

译文

TMPM365FYXBG

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：“TMPM365FYXBG” 2013-06-12

翻译日: 2014-11-28

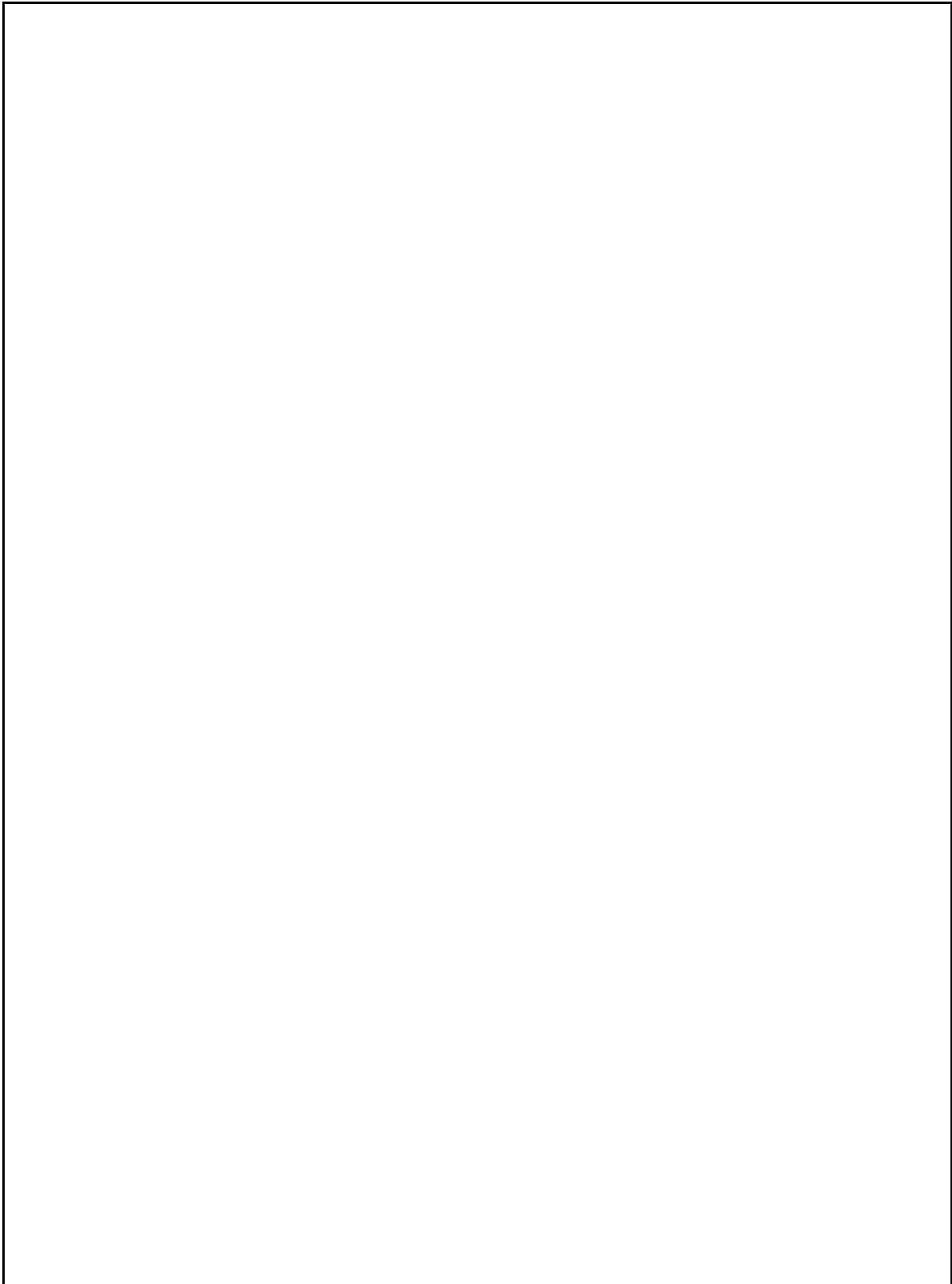
译文

TOSHIBA

32 位RISC微控制器
TX03 系列

TMPM365FYXBG

东芝公司
半导体&存储产品公司





ARM, ARM Powered, AMBA, ADK, ARM9TDMI, TDMI, PrimeCell, RealView, Thumb, Cortex, Coresight, ARM9, ARM926EJ-S, Embedded Trace Macrocell, ETM, AHB, APB, KEIL 是 ARM Limited 在 EU 及其他国家的注册商标或商标。

Important Notices

Make sure to read read this chapter before using the product.

1 Serial bus interface

There are restrictions on the use of I2C bus mode when the multi-master function is used.

1.1 Description

When the multi-master function is used in I2C bus mode, if these masters start the communications simultaneously, the following phenomena may occur:

1. Communications may be locked up.
2. SCL pulse widths shorten; therefore these pulses may not satisfy I2C Specifications.

1.2 Condition

These phenomena occur only when the multi-master function is used in I2C bus mode. If a single master is used, these phenomena do not occur.

1.3 Workaround

There is no workaround for these phenomena. Perform recovery process by software.

1.4 How to Recover from These Phenomena

Perform recovery process by software.

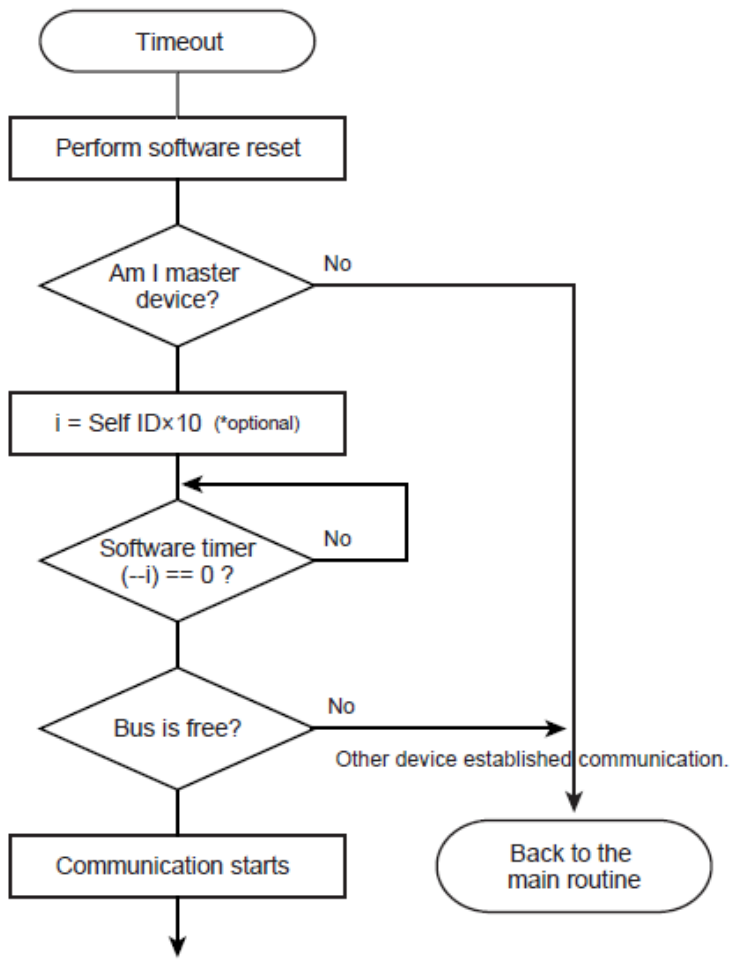
By using a timer, add timeout process to check whether communication is in a lock-up state.

An example of recovery process:

1. Start a timer count synchronously with start of the transmission.
2. If a serial interface interrupt (INTSBIx) does not occur in a certain period, the MCU determines the timeout.
3. If the MCU determines the timeout, communications may be locked up. Perform software reset on the serial bus interface circuit. This circuit is initialized to release communication from the lock up state.
4. Resend transmission data.

Mostly, Process 1 to 4 are enough to recovery; however if the multiple products are connected to the same bus line, add a delay time between each product's recovery process before Process 4 (resending data) is performed. This delay makes a time difference between each master; therefore bus collision can be avoided when the data is sent again.

Example: Recovery process after a timeout is detected.



2 Transitions to Low-power Consumption Mode and Generating a Non-maskable Interrupt

This chapter describes the precautions at which non-maskable interrupt (NMI) occurs when the MCU enters low-power consumption mode.

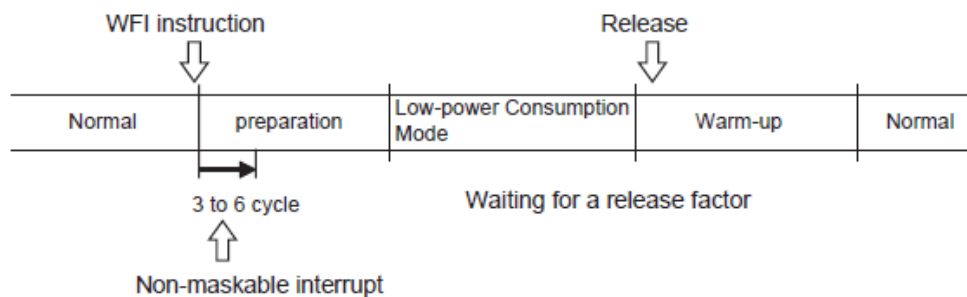
- STOP1

2.1 Description

When the WFI instruction is executed to enter the above-mentioned low-power consumption mode, if a nonmaskable interrupt (NMI) occurs, the MCU may enter low-power consumption mode without the process for releasing low-power consumption mode.

Note 1: An NMI notice and flag setting to the CPU are normal; therefore the NMI process after low-power consumption mode is released can be performed.

Note 2: When the MCU has entered low-power consumption mode, factors other than an NMI are accepted; however an NMI may not be accepted.



2.2 Conditions

- The WFI instruction is executed to enter low-power consumption mode.
- A non-maskable interrupt occurs after WFI instruction execution and within 3 to 6 cycles.

2.3 Workaround

Do not use a non-maskable interrupt as a release factor for the above-mentioned low-power consumption modes.

To avoid generating a non-maskable interrupt, the following settings should be specified before the MCU enters the above-mentioned low-power consumption mode.

- NMI pin: This input pin must be fixed to "High".
- Watchdog timer: Stop the watchdog timer or set the reset function.
- Voltage detection circuit: Stop the voltage detection circuit or set the reset function.

概述：对本规范项下SFR(特殊功能寄存器)的说明的注释

SFR (特殊功能寄存器) 是一种外围电路控制寄存器(IP)。

IP的SFR地址见存储器映像章节，SFR详见各IP章节。

本规范中采用的SFR的定义遵照下列规则。

- a. 以各IP的SFR表作为一个例子
 - 各IP章节中的SFR表列出了寄存器名称，地址和简要说明。
 - 所有寄存器具有一个 32 位的唯一地址。除一些例外情况外，寄存器地址定义如下："基址 + (唯一)地址"

基址 = 0x0000_0000

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

注：SAMCR寄存器地址宽 32 位，从地址 0x0000_0004 (基址(0x00000000) +唯一地址 (0x0004)) 起。

注：上述寄存器只是出于解释目的而给出的例子，不是为了证明目的。

在该微控制器中，该寄存器并不存在。

- b. SFR (寄存器)
 - 各寄存器基本由 32-位寄存器组成(有一些例外)。
 - 各寄存器的说明列出了位，比特符号，类型，复位后的初始值和功能。

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	读取 "0"。
9-7	MODE[2:0]	R/W	操作模式设置 000: 采样模式 0 001: 采样模式 1 010: 采样模式 2 011: 采样模式 3 非上述设置: 保留
6-0	TDATA[6:0]	W	传输的数据

注: 类型分成下列三类。

R/W	READ WRITE
R	READ
W	WRITE

c. 数据说明

说明中采用的符号的意义如下。

- x: 通道号/端口
- n, m: 位号

d. 寄存器说明

寄存器说明如下。

- 寄存器名称 <比特符号>
示例: SAMCR<MODE> = "000"或SAMCR<MODE[2:0]> = "000"
<MODE[2:0]>指在比特符号模式时第 2 位 ~ 第 0 位(3 位宽)。
- 寄存器名称[位]
示例: SAMCR[9:7] = "000"
它指寄存器SAMCR的第 9 位 ~ 第 7 位(3 位宽)。

译文

修订记录

日期	修订版	备注
2013/6/12	1	首次发布

译文

目录

概述：对本规范项下 SFR (特殊功能寄存器) 的说明的注释

TPM365FYXBG

1.1 特征	1
1.2 方块图	4
1.3 引脚分配 (顶视图)	5
1.4 引脚名称与功能	6
1.4.1 按引脚分类	6
1.4.2 按端口分类	12
1.5 引脚编号与电源引脚	18
1.6 内部稳压器引脚	18

2. 处理器内核

2.1 有关该处理器内核的信息	19
2.2 可配置的选项	19
2.3 异常/中断	20
2.3.1 中断输入的数目	20
2.3.2 优先级中断位的数目	20
2.3.3 SysTick	20
2.3.4 SYSRESETREQ	20
2.3.5 LOCKUP	20
2.3.6 辅助故障状态寄存器	20
2.4 事件	21
2.5 电源管理	21
2.6 独占通道	21

3. 调试接口

3.1 规范概述	23
3.2 SWJ-DP	23
3.3 ETM	23
3.4 引脚功能	24
3.5 在停止模式时的外设功能	25
3.6 与调试工具的连接	26
3.6.1 关于与调试工具的连接	26
3.6.2 将调试接口引脚用作通用端口时的要点	26

4. JTAG 接口

4.1	概述	27
4.2	信号汇总和连接的例子	28
4.3	概述	29
4.4	JTAG 控制器和寄存器	29
4.5	指令寄存器	30
4.6	边界扫描寄存器	31
4.7	测试接入端口 (TAP)	32
4.8	TAP 控制器	32
4.9	使 TAP 控制器复位	32
4.10	TAP 控制器状态转换	33
4.11	边界扫描顺序	36
4.12	JTAG 控制器单元支持的指令	37

5. 存储器地址

5.1	存储器地址	41
5.1.1	TMPM365FYXBG 存储器地址	42
5.2	SFR 区域细节	43

6. 复位

6.1	初始状态	45
6.2	冷复位	45
6.3	热复位	46
6.3.1	复位时间	46
6.4	复位后	46

7. 看门狗定时器 (WDT)

7.1	配置	47
7.2	寄存器	48
7.2.1	WDMOD (看门狗定时器寄存器地址)	48
7.2.2	WDCR (看门狗定时器控制寄存器)	51
7.3	操作	52
7.3.1	基本运行	52
7.3.2	运行模式和状态	52
7.4	在检测到故障(失控)时的运行情况	53
7.4.1	INTWDT 中断生成	53
7.4.2	内部复位生成	54
7.5	控制寄存器	55
7.5.1	看门狗定时器寄存器地址 (WDMOD)	55
7.5.2	看门狗定时器控制寄存器(WDCR)	55
7.5.3	设置例子	56
7.5.3.1	禁用控制	
7.5.3.2	启用控制	
7.5.3.3	看门狗定时器清除控制	
7.5.3.4	看门狗定时器的检测时间	

8. 时钟/模式控制

8.1 特征	58
8.2 寄存器	59
8.2.1 寄存器列表.....	59
8.2.2 CGSYSCR (系统控制寄存器).....	60
8.2.3 CGOSCCR (振荡控制寄存器).....	62
8.2.4 CGSTBYCR (待机控制寄存器).....	64
8.2.5 CGPLLSEL (PLL 选择寄存器).....	65
8.2.6 CGUSBCTL (USB 时钟控制寄存器).....	66
8.2.7 CGPROTECT (保护寄存器).....	67
8.3 时钟控制	68
8.3.1 时钟类型.....	68
8.3.2 复位之后的初始值.....	68
8.3.3 时钟系统图.....	69
8.3.4 预热功能.....	70
8.3.5 时钟乘法电路 (PLL).....	72
8.3.5.1 启动运行.....	
8.3.6 系统时钟.....	74
8.3.6.1 系统时钟设置.....	
8.3.7 预分频时钟控制.....	76
8.3.8 系统时钟引脚输出功能.....	76
8.4 模式与模式推移	77
8.4.1 模式推移.....	77
8.5 运行模式	78
8.5.1 NORMAL 模式.....	78
8.6 低功耗模式	78
8.6.1 IDLE 模式.....	78
8.6.2 STOP1 模式.....	79
8.6.3 低功耗模式设置.....	80
8.6.4 各模式下的操作状态.....	81
8.6.5 释放低功耗模式.....	82
8.6.6 预热.....	83
8.6.7 模式切换时的时钟操作.....	84
8.6.7.1 操作模式切换: NORMAL → STOP1 → NORMAL.....	

9. 异常

9.1 概述	85
9.1.1 异常类型.....	85
9.1.2 处理流程图.....	86
9.1.2.1 异常请求与检测.....	
9.1.2.2 异常处理与转入该中断服务程序 (先占).....	
9.1.2.3 执行 ISR.....	
9.1.2.4 异常出口.....	
9.2 复位异常	91
9.3 非屏蔽中断 (NMI)	92
9.4 SysTick	92
9.5 中断	93
9.5.1 中断源.....	93
9.5.1.1 中断路径.....	
9.5.1.2 生成.....	
9.5.1.3 传输.....	
9.5.1.4 使用外部中断引脚时的预防措施.....	
9.5.1.5 中断源列表.....	
9.5.1.6 有效电平.....	
9.5.2 中断处理.....	98
9.5.2.1 流程图.....	
9.5.2.2 准备工作.....	
9.5.2.3 由时钟发生器进行检测.....	
9.5.2.4 由 CPU 进行检测.....	
9.5.2.5 CPU 处理.....	
9.5.2.6 中断服务程序(ISR).....	
9.6 例外情况/与中断相关的寄存器	104
9.6.1 寄存器列表.....	104

9.6.2	NVIC 寄存器	105
9.6.2.1	SysTick 控制器与状态寄存器	
9.6.2.2	SysTick 重新加载值寄存器	
9.6.2.3	SysTick 当前值寄存器	
9.6.2.4	SysTick 校准值寄存器	
9.6.2.5	中断设置-启用寄存器 1	
9.6.2.6	中断设置-启用寄存器 2	
9.6.2.7	中断清除-启用寄存器 1	
9.6.2.8	中断清除-启用寄存器 2	
9.6.2.9	中断设置-挂起寄存器 1	
9.6.2.10	中断设置-挂起寄存器 2	
9.6.2.11	中断清除-挂起寄存器 1	
9.6.2.12	中断清除-挂起寄存器 2	
9.6.2.13	中断优先权寄存器	
9.6.2.14	矢量表位移寄存器	
9.6.2.15	应用中断与复位控制寄存器	
9.6.2.16	系统处理器优先寄存器	
9.6.2.17	系统处理器控制器与状态寄存器	
9.6.3	时钟发生器寄存器	128
9.6.3.1	CGIMCGA (CG 中断模式控制寄存器 A)	
9.6.3.2	CGIMCGB (CG 中断模式控制寄存器 B)	
9.6.3.3	CGIMCGC (CG 中断模式控制寄存器 C)	
9.6.3.4	CGICRCG (CG 中断请求清除寄存器)	
9.6.3.5	CGNMIFLG (NMI 标志寄存器)	
9.6.3.6	CGRSTFLG (复位 标志寄存器)	

10. 输入/输出端口

10.1	端口功能	137
10.1.1	功能清单	137
10.1.2	端口寄存器概述	141
10.1.3	STOP1 模式下端口状态	142
10.2	端口功能	143
10.2.1	端口 A (PA0 ~ PA7)	143
10.2.1.1	端口 A 寄存器	
10.2.1.2	PADATA (端口 A 数据寄存器)	
10.2.1.3	PACR (端口 A 输出控制寄存器)	
10.2.1.4	PAOD (端口 A 开漏控制寄存器)	
10.2.1.5	PAPUP (端口 A 上拉控制寄存器)	
10.2.1.6	PAIE (端口 A 输入控制寄存器)	
10.2.2	端口 B (PB0 ~ PB7)	147
10.2.2.1	端口 B 寄存器	
10.2.2.2	PBDATA (端口 B 数据寄存器)	
10.2.2.3	PBCR (端口 B 输出控制寄存器)	
10.2.2.4	PBOD (端口 B 开漏控制寄存器)	
10.2.2.5	PBPUP (端口 B 上拉控制寄存器)	
10.2.2.6	PBIE (端口 B 输入控制寄存器)	
10.2.3	端口 C (PC0 ~ PC2)	152
10.2.3.1	端口 C 寄存器	
10.2.3.2	PCDATA (端口 C 数据寄存器)	
10.2.3.3	PCCR (端口 C 输出控制寄存器)	
10.2.3.4	PCFR1 (端口 C 功能寄存器 1)	
10.2.3.5	PCFR3 (端口 C 功能寄存器 3)	
10.2.3.6	PCFR4 (端口 C 功能寄存器 4)	
10.2.3.7	PCOD (端口 C 开漏控制寄存器)	
10.2.3.8	PCPUP (端口 C 上拉控制寄存器)	
10.2.3.9	PCIE (端口 C 输入控制寄存器)	
10.2.4	端口 D (PD0 ~ PD7)	157
10.2.4.1	端口 D 寄存器	
10.2.4.2	PDDATA (端口 D 数据寄存器)	
10.2.4.3	PDCR (端口 D 输出控制寄存器)	
10.2.4.4	PDFR3 (端口 D 功能寄存器 3)	
10.2.4.5	PDOD (端口 D 开漏控制寄存器)	
10.2.4.6	PDPUP (端口 D 上拉控制寄存器)	
10.2.4.7	PDIE (端口 D 输入控制寄存器)	
10.2.5	端口 E (PE0 ~ PE7)	162
10.2.5.1	端口 E 寄存器	
10.2.5.2	PEDATA (端口 E 数据寄存器)	
10.2.5.3	PECR (端口 E 输出控制寄存器)	
10.2.5.4	PEFR1 (端口 E 功能寄存器 1)	
10.2.5.5	PEFR3 (端口 E 功能寄存器 3)	
10.2.5.6	PEFR4 (端口 E 功能寄存器 4)	

译文

10.2.5.7	PEOD (端口 E 开漏控制寄存器)	
10.2.5.8	PEPUP (端口 E 上拉控制寄存器)	
10.2.5.9	PEIE (端口 E 输入控制寄存器)	
10.2.6	端口 F (PF0 ~ PF7)	169
10.2.6.1	F 端口寄存器	
10.2.6.2	PFDATA (F 端口数据寄存器)	
10.2.6.3	PF0CR (F 端口 输出控制寄存器)	
10.2.6.4	PF0FR2 (F 端口功能寄存器 2)	
10.2.6.5	PF0FR3 (F 端口功能寄存器 3)	
10.2.6.6	PF0OD (F 端口开漏控制寄存器)	
10.2.6.7	PF0PUP (F 端口上拉控制寄存器)	
10.2.6.8	PF0IE (F 端口输入控制寄存器)	
10.2.7	端口 G (PG0 ~ PG5)	177
10.2.7.1	端口 G 寄存器	
10.2.7.2	PGDATA (端口 G 数据寄存器)	
10.2.7.3	PG0CR (端口 G 输出控制寄存器)	
10.2.7.4	PG0FR1 (端口 G 功能寄存器 1)	
10.2.7.5	PG0FR3 (端口 G 功能寄存器 3)	
10.2.7.6	PG0FR4 (端口 G 功能寄存器 4)	
10.2.7.7	PG0OD (端口 G 开漏控制寄存器)	
10.2.7.8	PG0PUP (端口 G 上拉控制寄存器)	
10.2.7.9	PG0IE (端口 G 输入控制寄存器)	
10.2.8	端口 H (PH0 ~ PH4)	183
10.2.8.1	端口 H 寄存器	
10.2.8.2	PHDATA (端口 H 数据寄存器)	
10.2.8.3	PH0CR (端口 H 输出控制寄存器)	
10.2.8.4	PH0FR1 (端口 H 功能寄存器 1)	
10.2.8.5	PH0FR3 (端口 H 功能寄存器 3)	
10.2.8.6	PH0OD (端口 H 开漏控制寄存器)	
10.2.8.7	PH0PUP (端口 H 上拉控制寄存器)	
10.2.8.8	PH0IE (端口 H 输入控制寄存器)	
10.2.9	端口 I (PI0 ~ PI7)	188
10.2.9.1	端口 I 寄存器	
10.2.9.2	PIDATA (端口 I 数据寄存器)	
10.2.9.3	PI0CR (端口 I 输出控制寄存器)	
10.2.9.4	PI0FR1 (端口 I 功能寄存器 1)	
10.2.9.5	PI0OD (端口 I 开漏控制寄存器)	
10.2.9.6	PI0PUP (端口 I 上拉控制寄存器)	
10.2.9.7	PI0PDN (端口 I 下拉控制寄存器)	
10.2.9.8	PI0IE (端口 I 输入控制寄存器)	
10.2.10	端口 J (PJ0 ~ PJ7)	196
10.2.10.1	端口 J 寄存器	
10.2.10.2	PJDATA (端口 J 数据寄存器)	
10.2.10.3	PJ0CR (端口 J 输出控制寄存器)	
10.2.10.4	PJ0FR2 (端口 J 功能寄存器 2)	
10.2.10.5	PJ0FR3 (端口 J 功能寄存器 3)	
10.2.10.6	PJ0PUP (端口 J 上拉控制寄存器)	
10.2.10.7	PJ0IE (端口 J 输入控制寄存器)	
10.2.11	端口 K (PK0~PK3)	200
10.2.11.1	端口 K 寄存器	
10.2.11.2	PKDATA (端口 K 数据寄存器)	
10.2.11.3	PK0CR (端口 K 输出控制寄存器)	
10.2.11.4	PK0FR2 (端口 K 功能寄存器 2)	
10.2.11.5	PK0FR3 (端口 K 功能寄存器 3)	
10.2.11.6	PK0PUP (端口 K 上拉控制寄存器)	
10.2.11.7	PK0IE (端口 K 输入控制寄存器)	
10.3	端口方块图	205
10.3.1	端口类型	205
10.3.2	类型 FT1	206
10.3.3	类型 FT2	207
10.3.4	类型 FT3	208
10.3.5	类型 FT4	209
10.3.6	类型 FT5	210
10.3.7	类型 FT6	211
10.4	附录 (端口设置列表)	212
10.4.1	端口 A 设置	212
10.4.2	端口 B 设置	213
10.4.3	端口 C 设置	214
10.4.4	端口 D 设置	215
10.4.5	端口 E 设置	216
10.4.6	端口 E 设置	217
10.4.7	端口 G 设置	218
10.4.8	端口 H 设置	219

10.4.9 端口 I 设置.....	220
10.4.10 端口 J 设置.....	221
10.4.11 端口 K 设置.....	222

11. DMA 控制器 (DMAC)

11.1 概述.....	223
11.2 DMA 传输方式.....	224
11.3 方块图.....	225
11.4 TMPM365FYXBG 产品信息.....	226
11.4.1 以外设-外设传输方式支持的外设功能.....	226
11.4.2 DMA 请求.....	226
11.4.3 中断请求.....	227
11.4.4 寄存器基址.....	227
11.5 寄存器说明.....	228
11.5.1 DMAC 寄存器列表.....	228
11.5.2 DMACxIntStatus (DMAC 中断状态寄存器).....	229
11.5.3 DMACxIntTCStatus (DMAC 中断端子计数状态寄存器).....	231
11.5.4 DMACxIntTClear(DMAC 中断端子计数清除寄存器).....	232
11.5.5 DMACxIntErrorStatus (DMAC 中断错误状态寄存器).....	233
11.5.6 DMACxIntErrClr(DMAC 中断错误清除寄存器).....	234
11.5.7 DMACxRawIntTC 状态 (DMACRaw 中断端子计数状态寄存器).....	235
11.5.8 DMACxRawIntErrorStatus (DMACRaw 错误中断状态寄存器).....	236
11.5.9 DMACxEnbldChns (DMAC 启用通道寄存器).....	237
11.5.10 DMACxSoftBReq (DMAC 软件突发请求寄存器).....	238
11.5.11 DMACxSoftSReq (DMAC 软件单一请求寄存器).....	240
11.5.12 DMACxConfiguration (DMAC 配置寄存器).....	242
11.5.13 DMACxCnSrcAddr (DMAC 通道 x 源地址寄存器).....	243
11.5.14 DMACxCnDestAddr (DMAC 通道 x 目的地址寄存器).....	244
11.5.15 DMACxCnLLI (DMAC 通道 x 链表项目寄存器).....	245
11.5.16 DMACxCnControl (DMAC 通道 n 控制寄存器).....	246
11.5.17 DMACxCnConfiguration (DMAC 通道 n 配置寄存器).....	249
11.6 特殊功能.....	252
11.6.1 分散/聚集功能.....	252
11.6.2 链表操作.....	253

12. 16 位定时器/事件计数器(TMRB)

12.1 概要.....	256
12.2 规格差异.....	257
12.3 配置.....	258
12.4 寄存器.....	260
12.4.1 通道对应寄存器列表.....	260
12.4.2 TBxEN (启用寄存器).....	261
12.4.3 TBxRUN ("运行"寄存器).....	262
12.4.4 TBxCR (控制寄存器).....	263
12.4.5 TBxMOD (模式寄存器).....	264
12.4.6 TBxFFCR (触发器控制寄存器).....	266
12.4.7 TBxST (状态寄存器).....	267
12.4.8 TBxIM (中断屏蔽寄存器).....	268
12.4.9 TBxUC (递增计数器捕捉寄存器).....	269
12.4.10 TBxRG0 (定时器寄存器 0).....	270
12.4.11 TBxRG1 (定时器寄存器 1).....	270
12.4.12 TBxCP0 (捕捉寄存器 0).....	271
12.4.13 TBxCP1(捕捉寄存器 1).....	271
12.4.14 TBxDMA(DMA 请求启用寄存器).....	272
12.5 每个电路操作说明.....	273
12.5.1 预分频器.....	273
12.5.2 递增计数器 (UC).....	279

12.5.3	定时器寄存器 (TBxRG0, TBxRG1).....	279
12.5.4	捕捉.....	280
12.5.5	捕捉寄存器 (TBxCP0, TBxCP1).....	280
12.5.6	递增计数器捕捉寄存器 (TBxUC).....	280
12.5.7	比较器 (CP0, CP1).....	280
12.5.8	定时器触发器 (TBxFF0).....	280
12.5.9	捕捉中断 (INTCAPx0, INTCAPx1).....	280
12.6	各个模式操作说明	281
12.6.1	16-位间隔计时器模式.....	281
12.6.2	16-位事件计数器模式.....	281
12.6.3	16-位 PPG(可编程脉冲发生)输出模式.....	282
12.6.5	外部触发器计数起动脉模式.....	284
12.7	利用捕捉功能的应用	285
12.7.1	由外部脉冲触发的单脉冲.....	285
12.7.2	频率测量.....	287
12.7.3	脉冲宽度测量.....	287
12.7.4	时间差测量.....	288

13. USB 装置控制器(USBD)

13.1	概述	289
13.2	系统结构	290
13.2.1	AHB 总线桥(UDC2AB).....	291
13.2.1.1	功能和特性	
13.2.1.2	配置	
13.2.1.3	时钟域	
13.2.2	东芝 USB-Spec2.0 装置控制器(UDC2).....	296
13.2.2.1	特点和功能	
13.2.2.2	标志规范	
13.2.2.3	进入 EP 的命令	
13.3	如何连接 USB 总线	303
13.4	寄存器	304
13.4.1	UDC2AB 寄存器	
13.4.1.1	UDC2AB 寄存器列表	
13.4.1.2	UDFSINTSTS(中断状态寄存器)	
13.4.1.3	UDFSINTENB(中断启用寄存器)	
13.4.1.4	UDFSMWTOUR(主机写入超时寄存器)	
13.4.1.5	UDFSC2STSET(UDC2 设置寄存器)	
13.4.1.6	UDFSMSTSET(DMAC 设置寄存器)	
13.4.1.7	UDFSDMACRDREQ(DMAC 读取请求寄存器)	
13.4.1.8	UDFSDMACRDVLD(DMAC 读取值寄存器)	
13.4.1.9	UDFSUDC2RDREQ(UDC2 读取请求寄存器)	
13.4.1.10	UDFSUDC2RDVLD(UDC2 读取值寄存器)	
13.4.1.11	UDFSARBTSSET(判优器设置寄存器)	
13.4.1.12	UDFSMWSADR(主机写入起始地址寄存器)	
13.4.1.14	UDFSMWCADR(主机写入当前地址寄存器)	
13.4.1.15	UDFSMWAHBADR(主机写入 AHB 地址寄存器)	
13.4.1.16	UDFSMRSADR(主机读取起始地址寄存器)	
13.4.1.17	UDFSMREADR(主机读取结束地址寄存器)	
13.4.1.18	UDFSMRCADR(主机读取当前地址寄存器)	
13.4.1.19	UDFSMRAHBADR(主机读取 AHB 地址寄存器)	
13.4.1.20	UDFSPWCTL(功率检测控制寄存器)	
13.4.1.21	UDFSMSTSTS(主机状态寄存器)	
13.4.1.22	UDFSTOUTCNT(超时计数寄存器)	
13.4.2	UDC2 寄存器.....	329
13.4.2.1	UDC2 寄存器	
13.4.2.2	如何访问 UDC2 寄存器	
13.4.2.3	UDFS2ADR(地址状态寄存器)	
13.4.2.4	UDFS2FRM(帧寄存器)	
13.4.2.5	UDFS2CMD(命令寄存器)	
13.4.2.6	UDFS2BRQ(bRequest-bmRequest 型寄存器)	
13.4.2.7	UDFS2WVL(wValue 寄存器)	
13.4.2.8	UDFS2WIDX(wIndex 寄存器)	
13.4.2.9	UDFS2WLGTH(wLength 寄存器)	
13.4.2.10	UDFS2INT(INT 寄存器)	
13.4.2.11	UDFS2INTEP(INT_EP 寄存器)	

13.4.2.12	UDFS2INTEPMSK(INT_EP_MASK 寄存器)	
13.4.2.13	UDFS2INTRX0(INT_RX_DATA0 寄存器)	
13.4.2.14	UDFS2INTNAK(INT_NAK 寄存器)	
13.4.2.15	UDFS2INTNAKMSK(INT_NAK_MASK 寄存器)	
13.4.2.16	UDFS2EP0MSZ(EP0_MaxPacketSize 寄存器)	
13.4.2.17	UDFS2EP0STS(EP0_Status 寄存器)	
13.4.2.18	UDFS2EP0DSZ(EP0_Datasize 寄存器)	
13.4.2.19	UDFS2EP0FIFO(EP0_FIFO 寄存器)	
13.4.2.20	UDFS2EPxMSZ(EPx_MaxPacketSize 寄存器)	
13.4.2.21	UDFS2EPxSTS(EPx_Status 寄存器)	
13.4.2.22	UDFS2EPxDSZ(EPx_Datasize 寄存器)	
13.4.2.23	UDFS2EPxFIFO(EPx_FIFO 寄存器)	
13.5	UDC2AB 操作描述	359
13.5.1	复位	359
13.5.2	中断信号	360
13.5.2.1	INTUSB 中断信号	
13.5.2.2	INTUSBWKUP 中断	
13.5.3	操作顺序	362
13.5.4	主机写入传输操作	364
13.5.4.1	主机读取传输	
13.5.4.2	主机写入传输操作	
13.5.5	USB 电源管理控制	368
13.5.5.1	电源管理控制信号连接图	
13.5.5.2	USB 总线电源 (VBUS) 连接/切断顺序	
13.5.6	USB 复位	369
13.5.7	暂停/恢复	370
13.5.7.1	切换到暂停状态	
13.5.7.2	从暂停状态恢复 (从 USB 主机恢复)	
13.5.7.3	从暂停状态 (切断) 恢复	
13.5.7.4	远程唤醒暂停状态	
13.6	USB 装置响应	377
13.7	EP 传输的控制流程	379
13.7.1	EP0	379
13.7.1.1	Control-RD 控制读传输	
13.7.1.2	Control-WR 控制写传输(非数据阶段)	
13.7.1.3	Control-WR 传输(带 DATA-阶段)	
13.7.1.4	使用 INT_STATUS_NAK 标志示例	
13.7.1.5	处理标准请求	
13.7.2	EPs (EP0 除外)	393
13.8	暂停/恢复状态	394
13.8.1	转入暂停状态	394
13.8.2	从暂停状态恢复	394
13.8.2.1	通过主机输出实现恢复	
13.8.2.2	通过 UDC2 远程唤醒实现恢复	
13.9	USB-Spec2.0 设备控制器附录	395
13.9.1	附录 A 系统电源管理	395
13.9.1.1	连接/断开操作	
13.9.1.2	复位操作	
13.9.1.3	暂停操作	
13.9.1.4	恢复操作	
13.9.2	与 "将奇数个字节设置为 MaxPacketSize" 有关的附录 B	401
13.9.2.1	在 UDFS2EPxMSZ 中设置一个奇数	
13.9.3	附录 C 等时译码器	405
13.9.3.1	利用等时传输访问某个 EP	
13.9.3.2	在采用等时传输时对针对 EP 的命令用法的限制	

14. 串行通道(SIO/UART)

14.1	概述	406
14.2	SIO 模块的规格差异	406
14.3	配置	407
14.4	寄存器说明	408
14.4.1	各通道中的寄存器列表	408
14.4.2	SCxEN (启用寄存器)	409
14.4.3	SCxBUF (缓冲寄存器)	410
14.4.4	SCxCR (控制寄存器)	411

14.4.5	SCxMOD0 (模式控制寄存器 0).....	413
14.4.6	SCxMOD1 (模式控制寄存器 1).....	414
14.4.7	SCxMOD2 (模式控制寄存器 2).....	415
14.4.8	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).....	418
14.4.9	SCxFCNF (FIFO 配置寄存器).....	420
14.4.10	SCxRFC (RX FIFO 配置寄存器).....	421
14.4.11	SCxTFC (TX FIFO 配置寄存器) (注 2).....	422
14.4.12	SCxRST (RX FIFO 状态寄存器).....	423
14.4.13	SCxTST (TX FIFO 状态寄存器).....	424
14.4.14	SCxDMA (DMA 请求启用寄存器).....	425
14.5	在各模式下的操作	426
14.6	数据格式	427
14.6.1	数据格式列表.....	427
14.6.2	奇偶控制.....	428
14.6.2.1	发送.....	
14.6.2.2	接收数据.....	
14.6.3	停止位长度.....	428
14.7	时钟控制	429
14.7.1	预分频器.....	429
14.7.2	串行时钟生成电路.....	433
14.7.2.1	波特率发生器.....	
14.7.2.2	时钟选择电路.....	
14.8	发送/接收缓冲器和 FIFO	438
14.8.1	配置.....	438
14.8.2	发送/接收缓冲器.....	438
14.8.3	FIFO.....	439
14.9	状态标志	439
14.10	错误标志	439
14.10.1	OERR 标志.....	440
14.10.2	PERR 标志.....	440
14.10.3	FERR 标志.....	440
14.11	接收	441
14.11.1	接收计数器.....	441
14.11.2	接收控制装置.....	441
14.11.2.1	I/O 接口模式.....	
14.11.2.2	UART 模式.....	
14.11.3	接收操作.....	441
14.11.3.1	接收缓冲器.....	
14.11.3.2	接收 FIFO 操作.....	
14.11.3.3	I/O 接口模式(具备 SCLK 输出).....	
14.11.3.4	读取已接收数据.....	
14.11.3.5	唤醒功能.....	
14.11.3.6	溢出错误.....	
14.12	发送	445
14.12.1	发送计数器.....	445
14.12.2	发送控制.....	445
14.12.2.1	I/O 接口模式.....	
14.12.2.2	UART 模式.....	
14.12.3	发送操作.....	445
14.12.3.1	发送缓冲器的操作.....	
14.12.3.2	发送 FIFO 操作.....	
14.12.3.3	I/O 接口模式/通过 SCLK 输出实现发送.....	
14.12.3.4	欠载操作错误.....	
14.13	握手功能	449
14.14	中断/错误生成计时	450
14.14.1	RX 中断.....	450
14.14.1.1	单缓冲器/双缓冲器.....	
14.14.1.2	FIFO.....	
14.14.2	TX 中断.....	451
14.14.2.1	单缓冲器/双缓冲器.....	
14.14.2.2	FIFO.....	
14.14.3	错误生成.....	452
14.14.3.1	UART 模式.....	
14.14.3.2	IO 接口模式.....	
14.15	软件复位	452
14.16	DMA 请求	452

14.17 在各模式下的操作	453
14.17.1 模式 0 (I/O 接口模式).....	453
14.17.1.1 发送数据	
14.17.1.2 接收	
14.17.1.3 发送和接收(全双工)	
14.17.2 模式 1 (7-位 UART 模式).....	464
14.17.3 模式 2 (8-位 UART 模式).....	464
14.17.4 模式 3 (9-位 UART 模式).....	465
14.17.4.1 唤醒功能	
14.17.4.2 协议	

15. 串行总线接口 (I2C/SIO)

15.1 配置	468
15.2 寄存器	469
15.2.1 各通道的寄存器.....	469
15.3 I2C 总线模式数据格式	470
15.4 I2C 总线模式下的控制寄存器	471
15.4.1 SBIxCR0 (控制寄存器 0).....	471
15.4.2 SBIxCR1 (控制寄存器 1).....	472
15.4.3 SBIxCR2(控制寄存器 2).....	474
15.4.4 SBIxSR (状态寄存器).....	475
15.4.5 SBIxBR0 (串行总线接口波特率寄存器 0).....	476
15.4.7 SBIxI2CAR (I2C 总线地址寄存器).....	477
15.5 I2C 总线模式下的控制器	478
15.5.1 串行时钟.....	478
15.5.1.1 时钟源	
15.5.1.2 时钟同步	
15.5.2 确认模式的设置.....	479
15.5.3 每次传输的位数的设置.....	479
15.5.4 从机寻址和地址识别模式.....	479
15.5.6 将 SBI 配置为发送器或接收器.....	480
15.5.7 将 SBI 配置为主机或从机.....	480
15.5.8 启动和停止条件的生成.....	480
15.5.9 中断服务请求和解除.....	481
15.5.10 判优丢失检测监控程序.....	482
15.5.11 从机地址匹配检测监控程序.....	483
15.5.12 一般调用检测监控程序.....	483
15.5.13 最后接收位监控程序.....	483
15.5.14 数据缓冲寄存器 (SBIxDBR).....	483
15.5.15 波特率寄存器 (SBIxBR0).....	484
15.5.16 软件复位.....	484
15.6 I2C 总线模式 2C 下的数据传输程序	485
15.6.1 设备初始化.....	485
15.6.2 启动条件和从机地址的生成.....	485
15.6.2.1 主模式	
15.6.2.2 从属模式	
15.6.3 数据字的传输.....	487
15.6.3.1 主模式 (<MST> = "1")	
15.6.3.2 从属模式 (<MST> = "0")	
15.6.4 停止条件的生成.....	492
15.6.5 重新启动程序.....	492
15.6.6 利用 DMA 进行数据传输.....	494
15.6.6.1 如何在主模式下传输数据	
15.6.6.2 如何在从属模式下传输数据	
15.7 SIO 模式控制寄存器	504
15.7.1 SBIxCR0 (控制寄存器 0).....	504
15.7.2 SBIxCR1 (控制寄存器 1).....	505
15.7.3 SBIxDBR (数据缓冲寄存器).....	506
15.7.4 SBIxCR2 (控制寄存器 2).....	507
15.7.5 SBIxSR (状态寄存器).....	508
15.7.6 SBIxBR0 (波特率寄存器 0).....	509

15.8 SIO 模式控制	510
15.8.1 串行时钟.....	510
15.8.1.1 时钟源.....	
15.8.1.2 移位缘.....	
15.8.2 传输模式.....	512
15.8.2.1 8-位传输模式.....	
15.8.2.2 8-位接收模式.....	
15.8.2.3 8-传输/接收模式.....	
15.8.2.4 传输结束时最后位的数据保留时间.....	

16. 模拟/数字转换器 (ADC)

16.1 概述	517
16.2 配置	518
16.3 寄存器	519
16.3.1 寄存器列表.....	519
16.3.2 ADCLK (转换时钟设置寄存器).....	520
16.3.3 ADMOD0 (模式控制寄存器 0).....	522
16.3.4 ADMOD1 (模式控制寄存器 1).....	523
16.3.5 ADMOD2 (模式控制寄存器 2).....	524
16.3.6 ADMOD3 (模式控制寄存器 3).....	526
16.3.7 ADMOD4 (模式控制寄存器 4).....	527
16.3.8 ADMOD5 (模式控制寄存器 5).....	528
16.3.9 ADMOD6 (模式控制寄存器 6).....	529
16.3.10 ADMOD7 (模式控制寄存器 7).....	530
16.3.11 ADCMPCR0 (监控控制寄存器 0).....	531
16.3.12 ADCMPCR1 (AD 监控控制寄存器 1).....	533
16.3.13 ADCMP0 (AD 转换结果比较寄存器 0).....	534
16.3.14 ADCMP1 (AD 转换结果比较寄存器 1).....	535
16.3.15 ADREG00~ADREG11 (普通转换结果寄存器 00~11).....	536
16.3.16 ADREGSP (最高优先级转换结果寄存器).....	537
16.4 操作说明	538
16.4.1 模拟参考电压.....	538
16.4.2 AD 转换模式.....	538
16.4.2.1 普通 AD 转换.....	
16.4.2.2 最高优先级 AD 转换.....	
16.4.3 AD 监控功能.....	539
16.4.4 选择输入通道.....	541
16.4.5 AD 转换详情.....	542
16.4.5.1 启动 AD 转换.....	
16.4.5.2 AD 转换.....	
16.4.5.3 普通 AD 转换期间的最高优先级 AD 转换请求.....	
16.4.5.4 停止重复转换模式.....	
16.4.5.5 重新激活 AD 转换.....	
16.4.5.6 转换完成.....	
16.4.5.7 中断生成时机和 AD 转换结果存储寄存器.....	

17. 闪存操作

17.1 特征	550
17.1.1 存储器大小和配置.....	550
17.1.2 功能.....	552
17.1.3 运行模式.....	552
17.1.3.1 模式说明.....	
17.1.3.2 模式判断.....	
17.1.4 存储器地址.....	554
17.1.5 保护/安全功能.....	555
17.1.5.1 保护功能.....	
17.1.5.2 安全功能.....	
17.1.6 寄存器.....	556
17.1.6.1 寄存器列表.....	
17.1.6.2 FCFLCS (闪存控制寄存器).....	
17.1.6.3 FCSECBIT (安全位寄存器).....	

17.2 闪存详细说明	558
17.2.1 功能	558
17.2.2 闪存运行模式	558
17.2.3 硬件复位	558
17.2.4 如何执行命令	559
17.2.5 命令说明	559
17.2.5.1 自动页面程序	
17.2.5.2 自动芯片擦除	
17.2.5.3 自动块擦除	
17.2.5.4 自动保护位程序	
17.2.5.5 自动保护位擦除	
17.2.5.6 ID 读取	
17.2.5.7 读取命令和读取/复位命令(软件复位)	
17.2.6 命令序列	562
17.2.6.1 命令序列表	
17.2.6.2 总线周期中地址位配置	
17.2.6.3 块地址(BA)	
17.2.6.4 如何指定保护位 (PBA)	
17.2.6.5 ID 读取代码 (IA, ID)	
17.2.6.6 命令序列示例	
17.2.7 流程图	568
17.2.7.1 自动程序	
17.2.7.2 自动擦除	
17.3 如何使用单启动模式对闪存进行重新编程	570
17.3.1 模式设置	570
17.3.2 接口规范	570
17.3.3 内存存储器限制	571
17.3.4 运行命令	571
17.3.4.1 RAM 传输	
17.3.4.2 闪存芯片擦除和保护位擦除	
17.3.5 和命令无关的常规操作	572
17.3.5.1 连续操作模式确定	
17.3.5.2 确认响应数据	
17.3.5.3 密码确定	
17.3.5.4 CHECK SUM 计算	
17.3.6 RAM 传输的转换格式	578
17.3.7 闪存芯片擦除 和保护位擦除的转换格式	579
17.3.8 启动程序完整流程图	581
17.3.9 片上 BOOT ROM 中使用重编程算法重编闪存程序	582
17.3.9.1 步骤-1	
17.3.9.2 步骤-2	
17.3.9.3 步骤-3	
17.3.9.4 步骤-4	
17.3.9.5 步骤-5	
17.3.9.6 步骤-6	
17.4 用户启动模式下编程	585
17.4.1 (1-A) 编程程序存储于闪存的步骤	585
17.4.1.1 步骤-1	
17.4.1.2 步骤-2	
17.4.1.3 步骤-3	
17.4.1.4 步骤-4	
17.4.1.5 步骤-5	
17.4.1.6 步骤-6	
17.4.2 (1-B)编程程序自外部主机进行传输的步骤	589
17.4.2.1 步骤-1	
17.4.2.2 步骤-2	
17.4.2.3 步骤-3	
17.4.2.4 步骤-4	
17.4.2.5 步骤-5	
17.4.2.6 步骤-6	

18. 端口部分等效电路示意图

18.1 PA0 ~ 7, PB0 ~ 7	594
18.2 PC0 ~ 2, PD0 ~ 7, PE0 ~ 7, PF1 ~ 7, PG0 ~ 4, PH0 ~ 4, PI0 ~ 7	594
18.3 PG5	595
18.4 PJ0 ~ 7, PK0 ~ 3	595

18.5	PF0	596
18.6	X1, X2	596
18.7	RESET, NMI	596
18.8	MODE	597
18.9	FTEST3	597
18.10	AVREFH, AVREFL	597

19. 电气特性

19.1	绝对最大额定值	598
19.2	DC 电气特性 (1/3)	599
19.3	DC 电气特性 (2/3)	600
19.4	DC 电气特性 (3/3)	601
19.5	12-位 ADC 电气特性	602
19.6	AC 电气特性	603
19.6.1	AC 测量条件	603
19.6.2	串行通道 (SIO/UART)	603
19.6.2.1	I/O 接口模式	
19.6.3	串行总线接口(I2C/SIO)	605
19.6.3.1	I2C 模式	
19.6.3.2	时钟同步 8-位 SIO 模式	
19.6.4	事件计数器	608
19.6.5	捕捉	608
19.6.6	外部中断	608
19.6.7	NMI	609
19.6.8	SCOUT 引脚 AC 特性	609
19.6.9	ADTRG 触发器输入引脚 AC 特性	609
19.6.10	USB 定时	609
19.6.11	调试通信	610
19.6.11.1	SWD 接口	
19.6.11.2	JTAG 接口	
19.6.12	ETM 追踪	611
19.6.13	片上振荡器	611
19.6.14	外部时钟输入	611
19.6.15	闪存特性	612
19.7	推荐振荡电路	613
19.7.1	陶瓷谐振器	613
19.7.2	晶体振荡器	613
19.7.2.1	印刷电路板设计注意事项	
19.8	使用注意事项	614
19.8.1	通电电源电压电平	614

20. 封装尺寸

TMPM365FYXBG

TMPM365FYXBG是一种带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
TMPM365FYXBG	256 K字节	24 KB	LFBGA105

TMPM365FYXBG的特性如下：

1.1 特征

1. ARM Cortex-M3 微处理器内核
 - a. 通过使用Thumb® -2 指令，已实现了编码效率的提高
 - 新 16-位Thumb指令改进了程序流程
 - 新 32-位Thumb指令改进了性能
 - 新Thumb混合的 16 / 32 位指令集能产生更快，更高效的编码
 - b. 已实现高性能，低功耗
 - 【高性能】
 - 用一个时钟就能执行 32-位乘法(32 x 32 = 32 位)
 - 根据被除数和除数，除法执行 2 ~ 12 次
 - 【低功耗】
 - 用低功耗库优化了设计
 - 具有能使微控制器内核停止工作的待机功能
 - c. 高速中断响应，适合实时控制
 - 可中断的长指令
 - 硬件自动处理压栈
2. 片上程序存储器和数据存储器
 - 片上闪存ROM： 256 K字节
 - 片上SRAM： 24 K字节
3. DMA控制器(DMAC： 2 个通道
 - 存储器 / 外设功能 / 外部存储器
4. 16 位定时器(TMRB)： 10 个通道
 - 16-位间隔计时器模式
 - 16-位计数器模式
 - 16-位PPG输出
 - 输入捕获功能

5. 看门狗定时器(WDT): 1 个通道
看门狗定时器生成复位或不可屏蔽中断(NMI)。
6. 串行通道(SIO/UART): 2 个通道
可选择UART模式或同步通信模式(4 字节FIFO)
7. 串行总线接口(I2C/SIO): 2 个通道
可选择I2C总线模式或时钟同步 8-位SIO模式。
8. USB装置控制器(USB) : 1 个通道
符合 2.0 版《通用串行总线规范》。
支持全通信速度(12 Mbps)(不支持低速)。
支持 8 个端点。
端点 0: 控制 64 字节 × 1-FIFO
端点 1: 扩至(装置 → 主机: IN传送) 64 字节 × 2-FIFO
端点 2: 扩至(主机 → 装置: OUT传送) 64 字节 × 2-FIFO
端点 3: 扩至(装置 → 主机: IN传送) 64 字节 × 2-FIFO
端点 4: 扩至(主机 → 装置: OUT传送) 64 字节 × 2-FIFO
端点 5: 扩至(装置 → 主机: IN传送) 64 字节 × 2-FIFO
端点 6: 扩至(主机 → 装置: OUT传送) 64 字节 × 2-FIFO
端点 7: 中断(装置 → 主机: IN传送) 64 字节 × 2-FIFO
端点 1 ~ 端点 7 能支持 4 种模式。
9. 12-位AD转换器(ADC): 12 个通道
用 16-位定时器进行启动/用外部触发器输入进行启动
固定通道/扫描通道模式
AD监测 2 个通道
转换时间 1 μs。(@fsys = 40 MHz), 1.67 μs。(@fsys = 48 MHz)
10. 中断源
内部: 47 个因素。优先顺序设置为 7 级
(除看门狗定时器中断外)。
外部: 10 个因素。优先顺序设置为 7 级
11. 不可屏蔽中断(NMI)
不可屏蔽中断(NMI)由看门狗定时器或 $\overline{\text{NMI}}$ 引脚生成。
12. 输入/输出端口(PORT): 74 个引脚(包括一个可承受 5 V 的输入)
13. 低功耗模式
低功耗模式IDLE, STOP1
14. 时钟发生器(CG)
片上PLL (x8, 系统时钟除以 2)

时钟齿轮功能：高速时钟分为 1/1, 1/2, 1/4, 1/8 或 1/16

15. 字节存储次序
从小到大
16. 调试接口
JTAG / SWD / SWV / TRACE (DATA 4 位)
17. JTAG 接口
边界扫描
18. 最大工作频率：48 MHz
19. 工作电压范围
2.7 V ~ 3.6 V (配备片上稳压器)
3.0 V ~ 3.45 V (当使用 USB 时)
20. 温度范围
-40 ~ 85 度 (除闪存写入/擦除时外)
0 ~ 70 度 (除闪存写入/擦除时外)
21. 封装
LFBGA105 (9 mm × 9 mm, 0.5 mm 节距)

1.2 方块图

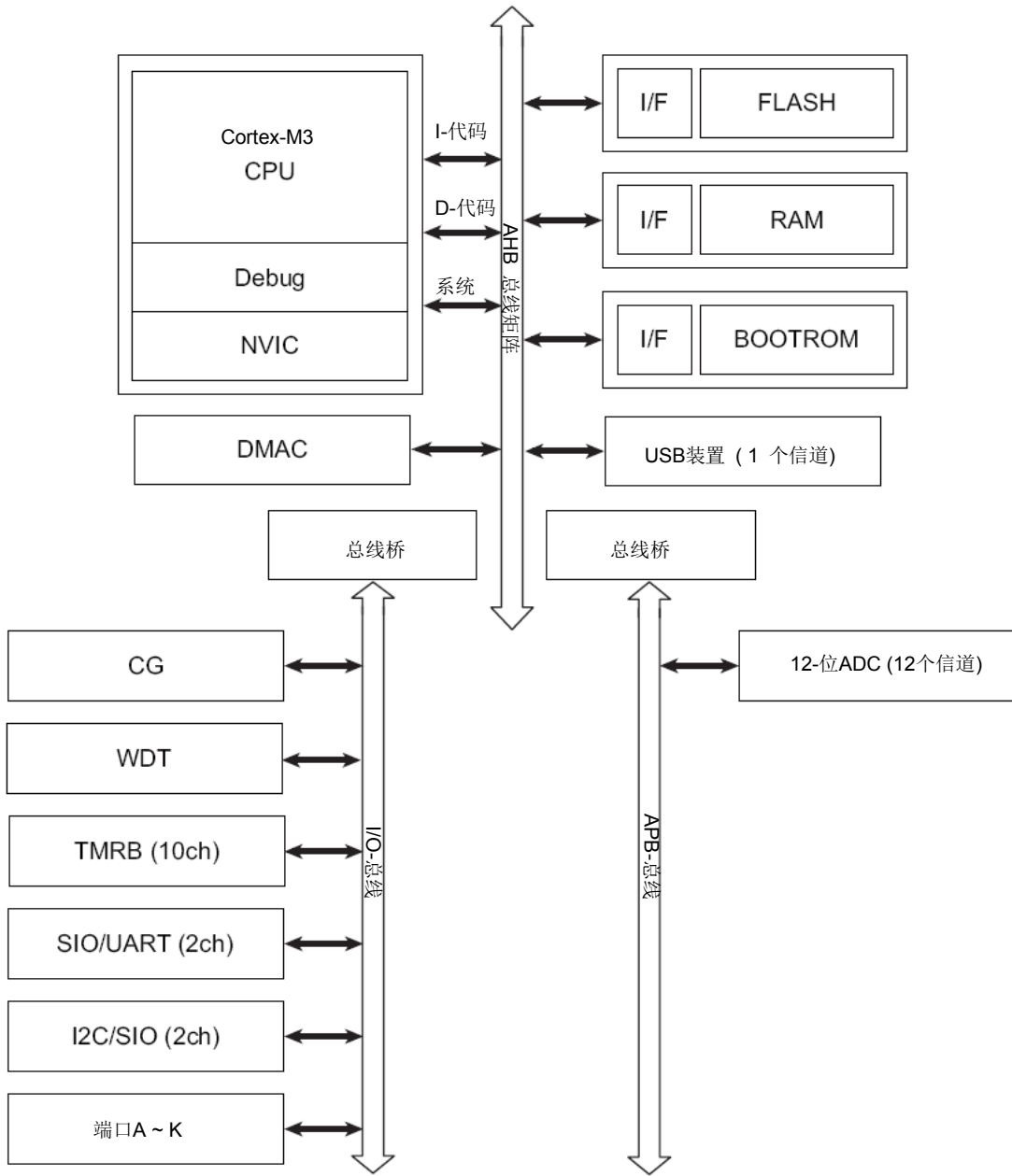


图 1-1 TMPM365FYXBG方块图

1.3 引脚分配 (顶视图)

TMPM365FYXBG引脚分配如图 1-2 所示。

A1	A2	A3	A4	A5	A6	A7	A8	A9	A10	A11	A12	A13	A14	A15
B1	B2	B3	B4	B5	B6	B7	B8	B9	B10	B11	B12	B13	B14	B15
C1	C2	-	-	-	-	-	-	-	-	-	-	-	C14	C15
D1	D2	-	-	-	-	-	-	-	-	-	-	-	D14	D15
E1	E2	-	-	-	-	-	-	-	-	-	-	-	E14	E15
F1	F2	-	-	-	-	-	-	-	-	-	-	-	F14	F15
G1	G2	-	-	-	-	-	-	-	-	-	-	-	G14	G15
H1	H2	-	-	-	-	-	-	-	-	-	-	-	H14	H15
J1	J2	-	-	-	-	-	-	-	-	-	-	-	J14	J15
K1	K2	-	-	-	-	-	-	-	-	-	-	-	K14	K15
L1	L2	-	-	-	-	-	-	-	-	-	-	-	L14	L15
M1	M2	-	-	-	-	-	-	-	-	-	-	-	M4	M15
N1	N2	N3	-	-	-	-	-	-	-	-	-	-	N14	N15
P1	P2	P3	P4	P5	P6	P7	P8	P9	P10	P11	P12	P13	P14	P15
R1	R2	R3	R4	R5	R6	R7	R8	R9	R10	R11	R12	R13	R14	R15

图 1-2 引脚分配 (BGA105)

1.4 引脚名称与功能

按引脚或端口分类的TMPM365FYXBG输入和输出引脚如表 1-1 和表 1-2 所示。各表包括多功能引脚的替代引脚名称和功能。

1.4.1 按引脚分类

表 1-1 按引脚分类的引脚名称和功能(1 / 6)

类型	引脚编号	引脚名称	输入/输出	功能
功能	A1	PK0 AIN08 INT2 TB1IN0	输入 输入 输入 输入	I/O端口 模拟输入 外部中断引脚 输入TMRB捕获触发器
功能	A2	PJ7 AIN07 INT9 TB0IN1	I/O 输入 输入 输入	I/O端口 模拟输入 外部中断引脚 输入TMRB捕获触发器
功能	A3	PJ5 AIN05	I/O 输入	I/O端口 模拟输入
功能	A4	PJ3 AIN03	I/O 输入	I/O端口 模拟输入
功能	A5	PJ1 AIN01	I/O 输入	I/O端口 模拟输入
PS	A6	RVDD3	-	内部稳压器电源引脚
PS	A7	RVSS	-	内部稳压器GND引脚
功能	A8	REGOUT	输出	稳压器输出引脚
功能	A9	REGIN	输入	稳压器输入引脚
功能	A10	PE2 SCLK0 TB2OUT CTS0	I/O I/O 输出 输入	I/O端口 串行时钟输入/输出 TMRB输出 握手输入引脚
PS	A11	DVDD3A	-	电源引脚
时钟	A12	X1	输入	连接高速振荡器 /外部时钟输入引脚
PS	A13	DVSSC	-	振荡器GND引脚
时钟	A14	X2	输出	已连接至高速振荡器
功能	A15	PE0 TXD0	I/O 输出	I/O端口 发送串行数据
功能	B1	PK1 AIN09 INT3 TB1IN1	I/O 输入 输入 输入	I/O端口 模拟输入 外部中断引脚 输入TMRB捕获触发器
功能	B2	PJ6 AIN06 TB0IN0	I/O 输入 输入	I/O端口 模拟输入 输入TMRB捕获触发器

表 1-1 按引脚分类的引脚名称和功能(2 / 6)

类型	引脚编号	引脚名称	输入/输出	功能
功能	B3	PJ4 AIN04	I/O 输入	I/O端口 模拟输入
功能	B4	PJ2 AIN02	I/O 输入	I/O端口 模拟输入
功能	B5	PJ0 AIN00	I/O 输入	I/O端口 模拟输入
PS	B6	RVDD3	-	内部稳压器电源引脚
功能	B7	PE7 INT4	I/O 输入	I/O端口 外部中断引脚
功能	B8	PE6 SCK1	I/O I/O	I/O端口 若串行总线接口在SIO模式下运行, 输入及输出时钟
功能	B9	PE5 SCL1/SI1	I/O I/O	I/O端口 在I2C模式时, 时钟引脚/在SIO模式时, 数据引脚
功能	B10	PE4 SDA1/SO1	I/O I/O	I/O端口 在I2C模式时, 数据引脚/在SIO模式时, 数据引脚
功能	B11	PE3 INT5 TB3OUT	I/O 输入 输出	I/O端口 外部中断引脚 TMRB输出
功能	B12	$\overline{\text{NMI}}$	输入	非屏蔽中断 (注) 配备噪声滤波器(约 30 ns (典型值))
PS	B13	DVSSA	-	GND引脚
控制	B14	MODE	输入	模式引脚 (注) 必须将MODE引脚连接至GND。
功能	B15	$\overline{\text{RESET}}$	输入	复位输入引脚 (注) 带有一个上拉与一个噪声滤波器(约30ns (典型值))
功能	C1	PK2 AIN10 TB6IN0	I/O 输入 输入	I/O端口 模拟输入 输入TMRB捕获触发器
功能	C2	PK3 AIN11 TB6IN1	I/O 输入 输入	I/O端口 模拟输入 输入TMRB捕获触发器
功能	C14	PE1 RXD0	I/O 输入	I/O端口 接收串行数据
功能	C15	PD0 TB7OUT	I/O 输出	I/O端口 TMRB输出
PS	D1	AVREFH	输入	用参考电源给AD转换器供电。 (注) 即使AD转换器未用, AVREFH仍必须接通电源。
PS	D2	AVDD3	输入	用电源给AD转换器供电。 (注) 即使AD转换器未用, AVDD3 仍必须接通电源。
功能	D14	PD2 TB9OUT	I/O 输出	I/O端口 TMRB输出

表 1-1 按引脚分类的引脚名称和功能(3 / 6)

类型	引脚编号	引脚名称	输入/输出	功能
功能	D15	PD1 TB8OUT	I/O 输出	I/O端口 TMRB输出
PS	E1	AVREFL	输入	用参考GND给AD转换器供电。 (注) 即使AD转换器未用, AVREFL仍必须接通GND。
PS	E2	AVSS	输入	AD转换器:GND引脚 (注) 即使AD转换器未用, AVSS仍必须接通GND。
功能	E14	PD4	I/O	I/O端口
功能	E15	PD3 ADTRG	I/O 输入	I/O端口 AD触发器输入
PS	F1	DVSSA	-	GND引脚
控制	F2	BSC	输入	边界扫描控制引脚
PS	F14	DVDD3C	-	USB电源引脚
PS	F15	DVDD3C	-	USB电源引脚
PS	G1	DVDD3A	-	电源引脚
功能/调试	G2	PI7 TRST	I/O 输入	I/O端口 调试引脚
PS	G14	DVSS3C	-	USB GND引脚
功能	G15	D+	I/O	USB引脚(D+)
功能/调试	H1	PI2 TRACECLK	I/O 输出	I/O端口 调试引脚
功能/调试	H2	PI6 TDI	I/O 输入	I/O端口 调试引脚
PS	H14	DVSS3C	-	USB GND引脚
功能	H15	D-	I/O	USB引脚(D-)
功能/调试	J1	PI1 TRACEDATA 0	I/O 输出	I/O端口 调试引脚
功能/调试	J2	PI5 TDO/SWV	I/O 输出	I/O端口 调试引脚
PS	J14	DVSSA	-	GND引脚
PS	J15	DVDD3A	-	电源引脚
功能/调试	K1	PI0 TRACEDATA 1	I/O 输出	I/O端口 调试引脚
功能/调试	K2	PI4 TMS/SWDIO	I/O I/O	I/O端口 调试引脚
功能	K14	PD6	I/O	I/O端口

表 1-1 按引脚分类的引脚名称和功能(4/6)

类型	引脚编号	引脚名称	输入/输出	功能
功能	K15	PD5	I/O	I/O端口
功能/调试	L1	PH0 TRACEDATA 2	I/O 输出	I/O端口 调试引脚
功能/调试	L2	PI3 TCK/SWCLK	I/O 输入	I/O端口 调试引脚
功能	L14	PD7 SCOUT	I/O 输出	I/O端口 系统时钟输出
功能	L15	PB7	I/O	I/O端口
功能/调试	M1	PH1 TRACEDATA 3	I/O 输出	I/O端口 调试引脚
功能	M2	PH2 TB4OUT	I/O 输出	I/O端口 TMRB输出
功能	M14	PB5	I/O	I/O端口
功能	M15	PB6	I/O	I/O端口
功能	N1	PH3 TB5OUT	I/O 输出	I/O端口 TMRB输出
功能	N2	PH4 INT8	I/O 输入	I/O端口 外部中断引脚
控制	N3	FTEST3	-	测试引脚 (注) 测试引脚必须左OPEN。
功能	N14	PB3	I/O	I/O端口
功能	N15	PB4	I/O	I/O端口
功能	P1	PG5 INT1 USBPON	I/O 输入 输入	I/O端口(可承受 5V的输入)(注) 外部中断引脚 USB连接检测引脚(VBUS检测)
功能	P2	PG4 TB4IN1	I/O 输入	I/O端口 输入TMRB捕获触发器
功能	P3	PG2 SCK0 TB3IN1	I/O I/O 输入	I/O端口 若串行总线接口在SIO模式下运行, 输入及输出时钟 输入TMRB捕获触发器
功能	P4	PG1 SCL0/SIO TB3IN0	I/O I/O 输入	I/O端口 在I2C模式时, 时钟引脚/在SIO模式时, 数据引脚 输入TMRB捕获触发器
功能	P5	PG0 SDA0/SO0	I/O I/O	I/O端口 在I2C模式时, 数据引脚/在SIO模式时, 数据引脚
功能	P6	PF2	I/O	I/O端口

表 1-1 按引脚分类的引脚名称和功能(5 / 6)

类型	引脚编号	引脚名称	输入/输出	功能
功能	P7	PF4 INT6 TB5IN0	I/O 输入 输入	I/O端口 外部中断引脚 输入TMRB捕获触发器
功能	P8	PF5 INT7 TB5IN1	I/O 输入 输入	I/O端口 外部中断引脚 输入TMRB捕获触发器
功能	P9	PF6	I/O	I/O端口
功能	P10	PF7	I/O	I/O端口
功能	P11	PA2	I/O	I/O端口
功能	P12	PA4	I/O	I/O端口
功能	P13	PA6	I/O	I/O端口
功能	P14	PB1	I/O	I/O端口
功能	P15	PB2	I/O	I/O端口
功能	R1	PG3 INT0 TB4IN0	I/O 输入 输入	I/O端口 外部中断引脚 输入TMRB捕获触发器
功能	R2	PC0 TXD1 TB2IN0	I/O 输出 输入	I/O端口 发送串行数据 输入TMRB捕获触发器
功能	R3	PC1 RXD1 TB2IN1	I/O 输入 输入	I/O端口 接收串行数据 输入TMRB捕获触发器
功能	R4	PC2 SCLK1 TB0OUT CTS1	I/O I/O 输出 输入	I/O端口 串行时钟输入/输出 TMRB输出 握手输入引脚
功能/控制	R5	PF0 BOOT TB6OUT	输出 输入 输出	输出端口 设置一个单启动模式: 该引脚可在RESET信号上升时进入单一引导模式“低”。 TMRB输出
功能	R6	PF1	I/O	I/O端口
功能	R7	PF3	I/O	I/O端口
PS	R8	DVSSA	-	GND引脚
PS	R9	DVDD3A	-	电源引脚
功能	R10	PA0	I/O	I/O端口
功能	R11	PA1	I/O	I/O端口

表 1-1 按引脚分类的引脚名称和功能(6 / 6)

类型	引脚 编号	引脚名称	输入/ 输出	功能
功能	R12	PA3	I/O	I/O端口
功能	R13	PA5	I/O	I/O端口
功能	R14	PA7	I/O	I/O端口
功能	R15	PB0	I/O	I/O端口

注：只有在输入启用时，这些引脚才能承受 5 V输入。注意当用作开漏输出时，这些引脚不能上拉到电源电压之上。

1.4.2 按端口分类

表 1-2 按端口分类的引脚名称和功能(1/6)

PORT	类型	引脚 编号	引脚名称	输入/ 输出	功能
PORT A	功能	R10	PA0	I/O	I/O端口
PORT A	功能	R11	PA1	I/O	I/O端口
PORT A	功能	P11	PA2	I/O	I/O端口
PORT A	功能	R12	PA3	I/O	I/O端口
PORT A	功能	P12	PA4	I/O	I/O端口
PORT A	功能	R13	PA5	I/O	I/O端口
PORT A	功能	P13	PA6	I/O	I/O端口
PORT A	功能	R14	PA7	I/O	I/O端口
PORT B	功能	R15	PB0	I/O	I/O端口
PORT B	功能	P14	PB1	I/O	I/O端口
PORT B	功能	P15	PB2	I/O	I/O端口
PORT B	功能	N14	PB3	I/O	I/O端口
PORT B	功能	N15	PB4	I/O	I/O端口
PORT B	功能	M14	PB5	I/O	I/O端口
PORT B	功能	M15	PB6	I/O	I/O端口
PORT B	功能	L15	PB7	I/O	I/O端口
PORT C	功能	R2	PC0 TXD1 TB2IN0	I/O 输出 输入	I/O端口 发送串行数据 输入TMRB捕获触发器
PORT C	功能	R3	PC1 RXD1 TB2IN1	I/O 输入 输入	I/O端口 接收串行数据 输入TMRB捕获触发器
PORT C	功能	R4	PC2 SCLK1 TB0OUT CTS1	I/O I/O 输出 输入	I/O端口 串行时钟输入/输出 TMRB输出 握手输入引脚
PORT D	功能	C15	PD0 TB7OUT	I/O 输出	I/O端口 TMRB输出
PORT D	功能	D15	PD1 TB8OUT	I/O 输出	I/O端口 TMRB输出
PORT D	功能	D14	PD2 TB9OUT	I/O 输出	I/O端口 TMRB输出

表 1-2 按端口分类的引脚名称和功能(2/6)

PORT	类型	引脚编号	引脚名称	输入/输出	功能
PORT D	功能	E15	PD3 ADTRG	I/O 输入	I/O端口 AD触发器输入
PORT D	功能	E14	PD4	I/O	I/O端口
PORT D	功能	K15	PD5	I/O	I/O端口
PORT D	功能	K14	PD6	I/O	I/O端口
PORT D	功能	L14	PD7 SCOUT	I/O 输出	I/O端口 系统时钟输出
PORT E	功能	A15	PE0 TXD0	I/O 输出	I/O端口 发送串行数据
PORT E	功能	C14	PE1 RXD0	I/O 输入	GND引脚 接收串行数据
PORT E	功能	A10	PE2 SCLK0 TB2OUT CTS0	I/O I/O 输出 输入	I/O端口 串行时钟输入/输出 TMRB输出 握手输入引脚
PORT E	功能	B11	PE3 INT5 TB3OUT	I/O 输入 输出	I/O端口 外部中断引脚 TMRB输出
PORT E	功能	B10	PE4 SDA1/SO1	I/O I/O	I/O端口 在I2C模式时, 数据引脚/在SIO模式时, 数据引脚
PORT E	功能	B9	PE5 SCL1/SI1	I/O I/O	I/O端口 在I2C模式时, 时钟引脚/在SIO模式时, 数据引脚
PORT E	功能	B8	PE6 SCK1	I/O I/O	I/O端口 若串行总线接口在SIO模式下运行, 输入及输出时钟
PORT E	功能	B7	PE7 INT4	I/O 输入	I/O端口 外部中断引脚
PORT F	功能/ 控制	R5	PF0 BOOT TB6OUT	输出 输入 输出	输出端口 设置一个单启动模式: 该引脚可在 RESET 信号上升时进入单一引导模式“低”。TMRB输出
PORT F	功能	R6	PF1	I/O	I/O端口
PORT F	功能	P6	PF2	I/O	I/O端口
PORT F	功能	R7	PF3	I/O	I/O端口
PORT F	功能	P7	PF4 INT6 TB5IN0	I/O 输入 输入	I/O端口 外部中断引脚 输入TMRB捕获触发器
PORT F	功能	P8	PF5 INT7 TB5IN1	I/O 输入 输入	I/O端口 外部中断引脚 输入TMRB捕获触发器

表 1-2 按端口分类的引脚名称和功能(3/6)

PORT	类型	引脚编号	引脚名称	输入/输出	功能
PORT F	功能	P9	PF6	I/O	I/O端口
PORT F	功能	P10	PF7	I/O	I/O端口
PORT G	功能	P5	PG0 SDA0/SO0	I/O I/O	I/O端口 在I2C模式时, 数据引脚/在SIO模式时, 数据引脚
PORT G	功能	P4	PG1 SCL0/SIO TB3IN0	I/O I/O 输入	I/O端口 在I2C模式时, 时钟引脚/在SIO模式时, 数据引脚 输入TMRB捕获触发器
PORT G	功能	P3	PG2 SCK0 TB3IN1	I/O I/O 输入	I/O端口 若串行总线接口在SIO模式下运行, 输入及输出时钟 输入TMRB捕获触发器
PORT G	功能	R1	PG3 INT0 TB4IN0	I/O 输入 输入	I/O端口 外部中断引脚 输入TMRB捕获触发器
PORT G	功能	P2	PG4 TB4IN1	I/O 输入	I/O端口 输入TMRB捕获触发器
PORT G	功能	P1	PG5 INT1 USBPON	I/O 输入 输入	I/O端口 (1 个可承受 5 V的输入)(注) 外部中断引脚 USB连接检测引脚(VBUS检测)
PORT H	功能/调试	L1	PH0 TRACEDA- TA2	I/O 输出	I/O端口 调试引脚
PORT H	功能/调试	M1	PH1 TRACEDA- TA3	I/O 输出	I/O端口 调试引脚
PORT H	功能	M2	PH2 TB4OUT	I/O 输出	I/O端口 TMRB输出
PORT H	功能	N1	PH3 TB5OUT	I/O 输出	I/O端口 TMRB输出
PORT H	功能	N2	PH4 INT8	I/O 输入	I/O端口 外部中断引脚
PORT I	功能/调试	K1	PI0 TRACEDA- TA1	I/O 输出	I/O端口 调试引脚
PORT I	功能/调试	J1	PI1 TRACEDA- TA0	I/O 输出	I/O端口 调试引脚
PORT I	功能/调试	H1	PI2 TRACECLK	I/O 输出	I/O端口 调试引脚
PORT I	功能/调试	L2	PI3 TCK/SWCLK	I/O 输入	I/O端口 调试引脚
PORT I	功能/调试	K2	PI4 TMS/SWDIO	I/O I/O	I/O端口 调试引脚
PORT I	功能/调试	J2	PI5 TDO/SWV	I/O 输出	I/O端口 调试引脚

表 1-2 按端口分类的引脚名称和功能(4/6)

PORT	类型	引脚编号	引脚名称	输入/输出	功能
PORT I	功能/调试	H2	PI6 TDI	I/O 输入	I/O端口 调试引脚
PORT I	功能/调试	G2	PI7 $\overline{\text{TRST}}$	I/O 输入	I/O端口 调试引脚
PORT J	功能	B5	PJ0 AIN00	I/O 输入	I/O端口 模拟输入
PORT J	功能	A5	PJ1 AIN01	I/O I	I/O端口 模拟输入
PORT J	功能	B4	PJ2 AIN02	I/O 输入	I/O端口 模拟输入
PORT J	功能	A4	PJ3 AIN03	I/O 输入	I/O端口 模拟输入
PORT J	功能	B3	PJ4 AIN04	I/O 输入	I/O端口 模拟输入
PORT J	功能	A3	PJ5 AIN05	I/O 输入	I/O端口 模拟输入
PORT J	功能	B2	PJ6 AIN06 TBOIN0	I/O 输入 输入	I/O端口 模拟输入 输入TMRB捕获触发器
PORT J	功能	A2	PJ7 AIN07 INT9 TBOIN1	I/O 输入 输入 输入	I/O端口 模拟输入 外部中断引脚 输入TMRB捕获触发器
PORT K	功能	A1	PK0 AIN08 INT2 TB1IN0	I/O 输入 输入 输入	I/O端口 模拟输入 外部中断引脚 输入TMRB捕获触发器
PORT K	功能	B1	PK1 AIN09 INT3 TB1IN1	I/O 输入 输入 输入	I/O端口 模拟输入 外部中断引脚 输入TMRB捕获触发器
PORT K	功能	C1	PK2 AIN10 TB6IN0	I/O 输入 输入	I/O端口 模拟输入 输入TMRB捕获触发器
PORT K	功能	C2	PK3 AIN11 TB6IN1	I/O 输入 输入	I/O端口 模拟输入 输入TMRB捕获触发器
-	功能	G15	D+	I/O	USB引脚(D+)
-	功能	H15	D-	I/O	USB引脚(D-)
-	功能	B15	$\overline{\text{RESET}}$	输入	复位输入引脚 (注) 带有一个上拉与一个噪声滤波器(约30ns (典型值))

表 1-2 按端口分类的引脚名称和功能(5/6)

PORT	类型	引脚编号	引脚名称	输入/输出	功能
-	功能	B12	$\overline{\text{NMI}}$	输入	非屏蔽中断 (注) 配备噪声滤波器(约 30 ns (典型值))
-	控制	B14	MODE	输入	模式引脚 (注) 必须将MODE引脚连接至GND。
-	控制	N3	FTEST3	-	测试引脚 (注) 测试引脚必须左OPEN。
-	控制	F2	BSC	输入	边界扫描控制引脚
-	时钟	A12	X1	输入	连接高速振荡器 / 外部时钟输入引脚
-	时钟	A14	X2	输出	已连接至高速振荡器
-	PS	G1	DVDD3A	-	电源引脚
-	PS	R9	DVDD3A	-	电源引脚
-	PS	J15	DVDD3A	-	电源引脚
-	PS	A11	DVDD3A	-	电源引脚
-	PS	F1	DVSSA	-	GND引脚
-	PS	R8	DVSSA	-	GND引脚
-	PS	J14	DVSSA	-	GND引脚
-	PS	B13	DVSSA	-	GND引脚
-	PS	A6	RVDD3	-	内部稳压器电源引脚
-	PS	B6	RVDD3	-	内部稳压器电源引脚
-	PS	A7	RVSS	-	内部稳压器GND引脚
-	PS	A13	DVSSC	-	振荡器GND引脚
-	PS	F14	DVDD3C	-	USB电源引脚
-	PS	F15	DVDD3C	-	USB电源引脚
-	PS	G14	DVSS3C	-	USB GND引脚
-	PS	H14	DVSS3C	-	USB GND引脚
-	PS	D1	AVREFH	输入	用参考电源给AD转换器供电。 (注) 即使AD转换器未用, AVREFH仍必须接通电源。
-	PS	E1	AVREFL	输入	用参考GND给AD转换器供电。 (注) 即使AD转换器未用, AVREFL仍必须接通GND。

表 1-2 按端口分类的引脚名称和功能(6/6)

PORT	类型	引脚编号	引脚名称	输入/输出	功能
-	PS	D2	AVDD3	输入	用电源给AD转换器供电。 (注) 即使AD转换器未用, AVDD3 仍必须接通电源。
-	PS	E2	AVSS	输入	AD转换器: GND引脚 (注) 即使AD转换器未用, AVSS仍必须接通GND。
-	PS	A8	REGOUT	输出	稳压器输出引脚
-	PS	A9	REGIN	输入	稳压器输入引脚

注：只有在输入启用时，这些引脚才能承受 5 V 输入。注意当用作开漏输出时，这些引脚不能上拉到电源电压之上。

1.5 引脚编号与电源引脚

表 1-3 引脚编号和电源

电源	电压范围	引脚编号	引脚名称
DVDD3A	2.7 ~ 3.6 V (当使用USB时: 3.0 ~ 3.45 V)	G1, R9, J15, A11	PA, PB, PC, PD, PE, PF, PG, PH, PI, X1, X2, FTEST3, $\overline{\text{RESET}}$, $\overline{\text{NMI}}$, MODE, BSC
AVDD3		D2	PJ, PK
RVDD3		A6, B6	-
DVDD3C		F14, F15, G14, H14	D+, D-

1.6 内部稳压器引脚

REGOUT和REGIN引脚与使内部稳压器稳定的电容器连接。

REGOUT和REGIN必须经电容器连接RVSS而给内部稳压器供电。

请以最短的距离连接电容器。

不是从这些端子给外部电路供电。

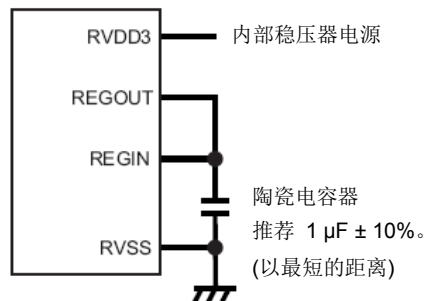


图 1-3 内部稳压器引脚

2. 处理器内核

该TX03系列配备有一个高性能的 32-位处理器内核(ARM Cortex-M3 处理器内核)。本处理器内核的操作信息见ARM Limited 发布的“Cortex-M3 技术参考手册”本节主要对该文件中未提及的TX03系列所独有的功能进行说明。

2.1 有关该处理器内核的信息

TMPM365FYXBG 处理器内核版本如下表所示。

请参看该CPU内核与CPU体系结构的详细信息，并请参看以下URL中的ARM手册“Cortex-M系列处理器”：

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

产品名称	内核版次
TMPM365FYXBG	r2p0

2.2 可配置的选项

Cortex-M3内核具备可选块。r2p0版的可选块为ETM™，MPU和WIC。TMPM365FYXBG可配置选项如下表所示。

可配置的选项	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跟踪 宏单元接口	无
TPIU	有
WIC	无
调试端口	JTAG/串行线

2.3 异常/中断

以下将对异常与中断进行说明。

2.3.1 中断输入的数目

可在Cortex-M3内核中，将中断输入的数目选择性地定义为 1 ~ 240 范围内的某个数值。

TMPM365FYXBG 有 57 个中断输入。中断输入的数目可在NVIC寄存器的 <INTLINESNUM[4:0]> 位中反映出来。在本产品中，若读取<INTLINESNUM[4:0]> 位，就会读出 0x01。

2.3.2 优先级中断位的数目

该Cortex-M3内核可选择性地配置优先级中断位的数目，其配置范围为 3 位 ~ 8 位。

TMPM365FYXBG有 3 个优先级中断位。优先级中断位的数目，用于指定中断优先权寄存器与系统处理器优先寄存器中的优先级。

2.3.3 SysTick

该Cortex-M3内核具备一个可生成SysTick异常的SysTick计时器。

SysTick异常，详见异常中的"SysTick"一节及NVIC寄存器中的SysTick寄存器。

2.3.4 SYSRESETREQ

当设置应用中断和复位控制寄存器的<SYSRESETREQ>位时，Cortex-M3内核会输出SYSRESETREQ信号。

当输出SYSRESETREQ信号时，TMPM365FYXBG提供相同的操作。

2.3.5 LOCKUP

当产生不能补救的异常时，Cortex-M3内核会输出LOCKUP信号，显示软件中包括的严重错误。

TMPM365FYXBG不使用该信号。为了从LOCKUP状态中返回，必须使用不可屏蔽中断(NMI)或复位。

2.3.6 辅助故障状态寄存器

该Cortex-M3内核配备辅助故障状态寄存器，用以向软件提供附加系统故障信息。

然而，TMPM365FYXBG未具有该功能。如果读取了辅助故障状态寄存器，则所读出的始终会是"0x0000_0000"。

2.4 事件

该Cortex-M3内核具有事件输出信号与事件输入信号。事件输出信号由SEV指令执行输出。如果输入了某事件，则该内核会从WFE指令所导致的低功耗模式返回。

TMPM365FYXBG不使用事件输出信号和事件输入信号。请不要使用SEV指令与WFE指令。

2.5 电源管理

该Cortex-M3内核配备有使用SLEEPING信号与SLEEPDEEP信号的功率调节系统。在设置系统控制寄存器的<SLEEPDEEP>位时，可输出SLEEPDEEP信号。

在以下情况下会输出这些信号：

- 等待中断(WFI)指令的执行
- 等待事件(WFE)指令的执行
- 若设置了系统控制寄存器的<SLEEPONEXIT>位，当中断服务程序(ISR)退出时。

TMPM365FYXBG不使用SLEEPDEEP信号，以确保不设置<SLEEPDEEP>位。其也不使用事件信号，因此请不要使用WFE指令。

有关电源管理的详细资料，请参看"时钟/模式控制"一节。

2.6 独占通道

在Cortex-M3内核中了，该DCode总线系统支持独占通道。然而，TMPM365FYXBG不使用该功能。



3. 调试接口

3.1 规范概述

TMPM365FYXBG包含与调试工具接口的串行线JTAG调试端口(SWJ-DP)单元和用于指令跟踪输出的嵌入式跟踪宏单元™(ETM)单元。跟踪数据经片上跟踪端口接口单元(TPIU)被输出到调试专用引脚(TRACE DATA[3:0], SWV)。

SWJ-DP, ETM和TPIU详见“Cortex-M3 技术参考手册”。

3.2 SWJ-DP

SWJ-DP支持串行线调试端口 (SWCLK, SWDIO) 和 JTAG 调试端口 (TDI, TDO, TMS, TCK, $\overline{\text{TRST}}$)。

3.3 ETM

ETM支持四个数据信号引脚(TRACE DATA[3:0])，一个时钟信号引脚(TRACECLK)及串行线观测器(SWV)的跟踪输出。

3.4 引脚功能

调试接口引脚也能用作通用端口。

在JTAG调试端口功能和串行线调试端口功能之间，PI3 和PI4 引脚被共用。在JTAG调试端口功能和SWV跟踪输入功能之间，PI5 引脚被共用。

表 3-1 SWJ-DP, ETM调试功能

SWJ-DP 引脚名称	通用 端口名称	JTAG调试功能		SW调试功能	
		I/O	说明	I/O	说明
TMS / SWDIO	PI4	输入	JTAG测试模式选择	I/O	串行线数据输入/输出
TCK / SWCLK	PI3	输入	JTAG测试检查	输入	串行线时钟
TDO / SWV	PI5	输出	JTAG测试数据输出	(输出)(注)	(串行线观测器输出)
TDI	PI6	输入	JTAG测试数据输入	-	-
$\overline{\text{TRST}}$	PI7	输入	JTAG测试 $\overline{\text{RESET}}$	-	-
TRACECLK	PI2	输出	跟踪时钟输出		
TRACEDATA0	PI1	输出	跟踪数据输出 0		
TRACEDATA1	PI0	输出	跟踪数据输出 1		
TRACEDATA2	PH0	输出	跟踪数据输出 2		
TRACEDATA3	PH1	输出	跟踪数据输出 3		

注：当SWV功能启用时。

在复位后，PI3, PI4, PI5, PI6, PI7 引脚被配置为调试端口功能引脚。其他调试接口引脚的功能需要按照要求进行编程。

使用低功耗模式时，注意下列几点。

注：若PI4 和PI5 被配置为TMS/SWDIO和TDO/SWV，则不管CGSTBYCR<DRVE>位的设置，甚至在STOP1 模式时，输出也继续被启用。

复位后调试接口引脚及相关端口的设置如表 3-2 所示。

表 3-2 复位后调试接口引脚及相关端口的设置

端口名称 (位名称)	调试功能	复位后相关端口的设置值				
		功能 (PxFR)	输入 (PxIE)	输出 (PxCR)	上拉 (PxPUP)	下拉 (PxPDN)
PI4	TMS/SWDIO	1	1	1	1	-
PI3	TCK/SWCLK	1	1	0	-	1
PI5	TDO/SWV	1	0	1	0	-
PI6	TDI	1	1	0	1	-
PI7	$\overline{\text{TRST}}$	1	1	0	1	-
PI2	TRACECLK	0	0	0	0	-
PI1	TRACEDATA0	0	0	0	0	-
PI0	TRACEDATA1	0	0	0	0	-
PH0	TRACEDATA2	0	0	0	0	-
PH1	TRACEDATA3	0	0	0	0	-

-：忽略

3.5 在停止模式时的外设功能

当Cortex-M3内核进入停止模式时，看门狗定时器(WDT)自动停止。其他外设功能继续运行。

3.6 与调试工具的连接

3.6.1 关于与调试工具的连接

与调试工具的连接，见制造商的建议。

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

注：禁止在STOP1 模式时用连接的调试工具测量功耗。

3.6.2 将调试接口引脚用作通用端口时的要点

在复位释放后，当用户程序将调试接口终端设置为通用端口时，此后调试工具将无法实现控制。

请注意为了再次连接调试工具，必须准备通过某种方法将通用端口改为调试接口功能的结构。

表 3-3 调试接口引脚使用表例子

	调试接口引脚						
	$\overline{\text{TRST}}$	TDI	TDO / SWV	TCK / SWCLK	TMS / SWDIO	TRACE DATA[3:0]	TRACE CLK
JTAG+SW (复位后)	0	0	0	0	0	×	×
JTAG+SW (无 $\overline{\text{TRST}}$)	×	0	0	0	0	×	×
JTAG+TRACE	0	0	0	0	0	0	0
SW	×	×	×	0	0	×	×
SW+SWV	×	×	0	0	0	×	×
调试功能禁用	×	×	×	×	×	×	×

0: 启用 × : 禁用(可用作通用端口)

4. JTAG接口

4.1 概述

TMPM365FYXBG配有一个与联合测试行动组(JTAG)规范相容的边界扫描接口,采用行业标准JTAG协议(IEEE标准 1149.1 · 1990 <包括IEEE标准 1449.1a · 1993>)。

本章说明了JTAG接口,边界扫描及接口使用的引脚和信号。

1. JTAG标准版本

IEEE标准 1149.1 · 1990 (包括IEEE标准 1149.1a · 1993)

2. JTAG指令

标准指令(BYPASS, SAMPLE/PRELOAD, EXTEST)

HIGHZ指令

CLAMP指令

然而,因为当JTAG正运行时,TMPM365FYXBG的内部电路复位启动,所以SAMPLE/RELOAD指令不运行。

3. IDCODE

不可用

4. 边界扫描寄存器(BSR)未包括的引脚

- a. 振荡器电路引脚 (X1, X2)
- b. JTAG控制引脚 (BSC)
- c. 电源/GND引脚 (包括ADC参考电源引脚)
- d. TEST引脚 (FTEST3)
- e. 功能引脚 ($\overline{\text{RESET}}$)
- f. 控制引脚 (MODE)

注:由于PF0引脚总是上拉的,在HIGHZ指令时,引脚输出高电平。

注:请注意模拟输入引脚的输入电平。

4.2 信号汇总和连接的例子

JTAG接口信号如下。

- TDI JTAG串行数据输入
- TDO JTAG串行数据输出
- TMS JTAG测试模式选择
- TCK JTAG串行时钟输入
- $\overline{\text{TRST}}$ JTAG测试复位输入
- ICE/JTAG测试选择输入(与启用信号相容)
- BSC 0: ICE, 1: JTAG

通过连接JTAG接口和与JTAG相符的开发工具，TMPM365FYXBG支持调试功能。调试信息，见使用的开发工具规范。

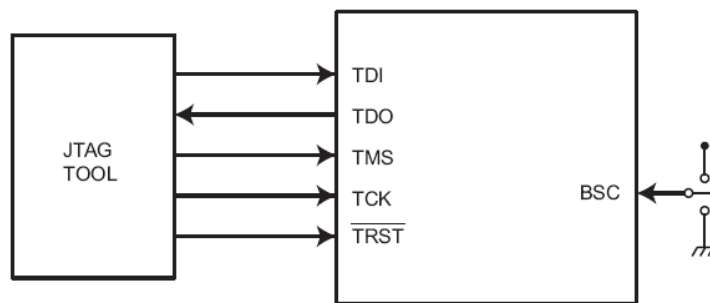


图 4-1 与JTAG开发工具连接的例子

模式设置引脚 (BSC)	运行模式
0	除边界扫描模式外，将该引脚设置为0。 TMPM365FYXBG在常规调试模式下运行。注：若内部BOOT正在运行，调试是不可用的。
1	TMPM365FYXBG在边界扫描模式下运行。

4.3 概述

随着曾经更密集集成电路(ICs)，表面贴装装置，印刷电路板(PCBs)上的双面元件装配及嵌入式凹口的演变，那些取决于内板和芯片连接等物理接触的电路内测试已变得越来越难以使用。越多的ICs已变得复杂，测试程序也变得越大越困难。

作为解决方案之一，边界扫描电路开始得到开发。边界扫描电路指在引脚和所述引脚所连IC内部电路之间放置的一系列移位寄存器。这些边界扫描单元被旁路掉；然而，当IC进入测试模式，测试程序能指导扫描单元沿移位寄存器路径传递数据，并进行各种诊断测试。为此，测试采用TCK，TMS，TDI，TDO和 $\overline{\text{TRST}}$ 五种信号。

JTAG边界扫描机构(在章节中简称为JTAG机构)实现在处理器，其所在印刷电路板及电路板上其他部件之间的连接测试。

JTAG机构无法单独测试处理器。

4.4 JTAG控制器和寄存器

处理器包含下列JTAG控制器和寄存器。

- 指令寄存器
- 边界扫描寄存器
- 旁路寄存器
- 装置标识寄存器
- 测试接入端口(TAP) 控制器

JTAG基本上是用TAP控制器状态机来监测TMS输入信号。当监测开始时，TAP控制器确定待实现的测试功能。它包括加载JTAG指令寄存器(IR)及通过数据寄存器(DR)开始串行数据扫描，如表 4-1 所示。当数据被扫描时，TMS引脚状态表示各个新数据字，并指示数据流的结束。数据寄存器应按照指令寄存器的内容进行选择。

4.5 指令寄存器

JTAG指令寄存器包括四个基于移位寄存器的单元。该寄存器用于选择要进行的测试和/或要接入的测试数据寄存器。如表 4-1 所示，该指令编码选择边界扫描寄存器或旁路寄存器。

表 4-1 JTAG指令寄存器位配置

指令编码 (MSB ~ LSB)	指令	所选数据寄存器
0000	EXTEST	边界扫描寄存器
0001	SAMPLE/PRELOAD	边界扫描寄存器
0100 ~ 1110	保留	保留
0010	HIGHZ	旁路寄存器
0011	CLAMP	旁路寄存器
1111	BYPASS	旁路寄存器

指令寄存器格式如图 4-2所示。

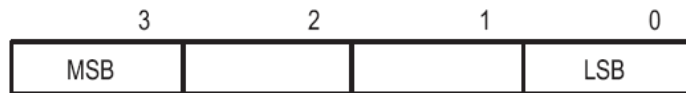


图 4-2 指令寄存器

指令编码从LSB移出到指令寄存器。

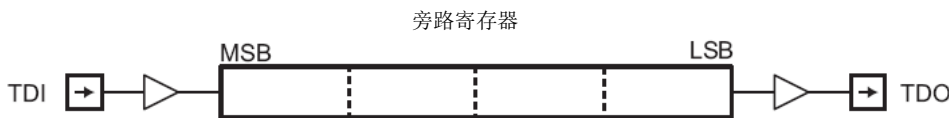


图 4-3 指令寄存器移位方向

旁路寄存器宽度 1 位。当TAP控制器在移位-DR (旁路) 状态时，TDI 引脚上的数据被移入旁路寄存器，旁路寄存器的输出移到TDO输出引脚上的数据输出。

本质上，旁路寄存器是一条替代路线，能旁路掉串行边界扫描链中的板级装置，这些板级装置对于具体测试并不需要。边界扫描链中的旁路寄存器的逻辑位置如图 4-4 所示。

旁路寄存器的使用会加速接入IC中的边界扫描寄存器，其在板级测试数据路径中保持有效。

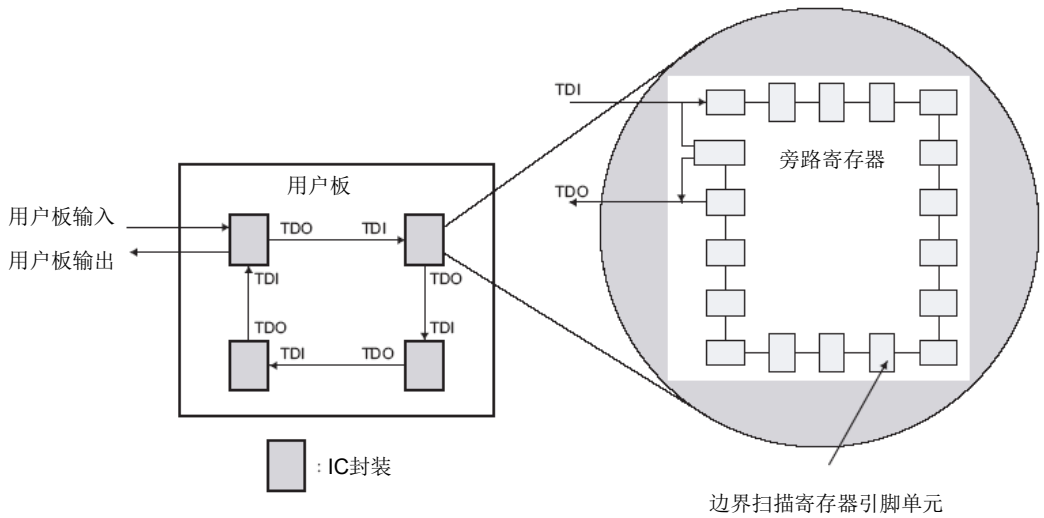


图 4-4 旁路寄存器的运行

4.6 边界扫描寄存器

除一些模拟输入和控制信号外，边界扫描寄存器提供TMPM365FYXBG处理器的所有输入和输出。通过将数据扫描到移位-DR状态的边界扫描寄存器中，TMPM365FYXBG的引脚实现要驱动的任何模式。当BSR在捕捉-DR状态时，通过启用边界扫描寄存器及使数据移位，就可检查处理器的输入数据。

边界扫描寄存器是单个 231-位-宽的基于移位寄存器的路径，该路径包含与TMPM365FYXBG上的输入引脚和输出引脚连接的单元。

TDI输入被加载到边界扫描寄存器的LSB。在TDO输出上，边界扫描寄存器的MSB被移出。

4.7 测试接入端口 (TAP)

测试接入端口(TAP)由五个信号引脚组成： $\overline{\text{TRST}}$ ，TDI，TDO，TMS和TCK。通过传递串行测试数据和指令，这些引脚控制测试。

如图 4-5 所示，数据被串行地扫描到TDI 引脚上的三个寄存器之一(指令寄存器，旁路寄存器或边界扫描寄存器)，或者从TDO引脚上的这三个寄存器之一中被扫出。

TMS输入控制主TAP控制器状态机的状态转换。TCK输入是一种特殊的测试时钟，它独立于任何具体芯片时钟或系统时钟，使串行JTAG数据能同步移位。

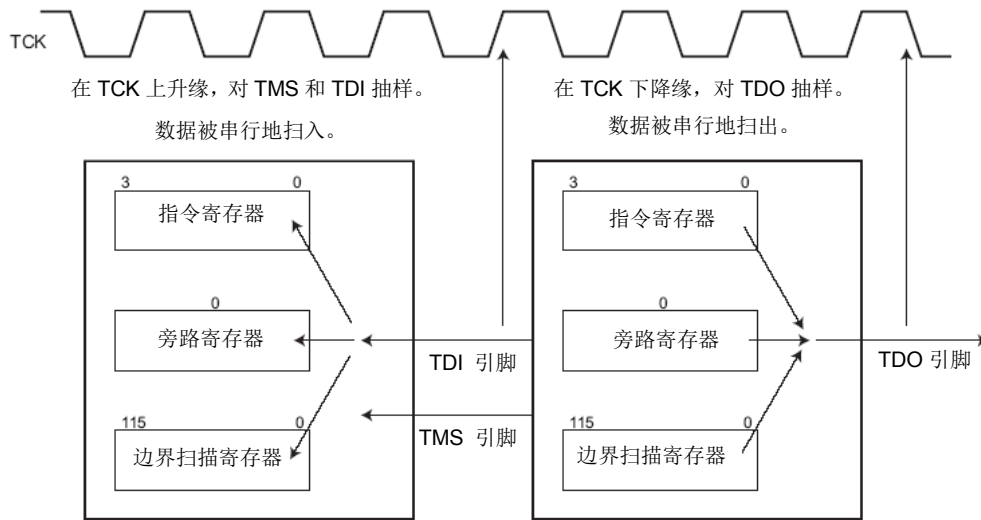


图 4-5 JTAG测试接入端口

在TCK输入时钟信号上升缘，对TDI和TMS引脚上的数据抽样。TDO 引脚上的数据在TCK时钟信号下降缘改变。

4.8 TAP控制器

处理器包含IEEE JTAG规范中规定的 16 状态TAP控制器。

4.9 使TAP控制器复位

用下列方法就能将TAP控制器状态机置于复位状态。

$\overline{\text{TRST}}$ 信号输入(低)的发出使TAP控制器复位。在处理器复位状态被释放后，应通过TCK输入五个连续上升缘保持输入信号被启用。保持TMS被启用以保持复位状态。

4.10 TAP控制器状态转换

TAP控制器状态转换图如图 4-6 所示。状态间的各个箭头用 1 或 0 标示，表示在TCK上升缘前必须设置TMS的逻辑值，以造成转换。

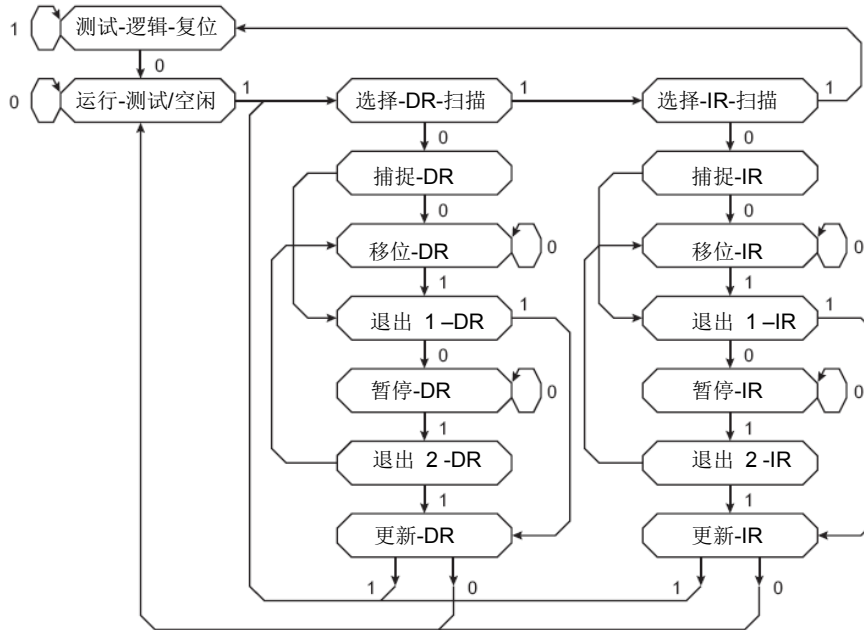


图 4-6 TAP控制器状态转换图

以下各段说明各控制器状态。图 4-6 中的左栏是数据栏，右栏是指令栏。数据栏和指令栏分别引用数据寄存器(DR)和指令寄存器(IR)。

- 测试-逻辑-复位

当TAP控制器在复位状态时，默认选择装置标识寄存器。边界扫描寄存器 MSB 被清除到 0，禁用输出。

当TMS为高时，TAP控制器保持该状态。若TAP控制器处于该状态时，TMS保持低，则控制器移到运行-测试/空闲状态。

- 运行-测试/空闲

在运行-测试/空闲状态下，只有在某些指令(例如内建自测试(BIST)指令)存在时，IC才能被置于测试模式。对于在该状态下不会造成任何活动的指令，当前指令选择的所有测试数据保持其先前状态。

当TMS保持低时，TAP控制器保持该状态。当TMS保持高时，控制器移到选择-DR-扫描状态。

- 选择-DR-扫描

这是一种临时控制器状态。此处，IC不执行任何具体功能。

若TAP控制器处于该状态时，TMS保持低，则控制器移到捕捉-DR状态。若TMS保持高，控制器移到选择-IR-扫描状态。

- 选择-IR-扫描

这是一种临时控制器状态。此处，IC不执行任何具体功能。

若TAP控制器处于该状态时，TMS保持低，则控制器移到捕捉-IR状态。若TMS保持高，控制器返回测试-逻辑-复位状态。

- 捕捉-DR

在该状态下，若当前指令所选测试数据寄存器有并行输入，则数据被并行载入数据寄存器的移位部分。若测试数据寄存器未有并行输入或者数据不必被载入所选数据寄存器，则数据寄存器保持其先前状态。

若TAP控制器处于该状态时，TMS保持低，则控制器移到移位-DR状态。若TMS保持高，控制器移到退出 1-DR状态。

- 移位-DR

在该控制器状态下，在TDI和TDO之间连接的测试数据寄存器串行地将数据移出。当TAP控制器处于该状态时，若TMS保持低，则它保持移位-DR状态，或者若TMS保持高，则它移到退出 1-DR状态。

- 退出 1-DR

这是一种临时控制器状态。

若TAP控制器处于该状态时，TMS保持低，则控制器移到暂停-DR状态。若TMS保持高，控制器移到更新-DR状态。

- 暂停-DR

该状态使指令寄存器所选数据寄存器的移位临时中止。指令寄存器和数据寄存器均保持其当前状态。

当TAP控制器处于这种状态时，若TMS保持低，则它移到暂停-DR状态，或者若TMS保持高，则它移到退出 2-DR状态。

- 退出 2-DR

这是一种临时控制器状态。

当TAP控制器处于这种状态时，若TMS保持低，则它返回移位-DR状态，或者若TMS保持高，则它移到更新-DR状态。

- 更新-DR

在该状态下，数据在TCK上升缘从移位寄存器路径被锁存到数据寄存器的并行输出上。当数据在相关移位寄存器路径中被移位时，并行输出保持的数据不会改变。

当TAP控制器处于该状态时，若TMS保持低，则它移到运行-测试/空闲状态，或者若TMS保持高，则它移到选择-DR-扫描状态。

- 捕捉-IR

在该状态下，数据被并行载入指令寄存器。加载的数据为0y0001。捕捉-IR状态用于测试指令寄存器。若有，指令寄存器中的故障可通过移出加载的数据而被检测出。

当TAP 控制器处于该状态，若TMS保持低，则它移到移位-IR状态，或者若TMS保持高，则它移到退出 1-IR状态。

- 移位-IR

在该状态下，指令寄存器连接在TDI和TDO之间，并在TCK上升缘将捕捉的数据移向其串行输出。

当TAP控制器处于该状态时，若TMS保持低，则它保持移位-IR状态，或者若TMS保持高，则它移到退出 1-IR状态。

- 退出 1-IR

这是一种临时控制器状态。

当 TAP 控制器处于该状态时，若TMS保持低，则它移到暂停 -IR 状态，或者若 TMS保持高，则它移到更新-IR状态。

- 暂停-IR

该状态使指令寄存器的移位临时中止。指令寄存器和数据寄存器均保持其当前状态。

当TAP控制器处于该状态时，若TMS保持低，则它保持暂停-IR状态，或者若TMS保持高，则它移到退出 2-IR状态。

- 退出 2-IR

这是一种 临时控制器状态。

当TAP控制器处于该状态时，若TMS保持低，则它移到移位-IR状态，或者若TMS保持高，则它移到更新-IR状态。

- 更新-IR

该状态使先前被移入指令寄存器中的指令在TCK上升缘被并行输出。然后，它变成当前指令，从而设置新操作模式。

当TAP控制器处于该状态时，若TMS保持低，则它移到运行-测试/空闲状态，或者若TMS保持高，则它移到选择-DR-扫描状态。

4.11 边界扫描顺序

关于处理器信号，边界扫描顺序如下表所示。

TDI → 1 (PK3) → 2 (PK2) → - → 69 (PI1) → 70 (PI2) → TDO

表 4-2 TMPM365FYXBG 处理器引脚的 JTAG 扫描顺序

编号	引脚名称	编号	引脚名称	编号	引脚名称	编号	引脚名称
	TDI						
1	PK3	21	PE0	41	PA4	61	PG4
2	PK2	22	PD0	42	PA3	62	PG5
3	PK1	23	PD1	43	PA2	63	PH4
4	PK0	24	PD2	44	PA1	64	PH3
5	PJ7	25	PD3	45	PA0	65	PH2
6	PJ6	26	PD4	46	PF7	66	PH1
7	PJ5	27	PD5	47	PF6	67	PH0
8	PJ4	28	PD6	48	PF5	68	PI0
9	PJ3	29	PD7	49	PF4	69	PI1
10	PJ2	30	PB7	50	PF3	70	PI2
11	PJ1	31	PB6	51	PF2		TDO
12	PJ0	32	PB5	52	PF1		
13	PE7	33	PB4	53	PF0		
14	PE6	34	PB3	54	PC2		
15	PE5	35	PB2	55	PC1		
16	PE4	36	PB1	56	PC0		
17	$\overline{\text{NMI}}$	37	PB0	57	PG0		
18	PE3	38	PA7	58	PG1		
19	PE2	39	PA6	59	PG2		
20	PE1	40	PA5	60	PG3		

4.12 JTAG控制器单元支持的指令

本节说明TMPM365FYXBG的 JTAG 控制器单元支持的指令。

1. EXTEST 指令

EXTEST指令用于外部互联测试。EXTEST指令使输出引脚的BSR单元在更新-DR状态移出测试模式，使输入引脚的BSR单元在捕捉-DR状态捕捉测试结果。

通常，在执行EXTEST前，用SAMPLE/PRELOAD将指令初始化模式移入边界扫描寄存器。若边界扫描寄存器未复位，模糊数据将在更新DR状态被传送，可能发生IC间的总线冲突。EXTEST指令被选时的数据流如图 4-7 所示。

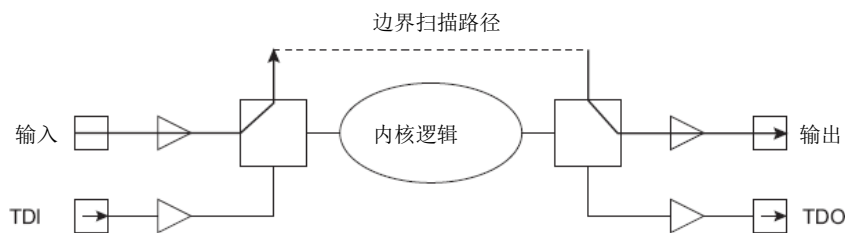


图 4-7 EXTEST指令被选时的测试数据流

下列步骤说明外部互联测试的基本测试程序。

1. 使TAP控制器复位到测试-逻辑-复位状态。
2. 用SAMPLE/PRELOAD指令加载指令寄存器。由此造成边界扫描寄存器在TDI和TDO之间被连接。
3. 通过移入某些数据，使边界扫描寄存器复位。
4. 将测试模式载入边界扫描寄存器。
5. 用EXTEST指令加载指令寄存器。
6. 将应用于输入引脚的数据捕捉到边界扫描寄存器中。
7. 移出捕捉的数据，同时移入下一个测试模式。
8. 在输出引脚的输出处，发出边界扫描寄存器中的测试模式。各测试模式重复第 6 步 ~ 第 8 步。

2. SAMPLE/PRELOAD指令

该指令针对TDI和TDO之间的边界扫描寄存器。顾名思义，SAMPLE/PRELOAD指令具有两个功能。

SAMPLE使IC的输入和输出引脚受到监测。当它这样做时，它不断开系统逻辑和IC引脚。SAMPLE在捕捉-DR状态下被执行。它主要用于在正常运行时捕捉TCK上升缘IC的I/O引脚值。SAMPLE/PRELOAD指令SAMPLE相的数据流如图 4-8 所示。

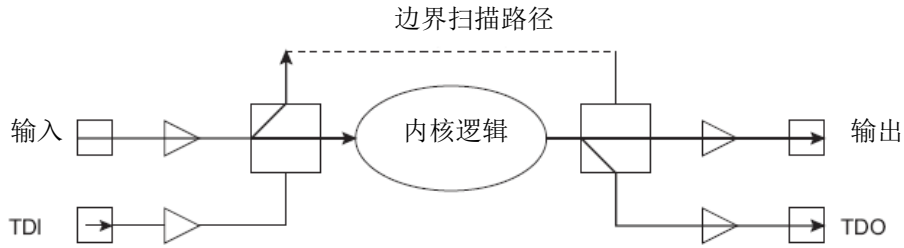


图 4-8 SAMPLE被选时的测试数据流

PRELOAD使边界扫描寄存器在其他指令被选前复位。例如，在选择EXTEST指令前，PRELOAD用于将复位数据载入边界扫描寄存器。在不干扰系统逻辑正常运行的情况下，PRELOAD实现边界扫描寄存器的数据移位。SAMPLE/PRELOAD指令PRELOAD相的数据流如图 4-9 所示。

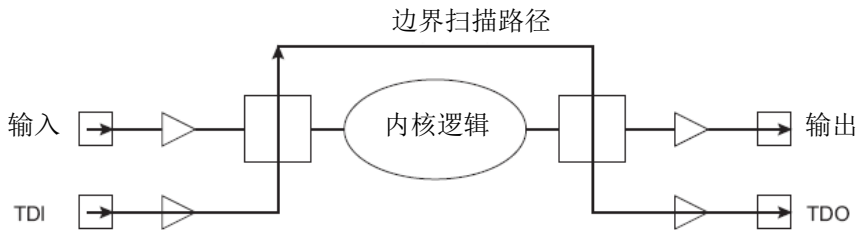


图 4-9 PRELOAD被选时的测试数据流

3. BYPASS指令

该指令针对JTDI和JTDO之间的旁路寄存器。当测试不需要控制或监测IC时，旁路寄存器提供最短的串行路径而将IC旁路掉(在JTDI和JTDO之间)。BYPASS指令不会对片上系统逻辑的正常运行造成干扰。BYPASS指令被选时通过旁路寄存器的数据流如图 4-10 所示。

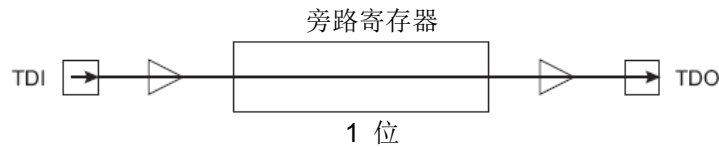


图 4-10 BYPASS指令被选时的测试数据流

4. CLAMP指令

CLAMP指令输出边界扫描寄存器按照PRELOAD指令编程的值，并执行旁路操作。CLAMP指令选择TDI和TDO之间的旁路寄存器。

5. HIGHZ指令

HIGHZ指令禁用内部逻辑电路的输出。当执行HIGHZ指令时，它将 3 状态输出引脚置于高阻抗状态。

HIGHZ指令也选择TDI和TDO之间的旁路寄存器。

• 注

本节说明处理器具体的JTAG边界扫描操作注意事项。

1. 由于PF0 引脚总是上拉，无论何时HIGHZ得到命令，高被输出。
2. 请注意模拟输入引脚的输入电平。
3. 用下列两种方法之一，就能释放JTAG电路的复位状态：

用 $\overline{\text{TRST}}$ ，初始化JTAG电路，然后解除 $\overline{\text{TRST}}$ 。

当把TMS引脚拉高时，提供至少 5 次时钟脉冲的TCK信号。



5. 存储器地址

5.1 存储器地址

TMPM365FYXBG的存储器地址以ARM Cortex-M3 处理器内核的存储器地址为基础。

内部ROM被地址到Cortex-M3 内核存储器编码，内部RAM被地址到SRAM区域，特殊功能寄存器(SFR)被地址到外设区域。

特殊功能寄存器(SFR)指示外设功能的I/O端口和控制寄存器。SRAM和SFR区域均包括在位段区域中。

CPU寄存器区域是处理器内核的内部寄存器区域。

有关各区域的详细资料，见“Cortex-M3 技术参考手册”。

注意若存储器故障启用，接入有"故障"指示的区域会造成存储器故障，或者若存储器故障禁用，会造成硬故障。切勿接入供应商特定区域。

5.1.1 TMPM365FYXBG存储器地址

TMPM365FYXBG存储器地址如图 5-1 所示。

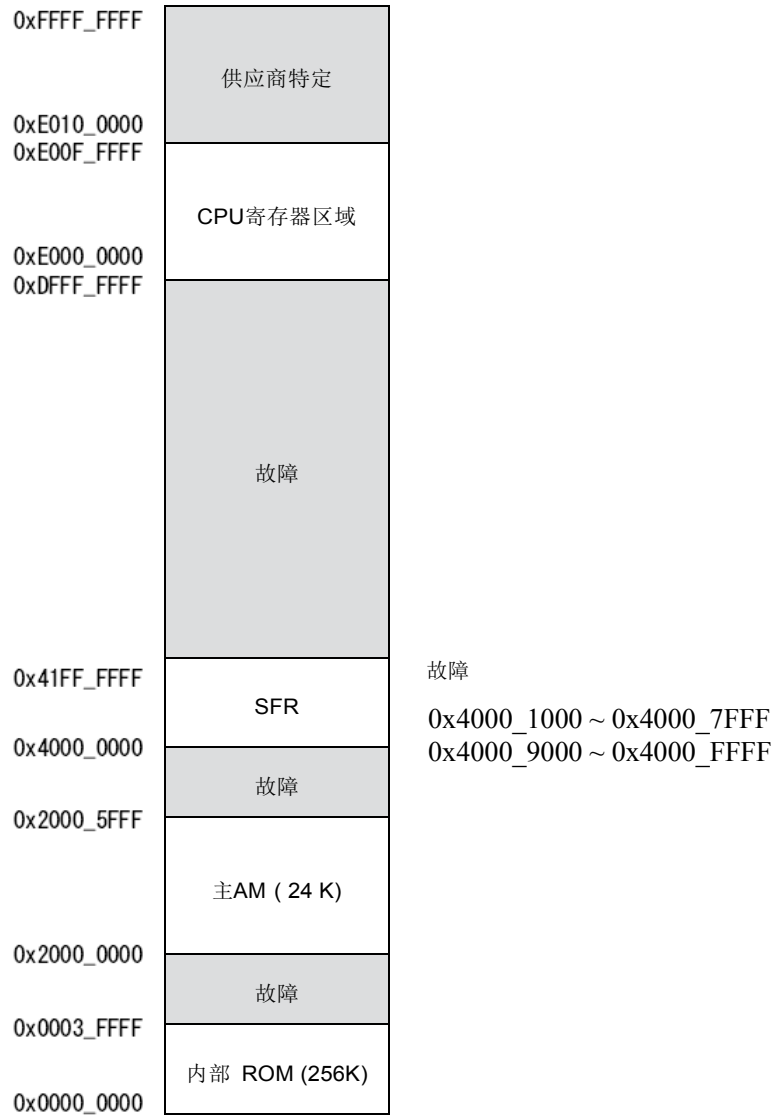


图 5-1 存储器地址

5.2 SFR区域细节

本节包含外设功能所分配到的SFR区域地址列表（0x4000_0000 ~ 0x41FF_FFFF）。

禁止接入表 5-1 中的保留区，非规定地址及各章中的保留区。关于SFR区域，各章中非规定区域被读作未定义值。写入该区域被忽略。

表 5-1 SFR区域细节

起始地址	结束地址	外设	保留
0x4000_0000	0x4000_3FFF	DMAC(2ch)	0x4000_0028 - 0x4000_002C 0x4000_0034
0x4000_4000	0x4000_7FFF	保留	
0x4000_8000	0x4000_9FFF	USB(1 ch)	
0x4000_A000	0x4003_FFFF	保留	
0x4004_0000	0x4004_7FFF	保留	
0x4004_8000	0x4004_BFFF	保留	
0x4004_C000	0x4004_FFFF	保留	
0x4005_0000	0x4005_3FFF	ADC(12ch)	0x4005_0064 - 0x4005_0073 0x4005_0F00 - 0x4005_0F8B
0x4005_4000	0x4005_BFFF	保留	
0x4005_C000	0x4005_CFFF	保留	
0x4005_D000	0x400B_FFFF	保留	
0x400C_0000	0x400C_1FFF	PORT	
0x400C_2000	0x400C_3FFF	保留	
0x400C_4000	0x400C_5FFF	TMRB(10ch)	
0x400C_6000	0x400D_FFFF	保留	
0x400E_0000	0x400E_0FFF	I2C/SIO	0x400E_0800 - 0x400E_0FFF
0x400E_1000	0x400E_1FFF	UART/SIO	0x400E_1134 - 0x400E_1137
0x400E_2000	0x400F_0FFF	保留	
0x400F_1000	0x400F_1FFF	保留	
0x400F_2000	0x400F_2FFF	WDT	0x400F_2100 - 0x400F_2FFF
0x400F_3000	0x400F_3FFF	CG	0x400F_3100 - 0x400F_3FFF
0x400F_4000	0x41FF_EFFF	保留	
0x41FF_F000	0x41FF_F03F	FLASH	0x41FF_F000 - 0x41FF_F007 0x41FF_F014 - 0x41FF_F017 0x41FF_F024 - 0x41FF_F02B
0x41FF_F040	0x41FF_FFFF	保留	



6. 复位

TMPM365FYXBG有四个复位源：外部复位引脚($\overline{\text{RESET}}$)，看门狗定时器(WDT)及应用中断和复位控制寄存器中的设置<SYSRESETREQ>。

WDT产生的复位见WDT章节。

<SYSRESETREQ>产生的复位见“Cortex-M3 技术参考手册”。

6.1 初始状态

刚好在上电之后，TMPM365FYXBG内部电路是未定义的。

寄存器设置和引脚状态是未定义的，直到外部复位引脚($\overline{\text{RESET}}$)在施加所有电源电压(DVDD3A, DVDD3C, RVDD3 和AVDD3)后接收低电平。

6.2 冷复位

为了保持稳定，上电顺序必须包括内部稳压器，内部闪存和内部振荡器的时间。

在 TX03 中，内部电路自动插入内部稳压器，内部闪存和内部振荡器的时间。为了保持该时间，在电源电压电平处于工作电压电平后至少 1 ms内，复位引脚必须保持“低”。

在复位引脚($\overline{\text{RESET}}$)电平为“高”后，内部复位信号被释放，约需 0.8 ms。

上电后的冷复位顺序如图 6-1 所示。

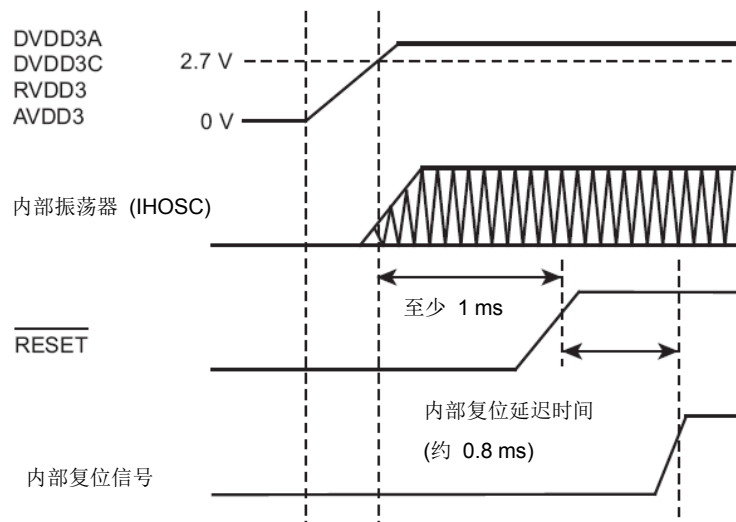


图 6-1 冷复位顺序

注：当恢复电源时，也应用上述顺序。

6.3 热复位

6.3.1 复位时间

前提条件是保证电源电压在工作范围内，并且高频振荡器正提供稳定振荡。

为了复位TMPM365FYXBG，保持外部复位引脚($\overline{\text{RESET}}$)至少 12 个系统时钟。

在复位引脚($\overline{\text{RESET}}$)为"高"后，内部复位信号被释放，约需 0.8 ms。

6.4 复位后

复位使Cortex-M3处理器内核大多数系统控制寄存器和内部功能寄存器初始化。

处理器内核的调试部件(FPB, DWT, ITM)寄存器，时钟发生器的CGRSTFLG寄存器和FCSECBIT寄存器以冷复位方式被初始化。

在复位后，PLL乘法电路是非激活的，必要时必须启用。

当完成复位异常处理时，程序分支到复位中断服务程序。

注：该复位操作可改变内部RAM状态。

7. 看门狗定时器 (WDT)

看门狗定时器(WDT)用于检测噪声或其他干扰造成的CPU故障(失控), 并修复它们, 使CPU恢复正常。

若看门狗定时器检测到失控现象, 它会生成INTWDT中断或复位。

注: INTWDT中断是不可屏蔽中断(NMI)的一个因素。

此外, 通过输出"低", 看门狗定时器将检测的故障从看门狗定时器引脚($\overline{\text{WDTOUT}}$)通知外设装置。

注: 该产品没有看门狗定时器输出引脚($\overline{\text{WDTOUT}}$)。

7.1 配置

看门狗定时器方块图如图 7-1 所示。

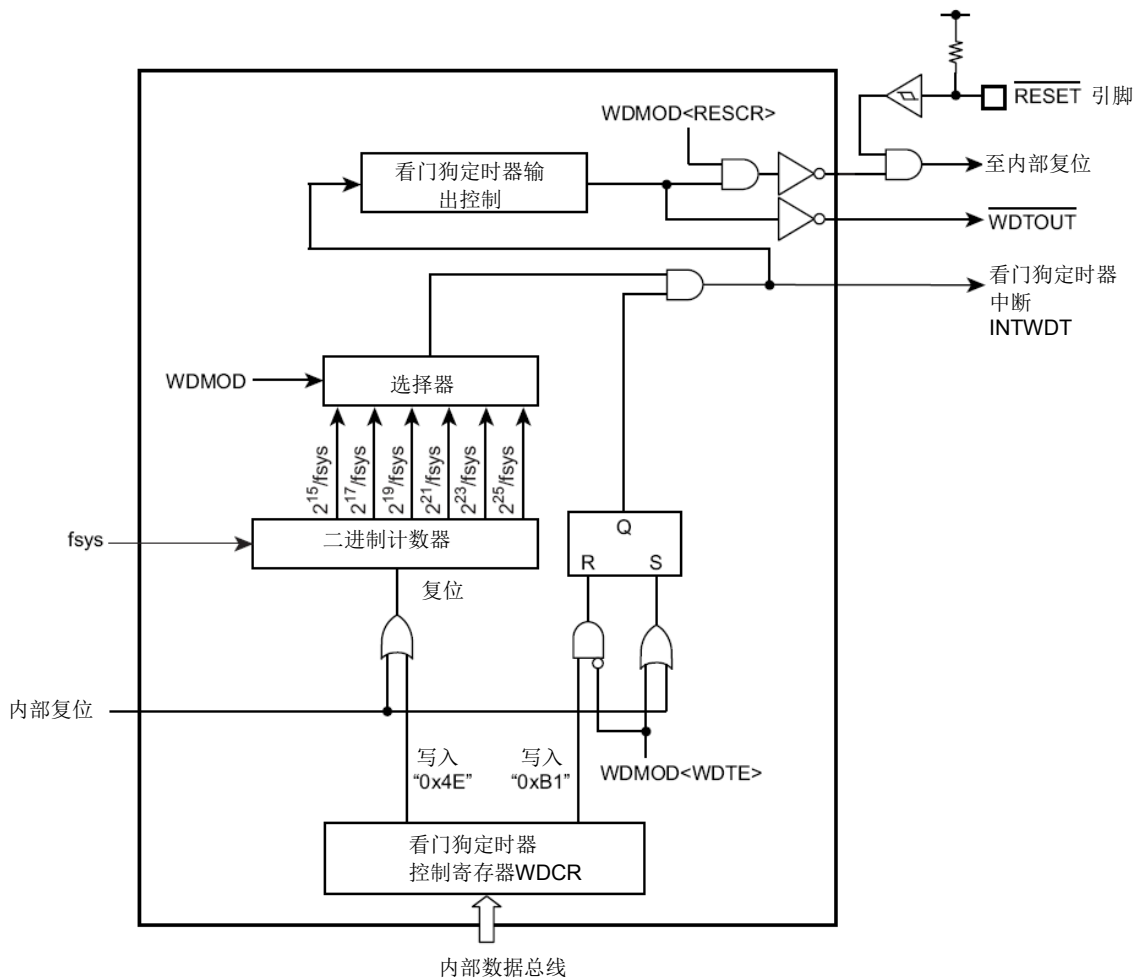


图 7-1 看门狗定时器方块图

7.2 寄存器

看门狗定时器控制寄存器和地址如下。

基址 = 0x400F_2000

寄存器名称		地址(基+)
看门狗定时器寄存器地址	WDMOD	0x0000
看门狗定时器控制寄存器	WDCR	0x0004

7.2.1 WDMOD (看门狗定时器模式寄存器地址)

	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 7-1) 000: $2^{15}/fsys$ 100: $2^{23}/fsys$ 001: $2^{17}/fsys$ 101: $2^{25}/fsys$ 010: $2^{19}/fsys$ 110: 禁止设置。 011: $2^{21}/fsys$ 111: 禁止设置。
3	-	R	读作 0。
2	I2WDT	R/W	在IDLE模式时运行 0: 停止 1: 在运行中
1	RESCR	R/W	在检测到故障后运行 0: INTWDT中断请求生成.(注: 1: 复位
0	-	R/W	写入 0。

注： INTWDT中断是不可屏蔽中断(NMI)的一个因素。

表 7-1 看门狗定时器检测时间($f_c = 48 \text{ MHz}$)

时钟齿轮值 CGSYSCR<GEAR[2:0]>	WDMOD<WDTP[2:0]>					
	000	001	010	011	100	101
000 (f_c)	0.68 ms	2.73 ms	10.92 ms	43.69 ms	174.76 ms	699.05 ms
100 ($f_c/2$)	1.37 ms	5.46 ms	21.85 ms	87.38 ms	349.53 ms	1.40 s
101 ($f_c/4$)	2.73 ms	10.92 ms	43.69 ms	174.76 ms	699.05 ms	2.80 s
110 ($f_c/8$)	5.46 ms	21.85 ms	87.38 ms	349.53 ms	1.40 s	5.59 s
111 ($f_c/16$)	10.92 ms	43.70 ms	174.8 ms	699.1 ms	2.80 s	11.18 s

7.2.2 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/清除编码 0xB1: 禁用编码 0x4E: 清除编码 其他: 保留

7.3 操作

7.3.1 基本运行

看门狗定时器由二进制计数器组成，这些计数器将系统时钟(fs_{sys}) 用作输入。检测时间由WDMOD<WDTP[2:0]>在2¹⁵，2¹⁷，2¹⁹，2²¹，2²³和2²⁵之间选择。规定的检测时间过去，看门狗定时器中断(INTWDT)生成，看门狗定时器输出引脚($\overline{\text{WDTOUT}}$)输出"低"。

为了检测噪声或其他干扰造成的CPU故障(失控)，在INTWDT中断生成前，应该用软件指令清除看门狗定时器二进制计数器。若二进制计数器未被清除，INTWDT不可屏蔽中断就会生成。因此，CPU进行故障检测，并运行故障对策程序，以恢复正常工作。

此外，可能通过连接看门狗定时器输出引脚和外设复位引脚而解决CPU故障(失控)问题。

注：本产品不包括看门狗定时器输出引脚($\overline{\text{WDTOUT}}$)。

7.3.2 运行模式和状态

在复位被清除后，看门狗定时器立即开始运行。

若不使用看门狗定时器，它应被禁用。

由于高速频率时钟停止，看门狗定时器无法使用。在切换到下列模式前，看门狗定时器应被禁用。在IDLE模式时，它的运行取决于WDMOD <I2WDT>的设置。

- STOP1 模式

此外，二进制计数器在调试模式时自动停止。

7.4 在检测到故障(失控)时的运行情况

7.4.1 INTWDT中断生成

INTWDT中断生成(WDMOD<RESCR>="0")的情况如图 7-2 所示。

当二进制计数器发生溢出时，INTWDT中断就会产生。它是不可屏蔽中断(NMI)的一个因素。因此，CPU检测不可屏蔽中断，并运行对策程序。

不可屏蔽中断因素为复数。CGNMIFLG识别不可屏蔽中断因素。在INTWDT中断的情况下，CGNMIFLG<NMIFLG0>就会被设置。

当INTWDT中断生成时，看门狗定时器输出引脚($\overline{\text{WDTOUT}}$)同时输出"低"。通过看门狗定时器的清除行为(将清除编码0x4E写入WDCR寄存器)， $\overline{\text{WDTOUT}}$ 变"高"。

注：本产品无看门狗定时器输出引脚($\overline{\text{WDTOUT}}$)。

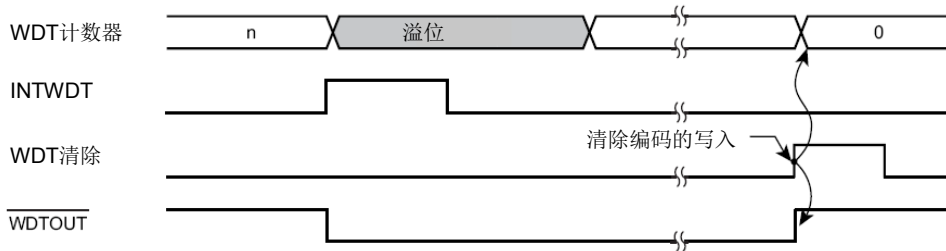


图 7-2 INTWDT 中断生成

7.4.2 内部复位生成

内部复位生成(WDMOD<RESCR>=" 1 ")如图 7-3 所示。

MCU通过二进制计数器的溢出而复位。在这种情况下，对于 32 种状态，复位状态继续。时钟被初始化，以致输入时钟(f_{sys})与内部高速频率时钟(f_{osc})相同。这意味着 $f_{sys} = f_{osc}$ 。

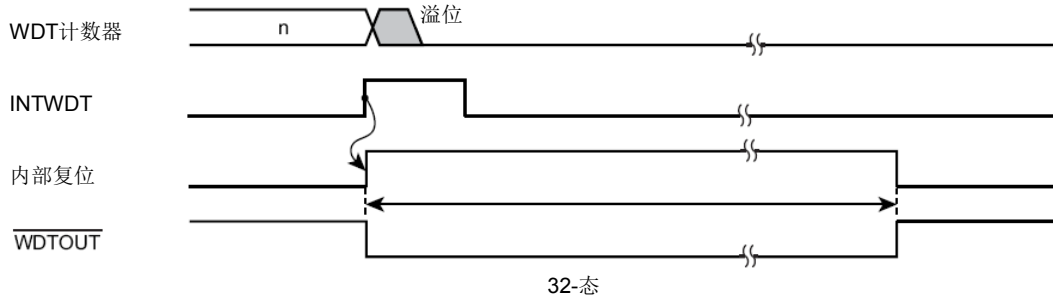


图 7-3 内部复位生成

7.5 控制寄存器

看门狗定时器(WDT)受两个控制寄存器WDMOD和WDCR控制。

7.5.1 看门狗定时器寄存器地址 (WDMOD)

1. 规定看门狗定时器的检测时间<WDTP[2:0]>。

将看门狗定时器检测时间设置为WDMOD<WDTP[2:0]>。在复位后，它被初始化到WDMOD<WDTP[2:0]> = "000"。

2. 启用/禁用看门狗定时器<WDTE>。

当复位时，WDMOD <WDTE>被初始化到"1"，并启用看门狗定时器。

为了禁用看门狗定时器，防止故障产生的错误写入，先将<WDTE> 位设置为"0"，再将禁用编码(0xB1)写入WDCR寄存器。

为了将看门狗定时器的状态从"禁用"变为"启用"，将<WDTE>位设置为"1"。

3. 看门狗定时器输出复位连接<RESCR>

该寄存器规定WDTOUT是用于内部复位还是用于中断。在复位后，WDMOD<RESCR>被初始化到"1"，内部复位通过二进制计数器的溢出而生成。

7.5.2 看门狗定时器控制寄存器(WDCR)

这是一个用于禁用定时器功能并控制二进制计数器的清除功能的寄存器。

7.5.3 设置例子

7.5.3.1 禁用控制

在WDMOD <WDTE>设置为"0"后，将禁用编码(0xB1)写入该WDCR寄存器，看门狗定时器就会被禁用，二进制计数器就会被清除。

	7	6	5	4	3	2	1	0		
WDMOD	←	0	-	-	-	-	-	-	将<WDTE>设置为"0"。	
WDCR	←	1	0	1	1	0	0	0	1	写入禁用编码 (0xB1)。

7.5.3.2 启用控制

将WDMOD <WDTE>设置为"1"。

	7	6	5	4	3	2	1	0	
WDMOD	←	1	-	-	-	-	-	-	将<WDTE>设置为"1"。

7.5.3.3 看门狗定时器清除控制

将清除编码(0x4E)写入WDCR寄存器就能清除二进制计数器，并且开始计数。

	7	6	5	4	3	2	1	0		
WDCR	←	0	1	0	0	1	1	1	0	写入清除编码 (0x4E)。

7.5.3.4 看门狗定时器的检测时间

在使用 $2^{21}/f_{sys}$ 的情况下，将WDMOD<WDTP[2:0]>设置为"011"。

	7	6	5	4	3	2	1	0	
WDMOD	←	1	0	1	1	-	-	-	-



8. 时钟/模式控制

8.1 特征

时钟/模式控制块用于选择时钟齿轮，预分频器时钟及PLL时钟乘法电路和振荡器的预热。

还有一个低功耗模式，该模式可通过模式推移减少功耗。本节主要对时钟工作模式与模式推移进行了说明。

该时钟/模式控制程序块具备以下功能：

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

除NORMAL模式外，TMPM365FYXBG能在低功耗模式下运行，按照其使用条件降低功耗。

8.2 寄存器

8.2.1 寄存器列表

下表给出了与CG相关的各寄存器与地址。

基址 = 0x400F_3000

寄存器名称		地址 (基+)
系统控制寄存器	CGSYSCR	0x0000
振荡控制寄存器	CGOSCCR	0x0004
待机控制寄存器	CGSTBYCR	0x0008
PLL选择寄存器	CGPLLSEL	0x000C
保留	-	0x0010
保留	-	0x0014
USB时钟控制寄存器	CGUSBCTL	0x0038
保护寄存器	CGPROTECT	0x003C

注：禁止接入"保留"区。

8.2.2 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	-	-	-	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	-	-	-	FPSEL	-	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
31-21	-	R	读作 0。
20	FCSTOP	R/W	ADC时钟 0: 有效 1: 停止 停止提供ADC时钟。 在复位后, 就可提供ADC时钟。 当设置"1"(停止)时, 必须先确认ADC已停止或完成。
19-18	-	R	读作 0。
17-16	SCOSEL[1:0]	R/W	SCOUT输出 00: 保留 01: fsys/2 10: fsys 11: φT0 从SCOUT引脚输出规定时钟。
15-14	-	R	读作 0。
13	-	R/W	读作 0。写入"0"
12	FPSEL	-	fperiph 0: fgear 1: fc 将源时钟规定为fperiph。 不管时钟齿轮模式, 选择fc就能固定fperiph。
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 将该预分频时钟指定给外设I/O。
7-3	-	R	读作"0"。
2-0	GEAR[2:0]	R/W	高速时钟齿轮(fc)齿轮 000: fc 100: fc/2 001: 保留 101: fc/4 010: 保留 110: fc/8 011: 保留 111: fc/16

注：切勿设置保留值。



8.2.3 CGOSCCR (振荡控制寄存器)

	31	30	29	28	27	26	25	24
比特符号	WUODR							
复位后	1	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	WUODR				HWUPSEL	EHOSCSEL	OSCSEL	XEN2
复位后	0	0	0	0	0	0	0	1
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	-	-	XEN1
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	-	-	-	-	-	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	WUODR[11:0]	R/W	预热计数器设置值。 对于上 12-位计数器值的定时器，设置 16-位定时器。
19	HWUPSEL	R/W	高速预热时钟。 0: 内部 (f_{IHOSC}) 1: 外部 (f_{EOSC}) 由高速振荡器选择预热计数器。
18	EHOSCSEL	R/W	外部振荡器。 0: 输入外部时钟 1: 外部晶体振荡器
17	OSCSEL	R/W	高速振荡器 0: 内部高速振荡器(EHOSC) 1: 外部高速振荡器(EHCLKIN) 由高速振荡器选择预热计数器。
16	XEN2	R/W	内部高速振荡器的运行 0: 停止 1: 振荡
15-12	-	R/W	在复位后写入"0"。
11-10	-	R	读作 0。
9	-	R/W	写入"0"。
8	XEN1	R/W	外部高速振荡器模式 0: 停止 1: 振荡
7-3	-	R/W	写入"00110"
2	PLLON	R/W	PLL(乘法电路)的运行(注 3) 0: 停止 1: 振荡
1	WUEF	R	预热定时器(WUP)的状态 0: WUP完成 1: WUP 有效 监测预热定时器的状态。

0	WUEON	W	振荡器预热定时器 (WUP)的运行 0: 忽略 1: WUP start 启用以启动该预热计时器。读作"0"。
---	-------	---	--

注 1: 预热设置见"8.3.4 预热功能"章节。

注 2: 当选择外部振荡器(输入外部时钟)时, 应在设置<EHOSCSEL>后选择<OSCSEL>。(切勿同时选择)

注 3: 在设置CGOSCCR<PLLON>="1"后, 进行预热操作, 然后设置CGPLLSEL<PLLSEL>="1"。

注 4: 由于内部高速振荡器启动, 在从STOP1模式返回后, 寄存器CGOSCCR的相关位<HWUPSEL>, <OSCSEL>, <XEN2>, <XEN1>, <PLLON>及和CGPLLSEL<PLLSEL>被初始化。

注 5: 当内部高速振荡器(IHOSC) 用作系统时钟时, 切勿使用PLL乘法。

注 6: 当使用内部高速振荡器(IHOSC)时, 切勿将它用作需要高精度保证的系统时钟。

8.2.4 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	-	-	-	-	-	-	-	DRVE
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	-	-	-	-	-	STBY		
复位后	0	0	0	0	0	0	1	1

位	比特符号	类型	功能
31-20	-	R	读作 0。
19-17	-	R/W	在复位后写入"0"。
16	DRVE	R/W	在STOP1模式时的引脚状态。 0: 在STOP1模式时未激活 1: 在STOP1模式时激活
15-3	-	R	读作 0
2-0	STBY[2:0]	R/W	低功耗模式 000: 保留 001: STOP1 010: 保留 011: IDLE 100: 保留 101: 保留 110: 保留 111: 保留 切勿设置保留值。

8.2.5 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	PLLSET							
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	PLLSET							PLLSEL
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。
15-1	PLLSET[14:0]	R/W	PLL乘法值(切勿使用非下列值) 0x381E: 8 倍乘法
0	PLLSEL	R/W	PLL的使用 0: f_{osc} 1: f_{PLL} (使用PLL) 规定使用或不用已被乘以PLL的时钟。 在复位后, "fosc(内部高速振荡器)"自动设置。当使用PLL时, 必须进行复位。

注 1: 当CGOSCCR<PLLON> = 0 (PLL停止)时, 选择PLL乘法值。

注 2: 选择表 8-2 所示PLL乘法值。

注 3: 由于内部高速振荡器启动, 在从STOP1模式返回后, 相关位<PLLSEL>, CGOSCCR<HWUPSEL>, <OSCSSEL>, <XEN2>, <XEN1>和<PLLON>被初始化。

注 4: 当内部高速振荡器(IHOSC) 用作系统时钟时, 切勿使用PLL乘法。

8.2.6 CGUSBCTL (USB时钟控制寄存器)

	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	-	-	-	-	-	-	USBCLKSEL	USBCLKEN
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
31-10	-	R	读作 0。
9	USBCLKSEL	R/W	USB源时钟选择 0: PLL时钟 1: 外部输入时钟 选择 USB装置块的源时钟。
8	USBCLKEN	R/W	USB源时钟控制 0: 时钟禁用 1: 时钟启用
7-1	-	R	读作 0。
0	-	R/W	写作 0。

注 1: 当修改<USBCLKSEL>时, <USBCLKEN>必须被清除为"0"。

注 2: 同时修改<USBCLKSEL>和<USBCLKEN>。

8.2.7 CGPROTECT (保护寄存器)

	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	CGPROTECT							
复位后	1	1	0	0	0	0	0	1

位	比特符号	类型	功能
31-6	-	R	读作"0"。
7-0	CGPROTECT	R/W	寄存器保护控制，0x400F_3000 ~ 0x400F_303B：将配置写入各寄存器。 0xC1：寄存器写入启用 除0xC1外：寄存器写入禁用 初始值为"0xC1"，表示将启用写入各寄存器，当写入非"0xC1"值时，除CGPROTECT寄存器之外的各寄存器不可能被写入。

8.3 时钟控制

8.3.1 时钟类型

各时钟被定义如下：

fosc	: 内部振荡器生成的时钟。从X1和 X2引脚来的时钟输入。
f _{PLL}	: 由PLL输出的 8 倍时钟。
fc	: CGPLLSEL<PLLSEL>所规定的时钟 (高速时钟)
fgear	: CGSYSCR<GEAR[2:0]>所规定的时钟(齿轮时钟)
fsys	: 与fgear时钟相同所规定的时钟(系统时钟)
fperiph	: CGSYSCR<FPSEL>所规定的时钟
φT0	: CGSYSCR<PRCK[2:0]> 所规定的时钟(预分频时钟)

齿轮时钟fgear和预分频器时钟φT0 可分频如下。

高速时钟	: fc, fc/2, fc/4, fc/8, fc/16
预分频器时钟	: fperiph, fperiph/2, fperiph/4, fperiph/8, fperiph/16, fperiph/32

8.3.2 复位之后的初始值

复位操作可初始化该时钟配置 (如以下所述)。

内部高速振荡器	: 振荡
外部高速振荡器	: 停止
PLL (锁相环电路)	: 停止
高速时钟齿轮	: fc (无分频)

复位操作可导致所有时钟配置均变为与fosc的相同。

fc = fosc
fsys = fosc
φT0 = fosc

8.3.3 时钟系统图

时钟系统图如 8-1 所示。

在复位后，箭头所示的选择器输入时钟被设置为默认值。

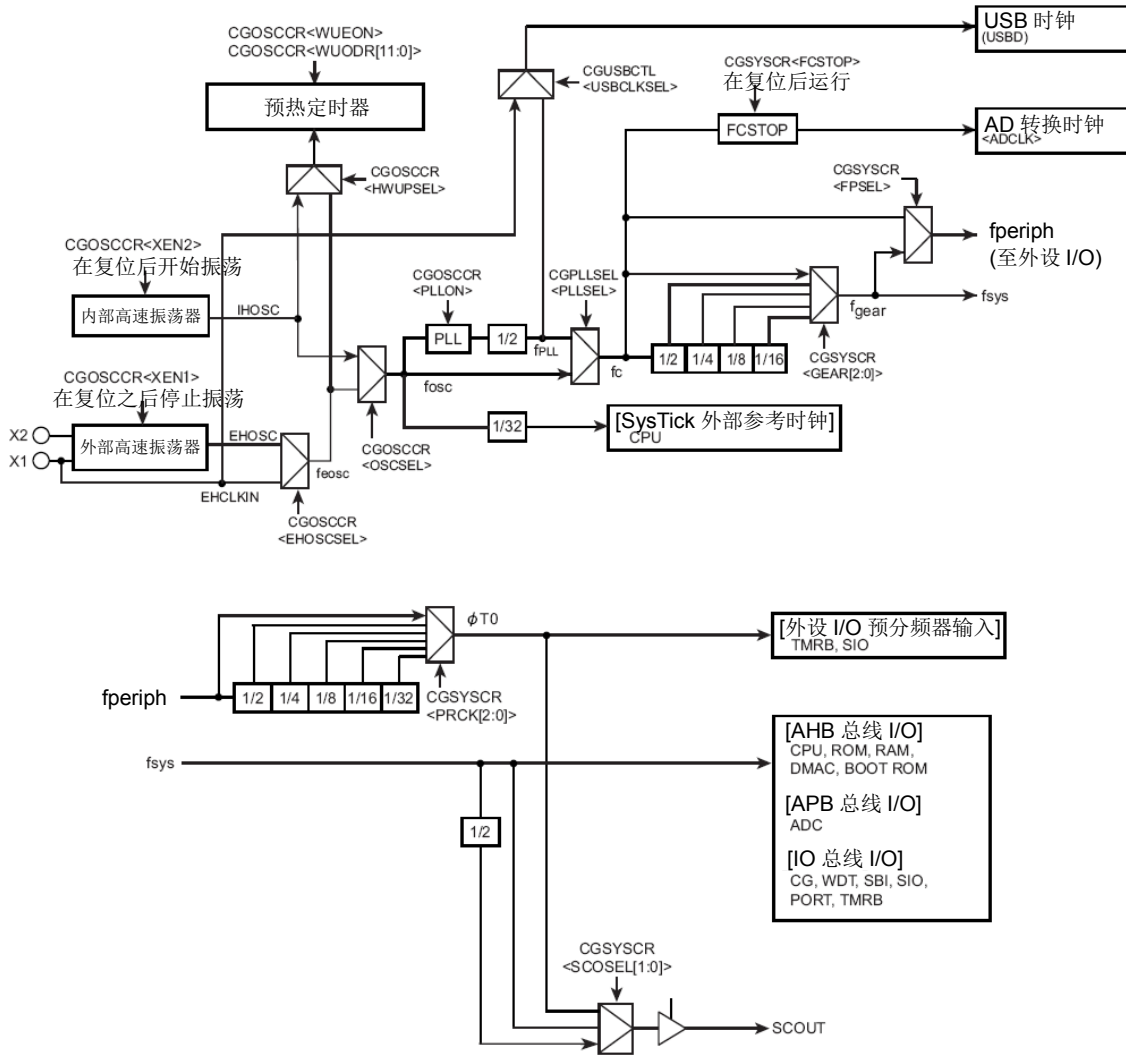


图 8-1 时钟方块图

8.3.4 预热功能

该预热功能可利用该预热计时器，固定该振荡器稳定时间与PLL。

预热功能详见"8.6.6 预热"。

如何配置该预热功能。

1. 规定向上计数时钟

在CGOSCCR<HWUPSEL>中规定预热计数器的向上计数时钟

2. 规定预热计数器值

值能用下列公式求得，舍入下 4 位，并设置为<WUODR[11:0]>位。通过设置CGOSCCR<WUODR[11:0]>，可选择预热时间。

$$\text{预热周期次数} = \frac{\text{设置的预热时间}}{\text{输入频率周期 (s)}}$$

<示例 1> 当使用高速振荡器 8 MHz，并设置预热时间 5 ms时。

$$\frac{\text{设置的预热时间}}{\text{输入频率周期 (s)}} = \frac{5\text{ms}}{1/8\text{MHz}} = 40000 \text{ 次} = 0x9C40$$

舍入下 4 位，将0x9C4设置为CGOSCCR<WUODR[11:0]>

3. 确认预热的开始和完成

通过软件(指令)，利用CGOSCCR<WUEON><WUEF>确认预热的开始与完成。

当CGOSCCR<WUEON>设置为"1"时，预热开始向上计数。预热的完成能用CGOSCCR<WUEF>进行确认。

预热功能设置的示例

表 8-1 <示例> 从STOP模式到NORMAL模式的转换(选择内部高速振荡器)

CGOSCCR<WUODR[11:0]> = "0x9C4"	: 规定预热时间
CGOSCCR<WUODR[11:0]> 读取	: 确认预热时间得到反映
CGOSCCR<XEN2> = "1"	: 内部高速振荡器(IHOSC)启用
CGOSCCR<WUEON> = "1"	: 开启预热定时器(WUP)
CGOSCCR<WUEF> 读取	: 保持等待, 直至状态变为"0"(预热完成)

注 1: 为了保持稳定, 在使用外部时钟时不必规定预热时间。

注 2: 预热定时器按照振荡时钟运行。若振荡频率波动, 它可能会包含错误。因此, 该预热时间应被视为近似时间。

注 3: 将CGOSCCR<WUODR[11:0]>设置为预热计数值, 等到该值得到反映, 然后通过执行命令"WFI", 切换到待机模式。

注 4: 由于内部高速振荡器启动, 当从STOP1 模式返回时, 相关位CGPLLSEL<PLLSEL>, CGOSCCR<HWUPSEL>, <OSCSEL>, <XEN2>, <XEN1>和<PLLON> 被初始化, 而CGOSCCR<WUODR[11:0]>不被初始化。

8.3.5 时钟乘法电路 (PLL)

该电路输出 f_{PLL} 时钟，它为高速振荡器输出时钟(f_{osc})的 8 倍。结果，振荡器输入频率可能是低频率，内部时钟变成高速。

8.3.5.1 启动运行

在复位之后该PLL被禁用。

为了使用PLL，当CGOSCCR<PLLON>为"0"时，将CGPLLSEL<PLLSET>设置为乘法值。然后等到 100 μ s(作为PLL初始稳定时间)已过去，再将<PLLON>设置为"1"，启动PLL的运行。此后，为了使用 f_{PLL} 时钟，先等到 100 μ s(作为锁定时间)已过去，再将CGPLLSEL <PLLSEL> 设置为 "1"。由此使 f_{osc} 能乘以 8。注意在使用预热功能时，必须有预热时间，直到PLL的运行变得稳定。

注：当使用内部振荡器 (IHOSC) 时，切勿使用PLL。

关于 8 倍值，仅允许下列设置。

乘数	<PLLSEL>
8	0x381E

PLL 启动顺序如下。

(1) PLL运行启动程序

在复位的初始状态 (注)

CGOSCCR<PLLSEL> = "0" (不使用PLL)

CGOSCCR<PLLON> = "0" (停止PLL)



PLL乘法设置

CGPLLSEL<PLLSET> = 乘法值设置数



PLL运行设置

CGOSCCR<PLLON> = "1" (运行PLL)



锁定时间

锁定时间必须约 100 μ s 。



PLL选择设置

CGPLLSEL<PLLSEL> = "1" (使用PLL)



能使用倍乘的系统时钟。

8.3.6 系统时钟

内部高速振荡时钟和外部高速振荡时钟是一个振荡器连接或输入时钟，能用作系统时钟的源时钟。

当使用内部高速振荡时钟时，切勿将它用作需要高精度保证的系统时钟。

当使用外部高速振荡时钟时，PLL功能能以乘法的方式得到使用。

源时钟		频率	使用PLL
内部高速振荡 (IHOSC)		10 MHz	无法使用
外部高速振荡	振荡器 (EHOSC)	8 ~ 12 MHz	废弃不用或 8 倍
	输入时钟 (EHCLKIN)	8 ~ 12 MHz, 48 MHz	

时钟二分频能用作系统时钟和ADC时钟。能分别使用的频率如下。

	系统时钟	ADC时钟
工作频率 (MHz)	1 ~ 48	40 (最大值)

系统时钟能由CGSYSCR<GEAR[2:0]>进行分频。虽然在运行时能变更设置，但是实际开关稍有延时。

按照PLL和时钟齿轮的设置，工作频率的示例如表 8-2 所示。

表 8-2 按照PLL乘法和时钟齿轮的设置，工作频率的设置示例

外部振荡器 (MHz)	外部时钟输入 (MHz)	PLL乘法	最大工作频率 fc (MHz)	A/DC 最大工作频率 (MHz)	时钟齿轮(CG) (注2) PLL = ON					时钟齿轮(CG) (注2) PLL = OFF				
					1/1	1/2	1/4	1/8	1/16	1/1	1/2	1/4	1/8	1/16
8	8	8	32	32	32	16	8	4	2	8	4	2	1	-
10	10		40	40	40	20	10	5	2.5	10	5	2.5	1.25	-
12	12		48	24 (注1)	48	24	12	6	3	12	6	3	1.5	-
-	48	-	48	24 (注1)	-	-	-	-	-	48	24	12	6	3

↑ 复位后的初始值

注 1: ADC最大工作频率 40 MHz，是ADxCLK<ADCLK.>寄存器规定的 2 分频fc / 2 频率。

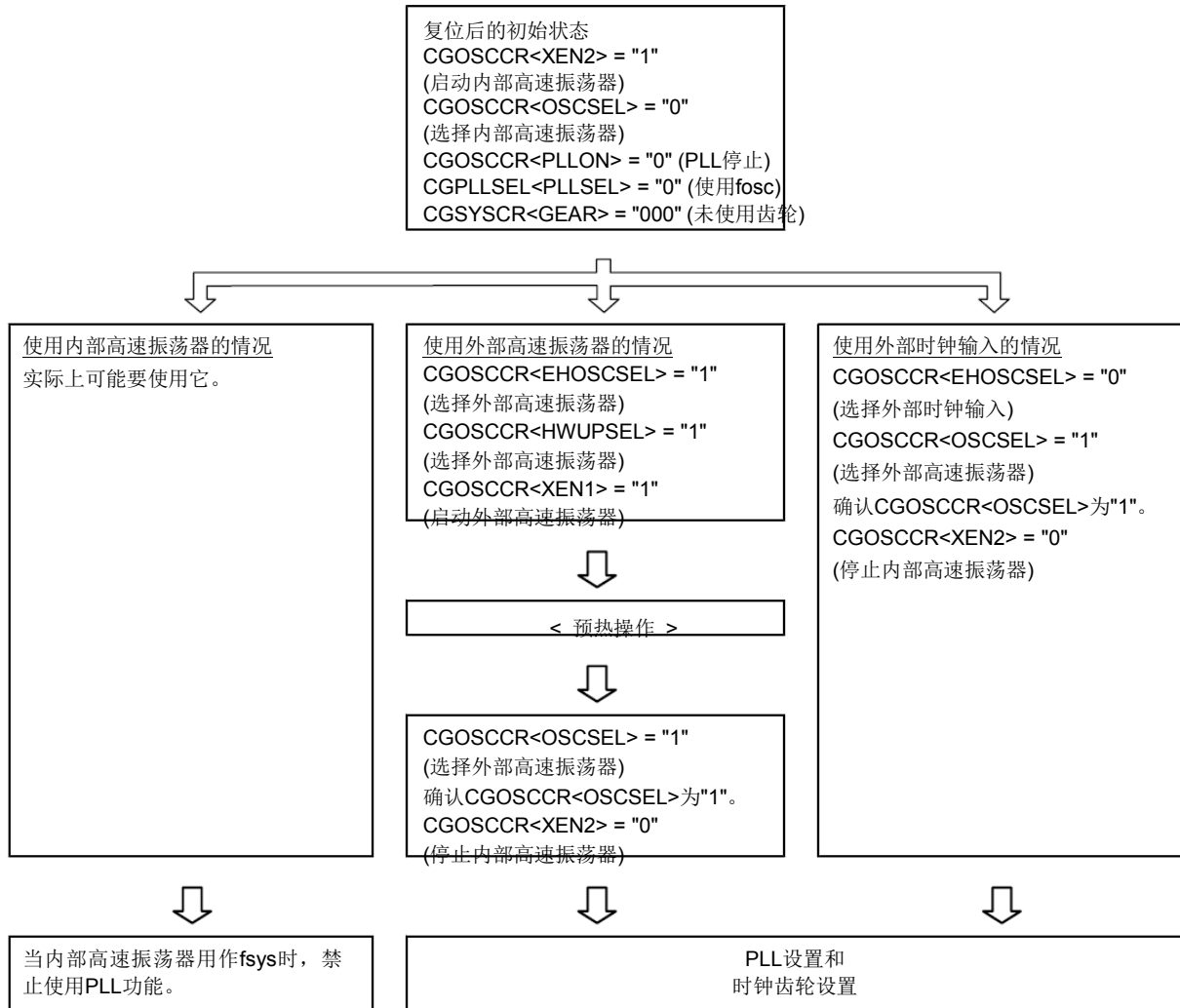
注 2: 当使用SysTick时，切勿使用 1 / 16 。

8.3.6.1 系统时钟设置

设置CGOSCCR，就能选择系统时钟。必要时，在选择时钟后，用CGPLLSEL和CGOSCCR进行PLL设置。并且用CGSYSCR设置时钟齿轮。

时钟设置顺序如下所示。

时钟设置顺序



8.3.7 预分频时钟控制

外设功能有一个对时钟分频的预分频器。作为要输入给各预分频器的 $\phi T0$ ，CGSYSCR<FPSEL>中规定的"fperiph"时钟能按照CGSYSCR<PRCK[2:0]>中的设置分频。在控制器复位之后，"fperiph/1"即被选作 $\phi T0$ 。

注 1：为了使用时钟齿轮，保证作出的时间设置能使各外设功能的预分频器输出 ϕTn 慢于fsys ($\phi Tn \leq fsys$)。在定时计数器或其它外设功能运行期间，不要切换该时钟齿轮。

只有当TBxCR<FTOSEL>为"1"时， ϕTn 才能小于或等于fsys。

注 2：在TMRB等外设功能运行前，切勿变更分频用的预分频器。

8.3.8 系统时钟引脚输出功能

TX03启用而从引脚输出系统时钟。PD7/SCOUT引脚能输出系统时钟fsys和fsys/2及外设I/O的预分频器输入时钟 $\phi T0$ 。

注 1：不保证SCOUT的系统时钟输出和内部时钟之间的相差(AC时序)。

注 2：当fsys从SCOUT引脚输出时，SCOUT引脚恰在变更时钟齿轮后输出意外波形。在意外波形影响系统的情况下，当变更时钟齿轮时，SCOUT引脚的输出应禁用。

端口用作SCOUT时的设置，见"输入/输出端口"。

当SCOUT引脚设置为SCOUT输出时，各模式下的引脚状态如表 8-3 所示。

表 8-3 各模式下的SCOUT输出状态

SCOUT的选择 CGSYSCR	模式	低功耗模式	
	NORMAL	IDLE	STOP1
<SCOSEL[1:0]> = " 00 "	保留		
<SCOSEL[1:0]> = " 01 "	输出fsys/2时钟		
<SCOSEL[1:0]> = " 10 "	输出fsys时钟		
<SCOSEL[1:0]> = " 11 "	输出 $\phi T0$ 时钟	固定为"0"或"1"。	

8.4 模式与模式推移

8.4.1 模式推移

IDLE和STOP1模式能用作低功耗模式，通过停止处理器内核而降低功耗。

模式转换图如图 8-2 所示。

有关"退出时睡眠"的详细资料，请参看"Cortex-M3 技术参考手册"。

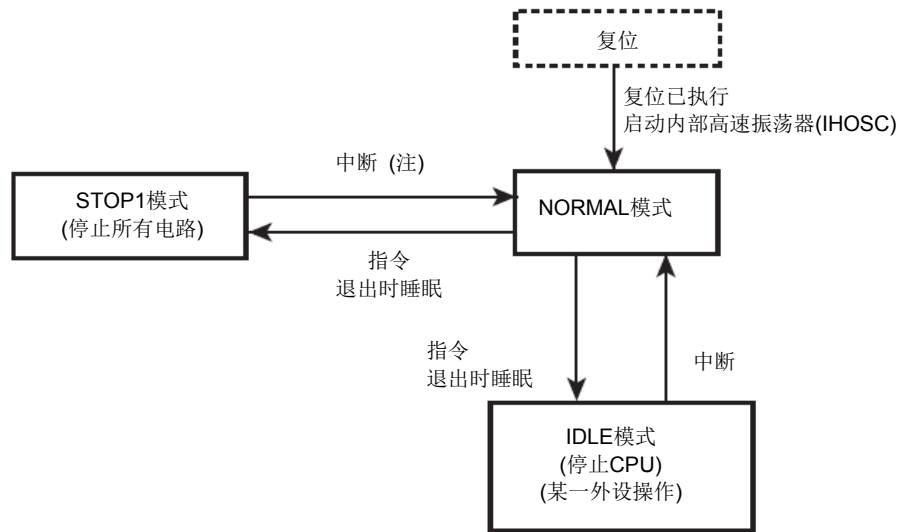


图 8-2 模式转换图

注：需进行预热。在变为STOP1 模式前，预热时间必须在NORMAL模式下设置。预热时间见"8.6.6 预热"。

8.5 运行模式

8.5.1 NORMAL模式

该模式可利用高速时钟运行该CPU内核与外设硬件。

在复位之后，其即被转入NORMAL模式。

8.6 低功耗模式

TX03有两种低功耗模式：IDLE和STOP1。为了切换到低功耗模式，应在系统控制寄存器CGSTBYCR<STBY[2:0]>中规定模式，并执行WFI(等待中断)指令。在这种情况下，执行复位或生成该中断即可解除该模式。以中断方式产生的释放要求提前设置。有关详细资料见"异常"一节。

注 1: TX03 不提供任何释放低功耗模式的事件。禁止通过执行WFE(等待事件)指令来切换到低功耗模式。

注 2: TX03 不支持在Cortex- M3 内核中配置SLEEPDEEP位的低功耗模式。禁止设置该系统控制寄存器的 <SLEEPDEEP> 位。

IDLE和STOP1 模式的特性如下。

8.6.1 IDLE模式

在该模式下，仅CPU被停止。各外设功能在其控制寄存器中有一位，用于在IDLE模式时启用或禁用操作。在进入IDLE模式时，在IDLE模式下其运行被禁止的各外设功能停止运行，并保持在当时的状态。

以下外设功能可在该IDLE模式下被启用或被禁用。有关设置的详细资料，见各外设功能相关章节。

- 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)
- 模拟数字转换器 (ADC)
- 看门狗定时器 (WDT)

注 1: 注意在IDLE模式时CPU不可能清除看门狗定时器功能的计数器。

注 2: 在切换到IDLE模式前，请停止USB时钟。(CGUSBCTL<USBCLKEN>="0")

8.6.2 STOP1 模式

当STOP1模式被释放时，内部振荡器启动，运行模式变为NORMAL模式。

通过设置CGSTBYCR<DRVE>，STOP1模式启用而选择引脚状态。在STOP1模式时引脚状态如表 8-4 所示。

注 1：当运行模式从STOP1模式变为NORMAL模式时，必须预热。当MCU返回时，必须预热。在STOP1模式前，预热时间必须在先前模式(NORMAL模式)中设置。预热时间详见"8.3.4 预热功能"。

注 2：由于内部高速振荡器启动，当从STOP1模式返回时，相关位CGPLLSEL<PLLSEL>，CGOSCCR<HWUPSEL>，<OS CSEL>，<XEN2>，<XEN1>和<PLLON> 被初始化，而CGOSCCR<WUODR[11:0]>不被初始化。

表 8-4 STOP1模式时的引脚状态

功能	引脚名称	I/O	STOP1	
			<DRVE> = "0"	<DRVE> = "1"
控制引脚	RESET, NMI, MODE, BSC	输入	o	o
振荡器	X1/EHCLKIN	输入	x	x
	X2	输出	"高"电平输出	
PORT	PI3, PI6, PI7 (TCK/SWCLK, TDI, TRST) (调试I/F设置, PxFRn<PxmFn>="1"的情况)	输入	取决于 PxIE[m]。	
	PI4 (TMS/SWDIO) (调试I/F设置, PxFRn<PxmFn>="1"的情况)	输入	取决于 PxIE[m]。	
	PI5, PI2, PI1, PI0, PH0, PH1 (TDO/SWV, TRACECLK, TRACEDATA0 ~ 3) (调试I/F设置, PxFRn<PxmFn>="1"的情况)	输出	取决于PxCR[m]。	
	PG3, PG5, PK0, PK1, PE7, PE3, PF4, PF5, PH4, PJ7 (INT0 ~ 9) (中断设置, PxFRn<PxmFn>="1"和 PxIE<PxmIE>="1"的情况)	输入	o	o
	若使用非上述引脚	输入	x	取决于 PxIE[m]。
		输出	x	取决于 PxCR[m]。

o: 有效输入或输出。

x: 无效输入或输出。

注: x: 端口号 / m: 对应位 / n: 功能寄存器号

8.6.3 低功耗模式设置

低功耗模式由CGSTBYCR<STBY[2:0]>的设置规定。

<STBY[2:0]>中的模式设置如表 8-5 所示。

表 8-5 低功耗模式设置

模式	CGSTBYCR <STBY[2:0]>
STOP1	001
IDLE	011

注：除上述<STBY[2:0]>中的各值外，不要设置其它任何值。

8.6.4 各模式下的操作状态

各模式下的操作状态如表 8-6 所示。

表 8-6 各模式下的操作状态

模块	NORMAL 内部高速振荡器 的使用 (IHOSC)	NORMAL 外部高速振荡器 的使用 (EHOSC)	IDLE 内部高速振荡器 的使用 (IHOSC)	IDLE 外部高速振荡器 的使用 (EHOSC)	STOP1(注 1)
处理器内核	o	o	-	-	-
DMAC	o	o	o	o	-
I/O端口	o	o	o	o	o(注 2)
SIO/UART	o	o	Δ	Δ	-
I2C/SIO	o	o	Δ	Δ	-
TMRB	o	o	Δ	Δ	-
WDT	o	o	Δ(注 4)	Δ(注 4)	-
USB	o	o	o	o	-
12-位ADC	o	o	Δ	Δ	-
CG	o	o	o	o	o
PLL	o	o	Δ	Δ	-
外部高速振荡器 (EHOSC)	Δ	o	Δ	o	-
内部高速振荡器 (IHOSC)	o	o(注 3)	o	o(注 3)	-
主RAM	o	o	o	o	o

o：在目标模式时，操作可用。

-：当切换到目标模式时，模块时钟自动停止。

Δ：在目标模式时，能通过软件选择启用或禁用模块操作。

注 1：在切换到STOP1模式时，停止“-”和“x”外设功能。通过停止AD转换器参考电压，就能降低泄漏电流。

注 2：状态取决于CGSTBYCR<DRVE>。

注 3：在复位或STOP1模式被释放后，时钟由内部高速振荡器提供。

注 4：注意在IDLE模式时CPU不可能清除看门狗定时器功能的计数器。

8.6.5 释放低功耗模式

可通过中断请求，非屏蔽中断 (NMI)或复位释放低功耗模式。可通过所选择的低功耗模式，确定可使用的释放源。

详见表 8-7。

表 8-7 各模式下的释放源

低功耗模式		IDLE	STOP1
释放源	中断	INT0 ~ 9 (注 1)	o
		INTTB0 ~ 9	o
		INTCAP00 ~ 91	o
		INTRX0 ~ 1, INTTX0 ~ 1	o
		INTSBI0 ~ 1	o
		INTUSB	o
		INTUSBWKUP	o
		INTAD / INTADHP / INTADM0 ~ 1	o
		INTDMAC0TC, INTDMAC0ERR	o
	SysTick中断	o	
不可屏蔽中断(INTWDT)	o		
不可屏蔽中断($\overline{\text{NMI}}$ 引脚)	o		
RESET ($\overline{\text{RESET}}$ 引脚)	o		

o: 在模式被释放后，启动中断处理。(复位使LSI初始化)

x: 不可用

注 1: 如拟转入该低功耗模式，可设置CPU以禁止除该释放源以外的所有中断。否则，未规定的中断会执行释放。

注 2: 当通过中断电平模式而释放IDLE，STOP1模式时，必须保持电平，直到中断处理启动。若电平在此之前改变，则正确的中断处理无法启动。

- 通过中断请求实现释放

为了通过中断来释放低功耗模式，必须事先设置CPU，以检测中断。除CPU中的设置外，还必须设置时钟产生器，以检测用于释放STOP1模式的中断。

- 通过不可屏蔽中断(NMI)实现释放

INTWDT仅可用于该IDLE模式。

- 通过复位实现释放

低功耗模式可通过 $\overline{\text{RESET}}$ 引脚的复位实现释放此后，模式切换到NORMAL模式，所有寄存器按正常复位的情况初始化。

注意经复位返回到STOP1模式不会引起自动预热。保持复位信号有效，直到振荡器的运行变得稳定。

- 通过SysTick实现释放

SysTick中断在IDLE模式时使用。

详见"中断"。

8.6.6 预热

在进行模式推移时，可能会要求进行预热，以使得内部振荡器可提供稳定振荡。

在从STOP1 到NORMAL的模式切换中，预热计数器和内部振荡器自动激活。然后在经过一段预热时间后，系统时钟输出启动。

为了进入STOP1模式，在执行指令前，必须在CGOSCCR<WUODR[11:0]>中设置预热时间。

注： 由于内部高速振荡器启动，当从STOP1模式返回时，相关位CGPLLSEL<PLLSEL>，CGOSCCR<HWUPSEL>，<OSCSEL>，<XEN2>，<XEN1>和<PLLON>被初始化，而CGOSCCR<WUODR[11:0]>不被初始化。

各模式切换是否需要预热设置如表 8-8 所示。

表 8-8 模式切换时的预热设置

模式推移	预热设置
NORMAL → IDLE	不需要
NORMAL → STOP1	不需要
IDLE → NORMAL	不需要
STOP1 → NORMAL	自动预热(注)

注：经复位返回到NORMAL模式不会引起自动预热。输入与冷复位相同的复位信号。

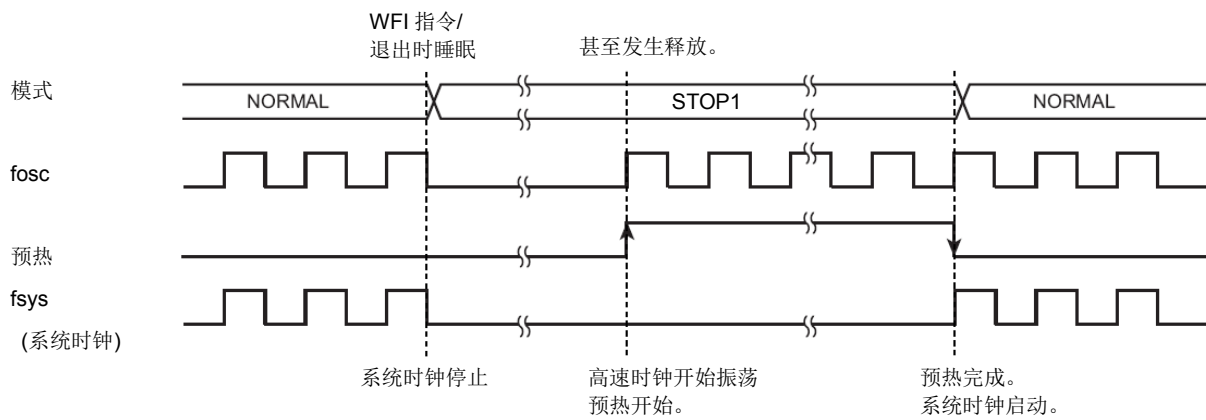
8.6.7 模式切换时的时钟操作

模式切换时的时钟操作如下。

8.6.7.1 操作模式切换: NORMAL → STOP1 → NORMAL

从STOP1模式返回到NORMAL模式时，预热自动激活。在进入STOP1模式前，必须将预热时间 0x119 设置在CGOSCCR<WUODR[11:0]>中，该时间是内部闪存的稳定时间。

在通过复位返回到NORMAL模式时，不会引发自动预热。输入与冷复位相同的复位信号。



9. 异常

本节对各异常的特征，类型与处理进行说明。

异常与CPU内核之间有着密切的关系。如有必要，请参看“Cortex-M3 技术参考手册”。

9.1 概述

异常造成CPU停止当前正执行的进程并处理另一个进程。

有两类异常：当某一错误条件发生时或者当产生异常的指令被执行时产生的异常；硬件产生的异常，例如外部引脚或外设功能的中断请求信号。

所有例外由CPU中的嵌套向量中断控制器(NVIC)按照各自优先等级进行处理。在发生异常时，CPU会将当前状态存储到该堆栈并转到相应的中断服务程序(ISR)。一旦该ISR完成，此前被存储该堆栈的信息即自动恢复。

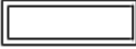
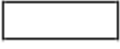
9.1.1 异常类型

Cortex-M3 存在以下异常类型。


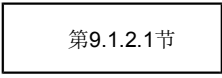




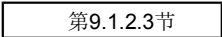
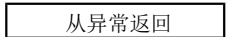
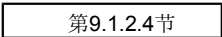
有关各异常的详细说明，请参看“Cortex-M3 技术参考手册”。

- 复位
- 不可屏蔽中断 (NMI)
- 硬故障
- 存储器管理
- 总线故障
- 使用故障
- SVCALL(管理程序调用)
- 调试监视器
- PendSV
- SysTick
- 外部中断

9.1.2 处理流程图

以下给出了异常/中断的处理方式。在下列说明中， 表示硬件处理。 表示软件处理。

各步骤将在本节后面的内容中进行说明。

处理	说明	参见
 由CG/CPU进行检测	CG/CPU检测到异常请求。	 第9.1.2.1节
↓		
 由CPU进行处理	CPU处理该异常请求。	 第9.1.2.2节
↓		
 转入ISR	CPU转入到相应的中断服务程序(ISR)。	
↓		
 ISR的执行	进行必要的处理。	 第9.1.2.3节
↓		
 从异常返回	CPU转入到另一ISR，或返回到前一程序。	 第9.1.2.4节

9.1.2.1 异常请求与检测

(1) 异常发生

异常的来源包括CPU的指令执行，存储器访问和外部中断引脚或外设功能的中断请求。

当CPU执行一个会造成异常的指令时或者当错误条件在指令执行时发生时，异常就会发生。

由于从不执行(XN)区域的指令取出或故障区域的访问破坏，异常也会发生。

中断请求产生于外部中断引脚或外设功能。关于用于释放待机模式的中断，相关设置必须在时钟发生器中进行。详见"9.5 中断"。

(2) 异常检测

如果同时发生多个异常，则 CPU 会取优先级最高的异常。

异常的优先级如表 9-1 所示。"可配置"指用户能给该异常分配一个优先级。存储器管理，总线故障和使用故障这些异常能被启用或禁用如果发生某被禁用异常，其将被作为硬故障接受处理。

表 9-1 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SYSRETRREQ
2	非屏蔽中断	-2	$\overline{\text{NMI}}$ 引脚或WDT
3	硬故障	-1	由于较高优先级故障正被处理或者它被禁用, 无法激活的故障
4	存储器管理	可配置	存储器保护单元(MPU)异常 (注 1) 从不执行(XN)区域的指令取出
5	总线故障	可配置	对存储器地址硬故障区域的访问破坏
6	使用故障	可配置	未定义的指令执行或其他与指令执行有关的故障
7~10	保留	-	
11	SVCcall	可配置	用SVC指令进行系统服务调用
12	调试监督程序	可配置	在CPU无故障时调试监督程序
13	保留	-	
14	PendSV	可配置	可挂起系统服务申请
15	SysTick	可配置	源自系统计时器的通知
16~	外部中断	可配置	外部中断引脚或外设功能 (注 2)

注 1: 本产品不含该MPU。

注 2: 各产品外部中断的源与数目各不相同。有关详细资料见"9.5.1.5 中断源列表"。

(3) 优先权设置

• 优先级

外部中断优先级可被设置为中断优先权寄存器，而其它异常会被设置到系统处理器优先寄存器中的 <PRI_n>位。

可改变配置 <PRI_n>，且优先级设置所需位的数目范围在 3 位到 8 位的范围内，具体范围视产品而定。因此，操作员可指定优先值的范围均视产品而定。

在 8 位配置的情况下，优先级配置范围 0 ~255。最高优先级为"0"。若存在具有相同优先级的多个要素，编号越小，优先级越高。

注: <PRI_n>位被定义为本产品的 3-位配置。

优先级分组

优先级组可被分为若干组。通过设置应用中中断和复位控制寄存器的<PRIGROUP>，<PRI_n>可分成占先式优先级和次级优先级。

优先级与先占优先级类似。若优先级与占先式优先级相同，则它与次级优先级比较。如果子优先级与优先级相同，则异常数目越小，优先级就越高。

优先级组的设置如表 9-2 所示。表中的占先式优先级和次级优先级是<PRI_n>被定义为 8 位配置情况下的编号。

表 9-2 优先级分组设置

<PRIGROUP[2:0]> 设置	<PRI_n[7:0]>		先占优先权的数目	子优先权的数目
	先占字段	子优先权字段		
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>的配置小于 8 位，则较低的位为"0"。例如，在 3-位配置的情况下，优先级设置为<PRI_n[7:5]>，<PRI_n[4:0]>为"00000"。

9.1.2.2 异常处理与转入该中断服务程序 (先占)

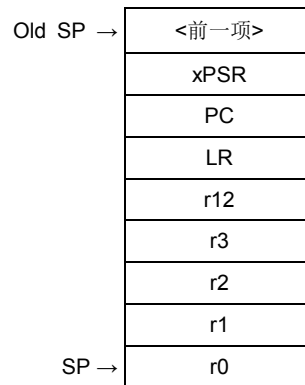
在异常发生时，CPU会中止当前正在执行的处理，并转到中断服务程序。这就是所谓的"先占"。

(1) 堆栈

在CPU检测到异常时，其会按以下顺序，将以下八个寄存器的内容压入到该堆栈：

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

到压栈完成时，SP渐减 8 个字。在寄存器内容已被压栈后，堆栈状态如下所示。



(2) 取出ISR

CPU启用指令以取出该中断处理，并将数据存储至寄存器。编制一个内含各异常ISR的前列地址的向量表。复位后，

向量表位于代码区中的地址 0x0000_0000。通过设置向量表偏移寄存器，

即可将向量表放置在代码或SRAM空间中的任何地址。

该向量表还应包含该主堆栈的初始值。

(3) 迟来

如果CPU在执行上一异常的ISR之前检测到优先级较高的异常，则CPU会首先处理该优先级较高的异常。这就是所谓的"迟来"。

迟后到达型异常可导致CPU取用新的向量地址，以转至相应的ISR，但CPU不会将寄存器内容推至存储栈。

(4) 向量表

该向量表的配置如以下所示。

用户必须始终设置前四个字(栈顶地址，复位ISR地址，NMI ISR地址，以及硬故障ISR地址)。如有必要，可设置其它异常的ISR地址。

偏移量	异常	内容	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	存储器管理	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	调试监督程序	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

9.1.2.3 执行ISR

ISR可执行相应异常的必要处理。ISR必须由用户编制。

ISR可能需包括中断请求清除用代码，以确保在恢复正常程序执行时不会再发生相同的中断。中断处理详见"9.5 中断"。

如果在当前异常的ISR执行期间发生优先级较高的异常，则CPU会放弃当前正在执行的ISR，并检修最近检测到的异常。

9.1.2.4 异常出口

(1) 从ISR返回后的执行

在从某ISR返回时，CPU会采取以下动作的其中之一：

- 尾链

如果存在一个挂起异常，且不存在任何堆栈异常，或挂起异常的优先级高于所有堆栈异常，则CPU会返回到挂起异常的ISR。

在这种情况下，CPU在退出一个ISR并进入另一个ISR时，跳过八个寄存器的堆栈上托和八个寄存器的推栈。这就是所谓的"尾链"。

- 返回到上一堆栈ISR

如果不存在任何挂起异常，或优先级最高的堆栈异常的优先级高于最高优先级挂起异常，则CPU会返回到上一堆栈ISR。

- 返回到前一程序

如果不存在待决或已堆栈异常，则CPU会返回前一程序。

(2) 异常出口顺序

在从某ISR返回时，CPU会执行以下操作：

- 弹出八个寄存器

从该堆栈弹出八个寄存器（PC，xPSR，r0～r3，r12与LR），并调节该SP。

- 加载当前激活的中断编号

从已堆栈xPSR加载当前的活动中断号。CPU会用其进行跟踪，并确定返回到哪个中断。

- 选择SP

如果返回到某异常(处理器模式)，则SP是SP_main。如果返回到线程模式，则SP可为SP_main或SP_process。

9.2 复位异常

复位异常产生于以下三个来源：

使用时钟发生器的该复位标志(CGRSTFLG)寄存器标识某个复位的源。

- 外部复位引脚
在某外部复位引脚从"低"变为"高"时，会发生复位异常。
- 由WDT导致的复位异常
看门狗定时器(WDT) 具备复位生成功能。有关详细资料见WDT相关节次。
- 由SYSRESETREQ导致的复位异常
通过在NVIC的应用中断和复位控制寄存器中设置SYSRESETREQ位，可生成复位。

9.3 非屏蔽中断 (NMI)

不可屏蔽中断产生于以下两个来源。

可利用时钟发生器的NMI标志(CGNMIFLG) 寄存器, 标识非屏蔽中断的源。

- 外部 $\overline{\text{NMI}}$ 引脚
当外部 $\overline{\text{NMI}}$ 引脚从"高"变为"低"时, 就会生成不可屏蔽中断。
- WDT导致的不可屏蔽中断
看门狗定时器(WDT)具备屏蔽中断生成功能。详见WDT章节。

9.4 SysTick

SysTick具备中断功能(采用CPU的系统记时器)。

用户在SysTick重新加载值寄存器中设置某个值, 并启用在SysTick控制和状态寄存器中的SysTick功能时, 计数器会加载在重新加载值寄存器中设置的值, 并开始倒数。在计数器达到"0"时, 即发生SysTick异常。可选用挂起异常, 并利用某个标志, 以掌握计时器何时达到"0"。

该SysTick校准值寄存器可保持某个重新加载值, 且保持的持续时间为10ms由系统记时器计时。计数时钟频率随产品而变化, 因此, 在SysTick校准值寄存器中的设置值也会随产品而变化。

注: 在本产品中, CGOSCCR <OSCSEL> <EHOSCSEL>选择的fosc乘以 32 用作外部参考时钟。

9.5 中断

本节对中断的路径，源与必要设置进行说明。

源自各中断源的中断信号可将中断请求通知CPU。

其可设置各中断的优先级，并处理具备最高优先级的中断请求。

可通过时钟发生器，将待机模式清除用中断请求通知CPU。因此，必须在时钟发生器中进行适当的设置。

9.5.1 中断源

9.5.1.1 中断路径

中断请求路径如图 9-1 所示。

并非用于释放待机的外设功能所发布的中断，即被直接输入到CPU(路径 1)。

用于释放待机的外设功能(路径 2)，以及源自外部中断引脚(路径 3)的中断被输入到时钟发生器，并通过待机释放逻辑(路径 4 与 5)被输入到CPU。

如果源自外部中断引脚的中断未被用于释放待机，则其会被直接输入到CPU(不通过待机释放逻辑)(路径 6)。

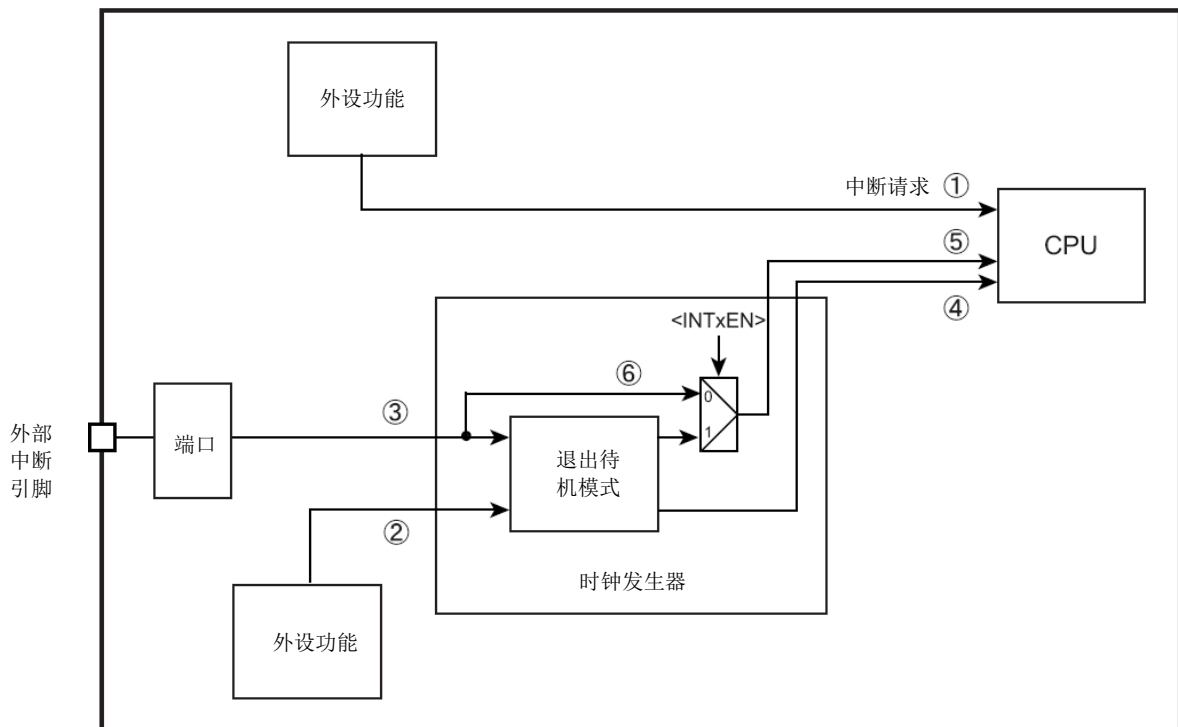


图 9-1 中断路径

9.5.1.2 生成

可从被指定为中断源的某外部引脚或外设功能，或通过设置NVIC的中断设置待决寄存器，生成一个中断请求。

- 从外部引脚
设置该端口控制寄存，使得该外部引脚可用作中断功能引脚。
- 从外设功能
设置该外设功能，使之可输出中断请求。有关详细资料见各外设功能相关章节。
- 通过设置中断设置挂起寄存器(强制挂起)
通过设置中断设置挂起寄存器的相关位，可生成一个中断请求。

9.5.1.3 传输

可将源自某外部引脚或外设功能的某中断信号直接发送给CPU，但其被用于退出待机模式的情形除外。

源自可用于待机模式清除的中断源的中断请求，会通过时钟发生器被传输到CPU。对于这些中断源，必须预先在时钟发生器中进行适当的设置。对于未被用于退出待机模式的外部中断源，无需设置时钟发生器即可使用。

9.5.1.4 使用外部中断引脚时的预防措施

如果操作员使用外部中断，则应注意以下内容，以免生成意外中断。

如果输入被禁用($PxIE < PxmIE = 0$)，则源自外部中断引脚的输入为"高"。如果外部中断未被用作待机释放触发器("图 9-1 中断路径"的路径 6)，则来自各外部中断引脚的输入信号就会而被直接发送至CPU。由于CPU会将"高"输入识别为中断，因此，如果CPU在输入被禁用时启用相应的中断，就会发生中断。

将中断引脚输入设置为"低"并启用，即可在未将外部中断设置为待机触发信号的情况下使用该外部中断。即可启用CPU中断。

9.5.1.5 中断源列表

表 9-3 所示为中断源的列表。

表 9-3 中断源列表

编号	中断源		有效电平 (释放等待和中断)	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	INTRX0	串行接收 (通道 0)		
9	INTTX0	串行传输 (通道 0)		
10	INTRX1	串行接收 (通道 1)		
11	INTTX1	串行传输 (通道 1)		
12	INTUSBWUP	USB唤醒中断	下降缘	CGIMCGC
13	-	保留		
14	INTSB0	串行总线接口 0		
15	INTSB1	串行总线接口 1		
16	INTADHP	最高优先级AD转换完成中断		
17	INTAD	AD转换完成中断		
18	INTADM0	AD转换监控功能中断 0		
19	INTADM1	AD转换监控功能中断 1		
20	INTTB0	TMRB0 匹配检测		
21	INTTB1	TMRB1 匹配检测		
22	INTTB2	TMRB2 匹配检测		
23	INTTB3	TMRB3 匹配检测		
24	INTTB4	TMRB4 匹配检测		
25	INTTB5	TMRB5 匹配检测		
26	INTTB6	TMRB6 匹配检测		
27	INTTB7	TMRB7 匹配检测		
28	INTTB8	TMRB8 匹配检测		
29	INTTB9	TMRB9 匹配检测		
30	INTUSB	USB中断		
31	-	保留		
32	-	保留		
33	-	保留		
34	INTUSBPON	USB通电连接检测中断(Vbus检测)	可选择	CGIMCGC
35	-	保留		
36	INTCAP00	TMRB0 输入捕捉 0		
37	INTCAP01	TMRB0 输入捕捉 1		
38	INTCAP10	TMRB1 输入捕捉 0		
39	INTCAP11	TMRB1 输入捕捉 1		
40	INTCAP20	TMRB2 输入捕捉 0		
41	INTCAP21	TMRB2 输入捕捉 1		

表 9-3 中断源列表

编号	中断源		有效电平 (释放等待和中断)	CG中断模式 控制寄存器
42	INTCAP30	TMRB3 输入捕捉 0		
43	INTCAP31	TMRB3 输入捕捉 1		
44	INTCAP40	TMRB4 输入捕捉 0		
45	INTCAP41	TMRB4 输入捕捉 1		
46	INTCAP50	TMRB5 输入捕捉 0		
47	INTCAP51	TMRB5 输入捕捉 1		
48	INTCAP60	TMRB6 输入捕捉 0		
49	INTCAP61	TMRB6 输入捕捉 1		
50	INTCAP70	TMRB7 输入捕捉 0		
51	INTCAP71	TMRB7 输入捕捉 1		
52	INTCAP80	TMRB8 输入捕捉 0		
53	INTCAP81	TMRB8 输入捕捉 1		
54	INTCAP90	TMRB9 输入捕捉 0		
55	INTCAP91	TMRB9 输入捕捉 1		
56	INT8	中断引脚 8		
57	INT9	中断引脚 9		
58	-	保留		
59	-	保留		
60	INTDMAC0TC	DMA 端子计数状态中断 0		
61	INTABTLOSS0	I2C 判优丢失中断 0		
62	INTDMAC0ERR	DMA 错误状态中断 0		
63	INTABTLOSS1	I2C 判优丢失中断 1		

9.5.1.6 有效电平

有效电平表明中断源信号的哪种改变会导致中断。CPU会再次组织"高"电平中的中断信号，将其作为中断信号系统将对从周围函数发送至CPU的中断信号配置到"高"输出，表示一次中断请求。

有效电平可被设置为中断时钟发生器，其可被用作待机释放用触发信号。源自周围函数的中断请求设置为上升缘或下降缘。源自中断引脚的中断请求可被设置为电平敏感型("高"或"低")或边缘触发型(上升或下降)。


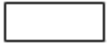
如果某中断源被用于清除待机模式，则必须设置相关时钟发生器寄存器。启用CGIMCGx<INTxEN>位，并规定CGIMCGx<EMCGx>位中的有效电平。必须为各外设功能发出的中断请求设置有效电平(如图 9-3 所示)。


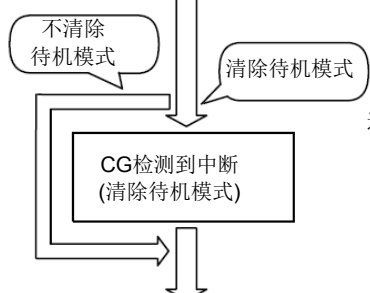


可用一个"高"电平信号将时钟发生器所检测到的中断请求通知CPU。

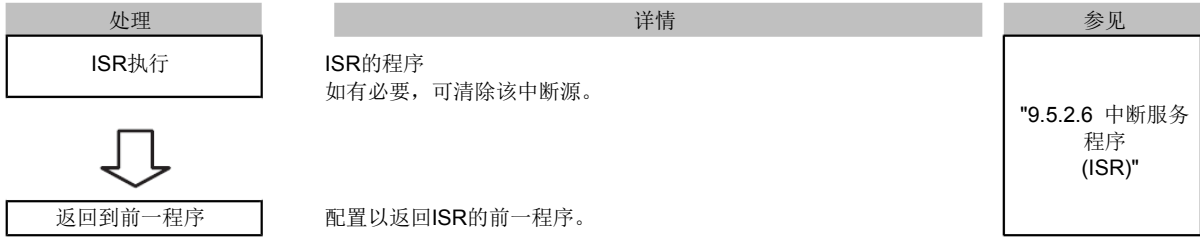
9.5.2 中断处理

9.5.2.1 流程图

下文给出了中断的处理方式。

在以下说明中，表示硬件处理。表示软件处理。

处理	详情	参见
<div style="border: 1px solid black; padding: 5px; margin-bottom: 5px; text-align: center;">检测设置</div> <div style="border: 1px solid black; padding: 5px; text-align: center;">发送中断信号的设置</div>	<p>为检测中断设置相关NVIC寄存器。 若采用每个中断源来清除等待模式，则还应设置时钟发生器。</p> <p>o通用设置 NVIC寄存器 o进行设置，以清除等待模式。</p> <p>时钟发生器 执行适当的设置，以根据中断类型发送中断信号。</p> <p>o进行设置，以便从外部引脚中断端口 o外设功能的中断设置 外设功能(有关详细资料见各外设功能相关节次)。</p>	<div style="border: 1px solid black; padding: 5px;">"9.5.2.2 准备工作"</div>
 <div style="border: 1px dashed black; padding: 2px; text-align: center;">中断生成</div>	<p>生成中断请求。</p>	
	<p>通过时钟发生器，将待机模式清除用中断线连接到CPU。</p>	<div style="border: 1px solid black; padding: 5px;">"9.5.2.3 由时钟发生器进行检测"</div>
<div style="border: 3px double black; padding: 5px; text-align: center;">CPU 检测到中断</div>	<p>CPU检测到该中断。</p> <p>如果多个中断请求同时发生，则会按照优先次序检测到优先级最高的中断请求。</p>	<div style="border: 1px solid black; padding: 5px;">"9.5.2.4 由CPU进行检测"</div>
 <div style="border: 3px double black; padding: 5px; text-align: center;">CPU 处理中断</div>	<p>CPU处理该中断。</p> <p>在进入该ISR之前，CPU会将寄存器内容压入该堆栈。</p>	<div style="border: 1px solid black; padding: 5px;">"9.5.2.5 CPU处理"</div>
		



9.5.2.2 准备工作

准备进行中断操作时, 需要注意配置的顺序, 以免发生意外的中断情况。

中断的执行及其配置的更改, 总体上必须按照以下顺序进行。通过CPU禁用该中断。从CPU按最远的路径进行配置。然后通过CPU启用中断。

要对时钟发生器进行配置, 必须依照本节所述的顺序, 以免发生意外的中断。首先设置该前置条件。其次, 清除时钟发生器中与该中断相关的数据, 然后启用该中断。

以下各节均按中断处理的顺序列出, 并将对其配制方法进行说明。

1. 通过CPU禁用中断
2. CPU寄存器设置
3. 预配置(1) (源自外部引脚的中断)
4. 预配置(2) (源自外设功能的中断)
5. 预配置(3) (中断设置挂起寄存器)
6. 配置该时钟发生器
7. 通过CPU启用中断

(1) 通过CPU禁用中断

为了使CPU不接受任何中断, 应向PRI·屏蔽寄存器写入"1"。所有中断与异常非可屏蔽中断与硬故障除外均会被屏蔽。

用"MSR"指令设置该寄存器。

中断屏蔽寄存器	
PRIMASK	← "1" (中断停用)

注 1: 用户存取级不能修改PRIMASK寄存器。

注 2: 如果在"1"被设置到PRIMASK寄存器导致发生故障, 则其会被作为硬故障处理。

(2) CPU寄存器设置

通过写入到NVIC寄存器的中断优先权寄存器中的<PRI_n>字段，操作员可指定某个优先级。

每个中断源配有 8 个位，用于分配优先等级(从0 ~ 255)，但每种产品实际使用的位数会有差别。0 级是最优先的等级。若多个中断源的优先级相同，则最小编号的中断源的优先级最高。

可通过在PRIGROUP域中的应用中断和复位控制寄存器来分配组优先级。

NVIC寄存器		
<PRI_n>	←	优先级
<PRIGROUP>	←	"组优先级" (如有必要可配置)

注: "n" 指相应的异常/中断。

本产品有三个位用于指定优先级。

(3) 预配置(1) (源自外部引脚的中断)

将"1"设置到相应引脚的端口功能函数寄存器。通过设置PxFRn[m]，即可允许将该引脚用作功能引脚。通过设置PxIE[m]，即可允许将该引脚用作输入端口。

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

注: x: 端口号 / m: 对应位 / n: 功能寄存器编号

在STOP模式以外的模式中，"设置PxIE(以启用输入)"可启用相应的中断输入无论PxFR设置如何。注意不得启用未使用的中断。此外，使用外部中断引脚时，应了解"9.5.1.4 预防措施"中的说明。

(4) 预配置(2) (源自外设功能的中断)

该设置随所用外设功能的不同而不同。有关外设功能的详细信息，参见各外设功能对应的章节。

(5) 预配置(3) (中断设置挂起寄存器)

在利用该中断设置挂起寄存器生成中断时，需将"1"设置到该寄存器的对应位。

NVIC寄存器		
中断设置-挂起[m]	←	"1"

注: m: 对应位



(6) 配置该时钟发生器

对于拟用于待机模式退出的中断源，操作员需在时钟发生器的CGIMCG寄存器中设置有效电平，并启用中断。CGIMCG寄存器能够对每个中断源进行配置。

在启用某中断之前，清除已保留的相应中断请求。这种处理可防止意外中断。为清除相应的中断请求，应向CGICRCG寄存器中写入所用中断对应的数值。各数值的具体信息，参见"9.6.3.4 CGICRCG(CG中断请求清除寄存器)"。

在未设置时钟发生器的情况下即可使用源自外部引脚的中断请求，但前提是其未被用于待机模式的退出。但必须输入"高"脉冲信号或"高电平"信号，以便CPU检测出该信号为中断请求。使用外部中断引脚时，还应注意"9.5.1.4 预防措施"一节中的说明。

时钟发生器寄存器		
CGIMCGn<EMCGm>	←	有效电平
CGICRCG<ICRCG>	←	拟使用中断的对应值
CGIMCGn<INTmEN>	←	"1" (中断启用)

注：n: 寄存器编号 / m:中断源分配的编号

(7) 通过CPU启用中断

通过CPU启用中断，如以下所述。

清除中断清除挂起寄存器中的已暂停中断。用该中断设置启动寄存器启用预定的中断。将寄存器的各位指定到单个中断源。

通过将"1"写入到该中断清除挂起寄存器的对应位，即可清除该已暂停的中断。若在"中断设置-启用寄存器"的对应位中写入"1"，则可启用相应的中断。

在该中断设置挂起寄存器设定中生成中断时，如果清除了各挂起中断，则中断触发用因数即告丢失。因此，不必进行该操作。

最后，PRIMASK寄存器被清零。

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

注 1: m: 对应位

注 2: 用户存取级不能修改PRIMASK寄存器。

9.5.2.3 由时钟发生器进行检测

如果用中断源退出待机模式，则会按照时钟发生器中所规定的有效电平进行检测，并通知CPU。

一旦检测到边触发中断请求，该中断请求即被保留在该时钟发生器中。对于电平敏感型中断请求，在检测到该请求前，必须将其保持在有效电平，否则，当信号电平由有效变为无效时，中断请求将不存在。

当时钟发生器检测到中断请求时，会一直向CPU发生“高”电平的中断信号，直到CG中断请求(CGICRCG)寄存器中已将中断请求信号清除。如果在未清除该中断请求的情况下退出待机模式，则在恢复正常运行时会再次检测到同一中断。务必清除ISR中的各中断请求。

9.5.2.4 由CPU进行检测

CPU可检测到具备最高优先级的中断请求。

9.5.2.5 CPU处理

一旦检测到中断，CPU就会在进入该ISR时将PC，PSR，r0-r3，r12和LR的内容压入到该堆栈中。

9.5.2.6 中断服务程序(ISR)

ISR需按照拟使用的应用进行特定的编程。本节将对服务程序编程时的建议项目，以及源清除方式进行说明。

(1) 在ISR运行期间进行压入

ISR一般会按要求将寄存器内容压入到堆栈，并处理某一中断。Cortex-M3核会自动将PC，PSR，r0-r3，r12和LR内容压倒堆栈中。无需对其进行额外编程。

如果需要，还可压入其它寄存器的内容。

即使正在执行某ISR，也会接收优先级较高的中断请求，以及NMI等异常。我们建议操作员压入可能会被重写的通用寄存器。

(2) 清除某个中断源

如拟用某中断源清除某待机模式，则必须用该CG中断请求清除(CGICRCG)寄存器清除各中断请求。

如某中断源被设置为电平敏感型，则其会继续存在，直至其在某源时被清除。因此，必须清除该中断源。中断源的清除，会自动清除从时钟发生器发出的中断请求信号。

若某中断信号设置为边缘敏感型，可设定CGICRCG请求中的相应数值，从而清除中断请求。再次出现有效电平时，可检测到新的中断请求。

9.6 例外情况/与中断相关的寄存器

本章所述的CPU寄存器，NVIC寄存器和时钟发生寄存器如下所示并附有相应的地址。

9.6.1 寄存器列表

NVIC寄存器

基址= 0xE000_E000

寄存器名称	地址
SysTick控制器与状态寄存器	0x0010
SysTick重新加载值寄存器	0x0014
SysTick当前值寄存器	0x0018
SysTick校准值寄存器	0x001C
中断设置-启用寄存器 1	0x0100
中断设置-启用寄存器 2	0x0104
保留	0x0108 ~ 0x017F
中断清除-启用寄存器 1	0x0180
中断清除-启用寄存器 2	0x0184
保留	0x0188 ~ 0x01FF
中断设置-挂起寄存器 1	0x0200
中断设置-挂起寄存器 2	0x0204
保留	0x0208 ~ 0x027F
中断清除-挂起寄存器 1	0x0280
中断清除-挂起寄存器 2	0x0284
保留	0x0288 ~ 0x03FF
中断优先权寄存器	0x0400 ~ 0x0460
保留	0x0464 ~ 0x0D07
矢量表位移寄存器	0x0D08
应用中断与复位控制寄存器	0x0D0C
系统处理器优先寄存器	0x0D18, 0x0D1C, 0x0D20
系统处理器控制器与状态寄存器	0x0D24

时钟发生器寄存器

基址 = 0x400F_3000

寄存器名称	地址
CG中断模式控制寄存器A	CGIMCGA 0x0040
CG中断模式控制寄存器B	CGIMCGB 0x0044
CG中断模式控制寄存器C	CGIMCGC 0x0048
保留	- 0x004C ~ 0x005F
CG中断请求清除寄存器	CGICRCG 0x0060
复位标志寄存器	CGRSTFLG 0x0064
NMI标志寄存器	CGNMIFLG 0x0068

9.6.2 NVIC寄存器

9.6.2.1 SysTick控制器与状态寄存器

	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: 计时器不会计数至 0 1: 计时器计数至 0 如果在上次读取后, 计时器计数至"0", 则返回"1"。 清除SysTick控制器与状态寄存器的任何已读取部分。
15-3	-	R	读作 0。
2	CLKSOURCE	R/W	0: 外部基准时钟(fosc/32) 1: CPU时钟(fsys)
1	TICKINT	R/W	0: 不要让SysTick挂起 1: 让SysTick挂起
0	ENABLE	R/W	0: 禁用 1: 启用 如果已设置"1", 则其会重新加载该重新加载值寄存器的值, 并开始运行。

注: 在本产品中, 外部基准时钟采用fosc, 可通过CGOSCCR <OSCSEL> <EHOSCSEL> 用 32选择fosc。

9.6.2.2 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	重新加载值 设置该值，使之在计时器达到"0"时加载到SysTick当前值寄存器。

9.6.2.3 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	[读取] 当前SysTick计时器值 [写入] 清除 通过将任意值写入到该寄存器，均可将其清0。 在清除该寄存器的同时，也会清除SysTick控制器与状态寄存器的<COUNTFLAG>位。

9.6.2.4 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: 具备基准时钟 1: 无基准时钟
30	SKEW	R	0: 校准值是 10 ms。 1: 校准值并非 10 ms。
29-24	-	R	读作 0。
23-0	TENMS	R	校准值 重新加载值将采用外部基准时钟(见注)发出的定时值(0x9C4)。

注：如果是重合多次，则请使用<TENMS>-1。

9.6.2.5 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 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 15)	SETENA (中断 14)	-	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	-	R	读作"0"。
30-14	SETENA	R/W	中断编号 [30:14] [写入] 1: 启用 [读取] 0: 禁用 1: 启用 每个位均与所指定的中断号对应。 通过将"1"写入到该寄存器中的某个位, 即可启用相应的中断。写入"0"无任何影响。通过读取各位, 即可查看相应中断的启用/禁用条件。
13	-	R	读作"0"。
12-0	SETENA	R/W	中断编号 [12:0] [写入] 1: 启用 [读取] 0: 禁用 1: 启用 每个位均与所指定的中断号对应。 通过将"1"写入到该寄存器中的某个位, 即可启用相应的中断。写入"0"无任何影响。通过读取各位, 即可查看相应中断的启用/禁用条件。

注: 有关中断和中断编号的说明, 参见 9.5.1.5 节"中断源列表"。

9.6.2.6 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 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 34)	-	-
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-28	SETENA	R/W	中断编号 [63:60] [写入] 1: 启用 [读取] 0: 禁用 1: 启用 每个位均与所指定的中断号对应。 通过将"1"写入到该寄存器中的某个位, 即可启用相应的中断。写入"0"无任何影响。通过读取各位, 即可查看相应中断的启用/禁用条件。
27-26	-	R	读作 0。
25-4	SETENA	R/W	中断编号 [57:36] [写入] 1: 启用 [读取] 0: 禁用 1: 启用 每个位均与所指定的中断号对应。 通过将"1"写入到该寄存器中的某个位, 即可启用相应的中断。写入"0"无任何影响。通过读取各位, 即可查看相应中断的启用/禁用条件。
3	-	R	读作"0"。
2	SETENA	R/W	中断编号 [34] [写入] 1: 启用 [读取] 0: 禁用 1: 启用 每个位均与所指定的中断号对应。 通过将"1"写入到该寄存器中的某个位, 即可启用相应的中断。写入"0"无任何影响。通过读取各位, 即可查看相应中断的启用/禁用条件。
1-0	-	R	读作 0。

注: 有关中断和中断编号的说明, 参见 9.5.1.5 节"中断源列表"。

9.6.2.7 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 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 15)	CLRENA (中断 14)	-	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	-	R	读作 0。
30-14	CLRENA	R/W	<p>中断编号 [30:14]</p> <p>[写入] 1: 禁用</p> <p>[读取] 0: 禁用 1: 启用</p> <p>每个位均与所指定的中断号对应。其可用于启用中断，以及检测各中断是否被禁用。通过将"1"写入到该寄存器中的某个位，即可启用相应的中断。写入"0"无任何影响。通过读取各位，即可查看相应中断的启用/禁用条件。</p>
13	-	R	读作 0。
12-0	CLRENA	R/W	<p>中断编号 [12:0]</p> <p>[写入] 1: 禁用</p> <p>[读取] 0: 禁用 1: 启用</p> <p>每个位均与所指定的中断号对应。其可用于启用中断，以及检测各中断是否被禁用。通过将"1"写入到该寄存器中的某个位，即可启用相应的中断。写入"0"无任何影响。通过读取各位，即可查看相应中断的启用/禁用条件。</p>

注：有关中断和中断编号的说明，参见 9.5.1.5 节"中断源列表"。

9.6.2.8 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 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 34)	-	-
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-28	CLRENA	R/W	<p>中断编号 [63:60]</p> <p>[写入] 1: 禁用</p> <p>[读取] 0: 禁用 1: 启用</p> <p>每个位均与所指定的中断号对应。其可用于启用中断，以及检测各中断是否被禁用。通过将"1"写入到该寄存器中的某个位，即可启用相应的中断。写入"0"无任何影响。通过读取各位，即可查看相应中断的启用/禁用条件。</p>
27-26	-	R	读作 0。
25-4	CLRENA	R/W	<p>中断编号 [57:36]</p> <p>[写入] 1: 禁用</p> <p>[读取] 0: 禁用 1: 启用</p> <p>每个位均与所指定的中断号对应。其可用于启用中断，以及检测各中断是否被禁用。通过将"1"写入到该寄存器中的某个位，即可启用相应的中断。写入"0"无任何影响。通过读取各位，即可查看相应中断的启用/禁用条件。</p>
3	-	R	读作 0。
2	CLRENA	R/W	<p>中断编号 [34]</p> <p>[写入] 1: 禁用</p> <p>[读取] 0: 禁用 1: 启用</p> <p>每个位均与所指定的中断号对应。其可用于启用中断，以及检测各中断是否被禁用。通过将"1"写入到该寄存器中的某个位，即可启用相应的中断。写入"0"无任何影响。通过读取各位，即可查看相应中断的启用/禁用条件。</p>
1-0	-	R	读作 0。

注：有关中断和中断编号的说明，参见 9.5.1.5 节“中断源列表”。

9.6.2.9 中断设置-挂起寄存器 1

	31	30	29	28	27	26	25	24
比特符号	-	SETPEND (断开 30)	SETPEND (断开 29)	SETPEND (断开 28)	SETPEND (断开 27)	SETPEND (断开 26)	SETPEND (断开 25)	SETPEND (断开 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 (断开 23)	SETPEND (断开 22)	SETPEND (断开 21)	SETPEND (断开 20)	SETPEND (断开 19)	SETPEND (断开 18)	SETPEND (断开 17)	SETPEND (断开 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 (断开 15)	SETPEND (断开 14)	-	SETPEND (断开 12)	SETPEND (断开 11)	SETPEND (断开 10)	SETPEND (断开 9)	SETPEND (断开 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 (断开 7)	SETPEND (断开 6)	SETPEND (断开 5)	SETPEND (断开 4)	SETPEND (断开 3)	SETPEND (断开 2)	SETPEND (断开 1)	SETPEND (断开 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	-	R	读作 0。
30-14	SETPEND	R/W	<p>中断编号 [30:14]</p> <p>[写入] 1: 挂起</p> <p>[读取] 0: 未挂起 1: 挂起</p> <p>各位对应于指定的编号, 强迫各中断进入挂起状态, 并可确定当前处于挂起状态的中断。通过将"1"写入到该寄存器中的某个位, 即可使相应的中断进入挂起状态。不过, 对于已处于挂起状态或已被禁用的中断而言, 写入"1"对其无任何影响。写入"0"无任何影响。通过读取该位, 即可返回相应中断的当前状态。通过将"1"写入到中断清除挂起寄存器中的某个对应位, 即可清除该寄存器中的该位。</p>
13	-	R	读作 0。
12-0	SETPEND	R/W	<p>中断编号 [12:0]</p> <p>[写入] 1: 挂起</p> <p>[读取] 0: 未挂起 1: 挂起</p> <p>各位对应于指定的编号, 强迫各中断进入挂起状态, 并可确定当前处于挂起状态的中断。通过将"1"写入到该寄存器中的某个位, 即可使相应的中断进入挂起状态。不过, 对于已处于挂起状态或已被禁用的中断而言, 写入"1"对其无任何影响。写入"0"无任何影响。通过读取该位, 即可返回相应中断的当前状态。通过将"1"写入到中断清除挂起寄存器中的某个对应位, 即可清除该寄存器中的该位。</p>

注: 有关中断和中断编号的说明, 参见 9.5.1.5 节"中断源列表"。

9.6.2.10 中断设置-挂起寄存器 2

	31	30	29	28	27	26	25	24
比特符号	SETPEND (断开 63)	SETPEND (断开 62)	SETPEND (断开 61)	SETPEND (断开 60)	-	-	SETPEND (断开 57)	SETPEND (断开 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 (断开 55)	SETPEND (断开 54)	SETPEND (断开 53)	SETPEND (断开 52)	SETPEND (断开 51)	SETPEND (断开 50)	SETPEND (断开 49)	SETPEND (断开 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 (断开 47)	SETPEND (断开 46)	SETPEND (断开 45)	SETPEND (断开 44)	SETPEND (断开 43)	SETPEND (断开 42)	SETPEND (断开 41)	SETPEND (断开 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 (断开 39)	SETPEND (断开 38)	SETPEND (断开 37)	SETPEND (断开 36)	-	SETPEND (断开 34)	-	-
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
31-28	SETPEND	R/W	<p>中断编号 [63:60] [写入] 1: 挂起 [读取] 0: 未挂起 1: 挂起</p> <p>各位对应于指定的编号, 强迫各中断进入挂起状态, 并可确定当前处于挂起状态的中断。 通过将"1"写入到该寄存器中的某个位, 即可使相应的中断进入挂起状态。不过, 对于已处于挂起状态或已被禁用的中断而言, 写入"1"对其无任何影响。写入"0"无任何影响。 通过读取该位, 即可返回相应中断的当前状态。 通过向中断清除-挂起寄存器中的对应位写入"1"的方式清除和中断设置-挂起寄存器位。</p>
27-26	-	R	读作 0。
25-4	SETPEND	R/W	<p>中断编号 [57:36] [写入] 1: 挂起 [读取] 0: 未挂起 1: 挂起</p> <p>各位对应于指定的编号, 强迫各中断进入挂起状态, 并可确定当前处于挂起状态的中断。 通过将"1"写入到该寄存器中的某个位, 即可使相应的中断进入挂起状态。不过, 对于已处于挂起状态或已被禁用的中断而言, 写入"1"对其无任何影响。写入"0"无任何影响。 通过读取该位, 即可返回相应中断的当前状态。 通过向中断清除-挂起寄存器中的对应位写入"1"的方式清除和中断设置-挂起寄存器位。</p>
3	-	R	读作 0。
2	SETPEND	R/W	<p>中断编号 [34] [写入] 1: 挂起 [读取] 0: 未挂起 1: 挂起</p> <p>各位对应于指定的编号, 强迫各中断进入挂起状态, 并可确定当前处于挂起状态的中断。 通过将"1"写入到该寄存器中的某个位, 即可使相应的中断进入挂起状态。不过, 对于已处于挂起状态或已被禁用的中断而言, 写入"1"对其无任何影响。写入"0"无任何影响。 通过读取该位, 即可返回相应中断的当前状态。 通过向中断清除-挂起寄存器中的对应位写入"1"的方式清除和中断设置-挂起寄存器位。</p>
1-0	SETPEND	R/W	读作 0。

注: 有关中断和中断编号的说明, 参见 9.5.1.5 节"中断源列表"。

9.6.2.11 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 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 15)	CLRPEND (中断 14)	-	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	-	R	读作 0。
30-14	CLRPEND	R/W	<p>中断编号 [30:14]</p> <p>[写入]</p> <p>1: 清除挂起中断</p> <p>[读取]</p> <p>0: 未挂起</p> <p>1: 挂起</p> <p>各位对应于指定的编号, 强迫各中断进入挂起状态, 并可确定当前处于挂起状态的中断。通过将"1"写入到该寄存器中的某个位, 即可清除相应的挂起中断。但写入"1"不会影响已使用的中断信号。写入"0"无任何影响。通过读取该位, 即可返回相应中断的当前状态。</p>
13	-	R	读作 0。
12-0	CLRPEND	R/W	<p>中断编号 [12:0]</p> <p>[写入]</p> <p>1: 清除挂起中断</p> <p>[读取]</p> <p>0: 未挂起</p> <p>1: 挂起</p> <p>各位对应于指定的编号, 强迫各中断进入挂起状态, 并可确定当前处于挂起状态的中断。通过将"1"写入到该寄存器中的某个位, 即可清除相应的挂起中断。但写入"1"不会影响已使用的中断信号。写入"0"无任何影响。通过读取该位, 即可返回相应中断的当前状态。</p>

注: 有关中断和中断编号的说明, 参见 9.5.1.5 节"中断源列表"。

9.6.2.12 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 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 34)	-	-
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
31-28	CLRPEND	R/W	<p>中断编号 [63:60]</p> <p>[写入]</p> <p>1: 清除挂起中断</p> <p>[读取]</p> <p>0: 未挂起</p> <p>1: 挂起</p> <p>各位对应于指定的编号, 强迫各中断进入挂起状态, 并可确定当前处于挂起状态的中断。通过将"1"写入到该寄存器中的某个位, 即可清除相应的挂起中断。但写入"1"不会影响已使用的中断信号。写入"0"无任何影响。通过读取该位, 即可返回相应中断的当前状态。</p>
27-26	-	R	读作 0。
25-4	CLRPEND	R/W	<p>中断编号 [57:36]</p> <p>[写入]</p> <p>1: 清除挂起中断</p> <p>[读取]</p> <p>0: 未挂起</p> <p>1: 挂起</p> <p>各位对应于指定的编号, 强迫各中断进入挂起状态, 并可确定当前处于挂起状态的中断。通过将"1"写入到该寄存器中的某个位, 即可清除相应的挂起中断。但写入"1"不会影响已使用的中断信号。写入"0"无任何影响。通过读取该位, 即可返回相应中断的当前状态。</p>
3	-	R	读作 0。
2	CLRPEND	R/W	<p>中断编号 [34]</p> <p>[写入]</p> <p>1: 清除挂起中断</p> <p>[读取]</p> <p>0: 未挂起</p> <p>1: 挂起</p> <p>各位对应于指定的编号, 强迫各中断进入挂起状态, 并可确定当前处于挂起状态的中断。通过将"1"写入到该寄存器中的某个位, 即可清除相应的挂起中断。但写入"1"不会影响已使用的中断信号。写入"0"无任何影响。通过读取该位, 即可返回相应中断的当前状态。</p>
1-0	-	R	读作 0。

注: 有关中断和中断编号的说明, 参见 9.5.1.5 节"中断源列表"。

9.6.2.13 中断优先级寄存器

各中断均具备中断优先级寄存器的八个位。

以下给出了各中断优先级寄存器的地址(与中断号对应)。

	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_15		PRI_14		-		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_30		PRI_29		PRI_28
0xE000_E420		-		PRI_34		-		-
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_57		PRI_56
0xE000_E43C		PRI_63		PRI_62		PRI_61		PRI_60

各产品拟用于优先级指定的位数各不相同。本产品有三个位用于指定优先级。

对于编号为0~3的中断信号，中断优先级寄存器的字段见下表。未使用的位在读取时会返回"0"，且写入未使用的位将不起任何作用。

	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	中断编号 3 的优先级
28-24	-	R	读作 0。
23-21	PRI_2	R/W	中断编号 2 的优先级
20-16	-	R	读作 0。
15-13	PRI_1	R/W	中断编号 1 的优先级
12-8	-	R	读作 0。
7-5	PRI_0	R/W	中断编号 0 的优先级
4-0	-	R	读作 0。

9.6.2.14 矢量表位移寄存器

	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 本矢量表位于： 0: 代码空间 1: SRAM空间
28-7	TBLOFF	R/W	偏移值 将偏移值从空间顶部设置到TBLBASE中的指定空间。 偏置必须根据表格中的例外规则的数量对齐。这表示最低对齐是 32 个字符，你可以把这 32 个字符用于多达 16 次的中断。对于更多的中断，必须通过凑整到最近的二次方调整对齐。
6-0	-	R	读作 0。

9.6.2.15 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 (写入)/ VECTKEY-STAT(读取)	R/W	寄存器键 [写入]如拟写入到该寄存器, 则 <VECTKEY>字段中必须有0x5FA字样。 [读取] 读作0x5FA05。
15	ENDIANESS	R/W	字节存储次序位: (注 1) 1: 从大到小 0: 从小到大
14-11	-	R	读作"0"。
10-8	PRIGROUP	R/W	中断优先级分组 000: 先占优先级七位, 子优先权一位 001: 先占优先级六位, 子优先权二位 010: 先占优先级五位, 子优先权三位 011: 先占优先级四位, 子优先权四位 100: 先占优先级三位, 子优先权五位 101: 先占优先级二位, 子优先权六位 110: 先占优先级一位, 子优先权七位 111: 先占优先级无位, 子优先权八位 位组合旨在将该中断优先权寄存器<PRI_n>划分为先占优先级与子优先级。
7-3	-	R	读作 0。
2	SYSRESET REQ	R/W	系统复位请求 1=CPU输出一个 SYSRESETREQ 信号.(注 2)
1	VECTCLR ACTIVE	R/W	清除活动矢量位 1: 清除活动NMI, 故障与中断的所有状态信息。 0: 不清除。该位可自行清除。 该应用负责令栈重新初始化。
0	VECTRESET	R/W	系统复位位 1: 复位系统 0: 不复位系统 复位该系统, 但各调试部件(FPB, DWT与ITM)除外; 方法是设置"1", 并对该位进行清零。

注 1: 从小到大是本产品的默认存储格式。

注 2: 在输出SYSRESETREQ时, 本产品会执行热复位通过热复位即可清除<SYSRESETREQ>。



9.6.2.16 系统处理器优先寄存器

各异常均具备系统处理器优先寄存器的八个位。

各例外情况对应的系统处理器优先权寄存器的地址，见下表。

	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 (存储器管理)	
0xE000_ED1C	PRI_11 (SVCall)		PRI_10		PRI_9		PRI_8	
0xE000_ED20	PRI_15 (SysTick)		PRI_14 (PendSV)		PRI_13		PRI_12 (调试监视器)	

各产品拟用于优先级指定的位数各不相同。本产品有三个位用于指定优先级。

以下给出了存储器管理，总线故障与使用故障用系统处理器优先寄存器的各字段。未使用的位在读取时会返回"0"，且写入未使用的位将不起任何作用。

	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	使用故障的优先级
20-16	-	R	读作 0。
15-13	PRI_5	R/W	总线故障的优先级
12-8	-	R	读作 0。
7-5	PRI_4	R/W	存储器管理的优先级
4-0	-	R	读作 0。

9.6.2.17 系统处理器控制器与状态寄存器

	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 PENDE	BUSFAULT PENDE	MEMFAULT PENDE	USGFAUL T PENDE	SYSTICKAC T	PENDSVAC T	-	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	BUSFAUL TENA	R/W	总线故障 0: 禁用 1: 启用
16	MEMFAULT ENA	R/W	存储器管理 0: 禁用 1: 启用
15	SVCALL PENDE	R/W	SVCALL 0: 未挂起状态 1: 进入挂起状态
14	BUSFAULT PENDE	R/W	总线故障 0: 未挂起状态 1: 进入挂起状态
13	MEMFAULT PENDE	R/W	存储器管理 0: 未挂起状态 1: 进入挂起状态
12	USGFAULT PENDE	R/W	使用故障 0: 未挂起状态 1: 进入挂起状态
11	SYSTICKAC T	R/W	SysTick 0: 无效 1: 激活
10	PENDSVAC T	R/W	PendSV 0: 无效 1: 激活
9	-	R	读作 0。
8	MONITORA CT	R/W	调试监督程序 0: 无效

位	比特符号	类型	功能
			1: 激活
7	SVCALLACT	R/W	SVCall 0: 无效 1: 激活
6-4	-	R	读作 0。
3	USGFAULT ACT	R/W	使用故障 0: 无效 1: 激活
2	-	R	读作 0。
1	BUSFAULT ACT	R/W	总线故障 0: 无效 1: 激活
0	MEMFAULT ACT	R/W	存储器管理 0: 无效 1: 激活

注：操作员在清除或设置各激活位时必须非常小心，原因是这些位的清除与设置并不能修复栈内容。

9.6.3 时钟发生器寄存器

9.6.3.1 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等待清除请求的有效电平设置。(101 ~ 111: 设置禁止) 000: "低"电平 001: "高"电平 010: 下降缘 011: 上升缘 100: 双边缘
27-26	EMST3[1:0]	R	INT3待机清除请求的有效电平 00: - 01: 上升缘 10: 下降缘 11: 双边缘
25	-	R	读作未定义。
24	INT3EN	R/W	INT3归零输入 0: 禁用 1: 启用
23	-	R	读作"0"。
22-20	EMCG2[2:0]	R/W	INT2待机清除请求的有效电平设置(101 ~ 111: 设置禁止) 000: "低"电平 001: "高"电平 010: 下降缘 011: 上升缘 100: 双边缘
19-18	EMST2[1:0]	R	INT2待机清除请求的有效电平 00: - 01: 上升缘 10: 下降缘 11: 双边缘
17	-	R	读作未定义。
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待机清除请求的有效电平设置。(101 ~ 111: 设置禁止) 000: "低"电平 001: "高"电平 010: 下降缘 011: 上升缘 100: 双边缘
11-10	EMST1[1:0]	R	INT1待机清除请求的有效电平 00: - 01: 上升缘 10: 下降缘 11: 双边缘
9	-	R	读作未定义。
8	INT1EN	R/W	INT1归零输入 0: 禁用 1: 启用
7	-	R	读作 0。
6-4	EMCG0[2:0]	R/W	INT0待机清除请求的有效电平设置(101 ~ 111: 设置禁止) 000: "低"电平 001: "高"电平 010: 下降缘 011: 上升缘 100: 双边缘
3-2	EMST0[1:0]	R	INT0待机清除请求的有效电平 00: - 01: 上升缘 10: 下降缘 11: 双边缘
1	-	R	读作未定义。
0	INT0EN	R/W	INT0清除输入 0: 禁用 1: 启用

注 1: <EMSTx> 仅在上升缘与下降缘的<EMCGx[2:0]>均被设置为"100"时才有效。可通过<EMSTx>检查等待复位所用的有效电平。如果用CGICRCG寄存器清除了各中断, 则<EMSTx>也同时被清除。

注 2: 请首先指定供该边使用的位, 然后再指定供<INTxEN>使用的位。自动禁止对其进行设置。

9.6.3.2 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待机清除请求的有效电平设置(101 ~ 111: 设置禁止) 000: "低"电平 001: "高"电平 010: 下降缘 011: 上升缘 100: 双边缘
27-26	EMST7[1:0]	R	INT7待机清除请求的有效电平 00: - 01: 上升缘 10: 下降缘 11: 双边缘
25	-	R	读作未定义。
24	INT7EN	R/W	INT7清除输入 0: 禁用 1: 启用
23	-	R	读作"0"。
22-20	EMCG6[2:0]	R/W	INT6待机清除请求的有效电平设置(101 ~ 111: 设置禁止) 000: "低"电平 001: "高"电平 010: 下降缘 011: 上升缘 100: 双边缘
19-18	EMST6[1:0]	R	INT6待机清除请求的有效电平 00: - 01: 上升缘 10: 下降缘 11: 双边缘
17	-	R	读作未定义。
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待机清除请求的有效电平设置(101 ~ 111: 设置禁止) 000: "低"电平 001: "高"电平 010: 下降缘 011: 上升缘 100: 双边缘

位	比特符号	类型	功能
11-10	EMST5[1:0]	R	INT5待机清除请求的有效电平 00: - 01: 上升缘 10: 下降缘 11: 双边缘
9	-	R	读作未定义。
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待机清除请求的有效电平设置(101 ~ 111: 设置禁止) 000: "低"电平 001: "高"电平 010: 下降缘 011: 上升缘 100: 双边缘
3-2	EMST4[1:0]	R	INT4待机清除请求的有效电平 00: - 01: 上升缘 10: 下降缘 11: 双边缘
1	-	R	读作未定义。
0	INT4EN	R/W	INT4清除输入 0: 禁用 1: 启用

注 1: <EMSTx> 仅在上升缘与下降缘的<EMCGx[2:0]>均被设置为"100"时才有效。可通过提交<EMSTx>, 来检查待机复位有效电平。如果用CGICRCG寄存器清除了各中断, 则<EMSTx>也同时被清除。

注 2: 请首先指定供该边使用的位, 然后再指定供<INTxEN>使用的位。自动禁止对其进行设置。

9.6.3.3 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	INTUSBWKUP设置命令的检测状态。(101 ~ 111: 设置禁止) 000: "低"电平 001: "高"电平 010: 下降缘 011: 上升缘 100: 双边缘
27-26	EMSTB[1:0]	R	读作未定义。
25	-	R	读作未定义。
24	INTBEN	R/W	INTUSBWKUP检测 0: 禁用 1: 启用
23	-	R	读作 0。
22-20	EMCGA[2:0]	R/W	INTUSBON设置命令的检测状态。(101 ~ 111: 设置禁止) 000: "低"电平 001: "高"电平 010: 下降缘 011: 上升缘 100: 双边缘
19-18	EMSTA[1:0]	R	读作未定义。
17	-	R	读作未定义。
16	INTAEN	R/W	INTUSBPON检测 0: 禁用 1: 启用
15	-	R	读作"0"。
14-12	EMCG9[2:0]	R/W	INT9等待清除请求的有效电平(101 ~ 111: 设置禁止) 000: "低"电平 001: "高"电平 010: 下降缘 011: 上升缘 100: 双边缘
11-10	EMST9[1:0]	R	INT9等待清除请求的有效电平 00: - 01: 上升缘 10: 下降缘 11: 双边缘
9	-	R	读作未定义。
8	INT9EN	R/W	INT9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	EMCG8[2:0]	R/W	INT8等待清除请求的有效电平设置。(101 ~ 111: 设置禁止) 000: "低"电平 001: "高"电平 010: 下降缘 011: 上升缘 100: 双边缘
3-2	EMST8[1:0]	R	INT8待机清除请求的有效电平 00: - 01: 上升缘 10: 下降缘 11: 双边缘
1	-	R	读作未定义。
0	INT8EN	R/W	INT8清除输入 0: 禁用 1: 启用

注 1: <EMSTx> 仅在上升缘与下降缘的<EMCGx[2:0]>均被设置为"100"时才有效。可通过提交<EMSTx>，来检查待机复位有效电平。如果用CGICRCG寄存器清除了各中断，则<EMSTx>也同时被清除。

注 2: 请首先指定供该边使用的位，然后再指定供<INTxEN>使用的位。自动禁止对其进行设置。

9.6.3.4 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 0_0001: INT1 0_1001: INT9 0_0010: INT2 0_1010: INTUSBPON 0_0011: INT3 0_1011: INTUSBWKUP 0_0100: INT4 0_0101: INT5 0_0110: INT6 0_0111: INT7 0_1100 ~ 1_1111: 保留 读作 0。

9.6.3.5 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源生成标志 0: 不适用 1: 从 $\overline{\text{NMI}}$ 引脚中生成
0	NMIFLG0	R	NMI源生成标志 0: 不适用 1: 从WDT中生成

注：当读取<NMIFLG>时，被清除到"0"。

9.6.3.6 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	-	-	-	DBGRSTF	-	WDTRSTF	-	PINRSTF
引脚复位后	0	0	0	0	未定义	0	未定义	1

位	比特符号	类型	功能
31-5	-	R	读作 0。
4	DBGRSTF	R/W	调试复位标志 (注 1) 0: 写入"0" 1: 从SYSRESETREQ复位
3	-	R/W	读作未定义。 写作 0。
2	WDTRSTF	R/W	WDT复位标志 0: 写入"0" 1: 从WDT复位
1	-	R/W	读作未定义。 写作 0。
0	PINRSTF	R/W	$\overline{\text{RESET}}$ 引脚标志 0: 写入"0" 1: 从 $\overline{\text{RESET}}$ 引脚复位

注 1: 该标志表示CPU的NVIC的应用中断和复位控制寄存器SYSRESETREQ位生成的复位。

注 2: 本产品通过外部复位初始化。

10. 输入/输出端口

10.1 端口功能

10.1.1 功能表

TMPM365FYXBG 有 74 个端口。除了端口功能外，这些端口可用作 I/O 引脚，用于外设功能。

表 10-1, 10-2 和 10-3 所示为端口功能表。

表 10-1 端口功能列表(端口A ~ C)

端口	引脚	输入/输出	上拉 下拉	施密特 输入	噪声滤波 器	可编程 开漏	功能引脚
PORTA							
	PA0	I/O	上拉	-	-	0	-
	PA1	I/O	上拉	-	-	0	-
	PA2	I/O	上拉	-	-	0	-
	PA3	I/O	上拉	-	-	0	-
	PA4	I/O	上拉	-	-	0	-
	PA5	I/O	上拉	-	-	0	-
	PA6	I/O	上拉	-	-	0	-
	PA7	I/O	上拉	-	-	0	-
PORTB							
	PB0	I/O	上拉	-	-	0	-
	PB1	I/O	上拉	-	-	0	-
	PB2	I/O	上拉	-	-	0	-
	PB3	I/O	上拉	-	-	0	-
	PB4	I/O	上拉	-	-	0	-
	PB5	I/O	上拉	-	-	0	-
	PB6	I/O	上拉	-	-	0	-
	PB7	I/O	上拉	-	-	0	-
PORTC							
	PC0	I/O	上拉	0	-	0	TXD1/ TB2IN0
	PC1	I/O	上拉	0	-	0	RXD1/ TB2IN1
	PC2	I/O	上拉	0	-	0	SCLK1/ TB0OUT/ $\overline{\text{CTS1}}$

0: 存在

-: 不存在

注: 在常见条件下, 噪声滤波器的噪声消除宽度约 30 ns。



表 10-2 端口功能列表(端口D ~ G)

端口	引脚	I/O端口	上拉 下拉	施密特 输入	噪声滤波 器	可编程 开漏	功能引脚
PORTD							
	PD0	I/O	上拉	o	-	o	TB7OUT
	PD1	I/O	上拉	o	-	o	TB8OUT
	PD2	I/O	上拉	o	-	o	TB9OUT
	PD3	I/O	上拉	o	-	o	$\overline{\text{ADTRG}}$
	PD4	I/O	上拉	-	-	o	-
	PD5	I/O	上拉	-	-	o	-
	PD6	I/O	上拉	-	-	o	-
	PD7	I/O	上拉	o	-	o	SCOUT
PORTE							
	PE0	I/O	上拉	o	-	o	TXD0
	PE1	I/O	上拉	o	-	o	RXD0
	PE2	I/O	上拉	o	-	o	SCLK0/ TB2OUT/ $\overline{\text{CTS0}}$
	PE3	I/O	上拉	o	o	o	INT5/ TB3OUT
	PE4	I/O	上拉	o	-	o	SDA1/SO1
	PE5	I/O	上拉	o	-	o	SCL1/SI1
	PE6	I/O	上拉	o	-	o	SCK1
	PE7	I/O	上拉	o	o	o	INT4
PORTF							
	PF0	输出	上拉	o	-	o	$\overline{\text{BOOT}}$ / TB6OUT
	PF1	I/O	上拉	o	-	o	-
	PF2	I/O	上拉	o	-	o	-
	PF3	I/O	上拉	o	-	o	-
	PF4	I/O	上拉	o	o	o	INT6/ TB5IN0
	PF5	I/O	上拉	o	o	o	INT7/ TB5IN1
	PF6	I/O	上拉	o	-	o	-
	PF7	I/O	上拉	o	-	o	-
PORTG							
	PG0	I/O	上拉	o	-	o	SDA0/SO0
	PG1	I/O	上拉	o	-	o	SCL0/ SI0/ TB3IN0
	PG2	I/O	上拉	o	-	o	SCK0/ TB3IN1
	PG3	I/O	上拉	o	o	o	INT0/ TB4IN0
	PG4	I/O	上拉	o	-	o	TB4IN1
	PG5	I/O	上拉	o	o	o	INT1/ USBPON

o 存在

-: 不存在

注: 在常见条件下, 噪声滤波器的噪声消除宽度约 30 ns。

表 10-3 端口功能列表(端口H~K)

端口	引脚	I/O端口	上拉 下拉	施密特 输入	噪声 滤波器	可编程 开漏	功能引脚
PORTH							
	PH0	I/O	上拉	o	-	o	TRACEDATA2
	PH1	I/O	上拉	o	-	o	TRACEDATA3
	PH2	I/O	上拉	o	-	o	TB4OUT
	PH3	I/O	上拉	o	-	o	TB5OUT
	PH4	I/O	上拉	o	o	o	INT8
PORTI							
	PI0	I/O	上拉	o	-	o	TRACEDATA1
	PI1	I/O	上拉	o	-	o	TRACEDATA0
	PI2	I/O	上拉	o	-	o	TRACECLK
	PI3	I/O	复位后下拉	o	-	-	TCK/ SWCLK
	PI4	I/O	复位后上拉	o	-	-	TMS/ SWDIO
	PI5	I/O	上拉	o	-	-	TDO/ SWV
	PI6	I/O	复位后上拉	o	-	-	TDI
	PI7	I/O	复位后上拉	o	o	-	TRST
PORTJ							
	PJ0	I/O	上拉	o	-	-	AIN00
	PJ1	I/O	上拉	o	-	-	AIN01
	PJ2	I/O	上拉	o	-	-	AIN02
	PJ3	I/O	上拉	o	-	-	AIN03
	PJ4	I/O	上拉	o	-	-	AIN04
	PJ5	I/O	上拉	o	-	-	AIN05
	PJ6	I/O	上拉	o	-	-	AIN06/ TB0IN0
	PJ7	I/O	上拉	o	o	-	AIN07/ INT9/ TB0IN1
PORTK							
	PK0	I/O	上拉	o	o	-	AIN08/ INT2/ TB1IN0
	PK1	I/O	上拉	o	o	-	AIN09/ INT3/ TB1IN1
	PK2	I/O	上拉	o	-	-	AIN10/ TB6IN0
	PK3	I/O	上拉	o	-	-	AIN11/ TB6IN1

o: 存在

-: 不存在

注: 在常见条件下, 噪声滤波器的噪声消除宽度约 30 ns。

10.1.2 端口寄存器概述

使用端口时，需配置下列寄存器。

- PxDATA: 端口x数据寄存器
要读取/写入端口数据
- PxCR: 端口x输出控制寄存器
控制输出。
控制输入时，需配置PxIE。
- PxFRn: 端口x功能寄存器n
设置功能。
通过设置"1"，就能激活指定的功能。
- PxOD: 端口x开漏控制寄存器
控制可编程开漏。
可编程开漏是通过设置PxOD而要实现的假开漏功能。
当PxOD设置为"1"时，输出缓冲器禁用而实现假开漏。
- PxPUP: 端口x上拉控制寄存器
要控制程序的上拉：
- PxPDN: 端口x下拉控制寄存器
控制可编程下拉。
- PxIE: 端口x输入控制寄存器
控制输入。
为避免直通电流，默认设置禁止输入。

10.1.3 STOP1 模式下端口状态

STOP1模式下的输入和输出，通过CGSTBYCR<DRVE>位来启用/禁用。

若通过 <DRVE> = "1" 启用了 PxIE 或 PxCR，则在 STOP1 模式下，输入和输出分别启用。若<DRVE>="0"，在除某些端口外，即使启用PxIE或PxCR，在STOP1模式下，输入和输出也被禁用。

表 10-4 所示为STOP模式下的引脚状况。

表 10-4 STOP模式下的端口状况

功能	引脚名称	I/O	STOP1		STOP2	
			<DRVE> = 0	<DRVE> = 1	<PTKEEP> = 0	<PTKEEP> = 1
控制引脚	RESET, NMI, MODE, BSC	输入	o	o	x	o
振荡器	X1/EHCLKIN	输入(注1)	x	x	x	x
	X2	输出(注1)	"高"电平输出	"高"电平输出	x	x
端口	PI3 ~PI5 (TRST, TDI, SWCLK/TCK) (调试I/F设置, 对于 PxFRn<PxmFn>="1")	输入	取决于 (PxIE[m])		x	输入保持, 但取决于 (PxIE[m])
	PI4 (SWDIO/TMS) (调试I/F设置, 对于 PxFRn<PxmFn>="1")	输入	取决于 (PxIE[m])		x	输入保持, 但取决于 (PxIE[m])
		输出	启用在数据有效时。被禁用数据无效时。		x	输出保持, 但取决于 (PxCR[m])
	PI5, PI2, PI1, PI0, PH0, PH1 (TDO/SWV, TRACECLK, TRACEDATA0 ~ 3) (调试I/F设置, 对于 PxFRn<PxmFn>="1")	输出	取决于 (PxCR[m])		x	输出保持, 但取决于 (PxCR[m])
	PG3, PG5, PK0, PK1, PE7, PE3, PF4, PF5, PH4, PJ7 (INT0 ~9) (中断设置, 对于 PxFRn<PxmFn>="1"和 PxIE<PxmiE>="1")	输入	o	o	x	o
	如使用除上述所列的	输入	x	取决于 (PxIE[m])	x	输入保持, 但取决于 (PxIE[m])
输出		x	取决于 (PxCR[m])	x	输出保持, 但取决于 (PxCR[m])	

o：启用输入或输出。

x：禁用输入或输出。

注：“x”指端口号，“m”指对应位号，“n”指功能寄存器编号。

10.2 端口功能

本章描述端口寄存器详细情况。

本章仅描述"电路型"读取电路的布置。具体电路图见 10.3 节"端口方块图"。

10.2.1 端口A(PA0 ~ PA7)

端口A为 8-位输入/输出端口。可规定该端口的输入和输出，单位为位。

通过复位，端口A的所有位均初始化为通用端口，输入输出和上拉则被禁用。

10.2.1.1 端口A寄存器

基址 = 0x400C_0000

寄存器名称		地址(基+)
端口A数据寄存器	PADATA	0x0000
端口A输出控制寄存器	PACR	0x0004
保留	-	0x0008
保留	-	0x000C
保留	-	0x0010
保留	-	0x0014
保留	-	0x0018
保留	-	0x001C
保留	-	0x0020
保留	-	0x0024
端口A开漏控制寄存器	PAOD	0x0028
端口A上拉控制寄存器	PAPUP	0x002C
保留	-	0x0030
保留	-	0x0034
端口A输入控制寄存器	PAIE	0x0038

注：禁止访问各"保留"区域。

10.2.1.2 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

10.2.1.3 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

10.2.1.4 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: 推挽 1: 开漏

10.2.1.5 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

10.2.1.6 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

10.2.2 端口B (PB0 ~ PB7)

端口B是通用型 8-位输入/输出端口。可规定该端口的输入和输出，单位为位。

通过复位，端口B的所有位均初始化为通用端口，输入输出和上拉则被禁用。

10.2.2.1 端口B寄存器

基址 = 0x400C_0100

寄存器名称		地址(基+)
端口B数据寄存器	PBDATA	0x0000
端口B输出控制寄存器	PBCR	0x0004
保留	-	0x0008
保留	-	0x000C
保留	-	0x0010
保留	-	0x0014
保留	-	0x0018
保留	-	0x001C
保留	-	0x0020
保留	-	0x0024
端口B开漏控制寄存器	PBOD	0x0028
端口B上拉控制寄存器	PBPUP	0x002C
保留	-	0x0030
保留	-	0x0034
端口B输入控制寄存器	PBIE	0x0038

注：禁止访问各"保留"区域。

10.2.2.2 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

10.2.2.3 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



10.2.2.4 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: 推挽 1: 开漏

10.2.2.5 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-PB0U P	R/W	上拉 0: 禁用 1: 启用

10.2.2.6 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

10.2.3 端口C (PC0 ~ PC2)

端口C是通用型 3-位输入/输出端口。可规定该端口的输入和输出，单位为位。除了通用输入/输出功能外，端口C还执行通用串口(UART/SIO)，16-位定时器(TMRB)的功能。

通过复位，端口C的所有位均初始化为通用端口，输入输出和上拉则被禁用。

端口C有三种功能寄存器。若将端口C用作通用端口，应将四个寄存器的对应位设置为"0"。若不将端口C作为通用端口，则应将功能寄存器的对应位设置为"1"。不得同时将其它寄存器设置为"1"。

10.2.3.1 端口C寄存器

基址 = 0x400C_0200

寄存器名称		地址(基+)
端口C数据寄存器	PCDATA	0x0000
端口C输出控制寄存器	PCCR	0x0004
端口C功能寄存器 1	PCFR1	0x0008
保留	-	0x000C
端口C功能寄存器 3	PCFR3	0x0010
端口C功能寄存器 4	PCFR4	0x0014
保留	-	0x0018
保留	-	0x001C
保留	-	0x0020
保留	-	0x0024
端口C开漏控制寄存器	PCOD	0x0028
端口C上拉控制寄存器	PCPUP	0x002C
保留	-	0x0030
保留	-	0x0034
端口C输入控制寄存器	PCIE	0x0038

注：禁止访问各"保留"区域。

10.2.3.2 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	-	-	-	-	-	PC2	PC1	PC0
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"。
2-0	PC2-PC0	R/W	端口C数据寄存器

10.2.3.3 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	-	-	-	-	-	PC2C	PC1C	PC0C
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。
2-0	PC2C-PC0C	R/W	输出 0: 禁用 1: 启用

10.2.3.4 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	-	-	-	-	-	PC2F1	PC1F1	PC0F1
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	PC2F1	R/W	0: PORT 1: SCLK1
1	PC1F1	R/W	0: PORT 1: RXD1
0	PC0F1	R/W	0: PORT 1: TXD1

10.2.3.5 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	-	-	-	-	-	PC2F3	PC1F3	PC0F3
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	PC2F3	R/W	0: PORT 1: TB0OUT
1	PC1F3	R/W	0: PORT 1: TB2IN1
0	PC0F3	R/W	0: PORT 1: TB2IN0

10.2.3.6 PCFR4 (端口C 功能寄存器 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	-	-	-	-	-	PC2F4	-	-
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	PC2F4	R/W	0: PORT 1: $\overline{\text{CTS1}}$
1-0	-	R	读作 0。

10.2.3.7 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	-	-	-	-	-	PC2OD	PC1OD	PC0OD
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。
2-0	PC2OD- PC0OD	R/W	0: 推挽 1: 开漏

10.2.3.8 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	-	-	-	-	-	PC2UP	PC1UP	PC0UP
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。
2-0	PC2UP-PC0UP	R/W	上拉 0: 禁用 1: 启用

10.2.3.9 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	-	-	-	-	-	PC2IE	PC1IE	PC0IE
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。
2-0	PC2IE-PC0IE	R/W	输入 0: 禁用 1: 启用

10.2.4 端口D (PD0 ~ PD7)

端口D是通用型 8-位输入/输出端口。可规定该端口的输入和输出，单位为位。除了通用输入/输出功能外，端口D还执行 16-位定时器(TMRB)，时钟输出和ADC触发器输入的功能。

通过复位，端口D的所有位均初始化为通用端口，输入输出和上拉则被禁用。

端口D有一种功能寄存器。若将端口D用作通用端口，应将 3 个寄存器的对应位设置为"0"。当将端口D用作非通用端口时，将功能寄存器的对应位设置为"1"。

10.2.4.1 端口D寄存器

基址 = 0x400C_0300

寄存器名称		地址(基+)
端口D数据寄存器	PDDATA	0x0000
端口D输出控制寄存器	PDCR	0x0004
保留	-	0x0008
保留	-	0x000C
端口D功能寄存器 3	PDFR3	0x0010
保留	-	0x0014
保留	-	0x0018
保留	-	0x001C
保留	-	0x0020
保留	-	0x0024
端口D开漏控制寄存器	PDOD	0x0028
端口D上拉控制寄存器	PDPUP	0x002C
保留	-	0x0030
保留	-	0x0034
端口D输入控制寄存器	PDIE	0x0038

注：禁止访问各"保留"区域。

10.2.4.2 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

10.2.4.3 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

10.2.4.4 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	PD7F3	-	-	-	PD3F3	PD2F3	PD1F3	PD0F3
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	PD7F3	R/W	0: PORT 1: SCOUT
6-4	-	R	读作"0"。
3	PD3F3	R/W	0: PORT 1: $\overline{\text{ADTRG}}$
2	PD2F3	R/W	0: PORT 1: TB9OUT
1	PD1F3	R/W	0: PORT 1: TB8OUT
0	PD0F3	R/W	0: PORT 1: TB7OUT

10.2.4.5 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: 推挽 1: 开漏

10.2.4.6 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

10.2.4.7 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

10.2.5 端口E (PE0 ~ PE7)

端口E是通用型 8-位输入/输出端口。可规定该端口的输入和输出，单位为位。除了通用输入/输出功能外，端口E还执行通用串口(SIO/UART)，外部中断输入，16-位定时器(TMRB)和串行总线接口(I2C/SIO)的功能。

通过复位，端口E的所有位均初始化为通用端口，输入输出和上拉则被禁用。

端口E有 3 类功能寄存器。若将端口E用作通用端口，应将四个寄存器的对应位设置为"0"。若不将端口E作为通用端口，则应将功能寄存器的对应位设置为"1"。不得同时将其它寄存器设置为"1"。

注：在非STOP模式时，若输入在PxIE中启用，则不管PxFR寄存器的设置，中断输入仍会启用。当给装置编程时，保证禁用不用的中断。

10.2.5.1 端口E寄存器

基址 = 0x400C_0400

寄存器名称		地址(基+)
端口E数据寄存器	PEDATA	0x0000
端口E输出控制寄存器	PECR	0x0004
端口E功能寄存器 1	PEFR1	0x0008
保留	-	0x000C
端口E功能寄存器 3	PEFR3	0x0010
端口E功能寄存器 4	PEFR4	0x0014
保留	-	0x0018
保留	-	0x001C
保留	-	0x0020
保留	-	0x0024
端口E开漏控制寄存器	PEOD	0x0028
端口E上拉控制寄存器	PEPUP	0x002C
保留	-	0x0030
保留	-	0x0034
端口E输入控制寄存器	PEIE	0x0038

注：禁止访问各"保留"区域。

10.2.5.2 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	读作 0。
7-0	PE7-PE0	R/W	端口E数据寄存器

10.2.5.3 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



10.2.5.4 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	PE7F1	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	PE7F1	R/W	0: PORT 1: INT4
6	PE6F1	R/W	0: PORT 1: SCK1
5	PE5F1	R/W	0: PORT 1: SCL1/SI1
4	PE4F1	R/W	0: PORT 1: SDA1/SO1
3	PE3F1	R/W	0: PORT 1: INT5
2	PE2F1	R/W	0: PORT 1: SCLK0
1	PE1F1	R/W	0: PORT 1: RXD0
0	PE0F1	R/W	0: PORT 1: TXD0

10.2.5.5 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	-	-	-	-	PE3F3	PE2F3	-	-
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	PE3F3	R/W	0: PORT 1: TB3OUT
2	PE2F3	R/W	0: PORT 1: TB2OUT
1-0	-	R	读作 0。

10.2.5.6 PEFR4 (端口E功能寄存器 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	-	-	-	-	-	PE2F4	-	-
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	PE2F4	R/W	0: PORT 1: $\overline{CTS0}$
1-0	-	R	读作 0。

10.2.5.7 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: 推挽 1: 开漏

10.2.5.8 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

10.2.5.9 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

10.2.6 端口F (PF0 ~ PF7)

端口F是一个通用型，1-位输出端口和 7-位输入/输出的端口。对于该端口，输入和输出值可在位单位中规定。除了通用输入/输出功能外，端口F还执行外部中断输入，16-位定时器(TMRB)和模式设置的功能。

端口F有两种功能寄存器。若将端口E用作通用端口，应将 3 个寄存器的对应位设置为"0"。若不将端口F作为通用端口，则应将功能寄存器的对应位设置为"1"。不得同时将其它寄存器设置为"1"。

通过复位，端口F的所有位均初始化为通用端口。PF0 输出位被禁用，上拉设置被启用，PF<7:1>输入/输出位和所有上拉设置位被禁用。

根据模式设置功能，当复位信号为低电平状态时，PF0/ $\overline{\text{BOOT}}$ 输入和上拉功能启用。在复位信号的上升缘，若PF0在"高"电平，装置进入单芯片模式，并从片上显存启动。若PF0电平为"低"电平，装置将进入单BOOT模式，从内部BOOT程序启动。对于单BOOT模式，参考"闪存运行"对应章节。

注：在STOP1 模式外的模式下，若输入已在PxIE中启用，则无论PxFR寄存器配置如何，均启用中断输入。当给装置编程时，保证禁用不用的中断。

10.2.6.1 F端口寄存器

基址 = 0x400C_0500

寄存器名称		地址(基+)
端口F数据寄存器	PFDATA	0x0000
端口F输出控制寄存器	PFCR	0x0004
保留	-	0x0008
端口F功能寄存器 2	PFFR2	0x000C
端口F功能寄存器 3	PFFR3	0x0010
保留	-	0x0014
保留	-	0x0018
保留	-	0x001C
保留	-	0x0020
保留	-	0x0024
端口F开漏控制寄存器	PFOD	0x0028
端口F上拉控制寄存器	PFPUP	0x002C
保留	-	0x0030
保留	-	0x0034
端口F输入控制寄存器	PFIE	0x0038

注：禁止访问各"保留"区域。



10.2.6.2 PFDATA (F端口数据寄存器)

	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	PF7	PF6	PF5	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
31-8	-	R	读作 0。
7-0	PF7-PF0	R/W	端口F数据寄存器

10.2.6.3 PFCR (F端口 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	PF7C	PF6C	PF5C	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
31-8	-	R	读作 0。
7-0	PF7C-PF0C	R/W	输出 0: 禁用 1: 启用



10.2.6.4 PFFR2 (F端口功能寄存器 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	-	-	PF5F2	PF4F2	-	-	-	-
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	PF5F2	R/W	0: PORT 1: INT7
4	PF4F2	R/W	0: PORT 1: INT6
3-0	-	R	读作 0。

10.2.6.5 PFFR3 (F端口功能寄存器 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	-	-	PF5F3	PF4F3	-	-	-	PF0F3
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	PF5F3	R/W	0: PORT 1: TB5IN1
4	PF4F3	R/W	0: PORT 1: TB5IN0
3-1	-	R	读作 0。
0	PF0F3	R/W	0: PORT 1: TB6OUT

10.2.6.6 PFOD (F端口开漏控制寄存器)

	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	PF7OD	PF6OD	PF5OD	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
31-8	-	R	读作 0。
7-0	PF7OD-PF0OD	R/W	0: 推挽 1: 开漏

10.2.6.7 PFPUP (F端口上拉控制寄存器)

	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	PF7UP	PF6UP	PF5UP	PF4UP	PF3UP	PF2UP	PF1UP	PF0UP
复位后	0	0	0	0	0	0	0	1

位	比特符号	类型	功能
31-8	-	R	读作 0。
7-1	PF7UP-PF1UP	R/W	上拉 0: 禁用 1: 启用
0	PF0UP	R/W	上拉 0: - 1: 总是设置为"1"。



10.2.6.8 PFIE (F端口输入控制寄存器)

	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	PF7IE	PF6IE	PF5IE	PF4IE	PF3IE	PF2IE	PF1IE	-
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	PF7IE-PF1IE	R/W	输入 0: 禁用 1: 启用
0	-	R	读作 0。

10.2.7 端口G (PG0 ~ PG5)

端口G是通用型 6 位输入/输出端口。可规定该端口的输入和输出，单位为位。除了通用输入/输出功能外，端口G还执行串行总线接口(12C/SIO)，外部中断输入，16-位定时器 (TMRB) 的功能，并能对与 V 总线相连的 USB 接头进行检测 (USBPON)。

通过复位，端口G的所有位均初始化为通用端口，输入输出和上拉则被禁用。

端口G有 3 种功能寄存器。若将端口G用作通用端口，应将 3 个寄存器的对应位设置为"0"。若不将端口E作为通用端口，则应将功能寄存器的对应位设置为"1"。不得同时将其它寄存器设置为"1"。

注 1: 在STOP模式外的模式下，若输入已在PxIE中启用，则无论PxFR寄存器配置如何，均启用中断输入。当给装置编程时，保证禁用不用的中断。

注 2: 只有当输入已启动的情况下，PG5才会操作5V输入。请注意，使用开漏输出时，上述引脚的上拉不能造成电压超过电源电压。

10.2.7.1 端口G寄存器

基址 = 0x400C_0600

寄存器名称		地址(基+)
端口G数据寄存器	PGDATA	0x0000
端口G输出控制寄存器	PGCR	0x0004
端口G功能寄存器 1	PGFR1	0x0008
保留	-	0x000C
端口G功能寄存器 3	PGFR3	0x0010
端口G功能寄存器 4	PGFR4	0x0014
保留	-	0x0018
保留	-	0x001C
保留	-	0x0020
保留	-	0x0024
端口G开漏控制寄存器	PGOD	0x0028
端口G上拉控制寄存器	PGPUP	0x002C
保留	-	0x0030
保留	-	0x0034
端口G输入控制寄存器	PGIE	0x0038

注：禁止访问各"保留"区域。

10.2.7.2 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	-	-	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
31-6	-	R	读作 0。
5-0	PG5-PG0	R/W	端口G数据寄存器

10.2.7.3 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	-	-	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
31-6	-	R	读作 0。
5-0	PG5C-PG0C	R/W	输出 0: 禁用 1: 启用

10.2.7.4 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	-	-	PG5F1	-	PG3F1	PG2F1	PG1F1	PG0F1
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-6	-	R	读作 0。
5	PG5F1	R/W	0: PORT 1: INT1
4	-	R	读作 0。
3	PG3F1	R/W	0: PORT 1: INT0
2	PG2F1	R/W	0: PORT 1: SCK0
1	PG1F1	R/W	0: PORT 1: SCL0/SI0
0	PG0F1	R/W	0: PORT 1: SDA0/SO0

10.2.7.5 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	-	-	-	PG4F3	PG3F3	PG2F3	PG1F3	-
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	PG4F3	R/W	0: PORT 1: TB4IN1
3	PG3F3	R/W	0: PORT 1: TB4IN0
2	PG2F3	R/W	0: PORT 1: TB3IN1
1	PG1F3	R/W	0: PORT 1: TB3IN0
0	-	R	读作 0。

10.2.7.6 PGFR4 (端口G功能寄存器 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	-	-	PG5F4	-	-	-	-	-
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	PG5F4	R/W	0: PORT 1: USBPON
4-0	-	R	读作 0。

10.2.7.7 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	-	-	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
31-6	-	R	读作 0。
5-0	PG5OD- PG0OD	R/W	0: 推挽 1: 开漏

注：只有在启用输入的情况下，上述引脚才能承受 5 V 输入电压。请注意，使用开漏输出时，上述引脚不能造成超过电源电压。

10.2.7.8 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	-	-	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
31-6	-	R	读作 0。
5-0	PG5UP- PG0UP	R/W	上拉 0: 禁用 1: 启用

10.2.7.9 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	-	-	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
31-6	-	R	读作 0。
5-0	PG5IE-PG0IE	R/W	输入 0: 禁用 1: 启用

10.2.8 端口H (PH0 ~ PH4)

端口H是通用型 5-位输入/输出端口。可规定该端口的输入和输出，单位为位。除了通用输入/输出功能外，端口H还执行调试跟踪接口，外部中断输入，16 位定时器(TMRB)的功能。

通过复位，端口H的所有位均初始化为通用端口，输入输出和上拉则被禁用。

端口H有两类寄存器。若将端口H用作通用端口，应将四个寄存器的对应位设置为"0"。若不将端口H作为通用端口，则应将功能寄存器的对应位设置为"1"。不得同时将其它寄存器设置为"1"。

注：在非STOP模式时，若输入在PxIE中启用，则不管PxFR寄存器的设置，中断输入仍会启用。当给装置编程时，保证禁用不用的中断。

10.2.8.1 端口H寄存器

基址 = 0x400C_0700

寄存器名称		地址(基+)
端口H数据寄存器	PHDATA	0x0000
端口H输出控制寄存器	PHCR	0x0004
端口H功能寄存器 1	PHFR1	0x0008
保留	-	0x000C
端口H功能寄存器 3	PHFR3	0x0010
保留	-	0x0014
保留	-	0x0018
保留	-	0x001C
保留	-	0x0020
保留	-	0x0024
端口H开漏控制寄存器	PHOD	0x0028
端口H上拉控制寄存器	PHPUP	0x002C
保留	-	0x0030
保留	-	0x0034
端口H输入控制寄存器	PHIE	0x0038

注：禁止访问各"保留"区域。

10.2.8.2 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	-	-	-	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
31-5	-	R	读作 0。
4-0	PH4-PH0	R/W	端口H数据寄存器

10.2.8.3 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	-	-	-	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
31-5	-	R	读作 0。
4-0	PH4C-PH0C	R/W	输出 0: 禁用 1: 启用

10.2.8.4 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	-	-	-	-	-	-	PH1F1	PH0F1
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	PH1F1	R/W	0: PORT 1: TRACEDATA3
0	PH0F1	R/W	0: PORT 1: TRACEDATA2

10.2.8.5 PHFR3 (端口H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	-	-	-	PH4F3	PH3F3	PH2F3	-	-
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	PH4F3	R/W	0: PORT 1: INT8
3	PH3F3	R/W	0: PORT 1: TB5OUT
2	PH2F3	R/W	0: PORT 1: TB4OUT
1-0	-	R	读作 0。

10.2.8.6 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	-	-	-	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
31-5	-	R	读作 0。
4-0	PH4OD- PH0OD	R/W	0: 推挽 1: 开漏

10.2.8.7 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	-	-	-	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
31-5	-	R	读作 0。
4-0	PH4UP-PH0U P	R/W	上拉 0: 禁用 1: 启用

10.2.8.8 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	-	-	-	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
31-5	-	R	读作 0。
4-0	PH4IE-PH0IE	R/W	输入 0: 禁用 1: 启用

10.2.9 端口I(PI0 ~ PI7)

端口I为 8-位输入/输出端口。可规定该端口的输入和输出，单位为位。除了一般输入/输出功能外，端口I还执行调试接口和调试跟踪接口的功能。

在复位后，将PI3, PI4, PI5, PI6 和PI7 配置为调试接口。PI7 初始化为TRST引脚，输入和上拉功能启用。PI6 初始化为 TD1 引脚，输入和上拉启用。PI3 初始化为 TCK/SWCLK 引脚，输入和上拉功能启用。PI4 初始化为TMS/SWDIO引脚，输入，输出和上拉均启用。PI5 初始化为TDO/SWV引脚，启用输出。其它引脚用作通用端口，输入，输出和上拉均禁用。

注 1: 若PI4 和PI5 配置为TMS/SWDIO或TDO/SWV 引脚，则无论CGSTBYCR<DRVE>位设置如何，即使在STOP1 模式下，输出也启用。

注 2: 若PI3 配置为TCK/SWCLK引脚，它能防止功耗模式完全有效。若未用TCK/SWCLK，则应将PI3 配置为通用端口。

10.2.9.1 端口I寄存器

基址 = 0x400C_0800

寄存器名称		地址(基+)
端口I 数据寄存器	PIDATA	0x0000
端口I 输出控制寄存器	PICR	0x0004
端口I 功能寄存器 1	PIFR1	0x0008
保留	-	0x000C
保留	-	0x0010
保留	-	0x0014
保留	-	0x0018
保留	-	0x001C
保留	-	0x0020
保留	-	0x0024
端口I 开漏控制寄存器	PIOD	0x0028
端口I 上拉控制寄存器	PIPUP	0x002C
端口I 下拉控制寄存器	PIPDN	0x0030
保留	-	0x0034
端口I 输入控制寄存器	PIIE	0x0038

注：禁止访问各"保留"区域。

10.2.9.2 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	PI7	PI6	PI5	PI4	PI3	PI2	PI1	PI0
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	PI7-PI0	R/W	端口 I 数据寄存器

10.2.9.3 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	PI7C	PI6C	PI5C	PI4C	PI3C	PI2C	PI1C	PI0C
复位后	0	0	1	1	0	0	0	0

位	比特符号	类型	功能
31-8	-	R	读作 0。
7-0	PI7-PI0C	R/W	输出 0: 禁用 1: 启用



10.2.9.4 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	PI7F1	PI6F1	PI5F1	PI4F1	PI3F1	PI2F1	PI1F1	PI0F1
复位后	1	1	1	1	1	0	0	0

位	比特符号	类型	功能
31-8	-	R	读作 0。
7	PI7F1	R/W	0: PORT 1: $\overline{\text{TRST}}$
6	PI6F1	R/W	0: PORT 1: TDI
5	PI5F1	R/W	0: PORT 1: TDO/SWV
4	PI4F1	R/W	0: PORT 1: TMS/SWDIO
3	PI3F1	R/W	0: PORT 1: TCK/SWCLK
2	PI2F1	R/W	0: PORT 1: TRACECLK
1	PI1F1	R/W	0: PORT 1: TRACEDATA0
0	PI0F1	R/W	0: PORT 1: TRACEDATA1

10.2.9.5 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	-	-	-	-	-	PI2OD	PI1OD	PI0OD
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。
2-0	PI2OD-PI0OD	R/W	0: 推拉 1: 开漏

10.2.9.6 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	PI7UP	PI6UP	PI5UP	PI4UP	-	PI2UP	PI1UP	PI0UP
复位后	1	1	0	1	0	0	0	0

位	比特符号	类型	功能
31-8	-	R	读作 0。
7-6	PI7UP-PI6UP	R/W	上拉 0: 禁用 1: 启用, 始终设置为"1"作为调试接口。
5	PI5UP	R/W	上拉 0: 禁用 1: 启用
4	PI4UP	R/W	上拉 0: 禁用 1: 启用, 始终设置为"1"作为调试接口。
3	-	R	读作 0。
2-0	PI2UP-PI0UP	R/W	上拉 0: 禁用 1: 启用

10.2.9.7 PIPDN (端口 I 下拉式控制寄存器)

	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	-	-	-	-	PI3DN	-	-	-
复位后	0	0	0	0	1	0	0	0

位	比特符号	类型	功能
31-4	-	R	读作 0。
3	PI3DN	R/W	下拉 0: 禁用 1: 启用, 始终设置为"1"作为调试接口。
2-0	-	R	读作 0。

10.2.9.8 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	PI7IE	PI6IE	PI5IE	PI4IE	PI3IE	PI2IE	PI1IE	PI0IE
复位后	1	1	0	1	1	0	0	0

位	比特符号	类型	功能
31-8	-	R	读作 0。
7-0	PI7IE-PI0IE	R/W	输入 0: 禁用 1: 启用

10.2.10 端口 J (PJ0 ~ PJ7)

端口 J 为通用型 8-位 输入/输出端口。对于此端口，可用位来规定输入和输出。除具有通用输入/输出功能外，端口 J 还具有 AD 转换器，外部中断输入和 16-位定时器(TMRB)的功能。

复位将端口 J 的所有位初始化为输入，输出和上拉式功能被禁用的通用型端口。

端口 J 具有两种类型的功能寄存器。当将端口 J 用作通用型端口时，将"0"设置至两个寄存器的对应位。当将端口 J 设置为通用型端口以外的端口时，则将"1"设置至功能寄存器的对应位。请勿同时把"1"设置到两个功能寄存器上。

将端口 J 用作 AD 转换器的模拟输入时，应禁止在 PJIE 上输入，且禁止在 PJPUP 上进行上拉。

注 1: 除把端口 J 和端口 K 的所有位用作模拟输入引脚外，可能会降低转换准确度。应验证这样做不会引起系统故障。

注 2: 在 STOP1 模式意外的其它模式中，如在 PxIE 中启用输入，则要启用中断输入，不管 PxFR 寄存器的设置如何。进行装置编程时，应确保不用的中断被禁用。

10.2.10.1 端口 J 寄存器

基址 = 0x400C_0900

寄存器名称		地址(基+)
端口 J 数据寄存器	PJDATA	0x0000
端口 J 输出控制寄存器	PJCR	0x0004
保留	-	0x0008
端口 J 功能寄存器 2	PJFR2	0x000C
端口 J 功能寄存器 3	PJFR3	0x0010
保留	-	0x0014
保留	-	0x0018
保留	-	0x001C
保留	-	0x0020
保留	-	0x0024
保留	-	0x0028
端口 J 上拉控制寄存器	PJPUP	0x002C
保留	-	0x0030
保留	-	0x0034
端口 J 输入控制寄存器	PJIE	0x0038

注：禁止访问"保留"区域。

10.2.10.2 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	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	PJ7-PJ0	R/W	端口 J 数据寄存器

10.2.10.3 PJCR (端口 J 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	PJ7C	PJ6C	PJ5C	PJ4C	PJ3C	PJ2C	PJ1C	PJ0C
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	PJ7C-PJ0C	R/W	输出 0: 禁用 1: 启用

10.2.10.4 PJFR2 (端口 J 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	PJ7F2	-	-	-	-	-	-	-
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: PORT 1: INT9
6-0		R	读作 0。

10.2.10.5 PJFR3 (端口 J 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	PJ7F3	PJ6F3	-	-	-	-	-	-
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	PJ7F3	R/W	0: PORT 1: TB0IN1
6	PJ6F3	R/W	0: PORT 1: TB0IN0
5-0	-	R	读作 0。

10.2.10.6 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

10.2.10.7 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

10.2.11 端口 K (PK0~PK3)

端口 K 为通用型 4 位输入/输出端口。对于此端口，可用位来规定输入和输出。除通用输入/输出功能外，端口 K 还具有 AD 转换器，外部中断输入和 16-位定时器(TMRB)功能。

端口 K 具有两种类型的功能寄存器。当将端口 K 用作通用型端口时，应将"0"设置至两个寄存器的对应位。当将端口 K 用作通用端口以外的端口时，应将"1"设置至功能寄存器的对应位。请勿同时把"1"设置到两个功能寄存器上。

复位将端口 J 的所有位初始化为输入，输出和上拉式功能被禁用的通用型端口。

将端口 K 用作 AD 转换器的模拟输入时，应禁用 PKIE 的输入功能并禁用 PKPUP 的上拉功能。

注 1: 除把端口 J 和端口 K 的所有位用作模拟输入引脚外，可能会降低转换准确度。应验证这样做不会引起系统故障。

注 2: 在 STOP1 模式意外的其它模式中，如在 PxE 中启用输入，则要启用中断输入，不管 PxFR 寄存器的设置如何。进行装置编程时，应确保不用的中断被禁用。

10.2.11.1 端口 K 寄存器

基址 = 0x400C_0A00

寄存器名称		地址(基+)
端口 K 数据寄存器	PKDATA	0x0000
端口 K 输出控制寄存器	PKCR	0x0004
保留	-	0x0008
端口 K 功能寄存器 2	PKFR2	0x000C
端口 K 功能寄存器 3	PKFR3	0x0010
保留	-	0x0014
保留	-	0x0018
保留	-	0x001C
保留	-	0x0020
保留	-	0x0024
保留	-	0x0028
端口 K 上拉控制寄存器	PKPUP	0x002C
保留	-	0x0030
保留	-	0x0034
端口 K 输入控制寄存器	PKIE	0x0038

注：禁止访问各"保留"区域。

10.2.11.2 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	-	-	-	-	PK3	PK2	PK1	PK0
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	PK3-PK0	R/W	端口 K 数据寄存器

10.2.11.3 PKCR(端口 K 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	-	-	-	-	PK3C	PK2C	PK1C	PK0C
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	PK3C-PK0C	R/W	输出 0: 禁用 1: 启用

10.2.11.4 PKFR2 (端口 K 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	-	-	-	-	-	-	PK1F2	PK0F2
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	PK1F2	R/W	0: PORT 1: INT3
0	PK0F2	R/W	0: PORT 1: INT2

10.2.11.5 PKFR3 (端口 K 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	-	-	-	-	PK3F3	PK2F3	PK1F3	PK0F3
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	PK3F3	R/W	0: PORT 1: TB6IN1
2	PK2F3	R/W	0: PORT 1: TB6IN0
1	PK1F3	R/W	0: PORT 1: TB1IN1
0	PK0F3	R/W	0: PORT

		1: TB1IN0
--	--	-----------



10.2.11.6 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	-	-	-	-	PK3UP	PK2UP	PK1UP	PK0UP
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-4	-	R	读作 0。
3-0	PK3UP-PK0UP	R/W	输出 0: 禁用 1: 启用

10.2.11.7 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	-	-	-	-	PK3IE	PK2IE	PK1IE	PK0IE
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-4	-	R	读作 0。
3-0	PK3IE-PK0IE	R/W	输出 0: 禁用 1: 启用

10.3 端口方块图

10.3.1 端口类型

端口分类如下。请参考下列页面中各端口类型的方块图。

图中虚线指"端口方块图"所述等效电路部分。

表 10-5 功能列表

类型	GP 端口	功能	模拟	上拉	下拉	可编程开漏	注
FT1	I/O	I/O	-	R	-	o	
FT2	I/O	I/O	-	NoR	NoR	o	由启动信号触发的功能输出
FT3	I/O	I/O	-	R	-	o	由启动信号触发的功能输出
FT4	I/O	输入(int)	-	R	-	o	配置有噪声滤波器
FT5	I/O	输入	o	R	-	-	
FT6	输出	输出	-	NoR	-	o	$\overline{\text{BOOT}}$ 复位期间启动输入

int: 中断输入

-: 不存在

o: 存在

R: 复位时强制禁用

NoR: 不受复位影响

10.3.2 类型 FT1

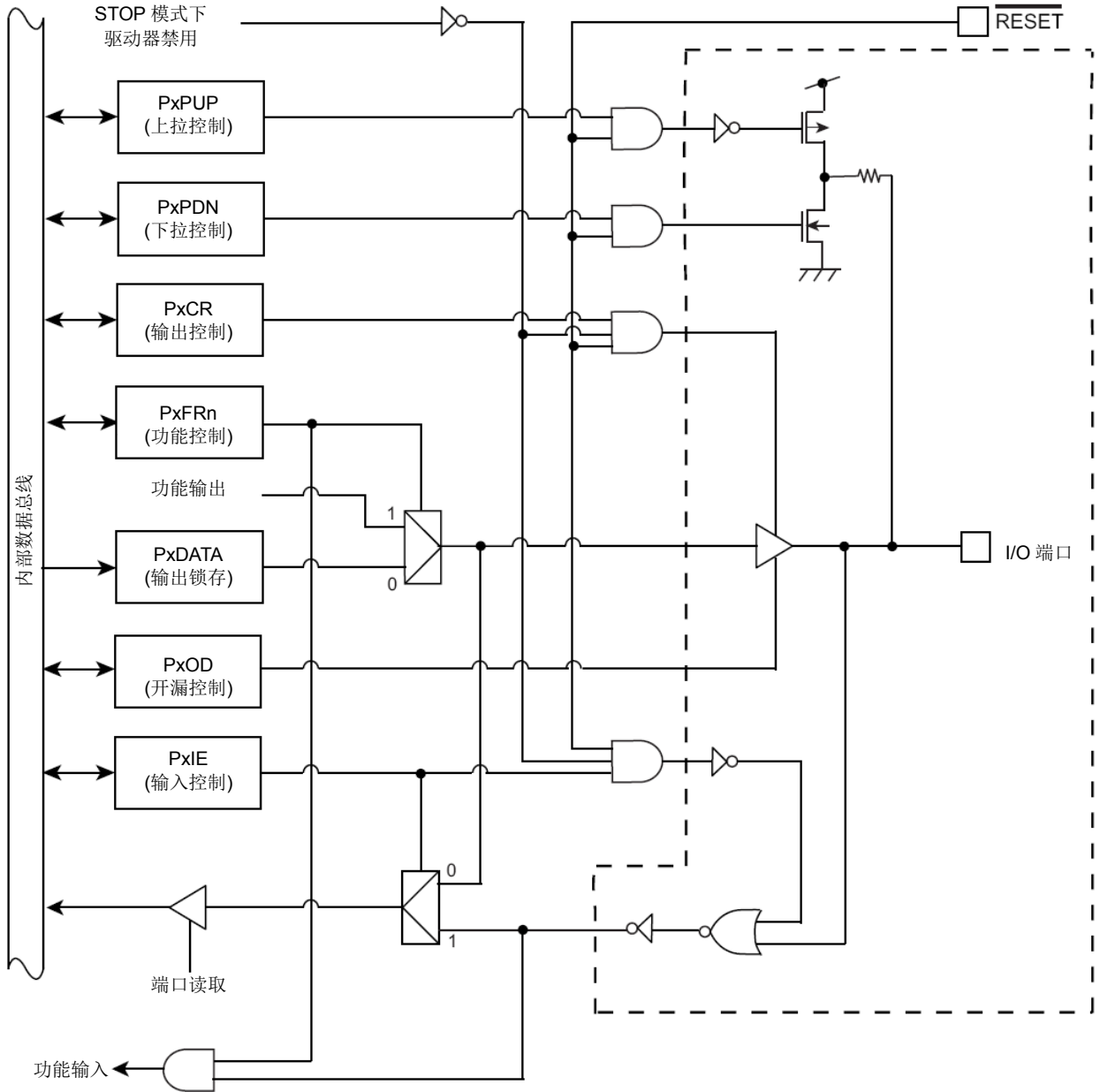


图 10-1 端口类型 FT1

10.3.3 类型 FT2

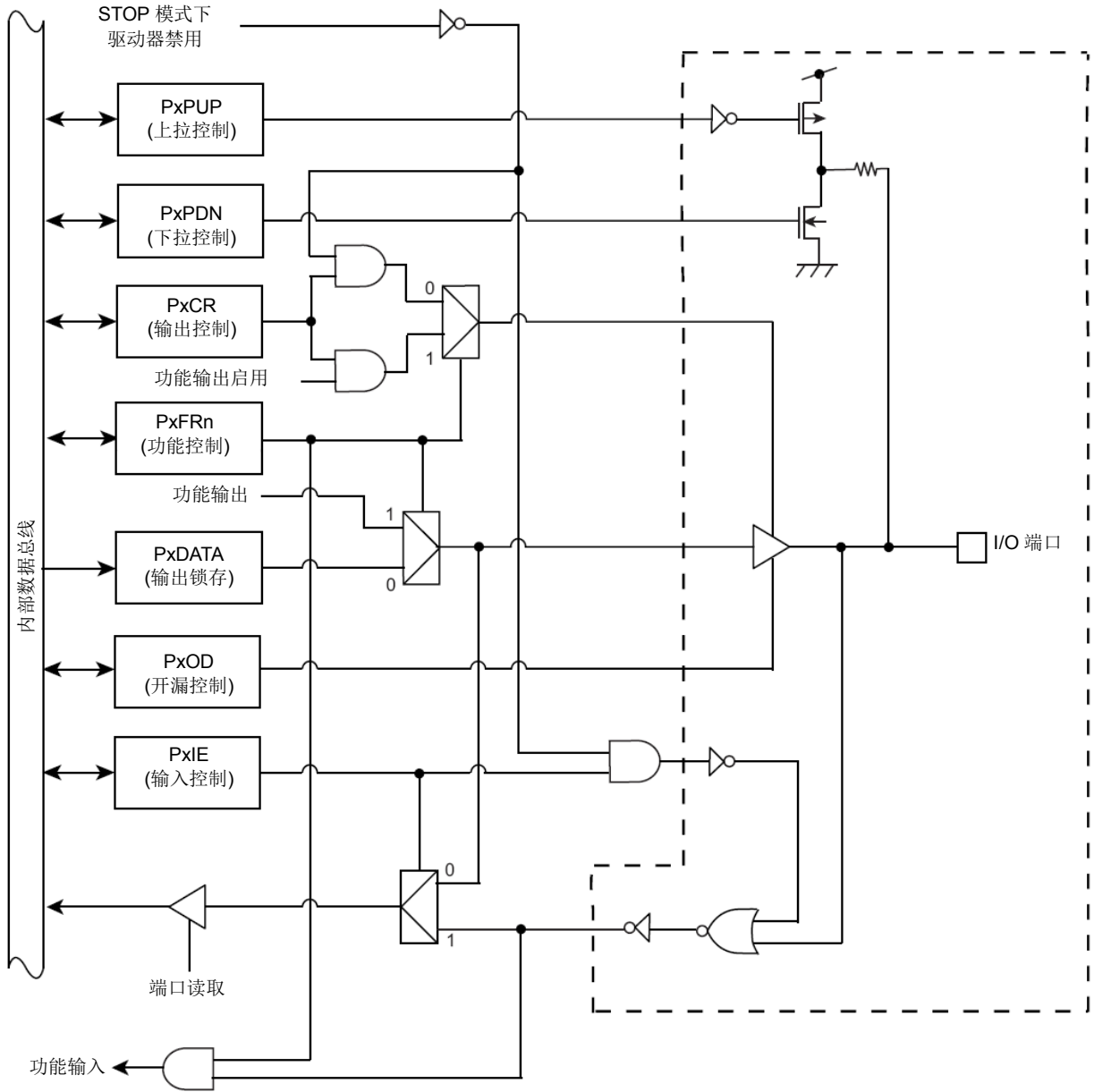


图 10-2 端口类型 FT2

10.3.4 类型 FT3

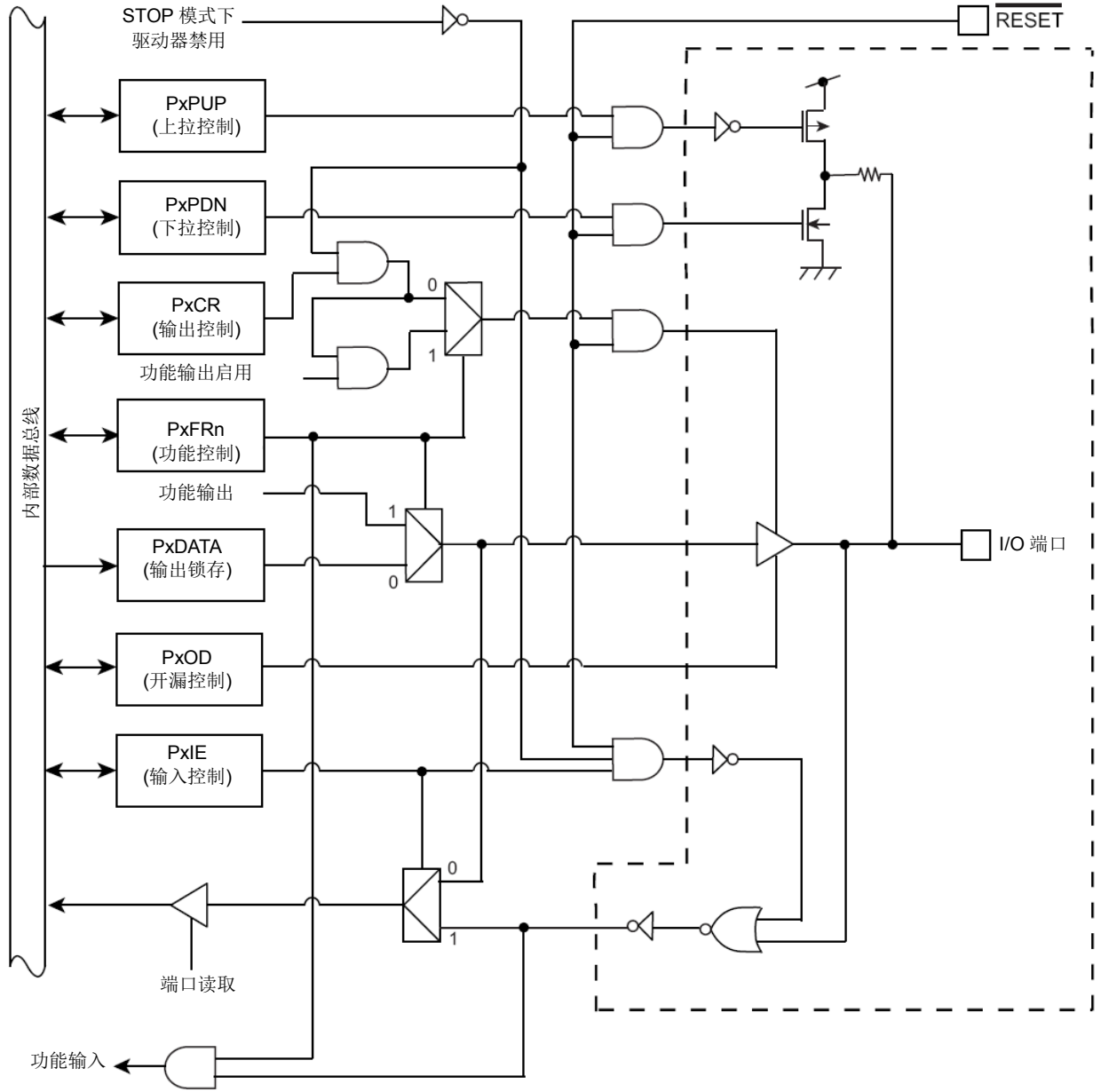


图 10-3 端口类型 FT3

10.3.5 类型 FT4

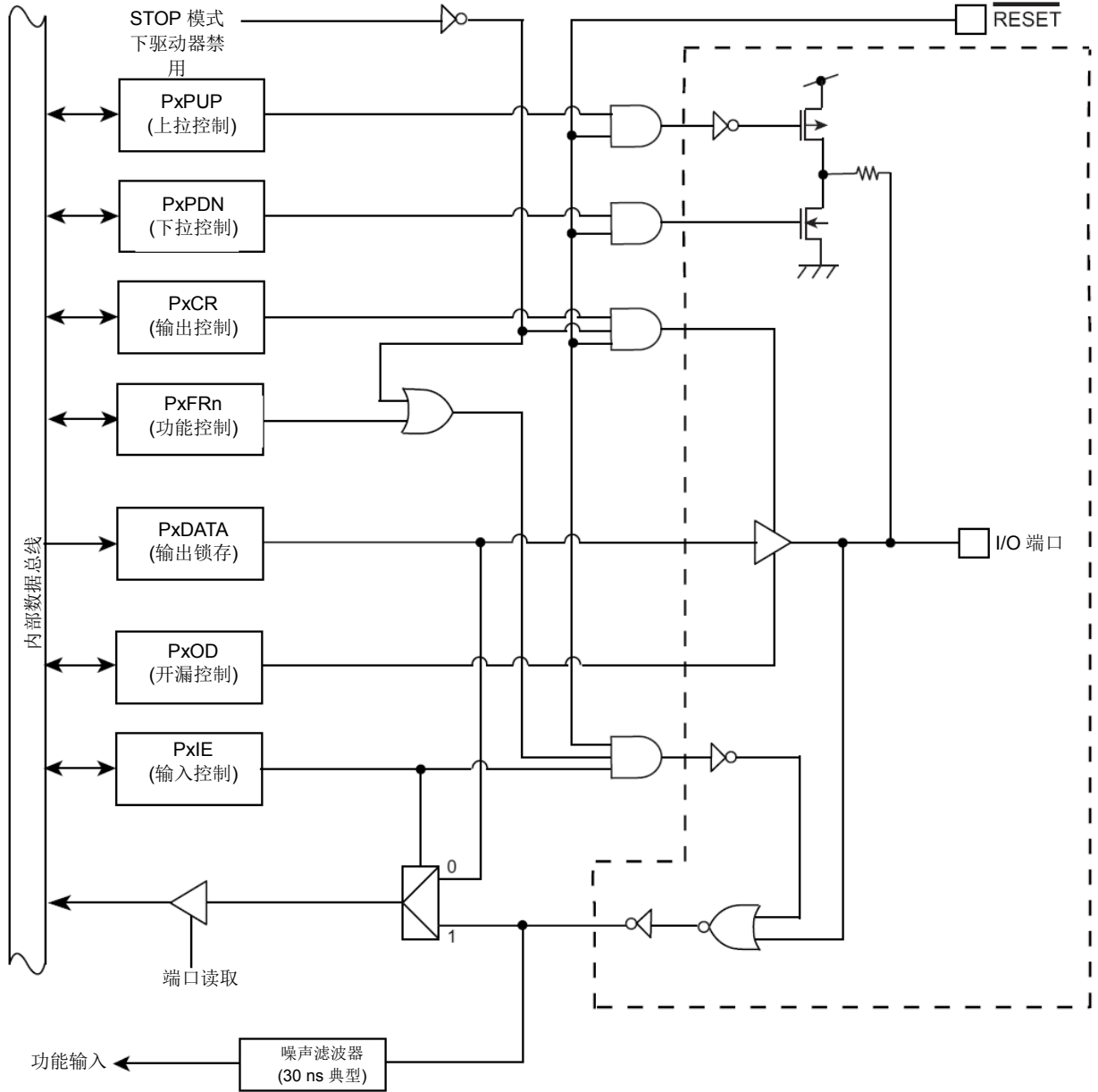


图 10-4 端口类型 FT4

10.3.6 类型 FT5

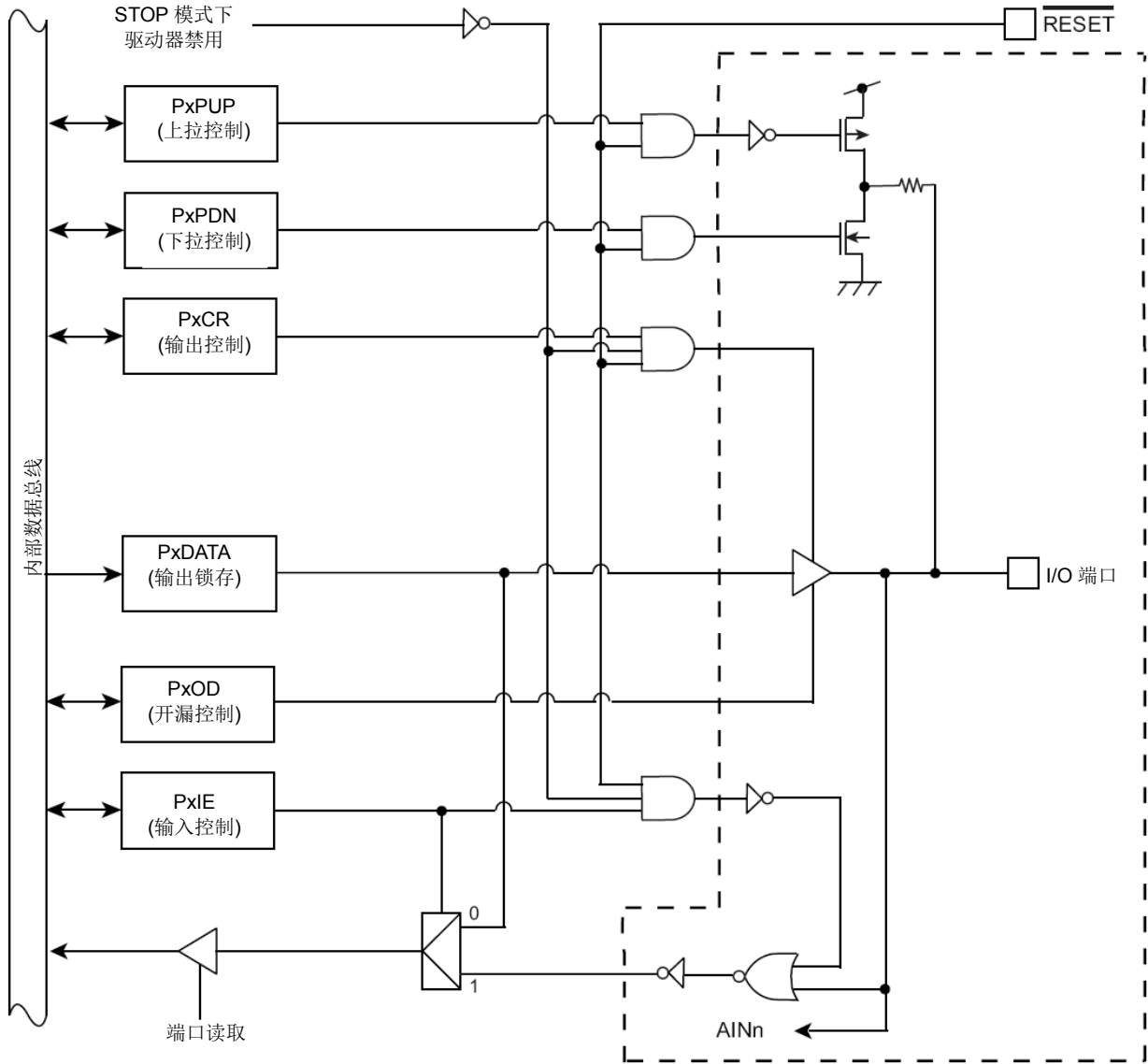


图 10-5 端口类型 FT5

10.3.7 类型 FT6

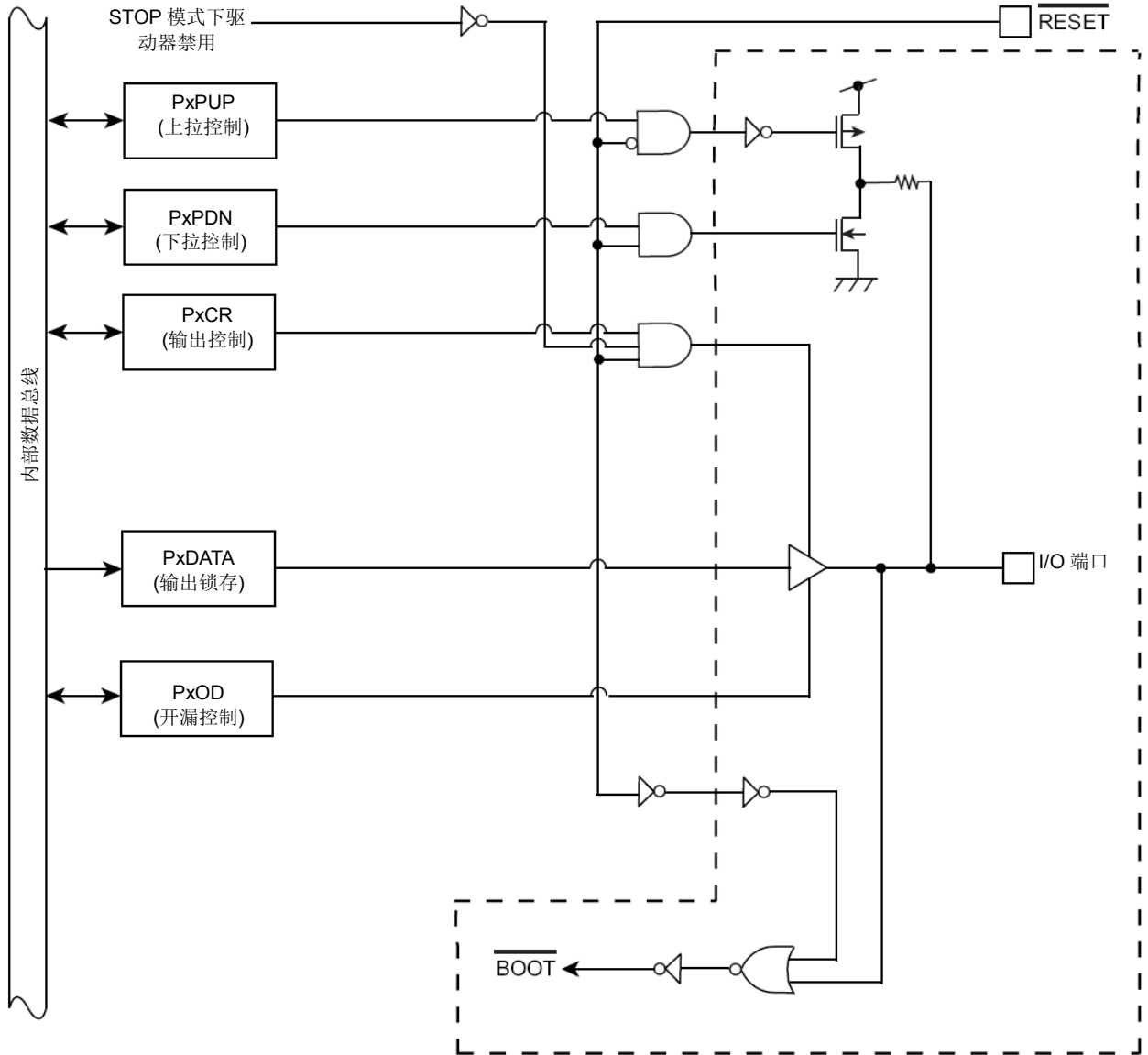


图 10-6 端口类型 FT6

10.4 附录 (端口设置列表)

下表说明了各功能的寄存器设置。

对于所有寄存器设置，应将"复位后"字段内不存在[0]的端口的初始化设置为"0"。

"X"位的设置可任意指定。

10.4.1 端口 A 设置

表 10-6 端口设置列表(端口 A)

引脚	端口类型	功能	复位后	PACR	PAOD	PAPUP	PAIE
PA0	FT1	输入端口		0	x	x	1
		输出端口		1	x	x	0
PA1	FT1	输入端口		0	x	x	1
		输出端口		1	x	x	0
PA2	FT1	输入端口		0	x	x	1
		输出端口		1	x	x	0
PA3	FT1	输入端口		0	x	x	1
		输出端口		1	x	x	0
PA4	FT1	输入端口		0	x	x	1
		输出端口		1	x	x	0
PA5	FT1	输入端口		0	x	x	1
		输出端口		1	x	x	0
PA6	FT1	输入端口		0	x	x	1
		输出端口		1	x	x	0
PA7	FT1	输入端口		0	x	x	1
		输出端口		1	x	x	0

10.4.2 端口 B 设置

表 10-7 端口设置列表(端口 B)

引脚	端口类型	功能	复位后	PBCR	PBOD	PBPUP	PBIE
PB0	FT1	输入端口		0	x	x	1
		输出端口		1	x	x	0
PB1	FT1	输入端口		0	x	x	1
		输出端口		1	x	x	0
PB2	FT1	输入端口		0	x	x	1
		输出端口		1	x	x	0
PB3	FT1	输入端口		0	x	x	1
		输出端口		1	x	x	0
PB4	FT1	输入端口		0	x	x	1
		输出端口		1	x	x	0
PB5	FT1	输入端口		0	x	x	1
		输出端口		1	x	x	0
PB6	FT1	输入端口		0	x	x	1
		输出端口		1	x	x	0
PB7	FT1	输入端口		0	x	x	1
		输出端口		1	x	x	0

10.4.3 端口 C 设置

表 10-8 端口设置列表(端口 C)

引脚	端口类型	功能	复位后	PCCR	PCFR1	PCFR3	PCFR4	PCOD	PCPUP	PCIE
PC0	FT1	输入端口		0	0	0	0	x	x	1
		输出端口		1	0	0	0	x	x	0
	FT1	TXD1(输出)		1	1	0	0	x	x	0
		TB2IN0(输入)		0	0	1	0	x	x	1
PC1	FT1	输入端口		0	0	0	0	x	x	1
		输出端口		1	0	0	0	x	x	0
	FT1	RXD1(输入)		0	1	0	0	x	x	1
		TB2IN1(输入)		0	0	1	0	x	x	1
PC2	FT1	输入端口		0	0	0	0	x	x	1
		输出端口		1	0	0	0	x	x	0
	FT1	SCLK1(输入)		0	1	0	0	x	x	1
		SCLK1(输出)		1	1	0	0	x	x	0
		TB0OUT(输出)		1	0	1	0	x	x	0
		$\overline{\text{CTS1}}$ (输入)		0	0	0	1	x	x	1

10.4.4 端口 D 设置

表 10-9 端口设置列表(端口 D)

引脚	端口类型	功能	复位后	PDCR	PDFR3	PDOD	PDPUP	PDIE
PD0	FT1	输入端口		0	0	x	x	1
		输出端口		1	0	x	x	0
	FT1	TB7OUT(输出)		1	1	x	x	0
PD1	FT1	输入端口		0	0	x	x	1
		输出端口		1	0	x	x	0
	FT1	TB8OUT(输出)		1	1	x	x	0
PD2	FT1	输入端口		0	0	x	x	1
		输出端口		1	0	x	x	0
	FT1	TB9OUT(输出)		1	1	x	x	0
PD3	FT1	输入端口		0	0	x	x	1
		输出端口		1	0	x	x	0
	FT1	$\overline{\text{ADTRG}}$ (输入)		0	1	x	x	1
PD4	FT1	输入端口		0	0	x	x	1
		输出端口		1	0	x	x	0
PD5	FT1	输入端口		0	0	x	x	1
		输出端口		1	0	x	x	0
PD6	FT1	输入端口		0	0	x	x	1
		输出端口		1	0	x	x	0
PD7	FT1	输入端口		0	0	x	x	1
		输出端口		1	0	x	x	0
	FT1	SCOUT (输出)		1	1	x	x	0

10.4.5 端口 E 设置

表 10-10 端口设置列表(端口 E)

引脚	端口类型	功能	复位后	PECR	PEFR1	PEFR3	PEFR4	PEOD	PEPUP	PEIE
PE0	FT1	输入端口		0	0	0	0	x	x	1
		输出端口		1	0	0	0	x	x	0
	FT1	TXD0 (输出)		1	1	0	0	x	x	0
PE1	FT1	输入端口		0	0	0	0	x	x	1
		输出端口		1	0	0	0	x	x	0
	FT1	RXD0 (输入)		0	1	0	0	x	x	1
PE2	FT1	输入端口		0	0	0	0	x	x	1
		输出端口		1	0	0	0	x	x	0
	FT1	SCLK0 (输入)		0	1	0	0	x	x	1
		SCLK0 (输出)		1	1	0	0	x	x	0
		TB2OUT (输出)		1	0	1	0	x	x	0
	$\overline{\text{CTS0}}$ (输入)		0	0	0	1	x	x	1	
PE3	FT1	输入端口		0	0	0	0	x	x	1
		输出端口		1	0	0	0	x	x	0
	FT4	INT5(输入)		0	1	0	0	x	x	1
	FT1	TB3OUT (输出)		1	0	1	0	x	x	0
PE4	FT1	输入端口		0	0	0	0	x	x	1
		输出端口		1	0	0	0	x	x	0
	FT1	SO1 (输出)		1	1	0	0	x	x	0
		SDA (I/O)		1	1	0	0	1	x	1
PE5	FT1	输入端口		0	0	0	0	x	x	1
		输出端口		1	0	0	0	x	x	0
	FT1	SCL1/SI1 (输入)		0	1	0	0	1	x	1
		SCL1/SI1 (输出)		1	1	0	0	1	x	0
PE6	FT1	输入端口		0	0	0	0	x	x	1
		输出端口		1	0	0	0	x	x	0
	FT1	SCK1(输入)		0	1	0	0	x	x	1
		SCK1 (输出)		1	1	0	0	x	x	1
PE7	FT1	输入端口		0	0	0	0	x	x	1
		输出端口		1	0	0	0	x	x	0
	FT4	INT4 (输入)		0	1	0	0	x	x	1

10.4.6 端口 F 设置

表 10-11 端口设置列表(端口 F)

引脚	端口类型	功能	复位后	PFCR	PFFR2	PFFR3	PFOD	PFPU	PFIE
PF0	FT6	输出端口		1	0	0	x	1	0
	FT1	TB6OUT(输出)		1	0	1	x	1	0
PF1	FT1	输入端口		0	0	0	x	x	1
		输出端口		1	0	0	x	x	0
PF2	FT1	输入端口		0	0	0	x	x	1
		输出端口		1	0	0	x	x	0
PF3	FT1	输入端口		0	0	0	x	x	1
		输出端口		1	0	0	x	x	0
PF4	FT1	输入端口		0	0	0	x	x	1
		输出端口		1	0	0	x	x	0
	FT4	INT6(输入)		0	1	0	x	x	1
	FT1	TB5IN0(输入)		0	0	1	x	x	1
PF5	FT1	输入端口		0	0	0	x	x	1
		输出端口		1	0	0	x	x	0
	FT4	INT7(输入)		0	1	0	x	x	1
PF6	FT1	输入端口		0	0	0	x	x	1
		输出端口		1	0	0	x	x	0
PF7	FT1	输入端口		0	0	0	x	x	1
		输出端口		1	0	0	x	x	0

注：PF0 输入与上拉启用并充当 $\overline{\text{BOOT}}$ 输入引脚时， $\overline{\text{RESET}}$ 则处于“低”状态

10.4.7 端口 G 设置

表 10-12 端口设置列表(端口 G)

引脚	端口类型	功能	复位后	PGCR	PGFR1	PGFR3	PGFR4	PGOD	PGPUP	PGIE
PG0	FT1	输入端口		0	0	0	0	x	x	1
		输出端口		1	0	0	0	x	x	0
	FT1	SO0 (输出)		1	1	0	0	x	x	1
	FT1	SDA0(I/O)		1	1	0	0	1	x	1
PG1	FT1	输入端口		0	0	0	0	x	x	1
		输出端口		1	0	0	0	x	x	0
	FT1	SI0(输入)		0	1	0	0	x	x	1
		SCL0(I/O)		1	1	0	0	1	x	1
		TB3IN0 (输入)		0	0	1	0	x	x	1
PG2	FT1	输入端口		0	0	0	0	x	x	1
		输出端口		1	0	0	0	x	x	0
	FT1	SCK0(输入)		0	1	0	0	x	x	1
		SCK0(输出)		1	1	0	0	x	x	0
		TB3IN1(输入)		0	0	1	0	x	x	1
PG3	FT1	输入端口		0	0	0	0	x	x	1
		输出端口		1	0	0	0	x	x	0
	FT4	INT0 (输入)		0	1	0	0	x	x	1
	FT1	TB4IN0(输入)		0	0	1	0	x	x	1
PG4	FT1	输入端口		0	0	0	0	x	x	1
		输出端口		1	0	0	0	x	x	0
	FT1	TB4IN1(输入)		0	0	1	0	x	x	1
PG5	FT1	输入端口		0	0	0	0	x	x	1
		输出端口		1	0	0	0	x	x	0
	FT4	INT1(输入)		0	1	0	0	x	x	1
	FT1	USBPON(输入)		0	0	0	1	x	x	1

注：PG5 为 5V 额定输入引脚。但是可编程上拉电压高达 DVDD3A 电压电平，因此 5V 电压额定输入引脚的上拉应通过外部上拉电阻进行设计。

10.4.8 端口 H 设置

表 10-13 端口设置列表(端口 H)

引脚	端口类型	功能	复位后	PHCR	PHFR1	PHFR3	PHOD	PHPUP	PHIE
PH0	FT1	输入端口		0	0	0	x	x	1
		输出端口		1	0	0	x	x	0
	FT1	TRACEDATA2(输出)		1	1	0	0	0	0
PH1	FT1	输入端口		0	0	0	x	x	1
		输出端口		1	0	0	x	x	0
	FT1	TRACEDATA3(输出)		0	1	0	0	0	0
PH2	FT1	输入端口		0	0	0	x	x	1
		输出端口		1	0	0	x	x	0
	FT1	TB4OUT (输出)		1	0	1	x	x	0
PH3	FT1	输入端口		0	0	0	x	x	1
		输出端口		1	0	0	x	x	0
	FT1	TB5OUT(输出)		1	0	1	x	x	0
PH4	FT1	输入端口		0	0	0	x	x	1
		输出端口		1	0	0	x	x	0
	FT4	INT8 (输入)		0	0	1	x	x	1

10.4.9 端口 I 设置

表 10-14 端口设置列表 (端口 I)

引脚	端口类型	功能	复位后	PICR	PIFR1	PIOD	PIPUP	PIPDN	PIIE
PI0	FT1	输入端口		0	0	x	x	x	1
		输出端口		1	0	x	x	x	0
	FT1	TRACEDATA1 (输出)		1	1	0	0	0	0
PI1	FT1	输入端口		0	0	x	x	x	1
		输出端口		1	0	x	x	x	0
	FT1	TRACEDATA0 (输出)		1	1	0	0	0	0
PI2	FT1	输入端口		0	0	x	x	x	1
		输出端口		1	0	x	x	x	0
	FT1	TRACECLK (输出)		1	1	0	0	0	0
PI3	FT1	输入端口		0	0	x	x	x	1
		输出端口		1	0	x	x	x	0
	FT2	TCK (输入)/ SWCLK (输入)	◦	0	1	0	0	1	1
PI4	FT1	输入端口		0	0	x	x	x	1
		输出端口		1	0	x	x	x	0
	FT2	TMS(输入)/ SWDIO (输入/输出)	◦	1	1	0	1	0	1
PI5	FT1	输入端口		0	0	x	x	x	1
		输出端口		1	0	x	x	x	0
	FT2	TDO(输出)/ SWV (输出)	◦	1	1	0	0	0	0
PI6	FT1	输入端口		0	0	x	1	x	1
		输出端口		1	0	x	1	x	0
	FT2	TDI (输入)	◦	0	1	0	1	0	1
PI7	FT1	输入端口		0	0	x	x	x	1
		输出端口		1	0	x	x	x	0
	FT2	$\overline{\text{TRST}}$ (输入)	◦	0	1	0	1	0	1

10.4.10 端口 J 设置

表 10-15 端口设置列表(端口 J)

引脚	端口类型	功能	复位后	PJCR	PJFR2	PJFR3	PJPUP	PJIE
PJ0	FT1	输入端口		0	0	0	x	1
		输出端口		1	0	0	x	0
PJ1	FT1	输入端口		0	0	0	x	1
		输出端口		1	0	0	x	0
PJ2	FT1	输入端口		0	0	0	x	1
		输出端口		1	0	0	x	0
PJ3	FT1	输入端口		0	0	0	x	1
		输出端口		1	0	0	x	0
PJ4	FT1	输入端口		0	0	0	x	1
		输出端口		1	0	0	x	0
PJ5	FT1	输入端口		0	0	0	x	1
		输出端口		1	0	0	x	0
PJ6	FT1	输入端口		0	0	0	x	1
		输出端口		1	0	0	x	0
	FT1	TB0IN0(输入)		0	0	1	x	1
PJ7	FT1	输入端口		0	0	0	x	1
		输出端口		1	0	0	x	0
	FT4	INT9 (输入)		0	1	0	x	1
	FT1	TB0IN1(输入)		0	0	1	x	1

10.4.11 端口 K 设置

表 10-16 端口设置列表(端口 K)

引脚	端口类型	功能	复位后	PKCR	PKFR2	PKFR3	PKPUP	PKIE
PK0	FT1	输入端口		0	0	0	x	1
		输出端口		1	0	0	x	0
	FT4	INT2 (输入)		0	1	0	x	1
	FT1	TB1IN0 (输入)		0	0	1	x	1
PK1	FT1	输入端口		0	0	0	x	1
		输出端口		1	0	0	x	0
	FT4	INT3 (输入)		0	1	0	x	1
	FT1	TB1IN1(输入)		0	0	1	x	1
PK2	FT1	输入端口		0	0	0	x	1
		输出端口		1	0	0	x	0
	FT1	TB6IN0(输入)		0	0	1	x	1
PK3	FT1	输入端口		0	0	0	x	1
		输出端口		1	0	0	x	0
	FT1	TB6IN1(输入)		0	0	1	x	1

11. DMA 控制器 (DMAC)

11.1 概述

下表列出了其主要功能。

表 11-1 DMA 控制器功能(1 单元)

项目	功能		说明
通道数	2ch		-
DMA 请求数	16		-
DMA 启动触发器	硬件启动		以外设电路的 DMA 请求开始
	软件启动		以写入 DMACxSoftBReq 寄存器开始。
总线主控	32 位 x 1(AHB)		-
优先级	高: CH0 低: CH1		固定
FIFO	4 个字符 x 2ch (1 个字符= 32 位)		-
总线宽度	8/16/32 位		可单独设置用于传输源和目的地。
突发量	1/4/8/16/32/64/128/256		-
传输数	高达 4095		-
地址	传输源地址	增加 未增加	可以说明来源和目的地址是否应增加或不 应增加。 (不支持地址包。)
	传输目的地址	增加 未增加	
字节存储次序	支持小端字节序。		-
传输方式	外设到存储器 存储器到外设 存储器到存储器 外设到外设		在选用"存储器到存储器"传输方式时,针对 DMA 启动, 不支持硬件启动。 更多信息, 参考 DMACxCnConfiguration。 在选用"外设到外设"时, 可把特殊外设指定 为来源或目的地。有关指定的外设, 参考 "11.4.1 以外设到外设传输支持的外设功能 "。
中断功能	传输末端中断(INTDMACxTC) 错误中断(INTDMACxERR)		-
特殊功能	分散/聚集功能		-

11.2 DMA 传输方式

表 11-2 DMA 传输方式

序号	DMA 传输方式	电路产生 DMA 请求	DMA 请求方式	说明									
1	存储器到外设	外设 (目标)	突发请求	如为 1 个字符传输时, 针对 DMA 控制器突发量, 设置为"1"。									
2	外设到存储器	外设 (源)	突发请求/ 单一请求	如传输数据量不是突发量的整数倍, 则可使用突发请求和单一请求。如传输数据量大于或等于突发量, 则忽略单一请求, 使用突发请求。如果传输数据量低于突发量, 则使用单一传输。									
3	存储器到存储器	DMAC	无	在无 DMAC 请求的情况下, 启用 DMAC, 启动数据传输。 (选择存储器-存储器模式, 把 DMACx Cn 配置<E> 设置为 "1") 在所有传输数据被全部传输时或 DMAC 通道被禁用时, DMAC 停止。									
4	外设到外设	外设 (源)	突发请求/ 单一请求	<table border="1"> <thead> <tr> <th>传输大小</th> <th>来源</th> <th>目标</th> </tr> </thead> <tbody> <tr> <td>(1)是突发量的整数倍</td> <td>突发请求</td> <td>突发请求</td> </tr> <tr> <td>(2)不是突发量的整数倍</td> <td>突发请求/ 单一请求</td> <td>-</td> </tr> </tbody> </table>	传输大小	来源	目标	(1)是突发量的整数倍	突发请求	突发请求	(2)不是突发量的整数倍	突发请求/ 单一请求	-
		传输大小	来源		目标								
(1)是突发量的整数倍	突发请求	突发请求											
(2)不是突发量的整数倍	突发请求/ 单一请求	-											
外设 (目标)	突发请求												

注: 当以存储器-存储器传输方式传输大量数据时, 建议使用低优先级通道。如使用较低优先级通道, 在较低优先级通道进行传输时, 可启动较高优先级通道。如使用较高优先级通道, 在较高优先级通道进行传输时, 不可启动较低优先级通道。

11.3 方块图

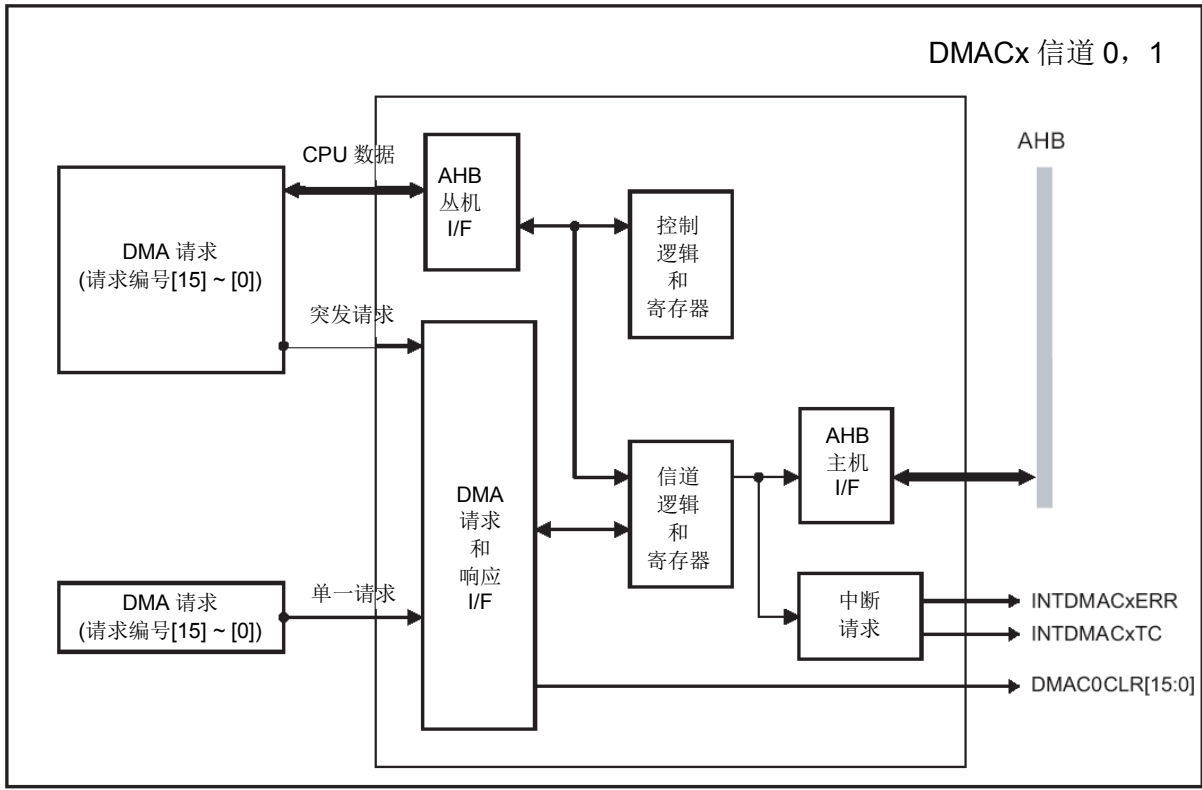


图 11-1 DMAC 方块图

11.4 TMPM365FYXBG 产品信息

11.4.1 以外设到外设传输方式支持的外设功能

以外设到外设传输方式支持的外设功能(寄存器)如下所示。

源	目标
外设寄存器	SCxBUF (x=0~1)
	TBxREG0~1 (x=0~9)
	TBxCP0~1 (x=0~9)
SCxBUF (x=0~1)	外设寄存器
TBxREG0~1 (x=0~9)	
TBxCP0~1 (x=0~9)	

11.4.2 DMA 请求

对应每个 DMA 请求编号的 DMA 请求如下所示。

表 11-3 DMA 请求因素

DMA 请求编号	相应外设	
	ch0,ch1	
	突发请求	单一请求
0	SIO0/UART0 接收	-
1	SIO0/UART0 传输	-
2	SIO1/UART1 接收	-
3	SIO1/UART1 传输	-
4	TMRB8 比较匹配	-
5	TMRB9 比较匹配	-
6	TMRB0 输入捕捉 0	-
7	TMRB4 输入捕捉 0	-
8	TMRB4 输入捕捉 1	-
9	TMRB5 输入捕捉 0	-
10	TMRB5 输入捕捉 1	-
11	正常 AD 转换末端	-
12	I2C0 / SIO0 接收(注)	-
13	I2C0 / SIO0 传输(注)	-
14	I2C1 / SIO1 接收(注)	-
15	I2C1 / SIO1 传输(注)	-

注： I2C 模式中可使用 DMAC，但不得在 SIO 模式中使用。

11.4.3 中断请求

传输完成中断	错误中断
INTDMAC0TC	INTDMAC0ERR

11.4.4 寄存器基址

基址
0x4000_0000

11.5 寄存器说明

11.5.1 DMAC 寄存器列表

每个寄存器的功能和地址如下所示：

寄存器名称		地址(基+)
DMAC 请求状态寄存器	DMACxIntStaus	0x0000
DMAC 中断端子计数状态寄存器	DMACxIntTCStatus	0x0004
DMAC 中断端子计数清除寄存器	DMACxIntTCClear	0x0008
DMAC 中断错误状态寄存器	DMACxIntErrorStatus	0x000C
DMAC 中断错误清除寄存器	DMACxIntErrClr	0x0010
DMAC 写后读中断端子计数状态寄存器	DMACxRawIntTCStatus	0x0014
DMAC 写后读错误中断状态寄存器	DMACxRawIntErrorStatus	0x0018
DMAC 启用通道寄存器	DMACxEnbldChns	0x001C
DMAC 软件突发请求寄存器	DMACxSoftBReq	0x0020
DMAC 软件单一请求寄存器	DMACxSoftSReq	0x0024
保留	-	0x0028
保留	-	0x002C
DMAC 配置寄存器	DMACxConfiguration	0x0030
保留	-	0x0034
DMAC 通道 0 源地址寄存器	DMACxC0SrcAddr	0x0100
DMAC 通道 0 目的地址寄存器	DMACxC0DestAddr	0x0104
DMAC 通道 0 链表项目寄存器	DMACxC0LLI	0x0108
DMAC 通道 0 控制寄存器	DMACxC0Control	0x010C
DMAC 通道 0 配置寄存器	DMACxC0Configuration	0x0110
DMAC 通道 1 源地址寄存器	DMACxC1SrcAddr	0x0120
DMAC 通道 1 目的地址寄存器	DMACxC1DestAddr	0x0124
DMAC 通道 1 链表项目寄存器	DMACxC1LLI	0x0128
DMAC 通道 1 控制寄存器	DMACxC1Control	0x012C
DMAC 通道 1 配置寄存器	DMACxC1Configuration	0x0130

注 1：通过采用字符(32 位)读出和字符写入访问寄存器。

注 2：禁止进入"保留" 区域。

注 3：对于为每个通道准备的寄存器，如通道结构是相同的，单位编号标示为"x"，通道编号标示为"n"。

注 4：当分配各个通道的寄存器是在未分配各个通道的寄存器被写入之后读出的，应在指令之间插入机器周期，或对未分配各个通道的寄存器读出两次。

11.5.2 DMACxIntStatus (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	-	-	写作 0。
1	IntStatus1	R	DMAC 通道 1 传输末端中断状态。 0: 未请求中断 1: 请求中断 在通过传输末端中断启用寄存器和错误中断启用寄存器后, DMAC 中断生成的状态。在出现传输错误或计数器完成计数时, 请求中断。
0	IntStatus0	R	DMAC 通道 0 中断生成状态。 0: 未请求中断 1: 请求中断 在通过传输末端中断启用寄存器和错误中断启用寄存器后, DMAC 中断生成的状态。在出现传输错误或计数器完成计数时, 请求中断。

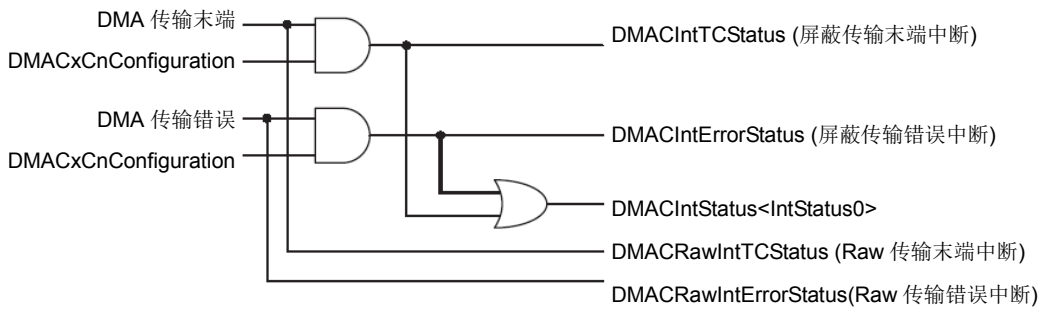


图 11-2 中断相关方块图



11.5.3 DMACxIntTCStatus (DMAC 中断端子计数状态寄存器)

	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	-	-	-	-	-	-	IntTCStatus 1	IntTCStatus 0
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 0。
1	IntTCStatus1	R	DMAC 通道 1 传输末端中断状态。 0: 未请求中断 1: 请求中断 后启用传输末端中断生成状态。
0	IntTCStatus0	R	DMAC 通道 0 传输末端中断状态。 0: 未请求中断 1: 请求中断 后启用传输末端中断生成状态。

11.5.4 DMACxIntTCClear(DMAC 中断端子计数清除寄存器)

	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	-	-	-	-	-	-	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 0。
1	IntTCClear1	W	清除 DMAC 通道 1 传输末端中断。 0: 什么也不做 1: 清除 当写入"1"时, DMACxIntTC 状态<IntTCStatus1>将被清除。
0	IntTCClear0	W	清除 DMAC 通道 0 传输末端中断。 0: 什么也不做 1: 清除 在写入"1"时, DMACxIntTCStatus<IntTCStatus0>将被清除。

11.5.5 DMACxIntErrorStatus (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	IntErrStatus0
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 0。
1	IntErrStatus1	R	DMAC 通道 1 错误中断生成状态。 0: 未请求中断 1: 请求中断 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: 未请求中断 1: 请求中断 启用后, 显示错误中断状态。

11.5.6 DMACxIntErrClr(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 0。
1	IntErrClr1	W	清除 DMAC 通道 1 传输末端中断。 0: 什么也不做 1: 清除 在写入"1"时, DMACxIntErrorStatus<IntErrStatus1>被清除
0	IntErrClr0	W	清除 DMAC 通道 0 传输末端中断。 0: 什么也不做 1: 清除 在写入"1"时, DMACxIntErrorStatus<IntErrStatus0>被清除。

11.5.7 DMACxRawIntTCStatus (DMAC Raw 中断端子计数状态寄存器)

	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	-	-	-	-	-	-	RawIntTCS 1	RawIntTCS 0
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 0。
1	RawIntTCS1	R	DMAC 通道 1 启用前传输末端中断生成状态 0: 未请求中断 1: 请求中断
0	RawIntTCS0	R	DMAC 通道 0 启用前传输末端中断生成状态 0: 未请求中断 1: 请求中断

11.5.8 DMACxRawIntErrorStatus (DMAC Raw 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	-	-	-	-	-	-	RawIntErrS 1	RawIntErrS 0
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 0。
1	RawIntErrS1	R	DMAC 通道 1 启用前错误中断状态。 0: 未请求中断 1: 请求中断
0	RawIntErrS0	R	DMAC 通道 0 启用前错误中断状态。 0: 未请求中断 1: 请求中断

11.5.9 DMACxEnbldChns (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	-	-	-	-	-	-	EnabledCH 1	EnabledCH 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
31-2	-	-	写作 0。
1	EnabledCH1	R	DMAC 通道 1 启用前错误中断状态。 0: 禁用 1: 启用 在完成 DMACxKn 控制寄存器(值变为 0)中的所有传输总次数后, 该标志被清除。
0	EnabledCH0	R	DMAC 通道 0 启用前错误中断状态。 0: 禁用 1: 启用 在完成 DMACxKn 控制寄存器(值变为 0)中的所有传输总次数后, 该标志被清除。 (零), 该标志被清除

11.5.10 DMACxSoftBReq (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 0。
15	SoftBReq15	R/W	由软件执行的 DMA 突发请求 (请求编号[15]) 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	由软件执行的 DMA 突发请求(请求编号[14]) 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	由软件执行的 DMA 突发请求(请求编号[13]) 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	由软件执行的 DMA 突发请求(请求编号[12]) 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	由软件执行的 DMA 突发请求(请求编号[11]) 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	由软件执行的 DMA 突发请求(请求编号[10]) 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	由软件执行的 DMA 突发请求(请求编号[9]) 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	由软件执行的 DMA 突发请求(请求编号[8]) 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	由软件执行的 DMA 突发请求(请求编号[7]) 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	由软件执行的 DMA 突发请求(请求编号[6]) 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	由软件执行的 DMA 突发请求(请求编号[5]) 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	由软件执行的 DMA 突发请求(请求编号[4]) 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	由软件执行的 DMA 突发请求(请求编号[3]) 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	由软件执行的 DMA 突发请求(请求编号[2]) 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	由软件执行的 DMA 突发请求(请求编号[1]) 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	由软件执行的 DMA 突发请求(请求编号[0]) 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: 请勿同时通过软件和硬件执行 DMA 请求。

注 2: 有关 DMA 请求编号, 参考"11.4.2 DMA 请求"。清除"0"至与未执行突发请求的 DMA 请求编号相对应的位。

11.5.11 DMACxSoftSReq (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	SoftSReq 14	SoftSReq 13	SoftSReq 12	SoftSReq 11	SoftSReq 10	SoftSReq 9	SoftSReq 8
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	SoftSReq 7	SoftSReq 6	SoftSReq 5	SoftSReq 4	SoftSReq 3	SoftSReq 2	SoftSReq 1	SoftSReq 0
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-16	-	-	写作 0。
15	SoftSReq 15	R/W	由软件执行的 DMA 单一请求(请求编号[15]) 读取 : 0 : 停止 DMA 单一传输 1 : 运行 DMA 单一请求 写入 : 0 : 无效 1 : DMA 单一请求
14	SoftSReq 14	R/W	由软件执行的 DMA 单一请求(请求编号[14]) 读取 : 0 : 停止 DMA 单一传输 1 : 运行 DMA 单一请求 写入 : 0 : 无效 1 : DMA 单一请求
13	SoftSReq 13	R/W	由软件执行的 DMA 单一请求(请求编号[13]) 读取 : 0 : 停止 DMA 单一传输 1 : 运行 DMA 单一请求 写入 : 0 : 无效 1 : DMA 单一请求
12	SoftSReq 12	R/W	由软件执行的 DMA 单一请求(请求编号[12]) 读取 : 0 : 停止 DMA 单一传输 1 : 运行 DMA 单一请求 写入 : 0 : 无效 1 : DMA 单一请求
11	SoftSReq 11	R/W	由软件执行的 DMA 单一请求(请求编号[11]) 读取 : 0 : 停止 DMA 单一传输 1 : 运行 DMA 单一请求 写入 : 0 : 无效 1 : DMA 单一请求
10	SoftSReq 10	R/W	由软件执行的 DMA 单一请求(请求编号[10]) 读取 : 0 : 停止 DMA 单一传输 1 : 运行 DMA 单一请求 写入 : 0 : 无效 1 : DMA 单一请求
9	SoftSReq 9	R/W	由软件执行的 DMA 单一请求(请求编号[9]) 读取 : 0 : 停止 DMA 单一传输 1 : 运行 DMA 单一请求 写入 : 0 : 无效 1 : DMA 单一请求
8	SoftSReq 8	R/W	由软件执行的 DMA 单一请求(请求编号[8]) 读取 : 0 : 停止 DMA 单一传输 1 : 运行 DMA 单一请求 写入 : 0 : 无效 1 : DMA 单一请求

位	比特符号	类型	功能
7	SoftSReq 7	R/W	由软件执行的 DMA 单一请求(请求编号[7]) 读取 : 0 : 停止 DMA 单一传输 1 : 运行 DMA 单一请求 写入 : 0 : 无效 1 : DMA 单一请求
6	SoftSReq 6	R/W	由软件执行的 DMA 单一请求(请求编号[6]) 读取 : 0 : 停止 DMA 单一传输 1 : 运行 DMA 单一请求 写入 : 0 : 无效 1 : DMA 单一请求
5	SoftSReq 5	R/W	由软件执行的 DMA 单一请求(请求编号[5]) 读取 : 0 : 停止 DMA 单一传输 1 : 运行 DMA 单一请求 写入 : 0 : 无效 1 : DMA 单一请求
4	SoftSReq 4	R/W	由软件执行的 DMA 单一请求(请求编号[4]) 读取 : 0 : 停止 DMA 单一传输 1 : 运行 DMA 单一请求 写入 : 0 : 无效 1 : DMA 单一请求
3	SoftSReq 3	R/W	由软件执行的 DMA 单一请求(请求编号[3]) 读取 : 0 : 停止 DMA 单一传输 1 : 运行 DMA 单一请求 写入 : 0 : 无效 1 : DMA 单一请求
2	SoftSReq 2	R/W	由软件执行的 DMA 单一请求(请求编号[2]) 读取 : 0 : 停止 DMA 单一传输 1 : 运行 DMA 单一请求 写入 : 0 : 无效 1 : DMA 单一请求
1	SoftSReq 1	R/W	由软件执行的 DMA 单一请求(请求编号[1]) 读取 : 0 : 停止 DMA 单一传输 1 : 运行 DMA 单一请求 写入 : 0 : 无效 1 : DMA 单一请求
0	SoftSReq 0	R/W	由软件执行的 DMA 单一请求(请求编号[0]) 读取 : 0 : 停止 DMA 单一传输 1 : 运行 DMA 单一请求 写入 : 0 : 无效 1 : DMA 单一请求

注 1: 请勿同时通过软件和硬件执行 DMA 请求。

注 2: 有关 DMA 请求编号, 参考“11.4.2 DMA 请求”。清除“0”至与未执行突发请求的 DMA 请求编号相对应的位。

11.5.12 DMACxConfiguration (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 0。
1	M	R/W	写作 0。
0	E	R/W	DMA 电路控制 0: 停止 1: 操作 当电路停止时, DMA 电路寄存器不得写入或读取。在操作 DMA 时, 应要始终设置<E>="1"。

11.5.13 DMACxCnSrcAddr (DMAC 通道 x 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。 在设置前，确保对源地址和位宽度进行确认。 下文是对源地址位宽度进行设置的限制条件。	
			源地址位宽度 DMACxCnControl<Swidth[2:0]>	最低有效地址的设置
			000: 字节(8 位)	无限制
			001: 半字符(16 位)	设置为 2 的倍数,(0x0, 0x02, 0x4, 0x06, 0x8, 0xA, 0xC...)
			010: 字符(32 位)	设置为 4 的倍数,(0x0, 0x4, 0x8, 0xC...)

因为启用通道"n"(DMACxCnConfiguration<E>="1")更新了写入寄存器的数据，所以在启用通道之前要设置 DMACxCnSrcAddr。

当 DMA 在运行时，DMACxCnSrcAddr 寄存器内的数值连续发生变化，因此读取数值是不固定的。

在传输期间，请勿更新 DMACxCnSrcAddr。为了变更 DMACxCnSrcAddr，在变化前务必禁用通道"n"(DMACxCnConfiguration<E>="0")。

11.5.14 DMACxCnDestAddr (DMAC 通道 x 目的地址寄存器)

	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	<p>设置 DMA 传输目的地址。 在设置前，确保对目的地址和位宽度进行确认。下文是对目的地址位宽度进行设置的限制条件。</p> <table border="1"> <thead> <tr> <th>目的地址位宽度 DMACxCControl<Dwidth[2:0]></th> <th>最低有效地址的设置</th> </tr> </thead> <tbody> <tr> <td>000: 字节(8 位)</td> <td>无限制</td> </tr> <tr> <td>001: 半字符(16 位)</td> <td>设置为 2 的倍数, (0x0, 0x02, 0x4, 0x06, 0x8, 0xA, 0xC...)</td> </tr> <tr> <td>010: 字符(32 位)</td> <td>设置为 4 的倍数, (0x0, 0x4, 0x8, 0xC...)</td> </tr> </tbody> </table>	目的地址位宽度 DMACxCControl<Dwidth[2:0]>	最低有效地址的设置	000: 字节(8 位)	无限制	001: 半字符(16 位)	设置为 2 的倍数, (0x0, 0x02, 0x4, 0x06, 0x8, 0xA, 0xC...)	010: 字符(32 位)	设置为 4 的倍数, (0x0, 0x4, 0x8, 0xC...)
目的地址位宽度 DMACxCControl<Dwidth[2:0]>	最低有效地址的设置										
000: 字节(8 位)	无限制										
001: 半字符(16 位)	设置为 2 的倍数, (0x0, 0x02, 0x4, 0x06, 0x8, 0xA, 0xC...)										
010: 字符(32 位)	设置为 4 的倍数, (0x0, 0x4, 0x8, 0xC...)										

请勿在传输期间更新 DMACxCnDestAddr。变更 DMACxCnDestAddr，务必在变更前禁用通道"n"(DMACxCnConfiguration<E>="0")。

11.5.15 DMACxNLLI (DMAC 通道 x 链表项目寄存器)

	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	设置下一个传输信息的第一个地址。 设置一个小于 0xFFFF_FFF0 的数值。 当<LLI> = 0 时, LLI 为最后一个链。 DMA 传输完成后, 禁用 DMA 通道。
1-0	-	R/W	写作 0。

注: 有关<LLI>详细操作, 参见"11.6 特殊功能"。

11.5.16 DMACxCnControl (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	<p>启用传输中断寄存器。</p> <p>0: 禁用</p> <p>1: 启用</p> <p>通过设置<I>="1" 和 DMACxCnConfiguration<ITC>="1", 生成传输末端中断。当分散/聚集功能在最后一次传输 DMAC 设置流程中使用, 通过设置这个位进行启用, 传输末端中断生成仅在最后一次传输中是启用的。为了在正常传输中产生中断, 把这个位设置为"1"并变更为启用模式。</p>
30-28	-	W	写作 0。
27	DI	R/W	<p>增加传输目的地址</p> <p>0: 无需增加</p> <p>1: 增量</p>
26	SI	R/W	<p>增加传输源地址</p> <p>0: 无需增加</p> <p>1: 增量</p>
25-24	-	W	写作 0。
23-21	Dwidth[2:0]	R/W	<p>传输目的地位宽度。</p> <p>000: 字节(8 位)</p> <p>001: 半字(16 位)</p> <p>010: 字符(32 位) 其它: 保留</p> <p>有关设定值, 参考表 11-4。</p>
20-18	Swidth[2:0]	R/W	<p>传输源位宽度</p> <p>000: 字节(8 位)</p> <p>001: 半字(16 位)</p> <p>010: 字符(32 位) 其它: 保留</p> <p>有关设定值, 参考表 11-4。</p>
17-15	DBSize[2:0]	R/W	<p>传输目的地突发量: 注 1:</p> <p>000: 1 拍 100: 32 拍</p> <p>001: 4 拍 101: 64 拍</p> <p>010: 8 拍 110: 128 拍</p> <p>011: 16 拍 111: 256 拍</p> <p>有关设定值, 参考表 11-4。</p>



位	比特符号	类型	功能
14-12	SBSize[2:0]	R/W	<p>传输源突发量: (注 1)</p> <p>000: 1 拍 100: 32 拍</p> <p>001: 4 拍 101: 64 拍</p> <p>010: 8 拍 110: 128 拍</p> <p>011: 16 拍 111: 256 拍</p> <p>有关设定值, 参考表 11-4。</p>
11-0	TransferSize [11:0]	R/W	<p>设置传输总次数。</p> <p>把由传输源位宽(4 个字节/2 个字节/ 1 个字节)指定的每个位宽的传输数据量设置为 <TransferSize[11:0]>。因为突发量会内部显示在每次 DMA 请求时的传输数据量, 所以在传输源位宽和传输总次数不发生变化时, 即使设置任何突发量, 传输数据量绝不会发生变化。</p> <p>通过 DMA 传输, 把<TransferSize[11:0]>的值增加为 0。</p> <p>如果这是读取的, 则数据数量的值不会传输。</p> <p>传输总次数用作传输源位宽的单元。</p> <p>例如:</p> <p>当<Swidth>="000"(8 位)时, 传输次数以字节单元表示。</p> <p>当<Swidth>="001"(16 位), 传输次数以半字单元表示。</p> <p>当<Swidth>="010"(32 位)时, 传输次数以字符单元表示。</p>

注: 拟以 DBsize 和 SBsize 设置的突发量与 AHB 总线 HBURST 无任何关系。

表 11-4 如何决定<Dwidth[2:0]>, <Swidth[2:0]>, <DBSize[2:0]>和<SBSize[2:0]>的值

<Dwidth[2:0]> / <Swidth[2:0]>	<p>设置数字以满足下述表达:</p> <p>传输源位宽 x 传输总次数=传输目的地位宽 x N (N: 整数)</p> <hr/> <p>(示例 1)传输源位宽: 8 位, 传输目的地位宽: 32 位, 传输总次数: 25 次</p> <p>8 位 x25 次= 200 位(25 字节)</p> <p>N = 200 ÷ 32 = 6.25 字符</p> <p>因为 6.25 不是一个整数, 所以上述设置是无效的。</p> <p>如果传输源位宽小于传输目的地位宽, 在设置传输总次数时必须小心。</p> <hr/> <p>(示例 2)传输源位宽: 32 位, 传输目的地位宽: 16 位, 传输总次数: 13 次</p> <p>32 位 x13 次= 416 位(13 字符)</p> <p>N = 416 ÷ 16 = 26 半字</p> <p>因为 26 是整数, 所以上述设置是有效的。</p>
<DBSize[2:0]> / <SBSize[2:0]>	<p>当执行"外设到存储器"或"存储器到外设"的传输方式时, 外设电路产生 DMA 请求信号, 以说明准备就绪。触发此信号, 以执行数据传输。(如是"存储器到存储器"的传输方式, 仅使用软件启动。)</p> <p>设置突发量, 以定义外设在每个 DMA 请求信号时传输的数据量。这个寄存器与包含多数据的 FIFO 缓冲器一起使用。</p>

11.5.17 DMACxCnConfiguration (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 0。												
18	Halt	R/W	控制装置接受 DMA 请求 0: 接受 DMA 请求 1: 忽略 DMA 请求												
17	Active	R	说明通道 FIFO 中是否存在数据。 0: FIFO 中不存在数据 1: FIFO 中存在数据												
16	Lock	R/W	设置锁定传输(不分开传输)。 0: 禁用锁定传输 1: 启用锁定传输 在启用锁定传输时, 在不释放总线的情况下连续执行许多指定突发传输。有关具体操作, 参见"11.6 特殊功能"。												
15	ITC	R/W	传输末端中断启用寄存器。 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	设置传输方法 <table border="1" style="margin-left: 20px; border-collapse: collapse; text-align: center;"> <tr> <th style="width: 30%;"><FlowCntrl[2:0]> 设定值</th> <th style="width: 70%;">传输方法</th> </tr> <tr> <td>000:</td> <td>存储器到存储器(注)</td> </tr> <tr> <td>001:</td> <td>存储器到外设</td> </tr> <tr> <td>010:</td> <td>外设到存储器</td> </tr> <tr> <td>011:</td> <td>外设到外设</td> </tr> <tr> <td>100~ 111:</td> <td>保留</td> </tr> </table>	<FlowCntrl[2:0]> 设定值	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:	外设到外设	100~ 111:	保留
<FlowCntrl[2:0]> 设定值	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:	外设到外设														
100~ 111:	保留														
10	-	W	写作 0。												
9-6	DestPeripheral[3:0]	R/W	设置传输目的地外设(注 2) 参考"11.4.2 DMA 请求"。 当存储器是传输目的地时, 此设定被忽略。												
5	-	W	写作 0。												
4-1	SrcPeripheral[3:0]	R/W	设置传输源外设(注 2) 参考"11.4.2 DMA 请求"。												

			当存储器是传输源时，此设定被忽略。
--	--	--	-------------------



位	比特符号	类型	功能
0	E	R/W	通道启用 0: 禁用 1: 启用 此位可用于启用/禁用通道。(在选用"存储器到存储器"传输方式时, 此位作为启动位。) 在完成由 DMACxCnControl <TransferSize>指定的传输数据量时, 对应的<E>自动清"0"。 在传输期间禁用通道会损失 FIFO 中的数据。在重启前对所有通道进行初始化。 使用<HALT>停止 DMA 请求, 以暂停传输, 并调查数据直至<Active>变为"0", 然后采用<E>位禁用通道。

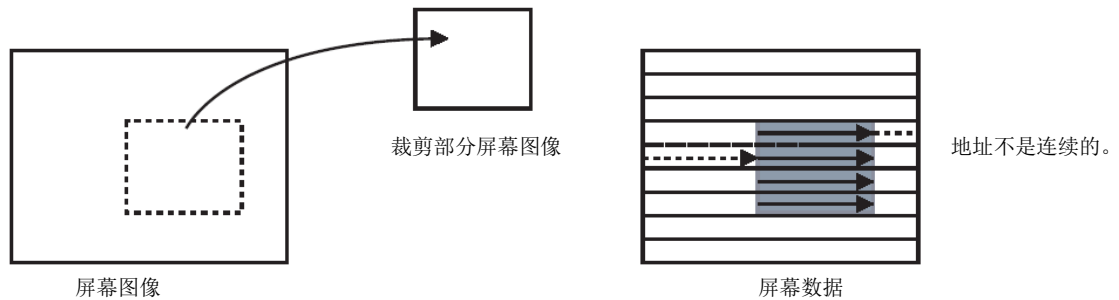
注 1: 在选用"存储器到存储器"传输方式时, 针对 DMA 启动, 不支持硬件启动。写入"1"<E>, 开始传输。

注 2: 在启用 DMACxENableChns<EnabledCHx>并把对应的 DMACxCnConfiguration<Halt>设置为"1"时, 在通道启用位(E:bit0)清"0"后写入它们。在没有这样做的情况下, 如在写入它们时发生从站错误, 则错误会通过复位恢复。关于从站错误, 当传输宽度和地址失配时, 会发生这种错误。

11.6 特殊功能

11.6.1 分散/聚集功能

在删除部分图像数据并对其进行传输时，不得把图像数据作为连续数据处理，而且地址会根据特殊规则发生突然变化。因为 DMA 只能通过使用连续地址传输数据，所以必需在地址发生变化的位置按要求进行设置。



分散/聚集功能每次可通过重新载入 DMA 设置的方式，对它们进行连续操作(传输源地址，目的地址，传输次数和传输总线宽度)，指定的 DMA 执行次数已通过预设的"链表"(此处的 CPU 不需要控制操作)完成。

在 DMACxCnLLI 寄存器中设置"1"，以启用/禁用操作。

可以链表设置的项目要配置下列 4 个字符：

1. DMACxCnSrcAddr
2. DMACxCnDestAddr
3. DMACxCnLLI
4. DMACxCnControl

它们可与中断操作一起使用。

中断取决于 DMACxCnControl 控制寄存器的计数末端中断启用位，可在每个 LLI 结束时产生。在使用此位时，可使用 LLI 增加条件，即使是在传输期间，以执行分支操作等。为清除中断，要控制 DMACxIntTCClear 寄存器的适当位。

11.6.2 链表操作

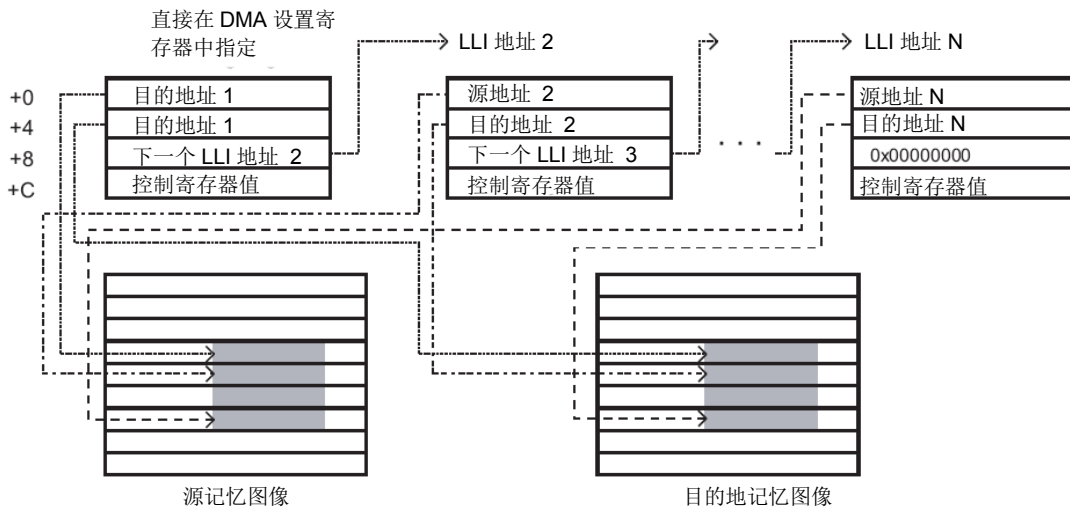
位操作分散/聚集功能，传输源和源数据区域需要首先通过创建一组链表进行确定。

每个设置称作 LLI(链表)。

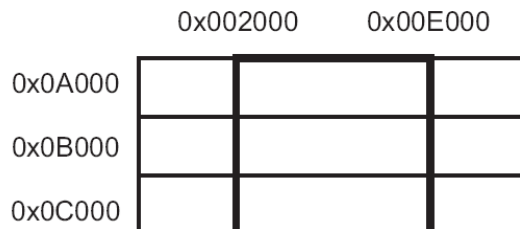
每个 LLI 控制一个数据块的传输。每个 LLI 表明正常 DMA 设置和连续数据的控制传输。每当每个 DMA 传输完成时，会加载下一个 LLI 设置，以继续 DMA 操作(菊花链)。

设置示例如下所示。

1. 第一个 DMA 传输设置应直接在 DMA 寄存器中进行。
2. 第二个和后续的 DMA 传输设置应在"下一个 LLI 地址 X"中的记忆设置的地址中写入。
3. 停止高达第 N 个 DMA 传输时，应将"下一个 LLI 地址 X"设置为 0x0000_0000。

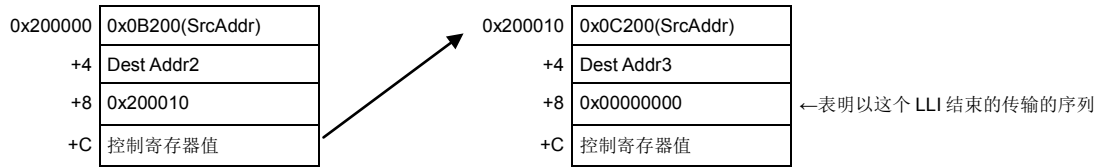


在由方形封闭的区域内传输数据时



设置寄存器	设置参数
+0 DMACxCnSrcAddr	:0x0A200
+4 DMACxCnDestAddr	:目的地址 1
+8 DMACxCnLLI	:0x200000
+C DMACxCnControl	:设置突发传输次数和传输次数等

链表





12. 16-位定时器/事件计数器(TMRB)

12.1 概要

TMRB 以下列 4 个操作模式操作:

- 16-位间隔定时器模式
- 16-位事件计数器模式
- 16-位可编程脉冲发生模式(PPG)
- 定时器同步模式

使用捕捉功能允许 TMRB 执行下列三种测量。

- 频率测量
- 脉冲宽度测量
- 时差测量

在本节的以下说明中, "x"表示通道编号。

12.2 规格差异

TMPM365FYXBG 包含 TMRB 的 10-通道。

每个通道独立运行，而且通道以相同方式运行，除表 12-1 中显示的规格差异外。

某些通道可把捕捉触发器和同步起动触发器放置在其它通道上。

1. TMRB 7 ~ TMRB 9 的触发器输出可用作其它通道的捕捉触发器。

- TB7OUT → 适用于 TMRB0 ~ TMRB1
- TB8OUT → 适用于 TMRB2 ~ TMRB3
- TB9OUT → 适用于 TMRB4 ~ TMRB6

2. 定时器同步模式(配置 TBxRUN)的起动触发器

- TMRB0 → 可同步起动 TMRB0 ~ TMRB3
- TMRB4 → 可同步起动 TMRB4 ~ TMRB7

3. 定时器预分频器同步模式(配置 TBxPRUN)的起动触发器

- TMRB0 → 可同步起动 TMRB0 ~ TMRB3
- TMRB4 → 可同步起动 TMRB4 ~ TMRB7

表 12-1 TMRB 模块的规格差异

规格	外部引脚		定时器之间的触发器功能		中断		内部连接		
	定时器触发器输出引脚	外部时钟/捕捉触发器输入引脚	捕捉触发器	同步起动触发器	捕捉中断	TMRB 中断	ADC 高优先级转换起动	ADC 正常转换起动	定时器 触发器输出 TBxOUT~SIO/UART (TXTRG :传输时钟)
TMRB0	TB0OUT	TB0IN0 TB0IN1	TB7OUT	-	INTCAP00 INTCAP01	INTTB0	-	-	-
TMRB1	-	TB1IN0 TB1IN1	TB7OUT	TB0PRUN, TB0RUN	INTCAP10 INTCAP11	INTTB1	-	-	-
TMRB2	TB2OUT	TB2IN0 TB2IN1	TB8OUT	TB0PRUN, TB0RUN	INTCAP20 INTCAP21	INTTB2	-	-	-
TMRB3	TB3OUT	TB3IN0 TB3IN1	TB8OUT	TB0PRUN, TB0RUN	INTCAP30 INTCAP31	INTTB3	-	-	-
TMRB4	TB4OUT	TB4IN0 TB4IN1	TB9OUT	-	INTCAP40 INTCAP41	INTTB4	INTCAP40	-	-
TMRB5	TB5OUT	TB5IN0 TB5IN1	TB9OUT	TB4PRUN, TB4RUN	INTCAP50 INTCAP51	INTTB5	-	INTCAP50	-
TMRB6	TB6OUT	TB6IN0 TB6IN1	TB9OUT	TB4PRUN, TB4RUN	INTCAP60 INTCAP61	INTTB6	-	-	-
TMRB7	TB7OUT	TB7IN0 TB7IN1	-	TB4PRUN, TB4RUN	INTCAP70 INTCAP71	INTTB7	-	-	-
TMRB8	TB8OUT	TB8IN0 TB8IN1	-	-	INTCAP80 INTCAP81	INTTB8	-	-	SIO0, SIO1
TMRB9	TB9OUT	TB9IN0 TB9IN1	-	-	INTCAP90 INTCAP91	INTTB9	-	-	-

12.3 配置

每个通道包括一个 16-位升计数器，2 个 16-位定时器寄存器(双缓冲的)，2 个 16-位捕捉寄存器，2 个比较器，1 个捕捉输入控制装置，1 个定时器触发器及其相关控制电路。定时器操作模式及定时器触发器是由寄存器控制的。

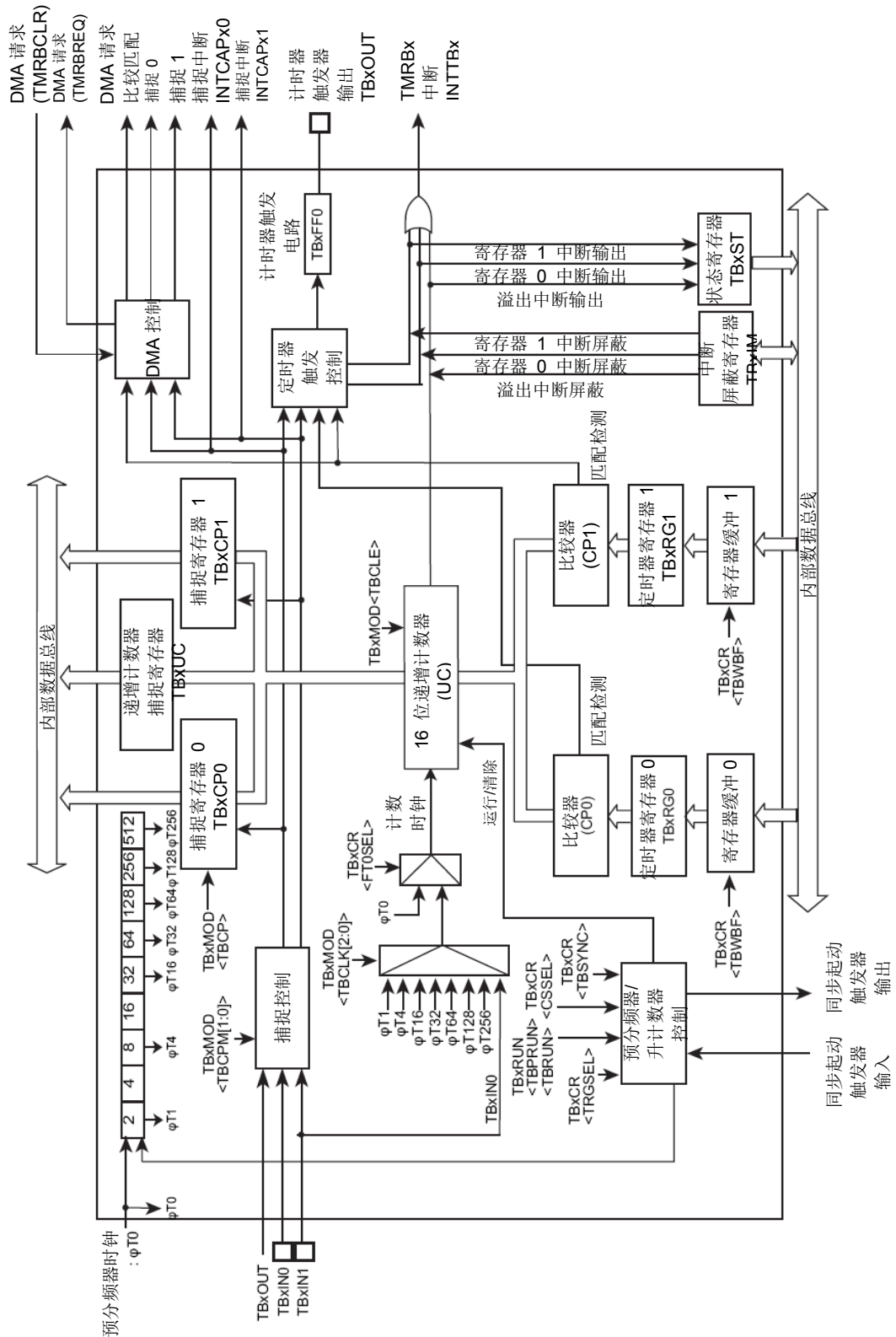


图 12-1 TMRBx 方块图(x= 0 ~ 9)



12.4 寄存器

12.4.1 通道对应寄存器列表

下表给出了各通道的寄存器名称与地址。

通道 x	基址
通道 0	0x400C_4000
通道 1	0x400C_4100
通道 2	0x400C_4200
通道 3	0x400C_4300
通道 4	0x400C_4400
通道 5	0x400C_4500
通道 6	0x400C_4600
通道 7	0x400C_4700
通道 8	0x400C_4800
通道 9	0x400C_4900

寄存器名称(x=0~9)		地址(基+)
启动寄存器	TBxEN	0x0000
RUN 寄存器	TBxRUN	0x0004
控制寄存器	TBxCR	0x0008
模式寄存器	TBxMOD	0x000C
触发电路控制寄存器	TBxFFCR	0x0010
状态寄存器	TBxST	0x0014
中断屏蔽寄存器	TBxIM	0x0018
递增计数器捕捉寄存器	TBxJC	0x001C
定时器寄存器 0	TBxRG0	0x0020
定时器寄存器 1	TBxRG1	0x0024
捕捉寄存器 0	TBxCP0	0x0028
捕捉寄存器 1	TBxCP1	0x002C
DMA 请求启用寄存器	TBxDMA	0x0030

注：在定时器工作期间，不可变更定时器控制寄存器，定时器模式寄存器及定时器触发器控制寄存器。在定时器停止后，才可对上述寄存器进行任何变更。

12.4.2 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	TBHALT	-	-	-	-	-	-
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	<p>TMRB_x 运行</p> <p>0: 禁用</p> <p>1: 启用</p> <p>规定 TMRB 运行。在该运行被禁用时，不会向 TMRB 模块中的其它寄存器提供任何时钟脉冲。可降低功耗。(这种情况禁用了向除 TBxEN 寄存器以外的其它寄存器读取和写入的功能。)如需使用 TMRB，可在对该 TMRB 模块中的各寄存器进行编程之前，启动该 TMRB 运行(即将其设置为"1")。如 TMRB 在被禁用之前执行了运行，则各寄存器中仍将保留该设置。</p>
6	TBHALT	R/W	<p>在调试停止期间时钟运行</p> <p>0: 运行</p> <p>1: 停止</p> <p>指示 TMRB 时钟在调试工具于使用中过渡到停止模式时，设置为运行或停止。</p>
5-0	-	R	读作 0。

12.4.3 TBxRUN ("运行"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	预分频器运行 0: 停止&清除 1: 计数
1	-	R	读作 0。
0	TBRUN	R/W	计数操作 0: 停止&清除 1: 计数

注：当计数器停止(<TBRUN>="0")并在读取 TBxUC<TBUC[15:0]>时，在计数器运行时读取的值被读出。

12.4.4 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	TBWWF	-	TBSYNC	FT0SEL	I2TB	TBINSEL	TRGSEL	CSSEL
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	TBWWF	R/W	双缓冲器 0: 禁用 1: 启用
6	-	R/W	写入"0"。
5	TBSYNC	R/W	同步模式切换 0: 个别的(通道单元) 1: 同步
4	FT0SEL	R/W	选择源时钟 $\phi T0$ 0: 选择除 $\phi T0$ 外的时钟(在 $TBxMOD<TBCLK[2:0]$ 时的时钟选择) 1: 选择 $\phi T0$
3	I2TB	R/W	在 IDLE 模式下运行 0: 停止 1: 运行
2	TBINSEL	R/W	写入"0"。
1	TRGSEL	R/W	选择外部寄存器。 0: 上升 1: 下降 在选择外部寄存器时控制(信号到 $TBxIN0$ 引脚的)边缘选择。
0	CSSEL	R/W	选择计数起动 0: 由软件起动 1: 由外部触发器起动

12.4.5 TBxMOD (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	-	TBCP	TBCPM		TBCLE	TBCLK		
复位后	0	1	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	TBCP	W	由软件进行捕捉控制 0: 由软件捕捉 1: 忽略 在写入"0"时, 捕捉寄存器 0 (TBxCP0)取计数值。 读作 1。
5-4	TBCPM[1:0]	R/W	捕捉定时 00: 禁用捕捉计时 01: TBxIN0↑, TBxIN1↑ 在 TBxIN0 引脚输入上升时, 把计数值带入捕捉寄存器 0 (TBxCP0)中。 在 TBxIN1 引脚输入上升时, 把计数值带入捕捉寄存器 1 (TBxCP1)中。 10: TBxIN0↑, TBxIN0↓ 在 TBxIN0 引脚输入上升时, 把计数值带入捕捉寄存器 0 (TBxCP0)中。 在 TBxIN0 引脚输入下降时, 把计数值带入捕捉寄存器 1 (TBxCP1)。 11: TBxOUT↑, TBxOUT↓ 在 16-位定时器匹配输出(TBxOUT)上升时, 把计数值带入捕捉寄存器 0 (TBnCP0)中; 在 TBxOUT 下降时, 把计数值带入捕捉寄存器 1 (TBnCP1)中。(x = 7, n = 0, 1), (x = 8, n = 2, 3), (x = 9, n = 4, 5, 6) (TMRB0 ~ 1: TB7OUT, TMRB2 ~ 3: TB8OUT, TMRB4 ~ 6: TB9OUT)
3	TBCLE	R/W	递增计数器控制器 0: 禁用递增计数器的清除 1: 启用递增计数器的清除。 清除并控制该递增计数器。 在已写入"0"时, 其可禁用递增计数器的清除。在已写入"1"时, 其可在存在寄存器 1 (TBxRG1)匹配的情况下清除递增计数器。
2-0	TBCLK[2:0]	R/W	选择该 TMRBx 源时钟。 对于 TMRBx 的源时钟, 不管<TBCLK [2:0]>的预设值如何, 在把其设置为 TBxCR<FT0SEL>="1"时, 选择φT0。 对于 TMRBx 的源时钟, 如果把其设置为 TBxCR<FT0SEL>="0"时, 选择下列时钟。 000: TBxIN0 引脚 输入 001: φT1 010: φT 4 011: φT 16 100: φT 32 101: φT 64 110: φT 128

		111: φ T 256
--	--	----------------------

注 1: 对应 TMRBx 在运行期间, 请勿对 TBxMOD 寄存器进行任何变更。

注 2: 当 TBxMOD 寄存器的通道是 7, 8 或 9 时, 禁止设置<TBCPM[1:0]>="11"。

12.4.6 TBxFFCR (触发器控制寄存器)

	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	-	-	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	在该递增计数器值被纳入 TBxCP1 时, TBxFF0 反转触发。 0: 禁用触发器 1: 启用触发 通过设置"1", 该计时器触发电路会在递增计数器被纳入该捕捉寄存器 1(TBxCP1)时反转。
4	TBC0T1	R/W	在该递增计数器值被纳入 TBxCP0 时, TBxFF0 反转触发。 0: 禁用触发器 1: 启用触发 通过设置"1", 该计时器触发电路会在递增计数器被纳入该捕捉寄存器 0(TBxCP0)时反转。
3	TBE1T1	R/W	在该递增计数器值与 TBxRG1 匹配时, TBxFF0 反转触发。 0: 禁用触发器 1: 启用触发 通过设置"1", 该计时器触发电路会在递增计数器与计时器寄存器 1(TBxRG1)匹配时反转。
2	TBE0T1	R/W	在该递增计数器值与 TBxRG0 匹配时, TBxFF0 反转触发。 0: 禁用触发器 1: 启用触发 通过设置"1", 该计时器触发电路会在递增计数器与计时器寄存器 0(TBxRG0)匹配时反转。
1-0	TBFF0C[1:0]	R/W	TBxFF0 控制 00: 反转 逆转 TBxFF0 的值(使用软件逆转)。 01: 设置 把 TBxFF0 设置为"1"。 10: 清除 把 TBxFF0 清为"0"。 11: 忽略 始终读作"11"。

12.4.7 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	溢位标志 0: 未发生溢位 1: 发生溢位 在某递增计数器溢出时, 即设置"1"。
1	INTTB1	R	匹配标志(TBxRG1) 0: 未检测到匹配 1: 检测到与 TBxRG1 匹配 在检测到计时器寄存器 1 (TBxRG1)的匹配时, 即设置"1"。
0	INTTB0	R	匹配标志(TBxRG0) 0: 未检测到匹配 1: 检测到与 TBxRG0 匹配 在检测到计时器寄存器 0 (TBxRG0)的匹配时, 即设置"1"。

注 1: 在清除该标志时, 应读取 TBxST 寄存器。

注 2: 当 TBxIM 寄存器的对应位禁用中断屏蔽配置时, 向 CPU 发出中断。

注 3: 即使 TBxIM 寄存器的屏蔽配置是有效的, 但要把状态设置为 TBxST 寄存器。

12.4.8 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	溢出中断屏蔽 0:禁用 1:启用 将该递增计数器溢出中断设置为禁用或启用。
1	TBIM1	R/W	匹配中断屏蔽 (TBxRG1) 0:禁用 1:启用 将计时器寄存器 1 (TBxRG1)的匹配中断屏蔽设置为启用或禁用。
0	TBIM0	R/W	匹配中断屏蔽 (TBxRG0) 0:禁用 1:启用 将计时器寄存器 0 (TBxRG0)的匹配中断屏蔽设置为启用或禁用。

注：即使 TBxIM 寄存器的屏蔽配置是有效的，但要把状态设置为 TBxST 寄存器。

12.4.9 TBxUC (递增计数器捕捉寄存器)

	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	通过读取递增计数器输出，捕捉某个值。 如果已读取 TBxUC，则可捕捉当前的递增计数器值。

注：在操作计数器和读取 TBxUC 时，会捕捉和读取递增计数器的值。

12.4.10 TBxRG0 (定时器寄存器 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	将某比较值设置到该递增计数器。

12.4.11 TBxRG1 (定时器寄存器 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	将某比较值设置到该递增计数器。

12.4.12 TBxCP0 (捕捉寄存器 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	已读取从该递增计数器捕捉到的某个值。

12.4.13 TBxCP1(捕捉寄存器 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	已读取从该递增计数器捕捉到的某个值。

12.4.14 TBxDMA(DMA 请求启用寄存器)

	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	-	-	-	-	-	TBDMAEN2	TBDMAEN1	TBDMAEN0
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	TBDMAEN2	R/W	选择 DMA 请求：比较匹配。 0：禁用 1：启用
1	TBDMAEN1	R/W	选择 DMA 请求：输入捕捉 1 0：禁用 1：启用
0	TBDMAEN0	R/W	选择 DMA 请求：输入捕捉 0 0：禁用 1：启用

注 1：当 TBxIM 寄存器的屏蔽配置有效时，DMA 请求不会发出。

注 2：DMA 请求因素分配通道是不同的，TMRB0 ~ 9。参见

12.5 每个电路操作说明

通道以相同方式操作，除表 12-1 中所示的规格不同外。

12.5.1 预分频器

利用现有的 4-位预分频，可生成递增计数器 UC 的源时钟。

预分频器 输入时钟 ϕT_0 是由 CG 中的 CGSYSCR<PRCK[2:0]>选择的 $f_{periph}/1$, $f_{periph}/2$, $f_{periph}/4$, $f_{periph}/8$, $f_{periph}/16$ 或 $f_{periph}/32$ 。外设时钟 f_{periph} 是 f_{gear} (由 CG 中的 CGSYSCR<FPSEL>选择的时钟)或 f_c (是在除以时钟齿轮前的时钟)

预分频器的运行或停止是以 $TBxRUN<TBPRUN>$ 设置的，其中写入"1"开始计数，写入"0"清除并停止计数。表 12-2 和表 12-3 表示预分频器输出时钟分辨率。

表 12-2 预分频器输出时钟分辨率($f_c = 48 \text{ MHz}$, $1/f_c = 0.02 \mu\text{s}$)

选择 外设时钟 CGSYSCR <FPSEL>	时钟齿轮值 CGSYSCR <GEAR[2:0]>	选择 预分频器时钟 CGSYSCR <PRCK[2:0]>	预分频输出时钟功能			
			TBxCR<FT0SEL >=" 1"(注 2)	TBxCR<FT0SEL>="0" (注 1)		
			$\phi T0$	$\phi T1$	$\phi T4$	$\phi T16$
0 (fgear)	000 (f_c)	000 (fperiph/1)	$f_c/2^0$ (0.02 μs) (注 2)	$f_c/2^1$ (0.04 μs)	$f_c/2^3$ (0.17 μs)	$f_c/2^5$ (0.67 μs)
		001 (fperiph/2)	$f_c/2^1$ (0.04 μs)	$f_c/2^2$ (0.08 μs)	$f_c/2^4$ (0.33 μs)	$f_c/2^6$ (1.33 μs)
		010 (fperiph/4)	$f_c/2^2$ (0.08 μs)	$f_c/2^3$ (0.17 μs)	$f_c/2^5$ (0.67 μs)	$f_c/2^7$ (2.67 μs)
		011 (fperiph/8)	$f_c/2^3$ (0.17 μs)	$f_c/2^4$ (0.33 μs)	$f_c/2^6$ (1.33 μs)	$f_c/2^8$ (5.33 μs)
		100 (fperiph/16)	$f_c/2^4$ (0.33 μs)	$f_c/2^5$ (0.67 μs)	$f_c/2^7$ (2.67 μs)	$f_c/2^9$ (10.67 μs)
		101 (fperiph/32)	$f_c/2^5$ (0.67 μs)	$f_c/2^6$ (1.33 μs)	$f_c/2^8$ (5.33 μs)	$f_c/2^{10}$ (21.33 μs)
	100 ($f_c/2$)	000 (fperiph/1)	$f_c/2^1$ (0.04 μs) (注 2)	$f_c/2^2$ (0.08 μs)	$f_c/2^4$ (0.33 μs)	$f_c/2^6$ (1.33 μs)
		001 (fperiph/2)	$f_c/2^2$ (0.08 μs)	$f_c/2^3$ (0.17 μs)	$f_c/2^5$ (0.67 μs)	$f_c/2^7$ (2.67 μs)
		010 (fperiph/4)	$f_c/2^3$ (0.17 μs)	$f_c/2^4$ (0.33 μs)	$f_c/2^6$ (1.33 μs)	$f_c/2^8$ (5.33 μs)
		011 (fperiph/8)	$f_c/2^4$ (0.33 μs)	$f_c/2^5$ (0.67 μs)	$f_c/2^7$ (2.67 μs)	$f_c/2^9$ (10.67 μs)
		100 (fperiph/16)	$f_c/2^5$ (0.67 μs)	$f_c/2^6$ (1.33 μs)	$f_c/2^8$ (5.33 μs)	$f_c/2^{10}$ (21.33 μs)
		101 (fperiph/32)	$f_c/2^6$ (1.33 μs)	$f_c/2^7$ (2.67 μs)	$f_c/2^9$ (10.67 μs)	$f_c/2^{11}$ (42.67 μs)
	101 ($f_c/4$)	000 (fperiph/1)	$f_c/2^2$ (0.08 μs) (注 2)	$f_c/2^3$ (0.17 μs)	$f_c/2^5$ (0.67 μs)	$f_c/2^7$ (2.67 μs)
		001 (fperiph/2)	$f_c/2^3$ (0.17 μs)	$f_c/2^4$ (0.33 μs)	$f_c/2^6$ (1.33 μs)	$f_c/2^8$ (5.33 μs)
		010 (fperiph/4)	$f_c/2^4$ (0.33 μs)	$f_c/2^5$ (0.67 μs)	$f_c/2^7$ (2.67 μs)	$f_c/2^9$ (10.67 μs)
		011 (fperiph/8)	$f_c/2^5$ (0.67 μs)	$f_c/2^6$ (1.33 μs)	$f_c/2^8$ (5.33 μs)	$f_c/2^{10}$ (21.33 μs)
		100 (fperiph/16)	$f_c/2^6$ (1.33 μs)	$f_c/2^7$ (2.67 μs)	$f_c/2^9$ (10.67 μs)	$f_c/2^{11}$ (42.67 μs)
		101 (fperiph/32)	$f_c/2^7$ (2.67 μs)	$f_c/2^8$ (5.33 μs)	$f_c/2^{10}$ (21.33 μs)	$f_c/2^{12}$ (85.33 μs)
	110 ($f_c/8$)	000 (fperiph/1)	$f_c/2^3$ (0.17 μs) (注 2)	$f_c/2^4$ (0.33 μs)	$f_c/2^6$ (1.33 μs)	$f_c/2^8$ (5.33 μs)
		001 (fperiph/2)	$f_c/2^4$ (0.33 μs)	$f_c/2^5$ (0.67 μs)	$f_c/2^7$ (2.67 μs)	$f_c/2^9$ (10.67 μs)
		010 (fperiph/4)	$f_c/2^5$ (0.67 μs)	$f_c/2^6$ (1.33 μs)	$f_c/2^8$ (5.33 μs)	$f_c/2^{10}$ (21.33 μs)
		011 (fperiph/8)	$f_c/2^6$ (1.33 μs)	$f_c/2^7$ (2.67 μs)	$f_c/2^9$ (10.67 μs)	$f_c/2^{11}$ (42.67 μs)
		100 (fperiph/16)	$f_c/2^7$ (2.67 μs)	$f_c/2^8$ (5.33 μs)	$f_c/2^{10}$ (21.33 μs)	$f_c/2^{12}$ (85.33 μs)
		101 (fperiph/32)	$f_c/2^8$ (5.33 μs)	$f_c/2^9$ (10.67 μs)	$f_c/2^{11}$ (42.67 μs)	$f_c/2^{13}$ (170.67 μs)
111 ($f_c/16$)	000 (fperiph/1)	$f_c/2^4$ (0.33 μs) (注 2)	$f_c/2^5$ (0.67 μs)	$f_c/2^7$ (2.67 μs)	$f_c/2^9$ (10.67 μs)	
	001 (fperiph/2)	$f_c/2^5$ (0.67 μs)	$f_c/2^6$ (1.33 μs)	$f_c/2^8$ (5.33 μs)	$f_c/2^{10}$ (21.33 μs)	
	010 (fperiph/4)	$f_c/2^6$ (1.33 μs)	$f_c/2^7$ (2.67 μs)	$f_c/2^9$ (10.67 μs)	$f_c/2^{11}$ (42.67 μs)	
	011 (fperiph/8)	$f_c/2^7$ (2.67 μs)	$f_c/2^8$ (5.33 μs)	$f_c/2^{10}$ (21.33 μs)	$f_c/2^{12}$ (85.33 μs)	
	100 (fperiph/16)	$f_c/2^8$ (5.33 μs)	$f_c/2^9$ (10.67 μs)	$f_c/2^{11}$ (42.67 μs)	$f_c/2^{13}$ (170.67 μs)	
	101 (fperiph/32)	$f_c/2^9$ (10.7 μs)	$f_c/2^{10}$ (21.34 μs)	$f_c/2^{12}$ (85.34 μs)	$f_c/2^{14}$ (341.34 μs)	

表 12-2 预分频器输出时钟分辨率($f_c = 48 \text{ MHz}$, $1/f_c = 0.02 \mu\text{s}$)

选择 外设时钟 CGSYSCR <FPSEL>	时钟齿轮值 CGSYSCR <GEAR[2:0]>	选择 预分频器时钟 CGSYSCR PRCK[2:0]>	预分频输出时钟功能			
			TBxCR<FT0SEL >=" 1"(注 2)	TBxCR<FT0SEL>="0" (注 1)		
			$\phi T0$	$\phi T1$	$\phi T4$	$\phi T16$
1 (fc)	000 (fc)	000 (fperiph/1)	$fc/2^0$ (0.02 μs) (注 2)	$fc/2^1$ (0.04 μs)	$fc/2^3$ (0.17 μs)	$fc/2^5$ (0.67 μs)
		001 (fperiph/2)	$fc/2^1$ (0.04 μs)	$fc/2^2$ (0.08 μs)	$fc/2^4$ (0.33 μs)	$fc/2^6$ (1.33 μs)
		010 (fperiph/4)	$fc/2^2$ (0.08 μs)	$fc/2^3$ (0.17 μs)	$fc/2^5$ (0.67 μs)	$fc/2^7$ (2.67 μs)
		011 (fperiph/8)	$fc/2^3$ (0.17 μs)	$fc/2^4$ (0.33 μs)	$fc/2^6$ (1.33 μs)	$fc/2^8$ (5.33 μs)
		100 (fperiph/16)	$fc/2^4$ (0.33 μs)	$fc/2^5$ (0.67 μs)	$fc/2^7$ (2.67 μs)	$fc/2^9$ (10.67 μs)
		101 (fperiph/32)	$fc/2^5$ (0.67 μs)	$fc/2^6$ (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)	$fc/2^5$ (0.67 μs)
		001 (fperiph/2)	$fc/2^1$ (0.04 μs) (注 2)	$fc/2^2$ (0.08 μs)	$fc/2^4$ (0.33 μs)	$fc/2^6$ (1.33 μs)
		010 (fperiph/4)	$fc/2^2$ (0.08 μs)	$fc/2^3$ (0.17 μs)	$fc/2^5$ (0.67 μs)	$fc/2^7$ (2.67 μs)
		011 (fperiph/8)	$fc/2^3$ (0.17 μs)	$fc/2^4$ (0.33 μs)	$fc/2^6$ (1.33 μs)	$fc/2^8$ (5.33 μs)
		100 (fperiph/16)	$fc/2^4$ (0.33 μs)	$fc/2^5$ (0.67 μs)	$fc/2^7$ (2.67 μs)	$fc/2^9$ (10.67 μs)
		101 (fperiph/32)	$fc/2^5$ (0.67 μs)	$fc/2^6$ (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)	$fc/2^5$ (0.67 μs)
		001 (fperiph/2)	-	-	$fc/2^4$ (0.33 μs)	$fc/2^6$ (1.33 μs)
		010 (fperiph/4)	$fc/2^2$ (0.08 μs) (注 2)	$fc/2^3$ (0.17 μs)	$fc/2^5$ (0.67 μs)	$fc/2^7$ (2.67 μs)
		011 (fperiph/8)	$fc/2^3$ (0.17 μs)	$fc/2^4$ (0.33 μs)	$fc/2^6$ (1.33 μs)	$fc/2^8$ (5.33 μs)
		100 (fperiph/16)	$fc/2^4$ (0.33 μs)	$fc/2^5$ (0.67 μs)	$fc/2^7$ (2.67 μs)	$fc/2^9$ (10.67 μs)
		101 (fperiph/32)	$fc/2^5$ (0.67 μs)	$fc/2^6$ (1.33 μs)	$fc/2^8$ (5.33 μs)	$fc/2^{10}$ (21.33 μs)
	110 (fc/8)	000 (fperiph/1)	-	-	-	$fc/2^5$ (0.67 μs)
		001 (fperiph/2)	-	-	$fc/2^4$ (0.33 μs)	$fc/2^6$ (1.33 μs)
		010 (fperiph/4)	-	-	$fc/2^5$ (0.67 μs)	$fc/2^7$ (2.67 μs)
		011 (fperiph/8)	$fc/2^3$ (0.17 μs) (注 2)	$fc/2^4$ (0.33 μs)	$fc/2^6$ (1.33 μs)	$fc/2^8$ (5.33 μs)
		100 (fperiph/16)	$fc/2^4$ (0.33 μs)	$fc/2^5$ (0.67 μs)	$fc/2^7$ (2.67 μs)	$fc/2^9$ (10.67 μs)
		101 (fperiph/32)	$fc/2^5$ (0.67 μs)	$fc/2^6$ (1.33 μs)	$fc/2^8$ (5.33 μs)	$fc/2^{10}$ (21.33 μs)
111 (fc/16)	000 (fperiph/1)	-	-	-	$fc/2^5$ (0.67 μs)	
	001 (fperiph/2)	-	-	$fc/2^4$ (0.33 μs)	$fc/2^6$ (1.33 μs)	
	010 (fperiph/4)	-	-	$fc/2^5$ (0.67 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)	$fc/2^4$ (0.33 μs) (注 2)	$fc/2^5$ (0.67 μs)	$fc/2^7$ (2.67 μs)	$fc/2^9$ (10.67 μs)	
	101 (fperiph/32)	$fc/2^5$ (0.67 μs)	$fc/2^6$ (1.33 μs)	$fc/2^8$ (5.33 μs)	$fc/2^{10}$ (21.33 μs)	

- 注 1: 必须选择预分频输出时钟 ϕTn , 以满足 $\phi Tn < f_{sys}$ 的条件(使得 ϕTn 慢于 f_{sys})。
- 注 2: 只有在把其设置为 $TBxCR<FT0SEL>="1"$ 时, 并在 $\phi Tn < f_{sys}$ 的条件下, 预分频器输出时钟 ϕTn 也可使用。
- 注 3: 注: 不要在该计时器运行期间改变时钟齿轮。
- 注 4: "-"表示禁止设置。

表 12-3 预分频器输出时钟分辨率(fc = 48MHz)

选择 外设时钟 CGSYSCR <FPSEL>	时钟齿轮值 CGSYSCR <GEAR[2:0]>	选择 预分频器时钟 CGSYSCR <PRCK[2:0]>	预分频器输出时钟功能			
			φT32	φT64	φT128	φT256
0 (fgear)	000 (fc)	000 (fperiph/1)	fc/2 ⁶ (1.33 μs)	fc/2 ⁷ (2.67 μs)	fc/2 ⁸ (5.33 μs)	fc/2 ⁹ (10.67 μs)
		001 (fperiph/2)	fc/2 ⁷ (2.67 μs)	fc/2 ⁸ (5.33 μs)	fc/2 ⁹ (10.67 μs)	fc/2 ¹⁰ (21.33 μs)
		010 (fperiph/4)	fc/2 ⁸ (5.33 μs)	fc/2 ⁹ (10.67 μs)	fc/2 ¹⁰ (21.33 μs)	fc/2 ¹¹ (42.67 μs)
		011 (fperiph/8)	fc/2 ⁹ (10.67 μs)	fc/2 ¹⁰ (21.33 μs)	fc/2 ¹¹ (42.67 μs)	fc/2 ¹² (85.33 μs)
		100 (fperiph/16)	fc/2 ¹⁰ (21.33 μs)	fc/2 ¹¹ (42.67 μs)	fc/2 ¹² (85.33 μs)	fc/2 ¹³ (170.67 μs)
		101 (fperiph/32)	fc/2 ¹¹ (42.67 μs)	fc/2 ¹² (85.33 μs)	fc/2 ¹³ (170.67 μs)	fc/2 ¹⁴ (341.33 μs)
	100 (fc/2)	000 (fperiph/1)	fc/2 ⁷ (2.67 μs)	fc/2 ⁸ (5.33 μs)	fc/2 ⁹ (10.67 μs)	fc/2 ¹⁰ (21.33 μs)
		001 (fperiph/2)	fc/2 ⁸ (5.33 μs)	fc/2 ⁹ (10.67 μs)	fc/2 ¹⁰ (21.33 μs)	fc/2 ¹¹ (42.67 μs)
		010 (fperiph/4)	fc/2 ⁹ (10.67 μs)	fc/2 ¹⁰ (21.33 μs)	fc/2 ¹¹ (42.67 μs)	fc/2 ¹² (85.33 μs)
		011 (fperiph/8)	fc/2 ¹⁰ (21.33 μs)	fc/2 ¹¹ (42.67 μs)	fc/2 ¹² (85.33 μs)	fc/2 ¹³ (170.67 μs)
		100 (fperiph/16)	fc/2 ¹¹ (42.67 μs)	fc/2 ¹² (85.33 μs)	fc/2 ¹³ (170.67 μs)	fc/2 ¹⁴ (341.33 μs)
		101 (fperiph/32)	fc/2 ¹² (85.33 μs)	fc/2 ¹³ (170.67 μs)	fc/2 ¹⁴ (341.33 μs)	fc/2 ¹⁵ (682.67 μs)
	101 (fc/4)	000 (fperiph/1)	fc/2 ⁸ (5.33 μs)	fc/2 ⁹ (10.67 μs)	fc/2 ¹⁰ (21.33 μs)	fc/2 ¹¹ (42.67 μs)
		001 (fperiph/2)	fc/2 ⁹ (10.67 μs)	fc/2 ¹⁰ (21.33 μs)	fc/2 ¹¹ (42.67 μs)	fc/2 ¹² (85.33 μs)
		010 (fperiph/4)	fc/2 ¹⁰ (21.33 μs)	fc/2 ¹¹ (42.67 μs)	fc/2 ¹² (85.33 μs)	fc/2 ¹³ (170.67 μs)
		011 (fperiph/8)	fc/2 ¹¹ (42.67 μs)	fc/2 ¹² (85.33 μs)	fc/2 ¹³ (170.67 μs)	fc/2 ¹⁴ (341.33 μs)
		100 (fperiph/16)	fc/2 ¹² (85.33 μs)	fc/2 ¹³ (170.67 μs)	fc/2 ¹⁴ (341.33 μs)	fc/2 ¹⁵ (682.67 μs)
		101 (fperiph/32)	fc/2 ¹³ (170.67 μs)	fc/2 ¹⁴ (341.33 μs)	fc/2 ¹⁵ (682.67 μs)	fc/2 ¹⁶ (1365.33 μs)
	110 (fc/8)	000 (fperiph/1)	fc/2 ⁹ (10.67 μs)	fc/2 ¹⁰ (21.33 μs)	fc/2 ¹¹ (42.67 μs)	fc/2 ¹² (85.33 μs)
		001 (fperiph/2)	fc/2 ¹⁰ (21.33 μs)	fc/2 ¹¹ (42.67 μs)	fc/2 ¹² (85.33 μs)	fc/2 ¹³ (170.67 μs)
		010 (fperiph/4)	fc/2 ¹¹ (42.67 μs)	fc/2 ¹² (85.33 μs)	fc/2 ¹³ (170.67 μs)	fc/2 ¹⁴ (341.33 μs)
		011 (fperiph/8)	fc/2 ¹² (85.33 μs)	fc/2 ¹³ (170.67 μs)	fc/2 ¹⁴ (341.33 μs)	fc/2 ¹⁵ (682.67 μs)
		100 (fperiph/16)	fc/2 ¹³ (170.67 μs)	fc/2 ¹⁴ (341.33 μs)	fc/2 ¹⁵ (682.67 μs)	fc/2 ¹⁶ (1365.33 μs)
		101 (fperiph/32)	fc/2 ¹⁴ (341.33 μs)	fc/2 ¹⁵ (682.67 μs)	fc/2 ¹⁶ (1365.33 μs)	fc/2 ¹⁷ (2730.67 μs)
111 (fc/16)	000 (fperiph/1)	fc/2 ¹⁰ (21.33 μs)	fc/2 ¹¹ (42.67 μs)	fc/2 ¹² (85.33 μs)	fc/2 ¹³ (170.67 μs)	
	001 (fperiph/2)	fc/2 ¹¹ (42.67 μs)	fc/2 ¹² (85.33 μs)	fc/2 ¹³ (170.67 μs)	fc/2 ¹⁴ (341.33 μs)	
	010 (fperiph/4)	fc/2 ¹² (85.33 μs)	fc/2 ¹³ (170.67 μs)	fc/2 ¹⁴ (341.33 μs)	fc/2 ¹⁵ (682.67 μs)	
	011 (fperiph/8)	fc/2 ¹³ (170.67 μs)	fc/2 ¹⁴ (341.33 μs)	fc/2 ¹⁵ (682.67 μs)	fc/2 ¹⁶ (1365.33 μs)	
	100 (fperiph/16)	fc/2 ¹⁴ (341.33 μs)	fc/2 ¹⁵ (682.67 μs)	fc/2 ¹⁶ (1365.33 μs)	fc/2 ¹⁷ (2730.67 μs)	
	101 (fperiph/32)	fc/2 ¹⁵ (682.66 μs)	fc/2 ¹⁶ (1365.34 μs)	fc/2 ¹⁷ (2730.67 μs)	fc/2 ¹⁸ (5461.34 μs)	

表 12-3 预分频器输出时钟分辨率($f_c = 48\text{MHz}$)

选择 外设时钟 CGSYSCR <FPSEL>	时钟齿轮值 CGSYSCR <GEAR[2:0]>	选择 预分频器时钟 CGSYSCR <PRCK[2:0]>	预分频输出时钟功能			
			$\phi T32$	$\phi T64$	$\phi T128$	$\phi T256$
1 (fc)	000 (fc)	000 (fperiph/1)	$fc/2^6$ (1.33 μs)	$fc/2^7$ (2.67 μs)	$fc/2^8$ (5.33 μs)	$fc/2^9$ (10.67 μs)
		001 (fperiph/2)	$fc/2^7$ (2.67 μs)	$fc/2^8$ (5.33 μs)	$fc/2^9$ (10.67 μs)	$fc/2^{10}$ (21.33 μs)
		010 (fperiph/4)	$fc/2^8$ (5.33 μs)	$fc/2^9$ (10.67 μs)	$fc/2^{10}$ (21.33 μs)	$fc/2^{11}$ (42.67 μs)
		011 (fperiph/8)	$fc/2^9$ (10.67 μs)	$fc/2^{10}$ (21.33 μs)	$fc/2^{11}$ (42.67 μs)	$fc/2^{12}$ (85.33 μs)
		100 (fperiph/16)	$fc/2^{10}$ (21.33 μs)	$fc/2^{11}$ (42.67 μs)	$fc/2^{12}$ (85.33 μs)	$fc/2^{13}$ (170.67 μs)
		101 (fperiph/32)	$fc/2^{11}$ (42.67 μs)	$fc/2^{12}$ (85.33 μs)	$fc/2^{13}$ (170.67 μs)	$fc/2^{14}$ (341.33 μs)
	100 (fc/2)	000 (fperiph/1)	$fc/2^6$ (1.33 μs)	$fc/2^7$ (2.67 μs)	$fc/2^8$ (5.33 μs)	$fc/2^9$ (10.67 μs)
		001 (fperiph/2)	$fc/2^7$ (2.67 μs)	$fc/2^8$ (5.33 μs)	$fc/2^9$ (10.67 μs)	$fc/2^{10}$ (21.33 μs)
		010 (fperiph/4)	$fc/2^8$ (5.33 μs)	$fc/2^9$ (10.67 μs)	$fc/2^{10}$ (21.33 μs)	$fc/2^{11}$ (42.67 μs)
		011 (fperiph/8)	$fc/2^9$ (10.67 μs)	$fc/2^{10}$ (21.33 μs)	$fc/2^{11}$ (42.67 μs)	$fc/2^{12}$ (85.33 μs)
		100 (fperiph/16)	$fc/2^{10}$ (21.33 μs)	$fc/2^{11}$ (42.67 μs)	$fc/2^{12}$ (85.33 μs)	$fc/2^{13}$ (170.67 μs)
		101 (fperiph/32)	$fc/2^{11}$ (42.67 μs)	$fc/2^{12}$ (85.33 μs)	$fc/2^{13}$ (170.67 μs)	$fc/2^{14}$ (341.33 μs)
	101 (fc/4)	000 (fperiph/1)	$fc/2^6$ (1.33 μs)	$fc/2^7$ (2.67 μs)	$fc/2^8$ (5.33 μs)	$fc/2^9$ (10.67 μs)
		001 (fperiph/2)	$fc/2^7$ (2.67 μs)	$fc/2^8$ (5.33 μs)	$fc/2^9$ (10.67 μs)	$fc/2^{10}$ (21.33 μs)
		010 (fperiph/4)	$fc/2^8$ (5.33 μs)	$fc/2^9$ (10.67 μs)	$fc/2^{10}$ (21.33 μs)	$fc/2^{11}$ (42.67 μs)
		011 (fperiph/8)	$fc/2^9$ (10.67 μs)	$fc/2^{10}$ (21.33 μs)	$fc/2^{11}$ (42.67 μs)	$fc/2^{12}$ (85.33 μs)
		100 (fperiph/16)	$fc/2^{10}$ (21.33 μs)	$fc/2^{11}$ (42.67 μs)	$fc/2^{12}$ (85.33 μs)	$fc/2^{13}$ (170.67 μs)
		101 (fperiph/32)	$fc/2^{11}$ (42.67 μs)	$fc/2^{12}$ (85.33 μs)	$fc/2^{13}$ (170.67 μs)	$fc/2^{14}$ (341.33 μs)
	110 (fc/8)	000 (fperiph/1)	$fc/2^6$ (1.33 μs)	$fc/2^7$ (2.67 μs)	$fc/2^8$ (5.33 μs)	$fc/2^9$ (10.67 μs)
		001 (fperiph/2)	$fc/2^7$ (2.67 μs)	$fc/2^8$ (5.33 μs)	$fc/2^9$ (10.67 μs)	$fc/2^{10}$ (21.33 μs)
		010 (fperiph/4)	$fc/2^8$ (5.33 μs)	$fc/2^9$ (10.67 μs)	$fc/2^{10}$ (21.33 μs)	$fc/2^{11}$ (42.67 μs)
		011 (fperiph/8)	$fc/2^9$ (10.67 μs)	$fc/2^{10}$ (21.33 μs)	$fc/2^{11}$ (42.67 μs)	$fc/2^{12}$ (85.33 μs)
		100 (fperiph/16)	$fc/2^{10}$ (21.33 μs)	$fc/2^{11}$ (42.67 μs)	$fc/2^{12}$ (85.33 μs)	$fc/2^{13}$ (170.67 μs)
		101 (fperiph/32)	$fc/2^{11}$ (42.67 μs)	$fc/2^{12}$ (85.33 μs)	$fc/2^{13}$ (170.67 μs)	$fc/2^{14}$ (341.33 μs)
111 (fc/16)	000 (fperiph/1)	$fc/2^6$ (1.33 μs)	$fc/2^7$ (2.67 μs)	$fc/2^8$ (5.33 μs)	$fc/2^9$ (10.67 μs)	
	001 (fperiph/2)	$fc/2^7$ (2.67 μs)	$fc/2^8$ (5.33 μs)	$fc/2^9$ (10.67 μs)	$fc/2^{10}$ (21.33 μs)	
	010 (fperiph/4)	$fc/2^8$ (5.33 μs)	$fc/2^9$ (10.67 μs)	$fc/2^{10}$ (21.33 μs)	$fc/2^{11}$ (42.67 μs)	
	011 (fperiph/8)	$fc/2^9$ (10.67 μs)	$fc/2^{10}$ (21.33 μs)	$fc/2^{11}$ (42.67 μs)	$fc/2^{12}$ (85.33 μs)	
	100 (fperiph/16)	$fc/2^{10}$ (21.33 μs)	$fc/2^{11}$ (42.67 μs)	$fc/2^{12}$ (85.33 μs)	$fc/2^{13}$ (170.67 μs)	
	101 (fperiph/32)	$fc/2^{11}$ (42.67 μs)	$fc/2^{12}$ (85.33 μs)	$fc/2^{13}$ (170.67 μs)	$fc/2^{14}$ (341.33 μs)	

注 1: 必须选择预分频器输出时钟 ϕTn , 以便符合 $\phi Tn < f_{\text{sys}}$ (以使 ϕTn 低于 f_{sys})。

注 2: 不要在该计时器运行期间改变时钟齿轮。

注 3: "-"表示禁止设置。

12.5.2 递增计数器 (UC)

UC 为 16-位二进制计数器。

- 源时钟

由 TBxMOD<TBCLK[2:0]>和 TBxCR<FT0SEL>指定的 UC 源时钟可从 $\phi T0$, $\phi T1$, $\phi T4$, $\phi T16$, $\phi T32$, $\phi T64$, $\phi T128$ 或 $\phi T256$ 或 TBxIN0 引脚的外部时钟选择。

仅在把其设置为 TBxCR<FT0SEL>="1"时, 预分频器输出时钟 $\wedge Tn < f_{sys}$ 。
- 计数开始/停止

计数器运行由 TBxRUN<TBRUN>规定。如果<TBRUN> = "1", 则 UC 开始计数; 如果<TBRUN> = "0", 则其停止计数并清除计数器值。
- UC 清除用时标
 1. 在检测到某匹配时

在比较器检测到计数器值与在 TBxRG1 中设置的值之间存在匹配时, 通过设置 TBxMOD<TBCLE> = "1", 即可清除 UC。如果 TBxMOD<TBCLE> = "0", UC 就以自由运转计数器运行。
 2. 在 UC 停止时

如果 TBxRUN<TBRUN>="0", 则 UC 停止计数并清除计数器值。
- UC 溢出

如果 UC 发生溢位, 会产生 INTTBx 溢位中断。

12.5.3 定时器寄存器 (TBxRG0, TBxRG1)

TBxRG0 与 TBxRG1 均为寄存器, 用于设置进行递增计数器值比较所需的值, 且各通道均内建有两个寄存器。如果比较器检测到在计时器寄存器中设置的值与某 UC 递增计数器中某值之间存在匹配关系, 则其会输出匹配检测信号。

TBxRG0 和 TBxRG1 是由与寄存器缓冲区配对的双缓冲配置组成的。在初始状态时, 该双缓冲被禁用。

控制用双缓冲的禁用或启用, 由 TBxCR<TBWBF>位指定。如果<TBWBF> = "0", 则双缓冲变为禁用。如果<TBWBF> = "1", 则其变为启用。在启用双缓冲时, 如果 UC 与 TBxRG1 相匹配时, 即完成自寄存器缓冲区至定时器寄存器(TBxRG0/1)之间的数据传输。在计数器停止时, 即使启用双缓冲, 双缓冲即以单缓冲运行, 直接数据可写入 TBxRG0 和 TBxRG1。

12.5.4 捕捉

该电路用于控制从 UC 递增计数器传送到 TBxCP0 与 TBxCP1 捕捉寄存器的锁存值的时序。锁存数据的计时由 TBxMOD<TBxCPM[1:0]>指定。

软件也可用于从 UC 递增计数器向捕捉寄存器输入数值；特别是，在每当把"0"写入 TBxMOD<TBxCP>时，把 UC 数值纳入 TBxCP0 捕捉寄存器。

12.5.5 捕捉寄存器 (TBxCP0, TBxCP1)

该寄存器可捕捉一个递增计数器(UC)值。

12.5.6 递增计数器捕捉寄存器 (TBxUC)

除上文所述的捕捉功能外，可通过读取 TBxUC 寄存器捕捉 UC 的当前计数值。

12.5.7 比较器 (CP0, CP1)

该寄存器可将该递增计数器(UC)与计时器寄存器(TBxRG0 和 TBxRG1)的值设置进行比较，以检测是否存在匹配。如果检测到匹配，会生成 INTTBx。

12.5.8 定时器触发器 (TBxFF0)

定时器触发器(TBxFF0)是通过由比较器发出的匹配信号及发送至捕捉寄存器的锁存信号逆转的。通过设置 TBxFFCR<TBxT1, TBC0T1, TBE1T1 与 TBE0T1>，即可启用或禁用其反转。

在复位之后，TBxFF0 的值变为未定义状态。通过将"00"写入到 TBxFFCR<TBxFF0C[1:0]>，即可让触发电路发生反转。写入"01"即可将其设置为"1"，通过写入"10"即可清"0"。

TBxFF0 的值可被输出到计时器输出引脚(TBxOUT)。如拟进行计时器输出，则必须事先对相应端口设置进行编程。

12.5.9 捕捉中断 (INTCAPx0, INTCAPx1)

中断 INTCAPx0 和 INTCAPx1 可在从 UC 递增计数器向 TBxCP0 和 TBxCP1 捕捉寄存器锁存数值时生成。中断时序由 CPU 指定。

12.6 各个模式操作说明

12.6.1 16-位间隔计时器模式

如果生成恒定期中断，要把间隔时间设置为定时器寄存器(TBxRG1)，以产生 INTTBx 中断。

		7	6	5	4	3	2	1	0	
TBxEN	←	1	X	X	X	X	X	X	X	启用 TMRBx 运行。
TBxRUN	←	X	X	X	X	X	0	X	0	停止计数运行。
中断设置-启用寄存器	←	*	*	*	*	*	*	*	*	允许通过把对应位设置为"1"的方式生成 INTTBx 中断。
TBxFFCR	←	X	X	0	0	0	0	1	1	禁用 TBxFF0 反转触发信号。
TBxMOD	←	0	1	0	0	1	*	*	*	改为将预分频输出时钟用作输入时钟。指定捕捉功能为禁用。
(***= 001 ~ 111)										
TBxRG1	←	*	*	*	*	*	*	*	*	指定时间间隔。(16 位)
	←	*	*	*	*	*	*	*	*	
TBxRUN	←	*	*	*	*	*	1	X	1	启动 TMRBx。

注: X; 忽略 -; 无变化

12.6.2 16-位事件计数器模式

可通过把输入时钟用作外部时钟(TBxIN0 引脚输入)的方式把其设置为事件计数器。

递增计数器在 TBxIN0 引脚输入的上升边缘递增相加。通过用软件进行值捕捉，并读取所捕捉的值，即有可能读取该计数值。

		7	6	5	4	3	2	1	0	
TBxEN	←	1	X	X	X	X	X	X	X	启用 TMRBx 运行。
TBxRUN	←	X	X	X	X	X	0	X	0	停止计数运行。
PxIE[m]	←								1	分配对应端口至 TBxIN0。
PxFR1[m]	←								1	
TBxFFCR	←	X	X	0	0	0	0	1	1	禁用为 TBxFF0 反向触发器
TBxMOD	←	0	1	0	0	0	0	0	0	作为输入时钟变更为 TBxIN0
TBxRUN	←	*	*	*	*	*	1	X	1	启动 TMRBx。
TBxMOD	←	X	0	0	0	0	0	0	0	软件捕捉已完成。

注 1: m: 端口对应位

注 2: X; 忽略

-; 无变化

12.6.3 16-位 PPG(可编程脉冲发生)输出模式

可输出具备任何频率与任何占空的方波(可编程方波)。输出脉冲的活动性可高可低。

在递增计数器(UC 的设定值)可匹配各计时器寄存器(TBxRG0 和 TBxRG1)的设定值时,通过触发计时器触发电路(TBxFF)进行反转,即可从 TBxOUT 引脚输出可编程方波。请注意 TBxRG0 和 TBxRG1 的设定值必须符合下列要求:

$$(TBxRG0 \text{ 的设定值}) < TBxRG1 \text{ 的设定值}$$

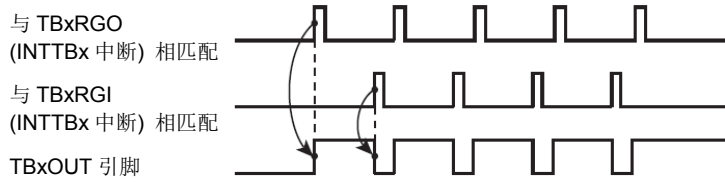


图 12-2 可编程脉冲发生(PPG)输出示例

在该模式下,在递增计数器的设定值匹配 TBxRG1 的设定值时,通过启用 TBxRG0 的双缓冲,即可将寄存器缓冲区 0 的值转入 TBxRG0。这便于对低占空进行处理。

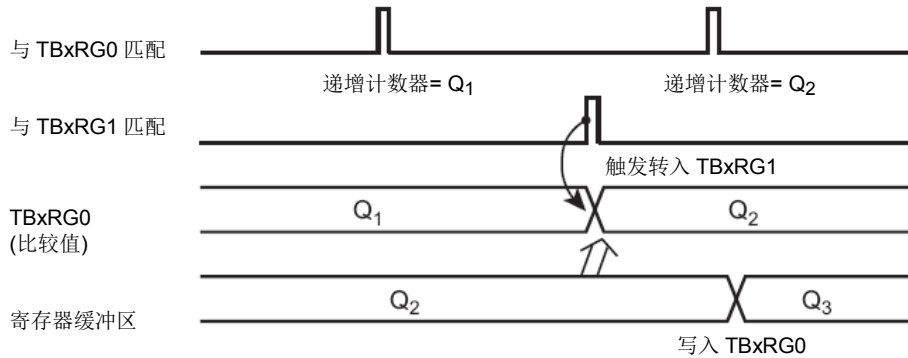


图 12-3 寄存器缓冲操作

该模式的方块图如以下所示。

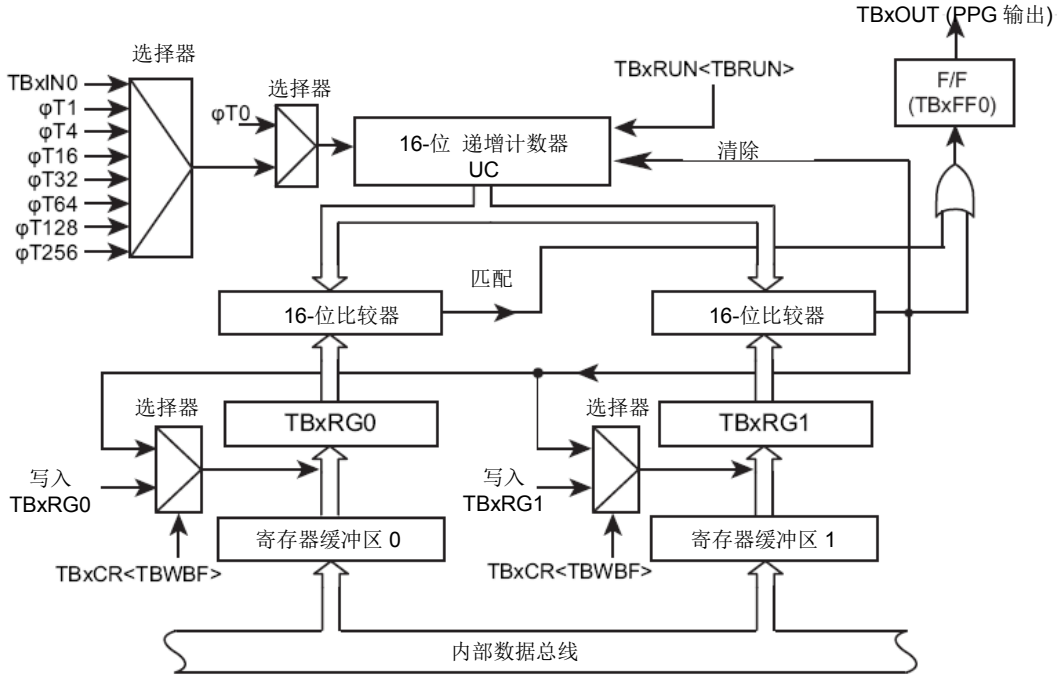


图 12-4 16-位 PPG 模式方块图

在该 16-位 PPG 输出模式下，各寄存器均必须按以下所列内容接受编程。

	7	6	5	4	3	2	1	0	
TBxEN	← 1	X	X	X	X	X	X	X	启用 TMRBx 运行。
TBxRUN	← X	X	X	X	X	0	X	0	停止计数运行。
TBxCR	← 0	0	-	X	-	X	X	X	禁用双缓冲。
TBxRG0	← *	*	*	*	*	*	*	*	指定某占空比(16 位)
TBxRG1	← *	*	*	*	*	*	*	*	指定一个周期。(16 位)
TBxCR	← 1	0	X	0	0	0	0	0	启用 TBxRG0 双缓冲。 (在发生 INTTBx 中断时，变更占空/周期)
TBxFFC R	← X	X	0	0	1	1	1	0	指定在检测到与 TBxRG0 或 TBxRG1 之间存在匹配时即触发 TBxFF0 反转，并将 TBxFF0 的初始值设置为"0"。
TBxMOD	← 0	1	0	0	1	*	*	*	指定该预分频输出时钟用作输入时钟，并禁用该捕捉功能。
									(** = 001 ~ 111)
PxCR[m]	←						1		
PxFR1[m]	←						1		将相应的端口分配到 TBxOUT。
TBxRUN	← *	*	*	*	*	1	X	1	启动 TMRBx。

注 1: m: 端口对应位
 注 2: X; 忽略
 -; 无变化

12.6.4 定时器同步模式

此模式使定时器同步启动。

如此模式与 PPG 输出一起使用，可采用输出驱动电机。

TMRB 是由两对 4 通道 TMRB 组成的。如果 1 个通道启动，剩下 3 个通道可同步启动。在 TMPM365FYXBG 中，允许使用下列组合。

起动触发器通道 (主机通道)	同步操作通道 (从机通道)
TMRB0	TMRB1, TMRB2, TMRB3
TMRB4	TMRB5, TMRB6, TMRB7

在 TBxCR<TBSYNC>位中指定使用定时器同步模式。

- * <TBSYNC> = "0": 定时器单独运行。
- * <TBSYNC> = "1": 定时器同步运行。

在主机通道把"0"设置为<TBSYNC>位。

如果在从机通道设置<TBSYNC>= "1", 启动时间与主机通道启动时间是同步的。不需要在从机通道设置 TBxRUN<TBPRUN, TBRUN>位的启动时间。

- 注 1: 把 TBxCR<TBSYNC>设置为"0", 除使用定时器同步模式外。定时器同步模式使其它通道操作处于等待状态, 直至 TMRB0 和 TMRB4 操作启动。
- 注 2: TMRB0 和 TMRB4 是定时器同步模式的主机时钟。因此, 必须把他们的<TBSYNC>位设置为"0"。
- 注 3: 不得把此模式应用于 TMRB8 和 TMRB9。

12.6.5 外部触发器计数起动模式

可通过外部信号把外部触发器计数起动模式设置为起动计数。

设置 TBxCR<CSSEL>, 选择计数起动模式。

- <CSSEL> = "0": 按照每个定时器通道的定时, 开始计数。
- <CSSEL> = "1": 通过外部信号开始计数。

设置 TBxCR<TRGSEL>, 选择外部触发信号活性边。

- <TRGSEL> = "0": 选择 TBxIN0 的上升缘。
- <TRGSEL> = "1": 选择 TBxIN0 的下降缘。

如果此模式是指定的, 定时器同步模式具有优先权。

12.7 利用捕捉功能的应用

该捕捉功能可用于改进多种应用 (包括以下所述):

1. 由外部脉冲触发的单次脉冲输出
2. 频率测量
3. 脉冲宽度测量
4. 时差测量

12.7.1 由外部脉冲触发的单脉冲

由外部脉冲触发的单次脉冲输出的实现方式如下:

利用预分频输出时钟, 使 16-位递增计数器在自由运行状态下运行并计数。外部脉冲是通过 TBxIN0 引脚输入的。利用该捕捉功能, 在该外部脉冲上升时生成一个触发信号, 该递增计数器的值随即被纳入该捕捉寄存器(TBxCP0)。

必须对 CPU 进行编程, 以使在外部触发脉冲上升时发生中断 INTCAPx0。此中断用于把定时器寄存器(TBxRG0)设置为 TBxCP0 值(c)和延迟时间(d), (c + d)的总和, 把定时器寄存器(TBxRG1)设置为 TBxRG0 值和单脉冲(c + d + p)的脉冲宽度(p)的总和。【TBxRG1 变更必须在下一次匹配前完成。】

此外, 必须将该计时器触发电路控制寄存器(TBxFFCR<TBE1T1, TBE0T1>)设置为"11"。这种情况用于在 TBxUC 与 TBxRG0 和 TBxRG1 匹配时, 触发定时器触发器(TBxFF0)发生逆转。单脉冲输出后, 通过 INTTBx 中断禁用触发器。

文本中使用的符号(c), (d) 和(p)与"图 12-5 单脉冲输出(延迟)"中的符号 c, d 和 p 相对应。

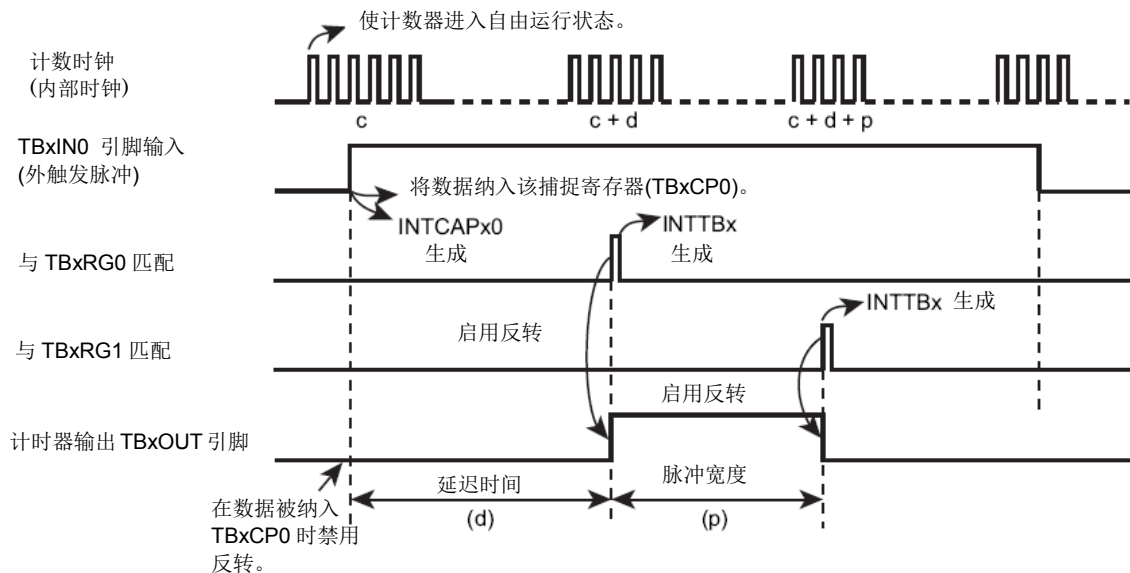


图 12-5 单脉冲输出 (延迟)

下列内容表示如果 2 ms 宽单脉冲是通过触发上升缘 TBxIN0 输入 3 ms 后输出时的设置。(ΦT1 选择用于计数。)

变更源时钟为 ΦT1。在 TBxIN0 上升缘获取计数值并纳入 TBxCP0。

	7	6	5	4	3	2	1	0		
[主过程] 由 TBxIN0 进行捕捉设置										
PxIE[m]	←							1	向 TBxIN0 分配对应端口	
PxFR1[m]	←							1		
TBxEN	←	1	X	X	X	X	X	X	启用 TMRBx 运行。	
TBxRUN	←	X	X	X	X	X	0	X	0	停止 TMRBx 运行
TBxMOD	←	0	1	0	1	0	0	0	1	把源时钟变更为 ΦT1。在 TBxIN0 上升缘获取计数值并纳入 TBxCP0。
TBxFFCR	←	X	X	0	0	0	0	1	0	清除 TBxFF0 反向触发器并禁用。
PxCR[m]	←							1		
PxFR1[m]	←							1	将相应的端口分配到 TBxOUT。	
中断设置启用寄存器	←	*	*	*	*	*	*	*	*	通过设置为"1", 允许生成由 INTCAPx0 中断对应位指定的中断。
TBxRUN	←	*	*	*	*	*	1	X	1	启动该 TMRBx 模块
[INTCAPx0 中断服务程序的处理] 脉冲输出设置										
TBxRG0	←	*	*	*	*	*	*	*	*	设定计数值。(TBxCAP0 + 3ms/ΦT1)
TBxRGI	←	*	*	*	*	*	*	*	*	设定计数值(TBxCAP0 + (3+2)ms/ΦT1)
TBxFFCR	←	X	X	-	-	1	1	-	-	如果 TBxRG0 与 TBxRGI 一致, 逆转 TBxFF0。
TBxIM	←	X	X	X	X	X	1	0	1	除 TBxRGI 对应中断以外的屏蔽。
中断设置启用寄存器	←	*	*	*	*	*	*	*	*	通过将相应的位设置为"1", 从而允许生成 INTTBx 中断所指定的中断。
[INTTBx 中断服务程序的处理] 禁止输出										
TBxFFCR	←	X	X	-	-	0	0	-	-	清除 TBxFF0 反向触发器设置。
中断启用清除寄存器	←	*	*	*	*	*	*	*	*	通过设置为"1", 禁止由 INTTBx 中断对应位指定的中断。

注 1: m: 端口对应位
 注 2: X; 忽略
 -; 无变化

如果不要求延时, 则通过生成 INTCAPx0 中断, TBxFF0 会在数据被纳入 TBxCP0 时反转, 且 TBxRG1 会被设置为该 TBxCP0 值(c)与该单次脉冲宽度(p)之和 (c + p)。必须在下一个匹配之前完成 TBxRG1 变换。

当 UC 与 TBxRG1 匹配时, TBxFF0 启用并通过生成 INTTBx 中断禁用。

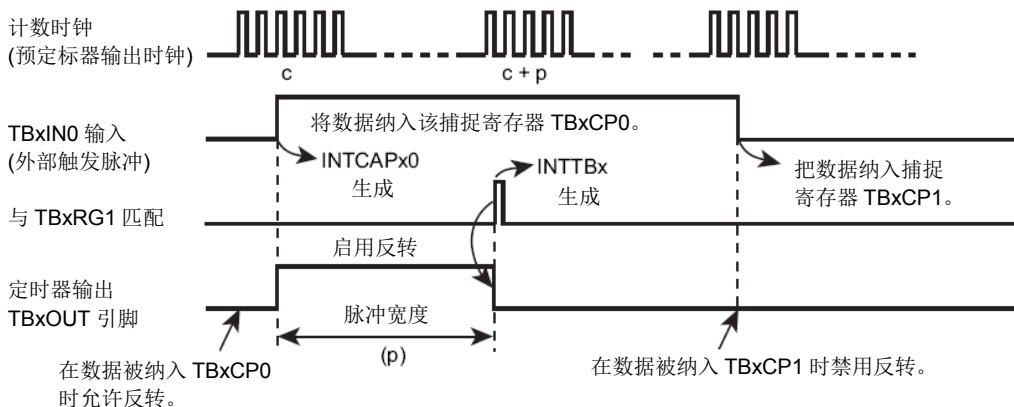


图 12-6 由外部脉冲触发的单脉冲输出(未延迟)

12.7.2 频率测量

可采用捕捉功能对外部时钟频率进行测量。

为测量频率，另一个 16-位定时器与 16-位事件计数器模式结合使用。例如，我们用 TMRB3 和 TMRB8 进行解释。16-位定时器 TMRB8 的 TB8OUT 用于指定测量时间。

TMRB3 计数时钟选择 TB3IN0 输入并通过采用外部时钟输入执行计数操作。如果把 TB3MOD<TBCPM[1:0]>设置为"11"，TMRB3 计数时钟在 TB8OUT 上升缘把计数器值纳入 TB3CP0，并在 TB8OUT 下边缘把计数器值纳入 TB3CP1。

此设置允许在 16-位定时器(TMRB8)的定时器触发器输出(TB8OUT)上升时把 16-位递增计数器 UC 的计数值纳入捕捉寄存器(TB3CP0)，并允许在 16-位定时器(TMRB8)的 TB8OUT 下降时把 UC 计数器值纳入捕捉寄存器(TB3CP1)。

然后，通过产生 INTTB8 的 16-位定时器中断，根据测量值从 TB3CP0 和 TB3CP1 之间的差值中获取频率。

例如，如果 TB3CP0 和 TB3CP1 之间的差值是 100，而且 TB8OUT 的电平宽度设置值是 0.5 s，则频率是 200 Hz ($100 \div 0.5 \text{ s} = 200 \text{ Hz}$)。

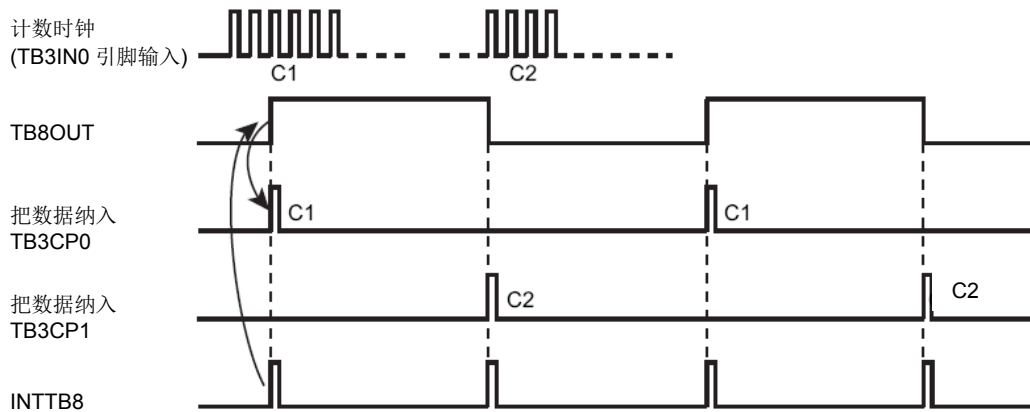


图 12-7 频率测量

12.7.3 脉冲宽度测量

通过使用该捕捉功能，可测量外部脉冲的"高"电平宽度。特别是，通过采用预分频器输出时钟将其纳入自由运转状态，外部脉冲通过 TBxIN0 引脚输入，同时使递增计数器(UC)进行递增相加。通过使用捕捉功能在外部脉冲的每次上升和下降时发生触发，而且把递增计数器的值纳入捕捉寄存器(TBxCP0, TBxCP1)。必须对 CPU 进行编程，以便通过 TBxIN0 引脚在外部脉冲输入下降缘时产生 INTCAPx1。

将 TBxCP0 与 TBxCP1 之差乘以内部时钟的时钟周期，即可计算得出该"高"电平脉冲宽度。

例如，如果 TBxCP0 和 TBxCP1 之间的差值是 100，且预分频器输出时钟的周期是 0.5 μs ，则脉冲宽度是 $100 \times 0.5 \mu\text{s} = 50 \mu\text{s}$ 。

在测量超过 UC 最长计数时间的脉冲宽度时，必须小心操作(其取决于所使用的源时钟)。必须用软件测量这种脉冲宽度。

也可测量外部脉冲的“低”电平宽度。这种情况下，可通过执行“图 12-8 脉冲宽度测量”中所示的第二阶段 INTCAPx0 中断处理，初步获取第一次发生的 C2 和第二次发生的 C1 之间的差值，此差值乘以预分频器输出时钟的周期，可得出“低”电平宽度。

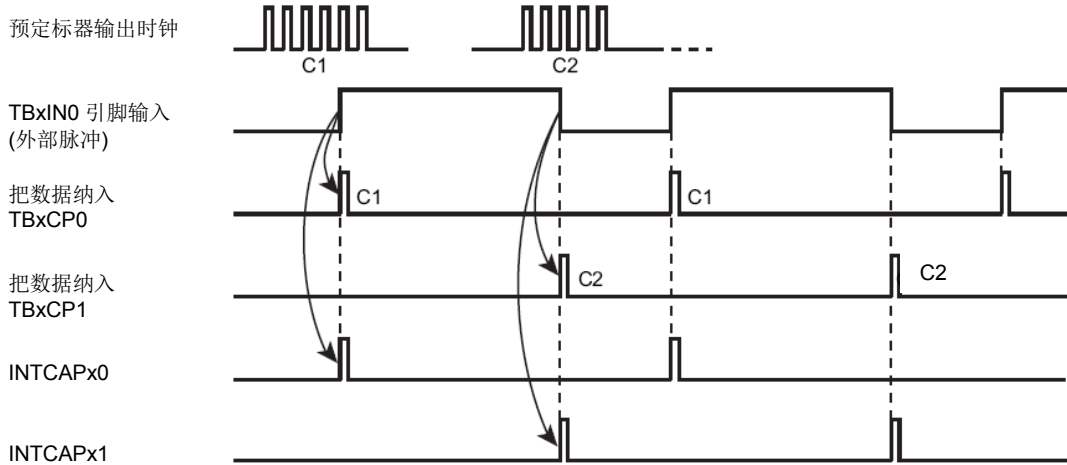


图 12-8 脉冲宽度测量

12.7.4 时间差测量

两个事件的时间差可通过捕捉功能测量。通过采用预分频器输出时钟将其纳入自由运转状态的方式，使递增计数器(UC)进行递增相加。

在 TBxIN0 引脚输入脉冲的上升缘把 UC 的值纳入捕捉寄存器(TBxCP0)。必须对 CPU 进行编程以在这时生成 INTCAPx0 中断。

在 TBxIN1 引脚输入脉冲的上升缘，把 UC 值纳入捕捉寄存器(TBxCP1)。必须对 CPU 进行编程，以在这时生成 INTCAPx1 中断。

可通过把 TBxCP1 和 TBxCP0 之间的差值乘以内部时钟的时钟周期，计算时间差。

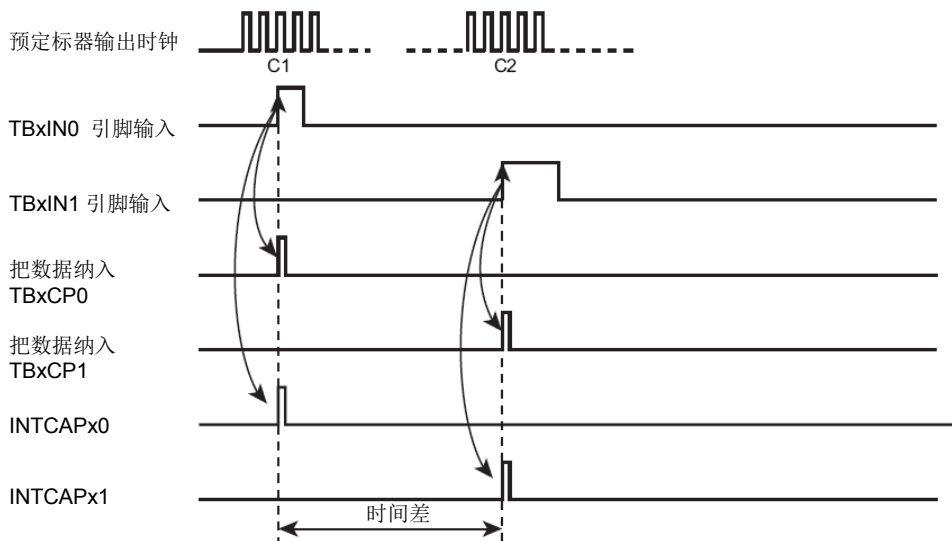


图 12-9 时间差测量

13. USB 装置控制器(USBD)

本节描述了有关 USB 装置控制器的内容。

在本节中，把端点描述为 EP。

13.1 概述

1. 符合通用串行总线规范 2.0 版。
2. 支持全速(不支持低速)。
3. USB 协议处理
4. 检测 SOF/USB_RESET/SUSPEND/RESUME.
5. 生成并检查数据包 ID。
6. 检查 CRC5。生成及检查 CRC16。
7. 支持 4 种传输模式(控制/中断/整体/等时)。
8. 支持 8 个 EPs。

表 13-1 端点

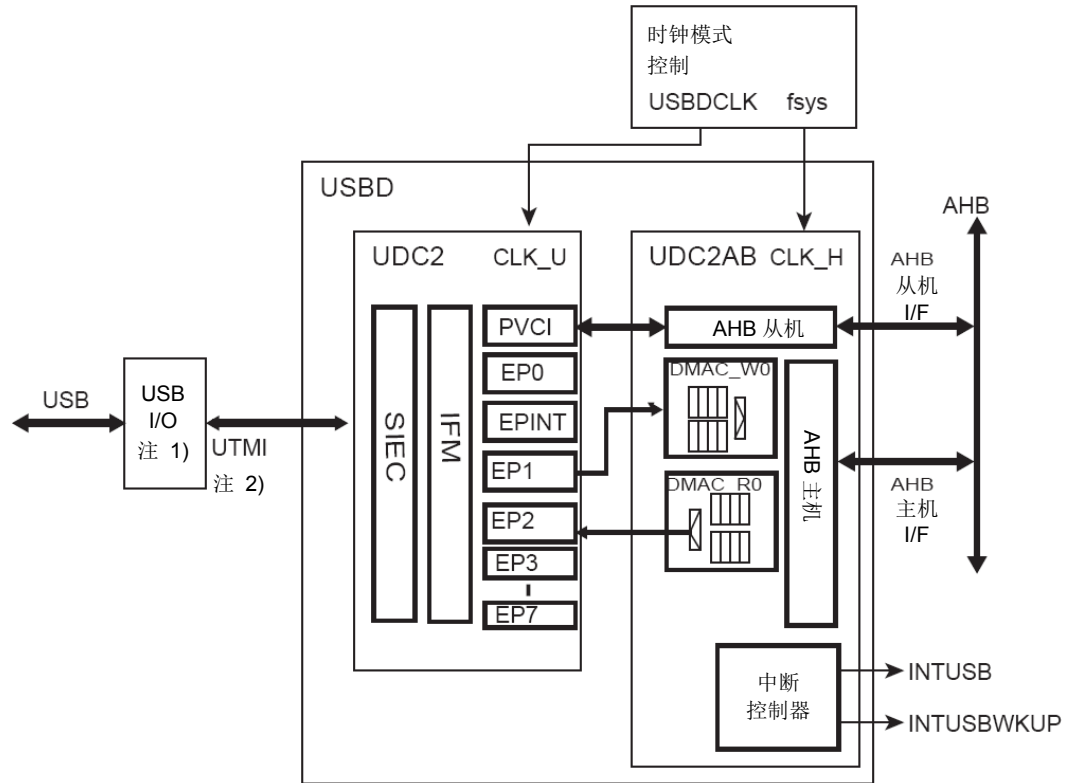
EP0:	控制	64 字节 x 1 FIFO
EP1:	控制 / 中断 / 整体 / 等时 (入)	64 字节 x 2 FIFO
EP2:	控制 / 中断 / 整体 / 等时 (出)	64 字节 x 2 FIFO
EP3:	控制 / 中断 / 整体 / 等时 (入)	64 字节 x 2 FIFO
EP4:	控制 / 中断 / 整体 / 等时 (出)	64 字节 x 2 FIFO
EP5:	控制 / 中断 / 整体 / 等时 (入)	64 字节 x 2 FIFO
EP6:	控制 / 中断 / 整体 / 等时 (出)	64 字节 x 2 FIFO
EP7:	控制 / 中断 / 整体 / 等时 (入)	64 字节 x 2 FIFO

9. 支持双数据包模式(除 EP 0 外)
10. 发送至中断控制器的中断源信号: INTUSB, INTUSBWKUP

13.2 系统结构

USB 装置控制器包括 USB-规格 2.0 装置控制器(以下称为 UDC2)和总线桥(以下称为 UDC2AB), 而总线桥用于把 UDC2 与 AHB 总线相连。

在本节中, "13.2.1 AHB 总线桥(UDC2AB)"描述了 UDC2AB 的配置, "13.2.2 东芝 USB-规格 2.0 装置控制器 (UDC2)" 描述了 UDC2 的配置。



注 1) TMPM365FYXBG 具有可用于全速而不是低速模式的 USB I/O。

本节中的字符"PHY"应读作 USB I/O。

注 2) USB 2.0 收发器宏单元接口

图 13-1 USB 装置控制器方块图

13.2.1 AHB 总线桥(UDC2AB)

UDC2AB 是东芝 USB-规格 2.0 装置 控制器(以下称为 UDC2)和 AHB 之间的总线桥。

UDC2AB 具有 DMA 控制器,该控制器支持 AHB 主机传输,并控制 AHB 上的指定地址和 UDC2 中的端点 FIFO(EP I/F)之间的传输。

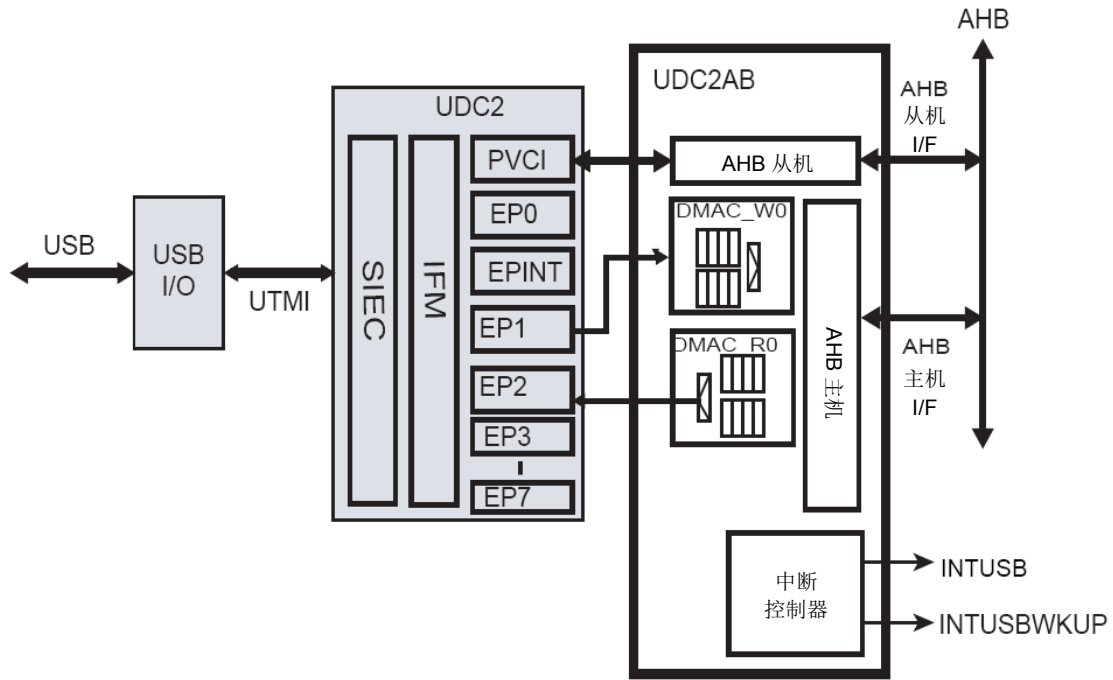


图 13-2 UDC2AB 方块图

13.2.1.1 功能和特性

UDC2AB 具有下述功能和特性：

1. 与 UDC2 相连接

对拟连接的 UDC2 的 EP 配置没有明确限制。但是，UDC2AB(AHB 主机功能)中的 DMA 控制器可与一个 Rx-EP 和一个 Tx-EP 相连。应采用 AHB 从机功能通过 UDC2 的 PPCI I/F 对其它 EP(含 EP0)进行访问。请注意不得通过 PPCI I/F 对主机与 DMA 控制器传输中 UDC2EP 的 EPx_FIFO 寄存器进行访问。

如果拟与 AHB 主机读取功能相连的 EP 的最大数据包尺寸是奇数，则对用途有一定的限制。有关更详细的信息，参见“(3)在主机读取传输中设置最大数据包尺寸”

2. AHB 功能

提供 AHB 主机和 AHB 从机功能。

a. AHB 主机功能

有两个 DMA 通道可用于 USB 装置控制器。分别位于 Rx-Ep 和 Tx-Ep。

表 13-2 AHB 主机功能

单次突发 (INCR/INCR8)传输	支持
分离传输	支持
从小到大	支持
保护控制	支持
早期突发终止	支持
地址总线宽度	32-位
数据总线宽度	32-位
字节字符传输	支持

字节存储次序转换图像如下所示。

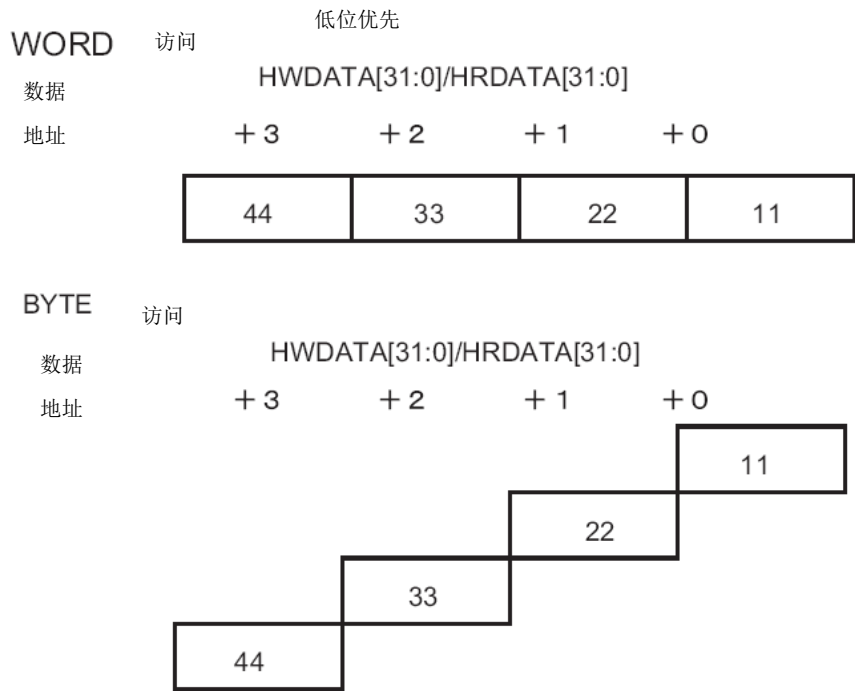


图 13-3 AHB 主机功能中的字节存储次序转换图像

b. AHB 从机功能

AHB 从机功能说明。

从小到大	支持
单次传输	支持
地址宽度	32-位
数据宽度	32-位
字节或字符中的传输	支持

字节存储次序转换图像如下所示。

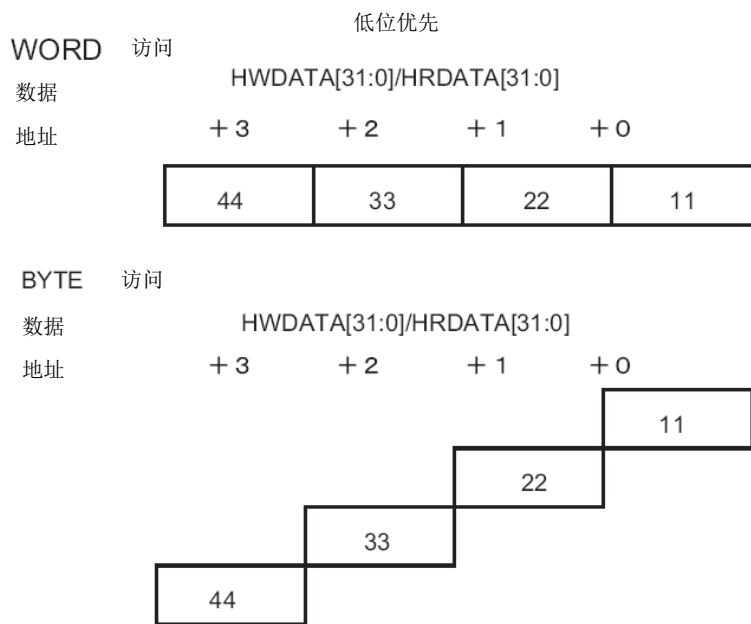


图 13-4 AHB 从机功能中的字节存储次序转换图像

13.2.1.2 配置

UDC2AB 主要由 AHB 从机功能和 AHB 主机功能组成,从机功能控制对 UDC2AB 内部寄存器和 UDC2 寄存器(UDC2 PPCI I/F)的访问,而主机功能控制 UDC2 EP I/F 的 DMA 访问。

AHB 主机功能具有两个内置通道: 主机读取通道(AHB ~ UDC2)和主机写入通道(UDC2 ~ AHB),这两个通道使得 DMA 在 Rx-EP 的 EP I/F 和 UDC2 的 Tx-EP 之间传输。每个通道有两个内置 8 字符缓冲区(共计 4 个)。

13.2.1.3 时钟域

由时钟/模式控制电路提供的 fsys 与 UDC2AB 的 CLK_H 相连.fsys 根据 TMPM365FYXBG 的低功耗模式停止或启动。

因为 CLK_H 不是在 fsys 于低功耗模式下处于停止状态时提供的,所以不会产生 INTUSB。

因此,为了检测 VBUS 的连接及断开,需要通过由 USBPON 引脚及 INTUSB 在 CLK_H 启动或停止时产生的 INTUSBPON 中选择拟使用的中断。

详情请参考"13.5.5.2 USB总线电源(VBUS)连接/断开顺序"。

时钟/模式控制电路提供的USBCLK连接到UDC2的CLK_U。USBCLK根据寄存器停止或开始。通过检测诸如挂起和重新开始来停止或启动CLK_U时，应利用软件对时钟/模式控制电路进行配置。

13.2.2 东芝 USB-规格2.0装置控制器(UDC2)

UDC2控制USB功能与通用串行总线之间的连接。UDC2自动处理USB协议，其PHY端接口可通过UTMI访问。

1. SIEC (序列接口引擎控制)块

控制块管理USB协议主要功能如下：

- 检查并生成PID
- 检查并生成CRC
- 检查装置地址

2. IFM块

块控制SIEC和EP主要功能如下：

- 接收OUT-Token时，将接收到的数据写入相应EP。
- 接收IN-Token时，读取相应EP的传输数据。
- 控制，管理UDC2.0的状态。

3. PVCI-I/F块

该控制块控制IFM和外部寄存器访问总线(PVCI)之间的数据读取和写入。

PVCI总线通过UDC2AB访问。

4. EP0块

该控制块控制"控制传输"过程中的数据发送和接收。利用控制传输的DATA-阶段发送和接收数据时，应通过PVCI-I/F访问本控制块中的FIFO。

5. EPx块

该控制块控制EPx(x = 1 ~ 7)数据的发送和接收。通过EP-I/F可直接访问FIFO。EP-I/F可进行突发数据传输。

请注意有两种EP：一种用来发送数据(EPTX)，另一种用来接收数据(EPRX)。EP的方向(发送/接收)将由硬件确定。

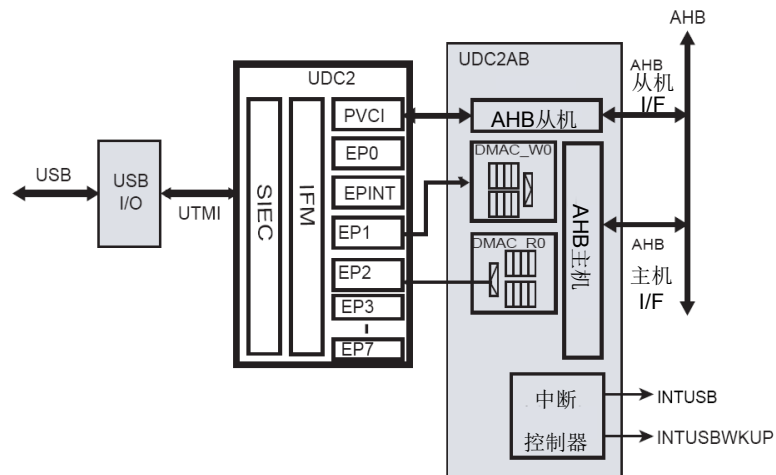


图 13-5 UDC2方块图

13.2.2.1 特点和功能

主要特点和功能如下：

1. 符合通用串行总线规范2.0版。
2. 支持全速(FS)(不支持低速)。
3. USB协议处理。
4. 检测SOF/USB_RESET/SUSPEND/RESUME。
5. 生成及检查封包ID。
6. 检查CRC5。生成及检查CRC16。
7. 支持 4 种传输模式(控制传输/中断传输/批量传输/等时传输)。
8. 支持高达 8 种EP。
9. 支持双封包传输模式(不包括EP0)。
10. EP 1 ~ 7 可直接访问FIFO(EP-I/F)。
11. 符合USB2.0收发器宏单元接口规范(UTMI)(48 MHz时 8 位)。

13.2.2.2 标志规范

发生事件时，UDC2核心单元在USB上输出各种事件作为标志。本节将针对这些标志进行讨论。

1. USB_RESET

接收USB_RESET, 启动"高"。由于接收USB_RESET会使UDC2返回到默认状态, 应用程序也须返回到默认状态。

全速运行情况下, 当USB总线上的SE0被识别2.5s或更长时间时, UDC2使该标志启用。UDC2驱动Chirp-K约1.5ms后, 在以下任一状态被识别时, 标志将解除启用:

- a. 主机的Chirp信号(K-J-K-J-K-J)被识别。
- b. 经过2ms或更长时间未识别主机的Chirp信号(K-J-K-J-K-J)。

注: 注意主机开始使用Chirp信号并且Chirp-K和Chirp-J的驱动时间取决于主机时, USB_RESET 标志的有效时间约1.74ms~3.5ms。

2. INT_SETUP

在控制传输过程中, 接收Setup-Token后, 启动"高"。识别中断时, 软件应读取Setup-Data存储寄存器(8 字节)以对请求作出判断。写入 1 到UDFS2INT<i_setup>即可使中断解除启用。识别中断的同时应清除UDFS2INT。

3. INT_STATUS_NAK

在控制传输过程中, 当主机进入STATUS-阶段并在UDC2处理DATA-阶段时传输封包时(发出"Setup_Fin"命令前), UDC2将返回"NAK", 并使标志"高"启动。识别此中断时, 软件应从命令寄存器发出"Setup_Fin"命令, 结束UDC2的STATUS-阶段。写入 1 到UDFS2INT<i_status_nak>, 即可使中断解除启用。识别中断的同时应清除UDFS2INT。

4. INT_STATUS

在控制传输过程中, 正常完成STATUS-阶段后, 启动"高"。写入 1 到UDFS2INT<i_status>, 即可使此中断解除启用。识别中断的同时应清除UDFS2INT。

5. INT_EP0

在控制传输的DATA-阶段, 发送或接收"ACK"时(交易正常结束时), 启动"高"。写入 1 到UDFS2INT<i_ep0>, 即可使此中断解除启用。识别中断的同时应清除UDFS2INT。

6. INT_EP

在EP中除EP0外，发送或接收"ACK"时(交易正常结束时)，"高"阶启用。在这种情况下，检查UDFS2INTEP便可识别哪个EP已被传输。写入 1 到UDFS2INT<i_ep>或写入 1 到UDFS2INTEP中的全部字节，即可使此中断无效。识别中断的同时应清除UDFS2INT。

7. INT_RX_ZERO

接收零-长度数据时，启用"高"。在控制传输过程中，只有在DATA-阶段接收到零-长度数据时，启用"高"。DATA-阶段接收到零-长度数据时，启用解除。读取UDFS2CMD<rx_nulpkt_ep>或检查UDFS2INTRX0可以识别哪个EP已接收数据。，写入1到UDFS2INT<i_rx_data0>或写入1到UDF2INTRX0的全部字节，即可使中断解除启用。识别中断的同时，应清除UDFS2INTRX0。

8. INT_SOF

接收SOF后，启用"高"。写入1到UDFS2INT<i_osf>，即可使此中断解除启用。识别中断的同时应清除UDFS2INT。

SOF是一种帧起始包。在全速传输过程中，SOF每隔1ms从主机传输到各装置。

9. INT_NAK

在EP中EP0除外，当传输NAK后，启用"高"。在这种情况下，检查UDFS2INTNAK可以识别哪个EP已传输NAK。写入1到UDFS2INT<i_nak>或写入1到UDFS2INTNAK的全部字节，即可使此中断解除启用。在默认情况下，传输NAK后，此标志将解除启用。因此，应写入0到UDFS2INTNAKMASK的相应EP，释放屏蔽，以使用此标志。

13.2.2.3 进入EP的命令

本节针对UDFS2CMD<ep>指定的EP就UDFS2CMD<com>发出的命令进行说明。

1. 0x0：保留

未规定。

2. 0x1：Setup_Fin

只向EP0发出此命令。

此命令是控制传输过程中设置DATA-阶段结束的命令。由于发出此命令前UDC2继续发送"NAK"到STATUS-阶段，当DATA-阶段结束或接收到INT_STATUS_NAK时，应发出命令。

注：注意控制-WR传输期间，首先读取Data-阶段接收到的所有数据，然后发出Setup-Fin命令。

3. 0x2：Set_DATA0

可向EP发出此命令EP0除外。不能向EP0发出此命令。

用于清除EP切换的命令。在正常传输下通过UDC2自动更新切换时，如需要通过软件清除，应发出该指令。

4. 0x03 : EP_Reset

可以向任一EP发出此命令。

用于清除EP数据和状态的命令。当希望在设置Set_Configuration和Set_Interface的EP或通过Clear_Feature复位EP等情况下使EP复位时，发出此命令。此命令将复位以下5点：

- a. 清除UDFS2EP0STS<toggle> / UDFS2EPxSTS<toggle>到DATA0。
- b. 清除UDFS2EP0STS<status> / UDFS2EPxSTS<status>到Ready。
- c. 清除UDFS2EP0MSZ<dset> / UDFS2EPxMSZ<dset>和UDFS2EP0DSZ / UDFS2EPxDSZ。
- d. 清除UDFS2EP0MSZ<tx0_data> / UDFS2EPxMSZ<tx_0data>。
- e. 清除UDFS2EPxSTS<disable>。

对各传输，UDC2通过硬件进行切换控制。如果在传输EP期间发出此命令，相应EP的切换也将被清除，这可能造成与主机丢失同步。如果接收到请求，当可能出现与主机同步的情况时，应发出此命令。

5. 0x4 : EP_Stall

可向EP发出此命令EP0除外。不能向EP0发出此命令。

将EP状态设置至"Stall"的命令。当希望在通过Set_Feature迟延某一EP等情况下将EP状态设置至"Stall"时，发出此命令。发出此命令时，"STALL"将始终发送回EP设置。但接收到Setup-Token后将清除EP0的Stall状态。

等时传输过程中，不能向EP发出此命令，因为在等时传输过程中不需要握手。如果在设置等时传输(通过UDFS2EOxSTS<t_type>)时向EP发送了此命令，将不会重发"STALL"。

6. 0x5 : EP_Invalid

可向EP发出此命令EP0除外。不能向EP0发出此命令。

将EP状态设置至"无效"的命令。当利用Set_Config 或 Set_Interface设置EP时禁用EP不再使用时，请发出此命令。发出此命令时，EP设置无反应。在传输各个EP期间，不应不得发出此命令。

7. 0x6 : 保留

未规定。

8. 0x7 : EP_Disable

可向EP发出此命令EP0除外。不能向EP0发出此命令。

使EP禁用的命令。发出此命令后，"NAK"将始终从EP设置返回。等时传输过程中，不能向EP发出此命令，因为在等时传输过程中不需要握手。如果在设置等时传输(通过UDFS2EPxSTS<t_type>)时向EP发送了此命令，将不会重发"NAK"。

9. 0x8 : EP_Enable

可向EP发出此命令EP0除外。不能向EP0发出此命令。

使EP启用的命令。发出此命令即可取消"EP_Disable"命令设置的禁用状态。

10. 0x9 : All_EP_Invalid

EP设置无效。

此命令是将所有EP(EP0除外)的状态设置成"无效"。当希望向所有EP发送"EP_Invalid"命令时,发出此命令。当将Set_Configuration和Set_Interface处理成与"EP_Invalid"命令一样时,发出此命令。

11. 0xA : USB_Ready

只向EP0发出此命令。

连接USB线的命令。确认与USB线连接并与主机通信成功后,发出此命令。只能在发出此命令后上拉D+数据,数据线连接状态将发送到主机。

请注意发出此命令后,UDC2(UDFS2ADR<configured> <addressed> <default>)的装置状态将被设置至"默认"。

12. 0xB : Setup_Received

向EP0发出此命令。

通知UDC2控制传输的SETUP-阶段已被识别的命令。接收INT_SETUP中断及识别请求代码后,发出此命令。由于发出此命令前UDC2继续将"NAK"发送回DATA-阶段/STATUS-阶段,应在INT_SETUP中断处理程序结束时发出此命令。

13. 0xC : EP_EOP

可以向任一EP发出此命令。

通知UDC2传输数据已被写入的命令。当传输其字节数小于最大传输字节数的数据时(EP或MaxPacketSize的FIFO容量,取较小值),发出此命令。发出此命令后将设置数据标志,并且数据将从主机发送回IN-Token。设置零-长度数据或MaxPacketSize数据时,不能使用此命令。

14. 0xD : EP_FIFO_Clear

可以向任一EP发出此命令。

清除EP数据的命令。UDFS2EPxMSZ<dset>和UDFS2EPxDSZ将同时被清除。例如在中断传输过程中,如果打算将FIFO当前存储的数据清除,发出此命令,然后传输数据到主机,并设置最新数据。如果在访问EP-I/F时发出此命令,EP的FIFO将不会被成功清除。发出此命令时,EP-I/F的epx_val应设置为0,然后发出命令。

15. 0xE : EP_TX_0DATA

可以向任一EP发出此命令。

将零-长度数据设置至EP的命令。发出此命令,即可传输零-长度数据。在Bulk-IN或其他传输过程中传输零-长度数据来表示最终传输时,读取UDFS2EPxDSZ,确认其值为0(EPx的FIFO中没有数据),然后设置此命令。从EP-I/F写入数据时,在数据被写入且epx_val变为0时设置此命令。设置此命令时,EP的UDFS2EPxMS<tx_0data>也将被设置。

确认UDFS2EPxMS<tx_0data>为0后，设置下一个数据。等时-IN传输过程中，如果EP的FIFO中未设置数据，零-长度数据将自动传输到IN-Token。不要发出此命令。

16. 0xF：保留

未规定。

USB传输过程中发出命令时，下列命令将暂停设置，并且在USB传输完成后继续设置。每个EP均会出现命令暂停的情况。

0x2: Set_DATA0
 0x3: EP_Reset
 0x4: EP_Stall
 0x5: EP_Invalid
 0x7: EP_Disable
 0x8: EP_Enable
 0x9: All_EP_Invalid
 0xD: EP_FIFO_Clear
 0xE: EP_TX_0DATA

因此，在USB传输过程中相继向同一EP发出各个命令时，这些命令将被覆盖，只有最后一个发出的命令有效。如果需要向某一EP相继发出命令，应在发出下一个命令前轮询UDFS2EPxSTS/UDFS2EPxDSZ，确认该命令有效。此外，利用EP_Reset/EP_FIFO_Clear命令清除FIFO后立即访问EP-I/F时，在重新开始访问EP-I/F前，轮询UDFS2EPxDSZ，确认该命令有效。

对于EP0，接收Setup-Token后发出Setup_Received命令前，下列命令无效：

0x1: Setup_Fin
 0x2: Set_DATA0
 0x3: EP_Reset
 0x4: EP_Stall
 0xC: EP_EOP
 0xD: EP_FIFO_Clear
 0xE: EP_TX_0DATA

当"EP_Stall"命令设置为EPx时，"Stall"将被设置为UDFS2EPxSTS<status>。设置EP_Disable时，1将被设置至UDFS2EPxSTS<disable>。当这两个命令(EP_Stall和EP_Disable)被设置为同一个EPx，状态变为"Stall"，并且UDFS2EPxSTS<disable>=1时，"STALL"将在传输过程中被传输。

"EP_Invalid"命令被设置为EPx时，"Invalid"将被设置至UDFS2EPxSTS<status>。当两个命令(EP_Invalid和EP_Disable)被设置至同一个EPx，状态变为"Invalid"，并且UDFS2EPxSTS<disable>=1时，传输过程将无反应。

当UDFS2EPxSTS<disable>和UDFS2EPxMSZ<tx_0data>均至1时，零-长度数据将在传输过程中被一次性传输。零-长度数据被成功传输后，将传输"NAK"。

13.3 如何连接USB总线

以下电路显示TMPM365FYXBG如何与USB总线连接。

输入VBUS信号到USBPON引脚，检测USB电源(VBUS)是否连接妥当。

需要利用USB-DDP上拉电阻和线内阻尼电阻到USB-DDM的"上拉"过程。此外，上拉电阻需添加一个利用端口的ON/OFF控制。未向VBUS施加电压时，应断开上拉电阻。

如果USB-DDP和USB-DDM不稳定，建议R₁增加一个下拉电阻。

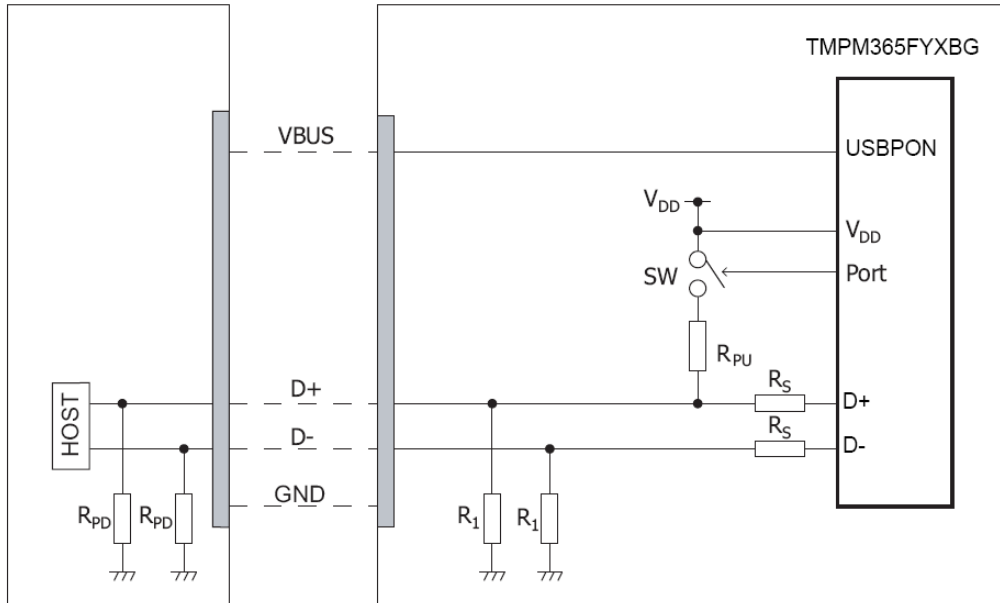


图13-6 USB总线和TMPM365FYXBG的连接图例

备注: R₁=500 kΩ或以上(建议值), R_S=33 Ω(建议值), R_{PU}=1.5 kΩ(建议值)

13.4 寄存器

USB的寄存器地址包括设置UDC2AB和UDC2的各个寄存器。

访问设置UDC2的寄存器时，UDC2AB将通过PVC I/F自动访问UDC2。

设置UDC2AB的寄存器是一个 32-位宽寄存器。设置UDC2的寄存器是一个16-位宽的寄存器，分配到[15:0]。[31:16]表示只读，为不确定值。

13.4.1 UDC2AB寄存器

13.4.1.1 UDC2AB寄存器列表

基址 =0x4000_8000

寄存器名称		地址(基+)
中断状态寄存器	UDFSINTSTS	0x0000
中断启用寄存器	UDFSINTENB	0x0004
主机写入超时寄存器	UDFSMWTOUT	0x0008
UDC2设置寄存器	UDFSC2STSET	0x000C
DMAC设置寄存器	UDFSMSTSET	0x0010
DMAC读取请求寄存器	UDFSDMACRDREQ	0x0014
DMAC读取值寄存器	UDFSDMACRDVL	0x0018
UDC2读取请求寄存器	UDFSUDC2RDREQ	0x001C
UDC2读取值寄存器	UDFSUDC2RDVL	0x0020
-	保留	0x0024~0x0038 (注 2)
判优器设置寄存器	UDFSARBTSET	0x003C
主机写入起始地址寄存器	UDFSMWSADR	0x0040
主机写入结束地址寄存器	UDFSMWEADR	0x0044
主机写入当前地址寄存器	UDFSMWCADR	0x0048 (注 1)
主机写入AHB地址寄存器	UDFSMWAHBADR	0x004C
主机读取起始地址寄存器	UDFSMRSADR	0x0050
主机读取结束地址寄存器	UDFSMREADR	0x0054
主机读取当前地址寄存器	UDFSMRCADR	0x0058 (注 1)
主机读取AHB地址寄存器	UDFSMRAHBADR	0x005C
-	保留	0x0060~0x007C (注 2)
功率检测控制寄存器	UDFSPWCTL	0x0080
主机状态寄存器	UDFSMSTSTS	0x0084
超時計数寄存器	UDFSTOUTCNT	0x0088 (注 1)
-	保留	0x008C~ 0x1FC

注1: 确保通过UDFSMACRDREQ进行"读取"访问。

注2: 表中的"保留"栏禁止访问。禁止在"保留"栏内读取/写入。

13.4.1.2 UDFSINTSTS(中断状态寄存器)

	31	30	29	28	27	26	25	24
比特符号	-	-	int_mw_rerro r	int_ powerdetect	-	-	int_dmac_ reg_rd	int_udc2_ reg_rd
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	int_mr_ahber r	int_mr_ep_ dset	int_mr_end_ add	int_mw_ ahberr	int_mw_ timeout	int_mw_end_ _add	int_mw_set_ add	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	int_usb_ reset_end	int_usb_reset	int_suspend_ _resume
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	int_nak	int_ep	int_ep0	int_sof	int_rx_zero	int_status	int_status_ nak	int_setup
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-30	-	R	不能读取。
29	int_mw_rerror	R/W	设置公用总线访问期间如果对EP的访问已经开始主机写入传输, 将被设置为1(UDFS2EPxSTS<bus_sel> 为0)(UDFS2EPxSTS<bus_sel>为0) 0: 未检测 1: 主机写入期间发生EP读取错误
28	int_ powerdetect	R/W	当UDC2AB的VBUSPOWER输入状态发生变化时, 将被设置为"1" 0: 未改变 1: 状态改变
27-26	-	R	不能读取。
25	int_dmac_ reg_rd	R/W	当通过设置UDFSDMACRDREQ而执行的寄存器访问完成时以及设置UDFSDMACCRDVL读取数值时, 将被设置为1 0: 未检测 1: 寄存器读取完成
24	int_udc2_ reg_ rd	R/W	当通过设置UDFSDMACRDREQ而执行的UDC2访问完成时以及设置UDFSDMACRDVL读取数值时, 将被设置为 1 当写入访问UDC2内部寄存器完成时, 也将被设置为 1 0: 未检测 1: 寄存器读取/写入完成
23	int_mr_ahberr	R/W	当主机读取传输操作期间发生AHB错误时, 此状态将被设置为1 发生此中断后, 主机写入传输控制块需通过UDFSMSTSET<mr_reset >复位。 0: 未检测 1: 发生AHB错误
22	int_mr_ep_ dse t	R/W	当在主机写入传输中使用的UDC2 Tx的EP FIFO可写(不全)时, 将被设置至1 0: FIFO不可写 1: FIFO可写
21	int_mr_end_ add	R/W	当主机读取传输完成时, 将被设置至1 0: 未检测 1: 主机读取传输完成
20	int_mw_ahberr	R/W	当主机写入传输操作期间发生AHB错误时, 此状态将被设置至1 发生此中断后, 主机写入传输控制块需通过UDFSMSTSET<mw_reset>复位 0: 未检测

			1: 发生AHB错误
--	--	--	------------



位	比特符号	类型	功能
19	int_mw_timeout	R/W	若在主机写入传输操作期间超时，此状态将被设置至1 0: 未检测 1: 主机写入传输超时
18	int_mw_end_add	R/W	当主机写入传输完成时，将被设置至1 0: 未检测 1: 主机写入传输超时
17	int_mw_set_add	R/W	当主机写入传输被禁用的同时通过主机写入传输发送的数据被设置至Rx相应EP时，此状态将被设置至1 0: 未检测 1: 主机写入传输地址请求
16-11	-	R	不能读取。
10	int_usb_reset_end	R/W	表示UDC2是否已使usb_reset信号解除启用 UDC2在USB_RESET是在usb_reset信号解除启用后将UDC2寄存器设置至初始值的计时。采用此位检测时间。 usb_reset信号状态可利用UDFSPWCTL<usb_reset>进行检查 0: 清除此位后，UDC2未使usb_reset信号解除启用 1: 表示UDC2已经使usb_reset信号解除启用
9	int_usb_reset	R/W	表示UDC2是否已使usb_reset信号有效。usb_reset信号状态可利用UDFSPWCTL<usb_reset>进行检查 0: 清除此位后，UDC2未使usb_reset信号有效。 1: 表示UDC2已使usb_reset信号有效
8	int_suspend_resume	R/W	每当UDC2的suspend_x信号改变时，1启用。此状态可以利用UDFSPWCTL<suspend_x>进行检查。 0: 状态未变 1: 状态已变
7	int_nak	R	可直接读取UDC2的int_nak信号。若要将其清除，应将UDC2的UDFS2INT或UDFS2INTNAK的对应位清除
6	int_ep	R	可直接读取UDC2的int_ep信号。若要将其清除，应清除UDFS2INT或UDFS2INTEP的对应位
5	int_ep0	R	可直接读取UDC2的int_ep0信号。若要将其清除，应清除UDFS2INT的对应位
4	int_sof	R	可直接读取UDC2的int_ep0信号。若要将其清除，应清除UDFS2INT的对应位
3	int_rx_zero	R	可直接读取UDC2的int_rx_zero信号。若要将其清除，应清除UDFS2INT或UDFS2INTRX0的对应位
2	int_status	R	可直接读取UDC2的int_status信号。若要将其清除，应清除UDFS2INT的对应位
1	int_status_nak	R	可直接读取UDC2的int_status_nak的信号。若要将其清除，应清除UDFS2INT的对应位
0	int_setup	R	可直接读取UDC2的int_setup信号。若要将其清除，应清除UDFS2INT的对应位

UDC2输出信号与此寄存器的[10:9]位和[7:0]位之间的连接如下所示:

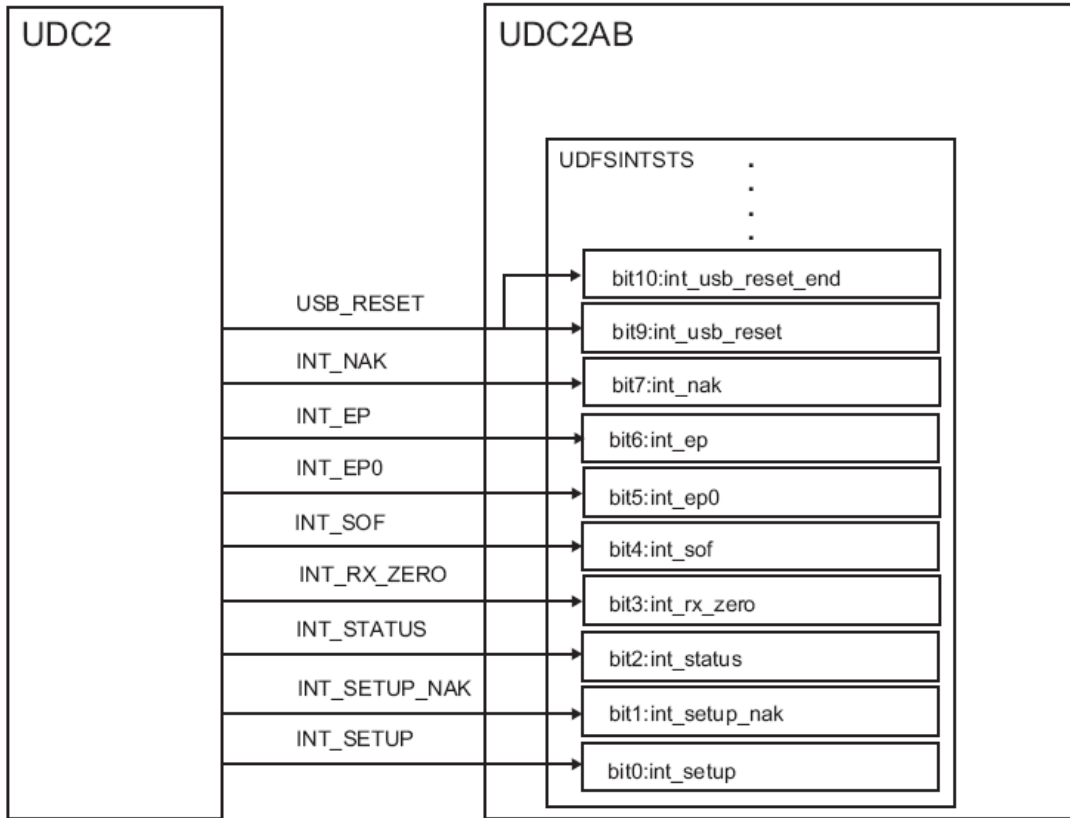


图 13-7 标志输出信号与中断位之间的连接图。

13.4.1.3 UDFSINTENB(中断启用暂存器)

	31	30	29	28	27	26	25	24
比特符号	-	-	mw_rerror_en	power_detect_en	-	-	dmac_reg_rd_en	udc2_reg_rd_en
复位后	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
比特符号	mr_ahberr_en	mr_ep_dset_en	mr_end_add_en	mw_ahberr_en	mw_timeout_en	mw_end_add_en	mw_set_add_en	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	-	-	-	-	-	usb_reset_end_en	usb_reset_en	suspend_resume_en
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
31-30	-	R	不能读取。
29	mw_rerror_en	R/W	控制mw_rerror中断 0: 禁用 1: 启用
28	power_detect_en	R/W	控制power_detect中断 0: 禁用 1: 启用
27-26	-	R	不能读取。
25	dmac_reg_rd_en	R/W	控制dmac_reg_rd中断 0: 禁用 1: 启用
24	udc2_reg_rd_en	R/W	控制udc2_reg_rd中断 0: 禁用 1: 启用
23	mr_ahberr_en	R/W	控制mr_ahberr中断 0: 禁用 1: 启用
22	mr_ep_dset_en	R/W	控制mr_ep_dset中断 0: 禁用 1: 启用
21	mr_end_add_en	R/W	控制mr_end_add中断 0: 禁用 1: 启用
20	mw_ahberr_en	R/W	控制mw_ahberr中断 0: 禁用 1: 启用
19	mw_timeout_en	R/W	控制mw_timeout中断 0: 禁用 1: 启用
18	mw_end_add_en	R/W	mw_end_add中断 0: 禁用 1: 启用
17	mw_set_add_en	R/W	mw_set_add中断 0: 禁用 1: 启用

16-11	-	R	不能读取。
-------	---	---	-------



位	比特符号	类型	功能
10	usb_reset_end_en	R/W	usb_reset_end中断 0: 禁用 1: 启用
9	usb_reset_en	R/W	usb_reset中断 0: 禁用 1: 启用
8	suspend_resume_en	R/W	suspend_resume中断 0: 禁用 1: 启用
7-0	-	R	不能读取。

13.4.1.4 UDFSMWTOUT(主机写入超时寄存器)

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

位	比特符号	类型	功能
31-1	timeoutset	R/W	<p>主机写入传输期间，不应更改此设置。在主机写入(Rx)EP数据结束后计数CLK_U的设置次数时，出现超时。</p> <p>超时计数器包括 32 位，前 31 位可通过寄存器的timeoutset [31:1]进行设置，而计数器的最后一位设置至1。</p> <p>当CLK_U为 48 MHz时，可以设置约 20 [ns] ~ 89 [s]作为超时值。</p> <p>当CLK_U停止时(PHY暂停等)，当计数器不工作时，不会发生超时中断。</p>
0	timeout_en	R/W	<p>用来启用主机写入超时。它被设置至默认启用。</p> <p>主机写入传输期间，不得更改此设置。</p> <p>0: 禁用 1: 启用</p>

13.4.1.5 UDFSC2STSET(UDC2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	-			eopb_enable	-	-	-	tx0
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-5	-	R	不能读取。
4	eopb_enable	R/W	<p>用于启用主机读取EOP。它被设置至默认启用。主机读取传输期间，不能改变此设置。</p> <p>如果位为 0，若最后一个字是 1 个字节，最终数据不会传输到UDC2。如果最后一个字是 2 字节，当epx_w_eop = 0时，最终数据将传输到UDC2。</p> <p>如果此位是 1，当epx_w_eop = 1时，无论最后一个字的字节数是多少，最终数据都会传输给UDC2。</p> <p>注： 详见"13.5.4.1 主机读取传输"一节。</p> <p>0: 主机读取EOP禁用 1: 主机读取EOP启用。</p>
3-1	-	R	不能读取。
0	tx0	R/W	<p>用于在EP连接至主机读操作侧传输NULL数据包。只在UDFSMSTSTS<mrepempty>为 1 时有效，否则该位被忽略。写入后它将自动被清除至0。</p> <p>将该位设置为 1 将显示UDC2 EP-I/F的epx_tx0data信号，1值在NULL数据包传输中被保留。设置该位后，Tx-EP下一数据在清除前无需再设置。</p> <p>0: 无运算。 1: 传输NULL数据包。</p>

13.4.1.6 UDFSMSTSET(DMAC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	-	-	-	-	-	-	-	m_burst_type
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	-	mr_reset	mr_abort	mr_enable	-	mw_reset	mw_abort	mw_enable
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-9	-	R	读取未定义
8	m_burst_type	R/W	<p>在主机写入/读取传输过程中进行突发数据传输时，选择HBURST[2:0]类型。UDC2AB的突发传输类型为INCR8(递增型 8 次突发传输)。在正常情况下也应相应设置 0(初始值)。但按照系统AHB规范，当INCR只能用作突发传输的类型时，设置 1 至此位。在这种情况下，UDC2AB将进行 8 次INCR传输。</p> <p>请注意，不能改变突发传输次数。</p> <p>在UDC2AB的初始设置中应设置此位。主机写入/读取传输开始后，不能改变此设置。</p> <p>注：UDC2AB只有在主机写入/读取传输过程中不能进行突发传输。它结合了突发传输和单传输特点。此位只影响突发传输过程。</p> <p>0: INCR8 1: INCR</p>
7	-	R	读取未定义
6	mr_reset	R/W	<p>初始化UDC2AB的主机读取传输控制块。但由于EP的FIFO未初始化，必须访问UDC2的UDFS2CMD，以便从此复位单独初始化相关EP。</p> <p>停止主机运行后应进行此复位。</p> <p>此位设置为 1 后将自动清0。清除前，不应进行后续的主机读取传输。</p> <p>0: 无运行 1: 复位</p>
5	mr_abort	W	<p>控制主机读取传输。设置 1 到此位可以停止主机读取运行。</p> <p>传输过程中取消后，主机读取缓冲区到UDC2的传输中断，且<mr_enable>位被清除，导致主机读取传输停止。</p> <p>将此位设置为1后，<mr_enable>位被禁用且为 0 时，取消完成。</p> <p>0: 无运行 1: 取消</p>
4	mr_enable	R/W	<p>控制主机读取传输。当传输地址设置完成时启用。</p> <p>当主机传输完成时将自动禁用。由于主机读取运行不能随此寄存器一起禁用，应采用<mr_abort> 位停止主机读取传输。</p> <p>0: 禁用 1: 启用</p>
3	-	R	读取未定义
2	mw_reset	R/W	<p>初始化主机写入传输控制块。但由于EP的FIFO未初始化，必须访问UDC2的UDFS2CMD，以便从此复位单独初始化相关EP。</p> <p>停止主机运行后应进行此复位。</p> <p>此位设置为 1 后将自动清0。清除前，不应进行后续的主机读取传输。</p> <p>0: 无运行 1: 复位</p>
1	mw_abort	W	<p>控制主机写入传输。设置 1 到此位不能停止主机写入运行。</p> <p>传输过程中取消后，主机写入缓冲区从UDC2的传输中断，且<mw_enable>位被清除，导致主机写入传输停止。将此位设置为 1 后，<mw_enable>位被禁用且为 0 时，取消完成。</p> <p>0: 无运行 1: 取消</p>
0	mw_enable	R/W	<p>控制主机写入传输。当传输地址设置完成时启用。</p> <p>当主机传输完成时将自动禁用。由于主机写入运行不能随此寄存器一起禁用，应采用<mw_abort> 位停止主机写入传输。</p> <p>0: 禁用 1: 启用</p>

13.4.1.7 UDFSDMACRDREQ(DMAC读取请求寄存器)

	31	30	29	28	27	26	25	24
比特符号	dmardreq	dmardclr	-	-	-	-	-	-
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	dmardadr						-	-
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31	dmardreq	R/W	此位用于请求对DMAC寄存器进行读取访问。将此位设置为 1 即可对<dmardadr>指定的地址进行读取访问。当读取访问完成且读取值存储在UDFSDMACRDVL时，此位将自动清除，UDFSINTSTS<dmac_reg_rd>也将被设置为 1。 0: 无运行 1: 发送读取请求
30	dmardclr	R/W	此位用于强制清除与DMAC相关的寄存器读取访问请求。将此位设置为 1，即可通过<dmardreq>停止寄存器读取访问请求，<dmardreq>的数值将被清零。强制清除完成后，此位将自动清除。 0: 无运行 1: 发送强制清除请求
29-8	-	R	读取未定义
7-2	dmardadr[5:0]	R/W	设置将要读取的寄存器地址(高 6 位)。应结合上述的<dmardreq>进行设置。应设置以下所有地址。 0x48: 读取UDFSMWCADR 0x58: 读取UDFSMRCADR 0x88: 读取UDFSTOUTCNT
1-0	-	R	读取未定义

13.4.1.8 UDFSDMACRDVL(DMAC读取值寄存器)

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

位	比特符号	类型	功能
31-0	dmardata[31:0]	R	此寄存器存储UDFSDMACRDREQ请求的数据。UDFSDMACRDREQ<dmardreq>设置为1时，不能访问此寄存器。

13.4.1.9 UDFSUDC2RDREQ(UDC2读取请求寄存器)

	31	30	29	28	27	26	25	24
比特符号	udc2rdreq	udc2rdclr	-	-	-	-	-	-
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	-	-	-	-	-	-	udc2rdadr	
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	udc2rdadr						-	-
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31	udc2rdreq	R/W	此位用于请求对UDC2寄存器进行读取访问。将此位设置为 1 即可对udc2rdadr位中设置的地址进行读取访问。当读取访问完成且读取值设置为UDFSDMACRDVL时，此位将自动清除，中断状态寄存器中的UDINTSTS<int_udc2_reg_rd>位将被设置为 1。 对UDC2寄存器进行写入访问过程中，它将作为状态位工作，表示正在访问，并显示数值 1。此位设置为 1 时，不能对UDC2寄存器进行其他访问。 0: 无运行 1: 发送读取请求
30	udc2rdclr	R/W	此位用来强制清除UDC2寄存器的读取/写入访问请求。将此位设置为 1，即可通过udc2rdreq强制停止寄存器读取请求/UDC2写入访问，udc2rdreq的数值将被清0。强制清除完成后，此位将自动清除。发生中断时，不会保存访问过程中的读取值和写入值。 0: 无运行 1: 发送强制清除请求
29-10	-	R	读取未定义
9-2	udc2rdadr[7:0]	R/W	设置要读取的UDC2寄存器[9:2]的地址。有关UDC2的寄存器地址的详细信息请参考"13.4.1.1 UDC2AB寄存器列表"。以上寄存器列表(0x0200 ~ 0x0334)中的偏移地址相互关联。用上述udc2rdreq进行设置。
1-0	-	R	读取未定义

13.4.1.10 UDFSUDC2RDVL(UDC2读取值寄存器)

	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	-	-	-	-	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8
比特符号	udc2rdata							
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	udc2rdata							
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-16	-	R	读取未定义
15-0	udc2rdata[15:0]	R	此寄存器存储UDFSDMACRDREQ请求的数据。当UDFSUDC2RDREQ <udc2rdreq>设置为 1 时，不能访问此寄存器。

13.4.1.11 UDFSARBTSET(判优器设置寄存器)

	31	30	29	28	27	26	25	24
比特符号	abt_en	-	-	abtmod	-	-	-	-
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	-	-	abtpri_w1		-	-	abtpri_w0	
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	-	-	abtpri_r1		-	-	abtpri_r0	
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31	abt_en	R/W	<p>当在DMAC和AHB之间进行访问时，启用判优器运行。</p> <p>设置此寄存器的<abtmod>，<abtpri_w1>，<abtpri_w0>，<abtpri_r1>和<abtpri_r0>时，此位应被设置为0。</p> <p>必须在开始DMA访问前将此位设置为1。</p> <p>0：禁用(不允许DMA访问)</p> <p>1：启用</p>
30-29	-	R	读取未定义
28	abdmod	R/W	<p>设置判优器的判优模式。当<abt_en>位设置为 0 时，才能进行写入访问。</p> <p>当此位设置为 0 时，将以轮询的方式给出AHB总线的访问权，无论各个<abtpri_w1>，<abtpri_w0>，<abtpri_r1>和<abtpri_r0>位设置了什么值。</p> <p>如果此位设置为 1，将根据设置到各个<abtpri_w1>，<abtpri_w0>，<abtpri_r1>和<abtpri_r0>位的值按访问优先级给出AHB总线的访问权。</p> <p>0：轮询</p> <p>1：固定优先级</p>
27-14	-	R	读取未定义
13-12	abtpri_w1	R/W	<p>当选择固定优先级模式时，设置主机写入 1 的DMA访问优先级。当<abt_en>位设置为 0 时，才能进行写入访问。</p> <p>优先级范围：00 (最高级) ~ 11 (最低级)。</p>
11-10	-	R	读取未定义
9-8	abtpri_w0	R/W	<p>当选择固定优先级模式时，设置主机写入 0 的DMA访问优先级。当<abt_en>位设置为 0 时，才能进行写入访问。</p> <p>优先级范围：00 (最高级) ~ 11 (最低级)。</p>
7-6	-	R	读取未定义
5-4	abtpri_r1	R	<p>当选择固定优先级模式时，设置主机读取 1 的DMA访问优先级。当<abt_en>位设置为 0 时，才能进行写入访问。</p> <p>优先级范围：00 (最高级) ~ 11 (最低级)。</p>
3-2	-	R	读取未定义
1-0	abtpri_r0	R/W	<p>当选择固定优先级模式时，设置主机读取 0 的DMA访问优先级。当<abt_en>位设置为 0 时，才能进行写入访问。</p> <p>优先级范围：00 (最高级) ~ 11 (最低级)。</p>

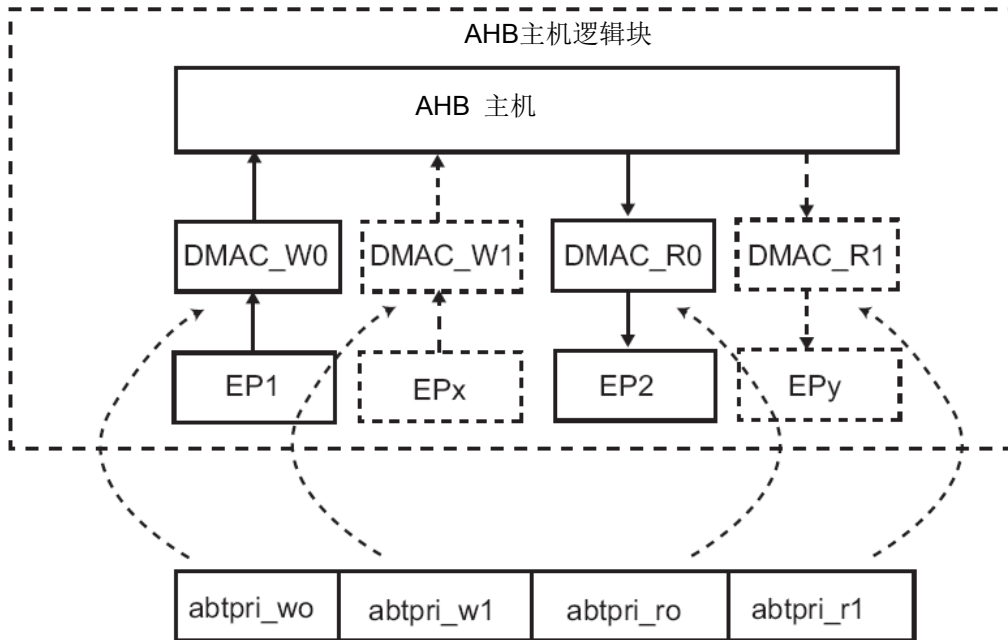
注：必须为<abtpri_w1>, <abtpri_w0>, <abtpri_r1>和<abtpri_r0>位设置不同的优先值。 如果设置的优先值相同，将无法设置 1 到<abt_en>。

(1) 判优器设置寄存器的DMAC与优先区的关系

当前UDC2AB规范支持一个主机写入DMAC(DMAC_W0)和一个主机读取DMAC(DMAC_R0)。不支持第二个主机写入DMAC(DMAC_W1)和第二个主机读取DMAC(DMAC_R1)。

事实上，为DMAC_W1和DMAC_R1设置优先级并没有任何意义，但却必须为上述abtpri_w1, abtpri_w0, abtpri_r1和abtpri_r0位设置不同的优先值。

可以为未封包的DMAC的相应寄存器区设置数值。判优器设置寄存器的优先区与以下DMAC相符：



判优器设置寄存器的优先级

图 13-8 DMAC与优先区的关系图

13.4.1.12 UDFSMWSADR (主机写入起始地址寄存器)

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

位	比特符号	类型	功能
31-0	mwsadr[31:0]	R/W	设置主机写入传输的起始地址。但由于此主机运行只支持地址增量，因此应设置小于UDFSMWEADR的值。

13.4.1.13 UDFSMWEADR(主机写入结束地址寄存器)

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

位	比特符号	类型	功能
31-0	mweadr[31:0]	R/W	设置主机写入传输的结束地址。但由于此主机只支持地址增量，因此应设置高于UDFSMWSADR的值。

13.4.1.14 UDFSMWCADR(主机写入当前地址寄存器)

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

位	比特符号	类型	功能
31-0	mwcadr[31:0]	R	显示主机写入传输过程中从EP到主机写入缓冲区的传输已经完成的地址。在传输过程中发生超时中断或出现错误时采用。 数据从EP设置到主机写入缓冲区时，此地址将递增，而在主机写入传输过程中，数据将存储到目标设备或主机写入缓冲区，直到显示地址为止。

13.4.1.15 UDFSMWAHBADR(主机写入AHB地址寄存器)

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

位	比特符号	类型	功能
31-0	mrsadr[31:0]	R	在主机写入传输过程中到目标设备的传输完成后，将显示地址。在传输过程中发生超时中断或出现错误时采用。 当数据设置到目标设备时，此地址将递增，而在主机写入传输过程中，数据将存储到目标设备，直到显示地址为止。

13.4.1.16 UDFSMRSADR(主机读取起始地址寄存器)

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

位	比特符号	类型	功能
31-0	mrsadr[31:0]]	R/W	设置主机读取传输的起始地址。但由于此主机只支持地址增量，因此应设置低于UDFSMWEADR的值。

13.4.1.17 UDFSMREADR(主机读取结束地址寄存器)

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

位	比特符号	类型	功能
31-0	mreadr[31:0]]	R/W	设置主机读取传输的结束地址。但由于此主机只支持地址增量，因此应设置高于UDFSMRSADR的值。

13.4.1.18 UDFSMRCADR(主机读取当前地址寄存器)

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

位	比特符号	类型	功能
31-0	mrcadr[31:0]	R	显示主机读取传输过程中从目标设备到EP的传输已经完成的地址。数据从主机读取缓冲区设置到EP时，此地址将递增，而在主机读取传输过程中，数据将存储到EP的FIFO中，直到显示地址为止。

13.4.1.19 UDFSMRAHBADR(主机读取AHB地址寄存器)

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

位	比特符号	类型	功能
31-0	mrahbadr[31:0]	R	显示主机读取传输过程中从目标设备到UDC2AB的传输已经完成的地址。从目标设备设置数据时，此地址将递增，而在主机读取传输过程中，数据将存储到缓冲区或EP的FIFO中，直到显示地址为止。

13.4.1.20 UDFSPWCTL (功率检测控制寄存器)

	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	wakeup_en	phy_re mote_wkup	phy_resetb	suspend_x	phy_susp end	pw_detect	pw_resetb	usb_reset
复位后	0	0	1	1	0	0	1	0

位	比特符号	类型	功能
31-8	-	R	读取 0。
7	wakeup_en	R/W	<p>在USB暂停时,如果打算将TMPM365FYXBG切换成低功耗模式来停止CLK_H,应在此位设置为"1"。</p> <p>如果此位设置为"1",当取消暂停状态 (<suspend_x>=1) 时, $\overline{\text{WAKEUP}}$ 信号将同步启用为 0。用INTUSBWKUP 可以使 TMPM365FYXBG由低功耗模式恢复到工作状态。有关如何使用此位的信息请参考"13.5.7 暂停/恢复"。</p> <p>0: 不启用 $\overline{\text{WAKEUP}}$ 信号。 1: 启用 $\overline{\text{WAKEUP}}$ 信号。</p>
6	phy_remo-to_wkup	R/W	<p>此位用于执行USB的远程唤醒功能。将此位设置为 1 可启用udc2_wakeup输出信号(UDC2唤醒输入引脚)为 1。但由于在UDC2没有检测到暂停状态(suspend_x = 1)时忽略了将此位设置为 1(未设置为 1),因此必须保证只有在检测到暂停状态时才进行设置。</p> <p>当USB恢复完成时(suspend_x被解除认定)将自动清0。有关如何使用此位的其他信息请参考"1.3.24.8 暂停/恢复"章节。</p> <p>有关如何使用此位的信息请参考"13.5.7 暂停/恢复"。</p> <p>0: 无运行 1: 唤醒</p>
5	phy_resetb	R/W	<p>将此位设置为 0 将使PHYRESET输出信号启用为 1。PHYRESET信号可以复位PHY。由于此位不会自动释放,因此必须在规定的PHY复位时间之后将其清除。</p> <p>0: 复位启用 1: 复位解除启用</p>
4	suspend_x	R	<p>检测暂停信号(与UDC2的suspend_x信号值的检测同步)。</p> <p>0: 暂停(<suspend_x> = 0) 1: 恢复(<suspend_x> = 1)</p>
3	phy_suspen-d	R/W	<p>将此位设置为 1 将使 $\overline{\text{PHYSUSPEND}}$ 输出信号启用为0(CLK_H同步)。它可用作一个引脚来暂停PHY。</p> <p>此位设置为1后不能访问UDC2寄存器和UDFSDMACRDREQ。</p> <p>恢复后(UDC2的suspend_x解除启用状态时)将自动清0。</p> <p>有关如何使用此位的信息请参考"13.5.7 暂停/恢复"。</p> <p>0: 恢复 1: 暂停</p>
2	pw_detect	R	<p>表示UDC2AB的VBUSPOWER输入状态。</p> <p>0: USB总线断开(VBUSPOWER = 0) 1: USB总线连接(VBUSPOWER = 1)</p>
1	pw_resetb	R/W	<p>UDC2AB软件复位(详见"13.5.1 复位"章节)。此位设置为0 后将使PW_RESETB输出引脚信号启用为0。</p> <p>主机运行停止时应进行复位。</p> <p>由于此位为不会自动释放,必须将其清除。</p> <p>0: 复位启用 1: 复位解除启用</p>
0	usb_reset	R	<p>与UDC2的usb_reset信号值同步。</p> <p>0: usb_reset = 0 1: usb_reset = 1</p>

13.4.1.21 UDFSMSTSTS(主机状态寄存器)

	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	-	-	-	mrepempty	mrbfemp	mwbfemp	mrepdset	mwepdset
复位后	0	0	0	1	1	1	0	0

位	比特符号	类型	功能
31-5	-	R	复位.未定义
4	mrepempty	R	此寄存器用于表示UDC2Rx的EP没有数据。 当用UDFSC2STSET<tx0>发送NULL封包时，保证此位设置为 1。(此位是eptx_empty输入信号，与CLK_H同步) 0: 表示EP有一部分数据。 1: 表示EP没有任何数据。
3	mrbfemp	R	表示UDC2AB中的主机读取DMA的缓冲区是否有数据。 0: 表示主机读取DMA的缓冲区有一部分数据。 1: 表示主机读取DMA的缓冲区没有任何数据。
2	mwbfemp	R	表示UDC2AB中的主机写入DMA的缓冲区是否有数据。 0: 表示主机写入DMA的缓冲区有一部分数据。 1: 表示主机写入DMA的缓冲区没有任何数据。
1	mrepdset	R	当要被传输的数据通过主机读取DMA传输设置为UDC2的Tx-EP时，此位将被设置为 1，导致EP中没有空间再写入数据。当UDC2通过来自主机的IN-Token传输数据时，它将转为 0。而当此位设置为0时，DMA可以传输给EP。(此位是eptx_dataset输入信号，与CLK_H同步) 0: 数据可以传输给EP。 1: EP中没有空间传输数据。
0	mwepdset	R	当接收到的数据设置为UDC2的Rx-EP时，此位将被设置为 1。当主机写入的DMA读取全部数据时，它将转为0。(此位是eprx_dataset输入信号，与CLK_H同步) 0: EP中没有数据。 1: EP中有一部分数据，可以读取。

13.4.1.22 UDFSTOUTCNT(超时计数寄存器)

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

位	比特符号	类型	功能
31-0	tmoutcnt[31:0]	R	此寄存器用于调试。启用UDFSMWTOU<timeout_en>时可以读取定时器的值。主机写入的EP(Rx-EP)变成空态后，每当CLK_U计数时将减量。直接指定地址不能读取此寄存器。若要读取此寄存器，应设置一个值到UDFSDMACRDREQ，然后从UDFSDMACRDVL读取该值。

13.4.2 UDC2 寄存器

13.4.2.1 UDC2 寄存器

基址 =0x4000_8000

寄存器名称	地址(基+)
UDC地址状态寄存器	UDFS2ADR 0x0200
UDC2 帧寄存器	UDFS2FRM 0x0204
保留	- 0x0208
UDC2 命令寄存器	UDFS2CMD 0x020C
UDC2 bRequest-bmRequest型寄存器	UDFS2BRQ 0x0210
UDC2 wValue寄存器	UDFS2WVL 0x0214
UDC2 wIndex寄存器	UDFS2WIDX 0x0218
UDC2 wLength寄存器	UDFS2WLGTH 0x021C
UDC2 INT寄存器	UDFS2INT 0x0220
UDC2 INT EP寄存器	UDFS2INTEP 0x0224
UDC2 INT EP屏蔽寄存器	UDFS2INTEPMSK 0x0228
UDC2 INT RX DATA0寄存器	UDFS2INTRX0 0x022C
UDC2 EP0 MaxPacketSize寄存器	UDFS2EP0MSZ 0x0230
UDC2 EP0 状态寄存器	UDFS2EP0STS 0x0234
UDC2 EP0 Datasize寄存器	UDFS2EP0DSZ 0x0238
UDC2 EP0 FIFO寄存器	UDFS2EP0FIFO 0x023C
UDC2 EP1 MaxPacketSize寄存器	UDFS2EP1MSZ 0x0240
UDC2 EP1 状态寄存器	UDFS2EP1STS 0x0244
UDC2 EP1 Datasize寄存器	UDFS2EP1DSZ 0x0248

UDC2 EP1 FIFO寄存器	UDFS2EP1FIFO	0x024C
------------------	--------------	--------

基址 =0x4000_8000

寄存器名称		地址(基+)
UDC2 EP2 MaxPacketSize寄存器	UDFS2EP2MSZ	0x0250
UDC2 EP2状态寄存器	UDFS2EP2STS	0x0254
UDC2 EP2 Datasize寄存器	UDFS2EP2DSZ	0x0258
UDC2 EP2 FIFO寄存器	UDFS2EP2FIFO	0x025C
UDC2 EP3 MaxPacketSize寄存器	UDFS2EP3MSZ	0x0260
UDC2 EP3 状态寄存器	UDFS2EP3STS	0x0264
UDC2 EP3 Datasize寄存器	UDFS2EP3DSZ	0x0268
UDC2 EP3 FIFO寄存器	UDFS2EP3FIFO	0x026C
UDC2 EP4 MaxPacketSize寄存器	UDFS2EP4MSZ	0x0270
UDC2 EP4 状态寄存器	UDFS2EP4STS	0x0274
UDC2 EP4 Datasize寄存器	UDFS2EP4DSZ	0x0278
UDC2 EP4 FIFO寄存器	UDFS2EP4FIFO	0x027C
UDC2 EP5 MaxPacketSize寄存器	UDFS2EP5MSZ	0x0280
UDC2 EP5 状态寄存器	UDFS2EP5STS	0x0284
UDC2 EP5 Datasize寄存器	UDFS2EP5DSZ	0x0288
UDC2 EP5 FIFO寄存器	UDFS2EP5FIFO	0x028C
UDC2 EP6 MaxPacketSize寄存器	UDFS2EP6MSZ	0x0290
UDC2 EP6 状态寄存器	UDFS2EP6STS	0x0294
UDC2 EP6 Datasize寄存器	UDFS2EP6DSZ	0x0298
UDC2 EP6 FIFO寄存器	UDFS2EP6FIFO	0x029C
UDC2 EP7 MaxPacketSize寄存器	UDFS2EP7MSZ	0x02A0
UDC2 EP7 状态寄存器	UDFS2EP7STS	0x02A4
UDC2 EP7 Datasize寄存器	UDFS2EP7DSZ	0x02A8
UDC2 EP7 FIFO寄存器	UDFS2EP7FIFO	0x02AC
保留	-	0x02B0 ~ 0x32C
UDC2 INT NAK寄存器	UDFS2INTNAK	0x0330
UDC2 INT NAK屏蔽寄存器	UDFS2INTNAKMSK	0x0334
保留	-	0x0338 ~ 0x03FC

注：上表中的“保留”栏和0x0400 ~ 0x0FFF的区域禁止访问。 这些区域禁止读取/写入。

13.4.2.2 如何访问UDC2寄存器

UDC2AB的AHB数据总线的 15-0 位连接到UDC数据总线。

位 31-16 只能读取(读取值: 未定义)。

可以通过一个WORD(32-位)来进行写入和读取访问。(但也可以通过一个BYTE(8-位)来写入访问EPx_FIFO寄存器。详见后续内容。)

完成写入和读取访问需花费一段时间。

在前一个UDC2 寄存器访问结束后必须用int_udc2_reg_rd中断开始后续访问,(读取时也可以采用UDFSUDC2RDREQ<udc2rdreq>来确认访问状态。)

- 写入访问

对UDC2寄存器进行写入访问时,直接写入相关地址。

- 读取访问

采用UDFSUDC2RDREQ和UDFSUDC2RDVLA对UDC2寄存器进行读取访问。

首先设置地址来访问UDFSUDC2RDREQ,然后从读取用的UDFSUDC2RDVLA读取数据。不能直接从"13.4.2.1 UDC2寄存器"中列出的地址读取数据。

- EPx_FIFO寄存器

对EPx_FIFO寄存器进行写入访问时,UDC2 PPCI I/F中需要进行低 1 字节访问。在这种情况下,可以对UDC2AB的低 1 字节进行BYTE访问。

进行读取访问时如果需要低 1 字节访问,应照例通过UDFSUDC2RDREQ进行访问,并从UDFSUDC2RDVLA读取数据。在这种情况下,可以通过WORD或BYTE来访问UDFSUDC2RDVLA。

- UDC2 中的保留寄存器。

不要访问将要连接的UDC2 不支持的EP寄存器和"保留"寄存器。(如果访问了这些寄存器,将自动从UDC2AB访问UDC2。写入访问时,将对UDC2进行虚写操作。读取访问时,UDC2(udc2_rdata)的读取数据是不确定值,并且这个值将被设置到UDFSUDC2RDVLA。)

- UDC2 暂停时的访问

UDC2 处于暂停状态时,如果来自时钟/模式控制电路的时钟(=CLK_U)源停止,将不能对UDC2 进行寄存器访问。在这种情况下,不要对UDC2 进行寄存器访问。如果在UDFSPWCTL<phy_suspend>设置为 1 时访问UDC2 寄存器,将返回一个AHB错误。

UDC2 寄存器的访问流程图如下所示。

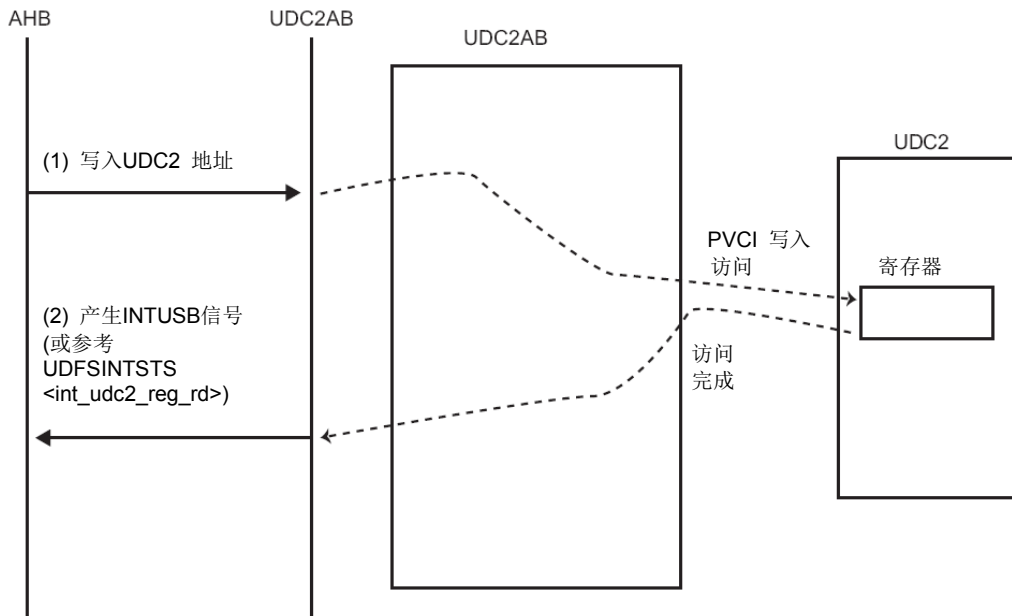


图 13-9 UDC2 寄存器的写入访问流程图

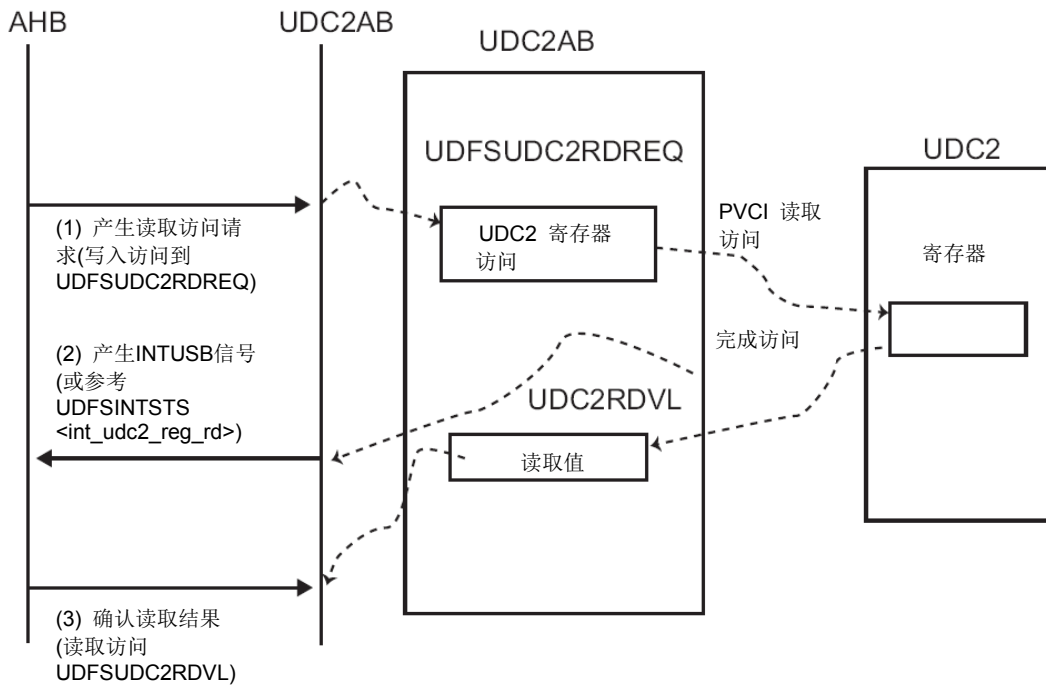


图 13-10 UDC2 寄存器的读取访问流程图

13.4.2.3 UDFS2ADR(地址状态寄存器)

	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	阶段_err	ep_bi_mode	cur_speed		suspend	configured	addressed	default
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	-	dev_adr						
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-16	-	R	读取未定义
15	阶段_err	R/W	表示到STATUS-阶段控制传输是否正常完成。 当在DATA-阶段/STATUS-阶段接收到Setup-Token或在"STALL"传输时将设置 1。 设置后，在下一个控制传输正常完成时将被清除。 0: 上述条件以外的其他条件。 1: 在DATA-阶段/STATUS-阶段或"STALL"传输时接收Setup-Token。
14	ep_bi_mode	R/W	选择是否双向采用EP作为驱动器。将此位设置为 1 后可获得一个EP号，此EP号将在USB通信中双向使用。 0: 单向 1: 双向
13-12	cur_speed[1:0]	R	表示USB总线上的当前传输模式 00: 保留 01: 全速 10: 保留 11: 保留
11	suspend	R	表示UDC2 是否处于暂停状态 0: 正常 1: 暂停
10	configured	R/W	设置UDC2 的当前设备状态。应按主机发送的请求进行设置。请注意，多位不能设置 1。 001: 默认(在默认/地址状态下当Set_address请求指定DeviceAddress = 0时进行设置(接收到USB_RESET时可通过硬件进行设置)) 010: 编址(当Set_address请求正常完成后，Set_configuration请求指定ConfigurationValue = 0时且处于地址/配置状态时进行设置) 100: 配置(接收到Set_configuration请求时进行设置)
9	addressed	R/W	
8	default	R/W	
7	-	R	读取未定义
6-0	dev_adr[6:0]	R/W	设置主机分配的设备地址。设备地址应在Set_address正常完成后(STATUS-阶段正常完成后)进行设置。

13.4.2.4 UDFS2FRM (帧寄存器)

	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	create_sof	-	f_status		-	frame		
复位后	0	0	1	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	frame							
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-16	-	R	读取未定义
15	create_sof	R/W	当主机的SOF因总线错误不可用时，设置是否内部产生SOF标志。如果希望在等时传输过程中通过SOP实现帧同步，应进行设置。如果启用，即使不能成功接收SOF-Token，内部帧计时器也会开始运行，并且将输出SOF标志。 0：不产生标志 1：产生标志
14	-	R	读取未定义
13-12	f_status[1:0]	R	表示帧数状态。 00：之前：如果<create_sof>启用时接收到Micro SOF/SOF之后已过 1 帧时间(全速：1 ms)时未接收到Micro SOF/SOF，将进行设置。在帧寄存器中，已设置最后一个Micro SOF/SOF中接收到的帧数。 01：有效：接收到Micro SOF/SOF时进行设置。表示帧寄存器中设置了一个有效帧数。 10：丢失：表示主机保存的帧数不与UDFS2FRM值同步。出现下列情况时也应做相应设置： 1. 当系统复位或暂停。 2. 如果<create_sof>启用时接收到前一个Micro SOF/SOF之后已过 2 帧时间(全速：1x2 ms)时未接收到下一个Micro SOF/SOF。 请注意，系统复位后或如果<create_sof>启用后暂停时将过渡到丢失状态。
11	-	R	读取未定义
10-0	frame[10:0]	R	表示接收到SOF时的帧数。 当<f_status>是"有效"状态时才有效。 由于未设置正确值，如果<f_status>是"之前"或"丢失"状态，不应采用。

13.4.2.5 UDFS2CMD (命令寄存器)

	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	int_toggle	-	-	-	rx_nulpkt_ep			
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	ep				com			
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-16	-	R	读取未定义
15	int_toggle	R/W	在Interrupt-IN传输过程中未接收到握手信号时进行DATA-PID触发。 0: 未接收时不要触发。 1: 未接收时触发
14-12	-	R	读取未定义
11-8	rx_nulpkt_ep [3:0]	R	表示接收到零-长度数据时的接收到的EP。 当INT_RX_ZERO标志被启用时, 读取此位来检查哪个EP被启用。一旦接收到零-长度数据且保留了EP号, 此寄存器的值将被保留, 直到下次接收到零-长度数据或硬件复位为止。如果OUT方向至少有一个EP, 每当接收到零-长度数据时此位将更新。在这种情况下, 可以采用UDFS2INTRX0来识别哪个EP已经接收数据。
7-4	ep[3:0]	R/W	当将要发出的命令有效时设置EP。(不要指定不存在的EP。)
3-0	com[3:0]	R/W	设置将向ep[3:0]中选择的EP发送的命令。其他信息请参考"13.2.2.3 发送给EP的命令"。 0x0: 保留 0x1: Setup_Fin 0x2: Set_DATA0 0x3: EP_Reset 0x4: EP_Stall 0x5: EP_Invalid 0x6: 保留 0x7: EP_Disable 0x8: EP_Enable 0x9: All_EP_Invalid 0xA: USB_Ready 0xB: Setup_Received 0xC: EP_EOP 0xD: EP_FIFO_Clear 0xE: EP_TX_0DATA 0xF: 保留

13.4.2.6 UDFS2BRQ(bRequest-bmRequest型寄存器)

	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	request							
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	dir	req_type		recipient				
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-16	-	R	读取未定义
15-8	request[7:0]	R	表示随Setup-Token(bRequest字段)接收到的第二字节数据。
7	dir	R	表示随Setup-Token(bmRequestType字段)接收到的第一字节数据。 控制传输方向。 0: Control-WR传输 1: Control-RD传输
6-5	req_type[1:0]	R	请求类型 00: 标准请求 01: 等级请求 10: 厂商请求 11: 保留
4-0	recipient[4:0]	R	接收请求的方式 0_0000: 设备 0_0001: 接口 0_0010: EP 0_0011: 其它 0_0100-1_1111: 保留

13.4.2.7 UDFS2WVL(wValue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	value							
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	value							
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-16	-	R	读取未定义
15-8	value[15:8]	R	表示随Setup-Token(wValue (高)字段)接收到的第四字节数据。
7-0	value[7:0]	R	表示随Setup-Token(wValue (低)字段)接收到的第三字节数据。

13.4.2.8 UDFS2WIDX (wIndex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	index							
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	index							
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-16	-	R	读取未定义
15-8	index[15:8]	R	表示随Setup-Token(wIndex (高)字段)接收到的第六字节数据。
7-0	index[7:0]	R	表示随Setup-Token(wIndex (低)字段)接收到的第五字节数据。

13.4.2.9 UDFS2WLGTH(wLength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	length							
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	length							
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-16	-	R	读取未定义
15-8	length[15:8]	R	表示随Setup-Token(wLength(高)字段)接收到的第八字节数据。
7-0	length[7:0]	R	表示随Setup-Token(wLength(低)字段)接收到的第七字节数据。

13.4.2.10 UDFS2INT(INT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	m_nak	m_ep	m_ep0	m_sof	m_rx_data0	m_status	m_status_nak	m_setup
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	i_nak	i_ep	i_ep0	i_sof	i_rx_data0	i_status	i_status_nak	i_setup
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-16	-	R	读取未定义
15	m_nak	R/W	设置是否将<i_nak>输出到INT_NAK引脚。 0: 输出 1: 无输出
14	m_ep	R/W	设置是否将<i_ep>输出到INT_EP引脚。 0: 输出 1: 无输出
13	m_ep0	R/W	设置是否将<i_ep0>输出到INT_EP0引脚。 0: 输出 1: 无输出
12	m_sof	R/W	设置是否将<i_sof>输出到INT_SOF引脚。 0: 输出 1: 无输出
11	m_rx_data0	R/W	设置是否将<i_rx_data0>输出到INT_RX_ZERO引脚。 0: 输出 1: 无输出
10	m_status	R/W	设置是否将<i_status>输出到INT_STATUS引脚。 0: 输出 1: 无输出
9	m_status_nak	R/W	设置是否将<i_status_nak>输出到INT_STATUS_NAK引脚。 0: 输出 1: 无输出
8	m_setup	R/W	设置是否将<i_setup>输出到INT_SETUP引脚。 0: 输出 1: 无输出
7	i_nak	R/W	通过EP传输EP0除外NAK时将设置为 1。 (用UDFS2INTNAKMASK可以选择向其输出INT_NAK标志的EP)。写入 1 到此位可使UDFS2INTNAK的每个位清零。
6	i_ep	R/W	成功传输到EP时EP0除外将设置为 1。 (用UDFS2INTNAKMASK可以选择向其输出标志的EP)。写入 1 到此位可使UDFS2INTEP的每个位清零。
5	i_ep0	R/W	成功传输到EP0 时将设置为 1。
4	i_sof	R/W	当接收到SOF-token时或在create_sof模式下已过 1 帧时间后将设置为 1。
3	i_rx_data0	R/W	接收到零-长度数据时将设置为 1。(用UDFS2INTNAKMASK可以选择向其输出标志的EP)。写入 1 到此位可使UDFSINTTRX0的每个位清零。在Control-RD传输的STATUS-阶段接收到零-长度时不会设置为 1。

位	比特符号	类型	功能
2	i_status	R/W	在控制传输过程中通过EP0 成功完成STATUS-阶段时将设置为 1。(在STATUS-阶段接收到零-长度数据并且在Control-RD传输过程中成功完成时, 以及在STATUS-阶段传输零-长度数据并且在Control-WR传输过程中成功完成时, 将设置为 1。)
1	i_status_nak	R/W	Control-RD传输过程中通过EP0 接收到STATUS-阶段封包时将设置为 1。当设置此位(表示DATA-阶段已完成)时, 通过UDFS2CMD设置"Setup-Fin"命令, 使UDC2的阶段进行到STATUS-阶段。在Control-WR传输的DATA-阶段接收到其大小是MaxPacketSize(64 字节)整数倍的数据时, 可以接收零-长度数据, 表示DATA-阶段结束。然后, 由于在STATUS-阶段接收In-token时通过i_status_nak可以识别DATA- 阶段的结束, 因此可以使UDC2进行到STATUS-阶段。
0	i_setup	R/W	在控制传输过程中通过EP0 接收到Setup-Token时将设置为 1。

13.4.2.11 UDFS2INTEP(INT_EP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	i_ep7	i_ep6	i_ep5	i_ep4	i_ep3	i_ep2	i_ep1	-
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-16	-	R	读取未定义
15-8	Reserved	R/W	写入"0"。
7-1	i_ep7 ~ i_ep1	R/W	此标志显示EP(EP0 除外)的传输/接收状态。 成功传输到EP0 除外EP时相关位将设置为 1。(用UDFS2INTEPMSK可以选择向其输出int_ep标志的EP)。 0: 未传输/接收数据 1: 传输/接收部分数据
0	-	R/W	读取未定义

13.4.2.12 UDFS2INTEPMSK(INT_EP_MASK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	m_ep7	m_ep6	m_ep5	m_ep4	m_ep3	m_ep2	m_ep1	m_ep0
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-16	-	R	读取未定义
15-8	Reserved	R/W	写入"0"。
7-0	m_ep7 ~ m_ep0	R/W	标志输出屏蔽控制 0: 输出 1: 无输出 设置是否将UDFS2INTEP和UDFS2INTRX0的标志分别输出到int_ep引脚和int_rx_zero引脚。当EP被屏蔽时,在传输相关EP成功完成时将设置UDFS2INTEP的每个位,但int_ep引脚不会被启用。同样,当EP被屏蔽时,将在相关EP接收到零-长度数据时也将设置UDFS2INTRX0的每个位,但int_rx_zero引脚不会被启用。但 0 位只对UDFS2INTRX0 有效。

(1) 如何使用UDFS2INT/UDFS2INTEP/UDFS2INTEPMSK

为EP1 ~ 3提供UDFS2INT / UDFS2INTEP / UDFS2INTEPMSK的使用示例。

1. 当采用带DMA(EP I/F)的EP1 和EP2 并且通过PVC I-I/F只采用EP3 时

UDFS2INT	<i_ep>	用作EP3 的中断源。清除时也采用此位。
	<m_ep>	用来屏蔽EP3 的中断源。
UDFS2INTEP	<i_ep1>	忽略
	<i_ep2>	忽略
	<i_ep3>	忽略
UDFS2INTEPMSK	<m_ep1>	设置 1, 屏蔽此位
	<m_ep2>	设置 1, 屏蔽此位
	<m_ep3>	写入 0。

2. 当通过PVC I-I/F采用EP2 和EP3 并且采用带DMA的EP1 时

初始化过后,设置 1 到要被采用的EP(带DMA)的UDFS2INTEPMSK,进行屏蔽。对多个EP作出中断响应时,必须采用UDFS2INTEP。
忽略UDFS2INT<i_ep>,始终启用<m_ep>为 0。

不能用UDFS2INT<i_ep>清除中断源。中断发生后，须检查UDFS2IN和UDFS2INTEP，以确定此中断源。清除此中断源时，用UDFS2INT中断的每个源位将其清除。

UDFS2INT	<i_ep>	写入 0。
	<m_ep>	写入 0。
UDFS2INTEP	<i_ep1>	忽略
	<i_ep2>	用作EP2 的中断源。清除时也采用此位。
	<i_ep3>	用作EP3 的中断源。清除时也采用此位。
UDFS2IN-TEPMSK	<m_ep1>	设置 1，屏蔽此位
	<m_ep2>	用来屏蔽EP2 的中断源。写入"0"。
	<m_ep3>	用来屏蔽EP3 的中断源。写入"0"。

13.4.2.13 UDFS2INTRX0(INT_RX_DATA0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	rx_d0_ep7	rx_d0_ep6	rx_d0_ep5	rx_d0_ep4	rx_d0_ep3	rx_d0_ep2	rx_d0_ep1	rx_d0_ep0
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-16	-	R	读取未定义
15-8	Reserved	R/W	写入"0"。
7-0	rx_d0_ep7 ~ rx_d0_ep0	R/W	此标志表示EP已接收Zero-Length数据。 0: 未接收零-长度数据 1: 接收零-长度数据 当EP接收到零-长度数据时相关位将设置为 1。(用UDFS2INTEPMSK可以选择向其输出int_rx_zero标志的EP)。 对于位 0 (EP0), 在处理请求的同时DATA-阶段接收到零-长度数据时, 将设置为 1。由于STATUS-阶段接收到零-长度数据时不会进行设置, 应采用int_status标志。

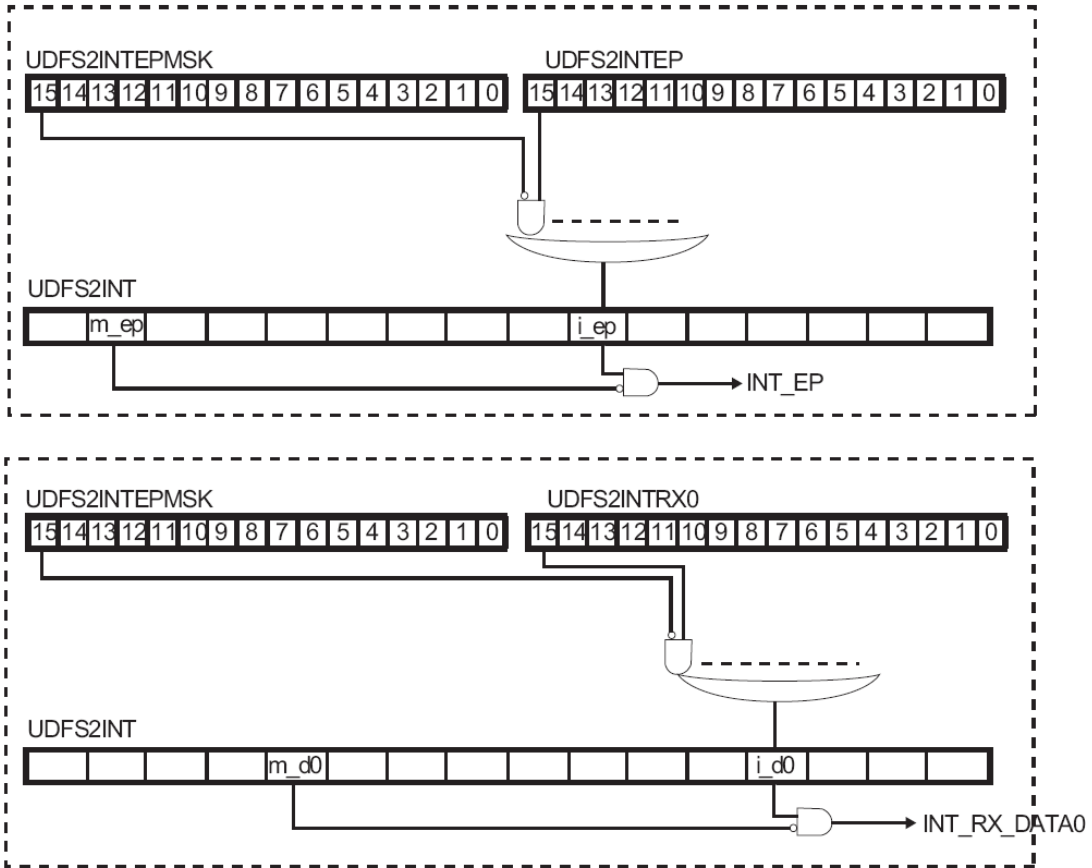


图 13-11 中断状态和屏蔽寄存器

13.4.2.14 UDFS2INTNAK(INT_NAK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	i_ep7	i_ep6	i_ep5	i_ep4	i_ep3	i_ep2	i_ep1	-
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-16	-	R	读取未定义
15-8	Reserved	R/W	写入"0"。
7-1	i_ep7 ~ i_ep1	R/W	此标志表示通过EP(EPO除外)传输NAK的状态。 0: NAK未传输 1: NAK已传输 当用EPO除外EP传输NAK时相关位将设置为 1。(用UDFS2INTEPMSK可以选择向其输出INT_NAK标志的EP)。
0	-	R	读取未定义

13.4.2.15 UDFS2INTNAKMSK(INT_NAK_MASK 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	m_ep7	m_ep6	m_ep5	m_ep4	m_ep3	m_ep2	m_ep1	-
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-16	-	R	读取未定义
15-8	保留	R/W	写入"0"。
7-1	m_ep7 ~ m_ep1	R/W	标志输出屏蔽控制 0: 输出 1: 无输出 设置是否将UDFS2INTNAK的标志分别输出到int_nak引脚。 EP被屏蔽时, 当相关EP的传输过程中传输NAK时, UDFS2INTNAK的每个位均将被设置, 但int_nak不会被启用。
0	-	R	读取未定义

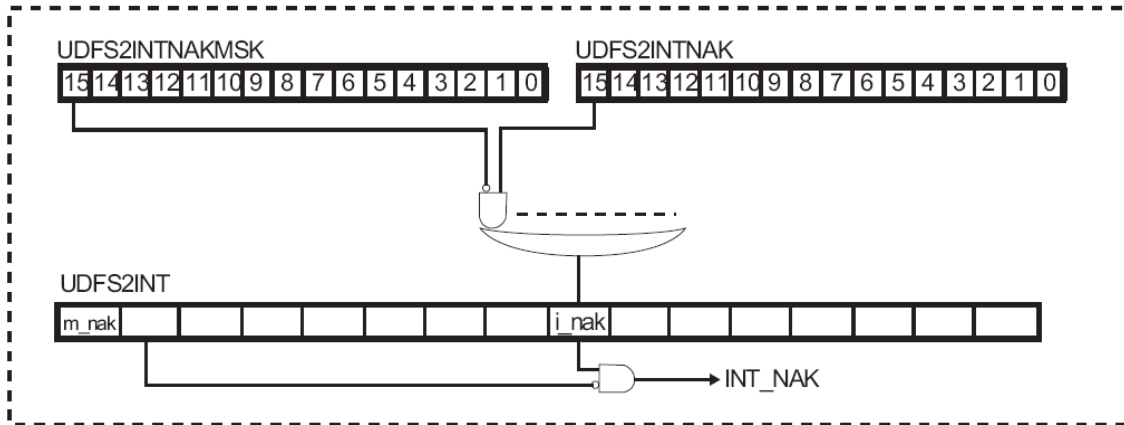


图 13-12 中断和状态寄存器

13.4.2.16 UDFS2EP0MSZ(EP0_MaxPacketSize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	tx_0data	-	-	dset	-	-	-	-
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	-	max_pkt						
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-16	-	R	读取未定义
15	tx_0data	R	通过UDFS2CMD将"EP_TX_0DATA"命令发送给EP0 时，此位将设置为 1，但在零-长度数据传输完成后又将清零。
14-13	-	R	读取未定义
12	dset	R	显示UDFS2EP0FIFO的状态。 接收到Setup-Token后将清0。 0: 无有效数据 1: 有有效数据
11-7	-	R	读取"0"。
6-0	max_pkt[6:0]	R/W	设置EP0的MaxPacketSize

13.4.2.17 UDFS2EP0STS(EP0_Status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	ep0_mask	-	toggle		status			-
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
31-16	-	R	读取未定义
15	ep0_mask	R	接收到Setup-Token后将设置为 1。发出"Setup_Received"命令时将清零。此位为 1 时没有数据写入UDFS2EP0FIFO。 0: 数据可以写入UDFS2EP0FIFO。 1: 数据不能写入UDFS2EP0FIFO。
14	-	R	读取未定义
13-12	toggle[1:0]	R	显示EP的当前触发值。 00: DATA0 01: DATA1 10: 保留 11: 保留
11-9	status[2:0]	R	显示EP0的当前状态。接收到Setup-Token时将被清除为"就绪(Ready)"状态。 000: 就绪(表示状态正常)。 001: 占空(当在STATUS-阶段返回"NAK"时进行设置) 010: 错误(当接收数据发生CRC错误时或传输数据后出现超时进行设置) 011: 暂停(当Control-RD传输过程中请求比Length更长的数据时将返回"STALL") 100 ~ 111: 保留
8-0	-	R	读取未定义

13.4.2.18 UDFS2EP0DSZ(EP0_Datasize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	-	size						
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-7	-	R	读取未定义
6-0	size[6:0]	R	显示UDFS2EP0FIFO中存储的有效数据字节数。 接收到Setup-Token时将被清除。

13.4.2.19 UDFS2EP0FIFO(EP0_FIFO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	data							
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	data							
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-16	-	R	读取未定义
15-0	data[15:0]	R/W	用于访问从PVCII-I/F到EP0 的数据。 有关访问此寄存器的方法请参考"13.7.1.1 Control-RD传输", "13.7.1.2 Control WR传输(无DATA-阶段)"和"13.7.1.3 Control-WR传输(有DATA-阶段)"。 当接收到请求时(当INT_SETUP中断被启用时), 存储在此寄存器中的数据将被清除。

13.4.2.20 UDFS2EPxMSZ(EPx_MaxPacketSize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	tx_0data	-	-	dset(注 1)	-	max_pkt		
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	max_pkt							
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-16	-	R	读取未定义
15	tx_0data	R	当通过UDFS2CMD将"EPx_TX_0DATA"命令发送给EPx或EP-I/F设置了Zero-Length数据时，此位将设置为 1，但在Zero-Length数据传输完成后将被清零。
14-13	-	R	读取未定义
12	dset	R	显示EPx_FIFO的状态。 0: 无有效数据 1: 有有效数据
11	-	R	读取未定义
10-0	max_pkt[10:0]	R/W	设置EPx的MaxPacketSize。 接收到Set_Configuration和Set_Interface时配置EP后进行设置。 设置传输EP的偶数。当USB上传输EP的MaxPacketSize是一个奇数时，设置一个偶数到max_pkt，并用奇数访问EP。(例如，当MaxPacketSize应为 1023 字节时，设置 1024 ~ max_pkt) 注：详见"13.9.2 附录B 如何设置字节的奇数作为MaxPacketSize"。

注 1: 当EPx是一个传输EP时并在重置为 1 后的<dset>的初始值，而当接收到EPx时又将清零。EP。

注 2: 当EPx是一个传输EP时USB_RESET设置为 1 后的<dset>的初始值，而当接收到EPx时仍将"保留"。

注 3: x=1 ~ 7

13.4.2.21 UDFS2EPxSTS(EPx_Status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	pkt_mode	bus_sel	toggle		status			禁用
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	dir	-	-	-	t_type		num_mf	
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-16	-	R	读取未定义
15	pkt_mode	R/W	选择EPx的封包模式。选择双模式可以为EPx保留两个封包数据。 0: 单模式 1: 双模式
14	bus_sel	R/W	选择总线, 访问EPx的FIFO。 0: 公用总线访问 1: 直接访问
13-12	toggle[1:0]	R	显示EPx的当前触发值。 00: DATA0 01: DATA1 10: DATA2 11: MDATA
11-9	status[2:0]	R	显示EPx的当前状态。 UDFSCMD发出EP_Reset后, 状态将为"就绪(Ready)". 000: 就绪(表示状态正常)。 001: 保留 010: 错误(当数据包发生接收错误或数据传输后出现超时进行设置)但当已设置"停止"或"无效"时不会再设置 "错误"状态。 011: 暂停(通过UDFS2CMD发出"EP-Stall"时将设置"暂停"状态) 100 ~ 110: 保留 111: 无效(表示此EP无效)。
8	禁用	R	显示是否允许Epx传输。如果"不允许", 对于发送给此EP的Token, 将始终返回"NAK". 0: 允许 1: 不允许
7	dir	R/W	为此EP设置传输方向。 0: OUT(主机到设备) 1: IN(设备到主机)
6-4	-	R	读取未定义
3-2	t_type[1:0]	R/W	为此EP设置传输模式。 00: 控制传输 01: 等时传输 10: 批量传输 11: 中断传输
1-0	num_mf[1:0]]	R/W	若选择等时传输模式, 在帧中设置传输次数。 00: 1-交易 01: 2-交易 10: 3-交易 11: 保留

注 1: 接收到Set_Configuration和Set_Interface后配置EP时应设置此寄存器。

注 2: $x=1 \sim 7$

注 3: 各个EP取决于产品规格。对于IN传输固定采用的EP1, EP3, EP5和EP7, <dir>只能设置为"1"。对于OUT传输固定采用的EP2, EP4和EP6, dir只能设置为"0"。

13.4.2.22 UDFS2EPxDSZ(EPx_Datasize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	-	-	-	-	-	size		
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	size							
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-11	-	R	读取未定义
10-0	size[10:0]	R	显示存储在EP1_FIFO中的有效数据字节数。在双封包模式下，将显示首先被访问的数据字节数。

注: x=1~7

13.4.2.23 UDFS2EPxFIFO (EPx_FIFO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	data							
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	data							
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-16	-	R	0.
15-0	data[15:0]	R/W	用于访问从PVCI-I/F到Epx的数据。

注: x=1~7

13.5 UDC2AB操作描述

13.5.1 复位

UDC2AB支持通过UDFSPWCTL<pw_resetb>进行软件复位。

也支持DMAC主传输的主通道复位，命令(UDFSMSTSET<mr_reset>/<mw_reset>)。

- 软件复位 (UDFSPWCTL<pw_resetb>)

每个寄存器的某些位由硬件复位初始化，而不是由保留了某个值的软件复位来初始化。各寄存器的详细说明，见"13.4.1.1 UDC2AB 寄存器列表"。

当检测到USB总线电源时，使用软件复位进行初始化是必要的。

- 主通道复位 (UDFSMSTSET<mr_reset><mw_reset>)

当<mw_reset>位被设置为主控写入传输块，并且<mr_reset>位为主控读取传输块时，只有相关的主块被初始化，UDC2AB寄存器不会被初始化。更多关于每种复位的信息，请参见"13.4.1.6 UDFSMSTSET(DMAC寄存器设置)"。

13.5.2 中断信号

UDC2AB 有两种中断输出信号，INTUSB和INTUSBWKUP。

13.5.2.1 INTUSB 中断信号

INTUSB中断输出信号由UDC2产生的中断信号和其他来源的中断信号组成。

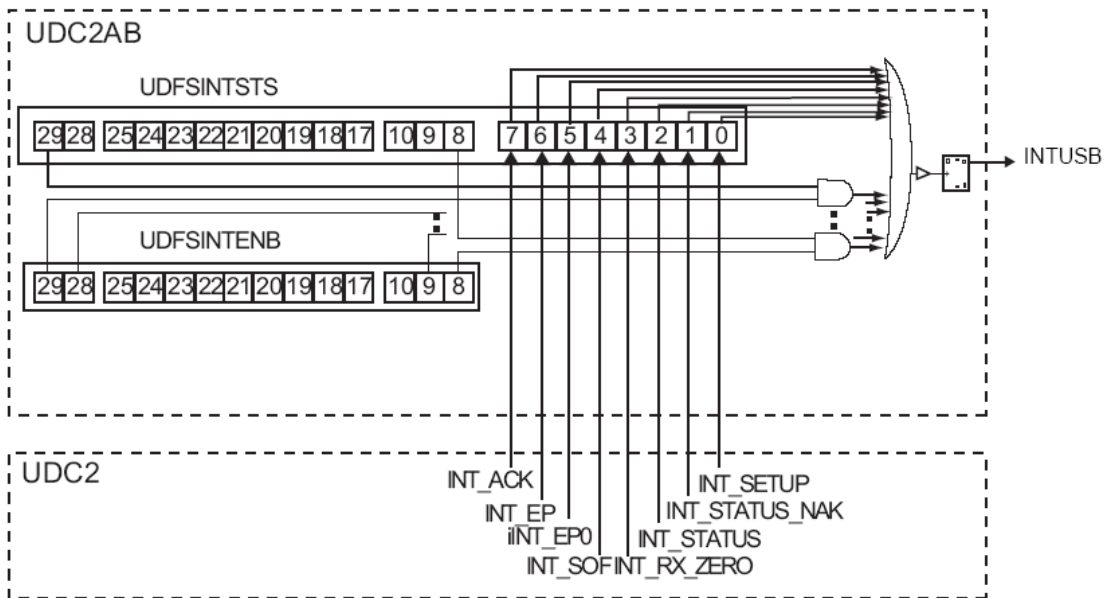
一旦中断条件满足，UDC2AB会设置相应的UDFSINTSTS位。当该位被设置后，如果相应的UDFSINTENB 位设置为"启用"，INTUSB将被启用。

当UDFSINTENB相关位设置为 "禁用" 时，UDFSINTSTS 相关位被设置为1，而 INTUSB 将不会被启用。

随着UDFSINTSTS的设置，UDFSINTENB的相关位被设置为 "启用"，INTUSB启用会在设置进行后立刻产生。

UDFSINTENB的初始值全部为0 (禁用)。

当CLK_H信号停止时INTUSB中断输出信号不会产生。



注：UDC2 标志由UDFS2INT信号屏蔽。

图 13-13 INTUSB信号和寄存器的关系

13.5.2.2 INTUSBWKUP中断

INTUSBWKUP中断发生于 $\overline{\text{WAKEUP}}$ 输出信号的下降缘。

当下列条件匹配时WAKEUP将被启用：UDFSPWCTL<wakeup_en>为1并取消暂停状态(UDFSPWCTL<suspend_x>=1)。当VBUS信号切断时(VBUSPOWER=0)，WAKEUP信号将被启用。

INTUSBWKUP中断信号的发生与CLK_H信号状态无关。

13.5.3 操作顺序

UDC2AB的操作顺序如下：

1. 硬件复位
2. 设置中断信号
配置INTUSB中断，INTUSBWKUP中断和USBPON中断。
3. VBUS (连接)检测和复位
详情见 "13.5.5.2 USB总线电源(VBUS) 连接/切断顺序"和"13.5.1复位"。
4. USB列举响应
详情请参见"13.6 USB装置响应"。
5. 主机读取 /主机写入信号传输
 - a. 主机读取信号传输
根据USB主机接收的请求信号进行主读取信号传输。详情请参见 "13.5.4.1 主机读取信号传输"。
 - b. 主机写入信号传输
根据USB主机的发送请求信号进行主机写入信号传输。详情请参见"13.5.4.2主机写入信号传输"。
6. VBUS (切断)检测
USB总线电源会在任何时间切断。
详情见"13.5.5.2 USB 总线电源 (VBUS)连接/切断"。

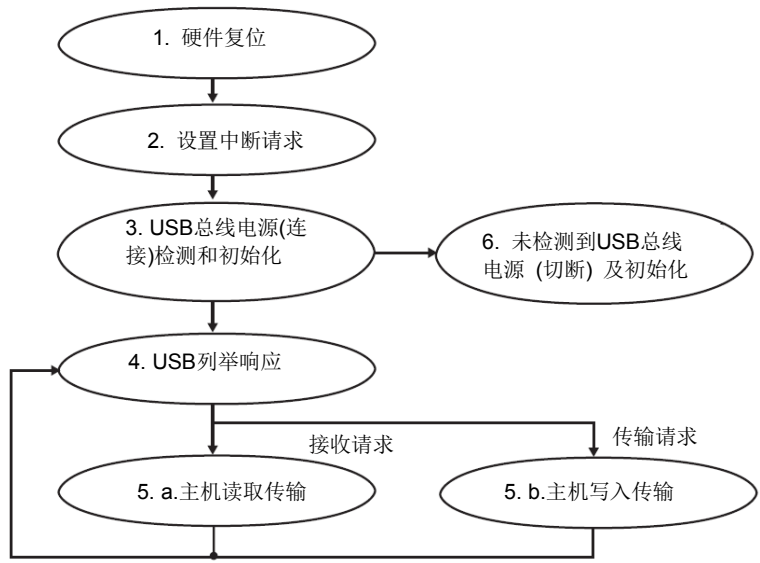


图 13-14 操作顺序

13.5.4 主机写入传输操作

本节描述了 UDC2AB的主机写入传输操作。

当启动一个主传输操作，请确认设置UDC2 (UDFS2EPxSTS<bus_sel>)的相关传输设置为直接访问方式。当DMAC设置为"普通总线访问"时，禁止启动 DMAC。

13.5.4.1 主机读取传输

(1) 主机读取方式

主机读取有两种模式。EOP启用模式和EOP禁用模式。

(a) EOP启用模式

在此描述当UDC2STSET<eopb_enable>设置为1时的主机读取信号传送（主机读取EOP启用）。主机读取信号传输操作如下：

1. 设置UDFSMWSADR信号和 UDFSMWEADR信号。
2. 设置UDFSMSTSET主机读取操作相关的各个位并将 <mr_enable>设置为1。
3. UDC2AB开始向UDC2的EP传输数据。UDC2从USB主机向IN token传输数据。
4. 当主机读取传输到达主机读取结束地址时，UDC2AB启用mr_end_add 中断。
5. 通过软件处理后，返回至1。

- 关于短包数据

当传输长度(主机读取结束地址-主机读取开始地址+1)与最大数据包长度不一致时，最后的IN传输将为短包传输。

示例：例如主机读取传输的长度是139字节而最大数据包长度是64字节。

传输发生于：

第1次	→	第2次	→	第3次
64字节		64字节		11字节

- 关于mr_end_add中断

当向UDC2 EP传输的数据完成后，发生mr_end_add中断。确认全部数据是否自UDC2传输至USB主机时，检查UDFSMSTSTS<mrepempty>。

(b) EOP禁用模式

在此描述为当 UDC2STSET<eopb_enable >设置为0 (主机读取EOP禁用) 的主机读取传输。主机读取信号传输操作如下：

1. 设置UDFSMWSADR和UDFSMWEADR相关的寄存器。
2. 设置UDFSMSTSET主机读取操作相关的各个位并将<mr_enable>设置为1。
3. UDC2AB开始向UDC2的EP传输数据。UDC2从USB主机向IN token传输数据。
4. 当主机读取传输到达主机读取结束地址时，UDC2AB启用mr_end_add 中断。主机读取传输操作时如果EP的FIFO达到最大数据包的长度时，UDC2 将从USB主机向IN token传输数据。但是，如果未达到最大数据包长度的数据将保存于FIFO内直至下一次传输。
5. 通过软件处理后，返回至1。

注：当UDC2AB使用EOP禁用模式时，即使将要发送的数据串已经开始传输了，短包数据也不会被发送。EOP 禁用模式仅用于数据串长度为最大包数据尺寸的整数倍情况时使用。

这种模式可以用于数据串长度为最大包数据长度的整数倍时使用。例如，下列传输是允许的：

示例：

第一次主机读取传输的数据长度	: 100字节
第二次主机读取传输的数据长度	: 28字节(第一第二次传输一共 = 128字节)
最大数据包长度	: 64字节

IN传输将进行两次，并且每次操作可以传输64字节。

(2) 主机读取传输操作的中止

可以进行下列操作以中止主机读取操作。

1. 使用UDC2命令寄存器来设置相关的EP为禁用状态 (EP_Disable)。(当不使用EP禁用来中止操作时，意外的数据可能被发送到USB主机。)
2. 要停止主机读取操作，应将UDFSMSTSET <mr_abort>设置为1(中止)。
3. 为了确认传输是否中止，可查验 UDFSMSTSET<mr_enable> 是否已经禁用为0。当mr_enable位为1时后续操作就不应该继续进行。
(中止操作时传输结束处的地址信息可以与主机读取当前地址和主机读取AHB地址寄存器确认。)
4. 为了初始化主机读取传输块，可将UDFSMSTSET <mr_reset> 设置为1 (复位)。
5. 使用命令寄存器(EP_FIFO_Clear)初始化FIFO相关EP。
6. 使用命令寄存器(EP_Enable)启用相关EP。

(3) 设置主机读取传输最大数据包长度

如果与UDC2AB主机读取功能相关的EP最大数据包长度是一个奇数，下列限制需要注意：

- 即使EP最大数据包长度需要作为一个奇数处理，UDFS2EPxMSZ<max_pkt>的设置也应该是一个偶数。

注：详情见 "13.9.2 附录 B 关于设置一个奇数字节长度作为MaxPacketSize"。

- 将UDC2STSET<eopb_enable> 设置为 1 (主机读取EOP启用)。
- 为一次主机读取传输(主机读取结束地址-主机读取起始地址+1)指定的数据长度不要超过奇数的最大数据包长度。

示例：

设置EP最大数据包长度 (这个值传输给USB主机)为 63 字节。

设置UDFS2EPxMSZ<max_pkt>为 64 字节。

保持为一次主机读取传输指定的数据传输长度为 63 字节或更少。

13.5.4.2 主机写入传输操作

(1) 主机写入传输操作顺序

主机写入传输操作应该如下所示：

1. 设置UDFSMWSADR信号和 UDFSMWEADR信号。
2. 设置UDFSMSTSET相关各个位并将<mw_enable>设置为1。
3. UDC2AB将从USB主机接收到的数据通过一次主机写入操作传输给EP。
4. 由于当到达主机写入结束地址时，主机写入操作结束(无超时处理)，mw_end_add 中断会启用，所以应通过软件进行必要的安排。接收正确的数据包后UDC2值应该回到1。

注：当数据包从禁用的UDFSMSTSET<mw_enable> USB主机正常被接收时，UDC2AB启用mw_set_add 中断。

(2) 超时

当来自于USB主机的OUT传输在传输期间到达主机写入结束地址之前停滞时，主机写入传输不会结束。为了应对这样的情况，可设置超时功能。

当设置超时功能时，存储在UDC2AB缓冲器里面的所有数据会传输给AHB。

超时可使用下列操作处理。

1. 在开始主机写入操作之前访问 UDFSMWTOUT 并进行超时设置(超时时间)，同时使<timeout_en>启用1。
2. 根据前面章节指南开始主机写入传输操作。

3. 当超时发生时，mw_timeout 中断将被启用。(mw_end_add 中断将不会被启用。)在这种情况下，主机写入操作不会完成并到达主机写入结束地址。UDFSMSTSET<mw_enable>清除为0。

4. UDFSMWCADR中的，代表传输给AHB结束完成的地址可确认。

请注意超时计数器在超时功能启用的主机写入传输操作期间会一直记数，但是当相关的EP收到USB主机发出的OUT操作信号时，计数器会复位至预设值并重新开始计数（见图13-15）。意思是指超时是"自当从USB主机向主机写入传输期间发生相关的EP的最后传输的时间点传输至预设时间"的时间，而非"自主机写入传输已经开始至预设的时间的时间点"。

如果不使用超时功能，应确保在开始主机写入操作之前将UDFSMWTOUT <timeout_en> 设置为"禁用 0"。在那种情况下，传输操作在到达预置的主机写入操作结束地址之前不会完成。

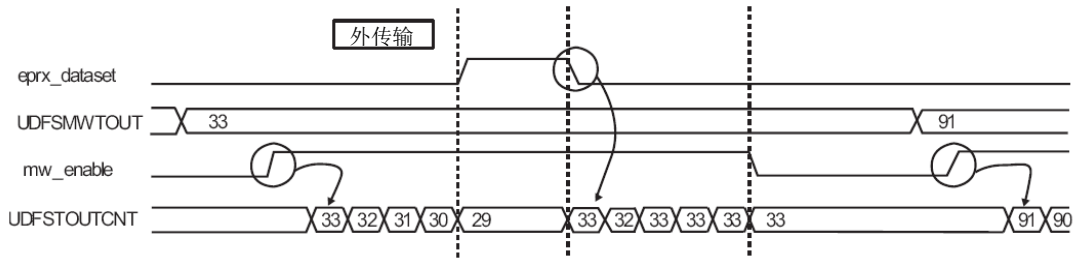


图 13-15 MW超时计数示例

(3) 主机写入操作中止

可使用下列操作中中止主机写入操作。

1. 使用UDFS2CMD将相关EP 状态设置为禁用 (EP_Disable)。
2. 为了停止主机写入操作，可将1(中止)设置为UDFSMSTSET<mw_abort>。
3. 为确保传输操作中中止，查验UDFSMSTSET<mw_enable> 是否已经禁用为0。当<mw_enable>为 1 时，不应该进行后续操作。(中止确认时传输结束处的地址信息包括主机写入当前地址和主机写入AHB地址寄存器。)
4. 为初始化主机写入传输块，应将1(复位)设置为UDFSMSTSET<mw_reset>。
5. 使用UDFS2CMD (EP_FIFO_Clear)初始化相关EP的FIFO。
6. 使用UDFS2CMD设置相关EP状态为启用 (EP_Enable)。

13.5.5 USB 电源管理控制

USB涉及电源管理的操作，除正常的数据包外，还包括USB总线电源检测，暂停和恢复也进行了阐述。本节讨论如何控制这些操作。

注：操作详情确保参见USB 2.0的技术规格书。

13.5.5.1 电源管理控制信号连接图

下图为关于电源管理控制的信号连接图。

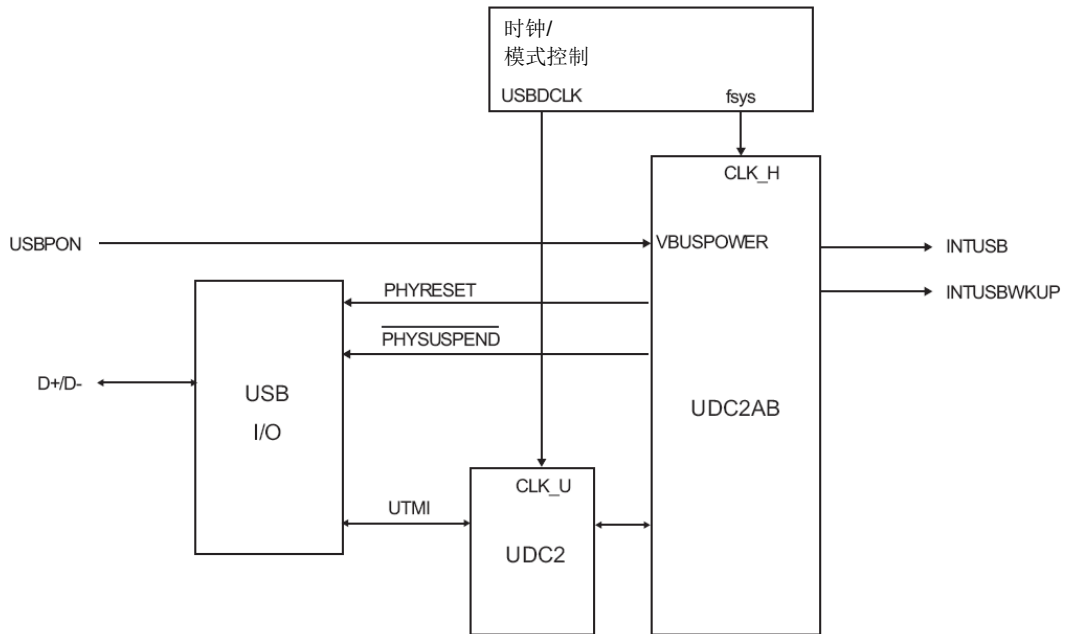


图 13-16 电源管理控制信号连接图

13.5.5.2 USB总线电源 (VBUS) 连接/切断顺序

(1) 连接

当CLK_H运行时，USB 总线电源 (VBUS) 的连接可使用 INTUSB (电源检测) 中断和UDFSPWCTL<pw_detect>检测到。当UCLK_H停止时，USB电源连接(VBUS)可使用INTUSBPON中断信号检测到。

检测到总线电源后(VBUS)，依据下列次序初始化UDC2AB和UDC2。

1. 使用 UDFSPWCTL <pw_resetb > 进行软件复位。(<pw_resetb> 位不会自动释放，应该由软件清位。)。
2. 访问UDC2AB和UDC2寄存器进行必要的初始设置。
3. 使用UDFS2CMD发布USB准备命令。UDC2 通过PHY通知USB主机连接信息。这种条件下允许 UDC2 接收来自USB主机的USB_RESET信号。
4. 一旦检测到来自USB主机USB_RESET信号，UDC2 会初始化UDC2 内部寄存器，与USB主机的接口可使用。当检测到USB_RESET信号时，usb_reset / usb_reset_end中断发生。

(2) 电源切断

如果 CLK_H 运行，USB 总线电源 (VBUS) 的切断可通过 INTUSB (电源检测)中断和UDFSPWCTL <pw_detect> 信号监测到。当CLK_H停止时，USB 电源 (VBUS)的切断可通过 INTUSBWKUP中断检测到。ÅB

当检测到USB总线电源(VBUS)切断时，每个主传输操作将不会自动停止。然后使用功率检测控制寄存器的pw_resetb位进行软件复位。

13.5.6 USB复位

USB_RESET信号不仅仅是当USB主机连接上时也可能是在任何时间被接收到。

当UDC2 接收到USB_RESET信号时，UDC2AB启用usb_reset / usb_reset_end 中断并返回默认状态。此时主传输操作不会自动停止。使用中止功能结束传输操作。使用USB_RESET信号初始化UDC2的某些寄存器的值，这些值是为其他寄存器保存的 (参见UDC2相关章节)。

当USB_RESET信号被识别时，UDC2寄存器的复位应该在usb_reset_end中断发生后进行。这是因为当UDC2 解除认定usb_reset信号时初始化了UDC2 寄存器。

13.5.7 暂停/恢复

13.5.7.1 切换到暂停状态

UDC2AB通过 INTUSB (suspend_resume) 中断和UDFSPWCTL <suspend_x> 信号通知UDC2暂停状态的监测。

既然在这种情况下主传输操作不会自动停止，如果需要，应该使用中止功能进行强制中止。

在使用软件中止了必要的处理后且PHY需要暂停的情况下（时钟停止），可设置UDFSPWCTL <phy_suspend> 使得 UDC2AB 启用 PHYSUSPEND，这样可将PHY置于暂停状态。

13.5.7.2 从暂停状态恢复(从USB主机恢复)

从暂停状态恢复的程序是基于CLK_H的条件执行的。
当恢复状态识别出来时，对重启主传输再次进行设置。

1. 停止CLK_H。

停止 CLK_H的程序和信号变化如下所示。

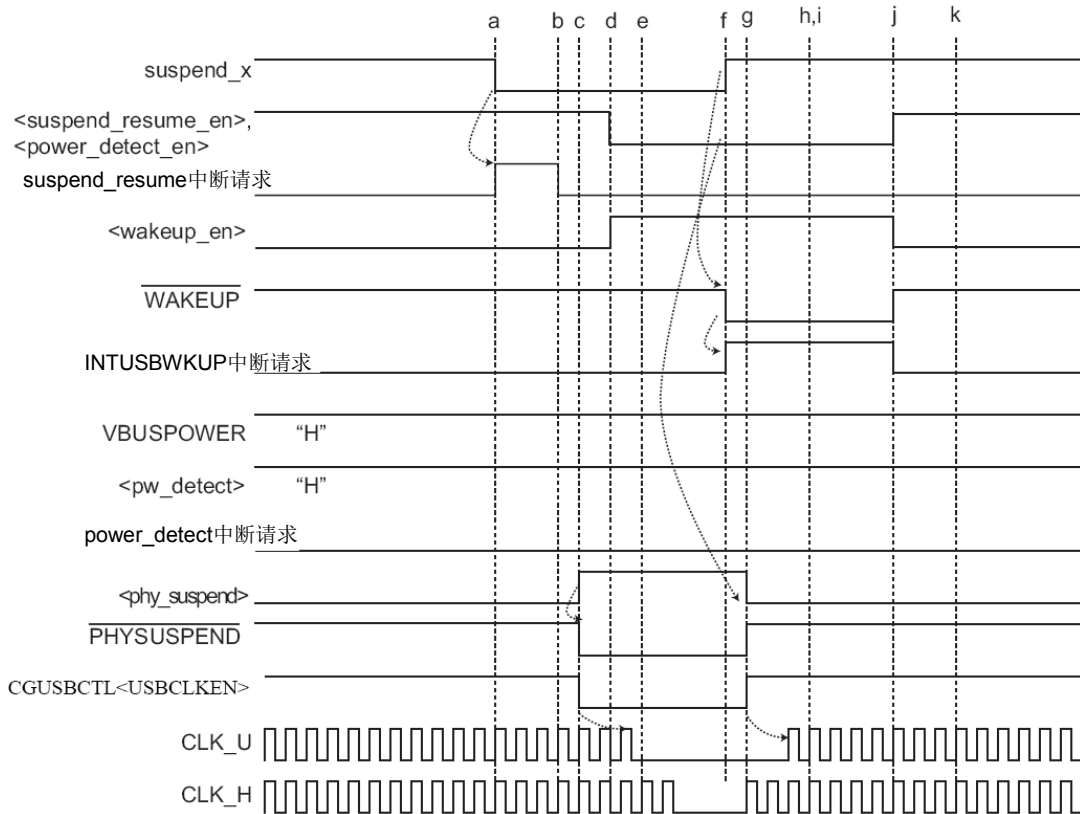


图13-17 暂停和恢复状态的信号操作 (当CLK_H 停止时)

- a. 通过检测到USB总线上的暂停状态，UDC2的suspend_x启用为0，且INTUSB(suspend_resume)有中断发生。
- b. INTUSB(suspend_resume)中断服务程序会清除中断因素。
- c. 设置UDFSPWCTL<phy_suspend>为 "1"。设置<phy_suspend>为"1"，启用 $\overline{\text{PHYSUSPEND}}$ 输出信号为"0"。
零会清除时钟/模式控制电路的 CGUSBCTL<USBCLKEN> 的 CGUSBCTL<USBCLKEN>，以停止 CLK_U。
- d. 设置UDFSPWCTL<wakeup_en>为 "1"。UDFSINTENB <power_detect_en> <suspend_resume_en> 清零不会产生INTUSB(power_detect, suspend_resume)中断。
- e. 发生INTUSBWKUP中断时，操作模式进入低-功耗模式并停止CLK_H。

- f. 通过USB总线上检测到恢复时， $\overline{\text{WAKEUP}}$ 输出信号将被异步启用为0。这个 $\overline{\text{WAKEUP}}$ 输出信号会使得INTUSBWKUP中断发生并取消低功耗模式。然后CLK_H 开始得电。
- g. CLK_H信号得电， $\overline{\text{PHYSUSPEND}}$ 输出信号自动启用为"1"，且 <phy_suspend> 清零。
设置时钟/模式控制电路的 CGUSBCTL<USBCLKEN> 为"1"以触发 CLK_U。
- h. 中断被启用 2.5 s后(当VBUS切断后需要时间稳定信号)检查 UDFSPWCTL<pw_detect>。如果 UDFSPWCTL<pw_detect>信号为 "1"， $\overline{\text{WAKEUP}}$ 被启用为恢复状态。如果UDFSPWCTL<pw_detect> 信号为 "0"， $\overline{\text{WAKEUP}}$ 启用为VBUS切断。
- i. 恢复时，请执行下列程序。切断时，请执行"13.5.7.3 从暂停状态(切断)恢复"中程序。
- j. 清除中断因素，<wakeup_en>会解除认定 $\overline{\text{WAKEUP}}$ 输出信号。设置 <suspend_resume_en> 信号为"1"。
- k. 从暂停状态恢复。

2. CLK_H启动

使CLK_H启动的程序以及信号变化如下所示。

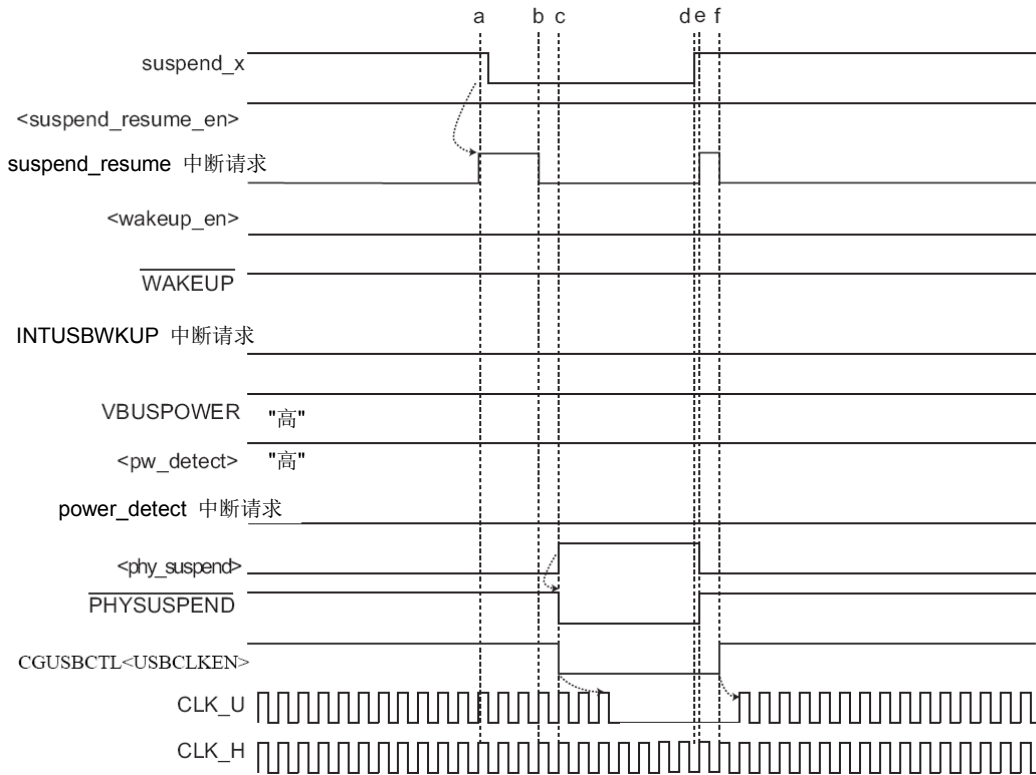


图 13-18 暂停/恢复信号操作 (启动CLK_H)

- a. 检测到USB总线的暂停状态时，INTUSB(suspend_resume)中断会发生。
- b. 清除INTUSB(suspend_resume)中断服务程序中的中断源。
- c. 将UDFSPWCTL<phy_suspend>信号设置为"1"。设置<phy_suspend>为"1"启用 $\overline{\text{PHYSUSPEND}}$ 输出信号为"0"。
设置时钟/模式电路的CGUSBCTL<USBCLKEN>为"0" 以停止 CLK_U。
- d. 检测到USB 总线上的恢复信号后，suspend_x 变为"1"。
另外，在检测到suspend_x的上升缘信号后， $\overline{\text{PHYSUSPEND}}$ 输出信号被解除认定为"1"。
- e. INTUSB(suspend_resume)中断发生。
- f. 在INTUSB(suspend_resume)服务程序中，中断源被清零。
设置时钟/模式电路的CGUSBCTL<USBCLKEN>为"1" 以启用 CLK_U。
- g. 解除认定 $\overline{\text{PHYSUSPEND}}$ 输出信号会恢复 CLK_U的供电。
- h. 从暂停状态恢复。

13.5.7.3 从暂停状态 (切断)恢复

从暂停状态(切断)恢复的程序以及信号变化如下所示。

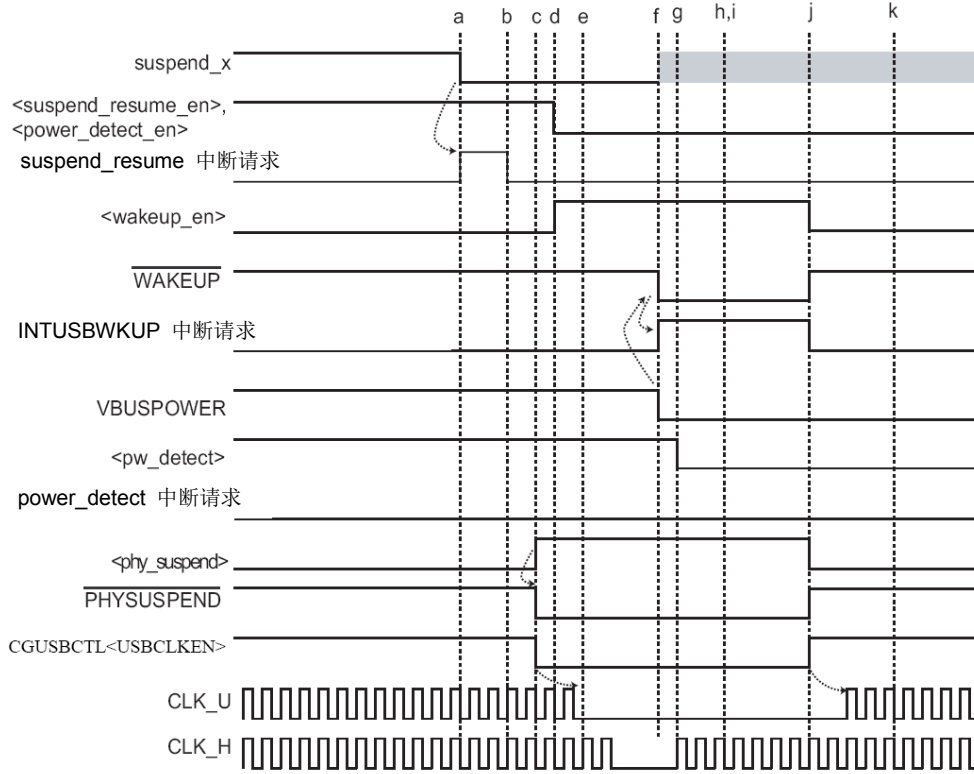


图 13-19 暂停/切断信号操作(停止CLK_H)

- a. 监视到USB总线的暂停状态，UDC2的suspend_x被启用为0，并产生INTUSB(suspend_resume)中断。
- b. 中断源被INTUSB(suspend_resume) 中断的服务程序清除。
- c. 设置UDFSPWCTL<phy_suspend>为 "1"。设置<phy_suspend>为"1"，启用 PHYSUSPEND 输出信号为"0"。
设置时钟/模式电路的CGUSBCTL<USBCLKEN>为"0" 以停止CLK_U。
- d. 设置UDFSPWCTL<wakeup_en>为 "1"。对UDFSINTENB<power_detect_en><suspend_resume_en>清零不会产生 INTUSBD (power_detect, suspend_resumu) 中断。
- e. 伴随INTUSBWKUP中断，操作模式进入低功耗模式以停止CLK_H。
- f. 如果USB总线上检测到切断信号，VBUSPOWER引脚信号变为"0"且 WAKEUP 输出信号异步启用为"0"。
- g. 根据 WAKEUP 输出信号产生 INTUSBWKUP 中断且低功耗模式取消。CLK_H上电启动。

- h. 中断被启用 2.5 s后(当VBUS切断后需要时间稳定信号)检查UDFSPWCTL<pw_detect>。如果UDFSPWCTL<pw_detect>信号为"1", $\overline{\text{WAKEUP}}$ 信号的启用取决于恢复状态。如果UDFSPWCTL<pw_detect>信号为"0", $\overline{\text{WAKEUP}}$ 信号通过 VBUS的切断信号启用。
- i. 如果主因是恢复, 执行"13.5.7.2 从暂停状态恢复(从USB主机恢复)"所述的顺序.如果主因是切断, 按照下列顺序执行。
- j. <phy_suspend>清零使得 $\overline{\text{PHYSUSPEND}}$ 输出信号解除认定。
设置时钟/模式电路的CGUSBCCTL<USBCLKEN> 为"1"启动CLK_U。
清除中断因素和<wakeup_en>信号以使 $\overline{\text{WAKEUP}}$ 输出信号解除认定。
- k. 使用软件设置UDFSPWCTL<pw_resetb>, 并初始化UDC2AB。

13.5.7.4 远程唤醒暂停状态

远程唤醒暂停状态的程序和信号变化如下所示。

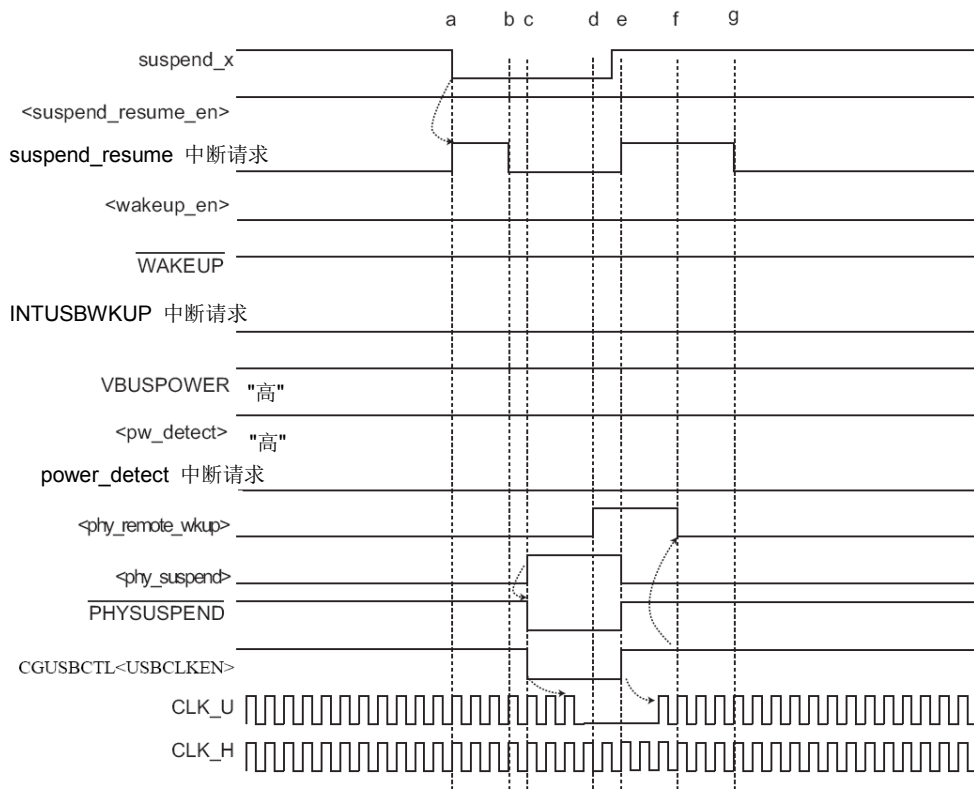


图 13-20 暂停操作/远程唤醒信号

- a. 当检测到USB总线的暂停状态时, UDC2的suspend_x被启用为0, 且INTUSB(suspend_resume)中断发生。
- b. 清除INTUSB(suspend_resume)中断服务程序中的中断源。
- c. 设置UDFSPWCTL<phy_suspend>信号为 "1". <phy_suspend>设置为"1"后, $\overline{\text{PHYSUSPEND}}$ 输出信号被启用为"0"

设置时钟/模式电路的CGUSBCTL<USBCLKEN>为"0" 以停止CLK_U。

- d. 当需要远程唤醒时，设置UDFSPWCTL<phy_remote_wkup>信号为1。设置<phy_remote_wkup>信号为"1" 会使得UDC2在USB总线上进行一次远程唤醒的请求。同时，<suspend_x>信号会异步失效解除认定为1。
- e. 解除认定<suspend_x>信号会引起INTUSB(suspend_resume)中断发生且 PHYSUSPEND 输出信号被解除认定为1。
- f. 设置时钟/模式电路的CGUSBCTL<USBCLKEN>为"1"以启用CLK_U。当CLK_U开始操作时，<phy_remote_wkup> 自动清"0"。
- g. 清除中断源。

13.6 USB装置响应

当检测到硬件复位时，UDC2进行内部初始化并设置各个的寄存器，检测到USB_RESET信号并举响应。本节会探讨UDC2在每一种状态下的操作以及如何从外部控制他们。

1. 当检测到硬件复位时

要确保UDC2在上电操作后硬件复位.硬件复位之后，UDC2初始化内部寄存器，所有EPs处于一种失效状态，这也意味着装置本身是"切断的。"

为将UDC2 的状态设置为"默认"，系统发布"USB_Ready"命令。这条命令的发布会使得UDC2 处于"全速"模式，启用D+上拉电阻并通知主机"连接"。

在这种状态下，仅从主机接收USB_RESET信号。

2. 当检测到USB_RESET信号时

当USB信号检测到总线复位 (USB_RESET)信号时，UDC2 初始化内部寄存器，并将装置设置为"默认"状态。在这种状态下只有EP 0 获得"准备"信号，启用与主机的接口。

3. 当接收到"Set_address"信号请求时

在接收到"Set_address"的请求后，通过将010设置给 UDFS2ADR <configured> <addressed> <default>并将接收到的地址值发送给<dev_adr>，UDC2会处于一种"寻址"状态。控制传输成功完成后，这个寄存器应该被设置(在STATUS阶段结束之后)。

在这种状态下除了EP 0传输给其它EPs的操作不能进行。

4. 当收到"Set_configuration"和"Set_interface"请求时

在收到"Set_configuration"和"Set_interface"请求后, 将值 100 赋给UDFS2ADR <configured> <addressed> <default>, UDC2 将处于"配置"状态。

在"配置"状态下, 可在已经设置的状态下向EP传输数据。

为使EP"准备", 必须进行下列设置:

- 为UDFS2EPxMSZ设置最大数据包长度
- 为UDFS2EPxSTS设置传输模式
- 向UDFS2CMD发布EP_Reset命令

当这些设置做好后EPs就可发送和接收数据。

图 13-21 "装置状态图"。

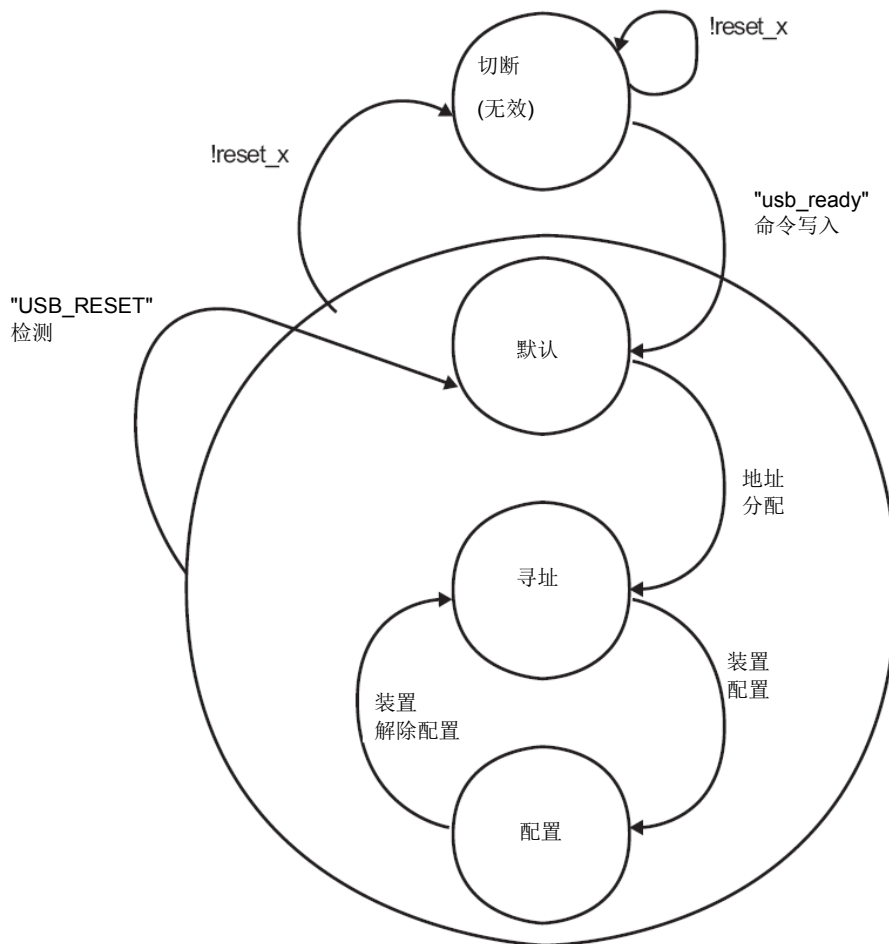


图 13-21 装置状态图

13.7 EP传输的控制流程

13.7.1 EP0

EP0 支持控制传输并作为装置控制接口。EP0 仅支持单数据包模式。

控制传输分为SETUP-阶段，DATA-阶段和STATUS-阶段

传输类型可分类为下列主要类型：

- Control-RD控制读传输
- Control-WR控制写传输(非数据阶段DATA-阶段)
- Control-WR控制写传输(数据阶段 DATA-阶段)

UDC2通过硬件控制这三个阶段。每种类型传输的流程如下所述。

13.7.1.1 Control-RD控制读传输

Control-RD控制读传输的控制流程如下所示。

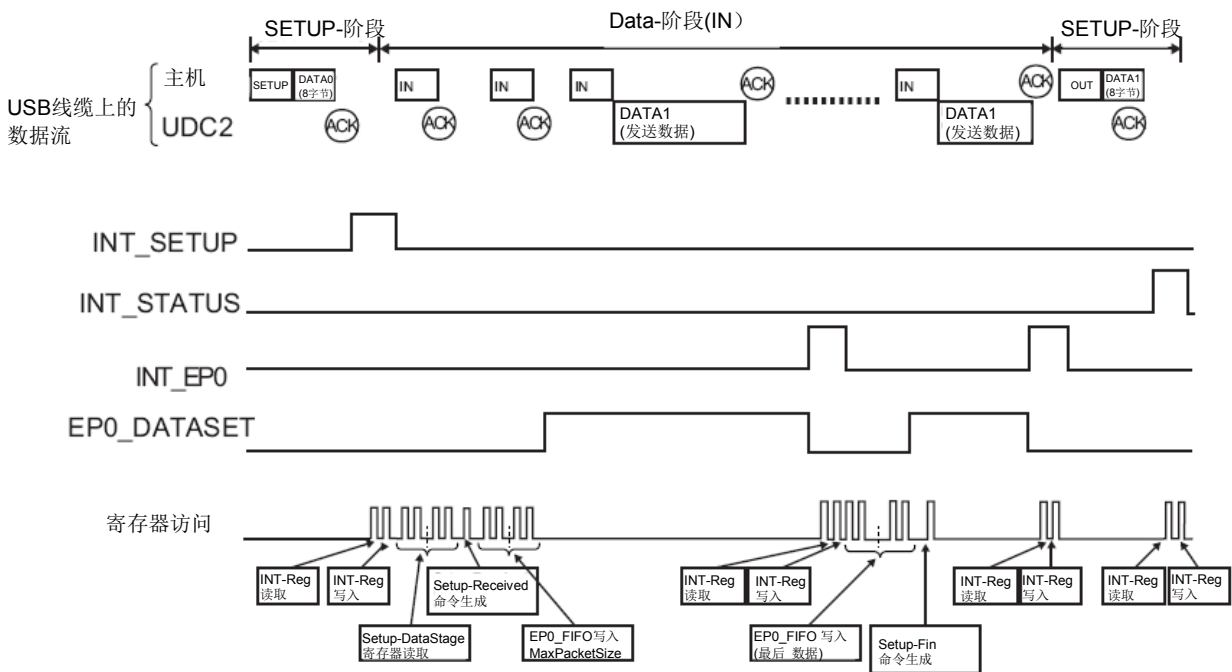


图 13-22 Control-RD传输的控制流程

下列描述是基于假定将UDFS2EP0MSZ<dset>设置为 "EP0_DATASET 标志"。

(1) SETUP-阶段

UDC2当接收到Setup-Token时启用INT_SETUP标志。当将UDFS2INT<i_setup> 写入1时此标志可清除。在对标志进行外部组合的情况下，读取UDFS2INT以确定哪个标志被启用，并将"1"写入相关位。

然后读取数据配置存储寄存器(bRequest-bmRequestType, wValue, wIndex和wLength寄存器)以确定请求类型。

最后，发布"Setup_Received"命令通知UDC2 SETUP-阶段已经完成。既然UDC2在命令发布之前不允许将数据写入EP0-FIFO，他会不停向来自主机的IN-Token返回"NAK"指令直至命令发布。

(2) DATA-阶段

将要传输给IN-Token的数据写入EP0-FIFO。如果要发送的数据的字节长度大于最大数据报长度 MaxPacketSize，在写入前将他们按最大数据包长度分组。当数据数达到MaxPacketSize时，EP0_DATASET标志被启用。

当数据已经无障碍传输给来自主机的IN-Token，UDC2 会使EP0_DATASET标志解除认定并使INT_EP0启用。任何保存的将要传输的数据应该写入EP0-FIFO。

如果将要写入的数据小于MaxPacketSize，发布"EP_EOP"命令给EP0以通知UDC2这是一个短数据包。通过这个命令，UDC2识别数据包结束位置并传输短包数据。

最后，发布"Setup_Fin"命令以通知UDC2 DATA-阶段完成。

(3) STATUS-阶段

当发布"Setup_Fin"命令时，UDC2会自动与 STATUS-阶段握手。当 STATUS-阶段无障碍完成时，INT_STATUS标志被启用。在 "Setup_Fin" 命令发布之前，当从主机接收到一个 STATUS-阶段的数据包时，UDC2 会返回 "NAK" 指令并使得INT_STATUS_NAK 标志启用。所以，如果这个标志被启用，一定要发布"Setup_Fin"命令。

13.7.1.2 Control-WR控制写传输(非DATA-阶段)

Control-WR控制写传输(非DATA-阶段)的控制流程如下所示。

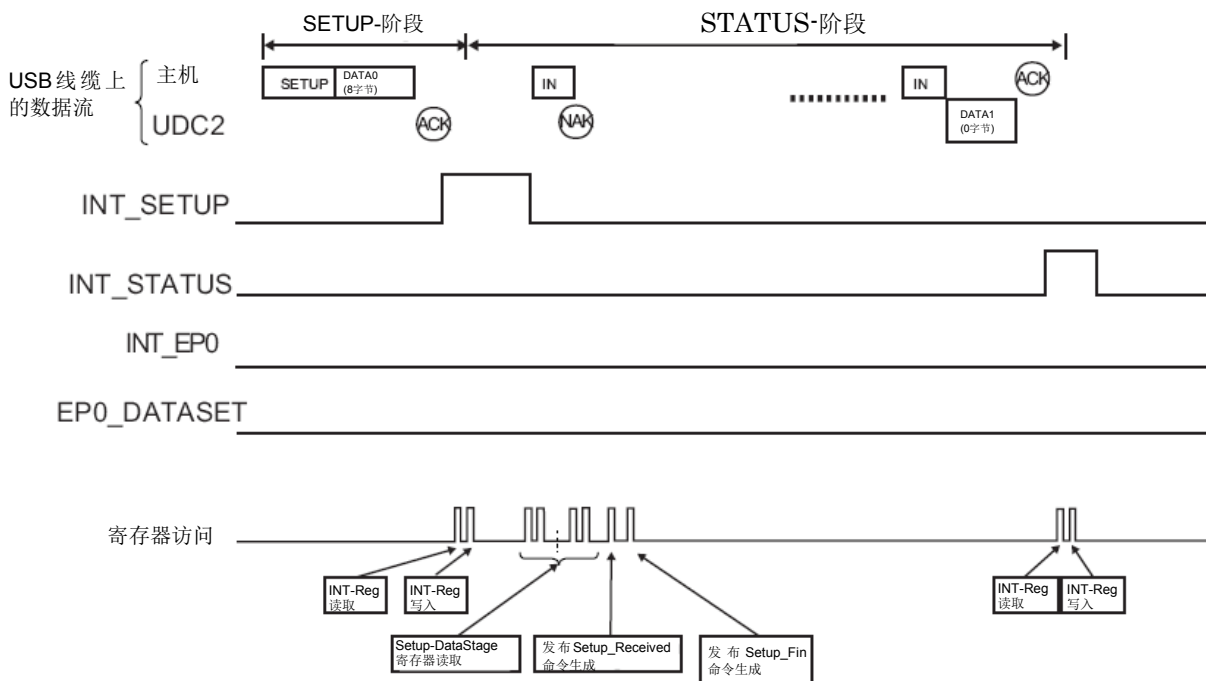


图 13-23 Control-WR控制写传输(无DATA-阶段)控制流程

(1) SETUP-阶段

执行"13.7.1.1 Control-RD传输"所述的相同程序。

(2) STATUS-阶段

发布"Setup_Received"命令之后, 根据每个请求寄存器对UDC2进行访问。当UDC2的所有寄存器访问均完成时发布"Setup_Fin"命令。后续程序基本与"13.7.1.1 Control-RD数据读传输"中的STATUS-阶段所述相同。UDC2 会不停返回"NAK" 指令直至 "Setup_Fin"命令发布。

注: 在'发布"Setup_Received"命令'和 '发布"Setup_Fin"命令'之间, 当依据每种请求对于UDC2的寄存器进行访问时, 在某些情况下STATUS-阶段结束之后需要进行寄存器访问, 比如设置访问请求和设置特征 (TEST_MODE)的情况下。对于标准请求所需要的程序如"13.7.1.5 处理标准请求"所述。

13.7.1.3 Control-WR传输(带DATA-阶段)

Control-WR (带DATA-阶段)的控制流程如下所示。

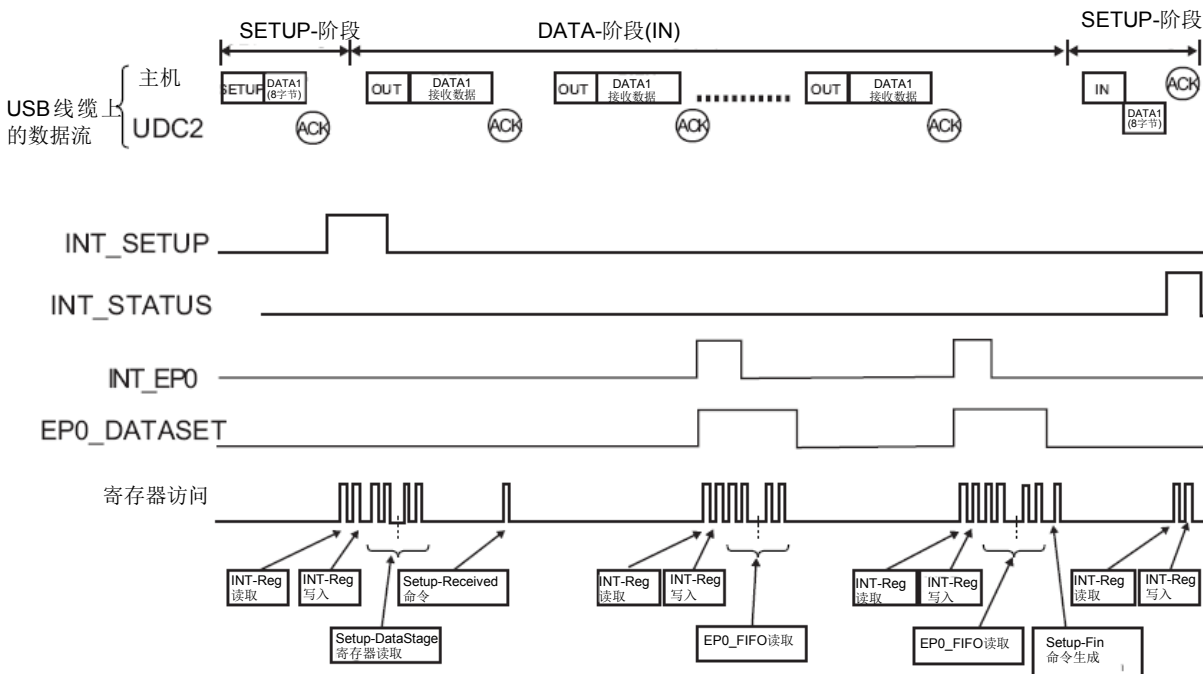


图 13-24 Control-WR (带DATA-阶段)控制流程

(1) SETUP-阶段

SETUP-阶段流程与"13.7.1.1 Control-RD "中所述模式相同。

(2) DATA-阶段

当数据从主机无障碍接收时, UDC2使EP0_DATASET标志和INT_EP0 标志启用。当此标志被启用时, 在确认UDFS2EP0FIFO中接收数据长度后从EP0_FIFO读取数据, 或者从EP0_FIFO读取数据使EP0_DATASET 标志启用。

当已经读取了接收数据字节长度时, UDC2使EP0_DATASET 标志解除认定。

(3) STATUS-阶段

STATUS-阶段 流程与"13.7.1.1 Control-RD "所述模式相同。

13.7.1.4 使用 INT_STATUS_NAK 标志示例

当处理非数据阶段的请求时，INT_STATUS_NAK 标志可在INT_SETUP 标志被清空之前通过从主机接收STATUS-阶段信号变为启用，尤其是在高速传输的时候。在需要尽可能避免这种多中断的情况下，可用请求无DATA-阶段的方法屏蔽INT_STATUS_NAK标志。在这种情况下，基本上应将UDFS2INT<m_status_nak>信号置1，而仅当有DATA-阶段的请求被接收时置0。（Control-RD传输的示例如下所示。）

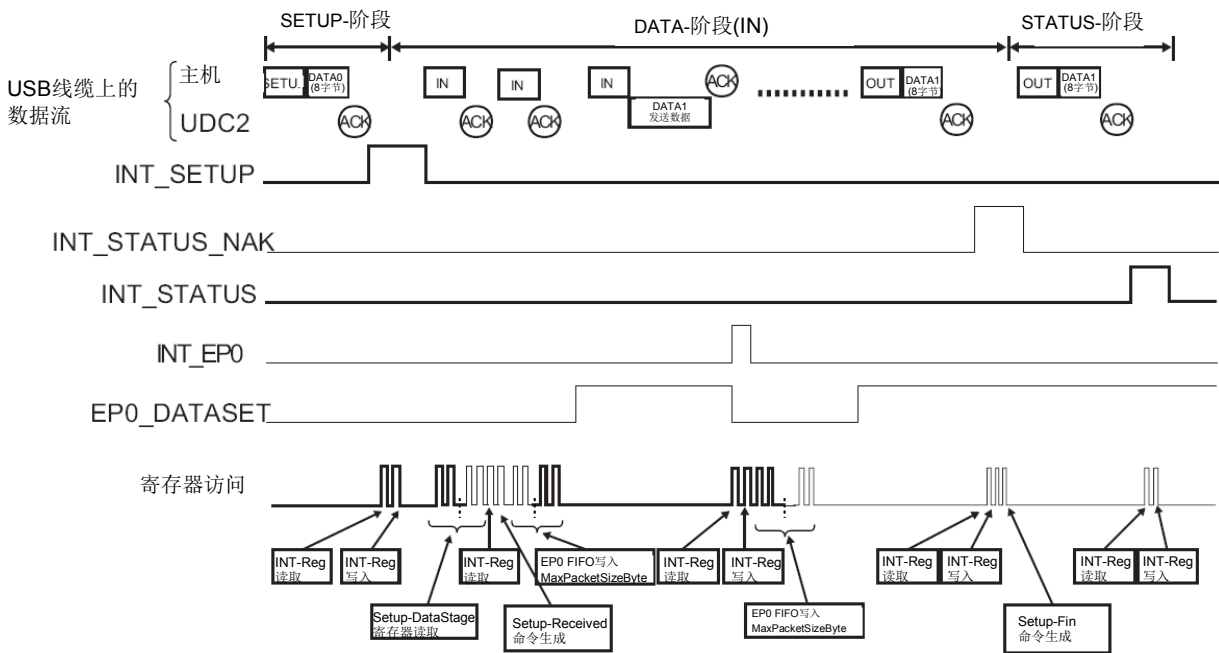


图13-25 在Control-RD传输中使用INT_STATUS_NAK标志示例

(1) SETUP-阶段

INT_SETUP标志启用后，清除被设置为1的UDFS2INT<i_setup>信号，它也应该被清除。

然后通过读取Setup-Data存储寄存器判断该请求是否具有DATA-阶段，设置UDFS2INT <m_status_nak> 信号为0。然后发布"Setup_Received" 命令。

(2) DATA-阶段→STATUS-阶段

INT_STATUS_NAK 标志被启用时，装置也应进行 STATUS 阶段。UDFS2INT<i_status_nak> ，然后发布 "Setup_Fin" 命令。也应将UDFS2INT<m_status_nak>设置为1以便为后续传输做好准备。

13.7.1.5 处理标准请求

收到标准请求时对UDC的寄存器访问实例如下所示。装置在每种状态下的每种请求基本上都有描述（默认，寻址和配置）。

想了解每种请求的寄存器访问的相关信息，请参见13.7.1.1 ， 13.7.1.2和 13.7.1.3。

不过，用户应注意的是，以下提供的说明并未包括USB2.0标准要求的所有详细资料。寄存器访问方法可能因各用户的用途而不同，因此，请务必参考 USB 2.0 规范。还应参考 "接收器"，"描述符类型"，"标准功能选择器"，"试验模式选择器"的USB2.0规范，以及以下说明中所列出的其它条款。

- 对"13.7.1.1 Control-RD传输"的标准要求。
 获取状态 获取说明 获取配置
 获取界面 获取帧
- 对"13.7.1.2 Control -WR传输 (无DATA-阶段)"的标准要求。
 清除功能 设置功能 设置地址
 设置配置 设置界面
- 对"13.7.1.3 Control -WR传输 (有DATA-阶段)"的标准要求。
 设置说明

注 1: 带有双下划线的说明性文字，表示此前访问过UDC2的寄存器。

注 2: 为简单起见，将按以下方式对UDFS2CMD的写入访问进行说明：

(示例 1) 在将 0x0 写入到UDFS2CMD<ep>，以及将 0x4 写入到<com>时

→ 向EP0 发出EP-Stall命令

(示例 2) 在将相关EP写入到UDFS2CMD <ep>，以及将 0x5 写入到<com>时

→ 向相关EP发出EP-Invalid命令

(1) 获取状态请求

为满足该请求，所指定接收端(接收器)的状态被返回。

bmRequestType	bRequest	wValue	wIndex	wLength	数据
1000_0000 1000_0001 1000_0010	GET_STATUS	零	清零 接口 EP	2	装置 界面或 EP 状态

- 为所有状态所共用：

如果wIndex指定的该EP/界面不存在，则向EP0发出EP-Stall命令。

默认状态：

USB2.0 规范并未对各装置的操作作出任何指定。

地址状态：

<接收器> = 装置: 将该装置(表 13-3)相关信息写入到UDFS2EP0FIFO。

<接收器> = 界面: 向EP0发出"EP-Stall"命令

<接收器> = EP: 如果wIndex =0(EP0)，则将EP0(表 13-5)相关信息写入到UDFS2EP0FIFO。如果wIndex≠0(EPx)，则向EP0发出EP-Stall命令。

配置状态:

<接收器> = 装置: 将该装置(表 13-3)相关信息写入到UDFS2EP0FIFO。

<接收器> = 界面: 如果该界面是I wIndex所指定的, 则将界面(表 13-4)相关信息写入到UDFS2EP0FIFO。

<接收器> = EP: 如果EP是wIndex所指定的, 则将相关EP(表 13-5)的信息写入到UDFS2EP0FIFO。

表 13-3 拟由"获取状态"请求返回的装置的相关信息

D15	D14	D13	D12	D11	D10	D9	D8
0	0	0	0	0	0	0	0
D7	D6	D5	D4	D3	D2	D1	D0
0	0	0	0	0	0	Remote Wakeup	Self Powered

RemoteWakeup (D1) 0 表示总线电源, 而 1 表示固有电源。

SelfPowered (D0) 0 表示远程唤醒功能已被禁用, 而 1 表示其已被启用。

表 13-4 拟由"获取状态"返回的界面的相关信息

D15	D14	D13	D12	D11	D10	D9	D8
0	0	0	0	0	0	0	0
D7	D6	D5	D4	D3	D2	D1	D0
0	0	0	0	0	0	0	0

请注意, 所有的位均为0。

表 13-5 拟由获取状态请求返回的EP的相关信息

D15	D14	D13	D12	D11	D10	D9	D8
0	0	0	0	0	0	0	0
D7	D6	D5	D4	D3	D2	D1	D0
0	0	0	0	0	0	0	Halt

Halt (D1) 如果该位为 1, 则其表示相关EP处于"停止"状态。

(2) 清除功能请求

为满足该请求，特定功能被清除并禁用。

bmRequestType	bRequest	wValue	wIndex	wLength	数据
1000_0000 1000_0001 1000_0010	CLEAR_FEATURE	功能选择器	零 接口 EP	零	无

为所有状态所共用：

如果指定了无法被清除(被禁用)的功能选择器(wValue)，则向EP0发出EP-Stall命令。

如果wIndex指定的该EP/界面不存在，则向EP0发出EP-Stall命令。

默认状态：

USB2.0 规范并未对各装置的操作作出任何指定。

地址状态：

<接收器> = 装置： 如果wValue=1，则在用户程序禁用DEVICE_REMOTE_WAKEUP功能。不要求对UDC2进行寄存器访问。

<接收器> = 界面： 向EP0 发出EP-Stall命令。

<接收器> = EP： 如果wIndex≠0(EPx)，向EP0 发出EP-Stall命令。

如果wValue=0，且wIndex=0(EP0)，则清除EP0的停止状态，但不要求对UDC2进行寄存器访问。

配置状态：

<接收器> = 装置： 如果wValue=1，则在用户程序禁用DEVICE_REMOTE_WAKEUP功能。不要求对UDC2进行寄存器访问。

<接收器> = 界面： 向EP0发出EP-Stall命令 (注)。

<接收器> = EP： 如果wValue=0，且wIndex=0(EPx)，则向相关命令发出EP-复位命令。如果wValue=0，且wIndex=0(EP0)，则清除EP0的停止状态，但不要求对UDC2进行寄存器访问。

注：根据对上边带2.0规范(即"不存在任何界面用功能选择器")的判读，拟使EP0停止工作。有关详细资料见USB规范。

(3) 设置功能请求

为满足该请求，需设置或启用特定功能。

BmRequestType	BRequest	wValue	wIndex		wLength	数据
1000_0000	SET_FEATURE	功能选择器	零	测试选择器	零	无
1000_0001			接口			
1000_0010			EP			

为所有状态所共用：

如果指定了无法被设置(被启用)的功能选择器(wValue)，则向EP0发出EP-Stall命令。

如果wIndex的低字节指定的该EP/界面不存在，则向EP0发出EP-Stall命令。

注：在使用供应商特有的非标测试选择器时，应进行适当的操作。

默认状态：

除了上述的TEST_MODE之外，USB2.0 规范并未对各装置的操作作出其它任何指定。

地址状态：

<接收器> = 装置： 如果wValue=1，则在userÅfs端禁用DEVICE_REMOTE_WAKEUP功能。不要求对UDC2进行寄存器访问。

<接收器> = 界面： 向EP0发出EP-Stall命令。

<接收器> = EP： 如果wIndex的低字节≠0(EPx)，则向EP0发出EP-Stall命令。
如果wValue=0，且wIndex的低字节=0 (EP0)，则让EP0进入停止状态。(注 2)：

配置状态：

<接收器> = 装置： 如果wValue=1，则在userÅfs端启用DEVICE_REMOTE_WAKEUP功能。不要求对UDC2进行寄存器访问。

<接收器> = 界面： 向EP0发出EP-Stall命令(注 1)。

<接收器> = EP： 如果wValue=0，且wIndex的低字节≠0(EPx)，则向EP0发出EP-Stall命令。
如果wValue=0，且wIndex的低字节=0 (EP0)，则让EP0进入停止状态(注 2)。

注 1：根据USB规范(即"不存在任何界面用功能选择器")的判读，拟使EP0停止。有关详细资料见USB规范。

注 2：在USB 2.0规范中，给出了"不必要，也不建议执行EP0的停止功能"等说明。因此，其可被理解为：在这种情况下，不必将UDC2 设置为停止状态。

为使EP0 实际上处于停止状态，用户必须管理该"停止状态"。

因而，在"停止状态"收到一项请求时，必须执行"在DATA-阶段/STATUS-阶段向EP0发出EP-Stall命令"这一步骤(即使EP0 被设置为停止状态，UDC2 也会在收到Setup-Token时取消该停止状态，并返回"ACK")。

就这点而论，在收到EP0所需的SetFeature/ClearFeature时，该步骤会随着用户的用途而有所不同。

(4) 设置地址请求

为满足该请求，需设置设备地址。

BmRequestType	BRequest	wValue	wIndex	wLength	数据
0000_0000	SET_ADDRESS	设备地址	零	零	无

对于该请求，需在该 STATUS - 阶段结束后 2 ms 之内进行以下所述的寄存器访问。
(在Setup_Fin发出之前，不应改变该设备地址)。

默认状态：

wValue=0: 保持该默认状态。不要求对UDC2进行寄存器访问。

wValue≠0: 将wValue设置为UDFS2ADR<dev_adr>，并将 010 设置为<已配置>，<已寻址>和 <默认>。
UDC2即进入该地址状态。

地址状态：

wValue=0: 将 0x00 设置为UDFS2ADR<dev_adr>，并将 010 设置为<已配置>，<已寻址>和 <默认>。
UDC2 即进入该默认状态。

wValue≠0: 将wValue设置为UDFS2ADR<dev_adr>。UDC2 即被设置为新的设备地址。

配置状态：

在USB 2.0 规范中，并未就各装置操作做出任何指定。

(5) 获取描述符请求

在收到该请求时，所指定的描述符会被返回。

BmRequestType	BRequestdt	wValue	wIndex	wLength	数据
1000_0000	GET_DESCRIPTOR	描述符类型和 描述符指标	零或 语言ID	描述符长度	描述符

为所有状态所共用：

将wValue所指定的描述符信息，写入到wLength所指定字节长度的UDFS2EP0FIFO。如果拟写入的该字节长度大于EP的MaxPacketSize，则用户需将该数据分开，并分数次进行写入(有关详细资料，请参考"13.7.1.1 Control-RD传输")(如果该描述符的长度长于wLength，则可写入来源于该描述符开头部分的wLength字节的信息。如果该描述符的长度小于wLength，则写入该描述符的全信息)。

如果该用户不支持wValue所指定的描述符，则可向EP0发出EP-Stall命令。

(6) 设置描述符请求

BmRequestType	BRequest	wValue	wIndex	wLength	数据
0000_0000	SET_DESCRIPTOR	装置类型和描述符指标	语言ID或零	描述符长度	描述符

为所有状态所共用：

在不支持该请求时，向EP0发出EP-Stall命令

默认状态：

USB2.0 规范并未对各装置的操作作出任何指定。

地址状态与配置状态：

读取UDC2从UDFS2EP0FIFO收到的说明的相关信息。

(7) 获取配置请求

BmRequestType	BRequest	wValue	wIndex	wLength	数据
1000 0000	GET_CONFIGURATION	零	零	1	配置值

默认状态:

在收到该请求时，当前设备的配置值会被返回。

地址状态:

将 0x00 写入到UDFS2EP0FIFO。如未进行如此配置，则应返回 0。

配置状态:

将当前配置值写入到UDFS2EP0FIFO。

由于已进行过如此配置，因此，应返回 0 以外的值。

(8) 设置配置请求

为满足该请求，需设置装置配置。

BmRequestType	BRequest	wValue	wIndex	wLength	数据
0000 0000	SET_CONFIGURATION	配置值	零	零	无

默认状态：

USB2.0 规范并未对各装置的操作作出任何指定。

地址状态：

在wValue = 0时：

- 保持该地址状态。不要求对UDC2进行寄存器访问。

当wValue≠0，且wValue是可匹配该描述符的配置值时：

- 将 100 设置为UDFS2ADR<已配置> <已寻址> <默认>。

<对于拟采用的EPs>

- 将MaxPacketSize设置为UDFS2EPxMSZ<max_pkt>。
- 将相应的值设置为UDFS2EPxSTS<pkt mode>, <bus sel>, <dir>, <t type>和<num mf>。
- 向相关EPs发出EP-Reset命令。

在wValue≠0，且wValue为不匹配该描述符的配置值时：

- 向EP0发出EP-Stall命令。

配置状态：

在wValue = 0时：

- 将 010 设置为UDFS2ADR<已配置> <已寻址> <默认>。
- 发出All-EP-Invalid命令。

在Value≠0，且其为可匹配该描述符的配置值时：

<对于拟采用的EPs>

- 将该MaxPacketSize设置为UDFS2EPxMSZ<max_pkt>。
- 将相应的值设置为UDFS2EPxSTS<pkt mode>, <bus sel>, <dir>, <t type>和<num mf>。
- 向相关EPs发出EP-Reset命令。

<对于拟变为未使用的EPs>

- 向相关EPs发出EP-Invalid命令。

在wValue≠0，且wValue为不匹配该描述符的配置值时：

- 向EP0发出EP-Stall命令。

(9) 获取界面请求

为满足该请求，需返回该指定界面所设置的AlternateSetting值。

BmRequestType	BRequest	wValue	wIndex	wLength	数据
1000_0001	GET_INTERFACE	零	接口	1	备用设置

为所有状态所共用：

如果该界面由wIndex指定，则向EP0 发出EP-Stall命令。

默认状态：

USB2.0 规范并未对各装置的操作作出任何指定。

地址状态：

向EP0 发出EP-Stall命令。

配置状态：

将wIndex所指定界面的当前备用设置值写入到UDFS2EP0FIFO。

(10) 设置界面请求

为满足该请求，需设置所指定界面的备用设置值。

BmRequestType	BRequest	wValue	wIndex	wLength	数据
0000_0001	SET_INTERFACE	备用设置	接口	零	无

为所有状态所共用：

如果wIndex所指定的界面不存在，或wValue所指定的备用设置不存在，则向EP0发出EP-Stall命令。

默认状态：

USB2.0 规范并未对各装置的操作作出任何指定。

地址状态：

向EP0发出EP-Stall命令。

配置状态：

<对于拟在指定界面的备用设置中使用的EPs>

- 将该MaxPacketSize设置为UDFS2EPxMSZ<max_pkt>。
- 将相应的值设置为UDFS2EPxSTS<pkt mode>, <bus sel>, <dir>, <t type>和<num mf>。
- EP-Reset向相关EPs发出EP-Reset命令。

<对于拟变为未使用的EPs>

- 向相关EPs发出EP-Invalid命令。

(11) Synch帧请求

为满足该请求，需返回该EP的Synch帧。

BmRequestType	BRequest	wValue	wIndex	wLength	数据
1000 0010	SYNCH_FRAME	零	EP	2	帧数

为所有状态所共用：

如果wIndex所指定的EP不支持该请求，则向EP0发出EP-Stall命令。

默认状态：

USB2.0 规范并未对各装置的操作作出任何指定。

地址状态：

向EP0发出EP-Stall命令。

配置状态：

将wIndex所指定EP的帧数写入到UDFS2EP0FIFO。

13.7.2 EP0 除外EPs

EP0 除外EPs 支持批量(发送/接收)，中断(发送/接收)以及等时(发送/接收)传输，用于发送与接收数据。它们还支持可启用高速数据通信的数据包模式。

13.8 暂停/恢复状态

UDC2 根据来自主机的信号状态，进入暂停状态。其还按照主机或 UDC2 的要求恢复操作，从暂停状态返回。

各状态之间的转换如以下所述。

13.8.1 转入暂停状态

虽然主机在正常状态会按照给定的时间间隔 (FS: 1 ms) 发出 SOF，但在其试图让该设备暂停时，会停止向该设备发出该SOF，且USB信号线的的数据不会改变，并保持在空闲状态。UDC2始终监控源自PHY的该"line_state"，并在空闲状态的检测时间长于 3 ms或以上时，判断其是否处于暂停状态或USB_RESET。如被判定为处于暂停状态，则会启用"suspend_x"为"低"，并进入暂停状态。

请注意，在UDC2 被暂停期间寄存器访问不可用，原因是CLK由时钟/方式控制电路供电。

13.8.2 从暂停状态恢复

可以两种方式从暂停状态恢复：从主机主机输出一个恢复状态，或从UDC2进行远程唤醒(输入一个恢复状态)。

各种情况下的恢复步骤如以下所述。

13.8.2.1 通过主机输出实现恢复

在某个恢复状态是由主机输出时，UDC2 取消启用suspend_x为"高"，并宣布从中止状态恢复。

13.8.2.2 通过UDC2 远程唤醒实现恢复

某些应用可能不支持远程唤醒功能，且其需在总线枚举时获得USB主机允许。用户不应启用"唤醒"，但经系统允许的情形除外。

如系统允许，则启用"唤醒"引脚会导致UDC2 向主机输出某个恢复状态，以启动恢复。请注意，在UDC2 被暂停时，时钟/模式控制电路也会停止向时钟供电，因此，用户应保持启用唤醒，直至其恢复。应在suspend_x被启用为"低" 2 ms 或以上之后，操作该远程唤醒。

13.9 USB-Spec2.0 设备控制器附录

13.9.1 附录A系统电源管理

在USB中，除正常传送操作之外，还指定了与主机提供的复位和暂停枚举和电源控制装置信号(D+/D-)有关的操作。该附录提供拟连接的USB 2.0 PHY，以及与D+/D-信号有关的步骤所需的系统级时钟控制的信息。有关各步骤的详细资料，请务必检查USB规范 2.0 版，USB I/O规范。

附录A中的各字符如以下所述。

1. 复位：

从USB主机(以下称为"该主机")进行USB设备初始化用D+/D-信号的操作(以下称为"该设备")。复位之后进行枚举，然后开始批量传输等正常传输操作。该设备一旦连接，即被复位。该设备还需支持在其它任何时间进行的复位操作。

2. 暂停

如果该D+/D-线路上连续 3 ms或以上未出现由主机启动的总线活动(包括SOF)，则需让设备进入暂停模式，以减少功耗。在这种情况下，设备必须执行某些操作(例如让时钟停止工作)。

3. 恢复

从暂停模式恢复正常操作用D+/D-信号的操作。可由主机或该设备启动恢复操作。从该设备恢复操作，被称为"远程唤醒"。

各操作的说明如下。该()内的时间，是USB 2.0 规范中的值。

13.9.1.1 连接/断开操作

(1) 连接操作

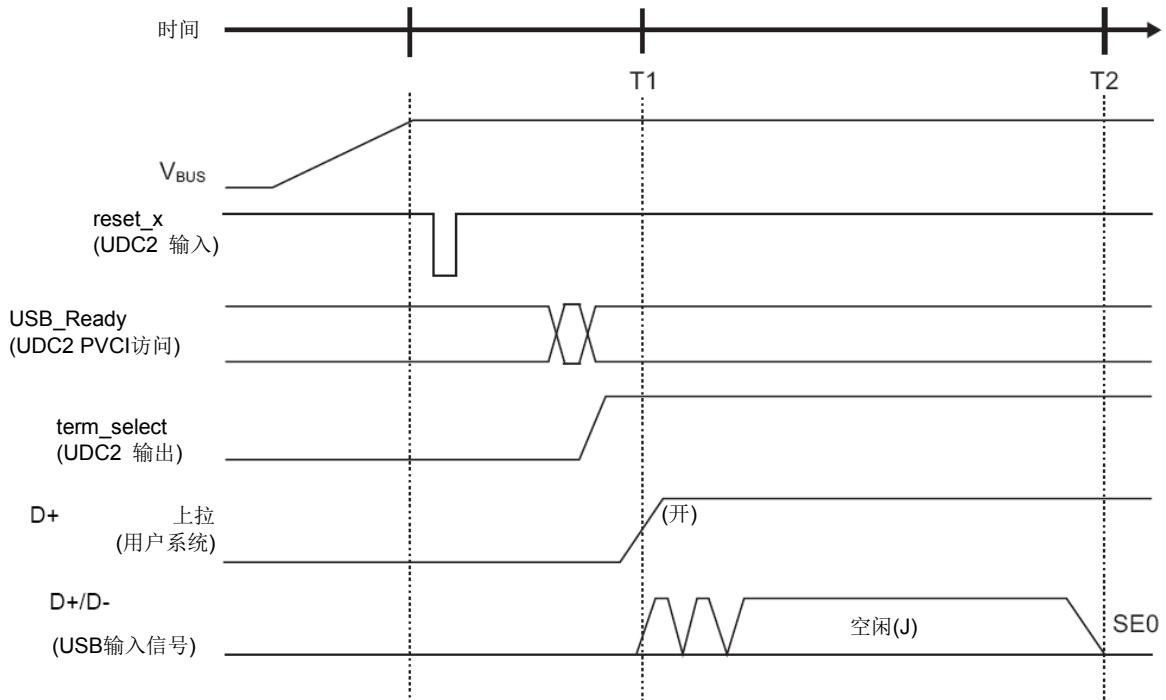


图 13-26 连接操作时序

T0: VBUS检测

在VBUS被检测到时，系统复位(reset_x input) 应被应用于UDC2。xcvr_select 为"高"，而term_select为"低"。

T1: 设备连接(不迟于T0之后 100 ms)

该设备必须在不迟于VBUS检测(T0)之后 100 ms时启用D+，以将已连接状态通知该主机。因此，在VBUS被检测到时，该设备随时可与主机通信，系统应访问UDC2中的UDFS2CMD，以设置USB_Ready命令。然后，用户系统用软件设置该端口，以启用该D+上拉。

T2: USB 复位开始 (100 ms之后 100 ms)

(2) 断开操作

在某个断开状态被检测到时，建议对UDC2 应用一次系统复位。

13.9.1.2 复位操作

此处的"复位"指USB 2.0 规范中所定义的"复位信号", 而不是系统复位。
(reset_x) ~ UDC2。

(1) 复位之后在FS模式下操作时

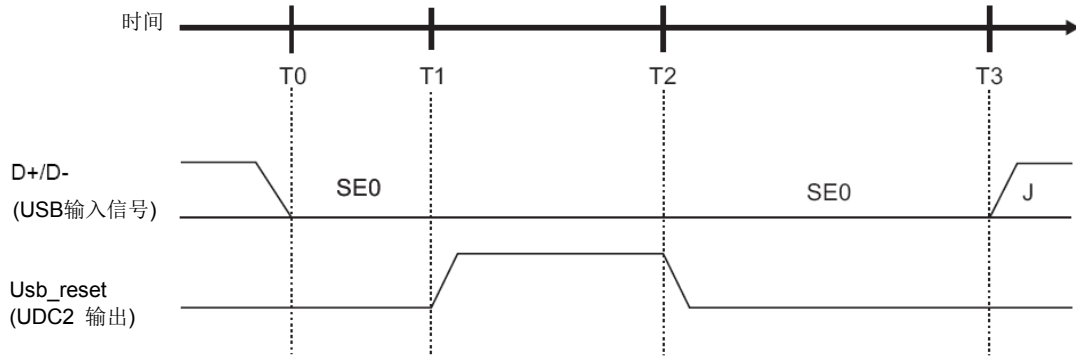


图 13-27 复位操作时序

T0: 重置开启

一旦从主机识别出SE0, UDC2 即开始计数, 以识别该复位。

T1: 复位识别(T0 之后 2.5 μ s)

在T0 之后约 68 μ s检测到SE0 时, UDC2 识别出源自主机的该复位, 并使usb_reset变为"高"。

T2: USB复位的取消启用

此时, usb_reset在T1之后 3.5 ms变为"低"。

T3: 复位结束(T0 之后 10 ms)

在源自主机的SE0完成, 且设备进入空闲状态时, 这表明复位操作结束。从主机进行的复位周期持续至少 10 ms。

(2) 复位操作说明

复位之后寄存器的初始化

在源自主机的复位完成时(在usb_reset从"高"变为"低"时), UDC2 的内部寄存器均被初始化(有关各寄存器的初始值, 请参看"13.4 寄存器")。

注意, 在usb_reset为"高"期间设置的各寄存器也会被初始化。因此, 在复位周期结束时, 应设置各UDC2 寄存器。

复位之后的DMA传输 (EP-I/F访问)

在DMA传输期间发生源自主机的复位时, UDFS2EPxSTS被初始化, 且总线访问模式被设置为"公用总线访问"。因此, 无法正确地继续进行DMA传输。在复位发生时, DMA控制器也必须被初始化。

在复位后的枚举操作期间, 配置各EP, 然后通过UDFS2CMD中设置EP_Reset命令, 初始化该EPs。

13.9.1.3 暂停操作

(1) 暂停操作

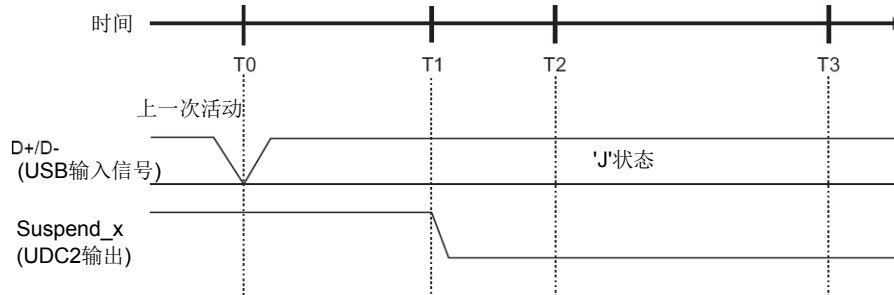


图 13-28 暂停操作时序

T0: 总线活动结束

在检测到源于主机(数据包端)的总线活动结束时, UDC2 开始计数, 以识别暂停。

T1: 暂停的识别(T0 之后 3 ms)

在T0 之后 3 ms检测到"FS-J"时, UDC2 识别出暂停, 并使suspend_x变为"低"。

T2: 远程唤醒开始启用(T0 之后 5 ms)

在T0 之后 5 ms时, 启用从该设备恢复操作(远程唤醒)。

T3: 过渡到暂停状态(T0 之后 10 ms)

该设备必须在T0 之后 10 ms内进入暂停状态。在这期间, 必须执行设备系统进入暂停状态所需的步骤(例如停止CLK_U)。

需控制时钟/模式控制电路, 以停止CLK_U ~ UDC2。

(2) 暂停操作说明

暂停状态期间的内部寄存器

在暂停状态期间, UDC2 可保有各内部寄存器值, FIFOs的内容, 以及各标志的状态。在通过恢复操作退出暂停状态之后, 仍会保留这些值和状态。

在CLK_H ~ UDC2 被停止时, 无法通过PVC-I/F和EP-I/F访问UDC2 中的内部寄存器。

13.9.1.4 恢复操作

(1) 通过主机恢复操作

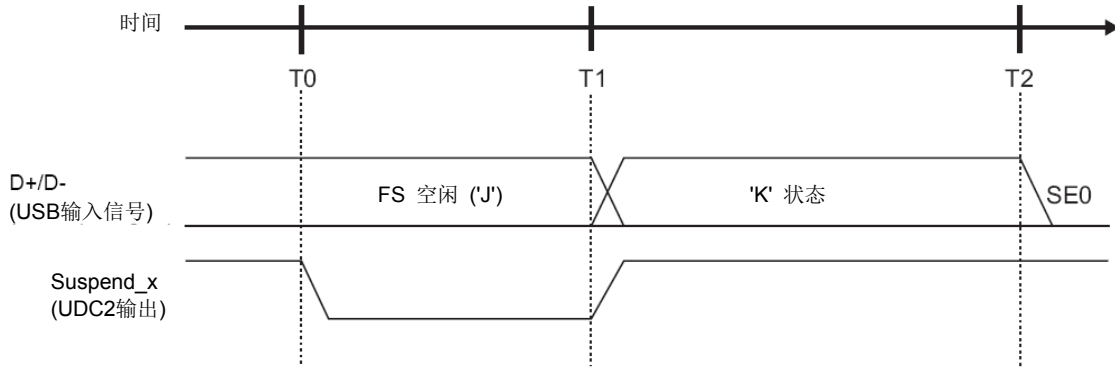


图 13-29 通过主机恢复操作计时

T0: UDC2 的suspend_x输出"低"。

T1: 主机恢复开始(无计时规范)

主机可在任意时间开始恢复操作("FS-K"),以将该设备从暂停状态唤醒。此时,UDC 将suspend_x设置为"高"(即使CLK_U ~ UDC2 被停止, suspend_x仍会变为"高")。

在暂停期间,在CLK_H ~ UDC2 停止时,可通过控制时钟/模式控制电路恢复该CLK_H。

在CLK ~ UDC2 被停止时,需控制clk_em。

T2: 主机恢复结束(T1 之后 20 ms)

主机恢复操作 ("FS-K")持续时间超过 20 ms,并在"SE0"之后完成。

(2) 通过装置恢复操作(远程唤醒)

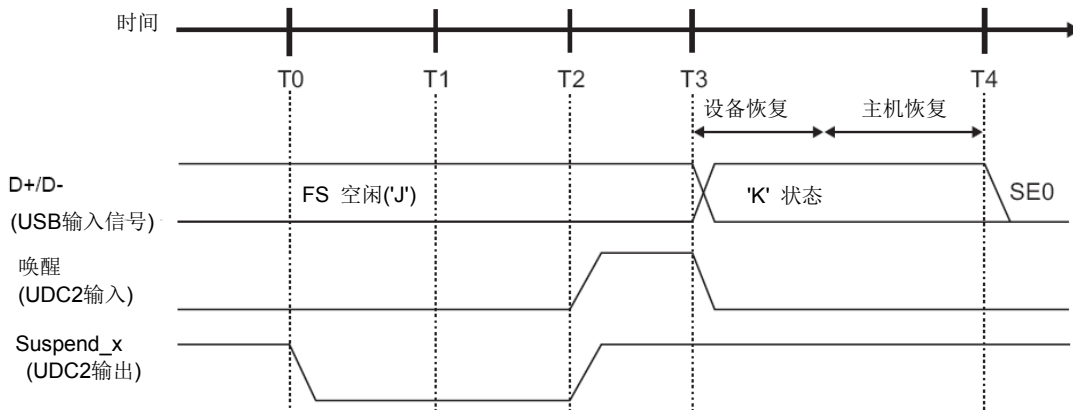


图 13-30 远程唤醒操作计时

T0: UDC2 的suspend_x输出"低"。

T1: 远程唤醒开始启用 (T0 之后 2 ms)

利用UDC2 的唤醒输入, 即可让该设备退出暂停状态。注意, USB规范禁止在暂停状态开始之后 5 ms时进行远程唤醒。至少应在T0 之后 2 ms将唤醒信号设置为"高", 原因是从暂停操作到T0 之间已消逝了 3 ms。

T2: 对UDC2 的唤醒输入为"高"(T1 之后)

将该唤醒信号设置为"高"。并未针对该操作指定任何的计时要求。此时, UDC将suspend_x设置为"高"(即使CLK_H ~ UDC2 被停止, suspend_x仍会变为"高")。UDC2 要求时钟输入开始恢复操作("FSK")。然后, 保持在"高"状态下的唤醒, 直至时钟电源得到恢复。

T3: 设备恢复的开始

在CLK_H输入至UDC2 被恢复时, UDC2 开始设备恢复("FS-K")。该设备恢复周期为 2 ms。在确认设备恢复之后, 主机开始主机恢复操作。

T4: 主机恢复结束(T3 之后 20 ms)

主机恢复操作 ("FS-K")持续时间超过 20 ms, 并在"SE0"之后完成。

(3) 恢复操作说明

对远程唤醒的限制如以下所述。

如需与设备系统一样支持远程唤醒, 则该设备必须在配置描述符中通知主机"远程唤醒功能已启用"。默认情况下, 即使可支持远程唤醒, 其也会被禁用。只有在被主机请求启用后, 才能使用远程唤醒。只有在满足这些条件的情况下, 才允许借助于唤醒输入使用远程唤醒。

在使用该功能时, 务必参看USB 2.0 规范的第 13.8 条中所提供的详细说明。

13.9.2 与 "将奇数个字节设置为 MaxPacketSize" 有关的附录B

13.9.2.1 在UDFS2EPxMSZ中设置一个奇数

USB规范允许将各EP的MaxPacketSize (以下称为MPS)设置为等时和中断传输所需的任一奇数或偶数个字节(对于控制器和成批交易, 仅可设置一个偶数)。

在UDC2 中, 需通过UDFS2EPxMSZ<max_pkt>设置MPS。UDC2 的EP FIFOs仅支持偶数个字节。因此, 通常建议将MPS设置为偶数个字节。

在通过奇字节使用MPS时, 有可能让<max_pkt max_pkt>变为奇数。不过, 在表 13-6 中给出了通过某总线访问方法实现的限制。如果是EP随机访问, 则无法在<max_pkt>中为某个发送EP设置一个奇数。在这种情况下, 应在<max_pkt>中设置一个偶数, 且对该EP FIFO的访问写入应受控, 以执行奇数个最多写入字节 (例如, 在 MPS 为 1023 字节时, 应将<max_pkt> 设置为 1024 字节)。

表 13-6 对max_pkt设置的限制

	接收EP	发送EP
公用总线访问(PVCI-IF)	可设置一个奇数或偶数	可设置一个奇数或偶数
EP随机访问(EP-I/F)	可设置一个奇数或偶数	仅可设置一个偶数

根据上述内容, 以下各页将说明如何将奇数个字节设置为各总线访问方法的MPS。

(1) 接收EP和公用总线访问

可在<max_pkt>中设置奇数个或偶数个字节。两种情况的访问方法相同。

(2) 发送EP和公用总线访问

可在<max_pkt>中设置奇数个或偶数个字节。

不过, 在为写入最大字节数(max_pkt=奇数)而进行公用总线访问时, 必须遵守以下各点的要求。

以下给出了一个示例, 其中<max_pkt>=5, 且拟写入该最大字节数(5 字节)。

在最后一次上次访问时 (第 5 字节), 需确认udc_be = 01。

由于其为MPS的访问, 不要在UDFS2CMD中发出EP_EOP命令。

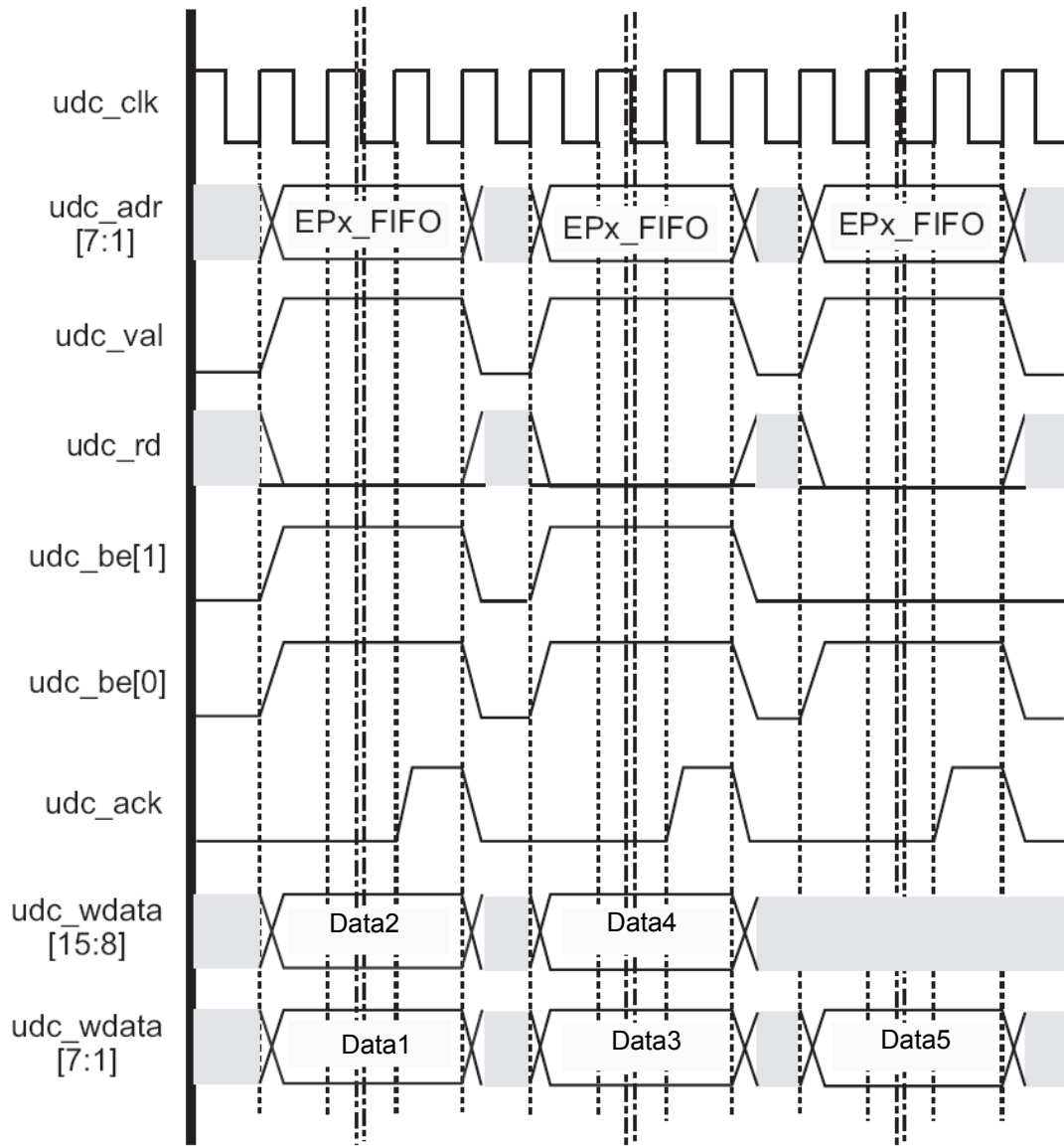


图 13-31 MPS写入max_pkt = 奇数访问(公用总线访问))

(3) 接收EP和EP随机访问

可在<max_pkt>中设置一个奇数或偶数。两种情况的访问方法相同。

(4) 发送EP和EP随机访问

仅可在<max_pkt>中设置偶数个字节。在将奇数个字节用作发送EP的MPS时，必须进行以下设置。

在MPS时为 1023 时

- 将 <max_pkt>设置为 1024。
- 可被写入到EP中的最大字节数为 1023 字节(不允许写入第 1024 个字节)。
- 拟由固件管理的EP描述符的"wMaxPacketSize"应被设置为 1023(即为拟通过获取描述符请求发送给USB主机的该)。

以下给出了一个示例，其中max_pkt=1024，且拟写入该最大字节数(1023 字节)。

- 在最后一次访问(第 1023 个字节)时，确认epx_w_be = 01。

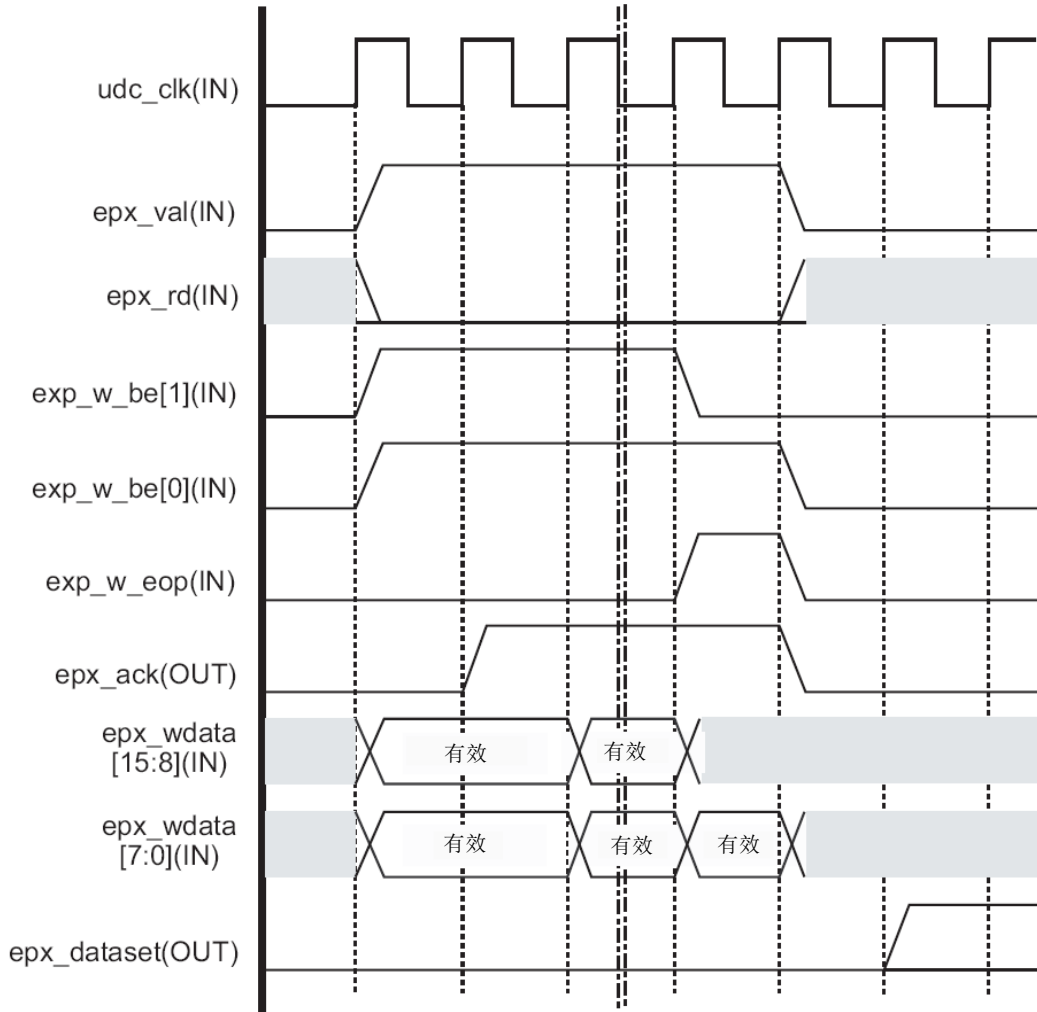


图 13-32 MPS (奇数)写入max_pkt = 偶数访问(EP随机访问)

13.9.3 附录C等时译码器

在等时传输期间，数据的等时性很关键，且每帧都可发生传输。因此，在利用等时传输访问某个EP(FIFO)时，要求必须达到某个性能水平(速度)。在UDC2中，可从PVCII-I/F和EP-I/F中选择针对各EP的访问方法。可从单模和双模中选择FIFO配置。不过，对于采用等时传输的EP而言，建议采用EP-IF和双模。

13.9.3.1 利用等时传输访问某个EP

在FS模式下，最大数据有效负载大小为 1023 字节。在利用双模传输 1023 字节时，RAM必须达到 2048 字节。每帧(1 ms)的传输均在FS模式下进行。在FS模式下，可在一帧内进行一次交易。

(必须在相关的UDC2中设置有效负载大小和交易数目等信息。还必须将该信息作为拟发送给主机的EP描述符信息进行管理)。

13.9.3.2 在采用等时传输时对针对EP的命令用法的限制

与其它传输相比，等时传输对握手，切换，一帧内的交易数目等项目都设置了某些限制，对可使用命令的类型进行了限制。一般不得在等时传输期间将命令发给EPs。在某个请求正被处理期间，如有必要可使用EP_Reset或EP_Invalid

(在将PVCII-I/F用作EP访问方法时，需使用EP_EOP命令)。

(关于该附录)

有关USB规范的相关描述，请务必查看USB(2.0 版)。

14. 串行通道(SIO/UART)

14.1 概述

该装置有两种模式可用于该串行通道，其中之一为同步通信模式(I/O接口模式)，另一种为异步通信模式(UART模式)。

其特点给出如下。

传输时钟

- 由预分频器从外设时钟($\phi T0$)频率分频为1/2, 1/8, 1/32, 1/128。
- 使"从预分频器输出时钟脉冲频率分频为 1-16成为可能。
- 使从预分频器输出时钟脉冲频率分频为 1, N+m/16 (N=2-15, m=1-15), 16。(仅UART模式)成为可能。
- 可用的系统时钟(仅UART模式)。

双缓冲器/FIFO

发送与接收的可用双缓冲区功能，以及可用FIFO缓冲区，总计最多4-字节。

I/O接口模式

- 传输模式：半双工(发送/接收)，全双工
- 时钟：输出(固定上升缘)/输入(可选上升/下降缘)
- 使"指定连续发送的间隔时间"成为可能。

UART模式

- 数据长度:7 位, 8 位, 9 位
- 增加奇偶校验位(应与 9 位数据长度对应)
- 拟使用唤醒功能的串行链接
- 握手功能 $\overline{\text{CTS}}$ 引脚

在以下说明中，"x"表示通道编号。

14.2 SIO模块的规格差异

TMPM365FYXBG具备两条SIO通道。

各通道均独立工作。以下列出了所使用的引脚，中断，DMA请求与各通道中的UART源时钟。

表 14-1 SIO模块的规格差异

	引脚名称			中断		DMA request	UART源时钟
	TXD	RXD	$\overline{\text{CTSx}}$ / SCLKx	接收中断	发送中断		
通道 0	PE0	PE1	PE2	INTRX0	INTTX0	支持	TB8OUT
通道 1	PC0	PC1	PC2	INTRX1	INTTX1	支持	TB8OUT

14.3 配置

图 14-1 给出了SIO方块图。

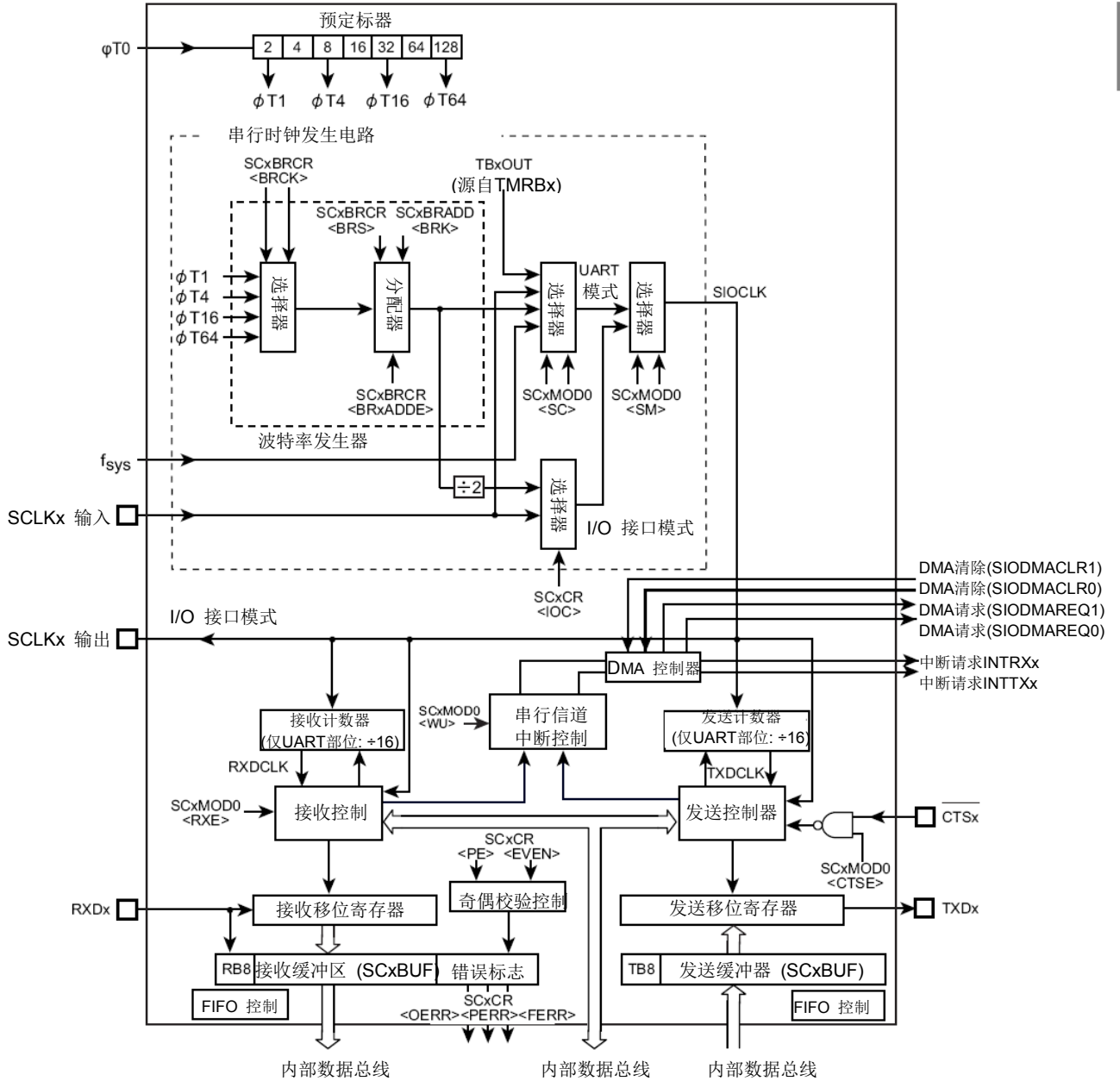


图 14-1 SIO方块图

14.4 寄存器说明

14.4.1 各通道中的寄存器列表

各通道寄存器与地址给出如下：

通道 x	基本地址
通道 0	0x400E_1000
通道 1	0x400E_1100

寄存器名称(x=0 ~ 1)		地址(基+)
启动寄存器	SCxEN	0x0000
缓冲寄存器	SCxBUF	0x0004
控制寄存器	SCxCR	0x0008
模式控制寄存器 0	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
DMA请求启用寄存器	SCxDMA	0x0034

注 1：不要在数据发送或接收期间修改任何控制寄存器

14.4.2 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	类型	功能
31-1	-	R	读作 0。
0	SIOE	R/W	SIO操作 0: 禁用 1: 启用 指定SIO操作。 设置<SIOE> = "1", 即可使用该SIO。 在该操作被禁用时, 不会向TMRB模块中的其它寄存器提供任何时钟脉冲。这样能降低功耗。 如果在执行SIO操作后即禁用, 则各寄存器中都会保持该设置, 但SCxTFC<TIL>除外。

注: 如果SCxEN<SIOE>="0" (停止SIO操作)或操作模式被改为空闲模式(SCxMOD1<I2SC>="0") (使SIO操作在IDLE模式下停止), 则SCxTFC会被重新初始化。

14.4.3 SCxBUF (缓冲寄存器)

SCxBUF可用作写入操作发送缓冲器或FIFO，以及读出操作的接收缓冲器或FIFO。

	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	类型	功能
31-8	-	R	读作"0"。
7-0	TB[7:0]/RB [7:0]	R/W	[写] TB :发送缓冲器/FIFO [读] RB :接收缓冲器/FIFO

14.4.4 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	类型	功能
31-8	-	R	读作"0"。
7	RB8	R	接收数据位8 (适用于UART) 在 9 位的UART模式下, 接收数据的第 9 位。
6	EVEN	R/W	奇偶性(适用于UART) 0: 奇数 1: 偶数 选择偶或奇校验。 "0"奇校验, "1": 偶校验 奇偶校验位仅可用于 7-或 8-位UART模式。
5	PE	R/W	加奇偶(适用于UART) 0: 禁用 1: 启用 可控制启用/禁用奇偶性。 奇偶校验位仅可用于 7-或 8-位UART模式。
4	OERR	R	超限操作错误标志(注) 0: 正常工作 1: 错误
3	PERR	R	奇偶性/欠载操作错误标志(注) 0: 正常工作 1: 错误
2	FERR	R	成帧错误标志(注) 0: 正常工作 1: 错误
1	SCLKS	R/W	时钟出入缘的选择 (适用于I/O接口) 在时钟脉冲输出模式下, 设置为"0"。 0: 该传输缓冲区中的数据被发送到SCLKx下降边上的TXDx引脚一次一位。由接收缓冲器按每次一位的速度, 在SCLKx的上升缘接收通过RXDx引脚所获得的数据。在这种情况下, SCLKx从高电平开始。 1: 该传输缓冲区中的数据被发送到SCLKx上升边上的TXDx引脚一次一位。由接收缓冲器按每次一位的速度, 在SCLKx的下降缘接收通过RXDx引脚所获得的数据。在这种情况下, SCLKx从低电平开始。
0	IOC	R/W	选择时钟脉冲(适用于I/O接口) 0: 波特率发生器 1: SCLK引脚输入

注：读取时，任何错误标志 (OERR, PERR, FERR) 都会被清“0”。



14.4.5 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	类型	功能
31-8	-	R	读作"0"。
7	TB8	R/W	接收数据位8 (适用于UART) 在 9 位的UART模式下, 接收数据的第 9 位。
6	CTSE	R/W	握手功能控制(适用于UART) 0: CTS被禁用 1: CTS启用 可控制握手功能。 设置值"1"可启用握手功能(利用 CTS 引脚)
5	RXE	R/W	接收控制(注) 0: 禁用 1: 启用
4	WU	R/W	唤醒功能(适用于UART) 0: 禁用 1: 启用 该功能仅在 9-位UART模式下可用。在其它模式下, 该功能无意义。 在其处于启用状态时, "中断"仅适用于 9-位UART模式下RB 9="1"时的情况。
3-2	SM[1:0]	R/W	指定传送模式。 00: I/O接口模式 01: 7-位长度UART模式 10: 8-位长度UART模式 11: 9-位长度UART模式
1-0	SC[1:0]	R/W	串行传送时钟=(利用UART) 00: 定时器TB8OUT 01: 波特率发生器 10: 内部时钟f _{sys} 11: 外部时钟(SCLK输入) (至于I/O接口模式, 可在控制寄存器(SCxCR)中设置串行传送时钟脉冲)

注: 在<RXE> 被设置为"0"时, 可设置各模式寄存器 (SCxMOD0, SCxMOD1和SCxMOD2)。然后将<RXE> 设置为"1"。

注: 在正在接收数据时, 不要停止接收操作(通过设置SCxMOD0<RXE> = "0")。

14.4.6 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	类型	功能
31-8	-	R	读作 0。
7	I2SC	R/W	IDLE 0: 停止 1: 操作 指定IDLE模式操作。
6-5	FDPX[1:0]	R/W	传送模式设置 00: 传送被禁止 01: 半双工(接收) 10: 半双工(传输) 11: 全双工 可在I/O接口模式下配置该传送模式。此外还可用于配置FIFO(如其已启用)。在UART模式下, 其仅用于指定FIFO配置。
4	TXE	R/W	发送控制器(注2) 0: 已禁用 1: 启用 该位可启用传输, 且在所有传送模式下均有效。
3-1	SINT[2:0]	R/W	连续传输的间隔时间(适用于I/O接口) 000: 无 001: 1SCLK 010: 2SCLK 011: 4SCLK 100: 8SCLK 101: 16SCLK 110: 32SCLK 111: 64SCLK 在已选择SCLK引脚的去情况下, 该参数仅对I/O接口模式有效。在其它模式下, 该功能无意义。 在I/O接口模式已启用双缓冲或FIFO的情况下, 可指定连续传输的间隔时间。
0	-	R/W	写作 "0"。

注 1: 首先指定所有模式, 然后启用该<TXE>位。

注 2: 不要在数据传输期间停止传输操作(通过设置<TXE>="0")。

注 3: 如果SCxEN<SIOE>="0" (停止SIO操作)或操作模式被改为空闲模式(SCxMOD1<I2SC>="0") (使SIO操作在IDLE模式下停止), 则SCxTFC会被重新初始化。

14.4.7 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	1	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	传输缓冲区为空标志 0: 满 1: 空 如果双缓冲被禁用, 则该标志可忽略。 该标志表示该传输双缓冲区为空。在该传输双缓冲区中的数据被移动到传输移位寄存器, 且该双缓冲区为空时, 可将该位设置为"1"。 通过将数据重新写入到该双缓冲区, 即可将该位设置为"0"。											
6	RBFLl	R	接收缓冲器已满标志。 0: 空 1: 满 该标志表示接收双缓冲区已满。 在接收操作已完成, 且所接收的数据被从接收移位寄存器移到该接收双缓冲区时, 该位会变为"1", 而读取该位即可将该位更改为"0"。 如果双缓冲被禁用, 则该标志可忽略。											
5	TXRUN	R	在传输标志中 0: 停止 1: 操作 该状态标志表示数据传输正在进行中。 <TXRUN>与<TBEMP>位可指示以下状态。 <table border="1" style="margin-left: 20px; border-collapse: collapse; text-align: center;"> <thead> <tr> <th style="width: 15%;"><TXRUN></th> <th style="width: 15%;"><TBEMP></th> <th style="width: 70%;">状态</th> </tr> </thead> <tbody> <tr> <td>1</td> <td>-</td> <td>传输正在进行中</td> </tr> <tr> <td rowspan="2">0</td> <td>1</td> <td>传输已完成</td> </tr> <tr> <td>0</td> <td>等待状态 (数据在传输缓冲区中)</td> </tr> </tbody> </table>	<TXRUN>	<TBEMP>	状态	1	-	传输正在进行中	0	1	传输已完成	0	等待状态 (数据在传输缓冲区中)
<TXRUN>	<TBEMP>	状态												
1	-	传输正在进行中												
0	1	传输已完成												
	0	等待状态 (数据在传输缓冲区中)												
4	SBLen	R/W	停止位(适用于UART) 0: 1-位 1: 2-位 这样可在UART模式下指定传输停止位的长度。 对于接收侧而言, 仅利用与<SBLen>设置无关的单个位即可做出该判定。											
3	DRCHG	R/W	设置传送方向 0: LSB先 1: MSB先 指定I/O接口模式下的数据传送方向。											

			在UART模式下，首先将该位设置为LSB。
--	--	--	-----------------------



位	比特符号	类型	功能												
2	WBUF	R/W	双缓冲区 0: 禁用 1: 启用 该参数可允许或禁止该传输/接收双缓冲区在I/O接口模式下传输(在SCLK输出/输入模式下)与接收(在SCLK输出模式下)数据, 以及在UART模式下发送数据。 在接口模式(SCLK输入)与UART模式下接收数据时, 双缓冲在0或1被设置到为<WBUF>位时均会启用。												
1-0	SWRST[1:0]]]	R/W	软件复位 用"01"盖写"10"即可触发软件复位。在执行软件复位时, 以下各位会被初始化, 且该发送/接收电路, 该发送电路和FIFO变为初始状态(见注 1 和注 2)。 <table border="1" style="margin-left: 20px; border-collapse: collapse; width: 60%;"> <thead> <tr> <th style="width: 40%;">寄存器</th> <th style="width: 60%;">位</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">SCxMOD0</td> <td style="text-align: center;"><RXE></td> </tr> <tr> <td style="text-align: center;">SCxMOD1</td> <td style="text-align: center;"><TXE></td> </tr> <tr> <td style="text-align: center;">SCxMOD2</td> <td style="text-align: center;"><TBEMP>, <RBFL>, <TXRUN></td> </tr> <tr> <td style="text-align: center;">SCxCR</td> <td style="text-align: center;"><OERR>, <PERR>, <FERR></td> </tr> <tr> <td style="text-align: center;">SCxDMA (注 2)</td> <td style="text-align: center;"><DMAEN1>, <DMAEN0></td> </tr> </tbody> </table>	寄存器	位	SCxMOD0	<RXE>	SCxMOD1	<TXE>	SCxMOD2	<TBEMP>, <RBFL>, <TXRUN>	SCxCR	<OERR>, <PERR>, <FERR>	SCxDMA (注 2)	<DMAEN1>, <DMAEN0>
寄存器	位														
SCxMOD0	<RXE>														
SCxMOD1	<TXE>														
SCxMOD2	<TBEMP>, <RBFL>, <TXRUN>														
SCxCR	<OERR>, <PERR>, <FERR>														
SCxDMA (注 2)	<DMAEN1>, <DMAEN0>														

注 1: 在数据传输进行期间, 任何软件复位操作均必须连续执行两次。

注 2: 在进行软件复位时, 软件复位指令的识别结束与执行开始之间的持续时间必须达到 2 个时钟脉冲。

14.4.8 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)

可在以下所列的寄存器中指定波特率发生器的分频比。

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	类型	功能
31-8	-	R	读作"0"。
7	-	R/W	写入"0"。
6	BRADDE	R/W	N + (16 - K)/16 分配器功能(适用于UART) 0: 禁用 1: 启用 该分频功能仅可用于UART模式。
5-4	BRCK[1:0]	R/W	选择波特率发生器的输入时钟。 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	类型	功能
31-4	-	R	读作"0"。
3-0	BRK[3:0]	R/W	指定分频" $N + (16 - K)/16$ "的K(适用于ART) 0000: 被禁止 0001: K = 1 0010: K = 2 ... 1111: K = 15

表 14-2 列出了波特率发生器分频比的设置值。

表 14-2 分频比的设置

	<BRADDE> = "0"	<BRADDE> = "1" (注 1) (仅UART模式)
<BRS>	指定"N" (注 2) (注 3)	
<BRK>	无设置要求	指定"K" (注 4)
分频比	除以N	$N + \frac{(16 - K)}{16}$ 分频

注 1: 在使用" $N + (16 - K)/16$ "的分频功能时, 务必在将K值设置为<BRK>之后将<BRADDE> 设置为"1"。该" $N + (16 - K)/16$ "分频功能仅可用于UART模式。

注 2: 作为分频比, 在UART模式下使用 " $N + (16 - K)/16$ "分频功能时, 不能将1 ("0001")或 16 ("0000")应用于N。

注 3: 仅在双缓冲被用于I/O接口模式时, 才可指定该波特率发生器的分频比"1"。

注 4: 禁止指定"K = 0"。

14.4.9 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	保证写入"000"						
4	RFST	R/W	<p>在RX FIFO中所用的字节</p> <p>0: 最大值</p> <p>1: 同RX FIFO的充满率</p> <p>当RX FIFO启用时, 选择使用的RX FIFO字节数(注 1)</p> <p>0: 配置的FIFO 最大字节数(另见<CNFG>)。</p> <p>1: 同SCxRFC<RIL[1:0]>所指定的接收中断生成的充满率</p>						
3	TFIE	R/W	<p>TX FIFO的TX中断</p> <p>0: 禁用</p> <p>1: 启用</p> <p>当TX FIFO启用时, 发送中断由该参数启用或禁用。</p>						
2	RFIE	R/W	<p>RX FIFO的RX中断</p> <p>0: 禁用</p> <p>1: 启用</p> <p>当RX FIFO启用时, 接收中断由该参数启用或禁用。</p>						
1	RXTXCNT	R/W	<p>RXE/TXE的自动禁用</p> <p>0: 无</p> <p>1: 已自动禁用</p> <p>控制发送和接收的自动禁用。设置"1"能实现下列操作</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 30%;">半双工RX</td> <td>当接收移位寄存器, 接收缓冲区和RX FIFO被充满时, SCxMOD0<RXE>自动设置为"0", 以禁止进一步的接收。</td> </tr> <tr> <td>半双工TX</td> <td>当TX FIFO, 发送缓冲区和发送移位寄存器为空时, SCxMOD1<TXE>自动设置为"0", 以禁止进一步的发送。</td> </tr> <tr> <td>全双工</td> <td>在满足以上两个条件中的任一条件之后, TXE/RXE会被自动设置为"0", 以禁止更进一步的发送和接收。</td> </tr> </table>	半双工RX	当接收移位寄存器, 接收缓冲区和RX FIFO被充满时, SCxMOD0<RXE>自动设置为"0", 以禁止进一步的接收。	半双工TX	当TX FIFO, 发送缓冲区和发送移位寄存器为空时, SCxMOD1<TXE>自动设置为"0", 以禁止进一步的发送。	全双工	在满足以上两个条件中的任一条件之后, TXE/RXE会被自动设置为"0", 以禁止更进一步的发送和接收。
半双工RX	当接收移位寄存器, 接收缓冲区和RX FIFO被充满时, SCxMOD0<RXE>自动设置为"0", 以禁止进一步的接收。								
半双工TX	当TX FIFO, 发送缓冲区和发送移位寄存器为空时, SCxMOD1<TXE>自动设置为"0", 以禁止进一步的发送。								
全双工	在满足以上两个条件中的任一条件之后, TXE/RXE会被自动设置为"0", 以禁止更进一步的发送和接收。								
0	CNFG	R/W	<p>启用 FIFO</p> <p>0: 禁用</p> <p>1: 启用</p> <p>如已被启用, 则SCxMOD1<FDPX[1:0]> 设置会按以下所述自动配置FIFO: (在模式控制寄存器 1 SCxMOD1<FDPX[1:0]>中指定TX/RX的类型。)</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 30%;">半双工RX</td> <td>RX FIFO 4 字节</td> </tr> <tr> <td>半双工TX</td> <td>TX FIFO 4 字节</td> </tr> <tr> <td>全双工</td> <td>RX FIFO 2 字节 + TX FIFO 2 字节</td> </tr> </table>	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 2 字节
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 2 字节								

注 1: 关于TX FIFO, 配置的最大字节数总是可用的。可用字节数指已写入TX FIFO的字节。

注 2: FIFO无法在 9位UART模式中使用。

14.4.10 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: 清除 当SCxRFC<RFCS>设置为"1"时, FIFO被清除, SCxRST<RLVL>为"000"。读指针也被初始化。															
6	RFIS	R/W	选择中断生成条件 0: 在数据达到指定的充满率时。 1: 在数据达到指定的充满率时, 或该数据在读取数据时超过指定的充满率时。															
5-2	-	R	读作 0。															
1-0	RIL[1:0]		用于生成RX中断的R/W FIFO充满率 <table border="1" style="margin-left: 20px;"> <tr> <td></td> <td>半双工</td> <td>全双工</td> </tr> <tr> <td>00</td> <td>4 字节</td> <td>2 字节</td> </tr> <tr> <td>01</td> <td>1 字节</td> <td>1 字节</td> </tr> <tr> <td>10</td> <td>2 字节</td> <td>2 字节</td> </tr> <tr> <td>11</td> <td>3 字节</td> <td>1 字节</td> </tr> </table>		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
	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: 在使用TX/RX FIFO缓冲器时, 在设置SIO传输模式(半双工/全双工)和启用FIFO (SCxFCNF<CNFG> = "1")之后, 必须清除TX/RX FIFO。

14.4.11 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TX FIFO。 在SCxTST<TFCS>被设置为"1"时, 该发送FIFO即被清除, 且SCxTST<TLVL>为"000"。写指针也被初始化。															
6	TFIS	R/W	选择中断生成条件。 0: 当数据达到指定的充满率时, 生成中断。 1: 当数据达到指定的充满率或者在新数据读取之际, 数据无法达到指定的充满率时, 生成中断。															
5-2	-	R	读作 0。															
1-0	TIL[1:0]	R/W	选择FIFO充满率。 <table border="1" style="margin-left: 40px;"> <tr> <td></td> <td>非全双工</td> <td>全双工</td> </tr> <tr> <td>00</td> <td>空</td> <td>空</td> </tr> <tr> <td>01</td> <td>1 字节</td> <td>1 字节</td> </tr> <tr> <td>10</td> <td>2 字节</td> <td>空</td> </tr> <tr> <td>11</td> <td>3 字节</td> <td>1 字节</td> </tr> </table>		非全双工	全双工	00	空	空	01	1 字节	1 字节	10	2 字节	空	11	3 字节	1 字节
	非全双工	全双工																
00	空	空																
01	1 字节	1 字节																
10	2 字节	空																
11	3 字节	1 字节																

注 1: 使用TX/RX FIFO缓冲区时, 在设置SIO传输模式 (半双工/全双工)并启用FIFO (SCxFCNF<CNFG> = "1")后, 必须清除TX/RX FIFO。

注 2: 在进行下列操作后, 再次配置SCxTFC寄存器。SCxEN<SIOE> = "0" (SIO操作停止) 条件如下:
SCxMOD1<I2SC> = "0" (禁止在中IDLE模式操作)并解除由WFI (等待中断)指令启动的低功耗模式。

14.4.12 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	类型	功能
31-8	-	R	读作 0。
7	ROR	R	RX FIFO超限操作(注) 0: 未生成 1: 生成。
6-3	-	R	读作 0。
2-0	RLVL[2:0]	R	RX FIFO充满率状态。 000: 空 001: 1 字节 010: 2 字节 011: 3 字节 100: 4 字节

注：在接收数据被从SCxBUF寄存器读出时，<ROR>位即被清除为"0"。

14.4.13 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	1	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-8	-	R	读作 0。
7	TUR	R	TX FIFO欠载操作(注) 0: 未生成 1: 生成。
6-3	-	R	读作 0。
2-0	TLVL[2:0]	R	TX FIFO充满率状态。 000: 空 001: 1 字节 010: 2 字节 011: 3 字节 100: 4 字节

注：在发送数据被写入到SCxBUF寄存器时，<TUR>位即被清除为"0"。

14.4.14 SCxDMA (DMA请求启用寄存器)

	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	-	-	-	-	-	-	DMAEN1	DMAEN0
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	DMAEN1	R/W	启用DMA请求。由接收中断INTRX生成DMA请求。 0: 禁用 1: 启用
0	DMAEN0	R/W	启用DMA请求。由接收中断INTTX生成DMA请求。 0: 禁用 1: 启用

注 1: 在DMA传输期间生成DMA请求时, 其不会得以保持并嵌套。

14.5 在各模式下的操作

表 14-3 给出了各模式和数据格式。

表 14-3 模式和数据格式

模式	模式类型	数据长度	传输方向	指定是否使用奇偶校验位	STOP位长度(发送)
模式 0	同步通信模式 (IO接口模式)	8位	LSB先/MSB先	-	-
模式 1	异步通信模式 UART模式	7位	LSB先	o	1 位或 2 位
模式 2		8位		o	
模式 3		9位		x	

模式 0 为同步通信，可用于扩展I/O。该模式可与SCLK同步发射并接收数据。SCLK既能用于输入，也能用于输出。

可从LSB先和MSB先中选择数据传输方向。该模式不得使用奇偶校验位或STOP位。

模式 1，模式 2 和模式 3 为异步模式，传输方向固定为LSB先。

奇偶校验位能在模式 1 和模式 2 时添加。模式 3 具有唤醒功能，在该功能中，主控制器经串行链路启动从机控制器(多控制器系统)。

可从 1 位和 2 位中选择传输中的STOP位。接收中的STOP位长度固定为一位。

14.6 数据格式

14.6.1 数据格式列表

图 14-2 给出了数据格式。

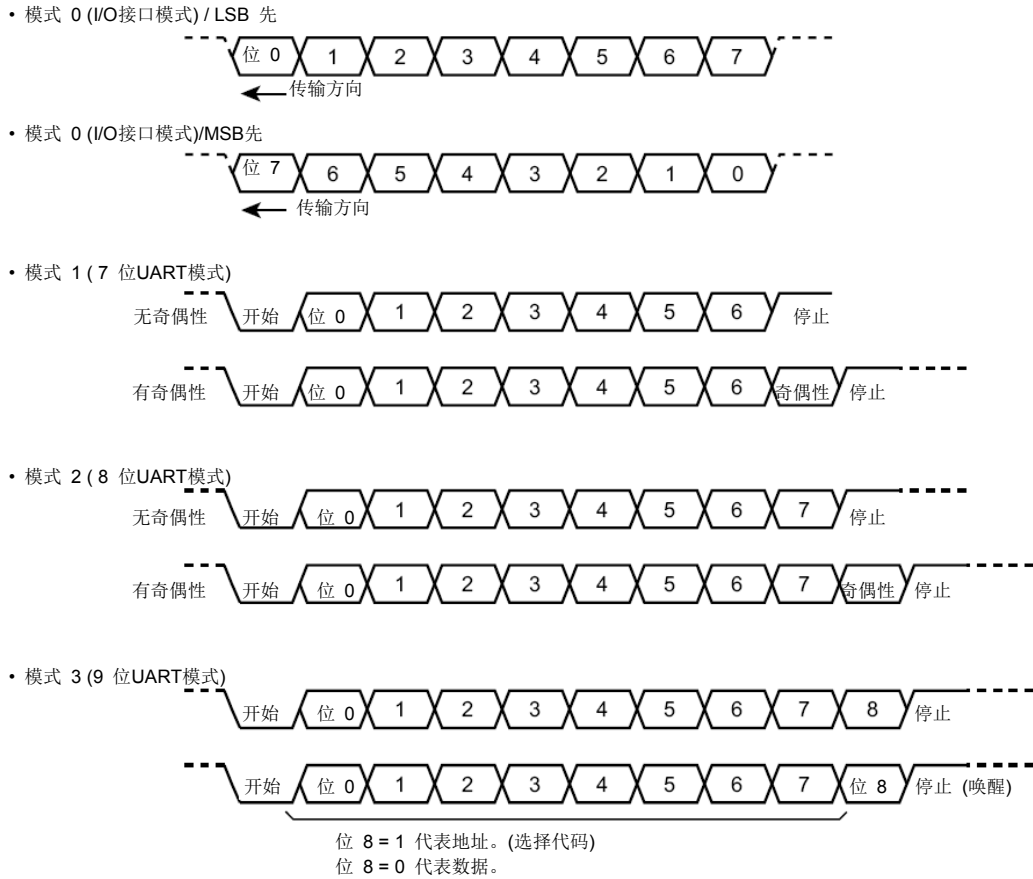


图 14-2 数据格式

14.6.2 奇偶控制

仅可在 7-或 8-位UART模式下添加该奇偶校验位。

将SCxCR<PE>设置为"1"就可启用奇偶性。

SCxCR的<EVEN>位选择偶数奇偶性或奇数奇偶性。

14.6.2.1 发送

在数据传输后，奇偶控制电路自动生成奇偶性，数据位于发送缓冲区。

在数据传输完成之后，奇偶校验位会被存储到SCxBUF<TB7>(适用于7-位UART模式)或SCxMOD<TB8>(适用于8-位UART模式)中。

<PE>和<EVEN>的设置必须在数据写入发送缓冲区之前完成。

14.6.2.2 接收数据

若接收的数据从接收移位寄存器移到接收缓冲区，则会生成奇偶性。

在 7-位UART模式时，生成的奇偶性与SCxBUF<RB7>中储存的奇偶性比较，而在8-位UART模式时，它与SCxCR<RB8>中储存的奇偶性比较。

若有任何差异，就会发生奇偶校验错误，SCxCR寄存器中的<PERR>设置为"1"。

在采用FIFO时，<RERR>会指示在其中一个数据内部生成了一个奇偶校验误差。

14.6.3 停止位长度

通过设置SCxMOD2<SBLEN>，可从一位或两位中选择UART传输模式时的STOP位长度。不管该位的设置，STOP位数据的长度在它被接收时被确定为一位。

14.7 时钟控制

14.7.1 预分频器

有一个把预分频器输入时钟 $\Phi T0$ 除以 2, 8, 32 和 128 的 7-位预分频器。

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 $\Phi T0$ 。

只有当波特率发生器由SCxMOD0<SC[1:0]> = "01"选为传输时钟时, 预分频器才能被激活。

下表给出了波特率发生器输入时钟的图形分辨率。

表 14-4 波特率发生器时钟图形分辨率 $fc = 40\text{ MHz}$

外围时钟选择 CGSYSCR <FPSEL>	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			
			$\phi T1$	$\phi T4$	$\phi T16$	$\phi T64$
0 (fgear)	000 (fc)	000 (fperiph/1)	$fc/2^1$ (0.05 μs)	$fc/2^3$ (0.2 μs)	$fc/2^5$ (0.8 μs)	$fc/2^7$ (3.2 μs)
		001 (fperiph/2)	$fc/2^2$ (0.1 μs)	$fc/2^4$ (0.4 μs)	$fc/2^6$ (1.6 μs)	$fc/2^8$ (6.4 μs)
		010 (fperiph/4)	$fc/2^3$ (0.2 μs)	$fc/2^5$ (0.8 μs)	$fc/2^7$ (3.2 μs)	$fc/2^9$ (12.8 μs)
		011 (fperiph/8)	$fc/2^4$ (0.4 μs)	$fc/2^6$ (1.6 μs)	$fc/2^8$ (6.4 μs)	$fc/2^{10}$ (25.6 μs)
		100 (fperiph/16)	$fc/2^5$ (0.8 μs)	$fc/2^7$ (3.2 μs)	$fc/2^9$ (12.8 μs)	$fc/2^{11}$ (51.2 μs)
		101 (fperiph/32)	$fc/2^6$ (1.6 μs)	$fc/2^8$ (6.4 μs)	$fc/2^{10}$ (25.6 μs)	$fc/2^{12}$ (102.4 μs)
	100 (fc/2)	000 (fperiph/1)	$fc/2^2$ (0.1 μs)	$fc/2^4$ (0.4 μs)	$fc/2^6$ (1.6 μs)	$fc/2^8$ (6.4 μs)
		001 (fperiph/2)	$fc/2^3$ (0.2 μs)	$fc/2^5$ (0.8 μs)	$fc/2^7$ (3.2 μs)	$fc/2^9$ (12.8 μs)
		010 (fperiph/4)	$fc/2^4$ (0.4 μs)	$fc/2^6$ (1.6 μs)	$fc/2^8$ (6.4 μs)	$fc/2^{10}$ (25.6 μs)
		011 (fperiph/8)	$fc/2^5$ (0.8 μs)	$fc/2^7$ (3.2 μs)	$fc/2^9$ (12.8 μs)	$fc/2^{11}$ (51.2 μs)
		100 (fperiph/16)	$fc/2^6$ (1.6 μs)	$fc/2^8$ (6.4 μs)	$fc/2^{10}$ (25.6 μs)	$fc/2^{12}$ (102.4 μs)
		101 (fperiph/32)	$fc/2^7$ (3.2 μs)	$fc/2^9$ (12.8 μs)	$fc/2^{11}$ (51.2 μs)	$fc/2^{13}$ (204.8 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)	$fc/2^9$ (12.8 μs)
		001 (fperiph/2)	$fc/2^4$ (0.4 μs)	$fc/2^6$ (1.6 μs)	$fc/2^8$ (6.4 μs)	$fc/2^{10}$ (25.6 μs)
		010 (fperiph/4)	$fc/2^5$ (0.8 μs)	$fc/2^7$ (3.2 μs)	$fc/2^9$ (12.8 μs)	$fc/2^{11}$ (51.2 μs)
		011 (fperiph/8)	$fc/2^6$ (1.6 μs)	$fc/2^8$ (6.4 μs)	$fc/2^{10}$ (25.6 μs)	$fc/2^{12}$ (102.4 μs)
		100 (fperiph/16)	$fc/2^7$ (3.2 μs)	$fc/2^9$ (12.8 μs)	$fc/2^{11}$ (51.2 μs)	$fc/2^{13}$ (204.8 μs)
		101 (fperiph/32)	$fc/2^8$ (6.4 μs)	$fc/2^{10}$ (25.6 μs)	$fc/2^{12}$ (102.4 μs)	$fc/2^{14}$ (409.6 μs)
	110 (fc/8)	000 (fperiph/1)	$fc/2^4$ (0.4 μs)	$fc/2^6$ (1.6 μs)	$fc/2^8$ (6.4 μs)	$fc/2^{10}$ (25.6 μs)
		001 (fperiph/2)	$fc/2^5$ (0.8 μs)	$fc/2^7$ (3.2 μs)	$fc/2^9$ (12.8 μs)	$fc/2^{11}$ (51.2 μs)
		010 (fperiph/4)	$fc/2^6$ (1.6 μs)	$fc/2^8$ (6.4 μs)	$fc/2^{10}$ (25.6 μs)	$fc/2^{12}$ (102.4 μs)
		011 (fperiph/8)	$fc/2^7$ (3.2 μs)	$fc/2^9$ (12.8 μs)	$fc/2^{11}$ (51.2 μs)	$fc/2^{13}$ (204.8 μs)
		100 (fperiph/16)	$fc/2^8$ (6.4 μs)	$fc/2^{10}$ (25.6 μs)	$fc/2^{12}$ (102.4 μs)	$fc/2^{14}$ (409.6 μs)
		101 (fperiph/32)	$fc/2^9$ (12.8 μs)	$fc/2^{11}$ (51.2 μs)	$fc/2^{13}$ (204.8 μs)	$fc/2^{15}$ (819.2 μs)
111 (fc/16)	000 (fperiph/1)	$fc/2^5$ (0.8 μs)	$fc/2^7$ (3.2 μs)	$fc/2^9$ (12.8 μs)	$fc/2^{11}$ (51.2 μs)	
	001 (fperiph/2)	$fc/2^6$ (1.6 μs)	$fc/2^8$ (6.4 μs)	$fc/2^{10}$ (25.6 μs)	$fc/2^{12}$ (102.4 μs)	
	010 (fperiph/4)	$fc/2^7$ (3.2 μs)	$fc/2^9$ (12.8 μs)	$fc/2^{11}$ (51.2 μs)	$fc/2^{13}$ (204.8 μs)	
	011 (fperiph/8)	$fc/2^8$ (6.4 μs)	$fc/2^{10}$ (25.6 μs)	$fc/2^{12}$ (102.4 μs)	$fc/2^{14}$ (409.6 μs)	
	100 (fperiph/16)	$fc/2^9$ (12.8 μs)	$fc/2^{11}$ (51.2 μs)	$fc/2^{13}$ (204.8 μs)	$fc/2^{15}$ (819.2 μs)	
	101 (fperiph/32)	$fc/2^{10}$ (25.6 μs)	$fc/2^{12}$ (102.4 μs)	$fc/2^{14}$ (409.6 μs)	$fc/2^{16}$ (1638 μs)	

表 14-4 波特率发生器时钟图形分辨率 $f_c = 40 \text{ MHz}$

外围时钟选择 CGSYSCR <FPSEL>	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			
			$\phi T1$	$\phi T4$	$\phi T16$	$\phi T64$
1 (fc)	000 (fc)	000 (fperiph/1)	$fc/2^1$ (0.05 μs)	$fc/2^3$ (0.2 μs)	$fc/2^5$ (0.8 μs)	$fc/2^7$ (3.2 μs)
		001 (fperiph/2)	$fc/2^2$ (0.1 μs)	$fc/2^4$ (0.4 μs)	$fc/2^6$ (1.6 μs)	$fc/2^8$ (6.4 μs)
		010 (fperiph/4)	$fc/2^3$ (0.2 μs)	$fc/2^5$ (0.8 μs)	$fc/2^7$ (3.2 μs)	$fc/2^9$ (12.8 μs)
		011 (fperiph/8)	$fc/2^4$ (0.4 μs)	$fc/2^6$ (1.6 μs)	$fc/2^8$ (6.4 μs)	$fc/2^{10}$ (25.6 μs)
		100 (fperiph/16)	$fc/2^5$ (0.8 μs)	$fc/2^7$ (3.2 μs)	$fc/2^9$ (12.8 μs)	$fc/2^{11}$ (51.2 μs)
		101 (fperiph/32)	$fc/2^6$ (1.6 μs)	$fc/2^8$ (6.4 μs)	$fc/2^{10}$ (25.6 μs)	$fc/2^{12}$ (102.4 μs)
	100 (fc/2)	000 (fperiph/1)	-	$fc/2^3$ (0.2 μs)	$fc/2^5$ (0.8 μs)	$fc/2^7$ (3.2 μs)
		001 (fperiph/2)	$fc/2^2$ (0.1 μs)	$fc/2^4$ (0.4 μs)	$fc/2^6$ (1.6 μs)	$fc/2^8$ (6.4 μs)
		010 (fperiph/4)	$fc/2^3$ (0.2 μs)	$fc/2^5$ (0.8 μs)	$fc/2^7$ (3.2 μs)	$fc/2^9$ (12.8 μs)
		011 (fperiph/8)	$fc/2^4$ (0.4 μs)	$fc/2^6$ (1.6 μs)	$fc/2^8$ (6.4 μs)	$fc/2^{10}$ (25.6 μs)
		100 (fperiph/16)	$fc/2^5$ (0.8 μs)	$fc/2^7$ (3.2 μs)	$fc/2^9$ (12.8 μs)	$fc/2^{11}$ (51.2 μs)
		101 (fperiph/32)	$fc/2^6$ (1.6 μs)	$fc/2^8$ (6.4 μs)	$fc/2^{10}$ (25.6 μs)	$fc/2^{12}$ (102.4 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)	-	$fc/2^4$ (0.4 μs)	$fc/2^6$ (1.6 μs)	$fc/2^8$ (6.4 μs)
		010 (fperiph/4)	$fc/2^3$ (0.2 μs)	$fc/2^5$ (0.8 μs)	$fc/2^7$ (3.2 μs)	$fc/2^9$ (12.8 μs)
		011 (fperiph/8)	$fc/2^4$ (0.4 μs)	$fc/2^6$ (1.6 μs)	$fc/2^8$ (6.4 μs)	$fc/2^{10}$ (25.6 μs)
		100 (fperiph/16)	$fc/2^5$ (0.8 μs)	$fc/2^7$ (3.2 μs)	$fc/2^9$ (12.8 μs)	$fc/2^{11}$ (51.2 μs)
		101 (fperiph/32)	$fc/2^6$ (1.6 μs)	$fc/2^8$ (6.4 μs)	$fc/2^{10}$ (25.6 μs)	$fc/2^{12}$ (102.4 μs)
	110 (fc/8)	000 (fperiph/1)	-	-	$fc/2^5$ (0.8 μs)	$fc/2^7$ (3.2 μs)
		001 (fperiph/2)	-	$fc/2^4$ (0.4 μs)	$fc/2^6$ (1.6 μs)	$fc/2^8$ (6.4 μs)
		010 (fperiph/4)	-	$fc/2^5$ (0.8 μs)	$fc/2^7$ (3.2 μs)	$fc/2^9$ (12.8 μs)
		011 (fperiph/8)	$fc/2^4$ (0.4 μs)	$fc/2^6$ (1.6 μs)	$fc/2^8$ (6.4 μs)	$fc/2^{10}$ (25.6 μs)
		100 (fperiph/16)	$fc/2^5$ (0.8 μs)	$fc/2^7$ (3.2 μs)	$fc/2^9$ (12.8 μs)	$fc/2^{11}$ (51.2 μs)
		101 (fperiph/32)	$fc/2^6$ (1.6 μs)	$fc/2^8$ (6.4 μs)	$fc/2^{10}$ (25.6 μs)	$fc/2^{12}$ (102.4 μs)
111 (fc/16)	000 (fperiph/1)	-	-	$fc/2^5$ (0.8 μs)	$fc/2^7$ (3.2 μs)	
	001 (fperiph/2)	-	-	$fc/2^6$ (1.6 μs)	$fc/2^8$ (6.4 μs)	
	010 (fperiph/4)	-	$fc/2^5$ (0.8 μs)	$fc/2^7$ (3.2 μs)	$fc/2^9$ (12.8 μs)	
	011 (fperiph/8)	-	$fc/2^6$ (1.6 μs)	$fc/2^8$ (6.4 μs)	$fc/2^{10}$ (25.6 μs)	
	100 (fperiph/16)	$fc/2^5$ (0.8 μs)	$fc/2^7$ (3.2 μs)	$fc/2^9$ (12.8 μs)	$fc/2^{11}$ (51.2 μs)	
	101 (fperiph/32)	$fc/2^6$ (1.6 μs)	$fc/2^8$ (6.4 μs)	$fc/2^{10}$ (25.6 μs)	$fc/2^{12}$ (102.4 μs)	

注 1: 必须选择该预分频器输出时钟 ϕTn , 以确保满足关系式" $\phi Tn \leq f_{\text{sys}} / 2$ "的要求(使得 ϕTn 慢于 $f_{\text{sys}} / 2$)。

注 2: 当SIO正操作时, 不得改变时钟齿轮。

注 3: 上表中的破折号表示设置被禁止。

表 14-5 波特率发生器时钟图形分辨率 $f_c = 48 \text{ MHz}$

外围时钟选择 CGSYSCR <FPSEL>	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			
			$\phi T1$	$\phi T4$	$\phi T16$	$\phi T64$
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)	$fc/2^{12}$ (85.3 μs)	$fc/2^{14}$ (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)	$fc/2^{12}$ (85.3 μs)	$fc/2^{14}$ (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)
111 (fc/16)	000 (fperiph/1)	$fc/2^5$ (0.667 μs)	$fc/2^7$ (2.67 μs)	$fc/2^9$ (10.7 μs)	$fc/2^{11}$ (42.7 μs)	
	001 (fperiph/2)	$fc/2^6$ (1.33 μs)	$fc/2^8$ (5.33 μs)	$fc/2^{10}$ (21.3 μs)	$fc/2^{12}$ (85.3 μs)	
	010 (fperiph/4)	$fc/2^7$ (2.67 μs)	$fc/2^9$ (10.7 μs)	$fc/2^{11}$ (42.7 μs)	$fc/2^{13}$ (171 μs)	
	011 (fperiph/8)	$fc/2^8$ (5.33 μs)	$fc/2^{10}$ (21.3 μs)	$fc/2^{12}$ (85.3 μs)	$fc/2^{14}$ (341 μs)	
	100 (fperiph/16)	$fc/2^9$ (10.7 μs)	$fc/2^{11}$ (42.7 μs)	$fc/2^{13}$ (171 μs)	$fc/2^{15}$ (683 μs)	
	101 (fperiph/32)	$fc/2^{10}$ (21.4 μs)	$fc/2^{12}$ (85.4 μs)	$fc/2^{14}$ (342 μs)	$fc/2^{16}$ (1366 μs)	

表 14-5 波特率发生器时钟图形分辨率 $f_c = 48 \text{ MHz}$

外围时钟选择 CGSYSCR <FPSEL>	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			
			$\phi T1$	$\phi T4$	$\phi T16$	$\phi T64$
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)	$fc/2^7$ (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)
	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)
111 (fc/16)	000 (fperiph/1)	-	-	$fc/2^5$ (0.667 μs)	$fc/2^7$ (2.67 μs)	
	001 (fperiph/2)	-	-	$fc/2^6$ (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)	-	$fc/2^6$ (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)	

注 1: 必须选择该预分频器输出时钟 ϕTn , 以确保满足关系式" $\phi Tn \leq f_{\text{sys}} / 2$ "的要求(使得 ϕTn 慢于 $f_{\text{sys}} / 2$)。

注 2: 当SIO正操作时, 不得改变时钟齿轮。

注 3: 上表中的破折号表示设置被禁止。

14.7.2 串行时钟生成电路

串行时钟电路是一个发送和接收时钟(SIOCLK)发生模块，由通过设置波特率发生器和模式就可选择时钟的电路组成。

14.7.2.1 波特率发生器

波特率发生器生成发送和接收时钟，以确定串行通道传输率。

(1) 波特率发生器输入时钟

将预分频器输出除以2, 8, 32 和 128, 从所得结果中选择波特率发生器输入时钟。

通过设置SCxBRCR<BRCK>, 便可选择该输入时钟。

(2) 波特率发生器输出时钟

波特率发生器中的输出时钟的分频比由SCxBRCR和SCxBRADD设置。

可采用下列分频比；在 I/O 接口模式时 1/N 分频，在UART模式时 1/N 或 $N + (16 - K) / 16$ 。

能选择的分频比如下表所示。

模式	分频功能设置 SCxBRCR<BRADDE>	由 N分频 SCxBRCR<BRS>	由 K分频 SCxBRADD<BRK>
I/O接口	除以N	1 ~ 16 (注)	-
UART	除以N	1 ~ 16	-
	$N + (16 - K) / 16$ 分频	2 ~ 15	1 ~ 15

注：仅可在双缓冲器已启用时采用1/N (N=1)分频比。

14.7.2.2 时钟选择电路

通过设置模式和寄存器选择时钟。

通过设置SCxMOD0<SM>指定模式。

通过设置SCxCR选择在I/O接口模式时的输入时钟。通过设置SCxMOD0<SC>，便可选择在UART模式时的时钟。

(1) I/O 接口模式下的传输时钟

表 14-6 给出了I/O接口模式下的时钟选择。

表 14-6 I/O接口模式下的时钟选择

模式 SCxMOD0<SM>	输入/输出选择 SCxCR<IOC>	时钟缘选择 SCxCR<SCLKS>	使用的时钟
I/O接口模式	SCLK输出	设置为"0"。 (固定为上升缘)	波特率发生器输出除以 2。
	SCLK输入	上升缘	SCLK输入上升缘
		下降缘	SCLK输入下降缘

为了获得最高波特率，波特率发生器必须设置如下。

注：在确定时钟设置时，应去确认交流电气特性满足要求。

时钟/模式控制块设置

- fc = 40 MHz
- fgear = 40 MHz (CGSYSCR<GEAR[2:0]> = "000" :fc被选中)
- φT0 = 40 MHz (CGSYSCR<PRCK[2:0]> = "000" :1 分频比)

SIO设置 (如使用双缓冲器)

- 时钟(SCxBRCR<BRCK[1:0]> = "00" :φT1 已被选择) = 20MHz
- 已分频的时钟脉冲频率(SCxBRCR<BRS[3:0]> = "0001" :1 分频比) = 20MHz

若采用双缓冲器，则可选择 1 分频比。在这种情况下，因为 20 MHz除以 2，所以波特率为 10 Mbps。

SIO设置 (如未使用双缓冲器)

- 时钟(SCxBRCR<BRCK[1:0]> = "00" :φT1 已被选择) = 20MHz
- 已分频的时钟脉冲频率(SCxBRCR<BRS[3:0]> = "0010" :2 分频比) = 10 MHz

若未采用双缓冲器，2 分频比为最高值。在这种情况下，10 MHz被除以 2，因此波特率为 5 Mbps。

为了采用SCLK输入，必须满足下列条件。

如使用双缓冲器

- SCLK周期 > 6 / fsys

最高波特率小于 $48 \div 6 = 8$ Mbps。

如未使用双缓冲器

- SCLK周期 > 8 / fsys

最高波特率小于 $48 \div 8 = 6$ Mbps。

(2) UART模式下的传输时钟

表 14-7 给出了UART模式下的时钟选择。UART模式下，使用前在接收计数器或发送计数器中将所选时钟除以 16。

表 14-7 UART模式下的时钟选择

模式 SCxMOD0<SM>	时钟选择 SCxMOD0<SC>
UART模式	定时器输出
	波特率发生器
	fsys
	SCLK输入

各时钟设置中波特率举例。

如使用波特率发生器

- fc = 40 MHz
 - fgear = 40 MHz (CGSYSCR<GEAR[2:0]> = "000" :fc被选中)
 - $\phi T0 = 40$ MHz (CGSYSCR<PRCK[2:0]> = "000" :1 分频比)
 - 时钟 = $\Phi T1 = 20$ MHz (SCxBRCR<BRCK[1:0]> = "00" : $\phi T1$ 被选中)
- 20 MHz被除以 16，因此最高波特率为1.25 Mbps。

表 14-8 给出了波特率发生器采用以下时钟设置时的波特率示例。

fc = 9.8304 MHz

fgear = 9.8304 MHz (CGSYSCR<GEAR[2:0]> = "000" : fc被选中)

$\phi T0 = 4.9152$ MHz (CGSYSCR<PRCK[2:0]> = "001" : 2 分频比)

表 14-8 UART模式波特率 (采用波特率发生器)

fc [MHz]	分频比N (SCxBRCR<BRS>)	$\phi T1$ (fc/4)	$\phi T4$ (fc/16)	$\phi T16$ (fc/64)	$\phi T64$ (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

如果采用SCLK输入

为了采用SCLK输入，必须满足下列条件。

- SCLK周期 > 2 / fsys

最高波特率必须小于 $48 \div 2 \div 16 = 1.5$ Mbps。

如使用f_{sys}

由于f_{sys}的最大值为48MHz，因此最高波特率为 $48 \div 16 = 3\text{Mbps}$ 。

如使用定时器输出

启用定时器输出时，必须设置下列条件：当计数器的值与TBxRG1的值匹配时，定时器触发器的输出反向。SIOCLK时钟频率为"TBxRG1设置值 × 2"。

波特率由下述公式计算所得。

波特率的计算

$$\text{传输率} = \frac{\text{由CGSYSCR<PRCK[1:0]>选择的时钟脉冲频率}}{(\text{TBxRG1} \times 2) \times 2 \times 16}$$

↑ 一个时钟周期为定时器触发器反向两次的时间。

↑ 如选择了定时器预分频器时钟φT1 (2 分频比)。

表 14-9 给出了定时器输出采用以下时钟设置时的波特率示例。

$$f_c = 32 \text{ MHz} / 9.8304 \text{ MHz} / 8 \text{ MHz}$$

$$f_{\text{gear}} = 32 \text{ MHz} / 9.8304 \text{ MHz} / 8 \text{ MHz} \text{ (CGSYSCR<GEAR[2:0]> = "000" : } f_c \text{ 被选中)}$$

$$\phi T_0 = 16 \text{ MHz} / 4.9152 \text{ MHz} / 4 \text{ MHz} \text{ (CGSYSCR<PRCK[2:0]> = "001" : 2 分频比)}$$

$$\text{定时器 计数 clock} = 4 \text{ MHz} / 1.2287 \text{ MHz} / 1 \text{ MHz} \text{ (TBxMOD<TBCLK[1:0]> = "01" : } \phi T_1 \text{ 被选中)}$$

表 14-9 UART模式波特率示例(采用定时器输出)

TBxRG0的设置	f _c		
	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

14.8 发送/接收缓冲器和FIFO

14.8.1 配置

图 14-3 给出了发送缓冲器，接收缓冲器和FIFO的配置。

使用缓冲器和FIFO，必须进行适当的设置。可按照模式对配置进行预定义。

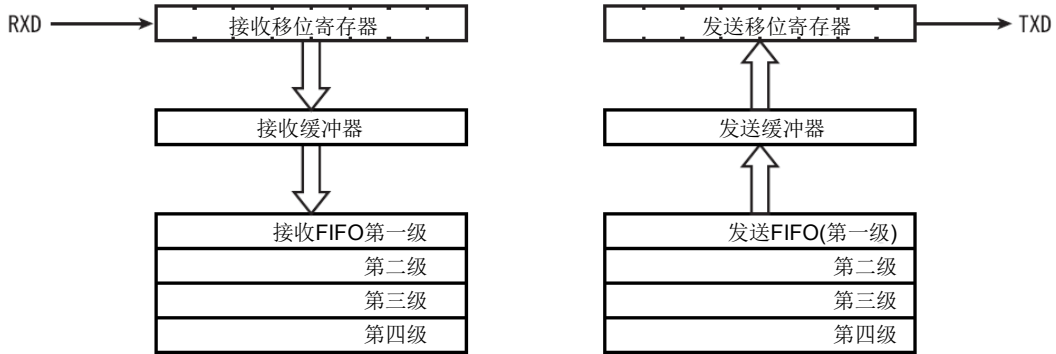


图 14-3 缓冲器和FIFO的配置

14.8.2 发送/接收缓冲器

对发送缓冲器和接收缓冲器进行双缓冲。缓冲器配置由SCxMOD2<WBUF>指定。

在使用接收缓冲器的情况下，若在I/O 接口模式时为了生成时钟输出而设置了SCLK输入，或者若选择了UART模式，则尽管有 <WBUF> 设置，仍会对其进行双缓冲。在其他模式时，它按照 <WBUF>的设置。

表 14-10 给出了各模式和缓冲器之间的相关性。

表 14-10 模式和缓冲器组成

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

14.8.3 FIFO

除上述双缓冲功能外，还能使用 4 字节FIFO。

为了启用FIFO，应通过将SCxMOD2<WBUF>设置为"1"及将SCxFCNF<CNFG>设置为"1"，启用双缓冲器。FIFO缓冲器配置由SCxMOD1<FDPX[1:0]>指定。

注：在使用TX/RX FIFO缓冲器时，在设置SIO传输模式(半双工/全双工)和启用FIFO (SCxFCNF<CNFG> = "1")之后，必须清除TX/RX FIFO。

表 14-11 给出了各模式和FIFO之间的相关性。

表 14-11 模式和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

14.9 状态标志

SCxMOD2寄存器有两类标志。只有在双缓冲器启用时，该位才有效。

<RBFLL>是一个显示接收缓冲器已满的标志。当收到一帧数据，并且数据从接收移位寄存器移到接收缓冲器时，该位变为"1"，而读取该位会使其变为"0"。

<TBEMP>表示发送缓冲器为空。当发送缓冲器中的数据被移到发送移位寄存器时，该位被设置为"1"。当数据被设置到发送缓冲器时，该位即被清"0"。

14.10 错误标志

在SCxCR寄存器中设有三个错误标志。标志的含义随模式而变化。在各模式时的含义如下表所示。

在读取SCxCR寄存器后，这些标志被清除到"0"。

模式	标志		
	<OERR>	<PERR>	<FERR>
UART	超限操作错误	奇偶错误	成帧错误
I/O接口 (SCLK输入)	超限操作错误	欠载操作错误 (当使用双缓冲器或FIFO时)	固定为 0
		固定为0 (当未使用双缓冲器和FIFO时)	
I/O接口 (SCLK输出)	未定义	未定义	固定为 0

14.10.1 OERR标志

UART和I/O接口模式下，在完成读取接收缓冲器前，当通过完成下一帧接收数据的接收而出错时，该位设置为"1"。启用接收FIFO时，接收的数据自动移至接收FIFO，不会产生超限操作错误，直到接收FIFO已满(或者直到可用的字节被完全占空)。

带SCLK输出的I/O接口模式下，SCLK输出在设置标志后停止。

注：在将I/O接口SCLK输出模式切换为其它模式时，需读取SCxCR寄存器并清除该溢出标志。

14.10.2 PERR标志

UART模式时，该标志表示奇偶校验错误；I/O接口模式下，该标志表示欠载操作错误。

UART模式下，当接收的数据产生的奇偶性不同于接收的奇偶性时，将<PERR>设置为"1"。

I/O接口模式下，启用当双缓冲器，<PERR>在下列条件下设置为"1"。

SCLK输入模式下，在完成发送移位寄存器的数据输出，并且在发送缓冲器中无数据后，当输入SCLK时，将<PERR>设置为"1"。

在SCLK输出模式下，在完成所有数据的输出后，将<PERR>设置为"1"，SCLK输出停止。

注：在将I/O接口SCLK输出模式切换为其它模式时，需读取SCxCR寄存器并清除该欠载操作标志。

14.10.3 FERR标志

若在中心周围对相应的停止位取样，该位被确定为"0"，则会产生成帧错误。不管SCxMOD2<SBLEN>寄存器中的停止位长度设置，停止位状态仅由1确定。

I/O接口模式下，该位固定至"0"。

14.11 接收

14.11.1 接收计数器

接收计数器为 4-位二进制计数器，并由 SIOCLK 向上计数。在 UART 模式时，16 个 SIOCLK 时钟脉冲用于接收单一数据位，在第七，第八和第九个脉冲对数据符号取样。从这三个样本中，多数逻辑用于决定接收的数据。

14.11.2 接收控制装置

14.11.2.1 I/O接口模式

在 SCLK 输出模式下且 SCxCR <IOC> 设置为 "0" 时，在向 SCLK 引脚输出的移位时钟的上升缘，对 RXD 引脚取样。

在 SCLK 输入模式下且 SCxCR <IOC> 设置为 "1" 时，按照 SCxCR <SCLKS> 的设置，在 SCLK 输入信号的上升缘或下降缘对串行接收数据 RXD 引脚取样。

14.11.2.2 UART 模式

接收控制器有一个起始位检测电路，该电路用于在检测到正常起始位时启动接收操作。

14.11.3 接收操作

14.11.3.1 接收缓冲器

接收的数据按 1 位储存在接收移位寄存器中。当储存一整套位元完成后，会产生中断 INTRx。

启用双缓冲器时，数据移到接收缓冲器 (SCxBUF)，接收缓冲器全满标志 (SCxMOD2<RBFL>) 设置为 "1"。通过读取接收缓冲器，接收缓冲器全满标志被清除到 "0"。

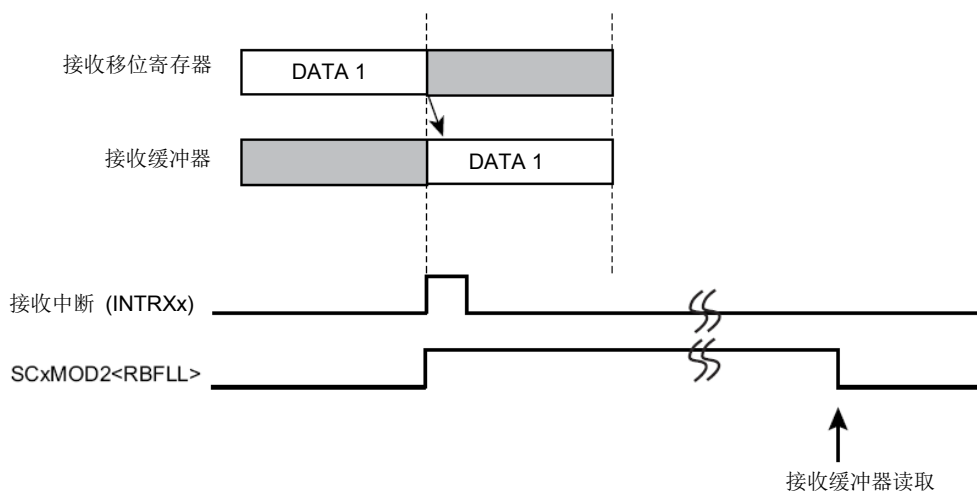


图 14-4 接收缓冲器操作

14.11.3.2 接收FIFO操作

启用FIFO时，接收的数据从接收缓冲器移到接收FIFO，接收缓冲器全满标志被立即清除。按照SCxRFC<RIL>的设置，会产生中断。

注：在UART模式下利用FIFO接收数据连同奇偶校验位时，奇偶错误标记会显示所接收数据中的奇偶校验误差。

半双工 RX 模式下的配置和操作说明如下。

- SCxMOD1[6:5] = 01 :传输模式设置为半双工模式
- SCxFCNF[4:0] = 10111 :在达到充满率后，自动禁止连续接收。
:在接收FIFO中使用的字节数与中断生成充满率相同。
- SCxRFC[1:0] = 00 :生成的接收中断设置为 4 字节的FIFO的充满率。
- SCxRFC[7:6] = 11 :清除接收FIFO，并设置中断生成的条件。

在设置上述FIFO配置后，将"1"写入SCxMOD0 <RXE>，可启动数据接收。当数据均被储存在接收移位寄存器，接收缓冲器和接收FIFO时，SCxMOD0<RXE>被自动清除，完成接收操作。

在上述条件下，若在达到充满率后启用连续接收，并且有可能连续接收数据，而数据位于FIFO，并读取FIFO中的数据。

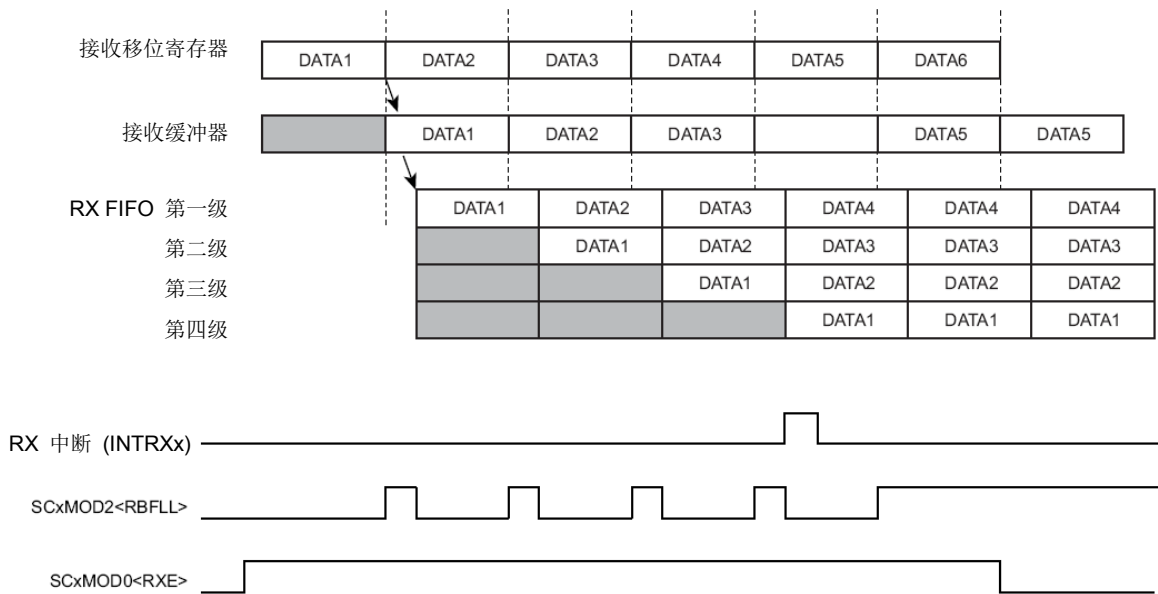


图 14-5 接收FIFO操作

14.11.3.3 I/O接口模式具备SCLK输出

在I/O接口模式和SCLK输出设置中,当所有接收的数据储存在接收缓冲器和FIFO时, SCLK输出停止。因此,此模式下,超限操作错误标志无意义。

SCLK输出停止和再输出的时间取决于接收缓冲器和FIFO。

(1) 单缓冲器的情况

在接收数据后,停止SCLK输出。此模式下,I/O接口通过握手可用传输装置传输各数据。

当读取缓冲器中的数据时,SCLK输出被重启。

(2) 双缓冲器的情况

在数据被接收至接收移位寄存器和接收缓冲器后,停止SCLK输出。

当读取数据时,SCLK输出被重启。

(3) FIFO的情况

在数据被接收至移位寄存器,接收缓冲器和FIFO后,停止SCLK输出。

当读取一字节数据时,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被传输到接收缓冲器,SCLK输出被重启。

若SCxFCNF<RXTXCNT>设置为"1",SCLK停止,接收操作也停止,并清除SCxMOD0<RXE>位。

14.11.3.4 读取已接收数据

尽管启用或禁用FIFO,仍会读取从接收缓冲器(SCxBUF)接收的数据。

当接收FIFO禁用时,通过该次读取,缓冲器全满标志SCxMOD2<RBFLL>被清除到"0"。在这种情况下,在读取接收缓冲器的数据前,在接收移位寄存器中能接收下一数据。在8位UART模式下要添加的奇偶校验位及在9位UART模式时最有效的位,将储存在SCxCR<RB8>中。

当接收FIFO可用时,由于8位数据能储存在FIFO中,9位UART模式禁用。8位UART模式下,虽然奇偶校验位丢失,但是奇偶校验错误得到确定,并且结果被储存在SCxCR<PERR>中。

14.11.3.5 唤醒功能

在9位UART模式下,通过将唤醒功能SCxMOD0<WU>设置为"1",就可以在唤醒模式下操作该从属控制器。在这种情况下,只有当SCxCR<RB8>设置为"1"时,才会产生INTRXx。

14.11.3.6 溢出错误

当FIFO禁用时，发生超限操作错误，在接收下一数据前，在没有完成数据读取的情况下设置超限操作标志。当发生超限操作错误时，虽然接收缓冲器和SCxCR<RB8>的内容未丢失，但是接收移位寄存器的内容丢失。

启用FIFO，发生超限操作错误，在FIFO已满而将下一数据移入接收缓冲器前，通过不读取数据而设置超限操作标志。这样FIFO的内容不会丢失。

带SCLK输出设置的I/O接口模式下，时钟输出自动停止，因此该标志无意义。

注：在将该I/O接口SCLK输出模式切换为其它模式时，需读取SCxCR寄存器并清除该溢出标志。

14.12 发送

14.12.1 发送计数器

传输计数器为 4-位二进制计数器，和在接收计数器的情况一样，由SIOCLK计数。UART 模式下，它在每第16个时钟脉冲时生成发送时钟(TXDCLK)。

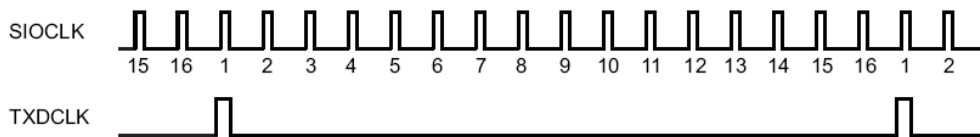


图 14-6 发送时钟的生成

14.12.2 发送控制

14.12.2.1 I/O接口模式

在SCLK输出模式下且SCxCR<IOC>设置为"0"时，发送缓冲器中的各位数据在从SCLK引脚输出的移位时钟的下降缘被输出到TXD引脚。

在SCLK输入模式下且SCxCR<IOC> 设置为"1"时，发送缓冲器中的各位数据按照 SCxCR<SCLKS>的设置，在SCLK输入信号的上升缘或下降缘被输出到TXD引脚。

14.12.2.2 UART模式

当发送数据写入发送缓冲器时，数据传输在下一TXDCLK的上升缘启动，并且也生成发送移位时钟信号。

14.12.3 发送操作

14.12.3.1 发送缓冲器的操作

如果双缓冲已被禁用，则CPU只会将数据写入到发送移位缓冲器，且一旦数据发送完成，即可生成发送中断INTTXx。

若双缓冲启用(包括发送FIFO启用的情况)，写入发送缓冲器的数据移至发送移位寄存器。同时，产生INTTXx中断，发送缓冲器空标志(SCxMOD2<TBEMP>)设置为"1"。该标志表示能写入下一发送数据。当下一数据写入发送缓冲器时，<TBEMP>标志被清除到"0"。

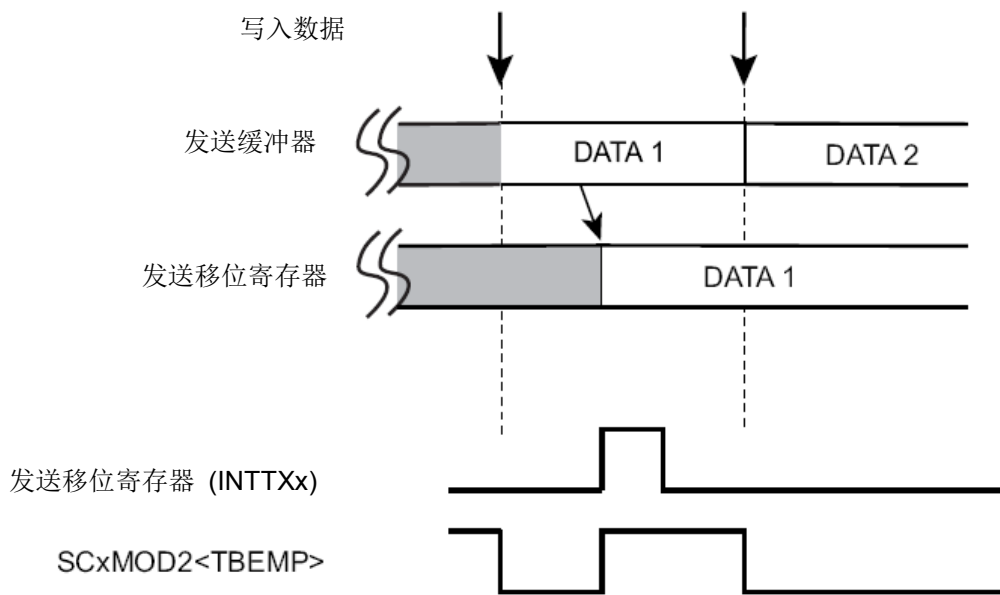


图 14-7 发送缓冲器的操作(双缓冲器已被启用)

14.12.3.2 发送FIFO操作

启用FIFO，用发送缓冲器和FIFO能储存最多5-字节的数据。

一旦启用传输，数据就从发送缓冲器转移到发送移位寄存器，并开始传输。若在FIFO中存在数据，数据被立即移到发送缓冲器，<TBEMP>标志被清除到"0"。

注：在使用TX FIFO缓冲器时，在设置SIO传输模式(半双工/全双工)和启用FIFO (SCxFCNF<CNFG> = "1")之后，必须清除TX FIFO。

通过将传输模式设置为半双工而发送4字节数据流的设置和操作如下所示。

SCxMOD1[6:5] = 10 :传输模式设置为半双工。

SCxFCNF[4:0] = 11011 :若FIFO为空，传输自动禁用。

在接收FIFO中使用的字节数与中断生成充满率相同。

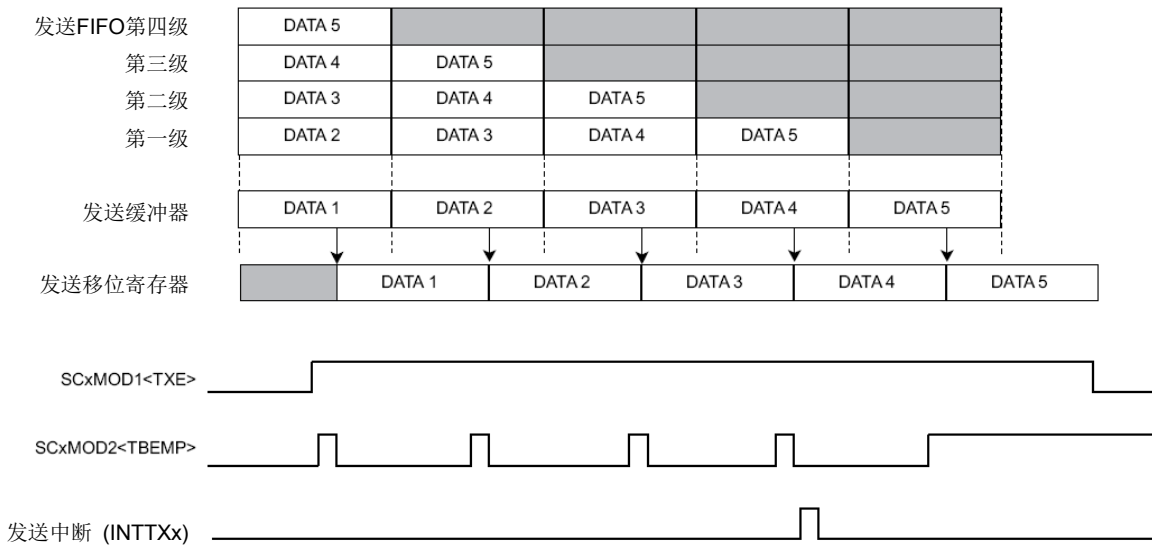
SCxTFC[1:0] = 00 :将中断生成充满率设置为"0"。

SCxTFC[7:6] = 11 :清除接收FIFO，并设置中断生成的条件。

SCxFCNF[0] = 1 :启用FIFO。

在以上设置配置完成之后，通过将5字节的数据写入发送缓冲器或FIFO，并将SCxMOD1<TXE>位设置为"1"，即可启动数据发送。当最后的发送数据移到发送缓冲器时，就会生成发送FIFO中断。在最后数据的发送完成时，时钟即停止不动，发送序列也被终止。

一旦配置上述设置，若传输未设置为自动禁用，则应写入发送数据，使传输持续进行。



14.12.3.3 I/O接口模式/通过SCLK输出实现发送

若为了在I/O 接口模式时生成时钟而设置SCLK，则当所有数据传输完成时，SCLK输出自动停止，不会发生欠载操作错误。

SCLK输出中止和恢复的时间随缓冲器和FIFO的使用而不同。

(1) 单缓冲器

每当转移一帧数据时，SCLK输出停止。可启用各数据与通信另一侧的握手。当下一数据写入缓冲器时，SCLK输出恢复。

(2) 双缓冲器

在发送移位寄存器和发送缓冲器的数据传输完成后，SCLK输出停止。当下一数据写入缓冲器时，SCLK输出恢复。

(3) FIFO

发送移位寄存器，发送缓冲器和FIFO中储存的所有数据的传输完成，SCLK输出停止。写入下一数据，SCLK输出恢复。

若配置SCxFCNF<RXTXCNT>，则在SCLK停止的同时，清除SCxMOD0<TXE>位，并且传输停止。

14.12.3.4 欠载操作错误

若在I/O接口SCLK输入模式时，发送FIFO禁用，并且若在下一帧时钟输入前在发送缓冲器中未设置任何数据，其在发送移位寄存器的数据传输完成后发生，则会发生欠载操作错误，SCxCR<PERR>设置为"1"。

带SCLK输出设置的I/O接口模式下，时钟输出自动停止，因此该标志无意义。

注：在将I/O界面SCLK输出模式切换为其它模式时，需读取SCxCR寄存器并清除该欠载操作标志。

14.13 握手功能

握手功能是为了启用 CTS(清除发送)引脚逐帧数据传输，并防止超限操作错误。功能由 SCxMOD0<CTSE>启用或禁用。

当 $\overline{\text{CTS}}$ 引脚设置为"高"阶时，虽然能完成当前数据传输，但是下一数据传输中止，直到 $\overline{\text{CTS}}$ 引脚恢复"低"电平。然而在这种情况下，在正常时间生成 INTTXx 中断，下一发送数据写入发送缓冲器，并且它会等到准备发送数据时为止。

- 注： (1) 如果 $\overline{\text{CTS}}$ 信号在期间被设置为"H"，则在当前发送完成之后，下一次数据发送即被暂停。
- (2) 在 $\overline{\text{CTS}}$ 被设置为"L"之后，在时钟的首个下降缘开始数据发送。

虽然未设有 $\overline{\text{RTS}}$ 引脚，但是通过给端口的一位分配 $\overline{\text{RTS}}$ 功能，就能轻易地实施握手控制功能。(在接收中断程序中)在数据接收完成后，通过将端口设置为"高"电平，就能要求发送侧中止数据传输。

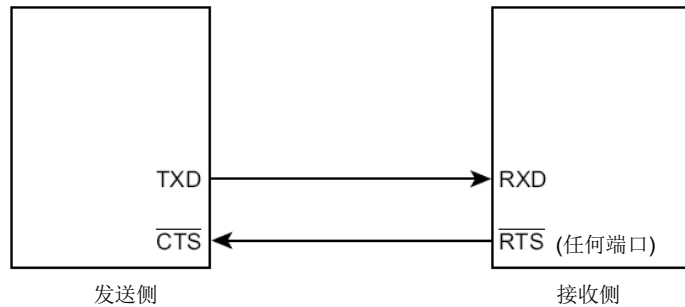


图 14-8 握手功能

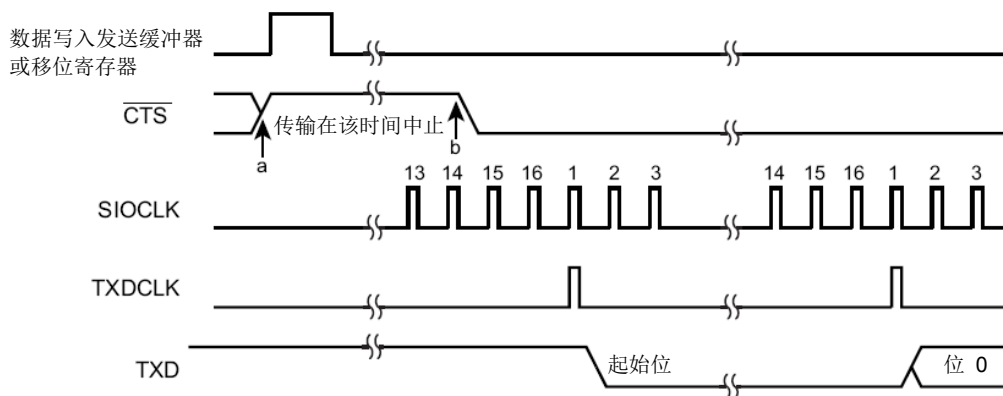


图 14-9 $\overline{\text{CTS}}$ 信号时序

14.14 中断/错误生成时序

14.14.1 RX中断

图 14-10 给出了接收操作的数据流和读取路径。

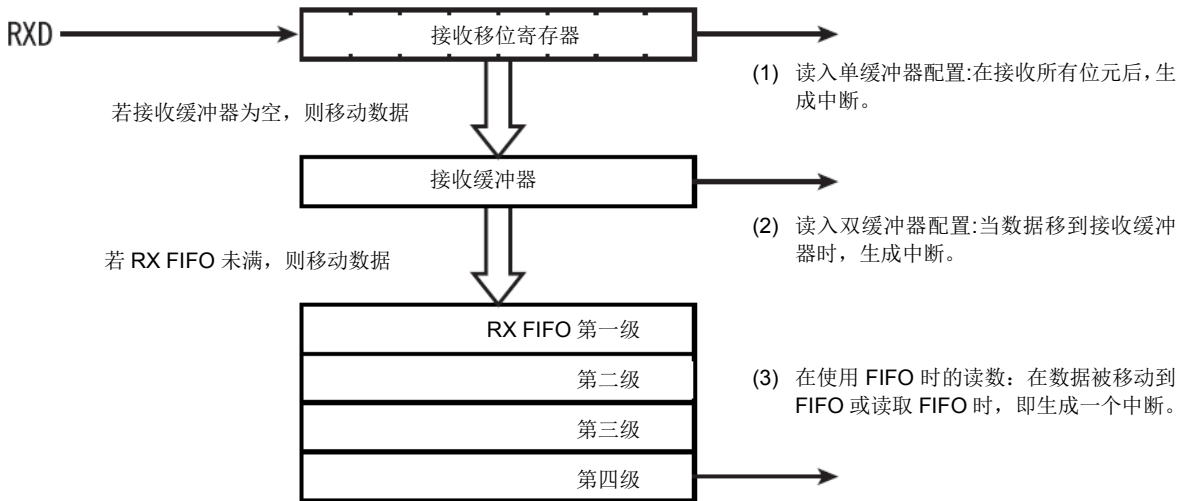


图 14-10 接收缓冲器/FIFO配置图表

14.14.1.1 单缓冲器/双缓冲器

如下表所示，在传输模式和缓冲器配置所确定的时间，生成RX中断。

缓冲器配置	UART模式	IO接口模式
单缓冲器	-	紧跟在最后一个SCLK的上升/下降缘 (上升或下降按SCxCR<SCLKS>设置确定)之后
双缓冲器	首个停止位的中心周围	紧跟在最后一个SCLK的上升/下降缘之后 (上升或下降按照SCxCR<SCLKS>的设置确定。) · 在通过读数缓冲器从移位寄存器将数据传送到缓冲器时。

注：在发生超限操作错误时，不会生成中断。

14.14.1.2 FIFO

在使用FIFO时，在操作和SCxRFC<RFIS>设置得到设立的条件，会生成接收中断。

一帧的所有位的接收完成。

读取FIFO

中断条件由表 14-12 所述的SCxRFC<RFIS>设置确定。

表 14-12 在采用FIFO时的接收中断条件

SCxRFC<RFIS>	中断条件
"0"	"FIFO充满率"等于"FIFO中断生成的充满率"。
"1"	"FIFO充满率"大于等于"FIFO中断生成的充满率"。

14.14.2 TX中断

图 14-11 给出了发送操作的数据流和读取路径。

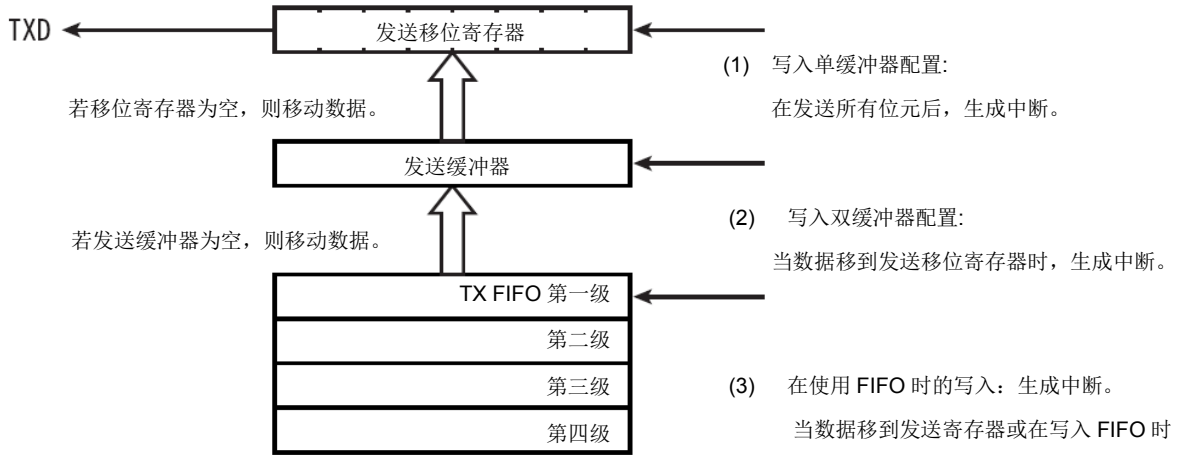


图 14-11 发送缓冲器FIFO配置图表

14.14.2.1 单缓冲器/双缓冲器

如下表所示, 在传输模式和缓冲器配置所确定的时间, 生成TX中断。

缓冲器配置	UART模式	IO接口模式
单缓冲器	恰在发送停止位前	紧跟在最后一个SCLK的上升/下降缘 (上升或下降按SCxCR<SCLKS>设置确定)之后
双缓冲器	当数据从发送缓冲器移到发送移位寄存器时。	

注: 如果已启用双缓冲器, 则在通过写入缓冲器而将数据从缓冲器移动到移位寄存器时, 也可生成一个中断。

14.14.2.2 FIFO

使用FIFO时, 在操作和SCxTFC<TFIS>设置得到设立的条件, 会生成发送中断。

一帧所有位发送完成。

写入FIFO

按表 14-13 所述的SCxTFC<TFIS>设置, 确定各中断条件。

表 14-13 在采用FIFO时的发送中断条件

SCxTFC<TFIS>	中断条件
"0"	"FIFO充满率"等于"FIFO中断生成的充满率"。
"1"	"FIFO充满率"小于等于"FIFO中断生成的充满率"。

14.14.3 错误生成

14.14.3.1 UART模式

模式	9 位	7 位 8 位 7 位 + 奇偶校验 8 位 + 奇偶校验
成帧错误 超限操作错误	在停止位中心周围	
奇偶错误	-	在奇偶校验位中心周围

14.14.3.2 IO 接口模式

超限操作错误	紧跟在最后一个SCLK的上升/下降缘之后 (上升或下降按照SCxCR<SCLKS>的设置确定。)
欠载操作错误	恰在下一SCLK上升缘或下降缘后。 (上升或下降按照SCxCR<SCLKS>的设置确定。)

注：在SCLK输出模式下，超限操作错误和欠载操作错误无意义。

14.15 软件复位

通过将SCxMOD2<SWRST[1:0]>写入为"10"(其后带有01)，即可生成软件复位。

因此，SCxMOD0<RXE>，SCxMOD1<TXE>，SCxMOD2<TBEMP><RBFL><TXRUN>，SCxCR <OERR><PERR><FERR>均被初始化。接收电路，发送电路和FIFO变成初始状态。其它状态则被保持。

14.16 DMA请求

在SIO/UART中断请求(INTRX_x，INTTX_x)计时时，可生成DMA请求。在采用DMA传输时，应设置SCxDMA (x=0, 1,)。

14.17 在各模式下的操作

14.17.1 模式0 (I/O 接口模式)

模式0由两种模式组成，即输出同步时钟的SCLK输出模式和接收外部来源的同步时钟的SCLK输入模式。

下列操作说明是针对FIFO禁用的情况。FIFO操作，详见先前描述接收/发送FIFO功能的章节。

14.17.1.1 发送数据

(1) SCLK输出模式

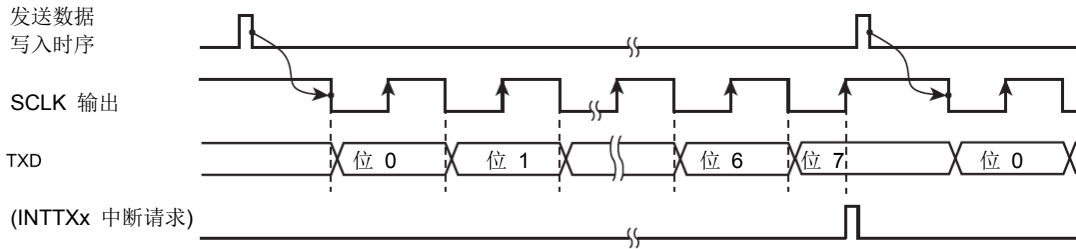
如果该发送双缓冲器已被禁用($SC_xMOD2<WBUF> = "0"$)

每当CPU把数据写入发送缓冲器时，数据从TXD引脚输出，时钟从SCLK引脚输出。当所有数据被输出时，生成中断INTTX_x。

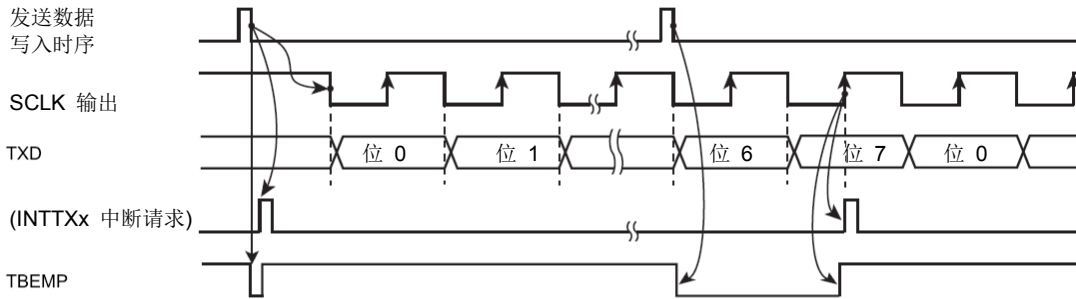
如果该发送双缓冲器已被启用($SC_xMOD2<WBUF> = "1"$)

当CPU把数据写入发送缓冲器，而数据传输停止时，或者当发送缓冲器的数据传输完成时，数据从发送缓冲器移到发送移位寄存器。同时，发送缓冲器空标志 $SC_xMOD2<TBEMP>$ 设置为"1"，生成INTTX_x中断。

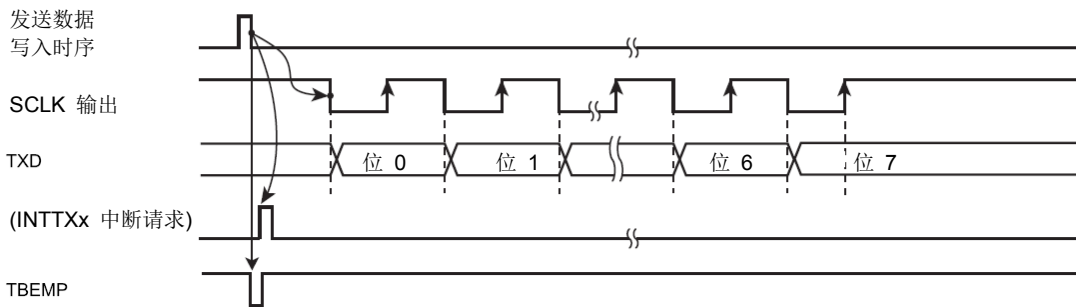
当数据从发送缓冲器移到发送移位寄存器时，若发送缓冲器无任何数据要移到发送移位寄存器，则不会生成INTTX_x中断，SCLK输出停止。



<WBUF> = "0" (若双缓冲禁用)



<WBUF> = "1" (若双缓冲启用, 并且在缓冲器中有数据)



<WBUF> = "1" (若双缓冲启用, 并且在缓冲器中无数据)

图 14-12 I/O接口模式(SCLK输出模式)下的发送操作

(2) SCLK输入模式

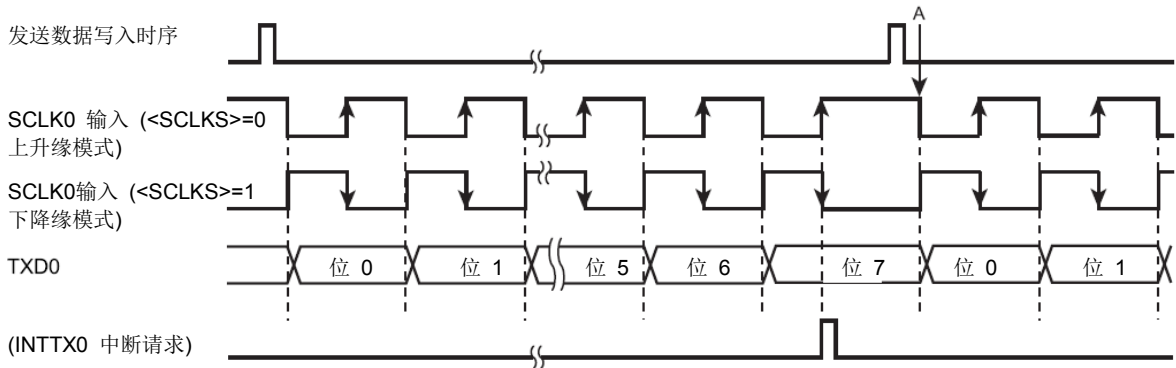
如果双缓冲已被禁用($SC_xMOD2<WBUF> = "0"$)

若在数据写入发送缓冲器的条件下输入SCLK, 则 8 位数据从TXD引脚输出。当所有数据被输出时, 生成中断INTTXx。必须在图 14-13 所示的计时点"A" 之前写入下一次发送数据。

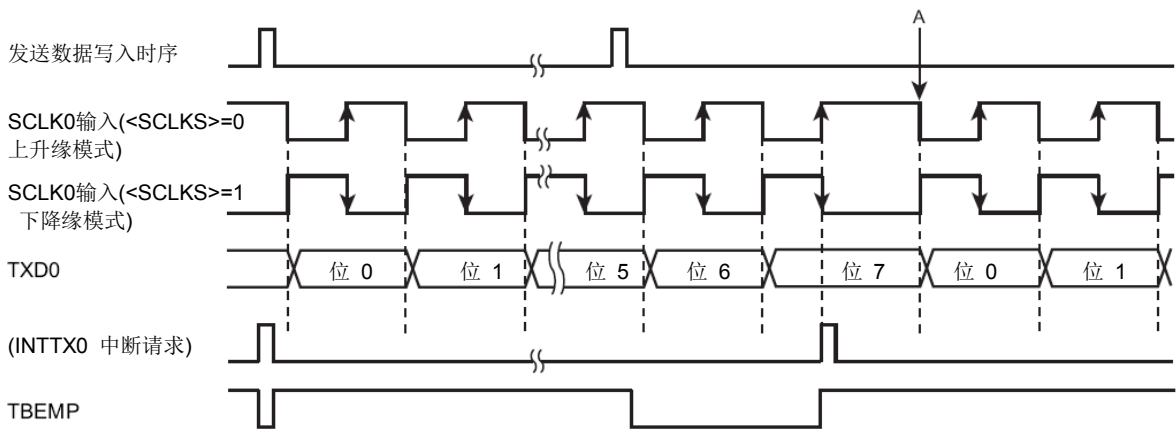
如果该双缓冲器已被启用($SC_xMOD2<WBUF> = "1"$)

当CPU在SCLK输入被激活前把数据写入发送缓冲器时, 或者当发送移位寄存器的数据传输完成时, 数据从发送缓冲器移到发送移位寄存器。同时, 发送缓冲器空标志 $SC_xMOD2<TBEMP>$ 设置为"1", 生成INTTXx中断。

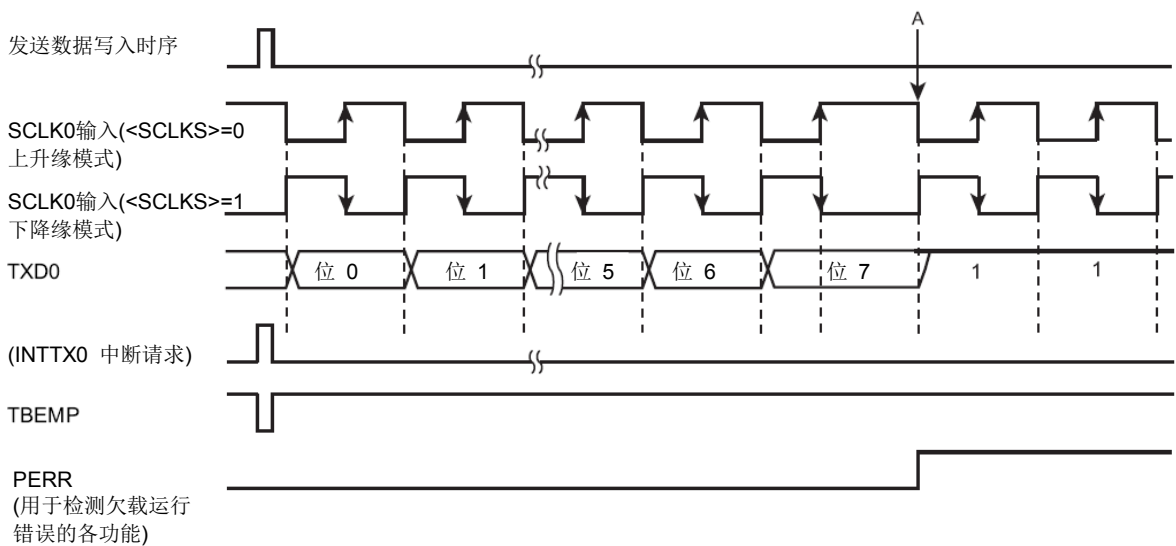
若SCLK输出被激活而在发送缓冲器中无数据, 则虽然内部位元计数器被启动, 但仍会发生欠载操作错误, 并发送 8 位虚拟数据(0xFF)。



<WBUF> = "0" (若双缓冲禁用)



<WBUF> = "1" (如果双缓冲器已启用且缓冲器 2 中有数据)



<WBUF> = "1" (如果双缓冲器已启用且缓冲器 2 中没有数据)

图 14-13 I/O接口模式 (SCLK输入状态) 下的发送操作

14.17.1.2 接收

(1) SCLK输出模式

将接收启用位SCxMOD0<RXE>设置为"1", 可启用SCLK输出。

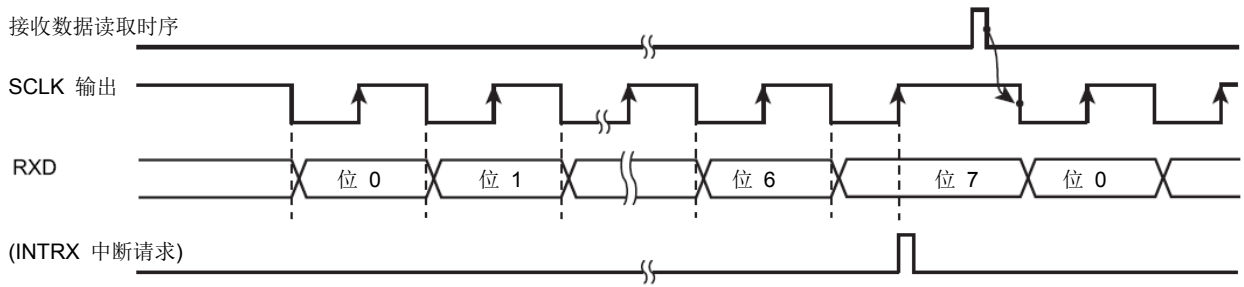
如果双缓冲器已被禁用(SCxMOD2<WBUF> = "0")

每当CPU读取接收的数据时, 时钟脉冲就会从SCLK引脚输出, 下一数据被储存在移位寄存器中。当接收到所有 8 位时, 生成INTRXx中断。

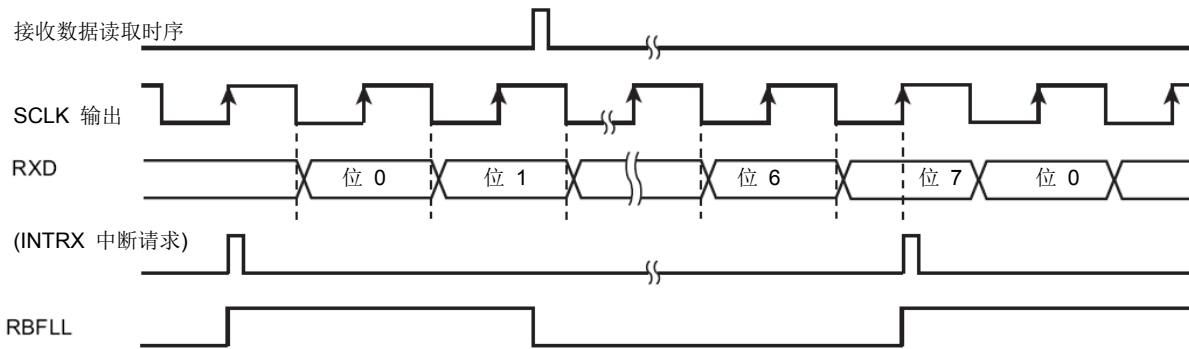
如果该双缓冲器已被启用(SCxMOD2<WBUF> = "1")

在移位寄存器中储存的数据移到接收缓冲器, 并且接收缓冲器能接收下一帧。数据从移位寄存器移到接收缓冲器, 接收缓冲器全满标志SCxMOD2<RBFL>设置为"1", 并且生成INTRXx。

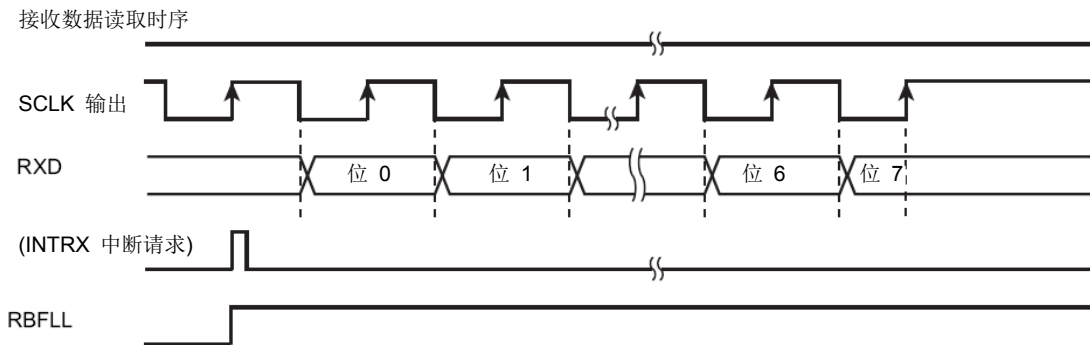
当数据在接收缓冲器中时, 若在完成下一 8 位的接收前无法从接收缓冲器读取数据, 则不生成INTRXx中断, 并且SCLK输出停止。在这种状态下, 读取接收缓冲器的数据可使移位寄存器中的数据移到接收缓冲器, 因此生成INTRXx中断, 数据接收恢复。



<WBUF> = "0" (双缓冲已被禁用)



<WBUF> = "1" (若双缓冲启用，并且从缓冲器读取数据)



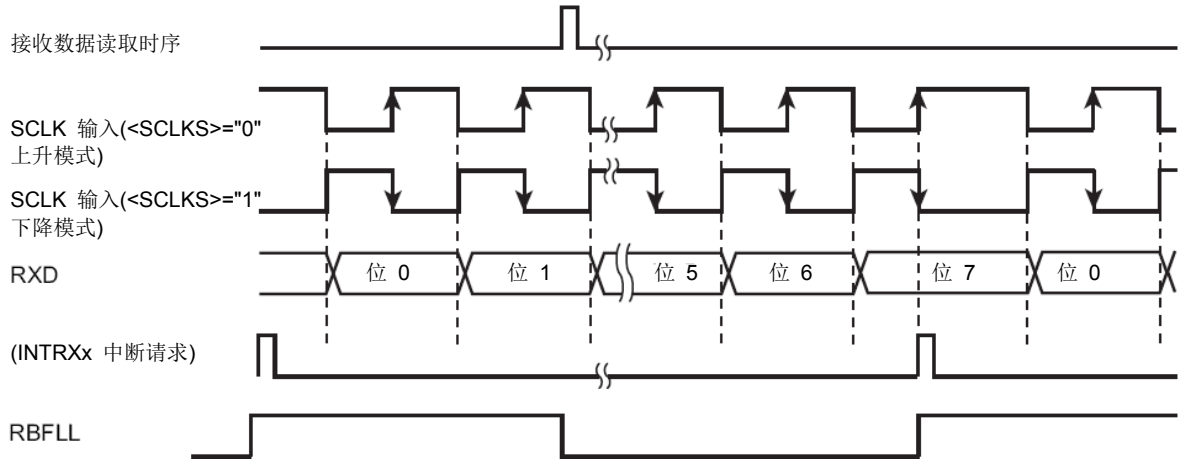
<WBUF> = "1" (若双缓冲启用，并且无法从缓冲器读取数据)

图 14-14 I/O接口模式(SCLK输出模式)下的接收操作

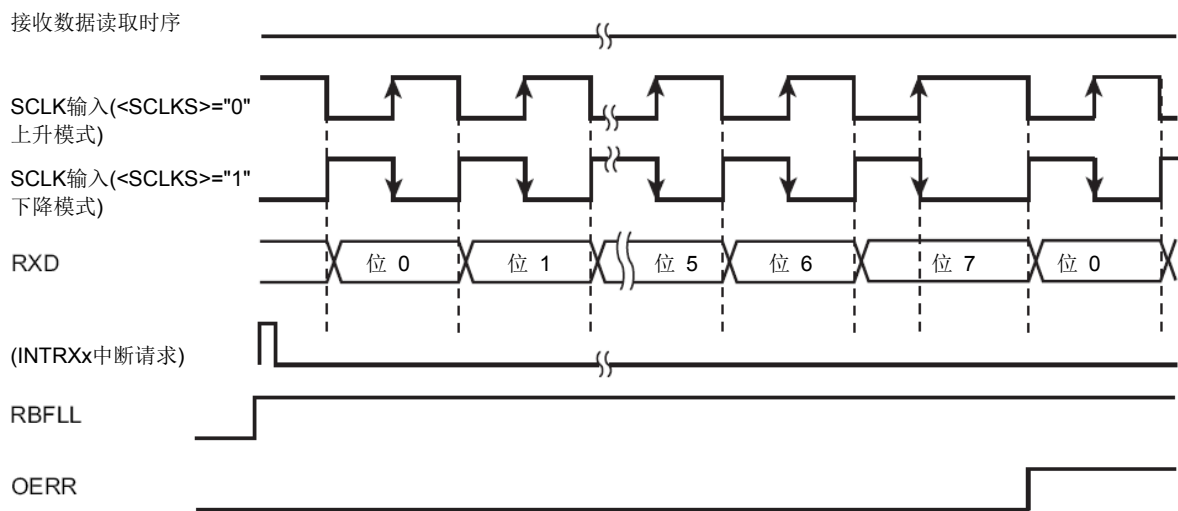
(2) SCLK输入模式

在SCLK输入模式时,接收双缓冲始终启用,接收的帧能从移位寄存器移到接收缓冲器,并且接收缓冲器能相继接收下一帧。

每当所接收的数据被移动到接收缓冲器,即可生成INTRx接收中断。



若从缓冲器读取数据



若无法从缓冲器读取数据

图 14-15 I/O接口模式(SCLK输入状态)下的接收操作

14.17.1.3 发送和接收(全双工)

(1) SCLK输出模式

如果SCxMOD2<WBUF>被设置为"0", 且双缓冲器已被禁用

SCLK当CPU把数据写入发送缓冲器时, 输出SCLK。

随后, 数据的8位被移入到接收缓冲器, 并生成INTRX_x接收中断。同时, 写入发送缓冲器的8位数据从TXD引脚输出, 当所有数据位的传输已完成时, 生成INTTX_x发送中断。然后, SCLK输出停止。

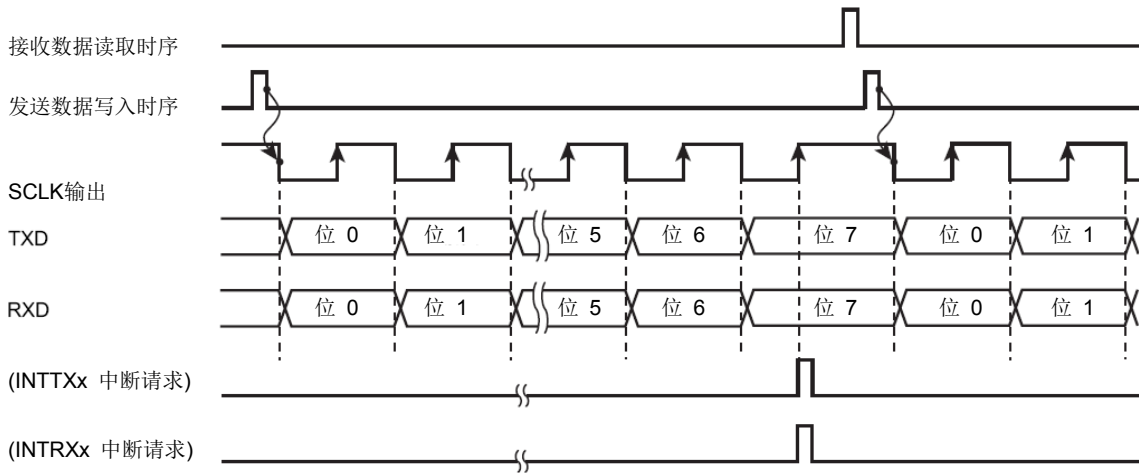
当从接收缓冲器读取数据并且下一发送数据由CPU写入发送缓冲器时, 下一轮数据传输和接收开始。读取接收缓冲器和写入发送缓冲器的顺序可自由确定。只有在满足这两个条件时, 数据传输才能恢复。

如果SCxMOD2<WBUF>被设置为"1", 且双缓冲器已被启用

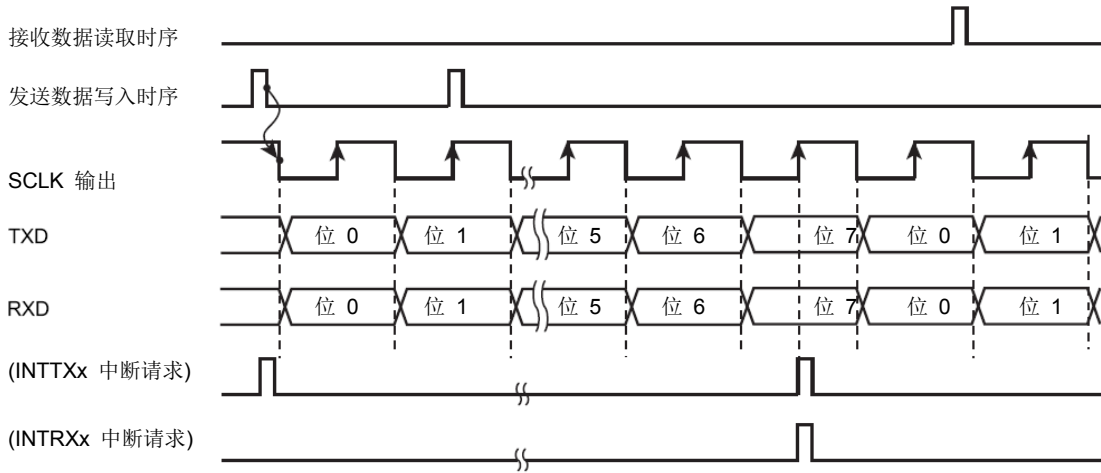
SCLK当CPU把数据写入发送缓冲器时, 输出SCLK。

8位数据被转移到接收移位寄存器, 移动到接收缓冲器, 并生成INTRX_x中断。当接收到8位数据时, 8位发送数据从TXD引脚输出。当所有数据位已被发送时, 生成INTTX_x中断, 下一数据从发送缓冲器移到发送移位寄存器。

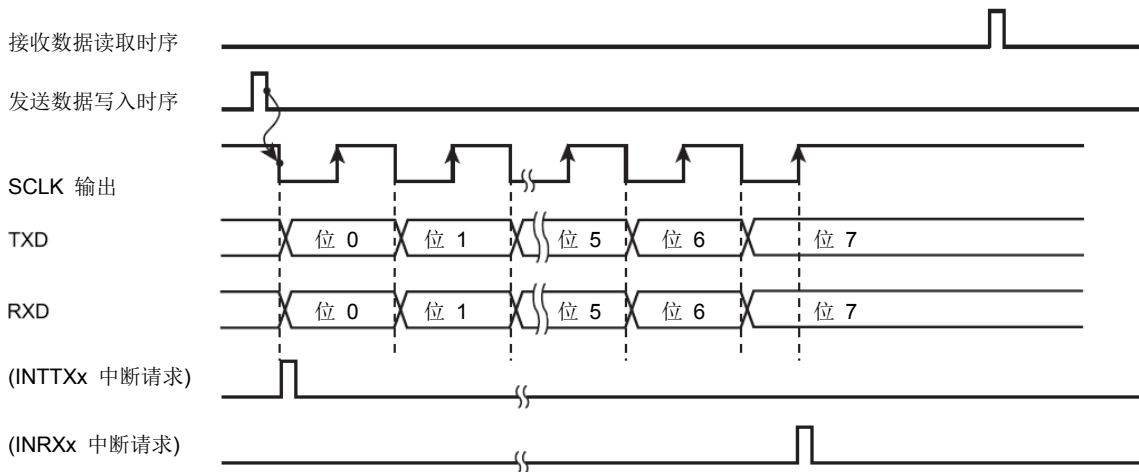
若发送缓冲器无数据要移到发送缓冲器时(SCxMOD2<TBEMP> = 1)或者当接收缓冲器全满时(SCxMOD2<RBFULL> = 1), SCLK输出停止。当满足这两个条件时(即接收数据被读取和发送数据被写入), SCLK输出恢复, 下一轮数据传输和接收开始。



<WBUF> = "0" (若双缓冲禁用)



<WBUF> = "1" (若双缓冲启用)



<WBUF> = "1" (若双缓冲启用)

图 14-16 I/O接口模式(SCLK输出模式)下的发送/接收操作

(2) SCLK输入模式

如果在接收数据时SCxMOD2<WBUF>被设置为"0"，且发送双缓冲器被禁用，则无论SCxMOD2 <WBUF>设置为何，双缓冲器都会始终处于已启用状态。

写入发送缓冲器的 8-位数据从TXD引脚输出，并且当SCLK输入被激活时，8位数据被转移到接收缓冲器。在数据传输完成后，生成INTTXx中断。在数据接收完成之后，数据被从移位寄存器移动到接收缓冲器时，即可生成INTTRXx中断。

注意，必须在下一帧的SCLK输入之前，将发送数据写入该发送缓冲器(必须在图14-17中的点A之前写入数据)。在下一帧数据的接收完成前，必须读取数据。

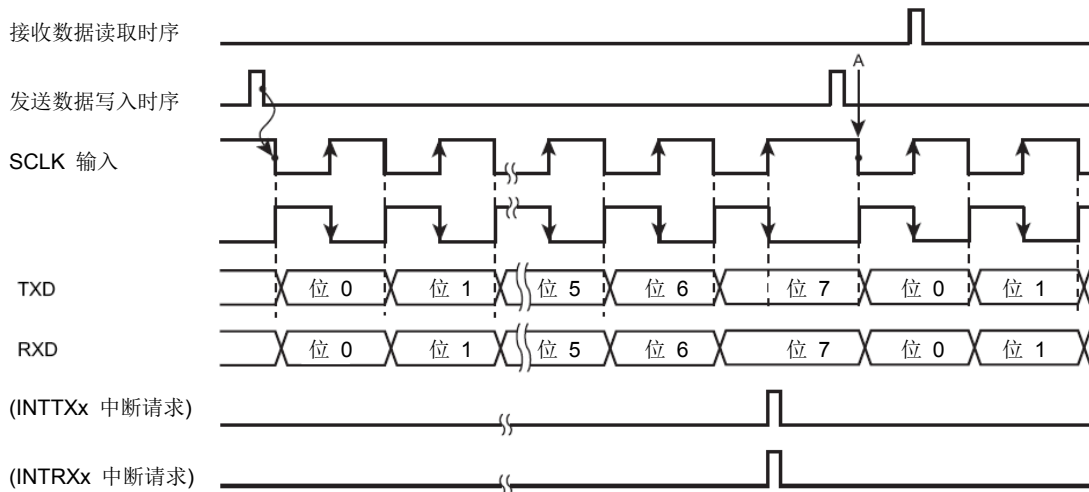
如果SCxMOD2<WBUF>被设置为"1"，且双缓冲器已被启用

在完成从发送移位寄存器的数据发送之后，在发送缓冲器数据被移动到该发送移位寄存器时生成中断INTRXx。同时，接收的数据被转移到移位寄存器，移动到接收缓冲器，并生成INTRXx中断。

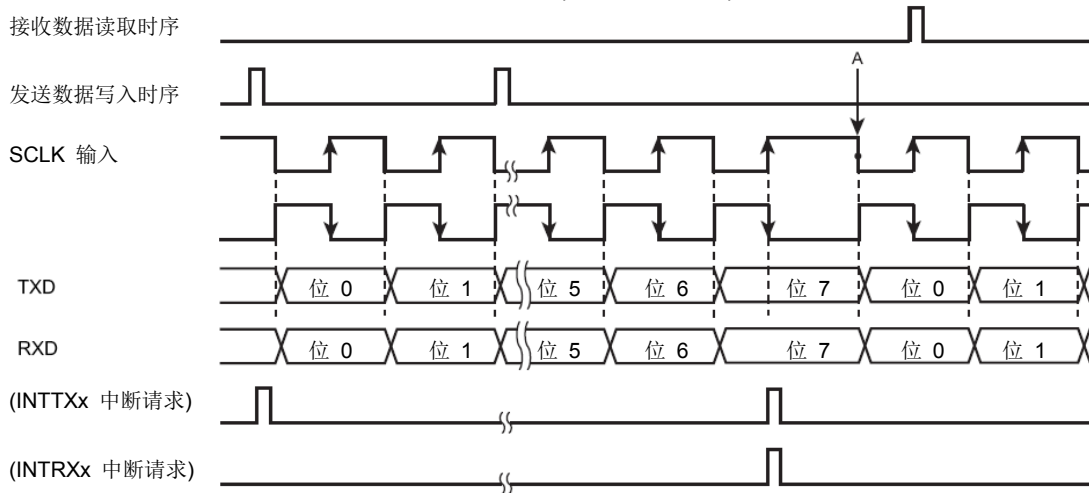
注意，必须在下一帧的SCLK输入之前，将发送数据写入该发送缓冲器(必须在图14-17中的点A之前写入数据)。在下一帧数据的接收完成前，必须读取数据。

在下一帧的SCLK输入后，发送移位寄存器(在该寄存器中，数据已从发送缓冲器中移动)的传输开始，同时接收数据被转移到接收移位寄存器。

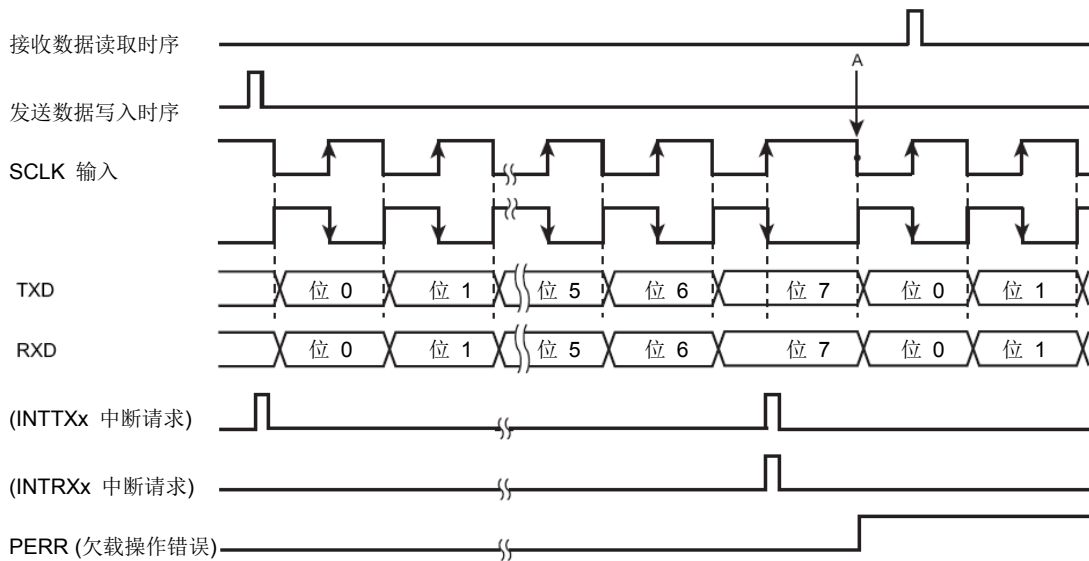
当收到帧的最后一位时，若接收缓冲器中的数据未被读取，则发生超限操作错误。同样，当下一帧的SCLK被输入时，若无数据写入发送缓冲器，则发生欠载操作错误。



<WBUF> = "0" (若双缓冲禁用)



<WBUF> = "1" (若双缓冲启用且无错误)



<WBUF> = "1" (若启用双缓冲，且错误生成)

图 14-17 I/O接口模式(SCLK输入模式)下的发送/接收操作

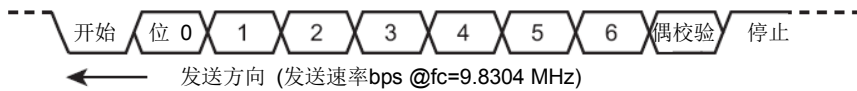
14.17.2 模式 1 (7 位UART模式)

通过将该串行模式控制寄存器 (SCxMOD<SM[1:0]>)设置为"01", 即可选择该 7 位UART模式。

在该模式下, 可将校验位添加到发送数据流; 串行模式控制寄存器(SCxCR<PE>)控制奇偶校验启用/禁用的设置。

当<PE>设置为"1" (启用)时, 可以用SCxCR<EVEN>位选择偶数或奇校验。用 SCxMOD2<SBLEN>指定停止位的长度。

以下列数据格式发送时, 控制寄存器的设置如下表如示。



计时条件	[系统时钟:	高速(fc)
		高速时钟齿轮:	X1 (fc)
		预分频器时钟:	fperiph/ 2 (fperiph = fsys)

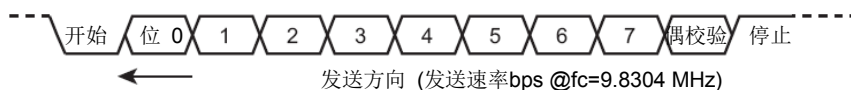
		7	6	5	4	3	2	1	0	
SCxMOD0	←	x	0	-	0	0	1	0	1	设置 7-位UART模式
SCxCR	←	x	1	1	x	x	x	0	0	偶校验启用
SCxBRCR	←	0	0	1	0	0	1	0	0	设置 2400 bps
SCxBUF	←	*	*	*	*	*	*	*	*	设置发送数据

X 忽略 -: 无变化

14.17.3 模式 2 (8-位UART模式)

通过将SCxMOD0<SM[1:0]> "10", 即可选择 8 位UART模式。该模式下, 能添加奇偶校验位, 并能用SCxCR<PE>控制奇偶校验的启用/禁用。若<PE> = "1" (启用), 则能用 SCxCR<EVEN>选择偶或奇校验。

为了以下列格式接收数据, 控制寄存器的设置如下:



计时条件	[系统时钟:	高速(fc)
		高速时钟齿轮:	X1
		预分频器时钟:	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	奇校验启用
SCxBRCR	←	0	0	0	1	0	1	0	0	设置 9600 bps
SCxMOD0	←	-	-	1	-	-	-	-	-	接收启用

X 忽略 - : 无变化

14.17.4 模式 3 (9-位UART模式)

通过将SCxMOD0<SM[1:0]> "11", 即可选择 9 位UART模式。在该模式下, 奇偶校验位必须禁用(SCxCR<PE> = "0")。

该最高有效位(第 9 位)即可被写入到用于数据发送的串行模式控制寄存器 0 (SCxMOD0)的位 7 <TB8>。该数据即被存入该串行控制寄存器SCxCR的位 7 <RB8>。

当向/从缓冲器写入或读取数据时, 最有效的位元必须在向/从SCxBUF写入或读取前先写入或读取。

可用SCxMOD2<SBLEN>指定停止位长度。

14.17.4.1 唤醒功能

在该 9-位UART模式下, 通过将该唤醒功能控制位SCxMOD0<WU>设置为"1", 各从属控制器即可在唤醒模式下运行。

在这种情况下, 只有在SCxCR<RB8>设置为"1"时, 才会生成中断INTRX_x。

注: 必须利用ODE寄存器, 将从属控制器的TXD引脚设置为开路漏极输出模式。

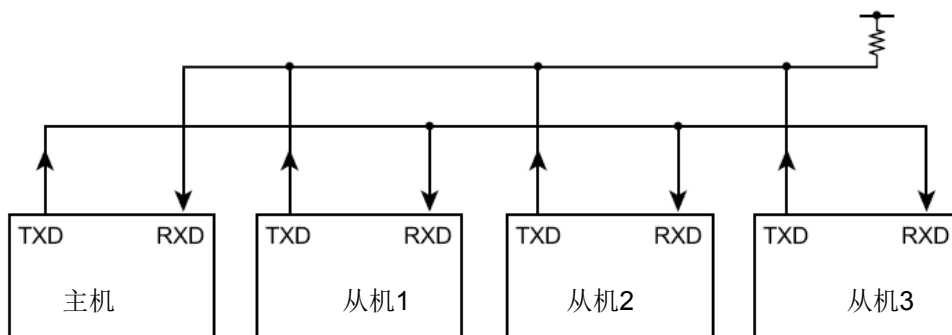
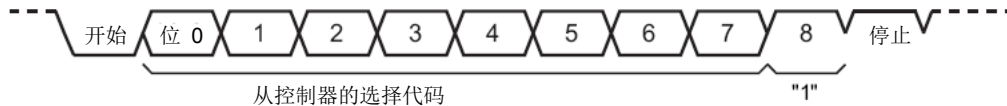


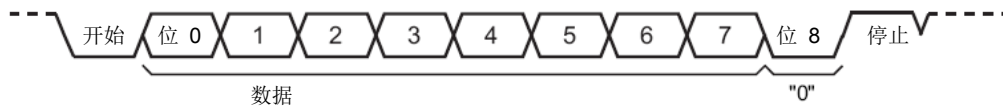
图 14-18 拟使用唤醒功能的各串行链接

14.17.4.2 协议

1. 选择各主从控制器的 9-位UART模式。
2. 将从控制器的SCxMOD<WU>设置为 "1", 使其随时可接收数据。
3. 主控制器拟发送一单帧的数据, 其中包含从控制器选择码(8 位)。在这种情况下, 最重要的位元(位 8) <TB8> 必须设置为"1"。



4. 各从控制器接收以上数据帧; 如果所接收的代码与控制器自身的选择码匹配, 则其会将WU位清"0"。
5. 主控制器发送数据给指定的从控制器(其SCxMOD<WU>位被清除为"0"的控制器)。在这种情况下, 最重要的位元(位 8) <TB8> 必须设置为"0"。



6. <WU>位被设置为"1"的从控制器会忽略该接收数据, 原因是最高有效位(位 8) <RB8> 被设置为"0", 从而未生成任何中断(INTRX_x)。此外, <WU>位被设置为"0"的从控制器可向主控制器发送数据, 以通知该数据已成功接收。

15. 串行总线接口 (I2C/SIO)

该TMPM365FYXBG带有 2 个串行总线接口(I2C/SIO)通道，其中包括以下两种运行模式：

I2C总线模式(具备多主控能力)

时钟同步 8-位SIO模式

在I2C总线模式下，I2C/SIO被通过SCL和SDA与外部设备连接。

在时钟同步 8-位SIO模式下，I2C/SIO被通过SCK，SI 和SO与外部设备连接。

下表给出了将I2C/SIO设置为各操作模式所需的编程。

表 15-1 使用串行总线接口时所需的端口设置

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	端口开路漏极输出控制寄存器
SBI0	I2C 总线模式	SCL0 : PG1 SDA0 : PG0	PGFR1[1:0] = 11	PGCR[1: 0] = 11	PGIE[1:0] = 11	PGOD[1:0] = 11
	SIO模式	SCK0 : PG2 SIO : PG1 SO0 : PG0	PGFR1[2:0] = 111	PGCR[2: 0] = 101(SCK0 输出) PGCR[2: 0] = 001(SCK0 输入)	PGIE[2:0] = 010(SCK0输 出) PGIE[2:0] = 110(SCK0输 入)	PGOD[2:0] = xxx
SBI1	I2C 总线模式	SCL1 : PE5 SDA1 : PE4	PEFR1[5:4] = 11	PECR[5:4] = 11	PEIE[5:4] = 11	PEOD[5:4] = 11
	SIO模式	SCK1 : PE6 S11 : PE5 SO1 : PE4	PEFR1[6:4] = 111	PECR[6:4] = 101(SCK1输 出) PECR[6:4] = 001(SCK1输 入)	PEIE[6:4] = 010(SCK1输 出) PEIE[6: 4] = 110(SCK1输 入)	PEOD[6:4] = xxx

注：x：忽略

15.1 配置

该配置如图 15-1 所示。

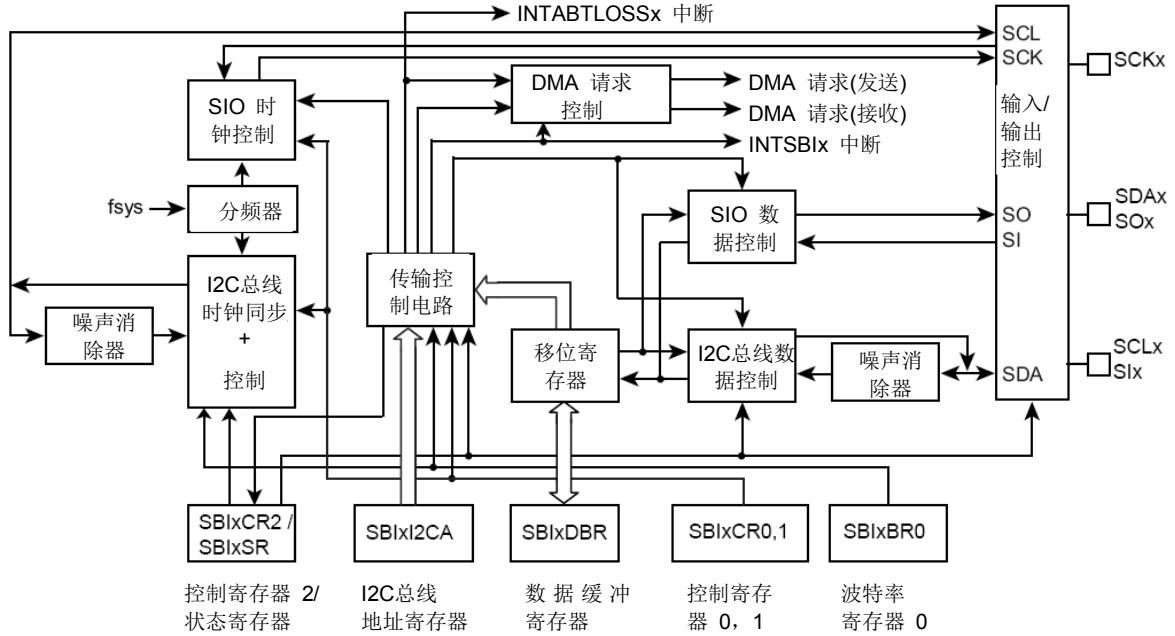


图 15-1 (I2C/SIO) 块接口

15.2 寄存器

以下寄存器控制串行总线接口，并提供其监控状态信息。

以下寄存器在不同模式下可执行不同的功能。有关详细资料，请参考"15.4 I2C总线模式下的控制寄存器"和"15.7 串行输入输出接口模式下的控制寄存器"。

15.2.1 各通道的寄存器

下表给出了各通道的寄存器和寄存器地址。

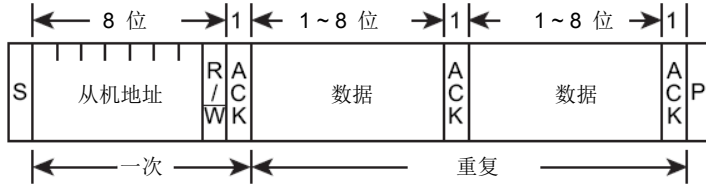
通道 x	基址
通道 0	0x400E_0000
通道 1	0x400E_0100

寄存器名称 (x=0, 1)		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

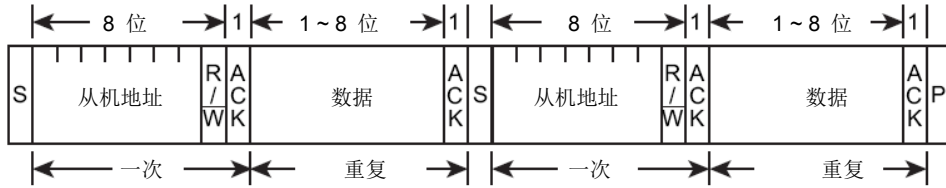
15.3 I2C总线模式数据格式

图 15-2 给出了I2C总线模式所采用的数据格式。

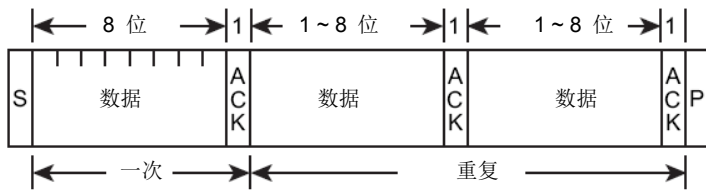
(a) 寻址格式



(b) 寻址格式 (具备重复起动条件)



(c) 自由数据格式 (主机发送器至从机接收器)



注) S: 启动条件

R/ \overline{W} : 方向位

ACK: 确认位

P: 停止条件

图 15-2 I2C 总线模式数据格式

15.4 I2C总线模式下的控制寄存器

以下寄存器可在I2C总线模式下控制串行总线接口，并提供其监控状态信息。

15.4.1 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	类型	功能
31-8	-	R	读作 0。
7	SBIEN	R/W	如果该位在其启用之后即被禁用，则各寄存器的设置均可保留。串行总线接口操作 0：禁用 1：启用 在使用该串行总线接口时，需启用第一个位。 在首次设置为启用时，可读取或写入相关的SBI寄存器。该位被禁用时所有时钟SBIXCR0除外均停止工作，因此，禁用该位可减少功耗。 如果该位在启用之后及被禁用，则各寄存器的设置均可保留。
6-0	-	R	读作 0。

注：在使用该串行总线接口时，先启用这个位。

15.4.2 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	类型	功能																																																	
31-8	-	R	读作 0。																																																	
7-5	BC[2:0]	R/W	选择每次传输的位数目(注 1) <table border="1" style="margin-left: 20px; border-collapse: collapse; text-align: center;"> <thead> <tr> <th rowspan="2"><BC></th> <th colspan="2">在 <ACK> = 0时</th> <th colspan="2">在 <ACK> = 1时</th> </tr> <tr> <th>时钟周期的数目</th> <th>数据长度</th> <th>时钟周期的数目</th> <th>数据长度</th> </tr> </thead> <tbody> <tr><td>000</td><td>8</td><td>8</td><td>9</td><td>8</td></tr> <tr><td>001</td><td>1</td><td>1</td><td>2</td><td>1</td></tr> <tr><td>010</td><td>2</td><td>2</td><td>3</td><td>2</td></tr> <tr><td>011</td><td>3</td><td>3</td><td>4</td><td>3</td></tr> <tr><td>100</td><td>4</td><td>4</td><td>5</td><td>4</td></tr> <tr><td>101</td><td>5</td><td>5</td><td>6</td><td>5</td></tr> <tr><td>110</td><td>6</td><td>6</td><td>7</td><td>6</td></tr> <tr><td>111</td><td>7</td><td>7</td><td>8</td><td>7</td></tr> </tbody> </table>	<BC>	在 <ACK> = 0时		在 <ACK> = 1时		时钟周期的数目	数据长度	时钟周期的数目	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
<BC>	在 <ACK> = 0时		在 <ACK> = 1时																																																	
	时钟周期的数目	数据长度	时钟周期的数目	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: 未生成确认时钟脉冲。 1: 生成了确认时钟脉冲。 从属模式 0: 未计数确认时钟脉冲。 1: 已计数确认时钟脉冲。																																																	
3	-		R读作1。																																																	
2-1	SCK[2:1]	R/W	选择内部SCL输出时钟频率(注2)。																																																	
	SCK[0]	W	<table border="1" style="margin-left: 20px; border-collapse: collapse; text-align: center;"> <tbody> <tr><td>000</td><td>n = 5</td><td>462 kHz</td></tr> <tr><td>001</td><td>n = 6</td><td>353 kHz</td></tr> <tr><td>010</td><td>n = 7</td><td>240 kHz</td></tr> <tr><td>011</td><td>n = 8</td><td>146 kHz</td></tr> <tr><td>100</td><td>n = 9</td><td>82 kHz</td></tr> <tr><td>101</td><td>n = 10</td><td>44 kHz</td></tr> <tr><td>110</td><td>n = 11</td><td>23 kHz</td></tr> <tr><td>111</td><td></td><td>保留</td></tr> </tbody> </table> <div style="margin-left: 40px;"> $\left. \begin{array}{l} \text{系统时钟: } f_{\text{sys}} \\ \text{时钟齿轮: } fc/1 \\ \text{频率} = \frac{f_{\text{sys}}}{2^n + 72} \text{ [Hz]} \end{array} \right\} \begin{array}{l} (= 48\text{MHz}) \end{array}$ </div>	000	n = 5	462 kHz	001	n = 6	353 kHz	010	n = 7	240 kHz	011	n = 8	146 kHz	100	n = 9	82 kHz	101	n = 10	44 kHz	110	n = 11	23 kHz	111		保留																									
000	n = 5	462 kHz																																																		
001	n = 6	353 kHz																																																		
010	n = 7	240 kHz																																																		
011	n = 8	146 kHz																																																		
100	n = 9	82 kHz																																																		
101	n = 10	44 kHz																																																		
110	n = 11	23 kHz																																																		
111		保留																																																		
	SWRMON	R	读取时<SWRMON>: 软件复位状态监控程序 0: 软件复位操作在进行中。 1: 软件复位操作未在进行中。																																																	

注 1: 在将运行模式切换为SIO模式之前, 将<BC[2:0]>清除为"000"。

注 2: 有关SCL线路时钟频率的详细资料, 请参考"15.5.1 串行时钟"。

注 3: 在复位之后, <SCK[0]/SWRMON>位即被读作"1"。不过, 如果在SBIxCR2寄存器选择SIO模式, 则<SCK[0]>位的初始值为"0"。

注 4: 选择某个频率的初始值为<SCK[2:0]>=000, 且与所读取的初始值无关。

注 5: 在主模式下<BC[2:0]>="001" 且<ACK>="0"时, SCL可能STOP条件生成之后被SCL线路的下降缘固定为"L", 且另一设备不能使用该母线。如果是与若干主机有关的总线, 则应在STOP条件生成之前, 将每次传输的位数目设置为等于或大于 2。

15.4.3 SBIXCR2(控制寄存器 2)

通过读取，该寄存器可用作SBIXSR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	MST	TRX	BB	PIN	SBIM		SWRST	
复位后	0	0	0	1	0	0	0	0

位	比特符号	类型	功能
31-8	-	R	读作 0。
7	MST	W	选择主机/从机 0: 从属模式 1: 主模式
6	TRX	W	选择发送/接收 0: 接收 1: 传输
5	BB	W	启动/停止条件生成 0: 停止条件已生成 1: 启动条件已生成
4	PIN	W	清除INTSBIX中断请求 0: - 1: 清除中断请求
3-2	SBIM[1:0]	W	选择串行总线接口操作模式(注) 00: 端口模式(禁用某个串行总线接口输出) 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"(其后带有"01")以生成一个复位。

注：确认各模式在通信会话期间未被改变。确保在将操作模式切换为端口模式之前，该总线处于空闲状态。确保在将操作模式从端口模式切换到I2C总线或时钟同步 8-位SIO模式之前，该端口处于"高"电平。

15.4.4 SBIXSR (状态寄存器)

通过向其写入，该寄存器起SBIxCR2寄存器的作用。

	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	MST	TRX	BB	PIN	AL	AAS	ADO	LRB
复位后	0	0	0	1	0	0	0	0

位	比特符号	类型	功能
31-8	-	R	读作 0。
7	MST	R	主机/从机选择监控程序 0: 从属模式 1: 主模式
6	TRX	R	发送/接收选择监控程序 0: 接收 1: 传输
5	BB	R	I2C总线状态监控程序 0: 自由 1: 占空
4	PIN	R	INTSBIX中断请求监控程序 0: 中断请求已生成 1: 中断请求已被清除
3	AL	R	判优丢失检测 0: - 1: 已检测
2	AAS	R	从机地址匹配检测 0: - 1: 已检测 (在也检测到一般调用时，即设置该位。)
1	ADO	R	一般调用检测 0: - 1: 已检测
0	LRB	R	最后接收位监控程序 0: 最后接收位"0" 1: 最后接收位"1"

15.4.5 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	类型	功能
31-8	-	R	读作 0。
7	-	R	读作 1。
6	I2SBI	R/W	IDLE模式下的操作 0: 停止 1: 操作
5-1	-	R	读作 1。
0	-	R/W	务必写入 0。

15.4.6 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	类型	功能
31-8	-	R	读作 0。
7-0	DB[7: 0]	R (接收)	接收数据
		W (发送)	传送数据

注 1: 必须将发送数据从MSB(位7)写入到该寄存器。已接收数据即可被存储到LSB中。

注 2: SBIXBDR具备独立的写入和读取缓冲器, 因此写数据无法被读取。因而无法使用位操作等读取修改-写入指令。

15.4.7 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	类型	功能
31-8	-	R	读作 0。
7-1	SA[6:0]	R/W	在SBI作为从机运行时，设置从机地址。
0	ALS	R/W	指定地址识别模式。 0：识别其从机地址。 1：不识别其从机地址(自由数据格式)。

注 1：请将I2C总线地址寄存器SBIxI2CAR的位 0 <ALS> 设置为"0"，但用户使用自由数据格式的情形除外。在其被设置为"1"时，其作为自由数据格式运行。选择拟被固定为发送功能的主机。选择拟被固定为接收功能的从机。

注 2：在从属模式下，不要将SBIxI2CAR设置为"0x00"。(如果SBIxI2CAR被设置为"0x00"，则会认识到从机地址可与在从属模式下接收到的I2C标准的START字节("0x01") 匹配。)

15.5 I2C总线模式下的控制器

15.5.1 串行时钟

15.5.1.1 时钟源

SBIxCR1<SCK[2:0]>可在主模式下，指定该串行时钟拟从SCL引脚输出的最高频率。

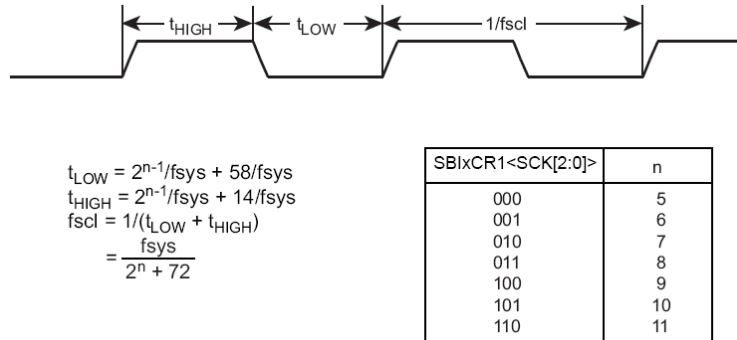


图 15-3 时钟源

注：按照通信标准的要求，标准模式和高速模式下的最高速度被分别指定为 100 kHz和 400 kHz注意，可依据所使用的f_{sys}和上文给出的计算公式，求出内部SCL时钟频率。

15.5.1.2 时钟同步

因受其引脚结构影响，需采用配线AND连接驱动I2C总线。将其时钟线拖至"低"电平的首个主机，超越在其时钟线上产生"高"电平时其它主机。产生"高"电平的主机必须就此进行检测并应答。

时钟同步可保证具备两个或以上主机的总线的正确数据传输。

例如，以下给出了带有两个主机的总线的时钟同步程序。

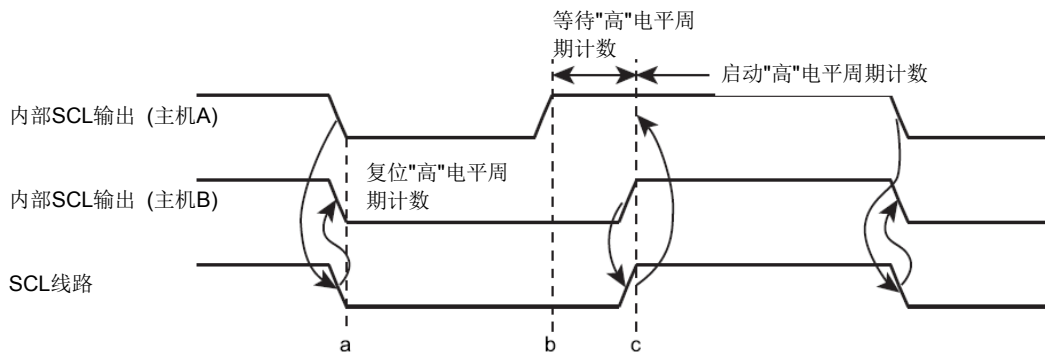


图 15-4 时钟同步示例

主机A在点a时将其内部SCL输出拖至"低"电平，使得SCL总线也变为"低"。主机B检测到该转变，重置其"高"电平周期计数器，并将其内部SCL输出电平拖至"低"电平。

主机A在点b完成其"低"电平周期的计数, 并使其内部SCL输出变为"高"电平。不过, 主机B仍然会使SCL总线保持在"低"电平, 主机A停止其"高"电平周期计数。在主机A检测到主机B使其内部SCL输出变为"高"电平, 并使该SCL总线在点c变为"高"电平之后, 其开始其"高"电平周期的计数。

在该主机完成"高"电平周期的计数之后, 主机将拖至"低", SCL总线随之变为"低"。

通过这种方法, 即可依据已被连接到该总线的主机中"高"电平周期最短的主机, 以及"低"电平周期最长的主机, 确定该总线的时钟。

15.5.2 确认模式的设置

将SBIxCR1<ACK>设置为 "1", 即可选择该确认模式。在作为主机运行时, SBI可添加一个确认信号。在从属模式下, 可对确认信号的时钟进行计数。在发送器模式下, SBI会在时钟周期期间解除该SDAx引脚, 以接收来自接收器的确认信号。在接收器模式下, SBI会在时钟周期期间将该SDAx引脚拖至"低"电平, 并生成确认信号。此外, 在从属模式下, 如果接收到一个一般调用地址, 则SBI会在该时钟周期期间将该SDAx引脚拖至"低"电平, 并生成确认信号。

通过将 <ACK>设置为"0", 即可触发非确认模式。在运行某个主机时, SBI不会生成确认信号的时钟。在从属模式下, 不会对确认信号的时钟进行计数。

15.5.3 每次传输的位数的设置

SBIxCR1<BC[2:0]>可指定拟发送或接收的下一数据的位数。

在启动条件下, <BC[2:0]>被设置为 "000", 导致从机地址和方向位被置于某个八位包中进行传送。在其他时候, <BC[2:0]>会保持此前某个编程值。

15.5.4 从机寻址和地址识别模式

通过将"0"设置到SBIxI2CAR<ALS>和SBIxI2CAR<SA[6:0]>中的某个从机地址, 即可寻址格式, 然后该SBI识别出主机所发送的从机地址, 并以该寻址格式接收数据。

如果<ALS>被设置为"1", 则SBI不会识别从机地址, 并会以自由数据格式接收数据。如果是自由数据格式, 则不会识别从机地址和方向位; 在该启动条件生成之后, 其即被识别为数据。

15.5.5 操作模式

通过设置SBIxCR2<SBIM[1:0]>, 即可控制操作模式。在I2C时操作时, 在将<SBIM[1:0]>设置为"10"之前, 应确保串行总线接口引脚位于"高"电平。此外, 还应确保在将操作模式切换到端口模式之前, 总线处于空闲状态。

15.5.6 将SBI配置为发送器或接收器

通过将SBIxCR2<TRX>设置为"1",即可将SBI配置为发送器。通过将<TRX>设置为"0",即可将SBI配置为接收器。

在从属模式时:

在数据被以寻址格式发送时。

在所接收的从机地址匹配在SBIxI2CAR指定的值时。

在某个一般调用地址被收到时;亦即,该启动条件后面的八位全部为零。

如果方向位($\overline{R/W}$)的值为"1",则<TRX>会被该硬件设置为"1"。如果该位为"0",则<TRX>会被设置为"0"。

作为主机,SBI可接收来自某个从机的确认。如果发送了方向位"1",则<TRX>会被该硬件设置为"0"。如果该方向位为"0",则<TRX>会变为"1"。如果该SBI不接收确认,则<TRX>保有先前的值。

在其检测到总线上的该停止条件,或判优丢失时,<TRX>即被硬件清除为"0"。

如果SBI被用于自由数据格式,则硬件不会改变<TRX>。

15.5.7 将SBI配置为主机或从机

通过SBIxCR2<MST>将设置为"1",即可将SBI配置为主机并进行操作。

通过将<MST>设置为"0",即可将SBI配置为从机。在其检测到总线上的该停止条件,或判优丢失时,<MST>即被硬件清除为"0"。

15.5.8 启动和停止条件的生成

在SBIxSR<BB>为"0"时,通过将"1"写入到SBIxCR2<MST, TRX, BB, PIN>,即可导致SBI启动某个启动条件生成序列,并输出未来将被写入数据缓冲寄存器的从机地址和方向位。必须将<ACK>预先设置为"1"。

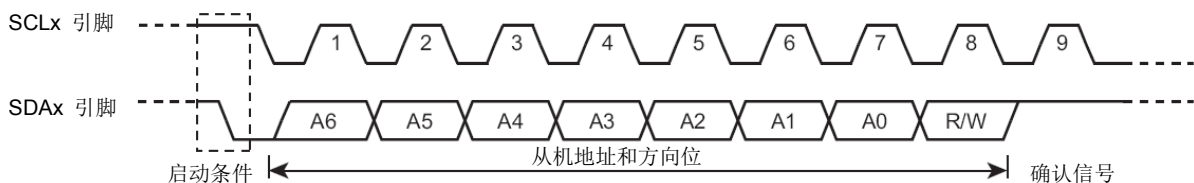


图 15-5 启动条件和从机地址的生成

在<BB>为"1"时,通过将"1"写入到<MST, TRX, PIN>,并将"0"写入到<BB>,即可导致SBI启动母线上的某个停止条件生成序列,且在该停止条件出现在该母线上之前,不得改动<MST, TRX, BB, PIN>的内容。

如果SCL总线在停止条件生成时被其它设备拖至"低", 则该停止条件会在SCL线路被解除时生成。

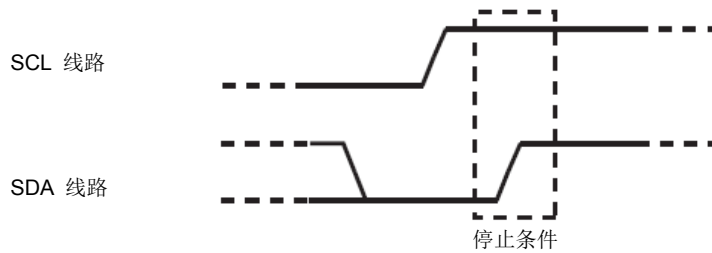


图 15-6 停止条件的生成

可读取SBIxSR<BB>, 即可检查总线状态。在总线上检测到该启动条件时(总线占空), <BB>即被设置为"1", 并在检测到停止条件时(总线空闲)被清除为"0"。

15.5.9 中断服务请求和解除

在主模式下, 在<BC>和<ACK> 所设置的时钟周期的数目的传输完成时, 串行总线接口请求(INTSBIx)即被生成。

在从属模式下, 可在以下条件下生成INTSBIx。

在所接收的从机地址可匹配被设置为SBIxI2CAR<SA[6:0]>的从机地址时所生成的确认信号输出发生之后。

在接收到一般调用地址时确认信号被生成之后。

在接收某个一般调用地址之后从机地址可匹配或某个数据传输完成时。

在地址识别模式下(<ALS> = "0"), 在所接收的从机地址可匹配在SBIxI2CAR指定的各值时, 或在某个一般调用(该启动条件之后的八位数据全部为"0")被接收时, INTSBIx即被生成。

在一个中断请求(INTSBIx)被生成时, SBIxCR2<PIN>即被清除为"0"。在<PIN>被清除为"0"的同时, SBI会将SCL线路拖至"低"电平。

在数据被写入到SBIxDBR或被从SBIxDBR读出时, <PIN>被设置为"1"。在<PIN>被设置为"1"之后, SCL线路的解除需花费一段时期 t_{LOW} 。在程序将"1"写入到<PIN>时, 其即被设置为"1"。不过, 写入"0"不会将该位清除为"0"。

注: 在主模式下丢失判优时, 如果从机地址不匹配(已生成INTSBIx), 则<PIN>不会被清除为"0"。

注: 在INTSBIx中断发生时, 会同时发生DMA请求。如果使用的是DMA, 则需设置DMA控制器的相应寄存器。

15.5.10 判优丢失检测监控程序

该I2C总线具备多主机能力(一根总线上可设置两个或更多主机),并要求利用总线判优程序,以确保正确的数据传输。

试图在总线占空期间生成该启动条件的某个主机丢失总线判优,SDA和SCL线路上未出现启动条件。I2C总线判优会发生在SDA线路上。某个启动条件在总线占空状态下被输出时,某个启动条件不会被输出到SCL或SDA线路,从而发生判优丢失。

一条总线上的两个主机的判优程序如下所述。

在到达点a之前,主机A和主机B输出相同的数据。在点a时,主机A输出"低"电平,主机B输出"高"电平。

然后,主机A将SDA总线拖至"低"电平,原因是该线路具备配线AND连接。在SCL线路在点b变为高时,从机会读取SDA线路数据(即主机A所发送的数据)。此时,主机B所发送的数据变为无效。

主机B的这种情况被称为"判优丢失"。主机B解除其SDA引脚,以确保其不会影响其它主机所发起的数据传输。如果两个或更多主机已经发送完全相同的第一个数据字,则会对第二个数据字继续进行判优过程。

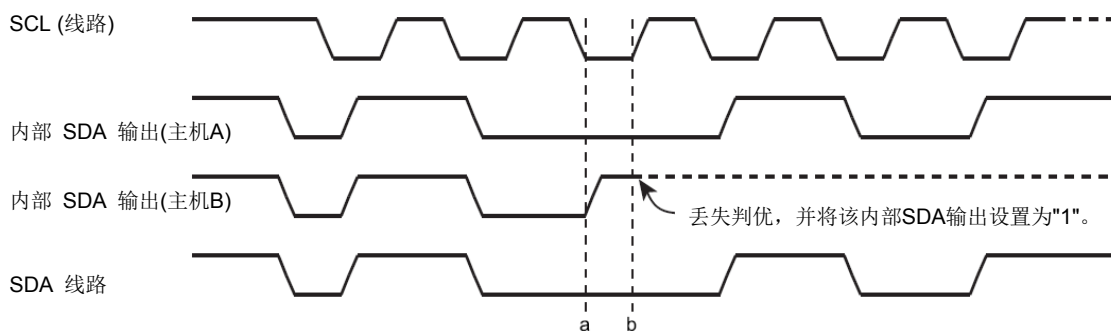


图 15-7 丢失判优

主机会比较SDA总线电平和SCL线路上升部位的内部SDA输出电平。如果这两个值之间存在差异,则判优丢失发生,SBIxSR<AL>被设置为"1",INTABLOSSx中断发生。

在<AL>被设置为"1"时,SBIxSR<MST与TRX>被清除为"0",导致SBI作为从属接收器运行。因此,在<AL>被设置为"1"之后,串行总线接口电路会在数据传输期间停止该时钟输出。

在数据被写入到SBIxDBR或被从SBIxCR2读出时,<AL>即被清除为"0"。

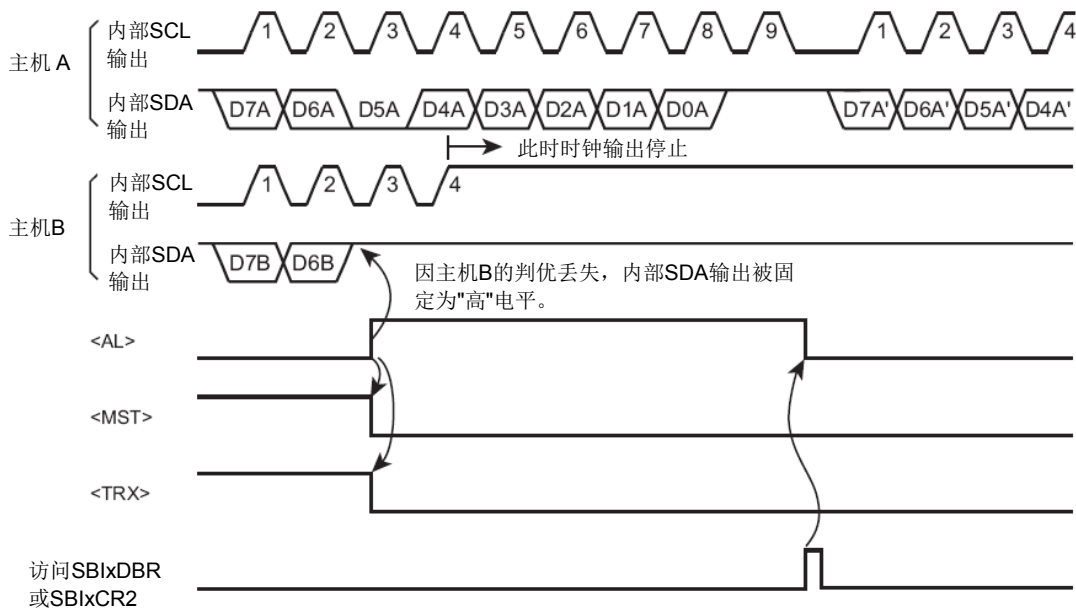


图 15-8 主机B丢失判优示例(D7A = D7B, D6A = D6B)

15.5.11 从机地址匹配检测监控程序

当SBI在地址识别模式下作为从机运行时(SBIxI2CAR<ALS>="0"), 一旦接收到一般调用地址, 或从机地址可匹配在SBIxI2CAR指定的值, SBIxSR<AAS>即被设置为"1"。

在<ALS>为"1"时, <AAS>在首个数据字被接收时被设置为"1"。在数据被写入到SBIxDBR或被从SBIxDBR读出时, <AAS>即被清除为"0"。

15.5.12 一般调用检测监控程序

在SBI作为从机运行时, 当其接收到一般调用地址时(亦即该启动条件后面的八位全部为零), SBIxSR<ADO>即被设置为"1"。

在母线上的启/停条件被检测到时, <ADO>即被清除为"0"。

15.5.13 最后接收位监控程序

SBIxSR<LRB>即被设置为SCL线路上升处的该SDA线路值。

在确认模式下, 在INTSBIx中断请求生成之后即读取SBIxSR<LRB>, 可导致ACK信号被读取。

15.5.14 数据缓冲寄存器 (SBIxDBR)

通过读取或写入SBIxDBR, 可引发读取所接收数据或写入已发送数据。

在SBI作为主机运行时, 通过将某个从机地址和某个方向位设置到该寄存器, 即可生成该启动条件。

15.5.15 波特率寄存器 (SBIxBR0)

该SBIxBR0<I2SBI>寄存器可决定在其进入IDEL模式时，SBI是否运行。

在执行一个指令以切换到待机模式之前，必须对该寄存器进行编程。

15.5.16 软件复位

如果该串行总线接口电路因外部噪声而锁定，则通过使用软件复位实现其初始化。

如果该串行总线接口电路因外部噪声而锁定，则通过使用软件复位实现其初始化。通过将"10" (其后带有"01")写入到SBIxCR2<SWRST[1:0]>，即可生成一个可初始化该串行总线接口电路的复位信号。在写入时，将<SBIM[1:0]>设置为"10"；I2C总线模式。在复位之后，所有控制寄存器和状态标志均会被初始化为其复位值。在该串行总线接口被初始化时，<SWRST [1:0]>即被自动清除为"0"。

注：软件复位可导致SBI操作模式从I2C模式切换到端口模式。

15.6 I2C总线模式2C下的数据传输程序

15.6.1 设备初始化

首先, 对 $SBIxCR1<ACK, SCK[2:0]>$ 进行编程。此时, 将 "000" 写入到 $SBIxCR1<BC[2:0]>$ 。

然后, 对 $SBIxI2CAR$ 进行编程, 方法是在 $<SA[6:0]>$ 指定一个从机地址, 并在 $<ALS>$ 指定地址识别模式(在使用寻址格式时, 必须将 $<ALS>$ 清除为 "0")。

在将该串行总线接口配置为从属接收器时, 首先需确保该串行总线接口引脚处于 "高" 状态。然后, 将 "0" 写入到 $SBIxCR2<MST, TRX, BB>$, 将 "1" 写入到 $<PIN>$, 将 "10" 写入到 $<SBIM[1:0]>$ 并将 "0" 写入到位1和位0。

注: 必须在所有被连接到该总线的设备被初始化之后, 任何设备均未生成启动条件的时段内, 完成该串行总线接口电路的初始化。如果不遵循该规则, 则可能不会正确的接收数据, 原因是其它设备可能在该串行总线接口电路的初始化完成之前开始传输。

		7	6	5	4	3	2	1	0	
$SBIxCR1$	←	0	0	0	X	0	X	X	X	指定ACK和SCL时钟。
$SBIxI2CAR$	←	X	X	X	X	X	X	X	X	指定一个从机地址和一种地址识别模式。
$SBIxCR2$	←	0	0	0	1	1	0	0	0	将该SBI配置为从属接收器。

注: X; 忽略。

15.6.2 启动条件和从机地址的生成

15.6.2.1 主模式

在主模式下, 必须执行以下步骤, 以生成该启动条件和一个从机地址。

首先, 确保总线处于空闲状态 ($<BB> = "0"$)。然后, 将 "1" 写入到 $SBIxCR1<ACK>$, 以选择确认模式。将拟发送的一个从机地址和一个方向位写入到 $SBIxDBR$ 。

在 $<BB> = "0"$ 时, 通过将 "1111" 写入到 $SBIxCR2<MST, TRX, BB, PIN>$, 在该母线上生成该启动条件。在该启动条件之后, SBI生成从SCL引脚生成九个时钟。SBI输出在 $SBIxDBR$ 中指定的该从机地址和方向位, 以及前八个时钟, 并解除第九个时钟中的SDA线路, 以接收一个来自该从机的确认信号。

在第九时钟的下降缘生成INTSBIx中断请求, $<PIN>$ 即被清除为 "0"。在主模式下, SBI将SCL线路保持在 "低" 电平, 同时 $<PIN> = "0"$ 。 $<TRX>$ 会在该INTSBIx中断请求生成时按发送方向改变其值, 但前提是已从该从机返回一个确认信号。

注: 在输出从机地址时, 需在写入到 $SBIxDBR$ 之前, 用软件检查并确认总线处于空闲状态。如果不遵循该规则, 则正在该总线上输出的数据可能会被破坏。

主程序中的设置

		7	6	5	4	3	2	1	0	
Reg.	←	SBIxSR								
Reg.	←	Reg. e 0x20								
如果Reg.	#	0x00								确保该总线处于空闲状态。
则										
SBIxCR1	←	X	X	X	1	0	X	X	X	选择该确认模式。
SBIxDBR	←	X	X	X	X	X	X	X	X	指定所需的从机地址和方向。
SBIxCR2	←	1	1	1	1	1	0	0	0	生成启动条件。

INTSBI0 中断程序示例

解除该中断请求。
处理
中断结束

15.6.2.2 从属模式

在从属模式下，SBI接收到该启动条件和一个从机地址。

在从该主机接收到启动条件之后，SBI在SCL线路上的前八个时钟期间，从主机接收到一个从机地址和一个方向位。

如果所接收的地址可匹配其在SBIxI2CAR指定的从机地址，或等于一般调用地址，则SBI会在第九个时钟期间将该SDA线路拖至"低"电平，并输出一个确认信号。

在第九时钟的下降缘生成INTSBIx中断请求，<PIN>即被清除为"0"。在从属模式下，SBI会将SCL线路保持在"低"电平，同时<PIN>为"0"。

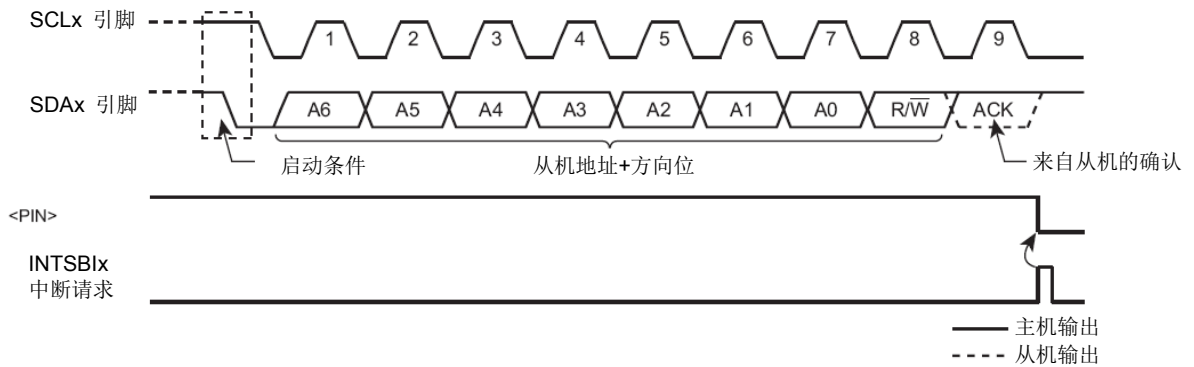


图 15-9 启动条件和从机地址的生成

15.6.3 数据字的传输

在数据字传输结束时，INTSBI_x中断即被生成，以测试<MST>并确定SBI是否处于主机或从机模式。

15.6.3.1 主模式 (<MST> = "1")

测试<TRX>以确定SBI是否被配置为发送器或接收器。

(1) 发送器模式(<TRX> = "1")

测试<LRB>。如果<LRB>为"1"，则表示接收器不要求提供更多的数据。

然后，主机生成后文所述的停止条件，以停止发送。

如果<LRB>为"0"，则表示接收器要求提供更多的数据。如果下一步拟发送的数据具备八位，则该数据会被写入到SBI_xDBR。如果该数据具备不同的长度，则<BC[2:0]>和<ACK>会被编程，且发送数据会被写入到SBI_xDBR。该数据的写入可使得<PIN>变为"1"，导致SCL引脚生成一个下一数据字传输用串行时钟，并导致SDA引脚传输该数据字。

在传输完成之后，该INTSBI_x中断请求即被生成，<PIN>即被清除为"0"，且SCL引脚被拖至"低"电平。

在发送更多的数据字时，需重新测试<LRB>，并重复以上程序。

INTSBI_x中断

如果MST = 0

则转到从机模式处理。

如果TRX = 0

则转到接收器模式处理。

如果LRB = 0

则转到停止条件生成处理。

SBI_xCR1 ← X X X X 0 X X X

指定拟发送的位的数目，并指定是否需要ACK。

SBI_xDBR ← X X X X X X X X

写入该发送数据。

中断处理结束。

注：X; 忽略。

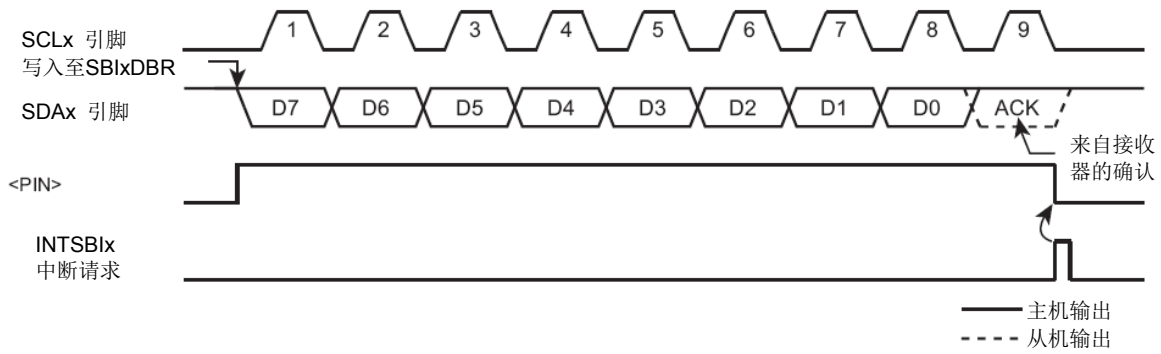


图 15-10 <BC[2:0]>= "000", <ACK>= "1" (发送器模式)

(2) 接收器模式(<TRX> = "0")

如果下一步拟发送的数据具备八位，则该发送数据会被写入到SBIxDBR。

如果该数据具备不同的长度，则<BC[2:0]>和<ACK>会被编程，且接收数据会被从SBIxDBR读取，以解除该SCL线路(在从机地址发送之后即被读取的数据未定义)。在读取该数据时，<PIN>即被设置为"1"，串行时钟被输出至SCL引脚，以传输下一个数据字。在最后的位中，在确认信号变为"低"电平时，"0"即被输出到SDA引脚。

然后，INTSBIx中断请求即被生成，<PIN>即被清除为"0"，并将SCL引脚拖至"低"电平。每从SBIxDBR读取一次已接收数据，就会有一字传输时钟和一个确认信号被输出。

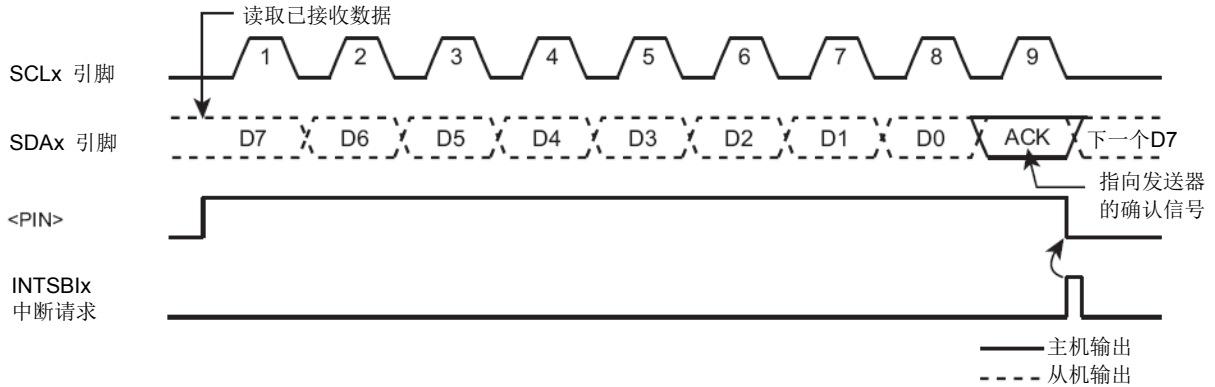


图 15-11 <BC[2:0]>= "000", <ACK>= "1" (接收器模式)

在终止从发送器进行的数据发送时，必须在读取第二个到最后一个数据字之前，将<ACK>清除为"0"。

这样可禁止生成最后数据字的确认时钟。

在传输完成时，一个中断请求即被生成。在中断处理完成之后，必须将<BC[2:0]>设置为"001"，并必须读取该数据，以确保生成 1-位传输的时钟。

此时，主接收器会将SDA总线保持在"高"电平，其会将发送器传输的结束作为确认信号发出。

在 1-位数据接收终止的中断处理过程中，该停止条件即被生成，用以终止数据传输。

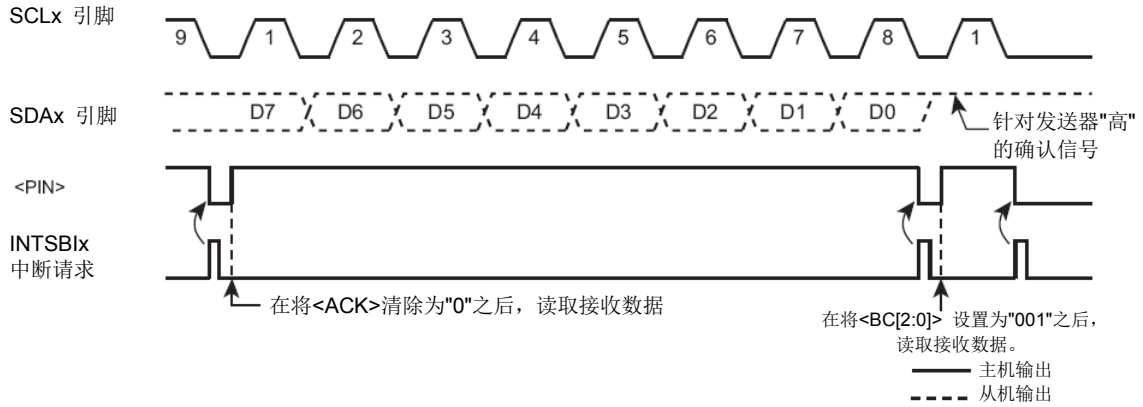


图 15-12 在主接收器模式下终止数据发送

示例：在接收N数据字时

INTSBIx 中断 (在数据发送之后)

		7	6	5	4	3	2	1	0	
SBIxCR1	←	X	X	X	0	X	X	X	X	设置拟接收的数据位的数目，并指定是否需要ACK。
Reg.	←	SBIxDBR								读取虚拟数据。
中断结束										

INTSBIx 中断 (第一个至第 (N-2)个数据接收)

		7	6	5	4	3	2	1	0	
Reg.	←	SBIxDBR								读取第一个至第(N-2)个数据字。
中断结束										

INTSBIx中断(第(N-1)个数据接收)

		7	6	5	4	3	2	1	0	
SBIxCR1	←	X	X	X	0	0	X	X	X	禁止确认时钟的生成。
Reg.	←	SBIxDBR								读取第(N-1)个数据字。
中断结束										

INTSBIx 中断 (第N个数据接收)

		7	6	5	4	3	2	1	0	
SBIxCR1	←	X	X	X	0	0	X	X	X	禁止确认时钟的生成。
Reg.	←	SBIxDBR								读取第N个数据字。
中断结束										

INTSBIx 中断 (在完成数据接收之后)

进行处理，以生成该停止条件。 终止数据发送。
中断结束

注：X; 忽略。

15.6.3.2 从属模式 (<MST> = "0")

在从属模式下，SBI会在四个时间点生成INTSBIx中断请求：

- 1) 在SBI已从主机接收到任何从机地址时。
- 2) 在SBI已接收到一个一般调用地址时。
- 3) 在所接收的从机地址可匹配其地址时。
- 4) 在应一般调用的要求完成一次数据传输时。

此外，如果在主模式SBI检测到判优丢失，则其会切换至从属模式。

一旦完成被检测出存在判优丢失现象的数据字传输，INTSBIx中断请求即被生成，<PIN>即被清除为"0"，且SCL引脚即被拖至"低"电平。

在数据被写入到SBIxDBR或被从SBIxDBR读取时，或在<PIN>被设置为"1"时，SCLx 引脚会在一段时间t_{Low}之后被解除。

在从属模式下，可执行正常从属模式处理或因判优而需进行的处理。

会对SBIxSR<AL>，<TRX>，<AAS>和<ADO>进行测试，以确定所需进行的处理。

"表 15-2 在从属模式下的处理"给出了从属模式状态和所需的处理。

示例：在从属接收器模式下，所接收的从机地址可匹配SBI的自身地址，且方向位为"1"时。

INTSBIx中断

如果TRX = 0

则转到其它处理。

如果AL = 0

则转到其它处理。

如果AAS = 0

则转到其它处理。

SBIxCR1	←	X	X	X	1	0	X	X	X	设置拟发送的位的数目。
SBIxDBR	←	X	X	X	X	X	X	X	X	设置该发送数据。

注：X; 忽略。

表 15-2 在从属模式下进行的处理

<TRX>	<AL>	<AAS>	<ADO>	状态	处理
1	1	1	0	在从机地址正被发送, SBI接收到发自其它主机的从机地址连同方向位"1"的同时, 判优丢失被检测到。	将某个数据字中的位的数目设置为<BC[2:0]>, 并将发送数据写入到SBIxDBR。
		1	0	在从机接收器模式下, SBI接收到发自该主机的一个从机地址连同方向位"1"。	
	0	0	0	在从机发送器模式下, SBI已完成一个数据字的发送。	测试LRB。如果其已被设置为"1", 则表示该接收器不要求提供更多的数据。将<PIN>设置为1, 并将<TRX>复位为0, 以解除该总线。如果<LRB>已被复位为"0", 则表示该接收器要求提供更多的数据。将某个数据字中的位的数目设置为<BC[2:0]>, 并将发送数据写入到SBIxDBR。
0	1	1	1/0	在从机地址正被发送, SBI接收到发自其它主机的从机地址连同方向位"0"或一个一般调用地址的同时, 判优丢失被检测到。	读取SBIxDBR (虚拟读取) 以将 <PIN>设置为 1, 或将"1"写入到<PIN>。
		0	0	在从机地址或数据字正在被发送的同时, 判优丢失被检测到, 该传输即被终止。	
	0	1	1/0	在从机接收器模式下, SBI接收到发自该主机的一个从机地址连同方向位"0"或一个一般调用地址。	
		0	1/0	在从机接收器模式下, SBI已完成一个数据字的接收。	

15.6.4 停止条件的生成

在SBIxSR<BB>为"1"时，将"1"写入到SBIxCR2<MST, TRX, PIN>，且将"0"写入到<BB>，可导致SBI在该总线上启动一个停止条件生成序列。

在该停止条件出现在该总线上之前，不要改动<MST, TRX, BB, PIN>的内容。

如果另一设备限制了该SCL总线，则SBI需等待至SCL线路被解除。

然后，SDA引脚会变为"高"，停止条件即被生成。

```

          7  6  5  4  3  2  1  0
SBIxCR2 ← 1  1  0  1  1  0  0  0    生成停止条件。
  
```

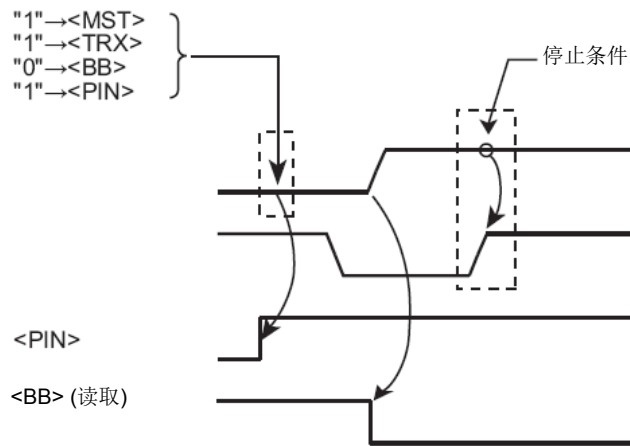


图 15-13 停止条件的生成

15.6.5 重新启动程序

在主机未终止对某个从机的传输的情况下改变数据传输方向时，就需要进行重新启动。在主模式生成重新启动的程序如以下所述。

首先，将SBIxCR2<MST, TRX, BB> 写入为"0"，并将"1"写入到<PIN>以解除该总线。此时，SDAx引脚被固定在"高"电平，SCLx引脚即被解除。由于该总线上未生成停止条件，因此，其它设备会将该总线识别为"占空"。

然后，测试SBIxSR<BB>并等待至其变为"0"，以确保该SCLx引脚被解除。

其次，测试<LRB>并等待至其变为"1"，以确保没有别的设备会将该SCLx总线拖至"低"电平。

一旦在执行以上程序之后确定该总线处于空闲状态，则应按照"15.6.2 启动条件和从机地址的生成"所述的程序，生成启动条件。

为满足重新启动的整置时间，在该总线被确定为空闲之后，必须利用软件创建至少为期 4.7 μ s 的等待期(在标准模式下)。

注 1: 在其"0"时，不要将<MST> 写为"0"。(重新启动无法启动)。

注 2: 在主机作为接收器工作时，必须在重新启动生成之前，完成从作为发送器工作的从机进行的数据发送。为完成数据传输，从机必须接收一个"高"电平确认信号。为此，在重新启动生成之前的<LRB>会变为"1"，即使在重新启动程序之后证实<LRB>="1"，SCL线路的上升缘也不会被检测到。

在重新启动程序以后"1"被确认。读取端口以检查 SCL 线的状态。

		7	6	5	4	3	2	1	0		
┌	SBIxCR2	←	0	0	0	1	1	0	0	0	解除该总线
└	如果SBIxSR<BB> ≠ 0										检查并确认SCL引脚已被解除。
┌	则										
└	如果SBIxSR<LRB> ≠ 1										检查并确认没有别的设备会将SCL引脚拖至"低"。
	则										
	4.7 μs等待										
	SBIxCR1	←	X	X	X	1	0	X	X	X	选择该确认模式。
	SBIxDBR	←	X	X	X	X	X	X	X	X	设置所需的从机地址和方向。
	SBIxCR2	←	1	1	1	1	1	0	0	0	生成启动条件。

注: X; 忽略。

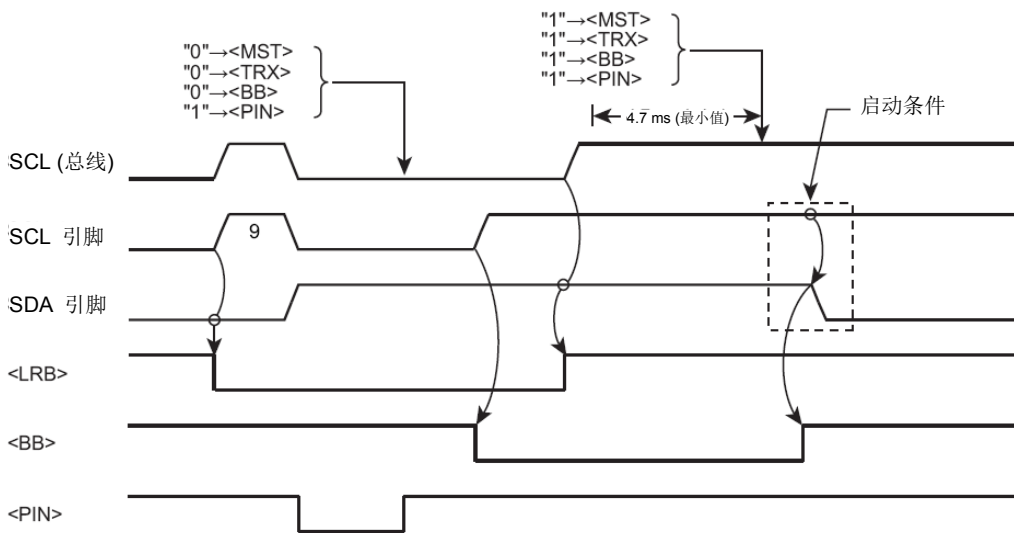


图 15-14 重新启动生成的时序图

15.6.6 利用DMA进行数据传输

在I2C模式下，可利用DMA传输数据。但需将一个主机和一个从机连接到一条总线上，并应预先确定传输数据的数目。

在I2C模式下，DMA请求与INTSBI_x请求同时发生，因此，通过在SBI_xDBR(数据缓冲器)和存储器之间进行DMA传输，即可连续传送数据。

DMA发送和接收请求需单独执行。

在使用DMA之前，需将发送和接收用DMAC控制寄存器设置为启用或禁用。

在用DMA传输数据时，应预先就发送和接收指定拟传输字节的数目。

在接收模式下，应用软件读取最后接收的数据，以及从第二个到最后一个被接收的数据。

如果在I2C传输期间发生判优丢失，则会发生INTABTLOSS_x中断和INTSBI_x中断，通信随即停止。DMA请求未发生。

注：在SIO模式下，无法进行DMA传输。

表 15-3 DMA请求编号

通道	传输/接收	DMA请求编号
通道0	接收	12
	传输	13
通道0	接收	14
	传输	15

15.6.6.1 如何在主模式下传输数据

1. 生成启动条件和从机地址。
2. 检查输出从机地址后面的INTSBIx中断服务程序中的<LRB>和<TRX>。
3. 在<LRB>为"1"时，输出一个停止条件并完成传输，原因是该总线上并不存在具备已发送从机地址的从设备。
4. 在<LRB>为"0"时，检查<TRX>，原因是该总线上存在具备已发送从机地址的从设备。该进程取决于<TRX>。

(1) 在<TRX>为"1" (传送器模式) 时

- a. 设置传输位宽度，裂解量和总传输数目等DMA。从拟发送数据的数目减去 1。启用以接收INTSBI的DMA请求。
- b. 将第一个数据写入到SBIDBRx。通过该写入启动发送第一个数据。
- c. 利用已完成传输后面的INTSBIx的DMA请求，启动DMA传输。发送第2个数据及后续数据。
- d. 在指定的DMA传输数目(N-1次)结束时，发生DMA传输结束中断。改变该设置，即可忽略中断服务程序中的DMA请求接收(在INTSBIx中断发生之前配置该设置，即可不生成DMA请求)
- e. 生成停止条件，以在INTSBIx中断服务程序停止发送。

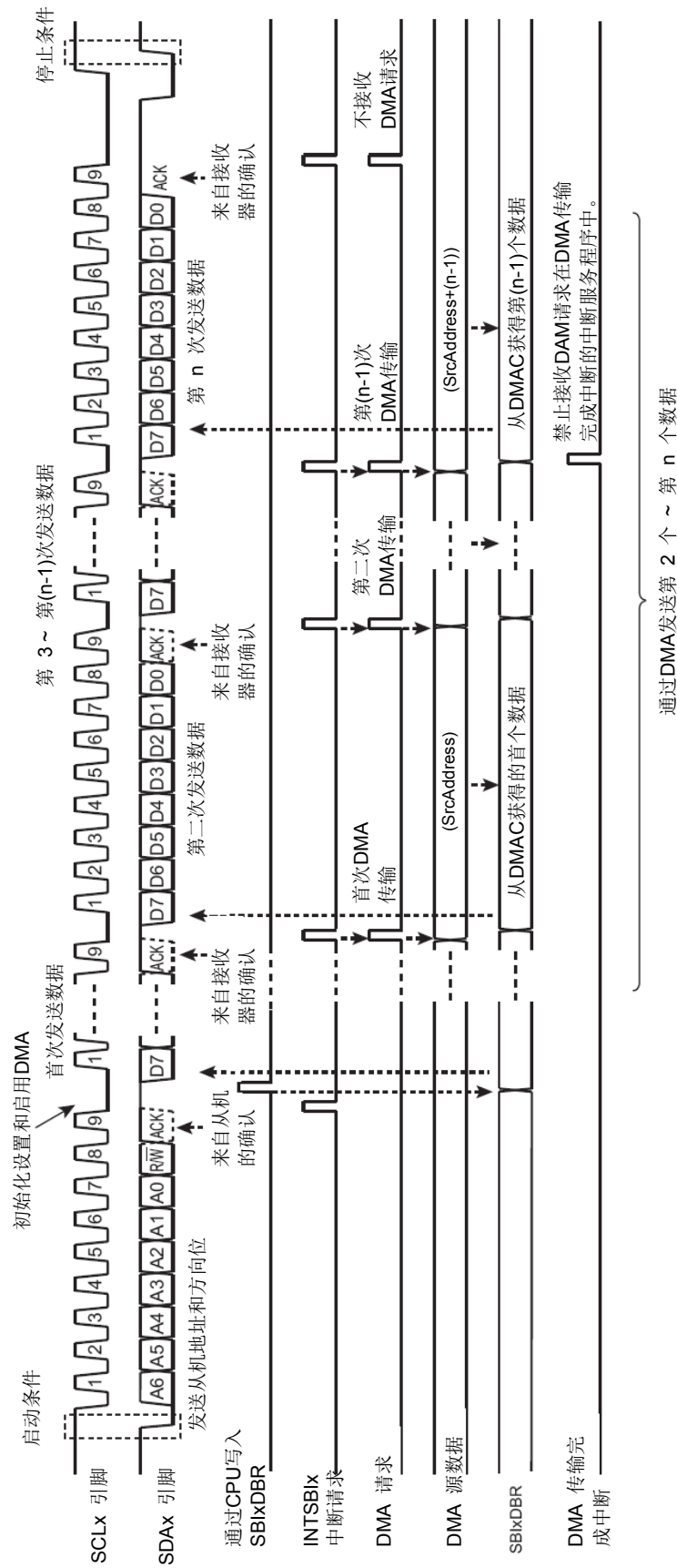


图 15-15 主机发送模式

- (2) 在<TRX>为"0"(接收器模式)时
- a. 设置传输位宽度，裂解量和总传输数目等DMA。从拟发送数据的数目减去 2。启用以接收INTSBI的DMA请求。
 - b. 读取来自SBIDBR_x的虚拟数据。通过该读取开始接收第一批数据。
 - c. 利用已完成接收后面的INTSBI_x的DMA请求，启动DMA传输。然后接收第二批数据和后续数据。
 - d. 在指定的DMA传输数目(N-2次)结束时，发生DMA传输结束中断。改变该设置，即可忽略中断服务程序中的DMA请求接收(在INTSBI_x中断发生之前配置该设置，即可不生成DMA请求)
 - e. 清除<ACK>，以免在INTSBI_x的中断服务程序生成确认时钟。自SBIDBR_x读取接受数据。
 - f. 将<BC>设置为"001"，并在INTSBI_x的中断服务程序读取已接收的数据。通过该设置，可接收具备高电平SDA的一位数据。发送器接收到一个否定应答。
 - g. 生成停止条件，以在INTSBI_x的中断服务程序停止发送。

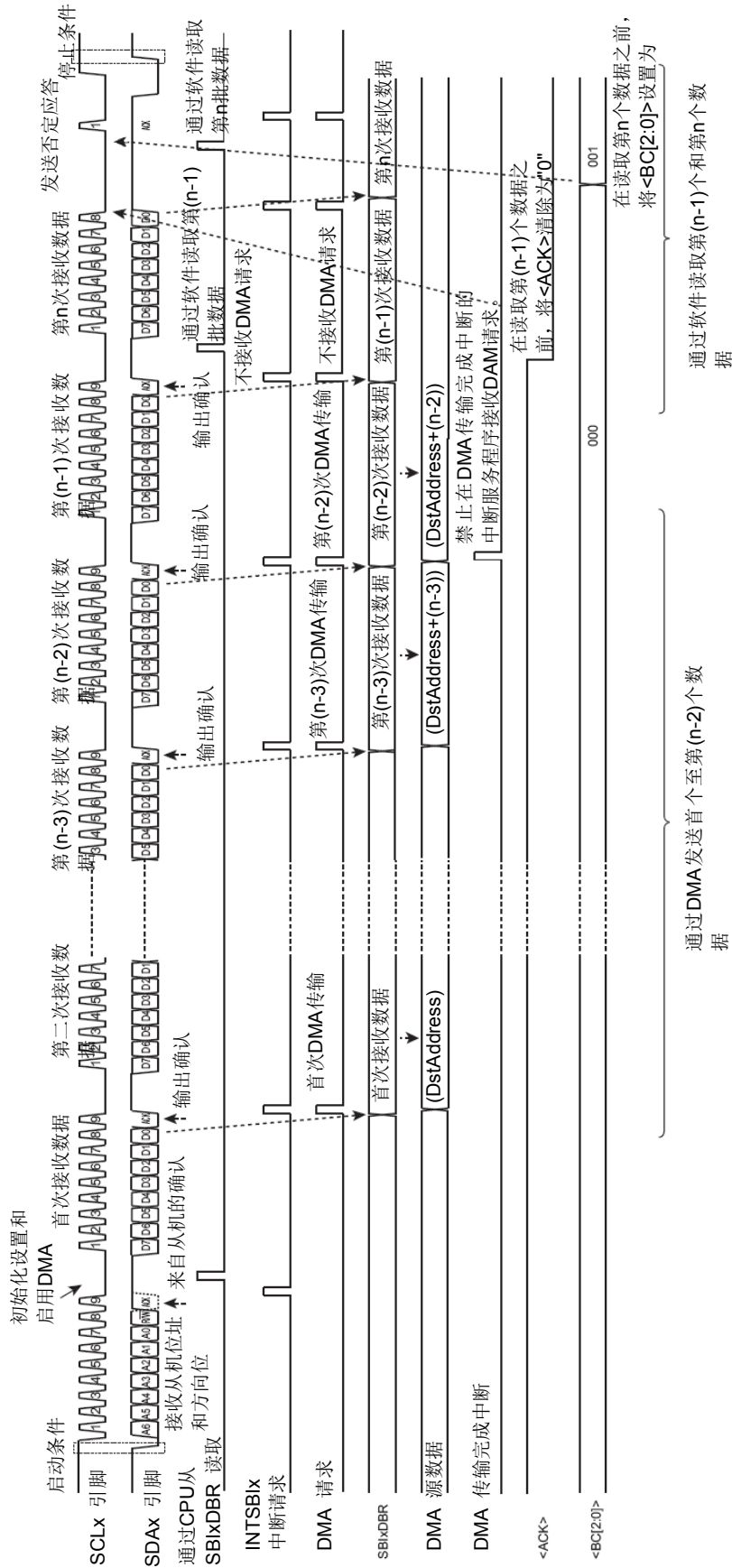


图 15-16 主接收器模式

15.6.6.2 如何在从属模式下传输数据

1. 接收启动条件和从机地址。
2. 检查接收从机地址后面的INTSBIx的中断服务程序中的<TRX>。
3. 检查<TRX>。该进程取决于<TRX>。

(1) 在<TRX>为"1" (传送器模式)时

- a. 设置传输位宽度, 裂解量和总传输数目等DMA。从拟发送数据的数目减去 1。启用以接收INTSBI的DMA请求。
- b. 将第一个数据写入到SBIDBRx。TMPM365FYXBG可从主机接收一个时钟, 并通过该写入开始发送第一个数据。
- c. 利用已完成传输后面的INTSBIx的DMA请求, 启动DMA传输。发送第2个数据及后续数据。
- d. 在指定的DMA传输数目(N-1次)结束时, 发生DMA传输结束中断。改变该设置, 即可忽略中断服务程序中的DMA请求接收(在INTSBIx中断发生之前配置该设置, 即可不生成DMA请求)
- e. 等待来自主机的停止条件, 且不将数据写入到INTSBIx的服务程序。

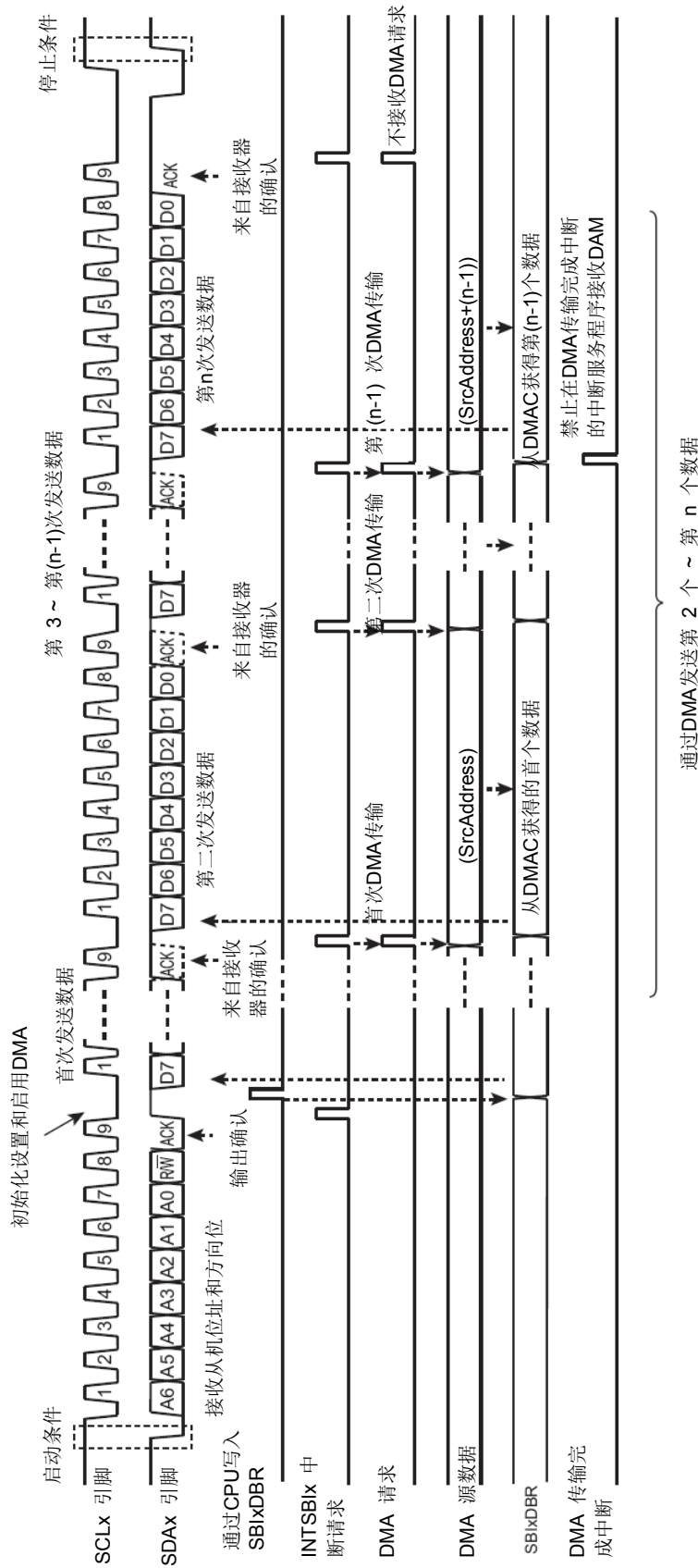
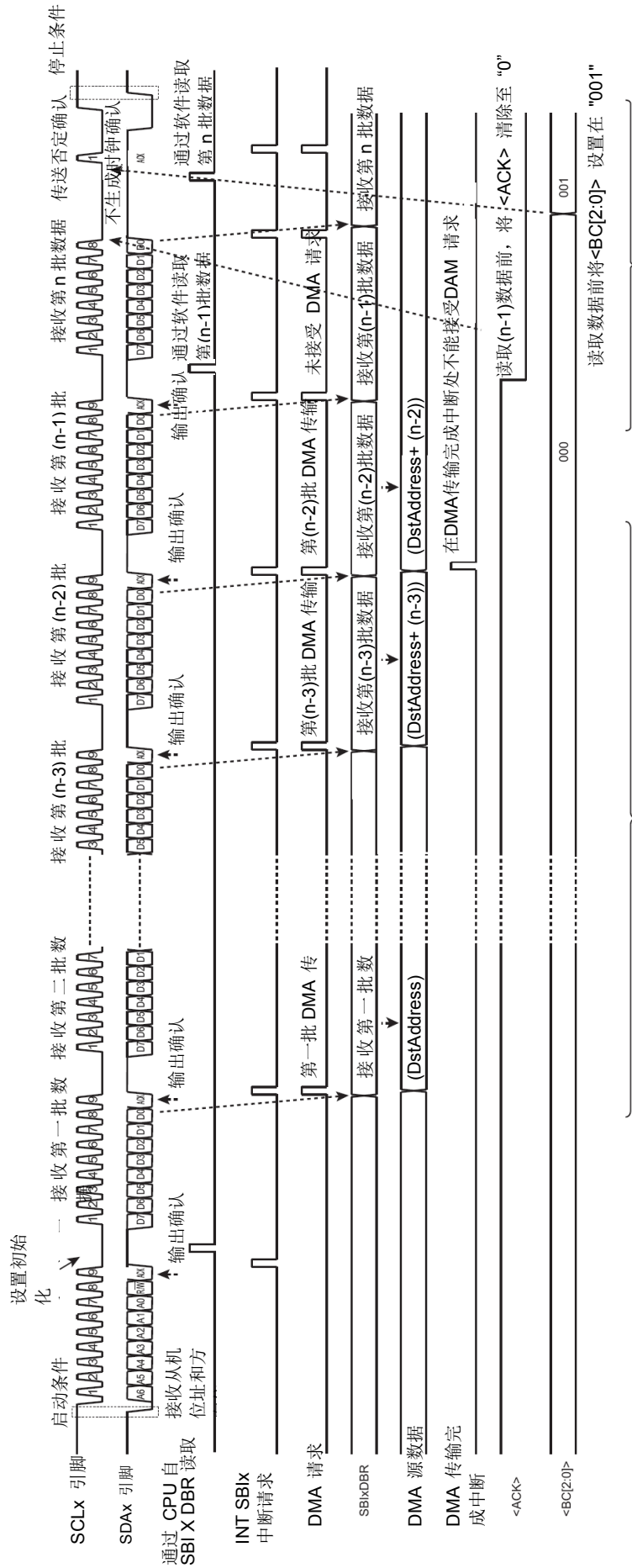


图 15-17 从机发送模式

(2) 当<TRX> 为"0"时(接收器模式)

- a. 设置 DMA，如传输位宽，裂解量和总传输数目等。自数据数目提取 2 以作传送。并启用接收 DMA INTSBI 请求。
- b. 自 SBIDBR_x 读取虚拟数据.通过该读取开始接收第一批数据。
- c. 完成接收后，通过 INTSBI_x 的 DMA 请求开始 DMA 传输。然后接收第二批数据和后续数据。
- d. 指定 DMA 传输数目(N-2 次)结束时，DMA 传输结束中断发生。改变设置，忽略该中断服务程序中的 DMA 接收请求。(设置配置为 INTSBI_x 中断发生前不生成 DMA 请求。)
- e. 清除 <ACK>，INTSBI_x 中断服务程序不生成确认时钟。自 SBIDBR_x 读取接受数据。
- f. 将 <BC> 设置为 "001" ， INTSBI_x 中断服务程序读取接受数据。通过该设置接收高级 SDA 一位数据。发送器接收否定确认。
- g. INTSBI_x 服务程序不写入数据，等待主设备停止条件。



通过 DMA 将其传输至 (n-2) th 数据

通过软件读取第(n-1) 和 第 n 数据

图 15-18 从机接收模式

15.7 SIO模式控制寄存器

以下寄存器控制时钟同步 8-位SIO模式串行总线接口，提供其状态信息，以便监控。

15.7.1 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	类型	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 使用串行总线接口前启用该位。 若该位禁用，由于除SBIXCR0外所有时钟停止，可减少功耗。 若串行总线接口操作启用后再禁用，设置仍然保留于各寄存器。
6-0	-	R	读作"0"。

15.7.2 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	读作 0。																								
7	SIOS	R/W	传输开始/停止 0: 停止 1: 开始																								
6	SIOINH	R/W	传输 0: 继续 1: 强制终止																								
5-4	SIOM[1:0]	R/W	选择传输模式 00: 传输模式 01: 保留 10: 传输/接收模式 11: 接收模式																								
3	-	R	读作 1。																								
2-0	SCK[2:0]	R/W	开启写入<SCK[2:0]>: 选择串行时钟频率.(注1) <table border="1" style="margin-left: 20px;"> <tr> <td>000</td> <td>n = 3</td> <td>3 MHz</td> </tr> <tr> <td>001</td> <td>n = 4</td> <td>1.5 MHz</td> </tr> <tr> <td>010</td> <td>n = 5</td> <td>750 k Hz</td> </tr> <tr> <td>011</td> <td>n = 6</td> <td>375 kHz</td> </tr> <tr> <td>100</td> <td>n = 7</td> <td>187.5 kHz</td> </tr> <tr> <td>101</td> <td>n = 8</td> <td>93.8 kHz</td> </tr> <tr> <td>110</td> <td>n = 9</td> <td>46.9 kHz</td> </tr> <tr> <td>111</td> <td>-</td> <td>外部时钟</td> </tr> </table> <div style="margin-left: 40px;"> $\left. \begin{array}{l} \text{系统时钟: } f_{\text{sys}} \\ \text{时钟齿轮: } fc/1 \text{ (} = 48\text{MHz) } \\ \text{频率} = \frac{f_{\text{sys}}/2}{2^n} \text{ [Hz]} \end{array} \right\}$ </div>	000	n = 3	3 MHz	001	n = 4	1.5 MHz	010	n = 5	750 k Hz	011	n = 6	375 kHz	100	n = 7	187.5 kHz	101	n = 8	93.8 kHz	110	n = 9	46.9 kHz	111	-	外部时钟
000	n = 3	3 MHz																									
001	n = 4	1.5 MHz																									
010	n = 5	750 k Hz																									
011	n = 6	375 kHz																									
100	n = 7	187.5 kHz																									
101	n = 8	93.8 kHz																									
110	n = 9	46.9 kHz																									
111	-	外部时钟																									

注 1: 复位后, <SCK[0]>位读作 "1"。但是, 若SIO模式选择于SBiXCR2寄存器, 首个数值读作"0"。本文中, "复位后"列所写数值为初始状态设置SIO模式后的数值。SBiXCR2 寄存器说明和SBiXSR寄存器说明相同。

注 2: 传输模式和序列时钟程序编制之前, 将 <SIOS> 设置为 "0" 且 <SIOINH>为 "1"。

注 3: 主模式中, 设置<BC[2:0]>="001" 且 <ACK>="1"时, 停止条件发生时, SCL线可固定于SCL线下降缘上的"L"。因此另一总线主设备无法使用该总线。若连接多台主设备至总线, 停止条件发生前, 将两个或更多设置为传输位数。

15.7.3 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	类型	功能
31-8	-	R	读作 0。
7-0	DB[7:0]	R	接受数据
		W	传送数据

注 1: 传输数据须自MSB (7位)写入寄存器。接受数据存于LSB。

注 2: 因 SBixDBR具独立读写缓冲, 写入数据无法读取。因此, 读写指令(如位处理)无法使用。

15.7.4 SBIxCR2 (控制寄存器 2)

通过向其读取，该寄存器起SBIxSR寄存器的作用。

	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	类型	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: 本文中, "复位后"列所写数值为初始状态设置SIO模式后的数值。

注 2: 通信会话时不得改变模式。

15.7.5 SBIXSR (状态寄存器)

通过向其写入，该寄存器起到SBIxCR2寄存器的作用。

	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(注)	1(注)	1(注)	0	0	1(注)	1(注)

位	比特符号	类型	功能
31-8	-	R	读作 0。
7-4	-	R	读作1。(注)
3.	SIOF	R	串行传送状态监控器。 0: 已完成 1: 进行中
2.	SEF	R	移位操作状态监控器 0: 已完成。 1: 进行中
1-0	-	R	读作1。(注)

注：本文中，“复位后”列所写数值为初始状态设置 SIO 模式后的数值。

15.7.6 SB_lxBR0 (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	类型	功能
31-8	-	R	读作 0。
7	-	R	读作 1。
6	I2SBI	R/W	IDLE模式操作。 0: 停止 1: 操作
5-1	-	R	读作 1。
0.	-	R/W	必须写"0"。

15.8 SIO模式控制

15.8.1 串行时钟

15.8.1.1 时钟源

通过对SBIxCR1<SCK[2:0]>进行编程，可选择内部或外部时钟。

(1) 内部时钟

内部时钟模式中，作为通过SCKx引脚向外输出的串行时钟，可选择七个频率之一。

传输开始时，SCKx引脚输出成为高电平。

若程序写入传输数据或读取接受数据时无法保持该串行时钟频率，SBI自动进入等待时间。在该时间段内，串行时钟自动停止，下一移位操作暂停，直至处理完成。

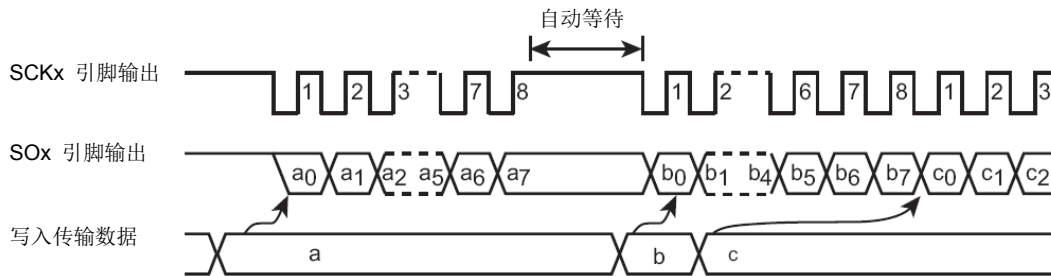


图 15-19 自动等待

(2) 外部时钟 (<SCK[2:0]> = "111")

SBI使用作为串行时钟自外供至SCKx 引脚的外部时钟。

正确移位操作，"高"和"低"电平的串行时钟须具备如下所示脉宽。

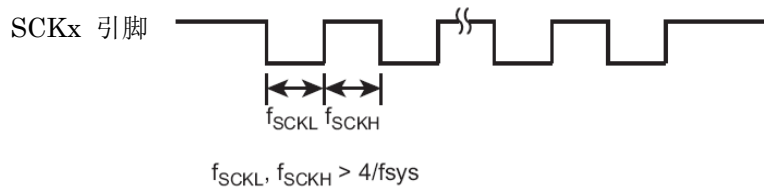


图 15-20 外部时钟输入最大传输频率

15.8.1.2 移位缘

前缘移位用于传送。后缘移位用于接收。

- 前缘移位
数据于串行时钟前缘（或SCKx引脚输入/输出下降缘）移位。
- 后缘移位
数据于串行时钟后缘（或SCKx引脚输入/输出上升缘）移位。

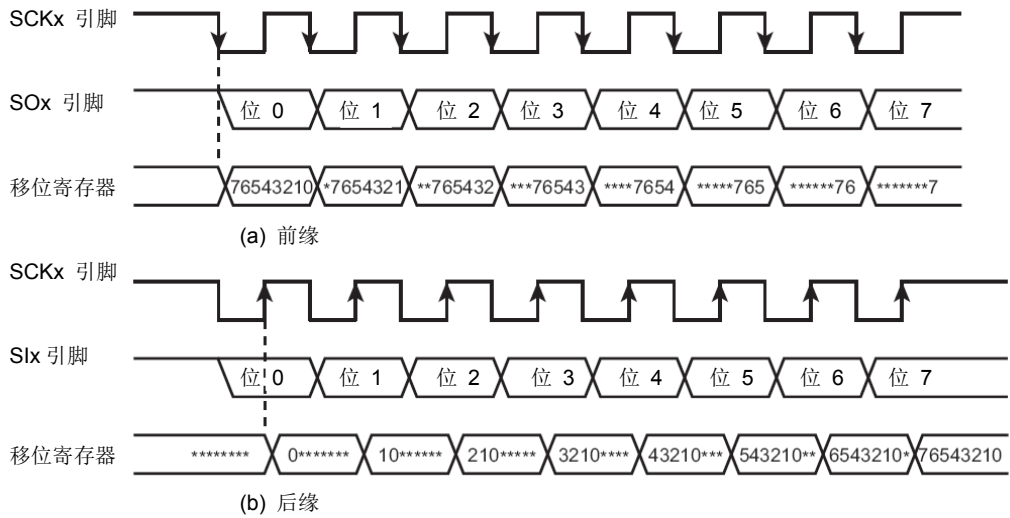


图 15-21 移位缘

15.8.2 传输模式

通过对SBIxCR1<SIOM[1:0]>进行编程，可选择传输模式，接收模式或传输/接收模式。

15.8.2.1 8-位传输模式

将控制寄存器设置为传输模式，将传输数据写入SBIxDBR。

写入传输数据后，往SBIxCR1<SIOS>写入"1"，开始传输。传输数据自SBIxDBR移往移位寄存器，输出至SO引脚，首先是最低有效位(LSB)，与串行时钟同步。传输数据传送至移位寄存器后，SBIxDBR为空，产生INTSBIx(缓冲-空)中断，请求下一传输数据。

内部时钟模式中，8位数据全部传输后，若加载下一数据，串行时钟将停止，自动进入等待状态。SBIxDBR加载下一传输数据时，等待状态将清除。

外部时钟模式中，SBIxDBR须在下一数据移位操作开始之前加载。因此，视乎产生中断请求时和SBIxDBR加载中断服务程序数据时之间的最大延迟，数据传送率各不相同。

传输开始时，前一传输数据最后位的同样数值于自将SBIxSR<SIOF>设置为"1"至SCK下降缘期间输出。

INTSBIx 中断服务程序中 将 <SIOS> 清至 "0" 或将 <SIOINH> 设置为 "1" 可终止传输。若<SIOS> 清除，传输结束前输出剩余数据。该程序检查SBIxSR<SIOF>，以判断是否传输已结束。

传输结束时，<SIOF>被清至"0"。若<SIOINH> 设置为"1"，传输立即终止，<SIOF>被清至"0"。

外部时钟模式时，下一数据移位前，<SIOS>须清至"0"。若<SIOS>在下一数据移位前未清至"0"，SBI输出虚拟数据并停止。

		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

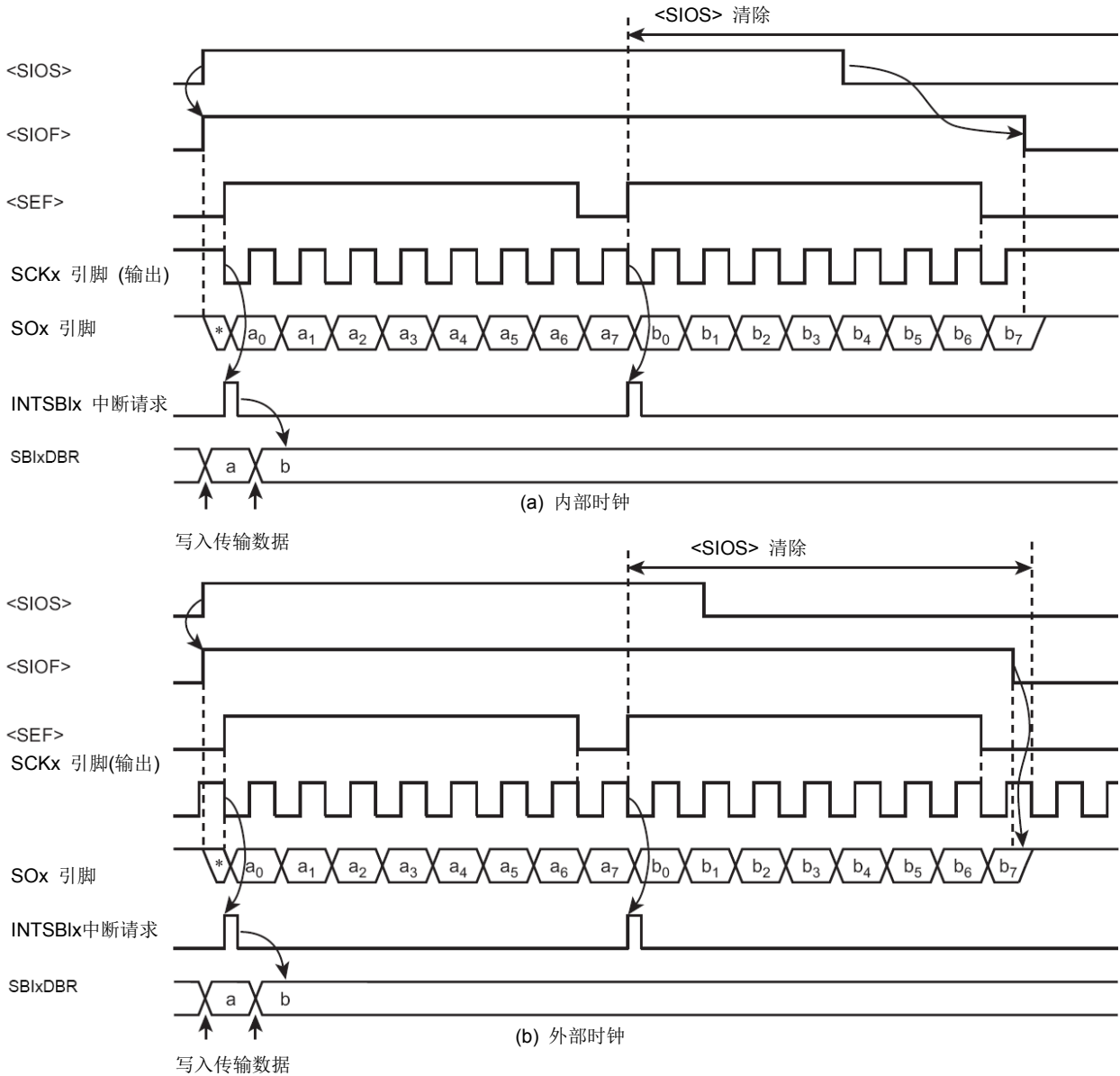


图 15-22 传输模式

- | | | |
|---|---------------------------|-------------------------|
| | 7 6 5 4 3 2 1 0 | |
| → | 若 SBIxSR<SIOF> ≠ 0 | 确认传输完成。 |
| → | 则 | |
| → | 若 SCK ≠ 1 | 通过监控端口，确认"1"设置为 SCK 引脚。 |
| → | 则 | |
| | SBIxCR1 ← 0 0 0 0 0 1 1 1 | 设置 <SIOS> = 0，完成传输。 |

15.8.2.2 8-位接收模式

将控制寄存器设置为接收模式。然后，"1"写入SBIxCR1<SIOS>启用接收。数据自SI引脚导入移位寄存器，首先是最低有效位(LSB)，与串行时钟同步。移位寄存器加载8-位后，传输接受数据至SBIxDBR，INTSBIx(缓冲-满)产生中断请求，请求读取接受数据。然后，中断服务程序自SBIxDBR读取接受数据。

内部时钟模式中，串行时钟将停止，自动处于等待状态，直至接受数据自SBIxDBR读取。

外部时钟模式中，和外部时钟同步执行移位操作。视乎产生中断请求时和读取接受数据之间的最大延迟，最大数据传输速率各不相同。

INTSBIx中断服务程序中将 <SIOS> 清至 "0" 或将 <SIOINH> 设置为 "1"，可终止接收。若 <SIOS> 清除，接收继续，直至所有接受数据位写入 SBIxDBR。该程序检查 SBIxSR<SIOF>，以判断是否接收已结束。接收结束时，<SIOF>被清至"0"。确认接收完成后，最后接受数据被读取。若<SIOINH>设置为"1"，接收立即终止，<SIOF>被清至"0"。(接受数据变为无效，无需读出。)

注：改变传输模式后，SBIxDBR内容将不保留。须将<SIOS>清至"0"才能完成进行中的接收，最后接受数据须在传输模式改变前读取。

		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

寄存器	←	SBIxDBR	读取接受数据.
-----	---	---------	---------

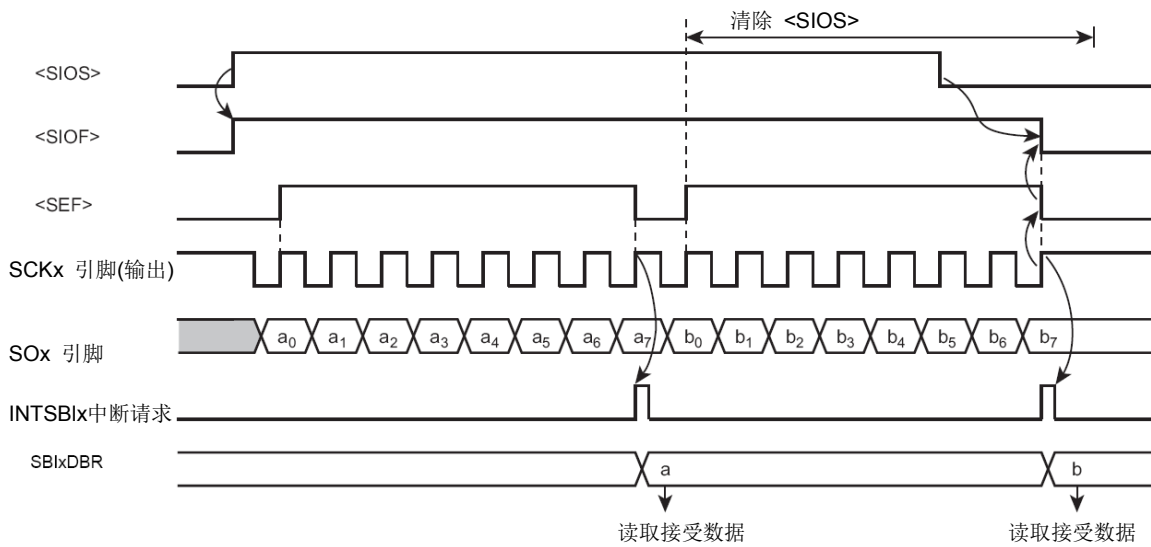


图 15-23 接收模式 (示例: 内部时钟)

15.8.2.3 8-位传输/接收模式

将控制寄存器设置为传输/接收模式。然后，将传输数据写入SBIxDBR，将SBIxCR1<SIOS>设置为"1"启用传输和接收。串行时钟下降时通过SOx引脚输出传输数据，串行时钟上升时通过SI引脚导入接受数据，首先是最低有效位(LSB)。移位寄存器加载8位数据后，将接受数据转至SBIxDBR，产生INTSBIx中断请求。中断服务程序自数据缓冲寄存器读取接受数据，写入下一传输数据。因SBIxDBR共享于传输和接收操作之间，接受数据须在下一传输数据写入之前被读取。

内部时钟操作中，串行时钟将自动处于等待状态，直至接受数据被读取，下一传输数据被写入。

外部时钟操作中，移位操作和外部串行时钟同步执行。因此，在下一移位操作开始之前，接受数据须被读取，下一传输数据须被写入。

视乎中断请求产生时和传输数据写入时之间的最大延迟，外部时钟操作的最大数据传输速率各不相同。

传输开始时，前一传输数据最后位的同样数值于自将<SIOF>设置为"1"至SCK下降缘期间输出。

INTSBIx中断服务程序中，将<SIOS>清至"0" 或将SBIxCR1<SIOINH> 设置为"1"，可终止传输和接收。。若<SIOS>清除，传输和接收继续，直至接受数据完全传输至SBIxDBR。该程序将检查SBIxSR<SIOF>，以判断传输和接收是否已结束。传输和接收结束时，<SIOF>被清至"0"。若 <SIOINH>设置为"1"，传输和接收立即终止，<SIOF>被清至"0"。

注：传输模式改变后，SBIxDBR内容将不保留。须将<SIOS>设置为"0" 才能完成进行中的传输和接收，最后接受数据 须在改变传输模式前被读取。

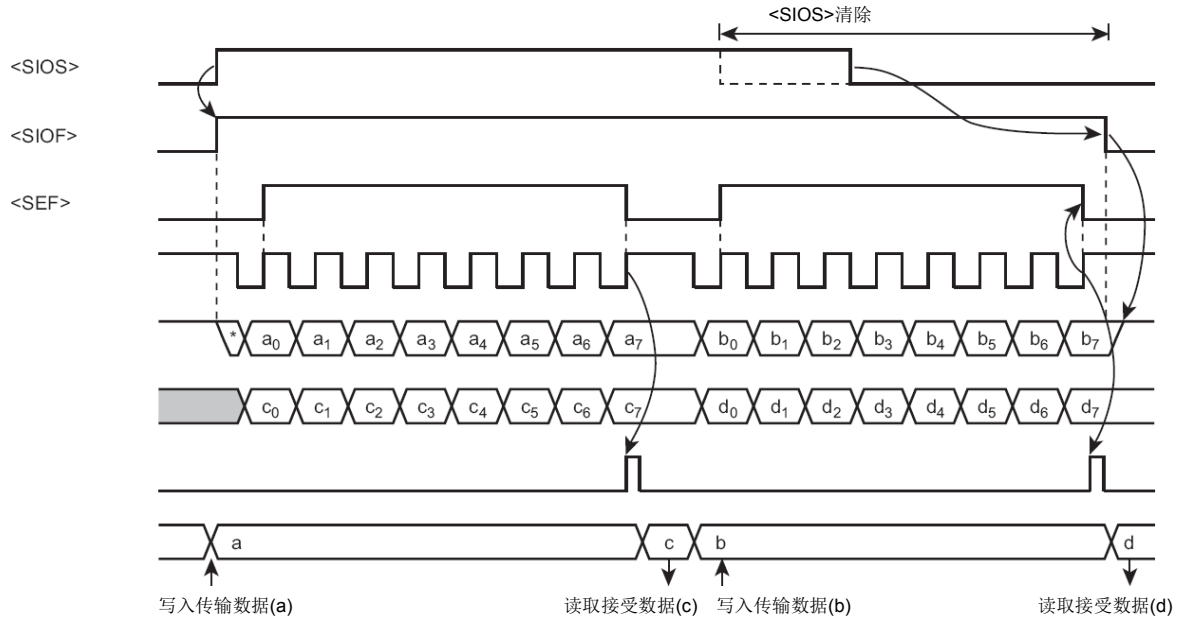


图 15-24 传输/接收模式 (示例: 内部时钟)

		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											
寄存器	←	SBIxDBR									读取接受数据.
SBIxDBR	←	X	X	X	X	X	X	X	X	写入传输数据.	

15.8.2.4 传输结束时最后位的数据保留时间

SBIxCR1<SIOS>= "0"的情况下, 传输数据的最后位如以下所示保留SCK上升缘数据。传输模式和传输/接收模式相同。

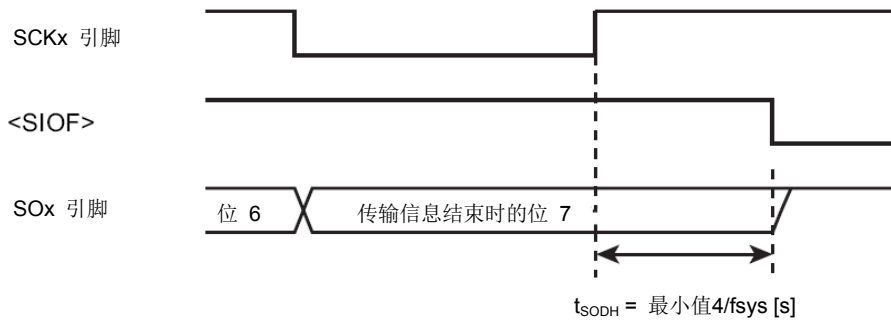


图 15-25 传输结束时最后位的数据保留时间

16. 模拟/数字转换器 (ADC)

16.1 概述

TMPM365FYXBG包括 12-位序列转换模拟/数字转换器(ADC)，配 12 位模拟量输入通道。这些 12 位模拟量输入通道（引脚 AIN00~ AIN11)也用于输入/输出端口。

该 12-位AD 转换器具备以下特性：

- 开始普通AD转换和最高优先级AD转换
 - 软件激活
 - 采用 16-位定时器 (TMRB)激活
 - 采用外部触发器输入 ($\overline{\text{ADTRG}}$ 引脚) 硬件激活
- AD转换
 - 固定通道单次转换模式
 - 通道扫描单次转换模式
 - 固定通道重复转换模式
 - 通道扫描重复转换模式
- 最高优先级AD转换
- 普通AD转换完成中断和最高优先级AD转换完成中断
- 普通AD转换和最高优先级AD转换具备以下状态标志。
 - 表示AD转换结果数据有效的标志<ADR_xRF>和表示AD转换结果数据被覆盖的标志<OVR_x>
 - 普通AD转换完成标志和最高优先级AD转换完成标志
 - 普通AD转换占空标志和最高优先级AD转换占空标志
- AD监控功能
 - AD 监控功能启用时，若比较结果匹配，则生成中断。
- 从1/fc到1/16fcAD转换时钟可控制。
- AD转换完成时，支持两类DMA请求。
- 支持待机模式。

16.2 配置

图 16-1 展示了AD转换器的方块图。

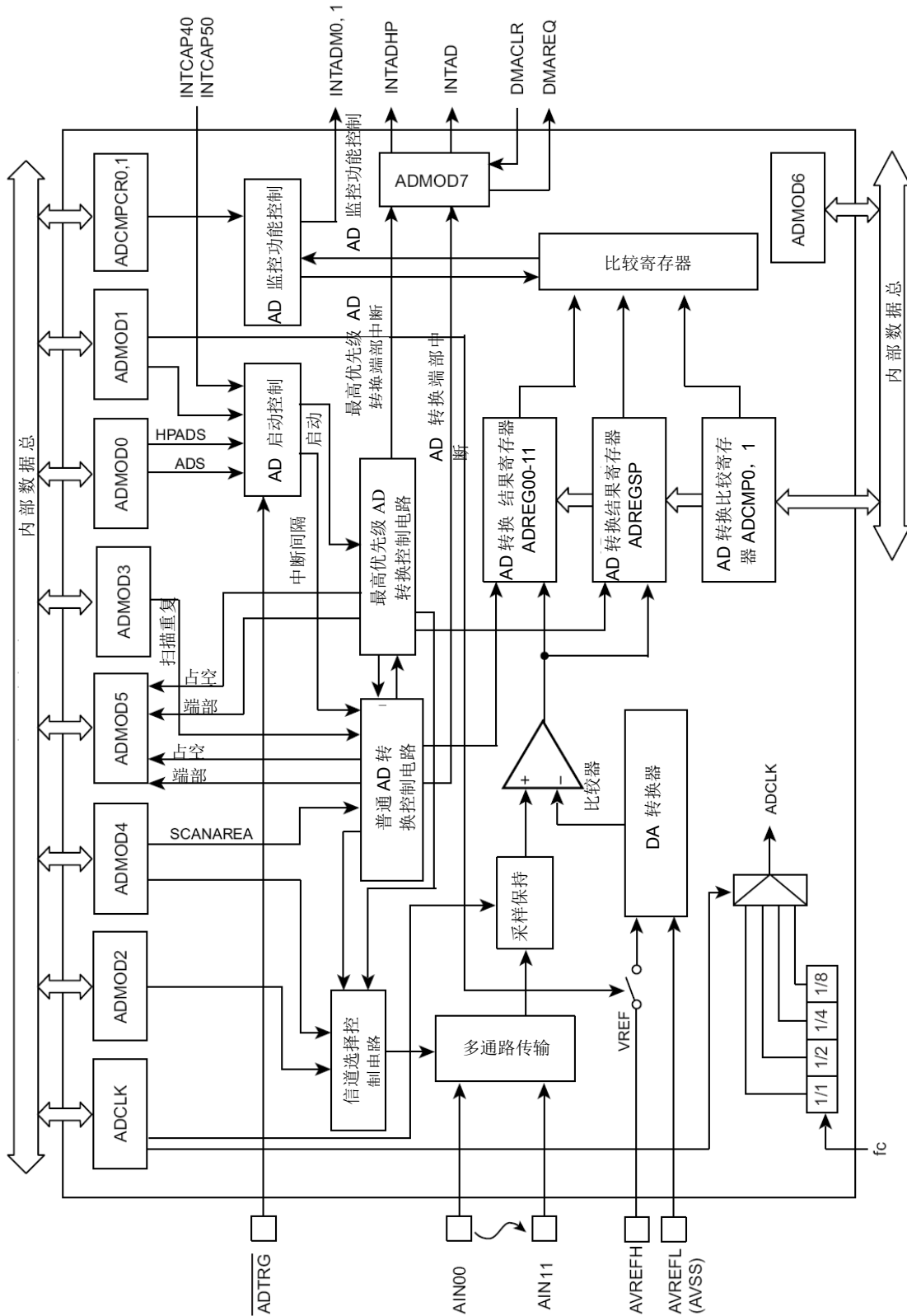


图 16-1 AD 转换器方块图

16.3 寄存器

16.3.1 寄存器列表

控制寄存器和AD转换器地址如下。

AD转换器由AD模式控制寄存器 (ADMOD0 ~ ADMOD7) 控制。AD转换结果存于12AD转换结果寄存器 ADREG00 至 ADREG11。最高优先级转换结果存于寄存器ADREGSP。

基址 = 0x4005_0000

寄存器名称		地址(基+)
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
模式控制寄存器 6	ADMOD6	0x001C
模式控制寄存器 7	ADMOD7	0x0020
监控功能控制寄存器 0	ADCMPCR0	0x0024
监控功能控制寄存器 1	ADCMPCR1	0x0028
转换结果比较寄存器 0	ADCMP0	0x002C
转换结果比较寄存器 1	ADCMP1	0x0030
转换结果寄存器 0	ADREG00	0x0034
转换结果寄存器 1	ADREG01	0x0038
转换结果寄存器 2	ADREG02	0x003C
转换结果寄存器 3	ADREG03	0x0040
转换结果寄存器 4	ADREG04	0x0044
转换结果寄存器 5	ADREG05	0x0048
转换结果寄存器 6	ADREG06	0x004C
转换结果寄存器 7	ADREG07	0x0050
转换结果寄存器 8	ADREG08	0x0054
转换结果寄存器 9	ADREG09	0x0058
转换结果寄存器 10	ADREG10	0x005C
转换结果寄存器 11	ADREG11	0x0060
保留	-	0x0064
保留	-	0x0068
保留	-	0x006C
保留	-	0x0070
转换结果寄存器 SP	ADREGSP	0x0074
保留	-	0x0F00
保留	-	0x0F04
保留	-	0x0F08

注: 禁止进入"保留" 区域。

16.3.2 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	ADSH				-	ADCLK			
复位后	0	0	0	0	0	0	0	1:	

位	比特符号	类型	功能
31-8	-	R	读作 0。
7-4	ADSH[3:0]	R/W	选择AD采样保持时间。 0000: $10 \times \langle \text{ADCLK} \rangle$ 0001: $20 \times \langle \text{ADCLK} \rangle$ 0010: $30 \times \langle \text{ADCLK} \rangle$ 0011: $40 \times \langle \text{ADCLK} \rangle$ 0100: $80 \times \langle \text{ADCLK} \rangle$ 0101 ~ 1111:保留
3.	-	R	读作 0。
2-0	ADCLK[2:0]	R/W	选择AD预分频器时钟。 000: f_c 001: $f_c/2$ 010: $f_c/4$ 011: $f_c/8$ 100 to 111: 保留

注 1: 指定 $4\text{MHz} \leq \text{ADCLK} \leq 40\text{MHz}$ 范围内的ADCLK. 例如, $f_{osc} = 12\text{MHz}$ 和 $\text{PLL} = 8$ 倍时, f_c 达 48MHz . 在这种情况下, 将 $\text{ADCLK} \langle \text{ADCLK}[2:0] \rangle$ 设置为 "000" 以外的数值。

注 2: 除非AD转换暂停且 $\text{ADMOD1} \langle \text{VREFON} \rangle = "0"$, 否则切勿改变 $\langle \text{ADCLK} \rangle$ 的设置。

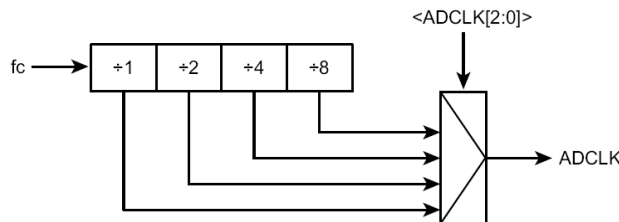


图 16-2 AD转换时钟(ADCLK)

转换所需时钟计数至少为 40 个时钟周期。

采样保持时间和转换时间示例如下所示。

<ADCLK[2:0] > 设置	<ADSH[3:0]>	转换时间		
		fc=32MHz	fc=40MHz	fc=48MHz
000 (fc)	转换时钟周期 × 10	1.25 μs	1.00 μs	-
	转换时钟周期 × 20	1.56 μs	1.25 μs	-
	转换时钟周期 × 30	1.88 μs	1.50 μs	-
	转换时钟周期 × 40	2.19 μs	1.75 μs	-
	转换时钟周期 × 80	3.44 μs	2.75 μs	-
001 (fc/2)	转换时钟周期 × 10	2.50 μs	2.00 μs	1.67 μs
	转换时钟周期 × 20	3.13 μs	2.50 μs	2.08 μs
	转换时钟周期 × 30	3.75 μs	3.00 μs	2.50 μs
	转换时钟周期 × 40	4.38 μs	3.50 μs	2.92 μs
	转换时钟周期 × 80	6.88 μs	5.50 μs	4.58 μs
010 (fc/4)	转换时钟周期 × 10	5.00 μs	4.00 μs	3.33 μs
	转换时钟周期 × 20	6.25 μs	5.00 μs	4.17 μs
	转换时钟周期 × 30	7.50 μs	6.00 μs	5.00 μs
	转换时钟周期 × 40	8.75 μs	7.00 μs	5.83 μs
	转换时钟周期 × 80	-	-	9.17 μs
011 (fc/8)	转换时钟周期 × 10	10.0 μs	8.00 μs	6.67 μs
	转换时钟周期 × 20	-	10.0 μs	8.33 μs
	转换时钟周期 × 30	-	-	10.00 μs
	转换时钟周期 × 40	-	-	-
	转换时钟周期 × 80	-	-	-

注 1: AD转换期间切勿改变AD转换设置。

注 2: 禁止对上表“-”所示单元进行设置。指定ADCLK设置范围为1μs~10μs。

16.3.3 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	-	-	-	-	-	-	HPADS	ADS
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	HPADS	W	激活最高优先级AD转换 0: 忽略 1: 开始转换 总读作"0"。
0	ADS	W	激活普通(软件)AD转换 (注3) 0: 忽略 1: 开始转换 总读作"0"。

注 1: ADC使用时, 先将"1" 写入ADMOD1<VREFON>, 然后通过设置ADMOD0<ADS> 或 <HPADS>启动AD转换或外部触发器。

注 2: 当最高优先级AD转换<HPADS>和普通AD 转换(软件)皆被启用, 选作ADTRG (外部触发器输入)时, 最高优先级AD转换优先激活, 普通AD转换未激活。

16.3.4 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	RCUT	-	HPADHWS	HPADHWE	ADHWS	ADHWE
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 应用程序控制 (注1和注2) 0: OFF 1: ON
6	I2AD	R/W	IDLE 模式中指定操作模式 0: 停止 1: 操作
5	RCUT	R/W	控制 AVREFH-AVREFL 电流 0: 只在转换中应用电流。 1: 随时应用电流, 复位中除外
4	-	R	读作"0"。
3	HPADHWS	R/W	选择最高优先级AD转换硬件激活源 0: 外触发 1: INTCAP40中断
2	HPADHWE	R/W	硬件因素触发激活最高优先级AD转换 0: 禁用 1: 启用
1	ADHWS	R/W	选择普通AD转换硬件激活源(注 3) 0: 外触发 1: INTCAP50中断
0	ADHWE	R/W	硬件因素触发激活普通AD转换 0: 禁用 1: 启用

注 1: 使用AD转换时, 将"1"写入ADMOD<VREFON>位, 等待3 μ s, 期间内部参考电压稳定, 然后通过将AD-MOD0<ADS>或<HPADS> 设置为 "1"启动AD转换或外部触发器。

注 2: 将<VREFON> 设置为"0", 完成AD转换后进入待机模式。

注 3: 用于H/W时, 外部触发器无法用于普通AD转换H/W激活。最高优先级AD转换激活。

注 4: 若需要通过IDLE或STOP模式降低电力电流, 若以下所示任一情况适用, 你须先停止AD转换器, 然后执行指令, 进入待机模式。

1. 进入IDLE模式时, ADMOD1 <I2AD> = "0"。

2. 进入STOP1 模式时。

16.3.5 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	HPADCH				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。
7-4	HPADCH[3:0]	R/W	以最高优先级AD转换选择模拟量输入通道。(见表16-1。)
3-0	ADCH[3:0]	R/W	以普通AD转换选择模拟量输入通道。(见表16-1。)

表 16-1 以普通AD转换或最高优先级AD转换选择输入通道

<HPADCH[3:0]>	模拟量输入通道 以最高优先级AD转换	<ADCH[3:0]>	模拟量输入通道 以普通AD转换
0000	AIN00	0000	AIN00
0001	AIN01	0001	AIN01
0010	AIN02	0010	AIN02
0011	AIN03	0011	AIN03
0100	AIN04	0100	AIN04
0101	AIN05	0101	AIN05
0110	AIN06	0110	AIN06
0111	AIN07	0111	AIN07
1000	AIN08	1000	AIN08
1001	AIN09	1001	AIN09
1010	AIN10	1010	AIN10
1011	AIN11	1011	AIN11



16.3.6 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	-	ITM			-	-	REPEAT	SCAN
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。
6-4	ITM[2:0]	R/W	以固定通道重复转换模式指定中断。见表 16-2。
3-2	-	R	读作 0。
1	REPEAT	R/W	指定重复模式 0: 单次转换模式 1: 重复转换模式
0	SCAN	R/W	指定扫描模式 0: 固定通道模式 1: 通道扫描模式

表 16-2 以固定通道重复转换模式进行AD转换的中断规格

<ITM[2:0]>	固定通道重复转换模式 <SCAN> = "0", <REPEAT> = "1"
000	每隔 1 次转换生成一次中断。
001	每隔 2 次转换生成一次中断。
010	每隔 3 次转换生成一次中断。
011	每隔 4 次转换生成一次中断。
100	每隔 5 次转换生成一次中断。
101	每隔 6 次转换生成一次中断。
110	每隔 7 次转换生成一次中断。
111	每隔 8 次转换生成一次中断。

注 1: 只有指定的固定通道重复模式, <REPEAT> = "1" 和 <SCAN> = "0", <ITM[2:0]>才有效。

注 2: 重复转换期间终止重复转换时(<REPEAT>=1, 以固定通道模式或通道扫描模式), <REPEAT> 为被"0"清除。这种情况下, 切勿改变设置, <REPEAT>位除外。

16.3.7 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	SCANAREA				SCANSTA			
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。
7-4	SCANAREA [3:0]	R/W	通道扫描范围 (1100 ~ 1111 禁止)
3-0	SCANSTA[3:0]	R/W	选择开始扫描的通道(1100 ~ 1111 禁止)

指定通道扫描单次模式时，将ADMOD3<SCAN>设置为"1"且<REPEAT> 至"0"。要指定通道扫描重复模式，将ADMOD3<SCAN> 设置为 "1"且<REPEAT> 至 "1"。

首先，选择开始扫描的通道。然后，选择扫描的通道数目，从指定的开始通道开始。

例如，ADMOD4<SCANSTA> 设置为 "0001"(AIN01)且<SCANAREA>设置为"0010" (3ch 扫描)，从 AIN01~AIN03的三条通道被扫描。

以下显示的是关于<SCANSTA>设置的<SCANAREA>可指定数值范围。

表 16-3 可指定通道扫描数值范围

<SCANSTA[3:0]>	开始扫描的通道	<SCANAREA[3:0]>	可指定通道扫描数值范围
0000	(AIN00)	0000~1011	(1ch~12ch)
0001	(AIN01)	0000~1010	(1ch~11ch)
0010	(AIN02)	0000~1001	(1ch~10ch)
0011	(AIN03)	0000~1000	(1ch~9ch)
0100	(AIN04)	0000~0111	(1ch~8ch)
0101	(AIN05)	0000~0110	(1ch~7ch)
0110	(AIN06)	0000~0101	(1ch~6ch)
0111	(AIN07)	0000~0100	(1ch~5ch)
1000	(AIN08)	0000~0011	(1ch~4ch)
1001	(AIN09)	0000~0010	(1ch~3ch)
1010	(AIN10)	0000~0001	(1ch~2ch)
1011	(AIN11)	0000	(1ch)

注：如设置并非上述所列，即使ADMOD0寄存器设置为激活AD转换，AD转换也不会被激活。

16.3.8 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	-	-	-	-	HPEOCF	HPADBF	EOCF	ADBF
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	HPEOCF	R	最高优先级AD转换完成标志 (注 1) 0: 转换之前或期间 1: 完成
2	HPADBF	R	最高优先级AD转换BUSY标志 0: 转换暂停期间 1: 转换期间
1	EOCF	R	正常AD转换完成标志 (注 1) 0: 转换之前或期间 1: 完成
0	ADBF	R	普通AD转换BUSY标志 0: 转换暂停期间 1: 转换期间

注 1: 通过读取ADMOD5寄存器, 该标志被"0"清除。

注 2: 若需要通过IDLE或STOP模式降低电力电流, 若以下所示任一情况适用, 你须先停止AD转换器, 然后执行指令, 进入待机模式。

1. 进入IDLE模式时, ADMOD1 <I2AD> = "0"。
2. 进入STOP1 模式时。

16.3.9 ADMOD6 (模式控制寄存器 6)

	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	-	-	-	-	-	-	ADRST	
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。
1-0	ADRST[1:0]	W	用01覆盖10，允许ADC软件复位。 软件复位初始化所有寄存器，ADCLK<ADCLK>除外。

注 1: 使用AD转换完成中断进行DMA传输时，首先软件复位ADMOD6 <ADRST>，然后，操作DMAC(DMA请求待机状态)并为ADC进行配置(激活)。

注 2: When 进行软件复位时，ADMOD1位<VREFON>为"1"属有效。

注3: 软件复位时，初始化须花费3μs。

16.3.10 ADMOD7 (模式控制寄存器7)

	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	-	-	-	-	-	-	INTADHPD- MA	INTADDMA
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	始终写为"0"。
1	INTADHPDMA	R/W	指定最高优先级AD转换DMA激活因素。 0: 禁用 1: 启用
0	INTADDMA	RW	指定普通AD转换DMA激活因素。 0: 禁用 1: 启用

16.3.11 ADCMPCR0 (监控控制寄存器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	-	-	-	-	CMPCNT0			
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	CMPOEN	-	CMPCOND0	ADBIG0	AINS0			
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-12	-	R	读作"0"。
11-8	CMPCNT0[3:0]	R/W	<p>确认判断前的比较数目。 达到计数数目时，生成中断。</p> <p>0000：每次 0110：超过7次 1100：超过13次 0001：超过2次 0111：超过8次 1101：超过14次 0010：超过3次 1000：超过9次 1110：超过15次 0011：超过4次 1001：超过10次 1111：超过16次 0100：超过5次 1010：超过11次 0101：超过6次 1011：超过12次</p>
7	CMP0EN	R/W	<p>AD监控功能 0 0：禁用 1：启用</p> <p>设置条件 <CMP0EN>="0" (禁用) 清除计数数目。</p>
6	-	R	读作"0"。
5	CMPCOND0	R/W	<p>为判断计数设置条件。 0：序列 1：累积</p> <p>采用序列法，对<ADBIG0>所设的条件持续并达到对<CMPCNT0>所设的数目时，发生AD监控中断。超过设置值后，每次判断条件为真时，AD监控发生中断。若条件不同于对<ADBIG0>所设条件，计数器清除。</p> <p>采用累积法，对<ADBIG0>所设条件累积，达到对<CMPCNT0>所设数目时，发生AD监控中断，计数器清除。即使条件不同于对<ADBIG0>所设数值，计数器数值有效。</p>
4	ADBIG0	R/W	<p>设置AD监控功能中断0 (INTADM0) 0：若转换结果寄存器的值大于比较寄存器0，生成中断。 1：若转换结果寄存器的值小于比较寄存器0，生成中断。</p> <p>每次完成对<AINS0[3:0]>所设的AD转换时，比较转换结果规格。若结果和<ADBIG0>设置匹配，计数器增加。</p>
3-0	AINS0[3:0]	R/W	<p>设置模拟输入，作为比较目标。</p> <p>0000：AIN00 0101：AIN05 1010：AIN10 0001：AIN01 0110：AIN06 1011：AIN11 0010：AIN02 0111：AIN07 0011：AIN03 1000：AIN08 0100：AIN04 1001：AIN09</p> <p>1100 ~ 1111:</p>

注： AD监控功能用于固定重复模式和扫描重复模式。

16.3.12 ADCMPCR1 (AD监控控制寄存器 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	-	-	-	-	CMPCNT1			
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	CMP1EN	-	CMPCOND1	ADBIG1	AINS1			
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-12	-	R	读作 0。
11-8	CMPCNT1[3:0]	R/W	<p>确认判断前的比较数目。 达到计数数目时，生成中断。</p> <p>0000：每次 1000：超过9次 0001：超过2次 1001：超过10次 0010：超过3次 1010：超过11次 0011：超过4次 1011：超过12次 0100：超过5次 1100：超过13次 0101：超过6次 1101：超过14次 0110：超过7次 1110：超过15次 0111：超过8次 1111：超过16次</p>
7	CMP1EN	R/W	<p>AD监控功能 1</p> <p>0: 禁用 1: 启用</p>
6	-	R	读作 0。
5	CMPCOND1	R/W	<p>为判断计数设置条件。</p> <p>0: 序列 1: 累积</p> <p>采用序列法，对<ADBIG0>所设的条件持续并达到对<CMPCNT0>所设的数目时，发生AD监控中断。超过设置值后，每次判断条件为真时，AD监控发生中断。若条件不同于对<ADBIG0>所设条件，计数器清除。</p> <p>采用累积法，对<ADBIG0>所设条件累积，达到对<CMPCNT0>所设数目时，发生AD监控中断，计数器清除。即使条件不同于对<ADBIG0>所设数值，计数器数值有效。</p>
4	ADBIG1	R/W	<p>设置AD监控器功能中断 1(INTADM1)</p> <p>0: 若转换结果值大于比较寄存器1，生成中断。 1: 若转换结果值小于比较寄存器1，生成中断。</p> <p>每次完成对<AINS0[3:0]>所设的AD转换时，比较转换结果规格。 若结果和<ADBIG0>设置匹配，计数器增加。</p>
3-0	AINS1[3:0]	R/W	<p>使用AD监控功能1时，选择目标转换结果寄存器。</p> <p>0000：ADREG00 非0000值：勿设。</p>

注：AD监控功能用于固定重复模式和扫描重复模式。

16.3.13 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	-	-	-	-	AD0CMP			
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	AD0CMP							
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-12	-	R	读作 0。
11-0	AD0CMP[11:0]	W	设置AD转换比较值。

注: 要将数值写入该寄存器, AD监控功能须禁用 (ADCMPCR0<CMP0EN> ="0")。

16.3.14 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
	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	-	-	-	-	AD1CMP			
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	AD1CMP							
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-12	-	R	读作 0。
11-0	AD1CMP[11:0]	W	设置AD转换比较值。

注: 要将数值写入该寄存器, AD监控功能须禁用 (ADCMP1EN = "0")。

16.3.15 ADREG00~ADREG11 (普通转换结果寄存器 00 ~ 11)

	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	-	ADPOSWF	ADOVRF	ADRF	ADR			
复位后	0	0	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
比特符号	ADR							
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-15	-	R	读作 0。
14	ADPOSWF	R	AIN端口的输出切换标志。 0: 无切换 1: 有切换 也用作AIN的总输入输出端口的PxDATA寄存器, 在AD转换期间改变时, 端口输出切换标志<ADPOSWF>设置为"1"。 这种情况下, 对应改变位的PxCR寄存器为"1"时, AD转换期间的输出切换可能影响到转换准确性。 寄存器ADREG00~ADREG11被读取时, 该位被"0"清除。
13	ADOVRF	R	超限标志 0: 未生成。 1: 生成。 若转换结果在读取<ADR>前被覆盖, 该位设置为"1"。寄存器ADREG00~ADREG11被读取时, 该位被"0"清除。
12	ADRF	R	AD转换结果保存标志 0: 转换结果未保存 1: 转换结果被保存。 若转换结果被保存, 该位设置为"1"。 寄存器ADREG00~ADREG11转换结果被读取时, 该位被"0"清除。
11-0	ADR[11:0]	R	AD转换结果 转换结果被保存。转换通道和转换结果寄存器相关性的有关信息, 参阅 16.4.5.7 章节表 16-5。

注: 其它模拟/输入输出端口用作输出端口时, AD转换期间切勿进行输出切换。

16.3.16 ADREGSP (最高优先级转换结果寄存器)

	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	-	-	ADOVRFSP	ADRFSP	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							
复位后	0	0	0	0	0	0	0	0

位	比特符号	类型	功能
31-15	-	R	读作 0。
14	-	R	读作未定义值。
13	ADOVRFSP	R	超限标志 0: 未生成 1: 生成 若最高优先级AD转换结果读取<ADRSP>前被覆盖, "1"设置。 ADREGSP寄存器被读取时, 该位被"0"清除。
12	ADRFSP	R	最高优先级AD转换结果保存标志 0: 转换结果未被保存。 1: 转换结果被保存。 最高优先级转换结果被保存, 该位设置为"1"。 ADREGSP转换结果被读取时, 该位被"0"清除。
11-0	ADRSP[11:0]	R	最高优先级AD转换结果 最高优先级转换结果被保存。转换通道和转换结果寄存器相关性的有关信息, 参阅 16.4.5.7 章节表 16-5。

16.4 操作说明

16.4.1 模拟参考电压

模拟参考电压"高"电平应用于AVRFEH引脚，"低"电平则应用于AVREFL引脚。

启动AD转换时，应将"1"写入<VREFON>位，等待3 μ s，期间内部参考电压应稳定，然后将"1"写入ADMOD0<ADS>位。

通过将"0"写入ADMOD1<RCUT>位，AVREFH – AVREFL的开启状态可转为关闭状态。

16.4.2 AD转换模式

支持两类AD转换：普通AD转换和最高优先级AD转换。

对于普通AD转换，支持以下四种操作模式。

16.4.2.1 普通AD转换

对于普通AD转换，支持以下四种操作模式，通过ADMOD3<REPEAT, SCAN>选择操作模式。

- 固定通道单次转换模式
- 通道扫描单次转换模式
- 固定通道重复转换模式
- 通道扫描重复转换模式

(1) 固定通道单次转换模式

若ADMOD3<REPEAT, SCAN>设置为"00"，固定通道单次转换模式中进行AD转换。

这种模式中，对ADMOD2<ADCH>选定的一个通道进行一次AD转换。AD转换后，ADMOD5<EOCF>设置为"1"，ADMOD5<ADBF>被清至"0"，生成AD转换完成中断请求 (INTAD)。<EOCF>一旦读取，则被清至"0"。

(2) 通道扫描单次转换模式

若ADMOD3 <REPEAT, SCAN>设置为"01"，通道扫描单次转换模式中进行AD转换。

此模式中，从ADMOD4 <SCANSTA>选定的开始通道开始，对ADMOD4 <SCANAREA>选定的扫描通道范围进行AD转换。AD扫描转换完成后，ADMOD5<EOCF>设置为"1"，ADMOD5<ADBF>被清至"0"，生成转换完成中断请求 (INTAD)。<EOCF>一旦读取，则被清至"0"。

(3) 固定通道重复转换模式

若ADMOD3<REPEAT, SCAN>设置为"10"，固定通道重复转换模式中进行AD转换。

这种模式中, ADMOD2 <ADCH>选定的一个通道重复进行AD转换。AD转换完成后, ADMOD5<EOCF>设置为"1"。ADMOD5<ADBF>未被清至"0"。保留为"1"。通过对 ADMOD3<ITM>进行适当设置, 可选择生成转换完成中断请求(INTAD)的时机。该中断 INTAD生成的时机, 和<EOCF>设置的相同。<EOCF>一旦读取, 则被清至"0"。

(4) 通道扫描重复转换模式

若ADMOD3<REPEAT, SCAN>设置为"11", 通道扫描重复转换模式中进行AD转换。

此模式中, 从ADMOD4 <SCANSTA> 选定的开始通道开始, 对 ADMOD4 <SCANAREA> 选定的扫描通道范围重复进行AD转换。每次一个AD扫描转换完成时, ADMOD5 <EOCF>设置为"1", 生成转换完成中断请求(INTAD)。ADMOD5 <ADBF>未被清至"0", 保留为"1"。<EOCF>一旦读取, 则被清至"0"。

16.4.2.2 最高优先级AD转换

通过中断进行中的普通AD转换, 可进行最高优先级AD转换。

固定通道单次转换为自动选择, 和ADMOD3 <REPEAT, SCAN>设置无关。启动操作的条件满足时, ADMOD2<HPADCH>选定的通道只进行一次转换。转换完成后, 生成最高优先级AD转换完成中断 (INTADHP), 显示AD转换完成的ADMOD5 <HPEOCF>设置为"1"。<HPADBF>归"0"。<HPEOCF> 标志一旦读取, 则被清至"0"。

最高优先级AD转换进行中时, 激活的最高优先级AD转换被忽略。

16.4.3 AD监控功能

该功能用于配置固定通道重复模式和扫描重复模式。

若ADCMPCR0<CMP0EN>和ADCMPCR1<CMP1EN>设置为"1", AD监控功能启用。两种监控功能同时启用。

这里以ADCMPCR0 (和ADCMPCR1相同)为例。

比较模拟输入设置为 ADACMPCR0/ADBCMPCR0<AINS0[3:0]>。大小判断设置为 <ADBIG0>。该比较计数条件设置为<CMPCOND0>。比较计数数目设置为<CMPCNT0[3:0]>。

AD转换开始后, AD转换器检查每次AD转换的比较条件(小于/大于比较寄存器数值)。若比较结果和<ADBIG0>设置一致, AD转换器相加等于计数器值。

比较条件分为两类: 顺序法和累积法。

顺序法中, 若为<ADBIG0>设置的条件持续, 达到为<CMPCNT0[3:0]>设置的计数数目, AD监控中断(INTADM0)发生。若比较结果即使在达到为<CMPCNT0[3:0]>设置的计数数目后仍然和条件相符, 中断发生, 计数器未清除。只有在条件未达到<ADBIG0>设置条件的时候, 计数器数值才会清零。

累积法中, 对<ADBIG0>所设条件满足的总次数达到对<CMPCNT0[3:0]>所设的数目时, 计数器被清零。此时AD监控中断(INTADM0)发生。即使比较的结果不同于计数器设置值, 计数器值仍得以保留。若ADCMPCR0寄存器配置的转换结果寄存器数值和目标寄存器的相同, AD转换器不累计。AD转换中断 (INTADM0) 也不发生。

每次结果保存于相应转换结果寄存器时，进行该比较操作。条件符合时(包括计数)，中断(INTADM0)发生。由于分配执行AD监控功能的转换结果寄存器通常并非由软件读取，转换结果保存标志ADREG<ADRF>和溢出标志ADREG<ADOVRF>保留设置。使用AD监控功能时，切勿使用转换结果寄存器。

1. AIN00 输入设置为固定通道重复转换模式，对AD转换结果比较寄存器数值(0x0888)进行比较。
 - ADMOD3=0x0002: 固定通道重复转换
AD转换完成中断 (INTAD) 被禁用。
 - ADCMPCR0 =0x0280: 比较目标通道: AIN00, 大小判断: 大于比较寄存器数值。比较计数条件: 顺序法 AD 监控功能: 启用。大小判断计数: 3 次计数。
 - ADCMP0=0x0888: AD转换结果比较寄存器(比较值 0x0888)

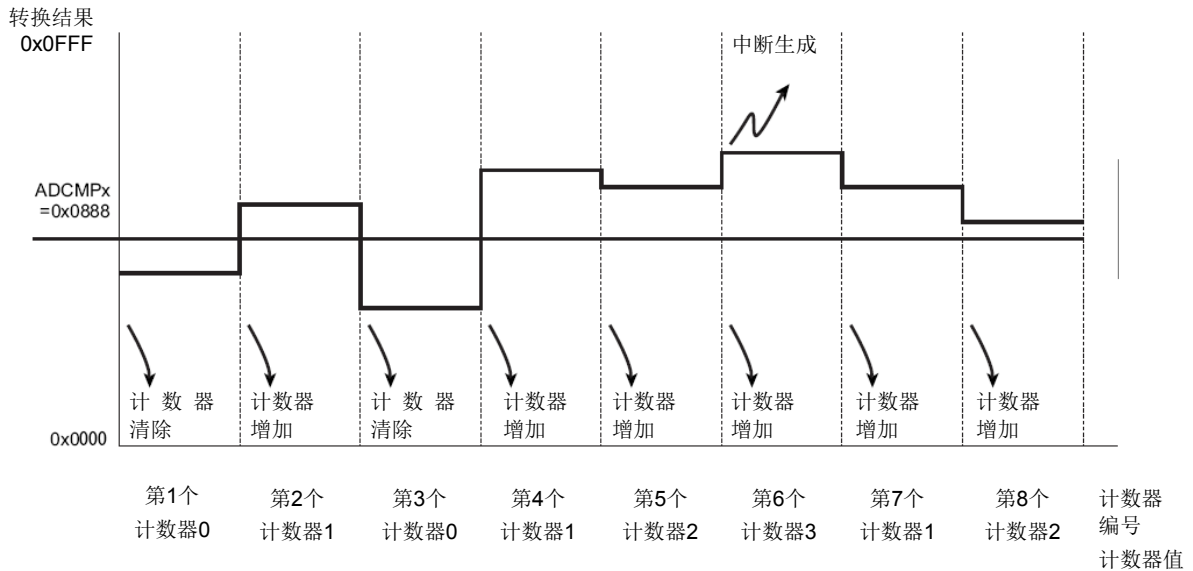


图 16-3 AD监控功能 (固定通道重复和顺序法)

2. AIN00 设置为固定通道重复转换，比较AD转换结果比较寄存器值 (0x0888)。
 - ADM0D3=0x0002: 固定通道重复转换
AD转换完成中断 (INTAD) 被禁用。
 - ADCMPCR0 =0x02A0: 比较目标通道: AINA00, 大小判断: 大于比较寄存器值。比较计数条件: 累积法。AD 监控功能: 启用。大小判断计数: 3 次计数
 - ADCMP0=0x0888: AD转换结果比较寄存器(比较值 0x0888)

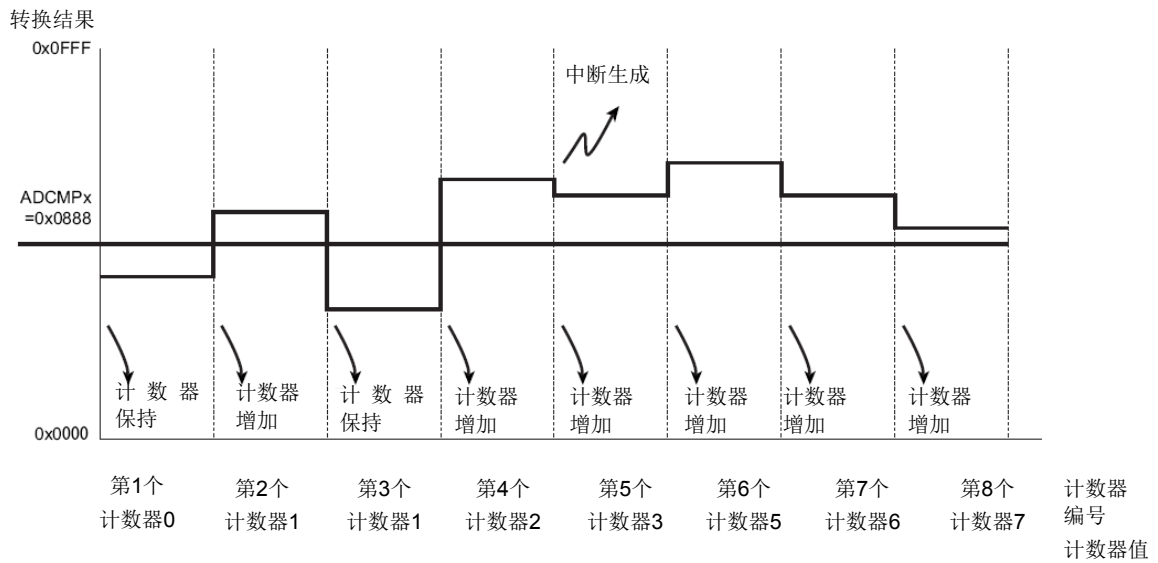


图 16-4 AD 监控功能 (固定通道重复和累积法)

16.4.4 选择输入通道

复位后，ADMOD3 <REPEAT, SCAN>初始化为"00"，ADMOD2 <ADCH[3:0]>初始化为"0000"。

根据如下所示AD转换器操作模式选择拟转换通道。

1. 普通AD转换模式

- 若模拟输入通道使用于固定状态(ADMOD3<SCAN> = "0")

通过对ADMOD2 <ADCH>进行适当设置，从AIN00至AIN11的模拟输入引脚选择一个通道

- 若模拟输入通道使用于扫描状态(ADMOD3<SCAN> = "1")

通过设置ADMOD4 <SCANSTA>，可指定待启动通道。而且，通过设置ADMOD4 <SCANAREA>，可指定拟扫描通道数目。

2. 最高优先级AD转换模式

通过对ADMOD2<HPADCH>进行适当设置，从AIN00~AIN11的模拟输入引脚选择一个通道。若最高优先级AD转换在普通AD转换期间已经被激活，暂停进行中的普通AD转换，在最高优先级AD转换完成后重新启动普通AD转换。

16.4.5 AD转换详情

16.4.5.1 启动AD转换

支持两类AD转换：普通AD转换和最高优先级AD转换。将ADMOD0<ADS>设置为"1"，激活普通AD转换。将ADMOD0<HPADS>设置为"1"，激活最高优先级AD转换。

普通AD转换有四种可用操作模式。进行普通AD转换时，须通过对 ADMOD3 <REPEAT, SCAN>进行适当设置选择其中一种操作模式。对于最高优先级AD转换而言，只有一种操作模式可用：固定通道单次转换模式。

使用通过ADMOD1<ADHWS>选择的H/W激活源，可激活普通AD转换，使用通过ADMOD1<HPADHWS>选择的HW激活源，可激活最高优先级AD转换。若 <ADHWS> 和 <HPADHWS>位为"0"，通过ADTRG引脚输入下降缘，可激活普通和最高优先级AD转换。若这些位为"1"，16-位定时器通道 5 生成的INTCAP50可激活普通AD转换，16-位定时器通道 4 生成的INTCAP40可激活最高优先级AD转换。

要允许H/W激活，将ADMOD1<ADHWE>设置为"1"进行普通AD转换，将ADMOD1<HPADHWE>设置为"1"进行最高优先级AD转换。

即时允许H/W激活后，软件激活仍然有效。

注：外部触发器用于最高优先级AD转换的HW激活源时，外部触发器无法设置为激活普通AD转换HW 启动。

16.4.5.2 AD转换

普通AD转换启动时，AD转换占空标志（ADMOD5 <ADBF>）显示进行中AD转换设置为"1"。

最高优先级AD转换启动时，最高优先级AD转换占空标志（ADMOD5 <HPADBF>）显示进行中AD转换设置为"1"。

那时，最高优先级AD转换启动前的普通AD转换占空标志ADMOD5<ADBF>数值被保留。

最高优先级AD转换启动前的普通AD转换完成标志数值ADMOD5<EOCF>被保留。

注：最高优先级AD转换进行时，须不得激活普通AD转换。最高优先级AD转换进行时若被激活，最高优先级AD转换完成标志无法设置，之前普通AD转换标志无法清除。

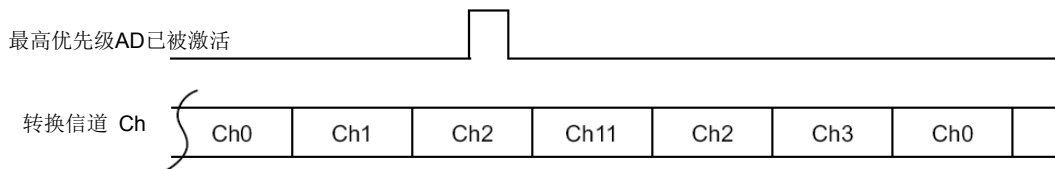
16.4.5.3 普通AD转换期间的最高优先级AD转换请求

若最高优先级AD转换在普通AD转换期间已经被激活，暂停进行中的普通AD转换，在最高优先级AD转换完成后重新启动普通AD转换。

若普通AD转换期间ADMOD0<HPADS>设置为"1"，进行中的普通AD转换被暂停，最高优先级AD转换启动；具体而言，ADMOD2<HPADCH>指定通道进行AD转换(固定通道单次转换)。该最高优先级AD转换结果被保存于存储寄存器ADREGSP后，正常AD转换恢复。

若正常AD转换期间最高优先级AD转换的H/W激活获授权，使用H/W激活源的激活要求满足时，进行中的AD转换停止，ADMOD2<HPADCH>指定通道启动最高优先级AD转换(固定通道单次转换)。该最高优先级AD转换结果被保存于存储寄存器ADREGSP后，正常AD转换恢复。

例如，若通道AIN00 至AIN03 的通道重复转换被激活，而且如果AIN02转换期间<HPADS>设置为"1"，AIN02转换暂停，<HPADCH> 指定通道进行转换(如下所示情况为AIN11)。转换结果被保存于ADREGSP后，自AIN02开始，通道重复转换恢复。



16.4.5.4 停止重复转换模式

要停止重复转换模式(固定通道重复转换模式或通道扫描转换模式)中的AD转换操作，将"0"写入ADM0D3<REPEAT>。进行中AD转换完成时，重复转换模式终止，ADM0D5<ADBF>设置为"0"。

16.4.5.5 重新激活AD转换

若正常AD转换期间ADMOD0 <ADS>设置为"1", 正常AD转换重新激活。被重新激活时, 进行中的AD转换暂停。那时, 正常AD转换占空标志 ADMOD5 <ADBF>, 正常AD转换完成标志 ADMOD5 <EOCF>以及保存结果标志ADREGm<ADOVRF>, <ADRF> 被清除为"0"。(m=00-11)

若AD转换期间正常AD转换H/W激活获授权, 使用H/W激活源激活条件满足时, 进行中AD转换停止。重新激活时, 进行中正常AD转换暂停。那时, 正常AD转换占空标志ADMOD5 <ADBF>, 正常AD转换完成标志ADMOD5 <EOCF> 以及保存结果标志 ADREGm <ADOVRF>, <ADRF> 被清除为"0"。(m=00-11)

16.4.5.6 转换完成

(1) 正常AD转换完成

正常AD转换完成时, AD转换完成中断 (INTAD)生成。AD转换结果被保存于存储寄存器, 两个寄存器改变: 显示AD转换完成的寄存器ADMOD5<EOCF>和寄存器ADMOD5<ADBF>。根据所选择的模式, 中断请求生成的时机和转换结果寄存器 <EOCF> <ADBF>改变的时机不同。

非固定通道重复转换模式中, 转换结果保存于相应通道的AD转换结果寄存器 (ADREG00~ADREG11)。

固定通道重复转换模式中, 转换结果按顺序保存于存储寄存器ADREG00至ADREG11。但是, 若<ITM>上的中断设置设置为每次完成AD转换时生成, 则转换结果仅保持于ADREG00。若<ITM>上的中断设置设置为每当8次AD转换完成时生成, 转换结果按顺序保存于ADREG00至ADREG07。

各模式中的中断请求, 标志改变以及转换结果寄存器如下所示。

- 固定通道单次转换模式

AD转换完成后, ADMOD5 <EOCF>设置为"1", ADMOD5 <ADBF>被清除为"0", 中断请求INTAD生成。
转换结果保存于相应通道的转换结果寄存器。

- 通道扫描单次转换模式

通道扫描转换完成后, ADMOD5 <EOCF>设置为"1", ADMOD5 <ADBF>被清除为"0", 中断请求INTAD生成。
转换结果保存于相应通道的转换结果寄存器。

- 固定通道重复转换模式

ADMOD5 <ADBF>未被清除为"0"。保留为"1"。对ADMOD3<ITM>进行适当设置, 可选择中断请求INTAD生成的时机。ADMOD5 <EOCF>设置的时机, 和该中断INTAD生成的相同。

- a. 1次转换

将 ADMOD2<ADCH[3:0]> 设置为 "0000"(AIN00) , 将 ADMOD3 <ITM[2:0]> 设置为"000", 每当AD转换完成一次时, 中断请求生成。在这种情况下, 转换结果通常保存于存储寄存器ADREG00。转换结果保存后, <EOCF>设置为"1"。

- b. 8次转换

将 ADMOD2 <ADCH[3:0]> 设置为 "1011"(AIN11) 并将 ADMOD3 <ITM[2:0]>设置为 "111", 每当8次AD转换完成时, 中断请求生成。这种情况下, 转换结果按顺序保存于存储寄存器 ADREG00~ADREG07。转换结果保存于ADREG07后, <EOCF>设置为"1", 随后转换结果的保存从ADREG00开始。

- 通道扫描重复转换模式

每次完成一个AD转换, 将ADMOD5<EOCF>设置至"1", 并生成一个中断请求INTAD。ADMOD5<ADBF>未被清至"0"。保留为"1"。

若ADMOD4 <SCANSTA[3:0]>设置为"0001" (AIN01) 且ADMOD4 <SCANAREA[7:4]>设置为"1110" (11Ch扫描), 每当一次AD转换完成时, ADMOD5 <EOCF>设置为"1", 中断请求INTAD生成。ADMOD5 <ADBF>未被清至"0", 保留为"1"。

AD转换结果保存于对应通道的AD转换结果寄存器。

(2) 最高优先级AD转换完成

最高优先级AD转换完成后, 最高优先级AD转换完成中断(INTADHP)生成, 显示最高优先级AD转换完成的ADMOD5<HPEOCF>设置为"1"。

AD转换结果保存于AD转换结果寄存器SP。

(3) 数据轮询

要确认AD转换完成而又无需使用中断, 可使用数据轮询。AD转换完成时, ADMOD5 <EOCF>设置为"1"。要确认AD转换完成并获得结果, 对该位进行轮询。

AD转换结果存储寄存器须为字存取读取。若<ADOVRF> = "0", <ADRF> = "1"且<ADPOSWF> = "0", 则已获得正确转换结果。

(4) DMA请求

生成正常AD转换完成中断 (INTAD)或最高优先级AD转换完成中断 (INTADHP)后, DMA请求下达。通过对ADMOD7寄存器进行适当设置, 中断生成后的DMA请求可设置为"禁用"或"启用"。AD转换完成中断(INTAD或INTADHP)生成后, DMA请求于2个系统时钟(fsys)内下达。

16.4.5.7 中断生成时机和AD转换结果存储寄存器

表 16-4 显示以下三个事项的关系：AD转换模式，中断生成时机和标志操作。表 16-5 表明模拟通道输入和AD转换结果寄存器之间的关系。

表 16-4 转换模式，中断生成时机和标志操作的关系

转换模式		扫描/重复模式设置 (ADM0D3)			中断生成时机	(ADM0D5)		
		<REPEAT>	<SCAN>	<ITM[2:0]>		<EOCF>/<HPEOCF> 设置时机 (见注1)	<ADBFN> (中断生成后)	<ADBFHP> (中断生成后)
正常转换	固定通道单次转换	0	0	-	生成完成后。	生成完成后。	0	-
	固定通道重复转换	1	0	000	每回完成一次转换。	完成一次转换后。	1	-
				001	每回完成2次转换。	完成2次转换后。	1	-
				010	每回完成3次转换。	完成3次转换后。	1	-
				011	每回完成4次转换。	完成4次转换后。	1	-
				100	每回完成5次转换。	完成5次转换后。	1	-
				101	每回完成6次转换。	完成6次转换后。	1	-
				110	每回完成7次转换。	完成7次转换后。	1	-
111	每回完成8次转换。	完成8次转换后。	1	-				
通道扫描单次转换	0	1	-	扫描转换完成后。	扫描转换完成后。	0	-	
通道扫描重复转换	1	1	-	一次扫描转换完成后。	一次扫描转换完成后。	1	-	
最高优先级转换		-	-	-	生成完成后。	转换完成	-	0

注 1：一旦读取， ADM0D5 <EOCF>和<HPEOCF>被清除。

注 2：重复模式中，即使中断生成， ADM0D5 <ADBFN>未被清除为"0"。要暂停重复操作， ADM0D3 <REPEAT>写入"0"且AD转换完成后， ADM0D5 <ADBFN>被清除为"0"。

表 16-5 模拟通道输入和AD转换结果寄存器之间的关系

固定通道 单次模式		固定通道 重复模式		
通道	存储寄存器	ADM0D3<ITM[2:0]>		存储寄存器
AIN00	ADREG00	000	每次AD转换后中断	ADREG00
AIN01	ADREG01	001	每隔2次AD转换则中断	ADREG00 ~ ADREG01
AIN02	ADREG02	010	每隔3次AD转换则中断	ADREG00 ~ ADREG02
AIN03	ADREG03	011	每隔4次AD转换则中断	ADREG00 ~ ADREG03
AIN04	ADREG04	100	每隔5次AD转换则中断	ADREG00 ~ ADREG04

表 16-5 模拟通道输入和AD转换结果寄存器之间的关系

固定通道 单次模式		固定通道 重复模式		
通道	存储寄存器	ADMOD3<ITM[2:0]>		存储寄存器
AIN05	ADREG05	101	每隔6次AD转换则中断	ADREG00 ~ ADREG05
AIN06	ADREG06	110	每隔7次AD转换则中断	ADREG00 ~ ADREG06
AIN07	ADREG07	111	每隔8次AD转换则中断	ADREG00 ~ ADREG07
AIN08	ADREG08			
AIN09	ADREG09			
AIN10	ADREG10			
AIN11	ADREG11			

通道扫描单次模式 / 重复模式 (例如: ADREG03 取决于扫描通道范围)		
ADMOD4<SCANSTA> (启动通道)	ADMOD4<SCANAREA> (扫描通道范围)	存储寄存器
AIN00	12 通道	ADREG00 ~ ADRE11
AIN01	11 通道	ADREG01 ~ ADRE11
AIN02	10 通道	ADREG02 ~ ADRE11
AIN03	9 通道	ADREG03 ~ ADRE11
AIN04	8 通道	ADREG04 ~ ADRE11
AIN05	7 通道	ADREG05 ~ ADRE11
AIN06	6 通道	ADREG06 ~ ADRE11
AIN07	5 通道	ADREG07 ~ ADRE11
AIN08	4 通道	ADREG08 ~ ADRE11
AIN09	3通道	ADREG09 ~ ADRE11
AIN10	2 通道	ADREG10 ~ ADRE11
AIN11	1 通道	ADREG11 ~ ADRE11

注: 通道扫描范围设置为通道扫描模式可分配数值以外时, 即使ADMOD0设置为激活AD转换, AD转换无法激活。

AD转换输入设计注意事项

<和AIN引脚相连外部信号源的输出阻抗>

和AIN引脚相连外部信号源的输出阻抗等于或小于以下公式所示 R_{EXAIN} 。

-外部信号源输出阻抗允许值的计算公式-

和AIN引脚相连输出阻抗的最大值: $R_{EXAIN} < Tscyc \div (ADCLK \times C_{ADC} \times \ln(2^{14})) - R_{AIN}$

MCU 信息	符号	最小值	典型	最大值	单位
ADC时钟频率	ADCLK	4	-	40	MHz
MCU中总AIN输入容量	C_{ADC}	-	-	12.2	pF
MCU中的AIN电阻	R_{AIN}	-	-	1	k Ω
采样保持时间内的循环数	Tscyc	10	-	80	Cycle

R_{EXAIN} 最大值列表 (ADCLK = 40 MHz)

Tscyc	R_{EXAIN}	单位
10	1.1	k Ω
20	3.2	k Ω
30	5.3	k Ω

AD转换输入设计注意事项

40	7.5	kΩ
80	15.9	kΩ

< 稳定能力增加 >

若需高速AD转换而采样保持时间不满足外部信号输出阻抗允许值计算公式条件，则增加AIN引脚稳定容量。增加的容量取决于外电路。虽然取决于外电路的容量不同于各板组，电路板可自约 0.1 μF 至 1 μF，予以适量增加。

将拟增加的容量设于AIN引脚旁。

< 样品保持时间调整>

通常，通过设置长久采样保持时间，你可令ADC电路比较器的输入电压和AIN引脚的输入电压相同，减少AD转换错误。

虽然，如采样保持时间时间过长，由于采样保持电路容纳的电压改变，可能增加AD转换错误。

由于适当的采样保持时间取决于各板组，请决定板组适当的采样保持时间。

AD转换器使用的注意事项

AD转换结果值，视乎电源电压波动而异，或可能受噪音影响。交替使用模拟输入引脚和端口时，由于转换准确度可能减弱，切勿在转换期间读取和写入。若AD转换期间输出端口电流波动，转换准确度也可能减弱。请通过程序采取措施，如取AD转换结果平均值。



17. 闪存操作

本节描述硬件配置和闪存操作。本节中，"1个字符"为 32 位。

17.1 特征

17.1.1 存储器大小和配置

表 17-1 和 图 17-1 显示TMPM365FYXBG的内置存储器大小和配置。

表 17-1 存储器大小和配置

存储器大小	块配置				每页字符数目	页数	写入时间		擦除时间	
	128 KB	64 KB	32 KB	16 KB			1 页	整个区	块擦除	芯片擦除
256 KB	-	3	1	2	64	1024	1.25 ms	1.28 s	0.1 s	0.4 s

注：上述值为理论值，不包括数据传输时间。每个芯片写入时间取决于用户所用的写入方法。

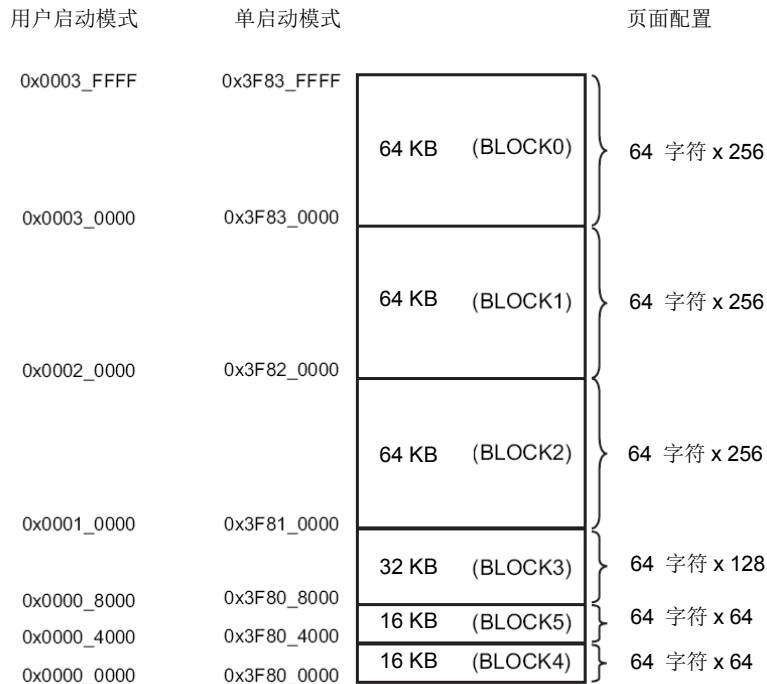


图 17-1 块配置

闪存配置单位描述为"块"和 "页".

- 页面

一页面为 64 字符。相同地址 [31:8]用于同一页面。组的第一地址为[7:0] = 0x00，组的最后一个地址为[7:0] = 0xFF。



• 块

一个块中的大小有16 KB, 32 KB 和 64 KB, 一个闪存包括若干不同大小的块。

按页进行写入操作。每页的写入时间为 1.25 ms。(典型值)

按块进行擦除(自动块擦除命令的使用)或对整个闪存进行擦除(自动芯片擦除命令的使用)。擦除时间因命令而已。若使用自动块命令, 每块擦除时间为 0.1秒(典型值)。若使用自动芯片擦除命令擦除整个区, 时间为 0.2 秒 (典型值)。

另外, 每个块可使用保护功能。保护功能详情, 参阅"17.1.5 保护/安全功能"。

17.1.2 功能

该装置内置的闪存, 除若干特殊功能外, 通常符合JEDEC标准。因此, 若用户当前正使用闪存作为外部存储器, 很容易在该装置内实行这些功能。而且, 为提供易写入或擦除操作, 该产品包括自动写入或芯片擦除的专用电路。

JEDEC兼容功能	修改, 添加或删除功能
<ul style="list-style-type: none"> • 自动编程 • 自动芯片擦除 • 自动块擦除 • 数据轮询/切换位 	<p><修改> 块写入/擦除保护 (只支持软件保护)</p> <p><删除> 擦除恢复 - 暂停功能</p>

17.1.3 运行模式

17.1.3.1 模式说明

该装置提供了单芯片模式和单启动模式。单芯片模式包括正常模式和用户启动模式。图 17-2 显示了模式的转换。

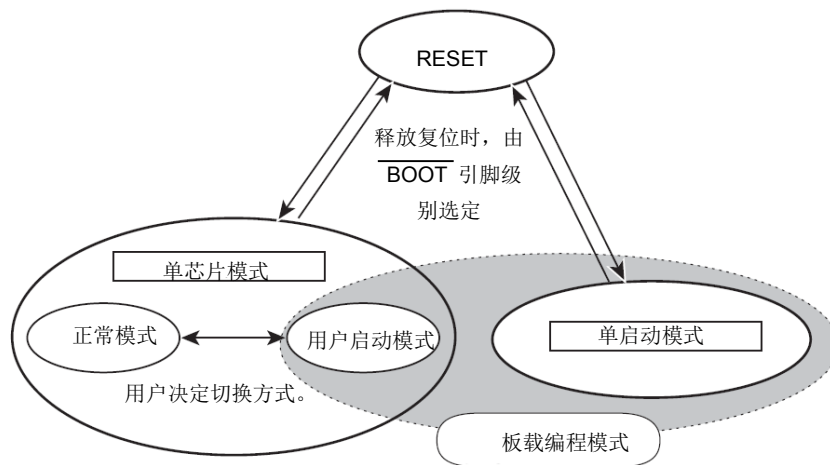


图 17-2 模式转换

(1) 单芯片模式

单芯片模式为复位后装置可从闪存启动的一种模式。该模式包括以下两种子模式。

- 正常模式
用户应用程序被执行的一种模式。
- 用户启动模式
闪存根据用户设置重新编程的一种模式。

用户可自由将正常模式切换到用户启动模式。例如，用户可设置为若端口A的PA0为"1"，模式为正常模式。若端口A的PA0为"0"，模式为用户启动模式。用户须在应用程序中准备例行程序，以对切换进行判断。

(2) 单启动模式

复位后闪存可从内置BOOT ROM (掩膜ROM)启动的一种模式。BOOT ROM包括可通过按用户设置的该装置串口重写闪存的算法。通过将串口连接至外表主机，可在上述协议和重新编程后的闪存进行数据传输。

(3) 板载编程模式

用户启动模式和单启动模式是闪存可按用户设置重新编程的模式。这两种模式被称为"板载编程模式"。

17.1.3.2 模式判断

复位释放时，通过BOOT引脚级别，可选择单芯片或单启动运行模式。

表 17-2 运行模式设置

运行模式	引脚	
	$\overline{\text{RESET}}$	$\overline{\text{BOOT}}$
单芯片模式	0 → 1	1
单启动模式	0 → 1	0

17.1.4 存储器地址

图 17-3 所示为单芯片模式和单启动模式中存储器映像的比较。单启动模式中，内置闪存地址至 0x3F80_0000 和后续地址，内置BOOT ROM地址至 0x0000_0000 ~ 0x0000_0FFF。

闪存和RAM地址如下所示。

FLASH 大小	RAM 大小	FLASH 地址	RAM 地址
256 KB	24 KB	0x0000_0000 ~ 0x0003_FFFF(单模式) 0x3F80_0000 ~ 0x3F83_FFFF(单启动模式)	0x2000_0000 ~ 0x2000_5FFF

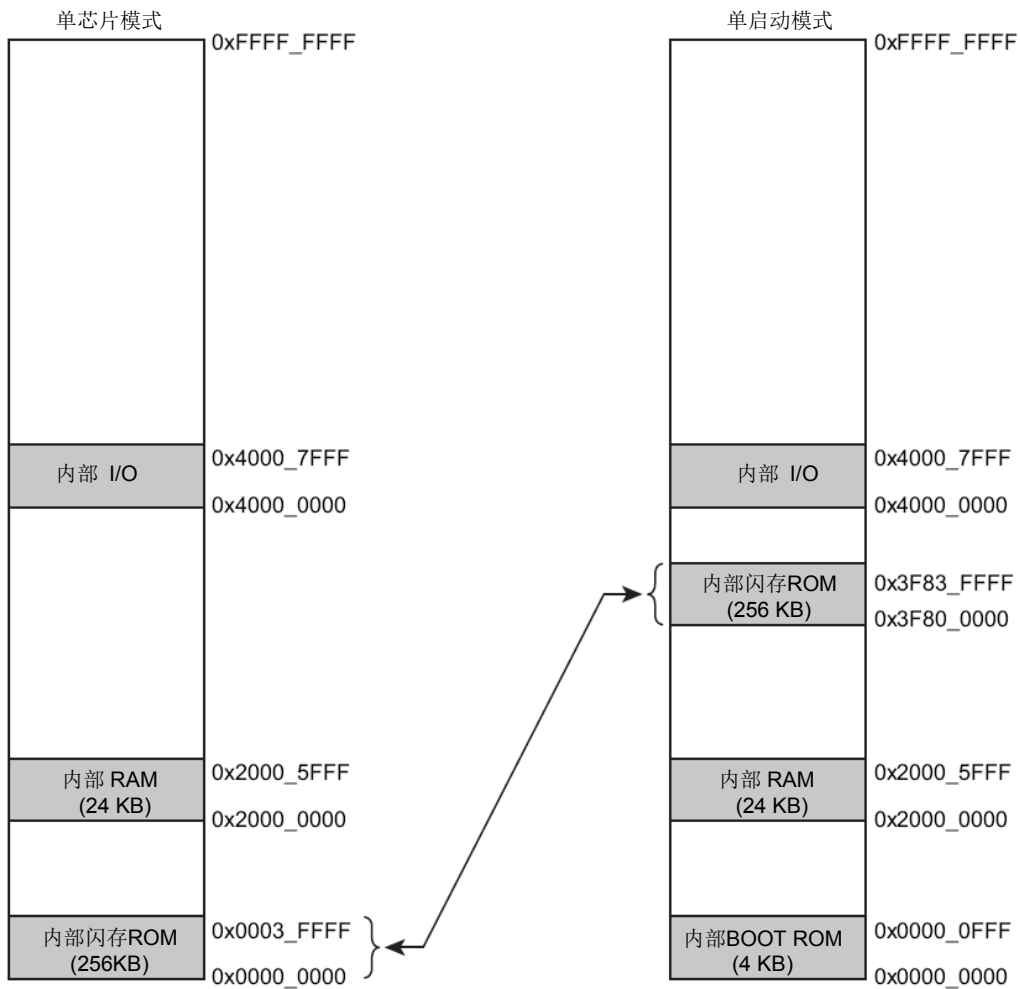


图 17-3 存储器地址比较

17.1.5 保护/安全功能

该装置对闪存具保护和安全管理功能。

1. 保护功能

写入/擦除操作可按块控制。

2. 安全功能

读取操作可从闪存程序写入器进行控制。除错功能的使用限制

17.1.5.1 保护功能

该功能按块对对写入/擦除操作进行控制。

启用该保护功能时，使用保护位编程命令，将相应块的保护位设置至"1"。若使用保护位擦除命令，将保护位设置至"0"，可取消块保护。通过FCFLCS<BLPRO[3:0]>可监控保护位。

保护位程序可通过 1 位单元进行编程，通过 4 位单元进行擦除。保护位编程/擦除详情，参阅"17.2.5 命令说明"。

17.1.5.2 安全功能

表 17-3 所示为安全功能启用时的操作。

表 17-3 安全功能启用时的操作。

项目	说明
读取闪存	CPU 能读取闪存
调试端口	JTAG, 串行线或追踪通信禁用。
命令执行至闪存	不接收命令写入闪存。若用户尝试擦除保护位，芯片擦除执行，所有保护位删除。

以下情况下启用安全功能：

1. FCSECBIT<SECBIT>设置至"1"。
2. 所有保护位 (FCFLCS<BLPRO>)设置至"1"。

通过冷复位，FCSECBIT <SECBIT> 设置至"1"。以下说明 FCSECBIT <SECBIT> 的重写。

注：以下项目 1 和 2 写入操作时，采用 32-位转移指令。

1. 写入指定代码 (0xa74a9d23) 到 FCSECBIT
2. 项目 1 运行后在 16 个块内写入数据。

17.1.6 寄存器

17.1.6.1 寄存器列表

基址 = 0x41FF_F000

寄存器名称		地址(基+)
安全位寄存器	FCSECBIT	0x0010
闪存控制寄存器	FCFLCS	0x0014

注：除上述地址列表外，切勿访问0x41FF_F000 ~ 0x41 FF_FFFF地址。

17.1.6.2 FCFLCS (闪存控制寄存器)

	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	-	-	BLPRO5	BLPRO4	BLPRO3	BLPRO2	BLPRO1	BLPRO0
复位后	0	0	(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	类型	功能
31-22	-	R	读作"0"。
19-21	BLPRO3- BLPRO0	R	块 5 至 0 保护状态 0: 无防护 1: 保护 保护元数值对应各块保护状态。若对应位显示"1"，对应块为保护状态。 保护状态下块无法重新编程。
15-1	-	R	读作"0"。
0	RDY/BSY	R	就绪/占空 (注 1) 0: 占空 (自动运行期间) 1: 就绪 (自动运行结束) 该位为自CPU对闪存进行监控的功能位。闪存自动运行时，该位输出"0"显示闪存占空。自动运行结束时，该位变为就绪状态，输出"1"。然后，接收下一命令。 若自动运行结果为失败，该位持续输出"0"。通过硬件复位，该位回到"1"。

注 1: 发出命令前，确保闪存就绪。若占空时下达命令，不仅命令无法发出，而且后续命令可能无法接收。这种情况下，使用硬件复位返回。硬件复位需要 0.5 μs或更多复位时间，和系统时钟无关。此时，直至复位后启用读取约需 2 ms。

注 2: 数值有相应保护状态。

17.1.6.3 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	类型	功能
31-1	-	R	读作"0"。
0	SECBIT	R/W	安全位 0: 安全功能设置禁用。 1: 安全功能设置启用。

注：通过冷复位对该寄存器进行初始化。

17.2 闪存详细说明

板载编程中，CPU执行编程命令或擦除闪存。该重新编程/擦除控制程序应由用户事先准备。由于闪存写入或擦除时闪存内容无法读取，需在内置RAM运行重新编程/擦除控制程序。除复位外，切勿生成中断/错误，以免程序异常终止。

17.2.1 功能

除一些特殊功能外，闪存通常符合JEDEC标准。然而，操作命令的地址指定方法和标准命令不同。

若执行写入/擦除操作，采用 32-位(1 字符)存储指令命令，将命令输入到闪存。命令输入后，内部自动执行写入或擦除操作。

表 17-4 闪存功能

主要功能	说明
自动页面程序	自动写入数据。
自动芯片擦除	自动擦除闪存整个区。
自动块擦除	自动擦除选定块。
写入/擦除保护	各块写入或擦除操作可单独控制。

17.2.2 闪存运行模式

闪存提供了两种主要运行模式：

- 读取存储器数据的模式（读取模式）
- 自动擦除或重写存储器数据的模式（自动运行模式）

通电后，休息后或自动运行模式正常完成后，闪存变成读取模式。读取模式中，闪存中存储的指令或读取的数据被执行。

若在读取模式期间输入命令，运行模式变为自动运行。若命令程序正常完成，运行模式返回读取模式，ID-读取命令除外。自动运行期间，无法对数据进行读取，闪存中存储的指令无法执行。

若命令程序异常结束，运行模式应强制返回读取模式。这种情况下，使用读取命令，读取/复位命令或硬件复位。

17.2.3 硬件复位

硬件复位指的是，自动编程/擦除操作强制取消，或自动运行异常结束时，使用冷复位或热复位返回读取模式。

若自动运行期间发生硬件复位，闪存停止自动运行，返回读取模式。若闪存自动程序/擦除操作期间发生硬件复位，硬件复位需 0.5 μ s或更长复位时间，和系统时钟无关。此时，直至复位后启用读取约需 2 ms。请注意，若自动运行期间发生硬件复位，无法正确进行数据写入操作。再次设置写入操作。

复位操作详情，参阅"复位"。给定复位输入后，CPU将读取复位矢量数据，然后启动复位后例行程序。

17.2.4 如何执行命令

利用存储指令，将命令序列写入闪存，以便命令得到执行。结合输入地址和数据，闪存可执行各项自动运行命令。命令执行详情，参阅"17.2.5 命令说明"。

对闪存存储指令的执行，称为"总线写入周期"。各命令包括若干总线写入周期。在闪存中，按指定命令执行总线写入周期的地址和数据时，可执行自动命令操作。执行非指定命令的周期时，闪存停止执行命令，返回读取模式。

若你在命令序列期间取消命令，或输入不同命令序列，可执行读取命令或读取/复位命令。然后，闪存停止执行命令，返回读取模式。读取命令和读取/复位命令被称为"软件复位"。

写入命令序列结束时，自动运行启动， $FCFLCS<RDY/BSY>$ 设置至"0"。自动运行正常结束时， $FCFLCS<RDY/BSY> = "1"$ 设置，闪存返回读取模式。

自动运行期间不接收新命令序列。若你要停止命令运行，可使用硬件复位。自动运行异常结束时($FCFLCS<RDY/BSY>$ 保持为"0")，闪存仍然锁定，不会返回读取模式。要返回读取模式，可使用硬件复位。若硬件复位停止命令运行，命令未得到正常执行。

命令执行注意事项：

1. 识别命令时，命令启动前，命令序列需处于读取模式。确认各命令的首个总线写入周期之前已设置 $FCFLCS<RDY/BSY> = 1$ 。建议读取命令依次执行。
2. 从闪存外部执行各命令序列。
3. 通过 1-字符(32-位)数据传输指令按顺序执行各总线写入周期。
4. 各命令序列期间，切勿访问闪存。除复位外，切勿生成中断或错误。
5. 下达命令后，若地址或数据未正确写入，确保通过使用软件复位返回读取模式。

17.2.5 命令说明

本节说明的是各命令内容。具体命令序列，参阅"17.2.6 命令序列"。

17.2.5.1 自动页面程序

(1) 操作说明

自动页面程序按页面写入数据。程序将数据写入多页时，页面程序需逐页执行。不可跨页写入。

写入闪存是指"1"数据单元变成"0"数据。不可自"0"数据变成"1"数据单元。要自"0"数据变成"1"数据单元，需擦除操作。

各已擦除页只允许运行一次自动页面程序。"1"或"0"数据单元都不能两次或多次写入数据。若已写入一次的页面重写，自动页面程序在自动块擦除或自动芯片擦除命令执行后，需再次设置。

注 1: 不经擦除操作对同一页面两次或多次执行页面程序，可能会损坏装置。

注 2: 不可写入保护块。

(2) 如何设置

第 1 ~ 3 总线写入周期显示的是自动页面程序命令。

第 4 总线写入周期中，写入页面的第一地址和数据。第 5 及之后周期上，按序写入一页数据。数据以 1-字符为单位(32-位)写入。

若写入页面一部分，设置"0xFFFFFFFF"为数据，意思是无需写入完整一页面。

装置内部无自动检验操作执行。因此，要确保读取编程后的数据，以确认其已正确写入。

若自动页面程序异常终止，则那页面已写入失败。建议勿使用装置或勿使用包括失效地址在内的块。

17.2.5.2 自动芯片擦除

(1) 操作说明

自动芯片擦除执行对象为所有地址的存储器单元。若保护块包括在内，这些块不会被擦除。若所有块受保护，将无法执行自动芯片擦除操作，将在输入命令序列后返回读取模式。

(2) 如何设置

第 1 ~ 6 总线写入周期显示的是自动芯片擦除命令。命令序列输入后，自动芯片擦除操作启动。

装置内部无自动检验操作执行。因此，要确保读取数据，以确认其已正确写入。

若自动芯片擦除异常终止，则用块擦除功能指定失效块。之后建议勿使用失效块。

17.2.5.3 自动块擦除

(1) 操作说明

通过自动擦除命令对指定块执行擦除操作。若指定块受保护，擦除操作无法执行。

(2) 如何设置

第 1 ~ 5 总线写入周期显示的是自动块擦除命令。在第 6 总线写入周期中，指定拟擦除块。命令序列输入后，自动块擦除操作启动。

装置内部无自动检验操作执行。因此，应确保读取数据，以确认其已正确写入。
若自动芯片擦除异常终止，之后建议勿使用失效块。

17.2.5.4 自动保护位程序

(1) 操作说明

自动保护位程序一次将"1"写入保护位。要将保护位设置"0"，则使用自动保护位擦除命令。
保护功能详情，参阅"17.1.5 保护/安全功能"。

(2) 如何设置

第 1 ~ 第 6 总线写入周期显示的是自动保护位程序命令。在第 7 总线写入周期中，指定拟写入保护位。命令序列输入后，自动保护位程序启动。利用FCFLCS<BLPRO>检查是否写入操作正常终止。

17.2.5.5 自动保护位擦除

(1) 操作说明

自动保护位擦除命令操作取决于安全状态。安全状态详情，参阅 "17.1.5 保护/安全功能"。

- 非安全状态
清除指定保护位至"0"。保护位擦除以 4-位为单位进行。
- 安全状态
擦除闪存所有地址后，擦除所有保护位。

(2) 如何设置

第 1 ~ 6 总线写入周期显示的是自动保护位擦除命令。第 7 总线写入周期中，指定拟擦除保护位。命令序列输入后，自动保护位擦除操作启动。

非安全状态中，擦除指定保护位。利用FCFLCS<BLPRO>检查是否擦除操作正常终止。

安全状态中，擦除闪存位所有地址和保护。确认是否数据和保护位正常擦除。若需要，执行自动保护位擦除，自动芯片擦除或自动块擦除。

所有情况都和其它命令相同，FCFLCS<RDY/BSY> 在自动保护位擦除命令操作期间变为"0"。操作完成后，FCFLCS<RDY/BSY>变为"1"，闪存会返回读取模式。要退出该操作，需硬件复位。

17.2.5.6 ID读取

(1) 操作说明

ID读取命令可读取包括闪存类型和三类代码，如来源程序代码，装置代码和宏代码。

(2) 如何设置

第 1~3 总线写入周期显示的是ID读取命令。第 4 总线写入周期中，指定拟读取代码。
第 4 总线写入周期后，任意闪存区中读取操作获得代码。

ID读取可依次执行。第 4 总线写入周期和读取 ID数值可重复执行。

ID读取命令不会自动返回读取模式。要返回读取模式，执行读取命令，读取/复位命令或硬件复位。

17.2.5.7 读取命令和读取/复位命令(软件复位)

(1) 操作说明

让闪存返回读取模式的命令。

ID读取命令执行时，宏停止于当前状态，未自动返回读取模式。要从该状态返回读取模式，使用读取命令或读取/复位命令。它也用于在命令输入中时取消命令。

(2) 如何设置

第 1 总线写入周期显示的是读取命令。第 1~3 总线写入周期显示的是读取/复位命令。
任一命令序列执行后，闪存返回读取模式。

17.2.6 命令序列

17.2.6.1 命令序列表

表 17-5 所示为各命令中总线写入周期的地址和数据。

除ID读取命令的第 5 总线周期外，所有命令为"总线写入周期"。总线写入周期通过 32-位(1 字符)数据传输指令进行。(下表只显示了低 8 位数据。)

地址详情，参阅表 17-6。将以下数值用于

表 17-6 中Addr[15:9]列所述"命令"。

注1) 总是将"0"设置于地址位[1:0]。

注2) 根据闪存大小，将以下数值设置于地址位[19]。

存储器大小为 1 MB或更少：总设置至"0"

存储器大小超过 1 MB：若总线写入 1 MB区或更少，位设置至"0"。

若总线写入超过 1 MB区，位设置至"1"。

表 17-5 命令序列

命令	第 1 总线周期	第 2 总线周期	第 3 总线周期	第 4 总线周期	第 5 总线周期	第 6 总线周期	第 7 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	-	-	-	-
	0xAA	0x55	0xF0	-	-	-	-
ID读取	0x54XX	0xAAXX	0x54XX	IA	0xXX	-	-
	0xAA	0x55	0x90	0x00	ID	-	-
自动页面程序	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	-
自动保护位程序	0x54XX	0xAAXX	0x54XX	0x54XX	0xAAXX	0x54XX	PBA
	0xAA	0x55	0x9A	0xAA	0x55	0x9A	0x9A
自动保护位擦除	0x54XX	0xAAXX	0x54XX	0x54XX	0xAAXX	0x54XX	0xXX
	0xAA	0x55	0x6A	0xAA	0x55	0x6A	0x6A

补充说明事项

- IA:ID地址
- ID:ID数据
- PA:程序页面地址
- PD:程序数据 (32-位数据)

第 4 总线周期后, 每页面按地址顺序输入数据

- BA:块地址 (见表 17-7)
- PBA:保护位地址 (见表 17-8, 表1-9)

17.2.6.2 总线周期中地址位配置

表 17-6 连同"表17-5 命令序列"一起使用。

根据正常总线写入周期地址配置自第 1 总线周期开始对地址设置进行设置。

表 17-6 总线写入周期中地址位配置

地址	地址 [31:19]	地址 [18]	地址 [17]	地址 [16]	地址 [15]	地址 [14]	地址 [13:11]	地址 [10]	地址 [9]	地址 [8]	地址 [7:0]
正常命令	正常总线写入周期地址配置										
	闪存区	推荐"0"		命令				Addr[1:0] = "0" (固定) 其它位= "0" (recom-推荐)			
ID读取	IA: ID 地址 (ID读取第4总线写入周期地址设置)										
	闪存区	推荐"0"	ID 地址	Addr[1:0] = "0" (固定) 其它位= "0" (推荐)							
块擦除	BA: 块地址(块擦除第6总线写入周期地址设置)										
	块地址 (表17-7)						Addr[1:0] = "0" (固定) 其它位= "0" (推荐)				
自动页面程序	PA: 程序页面地址 (页面程序第4总线写入周期地址设置)										
	页面地址										Addr[1:0] = "0" (固定) 其它位= "0" (recom-推荐)
保护位程序	PBA: 保护位地址(保护位程序第7总线写入周期地址设置)										
	闪存区	保护位选择 (表17-8)	固定为 "0"				保护位选择 (表17-8)	Addr[1:0] = "0" (固定) 其它位= "0" (recom-推荐)			
保护位擦除	PBA: 保护位地址 (保护位擦除第7总线写入周期地址设置)										
	闪存区	保护位选择 (表17-9)	固定为 "0"				Addr[1:0] = "0" (固定) 其它位= "0" (推荐)				

17.2.6.3 块地址(BA)

表 17-7 所示为块地址。指定自动块擦除命令第 6 总线写入周期中拟擦除块所含地址。

表 17-7 块地址

块	地址 (用户启动模式)	地址 (单启动模式)	大小 (K字节)
4	0x0000_0000 ~ 0x0000_3FFF	0x3F80_0000 ~ 0x3F80_3FFF	16
5	0x0000_4000 ~ 0x0000_7FFF	0x3F80_4000 ~ 0x3F80_7FFF	16
3	0x0000_8000 ~ 0x0000_FFFF	0x3F80_8000 ~ 0x3F80_FFFF	32
2	0x0001_0000 ~ 0x0001_FFFF	0x3F81_0000 ~ 0x3F81_FFFF	64
1	0x0002_0000 ~ 0x0002_FFFF	0x3F82_0000 ~ 0x3F82_FFFF	64
0	0x0003_0000 ~ 0x0003_FFFF	0x3F83_0000 ~ 0x3F83_FFFF	64

17.2.6.4 如何指定保护位 (PBA)

编程时保护位以1位为单位进行指定，擦除时则以4-位为单位。

表 17-8 和表 17-9 所示为自动保护位程序的保护位选择表。地址示例列表明，上侧所述地址用于单次模式，而下侧用于单启动模式。

合计自动保护位擦除命令擦除四个保护位。

表 17-8 保护位程序地址

块	保护位	第 7 总线写入周期地址							地址示例 [31:0]	
		地址 [18]	地址 [17]	地址 [16]	地址 [15:11]	地址 [10]	地址 [9]	地址 [8]		
块0	<BLPRO[0]>	0	0	固定为 "0"				0	0	0x0000_0000 0x3F80_0000
块1	<BLPRO[1]>	0	0					0	1	0x0000_0100 0x3F80_0100
块2	<BLPRO[2]>	0	0					1	0	0x0000_0200 0x3F80_0200
块3	<BLPRO[3]>	0	0					1	1	0x0000_0300 0x3F80_0300
块4	<BLPRO[4]>	0	1					0	0	0x0002_0000 0x3F82_0000
块5	<BLPRO[5]>	0	1					0	1	0x0002_0100 0x3F82_0100

表 17-9 保护位地址

块	保护位	第 7 总线写入周期地址		地址示例 [31:0]
		地址 [18]	地址 [17]	
块0 - 3	<BLPRO[0:3]>	0	0	0x0000_0000 0x3F80_0000
块4 - 5	<BLPRO[4:5]>	0	0	0x0002_0000 0x3F82_0000

17.2.6.5 ID读取代码 (IA, ID)

表 17-10 所示为如何用ID读取命令指定代码和内容。

地址示例列表明，上侧所述地址用于单次模式，而下侧则用于单启动模式

表 17-10 ID读取命令代码和内容

代码	ID[7:0]	IA[13:12]	地址示例 [31:0]
制造来源授权标识码	0x98	0y00	0x0000_0000 0x3F80_0000
装置 code	0x5A	0y01	0x0000_4000 0x3F80_4000
-	保留	0y10	-
宏代码	0x13	0y11	0x0000_C000 0x3F80_C000

17.2.6.6 命令序列示例

(1) 单次模式

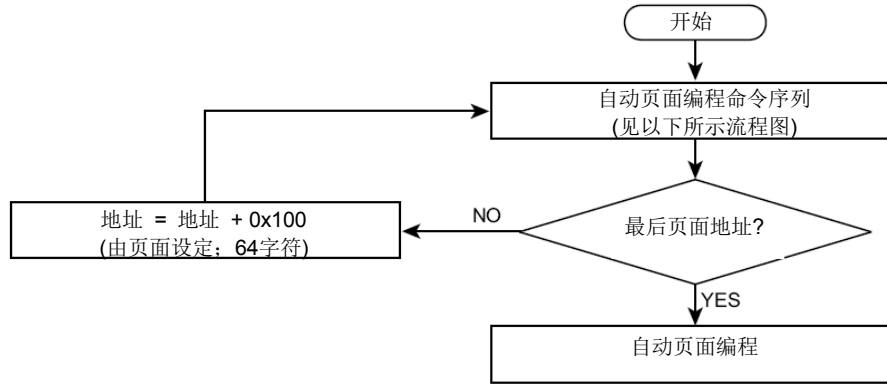
命令	总线周期							
		1	2	3	4	5	6	7
读取	地址	0x0000_0000	-	-	-	-	-	-
	数据	0x0000_00F0	-	-	-	-	-	-
读取/复位	地址	0x0000_5400	0x0000_0AA0	0x0000_5400	-	-	-	-
	数据	0x0000_00AA	0x0000_0055	0x0000_00F0	-	-	-	-
ID读取	地址	0x0000_5400	0x0000_0AA0	0x0000_5400	IA	0x0000_00XX	-	-
	数据	0x0000_00AA	0x0000_0055	0x0000_0090	0x0000_0000	ID	-	-
自动页面程序	地址	0x0000_5400	0x0000_0AA0	0x0000_5400	PA	下列周期中，每页依次写入地址和数据。		
	数据	0x0000_00AA	0x0000_0055	0x0000_00A0	PD			
自动芯片擦除	地址	0x0000_5400	0x0000_0AA0	0x0000_5400	0x0000_5400	0x0000_0AA0	0x0000_5400	-
	数据	0x0000_00AA	0x0000_0055	0x0000_0080	0x0000_00AA	0x0000_0055	0x0000_0010	-
自动块擦除	地址	0x0000_5400	0x0000_0AA0	0x0000_5400	0x0000_5400	0x0000_0AA0	BA	-
	数据	0x0000_00AA	0x0000_0055	0x0000_0080	0x0000_00AA	0x0000_0055	0x0000_0030	-
自动保护位程序	地址	0x0000_5400	0x0000_0AA0	0x0000_5400	0x0000_5400	0x0000_0AA0	0x0000_5400	PBA
	数据	0x0000_00AA	0x0000_0055	0x0000_009A	0x0000_00AA	0x0000_0055	0x0000_009A	0x0000_009A
自动保护位擦除	地址	0x0000_5400	0x0000_0AA0	0x0000_5400	0x0000_5400	0x0000_0AA0	0x0000_5400	PBA
	数据	0x0000_00AA	0x0000_0055	0x0000_006A	0x0000_00AA	0x0000_0055	0x0000_006A	0x0000_006A

(2) 数据单启动模式

命令	总线周期							
		1	2	3	4	5	6	7
读取	地址	0x3F80_0000	-	-	-	-	-	-
	数据	0x0000_00F0	-	-	-	-	-	-
读取/复位	地址	0x3F80_5400	0x3F80_0AA0	0x3F80_5400	-	-	-	-
	数据	0x0000_00AA	0x3F80_0055	0x3F80_00F0	-	-	-	-
ID读取	地址	0x3F80_5400	0x3F80_0AA0	0x3F80_5400	IA	0x0000_00XX	-	-
	数据	0x0000_00AA	0x0000_0055	0x0000_0090	0x0000_0000	ID	-	-
自动页面程序	地址	0x3F80_5400	0x3F80_0AA0	0x3F80_5400	PA	下列周期中，每页依次写入地址和数据。		
	数据	0x0000_00AA	0x0000_0055	0x0000_00A0	PD			
自动芯片擦除	地址	0x3F80_5400	0x3F80_0AA0	0x3F80_5400	0x3F80_5400	0x3F80_0AA0	0x3F80_5400	-
	数据	0x0000_00AA	0x0000_0055	0x0000_0080	0x0000_00AA	0x0000_0055	0x0000_0010	-
自动块擦除	地址	0x3F80_5400	0x3F80_0AA0	0x3F80_5400	0x3F80_5400	0x3F80_0AA0	BA	-
	数据	0x0000_00AA	0x0000_0055	0x0000_0080	0x0000_00AA	0x0000_0055	0x0000_0030	-
自动保护位程序	地址	0x3F80_5400	0x3F80_0AA0	0x3F80_5400	0x3F80_5400	0x3F80_0AA0	0x3F80_5400	PBA
	数据	0x0000_00AA	0x0000_0055	0x0000_009A	0x0000_00AA	0x0000_0055	0x0000_009A	0x0000_009A
自动保护位擦除	地址	0x3F80_5400	0x3F80_0AA0	0x3F80_5400	0x3F80_5400	0x3F80_0AA0	0x3F80_5400	PBA
	数据	0x0000_00AA	0x0000_0055	0x0000_006A	0x0000_00AA	0x0000_0055	0x0000_006A	0x0000_006A

17.2.7 流程图

17.2.7.1 自动程序



自动页面编程命令序列 (地址/ 命令)

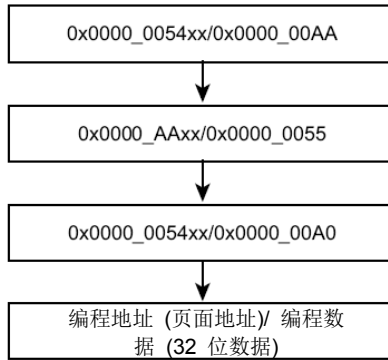


图 17-4 自动程序流程图

17.2.7.2 自动擦除

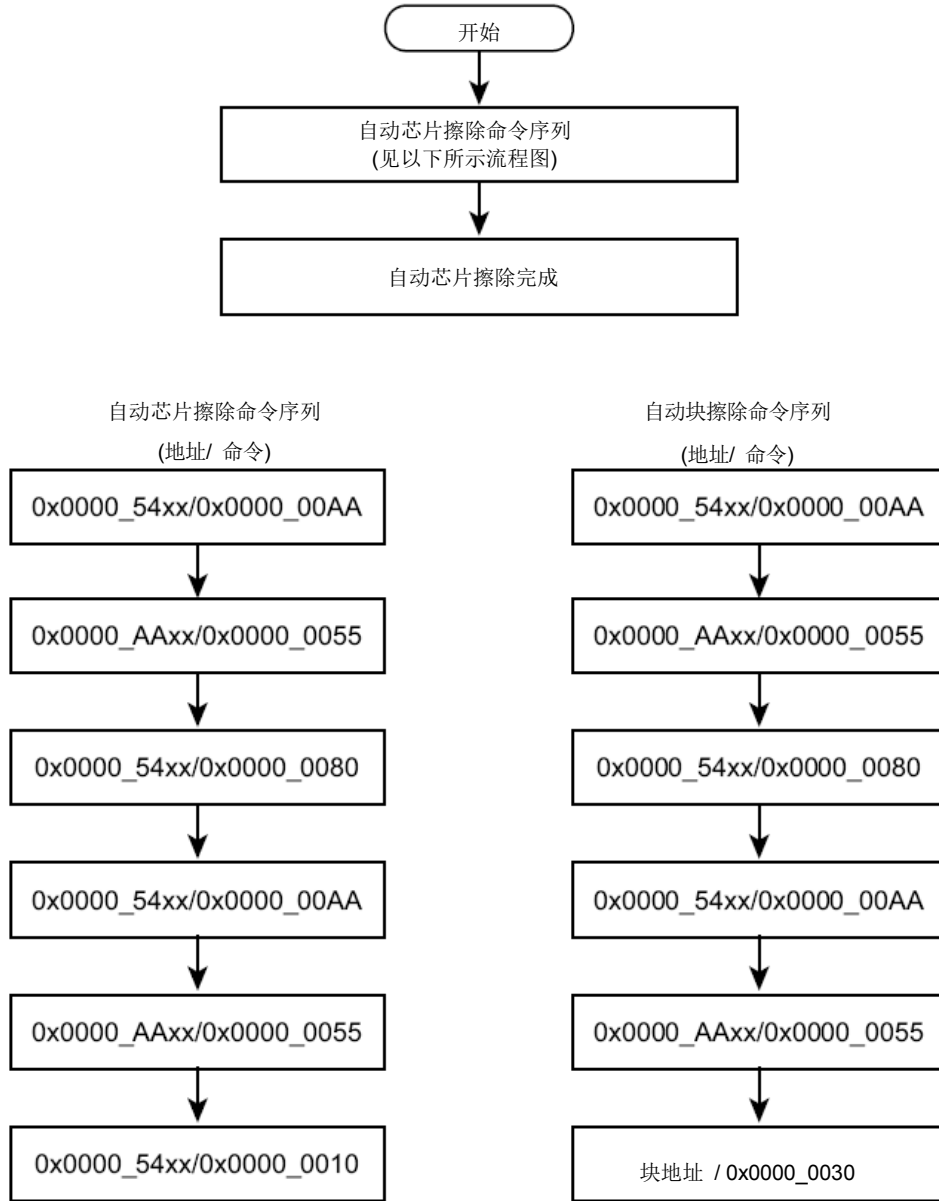


图 17-5 自动擦除流程图

17.3 如何使用单启动模式对闪存进行重新编程

单启动模式运用包含于内置 BOOT ROM 中的程序对闪存进行重新编程。这种模式中，BOOT ROM 地址到含中断矢量表区，闪存地址到非BOOT ROM区的另一地址区。

启动模式中，使用串行命令/数据传输对闪存重新编程。将该装置的串行通道(SIO/UART)连接到外部主机，重编程序从外部主机复制到内置RAM。对重编程序闪存执行RAM中的重编例行程序。主机通信详情，遵照后面所述协议。

即使在单启动模式中，除复位外，切勿生成中断/错误，以免程序异常终止。

要保护单芯片模式(正常操作模式)中闪存的内容，重新编程完成后，建议对相关闪存块进行保护，以防止随后单芯片操作时意外擦除。

17.3.1 模式设置

为执行板载编程，该装置为单启动模式启动。以下设置至单启动模式设置。

$$\overline{\text{BOOT}} = 0$$
$$\overline{\text{RESET}} = 0 \rightarrow 1$$

$\overline{\text{BOOT}}$ 引脚事先进行上述设置，复位引脚设置至"0"。然后，解除复位引脚，装置将以单启动模式启动。

17.3.2 接口规范

本节所述为单启动模式的SIO/UART通信格式。连续操作对UART (异步通信) 和 I/O接口模式都支持。为进行板载编程，还要对可编程控制器通信格式进行设置。

- UART 通信

通信通道：通道0

串行传输模式：UART (异步)，半双工，LSB优先

数据长度：8-位

奇偶校验位：无。

STOP位：1-位

波特率：任意波特率

- I/O 接口模式

通信通道：通道0

串行传输模式：I/O 接口，全双工，LSB优先

同步信号(SCLK0)：输入模式上升缘

握手信号：PE4 (输出模式)

波特率：任意波特率

启动程序操作时钟/模式控制块作为初始条件的设置。时钟初始设置详情，参阅"时钟/模式控制"。

正如"17.3.5.1 连续操作模式确定"所阐述的，波特率取决于 16 位定时器(TMRB)。对波特率进行判断时，由所需波特率 1/16 进行通信。因此，通信波特率须在可测量范围内。定时器计数时钟操作于 $\Phi T1$ ($f_c/2$)。

I/O接口模式的握手引脚在接收状态等待时输出"低"，在传输状态时输出"高"。通信前检查握手引脚，且须遵照通信协议。

表 17-11 所示为用于启动程序的引脚。除这些引脚外，不可用于启动程序。

表 17-11 引脚连接

引脚		接口	
		UART	I/O接口模式
模式设置引脚	\overline{BOOT}	o	o
复位引脚	\overline{RESET}	o	o
通信引脚	TXD0 (PE0)	o	o
	RXD0 (PE1)	o	o
	SCLK0 (PE2)	x	o (输入模式)
	PE4	x	o (输出模式)

o:已用 x:未用

17.3.3 内存储器限制

注意：单启动模式对内置RAM和内置闪存有限制，如表 17-12 所示。

表 17-12 单启动模式中对存储器的限制

存储器	限制
内部RAM	通过0x2000_0000 ~ 0x2000_03FF，存储器被用作启动程序的工作区。通过RAM结束地址保存程序0x2000_0400。
内部闪存	以下地址分配用于保存软件ID信息和密码。不建议将程序保存于下列地址。0x3F83_FFF0 ~ 0x3F83_FFFF

注：若密码 为已擦除数据(0xFF)，由于是容易猜测的密码，很难保护到数据的安全。即使不用单启动模式，也建议设置个唯一的数值作为密码。

17.3.4 运行命令

启动程序提供有以下运行命令。

表 17-13 运行命令数据

运行命令数据	运行模式
0x10	RAM传输
0x40	闪存芯片擦除和保护位擦除

17.3.4.1 RAM传输

RAM传输将数据自控制器保存到内置RAM。传输正常完成时，用户程序启动。除用于启动程序的0x2000_0000 ~ 0x2000_03FF外用，户程序可使用0x2000_0400存储地址或之后的部分。CPU将从 RAM存储开始地址开始执行。

该RAM传输功能启用用户限定的板载编程控制。为执行用户程序的板载编程，使用17.2.6所述闪存命令序列。

17.3.4.2 闪存芯片擦除和保护位擦除

闪存芯片擦除和保护位擦除命令擦除闪存整个块和所有块的写入/存储保护，和写入/存储保护或安全状态无关。命令完成时，FCSECBIT<SECBIT>设置至"1"。

17.3.5 和命令无关的常规操作

本节所述为启动程序执行时的常规操作。

17.3.5.1 连续操作模式确定

控制器通过UART进行通信时，于所需波特率将第 1 字节设置至0x86。控制器通过I/O接口模式通信时，于所需波特率 1/16 将第 1 字节设置至0x30。图 17-6 所示为各情况下的波形。

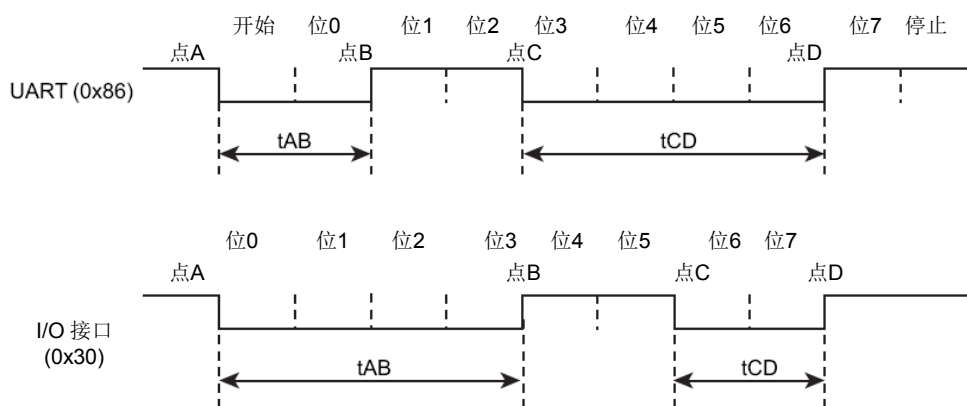


图 17-6 连续操作模式确定数据

图 17-7 所示为启动程序的流程图。通过 t_{AB} ， t_{AC} 和 t_{AD} 的时间运用 16 位定时器 (TMRB)，复位后，提供有连续操作模式确定数据的第 1 字节 (0x86, 0x30)。图 17-7 中，CPU监控接收引脚的电平，在接收引脚电平改变时，获得一定时器值。因此， t_{AB} ， t_{AC} 和 t_{AD} 的定时器值存在误差范围。此外，注意：若于高波特率传输，CPU 可能无法确定接收引脚电平。尤其是，I/O接口因为其波特率通常要比UART的高很多，容易产生此问题。为避免这个问题，控制器应于I/O接口模式中所需波特率的 1/16 传送数据。

图 17-8 中流程图表明，连续操作模式取决于接收引脚时间长度的长或短。若长度为 $t_{AB} \leq t_{CD}$ ，连续操作模式确定为UART 模式。无论自动波特率设置是否启用， t_{AD} 的时间已被使用。若长度为 $t_{AB} > t_{CD}$ ，连续操作模式确定为I/O 接口模式。注意： t_{AB} ， t_{AC} 和 t_{AD} 的定时器值存在误差范围。若波特率高，操作频率低，各定时器值会变小。这可能导致意外确定发生。(要防止该问题，在编程程序内对进行UART重置。)

例如，连续操作模式在预期模式模式为UART模式时可能确定为I/O接口模式。为避免该问题，使用UART模式时，控制器应考虑到超时期限，届时预计将接收到来自目标板的回应(0x86)。控制器若无法在允许时间内获得回应，应停止通信。运用I/O接口模式时，传输第一串行字节后，控制器应在一段空闲时间后发送SCLK时钟，以获得确认响应。若接收到的确认响应并非0x30，控制器应停止进一步通信。

预期模式为I/O接口模式时，无需第一字节为0x30，只要如上所示 $t_{AB} > t_{CD}$ 。0x91，0xA1或0xB1可作为第一字节代码发送，以确定点A和点C的下降缘和点B和点D的上升缘。若 $t_{AB} > t_{CD}$ 成立，SIO由操作模式确定的解析度选定，即使第一字节传输的代码并非，0x30，第二字节代码为，0x30(确定I/O接口模式的第一字节代码被描述为，0x30)。

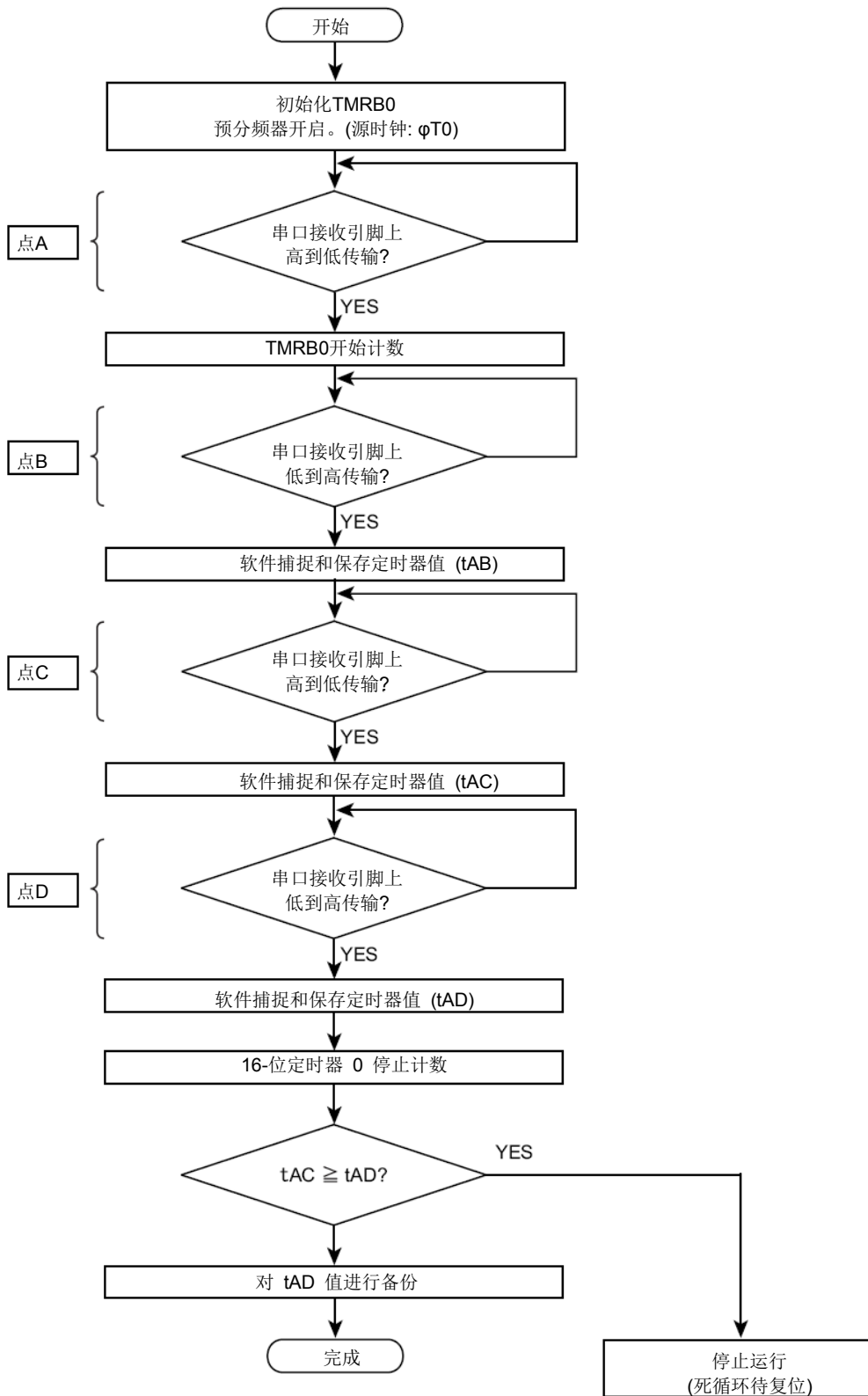


图 17-7 连续操作模式接收流程图

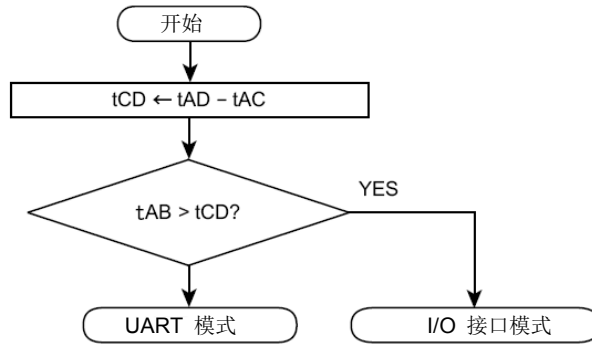


图 17-8 连续操作模式确定流程图

17.3.5.2 确认响应数据

启动程序以特定码表示进程状态，并将其发送到控制器。表 17-14 至表 17-17 所示为对各接收数据的确认响应值。

表 17-15 至表 17-17 中，确认响应的高四位和操作命令数据的相等。第3位所示为接收错误。第 0 位所示为失效操作命令错误，检验和错误或密码错误。第 1 位和第 2 位总为"0"。I/O 接口模式中不执行接收错误检查。

表 17-14 对连续操作确定数据的ACK响应

传送数据	说明
0x86	确定UART 通信可能。(注)
0x30	确定I/O接口通信可能。

注：连续操作确定为UART，若波特率设置确定为不可接收，启动程序无需发回响应即中止。

表17-15 对操作命令数据的ACK响应

传送数据	说明
0x?8 (注)	操作命令数据中发生接收错误
0x?1 (注)	可正常接收未定义运算的命令数据。
0x10	确定为RAM传输 命令
0x20	确定为闪存SUM 命令
0x30	确定为装置代码读取命令
0x40	确定为闪存芯片擦除命令

注：ACK响应数据的高4位和之前命令数据的那些相同。

表 17-16 对CHECK SUM数据的ACK响应

传送数据	说明
0xN8 (注)	发生接收错误。
0xN1 (注)	发生CHECK SUM或密码错误。
0xN0 (注)	CHECK SUM值正确。

注：ACK响应数据的高 4 位和运行命令数据的那些相同。

表 17-17 对闪存芯片擦除和保护位擦除操作的ACK响应

传送数据	说明
0x54	确认为擦除启用命令
0x4F	擦除命令完成。
0x4C	擦除命令异常终止。

17.3.5.3 密码确定

启动程序使用以下区确定是否需要密码或用作密码。

区域	地址
密码需要确定	0x3F83_FFF0 (1 字节)
密码区	0x3F83_FFF4 ~ 0x3F83_FFFF (12字节)

RAM传输命令进行密码验证，和必要性判断数据无关。闪存芯片擦除或保护位擦除命令只在必要性判断确定为"需要"时才进行密码验证。

密码要求设置	数据
需要密码	非0xFF
无密码	0xFF

若密码设置至0xFF（已擦除数据），由于是容易猜测的密码，很难保护到数据的安全。即使不用单启动模式，也建议设置个唯一的数值作为密码。

(1) 使用RAM传输命令进行密码验证

若所有这些地址位置包含非0xFF的相同数据字节，该情况确定为密码区错误，如图 17-9 所示。这种情况下，启动程序还错误确认(0x11)，响应检验和的第 17 个字节，和密码验证无关。

启动程序对接收数据(密码数据)的第 5 个字节至第 16 个字节进行验证。若所有 12 个字节不相匹配，发生密码错误。若确认密码错误，对CHECK SUM 数据的第 17 个(字节)的ACK响应数据为密码错误。

即使安全功能启用，也可进行密码验证。

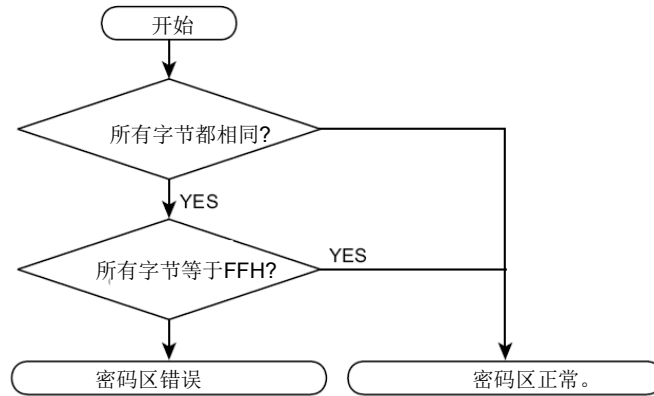


图 17-9 密码区检查流程图

(2) 对闪存芯片擦除和保护位擦除命令的密码验证

密码启用擦除密码必要性确定区时，如图 17-10 所示，密码为完全相同数据，密码区错误发生。若确认密码区错误，对CHECK SUM的第 17 个字节的ACK响应发送 0x41，和密码验证无关。

启动程序对接收数据(密码数据)的第 5 个字节至第 16 个字节进行验证。若所有 12 个字节不相匹配，发生密码错误。若确认密码错误，对CHECK SUM 数据的第 17 个(字节)的ACK响应数据为密码错误。

即使安全功能启用，也可进行密码验证。

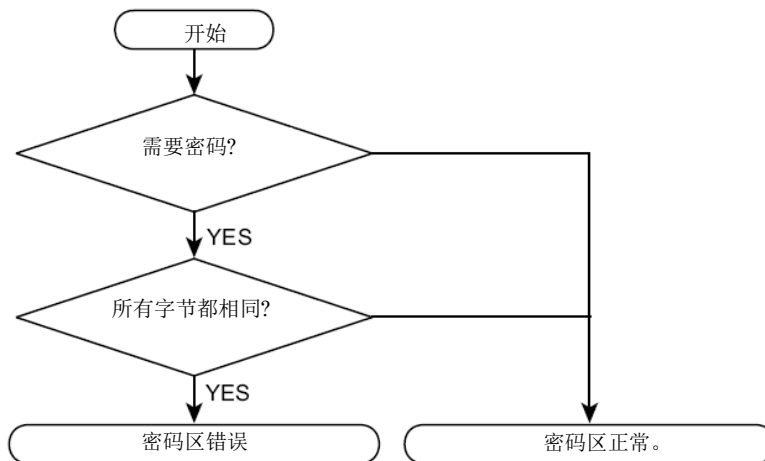


图 17-10 密码区检查流程图

17.3.5.4 CHECK SUM计算

检验和采用 8-位加法进行计算传输数据，去掉进位，取总和的二进制补码。控制器传输检验和字节时，须进行相同的检验和操作。

CHECK SUM示例

要计算 0xE5 和 0xF6 系列的检验和，运行 8 位加法。

$$0xE5 + 0xF6 = 0x1DB$$

取总和二进制补码至低 8-位，即为检验和值。因此启动程序发送 0x25 至控制器。

$$0 - 0xDB = 0x25$$

17.3.6 RAM传输的转换格式

本节所示为RAM传输命令格式。表中传送方向如下所示：

传输方向 (C→T):控制器至TMPM365FYXBG

传输方向 (C←T):TMPM365FYXBG至控制器

传输位数	传输方向	传输数据	说明
1	C→T	连续操作模式和波特率设置	发送数据确定连续操作模式。模式确定详情，参阅"17.3.5.1 连续操作模式确定"。
		[UART 模式] 0x86	发送0x86。若UART模式确定，程序判断是否波特设置可能。若否，程序停止，通信关闭。
		[I/O接口模式] 0x30	于所需波特率的 1/6 发送 0x30。第 2 个字节也于所需波特率的 1/6 发送。从第 3 个字节或之后开始，你可以于所需波特率发送数据。
2	C←T	对连续操作模式的ACK响应	传输数据的第 2 个字节，为对响应连续操作设置模式数据的第1字节的ACK响应数据。若设置可能，设置SIO/ UART。传输缓冲写入数据前，已设置接收启用时机。
		[UART 模式] 正常状态:0x86	若设置确定为可能，发送0x86。若否，操作中止，无需发回响应。控制器完成发送数据的第 1 个字节，需超时时间(5秒)。若超时时间内数据 (0x86) 未正常接收，通信不可能。
		[I/O 接口模式] 正常状态:0x30	将数据 (0x30) 传输缓冲，等待SCLK0时钟。控制器完成发送数据的第1个字节，若干ms (空闲时间)后，输出SCLK时钟。此时，波特率设置至所需波特率的1/16。若接收数据为 0x30，通信可能。第 3 个字节后，设置至所需波特率进行通信。
3	C→T	操作命令数据 (0x10)	发送RAM传输命令数据(0x10)。
4	C←T	对操作命令的ACK响应 正常状态: 0x10 异常状态: 0xX1 通信错误: 0xX8	对操作命令的ACK响应数据。 首先，检查是否接收数据的第 3 字节存在错误。(仅限于UART模式) 若存在接收错误，发送ACK响应数据0xX8，意为异常通信，等待下一操作命令(第 3 字节)。传输数据高 4 位未定义。(和紧接操作命令前的高4位相同。)注意：I/O接口中，不进行接收错误检查。 然后，若接收数据的第3字节响应 表17-13 的任一操作命令，接收数据获响应。RAM传输时，0x10获响应，传输数据分至RAM传输服务例行程序。若数据未响应表 17-13 的命令，发送ACK响应数据0xX1，意为操作命令错误，等待下一操作命令(第 3 字节)。传输数据高 4 位未定义。(和紧接操作命令数据前的高 4 位相同。)
5 to 16	C→T	密码数据 (12 字节) 0x3F83_FF04 ~ 0x3F83_FF0F	检查密码区的数据。密码区检查详情，参阅 "17.3.5.3 密码确定"。 通过闪存数据的0x3F83_FFF0~0x3F83_FFFF比较接收数据的第5字节至第16字节。若数据和地址不匹配，设置密码错误标志。
17	C→T	CHECK SUM 值的第 5 个~ 第 16 个字节	发送CHECK SUM值的第 5 ~ 第 16 个字节。 CHECK SUM 计算详情，参阅17.3.5.4。

传送字节数	传输方向	传输数据	说明
18	C←T	对CHECK SUM数值的ACK回应 正常状态: 0x10 异常状态: 0x11 通信错误: 0x18	首先, 检查是否接收数据的第 5 ~ 第 17 字节存在错误。(仅限于UART模式) 若存在接收错误, 发送ACK回应数据0x18, 意为通信异常, 等待下一操作指令(第 3 字节)。然后, 检查CHECK SUM数据的第 17 字节。若错误存在, 发送0x11, 等待下一操作指令(第 3 字节)。最后, 检查密码验证结果。若存在密码错误, 发送ACK回应数据 0x11 意为密码错误, 等待下一操作指令(第 3 字节)。若所有步骤正常结束, 发送正常ACK回应数据 0x10。
19	C→T	RAM存储开始地址 31 ~ 24	发送块传输开始地址, 用于RAM存储。第 19 字节对应地址的第 31 ~ 24 位。第 22 字节对应地址的第 7 ~ 0 位。指定至RAM地址 0x2000_0400 直至最后地址的地址。
20	C→T	RAM存储开始地址 23 ~ 16	
21	C→T	RAM存储开始地址 15 ~ 8	
22	C→T	RAM存储开始地址 7 ~ 0	
23	C→T	RAM存储字节数 15 ~ 8	设定字节数, 以进行块传输。第 23 字节对应传输字节的第 15 位~第 8 位。第 24 字节对应传输字节的第 7 位~第 0 位。指定拟存储于自RAM地址 0x2000_0400 直至最后地址的数据。
24	C→T	RAM存储字节数 7 ~ 0	
25	C→T	CHECK SUM 数值的第 19 个 ~ 24 字节	发送CHECK SUM值的第 19 字节 ~ 24 字节。
26	C←T	对CHECK SUM数值的ACK回应 正常状态: 0x10 异常状态: 0x11 通信错误: 0x18	首先, 检查是否接收数据的第 19 ~ 第 25 字节存在错误。(仅限于UART模式) 若存在接收错误, 发送ACK回应数据 0x18, 意为通信异常, 等待下一操作指令(第 3 字节)。然后, 检查CHECK SUM数据的第25字节。若错误存在, 发送0x11, 等待下一操作指令(第 3 字节)。若所有步骤正常结束, 发送正常ACK回应数据 0x10。
27 to m	C→T	RAM已存数据	发送用于RAM已存储数据, 于第23字节至24字节中指定数据的相同字节。
m+1	C→T	CHECK SUM 数值的 27 ~ m 字节	发送CHECK SUM值的第 27 字节至 m 字节
m+2	C←T	对CHECK SUM数值的ACK回应 正常状态: 0x10 异常状态: 0x11 通信错误: 0x18	首先, 检查是否接收数据的第 27 ~ m+1 字节存在错误。(仅限于UART模式) 若存在接收错误, 发送ACK回应数据0x18, 意为通信异常, 等待下一操作指令(第 3 字节)。然后, 检查CHECK SUM数据的m+1字节。若错误存在, 发送 0x11, 等待下一操作指令(第 3 字节)。若所有步骤正常结束, 发送正常ACK回应数据 0x10。
-	-	-	若ACK回应数据的第m+2 字节为正常ACK回应数据, 传输数据分至第 19 字节至第 22 字节指定的地址。

17.3.7 闪存芯片擦除 和保护位擦除的转换格式

本节所示为闪存芯片擦除和保护位擦除命令。表中传送方向如下所示:

传送方向 (C→T): 控制器至 TMPM365FYXBG

传送方向 (C←T): TMPM365FYXBG ~ 控制器

传送字节数	传输方向	传输数据	说明
1	C→T	连续操作模式和波特率设置	发送数据确定连续操作模式。模式确定详情，参阅"17.3.5.1 连续操作模式确定"。
		UART模式 0x86	发送0x86。若UART模式确定，检查是否波特率设置可完成。若否，则进行停止通信操作。
		I/O接口模式 0x30	于所需波特率的 1/16 发送 0x30。和第 1 字节相同，于所需波特率的 1/16 发送第 2 字节。所需波特率 可用于第 3 字节之后。
2	C←T	对连续操作模式的ACK响应	传输数据的第 2 个字节，为对响应连续操作设置模式数据的第1字节的ACK响应数据。若设置可能，设置SIO/ UART。传输缓冲写入数据前，已设定接收启用时机。
		[UART 模式] 正常状态: 0x86	若设置确定为可能，发送0x86。若否，操作中止，无需发回响应。控制器完成发送数据的第 1 个字节，需超时间(5秒)。若超时间内数据 (0x86) 未正常接收，通信不可能。
		[I/O 接口模式] 正常状态: 0x30	将数据 (0x30) 传输缓冲，等待SCLK0时钟。控制器完成发送数据的第1个字节，若干ms (空闲时间)后，输出SCLK时钟。此时，波特率设置为所需波特率的1/16。若接收数据为 0x30，通信可能。第 3 个字节后，设置为所需波特率进行通信。
3	C→T	操作命令数据(0x40)	发送 闪存芯片擦除和保护位擦除命令数据(0x40)。
4	C←T	对操作命令的ACK回应 正常状态: 0x40 异常状态: 0xX1 通信错误: 0xX8	对操作命令的ACK响应数据。 首先，检查是否接收数据的第 3 字节存在错误。(仅限于UART模式) 若存在接收错误，发送ACK响应数据0xX8，意为异常通信，等待下一操作命令(第 3 字节)。传输数据高 4 位未定义。(和紧接操作命令数据前的高 4 位相同。)注意: I/O接口中，不进行接收错误检查。 然后，若接收数据的第3字节响应 表17-13 的任一操作命令，接收数据获响应。若数据和表 17-13 中命令不符，发送ACK回应数据0xX1，意为操作命令错误，等待下一个操作命令。(第 3 字节) 传输数据高 4 位未定义。(使用了和紧接操作命令前的高4位。)
5 ~ 16	C→T	密码数据 (12 字节) 0x3F83_FF04 ~ 0x3F83_FF0F	若密码必要性设置为"无"，该数据为虚拟数据。 若密码必要性设置为"必要"，检查密码区数据。密码区数据检查方法，参阅"17.3.5.3 密码确定"。 将接收数据的第 5 ~ 16 字节和闪存数据的0x3F83_FFF0 ~ 0x3F83_FFFF按顺序进行比较。若数据不相匹配，设定密码错误标志。
17	C→T	CHECK SUM值的第5 ~ 16 字节	发送CHECK SUM值的第 5 ~ 16 字节。 CHECK SUM计算方法，参阅"17.3.5.4 CHECK SUM计算"。
18	C←T	对CHECK SUM值的ACK回应 正常状态: 0x40 异常状态: 0x41 通信错误: 0x48	若密码必要性设置为"无"，发送正常ACK回应数据 0x40。若密码必要性设置为"必要"，首先检查是否接收数据第 5 字节 ~ 17 字节存在接收错误。(仅限于UART 模式) 若接收错误存在，发送ACK回应数据 0x48意为通信异常，等待下一个操作命令(第 3 字节)。 然后，检查CHECK SUM数据的第 17 字节。若错误发生，发送 0x41，等待下一个操作命令(第 3 字节) 最后，检查密码验证结果。若 密码错误存在，发送ACK回应数据 0x41，意为密码错误，等待下一个操作命令(第 3 字节) 若所有步骤正常结束，发送正常ACK回应数据 0x40。
19	C→T	擦除启用命令数据 (0x54)	发送启用命令数据(0x54)。
20	C←T	对擦除启用命令的ACK回应 正常状态: 0x54 异常状态: 0xX1 通信错误: 0x58	首先，检查是否接收数据的第19字节存在错误。若接收错误存在，发送ACK回应数据(第 3 位) 0x58，意为通信异常，等待下一个操作命令(第 3 字节)。 然后，若接收数据的第 19 字节和擦除启用命令不符，接收数据获回应(正常ACK回应数据)。这种情况下，0x54 获回应，传输数据分至闪存芯片擦除例行程序。 若数据和擦除启用命令不符，发送ACK回应数据 (第 0 位) 0xX1，等待下一个操作命令。传输数据高 4 位未定义。(使用了和紧接操作命令前的高4位。)
21	C→T	对擦除命令的ACK回应 正常状态: 0x4F 异常状态: 0x4C	若操作正常完成，结束码(0x4F)被返回。 若擦除错误发生，错误码(0x4C) 被返回。
	-	-	等待下一个操作命令。

17.3.8 启动程序完整流程图

本节所示为启动程序全流程图。

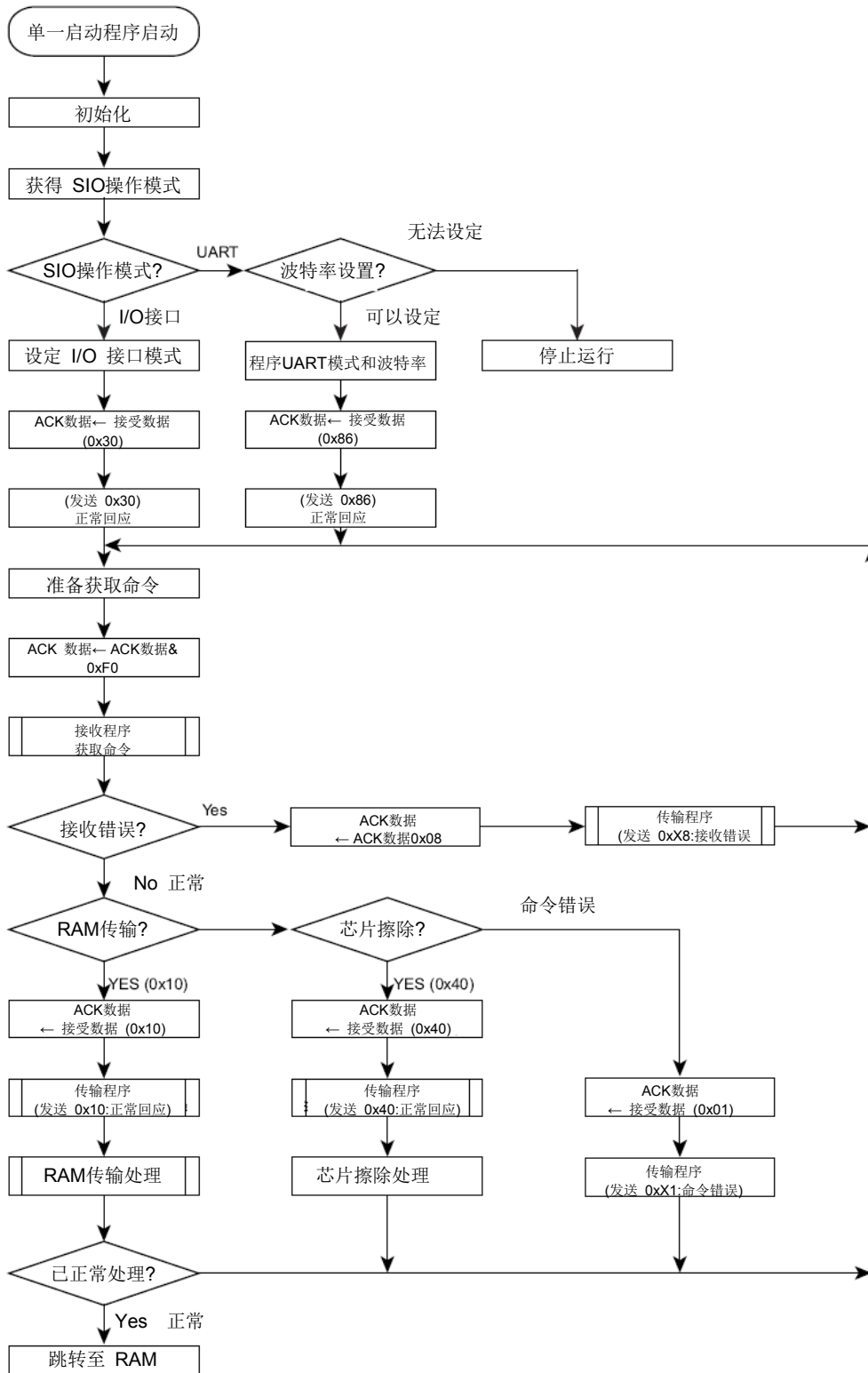


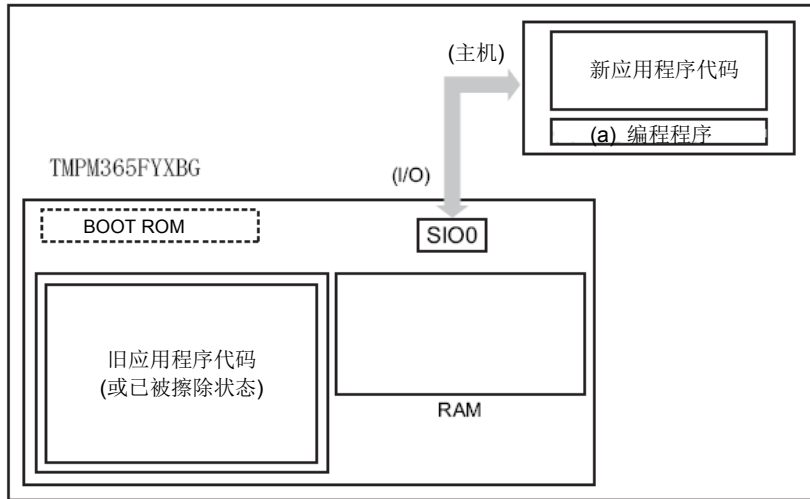
图 17-11 启动程序完整流程图

17.3.9 片上BOOT ROM中使用重编程序算法重编闪存程序

本节所述为片上启动ROM中使用重编程序算法重编闪存程序。

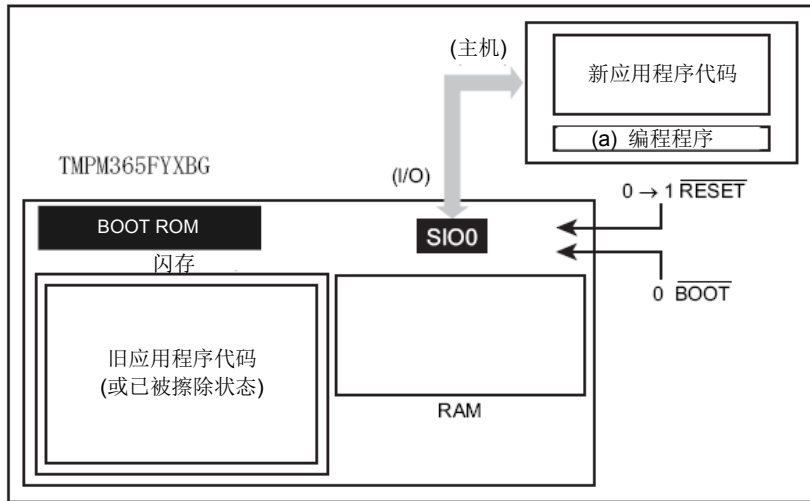
17.3.9.1 步骤-1

闪存条件不在乎是否前版构成的用户程序已写入或擦除。既然编程程序编程数据通过SIO (SIO0)被传输，SIO0必定已连接至外部主机。编程程序 (a) 在主机上已准备好。



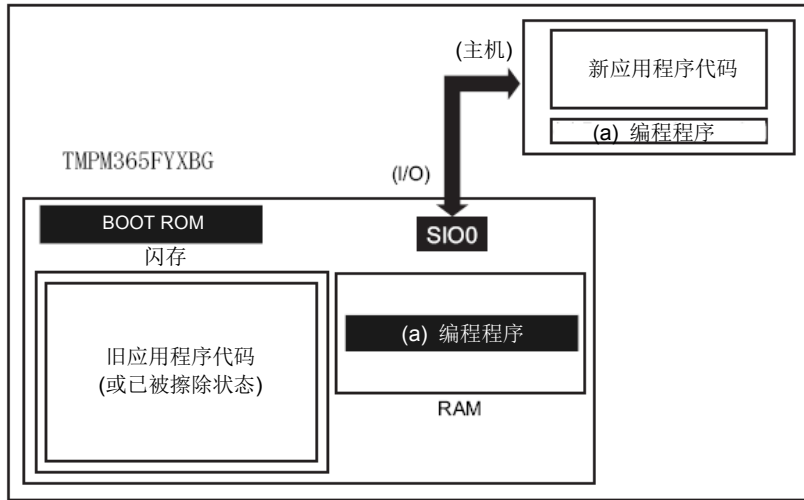
17.3.9.2 步骤-2

通过启动模式中的引脚条件设置释放复位，启动BOOT ROM。根据启动模式步骤，将编程程序 (a)通过SIO0自源(主机)进行传输。用户应用程序有密码的，进行密码验证。(若闪存被擦除，通过密码对擦除数据 (0xFF)进行处理。)



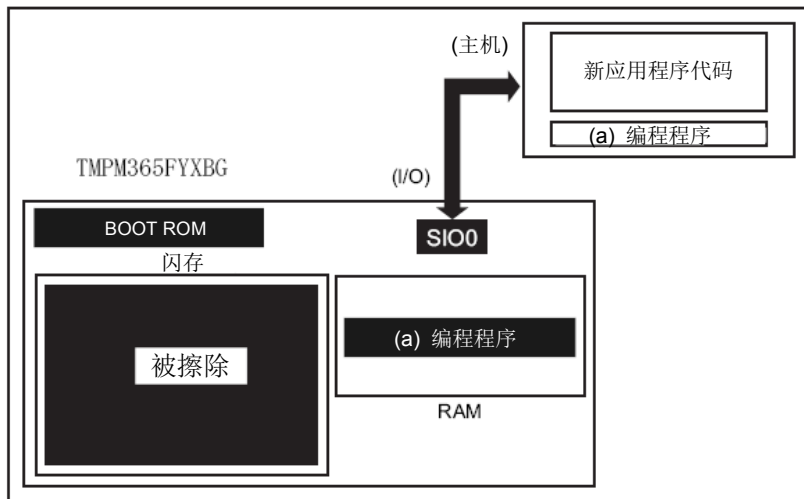
17.3.9.3 步骤-3

若完成密码验证，启动程序自主机将编程程序 (a) 传输入片上RAM。编程程序须存储于自 0x2000_0400 至RAM结束地址范围内



17.3.9.4 步骤-4

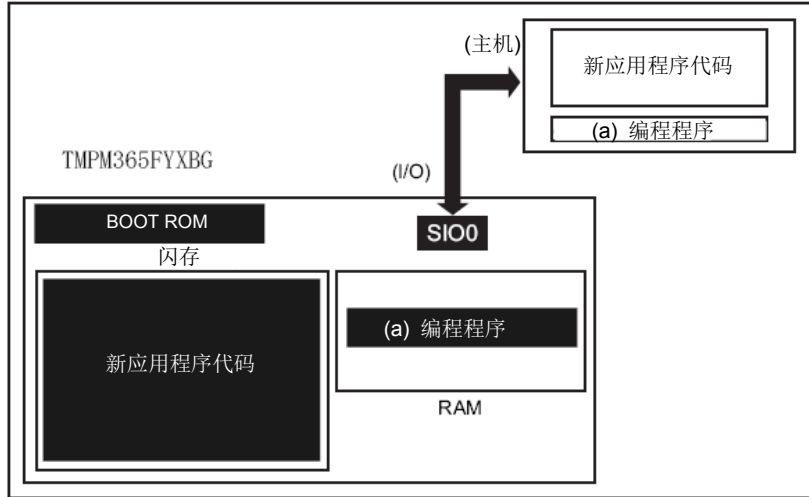
启动程序跳转至片上RAM内的编程程序(a)，以擦除含旧应用程序代码的闪存块。使用块擦除或芯片擦除命令



17.3.9.5 步骤-5

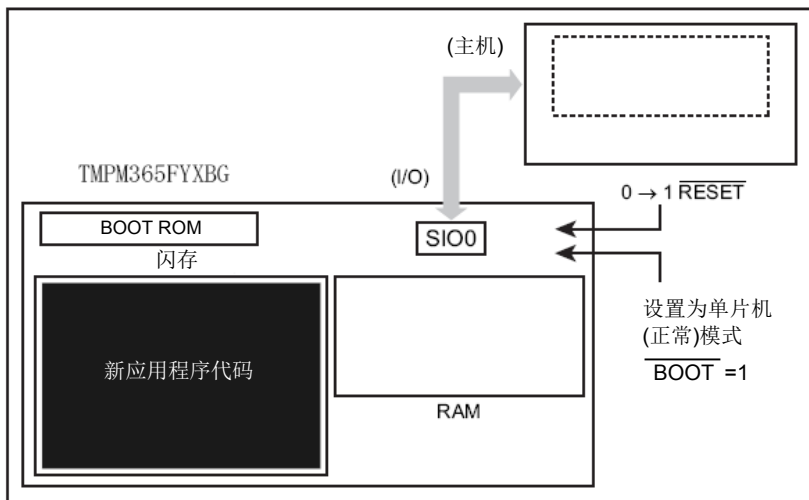
启动程序执行编程程序(a)自主机下载新应用程序代码，并将其编程至已擦除闪存区。编程完成时，须在用户程序中设定写入或擦除该闪存区保护。

以下示例中，新程序代码通过相同SIO0通道出自相同主机用于编程程序。但是，一旦编程程序已开始执行，可自由更换传输路径和传输源。创建硬件板卡并编程程序以满足你的特定需求。



17.3.9.6 步骤-6

闪存编程完成时，令板卡断电，并断开主机和目标板卡之间的连接线。再次通电，令装置以单片机(正常)模式重启以执行新程序。



17.4 用户启动模式下编程

用户启动模式为使用用户定义的闪存编程程序。用于用户应用程序上的闪存程序代码数据传输总线不同于串行I/O之时。运行于单片机模式；因此，单片机模式中，需自被激活的用户应用程序正常模式切换至用户编程闪存启动模式。尤其是，在用户应用程序中增加了模式判断程序至复位服务程序。

需根据用户系统设置情况设定切换模式的条件。而且，用户独特编制的闪存编程程序需设定于新应用程序中。该程序在切换至用户启动模式后用于编程。内置闪存的数据在擦除/重编程模式时无法读出。因此，重编程程序须在存储于闪存区以外的区中进行。一旦重编程完成，建议保护相关闪存块，以避免意外重编程。确保切勿生成复位外的中断/错误，以免用户启动模式中异常终止。

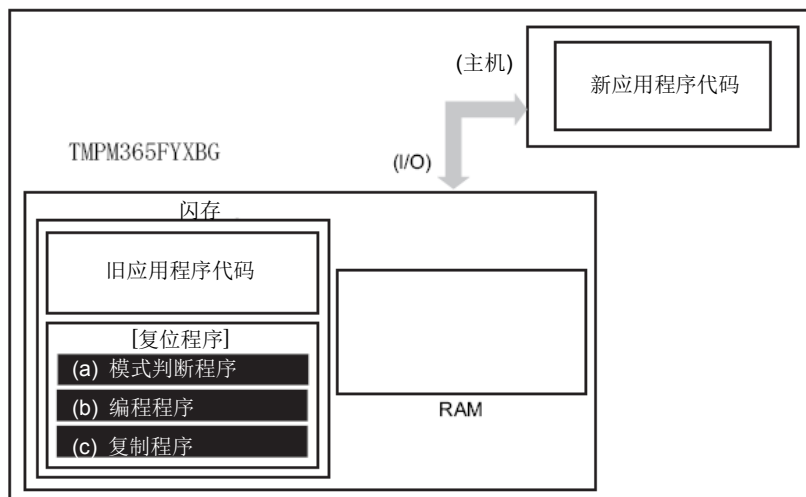
选取两种情况作为示例，如存储于闪存 (1-A)，传输自外部装置 (1-B)的重编程程序方法，下节对该步骤予以阐述。编程/擦除至闪存详情，参阅"17.2 闪存详细说明"。

17.4.1 (1-A) 编程程序存储于闪存的步骤

17.4.1.1 步骤-1

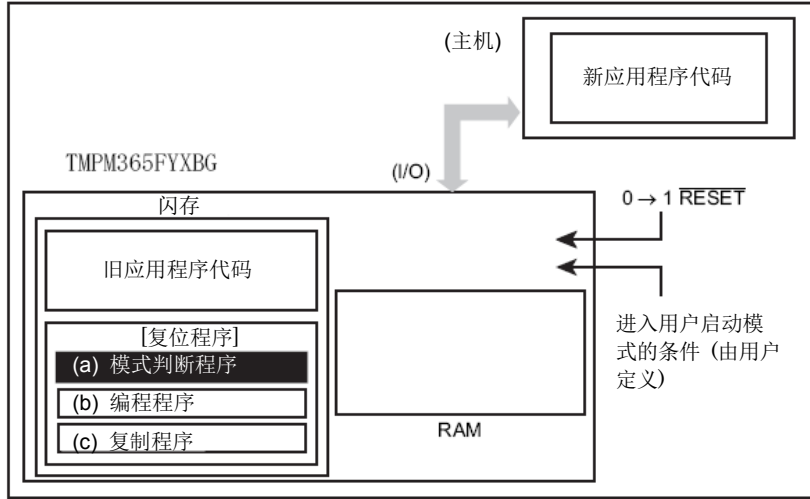
用户确定进入用户启动模式和拟用于传输数据的I/O总线的条件(如引脚状态)。然后创建适当的电路设计和程序。将装置安装于印刷电路板之前，用编程设备如闪存程序写入器将以下三个编程程序写入任意闪存块。

- | | |
|-------------|-----------------------------|
| (a) 模式判断程序: | 确定是否切换至用户启动模式的程序 |
| (b) 闪存编程程序: | 自主机控制器和重编程闪存下载新程序的程序。 |
| (c) 复制程序: | 将(b)中所述数据复制至内置RAM或外部存储装置的程序 |



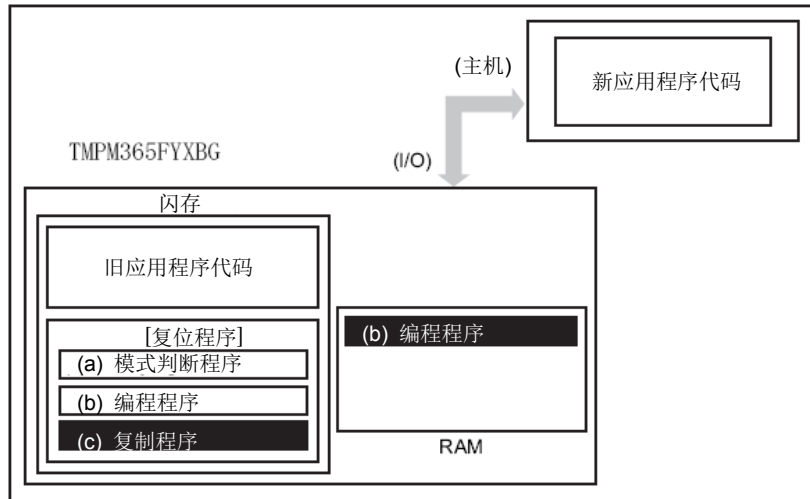
17.4.1.2 步骤-2

本节阐述了存储于复位程序的编程程序的情况。首先，复位程序确定进入用户启动模式。若模式切换条件符合，装置进入用户启动模式对数据进行重编程。



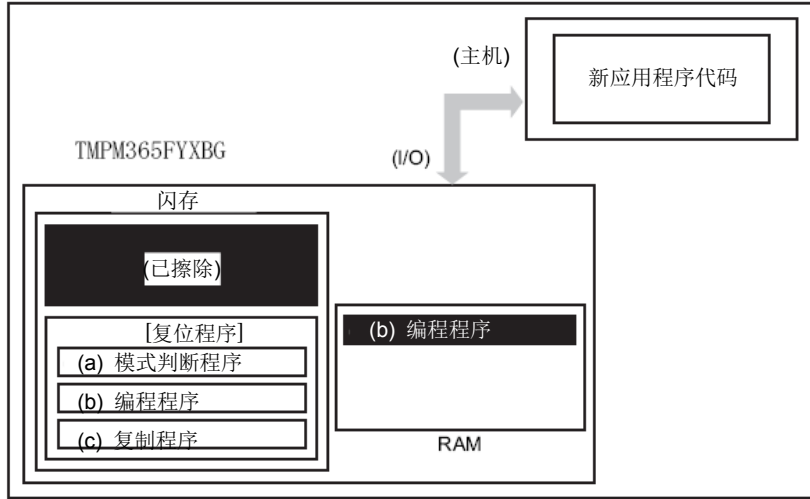
17.4.1.3 步骤-3

一旦装置进入用户启动模式，执行复制程序 (C)，自主机控制器下载闪存编程程序 (b)至内置RAM。



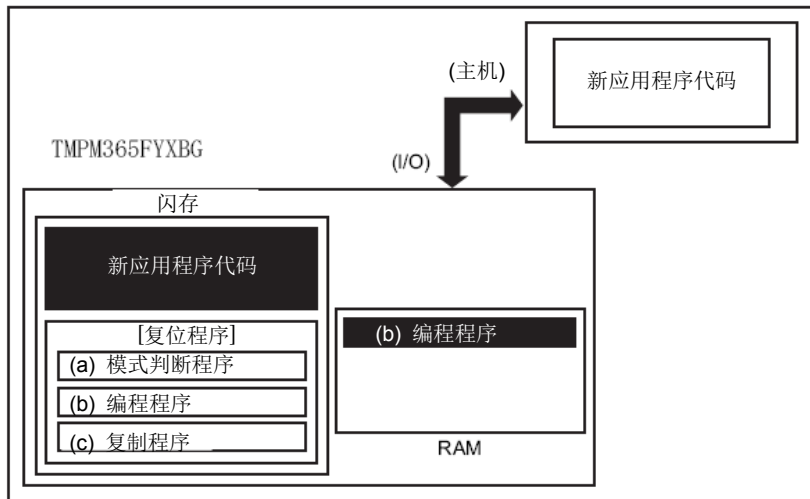
17.4.1.4 步骤-4

跳转至在内置RAM中的重编程程序，释放旧应用程序的写入/擦除保护，擦除块单元中的闪存。



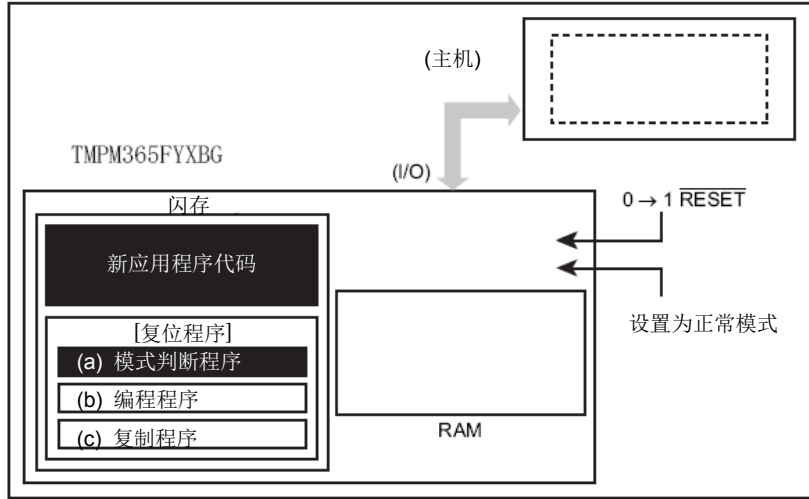
17.4.1.5 步骤-5

继续执行闪存编程程序，自主机控制器下载新程序数据，将其编程至已擦除闪存块。编程完成后，须设定用户程序区闪存块的写入/擦除保护。



17.4.1.6 步骤-6

设复位为"0"。一旦复位，闪存设置为正常模式。复位后，CPU 将随新应用程序启动。



17.4.2 (1-B)编程程序自外部主机进行传输的步骤

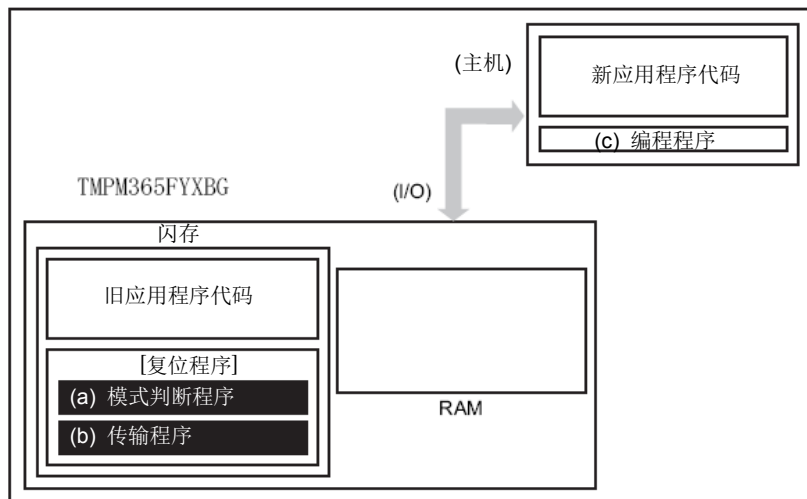
17.4.2.1 步骤-1

用户确定进入用户启动模式和拟用于传输数据的I/O总线的条件(如引脚状态)。然后创建适当的电路设计和程序。将装置安装于印刷电路板之前，使用编程设备如闪存程序写入器将以下两个调试程序写入任意闪存块。

- (a) 模式判断程序: 确定是否切换至重编程操作的程序
- (b) 传输程序: 自外部装置获取重编程程序的程序

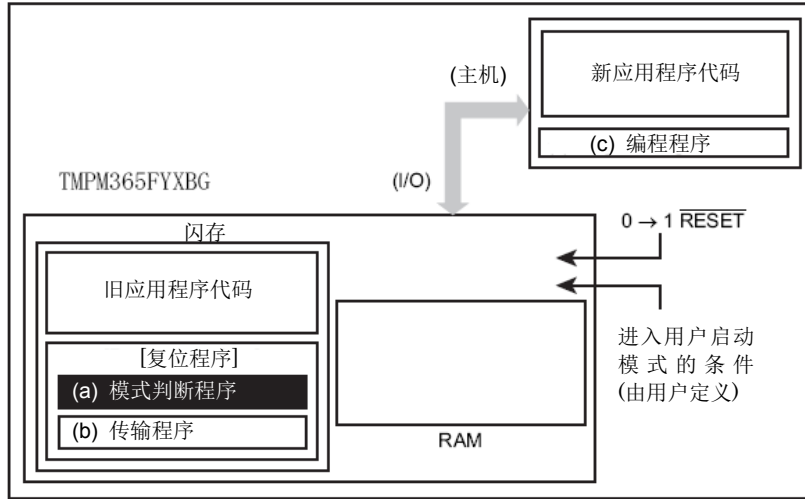
另外，准备须存储于主机控制器，如下所示的重编程程序。

- (c) 重编程程序: 对数据进行重编程的程序



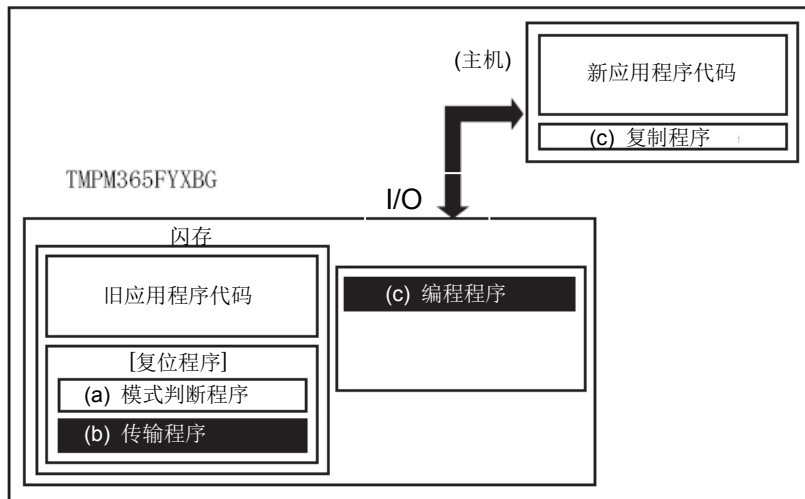
17.4.2.2 步骤-2

本节所述为编程程序 存储于复位程序的情况。首先，复位程序确定进入用户启动模式。若模式切换条件符合，装置进入用户启动模式对数据进行重编程。



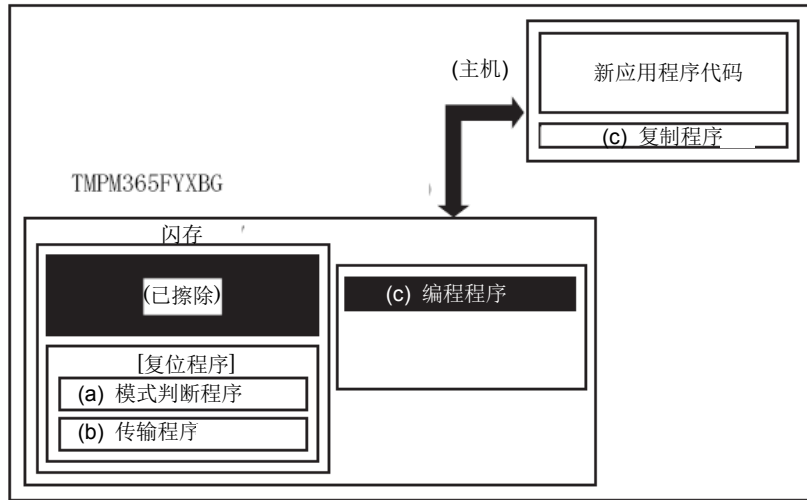
17.4.2.3 步骤-3

一旦装置进入用户启动模式，执行传输程序(b)，自主机控制器下载编程程序 (c)至内置RAM。



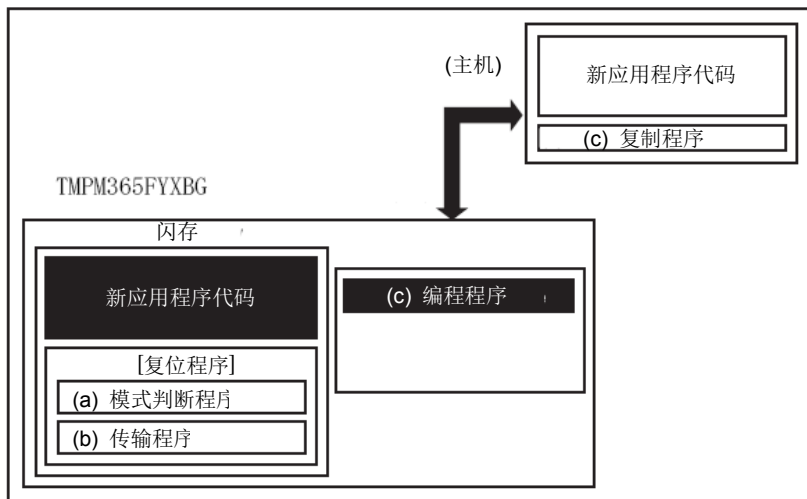
17.4.2.4 步骤-4

跳转至在内置RAM中的重编程程序，释放旧应用程序的写入/擦除保护，擦除块单元中的闪存。



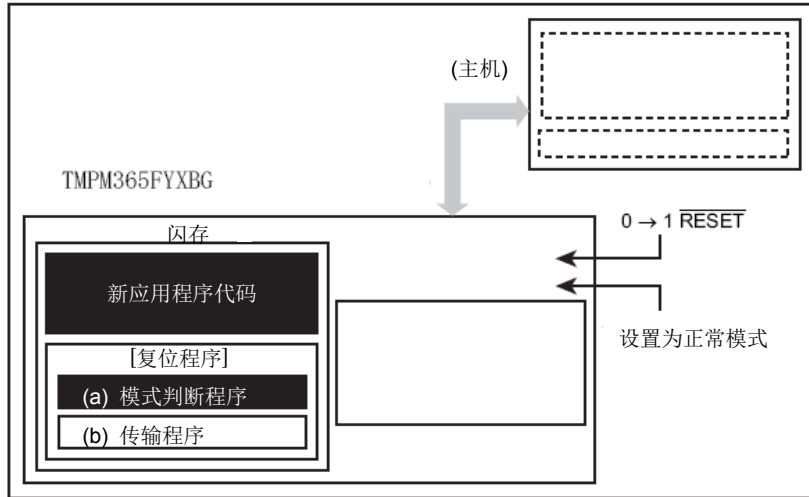
17.4.2.5 步骤-5

继续执行闪存编程程序(c)，自主机控制器下载新程序数据，将其编程至已擦除闪存块。编程完成后，须设定用户程序区闪存块的写入/擦除保护。



17.4.2.6 步骤-6

设 $\overline{\text{RESET}}$ 为"0"。一旦复位，闪存设置为正常模式。复位后，CPU 将随新应用程序启动。

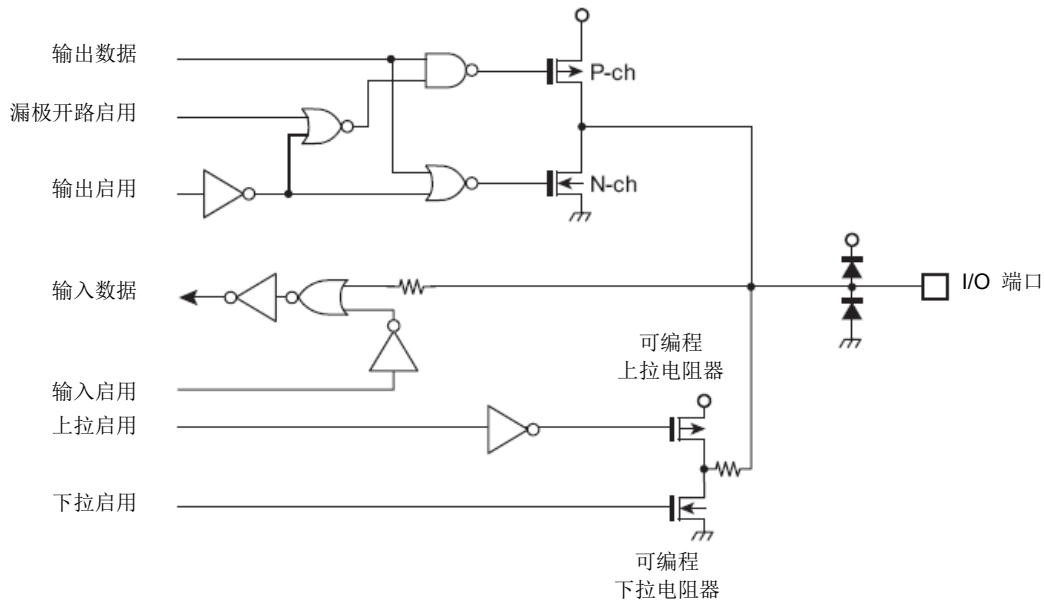




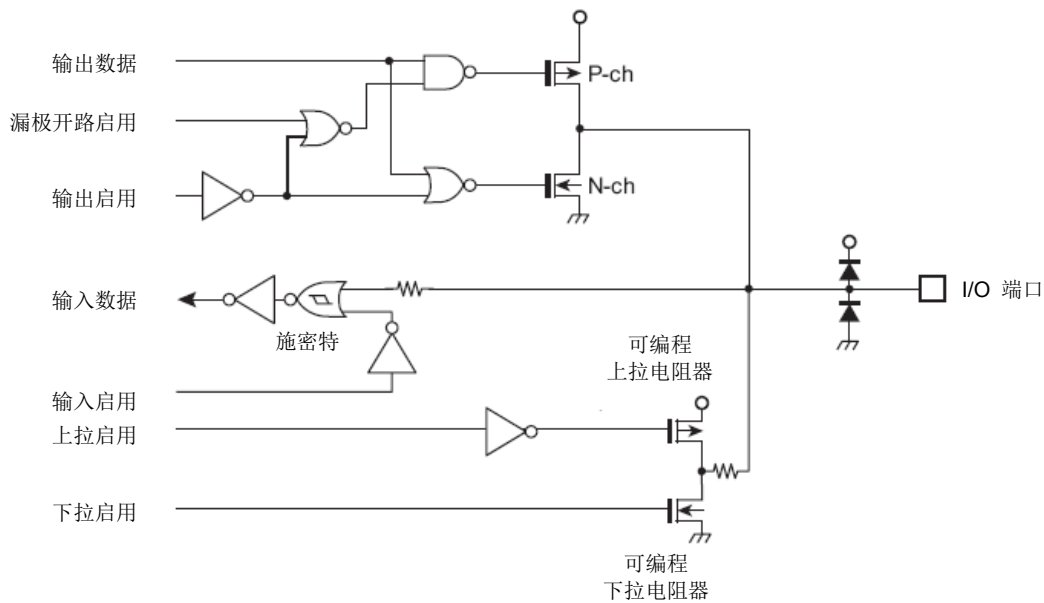
18. 端口部分等效电路示意图

基本上，所写入的逻辑门符号和用于标准CMOS逻辑IC [74HCXX] 系列的相同。
 输入保护电阻范围为自几十Ω至几百Ω。反馈电阻器和阻尼电阻器以典型值表示。

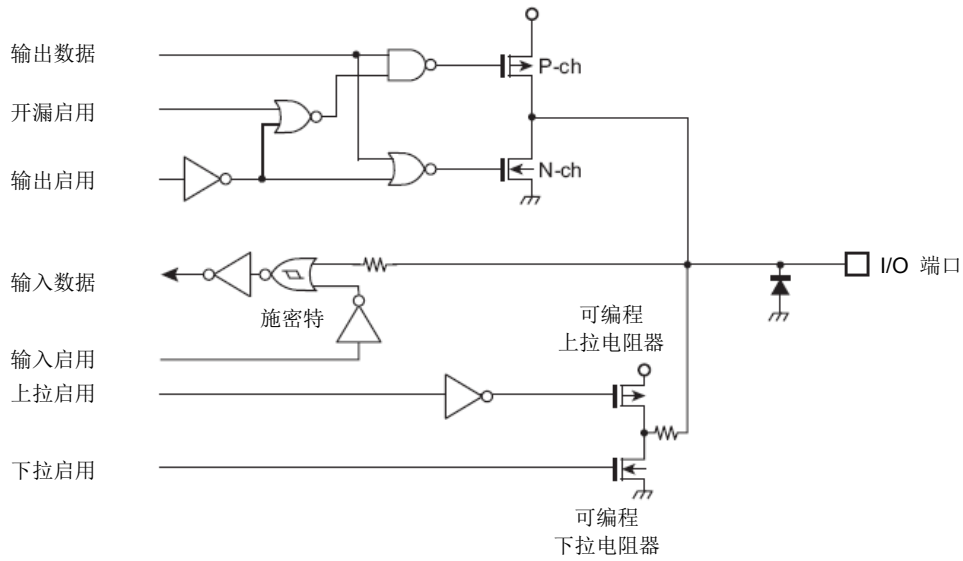
18.1 PA0 ~ 7, PB0 ~ 7



18.2 PC0 ~ 2, PD0 ~ 7, PE0 ~ 7, PF1 ~ 7, PG0 ~ 4, PH0 ~ 4, PI0 ~ 7

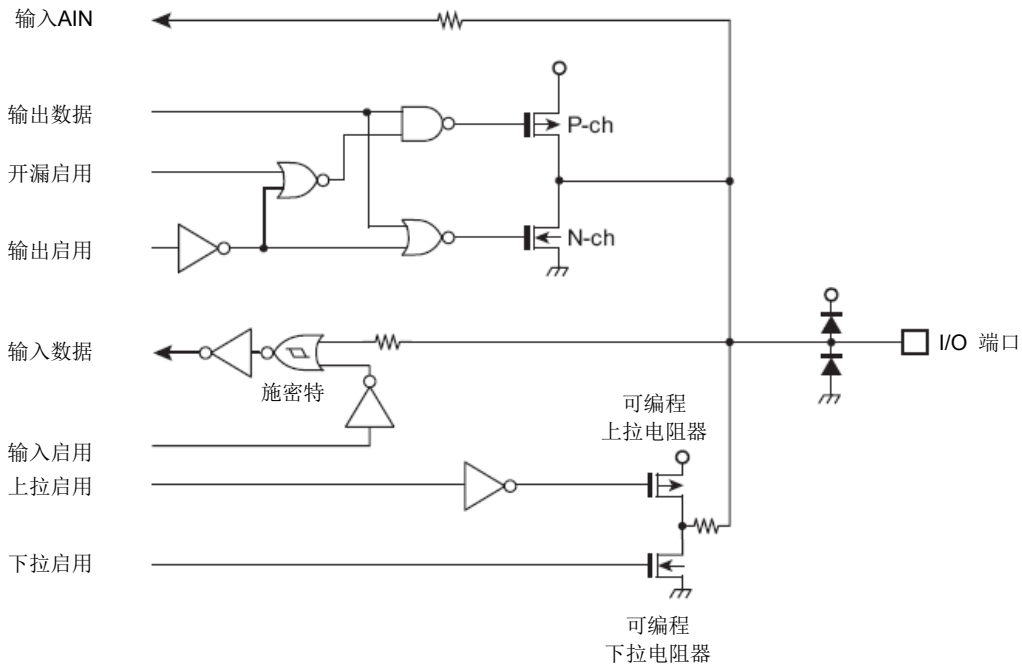


18.3 PG5

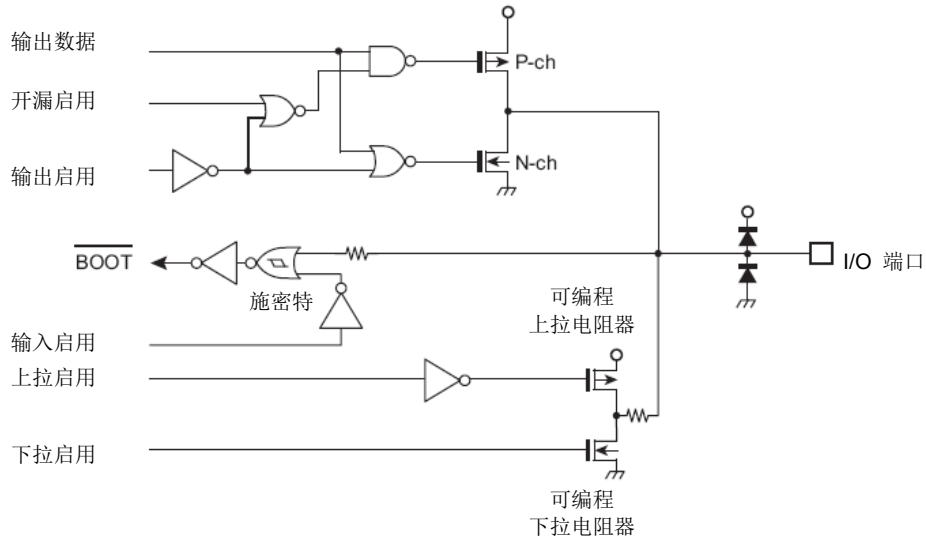


注：只有在输入启用时，这些引脚可承受 5V 输入

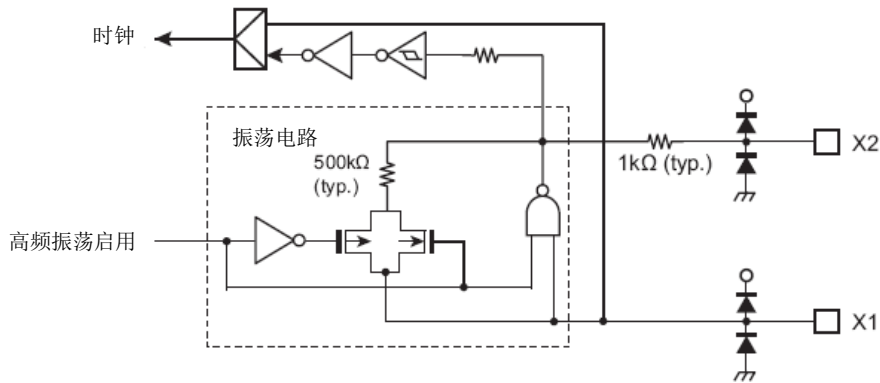
18.4 PJ0 ~ 7, PK0 ~ 3



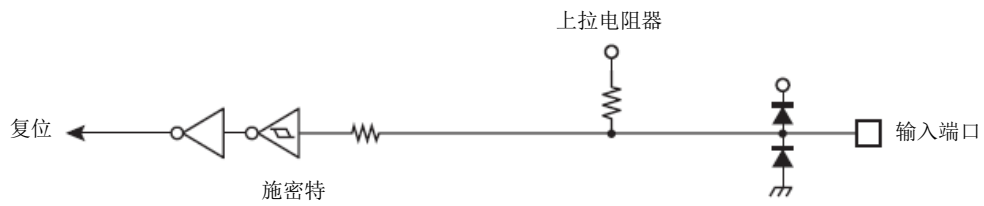
18.5 PF0



18.6 X1, X2



18.7 $\overline{\text{RESET}}$, $\overline{\text{NMI}}$

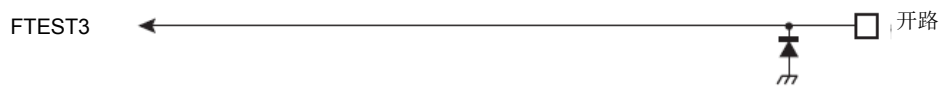


18.8 MODE



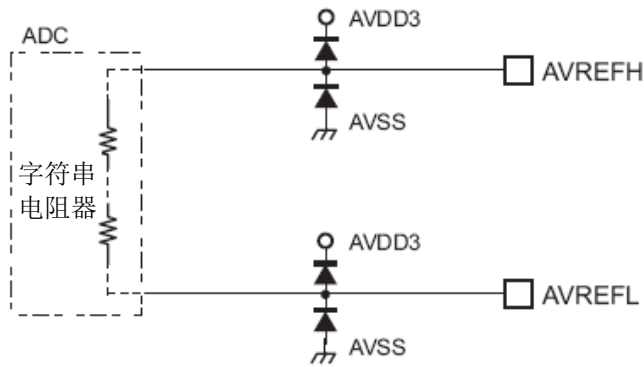
(注) MODE引脚固定于 GND。

18.9 FTEST3



(注) FTEST3 引脚固定于开路。

18.10 AVREFH, AVREFL



19. 电气特性

19.1 绝对最大额定值

参数		符号	额定值	单位
电源电压		DVDD3A	-0.3 ~ 3.9	V
		DVDD3C	-0.3 ~ 3.9	
		AVDD3	-0.3 ~ 3.9	
		RVDD3	-0.3 ~ 3.9	
输入电压	以下端口除外	V_{IN1}	-0.3 ~ VDD + 0.3	V
输入电压	PG5 (5 V 容差输入)	V_{IN2}	-0.3 ~ 5.5	V
低电平输出电流	每引脚	I_{OL}	5	mA
	合计	ΣI_{OL}	50	
高电平输出电流	每引脚	I_{OH}	-5	
	合计	ΣI_{OH}	-50	
功耗 (Ta = 85 °C)		PD	600	mW
焊接温度(10 s)		T_{SOLDER}	260	°C
存储温度		T_{STG}	-40 ~ 125	°C
工作温度	闪存W/E时除外	T_{OPR}	-40 ~ 85	°C
	闪存 W/E时		0 ~ 70	

注：绝对最大额定值为最恶劣可能条件下不可超过的运行和环境条件极限值。装备制造商的设计，应在电流，电压，耗电，温度等方面不超过绝对最大额定值。暴露于以上所列情况下，可能导致装置受到永久性损坏或影响装置可靠性，可能增加由于IC爆炸和/或燃烧导致的人身伤害潜在风险。

19.2 DC电气特性 (1/3)

DVSSA = DVSSB = AVSS = RVSS = DVSSC = 0V

Ta = -40 to 85 °C

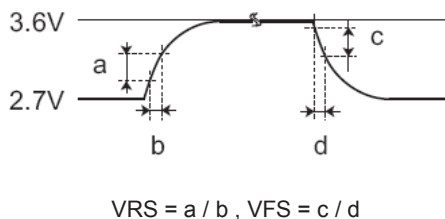
参数		符号	条件	最小值	典型值 (注1)	最大值	单位	
供应电压	DVDD3A DVDD3C AVDD3 RVDD3 (注 2)	DVDD3A AVDD3 RVDD3 DVDD3C	f _{osc} = 8 ~ 12 MHz f _{sys} = 1 ~ 48 MHz	无 USB	2.7	-	3.6	V
				有 USB	3.0	-	3.45	V
低电平输入电压	PA, PB, PC, PD, PE, PF, PG, PH, PI, PJ, PK, $\overline{\text{RESET}}$, $\overline{\text{NMI}}$, MODE, BSC	V _{IL1}	2.7 V ≤ DVDD3A ≤ 3.6 V	-0.3	-	0.2 DVDD3A	V	
	X1	V _{IL2}	2.7 V ≤ RVDD3 ≤ 3.6 V			0.2 RVDD3		
高电平输入电压	PA, PB, PC, PD, PE, PF, PG (PG5除外), PH, PI, PJ, PK, $\overline{\text{RESET}}$, $\overline{\text{NMI}}$, MODE, BSC	V _{IH1}	2.7 V ≤ DVDD3A ≤ 3.6 V	0.8 DVDD3 A	-	DVDD3A +0.3	V	
	PG5	V _{IH3}				5.5		
	X1	V _{IH2}	2.7 V ≤ RVDD3 ≤ 3.6 V	0.8 RVDD3	RVDD3 +0.3			
低电平输出电压		V _{OL}	I _{OL} = 2 mA DVDD3A ≥ 2.7 V	-	-	0.4	V	
高电平输出电压		V _{OH}	I _{OH} = -2 mA DVDD3A ≥ 2.7 V	2.4	-	DVDD3A	V	
输入泄漏电流		I _{LI1}	0.0 ≤ V _{IN} ≤ DVDD3A 0.0 ≤ V _{IN} ≤ AVDD3	-	0.02	±5	μA	
输出泄漏电流		I _{LO}	0.2 ≤ V _{IN} ≤ DVDD3A - 0.2 0.2 ≤ V _{IN} ≤ AVDD3 - 0.2	-	0.05	±10		
复位上拉电阻器		RRST	2.7 V ≤ DVDD3A ≤ 3.6 V	-	50	150	kΩ	
滞后电压		V _{TH}	2.7 V ≤ DVDD3A ≤ 3.6 V	0.3	0.6	-	V	
可编程上拉/下拉电阻器		PKH	2.7 V ≤ DVDD3A ≤ 3.6 V	-	50	150	kΩ	
引脚电容 (电源引脚除外)		C _{IO}	f _c = 1 MHz	-	-	10	pF	
运行范围内的电源变化速率		VRS	DVDD3A = DVDD3C = AVDD3 = RVDD3	-	-	10.0	mV/μs	
		VFS		-	-	-3.33		

注 1: Ta = 25 °C, DVDD3A = DVDD3C = RVDD3 = AVDD3 = RVDD3 = 3.3 V, 除非另外注明。

注 2: 须向DVDD3A, DVDD3C, AVDD3和RVDD3提供相同电压。

注 3: 确保在DVDD3A, DVDD3C, AVDD3和RVDD3低于最小工作电压 2.7 V时(使用USB时为 3.0 V), 所有电源, 包括AVDD3, 断电后再通电。

注 4: 作为特性, VRS(上升)和VFS(下降)应以严格标准进行测量。



19.3 DC电气特性 (2/3)

AVDD3 = RVDD3 = 2.7 V ~ 3.6 V (注 2)

Ta = -40 to 85 °C

参数	符号	条件	最小值	典型值	最大值	单位
低电平 输出电流	I_{OL1}	每引脚 2.7 V ≤ DVDD3A ≤ 3.6 V <PA0 ~ PA7, PB0 ~ PB7, PC0 ~ PC7, PD0 ~ PD7, PE0 ~ PE7, PF0 ~ PF7, PG0 ~ PG5, PH0 ~ PH4, PI0 ~ PI7, PJ0 ~ PJ7, PK0 ~ PK7>	-	-	2	mA
	ΣI_{OL1}	每组, 端口 A	-	-	10	
	ΣI_{OL2}	每组, 端口 B	-	-	10	
	ΣI_{OL3}	每组, 端口 C	-	-	10	
	ΣI_{OL4}	每组, 端口 D	-	-	10	
	ΣI_{OL5}	每组, 端口 E	-	-	10	
	ΣI_{OL6}	每组, 端口 F	-	-	10	
	ΣI_{OL7}	每组, 端口 G	-	-	20	
	ΣI_{OL8}	每组, 端口 H	-	-	10	
	ΣI_{OL9}	每组, 端口 I	-	-	10	
	ΣI_{OL10}	每组, 端口 J	-	-	10	
	ΣI_{OL11}	每组, 端口 K	-	-	10	
ΣI_{OL}	合计, 所有端口	-	-	35		
高电平输出电流	I_{OH1}	每引脚 2.7 V ≤ DVDD3A, DVDD3C ≤ 3.6 V <PA0 ~ PA7, PB0 ~ PB7, PC0 ~ PC7, PD0 ~ PD7, PE0 ~ PE7, PF0 ~ PF7, PG0 ~ PG5, PH0 ~ PH4, PI0 ~ PI7, PJ0 ~ PJ7, PK0 ~ PK7>	-	-	-2	mA
	ΣI_{OH1}	每组, 端口 A	-	-	-10	
	ΣI_{OH2}	每组, 端口 B	-	-	-10	
	ΣI_{OH3}	每组, 端口 C	-	-	-10	
	ΣI_{OH4}	每组, 端口 D	-	-	-10	
	ΣI_{OH5}	每组, 端口 E	-	-	-10	
	ΣI_{OH6}	每组, 端口 F	-	-	-10	
	ΣI_{OH7}	每组, 端口 G	-	-	-10	
	ΣI_{OH8}	每组, 端口 H	-	-	-10	
	ΣI_{OH9}	每组, 端口 I	-	-	-10	
	ΣI_{OH10}	每组, 端口 J	-	-	-10	
	ΣI_{OH11}	每组, 端口 K	-	-	-10	
ΣI_{OH}	合计, 所有端口	-	-	-35		

注 1: 须向DVDD3A, DVDD3C, AVDD3和RVDD3提供相同电压。

注 2: 3.0 V ~ 3.45 V (使用USB时)

19.4 DC电气特性 (3/3)

DVDD3A = DVDD3C = AVDD3 = RVDD3 = 2.7 V ~ 3.6 V (注 6)

Ta = -40 ~ 85 °C

参数	符号	条件	最小值	典型值(注1)	最大值	单位
NORMAL (注 2) (注 3)	I _{DD}	齿轮1/1 fsys = 48 MHz TMRB, ADC, SIO/UART, I2C/SIO 和USB 运行。	-	31.8	42.0	mA
NORMAL (注 2) (注 3)		齿轮1/1 fsys = 48 MHz TMRB, ADC, SIO/UART 和I2C/SIO 运行。	-	22.3	32.0	
IDLE (注4)		齿轮1/1 fsys = 48 MHz 禁用所有外围设备。	-	10.0	16.0	
STOP1 (注5)		-	-	14.1	800	μA

注 1: Ta = 25 °C, DVDD3A = DVDD3C = AVDD3 = RVDD3 = 3.3 V, 除非另外注明。

注 2: I_{DD} NORMAL: 用测试程序版本测量。2.1 运行于FLASH。

注 3: I_{DD} NORMAL: 通过DVDD3A, DVDD3C 和AVDD3的电流不包括在内。

注 4: I_{DD} IDLE: 测量时所有功能停止。通过DVDD3A, DVDD3C, AVDD3和RVDD3的电流包括在内。

注 5: I_{DD} STOP1:通过DVDD3A, DVDD3C, AVDD3和RVDD3的电流包括在内。

注 6: 3.0 V ~ 3.45 V (使用USB时)

19.5 12-位ADC电气特性

DVDD3A = DVDD3C = AVDD3 = RVDD3 = AVREFH = 2.7 V ~ 3.6 V

DVSSA = DVSS3C = AVSS = RVSS = DVSSC = 0 V

Ta = -40 to 85 °C

参数	符号	条件	最小值	典型值	最大值	单位
模拟参考电压(+)	AVREFH	-	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	-	-	1.5	2.0	mA
	非AD转换	-	-	0.02	0.1	μA
INL误差	-	AIN 电阻 ≤ 600 Ω AIN 负载电容 ≥ 0.1 μF 转换时间 ≥ 1.0 μs	-	4	8	LSB
DNL误差			-	2	8	
偏移误差			-	3	8	
满刻度误差			-	3	8	
INL误差	-	AIN 电阻 ≤ 600 Ω AIN 负载电容 ≥ 33 pF 转换时间 ≥ 1.66 μs	-	3	8	
DNL误差			-	2	8	
偏移误差			-	4	8	
满刻度误差			-	2	8	
转换时间	Tconv	-	1.0	-	10	μs

注 1: $1\text{LSB} = (\text{AVREFH} - \text{AVSS}) / 4096 [\text{V}]$

注 2: 所有外围功能禁用, ADC除外。

19.6 AC电气特性

19.6.1 AC 测量条件

除非另外注明，本节AC特性数据在以下条件下进行测量

- 输出电平: 高 = $0.8 \times DVDD3A$, $0.8 \times DVDD3C$
- 输出电平: 低 = $0.2 \times DVDD3A$, $0.2 \times DVDD3C$
- 输入电平: 参阅低电平输入电压和高电平输入电压 "DC电气特性"。
- 负载容量: $CL = 30 \text{ pF}$

19.6.2 串行通道 (SIO/UART)

19.6.2.1 I/O接口模式

下表中，字母x代表SIO运行时钟周期时间，和 f_{sys} 周期时间相同。因时钟齿轮功能编程而异。

(1) SCLK输入模式

[数据输入]

参数	符号	等式		$f_{sys} = 40 \text{ MHz}$		48 MHz		单位
		最小值	最大值	最小值	最大值	最小值	最大值	
SCLK 时钟高宽度 (输入)	t_{SCH}	$4x$	-	100	-	83.3	-	ns
SCLK 时钟低宽度 (输入)	t_{SCL}	$4x$	-	100	-	83.3	-	
SCLK 周期	t_{SCY}	$t_{SCH} + t_{SCL}$	-	200	-	166.6	-	
有效数据输入 → SCLK 上升 / 下降 (注1)	t_{SRD}	30	-	30.0	-	30.0	-	
SCLK 上升 / 下降 (注1) → 输入数据保持	t_{HSR}	$x + 30$	-	55.0	-	50.8	-	

[数据输出]

参数	符号	等式		$f_{sys} = 40 \text{ MHz}$		48 MHz		单位
		最小值	最大值	最小值	最大值	最小值	最大值	
SCLK 时钟高宽度 (输入)	t_{SCH}	$4x$	-	120 (注3)	-	107.5 (iç3)	-	ns
SCLK 时钟低宽度 (输入)	t_{SCL}	$4x$	-	120 (注3)	-	107.5 (iç3)	-	
SCLK 周期	t_{SCY}	$t_{SCH} + t_{SCL}$	-	240	-	215	-	
输出数据 → SCLK 上升 / 下降 (注1)	t_{OSS}	$t_{SCY}/2 - 3x - 45$	-	0.00 (注2)	-	0.00 (iç2)	-	
SCLK 上升 / 下降 (注1) → 输出数据 保持	t_{OHS}	$t_{SCY}/2$	-	120	-	107.5	-	

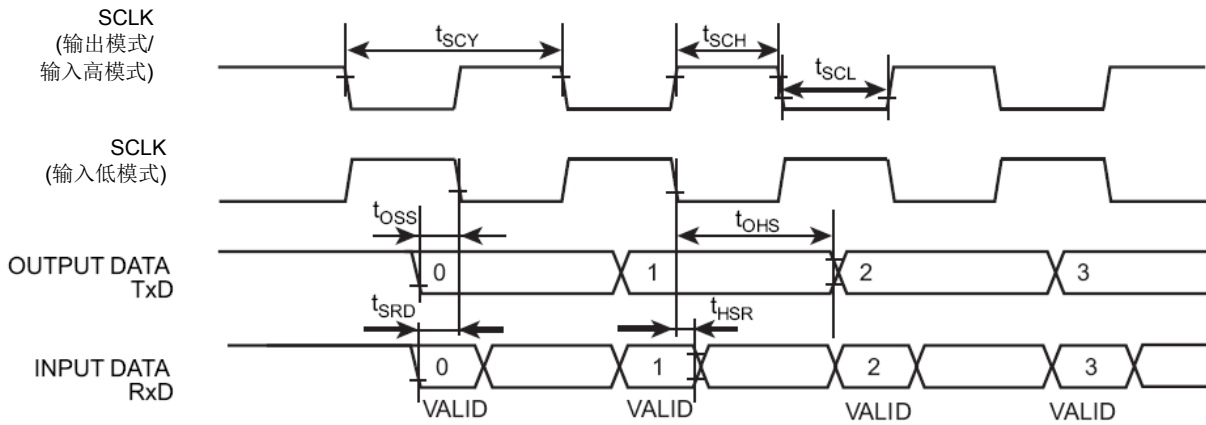
注 1: SCLK 上升/下降: SCLK 上升模式使用SCLK的上升定时。SCLK下降模式使用SCLK的下降定时。

注 2: 使用在计算值保持为正的范围内SCLK的频率。

注 3: 该数值显示的是使 t_{OSS} 为零或更大的最小值。

(2) SCLK输出模式

参数	符号	等式		40 MHz		48 MHz		单位
		最小值	最大值	最小值	最大值	最小值	最大值	
SCLK 周期 (可编程)	t_{SCY}	4x	-	100	-	83.3	-	ns
输出数据 ← SCLK 上升	t_{OSS}	$t_{SCY}/2 - 30$	-	20	-	11.7	-	
SCLK上升→ 输出保持 数据保持	t_{OHS}	$t_{SCY}/2 - 30$	-	20	-	11.7	-	
有效数据输入 ← SCLK上升	t_{SRD}	45	-	45	-	45	-	
SCLK上升→ 输入数据 保持	t_{HSR}	0	-	0	-	0	-	



19.6.3 串行总线接口(I2C/SIO)

19.6.3.1 I2C模式

下表中，字母x代表I2C运行时钟周期时间，和fsys周期时间相同。因时钟齿轮功能编程而异。
n 表示编程入SBIxCR 内SCK字段的n数值（SCL输出频率选择）。

参数	符号	等式		标准模式		快速模式		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
重复START 条件的设置时间	t _{SU; STA}	(注 5)	-	4.7	-	0.6	-	μs
数据保持时间(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 条件的设置时间	t _{SU; STO}	-	-	4.0	-	0.6	-	μs
停止条件和起始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) = $(2^{n-1} + 58)/x$

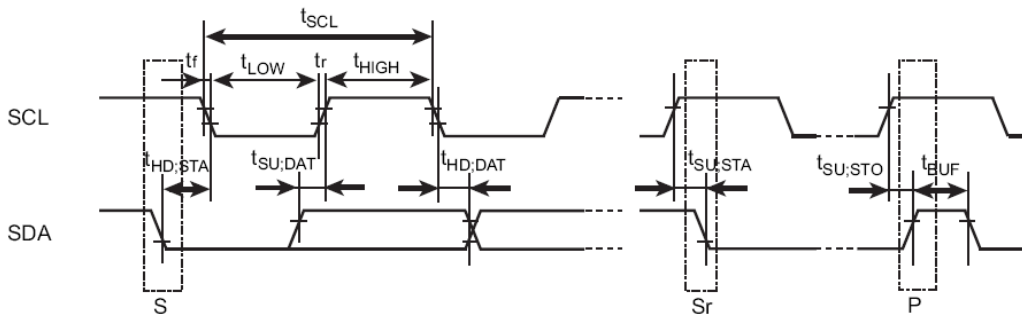
注 2: SCL时钟高宽度 (输出) = $(2^{n-1} + 14)/x$ 关于I2C-总线规范，标准模式的最高速度为 100 kHz，快速模式为 400 kHz。
内部SCL频率设置应和上述所示注 1 和注 2 相符。

注 3: 输出数据保持时间等于内部SCL的 4 倍(4x)。

注 4: 飞利浦I2C-总线规范表明，装置内部须为跨越SCL下降缘未定义区域的SDA信号提供至少 300 ns的保持时间。然而，该SBI并不符合该要求。另外，SCL的输出缓冲不包括下降缘的斜度控制；因此，装备制造商的设计应符合表中所示输入数据保持时间，包括SCL和SDA线路的tr/tf。

注 5: 取决于软件

注 6: 飞利浦I2C-总线规范指示，若快速模式装置电源关闭，SDA 和SCL I/O引脚须为浮动，以免阻塞总线线路。然而，该SBI并不符合该要求。



S: 起始条件
Sr: 重复起始条件
P: 停止条件

19.6.3.2 时钟同步 8-位SIO模式

下表中，字母x代表I2C运行时钟周期时间，和f_{sys}周期时间相同。因时钟齿轮功能编程而异。

(1) SCK输入模式(以下电气特性适用于 50%占空周期的SCK信号。)

[数据输入]

参数	符号	等式		f _{sys} = 40 MHz		48 MHz		单位
		最小值	最大值	最小值	最大值	最小值	最大值	
SCK时钟高宽度 (输入)	t _{SCH}	4x	-	100	-	83.3	-	ns
SCK时钟低宽度 (输入)	t _{SCL}	4x	-	100	-	83.3	-	
SCK 周期	t _{SCY}	t _{SCH} + t _{SCL}	-	200	-	166	-	
有效数据输入 → SCK 上升	t _{SRD}	30 - x	-	5	-	9	-	
SCK 上升 → 输入数据 保持	t _{HSR}	2x + 30	-	80	-	71.7	-	

[数据输出]

参数	符号	等式		f _{sys} = 40 MHz		48 MHz		单位
		最小值	最大值	最小值	最大值	最小值	最大值	
SCK时钟高宽度 (输入)	t _{SCH}	4x	-	120 (注2)	-	108 (注2)	-	ns
SCK时钟低宽度 (输入)	t _{SCL}	4x	-	120 (注2)	-	108 (注2)	-	
SCK 周期	t _{SCY}	t _{SCH} + t _{SCL}	-	240	-	215	-	
输出数据 → SCLK 上升	t _{OSS}	t _{SCY} /2 - 3x - 45	-	0 (注1)	-	0 (注1)	-	
SCK 上升 → 输出数据 保持	t _{OHS}	t _{SCY} /2 + x	-	145	-	128	-	

注 1: 使用在计算值保持为正的范围内SCK的频率。

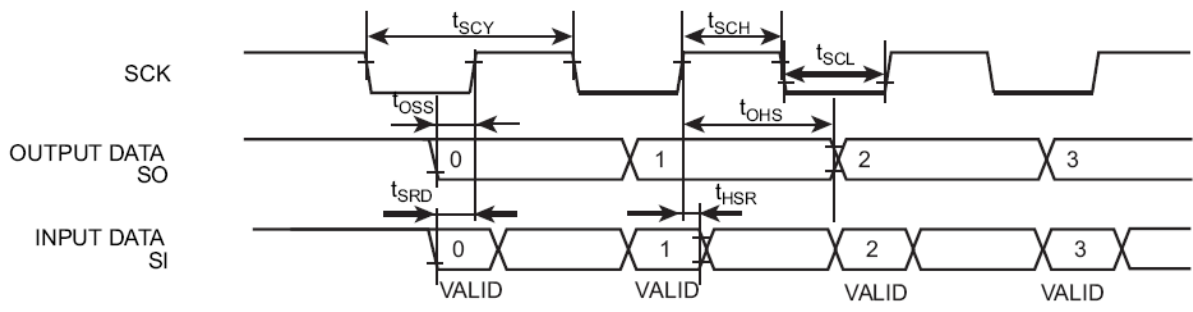
注 2: 该数值显示的是使t_{OSS}为零或更大的最小值。

(2) SCK输出模式(以下电气特性适用于 50%占空周期的SCK信号。)

参数	符号	等式		40 MHz		48 MHz		单位
		最小值	最大值	最小值	最大值	最小值	最大值	
SCK 周期 (可编程)	t _{SCY}	16x	-	400	-	333	-	ns
输出数据 → SCK 上升	t _{OSS}	t _{SCY} /2 - 20	-	180	-	147	-	
SCK 上升 → 输出数据 保持	t _{OHS}	t _{SCY} /2 - 20	-	180	-	147	-	
有效数据输入 → SCK 上升	t _{SRD}	x + 45	-	70	-	65.8	-	
SCK 上升 → 输入数据 保持	t _{HSR}	0	-	0	-	0	-	

注 1: 自动等待后SCK周期变为 14x。

注 2: 自动等待后t_{OSS}可能为t_{SCY}/2-x-20。



19.6.4 事件计数器

下表中，字母x代表TMRB运行时钟周期时间，和f_{sys}周期时间相同。因时钟齿轮功能编程而异。

参数	符号	等式		40 MHz		48 MHz		单位
		最小值	最大值	最小值	最大值	最小值	最大值	
时钟低脉冲宽度	t _{VCKL}	2x + 100	-	150	-	142	-	ns
时钟高脉冲宽度	t _{VCKH}	2x + 100	-	150	-	142	-	ns

19.6.5 捕捉

下表中，字母x代表TMRB运行时钟周期时间，和f_{sys}周期时间相同。因时钟齿轮功能编程而异。

参数	符号	等式		40 MHz		48 MHz		单位
		最小值	最大值	最小值	最大值	最小值	最大值	
低脉冲宽度	t _{CPL}	2x + 100	-	150	-	142	-	ns
高脉冲宽度	t _{CPH}	2x + 100	-	150	-	142	-	ns

19.6.6 外部中断

下表中，字母x代表f_{sys}周期时间。

1. STOP1释放中断除外

参数	符号	等式		40 MHz		48 MHz		单位
		最小值	最大值	最小值	最大值	最小值	最大值	
INT0~9低电平脉冲宽度	t _{INTAL}	x + 100		125	-	121	-	ns
INT0~9高电平脉冲宽度	t _{INTAH}	x + 100	-	125	-	121	-	ns

2. STOP1释放中断

参数	符号	最小值	最大值	单位
INT0~9低电平脉冲宽度	t _{INTBL}	100	-	ns
INT0~9高电平脉冲宽度	t _{INTBH}	100	-	ns

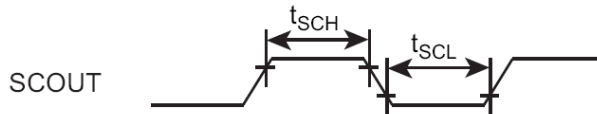
19.6.7 NMI

参数	符号	最小值	最大值	单位
NMI 低电平脉冲宽度	t_{INTCL}	100	-	ns

19.6.8 SCOUT引脚AC特性

参数	符号	等式		40 MHz		48 MHz		单位
		最小值	最大值	最小值	最大值	最小值	最大值	
高脉冲宽度	t_{SCH}	$0.5T - 5$	-	7.5	-	5.4	-	ns
低脉冲宽度	t_{SCL}	$0.5T - 5$	-	7.5	-	5.4	-	ns

注：上表中，字母T代表SCOUT输出时钟的周期时间。



19.6.9 ADTRG 触发器输入引脚AC特性

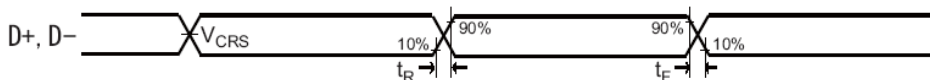
下表中，字母x代表 f_{sys} 周期时间。因时钟齿轮功能编程而异。

参数	符号	等式		40MHz		48MHz		单位
		最小值	最大值	最小值	最大值	最小值	最大值	
低电平脉冲宽度	T_{adl}	$2x + 20$	-	32.5	-	32.5	-	ns
高电平脉冲间隔	T_{adh}	$2x + 20$	-	32.5	-	32.5	-	

19.6.10 USB 定时

DVDD3A = DVDD3C = AVDD3 = RVDD3 = 3.0 ~ 3.45V, $f_{sys} = 48\text{ MHz}$

参数	符号	最小值	最大值	单位
D+, D-电源上升时间	t_R	4	20	ns
D+, D-电源延迟时间	t_F	4	20	
数据线交变电压	V_{CRS}	1.3	2.0	V



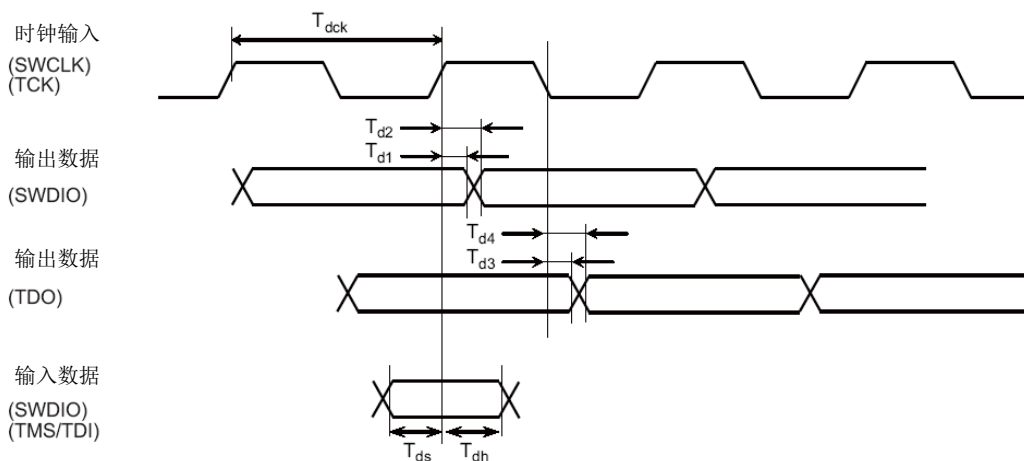
19.6.11 调试通信

19.6.11.1 SWD 接口

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 保持	T_{d1}	4	-	
CLK 下降 → 输出数据 保持	T_{d2}	-	30	
输入数据 有效 ← CLK 上升	T_{ds}	20	-	
CLK 上升 → 输入数据 保持	T_{dh}	15	-	

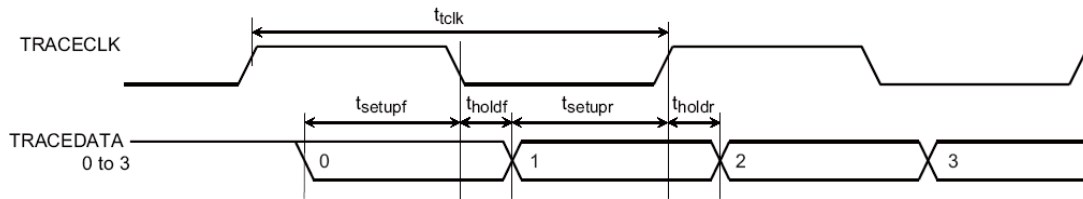
19.6.11.2 JTAG 接口

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 保持	T_{d3}	4	-	
CLK 下降 → 输出数据 保持	T_{d4}	-	50	
输入数据 有效 ← CLK 上升	T_{ds}	20	-	
CLK 上升 → 输入数据 保持	T_{dh}	15	-	



19.6.12 ETM追踪

参数	符号	最小值	最大值	单位
TRACECLK 周期	t_{clk}	50	-	ns
TRACEDATA 有效 ← TRACECLK 上升	t_{setupr}	2	-	ns
TRACECLK 上升 → TRACEDATA 保持	t_{holdr}	1	-	ns
TRACEDATA 有效 ← TRACECLK 下降	t_{setupf}	2	-	ns
TRACECLK 下降 → TRACEDATA 保持	t_{holdf}	1	-	ns



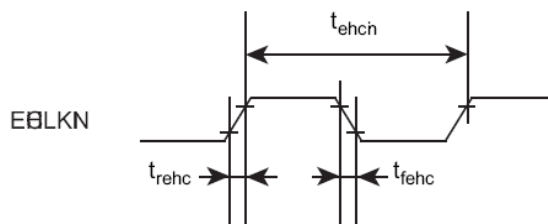
19.6.13 片上振荡器

参数	符号	条件	最小值	典型值	最大值	单位
振荡频率	IHOSC	$T_a = -40 \sim 85^\circ\text{C}$	9	10	11	MHz

注：片上振荡器不能用于需振荡准确度的系统时钟(f_{sys})。

19.6.14 外部时钟输入

参数	符号	最小值	典型值	最大值	单位
外部时钟频率	t_{ehcin}	8	-	48	MHz
外部时钟占空比	-	45	-	55	%
外部时钟输入 上升时间	t_{rehc}	-	-	10	ns
外部时钟输入下降时间	t_{feh}	-	-	10	ns



19.6.15 闪存特性

参数	条件	最小值	典型值	最大值	单位
闪存重写保证	DVDD3A = AVDD3 = RVDD3 = 2.7 V ~ 3.6 V, DVDD3C = 2.7 V ~ 3.6 V, Ta = 0 ~ 70 °C	-	-	100	次数

19.7 推荐振荡电路

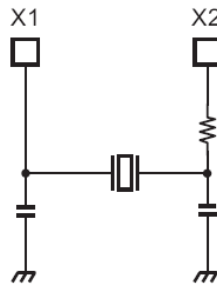


图 19-1 高频振荡连接

注：为获得稳定振荡，负载容量和振荡器位置须正确配置。由于这些因素受基板模式影响很大，请采用你使用的基板评估振荡稳定性。

TX03已经以下振荡器供应商评估。选择外部部件时，请参阅该信息。

19.7.1 陶瓷谐振器

TX03推荐村田制作所生产的高频振荡器。详情请参阅以下网址。

<http://www.murata.co.jp>

19.7.2 晶体振荡器

TX03推荐京瓷晶体元件株式会社生产的高频振荡器。详情请参阅以下网址。

<http://www.kinseki.co.jp>

19.7.2.1 印刷电路板设计注意事项

确保印刷电路板模式设计为将晶体元件和其它振荡元件连接，那样此类模式长度会尽可能变得最短，以免由于杂散电容和配线电感的原因导致特性恶化。对于多层电路板而言，重要的是切勿在振荡电路正下方接地线和其它信号模式。

更多详情，请参阅振荡器供应商网址。

19.8 使用注意事项

19.8.1 通电电源电压电平

通电时，电源上升须小于下表中数值。

TMPM365FYXBG配有若干电源引脚。它们须同时供电。且外部复位引脚（ $\overline{\text{RESET}}$ ）通电时须为"低"电平。

电源引脚 = DVDD3A, DVDD3C, AVDD3, RVDD3

C = 0.1 μF

Ta = -40 ~ 85°C

参数	条件	最小值	典型值	最大值	单位
通电时的电源上升	0V → 2.7V ~ 3.6V	-	-	10	mV/ μs

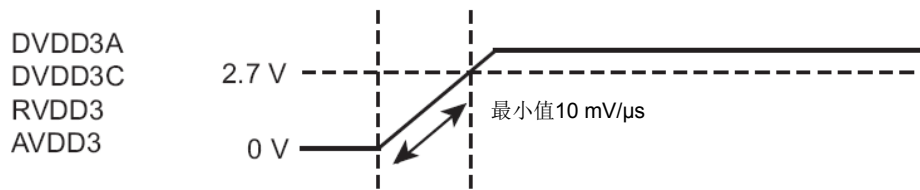
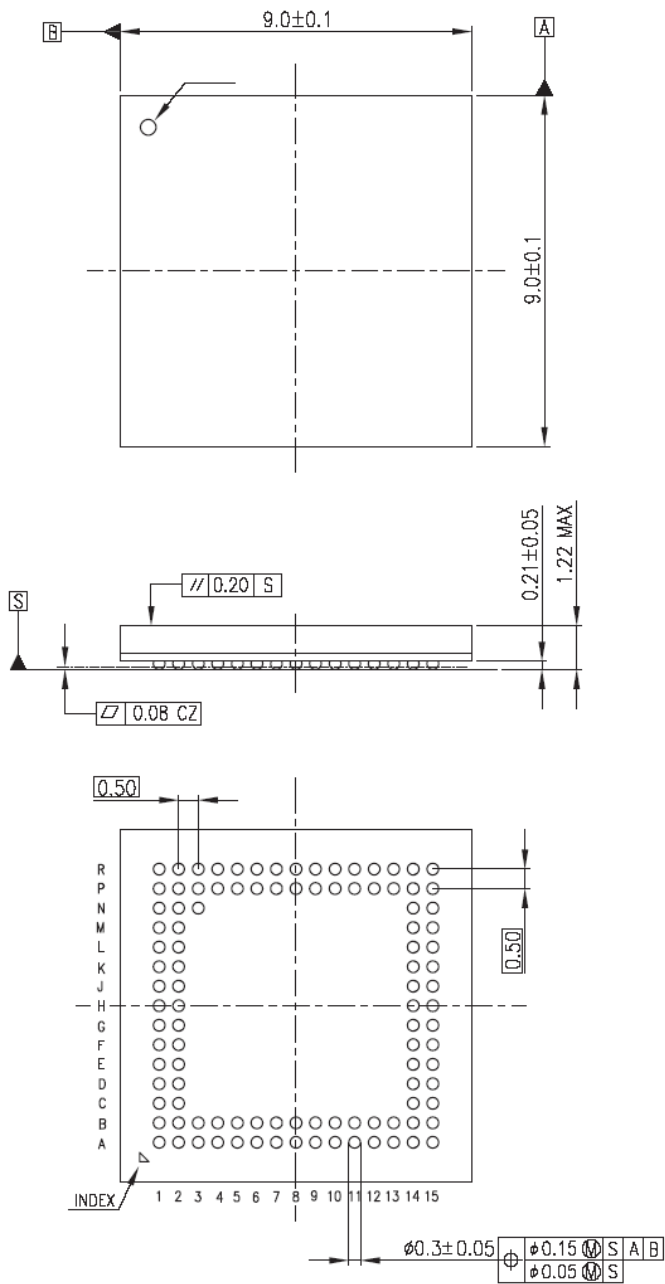


图 19-2 通电电源电压电平

20. 封装尺寸

类型: P-LFBGA105-0909-050-001

"单位: mm"





译文

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.**

译文