	$fc/2^6$ (1.33 μs)
			001 (fperiph/2)	$fc/2^3$ (0.17 μs)	$fc/2^5$ (0.66 μs)	$fc/2^7$ (2.67 μs)
			010 (fperiph/4)	$fc/2^4$ (0.33 μs)	$fc/2^6$ (1.33 μs)	$fc/2^8$ (5.33 μs)
			011 (fperiph/8)	$fc/2^5$ (0.66 μs)	$fc/2^7$ (2.67 μs)	$fc/2^9$ (10.67 μs)
			100 (fperiph/16)	$fc/2^6$ (1.33 μs)	$fc/2^8$ (5.33 μs)	$fc/2^{10}$ (21.33 μs)
			101 (fperiph/32)	$fc/2^7$ (2.67 μs)	$fc/2^9$ (10.67 μs)	$fc/2^{11}$ (42.67 μs)
		101 (fc/4)	000 (fperiph/1)	$fc/2^3$ (0.17 μs)	$fc/2^5$ (0.66 μs)	$fc/2^7$ (2.67 μs)
			001 (fperiph/2)	$fc/2^4$ (0.33 μs)	$fc/2^6$ (1.33 μs)	$fc/2^8$ (5.33 μs)
			010 (fperiph/4)	$fc/2^5$ (0.66 μs)	$fc/2^7$ (2.67 μs)	$fc/2^9$ (10.67 μs)
			011 (fperiph/8)	$fc/2^6$ (1.33 μs)	$fc/2^8$ (5.33 μs)	$fc/2^{10}$ (21.33 μs)
			100 (fperiph/16)	$fc/2^7$ (2.67 μs)	$fc/2^9$ (10.67 μs)	$fc/2^{11}$ (42.67 μs)
			101 (fperiph/32)	$fc/2^8$ (5.33 μs)	$fc/2^{10}$ (21.33 μs)	$fc/2^{12}$ (85.33 μs)
		110 (fc/8)	000 (fperiph/1)	$fc/2^4$ (0.33 μs)	$fc/2^6$ (1.33 μs)	$fc/2^8$ (5.33 μs)
			001 (fperiph/2)	$fc/2^5$ (0.66 μs)	$fc/2^7$ (2.67 μs)	$fc/2^9$ (10.67 μs)
			010 (fperiph/4)	$fc/2^6$ (1.33 μs)	$fc/2^8$ (5.33 μs)	$fc/2^{10}$ (21.33 μs)
			011 (fperiph/8)	$fc/2^7$ (2.67 μs)	$fc/2^9$ (10.67 μs)	$fc/2^{11}$ (42.67 μs)
			100 (fperiph/16)	$fc/2^8$ (5.33 μs)	$fc/2^{10}$ (21.33 μs)	$fc/2^{12}$ (85.33 μs)
			101 (fperiph/32)	$fc/2^9$ (10.67 μs)	$fc/2^{11}$ (42.67 μs)	$fc/2^{13}$ (170.67μs)

表 11-3 预分频输出时钟分辨率 (fc = 48MHz, fs = 32.768kHz)

选择 φT0 CGSYSCR <FPSEL1>	选择外设时钟 CGSYSCR <FPSEL0>	选择齿轮时钟 CGSYSCR <GEAR[2:0]>	选择预分频标志时钟 CGSYSCR <PRCK[2:0]>	预定标志输出时钟功能		
				φT1	φT4	φT16
0	1 (fc)	000 (fc)	000 (fperiph/1)	$fc/2^1$ (0.04 μs)	$fc/2^3$ (0.17 μs)	$fc/2^5$ (0.66 μs)
			001 (fperiph/2)	$fc/2^2$ (0.08 μs)	$fc/2^4$ (0.33 μs)	$fc/2^6$ (1.33 μs)
			010 (fperiph/4)	$fc/2^3$ (0.17 μs)	$fc/2^5$ (0.66 μs)	$fc/2^7$ (2.67 μs)
			011 (fperiph/8)	$fc/2^4$ (0.33 μs)	$fc/2^6$ (1.33 μs)	$fc/2^8$ (5.33 μs)
			100 (fperiph/16)	$fc/2^5$ (0.66 μs)	$fc/2^7$ (2.67 μs)	$fc/2^9$ (10.67 μs)
			101 (fperiph/32)	$fc/2^6$ (1.33 μs)	$fc/2^8$ (5.33 μs)	$fc/2^{10}$ (21.33 μs)
		100 (fc/2)	000 (fperiph/1)	–	$fc/2^3$ (0.17 μs)	$fc/2^5$ (0.66 μs)
			001 (fperiph/2)	$fc/2^2$ (0.08 μs)	$fc/2^4$ (0.33 μs)	$fc/2^6$ (1.33 μs)
			010 (fperiph/4)	$fc/2^3$ (0.17 μs)	$fc/2^5$ (0.66 μs)	$fc/2^7$ (2.67 μs)
			011 (fperiph/8)	$fc/2^4$ (0.33 μs)	$fc/2^6$ (1.33 μs)	$fc/2^8$ (5.33 μs)
			100 (fperiph/16)	$fc/2^5$ (0.66 μs)	$fc/2^7$ (2.67 μs)	$fc/2^9$ (10.67 μs)
			101 (fperiph/32)	$fc/2^6$ (1.33 μs)	$fc/2^8$ (5.33 μs)	$fc/2^{10}$ (21.33 μs)
		101 (fc/4)	000 (fperiph/1)	–	$fc/2^3$ (0.17 μs)	$fc/2^5$ (0.66 μs)
			001 (fperiph/2)	–	$fc/2^4$ (0.33 μs)	$fc/2^6$ (1.33 μs)
			010 (fperiph/4)	$fc/2^3$ (0.17 μs)	$fc/2^5$ (0.66 μs)	$fc/2^7$ (2.67 μs)
			011 (fperiph/8)	$fc/2^4$ (0.33 μs)	$fc/2^6$ (1.33 μs)	$fc/2^8$ (5.33 μs)
			100 (fperiph/16)	$fc/2^5$ (0.66 μs)	$fc/2^7$ (2.67 μs)	$fc/2^9$ (10.67 μs)
			101 (fperiph/32)	$fc/2^6$ (1.33 μs)	$fc/2^8$ (5.33 μs)	$fc/2^{10}$ (21.33 μs)
		110 (fc/8)	000 (fperiph/1)	–	–	$fc/2^5$ (0.66 μs)
			001 (fperiph/2)	–	$fc/2^4$ (0.33 μs)	$fc/2^6$ (1.33 μs)
			010 (fperiph/4)	–	$fc/2^5$ (0.66 μs)	$fc/2^7$ (2.67 μs)
			011 (fperiph/8)	$fc/2^4$ (0.33 μs)	$fc/2^6$ (1.33 μs)	$fc/2^8$ (5.33 μs)
			100 (fperiph/16)	$fc/2^5$ (0.66 μs)	$fc/2^7$ (2.67 μs)	$fc/2^9$ (10.67 μs)
			101 (fperiph/32)	$fc/2^6$ (1.33 μs)	$fc/2^8$ (5.33 μs)	$fc/2^{10}$ (21.33 μs)
1	*	*	*	fs/2 (61μs)	$fs/2^3$ (244 μs)	$fc/2^5$ (977 μs)

注 1: 须选择预分频标志输出时钟 φTn，以满足 φTn < fsys (故 φTn 慢于 fsys)。

注 2: 当计时器在操作时，勿改变时钟齿轮。

注 3: “.“ 表示设置被禁止。 “*“ 表示忽略。

表 11-4 预分频标志输出时钟分辨率 (fc = 32MHz, fs = 32.768kHz)

选择 φT0 CGSYSCR <FPSEL1>	选择外设时钟 CGSYSCR <FPSEL0>	选择齿轮时钟 CGSYSCR <GEAR[2:0]>	选择预分频标志时钟 CGSYSCR <PRCK[2:0]>	预定标志输出时钟功能		
				φT1	φT4	φT16
0	0 (fgear)	000 (fc)	000 (fperiph/1)	$fc/2^1$ (0.0625 μs)	$fc/2^3$ (0.25 μs)	$fc/2^5$ (1.0 μs)
			001 (fperiph/2)	$fc/2^2$ (0.125 μs)	$fc/2^4$ (0.5 μs)	$fc/2^6$ (2.0 μs)
			010 (fperiph/4)	$fc/2^3$ (0.25 μs)	$fc/2^5$ (1.0 μs)	$fc/2^7$ (4.0 μs)
			011 (fperiph/8)	$fc/2^4$ (0.5 μs)	$fc/2^6$ (2.0 μs)	$fc/2^8$ (8.0 μs)
			100 (fperiph/16)	$fc/2^5$ (1.0 μs)	$fc/2^7$ (4.0 μs)	$fc/2^9$ (16.0 μs)
			101 (fperiph/32)	$fc/2^6$ (2.0 μs)	$fc/2^8$ (8.0 μs)	$fc/2^{10}$ (32.0 μs)
		100 (fc/2)	000 (fperiph/1)	$fc/2^2$ (0.125 μs)	$fc/2^4$ (0.5 μs)	$fc/2^6$ (2.0 μs)
			001 (fperiph/2)	$fc/2^3$ (0.25 μs)	$fc/2^5$ (1.0 μs)	$fc/2^7$ (4.0 μs)
			010 (fperiph/4)	$fc/2^4$ (0.5 μs)	$fc/2^6$ (2.0 μs)	$fc/2^8$ (8.0 μs)
			011 (fperiph/8)	$fc/2^5$ (1.0 μs)	$fc/2^7$ (4.0 μs)	$fc/2^9$ (16.0 μs)
			100 (fperiph/16)	$fc/2^6$ (2.0 μs)	$fc/2^8$ (8.0 μs)	$fc/2^{10}$ (32.0 μs)
			101 (fperiph/32)	$fc/2^7$ (4.0 μs)	$fc/2^9$ (16.0 μs)	$fc/2^{11}$ (64.0 μs)
		101 (fc/4)	000 (fperiph/1)	$fc/2^3$ (0.25 μs)	$fc/2^5$ (1.0 μs)	$fc/2^7$ (4.0 μs)
			001 (fperiph/2)	$fc/2^4$ (0.5 μs)	$fc/2^6$ (2.0 μs)	$fc/2^8$ (8.0 μs)
			010 (fperiph/4)	$fc/2^5$ (1.0 μs)	$fc/2^7$ (4.0 μs)	$fc/2^9$ (16.0 μs)
			011 (fperiph/8)	$fc/2^6$ (2.0 μs)	$fc/2^8$ (8.0 μs)	$fc/2^{10}$ (32.0 μs)
			100 (fperiph/16)	$fc/2^7$ (4.0 μs)	$fc/2^9$ (16.0 μs)	$fc/2^{11}$ (64.0 μs)
			101 (fperiph/32)	$fc/2^8$ (8.0 μs)	$fc/2^{10}$ (32.0 μs)	$fc/2^{12}$ (128.0 μs)
		110 (fc/8)	000 (fperiph/1)	$fc/2^4$ (0.5 μs)	$fc/2^6$ (2.0 μs)	$fc/2^8$ (8.0 μs)
			001 (fperiph/2)	$fc/2^5$ (1.0 μs)	$fc/2^7$ (4.0 μs)	$fc/2^9$ (16.0 μs)
			010 (fperiph/4)	$fc/2^6$ (2.0 μs)	$fc/2^8$ (8.0 μs)	$fc/2^{10}$ (32.0 μs)
			011 (fperiph/8)	$fc/2^7$ (4.0 μs)	$fc/2^9$ (16.0 μs)	$fc/2^{11}$ (64.0 μs)
			100 (fperiph/16)	$fc/2^8$ (8.0 μs)	$fc/2^{10}$ (32.0 μs)	$fc/2^{12}$ (128.0 μs)
			101 (fperiph/32)	$fc/2^9$ (16.0 μs)	$fc/2^{11}$ (64.0 μs)	$fc/2^{13}$ (256.0 μs)

表 11-4 预分频输出时钟分辨率 (fc = 32MHz, fs = 32.768kHz)

选择 φT0 CGSYSCR <FPSEL1>	选择外设时钟 CGSYSCR <FPSEL0>	选择齿轮时钟 CGSYSCR <GEAR[2:0]>	选择预分频时钟 CGSYSCR <PRCK[2:0]>	预定标志输出时钟功能		
				φT1	φT4	φT16
0	1 (fc)	000 (fc)	000 (fperiph/1)	$fc/2^1$ (0.0625 μs)	$fc/2^3$ (0.25 μs)	$fc/2^5$ (1.0 μs)
			001 (fperiph/2)	$fc/2^2$ (0.125 μs)	$fc/2^4$ (0.5 μs)	$fc/2^6$ (2.0 μs)
			010 (fperiph/4)	$fc/2^3$ (0.25 μs)	$fc/2^5$ (1.0 μs)	$fc/2^7$ (4.0 μs)
			011 (fperiph/8)	$fc/2^4$ (0.5 μs)	$fc/2^6$ (2.0 μs)	$fc/2^8$ (8.0 μs)
			100 (fperiph/16)	$fc/2^5$ (1.0 μs)	$fc/2^7$ (4.0 μs)	$fc/2^9$ (16.0 μs)
			101 (fperiph/32)	$fc/2^6$ (2.0 μs)	$fc/2^8$ (8.0 μs)	$fc/2^{10}$ (32.0 μs)
		100 (fc/2)	000 (fperiph/1)	–	$fc/2^3$ (0.25 μs)	$fc/2^5$ (1.0 μs)
			001 (fperiph/2)	$fc/2^2$ (0.125 μs)	$fc/2^4$ (0.5 μs)	$fc/2^6$ (2.0 μs)
			010 (fperiph/4)	$fc/2^3$ (0.25 μs)	$fc/2^5$ (1.0 μs)	$fc/2^7$ (4.0 μs)
			011 (fperiph/8)	$fc/2^4$ (0.5 μs)	$fc/2^6$ (2.0 μs)	$fc/2^8$ (8.0 μs)
			100 (fperiph/16)	$fc/2^5$ (1.0 μs)	$fc/2^7$ (4.0 μs)	$fc/2^9$ (16.0 μs)
			101 (fperiph/32)	$fc/2^6$ (2.0 μs)	$fc/2^8$ (8.0 μs)	$fc/2^{10}$ (32.0 μs)
		101 (fc/4)	000 (fperiph/1)	–	$fc/2^3$ (0.25 μs)	$fc/2^5$ (1.0 μs)
			001 (fperiph/2)	–	$fc/2^4$ (0.5 μs)	$fc/2^6$ (2.0 μs)
			010 (fperiph/4)	$fc/2^3$ (0.25 μs)	$fc/2^5$ (1.0 μs)	$fc/2^7$ (4.0 μs)
			011 (fperiph/8)	$fc/2^4$ (0.5 μs)	$fc/2^6$ (2.0 μs)	$fc/2^8$ (8.0 μs)
			100 (fperiph/16)	$fc/2^5$ (1.0 μs)	$fc/2^7$ (4.0 μs)	$fc/2^9$ (16.0 μs)
			101 (fperiph/32)	$fc/2^6$ (2.0 μs)	$fc/2^8$ (8.0 μs)	$fc/2^{10}$ (32.0 μs)
		110 (fc/8)	000 (fperiph/1)	–	–	$fc/2^5$ (1.0 μs)
			001 (fperiph/2)	–	$fc/2^4$ (0.5 μs)	$fc/2^6$ (2.0 μs)
			010 (fperiph/4)	–	$fc/2^5$ (1.0 μs)	$fc/2^7$ (4.0 μs)
			011 (fperiph/8)	$fc/2^4$ (0.5 μs)	$fc/2^6$ (2.0 μs)	$fc/2^8$ (8.0 μs)
			100 (fperiph/16)	$fc/2^5$ (1.0 μs)	$fc/2^7$ (4.0 μs)	$fc/2^9$ (16.0 μs)
			101 (fperiph/32)	$fc/2^6$ (2.0 μs)	$fc/2^8$ (8.0 μs)	$fc/2^{10}$ (32.0 μs)
1	*	*	*	fs/2 (61μs)	$fs/2^3$ (244 μs)	$fc/2^5$ (977 μs)

注 1: 须选择预分频标志输出时钟 φTn，以满足 φTn < fsys (故 φTn 慢于 fsys)。

注 2: 当计时器在操作时，勿改变时钟齿轮。

注 3: "." 表示设置被禁止。"*" 表示忽略。

11.5.2 上升计数器 (UC)

UC 是一个 16-位二进制计数器。

- 源时钟

  UC 源时钟，由 TBxMOD<TBCLK[1:0]>指定，可从三种类型-φT1， ϕT4 和ϕT16-中选定，或从 TBxIN0 引脚的外部时钟引脚选定。

- 计数器开始/停止

  计时器操作由 TBxRUN<TBRUN>指定。如果<TBRUN> = “1“，则 UC 开始计数，如果<TBRUN> = “0“，则 UC 停止计数并清除计数值。

- 清除 UC 计时

  1. 当检测到一个匹配时。

     通过设置 TBxMOD<TBCLE> = “1“，当比较仪探测到计数值和 TBxRG1 的设置值之间的匹配时，UC 被清除。如果 TBxMOD<TBCLE> = “0“，则 UC 将作为不同步计数器来操作。

  2. 当 UC 停止时

     如果 TBxRUN<TBRUN> = “0“ ，则 UC 停止计数并清除计数值。

- UC 溢出

  如果生成 UC 溢出，则 INTTBx 溢出中断生成。

11.5.3 计时器寄存器 (TBxRG0，TBxRG1)

TBxRG0 和 TBxRG1 寄存器用于设置上升计数器数值的比较值。这两个寄存器的通道相互联通。当比较仪检测出此寄存器一设置值和 UC 上升计数器一设置值之间的一个匹配时，它会输出匹配检测信号。

TBxRG0 和 TBxRG1 包含双缓冲配置，与寄存器缓冲配对。双缓冲在初始阶段禁用。

通过 TBxCR<TBWBF>位控制双缓冲禁用和启用。当<TBWBF> = “0“时，双缓冲禁用。当<TBWBF> = “1“ 时，启用。当双缓冲启用时，如果 UC 与 TBxRG1 存在匹配，则数据将从寄存器缓冲传输到计时器寄存器 (TBxRG0/1)。当计数器在双缓冲启用状态下停止，则双缓冲将以单缓冲的模式操作，并立即将数据写入 TBxRG0 和 TBxRG1 中。

11.5.4 捕获

为一个控制自 UC 上升计数器到 TBxCP1 捕获寄存器的锁存值电路。对锁存数据的计时由 TBxMOD<TBCPM[1:0]>指定。

软件也可用于将数值从 UC 上升计数器导入捕获寄存器中；尤其是每次当 “0“ 被写入 TBxMOD<TBCP>时，UC 数值被计入 TBxCP0 捕获寄存器。

11.5.5 捕获寄存器 (TBxCP0，TBxCP1)

此寄存器捕获一上升计数器 (UC)数值。

11.5.6 上升计数器捕获寄存器 (TBxUC)

除上面所述捕获功能外，UC 的当前计数值可通过读取 TBxUC 寄存器来捕获。

11.5.7 比较器 (CP0，CP1)

此寄存器对上升计数器 (UC)和计时器寄存器 (TBxRG0 和 TBxRG1)的设置值进行比较，以检测是否存在匹配。当检测到匹配时，生成 INTTBx。

11.5.8 计时器触发器 (TBxFF0)

比较仪输出的匹配信号和捕获寄存器输出的锁存信号会使计时触发 (TBxFF0)逆转。逆转的启用和禁用可通过设置 TBxFFCR<TBC1T1，TBC0T1，TBE1T1，TBE0T1>来实现。

复位后，TBxFF0 的值变为未定义。通过向 TBxFFCR<TBFF0C[1:0]>写入 “00“来实现触发逆转。可通过写入 “01“来设置 “1“，通过写入 “10“来设置 “0“ 。

TBxFF0 的值可以输出到计时输出引脚 (TBxOUT)。执行计时输出时，相应的端口设置要提前进行编程。

11.5.9 捕获中断 (INTCAPx0，INTCAPx1)

当锁存值从 UC 上升计数器传输到 TBxCP0 和 TBxCP1 捕获寄存器时，生成中断 INTCAPx0 和 INTCAPx1。中断计时由 CPU 指定。

11.6 各模式操作说明

11.6.1 16-位间歇计时器模式

当产生持续期间中断时，向计时器寄存器 (TBxRG1)设置间歇时间来产生相应的 INTTBx 中断。

		7	6	5	4	3	2	1	0	
TBxEN	←	1	X	X	X	X	X	X	X	启用 TMRBx 操作。
TBxRUN	←	X	X	X	X	X	0	X	0	停止计数操作。
中断设置-启用寄存器	←	*	*	*	*	*	*	*	*	通过设置相应位至 "1"使 INTTBx 中断。
TBxFFCR	←	X	X	0	0	0	0	1	1	禁用 TBxFF0 逆转触发。
TBxMOD	←	X	0	1	0	0	1	*	*	预分频标志输出时钟变更为输入时钟。指定捕获功能来禁用。
							(** = 01, 10, 11)			
TBxRG1	←	*	*	*	*	*	*	*	*	指定一时间间歇。(16 位)。
	←	*	*	*	*	*	*	*	*	
TBxRUN	←	*	*	*	*	*	1	X	1	启动 TMRBx.

注:X：忽略

-：无改变

11.6.2 16-位事件计数模式

将一输入时钟作为一外部时钟 (TBxIN0 引脚输入)使其成为事件计数器。

上升计数器将计数 TBxIN0 引脚输入的上升沿。它能读取利用软件和读取捕获值而获得的计数值 。

		7	6	5	4	3	2	1	0	
TBxEN	←	1	X	X	X	X	X	X	X	启用 TMRBx 操作。
TBxRUN	←	X	X	X	X	X	0	X	0	停止计数 TMRBx。
设置 PORT 寄存器										将相应端口分配至 TBxIN0。
TBxFFCR	←	X	X	0	0	0	0	1	1	禁用 TBxFF0 逆转触发
TBxMOD	←	X	0	1	0	0	0	0	0	将 TBxIN0 转变为计入时钟。
TBxRUN	←	*	*	*	*	*	1	X	1	启动 TMRBx.
TBxMOD	←	X	0	0	0	0	0	0	0	软件捕获完成。

注:X：忽略

-：无改变

11.6.3 16-位PPG (可编程脉冲生成)输出模式

可输出带频率和占空方波 (可编程方波)。输出脉冲可为低激活或高激活。

当上升计数器 (UC)的设置值与计时器寄存器 (TBxRG0 和 TBxRG1)的设置值有匹配时，通过触发‘计时器触发器’ (TBxFF)进行逆转，则可编程方波可从 TBxOUT 引脚处输出。注意，TBxRG0 和 TBxRG1 的设置值应满足以下要求:

TBxRG0 设置值< TBxRG1 设置值

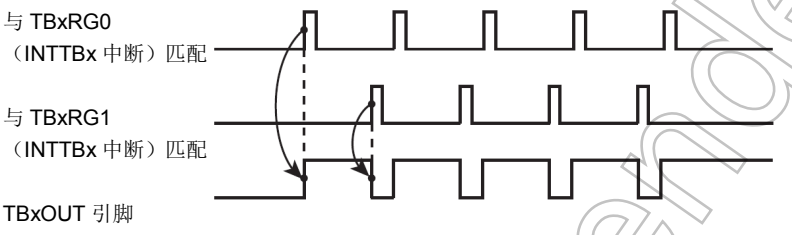


图 11-3 可编程脉冲产生 (PPG)的输出举例

在此模式下，当上升计数器设置值和 TBxRG1 设置值匹配时，通过启用 TBxRG0 的双缓冲，寄存器缓冲 0 的数值被转入 TBxRG0。此操作对处理小任务有用。

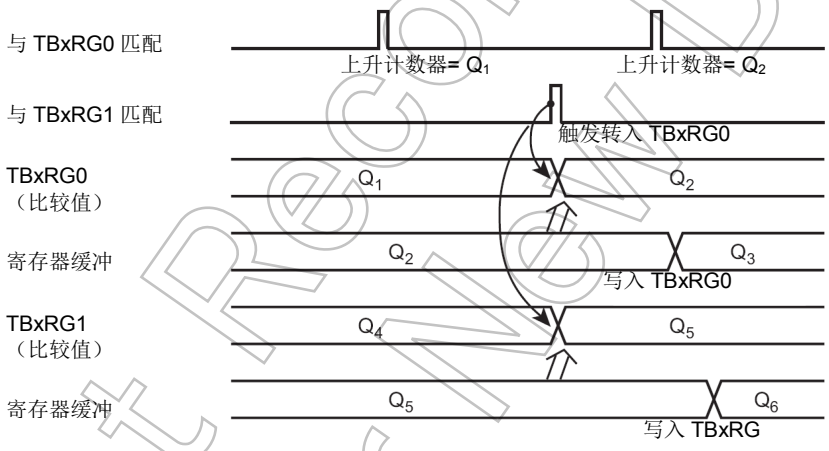


图 11-4 寄存器缓冲操作

此模式的方块图显示如下。

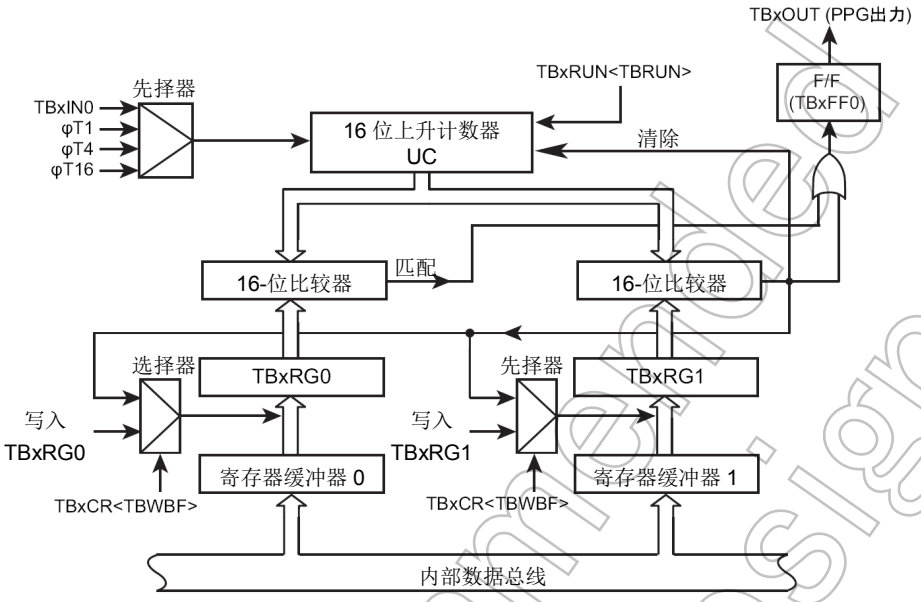


图 11-5 16-位 PPG 模式方块图

16-位 PPG 中的每个寄存器的编程应遵循以下所列内容。

		7	6	5	4	3	2	1	0	
TBxEN	←	1	X	X	X	X	X	X	X	启用 TMRBx 操作。
TBxRUN	←	X	X	X	X	0	X	0	停止计数操作。	
TBxCR	←	0	0	−	X	−	X	0	0	禁用双缓冲。
TBxRG0	←	*	*	*	*	*	*	*	*	指定一个占空比。(16 位))
	←	*	*	*	*	*	*	*	*	
TBxRG1	←	*	*	*	*	*	*	*	*	指定一个循环。(16 位))
	←	*	*	*	*	*	*	*	*	
TBxCR	←	1	0	−	X	−	X	0	0	启用 TBxRG0 双缓冲。 (当 INTTBx 中断产生之时，变更占空/循环)
TBxFFCR	←	X	X	0	0	1	1	1	0	当有对应 TBxRG0 或 TBxRG1 的匹配时，做相应指定以触发 TBxFF0 进行逆转，并将 TBxFF0 的初始值设置为"0"。
TBxMOD	←	X	0	1	0	0	1	*	*	指定预分频器输出时钟为输出时钟，且禁用捕获功能。 (** = 01, 10, 11)
设置 PORT 寄存器。										对 TBxOUT 分配相应的端口。
TBxRUN	←	*	*	*	*	*	1	X	1	启动 TMRBx.

注:X；忽略 −；无改变

11.6.4 计时器同步模式

此模式启用计时器同步启动。

当此模式与 PPG 输出一起使用时，输出可驱动电机。

TMRB 由对 4-通道 TMRB 组成。当一个通道启动时，剩下的 3 个通道可同步启动。在 TMPM364F10FG 中，可使用以下组合。

启动触发通道 (主通道)	同步操作通道 (从通道)
TMRB0	TMRB1, TMRB2, TMRB3
TMRB4	TMRB5, TMRB6, TMRB7
TMRB8	TMRB9, TMRBA, TMRBB
TMRBC	TMRBD, TMRBE, TMRBF

对计时器器同步模式的使用由 TBxCR<TBSYNC>位指定。

<TBSYNC> = “0“: 计时器独立操作。

<TBSYNC> = “1“: 计时器同步操作。

将 “0“ 设置到主通道中<TBSYNC> 位。

当<TBSYNC>= “1“被设置于从通道时，启动计时与主通道启动计时同步。 不要求在从通道中设置 TBxRUN<TBPRUN, TBRUN>位的启动计时。

11.7 捕获功能的应用

捕获功能用于开发多种应用，包括以下所述:

1. 由外部脉冲触发的单脉冲输出。
2. 频率测量
3. 脉冲宽度测量
4. 时差测量

11.7.1 由外部脉冲触发的单脉冲输出。

由一外部脉冲触发的单脉冲的执行如下:

通过将其置于利用以预分频器输出时钟在自由运行状态下进行计数做成 16-位计数器。外部脉冲通过 TBxIN0 引脚输入。通过利用捕获功能在外部脉冲的上升沿生成一个触发器，并将上升计数器数值计入捕获寄存器 (TBxCP0)。

CPU 必须进行相应编程使每个中断 INTCAPx0 在外部触发脉冲的上升沿中生成。此中断用于设置 TBxRG0 值的总和与延迟时间 (d)，(c+d)上的计时器寄存器 TBxCP0 和设置在 TBxCP0 数值总和上的计时器寄存器，以及单脉冲的脉冲宽度 (p)，(c+d+p)。TBxRG1 变更必须在下一个匹配之前完成。

另外，计时器触发控制寄存器 (TBxFFCR<TBE1T1, TBE0T1>)必须设置为 “11“。当 UC 匹配 TBxRG0 和 TBxRG1 时，该操作触发计时器触发 (TBxFF0)使其逆转。当单脉冲输出后，该触发器在通过 INTIBx 禁用。

此文件中使用的符号 (c)， (d)和 (p)对应于 “图 11-6 单脉冲输出 (带延迟) “中的符号 c，d 和 p。

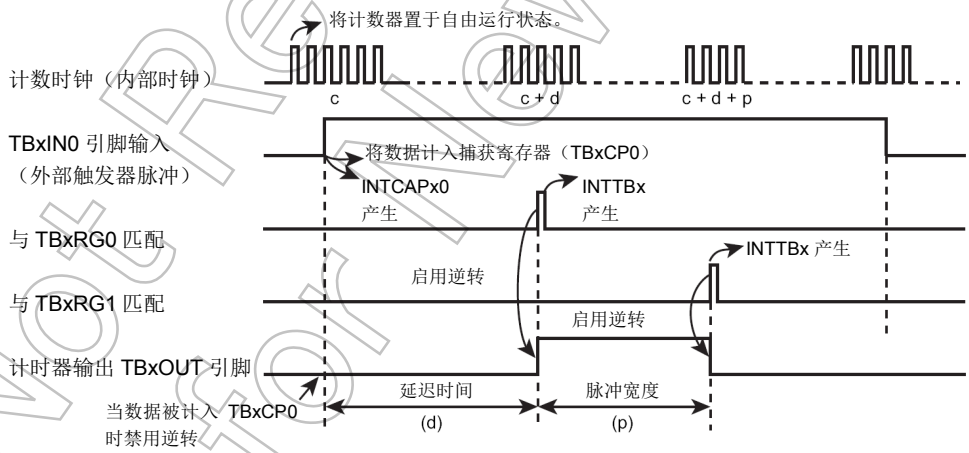


图 11-6 单脉冲输出 (带延迟)

以下显示，在上升沿中触发 TBxIN0 输入产生 3ms 宽度脉冲后，输出 2 ms 宽度的单脉冲情况下的设置。 (选择 φT1 来计数。)

		7	6	5	4	3	2	1	0	
[主处理] 通过 TBxIN0 设置进行捕获。										
设置 PORT 寄存器。										将相应端口分配至 TBxIN0。
TBxEN	←	1	X	X	X	X	X	X	X	启用 TMRBx 操作。
TBxRUN	←	X	X	X	X	X	0	X	0	停止计数操作。
TBxMOD	←	X	0	1	0	1	0	0	1	将源时钟变为为 φT1。TBxIN0 将计数值计入 TBxCP0 上升沿。
TBxFFCR	←	X	X	0	0	0	0	1	0	清除 TBxFF0 逆转触器发并禁用。
设置 PORT 寄存器。										对 TBxOUT 分配相应的端口。
中断设置启用寄存器	←	*	*	*	*	*	*	*	*	设置至 "1"，由 INTCAPx0 中断对应位指定来生成中断。
TBxRUN	←	*	*	*	*	*	1	X	1	启动 TMRBx 模块。
[处理 INTCAPx0 中断服务路径] 脉冲输出设置。										
TBxRG0	←	*	*	*	*	*	*	*	*	设置计数值。（TBxCAP0 + 3ms/φT1）
	←	*	*	*	*	*	*	*	*	
TBxRG1	←	*	*	*	*	*	*	*	*	设置计数值。（TBxCAP0 +（3+2）ms/φT1）
	←	*	*	*	*	*	*	*	*	
TBxFFCR	←	X	X	−	−	1	1	−	−	当 UC 与 TBxRG0 和 TBxRG1 相一致时，将 TBxFF0 逆转。
TBxIM	←	X	X	X	X	X	1	0	1	将 TBxRG1 对应中断以外的中断屏蔽。
中断设置-启用寄存器	←	*	*	*	*	*	*	*	*	设置至 "1"，由，INTTBx 中断对应位指定来生成中断。
[INTTBx 中断服务路径的处理] 输出禁用										
TBxFFCR	←	X	X	−	−	0	0	−	−	清除 TBxFF0 逆转触发器设置。
中断启用清除寄存器	←	*	*	*	*	*	*	*	*	设置至 "1"，由 IINTTBx 中断对应位指定来禁止中断。

注:X；忽略

−；无改变

当延迟无要求时，在数据计入 TBxCP0 时，且 TBxRG1 设置到 TBxCP0 的数值 (c)的总和，加上单脉冲宽度 (p)， (c+p)，通过产生 INTCAPx0 中断，将 TBxFF0 逆转，在下一匹配前必须完成 TBxRG1 变更。

当 UD 与 TBxRG1 匹配时，使 TBxFF0 逆转，并通过产生 INTTBx 中断禁用。

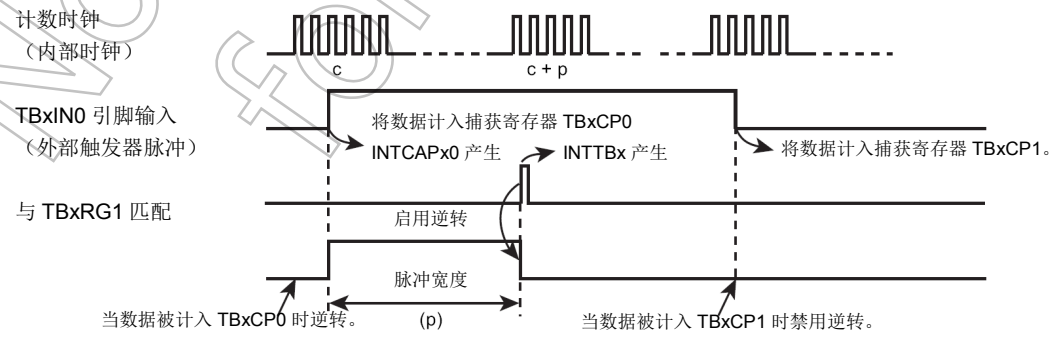


图 11-7 由外部脉冲触发的单脉冲输出触发 (无延迟)

11.7.2 频率测量

外部时钟的频率通过使用捕获功能来测量。

测量频率时，另一个 16-位计时器应与 16-位事件计数器模式联合使用。以 TMRB5 和 TMRB0 作为例子进行说明。16 位计时器 TMRB0 的 TB0OUT 用于指定测量时间。

TMRB5 计数时钟选择 TB5IN0 输入并通过使用外部时钟输入来执行计数操作。如果 TB5MOD<TBCPM[1:0]>被设置为 “11“，则 TMRB5 计数时钟将于 TB0OUT 上升沿把计数器数值计入 TB5CP0，并于 TB0OUT 下降沿将计数器数值计入 TB5CP1。

此设置使 16-位上升计数器 UC 的一个计数值在 16-位计时器 (TMRB0)的计时触发输出的上升沿上被计入捕获寄存器 (TB5CP0)，同时一个 UC 计数器数值在 16-位计时器 (TMRB0)的 TB0OUT 的下降沿上被计入捕获寄存器 (TB5CP1)。

通过产生 16-位计时器中断，在测得的 TB5CP0 和 TB5CP1 之间的差值中获得一频率值。

例如，若 TB5CP0 和 TB5CP1 之间的差值为 100，TB0OUT 的阶宽设置值为 0.5 s，则频率为 200Hz (100÷0.5 s = 200 Hz)。

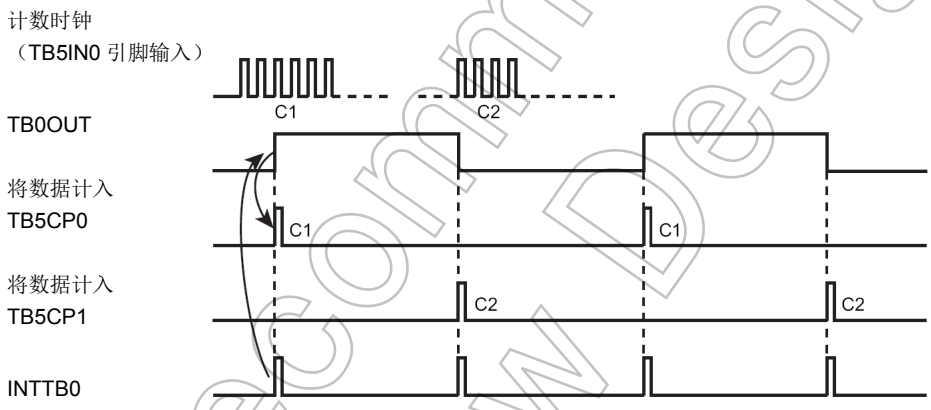


图 11-8 频率测量

11.7.3 脉冲宽度测量

通过使用捕获功能，可测量外部脉冲的 “高“电平宽度。尤其是，当通过预分频器输出时钟将其置于一自由运行状态下时，外部脉冲通过 TBxIN0 引脚输入，成为上升计数器 (UC)的计数。利用捕获功能在外部脉冲各上升和下降沿中产生以触发器，并将上升计数器数值计入捕获寄存器 (TBxCP0，TBxCP1)，从而生成以触发器。必须对 CPU 进行编程，使通过 TBxIN0 引脚的外部脉冲的下降沿处生成 INTCAPx1。

“高“电平脉冲宽度可通过一间歇时钟的时钟周期乘以在 TBxCP0 和 TBxCP1 之间产生的差值而计算出来。

例如，如果 TBxCP0 和 TBxCP1 之间的差值为 100，而预分频器的输出时钟周期为 0.5 μs，则脉冲宽度为 100×0.5 μs = 50 μs。

必须小心测量脉冲的宽度，超过所使用源时钟的 UC 最大计数时间。这种脉冲宽度的测量必须依靠软件。

外部脉冲的 “低“电平宽度也可测量。在这种情况下，第一次产生的 C2 和第二次产生的 C1 之间的差值，由如 “图 11-9 脉冲宽度测量“ 所述的 INTCAPx0 中断处理的第二阶段进行初始化处理获得，此差值乘以预分频器输出时钟周期得到 “低“阶宽度。

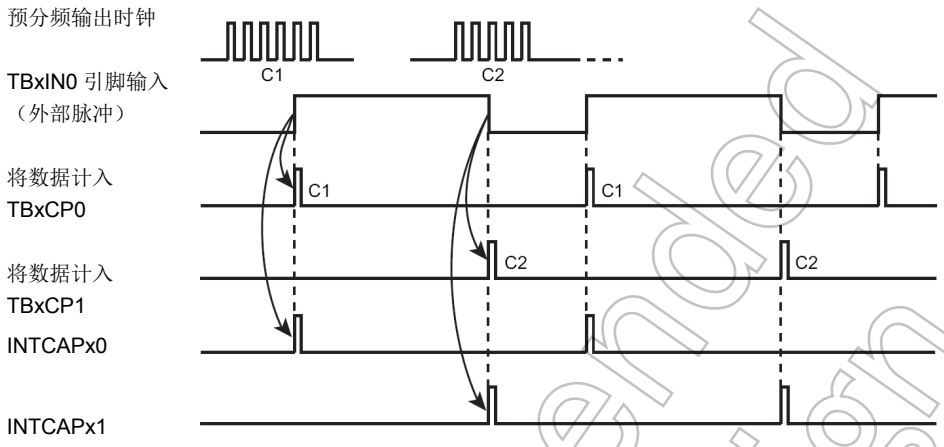


图 11-9 脉冲宽度测量

11.7.4 时差测量

两个事件发生之间的时差可通过捕获功能测量。利用预分频输出时钟将上升计数器置于自由操作状态下，成为上升计数器 (UC)进行计数。

上升计数器 UC 的数值在 TBxIN0 引脚输入脉冲的上升沿被计入捕获寄存器 (TBxCP0)。此时须对 CPU 进行编程以产生 INTCAPx0 中断。

上升计数器 UC 的数值在 TBxIN1 引脚输入脉冲的上升沿被计入捕获寄存器 (TBxCP1)。此时须对 CPU 进行编程以产生 INTCAPx1 中断。

时差可通过一间歇时钟的时钟周期乘以在 TBxCP1 和 TBxCP0 之间产生的差值计算出来。

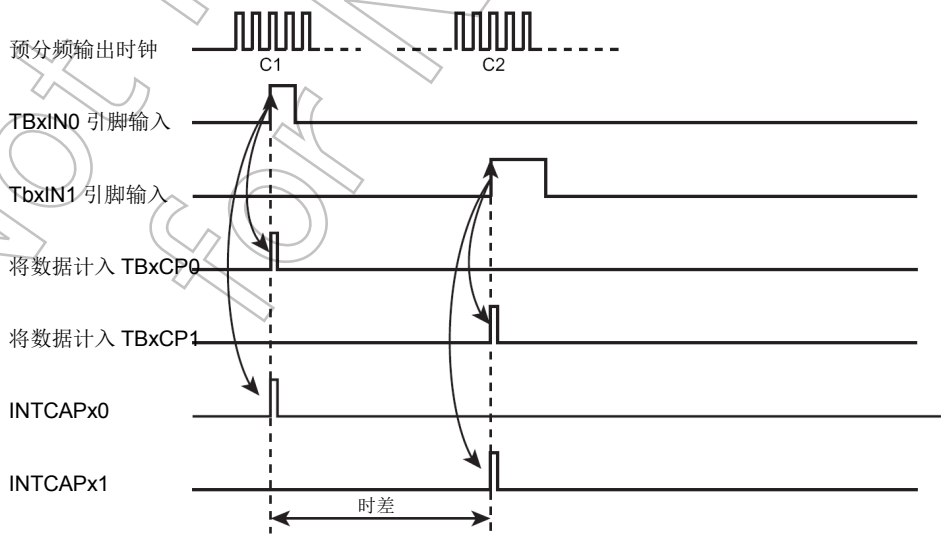


图 11-10 时差测量

12. 串行通道 (SIO/UART)

12.1 概述

此仪器用于串行通道时有两个模式，一个是同步通信模式 (I/O 接口模式)，另一个是异步通信模式 (UART 模式)。

其特征描述如下:

- 传输时钟
  - 通过将外围时钟 (φT0)频率划分为1/2，1/8，1/32，1/128来形成传输时钟。
  - 预分频器输出时钟频率除以1～16中的每个数字。
  - 预分频器输出频率仅可除以1，N+m/16 (N=2-15, m=1-15)，及 16中的每个数。(仅UART 模式) 。
  - 系统时钟可用。 (仅UART 模式)
- 双缓冲 / FIFO

  双缓冲功能和FIFO缓冲 (传输和接收总计)可用到4个字节。
- I/O 接口模式
  - 传输模式:半双向 (传送 / 接收)和全双向通信
  - 时钟:输出 (固定上升沿)/计入 (选择上升/下降沿)
  - 可在执行连续传输的范围内设置时间间隔。
- UART 模式
  - 数据长度:7，8，9位
  - 增加校检位 (针对9-位数据长度)
  - 使用唤醒功能的串行链路
  - 带 $\overline{\text{CTS}}$ 引脚的握手功能

在以下说明中，"x"表示通道号。

12.2 SIO模块规格的差异

TMPM364F10FG 有 12 个通道。

每个通道功能均是独立的。每个通道的引脚及中断表示如下:

表12-1 SIO模块各个通道的差异

	引脚名称			中断		串行时钟的计时器	DMA
	TXD	RXD	$\overline{\text{CTS}}$ /SCLK	接收 中断	传输 中断		
通道0	PE4	PE5	PE6	INTRX0	INTTX0	TB5OUT	支持
通道1	PL4	PL5	PL6	INTRX1	INTTX1	TB5OUT	支持
通道2	PM1	PM2	PM0	INTRX2	INTTX2	TB5OUT	支持
通道3	PM5	PM6	PM4	INTRX3	INTTX3	TB5OUT	支持
通道4	PN0	PN1	PN2	INTRX4	INTTX4	TB6OUT	支持
通道5	PN4	PN5	PN6	INTRX5	INTTX5	TB6OUT	支持
通道6	PO0	PO1	PO2	INTRX6	INTTX6	TB6OUT	支持
通道7	PO4	PO5	PO6	INTRX7	INTTX7	TB6OUT	支持
通道8	PC0	PC1	PC2	INTRX8	INTTX8	TB9OUT	支持
通道9	PC4	PC5	PC6	INTRX9	INTTX9	TB9OUT	支持
通道10	PD0	PD1	PD2	INTRX10	INTTX10	TB9OUT	支持
通道11	PD4	PD5	PD6	INTRX11	INTTX11	TB9OUT	支持

12.3 配置

图 12-1 显示了SIO方块图。

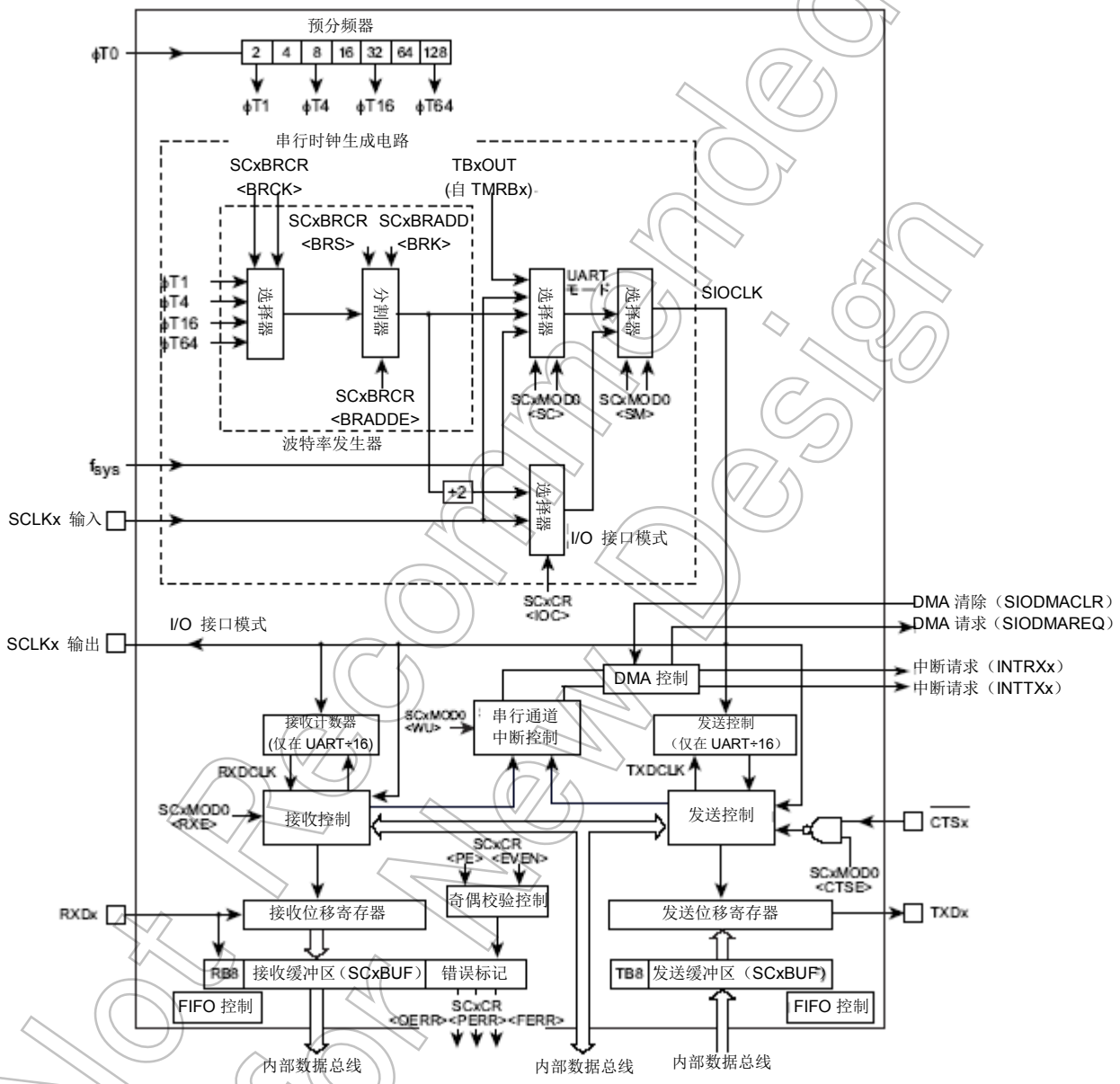


图 12-1 超级 SIO 方块图

12.4 寄存器说明

12.4.1 各通道寄存器列表

下表中显示了各寄存器及其地址。

通道 x	基址
通道 0	0x400E_1000
通道 1	0x400E _ 1100
通道 2	0x400E _ 1200
通道 3	0x400E _ 1300
通道 4	0x400E _ 1400
通道 5	0x400E _ 1500
通道 6	0x400E _ 1600
通道 7	0x400E _ 1700
通道 8	0x400E _ 1800
通道 9	0x400E _ 1900
通道 10	0x400E _ 1A00
通道 11	0x400E _ 1B00

寄存器名称 (x=0 ~ 11)		地址 (基+)
启用寄存器	SCxEN	0x0000
缓冲寄存器	SCxBUF	0x0004
控制寄存器	SCxCR	0x0008
模式控制寄存器	SCxMOD0	0x000C
波特率发生器控制寄存器	SCxBRCR	0x0010
波特率发生器控制寄存器 2	SCxBRADD	0x0014
模式控制寄存器 1	SCxMOD1	0x0018
模式控制寄存器 2	SCxMOD2	0x001C
RX FIFO 配置寄存器	SCxRFC	0x0020
TX FIFO 配置寄存器	SCxTFC	0x0024
RX FIFO 状态寄存器	SCxRST	0x0028
TX FIFO 状态寄存器	SCxTST	0x002C
FIFO 配置寄存器	SCxFCNF	0x0030

注 1: 切勿在发送或接收数据的过程中对任何控制寄存器进行修改。

注 2: 切勿在接收数据的过程中清除 SCxMOD0<RXE>。

注 3: 切勿在发送数据的过程中清除 SCxMOD1<TXE>。

12.4.2 SCxEN (启用寄存器)

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

位	比特符号	型号	功能
31-1	−	R	读作 “0“
0	SIOE	R/W	SIO 运行 0：禁用 1：操作 可用于指定 SIO 的运行状态。 如需使用 SIO，应设定 <SIOE> = “1“。 当运行处于禁用状态时，将不会为有时钟向 SIO 模块内的其他寄存器输入任何信号。这可以降低对功率的消耗。 在先运行 SIO 再将其禁用的情况下，系统会保留除 SCxTFC<TIL> 以外其他所有寄存器中的相关设置。

注 1: 如果设定了 SCxEN<SIOE>= “0“ (停止运行 SIO)，或在 SCxMOD1<I2SC>= “0“ (IDLE 模式下停止运行 SIO)的条件下将运行模式切换到 “IDLE“模式，则 SCxTFC 会重新进行初始化。

注 2: 在采用发送/接收 SIO 中断的条件下进行 DMA 传输时，首先应通过 SCxMOD2<SWRST[1:0]> 的设定生成软件复位，接着启用 DMAC (DMA 请求等待状态)，然后再启动 SIO 的发送/接收。

12.4.3 SCxBUF (缓冲寄存器)

SCxBUF 可作为写入操作的发送缓冲区或FIFO缓冲区，也可作为读取操作的接收缓冲区或FIFO缓冲区。

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2。	1	0
比特符号	TB/RB							
复位后	0	0	0	0	0	0	0	0

位	比特符号	型号	功能
31-8	−	R	读作 “0“。
7-0	TB[7:0] / RB [7:0]	R/W	[写入] TB： 发送缓冲区 / FIFO [读取] RB： 接收缓冲区 / FIFO

12.4.4 SCxCR (控制寄存器)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2。	1	0
比特符号	RB8	EVEN	PE	OERR	PERR	FERR	SCLKS	IOC
复位后	0	0	0	0	0	0	0	0

位	比特符号	型号	功能
31-8	−	R	读作 “0“。
7	RB8	R	接收第 8 位数据 (用于 UART) 9 位 “UART“模式下接收数据中的第 9 位
6	EVEN	R/W	奇偶校验 (用于 UART) 0：奇数 1：偶数 可用于选择偶数校验或奇数校验。 “0“ :偶数校验； “1“： 偶数同位 奇数校验位只能在 7-位和 8-位 “UART“模式下使用。
5	PE	R/W	添加奇偶校验 (用于 UART) 0：禁用 1：启用 启动用/禁用奇数校验功能的控制 奇数校验位只能在 7-位和 8-位 “UART“模式下使用。
4	OERR	R	溢出错误标记 (注) 0：正常工作 1：错误
3	PERR	R	奇数校验/欠载错误标记 (注) 0: 正常工作 1: 错误
2	FERR	R	组帧错误标记 (注) 0: 正常工作 1: 错误
1	SCLKS	R/W	选择输入时钟边沿 (用于 I/O 接口) 在时钟输出模式下设为 “0“。 0: SCLKx 出现下降沿时，将存储于发送缓冲区内的数据按一次发送一个数位的方式发送到 TXDx 引脚。 在 SCLKx 出现上升沿时，将从 RXDx 引脚输入的数据按一次接收一个数位的方式接收到缓冲区中。 在这一情况下，SCLKx 会从高电平开始工作。 1: SCLKx 出现上升沿时，将存储于发送缓冲区内的数据按一次发送一个数位的方式发送到 TXDx 引脚。 SCLKx 出现下降沿时，将从 RXDx 引脚输入的数据按一次接收一个数位的方式接收到缓冲区中。 在这一情况下，SCLKx 会从低电平开始工作。
0	IOC	R/W	选择时钟 (适用于 I/O 接口) 0: 波特率发生器 1: SCLK 引脚输入

12.4.5 SCxMOD0 (模式控制寄存器 0)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2。	1	0
比特符号	TB8	CTSE	RXE	WU	SM		SC	
复位后	0	0	0	0	0	0	0	0

位	比特符号	型号	功能
31-8	−	R	读作 “0“。
7	TB8	R/W	发送第 8 位数据 (用于 UART) 9 位“UART“模式下写入发送数据中的第 9 位
6	CTSE	R/W	握手功能的控制 (适用于 UART) 0: 禁用 CTS 1: 启用 CTS 控制握手功能 将数值设为 “1“即可启用使用 $\overline{\text{CTS}}$ 引脚的握手功能。
5	RXE	R/W	接收控制 (注 1) (注 2) 0: 禁用 1: 启用
4	WU	R/W	唤醒功能 (用于 UART) 0: 禁用 1: 启用 此功能仅适用于 9 位 UART 模式。在其他模式下，此功能则无意义。 如果其设定为启用状态，则在 9 位 UART 模式下，只有当 RB9 = “1“ 时才会产生中断。
3-2	SM[1:0]	R/W	可用于指定传输模式。 00: I/O 接口模式 01: 7-位长度的 UART“模式 10: 8-位长度的“UART“模式 11: 9-位长度的 UART 模式
1-0	SC[1:0]	R/W	串行传输时钟 (用于 UART) 00: 定时器 TB 9OUT 详见表 12-1。 01: 波特率发生器 10: 内部时钟 fsys 11: 外部时钟 (SCLK 输入) (对于 I/O 接口模式，串行传输时钟可以在控制寄存器 (SCxCR)中进行设置。

注 1:在规定各模式寄存器 (SCxMOD0，SCxMOD1，SCxMOD2)后再对 <RXE> 进行设定。

注 2:切勿在接收数据的过程中对 SCxMOD0<RXE> 进行清除。

12.4.6 SCxMOD1 (模式控制寄存器 1)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2。	1	0
比特符号	I2SC	FDPX		TXE	SINT			-
复位后	0	0	0	0	0	0	0	0

位	比特符号	型号	功能
31-8	−	R	读作 “0“。
7	I2SC	R/W	IDLE 0: 停止 1: 操作 可用于指定 IDLE 模式的运行。
6-5	FDPX[1:0]	R/W	传输模式设置 00: 禁止传输 01: 半双工 (接收) 10: 半双工 (发送) 11: 全双工 可用于对在 I/O 接口模式下的传输模式进行配置。此外，在启用了 FIFO 的情况下，还可用于对 FIFO 进行配置。 在 UART 模式下，其只能用于指定 FIFO 配置。
4.	TXE	R/W	发送控制 (注 1) (注 2) 0: 禁用 1: 启用 该位可用于启用传输功能，其适用于所有的传输模式。
3-1	SINT[2:0]	R/W	连续发送的间隔时间 (适用于 I/O 接口) 000:无 001: 1SCLK 010: 2SCLK 011: 4SCLK 100: 8SCLK 101: 16SCLK 110: 32SCLK 111: 64SCLK 在选择了 SCLK 引脚输出的情况下，此参数只适用于 I/O 接口模式。在其他模式下，此功能则无意义。 可用于指定在 I/O 接口模式下启用了双缓冲功能或 FIFO 的条件下的连续发送的时间间隔。
0	−	R/W	写入数值为 “0“。

注 1: 规定所有的模式后，启用 <TXE> 数位。

注 2: 切勿在发送数据的过程中停止发送操作 (通过设定 <TXE> = “0“)。

注 3: 在使用 SIO 的发送/接收中断进行 DMA 传输时，将无法使用全双式发送功能。

注 4: 如果设定了 SCxEN<SIOE>= “0“ (停止运行 SIO)，或在 SCxMOD1<I2SC>= “0“ (IDLE 模式下停止运行 SIO)的条件下将运行模式切换 ~ 了 IDE 模式，则 SCxTFC 会重新初始化。

12.4.7 SCxMOD2 (模式控制寄存器 2)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	TBEMP	RBFLL	TXRUN	SBLEN	DRCHG	WBUF	SWRST	
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-8	−	R	读作 “0“。
7	TBEMP	R	发送缓冲区空载标记。 0: 满 1: 空 如果已禁用了双缓冲功能，则这一标记的意义不大。 此标记用以显示双发送缓冲区处于空载状态。当双发送缓冲区内存储的数据移动 ~ 发送位移寄存器内，且使双缓冲区为空载状态时，此数位设置为 “1“。 如需重新向双缓冲区写入数据，则应将此数位设定为为 “0“。
6	RBFLL	R	接收缓冲区满载标记。 0: 空 1: 满 如果已禁用了双缓冲功能，则这一标记的意义不大。 此标记用以显示双发送缓冲区处于满载状态。 在已完成接收操作，且接收到的数据已从接收位移寄存器内移动到了双接受缓冲区中，则此数位的数值会变为 “1“；而在读取时，这一数位则变为 “0“。
5	TXRUN	R	正在发送标志 0: 停止 1: 操作 为数据传输正在进行中的状态标志。 <TXRUN>与<TBEMP>位表示以下状态。 <TXRUN> | <TBEMP> | 状态 1 | - | 发送正在进行中 0 | 1 | 发送完成 0 | 0 | 等待状态，数据在发送缓冲器中。
4	SBLEN	R/W	STOP 位 (用于UART) 0: 1-位 1: 2-位 规定了UART模式下传输停止位的长度。 在接收侧，仅使用单个位来决定，不考虑<SBLEN>设置。
3	DRCHG	R/W	设置传输方向 0:LSB优先 1:MSB优先 规定了I/O接口模式下数据传输的方向。 在UART模式中，首先将此位设置到 LSB。
2	WBUF	R/W	双缓冲器 0: 禁用 1: 启用 此参数启用或禁用传输/接收双缓冲器，以在I/O接口模式中传输 (在SCLK输出 / 输入两种模式下)与接收 (以SCLK输出模式)数据，并以UART模式传输数据。 在I/O接口模式 (SCLK输入)与UART模式下接收数据时，在 “0“或 “1“被设置到<WBUF>位的两种情况下双缓冲器都启用。

<table>
<tr><th>位</th><th>比特符号</th><th>类型</th><th>功能</th></tr>
<tr><td>1-0</td><td>SWRST[1:0]</td><td>R/W</td><td>软件复位
用 “10 “重写 “01 “生成软件复位。在执行该软件复位时，将以下位初始化，并传输/接收电路，传输电路与FIFO存储器变为初始状态 (见注1与注2)。
<table>
<tr><th>寄存器</th><th>位</th></tr>
<tr><td>SCxMOD0</td><td><RXE></td></tr>
<tr><td>SCxMOD1</td><td><TXE></td></tr>
<tr><td>SCxMOD2</td><td><TBEMP>, <RBFLL>, <TXRUN></td></tr>
<tr><td>SCxCR</td><td><OERR>, <PERR>, <FERR></td></tr>
</table></td></tr>
</table>

注 1: 当数据传输正在进行时，任何软件复位操作必须连续执行两次。

注 2: 软件复位指令识别结束与执行开始之间的时间内，软件复位需要 2 个时钟脉冲持续时间。

12.4.8 SCxBRCR (波特率发生器控制寄存器)，SCxBRADD (波特率发生器控制寄存器 2)

波特率发生器的分频比可在如下所示的寄存器中指定。

SCxBRCR

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	-	BRADDE	BRCK		BRS			
复位后	0	0	0	0	0	0	0	0

位	比特符号	型号	功能
31-8	-	R	读作 “0“。
7	-	R/W	写入数值为 “0“。
6	BRADDE	R/W	N + (16 − K)/16分频器功能 (用于UART) 0: 禁用 1: 启用 此分频功能仅用于UART模式。
5-4	BRCK[1:0]	R/W	在波特率发生器上选择输入时钟。 00: φT1 01: φT4 10: φT16 11: φT64
3-0	BRS[3:0]	R/W	分频比 “N“ 0000: 16 0001: 1 0010: 2 : 1111: 15

SCxBRADD

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

位	比特符号	型号	功能
31-4	-	R	读作 “0“ 。
3-0	BRK[3:0]	R/W	为 “N+ (16 - K)/16 “分频指定 K (用于UART) 0000: 禁止 0001: K = 1 0010: K = 2 : 1111: K = 15

表 12-2 列出了波特率发生器分频比的设置。

表 12-2 设置分频比

	<BRADDE> = “0“	<BRADDE>= “1“ (注1) (仅UART模式)
<BRS>	指定 “N“ (注2) (注3)	
<BRK>	不需要设置	指定 “K“ (注4)
分频比	除以N	$N + \frac{(16 - K)}{16}$ 分频

注 1: 使用 “N + (16 - K)/16“分频功能时，在将 K 值设置到 <BRK>后，务必将<BRADDE>设置到 “1“。“N + (16 - K)/16“分频功能只能在 UART 模式中使用。

注 2: 作为分频比，在 UART 模式中使用 “N + (16 - K)16“分频功能时，1 (“0001“)或 16 (“0000“)不适用于 N。

注 3: 只有当双缓冲器在 I/O 接口模式中使用时，才可指定波特率发生器的分频比 “1“。

注 4: 禁止指定 “K = 0“。

12.4.9 SCxFCNF (FIFO配置寄存器)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	-	-	-	RFST	TFIE	RFIE	RXTXCNT	CNFG
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-8	-	R	读作 “0“。
7-5	-	R/W	务必写入 “000“。
4	RFST	R/W	FIFO中所用字节 0: 最大值 1: 与 RX FIFO 的FILL电平相同 在启用RX FIFO时，选择所用的RX FIFO 字节数 (注1) 0:所配置的FIFO 最大字节数 (亦见<CNFG>)。 1: 与SCxRFC<RIL[1:0]>指定的接收中断生成的空满度相同
3	TFIE	R/W	TX FIFO 的TX中断 0: 禁用 1: 启用 启用TX FIFO时，通过此参数启用或禁用传输中断。
2	RFIE	R/W	RX FIFO的RX中断 0:禁用 1:启用 启用RX FIFO时，通过此参数启用或禁用接收中断。
1	RXTXCNT	R/W	RXE / TXE的自动禁用 0: 无 1: 自动禁用 控制传输与接收的自动禁用。 设置 “1“启用操作如下 半双工RX：当接收移位寄存器时，接收缓冲器与RX FIFO存储器写满，SCxMOD0<RXE>自动设置到 “0“，防止进一步的接收。 半双工TX：当TX FIFO，传输缓冲器及传输移位寄存器未占用时，SCxMOD1<TXE>被自动设置到 “0“，防止进一步的传输。 全双工：当上述两个条件的任何一个得到满足时，TXE/RXE均自动设置到 “0“，防止进一步的传输与接收。
0	CNFG	R/W	启用FIFO 0:禁用 1:启用 FIFO启用位。 (注2) 如果将<CNFG>设置到 “1 “，则SCxMOD1<FDPX[1:0]>设置即自动将FIFO配置如下: 半双工RX：RX FIFO 4 字节 半双工TX：TX FIFO 4 字节 全双工：RX FIFO 2 字节+TX FIFO存储器 2 字节

注 1: 关于 TX FIFO，要配置的最大字节数始终可用。可用字节数为已写入到 TX FIFO 的字节。

注 2: 该 FIFO 不能在 9 位 UART 模式中使用。

注 3: 在使用 SIO 传输接收中断的直接存储器存取 (DMA)传输中，不能使用 FIFO 功能。

12.4.10 SCxRFC (RX FIFO配置寄存器)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	RFCS	RFIS	-	-	-	-	RIL	
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-8	-	R	读作 “0“。
7	RFCS	W	RX FIFO清除 (注1) 1: 清除 在SCxRFC<RFCS>设置到 “1“时，RX FIFO则被清除，并且SCxRST<RLVL>为 “000 “。以及读取指针得以初始化。
6	RFIS	R/W	选择中断生成条件 0: 当数据达到指定的空满度时 1: 当数据达到指定的空满度时或数据读取时数据超过指定的空满度。
5-2	-	R	读作 “0“。
1-0	RIL[1:0]	R/W	生成RX中断的R/W FIFO空满度

	半双工	全双工
00	4 字节	2 字节
01	1 字节	1 字节
10	2 字节	2 字节
11	3 字节	1 字节

注 1: **使用传输/接收 FIFO 缓冲器时，必须在设置 SIO 传输模式 (半双工 / 全双工)与启用 FIFO (SCxFCNF<CNFG>= “1“)之后清除 TX/RX FIFO。**

注 2: **DMA 传输不通过 FIFO 空满度中生成的中断启动。**

12.4.11 SCxTFC (TX FIFO配置寄存器) (注2)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	TFCS	TFIS	-	-	-	-	TIL	
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-8	-	R	读作 “0“。
7	TFCS	W	TX FIFO清除 (注1) 1:清除 当SCxTST<TFCS>设置到 “1“时，TX FIFO被清除，且SCxTST<TLVL>为 “000“。同时写入指针初始化。
6	TFIS	R/W	选择中断生成条件。 0: 在数据达到指定的空满度时生成中断。 1: 数据达到指定的空满度时或数据读取时数据不能达到指定的空满度时生成中断。
5-2	-	R	读作 “0“。
1-0	TIL[1:0]	R/W	发生传输中断的空满度。 00: 半双工 空 / 全双工 空 01: 半双工 1 字节 / 全双工 1 字节 10: 半双工 2 字节 / 全双工 空 11: 半双工 3 字节 / 全双工 1 字节

注 1: **使用 TX/RX FIFO 缓冲器时，必须在设置 SIO 传输模式 (半双工 / 全双工)且启用 FIFO (SCxFCNF<CNFG>= “1“)之后清除 TX/RX FIFO。**

注 2: **在执行以下操作后，再次配置 SCxTFC 寄存器。SCxEN<SIOE>= “0“ (SIO 操作停止)条件如下:SCxMOD1<I2SC>= “0“ (IDLE 模式下禁止操作)，释放由 WFI (等待中断)指令启动的低功耗模式。**

注 3: **DMA 传输不通过 FIFO 空满度中生成的中断启动。**

12.4.12 SCxRST (RX FIFO状态寄存器)

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

位	比特符号	型号	功能
31-8	-	R	读作 “0“ 。
7	ROR	R	RX FIFO超载 (注) 0:未生成 1:生成
6-3	-	R	读作 “0“。
2-0	RLVL[2:0]	R	RX FIFO空满度状态 000: 空 001: 1 字节 010: 2 字节 011: 3 字节 100: 4 字节

注: 当从 SCxBUF 寄存器中读取接收数据时，<ROR>位即清 “0“。

12.4.13 SCxTST (TX FIFO状态寄存器)

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

位	比特符号	型号	功能
31-8	-	R	读作“0“ 。
7	TUR	R	TX FIFO欠载 (注) 0: 未生成 1: 生成
6-3	-	R	读作“0“。
2-0	TLVL[2:0]	R	传TX FIFO空满度状态 000:空 001: 1 字节 010: 2 字节 011: 3 字节 100: 4 字节

注: 当传输数据写入到 SCxBUF 寄存器时，<TUR>位被清 “0“。

12.5 各模式下的操作

表 12-3 显示各模式与数据格式。

表 12-3 模式与数据格式

模式	模式类型	数据长度	传输方向	指定 是否使用 校验位	停止位长度 (传输)
模式 0	同步通信 模式 (IO 接口模式)	8 位	LSB 优先 / MSB 优先	-	-
模式 1	异步通信模式 (UART 模式)	7 位	LSB 优先	o	1 位或 2 位
模式 2		8 位		o	
模式 3		9 位		×	

模式 0 为同步通信模式，可用于扩展 I/O。该模式传输和接收数据与 SCLK 同步。SCLK 可用于输入和输出。

数据传输的方向可从 LSB 优先与 MSB 优先中选择。不允许该模式添加较验位或停止位。

模式 1，模式 2 及模式 3 均为异步通信模式，且传输方向固定在 LSB 优先。

模式 1 与模式 2 中可添加校验位。模式 3 具有唤醒功能。在该唤醒功能中，主控制器可通过串行链接 (多控制器系统)启动从控制器。

传输中的 STOP 位可从 1 位与 2 位中选择。接收中的 STOP 位长度固定在一个位上。

12.6 数据格式

12.6.1 数据格式列表

图 12-2 显示了数据格式。

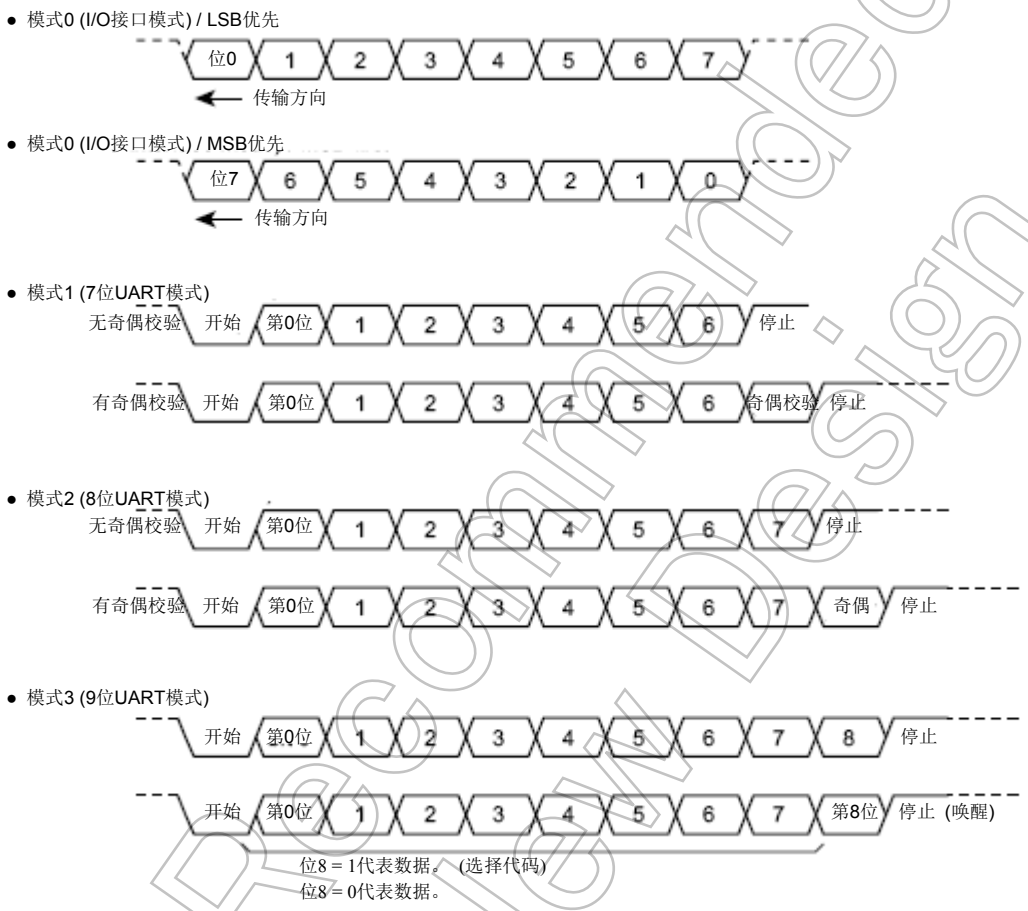


图 12-2 数据格式

12.6.2 奇偶校验控制

仅可在 7-或 8-位 UART 模式中添加奇偶校验位。

设置 “1“ 到 SCxCR<PE>启用奇偶校验。

此 SCxCR<EVEN>选择偶校验或奇校验。

12.6.2.1 传输

进行数据传输时，奇偶校验控制电路自动与传输缓冲器中的数据生成奇偶校验。

数据传输完成后，奇偶校验位以 7-位 UART 模式储存在 SCxBUF<TB7>中，以 8-位 UART 模式储存在 SCxMOD<TB8>中。

<PE>与<EVEN>设置必须在数据写入到传输缓冲器之前完成。

12.6.2.2 接收数据

当接收到的数据从接收移位寄存器转 ~ 接收缓冲器时，生成奇偶校验。

在 7-位 UART 模式中，生成的奇偶校验与储存在 SCxBUF<RB7>中的奇偶校验进行比较，而在 8-位 UART 模式中，则与 SCxCR<RB8>中的奇偶校验进行比较。

当存在差异时，会出现奇偶校验误差，而且 SCxCR 寄存器的<PERR>设置到 “1“。在使用 FIFO 时，<RERR>表示在所接收的数据之一中产生了奇偶校验误差。

12.6.3 停止位长度

UART 传输模式中 STOP 位的长度可通过设置 SCxMOD2<SBLEN>从一个位或两个位中选择。在接收到 STOP 位数据而不管该位的设置时，此 STOP 位数据的长度被判定为一个位。

12.7 时钟控制

12.7.1 预分频器

有一个 7-位预分频器按 2，8，32 及 128 将预分频器输入时钟 φT0 进行分频。

使用时钟/模式控制块中的 CGSYSCR 寄存器选择预分频器的输入时钟 φT0。

只有当波特率发生器被 SCxMOD0<SC[1:0]>= “01“选为传输时钟时，该预分频器才处于活动状态。

表 12-4 (操作频率 64MHz)，表 12-5 (操作频率 48MHz)及表 12-6 (操作频率 32MHz)显示了到波特率发生器的输入时钟分辨率。

表 12-4 波特率发生器的时钟分辨率 fc = 64MHz，fs = 32.768kHz

ΦT0 选择 CGSYSCR <FPSEL1>	外设时钟选择 CGSYSCR <FPSEL0>	时钟齿轮值 CGSYSCR <GEAR[2:0]>	预分频器时钟选择 CGSYSCR <PRCK[2:0]>	预分频器输出时钟分辨率			
				φT1	φT4	φT16	φT64
0	0 (fgear)	000 (fc)	000 (fperiph/1)	$fc/2^1$ (0.0312 μs)	$fc/2^3$ (0.125 μs)	$fc/2^5$ (0.5 μs)	$fc/2^7$ (2.0 μs)
			001 (fperiph/2)	$fc/2^2$ (0.0625 μs)	$fc/2^4$ (0.25 μs)	$fc/2^6$ (1.0 μs)	$fc/2^8$ (4.0 μs)
			010 (fperiph/4)	$fc/2^3$ (0.125 μs)	$fc/2^5$ (0.5 μs)	$fc/2^7$ (2.0 μs)	$fc/2^9$ (8.0 μs)
			011 (fperiph/8)	$fc/2^4$ (0.25 μs)	$fc/2^6$ (1.0 μs)	$fc/2^8$ (4.0 μs)	$fc/2^{10}$ (16.0 μs)
			100 (fperiph/16)	$fc/2^5$ (0.5 μs)	$fc/2^7$ (2.0 μs)	$fc/2^9$ (8.0 μs)	$fc/2^{11}$ (32.0 μs)
			101 (fperiph/32)	$fc/2^6$ (1.0 μs)	$fc/2^8$ (4.0 μs)	$fc/2^{10}$ (16.0 μs)	$fc/2^{12}$ (64.0 μs)
		100 (fc/2)	000 (fperiph/1)	$fc/2^2$ (0.0625 μs)	$fc/2^4$ (0.25 μs)	$fc/2^6$ (1.0 μs)	$fc/2^8$ (4.0 μs)
			001 (fperiph/2)	$fc/2^3$ (0.125 μs)	$fc/2^5$ (0.5 μs)	$fc/2^7$ (2.0 μs)	$fc/2^9$ (8.0 μs)
			010 (fperiph/4)	$fc/2^4$ (0.25 μs)	$fc/2^6$ (1.0 μs)	$fc/2^8$ (4.0 μs)	$fc/2^{10}$ (16.0 μs)
			011 (fperiph/8)	$fc/2^5$ (0.5 μs)	$fc/2^7$ (2.0 μs)	$fc/2^9$ (8.0 μs)	$fc/2^{11}$ (32.0 μs)
			100 (fperiph/16)	$fc/2^6$ (1.0 μs)	$fc/2^8$ (4.0 μs)	$fc/2^{10}$ (16.0 μs)	$fc/2^{12}$ (64.0 μs)
			101 (fperiph/32)	$fc/2^7$ (2.0 μs)	$fc/2^9$ (8.0 μs)	$fc/2^{11}$ (32.0 μs)	$fc/2^{13}$ (128.0 μs)
		101 (fc/4)	000 (fperiph/1)	$fc/2^3$ (0.125 μs)	$fc/2^5$ (0.5 μs)	$fc/2^7$ (2.0 μs)	$fc/2^9$ (8.0 μs)
			001 (fperiph/2)	$fc/2^4$ (0.25 μs)	$fc/2^6$ (1.0 μs)	$fc/2^8$ (4.0 μs)	$fc/2^{10}$ (16.0 μs)
			010 (fperiph/4)	$fc/2^5$ (0.5 μs)	$fc/2^7$ (2.0 μs)	$fc/2^9$ (8.0 μs)	$fc/2^{11}$ (32.0 μs)
			011 (fperiph/8)	$fc/2^6$ (1.0 μs)	$fc/2^8$ (4.0 μs)	$fc/2^{10}$ (16.0 μs)	$fc/2^{12}$ (64.0μs)
			100 (fperiph/16)	$fc/2^7$ (2.0 μs)	$fc/2^9$ (8.0 μs)	$fc/2^{11}$ (32.0 μs)	$fc/2^{13}$ (128.0 μs)
			101 (fperiph/32)	$fc/2^8$ (4.0 μs)	$fc/2^{10}$ (16.0 μs)	$fc/2^{12}$ (64.0μs)	$fc/2^{14}$ (256.0 μs)
		110 (fc/8)	000 (fperiph/1)	$fc/2^4$ (0.25 μs)	$fc/2^6$ (1.0 μs)	$fc/2^8$ (4.0 μs)	$fc/2^{10}$ (16.0 μs)
			001 (fperiph/2)	$fc/2^5$ (0.5 μs)	$fc/2^7$ (2.0 μs)	$c/2^9$ (12.8 μs)	$fc/2^{11}$ (32.0 μs)
			010 (fperiph/4)	$fc/2^6$ (1.0 μs)	$fc/2^8$ (4.0 μs)	$fc/2^{10}$ (16.0 μs)	$fc/2^{12}$ (64.0μs)
			011 (fperiph/8)	$fc/2^7$ (2.0 μs)	$fc/2^9$ (8.0 μs)	$fc/2^{11}$ (32.0 μs)	$fc/2^{13}$ (128.0 μs)
			100 (fperiph/16)	$fc/2^8$ (4.0 μs)	$fc/2^{10}$ (16.0 μs)	$fc/2^{12}$ (64.0μs)	$fc/2^{14}$ (256.0 μs)
			101 (fperiph/32)	$fc/2^9$ (8.0 μs)	$fc/2^{11}$ (32.0 μs)	$fc/2^{13}$ (128.0 μs)	$fc/2^{15}$ (512.0 μs)

表 12-4 波特率发生器的时钟分辨率 fc = 64MHz，fs = 32.768kHz

ΦT0 选择 CGSYSCR <FPSEL1>	外设时钟选择 CGSYSCR <FPSEL0>	时钟齿轮值 CGSYSCR <GEAR[2:0]>	预分频器时钟选择 CGSYSCR <PRCK[2:0]>	预分频器输出时钟分辨率			
				φT1	φT4	φT16	φT64
0	1 (fc)	000 (fc)	000 (fperiph/1)	$fc/2^1$ (00312 μs)	$fc/2^3$ (0.125 μs)	$fc/2^5$ (0.5 μs)	$fc/2^7$ (2.0 μs)
			001 (fperiph/2)	$fc/2^2$ (0.0625 μs)	$fc/2^4$ (0.25 μs)	$fc/2^6$ (1.0 μs)	$fc/2^8$ (4.0 μs)
			010 (fperiph/4)	$fc/2^3$ (0.125 μs)	$fc/2^5$ (0.5 μs)	$fc/2^7$ (2.0 μs)	$fc/2^9$ (8.0 μs)
			011 (fperiph/8)	$fc/2^4$ (0.25 μs)	$fc/2^6$ (1.0 μs)	$fc/2^8$ (4.0 μs)	$fc/2^{10}$ (16.0 μs)
			100 (fperiph/16)	$fc/2^5$ (0.5 μs)	$fc/2^7$ (2.0 μs)	$fc/2^9$ (8.0 μs)	$fc/2^{11}$ (32.0 μs)
			101 (fperiph/32)	$fc/2^6$ (1.0 μs)	$fc/2^8$ (4.0 μs)	$fc/2^{10}$ (16.0 μs)	$fc/2^{12}$ (64.0 μs)
		100 (fc/2)	000 (fperiph/1)	–	$fc/2^3$ (0.125 μs)	$fc/2^5$ (0.5 μs)	$fc/2^7$ (2.0 μs)
			001 (fperiph/2)	$fc/2^2$ (0.0625 μs)	$fc/2^4$ (0.25 μs)	$fc/2^6$ (1.0 μs)	$fc/2^8$ (4.0 μs)
			010 (fperiph/4)	$fc/2^3$ (0.125 μs)	$fc/2^5$ (0.5 μs)	$fc/2^7$ (2.0 μs)	$fc/2^9$ (8.0 μs)
			011 (fperiph/8)	$fc/2^4$ (0.25 μs)	$fc/2^6$ (1.0 μs)	$fc/2^8$ (4.0 μs)	$fc/2^{10}$ (16.0 μs)
			100 (fperiph/16)	$fc/2^5$ (0.5 μs)	$fc/2^7$ (2.0 μs)	$fc/2^9$ (8.0 μs)	$fc/2^{11}$ (32.0 μs)
			101 (fperiph/32)	$fc/2^6$ (1.0 μs)	$fc/2^8$ (4.0 μs)	$fc/2^{10}$ (16.0 μs)	$fc/2^{12}$ (64.0 μs)
		101 (fc/4)	000 (fperiph/1)	–	$fc/2^3$ (0.2 μs)	$fc/2^5$ (0.8 μs)	$fc/2^7$ (3.2 μs)
			001 (fperiph/2)	–	$fc/2^4$ (0.25 μs)	$fc/2^6$ (1.0 μs)	$fc/2^8$ (4.0 μs)
			010 (fperiph/4)	$fc/2^3$ (0.125 μs)	$fc/2^5$ (0.5 μs)	$fc/2^7$ (2.0 μs)	$fc/2^9$ (8.0 μs)
			011 (fperiph/8)	$fc/2^4$ (0.25 μs)	$fc/2^6$ (1.0 μs)	$fc/2^8$ (4.0 μs)	$fc/2^{10}$ (16.0 μs)
			100 (fperiph/16)	$fc/2^5$ (0.5 μs)	$fc/2^7$ (2.0 μs)	$fc/2^9$ (8.0 μs)	$fc/2^{11}$ (32.0 μs)
			101 (fperiph/32)	$fc/2^6$ (1.0 μs)	$fc/2^8$ (4.0 μs)	$fc/2^{10}$ (16.0 μs)	$fc/2^{12}$ (64.0 μs)
		110 (fc/8)	000 (fperiph/1)	–	–	$fc/2^5$ (0.8 μs)	$fc/2^7$ (2.0 μs)
			001 (fperiph/2)	–	$fc/2^4$ (0.25 μs)	$fc/2^6$ (1.0 μs)	$fc/2^8$ (4.0 μs)
			010 (fperiph/4)	–	$fc/2^5$ (0.5 μs)	$fc/2^7$ (2.0 μs)	$fc/2^9$ (8.0 μs)
			011 (fperiph/8)	$fc/2^4$ (0.25 μs)	$fc/2^6$ (1.0 μs)	$fc/2^8$ (4.0 μs)	$fc/2^{10}$ (16.0 μs)
			100 (fperiph/16)	$fc/2^5$ (0.5 μs)	$fc/2^7$ (2.0 μs)	$fc/2^9$ (8.0 μs)	$fc/2^{11}$ (32 μs)
			101 (fperiph/32)	$fc/2^6$ (1.0 μs)	$fc/2^8$ (4.0 μs)	$fc/2^{10}$ (16.0 μs)	$fc/2^{12}$ (64.0 μs)
1	*	*	*	fs/2 (61μs)	$fs/2^3$ (244 μs)	$fs/2^5$ (977 μs)	$fs/2^7$ (3.91 ms)

注 1: **必须选择预分频器输出时钟 φTn，以满足关系式 "φTn ≤ fsys /2 "(使 φTn 慢于 fsys)。**

注 2: **SIO 操作时勿更改时钟齿轮。**

注 3: **"–"表示该设置被禁止，"*"表示忽略上表。**

表 12-5 波特率发生器的时钟分辨率 fc = 48MHz，fs = 32.768kHz

ΦT0 选择 CGSYSCR <FPSEL1>	外设时钟选择 CGSYSCR <FPSEL0>	时钟齿轮值 CGSYSCR <GEAR[2:0]>	预分频器时钟选择 CGSYSCR <PRCK[2:0]>	预分频器输出时钟分辨率			
				φT1	φT4	φT16	φT64
0	0 (fgear)	000 (fc)	000 (fperiph/1)	$fc/2^1$ (0.0417 μs)	$fc/2^3$ (0.167 μs)	$fc/2^5$ (0.667 μs)	$fc/2^7$ (2.67 μs)
			001 (fperiph/2)	$fc/2^2$ (0.0833 μs)	$fc/2^4$ (0.333 μs)	$fc/2^6$ (1.33 μs)	$fc/2^8$ (5.33 μs)
			010 (fperiph/4)	$fc/2^3$ (0.167 μs)	$fc/2^5$ (0.667 μs)	$fc/2^7$ (2.67 μs)	$fc/2^9$ (10.7 μs)
			011 (fperiph/8)	$fc/2^4$ (0.333 μs)	$fc/2^6$ (1.33 μs)	$fc/2^8$ (5.33 μs)	$fc/2^{10}$ (21.3 μs)
			100 (fperiph/16)	$fc/2^5$ (0.667 μs)	$fc/2^7$ (2.67 μs)	$fc/2^9$ (10.7 μs)	$fc/2^{11}$ (42.7 μs)
			101 (fperiph/32)	$fc/2^6$ (1.33 μs)	$fc/2^8$ (5.33 μs)	$fc/2^{10}$ (21.3 μs)	$fc/2^{12}$ (85.3 μs)
		100 (fc/2)	000 (fperiph/1)	$fc/2^2$ (0.0833 μs)	$fc/2^4$ (0.333 μs)	$fc/2^6$ (1.33 μs)	$fc/2^8$ (5.33 μs)
			001 (fperiph/2)	$fc/2^3$ (0.167 μs)	$fc/2^5$ (0.667 μs)	$fc/2^7$ (2.67 μs)	$fc/2^9$ (10.7 μs)
			010 (fperiph/4)	$fc/2^4$ (0.333 μs)	$fc/2^6$ (1.33 μs)	$fc/2^8$ (5.33 μs)	$fc/2^{10}$ (21.3 μs)
			011 (fperiph/8)	$fc/2^5$ (0.667 μs)	$fc/2^7$ (2.67 μs)	$fc/2^9$ (10.7 μs)	$fc/2^{11}$ (42.7 μs)
			100 (fperiph/16)	$fc/2^6$ (1.33 μs)	$fc/2^8$ (5.33 μs)	$fc/2^{10}$ (21.3 μs)	$fc/2^{12}$ (85.3 μs)
			101 (fperiph/32)	$fc/2^7$ (2.67 μs)	$fc/2^9$ (10.7 μs)	$fc/2^{11}$ (42.7 μs)	$fc/2^{13}$ (171 μs)
		101 (fc/4)	000 (fperiph/1)	$fc/2^3$ (0.167 μs)	$fc/2^5$ (0.667 μs)	$fc/2^7$ (2.67 μs)	$fc/2^9$ (10.7 μs)
			001 (fperiph/2)	$fc/2^4$ (0.333 μs)	$fc/2^6$ (1.33 μs)	$fc/2^8$ (5.33 μs)	$fc/2^{10}$ (21.3 μs)
			010 (fperiph/4)	$fc/2^5$ (0.667 μs)	$fc/2^7$ (2.67 μs)	$fc/2^9$ (10.7 μs)	$fc/2^{11}$ (42.7 μs)
			011 (fperiph/8)	$fc/2^6$ (1.33 μs)	$fc/2^8$ (5.33 μs)	$fc/2^{10}$ (21.3 μs)	$fc/2^{12}$ (85.3 μs)
			100 (fperiph/16)	$fc/2^7$ (2.67 μs)	$fc/2^9$ (10.7 μs)	$fc/2^{11}$ (42.7 μs)	$fc/2^{13}$ (171 μs)
			101 (fperiph/32)	$fc/2^8$ (5.33 μs)	$fc/2^{10}$ (21.3 μs)	$fc/2^{12}$ (85.3μs)	$fc/2^{14}$ (341 μs)
		110 (fc/8)	000 (fperiph/1)	$fc/2^4$ (0.333 μs)	$fc/2^6$ (1.33 μs)	$fc/2^8$ (5.33 μs)	$fc/2^{10}$ (21.3 μs)
			001 (fperiph/2)	$fc/2^5$ (0.667 μs)	$fc/2^7$ (2.67 μs)	$fc/2^9$ (10.7 μs)	$fc/2^{11}$ (42.7 μs)
			010 (fperiph/4)	$fc/2^6$ (1.33 μs)	$fc/2^8$ (5.33 μs)	$fc/2^{10}$ (21.3 μs)	$fc/2^{12}$ (85.3 μs)
			011 (fperiph/8)	$fc/2^7$ (2.67 μs)	$fc/2^9$ (10.7 μs)	$fc/2^{11}$ (42.7 μs)	$fc/2^{13}$ (171 μs)
			100 (fperiph/16)	$fc/2^8$ (5.33 μs)	$fc/2^{10}$ (21.3 μs)	$fc/2^{12}$ (85.3μs)	$fc/2^{14}$ (341 μs)
			101 (fperiph/32)	$fc/2^9$ (10.7 μs)	$fc/2^{11}$ (42.7 μs)	$fc/2^{13}$ (171 μs)	$fc/2^{15}$ (683 μs)

表 12-5 波特率发生器的时钟分辨率 fc = 48MHz，fs = 32.768kHz

ΦT0 选择 CGSYSCR <FPSEL1>	外设时钟选择 CGSYSCR <FPSEL0>	时钟齿轮值 CGSYSCR <GEAR[2:0]>	预分频器时钟选择 CGSYSCR <PRCK[2:0]>	预分频器输出时钟分辨率			
				φT1	φT4	φT16	φT64
0	1 (fc)	000 (fc)	000 (fperiph/1)	$fc/2^1$ (0.0417 μs)	$fc/2^3$ (0.167 μs)	$fc/2^5$ (0.667 μs)	$fc/2^7$ (2.67 μs)
			001 (fperiph/2)	$fc/2^2$ (0.0833 μs)	$fc/2^4$ (0.333 μs)	$fc/2^6$ (1.33 μs)	$fc/2^8$ (5.33 μs)
			010 (fperiph/4)	$fc/2^3$ (0.167 μs)	$fc/2^5$ (0.667 μs)	$fc/2^7$ (2.67 μs)	$fc/2^9$ (10.7 μs)
			011 (fperiph/8)	$fc/2^4$ (0.333 μs)	$fc/2^6$ (1.33 μs)	$fc/2^8$ (5.33 μs)	$fc/2^{10}$ (21.3 μs)
			100 (fperiph/16)	$fc/2^5$ (0.667 μs)	$fc/2^7$ (2.67 μs)	$fc/2^9$ (10.7 μs)	$fc/2^{11}$ (42.7 μs)
			101 (fperiph/32)	$fc/2^6$ (1.33 μs)	$fc/2^8$ (5.33 μs)	$fc/2^{10}$ (21.3 μs)	$fc/2^{12}$ (85.3 μs)
		100 (fc/2)	000 (fperiph/1)	-	$fc/2^3$ (0.167 μs)	$fc/2^5$ (0.667 μs)	$fc/2^7$ (2.67 μs)
			001 (fperiph/2)	$fc/2^2$ (0.0833 μs)	$fc/2^4$ (0.333 μs)	$fc/2^6$ (1.33 μs)	$fc/2^8$ (5.33 μs)
			010 (fperiph/4)	$fc/2^3$ (0.167 μs)	$fc/2^5$ (0.667 μs)	fc/27 (2.67 μs)	$fc/2^9$ (10.7 μs)
			011 (fperiph/8)	$fc/2^4$ (0.333 μs)	$fc/2^6$ (1.33 μs)	$fc/2^8$ (5.33 μs)	$fc/2^{10}$ (21.3 μs)
			100 (fperiph/16)	$fc/2^5$ (0.667 μs)	$fc/2^7$ (2.67 μs)	$fc/2^9$ (10.7 μs)	$fc/2^{11}$ (42.7 μs)
			101 (fperiph/32)	$fc/2^6$ (1.33 μs)	$fc/2^8$ (5.33 μs)	$fc/2^{10}$ (21.3 μs)	$fc/2^{12}$ (85.3 μs)
		101 (fc/4)	000 (fperiph/1)	-	$fc/2^3$ (0.167 μs)	$fc/2^5$ (0.667 μs)	$fc/2^7$ (2.67 μs)
			001 (fperiph/2)	-	$fc/2^4$ (0.333 μs)	$fc/2^6$ (1.33 μs)	$fc/2^8$ (5.33 μs)
			010 (fperiph/4)	$fc/2^3$ (0.167 μs)	$fc/2^5$ (0.667 μs)	$fc/2^7$ (2.67 μs)	$fc/2^9$ (10.7 μs)
			011 (fperiph/8)	$fc/2^4$ (0.333 μs)	$fc/2^6$ (1.33 μs)	$fc/2^8$ (5.33 μs)	$fc/2^{10}$ (21.3 μs)
			100 (fperiph/16)	$fc/2^5$ (0.667 μs)	$fc/2^7$ (2.67 μs)	$fc/2^9$ (10.7 μs)	$fc/2^{11}$ (42.7 μs)
			101 (fperiph/32)	$fc/2^6$ (1.33 μs)	$fc/2^8$ (5.33 μs)	$fc/2^{10}$ (21.3 μs)	$fc/2^{12}$ (85.3 μs)
		110 (fc/8)	000 (fperiph/1)	-	-	$fc/2^5$ (0.667 μs)	$fc/2^7$ (2.67 μs)
			001 (fperiph/2)	-	$fc/2^4$ (0.333 μs)	$fc/2^6$ (1.33 μs)	$fc/2^8$ (5.33 μs)
			010 (fperiph/4)	-	$fc/2^5$ (0.667 μs)	$fc/2^7$ (2.67 μs)	$fc/2^9$ (10.7 μs)
			011 (fperiph/8)	$fc/2^4$ (0.333 μs)	$fc/2^6$ (1.33 μs)	$fc/2^8$ (5.33 μs)	$fc/2^{10}$ (21.3 μs)
			100 (fperiph/16)	$fc/2^5$ (0.667 μs)	$fc/2^7$ (2.67 μs)	$fc/2^9$ (10.7 μs)	$fc/2^{11}$ (42.7 μs)
			101 (fperiph/32)	$fc/2^6$ (1.33 μs)	$fc/2^8$ (5.33 μs)	$fc/2^{10}$ (21.3 μs)	$fc/2^{12}$ (85.3 μs)
1	*	*	*	fs/2 (61μs)	$fs/2^3$ (244 μs)	$fs/2^5$ (977 μs)	$fs/2^7$ (3.91 ms)

注 1: **必须选择预分频器输出时钟脉冲 φTn，以满足关系式 "φTn ≤ fsys /2" (使 φTn 慢于 fsys)。**

注 2: **SIO 操作时勿更改时钟齿轮。**

注 3: **"–"表示该设置被禁止， "*"表示忽略上表。**

表 12-6 波特率发生器的时钟分辨率 fc = 32MHz，fs = 32.768kHz

ΦT0 选择 CGSYSCR <FPSEL1>	外设时钟选择 CGSYSCR <FPSEL0>	时钟齿轮值 CGSYSCR <GEAR[2:0]>	预分频器时钟选择 CGSYSCR <PRCK[2:0]>	预分频器输出时钟分辨率			
				φT1	φT4	φT16	φT64
0	0 (fgear)	000 (fc)	000 (fperiph/1)	$fc/2^1$ (0.0625 μs)	$fc/2^3$ (0.25 μs)	$fc/2^5$ (1.0 μs)	$fc/2^7$ (4.0 μs)
			001 (fperiph/2)	$fc/2^2$ (0.125 μs)	$fc/2^4$ (0.5 μs)	$fc/2^6$ (2.0 μs)	$fc/2^8$ (8.0 μs)
			010 (fperiph/4)	$fc/2^3$ (0.25 μs)	$fc/2^5$ (1.0 μs)	$fc/2^7$ (4.0 μs)	$fc/2^9$ (16.0 μs)
			011 (fperiph/8)	$fc/2^4$ (0.5 μs)	$fc/2^6$ (2.0 μs)	$fc/2^8$ (8.0 μs)	$fc/2^{10}$ (32.0 μs)
			100 (fperiph/16)	$fc/2^5$ (1.0 μs)	$fc/2^7$ (4.0 μs)	$fc/2^9$ (16.0 μs)	$fc/2^{11}$ (64.0 μs)
			101 (fperiph/32)	$fc/2^6$ (2.0 μs)	$fc/2^8$ (8.0 μs)	$fc/2^{10}$ (32.0 μs)	$fc/2^{12}$ (128.0 μs)
		100 (fc/2)	000 (fperiph/1)	$fc/2^2$ (0.125 μs)	$fc/2^4$ (0.5 μs)	$fc/2^6$ (2.0 μs)	$fc/2^8$ (8.0 μs)
			001 (fperiph/2)	$fc/2^3$ (0.25 μs)	$fc/2^5$ (1.0 μs)	$fc/2^7$ (4.0 μs)	$fc/2^9$ (16.0 μs)
			010 (fperiph/4)	$fc/2^4$ (0.5 μs)	$fc/2^6$ (2.0 μs)	$fc/2^8$ (8.0 μs)	$fc/2^{10}$ (32.0 μs)
			011 (fperiph/8)	$fc/2^5$ (1.0 μs)	$fc/2^7$ (4.0 μs)	$fc/2^9$ (16.0 μs)	$fc/2^{11}$ (64.0 μs)
			100 (fperiph/16)	$fc/2^6$ (2.0 μs)	$fc/2^8$ (8.0 μs)	$fc/2^{10}$ (32.0 μs)	$fc/2^{12}$ (128.0 μs)
			101 (fperiph/32)	$fc/2^7$ (4.0 μs)	$fc/2^9$ (16.0 μs)	$fc/2^{11}$ (64.0 μs)	$fc/2^{13}$ (256.0 μs)
		101 (fc/4)	000 (fperiph/1)	$fc/2^3$ (0.25 μs)	$fc/2^5$ (1.0 μs)	$fc/2^7$ (4.0 μs)	$fc/2^9$ (16.0 μs)
			001 (fperiph/2)	$fc/2^4$ (0.5 μs)	$fc/2^6$ (2.0 μs)	$fc/2^8$ (8.0 μs)	$fc/2^{10}$ (32.0 μs)
			010 (fperiph/4)	$fc/2^5$ (1.0 μs)	$fc/2^7$ (4.0 μs)	$fc/2^9$ (16.0 μs)	$fc/2^{11}$ (64.0 μs)
			011 (fperiph/8)	$fc/2^6$ (2.0 μs)	$fc/2^8$ (8.0 μs)	$fc/2^{10}$ (32.0 μs)	$fc/2^{12}$ (128.0 μs)
			100 (fperiph/16)	$fc/2^7$ (4.0 μs)	$fc/2^9$ (16.0 μs)	$fc/2^{11}$ (64.0 μs)	$fc/2^{13}$ (256.0 μs)
			101 (fperiph/32)	$fc/2^8$ (8.0 μs)	$fc/2^{10}$ (32.0 μs)	$fc/2^{12}$ (128.0 μs)	$fc/2^{14}$ (512.0 μs)
		110 (fc/8)	000 (fperiph/1)	$fc/2^4$ (0.5 μs)	$fc/2^6$ (2.0 μs)	$fc/2^8$ (8.0 μs)	$fc/2^{10}$ (32.0 μs)
			001 (fperiph/2)	$fc/2^5$ (1.0 μs)	$fc/2^7$ (4.0 μs)	$fc/2^9$ (16.0 μs)	$fc/2^{11}$ (64.0 μs)
			010 (fperiph/4)	$fc/2^6$ (2.0 μs)	$fc/2^8$ (8.0 μs)	$fc/2^{10}$ (32.0 μs)	$fc/2^{12}$ (128.0 μs)
			011 (fperiph/8)	$fc/2^7$ (4.0 μs)	$fc/2^9$ (16.0 μs)	$fc/2^{11}$ (64.0 μs)	$fc/2^{13}$ (256.0 μs)
			100 (fperiph/16)	$fc/2^8$ (8.0 μs)	$fc/2^{10}$ (32.0 μs)	$fc/2^{12}$ (128.0 μs)	$fc/2^{14}$ (512.0 μs)
			101 (fperiph/32)	$fc/2^9$ (16.0 μs)	$fc/2^{11}$ (64.0 μs)	$fc/2^{13}$ (256.0 μs)	$fc/2^{15}$ (1024 μs)

表 12-6 波特率发生器的时钟分辨率 fc = 32MHz，fs = 32.768kHz

ΦT0 选择 CGSYSCR <FPSEL1>	外设时钟选择 CGSYSCR <FPSEL0>	时钟齿轮值 CGSYSCR <GEAR[2:0]>	预分频器时钟选择 CGSYSCR <PRCK[2:0]>	预分频器输出时钟分辨率			
				φT1	φT4	φT16	φT64
0	1 (fc)	000 (fc)	000 (fperiph/1)	$fc/2^1$ (0.0625 μs)	$fc/2^3$ (0.25 μs)	$fc/2^5$ (1.0 μs)	$fc/2^7$ (4.0 μs)
			001 (fperiph/2)	$fc/2^2$ (0.125 μs)	$fc/2^4$ (0.5 μs)	$fc/2^6$ (2.0 μs)	$fc/2^8$ (8.0 μs)
			010 (fperiph/4)	$fc/2^3$ (0.25 μs)	$fc/2^5$ (1.0 μs)	$fc/2^7$ (4.0 μs)	$fc/2^9$ (16.0 μs)
			011 (fperiph/8)	$fc/2^4$ (0.5 μs)	$fc/2^6$ (2.0 μs)	$fc/2^8$ (8.0 μs)	$fc/2^{10}$ (32.0 μs)
			100 (fperiph/16)	$fc/2^5$ (1.0 μs)	$fc/2^7$ (4.0 μs)	$fc/2^9$ (16.0 μs)	$fc/2^{11}$ (64.0 μs)
			101 (fperiph/32)	$fc/2^6$ (2.0 μs)	$fc/2^8$ (8.0 μs)	$fc/2^{10}$ (32.0 μs)	$fc/2^{12}$ (128.0 μs)
		100 (fc/2)	000 (fperiph/1)	-	$fc/2^3$ (0.25 μs)	$fc/2^5$ (1.0 μs)	$fc/2^7$ (4.0 μs)
			001 (fperiph/2)	$fc/2^2$ (0.125 μs)	$fc/2^4$ (0.5 μs)	$fc/2^6$ (2.0 μs)	$fc/2^8$ (8.0 μs)
			010 (fperiph/4)	$fc/2^3$ (0.25 μs)	$fc/2^5$ (1.0 μs)	$fc/2^7$ (4.0 μs)	$fc/2^9$ (16.0 μs)
			011 (fperiph/8)	$fc/2^4$ (0.5 μs)	$fc/2^6$ (2.0 μs)	$fc/2^8$ (8.0 μs)	$fc/2^{10}$ (32.0 μs)
			100 (fperiph/16)	$fc/2^5$ (1.0 μs)	$fc/2^7$ (4.0 μs)	$fc/2^9$ (16.0 μs)	$fc/2^{11}$ (64.0 μs)
			101 (fperiph/32)	$fc/2^6$ (2.0 μs)	$fc/2^8$ (8.0 μs)	$fc/2^{10}$ (32.0 μs)	$fc/2^{12}$ (128.0 μs)
		101 (fc/4)	000 (fperiph/1)	-	$fc/2^3$ (0.25 μs)	$fc/2^5$ (1.0 μs)	$fc/2^7$ (4.0 μs)
			001 (fperiph/2)	-	$fc/2^4$ (0.5 μs)	$fc/2^6$ (2.0 μs)	$fc/2^8$ (8.0 μs)
			010 (fperiph/4)	$fc/2^3$ (0.25 μs)	$fc/2^5$ (1.0 μs)	$fc/2^7$ (4.0 μs)	$fc/2^9$ (16.0 μs)
			011 (fperiph/8)	$fc/2^4$ (0.5 μs)	$fc/2^6$ (2.0 μs)	$fc/2^8$ (8.0 μs)	$fc/2^{10}$ (32.0 μs)
			100 (fperiph/16)	$fc/2^5$ (1.0 μs)	$fc/2^7$ (4.0 μs)	$fc/2^9$ (16.0 μs)	$fc/2^{11}$ (64.0 μs)
			101 (fperiph/32)	$fc/2^6$ (2.0 μs)	$fc/2^8$ (8.0 μs)	$fc/2^{10}$ (32.0 μs)	$fc/2^{12}$ (128.0 μs)
		110 (fc/8)	000 (fperiph/1)	-	-	$fc/2^5$ (1.0 μs)	$fc/2^7$ (4.0 μs)
			001 (fperiph/2)	-	$fc/2^4$ (0.5 μs)	$fc/2^6$ (2.0 μs)	$fc/2^8$ (8.0 μs)
			010 (fperiph/4)	-	$fc/2^5$ (1.0 μs)	$fc/2^7$ (4.0 μs)	$fc/2^9$ (16.0 μs)
			011 (fperiph/8)	$fc/2^4$ (0.5 μs)	$fc/2^6$ (2.0 μs)	$fc/2^8$ (8.0 μs)	$fc/2^{10}$ (32.0 μs)
			100 (fperiph/16)	$fc/2^5$ (1.0 μs)	$fc/2^7$ (4.0 μs)	$fc/2^9$ (16.0 μs)	$fc/2^{11}$ (64.0 μs)
			101 (fperiph/32)	$fc/2^6$ (2.0 μs)	$fc/2^8$ (8.0 μs)	$fc/2^{10}$ (32.0 μs)	$fc/2^{12}$ (128.0 μs)
1	*	*	*	fs/2 (61μs)	$fs/2^3$ (244 μs)	$fs/2^5$ (977 μs)	$fs/2^7$ (3.91 ms)

注 1: **必须选择预分频器输出时钟脉冲 φTn，以满足关系式 "φTn ≤ fsys /2" (使 φTn 慢于 fsys)。**

注 2: **SIO 操作时勿更改时钟齿轮。**

注 3: **"–"表示该设置被禁止，"*"表示忽略上表。**

12.7.2 串行时钟生成电路

串行时钟电路为生成传输与接收时钟脉冲 (SIOCLK)的块，由各个电路组成。在这些电路中，可通过波特率发生器的设置与模式来选择时钟脉冲。

12.7.2.1 波特率发生器

波特率发生器生成传输与接收时钟块，以确定串行通道传输速率。

(1) 波特率发生器输入时钟

波特率发生器的输入时钟从预分频器输出中选择，预分频器输出除 2，8，32 及 128 进行分频。

此输入时钟通过设置 SCxBRCR<BRCK>来选择。

(2) 波特率发生器输出时钟

波特率发生器中输出时钟的分频率通过 SCxBRCR 与 SCxBRADD 设置。

可采用以下分频率；I/O 接口模式中的 1/N 分频，UART 模式中的 1/N 或 N + (16-K)/16。

下表显示了可以选择的分频率。

模式	分频功能设置 SCxBRCR<BRADDE>	用 N 分频 SCxBRCR<BRS>	用 K 分频 SCxBRADD<BRK>-
I/O 接口	用 N 分频	1 ~ 16 (注)	-
UART	用 N 分频	1 ~ 16	-
	N + (16-K)/16分频	2 ~ 15	1 ~ 15

注: **只有当启用双缓冲器时才可使用 1/N (N=1)分频率。**

12.7.2.2 时钟选择电路

可通过设置模式和寄存器来选择时钟。

可通过设置 SCxMOD0<SM>指定各个模式。

通过设置 SCxCR 选择 I/O 接口模式中的输入时钟。

通过设置 SCxMOD0<SC>选择 UART 模式中的时钟。

(1) I/O 接口模式中的传输时钟

表 12-7 显示 I/O 接口模式中的时钟选择。

表 12-7 I/O 接口模式中的时钟选择

模式 SCxMOD0<SM>	I/O选择 SCxCR<IOC>	时钟沿选择 SCxCR<SCLKS>	用时钟
I/O 接口模式	SCLK输出	设置到 “0“ (固定到上升沿)	用2分频波特率发生器 输出
	SCLK输入	上升沿	SCLK输入上升沿
		下降沿	SCLK输入下降沿

获取最高的波特率时，波特率发生器必须设置如下。

注:**在决定时钟设置时，应确信满足 AC 电气特性。**

时钟/模式控制块设置

- fc = 40MHz
- fgear = 40MHz (CGSYSCR<GEAR[2:0]>= “000“: fc 已选择)
- φT0 = 40MHz (CGSYSCR<PRCK[2:0:] >= “000“: 1 分频比)

SIO 设置 (使用双缓冲器)

- 时钟 (SCxBRCR<BRCK[1:0]>= “00“: φT1 已选择) = 20MHz
- 已分频时钟脉冲频率 (SCxBRCR<BRS[3:0] >= “0001“: 1 分频比) = 20MHz

当使用双缓冲器时，可选择 1 分频比。在这种情况下，由于 20MHz 被 2 除，波特率为 10Mbps。

SIO 设置 (未使用双缓冲器)

- 时钟 (SCxBRCR<BRCK[1:0]>= “00“: φT1 已选择) = 20MHz
- 已分频时钟脉冲频率 (SCxBRCR<BRS[3:0] >= “0010“ : 2 分频比) = 10MHz

未使用双缓冲器时，2 分频比为最高。在这种情况下，由于 10MHz 被 2 除，波特率为 5Mbps

使用 SCLK 输入时，必须满足以下条件。

使用双缓冲器时

- SCLK 循环 > 6/fsys

最高波特率小于 40 ÷ 6 = 6.66Mbps。

未使用双缓冲器时

- SCLK 循环 > 8/fsys

最高波特率小于 40 ÷ 8 = 5.0Mbps。

(2) UART 模式中的传输时钟

表 12-8 显示 UART 模式中的时钟选择。在 UART 模式中，在使用之前，接收计数器或传输计数器中所选择的时钟频率除以 16。

表 12-8 UART 模式中的时钟选择

模式 SCxMOD0<SM>	时钟选择 SCxMOD0<SC>
UART模式	计时器输出
	波特率发生器
	f_{sys}
	SCLK输入

各时钟设置中的波特率示例。

使用波特率发生器时。

- fc = 40MHz
- fgear = 40MHz (CGSYSCR<GEAR[2:0]> = "000" : fc 已选择)
- φT0 = 40MHz (CGSYSCR<PRCK[2:0]> = "000" : 1 分频比)
- Clock = φT1 = 20MHz (SCxBRCR<BRCK[1:0]>= "00" : φT1 已选择)

由于 20MHz 除以 16，最高波特率为 1.25MHz。

表 12-9 显示使用以下时钟设置的波特率发生器时的波特率示例。

fc = 9.8304MHz

fgear = 9.8304MHz (CGSYSCR<GEAR[2:0]> = "000" :fc 已选择)

ΦT0 = 4.9152MHz (CGSYSCR<PRCK[2:0]> = "001" :2 分频比)

表 12-9 UART 模式波特率示例 (使用波特率发生器)

fc [MHz]	分频比 N (SCxBRCR<BRS[3:0]>)	φT1 (fc/4)	φT4 (fc/16)	φT16 (fc/64)	φT64 (fc/256)
9.830400	2	76.800	19.200	4.800	1.200
	4	38.400	9.600	2.400	0.600
	8	19.200	4.800	1.200	0.300
	16	9.600	2.400	0.600	0.150

单位:kbps

利用 SCLK 输入时

应使用 SCLK 输入，必须满足以下条件。

- SCLK 循环 > 2/fsys

最高波特率必须小于 40 ÷ 2 ÷ 16 = 1.25 Mbps

使用 fsys 时

由于最高 fsys 值为 40MHz，最高波特率则为 40 ÷ 16 = 2.5 Mbps。

使用计时器输出时

启用计时器输出时，必须设置以下条件:计时器触发器输出在计数器的值与 TBxRG1 的值匹配时转换。SIOCLK 时钟脉冲频率为 "TBxRG1×2 的设定值"。

波特率可使用以下方程式求出。

波特率计算

$$\text{传输率} = \frac{\text{CGSYSCR<PRCK[1:0]>选择的时钟脉冲频率}}{(\text{TBxRG1} \times 2) \times 2 \times 16}$$

(TBxRG1 × 2)：一个时钟脉冲周期为计时器触发器转换两次的期限。

2：如果选择了计时器预分频器时钟fT1 (2分频比)

表 12-10 显示采用以下时钟设置使用计时器输出时的波特率示例。

fc = 32MHz / 9.8304MHz / 8MHz

fgear = 32MHz / 9.8304MHz / 8MHz (CGSYSCR<GEAR[2:0]> = “000“ :fc 已选择)

φT0 = 16MHz / 4.9152MHz / 4MHz (CGSYSCR<PRCK[2:0]> = “001“ :2 分频)

计时器计数时钟

= 4MHz / 1.2287MHz / 1MHz (TBxMOD<TBCLK[1:0]>= “01“ :φT1 已选择)

表 12-10 UART 模式波特率的示例 (使用计时器输出)

TBxRG设置	fc		
	32MHz	9.8304MHz	8MHz
0x0001	250	76.8	62.5
0x0002	125	38.4	31.25
0x0003	-	25.6	-
0x0004	62.5	19.2	15.625
0x0005	50	15.36	12.5
0x0006	-	12.8	-
0x0008	31.25	9.6	-
0x000A	25	7.68	6.25
0x0010	15.625	4.8	-
0x0014	12.5	3.84	3.125

单位 : kbps

12.8 传输 / 接收缓冲器与FIFO

12.8.1 配置

图 12-3 显示了传输缓冲器接收缓冲器与 FIFO 的配置.

使用缓冲器与 FIFO 需要正确设置。配置可依据模式进行预定。

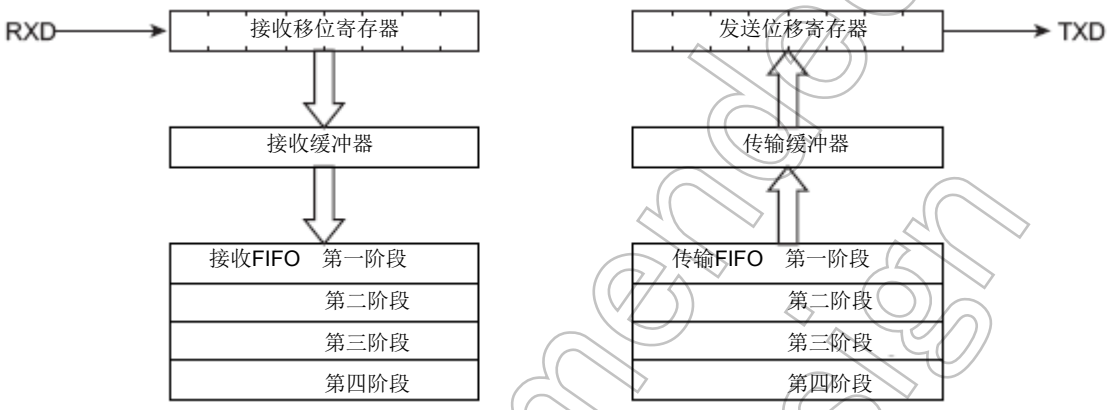


图 12-3 缓冲器与 FIFO 的配置

12.8.2 传输 / 接收缓冲器

传输缓冲器与接收缓冲器为双缓冲器。该缓冲器配置由 SCxMOD2<WBUF>指定。

在使用接收缓冲器情况下，如果 SCLK 输入设置成以 I/O 接口模式生成时钟脉冲输出，或选择 UART 模式，即使是<WBUF>设置，那也是双缓冲器。在其它模式中，则依据<WBUF>设置。

表 12-11 显示了模式与缓冲器之间的相关性。

表 12-11 模式与缓冲器的组成

模式		SCxMOD2<WBUF>	
		“0“	“1“
UART	传输	单	双
	接收	双	双
I/O接口 (SCLK输入)	传输	单	双
	接收	双	双
I/O接口 (SCLK输出)	传输	单	双
	接收	单	双

12.8.3 FIFO

除以上所述的双缓冲器功能之外，还可使用 4-字节 FIFO。

要启用 FIFO，通过将 SCxMOD2<WBUF>设置到 “1“，将 SCxFCNF<CNFG>设置到 “1“，即可启用双缓冲器。FIFO 缓冲器配置由 SCxMOD1<FDPX[1:0]>指定。

注:**使用 TX/RX FIFO 缓冲器时，必须在设置 SIO 传输模式 (半双工/ 全双工)与启用 FIFO (SCxFCNF<CNFG>= "1")之后清除 TX/RX FIFO。**

表 12-12 显示模式与 FIFO 之间的关系。

表 12-12 模式与 FIFO 组成

	SCxMOD1<FDPX[1:0]>	RX FIFO	TX FIFO
半双工 RX	"01"	4字节	-
半双工 TX	"10"	-	4字节
全双工	"11"	2字节	2字节

12.9 状态标志

SCxMOD2 寄存器有两类标志。只有当启用双缓冲器时此位才显得重要。

<RBFLL>为显示接收缓冲器已满的标志。在接收到一个数据帧时，该数据即从接收移位寄存器转 ~ 接收缓冲器，在读取该位将其转变到 "0"时，该位则转变到 "1"。

<TBEMP>显示该传输缓冲器为空。在传输缓冲器中的数据转到传输移位寄存器时，该位被设置到 "1"，当数据被设置到传输缓冲器时，该位则被清除到 "0"。

12.10 错误标志

SCxCR 寄存器中设有三个错误标志。此标志的含意依据模式而改变。下表显示各模式中的含意。

在读取 SCxCR 寄存器之后，这些标志被清到 "0"。

模式	标志		
	<OERR>	<PERR>	<FERR>
UART	超载错误	奇偶检验错误	成帧错误
I/O接口 (SCLK输入)	超载错误	欠载错误 (在使用双缓冲器或FIFO时)	固定到 "0"
		固定到 "0" (不使用双缓冲器与FIFO时)	
I/O接口 (SCLK输出)	未定义	未定义	固定到 "0"

12.10.1 OERR标志

在 UART 和 I/O 两种接口模式中，在接收缓冲器已经读取之前完成下一帧接收数据接收而产生错误时，该位被设置到 "1"。当启用 RX FIFO 时，接收数据则自动转到 RX FIFO，直到 RX FIFO 已满 (或直到可用位被全部占用)均不会产生超载错误。

在带 SCLK 输出模式的 I/O 接口中，一旦设置了该标志，SCLK 输出即停止。

注:**将 I/O 接口 SCLK 输出模式切换到其它模式时，读取 SCxCR 寄存器并清除超载标志。**

12.10.2 PERR标志

此标志表示 UART 模式中的奇偶检验错误且 I/O 接口模式中的欠载错误。

在 UART 模式中，已接收数据生成的奇偶校验与已接收的奇偶校验不同时，<PERR>在被设置到“1“。

在 I/O 接口模式下，当启用双缓冲器时，在以下条件下<PERR>被设置到 “1“。

在 SCLK 输入模式下，当完成传输移位寄存器的数据输出，传输缓冲器中无数据后再输入 SCLK 时，<PERR>被设置到 “1“。

在 SCLK 输出模式下，当完成所有数据输出后，<PERR>被设置到 “1“，SCLK 输出停止。

注:**将 I/O 接口 SCLK 输出模式切换到其它模式时，读取 SCxCR 寄存器并清除欠载标志**。

12.10.3 FERR标志

如果相应停止位通过在中心周围进行位取样而被确定为 “0“，则生成成帧错误。不管 SCxMOD2<SBLEN>寄存器中的停止位长度设置如何，停止位的状态仅由 1 来确定。

在 I/O 接口模式下，该位被固定到 “0“。

12.11 接收

12.11.1 接收计数器

接收计数器为 4-位二进制计数器，通过 SIOCLK 进行递增计数。

在 UART 模式下，使用十六个 SIOCLK 时钟脉冲接收单个数据位，并在第七，第八及第九个脉冲处对数据符号采样。在这三个样品中，用多数逻辑决定接收的数据。

12.11.2 接收控制装置

12.11.2.1 I/O 接口模式

在 SCLK 输出模式中，将 SCxCR<IOC>设置到 "0"，在输出到 SCLK 引脚的移位时钟的上升沿进行 RXD 引脚采样。

在 SCLK 输入模式下，将 SCxCR<IOC>设置到 "1"，串行接收数据的 RXD 引脚，依据 SCxCR<SCLKS>设置，在 SCLK 输入信号的上升沿或下降沿上进行采样。

12.11.2.2 UART模式

接收控制装置有一个开始位探测电路，用于在检测到正常开始位时启动接收操作。

12.11.3 接收操作

12.11.3.1 接收缓冲器

已接收的数据由接收移位寄存器中的 1 位来储存。在一整套位已经储存时，即生成中断 INTTRX。

当启用双缓冲器时，该数据转到接收缓冲器 (SCxBUF)，且此接收缓冲器已满标志 (SCxMOD2<RBFLL>)被设置到 "1"。读取接收缓冲器将接收缓冲器已满标志清 "0"。对于单缓冲器，该接收缓冲器标志则没有任何值。

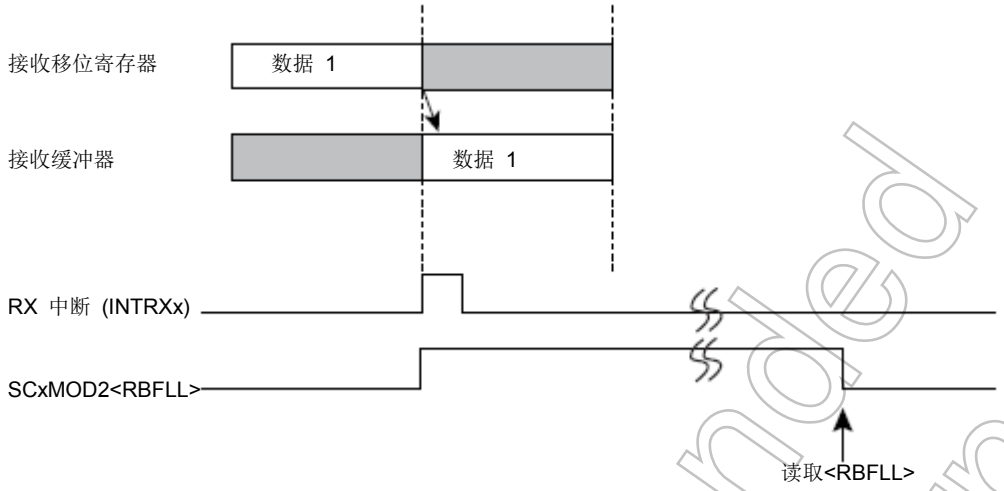


图 12-4 接收缓冲器操作

12.11.3.2 接收FIFO操作

在启用 FIFO 时，已接收数据从接收缓冲器转到 RX FIFO，且接收缓冲器已满标志被立即清除。中断则将根据 SCxRFC<RIL>设置生成。

注:**在以 UART 模式下通过使用 FIFO 接收到带有奇偶检验位的数据时，奇偶错误标记则显示接收数据中发生奇偶检验错误。**

半双工接收模式中的配置与操作说明如下。

SCxMOD1[6:5]=01	:传输模式被设置到半双工模式。
SCxFCNF[4:0]=10111	:达到空满度后自动阻止连续接收。
	:RX FIFO 中使用的位数与中断生成空满度相同。
SCxRFC[1:0]=00	:FIFO 的空满度，其中，生成的接收中断设为 4-字节。
SCxRFC[7:6]=11	:清除 RX FIFO 并设置中断生成的条件。

在设置以上 FIFO 配置后，将 “1“写入到 SCxMOD0<RXE>即开始数据接收。在所有数据被储存到接收移位寄存器，接收缓冲器与 RX FIFO 时，SCxMOD0<RXE>被自动清除，接收操作即告完成。

在上述条件中，如果在达到空满度后启用连续接收，通过读取 FIFO 中的数据就能连续接收数据。

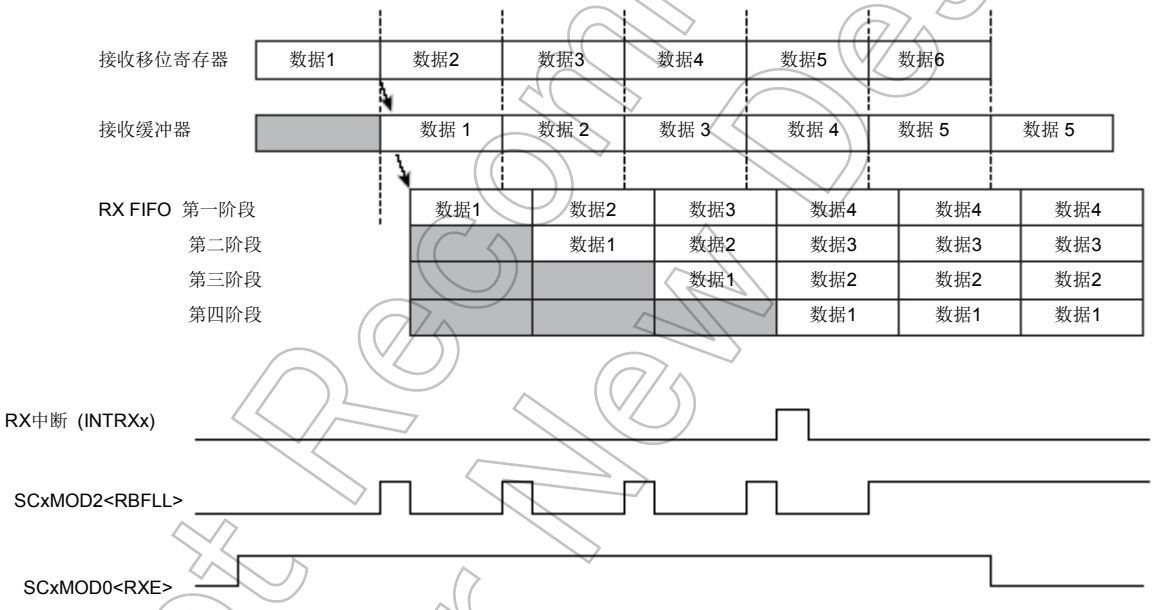


图 12-5 RX FIFO 操作

12.11.3.3 带SCLK输出的 I/O接口模式

在 I/O 接口模式与 SCLK 输出设置中，当所有接收的数据都储存在接收缓冲器与 FIFO 中时，SCLK 输出即停止。因而，在该模式下，超载错误标志没有意义。

SCLK 输出停止与重新输出的时间取决于接收缓冲器与 FIFO。

(1) 单缓冲器情况

接收一个数据后即停止 SCLK 输出。在该模式中，I/O 接口可通过握手与传输装置传输各个数据。

当读取某个缓冲器中的数据时，SCLK 输出则重新开始。

(2) 双缓冲器

在将数据接收到接收移位寄存器与接收缓冲器中之后即停止 SCLK 输出。

当读取该数据时，SCLK 输出重新开始。

(3) FIFO

在将数据接收到某个移位寄存器，接收缓冲器与 FIFO 之后即停止 SCLK 输出。

当读取一个字节数据时，接收缓冲器中的数据被传输到 FIFO，接收移位寄存器中的数据则被传输到接收缓冲器，SCLK 输出重新开始。

如果将 SCxFCNF<RXTXCNT>设置到 “1“，随着 SCxMOD0<RXE>位的清除，SCLK 停止，接收操作也停止。

12.11.3.4 读取接收数据

尽管启用或禁用 FIFO，但仍从接收缓冲器 (SCxBUF)中读取已接收的数据。

当 RX FIFO 禁用时，缓冲器已满标志 SCxMOD2<RBFLL>由该读取清 “0“。在这种情况下，在从接收缓冲器读取某个数据之前，接收移位寄存器中则可接收下一个数据。在 8-位 UART 模式中要添加的奇偶检验位以及 9-位 UART 模式中最高有效位将被储存在 SCxCR<RB8>中。

当 RX FIFO 可用时，9-位 UART 模式则被禁止，因为多达 8-位数据可以储存在 FIFO 中。在 8-位 UART 模式中，奇偶检验位丢失，但确定了奇偶校验错误，并将结果储存在 SCxCR<PERR>中。

12.11.3.5唤醒功能

在 9-位 UART 模式中，从控制器可通过将唤醒功能 SCxMOD0<WU>设置到 “1“，以唤醒模式操作。在这种情况下，只有当 SCxCR<RB8>设置到 “1“时，中断 INTRXx 才会生成。

12.11.3.6 超载错误

当禁用 FIFO 时，该超载错误发生，并且在接收下一个数据之前，一个超载错误没有完成读取数据。当一个超载错误时，接收缓冲器与 SCxCR<RB8>的内容没有丢失，但接收移位寄存器的内容丢失了。

当启用 FIFO 时发生超载错误，在 FIFO 已满时，通过在将下一个数据转入接收缓冲器之前不读取数据来设置超载标志。在这种情况下 FIFO 的内容不会丢失。

在 I/O 接口模式中进行 SCLK 输出设置，时钟输出自动地停止，因此，该标志没有意义。

注:**当该模式从 I/O 接口 SCLK 输出模式变成另一个模式时，读取 SCxCR 并清除超载标志**。

12.12 传输

12.12.1 传输计数器

此传输计数器为 4-位二进制计数器，通过 SIOCLK 计数，如接收计数器的情况一样。在 UART 模式中，其在每第 16 个时钟脉冲时生成传输时钟 (TXDCLK)。

图 12-6 UART 模式中传输时钟脉冲的生成

12.12.2 传输控制

12.12.2.1 I/O接口模式

在 SCLK 输出模式中将 SCxCR<IOC>设置到 "0"，传输缓冲器中数据的各个位被输出 ~ 从 SCLK 引脚输出的移位时钟下降沿上的 TXD 引脚。

在 SCLK 输入模式中将 SCxCR<IOC>设置到 "1"，传输缓冲器中数据的各个位输出 ~ 根据 SCxCR<SCLKS>设置的 SCLK 输入信号上升沿或下降沿上的 TXD 引脚。

12.12.2.2 UART模式

当把传输数据写入传输缓冲器时，数据传输在下一个 TXDCLK 的上升沿上启动，同时还生成传输移位时钟信号。

12.12.3 传输操作

12.12.3.1 传输缓冲存储器的运行

当禁用双缓冲器时，CPU 仅将数据写入传输转换缓冲器，数据传输完成即生成传输中断 INTTXx。

如果启用双缓冲器 (包括启用传输 FIFO)，写入传输缓冲器的日期被转到传输移位寄存器。同时生成 INTTXx 中断，传输缓冲器已空标志 (SCxMOD2<TBEMP>)被设置到 “1“。此标志表示下一个传输数据可以写入。当将下一个数据写入传输缓冲器时，该<TBEMP>标志则被清 “0“。

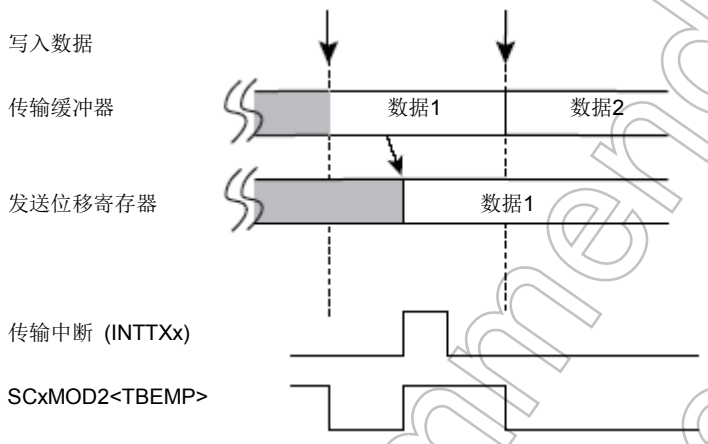


图 12-7 传输缓冲器的操作 (双缓冲器启用)

12.12.3.2 传输FIFO操作

当启用 FIFO 时，使用传输缓冲器与 FIFO 可储存最多 5-字节数据。一旦传输启用，数据则从传输缓冲器被传输到传输移位寄存器，开始传输。如果 FIFO 中存在数据，该数据立即转到传输缓冲器，且<TBEMP>标志被清 “0“。

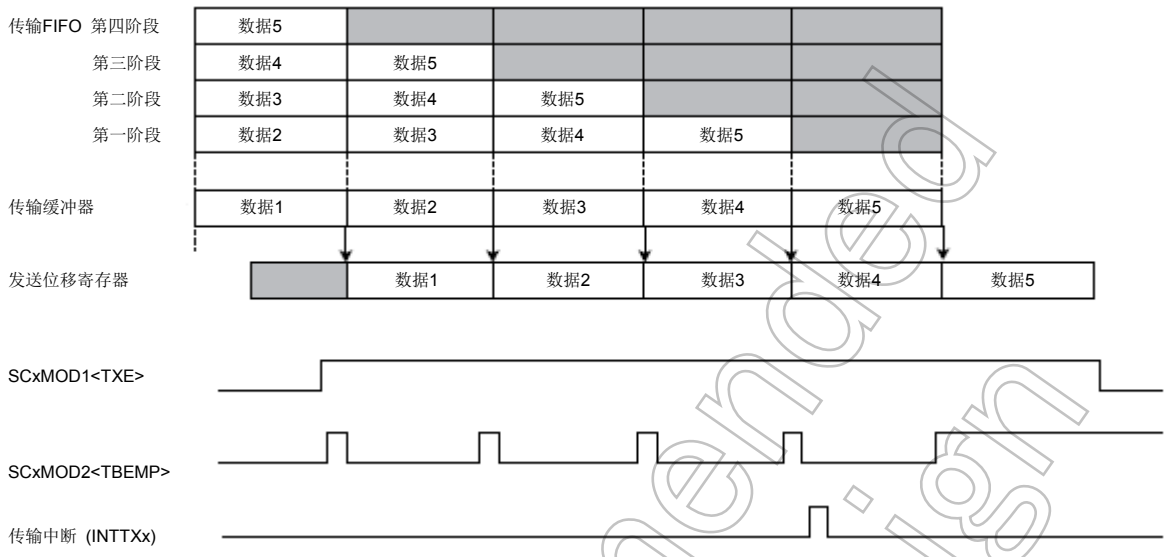
注:**使用 TX FIFO 缓冲器时，必须在设置 SIO 传输模式 (半双工 / 全双工)与启用 FIFO (SCxFCNF<CNFG>= “1“)之后，清除 TX FIFO。**

通过将传输模式设置到半双工来传输 4-字节数据流的设置与操作显示如下。

```
SCxMOD1[6:5] =10     :传输模式被设置到半双工。
SCxFCNF[4:0] = 11011 :如果 FIFO 为空，传输自动禁用。
                     :用于 RX FIFO 的位数与中断的相同
                     :生成空满度
SCxTFC[1:0] = 00     :将中断生成空满度设置到 “0“。
SCxTFC[7:6] = 11     :清除 RX FIFO 并设置中断生成的条件。
SCxFCNF[0] = 1       :启用 FIFO
```

在配置以上设置后，将 5 字节数据写入传输缓冲器或 FIFO，并将 SCxMOD1<TXE>位设置到 “1“，即可启动数据传输。当上一个传输数据被转到传输缓冲器时，即生成传输 FIFO 中断。当上一个数据传输完成时，时钟被停止，传输顺序终止。

一旦配置以上设置，如果该传输没有设置成自动禁用，那么，该传输应持续写入传输数据。



12.12.3.3 I/O接口模式/通过SCLK 输出传输

如果 SCLK 被设置到以 I/O 接口模式生成时钟脉冲，当所有数据传输完成时，SCLK 输出自动停止，并且不会出现欠载错误。

SCLK 输出的挂起与恢复时间视缓冲器与 FIFO 用途而不同。

(1) 单缓冲器

每次传输一个帧数据时 SCLK 输出即停止。每个数据与通信另一方握手即可启用。当把下一个数据写入缓冲器时，SCLK 输出即恢复。

(2) 双缓冲器

传输移位寄存器与传输缓冲器的数据传输一完成，SCLK 输出即停止。当把下一个数据写入缓冲器时，该 SCLK 输出即恢复。

(3) FIFO

储存在传输移位寄存器，传输缓冲器与 FIFO 中的所有数据传输完成，SCLK 输出即停止。写入下一个数据，SCLK 输出继续随即恢复。

如果配置了 SCxFCNF<RXTXCNT>，在 SCLK 停止与传输停止的同时，SCxMOD0<TXE>被清除。

12.12.3.4 欠载运行错误

如果传输 FIFO 在 I/O 接口 SCLK 输入模式中禁用，当在下一个帧时钟输入之前传输缓冲器中尚无数据设置时，这种情况一旦从传输移位寄存器的数据传输完成即发生，欠载错误就会发生，SCxCR<PERR>被设置到 “1“。

在 I/O 接口模式中进行 SCLK 输出设置，时钟输出自动地停止，因此，该标志没有意义。

注:**在将 I/O 接口 SCLK 输出模式切换到其它模式之前，读取 SCxCR 寄存器并清除欠载标志。**

12.13 握手功能

握手功能就是利用 CTS (清除发送)引脚启用逐帧数据传输，防止超载错误。可通过 SCxMOD0<CTSE>启用或禁用该功能。

当 $\overline{\text{CTS}}$ 引脚被设置到 “高“电平时，可完成当前的数据传输，但下一个数据传输即挂起，直到 $\overline{\text{CTS}}$ 引脚返回 “低“电平。然而，在这种情况下，在正常定时中 INTTXx 中断生成，下一个传输数据则被写入传输缓冲器，并等待，直到准备好传输数据。

注 1:**如果在传输期间 $\overline{\text{CTS}}$ 信号设置到 “高“，下一个数据传输则在当前传输完成后被挂起。**

注 2:**在 $\overline{\text{CTS}}$ 被设置到 “低“ 之后，数据传输则在 TXDCLK 时钟脉冲的第一个下降沿启动。**

虽然并未配置 $\overline{\text{RTS}}$ 引脚，但给 RTS 功能分配一个端口位，即可容易地执行握手控制功能。数据接收一完成就将该端口设置“高“ (在接收中断例行程序中)，即可请求传输侧挂起数据传输。

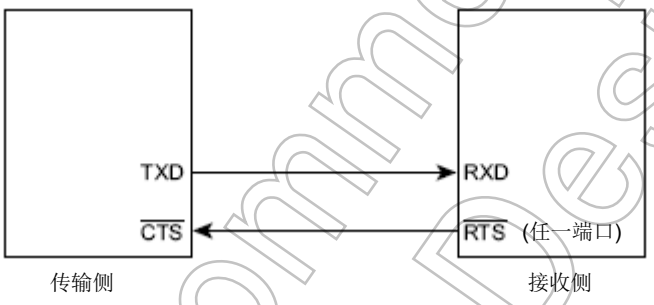


图12-8 握手功能

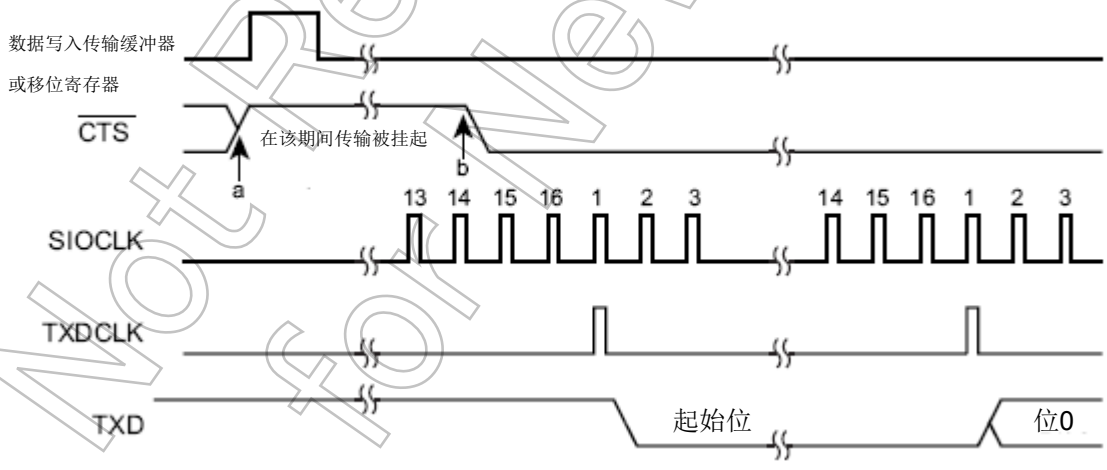


图12-9 $\overline{\text{CTS}}$ 信号计时

12.14 中断/错误生成时间

12.14.1 接收中断

图 12-10 显示了接收操作数据流程与读取路由。

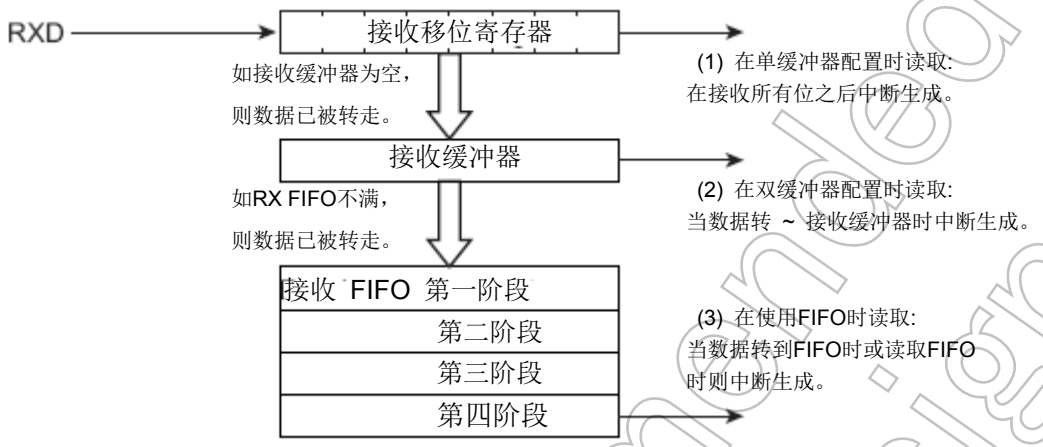


图 12-10 接收缓冲器 / FIFO 配置示意图

12.14.1.1 单缓冲器 / 双缓冲器

RX 中断生成的时间取决于传输模式与缓冲器配置，具体如下。

缓冲器配置	UART模式	I/O接口模式
单缓冲器	-	紧接上一个SCLK的上升 / 下降沿之后 (上升或下降根据SCxCR<SCLKS>设置确定)
双缓冲器	围绕第一个停止位的中心	紧接上一个SCLK的上升 / 下降沿之后 (上升或下降根据SCxCR<SCLKS>设置确定)一旦读取缓冲器，数据即从移位寄存器传输 ~ 缓冲器。

注:**发生超载错误时中断不生成。**

12.14.1.2 FIFO

使用 FIFO 时生成接收中断，条件是，以下任何操作或 SCxRFC<RFIS>设置已确定。

一个帧的所有位接收完成

读取 FIFO

中断条件由 SCxRFC<RFIS>设置决定，如表 12-13 所述。

表 12-13 使用 FIFO 时的接收中断条件.

SCxRFC<RFIS>	中断条件
“0“	“FIFO的空满度等于FIFO中断生成的空满度。
“1“	“FIFO的空满度 “大于或等于 “FIFO中断生成的空满度“

12.14.2 传输中断

图 12-11 显示了传输操作数据流程与读取路由。

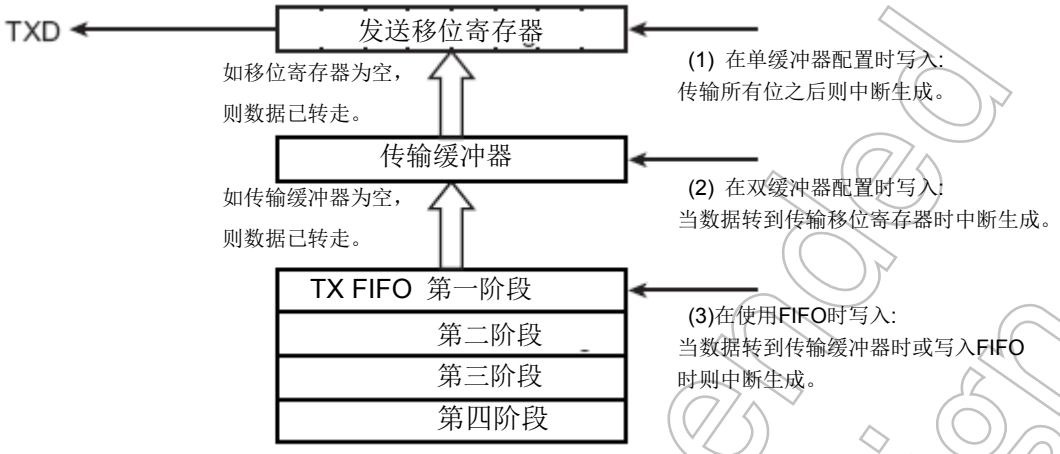


图 12-11 传输缓冲器/FIFO 配置示意图

12.14.2.1单缓冲器/双缓冲器

传输中断生成时间取决于传输模式与缓冲器配置，具体如下。

缓冲器配置	UART模式	I/O接口模式
单缓冲器	停止位传输前	紧接上一个SCLK的上升 / 下降沿之后 (上升或下降根据SCxCR<SCLKS>设置确定)
双缓冲器	当数据从传输缓冲器转到传输移位寄存器时。	

注:**如启用了双缓冲器，当数据从缓冲器转到移位寄存器时，将中断写入缓冲器则生成中断。**

12.14.2.2 FIFO

使用 FIFO 时生成传输中断，条件是，以下任何操作或 SCxTFC<TFIS>设置已确定。

一个帧的所有位均传输完成。

写入 FIFO 中断条件由 SCxTFC<TFIS>设置决定，如表 12-14 所述。

表 12-14 使用 FIFO 时的传输中断条件.

SCxTFC<TFIS>	中断条件
“0“	“FIFO的空满度等于FIFO中断生成的空满度。“
“1“	“FIFO的空满度 “小于或等于“FIFO中断生成的空满度“

12.14.3 错误生成

12.14.3.1 UART模式

模式	9位	7位 8位 7位+奇偶校验位 8位+奇偶校验位
成帧错误 超载错误	停止位中心周围	
奇偶校验错误	-	奇偶校验位中心周围

12.14.3.2 I/O接口模式

超载错误	在最后一个SCLK的上升 / 下降沿之后立即出现。 (上升或下降是根据SCxCR<SCLKS>设置而定。)
欠载错误	在下一个SCLK的上升或下降沿之后立即出现。 (上升或下降是根据SCxCR<SCLKS>设置而定。)

注:**超载错误和欠载错误在 SCLK 输出模式下并无任何含义。**

12.15 软件复位

将 SCxMOD2<SWRST[1:0]>写为 “10“，再跟着写为 “01“即生成软件复位。结果是，SCxMOD0 <RXE>，SCxMOD1 <TXE>，SCxMOD2 <TBEMP><RBFLL><TXRUN>，SCxCR<OERR><PERR> <FERR>初始化。接收电路，传输电路和 FIFO 变为初始状态。其它状态保持不变。

12.16 各模式下的操作

12.16.1 模式0 (I/O接口模式)

模式 0 包含两种模式，即 SCLK 输出模式用于输出同步时钟信息，和 SCLK 输入模式，用于从外源接收同步时钟信息。

在 FIFO 禁用时的操作性说明如下。关于 FIFO 的详细操作，见前面所述接收/发送 FIFO 功能。

12.16.1.1传输数据

(1)SCLK 输出模式

当禁用传输双缓冲区时 (SCxMOD2<WBUF> = "0")

数据从 TXD 引脚输出，每当 CPU 向传输缓冲区写入数据时，时钟自 SCLK 引脚输出。当所有数据都输出时，会生成以中断 (INTTXx)。

当启用传输缓冲区时 (SCxMOD2<WBUF> = "1")

当 CPU 写入数据到传输缓冲器时，在数据发送中止的同时，或者，在数据从传输缓冲器(移位寄存器)传输完成时，数据从传输缓冲区移动到传输移位寄存器。同时，缓冲区空置标志 SCxMOD2<TBEMP>设置到 "1"，INTTXx 中断生成。

当数据从发送缓冲区移动到传输移位寄存器，若发送缓冲区并没有数据可移动到传输移位寄存器，则不会产生 INTTXx 中断，SCLK 输出停止。

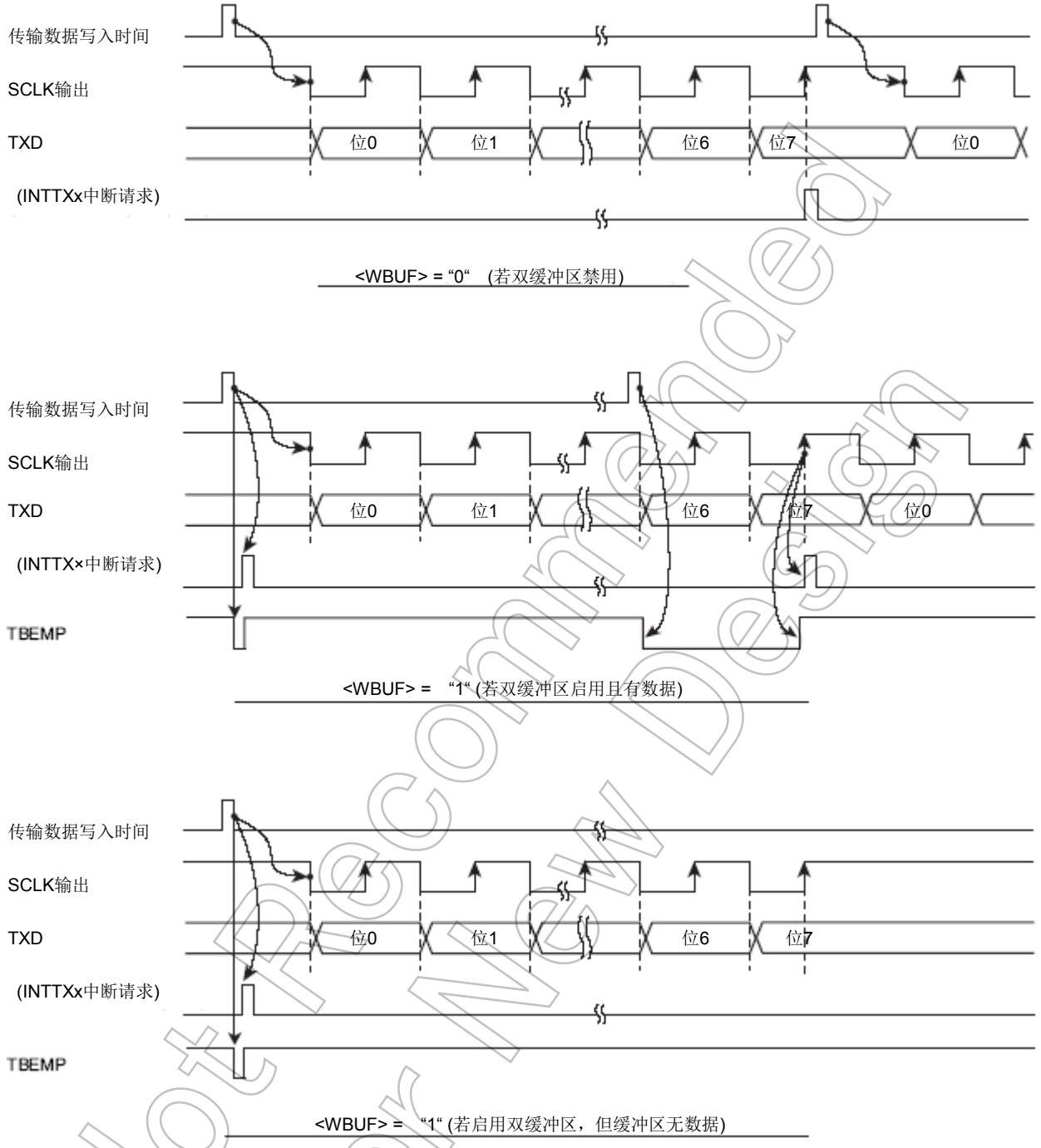


图 12-12 I/O 接口模式 (SCLK 输出模式)下的传输操作

(2)SCLK 输入模式

当禁用双缓冲区 (SCxMOD2<WBUF> = “0“)时

若在传输缓冲区写入数据的情况下输入 SCLK，8 位数据则从 TXD 引脚输出。当所有数据都输出时，产生 INTTXx 中断。下一个传输数据必须在图 12-13 所示的时间点 “A“之前写入。

当启用双缓冲区 (SCxMOD2<WBUF> = “1“)时

在 SCLK 输入被激活前 CPU 把数据写入到传输缓冲区时，或者，在数据从传输移位寄存器传输完成时，数据则从传输 缓冲区移动到传输移位寄存器。同时，把传输缓冲区的空置标志 SCxMOD2<TBEMP>设置到 “1“ ，INTTXx 中断生成。

若 SCLK 输入激活，但传输缓冲区无数据，即使启动内部位元计数器，也会出现欠载错误，发送的是 8-位虚拟数据 (0xFF)。

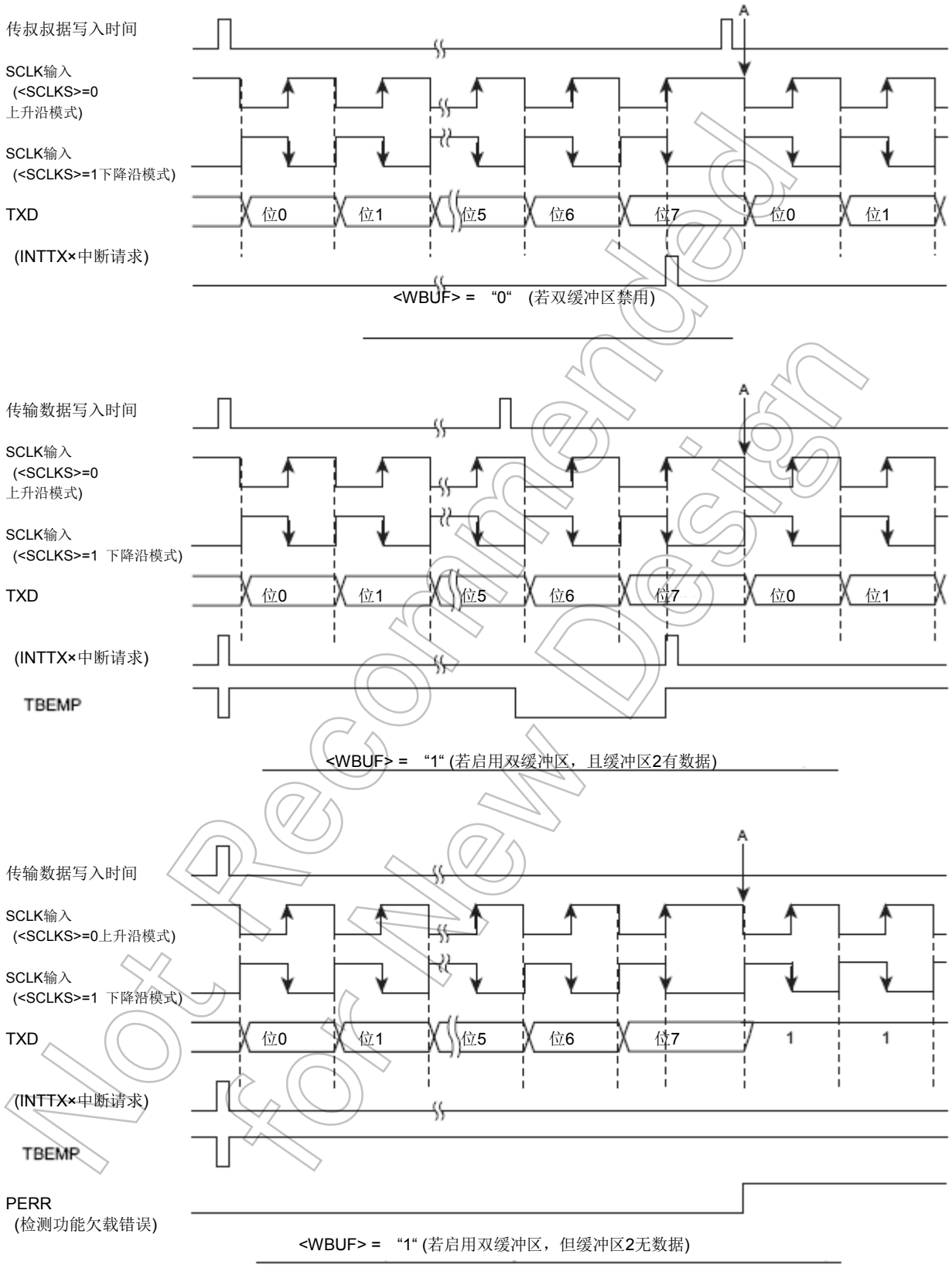


图 12-13 I/O 接口模式 (SCLK 输入模式)下的传输操作

12.16.1.2 接收

(1)SCLK 输出模式

将接收启用位 SCxMOD0<RXE>设置到 “1“，可启动 SCLK 输出。

当禁用双缓冲区 (SCxMOD2<WBUF> = “0“)时

每当 CPU 读取接收的数据，时钟脉冲则从 SCLK 引脚输出，且下一个数据会存储到移位寄存器。当 8 位都收到时，INTRXx 中断即生成。

当启用双缓冲区 (SCxMOD2<WBUF> = “1“)时

存储在移位寄存器的数据移动到接收缓冲区，接收缓冲区方可够接收下一帧。当某一数据从移位寄存器移动到接收缓冲区时，接收缓冲区的全满标志 SCxMOD2<RBFLL>设置到 “1“，即生成 INTRXx 中断。

当数据在接收缓冲区时，若在完整接收下一个 8 位之前无法从接收缓冲区读取数据，则不会产生 INTRXx 中断，SCLK 输出停止。在这种状态下，从接收缓冲区读取数据，可以使移位寄存器中的数据移动到接收缓冲区，从而产生 INTRXx 中断，数据接收重新开始。

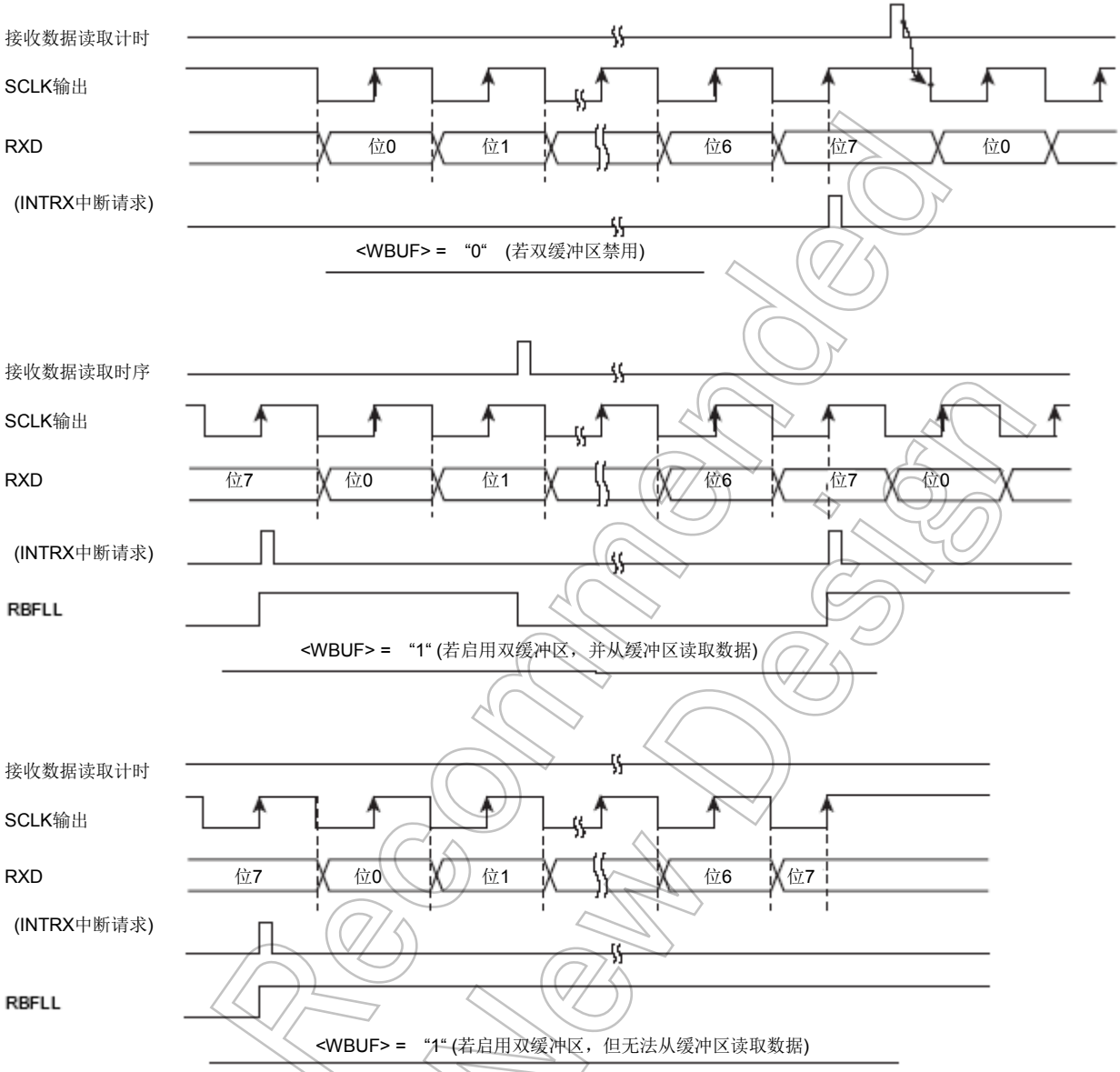


图 12-14 I/O 接口模式 (SCLK 输出模式)下的接收操作

(2) SCLK 输入模式

在 SCLK 输入模式下，始终启用接收双缓冲区，接收到的帧则可从移位寄存器移动到接收缓冲区，接收缓冲区可依次接收下一帧。

每当接收到的数据移动到接收缓冲区，会生成 INTRXx 接收中断。

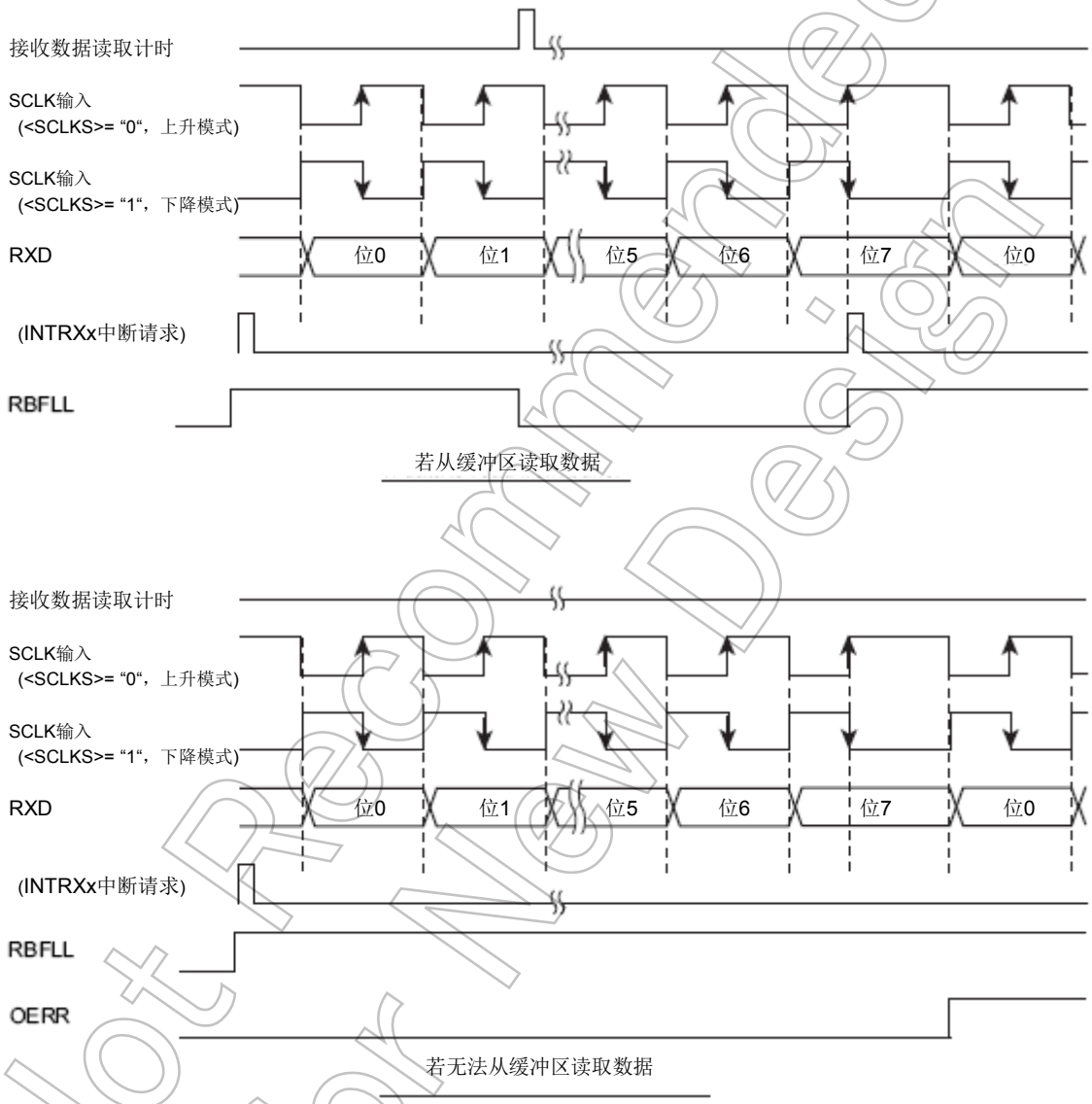


图 12-15:I/O 接口模式 (SCLK 输入模式)下的接收操作

12.16.1.3传输与接收 (全双工)

(1)SCLK 输出模式

当把 SCxMOD2<WBUF>设置 “0“，并禁用双缓冲区时

当 CPU 把数据写入到传输缓冲区，SCLK 就会输出。

随后，8 位数据转移到接收缓冲区，生成 INTRXx 接收中断。同时，写入到传输缓冲区的 8 位数据从 TXD 引脚输出，当所有数据传输完成时，会生成 INTRXx 传输 中断。然后，SCLK 输出停止。

从接收缓冲区读取数据，且 CPU 把下一个传输数据写入到传输缓冲区时，开始下一轮的数据传输与接收。读取接收缓冲区与写入到传输缓冲区的顺序可自定。只有这两种条件均满足时，数据传输才会重新开始。

当把 SCxMOD2<WBUF>设置为 “1“，并启用双缓冲区时

当 CPU 把数据写入到传输缓冲区，SCLK 就会输出。

8 位数据转移到接收移位寄存器，移动到接收缓冲区，生成 INTRXx 中断。在接收 8 位数据的同时，8 位传输数据则从 TXD 引脚输出。当传输完所有数据时，会生成 INTTXx 中断，而下一个数据会从传输 缓冲区移动到传输移位寄存器。

若传输缓冲区并无数据可移动到传输缓冲区 (SCxMOD2<TBEMP> = 1)，或者，当接收缓冲区全满 (SCxMOD2<RBFULL> = 1)时，SCLK 输出停止。当读取接收数据和写入传输数据这两种条件均满足时，SCLK 输出重新开始，并开始下一轮的数据传输与接收。

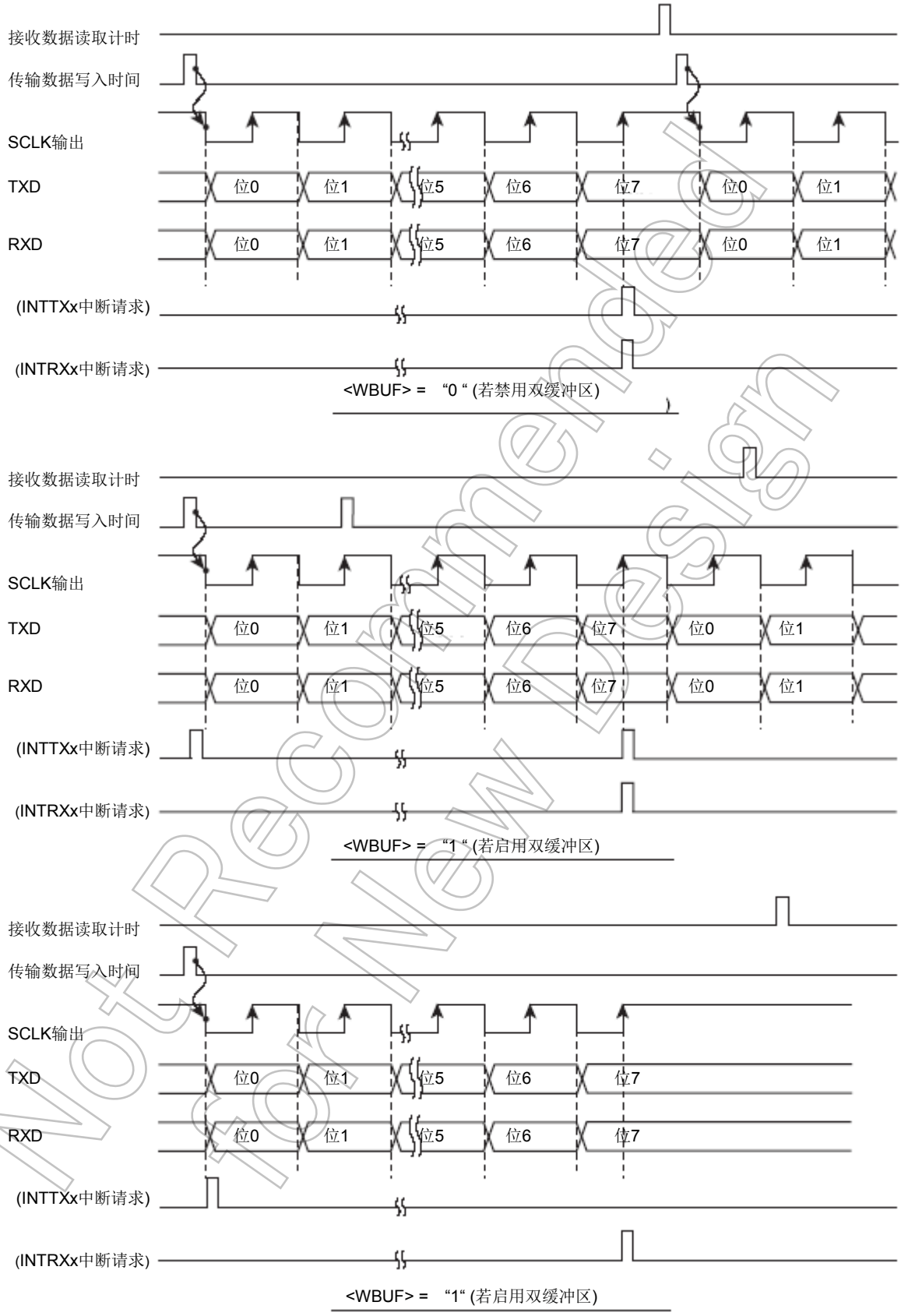


图 12-16　I/O 接口模式 (SCLK 输出模式)下的传输 / 接收操作

(2) SCLK 输入模式

当把 SCxMOD2<WBUF>设置到 “0“，并禁用传输双缓冲区时

当接收数据时，始终启用双缓冲区，无需考虑 SCxMOD2 <WBUF>的设置。

当 SCLK 输入激活时，写入传输缓冲区的 8 位数据就会从 TXD 引脚输出，且 8 位数据就会转移到接收缓冲区。数据传输完成后，生成 INTTXx 中断。数据接收完成后，当数据从移位寄存器移动到接收缓冲区时，生成 INTTRXx 中断。

注意传输数据必须在 SCLK 输入到下一帧之前写入到传输缓冲区 (必须在图 10-17 的点 A 之前写入数据)。必须在完整接收下一帧数据之前读取数据。

当把 SCxMOD2<WBUF>设置到 “1“，并启用双缓冲区时

数据从传输移位寄存器传输完成后，当传输缓冲区数据移动到传输移位寄存器时，生成 INTRXx 中断。同时，接收到的数据会转移到移位寄存器，移动到接收缓冲区，生成 INTRXx 中断。

注意传输数据必须在 SCLK 输入到下一帧之前写入传输缓冲区 (必须在图 12-17 的点 A 之前写入数据)。必须在完整接收下一帧数据之前读取数据。

SCLK 输入到下一帧后，开始从传输移位寄存器传输 (已从传输缓冲区移动数据)，同时把接收数据转移到接收移位寄存器。

若接收缓冲区的数据尚未在接收帧的最后一位时读取，会出现超载错误。同样地，若 SCLK 输入到下一帧时并无数据写入传输缓冲区，会出现欠载错误。

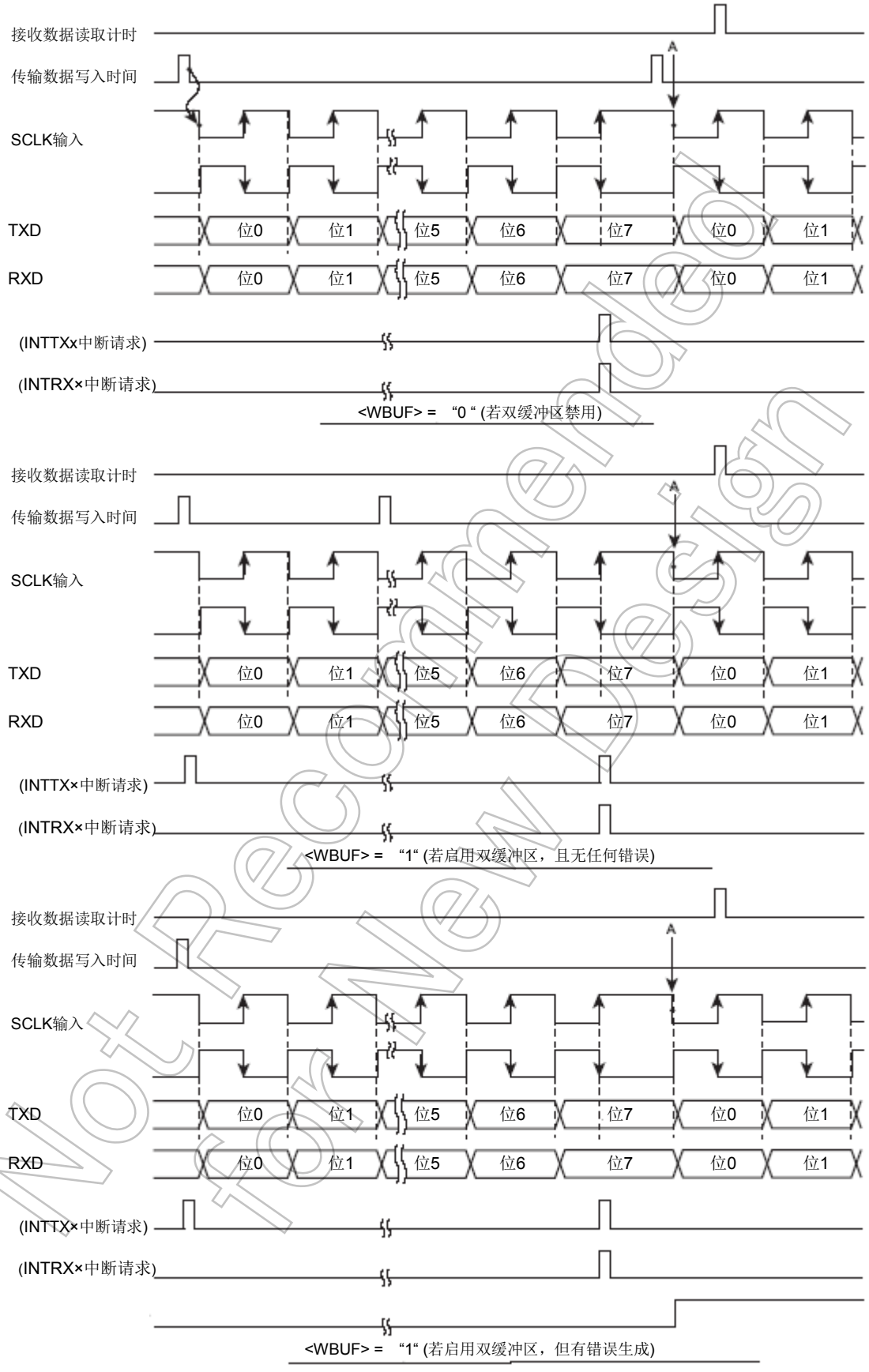


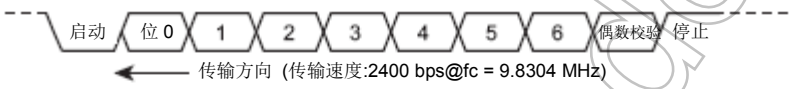
图 12-17 I/O 接口模式 (SCLK 输入模式)下的传输 / 接收操作

12.16.2 模式1 (7-位UART模式)

把串行模式控制寄存器 (SCxMOD<SM[1:0]>)设置到 “01“，可选择 7-位 UART 模式。

在这种模式下，可把奇偶校验位加到传输数据流；串行模式控制寄存器 (SCxCR<PE>)会控制奇偶校验的启用/禁用设置。把<PE>设置为 “1“ (启用)时，可利用 SCxCR<EVEN>位选择偶数或奇数校验。利用 SCxMOD2<SBLEN>可指定停止位的长度。

下表所示的是按以下格式传输数据的控制寄存器设置。



传输方向 (传输速度:2400 bps@fc = 9.8304 MHz)

时钟条件	系统时钟 :	高速 (fc)
	高速时钟齿轮 :	x1 (fx)
	预分频时钟 :	fperiph/2 (fperiph = fsys)

		7	6	5	4	3	2	1	0	
SCxMOD0	←	x	0	-	0	0	1	0	1	设置7-位UART模式
SCxCR	←	x	1	1	x	x	x	0	0	启用偶数校验
SCxBRCR	←	0	0	1	0	0	1	0	0	设置2400bps
SCxBUF	←	*	*	*	*	*	*	*	*	设置传输 数据

x : 忽略- : 无变化

12.16.3 模式2 (8-位UART模式)

把 SCxMOD0<SM[1:0]>设置为 “10“，可选择 8-位 UART 模式。在这种模式下，可增加奇偶校验位，并利用 SCxCR<PE>控制奇偶校验的启用/禁用。若<PE> = “1“ (启用)，可利用 SCxCR<EVEN>选择偶数或奇数校验。

按以下格式接收数据的控制寄存器设置如下:

传输方向 (传输速度:9600bps@fc=9.8304MHz)

时钟条件	系统时钟:	高速 (fc)
	高速时钟齿轮:	x1 (fx)
	预分频时钟:	fperiph/2 (fperiph = fsys)

		7	6	5	4	3	2	1	0	
SCxMOD0	←	x	0	0	0	1	0	0	1	设置8-位UART模式
SCxCR	←	x	0	1	x	x	x	0	0	启用偶数校验
SCxBRCR	←	0	0	0	1	0	1	0	0	设置9600bps
SCxMOD0	←	-	-	1	-	-	-	-	-	接收使能

x：忽略-：无变化

12.16.4 模式3 (9-位UART模式)

把 SCxMOD0<SM[1:0]>设置为 “11“，可选择 9-位 UART 模式。在这种模式下，必须禁用奇偶校验位 (SCxCR<PE> = “0“)。

把最高有效位 (第 9 位)写入串行模式控制寄存器 0 (SCxMOD0)的位 7<TB8>，以便传输数据。数据存储于串行控制寄存器 SCxCR 的位 7<RB8>。

把数据写入到缓冲区或从缓冲区读取数据时，最高有效位必须在写入 SCxBUF 或从 SCxBUF 读取之前写入或读取。利用 SCxMOD2<SBLEN>可规定的停止位的长度。

12.16.4.1 唤醒功能

在 9-位 UART 模式下，把唤醒功能控制位 SCxMOD0<WU>设置为 “1“，可在唤醒模式下运行从属控制器。

在这种情况下，只有把 SCxCR<RB8>设置为 “1“才会生成 INTRXx 中断。

注:**从控制器的 TXD 引脚必须利用 ODE 寄存器设置为开漏输出模式。**

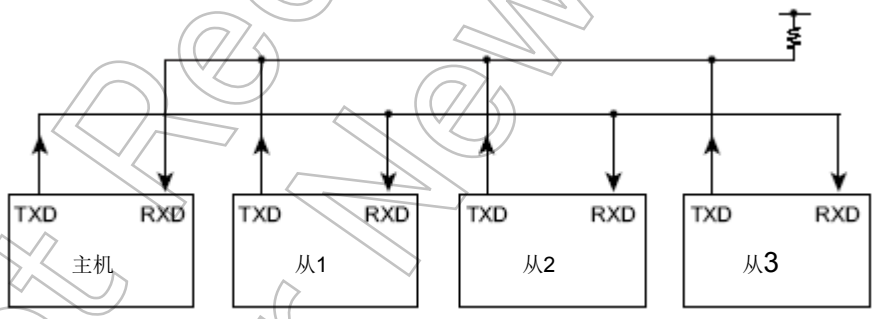
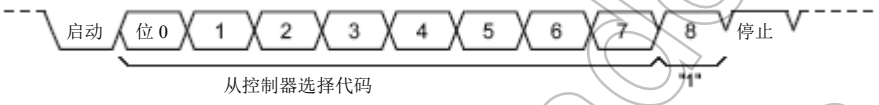


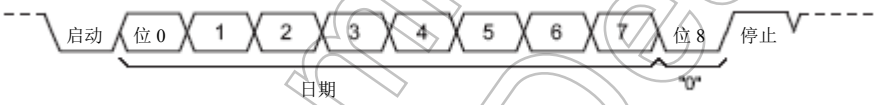
图 12-18 串行链接使用唤醒功能

12.16.4.2 协议

1. 选择针对主机和从控制器的 9-位 UART 模式。
2. 把 SCxMOD<WU>设置到 “1“，使从控制器准备接收数据。
3. 主机用于传输单一帧的数据 (包括从控制器选择代码 (8 位))。在此模式下，必须把最高有效位 (位 8)<TB8>设置到 “1“。



4. 各从控制器接收上述数据帧；若接收到的代码与控制器本身的选择代码相一致，<WU>位被清除 到 “0“。
5. 主机将数据传输到指定的从控制器 (该控制器的 SCxMOD<WU>位被清 “0“)。此模式下，必须把最高有效位 (位 8)<TB8>设置到 “0“。



6. <WU>位被设置到 “1“的从控制器会忽略接收数据，因为最高有效位 (位 8)<RB8>被设置到 “0“，因而没有生成中断 (INTRXx)。同时，<WU>位被设置到 “0“的从控制器可把数据传输到主机，提示数据已成功接收。

13. 同步串行端口 (SSP)

13.1 概述

此 LSI 含有 SSP (同步串行端口)及 1 条通道。该通道具有以下特征:

通信协议		包括SPI在内的三种类型的同步串行端口 摩托 SPI(SPI)帧格式 TI同步 (SSI)帧格式 National Microwire (Microwire)帧格式
操作模式		主/从模式
传输 FIFO		16位宽 / 8 层深
接收FIFO		16位宽 / 8层深
传输/接收的数据量		4 ~ 16 位
中断类型		传输中断 接收中断 收到超载中断 超时中断
通信速率	主模式	fsys (64MHz)/ 4 (max. 16Mbps)
	从模式	fsys (64MHz)/ 12 (max. 5.3Mbps)
DMA		受支持
内部测试功能		内部回送测试模式可用。
控制引脚		SPCLK，SPFSS，SPDO，SPDI

13.2 方块图

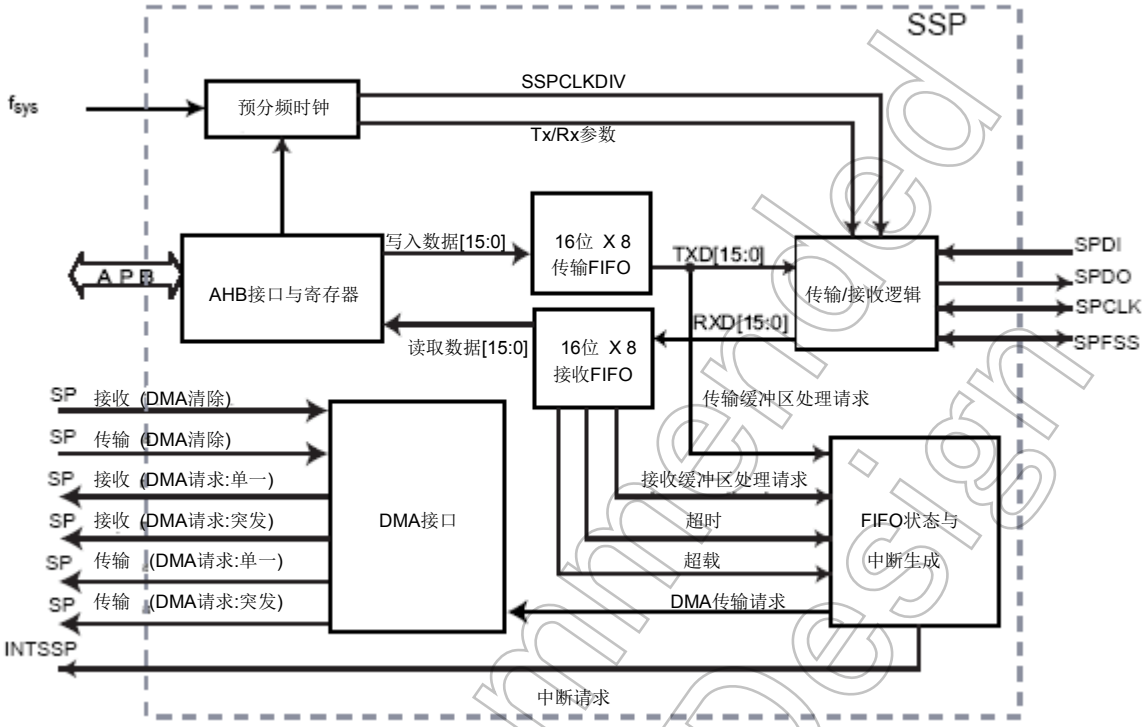


图 13-1 SSP 方块图

13.3 寄存器

13.3.1 寄存器列表

基址= 0x4004_0000

寄存器名称		地址（基+）
控制寄存器0	SSPCR0	0x0000
控制寄存器1	SSPCR1	0x0004
接收FIFO (读取)与传输FIFO (写入)数据寄存器	SSPDR	0x0008
状态寄存器	SSPSR	0x000C
预分频时钟寄存器	SSPCPSR	0x0010
中断启用/禁用寄存器	SSPIMSC	0x0014
启用前的中断状态寄存器	SSPRIS	0x0018
启用后的中断状态寄存器	SSPMIS	0x001C
中断清除寄存器	SSPICR	0x0020
DMA控制寄存器	SSPDMACR	0x0024
保留	-	0x0028 ~ 0x0FFC

注 1: 上表中的寄存器仅允许访问字基 (32 位)。

注 2: 禁止访问 “保留“区域。

13.3.2 SSPCR0 (控制寄存器 0)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	未定义	未定义	未定义	未定义	未定义	未定义	未定义	未定义
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	未定义	未定义	未定义	未定义	未定义	未定义	未定义	未定义
	15	14	13	12	11	10	9	8
比特符号	SCR							
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	SPH	SPO	FRF		DSS			
复位后	0	0	0	0	0	0	0	0

位	比特符号	型号	功能
31-16	-	W	写作 “0“。
15-8	SCR[7:0]	R/W	用于串行时钟频率设置。 参数 : 0x00 ~ 0xFF 生成SSP传输位速率和接收位速率的位。 利用以下等式可得出位速率: 比特率 = fsys / (<CPSDVSR> × (1+ <SCR>)) <CPSDVSR>是2 ~ 254之间的偶数，由SSPCPSR寄存器进行编程，<SCR>则取0 ~ 225之间的值。
7	SPH	R/W	SPCLK 相: 0: 在第1时钟沿捕捉数据。 1: 在第2时钟沿捕捉数据。 这仅适用于摩托罗拉SPI帧格式。见“摩托罗拉SPI帧格式“所述。
6	SPO	R/W	SPCLK 极性: 0:SPCLK处于低状态。 1:SPCLK处于高状态。 这仅适用于摩托罗拉SPI帧格式。见“摩托罗拉SPI帧格式“所述。
5-4	FRF[1:0]	R/W	帧格式: 00: SPI帧格式 01: SSI串行帧格式 10: Microwire帧格式 11: 保留，操作尚未明确
3-0	DSS[3:0]	R/W	数据量选择: 0000: 保留，未确定的操作 1000: 9位数据 0001: 保留，未确定的操作 1001: 10位数据 0010: 保留，未确定的操作 1010: 11位数据 0011: 4位数据 1011: 12位数据 0100: 5位数据 1100: 13位数据 0101: 6位数据 1101: 14位数据 0110: 7位数据 1110: 15位数据 0111: 8位数据 1111: 16位数据

注:选择从模式时，把预分频时钟设置到 SSPCR0<SCR[7:0]> = 0x00，SSPCPSR<CPSDVSR[7:0]> = 0x02

13.3.3 SSPCR1 (控制寄存器 1)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	未定义	未定义	未定义	未定义	未定义	未定义	未定义	未定义
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	未定义	未定义	未定义	未定义	未定义	未定义	未定义	未定义
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	未定义	未定义	未定义	未定义	未定义	未定义	未定义	未定义
	7	6	5	4	3	2	1	0
比特符号	-	-	-	-	SOD	MS	SSE	LBM
复位后	未定义	未定义	未定义	未定义	0	0	0	0

位	比特符号	型号	功能
31-4	-	W	写作 “0“。
3	SOD	R/W	从模式SPDO输出控制: 0: 启用 1: 禁用 从模式输出禁用。此位元仅在从模式 (<MS>= “1“)下有关联。
2	MS	R/W	主/从模式选择: (注) 0: 配置为主机的装置 1: 配置为从控制器的装置
1	SSE	R/W	SSP启用/禁用 0:禁用 1:启用
0	LBM	R/W	回送模式 0:启用正常串行端口操作。 1:传输 串行移位器输出与接收串行移位器内部相连接。

注:此位元用于主机与从控制器之间的切换。应确保在从模式下按以下步骤进行传输配置:

1) 设置到从模式 :<MS>=1
2) 设置 FIFO 中的传输数据 :<DATA>=0x****
3) 将 SSP 设置为启用 :<SSE>=1

13.3.4 SSPDR (数据寄存器)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	未定义	未定义	未定义	未定义	未定义	未定义	未定义	未定义
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	未定义	未定义	未定义	未定义	未定义	未定义	未定义	未定义
	15	14	13	12	11	10	9	8
比特符号	数据							
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	数据							
复位后	0	0	0	0	0	0	0	0

位	比特符号	型号	功能
31-16	-	W	写作 “0“。
15-0	DATA[15:0]	R/W	传输 /接收FIFO数据: 0x0000 ~ 0xFFFF 读取: 接收 FIFO 写入: 传输 FIFO 若程序使用的数据量小于16位，则把数据写入合适的LSB。传输 控制电路会忽略MSB侧未使用的位元。接收控制电路会自动地把数据接收到合适的LSB。

13.3.5 SSPSR (状态寄存器)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	未定义	未定义	未定义	未定义	未定义	未定义	未定义	未定义
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	未定义	未定义	未定义	未定义	未定义	未定义	未定义	未定义
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	未定义	未定义	未定义	未定义	未定义	未定义	未定义	未定义
	7	6	5	4	3	2	1	0
比特符号	-	-	-	BSY	RFF	RNE	TNF	TFE
复位后 F	未定义	未定义	未定义	0	0	0	1	1

位	比特符号	型号	功能
31-5	-	W	写作 “0“。
4	BSY	R	忙碌标记: 0: 空闲 1: 忙 <BSY>= “1“表示SSP目前正在传输 和/或接收帧，或传输FIFO没有清空。
3	RFF	R	接收FIFO全满标记: 0: 接收FIFO并非全满。 1: 接收FIFO全满。
2	RNE	R	接收FIFO清空标记: 0: 接收FIFO清空。 1: 接收FIFO没有清空。
1	TNF	R	传输 FIFO全满标记: 0: 传输 FIFO全满。 1: 传输 FIFO没有全满。
0	TFE	R	传输 FIFO清空标记: 0: 传输 FIFO没有清空。 1: 传输 FIFO清空。

13.3.6 SSPCPSR (时钟预分频寄存器)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	未定义	未定义	未定义	未定义	未定义	未定义	未定义	未定义
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	未定义	未定义	未定义	未定义	未定义	未定义	未定义	未定义
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	未定义	未定义	未定义	未定义	未定义	未定义	未定义	未定义
	7	6	5	4	3	2	1	0
比特符号	CPSDVSR							
复位后	0	0	0	0	0	0	0	0

位	比特符号	型号	功能
31-8	-	W	写作 “0“。
7-0	CPSDVSR[7:0]	R/W	时钟预分频除数: 设置2 ~ 254之间的偶数。 时钟预分频除数:必须是2 ~ 254之间的偶数，取决于fsys的频率。读取时，最低有效位始终归零。

注:选择从模式时，把时钟预分频设置到 SSPCR0<SCR[7:0]> = 0x00，SSPCPSR<CPSDVSR[7:0]> = 0x02

13.3.7 SSPIMSC (中断启用/禁用寄存器)

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

位	比特符号	型号	功能
31-4	-	W	写作 “0“。
3	TXIM	R/W	启用传输 FIFO中断: 0: 禁用 1: 启用 若传输 FIFO半空或不足半空，则出现启用或禁用条件中断。
2	RXIM	R/W	启用接收FIFO中断: 0: 禁用 1: 启用 若接收FIFO半满或不足半满，则出现启用或禁用条件中断。
1	RTIM	R/W	启用接收超时中断: 0: 禁用 1: 启用 启用或禁用条件中断，表示数据在接收FIFO存在超时，不能读取数据。
0	RORIM	R/W	接收超载中断启用: 0: 禁用 1: 启用 启用或禁用条件中断，表示数据是在接收FIFO处于全满的条件时写入的。

13.3.8 SSPRIS (启用前的中断状态寄存器)

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

位	比特符号	型号	功能
31-4	-	W	写作 “0“。
3	TXRIS	R	启用前的传输中断标记: 0: 中断未出现 1: 中断出现
2	RXRIS	R	启用前的接收中断标记: 0: 中断未出现 1: 中断出现
1	RTRIS	R	启用前的超时中断标记: 0: 中断未出现 1: 中断出现
0	RORRIS	R	启用前的超载中断标记: 0: 中断未出现 1: 中断出现

13.3.9 SSPMIS (启用后的中断状态寄存器)

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

位	比特符号	型号	功能
31-4	-	W	写作 “0“。
3	TXMIS	R	启用后的传输中断标记: 0: 中断未出现 1: 中断出现
2	RXMIS	R	启用后的接收中断标记: 0: 中断未出现 1: 中断出现
1	RTMIS	R	启用后的超时中断标记: 0: 中断未出现 1: 中断出现
0	RORMIS	R	启用后的超载中断标记: 0: 中断未出现 1: 中断出现

13.3.10 SSPICR (中断清除寄存器)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	未定义	未定义	未定义	未定义	未定义	未定义	未定义	未定义
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	未定义	未定义	未定义	未定义	未定义	未定义	未定义	未定义
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	未定义	未定义	未定义	未定义	未定义	未定义	未定义	未定义
	7	6	5	4	3	2	1	0
比特符号	-	-	-	-	-	-	RTIC	RORIC
复位后	未定义	未定义	未定义	未定义	未定义	未定义	未定义	未定义

位	比特符号	型号	功能
31-2	-	W	写作 “0“。
1	RTIC	W	清除超时中断标记: 0:无效 1:清除
0	RORIC	W	清除超载中断标记: 0:无效 1:清除

13.3.11 SSPxDMACR (DMA控制寄存器)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	未定义	未定义	未定义	未定义	未定义	未定义	未定义	未定义
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	未定义	未定义	未定义	未定义	未定义	未定义	未定义	未定义
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	未定义	未定义	未定义	未定义	未定义	未定义	未定义	未定义
	7	6	5	4	3	2	1	0
比特符号	-	-	-	-	-	-	TXDMAE	RXDMAE
复位后	未定义	未定义	未定义	未定义	未定义	未定义	0	0

位	比特符号	型号	功能
31-2	-	W	写作 “0“。
1	TXDMAE	R/W	传输FIFO DMA控制: 0: 禁用 1: 启用
0	RXDMAE	R/W	传输FIFO DMA控制: 0: 禁用 1: 启用

13.4 SSP概述

此 LSI 含有 SSP 及 1 条通道。

SSP 是可以与外设装置进行串行通信的接口，具有三种类型的同步串行接口功能。

SSP 可对从外设装置接收到的数据进行串并行转换。

在传输模式下传输缓冲独立 16 位宽，8 层传输 FIFO 的数据，在接收模式下接收缓冲 16 位宽，8 层接收 FIFO。串行数据通过 SPDO 传输，通过 SPDI 接收。

SSP 带有可编程的预分频器，可利用输入时钟 fsys 生成串行输出时钟 SPCLK。在控制寄存器 SSPCR0 与 SSPCR1 对 SSP 的运行模式，帧格式和数据量进行编程。

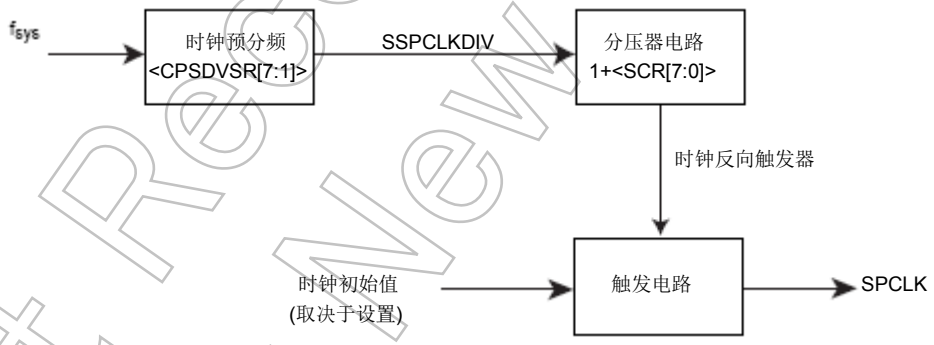
13.4.1 时钟预分频

配置为主机时，可采用含有两个自由运行的串联计数器的时钟预分频提供串行输出时钟 SPCLK。

可通过 SSPCPSR 寄存器对时钟预分频进行编程，分两步用因数 2 ~ 254 除以 fsys。由于没有使用 SSPCPSR 寄存器的最低有效位，因此，不可能被偶数除尽。

用 1 ~ 256 的因数进一步除以预分频器输出值，SSPCR0 寄存器编程的值加 1 即获得该输出值，从而得出主机的输出时钟 SPCLK。

比特率= fsys / (<CPSDVSR>× (1+<SCR>))



13.4.2 传输 FIFO

为一 16-位宽，8-层传输 FIFO 缓冲区，在主模式和从模式下共享。

13.4.3 接收FIFO

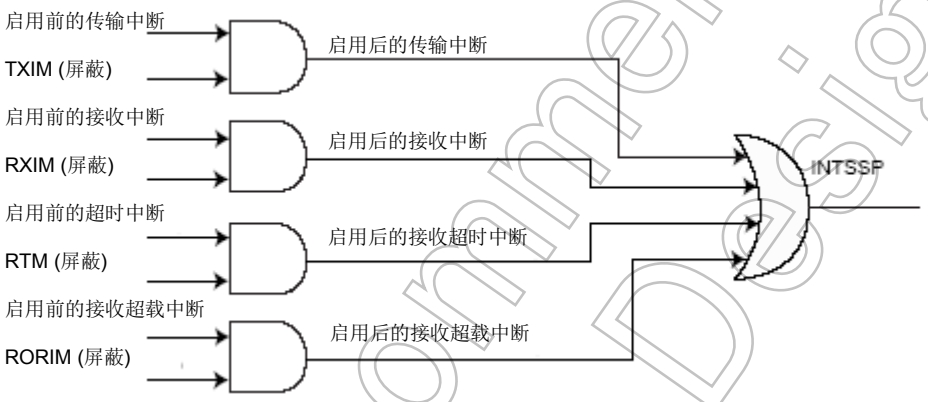
为一 16-位宽，8-层接收 FIFO 缓冲区，在主模式和从模式下共享。

13.4.4 中断生成逻辑电路

生成的每一种中断均可单独屏蔽。

传输中断	当传输 FIFO拥有的可用空间多于或等于全部容量的一半时，会出现一个条件中断。 (传输FIFO中的有效数据项 ≤ 4)
接收中断	当接收FIFO拥有的有效数据多于或等于全部容量的一半时，会出现一个条件中断。 (接收FIFO中的有效数据项 ≥ 4)
超时中断	条件中断，表示数据在接收FIFO时超时。
超载中断	条件中断，表示数据是在全满时写入接收FIFO的。

此外，个别被屏蔽来源可组合为单一中断。若发生上述任何中断，则发生组合中断 INTSSP。



a. 传输中断

若传输 FIFO 中有四条或以下的有效条目，则发生传输中断。

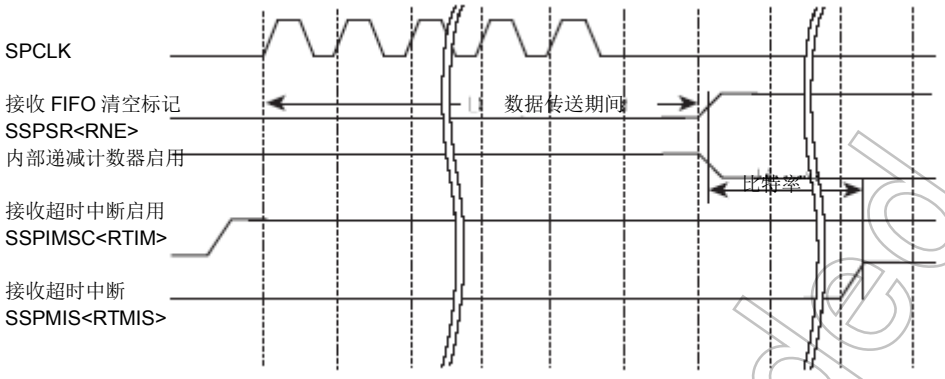
当 SSP 运行禁用时 (SSPCR1 <SSE> = "0")，也会生成传输中断。利用该中断可将第一个传输数据写入 FIFO。

b. 接收中断

当接收 FIFO 中有四条或以上的有效条目，则发生接收中断。

c. 超时中断

当接收 FIFO 未清空，且 SSP 在固定 32-位周期 (比特率)内保持空闲，表明超时中断。此机制确保用户明白数据仍然处于接收 FIFO 状态，有服务请求。这种操作在主模式和从模式下均会出现。当超时中断生成时，所有数据从接收 FIFO 读取。即使所有数据均未读取，也可以传输/接收数据，条件是接收 FIFO 拥有可用空间，且待传输的数据量并未超过接收 FIFO 的可用空间。数据传送一开始，超时中断即被清除。当接收 FIFO 无可用空间时传输/接收数据，超时中断则不会被清除，还会生成超载中断。



d. 超载中断

若在接收 FIFO 已全满时接收下一个数据 (第 9 个数据项)，数据接收后会立即生成超载中断。超载中断生成后接收的数据 (包括第 9 个数据项)失效并作废。不过，若在接收第 9 个数据项的同时 (在中断生成之前)从接收 FIFO 读取数据，第 9 个数据项就会以有效数据被写入接收 FIFO。把“1“写入 SSPICR<RORIC>寄存器，然后从接收 FIFO 读取所有数据，就可以在生成超载中断时合理传送数据。即使所有数据均未读取，也可以传输/接收数据，条件是，接收 FIFO 拥有可用空间，且待传输的数据量并未超过接收 FIFO 的可用空间。注意若没有在清除超载中断后某个 32-位周期 (位速率)内读取接收 FIFO (假设接收 FIFO 没有清空)，则会生成超时中断。

13.4.5 DMA接口

通过 SSPxDMACR 寄存器控制 SSP 的 DMA 运行。

若数据多于接收 FIFO 的水印水平 (FIFO 的一半)，表明有接收 DMA 请求。

若传输 FIFO 的剩余数据量少于水印水平 (FIFO 的一半)，表明有传输 DMA 请求。

提供 DMA 控制器发出的传输 /接收 DMA 请求清除信号的输入引脚，要清除传输/接收 DMA 请求，由 DMA 控制器发出的一个传输/接收 DMA 请求清除信号来执行。

把 DMA 突发长度设置为 4 个字。

注:对于剩下的三个字，SSP 并不会发出突发请求。

在发出相关 DMA 清除信号之前，每一个请求信号都仍然有效。取消请求清除信号后，可再次激活请求信号，取决于上述条件。若 SSP 被禁用或 DMA 启用信号被清除，所有请求信号均被取消。

下表显示了传输 FIFO 与接收 FIFO 的 DMABREQ 触发点:

	突发长度	
水印水平	传输 (清空位置数量)	接收 (满充位置数量)
1/2	4	4

13.5 SSP运行

13.5.1 SSP初始设置

SSP 通讯协议必须在 SSP 关闭时设置。

控制寄存器 SSPCR0 和 SSPCR1 需要 SSP 设置为在下列任何一项协议项下操作的主设备或从设备。此外，还可进行与时钟预分频寄存器 SSPCPSR 和 SSPCR0 <SCR>通讯速度相关的设置。

此 SSP 支持下列协议:

- SPI
- SSI
- 微总线

13.5.2 启用SSP

随着运行启动，传输数据写入 FIFO，并或者随着传输数据写入 FIFO，运行启动，则开始传输。

然而，如果运行启动时，传输 FIFO 中只包含四个或更少输入项目，传输中断。此中断可用于写入初始数据。

注:当 SSP 处于 SPI 从模式且不使用 SPFSS 引脚时，应确保操作启用之前 FIFO 内传输一个或多个字节的数据。如果在传输 FIFO 无效的情况下操作启用，则传输数据不会正确输出。

13.5.3 时钟速率

设置系统时钟脉冲频率时，必须满足下列条件:

在主模式下:

f_{SPCLK} (最大值) → f_{sys} /4

f_{SPCLK} (最小值)→ f_{sys} / (254 × 256)

在从模式下:

f_{SPCLK} (最大值) → f_{sys} /12

f_{SPCLK} (最小值) → f_{sys} / (254×256)

注:主模式下最大波特率等于或小于 16Mbps。

13.6 帧格式

每一个帧格式的宽度在 4 ~ 16 位之间变化，具体取决于编程数据量，且帧格式将从 MSB 开始传输。

串行时钟 (SPCLK)

信号处于 SSI 格式和微总线帧格式的 “低“电平，并处于 SPI 格式下的非激活状态 (同时 SSP 在空闲状态下)。此外，数据只有在传输期间，才能以设定的位速率输出。

串行帧 (SPFSS)

在 SPI 和微总线帧格式中，信号被设置到 “低“ 活跃状态，且在帧传输过程中总是表现在 “低“电平。

在 SSI 帧格式中，在各帧传输前信号仅在 1 位速率期间出现。在此帧格式下，输出数据在 SPCLK 上升沿传输，而在 SPCLK 的下降沿接收数据。

各帧格式详情见第 13.6.1 ~ 13.6.3 节。

13.6.1 SSI帧格式

此模式下，SSP 处于空闲状态，SPCLK 与 SPFSS 被强制设为 “低“，传输数据行 SPDO 变为 Hi-Z。当数据写入传输 FIFO 时，主设备将 1 个 SPCLK 的 “高“ 脉冲输出到 SPFSS 行。所传输的数据将从传输 FIFO 传输到传输串行移位寄存器。4 ~ 16 位数据将在 SPCLK 下一个上升沿从 SPDO 引脚输出。

同样，接收的数据将在 SPCLK 下降沿从 MSB 开始输入到 SPDI 引脚。接收的数据将在相关 LSB 数据锁存后，在 SPCLK 上升沿从串行移位寄存器转入接收 FIFO 中。

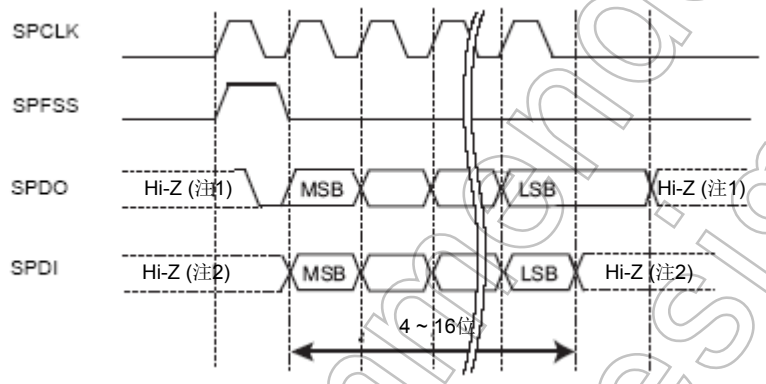


图13-2 SSI帧格式 (单次传输期间的传输/接收)

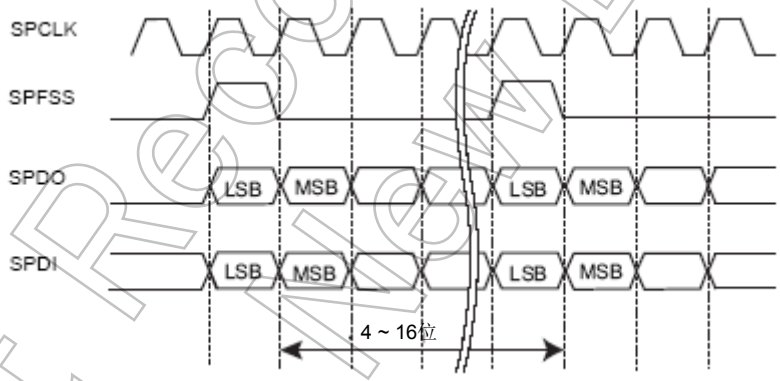


图 13-3 SSI 帧格式 (连续传输期间的传输/接收)

注 1: 传输禁用时，SPDO 终端无输出且处于高阻抗状态。该终端需要增加适当上拉/下拉电阻，使电压电平有效。

注 2: SPDI 终端始终输入，且内部网关开启。传输信号处于高阻抗状态时，该终端需要增加适当上拉/下拉电阻，使电压电平有效。

13.6.2. SPI帧格式

SPI接口有4行。SPFSS用于从设备的选择。SPI格式其中一个主要特征是SSPCR0寄存器中的<SPO>与<SPH>位可用于设定 SPCLK 操作计时。

SSPCR0 <SPO>用于设置 SPCLK 保持空闲状态的电平。

SSPCR0 <SPH>用于选择数据锁存的时钟脉冲边沿。

	SSPCR0<SPO>	SSPCR0<SPH>
0	“低“ 态	捕获第 1 个时钟边沿的数据
1	“高“ 态	捕获当 2 个时钟边沿的数据

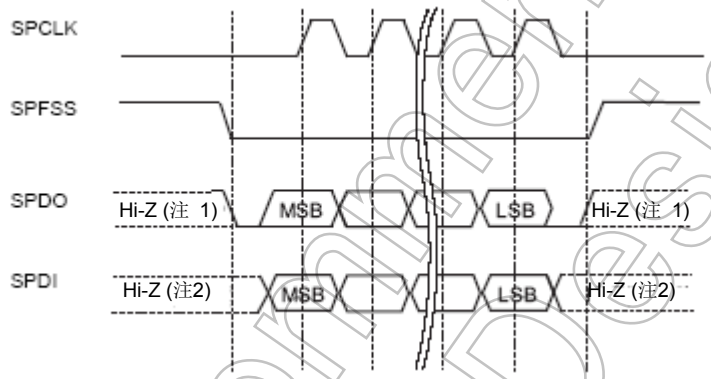


图 13-4 SPI 帧格式 (单次传输，<SPO> = “0 “ & <SPH> = “0“)

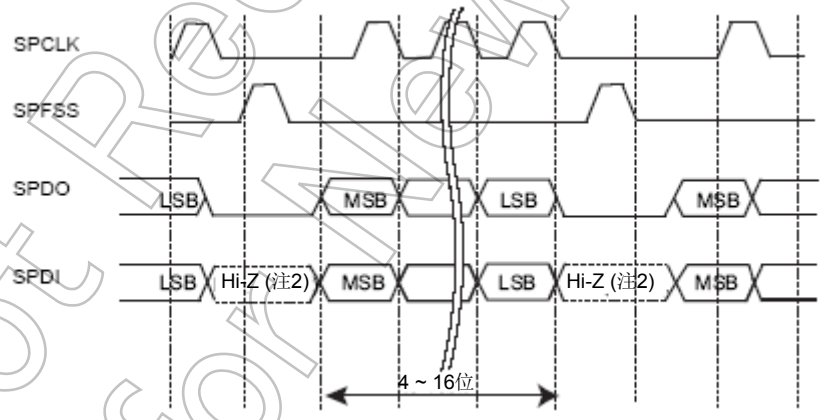


图 13-5 SPI 帧格式 (连续传输，<SPO> = “0“ & <SPH> = “0“)

注 1: 传输禁用时，SPDO 终端无输出且处于高阻抗状态。该终端需要增加适当上拉/下拉电阻，使电压电平有效。

注 2: SPDI 终端始终输入，且内部网关开启。传输信号处于高阻抗状态时，该终端需要增加适当上拉/下拉电阻，以获得有效电压电平。

空闲期内，在设置为 <SPO> = “0“的情况下:

SPCLK 信号设为 “低“。

SPFSS 设为 “高“。

・ 传输数据线 SPDO 设为 “低“。

当启用 SSP 且传输 FIFO 中存在有效数据时，受“低“驱动的 SPFSS 主信号通知传输开始。

从而启动主数据 SPDI 输入行中从属数据的传输。

SPCLK 周期过半时，主数据转到 SPDO 引脚。此时，主数据和从属数据设置完成。SPCLK 周期过了另一半时，SPCLK 主时钟引脚变 “高“。此后，数据在 SPCLK 信号上升沿被捕获，在其下降沿传输。SPCLK 周期过了另一半时，SPCLK 主时钟引脚变 “高“。

单次传输时，一旦所有位的数据字被全部传输，SPFSS 行将返回到空闲 “高“状态，在捕获最后一位后，一个 SPCLK 周期结束。

但是，对于连续传输，SPFSS 信号必须在个别数据字传输之间的 HIGH 处进行脉冲调制。这是因为当从属选择引脚将数据冻结在其外围寄存器且<SPH>位为逻辑 0 时，不能进行变更。

因此，当启用串行外围接口数据写入时，主设备必须驱动单个数据传输之间的从属设备的 SPFSS 引脚。一旦连续传输完成，且捕获最后一位后一个 SPCLK 周期结束，SPFSS 引脚将返回到空闲状态。

13.6.3 微总线帧格式

微总线帧格式采用以特殊主/从属信息传输方法，在半双工模式下操作。此模式下，微总线帧开始时，一条 8-位控制信息被传输到从属设备。传输期间，SSP 不会收到任何输入数据。信息传输完毕后，从属设备对其进行解码，在 8-位控制信息的最后一个位传输结束一个串行时钟后，从属设备对请求的数据做出响应。返回数据的位长度可能在 4 ~ 16 位之间变化，从而确保微总线帧总长度在任何情况下都在 13 ~ 25 位之间变化。

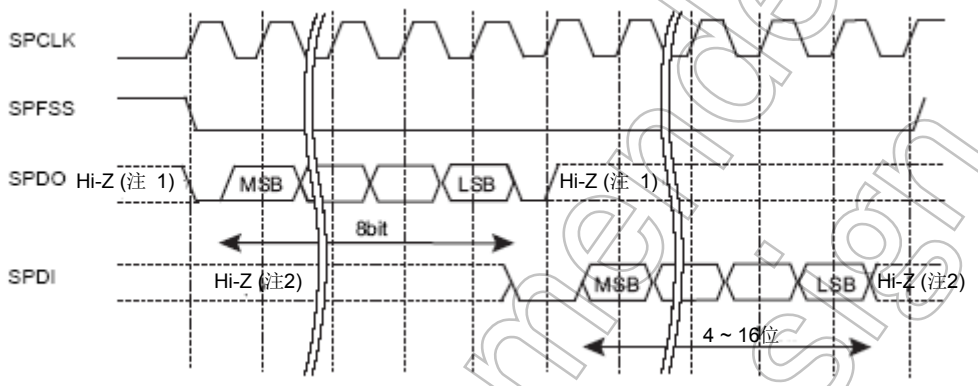


图 13-6 微总线帧格式 (单次传输)

注 1: 传输禁用时，SPDO 终端处于高阻抗状态，无任何输出。该终端需要增加适当上拉/下拉电阻，以固定电压电平。

注 2: SPDI 终端始终输入，且内部网关开启。传输信号处于高阻抗状态时，该终端需要增加适当上拉/下拉电阻，以获得有效电压电平。

尽管微总线帧格式与 SPI 格式类似，其仍然采用主/从属信息传输方法进行半双工通讯。每一次串行传输均通过一个 8-位控制字启动，该控制字被传输到芯片外从属设备。传输期间，SSP 不会接收任何输入数据。信息传输完毕，片外从属设备对其进行解码，且在该 8-位控制信息的最后一个位传输结束一个串行时钟后，从属设备对请求的数据做出响应。返回数据的位长度可能在 4 ~ 16 位之间变化，从而确保微总线帧总长度在任何情况下都在 13 ~ 25 位之间变化。空闲期内在采用该配置的情况下:

将 SPCLK 信号设为 “低“。

将 SPFSS 设为 “高“。

将传输数据行 SPDO 设为 “低“。

通过将一个控制位写入传输 FIFO，启动数据传输。SPFSS 下降沿会将传输 FIFO 底部输入部分存储的参数值转入传输逻辑的串行移位寄存器，8-位控制帧的 MSB 将被移到 SPDO 引脚。

数据传输期间，SPFSS 始终处于 “低“，而 SPDI 引脚始终处于三种状态之间。片外从属设备在每一个 SPCLK 上升沿将每一个控制位锁入其串行移位器。

最后一个位被从属设备锁存后，控制位在一个时钟等待状态被解码，从属设备通过将数据传回 SSP 而响应。每一个位都在 SPCLK 下降沿被拖入 SPDI 分帧线。

相反，SSP 在 SPCLK 上升沿锁存每一个位。在微总线帧末端，对于单次传输，SPFSS 信号在最后一个位被锁存入接收串行移位器后一个时钟周期内被推 “高“，使该数据被转移到接收 FIFO。

注:片外从属设备在最低有效位被接收移位器锁存后或在 SPFSS 引脚进入 “高”时，在 SPCLK 下降沿使接收线达到三种状态。

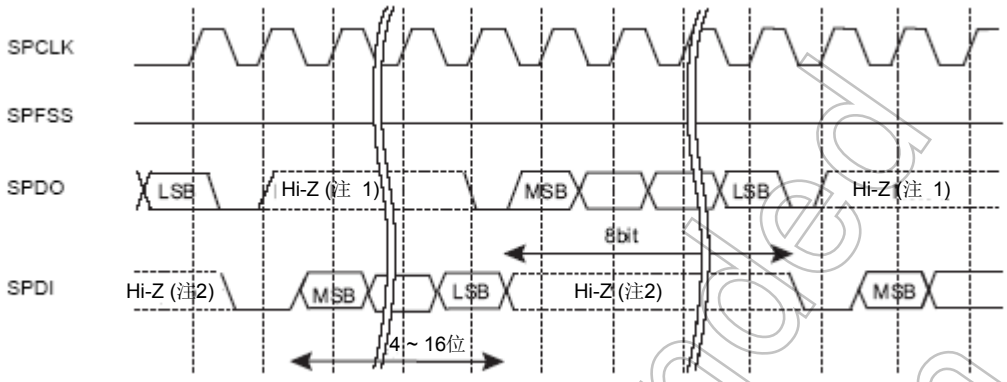


图 13-7 微总线帧格式 (连续传输)

注 1:传输禁用时，SPDO 终端处于高阻抗状态，无任何输出。该终端需要增加适当上拉/下拉电阻，以固定电压电平。

注 2:SPDI 终端始终输入，且内部网关开启。传输信号处于高阻抗状态时，该终端需要增加适当上拉/下拉电阻，以固定电压电平。

数据连续传输开始和结束的方式与单次传输完全相同。然而，SPFSS 分帧线行连续处于低电平 (保持低)，数据传输出现背对背。

下一帧控制位从当前帧接收的数据最低有效位后直接传输。该帧最低有效位被锁存到 SSP 后，所接收的每一个参数值都在 SPCLK 下降沿从接收移位器中转出。

注:[连接实例] SSP 不支持系统内主设备与从属设备之间的动态切换。每一例中，SSP 要么作为主设备，要么作为从属设备进行配置和连接。

14. 串行总线接口 (I2C/SIO)

TMPM364F10FG 系统包含 5 个串行总线接口 (I2C/SIO)通道，其中包含下列两种操作模式:

- I2C 总线模式 (具备多主设备能力)
- 时钟同步 8-位 SIO 模式

在 I2C 总线模式下，I2C/SIO 通过 SCL 和 SDA 与外部设备连接。

在时钟同步 8-位 SIO 模式下，I2C/SIO 通过 SCL，串行输入和串行输出与外部设备连接。

下表所列为 I2C/SIO 在各操作模式下所要求的编程。

表 14-1 使用串行总线接口的端口设置

通道	操作模式	引脚	端口功能寄存器	端口输出控制寄存器	端口输入控制寄存器	端口开漏输出控制寄存器
SBI0	I2C 总线模式	SCL0 :PL1 SDA0 :PL0	PLFR1[1:0] = 11	PLCR[1:0] = 11	PLIE[1:0] = 11	PLOD[1:0] = 11
	SIO模式	SCK0 :PL2 SI0 :PL1 SO0 :PL0	PLFR1[2:0] = 111	PLCR[2:0] = 101 (SCK0 输出) PLCR[2:0] = 001 (SCK0 输入)	PLIE[2:0] = 010 (SCK0 输出) PLIE[2:0] = 110 (SCK0 输入)	PLOD[2:0] = xxx
SBI1	I2C 总线模式	SCL1 :PG1 SDA1 :PG0	PGFR1[1:0] = 11	PGCR[1:0] = 11	PGIE[1:0] = 11	PGOD[1:0] = 11
	SIO模式	SCK1 :PG2 SI1 :PG1 SO1 :PG0	PGFR1[2:0] = 111	PGCR[2:0] = 101 (SCK1 输出) PGCR[2:0] = 001 (SCK1 输入)	PGIE[2:0] = 010 (SCK1 输出) PGIE[2:0] = 110 (SCK1 输入)	PGOD[2:0] = xxx
SBI2	I2C 总线模式	SCL2 :PG5 SDA2 :PG4	PGFR1[5:4] = 11	PGCR[5:4] = 11	PGIE[5:4] = 11	PGOD[5:4] = 11
	SIO模式	SCK2 :PG6 SI2 :PG5 SO2 :PG4	PGFR1[6:4] = 111	PGCR[6:4] = 101 (SCK2 输出) PGCR[6:4] = 001 (SCK2 输入)	PGIE[6:4] = 010 (SCK2 输出) PGIE[6:4] = 110 (SCK2 输入)	PGOD[6:4] = xxx
SBI3	I2C 总线模式	SCL3 :PH1 SDA3 :PH0	PHFR1[1:0] = 11	PHCR[1:0] = 11	PHIE[1:0] = 11	PHOD[1:0] = 11
	SIO模式	SCK3 :PH2 SI3 :PH1 SO3 :PH0	PHFR1[2:0] = 111	PHCR[2:0] = 101 (SCK3 输出) PHCR[2:0] = 001 (SCK3 输入)	PHIE[2:0] = 010 (SCK3 输出) PHIE[2:0] = 110 (SCK3 输入)	PHOD[2:0] = xxx
SBI4	I2C 总线模式	SCL4 :PH5 SDA4 :PH4	PHFR1[5:4] = 11	PHCR[5:4] = 11	PHIE[5:4] = 11	PHOD[5:4] = 11
	SIO模式	SCK4 :PH6 SI4 :PH5 SO4 :PH4	PHFR1[6:4] = 111	PHCR[6:4] = 101 (SCK4 输出) PHCR[6:4] = 001 (SCK4 输入)	PHIE[6:4] = 010 (SCK4 输出) PHIE[6:4] = 110 (SCK4 输入)	PHOD[6:4] = xxx

注:X:忽略

14.1 配置

配置如图 14-1 所示。

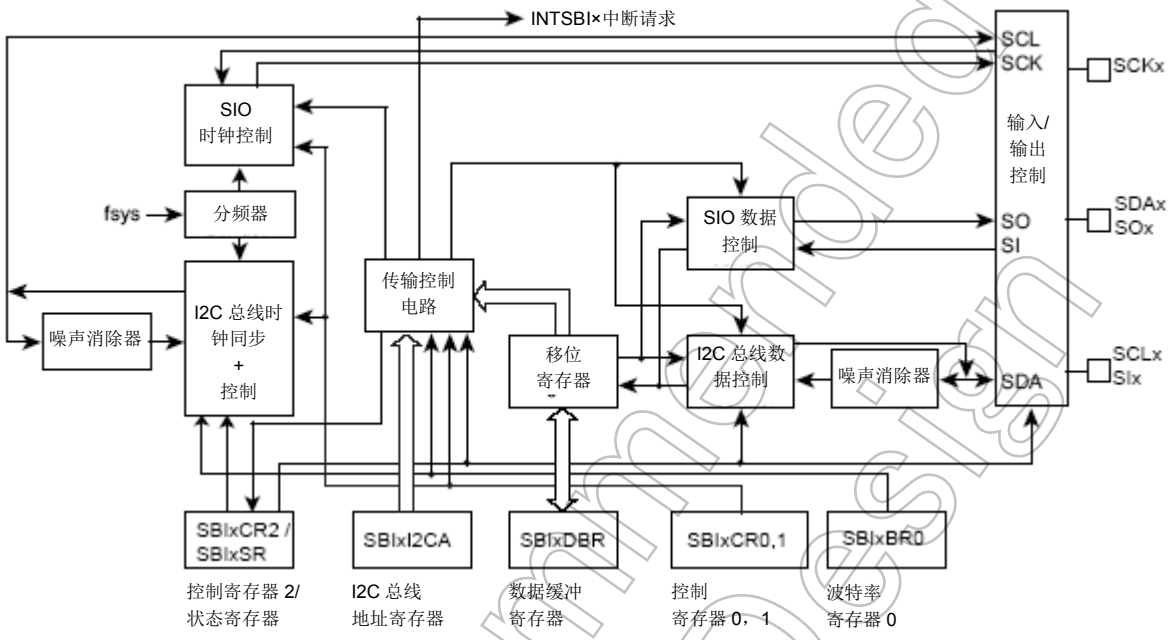


图 14-1 (I2C/SIO)块接口

14.2 寄存器

下列寄存器主要控制串行总线端口并提供其监测状态信息。

下列寄存器主要完成不同功能，具体取决于寄存器操作模式。详情见“14.4 I2C 总线模式控制寄存器“和 “14.7 SIO 模式控制寄存器“。

14.2.1 各通道寄存器

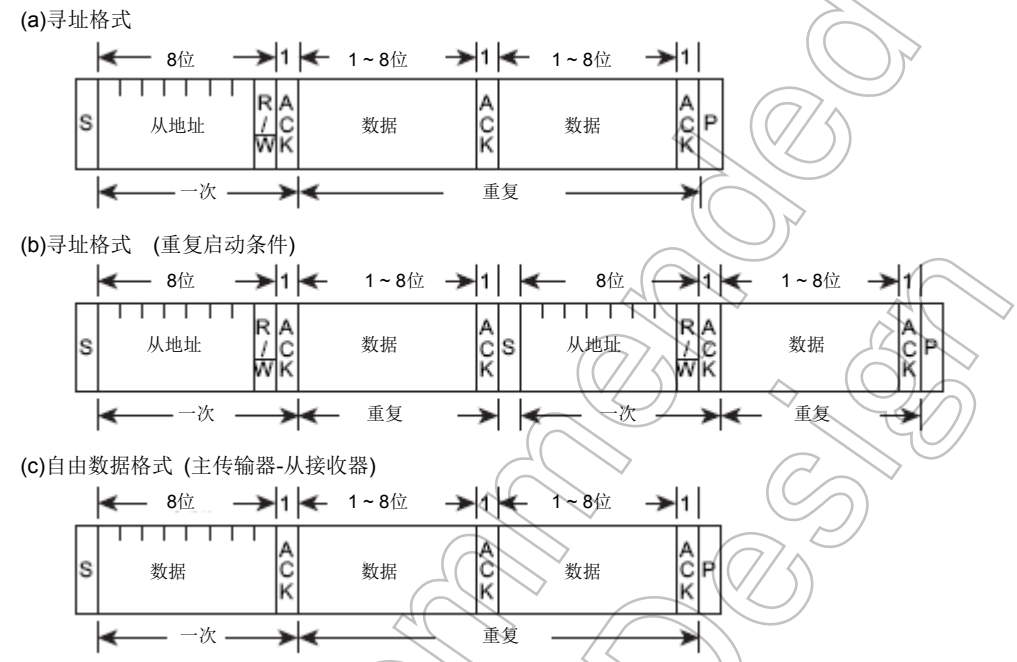
下表所列为各通道寄存器和寄存器地址。

通道 x	基址
通道 0	0x400E_0000
通道 1	0x400E_0100
通道 2	0x400E_0200
通道 3	0x400E_0300
通道 4	0x400E_0400

寄存器名称 (x=0, 1, 2, 3, 4)		地址 (基+)
控制寄存器 0	SBIxCR0	0x0000
控制寄存器 1	SBIxCR1	0x0004
数据缓冲寄存器	SBIxDBR	0x0008
I2C总线地址寄存器	SBIxI2CAR	0x000C
控制寄存器 2	SBIxCR2 (写入)	0x0010
状态寄存器	SBIxSR (读取)	
波特率寄存器 0	SBIxBR0	0x0014

14.3 I2C总线模式数据格式

图 14-2 所示为 I2C 总线模式下使用的数据格式。



注) S :启动条件
R / W :方向位
ACK : 确认位
P : 停止条件

图 14-2 I2C 总线模式数据格式

14.4 I2C总线模式控制寄存器

下列寄存器主要控制 I2C 总线模式下的串行总线接口并提供其监测状态信息。

14.4.1 SBIxCR0 (控制寄存器 0)

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

位	比特符号	型号	功能
31-8	-	R	读作 0。
7	SBIEN	R/W	串行总线接口操作 0:禁用 1:启用 用该串行总线接口时，必须首先启动该位。 在第一次设置启动时，相关SBI寄存器可读取或写入。 当该位禁用时，除SBIxCR0外的所有时钟都停止，因此，可停用该位以降低功耗。 当该位在一次启动后被禁用，各个寄存器的设置都被保留。
6-0	-	R	读作 0。

注:使用该串行总线接口时，必须首先启动该位。

14.4.2 SBIxCR1 (控制寄存器 1)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	BC			ACK	-	SCK2	SCK1	SCK0 / SWRMON
复位后	0	0	0	0	1	0	0	1 (注 3)

位	比特符号	型号	功能
31-8	-	R	读作 0。
7-5	BC[2:0]	R/W	选择每次传输时的位数 (注 1)

<BC>	<ACK> = 0时		<ACK> = 1时	
	时钟周期数	数据长度	时钟周期数	数据长度
000	8	8	9	8
001	1	1	2	1
010	2	2	3	2
011	3	3	4	3
100	4	4	5	4
101	5	5	6	5
110	6	6	7	6
111	7	7	8	7

4	ACK	R/W	主模式 0: 确认时钟脉冲未生成。 1: 确认时钟脉冲已生成。 从模式 0: 确认时钟脉冲未计数。 1: 确认时钟脉冲计数。
3	-	R	读作 1。
2-1	SCK[2:1]	R/W	选择内部SCL输出频率 (注2)
0	SCK[0]	W	

000	n = 5	615 kHz
001	n = 6	471 kHz
010	n = 7	320 kHz
011	n = 8	195 kHz
100	n = 9	110 kHz
101	n = 10	58 kHz
110	n = 11	30 kHz
111		保留

系统时钟:fsys (=64MHz)
时钟齿轮 fc/1
频率 = $\frac{fsys}{2^n + 72}$ [Hz]

	SWRMON	R	读取<SWRMON>:软件复位状态监控器 0:软件复位操作正在进行中。 1:软件复位操作未进行。

注 1: 在从操作模式切换 ~ SIO 模式之前 <BC[2:0]>清为 “000“。

注 2: 如需有关 SCL 线路时钟脉冲频率的详细信息，请参见“14.5.1 串行时钟“ 。

注 3: 复位后，<SCK[0]/SWRMON>位为 “1“。然而，当在 SBIxCR2 寄存器中选择 SIO 模式时，<SCK[0]>位初始值为 “0“。

注 4: 频率选择的初始值为<SCK[2:0]>=000，不受初始读数值的影响。

注 5: 主模式下，<BC[2:0]> = “001“，<ACK>= “0“时，SCL 总线路可在 STOP 条件生成后通过 SCL 总线路下降沿固定到 “L“电平，而其他主设备则不能使用该总线。如果是与多台主设备连接的总线，STOP 条件生成之前，每次传输的位数应等于或大于 2。

14.4.3 SBIxCR2 (控制寄存器 2)

此寄存器以 SBIxSR 寄存器读取。

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	MST	TRX	BB	PIN	SBIM		SWRST	
复位后	0	0	0	1	0	0	0	0

位	比特符号	型号	功能
31-8	-	R	读作 0。
7	MST	W	选择主/从模式 0:从模式 1:主模式
6	TRX	W	选择传输/ 接收 0:接收 1:传输
5	BB	W	启动/停止条件生成: 0: 停止条件生成 1: 启动条件生成
4	PIN	W	解除INTSBIx中断请求 0: - 1: 解除中断请求
3-2	SBIM[1:0]	W	选择串行总线接口操作模式 (注) 00: 端口模式 (关闭串口总线接口输出) 01 :SIO模式 10: I2C总线模式 11: 保留
1-0	SWRST[1:0]	W	软件复位生成 写入 “10“然后写入 “01“生成复位值。 写入时，<SBIM[1:0]>设置到 “10“；I2C总线模式。

注:**确保通讯会话期间，相关模式未改变。确保总线操作模式切换到端口模式之前可自由切换。确保该端口操作模式从端口模式切换 ~ I2C 总线或时钟同步 8-位 SIO 模式之前始终处于 “高“电位。**

14.4.4 SBIxSR (状态寄存器)

该寄存器以 SBIxCR2 寄存器写入。

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	MST	TRX	BB	PIN	AL	AAS	ADO	LRB
复位后	0	0	0	1	0	0	0	0

位	比特符号	型号	功能
31-8	-	R	读作 0。
7	MST	R	主/从选择监视器 0: 从机模式 1: 主机模式
6	TRX	R	传输/接收选择监视器 0: 接收 1: 传输
5	BB	R	I2C总线状态监视器 0: 空闲 1: 忙碌
4	PIN	R	INTSBIx中断请求监视器 0:中断请求发出 1: 中断请求清除
3	AL	R	仲裁丢失检测 0: - 1: 检测
2	AAS	R	从地址匹配检测 0:- 1:检测 (也可在检测到全呼叫时设置该位元。)
1	ADO	R	广播检测 0:- 1:检测
0	LRB	R	最后接收的位监控器 0:最后接收的位 “0“ 1:最后接收的位 “0“

14.4.5 SBIxBR0 (串行总线接口波特率寄存器0)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	-	I2SBI	-	-	-	-	-	-
复位后	1	0	1	1	1	1	1	0

位	比特符号	型号	功能
31-8	-	R	读作 0。
7	-	R	读作 1。
6	I2SBI	R/W	在空闲模式下操作 0: 停止 1: 操作
5-1	-	R	读作 1。
0	-	R/W	确保写入 “0“。

14.4.6 SBIxDBR (串行总线接口数据缓冲寄存器)

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

位	比特符号	型号	功能
31-8	-	R	读作 0。
7-0	DB[7:0]	R (接收)/ W (传输)	接收数据 / 传输数据

注 1: **该传输数据必须从 MSB (位 7)写入寄存器。接收的数据应存入最低有效位。**

注 2: **由于 SBIxBDR 有独立的读写缓冲区，因此，不能读取写入的数据，因而不能使用读取-修改-写入指令 (比如位处理等)。**

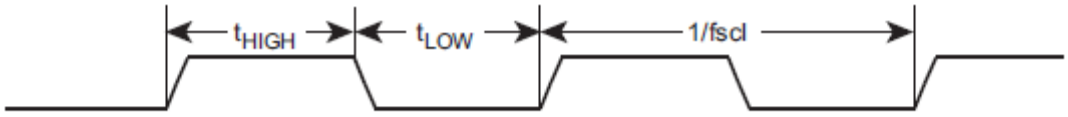
14.4.7 SBIxI2CAR (I2C总线地址寄存器)

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

位	比特符号	型号	功能
31-8	-	R	读作 0。
7-1	SA[6:0]	R/W	SBI作为从属设备时，应设置从地址。
0	ALS	R/W	规定地址确认模式。 0: 对其从地址予以确认。 1: 不对其从地址予以确认 (自由数据格式)。

注 1:**请将 I2C 总线地址寄存器 SBIxI2CAR 的 0 位<ALS>设为 “0“，但使用自由数据格式的情况除外。该位设置到 “1 “时，可用自由数据格式操作。选择主模式固定用于传输数据。选择从模式固定用于接收数据。**

注 2:**不得在从模式下 SBIxI2CAR 设置到 “0x00“。 (如果 SBIxI2CAR 设置到 “0x00“，可认定从地址与从模式下接收的 I2C 标准 “启动“位 (“0x01“)匹配。)**



$t_{LOW} = 2^{n-1}/fsys + 58/fsys$
$t_{HIGH} = 2^{n-1}/fsys + 14/fsys$
$fscl = 1/(t_{LOW} + t_{HIGH})$
$= \frac{fsys}{2^n + 72}$

SBIxCR1<SCK[2:0]>	n
000	5
001	6
010	7
011	8
100	9
101	10
110	11

。

$t_{LOW} = 2^{n-1}/fsys + 58/fsys$
$t_{HIGH} = 2^{n-1}/fsys + 14/fsys$
$fscl = 1/(t_{LOW} + t_{HIGH})$
$= \frac{fsys}{2^n + 72}$

SBIxCR1<SCK[2:0]>	n
000	5
001	6
010	7
011	8
100	9
101	10
110	11

图 14-3 时钟源

注:**标准模式下的最大速度和高速模式下的最大速度 分别依照通讯标准指定为 100kHz 和 400kHz。注意:内部 SCL 的时钟脉冲频率 由所使用的系统频率 f sys 确定并采用上文所列公式计算。**

14.5.1.2 时钟同步

因 I2C 总线的引脚结构，I2C 总线采用有线-AND 连接方式驱动。第一个主设备时钟线引到 “低“电平，重写其他在其时钟行产生 “高“电平的主设备，而且必须由产生 “高“电平的主设备检测并做出响应。

时钟同步旨在确保由两台或多台主设备的总线上正确的数据传输。

例如:下文所示为由两台主设备的总线时钟同步程序。

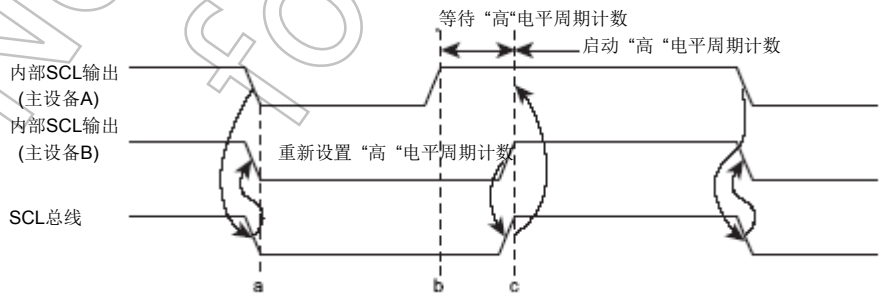


图 14-4 时钟同步举例

主设备 A 在 a 点将其内部 SCL 输出引入 “低“电平，同时将 SCL 总线带入 “低“电平。主设备 B 对该转变信息进行检测，重新设置其 “高“电平周期计数器，并将其内部 SCL 输出阶引入 “低“电平。

主设备 A 在 b 点完成其 “低“位周期计数，并将其内部 SCL 输出带入 “高“电平。然而，主设备 B 仍使 SCL 总线保持在 “低“电平，而主设备 A 停止其 “高“电平周期计数。主设备 A 检测到主设备 B 将其内部 SCL 输出带入 “高“电平并在 c 点 SCL 总线带入 “高“电平后，开始其“高“电平周期计数。

该主设备完成“高“电平周期计数后，SCL 引脚引入“低“电平，SCL 总线也 进入“低“电平。

这样，该总线上的时钟 由与该总线连接的主设备中 “高“电平周期最短的主设备和 “低“电平周期最长的主设备确定。

14.5.2 设置确认模式

SBIxCR1<ACK>设置到 “1“，选择确认模式。作为主设备操作时，SBI 增加一个确认信号时钟。确认信号时钟 在从模式下计数。SBI 可在传输器模式下，在时钟周期内释放 SDAx 引脚，接收来自接收器的确认信号。同时 SBI 可在接收器模式下，在时钟周期内 SDAx 引脚引入 “低“电平，并生成确认信号。

此外，在从模式下，如果接收到一个全呼地址，SBI 也可在时钟周期内将 SDAx 引脚引入 “低“电平，并生成确认信号。将<ACK>设置到 “0“ ，可激活非确认模式。作为主设备操作时，SBI 不生成确认信号时钟。确认信号时钟 在从模式下计数。

14.5.3 设置每次传输的位数

SBIxCR1<BC[2:0]>旨在确定下一个传输或接收数据的位数。

在启动条件下，<BC[2:0]>设为 “000“，使得从地址和方向位在八位数据包中传输。而在其他时间，<BC[2:0]> 保持事先已编程的参数值。

14.5.4 从地址与地址确认模式

对 SBIxI2CAR<ALS>以及 SBIxI2CAR<SA[6:0]>的一个从地址进行 “0“设置，设置寻址格式，然后由 SBI 确认从主设备传输的一个从地址，并接收该寻址格式的数据。

如果<ALS>设置到 “1“ ，SBI 不对一个从地址进行确认，而接收自由数据格式的数据。对于自由数据格式，从地址和方向位得不到确认；它们在启动条件生成后立即被确认为数据。

14.5.5 操作模式

SBIxCR2<SBIM[1:0]>的设置旨在控制该操作模式。要在 I2C 模式下操作，必须确保串行总线接口引脚在将 <SBIM[1:0]>设置到 “10“之前处于 “高“电平。此外，还必须确保总线操作模式切换到端口模式之前可自由切换。

14.5.6 SBI设置为传输器或接收器

SBIxCR2<TRX>设置到 "1"可将 SBI 设置为传输器。<TRX>设置到 "0"可将 SBI 设置为接收器。

在从模式下:

在寻址格式下传输数据时;

接收到的从地址与 SBIxI2CAR 规定的参数值匹配时;

接收全呼地址时;即满足启动条件的八位位元均为零。

当该方向位 (R/$\overline{W}$)数值为 "1" 时,<TRX> 通过硬件设为 "1"。当该方向位 (R/$\overline{W}$)数值为 "0"时,<TRX>设置到 "0"。

SBI 作为主设备时,应接收从设备发出的确认信号。如果发送方向位值 "1",<TRX> 通过硬件设置到 "0"。如果方向位为 "0",<TRX> 改为 "1"。如果 SBI 没有收到确认信号,<TRX> 保留原值。

<TRX>检测到总线上的停止条件或仲裁丢失时,通过硬件清 "0"。

如果在自由数据格式下使用 SBI,不得通过硬件更改<TRX>。

14.5.7 将SBI设置为主设备或从设备

将 SBIxCR2<MST>设置到 "1",同时将 SBI 设置为主设备操作。

将<MST>设为 "0",同时将 SBI 设为从设备。< MST >检测到总线上的停止条件或仲裁丢失时,通过硬件清 "0 "。

14.5.8 生成启动条件和停止条件

SBIxSR<BB>为 "0"时,将 "1" 写入 SBIxCR2<MST,TRX,BB,PIN>将促使 SBI 启动总线 "启动条件 "生成程序,同时输出数据缓冲寄存器中可能写入的从地址和方向位。

而<ACK>必须事先设到 "1"。

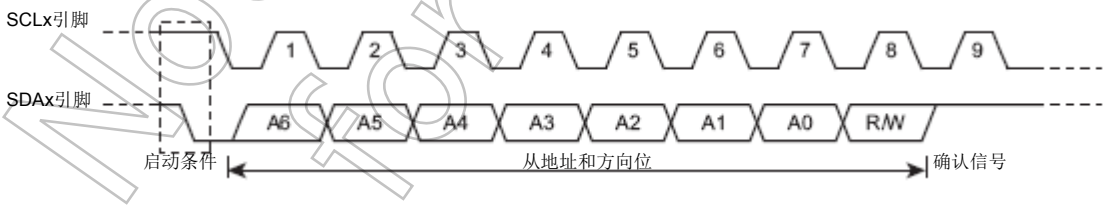


图 14-5 生成启动条件和从地址

<BB>为 "1"时,将 "1"写入<MST,TRX,PIN>,将 "0"写入<BB> 促使 SBI 开始执行总线 "停止条件"生成序列。在总线上出现停止条件之前,<MST,TRX,BB,PIN>的内容不得变更。

停止条件生成时，可通过其他设备将 SCL 总线引入 “低“，并在 SCL 总线释放后生成停止条件。

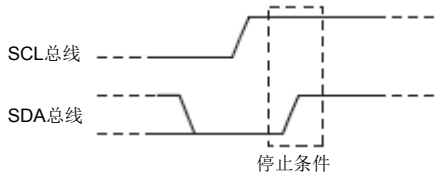


图 14-6 生成停止条件

可读取 SBIxSR<BB>，检查总线状态。在总线上检测到启动条件时 (总线忙)，将<BB>设置到 “1“；检测到停止条件时 (总线空闲)将<BB>清 “0“。

14.5.9 中断服务请求及其解除

完成<BC>和<ACK>设置的时钟周期数转移时，串行总线接口请求 (INTSBIx)在主模式下生成。

在从模式下，INTSBIx 在下列条件下生成。

收到的从地址与 SBIxI2CAR<SA[6:0]>设置的从地址匹配时生成的确认信号输出后。

在收到全呼叫地址的情况下生成确认信号后。

收到全呼地址后从地址匹配或数据传输完成时。

在地址确认模式下 (<ALS> = “0“)，INTSBIx 将在收到的从地址与 SBIxI2CAR 规定的参数值匹配时或在收到全呼叫地址的情况下 (满足启动条件后的八位均 “0“)生成。

生成中断请求 (INTSBIx)时，SBIxCR2<PIN> 清 “0“。<PIN>清 “0“时，SBI 将 SCL 总线引入 “低“电平。

数据从 SBIxDBR 写入或读取时，<PIN>设置到 “1“。<PIN>设置到 “1“后需要一个 t_{LOW} 周期释放 SCL 总线。程序 “1“ 写入<PIN>时，设置到 “1“。然而，写入 “0“ 不能将该位清为 “0“。

注:主模式下仲裁丢失时，如果从地址不匹配，<PIN>不清为 “0“ (生成 INTSBIx)。

14.5.10 仲裁丢失检测监控器

I2C 总线具备多主设备功能 (一条总线上有两台或多台主设备)并且要求执行总线仲裁程序，确保数据传输正确。

在总线繁忙时试图生成启动条件的主设备 在总线仲裁中败诉，SDA 总线和 SCL 总线上无启动条件出现。I2C 总线仲裁在 SDA 总线上进行。

一根总线上两台主设备的仲裁程序如下图所示:

在到达 a 点之前，主设备 A 和主设备 B 输出相同的数据。而在 a 点，主设备 A 输出 “低“电平数据，而主设备 B 输出 “高 “位数据。

然而，主设备 A 将 SDA 总线引入 “低“电平，这是因为该总线为有线-AND 连接。SCL 总线在 b 点进入 “高“电平时，从设备 读取 SDA 总线数据 (即主设备 A 传输的数据)。同时，主设备 B 传输的数据失效。

主设备 B 的该状态被称为 “仲裁丢失“。此时，主设备 B 释放其 SDA 引脚，以确保其不对另一台主设备发起的数据传输构成影响。如果两台或多台主设备传输的第一个数据字完全相同，仲裁程序在第二个数据字继续实施。

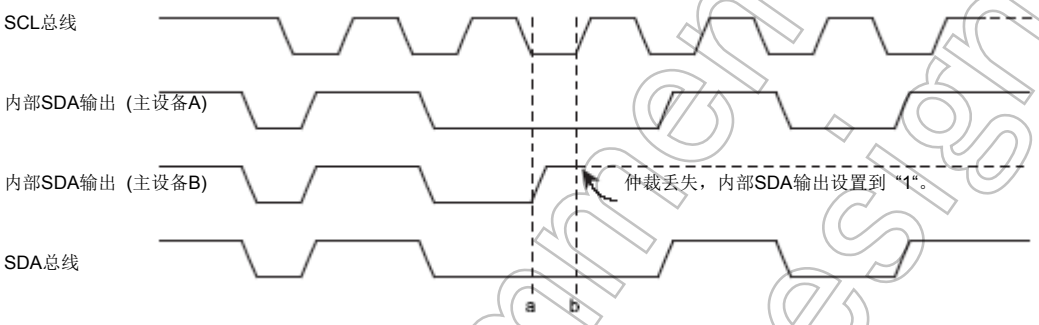


图 14-7 仲裁丢失

一台主设备在 SCL 总线上升沿对 SDA 总线电平和内部 SDA 输出电平进行比较。如果两个电平值之间存在差异，则仲裁丢失，同时 SBIxSR<AL> 被设为 “1“。

<AL>设置为 “1“时，SBIxSR<MST, TRX> 清为 “0“，SBI 作为从接收器操作。因此，串行母线接口电路在<AL>设置到 “1“后的数据传输期间，停止时钟输出。

数据从 SBIxDBR 中写入或读取时，或数据写入 SBIxCR2 时，<AL> 清为 “0“。

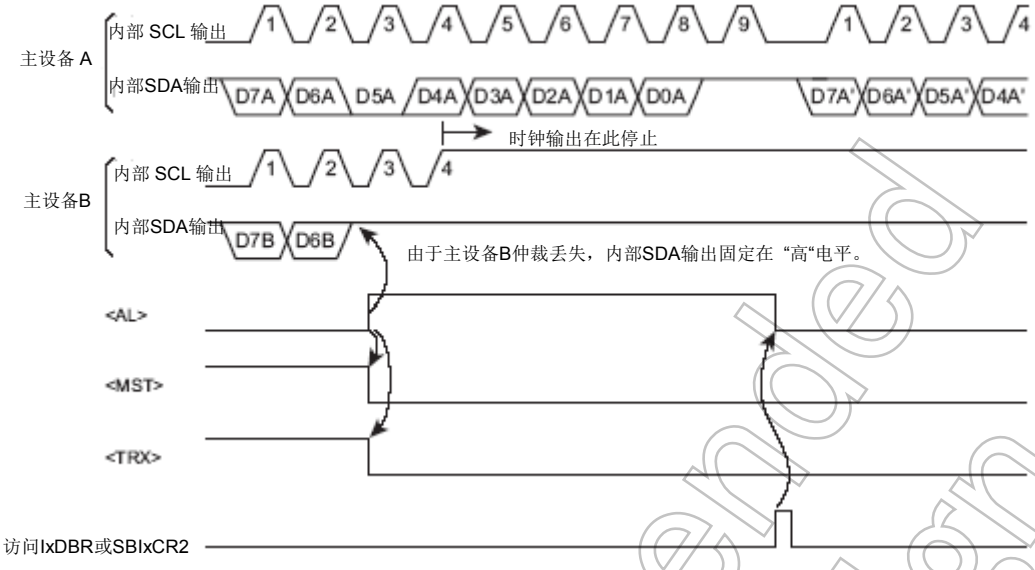


图 14-8 主设备 B 仲裁丢失 (D7A = D7B, D6A = D6B)的举例

14.5.11 从属地址匹配检测监控器

当 SBI 作为从属设备在地址识别模式 (SBIxI2CAR<ALS>= “0“)下操作时，在接受一般电话地址或与规定值相匹配的从属地址时，将 SBIxSR<AAS>设置到 “1“。

当<ALS>为 “1“时，收到第一个数据字时，将<AAS>设置到 “1“。当数据写入或从 SBIxDBR 读取时，<AAS>会被清 “0“。

14.5.12 一般电话检测监控器

当 SBI 以一个从属设备操作时，在接受一般搜寻地址时将 SBIxSR<ADO>设置到 “1“，即起始条件随后的八位均为零。

在总线上检测到起始或停止条件时，<ADO>被清为 “0“。

14.5.13 最新收到的位元监控器

SBIxSR<LRB> 设置到 SCL 线上升时读取的 SDA 线值。

在应答模式下，INTSBIx 中断请求产生后立即读取 SBIxSR<LRB>，可读取 ACK 信号。

14.5.14 数据缓冲寄存器 (SBIxDBR)

读取或写入 SBIxDBR 将开始读取接收到的数据或写入的传输数据。

当 SBI 作为一个主控设备时，为寄存器设置一个从属地址和指令位，生成起始条件。

14.5.15 波特率寄存器 (SBIxBR0)

进入 IDLE 模式时，SBIxBR0<I2SBI>寄存器确定 SBI 是否运行。

在执行切换到备用模式指令时，必须对该寄存器进行编程。

14.5.16 软件复位

如果串行总线接口电路因外部干扰而锁定，可使用软件复位来初始化。

将 “10“ 和 “01“写入 SBIxCR2<SWRST[1:0]>，会产生一个启动串行总线接口电路的复位信号。写入时，设置<SBIM[1:0]>为 “10“；I2C 总线模式。复位后，所有控制寄存器和状态标志将会被初始化到复位值。串行总线接口电路被初始化时，<SWRST>自动清为 “0“。

注:软件复位使 SBI 运行模式从 I2C 模式切换到端口模式。

14.6 I2C总线模式I2C数据传输程序

14.6.1 设备初始化

首先，编程 SBIxCR1<ACK, SCK[2:0]>，同时将 “000“写入 SBIxCR1<BC[2:0]>。

其次，通过指定一个<SA[6:0]>从属地址和<ALS>的地址识别模式以对 SBIxI2CAR 进行编程。 (在使用寻址格式时，<ALS>必须清为 “0“。)

将串行总线接口配置为从属接收器时，首先应确保串行总线接口引脚处于 “高“电平。然后将 “0“写入 SBIxCR2<MST, TRX, BB>，将 “1“写入<PIN>，将 “10“写入<SBIM[1:0]>，“0“写入 1 和 0 位。

注:串行总线接口电路初始化必须在某个段时间内完成，这段时间内所有连接到总线的设备初始化后任何设备没有生成启动条件。如果不遵守这一原则，可能不会正确地收到数据，因为其它设备可能在串行总线接口电路初始化完成之前已开始传输。

		7	6	5	4	3	2	1	0	
SBIxCR1	←	0	0	0	X	0	X	X	X	规定 ACK 和 SCL 时钟。
SBIxI2CAR	←	X	X	X	X	X	X	X	X	规定一个从属地址和一个地址识别模式。
SBIxCR2	←	0	0	0	1	1	0	0	0	将 SBI 配置成一个从属接收器。

注:X；忽略

14.6.2 生成起始条件和从属地址

14.6.2.1 主机模式

在主控模式下，需要采取以下步骤来生成起一个始条件和从属地址。

首先，确保总线闲置 (<BB>= “0“)。然后， 将 “1“ 写入 SBIxCR1<ACK>选择应答模式。将一个从属地址和一个待传输的指令位写入 SBIxDBR。

当<BB>= “0“时， 将 “1111“ 写入 SBIxCR2<MST, TRX,BB,PIN>在总线上生成起始条件。生成起始条件之后，SBI 通过 SCL 引脚生成了九个时钟。SBI 输出从属地址和 SBIxDBR 最初 8 个时钟情况下规定的方向位，并且在第九个时钟里释放 SDA 线以接收从属设备的一个应答信号。

在第九个时钟下降时产生 INTSBIx 中断请求并且将<PIN>清为 “0“ 。在主控模式下，当<PIN>= “0“ 时，SBI 保持 SCL 线在 “低“电平。如果一个应答信号已经从从属设备返回，<TRX>根据 INTSBIx 中断请求生成时传输的方向位改变其数值。

注:输出从属地址时，在写入 SBIxDBR 之前，需要用软件检查总线是否空闲。如果不遵守这一原则，在总线上待输出的数据有可能会损坏。

主程序中的设置

```
                    7  6  5  4  3  2  1  0
Reg.          ←  SBIxSR                          确保总线空
Reg.          ←  Reg. e 0x20                     选择应答模式
if Reg.       ≠  0x00                            规定需要的从属地址和方向位
Then                                             生成起始条件
SBIxCR1       ←  X  X  X  1  0  X  X  X
SBIxDBR       ←  X  X  X  X  X  X  X  X
SBIxCR2       ←  1  1  1  1  1  0  0  0

                 中断程序示例
                 清除中断请求
                 处理中
                 中断结束
```

14.6.2.2 从模式

在从模式下，SBI 接收到起始条件和一从属地址。

收到主控设备的起始条件之后，在 SCL 线路上的最初 8 个时钟期间，SBI 接收到一个来自主控设备的从属地址和方向位。

如果收到的地址与其 SBIxI2CAR 中指定的从属地址相匹配，或等于广播寻址，在第九个时钟期间，SBI 将 SDA 线路拉到 “低“电平，并输出一个应答信号。

在第九个时钟下降时产生 INTSBIx 中断请求并且将<PIN>清为 “0“。在从属模式下，SBI 保持 SCL 线路在 “低“电平，同时<PIN>清为 “0“。

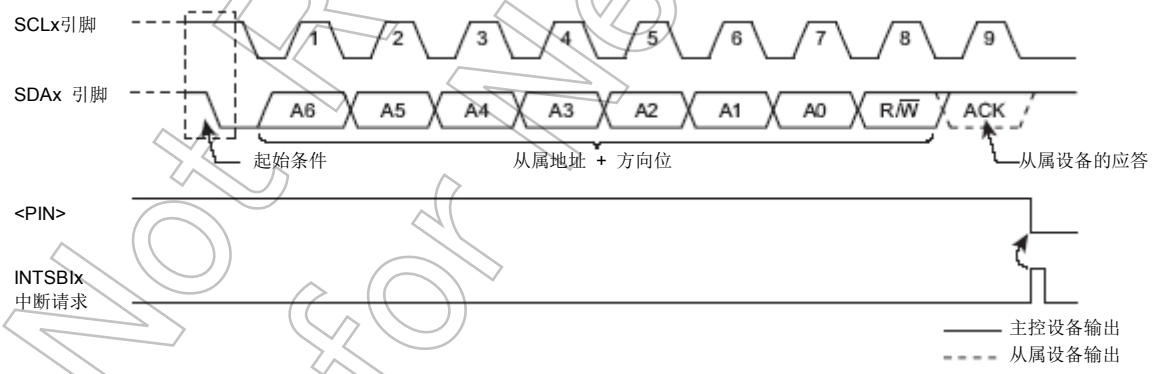


图 14-9 起始条件和从属地址生成

14.6.3 传输一个数据字

数据字传输结束时，INTSBIx 中断命令生成，测试<MST>以判定 SBI 是否在主控模式或从属模式下。

14.6.3.1 主机模式 (<MST> = “1“)

测试<TRX>以判断是否将 SBI 配置为一个传输器或接收器。

(1) 传输器模式 (<TRX> = “1“)

测试<LRB> 如果<LRB>为 “1“，表示接收器不需要更多数据。

然后主设备生成了所描述的停止条件，停止传输。

如果<LRB>为 “0“，即表示接收器需要更多的数据。如果待传输的下一组数据有八位，则数据被写入 SBIxDBR。如果数据有不同的长度，则对<BC[2:0]>和<ACK>进行编程，并将传输数据写入 SBIxDBR。写入数据使<PIN>为 “1“，使 SCL 引脚生成一个串行时钟，传输下一组数据字，并使 SDA 引脚传输数据字。

传输完成之后，INTSBIx 中断请求生成。<PIN> 清除为 “0“，SCL 引脚被拉 “低“电平。

传输更多的数据字时，再次测试<LRB>并重复上述过程。

INTSBIx 中断

if MST = 0

然后开始从属模式处理。

if TRX = 0

然后开始接收器模式处理。

if LRB = 0

然后开始处理生成停止条件。

SBIxCR1	←	X	X	X	X	0	X	X	X
SBIxDBR	←	X	X	X	X	X	X	X	X

规定待传输的位数，规定是否需要 ACK。

写入传输数据。

中断处理结束。

注:**X：忽略**

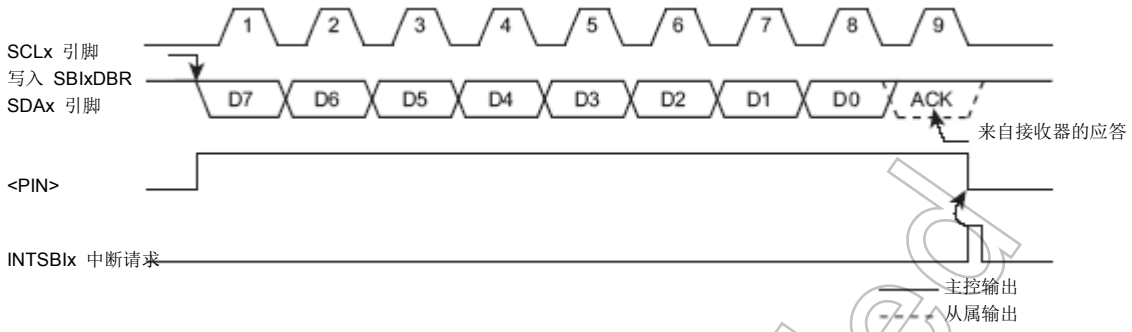


图 14-10 <BC[2:0]>= “000“,<ACK>= “1“ (传输器模式)

(2) 接收器模式 (<TRX> = “0“)

如果待传输的下一组数据为八位，则将传输数据写入 SBIxDBR。

如果数据有不同长度，则对<BC[2:0]>和<ACK>进行编程，并从 SBIxDBR 读取接收到的数据，释放 SCL 传输线。 (从属地址的传输后立即读取的数据将不被赋值未定义。) 读取该数据时，<PIN>设置到 “1“ ，串行时钟输出到 SCL 引脚，传输下一组数据字。在最后一个位中，当应答信号变为 “低“电平时，将 “0“输出到 SDA 引脚。

此后，TSBIx 中断请求生成，<PIN>清除为 “0“，SCL 引脚拉 “低“电平。每次从 SBIxDBR 读取接收数据时，输出一个单字传输时钟和一个应答信号。

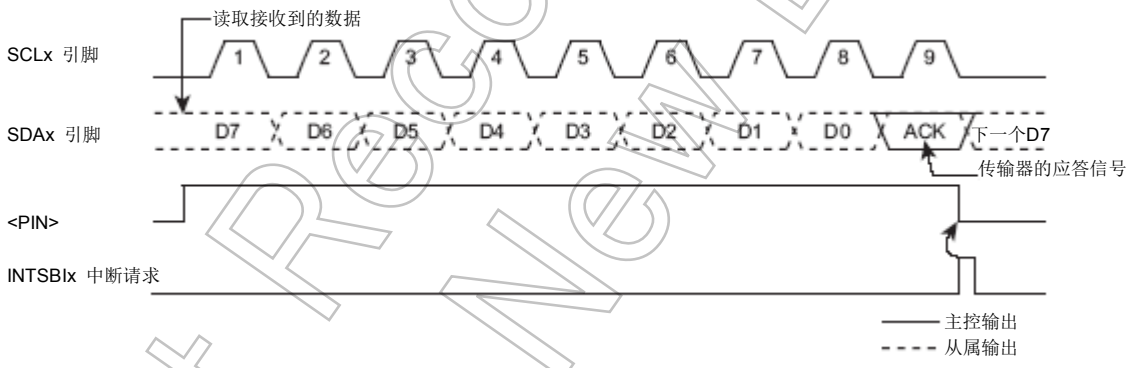


图 14-11 <BC[2:0]>= “000“，<ACK>= “1“ (接收器模式)

结束传输器的传输时，在读取倒数第二个数据字之前，<ACK>必须立即清 “0“。

禁止生成最后数据字的应答时钟。

传输完成时，中断请求生成。中断处理后，<BC[2:0]> 必须设置到 “001“并读取数据，因此时钟生成，进行 1 位传输。

此时，主接收器保持 SDA 总线线路在 “高“电平，表示作为一个应答信号传输到传输器结束。

在中断接收 1-位数据的中断处理过程中，停止条件生成，数据传输结束。

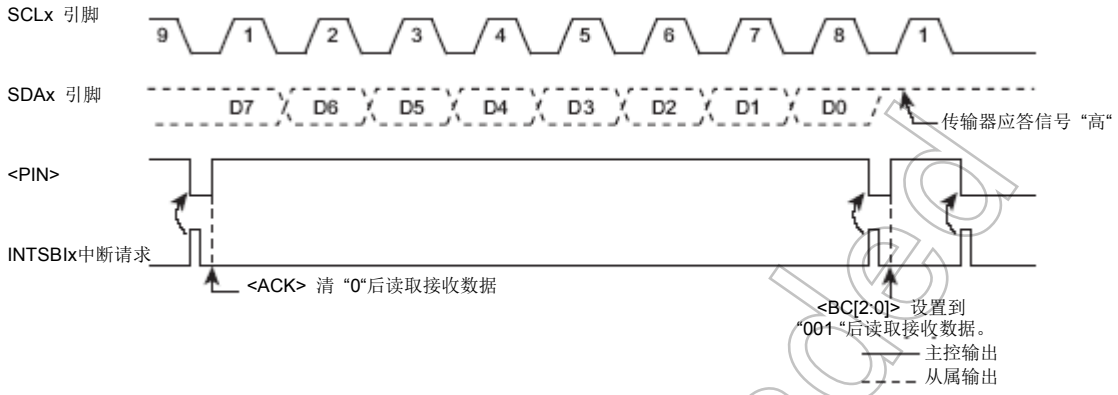


图 14-12 主接收器模式下终止数据传输

示例:当接收 N 个数据字时

INTSBIx 中断 (数据传输之后)

		7	6	5	4	3	2	1	0	
										设置待接收的数据的位数，并规定是否需要 ACK。
SBIxCR1	←	X	X	X	X	0	X	X	X	读取假设数据。
Reg.	←	SBIxDBR								

中断结束

INTSBIx 中断 (第一 ~ (N-2)个数据接收)

		7	6	5	4	3	2	1	0	
										读取 第一 ~ 第 (N-2)个数据字。
Reg.	←	SBIxDBR								

中断结束

INTSBIx 中断 (第 (N-1)个数据接收)

		7	6	5	4	3	2	1	0	
										未能生成应答时钟。
SBIxCR1	←	X	X	X	0	0	X	X	X	读取 第 (N-1)个数据字。
Reg.	←	SBIxDBR								

中断结束

INTSBIx 中断 (第 N 个数据接收)

		7	6	5	4	3	2	1	0	
										未能生成应答时钟。
SBIxCR1	←	0	0	1	0	0	X	X	X	读取第 N 个数据字。
Reg.	←	SBIxDBR								

中断结束

INTSBIx 中断 (完成数据接收之后)

处理生成停止条件。 终止数据传输。

中断结束

注:X；忽略

14.6.3.2 从模式 (<MST> = “0“)

在从模式下，SBI 生成 INTSBIx 中断请求，存在以下四种情况:

1)当 SBI 已经从主控设备收到任何从属地址时。

2)当 SBI 已经收到一个广播寻址时。

3)当收到的从属地址与其地址匹配时。

4)当广播响应的数据传输完成时。

而且，如果 SBI 在主控模式下检测到仲裁丢失，将切换到从模式。

一旦检测到仲裁丢失的数据字传输完成，INTSBIx 中断请求生成，<PIN>清 “0“，并且 SCL 引脚拉到 “低“电平。

当数据写入或从 SBIxDBR 读取或<PIN>设置到 “1“ 时，SCLx 引脚在一个 t_{LOW} 周期后释放。

在从模式下，进行正常从模式处理或仲裁丢失处理。

测试 SBIxSR<AL>, <TRX>, <AAS> 和 <ADO>判断是否需要处理。

“表 14-2 从模式下处理“显示从模式条件和需要的处理。

示例:当接收到的从属地址与 SBI 本身的地址匹配时，并且方向位在从属接收器模式下为 “1“时。

INTSBIx 中断

if TRX = 0

然后进行其它处理。

if AL = 0

然后进行其它处理。

if AAS = 0

然后进行其它处理。

SBIxCR1	←	X	X	X	1	0	X	X	X	设置待传输的位数。
SBIxDBR	←	X	X	X	X	X	X	X	X	设置传输数据。

注:X：忽略

表 14-2 从属模式下的处理

<TRX>	<AL>	<AAS>	<ADO>	状态	处理中
1	1	1	0	在从属地址传输的同时，检测到仲裁丢失，SBI收到了一个方向位为 “1“的从属地址时，由另外一个主控设备传输。	将数据字中的位数设置到 <BC[2:0]> ，并将传输数据写入SBIxDBR。
	0	1	0	在从属接收模式下，SBI 接收到一个由主控设备传输的方向位为 “1 “的从属地址。	
		0	0	在从属传输模式中，SBI完成了一个数据字的传输。	测试LRB。如果设置为 “1“, 即表示接收器不需要更多的数据。设置<PIN> 为1并复位<TRX> 到 0 ，释放总线。如果 <LRB> 已经被复位到 “0“，即表示接收器需要更多的数据。将数据字中的位数设置到<BC[2:0]> ，并将传输数据写入SBIxDBR。
0	1	1	1/0	在从属地址传输的同时，检测到仲裁丢失，且SBI 接收到一个由另外一个主控设备传输方向位为 “0“的从属地址或一个广播寻址。	读取 SBIxDBR （一个假设读取），设置<PIN>为1或将 “1 “写入<PIN>。
		0	0	传输从属地址或一个数据字时，检测到仲裁丢失，传输终止。	
	0	1	1/0	在从属接受器模式下，SBI接收到了一个方向位为 “0“的从属地址或由主控设备传输的广播寻址。	
		0	1/0	在从属接收器模式里，SBI完成一个数据字的接收。	将数据字中的位数设置到<BC[2:0]> 并从SBIxDBR读取接收到的数据。

14.6.4 起始条件生成

当 SBIxSR<BB>为 “1“时，把 “1“写入 SBIxCR2<MST,TRX,PIN>，“0“写入<BB>，使 SBI 启动一个序列用于产生总线的停止条件。

不得改变<MST, TRX, BB, PIN>的内容，直到总线上出现停止条件。

如果有另外的设备在 SCL 总线线路上，SBI 等待，直到 SCL 线释放。

此后，SDA 引脚上升到 “高“电平时，使待生成的停止条件产生。

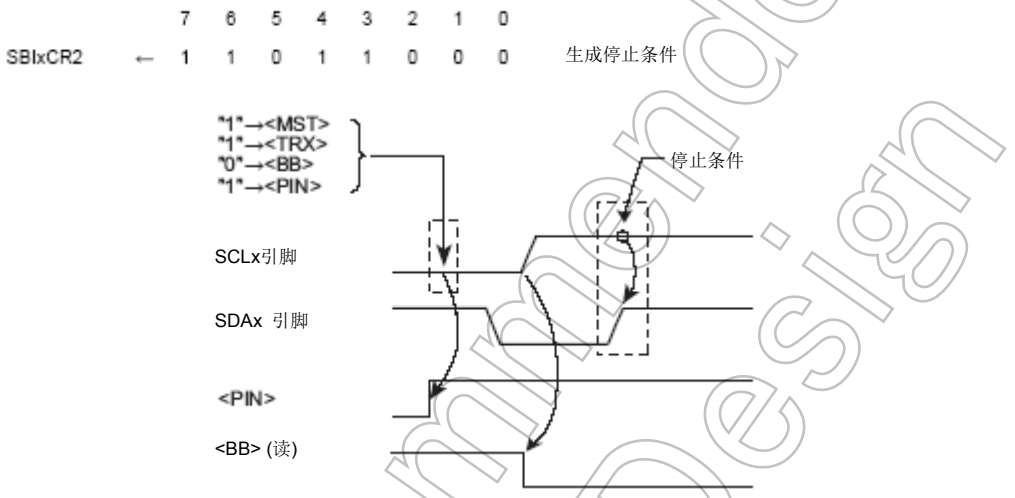


图 14-13 停止条件生成

14.6.5 重新启动程序

当主设备改变数据传输方向而不终止向从属设备的传输时重新启动。在主控模式下的重新启动程序如下所述。

首先，写入 SBIxCR2<MST, TRX, BB> 到 “0“ 并将 “1“写入<PIN>，释放总线。此时，SDAx 引脚保持在 “高“电平，SCLx 引脚释放。由于总线上没有产生停止条件，所以其它设备识别出总线忙碌。

然后，测试 SBIxSR<BB> 并等待，直到它变为 “0“，以确保 SCLx 引脚释放。

其次，测试<LRB>并等待，直到它变为 “1“，以确保没有其它设备将 SCLx 总线线路拉到 “低“电平。

一旦按照上述步骤确定总线空闲时，请遵守“14.6.2 从属地址和起始条件生成“中描述的步骤，生成起始条件。

为满足重新启动的设置时间，在判定总线空闲时，必须通过软件建立至少 4.7μs 的等待时期 (在标准模式下)。

注 1: 不得写入<MST>到 “0“，当它为“0“时。 (不能开始重新启动。)

注 2: 当主设备以一个接收器运行时，来自一个用作传输器的从属设备的数据传输必须在生成重新启动之前完成。为了完成数据传输，从属设备必须接受到一个 “高“电平应答信号。为此，<LBR>在生成重新启动之前变为 “1“,没有探测到 SCL 线的上升沿，甚至 是在 <LBR>=

“1“通过重新启动程序确认时。检查 SCL 线路的条件，读取端口。

		7	6	5	4	3	2	1	0	
SBIxCR2	←	0	0	0	1	1	0	0	0	释放总线。

if SBIxSR<BB> ≠ 0 检查 SCL引脚释放。
然后
if SBIxSR<LRB> ≠ 1 检查没有其它设备将SCL引脚拉到 “低“。
然后
4.7μs 等待

SBIxCR1	←	X	X	X	1	0	X	X	X	选择应答模式。
SBIxDBR	←	X	X	X	X	X	X	X	X	设置需要的从属地址和指令。
SBIxCR2	←	1	1	1	1	1	0	0	0	生成起始条件。

注:X；忽略

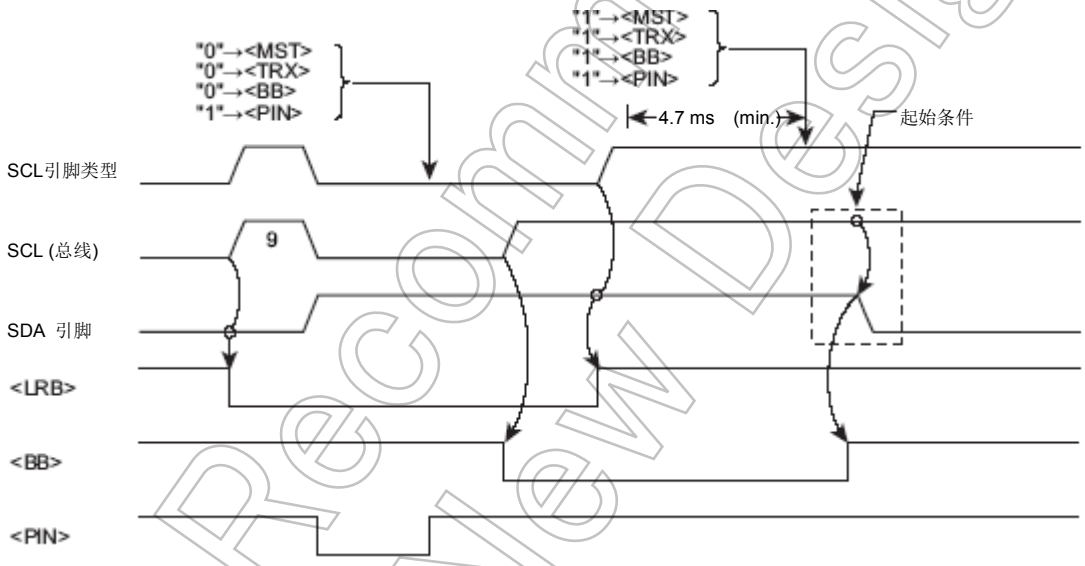


图 14-14 生成一个重新启动的时序图

14.7 SIO模式控制寄存器

以下寄存器控制时钟同步 8-位 SIO 模式中的串行总线接口，并提供监控状态信息。

14.7.1 SBIxCR0 (控制寄存器 0)

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

位	比特符号	型号	功能
31-8	-	R	读作 0。
7	SBIEN	R/W	串行总线接口操作。 0:禁用 1: 启用 在使用串行总线接口之前启动该数位。 如果禁用此位，除SBIxCR0之外所有时钟都停止而降低了功耗。 启用串行总线接口操作然后停止，设置仍然保留在每个寄存器中。
6-0	-	R	读作 0。

14.7.2 SBIxCR1 (控制寄存器 1)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	SIOS	SIOINH	SIOM		-	SCK		
复位后	0	0	0	0	1	0	0	0 (注 1)

位	比特符号	类型	功能
31-8	-	R	读为 0。
7	SIOS	R/W	传输开始/停止 0: 停止 1: 开始
6	SIOINH	R/W	转换 0: 继续 1: 强行终止
5-4	SIOM[1:0]	R/W	选择传输模式 00: 传输模式 01: 保留 10:传输/接受模式 11:接收模式
3	-	R	读为 1。
2-0	SCK[2:0]	R/W	写入<SCK[2:0]>:选择串行时钟频率。 (注1)

000	n = 3	4 MHz
001	n = 4	2 MHz
010	n = 5	1 MHz
011	n = 6	500 kHz
100	n = 7	250 kHz
101	n = 8	125 kHz
110	n = 9	62.5 kHz
111	–	External clock

系统时钟 : fsys (=64MHz)
时钟齿轮 : fc/1
频率 : $\frac{fsys/2}{2^n}$ [Hz]

注 1: **复位后，<SCK[0]> 位读作 “1“。但是，如果在 SBIxCR2 寄存器中选择了 SIO 模式，则初始值读作 “0“。在本文件中，写入“复位后“栏的值为在初始条件下设备 SIO 模式的值。SBIxCR2 寄存器和 SBIxSR 寄存器的描述相同。**

注 2: **在编程传输模式和串行时钟前，设置<SIOS> 为 “0“，<SIOINH>为 “1“。**

14.7.3 SBIxDBR (数据缓冲寄存器)

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

位	比特符号	型号	功能
31-8	-	R	读作 0。
7-0	DB[7:0]	R	接收数据
		W	传输数据

注 1: **该传输数据必须从 MSB (位 7)写入寄存器。接收到的数据存储于 LSB。**

注 2: **由于 SBIxDBR 有独立的读取和写入缓存，因此写入的数据不能被读取。因而不能使用读取-修改-写入指令 (比如位处理等)。**

14.7.4 SBIxCR2 (控制寄存器 2)

写入时，作为 SBIxSR 寄存器使用。

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	-	-	-	-	SBIM		-	-
复位后	1 (注 1)	1 (注 1)	1 (注 1)	1 (注 1)	0	0	1 (注 1)	1 (注 1)

位	比特符号	型号	功能
31-8	-	R	读作 “0“。
7-4	-	R	读作 1。 (注1)
3-2	SBIM[1:0]	W	选择串行总线接口操作模式 (注2) 00: 端口模式 01: SIO模式 10: I2C总线模式 11: 保留
1-0	-	R	读作 1。 (注1)

注 1: **在本文件中，写入“复位后“栏的值为在初始条件下设置的 SIO 模式的值。**

注 2: **应确保在通信会话期间模式没有改变。**

14.7.5 SBIxSR (状态寄存器)

写入时，作为 SBIxCR2 寄存器使用。

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	-	-	-	-	SIOF	SEF	-	-
复位后	1 (注 1)	1 (注 1)	1 (注 1)	1 (注 1)	0	0	1 (注 1)	1 (注 1)

位	比特符号	型号	功能
31-8	-	R	读作 0。
7-4	-	R	读作 1。 (注1)
3	SIOF	R	串行传输状态监控器。 0: 完成 1: 进行中
2	SEF	R	移位操作状态监控器 0: 完成 1: 进行中
1-0	-	R	读作 1。 (注1)

注:**在本文件中，写入"复位后"栏的值为在初始条件下设置的 SIO 模式的值。**

14.7.6 SBIxBR0 (波特率寄存器 0)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	-	I2SBI	-	-	-	-	-	-
复位后	1	0	1	1	1	1	1	0

位	比特符号	型号	功能
31-8	-	R	读作 0。
7	-	R	读作 1。
6	I2SBI	R/W	IDLE模式下运行。 0: 停止 1: 操作
5-1	-	R	读作 1。
0	-	R/W	确保写入 “0“。

14.8 SIO模式下控制

14.8.1 串行时钟

14.8.1.1 时钟源

通过编程 SBIxCR1<SCK[2:0]>，可以选择内部或外部时钟。

(1) 内部时钟

在内部时钟模式下，可以选择 7 个频率中的一个作为串行时钟，通过 SCKx 引脚输出到外部。

在传输开始时，SCKx 引脚输出变为 “高“电平。

如果程序未能与写入传输数据或读取接收数据时的串行时钟保持同步，SBI 自动进入等待期。在此期间，串行时钟自动停止，并且下一次移位操作终止，直到处理程序完成。

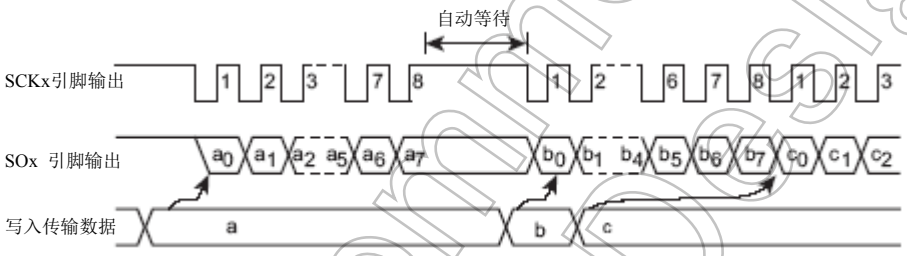


图 14-15 自动等待

(2) 外部时钟 (<SCK[2:0]> = “111“)

SBI 将一外部时钟引到 SCKx 引脚作为一个串行时钟。

为实现正确地移位操作，串行时钟在 “高“ 和 “低“电平时必须有如下所示的脉冲宽度。

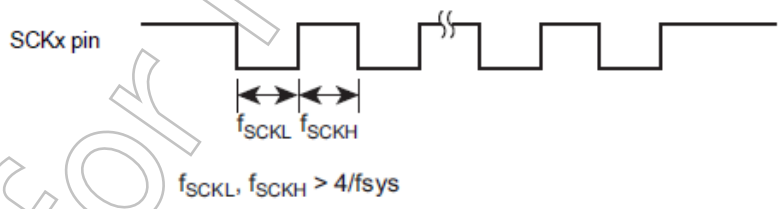


图 14-16 外部时钟输入的最大传输频率

14.8.1.2 移位边缘

传输程序采用上升-沿移位。接收程序采用下降-沿移位。

- 上升-沿移位

  数据在串行时钟的上升沿实现移位 (或者 SCKx 引脚的输入/输出的下降沿)。

- 下降-沿移位

  数据在串行时钟的下降沿实现移位 (或者在 SCKx 引脚的输入/输出的上升沿)。

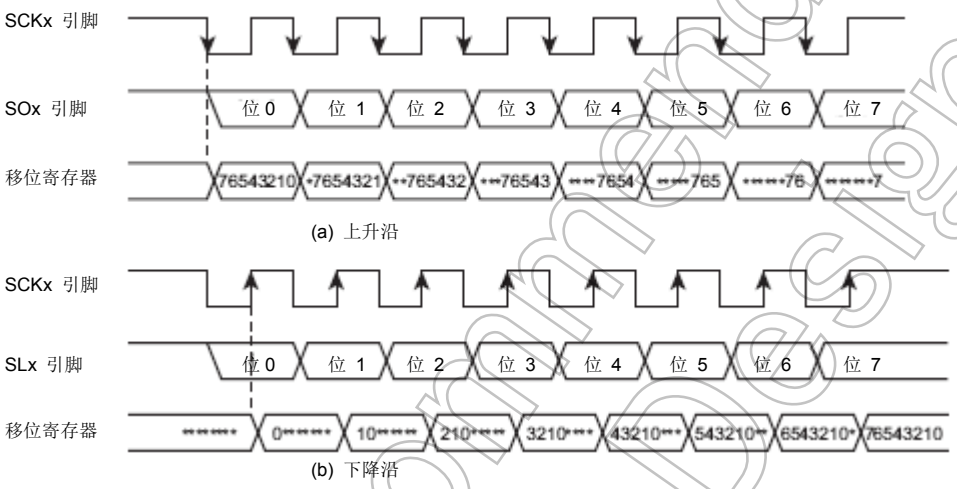


图 14-17 移位边沿

14.8.2 传输模式

通过编程 SBIxCR1<SIOM[1:0]>，可选择传输模式，接收模式或传输/接收模式。

14.8.2.1 8-位传输模式

设置控制寄存器为传输模式并将传输数据写入 SBIxDBR。

在写入传输数据之后，将 "1"写入 SBIxCR1<SIOS>，开始传输。在与最初的最低有效位 (LSB)一致，且与串行时钟同步的情况下，传输数据从 SBIxDBR 转到移位寄存器并且输出到 SO 引脚。一旦传输数据传输 ~ 移位寄存器，SBIxDBR 为空，且 INTSBIx (缓冲区空) 中断生成，请求下组传输数据。

在内部时钟模式下，串行时钟停止，并且如果下组数据在 8 位数据已经全部传输之后还未加载，则自动进入等待模式。当 SBIxDBR 使用下组传输数据加载时，等待命令清除。

在外部时钟模式下，SBIxDBR 必须在下组数据移位操作开始前使用数据进行加载。因此，数据传输率的变化取决于中断请求生成时间和 SBIxDBR 在中断服务编程中使用数据加载之间的最大等待时间。

开始传输时，在设置 SBIxSR<SIOF>为 "1"到 SCK 下降沿期间，输出一个与先前传输的数据的最后位相同的值。

可通过清除<SIOS>为 "0" 或在 IN-TSBIx 中断服务编程中设置<SIOINH> 为 "1"来终止传输。如果<SIOS>清除，剩余数据在传输结束前输出。编程检查 SBIxSR<SIOF>以判断传输是否必须结束。传输结束时，<SIOF>清 "0"。如果<SIOINH>设置到 "1"，立即放弃传输，且<SIOF>清 "0"。

外部时钟模式下，在下组数据移位之前，<SIOS>必须清 "0"。如果 <SIOS> 在下组数据移位前被清 "0"，SBI 输出虚假数据并停止。

		7	6	5	4	3	2	1	0	
SBIxCR1	←	0	1	0	0	0	X	X	X	选择传输模式。
SBIxDBR	←	X	X	X	X	X	X	X	X	写入传输数据。
SBIxCR1	←	1	0	0	0	0	X	X	X	开始传输。

INTSBIx 中断

SBIxDBR	←	X	X	X	X	X	X	X	X	写入传输数据。

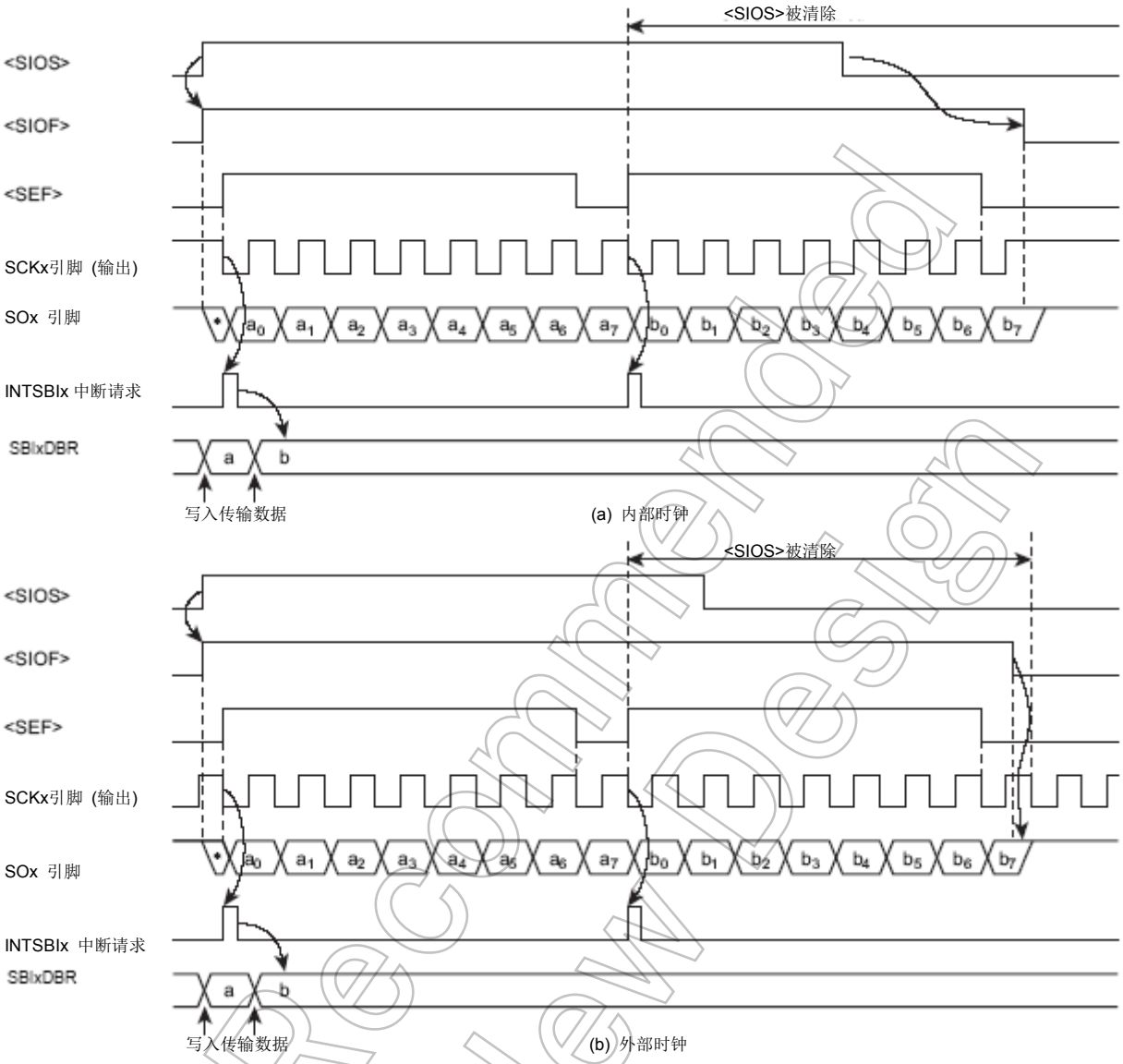


图 14-18 传输模式

示例:通过<SIO> 编程 (外部时钟)终止传输示例

```
                    7   6   5   4   3   2   1   0
如果 SBIxSR<SIOF> ≠ 0                          识别传输完成情况。
然后
如果SCK ≠ 1                                    识别 "1" 通过监控端口设置到SCK引脚。
然后
SBIxCR1     ←       0   0   0   0   0   1   1   1   通过设置<SIOS> = 0完成传输。
```

14.8.2.2 8-位接收模式

把控制寄存器设置为接收模式。然后把 “1“写入 SBIxCR1 <SIOS>启动接收。在与最初的最低有效位 (LSB)一致，且与串行时钟同步的情况下，数据从 SI 引脚移入移位寄存器。一旦移位寄存器加载 8-位数据，所接收到的数据传送到 SBIxDBR，且 INTSBIx (缓冲区满)中断请求产生，请求读出接收到的数据。然后中断服务程序读取从 SBIxDBR 接收到的数据。

在内部时钟模式下，串行时钟停止并自动处于等待条件，直到从 SBIxDBR 读取接收到的数据。

在外部时钟模式下，移位操作与外部时钟同步执行。最大数据传输率不同，取决于产生中断请求和读取所接收的数据之间最大等待时间。

接收可以通过清除<SIOS>为 “0“或在 INTSBIx 中断服务程序中设置<SIOINH>为 “1“来终止。如果<SIOS>被清除，接收继续进行，直到接收到的数据的所有位被写入 SBIxDBR。程序检查 SBIxSR<SIOF>以判定接收是否已经终止。接收结束时，<SIOF>被清 “0“。确认接收完成后，读出最后接收到的数据。当<SIOINH>设置到 “1“时，接收立即中止，<SIOF>被清 “0“。 (接收到的数据将无效，不需要读取)

注:传输模式改变后，SBIxDBR 的内容将不保留。必须通过清除<SIOS>为 “0“来完成正在进行的接收，而最后接收到的数据必须在传输模式改变之前读取。

		7	6	5	4	3	2	1	0	
SBIxCR1	←	0	1	1	1	0	X	X	X	选择接收模式。
SBIxCR1	←	1	0	1	1	0	X	X	X	开始接收。

INTSBIx 中断

Reg.	←	SBIxDBR	读取接收数据。

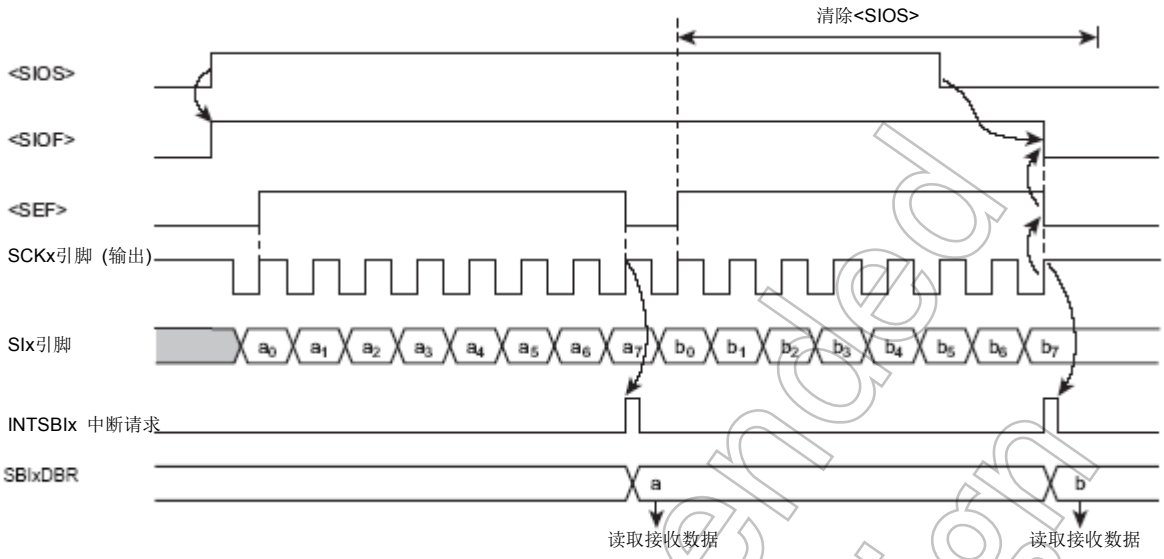


图 14-19 接收模式 (示例:内部时钟)

14.8.2.3 8位传输/接收模式

设置控制寄存器为传输/接收模式。然后写入传输数据到 SBIxDBR 并设置 SBIxCR1 <SIOS> 到“1“开始传输和接收。传输数据在串行时钟下降沿通过 Sox 引脚输出，并在串行时钟上升沿，且与最初的最低有效位一致时，通过 SI 引脚 (LSB)来接收收到的数据。一旦移位寄存器加载 8-位数据，它所接收到的数据传输到 SBIxDBR 并 INTSBIx 产生中断请求。中断服务程序从数据缓存寄存器中读取数据，并写入下一组传输数据。因为 SBIxDBR 在传输和接收操作之间共享，所以接收到的数据必须在下组传输数据写入前读取。

在内部时钟操作模式下，串行时钟将自动进入等待状态，直到读取接收到的数据，并写入下组传输的数据。

在外部时钟模式下。移位操作与外部串行时钟同步执行。因此，必须读取接收到的数据，且下组传输数据必须在下组移位操作开始前写入。外部时钟操作的最大数据传输率的变化取决于中断请求产生和写入传输数据之间的最大等待时间。

传输开始时，在将<SIOF>设置到 “1“到 SCK 的下降沿期间，输出与先前传输数据的最后位相同的值。

可通过<SIOS>清除为或在 INTSBIx 中断服务程序中设置 SBIxCR1<SIOINH>为 “1“来终止传输和接收。当<SIOS>被清除时，继续传输和接收，直到接收到的数据完全传送到 SBIxDBR。程序检查 SBIxSR<SIOF>，判定传输和接收是否都已终止。在传输和接收结束时，<SIOF>被清为 “0“。当<SIOINH>被设置到 “1“时，传输和接收立即终止且<SIOF>被清 “0“。

注:**传输模式改变后，SBIxDBR 的内容将不保留。必须通过<SIOS>清 “0“来完成正在进行的传输和接收，并且最后接收到的数据必须在传输模式改变之前读出。**

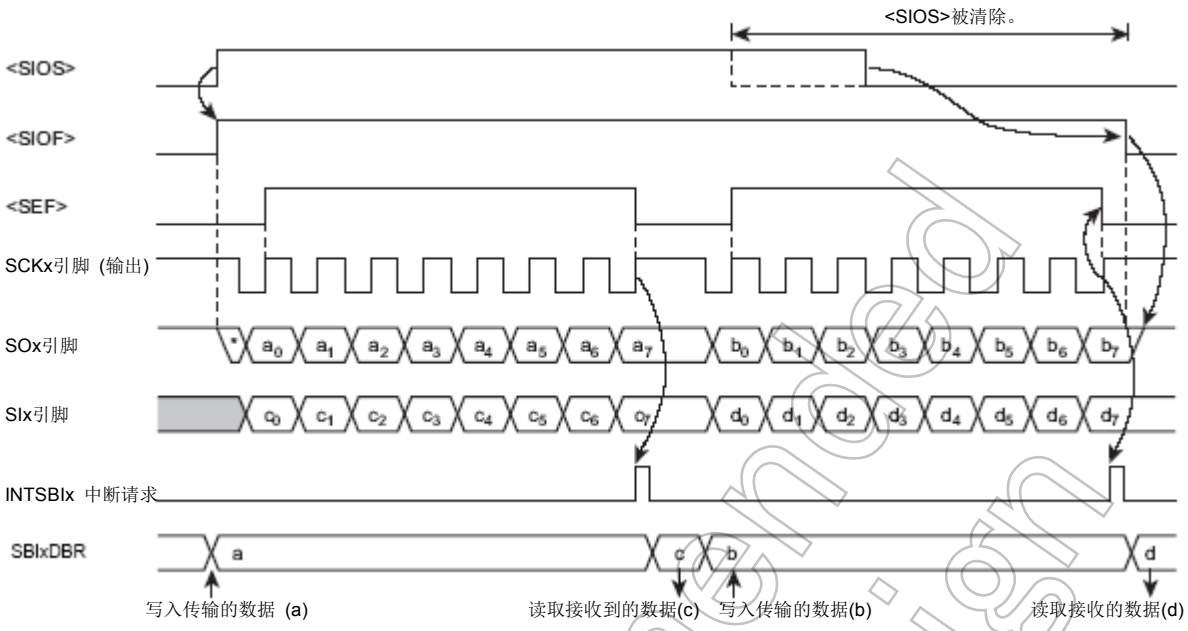


图14-20 传输/接收模式 (示例:内部时钟)

		7	6	5	4	3	2	1	0	
SBIxCR1	←	0	1	1	0	0	X	X	X	选择传输模式。
SBIxDBR	←	X	X	X	X	X	X	X	X	写入传输数据。
SBIxCR1	←	1	0	1	0	0	X	X	X	开始接收/传输。

INTSBIx中断

Reg.	←	SBIxDBR								读取接收数据。
SBIxDBR	←	X	X	X	X	X	X	X	X	写入传输数据。

14.8.2.4 传输结束最后位数据保留时间

在 SBIxCR1<SIOS>= “0“条件下，传输数据的最末位保留如下所示的 SCK 上升沿的数据。传输模式和传输/接收模式相同。

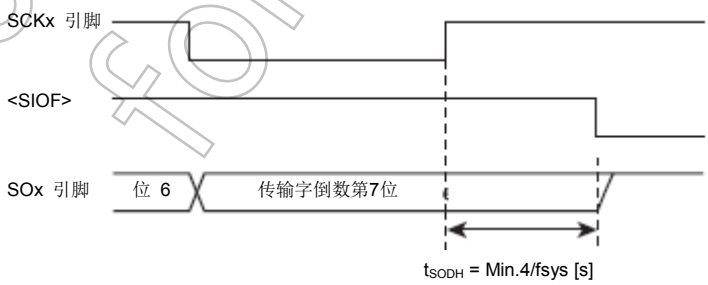


图14-21 传输结束时最末位的数据保留时间

15. 消费性电子产品控制 (CEC)

15.1 概述

CEC 功能传输和接收数据，符合消费性电子控制 (以下简称 CEC)协议。

适合 HDMI 1.3a 接口规范。

15.1.1 接收

- 全屏时 16 位定时器的时钟取样/事件计数器
  - 可调噪声消除时间
- 每 1 字节数据接收
  - 弹性数据取样点
  - 计时当检测出某一地址不一致时也接收数据。
- 误差检测
  - 周期误差 (min./max.)
  - ACK 冲突
  - 波形误差

15.1.2 传输

- 每 1 字节数据传输
  - 通过总线自由状态下自动检测触发
- 弹性波形
  - 可调节上升沿和周期
- 误差检测
  - 调停丢失
  - ACK 响应误差

15.1.3 注意事项

当逻辑地址不符时启用接收数据时 (CECRCR1<CECOTH> = “1“)，若启动程序发送的新信息以起始位开始，却未发送含 EOM= “1“的最后一个模块时，即可确定 ACK 位的最大周期误差，并生成中断。接着，信息接收就可以正常进行了。

15.2 方块图

图 15.1 为 CEC 的方块图

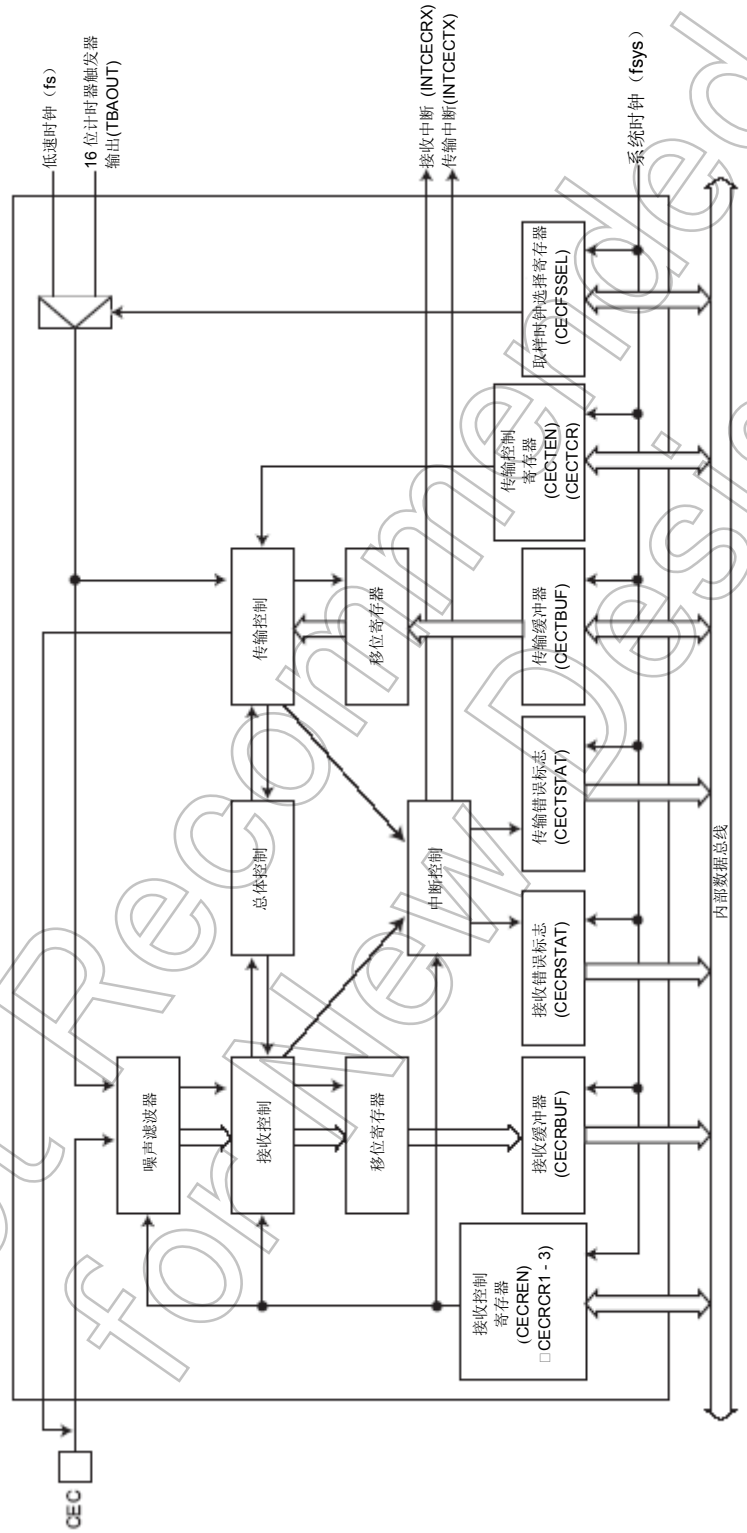


图 15.1 CEC 方方块图

15.3 寄存器

15.3.1 寄存器列表

CEC 控制寄存器和地址见下表。

基址=0x400E_2000

寄存器		地址 (基+)
CEC启用寄存器	CECEN	0x0000
逻辑地址寄存器	CECADD	0x0004
软件复位寄存器	CECRESET	0x0008
接收启用寄存器	CECREN	0x000C
接收缓冲寄存器	CECRBUF	0x0010
接收控制寄存器 1	CECRCR1	0x0014
接收控制寄存器 2	CECRCR2	0x0018
接收控制寄存器 3	CECRCR3	0x001C
传输启用寄存器	CECTEN	0x0020
传输缓冲寄存器	CECTBUF	0x0024
传输控制寄存器	CECTCR	0x0028
接收中断状态寄存器	CECRSTAT	0x002C
传输中断状态寄存器	CECTSTAT	0x0030
CEC取样时钟优选寄存器	CECFSSEL	0x0034

15.3.2 CECEN (CEC启用寄存器)

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

位特	比特符号	型号	功能
31-3	-	R	读作 0。
2	-	R/W	写入 “0“。
1	-	R/W	写入数值为 “1“。
0	CECEN	R/W	CEC 操作 0: 禁用 1: 启用 规定CEC操作。 使用之前启用CEC。 CEC操作禁用时，除CECEN寄存器外，CEC模块无时钟。 这样可降低功耗。 如果CEC启用后禁用，需保留每一个寄存器组。

15.3.3 CECADD (逻辑地址寄存器)

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

位	比特符号	型号	功能
31-16	-	R	读作 0。
15-0	CECADD[15:0]	R/W	逻辑地址15 ~ 0 规定分配给CEC的逻辑地址。由于每一个位对应一个地址，可同时设置多个地址。

注意:无论寄存器如何设置，总能收到广播信息。将逻辑地址分配给 15，将逻辑 “0“作为 ACK 响应应答发送给广播信息。

15.3.4 CECRESET (软件复位寄存器)

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

位	比特符号	型号	功能
31-1	-	R	读作 0
0	CECRESET	W	软件复位 0: 禁用 1: 启用 停止所有CEC操作并对寄存器进行初始化。 将该位设置到 “1“会导致: 接收:立刻停止。已接收数据丢失。 传输 (包括CEC线路) 立刻停止。 寄存器:除CECEN外，对所有的寄存器进行初始化. 读作 0。

15.3.5 CECREN (接收启用寄存器)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	-	-	-	-	-	-	-	CECREN
复位后	0	0	0	0	0	0	0	未定义

位	比特符号	型号	功能
31-1	-	R	读作 0
0	CECREN	R/W	接收控制 [写] 0: 禁用 1:启用 [读] 0: 停止 1: 操作中 控制CEC接收。 对该位写入 “0“或者 “1“启用或禁止接收数据。写入 “1“，该位可接收数据。 读该位可以监控接收回路状态。可以检查设置是否正确。

注 1: 设置 CECRCR1，CECRCR2 和 CECRCR3 后，启用<CECREN>位。

注 2: 花很少时间将<CECREN>。位的设置反映到回路。

15.3.6 CECRBUF (接收缓冲寄存器)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	CECACK	CECEOM
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	CECRBUF							
复位后	0	0	0	0	0	0	0	0

位	比特符号	型号	功能
31-10	-	R	读作 0。
9	CECACK	R	ACK 位 读取接收到的ACK 位。
8	CECEOM	R	EOM 位 读取接收到的EOM 位。
7-0	CECRBUF[7:0]	R	接收到的数据 读取收到的一个位数据。位7为MSB。

注 1: 对寄存器进行写入操作无效。

注 2: 一旦发生接收中断，立即读寄存器。随后的读取数据可能不安全。

15.3.7 CECRCR1 (接收控制寄存器 1)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	CECACKDIS
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	CECHNC		-	CECLNC		
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	CECMIN			-	CECMAX		
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	-	CECDAT			CECTOUT		CECRIHLD	CECOTH
复位后	0	0	0	0	0	0	0	0

位	比特符号	型号	功能
31-25	-	R	读作 0。
24	CECACKDIS	R/W	逻辑 “0“作为ACK应答 0: 发送 1: 不发送 规定了地址与逻辑地址中设置的地址一致时，是否发送逻辑 “0“作为ACK对数据块的应答。 (不管位设置如何，检测地址一致性时为ACK应答，发送逻辑 “0“到标题块作对应的地址)
23-22	-	R	读作 0。
21-20	CECHNC[1:0]	R/W	取消噪声取样 “高 “的数目 00:无 (观察fs时钟一次) 01: 1/fs (观察两次连续fs时钟。) 10: 2/fs (观察三次连续fs时钟。) 11: 3/fs (观察四次连续fs时钟。) 规定当检测到 “高“时，规定每个取样周期噪声消除的时间为1/fs。 如果没有对与规定周期数量相同的 “高“进行取样，将被视作噪音
19	−	R	读作 0 。
18-16	CECLNC[2:0]	R/W	取消噪声取样 “低“ 的数目 000:无 (观察全屏时钟一次) 100: − (保留) 001: 1/fs (观察两次连续fs时钟) 101: − (保留) 010: 2/fs (观察三次连续fs时钟) 110: − (保留) 011: 3/fs (观察四次连续fs时钟) 111: − (保留) 当检测到 “低“时，规定每个取样周期噪声消除的时间为1/fs。 若没有对与规定周期数量相同的 “低“取样，将被视作噪声。
15	-	R	读作 0。
14-12	CECMIN[2:0]	R/W	确定最小周期误差时间 000: 67/fs (约 2.045ms) 100: 67/fs − 1/fs 001: 67/fs + 1/fs 101: 67/fs − 2/fs 010: 67/fs + 2/fs 110: 67/fs − 3/fs 011: 67/fs + 3/fs 111: 67/fs − 4/fs 规定确定一个有效位所需的最少时间。 基准时间为67/fs (约 2.045) ms。在−4/fs ~ +3/fs范围内用一个取样周期对其进行规定。 当一个位周期位规定时间短时会出现中断，并且向CEC输出 ”低” 约3.63ms的时间。 超过了规定的时间。
11	−	R	读作 0。

位	比特符号	型号	功能
10-8	CECMAX[2:0]	R/W	确定最大周期的误差时间。 000: 90/fs (约 2.747ms) 100: 90/fs − 1/fs 001: 90/fs + 1/fs 101: 90/fs − 2/fs 010: 90/fs + 2/fs 110: 90/fs − 3/fs 011: 90/fs + 3/fs 111: 90/fs − 4/fs 规定确定一个有效位所需的最长的时间。 基准时间为90/fs (约 2.747ms)。在−4/fs ~ +3/fs范围内用一个采样周期对其进行规定。 当一个位周期位超过规定时间短时会出现中断。
7	−	R	读作 0。
6-4	CECDAT[2:0]	R/W	判定数据点为0或1。 000: 34/fs (约1.038ms) 100:34/fs − 2/fs 001: 34/fs + 2/fs 101:34/fs − 4/fs 010: 34/fs + 4/fs 110:34/fs − 6/fs 011: 34/fs + 6/fs 111:保留 规定判定数据点为逻辑 “0“或 “1“。 基准时间基准时间为34/fs (约1.038 ms)。用2/fs单位在±6/fs范围内对其进行说明。
3-2	CECTOUT[1:0]	R/W	确超时的周期 00: 1 位周期 01: 2 位周期 10: 3 位周期 11: 保留 规定判定超时时间。规定判定数据点为逻辑 “0“或 “1“。 当<CECRIHLD>位有效时，该设置用来检测一个超时。
1	CECRIHLD	R/W	误差中断挂起 0: 不挂起 1: 挂起 说明是否需要挂起接受误差中断 (最大周期误差，缓冲超限和波形) 错误 误差检测时设置 “1“不中断。.若数据继续向一个ACK位发送，用一个反向逻辑执行ACK应答。如果随后的位发生中断，在<CECTOUT>条件下其会被确定为一个超时。 ACK应答或超时确定后，会产生中断。
0	CECOTH	R/W	逻辑地址不符时的数据接收 0: 未收到 1: 收到 指定当目标地址与CECADD寄存器中的地址不符时，是否要接收数据。

注 1: <CECHNC>, <CECLNC> 和<CECDAT>条件下的设置也用于接收传输中的 ACK 应答。

注 2: 传输或接收过程中改变配置可能会对正常操作造成影响。改变之前，要设置 CECREN <CECREN> 位禁止接收或对<CECREN>位和 CECTEN <CECTRANS> 位进行读操作，确保操作已经停止。

注 3: 无论<CECOTH>寄存器如何设置，总会接收到广播信息。

注 4: <CECLNC>必须和 CECTCR<CECDTRS>的设置相同。

15.3.8 CECRCR2 (接收控制寄存器 2)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	CECSWAV3			-	CECSWAV2		
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	-	CECSWAV1			-	CECSWAV0		
复位后	0	0	0	0	0	0	0	0

位	比特符号	型号	功能
31-15	-	R	读作 0。
14-12	CECSWAV3[2:0]	R/W	检测起始位最大周期。 000: 154/fs (约 4.700 ms) 100: 154/fs + 4/fs 001: 154/fs + 1/fs 101: 154/fs + 5/fs 001: 154/fs + 2/fs 110: 154/fs + 6/fs 011: 154/fs + 3/fs 111: 154/fs + 7/fs
11	-	R	读作 0。
10-8	CECSWAV2[2:0]	R/W	检测起始位最小周期。 000: 141/fs (约 4.303 ms) 100:141/fs − 4/fs 001: 141/fs − 1/fs 101:141/fs − 5/fs 001: 141/fs − 2/fs 110:141/fs − 6/fs 011: 141/fs − 3/fs 111:141/fs − 7/fs
7	-	R	读作 0。
6-4	CECSWAV1[2:0]	R/W	起始位上升时间的最大值。 000: 128/fs (约 3.906 ms) 100:128/fs + 4/fs 001: 128/fs + 1/fs 101:128/fs + 5/fs 001: 128/fs + 2/fs 110:128/fs + 6/fs 011: 128/fs + 3/fs 111:128/fs + 7/fs
3	-	R	读作 0。
2-0	CECSWAV0[2:0]	R/W	起始位起始时间的最小值。 000: 115/fs (约 3.510 ms) 100:115/fs − 4/fs 001: 115/fs − 1/fs 101:115/fs − 5/fs 001: 115/fs − 2/fs 110:115/fs − 6/fs 011: 115/fs − 3/fs 111:115/fs − 7/fs

<CECSWAV3>: 规定检测一个起始位的周期。

<CECSWAV2>: <CECSWAV3>为最大周期数。基准时间为 154/fs (约 4.700 ms)。以一个取样周期为单位在 0 ~ +7/fs 之间对其进行说明。

<CECSWAV2>为最小周期数。基准时间为 141/fs (约 4.303 ms)。以一个取样周期为单位在 0 ~ -7/fs 范围内对其进行说明。

<CECSWAV1>: 检测过程中对起始位的上升时间进行说明。

<CECSWAV0>: <CECSWAV1>为上升时间最大值。基准时间为 128/fs (约 3.906 ms) 。以一个取样周期为单位在 0 ~ +7/fs 范围内对其进行说明。

<CECSWAV0>为上升时间最小值。基准时间为 115/fs (约 3.510 ms)。在 0 ~ -7/fs 范围内用一个采样周期对其进行规定。

注:接收数据时改变配置可能会影响正常操作。改变前，设置 CECREN <CECREN>禁止接收和读取<CECREN>位，以确保操作停止。

15.3.9 CECRCR3 (接收控制寄存器 3)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	CECSWAV3			-	CECSWAV2		
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	CECSWAV1			-	CECSWAV0		
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	-	--	-	-	-	-	-	CECWAVEN
复位后	0	0	0	0	0	0	0	0

位	比特符号	型号	功能
31-23	−	R	读作 0。
22-20	CECWAV3[2:0]	R/W	逻辑 “0“最近的上升时间判定为正常波形。 000: 56/fs (约 1.709 ms) 100: 56/fs + 4/fs 001: 56/fs + 1/fs 101 :56/fs + 5/fs 001: 56/fs + 2/fs 110: 56/fs + 6/fs 011: 56/fs + 3/fs 111: 56/fs + 7/fs
19	−	R	读作 0。
18-16	CECWAV2[2:0]	R/W	逻辑 “0“最快上升时间判定正常。 000: 43/fs (约1.312 ms) 100: 43/fs − 4/fs 001: 43/fs − 1/fs 101: 43/fs − 5/fs 010: 43/fs − 2/fs 110: 43/fs − 6/fs 011: 43/fs − 3/fs 111: 43/fs − 7/fs
15	−	R	读作 0。
14-12	CECWAV1[2:0]	R/W	逻辑 “1“最近上升时间判定为正常波形。 000: 26/fs (约 0.793 ms) 100: 26/fs + 4/fs 001: 26/fs + 1/fs 101: 26/fs + 5/fs 001: 26/fs + 2/fs 110:2 6/fs + 6/fs 011: 26/fs + 3/fs 111: 26/fs + 7/fs
11	−	R	读作 0
10-8	CECWAV0[2:0]	R/W	逻辑 “1“最快上升时间判定为正常。 000: 13/fs (约 0.396 ms) 100: 13/fs − 4/fs 001: 13/fs − 1/fs 101: 13/fs − 5/fs 010: 13/fs − 2/fs 110: 13/fs − 6/fs 011: 13/fs − 3/fs 111: 13/fs − 7/fs
7-2	−	R	读作 0。
1	CECRSTAEN	R/W	起始位检测。 1: 启用 0: 禁用 发现起始位接收，产生中断。
0	CECWAVEN	R/W	波形误差检测 1: 启用 0: 禁用 检测到已接收波形不同于已确认波形，会产生波形误差中断。 如已被启用，根据<CECWAV0> <CECWAV1> <CECWAV2> <CECWAV3>的设置会检测到误差。

<CECWAV3>: 如果<CECWAVEN>位设置到“1“，该设置已被启用。

通过对这些位设置，一旦接收到的波形上升沿位正常逻辑 “0“的上升沿来得晚，就可以检测到误差。基准时间为 56/fs (约 1.709ms)。

启用以将其规定在 1/fs 的 0 ~ +7/fs。如果从位的起始点到<CECWAV3>中的规定值都未检测到上升沿，接收到的波形被视为误差

<CECWAV2>: 当<CECWAVEN>位设置到“1“时，启用该设置。

<CECWAV1>: 通过对这些位的设置，如果已接收波形的上升沿位逻辑 “0“来得快，而位正常逻辑 “1“上升沿来得晚，即可检测到误差。

<CECWAV1>位的基准时间为 26/fs (约 0.793ms))。启用以将其规定在 1/fs 的 0 ~ +7/fs。

<CECWAV2>位的基准时间为 43/fs (约 1.312ms)。以一个取样周期为单位在 0 ~ −7/fs 之间对其进行说明。

如果<CECWAV2>位 and <CECWAV1>位之间检测到上升沿，误差产生。

<CECWAV0>: 当<CECWAVEN>位设置到“1“时，启用该设置。

通过对这些位的设置，如果已接收波形的上升沿位正常逻辑 “1“的波形来得快，便可检测到误差。

基准时间为 13/fs (约 0.396ms) 启用以将其规定在 1/fs 的 0 ~ −7/fs。

如果从位的起始点到<CECWAV0>中的规定值都未检测到上升沿，接收到的波形被视为误差.

注:接收数据时改变配置可能会影响正常操作。改变前，设置 CECREN <CECREN>禁止接收和读取<CECREN>位，以确保操作停止。

15.3.10 CECTEN (传输启用寄存器)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	-	-	-	-	-	-	CECTRANS	CECTEN
复位后	0	0	0	0	0	0	0	未定义

位	比特符号	型号	功能
31-2	−	R	读作 0。
1	CECTRANS	R	传输状态 0: 未进行 1: 进行中 说明传输是否正在进行。 启动起始位传输时显示 “1“。传输结束或产生中断时显示 “0“。 对该位进行写入操作被忽略。
0	CECTEN	W	传输控制 0: 禁用 1: 启用 控制CEC传输。 写入该位被启用或禁止传输。对该位写入 “1“开始传输。传输结束中断或误差中断，该位自动清除。

注 1: 设置 CECTBUF 和 CECTCR 寄存器后设置<CECTEN>。

注 2: 改变设置或启用传输和接收前，停止传输和接收。

15.3.11 CECTBUF (传输缓冲寄存器)

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

位	比特符号	型号	功能
31-9	-	R	读作 0。
8	CECTEOM	R/W	EOM位指定EOM位进行传输。
7-0	CECTBUF[7:0]	R/W	传输数据 指定一个位数据进行传输。位7为MSB。

15.3.12 CECTCR (传输控制寄存器)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	CECSTRS			-	CECSPRD		
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	CECDTRS			CECDPRD			
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	-	-	-	CECBRD	CECFREE			
复位后	0	0	0	0	0	0	0	0

位	比特符号	型号	功能
31-23	−	R	读作 0。
22-20	CECSTRS[2:0]	R/W	起始位上升时间。 000: 基准时间　100:基准时间− 4/fs 001: 基准时间− 1/fs　101:基准时间− 5/fs 010: 基准时间− 2/fs　110:基准时间− 6/fs 011: 基准时间− 3/fs　111:基准时间− 7/fs 规定起始位上升时间。 基准时间为121/fs (约3.693ms)。启用以将其规定在1/fs的0 ~ −7/fs。
19	−	R	读作 0。
18-16	CECSPRD[2:0]	R/W	起始位周期 000: 基准时间　100: 基准时间− 4/fs 001: 基准时间− 1/fs　101: 基准时间− 5/fs 010: 基准时间− 2/fs　110: 基准时间− 6/fs 011: 基准时间− 3/fs　111: 基准时间− 7/fs 规定一个起始位周期 基准时间为147/fs (约 4.486 ms)。启用以将其规定在1/fs的0 ~ −7/fs。
15	−	R	读作 0。
14-12	CECDTRS[2:0]	R/W	数据位上升时间。 000: 基准时间　100: 保留 001: 基准时间− 1/fs　101: 保留 010: 基准时间− 2/fs　110: 保留 011: 基准时间− 3/fs　111: 保留 规定一个数据位的上升时间。 基准时间为 20/fs (约0.610 ms，逻辑为 “1“)或者 49/fs (约1.495 ms, 逻辑为 “0 “)。以一个取样周期为单位在0 ~ -3/fs之间，启用说明。
11-8	CECDPRD[2:0]	R/W	数据位周期 0000: 基准时间　1000:基准时间− 8/fs 0001: 基准时间− 1/fs　1001:基准时间− 9/fs 0010: 基准时间− 2/fs　1010:基准时间− 10/fs 0011: 基准时间− 3/fs　1011:基准时间− 11/fs 0100: 基准时间− 4/fs　1100:基准时间− 12/fs 0101: 基准时间− 5/fs　1101:基准时间− 13/fs 0110: 基准时间− 6/fs　1110:基准时间− 14/fs 0111: 基准时间− 7/fs　1111:基准时间− 15/fs 指定一个数据位周期. 基准时间为 79/fs (约2.411 ms)。以一个取样周期为单位在0 ~ -15/fs对其进行说明。

位	比特符号	型号	功能
7-5	−	R	读作 0。
4	CECBRD	R/W	广播传输 0: 无广播传输 1: 广播传输 传输广播信息时对该位设到“1“。
3-0	CECFREE[3:0]	R/W	总线空闲时间 0000: 1位周期　1000: 9位周期 0001: 2位周期　1001: 10位周期 0010: 3位周期　1010: 11位周期 0011: 4位周期　1011: 12位周期 0100: 5位周期　1100: 13位周期 0101: 6位周期　1101: 14位周期 0110: 7位周期　1110: 15位周期 0111: 8位周期　1111: 16位周期 指定传输前检查的总线空闲时间。 在规定的周期内，确保CEC总线处于闲置状态，之后开始传输。

注:使用<CECDTRS>时，其设置必须与 CECRCR1<CECLNC>相同。

15.3.13 CECRSTAT (接收中断状态寄存器)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	-	CECRIWAV	CECRIOR	CECRIACK	CECRIMIN	CECRIMAX	CECRISTA	CECRIEND
复位后	0	0	0	0	0	0	0	0

位	比特符号	型号	功能
31-7	−	R	读作 0。
6	CECRIWAV	R	中断标志 0: 无波形误差 1: 波形误差 表示检测到波形误差。 在CECRCR3 <CECWAVEN>条件下，波形误差检测启用后，出现误差。
5	CECRIOR	R	中断标志 0: 无接收缓冲超限 1: 接收缓冲超限 数据读取已设定前，显示接收缓冲区收到下一个数据。
4	CECRIACK	R	中断标志 0: 无ACK冲突 1: ACK冲突 在指定时间输出ACK位后，表示检测到 “0“。
3	CECRIMIN	R	中断标志 0: 无最小周期误差 1: 最小周期误差 表示一个位周期位CECRCR1<CECMIN>条件下规定的最小周期误差检测时间短。
2	CECRIMAX	R	中断标志 0: 无最大周期误差 1: 最大周期误差 表示一个位周期位CECRCR1<CECMAX>条件下规定的最大周期误差检测时间长。
1	CECRISTA	R	中断信号 0: 无起始位检测 1: 起始位检测 表示检测到起始位。
0	CECRIEND	R	中断标志 0: 未完成一字节数据接收 1: 完成一字节数据接收 表示完成一位数据接收。

注:对该位进行写入操作无效。

15.3.14 CECTSTAT (传输中断状态寄存器)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	-	-	-	CECTIUR	CECTIACK	CECTIAL	CECTIEND	CECTISTA
复位后	0	0	0	0	0	0	0	0

位	比特符号	型号	功能
31-5	−	R	读作 0。
4	CECTIUR	R	中断标志 0: 无传输缓冲 1: 传输缓冲 表示一个位数据传输中下一个数据未到传输缓冲区。
3	CECTIACK	R	中断标志 0: 无ACK误差检测 1: ACK误差检测 表示出现其中以下一种情况。 向广播地址 传输时未检测到逻辑0 ・向广播地址传输时未检测到逻辑“1“。
2	CECTIAL	R	中断标志 0:无忠诚丢失 1:仲裁丢失出现 当输出“高“时，表示检测到“低“。
1	CECTIEND	R	中断标志 0: 未完成数据传输 1: 完成数据传输 完成包括EOM位的数据传输。
0	CECTISTA	R	中断标志 0: 未开始传输 1: 开始传输 表示开始一字节数据传输。

注:对该位进行写入操作无效。

15.3.15 CECFSSEL (取样时钟选择寄存器)

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

位	比特符号	型号	功能
31-1	−	R	读作 0。
0	CECCLK	R/W	CEC取样时钟 0: 低速时钟 (fs) 1: TBAOUT 将取样时钟设到CEC功能。启用选择低速时钟或CEC取样时钟的计时器输出。设定TBAOUT，使计时器输出范围在30kHz ~ 34kHz之间。

注:通过 CECFSSEL 寄存器改变取样时钟时，通过 CECEN<CECEN>寄存器停止 (禁止)CEC 操作一次。再次启动 (允许)CEC 操作后首先设定 CECFSSEL 寄存器，再设置其他与 CEC 相关的寄存器。若通过 CECRESET 寄存器进行软件复位，则需在改变取样时钟时首先设定 CECFSSEL 寄存器，再对其他与 CEC 有关的寄存器进行设定。

15.4 操作

15.4.1 取样时钟

用 32.768kHz 的低速时钟 (fs)或者 16 位计时器/事件计数器输出的 TBAOUT 对 CEC 行进行取样。

取样时钟可与 CECFSSEL 寄存器的<CECCLKC>位一起配置。

15.4.2 接收

15.4.2.1 基本操作

当检测到起始位时，生成一个起始位中断。通过生成起始位中断来设置 CECR-STAT<CECRISTA>。CECRCR3<CECRSTA>设置到 “1“时，起始位中断生成。

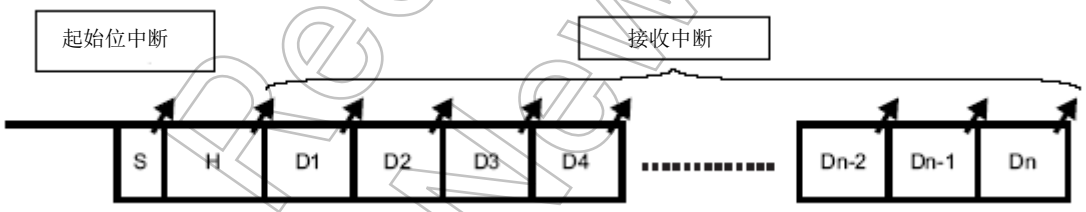
当接收到一位数据，EOM 位和 ACK 位时，接收数据储存在 CECRBUF 寄存器中，接收中断出现。通过生成接收中断设置 CECRSTAT<CECRIEND>。

在 CECRBUF 寄存器中存储 8 位数据，EOM 位及 ACK 位存。ACK 位不会在 CEC 回路内部生成。同其他数据一样，该位通过观察 CEC 信号生成。

接收到一个数据块后，继续进行接收，直到检测到含有设置到 “1“的 EOM 位的最后一个数据块。检测完最后一个数据块，CEC 变成起始位等待模式。

接收数据时检测到误差会导致一个误差中断，CEC 就会等待下一个起始位。已接收数据丢失。

注:数据接收详见 “15.1.3 注意事项“。



15.4.2.2 预配置

接收数据前，需要对逻辑地址寄存器<CECADD>，接收控制寄存器 1 <CECRCR1>，接收控制寄存器 2<CECRCR2> 及接收控制寄存器 3 <CECRCR3>进行接收设置。

(1) 逻辑地址配置

对分配到该产品的 CECADD 寄存器的逻辑地址进行配置。由于该寄存器里的每个位都有对应地址，故多个地址可同时配置。

注:无论怎样对 CECADD 寄存器进行配置都不影响接收广播信息。将逻辑地址分配给 15，将逻辑 “0“作为 ACK 响应应答发送给广播信息。

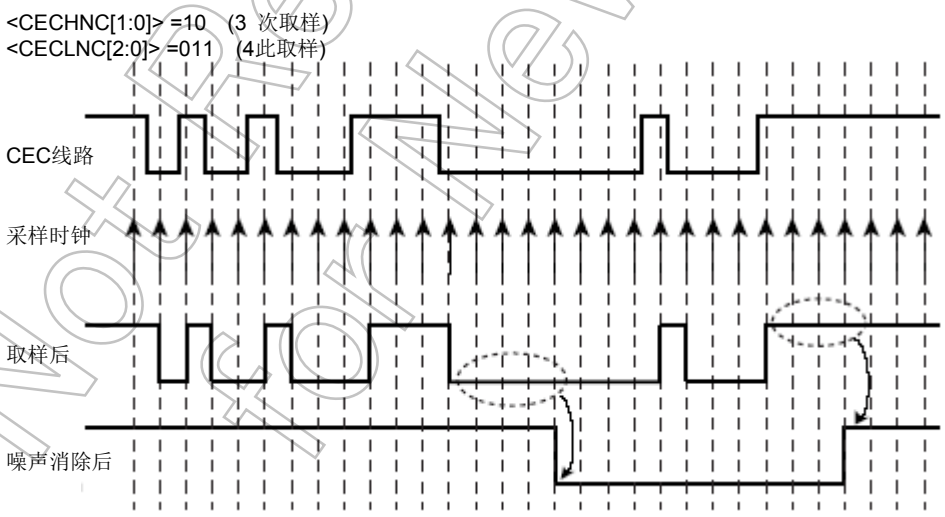
(2) 噪声消除时间

噪声消除时间可与 CECRCR1 寄存器里的<CECHNC> 和 <CECLNC>位一起配置。如果未对与规定值相同数量的 “高“或 “低“进行取样，则被视作噪声。可检测到 “高“和 “低“的时间分别进行配置。

在取样时钟的任何一个上升沿监控 CEC 线路。如果 CEC 线路由 “高 “向 “低 “转变，且对<CECLNC> 位里规定的相同数量监测为 “低“，将完全识别该变化。如果 CEC 线路由 “低 “向 “高“转变，而<CECHNC>位里规定的相同数量取样为 “高“，将完全识别该变化。

注:<CECLNC>设置应和 CECTCR<CECDTRS>相同。

以下内容说明一个噪声消除例子，配置为<CECHNC [1:0]> = “10“ (3 次取样)，且<CECLNC[2:0]> = “011“ (4 次取样)。通过消除噪声，“0“被取样四次后，信号 “1“转变为 “0“。“1“被取样三次后，信号 “0“转移到 “1“。



(3) 周期误差

对 CECRCR1<CECMIN>和<CECMAX>位进行配置以检测一个周期误差。

可从每个取样时钟周期中检测到一个周期误差，以一个取样周期为单位，范围在−4/fs ~ +3/fs 之间，最大值 (67/fs，约 2.045ms)或最小值 (90/fs，约 2.747ms)。

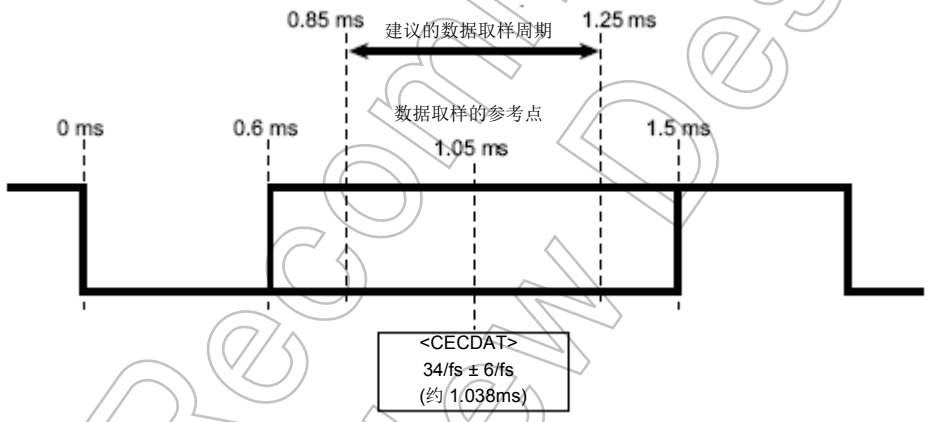
接收数据时检测到误差会导致一个误差中断，CEC 就会等待下一个起始位。已接收数据丢失。

(4) 判定数据点

将判定数据点的 CECRCR1 <CECDAT>位设置设置到“0“ 或 “1“。

起始点的基准时间为 34/fs（约 1.038ms)，也可以与以两个取样周期为单位的±6/fs 一起配置。

数据取样的推荐周期



(5) ACK 应答

设置 CECRCR1 <CECACKDIS> 位，说明当目标地址与逻辑地址寄存器中的地址相符时，是否发送逻辑 “0“作为 ACK 对数据块的应答。

无论<CECACKDIS>的位设置如何，逻辑“0“都被作为 ACK 应答发送到标题字组。

以下列出 ACK 应答。

“YES“说明 CEC 输出“0“作为对源自数据传输装置的 ACK 信号的应答 (ACK 位:逻辑 “0“)。

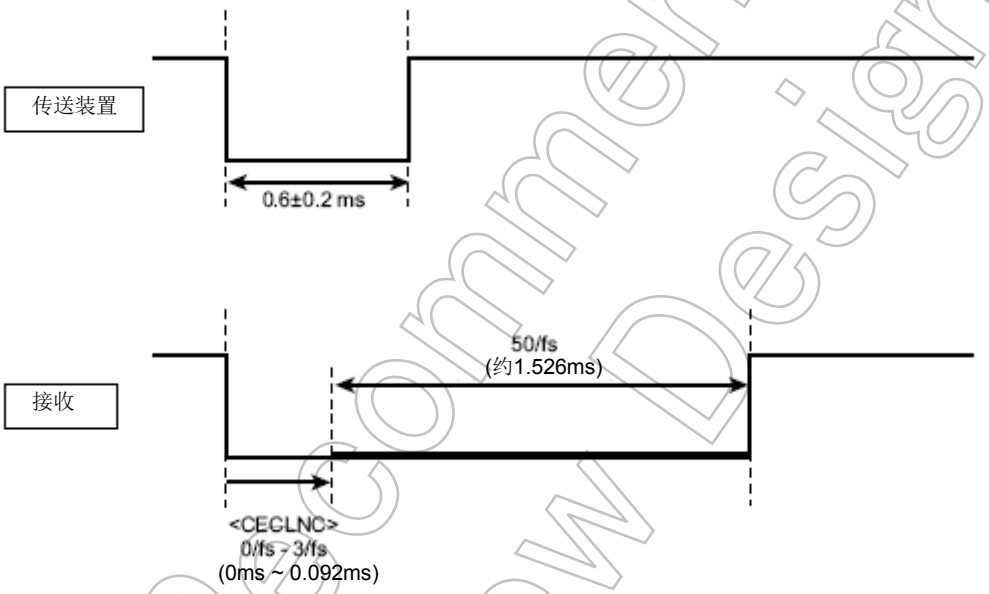
“NO“说明 CEC 不会输出“0“作为对源自数据传输装置的 ACK 信号的应答 (ACK 位:逻辑 “1“)。

寄存器设置		标题地址		数据块地址	
		符合性	不符合	符合性	不符合
CECRCR1 <CECACKDIS>	“0“ (对应逻辑 “0“)	YES	No	YES	No
	“1“ (不对应逻辑 “0“)			No	No

下面介绍 ACK 应答时间。

当检测到启动程序 ACK 位的下降沿时，该 IP 就会输出时长约为 1.526 ms 的“低“信号。“低“输出的启动时间由 CECRCR1<CECLNC> 位指定，该位也设置噪声取消时间。

注:<CECLNC>设置应和 CECTCR<CECDTRS>相同。



(6) 接收误差中断挂起

<CECRIHLD> 位，说明是否挂起接收误差中断 (最大周期误差，缓冲超限和波形误差)。当设置设置到“1“时，则检测到误差时不中断。

若数据持续流向 ACK 位，需用反向逻辑执行 ACK 应答。若随后位发生中断，需在 CECRCR1 寄存器<CECTOUT>设置的基础上将其定义为超时。ACK 应答或超时确定后，会产生中断。

(7) 判定超时周期

设置 CECRCR1<CECTOUT> 位，指定判定超时时间。

当 CECRCR1 <CECRIHLD>里指定的接收误差中断挂起设置有效时，需作此设置。

(8) 逻辑地址不符时的数据接收

即可指定当目标地址与 CECADD 寄存器内的设置地址不符时，是否要接收数据。

此时，通过检测误差可以执行普通数据接收和产生中断。然而，无论是头块还是数据块，都不会执行 ACK 应答。

注 1:无论<CECOTH>寄存器如何设置，总会接收到广播信息。

注 2:如果启动程序发送以起始位开头的信息，却未发送含 EOM= “1“的最后一个数据块，此时可判定最大周期误差，并产生中断。接着，信息接收就可以正常进行了。

(9) 起始位检测

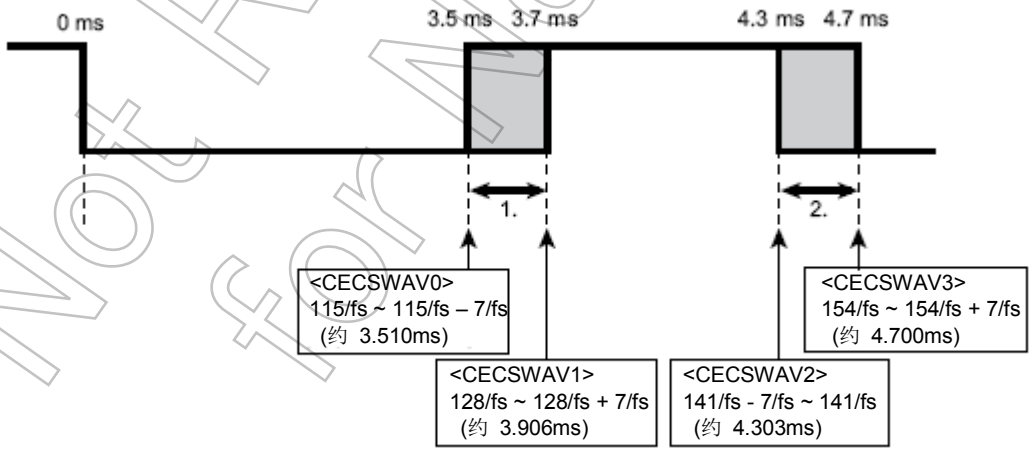
设置 CECRCR2 寄存器，可以分别指定起始位检测的上升时间和周期。

<CECSWAV0>指定最快的起始位上升时间。<CECSWAV1>指定最近的起始位上升时间(下图中 1 指示这段时间)。

<CECSWAV2>指定起始位的最小周期，<CECSWAV3>指定起始位的最大周期 (下图中 2. 指示这段时间)。

如果期间 1.中的上升沿和期间 2.中的下降沿被检测出来，起始位将被视作有效。

规格信号测试计时允许值 (起始位)。



(10) 波形误差检测

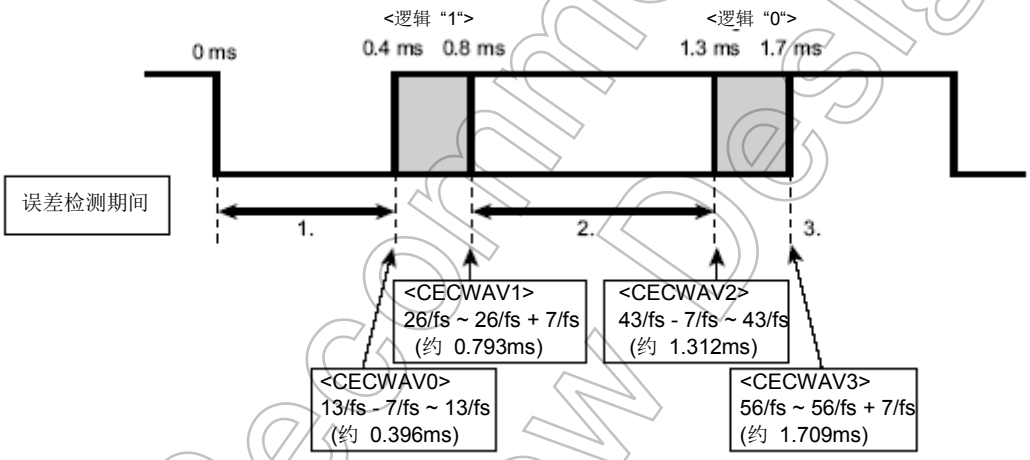
当已接收波形不在规定公差范围内时检测到误差时，设置 CECRCR3 寄存器。

当 CECRCR3 寄存器的 <CECWAVEN> 位被启用后，检测到误差。规定 <CECWAV0>, <CECWAV1>, <CECWAV2> 和 <CECWAV3> 位的检测时间。

如果在如下所示 1 期间.或 2 期间.检测到上升沿，或在 3.期间的规定时间内未检测到上升沿，波形误差中断就会产生。

1. 处于一个位起点和逻辑 “1“最快的上升时间之间。
2. 处于逻辑 “1“最近的上升时间和逻辑 “0“最快上升时间之间。
3. 逻辑 “0“最近的上升时间。

信号转换时期的容许 (数据位)



15.4.2.3 启用接收

设置好 CECADD, CECRCR1, CECRCR2 和 CECRCR3 寄存器后，启用 CECREN <CECREN>位，可以接收 CEC。检测一个起始位启动接收。

注:接收期间改变 CECADD, CECRCR1, CECRCR2 和 CECRCR3 寄存器的设置，可能会影响其正常操作。改变下图中的寄存器之前，设置 CECREN <CECREN>位禁止接收，读<CECREN>位和 CECTEN<CECTRANS>位以确保操作停止。

寄存器名称	比特符号	设置项
CECADD	<CECADD[15:0]>	逻辑地址
CECRCR1	<CECHNC><CECLNC>	噪声消除时间
	<CECMIN><CECMAX>	判定周期误差时间
	<CECOTH>	逻辑地址不符时的数据接收
CECRCR2	<CECSWAV0><CECSWAV1> <CECSWAV2><CECSWAV3>	起始位检测
CECRCR3	<CECWAV0><CECWAV1> <CECWAV2><CECWAV3>	波形误差检测 (启用后)

15.4.2.4 检测错误中断

接收数据时检测到误差会导致一个误差中断，CEC 就会等待下一个起始位。已接收数据丢失。

可挂起一个接收误差中断 (最大周期误差，接收缓冲溢出和波形误差)，继续接收并发送反向 ACK 应答。

可通过检查与中断对应的 CECRSTAT 寄存器的位来检查中断因素。

15.4.2.5 接收错误详情

(1) 周期误差

接收期间衡量两个连续位下降沿的一段时间。若该段时间与规定的最小值和最大值不符，周期误差中断就会产生。

最大周期时间和最小周期时间由 CECRCR1<CECMIN>和<CECMAX> 位设置。最大值为 90/fs (约 2.747ms) ，最小值为 67/fs (约 2.045ms)。可规定以 1/fs 为单位，在范围−4/fs ~ +3/fs 之间检测周期误差。

一旦出现周期误差中断，就需设置 CECRSTAT <CECRIMIN> 位或<CECRIMAX>位。

最小周期误差会导致 CEC 输出约 3.63 ms 时间的“低“。

注 1:一旦检测到最小周期误差，“低“检测噪声消除时间后会输出“低“。

注 2:如果启动程序发送以起始位开头的信息，却未发送含 EOM= “1“的最后一个数据块，此时可判定最大周期误差，并产生中断。详情见 “15.1.3 注意事项“。

(2) ACK 冲突

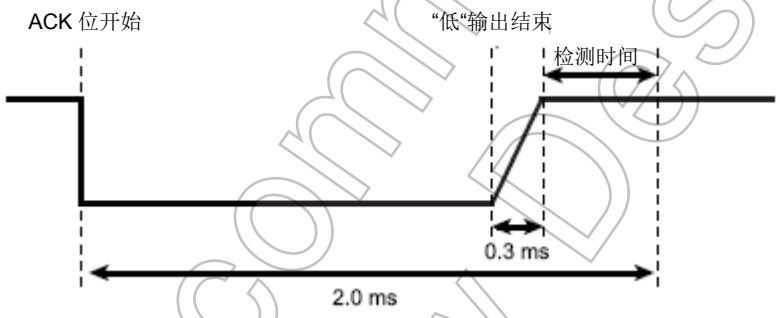
ACK 应答时，在规定输出时间之后检测到低会产生 ACK 冲突中断或最小周期误差中断。

该 ACK 冲突中断设置 CECRSTAT。最小周期误差中断设置 CECRSTAT <CECRIMIN> 位。

以下描述检测期间和检测方法。

输出“低“周期结束后约 0.3ms 检测开始，ACK 位的起始点 (下降沿)结束后约 2.0 ms 检测结束。

“低“输出周期结束后 0.3ms，CEC 检查其线路是否为 “0“。若为低，则 ACK 冲突中断出现。若为 “高“，而检测时间又检测到“低“，最小周期误差中断出现。最小周期误差导致 CEC 输出 “低“长达约 3.63ms 时间。



(3) 接收缓冲超限

当读接收缓冲区储存的数据之前已完成下一个数据接收时，接收缓冲超限中断发生。

该中断设置 CECRSTAT <CECRIOR>位。

(4) 波形误差

当在 CECRCR3 启用波形误差检测时，波形误差出现。

检测到的波形与判定波形不同时，会导致波形误差出现。此时中断产生。

该中断设置 CECRSTAT <CECRIWAV> 位。

(5) 接收误差中断挂起

误差检测时，可规定挂起最大周期误差，缓冲超限和波形误差时不用产生中断。可以在 CECRCR1 <CECRIHLD>位做此设置。可以在 CECRCR1 <CECRIHLD>位做此设置。要启用该设置，需要对 CECRCR1 <CECTOUT>位进行超时设置。

挂起启用条件下，若 CEC 持续接收下一个位，包括 ACK 位在内的所有接收都已结束，则执行反向 ACK 应答时会产生中断。将 “1“设置给 CECRSTAT 寄存器的位:表示接收完成的 <CECRIEND> 位和与检测到的误差对应的位。

若下一位接收中断，CEC 会开始测量超时时间，超时后会产生中断。将 “1“ 设置给与检测到的误差对应的 CECRSTAT 寄存器中的位。

如传输过程中总线空闲等待时间的情况一样，超时时间从收到最后一个位后开始测量。

中断挂起信息会保留 ~ 收到 EOM 位或超时产生时为止。如此，中断挂起时如果收到多个位，接收的每个位都会发生中断。将 “1“设置给 CECRSTAT 寄存器的位:表示接收完成的 <CECRIEND>位，还有与检测到的误差对应的位。挂起中断和接收完成标志被设置给 CECRSTAT 寄存器中的位。

注 1:挂起中断时，一旦在下一个已接收位中检测到最小周期误差，最小周期误差中断便会出现。输出 “低 “到 CEC 约 3.63 ms。挂起中断和最小周期误差标志被设置给 CECRSTAT 寄存器中的位。

注 2:若挂起中断期间只发生中断而未发生最小周期误差中断，那么 CEC 继续接收直到 ACK 应答或超时为止。所有已检测到的中断标志都设置给 CECRSTAT 寄存器中的位。

15.4.2.6 停止接收

给 CECREN <CECREN> 位写入 “0“禁止数据接收。若在数据接收期间禁止数据接收，接收操作将停止，已接收数据将废弃。

注:如果禁止接收时，“低“作为最小周期误差的信号，那么“低“输出也会停止。

15.4.3 传输

15.4.3.1 基本操作

传输设置时，CEC 首先确认总线空闲等待状态，检查 CEC 下降沿信号是否在规定的位周期内存在，接着才发送一个起始位。始终确认总线空闲等待状态。如此，一旦总线空闲等待条件完备，传输设置结束后会立即开始传输。

传输完一个起始位后，CEC 将存储于传输缓冲区的一位数据和 EOM 数据传送到移位寄存器中。当开始传输一位数据的首位时，传输中断出现，CECTSTAT<CECTISTA>被设置。传输中断产生后，下一个一位数据便做好向传输数据缓冲区输送的准备。

一位传输任务按照 8 位数据，EOM 位，ACK 位应答确认的传输顺序完成。

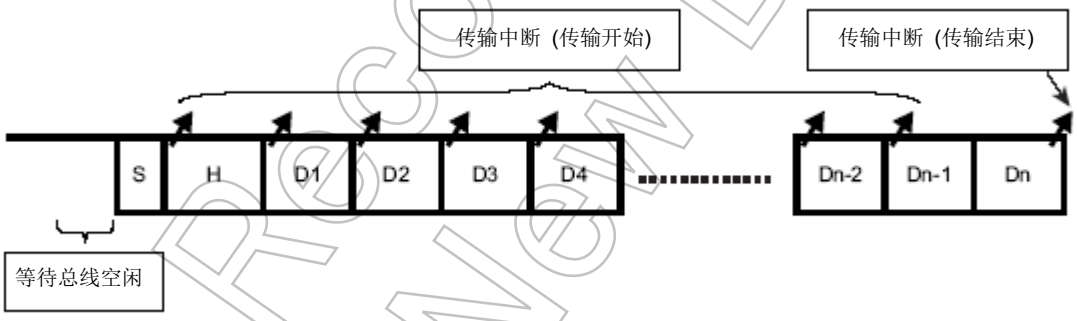
数据继续传输，直到 EOM 设置到“1“。

若 EOM 设置到“1“, 确认完数据，EOM，ACK 位传输和 ACK 位应答后，传输中断会结束。传输中断出现时，CECTSTAT<CECTIEND> 被设置。

中断产生会结束一些列传输过程，此时 CECTEN<CECTEN> 清除。

若在传输过程中出现误差，会出现误差中断来终止传输。

即使启用接收，传输过程中也不允许执行接收。



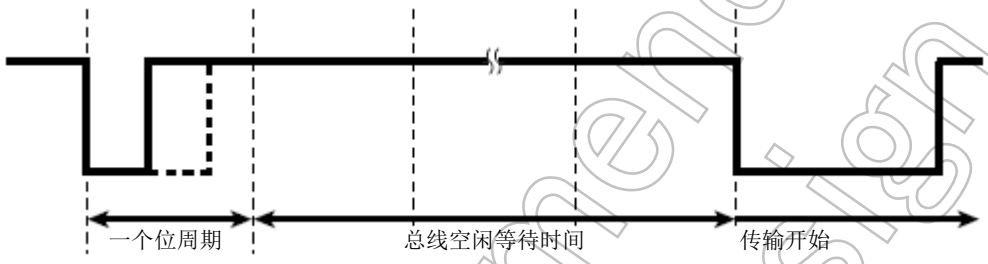
15.4.3.2 预配置

传输数据之前，需要对传输控制寄存器 (CECTCR)和传输缓冲寄存器 (CECTBUF)进行传输设置。

(1) 总线空闲等待时间

在 CECTCR<CECFREE>位指定总线空闲等待时间。可在 1 ~ 16 位周期范围内规定。

最后一位下降沿后对总线空闲等待时间计数，开始一个位周期。相对于位周期规定的数目，如果信号停留在较高水平，则开始传输。



(2) 广播信息传输

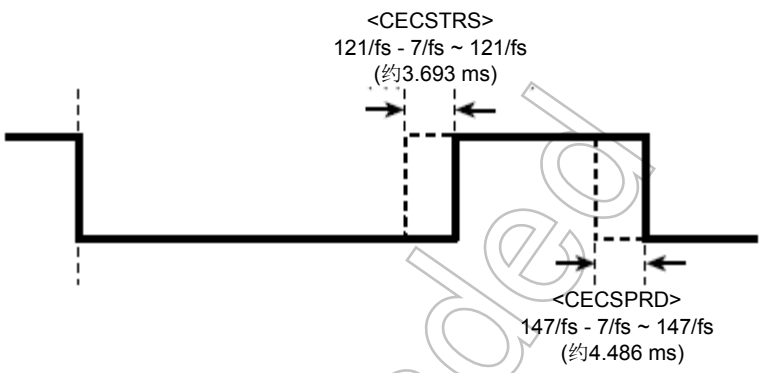
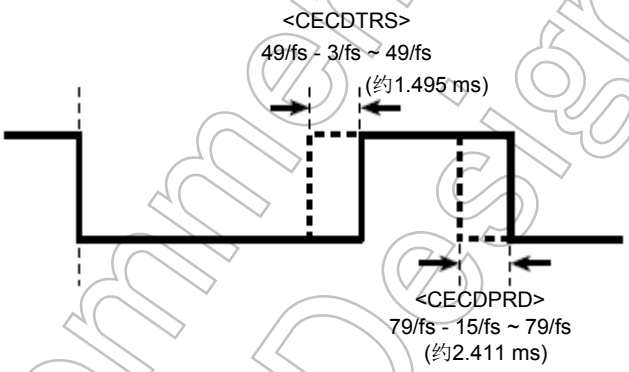
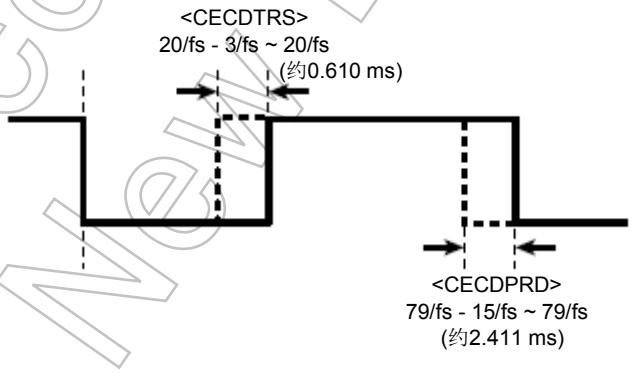
广播信息传输时对 CECTCR <CECBRD>位进行设置。若该位已设置，在 ACK 周期出现误差期间，逻辑 “0“应答。若该位未设置，则在 ACK 周期出现误差期间，逻辑 “1“应答。

(3) 调节传输波形

起始位和数据位均可调整上升时间和周期。有了 CECTCR <CECSTRS> <CECSPRD> <CECDTRS> <CECDPRD> 位, 就可以判定最快上升/周期时间和参考值之间的时间。

下列数字表示了随着起始位设置到逻辑 “0“和逻辑 “1“，波形如何变化。

注:<CECDTRS>的设置应与 CECRCR1<CECLNC>设置相同。

起始位	
逻辑“0“	
逻辑“1“	

(4) 准备传输数据

用 CECTBUF 寄存器设置一位传输数据和 EOM 数据。

15.4.3.3 检测传输错误

传输过程中进行误差检测会产生传输中断和传输终止。此举会导致 CECTEN <CECTEN> 位清除。

要判定误差因素，CECTSTAT 寄存器需要有与每个中断相应的位。可以通过检查这些位来判定中断因素。

注:试图通过误差终止传输会导致向输往 CEC 的波形不正常。因为误差一旦出现，输出立即停止。

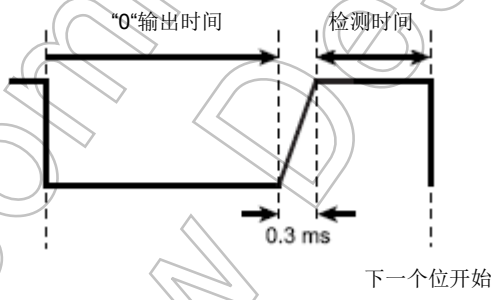
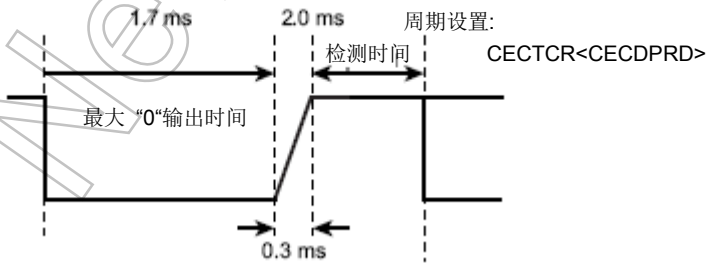
15.4.3.4 传输错误详情

(1) 仲裁丢失

当 CEC 在完成一个合适的低持续时间后检测到"低"时会出现仲裁丢失误差。

检测仲裁丢失误差设置 CECTSTAT <CECTIAL>位。

下图为两种类型的仲裁丢失检测时期。

起始位 数据位 EOM位	
ACK 位	

(2) ACK 误差

ACK 应答与 CECTCR <CECBRD> 位中的规定配置不一致时，会出现 ACK 误差中断。

当 ACK 误差中断出现时，对 CECTSTAT <CECTIACK>位进行配置。

下列情况可检测到 ACK 误差。

配置	认定为ACK误差当
<CECBRD> = 0 广播传输？: No	ACK应答为逻辑 “1“
<CECBRD> = 1 广播传输？: Yes	ACK 应答为逻辑 “0“

(3) 传输缓冲欠载

以下事件会造成传输缓冲欠载。

1. 传输缓冲区的数据传到移位寄存器。
2. 中断出现。
3. 传输一位数据。
4. 开始传输一位随后出现的数据时未将数据设置于传输缓冲区。欠载误差出现时，CECTSTAT <CECTIUR>位被设置。

(4) ACK 误差和传输缓冲超载顺序

如果一位数据传输结束时检测到 ACK 误差和传输缓冲欠载的中断因素，传输缓冲欠载有优先权。

传输缓冲欠载中断首先出现，ACK 误差中断次之。

15.4.3.5 停止传输

若要终止传输，发送包括表示为 “1“的 EOM 位数据。此时会产生传输结束中断

请注意，若在传输过程中传输起始位设置到 “0“将无法保证正常运行。

15.4.3.6 重传

误差检测导致传输终止。重新进行传输时，对启动传输条件和数据进行设置。

15.4.4 软件复位

整个 CEC 功能可由软件启动。

将 CECRESET <CECRESET>位设置到 “1“导致下列操作。

接收 : 立即停止。已接收数据丢失。

传输 : 立即停止包括向 CEC 线路的输出。

寄存器 : 除 CECEN 外，对所有的寄存器进行初始化。

请注意传输过程中的软件复位可能会造成 CEC 线路波形与规定的不同。

16. CAN控制器 (CAN)

此产品包含一个 CAN 控制器通道。

16.1 概述

- 与 CAN2.0 B(有效)版本的兼容
- 支持标准和扩展格式
- 数据帧与远传帧均支持每种格式
- 32 个邮箱 (其中 31 个接收和传输邮箱，1 个仅能接收的邮箱)
- CAN 总线传输频率最高达到 1Mbps (系统时钟至少达到 48MHz)
- 位定时参数相当于 Intel 82527™
- 内置传输频率预定标器
- 信息传输的命令通过以下两种类型的内部仲裁进行选择:
  - 小容量数据的邮箱优先传输
  - 高级优先级识别符的邮箱优先传输
- 接收和传输信息的时间戳功能
- 操作模式

正常操作模式	
配置模式	
休眠模式	通过CAN总线活动状态检测或对主控寄存器MCR进行写入访问，CAN被唤醒 (当CANMCR<WUBA> = “1“)
挂起模式	CAN总线处于非激活状态
测试回环模式	自动应答
测试错误模式	可写入的错误计数器

- 两种系统下的信息接收屏蔽功能 -可编程的全局接收屏蔽 (一般针对邮箱 0 ~ 31 而言) –可编程局部接收屏蔽 (仅针对邮箱 31)
- 针对 ID 扩展位的接收屏蔽位
- 中断信号

INTCANRX	: CAN接收完全中断
INTCANTX	: CAN传输完全中断
INTCANGB	: CAN全局中断 8种原因可产生中断: 警告电平，消极报错状态，总线关闭状态)

16.2 方块图

图 16-1 显示出 CAN 控制器的框图

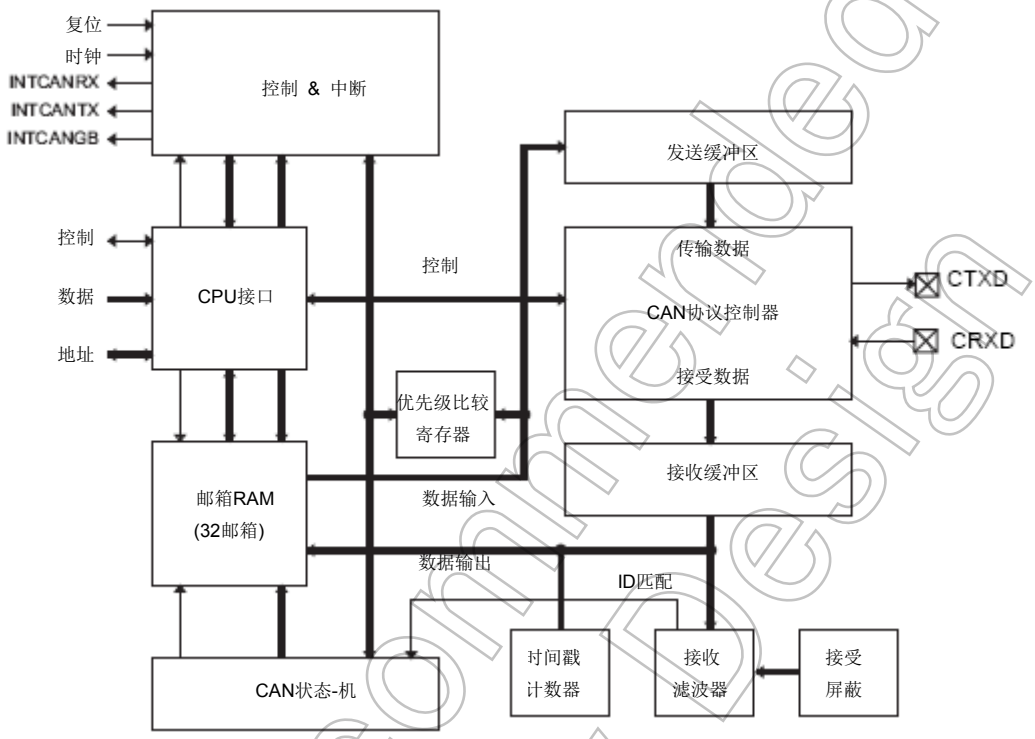


图 16-1 CAN 控制器的方块图

16.3 CAN接口

接入 CAN 总线的接口是一个 CRXD 输入脚线和一个 CTXD 输出引脚通过 CAN 总线收发器来连接这些引脚 (与 ISO/DIS 11898 兼容)。

高速和低速收发器存在区别。在芯片中的这些引脚的电特性 (比如，3.3 V ~ 5 V)的 IP 条件必须考虑以满足收发器的需求。

16.4 寄存器

16.4.1 寄存器列表

邮箱x	基址
通道0	0x4000_2000
通道1	0x4000_2020
通道2	0x4000_2040
通道3	0x4000_2060
通道4	0x4000_2080
通道5	0x4000_20A0
通道6	0x4000_20C0
通道7	0x4000_20E0
通道8	0x4000_2100
通道9	0x4000_2120
通道10	0x4000_2140
通道11	0x4000_2160
通道12	0x4000_2180
通道13	0x4000_21A0
通道14	0x4000_21C0
通道15	0x4000_21E0
通道16	0x4000_2200
通道17	0x4000_2220
通道18	0x4000_2240
通道19	0x4000_2260
通道20	0x4000_2280
通道21	0x4000_22A0
通道22	0x4000_22C0
通道23	0x4000_22E0
通道24	0x4000_2300
通道25	0x4000_2320
通道26	0x4000_2340
通道27	0x4000_2360
通道28	0x4000_2380
通道29	0x4000_23A0
通道30	0x4000_23C0
通道31	0x4000_23E0

寄存器名称 (x=0到31)		地址 (基+)
信号ID字段寄存器	CANMBxID	0x0000
时间戳值 / 将信息控制寄存器	CANMBxTSVMCF	0x0008
数据字段寄存器	CANMBxDL	0x0010
数据字段寄存器	CANMBxDH	0x0018

基址 = 0x4000_2400

寄存器名称		地址 (基+)
邮箱配置寄存器	CANMC	0x0000
邮箱去向寄存器	CANMD	0x0008
传输请求设置寄存器	CANTRS	0x0010
传输请求复位寄存器	CANTRR	0x0018
传输应答登记寄存器	CANTA	0x0020
中止应答登记寄存器	CANAA	0x0028
接收信号等待寄存器	CANRMP	0x0030
接收信息丢失寄存器	CANRML	0x0038
局部验收屏蔽寄存器	CANLAM	0x0040
全局验收屏蔽寄存器	CANGAM	0x0048
主控寄存器	CANMCR	0x0050
全局状态寄存器	CANGSR	0x0058
位配置寄存器 1	CANBCR1	0x0060
位配置寄存器 2	CANBCR2	0x0068
全局中断标志寄存器	CANGIF	0x0070
全局中断屏蔽寄存器	CANGIM	0x0078
邮箱传输中断标志寄存器	CANMBTIF	0x0080
邮箱接收中断标志寄存器	CANMBRIF	0x0088
邮箱中断屏蔽寄存器	CANMBIM	0x0090
更改数据请求寄存器	CANCDR	0x0098
远程帧悬挂寄存器	CANRFP	0x00A0
CAN错误计数器寄存器	CANCEC	0x00A8
时间戳计数预定标器寄存器	CANTSP	0x00B0
时间戳计数寄存器	CANTSC	0x00B8

16.4.2 CANMBxID (信息ID字段寄存器)

	31	30	29	28	27	26	25	24
比特符号	IDE	GAME/LAME	RFH	ID				
复位后								
	23	22	21	20	19	18	17	16
比特符号	ID							
复位后								
	15	14	13	12	11	10	9	8
比特符号	ID							
复位后								
	7	6	5	4	3	2	1	0
比特符号	ID							
复位后								

位	比特符号	型号	功能
31	IDE	R/W	ID扩展位 0: 采用从<ID28>到 <ID18>的标准格式 (11-位ID) 1: 采用从<ID28>到 <ID0>的标准格式 (29-位ID) 通过选择是否接收或传输扩展格式 (<IDE>= “1“) 或标准格式 (<IDE>= “0“)来进行邮箱设置
30	GAME / LAME	R/W	全局 (GAME) / 局部 (LAME) 接收屏蔽启用位 0: 接收屏蔽不能用作接收过滤。 1: 接收屏蔽可以用作接收过滤。 <GAME>是为在邮箱0～30中共享的全局接收屏蔽GAM的允许位，<LAME>是为仅为邮箱31所用的局部接收屏蔽LAM的允许位。 当 <GAME>=0 或<LAME>=0，当接收的信息ID与邮箱ID一致时，接收的信息就会被存储在邮箱中。 接收屏蔽功能不适用于传输邮箱这种情况下，总是设置 <GAME>到 “0“。
29	RFH	R/W	远程帧处位 (仅针对传输邮箱) 0:传输邮箱不会对远程帧作出响应软件必须来控制远程帧 1:传输邮箱对远程帧作出响应 (<TRS>位被设置) <RFH> 决定着传输邮箱是否会自动对远程帧接收作出响应。 当接收到的远程帧IP与<RFH>= “1“ 和 <GAME>= “1“的传输邮箱中的ID匹配时，该邮箱中的ID将被远程ID所重写，邮箱将自动对重写后的ID的远程帧作出响应。 就像接收邮箱中的数据帧一样进行处理 (<RMP>位和<RFP>位被设置)
28-0	ID[28:0]	R/W	信息ID 标准格式 (11-位ID): 从<ID28> ~ <ID18>被使用。 扩展格式 (29-位ID): 从<ID28> ~ <ID0> 被使用。 为了确定信息ID的优先级别，从ID的最高位开始连续拥有最多 “0“的信息ID具有较高的优先级别。

在初始设置时，进行邮箱 ID 登记为了在邮箱被激活之后，改变信息 ID 字段或一个邮箱，需首先清除与邮箱相对应的 CANMC 寄存器内的<MCx>位为 “0“，然后在写入一个新 ID 之前，禁用 CAN 控制器的邮箱。

16.4.3 CANMBxTSVMCF (时间戳值 / 信息控制字段寄存器)

	31	30	29	28	27	26	25	24
比特符号	TSV							
复位后								
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后								
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后								
	7	6	5	4	3	2	1	0
比特符号	-	-	-	RTR	DLC			
复位后								

位	比特符号	型号	功能
31-16	TSV[15:0]	R/W	时间戳计数值 当信息被成功的接收或传输时，可读的16-位的时间戳计数值被存储下来。当信息接收或传输失败时没有计数值被设置。 想更好的了解时间戳计数功能，请参考第16.5.6中的时间戳功能。
15-5	–	R	读取未定义。 写作 “0“。
4	RTR	R/W	远程帧传输请求。 0:数据帧 1:远程帧
3-0	DLC[3:0]	R/W	数据长度码 设置信息数据的长度 (字节数)

<DLC[3:0]>	位数	相应的数据
0000	0 字节	无
0001	1 字节	D0
0010	2 字节	D0，D1
0011	3 字节	D0，D1，D2
0100	4 字节	D0，D1，D2，D3
0101	5 字节	D0，D1，D2，D3，D4
0110	6 字节	D0，D1，D2，D3，D4，D5
0111	7 字节	D0，D1，D2，D3，D4，D5，D6
1000	8 字节	D0，D1，D2，D3，D4，D5，D6，D7

当<DLC3:0>= “1001“ 或更大的数值被设置时，数据长度以8字节进行处理。

时间戳值不需要在初始条件下进行设置。

在接收邮箱的条件下，信息控制字段不需要进行初始编程。当接收到的信息被存储在邮箱中，<RTR> 和 <DLC[3:0]> 也同时被存储在信息控制字段中。传输邮箱需要初始设置。

在邮箱被激活之后，为了改变一个传输邮箱的信息控制字段 (设置为 <RFH>= “1“)，需首先清除 CANMC<MCx>位为 “0“ ，然后在写入一个新的<RTR>和<DLC[3:0]>之前，禁用 CAN 控制器的邮箱。传输邮箱被设置为<RFH>= “0“的信息控制字段可以在不受 CANMC<MCx>位设置的影响下进行更改，但是用户需要在重新写入一个<RTR>和 <DLC[3:0]>之前，检查 CANTRS<TRSx>位为 “0“。

16.4.4 CANMBxDH/CANMBxDL (数据字段寄存器)

对于数据传输，按照邮箱中<DLC[3:0]>的数据字数设置，传输数据。

关于数据接收，接收信息中的数据长度代码被复制到邮箱的<DLC[3:0]>中，而且仅设置在<DLC[3:0]> 中的数据字节数才有效

邮箱是可读写的，但是不能在接收邮箱中写入数据字段如果写入了数据字段，在接收到的数据中可能会出现不匹配的现象。

为了更新一个传输邮箱设置的数据字段为<RFH>= “1“，把 CANCDR<CDR>设置到 “1“，并在写入新数据之前，暂时停止传输请求。为了更新一个传输邮箱设置的数据字段到 <RFH>= “0“，确定在写入新数据之前，CANCDR<CDR>位为 “0“。

CANMBxDH

	31	30	29	28	27	26	25	24
比特符号	D7							
复位后								
	23	22	21	20	19	18	17	16
比特符号	D6							
复位后								
	15	14	13	12	11	10	9	8
比特符号	D5							
复位后								
	7	6	5	4	3	2	1	0
比特符号	D4							
复位后								

位	比特符号	型号	功能
31-24	D7[7:0]	R/W	存储传输和接收的数据。
23-16	D6[7:0]	R/W	存储传输和接收的数据。
15-8	D5[7:0]	R/W	存储传输和接收的数据。
7-0	D4[7:0]	R/W	存储传输和接收的数据。

CANMBxDL

	31	30	29	28	27	26	25	24
比特符号	D3							
复位后								
	23	22	21	20	19	18	17	16
比特符号	D2							
复位后								
	15	14	13	12	11	10	9	8
比特符号	D1							
复位后								
	7	6	5	4	3	2	1	0
比特符号	D0							
复位后								

位	比特符号	型号	功能
31-24	D3[7:0]	R/W	存储传输和接收的数据。
23-16	D2[7:0]	R/W	存储传输和接收的数据。
15-8	D1[7:0]	R/W	存储传输和接收的数据。
7-0	D0[7:0]	R/W	存储传输和接收的数据。

16.4.5 CANMC (邮箱配置寄存器)

	31	30	29	28	27	26	25	24
比特符号	MC31	MC30	MC29	MC28	MC27	MC26	MC25	MC24
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	MC23	MC22	MC21	MC20	MC19	MC18	MC17	MC16
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	MC15	MC14	MC13	MC12	MC11	MC10	MC9	MC8
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	MC7	MC6	MC5	MC4	MC3	MC2	MC1	MC0
复位后	0	0	0	0	0	0	0	0

位	比特符号	型号	功能
31-0	MC31 ~ MC0	R/W	访问邮箱配置 (每个位都与邮箱31 ~ 0相对应) 0:因CAN控制器，禁用相应的邮箱MBx 1:因CAN控制器，激活相应的邮箱MBx CPU写入访问

	ID字段	<RFH>= “1“的传输邮箱	数据字段	控制字段
<MCx>=0	启用	启用	启用	启用
<MCx>=1	禁用	禁用	启用	启用

注:在运行中的 CANMC 改编程序中，需要考虑以下方面。

接收:针对接收邮箱，需要确定正在进行接收的时候，邮箱没有被禁用。如果一个邮箱正在接收的过程中被禁用或被重新配置，当前帧可能被接收。

传输:当 CAN 控制器正在传输数据 (CANTRS<TRSx>= “1“)时， 在传输完成 (CANTRS<TRSx>= “0“)之后，清除<MCx> 为 “0“。

16.4.6 CANMD (邮箱去向寄存器)

	31	30	29	28	27	26	25	24
比特符号	MD31	MD30	MD29	MD28	MD27	MD26	MD25	MD24
复位后	1	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	MD23	MD22	MD21	MD20	MD19	MD18	MD17	MD16
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	MD15	MD14	MD13	MD12	MD11	MD10	MD9	MD8
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	MD7	MD6	MD5	MD4	MD3	MD2	MD1	MD0
复位后	0	0	0	0	0	0	0	0

位	比特符号	型号	功能
31	MD31	R	邮箱去向 ： 邮箱31 邮箱31是只接收邮箱它总是被设置到“1“并且不会改变。
30-0	MD30 ~ MD0	R/W	邮箱去向 ： 邮箱30 ~ 0 (每个位都与邮箱30 ~ 0相对应) 0:设置为一个传输邮箱 1:设置为一个接收邮箱 每个邮箱都可以被设置为传输邮箱或接受邮箱。

在初始条件下设置 CANMD 寄存器。程序正在运行时，邮箱的去向不能被改变。为了更改 CANMD 寄存器设置，在更改之前设置相应的 CANMC<MCx>位到 “0“。

16.4.7 CANTRS (传输请求寄存器)

	31	30	29	28	27	26	25	24
比特符号	-	TRS30	TRS29	TRS28	TRS27	TRS26	TRS25	TRS24
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	TRS23	TRS22	TRS21	TRS20	TRS19	TRS18	TRS17	TRS16
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	TRS15	TRS14	TRS13	TRS12	TRS11	TRS10	TRS9	TRS8
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	TRS7	TRS6	TRS5	TRS4	TRS3	TRS2	TRS1	TRS0
复位后	0	0	0	0	0	0	0	0

位	比特符号	型号	功能
31	−	R	读取：读作“0“。 写入：写作“0“。
30-0	TRS30 ~ TRS0	R/W	传输请求设置 (每个位都与邮箱30 ~ 0相对应) 设置<TRSx>需要相应的邮箱x进行信息传输。 当传输被多个邮箱同时请求时，根据与MCR<MTOS>位对应的优先级别进行信息传输。 从CPU向传输邮箱×中写入“1“可以设置位。从CPU中写入“0“是无效的。

注:邮箱31是只接收邮箱。

传输请求设置寄存器可以通过从 CPU 向传输邮箱的 CANTRS<TRSx> 位写入“1“来进行设置。接收邮箱的 CANTRS<TRSx>位不能被设置。

当信息已经被成功发出或通过设置 CANTRR<TRRx>位为“1“来复位传输请求时，CANTRS<TRSx>位被清“0“。

当传输失败，传输过程会一直重复直到成功为止或者通过设置 CANTRR<TRRx>位到“1“来进行传输请求的复位。

当 CANTRS<TRSx>位为“1“，邮箱×不能被写入。

16.4.8 CANTRR (传输请求寄存器)

	31	30	29	28	27	26	25	24
比特符号	-	TRR30	TRR29	TRR28	TRR27	TRR26	TRR25	TRR24
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	TRR23	TRR22	TRR21	TRR20	TRR19	TRR18	TRR17	TRR16
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	TRR15	TRR14	TRR13	TRR12	TRR11	TRR10	TRR9	TRR8
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	TRR7	TRR6	TRR5	TRR4	TRR3	TRR2	TRR1	TRR0
复位后	0	0	0	0	0	0	0	0

位	比特符号	型号	功能
31	-	R	读取 ：读作 “0“。 写入 ：写作 “0“。
30-0	TRR30 ~ TRR0	R/W	传输请求复位 (每个位都与邮箱30 ~ 0相对应。) 通过设置 <TRRx> 取消相应邮箱x的信息传输。 从CPU向传输邮箱×中写入 “1“可以设置位。从CPU写入 “0“是无效的。

注: 邮箱31是只接收邮箱。

传输请求复位寄存器可以通过从 CPU 向传输邮箱中仅有的 CANTRR<TRRx>位写入 “1“来进行复位。接收邮箱的 CANTRR<TRRx>位不能被设置。

当信息被成功传输或被中止，通过内在逻辑，CANTRR<TRRx>将被清除为 “0“。从 CPU 中写入 “1“是无效的。

当 CANTRR<TRRx>位为 “1“时，不要向邮箱 x 中写入。

设置 CANTRR<TRRx>位来取消通过 CANTRS<TRSx>位设置的邮箱 x 的信息传输，如果出现这种情况，已经执行的操作将可能出现以下三种情况:

a. 一个信息传输请求还未被执行。
一个信息传输请求将被立即清除。
(CANTRS<TRSx> = 0, CANTRR<TRRx> = 0, CANAA<AAx> = 1)

b. 一个信息传输请求当前被传输，但出现一个仲裁丢失错误或在 CAN 总线中检测到一个错误。
信息传输请求被清除且传输被取消。
(CANTRS<TRSx> = 0, CANTRR<TRRx> = 0, CANAA<AAx> = 1)

c. 一个信息传输请求当前被传输，未出现仲裁丢失错误且在 CAN 总线中未检测错误。
信息传输请求将不会被清除且传输将被完成。
(CANTRS<TRSx> = 0，CANTRR<TRRx> = 0，CANTA<TAx> = 1)

16.4.9 CANTA (传输应答寄存器)

	31	30	29	28	27	26	25	24
比特符号	-	TA30	TA29	TA28	TA27	TA26	TA25	TA24
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	TA23	TA22	TA21	TA20	TA19	TA18	TA17	TA16
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	TA15	TA14	TA13	TA12	TA11	TA10	TA9	TA8
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	TA7	TA6	TA5	TA4	TA3	TA2	TA1	TA0
复位后	0	0	0	0	0	0	0	0

位	比特符号	型号	功能
31	−	R	读取：读作“0“。 写入：写作“0“。
30-0	TA30 ~ TA0	R/W	传输应答 (每位都与邮箱30 ~ 0相对应) 当邮箱×中的信息被成功传输，<TAx>位将被设置为“1“。 通过从CPU向<TAx>位或TRS<TRSx>位中写到“1“，<TAx>位将被清除。

注: 邮箱31是只接收邮箱。

当邮箱×中的信息被成功传输，CANTA<TAx>位将被设置到“1“。通过把在邮箱中断屏蔽寄存器中相应的<MBIMx>位设置到“1“，当邮箱中断被激活，邮箱传输中断标志寄存器 CANMBTIF 的<MBTIFx>位被设置到“1“，而且会出现 CAN 传输完成中断。

从 CPU 向<TAx>位或 CANTRS<TRSx>位写入“1“将清除<TAx>位。从 CPU 向<TAx>位或 CANTRS<TRSx>位写到“0“是无效的。

16.4.10 CANAA (中止应答寄存器)

	31	30	29	28	27	26	25	24
比特符号	-	AA30	AA29	AA28	AA27	AA26	AA25	AA24
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	AA23	AA22	AA21	AA20	AA19	AA18	AA17	AA16
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	AA15	AA14	AA13	AA12	AA11	AA10	AA9	AA8
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	AA7	AA6	AA5	AA4	AA3	AA2	AA1	AA0
复位后	0	0	0	0	0	0	0	0

位	比特符号	型号	功能
31	−	R	读取 : 读作 “0“。 写入 : 写作 “0“。
30-0	AA30 ~ AA0	R/W	中止应答 (每位都与邮箱30 ~ 0相对应) 当邮箱×中的信息没有被成功传输， <AAx>位将被设置到 “1“ 。 通过从CPU向<AAx>位或CANTRS<TRSx>位中写入 “1“，<AAx>位将被清除。

注:邮箱31是只接收邮箱。

当邮箱×中的信息没有被成功的传输，CANAA<AAx>字节将被设置为 “1“。当在全局中断标志寄存器中 CANGIF<TRMABF>位也被设置为“1“，通过设置全局中断屏蔽寄存器中的 CANGIM<TRAMABM>为 “1“来激活传输中止中断，这将导致 CAN 全局中断 INTCANGB。

从 CPU 向<AAx>位或 CANTRS<TRSx> 位写入 “1“将清除<AAx>位。从 CPU 向<AAx>位或 CANTRS<TRSx> 位写入 “0“是无效的

16.4.11 CANCDR (更改数据请求寄存器)

	31	30	29	28	27	26	25	24
比特符号	-	CDR30	CDR29	CDR28	CDR27	CDR26	CDR25	CDR24
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	CDR23	CDR22	CDR21	CDR20	CDR19	CDR18	CDR17	CDR16
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	CDR15	CDR14	CDR13	CDR12	CDR11	CDR10	CDR9	CDR8
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	CDR7	CDR6	CDR5	CDR4	CDR3	CDR2	CDR1	CDR0
复位后	0	0	0	0	0	0	0	0

位	比特符号	型号	功能
31	−	R	读取：读作 “0“。 写入：写作 “0“。
30-0	CDR30 ~ CDR0	R/W	更改数据请求 (每个位都与邮箱30 ~ 0相对应。) 当传输邮箱的<CDRx> 位被设置到 “1“，此邮箱的传输请求被忽略。 它意味着设置了CANTRS<TRSx>位和<CDRx>位的邮箱x将会被清出内部仲裁范围，并且如果传输还未开始，它将不会被传输。当 <CDRx> 位被清除为 “0“之后，邮箱x重新被包含在内部仲裁范围之内。

注:邮箱31是只接收邮箱。

当更新已启用远程帧的自动应答 (CANMBnID<RFH>= “1“)的传输邮箱×的数据字段时，更改数据请求寄存器 CABCDR 是有效的。启用自动应答的邮箱 x，根据接收的远程帧开始自动进行信息传输，因此它可能在数据传输的过程中更新数据字段 (在这种情况，更新后的数据在传输过程被中途输出)。通过设置<CDRx>位到 “1“，暂时中止数据传输，可以避免数据字段的更新。

16.4.12 CANRMP (接收信息悬挂寄存器)

	31	30	29	28	27	26	25	24
比特符号	RMP31	RMP30	RMP29	RMP28	RMP27	RMP26	RMP25	RMP24
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	RMP23	RMP22	RMP21	RMP20	RMP19	RMP18	RMP17	RMP16
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	RMP15	RMP14	RMP13	RMP12	RMP11	RMP10	RMP9	RMP8
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	RMP7	RMP6	RMP5	RMP4	RMP3	RMP2	RMP1	RMP0
复位后	0	0	0	0	0	0	0	0

位	比特符号	型号	功能
31-0	RMP31 ~ RMP0	R/W	接收信息等待 (每个位都与邮箱31 ~ 0相对应。) 接收到一个信息或者接收到的内容被写入邮箱×之后，<RMPx>位被设置到 “1“。 接收数据被读取之后，向<RMPx>位中写入 “1“，可以清除<RMPx>位。

当邮箱×中的信息已经被成功的接收，CANRMP<RMPx>位被设置到 “1“。通过在邮箱中断屏蔽寄存器中 CANMBIM 设置相应的<MBIMx>位到 “1“来启用邮箱中断，邮箱接收中断标志寄存器 CANMBRIF 的<MBRIFx>位被设置到 “1“，CAN 接收完成中断 INTCANRX。

通过从 CPU 向<RMPx>中写入 “1“ 来清除 <RMPx> 位。从 CPU 向<RMPx>位中写入 “1“是无效的。

16.4.13 CANRML (接收信号丢失寄存器)

	31	30	29	28	27	26	25	24
比特符号	RML31	RML30	RML29	RML28	RML27	RML26	RML25	RML24
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	RML23	RML22	RML21	RML20	RML19	RML18	RML17	RML16
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	RML15	RML14	RML13	RML12	RML11	RML10	RML9	RML8
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	RML7	RML6	RML5	RML4	RML3	RML2	RML1	RML0
复位后	0	0	0	0	0	0	0	0

位	比特符号	型号	功能
31-0	RML31 ~ RML0	R/W	接收信号丢失 (每个位都与邮箱31 ~ 0相对应。) 当 <RMPx>被设置为 “1“的邮箱接收到以下信息，接收到的信息内容被重写到邮箱×中，并且<RMLx>位被设置到 “1“。 向<RMPx>位中写入 “1“可以清除<RMLx>位。

通过内部逻辑，CANRML<RMLx>位被设置，也可以通过从 CPU 向 CANRMP<RMPx>位中写入 “1“使其被清除。同时，<RMPx>位也被清除。从 CPU 向 <RMLx>位中写入 “1“或 “0“ 是无效的。

随着 CANRMP<RMPx>位被设置为 “1“，如果邮箱×接收到以下信息，在接收信息丢失寄存器 CANRML 中相应的 <RMLx> 位被设置 “1“。在这种情况下，邮箱×被新接收的信息所重写。

在全局中断标志寄存器 CANGIF 中的<TRMABF>位也被设置为 “1“，通过设置全局中断屏蔽寄存器 CANGIM 中的<TRMABM>位到 “1“启用传输中止中断，发生 CAN 全局中断 INTCANGB。

当通过启用在全局中断屏蔽寄存器中的<RMLIM>位设置到 “1“启用接收信息丢失中断时，发生 CAN 全局中断 INTCANGB。

表 16-1 显示出在接收信息前后的 CANRMP 和 CANRML 寄存器的变化。

表 16-1 显示出接收信息前后的 RMP 和 RML 寄存器变化

ID	接收前		接收后		运行
	<RMPx>	<RMLx>	<RMPx>	<RMLx>	
不匹配	忽略	忽略	忽略	忽略	接收到的信息没有被存储在任何邮箱中。
匹配	0	0	1	0	接收到的信息被存储在与之匹配的ID的邮箱中。
	1	0	1	1	接收到的信息被重写入与之匹配的ID的邮箱中。
	1	1	1	1	这表明先前的信息已丢失。

16.4.14 CANRFP (远程帧悬挂寄存器)

	31	30	29	28	27	26	25	24
比特符号	RFP31	RFP30	RFP29	RFP28	RFP27	RFP26	RFP25	RFP24
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	RFP23	RFP22	RFP21	RFP20	RFP19	RFP18	RFP17	RFP16
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	RFP15	RFP14	RFP13	RFP12	RFP11	RFP10	RFP9	RFP8
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	RFP7	RFP6	RFP5	RFP4	RFP3	RFP2	RFP1	RFP0
复位后	0	0	0	0	0	0	0	0

位	比特符号	型号	功能
31-0	RFP31 ~ RFP0	R/W	远程帧等待 (每个位都与邮箱31 ~ 0相对应。) 当配置为接收的邮箱×接收到一个远程帧，<RFPx> 位和CANRMP<RMPx>位被设置为 “1“。 通过向CANRMP<RMPx>位写入 “1“，<RFPx> 位被清除。

通过内部逻辑，CANRFP<RFPx>位被设置，通过从 CPU 向 CANRMP<RMPx>位写入 “1“使其被清除。同时，<RMPx>位也被清除。从 CPU 向<RMPx>位里写入 “0“或向<RFPx>位里写入 “1“或 “0“都是无效的。

即使当<RFPx>= “1“的邮箱×通过数据帧接收被重写，<RFPx>位也能被清除。

当通过设置全局中断屏蔽寄存器中的<RFPM>位到 “1“来启动远程帧等待中断时，将会出现 CAN 全局中断 INTCANGB。

16.4.15 CANLAM (局部接收屏蔽寄存器)

局部接收屏蔽寄存器 CANLAM 仅被使用来为邮箱 31 的接收信息 ID 进行筛选。这种功能允许对邮箱 31 的接收信息的任何 ID 位进行局部屏蔽。

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

位	比特符号	型号	功能
31	LAMI	R/W	邮箱31的<IDE> 位屏蔽 0:未屏蔽 1:被屏蔽 在<LAMI>= “0“的情况下，根据邮箱31的<IDE> 位，标准或扩展格式的信息被接收。 在<LAMI>= “1“的情况下，不管邮箱31的<IDE>位的情况如何，标准或扩展格式的信息均被接收。
30-29	−	R	读取 ：读作 “0“。 写入 ：写作 “0“。
28-0	LAM[28:0]	R/W	接收信息ID屏蔽 0:未屏蔽 当接收信息ID的相应位和邮箱ID一致时，接收信息被接收。 1:被屏蔽 不管接收信息相应位的数值情况，接收信息被接收。

在扩展格式下，< ID[28:0] >和< LAM[28:0] >被用于筛选。

在标准格式下，< ID[28:18] >和< LAM[28:18] >被用于筛选。

当标准格式的信息被接收， 扩展 ID (<ID[17:0]>)将变成一个未定义值。因此，标准和扩展格式不能被推荐让相同的邮箱交替接收。

请在初始化条件 (配置模式下)设置 CANLAM，在运行过程中不能改变设置。接收信息的时候，设置被改变，在更改设置过程中的 CANLAM 数值被使用作为接收信息 ID 筛选。

16.4.16 CANGAM (全局接收屏蔽寄存器)

全局接收屏蔽寄存器 CANGAM 将用来进行邮箱 0～30 的接收信息的筛选。这种功能允许对邮箱 0～30 的接收信息的任何 ID 位进行全局屏蔽。

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

位	比特符号	型号	功能
31	GAMI	R/W	邮箱0 ~ 30的 <IDE>位屏蔽 0:未屏蔽 1:被屏蔽 在<GAMI> = “0“的情况下，根据邮箱0 ~ 30的<IDE>位，标准或扩展格式的信息被接收。 在<GAMI> = “1“的情况下，不管邮箱0 ~ 30的<IDE> 位的情况如何，标准或扩展格式的信息均被接收。
30-29	−	R	读取：读作 “0“ 写入：写作 “0“。
28-0	GAM[28:0]	R/W	接收信息ID屏蔽 0:未屏蔽 当接收信息ID的相应位和邮箱ID一致时，接收信息被接收。 1:被屏蔽 不管接收信息相应位的数值情况，接收信息被接收。

在扩展格式情况下，< ID[28:0] > 和< GAM[28:0] >被用于筛选。

在扩展格式情况下，< ID[28:18] >和< GAM[28:18] >被用于筛选。

当标准格式的信息被接收， 扩展 ID (<ID[17:0]>)将变成一个未定义值。因此，标准和扩展格式不能被推荐来让相同的邮箱交替接收。

请在初始化条件 (配置模式)下设置 CANGAM，在运行过程中不能改变设置。接收信息的时候，设置被改变，在设置更改过程中的 CANLAM 数值被使用作为接收信息 ID 筛选。

16.4.17 CANMCR (主控寄存器)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	SUR	-	TSTLB	TSTERR
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	CCR	SMR	-	WUBA	MTOS	-	TSCC	SRES
复位后	1	0	0	0	0	0	0	0

位	比特符号	型号	功能
31-12	−	R	读取：读作“0“。 写入：写作“0 “。
11	SUR	R/W	挂起模式请求 0:取消挂起模式 (正常运行) 1:请求挂起模式
10	−	R	读取：读作“0“。 写入：写作“0“。
9	TSTLB	R/W	测试回环 0:取消测试回环模式 (正常运行) 1:请求测试回环模式 (此模式支持独立操作)
8	TSTERR	R/W	测试错误 0:取消测试错误模式 (正常运行) 1:请求测试错误模式 (在此模式下，可以写入CAN错误计数寄存器 (CANCEC)).
7	CCR	R/W	更改配置请求 0:取消配置模式 (正常运行) 1:请求配置模式 (在此模式下，可以写入位配置寄存器，CANBCR1和CANBCR2。)
6	SMR	R/W	休眠模式请求 0:取消休眠模式 (正常运行) 1:请求休眠模式 (在此模式下，CAN控制器的时钟停止，计数错误并且传输请求被复位。)
5	−	R	读数：读作“0“。 写入：写作“0 “。
4.	WUBA	R/W	唤醒总线活动 0:仅在对CAMCR寄存器进行写入访问的时候才被唤醒。 1:当监测到总线处于活动状态或对CAMCR进行写入访问时被唤醒
3	MTOS	R/W	邮箱传输顺序选择 0:邮箱按照其号码的升序来传输信息。 1:照信息ID的优先级别来传输邮箱中的信息。
2	−	R	读取：读作“0“。 写入：写作“0“。
1	TSCC	R/W	时间戳计数清 0: 禁用 1:清除时间戳计数值为“0“。 (注1) 此位仅用作写入，读取时总是显示“0“。

位	比特符号	型号	功能
0	SRES (注释2)	R/W	软件复位 0:禁用 1:利用软件复位CAN控制器。 此位仅用作写入，读取时总是显示 “0“。

注 1:通过写入 CANTSP 寄存器和向 CANTSP 寄存器写入 “0“，时间戳计数都将被清零。

注 2:软件复位后，在接下来的时间里 CAN 中所有寄存器必须能被访问。

(1) 当 CAN 总线没有执行通信，请至少等待 16 个 CPU 时钟。

(2) 当 CAN 总线执行通信，请至少等待 88 个 CPU 时钟。

16.4.18 CANBCR1 (位配置寄存器 1)

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

位	比特符号	型号	功能
31-10	−	R	读取：读作 “0“。 写入：写作 “0“。
9-0	BRP[9:0]	R/W	波特率预定标器 设定值:0 ~ 1023

16.4.19 CANBCR2 (位配置寄存器2)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	SJW	
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	SAM	TSEG2			TSEG1			
复位后	0	0	0	0	0	0	0	0

位	比特符号	型号	功能
31-10	−	R	读取 : 读作 “0“。 写入 : 写作 “0 “。
9-8	SJW[1:0]	R/W	重新同步跳宽。 00 : 1 × TQ 01 : 2 × TQ 10 : 3 × TQ 11 : 4 × TQ
7	SAM	R/W	设置采样计数 0 : 单个采样 1 : 三重采样
6-4	TSEG2[2:0]	R/W	完成样本点后位时间的设置 000 : 预留 100 : 5 × TQ 001 : 2 × TQ 101 : 6 × TQ 010 : 3 × TQ 110 : 7 × TQ 011 : 4 × TQ 111 : 8 × TQ
3-0	TSEG1[3:0]	R/W	完成样本点前位时间的设置 (除SYNCSEG外)。 0000 : 预留 1000 : 9 × TQ 0001 : 2 × TQ 1001 : 10 × TQ 0010 : 3 × TQ 1010 : 11 × TQ 0011 : 4 × TQ 1011 : 12 × TQ 0100 : 5 × TQ 1100 : 13 × TQ 0101 : 6 × TQ 1101 : 14 × TQ 0110 : 7 × TQ 1110 : 15 × TQ 0111 : 8 × TQ 1111 : 16 × TQ

16.4.20 CANTSC (时间戳计数寄存器)

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

位	比特符号	型号	功能
31-16	–	R	读取：读作 “0“。 写入：写作 “0 “。
15-0	TSC[15:0]	R/W	时间戳计数器 自运行的16-位计数器

通过全局中断标志寄存器 (CANGIF)的时间戳计数器溢出中断标志<TSOIF>和全局状态寄存器(CANGSR)的时间戳溢出标志<TSO>，可以检测到CANTSC的溢出情况。通过向CANGIF寄存器中的<TSOIF>写入 “1“可以清除这两个标志位。

CANTSC 有一个 4-位的预定标器通电后，时间戳计数器直接从位时钟 (<TSP[3:0]>= “0“)开始驱动。时间戳计数器的周期 T_{TSC} 将通过下列公式计算:

T_{TSC} = TBIT x (TSP + 1)

16.4.21 CANTSP (时间戳计数预定标器寄存器)

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

位	比特符号	型号	功能
31-4	−	R	读取：读作 “0“。 写入：写作 “0“。
3-0	TSP[3:0]	R/W	时间戳计数预定标器 设置4-位TSC的值加载到预定标器

为了保证 CANTSC 值在向邮箱写入的周期内不会改变，必须使保持寄存器生效。如果信息已接收或成功传输，CANTSC 的值将被复制到保持寄存器,然后从保持寄存器写入到邮箱。如果帧结尾的最后一位以外都没有错误，那么接收器将成功进行接收。如果直到帧结尾的最后一位都没有错误，那么传输器将成功进行传输。 (参照 CAN 说明 2.0B)

16.4.22 CANGSR (全局状态寄存器)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	误
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	MIS				RM	TM	-	SUA
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	CCE	SMA	-	-	TSO	BO	EP	EW
复位后	1	0	0	0	0	0	0	0

位	比特符号	型号	功能
31-17	−	R	读取 ：读作 “0“。 写入 ：写作 “0“。
16-12	MIS[4:0]	R	槽内信息 显示位于传输缓冲区的信息的邮箱号码。 00000 :信息邮箱 0　01011 :信息邮箱 11　10110 :信息邮箱 22 00001 :信息邮箱 1　01100 :信息邮箱 12　10111 :信息邮箱 23 00010 :信息邮箱 2　01101 :信息邮箱 13　11000 :信息邮箱 24 00011 :信息邮箱 3　01110 :信息邮箱 14　11001 :信息邮箱 25 00100 :信息邮箱 4　01111 :信息邮箱 15　11010 :信息邮箱 26 00101 :信息邮箱 5　10000 :信息邮箱 16　11011 :信息邮箱 27 00110 :信息邮箱 6　10001 :信息邮箱 17　11100 :信息邮箱 28 00111 :信息邮箱 7　10010 :信息邮箱 18　11101 :信息邮箱 29 01000 :信息邮箱 8　10011 :信息邮箱 19　11110 :信息邮箱 30 01001 :信息邮箱 9　10100 :信息邮箱 20　11111 :传输缓存区内无信息 01010 :信息邮箱 10　10101 :信息邮箱 21
11	RM	R	接收模式 0:CAN 控制器未接收信息 1:CAN 控制器接收信息
10	TM	R	传输模式 0:CAN 控制器未传输信息 1:CAN 控制器传输信息
9	−	R	读取:读作 “0“。 写入:写作 “0“。
8	SUA	R	挂起模式应答 0:CAN 控制器未处于挂起模式 1:CAN 控制器处于挂起模式
7	CCE	R	允许更改配置 0:CAN 控制器未处于配置模式 1:CAN 控制器处于配置模式 在这种模式下，可以对位配置寄存器，CANBCR1 和 CANBCR2 进行写入。
6	SMA	R	休眠模式应答 0:CAN 控制器未处于休眠模式。 1:CAN 控制器处于休眠模式。 在这种模式下，CAN 寄存器的时钟停止，计数错误且传输请求被复位。
5-4	−	R	读取 ：读作 “0“。 写入 ：写作 “0“ 。
3	TSO	R	时间戳溢出 0:时间戳计数未溢出。 1:在该位最后被清除到“0“之后，时间戳计数至少已经溢出一次。为了清除这一位，需要清除 CANGIF 寄存器中的<TSOIF>位到“0“。

位	比特符号	型号	功能
2	BO	R	总线关闭状态 0:总线打开状态 (正常运行) 1:总线关闭状态 当CAN总线频繁无规律的发生错误，并且传输错误计数器<TEC>达到上限值256时，CAN控制器进入总线关闭状态。当没有信息被传输和接收时，错误计数器未定义。当总线脱离恢复序列后，CAN控制器自动进入总线打开状态。
1	EP	R	消极报错状态 0:CAN控制器未处于消极报错模式。 1:CAN控制器处于消极报错模式。
0	EW	R	警告状态 0:<TEC> 和<REC>的值都是小于或等于96。 1:如果<TEC> 和<REC>值中有一个超过96，就已经达到的警告标准。

16.4.23 CANCEC (错误计数寄存器)

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

位	比特符号	型号	功能
31-16	–	R	读取 ：读作 “0“。
			写入 ：写作 “0 “。
15-8	TEC[7:0]	R	8-位传输错误计数器 (复位释放后)
		R/W	8-位传输错误计数器 (CANMCR<TSTERR>= “1“)
7-0	REC[7:0]	R	8-位接收错误计数器 (复位释放后)
		R/W	8-位接收错误计数器 (CANMCR<TSTERR>= “1“)

CAN 控制器包括两个报错计数器:接收错误报错计数器<REC>和传输报错计数器<TEC>。这两个报错计数器的值都可从 CPU 中读出。对报错计数器的写入访问只能在测试错误模式下 (在 CANMCR 寄存器中的<TSTERR>位是 “1“)运行。在对 CANCEC 寄存器进行写入情况下，小于 8 位的<REC>数据被写入，高于 8 位的 (TEC)也会被写入。

根据 CAN 说明 2.0B，CAN 报错计数器向上或向下进行计数。

在超过消极报错上限值 (128)之后，<REC>不会增加。当<REC>=128，在信息正确接收后，<REC>设定值被设置为介于 119 和 127 的一个数值。在达到 “总线关闭 “状态后，两个报错计数器未被定义。

如果已经达到 “总线关闭“状态，在总线上出现 11 个连续的空闲位之后，接收报错计数器将会上升。如果错误计数器达到 128，模块自动更换到误激活状态。所有内置标志位被复位，报错计数器将被清除到 “0“。配置寄存器将保存设定值。在 “总线关闭“状态过程中，计数器的值未被定义。

当 CAN 进入配置模式，报错计数器将被清除。

16.4.24 CANGIF (全局中断标志寄存器)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	RFPF	WUIF	RMLIF	TRMABF	TSOIF	BOIF	EPIF	WLIF
复位后	0	0	0	0	0	0	0	0

位	比特符号	型号	功能
31-8	–	R	读取:读作 “0“。 写入:写作 “0“。
7	RFPF	R/W	远程帧待定标志位 0:未接收到远程帧。 1:接收到远程帧。 (在接收邮箱中) 因<TFH>位到 “1“，位将不会在与传输邮箱匹配时被设置。
6	WUIF	R/W	唤醒中断标志 0:休眠模式或正常运行模式 1:休眠模式已经被取消。
5	RMLIF	R/W	接收信息丢失中断标志 0:没有发生接收信息丢失的错误。 1:在接受邮箱中， 至少有一个邮箱会出现接收信息丢失的错误.
4.	TRMABF	R/W	传输中止标志位 0:无发生传输中止。 1:发生运送中止。 (至少有1位在CANAA寄存器中被设置。)
3	TSOIF	R/W	时间戳计数溢出中断标志位 0:在这个位被清除之后，在时间戳计数器中未出现溢出现象。 1:在这个位被清除之后，在时间戳计数器中至少出现一次溢出现象。
2	BOIF	R/W	总线关闭中断标志位 0: CAN控制器处于总线打开模式。 1: CAN控制器处于总线关闭模式。
1	EPIF	R/W	消极报错中断标志位 0: CAN控制器处于误激活模式。 1: CAN控制器处于消极报错模式。
0	WLIF	R/W	警告级别中断标志位 0:错误计数器都未达到警告级别。 1: 至少有一个错误计数器达到警告级别。

如果满足相应的全局中断条件，全局中断标志位寄存器 (CANGIF)的每一个中断标志位将被设置到 “1“。当全局中断标志位被设置到 “1“时，如果全局中断屏蔽寄存器 (CANGIM)中相应的位到 “1“ (启用中断)，CAN 全局中断 (INTCANGB) “高“。

通过向 CANGIF 寄存器中相应的位中写入 “1“来清除 CANGIF 寄存器。向其中写入 “0“是无效的。

16.4.25 CANGIM (全局中断屏蔽寄存器)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	RFPFM	WUIM	RMLIM	TRMABF	TSOIM	BOIM	EPIM	WLIM
复位后	0	0	0	0	0	0	0	0

位	比特符号	型号	功能
31-8	−	R	读取 ：读作 “0“。 写入 ：写作 “0“。
7	RFPM	R/W	远程帧待定中断屏蔽 0:中断禁用 1:中断启用
6	WUIM	R/W	唤醒中断屏蔽 0:中断禁用 1:中断启用
5	RMLIM	R/W	接收信息丢失中断屏蔽 0:中断禁用 1:中断启用
4	TRMABF	R/W	传输中止中断屏蔽 0:中断禁用 1:中断启用
3	TSOIM	R/W	时间戳计数溢出中断屏蔽 0:中断禁用 1:中断启用
2	BOIM	R/W	总线关闭中断屏蔽 0:中断禁用 1:中断启用
1	EPIM	R/W	消极报错中断屏蔽 0:中断禁用 1:中断启用
0	WLIM	R/W	警告级别中断屏蔽 0:中断禁用 1:中断启用

全局中断屏蔽寄存器(CANGIM)控制着是否启用或禁用相应的 CANGIF 寄存器的每一个中断条件的全局中断。当 CANGIF 寄存器中的位到 “0“时，相应的 CAN 全局中断 (INTCANGB)被禁用。当 CANGIF 寄存器中的位到 “1“，相应的 CAN 全局中断 (INTCANGB)被启用。

复位操作清除 CANGIM 寄存器中所有的位到“0“来禁用全局中断。

16.4.26 CANMBIM (邮箱中断屏蔽寄存器)

	31	30	29	28	27	26	25	24
比特符号	MBIM31	MBIM30	MBIM29	MBIM28	MBIM27	MBIM26	MBIM25	MBIM24
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	MBIM23	MBIM22	MBIM21	MBIM20	MBIM19	MBIM18	MBIM17	MBIM16
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	MBIM15	MBIM14	MBIM13	MBIM12	MBIM11	MBIM10	MBIM9	MBIM8
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	MBIM7	MBIM6	MBIM5	MBIM4	MBIM3	MBIM2	MBIM1	MBIM0
复位后	0	0	0	0	0	0	0	0

位	比特符号	型号	功能
31-0	MBIM31 ~ MBIM0	R/W	邮箱中断屏蔽 0:禁用相应的邮箱中断 1:启用相应的邮箱中断

CANMBIM 中的设定值决定着邮箱中断产生的启用或禁用。如果 CANMBIM 中的一个位为 “0“，相应邮箱的中断产生被禁用，如果 CANMBIM 中的位为 “1“，相应邮箱的中断产生被启用。复位 CANMBIM 值为 “0“。

16.4.27 CANMBTIF (邮箱传输中断标志寄存器)

	31	30	29	28	27	26	25	24
比特符号	-	MBTIF30	MBTIF29	MBTIF28	MBTIF27	MBTIF26	MBTIF25	MBTIF24
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	MBTIF23	MBTIF22	MBTIF21	MBTIF20	MBTIF19	MBTIF18	MBTIF17	MBTIF16
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	MBTIF15	MBTIF14	MBTIF13	MBTIF12	MBTIF11	MBTIF10	MBTIF9	MBTIF8
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	MBTIF7	MBTIF6	MBTIF5	MBTIF4	MBTIF3	MBTIF2	MBTIF1	MBTIF0
复位后	0	0	0	0	0	0	0	0

位	比特符号	型号	功能
31	−	R	读取：读作 “0“。 写入：写作 “0 “。
30-0	MBTIF30 ~ MBTIF0	R/W	邮箱传输中断标志位 (每个位都与邮箱30 ~ 0相对应。) 当邮箱×中的信息传输成功且CAN-MBIM寄存器中断屏蔽被启用 (<MBIMx>= “1“)时，<MBTIFx>位被设置为 “1 “，传输完成中断 (INTCANTX)变为 “高 “电平。 当CANMBIM<MBIMx>位为 “0“时，<MBTIFx> 位未被设置且INTCANTX保持在 “低“电平 通过读取CANTA寄存器进行传输完成的检查。 只要CANMBTIF寄存器中存在一个位为 “1“，那么INTCANTX就是 “高“级别。通过从CPU向<MBTIFx>中写入 “1“，<MBTIFx>的位被清除。 向其中写入 “0“值是无效的。

当邮箱设置为接收时，CANMBTIF 寄存器中相应的位被读作 “0“值。当邮箱设置为传输时，CANMBRIF 寄存器中相应的位被读作 “0“值。

16.4.28 CANMBRIF (邮箱接收中断标志寄存器)

	31	30	29	28	27	26	25	24
比特符号	MBRIF31	MBRIF30	MBRIF29	MBRIF28	MBRIF27	MBRIF26	MBRIF25	MBRIF24
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	MBRIF23	MBRIF22	MBRIF21	MBRIF20	MBRIF19	MBRIF18	MBRIF17	MBRIF16
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	MBRIF15	MBRIF14	MBRIF13	MBRIF12	MBRIF11	MBRIF10	MBRIF9	MBRIF8
复位后	0	0	0	0	0	0	0	0
	7	6	5	4.	3	2	1	0
比特符号	MBRIF7	MBRIF6	MBRIF5	MBRIF4	MBRIF3	MBRIF2	MBRIF1	MBRIF0
复位后	0	0	0	0	0	0	0	0

位	比特符号	型号	功能
31-0	MBRIF31 ~ MBRIF0	R/W	邮箱接收中断标志位 (每个位都与邮箱31 ~ 0相对应。) 当邮箱×成功的接收信息且CANMBIM寄存器中断屏蔽被启用 (<MBIMx>= “1“)时，<MBRIFx>位被设置为 “1“，而且接收完成中断 (INTRX)变为 “高“电平。 当MBIM寄存器中<MBIMx>位为 “0“值时，<MBRIFx> 位未被设置且INTRX保持在 “低“电平。通过读取CANRMP寄存器来进行接收完成检查。 只要CANMBRIF寄存器中存在一个位为 “1“，那么INTCANRX就是 “高 “级别。通过从CPU向<MBRIFx>中写入 “1“，<MBRIFx>位被清除。 向其中写入 “0“值是无效的。

16.5 电路运行说明

16.5.1 邮箱

邮箱由单一端口 RAM (从内置 CAN 磁心和 CPU 进行访问)组成。CPU 通过更改邮箱的设置和控制寄存器来控制 CAN 控制器。邮箱和控制寄存器的设置被用来处理接收筛选，信息传输和中断进程等操作。

为了启动传输，对与传输信息指定的邮箱相对应的传输请求位进行设置。在位被设置后，所有的传输程序和错误进程 (当发生错误时)将在 CPU 不参与的情况下执行。当邮箱被设置为接收时，CPU 通过读取指令来读取邮箱数据。用户同样可以通过设置来实现:当信息已成功的接收或传递时，一个中断指令将向 CPU 发出。

总共有 32 个邮箱被提供，每个邮箱包含 8 字节数据，29 个位 ID 和多个控制位。邮箱 (除邮箱 31 外)都能被设置为传输或接收。邮箱 31 为只接收邮箱.与邮箱 0～30 相比，邮箱 31 能使用其他接收屏蔽来接收不同的信息 ID 群组，为此设计邮箱 31。

图 16-2 显示出邮箱的配置。

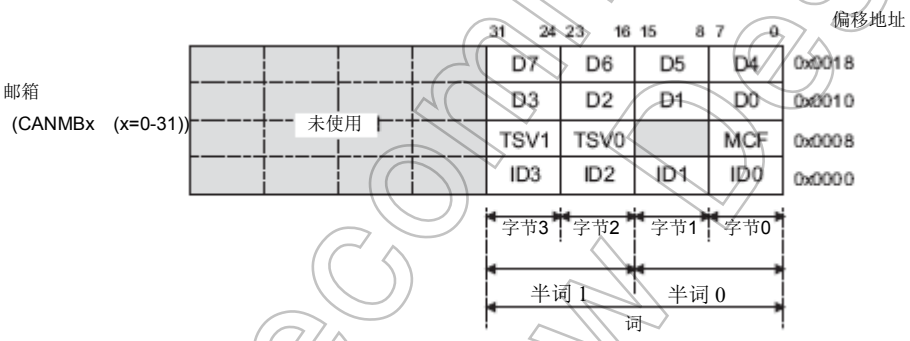


图 16-2 邮箱的配置

1. 信息 ID 字段 (ID3～ID0)

   ID 扩展位<IDE>
   全局/局部接收屏蔽允许位<GAME / LAME>
   远程帧处理位<RFH>
   29 位的信息 ID <ID[28:0]>

2. 信息控制字段 (MCF)

   远程帧传输请求位<RTR>
   4 位的数据长度<DLC[3:0]>

3. 时间戳数值 (TSV1，TSV0)
   在接收/传输信息过程中储存了时间戳计数值。 (TSV[15:0]>

4. 数据字段 (D7～D0)
   8 位的数据<D7[7:0]>～<D0[7:0]>

16.5.2 传输控制寄存器

传输控制由两个寄存器组成。一个是传输请求设置寄存器 CANTRS,另一个是传输请求复位寄存器 CANTRR。因此，可以在状态机的传输邮箱处理中没有发生冲突的情况下，清除传输请求。这个原理也阻止了对正在传输的邮箱的传输请求的清除。

当写入数据和邮箱 X 的 ID 配置为传输邮箱 (CANMD<MDx>= “0“)被执行，对邮箱 X 的存取是允许的 (CANMC<MCx>= “1“)，设置 CANTRS<TRSx>的位值到“1“，引起邮箱 X 中的信息传输.

如果多于 1 个邮箱被配置为传输邮箱且多个相应的 TRS 位被设置，那么信息将按照选择的顺序被发出。传输顺序取决于主控寄存器 CANMCR 中<MTOS>的位。

如果 CANMCR<MTOS> 位为 “0“，较小数值的邮箱拥有较高的优先级别。例如,如果邮箱 CANMB0，CANMB2 和 CANMB5 被配置为传输邮箱且相应的 CANTRS<TRSx>位被设置到 “1“，那么信息按照下面的顺序进行传输:CANMB0，CANMB2 和 CANMB5。如果在 CANMB2 信息处理过程中给 CANMB0 设置一个新的传输请求，然后在接下来的内部仲裁运行中，CANMB0 将被作为下一个传输信息，并在完成 CANMB2 传输之后开始对其进行传输。当 CANMB2 信息正在被传输时发生仲裁丢失的错误，这个就将发生。CANMB0 信息将代替仲裁丢失的 CANMB2 被传输。

如果 CANMCR<MTOS> 位为 “1“，因传输被请求，在这些邮箱中，具有最高优先级别 ID 的邮箱将被传输。在发生仲裁丢失错误后的传输中，因此时传输被请求，拥有最高优先级别 ID 的邮箱中的信息将被传输。

16.5.3 接收控制寄存器

接收信息的 ID 与设置为接收邮箱的 ID 进行匹配。ID 的匹配取决于邮箱中全局/局部接收屏蔽允许位 MBnID3 的<GAME> / <LAME>值和全局/局部接收屏蔽寄存器 GAM / LAM 中存储的数据。

检测到匹配成功后，接收信息的 ID，控制位和数据位都将被写入匹配邮箱中。同时,当相应的接收信息待定位 CANRMP<RMPx>被设置到 “1“且邮箱中断被启用 (CANMBIM<MBIMx>= “1“), 那么将发生 CAN 接收完成中断 INTCANRX。检测到匹配成功后, 将不会继续进行 ID 匹配。

如果接收信息的 ID 没有与邮箱 0～30 匹配成功，此 ID 将会和只接收邮箱 31 进行匹配。检测到匹配成功后，接收信息的设置将被写入只接收邮箱 31。

如果未监测到匹配成功, 接收信息将不会被存储到邮箱中，邮箱也不会发生更改。

读取数据后，CPU 必须清除 <RMPx> 位。当<RMPx> 位被设置到 “1“时,如果邮箱×收到下一条信息，相应的接收信息丢失位<RMLx>将被设置到 “1“。在这种情况下, 邮箱×被新信息所重写。

图 16-3 显示出发生接收信息丢失的时间设置。

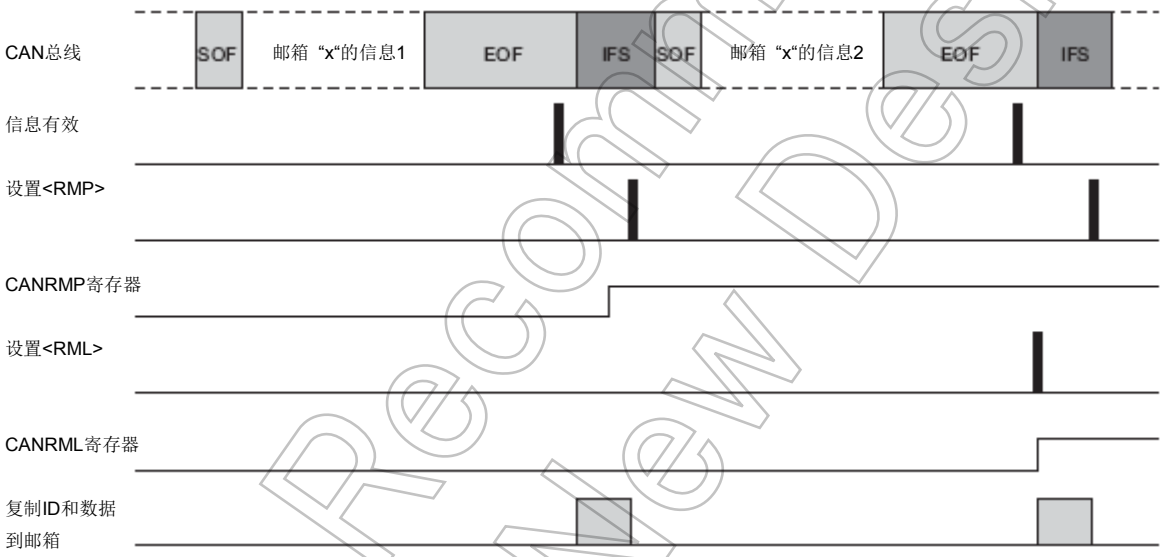


图 16-3 发生接收信息丢失的时间设置

16.5.4 远程帧控制寄存器

收到远程帧后，远程帧 ID 与邮箱 ID 进行匹配。ID 匹配取决于邮箱中全局/局部接收屏蔽允许位 CAN-MBxID 的<GAME> / <LAME>值以及全局/局部接收屏蔽寄存器 GAM/LAM 中存储的数据。

检测到 ID 匹配后，将不会继续进行 ID 匹配。

当远程帧 ID 与远程帧处理位 CANMBxID<RFH>设置到 “1“的传输邮箱 n 进行 ID 匹配时，CANTRS<TRSx>位将被设置到 “1“以使信息对远程帧进行响应并被传输。针对 CANMBxID<RFH>位被设置到 “0“的传输邮箱，即使 ID 匹配，邮箱也不会对远程帧作出响应。

如果 ID 与接收邮箱 n 的 ID 相匹配，接收的信息将和数据帧一样被处理，而且 CANRMP<RMPx>和 CANRFP<RFPx>位都将被设置到 “1“。

当远程帧 ID 与 CANMBxID<RFH>和<GAME>位都设置为 “1“的邮箱 n 的 ID 相匹配，邮箱 x 的 ID 将被远程帧的 ID 重写，并且这个邮箱将自动对使用此 ID (<TRSx>位被设置到传输一个数据帧)的远程帧作出响应。因此，当使用全局接收屏蔽寄存器 CANGAM 时，根据屏蔽数值的情况，一个邮箱 x 可以对多个远程帧 ID 作出响应。

16.5.5 接收筛选

针对邮箱 0～30，如果邮箱中的位<GAME>被设置，使用全局接收屏蔽寄存器 CANGAM。接收的信息将被存储在第一个与之 ID 匹配的邮箱里。除非邮箱 0～30 没有匹配 ID，接收信息将与只接收邮箱(邮箱 31)相匹配。如果邮箱 31 中的<LAME>位被设置, 会使用局部接收屏蔽寄存器 CANLAM。

图 16-4 显示出接收筛选。

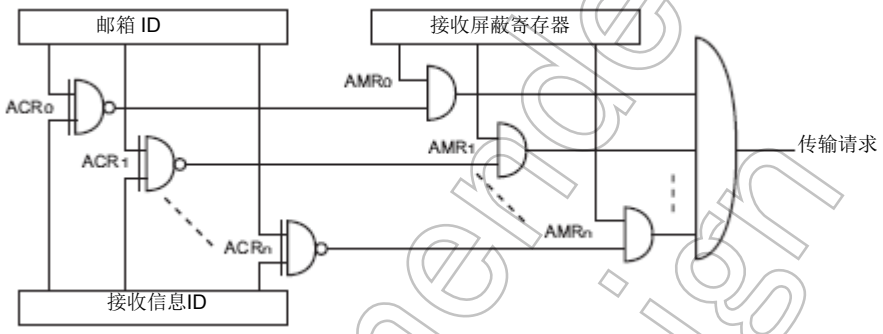


图 16-4 接收筛选

16.5.6 时间戳功能

在 CAN 控制器中一个自由运行 16 位时间戳计数器 (CANTSC)被执行来显示信息接收和传输时间。当收到的信息已被存储或一个信息已经被传输，CANTSC 的内容将被写入到相应邮箱的时间戳数值(TSV)里。

CANTSC 被 CAN 总线线路的位时钟驱动。当 CAN 的运算模式处于配置模式或睡眠模式，CANTSC 将被停止。通电复位后，对时间戳计数预分频寄存器 (CANTSP)的写入将清除 CANTSC 为 “0“。在配置模式和正常运算模式下，CPU 中的 CANTSC 是可读取和可写入的。

图 16-5 显示出时间戳计数器的结构。

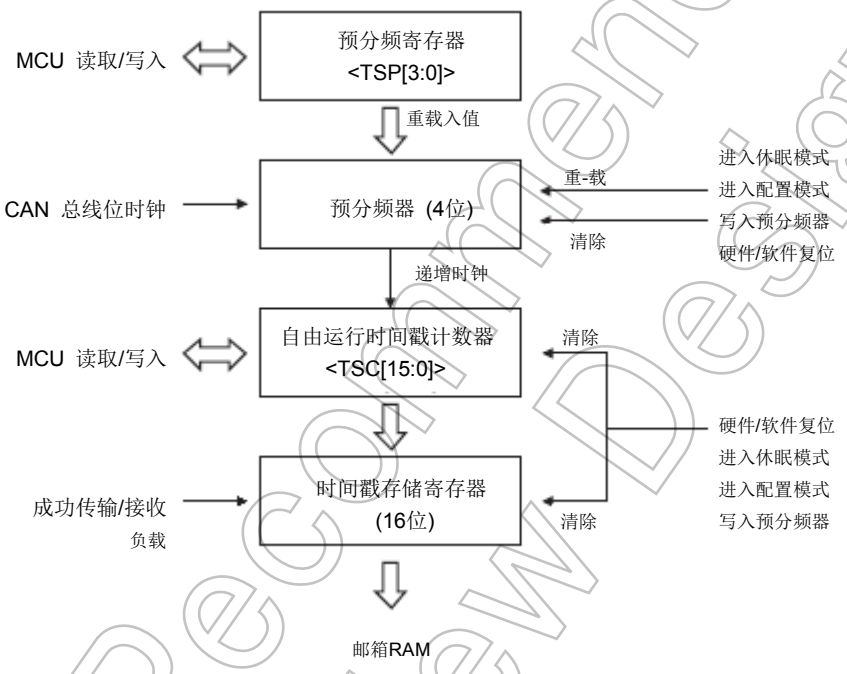


图 16-5 计时器标识计数器

在以下几种情况，自由运行的时间戳计数器和时间戳存储寄存器将被清除:

复位后 (开机复位或者软件复位)

当控制器进入配置模式

当控制器进入睡眠模式

当执行对 CANTSP 寄存器的写入访问。

16.5.7 中断控制

CAN 控制器有以下中断源。这些终端源分成 3 组，并且每组有一个中断输出。

CAN 传输完成中断 (INTCANTX)

传输完成时发生

CAN 接收完成中断 (INTCANRX)

接收完成时发生

CAN 全局中断 (INTCANGB)

发生在上述之外的 8 个源。

数据源		组
传输中断	:信息传输成功。	INTCANTX
接收中断	:信息接收成功。	INTCANRX
警告级别中断	:两个错误计数器中至少一个大于等于97。	INTCANGB
消极报错中断	:CAN进入消极报错模式。	
总线关闭中断	:CAN 进入总线关闭模式。	
时间戳溢出中断传输中止中断		
接收信息丢失中断		
唤醒中断	:从睡眠模式唤醒后，生成中断。	
远程帧接收中断		

由于邮箱中断，从全局中断分离出两个中断输出线路.邮箱接收完成中断 (INTCANRX)和邮箱传输完成中断 (INTCANTX)独立存在于邮箱设置中。

这里有两个中断标志位寄存器和一个中断屏蔽寄存器。一个中断标志寄存器是为了邮箱接收中断标志位寄存器 (CANMBRIF) ，另一个是为了邮箱传输中断标志位寄存器 (CANMBTIF)。此外, 邮箱中断屏蔽寄存器 (CANMBIM)用来设置是否启用或禁用每个邮箱设置中断。传输和接收的邮箱均可以使用 CANMBIM 寄存器。

图 16-6 显示出 CAN 中断信号的框图。

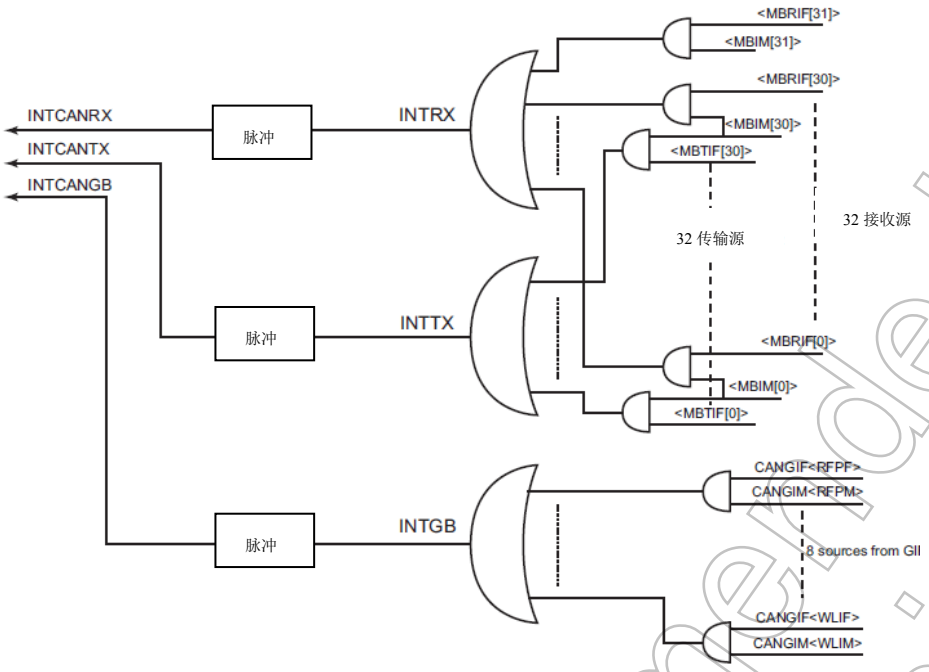


图 16-6 CAN 中断信号的框图

由于被 AND 邮箱中断屏蔽寄存器 CANMBIM 的每个位一起的邮箱接收中断标志寄存器 CANMBRIF 发出的 32 个信号源，CAN 接收完成中断信号 INTRX 是一个 OR 信号。

由于被 AND 邮箱中断屏蔽寄存器 CANMBIM 的每个位一起的邮箱传输中断标志寄存器 CANMBTIF 发出的 31 信号源，CAN 传输完成中断信号 INTTX 是一个 OR 信号。

由于被 AND 全局中断屏蔽寄存器 CANGIM 的每个位一起的全局中断标志寄存器 CANGIF 发出的 8 个信号源，CAN 全局中断信号 INTGB 是一个 OR 信号。

16.6 操作模式

16.6.1 配置模式

CAN 控制器在启动运行前需要初始设置 (位配置寄存器，CANBCR1 和 CANBCR2 的设置)。对 CANBCR1 和 CANBCR2 的写入只有当 CAN 控制器处于配置模式才是可行的。

复位之后，CANMCR<CCR> 和 CANGSR<CCE> 均被设置到 “1“，同时配置模式被设置。通过向 <CCR> 位中写入 “0“来设置 CAN 控制器为正常运行模式。离开配置模式后，<CCE> 位被清除到 “0“并启动通电序列。通电顺序监测到位于 CAN 总线线路上的 11 个连续隐性位。检测后,CAN 控制器的总线已打开，准备运行。

通过向<CCR>位中写到 “1“来设置 CAN 控制器从正常运行模式进入配置模式。CAN 控制器进入配置模式后，<CCE>位被设置到 “1“。

图 16-7 显示出 CAN 控制器初始设置的流程图。

当 CAN 控制器进入配置模式，CAN 错误计数器 (CANCEC)，时间戳计数器 (CANTSC)和时间戳存储寄存器将被清除。

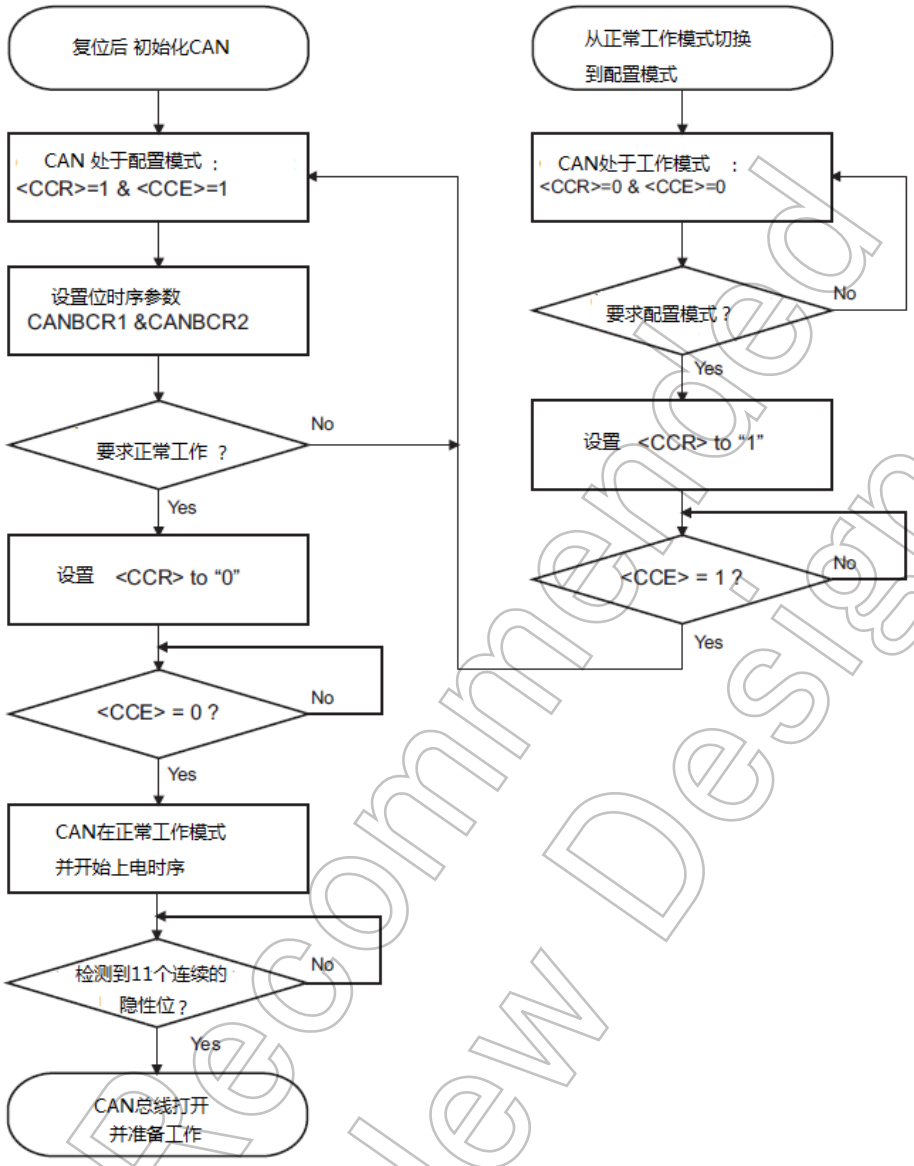


图 16-7 CAN 控制器的初始设置流程图

16.6.2 休眠模式

通过向 CANMCR 寄存器中的<SMR>位写入 “1“来实现休眠模式。在 CAN 控制器进入休眠模式以后， CANGSR<SMA>位被设置到 “1“。

CANGSR 寄存器的读取值为 0xF040。这意味着在传输缓存中没有信息，当<SMA>位为 “1“时，休眠模式被激活。向其他所有的寄存器传输的的读取值是 0x0000。除 CANMCR 寄存器外，对其他所有的寄存器的写入访问都将被忽略。

如果监测到写入 CANMCR 寄存器的访问，或者在 CANMCR <WUBA>设置到 “1“的 CAN 总线中监测到任何的总线活动，CAN 控制器将取消休眠模式 (被唤醒)并进入通电状态。CAN 控制器直到在 CANRX 输入终端监测到 11 个连续隐性位，它才开始进入总线活动状态。唤醒信息是无效的。

在休眠模式下，CAN 错误计数和所有的传输请求设置的 CANTRS<TRSx>位和传输请求重新设置的 CANTRR<TRRx>位将被清除。CAN 控制器取消休眠模式后，<SMR>位和<SMA>位将被清除。

当 CAN 控制器正在传输一个信息的时候 (CANMCR<SMR>= “1“)，如果休眠模式被请求， CAN 控制器在发现以下情况之后进入休眠模式:

信息已经被成功的传输。

在一个仲裁丢失错误之后，信息已经被成功的传输。

在一个仲裁丢失错误之后，信息已经被成功的接收。

16.6.3 挂起模式

通过向 CANMCR<SUR> 位中写入 “1“，挂起模式被请求。如果 CAN 总线线路没有闲置，目前的信息传输/接收将在激活挂起模式之前完成。在 CAN 控制器已经进入挂起模式之后，CANGSR<SUA>位被设置到 “1“。

在挂起模式下，CAN 控制器没有在 CAN 总线线路中被激活。这意味着错误帧和错误应答都将不会被传输。错误计数器和 CANGSR<EP>位将都不会被清除。

如果挂起模式被请求，在总线脱离恢复序列执行的过程中，CAN 控制器在脱离恢复序列完成之后进入挂起模式。

为了重新启动 CAN 控制器，<SUR> 位需要被编程为 “0“。在离开关闭状态的总线或未被激活的状态下，CAN 控制器重新启动脱离恢复序列的总线。

CAN 控制器通过向<SUR> 位中写入 “0“来取消挂起模式。

16.6.4 测试回环模式

在测试回环模式下， CAN 控制器可以接受自身发出的信息并产生自身的应答位。不需要其他的 CAN 节点使其运行。

只有在 CAN 控制器处于挂起模式下，测试回环模式才能够被启用或禁用。在测试回环模式下，CAN 控制器可以从一个邮箱传输一个信息并在另一个邮箱接收。邮箱的设置与正常运行下的设置一致。

16.6.5 测试错误模式

在测试错误模式下，向 CAN 错误计数寄存器 (CANCEC)进行写入操作是可行的。低于 8 位的数值被同时写入传输错误计数器 (CANTEC)和接收错误计数器 (CANREC)。被写入错误计数器的最大值是 255。强制从 CAN 控制器写入关闭模式的总线的错误计数值 256 不能被写入。

只有在 CAN 控制器处于挂起模式下，测试回环模式才能够被启用或禁用。

图 16-8 显示出测试回环模式和测试错误模式设置的流程图。

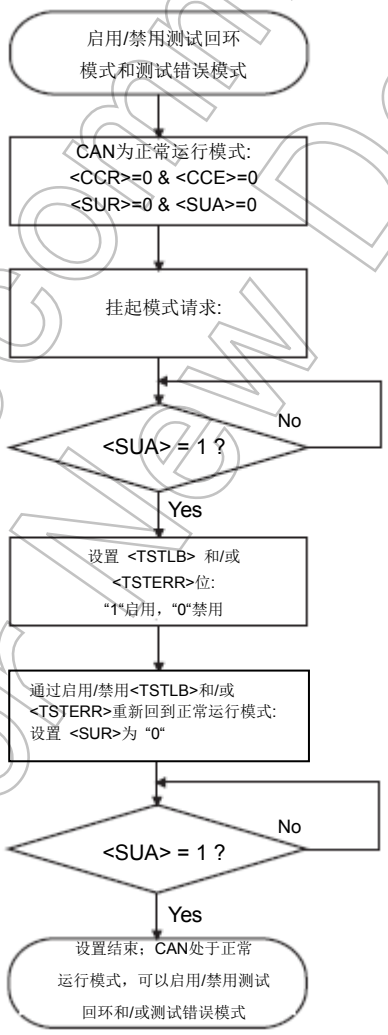


图 16-8 显示出测试回环模式和测试错误模式设置的流程图

16.7 操作说明

16.7.1 接收信息

图 16-9 显示出应用 CAN 接收完成中断 (INTCANRX)的信息接收的示意图。

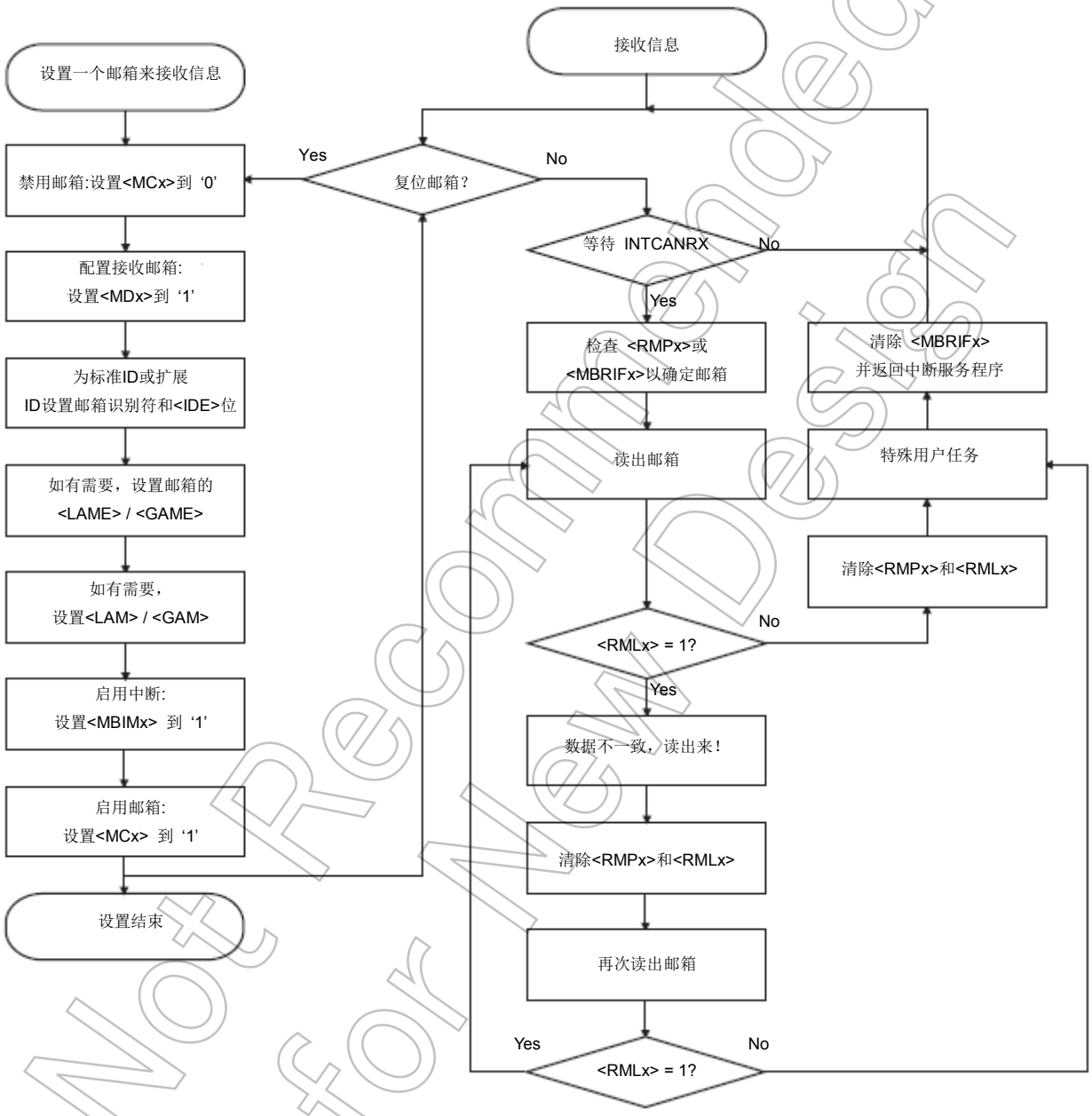


图16-9 信息接收流程图

定时询问代替接收中断也是可行的。在这种情况下，在流程图上方的 "等待 INTCANRX"必须被 CANRMP 定时询问所取代。而且，启用中断和清除 CANMBRIF 必须从流程图中删除。

16.7.2 传输信息

图 16-10 显示出示意图应用 CAN 接收完成中断 (INTCANRX)的信息传输。

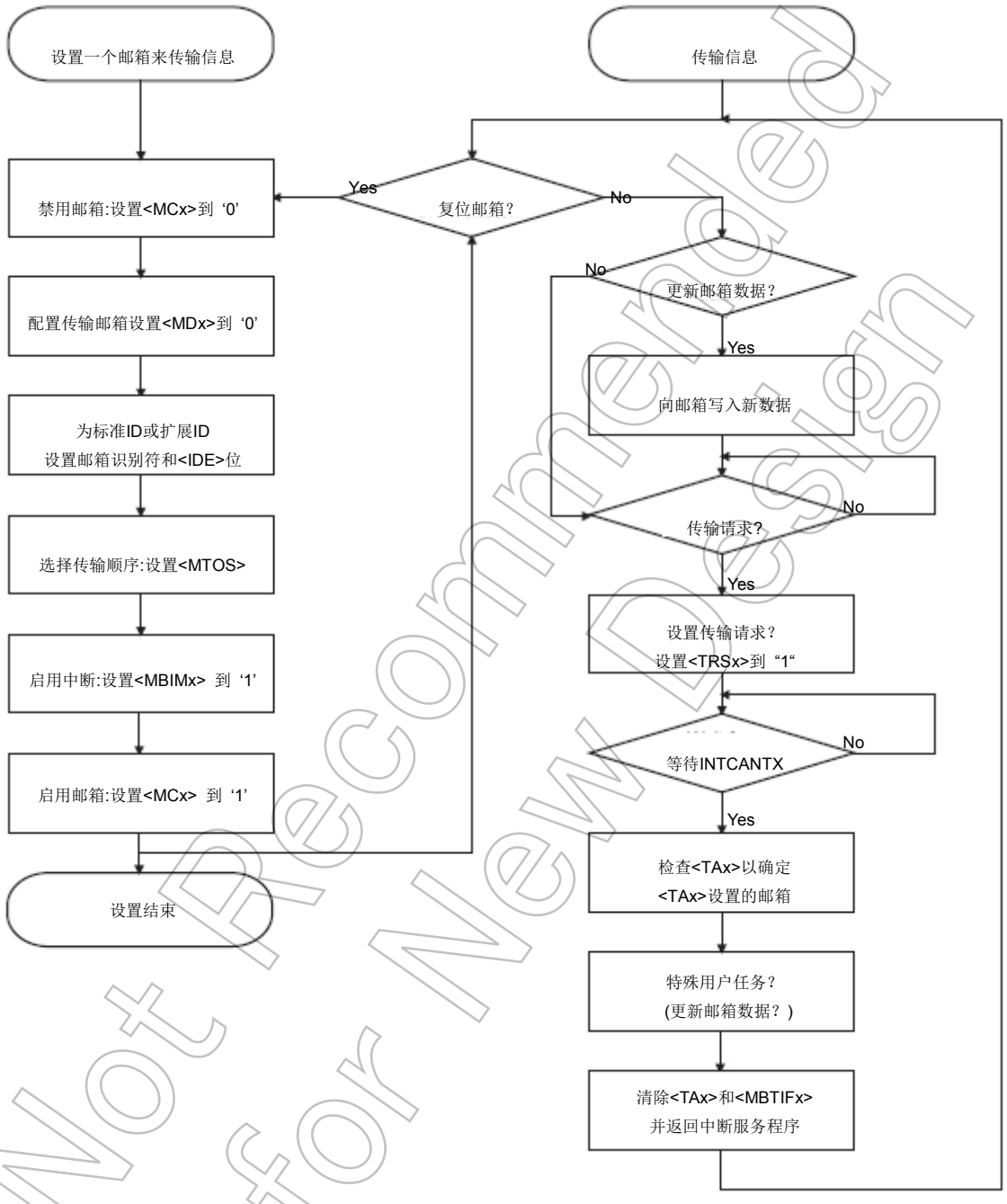


图 16-10 信息传输流程图

定时询问代替接收中断也是可行的。在这种情况下，在流程图上方的 "等待 INTCANRX "必须被定时询问 TA 所取代。而且，启用中断和清除 CANMBRIF 必须从流程图中删除。

16.7.3 远程帧处理

图 16-11 显示出通过使用自动回复功能来进行远程帧处理的示意图。当传输邮箱的<RFH>位被设置到 “1“时，这种功能是可用的。当更新邮箱数据时，为了避免数据不一致，CANCDR 寄存器在邮箱数据更新过程中控制着传输情况。

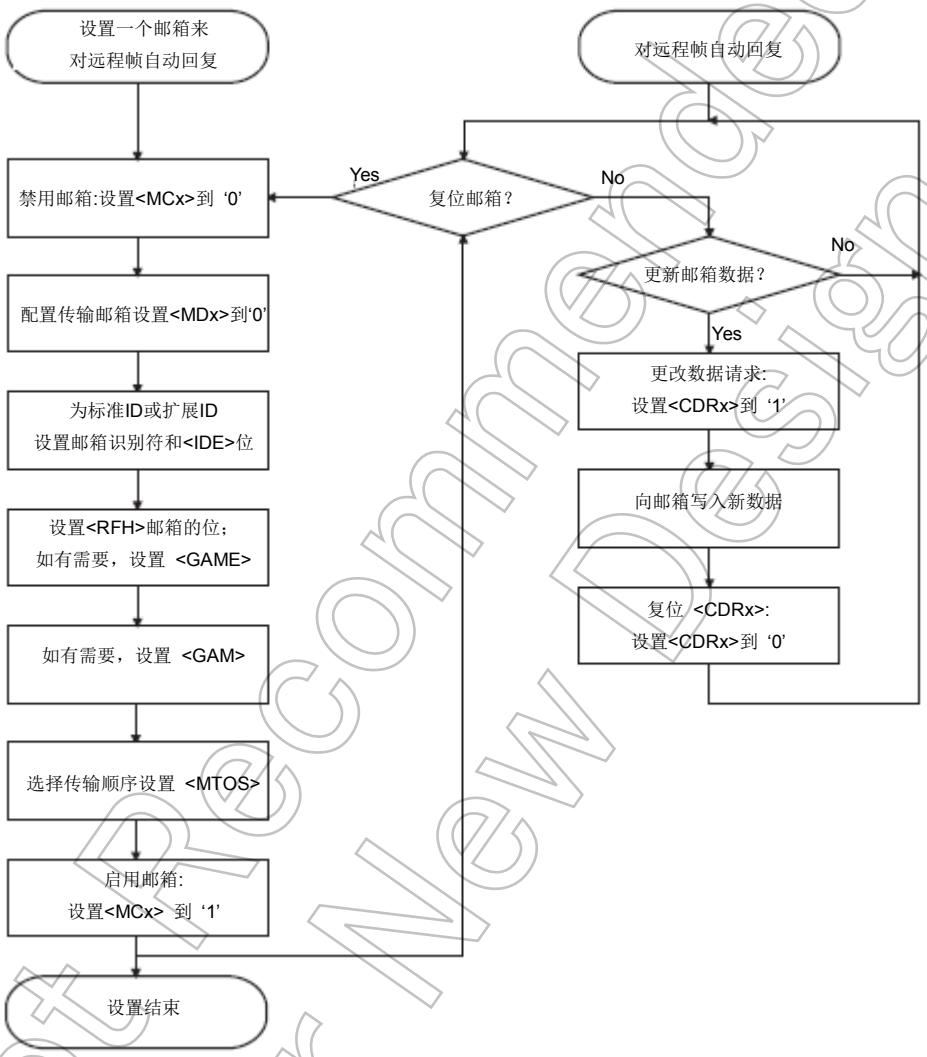


图 16-11 使用自动回复功能来进行远程帧处理的流程图

16.8 位配置

TSEG1，TSEG2 和 BRP 等参数决定着一个位的长度。所有在 CAN 总线上的控制器必须具有相同的传输速率和位长度。在个别控制器的不同时钟频率情况下，传输速率都必须被上述参数所调节。在位时序逻辑情况下，将执行向所需的位时序的参数转换。配置寄存器 CANBCR1 和 CANBCR2 包含关于位时序的数据。它的定义对应于 CAN 规格 2 (相当于 Intel 82527)。

图 16-12 显示出 CAN 位时序。

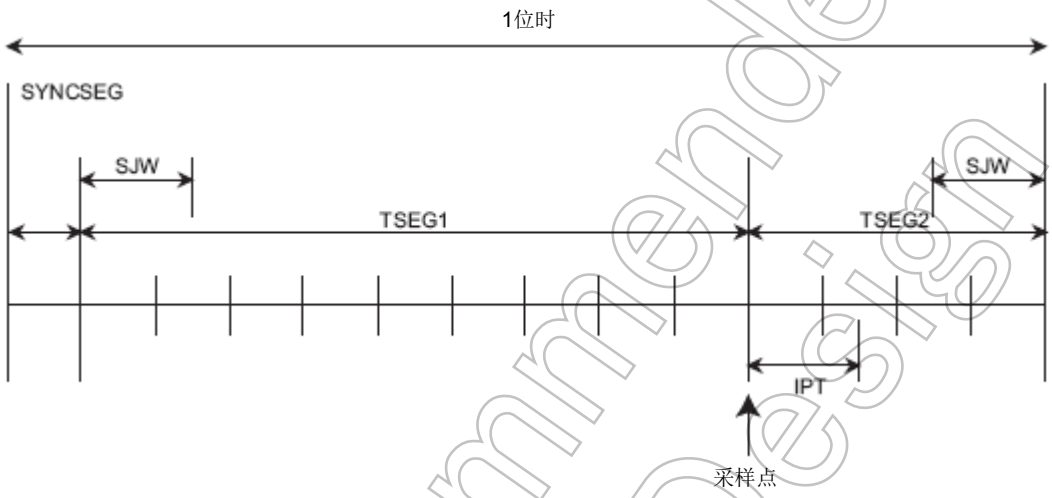


图 16-12 CAN 位时序

T_{SCL} (CAN 系统时钟)被定义为:

$$T_{SCL} = \frac{<BRP[9:0]>+1}{f_{CANOSC}}$$

$1 \times T_{SCL} = 1 \times T_Q$ (T_Q :时间量)

f_{CANOSC} 是为产生 CAN 传输速率的时钟。通过用系统时钟 f_{SYS} 除以 4 得到的时钟被应用为 CAN 传输速率产生器的时钟。如果 f_{SYS} = 48MHz 那么 f_{CANOSC} = 12MHz。

同一步段 SYNCSEG 总是具有一个 T_Q 的长度。

传输速率被定义为:

$$BR = \frac{1}{((<TSEG1[13:0]>+1)+(<TSEG2[2:0]>+1)+1)\times T_{SCL}}$$

注:<TSEG1[3:0]>和<TSEG2[2:0]>是 CANBCR2 寄存器的数值。它不是 T_Q 单位数值。

信息处理时间 (IPT)属于相同的步段，开始于因采样位级的处理而保存下来的采样点。信息处理时间等于 3 个 CAN 系统时钟周期。

<SJW[1:0]>表明当再次同步时，在位长度中，有多少时间量 (T_Q)数值允许被延长或缩短。介于“1“ (<SJW[1:0]> 00)和 “4“ (<SJW[1:0]> 11)之间的数值是可调整的。总线线路被采样，并在每个位表格里，对每个总线信号的下降沿执行同步。针对 <SJW[1:0]>，设置一个小于或等于<TSEG2[2:0]>的数值。

设置<SAM>位来启用总线线路的多重采样。三个采样值的多数决定产生的结果决定了它的级别。在样本点和先前最后 2 个 CAN 系统时钟点中进行取样。当<BRP[9:0]>小于 4，不管在<SAM>位中设置的数值情况，仅执行一次取样。表 16-2 显示出传输速率被设置时的限制条件

表 16-2 显示出传输速率被设置时的限制条件

<BRP[9:0]>	T_Q长度 (CAN时钟周期的次数)	IPT长度 (CAN时钟周期的次数)	最小TSEG2长度 (T_Q单位)
0	1	3	3
1	2	3	2
> 1	<BRP[9:0]>+1	3	2

TSEG1 的制约条件

TSEG1 ≥ TSEG2 : TSEG1 的长度应该大于或等于 TSEG2 的长度。

SJW 的限制条件

SJW ≤ TSEG2 : 针对同步跳转宽度，需要设置一个小于或等于 TSEG2 的数值。

SAM 的限制条件

在<BRP[9:0]>小于 4 的情况下，不允许进行三重采样。针对<BRP[9:0] < 4 的情况，不管 SAM 的数值情况，都只执行一次采样。

示例:针对 500 Kbit/s

一个位的长度是 2μs。如果 f_{OSC} = 12 MHz，传输速率预定标器被设置到 “1“ 。这意味着为了这些数据传输速率的一个位必须使用 $12T_Q$ 的长度来进行编程。根据以上公式，设计值总是比计算值小一点:

<BRP[9:0]> = 0y00_0000_0001

<TSEG1[3:0]> = 0y0110 (7 T_Q)

<TSEG2[2:0]> = 0y011 (4 T_Q)

在这种情况下，采样点是 8/12 66%。

随着 SJW，TSEG2 3 达到全频；对 TSEG1/TSEG2 的其他组合都是可能的。

SJW 应该尽量被设置到最大值。SJW 不允许大于 TSEG2。

因<BRP[9:0]>小于 4，不能设置总线的三重采样。因此，应该设置 SAM= “0“。

17. 远程控制信号预处理器 (RMC)

17.1 基本操作

远程控制信号预处理器 (以下 RMC)接收移去载波的远程控制信号。

17.1.1 远程控制信号的接收

- 采样时钟可选为低频率时钟 (32.768kHz)或定时器输出。
- 噪音消除时间可调整。
- 先导检测
- 批处理接收数据达 72 位。

17.2 方块图

图 17-1 为 RMC 的方块图

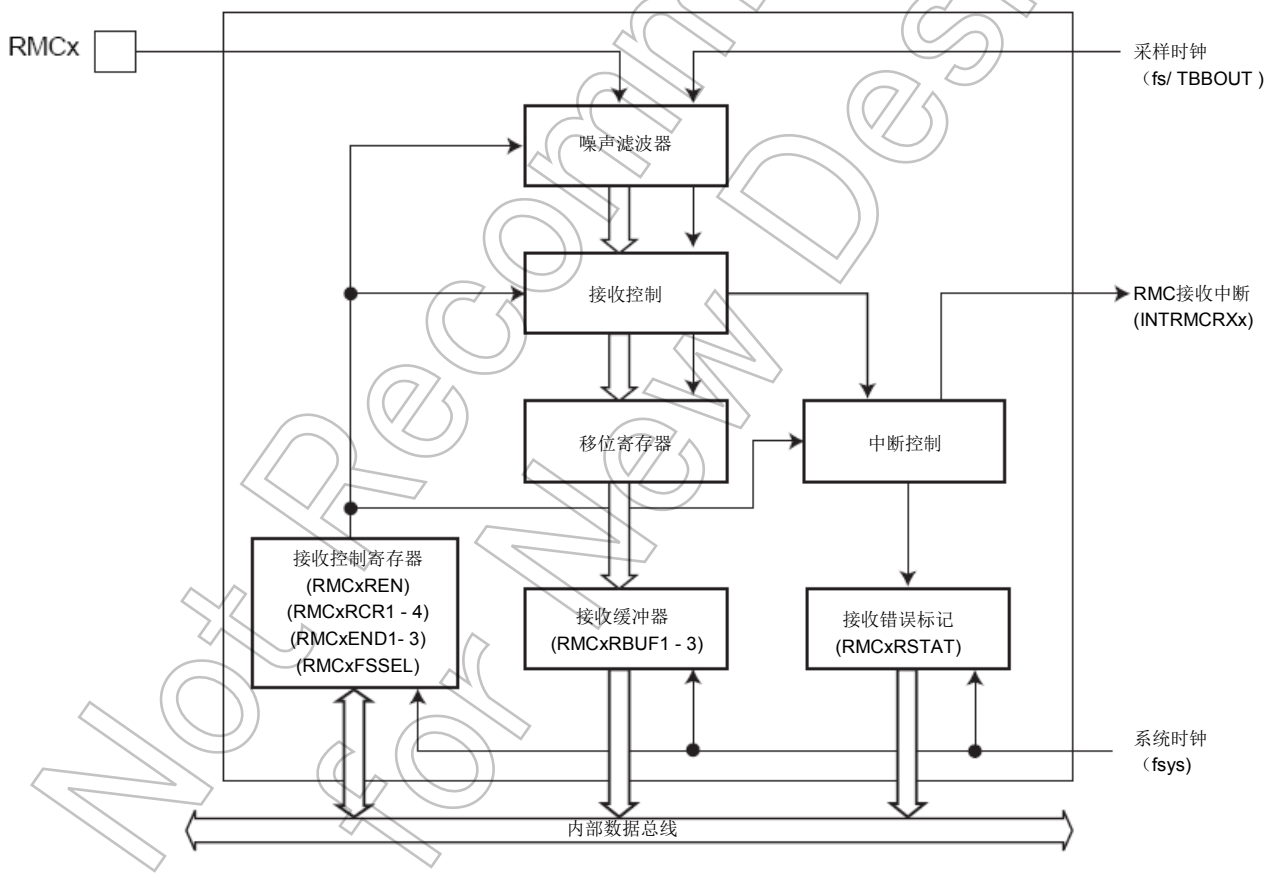


图 17-1 RMC 方块图

17.3 寄存器

17.3.1 寄存器列表

RMC 控制寄存器地址和名称如下所示。

通道 x	基址
通道 0	0x400E_3000
通道 1	0x400E_3100

寄存器 (x=0,1)		地址 (基+)
启用寄存器	RMCxEN	0x0000
接收启用寄存器	RMCxREN	0x0004
接收数据缓冲寄存器1	RMCxRBUF1	0x0008
接收数据缓冲寄存器2	RMCxRBUF2	0x000C
接收数据缓冲寄存器3	RMCxRBUF3	0x0010
接收控制寄存器 1	RMCxRCR1	0x0014
接收控制寄存器 2	RMCxRCR2	0x0018
接收控制寄存器 3	RMCxRCR3	0x001C
接收控制寄存器 4	RMCxRCR4	0x0020
接收状态寄存器	RMCxRSTAT	0x0024
接收终端位数寄存器 1	RMCxEND1	0x0028
接收终端位数寄存器 2	RMCxEND2	0x002C
接收终端位数寄存器 3	RMCxEND3	0x0030
源时钟选择寄存器	RMCxFSSEL	0x0034

17.3.2 RMCxEN (启用寄存器)

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

位	比特符号	型号	功能
31-2	−	R	读作 0。
1	−	R/W	写入数值为 “1“。
0	RMCEN	R/W	控制RMC操作 0:禁用 1:启用 为使RMC起作用，首先启用RMCEN位。 当操作禁用时，除启用寄存器的时钟外，所有RMC时钟停止以节省功耗。 当RMC启用然后禁用时，各寄存器的设置保持原来的状态。

17.3.3 RMCxREN (接收启用寄存器)

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

位	比特符号	型号	功能
31-1	−	R	读作 0。
0	RMCREN	R/W	接收 0: 禁用 1: 启用 控制RMC接收。 将该位设到 “1“，启用接收。

注:在设置 RMCxRCR1，RMCxRCR2，RMCxRCR3 后，启用<RMCREN> 位。

17.3.4 RMCxRBUF1 (接收数据缓冲寄存器 1)

	31	30	29	28	27	26	25	24
比特符号	RMCRBUF (接收31 ~ 24位的数据)							
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	RMCRBUF (接收23 ~ 16位的数据)							
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	RMCRBUF (接收15 ~ 8位的数据)							
复位后	0	0	0	0	0	0	0	0
	7	6	5	4.	3	2	1	0
比特符号	RMCRBUF (接收7 ~ 0位的数据)							
复位后	0	0	0	0	0	0	0	0

位	比特符号	型号	功能
31-0	RMCRBUF[31:0]	R	接收数据 (31 ~ 0位) 读取接收数据的4字节。 (31 ~ 0位)

17.3.5 RMCxRBUF2 (接收数据缓冲寄存器 2)

	31	30	29	28	27	26	25	24
比特符号	RMCRBUF (接收63 ~ 54位的数据)							
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	RMCRBUF (接收55 ~ 48位的数据)							
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	RMCRBUF (接收47 ~ 40位的数据)							
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	RMCRBUF (接收39 ~ 32位的数据)							
复位后	0	0	0	0	0	0	0	0

位	比特符号	型号	功能
31-0	RMCRBUF[63:32]	R	接收数据 (63 ~ 32位) 读取接收数据的4字节。 (63 ~ 32 位)

17.3.6 RMCxRBUF3 (接收数据缓冲寄存器 3)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	RMCRBUF (接收71 ~ 64位的数据)							
复位后	0	0	0	0	0	0	0	0

位	比特符号	型号	功能
31-8	-	R	读作 0。
7-0	RMCRBUF[71:64]	R	接收数据 (71 ~ 64)。 读取接收数据的1字节。 (71 ~ 64位)。

注: 接收的位以 MSB 优先的顺序存储在数据缓冲寄存器中，而最后接收到的位存储于 LSB 中 (0 位)。如果远程控制信号由 LSB 优先算法接收，则接受的数据以相反的顺序存储。

17.3.7 RMCxRCR1 (接收控制寄存器 1)

	31	30	29	28	27	26	25	24
比特符号	RMCLCMAX							
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	RMCLCMIN							
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	RMCLLMAX							
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	RMCLLMIN							
复位后	0	0	0	0	0	0	0	0

位	比特符号	型号	功能
31-24	RMCLCMAX[7:0]	R/W	规定先导检测的最大周期。 最大周期计算公式。<RMCLCMAX> × 4/fs [s].
23-16	RMCLCMIN[7:0]	R/W	规定先导检测的最小周期。 最小周期计算公式:<RMCLCMIN> × 4/fs [s].
15-8	RMCLLMAX[7:0]	R/W	规定先导的最大低宽度。 最大低宽度计算公式:<RMCLLMAX> × 4/fs [s]
7-0	RMCLLMIN[7:0]	R/W	规定先导最小低宽度。 最小低宽度算公式: <RMCLLMIN> × 4/fs [s] 若RMCxRCR2<RMCLD> = 1，即低脉冲幅宽值小于规定值，则定义为数据位。

注:在配置寄存器时，必须遵循以下规则。

先导	规则
低宽度 +高宽度	<RMCLCMAX[7:0]> > <RMCLCMIN[7:0]> <RMCLLMAX[7:0]> > <RMCLLMIN[7:0]> <RMCLCMIN[7:0]> > <RMCLLMAX[7:0]>
只有高宽度	<RMCLCMAX[7:0]> > <RMCLCMIN[7:0]> <RMCLLMAX[7:0]> = 0x00 <RMCLLMIN[7:0]> = 忽略
无先导	<RMCLCMAX[7:0]> = 0x00 <RMCLCMIN[7:0]> = 忽略 <RMCLLMAX[7:0]> = 忽略 <RMCLLMIN[[7:0]> = 忽略

17.3.8 RMCxRCR2 (接收控制寄存器 2)

	31	30	29	28	27	26	25	24
比特符号	RMCLIEN	RMCEDIEN	-	-	-	-	RMCLD	RMCPHM
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	RMCLL							
复位后	1	1	1	1	1	1	1	1
	7	6	5	4	3	2	1	0
比特符号	RMCDMAX							
复位后	1	1	1	1	1	1	1	1

位	比特符号	型号	功能
31	RMCLIEN	R/W	先导检测中断 0: 未生成 1: 生成
30	RMCEDIEN	R/W	远程控制输入下降沿中断 0: 未生成 1: 生成
29-26	−	R	读作 0。
25	RMCLD	R/W	不论有无先导，接收远程控制信号。 0: 禁用 1: 启用
24	RMCPHM	R/W	通过相位调节接收远程控制信号。 0: 通过相位调节不接收远程控制信号。 (通过周期调节接收数据。) 1: 通过固定频率脉冲调节接收远程控制信号。 通过脉冲调节接收固定频率远程控制信号，设置此位为 “1“。
23-16	−	R	读作0。
15-8	RMCLL[7:0]	R/W	触发接收完成和中断产生的超低宽度。 0000_0000 ~ 1111_1110: 以 <RMCLL> × 1/fs [s] 完成接收和产生中断。 1111_1111: 不用作触发器。
7-0	RMCDMAX[7:0]	R/W	触发接收完成和中断产生的最大数据位周期。 0000_0000 ~ 1111_1110:以<RMCDMAX> × 1/fs [s] 完成接收和产生中断。 1111_1111: 不用作触发器。

17.3.9 RMCxRCR3 (接收控制寄存器 3)

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

位	比特符号	型号	功能
31-15	-	R	读作 0。
14-8	RMCDATH[6:0]	R/W	确定相位法中的信号类型的大的阈值。 阈值计算公式: <RMCDATH> × 1/fs [s] 规定大阈值 (在1.5T和2T范围之内)以确定相位法中远程控制信号。如果测量的周期超过阈值，该位确定为 “10“。 如果不是，该位判定为 “01“。
7	-	R	读作 0。
6-0	RMCDATL[6:0]	R/W	判定0或者1的阈值，小的阈值以判定相位法中的信号类型。 阈值计算公式: <RMCDATL> × 1/fs [s] 规定两种阈值:阈值判定数据位是否为0或者1;较小的阈值 (在1T到1.5T范围内)以判定相位法中远程控制信号的类型。 对于数据位的判定，如果测量的周期超过阈值，则数据位判定为 “1“。 如果不是，数据位判定为 “0“。阈值计算公式: <RMCDATL> × 1/fs [s]. 对于远程控制信号类型的判定，如果测量的周期超过阈值，数据位判定为 “01“。如果不是，数据位判定为 “00“。

注:当接收控制寄存器 2 的<RMCPHM>位为 “0“，<RMCDATH[6:0]>未启用。当<RMCPHM>为 “1“时，数据位启用。

17.3.10 RMCxRCR4 (接收控制寄存器 4)

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

位	比特符号	型号	功能
31-8	−	R	读作 0。
7	RMCPO	R/W	远程控制输入信号 0: 未反向 1: 反向
6-4	−	R	读作 0。
3-0	RMCNC[3:0]	R/W	规定噪声消除时间。 0000: 无消除。 0001 ~ 1111:撤销 噪声消除时间计算公式。<RMCNC> × 1/fs [s]

17.3.11 RMCxRSTAT (接收状态寄存器)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	RMCRLIF	RMCLOIF	RMCDMAXIF	RMCEDIF	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	RMCRLDR	RMCRNUM						
复位后	0	0	0	0	0	0	0	0

位	比特符号	型号	功能
31-16	−	R	读作 0。
15	RMCRLIF	R	中断源标记 0: 无先导中断生成。 1: 先导中断生成。
14	RMCLOIF	R	中断源标记 0: 无低宽度检测中断生成。 1: 低宽度检测中断生成。
13	RMCDMAXIF	R	中断源标记 0: 无最大数据位周期中断生成。 1: 最大数据位周期中断生成。
12	RMCEDIF	R	中断源标记 0: 无下降沿中断生成。 1: 下降沿中断生成。
11-8	−	R	读作 0。
7	RMCRLDR	R	先导检测。 0: 禁用先导检测。 1: 启用先导检测。
6-0	RMCRNUM[6:0]	R	接收到的数据位个数。 000_0000:没有数据位 (只有先导)。 000_0001 ~ 100_1000: 1 ~ 72位 100_1001 ~ 111_1111: 73位以及更多 指定接收的位数目作为远程控制信号数据。在接收过程中位数目不能监控。在接收完后，位数目被存储。

注 1: 每产生一次中断，该寄存器更新一次。对寄存器进行写入操作无效。

注 2: RMC 保持接收 73 位或者更多位数据，直到通过检测最大数据位周期或者超低宽度结束接收。在这种情况下，在数据缓冲器中接收的数据无法保证。

17.3.12 RMCxEND1 (接收终端位数寄存器 1)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	-	RMCEND1						
复位后	0	0	0	0	0	0	0	0

位	比特符号	型号	功能
31-7	-	R	读作 0。
6-0	RMCEND1[6:0]	R/W	规定接收数据位的数目 000_0000： 无特别接收数据位 000_0001 ~ 100_1000 :规定接收数据位数 (1 ~ 72 位) 100_1001 ~ 111_1111 :不设置该值

17.3.13 RMCxEND2 (接收终端位数寄存器 2)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4.	3	2	1	0
比特符号	-	RMCEND2						
复位后	0	0	0	0	0	0	0	0

位	比特符号	型号	功能
31-7	-	R	读作 0。
6-0	RMCEND2[6:0]	R/W	规定接收数据位的数目 000_0000： 无特别接收数据位 000_0001 ~ 100_1000 :规定接收数据位数 (1 ~ 72 位) 100_1001 ~ 111_1111 :不设置该值

17.3.14 RMCxEND3 (接收终端位数寄存器 3)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4.	3	2	1	0
比特符号	-	RMCEND3						
复位后	0	0	0	0	0	0	0	0

位	比特符号	型号	功能
31-7	-	R	读作 0。
6-0	RMCEND3[6:0]	R/W	规定接收数据位数 000_0000：无特别接收数据位 000_0001 ~ 100_1000：规定接收数据位数 (1 ~ 72 位) 100_1001 ~ 111_1111：不设置该值

注 1:根据 RMCxEND1，RMCxEND2 和 RMCxEND3 的规定，可以设置三种类型的数据接收位。

注 2:RMCxEND1，RMCxEND2 及 RMCxEND3 须与最大数据位周期配合使用。

17.3.15 RMCxFSSEL (源时钟选择寄存器)

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

位	比特符号	型号	功能
31-1	-	R	读作 0。
0	RMCCLK	R/W	规定RMC功能的采样时钟 0：低频时钟 (32.768kHz) 1：定时器输出 (TBBOUT) 对于RMC功能的采样，可以实现设置低频时钟或者定时器输出。 通过TBBOUT的定时器输出的设置范围为30 ~ 34kHz。

注: 通过使用 RMCxFSSEL 改变采样时钟，首先通过使用 RMCxEN<RMCEN>禁止 RMC 的运行。然后,再次启用 RMC，在设置其他 RMC 寄存器之前设置 RMCxFSSEL。

17.4 操作说明

17.4.1 远程控制信号接收

17.4.1.1 取样时钟

远程控制信号通过低速 32.768kHZ 时钟(fs)采集。

17.4.1.2 基本操作

当检测到先导时，RMC 设置 RMCxRSTAT<RMCRLDR>位。

此时, 如果设置 RMCxRCR2<RMCLIEN> 位，先导检测会生成先导检测中断。 当发生先导顺序检测中断时，设置 RMCxRSTAT<RMCRLIF>位。

在先导检测之后，各数据位按顺序判定为 “0“或者 “1“ 。结果存储在 72 位的 RMCxRBUF1，RMCxRBUF2 和 RMCxRBUF3 寄存器中。通过设置 RMCxRCR2< RMCEDIEN> 位元，在每个数据位的下降沿产生一个远程控制信号输入下降沿中断。当当一个远程控制信号输入下降沿中断产生时，设置 RMCxRSTAT< RMCEDIF >位。

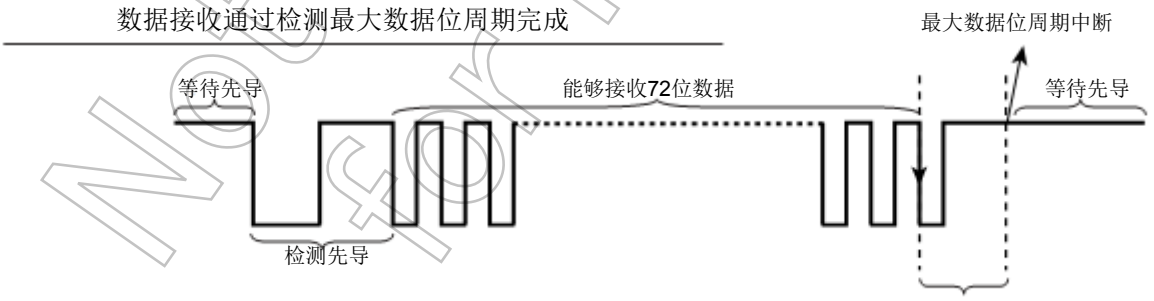
当检测到最大数据位周期并且低宽度与设定值相匹配时，数据接收结束，然后产生中断。如果寄存器 RMCxEND1，RMCxEND2 及 RMCEND3 的<RMCEND1>，<RMCEND2> 和 <RMCEND3>已经配置完成，并且仅在位数接收在检测到最大数据位周期之前的情况下，数据接收结束和产生中断。通过读取远程控制接收状态寄存器以检查 RMC 的状态。

如果在接收完成时检查 RMC 的状态，需读取远程控制接收状态寄存器。

在接收完成后，RMC 将等待下一个先导。

通过设置 RMC 以在没有先导的情况下接收信号，RMC 可以识别接收的信号为数据，并且在没有检查到先导的情况下，开始接收数据。

如果下一次数据接收在读取前一次接收的数据之前完成，前一次接收的数据会被下一次数据覆盖。



17.4.1.3 准备

在开始接收过程之前，通过使用远程控制信号远程控制寄存器 (RMCxRCR1, RMCxRCR2 和 RMCxRCR3, RMCxRCR4)，配置如何接收远程控制信号。

(1) 噪声消除时间设置

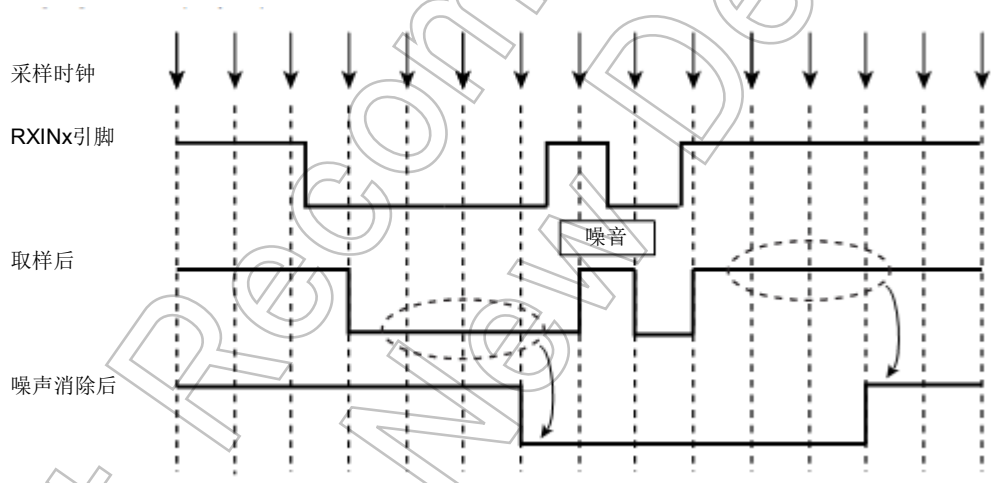
通过 RMCxRCR4 <RMCNC[3:0]>位配置噪声消除时间。

噪声消除用于远程控制信号，该信号是由采样时钟采集的。

监控每个采样时钟的上升沿时采集到的远程控制信号。如果监控到 “高“ , 在监控<RMCNC>中规定的 “低“周期之后，RMC 可以识别已被改变为 “低“。如果监控到 “低“，在监控<RMCNC>中规定的 “高“周期之后，RMC 可以识别已被改为 “高“的信号。

根据<RMCNC [3:0]> = “0011“ (3 个周期)的噪声消除设置，RMC 的运行原理如下所示。继噪声消除之后，在检测到三个 “低“周期后，信号从“高“ 改为 “低“。在检测到三个 “高“周期后，信号从“低“ 改为 “高“。

<RMCNC [3:0]> = 0011 (3周期)

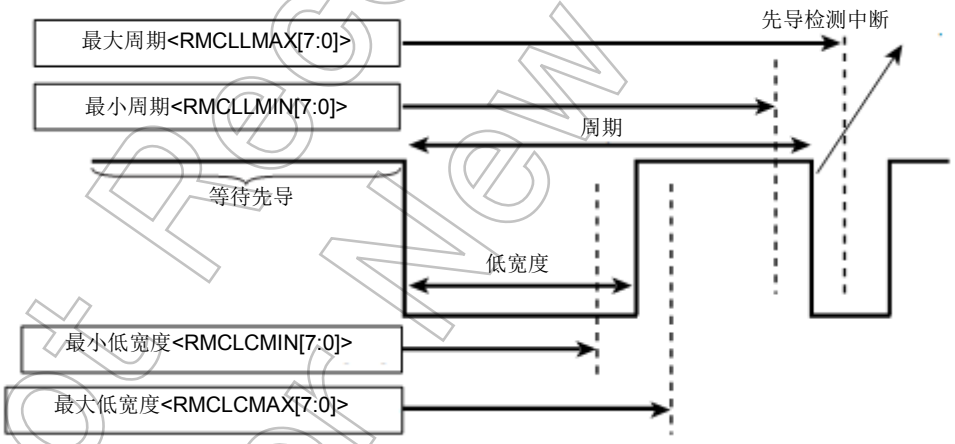


(2) 检测先导设置

设置先导周期及其低宽度为 RMCxRCR1 <RMCLLMIN[7:0]> <RMCLLMAX[7:0]> <RMCLCMIN[7:0]> <RMCLCMAX[7:0]>位。在配置以上设置时，遵循以下规则。

先导	规则
低宽度 + 高宽度	<RMCLCMAX[7:0]> > <RMCLCMIN[7:0]> <RMCLLMAX[7:0]> > <RMCLLMIN[7:0]> <RMCLCMIN[7:0]> > <RMCLLMAX[7:0]>
只有高宽度	<RMCLCMAX[7:0]> > <RMCLCMIN[7:0]> <RMCLLMAX[7:0]> = 0y00000000 <RMCLLMIN[7:0]> = 忽略
无先导	<RMCLCMAX[7:0]> = 0y00000000 <RMCLCMIN[7:0]> = 忽略 <RMCLLMAX[7:0]> = 忽略 <RMCLLMIN[7:0]> = 忽略

先导波形和 RMCxRCR1 寄存器设置如下所示。



当检测先导时，需要产生中断时，可配置 RMCxRCR2 <RMCLIEN>位。

没有先导的远程控制信号不能产生先导检测中断。

(3) 0/1 判定数据位设置

根据下降沿周期，判定数据位为 0 或者 1。

有两种判定方式:

1. 通过阈值判定。

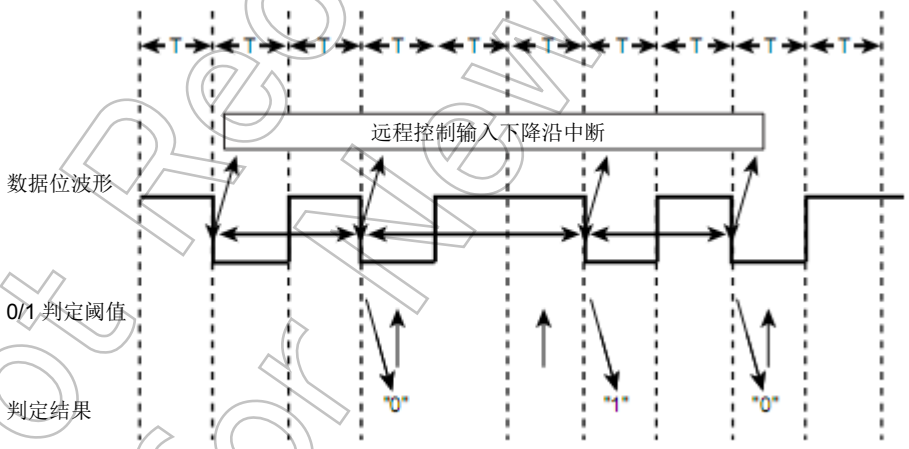
   配置阈值到 RMCxRCR3<RMCDATL[6:0]> 位，该位判定数据位为 “0“ 或者 “1“。如果测定值等于或者大于阈值，那么判定数据位为 “1“。如果测定值小于阈值，那么判定数据位为 “0“。

2. 通过下降沿中断输入来判定。

   通过将 “1“设置到 RMCxRCR2<RMCEDIEN>，可在各个数据位下降沿产生远程控制信号输入下降沿中断。同时和定时器应用此中断，可以启用软件来完成测定进程。

数据位的判定模式如下所示:。

通过<RMCDATL[6:0]>位将 0/1 判定的阈值设到 2.5T。



对于相位法中的远程信号的数据位判定，见 17.4.1.8 “相位法中接收远程控制信号“说明。

(4) 接收完成设置

为完成数据接收，需要设置检测最大数据位周期和超低幅。如果规定了多个因子，当检测到第一个因子时便完成接收。确保配置接收完成设置。

1. 通过最大数据位周期完成

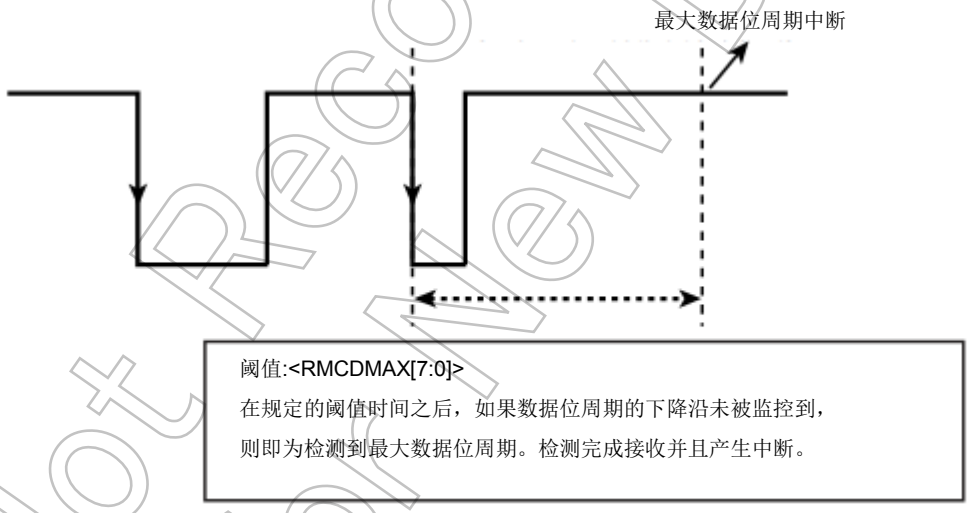
通过检测最大数据位周期完成接收，需要配置 RMCxRCR2 <RMCDMAX[7:0]>位。

在经过由<RMCDMAX[7:0]>所规定的的阈值时间之后，如果数据位周期的下降沿未被监控到，即检测到最大数据位周期。检测完成接收，并且产生中断。在产生中断输入之后，RMCxRSTAT< RMCDMAXIF >位设到 "1"。

要通过设置接收数据的数量来完成接收，即设置每个<RMCEND1> ，<RMCEND2> 和 <RMCEND3>的 RMCxEND 的 1 到 3 号寄存器。在这种情况下，当设置的接收位数与其中在 MAX 迸发的时候收到的接收数据数位的数目相符合时，设置每个 <RMCEND1> ，<RMCEND2>的 RMCxEND 1 ~ 3 号寄存器，这是通过数据位周期的 MAX 中断产生的。

如将 RMCxEND3 规定为 1，则可以设置 3 种接收数据位。

如果可以接收最大数据位，位数与在<RMCEND1>，<RMCEND2>，<RMCEND3>中规定的不相符合，则等待先导接收。

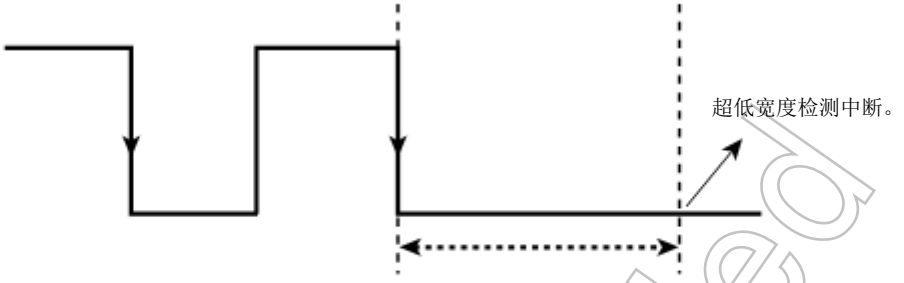


2. 通过检测低宽度完成

通过检测低宽度以完成接收，需要配置 RMCxRCR2 <RMCLL[7:0]>位。

在检测到数据位下降沿之后，如果信号显示为弱的时间超过规定值时，即检测到超低宽度。检测完成接收并且产生中断。

在中断输入产生后，RMCxRSTAT<RMCLOIF>位设到 "1"。



阈值:<RMCLL[7:0]>
当信号显示为弱的时间超过规定值时，
即检测到超低宽度。

17.4.1.4 启用接收

在配置 RMCxRCR1，RMCxRCR2，RMCxRCR3 和 RMCxRCR4 寄存器之后，通过启用 RMCxREN <RMCREN>位，RMC 即可接收。检测先导开始接收。

注意:在接收过程中改变 RMCxRCR1，RMCxRCR2，RMCxRCR3 和 RMCxRCR4 寄存器的配置会危害其正常操作。接收过程中改变配置时要特别小心。

17.4.1.5 停止接收

通过清除 RMCxREN <RMCREN>位到 "0" (接收禁用)，RMC 停止接收。

在接收过程中清除此位，立即停止接收并且废弃已经接收的数据。

17.4.1.6 接收等待前导字符中无前导字符的遥控信号

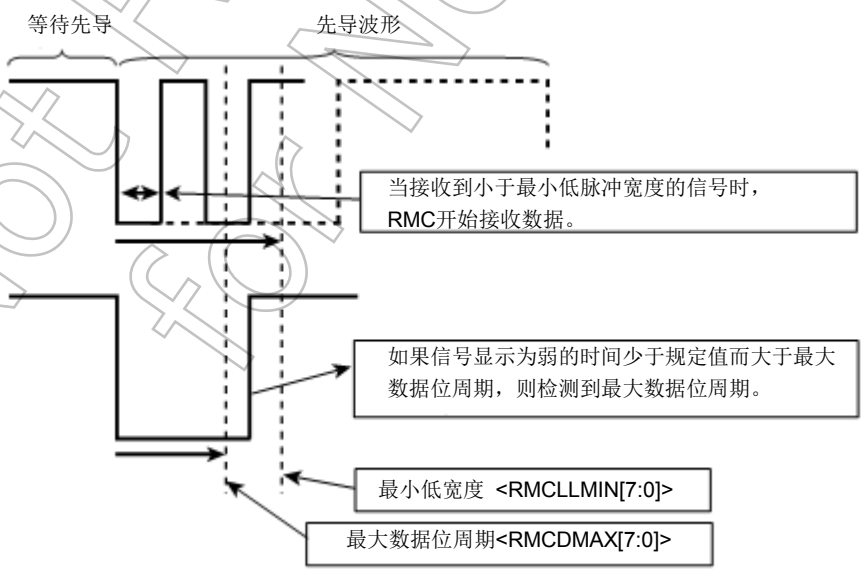
设置 RMCxRCR2 <RMCLD>，启用 RMC 以接收信号，不论有无先导。

通过设置 RMCxRCR2 <RMCLD>，如果 RMC 识别出低宽度小于先导检测最大低宽度 (由 RMCxRCR1 <RMCLLMAX [7:0]> 位规定)的信号，则 RMC 开始接收数据。RMC 保持接收数据状态直到接收到最终数据位。

如果启用 RMCxRCR2 <RMCLD>，不论信号有没有先导，即为使用与错误检测，接收完成和数据位测定相同的设置。

因此可接收的远程控制信号是有限的。

RMCxRCR2<RMCLD> = 1



17.4.1.7 仅带低宽度的前导字符

以仅有低宽度波形的先导为起始远程控制信号的情况，如下所示。

此信号始于仅有低宽度的先导数列和始于上升沿的一数据位周期。为启用该信号，必须通过设置 RMCxRCR4 <RMCPO>到 “1“使之反向之后，才能发送出去。

这是由于 RMC 的设置是从下降沿来检测数据位的。

为检测先导数列，以<RMCLLMAX[7ÅF0] >=0y0000 _ 0000,<RMCLCMAX[7:0]>><RMCLCMIN[7:0]>配置低脉冲宽度先导。

在这种情况下，<RMCLLMIN[7:0]>的值设置为“忽略“。

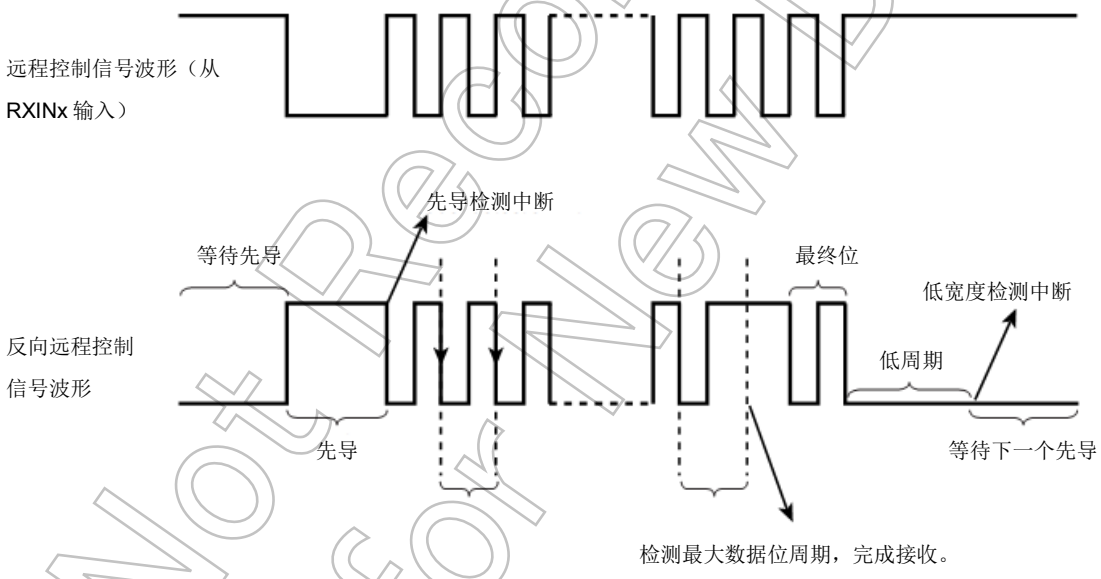
当检测是数据 “0“或数据 “1“时，用 RMCxRCR3 <RMCDATL[6:0]>配置 0/1 检测的阈值。

最大数据位周期配置为 RMCxRCR2 的<RMCDMAX[7:0]>。

当完成数据接收时，用 RMCxRCR2 的<RMCDMAX[7:0]> 来配置最大数据位周期。并用<RMCLLMIN[7:0]> 来配置低-脉冲检测。

在检测最大数据位周期，并用接收到的最后位判定低脉冲之后，数据接收完成。

RMC 产生一个中断，并等待下一个先导。



17.4.1.8 以相位法接收遥控信号

RMC 能够以相位法接收远程控制信号，而该信号的周期为固定的。相位法中的信号具有三种波形类型 (如下图所示)。

通过设置两个阈值，判定一个远程控制信号类型。RMC 转化信号为数据 “0“ 或者 “1“。接收完成后，接收的数据 “0“和 “1“存储在 RMCxRBUF1, RMCxRBUF 2 和 RMCxRBUF3 中。

通过设置 RMCxRCR2<RMCPHM> = “1“, RMC 启用以接收相位法的远程控制信号。各个阈值

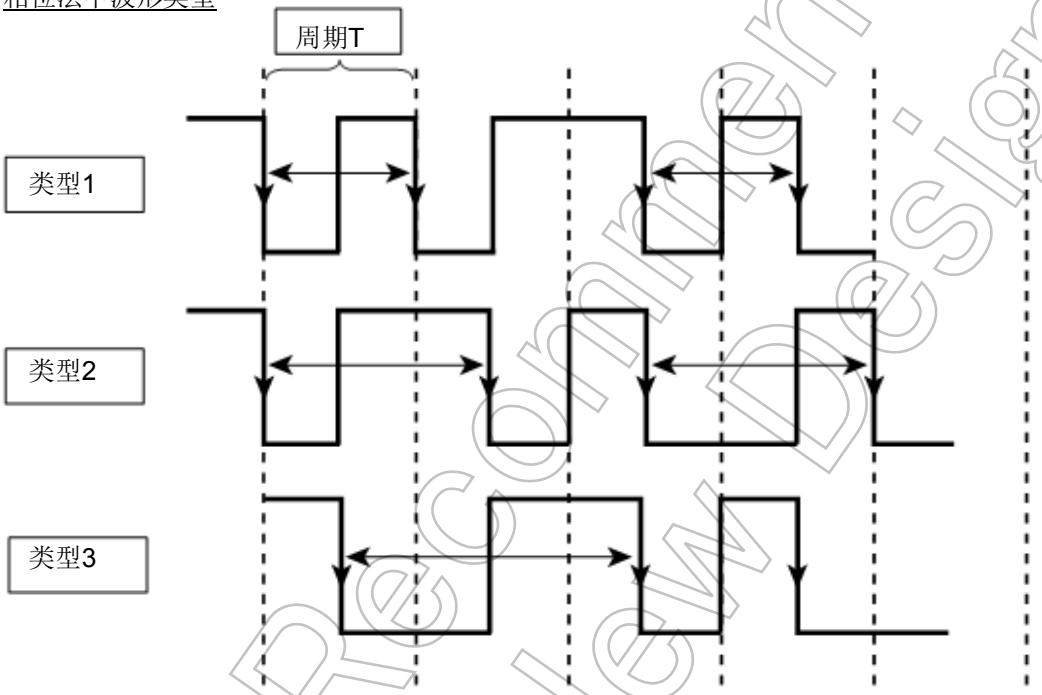
可以通过 RMCxRCR3 <RMCDATL[6:0]>位和<RMCDATH[6:0]>位设置。

通过两个阈值来区别三种波形类型。如果两个下降沿之间的周期为 “T“，则三种类型的周期为 1T，1.5T，2T。两个阈值的详情如下。

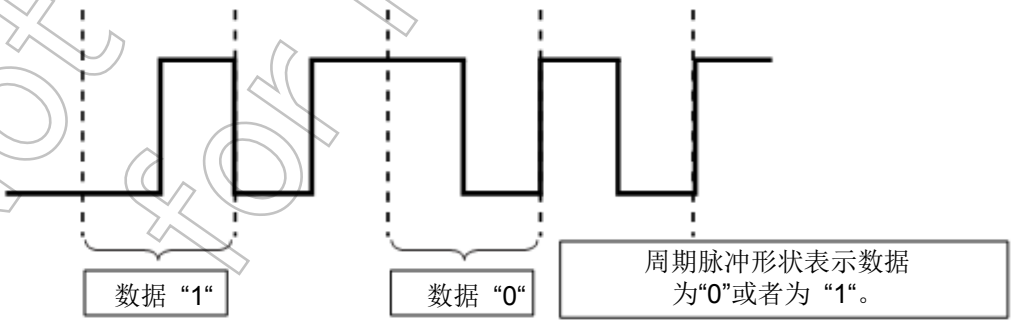
	由此判定	阈值	寄存器位数设置
阈值 1	类型 1& 类型 2	1T ~ 1.5T	RMCxRCR3<RMCDATL[6:0]>
阈值 2	类型 1& 类型 3	1.5T ~ 2T	RMCxRCR3<RMCDATH[6:0]>

为判定相位法中远程控制信号，需要三种数据波形和前期数据。另外，信号需要从 “11“开始。

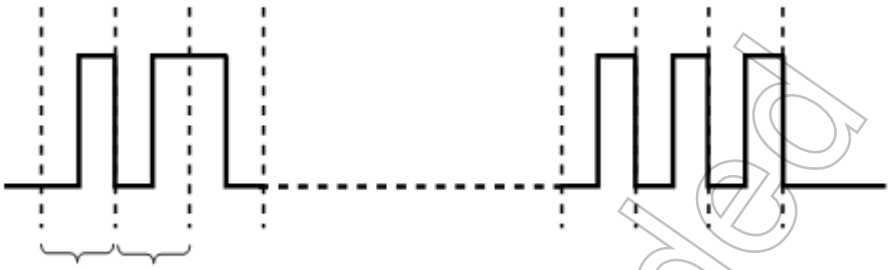
相位法中波形类型



相位法中远程控制信号数据



相位法远程控制信号



前两位数据应为“11“。

18. USB 主机控制器

USB 主机控制器 (USBHC)符合 USB 规范修订版 2.0 和开放 HCI 规范发布版 1.0A，并支持 12 Mbps (全速)的 USB 传输速度。USBHC 经由总线桥逻辑连接到 CPU。

USCHC 受到某些限制。有关详细内容，见 “18.8 有关使用 USB 主机控制器的限制“。

18.1 系统综述

USBHC 的主要特点如下:

1. 支持全-速 (12 Mbp)USB 装置。

   不支持低-速 (1.5 Mbps)USB 装置。

2. 支持控制，批量，中断和等时传输 (有一些限制。请参见 “18.8 有关使用 USB 主机控制器的限制“。

3. 为实现与 CPU 的连接，总线桥接逻辑包含两个 16-字节的 FIFO 缓冲器 (IN 和 OUT)，允许最多 16 字节的突发传输。

4. 支持总线桥逻辑中的 FIFO 缓冲器与片上 SRAM 之间的数据传输。可访问的 SRAM 为 RAM0 (0x2000_0000 ~ 0x2000_1FFF)和 RAM1 (0x2000_2000 ~ 0x2000_3FFF)。

注:**当 USBHC 正访问 RAM0 和 RAM1 时，如果 CPU 或 DMA 同时访问 RAM0 和 RAM1，则 CPU 和 DMA 优先访问。详细内容见“18.7.5 对 RAM0 和 RAM1 的竞争访问“。**

18.2 系统配置

USBHC 由以下三个模块组成:

1. USBHC 核心 (OHCI)
2. USB 收发器
3. CPU 总线桥逻辑

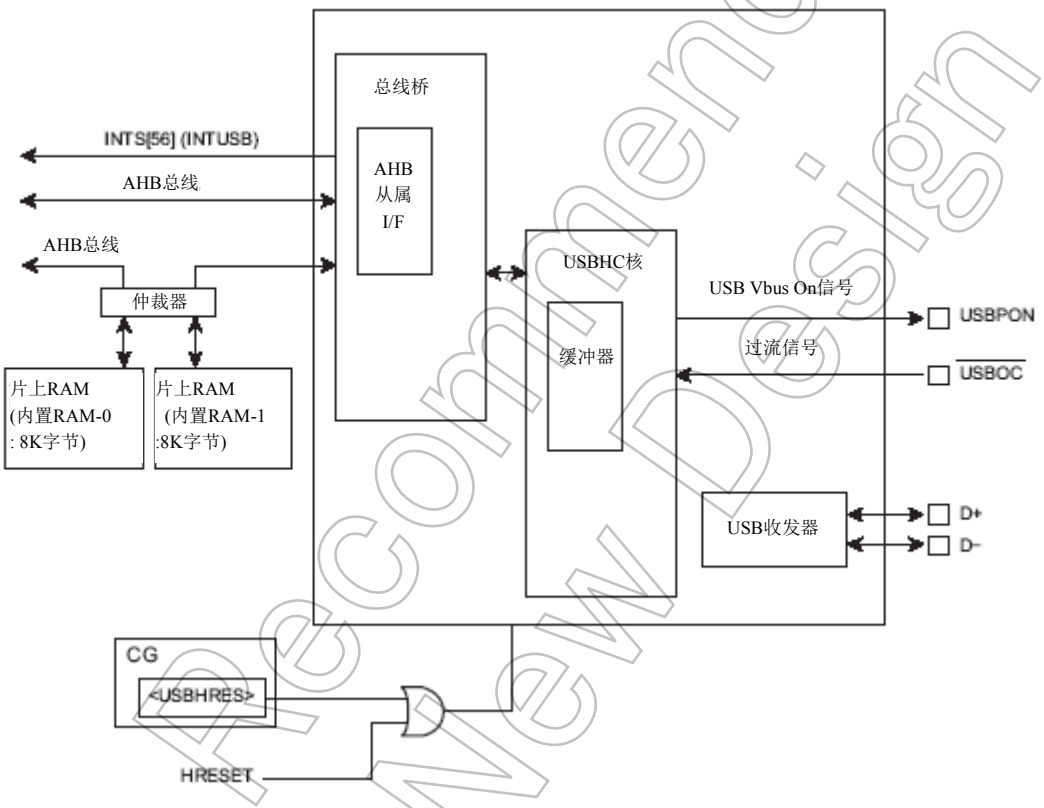


图 18-1 USB 主机控制器方块图

18.3 中断

USBHC 生成以下中断:

- 调度超载
- HcDoneHead 回写
- 帧起始
- 恢复检测
- 不可恢复的错误
- 帧编号溢出
- 根集线器状态改变
- 所有权改变

当一事件导致中断发生时，USBHC 设置 HcInterruptStatus 寄存器中的对应位。此时，如果 MasterInterruptEnable (MIE)位被启用，且 HcInterruptEnable 寄存器中的对应位被启用，则产生一 USB 中断 (INTUSB)。

USBHS 驱动软件可通过在 HcInterruptStatus 寄存器中写入 1 来将每一位置零。 (驱动软件不能设置这些位，且 USBHC 不能将这些位清除。)

18.4 复位

USBHC 是由硬件或软件复位初始化.

18.4.1 硬件复位

硬件复位是由外部复位引脚或内部事件生成 (WDT 复位，USBHC 复位或备份模式复位)。

- 初始化所有寄存器。
- 复位信号由一个外部下拉电阻输出到总线上。 (D+ = D- = 0)

  (USB 收发器处于 SUSPEND 状态。)
- USB 状态变化到 USBRESET 状态。
- 列表处理和 SOF 令牌生成被禁用。
- HcFmNumber 寄存器的 FrameNumber 字段不增加。

18.4.2 软件复位

当 “1“被写入 HcCommandStatus 寄存器中的 HostControllerReset 位时，发生一软件复位。

- 除以下情况之外，所有 OHCI 寄存器被初始化:

  -- HcControl 寄存器中的 RemoteWakeupConnected 和 InterruptRouting 位保持不变。

  -- HcBCR0 寄存器未被初始化。
- USBHC 输出复位信号到 USB 总线上 (D+ =D-=0)
- USB 状态变化到 USBSUSPEND 状态。

(HcControl 寄存器中的 FunctionalState 位被置为 0x03，以过渡到 USBSUSPEND 状态。)

18.5 总线电源控制

USBHC 有一用于外部 VBUS 电源 IC 的控制信号。该信号由 USBPON (PG6)引脚控制。

为将 PG6 用作 USBPON 引脚，G 端口控制寄存器 (PGFR2)必须正确设置。然后，将 OCI 寄存器中 HcRhStatus 寄存器的 LPSC 位置为 “1“，USBPON 引脚输出 “高“电平。

$\overline{\text{USBOC}}$ (PG7)引脚用于检测过流情况。当在该引脚上检测到低电平时，USBHC 将 HcRhStatus 寄存器中的 OCI 位置为 “1“。 (要将 PG7 用作 $\overline{\text{USBOC}}$ 引脚，G 端口控制寄存器 (PGFR2)必须正确设置。)

18.6 寄存器

USBHC 包含一组被映射到内存空间的符合开放 HCI 规范的控制寄存器。用于与 CPU 连接的总线桥逻辑也包括控制寄存器。

可通过一个 32-位总线直接从 CPU 访问这些寄存器。

基址 = 0x4000_3000

寄存器名称		地址 (基+)
Hc修订寄存器	HcRevision	0x0000
Hc控制寄存器	HcControl	0x0004
Hc命令状态寄存器	HcCommandStatus	0x0008
Hc中断状态寄存器	HcInterruptStatus	0x000C
Hc中断启用寄存器	HcInterruptEnable	0x0010
Hc中断禁用寄存器	HcInterruptDisable	0x0014
Hc主机控制器通信区域寄存器	HcHCCA	0x0018
Hc周期当前端点描述符寄存器	HcPeriodCurrentED	0x001C
Hc控制头端点描述符寄存器	HcControlHeadED	0x0020
Hc控制当前端点描述符寄存器	HcControlCurrentED	0x0024
Hc批量头端点描述符寄存器	HcBulkHeadED	0x0028
Hc批量当前端点描述符寄存器	HcBulkCurrentED	0x002C
Hc完成标题寄存器	HcDoneHead	0x0030
Hc帧间隔寄存器	HcFmInterval	0x0034
Hc帧保持寄存器	HcFmRemaining	0x0038
Hc帧编号寄存器	HcFmNumber	0x003C
Hc周期起始寄存器	HcPeriodStart	0x0040
Hc低速阈值寄存器	HcLSThreshold	0x0044
Hc根集线器描述符A寄存器	HcRhDescriptorA	0x0048
Hc根集线器描述符B寄存器	HcRhDescripterB	0x004C
Hc根集线器状态寄存器	HcRhStatus	0x0050
Hc根集线器端口状态寄存器	HcRhPortStatus1	0x0054
Hc BCR0寄存器	HcBCR0	0x0080

注 1: **开放 HCI 规范发布版 1.0a 规定 HcFmRemaining 寄存器中的 FrameRemaining (FR)和 FrameRemainingToggle (FRT)位，以及 HcFmNumber 寄存器中的 FramNumber (FN)位对主机控制驱动程序 (HDC)是只读的。然而， USBHC 允许 HCD 以调试的目的对这些寄存器进行写访问。如果 HCD 写入这些寄存器，会发生未定义的结果。这些位不可以由 HCD 写入。**

注 2: **可参见 "18.6.1 HcRevision 寄存器"到"18.6.23 HcBCR0 寄存器"等节。有关基于 OHCI 的讲解，请参见规范"开放 HCI 规范发布版 1.0a "。**

18.6.1 HcRevision寄存器

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

位	比特符号	类型 (HDC)	类型 (HC)	功能
31-8	–	–	–	保留
7-0	REV[7:0]	R:	R:	文件名:修订 该只读字段包含由HC实现的对HCI规范版本的BCD码表示法。 例如，0x11的值对应于版本1.1。所有符合该规范的HC实现都有一个0X10的值。

18.6.2 HcControl寄存器

HcControl 寄存器定义了主机控制器的操作模式。除 HostControllerFunctionalState 和 RemoteWakeup-Connected 之外，该寄存器中的大部分字段只能由主机控制器驱动程序修改。

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后								
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后								
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	RWE	RWC	IR
复位后						0	0	0
	7	6	5	4	3	2	1	0
比特符号	HCFS		BLE	CLE	IE	PLE	CBSR	
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型 (HCD)	类型 (HC)	功能
31-11	–	–	–	保留
10	RWE	R/W	R	文件名:Remote Wake-up Enable 检测到上行恢复信号时，HCD用该位来实现启用或禁用远程唤醒功能。当该位以及HcInterruptStatus中的ResumeDetected位被设置，一远程唤醒信号被发送到主机系统。将该位置对不会影响硬件中断的产生。
9	RWC	R/W	R/W	文件名: Remote Wake-up Connected 该位表明HC是否支持远程唤醒信号。如果该系统支持且使用远程唤醒，则在POST期间，由系统固件负责设置该位置。HC在硬件复位时将该位置零，但在软件复位时不更改该位。
8	IR	R/W	R	文件名Interrupt Routing 该位决定由HcInterruptStatus中注册的事件所产生中断的路由选择。如果将该位置零，所有中断将路由到正常主机总线中断机制。如果将设置该位置，中断将路由到系统管理中断。HCD在硬件复位时将该位置零，但在软件复位时不更改该位。HCD使用该位作为标记，以指示HC的所有权。
7-6	HCFS[1:0]	R/W	R/W	文件名:Host Controller Functional State for USB 00 : USBRESET 01 : USBRESUME 10 : USBOPERATIONAL 11 : USBSUSPEND 从另一个状态到USBOPERATIONAL的过渡导致1毫秒后开始SOF生成。HCD可通过读HcInterruptStatus的StartofFrame字段确定HC是否已开始发送SOF。 此字段仅在处于USBSuspend状态时才可被HC修改。从下游端口检测到恢复信令后，HC可以由USBSUSPEND状态变为USBRESUME状态。 HC在一软件复位后进入USBSUSPEND，而在一硬件复位后进入USBRESET。后者也复位根集线器，且断言随后发送到下游端口的复位信令。
5	BLE	R/W	R	文件名:Bulk List Enable 设置该位置以启用下一帧中的批量列表处理。如果该位被HCD清除，下一个SOF后不会发生批量列表处理。每当HC决定处理该列表时都会检查该位。禁用时，HCD修改该列表。如果HcBulkCurrentED正指向一个待删除的ED，HCD必须在重启列表处理之前通过更新HcBulkCurrentED前移指针。

<table>
<tr><th>位</th><th>比特符号</th><th>类型
(HCD)</th><th>类型
(HC)</th><th>功能</th></tr>
<tr><td>4</td><td>CLE</td><td>R/W</td><td>R</td><td>文件名:控制列表启用
将该位置被设置以启用下一帧中的控制列表处理。如果该位被HCD清除，下一个SOF后不会发生控制列表处理。每当HC决定处理该列表时都必须检查该位。禁用时，HCD修改该列表。如果HcControlCurrentED正指向一个待删除的ED，HCD必须在重启列表处理之前通过更新HcControlCurrentED前移指针。</td></tr>
<tr><td>3</td><td>IE</td><td>R/W</td><td>R</td><td>文件名:等时启用
该位由HCD用来启用/禁用等时ED处理。在处理一帧中的周期列表时，每当HC发现一个等时ED (F=1)都会检查该位的状态。如被设置 (启用)，HC继续处理ED。如被清除 (禁用)，HC停止对周期列表 (当前只包含等时ED)的处理，并开始处理批量/控制列表。将该位置1的操作保证在下一帧 (不是当前帧)中生效。
* 该产品对等时传输存在一些限制。</td></tr>
<tr><td>2</td><td>PLE</td><td>R/W</td><td>R</td><td>文件名:周期列表启用
设置该位置以启用下一帧中的周期列表处理。如果该位被HCD清除，下一个SOF后不会发生周期列表处理。HC开始处理该列表前必须检查该位。</td></tr>
<tr><td>1-0</td><td>CBSR[1:0]</td><td>R/W</td><td>R</td><td>文件名:控制批量伺服比
该文件指定控制和批量ED的服务比。在处理任意非周期性列表前，HC必须将指定比与关于多少非空控制ED已被处理的内部计数进行比较，以决定是否继续服务于另一个控制ED或切换到批量ED。当穿越帧边界时，内部计数将被保留。在复位的情况下，HCD负责恢复该值。
<table>
<tr><th><CBSR[1:0]></th><th>控制ED号超过被服务批量ED号</th></tr>
<tr><td>00</td><td>1 : 1</td></tr>
<tr><td>01</td><td>2 : 1</td></tr>
<tr><td>10</td><td>3 : 1</td></tr>
<tr><td>11</td><td>4 : 1</td></tr>
</table>
</td></tr>
</table>

18.6.3 HcCommandStatus寄存器

HcCommandStatus 寄存器被主机控制器用于接收由主机控制器驱动程序发出的命令，并反映主机控制器的当前状态。对于主机控制器驱动程序，它显示为“写到设置“寄存器。主机控制器必须保证:寄存器中写为 “1“的位被置位，而寄存器中写为 “0“的位保持不变。主机控制器驱动程序可能会向主机控制器发出多个不同的命令，而不用担心损坏之前发出的命令。主机控制器驱动程序可对所有位进行正常的读访问。

SchedulingOverrunCount 字段表明主机控制器已检测到调度超载错误所使用的帧编号。若 EOF 之前周期列表未完成，该事件会发生。当检测到调度超载错误时，主机控制器增加计数器的值，并将 HcInterruptStatus 寄存器中的 SchedulingOverrun 字段设置。

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后								
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	SOC	
复位后							0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后								
	7	6	5	4	3	2	1	0
比特符号	-	-	-	-	OCR	BLF	CLF	HCR
复位后					0	0	0	0

位	比特符号	类型 (HCD)	类型 (HC)	功能
31-8	–	–	–	保留
17-16	SOC[1:0]	R	R/W	文件名: SchedulingOverrunCount 这些位在每次scheduling overrun错误时会增加.它被初始化到 00，且在11时回绕。当检测到调度超载时，即使HcInterruptStatus中的调度超载已被设置，其仍将增加。这被HCD用来监测任何持续的调度问题。
15-4	–	–	–	保留
3	OCR	R/W	R/W	文件名: Ownership Change Request 该位由OS HCD设置，以请求改变HC的控制权。将该位被置1时，HC将HcInterruptStatus中的OwnershipChange字段设置。发生转变后，该位被清除并保持，直到从OS HCD收到下一个请求。
2	BLF	R/W	R/W	文件名:Bulk List Field 该位用来指示Bulk List上是否有任何TD。每当对Bulk List中的ED添加一个TD时，该位被HCD设置。 当HC开始处理Bulk List头时，它会检查BF。只要BulkListFilled为0，HC将不会开始处理Bulk List。如果BulkListFilled为1，HC将开始处理该Bulk List，且将BF置为0。如果HC在列表上发现一个TD，则HC将BulkListFilled置为1，从而引起Bulk List处理延续。如果批量列表上没有TD，且如果HCD未将BulkListFilled设置，则当HC处理完Bulk List时BulkListFilled将依然为0，且Bulk List处理将停止。
1	CLF	R/W	R/W	文件名:Control List Field 该位用来指示Control List上是否有任何TD。每当它为Control List中的ED添加一个TD时，它被HCD设置。 当HC开始处理Control List头时，它会检查CLF 。只要ControlListFilled为0，HC将不会开始处理Control List。如果CF为1，HC将开始处理Control List，且将ControlListFilled置为0。如果HC发现列表上有TD，则HC将ControlListFilled置为1，以使Control List处理延续。如果Control List上没有TD，且如果HCD未将ControlListFilled设置，则当HC处理完Control List时ControlListFilled将依然为0，且Control List处理将停止。
0	HCR	R/W	R/W	文件名:Host Controller Reset 该位被HCD设置以启动HC的软件复位。不管HC的功能状态是什么，它都会变化到USBSuspend状态，除非另有规定，其中的大部分操作寄存器被复位，例如HcControl的InterruptRouting字段，且不允许主机总线访问。复位操作完成后，该位由HC清除。复位操作必须在10 s内完成。当被设置时，该位不应引起根集线器复位，且应无后续复位信令被要求到下游端口。

18.6.4 HcInterruptStatus寄存器

该寄存器提供引起硬件中断的各种事件的状态。当发生一个事件时，主机控制器将该寄存器的相应位设置。当该位被设置时，如果该中断在 HcInterruptEnable 寄存器中被启用且 MasterInterruptEnable 位被设置，则产生一个硬件中断。主机控制器驱动程序可通过在待清除位的位置写入 “1“来将此寄存器中的特定位清除。主机控制器驱动程序可能不将这些位中的任一位设置。主机控制器将永不对该位清除。

	31	30	29	28	27	26	25	24
比特符号	-	OC	-	-	-	-	-	-
复位后		0						
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后								
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后								
	7	6	5	4	3	2	1	0
比特符号	-	RHSC	FNO	UE	RD	SF	WDH	SO
复位后		0	0	0	0	0	0	0

位	比特符号	类型 (HCD)	类型 (HC)	功能
31	–	–	–	保留
30	OC	R/W	R/W	文件名: Ownership Change 当HCD对HcCommandStatus中的ChangeRequest所有权字段置位时，该位由HC设置。该事件在去屏蔽时总会立即产生一个系统管理中断 (SMI)。 当SMI引脚未被执行时，该位被绑定为0。
29-7	–	–	–	保留
6	RHSC	R/W	R/W	文件名: Root Hub Status Change 当HcRhStatus的内容或任一HcRhPortStatus [Numberof - DownstreamPort]的内容已经改变时，该位被设置。
5	FNO	R/W	R/W	文件名: Frame Number Overflow 当HcFmNumber的MSB (位15)值发生改变，从0 ~ 1或从1 ~ 0，且HccaFrameNumber已被更新后，该位被设置。
4	UE	R/W	R/W	文件名: Unrecoverable Error 当HC检测到与USB无关的系统错误时，该位被设置。在系统错误得到修正之前，HC不应继续进行任何处理或信令。HC复位后，HCD将该位清除。
3	RD	R/W	R/W	文件名: Resume Detected 当HC检测到USB上一装置正在要求恢复信令时，该位被设置。从无恢复信令到恢复信令的过渡导致该位被设置。当HCD设置USBRESUME状态时，不对该位设置。
2	SF	R/W	R/W	文件名:Startof Frame 该位在每一帧起始处及HccaFrameNumber更新后被设置。同时，HC也生成SOF令牌。
1	WDH	R/W	R/W	文件名: Write Back Done Head 在HC将HcDoneHead写入HccaDoneHead后，该位被立即设置。直到该位被清除，HccaDoneHead的进一步更新才会发生。HCD保存完HccaDoneHead的内容后，应只对该位清除。
0	SO	R/W	R/W	文件名: Scheduling Overrun 当USB对当前帧的Scheduling Overrun且HccaFrameNumber已更新时，该位被设置。Scheduling Overrun也将导致HcCommandStatus的SchedulingOverrunCount增加。

18.6.5 HcInterruptEnable寄存器

HcInterruptEnable 寄存器中的各启用位对应于 HcInterruptStatus 寄存器中的相关中断位。HcInterruptEnable 寄存器用于控制生成硬件中断的事件。当 HcInterruptStatus 寄存器中的某位被设置，HcInterruptEnable 寄存器中相应的位被设置，且 MasterInterruptEnable 位被设置，则主机总线上请求一个硬件中断。

向此寄存器中的某位写入 “1“ 设置相应的位，而对此寄存器中的某位写入 “0“将不该变相应的位。读状态下，该寄存器的当前值被返回。

	31	30	29	28	27	26	25	24
比特符号	MIE	OC	-	-	-	-	-	-
复位后	0	0						
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后								
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后								
	7	6	5	4	3	2	1	0
比特符号	-	RHSC	FNO	UE	RD	SF	WDH	SO
复位后		0	0	0	0	0	0	0

位	比特符号	类型 (HCD)	类型 (HC)	功能
31	MIE	R/W	R	文件名: Master Interrupt Enable HC忽略该字段中写入的 ‘0’。由于该寄存器中其他位指定的事件，写入该字段的 ‘1‘启用中断生成。HCD将此用作主中断启用。
30	OC	R/W	R	文件名: Ownership Change 0: 忽略 1: 启用由于Ownership Change中断生成。
29-7	–	–	–	保留
6	RHSC	R/W	R	文件名: Root Hub Status Change 0: 忽略 1: 启用由于Root Hub Status Change中断生成。
5	FNO	R/W	R	文件名: Frame Number Overflow 0: 忽略 1: 启用由于Frame Number Overflow中断生成。
4	UE	R/W	R	文件名: Unrecoverable Error 0: 忽略 1: 启用由于Unrecoverable Error中断生成。
3	RD	R/W	R	文件名: Resume Detected 0: 忽略 1: 启用由于Resume Detected中断生成。
2	SF	R/W	R	文件名:Startof Frame 0: 忽略 1:启用由于Startof Frame中断生成。
1	WDH	R/W	R	文件名: Write Back Done Head 0: 忽略 1: 启用由于HcDoneHeadWriteback中断生成。
0	SO	R/W	R	文件名: Scheduling Overrun 0: 忽略 1: 启用由于Scheduling Overrun中断生成。

18.6.6 HcInterruptDisable寄存器

HcInterruptDisable 寄存器中各禁用位对应于 HcInterruptStatus 寄存器中的相应中断位。HcInterruptDisable 寄存器与 HcInterruptEnable 寄存器耦合。因此向该寄存器中的某位写入 "1"，将清零 HcInterruptEnable 寄存器中的相应位，而向该寄存器中的某位写入 "0"，将不改变 HcInterruptEnable 寄存器中的相应位。在读取状态下，HcInterruptEnble 寄存器中的当前值被返回。

	31	30	29	28	27	26	25	24
比特符号	MIE	OC	-	-	-	-	-	-
复位后	0	0						
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后								
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后								
	7	6	5	4	3	2	1	0
比特符号	-	RHSC	FNO	UE	RD	SF	WDH	SO
复位后		0	0	0	0	0	0	0

位	比特符号	类型 (HCD)	类型 (HC)	功能
31	MIE	R/W	R	文件名:Master Interrupt Enable HC忽略该字段中写入的 '0'。由于该寄存器中其他位指定的事件，写入该字段的 '1' 禁用中断生成。该字段在硬件或软件复位之后被设置。
30	OC	R/W	R	文件名:Ownership Change 0: 忽略 1: 禁用由于Ownership Change中断生成。
29-7	–	–	–	保留
6	RHSC	R/W	R	文件名:Root Hub Status Change 0: 忽略 1: 禁用由于Root Hub Status Change中断生成。
5	FNO	R/W	R	文件名:Frame Number Overflow 0: 忽略 1: 禁用由于Frame Number Overflow中断生成。
4	UE	R/W	R	文件名: Unrecoverable Error 0: 忽略 1: 禁用由于Unrecoverable Error中断生成。
3	RD	R/W	R	文件名:Resume Detected 0:忽略 1: 禁用由于Resume Detected中断生成。
2	SF	R/W	R	文件名:Startof Frame 0:忽略 1: 禁用由于Start of Frame中断生成。
1	WDH	R/W	R	文件名:Write Back Done Head 0: 忽略 1: 禁用由于HcDoneHeadWriteback中断生成。
0	SO	R/W	R	文件名:Scheduling Overrun 0: 忽略 1: 禁用由于Scheduling Overrun中断生成。

18.6.7 HcHCCA寄存器

HcHCCA 寄存器含有 Host Controller Communication Area 的物理地址。主机控制器驱动程序通过将全 1s 写入 HcHCCA 及读取 HcHCCA 的内容来确定队列限制排列是通过检查低阶位中零的数目来评估的。最小的排列是 256 字节，因此在读取时，位 0～7 必须总是返回 "0"。此区域用于保存由主机控制器和主机控制器驱动程序两者访问的控制结构和中断情况表。

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

位	比特符号	类型 (HCD)	类型 (HC)	功能
31-8	HCCA[23:0]	R/W	R	文件名: Host Controller Communication Area 这是主机控制器通讯区域的基址。
7-0	–	–	–	保留

18.6.8 HcPeriodCurrentED寄存器

HcPeriodCurrentED 寄存器包含当前等时或中断端点描述符的物理地址。

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

位	比特符号	类型 (HCD)	类型 (HC)	功能
31-4	PCED[27:0]	R	R/W	文件名: PeriodCurrentED 这被HC用来指向将在当前帧中被处理的周期列表之一的头。在周期ED被处理之后，由HC将该寄存器的内容更新。
3-0	–	–	–	保留

18.6.9 HcControlHeadED寄存器

HcControlHeadED 寄存器包含控制列表首端点描述符的物理地址。

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

位	比特符号	类型 (HCD)	类型 (HC)	功能
31-4	CHED[27:0]	R/W	R	文件名: Control Head ED HC从HcControlHeadED指针开始遍历控制列表。在HC初始化期间，从HCCA加载该内容。
3-0	–	–	–	保留

18.6.10 HcControlCurrentED寄存器

该 HcControlCurrentED 寄存器包含控制列表当前端点描述符的物理地址。

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

位	比特符号	类型 (HCD)	类型 (HC)	功能
31-4	CCED[27:0]	R/W	R/W	文件名: Control Current ED 伺服完当前ED后，指针前移到下一个ED。HC将在上一帧停止处继续处理列表。当它到达控制列表的末尾，HC检查HcCommandStatus的ControlListFilled字段。如被设置，HC将HcControlHeadED的内容复制到HcControlCurrentED并将该位清除。如未设置，HC不做任何处理。只有当HcControl的ControlListEnable字段被清零时，才允许HCD修改该寄存器。当被设置时， HCD只读取该寄存器的瞬时值。最初，其被置为零以指示控制列表的末尾。
3-0	–	–	–	保留

18.6.11 HcBulkHeadED寄存器

HcBulkHeadED 寄存器包含批量列表首端点描述符的物理地址。

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

位	比特符号	类型 (HCD)	类型 (HC)	功能
31-4	BHED[27:0]	R/W	R	文件名:Bulk Head ED HC从HcBulkHeadED指针开始遍历批量列表。在HC初始化期间，从HCCA加载该内容。
3-0	–	–	–	保留

18.6.12 HcBulkCurrentED寄存器

HcBulkCurrentED 寄存器包含批量列表当前端点的物理地址。由于批量列表将以轮询的方式被伺服，端点将以其插入到列表中的顺序排序。

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

位	比特符号	类型 (HCD)	类型 (HC)	功能
31-4	BCED[27:0]	R/W	R/W	文件名: Bulk Current ED 服务完当前ED后，HC前移到下一个ED。HC在上一帧停止处继续处理该列表。当它到达批量列表的末尾时，HC检查HcControl的ControlListFilled字段。如被设置，HC将HcBulkHeadED的内容复制到HcBulkCurrentED并将该位清除。如未设置，HC不做任何处理。只有当HcControl的BulkListEnable被清零时，才允许HCD修改该寄存器。当被设置时，HCD只读取该寄存器的瞬时值。其在最初被置为零，以指示批量列表的末尾。
3-0	–	–	–	保留

18.6.13 HcDoneHead寄存器

HcDoneHead 寄存器包含被添加到完成队列中最后完成的传输描述符的物理地址。正常运行时，主机控制器驱动程序不需读取该寄存器，因为该寄存器的内容被周期性地写入 HCCA。

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

位	比特符号	类型 (HCD)	类型 (HC)	功能
31-4	DH[27:0]	R	R/W	文件名: Done Head 当完成一TD后，HC将HcDoneHead的内容写到该TD的下一TD字段。HC然后用该TD的地址重写HcDoneHead的内容。 每当HC将该寄存器的内容写入HCCA时，其被置为零。它还将HcInterruptStatus的WritebackDoneHead设置。
3-0	–	–	–	保留

18.6.14 HcFmInterval寄存器

HcFmInterval 寄存器包含一个指示一帧中位时间间隔 (也就是，连续两个 SOF 之间)的 14-位值，以及一个指示主机控制器在不造成调度超载的情况下可以发送或接收的全速最大数据包容量的 15-0 位值。通过在每个 SOF 上写入一大于当前值的新值, 主机控制器驱动程序可以实现在 FrameInterval 上进行微调。这为主机控制器提供了必要的可编程性，以实现与外部时钟资源同步，并调整任何未知的局部时钟偏移。

	31	30	29	28	27	26	25	24
比特符号	FIT	FSMPS						
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	FSMPS							
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	FI					
复位后	0	0	1	0	1	1	1	0
	7	6	5	4	3	2	1	0
比特符号	FI							
复位后	1	1	0	1	1	1	1	1

位	比特符号	类型 (HCD)	类型 (HC)	功能
31	FIT	R/W	R	文件名: Frame Interval Toggle HCD每当在FrameInterval中加载一个新值时将该位切换。
30-16	FSMPS[14:0]	R/W	R	文件名:FS Largest Data Packet 该字段指定了这样一值，其在每帧开始处被加载到Largest Data Packet Counter。该计数器的值表示在不造成调度溢出的情况下，HC在任意给定时间可以发送或接收数据位的最大数量。该字段值由HCD计算得出。
15-14	-	-	-	保留
13-0	FI[13:0]	R/W	R	文件名: Frame Interval 其指定了两个连续SOF在位时间上的时间间隔。标称值被设置为11,999。 在复位HC之前，HCD 应存储该字段的当前值。通过设置HcCommandStatus的HostController-Reset字段，将使HC把该字段复位为其标称值。 HCD可选择在完成复位序列后恢复已保存的值。

18.6.15 HcFmRemaining寄存器

HcFmRemaining 寄存器是一个显示当前帧中位保持时间的 14-位递减计数器。

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

位	比特符号	类型 (HCD)	类型 (HC)	功能
31	FRT	R	R/W	文件名: Frame Remaining Toggle 每当FrameRemaining变为0时，从HcFmInterval的FrameIntervalToggle字段加载该位。该位被HCD用于FrameInterval和FrameRemaining之间的同步。
30-14	–	–	–	保留
13-0	FR[13:0]	R	R/W	文件名: FrameRemaining 该计数器在每一位时间递减。当变为0时，计数器在下一位时间边界通过加载HcFmInterval中指定的FrameInterval值将其复位。当进入USBOPERATIONAL状态时，HC用HcFmInterval中的FrameInterval重新加载内容，并使用下一个SOF更新后的值。

18.6.16 HcFmNumber寄存器

HcFmNumber 寄存器是一个 16-位计数器。它提供一个在主机控制器和主机控制器驱动程序中发生事件之间的定时参考。主机控制器驱动程序可以使用这个寄存器指定的 16-位值，并产生一个 32-位的帧编号，而不需要频繁访问该寄存器。

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

位	比特符号	类型 (HCD)	类型 (HC)	功能
31-16	–	–	–	保留
15-0	FN[15:0]	R	R/W	文件名:Frame Number 当HcFmRemaining重新加载时，增加。在0xffff之后，变回到0x0000。当进入USBOPERATIONAL状态时，帧编号将自动增加。当HC在每个帧边界将帧数增大且在读取该帧的第一个ED前发送一个SOF之后，内容被写入HCCA。在写入HCCA之后，HC将HcInterruptStatus中的StartofFrame设置。

18.6.17 HcPeriodicStart寄存器

HcPeriodicStart 寄存器有一决定 HC 应开始处理周期列表最早时间的 14-位可编程值。

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

位	比特符号	类型 (HCD)	类型 (HC)	功能
31-14	–	–	–	保留
13-0	PS[13:0]	R/W	R	文件名: Periodic Start 在硬件复位后，该字段被清除。随后，该字段在HC初始化期间由HCD设置。该值通过计算得出，与HcFmInterval大约有10%的偏差。典型值将为3E67h。当HcFmRemaining达到指定值时，周期性列表处理的优先级将高于控制/批量处理。因此，在完成正在进行中的当前控制或批量交易后，HC将开始处理中断列表。

18.6.18 HcLSThreshold寄存器

HcLSThreshold 寄存器包含一由主机控制器所使用的 12-位值，用于确定 EOF 前是否传送一最大为 8-字节 LS 包。无论主机控制器还是主机控制器驱动程序都不允许修改该值。

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

位	比特符号	类型 (HCD)	类型 (HC)	功能
31-12	–	–	–	保留
11-0	LST[11:0]	R/W	R	文件名:LS Threshold 该字段包含一个在启动低速事务之前与FrameRemaining字段相比的值。仅当帧在FrameRemaining时才开始处理。考虑到传输和设置架空，该值由HCD计算。

18.6.19 HcRhDescriptorA寄存器

HcRhDescriptorA 寄存器是描述根集线器特点的两个寄存器之一。复位值为执行特定。可由 HCD 对描述符长度，描述符类型和集线器类描述符的集线器控制器当前字段进行仿真。所有其他字段都位于 HcRhDescriptorA 和 HcRhDescriptorB 寄存器。

	31	30	29	28	27	26	25	24
比特符号	POTPGT							
复位后	0	0	0	0	0	0	1	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后								
	15	14	13	12	11	10	9	8
比特符号	-	-	-	NOCP	OCPM	DT	NPS	PSM
复位后				0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	NDP							
复位后	0	0	0	0	0	0	0	1

位	比特符号	类型 (HCD)	类型 (HC)	功能
31-24	POTPGT[7:0]	R/W	R	文件名:On To Power Good Time 该字节规定了HCD访问根集线器的已通电端口之前需等待的时间。它为专门执行。时间单位为2ms。持续时间按POTPGT×2 ms计算。
23-13	–	–	–	保留
12	NOCP	R/W	R	文件名:No Over current Protection 该位描述根集线器端口的过流状态是如何报告的。当该位被清除，OverCurrentProtectionMode字段指定全局或每端口报告。 0: 对于所有组下行埠，过流状态是统一报告的。 1: 不支持过流保护。
11	OCPM	R/W	R	文件名: Over Current Protection Mode 该位描述根集线器端口的过流状态是如何报告的。复位时，这些字段应反映与PowerSwitchingMode相同的模式。该字段仅在NoOverCurrentProtection字段被清零时有效。 0: 对于所有组下行埠，过流状态是统一报告的。 1: 过流状态以每端口的形式进行报告。
10	DT	R	R	设备类型 该位指定根集线器不是复合装置。根集线器不允许为复合装置。该字段应始终读/写0。
9	NPS	R/W	R	文件名: No Power Switching 这些位用于规定是否支持电源切换或端口始终通电。其为执行特定的。当该位被清除时，PowerSwitchingMode指定全局或每端口的切换。 0: 端口为电源可切换。 1: 当HC接通电源时，端口始终通电。
8	PSM	R/W	R	文件名: Power Switching Mode 该位用于规定根集线器端口的电源切换是如何控制的。其为执行特定的。该字段仅在NoPowerSwitching字段被清零时有效。 0: 所有端口同时通电。 1: 每个端口单独通电。该模式允许全局开关或各端口切换控制端口电源。若PortPowerControlMask位被设置，该端口仅响应端口电源命令 (Set/ClearPortPower)。如果端口掩码被清零，则该端口仅由全局电源开关控制。 (设置/清除全局电源).
7-0	NDP[7:0]	R	R	文件名:Number DownStream Ports 这些位指定由根集线器支持的下游端口的数目。其为执行特定的。此模块有一个端口所以读取 “0X01“ 。

18.6.20 HcRhDescriptorB寄存器

The HcRhDescriptorB 寄存器为描述根集线器特征的两种寄存器之一。这些字段在初始化期间被写入，以与系统实施相一致。复位值为执行特定。

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

位	比特符号	类型 (HCD)	类型 (HC)	功能
31-16	PPCM[15:0]	R/W	R	文件名:Port Power Control Mask 当PowerSwitchingMode被设置时，每一位指示端口是否受全局电源控制命令的影响。当被设置时，端口的电源状态仅受每端口电源控制 (Set/ClearPortPower)的影响。当被清除时，端口受全局电源开关(Set/ClearGlobalPower)的控制。若该装置被设置为全局切换模式 (PowerSwitchingMode=0)，该字段无效。 位0: 保留 位1: #1端口上的联动-电源掩码 位2: #2端口上的联动-电源掩码 (注) 位15: #15端口上的联动-电源掩码
15-0	DR[15:0]	R/W	R	文件名:Deveice Removable 每一位针对根集线器的一个端口。当该位为 “0“时，附接装置是可移除的。当该位为 “1“时，附接装置不可移除。 位0: 保留 位1: 附接到端口#1的装置 位2: 附接到端口#2的装置 (注) 位15: 附接到端口#15的装置

注:**由于该主机控制器不具有端口#2 ~ 端口#15，在相应位中写入 “0“。**

18.6.21 HcRhStatus 寄存器

HcRhStatus 寄存器分为两个部分双字值的低位字表示集线器状态字段，高位字表示集线器状态改变字段。预留位应始终写入 “0“。

	31	30	29	28	27	26	25	24
比特符号	CRWE	-	-	-	-	-	-	-
复位后	未定义							
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	OCIC	LPSC
复位后							0	0
	15	14	13	12	11	10	9	8
比特符号	DRWE	-	-	-	-	-	-	-
复位后	0							
	7	6	5	4	3	2	1	0
比特符号	-	-	-	-	-	-	OCI	LPS
复位后							0	0

位	比特符号	类型 (HCD)	类型 (HC)	功能
31	CRWE	W	R	文件名: Clear Remote Wakeup Enable 写入 “1“将DeviceRemoveWakeupEnable清零。写入“0“无效。
30-18	–	–	–	保留
17	OCIC	R/W	R/W	文件名:Over Current Indicator Change 当该寄存器的OCI字段发生改变时，该位由硬件设置。HCD通过写入 “1“将该位清除。写入 “0“无效。
16	LPSC	R/W	R	文件名:Local Power Status Change (读)LocalPowerStatusChange 根集线器不支持局部电源状态特征，因此，该位始终读为 “0“。 (写)SetGlobalPower 在全局电源模式中 (PowerSwitchingMode= “0“)，在该位中写入 “1“以打开全部端口的电源 (set PortPowerStatus)。在每端口电源模式中，仅在PortPowerControlMask位未被设置的端口上对PortPowerStatus设置。写入 “0“无效。
15	DRWE	R/W	R	文件名:Device Remote Wakeup Enable (读取)DeviceRemoteWakeupEnable 该位启用ConnectStatusChange位作为恢复事件，导致USBSUSPEND到USBRESUME状态的过渡，并设置ResumeDetected中断。 0: ConnectStatusChange不是一远程唤醒事件。 1: ConnectStatusChange是一远程唤醒事件。 (写入) 写入“1“ 以对DeviceRemoveWakeupEnable置位。写入 “0“无效。 (参见“18.8 关于使用USB主机控制器的限制“)
14-2	–	–	–	保留
1	OCI	R	R/W	文件名: Over Current Indicator 当实施全局报告时，该位报告过电流状态。当将该位设置时，存在一过流状态。当将该位清除时，所有电源操作正常。如果实施每端口过流保护，该位始终为 “0“。
0	LPS	R/W	R	文件名: LocalPowerStatus (读)LocalPowerStatus 根集线器不支持局部电源状态特征，因此，该位始终读为 “0“。 (写)ClearGlobalPower 在全局电源模式下 (PowerSwitchingMode=0) ，向该位中写入“1“以关闭所有端口的电源 (将PortPowerStatus清零)。在前-端口电源模式下，它仅将PortPowerControlMask位未被设置端口的PortPowerStatus清除。写入 “0“无效。

18.6.22 HcRhPortStatus1寄存器

HcRhPortStatus1 寄存器用来以每端口的形式控制和报告端口事件。NumberDownstreamPorts 表示在硬件中实施的 HcRhPortStatus 寄存器的数量。低位字用来反映端口的状态，而高位字反映状态变化位。某些状态位以特殊的写入行为来实现 (见下文)。如果事务 (通过握手令牌)正在进行时，发生一个端口状态变化的写入，由此产生的端口状态变化必须推迟，直到交易完成。预留位应始终写入 “0“。

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后								
	23	22	21	20	19	18	17	16
比特符号	-	-	-	PRSC	OCIC	PSSC	PESC	CSC
复位后				0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	LSDA	PPS
复位后				0	0	0	未定义	0
	7	6	5	4	3	2	1	0
比特符号	-	-	-	PRS	POCI	PSS	PES	CCS
复位后				0	0	0	0	0

位	比特符号	类型 (HCD)	类型 (HC)	功能
31-21	–	–	–	保留
20	PRSC	R/W	R/W	文件名:Port Reset Status Change 该位在10 ms端口复位信号结束时被设置。 HCD写入 “1“以将该位清除。写入“0“无效。 0: 端口复位未完成。 1: 端口复位完成。
19	OCIC	R/W	R/W	文件名:Port Over Current indicator Change 该位仅在过流情况以前端口的形式报告时有效。当根集线器改变PortOverCurrentIndicator位时，该位被设置。 HCD写入“1“ 以将该位清除。 写入“0“无效。 0: PortOverCurrentIndicator无变化 1: PortOverCurrentIndicator 已变化。
18	PSSC	R/W	R/W	文件名: Port Suspend Status Change 该位在完全恢复序列完成时被设置。该序列包括20 s恢复脉冲，LS EOP，和3ms的重同步延迟。HCD写入“1“以将该位清除。写入 “0“ 无效。当ResetStatusChange被设置时，该位被清除。 0: 恢复未完成。 1: 恢复完成。
17	PESC	R/W	R/W	文件名: Port Enable Status Change 当硬件事件导致PortEnableStatus位被清除时，该位被设置。来自HCD写入的变化不会将该位设置。HCD写入“1“ 以将该位清除。写入“0“无效。 0: PortEnableStatus不发生变化。 1: PortEnableStatus发生变化
16	CSC	R/W	R/W	文件名:Connect Status Change 每当发生连接或断开事件时，该位被设置。HCD写入 “1“以将其清除。 位 写入“0“无效。当发生SetPortReset，SetPortEnable，或SetPortSuspend写入时，如果CurrentConnectStatus被清零，则该位被设置，以使驱动程序重新评估连接状态，因为端口断开时不该发生这些写入。 0: CurrentConnectStatus不发生变化 1: CurrentConnectStatus发生变化 (注) 如果DeviceRemovable [NDP]位被设置，当该装置被连接到系统时，该位通知仅设置一个根集线器复位。
15-10	–	–	–	保留

位	比特符号	类型 (HCD)	类型 (HC)	功能
9	LSDA	R/W	R/W	(读入)Low Speed Device Attached 该位指示连接到这个端口的装置的速度。当被设置时，一低速装置连接到该端口。当被清除时，一全速装置连接到该端口。该字段仅在CurrentConnectStatus被设置时有效。 0: 全速装置连接。 1: 低速装置连接。 (写)Clear Port Power HCD通过在该位中写入 “1“将PortPowerStatus位清除。写入“0“无效。
8	PPS	R/W	R/W	(读入) Port Power Status 无论实施何种电源切换类型，该位始终反映端口的电源状态。如果检测到过流情况，该位被清除。HCD通过写SetPortPower或SetGlobalPower将该位设置。HCD通过写ClearPortPower或ClearGlobalPower将该位清除。由PowerSwitchingMode和PortPortControlMask [NDP]确定启用哪个电源控制开关。在全局切换模式(PowerSwitchingMode=“0“) 下， 只有Set/ ClearGlobalPower 控制该位。在每端口电源切换模式(PowerSwitchingMode=“1“) 下， 如果该端口的PortPowerControlMask[NDP] 位被设置，仅启用Set/ClearPortPower命令。如果掩码未被设置，仅启用Set/ ClearGlobalPower命令。当端口电源被禁用时，CurrentCon-nectStatus，PortEnableStatus，PortSuspendStatus和PortResetStatus应被复位。 0: 端口关闭 1: 端口通电 (写入)设置端口电源 HCD写入‘1’以将PortPowerStatus位设置。写入‘0’时无任何影响。 注: 如果不支持电源切换，该位始终读为 “1“。
7-5	–	–	–	预留
4	PRS	R/W	R/W	(读取)Power Reset Status 当该位由对SetPortReset的写入设置时，端口复位信令被断言。当复位完成后，该位在PortResetStatusChange设置时被清除。如果CurrentConnectStatus被清除，该位无法被设置。 0: 端口复位信号是不活跃的 1: 端口复位信号激活 (写)设置端口复位 HCD通过在该位中写 “1“ 以设置端口复位信令。写入 “0“无效。如果CurrentConnectStatus被清除，该写入不会将PortResetStatus设置，而是将ConnectStatusChange设置。通知驱动程序，其试图复位断连端口。
3	POCI	R/W	R/W	(读取)Port Over current Indicator 该位仅在根集线器以如下方式配置时有效，即以每端口的形式报告过流情况。如果不支持前端口过流报告，该位被置为“0“。如被清除，对该端口的所有电源操作是正常的。如被设置，该端口中存在过流情况。该位始终反映过流输入信号。 0: 无过流情况 1: 检测到过流情况 (写)ClearSuspendStatus HCD写入“1“以启动恢复。写入“0“无效。恢复仅在PortSuspendStatus被设置时启动。
2	PSS	R/W	R/W	(读)Port Suspend Status 该位指示该端口被暂停或处于恢复序列。它由SetSuspendState写入设置，而当PortSuspendStatusChange在恢复间隔结束被设置时清除。如果CurrentConnectStatus被清除，该位不能被设置。当PortResetStatusChange被设置在端口复位结束或当HC被置为USBRESUME状态时，该位也被清除。如果上游恢复正在进行，其应传播到HC。 0: 端口未被挂起。 1: 端口被挂起。 (写)设置端口挂起 HCD通过将“1“写入该位对PortSuspendStatus位设置。写入“0“无效。如果CurrentConnectStatus被清除，该写入操作不对PortSuspendStatus设置，而是对ConnectStatusChange进行设置。通知驱动程序试图挂起一个断连端口。

位	比特符号	类型 (HCD)	类型 (HC)	功能
1	PES	R/W	R/W	(读取) Port Enable Status 该位表示端口被启用或禁用。当检测到过流情况，断开事件，电源关闭，或操作总线错误 (如干扰)时，根集线器可将该位清除。该变化也导致PortEnabledStatusChange被设置。HCD通过写SetPortEnable将该位设置，通过写入ClearPortEnable将该位清除。当CurrentConnectStatus被清除时，该位不能被设置。当ResetStatusChange被设置时完成端口复位，或当SuspendStatusChange被设置时完成端口挂起，若该位尚未设置，将其设置。 0: 端口被禁用。 1: 端口被启用。 (写)设置端口启用 HCD通过写入 “1“将PortEnableStatus设置。写入“0“无效。如果CurrentConnectStatus被清除，该写入不会将PortEnableStatus设置，而是将ConnectStatusChange设置。此通知驱动程序试图启用一个断连端口。
0	CCS	R/W	R/W	(读取)Current Connect Status 该位反映组下杭埠的当前状态。 0: 无装置连接 1: 装置已连接 (写入)Clear Port Enable HCD将 “1“写入该位，以将PortEnableStatus位清除。写入 “0“无效。CurrentConnectStatus不受任何写入影响。 注: 当连接装置无法移除时 (DeviceRemoveable[NDP])，该位始终读 ‘1’。

18.6.23 HcBCR0寄存器

HcBCR0 寄存器控制 USBHC 和 USB 收发器 SUSPEND 状态的启用或禁用的过流输入。

USBHC 在 SLOW 模式或低功耗模式下是否激活。因此，在进入 SLOW 模式或低功耗模式之前，将 USBHC 和 USB 收发器置于 SUSPEND 状态。

	31	30	29	28	27	26	25	24
比特符号	-	TRNS_SUSP	OVCE	-	-	-	-	-
复位后	0	1	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4.	3	2	1	0
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31	–	R	保留
30	TRNS_SUSP	R/W	文件名:Tranceiver Suspend 该位控制USB收发器的SUSPEND状态。 为了在USB收发器处于SUSPEND状态时进入SLOW模式或低功耗模式，将该位置为 “1“。 0:- (由USB主机控制器控制) 1: 挂起
29	OVCE	R/W	文件名:USB Host Over Current input Enable 0: 启用 1: 禁用
28-0	–	R	保留

18.7 有关使用USB主机控制器的标注

18.7.1 设置USB时钟

复位释放后，PLL 未激活。因此，访问 USB 主机前，必须将 CGPLLSEL <PLLSEL>置为 “1“。

18.7.2 振荡器推荐

要为 USBHC 生成一个时钟，我们建议使用具有±50ppm 或以内精度的 12-MHz 晶体振荡器，以符合 USB 规范。

注意根据实施环境的要求，条件和变化，由片上 PLL 生成的 USB 时钟可能不能满足 USB 规范要求。

18.7.3 进入SLOW模式和低功耗模式

在进入 SLOW 模式和低功耗模式之前，USBHC 必须被置为 SUSPEND 状态。然后关闭 Vbus。

(软件设置举例)

1	禁用中断				
2	HcCommandStatus	<HCR>	= 1	(写入)	:USB主机控制器的软件复位
3	HcControl	<HCFS>	= 1 1	(读取)	:检查向SUSPEND状态的传输。
4.	HcBCR0	<TRANS_SUSP>	= 1	(写入)	:将D +和D -引脚的状态变为SUSPEND状态
5	输出 “0“ ~ PG6				:Vbus OFF
6	设置到低功耗和其他模式。				:外设功能，端口设置，预热设置等的停止。关于转移BACKUP模式，请参见该节。
7	停止PLL				
8	执行WFI指令				

18.7.4 不使用USB时

D+和 D-引脚必须下拉。

18.7.5 对RAM0和RAM1的竞争访问

当 USBHC 在访问 RAM0 或 RAM1 时，如果 CPU/DMA 访问 RAM0 或 RAM1，则中断 USBHC 的连接，因为 CPU/DMA 具有更高的访问优先级。当 CPU/DMA 访问完成时，重新连接 USBHC。请注意，如果由 CPU/DMA 引起的 USBHC 中断时间过长，可能会导致 USB 的传输问题。，在 IN 传输的情况下，CPU / DMA 必须在包含于 USBHC 中的 64-字节 FIFO 已满期间 (在 42.66μs 时间内)完成访问。

18.8 有关使用USB主机控制器的限制

1. 对于等时传输，待传输的帧号在等时传输描述符 (ITD)中定义。然而，帧编号没有在主机和软件之间同步。如果前一帧要执行的描述符调度推迟，主机判定发生时间错误，并将 DATAOVERRUN 写回到 ITD 的 CC 字段。此时，如果满足以下条件，主机将写回不正确的状态 (NOERROR)。

   <条件>

   如果在满足以下两个条件的方式下对传输进行调度，就会发生上述问题:

   1 ITD.FC[2:0] = R[2:0]

   2 ITD.FC[2:0] < R[15:0]

   凡在 ITD.FC 指示 ITD 被执行次数的数量时，R = HcFmNumber (当前帧号)- ITD.SF (传输起始帧号)。

   确保每个 ITD 同步到当前帧号。如不满足该条件，不应连接该 ITD。

2. 如果 USB 系统上发生一个致命的错误，且主机检测到这个错误，OHCI 核心将 HcInterrutStatus.UE [4]设置。

   此时，如果 HcInterruptEnable.UE [4]寄存器已经被置位，将产生一个硬件中断。检测到该中断后，需要将一软件复位 (HcCommandStatus.HCR [0] = 0y1)从不可恢复的错误状态恢复，然后主机变化到 SUSPEND 状态。

3. 软件复位后，OHCI 寄存器被初始化.如果发生装置的远程唤醒，主机将保持 SUSPEND 状态。当使用远程唤醒功能时，必须准备一个用于从 SUSPEND 状态恢复的程序.

4. 当发生过流错误时，如果 HcRhDescriptorA.NPS [9]寄存器已被置为 “1“，则 HcRhPortStatus1.PRS [4]和 HcRhPortStatus1.PSS[2]位不会被清除。因此，不要将 HcRhDescriptorA.NPS[9]置为 “1“，且不要将 HcRhDescriptorB.DR[端口号]置为 “1“。

5. 当 HcRhStatus.DRWE [15]被置为 “1“时，远程唤醒事件不会引起从 USBSUSPEND 到 USBRESUME 状态的过渡。当使用远程唤醒事件时,由 HCD 对 HcInterruptStatu.RD[3] 的监视引起状态过渡。而当不使用远程唤醒事件时，不要将 HcRhStatus.DRWE [15]置为 “1“。

6. 在支持过流情况的系统中，将 HcRhDescriptorA.NOCP [12]置为 “0“。

7. 在产生干扰时执行端口复位，端口将保持禁用状态。端口复位顺序如下。

示例

1	HcRhPortStatus1	<PRS>	= 1	(写)	:执行端口复位
2	检测 HW 中断				
3	HcRhPortStatus1	<PRSC>	= 1	(读)	:端口为无效状态。
	HcRhPortStatus1	<PES>	= 0		
	HcRhPortStatus1	<CCS>	= 1		
4	清除 HW 中断				
5	HcRhPortStatus1	<PRS>	= 1	(写)	:重试端口复位
6	监测是否 HW 中断				
7	HcRhPortStatus1	<PRSC>	= 1	(读)	:端口在有效状态。
	HcRhPortStatus1	<PES>	= 1		
	HcRhPortStatus1	<CCS>	= 1		

8. 当执行全速等时传输时，若发生调度超载，可能会产生不可恢复的错误。当发生不可恢复的错误时，通过软件复位来恢复不可恢复的错误。

9. 从 USB 装置接收到的数据通过总线 I/F 作为接收缓冲器被写入到一指定内部 RAM，尽管如此，数据并不是在如下条件下被写入到内部 RAM。将起始地址与 4 的倍数对齐，不要指定从接收缓冲区 4n+1 位开始的地址。

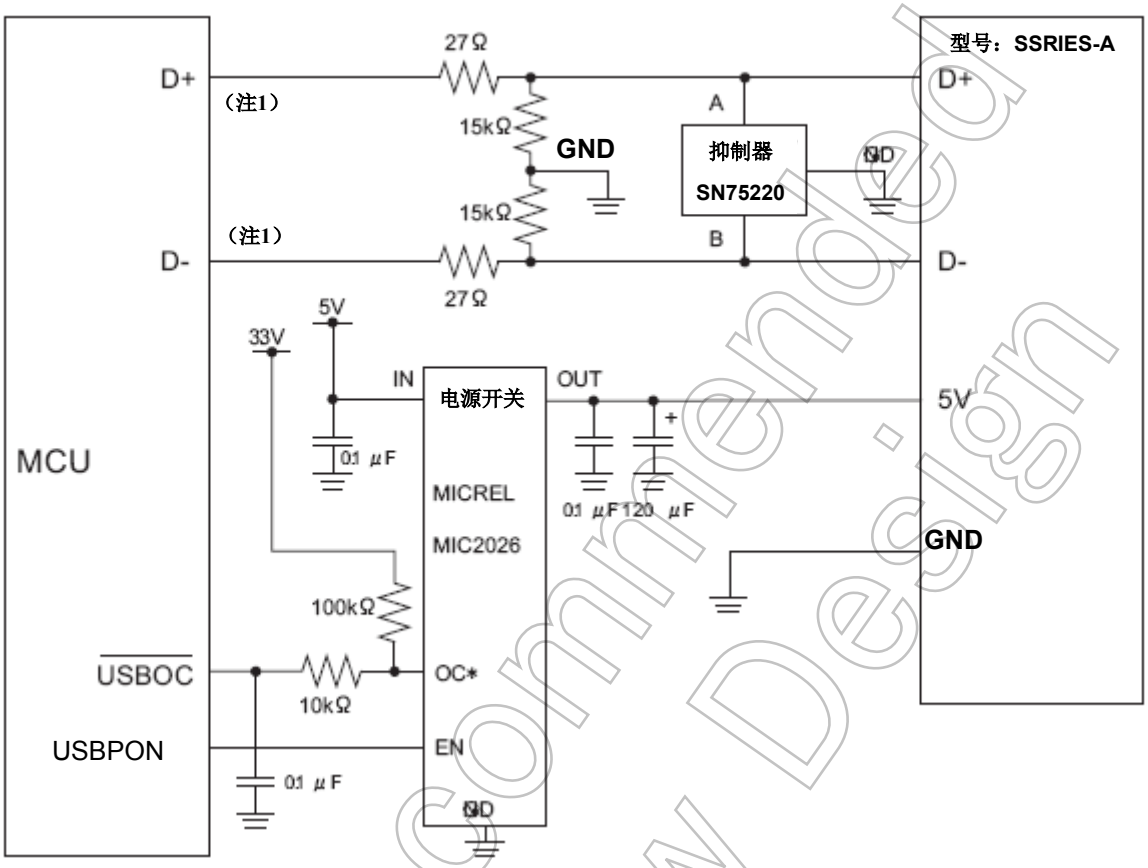
4n 表示该地址与 4 字节对齐。

<条件忽略写访问>

当下列条件全部满足时，写访问被忽略。

1. 接收到的数据为 MPS (最大值数据包大小) × n +2
2. 写缓冲区的最后 2 个字节的地址是 4n +1 和 4n +2。

18.9 连接实例



*低信号有效

总线电源开关装置	: TPS2052 (德州仪器)
	: MIC2026-1BM (MICREL)
	: MIC2026-1BN (MICREL)
瞬态电压抑制器装置	: SN75240 (德州仪器)
阻值精度	: 5%
阻值等级	: 推荐27É¹为1/2W
电容器	: 推荐低ESR类型120μF电容器 (OS-CON等)

注 1: **不要使用超过绝对最大额定值的电压。**

注 2: **设计电路板时，请确保 D +和 D -引脚被置于离 USB 插座距离相等的地方。**

注 3: **USB 规范中并未要求使用抑制装置。**

注 4: **复位释放后，USBPON 和 $\overline{\text{USBOC}}$ 引脚作为输入端口被初始化。对引脚进行处理，以使外部控制电路不受影响。**

19. 看门狗定时器 (WDT)

看门狗定时器 (WDT)用于检测由噪音或其他干扰引起的 CPU 故障 (失控)，并用于修复故障使 CPU 正常运行。

当检测到故障时，看门狗定时器使 INTWDT 生成中断或复位。

注:INTWDT 中断是非屏蔽中断 (NMI)的一个因素。

而且，看门狗定时器 ($\overline{\text{WDTOUT}}$)引脚输出 “低“时，它则通知外围设备检测故障。

19.1 配置

图 19-1 展示了看门狗定时器的方块图。

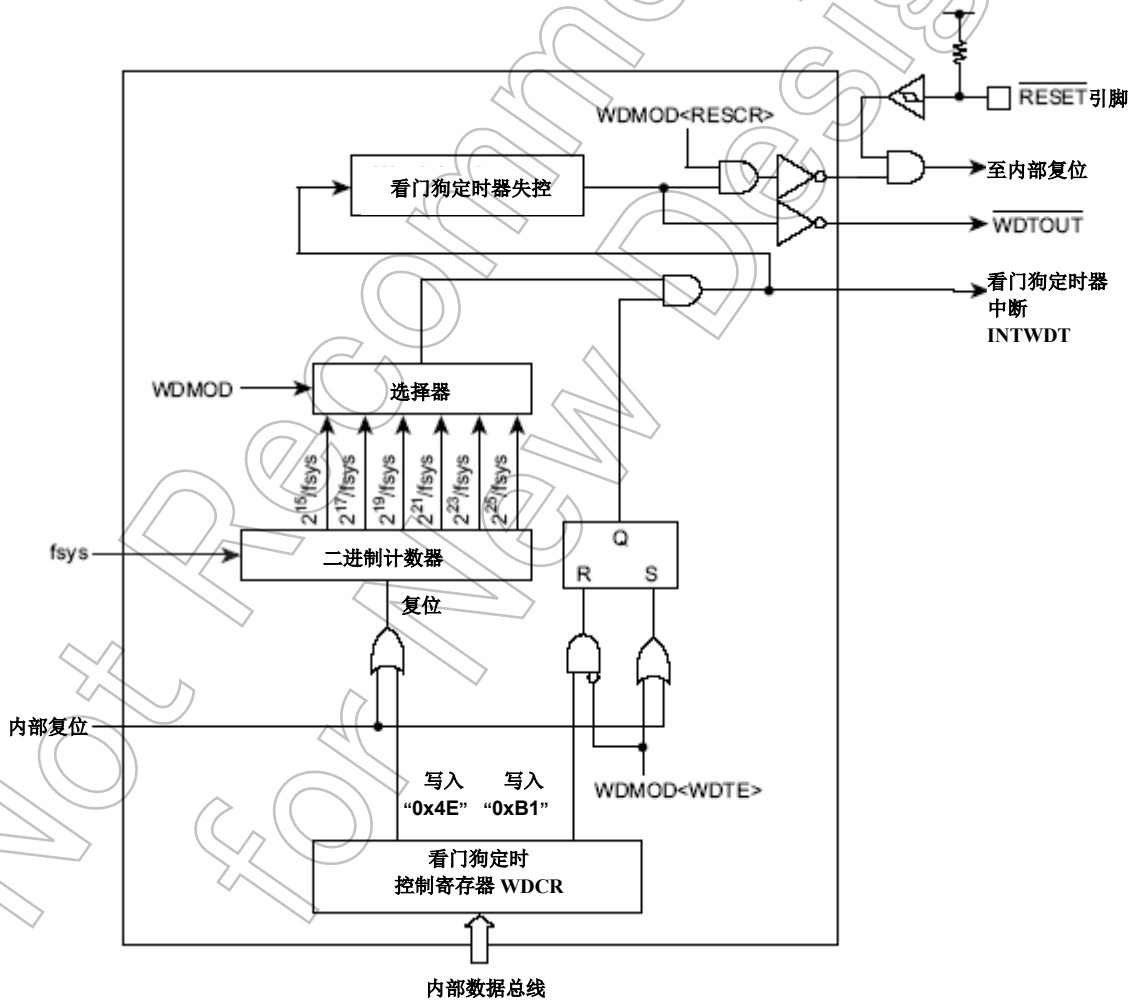


图 19-1 看门狗定时器方块图

19.2 寄存器

看门狗定时器控制寄存器和地址如下所示。

基址 = 0x400F_2000

寄存器名称		地址 (基址+)
看门狗定时器模式寄存器	WDMOD	0x0000
看门狗定时器控制寄存器	WDCR	0x0004

19.2.1 WDMOD (看门狗模式寄存器)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	WDTE	WDTP			-	I2WDT	RESCR	-
复位后	1	0	0	0	0	0	1	0

位	比特符号	类型	功能
31-8	–	R	读作 0。
7	WDTE	R/W	启用/禁用控制 0:禁用 1:启用
6-4	WDTP[2:0]	R/W	选择WDT检测时间 (见表19-1) 000: 2^{15}/fsys 100: 2^{23}/fsys 001: 2^{17}/fsys 101: 2^{25}/fsys 010: 2^{19}/fsys 110:设置禁止. 011: 2^{21}/fsys 111:设置禁止.
3	–	R	读作 0。
2	I2WDT	R/W	IDLE模式时操作 0: 停止 1:运行中
1	RESCR	R/W	故障检测后运行 0: INTWDT中断请求产生。 (注) 1: 复位
0	–	R/W	写入 0。

注:INTWDT 中断是非屏敝中断 (NMI)的一个因素。

表 19-1 看门狗定时器检测时间 (fc = 64MHz)

时钟齿轮值 CGSYSCR<GEAR[2:0]>	WDMOD<WDTP[2:0]>					
	000	001	010	011	100	101
000 (fc)	0.51 ms	2.05 ms	8.19 ms	32.77 ms	131.07 ms	524.29 ms
100 (fc/2)	1.02 ms	4.10 ms	16.38 ms	65.54 ms	262.14 ms	1.05 s
101 (fc/4)	2.05 ms	8.19 ms	32.77 ms	131.07 ms	524.29 ms	2.10 s
110 (fc/8)	4.10 ms	16.38 ms	65.54 ms	262.14 ms	1.05 s	4.19 s

19.2.2 WDCR (看门狗定时器控制寄存器)

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

位	比特符号	类型	功能
31-8	–	R	读作 0。
7-0	WDCR	W	禁用/清除代码 0xB1:禁用代码 0x4E: 清除代码 其它:保留

19.3 操作

19.3.1 基本操作

看门狗定时器由二进制计数器使用系统时钟 (fsys)作为输入组成。检测时间通过 WDMOD<WDTP[2:0]>，可选择每隔 2^{15}， 2^{17}， 2^{19},，2^{21},，2^{23} 及 2^{25} 的时间检测一次。过了指定的检测时间，看门狗定时器中断 (INTWDT)生成，且看门狗定时器引脚 ($\overline{\text{WDTOUT}}$)会输出 “低“。

检测由于噪音或其它干扰引起的 CPU 故障 (失控)噪音或其他障碍，在 INTWDT 中断生成之前，看门狗定时器的二进制计数器根据软件指令将其清除。如果二进制计数器没有清除，非屏蔽中断会通过 INTWDT 生成由此 CPU 检测故障 (失控)，故障对策程序执行回到正常操作。

此外，它通过连接看门狗定时器引脚复位外围设备引脚来解决 CPU 故障 (失控)问题。

19.3.2 操作模式和状态

复位被清除后，看门狗定时器立即开始运行。

如果不使用看门狗定时器时，就要禁用。

看门狗定时器不能作为停止的高频时钟使用。过渡到低模式之前，看门狗定时器应禁用。在 IDLE2 模式下,其操作取决于 WDMOD < I2WDT >设置。

-STOP 模式
-SLEEP 模式
-SLOW 模式
-BACKUP STOP 模式
-BACKUP SLEEP 模式

而且二进制计数器在调试模式下自动停止。

19.4 故障 (失控)检测时的操作

19.4.1 INTWDT中断产生

图 19-2 显示 INTWDT 中断产生 (WDMOD<RESCR>= “0“)。

当二进制计数器溢出时，INTWDT 中断产生。它是非屏蔽中断 (NMI)的一个因素。因此，CPU 检测非屏蔽中断同时执行响应程序。

非屏蔽中断的因素为复数。CGNMIFLG 识别非屏蔽中断的因素。在 INTWDT 中断的情况下，CGNMIFLG<NMIFLG0>被设置。

当 INTWDT 中断产生时，看门狗定时器 ($\overline{\text{WDTOUT}}$)同时输出 “低“。 $\overline{\text{WDTOUT}}$ 变为 “高“时，看门狗定时器清除写入 WDCR 寄存器的可用代码 0x4E。

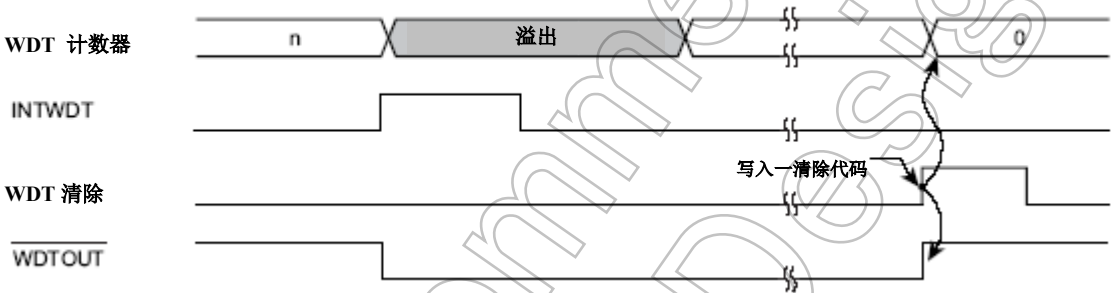


图 19-2 INTWDT 中断产生

19.4.2 内部复位生成

图 19-3 显示内部复位 (WDMOD<RESCR>= "1")。

MCU 通过二进制计数器复位。在这种情况下，复位状态扩展到 32 态。时钟初始化便于 clock (fsys) 和高频时钟 (fosc)输入相同。即 fsys = fosc。

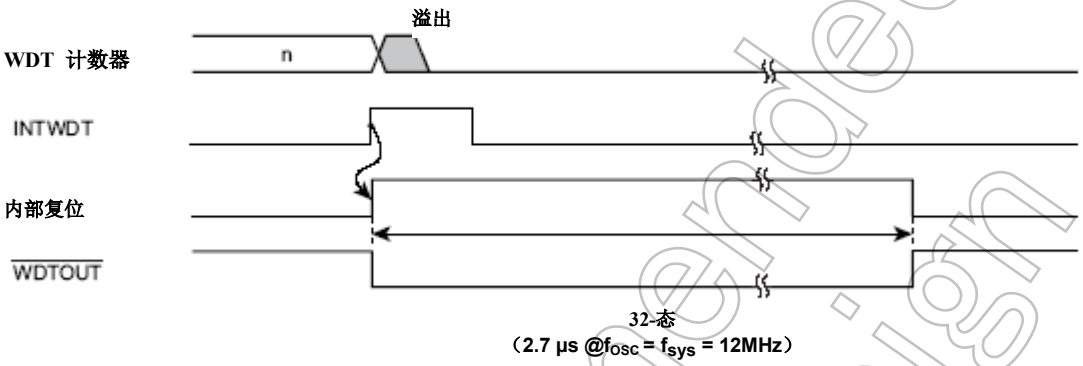


图 19-3 内部复位

19.5 控制寄存器

看门狗定时器 (WDT)由 WDMOD 和 WDCR 两个控制寄存器来控制。

19.5.1 看门狗定时器模式寄存器 (WDMOD)

1. 规定看门狗定时器<WDTP[2:0]>的检测时间。

   设置 WDMOD<WDTP[2:0]>看门狗定时器检测时间。复位后，初始化 WDMOD<WDTP[2:0]> = “000“。

2. 启用/禁用看门狗定时器<WDTE>。

   复位时，WDMOD <WDTE>初始化到 “1“，启用看门狗定时器。
   禁用看门狗定时器以免由故障引起的错误写入，首先 <WDTE> 位设置到 “0“, 然后禁用代码 (0xB1)必须写入 WDCR 寄存器。

   为使看门狗定时器的状态从 “禁用“变为 “启用“，需将<WDTE> 位设到 “1“。

3. 看门狗定时器外复位连接<RESCR>

   寄存器规定 WDTOUT 是否用于内部复位或中断。复位后， WDMOD<RESCR>初始化到 “1“，内部复位由二进制计数器溢出产生。

19.5.2 看门狗定时器控制寄存器 (WDCR)

为一个禁用看门狗定时器功能且控制二进制计数器清除功能的寄存器。

19.5.3 设置举例

19.5.3.1 禁用控制

设置 WDMOD <WDTE> 到 “0“后，WDCR 寄存器写入禁用代码 (0xB1)，看门狗定时器即禁用，且二进制计数器也被清除。

		7	6	5	4	3	2	1	0	
WDMOD	←	0	–	–	–	–	–	–	–	设置 <WDTE> 为 “0“。
WDCR	←	1	0	1	1	0	0	0	1	写入禁用代码 (0xB1)。

19.5.3.2 启用控制

设置 WDMOD <WDTE> 为 “1“。

		7	6	5	4	3	2	1	0	
WDMOD	←	1	–	–	–	–	–	–	–	设置<WDTE> 为 “1“。

19.5.3.3 看门狗计时器清除控制

WDCR 寄存器写入可用代码 (0x4E) 以清除二进制计数器并开始计数。

		7	6	5	4	3	2	1	0	
WDCR	←	0	1	0	0	1	1	1	0	写入清除代码 (0x4E)。

19.5.3.4 看门狗计时器检测时间

在 2^{21}/fsys 被使用的情况下，WDMOD<WDTP[2:0]> 设置为 “011“。

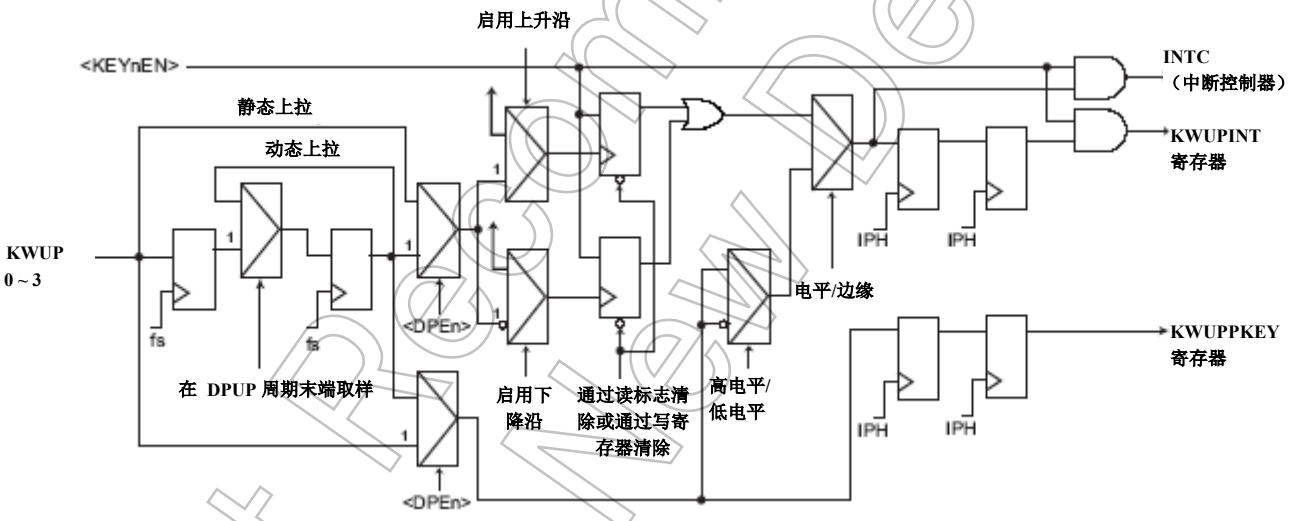
		7	6	5	4	3	2	1	0
WDMOD	←	1	0	1	1	–	–	–	–

20. 触键唤醒

20.1 概述

- TMPM364F10FG 有 4 个键盘输入，分别为 KWUP0 ~ KWUP3，可用于释放 STOP 模式或用于外部中断.注意中断处理是通过 4 个输入的一个中断因素来执行的。(已在 CG 块中编程)，每个键盘输入通过编程 (KWUPCRn<KEYnEN>)可配置为使用或不使用。
- 每个输入的活跃状态能够通过编程 KWUPCRn<KEYn>配置为上升沿，下降沿，双边沿，高电平和低电平。
- 中断请求时通过在中断处理中给按键中断请求清除记录 KWUPCLR 进行编程而清除。
- KWUP 输入引脚具有上拉功能，可通过编程 KWUPCRn<DPEn>在静态上拉和动态上拉之间切换。四个输入的每一个都需要编程。

20.2 方块图



图形 20-1 按键唤醒方块图

20.3 寄存器的详细描述

20.3.1 寄存器列表

基址 = 0x400F_1000

寄存器名称		地址 (基址+)
控制寄存器 0	KWUPCR0	0x0000
控制寄存器 1	KWUPCR1	0x0004
控制寄存器 2	KWUPCR2	0x0008
控制寄存器 3	KWUPCR3	0x000C
端口监控寄存器	KWUPPKEY	0x0080
上拉周期寄存器	KWUPCNT	0x0084
中断全部清除寄存器	KWUPCLR	0x0088
中断监控寄存器	KWUPINT	0x008C

20.3.2 KWUPCR0 (控制寄存器 0)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	DPE0	KEY0			-	-	-	KEY0EN
复位后	0	0	1	0	0	0	0	0

位	比特符号	类型	功能
31-8	–	R	读作“0“。
7	DPE0	R/W	选择静态上拉或动态上拉。 0: 静态上拉 1: 动态上拉
6-4	KEY0[2:0]	R/W	选择KWUP0的输入工作状态。 000: “低“ 电平 001: “高“ 电平 010: 下降沿 011: 上升沿 100: 双沿 以上除外: 保留
3-1	-	R	读作“0“。
0	KEY0EN	R/W	选择启用或禁用 KWUP中断KWUP0。 0: 禁用 1: 启用

20.3.3 KWUPCR1 (控制寄存器 1)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	DPE1	KEY1			-	-	-	KEY1EN
复位后	0	0	1	0	0	0	0	0

位	比特符号	类型	功能
31-8	−	R	读作“0“。
7	DPE1	R/W	选择静态上拉或动态上拉。 0: 静态上拉 1: 动态上拉
6-4	KEY1[2:0]	R/W	选择KWUP1的输入活跃状态。 000: “低“ 电平 001: “高“ 电平 010: 下降沿 011: 上升沿 100: 双沿 以上除外: 保留
3-1	−	R	读作“0“。
0	KEY1EN	R/W	选择启用或禁用KWUP 中断KWUP1。 0: 禁用 1: 启用

20.3.4 KWUPCR2 (控制寄存器 2)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	DPE2	KEY2			-	-	-	KEY2EN
复位后	0	0	1	0	0	0	0	0

位	比特符号	类型	功能
31-8	–	R	读作“0“。
7	DPE2	R/W	选择静态上拉或动态上拉。 0: 静态上拉 1: 动态上拉
6-4	KEY2[2:0]	R/W	选择输入KWUP2的激活状态。 000: “低“ 电平 001: “高“ 电平 010: 下降沿 011: 上升沿 100: 双沿 以上除外: 保留
3-1	–	R	读作“0“。
0	KEY2EN	R/W	选择启用或禁用KWUP 中断 KWUP2。 0: 禁用 1: 启用

20.3.5 KWUPCR3 (控制寄存器 3)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4.	3	2	1	0
比特符号	DPE3	KEY3			-	-	-	KEY3EN
复位后	0	0	1	0	0	0	0	0

位	比特符号	类型	功能
31-8	–	R	读作“0“。
7	DPE3	R/W	选择静态上拉或动态上拉。 0: 静态上拉 1: 动态上拉
6-4	KEY3[2:0]	R/W	选择KWUP3的输入工作状态。 000: “低“ 电平 001: “高“ 电平 010: 下降沿 011: 上升沿 100: 双沿 以上除外: 保留
3-1	–	R	读作“0“。
0	KEY3EN	R/W	选择启用或禁用KWUP 中断KWUP3。 0: 禁用 1: 启用

20.3.6 KWUPPKEY (端口监控寄存器)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	-	-	-	-	PKEY3	PKEY2	PKEY1	PKEY0
复位后	0	0	0	0	0	0	0	0

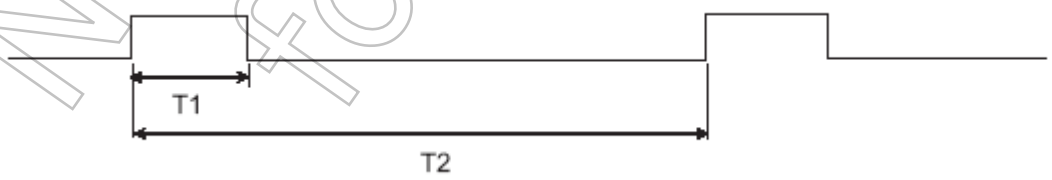
位	比特符号	类型	功能
31-4	–	R	读作“0“。
3-0	PKEY3 ~ PKEY0	R	端口状态 0: “低“ 1: “高“ 对于端口状态，可以用KWUPPKEY<PKEYn>监控外部状态。在动态上拉期间取监控样。

20.3.7 KWUPCNT (上拉周期寄存器)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	-	-	T2S		T1S		-	-
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-6	–	R	读作 “0“。
5-4	T2S[1:0]	R/W	动态上拉周期 00: 256/fs (7.8 ms @ fs = 32.768 kHz) 01: 512/fs (15.6 ms @ fs = 32.768 kHz) 10: 1024/fs (31.3 ms @ fs = 32.768 kHz) 11: 2048/fs (62.5 ms @ fs = 32.768 kHz) 通过<T2S[1:0]>对T2周期进行重复动态上拉操作.
3-2	T1S[1:0]	R/W	动态上拉期间 00: 2/fs (61.0 μs @fs = 32.768 kHz) 01: 4/fs (122.1 μs @fs = 32.768 kHz) 10: 8/fs (244.1 μs @fs = 32.768 kHz) 11: 16/fs (488.3 μs @fs = 32.768 kHz) 通过<TS[1:0]>在T1期间激活上拉，剩余期间没有激活。
1-0	-	R	读作 “0“。

动态上拉操作如下.



图形 20-2 动态上拉操作

注 1: 在使用的动态上拉期间激活 fs。

注 2: 在改变动态上拉设置之后，等待键盘输入 T1 期间。

20.3.8 KWUPCLR (全部中断要求清除寄存器)

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

位	比特符号	类型	功能
31-4	−	R	读作“0“。
3-0	KEYCLR[3:0]	W	KWUP全部中断请求通过写“1010“来清除。 读作“0“。

20.3.9 KWUPINT (中断监控寄存器)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	-	-	-	-	KEYINT3	KEYINT2	KEYINT1	KEYINT0
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-4	−	R	读作“0“。
3-0	KEYINT3 ~ KEYINT0	R	中断 0:No 1:Yes

当 KWUPCRn<KEYnEN>= “1“ 并激活信号，进入按键唤醒端口，KWUPINT 的<KEYINTn>对应通道设置为 “1“，用于中断执行。

KWUPINT 是只读寄存器，由于读取该寄存器相应的位设定为 “1“，中断请求被清除。通过 KWUPCLR 寄存器即刻清除全部中断。

在通过 KWUPCRn<KEYn>设置工作状态为 “电平“ 输入时，KWUPINT 对应于按键唤醒位保留到 “1“ 在读取时不会改变外部输入设置功能。

20.4 按键唤醒操作

TMPM364F10FG 有 4 个按键输入引脚，KEY0 ~ KEY3.对 CGIMCGF<INTLEN>寄存器编程并写入 GG，从而确定是否使用按键输入释放 STOP 模式或用于正常中断。设置<INTLEN>为 “1“，使 KEY0 ~ KEY3 的全部按键输入,用于中断以释放 STOP 模式。

对 KWUPCRn<KEYnEN>编程，用来启用或禁用每个按键输入引脚的中断输入。同时，编程 KWUPCRn<KEYn>，定义要使用的每个按键输入引脚的操作状态。

在 KWUP 块中执行按键输入检测，在有效高电平时，检测结果通知 CG 中的 CGIMCGF。随后，给 CGIMCGF<EMCGL[2:0]> 编程到 “001“，以确定检测电平到高电平。

设置 CGINCGF<INTLEN>为 “0“ (默认)，配置全部输入引脚 KEY0 ~ KEY3 为正常的中断。在这种情况下，为了实现 CPU 的检测中断请求，必须输入 “高“脉冲或 “高“电平信号

采用相同的方式编程 KWUPCRn，启用或禁用每个按键输入，并定义其工作状态。在中断处理期间，写 “1010“到 KWUPCLR<KEYCLR[3:0]>，清除全部按键中断请求。

注:如果产生两个或多个按键输入，全部的按键输入请求将通过清除中断请求进行清除。

20.5 上拉功能

每个按键输入具有上拉功能并且能够通过在端口对寄存器设置进行编程。设置静态上拉时，不能依赖于 KWUPCRn<KEYnEN>，需要使用上拉。有关动态上拉，参考 “20.3.7 KWUPCNT (上拉周期寄存器)“。

20.5.1 当采用带有启用上拉的KWUP输入时

a. 电源打开之后，可以进行第一次设置 (例如 ： 端口 J7 在双边沿处有中断)

```
PJFR2<PJ7F2>                    =  1          :功能设置到 KWUP3
PJPUP<PJ7UP>                    =  1          :上拉开控制
PJIE<PJ7IE>                     =  1          :启用输入功能
KWUPCR3<KEY3EN>                 =  0          :禁用中断
KWUPCR3<KEY3[2:0]>              =  1  0  0    :改变工作状态 (双边沿)
等待完成上拉
KWUPCLR<KEYCLR[3:0]>            =  1  0  1  0 :清除中断请求
KWUPCR3<KEY3EN>                 =  1          :启用中断
CGIMCGF<EMCGL[2:0]>             =  0  0  1    :改变激活电平 ( “高“ 电平)
CGIMCGF<INTLEN>                 =  1          :启用INTKWUP
```

b. 在操作期间改变输入的 KWUP 工作状态时

```
中断启用清除寄存器 2 [1]        =  1          :禁用INTKWUP
KWUPCR3<KEY3EN>                 =  0          :禁用中断
KWUPCR3<KEY3[2:0]>              =  0  0  0    :改变工作状态 ( “低“ 电平)
KWUPCLR<KEYCLR[3:0]>            =  1  0  1  0 :清除中断请求
KWUPCR3<KEY3EN>                 =  1          :启用中断
中断优先设置寄存器               =  *  *  *    :设置中断优先权
 <PRI_33>                                     (***= 000 ~ 111)
中断启用设置寄存器 2 [1]         =  1          :启用INTKWUP
```

c. 在操作期间启用 KWUP 输入时

```
中断启用清除寄存器 2 [1]        =  1          :禁用INTKWUP
KWUPCR3<KEY3EN>                 =  0          :禁用中断
KWUPCR3<KEY3[2:0]>              =  *  *  *    :设置工作状态
                                               (***= 000 ~ 100)
KWUPCLR<KEYCLR[3:0]>            =  1  0  1  0 :清除中断请求
KWUPCR3<KEY3EN>                 =  1          :启用中断
中断优先设置寄存器               =  *  *  *    :设置中断优先权
<PRI_33>                                       (***= 000 ~ 111)
中断启用设置寄存器 2 [1]         =  1          :启用INTKWUP
```

20.5.2 假设使用带有上拉禁用KWUP输入

a. 电源打开之后，进行第一次设置时

```
PJFR2<PJ7F2>                =  1                :功能设置到 KWUP3
PJPUP<PJ7UP>                =  0                :上拉切断控制
PJIE<PJ7IE>                 =  1                :启用输入功能
KWUPCR3<KEY3EN>             =  0                :禁用中断
KWUPCR3<KEY3[2:0]>          =  0  0  0          :设置工作状态 ( “低“ 电平)
KWUPCLR<KEYCLR[3:0]>        =  1  0  1  0       :清除中断请求
KWUPCR3<KEY3EN>             =  1                :中断启用
CGIMCGF<EMCGL[2:0]>         =  0  0  1          :改变工作电平 ( “高“ 电平)
CGIMCGF<INTLEN>             =  1                :启用INTKWUP
```

b. 在操作期间，改变 KWUP 输入的工作状态时

```
中断启用清除寄存器 2 [1]    =  1                :禁用INTKWUP
KWUPCR3<KEY3EN>             =  0                :禁用中断
KWUPCR3<KEY3[2:0]>          =  *  *  *          :改变工作状态
                                                 (***= 000 ~ 100)
KWUPCLR<KEYCLR[3:0]>        =  1  0  1  0       :清除中断请求
KWUPCR3<KEY3EN>             =  1                :启用中断
中断优先设置寄存器           =  *  *  *          :设置中断优先权
 <PRI_33>                                        (***= 000 ~ 111)
中断启用设置寄存器 2 [1]     =  1                :启用INTKWUP
```

c. 在操作期间，启用 KWUP 输入时

```
中断启用清除寄存器 2 [1]    =  1                :禁用INTKWUP
KWUPCR3<KEY3EN>             =  0                :禁用中断
KWUPCR3<KEY3[2:0]>          =  *  *  *          :改变工作状态
                                                 (***= 000 ~ 100)
KWUPCLR<KEYCLR[3:0]>        =  1  0  1  0       :清除中断请求
KWUPCR3<KEY3EN>             =  1                :启用中断
中断优先设置寄存器           =  *  *  *          :设置中断优先权
 <PRI_33>                                        (***= 000 ~ 111)
中断启用设置寄存器 2 [1]     =  1                :启用INTKWUP
```

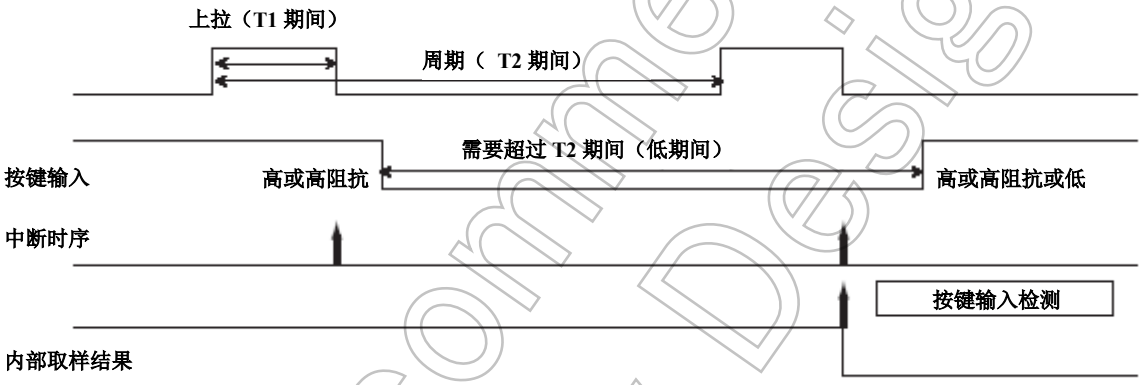
20.6 KWUP 输入检测定时

1. PJPUP<PJnUP>= “1“, KWUPCRn<DPEn>= “0“ 始终保持上拉

每个按键输入的工作状态可以通过设置 KWUPCRn<KEYn>定义为高或低电平，上升沿或下降沿。持续检测按键输入的工作状态。

2. PJPUP<PJnUP>= “1“ , KWUPCRn<DPEn>= “1“ 具有动态上拉

在 T1 期间末端的 fs 前一个时钟边缘前，检测每个按键输入的工作状态 (中断检测)。因此，需要不短于 T2 时长的按键输入。在按键输入检测之前，有多达 T2 时长的延迟。下列的图形显示了定义下降沿工作状态的例子。



21. 备份模块

21.1 特征

BACKUP模式为系统操作模式之一。在低功耗时，启动MCU操作。通过切断除备份模块外的整个模块电源，如CPU或其它外围IPs，该模式将极大降低功率消耗。

BACKUP模式含有两个模式:

- BACKUP SLEEP模式 (启动低频振荡器)

- BACKUP STOP模式 (禁用低频振荡器)

21.2 方块图

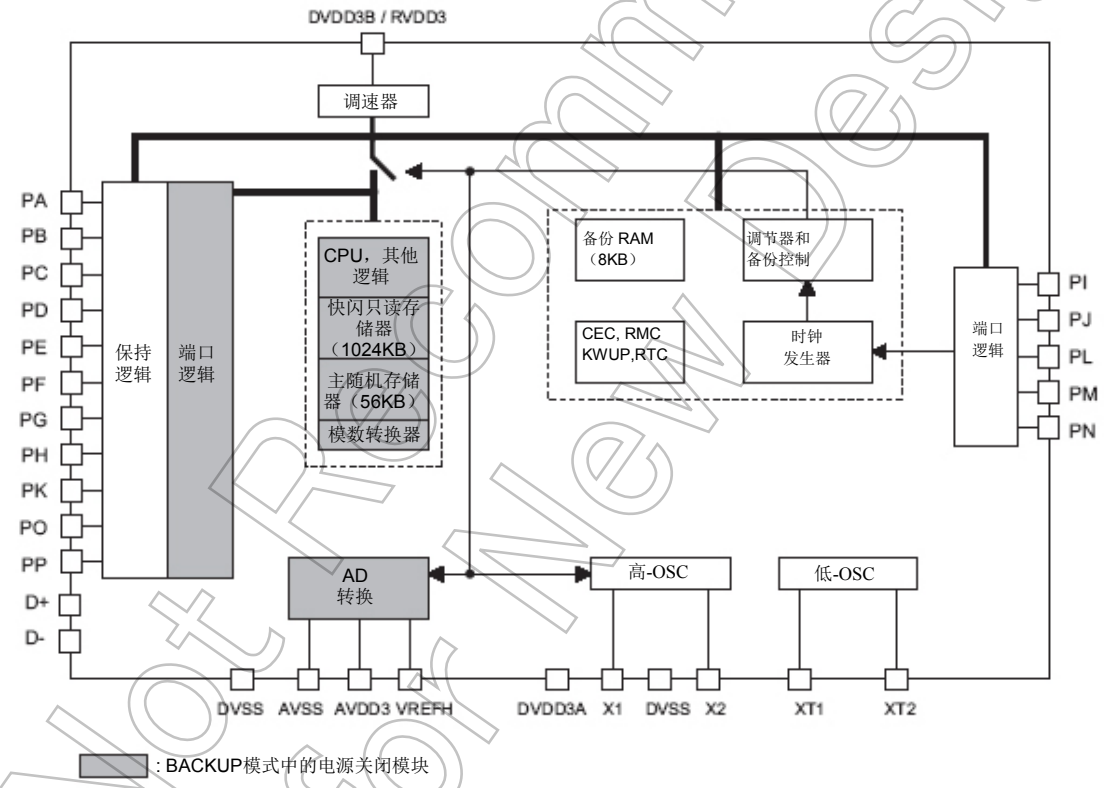


图 21-1 backup模式中的电源关闭模块

21.3 BACKUP模式运行

BACKUP模式仅对应于单片模式 (在复位之后，MCU从内置闪存启动)。BACKUP模式下，不支持单引导启动模式 (复位后MCU从内置引导启动ROM启动)。

21.3.1 BACKUP模式下可运行的外围设备

- 在BACKUP SLEEP模式下

  端口输出，触键唤醒 (KWUP)，控制扩充字符 (CEC)，遥控电路 (RMC)，实时时钟 (RTC)，低速振荡器，BACKUP RAM中的数据 (8KB)
- 在BACKUP STOP模式下

  端口输出，触键唤醒 (KWUP)，BACKUP RAM中的数据 (8KB)

21.3.1.1 转换到BACKUP模式

图21-2显示NORMAL模式，SLOW模式与BACKUP模式 (BACKUP SLEEP与BACK STOP)之间的状态转换。BACK模式 (BACKUP SLEEP与BACKUP STOP)将通过释放源返回到先前转换模式。

此外，当复位操作发生时，各模式转变到复位处理程序。

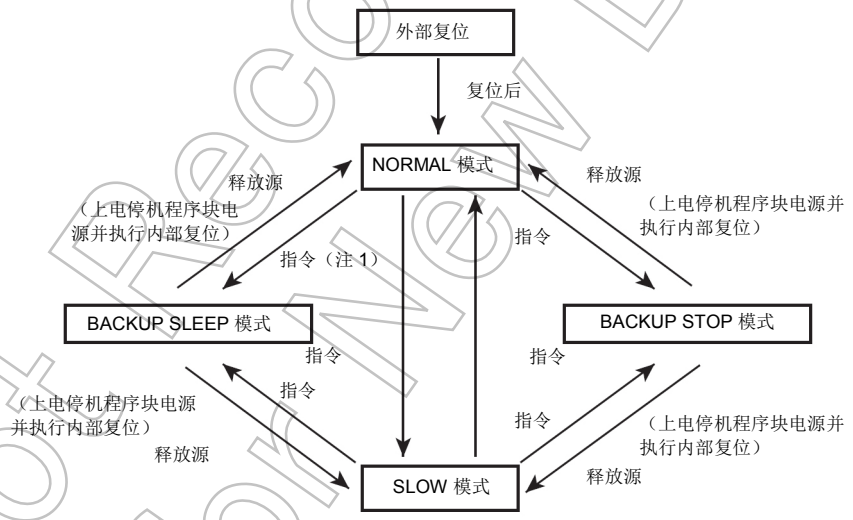


图21-2 BACKUP 模式转换图

注1: 如果低速振荡器在正常模式中被停止，确保启动低转振荡并在转换到BACKUP SLEEP模式之前确认振荡稳定。

注2: 变换到BACKUP模式的程序必须在内置闪速ROM或内置RAM中执行。

注3: 勿通过复位释放BACKUP模式。

21.3.1.2 备份转换流程

(1) BACKUP模式准备

变换到备份模式的准备程序必须在内置闪存ROM或内置RAM中执行。

1. 停止外设与保存数据

在NORMAL模式与SLOW模式中均停止外设功能，包括DMAC，SMC与WDT。在转换到BACKUP SLEEP模式时，无需停止用于此模式的外设功能 (CEC，RMC，RTC与KWUP)。有必要保存储存在BACKUP RAM中的数据。BACKUP RAM仅用于0x2000_E000 ~ 0x2000_FFFF的8KB数据。

2. 禁止中断

要阻止出现转换到BACKUP模式的障碍，必要时，将中断请求设置到禁用。要注意的是，不能禁用 $\overline{\text{NMI}}$ 中断与INTRTC中断请求，这样，必须预先避免这些中断请求。

3. 设置端口保持功能 (CGSTBYCR <PTKEEP>)

端口保持功能在CGSTBCR <PTKEEP>设置到 "1"时维持端口的暂时状态。目标端口为A ~ H，K，O，P，SWDIO，$\overline{\text{NMI}}$ 。端口保持功能能够维持上拉/下拉寄存器的输入启用/禁用，端口0/1输出状态与开/关状态。

通过在转换到BACKUP模式之前所做的这些设定，端口保持功能可保持该端口状态。在使用该端口保持功能时，各端口的端口寄存器必须得到正确设置。

端口I，J ，L，M与N的输入 / 输出状态取决于端口寄存器，与端口保持功能无关。使用这些端口，即可设置BACKUP模式的中断。

所有不必要的端口必须通过输入启用控制寄存器进行设置。

4. 时钟相关的设置与预热时间

通过设定CGOSCCR<PLLON>= "0"停止PLL电路。通过CGSYSCR<GEAR[2:0]>= "000"将高速时钟设置到fc (1/1)。通过CGOSCCR <XTEN>启动低速振荡器则需要使用BACKUPSLEEP模式。

此外，有必要通过CGOSCCR <WUPT[11:0]><WUPTL[2:0]>来设置从BACKUP模式返回的预热时间。预热时间参阅 "时钟 / 模式控制"一节。

(2) 转换到BACKUP模式

1. 设置各个模式并清除BACKUP模式的释放源

通过CGSTBYCR<STBY>寄存器设置到BACKUP STOP模式或BACKUP SLEEP模式。

2. 转换到BACKUP模式

清除从BACKUP模式释放的中断，然后执行WFI指令。

使用BACKUP模式 (关于调试工具)的注意事项
当MCU转变到BACKUP模式时，与调试工具的通信断开。在这种情况下，有必要重新连接到调试工具。

(3) 从BACKUP模式返回 (释放)

1. BACKUP模式的释放源

BACKUP STOP与BACKUP SLEEP的释放源显示如下。

BACKUP模式	BACKUP模式的释放源
BACKUP STOP	INT0 ~ 4,8,E,F, INTKWUP (静态)
BACKUP SLEEP	INT0 ~ 4,8,E,F, INTKWUP (动态 / 静态), INTRTC，INTCECRX，INTRMCRX0，INTRMCRX1

2. 通过BACKUP模式释放源进行释放操作

当接收了释放源事件时，调节器开始给停机程序块供给电源。取决于各个返回的模式，高速振荡器与低速振荡器将开始运行。

预热定时器将在振荡平稳时启动。预热时间期间，从BACKUP模式返回的电源停机程序块内部复位信号处于持续活动级。预热时间经过之后，内部复位清除，MCU返回到BACKUP模式的先前模式。

BACKUP模式释放后的注意事项
读取CGRSTFLG寄存器，可发现已发生的复位。
• 确保在通过 (CGSTBYCR <PTKEEP>= “0“)释放端口保持功能之前完成端口A，B，C，D，E，F，G，H，K，O与P的设置。

21.3.1.3 转换流程图

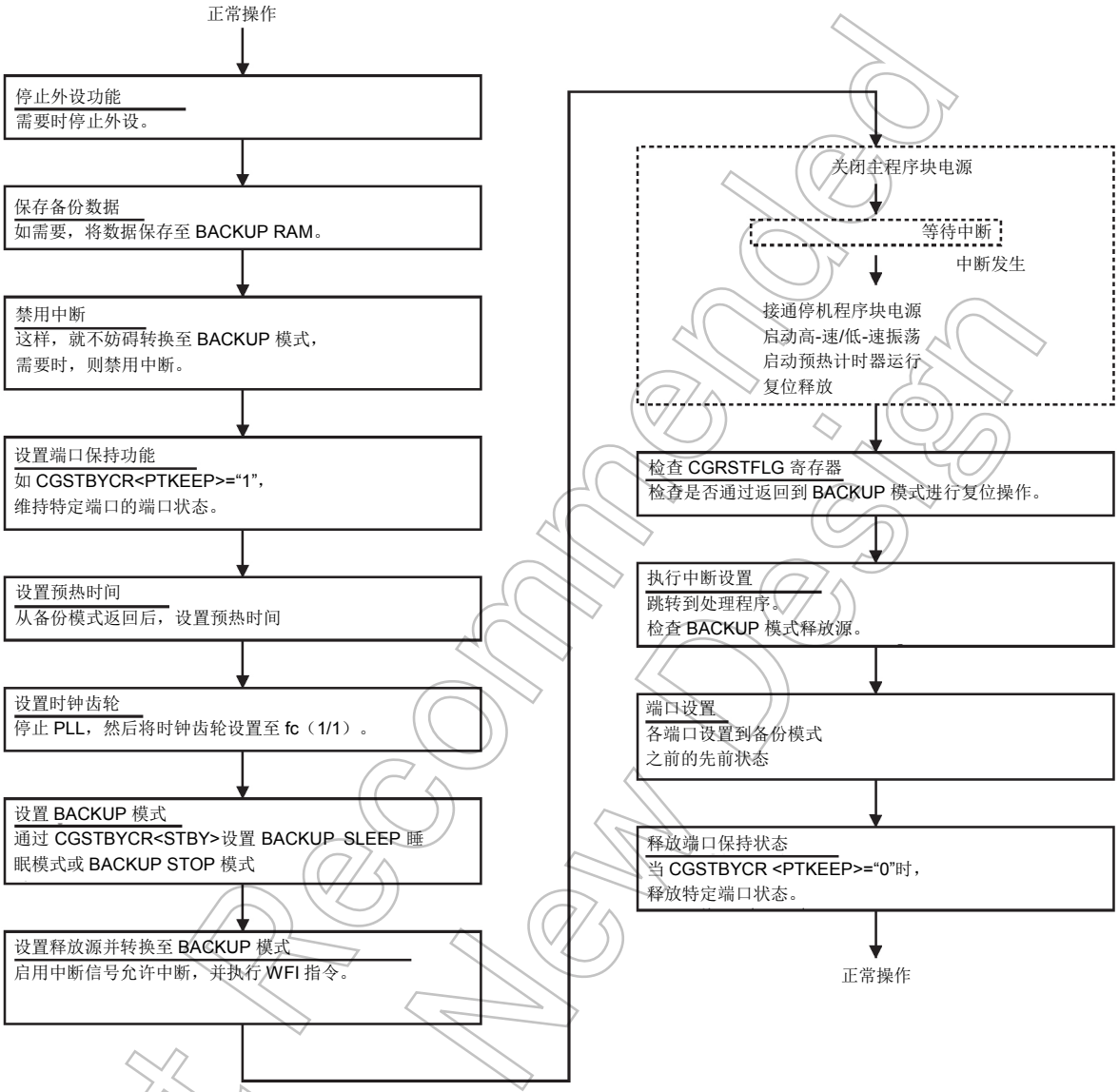


图21-3 启动转换流程图

21.3.1.4 BACKUP模式时序图

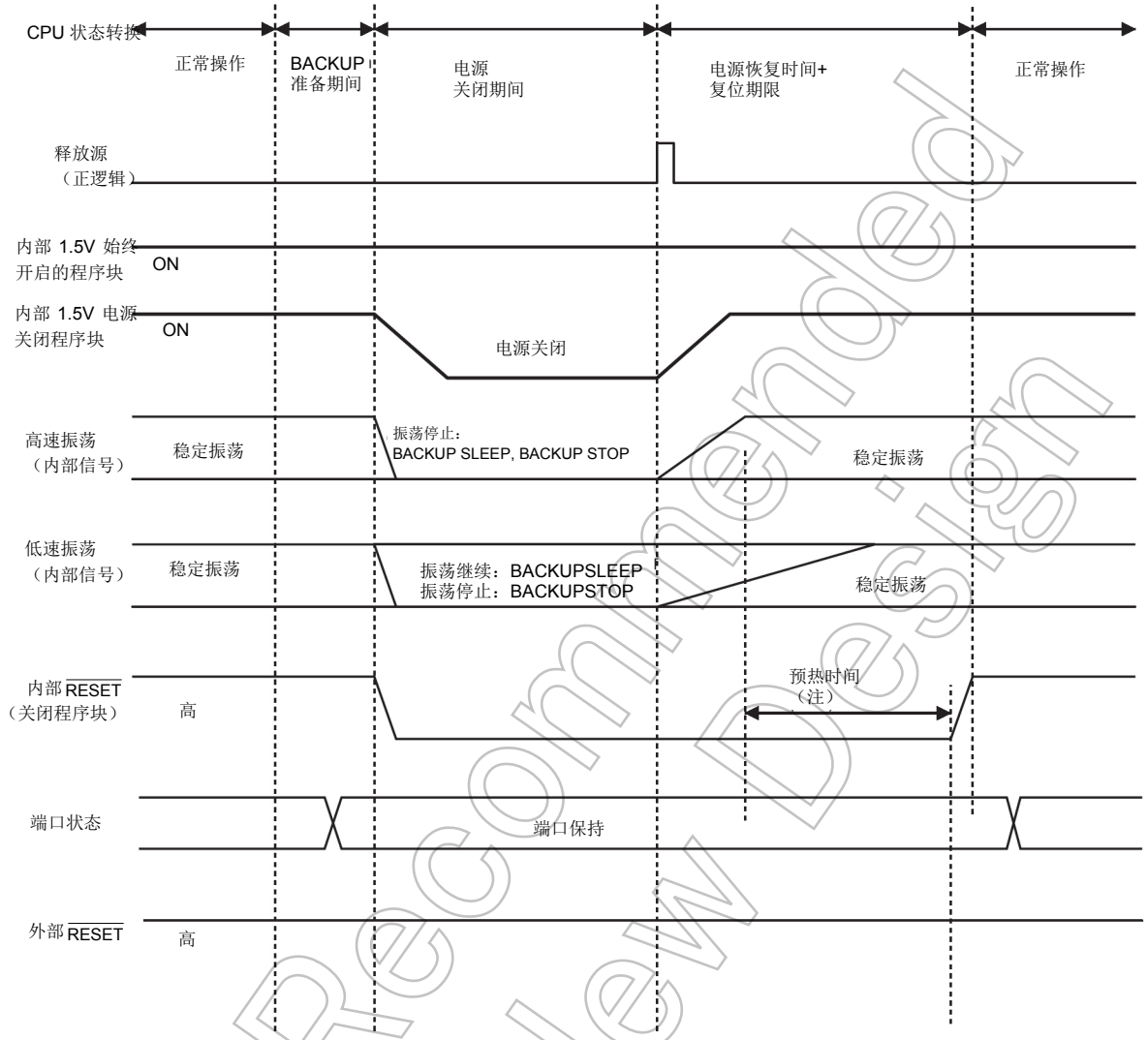


图21-4 BACKUP模式顺序

注：“图21-4 BACKUP模式顺序“显示各转换模式；NORMAL→BACKUP STOP→NORMAL或NORMAL→BACKUP SLEEP→NORMAL。本该转换中，预热计数器时钟采用高速振荡器。预热时间必须设置到500Éþs以上。

22. 模拟/数字转换器 (ADC)

22.1 概要

TMPM364F10FG具有一个10-位，有序转换的模拟/数字转换器 (AD转换器)。该AD转换器配备有16个模拟输入通道。

这16个模拟输入通道 (引脚AIN0~AIN15)同时用作输入/输出端口。

注 1: 要保障转换精度，必须给ADCBAS寄存器设置规定值。

注 2: 如果有必要以IDLE或STOP模式运行TMPM364F10FG来降低电流，如果下面所示的任何一种情况均适用，必须首先停止该AD转换器，然后执行指令，使TMPM364F10FG进入待机模式。

1. 在ADMOD 1<I2AD>为 “0 “时，必须将TMPM364F10FG置入IDLE模式。
2. 必须将TMPM364F10FG置于STOP模式。

22.2 配置

图22-1 显示了该AD转换器的方块图。

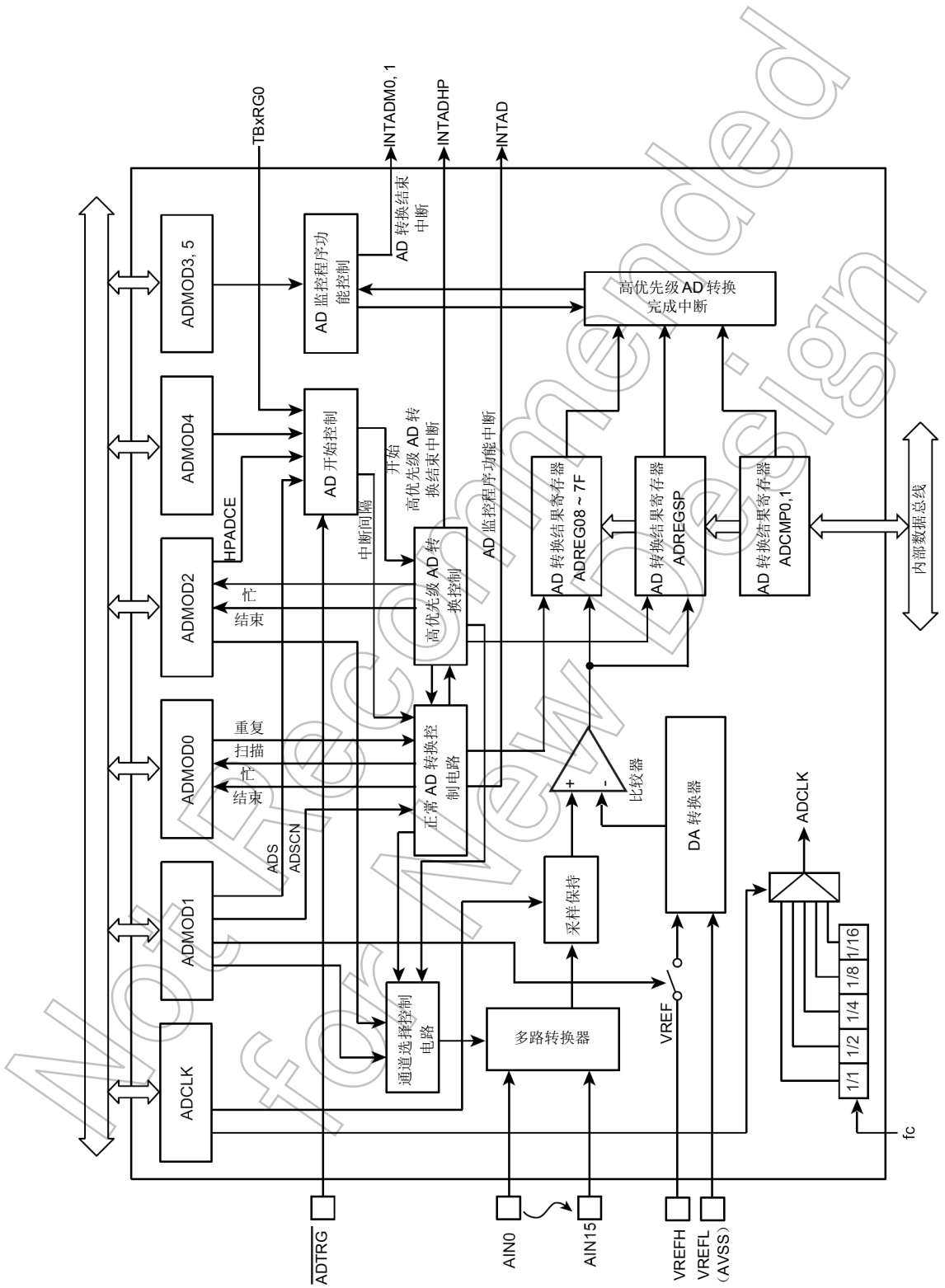


图22-1 AD转换器方块图

22.3 寄存器

22.3.1 寄存器列表

AD转换器的控制寄存器与地址如下。

AD转换器由AD模式控制寄存器（ADMOD0 ~ ADMOD5)控制。AD转换结果储存在ADREG08 ~ ADREG7F的8个AD转换结果寄存器中。最高优先级转换结果则储存在寄存器ADREGSP中。

要保障转换精度，必须给ADCBAS寄存器设置规定值。

基址=0x400F_0000

寄存器名称		地址（基址+)
转换时钟设置寄存器	ADCLK	0x0000
模式控制寄存器 0	ADMOD0	0x0004
模式控制寄存器 1	ADMOD1	0x0008
模式控制寄存器 2	ADMOD2	0x000C
模式控制寄存器 3	ADMOD3	0x0010
模式控制寄存器 4	ADMOD4	0x0014
模式控制寄存器 5	ADMOD5	0x0018
转换精度设置寄存器	ADCBAS	0x0020
保留	-	0x0024
保留	-	0x0028
转换结果寄存器 08	ADREG08	0x0030
转换结果寄存器 19	ADREG19	0x0034
转换结果寄存器 2A	ADREG2A	0x0038
转换结果寄存器 3B	ADREG3B	0x003C
转换结果寄存器 4C	ADREG4C	0x0040
转换结果寄存器 5D	ADREG5D	0x0044
转换结果寄存器 6E	ADREG6E	0x0048
转换结果寄存器 7F	ADREG7F	0x004C
转换结果寄存器 SP	ADREGSP	0x0050
转换结果比较寄存器 0	ADCMP0	0x0054
转换结果比较寄存器 1	ADCMP1	0x0058

注:禁止存取“保留“地址。

22.3.2 ADCBAS (转换精度设置寄存器)

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

位	比特符号	类型	功能
31-8	−	R	读作 “0“。
7-0	ADCBAS[7:0]	R/W	写作 “0x58“。

注:要保证转换精度，必须给ADCBAS寄存器设置规定值 (0x0000_0058)。

22.3.3 ADCLK (转换时钟设置寄存器)

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

位	比特符号	类型	功能
31-8	−	R	读作 “0“。
7-4	TSH[3:0]	R/W	选择AD采样保持时间。 1000: 8 转换时钟 1001: 16 转换时钟 1010: 24 转换时钟 1011: 32 转换时钟 0011: 64 转换时钟 1100: 128 转换时钟 1101: 512 转换时钟 其它 : 保留
3	−	R	读作 “0“。
2-0	ADCLK[2:0]	R/W	选择AD转换时钟 000: fc 001: fc/2 010: fc/4 011: fc/8 100: fc/16 其它:保留

转换需要的时钟数最低为46个时钟。

样本保持时间与转换时间的示例如以下所示。

(示例: 如果 fc = 64MHz)

<TSH[3:0]>	采样保持时间	转换时间 (<ADCLK[2:0]>设置)				
		000 (fc)	001 (fc/2)	010 (fc/4)	011 (fc/8)	100 (fc/16)
1000 (8 转换时钟)	0.125 µs	保留	1.44µs	2.88µs	5.75µs	11.5µs
1001 (16 转换时钟)	0.25 µs	保留	1.69µs	3.38µs	6.75µs	13.5µs
1010 (24 转换时钟)	0.375 µs	保留	1.94µs	3.88µs	7.75µs	15.5µs
1011 (32 转换时钟)	0.5 µs	保留	2.19µs	4.38µs	8.75µs	17.5µs
0011 (64 转换时钟)	1.0 µs	保留	3.19µs	6.38µs	12.75µs	25.5µs
1100 (128 转换时钟)	2.0 µs	保留	5.19µs	10.38µs	20.75µs	41.5µs
1101 (512 转换时钟)	8.0 µs	保留	17.19µs	34.38µs	68.75µs	137.5µs

(示例 ： 如果 fc = 40MHz)

<TSH[3:0]>	采样保持时间	转换时间 (<ADCLK[2:0]>设置)				
		000 (fc)	001 (fc/2)	010 (fc/4)	011 (fc/8)	100 (fc/16)
1000 (8 转换时钟)	0.2 μs	1.15μs	2.3μs	4.6μs	9.2μs	18.4μs
1001 (16 转换时钟)	0.4 μs	1.35μs	2.7μs	5.4μs	10.8μs	21.6μs
1010 (24 转换时钟)	0.6 μs	1.55μs	3.1μs	6.2μs	12.4μs	24.8μs
1011 (32 转换时钟)	0.8 μs	1.75μs	3.5μs	7.0μs	14.0μs	28.0μs
0011 (64 转换时钟)	1.6 μs	2.55μs	5.1μs	10.2μs	20.4μs	40.8μs
1100 (128 转换时钟)	3.2 μs	4.15μs	8.3μs	16.6μs	33.2μs	66.4μs
1101 (512 转换时钟)	12.8 μs	13.75μs	27.5μs	55.0μs	110.0μs	220.0μs

注1:在AD转换期间，不要更改AD转换时钟的设定。

注2:请在ADCLK≤40MHz的频率时使用。在fosc为8MHz，并且PLL为8次时，fc则为64MHz，在这种情况下，规定将ADCLK <ADCLK>= “000“除外。

22.3.4 ADMOD0 (模式控制寄存器 0)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	EOCFN	ADBFN	-	ITM		REPEAT	SCAN	ADS
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-8	−	R	读作 “0“。
7	EOCFN	R	正常AD转换完成标志 (注1) 0: 在转换之前或在转换期间 1: 完成
6	ADBFN	R	正常AD转换忙标志 0: 转换停止 1: 转换期间
5	−	R	读作 “0“。
4-3	ITM[1:0]	R/W	规定固定通道重复转换模式中的中断 (参阅下表与注2)
2	REPEAT	R/W	规定重复模式 0: 单转换模式 1: 重复转换模式
1	SCAN	R/W	规定扫描模式 0: 固定通道模式 1: 通道扫描模式
0	ADS	R/W	开始AD转换启动 (注3) 0: 不指定 1: 开始转换 始终读取为 “0“。

规定固定通道重复转换模式中的AD转换中断

<ITM[1:0]>	固定通道重复转换模式 <SCAN> = “0“, <REPEAT> = “1“
00	每一个转换生成一次中断。
01	每4个转换生成一次中断。
10	每8个转换生成一次中断。
11	禁止设置

注1: 该位为 “0“，读取时清除。

注2: 仅在固定通道重复模式 (<REPEAT> = “1“, <SCAN> = “0“)中规定时有效

注3: 必须在设置该模式后进行转换。

注4: 在利用AD转换完成中断执行直接存储器存取 (DMA)转输时，首先进行软件复位。
然后启动DMA运行 (DMA请求等待模式)，并启动ADC设置。

22.3.5 ADMOD1 (模式控制寄存器 1)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	VREFON	I2AD	ADSCN	-	ADCH			
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-8	–	R	读作 “0“。
7	VREFON	R/W	VREF申请控制 (注1与注2) 0: OFF 1: ON
6	I2AD	R/W	规定IDLE模式中的运行模式 0:STOP 1:操作
5	ADSCN	R/W	规定通道扫描模式中的运行模式 0: 4 通道扫描 1: 8 通道扫描
4	–	R/W	写作 “0“。
3-0	ADCH[3:0]	R/W	选择模拟输入通道 (参阅下表。)

选择模拟输入通道

ADMOD0<SCAN> ADMOD1<ADCH[3:0]>	0 固定通道	1 通道扫描 (<ADSCN> = 0)	1 通道扫描 (<ADSCN> = 1)
0000	AIN0	AIN0	AIN0
0001	AIN1	AIN0 ~ AIN1	AIN0 ~ AIN1
0010	AIN2	AIN0 ~ AIN2	AIN0 ~ AIN2
0011	AIN3	AIN0 ~ AIN3	AIN0 ~ AIN3
0100	AIN4	AIN4	AIN0 ~ AIN4
0101	AIN5	AIN4 ~ AIN5	AIN0 ~ AIN5
0110	AIN6	AIN4 ~ AIN6	AIN0 ~ AIN6
0111	AIN7	AIN4 ~ AIN7	AIN0 ~ AIN7
1000	AIN8	AIN8	AIN8
1001	AIN9	AIN8 ~ AIN9	AIN8 ~ AIN9
1010	AIN10	AIN8 ~ AIN10	AIN8 ~ AIN10
1011	AIN11	AIN8 ~ AIN11	AIN8 ~ AIN11
1100	AIN12	AIN12	AIN8 ~ AIN12
1101	AIN13	AIN12 ~ AIN13	AIN8 ~ AIN13
1110	AIN14	AIN12 ~ AIN14	AIN8 ~ AIN14
1111	AIN15	AIN12 ~ AIN15	AIN8 ~ AIN15

注 1: 在开始AD转换之前，写入 “1“到 <VREFON>位，等待3μs时间，其间，内部基准电压应稳定下来，然后写入 “1“ 到 ADMOD0<ADS>。

注 2: 将<VREFON>设置为 “0“，一经AD转换完成即进入待机模式。

22.3.6 ADMOD2 (模式控制寄存器 2)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	EOCFHP	ADBFHP	HPADCE	-	HPADCH			
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-8	−	R	读作 “0“。
7	EOCFHP	R	最优先级AD转换完成标志 (注1) 0: 在转换之前或在转换期间 1: 完成
6	ADBFHP	R	最优先级AD转换BUSY标志 0: 转换停止期间 1: 转换期间
5	HPADCE	R/W	激活最优先级转换 0: 不指定 1: 开始转换 始终读取 “0“。
4	−	R/W	写作 “0“。
3-0	HPADCH[3:0]	R/W	激活最优先级转换时选择模拟输入通道。 (见下表)

<HPADCH[3:0]>	执行最优先级转换时选择模拟输入通道
0000	AIN0
0001	AIN1
0010	AIN2
0011	AIN3
0100	AIN4
0101	AIN5
0110	AIN6
0111	AIN7
1000	AIN8
1001	AIN9
1010	AIN10
1011	AIN11
1100	AIN12
1101	AIN13
1110	AIN14
1111	AIN15

注1:该位为 “0 “，读取时清除。

注 2: 选择通道后规定<HDADCE>。

22.3.7 ADMOD3 (模式控制寄存器3)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	-	-	ADOBIC0	ADREGS0				ADOBSV0
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-8	−	R	读作 “0“。
7	−	R/W	写作 “0“。
6	−	R	读作 “0“。
5	ADOBIC0	R/W	设置AD监控程序功能中断 0 0：如果转换结果值小于比较寄存器 0，则生成中断。 1：如果转换结果值大于比较寄存器 0，则生成中断。
4-1	ADREGS0[3:0]	R/W	使用AD监控程序功能0时选择目标转换结果寄存器 (见下表)。
0	ADOBSV0	R/W	模拟数字监控程序功能 0 0：禁用 1：启用

<ADREGS0[3:0]>	待比较的转换结果寄存器	<ADREGS0[3:0]>	待比较的转换结果寄存器
0000	ADREG08	0100	ADREG4C
0001	ADREG19	0101	ADREG5D
0010	ADREG2A	0110	ADREG6E
0011	ADREG3B	0111	ADREG7F
-	-	1xxx	ADREGSP

22.3.8 ADMOD4 (模式控制寄存器 4)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	HADHS	HADHTG	ADHS	ADHTG	-	-	ADRST	
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-8	−	R	读作 “0“。
7	HADHS	R/W	激活最优先级AD转换的H/W源 0: 外触发器 1: 与计时器寄存器 0 (TB5RG0)相匹配
6	HADHTG	R/W	激活最优先级AD转换的H/W 0: 禁用 1: 启用
5	ADHS	R/W	激活正常AD转换的H/W源 (注1) 0: 外触发器 1: 与计时器寄存器 (TB6RG0)相匹配
4	ADHTG	R/W	激活正常AD转换的H/W 0: 禁用 1: 启用
3-2	−	R	读作 “0“。
1-0	ADRST[1:0]	W	用 “01“覆盖 “10“则允许ADC进行软件复位。 (注2)

注 1: 在外触发器用于最优先级AD转换的HW激活时，该外触发器则不能用于AD转换的H/W激活。

注 2: 软件复位使所有寄存器初始化，但ADCLK<ADCLK>之外。

注 3: 这使用于H/W激活的外触发器禁用。因此，不能将 “0“设置为<HADHS>与<ADHS>。

22.3.9 ADMOD5 (模式控制寄存器 5)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	-	-	ADOBIC1	ADREGS1				ADOBSV1
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-6	–	R	读作 “0“。
5	ADOBIC1	R/W	设置AD监控程序功能中断 1。 0: 如果转换结果值小于比较寄存器 1，则生成中断。 1: 如果转换结果值大于比较寄存器 1，则生成中断。
4-1	ADREGS1[3:0]	R/W	使用AD监控程序功能1时选择目标转换结果寄存器 (见下表)。
0	ADOBSV1	R/W	模拟数字监控功能 1 0: 禁用 1: 启用

<ADREGS1[3:0]>	待比较的转换结果寄存器	<ADREGS1[3:0]>	待比较的转换结果寄存器
0000	ADREG08	0100	ADREG4C
0001	ADREG19	0101	ADREG5D
0010	ADREG2A	0110	ADREG6E
0011	ADREG3B	0111	ADREG7F
–	–	1xxx	ADREGSP

22.3.10 ADREG08 (转换结果寄存器 08)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	ADR0							
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	ADR0		-	-	-	-	OVR0	ADR0RF
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-16	−	R	读作 “0“。
15-6	ADR0[9:0]	R	AD转换结果 转换结果已储存。有关转换通道与转换结果寄存器之间的相关性信息，见22.4.5.7中的表22-2。
5-2	−	R	读作 “0“。
1	OVR0	R	超载标志 0: 不生成 1: 生成 如果转换结果在读取<ADR0>时被覆盖，则设置为 “1“。 该位为 “0“，读取时清除。
0	ADR0RF	R	AD转换结果储存标志 0: 转换结果未储存 1: 转换结果已储存。 如果转换结果已储存，则设置为 “1“。 该位为 “0“，在转换结果读取时清除。

注:该寄存器的存取必须以半字或一字存取。

22.3.11 ADREG19 (AD转换结果寄存器 19)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	ADR1							
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	ADR1		-	-	-	-	OVR1	ADR1RF
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-16	−	R	读作 “0“。
15-6	ADR1[9:0]	R	AD转换结果 转换结果已储存。有关转换通道与转换结果寄存器之间的相关性信息，见22.4.5.7中的表22-2。
5-2	−	R	读作 “0“。
1	OVR1	R	超载标志 0: 不生成 1: 生成 如果转换结果在读取<ADR1>时被覆盖，则设置为 “1“。 该位为 “0“，读取时清除。
0	ADR1RF	R	AD转换结果储存标志 0: 转换结果未储存 1: 转换结果已储存。 如果转换结果已储存，则设置为 “1“。 该位为 “0“，在转换结果读取时清除。

注:该寄存器的存取必须以半字或一字存取。

22.3.12 ADREG2A (AD转换结果寄存器 2A)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	ADR2							
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	ADR2		-	-	-	-	OVR2	ADR2RF
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-16	−	R	读作 “0“。
15-6	ADR2[9:0]	R	AD转换结果 转换结果已储存。有关转换通道与转换结果寄存器之间的相关性信息，见22.4.5.7中的表22-2。
5-2	−	R	读作 “0“。
1	OVR2	R	超载标志 0: 不生成 1: 生成 如果转换结果在读取<ADR2>时被覆盖，则设置为 “1“。 该位为 “0“，读取时清除。
0	ADR2RF	R	AD转换结果储存标志 0: 转换结果未储存 1: 转换结果已储存。 如果转换结果已储存，则设置为 “1“。 该位为 “0“，在转换结果读取时清除。

注:该寄存器的存取必须以半字或一字存取。

22.3.13 ADREG3B (AD转换结果寄存器 3B)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	ADR3							
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	ADR3		-	-	-	-	OVR3	ADR3RF
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-16	−	R	读作 “0“。
15-6	ADR3[9:0]	R	AD转换结果 转换结果已储存。有关转换通道与转换结果寄存器之间的相关性信息，见22.4.5.7中的表22-2。
5-2	−	R	读作 “0“。
1	OVR3	R	超载标志 0: 不生成 1: 生成 如果转换结果在读取<ADR3>时被覆盖，则设置为 “1“。 该位为 “0“，读取时清除。
0	ADR3RF	R	AD转换结果储存标志 0: 转换结果未储存 1: 转换结果已储存。 如果转换结果已储存，则设置为 “1“。 该位为 “0“，在转换结果读取时清除。

注:该寄存器的存取必须以半字或一字存取。

22.3.14 ADREG4C (AD转换结果寄存器 4C)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	ADR4							
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	ADR4		-	-	-	-	OVR4	ADR4RF
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-16	−	R	读作 “0“。
15-6	ADR4[9:0]	R	AD转换结果 转换结果已储存。有关转换通道与转换结果寄存器之间的相关性信息，见22.4.5.7中的表22-2。
5-2	−	R	读作 “0“。
1	OVR4	R	超载标志 0: 不生成 1: 生成 如果转换结果在读取<ADR4>时被覆盖，则设置为 “1“。 该位为 “0“，读取时清除。
0	ADR4RF	R	AD转换结果储存标志 0: 转换结果未储存 1: 转换结果已储存。 如果转换结果已储存，则设置为 “1“。 该位为 “0“，在转换结果读取时清除。

注:该寄存器的存取必须以半字或一字存取。

22.3.15 ADREG5D (AD转换结果寄存器 5D)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	ADR5							
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	ADR5		-	-	-	-	OVR5	ADR5RF
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-16	−	R	读作 “0“。
15-6	ADR5[9:0]	R	AD转换结果 转换结果已储存。有关转换通道与转换结果寄存器之间的相关性信息，见22.4.5.7中的表22-2。
5-2	−	R	读作 “0“。
1	OVR5	R	超载标志 0:不生成 1:生成 如果转换结果在读取<ADR5>时被覆盖，则设置为 “1 “。 该位为 “0“，读取时清除。
0	ADR5RF	R	AD转换结果储存标志 0:转换结果未储存 1:转换结果已储存。 如果转换结果已储存，则设置为 “1“。 该位为 “0“，在转换结果读取时清除。

注:该寄存器的存取必须以半字或一字存取。

22.3.16 ADREG6E (AD转换结果寄存器 6E)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	ADR6							
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	ADR6		-	-	-	-	OVR6	ADR6RF
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-16	−	R	读作 “0“。
15-6	ADR6[9:0]	R	AD转换结果 转换结果已储存。有关转换通道与转换结果寄存器之间的相关性信息，见22.4.5.7中的表22-2。
5-2	−	R	读作 “0“。
1	OVR6	R	超载标志 0: 不生成 1: 生成 如果转换结果在读取<ADR6>时被覆盖，则设置为 “1 “。 该位为 “0 “，读取时清除。
0	ADR6RF	R	AD转换结果储存标志 0: 转换结果未储存 1: 转换结果已储存。 如果转换结果已储存，则设置为 “1“。 该位为 “0“，在转换结果读取时清除。

注:该寄存器的存取必须以半字或一字存取。

22.3.17 ADREG7F (AD转换结果寄存器 7F)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	ADR7							
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	ADR7		-	-	-	-	OVR7	ADR7RF
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-16	−	R	读作 “0“。
15-6	ADR7[9:0]	R	AD转换结果 转换结果已储存。有关转换通道与转换结果寄存器之间的相关性信息，见22.4.5.7中的表22-2。
5-2	−	R	读作 “0“。
1	OVR7	R	超载标志 0: 不生成 1: 生成 如果转换结果在读取<ADR7>时被覆盖，则设置为 “1“。 该位为 “0“，读取时清除。
0	ADR7RF	R	AD转换结果储存标志 0: 转换结果未储存 1: 转换结果已储存。 如果转换结果已储存，则设置为 “1“。 该位为 “0“，在转换结果读取时清除。

注:该寄存器的存取必须以半字或一字存取。

22.3.18 ADREGSP (AD转换结果寄存器SP)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	ADRSP							
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	ADRSP		-	-	-	-	OVRSP	ADRSPRF
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-16	−	R	读作 “0“。
15-6	ADRSP[9:0]	R	AD转换结果 最优先级AD转换结果已储存。
5-2	−	R	读作 “0“。
1	OVRSP	R	超载标志 0: 不生成 1: 生成 如果转换结果在读取<ADRSP>时被覆盖，则设置为 “1“。 该位为 “0“，读取时清除。
0	ADRSPRF	R	AD转换结果储存标志 0: 转换结果未储存 1: 转换结果已储存。 如果转换结果已储存，则设置为 “1“。 该位为 “0“，在转换结果读取时清除。

注:该寄存器的存取必须以半字或一字存取。

22.3.19 ADCMP0 (AD转换结果比较寄存器 0)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	ADCOMO							
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	ADCOM0		-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-16	−	R	读作 “0“。
15-6	ADCOM0[9:0]	R/W	在AD监控程序功能0启用时，即设置一个值，用该值与ADMOD3<ADREGS0>规定的转换结果寄存器的值进行比较。
5-0	−	R	读作 “0“。

注:要将这些值写入该寄存器，必须禁用模拟数字监控程序功能0 (ADMOD3<ADOBSV0>= “0“)。

22.3.20 ADCMP1 (AD转换结果比较寄存器1)

	31		30	29	28		27	26	25	24
比特符号	-	-	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0	0	0
	23		22	21	20		19	18	17	16
比特符号	-	-	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0	0	0
	15		14	13	12		11	10	9	8
比特符号	ADCOM1									
复位后	0	0	0	0	0	0	0	0	0	0
	7		6	5	4		3	2	1	0
比特符号	ADCOM1		-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-16	−	R	读作 “0“。
15-6	ADCOM1[9:0]	R/W	在AD监控程序功能0启用时，即设置一个值，用该值与ADMOD5<ADREGS1>规定的转换结果寄存器的值进行比较。
5-0	−	R	读作 “0“。

注:要将这些值写入该寄存器，必须禁用AD监控程序功能1 (ADMOD5<ADOBSV1> = “0“)。

22.4 操作说明

22.4.1 模拟基准电压

模拟基准电压的 “高“电平应施加到VRFEH引脚，而 “低“电平应施加到VREFL引脚。

要开始进行AD转换，确信首先将 “1“写入<VREFON>位，等待3 μs时间，其间，内基准电压应稳定下来，然后将 “1“写入ADMOD0<ADS>位。

将 “0“写入到ADMOD1<VREFON>位，VREFH-VREFL的已接通状态可转化为已断开状态。要切换到功率消耗模式，则在转换之后将 “0“设置为<VREFON>位。

注:VREFL与AVSS由TMPM364F10FG共享。

22.4.2 AD转换模式

有了两类AD转换得到支持:正常AD转换与最优先级AD转换。

对于正常AD转换，则支持以下四种运行模式。

22.4.2.1 正常AD转换

对于正常AD转换，支持以下四种运行模式，并且采用ADMOD0<REPEAT,SCAN>选择运行模式。

- 固定通道单个转换模式
- 通道扫描单个转换模式
- 固定通道重复转换模式
- 通道扫描重复转换模式

(1) 固定通道单个转换模式

如果ADMOD0<REPEAT, SCAN>被设置到“00“，那么，AD转换则以固定通道单个转换模式完成。

在该模式中，每个选择的通道执行一次AD转换。在AD转换完成后，ADMOD0<EOCFN>则被设置到 “1“，ADMOD0<ADBFN>则被清除为 “0“，随即生成AD转换完成中断请求(INTAD)。<EOCFN>一旦被读取则清除为 “0“。

(2) 通道扫描单个转换模式

如果ADMOD0<REPEAT, SCAN>被设置到 “01“，那么，AD转换则以通道扫描单个转换模式执行。

在该模式中，每个选择的扫描通道执行一次AD转换。在AD扫描转换完成后，ADMOD0<EOCFN>则被设置到 “1“，ADMOD0<ADBFN>则被清除为 “0“，随即生成AD转换完成中断请求(INTAD)。<EOCFN>则被清除为 “0“。

(3) 固定通道重复转换模式

如果ADMOD0<REPEAT, SCAN>被设置为“10“，那么，AD转换以固定通道重复转换模式执行。

在该模式中，一个选择的通道重复地执行AD转换。在AD转换完成后，ADMOD0<EOCFN>被设置为 “1“。而ADMOD0<ADBFN>不清除为 “0“。仍保留在 “1“。生成转换完成中断请求(INTAD)的时间可用过将ADMOD0<ITM>设置到一个合适的设定值来选取。<EOCFN>采用与生成本中断INTAD相同的时间来设置。

读取<EOCFN>即清除为 “0“。

(4) 通道扫描重复转换模式

如果ADMOD0<REPEAT，SCAN>被设置为 “11“，那么，AD转换则以通道扫描重复转换模式执行。

在该模式中，一个选择的扫描通道重复地执行AD转换。每次完成一个AD扫描转换，ADMOD0<EOCFN>被设置为 “1“，并生成转换完成中断请求(INTAD)。而ADMOD0<ADBFN>不清除为 “0“。仍保留在 “1“。<EOCFN>一旦被读取则清除为 “0“。

22.4.2.2 最优先级AD转换

中断进行中的正常AD转换，即可执行最优先级AD转换。

固定通道单个转换为自动选择，而不理会ADMOD0<REPEAT，SCAN>设定。在满足开始运行的条件时，由ADMOD2<HPADCH>指定的通道仅执行一次转换。转换完成时，生成最优先级AD转换完成中断 (INTADHP)，显示AD转换完成的ADMOD2<EOCFHP>则被设置为 “1“。<ADBFHP>则返回到 “0“。一经读取，EOCFHP标志即清除为 “0“。

对最优先级AD转换运行时激活的最优先级AD转换，则不予理会。

注:最优先级A/D转换中断不能生成直接存储器存取(DMA)传送请求。要生成DMA传送请求，则请使用A/D转换完成中断(INTAD)。

22.4.3 AD监控程序功能

有两条AD监控程序功能通道。

如果将ADMOD3<ADOBSV0>与ADMOD5<ADOBSV1>设置为 “1“，则AD监控程序功能被启用。如果ADMOD3<ADREGS0>与ADMOD5<ADREGS1>规定的转换结果寄存器的值大于或小于比较寄存器的值 (“更大“ 或 “更小“由ADMOD3<ADOBIC0>与ADMOD5<ADBIC1>分配)，即生成该AD监控程序功能中断 (INTADM0，INTADM1)。每一次均执行这种比较操作，结果储存在相应的转换结果寄存器中。

如果分配执行AD监控程序功能的转换结果寄存器连续使用而不读取转换结果，其转换结果则被覆盖。转换结果储存标志<ADRxRF>与超载标志<OVRx>保持待设定。

22.4.4 选择输入通道

复位后，ADMOD0<REPEAT，SCAN>被初始化到 “00“，ADMOD1<ADCH[3:0]>则被初始化为 “0000“。

待转换的通道根据AD转换器运行模式来选择，如以下所示。

1. 正常AD转换模式

- 如果模拟输入通道用于固定状态 (ADMOD0<SCAN>= “0“)

  则通过将ADMOD1<ADCH>设置到一个适合的设置值，从模拟输入引脚AIN0 ~ AIN15中选取一个通道。

- 如果模拟输入通道用于扫描状态 (ADMOD0<SCAN>= “1“)

  则将ADMOD1<ADCH>与ADSCN设置为一个适合的设置值，从众多扫描模式中选取一个扫描模式。

2. 最优先级AD转换模式

将ADMOD2<HPADCH>设置到一个适合的设置值，从AIN0 ~ AIN15的模拟输入引脚中选取一个通道。

22.4.5 AD转换详细情况

22.4.5.1 开始模拟AD转换

将ADMOD0<ADS>设置到 “1“来激活正常AD转换。将ADMOD2<HPADCE>设置为 “1“来激活最优先级AD转换。

正常AD转换有四个运行模式可用。在执行正常AD转换中，必须将ADMOD0<REPEAT，SCAN>设置为一个适合的设置值来从中选取一个运行模式。对于最优先级AD转换，仅可使用一个运行模式:固定通道单个转换模式。

使用由ADMOD4<ADHS>选择的H/W激活源即可激活正常AD转换，而最优先级AD转换则可使用由ADMOD4<HADHS>选择的H/W激活源来激活。如果<ADHS>与<HADHS>的位为 “0“，则正常与最优先级AD转换根据通过ADTRG引脚的下降沿的输入来激活。如果这些位为 “1“，则正常AD转换根据由16-位计时器6生成的TB6RG0来激活，而最优先级AD转换则根据由16-位计时器5生成的TB5RG0来激活。

为了H/W能够激活，将ADMOD4<ADHTG>设置为 “1“进行正常AD转换，将ADMOD4<HADHTG>设置为 “1“进行最优先级AD转换。

即便在H/W已能够激活之后，软件激活仍然有效。

注 1: 在外触发器用于最优先级转换的HW激活源时，则不能设置外触发器来激活正常AD转换H/W。

注 2: TMPM364F10FG禁用用于H/W激活的外触发器。因此，不能将 “0“设置为<HADHS>与<ADHS>。

22.4.5.2 AD转换

在正常AD转换开始时，显示AD转换正在进行的AD转换忙标志 (ADMOD0<ADBFN>)被设置为“1“。

在最优先级AD转换开始时，显示AD转换正在进行的最优先级AD转换忙标志 (ADMOD2<ADBFHP>)被设置为 “1“。在那时，最优先级AD转换开始之前，正常AD转换的忙标志ADMOD0<ADBFN>的值得以保留，在最优先级AD转换开始之前，正常AD转换的转换完成标志ADMOD0<EOCFN>的值得以保留。

注:当最优先级AD转换正在进行时，不可激活正常AD转换。

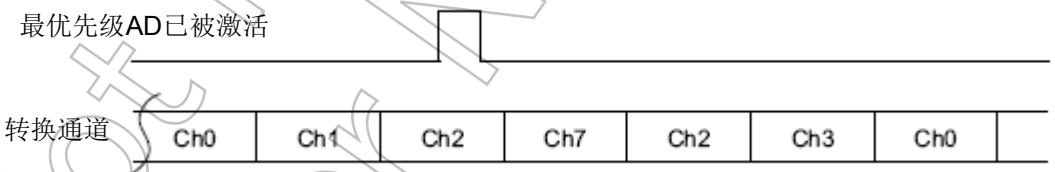
22.4.5.3 正常AD转换期间的最优先级AD转换

如果最优先级AD转换已在正常AD转换期间被激活，那么进行中的正常AD转换则被挂起，最优先级AD转换完成后重新开始正常AD转换。

如果ADMOD2<HPADCE>在正常AD转换期间被设置为 “1“，进行中的正常AD转换则被挂起，最优先级AD转换开始；具体地说，就是为ADMOD2<HPADCH>分派的通道执行AD转换 (固定通道单个转换)。在该最优先级AD转换的结果储存于存储寄存器ADREGSP中之后，正常AD转换再继续进行。

如果最优先级AD转换的H/W激活在正常AD转换期间得到许可，进行中的AD转换则在使用H/W激活源的激活要求满足时中止，则开始为ADMOD2<HPADCH>分派的通道进行最优先级AD转换 (固定通道单个转换)。在该最优先级AD转换的结果储存于存储寄存器ADREGSP中之后，正常AD转换再继续进行

举例来说，如果为AIN0 ~ AIN3的通道实行了通道重复转换激活，如果<HPADCE>在AIN2转换期间设置为 “1“，则AIN2转换被挂起，并且对由<HPADCH> (如下所示的情况下为AIN7)分派的通道执行转换。转换结果储存于ADREGSP中后，从AIN2开始，再继续进行通道重复转换。



22.4.5.4 停止重复转换模式

要在重复转换模式 (固定通道重复转换模式或通道扫描重复转换模式)中停止AD转换操作，将“0“写为ADMOD0<REPEAT>。当进行中的AD转换完成时，重复转换模式终止，ADMOD0<ADBFN>被设置为 “0“。

22.4.5.5 激活正常AD转换

如要在转换正在进行中时重新激活正常AD转换，则必须在启动AD转换之前执行软件复位(ADMOD3<ADRST>)。H/W激活法不可用于重新激活正常AD转换。

22.4.5.6 转换完成

(1) 正常AD转换完成

在正常AD转换完成时即生成AD转换完成中断 (INTAD)。AD转换的结果储存于存储寄存器中，且两个寄存器变化:表示AD转换完成的寄存器ADMOD0<EOCFN>与寄存器ADMOD0<ADBFN>。中断请求，转换寄存器存储寄存器与<EOCFN><ADBFN>根据所选择的模式均随不同的时间而变化。

在除了固定通道重复转换模式之外的模式中，转换结果储存于相当于一个通道的AD转换结果寄存器 (ADREG08 ~ ADRG7F)中。

在固定通道重复转换模式中，转换结果依序储存于存储寄存器ADREG08 ~ ADREG7F中。但是，如果对<ITM>设置的中断被设置到每次一个AD转换完成时就生成，那么，该转换结果就仅储存于ADREG08中。如果对<ITM>设置的中断被设置到每次四个AD转换完成时就生成，那么，该转换结果就储存于ADREG08 ~ ADREG3B中。如果对<ITM>设置的中断被设置到每次八个AD转换完成时就生成，那么，转换结果就依序储存于ADREG08H ~ ADREG7F。

中断请求，标志改变与转换结果寄存器所示如下。

・ 固定通道单个转换模式

AD转换完成之后，ADMOD0<EOCFN>被设置为 “1“，ADMOD0<ADBFN>被清除为 “0“，中断请求随即生成。
转换结果储存于与某个通道对应的转换结果寄存器中。

・ 通道扫描单个转换模式

通道扫描转换完成之后，ADMOD0<EOCFN>被设置为 “1“，ADMOD0<ADBFN>被设置为“0“，中断请求INTAD随即生成。
转换结果储存于与某个通道对应的转换结果寄存器中。

・ 固定通道重复转换模式

ADMOD0<ADBFN>不清除为 “0“。仍保留为 “1“。生成中断请求 (INTAD)的时间可用过将ADMOD0<ITM>设置到一个合适的设定值来选取。ADMOD0<EOCFN>采用与生成本中断INTAD相同的时间来设置。

a. 一个转换

将<ITM[1:0]>设置为 “00“时，每次完成一个AD转换时即生成一个中断请求。在这种情况下，转换结果始终储存在存储寄存器ADREG08中。转换结果储存之后，<EOCFN>转变为 “1“。

b. 四个转换

将<ITM[1:0]>设置为 “01“时，每次完成四个AD转换时即生成一个中断请求。在这种情况下，转换结果依序储存在存储寄存器ADREG08 ~ ADREG3B中。在转换结果储存到ADREG3B中之后，<EOCFN>则被设置为 “1“，且随后的转换结果储存即从ADREG08开始。

c. 八个转换

将<ITM[1:0]>设置为 “10“时，每次完成八个AD转换时即生成一个中断请求。在这种情况下，转换结果依序储存在存储寄存器ADREG08 ~ ADREG7F中。在转换结果储存到ADREG7F中之后，<EOCFN>则被设置为 “1“，且随后的转换结果储存即从ADREG08开始。

- 通道扫描重复转换模式

每次完成一个AD转换时，ADMOD0<EOCF>被设置为 “1“，随即生成中断请求INTAD。ADMOD0<ADBFN>不清除为 “0“。仍保留为 “1“ 。

AD转换结果储存在与某个通道对应的AD转换结果寄存器中。

(2) 最优先级AD转换完成

AD转换完成之后，生成最优先级AD转换完成中断 (INTADHP)，表明最优先级AD转换完成的ADMOD2<EOCFHP>则被设置为 “1“。

AD转换结果储存于AD转换结果寄存器SP之中。

(3) 数据轮询

如要确认是否不使用中断即可完成AD转换，可以使用数据轮询。在AD转换完成时，ADMOD0<EOCFN>被设置为 “1“。要确认AD转换是否完成，并获得结果，则轮询该位。

必须采用半字或一字存取读取AD转换结果存储寄存器。如果<OVRx>= “0“，且<ADRxRF>= “1“，即已获得正确的转换结果。

22.4.5.7 中断生成时间与AD转换结果存储寄存器

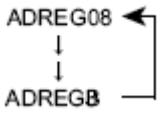
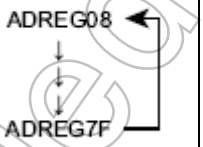
表22-1显示了以下三项的关系:AD转换模式，中断生成时间与标志操作。表22-2显示了模拟通道输入与AD转换结果寄存器之间的关系。

表22-1 转换模式，中断生成时间与标志操作中的关系

转换模式		扫描/重复方式设置 (ADMOD0)			中断生成时间。	<EOCFN>/ <EOCFHP> 设置时间 (注)	ADMOD0	ADMOD2
		<REPEAT>	<SCAN>	<ITM[1: 0]>			<ADBFN> (中断生成之后)	<ADBFHP>
正常转换	固定通道单个转换	0	0	-	生成完成之后	转换完成之后。	0	-
	固定通道重复转换	1	0	00	每次完成一个转换时。	完成一个转换之后。	1	-
				01	每次完成四个转换时。	完成四个转换之后。	1	-
				10	每次完成八个转换时。	完成八个转换之后。	1	-
	通道扫描单个转换	0	1	-	扫描转换完成之后。	扫描转换完成之后。	0	-
	通道扫描重复转换	1	1	-	完成一个扫描转换之后。	完成一个扫描转换之后。	1	-
最优先级转换		-	-	-	转换完成之后。	转换完成	-	0

注:ADMOD0<EOCFN>与ADMOD2<EOCFHP>一经读取即清除。

表22-2 模拟通道输入与AD转换结果寄存器之间的关系

模拟输入通道	正常AD转换				最优先级AD转换
	右侧所示除外的转换模式	固定通道重复转换模式(每一个转换)	固定通道重复转换模式(每四个转换)	固定通道重复转换模式(每八个转换)	
AIN0	ADREG08	ADREG08固定			ADREGSP
AIN1	ADREG19				
AIN2	ADREG2A				
AIN3	ADREG3B				
AIN4	ADREG4C				
AIN5	ADREG5D				
AIN6	ADREG6E				
AIN7	ADREG7F				
AIN8	ADREG08				
AIN9	ADREG19				
AIN10	ADREG2A				
AIN11	ADREG3B				
AIN12	ADREG4C				
AIN13	ADREG5D				
AIN14	ADREG6E				
AIN15	ADREG7F				

注:要存取转换结果寄存器，使用半字或一字存取。

注意事项
AD转换的结果值可能因电源电压的波动而不同，或可能受到噪音的影响。 交替使用模拟输入引脚与端口时，在转换期间不要读取和写入端口，这是因为转换精度可能会降低。如果AD转换期间输出端口电流波动，转换精度也会降低。 请利用程序采取应对措施，比如求AD转换结果的均值。

23. 实时时钟 (RTC)

23.1 功能

1. 时钟 (时，分与秒)
2. 日历 (月，周，日期与闰年)
3. 可选12 (am/ pm)24 小时显示
4. 时间调整 + 或 −30秒 (通过软件)
5. 报警器 (报警输出)
6. 报警中断

23.2 方块图

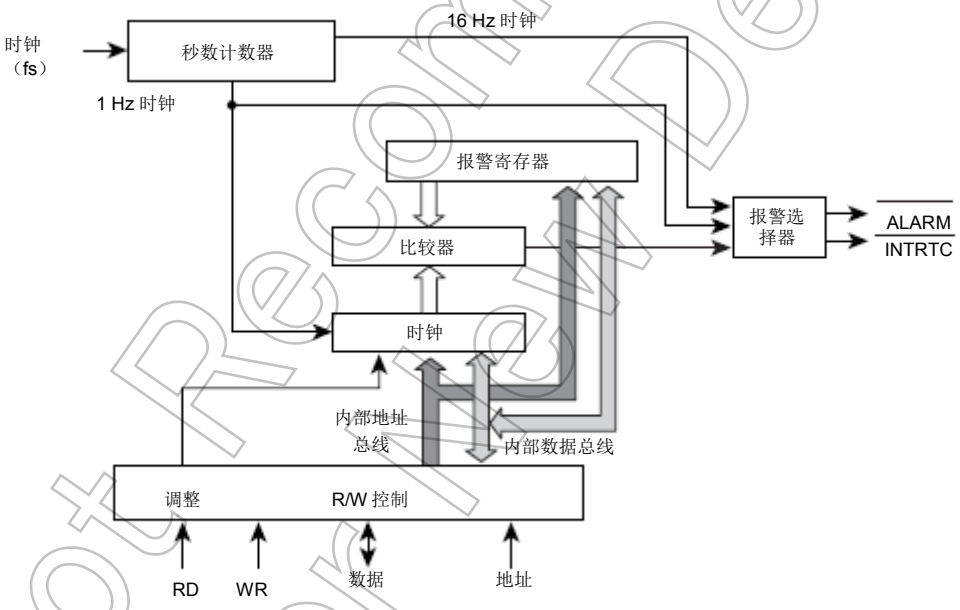


图23-1 方块图

注 1: 西历年列:本产品仅使用年份的最后二位数。99之后的年份是00年份。在处理西历年时请考虑头两个数字。

注 2: 闰年:闰年可被4除尽，不包括可被100除尽的年份；可被100除尽的年份不能认为是闰年。任何可被400除尽的年份为闰年。本产品被认为是可被4除尽的闰年，不要考虑上述例外。例外情况需求调整。

23.3 详细描述寄存器

23.3.1 寄存器列表

与实时时钟相关的寄存器与地址显示如下。

实时时钟有两个功能，PAGE0 (时钟)与PAGE1 (报警)，它们共享寄存器的某些部分。

PAGE可通过设置RTCPAGER<PAGE>来选取。

基址=0x4004_0100

寄存器名称		地址 (基址+)
秒钟列寄存器 (仅PAGE0)	RTCSECR	0x0000
分钟列寄存器	RTCMINR	0x0001
小时列寄存器	RTCHOURR	0x0002
- (注1)	-	0x0003
周日列寄存器	RTCDAYR	0x0004
日份列寄存器	RTCDATER	0x0005
月份列寄存器 (PAGE0) 24-小时，12-小时选择寄存器 (PAGE1)	RTCMONTHR	0x0006
年份列寄存器 (PAGE0) 闰年寄存器 (PAGE1)	RTCYEARR	0x0007
PAGE寄存器	RTCPAGER	0x0008
- (注1)	-	0x0009
- (注1)	-	0x000A
- (注1)	-	0x000B
复位寄存器	RTCRESTR	0x000C
保留	-	0x000D
- (注1)	-	0x000E
- (注1)	-	0x000F

注 1: “0“通过读取地址来读取。写入不理会。

注 2: 禁止访问 “保留“区域。

23.3.2 控制寄存器

复位操作使以下寄存器初始化。

- RTCPAGER<PAGE>，<ADJUST>，<INTENA>
- RTCRESTR<RSTALM>，<RSTTMR>，<DIS16HZ>，<DIS1HZ>

其它时钟相关的寄存器不由复位操作来初始化。

使用RTC之前，在有关寄存器中设置时间，月份，日份，周日，年份与闰年。

设置时钟数据，调整秒钟或复位时钟时需要格外小心。

要了解更多信息，参阅 “23.4.3 进入低功耗模式“。

表23-1 PAGE0 (时钟功能)寄存器

符号	位7	位6	位5	位4	位3	位2	位1	位0	功能
RTCSECR	-	40秒钟	20秒钟	10秒钟	8秒钟	4秒钟	2秒钟	1秒钟	秒钟列
RTCMINR	-	40分钟	20分钟	10分钟	8分钟	4分钟	2分钟	1分钟	分钟列
RTCHOURR	-	-	20小时 PM/AM	10小时	8小时	4小时	2小时	1小时	小时列
RTCDAYR	-	-	-	-	-	周日			周日列
RTCDATER	-	-	20日	10日	8日	4日	2日	1日	日份列
RTCMONTHR	-	-	-	十月	八月	四月	二月	一月	月份列
RTCYEARR	80年	40年	20年	10年	8年	4年	2年	1年	年份列 (下两列)
RTCPAGER	中断启用	-	-	调整功能	时钟启用	报警器启用	-	PAGE 设置	PAGE寄存器
RTCRESTR	1Hz 启用	16Hz 启用	时钟 复位	报警 复位	-	始终写入“1“			复位寄存器

注:读取PAGE0的RTCSECR，RTCMINR，RTCHOURR，RTCDAYR，RTCMONTHR，RTCYEARR可获取当前状态。

表23-2 PAGE1 (报警功能)寄存器

符号	位7	位6	位5	位4	位3	位2	位1	位0	功能
RTCSECR	-	-	-	-	-	-	-	-	-
RTCMINR	-	40分钟	20分钟	10分钟	8分钟	4分钟	2分钟	1分钟	分钟列
RTCHOURR	-	-	20小时 PM/AM	10小时	8小时	4小时	2小时	1小时	小时列
RTCDAYR	-	-	-	-	-	周日			周日列
RTCDATER	-	-	20日	10日	8日	4日	2日	1日	日份列
RTCMONTHR	-	-	-	-	-	-	-	24/12	24-小时时钟模式
RTCYEARR	-	-	-	-	-	-	闰年设置		闰年模式
RTCPAGER	中断启用	-	-	调整功能	时钟启用	报警器启用	-	PAGE设置	PAGE寄存器
RTCRESTR	1Hz启用	16Hz 启用	时钟复位	报警复位	-	始终写入“1“			复位寄存器

注 1: 读取PAGE1的RTCMINR,RTCHOURR,RTCDAYR,RTCMONTHR,RTCYEARR可获取当前状态。

注 2: PAGE0的RTCSECR，RTCMINR，RTCHOURR，RTCDAYR，RTCDATER，RTCMONTHR，RTCYEARR与PAGE1的RTCYEARR (闰年)必须读取两次，并比较获取的数据。

23.3.3 控制寄存器的详细描述

23.3.3.1 RTCSECR (第二列寄存器 (仅适用于PAGE0))

	7	6	5	4	3	2	1	0
比特符号	-	SE						
复位后	0	未定义	未定义	未定义	未定义	未定义	未定义	未定义

位	比特符号	类型	功能
7	–	R	读作 0。
6-0	SE	R/W	设置秒钟的位数寄存器 000_0000 : 00秒钟 001_0000 : 10秒钟 010_0000 : 20秒钟 000_0001 : 01秒钟 001_0001 : 11秒钟 ・ 000_0010 : 02秒钟 001_0010 : 12秒钟 011_0000 : 30秒钟 000_0011 : 03秒钟 001_0011 : 13秒钟 ・ 000_0100 : 04秒钟 001_0100 : 14秒钟 100_0000 : 40秒钟 000_0101 : 05秒钟 001_0101 : 15秒钟 ・ 000_0110 : 06秒钟 001_0110 : 16秒钟 101_0000 : 50秒钟 000_0111 : 07秒钟 001_0111 : 17秒钟 ・ 000_1000 : 08秒钟 001_1000 : 18秒钟 ・ 000_1001 : 09秒钟 001_1001 : 19秒钟 101_1001 : 59秒钟

注:禁止除上列以外的设置。

23.3.3.2 RTCMINR (分钟列寄存器 (PAGE0/1))

	7	6	5	4	3	2	1	0
比特符号	-	MI						
复位后	0	未定义	未定义	未定义	未定义	未定义	未定义	未定义

位	比特符号	类型	功能
7	–	R	读作0。
6-0	MI	R/W	设置分钟的位数寄存器。 000_0000 : 00分钟 001_0000 : 10分钟 010_0000 : 20分钟 000_0001 : 01分钟 001_0001 : 11分钟 ・ 000_0010 : 02分钟 001_0010 : 12分钟 011_0000 : 30分钟 000_0011 : 03分钟 001_0011 : 13分钟 ・ 000_0100 : 04分钟 001_0100 : 14分钟 100_0000 : 40分钟 000_0101 : 05分钟 001_0101 : 15分钟 ・ 000_0110 : 06分钟 001_0110 : 16分钟 101_0000 : 50分钟 000_0111 : 07分钟 001_0111 : 17分钟 ・ 000_1000 : 08分钟 001_1000 : 18分钟 ・ 000_1001 : 09分钟 001_1001 : 19分钟 101_1001 : 59分钟

注:禁止除上列以外的设置。

23.3.3.3 RTCHOURR (小时列寄存器 (PAGE0/1))

(1) 24小时时钟模式 (RTCMONTHR<MO0>= “1“)

	7	6	5	4	3	2	1	0
比特符号	-	-	HO					
复位后	0	0	未定义	未定义	未定义	未定义	未定义	未定义

位	比特符号	类型	功能
7-6	–	R	读作0。
5-0	HO	R/W	设置小时的位数寄存器. 00_0000 : 0 点钟 01_0000 : 10 点钟 10_0000 : 20 点钟 00_0001 : 1 点钟 01_0001 : 11 点钟 10_0001 : 21 点钟 00_0010 : 2 点钟 01_0010 : 12 点钟 10_0010 : 22 点钟 00_0011 : 3 点钟 01_0011 : 13 点钟 10_0011 : 23 点钟 00_0100 : 4 点钟 01_0100 : 14 点钟 00_0101 : 5 点钟 01_0101 : 15 点钟 00_0110 : 6 点钟 01_0110 : 16 点钟 00_0111 : 7 点钟 01_0111 : 17 点钟 00_1000 : 8 点钟 01_1000 : 18 点钟 00_1001 : 9 点钟 01_1001 : 19 点钟

注:禁止除上列以外的设置。

(2) 12-小时时钟模式 (RTCMONTHR<MO0> = “0“)

	7	6	5	4	3	2	1	0
比特符号	-	-	HO					
复位后	0	0	未定义	未定义	未定义	未定义	未定义	未定义

位	比特符号	类型	Functon
7-6	–	R	读作 0。
5-0	HO	R/W	设置小时的位数寄存器 (AM) (PM) 00_0000 : 0点钟 10_0000 : 0点钟 00_0001 : 1点钟 10_0001 : 1点钟 00_0010 : 2点钟 10_0010 : 2点钟 00_0011 : 3点钟 10_0011 : 3点钟 00_0100 : 4点钟 10_0100 : 4点钟 00_0101 : 5点钟 10_0101 : 5点钟 00_0110 : 6点钟 10_0110 : 6点钟 00_0111 : 7点钟 10_0111 : 7点钟 00_1000 : 8点钟 10_1000 : 8点钟 00_1001 : 9点钟 10_1001 : 9点钟 01_0000 : 10点钟 11_0000 : 10点钟 01_0001 : 11点钟 11_0001 : 11点钟

注:禁止除上列以外的设置。

23.3.3.4 RTCDAYR (周日列寄存器 (PAGE0/1))

	7	6	5	4	3	2	1	0
比特符号	-	-	-	-	-	WE		
复位后	0	0	0	0	0	未定义	未定义	未定义

位	比特符号	类型	功能
7-3	–	R	读作 0。
2-0	WE	R/W	设置周日的位数寄存器。 000: 星期日 001: 星期一 010: 星期二 011 :星期三 100: 星期四 101: 星期五 110: 星期六

注:禁止除上列以外的设置。

23.3.3.5 RTCDATER (日列寄存器 (仅适用于PAGE0/1))

	7	6	5	4	3	2	1	0
比特符号	-	-	DA					
复位后	0	0	未定义	未定义	未定义	未定义	未定义	未定义

位	比特符号	类型	功能			
7-6	–	R	读作 0。			
5-0	DA	R/W	设置日的位数寄存器。			
				01_0000：第10日	10_0000：第20日	11_0000：第30日
			00_0001：第1日	01_0001：第11日	10_0001：第21日	11_0001：第31日
			00_0010：第2日	01_0010：第12日	10_0010：第22日	
			00_0011：第3日	01_0011：第13日	10_0011：第23日	
			00_0100：第4日	01_0100：第14日	10_0100：第24日	
			00_0101：第5日	01_0101：第15日	10_0101：第25日	
			00_0110：第6日	01_0110：第16日	10_0110：第26日	
			00_0111：第7日	01_0111：第17日	10_0111：第27日	
			00_1000：第8日	01_1000：第18日	10_1000：第28日	
			00_1001：第9日	01_1001：第19日	10_1001：第29日	

注 1: 禁止除上列以外的设置。

注 2: 不要设置不存在的日 (例如2月30日)。

23.3.3.6 RTCMONTHR (月列寄存器 (仅适用于PAGE0))

	7	6	5	4	3	2	1	0
比特符号	-	-	-	MO				
复位后	0	0	0	未定义	未定义	未定义	未定义	未定义

位	比特符号	类型	功能
7-5	–	R	读作 0。
4-0	MO	R/W	设置月份位数寄存器。 0_0001 : 一月 0_0111 : 七月 0_0010 : 二月 0_1000 : 八月 0_0011 : 三月 0_1001 : 九月 0_0100 : 四月 1_0000 : 十月 0_0101 : 五月 1_0001 : 十一月 0_0110 : 六月 1_0010 : 十二月

注:禁止除上列以外的设置。

23.3.3.7 RTCMONTHR (24-小时时钟或12-小时时钟选择 (仅适用于PAGE1))

	7	6	5	4	3	2	1	0
比特符号	-	-	-	-	-	-	-	MO0
复位后	0	0	0	0	0	0	0	未定义

位	比特符号	类型	功能
7-1	–	R	读作 0。
0	MO0	R/W	0: 12-小时 1: 24-小时

注:实时时钟在运行时不要更改RTCMONTHR<MO0>。

23.3.3.8 RTCYEARR (年列寄存器 (仅适用于PAGE0))

	7	6	5	4	3	2	1	0
比特符号	YE							
复位后	未定义	未定义	未定义	未定义	未定义	未定义	未定义	未定义

位	比特符号	类型	功能
7-0	YE	R/W	设置年份位数寄存器。 0000_0000 : 00年　0001_0000 : 10年　0110_0000 : 60年 0000_0001 : 01年　・　・ 0000_0010 : 02年　0010_0000 : 20年　0111_0000 : 70年 0000_0011 : 03年　・　・ 0000_0100 : 04年　0011_0000 : 30年　1000_0000 : 80年 0000_0101 : 05年　・　・ 0000_0110 : 06年　0100_0000 : 40年　1001_0000 : 90年 0000_0111 : 07年　・　・ 0000_1000 : 08年　01001_0000 : 50年　・ 0000_1001 : 09年　・　1001_1001 : 99年

注:禁止除上列以外的设置。

23.3.3.9 RTCYEARR (闰年寄存器 (仅适用于PAGE1))

	7	6	5	4	3	2	1	0
比特符号	-	-	-	-	-	-	LEAP	
复位后	0	0	0	0	0	0	未定义	未定义

位	比特符号	类型	功能
7-2	−	R	读作 0。
1-0	LEAP	R/W	00 : 闰年 01 : 闰年后的一年 10 : 闰年后的两年 11 : 闰年后的三年

23.3.3.10 RTCPAGER (PAGE寄存器 (PAGE0/1))

	7	6	5	4	3	2	1	0
比特符号	INTENA	-	-	ADJUST	ENATMR	ENAALM	-	PAGE
复位后	0	0	0	0	未定义	未定义	0	0

位	比特符号	类型	功能
7	INTENA	R/W	INTRTC 0:禁用 1:启用
6-5	−	R	读作 0。
4	ADJUST	R/W	[写] 0:不指定 1:设置ADJUST请求 调整秒钟。秒数计数器递增计数时该请求被取样。 如果经过的时间在0与29秒之间，秒数计数器被清为 “0“。 如果经过的时间在30与59秒之间，分钟数计数器进位，秒数计数器被清为 “0“。 [读取] 0: ADJUST未请求 1: ADJUTS已请求 如果读取 “1“，表明ADJUST正在执行。如果读取 “0“，表明执行已完成。
3	ENATMR	R/W	时钟 0: 禁用 1: 启用
2	ENAALM	R/W	ALARM 0: 禁用 1: 启用
1	−	R	读作 0。
0	PAGE	R/W	PAGE选择 0:选择PAGE0 1:选择PAGE1

注 1: 不能执行读取-修改-写入操作。

注 2: 如要将中断启用位设置到<ENATMR>，<ENAALM>与<INTENA>，必须遵循此处规定的顺序。确保不进行同时设置 (确保中断启用与时钟/报警启用之间有一个时间滞后)。要改变<ENATMR>与<ENAALM>的设置，必须先禁用<INTENA>。

示例:时钟设置/报警器设置

	7	6	5	4	3	2	1	0	
RTCPAGER←	0	0	0	0	1	1	0	0	启用时钟与报警器
RTCPAGER←	1	0	0	0	1	1	0	0	启用中断

23.3.3.11 RTCRESTR (复位寄存器 (PAGE0/1))

	7	6	5	4	3	2	1	0
比特符号	DIS1HZ	DIS16HZ	RSTTMR	RSTALM	-	-	-	-
复位后	1	1	0	0	0	1	1	1

位	比特符号	类型	功能
7	DIS1HZ	R/W	1 Hz 0:启用 1: 禁用
6	DIS16HZ	R/W	16 Hz 0: 启用 1: 禁用
5	RSTTMR	R/W	[写] 0: 不指定 1: 秒数计数器复位 复位秒数计数器。使用低速时钟进行请求采样。 [读] 0: 无复位请求 1: 复位已请求 如果读取“1“，则表明复位正在执行。如果读取“0“，表明执行已完成。
4	RSTALM	R/W	0:忽略 1: 报警复位 报警器寄存器 (分钟列，小时列，日列，周日列)初始化如下。 分钟:00，小时:00，日:01，周日:星期日
3	−	R	读作 0。
2-0	−	R/W	写入“1“。

注 1: 不能执行读取-修改-写入操作。

<DIS1HZ>与<DIS16MHZ>，用于报警器的RTCPAGER<ENAALM>，1Hz中断与16Hz中断，其设置显示以下。

<DIS1HZ>	<DIS16HZ>	RTCPAGER <ENAALM>	中断源信号
1	1	1	警报
0	1	0	1 Hz
1	0	0	16 Hz
其它			中断未生成。

23.4 操作说明

RTC结合了一个从32.768kHz信号生成1Hz信号的秒数计数器。

在使用RTC时候，必须考虑秒数计数器的运行。

23.4.1 读取时钟数据

1. 使用1Hz中断

该1Hz中断的生成与秒数计数器的递增计数同步。

如果读取数据在1Hz中断之后发生，则数据可正确读取。

2. 使用成对读取

有一种可能性，就是如果内部计数器在读取期间进位，则时钟数据读取可能不正确。要确保正确的数据读取，按如下所示读取时钟数据两次。连续成对数据读取需要匹配。

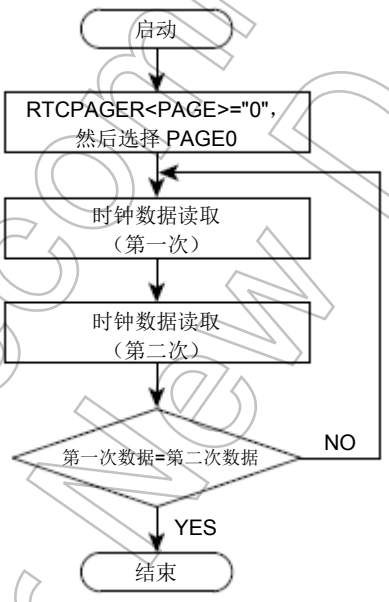


图23-2 时钟数据读取流程图

23.4.2 写入时钟数据

在写入期间进位破坏了正确的数据写入。以下程序确保正确的数据写入。

1. 使用1Hz中断

该1Hz中断的生成与秒数计数器递增计数同步。如果数据在1Hz中断与随后的一秒钟计数之间的时间内写入，则写入正确地完成。

2. 复位计数器

在复位秒数计数器后写入数据。

该1Hz中断在紧接计数器复位后启用中断的一秒钟之后生成。

该时间必须在中断之后一秒钟之内设置。

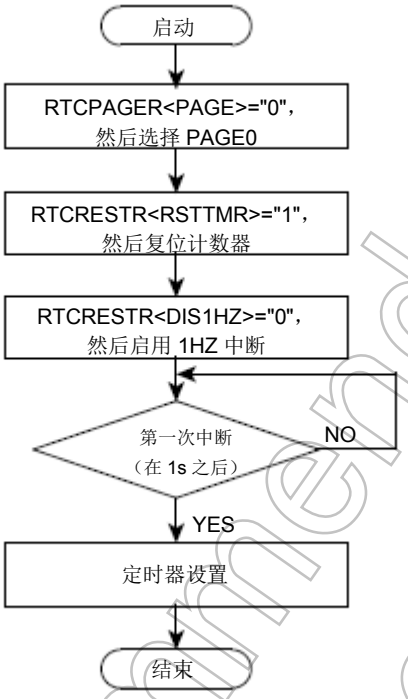


图23-3 时钟数据写入流程图

3. 禁用时钟

写入 “0“到RTCPAGER<ENATMR>禁用时钟运行，包括进位。

1Hz-中断之后停止时钟。秒数计数器保持计数。

再次设置时钟并在下一个1Hz-中断之前的一秒钟之内启用时钟

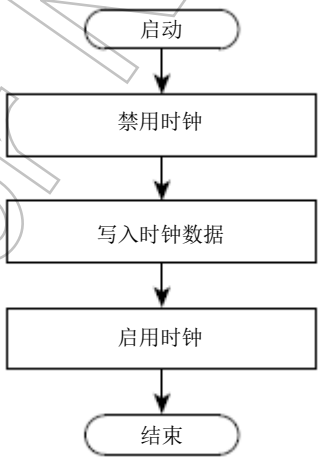


图23-4 禁用时钟流程图

23.4.3 进入低电耗模式

要进入睡眠模式 (在此模式中，系统时钟在改变时钟数据，调整秒钟或复位时钟之后停止)，务必遵守以下程序之一

1. 在更改时钟设置寄存器，设置RTCPAGER<ADJUST>位或设置RTCRESTR<RSTTMR>位之后，等待一秒钟以生成一个中断。

2. 在更改时钟设置寄存器，设置RTCPAGER<ADJUST>位或设置RTCRESTR<RSTTMR>位之后，读取相应时钟寄存器的值，<ADJUST>或<RSTTMR>，确保所做的设置已反映出来。

23.5 报警器功能

写入 “1“到 RTCPAGER<PAGE>即启用 PAGE1 寄存器的报警器功能。以下三个信号之一则输出到 $\overline{\text{ALARM}}$ 引脚。

1. “低“脉冲 (报警寄存器与时钟一致时)
2. 1Hz循环 “低“脉冲
3. 16Hz循环 “低“脉冲

在以上所示的任何情况下，INTRTC输出一个低速时钟循环脉冲。它还同时输出INTRTC中断请求。该INTRTC中断信号为下降沿触发信号。在CG中断模式控制寄存器中将该下降沿规定为活动状态

23.5.1 “低“脉冲 (报警寄存器与时钟一致时)

“低“脉冲在PAGE0时钟寄存器与PAGE1报警器寄存器的值对应时输出到 $\overline{\text{ALARM}}$ 引脚。INTRTC中断则得以生成，报警器得以触发。

报警器设置

在报警器禁止时进行报警器初始化。写入“1“到RTCRESTR <RSTALM>。

这就使报警器设置为00分钟，00小时，01日与星期日。

写入数据到相关的PAGE1寄存器即完成报警器的分钟，小时，日期与日的设置。

采用RTCPAGER<ENAALM>位启用报警器。采用RTCPAGER<INTENA>位启用中断。

以下是在星期一5日中午 (12:00时)从报警引脚输出报警的实例程序。

	7	6	5	4	3	2	1	0	
RTCPAGER ←	0	0	0	0	1	0	0	1	禁用报警，设置PAGE1
RTCRESTR ←	1	1	0	1	0	0	0	0	报警器初始化
RTCDAYR ←	0	0	0	0	0	0	0	1	星期一
RTCDATER ←	0	0	0	0	0	1	0	1	第5日
RTCHOURR ←	0	0	0	1	0	0	1	0	设置12点
RTCMINR ←	0	0	0	0	0	0	0	0	设置00分钟
RTCPAGER ←	0	0	0	0	1	1	0	0	启用报警器
RTCPAGER ←	1	0	0	0	1	1	0	0	启用中断

上述报警器与低速时钟同步工作。在CPU高频振荡运行时，对于设置有效的时间寄存器，在fs处可能发生最多一个时钟脉冲延时 (大约30μs)。

注: 要使报警器反复工作 (如每星期三12:00时)，必须在报警设置时间与RTC计数匹配时生成的INTRTC中断例行程序期间设置下一个报警。

23.5.2 1Hz 周期 “低“脉冲

通过在设置 RTCPAGER<ENAALM>= “0“，RTCRESTR<DIS1Hz>= “0“与<DIS16Hz>= “1“之后再设置 RTCPAGER<INTENA>= “1“，RTC 给 $\overline{\text{ALARM}}$ 引脚输出一个低速 1Hz 时钟脉冲的 “低“脉冲周期。它还同时生成一个中断 INTRTC 中断。

23.5.3 16Hz 周期 “低“脉冲

通过在设置RTCPAGER<ENAALM>= “0“，RTCRESTR<DIS1Hz>= “1“以及<DIS16Hz>= “0“之后再设置 RTCPAGER<INTENA>= “1“，RTC给 $\overline{\text{ALARM}}$ 引脚输出一个低速16Hz时钟脉冲的 “低“脉冲周期。它还同时生成一个INTRTC中断。

24. 闪存

本节叙述了闪存的硬件配置与操作。

24.1 闪存

24.1.1 特点

1. 存储器容量

TMPM364F10FG包含有闪存。存储器容量与配置如下表所示。

可向每个程序块进行单独的写入存取。在CPU存取内部闪存时，使用32-位数据总线宽度。

2. 写入/擦除时间

写入每页都执行。TMPM364F10FG含有128个字符。

页写入需要1.25ms (标准)，而不考虑字符数。

一个程序块擦除需要0.1sec (标准)

下表显示了每片的写入与擦除时间。

产品名称	存储器容量	块配置			字符数	写入时间	擦除时间
		128 KB	64 KB	32 KB			
TMPM364F10FG	1024 KB	7	1	2	128	2.56 秒	1.0 秒

注: **上述值为理论值，不包括数据传递时间。每片写入时间取决于用户采用的写入方法。**

3. 程序设计方法

有两类在板编程模式，供用户在装置安装到用户的板上时为该装置设计 (重写)程序:

a. 用户引导启动模式

可支持用户的原重写方法。

b. 单引导启动模式

可支持使用串行数据传送的重写方法 (东芝的唯一方法)。

4. 重写方法

本装置中包含的闪存除有些特殊功能之外，一般都符合适用的电子设备工程联合协会JEDEC标准。因此，如果用户目前使用外部闪存装置，则容易将这些功能融入该装置中。此外，由于这些功能均采用闪存芯片中内置的电路自动地执行，用户不需要编制其自己的程序来实现复杂的写入与擦除功能。

符合JEDEC标准的功能	经修改，增加或删除的功能
・自动编程 ・自动芯片擦除 ・自动块擦除 ・数据轮询 / 轮转位	<Modified>程序块保护 (仅支持软件保护) <Deleted>擦除恢复-挂起功能

5. 保护/安全功能

该装置配备有读取保护功能，防止从任何外部写入器装置读取闪存数据。另一方面，仅通过基于命令的软件编程即可获得重写保护。不支持采用+12VDC的任何软件设置方法。ROM保护与安全功能的详情见“ROM保护“一章。

注: **如果密码设置到0xFF (已擦除数据)，则由于容易猜到密码，难以安全地保护数据。即便不采用Single Boot模式，建议将唯一值设置为密码。**

24.1.2 闪存区方块图

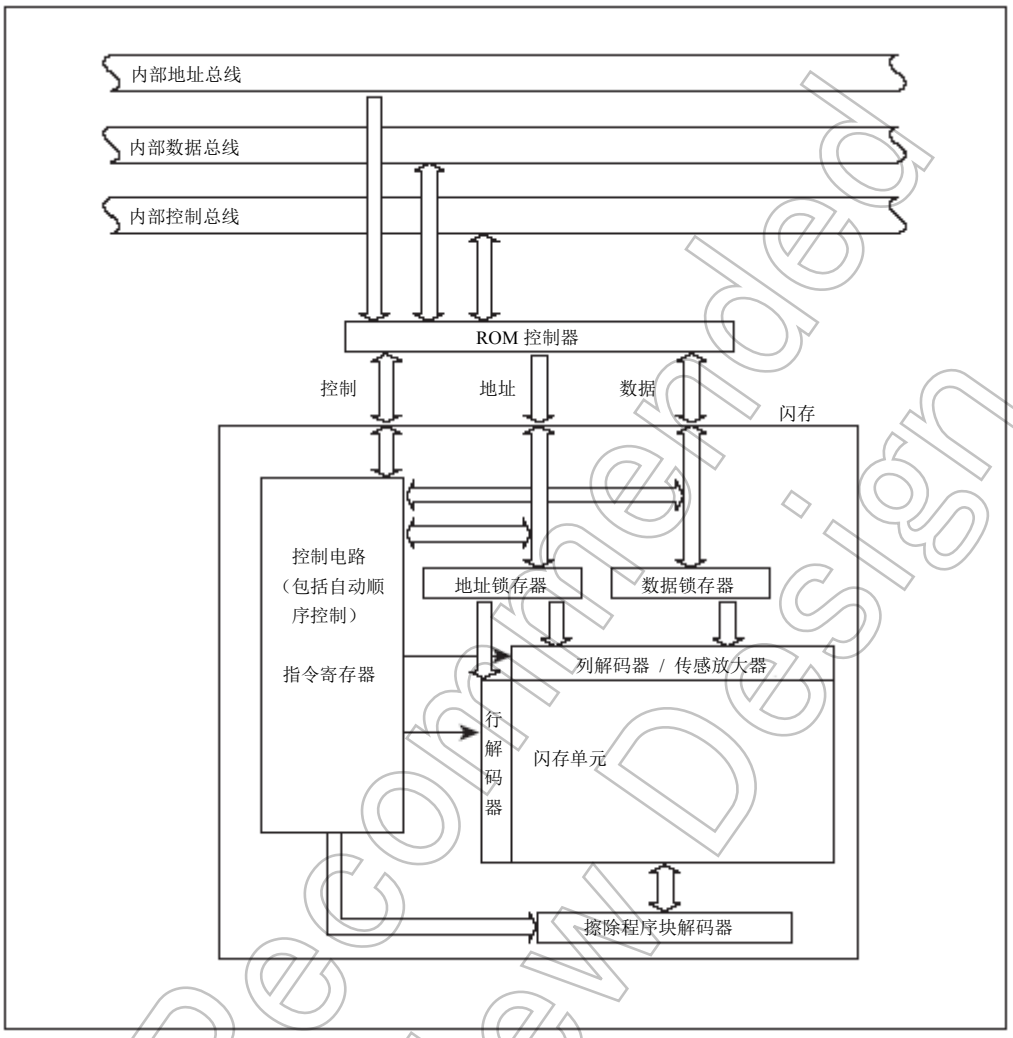


图24-1 闪存区方块图

24.2 工作模式

本装置具有三种工作模式，包括不使用内部闪存的模式。

表24-1 工作模式

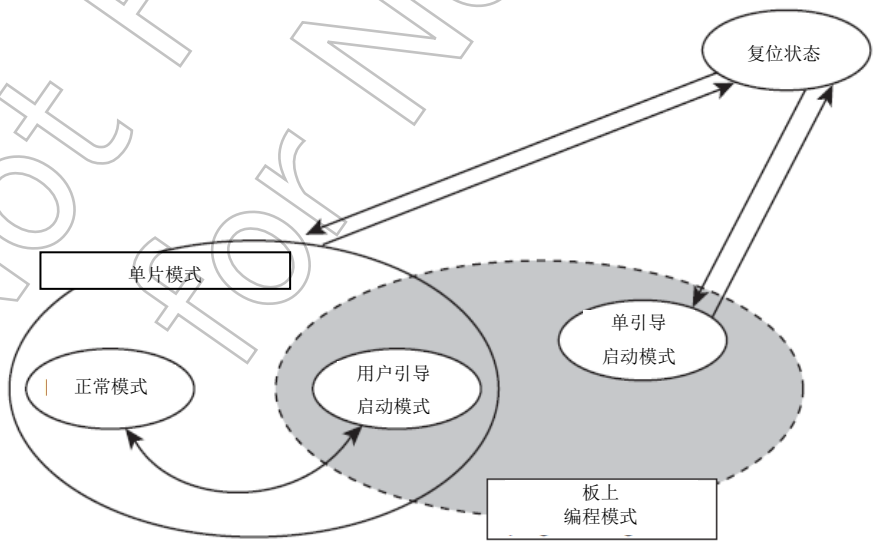
操作模式	操作详情
单片模式 正常模式 用户引导启动模式	复位清除之后从内部闪存启动。 在这一操作模式中，确定了两种不同的模式，比如执行用户应用程序的模式与在用户设置的板上重写闪存的模式。前者称之为 “正常模式“，后者称之为 “用户引导启动模式“。 用户可唯一地配置系统，以在这两个模式之间切换。举例来说，用户可自由设计系统，这样，在端口 “A0“设置到 “1“时即选择正常模式，在设置到 “0 “时即选择用户引导启动模式。用户应将某个例行程序编成应用程序的一部分，以在模式的选择上做出决定。
单引导启动模式	复位清除之后从内部BOOT ROM (MASK ROM)启动。在Boot ROM中，编写出通过本装置的串行端口启用闪存在用户设置上重写的算法程序。通过串行端口连接到一个外部主计算机，该内部闪存可通过依照预先规定的协议传递数据来编程。

在上表所列的闪存操作模式当中，User Boot模式与Single Boot模式均为可编程模式。这两个模式，User Boot模式与Single Boot模式，称之为 “板上编程“模式，内部闪存的板上重写可在用户的设置上进行。

单片或单个引导启动操作模式均可在装置处于复位状态时通过外部设置 $\overline{\text{BOOT}}$ (PI0)引脚的电平来选取。

表24-2 工作模式设置

操作模式	引脚	
	$\overline{\text{RESET}}$	$\overline{\text{BOOT}}$ (PI0)
单片模式	0 → 1	1
单引导启动模式	0 → 1	0



用户设置切换方法

图24-2 模式转换图

24.2.1 复位操作

如要对该装置复位，则要确保电源电压在工作电压范围之内，内振荡器已经稳定下来，$\overline{\text{RESET}}$ 输入保持在 "0"的最小持续时间达到 12 个系统时钟脉冲 (64MHz 运行 0.19μs；在复位之后采用 "1/1"时钟档模式)。

注1:**有必要在一旦接通电源时RESET输入采用 "0"，其最小持续时间为700μs而不考虑工作频率。**

注2:**当闪速自动编程或擦除在进行中时，需要至少0.5 μs的复位期限而不考虑系统时钟脉冲频率。在这一条件下，在复位之后需要大约 2 ms 启用读取。**

24.2.2 User Boot模式 (单片模式)

User Boot模式就是使用用户定义的闪存编程例行程序。该模式在以前应用的闪存程序代码的数据传输母线以及串行输入/输出的数据传输母线不同时才使用。它在单片模式下运行，需要从正常模式 (其中用户应用在单片模式下激活)转换到进行闪存编程的User Boot模式。具体地说，就是给用户应用中的复位程序增加一个模式判断例行程序。

进行模式切换的条件需利用符合用户系统设置条件的TMPM364F10FG的输入/输出来进行设置。同样，用户唯一编写的闪存编程例行程序则需要在新的应用中进行设置。本例行程序用于在切换到User Boot模式之后的编程。由于内部闪存中的数据在删除/写入模式期间不能读出，必须在其储存到闪存外的区域时完成编程程序。一旦完成重新编程，建议保护相关闪存程序块，防止其在随后的单片 (正常模式)操作期间意外误用。确信不引起任何异常情况，包括User Boot模式时的不可屏蔽中断。

(1-A)与 (1-B)为采用内部闪存与外存储器中的例行程序进行编程的示例。擦除与编程顺序的详细说明参阅 "24.3 闪存的在板编程 (重写/擦除) "。

24.2.2.1 (1-A)方法 1: 在闪存中储存一个编程程序

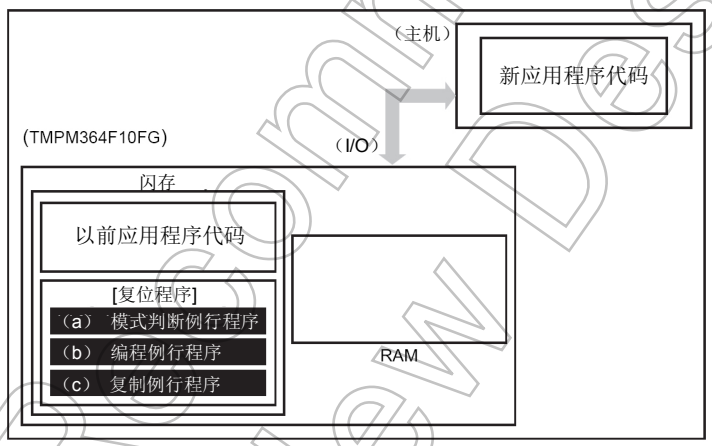
(1) 步骤-1

确定闪存进入用户引导启动模式以及用于转输新程序代码的输入/输出总线所需要的条件(如，引脚状态)。相应地建立硬件和软件。在将TMPM364F10FG安装到印制电路板上之前，使用编程设备将以下程序的例行程序写入一任意闪存程序块。

(a) 模式判断例行程序: 确定是否切换到用User Boot模式的代码

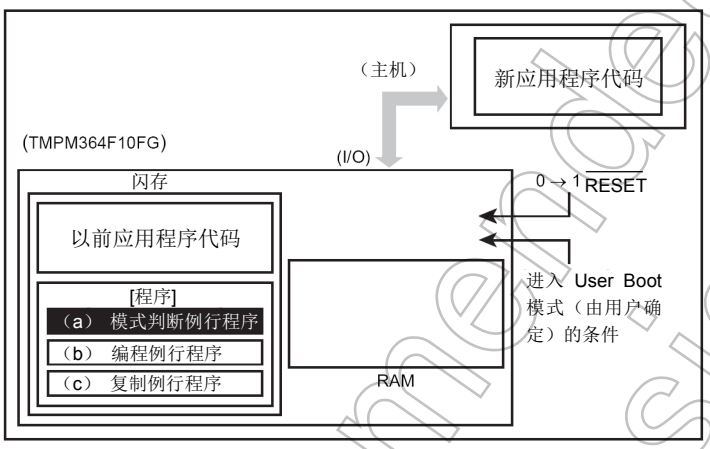
(b) 编程例行程序: 从主机控制器下载新的程序代码，对闪存重新编程的代码

(c) 复制例行程序: 从TMPM364F10FG闪存中将 (b)中所述的数据复制到任何一个TMPM364F10FG片上RAM或外存储器装置的代码。



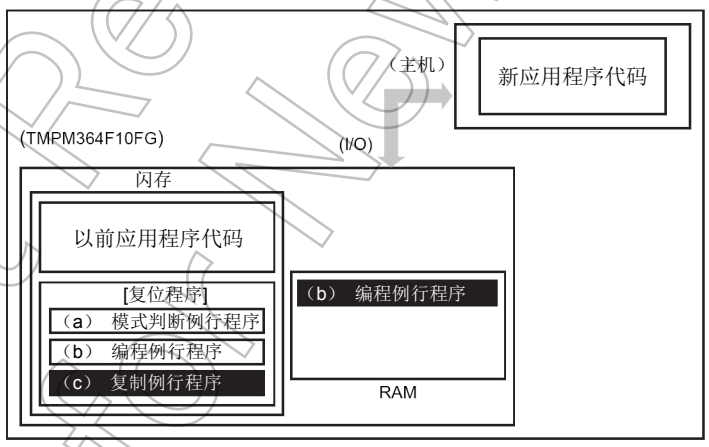
(2) 步骤-2

以下叙述即是编程例行程序安装在复位处理程序中的情况。在 $\overline{\text{RESET}}$ 引脚解除后，复位程序决定是否将 TMPM364F10FG 闪存转入 User Boot 模式。如果模式切换条件得以满足，闪存则进入用户引导启动模式。 (所有中断，包括 NMI 均不得在 User Boot 模式中使用。)



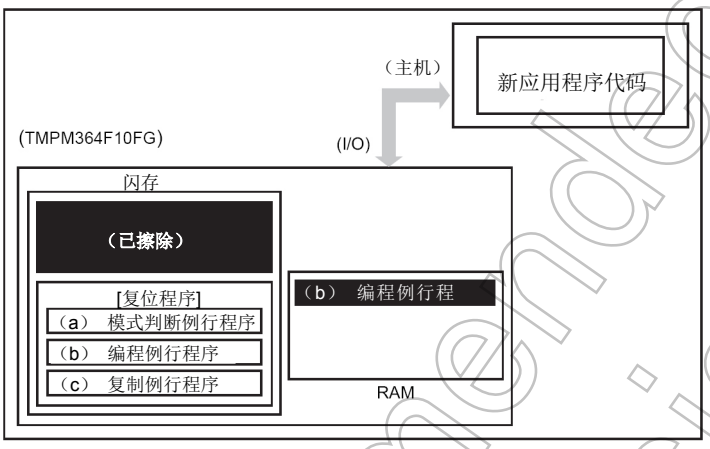
(3) 步骤-3

一旦发生转换到 User Boot 模式，则执行复制例行程序(c)，将闪存编程例行程序(b)复制到 TMPM364F10FG 片上 RAM。



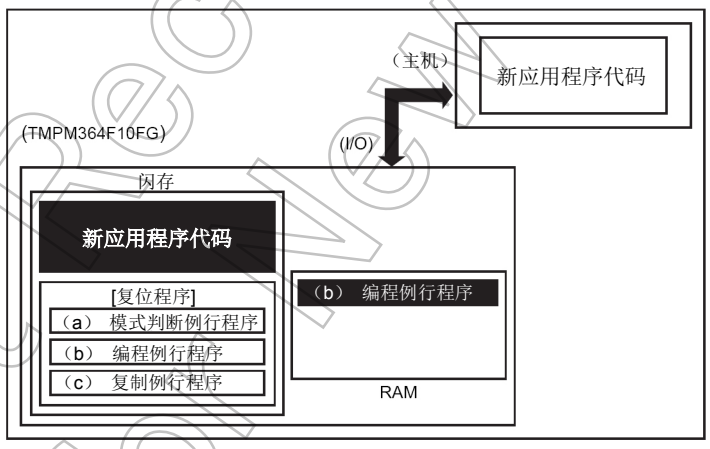
(4) 步骤-4

程序执行跳转到片上 RAM 中的闪存编程例行程序，以清除写入或擦除保护，并擦除包含以前应用程序代码的闪存程序块。



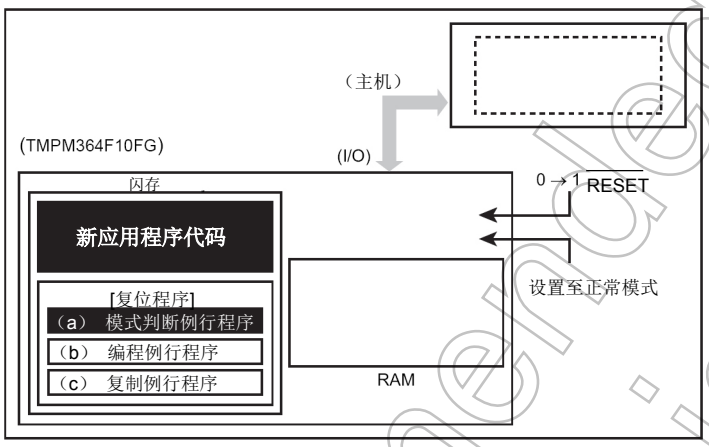
(5) 步骤-5

继续执行闪存编程例行程序，以从主机控制器下载新的程序代码，并将其编程到擦除闪存程序块。在编程完成时，用户程序区中该闪存程序块的写入或擦除保护必须予以设置。



(6) 步骤-6

将 $\overline{\text{RESET}}$ 设置为 “0“，以使 TMPM364F10FG 复位。一旦复位，将片上闪存设置到正常模式。$\overline{\text{RESET}}$ 解除之后，CPU 将启动执行新的应用程序代码。



24.2.2.2 (1-B) 方法 2：从外部主机传输一个编程程序

(1) 步骤-1

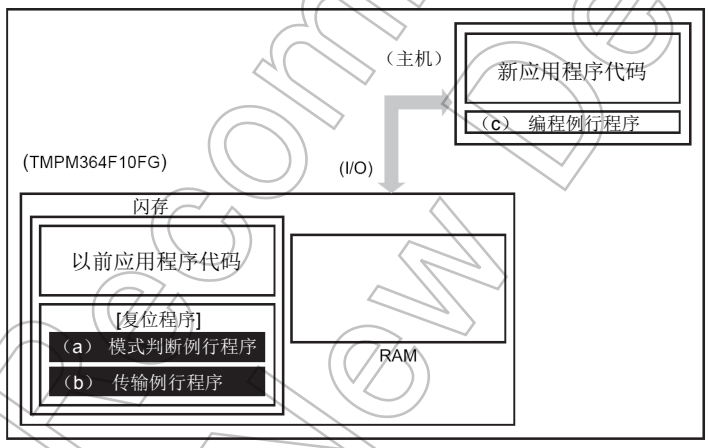
确定闪存进入User Boot模式以及用于转输新程序代码的I/O总线所需要的条件(如，引脚状态)。在将TMPM364F10FG安装到印制电路板上之前，使用编程设备将以下程序的例行程序写入一任意闪存程序块。

(a) 模式判断例行程序: 确定是否切换到User Boot模式的代码

(b) 转输例行程序: 从主机控制器下载新程序代码的代码

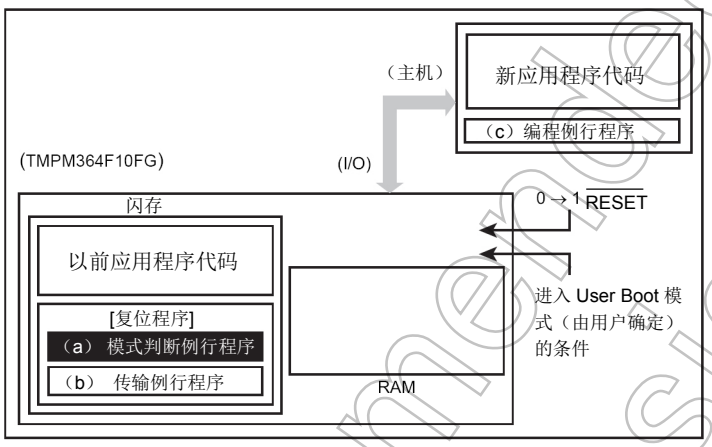
此外，在主机控制器上准备如下所示的编程例行程序:

(c)编程例行程序: 从外部主机控制器上下载新程序代码并对闪存再编程的代码



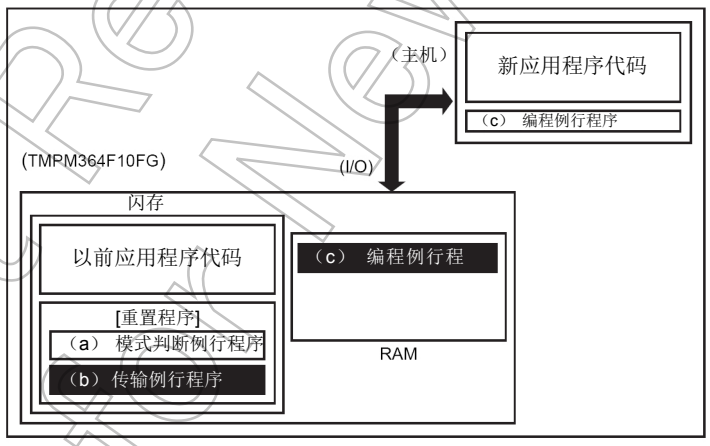
(2) 步骤-2

以下叙述即是编程例行程序安装在复位处理程序中的情况。RESET解除之后，该复位程序决定是否将TMPM364F10FG闪存转入User Boot模式。如果模式切换条件得以满足，闪存则进入User Boot模式。 (所有中断，包括NMI不能在User Boot模式中使用)。



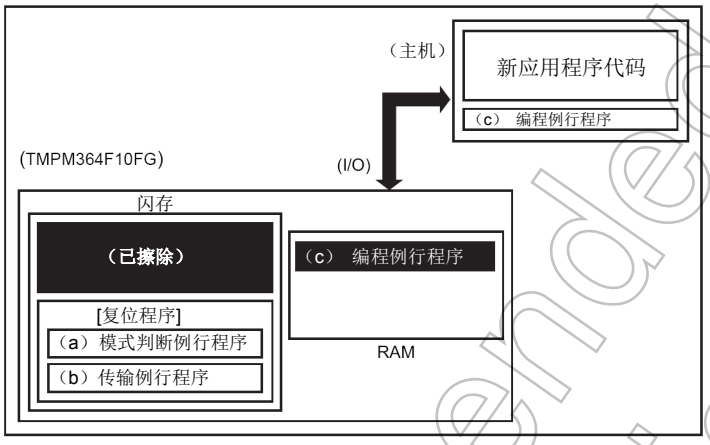
(3) 步骤-3

一旦进入用户引导启动模式，则执行转输例行程序 (b)，以从主机控制器中下载闪存编程例行程序 (c)到TMPM364F10FG片上RAM。



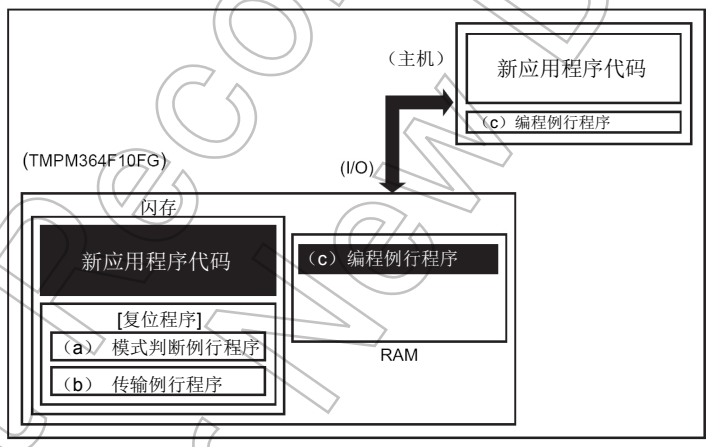
(4) 步骤-4

程序执行跳转到片上RAM中的闪存编程例行程序，以清除写入或擦除保护，并擦除包含以前应用程序代码的闪存程序块。



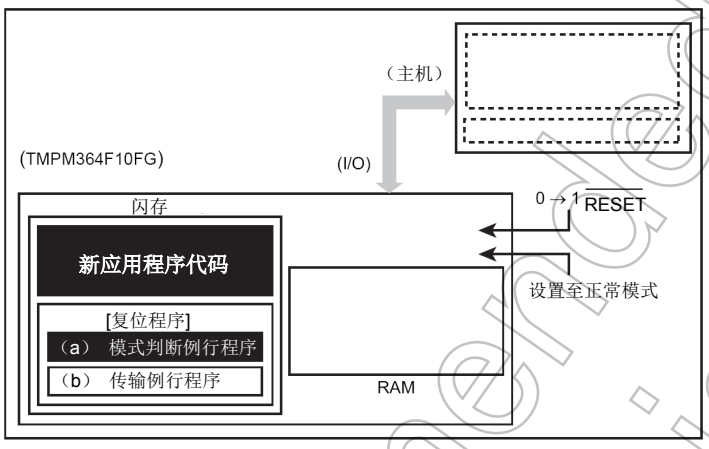
(5) 步骤-5

继续执行闪存编程例行程序，以从主机控制器下载新的程序代码，并将其编程到已擦除闪存程序块。在编程完成时，用户程序区中闪存程序块的写入或擦除保护必须予以设置。



(6) 步骤-6

将 $\overline{\text{RESET}}$ 设置为 “0“低以使TMPM364F10FG复位。一旦复位，将片上闪存设置到正常模式。$\overline{\text{RESET}}$ 解除之后，CPU将启动执行新的应用程序代码。



24.2.3 Single Boot模式

在Single Boot模式中，可使用TMPM364F10FG片上引导启动ROM中所含的程序对闪存进行再编程。该引导启动ROM为屏蔽的ROM。在一经复位便选取Single Boot模式时，引导启动ROM在闪存映射到与其不同的某个地址区域时，则被映射到该地址区域，包括中断向量表。

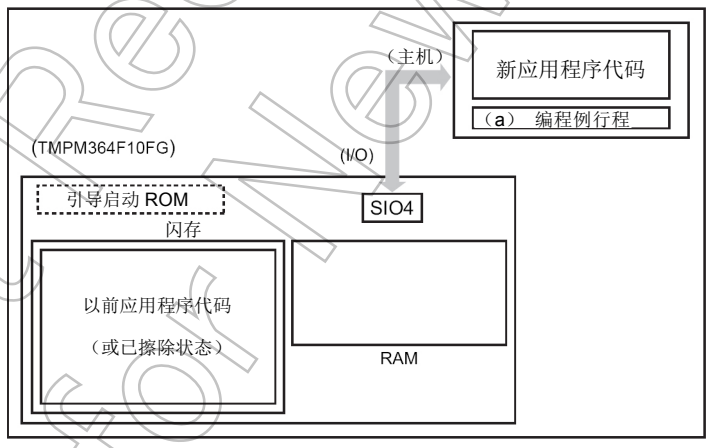
Single Boot模式允许闪存的串行程序设计。TMPM364F10FG的SIO通道4 (SIO4)连接到一个外部主机控制器。通过这一串行链环，编程例行程序被从主机控制器上下载到TMPM364F10FG片上RAM。然后，通过执行编程例行程序对闪存进行再编程。主机既发送指令又发送编程数据对闪存进行再编程。SIO4与主机之间的通信必须遵循随后所述的协议。为了保护闪存的内容安全，应用密码的有效性在将编程例行程序下载到片上RAM之前应验证。如果密码匹配失败，编程例行程序本身的转输即中止。与User Boot模式的情况一样，正在擦除或对闪存编程时，必须禁用Single Boot模式中的所有中断，包括NMI。在Single Boot模式中，引导启动ROM程序33以正常模式执行。

一旦再编程完成，建议在随后的单片 (正常模式)操作期间，对相关闪存程序块设置写入/擦除保护，防止意外的误用。

24.2.3.1 (2-A)片上Boot ROM中使用程序

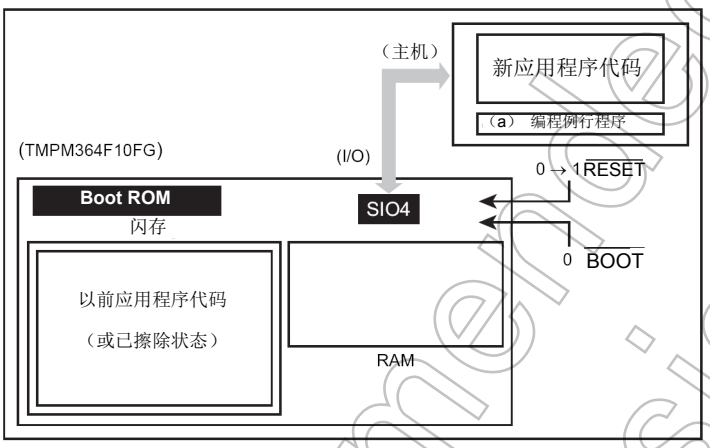
(1) 步骤-1

在执行编程例行程序之前，不需要执行包含旧版程序代码的闪存程序块。由于编程例行程序与编程数据均通过SIO (SIO4)输送，必须将SIO4连接到主机控制器。在主机控制器上准备编程例行程序 (a)。



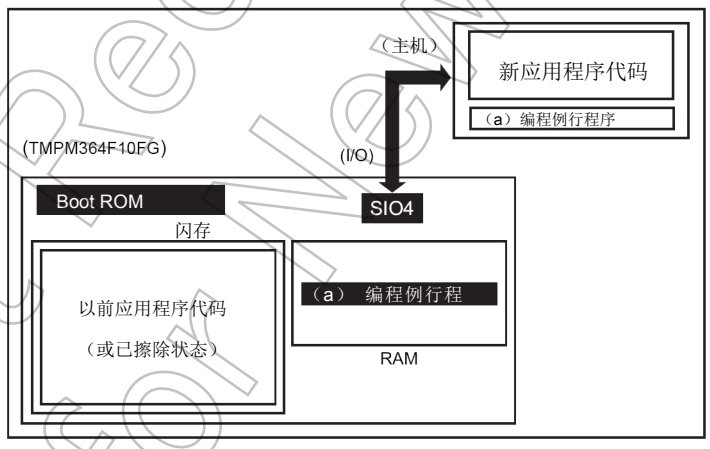
(2) 步骤-2

在 $\overline{\text{BOOT}}$ 引脚已经设置为 “0“时，候将 $\overline{\text{RESET}}$ 引脚设置为 “1“，以取消TMPM364F10FG的复位。复位之后，CPU从片上引导启动ROM重新引导启动。将从主机控制器通过SIO4输送的12-字节密码首先与特殊闪存位置的内容进行比较。 (如果闪存程序块已经被擦除，则密码为0xFF)。



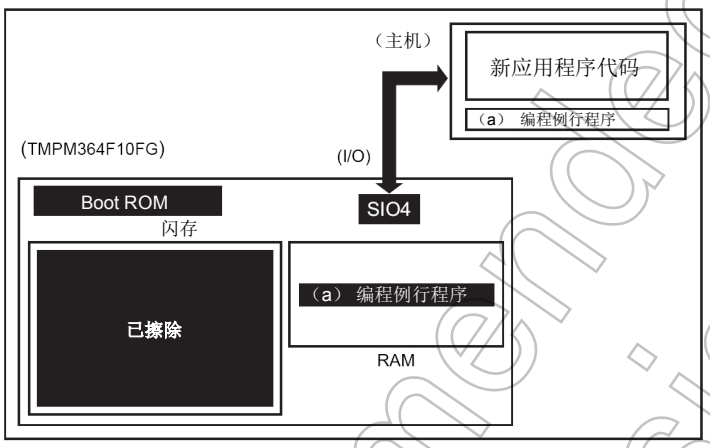
(3) 步骤-3

如密码正确，该引导启动程序则从主机控制器将编程例行程序 (a)下载到TMPM364F10FG的片上RAM。编程例行程序必须储存在RAM结束地址的0x2000_0400范围内。



(4) 步骤-4

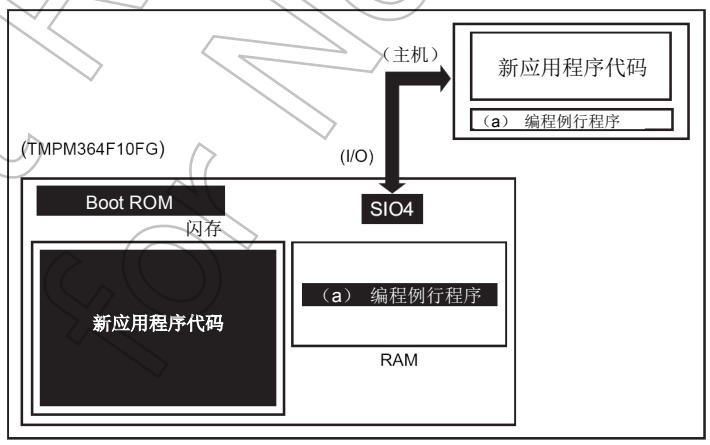
CPU跳转到片上RAM中的编程例行程序 (a)，以擦除包含以前应用程序代码的闪存程序块。可使用程序块擦除或片擦除指令。



(5) 步骤-5

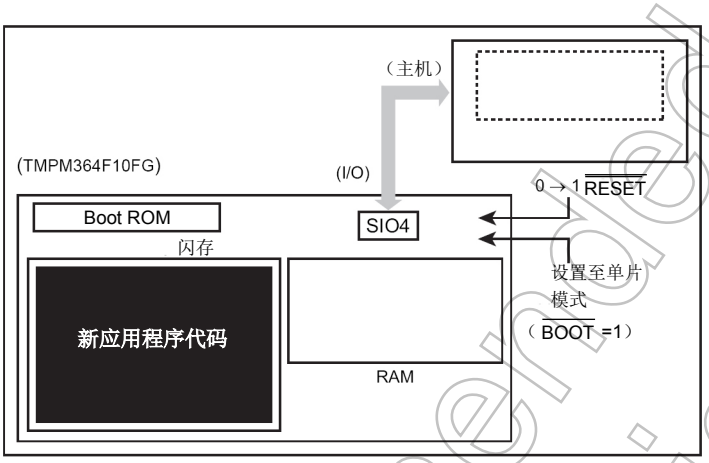
下一步，编程例行程序 (a)从主机控制器下载新的应用程序代码，并将其编程到已擦除闪存程序块。在编程完成时，用户程序区中该闪存程序块的写入或擦除保护必须予以设置。

在以下示例中，至于编程例行程序，新的程序代码则通过相同的SIO4通道来自相同的主机控制器。然而，一旦编程例行程序已开始在片上RAM中执行，可自由更改转输路径以及转输源。建立板硬件与编程例行程序以适应特别需要。



(6) 步骤-6

在闪存的编程完成时，断开板的电源并断开主机与目标板之间的电缆。再次接通电源使TMPM364F10FG在单片 (正常)模式中重新引导启动，以执行新的程序。



24.2.4 Single Boot模式配置

要执行在板编程，按照如下所示的配置，采用单引导启动模式引导启动TMPM364F10FG。

$\overline{\text{BOOT}}$ (PI0) = 0

$\overline{\text{RESET}}$ = 0 →1

将 $\overline{\text{RESET}}$ 输入设置为 “0“，并将各个 $\overline{\text{BOOT}}$ (PI0)引脚设置到以上所示的值，然后解除 $\overline{\text{RESET}}$ 引脚 (高)。

24.2.5 内存地图

图24-3显示正常模式与单引导启动模式中内存地图的比较。单引导启动模式中，内部闪存被映射到0x3F80_0000以及随后的地址，内部引导启动ROM (Mask ROM)则被映射到0x0000_0000 ~ 0x0000_0FFF。

各装置的内部闪存与RAM地址显示如下。

产品名称	闪存容量	RAM容量	闪存地址 (Single Chip/Single Boot模式)	RAM地址
TMPM364F10FG	1024 KB	64 KB	0x0000_0000-0x000F_FFFF 0x3F80_0000-0x3F8F_FFFF	0x2000_0000-0x2000_FFFF

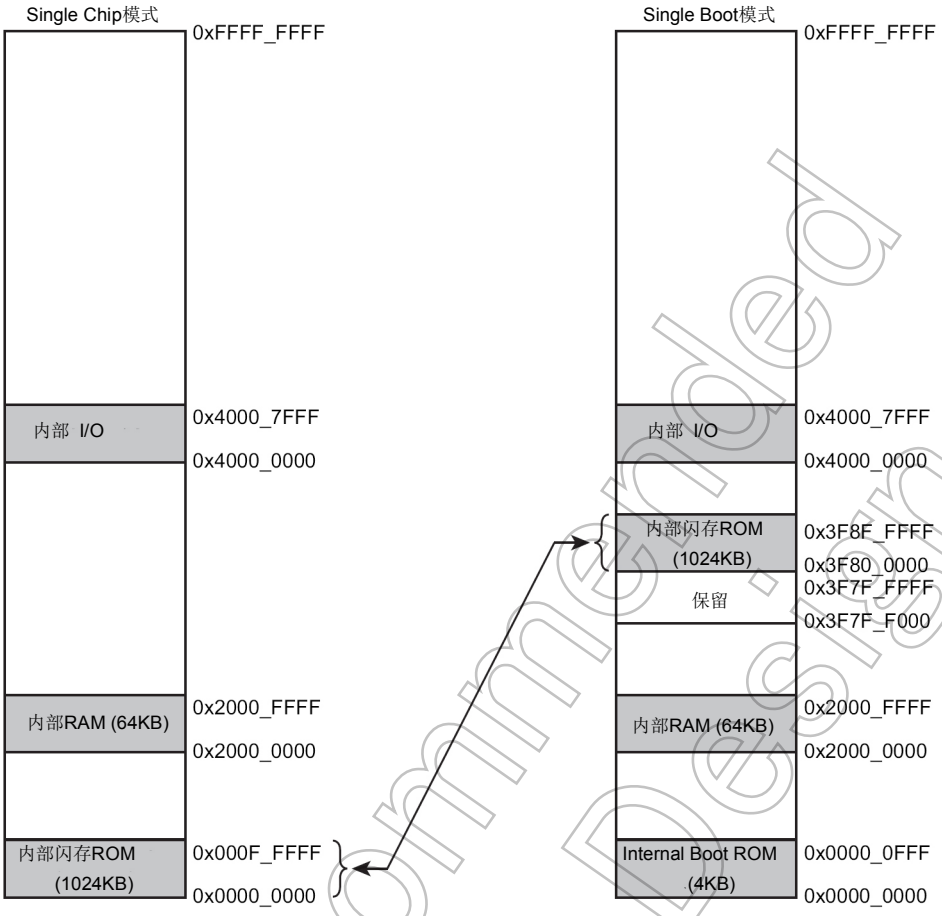


图24-3 TMPM364F10FG的内存地图

24.2.6 接口规格

在单引导启动模式中，SIO通道用于与程序控制器的通信。相同配置被用于执行在板编程的程序控制器的通信格式。支持UART (异步)与I/O接口 (同步)模式。该通信格式显示如下

- UART通信

  通信通道 : SIO 通道 4

  串行转输模式:UART (异步)，半-双工制，LSB先

  数据长度 : 8 位

  校验位 : 无

  STOP位 : 1 位

  波特率 : 任意波特率

- I/O接口模式

  通信通道 : SIO 通道 4

  串行转输模式 : I/O接口模式，全-双工制，LSB先传输

  同步时钟 (SCLK4) : 输入模式

  信号交换信号 : PN3，配置为输出模式

  波特率 : 任意波特率

表24-3 需要引脚连结

引脚		接口	
		UART	I/O接口模式
电源引脚	RVDD3	o	o
	AVDD3	o	o
	DVDD3B	o	o
	DVDD3A	o	o
	AVSS	o	o
	DVSS	o	o
模式设置引脚	$\overline{\text{BOOT}}$ (PI0)	o	o
复位引脚	$\overline{\text{RESET}}$	o	o
通信引脚	TXD4 (PN0)	o	o
	RXD4 (PN1)	o	o
	SCLK4 (PN2)	×	O (输入模式)
	PN3	×	O (输出模式)

24.2.7 数据传输格式

表24-4，表24-6 ~ 表24-9举例说明各工作模式时的操作指令与数据传输格式。连同本节一起，参阅“24.2.10 引导启动程序的运行“。

表24-4 Single Boot模式指令

代码	指令
0x10	RAM传输
0x20	显示闪存总数
0x30	显示产品信息
0x40	芯片与保护位擦除

24.2.8 内存储器限制

Single Boot模式对内部RAM与ROM设置有限制，如表24-5所示。

表24-5 Single Boot模式中的限制

存储器	细节
内部RAM	引导启动ROM中包含的程序使用0x2000_0000 ~ 0x2000_03FF的区作为一个工作区。 将RAM传输程序储存到从0x2000_0400到RAM的结束地址。
内部ROM	以下地址分配给储存软件标识信息与密码。 不推荐将程序储存在这些地址中。 0x3F8F_FFF0 ~ 0x3F8F_FFFF

24.2.9 引导启动程序转输格式

下表显示了各个引导启动程序指令的转输格式。本节与“24.2.10 引导启动程序的运行“一章结合使用。

24.2.9.1 RAM传输

表24-6 RAM传输指令的传输格式

	字节	从控制器传输到 TMPM364F10FG的数据	波特率	从TMPM364F10FG传输到 控制器的数据
Boot ROM	1 字节	串行运行模式与波特率 对于UART 模式 : 0x86 对于I/O 接口模式 : 0x30	要求的波特率 (注1)	-
	2 字节	-		串行运行模式ACK字节 ・ 对于UART模式 - 正常应答 : 0x86 (如果波特率不能正确设置，则引导启动程序异常终止。) ・ 对于 I/O接口模式 - 正常应答 : 0x30
	3 字节	指令码 (0x10)		-
	4 字节	-		指令码ACK应答字节 (注2) - 正常应答 : 0x10 - 否定应答 : 0xX1 - 通信错误 : 0xX8
	5 字节 ~ 16 字节	密码顺序 (12 字节) 0x3F8F_FFF4 ~ 0x3F8F_FFFF		-
	17 字节	字节5 ~ 16的校验和值		-
	18 字节	-		校验和ACK字节 (注2) - 正常应答 : 0x10 - 否定应答 : 0xX1 - 通信错误 : 0xX8
	19 字节	RAM储存开始地址31 ~ 24		-
	20 字节	RAM储存开始地址23 ~ 16		-
	21 字节	RAM储存开始地址15 ~ 8		-
	22 个字节	RAM储存开始地址7 ~ 0		-
	23 字节	RAM储存开始地址15 ~ 8		-
	24 字节	RAM储存开始地址7 ~ 0		-
	25 字节	字节19 ~ 24的校验和值		-
	26 字节	-		校验和ACK字节 (注2) - 正常应答:0x10 - 否定应答:0xX1 - 通信错误:0xX8
	27 字节 ~ m字节	RAM储存数据		-
	m+ 1 字节	字节27 ~ m的校验和值		-
	m+ 2 字节	-		校验和ACK字节 (注2) - 正常应答 :0x10 - 否定应答 : 0xX1 - 通信错误 : 0xX8
RAM	m+ 3 字节	-		跳转到RAM储存开始地址

注 1 :**在I/O接口模式中，第1与第2字节的传输波特率必须是所要求波特率的1/16。**

注 2: **在任何否定应答的情况下，引导启动程序返回到其等待指令码 (第3字节)的状态。在I/O接口模式中，如果发生通信错误，则不出现否定应答。**

注 3: **第19 ~ 第25字节必须在0x2000_0400 ~ RAM结束地址的RAM地址范围之内。**

24.2.9.2 显示闪存总数

表24-7 显示闪存总和指令的转输格式

	字节	从控制器传输到 TMPM364F10FG的数据	波特率	从TMPM364F10FG传输到 控制器的数据
Boot ROM	1 字节	串行运行模式与波特率 对于UART模式 : 0x86 对于I/O接口模式 : 0x30	要求的波特率 (注1)	-
	2 字节	-		串行运行模式ACK字节 ・ 对于UART模式 - 正常应答 : 0x86 (如果波特率不能正确设置，则引导启动程序异常终止。) ・ 对于I/O接口模式 - 正常应答 : 0x30
	3 字节	指令码 (0x20)		-
	4 字节	-		指令码确认ACK字节 (注2) - 正常应答 : 0x10 - 否定应答 : 0xX1 - 通信错误 : 0xX8
	5 字节	-		总和 (高位字节)
	6 字节	-		总和 (低位字节)
	7 字节	-		字节5与6的校验和值
	8 字节	(等待下一个指令码。)		-

注 1：**在I/O接口模式中，第1与第2字节的传输波特率必须是所要求波特率的1/16。**

注 2：**在任何否定应答的情况下，引导启动程序返回到其等待指令码 (第3字节)的状态。在 I/O 接口模式中，如果发生通信错误，则不出现否定应答。**

24.2.9.3 显示产品信息的输送格式

表24-8 显示产品信息指令的转输格式

	字节	从控制器传输到 TMPM364F10FG的数据	波特率	从TMPM364F10FG传输到 控制器的数据
Boot ROM	1 字节	串行运行模式与波特率 对于UART模式 : 0x86 对于I/O接口模式 : 0x30	要求的波特率 (注1)	−
	2 字节	−		串行运行模式ACK字节 ・对于UART模式 - 正常应答 : 0x86 (如果波特率不能正确设置，则引导启动程序异常终止。) ・ 对于I/O接口模式 - 正常应答 : 0x30
	3 字节	指令码 (0x30)		−
	4 字节	−		指令码ACK字节 (注2) - 正常应答 : 0x10 - 否定应答 : 0xX1 - 通信错误 : 0xX8
	5 字节	−		地址0x3F8F_FFF0处的闪存数据
	6 字节	−		地址0x3F8F_FFF1处的闪存数据
	7 字节	−		地址0x3F8F_FFF2处的闪存数据
	8 字节	−		地址0x3F8F_FFF3处的闪存数据
	9 字节 ~ 20 字节	−		产品名称 (12-字节ACCII代码) 从第9字节开始: 'TMPM360F1_ _'
	21 字节 ~ 24 字节	−		密码比较开始地址 (4 字节) 从第21字节开始: 0xF4, 0xFF, 0x8F, 0x3F
	25 字节 ~ 28 字节	−		RAM开始地址 (4 字节) 从第25字节开始: 0x00, 0x00, 0x00, 0x20
	29 字节 ~ 32 字节	−		模型数据 (4 字节) 从第29字节开始: 0x00, 0x00, 0x00, 0x00
	33 字节 ~ 36 字节	−		RAM结束地址 (4 字节) 从第33字节开始: 0xFF, 0xFF, 0x00, 0x20
	37 字节 ~ 40 字节	−		模型日期 (4 字节) 从第37字节开始: 0x00, 0x00, 0x00, 0x00
	41 字节 ~ 44 字节	−		模型日期 (4 字节) 从第41字节开始: 0x00, 0x00, 0x00, 0x00

表24-8 显示产品信息指令的转输格式

	字节	从控制器传输到 TMPM364F10FG的数据	波特率	从TMPM364F10FG传输到 控制器的数据
	45 字节 ~ 46 字节	-		模型数据 (2 字节) 从第45字节开始： 0x00, 0x00
	47 字节 ~ 50 字节	-		闪存开始地址 (4 字节) 从第47 字节开始: 0x00, 0x00, 0x80, 0x3F
	51 字节 ~ 54 字节	-		闪存结束地址 (4 字节) 从第51 字节开始: 0xFF, 0xFF, 0x8F, 0x3F
	55 字节 ~ 56 字节	-		闪存程序块计数 (2 字节) 从第55 字节开始: 0x0A, 0x00
	57 字节 ~ 60字节	-		相同容量 (16K)闪存程序块组的开始地址 (4 字节) 从第57字节开始: 0x00, 0x00, 0x00, 0x00 TMPM364F10FG没有16KB程序块。
	61 字节 ~ 64 字节	-		相同容量 (16K)闪存程序块的容量 (半字) (4 字节) 从第61字节开始: 0x00, 0x00, 0x00, 0x00 TMPM364F10FG没有16KB程序块。
	65 字节	-		相同容量 (16K)闪存程序块的数量(1 字节) 0x00 TMPM364F10FG没有16KB程序块。
	66 字节 ~ 69 字节	-		相同容量 (32K)闪存程序块组的开始地址 (4 字节) 从第66字节开始: 0x00, 0x00, 0x8F, 0x3F
	70 字节 ~ 73 字节	-		相同容量 (32K)闪存程序块的容量 (半字) (4 字节) 从第70字节开始: 0x00, 0x40, 0x00, 0x00
	74 字节	-		相同容量 (32K)闪存程序块的数量(1字节) 0x02
	75 字节 ~ 78 字节	-		相同容量 (32K)闪存程序块组的开始地址 (4 字节) 从第75字节开始: 0x00, 0x00, 0x81, 0x3F
	79 字节 ~ 82 字节	-		相同容量 (32K)闪存程序块的容量 (半字) (4 字节) 从第79字节开始: 0x00, 0x80, 0x00, 0x00

表24-8 显示产品信息指令的转输格式

	字节	从控制器传输到 TMPM364F10FG的数据	波特率	从TMPM364F10FG传输到 控制器的数据
	83 字节	-		相同容量 (64K)闪存程序块的数量 (1 字节) 0x01
	84 字节 ~ 87 字节	-		相同容量 (128K)闪存程序块组的开始地址 (4 字节) 从第84字节开始: 0x00, 0x00, 0x82, 0x3F
	88 字节 ~ 91 字节	-		相同容量 (128K)闪存程序块的容量 (半字) (4 字节) 从第88字节开始: 0x00, 0x00, 0x01, 0x00
	92 字节	-		相同容量 (128K)闪存程序块的数量 (1 字节) 0x07
	93 字节	-		字节5 ~ 92的校验和值
	94 字节	(等待下一个指令码。)		-

注 1: **在I/O接口模式中，第1与第2字节的传输波特率必须是所要求波特率的1/16。**

注 2: **在任何否定应答的情况下，引导启动程序返回到其等待指令码 (第3字节)的状态。在I/O接口模式中，如果发生通信错误，则不出现否定应答。**

24.2.9.4 片擦除与保护位擦除

表24-9 芯片与保护位清除指令的传输格式

	字节	从控制器传输到 TMPM364F10FG的数据	波特率	从TMPM364F10FG传输到 控制器的数据
Boot ROM	1 字节	串行运行模式与波特率 对于UART模式 : 0x86 对于I/O接口模式 : 0x30	要求的波特率 (注1)	-
	2 字节	-		串行运行模式ACK字节 ・ 对于UART模式 - 正常应答 : 0x86 (如果波特率不能正确设置，则引导启动程序异常终止。) ・ 对于I/O接口模式 - 正常应答 : 0x30
	3 字节	指令码 (0x40)		-
	4 字节	-		指令码ACK字节 (注2) - 正常应答 : 0x10 - 否定应答 : 0xX1 - 通信错误 : 0xX8
	5 字节	芯片擦除指令码 (0x54)		-
	6 字节	-		指令码ACK字节 (注2) -正常应答 : 0x10 -否定应答 : 0xX1 -通信错误 : 0xX8
	7 字节	-		芯片擦除指令码ACK字节 -正常应答 : 0x4F -否定应答 : 0x4C
	8 字节	(等待下一个指令码。)		-

注 1 :**在 I/O 接口模式中，第1与第2字节的传输波特率必须是所要求波特率的1/16。**

注 2: **在任何否定应答的情况下，引导启动程序返回到其等待指令码 (第3字节)的状态。在 I/O 接口模式中，如果发生通信错误，则不出现否定应答。**

24.2.10 引导启动程序的运行

当选择单引导启动模式时，该引导启动程序自动地执行启动。引导启动程序提供了四个指令，其详细情况在以下分节中提供。

1. RAM传输指令

RAM传输指令储存从主机控制器输送到片上RAM的程序代码，并且在该传输一旦完成时执行该程序。用户程序RAM空间可分配到从0x2000_0400到RAM结束地址的范围，然而，引导启动程序区 (0x2000_0000 ~ 0x2000_03FF)则不可用。用户程序在所分配的RAM地址开始。

RAM传输指令可用于下载其本身的闪存编程例行程序;这就提供了以唯一的方式控制闪存在板编程的能力。该编程例行程序必须利用第24.3节中叙述的闪存指令序列。在启动一项传输之前，该RAM传输指令即对照储存于闪存中的密码顺序，对来自控制器的密码顺序进行验证。

注:**如果密码设置到0xFF (已擦除数据)，则由于容易猜到密码，难以安全地保护数据。即便不采用单引导启动模式，建议将唯一值设置为密码。**

2. 显示闪存SUM指令

该显示闪存总和指令增加了闪存的整个容量。引导启动程序不提供读出该闪存容量的指令。反而，该闪存总和指令可被用来进行软件修订管理。

3. 显示产品信息指令

显示产品信息指令提供了产品名称，片上存储器配置等等。该指令还读出如下所示地址的闪存位置的内容。除显示闪存总和指令之外，这些位置可被用来进行软件修订管理。

产品名称	区域
TMPM364F10FG	0x3F8F_FFF0 ~ 0x3F8F_FFF3

4. 闪存芯片擦除与保护位擦除指令

该指令自动地擦除闪存的整个区。即便在禁止写入与擦除任何程序块的情况下，存储器单元中的所有程序块及其保护条件均被擦除。在该指令完成时，FCSECBIT<SECBIT>位被设置为“1“。在用户忘记密码时，该指令用来恢复引导启动编程运行。因此不执行密码验证。

24.2.10.1 RAM 传输指令

该指令的传输格式见表24-6。

1. 第1字节规定了使用两个串行操作模式中的哪一个。如何确定串行操作模式的详细说明，见随后叙述的 “24.2.10.6串行操作模式的确定“。如果该模式被确定为UART模式，该引导启动程序检查是否可进行波特率设置。在第1字节处理期间，禁止接收操作。(SC4MOD0<RXE>=0)

   - 以UART模式进行通信

     第1字节被设置为 “0x86“，而且通过设置UART，以规定的波特率从控制器传输到目标板。如果串行操作模式被确定为UART，那么，引导启动程序则检查是否可进行波特率设置。如果该波特率不能设置，则引导启动程序异常终止，任何随后的通信则不能完成。有关判定波特率设置是否可能的方法，请参阅 “波特率设置“。

   - 以I/O接口模式进行通信

     将第1字节设置为 “0x30“，并通过同步设置，以所要求波特率的1/16，从控制器传输到目标板。和第1字节一样，所规定波特率的1/16用于第2传输。从第3字节 (工作指令数据)开始，用户可以规定的波特率发送数据。

     在I/O端口模式中，CPU把接收终端看作输入端口并监控I/O端口的电平。如果波特率高，或工作频率高，CPU则不可能辨别I/O端口的电平。要避免这种情况，在I/O端口中，将波特率设置在所要求波特率的1/16。在串行操作模式确定为I/O接口模式时，即设置了SCLK输入模式。控制器必须保证，其接收时间限制在选择波特率下得到满足。在I/O接口模式情况下，引导启动程序不检查接收错误标志;因而，也就没有错误应答 (位 3, 0x08)。

2. 第2字节，从目标板传输到控制器，是对设置串行操作模式的第1字节的应答。在第1字节被确定为UART，并且可设置在规定的波特率时，即传输数据 “0x86“。在第1字节被确定为 I/O接口时，即传输数据 “0x30“。

   - UART模式

     第2字节用于辨别是否可以设置波特率。如果可以设置波特率，则将SC4BRCE的值更新，并将数据 “0x86“发送到控制器。如果不能设置波特率，发送操作即停止，不传输任何数据。在第1字节的传输完成之后，控制器允许5秒钟的暂停。如果在允许的暂停期限内不接收0x86，控制器应放弃通信。在将0x86输入到SIO发送缓冲器之前，通过设置SC4MOD0<RXE>=1来允许接收操作。

   - I/O接口模式

     引导启动程序设置SC4MOD0与SC4CR寄存器的值，以配置I/O接口模式，并将0x30写为SC4BUF。然后，SIO4等待来自控制器的SCLK4信号。在第1字节的传输完成之后，控制器应在某个闲置时间 (几微妙)后将SCLK时钟脉冲发送到目标板。这必须在所要求波特率的1/16下完成。如果从目标板到控制器的第2字节为0x30，那么，该控制器将视其为通信可能。从第3字节开始，用户可以规定的波特率发送数据。通过在将0x86输入到SIO之前设置SC4MOD0<RXE>=1，即容许进行接收操作。

3. 从控制器传输到目标板的第3字节为指令。RAM传输指令的代码为0x10。

4. 第4字节，从目标板传输到控制器，是对第3字节的应答。在发送回应答之前，引导启动程序检查接收错误。如果有接收错误，则引导启动程序输送0xX8 (位3)，并返回到其再次等待指令 (第3字节)的状态。在这种情况下，应答的高位四字节不确定-保持与以前发出指令的高位四字节相同的值。在为I/O接口模式配置SIO0时，该引导启动程序不检查接收错误。

   如果第3字节相当于表24-4中所列的任何指令码，该引导启动程序则将其返回到控制器。在接收到RAM传输指令时，引导启动程序则回送0x10的值，然后转移到RAM 传输例行程序。一旦进行这一转移，密码验证即告完成。密码验证在随后的 “密码“一节中详述。如果第3字节不是有效指令，引导启动程序将0xX1 (位0)发送回控制器，并返回到其再次等待指令 (第3字节)的状态。在这种情况下，应答的高位四字节不确定–它们保持与以前发出指令的高位四字节相同的值。

5. 第5 ~ 第16字节，从控制器传输到目标板，为一个12-字节密码。各字节与闪存中以下地址的内容进行比较。验证从第5字节开始。如果密码验证失败，RAM传输例行程序设置密码错误标志。

产品名称	区域
TMPM364F10FG	0x3F8F_FFF4 ~ 0x3F8F_FFFF

6. 第17字节为密码顺序 (第5 ~ 第16字节)的校验和值。要计算12-字节密码的校验和值，彼此增加12字节，不考虑进位，并采用低位8位计算出8-位二补数，然后从控制器传输该校验和值。校验和计算在是随后的 “校验和计算“一节中详述。

7. 第18字节，从目标板传输到控制器，为第5 ~ 第17字节的应答。首先，RAM传输例行程序检查第5 ~ 第17字节中的接收错误。如果有接收错误，则引导启动程序发送回0x18 (位3)并返回到其再次等待指令 (如第3字节)的状态。在这种情况下，应答的高位四字节与以前发出指令 (如1)的值相同。在为 I/O接口模式配置SIO4时，RAM传输例行程序不检查接收错误。

   其次，RAM传输例行程序执行校验和操作，以确保第17字节数据的完整性。增加一系列第5 ~ 第16字节必须产生0x00 (进位丢掉)。如果出现校验和错误，RAM传输例行程序发送回0x11到控制器并返回到其再次等待指令 (如第3字节)的状态。

   最后，检查密码验证结果。如果产生以下情况，引导启动程序则输送应答 (位0, x11)，作为口令错误，并等待下一个工作指令 (第3字节)。

   - 不管密码比较的结果如何，闪存中密码的所有12个字节除了0xFF之外均为相同的值。
   - 并非从控制器传输的整个密码字节均与闪存中所含的相匹配。

   在所有上述验证成功时，RAM传输例行程序给控制器返回一个正常应答 (0x10)。

8. 第19 ~ 第22字节，从控制器传输到目标板，表明RAM区域的开始地址，在该区域应储存随后的数据 (比如闪存编程例行程序)。第19字节对应该地址的位31 ~ 24，第22字节对应该地址的位7 ~ 0。所储存RAM的开始地址必须是偶数地址。

9. 第23与第24字节，从控制器传输到目标板，表明将要从控制器输送储存在RAM中的字节数量。第23字节对应待输送字节数量的位15 ~ 8，第24字节对应字节数量的位7 ~ 0。

10. 第25字节为第19 ~ 第24字节的校验和值。要计算该校验和值，一同添加所有这些字节，不考虑进位，并采用低位8位计算8-位二补数，然后从控制器传输该校验和值。校验和计算在随后的 “24.2.10.9校验和计算“一节中详细叙述。

11. 第26字节，从目标板传输 ~ 控制器，为数据的第19 ~ 第25字节的应答。首先，RAM传输例行程序检查第19 ~ 第25字节中的接收错误。如果有接收错误，RAM传输例行程序发送回0x18并再次返回到指令等待状态 (比如第3字节)。在这种情况下，应答的高位四字节与以前发出指令 (如1)的值相同。在为输入/输出接口模式配置SIO4时，RAM传输例行程序不检查接收错误。

    其次，RAM传输例行程序执行校验和操作，以确保数据完整性。添加一系列第19 ~ 第24字节必须产生0x00 (进位丢掉)。在校验和错误情况下，RAM传输例行程序发送回0x11到控制器，并返回到其再次等待指令 (比如第3字节)的状态。

    - 第19 ~ 第25字节数据必须在0x2000_0400 ~ RAM结束地址的范围之内。

    在上述检查成功时，RAM传输例行程序给控制器返回一个正常应答 (0x10)。

12. 控制器的第27 ~ 第m字节储存于TMPM364F10FG的片上RAM中。储存始于第19 ~ 22字节规定的地址，并继续第23 ~ 第24字节规定的字节数量。

13. 第 (m+1)字节为校验和值。要计算该校验和值，一同添加第27 ~ 第m字节，不考虑进位并使用低位8位计算出8-位二补数，然后从控制器传输该校验和值。校验和计算在随后的“24.2.10.9校验和计算“一节中详细叙述。

14. 第 (m+2)字节为第27 ~ 第 (m+1)字节的应答。首先，RAM传输例行程序检查第27 ~ 第 (m+1)字节中的接收错误。如果有接收错误，则RAM传输例行程序发送回0x18 (位3)并返回到其再次等待指令(如第3字节)的状态。在这种情况下，应答的高位四字节与以前发出指令 (如1)的值相同。在为 I/O接口模式配置SIO4时，RAM传输例行程序不检查接收错误。

    其次，RAM传输例行程序执行校验和操作，以确保数据完整性。添加一系列第27 ~ 第 (m+1)字节必须产生0x00 (进位丢掉)。在校验和错误情况下，RAM传输例行程序发送回0x11 (位0)到控制器，并再次返回到指令等待状态 (比如第3字节)。在上述检查成功完成时，RAM传输例行程序给控制器返回一个正常应答 (0x10)。

15. 如果第 (m+2)字节为正常应答，则转移到第19 ~ 第22字节规定的地址。

24.2.10.2 显示闪存SUM指令

该指令的传输格式见表24-7。

1. 第1与第2字节的处理与RAM传输指令相同。
2. 第3字节为指令，由目标板从控制器接收。显示闪存总和指令的代码为0x20。
3. 第4字节，从目标板传输到控制器，是对第3字节的应答。在发送回应答之前，引导启动程序检查接收错误。如果有接收错误，则引导启动程序输送0xX8 (位3)，并再次返回到指令等待状态。在这种情况下，应答的高位四字节不确定-它们保持与以前发出指令的高位四字节相同的值。在为I/O接口模式配置SIO4时，该引导启动程序不检查接收错误。
   如果第3字节相当于表24-4中所列的任何指令码，该引导启动程序则将其返回到控制器。在接收显示闪存总和指令时，引导启动程序回送0x20的值，然后转移到显示闪存总和例行程序。如果第3字节不是有效指令，则引导启动程序发送回0xX1 (位0) 到控制器，并再次返回到指令等待状态 (第3字节)。在这种情况下，应答的高位四字节不确定-它们保持与以前发出指令的高位四字节相同的值。
4. 显示闪存总和例行程序一同添加闪存的所有字节。第5与第6字节，从目标板传输到控制器，分别表明该总和的高位与低位字节。总和计算的详情见"24.2.10.8显示闪存总和指令的计算"一节。
5. 第7字节为第5与第6字节的校验和值。要计算该校验和值，一同添加第5与第6字节，不考虑进位并使用低位8位计算出8-位二补数，然后从控制器传输该校验和值。
6. 第8字节为下一个指令码。

24.2.10.3 显示产品信息指令

该指令的传输格式见表24-8。

1. 第1与第2字节的处理与RAM传输指令相同。
2. 第3字节为指令，由目标板从控制器接收。显示产品信息指令的代码为0x30。
3. 第4字节，从目标板传输到控制器，是对第3字节的应答。在发送回应答之前，引导启动程序检查接收错误。如果有接收错误，则引导启动程序输送0xX8 (位 3)，并再次返回

到指令等待状态。在为输入/输出接口模式配置SIO4时，该引导启动程序不检查接收错误。

如果第3字节相当于表24-4中所列的任何指令码，该引导启动程序则将其返回到控制器。在接收到 “显示闪存求和“指令时，该引导程序回显一个值0x30，然后转移到 “显示闪存求和“例行程序上。如果第3字节不是有效指令，引导启动程序将0xX1 (位0)发送回控制器，并返回到其再次等待指令 (第3字节)的状态。在这种情况下，应答的高位四字节不确定-它们保持与以前发出指令的高位四字节相同的值。

4. 从目标板传输到控制器的第5～ 第8个字节，为从闪存中的以下所示的地址读取的数据。通过将软件ID储存于这些位置，可实现对软件版本的管理。

产品名称	面积
TMPM364F10FG	0x3F8F_FFF0 ~ 0x3F8F_FFF3

5. 位该目标板到该控制器的第9～20字节，表示以下ASCⅡ代码中的产品名称 (其中[ ]为空格)。

产品名称	核
TMPM364F10FG	T, M, P, M, 3, 6, 0, F, 1, _, [ ], _

6. 目标板传输到该控制器的第21～24字节表示密码闪存区的起始地址。各产品有自己的起始地址，如下所述。以下各值的传输均为从第从第21个字节开始，传输以下数值。

产品名称	地址
TMPM364F10FG	0xF4, 0xFF, 0x8F, 0x3F

7. 目标板传输到控制器的第25～28个字节表示片内RAM的起始地址。TMPM364F10FG本身带有起始地址 (如以下所示)。从第25个字节开始，传输以下数值。

产品名称	地址
TMPM364F10FG	0x00, 0x00, 0x00, 0x20

8. 目标板传输到控制器的第29～32字节为虚拟数据。自第29字节开始，传输0x00，0x00，0x00及0x00。

9. 目标板传输到控制器的第33～36字节，表示片内RAM的结束地址。TMPM364F10FG具备其自身的结束地址 (如以下所示)。自第33字节开始传输以下数值。

产品名称	地址
TMPM364F10FG	0xFF, 0xFF, 0x00, 0x20

10. 目标板到控制器的第37～40字节，为0x00，0x00，0x00及0x00。目标板传输到控制器的第41～44个字节，为0x00，0x00，0x00及0x00。

11. 所传输的第45与第46个字节为0x00，0x00。

12. 目标板传输到控制器的第47～50个字节表示片内闪存的起始地址，为0x00，0x00，0x80及0x3F。

产品名称	地址
TMPM364F10FG	0x00, 0x00, 0x80, 0x3F

13. 目标板传输到控制器的第51～54个字节，表示片内闪存的结束地址。各产品本身带有结束地址 (见下述)。自第51个字节开始传输以下数值。

产品名称	地址
TMPM364F10FG	0xFF, 0xFF, 0x8F, 0x3F

14. 目标板传输到控制器的第55～ 第56个字节，表示可用闪存块的数目。各产品均可传输其自身的数目 (如以下所示)。以下各值的传输均为从第55个字节开始。

产品概述	闪存块的数目
TMPM364F10FG	0x0A, 0x00

15. 目标板传输到控制器的第57～ 第92个字节，包含各闪存块的信息。相同规格的若干闪存块可作为一组接受处理。闪存块信息可指示某组的起始地址，该组所含各闪存块的规格(半字)，以及该组闪存块的数目。第57～ 第65个字节为16-K字节闪存块的相关信息。第66～ 第74个字节为32-K字节闪存块的相关信息。第75～ 第83个字节为64K字节闪存块的相关信息。第84～ 第92个字节为128-K字节闪存块的相关信息。有关所传输的字节的值，见表24-8。

16. 目标板传输到控制器的第93个字节，为第5～ 第92个字节的校验和值。如需计算该校验和值，则应将所有这些字节同时相加，不考虑进位，并利用下8-计算这两个8-位对象的余数。

17. 第94个字节为下一指令码。

24.2.10.4 片与保护位擦除指令

有关该指令的传输格式，见表24-9。

1. 第1与第2字节的处理与RAM传输指令相同。
2. 从该控制器到TMPM364F10FG

   目标板收自该控制器的第3个字节属于指令。“显示产品资料“指令代码为0x40。
3. 从TMPM364F10FG到该控制器

   目标板传输到控制器的第4字节，是对第3字节的确认响应。
   在发回该确认响应之前，该引导程序会检查是否存在接收错误。如果存在接收错误，则该引导程序会传输0xX8 (位 3)，并重新返回到指令等待状态。在这种情况下，应答的高位四字节不确定-它们保持与以前发出指令的高位四字节相同的值。

   如果第3字节相当于表24-4中所列的任何指令码，该引导启动程序则将其返回到控制器。在接收到 “显示闪存求和“指令时，该引导程序会返回一个值，即0x40。如果第3字节不是有效指令，引导启动程序将0xX1 (位0)发送回控制器，并返回到其再次等待指令 (第3字节)的状态。在这种情况下，应答的高位四字节不确定-它们保持与以前发出指令的高位四字节相同的值。
4. 从该控制器到TMPM364F10FG

   目标板传输到控制器的第5个字节，是“芯片擦除启动“命令码 (0x54)。
5. 从TMPM364F10FG到该控制器

   目标板传输到控制器的第6字节，是对第5字节的确认响应。
   在发回该确认响应之前，该引导程序会检查是否存在接收错误。如果存在接收错误，则该引导程序会传输0xX8 (位 3)，并重新返回到指令等待状态。在这种情况下，应答的高位四字节不确定-它们保持与以前发出指令的高位四字节相同的值。

   如果第5个字节等于拟启动擦除的任何操作码，则该引导程序会将其返回到该控制器。在收到 “芯片与保护擦除命令“时，该引导程序会返回一个值即0x54，然后转移到该“芯片擦除“例行程序。如果该第5个字节不是有效命令，则该引导程序会将0xX1 (位0)发回控制器，并重新返回到等待指令状态 (第三字节)。在这种情况下，应答的高位四字节不确定-它们保持与以前发出指令的高位四字节相同的值。
6. 从TMPM364F10FG到控制器

   第7个字节表示该“芯片擦除命令“是否正常完成。

   如已正常完成，则发送完成码 (0x4F)。

   如检测到一个错误，则发送错误代码 (0x4C)。

7. 第9字节为下一指令码。

24.2.10.5 应答响应

该引导程序可用具体代码说明处理状态。表24-10中给出了对所接收数据的可能确认响应的值。该确认响应的上四位字节，等于正在被执行的命令的字节。第3位表示接收错误。第零位表示无效命令错误，校验和错误或密码错误。第1位与第2位始终为 “0“。不应在输入/输出接口模式下接收误差校验。

表24-10 对串行操作模式位组的ACK响应

返回值	含义
0x86	SIO经过配置后，可按UART模式操作。参见注释
0x30	该SIO经过配置后，可按 I/O 接口模式操作。

注:**在UART模式下，如果无法设定波特率设置，则会停止通信，且无任何响应。**

表24-11 对该指令字节的ACK响应

返回值	含义
0x?8 (见注释)	在接收某个指令码期间发生接收错误。
0x?1 (见注释)	接收到一个未定义的指令码 (接收正常完成。)
0x10	接收到RAM传输指令。
0x20	接收到示闪存求和指令。
0x30	接收到显示产品资料指令。
0x40	接收到芯片擦除指令。

注:**该确认响应的上四位位，与先前的命令码的相同。**

表24-12 对校验和位组的ACK响应

返回值	含义
0xN8 (见注释)	出现一个接收错误。
0xN1 (见注释)	出现一个校验和或密码错误。
0xN0 (见注释)	校验和正确。

注: **确认响应的上四位字节，与该操作指令码的相同。例如，在出现密码错误时，其为1 (N；RAM传输指令数据 [7:4])。**

表24-13 对芯片与保护位擦除字节的确认响应

返回值	含义
0x54	接收到芯片擦除启动指令。
0x4F	芯片擦除已完成。
0x4C	芯片擦除的完成反常。

24.2.10.6 串行操作模式确定

串行操作模式由来自该控制器的第一个字节决定。如需将UART模式用于该控制器与该目标板之间的通信，该控制器必须首先以所要求的波特率，将一个值0x86发送到该目标板。如使用I/O接口模式，则控制器必须按所要求波特率的1/16，发送一个值即0x30。在图24-4中，给出了第一个字节在各模式下的波形。

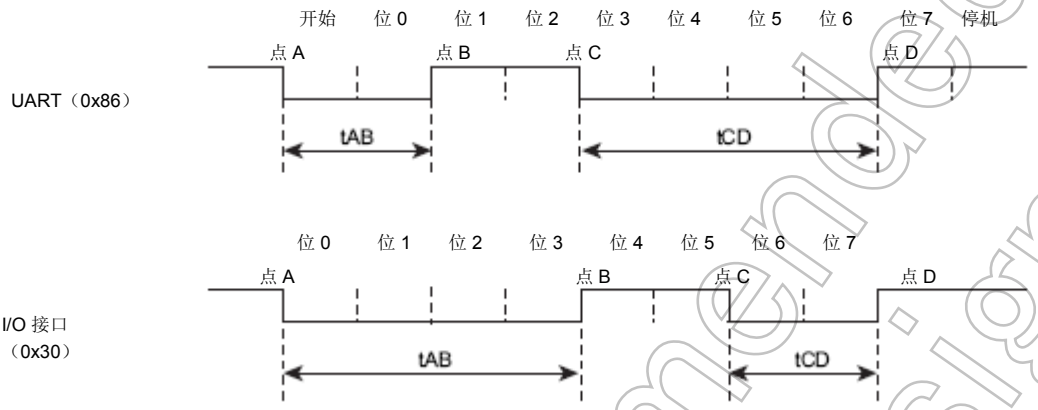


图24-4 串行操作模式字节

在 $\overline{\text{RESET}}$ 被解除后，该引导程序会监视来自该控制器的第一个串行字节 (SIO接收被禁用)，并计算tAB，tAC与tAD的间隔。在图24-5中，给出了一份描述tAB，tAC与tAD间隔确定步骤的流程图。如该流程图所示，第一个串行字节中每发生二次逻辑转换，该引导程序就捕获一次计时器计数。因此，经计算得出的tAB，tAC与tAD时间间隔很可能仅存在微小误差。如果该传输是以高波特率进行的，则该中央处理器可能无法跟上该串行接收引脚发生的逻辑转换的速度。尤其是输入/输出接口模式可能会存在这一问题，原因是其波特率一般均远高于UART模式的波特率。为避免出现这种情况，该控制器应按所要求波特率的1/16发送第一个串行字节。

在图24-5所给出的流程图中，对该引导程序辨别UART模式与I/O接口模式的方法进行了说明。如果tAB的长度等于或小于tCD的长度，则会确定串行操作模式采用UART模式。如果tAB的长度大于tCD的长度，则会确定串行操作模式采用I/O接口模式。注意，如果该波特率过高或计时器工作频率过低，则各计时器值会变小。这可能导致该控制器发生非故意的动作。为防止发生该问题，可在编制程序内部复位UART模式。

例如，在预定的模式为UART模式时，可将串行操作模式确定为I/O接口模式。为避免出现这种情况，在运用UART模式时，该控制器应留有超时周期，在该超时周期内，其可接收来自该目标板的一个回波(0x86)。如果该控制器未能在该容许时之内收到该回波，则其会中止该通信过程。在使用I/O接口模式时，一旦传输了第一个串行字节，则该控制器即应在某一空闲时间消逝后发送该SCLK时钟，并获得确认响应。如果所收到的确认响应不是0x30，则该控制器会中止下一步的通信。

在预定的模式是I/O接口模式时，只要tAB大于tCD，则第一个字节就不必是0x30 (如上所述)。可将0x91，0xA1或0xB1作为第一字节代码发送，以确定点A与点C的下降沿，以及点B与点D的上升沿。如果tAB大于tCD，且在进行操作模式确定时决定选择SIO接口，则尽管第一个字节上的传输代码不是0x30 (用于确定I/O接口模式的第一字节代码，被描述为0x30)，第二字节代码仍为0x30。

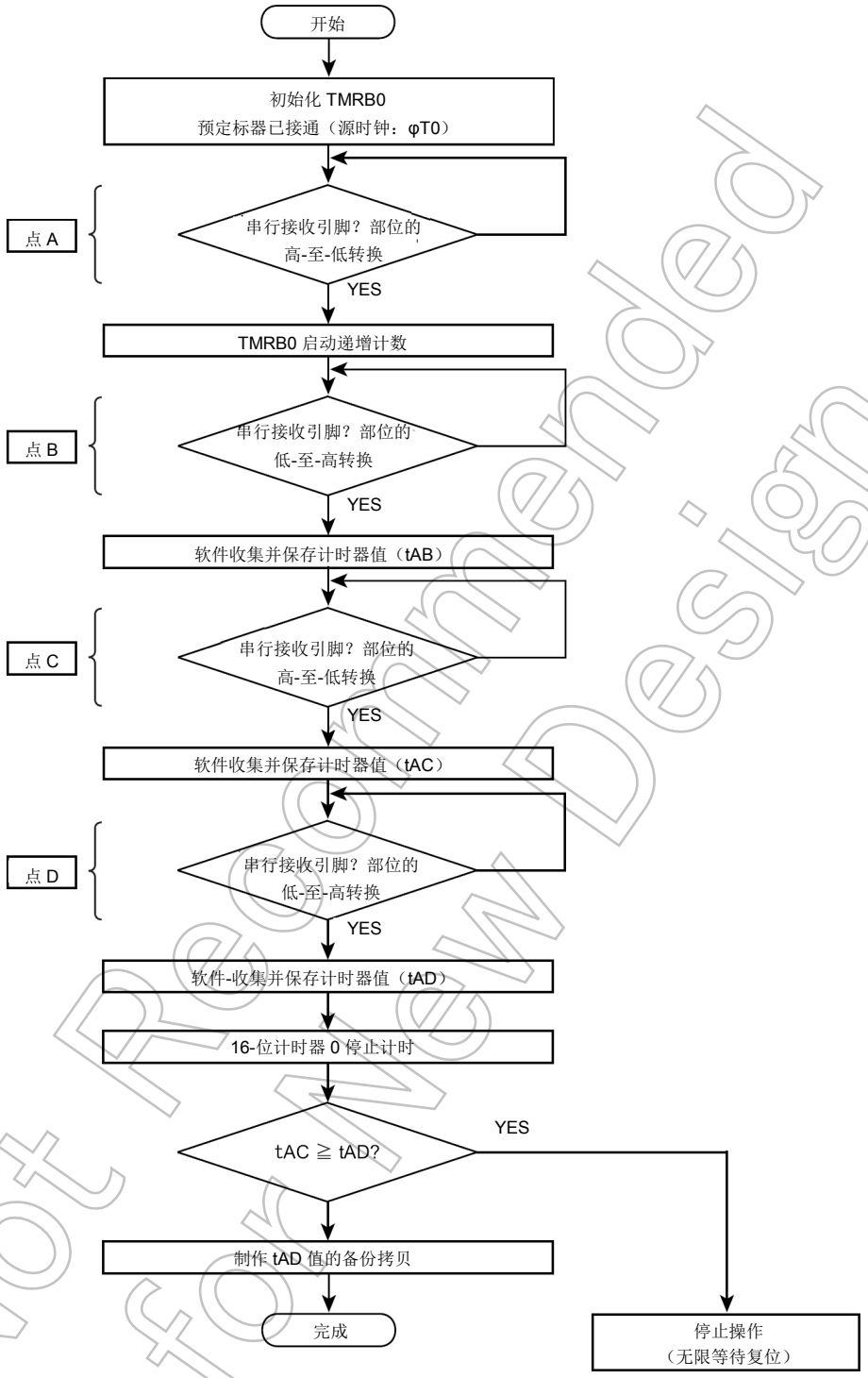


图24-5 串行操作模式字节接收流程图

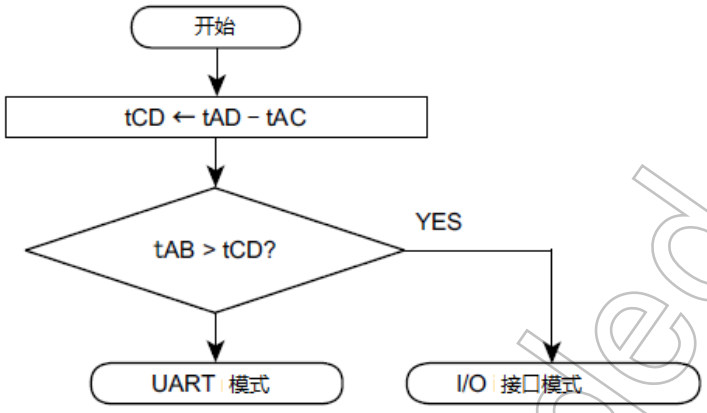


图24-6 串行操作模式确定流程图

24.2.10.7 密码

该RAM传输指令 (0x10)，可导致该引导程序执行密码验证。在该操作码回波后，该引导程序会对该闪存内部的12字节密码的内容进行核实。下表列出了各产品的密码区。

产品名称	面积
TMPM364F10FG	0x3F8F_FFF4 ~ 0x3F8F_FFFF

注: 如果密码被设置为0xFF (已擦除数据区域)，则可因密码易于猜出而难以可靠地保护数据。即便不采用单引导启动模式，建议将唯一值设置为密码。

如果所有这些地址位置均包含相同的数据字节 (除了0xFF)，则可发生密码区错误 (如图24-7所示)。在这种情况下，无论发自该控制器的密码序列是否为全 0xFFs，该引导程序都会根据该校验和字节 (第17字节)的情况，返回一个错误确认 (0x11)。

会将来自该控制器的接收数据 (第5 ~ 第16字节)，与存储在闪存中的密码进行比较。所有12字节均必须匹配，才可以通过该密码验证。否则会发生密码错误，从而导致该引导程序根据该校验和字节 (第17字节)对错误确认进行答复。

即使安全功能已启动，也会进行该密码验证。

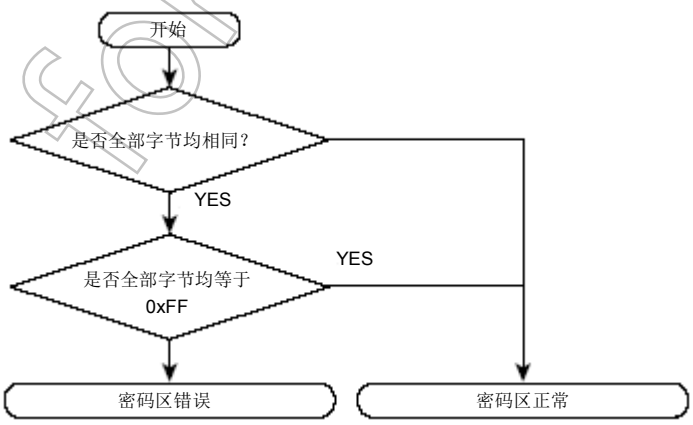


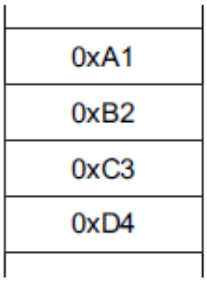
图24-7 密码区验证流程图

24.2.10.8 显示闪速存储器和指令的计算

由一个半字数量响应该求和计算 (“字节 +字节 + 字节 + ・・・“)的结果。该“显示闪存求和

“命令会将该闪存的所有512 K字节相加在一起，并以半字数量形式提供该总和。该和被发送到该控制器，首先是上八位，接着是下八位。

示例）



为简洁起见，假定该闪存的深度为四个单元。则该四个字节之和可计算如下:

0xA1 + 0xB2 + 0xC3 + 0xD4 = 0x02EA

因此，首先将 0x02 发送到该控制器，接着发送 0xEA。

24.2.10.9 和校验计算

不考虑进位，通过将各字节相加在一起，即可计算出一连串数据字节的校验和字节，并用下8位计算这两个8-位对象的余数。该 “显示闪存求和“指令与“显示产品资料“指令执行该校验和计算。该控制器必须在传输校验和字节时，执行相同的校验和运算。

示例)假定该 “显示闪存求和“指令以0xE5和0xF6的形式提供该和的上下字节。一连串0xE5与0xF6校验和的计算如以下所示:

将各字节相加

0xE5 + 0xF6 = 0x1DB

利用下8位计算这两个对象的余数 (其为校验和字节)。然后将0x25发送到该控制器。

0 - 0xDB = 0x25

24.2.11 一般引导程序流程图

图24-8给出了该引导程序的总体流程图。

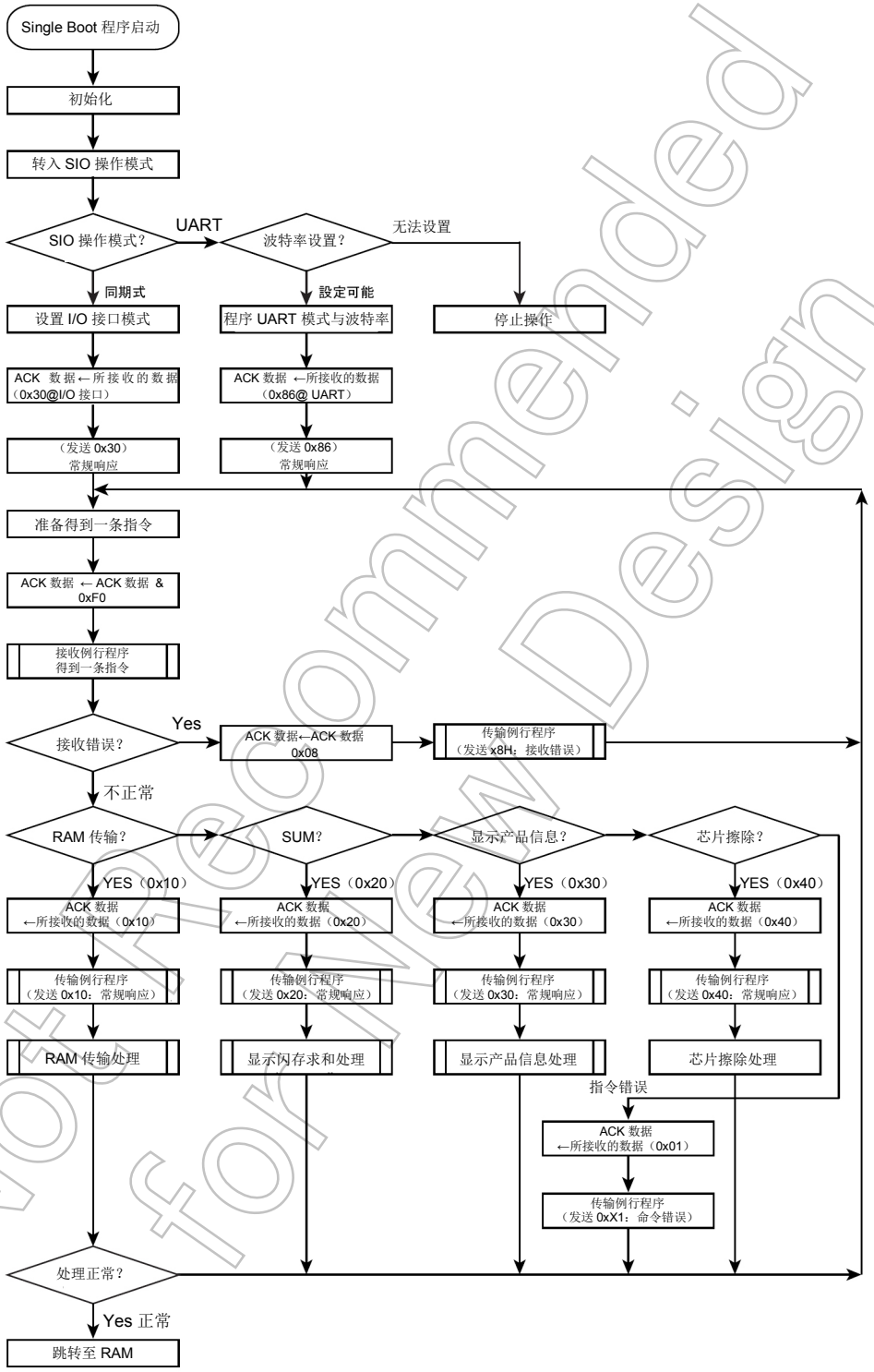


图24-8 总体引导程序流程图

24.3 闪存的板上编程 (重写/擦除)

在板上编程中，CPU应执行重写或擦除该闪存的软件命令。该重写/擦除控制程序应由用户事先编制。由于在闪存被写入或擦除期间无法读取其内容物，需在转换到用户引导模式之后，通过内部RAM操作该重写/擦除程序。

24.3.1 闪存

除了某些功能外，闪存数据的写入与擦除应按标准JEDEC 指令进行。

在写入或擦除过程中，应使用该中央处理器的数据传送指令将各指令输入到该闪存。一旦输入指令，实际写入或擦除操作就会在内部自动进行。

表24-14 闪存功能

主要功能	说明
自动页程序	自动将数据写入每页。
自动芯片擦除	自动擦除该闪存的整个区域。
自动闪存块擦除	自动擦除所选定的某闪存块。
保护功能	可针对各闪存块单独禁止写入或擦除操作。

24.3.1.1 程序块配置

(1)TMPM364F10FG

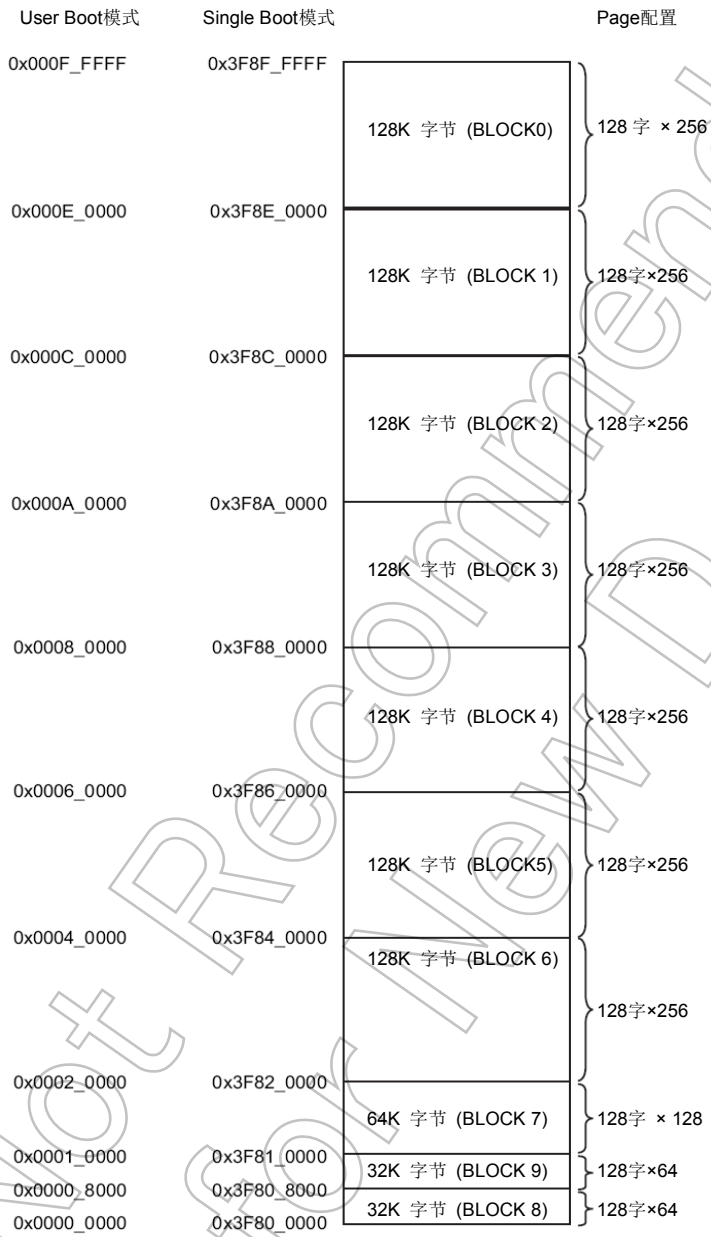


图24-9 闪存 (TMPM364F10FG)的闪存块配置

24.3.1.2 基本操作

该闪存装置具备以下两种操作模式:

- 读取存储器数据模式 (读取模式)
- 自动擦除或重写存储器数据模式 (自动操作)

在其处于存储器读取模式期间，通过执行一个命令序列，即可转换到该自动模式。在自动操作模式下，无法读取闪存数据，也无法执行存储在该闪存中的任何指令。在自动操作模式下，任何中断或异常生成均不能将该装置设置到读取模式，但已生成硬件复位的情形除外。在自动操作期间，务必不要引发任何异常，但调试端口已连接时的复位与调试异常除外。任何异常生成均不能将该装置设置到读取模式，但已生成硬件复位的情形除外。

(1) 读取

在读取数据时，必须将闪存设置为读取模式。在中央处理器复位被取消，或自动操作被正常终止时，应在加电之后立即将该闪存设置为读取模式。为从其它模式返回该读取模式，或在自动操作已经异常终止之后，应使用读取/复位命令 (属于软件指令，后文将述及)或硬件复位。在即将执行写在该闪存上的任何指令时，该装置也必须处于读取模式。

- 读取 / 复位指令与读取指令 (软件复位)

  在使用标识读取指令时，该读取操作会被终止，而不是自动返回到读取模式。在这种情况下，可使用该读取/复位指令使该闪存返回到读取模式。在必须取消某条尚未完全写入的指令时，也必须使用该读取/复位指令。该读取指令用于在执行32位数据传送指令以将该数据 “0x0000_00F0“写入到该闪存的任意地址之后，返回到读取模式。

- 利用该读取/复位指令，可使该装置可在完成第三个总线写入周期之后，返回到读取模式。

(2) 指令写入

该闪存使用该指令控制方法。通过对该闪存执行一个指令序列，即可执行各条指令。该闪存可根据所应用的该地址与数据组合执行自动操作指令 (请参看“指令序列“)。

如果必须取消进行中的某项指令写入操作，或在已输入任何不正确的指令序列时，可执行该读取/复位指令。然后，该闪存可终止该指令的执行，并返回到读取模式。

而指令一般由若干总线周期组成，对该闪存应用的32-位 (字)数据传输指令操作被称作“总线写入周期“。该总线写入周期具备一个特定的连续次序，在总线写入周期数据与指令写入地址的操作符合预定义的特定顺序时，该闪存可执行一次自动操作。如果任何总线写入周期不符合某一预定义的指令写入序列，则该闪存可终止该指令执行，并返回到读取模式。

注 1: **指令序列是从该闪存区外部执行的。**

注 2: **各总线写入周期必须通过32-位数据传输命令连续执行。在指令序列被执行期间，禁止对该闪存进行存取操作。也不要生成任何中断 (已连接调试端口时的调试异常除外)。如果进行了如此操作，可导致对该闪存的意外阅读存取，而该指令序列发生器不能正确识别**

该指令。而其可导致该指令序列的异常终止，还可导致错误识别该指令。

注 3: **为确保该指令序列发生器可识别某条指令，该装置在执行该指令之前必须处于读取模式。如果 FCFLCS <RDY / BSY>被设置为 "1"，则务必在第一个总线写入周期之前进行检查。建议随后执行读取指令。**

注 4: **在发布某条指令时，如果任何地址或数据写入不正确，则务必执行软件复位，重新返回到读取模式。**

24.3.1.3 复位 (硬件复位)

硬件复位用于在自动编程/擦除期间或自动操作过程中出现异常终止而被强制终止时，取消由该指令写入操作设置的操作模式。

该闪存具备一个复位输入存储块，该复位输入与该CPU复位信号连接。因此，在该装置的 $\overline{\text{RESET}}$ 引脚被设置为VIL时，或在CPU因监视时钟的任何溢出而导致复位时，该闪存可返回到读取模式，并终止可能处于进行中的任何自动操作。还应注意的是，在自动操作期间应用硬件复位可导致数据的不正确重写。在这种情况下，务必重新执行重写。

有关CPU复位操作，请参看 "1.2.1 复位操作"一节。在给定的复位输入结束后，该CPU可从该闪存读取向量数据，并在该复位被取消后启动操作。

24.3.1.4 指令

(1) 自动Page编程

写入到闪存装置，就是将 "1"数据单元改为 "0"数据单元。任何 "0"数据单元均无法被改为 "1"数据单元。为将 "0"数据单元改为 "1"数据单元，需执行一次擦除操作。

利用该装置的自动页编程功能，可将数据写入到各页。该TMPM364F10FG每页包含128个字。128字闪存块由同一[31:9]地址定义。其从地址[8:0] = 0x00开始，并在地址[8:0] = 0x1FF结束。该编程装置在下文中称为 "页"。

"写入到数据单元"由内部序列发生器自动执行，CPU无需进行任何外部控制。通过 FCFLCS[0]<RDY/BSY>，可检查自动页编程的状态 (其是否处于写入操作)。

在处于自动页编程模式时，任何新的指令序列均不会被接受。如果必须中断该自动页编程，则可使用该硬件复位功能。如果该操作被硬件复位操作终止，则需在擦除该页后，立即重新执行该自动页编程，原因是 "写入到该页"尚未被正常终止。

仅允许在已经擦除某页后，立即执行自动页编程操作。任何编程均不得执行两次或两次以上。注意，在对曾被写入的某页执行重写之前，必须执行自动闪存块擦除或自动芯片擦除指令，然后才能重新执行该自动页编程指令。注意，在未擦除内容的情况下试图对某页写入两次或两次以上，可导致装置损伤。

不会以内部方式对该装置执行自动验证操作。因此，务必读取已编程的数据，以确认写入正确。在该指令周期的第三个总线写入周期结束时，该自动页编程操作启动。在第五个总线写入周期结束之后，数据将被连续写入，该写入操作是从第四个总线写入周期中所规定地址的下一个地址开始的 (在第四个总线写入周期中，页顶端地址将被命令写入) (数据的32位会被同时输入)。

务必在第四个总线周期结束之后，以书面命令形式使用该32-位数据传送指令。此时，不应将任

何32-位数据传送指令置于跨字边界处。在第五个总线写入周期结束之后，数据会被命令写入到同一页区。即使必须仅部分写入该页，仍需为该整个页执行自动页编程。在这种情况下，应将第四个总线写入周期的地址输入设置为该页的顶端地址。务必执行指令写入操作，并针对该数据单元将输入数据设置为 “1“，而不是 “0“。例如，如果某页的顶端地址不允许写入，则可以指令写入形式，将第四个总线写入周期中的输入数据设置为0xFFFFFFFF。

一旦执行了第三个总线周期，自动页编程即开始运行。通过监视 FCFLCS<RDY / BSY>，可检查该情况。在处于自动页编程模式时，任何新的指令序列均不会被接受。如果必须停止操作，可使用该硬件复位功能。请谨慎进行该项操作，原因是如果该操作被中断，将无法正常写入数据。在单个页已被命令写入，并正常终止了自动页写入过程时，应将FCFLCS<RDY / BSY> 设置为 “1“，然后其返回该读取模式。

在拟写入多个方页时，需对各页执行该页编程指令，原因是通过该页编程指令的单次执行写入的页数目，仅限一个页。自动页编程被禁止跨页处理输入数据。

无法将数据写入到受保护的闪存块。在自动编程结束后，其会自动返回到读取模式。通过监视 FCFLCS<RDY / BSY>，可检查该情况。如果自动编程失败，则该闪存会被锁定在当前模式，且不会返回到读取模式。如欲返回该读取模式，需执行硬件复位以复位该闪存或该装置。在这种情况下，如果写入到该地址也已失败，则建议不使用该装置，或不使用包含该失败地址的该闪存块。

注:**在该自动PAGE编程指令的第四个总线写入周期结束之后，软件复位变为无效。**

(2) 自动芯片擦除

在该指令周期的第六个总线写入周期结束时，该自动芯片擦除操作启动。

通过监视 FCFLCS<RDY / BSY>，可检查该情况。在未以内部方式对该装置执行自动验证操作期间，务必读取该数据，以确认数据已被正确擦除。在处于自动芯片擦除操作模式时，任何新的指令序列均不会被接受。如果必须停止操作，可使用该硬件复位功能。如果不得不停止该操作，则需重新执行自动芯片擦除操作，原因是该数据擦除操作尚未被正常终止。

任何受保护的闪存块均无法擦除。如果所有均处于受保护状态，则不会执行该自动芯片擦除操作，其会在完成该指令序列的第六个总线读取周期之后，返回到读取模式。在某个自动芯片擦除操作被正常终止时，其会自动返回到读取模式。如果某次自动芯片擦除操作失败，则该闪存会被锁定在当前模式，且不会返回到读取模式。

如欲返回该读取模式，需执行硬件复位以复位该装置。在这种情况下，无法检测到发生故障的该闪存块。建议不再使用该装置，也可利用该闪存块擦除功能确定已发生故障的闪存块，以确保不再使用已确定的闪存块。

(3) 自动闪存块擦除 (适用于各闪存块)

在该指令周期的第六个总线写入周期结束时，该自动闪存块擦除操作启动。

通过监视FCFLCS <RDY / BSY>，可检查该自动闪存块擦除操作的状态。在未以内部方式对该装置执行自动验证操作期间，务必读取该数据，以确认数据已被正确擦除。处于自动闪存块擦除操作模式时，任何新的指令序列均不会被接受。如果必须停止操作，可使用该硬件复位功能。

在这种情况下，需重新执行该自动闪存块擦除操作，原因是该数据擦除操作尚未被正常终止。

任何受保护的闪存块均无法擦除。如果某次自动闪存块擦除操作失败，则该闪存会被锁定在该模式，且不会返回到读取模式。在这种情况下，可执行硬件复位，以复位该装置。

(4) 保护位的自动编程 (适用于各闪存块)

该装置在执行时具备保护位。可为各闪存块设置该保护。有关保护位地址表，见表24-18。该装置将1位作为保护位分配给1个闪存块。该适用保护位由第七个总线写入周期中的PBA中指定。通过对该保护位进行自动编程，可针对各闪存块单独禁止写入和/或擦除功能 (旨在提供保护)。可利用后文将述及的FCFLCS<BLPRO>检查各闪存块的保护状态。通过监视FCFLCS <RDY / BSY>,可检查用于设置保护位的自动编程操作的状态。在自动编程正在对保护位进行编程期间，任何新的指令序列均不会被接受。如果必须停止该编程操作，可使用该硬件复位功能。在这种情况下，需重新执行该编程操作，原因是保护位的编程未必正确。如果所有保护位均已经编程，则所有FCFLCS <BLPRO>都会被设置为 “1“，表示其处于受保护状态。这样，将禁用针对所有闪存块的后续写入与擦除操作。

注: **在该自动保护位编程指令的第七个总线写入周期内，软件复位无效。在输入第七个总线写入周期之后，FCFLCS <RDY/BSY>变为 “0“。**

(5) 保护位的自动擦除

在执行该自动保护位擦除指令时，所得到的结果可能有所不同，具体结果取决于该保护位与该安全位的状态。在FCSECBIT<FCSECBIT>被设置为 “1“时，这取决于该FCFLCS寄存器中的所有〈BLPRO〉是否均被设置为 “1“。在执行该自动保护位擦除指令之前，务必检查FCFLCS <BLPRO>的值。有关详细资料，见 “保护/安全功能“一章。

- 在所有FCFLCS <BLPRO>均被设置为 “1“时 (所有保护位均已编程):
  若自动保护位擦除指令为写入指令，该闪存可在该装置内部自动初始化。在第七个总线写入周期结束时，该闪存数据单元的整个区域均被擦除，然后各保护位也被擦除。通过监视 FCFLCS<RDY / BSY>，可检查该操作。如果该保护位擦除自动操作被正常终止，则FCFLCS会被设置为 “0x00000001“。由于未以内部方式对该装置执行自动验证操作，因此，请务必读取该数据，以确认其已被正确擦除。在第七个总线周期进行完毕之后，如需在自动操作期间返回到该读取模式，则需使用该硬件复位，以复位该装置。如该操作成功完成，则需在重调揩到读取模式之后，通过FCFLCS<BLPRO>检查各保护位的状态，并根据需要执行自动保护位擦除，自动芯片擦除，或自动闪存块擦除操作。

- 在FCFLCS <BLPRO>包含有 “0“时 (并非所有的保护位均已编程):

  如果该自动保护位被清除为 “0“，则保护状态即被取消。利用该装置，可将保护位编程到单独的闪存块，并在四位字节单元中执行位-擦除操作 (如表24-18所述)。目标位按规定位于第七个总线写入周期。可利用后文将述及的FCFLCS〈BLPRO〉检查各闪存块的保护状态。通过监视FCFLCS <RDY / BSY>，可检查自动保护位编程操作的状态。在旨在擦除保护位的自动操作被正常终止时，经选定拟用于擦除的 FCFLCS <BLPRO>的各保护位均被设置为 “0“。

无论何种情况，任何一个新的命令时序当其在自动操作时擦除保护位时都不会被接收。需要使用硬件复位功能停止其操作。当自动擦除保护位的操作被正常终止时，它将返回读取模式。

注:**在自动操作时FCFLCS <RDY / BSY>位为“0“，当自动操作被终止时，其回到“1“。**

(6) ID-读取

使用ID-读取指令，可获得设备内的闪存类型和其它信息。加载的数据将因第四和随后的总线写入循环的地址[15:14]不同而不同 (推荐的输入数据为0x00)。在第四总线写入循环之后，当一任意闪存区被读取时，ID值将被加载。当ID-读取命令的第四总线写入循环通过后，设备不会自动返回到读取模式。在此条件下，第四总线写入循环和ID-读取指令可重复执行。利用读取/复位命令或硬件复位命令返回读取模式。

24.3.1.5 闪存控制/状态寄存器

基址 = 0x41FF_F000

寄存器名称		地址 (基+)
保留	-	0x0000
保留	-	0x0004
安全位寄存器	FCSECBIT	0x0010
闪速控制寄存器	FCFLCS	0x0020
保留	-	0x0024
保留	-	0x0028

注:不得访问保留地址。

(1) FCFLCS (闪速控制寄存器)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	BLPRO9	BLPRO8
复位后	0	0	0	0	0	0	(注2)	(注2)
	23	22	21	20	19	18	17	16
比特符号	BLPRO7	BLPRO6	BLPRO5	BLPRO4	BLPRO3	BLPRO2	BLPRO1	BLPRO0
复位后	(注2)	(注2)	(注2)	(注2)	(注2)	(注2)	(注2)	(注2)
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	-	-	-	-	-	-	-	RDY / BSY
复位后	0	0	0	0	0	0	0	1

位	比特符号	型号	功能
31 ~ 26	−	R	读作 0。
25 ~ 16	BLPRO9 ~ BLPRO0	R	对块9 ~ 0的保护 0: 禁用 1: 启用 每一个保护位代表相应块的保护状态。当一个位被设置为 “1“时，表示对应于此位的块被保护。当块被保护时，不能向其写入数据。
15 ~ 1	−	R	读作 0。
0	RDY/BSY	R	准备 / 忙碌 (注1) 0:自动操作 1:自动操作终止。 准备/忙碌标志位 RDY/BSY输出是作为监控自动操作状态的一种手段。此位为CPU监控该功能的一个功能位。当闪存在自动操作时，它输出 “0“来表示它忙碌。当自动操作被终止时，返回准备状态并输出 “1“来接受下一指令。当自动操作执行失败时，此位保持 “0“输出。硬件复位返回 “1“。

注 1: **该命令必须在准备状态下发出。在忙碌状态下发出此命令会禁用正确的指令传输和下一个的命令输入。要退出此条件，复位系统。系统复位要求至少0.5 μs的时间，不管系统时钟频率如何。在这一条件下，在复位之后需要大约 2 ms 启用读取。**

注 2: **数值将因施加的保护不同而不同。**

(2) FCSECBIT (安全位寄存器)

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

位	比特符号	型号	功能
31-1	−	R	读作 0。
0	SECBIT	R/W	安全位 0:禁用 1:启用

注:**通过冷复位初始化寄存器。**

24.3.1.6 指令序列表

表24-15 显示每一闪存地址和命令数据。

除读取命令的第二个总线循环，读取/复位命令的第四个总线循环及ID-读取命令的第五个循环外，总线周期为 “总线写入循环“。总线写入循环由32-位 (字组)数据传输命令来执行。 (在下面表格中，只有较低的8-位数据被显示。)

获取地址位配置的详细情况见表24-16。使用表24-15中的 “Addr.“数值来对应表24-16中的常规命令的地址[15:8]。

注:**在整个总线周期中，对于地址位[1:0]始终设置 “0“。**

表24-15 来自内部CPU的闪存访问

命令序列	第一总线周期	第二总线周期	第三总线周期	第四总线周期	第五总线周期	第六总线周期	第七总线周期
	地址	地址	地址	地址	地址	地址	地址
	数据	数据	数据	数据	数据	数据	数据
读取	0xXX	–	–	–	–	–	–
	0xF0	–	–	–	–	–	–
读取 / 复位	0x54XX	0xAAXX	0x54XX	RA	–	–	–
	0xAA	0x55	0xF0	RD	–	–	–
ID-读取	0x54XX	0xAAXX	0x54XX	IA	0xXX	–	–
	0xAA	0x55	0x90	0x00	ID	–	–
自动页编程	0x54XX	0xAAXX	0x54XX	PA	PA	PA	PA
	0xAA	0x55	0xA0	PD0	PD1	PD2	PD3
自动芯片擦除	0x54XX	0xAAXX	0x54XX	0x54XX	0xAAXX	0x54XX	–
	0xAA	0x55	0x80	0xAA	0x55	0x10	–
自动块擦除	0x54XX	0xAAXX	0x54XX	0x54XX	0xAAXX	BA	–
	0xAA	0x55	0x80	0xAA	0x55	0x30	–
保护位编程	0x54XX	0xAAXX	0x54XX	0x54XX	0xAAXX	0x54XX	PBA
	0xAA	0x55	0x9A	0xAA	0x55	0x9A	0x9A
保护位擦除	0x54XX	0xAAXX	0x54XX	0x54XX	0xAAXX	0x54XX	PBA
	0xAA	0x55	0x6A	0xAA	0x55	0x6A	0x6A

补充说明

- RA: 读取地址
- RD: 读取数据
- IA: ID 地址
- ID: ID 数据
- PA: 程序页地址

  PD: 程序数据 (32位数据)

  在第四总线周期后，以地址次序来为页输入数据
- BA: 块地址
- PBA: 保护位地址

24.3.2 对总线写入周期的地址位配置。

表24-16与 “表24-15来自内部CPU的闪存访问“一同使用。

地址设置可依据第一总线周期的正常总线写入周期地址配置来操作。针对表24-16地址位配置，建议“0“为改变总线写入周期的必要操作。

地址	Addr [31:20]	Addr [19]	Addr [18]	Addr [17]	Addr [16]	Addr [15]	Addr [14]	Addr [13:11]	Addr [10]	Addr [9]	Addr [8]	Addr [7:0]
正常命令	正常总线写入循环地址配置											
	闪存区域	推荐 “0“				命令						Addr[1:0]= “0“ (固定) 其他:0 (建议)
ID-读取	IA: ID地址 (为ID读取操作设置第四总线写入周期地址)											
	闪存区域	推荐 “0“				ID地址		Addr [1:0]= “0“ (固定)，其他:0 (建议)				
块擦除	BA: 块地址 (为ID读取操作设置第六总线写入周期地址)											
	块选择(表24-16)							Addr [1:0]= “0“ (固定)，其他:0 (建议)				
自动页编程	PA: 程序页地址 (为页编程操作设置第四总线写入周期地址)											
	页选择											Addr [1:0]= “0 “ (固定) 其他:0 (建议)
保护位编程	PBA:保护位地址 (为保护位编程设置第七总线写入周期地址)											
	闪速区域	固定到 “0“。	保护位选择 (表24-17)		固定到 “0“。				保护位选择 (表24-17)			Addr [1:0]= “0“(固定) 其他:0 (建议)
保护位擦除	PBA:保护位地址 (为保护位擦除而设置第七总线擦除周期地址)											
	闪速区域	固定到 “0“。	保护位选择 (表24-18)		固定到 “0“。				Addr [1:0]= “0“ (固定) 其他:0 (建议)			

作为块地址时，指定任何在块中要被擦除的地址。

表24-16 块地址表

块	地址 (用户启动模式)	地址 (单启动模式)	尺寸 (Kbyte)
8	0x0000_0000 ~ 0x0000_7FFF	0x3F80_0000 ~ 0x3F80_7FFF	32
9	0x0000_8000 ~ 0x0000_FFFF	0x3F80_8000 ~ 0x3F80_FFFF	32
7	0x0001_0000 ~ 0x0001_FFFF	0x3F81_0000 ~ 0x3F81_FFFF	64
6	0x0002_0000 ~ 0x0003_FFFF	0x3F82_0000 ~ 0x3F83_FFFF	128
5	0x0004_0000 ~ 0x0005_FFFF	0x3F84_0000 ~ 0x3F85_FFFF	128
4	0x0006_0000 ~ 0x0007_FFFF	0x3F86_0000 ~ 0x3F87_FFFF	128
3	0x0008_0000 ~ 0x0009_FFFF	0x3F88_0000 ~ 0x3F89_FFFF	128
2	0x000A_0000 ~ 0x000B_FFFF	0x3F8A_0000 ~ 0x3F8B_FFFF	128
1	0x000C_0000 ~ 0x000D_FFFF	0x3F8C_0000 ~ 0x3F8D_FFFF	128
0	0x000E_0000 ~ 0x000F_FFFF	0x3F8E_0000 ~ 0x3F8F_FFFF	128

注:**作为从第一 ~ 第五总线周期的地址，指定块中的以上地址来进行擦除。**

表24-17 保护位编程地址表

块	保护位	第七总线写入周期地址				
		地址 [18]	地址 [17]	地址 [16:11]	地址 [10]	地址 [9]
块0	<BLPRO[0]>	0	0	固定到 “0“。	0	0
块1	<BLPRO[1]>	0	0		0	1
块2	<BLPRO[2]>	0	0		1	0
块3	<BLPRO[3]>	0	0		1	1
块4	<BLPRO[4]>	0	1		0	0
块5	<BLPRO[5]>	0	1		0	1
块6	<BLPRO[6]>	0	1		1	0
块7	<BLPRO[7]>	0	1		1	1
块9	<BLPRO[9]>	1	0		0	1
块8	<BLPRO[8]>	1	0		0	0

表24-18 保护位擦除地址表

块	保护位	第七总线写入周期地址 [18:17]	
		地址 [18]	地址 [17]
块0 到 3	<BLPRO[0:3]>	0	0
块4 到 7	<BLPRO[4:7]>	0	1
块8 到 9	<BLPRO[8:9]>	1	0

注:**保护位擦除命令不能被单个块擦除。**

表24-19 ID-读取命令的第四总线写入循环ID地址 (IA)且数据将被以下32-位数据传输命令 (ID)读取

IA[15: 4]	ID[7:0]	代码
0y00	0x98	生产商代码
0y01	0x5A	设备代码
0y10	保留	–
0y11	0x10	宏代码

24.3.2.1 流程图

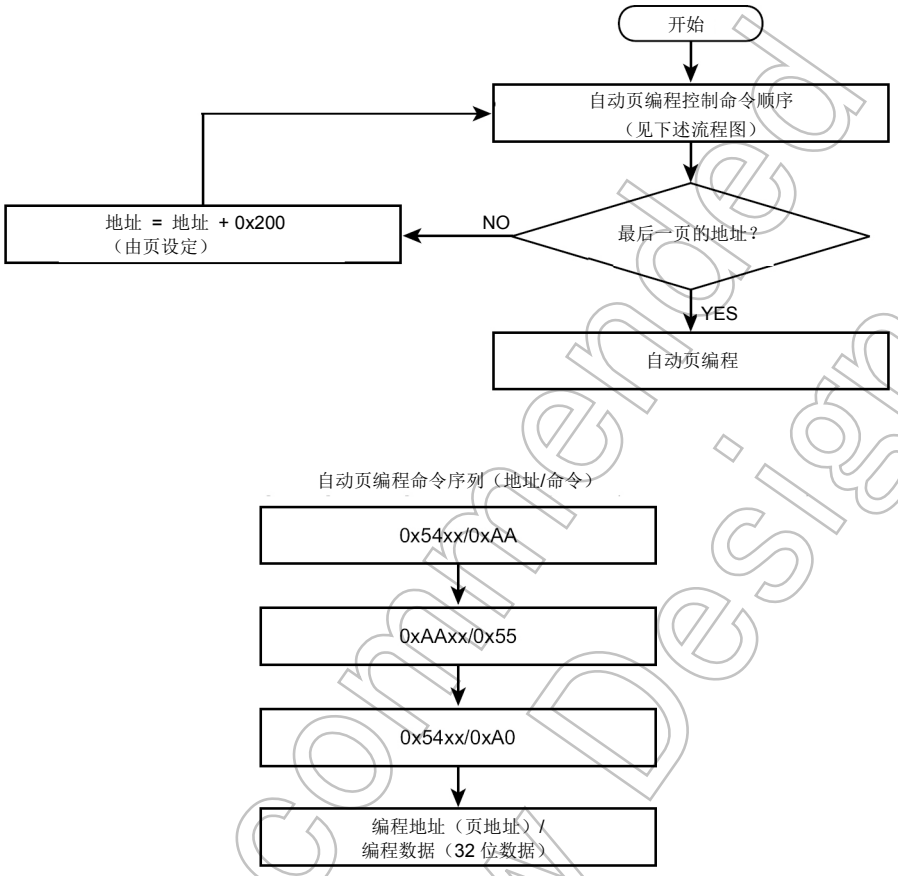


图24-10 自动编程

注:**命令序列由0x54xx 或 0x55xx来执行。**

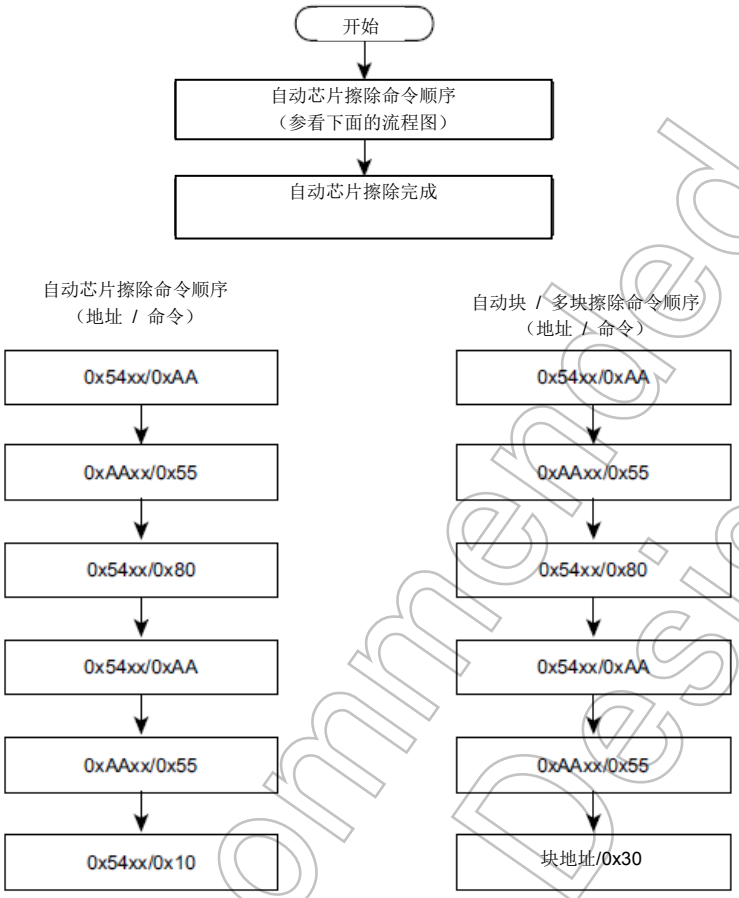


图24-11 自动擦除

注:命令序列由0x54xx 或 0x55xx来执行。

25. ROM保护

25.1 概述

TMPM364F10FG提供两种ROM保护/ 安全功能。

一种针对内部闪存ROM数据的写入/ 擦除-保护功能。

另一种是安全功能，其限制内部闪存ROM数据读出和调试。

25.2 特征

25.2.1 写入/ 擦除-保护功能

写入/擦除保护功能使内部闪存禁止对每一块的写入和擦除操作。

要启用功能，通过将 “1“写入对应位来对块进行保护。将 “0“写入此位删除保护。

这些位的保护设置可以通过FCFLCS<BLPRO[9:0]>位进行监控。编程详细见 “闪存“一章说明。

25.2.2 安全功能

安全功能限制闪存ROM数据读出和调试。

该功能在以下显示的条件下获得。

1. FCSECBIT <SECBIT>位被设置为 “1“。
2. 所有用于写入/擦除-保护功能的保护位 (FCFLCS<BLPRO> 位)都设置为 “1“。

注:**FCSECBIT <SECBIT>在上电复位后立刻在加电复位复位中设置为 “1“。**

表25-1 显示安全功能限制详情。

表25-1 安全功能限制

项目	详情
1) ROM 数据读出	数据可以从CPU读取。
2) 调试端口	SW通信及追踪均受限。
3) 闪存命令	禁止写入命令到闪存。尝试擦除用于写入/擦除保护的位的内容擦除所有保护位。

25.3 寄存器

基址 = 0x41FF_F000

寄存器名称		地址 (基+)
保留	-	0x0000, 0x0004
安全位寄存器	FCSECBIT	0x0010
闪存控制寄存器	FCFLCS	0x0020
保留	-	0x0024, 0x0028

注:**禁止访问"保留"区域。**

25.3.1 FCFLCS (闪存控制寄存器)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	BLPRO9	BLPRO8
复位后	0	0	0	0	0	0	(注2)	(注2)
	23	22	21	20	19	18	17	16
比特符号	BLPRO7	BLPRO6	BLPRO5	BLPRO4	BLPRO3	BLPRO2	BLPRO1	BLPRO0
复位后	(注2)	(注2)	(注2)	(注2)	(注2)	(注2)	(注2)	(注2)
	15	14	13	12	11	10	9	8
比特符号								
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	-	-	-	-	-	-	-	RDY/BSY
复位后	0	0	0	0	0	0	0	1

位	比特符号	型号	功能
31-26	−	R	读作 0。
25-16	BLPRO9- BLPRO0	R	对块9～0的保护 0: 禁用 1: 启用 保护状态位 每一个保护位代表相应块的保护状态。当一个位被设到“1“时，表示与该位对应的块被保护。当块被保护时，不能向其写入数据。
15-1	−	R	读作 0。
0	RDY/BSY	R	准备/忙碌 (注 1) 0: 自动工作 1:自动工作被终止 准备/忙碌标志位 RDY/BSY输出是作为监控自动工作状态的一种手段。此位为CPU监控该功能的一个功能位。当闪存在自动工作时，它输出“0“来表示它忙碌。当自动工作被终止时，返回准备状态并输出“1“来接受下一指令。当自动工作执行失败时，此位保持“0“输出。通过硬件复位复位“1“。

注 1: **该命令必须在准备状态下发出。在忙碌状态下发出此命令会禁用正确的指令传输和下一个的命令输入。要退出此条件，复位系统。系统复位要求至少0.5 ms，不管系统时钟频率如何。在这一条件下，在复位之后需要大约 2 ms 启用读取。**

注 2: **数值将因施加的保护不同而不同。**

25.3.2 FCSECBIT (安全位寄存器)

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

位	比特符号	型号	功能
31-1	−	R	读作 0。
0	SECBIT	R/W	安全位 0：禁用 1：启用

注:通过冷复位初始化寄存器。

25.4 写入和擦除

写入和擦除保护位在单芯片模式，单启动模式和写入模式下可获得。

25.4.1 保护位

对保护位的写入是以逐块为基础实现的。

当对所有块都设定到 “1“时，应在设定FCSECBIT <SECBIT>位 “0“后才能进行擦除。设置 “1“将擦除所有保护位。写入和擦除保护位时，应使用命令序列。

详情见“闪存“章节。

25.4.2 安全位

启用安全功能的FCSECBIT <SECBIT>位在上电复位后立即在上电复位中设到 “1“。位被以下步骤重新写入。

1. 写入代码0xa74a9d23 到 FCSECBIT 寄存器。
2. 自上面的.1. 写入16 时钟内的数据

注:以上步骤仅当使用32-位传输命令时才启用。

26. RAM接口

释放复位后，RAM等待时间 (从0x2000_4000 ~ 0x2000_BFFF)被设置为一个等待。可使用零等待。

26.1 寄存器列表

控制寄存器及地址显示如下。

基址= 0x41FF_F058

寄存器名称		地址 (基+)
RAM接口寄存器	RCWAIT	0x0000

26.1.1 RCWAIT (RAM 接口寄存器)

	31	30	29	28	27	26	25	24
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	-	-	-	-	-	-	-	RAM1WAIT
复位后	0	0	0	0	0	0	0	1

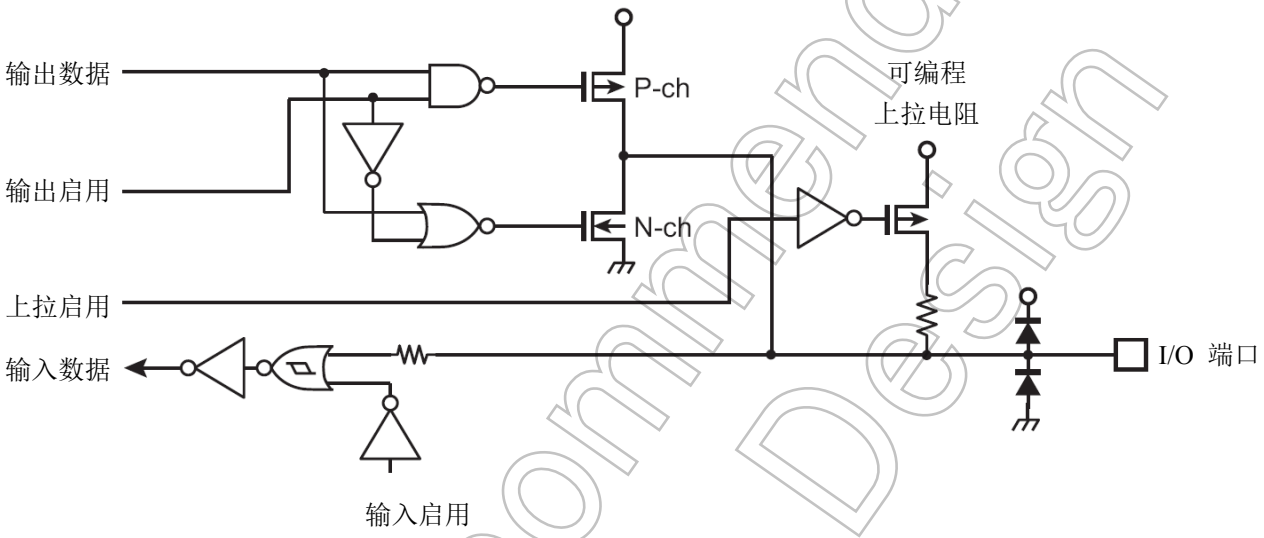
位	比特符号	型号	ã@ì\
31-1	−	R	读作 “0“
0	RAM1WAIT	R/W	指定RAMWAIT 0:0WAIT 1:1WAIT

27. 端口区等效电路示意图

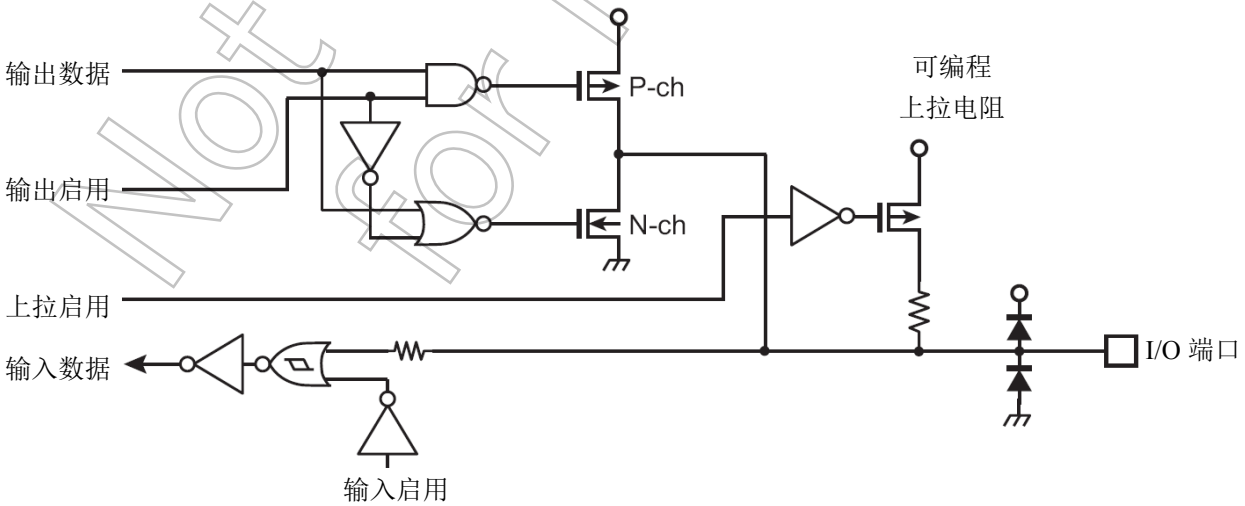
门符号写入与用于标准CMOS 逻辑 IC [74HCXX]系列门符号基本上相同。

输入保护电阻范围从几十Ω到几百Ω。阻尼电阻X2和XT2将与一典型值一同显示。

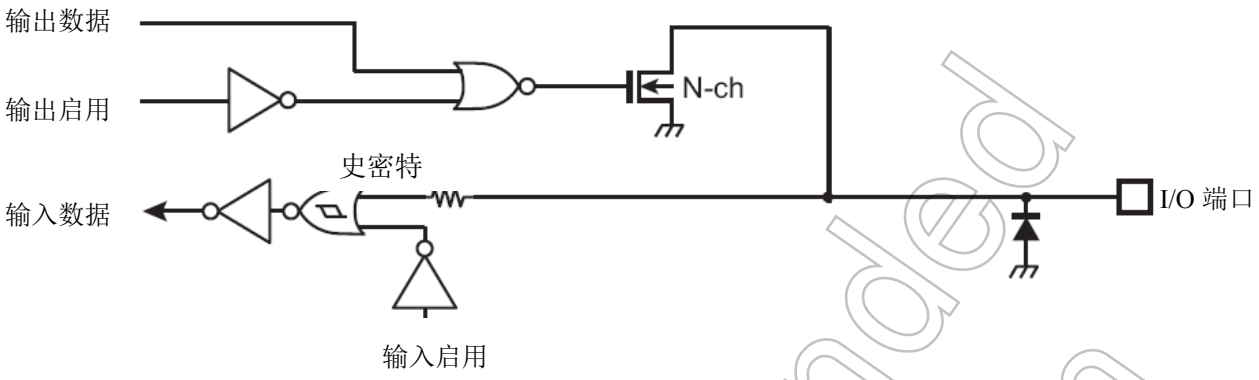
27.1 PA0 ~ 7, PB0 ~ 7, PP1, PP3 ~ 5



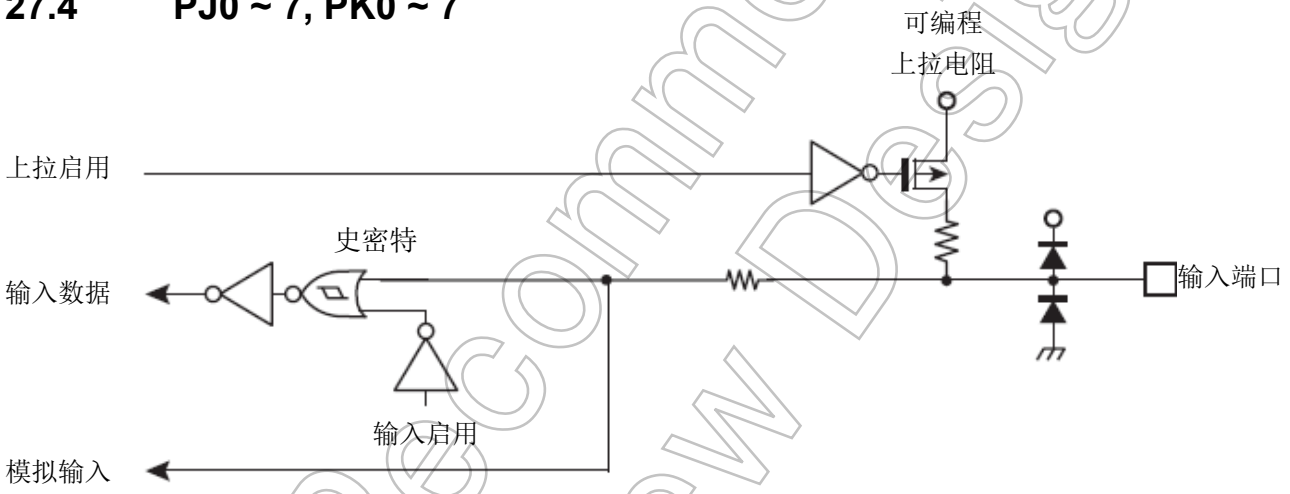
27.2 PC0 ~ 7, PD0 ~ 7, PE0 ~ 7, PF0 ~ 4, PG0 ~ 7, PH0 ~ 7, PI0, PL0 ~ 7, PM0 ~ 7, PN0 ~ 7, PO0 ~ 7, PP0, PP2, PP6



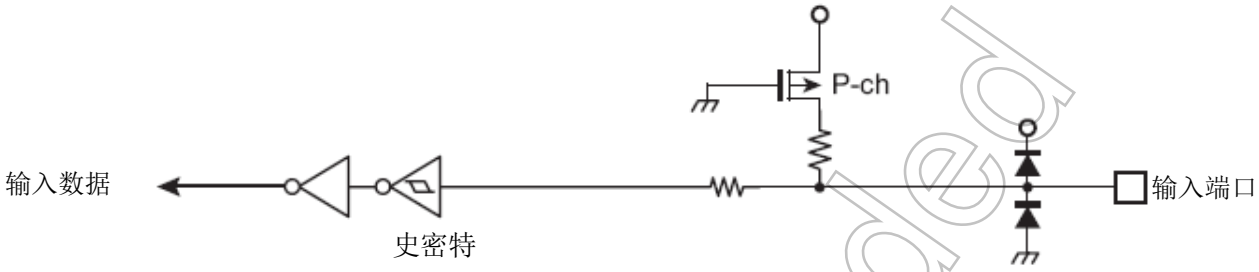
27.3 PI1



27.4 PJ0 ~ 7, PK0 ~ 7

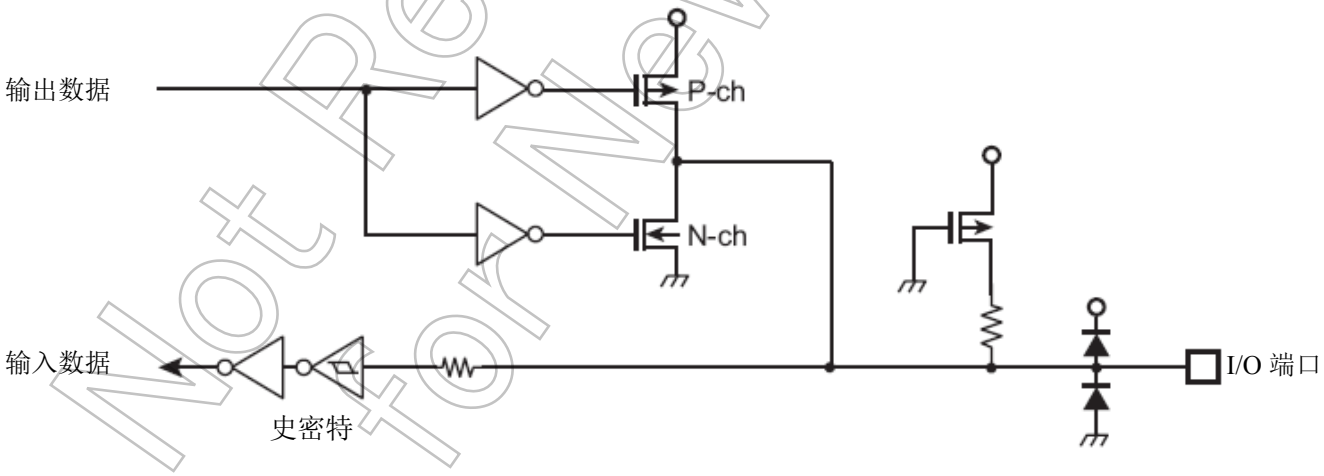


27.5 $\overline{\text{RESET}}$, $\overline{\text{NMI}}$

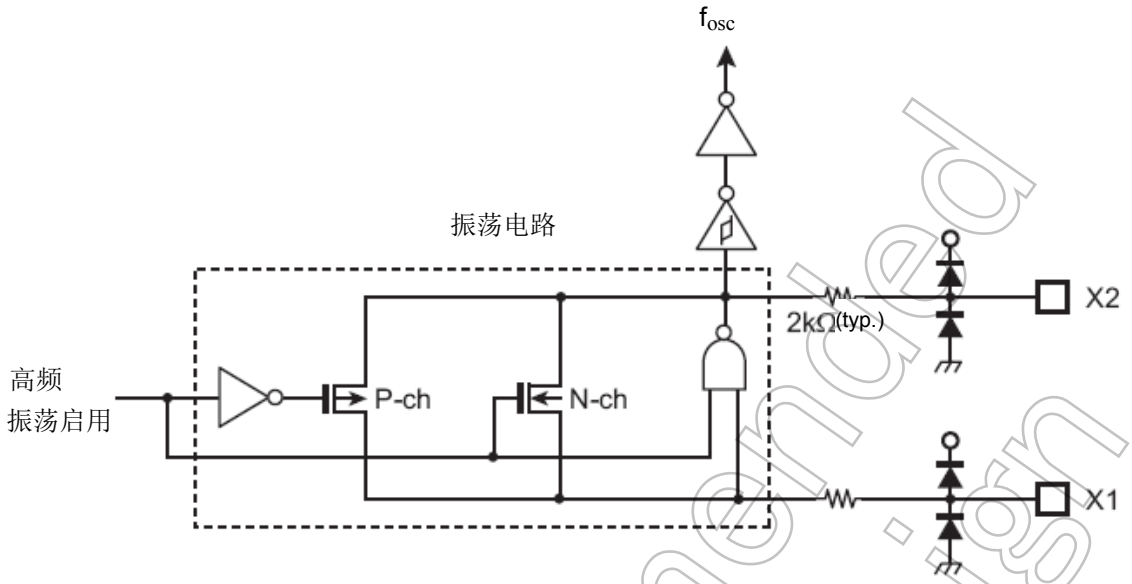


27.6 MODE, SWCLK

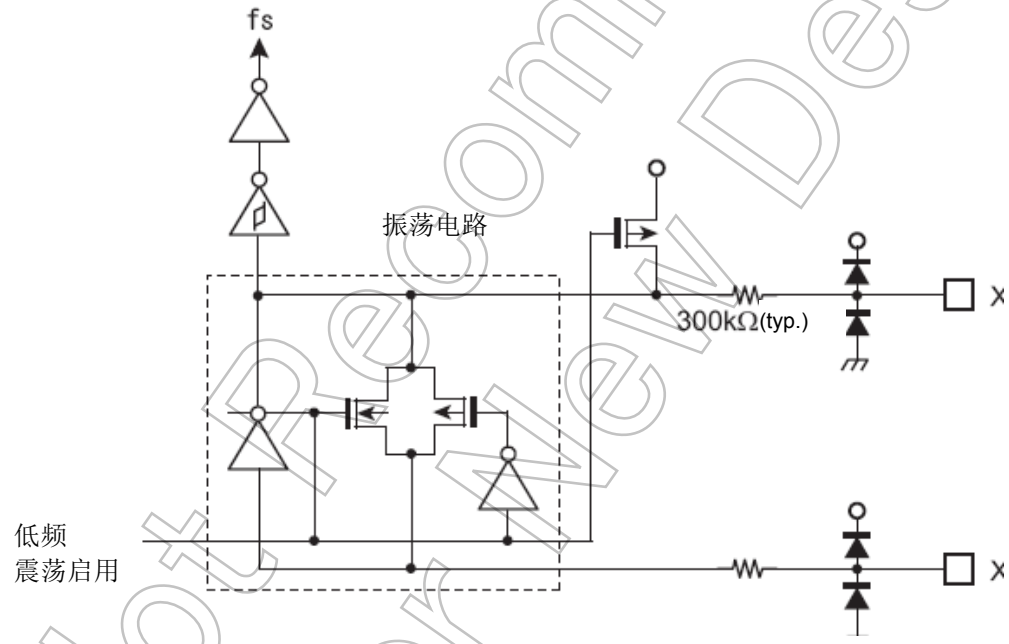
27.7 SWDIO



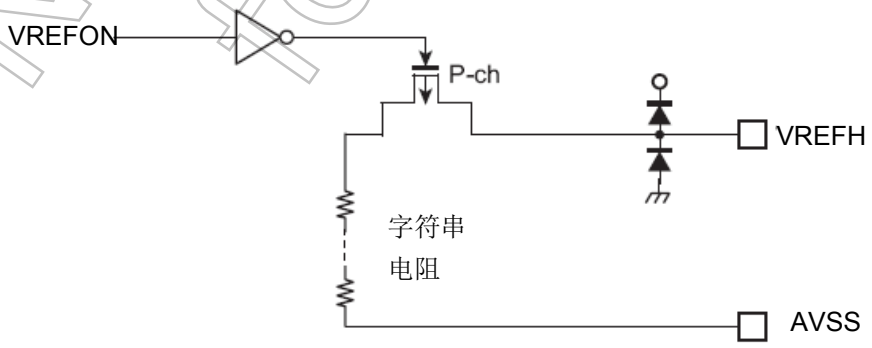
27.8 X1, X2



27.9 XT1, XT2



27.10 VREFH, AVSS



28. 电气特性

28.1 绝对最大额定值

参数		符号	额定值	单位
电源电压		DVDD3 (注2)	−0.3 ~ 3.9	V
		AVDD3	-0.3 ~ 3.9	
		RVDD3	-0.3 ~ 3.9	
输入电压		V_{IN}	−0.3 ~ DVDD3 + 0.3	V
低电平输出电流	每引脚	I_{OL}	5	mA
	全部	ΣI_{OL}	50	
高电平输出电流	每引脚	I_{OH}	−5	
	全部	ΣI_{OH}	−50	
功耗 (Ta = 85 °C)		PD	600	mW
焊接温度 (10 s)		T_{SOLDER}	260	°C
贮存温度		T_{STG}	-40 ~ 125	°C
工作温度	除开在闪存 W/E	T_{OPR}	-40 ~ 85	°C
	在闪存 W/E		0 ~ 70	

注 1: **绝对最大额定值是工作和环境条件的限制数值，即便在最差条件下都也不得超过。设备生产商的设计应使绝对额定值值在不同方面都不能被超过，包括电流，电压，能量消耗，温度等。否则会对设备造成永久伤害，或影响设备可信度，甚至 会因IC爆炸或燃烧而增加潜在的人员受伤。**

注 2: DVDD3 = DVDD3A = DVDD3B

28.2 DC 电气特性 (1/3)

Ta = −40 ~ 85 °C

参数		符号	条件		最小值	典型 (注1)	最大值	单位
电源电压	DVDD3A = AVDD3=DVDD3B= RVDD3 (注3) DVSS = AVSS = 0V	DVDD3A AVDD3 DVDD3B RVDD3	f_{OSC} = 8 ~ 16.0 MHz f_{sys} = 1 ~ 64 MHz f_S = 30 ~ 34 kHz		2.7	−	3.6	V
低-电平输入电压	PJ0 ~ 7, PK0 ~ 7 (注2)	V_{IL1}	2.7 V ≤ AVDD3 ≤ 3.6 V		−0.3	−	0.25 AVDD3	V
	PA0 ~ 7, PB0 ~ 7, PP1, PP3 ~ 5	V_{IL2}	2.7 V ≤ DVDD3 ≤ 3.6 V				0.3 DVDD3	
	PC0 ~ 7, PD0 ~ 7, PE0 ~ 7, PF0 ~ 4, PG0 ~ 7, PH0 ~ 7, PI0 ~ 1, PL0 ~ 7, PM0 ~ 7, PN0 ~ 7, PO0 ~ 7, PP0/2/6 $\overline{\text{RESET}}$, $\overline{\text{NMI}}$, MODE, SWDIO, SWCLK	V_{IL3}					0.25 DVDD3	
	X1	V_{IL4}	2.7 V ≤ DVDD3A ≤ 3.6 V				0.1 DVDD3A	
高-电平输入电压	PJ0 ~ 7, PK0 ~ 7 (注2)	V_{IH1}	2.7 V ≤ AVDD3 ≤ 3.6 V		0.75 AVDD3	−	AVDD3 + 0.3	V
	PA0 ~ 7, PB0 ~ 7, PP1, PP3 ~ 5	V_{IH2}	2.7 V ≤ DVDD3 ≤ 3.6 V		0.7 DVDD3		DVDD3 + 0.3	
	PC0 ~ 7, PD0 ~ 7, PE0 ~ 7, PF0 ~ 4, PG0 ~ 7, PH0 ~ 7, PI0 ~ 1, PL0 ~ 7, PM0 ~ 7, PN0 ~ 7, PO0 ~ 7, PP0/2/6 $\overline{\text{RESET}}$, $\overline{\text{NMI}}$, MODE, SWDIO, SWCLK	V_{IH3}			0.75 DVDD3			
	X1	V_{IH4}	2.7 V ≤ DVDD3A ≤ 3.6 V		0.9 DVDD3A		DVDD3A + 0.3	
低-电平输出电压	除以下外	V_{OL1}	IOL = 2 mA	DVDD3 ≥ 2.7 V	−	−	0.4	V
	PL0/1/4/5, PG0/1/4/5	V_{OL2}	IOL = 3 mA		−	−		
高-电平输出电压		V_{OH}	IOH = −2 mA	DVDD3 ≥ 2.7 V	2.4	−	−	V
输入泄漏电流		I_{LI}	0.0 ≤ V_{IN} ≤ DVDD3 0.0 ≤ V_{IN} ≤ AVDD3		−	0.02	±5	μA
输出泄漏电流		I_{LO}	0.2 ≤ V_{IN} ≤ DVDD3 − 0.2 0.2 ≤ V_{IN} ≤ AVDD3 − 0.2		−	0.05	±10	
复位时的上拉寄存器		RRST	DVDD3 = 2.7 V ~ 3.6 V		−	50	150	kΩ
滞后电压		VTH	2.7 V ≤ DVDD3 ≤ 3.6 V		0.3	0.6	−	V
可编程上拉 / 下拉寄存器		PKH	DVDD3 = 2.7 V ~ 3.6 V		−	50	150	kΩ
引脚电容 (除电源供应引脚外)		C_{IO}	fc = 1 MHz		−	−	10	pF

注 1: Ta = 25 °C，DVDD3 = RVDD3 = AVDD3 = 3.3 V， 除非另有说明。

注 2: **当PJ和PK端口用于输入端口。**

注 3: **DVDD3A, DVDD3B, AVDD3, RVDD3必须施加相同的电压。**

28.3 DC 电气特性 (2/3)

DVDD3A = DVDD3B = AVDD3 = RVDD3 = 2.7 V ~ 3.6 V, Ta =−40 ~ 85 °C

参数	符号	条件	最小值	典型 (注1)	最大值	单位
低-电平输出电流	I_{OL1}	除以下 (每引脚)	−	−	2	mA
	I_{OL2}	PL0/1/4/5, PG0/1/4/5 (每引脚)	−	−	3	mA
	ΣI_{OL1}	每组群 (端口 L/I)	−	−	20	mA
	ΣI_{OL2}	每组群 (端口 M/N/O/P)	−	−	27	mA
	ΣI_{OL3}	每组群 (端口 A/B/C/D/E)	−	−	27	mA
	ΣI_{OL4}	每组群 (端口 F/G/H)	−	−	27	mA
	ΣI_{OL5}	合计 (所有端口)	−	−	35	mA
高-电平输出电流	I_{OH}	每引脚	−	−	−2	mA
	ΣI_{OH1}	每组群 (端口 I/L/M/N/O/P)	−	−	−13	mA
	ΣI_{OH2}	每组群 (端口 A/B/C/D/E)	−	−	−13	mA
	ΣI_{OH3}	每组群 (端口 F/G/H)	−	−	−13	mA
	ΣI_{OH4}	合计 (所有端口)	−	−	−35	mA

注1: Ta = 25 °C, DVDD3 = RVDD3 = AVDD3 = 3.3 V，除非另有说明。

注2: **DVDD3, DVDD3A, DVDD3B, AVDD3, RVDD3必须施加相同的电压。**

注3: **高电平输出电流 (ΣI_{OH})为每个电能供应引脚电流的总数。**

28.4 DC 电气特性 (3/3)

DVDD3A = DVDD3B = AVDD3 = RVDD3 = 2.7 V ~ 3.6 V, Ta =−40 ~ 85 °C

参数	符号	条件	最小值	典型 (注1)	最大值	单位
正常 (注2) 齿轮1/1	I_{DD}	f_{sys} = 64 MHz (f_{OSC} = 8 MHz)	−	85	95	mA
IDLE2 (注3)			−	40	48	
IDLE1 (注4)		f_{sys} = 1 MHz (f_{OSC} = 8 MHz, PLL= OFF, CG = 1/8)	−	1.3	5	
SLOW		f_S = 32.768 kHz	−	1	6	
SLEEP (注5)			−	260	2450	μA
STOP		−	−	250	2400	
BACKUP SLEEP (注6)		f_S=32.768 kHz	−	35	210	
BACKUP STOP (注7)		−	−	25	200	

注 1: Ta = 25 °C, DVDD3 = RVDD3 = AVDD3 = 3.3 V，除非另有说明。

注 2: IDD NORMAL:测量使用:

操作程序:在闪存中操作的dhrystone ver.2.1。

外围操作:所是功能的操作排除ADC。

注 3: **IDD IDLE2：测量使用:**

外围操作：全部外围设备均操作。

注 4: **IDD IDLE1：测量使用:**

外围操作 ：部分外围设备操作。

注 5: **IDD SLEEP：测量使用:**

外围操作: CEC，RMC 和 RTC 操作。

注 6: **IDD BACKUP SLEEP：测量使用:**

外围操作 ：CEC, RMC 和 RTC 操作，保存BACKUP RAM 的内容，其他外围设备下电。

注 7: **IDD BACKUP SLEEP:测量使用:**

外围操作 ：保存BACKUP RAM中内容，其它外围设备下电。

28.5 10-位 ADC 电气特性

DVDD3A = DVDD3B = AVDD3 = RVDD3 = 2.7 V ~ 3.6 V

AVSS = DVSS, Ta = −40 ~ 85 °C

参数		符号	条件	最小值	典型	最大值	单位
模拟参考电压 (+)		VREFH	−	2.7	3.3	3.6	V
模拟输入电压		VAIN	−	AVSS	−	VREFH	V
模拟参考电压的电源电流。	AD 转换	IREF	DVSS = AVSS	−	2.5	5.5	mA
	非-AD 转换			−	0.02	5	μA
电源电流	AD 转换	−	除IREF	−	−	3	mA
INL 错误		−	AIN 电阻 ≤ 1.3 kΩ AIN 负载电容 ≥ 0.1Fμ 转变时间 ≥ 1.5 μs	-	±2	±3	LSB
DNL 错误				-	±1	±2	
偏差错误				-	±2	±4	
满刻度错误				-	±2	±4	

注 1: 1LSB = (VREFH – AVSS) / 1024 [V]
注 2: **外设功能禁用。**

28.6 AC 电气特性

28.6.1 AC 测量条件

此章节AC特性数据的测量在如下条件下进行，除非另有说明。

- 输出电平: 高 = 0.8 × DVDD3
- 输出电平: 高 = 0.2 × DVDD3
- 输入电平: 见 “DC电气特性“一章中低-电平输入电压和高-电平输入电压说明
- 负载能力: CL = 30pF

注: “**等式“条件为DVDD3 =2.7V ~ 3.6 V。**

28.6.2 静态存储控制器 (SMC)

"T"为表中等式的总线频率 (fsys)的1/2周期。

AC测量条件

- 输出电平 : 高 = 0.7 × DVDD3, 低 = 0.3 × DVDD3
- 输入电平 : 高= 0.7 × DVDD3, 低 = 0.3 × DVDD3
- 负载能力 : CL = 40pF

28.6.2.1 基本总线周期 (读取)

参数	符号	等式		f_{sys} = 64 MHz T=31.25 N = 4 M = 1 K = 5 L = 2 P = 2 Q = 2	单位
		最小值	最大值		
SMCCLK	t_{CYC}	31.25	2000	31.3	ns
A1 ~ A23 有效 → D0 ~ D15 输入 (单独总线模式)	t_{ADH}	–	(N)T − 35.0	90.0	
A1 ~ A23 有效 → D0 ~ D15输入 (多元总线模式)	t_{ADL}	–	(N)T − 35.0	90.0	
A1 ~ A23 有效 → D0 ~ D15 输入 (页访问)	t_{AD1}	–	(Q)T − 35.0	27.5	
$\overline{OE}$ 下降沿 → D0 ~ D15 输入	t_{OED}	–	(N − M)T − 25.0	68.8	
$\overline{OE}$ 低-电平脉冲宽度	t_{OEW}	(N − M)T − 13.0	–	80.8	
A1 ~ A23 有效 → $\overline{OE}$ 下降沿 (单独总线模式)	t_{AOEH}	(M)T − 15.0	–	16.3	
A1 ~ A16 有效 → $\overline{OE}$ 下降沿 (多元总线模式)	t_{AOEL}	(M)T − 15.0	–	16.3	
$\overline{OE}$ 上升沿 → D0 ~ D15 保留	t_{HR}	0.00	–	0.00	
A1 ~ A23 有效 → D0 ~ D15 保留	t_{HA}	0.00	–	0.00	
$\overline{OE}$ 高-电平脉冲宽度	t_{OEHW}	(M)T − 13.0	–	18.3	
$\overline{ALE}$ 低-电平脉冲宽度	t_{LL}	T − 13.0	–	18.3	
A1 ~ A16 有效 → $\overline{ALE}$ 上升沿	t_{AL}	T − 15.0	–	16.3	
$\overline{ALE}$ 上升沿 → A1 ~ A16 保留	t_{LA}	T − 10.0	–	21.3	
$\overline{OE}$ 上升沿 → $\overline{ALE}$ 下降沿	t_{CLR}	(P)T − 13.0	–	49.5	
$\overline{OE}$ 上升沿 → A1 ~ A16 保留	t_{CAR}	(P)T − 13.0	–	49.5	
$\overline{OE}$ 上升沿 → A1 ~ A16 输出	t_{RAE}	(P)T − 13.0	–	49.5	

注: "**等式**"**测量条件:**

N = t_{RC} 周期 ≥ 3 , M = t_{CEOE} 周期 ≥ 1
K = t_{WC} 周期 ≥ 3 , L = t_{WP} 周期 ≥ 1
P = t_{TR} 周期 ≥ 1 , Q = t_{PC} 周期 ≥ 1

28.6.2.2 基本总线周期 (写入)

参数	符号	等式		fsys = 64 MHz N = 4 M = 1 K = 5 L = 2 P = 2 Q = 2	单位
		最小值	最大值		
D0 ~ D15 有效 → $\overline{WE}$ 上升沿 (单独总线模式)	t_{DW}	(L + 1)T − 23.0	−	70.8	ns
D0 ~ D15 有效 → $\overline{WE}$ 上升沿 (多元总线模式)	t_{DW1}	(L)T − 23.0	−	39.5	
$\overline{WE}$ 低-电平脉冲宽度 (单独总线模式)	t_{WW}	(L)T − 13.0	−	49.5	
$\overline{WE}$ 低-电平脉冲宽度 (多元总线模式)	t_{WW1}	(L + 1)T − 13.0	−	80.8	
A1 ~ A23 有效 → $\overline{WE}$ 下降沿	t_{AW}	T− 15.0	−	16.3	
$\overline{WE}$ 上升沿 → A1 ~ A23 保留	t_{WA}	(K − L)T − 13.0	−	80.8	
$\overline{WE}$ 上升沿→D0 ~ D15 保留 (单独总线模式)	t_{WD}	(K − L − 1)T − 10.0	−	52.5	
$\overline{WE}$ 上升沿→D0 ~ D15 保留 (多元总线模式)	t_{WD1}	(K − L − 2)T − 10.0	−	21.3	
$\overline{WE}$ 上升沿 → A1 ~ A16 保留	t_{CLW}	(K − L − 2 + P)T − 13.0	−	80.8	
$\overline{WE}$ 上升沿 → $\overline{ALE}$ 下降沿	t_{CAW}	(K − L − 2 + P)T − 13.0	−	80.8	

注: "**等式**"**测量条件:**

N = t_{RC} 周期 ≥ 3 , M = t_{CEOE} 周期 ≥ 1

K = t_{WC} 周期 ≥ 3 , L = t_{WP} 周期 ≥ 1

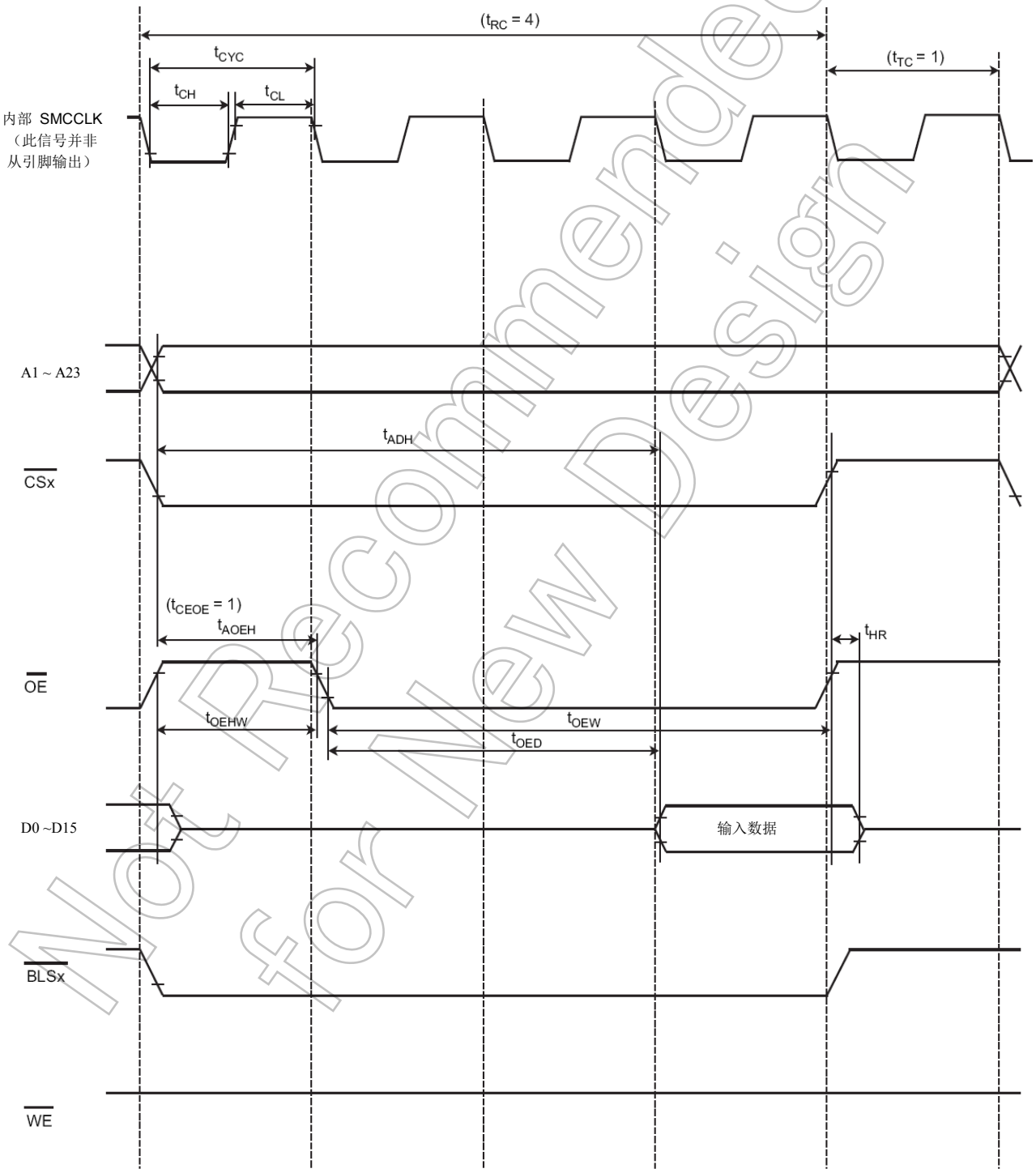
P = t_{TR} 周期 ≥ 1 , Q = t_{PC} 周期 ≥ 1

28.6.2.3 读取/写入周期示例

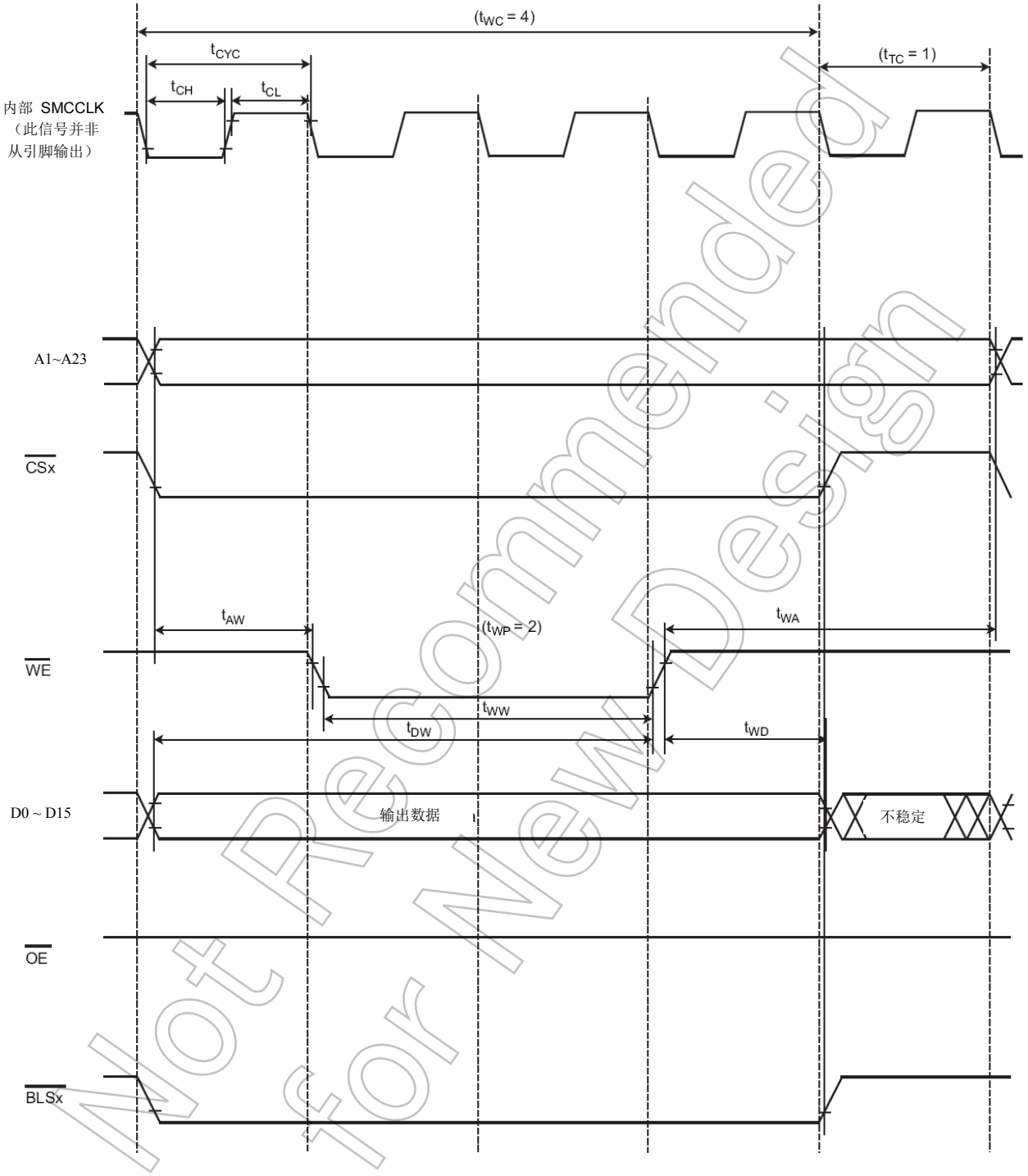
当从外部总线访问时，除有效读取周期外还会产生一个虚拟读取周期。由此，外部存储器不能用做FIFO.

以下的操作图仅显示有效读取周期，虚拟周期除外。

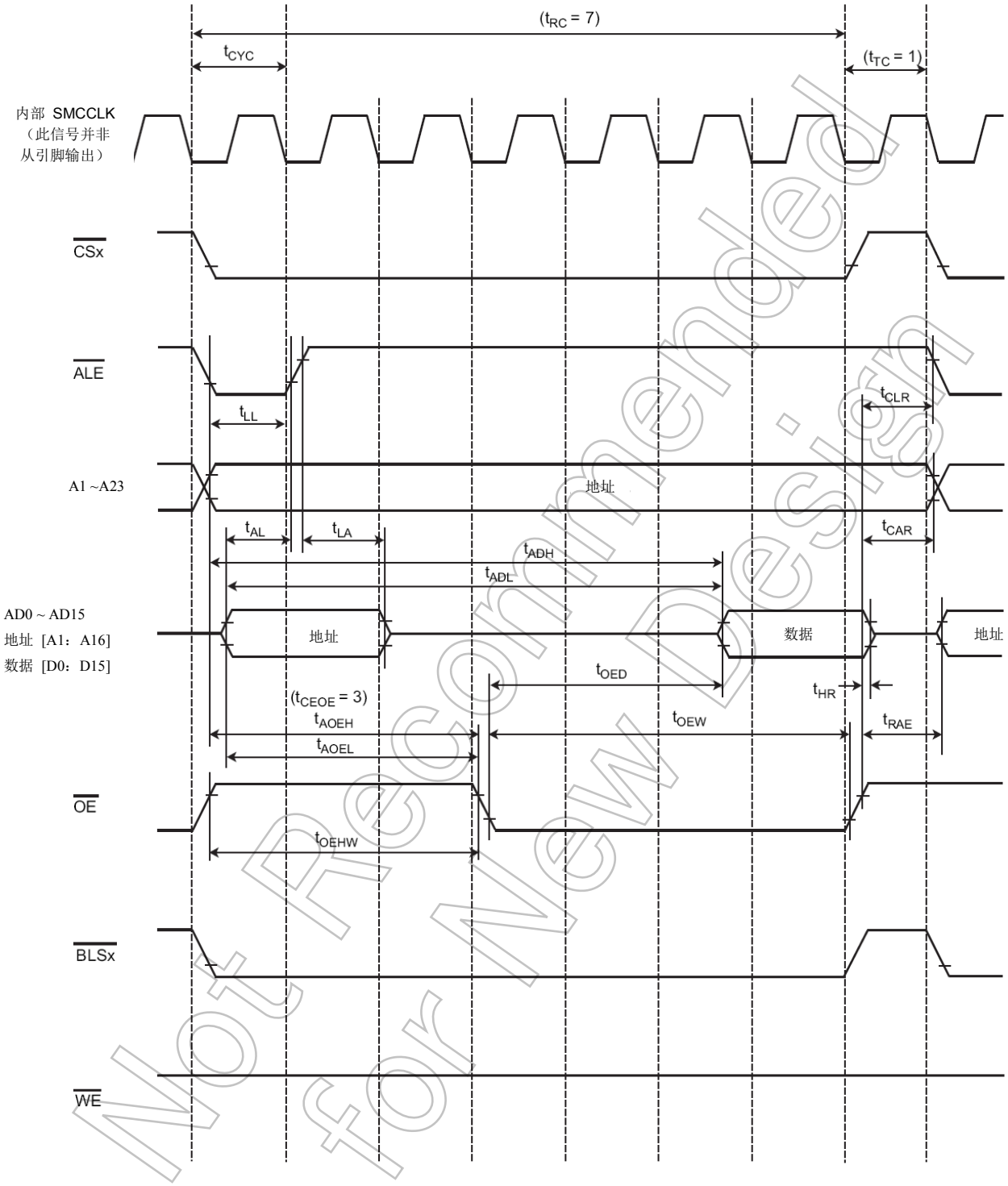
(1) 存储器读取周期 (单独总线) (t_{RC}=4, t_{CEOE}=1)



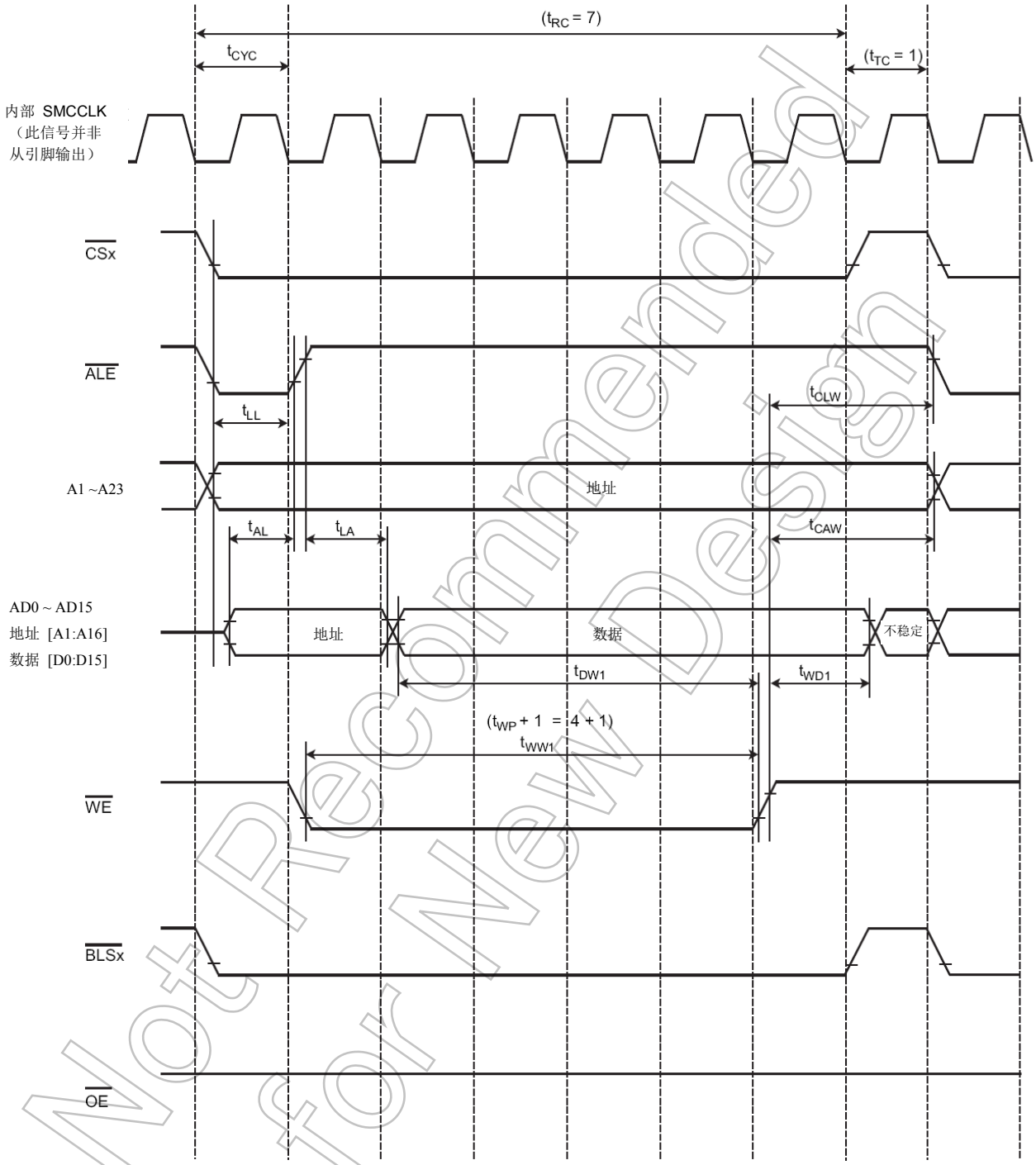
(2) 存储器写入周期 (单独总线) (t_{WC}=4, t_{WP}=2)



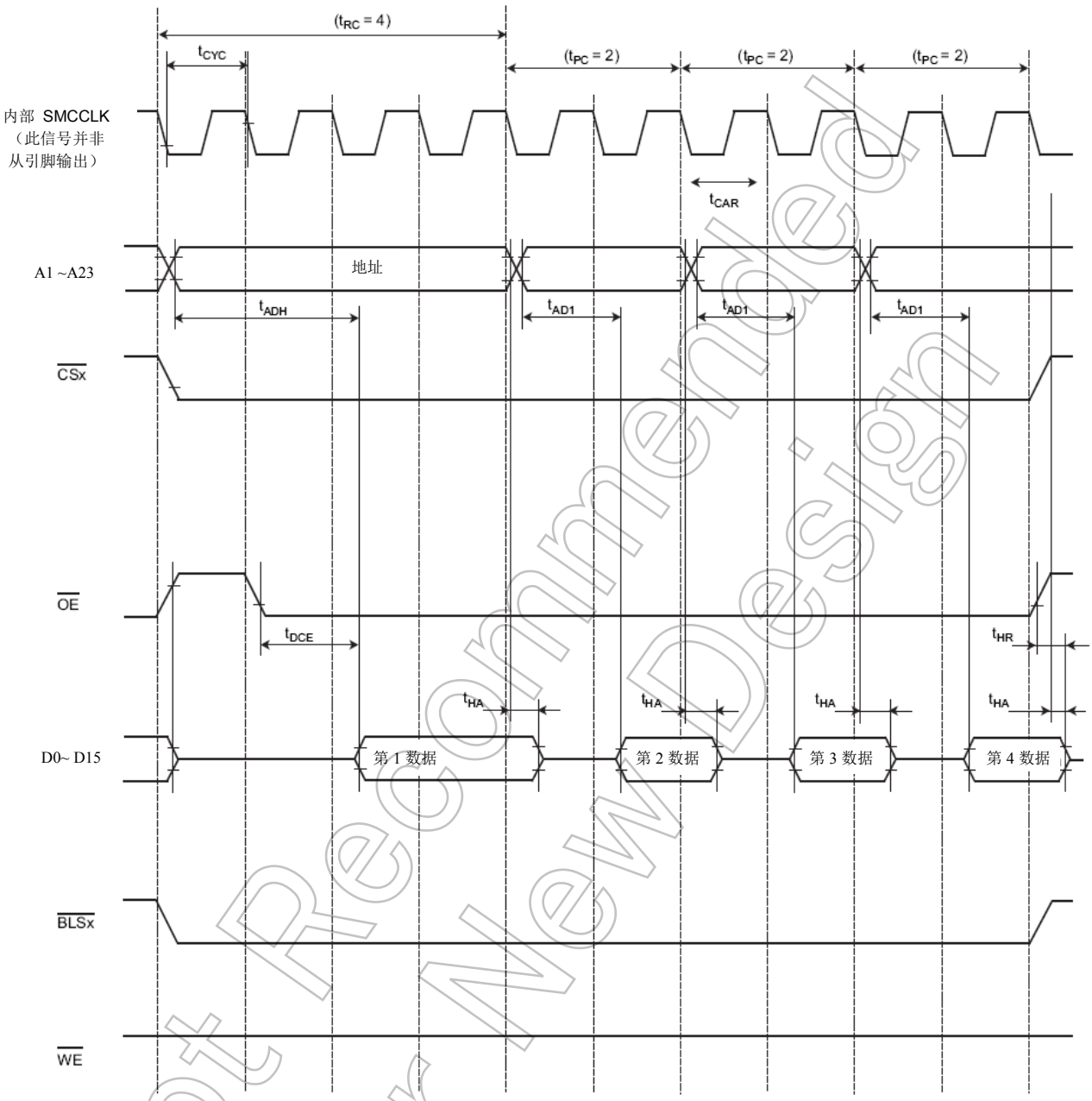
(3) 存储器读取周期 (多元总线) (t_{RC}=7, t_{CEOE}=3)



(4) 存储器写入周期 (多元总线) (t_{WC}=7, t_{WP}=4)



(5) 存储器读取周期 (单独总线页访问) (t_{RC}=4, t_{CEOE}=1, t_{PC}=2 4突入)



28.6.3 串行接口 (SIO/UART)

28.6.3.1 I/O接口模式

在下表中，字母×代表:SIO工作时钟周期时间与fsys周期时间相等。它根据时钟齿轮功能的编程而是不同。

(1) SCLK 输入模式

[数据输入]

参数	符号	等式		f_{sys} = 64 MHz		单位
		最小值	最大值	最小值	最大值	
SCLK 时钟 高宽度 (输入)	t_{SCH}	4X	−	62.5	−	ns
SCLK 时钟 低宽度 (输入)	t_{SCL}	4X	−	62.5	−	
SCLK 周期	t_{SCY}	$t_{SCH} + t_{SCL}$	−	125	−	
有效数据输入→ SCLK 上升 / 下降 (注1)	t_{SRD}	30	−	30.0	−	
SCLK 上升 / 下降 (注1) → 输入数据保留	t_{HSR}	x + 30	−	45.6	−	

[数据输出]

参数	符号	等式		f_{sys} = 64 MHz		单位
		最小值	最大值	最小值	最大值	
SCLK 时钟 高宽度 (输入)	t_{SCH}	4X	−	91.9 (Note3)	−	ns
SCLK 时钟 低宽度 (输入)	t_{SCL}	4X	−	91.9 (Note3)	−	
SCLK 周期	t_{SCY}	$t_{SCH} + t_{SCL}$	−	184	−	
输出数据→ SCLK 上升 / 下降 (注1)	t_{OSS}	$t_{SCY}/2 - 3\times - 45$	−	0.00 (注2)	−	
SCLK 上升 / 下降 (注1)→ 输出数据保留	t_{OHS}	$t_{SCY}/2$	−	91.9	−	

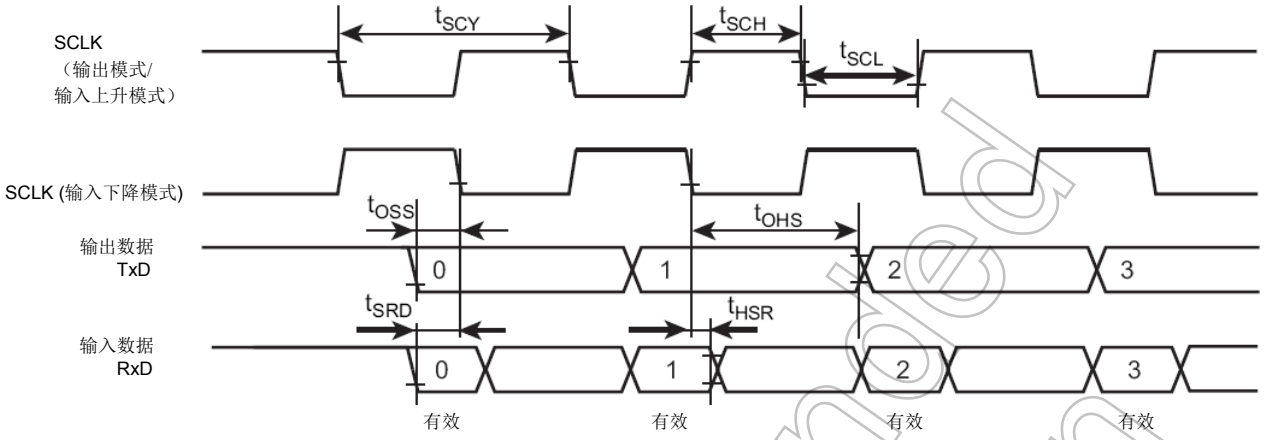
注 1: SCLK上升/下降 : SCLK上升模式使用 SCLK上升时间。SCLK下降模式使用SCLK下降时间。

注 2: **对SCLOK频率使用的范围的条件是计算值保持为正。**

注 3: **该值将表示启用t_{OSS}为零或以上的最小值。**

(2) SCLK 输出模式

参数	符号	等式		fsys = 64 MHz		单位
		最小值	最大值	最小值	最大值	
SCLK周期 (可编程)	t_{SCY}	4X	−	62.5		ns
输出数据→SCLK上升	t_{OSS}	$t_{SCY}/2 - 20$	−	11.3		
SCLK 上升 → 输出数据保留	t_{OHS}	$t_{SCY}/2 - 20$	−	11.3		
有效数据输入→ SCLK上升	t_{SRD}	x + 45	−	45		
SCLK 上升 → 输入数据保留	t_{HSR}	0	−	0		



28.6.4 串行总线接口 (I2C/SIO)

28.6.4.1 I2C模式

在下表中，字母×表示I_2C操作时钟周期时间与fsys周期时间相等。它根据时钟齿轮功能的编程而是不同。

n表示被编程到SBIxCR 中的SCK (SCL输出频率选择)栏位的n值。

参数	符号	等式		标准模式		快模式		单位
		最小值	最大值	最小值	最大值	最小值	最大值	
SCL 时钟频率	t_{SCL}	0	–	0	100	0	400	kHz
START条件保持时间	$t_{HD;\ STA}$	–	–	4.0	–	0.6	–	μs
SCL低宽度 (输入) (注1)	t_{LOW}	–	–	4.7	–	1.3	–	μs
SCL高宽度 (输入) (注2)	t_{HIGH}	–	–	4.0	–	0.6	–	μs
START条件的设置时间	$t_{SU;\ STA}$	(注5)	–	4.7	–	0.6	–	μs
数据保留时间 (输入) (注3,4)	$t_{HD;\ DAT}$	–	–	0.0	–	0.0	–	μs
数据设置时间	$t_{SU;\ DAT}$	–	–	250	–	100	–	ns
STOP条件设置时间	$t_{SU;\ STO}$	–	–	4.0	–	0.6	–	μs
STOP条件和START条件之间的总线空闲时间。	t_{BUF}	(注5)	–	4.7	–	1.3	–	μs

注 1: SCL时钟低宽度 (输出) : $(2^{n-1} + 58)/x$

注 2: SCL时钟高宽度 (输出) : $(2^{n-1} + 14)/x$

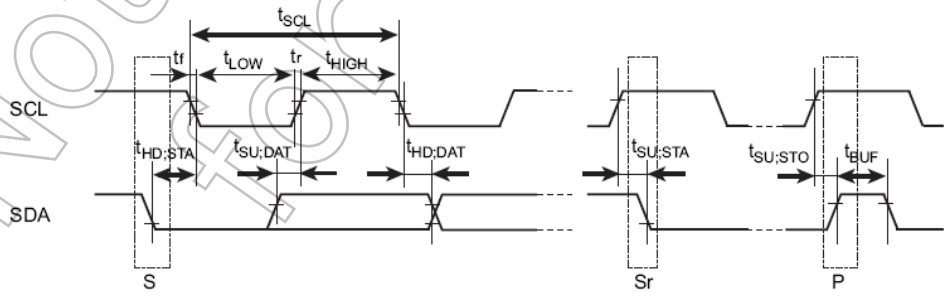
在I2C-总线规格上，标准模式的最大速度为100kHz, 闪存模式的为400khz。内部SCL频率设置要符合于上面的注1 & 注2。

注 3: **输出数据保留时间相等于内部SCL的4x。**

注 4: **飞利浦I2C-总线规格为，设备应在内部提供相应保留时间最少为300 ns，来让SDA信号桥接SCL下降沿的未定义区域。然而，此SBI并不满足此要求。另外，SCL的输出缓冲并不包含对下降沿的控制；设备生产商要进行设计以便使下表中的输入数据保留时间得到满足，包括SCL和SDA线的tr/tf。**

注5:**软件依据**

注6:飞利浦I2C-总线规格表明，当快模式设备的电源切断时，SDA 和SCL I/O引脚应该漂浮，以便不妨碍主线。然而，此SBI并不满足此要求。



S: START条件

Sr: RESTART条件

P: STOP条件

28.6.4.2 时钟-同步8-位SIO模式

在下表中，字母x表示SBI操作时钟周期时间与f_{sys}周期时间相等。它根据时钟齿轮功能的编程而是不同。

(1) SCK输入模式 (以下的电器规格用于带50%工作周期的SCK信号。)

[数据输入]

参数	符号	等式		f_{sys} = 64 MHz		单位
		最小值	最大值	最小值	最大值	
SCL 时钟高宽度 (输入):	t_{SCH}	4x	–	62.5	–	ns
SCL 时钟低宽度 (输入):	t_{SCL}	4x	–	62.5	–	
SCK 周期	t_{SCY}	$t_{SCH} + t_{SCL}$	–	125	–	
SCK 上升→ 有效数据输入	t_{SRD}	30-x	–	14.4	–	
SCK 上升→ 有效数据保留	t_{HSR}	2x + 30	–	61.3	–	

[数据输出]

参数	符号	等式		fsys = 64 MHz		单位
		最小值	最大值	最小值	最大值	
SCL 时钟高宽度 (输入):	t_{SCH}	4x	–	91.9 (注2)	–	ns
SCL 时钟低宽度 (输入):	t_{SCL}	4x	–	91.9 (注1)	–	
SCK 周期	t_{SCY}	t_{SCH} + tSCL	–	184	–	
SCK 上升 → 输出数据	t_{OSS}	t_{SCY}/2 - 3x - 45	–	0 (注1)	–	
SCK 上升 → 输出数据保留	t_{OHS}	t_{SCY}/2 + x	–	107.6	–	

注 1: **对SCLOK频率使用的范围的条件是计算值保持为正。**

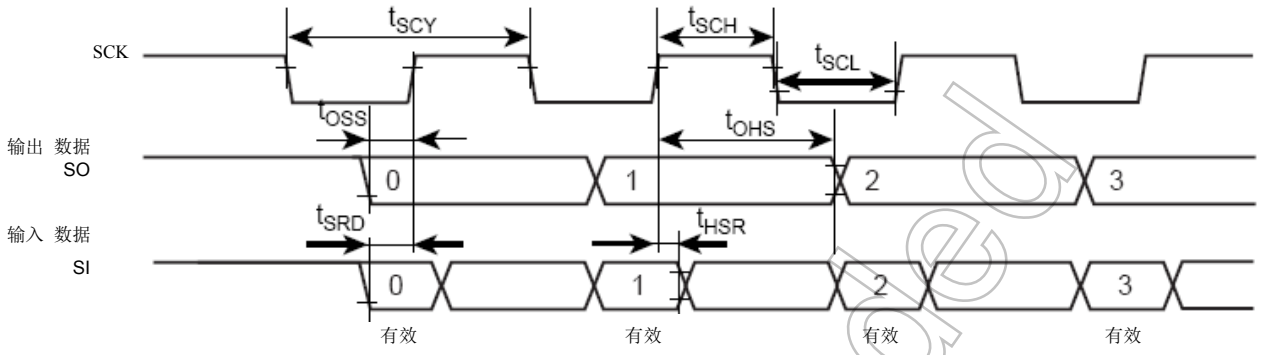
注 2: **该值将表示启用t_{OSS}为零或以上的最小值。**

(2) SCK输出模式 (以下的电气规格用于带50%工作周期的SCK信号。)

参数	符号	等式		f_{sys} = 64 MHz		单位
		最小值	最大值	最小值	最大值	
SCK 周期 (可编程)	t_{SCY}	16x	–	250	–	ns
输出 数据 → SCK 上升	t_{OSS}	t_{SCY}/2 - 20	–	105	–	
SCK 上升 → 输出数据保留	t_{OHS}	t_{SCY}/2 - 20	–	105	–	
有效数据输入→ SCK 上升	t_{SRD}	x + 45	–	60.6	–	
SCK上升→ 有效数据保留	t_{HSR}	0	–	0	–	

注1:自动等待后的SCK周期将变为14x。

注2:自动等待后的tOSS 会是 t_{SCY}/2-x-20。



28.6.5 SSP 控制器 (SSP)

"T"为表中等式的总线频率 (f_{sys})的1/2周期。

AC测量条件

输出高电平 = 0.8 × DVDD3, 低 = 0.2 × DVDD3

输入参考电压: DC电气特性中低-电平输入电压和高-电平输出电压。

负载电容 = 30pF

注:**下表中的 "等式"栏显示在DVDD3 2.7 ~ 3.6 V条件下的规格。**

参数	符号	等式		f_{sys} = 64 MHz m = 4 n = 12	单位
		最小值	最大值		
SPCLK 周期 (主)	T_m	(m)T 始终要多过50 ns	-	62.5 (16MHz)	ns
SPCLK 周期 (从)	T_s	(n)T	-	187.5 (8MHz)	
SPCLK 上升时间	t_r	-	10.0	10.0	
SPCLK 下降时间	t_f	-	10.0	10.0	
主模式 : SPCLK 低-电平脉冲宽度	t_{WLM}	(m)T / 2 - 10.0	-	21.3	
主模式 : SPCLK 高-电平脉冲宽度	t_{WHM}	(m)T / 2 - 10.0	-	21.3	
从模式 : SPCLK 低-电平脉冲宽度	t_{WLS}	(n)T / 2 - 10.0	-	83.8	
从模式 : SPCLK 高-电平脉冲宽度	t_{WHS}	(n)T / 2 - 10.0	-	83.8	
主模式 : SPCLK 上升 / 下降 → 输出有效	t_{ODSM}	-	15.0	15.0	
主模式 : SPCLK 上升 / 下降 → 输出数据保留	t_{ODHM}	(m)T / 2 - 10.0	-	21.3	
主模式 : 输入数据有效 → SPCLK 上升 / 下降	t_{IDSM}	15.0	-	15.0	
主模式 : SPCLK 上升 / 下降 → 输入数据保留	t_{IDHM}	5.00	-	5.00	
主模式 : SPFSS有效 → SPCLK 上升 / 下降	t_{OFSM}	(m)T - 10.0	(m)T + 10.0	52.5 - 72.5	
从模式: SPCLK 上升 / 下降 → 输出数据有效	t_{ODSS}	-	(3T) + 22.0	68.9	
从模式 : SPCLK 上升 / 下降 → 输出数据保留	t_{ODHS}	(n)T / 2 + (2T)	-	125	
从模式 : 输入数据有效 → SPCLK 上升 / 下降	t_{IDSS}	0.00	-	0.00	
从模式 : SPCLK 上升 / 下降 → 输入数据保留	t_{IDHS}	(3T) + 10.0	-	56.9	
从模式 : SPFSS有效 → SPCLK 上升 / 下降	t_{OFSS}	(n)T - 15.0	-	172.5	

注:**波特率时钟在如下条件下设定**

主模式：

m= (<CPSDVSR>× (1+<SCR>))= f_{sys} /f_{SPCLK} <CPSDVR> 仅被设为偶数且 “m“

设置应符合 65024≥ m≥4

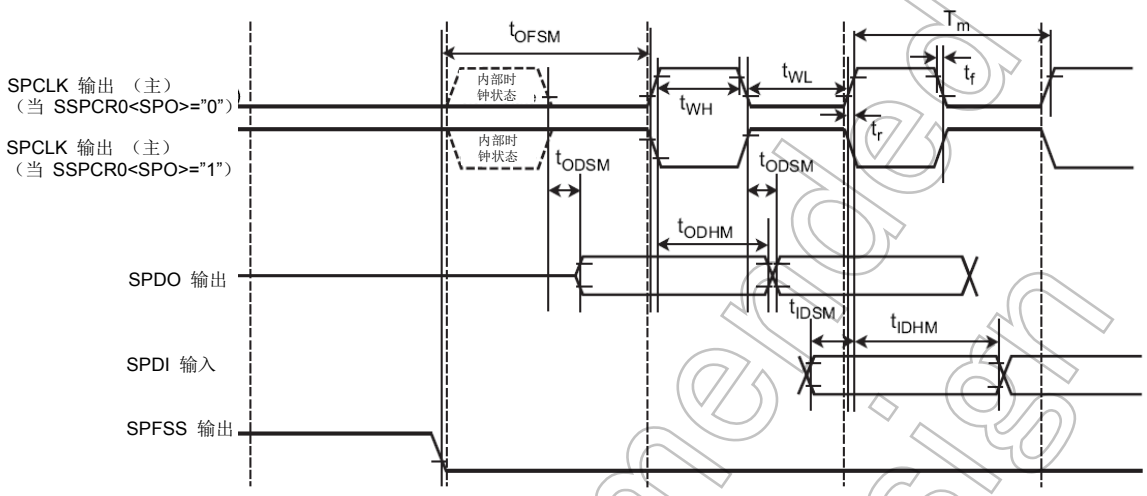
从模式：

n= (<CPSDVSR>× (1+<SCR>))= f_{sys} /f_{SPCLK} “n“设置要符合65024≥ n≥12

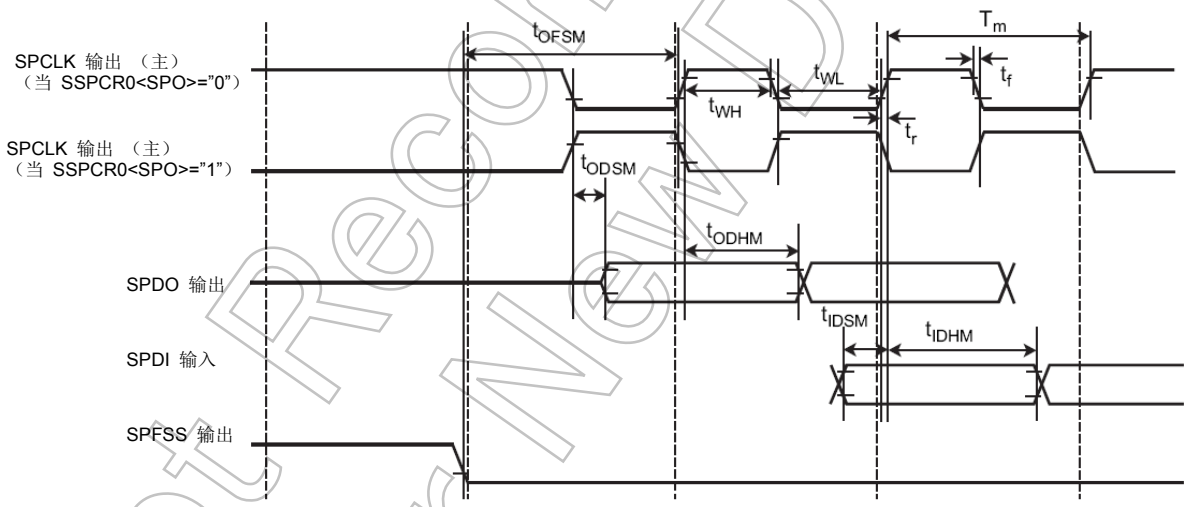
28.6.5.1 SSP SPI 模式 (主)

$fsys / 4 \geq f_{SPCLK} \geq fsys / 65024$

(1) 主 SSPCR0<SPH> = “0“ (数据在第一沿被锁存)



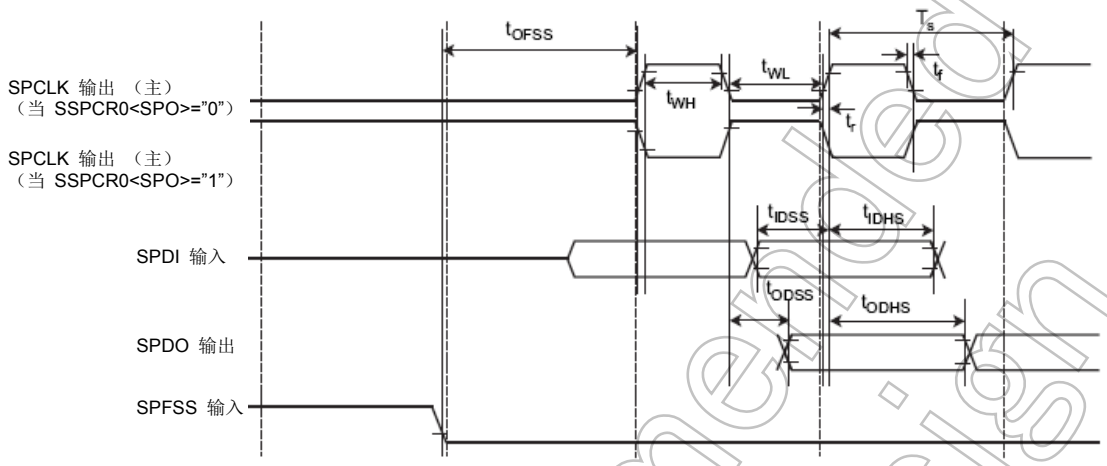
(2) 主 SSPCR0<SPH> = “1“ (数据在第二沿被锁存)



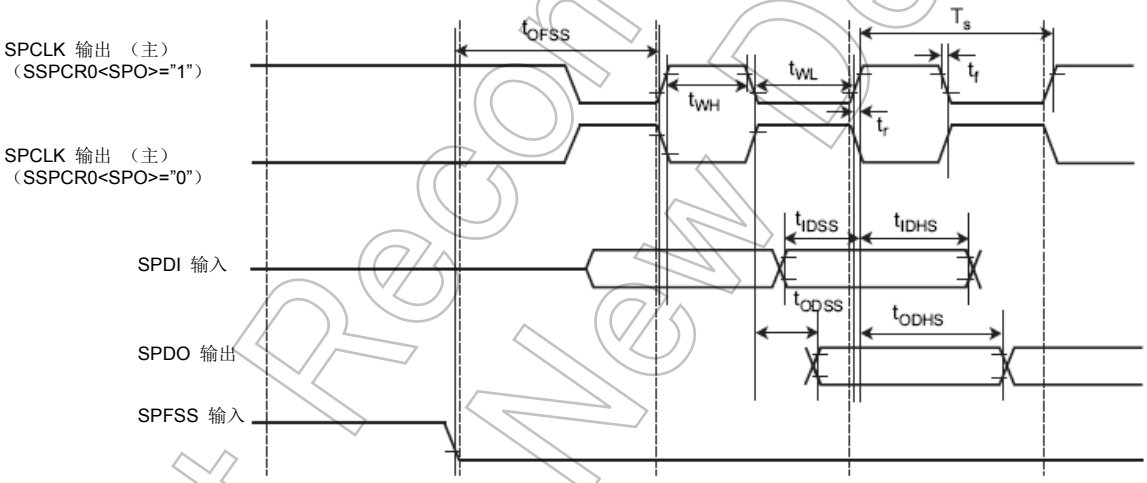
28.6.5.2 SSP SPI 模式 (从)

$$fsys / 12 \geq f_{SPCLK} \geq f_{sys} / 65024$$

(1) 从属 SSPCR0<SPH> = “0“ (数据在第一沿被锁存)



(2) 从属 SSPCR0<SPH> = “1“ (数据在第二沿被锁存)

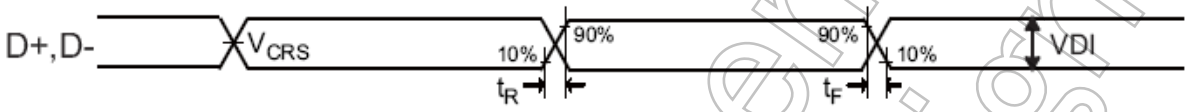


28.6.6 USB主机控制器

USB主机控制器的时钟与系统时钟fsys相同。

DVDD3A = 3.3±0.3V

参数	符号	最小值	最大值	单位
D+, D− 上升时间	t_R	4	20	ns
D+, D− 下降时间	t_F	4	20	
差分的正常模式范围	VDI	0.2	−	V
过信号电压	V_{CRS}	1.3	2.0	



28.6.7 16-位计时器 / 奇偶计数器

28.6.7.1 事件计数器

在下表中，字母x表示16-位计时器/奇偶计数器操作时钟周期时间与fsys周期时间相等。它根据时钟齿轮功能的编程而是不同。

参数	符号	等式		f_{sys} = 64 MHz		单位
		最小值	最大值	最小值	最大值	
时钟 低电平脉冲宽度	t_{VCKL}	2x + 100	−	131.3	−	ns
时钟高电平脉冲宽度	t_{VCKH}	2x + 100	−	131.3	−	ns

28.6.7.2 捕捉

在下表中，字母x表示16-位计时器/奇偶计数器操作时钟周期时间与f_{sys}周期时间相等。它根据时钟齿轮功能的编程而是不同。

参数	符号	等式		f_{sys} = 64 MHz		单位
		最小值	最大值	最小值	最大值	
时钟 低-电平脉冲宽度	t_{CPL}	2x + 100	−	131.3	−	ns
时钟 高-电平脉冲宽度	t_{CPH}	2x + 100	−	131.3	−	ns

28.6.8 外部中断

下表中，字母x表示f_{sys} 周期时间

1. 除STOP释放中断外

参数	符号	等式		f_{sys} = 64 MHz		单位
		最小值	最大值	最小值	最大值	
INT0 ~ D 低-电平脉冲宽度	t_{INTAL}	x + 100	-	115.6	-	ns
INT0 ~ D 高-电平脉冲宽度	t_{INTAH}	x + 100	-	115.6	-	ns

2. STOP释放中断

参数	符号	最小值	最大值	单位
INT0 ~ D 低-电平脉冲宽度	t_{INTBL}	100	-	ns
INT0 ~ D 高-电平脉冲宽度	t_{INTBH}	100	-	ns

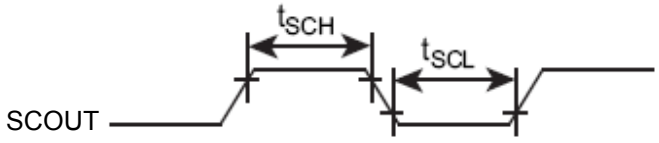
28.6.9 $\overline{NMI}$

参数	符号	最小值	最大值	单位
$\overline{NMI}$ 低电平脉冲宽度	t_{INTCL}	100	-	ns

28.6.10 SCOUT 引脚 AC 特征

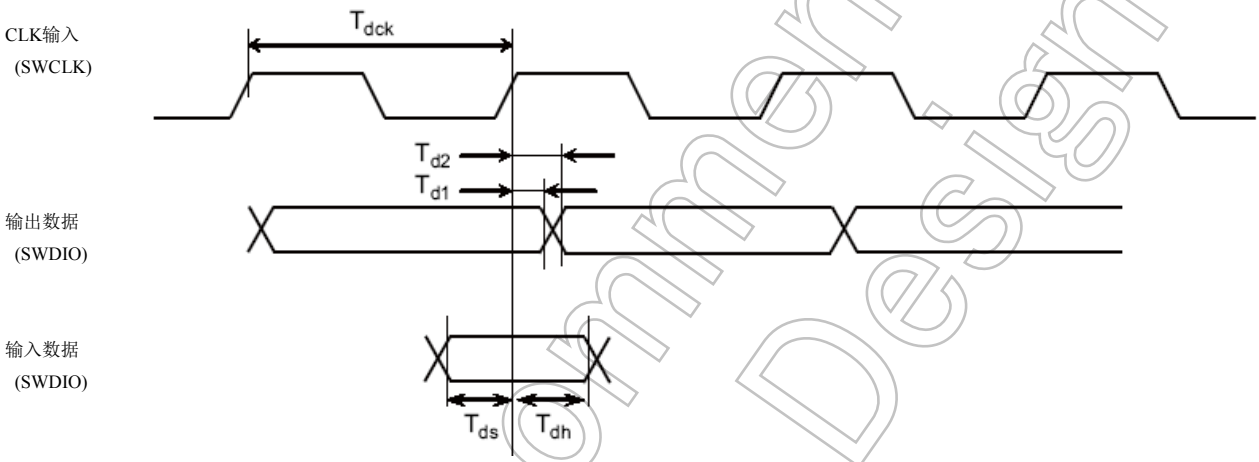
参数	符号	等式		f_{sys} = 48 MHz		单位
		最小值	最大值	最小值	最大值	
高-电平脉冲宽度	t_{SCH}	0.5T-5	-	5.5	-	ns
低-电平脉冲宽度	t_{SCL}	0.5T-5	-	5.5	-	ns

注:**在上表中，字母T表示SCOUT输出时钟的周期时间。**



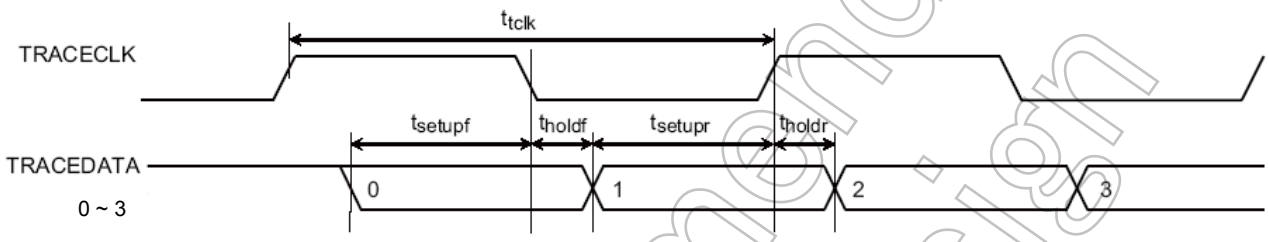
28.6.11 调试通信

参数	符号	最小值	最大值	单位
CLK 周期	T_{dck}	100	-	ns
CLK 上升 → 输出数据保留	T_{d1}	4	-	ns
CLK 上升 → 输出数据是效	T_{d2}	-	30	ns
输入数据是效→ CLK上升	T_{ds}	20	-	ns
CLK上升→ 输入数据保留	T_{dh}	15	-	ns



28.6.12 ETM 跟踪

参数	符号	最小值	最大值	单位
TRACECLK 周期	t_{tclk}	31.25	-	ns
TRACEDATA 有效 ← TRACECLK 上升	t_{setupr}	2	-	ns
TRACECLK 上升 → TRACEDATA 保留	t_{holdr}	1	-	ns
TRACEDATA 有效 ← TRACECLK 上升	t_{setupf}	2	-	ns
TRACECLK 下降 → TRACEDATA 保留	t_{holdf}	1	-	ns



28.7 闪存特性

28.7.1 擦除 / 写入特性

参数	条件	最小值	典型	最大值	单位
E / W 周期数	DVDD3 = AVDD3 = RVDD3 = 2.7 V ~ 3.6 V Ta = 0 ~ 70 °C	-	-	100	周期

28.8 振荡电路

振荡电路显示如下:

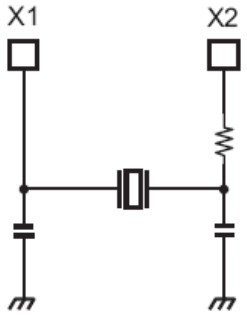


图 28-1 高-频率振荡连接

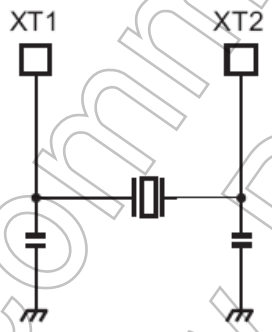


图 28-2 低-频率振荡连接

注: **获取表中振荡时，振荡器的负载能力和位置要调配到位。鉴于这些因素都受基片模式的影响，请所使用的基片来评估振荡稳定性。**

TX03由以下的振荡器卖商评估。当选择外部部件时请参考此信息

28.8.1 陶瓷振荡器

TX03推荐村田制造有限公司高-频率振荡器。

请参照以下URL获取详情。

http://www. murata. co. jp

28.8.2 晶体振荡器

TX03推荐京瓷晶体设备公司高-频率振荡器。

请参照以下URL获取详情。

http://www. kinseki. co. jp

28.8.3 设计印刷电路板注意事项

确保所设计的印刷电路板模式能链接一个晶体单位，其与其他振荡元件相接，由此该模式的长度可达到最短，以防止杂散电容和线路电感而造成的特性衰减。对于多层电路板，不要在振荡电路下面的地表及其他信号模式处圈线。

更多信息请浏览振荡器卖商的URL网址。

28.9 操作注意事项

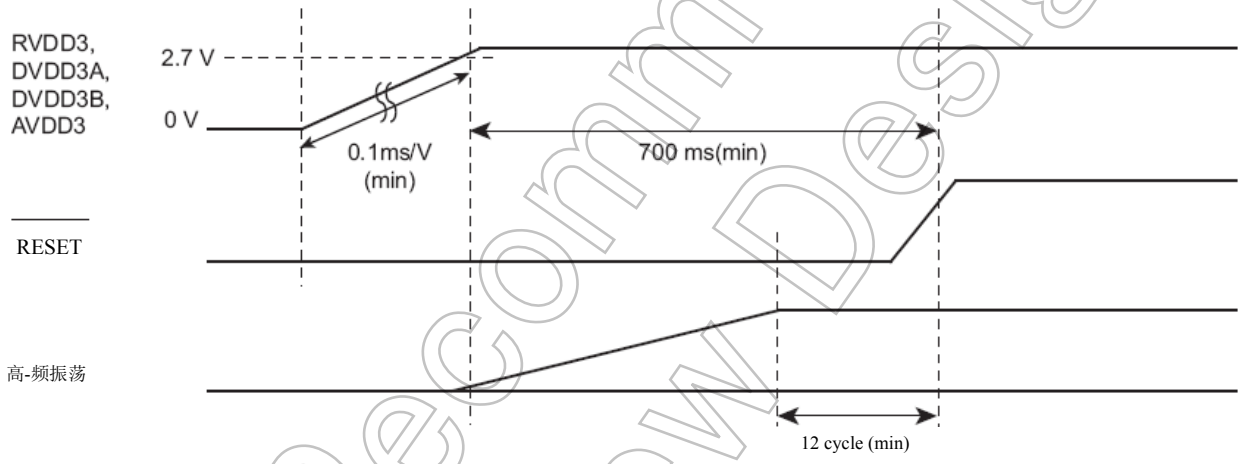
28.9.1 电源告示

28.9.1.1 通电须知

电源要升高 (从 0V ~ 2.7V)且速度为0.1ms/V或更低。

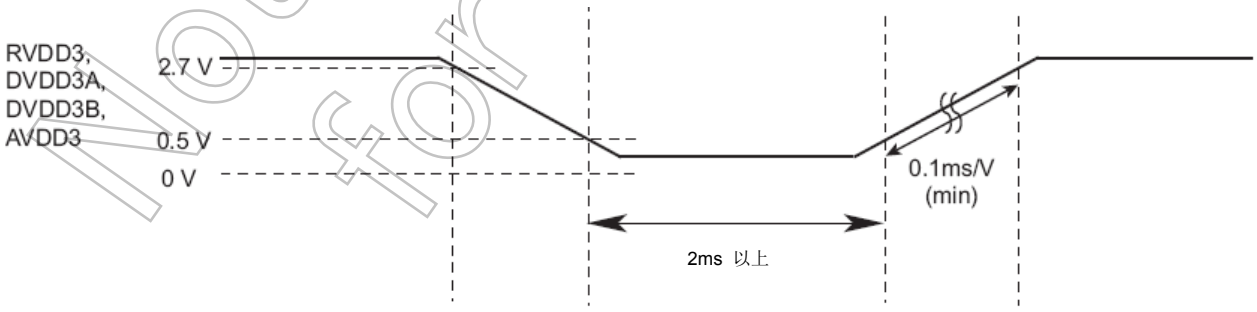
电源开启顺序必须考虑内部调节器与振荡器达到稳定的时间。在TX03，内部校准器要求至少有700 μs来达到稳定。

要获得稳定振荡所需要的时间因系统而异。冷复位时，外部复位引脚必须保持足够长的 “低“持续时间，使得内部调节器与振荡器达到稳定。
上电次序如下所示:



28.9.1.2 再次通电须知

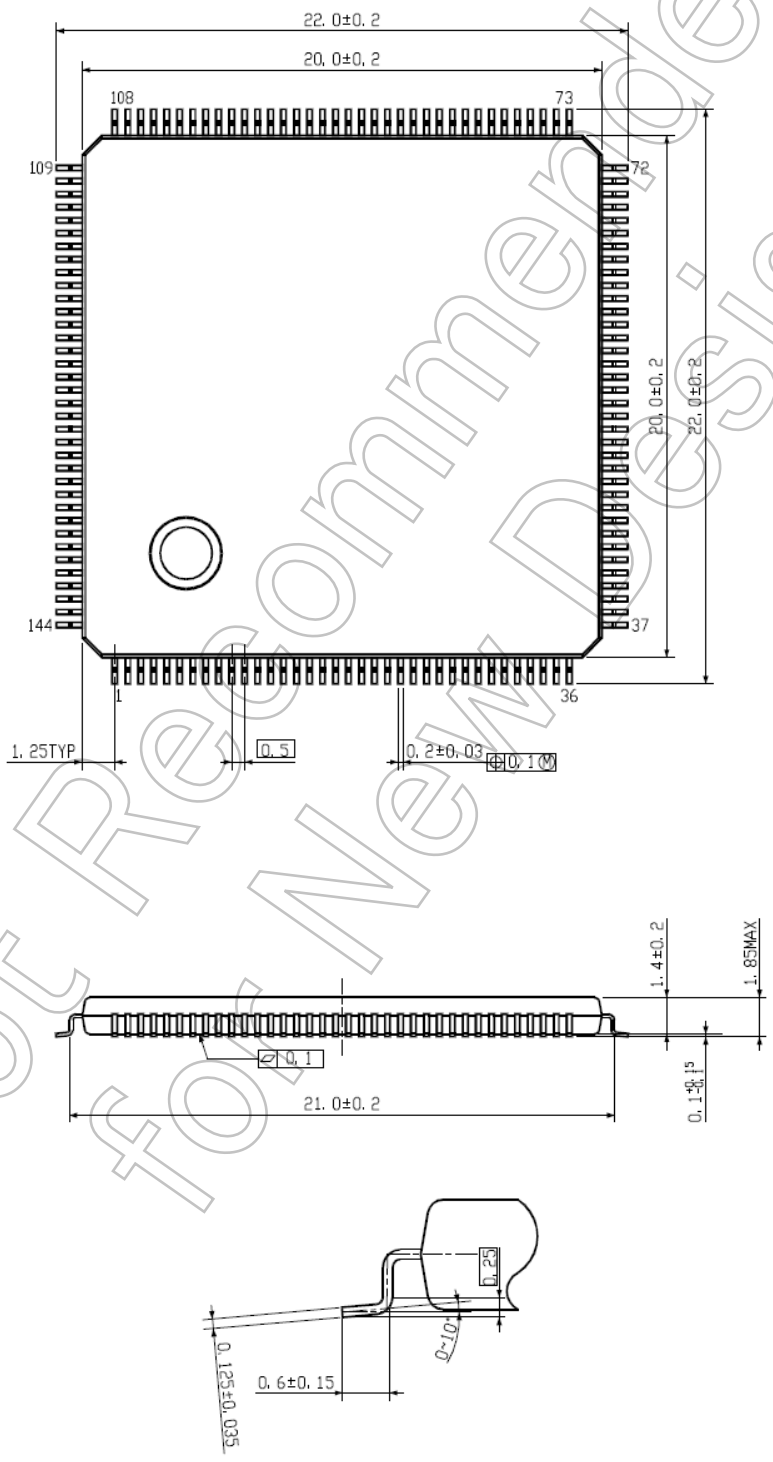
在断电后再上电之时，电源电压等于或小于0.5V并保持此水平2ms以上。此后，电源电压的上升速度当为0.1ms/V或更慢。应关复位信息见28.9.1.1。



29. 封装尺寸

型号: P-LQFP144-2020-0.50E

单位:mm



RESTRICTIONS ON PRODUCT USE

- Toshiba Corporation, and its subsidiaries and affiliates (collectively "TOSHIBA"), reserve the right to make changes to the information in this document, and related hardware, software and systems (collectively "Product") without notice.
- This document and any information herein may not be reproduced without prior written permission from TOSHIBA. Even with TOSHIBA's written permission, reproduction is permissible only if reproduction is without alteration/omission.
- Though TOSHIBA works continually to improve Product's quality and reliability, Product can malfunction or fail. Customers are responsible for complying with safety standards and for providing adequate designs and safeguards for their hardware, software and systems which minimize risk and avoid situations in which a malfunction or failure of Product could cause loss of human life, bodily injury or damage to property, including data loss or corruption. Before customers use the Product, create designs including the Product, or incorporate the Product into their own applications, customers must also refer to and comply with (a) the latest versions of all relevant TOSHIBA information, including without limitation, this document, the specifications, the data sheets and application notes for Product and the precautions and conditions set forth in the "TOSHIBA Semiconductor Reliability Handbook" and (b) the instructions for the application with which the Product will be used with or for. Customers are solely responsible for all aspects of their own product design or applications, including but not limited to (a) determining the appropriateness of the use of this Product in such design or applications; (b) evaluating and determining the applicability of any information contained in this document, or in charts, diagrams, programs, algorithms, sample application circuits, or any other referenced documents; and (c) validating all operating parameters for such designs and applications. **TOSHIBA ASSUMES NO LIABILITY FOR CUSTOMERS' PRODUCT DESIGN OR APPLICATIONS**.
- **PRODUCT IS NEITHER INTENDED NOR WARRANTED FOR USE IN EQUIPMENTS OR SYSTEMS THAT REQUIRE EXTRAORDINARILY HIGH LEVELS OF QUALITY AND/OR RELIABILITY, AND/OR A MALFUNCTION OR FAILURE OF WHICH MAY CAUSE LOSS OF HUMAN LIFE, BODILY INJURY, SERIOUS PROPERTY DAMAGE AND/OR SERIOUS PUBLIC IMPACT ("UNINTENDED USE").** Except for specific applications as expressly stated in this document, Unintended Use includes, without limitation, equipment used in nuclear facilities, equipment used in the aerospace industry, medical equipment, equipment used for automobiles, trains, ships and other transportation, traffic signaling equipment, equipment used to control combustions or explosions, safety devices, elevators and escalators, devices related to electric power, and equipment used in finance-related fields. **IF YOU USE PRODUCT FOR UNINTENDED USE, TOSHIBA ASSUMES NO LIABILITY FOR PRODUCT.** For details, please contact your TOSHIBA sales representative.
- Do not disassemble, analyze, reverse-engineer, alter, modify, translate or copy Product, whether in whole or in part.
- Product shall not be used for or incorporated into any products or systems whose manufacture, use, or sale is prohibited under any applicable laws or regulations.
- The information contained herein is presented only as guidance for Product use. No responsibility is assumed by TOSHIBA for any infringement of patents or any other intellectual property rights of third parties that may result from the use of Product. No license to any intellectual property right is granted by this document, whether express or implied, by estoppel or otherwise.
- **ABSENT A WRITTEN SIGNED AGREEMENT, EXCEPT AS PROVIDED IN THE RELEVANT TERMS AND CONDITIONS OF SALE FOR PRODUCT, AND TO THE MAXIMUM EXTENT ALLOWABLE BY LAW, TOSHIBA (1) ASSUMES NO LIABILITY WHATSOEVER, INCLUDING WITHOUT LIMITATION, INDIRECT, CONSEQUENTIAL, SPECIAL, OR INCIDENTAL DAMAGES OR LOSS, INCLUDING WITHOUT LIMITATION, LOSS OF PROFITS, LOSS OF OPPORTUNITIES, BUSINESS INTERRUPTION AND LOSS OF DATA, AND (2) DISCLAIMS ANY AND ALL EXPRESS OR IMPLIED WARRANTIES AND CONDITIONS RELATED TO SALE, USE OF PRODUCT, OR INFORMATION, INCLUDING WARRANTIES OR CONDITIONS OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, ACCURACY OF INFORMATION, OR NONINFRINGEMENT.**
- Do not use or otherwise make available Product or related software or technology for any military purposes, including without limitation, for the design, development, use, stockpiling or manufacturing of nuclear, chemical, or biological weapons or missile technology products (mass destruction weapons). Product and related software and technology may be controlled under the applicable export laws and regulations including, without limitation, the Japanese Foreign Exchange and Foreign Trade Law and the U.S. Export Administration Regulations. Export and re-export of Product or related software or technology are strictly prohibited except in compliance with all applicable export laws and regulations.
- Please contact your TOSHIBA sales representative for details as to environmental matters such as the RoHS compatibility of Product. Please use Product in compliance with all applicable laws and regulations that regulate the inclusion or use of controlled substances, including without limitation, the EU RoHS Directive. **TOSHIBA ASSUMES NO LIABILITY FOR DAMAGES OR LOSSES OCCURRING AS A RESULT OF NONCOMPLIANCE WITH APPLICABLE LAWS AND REGULATIONS.**