免责声明

本 AT91SAM9261 中文数据手册为 Mcuzone 在 ATMEL 的 AT91SAM9261 英文数据手册(内部版本 6062G-ATARM-13-Mar-07)基础上历时一年半翻译而成。首先感谢您下载并阅读本文档，在阅读以下正文内容时我们即认为您已经阅读并同意以下声明内容：

Mcuzone 在不涉及版权的前提下免费公布本文档，故；

一、对于因使用、复制、修改、散布此文档等行为而导致触犯相关法律者，其应自行承担全部责任；

二、Mcuzone 并不保证翻译内容的准确性、可靠性或完整性，对于因使用本文档而导致的损失由使用者自行承担；

三、Mcuzone 没有义务修正错误、更新内容、或通知用户；使用者应以最新英文原稿为参考以获得更准确的信息。

AT91SAM9261 中文数据手册

Rev 1.09.02

Based on 6062G–ATARM–13-Mar-07

翻译：Quillan (QQ: 289011331)
校对：微控电子(http://www.mcuzone.com)
发布：http://www.mcuzone.com http://www.ATARM.com

特点

- 融合了 ARM926EJ-S™ ARM® Thumb® 处理器
  - 扩展 DSP 指令
  - ARM Jazelle 技术提供了 Java 加速功能
  - 16K 字节数据缓存，16K 字节指令缓存，写缓冲器
  - 工作于 190 MHz 时性能高达 210 MIPS
  - 存储器管理单元
  - 嵌入式 ICE ，支持调试信道
  - 中等规模的嵌入式宏单元结构
- 附加的嵌入式存储器
  - 32K 字节片内 ROM,最大总线速率下单周期访问
  - 160K 字节片内 SRAM，最大处理器或总线速率下单周期访问
- 外部总线接口（EBI）
  - 支持 SDRAM，静态存储器，NAND Flash 和 CompactFlash
- LCD 控制器
  - 支持被动或主动显示
  - 在 STN 彩色模式下达 16 位深每像素
  - 在 TFT 模式下达 16 M 色（24 位深每像素），分辨率高达 2048*2048
- USB
  - USB 2.0 全速（12M 位每秒）主机双端口
    双重片上收发器
    集成 FIFOs 和专用 DMA 通道
  - USB 2.0 全速（12M 位每秒）设备端口
    片上收发器，2K 字节可配置的集成 FIFOs
- 总线矩阵
  - 管理五个主控和五个从控
  - 启动模式选择选项
  - Remap 命令
- 全特征系统控制器(SYSC)提供了有效系统管理，包括
  - 复位控制器，掉电控制器，支持总共 16 字节的四个 32-bit 电池备份寄存器
  - 时钟发生器和功率管理控制器
  - 先进的中断控制器和调试部件
  - 周期间隔定时器，看门狗定时器和实时定时器
  - 三个 32 位 PIO 控制器
- 复位控制器(RSTC)
  - 基于上电复位的单元，复位源辨认和复位输出控制
- 掉电控制器(SHDWC)
  - 可编程掉电引脚控制和唤醒电路
- 时钟发生器(CKGR)
  - 电池备份电源上的 32.768KHz 低功率振荡器，提供一个永久的慢速时钟
  - 3 到 20MHz 的片上振荡器和两个 PLL
- 功率管理控制器(PMC)

  - 超慢速时钟操作模式，软件可编程功率优化能力
  - 四个可编程外部时钟信号
- 先进的终端控制器(AIC)
  - 可单独屏蔽的，8 级优先级，向量中断源
  - 三个外部中断源和一个快速中断源，伪中断保护
- 调试部件(DBGU)
  - 2 线 USART 兼容接口，可通过编程禁止通过 ICE 访问
- 周期间隔定时器(PIT)
  - 20 位间隔定时器加 12 位间隔计数器
- 看门狗定时器(WDT)
  - 受预设值保护的、一次性可编程的、运行在慢速时钟的 12 位窗口计数器
- 实时定时器(RTT)
  - 运行于慢速时钟的 32 位自由运行的(备份)计数器
- 三个 32 位并行输入/输出控制器(PIO)PIOA,PIOB 和 PIOC
  - 96 可编程 I/O 口线多路复用支持达两个外设 I/O 口
  - 在每个 I/O 口线上具有输入改变中断能力
  - 单独得可编程开漏，上拉电阻和同步输出
- 19 个外设 DMA 通道(PDC)
- 多媒体卡接口(MCI)
  - 支持 SD 卡和 MultiMediaCard（MMC 卡）
  - 自动协议控制,通过 PDC 与 MMC 和 SD 卡进行快速自动数据传输
- 三个同步串行控制器(SSC)
  - 每个接收器和发送器都具有独立的时钟和帧同步信号
  - 支持 IIS 模拟接口，支持时分多路复用
  - 支持 32 位数据传输的高速连续数据流功能
- 三个通用同步/异步收发器(USART)
  - 独立的波特率发生器，IrDA 红外调制/解调
  - 支持 ISO7816 T0/T1 智能卡，硬件和软件握手信号，支持 RS485
- 两个 主/从串行外设接口(SPI)
  - 8 到 16 位可编程数据长度，四个外部外设片选
- 一个三通道 16 位定时器/计数器(TC)
  - 三个外部时钟输入，每个通道有两个多用途 I/O 引脚
  - 倍速 PWM 发生功能，捕捉波形模式，递增/递减计数功能
- 一个两线接口(TWI)
  - 支持主控模式，支持所有两线 Atmel EEPROM
- IEEE 1149.1 JTAG 边界扫描,可以支持所有数字引脚
- 电源
  - 为 VDDCORE 和 VDDBU 提供 1.08V 到 1.32V 电压
  - 为 VDDOSC 和 VDDPLL 提供 3.0V 到 3.6V 电压
  - 为 VDDIOP（外设 I/O 口）提供 2.7V 到 3.6V 电压
  - 为 VDDIOM（存储器 I/O 口）提供 1.65V 到 1.95V 和 3.0V 到 3.6V 电压
- 符合 RoHS 的 217 球的 LFBGA 封装

一．描述

AT91SAM9261 是以 ARM926EJ-S ARM Thumb 处理器为核心的完全的片上系统，它扩展了 DSP 指令集和 Jazelle Java 加速器。主时钟频率 190MHz 时性能高达 210 MIPS

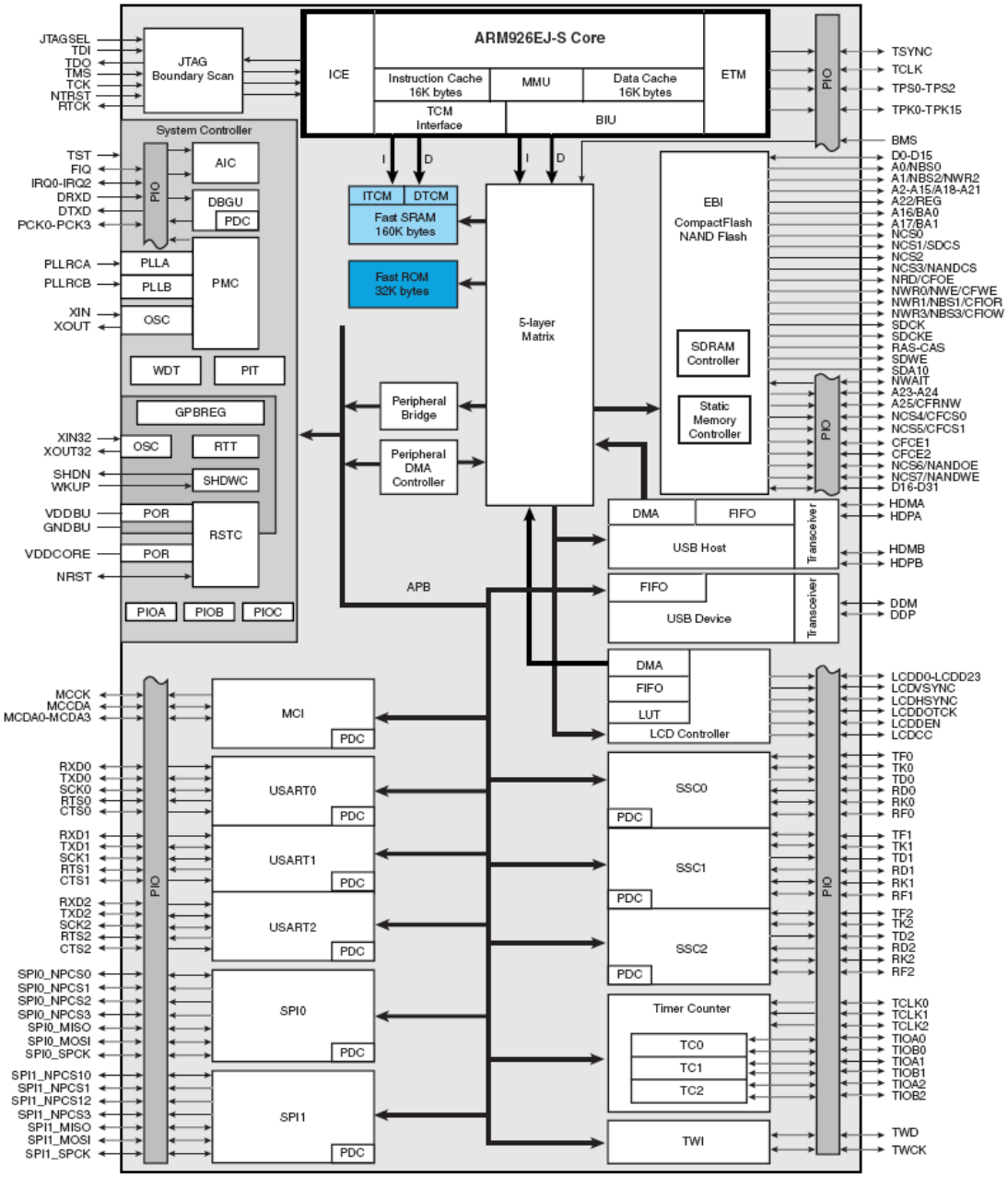
AT91SAM9261 是一个为带 LCD 显示应用而优化了的主机处理器。它的集成 LCD 控制器支持 BW 而且达到 16M 色彩，主动和被动 LCD 显示。160K 字节的片上 SRAM 可配置为帧缓冲，能将 LCD 刷新对整体处理器性能上的影响减到最小。外部总线接口包括支持同步 DRAM（SDRAM）和支持静态存储器的控制器，并有特殊接口电路以支持 CompactFLASH、NAND Flash。

AT91SAM9261 集成了一个支持影射的基于 ROM 的 Boot loader，例如，从外部 DataFlash 影射到外部 SDRAM。由软件控制的功率管理控制器(PMC)通过有选择的启用和关闭处理器、各种外设和工作频率的调解能使系统功率损耗保持最小。

AT91SAM9261 还集成了宽范围的调试特性。包括 JTAG-ICE,一个专用的 UART 调试通道(DBGU)和嵌入式实时追踪。这使所有应用的调试和开发，特别是严格要求实时性的应用得以实现。

二．方框图

图 2-1. AT91SAM9261 方框图



三．信号描述

表 3-1.信号描述列表

信号名称	功能	类型	有效电平	备注
电源				
VDDIOM	EBI I/O 口线电源	电源		1.65V-1.95V,3.0V-3.6V
VDDIOP	外设 I/O 口线电源	电源		2.7V-3.6V
VDDBU	备份 I/O 口线电源	电源		1.08V-1.32V
VDDPLL	PLL 电源	电源		3.0V-3.6V
VDDOSC	振荡器电源	电源		3.0V-3.6V
VDDCORE	内核电源	电源		1.08V-1.32V
GND	地	地		
GNDPLL	PLL 地	地		
GNDOSC	振荡器地	地		
GNDBU	备份电源地	地		
时钟，振荡器和 PLL				
XIN	主振荡器输入	输入		
XOUT	主振荡器输出	输出		
XIN32	慢速时钟振荡器输入	输入		
XOUT32	慢速时钟振荡器输出	输出		
PLLRCA	PLL 滤波器	输入		
PLLRCB	PLL 滤波器	输入		
PCK0-PCK3	可编程时钟输出	输出		
掉电，唤醒逻辑				
SHDN	掉电控制	输出		不要和 VDDBU 接一起
WKUP	唤醒输入	输入		接受 0V-VDDBU
ICE 和 JTAG				
TCK	测试时钟	输入		无上拉电阻
RTCK	返回的测试时钟	输出		无上拉电阻
TDI	测试数据输入	输入		无上拉电阻
TDO	测试数据输出	输出		
TMS	测试模式选择	输入		无上拉电阻
NTRST	测试复位信号	输入	低电平	上拉电阻
JTAGSEL	JTAG 选择	输入		下拉电阻，识别电压 0V-VDDBU
ETM				
TSYNC	追踪同步信号	输出		
TCLK	追踪时钟	输出		
TPS0-TPS2	追踪 ARM 管道状态	输出		
TPK0-TPK15	追踪信息包端口	输出		

表 3-1.信号描述列表（续）

信号名称	功能	类型	有效电平	备注
复位/测试				
NRST	微控制器复位	I/O	低电平	上拉电阻
TST	测试模式选择	输入		下拉电阻
BMS	启动模式选择	输入		
调试部件				
DRXD	调试数据接收	输入		
DTXD	调试数据发送	输出		
AIC				
IRQ0-IRQ2	外部中断输入	输入		
FIQ	快速中断输入	输入		
PIO				
PA0-PA31	并行 IO 控制器 A	I/O		复位上拉输入
PB0-PB31	并行 IO 控制器 B	I/O		复位上拉输入
PC0-PC31	并行 IO 控制器 C	I/O		复位上拉输入
EBI				
D0-D31	数据总线	I/O		复位上拉输入
A0-A25	地址总线	输出		复位 0
NWAIT	外部等待信号	输入	低电平	
SMC				
NCS0-NCS7	片选线	输出	低电平	
NWR0-NWR3	写信号	输出	低电平	
NRD	读信号	输出	低电平	
NWE	写使能	输出	低电平	
NBS0-NBS3	字节屏蔽信号	输出	低电平	
CompactFlash 支持				
CFCE1-CFCE2	CompactFlash 芯片使能	输出	低电平	
CFOE	CompactFlash 输出使能	输出	低电平	
CFWE	CompactFlash 写使能	输出	低电平	
CFIOR	CompactFlash IO 读	输出	低电平	
CFIOW	CompactFlash IO 写	输出	低电平	
CFRNW	CompactFlash 只读	输出	低电平	
CFCS0-CFCS1	CompactFlash 片选	输出	低电平	
NAND Flash 支持				
NANDOE	NAND Flash 输出使能	输出	低电平	
NANDWE	NAND Flash 写使能	输出	低电平	
NANDCS	NAND Flash 片选	输出	低电平	

表 3-1.信号描述列表(续)

信号名称	功能	类型	有效电平	备注
SDRAM 控制器				
SDCK	SDRAM 时钟	输出		
SDCKE	SDRAM 时钟使能	输出	高电平	
SDCS	SDRAM 控制器片选	输出	低电平	
BA0-BA1	Bank 选择	输出		
SDWE	SDRAM 写使能	输出	低电平	
RAS-CAS	行列信号	输出	低电平	
SDA10	SDRAM 地址 10	输出		
多媒体卡接口				
MCCK	多媒体卡时钟	输出		
MCCDA	多媒体卡 A 命令	I/O		
MCDA0-MCDA3	多媒体卡 A 数据	I/O		
USART				
SCK0-SCK2	串行时钟	I/O		
TXD0-TXD2	数据传送	输出		
RXD0-RXD2	数据接收	输入		
RTS0-RTS2	发送请求	输出		
CTS0-CTS2	发送清零	输入		
同步串行控制器				
TD0-TD2	传输数据	输出		
RD0-RD2	接收数据	输入		
TK0-TK2	传输时钟	I/O		
RK0-RK2	接收时钟	I/O		
TF0-TF2	传输帧同步	I/O		
RF0-RF2	接受帧同步	I/O		
定时器/计数器				
TCLK0-TCLK2	外部始钟输入	输入		
TIOA0-TIOA2	I/O 口线 A	I/O		
TIOB0-TIOB2	I/O 口线 B	I/O		
SPI				
SPI0_MISO-SPI1_MISO	主输入从输出	I/O		
SPI0_MISO-SPI1_MISO	主输出从输入	I/O		
SPI0_SPCK-SPI1_SPCK	SPI 串行时钟	I/O		

表 3-1.信号描述列表(续)

信号名称	功能	类型	有效电平	备注
SPI0_NPCS0- SPI1_NPCS0	SPI 外设片选 0	I/O	低电平	
SPI0_NPCS1- SPI0_NPCS3 SPI1_NPCS1- SPI_NPCS3	SPI 外设片选	输出	低电平	
2-Wire 接口（IIC）				
TWD	2-Wire 串行数据	I/O		
TWCK	2-Wire 串行时钟	I/O		
LCD 控制器				
LCDD0-LCDD23	LCD 数据总线	输出		
LCDVSYNC	LCD 垂直同步	输出		
LCDHSYNC	LCD 水平同步	输出		
LCDDOTCK	LCD 点阵数据时钟	输出		
LCDDEN	LCD 数据使能	输出		
LCDCC	LCD 对比度控制	输出		
USB 设备端口				
DDM	USB 设备端口数据-	模拟		
DDP	USB 设备端口数据+	模拟		
USB 主机端口				
HDMA	USB 主机端口 A 数据-	模拟		
HDPA	USB 主机端口 A 数据+	模拟		
HDMB	USB 主机端口 B 数据-	模拟		
HDPB	USB 主机端口 B 数据+	模拟		

四 封装和引脚排列

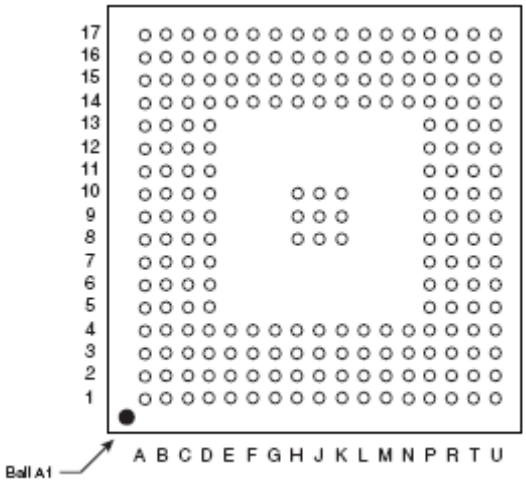
AT91SAM9261 是一个符合 RoHS 的 217 球的 LFBGA 封装，15*15mm，0.8mm 的焊球间距

4.1 217-球 LFBGA 封装轮廓

图 4-1 217-球 LFBGA 封装的定位。

具体的机械描述在产品数据手册的”AT91SAM9261 机械特性”一章中有介绍

图 4-1. 217-球 LFBGA 封装轮廓(正面俯视)



4.2 引脚排列

表 4-1. AT91SAM9261 引脚排列(217-球 LFBGA 封装)

Pin	Signal Name
A1	A19
A2	A16/BA0
A3	A14
A4	A12
A5	A9
A6	A6
A7	A3
A8	A2
A9	NC
A10	XOUT32
A11	XIN32
A12	DDP
A13	HDPB
A14	HDMB
A15	PB27
A16	GND
A17	PB24
B1	A20
B2	A18
B3	A15
B4	A13
B5	A11
B6	A7
B7	A4
B8	A1/NBS2/NWR2
B9	VDDBU
B10	JTAGSEL
B11	WKUP
B12	DDM
B13	PB31
B14	HDMA
B15	PB26
B16	PB25
B17	PB19
C1	A22
C2	A21
C3	VDDIOM
C4	A17/BA1
C5	VDDIOM
C6	A8
C7	GND
C8	VDDIOM
C9	GNDBU
C10	TST
C11	GND
C12	HDPA
C13	PB30
C14	NC
C15	VDDIOP
C16	PB21
C17	TMS
D1	NCS2
D2	NCS1/SDCS
D3	GND
D4	VDDIOM

Pin	Signal Name
D5	VDDCORE
D6	A10
D7	A5
D8	A0/NBS0
D9	SHDN
D10	NC
D11	VDDIOP
D12	PB29
D13	PB28
D14	PB23
D15	PB20
D16	PB17
D17	TCK
E1	NWR1/NBS1/CFIOR
E2	NWR0/NWE/CFWE
E3	NRD/CFOE
E4	SDA10
E14	PB22
E15	PB18
E16	PB15
E17	TDI
F1	SDCKE
F2	RAS
F3	NWR3/NBS3/CFIOW
F4	NCS0
F14	PB16
F15	NRST
F16	TDO
F17	NTRST
G1	D0
G2	D1
G3	SDWE
G4	NCS3/NANDCS
G14	PB14
G15	PB12
G16	PB11
G17	PB8
H1	D2
H2	D3
H3	VDDIOM
H4	SDCK
H8	GND
H9	GND
H10	GND
H14	PB10
H15	PB13
H16	PB7
H17	PB5
J1	D4
J2	D5
J3	GND
J4	CAS
J8	GND
J9	GND
J10	GND

Pin	Signal Name
J14	VDDIOP
J15	PB9
J16	PB6
J17	PB4
K1	D6
K2	D8
K3	D10
K4	D7
K8	GND
K9	GND
K10	GND
K14	VDDCORE
K15	PB3/BMS
K16	PB1
K17	PB2
L1	D9
L2	D11
L3	D12
L4	VDDIOM
L14	PA30
L15	PA27
L16	PA31
L17	PB0
M1	D13
M2	D15
M3	PC18
M4	VDDCORE
M14	PA25
M15	PA26
M16	PA28
M17	PA29
N1	D14
N2	PC17
N3	PC31
N4	VDDIOM
N14	PA22
N15	PA21
N16	PA23
N17	PA24
P1	PC16
P2	PC30
P3	PC22
P4	PC24
P5	PC28
P6	PC1
P7	PC7
P8	PC11
P9	GNDPLL
P10	PA3
P11	VDDIOP
P12	VDDCORE
P13	PA15
P14	PA16
P15	VDDIOP
P16	PA19

Pin	Signal Name
P17	PA20
R1	PC19
R2	PC21
R3	GND
R4	PC27
R5	PC29
R6	PC4
R7	PC8
R8	PC12
R9	PC14
R10	VDDPLL
R11	PA0
R12	PA7
R13	PA10
R14	PA13
R15	PA17
R16	GND
R17	PA18
T1	PC20
T2	PC23
T3	PC26
T4	PC2
T5	VDDIOP
T6	PC5
T7	PC9
T8	PC10
T9	PC15
T10	VDDOSC
T11	GNDOSC
T12	PA1
T13	PA4
T14	PA6
T15	PA8
T16	PA11
T17	PA14
U1	PC25
U2	PC0
U3	PC3
U4	GND
U5	PC6
U6	VDDIOP
U7	GND
U8	PC13
U9	PLLRCB
U10	PLLRCA
U11	XIN
U12	XOUT
U13	PA2
U14	PA5
U15	PA12
U16	PA9
U17	RTCK

注意：1.阴影部分引脚由 VDDIOM 提供电源

五 电源事宜

5.1 电源

AT91SAM9261 有 6 种类型的电源引脚：

- VDDCORE 引脚：核心电源，包括处理器，存储器和外设；电压范围从 1.08V 到 1.32V，额定值 1.2V
- VDDIOM 引脚：外部总线接口 I/O 口线电源；电压范围从 1.65V-1.95V 和 3.0V-3.6V, 额定值 1.8V-3.3V
- VDDIOP 引脚: 外设 I/O 口线和 USB 收发器电源；电压范围 2.7V-3.6V, 额定值 3.3V
- VDDBU 引脚：慢速时钟振荡器和部分系统控制器电源；电压范围 1.08V-1.32V, 额定值 1.2V
- VDDPLL 引脚：PLL 部件电源；电压范围 3.0V-3.6V, 额定值 3.3V
- VDDOSC 引脚：主振荡器部件电源；电压范围 3.0V-3.6V, 额定值 3.3V

VDDIOM 和 VDDIOP 的双电源供应在第 10 页表 4-1 中有介绍。这些电源使得用户能对存储器接口和外设接口设备分配不同的电源。
接地引脚 GND 是 VDDCORE,VDDUIM 和 VDDIOP 引脚电源的公共端。VDDBU,VDDOSC 和 VDDPLL 分别使用了各自的电源地名称。分别为 GNDBU,GNDOSC 和 GNDPLL。

5.2 功耗

AT91SAM9261 在 25 摄氏度时在 VDDCORE 上的静态电流损耗大约 550 微安。当温度上升到 85 摄氏度时静态电流上升到 5.5 毫安。
在 VDDBU 上，25 摄氏度时，电流不超过 3 微安，但是 85 摄氏度时能上升到 20 微安。
对于动态功率损耗，AT91SAM9261 在典型条件下(1.2V,25 摄氏度)以最大速率，处理器运行在全速运算状态，VDDCORE 电源消耗最大 50 毫安电流。

六． I/O 口线事宜

6.1 JTAG 端口引脚

TMS,TDI 和 TCK 是施密特触发器输入而且无上拉电阻。
TDO 和 RTCK 是输出，输出电压可以达到 VDDIOP，且无上拉电阻。
当 JTAGSEL 引脚维持高电平(接到 VDDBU)时被用作 JTAG 边界扫描。此引脚集成了一个连接于 GNDBU 的 15K 欧姆的下拉电阻。所以在正常运行时可以悬空。
当 NTRST 引脚维持低电平时被用来初始化嵌入式 ICE TAP 控制器。此引脚集成了一个连接到 VDDIOP 的 15K 欧姆上拉电阻，所以正常运行时可以悬空。

6.2 测试引脚

当测试引脚(TST)维持高电平(有效)时被用作生产测试目的。此引脚集成了一个连接于 GNDBU 的 15K 欧姆的永久上拉电阻，所以正常运行时可以悬空。当以高电平驱动此引脚时将导致难以预料的结果。

6.3 复位引脚

NRST 是一个集成了一个不可编程的上拉电阻的开漏输出。
输出电压可以达到 VDDIOP。由于 9261 集成了上电复位单元，当没有外部复位电路的情况下，NRST 引脚也可以直接悬空处理。
NRST 引脚集成了一个至少为 100K 欧姆的永久上拉电阻并连接到 VDDIOP。
NRST 信号嵌于边界扫描中。

6.4 PIO（并行 IO）控制器 A,B 和 C 口线

所有的 I/O 口线 PA0 到 PA31，PB0 到 PB31，和 PC0 到 PC31 都集成了一个 100K 欧姆可编程的上拉电阻，PIO 控制器可以对每个 I/O 口线的上拉电阻进行控制。
复位后，所有的 I/O 块都缺省为上拉电阻使能，除了那些在复位时，要求和外部总线接口信号多路复用的 I/O 口线被使能为外设。这一点在 PIO 控制器多路复用表里的‘复位状态’栏里已明确指出。

6.5 掉电逻辑引脚(SHDN)

SHDN 引脚由掉电控制器驱动，是一个输出引脚。
唤醒引脚(WKUP)是一个输入引脚。他仅接收 0V 到 VDDBU 之间的电压信号。

七 处理器和体系结构

7.1 ARM926EJ-S 处理器

- 基于 ARM v5TEJ 体系结构带 Jazelle Java 加速技术的 RISC 处理器
- 两个指令集
  -ARM 高性能 32 位指令集
  -Thumb 高编码密集度 16 位指令集
- DSP 指令扩展
- 五级流水线体系结构
  - 指令获取(F)
  - 指令解码(D)
  - 指令执行(E)
  - 数据存储(M)
  - 寄存器写（W）
- 16K 字节数据缓存，16K 字节指令缓存
  - 虚拟地址4路相连缓存(Virtually-addressed 4-way Associative Cache)
  - 八个字每路
  - 写通(write through)和回写(write back)操作
  - 伪随机或轮转法更新缓存条目
- 写缓冲器
  - 16 字数据缓冲和 4 地址缓冲的主写缓冲

- 8 字入口和一个单独的地址入口的 DCache 写返回缓冲
- Software Control Drain

• 标准的 ARM v4 和 v5 存储器管理部件(MMU)
- 段存取允许
- 对每四分之一的大页和小页能被分开明确指定的存取权限
- 16 个嵌入的区域

• 总线接口部件(BIU)
- 仲裁和调度 AHB 请求
- 为指令和数据存取提供完全的 AHB 系统灵活性的分离的主控
- 32 位指令接口和 32 位数据接口的分离的地址和数据总线
- 在地址和数据总线上，数据可以是 8 位(字节)，16 位（半字）或 32 位（字）

7.2 调试和测试功能部件

• 集成嵌入式的实时在线仿真器
- 两个实时监视点部件
- 两个独立的寄存器：调试控制寄存器和调试状态寄存器
- 可通过 JTAG 协议进入的测试存取端口
- 调试通信通道

• 调试部件
- 两线 UART
- 调试通信通道中断控制
- 芯片 ID 寄存器

• 嵌入式跟踪宏单元：ETM9
- 中等以上规模实现
- 半速率时钟模式
- 四对地址比较器
- 两个数据比较器
- 八个存储器映射解码器输入
- 两个 16 位计数器
- 一个3级Sequencer
- 一个 45 字节的 FIFO

• 在所有的数字引脚上的 IEEE1149.1JTAG 边界扫描

7.3 总线矩阵

• 五个主控和五个从控
- 处理来自 ARM926EJ-S,USB 主端口，LCD 控制器和外围的 DMA 控制器到内部 ROM，内部 SRAM，EBI,APB，LCD 控制器和 USB 主端口的请求。
- 轮转法仲裁（支持三种模式：无缺省主控，最后的存取缺省主控，固定的缺省主控）
- 随存储槽周期（Slot Cycle）限制的突发中断

• 每个主控提供一个地址解码器
- 三个不同的从控可以被指定给每一个解码的存储区域：一个给内部启动，一个给外部启动，一个在重映射后。

• 启动模式选项

- 非易失的启动存储器可以是内部的或外部的
- 依据复位时 BMS 引脚的电平决定启动选项

4 重映射命令

- 允许一个内部 SRAM 重映射来代替启动非易失性存储器
- 允许异常向量的动态配置处理

7.4 外围 DMA 控制器

• 在没有处理器干预下从外围转存到任何存储空间或从任何存储空间转存到外围。

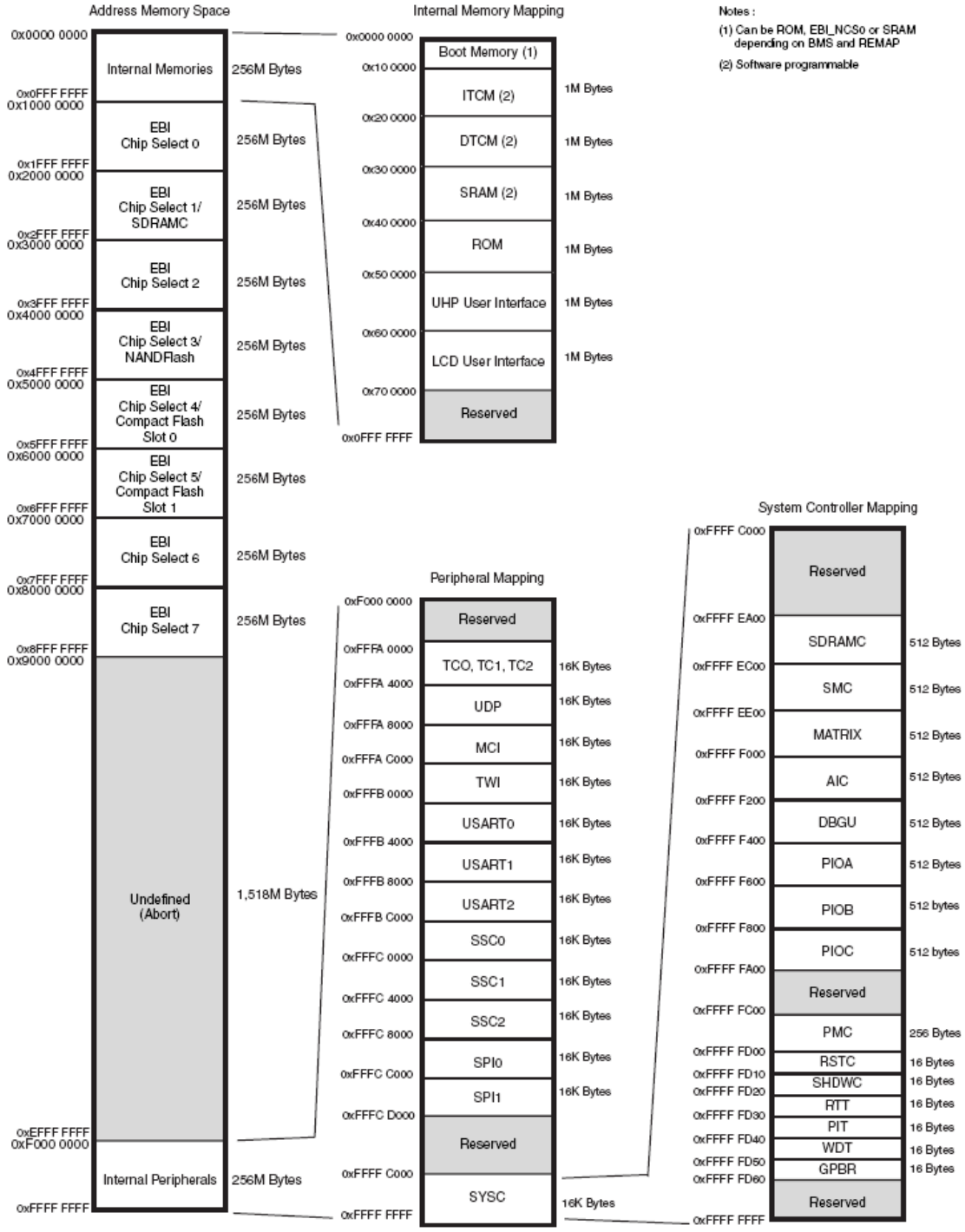
• 下一个指针支持，在缓冲器管理上禁止强实时约束

• 十九个通道

- 每个 USRAT 有两个
- 调试部件有两个
- 每个串行同步控制器有两个
- 每个串行外围接口有两个
- 多媒体卡接口有一个

8 存储器

图 8-1 AT91SAM9261 存储器映射



总线矩阵完成一个初级的地址解码，也就是，先进的高性能总线(AHB)对其附加功能部件的主控和从控接口的实现。

解码时把 4 G 字节的地址空间分成 16 个 256 M 字节的区域，1-8 号区域通过片选信号 NCS0－NCS7 指向到 EBI。区域 0 是为片内存储器寻址的保留区域，并且一个二级解码提供 1M 字节的内部存储区。15 号区域是为外设预留的且提供访问先进的外围总线(APB)的权限.

其他的区域是未被使用的，当在该区域内进行存取操作时，会向主控提供一个中断。
总线矩阵管理五个主控和五个从控。
每个主控有其自己的总线和其自己的解码器，这样就允许每个主控有一个不同的存储器映射。
对于主控 0 和主控 1(ARM926 指令和数据)，三个不同的从控在地址 0x0 被指定到存储空间解码：一个给内部启动，一个给外部启动，一个在重映射后。细节见表 8-3。

表 8-1 总线矩阵主控器列表

主控 0	ARM926 指令
主控 1	ARM926 数据
主控 2	PDC
主控 3	LCD 控制器
主控 4	USB 主机

每个从控有其自己的仲裁，这样就允许每个从控有一个不同的仲裁。

表 8-2 总线矩阵从控器列表

从控 0	内部 SRAM
从控 1	内部 ROM
从控 2	LCD 控制器和 USB 主机端接口
从控 3	外部总线接口
从控 4	内部外设

8.1 嵌入式存储器

- 32KB ROM
  - 全速总线下单周期存取
- 160KB 快速 SRAM
  - 全速总线下单周期存取
  - 处理器全速下支持 ATM926EJ-S TCM 接口

8.1.1 内部存储器映射

表 8-3 列出了每个主控的内部存储器映射，取决于复位时重映射状态和 BMS 状态。

表 8-3 内部存储映射

Address	主控 0：ARM926 指令			主控 1：ARM926 数据		
	REMAP(RCB0)=0		REMAP(RCB0)=1	REMAP(RCB1)=0		REMAP(RCB)=1
	BMS=1	BMS=0		BMS=1	BMS=0	
0x0000 0000	Int.ROM	EBI NCS0	Int. RAM C	Int. ROM	EBI NCS0	Int. RAM C

注意：1. EBI NCS0 被用于连接一个 16 位宽的非易失性存储器。存储结构由复位时的 SMC 设置，SMC 脉冲，SMC 时钟周期和 SMC 模式 CS0 寄存器的状态所定义。

8.1.1.1 内部 SRAM

AT91SAM9261 集成了一个 160K 字节的高速 SRAM。这个内部的 SRAM 被分为三个区域。其存储器映射的细节见上面表 8-3。

• 内部 SRAM A 是 ATM926EJ-S 指令 TCM 并且用户可以通过 CP15 指令在 ARM926 指令存储空间映射 SRAM 块到任何地方。SRAM 块可以通过 AHB 总线在地址 0x0010 0000 被 ARM926 数据主控和 AHB

主控访问。

• 内部 SRAM B 是 ATM926EJ-S 数据 TCM 并且用户可以通过 CP15 指令在 ARM926 数据存储空间映射 SRAM 块到任何地方。SRAM 块可以通过 AHB 总线在地址 0x0020 0000 被 ARM926 数据主控和 AHB 主控访问。

• 内部 SRAM C 只能被所有的 AHB 主控存取。

复位后且直到重映射命令执行前，SRAM 块能被所有的 AHB 主控通过 AHB 总线 在地址 0x0030 0000 访问。重映射后，SRAM 块也能被 ARM926 指令和 ARM926 数据主控通过 AHB 总线在地址 0x0 访问。在 160k 字节 SRAM 可用空间，被指定到每个块的 16K 的倍数字节（见表 8-4）的存储量是可以软件编程的。此表提供了根据内部 SRAM A 和内部 SRAM B 的大小决定内部 SRAM C 的存储量。

表 8-4 内部 SRAM 块存储量

内部 SRAM C		内部 SRAM A(ITCM)			
		0	16K 字节	32 K 字节	64 K 字节
内部 SRAM B(DCTM)	0	160 K 字节	144 K 字节	128 K 字节	96 K 字节
	16 K 字节	144 K 字节	128 K 字节	112 K 字节	80 K 字节
	32 K 字节	128 K 字节	112 K 字节	96 K 字节	64 K 字节
	64 K 字节	96 K 字节	80 K 字节	64 K 字节	32 K 字节

注意：在 10 个 16K 字节块构成的内部 SRAM，两个被永久指定为内部 SRAM C.

复位后，整个存储器（160KB）被指定为内部 SRAM C。

被指定为 SRAM A,SRAM B,SRAM C 的存储块并不是连续的，当用户动态地改变内部 SRAM 结构时，新的 16K 字节块组织将从软件方面影响先前的结构。

表 8-5 16K 字节块分配

解码区	地址	结构举例和 16K 字节块指定详情			
		ITCM =0k 字节 DTCM=0k 字节 AHB=160k 字节（1）	ITCM=64k 字节 DTCM=64k 字节 AHB=32k 字节	ITCM=32k 字节 DTCM=64k 字节 AHB=64k 字节	ITCM=32k 字节 DTCM=16k 字节 AHB=112k 字节
内部 SRAM A (ITCM)	0x0010 0000		RB3	RB3	RB3
	0x0010 4000		RB2	RB2	RB2
	0x0010 8000		RB1		
	0x0010 C000		RB0		
内部 SRAM B DTCM	0x0020 0000		RB7	RB7	RB7
	0x0020 4000		RB6	RB6	
	0x0020 8000		RB5	RB5	
	0x0020 C000		RB4	RB4	
内部 SRAM B (AHB)	0x0030 0000	RB9	RB9	RB9	RB9
	0x0030 4000	RB8	RB8	RB8	RB8
	0x0030 8000	RB7		RB1	RB2
	0x0030 C000	RB6		RB0	RB5
	0x0031 0000	RB5			RB4

	0x0031 4000	RB4			RB1
	0x0031 8000	RB3			RB0
	0x0031 C000	RB2			
	0x0032 0000	RB1			
	0x0032 4000	RB0			

注意：（1）复位后的配置信息

8.1.1.2 内部 ROM

AT91SAM9261 集成了一个映射地址为 0x0040 0000 的 32KB 的内部 ROM。如果复位期间 BMS 是高电平，也可以复位后在地址 0x0 和重映射前访问。

8.1.1.3 USB 主端口

AT91SAM9261 集成了一个 USB 主端口开放主机控制器接口(OHCI). 此接口的寄存器在 AHB 总线上可直接访问并且能像标准的内部存储器一样在地址 0x0050 0000 被映射。

8.1.1.4 LCD 控制器

AT91SAM9261 集成了一个 LCD 控制器。其接口的寄存器在 AHB 总线上可直接访问并且能像标准的内部存储器一样在地址 0x0060 0000 被映射。

8.1.2 启动策略

系统总是在地址 0x0 启动。为了确保最高的成功启动概率，储存器布局可由两个参数配置。

重映射(REMAP)允许用户映射第一个内部 SRAM 存储器到 0x0。只要矩阵总线的每个主控已经启动，REMAP 可由软件完成。当 REMAP=1，BMS 被忽略。更多细节请参考“Bus Matrix Section”。

当 REMAP=0，BMS 允许用户展开到 0x0，可以是 ROM 或外部存储器，复位时由硬件完成。

注意：不被这些参数影响的存储器块总是在他们特殊的指定地址能被看见。完全的存储器映射请参考 16 页图 8 -1。

AT91SAM9261 总线矩阵管理了一个在复位时取决于 BMS 引脚上电平的启动存储器。在地址 0x0 和 0x000F FFFF 间映射的内部存储器区被预留用于此目的。

如果 BMS 检测到 1，启动的存储器是嵌入的 ROM。

如果 BMS 检测到 0，启动的存储器是连接于外部总线接口片选 0 的存储器。

8.1.2.1 BMS=1，在嵌入的 ROM 上启动

系统用启动程序启动。

- DataFlash 启动
  - -从 SPI DataFlash 进入到内部 SRAM 下载并运行一个应用程序
  - -从 SPI DataFlash 调入的代码大小受 SRAM 大小限制
  - -自动监测有效的应用程序
  - -SPI DataFlash 连接到 SPI NPCSO
- 万一在外部 SPI DataFlash 检测到无有效的应用程序则启动上传
  - -SAM-BA 应用程序，一种小型监控程序(读/写/运行)接口
  - -自动检测通信连接

一个 DBGU（XModem 协议）上的串行通信
USB 设备端口(CDC 协议)

8.1.2.2 BMS=0,从外部存储器启动

- 慢时钟(32,768Hz)启动
- 带缺省配置的静态存储控制器启动，字节选择模式，16 位数据总线，片选控制的读/写，允许在 16 位非易失存储器上启动。

客户编程的软件必须完成了一个完整的配置。

当在 32kHz EBI CS=0(BMS=0)时，为了加速启动次序，用户必须执行以下步骤：

1. 编程 PMC（主振荡器使能或旁路模式）。
2. 编程并启动 PLL。
3. 重编程 SMC 设置，周期，保持，CS0 的模式定时寄存器以适应新时钟。
4. 主时钟切换至新值

8.1.3 ETM 存储器

八个 ETM9 中等规模以上（Medium+）存储器映射解码器输入被连接到客户地址解码器。

结果的存储器映射总结在表 8-6 中

表 8-6。ETM9 存储器映射

产品资源	区域	存取类型	起始地址	结束地址
SRAM	内部	数据	0x0000 0000	0x002F FFFF
SRAM	内部	取	0x0000 0000	0x002F FFFF
ROM	内部	数据	0x0040 0000	0x004F FFFF
ROM	内部	取	0x0040 0000	0x004F FFFF
外部总线接口	外部	数据	0x1000 0000	0x8FFF FFFF
外部总线接口	外部	取	0x1000 0000	0x8FFF FFFF
用户外设	内部	数据	0xF000 0000	0xFFFF BFFF
系统外设	内部	数据	0xFFFF C000	0xFFFF FFFF

8.2 外部存储器

外部存储器可以通过外部总线接口（总线矩阵从控 3）访问。

存储器映射见 16 页表 8-1

9. 系统控制器

系统控制器管理所有重要的微控制器块：中断，时钟，电源，定时器，调试和复位。

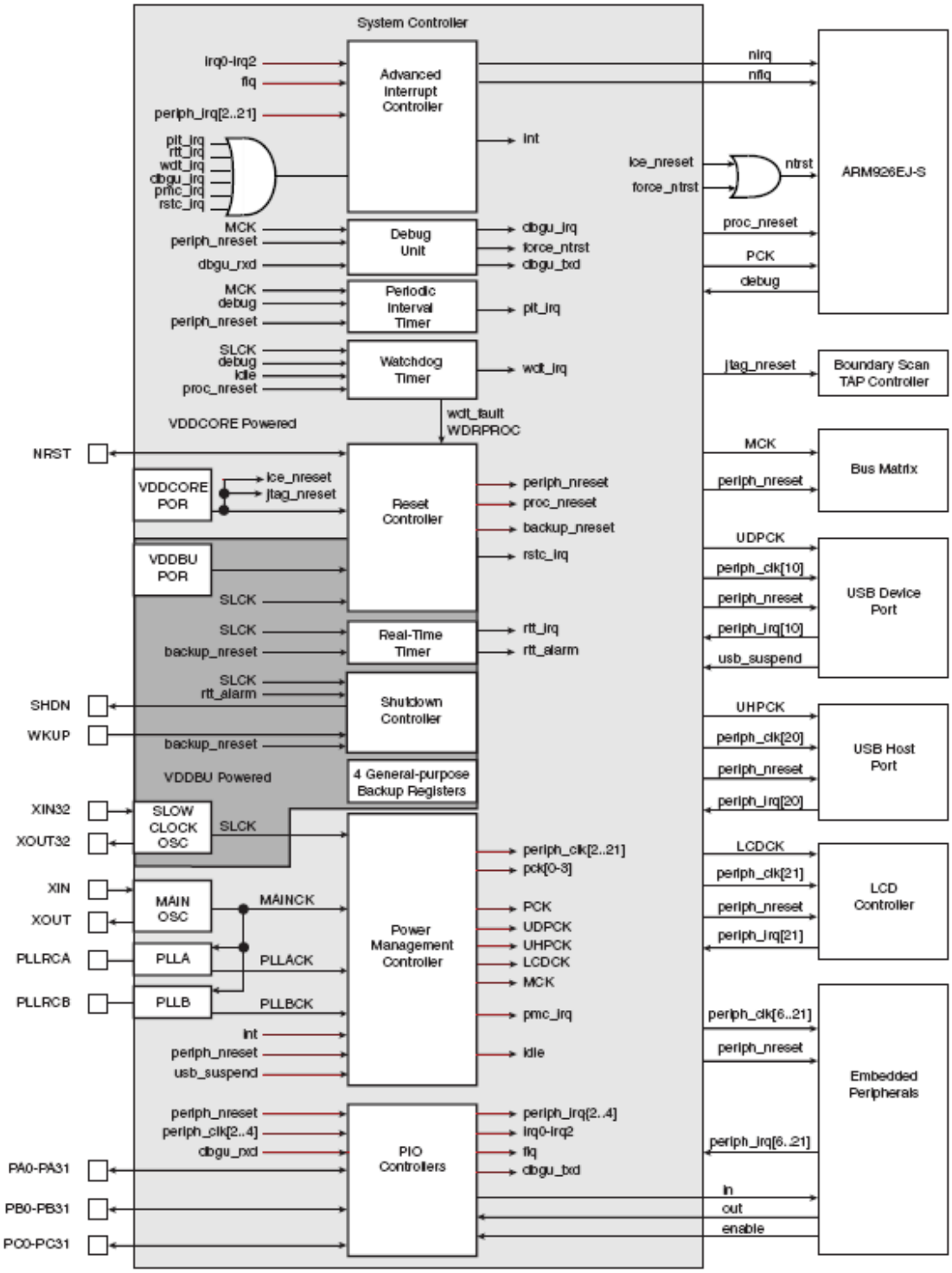
系统外设全被映射在最高的 6K 字节的地址空间内，地址 0xFFFF EA00 到 0xFFFF FFFF 之间。每个外设有一个 256 或 512 字节的地址空间，表示 64 或 128 寄存器。

23 页表 9-1 显示了系统控制器方块图

16 页表 8-1 显示了系统控制器外设用户接口的映射。

9.1 方块图

图 9-1 系统控制器方块图



9.2 复位控制器

- 基于两个 Power-on-Reset 单元
- 最近一次复位状态
  -冷复位，首次复位，软件复位，用户复位，看门狗复位，唤醒复位
- 控制内部复位和 NRST 引脚输出

9.3 掉电控制器

- 掉电和唤醒逻辑：
  -软件可编程的 SHDN（Shutdown）引脚
  -WKUP 引脚电平改变或报警时可触发

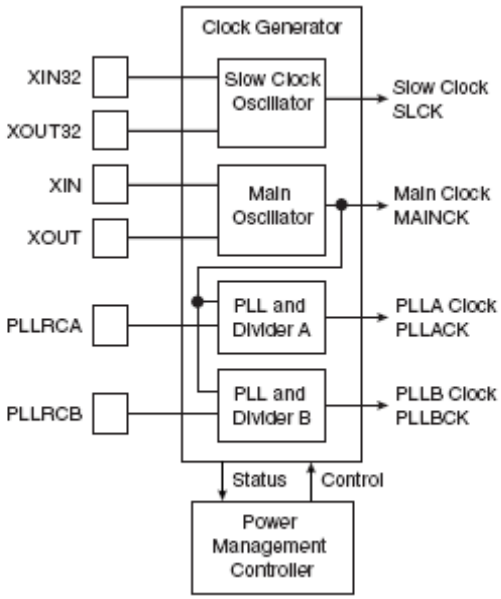
9.4 通用备份寄存器

- 四个 32 位通用备份寄存器

9.5 时钟发生器

- 集成低功耗 32768Hz 慢时钟振荡器
  -向系统提供永久慢时钟
- 集成主振荡器
  - 振荡器旁路功能
  - 支持 3 到 20MHz 晶振
- 集成两个 PLL
  - 输出 80 到 240MHz 时钟
  - 集成了一个输入分频器来增大输出精度
  - 1MHz 最小输入频率
- 提供 SLCK,MAINCK,PLLACK 和 PLLBCK。

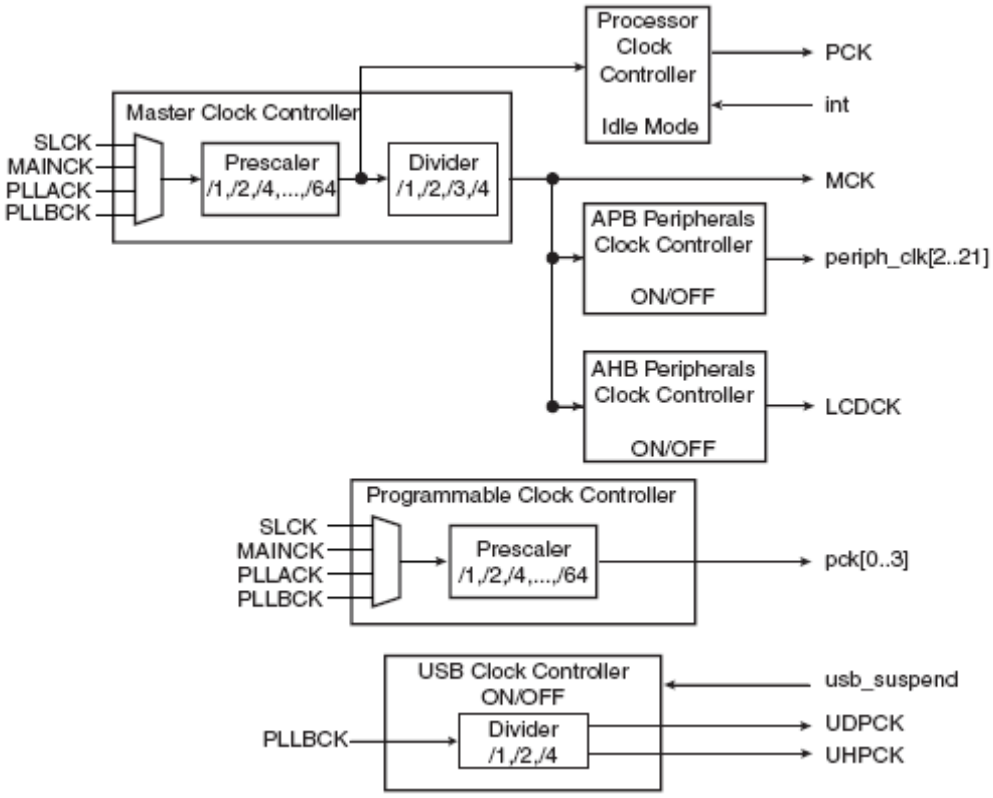
图 9-2　时钟发生器方块图



9.6 电源管理控制器

- 电源管理控制器提供：
  -处理器时钟(PCK)
  -主控时钟(MCK)
  -USB 时钟 USBCK (HCK0)
  -LCD 控制器时钟 LCDCK (HCK1)
  -多至 30 路外部时钟
  -四个可编程时钟输出：PCK0 到 PCK3

图 9-3。电源管理控制器方块图



9.7 周期性间隔定时器

- 包括一个 20 位的精确度优于 1 微秒的周期计数器
- 包括一个 12 位间隔覆盖计数器
- 实时操作系统或 Linux/Windows CE 适用的节拍发生器

9.8 看门狗定时器

- 12 位键保护（Key-protected）一次性编程计数器
- 窗口的，在看门狗的访问下防止处理器进入死锁

9.9 实时定时器

- 32 位自由运行备份计数器
- 报警寄存器可以发生一个系统唤醒信号

9.10 先进的中断控制器

- 控制一个 ARM 处理器的中断口线（nIRQ 和 nFIQ）
- 32 个独立可屏蔽的和中断向量源
  - 中断源 0 为快速中断输入(FIQ)预留。
  - 中断源 1 为系统外设（PIT,RTT,PMC,DBGU,等）预留
  - 中断源 2 到中断源 31 控制多达 30 个嵌入式外设中断或外部中断
  - 可编程的边沿触发或电平触发的内部中断源
  - 可编程的正/负边沿触发或高低电平触发
- 四个外部中断源
- 8 个优先级控制器
  - 驱动一般的处理器中断
  - 处理 1 到 31 号中断源的优先级
  - 在低优先级中断服务期间优先响应高优先级中断
- 定向
  - 优化中断服务路径和执行
  - 每个中断源对应一个 32 位的定向寄存器（Vector Register）
  - 中断定向寄存器读取相对应的当前中断向量
- 保护模式
  - 处于保护模式时，通过阻止自动操作来方便调试
- 快速强制（Fast Forcing）
  -允许重定向任何普通的中断源到 FIQ
- 一般中断屏蔽
  - 在事件没有触发一个中断时提供处理器同步化

9.11 调试部件

- 由 4 部分功能组成
  - 双引脚 UART
  - 调试通信通道(DCC)支持
  - 芯片 ID 寄存器
  - ICE 访问控制
- 双引脚 UART
  - 功能和标准的 Atmel USART 100%完全兼容
  - 带通用可编程波特率发生器的独立接收器和发射器
  - 偶校验，奇校验，1 或 0 奇偶校验
  - 奇偶校验，成帧（Framing）和超限错误（Overrun Error）检测
  - 自动回显，本地回送和远程回送通道模式
  - 支持接收器和发射器的两个 PDC 通道
- 调节通信通道支持
  - 从 ARM 处理器提供 COMMRX 和 COMMTX 信号的可见性
- 芯片 ID 寄存器
  - 器件版本 ID，嵌入式存储器大小，外设
- ICE 访问保护
  - 可以通过软件禁用 ARM 处理器的 ICE 访问
  - 通过使 ARM 处理器 ICE 的 NTRST 信号有效实施保护

9.12 PIO 控制器

- 三个 PIO 控制器，每个控制达 32 个可编程 I/O 口线
  - PIOA 有 32 个 I/O 口线
  - PIOB 有 32 个 I/O 口线
  - PIOC 有 32 个 I/O 口线
- 通过设置/清零寄存器完全可编程
- 每个 I/O 口线与两个外设功能多路复用
- 对于每个 I/O 口线（无论被指定为一个外设或被用于通用 I/O）
  - 输入改变中断
  - 尖峰滤波器（Glitch filter）
  - 多驱动选项使能漏极开路驱动
  - 每个 IO 口线均有可编程上拉电阻
  - 引脚数据状态寄存器，任何时间提供引脚上电平的可见性
- 同步输出，实现在单周期写时对若干 I/O 口线的置位和清零

10. 外设

10.1 用户接口

用户外设被映射于地址空间介于地址 0xFFFA 0000 和 0xFFFC FFFF 的高 256M 字节。每个用户外设都分配了 16K 字节的地址空间。

完全的存储器映射见前述 16 页图 8-1

1.2 外设识别

表 10-1 定义了 AT91SAM9261 的外设识别。一个外设识别需要用先进的中断控制器对外设中断的控制并且用电源管理控制器对外设时钟的控制。

表 10-1　外设识别

外设 ID	外设助记符	外设名称	外部中断
0	AIC	先进的中断控制器	FIQ
1	SYSIRQ	系统中断	
2	FIOA	并行 I/O 控制器 A	
3	FIOB	并行 I/O 控制器 B	
4	PIOC	并行 I/O 控制器 C	
5	-	预留	
6	US0	USART0	
7	US1	USART1	
8	US2	USART2	
9	MCI	多媒体卡接口	
10	UDP	USB 设备端口	
11	TWI	双总线接口	
12	SPI0	串行外设接口 0	
13	SPI1	串行外设接口 1	
14	SSC0	同步串行控制器 0	
15	SSC1	同步串行控制器 1	
16	SSC2	同步串行控制器 2	
17	TC0	定时器/计数器 0	
18	TC1	定时器/计数器 1	
20	UHP	USB 主机端口	

21	LCDC	LCD 控制器	
22~28	-	保留	
29	AIC	先进的中断控制器	IRQ0
30	AIC	先进的中断控制器	IRQ1
31	AIC	先进的中断控制器	IRQ2

注意：通过在 PMC 的时钟置位/清零寄存器来设置 AIC，SYSIRQ,UHP,LCDC 和 IRQ0－IRQ2 位是无效的。

10.3 PIO 口线上的外设复用

AT91SAM9261 设置了三个 PIO 控制器，PIOA,PIOB 和 PIOC，这些 I/O 口线和外设进行了复用。

每个 PIO 控制器控制达 32 个口线。每个口线可以被指定到两个外围功能中的一个，A 或 B。32 页表 10-2，33 页表 10-3 和 34 页表 10-4 定义了外设 A 和 B 的 I/O 口线怎样被多路复用于 PIO 控制器。“功能”和“备注”两栏插入了用户自己的备注;他们可能被用于追踪在一个应用程序里被定义成怎样的引脚。

注意一些只有输出的外围功能在表中可能被重复定义。

“复位状态”一栏显示复位后 PIO 口线是 I/O 模式下或外围模式下。

如果提到 I/O 口，复位后 PIO 口线是带上拉电阻的输入，所以只要复位键释放，外设就保持在一个静止状态。结果，与寄存器 PIO_PSR(外围状态寄存器)中的 PIO 口线对应的位复位为低电平。

如果在“复位状态”栏中提到一个信号名称，PIO 口线被指定到此功能和 PIO_PSR 寄存器中对应的位复位为高电平。这是引脚控制存储器的情况，特别是地址块，只要复位键一释放就要驱动地址块引脚。注意在此情况下也要使能上拉电阻。

10.3.1. 资源多路复用

10.3.1.1 LCD 控制器

LCD 控制器可以和若干 LCD 控制面板对接。完全支持 4，8 或 16 位每像素。连接 24 位每像素的 TFT 面板需禁用 SSC0 和 SPI1 的片选块 0。

16 位每像素的 TFT 控制面板通过外设 B 功能实现对接，随色彩数据输出 LCDD3 到 LCDD7,LCDD11 到 LCDD15 和 LCDD19 到 LCDD23. 亮度位在 LCDD2,LCDD10 和 LCDD18 上输出。用外设 B 不能限制用 SSC0 和 SPI1 块。

10.3.1.2 ETM

用 ETM 阻止：

- 用 USART1 和 USART2 控制信号，特别是 SCK 口线。SCK 口线在串行上需要用 USART 像 ISO7816 和 RTS 和 CTS 来控制硬件握手。万一 ETM 和一个 ISO7816 连接都需要，USART0 只能用作智能卡接口。
- 使用 SSC1
- 寻址一个大于 8M 字节的静态存储器需要用 A23 和 A24 地址线。
- 用 SPI0 和 SPI1 的片选 1 到 3

10.3.1.3 EBI

如果不需要，NWAIT 功能(外部等待请求)能通过软件禁用，允许此引脚被用作一个 PIO。

10.3.1.4 32 位数据总线

用一个 32 位数据总线阻止：

- 使用 3 个定时器计数器通道的输出和处触发输入
- 使用 SSC2

10.3.1.5 NAND 闪存接口

用 NAND 闪存接口：

用 NCS3，NCS6 和 NCS7 访问其他的并行设备

10.3.1.6 CompactFlash 接口

用 CompactFlash 接口阻止：

用 NCS4 和/或 NCS5 访问其他的并行设备

10.3.1.7 SPI0 和 MultiMedia 卡接口

由于 DataFlash 卡和 SD 卡兼容，对多路复用 SPI 和 MCI 很有用。这里，SPI0 信号和 MCI 多路复用。

10.3.1.8 USARTs

- 使用 USART1 和 USART2 控制信号阻止使用 ETM。
- 另外，使用带其本身控制信号的 USART0 防止使用若干时钟输出和中断块。

10.3.1.9 时钟输出

- 使用时钟输出和 PIO A 多路复用来防止使用调试部件和/或双总线接口。
- 另外，使用时钟输出的辅助措施阻止使用 LCD 控制器接口和/或 USART0。

10.3.1.10 中断口线

- 使用 FIQ 阻止使用 USART0 控制信号。
- 使用 IRQ0 阻止使用 NWAIT EBI 信号。
- 使用 IRQ1 和/或 IRQ2 阻止使用 SPI1。

10.3.2 PIO 控制器 A 多路复用

表 10-2。PIO 控制器 A 的多路复用

PIO 控制器 A					应用情况		
I/O 口线	外设 A	外设 B	备注	复位状态	供电	功能	备注
PA0	SPI0_MISO	MCDA0		I/O	VDDIOP		
PA1	SPI0_MOSI	MCCDA		I/O	VDDIOP		
PA2	SPI0_SPCK	MCCK		I/O	VDDIOP		
PA3	SPI0_NPCS0			I/O	VDDIOP		
PA4	SPI0_NPCS1	MCDA1		I/O	VDDIOP		
PA5	SPI0_NPCS2	MCDA2		I/O	VDDIOP		
PA6	SPI0_NPCS3	MCDA3		I/O	VDDIOP		
PA7	TWD	PDK0		I/O	VDDIOP		
PA8	TWCK	PCK1		I/O	VDDIOP		
PA9	DRXD	PCK2		I/O	VDDIOP		
PA10	DTXD	PCK3		I/O	VDDIOP		
PA11	TSYNC	SCK1		I/O	VDDIOP		
PA12	TCLK	RTS1		I/O	VDDIOP		
PA13	TPS0	CTS1		I/O	VDDIOP		
PA14	TPS1	SCK2		I/O	VDDIOP		
PA15	TPS2	RTS2		I/O	VDDIOP		
PA16	TPK0	CTS2		I/O	VDDIOP		
PA17	TPK1	TF1		I/O	VDDIOP		
PA18	TPK2	TK1		I/O	VDDIOP		
PA19	TPK3	TD1		I/O	VDDIOP		
PA20	TPK4	RD1		I/O	VDDIOP		
PA21	TPK5	RK1		I/O	VDDIOP		
PA22	TPK6	RF1		I/O	VDDIOP		
PA23	TPK7	RTS0		I/O	VDDIOP		
PA24	TPK8	SPI1_NPCS1		I/O	VDDIOP		
PA25	TPK9	SPI1_NPCS2		I/O	VDDIOP		
PA26	TPK10	SPI1_NPCS3		I/O	VDDIOP		
PA27	TPK11	SPI0_NPCS1		I/O	VDDIOP		
PA28	TPK12	SPI0_NPCS2		I/O	VDDIOP		
PA29	TPK13	SPI0_NPCS3		I/O	VDDIOP		
PA30	TPK14	A23		A23	VDDIOM		
PA31	TPK15	A24		A24	VDDIOM		

10.3.3 PIO 控制器 B 多路复用

表 10-3 PIO 控制器 B 上的多路复用

PIO 控制器 B					应用情况		
I/O 块	外设 A	外设 B		复位状态	供电	功能	备注
PB0	LCDVSYNC			I/O	VDDIOP		
PB1	LCDHSYNC			I/O	VDDIOP		
PB2	LCDDOTCK	PCK0		I/O	VDDIOP		
PB3	LCDDEN			I/O	VDDIOP		
PB4	LCDCC	LCDD2		I/O	VDDIOP		
PB5	LCDD0	LCDD3		I/O	VDDIOP		
PB6	LCDD1	LCDD4		I/O	VDDIOP		
PB7	LCDD2	LCDD5		I/O	VDDIOP		
PB8	LCDD3	LCDD6		I/O	VDDIOP		
PB9	LCDD4	LCDD7		I/O	VDDIOP		
PB10	LCDD5	LCDD8		I/O	VDDIOP		
PB11	LCDD6	LCDD9		I/O	VDDIOP		
PB12	LCDD7	LCDD10		I/O	VDDIOP		
PB13	LCDD8	LCDD11		I/O	VDDIOP		
PB14	LCDD9	LCDD12		I/O	VDDIOP		
PB15	LCDD10	LCDD13		I/O	VDDIOP		
PB16	LCDD11	LCDD19		I/O	VDDIOP		
PB17	LCDD12	LCDD20		I/O	VDDIOP		
PB18	LCDD13	LCDD21		I/O	VDDIOP		
PB19	LCDD14	LCDD22		I/O	VDDIOP		
PB20	LCDD15	LCDD23		I/O	VDDIOP		
PB21	TF0	LCDD16		I/O	VDDIOP		
PB22	TK0	LCDD17		I/O	VDDIOP		
PB23	TD0	LCDD18		I/O	VDDIOP		
PB24	RD0	LCDD19		I/O	VDDIOP		
PB25	TK0	LCDD20		I/O	VDDIOP		
PB26	RF0	LCDD21		I/O	VDDIOP		
PB27	SPI1_NPCS1	LCDD22		I/O	VDDIOP		
PB28	SPI1_NPCS0	LCDD23		I/O	VDDIOP		
PB29	SPI1_SPCK	IRQ2		I/O	VDDIOP		
PB30	SPI1_MISO	IRQ1		I/O	VDDIOP		
PB31	SPI1_MOSI	PCK2		I/O	VDDIOP		

10.3.4 PIO 控制器 C 多路复用

表 10-4 PIO 控制器 C 上的多路复用

PIO 控制器 C					应用情况		
I/O 块	外设 A	外设 B	备注	复位状态	供电	功能	备注
PC0	NANDOE	NCS6		I/O	VDDIOM		
PC1	NANDWE	NCS7		I/O	VDDIOM		
PC2	NWAIT	IRQ0		I/O	VDDIOM		
PC3	A25/CFRNW			A25	VDDIOM		
PC4	NCS4/CFCS0			I/O	VDDIOM		
PC5	NCS5/CFCS1			I/O	VDDIOM		
PC6	CFCE1			I/O	VDDIOM		
PC7	CFCE2			I/O	VDDIOP		
PC8	TXD0	PCK2		I/O	VDDIOP		
PC9	RXD0	PCK3		I/O	VDDIOP		
PC10	RTS0	SCK0		I/O	VDDIOP		
PC11	CTS0	FIQ		I/O	VDDIOP		
PC12	TXD1	NCS6		I/O	VDDIOM		
PC13	RXD1	NCS7		I/O	VDDIOM		
PC14	TXD2	SPI1_NPCS2		I/O	VDDIOM		
PC15	RXD2	SPI1_NPCS3		I/O	VDDIOM		
PC16	D16	TCLK0		I/O	VDDIOM		
PC17	D17	TCLK1		I/O	VDDIOM		
PC18	D18	TCLK2		I/O	VDDIOM		
PC19	D19	TIOA0		I/O	VDDIOM		
PC20	D20	TIOB0		I/O	VDDIOM		
PC21	D21	TIOA1		I/O	VDDIOM		
PC22	D22	TIOB1		I/O	VDDIOM		
PC23	D23	TIOA2		I/O	VDDIOM		
PC24	D24	TIOB2		I/O	VDDIOM		
PC25	D25	TF2		I/O	VDDIOM		
PC26	D26	TK2		I/O	VDDIOM		
PC27	D27	TD2		I/O	VDDIOM		
PC28	D28	RD2		I/O	VDDIOM		
PC29	D29	RK2		I/O	VDDIOM		
PC30	D30	RF2		I/O	VDDIOM		
PC31	D31	PCK1		I/O	VDDIOM		

10.3.5 系统中断

系统中断源 1 是来源于以下中断信号的线或：

- SDRAM 控制器
- 调试部件
- 周期间断定时器
- 实时定时器
- 看门狗定时器
- 复位控制器
- 电源管理控制器

这些外设的时钟不能被撤销并且外设 ID1 仅能被用于先进中断控制器。

10.3.6 外部中断

所有的外部中断信号，比如快速中断型号 FIQ 或者中断信号 IRQ0-IRQ2，使用一个专门的外设 ID。但是并没有时钟控制和这些外设 ID 相关联。

10.4 外部总线接口

- 集成了 2 个外部存储器控制器
  - 静态存储器控制器
  - SDRAM 控制器
- 支持 NAND Flash 和 CompactFlash 的逻辑电路
  - NAND Flash 支持：支持 8 位，16 位设备
  - CompactFlash 支持：除了信号-IOIS16（I/O 和 True IDE 模式）和-ATA SEL（True IDE 模式）不可控制，支持所有的模式 (Attribute memory，普通存储器，I/O，True IDE)
- 优化的外部总线
  - 16 或 32 位数据总线
  - 多达 26 位的地址总线，达 64M 字节可寻址范围
  - 八个片选，每个保留给八个存储器区中之一
  - 优化引脚多路复用，减少外部存储器上延迟时间

• 位于用户接口矩阵中的 EBI_CSA 寄存器负责管理可配置的片选分配。

-NCS0 上的静态存储器控制器
-NCS1 上的 SDRAM 控制器或静态存储器控制器
-NCS2 上的静态存储器控制器
-NCS3 上的静态存储器控制器，可任选的 NAND Flash 后援
-NCS6-NCS7 上的静态存储器控制器

10.5 静态存储器控制器

- 外部存储器映射，每个片选线 256M 字节地址空间
- 多达八个片选线
- 8-，16-，或 32 位数据总线
- 支持多重访问模式

-字节写或字节选块
-支持在页面模式下异步读(4 到 32 字节页面大小)

- 多重设备自适应性
  -配合 LCD 模块使用
  -为每个存储器体控制信号可编程设置，脉冲和保持时间
- 多重等待状态管理
  -可编程等待状态发生
  -外部等待请求
  -可编程数据浮动时间
- 支持慢时钟模式

10.6 SDRAM 控制器

- 支持的设备
  -标准和低功率 SDRAM（移动 SDRAM）
- 支持多种配置
  -2K,4K,8K 行地址存储分配
  -带两个或四个内部 Bank 的 SDRAM
  -带 16 或 32 位数据宽度的 SDRAM
- 编程方式
  -字，半字，字节存取
  -当到达存储器边界时自动的页面切换
  -多存储体交替存取
  -通过软件设定时序参数
  -自动刷新操作，刷新率可编程
- 节能能力
  -自刷新，支持掉电和深度掉电模式
- 错误检测
  -刷新错误中断
- SDRAM 通过软件上电初始化
- 支持 1， 2，和 3CAS 延迟时间
- 未使用自动预充电命令

10.7 串行外设接口（SPI）

- 支持与多个串行外部设备通信
  -带外部解码器的四个片选允许和多达 15 个外设通信
  -串行存储器，像 DataFlash 和 3 线 EEPROMs
  -串行外设，像 ADCs,DACs,LCD 控制器，CAN 控制器和传感器
  -外部协处理器
- 主控或从控串行外设总线接口
  -每个片选 8-到 16-位可编程数据长度
  -每个片选可编程相位和优先级
  -每个片选在顺序传输间和在时钟和数据间可编程传输延迟
  -在顺序传输间可编程延迟
  -错误检测可选模式
- 支持超快速传输
  -传输波特率可达 MCK

-在同一个设备上片选口线可能被激活去加速传输

10.8 双线接口

- 标准的双总线串行存储器兼容能力
- 从控地址一，二或三字节
- 顺序读/写操作

10.9 USART

- 可编程的波特率发生器
- 5 到 9 位全双工同步或异步串行通信
  -在异步模式 1，1.5 或 2 个停止位或在同步模式 1 或 2 位停止位
  -奇偶校验位生成错误检测
  -成帧错误检测和超限错误检测
  -最高位在先或最低位在先
  -可选的中断生成和检测
  -8 或 16 倍过采样
  -硬件握手 RTS-CTS
  -接收器超时和发送器时间警戒
  -可任选的带地址生成和检测的多接入点模式
  -可任选的 Manchester 编码
- 带驱动控制信号的 RS485
- ISO7816，T=0 或 T=1 带智能卡的接口协议
  -NACK 控制，带重复和迭代限制的错误计数器
- IrDA 调制和解调
  -最高可达 115.2Kbps
- 测试模式
  -远程回送，本地回送，自动回显

10.10 同步串行控制器

- 提供串行同步通信连接用于音频和通信（带 CODECs 在主控或从控模式，IIS，TDM 总线，磁卡读卡器和更多）。
- 包括一个独立的接受器和发射器和一个公用时钟分频器
- 提供一个可配置的帧同步可数据长度
- 接收器和发射器可被编程用于自启动或在帧同步信号上检测不同的事件
- 接收器和发射器包含一个数据信号，一个时钟信号和一个帧同步化信号。

10.11 时器/计数器

- 3 个 16 位的定时器计数器通道
- 功能包括：
  -频率测量
  -事件计数
  -间隔测量
  -脉冲生成
  -延时定时
  -脉冲宽度调制
  -上/下计数能力

- 每个通道是用户可配置的并且包括：
  -三个外部时钟输入
  -五个内部时钟输入
  -两个多用途输入/输出信号
- 两个全局寄存器在所有的三个 TC 通道上运行

10.12 Multimedia 卡接口

- 和 MultiMedia 卡规范版本 2..2 兼容
- 和 SD 存储器卡规范版本 1.0 兼容
- 卡的时钟速率达到主控时钟的 2 分频
- 未用时嵌入式的电源管理器降低时钟速率
- 每个 MCI 有两个插槽，每个支持
  -用于一个 Multimedia 总线（多达 30 个卡）的一个插槽或
  -一个 SD 存储卡
- 支持流，块和多块数据读和写

10.13 USB

- USB 主机端口：
  -兼容 Open HCI Rev1.0 规范
  -兼容 USB V2.0 全速和低速规范
  -支持 1.5Mbps 低速和 12Mbps 高速 USB 设备
  -根集线器集成了两个下行数据流 USB 端口
  -两个嵌入式 USB 收发器
  -无过流检测
  -支持电源管理
  -在总线矩阵上作为一个主控操作
- USB 设备端口：
  -兼容 USB V2.0 全速，12M 位每秒
  -集成 USB V2.0 全速收发器
  -集成端点（endpoints）双端口 RAM
  -挂起/继续执行 逻辑
  -用于isochronous和bulk端点的ping-pong缓冲模式(两个存储体)
  -六个通用端点（endpoints）：
    端点 0：8 字节，无交替模式
    端点 1，端点 2：64 字节，交替模式
    端点 3：64 字节，无交替模式
    端点 4，端点 5：256 字节，交替模式
- 集成了 pad 上拉电阻，并可通过 MAXTRIX 用户机接口中的 USB_PUCR 寄存器进行配置

10.14 LCD 控制器

- 支持单，双扫描彩色和单色被动 STN LCD 控制面板
- 支持单扫描主动的 TFT LCD 控制面板
- 支持 4 位单扫描，8 位单或双扫描，16 位双扫描 STN 接口
- 支持多达 24 位的单扫描 TFT 接口
- 单色 STN 有多达 16 灰阶，彩色 STN 显示多达 4096 色
- 每像素（palletized）1，2 位。单色 STN 每像素（non-palletized）4

位

- 每像素（palletized）1，2，4，8 位，彩色 STN 每像素（non-palletized）16 位
- 每像素（palletized）1，2，4，8 位，TFT 每像素（non-palletized）16，24 位
- 单时钟域架构
- 分辨率支持达 2048*2048

11. ARM926EJ-S 处理器概论

11.1 概论

ARM926EJ-S 处理器是通用微处理器 ARM9 家族中的一员。
ARM926EJ-S 属于 5TEJ 版 ARM 架构，针对的是多任务应用，包括全储存器管理，高性能，小核心尺寸和低功耗都是其重要的特点。

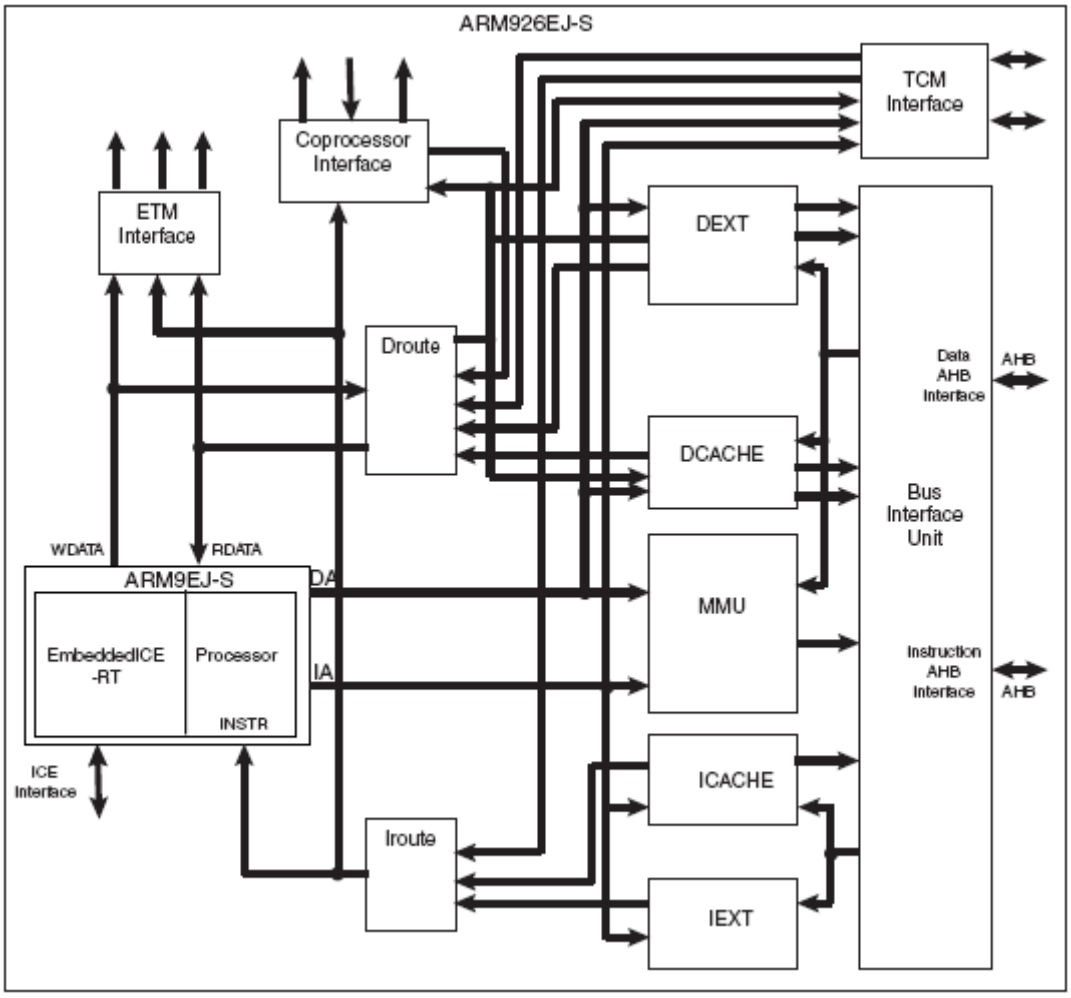
ARM926EJ-S 处理器支持 32 位 ARM 和 16 位 THUMB 指令集，使得用户能在高性能和高代码密度上取得平衡。支持 8 位 Java 指令集并且包括 Java 字节代码有效执行的功能部件，提供和 JIT（Just-In-Time 编译器）相似的 Java 性能，这些性能为下一代 Java 无线和嵌入式的设备提供了有力支持。为了提高 DSP 性能，还包含了一个增强的乘法器设计。

ARM926EJ-S 处理器支持 ARM 调试架构，包括对辅助硬件和软件调试的逻辑。
ARM926EJ-S 提供了一个完全高性能的处理器子系统，包括：

- ARM9EJ-S 整核
- 一个存储器管理部件（MMU）
- 独立的指令和数据 AMBA™ AHB 总线接口
- 独立的指令和数据 TCM 接口

11.2 方块图

图 11-1　　ARM926EJ-S 内部功能方块图



11.3 ARM9EJ-S 处理器

11.3.1 ARM9EJ-S 运行状态

ARM9EJ-S 处理器能用三种不同状态操作，每种状态带一个特殊指令集：

- ARM 状态：32 位，字对齐 ARM 指令
- THUMB 状态：16 位，半字对齐的 THUMB 指令
- Jazelle 状态：可变长度，字节对齐的 Jazelle 指令

在 Jazelle 状态，按字（words）取所有的指令。

11.3.2 状态切换

ARM9EJ-S 核的操作状态在以下状态间切换：

- 用 BX 和 BLX 指令切换 ARM 状态和 THUMB 状态，并加载到 PC
- 用 BXJ 指令切换 ARM 状态和 Jazelle 状态

所有的异常都在 ARM 状态下进入，处理和退出。如果一个异常发生在 Thumb 状态或 Jazelle 状态，处理器还原到 ARM 状态。从异常模式处理程序退出时自动切换回 Thumb 或 Jazelle 状态。

11.3.3 指令流水线

ARM9EJ-S 核使用两种流水线去增加传输至处理器的指令流的速度。

一个五级(五个时钟周期)流水线被用于 ARM 和 Thumb 状态。由取指，译码，执行，存储和回写阶段组成。

一个六级（六个时钟周期）流水线被用于 Jazelle 状态。由取指，Jazelle/译码(两时钟周期)，执行，存储和回写阶段。

11.3.4 存储器访问

ARM9EJ-S 核支持字节(8 位)，半字(16 位)和字(32 位)访问。字必须对齐到四字节边界，半字必须对齐到两字节边界而字节可以被放置在任何字节边界。

因为流水线的性质，一个当前操作需要的值在被放置到寄存器前可能被一个先前的指令操作(使得当前流水线预取的值失效，译者注)。ARM9EJ-S 的控制逻辑自动检测这些情况并停止内核或传输数据。

11.3.5 Jazelle 技术

在 ARM 处理器上，Jazelle 技术直接且有效的执行 Java 字节代码，为下一代 Java 无线应用和嵌入式设备提供高性能。

ARM9EJ-S 新的 Java 特性可以被描述为 JVM(Java 虚拟机)的一个硬件模拟。Java 模式将表现为另外一种状态：取代执行 ARM 或 Thumb 指令，而执行 Java 字节代码。实现于 ARM9EJ-S 上 Java 字节代码解码器逻辑实现了 95%可执行字节代码的解码，并且在无任何额外开销的情况下把它们转换为 ARM 指令，同时，较小频率使用的字节解码被分解成顺序优化的 ARM 指令。硬件/软件拆分对编程者，应用程序，以及操作系统都是不可见的。所有现存的 ARM 寄存器在 Jazelle 状态被重复使用，并且在此模式下所有的寄存器都有特殊的功能。

最小的中断延迟被维护于 ARM 状态和 Java 状态切换。所以字节代码执行可以被重新开启，以方便中断处理程序执行，一个中断自动触发其内核从 Java 状态到 ARM 状态改变。这意味着当执行字节代码时没有特殊的为控制中断而准备的规定，无论是硬件还是软件。

11.3.6 AEM9EJ-S 操作模式

在所有的状态下，有七种操作模式：

- 用户模式是一般的 ARM 程序执行状态。被用于执行大多数应用程序
- 快速中断（FIQ）模式被用于控制快速中断。适用于高速数据传输或通道进程
- 中断(IRQ)模式被用于一般目的中断的处理
- 管理模式是一种操作系统的保护模式
- 中止(abort)模式在一个数据或指令预取中止后进入
- 系统模式是一个操作系统的特权用户模式
- 未定义模式当一个未定义指令异常发生后进入

模式转换可在软件控制下发生，或可能被外部中断或异常处理所引发。

大多数的应用程序执行是在用户模式下进行的。非用户模式，又称特权模式，用于处理中断或异常，以及访问被保护的资源。

11.3.7 AEM9EJ-S 寄存器

AEM9EJ-S 内核共有 37 个寄存器

- 31 个通用的 32 位寄存器
- 6 个 32 位的状态寄存器

表 11-1　显示了所有模式下所有寄存器

表 11-1　ARM9TDMI 模式和寄存器层

用户和系统模式	管理模式	异常模式	未定义模式	中断模式	快速中断模式
R0	R0	R0	R0	R0	R0
R1	R1	R1	R1	R1	R1
R2	R2	R2	R2	R2	R2
R3	R3	R3	R3	R3	R3
R4	R4	R4	R4	R4	R4
R5	R5	R5	R5	R5	R5
R6	R6	R6	R6	R6	R6
R7	R7	R7	R7	R7	R7
R8	R8	R8	R8	R8	R8_FIQ
R9	R9	R9	R9	R9	R9_FIQ
R10	R10	R10	R10	R10	R10_FIQ
R11	R11	R11	R11	R11	R11_FIQ
R12	R12	R12	R12	R12	R12_FIQ
R13	R13_SVX	R13_ABORT	R13_UNDEF	R13_IRQ	R13_FIQ
R14	R14_SVC	R14_ABORT	R14_UNDEF	R14_IRQ	R14_FIQ
PC	PC	PC	PC	PC	PC
CPSR	CPSR	CPSR	CPSR	CPSR	CPSR
	SPSR_SVC	SPSR _ABORT	SPSR _UNDEF	SPSR _IRQ	SPSR _FIQ

灰色框为特定模式下的备份寄存器

ARM 状态寄存器集包括 16 个直接存取寄存器，r0 到 r15，和一个附加寄存器：当前程序状态寄存器(CPSR)。寄存器 r0 到 r13 是用于保存数据或地址值的通用寄存器。寄存器 r14 被用作一个连接寄存器(Link Register)，当 BL 或 BLX 被执行的时候，用来保存 r15 的值(返回地址)。寄存器 r15 被用作程序计数器(PC)，当前程序状态寄存器(CPSR)包含条件代码标志位和当前模式位。

在特权模式(FIQ，管理模式，中止模式，IRQ，未定义模式)，特定模式备份寄存器(FIQ 模式的 r8 到 r14 或其他模式的 r13 到 r14)可用。当中断或异常发生，或当 BL 或 BLX 指令在中断或异常例程的情况被执行，对应的备份寄存器 r14_FIQ，r14_SVC，r14_ABT，r14_IRQ，r14_UND 相似的被用于保存 r15(PC)的值(为每个模式的返回地址)。还有另外的一个叫保存程序状态寄存器（SPSR）的寄存器，在特权模式下代替 CPSR 变成可用。此寄存器保存进入到当前模式(特权模式)之前模式的代码标志位及模式位。

出于软件一致性，在所有的模式下，r13 被用作堆栈指针。

所有的上面描述的寄存器的功能和用途都应该服从 ARM Procedure Call Standard（APCS），包括：

- 寄存器用途的约束
- 堆栈约定

- 参数传递并返回结果的约定

Thumb 状态寄存器集是一个 ARM 状态集的一个子集。编程者可直接访问：

- 八个通用寄存器 R0-R7
- 堆栈指针,SP
- 连接寄存器，LR(ARM R14)
- PC
- CPSR

每个特权模式都有备份寄存器 SPs，LRs 和 SPSRs（更多的细节见 ARM9EJ-S 技术参考手册，ref.DDI0222B,revision r1p2 2-12 页）。

11.3.7.1 状态寄存器

ARM9EJ-S 内核包括一个 CPSR，和五个异常模式处理程序使用的 SPSRs。程序状态寄存器：

- 保存最近 ALU 操作运行的信息
- 使能和禁用中断
- 设置处理器操作模式

图 11-2 状态寄存器格式

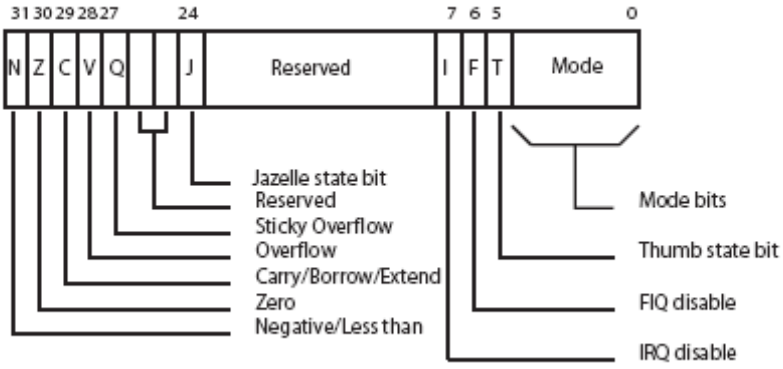


图 11-2 展示了状态寄存器格式：

- 四个 ALU 标志位 N：Negative(负)，Z：Zero（零），C：Carry（进位），V：Overflow (溢出)
- 黏着（sticky）溢出标志位(Q)可以被特定的乘法和小数运算指令像 QADD，QDADD，QSUB，QDSUB，SMLAxy，以及 AMLAWy 置位，完成需要的 DSP 操作。

Q 标志位是黏着的，意味着当其被一指令置位后，此标志位将一直保持置位，除非通过一个 MSR 指令写 CPSR 明确地清零。指令并不能依 Q 标志位的状态按条件执行。

- CPSR 中的 J 位表示 ARM9EJ-S 内核在 Jazelle 状态：
  -J=0：处理器在 ARM 或 Thumb 状态，取决于 T 位
  -J=1：处理器在 Jazelle 状态
- 模式：五位编码用于确定当前处理器模式

11.3.7.2 异常

异常类型和优先级

ARM9EJ-S 支持五种异常。每种类型在一个特权模式下驱动 ARM9EJ-S。

异常的类型有：

- 快速中断（FIQ）
- 一般中断（IRQ）
- 数据和预取指中止（Abort）
- 未定义指令（未定义）
- 软件中断和复位（管理模式）

当一个异常发生，R14 的备份寄存器和异常模式下的 SPSR 用来保存状态。

在同一时间可以发生多于一个的异常。因此 ARM9EJ-S 根据以下的优先级运行引发的异常：

- 复位（最高优先级）
- 数据中止
- FIQ
- IRQ
- 预取指中止
- BKPT，未定义的指令，和软件中断(SWI)(最低优先级)

BKPT，或未定义指令，和 SWI 异常是互斥的。

在优先级序列中有一个例外，当 FIQ 使能，并且发生数据中止异常时一个 FIQ 也同时发生，ARM9EJ-S 内核进入数据中断（Data Aborts）处理程序，然后立即进入 FIQ 向量。一个 FIQ 的一般中断返回导致(先前的)数据中断（Data Aborts）处理继续执行。数据中断（Data Aborts）必须是比 FIQ 有较高优先级以确保传输错误不会漏过检测。

异常模式和控制

无论何时，当异常发生，一个程序的正常流程必须暂时停止，例如，处理一个外设中断。

当处理一个 ARM 的异常时，ARM9EJ-S 内核执行以下操作：

1 保存下一条指令的地址到进入的新模式所对应的 Link 寄存器

当异常从以下状态进入：

- ARM 和 Jazelle 状态，ARM9EJ-S 把下一个指令的地址复制到 LR(根据异常，当前 PC(r15)+4 或 PC+8)
- THUMB 状态，ARM9EJ-S 写 PC 的值到 LR，偏移一个值（根据异常，当前 PC+2,PC+4 或 PC+8），使得程序能从正确地址返回重开始的值。

2 复制 CPSR 到相应的 SPSR

3 根据异常状态强制设置 CPSR 相关模式位

4 强制 PC 从相关的异常向量获取下一个指令

寄存器 r13 也是备份的，通过异常模式向每个异常处理提供专用堆栈指针。

ARM9EJ-S 也可以置位中断禁止标志位，以阻止不可管理的异常嵌套。

当一个异常处理完成，异常处理必须把备份 LR 的值减去一个偏移量的后移到 PC，把 SPSR 复制到 CPSR。偏移量根据异常类型变化。此操作恢复 PC 和 CPSR。

快速中断模式有七个专用寄存器 r8-r14(备份寄存器)，可以减少甚至不需要对寄存器的保存操作，以降低上下文切换的额外消耗。

预取指中止是中止模式之一，表示当前存储器访问不能完成。当一个预取指中止发生，ARM9EJ-S 标记预取指令为无效，但指令在流水线中未到达执行阶段则不执行例外异常模式。如果指令没有执行，例如由于一个分支在流水线中发生，则中止不发生。

断点指令（BKPT）是 ARM9EJ-S 的一个新的特性，被设计用于解决预取指中止的问题。一个断点指令像导致一个预取指中止的指令一样执行。
断点指令不会使 ARM9EJ-S 去执行预取指中止异常模式，直到指令到达流水线的执行阶段。如果指令未执行，例如因为一个分支在流水线中发生，断点不再发生。

11.3.8 ARM 指令集一览

ARM 指令集被分成：

- 分支（Branch）指令
- 数据处理指令
- 状态寄存器传输指令
- 装载和存储指令
- 协处理器指令
- 异常产生（Exception-generating）指令

ARM 指令可以被有条件的执行。每个指令包含一个 4 位条件代码字段(位[31：38])。

表 11-2 给出了 ARM 指令助记符列表。

表 11-2. ARM 指令助记符列表

助记符	操作
MOV	Move 传送
ADD	Add 加
SUB	Subtract 减
RSB	Reverse Subtract 逆减
CMP	Compare 比较
TST	Test 测试
AND	逻辑与
EOR	逻辑异或
MUL	乘
SMULL	带符号长乘
SMLAL	带符号长乘累加
MSR	传送到状态寄存器
B	分支
BX	分支并交换
LDR	装载字
LDRSH	装载带符号半字

LDRSB	装载带符号字节
LDRH	装载半字
LDRB	装载字节
LDRBT	装载带转换的寄存器字节
LDRT	装载带转换的寄存器
LDM	批量装载
SWP	字交换
MCR	传送至协处理器
LDC	装载至协处理器
CDP	协处理器数据处理

助记符	操作
MVN	取反传送
ADC	带进位加
SBC	带借位减
RSC	带借位逆减
CMN	取反比较
TEQ	测试等价
BIC	位清零
ORR	逻辑或
MLA	乘累加
UMULL	无符号长乘
UMULAL	无符号长乘累加
MRS	从状态寄存器传送
BL	分支并链接
SWI	软件中断
STR	存储字
STRH	存储半字
STRB	存储字节
STRBT	存储带转换的寄存器字节
STRT	存储带转换的寄存器
STM	批量存储
SWPB	交换字节
MRC	从协处理器传送
STC	从协处理器存储

11.3.9 新 ARM 指令集

表 11-3 新 ARM 指令助记符列表

助记符	操作
BXJ	分支并切换至 Java

BLX	分支，连接并切换
SMLAxy	带符号乘累加 16*16 位
SMLAL	带符号长型乘累加
SMLAWy	带符号乘累加 32*16 位
SMULxy	带符号乘 16*16 位
SMULWy	带符号乘 32*16 位
QADD	饱和加
QDADD	饱和双精度加
QSUB	饱和减
QDSUB	饱和双精度减

助记符	操作
MRRC	从协处理器传送双字
MCR2	ARM 寄存器到协处理器的选择传送
MCRR	传送双字到协处理器
CDP2	选择协处理器数据处理
BKPT	断点
PLD	软预装载，存储器从地址装载
STRD	存储双字
STC2	从协处理器选择存储
LDRD	装载双字
LDC2	选择装载到协处理器
CLZ	前导零计数

注意：一个 Thumb BLX 包含两个连续的 Thumb 指令，并需四个时钟周期。

11.3.10 Thumb 指令集概论

Thumb 指令集是 ARM 指令集的一个重编码子集。

Thumb 指令集被分成：

- 分支指令
- 数据处理指令
- 装载和存储指令
- 批量装载和批量存储指令
- 异常产生指令

表 11-4 给出了 Thumb 指令助记符列表。

表 11-4 Thumb 指令助记符列表

助记符	操作
MOV	传送
ADD	加
SUB	减
CMP	比较
TST	测试
AND	逻辑与
EOR	逻辑异或
LSL	逻辑左移

ASR	算数右移
MUL	乘
B	分支
BX	分支并切换
LDR	装载字
LDRH	装载半字
LDRB	装载字节
LDRSH	装载带符号半字
LDMIA	批量装载
PUSH	寄存器入栈
BCC	条件分支
MVN	取反传送
ADC	带进位加
SBC	带借位减
CMN	取反比较
NEG	取反
BIC	位清零
ORR	逻辑或
LSR	逻辑右移
ROR	循环右移
BLX	分支，连接，切换
BL	分支并连接
SWI	软件中断
STR	存储字
STRH	存储半字
STRB	存储字节
LDRSB	装载带符号字节
STMIA	批量存储
POP	出栈到寄存器
BKPT	断点

11.4 CP15 协处理器

协处理器 15，或系统控制协处理器 CP15，被用于配置和控制下表中所有项：

- ARM9EJ-S
- 高速缓冲（ICache,DCache 和写缓冲器）
- TCM
- MMU
- 其他系统选项

为了控制这些功能部件，CP15 提供了 16 个附加寄存器。见表 11-5

表 11-5　　CP15 寄存器

寄存器	名称	读/写
0	ID 代码*	读/不可预测

0	Cache 类型*	读/不可预测
0	TCM 状态*	读/不可预测
1	控制	读/写
2	转换表基址	读/写
3	域访问控制	读/写
4	保留	无
5	数据错误状态*	读/写
5	错误地址*	读/写
7	Cache 操作	读/写
8	TLB 操作	不可预测/写
9	Cache 锁定+	读/写
9	TCM 区域	读/写
10	TLB 锁定	读/写
11	保留	无
12	保留	无
13	FCSE PID*	读/写
13	上下文 ID*	读/写
14	保留	无
15	测试配置	读/写

注意：*寄存器位置 0，5 和 13 每个提供访问多个寄存器的能力。
被访问的寄存器取决于操作码 2 字段的值。
+寄存器位置 9 提供访问多个寄存器的能力。访问的寄存器取决于 CRm 字段的值。

11.4.1 CP15 寄存器访问

CP15 寄存器仅可以在特权模式以下列方式访问：

• MCR(从 ARM 寄存器传送到协处理器)指令被用于写一个 ARM 寄存器到 CP15.

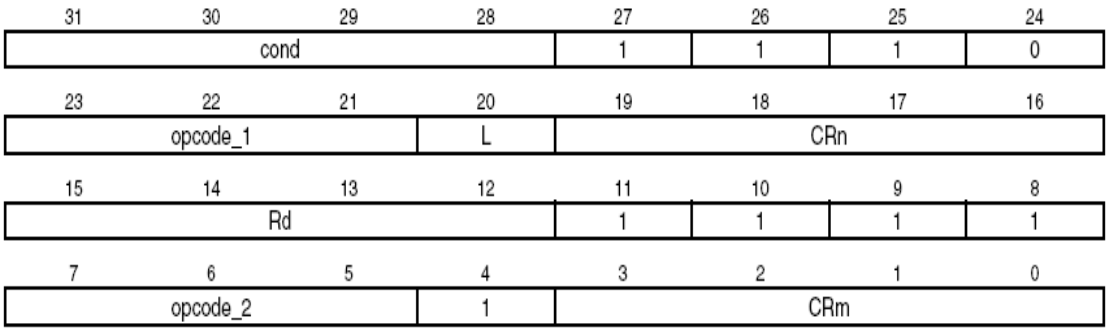
• MRC(从协处理器传送到 ARM 寄存器)指令被用于读 CP15 的值到一个 ARM 寄存器。

其他指令像 CDP,LDC,STC 可能导致一个未定义指令异常。

这些指令的汇编程序代码是：

MCR/MRC{cond} p15,opcode_1,Rd,CRn,CRm,opcode_2.

MCR,MRC 指令位模式如下所示：



• CRm[3：0]：指定协处理器操作

决定特定的协处理器操作。它的值取决于使用的 CP15 寄存器。详情请参考 CP15 特定寄存器行为。

- 操作码 2 [7：5]

决定特定协处理器操作码。缺省设置为零。

- RD[15：12]：ARM 寄存器

定义值将被传送至协处理器的 ARM 寄存器。如果选择 R15，结果是不可预测的。

- CRn[19：16]：协处理器

决定目标协处理器寄存器。

- L：指令位

0=MCR 指令

1=MRC 指令

- 操作码段 1[23：20]：协处理器代码

定义协处理器特定代码。其值对 CP15 是 c15。

- Cond[31: 28]: 条件

更多的细节，见 ARM926EJ-S　TRM 第二章，ref.DDI0198B.

11.5　存储器管理部件（MMU）

ARM926EJ-S 处理器实现了一个增强的 ARM V5 架构的 MMU，以向操作系统 Symbian OS,Windows CE，和 Linux 等提供其所需的虚拟存储特性。这些虚拟存储功能特性是存储器访问权限控制和虚拟地址到物理地址转换。

由 CPU 内核产生的虚拟地址被 FCSE(快速上下文变换扩展)用 CP15 寄存器 13 的值变换到一个修改了的虚拟地址（MVA）。MMU 用一个存储于物理存储器的单一，两级页表集转换 MVA 到物理地址。页表集中的每个入口包含访问权限和与虚拟地址对应的物理地址。

一级转换表包含 4096 个入口，与 MVA 的位[31：20]对应。这些入口包含一个指向含有相同属性信息（访问权限，域等）的 1MB 物理存储扇区的指针，或二级转换表一个入口的指针：粗页表或者细页表。

二级转换表包含两个子表，粗页表和细页表。粗页表里的一个入口包含一个指针，指向相同访问权限的大页和小页。细页表中的一个入口包含一个到大，小和极小页的指针。

表 11-6 展示了物理内存中每页的不同属性。

表 11-6 映射详述

映射名称	映射大小	访问允许	子页大小
段	1MB	段	-
大页	64KB	4 个分离的子页	16KB
小页	4KB	4 个分离的子页	1KB
极小页	1KB	极小页	-

MMU 包括：

- 访问控制逻辑
- 转换旁视缓冲区（TLB，Translation Look-aside Buffer）

- 转换表遍历硬件

11.5.1 访问控制逻辑

访问控制逻辑控制转换表中的每个入口的访问(权限)信息。

访问控制逻辑检查两种访问信息：域和访问许可。域（domain）是用于一个存储区域的访问控制的主要机制；共有 16 个域。它定义了操作能够进行的必要的条件。域决定了访问是否被允许还是被忽略。

二级访问控制机制是被为段，大页，小页和极小页定义的访问控制。段和极小页面有一个单一访问允许集，但大页和小页可以和 4 个访问允许集关联，每个子页(四分之一页)一个访问允许集。

11.5.2 转换旁视缓冲区(TLB)

转换旁视缓冲区(TLB)缓存已转换的表项从而避免每次都转换。当 TLB 包含一个 MVA(修改了的虚拟地址)入口，访问控制逻辑决定访问是否被允许并且输出对应于 MVA 的物理地址。如果访问被禁止，MMU 发信号给 CPU 内核来中止。

如果 TLB 不包含 MVA 的一个入口，转换表遍历硬件被引入，从物理内存中的转换表中检索转换信息。

11.5.3 转换表遍历硬件

转换表遍历硬件是一个逻辑，此逻辑遍历存储于物理内存中的转换表，得到物理地址和访问许可并且更新 TLB。

硬件遍历中的阶段数可以是一或二，取决于地址是否被标记为一个段映射访问或一个页面映射访问。

有三种容量的页面映射访问，一种容量的段映射访问。页面映射访问用于大页，小页和极小页。转换过程总是以一级读取启动。一个段映射访问仅需要一级读取，但一个页面映射访问需要一个额外的二级读取。MMU 的更多细节，请参考 ARM926EJ-S 参考手册第三章，ref.DDI0198B。

11.5.4 MMU 错误

MMU 在以下几种错误时产生一个中止：

- 对齐错误（仅对数据访问）
- 转换错误
- 域错误
- 权限错误

MMU 的访问控制机制检测发生这些错误的条件。如果错误是存储器访问引起的，MMU 中止访问并发错误信号给 CPU 内核。MMU 保存数据访问错误所产生的状态和错误地址信息，并将这些信息保存到数据错误状态寄存器和错误地址寄存器中。MMU 也保存由取指产生的错误信息到指令错误状态寄存器中。

当错误发生，错误状态寄存器（CP15 中的寄存器 r5）指示数据或预取指中止的原因，和中止的访问域数。错误地址寄存器（CP15 中寄存器 r6）保存与导致数据中止的访问相关联的 MVA。MMU 的更多细节，请参考 ARM926EJ-S 参考手册

第三章，ref.DDI0198B。

11.6 高速缓存和写缓冲

ARM926EJ-S 包含一个 16KB 的指令缓存(ICache)，一个 16KB 数据缓存（DCache），和一个写缓冲。虽然 ICache 和 DCache 拥有共同的特性，每种仍然有若干特定的机制。

高速缓存（ICache和DCache）是四路组相联的(four-way set associative), addressed, indexed and tagged using the Modified Virtual Address (MVA),带有一个8字长的cache line，DCache有两个dirty位。ICache和DCache提供了缓存锁定，缓存污染控制和line替换机制。

ARM926EJ-S 缓存支持一种新特性，叫做 allocate on read-miss(读取未命中), 也就是大家所共知的 wrapping。此功能特性使缓存能首先执行关键字缓存再充填。这就是说当对一个字的请求导致一个读取未命中(read-miss)时，则缓存执行一个 AHB 访问。缓存不是装载整个 line（8 words），而是首先装载关键字，所以处理器可以快速的读取，然后是剩余的字，而不管字被分配在 cache line 的哪个位置。

高速缓存和写缓冲被 CP15 寄存器 r1（控制），CP15 寄存器 r7（缓存操作）和 CP15 寄存器 r9(缓存锁定) 所控制。

11.6.1 指令缓存（ICache）

ICcahe 缓存取出来的指令，等待处理器执行。ICcache 可以通过写 1 到 CP15 寄存器 r1 的 I 位来使能，写 0 到该位来禁用。

当 MMU 被使能，所有的取指都取决于转换和权限检查。如果 MMU 被禁用，所有的取指是可缓存的，不做任何保护检查并且物理地址被平面映射(flat-ampped)到 MVA(即一一对应，不存在转换，译者注)。随着 MVA 被禁用，上下文切换将引发 ICache 清除和/或使无效。

当 ICache 被禁用，所有的取指直接出现在外部存储器（AHB）(见 ARM926EJ-S TRM 中 4-4 页表 4-1 和 4-2，ref.DDI0198B)。

复位时，ICache 条目是无效的并且 ICache 是禁用的。为了获得最佳性能，ICache 应该在复位后尽可能快地使能。

11.6.2 数据缓存（DCache）和写缓冲

ARM926EJ-S 包含一个 DCache 和一个写缓冲，以降低主内存带宽和延迟对数据访问性能上的影响。DCache 和写缓冲的操作是紧密相连的。

11.6.2.1 DCache

DCache 需要由 MMU 来使能。所有的数据访问取决于 MMU 权限和转换检查。被 MMU 中止的数据访问不会导致 line 填充或数据访问出现在 AMBA AHB 接口。如果 MMU 被禁用，所有的数据访问是不可缓存的，不可缓冲的，没有保护检查，并出现在 AHB 总线。所有的地址是平面映射(flat-ampped)的，VA=MVA=PA,每次上下文切换发生将引起 DCache 清除和/或使无效。

DCache 保存每条 line 被装载时的源物理地址标记(PA Tag)，并使用该 Tag 将修改后的 line 回写到外部存储器。这就是说回写操作无需 MMU 干预。

DCache 中的每条 line（8 words）有两个 dirty 标志位，一个用于首 4 words，另一个用于次 4 words。这些位，如果置位，标记关联的半条 line 为 dirty。如果 cache

line 由于一个 line 填充或一个缓存清除操作被代替，则 dirty 标志位被用于决定是否所有的，一半或无需将 cache line 回写到存储器。

DCache 可以通过写 1 或 0 到 CP15 寄存器 r1 中的位 C（见 ARM926EJ-S TRM 中 4-5 页表 4-3 和 4-4，ref.DDI0222B）来使能或禁用.

DCache 支持写通(write-though)或者写回(write-back)缓存操作，具体操作模式由存储器区域的 MMU 转换表中的 C 位和 B 位所选择。

DCache包含一个8 data word入口，单地址入口写回缓冲，用于在cache line eviction或dirty line清除时保存写回的数据。

写缓冲可以保存 16 字数据和 4 个独立的地址。DCache 和写缓冲操作是紧密相连的，它们的配置被 MMU 转换表中的页面描述符在每个段设置。

11.6.2.2 写缓冲

ARM926EJ-S 包含一个写缓冲，此写缓冲有一个 16 字数据缓冲和一个四地址缓冲。写缓冲适用于对某个可缓冲区域进行的所有写操作，包括写通区域和写回区域。写缓冲还可以避免处理器写到外部存储器时发生 stall。当一个存储产生，数据以内核速度（高速）被写到写缓冲。写缓冲接着以总线速度(比内核速度慢)完成数据存储到外部存储器的过程。在此期间，ARM9EJ-S 处理器可以执行其他任务。

DCache 和写缓冲支持写回和写通存储器区域，由 MMU 转换表的中的段和页面描述符中的 C 位，B 位控制。

写通操作

当一个缓存写命中(hit)，Dcache 被更新。更新后的数据被写入写缓冲，后者将其写入外部存储器。

当一个缓存写未命中(miss)，被轮换(round robin)或其他算法所选的一条 line 被存储在写缓冲器，写缓冲器传输其内容到外部存储器。

写回操作

当一个缓存写命中，缓存 line 或半条 line 被标记为 dirty，就是说缓存的内容没有被更新到对应的外部存储器中。

当一个缓存写未命中，被轮换(round robin)或其他算法所选的一条 line 被存储在写缓冲器，写缓冲器传输其内容到外部存储器。

11.7 紧耦合存储器接口(TCM)

11.7.1 TCM 描述

ARM926EJ-S 处理器有一个紧耦合存储器接口，该接口使得处理器能直接访问独立的指令和数据 TCM(ITCM 和 DTCM)。TCM 被用于存储实时性和性能要求极高的代码，它还提供一个 DMA 支持机制。不像 AHB 访问外部存储器，访问 TCM 是快速，确定的，不造成总线的负荷。

用户可以在以下范围内独立地配置每个 TCM 的容量：[0KB 到 64KB]的 ITCM 和 [0KB 到 64KB]的 DTCM。

TCM 可以使用两种方法配置： HMATRIX TCM 寄存器和 CP15 中的 TCM 区域(region)寄存器（寄存器 r9），两步都应该被执行。HMATRIX TCM 寄存器设置 TCM 容量而 CP15 中的 TCM 区域寄存器（寄存器 r9）映射 TCM 并使能 TCM。

ARM9EJ-S 内核的数据域(data side)可以访问 ITCM。为了能将代码载入 ITCM，这是必要的，比如 SWI 和仿真指令处理程序，以及访问基于 PC 的文字池。

11.7.2 使能和禁用 TCM

在任何使能操作之前，用户应该配置 HMATRIX TCM 寄存器中的 TCM 容量。

接着由 CP15 中的 TCM 区域寄存器（寄存器 r9）使能 TCM。用户应该使用由 HMATRIX TCM 寄存器所确定的容量。更多的细节和编程技巧，请参考 ARM926EJ-S 中的第 2.3 章，ref.DDI0222B.

11.7.3 TCM 映射

TCM 可以被分配到存储器映射中的任何地方，使用一个单独的区域作为 ITCM 而使用另一个独立的区域作为 DTCM。TCM 被设置于物理地址(TCM is physically addressed)并能被放置在物理地址空间的任何地方。然而，一个 TCM 的基地址必须和其容量对齐(align)，并且 DTCM 和 ITCM 区必须无重叠。TCM 映射通过 CP15 中的 TCM 区寄存器(r9)来执行。用户应该给 TCM 输入正确的映射地址。

11.8 总线接口部件

ARM926EJ-S 以一个总线接口部件（BIU）来仲裁和调度 AHB 请求。BIU 实现一个多层 AHB，基于 AHB-Lite 协议，此协议使能一个系统中多种 AHB 主控和从设备之间的并行访问路径。这通过用一个更复杂的互联矩阵来实现，它增加了总线带宽，提供了一个更灵活的系统架构。

多主控总线架构有许多好处：

- 它使得开发一个增加了总线带宽，拥有灵活架构的多主控系统得以实现。
- 每个 AHB 层变得简单，因为只有一个主控，无需仲裁或主控到从设置的多路选择。实现 AHB-Lite 协议的 AHB 层，不必支持请求和应答，也不必支持重试和拆分事务(split transactions)。
- 当多个主控同时想要访问相同的从设备时，仲裁变得有效。

11.8.1 支持的传送

ARM926EJ-S 处理器把所有的 AHB 访问当作单字，4 字突发或 8 字突发执行，任何大小不是 1，4，8 字的 ARM9EJ-S 内核请求被分成 1，4，8 字大小的包。**注意**：Atmel 总线和 AHB-Lite 协议兼容，因此，不支持拆分和重试请求。

表 8 给出了支持的传输方式和使用到他们的不同事务。

表 11-7 支持传送

HBurst[2:0]	描述	
SINGLE	单传送	字，半字，或字节的单传送 • 数据写（NCNB,NCB,WT,或 DCache 中失败的 WB） • 数据读（NCNB,或 NCB） • NC 取指（预取指和非预取指） • 页表遍历读
INCR4	4 字增长突发	若使能，half-line 缓存写回，预取指。4 字突发 NCNB，NCB,WT,或 WB 写
INCR8	8 字增长突发	Full-line 缓存写返回，8 字突发 NCNB，NCB,WT,或 WB 写

WRAP8	8 字 wrapping 突发	Cache 填充

11.8.2 Thumb 取指

所有取指，无论 ARM9EJ-S 内核的状态如何，都被处理为在 AHB 上的 32 位访问。如果 ARM9EJ-S 运行在 Thumb 状态，则两条指令可以一次被读取。

11.8.3 地址对齐

ARM926EJ-S BIU 执行地址对齐检查并对齐 AHB 地址到必要的边界。16 位访问对齐到半字边界，32 位访问对齐到字边界。

12. AT91SAM9261 调试和测试

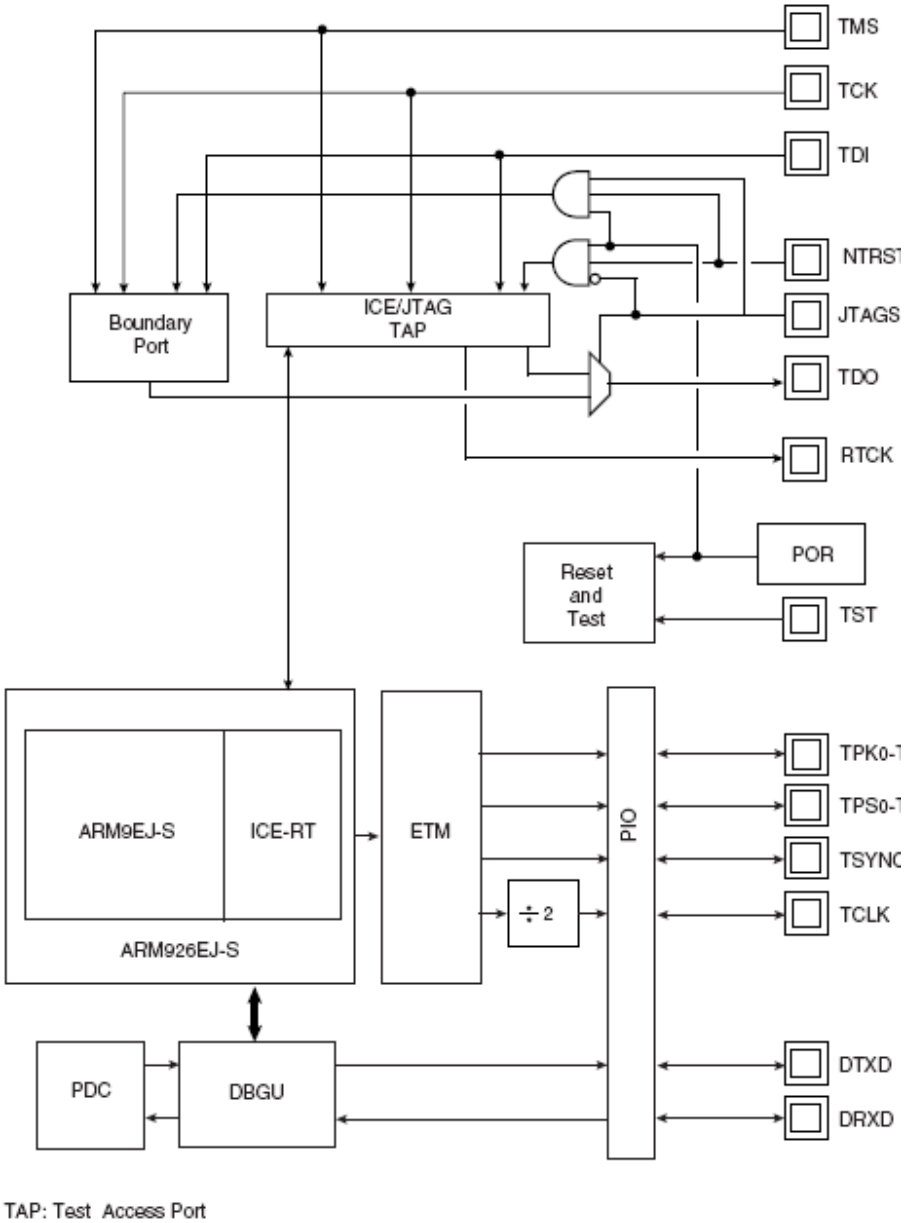
12.1 概述

AT91SAM9261 有许多补充的调试和测试能力。一个通用的 JTAG/ICE (In-Circuit Emulator)端口被用来进行标准的调试，像下载代码和单步执行程序。ETM(Embedded Trace Macrocell:嵌入式跟踪宏单元)提供更多的成熟的调试特性，象地址和数据比较器，半速率时钟模式，计数器，sequencer 和 FIFO。调试部件提供一个双引脚 UART，可以用于上传一个应用程序到内部 SRAM。它管理内部 COMMTX 和 COMMRX 信号的中断处理，这些信号跟踪调试通信通道的行为。

一套调试和测试专用的输入/输出引脚使得基于 PC 的测试环境可以直接访问以上这些功能。

12.2 方块图

图 12-1 调试和测试 方块图



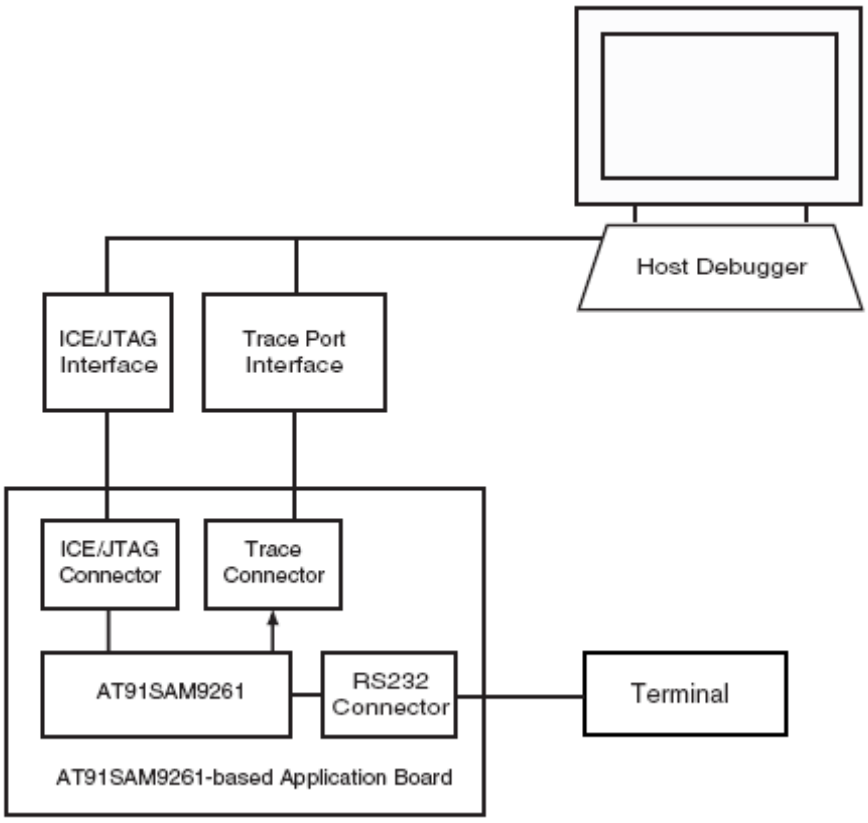
TAP: Test Access Port

12.3 应用举例

12.3.1 调试环境

62 页图 12-3 展示了一个完全的调试环境的例子。ICE/JTAG 接口被用作标准调试功能，下载代码和单步执行程序。跟踪端口(Trace Port)接口被用作跟踪信息。一个运行在个人计算机上的软件调试器利用 ICE/JTAG 接口提供一个配置跟踪端口接口的用户接口。

图 12-2 应用程序调试和测试环境举例

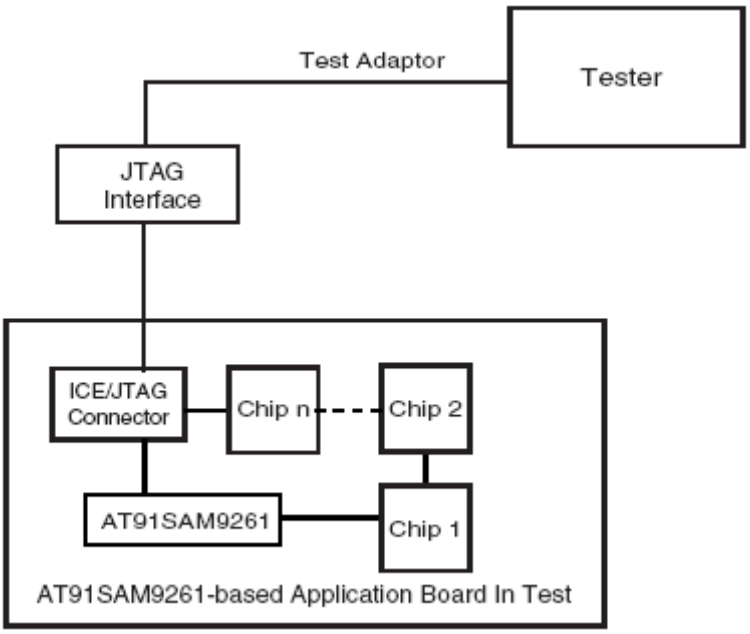


12.3.2 测试环境

图 12-3 展示了一个测试环境例子。测试向量被测试器发送并解释。

在此例中，‘测试板’使用许多 JTAG 兼容设备设计。这些设备可以被连接形成一个扫描链。

图 12-3 应用程序测试环境举例



12.4 调试和测试引脚描述

表 12-1 调试和测试引脚列表

引脚名称	功能	类型	有效电平
复位/测试			
NRST	微控制器复位	输入/输出	低
TST	测试模式选择	输入	高
ICE 和 JTAG			
TCK	测试时钟	输入	
TDI	测试数据输入	输入	
TDO	测试数据输出	输出	
TMS	测试模式选择	输入	
NTRST	测试模式复位信号	输入	低
RTCK	返回测试时钟	输出	
JTAGSEL	JTAG 选择	输入	
ETM			
TSYNC	跟踪同步信号	输出	
TCLK	跟踪时钟	输出	
TPS0-TPS2	跟踪 ARM 管道状态	输出	
TPK0-TPK15	跟踪包端口	输出	
调试部件			
DRXD	调试接收数据	输入	
DTXD	调试传输数据	输出	

12.5 功能描述

12.5.1 测试引脚

一个专用引脚，TST，被用于定义设备操作模式。用户必须确保此引脚拉至低电平以确保正常操作。和此引脚关联的其他值为生产测试预留。

12.5.2 嵌入式电路内部仿真器(Embedded In-circuit Emulator)

ARM9EJ-S 嵌入式 ICE-RT™ 通过 ICE/JTAG 端口支持。它由一个 ICE 接口连接到一个计算机主机。调试支持由一个内嵌在 ARM926EJ-S 中的 ARM9EJ-S 内核实现。ARM926EJ-S 的内部状态由一个 ICE/JTAG 端口检查，此端口允许指令在不通过外部数据总线的情况下被串行插入内核的流水线。因此，当在调试状态，一个批量存储（STM）可以被插入到指令流水线，这将可以导出 ARM9EJ-S 寄存器的内容。此数据可以在不影响系统其余部分的情况下被串行移出。

ARM9EJ-S 处理器内部有两条扫描链，扫描链支持测试，调试，和 EmbeddedICE-RT 的编程。此扫描链被 ICE/JTAG 控制。

当 JTAGSEL 为低电平时，EmbeddedICE 模式被选择。直接在 ICE 与 JTAG 操作之间转换是不能实现的。在 JTAGSEL 电平改变后必须进行芯片复位。

关于 EmbeddedICE-RT 更多的细节，参见 ARM 文件 ARM9EJ-S 技术参考手册（DDI0222A）。

12.5.3 JTAG 信号描述

TMS 是测试模式输入，此输入控制测试接口状态机器的转换。

TDI 是测试数据输入，此输入提供数据到 JTAG 寄存器（边界扫描寄存器，指令寄存器，或其他数据寄存器）。

TDO 是测试数据输出，此输出被用作从 JTAG 寄存器到控制测试设备串行的输出数据。它从边界扫描链（或其他 JTAG 寄存器）传送采样值并传送采样值到串行测试电路中的下一个芯片。

NTRST（在 IEEE 标准 1149.1 中为可选信号）是一个测试复位输入，此输入在 ARM 内核是强制的并且用于复位调试逻辑。在 Atmel 基于 ARM926EJ-S 的内核中，NTRST 是一个上电复位输出。上电时有效。如果必要，用户还可以保持 NTRST 有效 2.5 个 MCK 周期来复位调试逻辑。

TCK 是测试时钟输入，此输入使能测试接口。TCK 由控制测试的设备而不是被测试设备产生。它可以使用任何频率脉冲。**注意，**ARM926EJ-S 内核上最大的 JTAG 时钟速率是 CPU 时钟的六分之一。在一个从 32.768kHz 慢时钟运行的 ARM9E 上，最大的初始 JTAG 时钟速率是 5.45kHz。

RTCK 是一个返回测试时钟。不是一个 IEEE 标准 1149.1 中的信号，这个信号被引入是为了使仿真器能更好的处理时钟。从一些 ICE 接口探测，此返回信号可以被用于同步 TCK 时钟而不用担心 ICE 接口时钟和系统时钟之间需要遵守给定的六分之一比率。此信号仅在 JTAG ICE 模式下可用，在边界扫描模式下不可用。

12.5.4 调试部件

调试部件提供一个双引脚(DXRD 和 TXRD) USART。此 USART 可以被用于一些调试和跟踪目的，并提供一个现场(in-site)编程和调试监视通信的理想方案。此外，两个与通道关联的外设 DMA 控制器使得这些任务包的处理时间减到最小。

调试部件还管理 COMMTX 和 COMMRX 信号的中断处理，这些信号来于 ICE 并用于跟踪调试通信通道的活动。调试部件可以阻止通过 ICE 接口来访问系统。

一个特定的寄存器，调试部件芯片 ID 寄存器，给出了产品版本和其内部结构的信息。

AT91SAM9261 调试部件芯片 ID 是一个 32 位宽度的值：0x0197 03A0。

12.5.5 嵌入式跟踪宏单元

AT91SAM926 有一个嵌入式跟踪宏单元(ETM)功能部件，ETM 紧密地连接于 ARM926EJ-S 处理器。嵌入式跟踪是一个标准的中等以上规模的实现，并包含以下资源：

- 四对地址比较器
- 两个数据比较器
- 八个存储器映射解码输入
- 两个 16 位计数器
- 一个 3-state sequencer
- 四个外部输入
- 一个外部输出
- 一个 45 字节 FIFO

AT91SAM9261 的嵌入式跟踪宏单元工作在半速率时钟模式因而集成了一个时钟分频器。因此，所有跟踪端口信号的最大频率不超过 ARM926EJ-S 时钟速率的一半。

ETM 输入和输出资源在 AT91SAM9261 中不被使用。

关于 ETM 的更多细节，参见 ARM 文档：

1．ETM（Rev2p2）技术参考手册（DDI0157F）

2．ETM 规格说明(IHI 0014J)

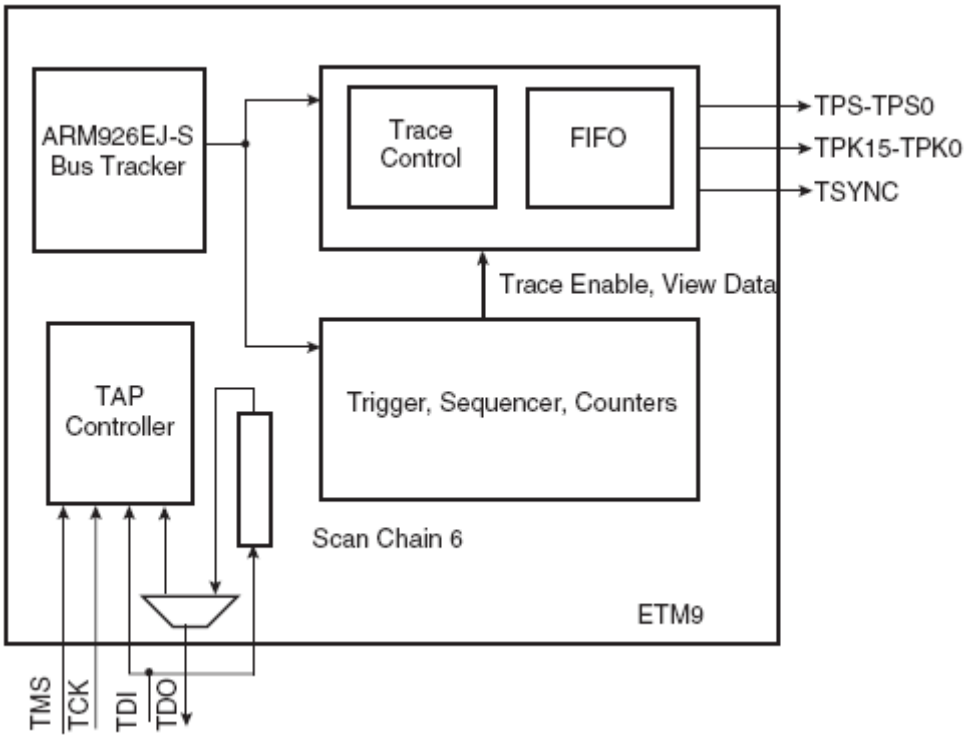
12.5.5.1 跟踪端口

跟踪端口由以下引脚构成：

- TSYNC-同步信号（表示在跟踪包端口上一个转移顺序的开始。）
- TCLK-跟踪端口时钟， ARM926EJ-S 处理器时钟频率的二分之一。
- TPS0 到 TPS2-表示在每个跟踪时钟边沿的处理器状态。
- TPK0 到 TPK15-跟踪包数据值。

跟踪包信息（地址，数据）和 TPS 表示的处理器状态相关联。一些处理器状态在跟踪包端口没有关联的附加数据（比如说，一个失败了的指令条件代码）。此包 8 位宽度，每周期可以输出多达两个包。

图 12-4 ETM9 方块图



12.5.5.2 实现细节

此段给出了嵌入式跟踪资源的概述。

Three-state Sequencer

Sequencer 有三可能的下一状态（一个专用于自己，两个用于其他），sequencer 在每个时钟周期都可以改变状态。状态转换由内部事件控制。如果用户需要 multi-stage trigger schemes，触发事件将基于一个 sequencer 状态。

地址比较器

在单一模式中，地址比较器将指令地址或数据地址和用户编程的地址进行比较。

在范围模式中，地址比较器被成对安排来形成一个虚拟地址范围资源。

地址比较器编程的细节是：

- 第一个比较器用范围开始地址编程。
- 第二个比较器用范围结束地址编程。
- 如果地址在以下范围内，则资源匹配：
  -（地址>=范围开始地址）AND（地址<范围结束地址）
- 如果两个地址比较器没有用同一种方法配置，则会产生不可预料的行为。

数据比较器

每个全地址比较器和一个特定数据比较器关联。仅当装载和存储操作发生时，一个数据比较器才被用于监测数据总线。

一个数据比较器有一个数值寄存器和一个屏蔽寄存器，因此能仅对一个预编程值的某些特定位和数据总线进行比较。

存储器解码输入

八个存储器映射解码输入被连接于客户地址解码器。地址解码器把存储器分成片上 SRAM 区，片上 ROM 区，和外设区。地址解码器还优化

ETM9 跟踪触发。

表 12—2 ETM 存储器映射输入层

产品资源	区域	访问类型	开始地址	结束地址
SRAM	内部	数据	0x0000 0000	0x002F FFFF
SRAM	内部	取指	0x0000 0000	0x002F FFFF
ROM	内部	数据	0x0040 0000	0x004F FFFF
ROM	内部	取指	0x0040 0000	0x004F FFFF
外部总线接口	外部	数据	0x1000 0000	0x8FFF FFFF
外部总线接口	外部	取指	0x1000 0000	0x8FFF FFFF
用户外设	内部	数据	0xF000 0000	0xFFFF BFFF
系统外设	内部	数据	0xFFFF C000	0xFFFF FFFF

FIFO

一个 45 字节 FIFO 被用于存储跟踪数据。FIFO 用于从跟踪包中分离流水线状态，因此，FIFO 可以被用于缓冲跟踪包。

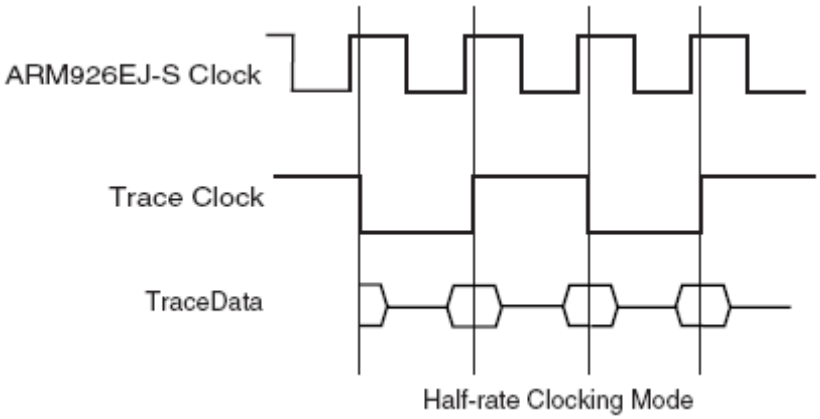
当 FIFO 溢出或当 FIFO 字节比用户编程的字节数少，则嵌入式跟踪宏单元检测到 FIFO 溢出。

半速率时钟模式

ETM9 在半速率模式下被实现，此模式允许在跟踪时钟的上升沿和下降沿进行数据跟踪。

半速率模式可以保证高速系统（达 100Mhz）的信号完整性。

图 12-5 半速率时钟模式



必须注意跟踪捕捉系统的选择，此系统必须要支持半速率时钟功能。

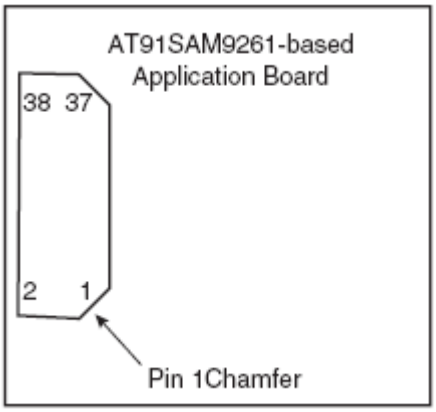
12.5.5.3 应用板限制

TCLK 信号需要小心设置，一些定时参数是必需的。

特定目标系统连接头采用 AMP Mictor 连接器。

连接器必须如图 12-6 所示定位于应用板。以下的 PCB 视图是从顶部俯视的，跟踪连接器安装在靠近板子边沿。这就使得跟踪端口分析器对互连目标的影响最小化。

图 12-6 AMP Mictor 连接器



12.5.6 IEEE 1149.1JTAG 边界扫描

IEEE 1149.1 JTAG 边界扫描允许独立于设备封装技术的引脚级访问。

IEEE 1149.1JTAG 边界扫描当 JTAGSEL 为高电平时被使能。SAMPLE,EXTEST 和 BYPASS 功能被实现。在 ICE 调试模式，ARM 处理器回应一个非 JTAG 芯片 ID 响应到 ICE 系统用于识别处理器。这不适用于 IEEE 1149.1 JTAG。

直接转换 JTAG 和 ICE 操作是不可能的。JTAGSEL 被改变后必须执行一个复位操作。

提供一个边界扫描描述符语言（BSDL）文件用于建立测试。

12.5.6.1 JTAG 边界扫描寄存器

边界扫描寄存器（BSR）包含 484 位，这些位对应于主动引脚和关联的控制信号。

每个 AT91SAM9261 输入/输出引脚对应到 BSR 中的一个 3 位寄存器。OUTPUT 位包含能被强制输出在引脚上的数据。INPUT 位提供加到引脚上的数据以供观测。CONTROL 位选择引脚的方向。

表 12-3 AT91SAM9261 JTAG 边界扫描寄存器

位号	引脚名称	引脚类型	关联的 BSR 单元
483	A18	OUT	OUTPUT
482	A[22:16]		CONTROL
481	A18	OUT	OUTPUT
480	A20	OUT	OUTPUT
479	A21	OUT	OUTPUT
478	A22	OUT	OUTPUT
477	NCS0	OUT	OUTPUT
476	A[7:0]		CONTROL
475	NCS1	OUT	OUTPUT
474	NCS0/NCS1/NCS2/NCS3 NRD/NWR0/NWR1/NWR3		CONTROL
473	NCS2	OUT	OUTPUT
472	NCS3	OUT	OUTPUT
471	NRD	OUT	OUTPUT
470	NWR0	IN/OUT	INPUT
469			OUTPUT
468	NWR1	IN/OUT	INPUT
467			OUTPUT
466		内部	I
465	NWR3	OUT	OUTPUT
464	SDRAMCKE	OUT	OUTPUT
463	SDRAMCKE/RAS/CAS SDA10/SDWE		CONTROL
462	SDRAMCLK	IN/OUT	INPUT
461			OUTPUT
460			CONTROL
459	RAS	OUT	OUTPUT
458	CAS	OUT	OUTPUT
457	SDWE	OUT	OUTPUT

表 12-3 AT91SAM9261 JTAG 边界扫描寄存器(续)

位号	引脚名称	引脚类型	关联的 BSR 单元
456	D0	IN/OUT	INPUT
455			OUTPUT
454			CONTROL
453		内部	
452	D1	IN/OUT	INPUT
451			OUTPUT
450			CONTROL
449	D2	IN/OUT	INPUT
448			OUTPUT
447			CONTROL
446	D3	IN/OUT	INPUT
445			OUTPUT
444			CONTROL
443	D4	IN/OUT	INPUT
442			OUTPUT
441			CONTROL
440		内部	
439	D5	IN/OUT	INPUT
438			OUTPUT
437			CONTROL
436	D6	IN/OUT	INPUT
435			OUTPUT
434			CONTROL
433	D7	IN/OUT	INPUT
432			OUTPUT
431			CONTROL
430	D8	IN/OUT	INPUT
429			OUTPUT
428			CONTROL
427		内部	
426	D9	IN/OUT	INPUT
425			OUTPUT
424			CONTROL
423	D10	IN/OUT	INPUT
422			OUTPUT
421			CONTROL

表 12-3 AT91SAM9261 JTAG 边界扫描寄存器(续)

位号	引脚名称	引脚类型	关联的 BSR 单元
420	D11	IN/OUT	INPUT
419			OUTPUT
418			CONTROL
417	D12	IN/OUT	INPUT
416			OUTPUT
415			CONTROL
414		内部	
413	D13	IN/OUT	INPUT
412			OUTPUT
411			CONTROL
410	D14	IN/OUT	INPUT
409			OUTPUT
408			CONTROL
407	D15	IN/OUT	INPUT
406			OUTPUT
405			CONTROL
404	PC16	IN/OUT	INPUT
403			OUTPUT
402			CONTROL
401		内部	
400	PC17	IN/OUT	INPUT
399			OUTPUT
398			CONTROL
397		内部	
396	PC18	IN/OUT	INPUT
395			OUTPUT
394			CONTROL
393		内部	
392	PC19	IN/OUT	INPUT
391			OUTPUT
390			CONTROL
389		内部	
388	PC30	IN/OUT	INPUT
387			OUTPUT
386			CONTROL
385		内部	

表 12-3 AT91SAM9261 JTAG 边界扫描寄存器(续)

位号	引脚名称	引脚类型	关联的 BSR 单元
384	PC31	IN/OUT	INPUT
383			OUTPUT
382			CONTROL
381		内部	
380	PC20	IN/OUT	INPUT
379			OUTPUT
378			CONTROL
377		内部	
376	PC21	IN/OUT	INPUT
375			OUTPUT
374			CONTROL
373		内部	
372	PC22	IN/OUT	INPUT
371			OUTPUT
370			CONTROL
369		内部	
368	PC23	IN/OUT	INPUT
367			OUTPUT
366			CONTROL
365		内部	
364	PC24	IN/OUT	INPUT
363			OUTPUT
362			CONTROL
361		内部	
360	PC25	IN/OUT	INPUT
359			OUTPUT
358			CONTROL
357		内部	
356	PC26	IN/OUT	INPUT
355			OUTPUT
354			CONTROL
353		内部	
352	PC27	IN/OUT	INPUT
351			OUTPUT
350			CONTROL
349		内部	

表 12-3 AT91SAM9261 JTAG 边界扫描寄存器(续)

位号	引脚名称	引脚类型	关联的 BSR 单元
348	PC28	IN/OUT	INPUT
347			OUTPUT
346			CONTROL
345		内部	
344	PC29	IN/OUT	INPUT
343			OUTPUT
342			CONTROL
341		内部	
340	PC0	IN/OUT	INPUT
339			OUTPUT
338			CONTROL
337		内部	
336	PC1	IN/OUT	INPUT
335			OUTPUT
334			CONTROL
333		内部	
332	PC2	IN/OUT	INPUT
331			OUTPUT
330			CONTROL
329		内部	
328	PC3	IN/OUT	INPUT
327			OUTPUT
326			CONTROL
325		内部	
324	PC4	IN/OUT	INPUT
323			OUTPUT
322			CONTROL
321		内部	
320	PC5	IN/OUT	INPUT
319			OUTPUT
318			CONTROL
317		内部	
316	PC6	IN/OUT	INPUT
315			OUTPUT
314			CONTROL
313		内部	

表 12-3 AT91SAM9261 JTAG 边界扫描寄存器(续)

位号	引脚名称	引脚类型	关联的 BSR 单元
312	PC7	IN/OUT	INPUT
311			OUTPUT
310			CONTROL
309		内部	
308	PC8	IN/OUT	INPUT
307			OUTPUT
306			CONTROL
305		内部	
304	PC9	IN/OUT	INPUT
303			OUTPUT
302			CONTROL
301		内部	
300	PC10	IN/OUT	INPUT
299			OUTPUT
298			CONTROL
297		内部	
296	PC11	IN/OUT	INPUT
295			OUTPUT
294			CONTROL
293		内部	
292	PC12	IN/OUT	INPUT
291			OUTPUT
290			CONTROL
289		内部	
288	PC13	IN/OUT	INPUT
287			OUTPUT
286			CONTROL
285		内部	
284	PC14	IN/OUT	INPUT
283			OUTPUT
282			CONTROL
281		内部	
280	PC15	IN/OUT	INPUT
279			OUTPUT
278			CONTROL
277		内部	

表 12-3 AT91SAM9261 JTAG 边界扫描寄存器(续)

位号	引脚名称	引脚类型	关联的 BSR 单元
276	PA0	IN/OUT	INPUT
275			OUTPUT
274			CONTROL
273		内部	
272	PA1	IN/OUT	INPUT
271			OUTPUT
270			CONTROL
269		内部	
268	PA2	IN/OUT	INPUT
267			OUTPUT
266			CONTROL
265		内部	
264	PA3	IN/OUT	INPUT
263			OUTPUT
262			CONTROL
261		内部	
260	PA4	IN/OUT	INPUT
259			OUTPUT
258			CONTROL
257		内部	
256	PA5	IN/OUT	INPUT
255			OUTPUT
254			CONTROL
253		内部	
252	PA6	IN/OUT	INPUT
251			OUTPUT
250			CONTROL
249		内部	
248	PA7	IN/OUT	INPUT
247			OUTPUT
246			CONTROL
245		内部	
244	PA8	IN/OUT	INPUT
243			OUTPUT
242			CONTROL
241		内部	

表 12-3 AT91SAM9261 JTAG 边界扫描寄存器(续)

位号	引脚名称	引脚类型	关联的 BSR 单元
240	PA9	IN/OUT	INPUT
239			OUTPUT
238			CONTROL
237		内部	
236	PA10	IN/OUT	INPUT
235			OUTPUT
234			CONTROL
233		内部	
232	PA11	IN/OUT	INPUT
231			OUTPUT
230			CONTROL
229		内部	
228	PA12	IN/OUT	INPUT
227			OUTPUT
226			CONTROL
225		内部	
224	PA13	IN/OUT	INPUT
223			OUTPUT
222			CONTROL
221		内部	
220	PA14	IN/OUT	INPUT
219			OUTPUT
218			CONTROL
217		内部	
216	PA15	IN/OUT	INPUT
215			OUTPUT
214			CONTROL
213		内部	
212	PA16	IN/OUT	INPUT
211			OUTPUT
210			CONTROL
209		内部	
208	PA17	IN/OUT	INPUT
207			OUTPUT
206			CONTROL
205		内部	

表 12-3 AT91SAM9261 JTAG 边界扫描寄存器(续)

位号	引脚名称	引脚类型	关联的 BSR 单元
204	PA18	IN/OUT	INPUT
203			OUTPUT
202			CONTROL
201		内部	
200	PA19	IN/OUT	INPUT
199			OUTPUT
198			CONTROL
197		内部	
196	PA20	IN/OUT	INPUT
195			OUTPUT
194			CONTROL
193		内部	
192	PA21	IN/OUT	INPUT
191			OUTPUT
190			CONTROL
189		内部	
188	PA22	IN/OUT	INPUT
187			OUTPUT
186			CONTROL
185		内部	
184	PA23	IN/OUT	INPUT
183			OUTPUT
182			CONTROL
181		内部	
180	PA24	IN/OUT	INPUT
179			OUTPUT
178			CONTROL
177		内部	
176	PA25	IN/OUT	INPUT
175			OUTPUT
174			CONTROL
173		内部	
172	PA26	IN/OUT	INPUT
171			OUTPUT
170			CONTROL
169		内部	

表 12-3 AT91SAM9261 JTAG 边界扫描寄存器(续)

位号	引脚名称	引脚类型	关联的 BSR 单元
168	PA27	IN/OUT	INPUT
167			OUTPUT
166			CONTROL
165		内部	
164	PA28	IN/OUT	INPUT
163			OUTPUT
162			CONTROL
161		内部	
160	PA29	IN/OUT	INPUT
159			OUTPUT
158			CONTROL
157		内部	
156	PA30	IN/OUT	INPUT
155			OUTPUT
154			CONTROL
153		内部	
152	PA31	IN/OUT	INPUT
151			OUTPUT
150			CONTROL
149		内部	
148	PB0	IN/OUT	INPUT
147			OUTPUT
146			CONTROL
145		内部	
144	PB1	IN/OUT	INPUT
143			OUTPUT
142			CONTROL
141		内部	
140	PB2	IN/OUT	INPUT
139			OUTPUT
138			CONTROL
137		内部	
136	PB3	IN/OUT	INPUT
135			OUTPUT
134			CONTROL
133		内部	

表 12-3 AT91SAM9261 JTAG 边界扫描寄存器(续)

位号	引脚名称	引脚类型	关联的 BSR 单元
132	PB4	IN/OUT	INPUT
131			OUTPUT
130			CONTROL
129		内部	
128	PB5	IN/OUT	INPUT
127			OUTPUT
126			CONTROL
125		内部	
124	PB6	IN/OUT	INPUT
123			OUTPUT
122			CONTROL
121		内部	
120	PB7	IN/OUT	INPUT
119			OUTPUT
118			CONTROL
117		内部	
116	PB8	IN/OUT	INPUT
115			OUTPUT
114			CONTROL
113		内部	
112	PB9	IN/OUT	INPUT
111			OUTPUT
110			CONTROL
109		内部	
108	PB10	IN/OUT	INPUT
107			OUTPUT
106			CONTROL
105		内部	
104	PB11	IN/OUT	INPUT
103			OUTPUT
102			CONTROL
101		内部	
100	PB12	IN/OUT	INPUT
99			OUTPUT
98			CONTROL
97		内部	

表 12-3 AT91SAM9261 JTAG 边界扫描寄存器(续)

位号	引脚名称	引脚类型	关联的 BSR 单元
96	PB13	IN/OUT	INPUT
95			OUTPUT
94			CONTROL
93		内部	
92	PB14	IN/OUT	INPUT
91			OUTPUT
90			CONTROL
89		内部	
88	PB15	IN/OUT	INPUT
87			OUTPUT
86			CONTROL
85		内部	
84	PB16	IN/OUT	INPUT
86			OUTPUT
85			CONTROL
84		内部	
83	PB17	IN/OUT	INPUT
82			OUTPUT
81			CONTROL
80		内部	
79	PB18	IN/OUT	INPUT
78			OUTPUT
77			CONTROL
76		内部	
75	PB19	IN/OUT	INPUT
74			OUTPUT
73			CONTROL
72		内部	
71	PB20	IN/OUT	INPUT
70			OUTPUT
69			CONTROL
68		内部	
67	PB21	IN/OUT	INPUT
66			OUTPUT
65			CONTROL
64		内部	

表 12-3 AT91SAM9261 JTAG 边界扫描寄存器(续)

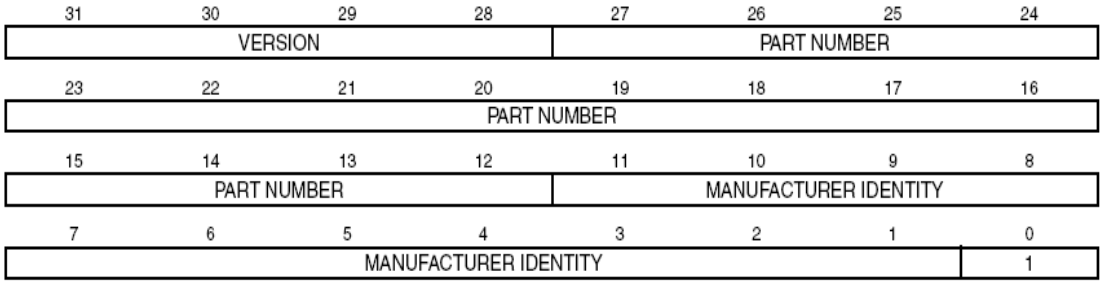
位号	引脚名称	引脚类型	关联的 BSR 单元
60	PB22	IN/OUT	INPUT
59			OUTPUT
58			CONTROL
57		内部	
56	PB23	IN/OUT	INPUT
55			OUTPUT
54			CONTROL
53		内部	
52	PB24	IN/OUT	INPUT
51			OUTPUT
50			CONTROL
49		内部	
48	PB25	IN/OUT	INPUT
47			OUTPUT
46			CONTROL
45		内部	
44	PB26	IN/OUT	INPUT
43			OUTPUT
42			CONTROL
41		内部	
40	PB27	IN/OUT	INPUT
39			OUTPUT
38			CONTROL
37		内部	
36	PB28	IN/OUT	INPUT
35			OUTPUT
34			CONTROL
33		内部	
32	PB29	IN/OUT	INPUT
31			OUTPUT
30			CONTROL
29		内部	
28	PB30	IN/OUT	INPUT
27			OUTPUT
26			CONTROL
25		内部	

表 12-3 AT91SAM9261 JTAG 边界扫描寄存器(续)

位号	引脚名称	引脚类型	关联的 BSR 单元
24	PB31	IN/OUT	INPUT
23			OUTPUT
22			CONTROL
21		内部	
20	A0	OUT	OUTPUT
19		内部	
18	A1	OUT	OUTPUT
17	A2	OUT	OUTPUT
16	A3	OUT	OUTPUT
15	A4	OUT	OUTPUT
14	A5	OUT	OUTPUT
13	A6	OUT	OUTPUT
12	A7	OUT	OUTPUT
11	A8	OUT	OUTPUT
10	A[15:8]		CONTROL
09	A9	OUT	OUTPUT
08	A10	OUT	OUTPUT
07	SDA10	OUT	OUTPUT
06	A11	OUT	OUTPUT
05	A12	OUT	OUTPUT
04	A13	OUT	OUTPUT
03	A14	OUT	OUTPUT
02	A15	OUT	OUTPUT
01	A16	OUT	OUTPUT
00	A17	OUT	OUTPUT

12.5.7 ID 代码寄存器

访问：只读



- VERSION[31：28]：产品版本号

设置为 0x0.

- PART NUMBER[27：12]：产品部件号

产品部件号是 0x5B08

- MANUFACTURER IDENTITY[11：1]

设置为 0x01F

Bit[0]根据 IEEE Std.1149.1 要求

设置为 0x1.

JTAG ID 代码值是 0x05B0_803F.

13. AT91SAM9261 启动程序

13.1 描述

启动程序集成了不同的程序，这些程序管理下载和/或上传产品的不同存储器。

首先，它初始化调试部件串行端口（DBGU）和 USB 设备端口。

接着执行 DataFlash 启动程序。此程序在连接于 SPI 的 DataFlash 中寻找七个有效的 ARM 异常向量的序列。所有这些向量必须是 B-branch 或 LDR 装载寄存器指令，除了第六个向量。第六个向量被用于存储下载的镜像大小。

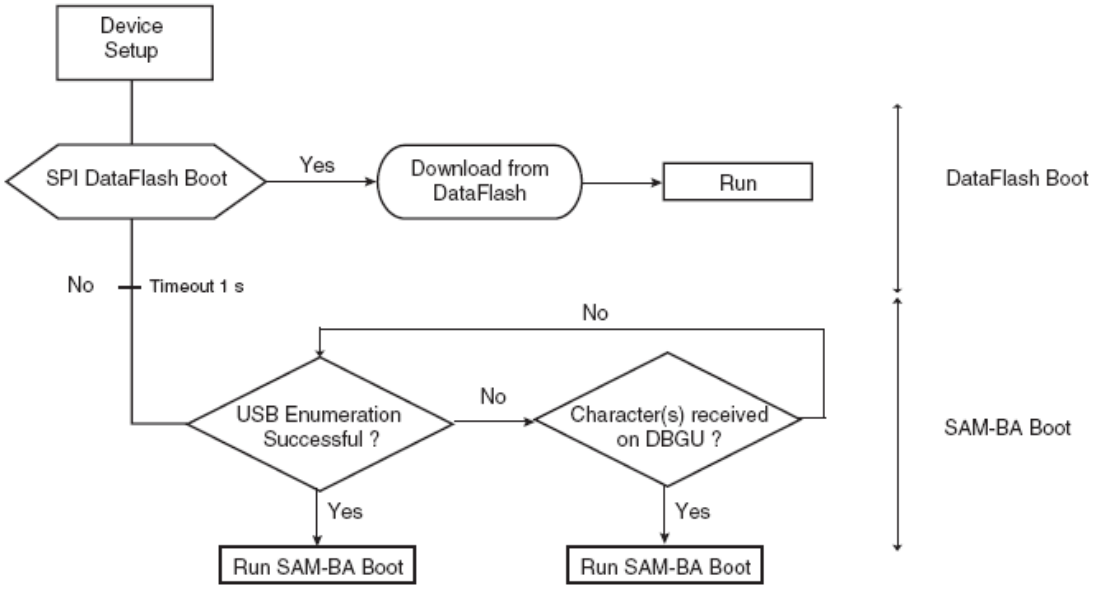
如果一个有效的序列被找到，代码将被下载到内部 SRAM。接着重映射，并跳到 SRAM 的首地址。

如果没有有效的 ARM 向量序列，则接着执行 SAM-BA。它等待 USB 设备或 DBGU 串行端口上的事件。

13.2 流程图

启动程序实现算法见图 13-1。

图 13-1 启动程序算法流程图



13.3 设备初始化

初始化按照如下描述的步骤进行：

1. 为 ARM 管理模式(SVC)设置堆栈
2. 主振荡器频率检测
3. C 变量初始化
4. PLL 设置：PLLB 被初始化并用来为 USB 设备产生一个 48MHz 时钟。一个置于电源管理控制器（PMC）中的寄存器决定主振荡器的频率，以及 PLLB 的正确参数。

表 13-1 定义了启动程序支持的晶振

表 13-1 由软件自动检测支持的晶振（MHz）

3.0	3.2768	3.6864	3.84	4.0
4.433619	4.608	4.9152	5.0	5.24288
6.0	6.144	6.4	6.5536	7.159090
7.3728	7.864320	8.0	9.8304	10.0
11.05920	12.0	12.288	13.56	14.31818
14.7456	16.0	17.734470	18.432	20.0

5． DBGU 串行端口的初始化（115200bps，8，N,1）

6． 禁用看门狗并使能用户复位

7． USB 设备端口的初始化

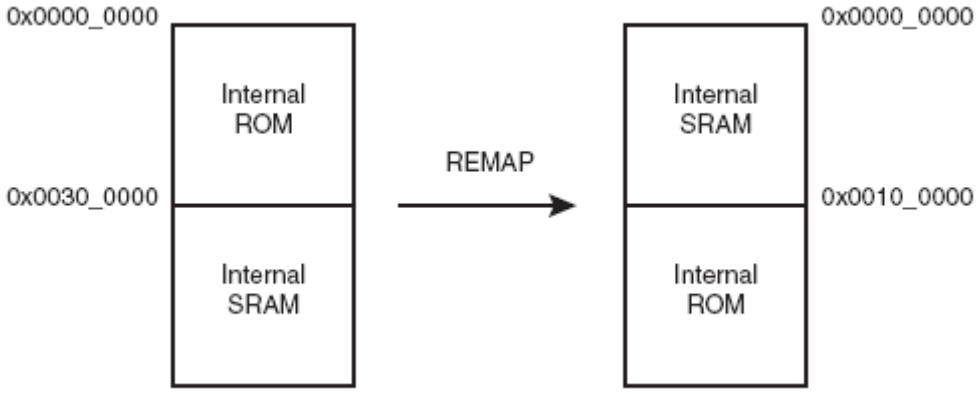
8． 跳至 DataFlash 启动序列

如果 DataFlash 启动失败：

9． 使能指令缓存

10． 跳至 SAM-BA 启动序列

图 13-2 下载完成后重映射方式



13.4 DataFlash 启动

DataFlash 启动程序在 SPI DataFlash 存储器中寻找一个有效的应用程序。有效的应用程序一旦被找到，该程序将被加载到内部 SRAM，在 remap 之后通过一个跳转到 0x00000000 的指令开始运行。此应用程序可能是应用程序代码或一个二级 bootloader。

所有的程序调用均基于 PC，不要使用绝对地址。

复位后，内部 ROM 的代码被映射到地址 0x0000_0000 和 0x0010_0000：

```
400000  ea000006  B       0x20                00  ea000006  B       0x20
400004  eafffffe  B       0x04                04  eafffffe  B       0x04
400008  ea00002f  B       _main               08  ea00002f  B       _main
40000c  eafffffe  B       0x0c                0c  eafffffe  B       0x0c
400010  eafffffe  B       0x10                10  eafffffe  B       0x10
400014  eafffffe  B       0x14                14  eafffffe  B       0x14
400018  eafffffe  B       0x18                18  eafffffe  B       0x18
```

13.4.1 有效镜像检测

DataFlash 启动软件通过分析开始的 28 位字节，对应 ARM 异常向量，来寻

找一个有效的应用程序。这些字节必须以 ARM 指令跳转或使用基于 PC 的相对地址装载 PC 来实现。

第六个向量，偏移量 0x14，包含下载镜像的大小。用户必须用自己的向量代替此向量。（见 87 页“ARM 第六个向量的结构”）。

图 13-3 LDR 操作码

31 28	27 24	23 20	19 16	15 12	11 0
1 1 1 0	1 1 I P	U 1 W 0	Rn	Rd	

图 13-4 分支(B)操作

31 28	27 24	23 0
1 1 1 0	1 0 1 0	偏移量（24 位）

无条件指令：0xE 用于 31 位到 28 位

用基于 PC 的相对地址装载 PC：

-Rn=Rd=PC=0xF

-I=1

-P=1

-U offset added(U==1) or subtracted(U==0)

-W=1

13.4.2 ARM 第六个向量的结构

ARM 第六个例外向量被用作存储 DataFlash 启动程序所需的信息。

此信息描述如下。

图 13-5 ARM 第六个向量的结构

31 0
以字节计算的下载的代码大小

13.4.2.1 举例

以下是有效向量的一个例子：

```
00    ea000006  B       0x20
04    eafffffe  B       0x04
08    ea00002f  B       _main
0c    eafffffe  B       0x0c
10    eafffffe  B       0x10
14    00001234  B       0x14      <- Code size = 4660 bytes
18    eafffffe  B       0x18
```

装载到 SRAM 的镜像大小包含在第六个 ARM 向量的位置。

因此用户必须用自己的应用程序提供正确的值代替此向量。

13.4.3 DataFlash 启动顺序

DataFlash 启动程序执行设备初始化，接着下载程序。

DataFlash 启动程序支持所有的 Atmel DataFlash 设备。表 13-2 总结了包括所有设备的用于 ARM 第六个向量的参数。

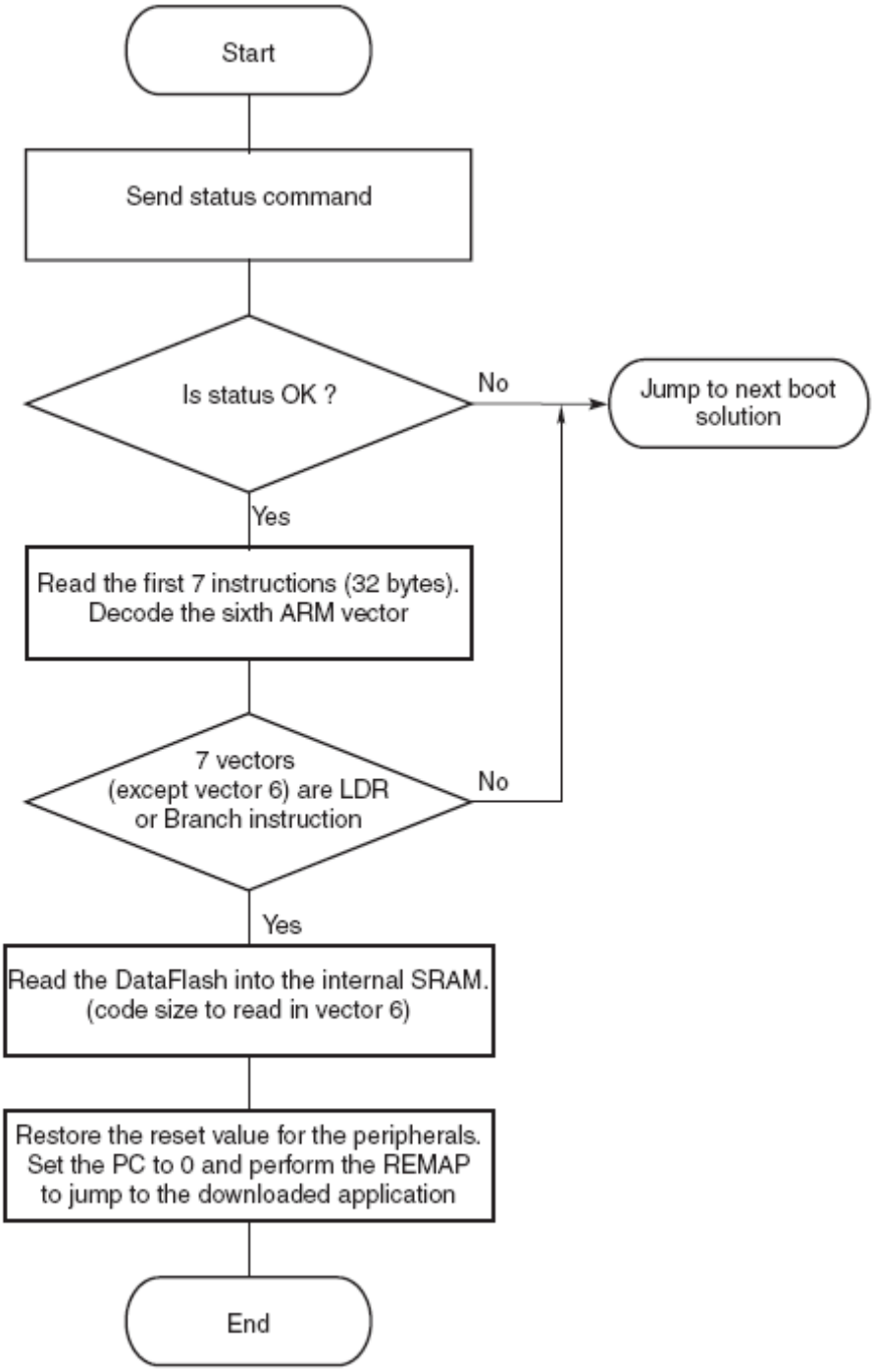
表 13-2 DataFlash 设备

设备	密度	页面大小（字节）	页数
AT45DB011B	1M 位	264	512
AT45DB021B	2 M 位	264	1024
AT45DB041B	4 M 位	264	2048
AT45DB081B	8 M 位	264	4096
AT45DB161B	16 M 位	528	4096
AT45DB321B	32 M 位	528	8192
AT45DB642B	64 M 位	1056	8192

DataFlash 有一状态寄存器，此寄存器决定访问设备时需要的所有参数。

DataFlash 启动程序被配置成和以后设计更新的 DataFlash 相兼容。

图 13-6　串行 DataFlash 下载



13.5　SAM-BA 启动

如果在 DataFlash 启动序列中未找到有效的 DataFlash 设备，SAM-BA 启动程序被执行。

SAM-BA 启动规则：

-检查 USB 设备的枚举事件

-检查 DBGU 上是否接收到字符

-一旦通信接口被识别，应用程序运行在一个等待不同命令的无限循环，这些命令见表 13-3

表 13-3 SAM-BA 启动可用的命令

命令	方式	自变量（s）	举例
O	写一个字节	Address,Value#	O200001,CA#
o	读一个字节	Address,#	o200001,#
H	写一个半字	Address, Value #	H200002,CAFÉ#
H	读一个半字	Address,#	h200002,#
W	写一个字	Address, Value #	W200000,CAFEDECA#
w	读一个字	Address,#	w200000,#
S	发送一个文件	Address,#	S200000,#
R	接收一个文件	Address,NbOfBytes#	R200000,1234#
G	到	Address,#	G200200#
V	显示版本	No argument	V#

- 写命令：写一个字节（O），一个半字（H）或一个字（W）到目标。
  -地址：十六进制地址。
  -值：字节，半字或字，用十六进制写。
  -输出：‘>’。
- 读命令：从目标读一个字节（o），一个半字（h）或一个字（w）。
  -地址：十六进制地址。
  -输出：十六进制读字节，半字或字，其后紧跟‘>’
- 发送一个文件（S）：发送一个文件到特定地址
  - 地址： 十六进制地址
  - NbOfBytes：用十六进制接收的字节数
  - 输出：‘>’
- Go（G）：跳至一个特定的地址并执行代码
  - 地址：十六进制地址跳转
  - 输出：‘>’
- 得到版本信息：返回 SAM-BA 启动版本
  - 输出：‘> ’

13.5.1 DBGU 串行端口

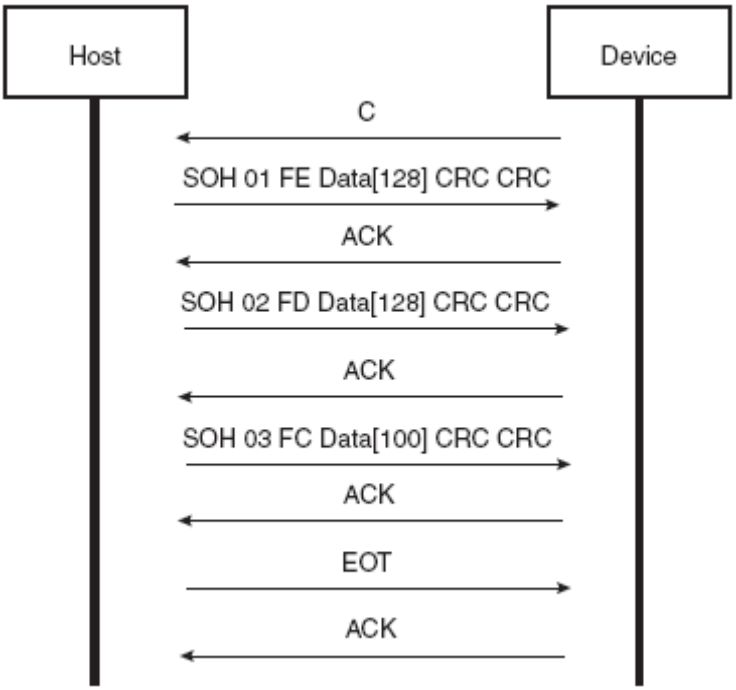
通过初始化为 115200，8，n，1 的 DBGU 串行端口来实现通信。
发送和接收文件命令采用 Xmodem 协议。任何执行此协议的终端可以被用于发送应用程序文件到目标。发送二进制文件的大小依赖于产品中的 SRAM 的大小。在所有的情形下，二进制文件的大小必须比 SRAM 的容量小，因为 Xmodem 协议需要一些 SRAM 存储空间才能工作。

13.5.2 Xmodem 协议

支持 Xmodem 协议的是 128 字节长的块。此协议用一个双字符 CRC-16 保证检测到一个最大位错误(maximum bit error)。
带 CRC 的 Xmodem 协议精确的提供发送器和接收器传送成功的报告。
每个传送块如:
<SOH><blk #><255-blk #><--128 data bytes--><checksum> in which:

– <SOH> = 01 hex
– <blk #> = binary number, starts at 01, increments by 1, and wraps 0FFH to 00H (not to 01)
- <255-blk #> = 1's complement of the blk#.
- <checksum> = 2 bytes CRC16\
图 13-7 展示了用此协议的一个传送。

图 13-7 Xmodem 传送举例



13.5.3 USB 设备端口

使用 USB 设备端口时必须提供一个 48MHz USB 时钟。在设备初始化过程早期已经通过对 PLLB 的配置完成。

设备采用 USB 通信设备类(CDC)，以利于已安装驱动程序的 PC 与 USB 设备交互。CDC 类在所有的 Window 版本中都被实现，从 Windows 98SE 到 Windows XP，CDC 文档可以在 www.usb.org 上得到。它描述了一种实现像 ISDN modem 和虚拟 COM 端口设备的方法。

USB 设备的厂商 ID 是 Atmel 的 ID：0x03EB。 产品 ID 是 0x6124。主机操作系统通过这些 ID 来正确安装合适的驱动。在 Windows 系统 ，INF 文件包含厂商 ID 和产品 ID 间的信息。

Atmel 提供一个 INF 例子可将设备看作一个新的串行端口，也提供另一个 SAM-BA 应用程序使用的客户驱动：atm6124.sys。更多细节，参考文档"USB 基本应用"(USB Basic Application)，文献号 6123。

13.5.3.1 枚举过程

USB 协议是一个主/从协议。主机通过控制端点发送请求到设备来开始枚举。设备处理在 USB 规范说明中定义的标准请求。

表 13-4 处理标准请求

请求	定义
GET_DESCRIPTOR	返回当前设备描述符
SET_ADDRESS	设置设备地址，并用于访问设备功能
SET_CONFIGURATION	设置设备配置
GET_CONFIGURATION	返回当前设备配置值
GET_STATUS	返回指定接受者的状态
SET_FEATURE	用于置位或使能特定的功能部件
CLEAR_FEATURE	用于清零或禁用一个特定的功能部件

设备还处理一些在 CDC 类中定义的类请求。

表 13-5 处理类请求

表

请求 1	定义
SET_LINE_CODING 3	配置 DTE 速率，停止位，奇偶校验位和字符位数。
GET_LINE_CODING 5	获得当前 DTE 速率，停止位，奇偶校验位和字符位数
SET_CONTROL_LINE_STATE 处	RS-232 信号用于告知 DCE 设备，DTE 设备现在存在。

不被处理的请求将被 STALL。

13.5.3.2 通信端点（endpoint）

有两个通信端点，而端点 0 被用作枚举。端网点 1 是一个 64 字节 Bulk(批量)输出端点，端点 2 是一个 64 字节 Bulk 输入端点。SAM-BA 启动命令被主机通过端点 1 发送。如果需要，信息被主机通过主机驱动分为若干有效载荷。如果命令需要一个响应，主机可以发出 IN 事务去获得响应。

13.6 硬件和软件限制

- 下载到 DataFlash 的代码必须低于 156K 字节。
- 代码总是从设备地址 0x0000_0000 下载到内部 SRAM 的地址 0x0000_0000（重映射后）.
- 下载的代码必须位置无关或连接在地址 0x0000_0000.
- DataFlash 必须被连接到 SPI 的 NPCS0。

SPI 驱动器用具有若干功能的 PIO 去和设备通信。当这些 PIO 被应用程序使用时必须谨慎。连接的设备可以在启动时被无意的驱动，在 SPI 输出引脚和连接的设备之间的电平冲突也可能出现。

为了确保正确的功能，推荐插入关键设备到其它的引脚。

表 13-6 包含引脚列表，这些引脚在启动程序执行期间被驱动。如果没有找到合适的启动程序，这些引脚将在小于 1 秒的启动序列期间被驱动。

当 SPCK 信号为 8MHz 时驱动的 DataFlash，下载 156K 字节代码所需的时间可被缩短到 200ms.

执行跳转至内部 SRAM 中的应用程序之前，所有在启动程序中使用到的 PIO 和设备被设置为它们在复位时的状态。

表 13-6 在启动程序执行期间被驱动的引脚

外设	引脚	PIO 口线
SPI0	MOSI	PIOA1
SPI0	MISO	PIOA0
SPI0	SPCK	PIOA2
SPI0	NPCS0	PIOA3
DBGU	DRXD	PIOA9
DBGU	DTXD	PIOA10

14. 复位控制器(RSTC)

14.1 描述

复位控制器(RSTC)，基于上电复位单元，不用任何外部元件即可处理系统所有的复位。它报告最近一次发生的复位。

复位控制器还可以独立或同时驱动外部复位和外设以及处理器复位。

14.2 方块图

图 14-1 复位控制器方块图

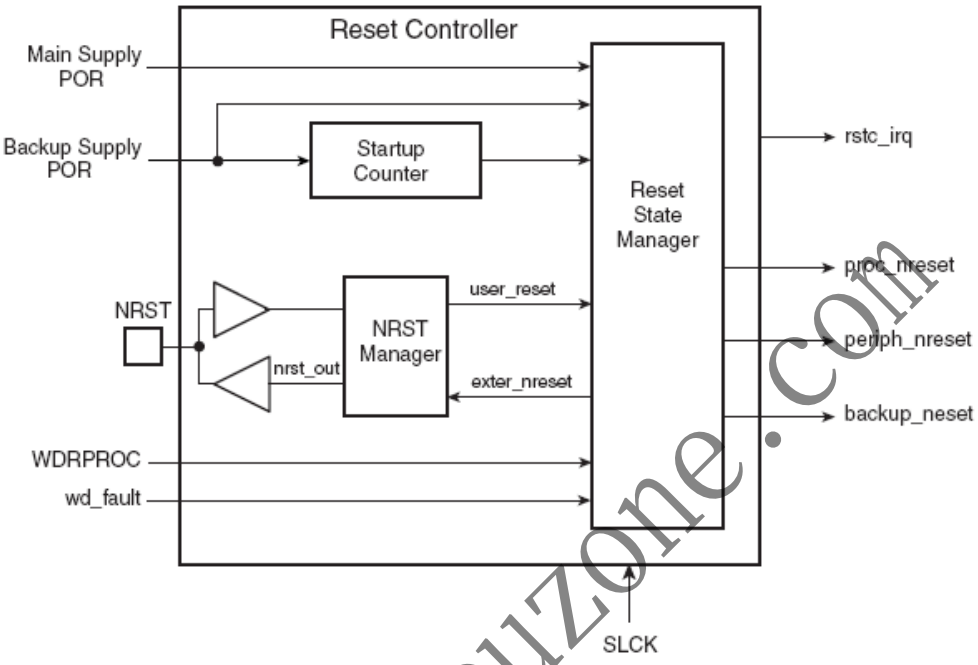
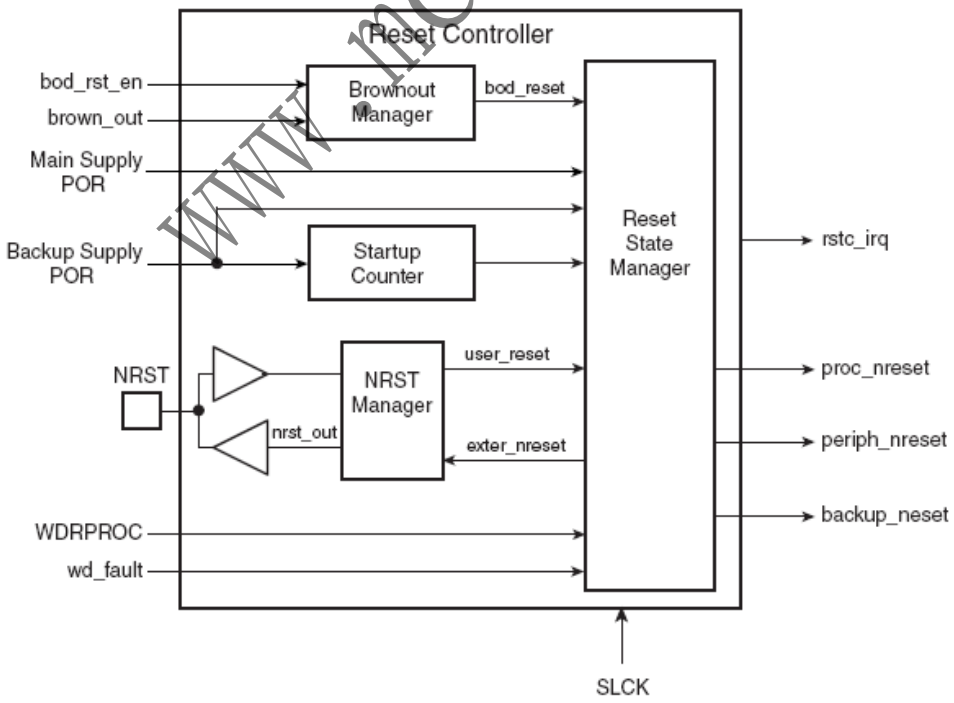


图 14-2 复位控制器方块图



14.3 功能描述

www.mcuzone.com

14.3.1 复位控制器概述

复位控制器由一个 NRST 管理器，一个启动计数器和一个复位状态管理器组成。运行在低速时钟并产生以下复位信号：

- proc_nreset:处理器复位。还复位看门狗定时器
- backup_nreset:影响所有由 VDDBU 提供电源的外设。
- periph_nreset:影响所有的集成外设。
- nrst_out:驱动 NRST 引脚。

这些复位信号在外部事件或软件作用时被复位控制器使能。当一个 NRST 信号需要输出时，复位状态管理器控制复位信号的发生并提供一个到 NRST 管理器的信号。

NRST 管理器在一个可编程的时间里形成 NRST 引脚上的有效信号，以此方式控制外部设备的复位。

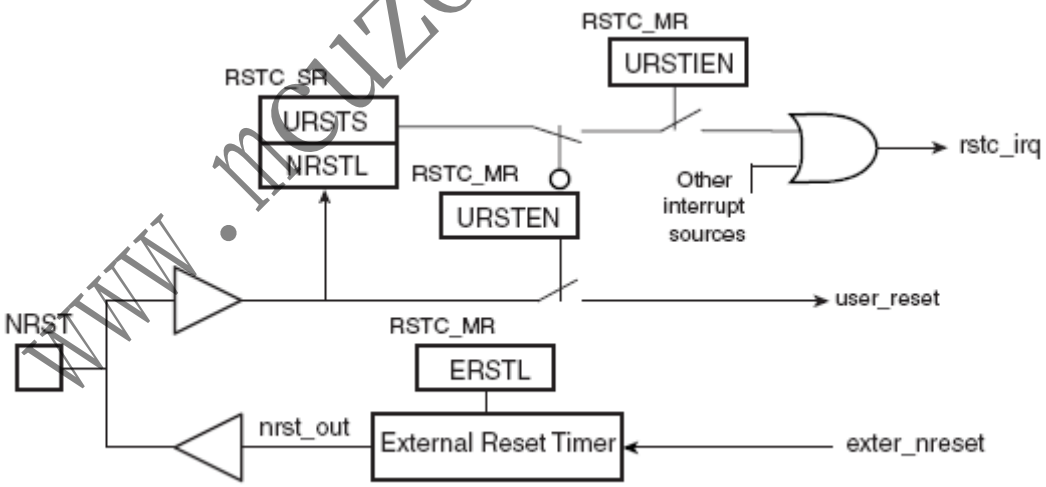
启动计数器等待晶振完全启动。等待晶振启动的时间的最大值可以查表获得，晶振启动时间最大值可以在产品文献的电特性段落中的晶振特性段落中找到。

复位控制器模式寄存器（RSTC_MR），允许复位控制器的配置，RSTC_MR 由 VDDBU 供电，所以只要 VDDBU 电压存在，其配置就会被保存。

14.3.2 NRST 管理器

NRST 管理器采样 NRST 输入引脚，并当复位状态管理器要求时，使此引脚为低电平。图 14-2 展示了 NRST 管理器的方块图。

图 14-3 NRST 管理器



14.3.2.1 NRST 信号或中断

NRST 管理器在慢速时钟采样 NRST 引脚。当此口线被检测为低电平，一个用户复位被报告给复位状态管理器。

然而，当一个 NRST 的有效信号产生， NRST 管理器可以被编程为不触发一个复位。将 RSTC_MR 中的 URSTEN 位写 0 则禁用用户复位发生。

NRST 引脚的电平可以在任何时间在 RSTC_SR 中的 NRSTL 位（NRST 电平）被读取。只要 NRST 引脚被激活，RSTC_SR 中的 URSTS 位被置位。此位仅在 RSTC_SR 被读取时清零。

复位控制器还可以被编程用来发生一个中断来代替发生一个复位。这样做的

www.mcuzone.com

话，RSTC_SR 中的 URSTIEN 位必须被写为 1。

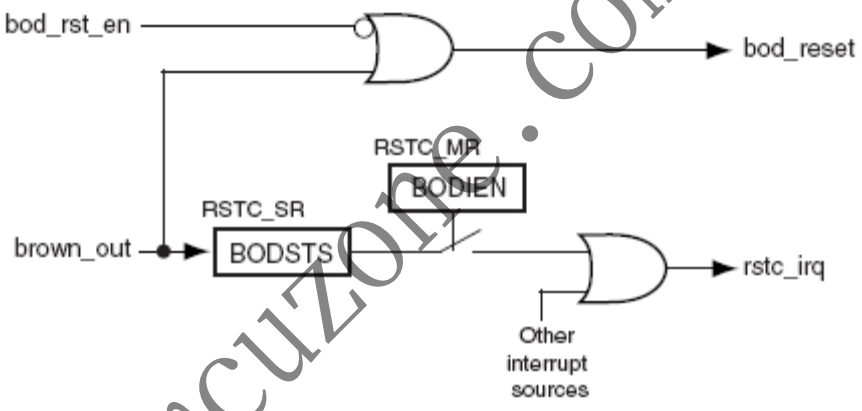
14.3.2.2 NRST 外部复位控制

复位状态管理器激活 ext_nreset 信号来激活 NRST 引脚。当前面的情况发生，在一段通过 RSTC_SR 中的 ERSTL 域编程的时间内，“nrst_out”信号被 NRST 管理器驱动为低。此有效持续时间，名叫 EXTERNAL_RESET_LENGTH，持续了 2^(ERSTL+1)个慢时钟周期。这给出了一个有效的在 60 微秒到 2 秒之间的持续时间。注意 ERSTL 设置为 0 对 NRST 脉冲表示持续一个 2 个周期时间。

此功能部件允许复位控制器形成 NRST 引脚电平，以此方式来保证 NRST 口线被驱使为低电平，为可能连接在系统复位上的外部设备提供符合的复位时间。

作为在可备份的 RSTC_MR 内的域，对于需要一个比慢时钟振荡器长的启动时间的设备，此域可以被用于形成系统上电复位。

图 14-4. Brownout 管理器



www.mcuzone.com

14.3.3 复位状态

复位状态管理器处理不同的复位源并产生内部复位信号。它报告在状态寄存器(RSTC_SR)的 RSTTYP 域中的复位状态。RSTTYP 域的更新在处理器复位释放时被执行。

14.3.3.1 通用复位

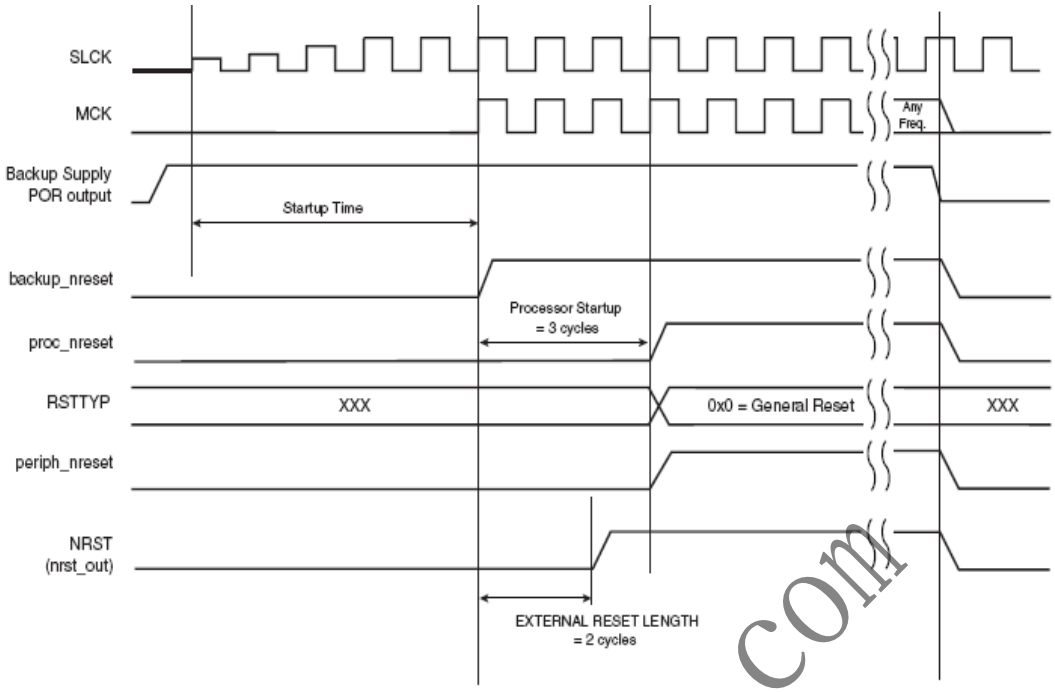
当 VDDBU 上电时产生一个通用复位。备用电源 POR 单元输出上升，用一个启动计数器过滤，此计数器由慢时钟操作。此计数器的目的是确保慢时钟晶振在启动设备前是稳定的。启动时间的长度严格编码到服从慢时钟振荡器启动时间。

此时间后，处理器时钟被使能为慢时钟，并且所有其他信号仍然在 2 个周期对适当的处理器和逻辑复位有效。接着，所有的复位信号被释放并且 RSTC_SR 中的 RSTTYP 域报告一个通用的复位。随着 RSTC_MR 复位，NRST 口线在 backup_nreset 后上升 2 周期，如 ERSTL 在值 0x0 缺省。

当 VDDBU 被备用电源 POR 单元检测为低，所有复位信号立即被激活，即使主电源 POR 单元未报告主电源掉电。

图 14-3 展示了通用复位怎样影响复位信号。

图 14-5 通用复位状态



14.3.3.2 唤醒复位

唤醒复位在主电源掉电时产生。当主电源 POR 输出有效，除了 backup_nreset 所有的复位信号有效。当主电源上电，POR 输出在慢时钟上重新同步化。处理器时钟接着在 2 个慢时钟周期内重新使能，依赖于 ARM 处理器的要求。

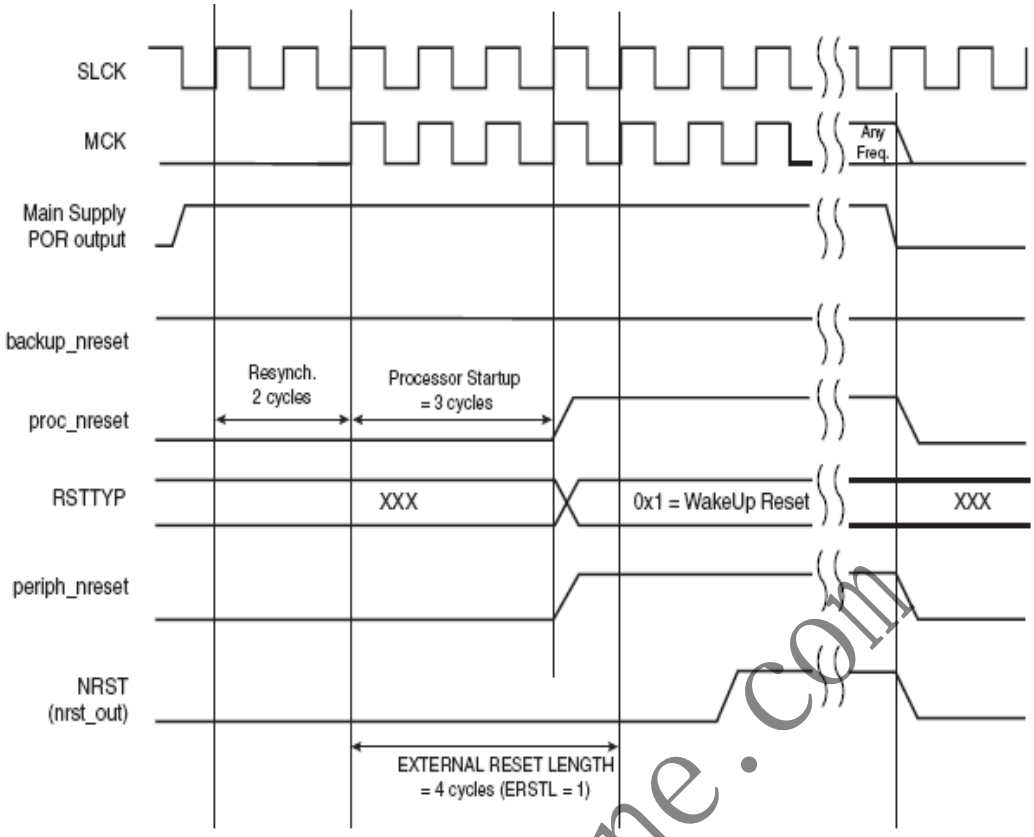
在延迟结束时，处理器和其他复位信号升高。RSTC_SR 中的 RSTTYP 域被更新为报告一个唤醒复位。

“nrst_out”仍然对 EXTERNAL_RESET_LENGTH 周期有效。RSTC_MR 被备份，编程了的周期数是可应用的。

当主电源被检测到下降（falling），立即使复位信号有效。此转换和主电源 POR 输出同步。

图 14-6 唤醒状态

www.mcuzone.com



14.3.3.3 用户复位

当在 NRST 引脚上检测到一个低电平时，并且 RSTC_MR 中的 URSTEN 位是 1，则进入用户复位。NRST 输入信号和 SLCK 同步以确保合适的系统操作。

只要在 NRST 上检测到一个低电平，用户复位就会被送入。处理器复位和外设复位有效。

用户复位在一个双周期同步时间和一个三周期处理器启动时间后，当 NRST 上升时消失。

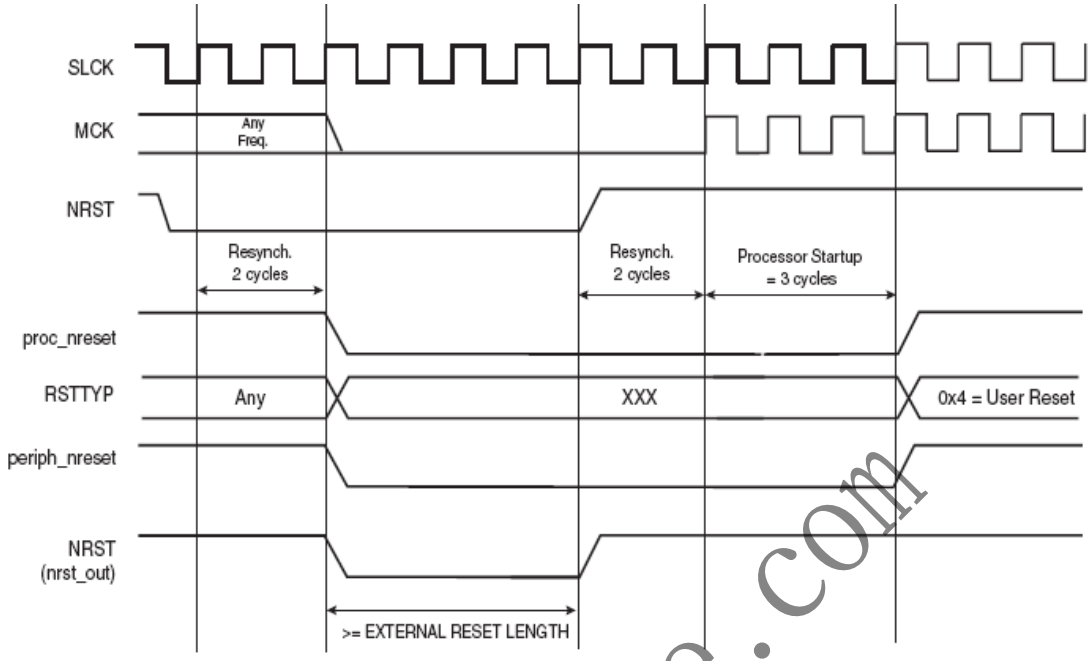
当处理器复位信号被释放，状态寄存器的 RSTTYP 域被用 0x4 的值装载，表示一个用户复位。

NRST 管理器保证 NRST 块对 EXTERNAL_RESET_LENGTH 慢时钟周期有效，当在 ERSTL 域中编程。

然而，如果 NRST 在 EXTERNAL_RESET_LENGTH 后未上升，比如是因为被外部驱使为低电平，内部复位仍然有效直到 NRSST 确切的上升。

www.mcuzone.com

图 14-8 用户复位状态



14.3.3.4 软件复位

复位控制器提供若干命令用于激活不同的复位信号。这些命令通过向控制寄存器(RSTC_CR)中的以下位来执行：

- PROCRST：写 PROCRST 为 1 来复位处理器和看门狗定时器。
- PERRST：写 PERRST 为 1 来复位所有的嵌入式外设，包括存储器系统，并且，特别是重映射命令。外设复位通常用作调试目的。
- EXTRST：写 EXTRST 为 1 使得在被模式寄存器（RSTC_MR）中的 ERSTL 域定义的一个时间里 NRST 引脚为低电平。

如果这些位中的至少一个通过软件被置位，就会进入软件复位。所有这些命令可以被独立或同时执行。软件复位持续 2 个时钟周期。

只要寄存器一被写入，内部复位信号就被激活。这在主控时钟（MCK）上可以检测到。当软件复位消失他们被释放，就是说，同步到 SLCK。

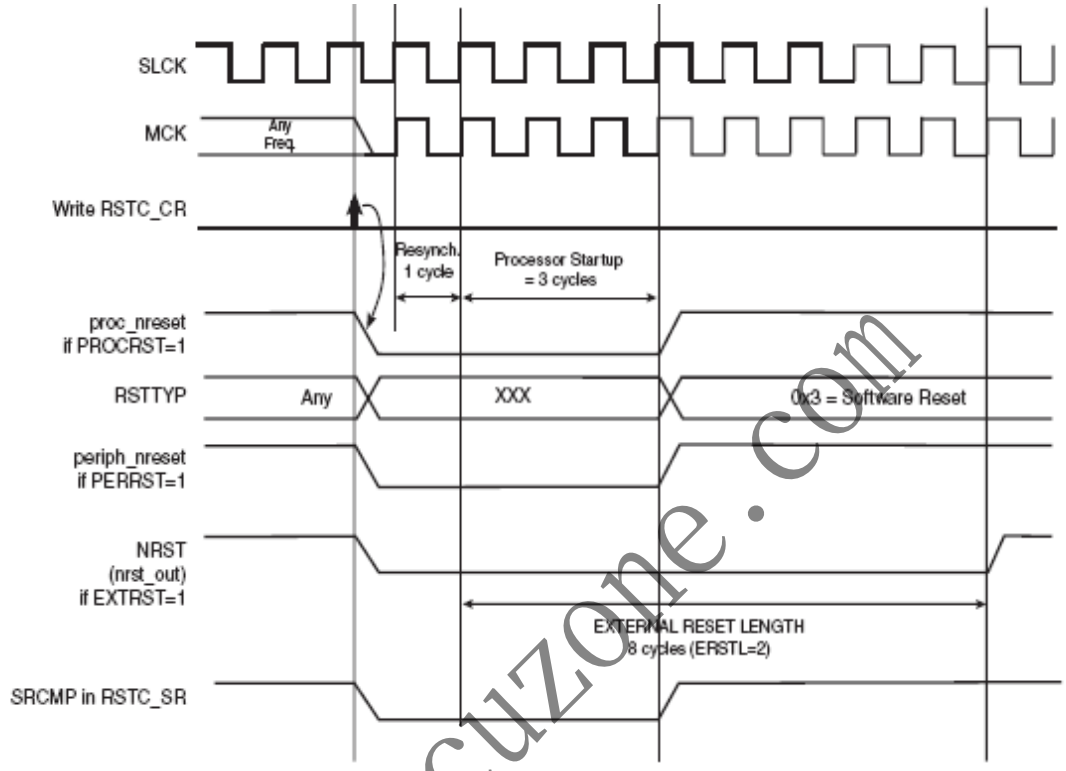
如果 EXTRST 置位，nrst_out 信号的激活取决于 ERSTL 域的编程。然而在 NRST 上的下降沿不会导致一个用户复位。

仅当 PROCRST 位被置位，复位控制器才报告在状态寄存器的 RSTTYP 域中的软件状态。其他复位在 RSTTYP 中不报告。

只要一个软件操作被检测到，则 SRCMP 位（进行中的软件复位命令）在状态寄存器（RSTC_SR）中被置位。只要软件复位一消失，SRCMP 就被清零。在 SRCMP 置位期间，没有其他软件复位会被执行，；并且向 RSTC_CR 中写任何值都无效。

www.mcuzone.com

图 14-9　软件复位



14.3.3.5　看门狗复位

当看门狗产生超时，则进入看门狗复位。此状态持续 2 个时钟周期。

当在看门狗复位时，复位信号的有效依赖于 WDT_MR 中的 WDRPROC 位：

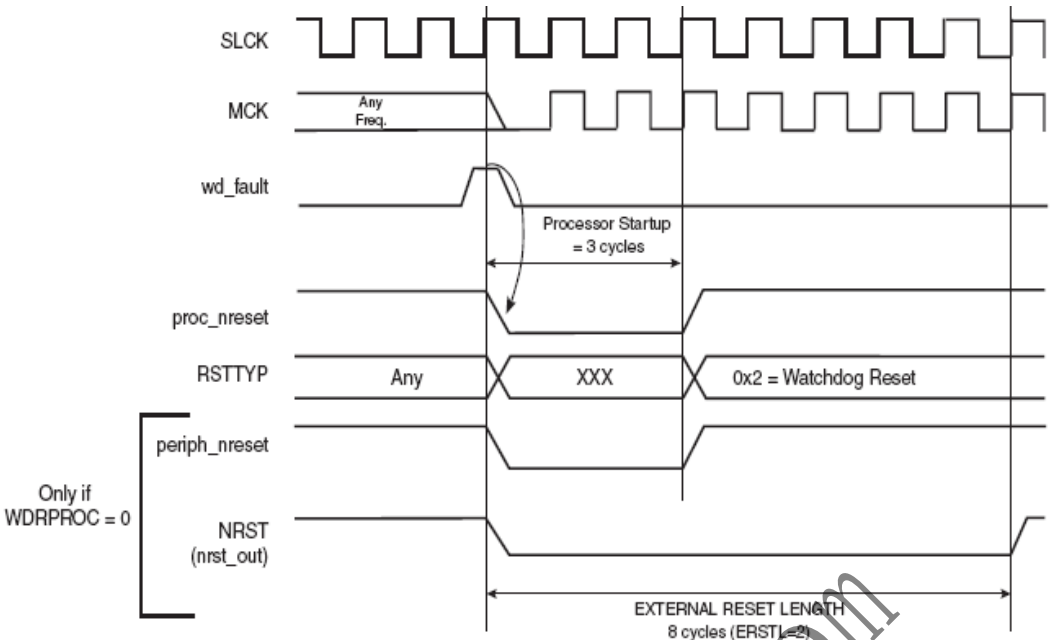
- 如果 WDRPROC 是 0，处理器复位和外设复位有效。NRST 口线也有效，依赖于 ERSTL 域的编程。然而，NRST 上的低电平并不导致一个用户复位状态。
- 如果 WDRPROC 是 1，仅处理器复位有效。

看门狗定时器被 proc_nreset 信号复位。因为如果 WDRSTEN 被置位看门狗超时总是导致一个处理器复位，所以看门狗定时器总是在一个看门狗复位后复位，并且看门狗被缺省使能并设置为最大周期。

当 WDT_MR 位中的 WDRSTEN 被复位，看门狗超时在复位控制器上无影响。

图 14-10 看门狗复位

www.mcuzone.com



14.3.4 复位状态优先级

复位状态管理器在不同的复位源间管理以下优先级，以降序给出：

- 备份复位
- 唤醒复位
- 看门狗复位
- 软件复位
- 用户复位

特殊情况如下：

- 当在用户复位：
  - 看门狗事件不可能发生，因为看门狗定时器正被 proc_nreset 信号复位。
  - 软件复位不可能发生，因为处理器复位正被激活。
- 当在软件复位：
  - 看门狗事件比当前状态优先级高。
  - NRST 无效
- 当在看门狗复位：
  - 处理器复位被激活并因此软件复位不可能被编程。
  - 用户复位不可能输入。

14.3.5 复位控制器状态寄存器

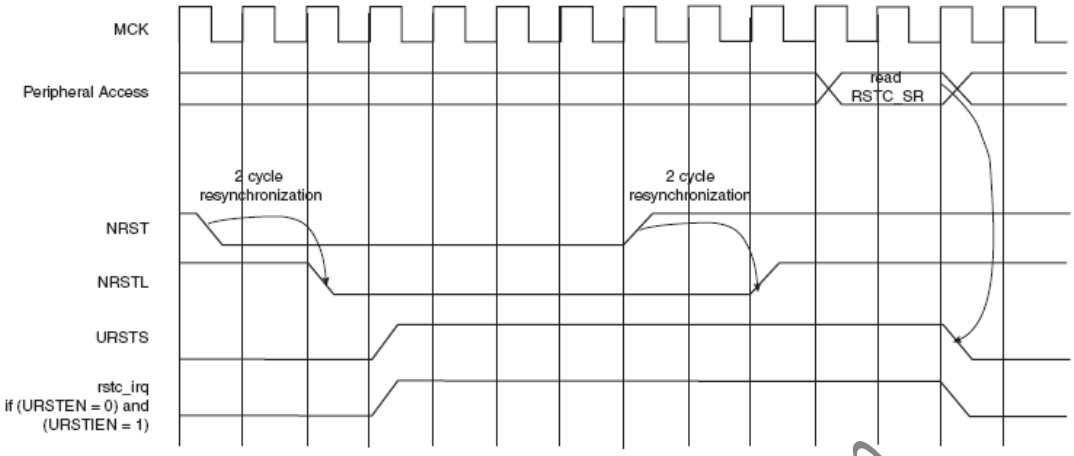
复位控制器状态寄存器(RSTC_SR)提供以下若干状态域：

- RSTTYP 域：此域给出了最后复位类型，如前段解释。
- SRCMP 位：此域表示软件复位命令在进进行中并且直到当前复位结束没有更远的软件复位应当被执行。此位在当前软件复位结束时自动清零。
- NRSTL 位：状态寄存器的 NRSTL 位给出了 NRST 引脚在每个 MCK 上升边沿采样的电平。
- URSTS 位：NRST 引脚 high-to-low 的转变置位 RSTC_SR 寄存器的 URSTS 位。此转变还可以在主控时钟（MCK）上升沿被检测到（见图 14-11）。如果用户复位被使无效并且如果中断被 RSTC_MR 寄存器中的 URSTIEN 位

www.mcuzone.com

使能，则 URSTS 位触发一个中断。读 RSTC_SR 状态寄存器复位 URSTS 位并清零中断。

图 14-11 复位控制器状态和中断



14.4 复位控制器(RSTC)用户接口

表 14-1 复位控制器（RSTC）寄存器映射

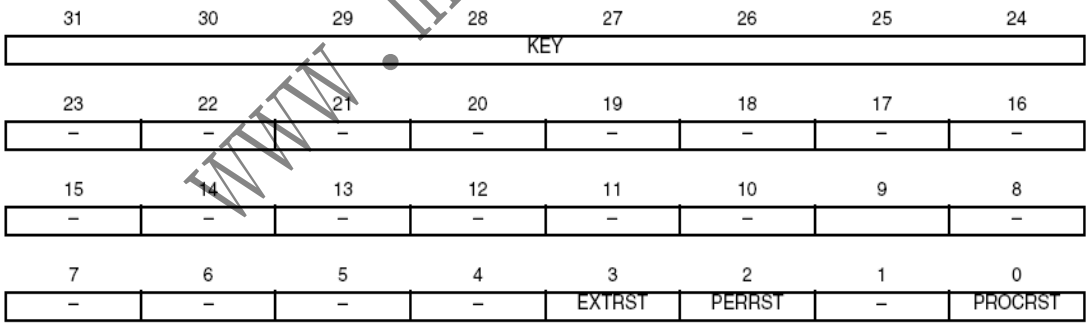
偏移量	寄存器	名称	访问	复位值	备份复位值
0x00	控制寄存器	RSTC_CR	只写	-	
0x04	状态寄存器	RSTC_SR	只读	0x0000_0001	0x0000_0000
0x08	模式寄存器	RSTC_MR	读/写	-	0x0000_0000

注意：1. RSTC_SR 的复位值报告为一个通用复位或一个唤醒复位，取决于最后上升的电源。

14.4.1 复位控制器控制寄存器

寄存器名称：RSTC_CR

访问类型：只写



- PROCRST：处理器复位

0=无效。

1=如果 KEY 正确，复位处理器。

- OERRST：外设复位

0 ＝无效。

1 ＝如果 KEY 正确，复位外设。

- EXTRST：外部复位

0=无效

1=如果 KEY 正确，NRST 引脚有效。

- KEY：口令

www.mcuzone.com

应该写入值 0xA5。在此域写任何其他值将退出操作。

14.4.2 复位控制器状态寄存器

寄存器名称：RSTC_SR

访问类型：只读

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

- URSTS：用户复位状态

0 ＝从最近一次读 RST_SR 到现在，在 NRST 上无 high-to-low 边沿发生。

1=从最近一次读 RST_SR 到现在 NRST 上至少有一次 high-to-low 转变被检测到。

- RSTTYP：复位类型

报告最近一次处理器复位的原因。读此 RSTC_SR 不复位此域。

表 1

RSTTYP			复位类型	备注
0	0	0	通用复位	VDDCORE 和 VDDBU 两个都上升
0	0	1	唤醒复位	VDDCORE 上升
0	1	0	看门狗复位	看门狗超时发生
0	1	1	软件复位	有软件命令处理器复位
1	0	0	用户复位	NRST 引脚检测到低电平

- NRSTL：NRST 引脚电平

在主控时钟(MCK)同步下的 NRST 引脚电平

- SRCMP：进程中的软件复位命令

0=无软件命令正被复位控制器执行。复位控制器等待软件命令。

1=一个软件复位命令正被复位控制器执行。复位控制器忙。

14.4.3 复位控制器模式寄存器

寄存器名称：RSTC_MR

访问类型：读/写

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

- URSTEN：用户复位使能

0=NRST 引脚上低电平的检测不会产生一个用户复位。
1=NRST 引脚上低电平的检测触发一个用户复位。

- URSTIEN：用户复位中断使能

0=RSTC_SR 中的 USRTS 位为 1 在 rstc_irq 上无效。
1=如果 URSTEN=0，RSTC_SR 中的 USRTS 位为 1 激活 rstc_irq。

- ERSTL：外部复位长度

此域定义外部复位长度。外部复位在$2^{(ERSTL+1)}$个慢时钟周期的时间里被激活。这允许有效延迟时间编程为 60 微秒和 2 秒之间。

- KEY：口令

应该写入值 0xA5。在此域写任何值都将退出操作。

www.mcuzone.com

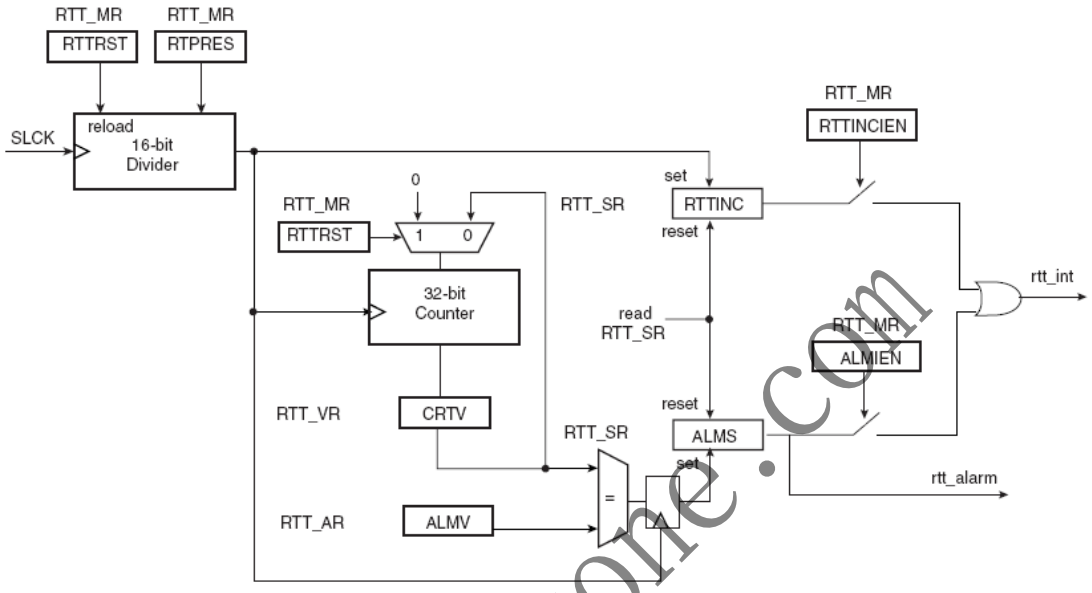
15 实时定时器(RTT)

15.1 概论

实时定时器围绕一个 32 位计数器构造并用于计数经历时间。它产生一个周期中断和/或根据一个编程了的值触发一个定时警报。

15.2 方块图

图 15-1 实时定时器



15.3 功能描述

实时定时器被用于计数经历时间。基于一个 32 位计数器构建，此计数器由一个可编程的 16 位分配器分频后的慢时钟提供时钟。其值可以在 RTT 模式寄存器（RTT_MR）的 RTPRES 域中被编程。

将 RTPRES 编程为 0x00008000 相当于用一个 1Hz 信号(如果慢时钟是 32.738Hz) 馈送实时计数器。32 位计数器可以加到2^{32}秒，相当于 136 多年，然后转到 0。

实时定时器还可以被用作一个带较低时基的自由运行定时器。通过将 RTPRES 编程为 3 可以获得最好的精度。将 RTPRES 编程为 1 或 2 也是可以的，但有可能导致丢失状态事件，因为状态寄存器在读操作 2 个慢时钟周期后被清零。因此如果 RTT 被配置来触发一个中断，此中断在读 RTT_SR 后 2 个慢时钟周期内产生。为了阻止若干中断处理程序的执行，中断必须在中断处理程序中禁用并当状态寄存器清零时重新使能。

实时定时器值（CRTV）可以在寄存器 RTT_VR（实时值寄存器）中任何时间被读取。由于此值可以从主控时钟异步更新，因此建议连续读两次该寄存器，如果活得相同的值再返回，以提高返回值的准确度。

计数器的当前值和报警寄存器 RTT_AR（实时报警寄存器）中写入的值作比较。如果计数器值符合报警值，则 RTT_SR 中的 SLM 位被置位。复位后，

www.mcuzone.com

报警信号寄存器被置位到其最大值，相当于 0xFFFF_FFFF。

RTT_SR 中的 RTTINC 位在实时计数器每次递增后被置位。此位可被用于开启一个周期中断，当 RTPRES 编程为 0x8000，且慢时钟是 32.768Hz 的话，周期为 1 秒。

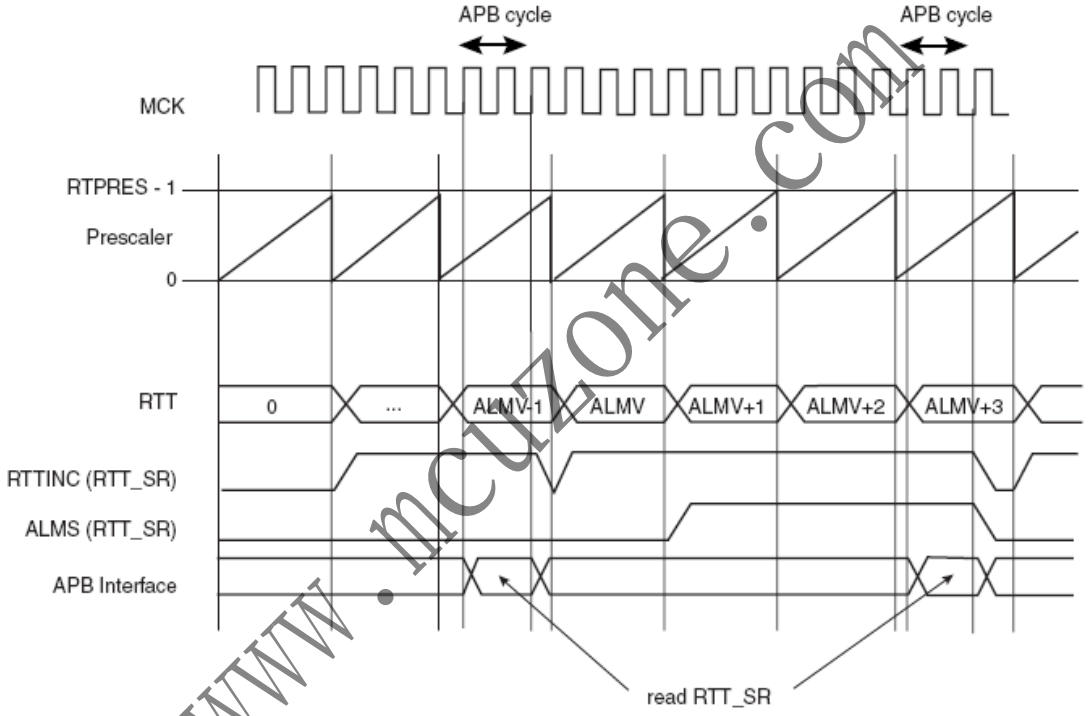
读 RTT_SR 状态寄存器复位 RTTINC 和 ALMS 域。

写 RTT_MR 中的 RTTRST 位会立即用新的编程值重装载和重开启时钟分频器。同时也复位 32 位计数器。

注意：由于慢时钟（SCLK）和系统时钟(MCK)间的不同步：

1. 在 RTT_MR 寄存器中的 RTTRST 位写以后，计数器的重开启和 RTT_VR 当前值寄存器的复位仅在 2 个慢时钟周期后是有效的。
2. 状态寄存器标志位在读 RTT_SR（状态寄存器）后 2 个慢时钟周期复位。

图 15-2 RTT 计数



www.mcuzone.com

15.4 实时定时器(RTT)用户接口

15.4.1 寄存器映射

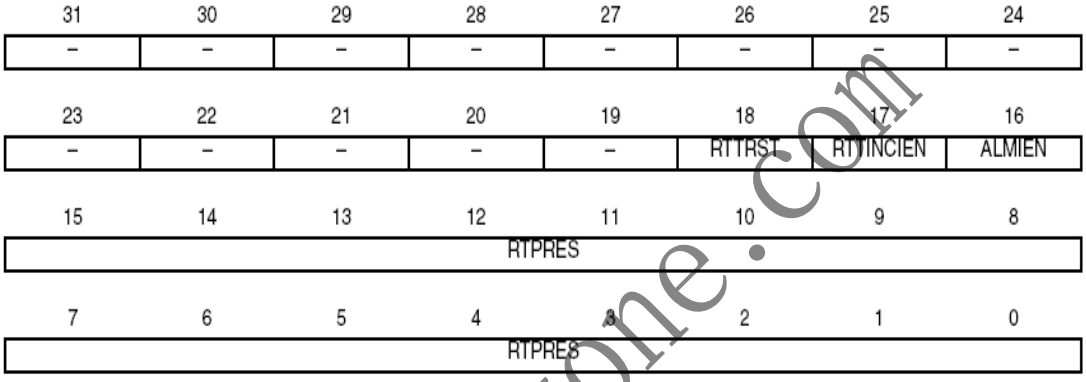
表 15-1 实时定时器寄存器映射

偏移量	寄存器	名称	访问	复位值
0x00	模式寄存器	RTT_MR	读/写	0x0000_8000
0x04	报警寄存器	RTT_AR	读/写	0xFFFF_FFFF
0x08	当前值寄存器	RTT_VR	只读	0x0000_0000
0x0C	状态寄存器	RTT_SR	只读	0x0000_0000

15.4.2 实时定时器模式寄存器

寄存器名称：RTT_MR

访问类型：读/写



- RTPRES：实时定时器预分频值

定义实时计数器递增需要的 SLCK 周期数。RTPTES 被定义如下：

RTPRES=0：预定标器周期等于 2^{16}

RTPRES≠0：预定标器周期等于 RTPRES。

- ALMIEN：报警中断使能

0=RTT_SR 中的 ALMS 位在发生中断时无效。

1=RTT_SR 中的 ALMS 位激活中断。

- RTTINCIEN：实时定时器递增中断使能

0=RTT_SR 中的 RTTINC 位发生中断时无效。

1=RTT_SR 中的 RTTINC 激活中断。

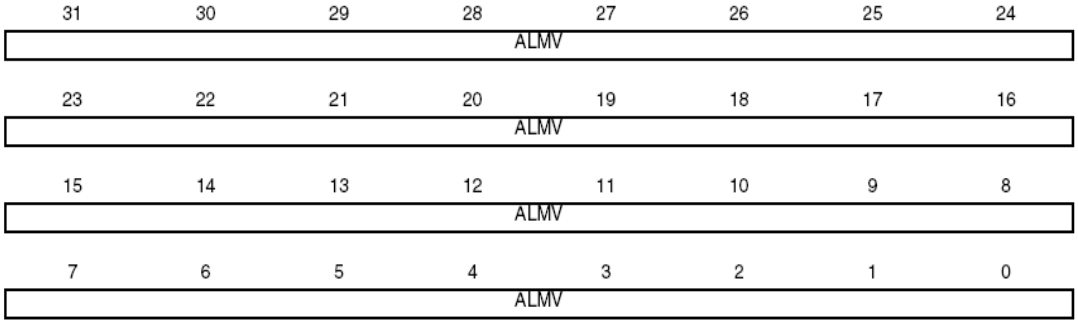
- RTTRST：实时定时器重启
- 1=用新编程值重装载并重启动时钟分频器。并复位 32 位计数器。

www.mcuzone.com

15.4.3 实时定时器报警寄存器

寄存器名称：RTT_AR

访问类型：读/写



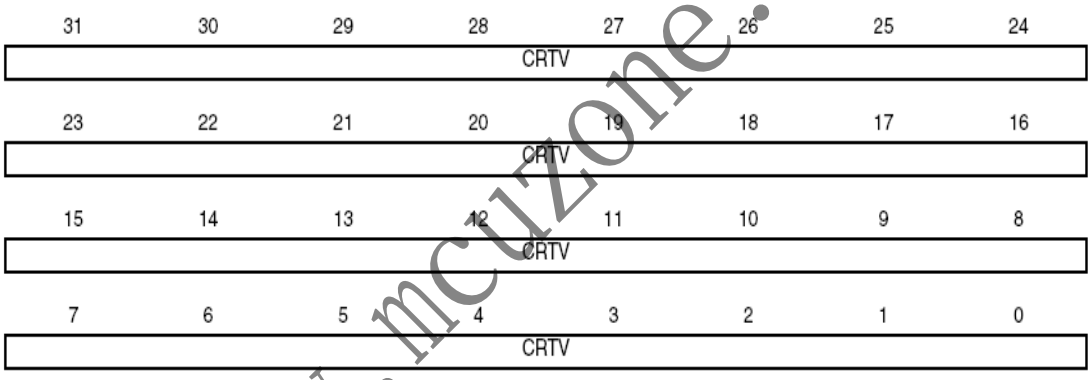
- ALMV：报警值

定义和实时定时器比较的报警值

15.4.4 实时定时器值寄存器

寄存器名称：RTT_VR

访问类型：只读



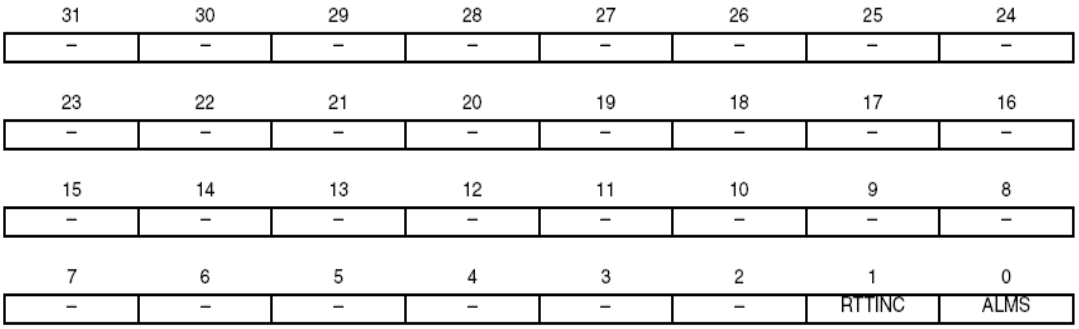
- CRTV：当前实时值

返回实时定时器的当前值。

15.4.5 实时定时器状态寄存器

寄存器名称：RTT_SR

访问类型：只读



- ALMS：实时报警状态

www.mcuzone.com

0=从最后一次读 RTT_SR 为止，实时报警未产生。

1=从最后一次读 RTT_SR 为止，产生实时报警。

- RTTINC：实时定时器递增

0 ＝从最后一次读 RTT_SR 为止，实时定时器未递增。

1 ＝从最后一次读 RTT_SR 为止，实时定时器递增。

www.mcuzone.com

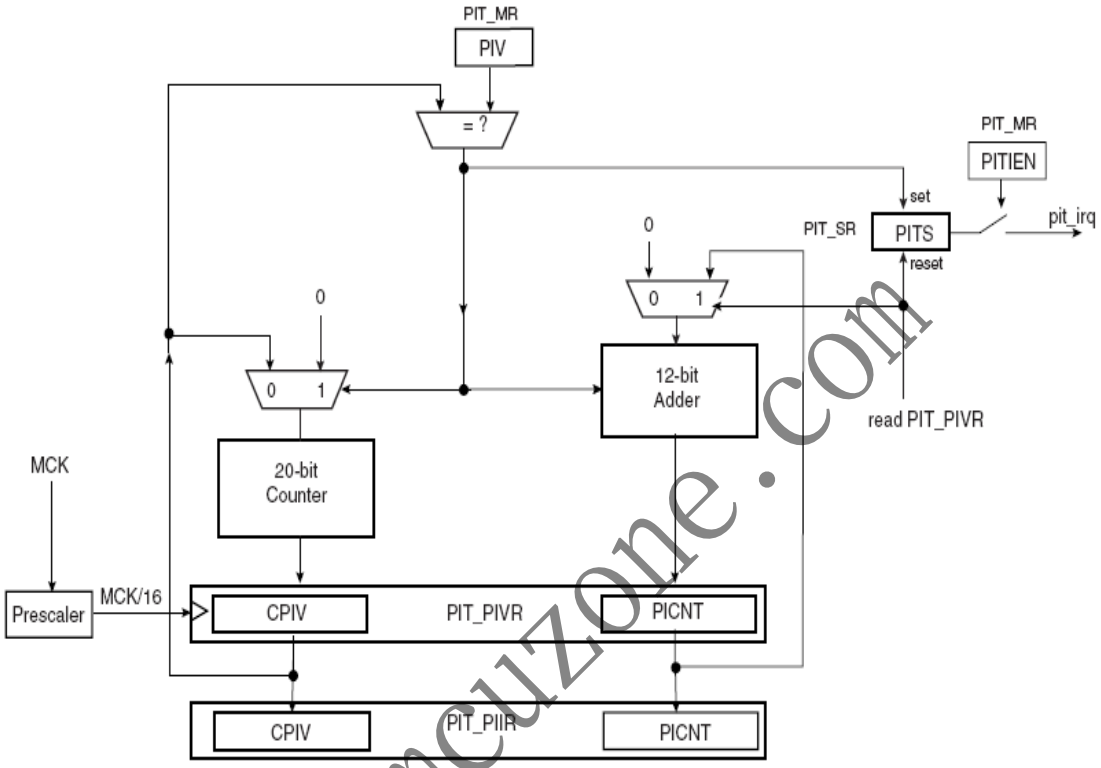
16 周期间断定时器(PIT)

16.1 概论

周期间断定时器（PIT）提供操作系统使用的程序调度中断。被设计来提供最大的准确度和最有效的管理，甚至对响应时间很长的系统。

16.2 方框图

图 16-1 周期间断定时器



16.3 功能描述

周期间断定时器为操作系统提供周期间断。

PIT 提供一个可编程的溢出计数器和一个 reset-on-read 功能部件。围绕两个计数器构建：一个 20 位 CPIV 计数器和一个 12 位 PICNT 计数器。两个计数器都工作在主控时钟的十六分之一。

第一个 20 位 CPIV 计数器从 0 递增到一个设置在模式寄存器（PIT_MR）中的 PIV 域的可编程溢出值。当计数器 CPIV 达到此值，计数器复位到 0 并递增周期间断计数器，PICNT。如果中断被使能（PIT_MR 中的 PITIEN），状态寄存器(PIT_SR)中的状态位 PIT 置位并触发一个中断。

在 PIT_MR 中写一个新 PIV 值不复位/重启计数器。

当 CPIV 和 PICNT 值通过读周期间断值寄存器（PIT_PIVR）得到，溢出计数器（PICNT）复位并且 PITS 清零，以此方式响应中断。PICNT 的值给出了周期间断从最后一次读 PIT_PIVR 到当前经过的周期数。

当 CPIV 和 PICNT 值通过读周期间断映射寄存器（PIT_PIIR）获得，则不会

www.mcuzone.com

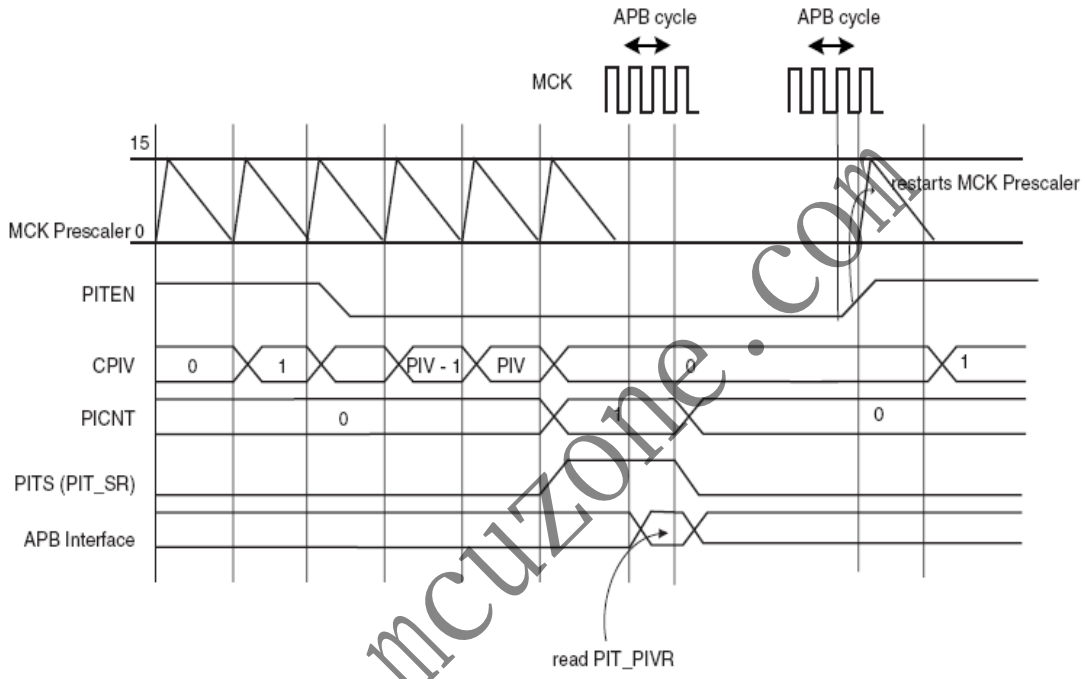
影响计数器 CPIV 和 PICNT，也不会影响 PITS 位。例如，一个分析器可以读没有清零任何未决的中断，然而一个定时器中断通过读 PIT_PIVR 清零中断。

PIT 可能被 PIT_MR 寄存器中的 PITEN 位使能或禁用（复位时禁止）。

当 CPIV 值是 0，仅 PITEN 位变成有效。图 16-2 说明了 PIT 计数。PIT 位复位（PITEN=0）后，CPIV 继续计数直到达到 PIV 值，并接着复位。只有 PITEN 再次设置，PIT 才重启计数。

当内核进入调试状态，则 PIT 停止。

图 16-2 用 PITEN 使能/使失效 PIT



www.mcuzone.com

16.4 周期间接定时器（PIT）用户接口

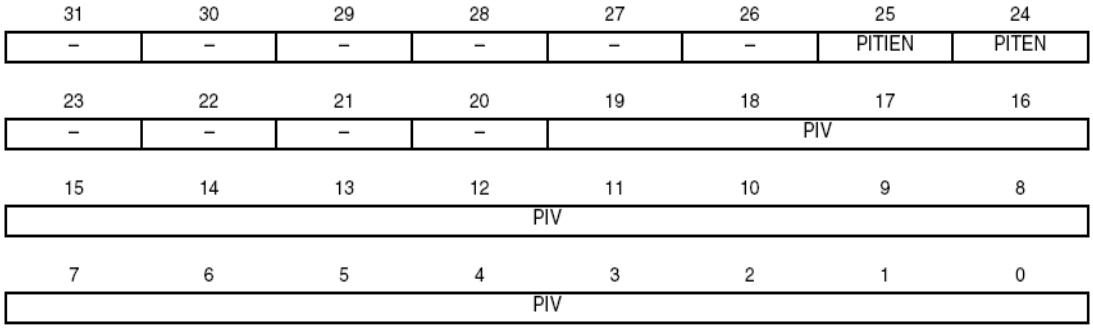
表 16-1 周期间断定时器(PIT)寄存器映射

偏移量	寄存器	名称	访问	复位值
0x00	模式寄存器	PIT_MR	读/写	0x000F_FFFF
0x04	状态寄存器	PIT_SR	只读	0x0000_0000
0x08	周期间断值寄存器	PIT_PIVR	只读	0x0000_0000
0x0C	周期间断映射寄存器	PIT_PIIR	只读	0x0000_0000

16.4.1 周期间断定时器模式寄存器

寄存器名称：PIT_MR

访问类型：读/写



- PIV：周期间断值

定义和周期间断定时器(CPIV)的主要的 20 位计数器比较的值。周期等于（PIV+1）。

- PITEN：周期间断定时器使能

0=当计数达到 PIV 值，周期间断定时器被禁用。

1=周期间断定时器使能。

- PITIEN：周期间断定时器中断使能

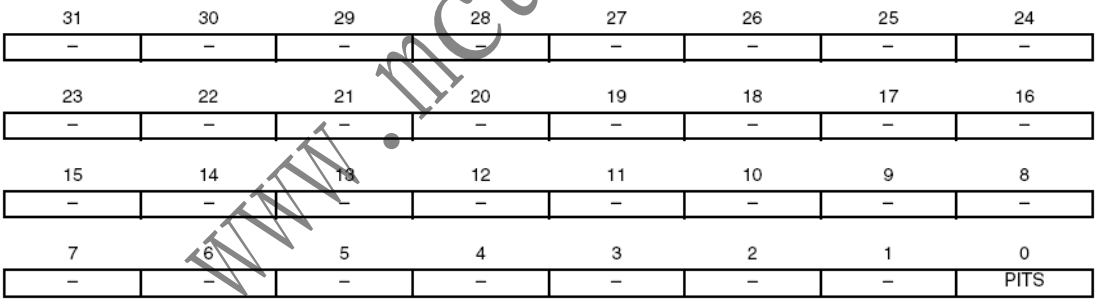
0=PIT_SR 中的 PITS 位对中断无效

1=PIT_SR 中的 PITS 位激活中断

16.4.2 周期间断定时器状态寄存器

寄存器名称：PIT_SR

访问类型：只读



- PITS：周期间断定时器状态

0=周期间断定时器从最近一次读 PIT_PIVR 到当前未计数到达 CPIV。

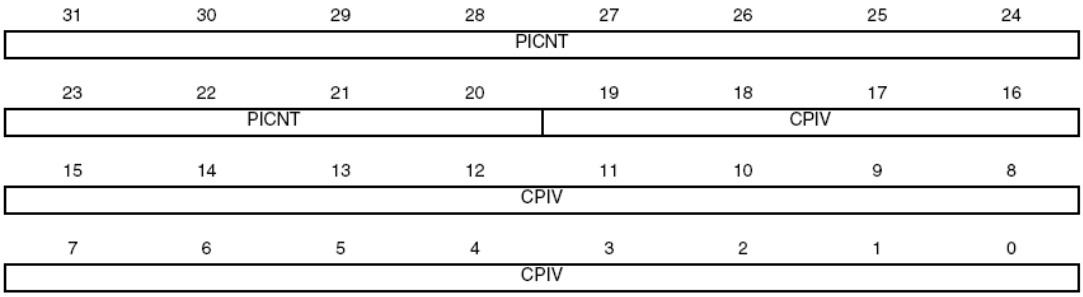
1=周期间断定时器从最近一次读 PIT_PIVR 到当前已计数到达 CPIV。

16.4.3 周期间断定时器值寄存器

寄存器名称：PIT_PIVR

访问类型：只读

www.mcuzone.com



读此寄存器清零 PIT_SR 中的 PITS 位。

- CPIV：当前周期间断值

返回周期间断定时器的当前值。

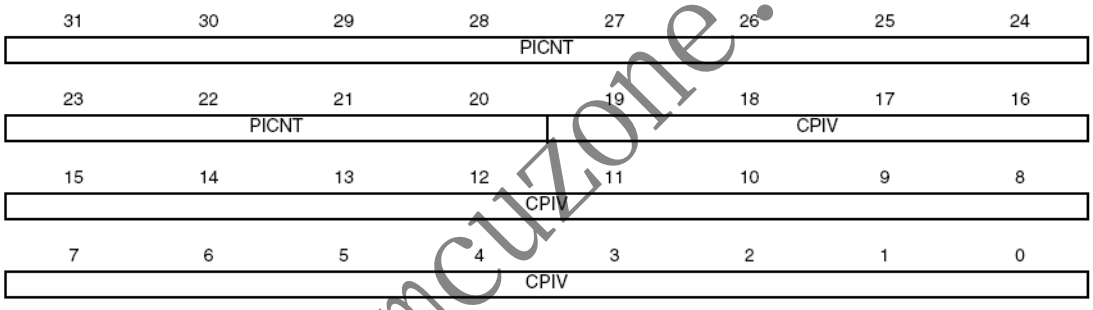
- PICNT：周期间断计数器

返回从最近一次读 PIT_PIVR 到当前周期间断发生的次数。

16.4.4 周期间断定时器映射寄存器

寄存器名称：PIT_PIIR

访问类型：只读



- CPIV：当前周期间断值

返回周期间断定时器的当前值

- PICNT：周期间断计数器

返回从最近一次读 PIT_PIVR 到当前周期间断发生的次数。

www.mcuzone.com

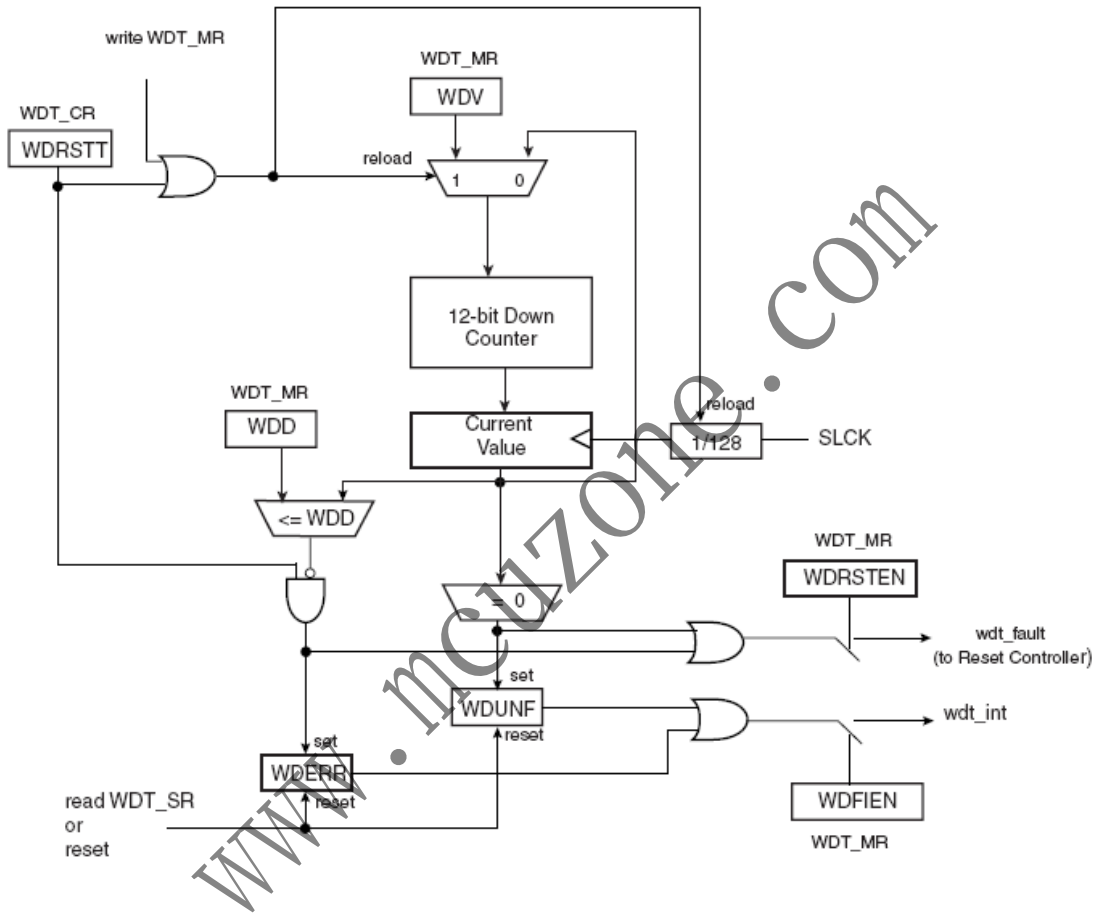
17 看门狗定时器（WDT）

17.1 概论

如果用户程序进入死锁状态，看门狗定时器可以防止系统锁定（lock-up）。它以一个 12 位的递减计数器组成。此计数器允许一个达 16 秒（慢时钟 32.768KHz）的看门狗周期。它还可仅产生一个通用复位或一个处理器复位。另外，当处理器在调试模式或空闲模式时可以停止看门狗。

17.2 方框图

图 17-1 看门狗定时器方块图



www.mcuzone.com

17.3 功能描述

如果用户程序进入死锁状态，看门狗定时器可以防止系统锁定（lock-up）。看门狗由 VDDCORE 供电。当处理器复位时将加载初始值。

看门狗围绕一个 12 位递减计数器构造，此计数器用定义在模式寄存器（WDT_MR）的 WV 域中的值装载。看门狗定时器用慢时钟除以 128 来作为时钟，最大看门狗周期为 16 秒（用一个典型的 32.768KHz 的慢时钟）。处理器复位后，WV 的值为 0xFFF，相当于用外部复位产生使能（一个备份复位后，WDRSTEN 域为 1）的最大值。这就意味着看门狗复位后缺省就运行，例如上电复位。如果用户不想用它就必须禁用它（通过置位 WDT_MR 中的 WDDIS 位）或必须重编程它来满足走大看门狗周期的应用需求。

看门狗模式寄存器（WDT_MR）仅可以被写一次。仅能用一个处理器复位复

位。写 WDT_MR 寄存器将用新编程的模式参数重新装载定时器。

普通操作中，用户通过一定时间间隔向控制寄存器的 WDRSTT 位写 1 ，在定时器下溢产生前重新装载看门狗。看门狗计数器接着立即从 WDT_MR 重装载并重启，慢时钟 128 分频器被复位和重启。WDT_CR 寄存器被写保护。如果没有适当的硬编码密钥（hard-coded key），写 WDT_CR 则无效。如果产生一个下溢，如果 WDRSTEN 位在模式寄存器被置位则复位控制器的“wdt_fault”信号被激活。此外，WDUNF 位在看门狗状态寄存器(WDT_SR)中被置位。

为阻止软件死锁时连续的触发看门狗，当看门狗计数器在 0 和 WDD 间的窗口，看门狗必须重装载，WDD 在看门狗模式寄存器 WDT_MR 中定义。

当看门狗计数器在 WDV 和 WDD 间任何重启看门狗的尝试都导致看门狗错误，即时看门狗在使无效状态。WDERR 位在 WDT_SR 中刷新，到复位控制器的“wdt_fault”信号是有效的。

注意此功能部件可以通过编程一个大于或等于 WDV 值的 WDD 值来禁用。在这样的配置中，重启看门狗定时器在整个[0;WDV]范围内是允许的并且不会出错。这是复位后的缺省配置（WDD 和 WDV 值相等）。

如果 WDFIEN 位在模式寄存器中置位，状态位 WDUNF（看门狗下溢）和 WDERR（看门狗错误）触发一个中断。如果 WDRSTEN 位置位，就像已在复位控制器编程器 Datasheet 中所解释的那样，则“wdt_fault”信号在复位控制器导致看门狗复位。此情况下，处理器和看门狗定时器被复位，WDERR 和 WDUNF 标志位被复位。

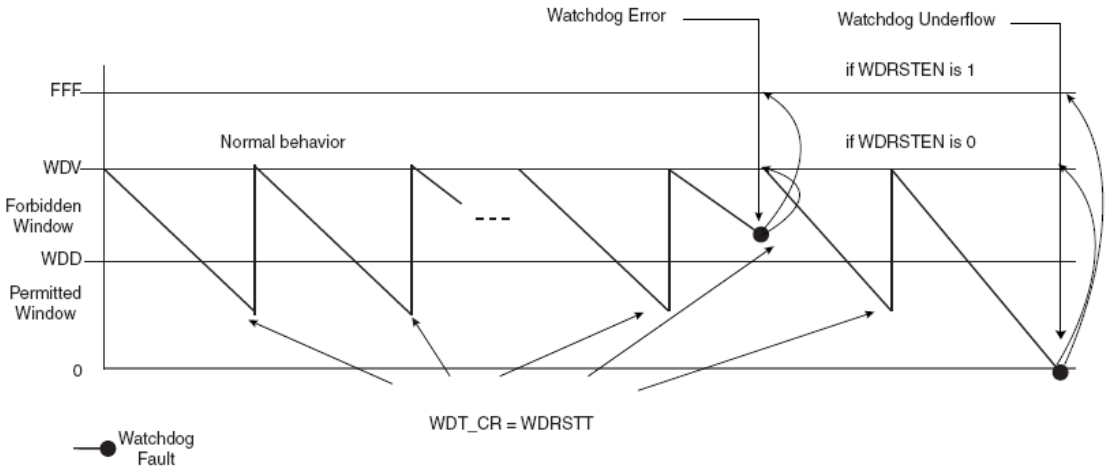
如果复位发生或如果 WDT_SR 已读，状态位被复位，中断被清零，并且到复位控制器的“wdt_fault”信号被使无效。

写 WDT_MR 重装载和重启递减计数器。

当处理器在调试状态或空闲模式，计数器可能被停止，取决于对 WDT_MR 中的 WDIDLEHLT 和 WDDBGHLR 位的编程值。

www.mcuzone.com

图 17-2 看门狗行为



www.mcuzone.com

17.4 看门狗定时器（WDT）用户接口

表 17-1 看门狗定时器(WDT)寄存器映射

偏移量	寄存器	名称	访问	复位值
0x00	控制寄存器	WDT_CR	只写	-
0x04	模式寄存器	WDT_MR	一次性读/写	0x3FFF_2FFF
0x08	状态寄存器	WDT_SR	只读	0x0000_0000

17.4.1 看门狗定时器控制寄存器

寄存器名称：WDT_CR

访问类型：只写

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

- WDRSTT：看门狗重启

0：无效。

1：重启看门狗。

- KEY：口令

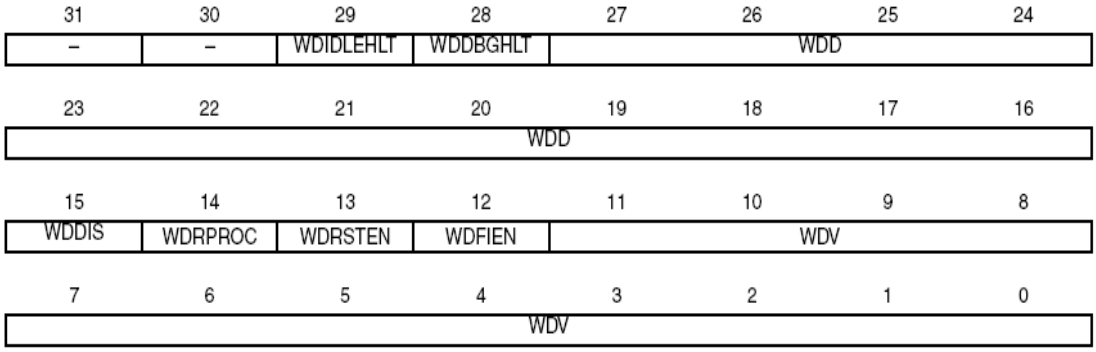
应写入 0xA5.在此域写任何其他值异常结束写操作。

www.mcuzone.com

17.4.2 看门狗定时器模式寄存器

寄存器名称：WDT_MR

访问类型：一次性读/写



- **WDV：看门狗计数器值**

定义装载于 12 位看门狗计数器中的值

- **WDFIEN：看门狗故障中断使能**

0：看门狗故障（下溢或错误）对中断无效

1：看门狗故障（下溢或错误）激活中断

- **WDRSTEN：看门狗复位使能**

0：看门狗故障（下溢或错误）对复位无效

1：看门狗故障（下溢或错误）触发一个看门狗复位

- **WDRPROC：看门狗复位处理器**

0：如果 WDRSTEN 是 1，看门狗故障（下溢或错误）激活所有的复位

1：如果 WDRSTEN 是 1 ，看门狗故障（下溢或错误）激活处理器复位

- **WDD：看门狗增量**

定义重装载看门狗定时器的允许范围

如果看门狗定时器值小于或等于 WDD，用 WDRSTT = 1 写 WDT_CR 重启定时器。

如果看门狗定时器值大于 WDD，用 WDRSTT = 1 写 WDT_CR 导致看门狗错误。

- **WDDBGHLT：看门狗调试停止**

0：当处理器处在调试状态，看门狗运行

1：当处理器处在调试状态，看门狗停止

- **WDIDLEHLT：看门狗空闲停止**

0：当系统处于空闲模式，看门狗运行

1：当系统处于空闲模式，看门狗停止

- **WDDIS：看门狗使无效**

0：使能看门狗定时器

1：是无效看门狗定时器

www.mcuzone.com

17.4.3 看门狗定时器状态寄存器

寄存器名称：WDT_SR

访问类型：只读

1 .

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

- **WDUNF：看门狗下溢**

0 ：从最近一次读 WDT_SR 到当前无看门狗下溢产生。

1：从最近一次读 WDT_SR 到当前至少一个看门狗下溢产生

- **WDERR：看门狗错误**

0 ：从最近一次读 WDT_SR 到当前无看门狗错误产生

1 ：从最近一次读 WDT_SR 到当前至少一个看门狗错误产生

www.mcuzone.com

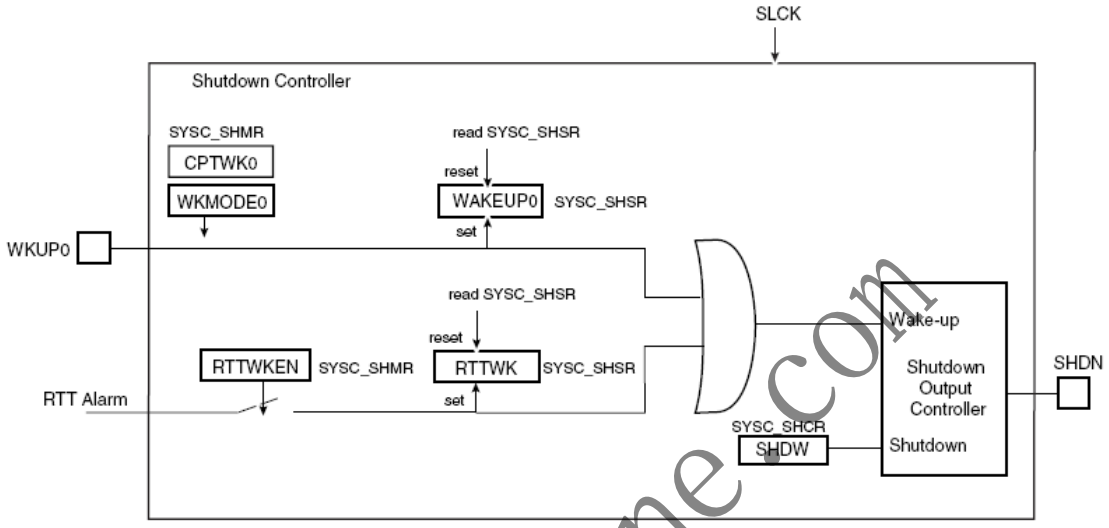
18 掉电控制器（SHDWC）

18.1 描述

掉电控制器控制电源 VDDIO 和 VDDCORE 和防抖动（debounced）输入口线上的唤醒检测。

18.2 方框图

图 18-1 掉电控制器方框图



18.3 I/O 口线描述

表 18-1 I/O 口线描述

名称	描述	类型
WKUP0	唤醒 0 输入	输入
SHDN	关机输出	输出

18.4 产品相关

18.4.1 电源管理

掉电控制器由慢时钟提供时钟。掉电管理控制器在关机控制器运行时无效。

18.5 功能描述

掉电控制器管理主电源。为实现此目的，其由 VDDBU 供电并且管理唤醒输入引脚和一个输出引脚，SHDN。

一个典型应用是把 SHDN 引脚连接到 DC/DC 转换器的输入来给系统提供主电源，特别是 VDDCORE 和/或 VDDIO。唤醒输入（WKUP0）连接到任何按钮或唤醒系统的信号。

软件可以通过向掉电控制寄存器（SHDW_CR）的 SHDW 位写 1 来控制 SHDN 引脚。向 SHDN_CR 写入后只需 2 个慢时钟周期即进入掉电。此寄存器受口令保护，所以必须有正确的口令才能向其写入值。如果正确写入，系统即进入掉电状态。

WKUP0 上的电平改变可以用来唤醒。唤醒在掉电模式寄存器（SHDN_MR）中进行配置。转换检测器可以被编程为检测正/负边沿或 WKUP0 上的任何电平改变。此检测也可以禁用。程序可通过定义 WKMODE0 来设置。

www.mcuzone.com

而且，防抖动电路可以编程为 WKUP0 所使用。防抖动电路过滤 WKUP0 上比编程在 HDW_MR 寄存器的 CPTWK0 中的 16SLCK 的周期要短的脉冲。如果某个引脚上的电平改变被检测到，则计数器启动。当计数器达到在对应的域中编程值，CPTWK0，则 SHDN 引脚被释放。如果在计数器到达相应值之前一个新的输入改变被检测到 ，则计数器被停止并清零。状态寄存器（SHDW_SR）的位 WAKEUP0 报告 WKUP0 上的编程事件的检测，并在读 SHDW_SR 后复位。

掉电控制器可以被编程为用 RTT 警报（RTT 警报上升沿的检测和 SLCK 同步）激活唤醒。这是通过写 SHDW_MR 寄存器的 RTTWKEN 域完成。当使能，RTT 警报状态报告在 SHDW_SR 状态寄存器的 RTTWK 域。当读 SHDW_SR 后复位。当使用 RTT 警报来唤醒系统，用户必须保证 RTT 警报状态标志在系统掉电前清零。否则，将没有状态标志位的上升沿会被检测到并导致唤醒失败。

www.mcuzone.com

18.6 掉电控制器(SHDWC)用户接口

18.6.1 寄存器映射

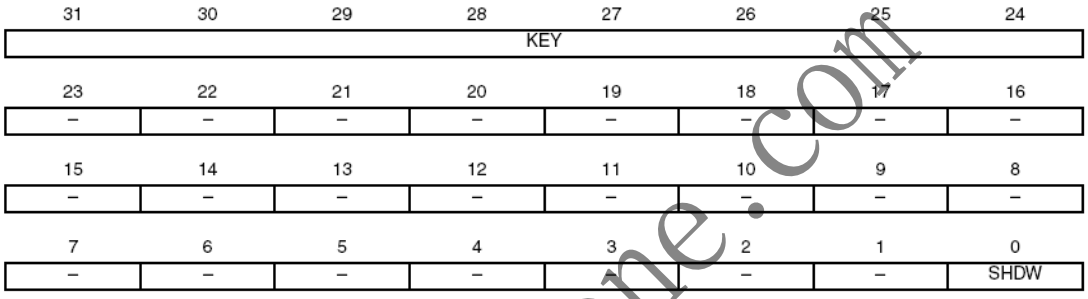
表 18-2 掉电控制器(SHDWC)寄存器

偏移量	寄存器	名称	访问	复位值
0x00	掉电控制寄存器	SHDW_CR	只写	-
0x04	掉电模式寄存器	SHDW_MR	读写	0x0000_0103
0x18	掉电状态寄存器	SHDW_SR	只读	0x0000_0000

18.6.2 掉电控制器寄存器

寄存器名称：SHDW_CR

访问类型：只写



- SHDW：掉电命令

0 ＝无效

1 ＝如果 KEY 正确，激活 SHDN 引脚。

- KEY：口令

应被写为 0xA5。在此域中写任何值将退出操作。

www.mcuzone.com

18.6.3 掉电模式寄存器

寄存器名称：SHDW_MR

访问类型：读/写

- WKMODE 0 ：唤醒模式 0

WKMODE[1： 0]		唤醒输入转换选择
0	0	无，在唤醒输入上不执行检测
0	1	低到高电平
1	0	高到低电平
1	1	上两列电平转变

- CPTWK0：Wake-up0 上的计数器

定义 16 慢时钟周期数，对应输入引脚上的电平检测在唤醒事件发生前应该持续。因为 WKUP0 的内部同步化，SHDN 引脚在 WKUP 上事件发生后经过（CPTWK*16+1）个慢时钟周期被释放。

- RTTWKEN：实时定时器唤醒使能

0 ＝RTT 警报信号在掉电控制器上无效

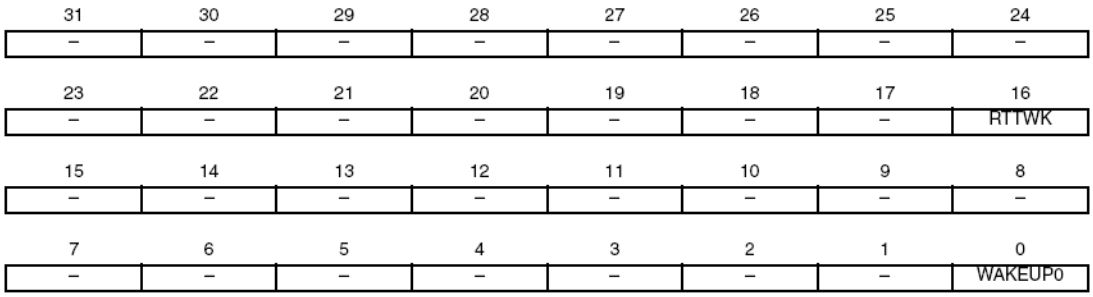
1=RTT 警报信号使得 SHDN 引脚释放。

www.mcuzone.com

18.6.4 掉电状态寄存器

寄存器名称：SHDW_SR

访问类型：只读



- WAKEUP0：Wake-up0 状态

0=从最近一次读 SHDW_SR 到当前对应的唤醒输入上无唤醒事件发生。

1=从最近一次读 SHDW_SR 到当前对应的唤醒输入上至少一次唤醒事件发生。

- RTTWK：实时定时器唤醒

0 ＝从最近一次读 SHDW_SR 到当前没有 RTT 唤醒事件发生。

1＝从最近一次读 SHDW_SR 到当前至少有一次 RTT 唤醒事件发生。

www.mcuzone.com

19 总线矩阵

19.1 概论

总线矩阵实现一个多层 AHB，基于 AHB-Lite 协议，并使能系统中多层 AHB 主控和从控间的并行访问路径，以此方式增加总带宽。总线矩阵互联了 5 个 AHB 主控和 5 个 AHB 从控。总线矩阵用户接口和 ARM 先进的外围总线兼容并且提供 5 个特殊功能寄存器(MATRIX_SFR)，这些寄存器允许总线矩阵支持特定应用部件。

19.2 存储器映射

总线矩阵给每个 AHB 主控接口提供一个解码器。解码器给每个 AHB 主控若干存储器映射。取决于产品，每个存储器区被指定为若干从控。当用不同的 AHB 从控（就是，外部 RAM，内部 ROM，内部 Flash,等）在相同的地址启动变得可能。

总线矩阵用户接口提供一个主控配置寄存器（MATRIX_MCFG）以对每个主控独立的执行一重映射操作。

19.3 特殊总线授权技术

总线矩阵提供一些推测性总线授权技术来提前预测一些主控的访问请求。此技术减小了初次访问的延迟时间。总线授权技术给每个从控设置一个缺省的主控。在当前访问的结尾，如果无其他请求待定，则从控仍然连接其关联的缺省主控。一个从控可以关联 3 种缺省的主控。无缺省主控，最近一次访问主控和固定缺省主控。

19.3.1 无缺省主控

当前访问的结尾，如果无其他请求待定，从控被从所有主控切断。无缺省主控适用于低功耗模式。

19.3.2 最近一次访问主控

当前访问的结尾，如果无其他请求待定，从控仍然连接最近执行访问请求的主控。

19.3.3 固定缺省主控

当前访问的结尾，如果无请求待定，从控仍然连接其固定缺省主控。不像最近访问主控，固定主控不改变 除非用户通过软件操作改变它。

从一种缺省主控变到另一种，总线矩阵用户接口提供 5 个从控配置寄存器，每个从控一个，给每个从控设置缺省主控。从控配置寄存器包含 2 个域：DEFMSTR_TYPE 和 FIXED_DEFMSTR。2 位 DEFMSTR_TYPE 标志位选择缺省主控类型（无缺省主控，最近访问主控，固定缺省主控）然而 3 位 FIXED_DEFMSTR 标志位选择一个固定缺省主控，假设 DEFMSTR_TYPE 被设置为一个固定缺省主控。见 133 页“总线矩阵用户接口”。

19.4 仲裁

总线矩阵提供仲裁功能，当冲突情况发生，此功能减小延迟时间，就是说，当两个或更多主控试图在同一时间访问相同的从控 。总线矩阵仲裁方法用轻微的改进了的循环算法，依赖于从控的从控配置寄存器中的参数，此算法对第一次访问特定主控授权总线。

有三种循环算法类型：

- 无缺省主控的循环算法
- 最近访问主控的循环算法
- 固定缺省主控的循环算法

www.mcuzone.com

19.4.1 无缺省主控的循环仲裁

这是被总线矩阵仲裁使用的主算法。在一个纯循环算法方式，它允许总线矩阵从不同的主控调度请求到相同的从控。在当前访问的结尾，如果无其他请求待定，从控从所有的主控断开。此配置对第一次访问产生一个延迟时间周期。无缺省主控的仲裁可以被用作执行突发操作较多的主控。

19.4.2 最近访问主控的循环仲裁

这是一个被总线矩阵仲裁使用的边缘循环算法。它允许总线矩阵对最近一次访问从控的主控缩短一个延迟时间周期。事实上，在当前传输的结尾，如果无其他主控请求待定，从控仍然连接至执行访问的最近一次主控。如果其他主控访问同样的从控，这些非特权主控仍然可获一个延迟时间周期。此技术可被用于主要执行单一访问的主控。

19.4.3 固定缺省主控的循环仲裁

这是一个边缘的循环算法。它允许总线矩阵仲裁对从控的固定主控缩短一个延迟时间周期。在当前访问的结尾，从控仍然连接它的固定缺省主控。通过此固定缺省主控尝试的任何请求不会导致任何延迟时间，然而其他非特权主控仍获得一个延迟周期。此技术可被用作主要执行单一访问的主控。

19.5 总线矩阵用户接口

表 19-1 寄存器映射

偏移量	寄存器	名称	访问	复位值
0x0000	主控配置寄存器	MATRIX_MCFG	只写	0x00000000
0x0004	从控配置寄存器 0	MATRIX_SCFG0	读/写	0x00000010
0x0008	从控配置寄存器 1	MATRIX_SCFG0	读/写	0x00000010
0x000C	从控配置寄存器 2	MATRIX_SCFG0	读/写	0x00000010
0x0010	从控配置寄存器 3	MATRIX_SCFG0	读/写	0x00000010
0x0014	从控配置寄存器 4	MATRIX_SCFG0	读/写	0x00000010
0x0018-0x0020	保留	-	-	-
0x0024	MATRIX TCM 配置寄存器	MATRIX_TCR	读/写	0x00000000
0x0028-0x002C	保留	-	-	-
0x0030	EBI 片选分配寄存器	EBI_CSA	读/写	0x00000000
0x0034	USBPAD 上拉控制寄存器	USB_PUCR	读/写	0x00000000

www.mcuzone.com

19.5.1 总线矩阵主控配置寄存器

寄存器名称：MATRIX_MCFG

访问类型：只写

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	–	–	RCB1	RCB0

- RCBx：AHB 主控 x 重映射命令位

0：无效

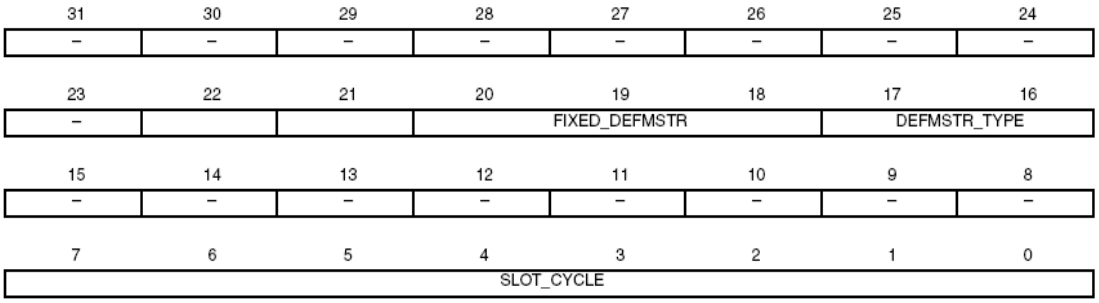
1：此命令位用于取反设置：写 1 交替的取消和使能主控 x 存储寻址从控的重映射。

www.mcuzone.com

19.5.2 总线矩阵从控配置寄存器

寄存器名称：MATRIX_SCFG0...MATRIX_SCFG4

访问类型：读/写



- SLOT_CYCLE：对一个突发访问允许的最大周期数

当一个突发访问到达 SLOT_CYCLE 界限，此访问可能被另外试图访问此从控的主控打断。

该限制被设置用于避免当突发周期很长时锁定速度非常慢的设备。

此界限不应该很小。一个不合理的小值会破坏每个突发访问并使得总线矩阵没有执行任何数据传送而花时间在仲裁。对 SLOT_CYCLE 而言 16 个周期是个合理的值。

- DEFMASTR_TYPE：缺省主控类型

0：无缺省主控

在当前从控访问结尾，如果无其他主控请求等待，从控切断所有主控连接。

这对一个脉冲串的第一次传输将导致一个周期延迟。

1：上一个缺省主控

在当前从控访问结尾，如果无其他主控请求等待，从控仍然连接访问过的上一个主控。

这导致当上一个主控试图再次访问从控时没有一个周期延迟。

2：固定缺省主控

在当前从控访问结尾，如果无其他主控请求待定，从控同在 FIXED_DEFMSTR 寄存器中有其索引的固定主控连接。

这导致当固定主控试图再次访问从控而没有一个周期延迟。

- FIXED_DEFMSTR:缺省主控的固定索引

这是对此值固定缺省主控的索引

www.mcuzone.com

19.5.3 总线矩阵 TCM 配置寄存器

寄存器名称：MATRIX_TCR

访问类型：读/写

复位值：0x0000_0000

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

- ITCM_SIZE:ITCM 使能存储器块容量
  0000 0KB(无 ITCM 存储器)
  0101：16KB
  0110：32KB
  0111：64KB
  其他：保留
- DTCM_SIZE：DTCM 使能存储器块容量

0000 0 KB（无 DTCM 存储器）

0101: 16 KB

0110: 32 KB

0111: 64 KB

其他：保留

www.mcuzone.com

19.5.4 EBI 片选分配寄存器

寄存器名称：EBI_CSA

访问类型：读/写

复位值：0x0000_0000

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	EBI_DBPUC
7	6	5	4	3	2	1	0
–	–	EBI_CS5A	EBI_CS4A	EBI_CS3A	–	EBI_CS1A	–

- EBI_CS1A：EBI 片选 1 指定

0=EBI 片选 1 被分配到静态存储器控制器

1=EBI 片选 1 被分配到 SDRAM 控制器

- EBI_CS3A：EBI 片选 3 指定

0=EBI 片选 3 仅分配到静态存储器控制器并且 EBI_NCS3 按 SMC 定义运转。

1=EBI 片选 3 被分配到静态存储器控制器并且 SmartMedia 逻辑被激活 。

- EBI_CS4A：EBI 片选 4 指定

0=EBI 片选 4 仅分配到静态存储器控制器并且 EBI_NCS4 按 SMC 定义运转。

1=EBI 片选 4 被分配到静态存储器控制器并且 CompactFlash（第一个槽）被激活。

- EBI_CS5A：EBI 片选 5 指定

0=EBI 片选 5 仅分配到静态存储器控制器并且 EBI_NCS5 按 SMC 定义运转。

1=EBI 片选 5 被分配到静态存储器控制器并且 CompactFlash（第二个槽）被激活。

- EBI_DBPUC：EBI 数据总线上拉配置

0=EBI D0-D15 数据总线位被内部上拉到 VDDIOM0 电源。

1=EBI D0-D15 数据总线位不内部上拉。

www.mcuzone.com

19.5.5 USB PAD 上拉控制寄存器

寄存器名称：USB_PUCR

访问类型：读/写

复位值：0x0000_0000

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

0：PAD 上拉使无效

1：PAD 上拉使能

www.mcuzone.com

20 外部总线接口（EBI）

20.1 概论

外部总线接口（EBI）被设计来确保若干外部设备和基于 ARM 设备的嵌入式存储器控制器之间数据成功传输。静态存储器和 SDRAM 控制器都被 EBI 上的外部存储器控制器所支持。这些外部存储器控制器可以处理若干类型的外部存储器和外设，像 SRAM,PROM，EPROM,EEPROM，Flash,和 SDRAM。

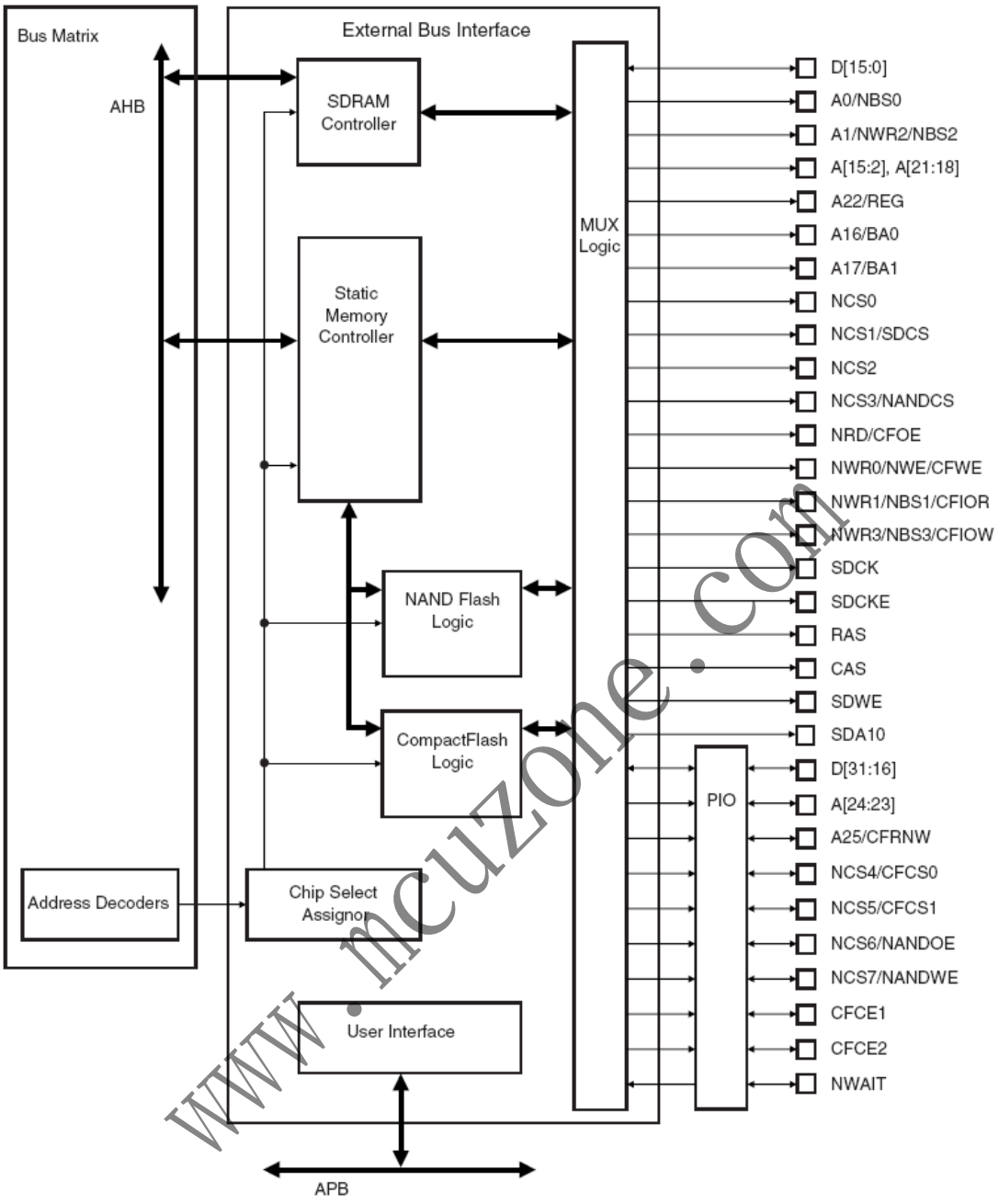
EBI 还通过集成的电路支持 CompactFlash 和 NAND Flash 协议，使得外部元件大量减少。此外，EBI 用多达八个外部设备处理数据传输，每个外部设备被指定到八个被嵌入式存储控制器定义的地址空间。数据传输通过一个 16 位或 32 位数据总线执行，地址总线多达 26 位，多达八个片选块（NCS[7:0]）和若干控制引脚,这些引脚通常在不同外部存储器控制器间多路复用。

20.2 方框图

图 20-1 展示了外部总线接口的组织。

www.mcuzone.com

图 20-1 展示了外部总线接口的组织



www.mcuzone.com

20.3 I/O 口线描述

20.4　　表 20-1　I/O 口线描述

名称	功能	类型	有效电平
EBI			
D0-D31	数据总线	I/O	
A0-A25	地址总线	输出	
NWAIT	外部等待信号	输入	低
SMC			
NCS0-NCS7	片选线	输出	低
NWR0-NWR3	写信号	输出	低
NRD	读信号	输出	低
MWE	写使能	输出	低
MBS0-NBS3	字节屏蔽信号	输出	低
EBI for CompactFlash Support			
CFCE1-CFCE2	CompactFlash 芯片使能	输出	低
CFOE	CompactFlash 输出使能	输出	低
CFWE	CompactFlash 写使能	输出	低
CFIOR	CompactFlashI/O 读使能	输出	低
CFIOW	CompactFlashI/O 写使能	输出	低
CFRNW	CompactFlash 读非写信号	输出	
CFCS0-CFCS1	CompactFlash 片选线	输出	低
EBI for NAND Flash Support			
NANDCS	NAND Flash 片选线	输出	低
NANDOE	NAND Flash 输出使能	输出	低
NANDWE	NAND Flash 写使能	输出	低
SDRAM 控制器			
SDCK	SDRAM 时钟	输出	
SDCKE	SDRAM 时钟使能	输出	高
SDCS	SDRAM 控制器片选线	输出	低
BA0-BA1	存储体选择	输出	
SDWE	SDRAM 写使能	输出	低
RAS-CAS	行和列信号	输出	低
NWR0-NWR3	写信号	输出	低
NBS0-NBS3	字节屏蔽信号	输出	低
SDA10	SDRAM A10 线	输出	

若干信号的连接不是直接通过 MUX 逻辑并依赖于此刻在使用的存储控制器。
142 页表 20-2 详述了两个存储控制器和 EBI 引脚的连接。

www.mcuzone.com

表 20-2　EBI 引脚和存储控制器 I/O 口线的连接

EBI 引脚	SDRAMC I/O 口线	SMC I/O 口线
NWR1/NBS1/CFIOR	NBS1	NWR1/NUB
A0/NBS0	不支持	SMC_A0/NLB
A1/NBS2/NWR2	不支持	SMC_A1
A[11:2]	SDRAMC_A[9:0]	SMC_A[11:2]
SDA10	SDRAMC_A10	不支持
A12	不支持	SMC_A12
A[14:13]	SDRAMC_A[12:11]	SMC_A[14:13]
A[25:15]	不支持	SMC_A[25:15]
D[31:16]	D[31:16]	D[31:16]
D[15:0]	D[15:0]	D[15:0]

20.4　应用举例

20.4.1　硬件接口

表 20-3 和表 20-4 详述了对每个存储控制器被用在 EBI 引脚和外部设备间的连接。

表 20-3　EBI 引脚和外部静态设备连接

引脚	接口设备的引脚					
	8 位静态设备	2×8 位静态设备	16 位位静态设备	4*8 位静态设备	2*16 位静态设备	32 位静态设备
控制器	**SMC**					
D0-D7						
D8-D15						
D16-D23	D0-D7	D0-D7	D0-D7	D0-D7	D0-D7	D0-D7
D24-D31	-	D8-D15	D8-D15	D8-D15	D8-D15	D8-D15
A0/NBS0	-	-	-	D16-D23	D16-D23	D16-D23
A1/NWR2/NBS2	A0	-	NLB	-	NLB	BE0
A2-A25	A1	A0	A0	WE	NBL	BE2
NCS0	A[2:25]	A[1:24]	A[1:24]	A[0:23]	A[0:23]	A[0:23]
NCS1/SDCS	CS	CS	CS	CS	CS	CS
NCS2	CS	CS	CS	CS	CS	CS
NCS3/NANDCS	CS	CS	CS	CS	CS	CS
NCS4/CFCS0	CS	CS	CS	CS	CS	CS
NCS5/CFCS1	CS	CS	CS	CS	CS	CS
NCS6/NAND0E	CS	CS	CS	CS	CS	CS
MCS7/NANDWE	CS	CS	CS	CS	CS	CS
NRD/CFOE	OE	OE	OE	OE	OE	OE
NWR0/NEW	WE	WE	WE	WE	WE	WE
NWR1/NBS1	-	WE	NUB	WE	NUB	BE1
NWR3/NBS3	-	-	-	WE	NUB	BE3

注意：　1．NWR1 使能高位字节写。NWR0 使能低位字节写。

2．NWRx 使能对应字节 x 写。（x=0,1,2 或 3）

3．NBS0 和 NBS1 分别使能低 16 位字的低位和高位字节

www.mcuzone.com

4．NBS2 和 NBS3 分别使能高 16 位字的低位和高位字节

5．BEx:字节 x 使能（x=0，1，2，或 3）

表 20-4EBI 引脚和外部设备连接

引脚	接口的设备引脚			
	SDRAM	CompactFlash	CompactFlash True IDE Mode	NAND Flash
控制器	**SDRAMC**	**SMC**		
D0 - D7	D0 - D7	D0 - D7	D0 - D7	I/O0-I/O7
D8 - D15	D8 - D15	D8 - D15	D8 - D15	I/O8-I/O15
D16 - D31	D16 - D31	A0	A0	
A0/NBS0	DQM0	A1	A1	
A2 - A10	A[0:8]	A[2:10]	A[2:10]	
A11	A9			
SDA10	A10			
A12				
A13 - A14	A[11:12]			
A15				
A16/BA0	BA0			
A17/BA1	BA1			
A18 - A20				
A21				CLE
A22		REG	REG	ALE
A23 - A24				
A25		CFRNW	CFRNW	
NCS0				
NCS1/SDCS	CS			
NCS2				CE
NCS3/NANDCS		CFCS0	CFCS0	
NCS4/CFCS0		CFCS1	CFCS1	
NCS5/CFCS1				RE
NCS6/NANDOE				WE
NCS7/NANDWE		-		
NRD/CFOE		OE	-	
NWR0/NWE/CFWE		WE	WE	
NWR1/NBS1/CFIOR	DQM1	IOR	IOR	
NWR3/NBS3/CFIOW	DQM3	IOW	IOW	
CFCE1		CE1	CS0	
CFCE2		CE2	CS1	
SDCK	CLK			
SDCKE	CKE			
RAS	RAS			
CAS	CAS			

www.mcuzone.com

SDWE	WE			
NWAIT		WAIT	WAIT	
Pxx		CD1 or CD2	CD1 or CD2	
Pxx				CE
Pxx				RDY

注意： 1．不是直接连接到 CompactFlash 槽。使能控制 EBI 数据总线和 CompactFlash 槽之间的双向缓冲。

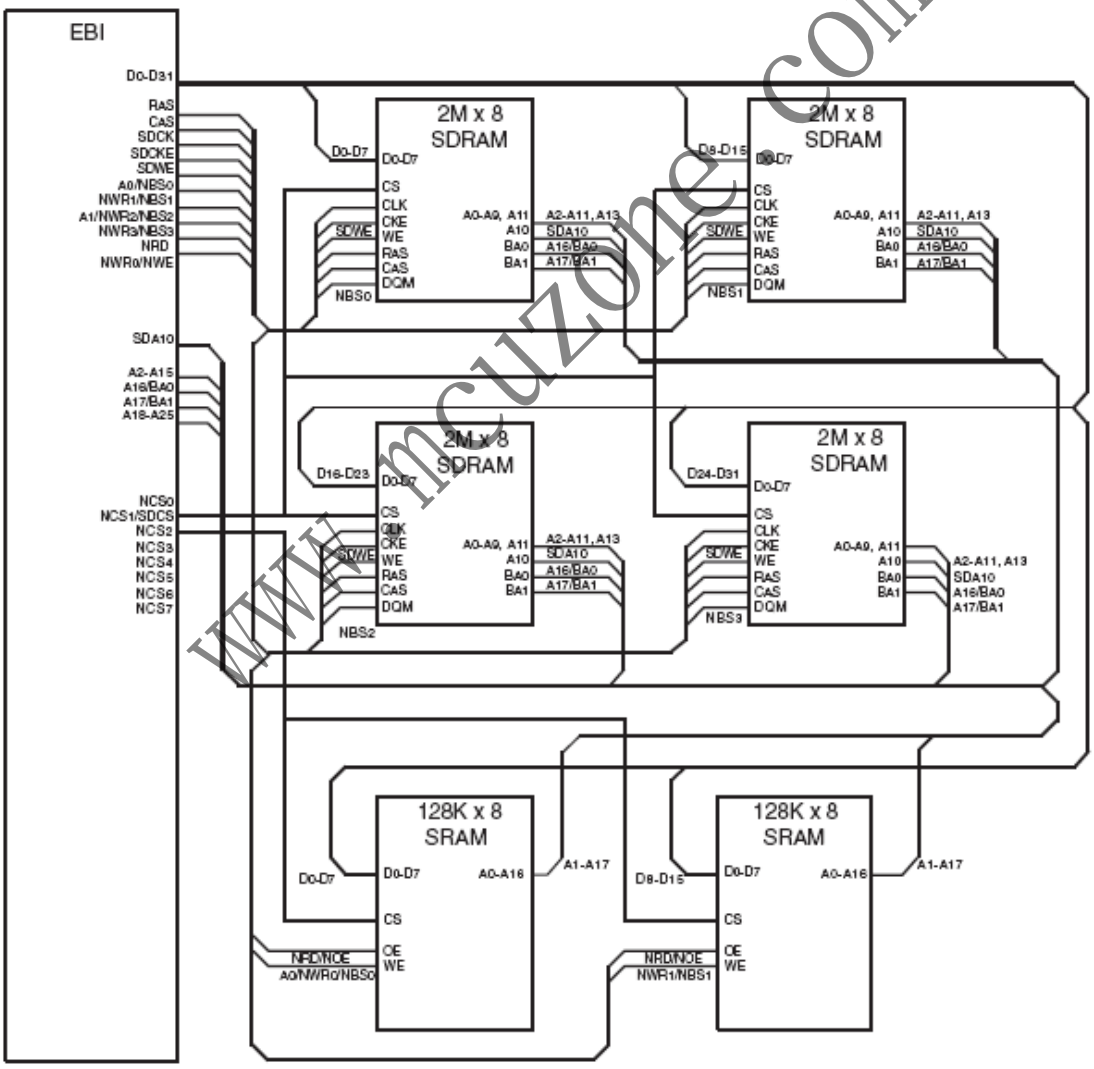
2．任何 PIO 线

3．CE 连接依赖于 NAND Flash。对标准的 NAND Flash 设备，必须被连接到任意空闲 PIO 线。对“CE don’t care” NAND Flash 设备，可以连接到 NCS3/NANDCS 或任何空闲 PIO 线。

20.4.2 连接举例

图 20-2 展示了 EBI 和外部设备连接的例子

图 20-2 EBI 连接到存储设备



20.5 产品相关性

20.5.1 I/O 口线

引脚用于接外部总线接口可以和 PIO 块多路复用。编程者必须首先编程 PIO 控制器来指定外部总线接口引脚到他们的外设功能。如果外部总线接口的 I/O 口线未被应用程序使用，则 I/O 口线可以被 PIO 控制器用作其他目的。

20.6 功能描述

EBI 在内部 AHB 总线（由总线矩阵驱动）和外部存储或外设间传输数据 。控制波形和外部地址的参数，数据和控制总线。它由以下元素组成：

- 静态存储控制器 （SMC）
- SRAM 控制器(SDRAMC)
- 片选分配部件 分配一个 AHB 地址空间到外部设备。
- 一个多路复用控制器电路共享不同控制器间的引脚。
- 可编程的 CompactFlash 支持逻辑
- 可编程的 NAND Flash 支持逻辑

20.6.1 总线多路复用

EBI 提供一个完整的控制信号集，此信号集通过存储器区请求功能的一个多路复用逻辑操作共享 32 位数据块，多达 26 位的地址块和控制信号。

当无外部访问正在被执行，多路复用被特别组织来在稳定状态保证地址和输出块的维护。多路复用还被指定来维护在存储控制器中定义的数据浮动时间。此外，SDRAM 的刷新周期在没有延迟其他外部存储控制器访问的情况下被 SDRAM 控制器独立的刷新。

20.6.2 上拉电阻控制

总线矩阵用户接口中的 EBI_CSA 寄存器可以使能不和 PIO 控制器块多路复用的数据总线块上的片上上拉电阻。上拉电阻在复位后被使能。置位 DBPUC 位禁用 D0-D15 块上的上拉电阻。使能 D16-D31 上的上拉电阻可通过编程适当的 PIO 控制器执行。

20.6.3 静态存储控制器

对静态存储控制器上的信息，参考静态存储控制器章节

20.6.4 SDRAM 控制器

对 SDRAM 控制器上的信息，参考 SDRAM 章节

20.6.5 CompactFlash 支持

外部总线接口集成了连接 CompactFlash 设备的电路。

CompactFlash 逻辑被 NCS4 和/或 NCS5 地址空间上的静态存储控制器（SMC）驱动。编程 EBI_CSA 寄存器上的 CS4A 和/或 CS5A 位到适当的值使能此逻辑。此寄存器的细节，请参考总线矩阵用户接口章节。访问一个外部 CompactFlash 设备通过访问保留到 NCS4 和/或 NCS5 的地址空间（就是说，对 NCS4 在 0x5000 0000 和 0x5FFF FFFF 之间，对 NCS5 在 0x6000 0000 和 0x6FFF FFFF 之间）完成。

所有的 CompactFlash 模式(Attribute Memory, Common Memory, I/O and True IDE)是被支持的，但是信号_IOIS16 (I/O 和 True IDE 模式) 和 _ATA SEL (True IDE 模式)不被处理。

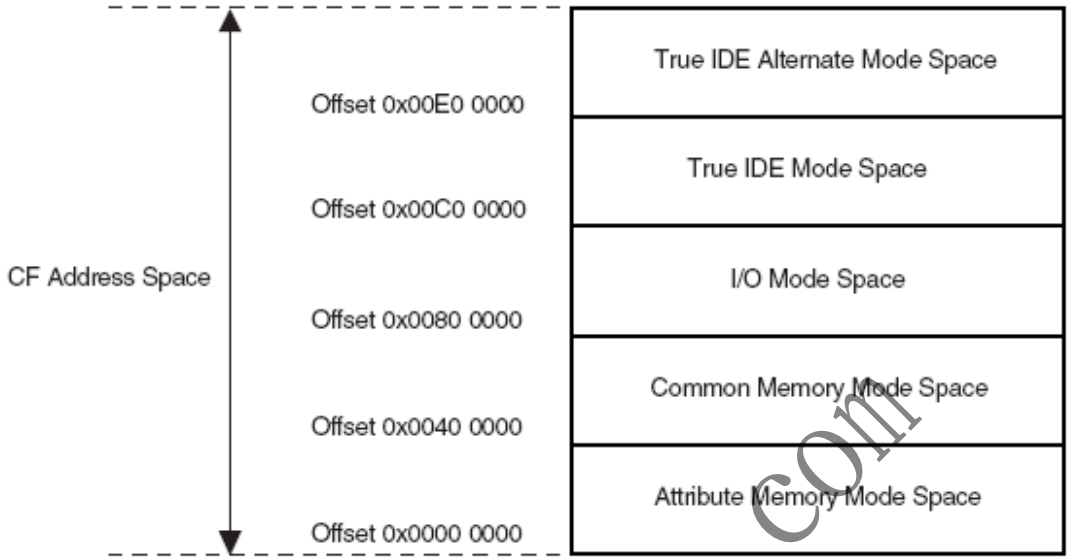
20.6.5.1 I/O 模式，Common Memory 模式，Attribute Memory 模式和 True IDE 模式

在 NCS4 和/或 NCS5 地址空间，当前传输地址区别 I/O 模式，Common Memory 模式，Attribute Memory 模式和 True IDE 模式。

www.mcuzone.com

不同模式通过一个如图 20-3 所说明的特殊存储映射被访问。传输地址的 A[23:21]位被用于选择需要的模式，如表 20-5 描述。

图 20-3 CompactFlash 存储映射



注意：EBI 的 A22 引脚被用于驱动 CompactFlash 设备（除了在 True IDE 模式）的 REG 信号。

表 20-5 CompactFlash 模式选项

A[23:21]	模式基本地址
000	Attribute Memory
010	Common Memory
100	I/O 模式
110	True IDE 模式
111	交替 True IDE 模式

20.6.5.2 CFCE1 和 CFCE2 信号

为了覆盖所有访问类型，SMC 必须交替的置位去驱动 8 位数据总线或 16 位数据总线。D[7:0]总线上的奇字节访问仅当 SMC 被配置用来驱动对应的 NCS 引脚（NCS4 和 或 NCS5）上的 8 位存储器。NCS4 和/或 NCS5 地址空间的片选寄存器(对应片选模式中的的 DBW 域)必须置位，如表 20-6 所示，去使能需要的访问类型。

NBS1 和 NBS0 是 SMC 的字节选择信号并当 SMC 在对应片选设置为字节选择模式时可用。

CFCE1 和 CFCE2 波形与相应的 NCSx 波形相同。关于这些波形和定时的细节，参考静态存储控制器选项。

表 20-6 CFCE1 和 CFCE2 真值表

模式	CFCE2	CFCE1	DBW	备注	SMC 访问模式
Attribute Memory	NBS1	NBS0	16 位	访问 D[7:0]偶字节	字节选择
Common Memory	NBS1	NBS0	16 位	访问 D[7:0]偶字节	字节选择

www.mcuzone.com

				访问 D[15:8]奇字节	
	1	0	8 位	访问 D[7:0]奇字节	无关紧要
I/O 模式	NBS1	NBS0	16 位	访问 D[7:0]偶字节 访问 D[15:8]奇字节	字节选择
	1	0	8 位	访问 D[7:0]奇字节	无关紧要
True IDE 模式					
Task file	1	0	8 位	访问 D[7:0]偶字节 访问 D[7:0]奇字节	无关紧要
数据寄存器	1	0	16 位	访问 D[7:0]偶字节 访问 D[15:8]奇字节	字节选择
交替 True IDE 模式					
控制寄存器 交替状态读	0	1	无 关紧要	访问 D[7:0]偶字节 访问 D[7:0]奇字节	无关紧要
驱动地址	0	1	8 位		无关紧要
True IDE Standby 模式或地址空间未指定到 CF	1	1	无 关紧要	无关紧要	无关紧要

20.6.5.3 读/写信号

在 I/O 模式和 True IDE 模式，CompactFlash 逻辑驱动 CFIOR 和 CFIOW 上的 SMC 读和写命令信号，当 CFOE 和 CFWE 信号无效。同样的，在 Common Memory 模式和 Attribute Memory 模式，SMC 信号在 CFOE 和 CFWE 信号上被驱动，当 CFIOR 和 CFIOW 无效。149 页图 20-4 展示了此逻辑的含义。通过置位地址设置和保持 NCS4(和/或 NCS5)片选上的时间到适当值支持属性存储模式，公用存储模式和 I / O 模式。

图 20-4 CompactFlash 读/写控制信号

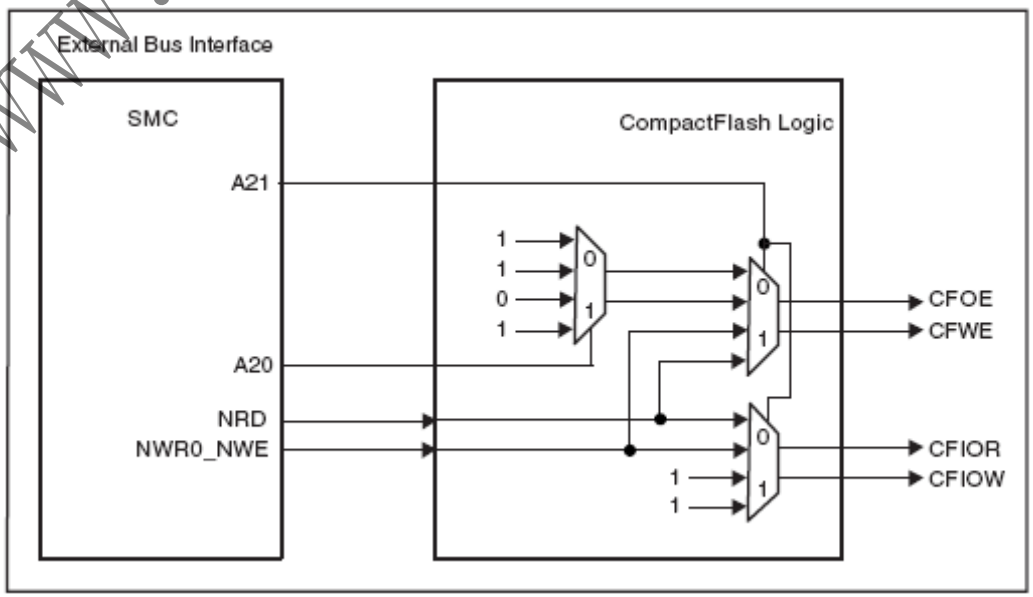


表 20-7 CompactFlash 模式选项

模式基地址	CFOE	CFWE	CFIOR	CFIOW

www.mcuzone.com

Attribute Memory Common Memory	NRD	NWR0_NWE	1	1
I/O 模式	1	1	NRD	NWR0_NWE
True IDE 模式	0	1	NRD	NWR0_NWE

20.6.5.4　　EBI 引脚上 CompactFlash 信号的多路复用

149 页表 20-8 和 150 页表 20-9 揭示了 CompactFlash 逻辑信号和 EBI 引脚上其他 EBI 信号的多路复用。只要 EBI_CSA 寄存器的 CS4A 和/或 CS5A 域置位，表 20-8 中 EBI 引脚严格的专用于 CompactFlash 接口。

当对应的 CompactFlash 接口使能（CS4A = 1 和/或 CS5A = 1），150 页表 20-9 中 EBI 引脚在所有存储区仍然可共享。

表 20-8　专用的 CompactFlash 接口多路复用表

引脚	**CompactFlash 信号**		**EBI 信号**	
	CS4A = 1	CS5A = 1	CS4A = 0	CS5A = 0
NCS4/CFCS0	CFCS0		NCS4	
NCS5/CFCS1		CFCS1		NCS5

表 20-9　共享的 CompactFlash 接口多路复用表

引脚	**访问 CompactFlash 设备**	**访问其他 EBI 设备**
	CompactFlash 信号	**EBI 信号**
NRD/CFOE	CFOE	NRD
NWR0/NWE/CFWE	CFWE	NWR0/NWE
NWR1/NBS1/CFIOR	CFIOR	NWR1/NBS1
NWR3/NBS3/CFIOW	CFIOW	NWR3/NBS3
A25/CFRNW	CFRNW	A25

20.6.5.5　　应用举例

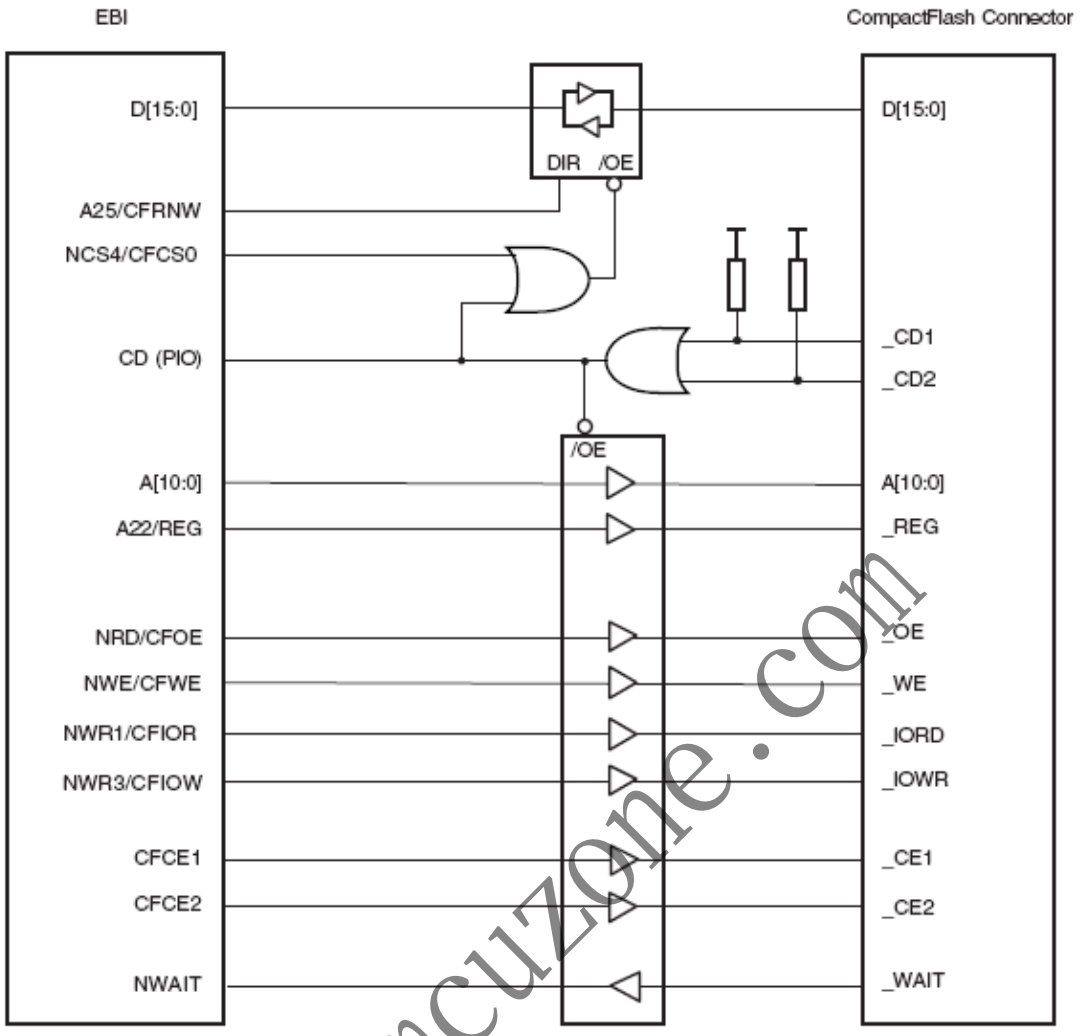
150页图20-5说明了CompactFlash应用。CFCS0 和CFRNW信号未直接连接到CompactFlash槽0，但是控制在EBI和CompactFlash设备间的缓冲的方向与输出使能。CFCS0信号的时序与NCS4信号相同。而且，CFRNW信号和地址总线一样，在整个传输中始终有效，。

CompactFlash _WAIT信号连接于静态存储控制器的NWAIT输入。

关于这些波形和时序的细节，参考静态存储控制器章节。

图20-5 CompactFlash应用程序举例

www.mcuzone.com



20.6.6 NAND Flash支持

EBI集成了连接NAND Flash设备的电路。

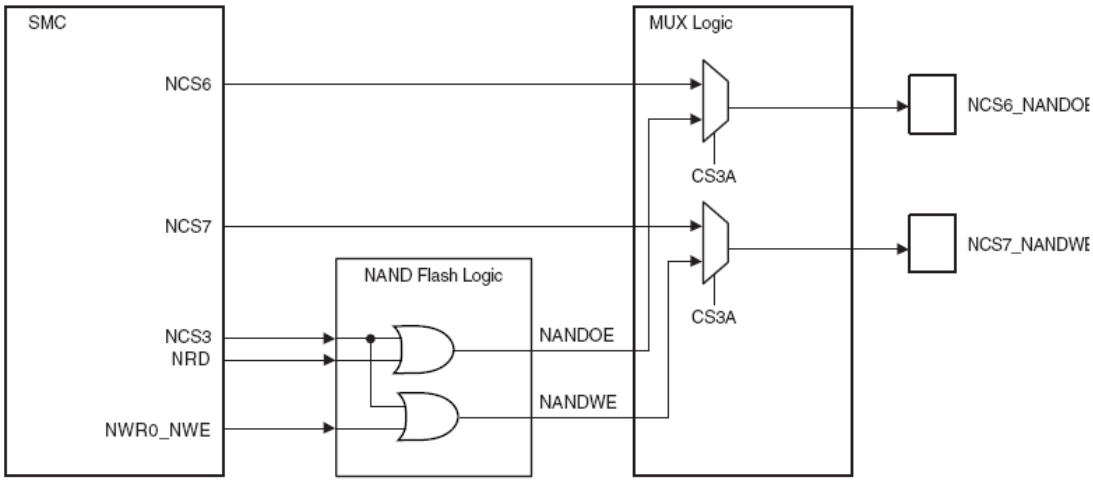
NAND Flash逻辑被NCS3地址空间上的静态存储控制器驱动。编程总线矩阵用户接口中EBI_CSA寄存器中的CS3A到适当的值使能NAND Flash逻辑。关于此寄存器的细节，参考总线矩阵用户接口段。通过访问NCS3（就是，在0x40000000和0x4FFFFFFF之间）的地址空间来访问外部NAND Flash设备。

当NCS3信号有效，NAND Flash逻辑驱动NANDOE和 NANDWE信号上SMC的读和写命令。只要访问NCS3地址空间失败，NANDOE 和 NANDWE就被使无效。关于这些波形的更多细节，参考静态存储控制器章节。

NANDOE 和 NANDWE信号同静态存储控制器的NCS6和NCS7多路复用。此多路复用在EBI的MUX逻辑被EBI_CSA寄存器中的CS3A位 控制。关于此寄存器的细节，参考总线矩阵用户接口章节。此时NCS6 和 NCS7变得不可用。执行在保留到NCS6 和 NCS7（就是说，在0x70000000和0x8FFFFFFF之间）的地址空间的一个访问可能导致一个不可预期的结果。

www.mcuzone.com

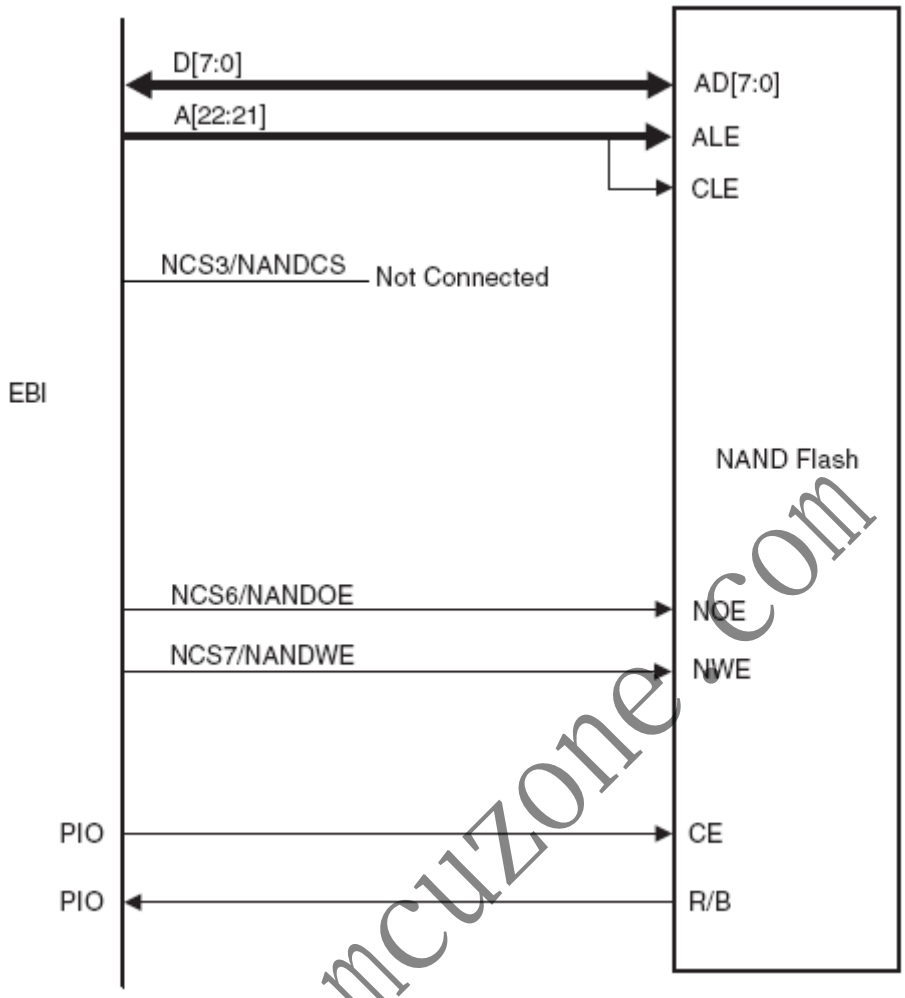
图20-6　EBI引脚上NAND Flash信号多路复用



NAND Flash设备上的地址锁存使能和命令锁存使能信号通过EBI地址总线的地址位A22和A21驱动。用户应该注意EBI地址总线上的任何位也可能被用作此目的。NAND Flash设备的数据总线上的命令，地址或数据字通过用他们在NCS3地址空间的地址来区别。设备的芯片使能（CE）信号和准备/忙（R/B）信号被连接到PIO口线。就算NCS3未被选择，CE信号仍然有效，防止此设备进入待机状态。

www.mcuzone.com

图20-7 NAND Flash应用程序举例



注意：外部总线接口还可支持16位设备。

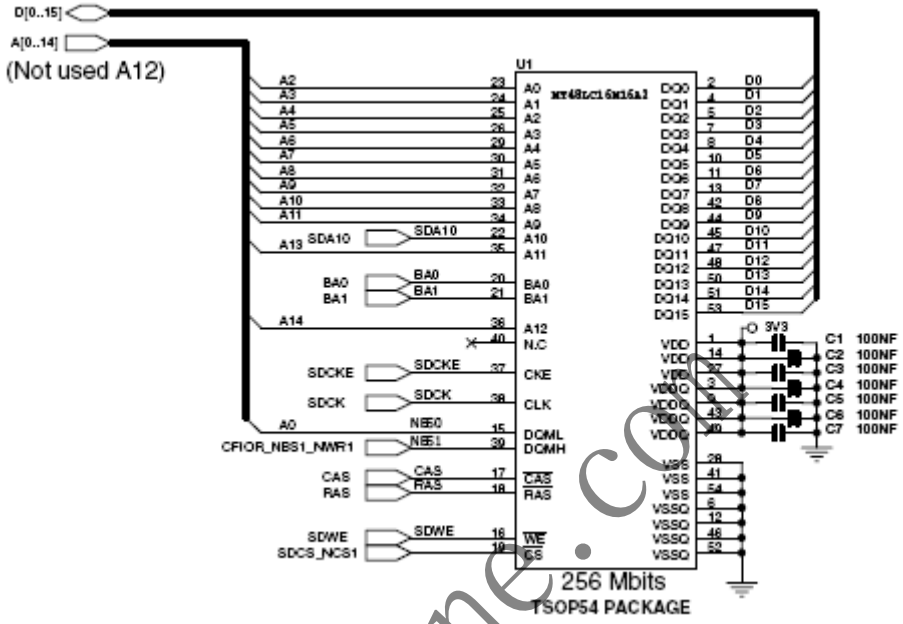
www.mcuzone.com

20.7 实现举例

所有硬件配置仅给出说明。用户应该参考存储器制造商网站去确认设备可用性。

20.7.1 16位SDRAM

20.7.1.1 硬件配置



20.7.1.2 软件配置

以下配置必须被执行：

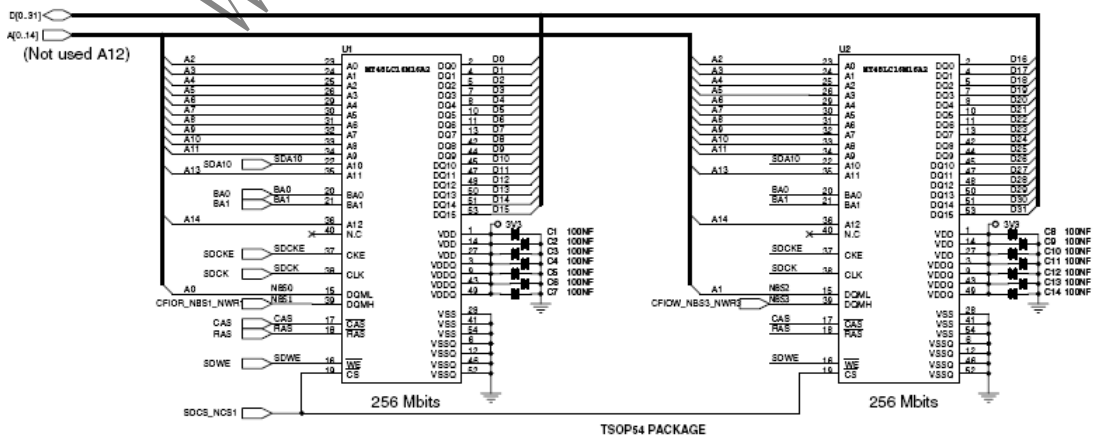
- 通过置位放置在总线矩阵存储器空间EBI片选制定寄存器中的EBI_CS1A位指定EBI CS1到SDRAM控制器。
- 初始化SDRAM控制器，依赖于SDRAM设备和系统总线频率

数据总线宽度编程为16位宽。

数据总线初始化顺序在SDRAM控制器的“SDRAM设备初始化”章节有描述。

20.7.2 32位SDRAM

20.7.2.1 硬件配置



20.7.2.2 软件配置

以下配置必须被执行：

www.mcuzone.com

• 通过置位位于在总线矩阵存储器空间EBI片选制定寄存器中的EBI_CS1A位指定EBI CS1到SDRAM控制器。

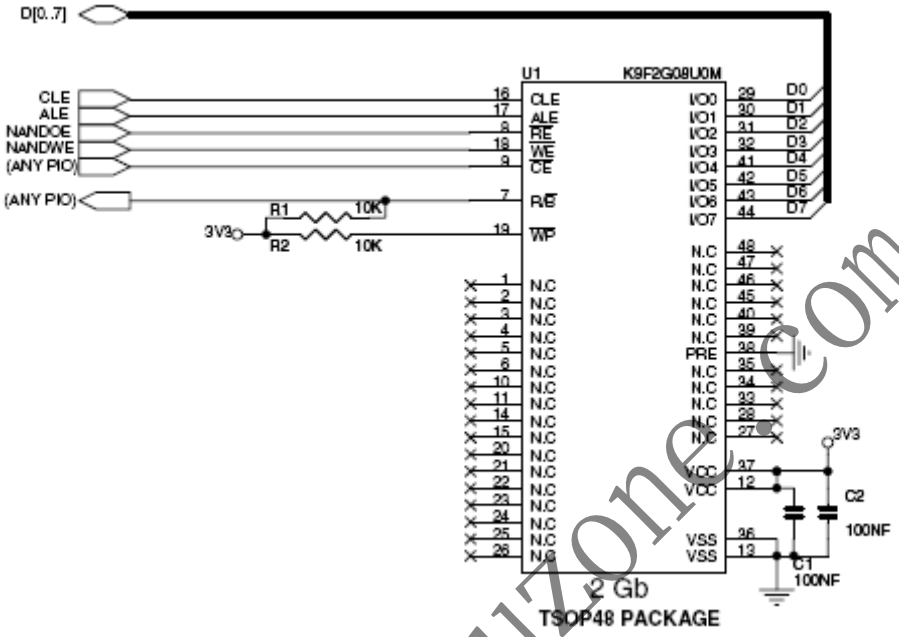
• 初始化SDRAM控制器，依赖于SDRAM设备和系统总线频率

数据总线宽度被编程为32位。数据口线D[16..31]和PIO块多路复用，因此专用的PIO必须在PIO控制器中被编程为相应的外设模式。

数据总线初始化顺序在SDRAM控制器的“SDRAM设备初始化”章节有描述。

20.7.3 8位NANDFlash

20.7.3.1 硬件配置



20.7.3.2 软件配置

以下配置必须被执行：

• 通过置位放置在总线矩阵存储器空间EBI片选制定寄存器中的EBI_CS3A位指定EBI CS3到SDRAM控制器。

• 给ALE / CLE功能保留A21 / A22。地址和命令锁存在访问期间分别通过置位地址位A21和A22来控制。

• NANDOE 和 NANDWE信号同PIO块多路复用，因此专用的PIO必须在PIO控制器中被编程为相应的外设模式。

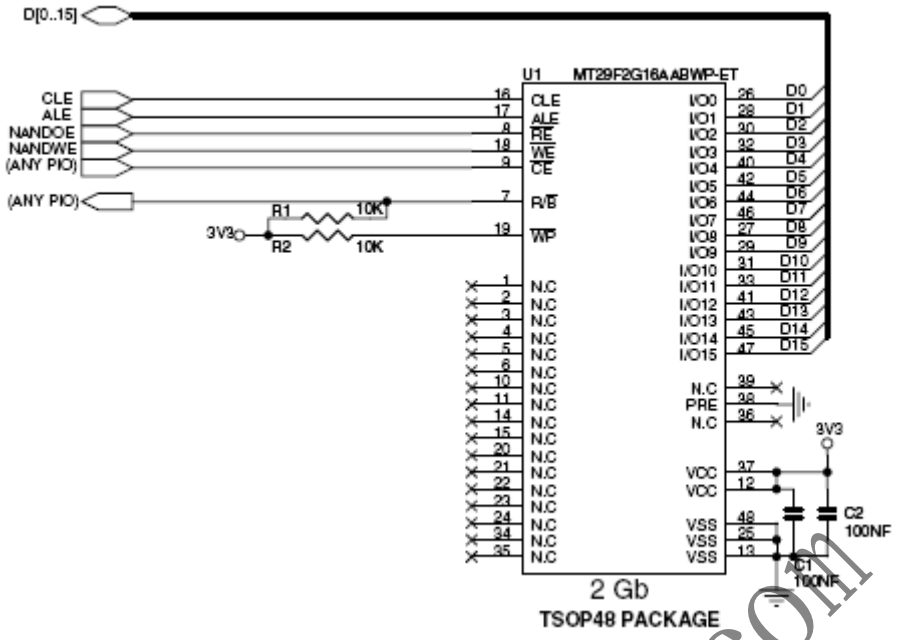
• 配置一个PIO口线作为输入来管理准备/忙信号

• 根据NAND Flash时序，数据总线宽度和系统总线频率配置静态存储控制器CS3的建立，脉冲，周期参数和模式。

20.7.4 16位NANDFlash

20.7.4.1 硬件配置

www.mcuzone.com

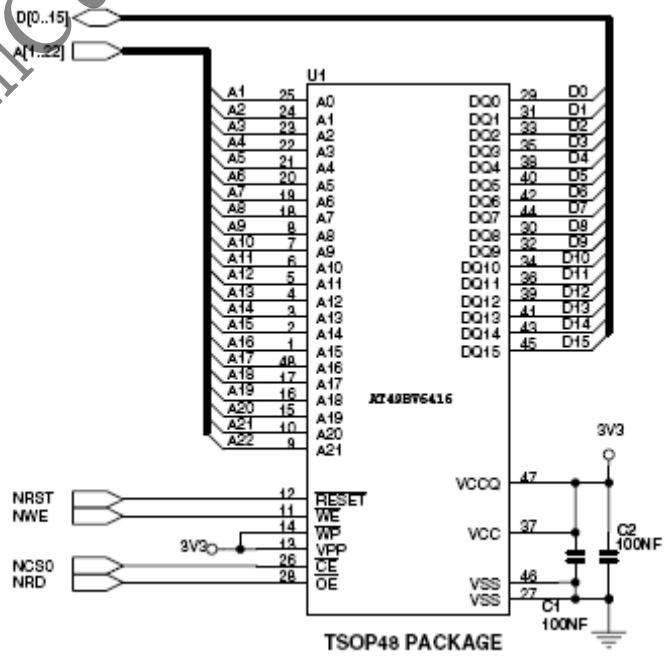


20.7.4.2 软件配置

除了在静态存储控制器的模式寄存器中被编程的数据总线宽度，软件配置和8位NANDFlash一样。

20.7.5 NOR Flash on NCS0

20.7.5.1 硬件配置



20.7.5.2 软件配置

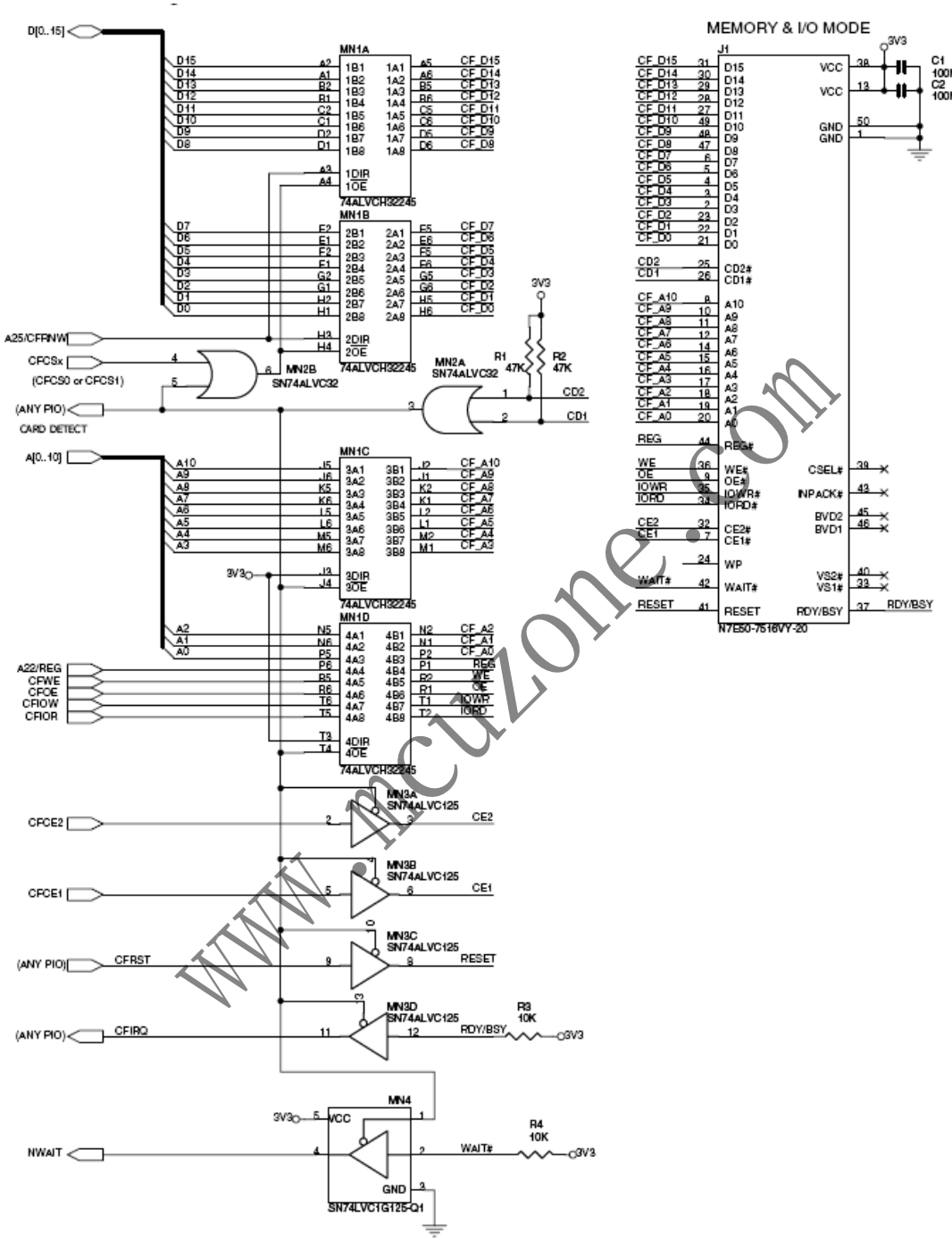
对静态存储控制器的缺省配置，字节选择模式，16位数据总线，读/写被片选控制，允许以慢时钟在16位非易失性存储器上启动。

www.mcuzone.com

对其他配置，根据Flash定时和系统总线频率配置静态存储控制器CS0建立，脉冲，周期参数和模式。

20.7.6 Compact Flash

20.7.6.1 硬件配置



20.7.6.2 软件配置

以下配置必须被执行：

- 通过置位放置在总线矩阵存储空间的EBI片选指定寄存器中的EBI_CS4A或/和 EBI_CS5A位，指定EBI CS4 和/或 EBI_CS5到CompactFlash 槽 0 或/和 槽 1。
- 地址线A23选择I/O（A23=1）或存储模式(A23=0)，A22用于REG功能。

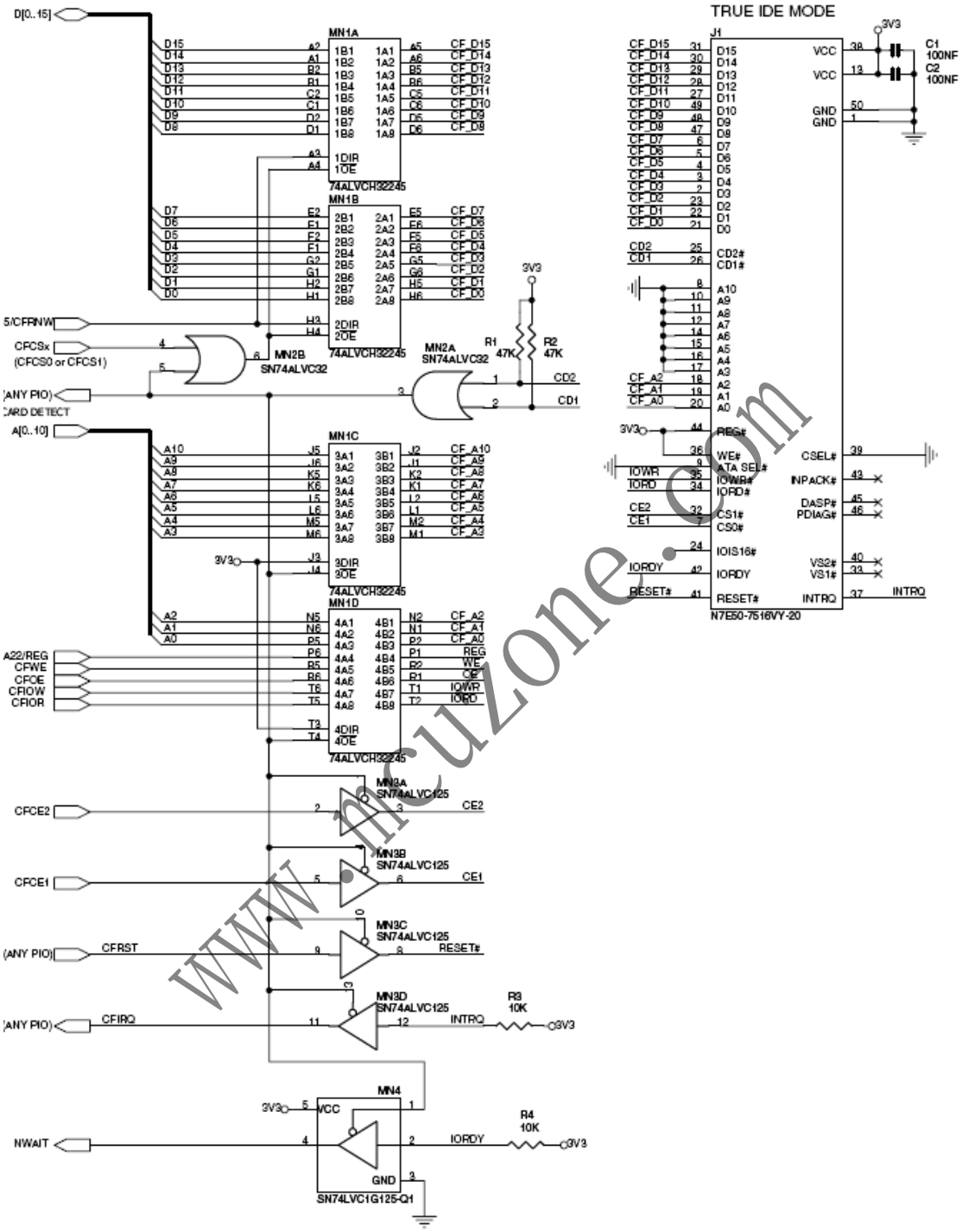
www.mcuzone.com

- A23, CFRNW, CFS0, CFCS1, CFCE1 和 CFCE2信号同PIO口线多路复用，因此专用的PIO必须在PIO控制器外设模式下被编程。
- 为CFRST配置一个PIO口线作为输入，分别为CFIRQ和CARD DETECT配置其他的两个PIO口线作为输入。
- 根据Compact Flash时序和系统总线频率，配置SMC CS4 和/或SMC_CS5 (给 slot 0 或 1)建立，脉冲，周期参数和模式。

www.mcuzone.com

20.7.7 Compact Flash True IDE

20.7.7.1 硬件配置



20.7.7.2 软件配置

以下配置必须被执行：

• 通过置位放置在总线矩阵存储空间的EBI片选指定寄存器中的EBI_CS4A或/和 EBI_CS5A位，指定EBI CS4 和/或 EBI_CS5到CompactFlash slot 0

www.mcuzone.com

或/和 slot 1。

- 地址线A21用于选择交替True IDE (A21=1) 或 True IDE (A21=0)模式。
- CFRNW, CFS0, CFCS1, CFCE1 和 CFCE2信号同PIO块多路复用，因此专用的PIO必须在PIO控制器外设模式下被编程。
- 为CFRST配置一个PIO口线作为输入，分别为CFIRQ和CARD DETECT配置其他的两个PIO口线作为输入。
- 根据Compact Flash时序和系统总线频率，配置SMC CS4 和/或SMC_CS5 (给 slot 0 或 1)的建立，脉冲，周期参数和模式。

www.mcuzone.com

21 静态存储控制器（SMC）

21.1 描述

静态存储控制器(SMC) 产生控制访问外部存储设备或外围设备的信号。它有8个片选和一个26 位的地址总线。32位数据总线可被配置到8-，16-，32位外部设备接口。分离的读和写控制信号允许直接存储器和外设接口。读和写信号波形是可全参数化的。

SMC可管理从外部设备到扩展当前访问德等待请求。提供一个自动慢时钟模式给SMC。在慢时钟模式，它会在读和写信号上从用户编程的波形到慢速率特定波形转换。SMC支持在页面模式异步脉冲串读访问多达32字节的页面大小。

21.2 I/O口线描述

表21-1　I/O口线描述

名称	描述	类型	有效电平
NCS[7:0]	静态存储控制器片选块	Output	低
NRD	读信号	Output	低
NWR0/NWE	写0/写使能信号	Output	低
A0/NBS0	地址位0/字节0选择信号	Output	低
NWR1/NBS1	写1 /字节1选择信号	Output	低
A1/NWR2/NBS2	地址位1/写2/字节2选择信号	Output	低
NWR3/NBS3	写3/字节3选择信号	Output	低
A[25:2]	地址总线	Output	
D[31:0]	数据总线	I/O	
NWAIT	外部等待信号	Input	低

21.3 多路复用信号

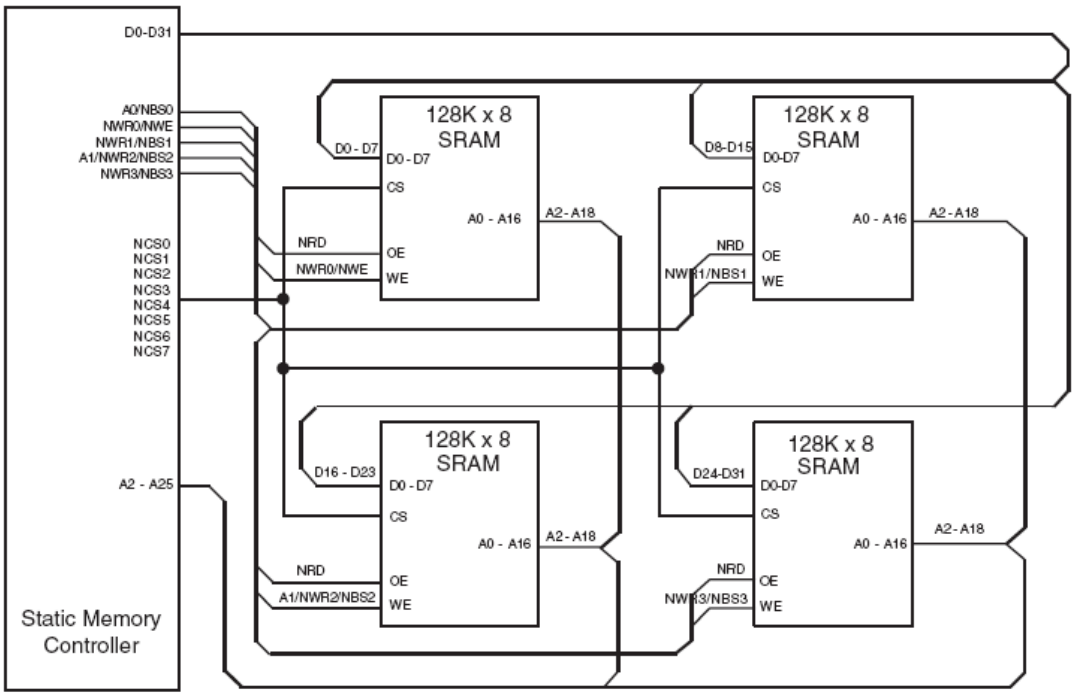
表21-2 静态存储控制器(SMC)多路复用信号

多路复用信号			相关功能
NWR0	NWE		字节写或字节选择访问，见165页“字节写或字节选择访问”
A0	NBS0		8位或16/32位数据总线，见165页“数据总线宽度”
NWR1	NBS1		字节写或字节选择访问，见165页“字节写或字节选择访问”
A1	NWR2	NBS2	8位或16/32位数据总线，见165页“数据总线宽度” 字节写或字节选择访问，见165页“字节写或字节选择访问”
NWR3	NBS3		字节写或字节选择访问，见165页“字节写或字节选择访问”

21.4 应用举例

21.4.1　硬件接口

图21-1 SMC连接到静态存储设备



21.5 产品相关性

21.5.1 I/O口线

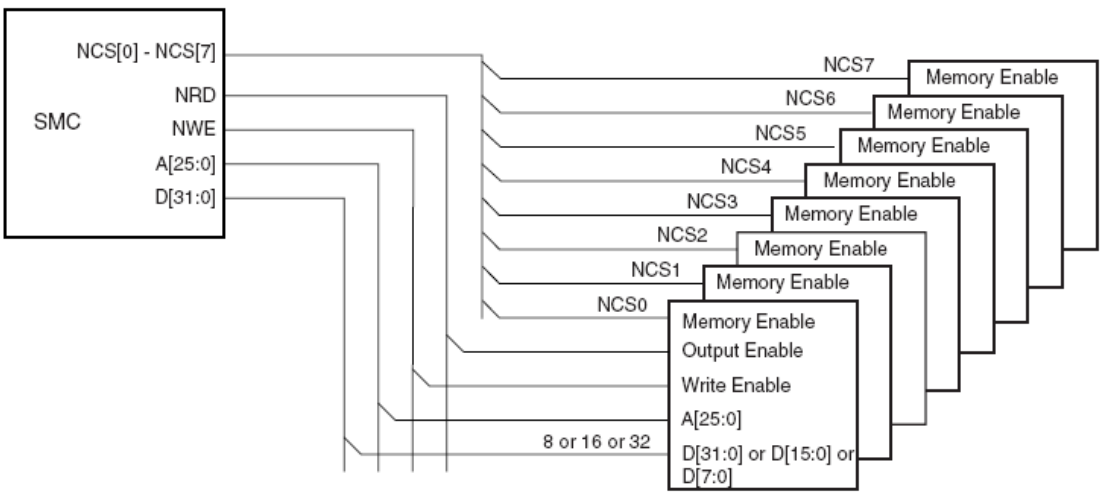
用来接静态存储控制器的引脚可以和PIO块多路复用。编程者必须首先编程PIO控制器来指定静态存储控制器引脚到他们的外设功能。如果SMC的I/O口线未被应用程序使用，则他们可被PIO控制器用于其它目的。

21.6 外部存储映射

SMC提供多达26位地址块，A[25:0] 。这就允许每个片选块到地址多达64M字节的存储量。

如果连接在一个片选上的物理存储设备比64M字节小，它将绕回并在此空间出现重复。SMC在页面（见图21-2）正确的处理任何有效的访问存储设备。A[25:0]仅对8位存储器有意义，A[25:1]用于16位存储，A[25:2]用于32位存储。

图21-2 八个外部设备的存储连接



21.7 连接到外部设备

21.7.1 数据总线宽度

8，16，或32位数据宽度可为每个片选选择。此选项为对应的片选被SMC_MODE（模式寄存器）中的DBW域控制。

图21-3展示了怎样在NCS2上连接一个512K x 8-位存储器。图21-4展示了怎样在NCS2上连接一个512K x 16-位存储器。图21-5展示了2个16位存储器连接为一个单32位存储器。

21.7.2 字节写或字节选择访问

带16位或32位数据总线的每个片选可一操作写访问两个不同类型中的一个：字节或字节选择访问。这为对应的片选被SMC_MODE寄存器的BAT域控制。

图21-3 8位数据总线的存储器连接

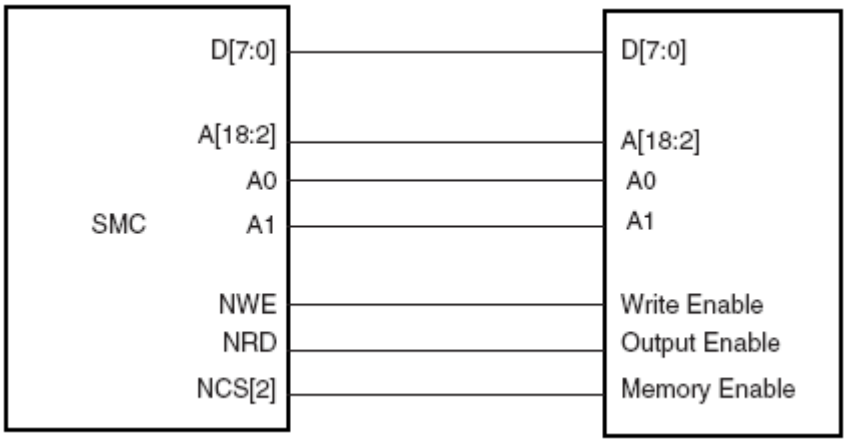


图21-4 16位数据总线的存储器连接

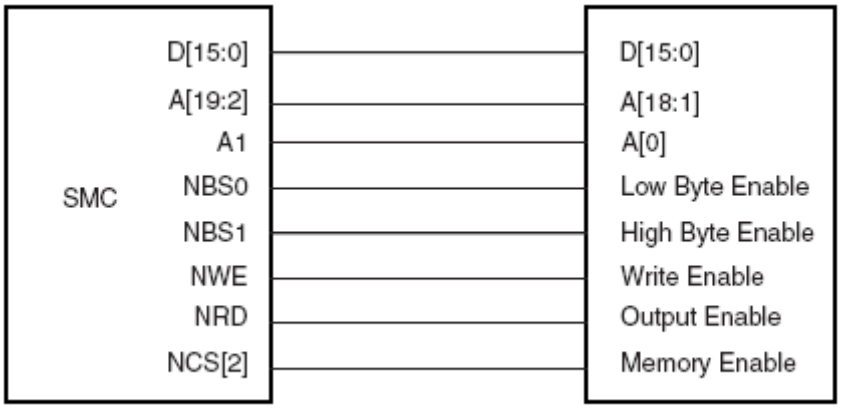
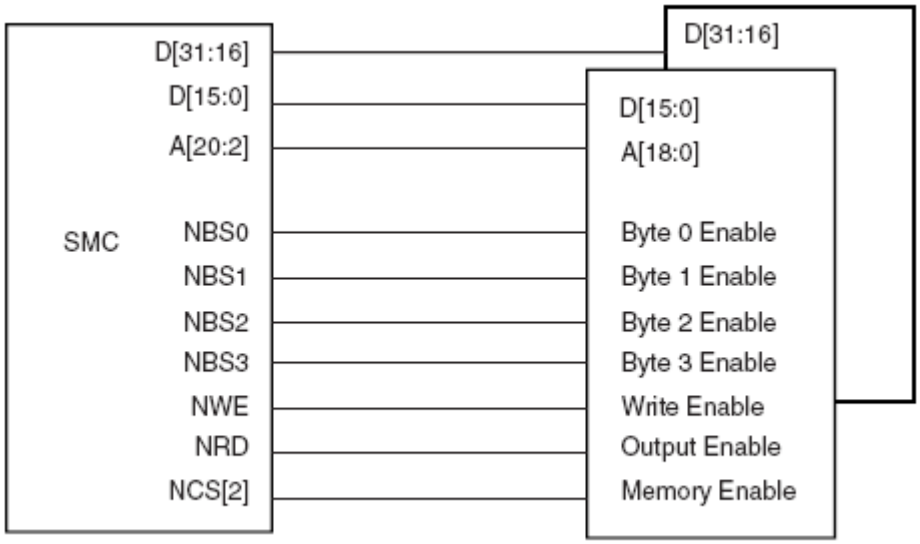


图21-5 32位数据总线的存储器连接



21.7.2.1 字节写访问

字节写访问支持数据总线每个字节一个字节写信号和一单个读信号。

注意SMC不允许在字节写访问模式启动。

1．对16位设备：SMC分别给16位总线的byte0（低位字节）和byte1（高位字节）提供NWR0 和 NWR1写信号。一单个读信号(NRD)被提供。

字节写访问被用于连接2*8位设备为一个16位存储器。

2．对32位设备：NWR0, NWR1, NWR2 和 NWR3，分别是byte0（低位字节），byte1，byte2，byte3（高位字节）的写信号。一单个读信号(NRD)被提供。

字节写访问被用于连接4*8位设备为一个32位存储器

字节写选项如图21-6 所示

21.7.2.2 字节选择访问

在此模式，读/写操作可以以一个字节电平使能/使无效。提供数据总线的每个字节一个字节选择块。一个NRD和一个NEW信号控制读和写。

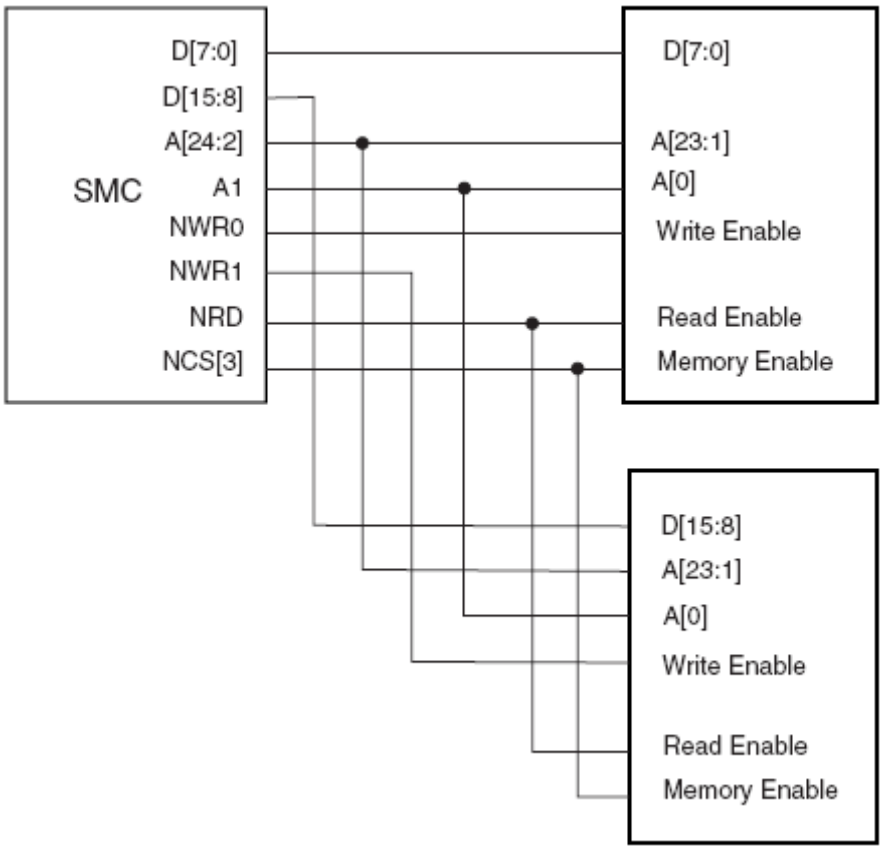
1．对16位设备：SMC分别给16位总线的byte0（低位字节）和byte1（高位字节）提供NBS0 和 NBS1选择信号。

字节选择访问用于连接一个16位设备

对32位设备：NBS0, NBS1, NBS2 和 NBS3，分别是byte0（低位字节），byte1，byte2，byte3（高位字节）的写信号.字节选择访问被用于连接2个16位设备。

图21-7 展示了在NCS3（BAT=字节选择访问）上在字节访问模式在一个32位数据总线上怎样连接2个16位设备。

图21-6　16位总线上2*8位设备的连接：字节写选项



21.7.2.3 信号多路复用

依赖于BAT，仅写信号或字节选择信号被使用。为在外部总线接口保存IO，SNC接口的控制信号被多路复用。表21-3 展示了依赖于数据总线宽度和字节访问类型的信号多路复用。

对32位设备，位A0和A1未被使用。对16 位设备，地址位A0未被使用。当字节选择选项被选择，则NWR1 到 NWR3未被使用。当字节写选项被选择，NBS0 到 NBS3未被使用。

图21-7 32位数据总线（字节选择选项）上2*16位数据总线的连接

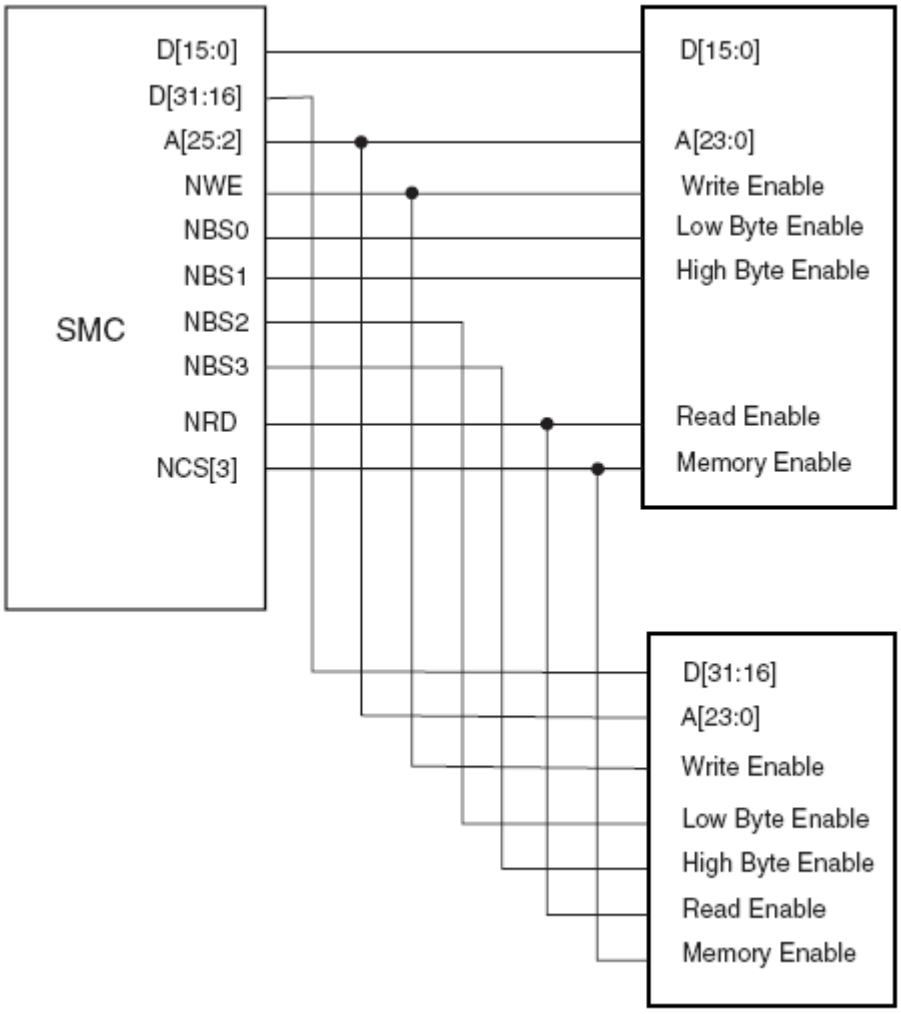


表21-3 SMC多路复用信号转换

信号名称	32位总线			16位总线		8位总线
设备类型	1*32位	2*16位	4*8位	1*16位	2*8为	1*8位
字节访问类型	字节选择	字节选择	字节写	字节选择	字节写	
NBS0_A0	NBS0	NBS0		NBS0		A0
NWE_NWR0	NWE	NWE	NWR0	NWE	NWR0	NWE
NBS1_NWR1	NBS1	NBS1	NWR1	NBS1	NWR1	
NBS2_NWR2_A1	NBS2	NBS2	NWR2	A1	A1	A1
NBS3_NWR3	NBS3	NBS3	NWR3			

21.8 标准读和写协议

在以下段落中，未考虑字节访问类型。字节选择块(NBS0 到NBS3)总有相同的定式 作为一个地址总线。NWE表示字节选择访问类型中的NEW信号或字节写访问类型中的字节写模块（NWR0 到 NWR3）之一。NWR0 到 NWR3和Nw E有相同的定时和协议。在相同的方式中，NCS表示NCS[0..7]片选块中的一个。

21.8.1 读波形

读周期如图21-8所示。

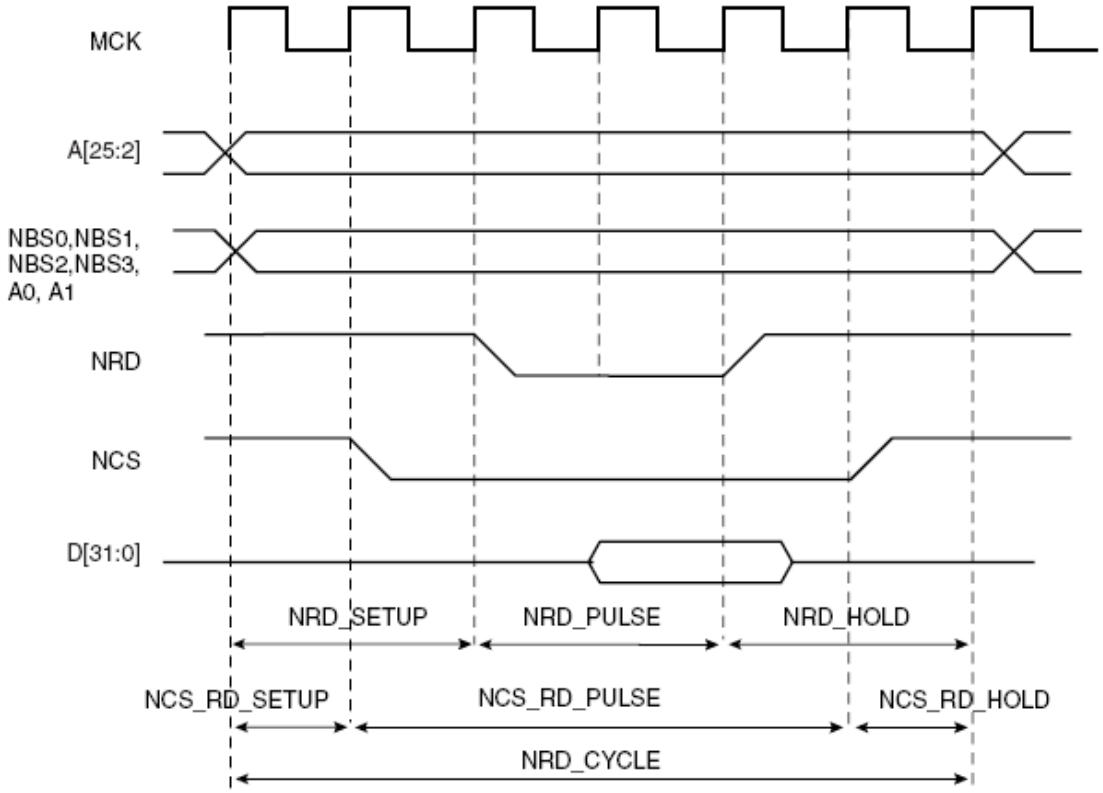
读周期和存储器地址总线上的地址置位一起开始，就是说：

对8位设备{A[25:2], A1, A0}

对16位设备{A[25:2], A1}

对32位设备A[25:2]

图21-8 标准读周期



21.8.1.1 NRD波形

NRD信号由一个设定定时，一个脉冲宽度和一个保持定时表述。

1． NRD_SETUP:NRD设定时间被定义为NRD下降沿前地址的设置。

2． NRD_PULSE:NRD脉冲长度是NRD下降沿和NRD上升沿间的时间。

3． NRD_HOLD:NRD保持时间被定义为NRD上升沿后地址的保持时间。

21.8.1.2 NCS波形

类似的，NCS信号可以被分成一个设定时间，脉冲长度和保持时间：

1．NCS_RD_SETUP:NCS设定时间被定义为NCS下降沿前地址的设定时间。

2．NCS_RD_PULSE:NCS脉冲长度是NCS下降沿河NCS上升沿间的实间。

3．NCS_RD_HOLD:NCS保持时间被定义为NCS上升沿后地址的保持时间。

21.8.1.3 读周期

NRD_CYCLE时间被定义为读周期总共的持续时间，就是说，从地址总线上地址置位到地址可能改变的点的时间。总的读周期时间等于：
NRD_CYCLE = NRD_SETUP + NRD_PULSE + NRD_HOLD

= NCS_RD_SETUP + NCS_RD_PULSE + NCS_RD_HOLD

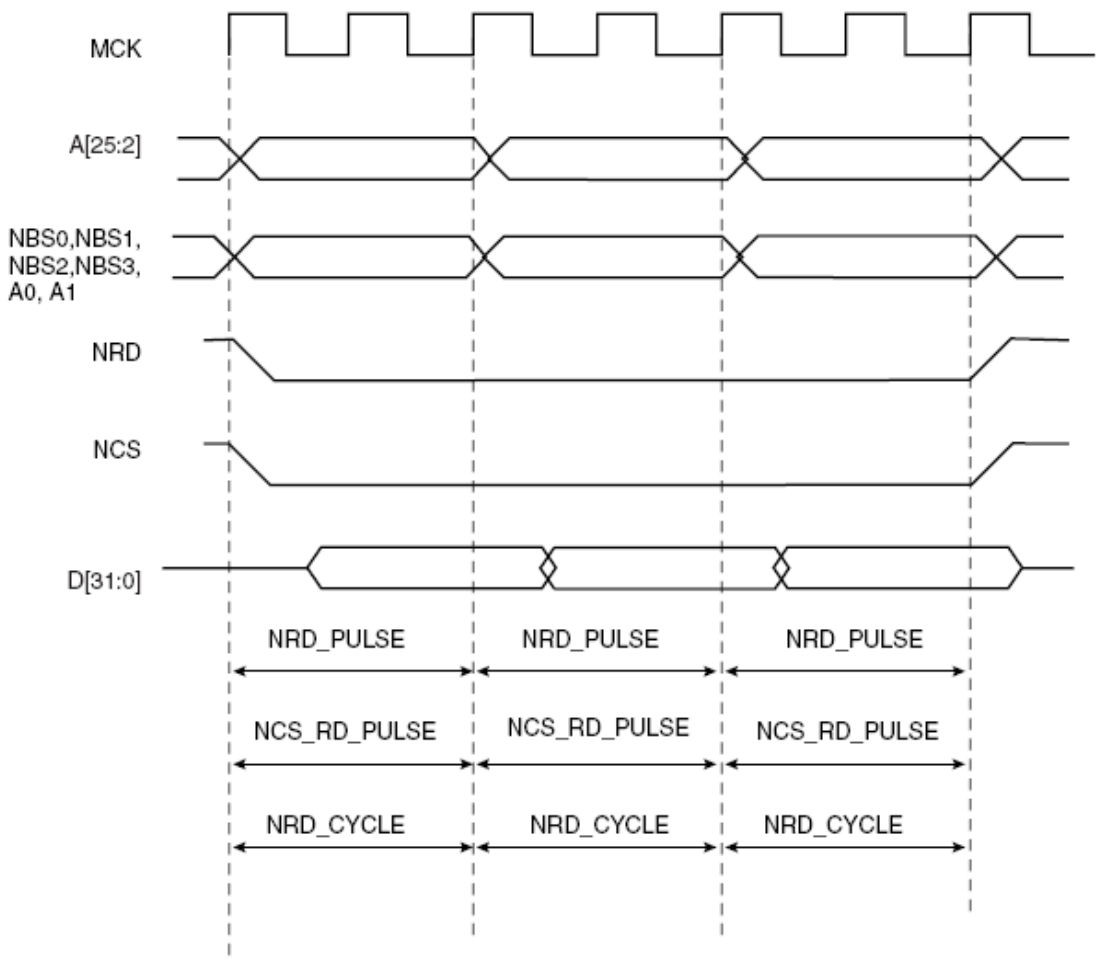
所有NRD和NCS定时对每个片选单独的定义为整的主控时钟周期数。为保证NRD和NCS定时是一致的，用户必须定义总的读周期代替保持定时。
NRD_CYCLE隐含的定义NRD保持时间和NCS保持时间为：
NRD_HOLD = NRD_CYCLE - NRD SETUP - NRD PULSE

NCS_RD_HOLD = NRD_CYCLE - NCS_RD_SETUP - NCS_RD_PULSE

21.8.1.4 空延迟设定和保持、

如果空设定和保持参数对NRD和/或NCS编程，NRD和NCS维持连续有效，假如在同样的存储器（见图21-9 ）中连续的读周期。

图21-9 NRD 和NCS上无Setup,五Hold读信号



21.8.1.5 空脉冲

编程空脉冲是不被允许的。脉冲必须至少设置到1。空值导致不可预期的操作。

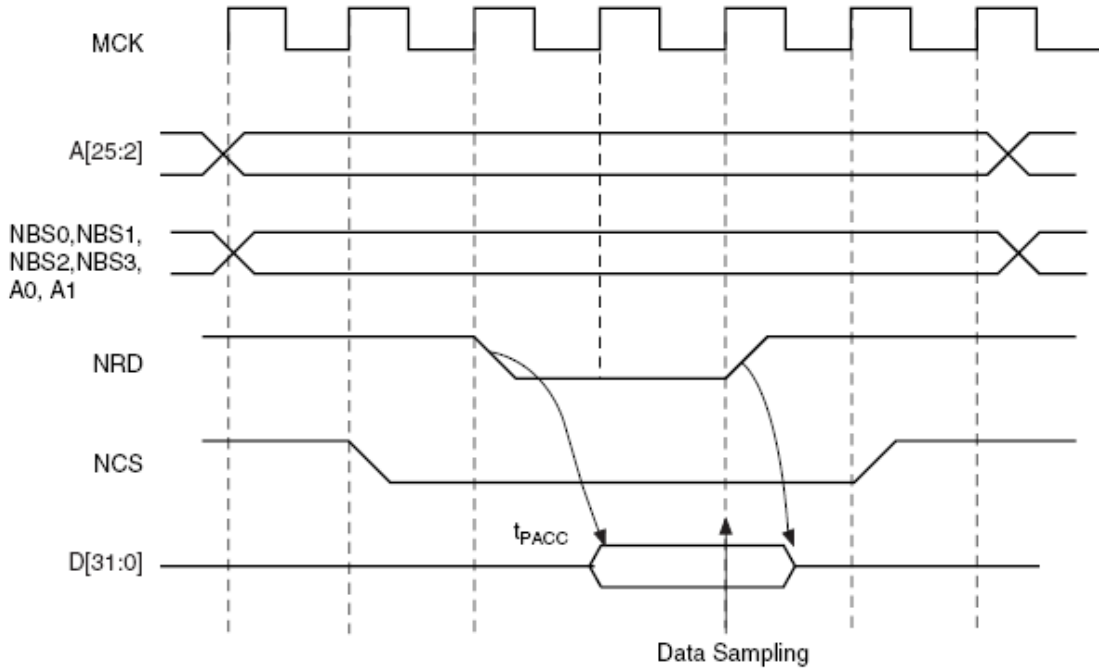
21.8.2 读模式

如NCS和NRD波形被独立的定义，SMC需要知道什么时候在数据总线上读数据是可行的。SMC不比较NCS和NRD定时来知道哪个信号更快。对应片选的SMC_MODE寄存器中的READ_MODE参数指示了NRD和NCS哪个信号控制读操作。

21.8.2.1 读操作由*NRD (READ_MODE = 1)* 控制

图21-10展示了一个典型异步RAM的读操作波形。NRD的下降沿后读数据是可用的t_{PACC}，在NRD的上升沿后变成'Z'。这种情况下，READ_MODE必须被设置为1（读由NRD控制），来指示数据在NRD的上升沿是可用的。SMC在产生NRD上升沿的主控的上升沿上的内部采样读数据，无论NRD的编程波形可能是什么。

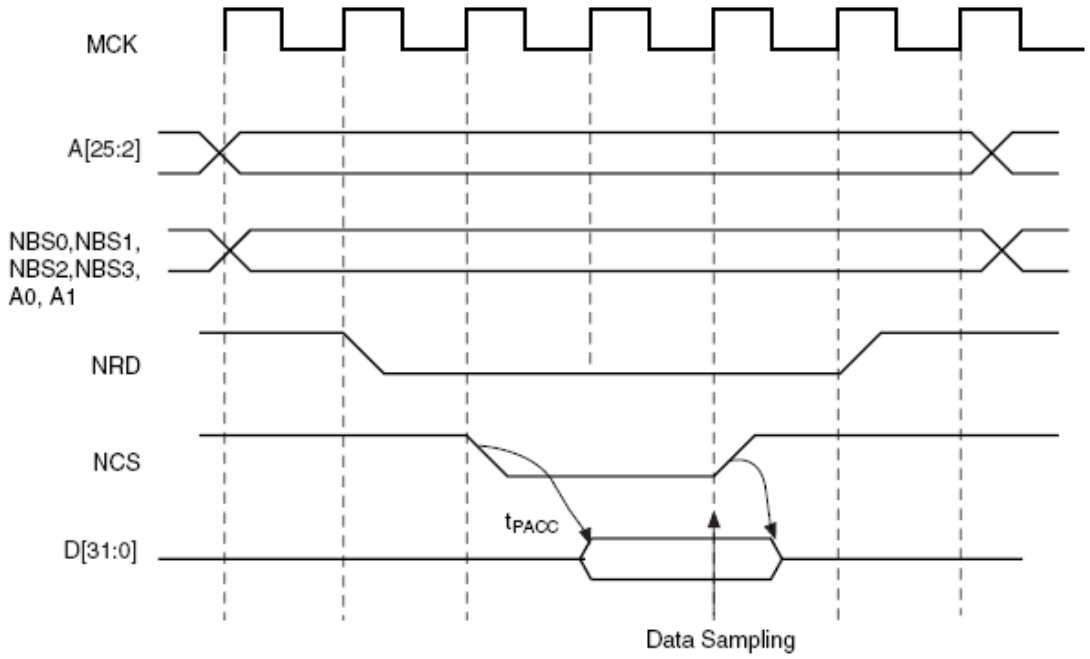
图21-10 READ_MODE = 1:NRD上升沿前数据由SMC采样



21.8.2.2 读由NCS(READ_MODE=0) 控制

图21-11展示了LCD模块的一个典型读周期。读数据在NCS下降沿后是有效的t_{PACC}并且维持有效直到NCS的上升沿。数据必须在NCS上升时被采样。此情况下，READ_MODE必须被设置为0（读由NCS控制）：SMC在产生NCS上升沿的主控时钟的上升沿内部采样数据，无论NRD的编程波形可能是什么。

图21-11 READ_MODE = 0:数据在NCS上升沿前由SMC采样



21.8.3 写波形

写协议和读协议类似。如图21-12描述。写周期在存储地址总线上同地址设置一起开始。

21.8.3.1 NWE 波形

NWE信号由设定定时，脉冲宽度和保持定时描述。

1． NWE_SETUP:NWE设定时间被定义为NWE下降沿以前地址和数据的设定。

2． NWE_PULSE:NWE脉冲长度是NWE下降沿和NWE上升沿之间的时间。

3． NWE_HOLD:NWE保持时间被定义为NWE上升沿后地址和数据的保持时间。

NWE波形应用于字节写访问模式中的所有字节写块: NWR0-NWR3。

21.8.3.2 NCS波形

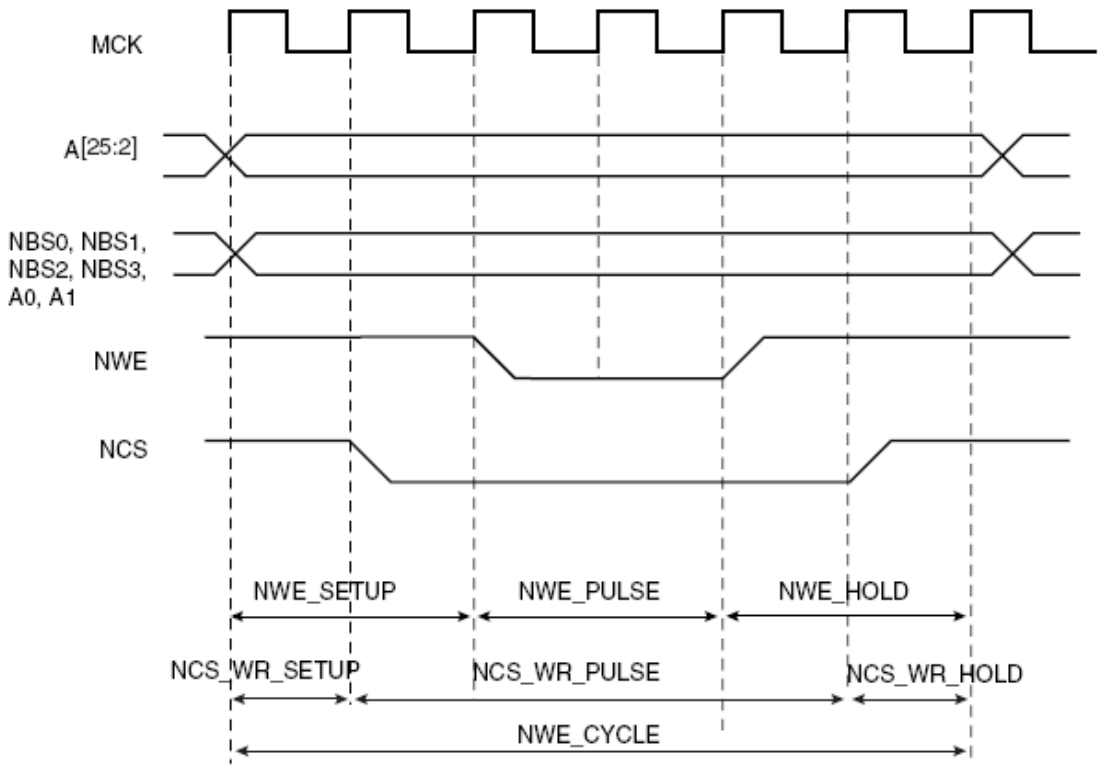
写操作中NCS信号波形和那些应用在读操作中的不同，但是 被独立的定义：

1． NCS_WR_SETUP:NCS设定时间被定义为NCS下降沿前地址的设定时间。

2． NCS_WR_PULSE:NCS脉冲长度是NCS下降沿和NCS上升沿之间的时间

3． NCS_WR_HOLD:NCS保持时间被定义为NCS上升沿后地址的保持时间。

图21-12 写周期



21.8.3.3 写周期

写周期时间被定义为写周期总的持续时间，就是从地址总线上的地址置位到地址可能改变的点的时间。总的写周期时间等于：

NWE_CYCLE = NWE_SETUP + NWE_PULSE + NWE_HOLD

= NCS_WR_SETUP + NCS_WR_PULSE + NCS_WR_HOLD

所有的NWE和NCS（写）定时独立的给每个片选定义为整的主控时钟数。

为保证NWE和NCS定时一致，用户必须定义总的写周期代替保持定时。这里隐含的定义了NWE保持时间和NCS（写）保持时间为：

NWE_HOLD = NWE_CYCLE - NWE_SETUP - NWE_PULSE

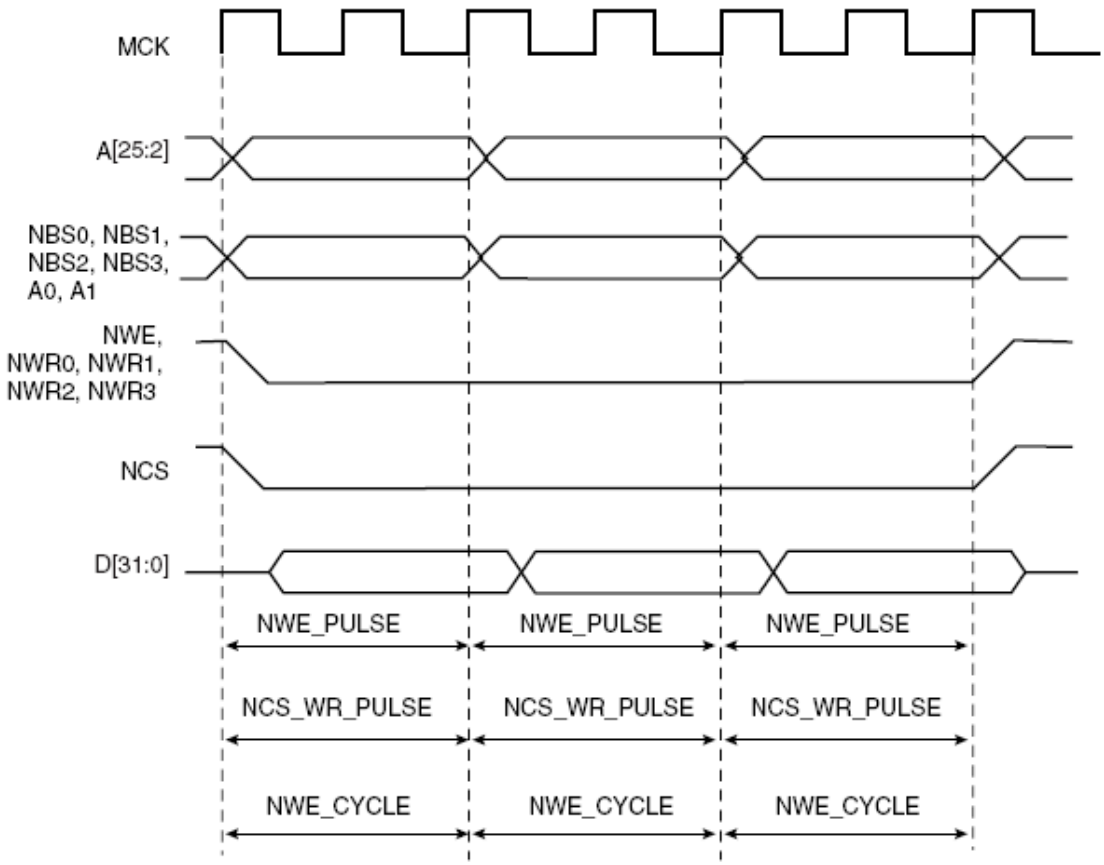
NCS_WR_HOLD = NWE_CYCLE - NCS_WR_SETUP - NCS_WR_PULSE

21.8.3.4 空延迟设定和保持

如果空设定参数对NWE和/或NCS编程，NWE和/或NCS维持继续有效，假如

在同样的存储器（见图21-13）中连续的写周期。然而，对于在NWE或NCS上升沿执行写操作的设备，如SRAM，必须编程一个设定或一个保持。

图21-13 写周期中NCS和NWE的空设定和保持值



21.8.3.5 空脉冲

编程空脉冲是不被允许的。脉冲必须至少被设定为1。一个空值导致不可预期的动作。

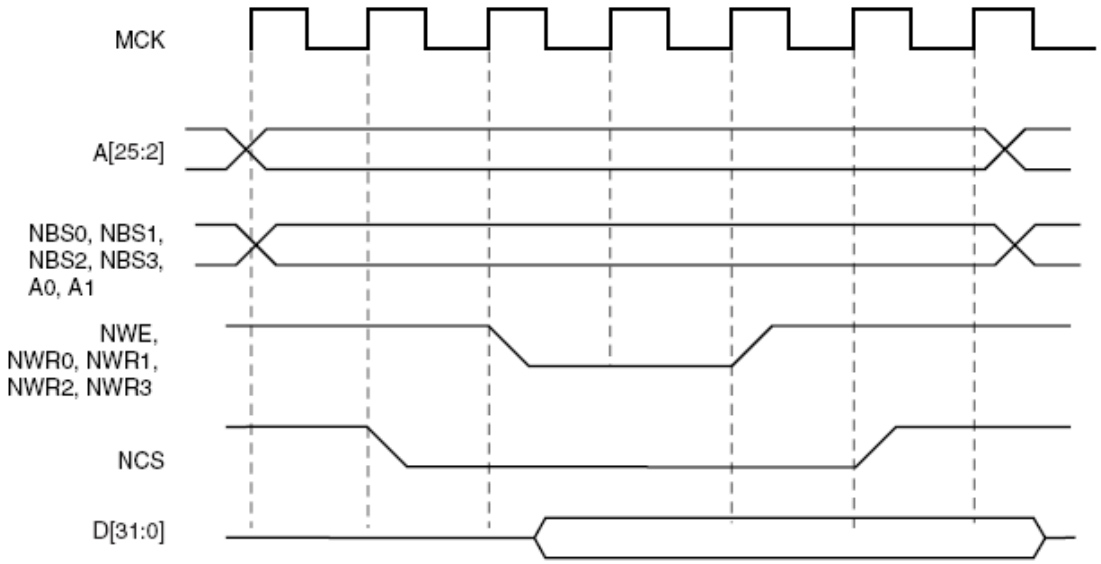
21.8.4 写模式

对应片选的SMC_MODE寄存器中的WRITE_MODE参数指示了哪个信号控制写操作。

21.8.4.1 写被NWE（WRITE_MODE=1）控制

图21-14展示了一个WRITE_MODE设置为1的写操作的波形。数据在脉冲和NWE信号的保持阶段被放入总线。内部数据缓冲在NWE_SETUP时间被关闭，并且知道写周期结束，不必关心NCS上的编程波形。

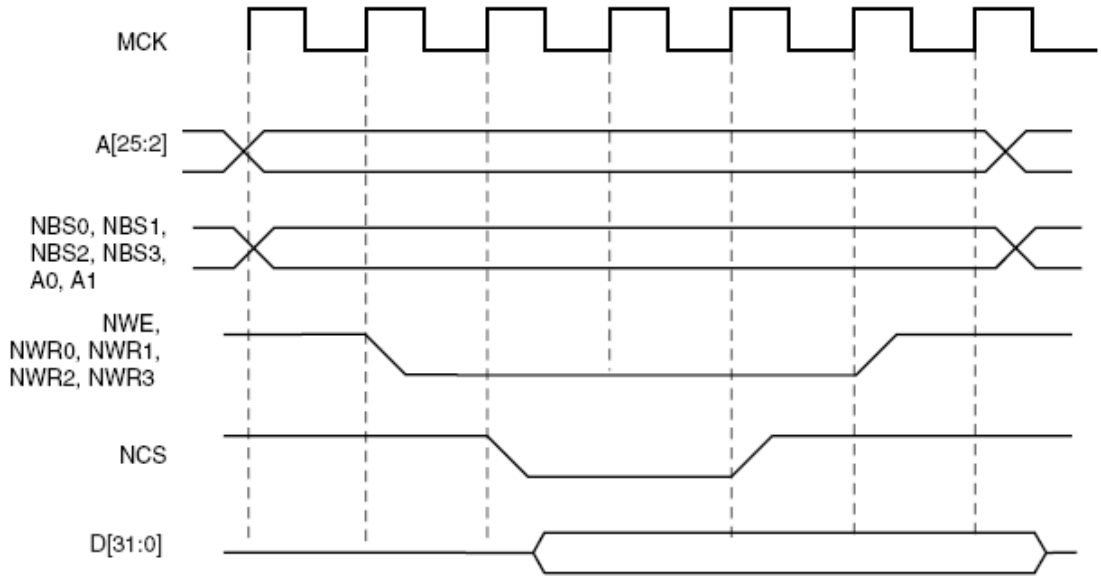
图21-14 WRITE_MODE = 1。 写操作有NWE控制



21.8.4.2 写有NCS（WRITE_MODE=0）控制

图21-15展示了一个WRITE_MODE设置为0的写操作的波形。数据在脉冲和NCS信号保持阶段被放入总线。内部数据缓冲在NCS_WR_SETUP时间被关闭，直到写周期结束，不比关心NWE上的编程波形。

图21-15 WRITE_MODE = 0。 写操作有NCS控制



21.8.5 编码定时操作

所有的定时参数被定义给一个片选并根据其类型在一个SMC_REGISTER被分组。

SMC_SETUP寄存器组所有的设定参数的定义：

NRD_SETUP, NCS_RD_SETUP, NWE_SETUP, NCS_WR_SETUP

SMC_PULSE寄存器组所有脉冲参数的定义：

NRD_PULSE, NCS_RD_PULSE, NWE_PULSE, NCS_WR_PULSE

SMC_CYCLE寄存器组所有周期参数的定义：

NRD_CYCLE, NWE_CYCLE

表21-4 展示了定时参数怎样被编码和他们被允许的范围。

表21-4 编码和定时参数的范围

编码值	位数	有效值	允许范围	
			编码值	有效范围
setup [5:0]	6	128 x setup[5] + setup[4:0]	0 ≤≤31	128 ≤≤128+31
pulse [6:0]	7	256 x pulse[6] + pulse[5:0]	0 ≤≤63	256 ≤≤256+63
cycle [8:0]	9	256 x cycle[8:7] + cycle[6:0]	0 ≤≤127	256 ≤≤256+127 512 ≤≤512+127 768 ≤≤768+127

21.8.6 定时参数的复位值

表21-5 给出了定时参数在复位时的缺省值。

表21-5定时参数的复位值

寄存器	复位值	
SMC_SETUP	0x00000000	所有的设定定时被设置为1
SMC_PULSE	0x01010101	所有的脉冲定时被设置为1
SMC_CYCLE	0x00010001	读和写操作持续3个主控时钟周期并提供一个保持周期
WRITE_MODE	1	写用NWE控制
READ_MODE	1	读用NRD控制

21.8.7 用法限定

SMC不检查用户编程参数的有效性。如果SETUP和PULSE参数的和大于对应的CYCLE参数，这将导致SMC不可预期的动作。

对读操作：

空的但是正地址设定和保持和NRD和/或NCS在存储器接口不能被保证，因为通过外部逻辑和填充块的这些信号的传播延迟。如果正设定和保持值必须验证，强烈推荐编程非空值可以来屏蔽在地址，NCS和NRD信号间的变形。

对写操作：

如果空保持值在NWE上被编程，SMC在NWE上升沿后可以保证一个正的地址，字节选择块，和NCS信号的保持。这仅对WRITE_MODE =1正确。见179页“前期读等待状态”。

对读和写操作：脉冲参数的一个空值是禁止的并可导致不可预期的动作。

在读和写周期，参考地址总线定义设定和保持时间参数。对外部设备，在

NCS和NRD信号（读）之间，或在NCS和NWE信号（写）之间需要设定和保持时间，这些设定和保持时间必须转换成参照地址总线的设定和保持时间。

21.9 自动等待状态（Automatic Wait States）

在确定的环境下，SMC自动的在访问避免总线竞争或操作冲突间插入空闲周期 。

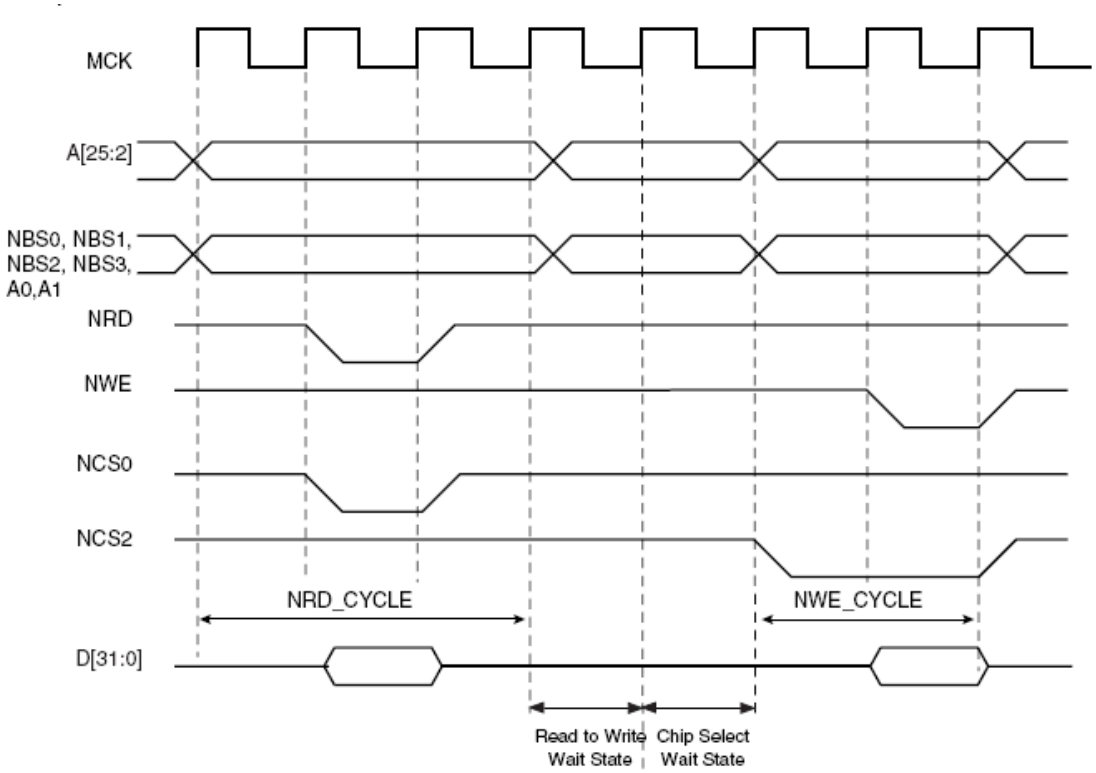
21.9.1 片选等待状态

SMC总在分离的片选上2个转换间插入一个空闲周期。此空闲周期确保在一个设备的重激活和下一设备的激活间没有总线竞争。

片选等待期间，把所有控制块变成待用的：NBS0 到 NBS3, NWR0 到 NWR3, NCS[0..7], NRD块被设置成1。

图21-16说明了访问片选0和片选2之间的一个片选等待状态。

图21-16 一个读访问NCS0和一个写访问NCS2之间的片选等待状态



21.9.2 早读等待（Early Read Wait State）

一些情况下，SMC在一个写访问和一个读访问之间插入一个等待状态来允许写周期到结束的时间在随后的读周期开始前。此等待状态不产生附加的片选等待状态。读周期因此仅发生在一个写和读访问到相同的设备（相同的片选）之间。

如果至少以下一种情况有效，则早读等待状态自动的插入：

1． 如果写控制信号无保持时间并且读控制信号无设定时间（图12-17）。

2． 在NCS写控制状模式(WRITE_MODE = 0)，如果NCS信号上无保持定时，并且NCS_RD_SETUP参数被设置为0，则无所谓读模式（图21-18）。写操作必须随NCS上升沿结束。没有早读等待状态，则写操作也不完全正确。

3． 在NWE控制模式(WRITE_MODE = 1)，如果无保持定时(NWE_HOLD =0)，写控制信号的反馈被用于控制地址，数据，片选和字节选择块。如

果外部写控制信号未撤销可能由于装载电容，一早读等待状态被插入，地址，数据和控制信号被维持在一个周期以上。见图21-19。

图21-17 早读等待状态：无保持写紧跟无设定读

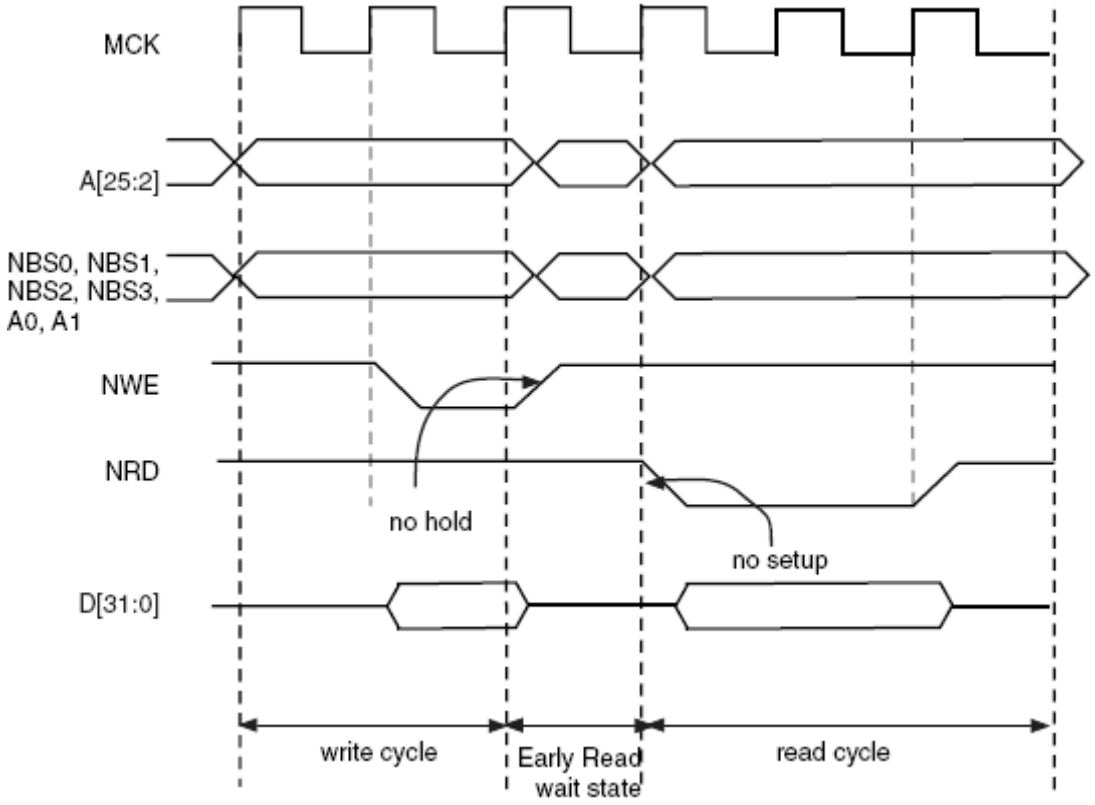


图21-18 早读等待状态：NCS控制的无保持的写紧跟一个无NCS设定的读

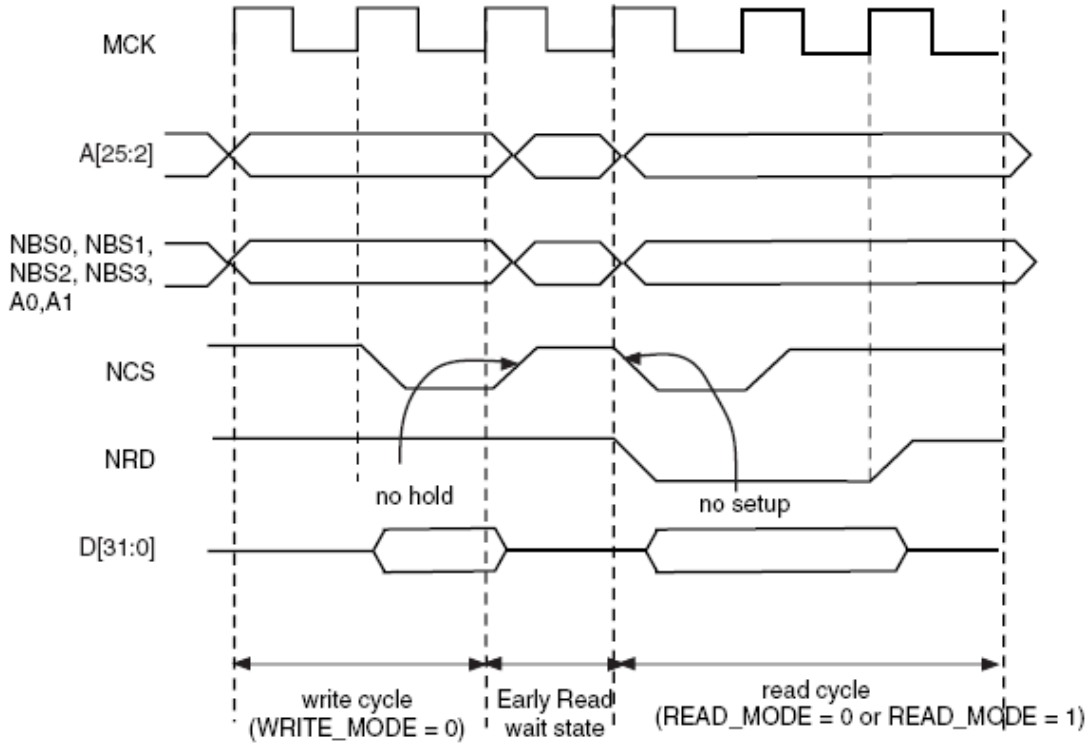
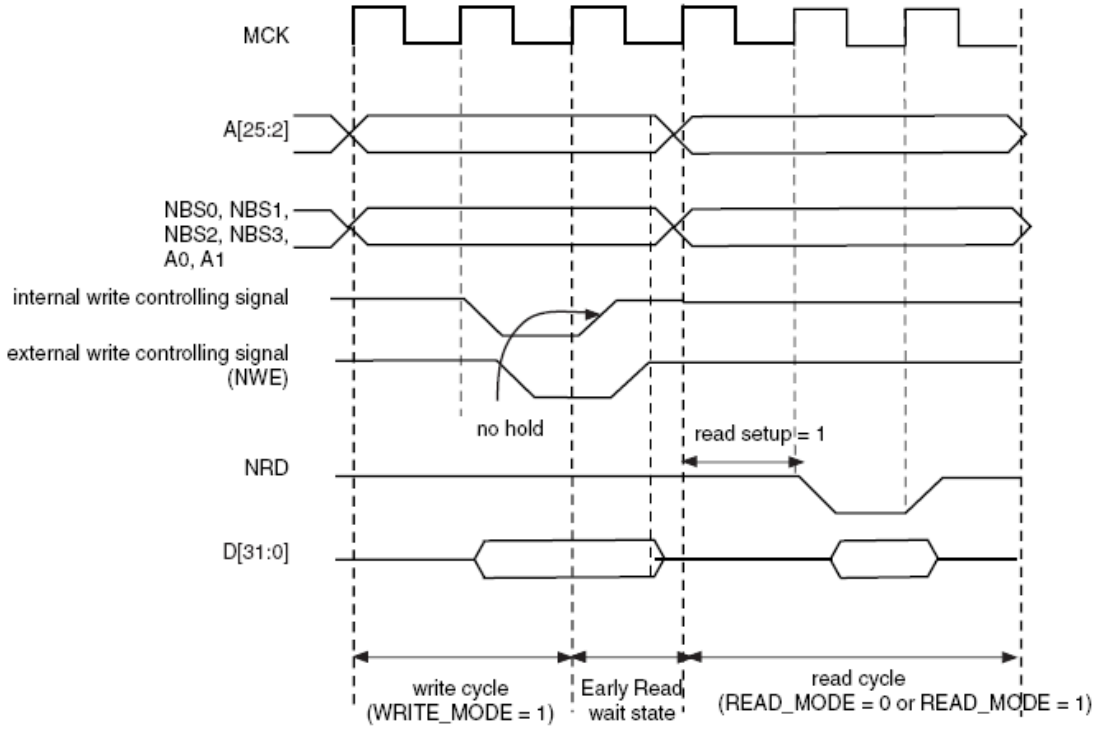


图21-19 早读等待状态：NWE控制的无保持写紧跟一个设定周期的读



21.9.3 重装载用户配置等待状态

用户可以通过写SMC用户接口改变任何任何配置参数。

当在用户接口检测到写入一个新的用户配置，SMC在开始下一个访问前插入一个等待状态。所谓的“重装载用户配置等待状态”被SMC用于装载新的参数集应用于下一次访问。

重装载配置等待状态不用于附加的片选等待状态。如果重编程用户接口前和后的访问被制定为不同设备（片选），接着，一个单片选等待状态被应用。另外，如果写用户接口前和后的访问被制定为相同的设备，重装载配置等待状态将被插入，即使变化并不影响当前片选。

21.9.3.1 用户程序

为插入一个重装载配置等待状态，SMC检测一个到用户接口的任何SMC_MODE寄存器的写访问。如果用户仅在用户接口修改定时寄存器(SMC_SETUP, SMC_PULSE, SMC_CYCLE 寄存器)，用户必须通过写SMC_MODE激活修改，即使模式参数上无改变。

21.9.3.2 慢时钟模式转变

当前转换结束后，当进入或退出慢时钟模式，重装载配置等待状态也被插入。（见193页“慢时钟模式”）

21.9.4 读取写等待状态

由于一个内部途径，一个等待周期总在连续读和写SMC访问间被插入。

此等待周期涉及在此文档读取写等待状态

当等待状态将被插入，此等待周期被用于附加片选和重装载用户配置等待状态。见179页图21-16。

21-10 数据浮动等待状态（Data Float Wait States）

一些存储设备不能快速的释放外部总线。对这样的设备，有必要在读访问后增加等待状态（数据浮动等待状态）：

1．开始读访问一个不同的外部存储器之前

2．开始写访问相同的设备或一个不同的外部设备。

每个外部存储设备的数据浮动输出时间(t_{DF})被编程于对应片选寄存器的TDF_CYCLES域中。TDF_CYCLES值指示外部设备释放总线前数据浮动等待周期数（0到15之间），并且表示允许数据输出在存储器使无效后到高阻抗的时间。

数据浮动等待状态不延迟内部存储器访问。因此，一个用长t_{DF}单访问外部存储器将减缓从内部存储器执行一个程序。

数据浮动等待状态管理依赖于对应片选SMC_MODE寄存器的READ_MODE和 TDF_MODE域。

21.10.1 READ_MODE

置位READ_MODE为1指示SMC，NRD信号负责关闭外部存储设备的三态缓冲器。数据浮动周期接着在NRD信号上升沿后开始并且持续TDF_CYCLES MCK周期。

当读操作由NCS信号(READ_MODE = 0)控制，在NCS上升沿后数据总线仍然繁忙期间，TDF域给出MCK周期数。

图21-20说明假设一个2周期(TDF_CYCLES = 2)的数据浮动周期，NRD控制模式(READ_MODE =1)中Data Float Period。图21-21展示了当被NCS(READ_MODE = 0)控制并TDF_CYCLES参数等于3时的读操作。

图21-20 NRD控制的读访问（TDF = 2）中TDF周期

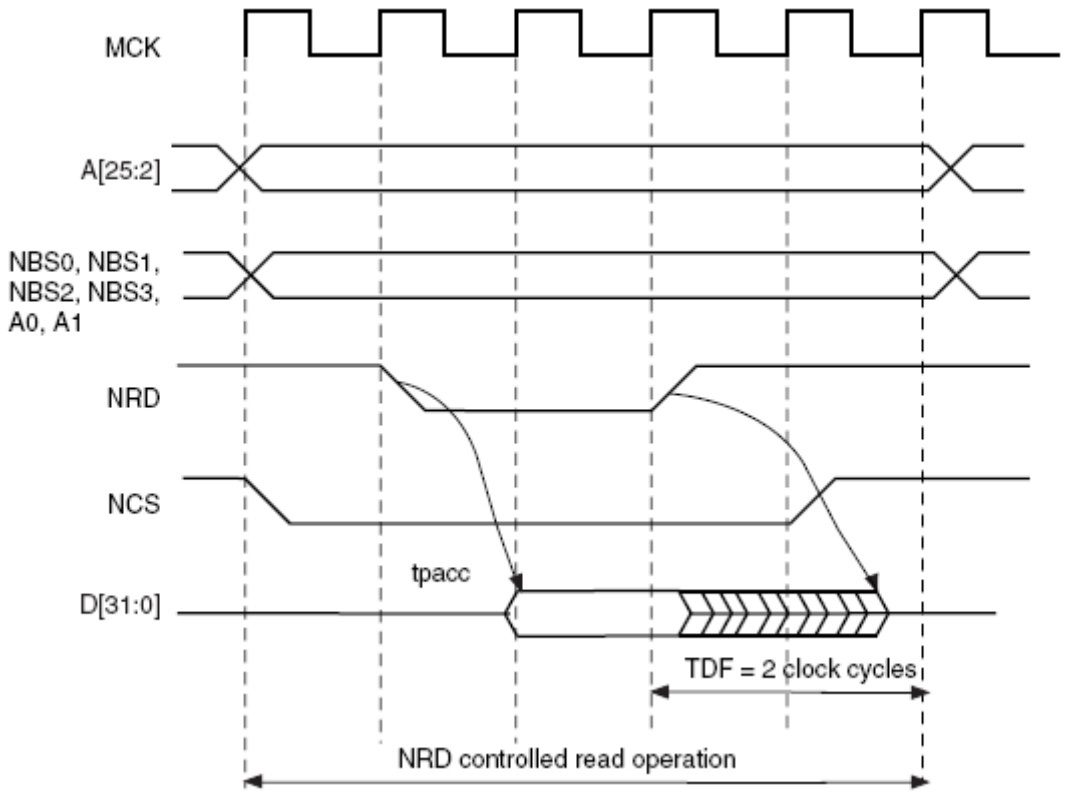
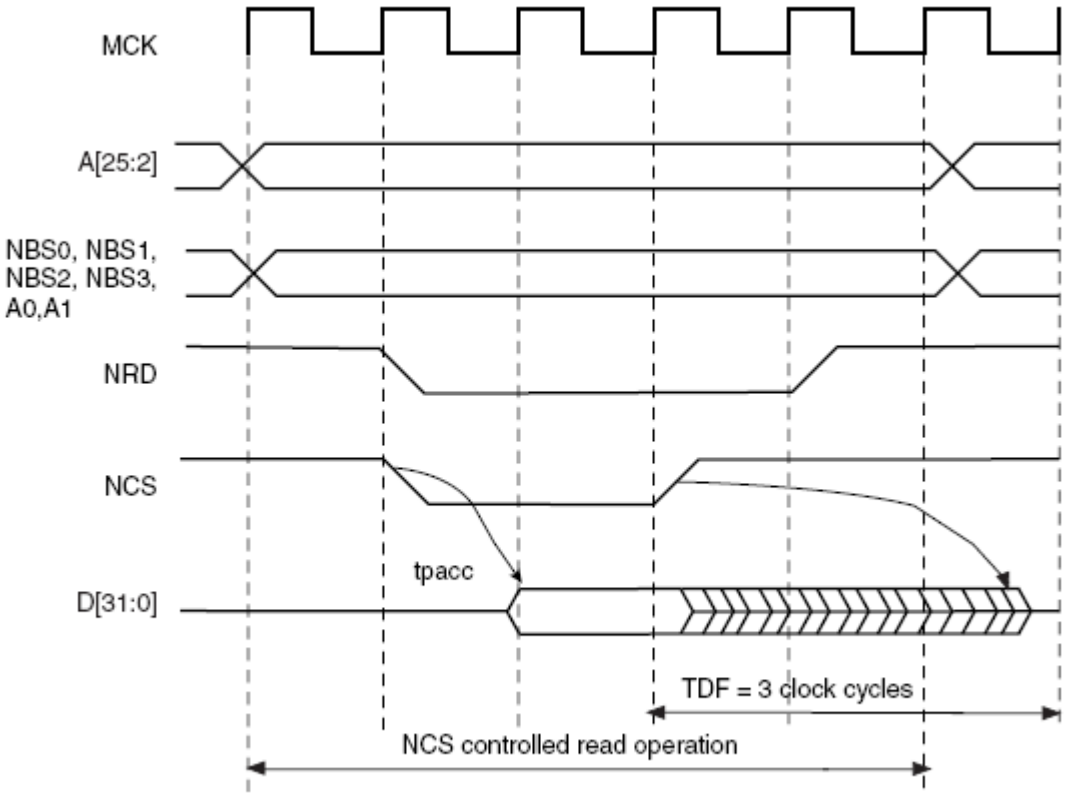


图21-21 NCS控制的读访问（TDF =3）中TDF周期



21-10　TDF最优化使能(TDF_MODE = 1)

当SMC_MODE寄存器的TDF_MODE模式被置位为1（TDF最优化被使能），SMC利用下一个访问的设定周期来优化插入等待状态周期数。

图21-22展示了片选0上，由NRD控制的读访问，紧跟由NWE控制的写访问。

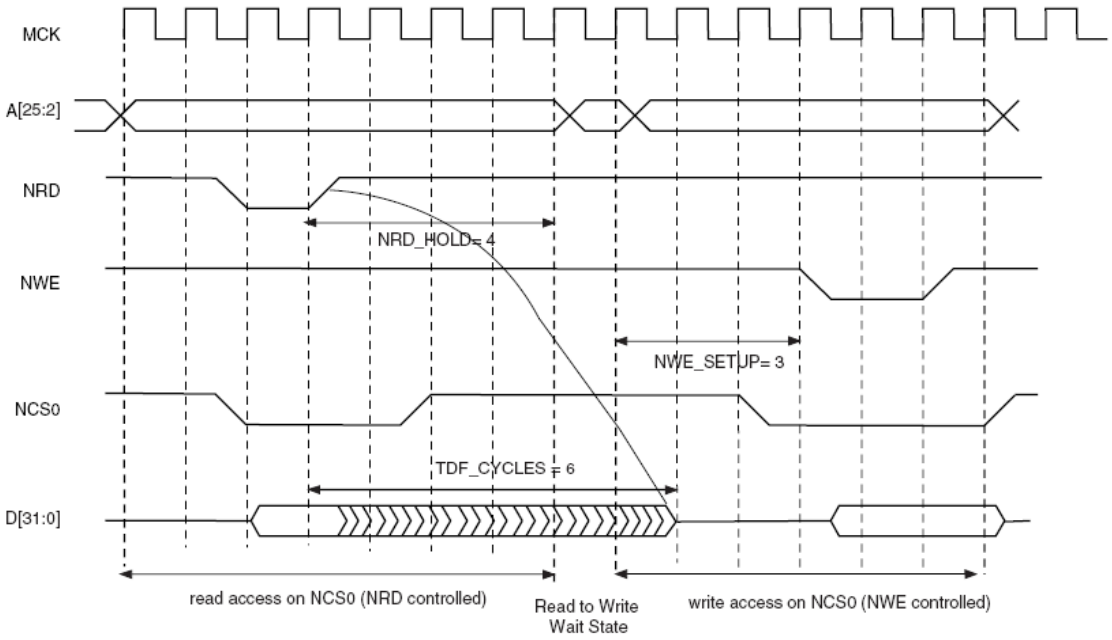
片选0已被编程：

NRD_HOLD = 4; READ_MODE = 1 (NRD 控制)

NWE_SETUP = 3; WRITE_MODE = 1 (NWE 控制)

TDF_CYCLES = 6; TDF_MODE = 1 (最优化使能).

图21-22　TDF最优化：如果当下一个访问开始TDF周期溢出则无等待状态被插入



21.10.2 TDF最优化使无效（TDF_MODE=0）

当最优化被使无效，TDF等待状态在读传输结尾被插入，所以当第二次访问开始时数据浮动周期结束。如果读控制信号的保持周期和数据浮动周期重叠，则将无附加的TDF等待状态被插入。

图21-23，图21-24和图21-25说明以下情况：

无TDF最优化

1．读访问紧跟另外一个片选上的读访问

2．读访问紧跟另外一个片选上的写访问

3．读访问紧跟同一个片选上的写访问

图21-23 TDF最优化使无效(TDF Mode = 0)。不同片选上2个读访问间的TDF等待状态

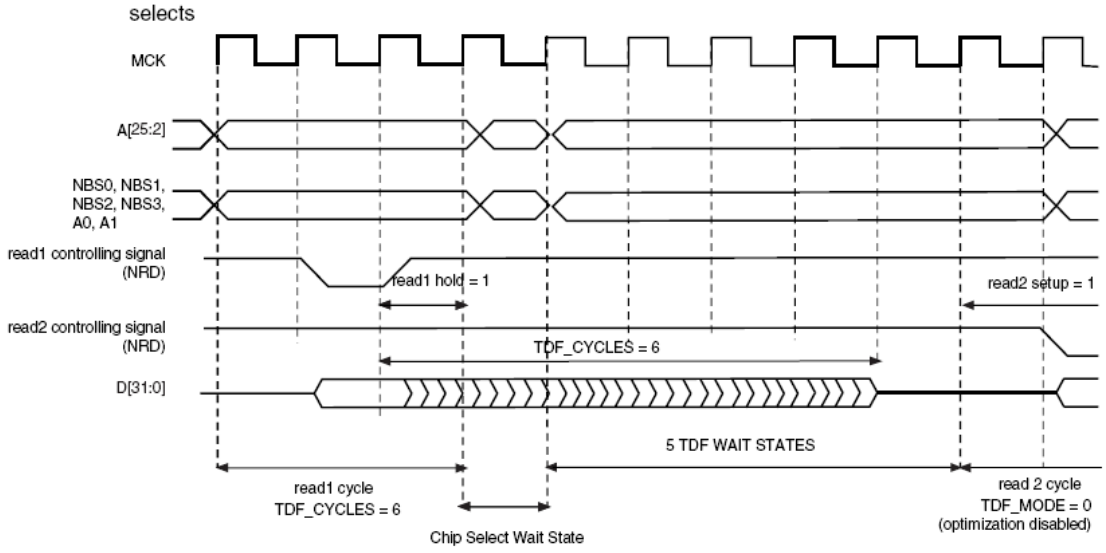


图21-24 TDF Mode = 0: 不同片选上2个写访问间的TDF等待状态

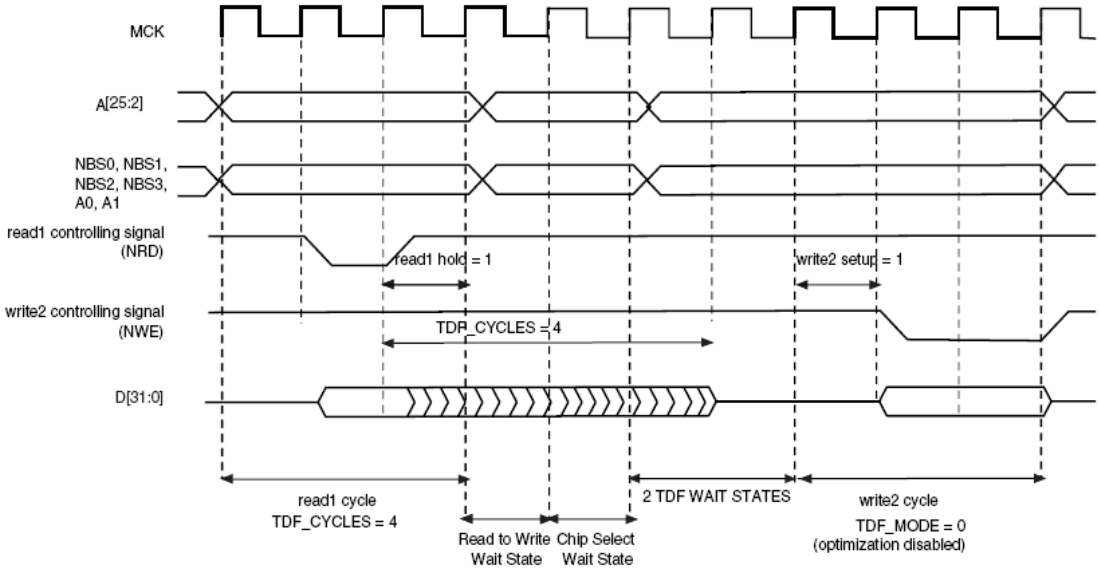
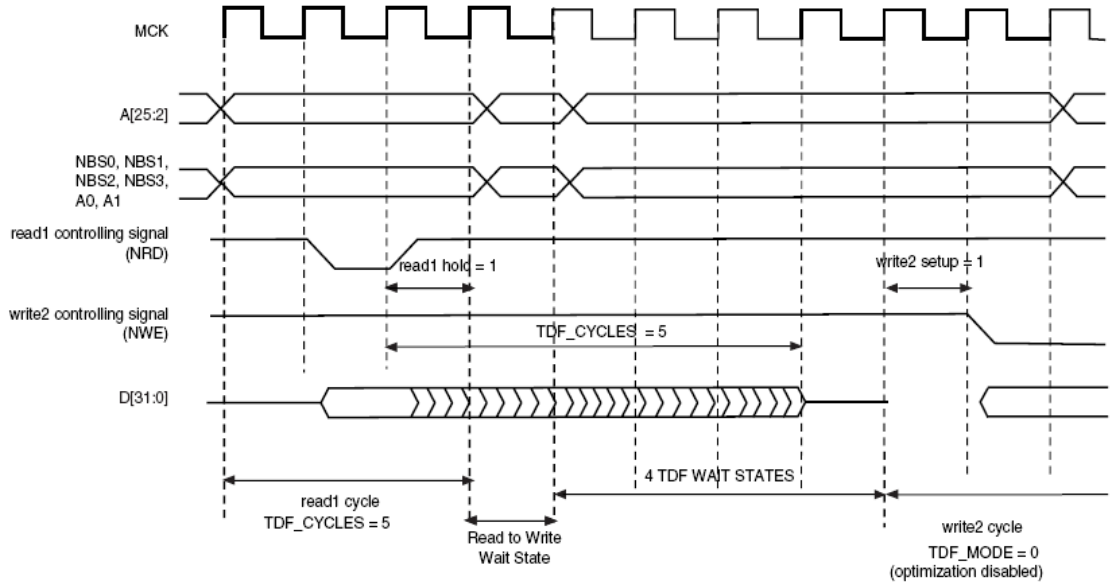


图21-25 TDF Mode = 0: 相同片选上2个读访问间的TDF等待状态



21.11 外部等待

任何访问被外部设备用SMC的NWAIT输入信号扩展。对应片选上SMC_MODE寄存器的EXNW_MODE域必须被设置为“10”（冻结模式）或“11”（就绪模式）。当EXNW_MODE被设置为“00”（使无效），NWAIT信号在对应片选上被简单的忽略。NWAIT信号延迟读或写操作关于读或写控制信号，依赖于对应片选的读和写模式。

21.11.1 限定

当EXNW_MODE之一被使能，对读/写控制信号强制性编程至少一个保持周期。因此，NWAIT信号不能被使用在页面模式（196页“异步页面模式”），或在慢时钟模式（193页“慢时钟模式”）。

NWAIT信号假设是对外部设备SMC读写请求的一个响应。NWAIT信号仅被SMC在读或写控制信号的脉冲状态被检查。预期周期以外NWAIT信号的激活不影响SMC的任何动作。

21.11.2 冻结模式(Frozen mode)

当外部设备激活NWAIT信号（低电平有效），在此信号的内部同步化后，SMC状态被动结，就是说，SMC计数器被冻结，并且所有的控制信号维持不变。当重同步化NWAIT信号无效，SMC完成访问，从其停止的点重新开始访问。见图21-26。此模式必须当外部设备使用NWAIT信号延迟访问和冻结SMC时被选择。

预期周期以外NWAIT信号的激活被忽略，如图21-27说明。

图21-26 在冻结模式用NWAIT信号写访问(EXNW_MODE = 10)

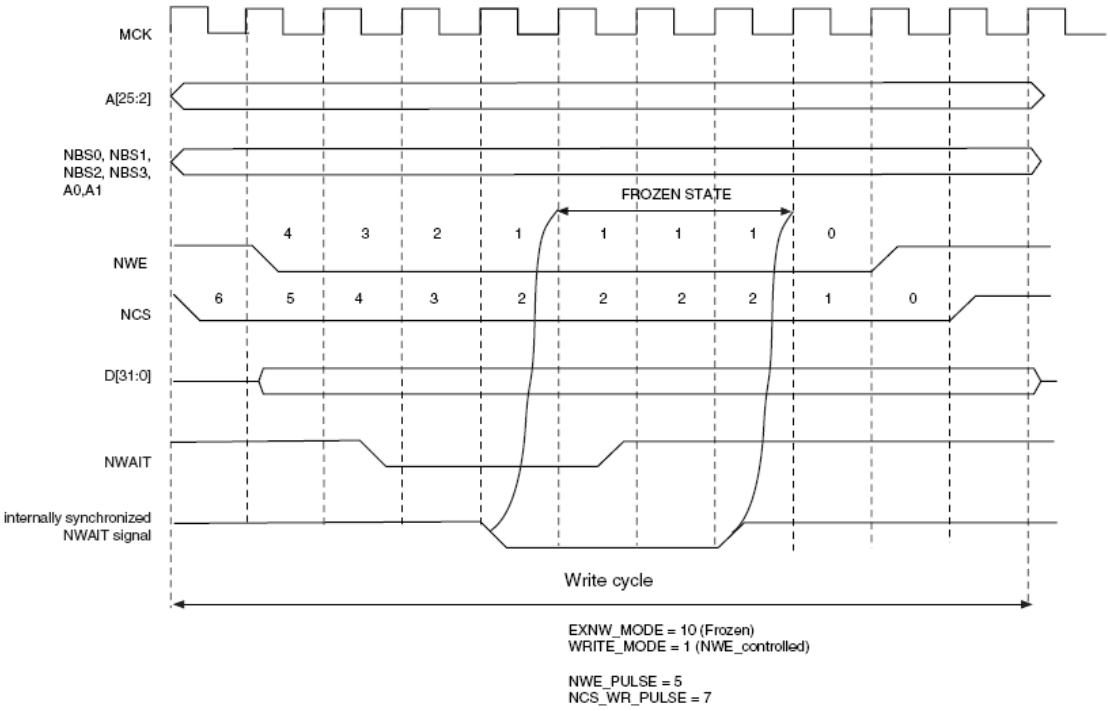
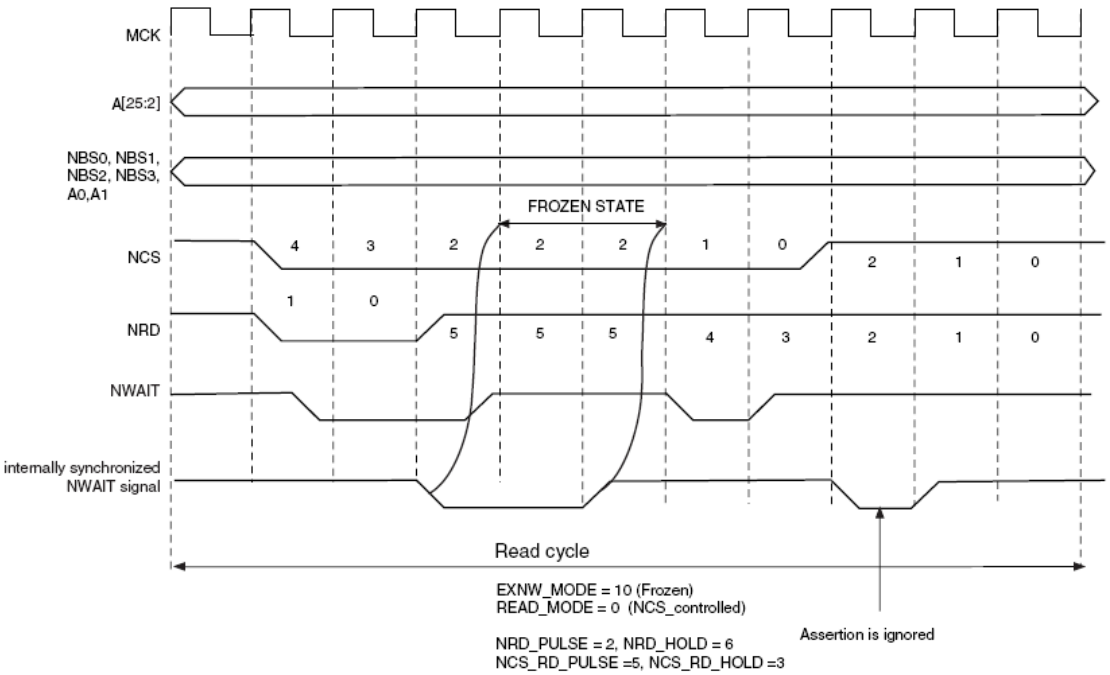


图21-27 在冻结模式用NWAIT信号读访问(EXNW_MODE = 10)



21.11.3 就绪模式(Ready mode)

在就绪模式(EXNW_MODE = 11)，SMC有不同的操作。通常，SMC通过向下计数读/写控制信号的设定和脉冲计数器来开始访问。在脉冲阶段的最后一个周期，检查重同步NWAIT信号。

如果SMC被激活，则待定如图21-28和图21-29中的访问。重新激活后，完成访问：执行访问的保持阶段。

此模式必须当外部设备使用NWAIT信号的重激活时被选择，指示其完成读或写操作的能力。

如果NWAIT信号在脉冲结束以前被重激活，或在控制读/写信号脉冲结束后被激活，它对访问长度无影响，如图21-29所示。

图21-28 写访问中NWAIT有效 ：读模式(EXNW_MODE = 11)

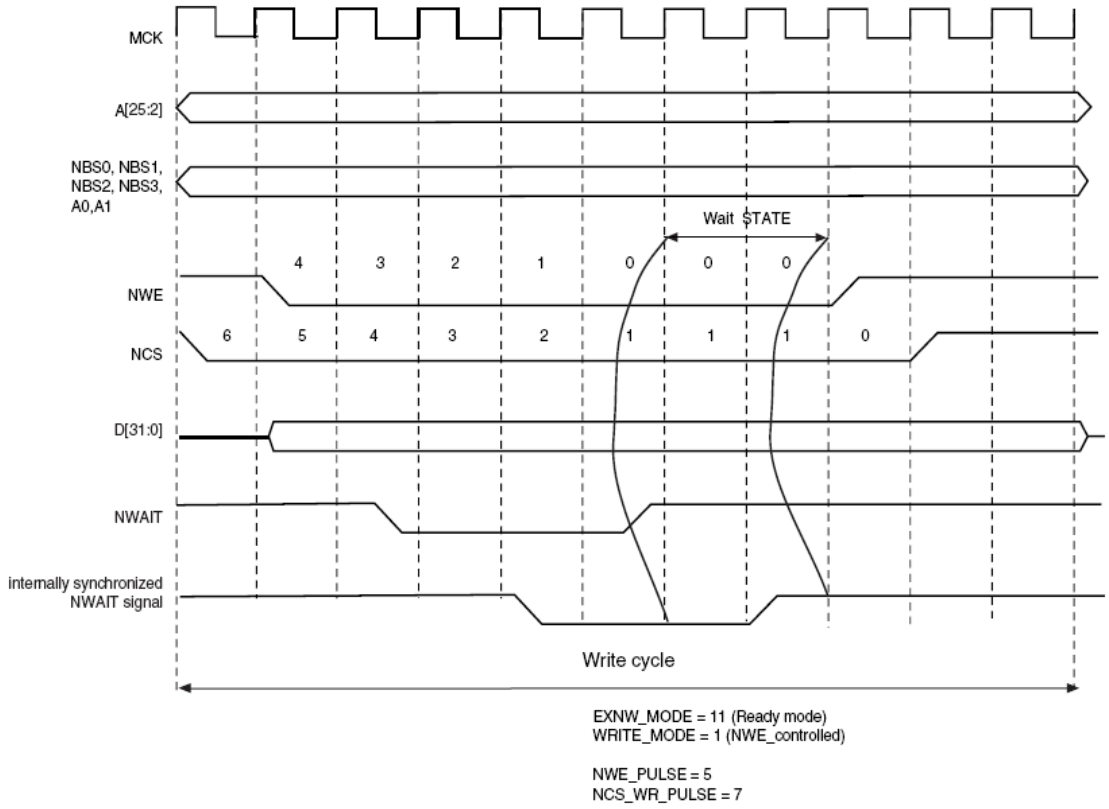
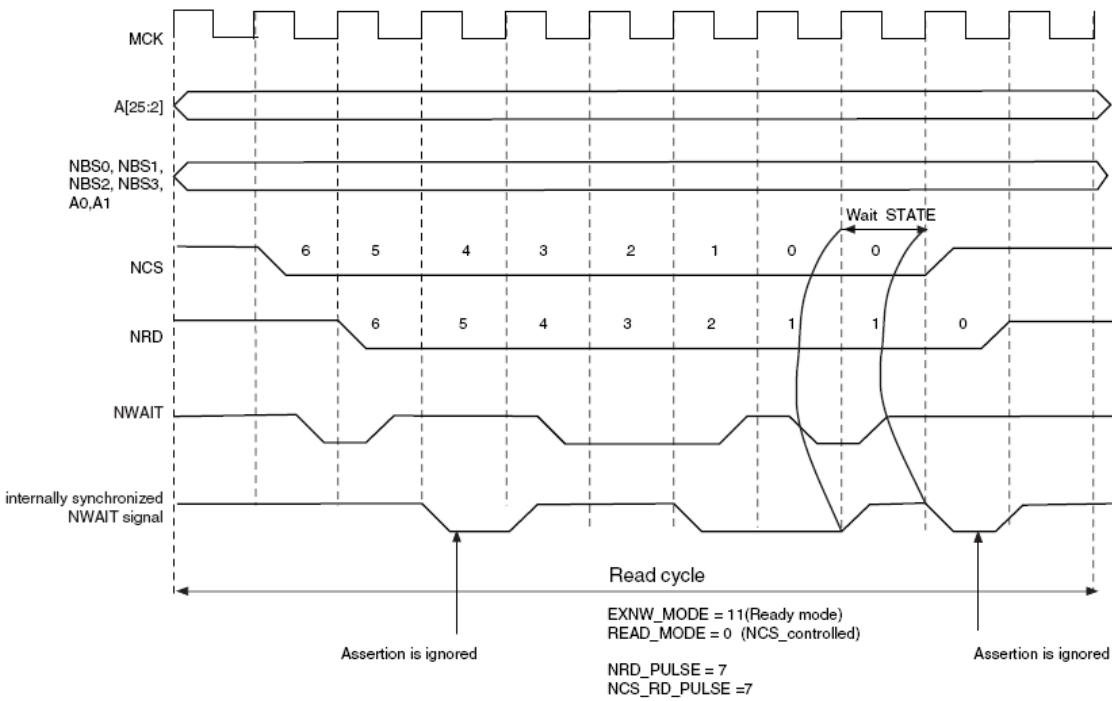


图21-29 读访问中NWAIT有效 ：读模式(EXNW_MODE = 11)



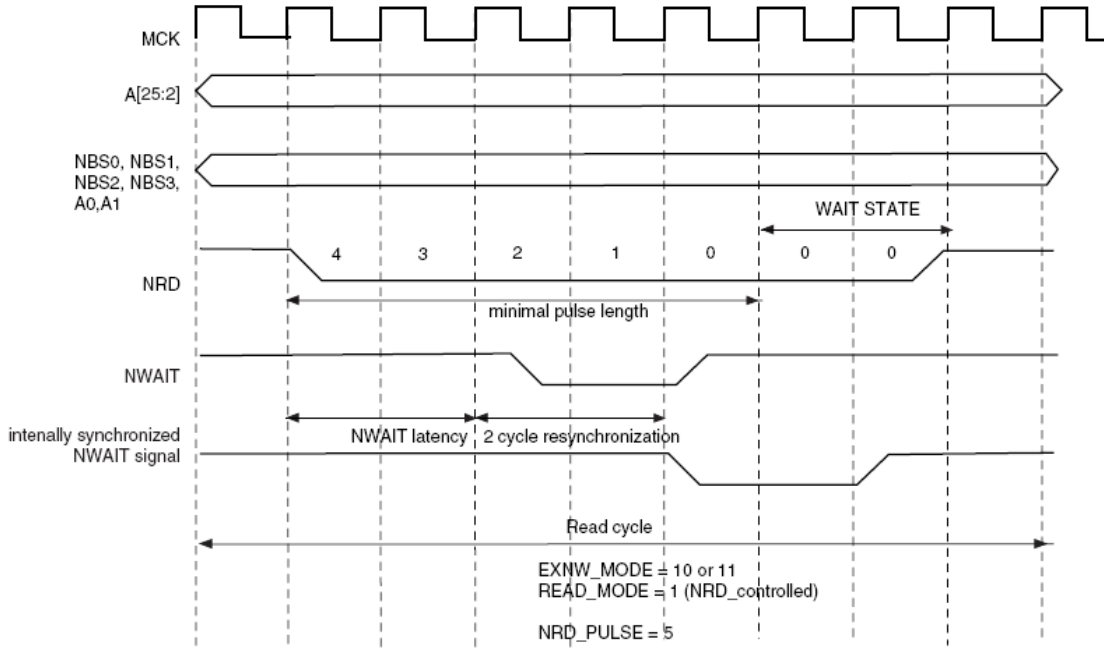
21.11.4 NWAIT等待时间和读/写定时

在读/写控制信号有效和NWAIT信号的设备使能之间可能有一个延迟时间。读/写控制信号的编程脉冲长度必须至少等于此延迟时间加上重同步的2周期+1周期。此外，SMC可以不用检测NWAIT信号有效就进入访问的保持状态。和在就绪模式一样，在冻结模式也是真的。如图21-30说明。

当EXNW_MODE被使能（就绪或冻结），用户必须编程读和写控制信号的脉冲长度，至少：

最小脉冲长度＝NWAIT延迟时间+ 2个重同步周期+1个周期

图21-30 NWAIT延迟时间



21.12 慢时钟模式

当被电源管理控制器驱动的内部信号被激活，SMC可自动运用一套“慢时钟模式”读/写波形，因为MCK已被转变到一个很慢的时钟速率（典型的32kHz时钟速率）。此模式下，用户编程的波形被忽略而慢时钟模式波形被运用。提供此模式以避免在很慢的时钟速率用适当的波形编程用户接口。当有效时，在所有片选上慢时钟有效。

21.12.1 慢时钟模式波形

图21-31说明在慢时钟模式读和写操作。它们在所有片选上有效。表21-6给出了慢时钟模式读和写参数值。

图21-31 慢时钟模式读/写周期

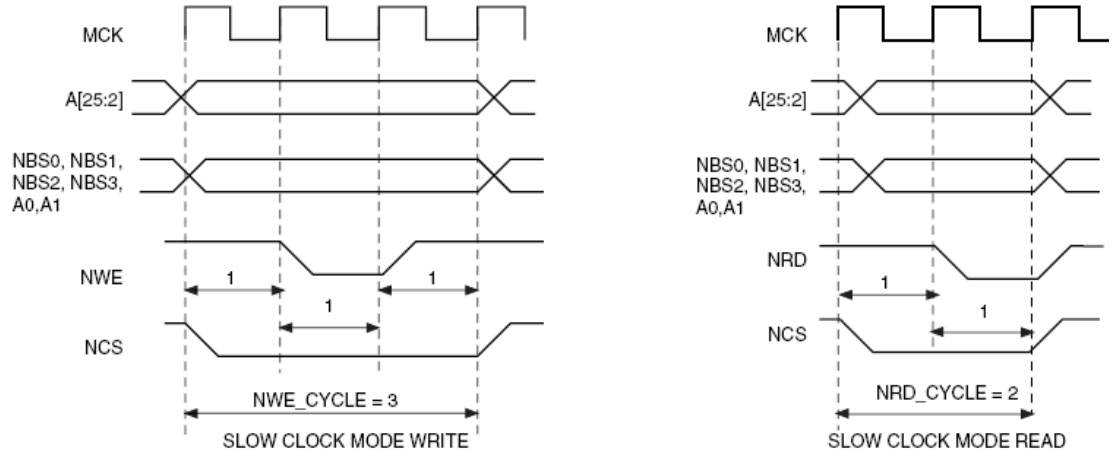


表21-6 慢时钟模式读和写定时参数

读参数	持续时间（周期）	写参数	持续时间（周期）
NRD_SETUP	1	NWE_SETUP	1
NRD_PULSE	1	NWE_PULSE	1
NCS_RD_SETUP	0	NCS_WR_SETUP	0
NCS_RD_PULSE	2	NCS_WR_PULSE	3
NRD_CYCLE	2	NWE_CYCLE	3

21.12.2 从慢时钟变换到正常模式/从正常模式变换到慢时钟模式

当从慢时钟变换到正常模式，当前慢时钟模式传送在高时钟速率用慢时钟模式参数完成。见194页图21-32。外部设备支持这样的定时可能还不够快。

图21-33说明了正确的从一个模式到另一模式转换的推荐程序。

图21-32 当SMC正在执行写操作时时钟速率转换发生

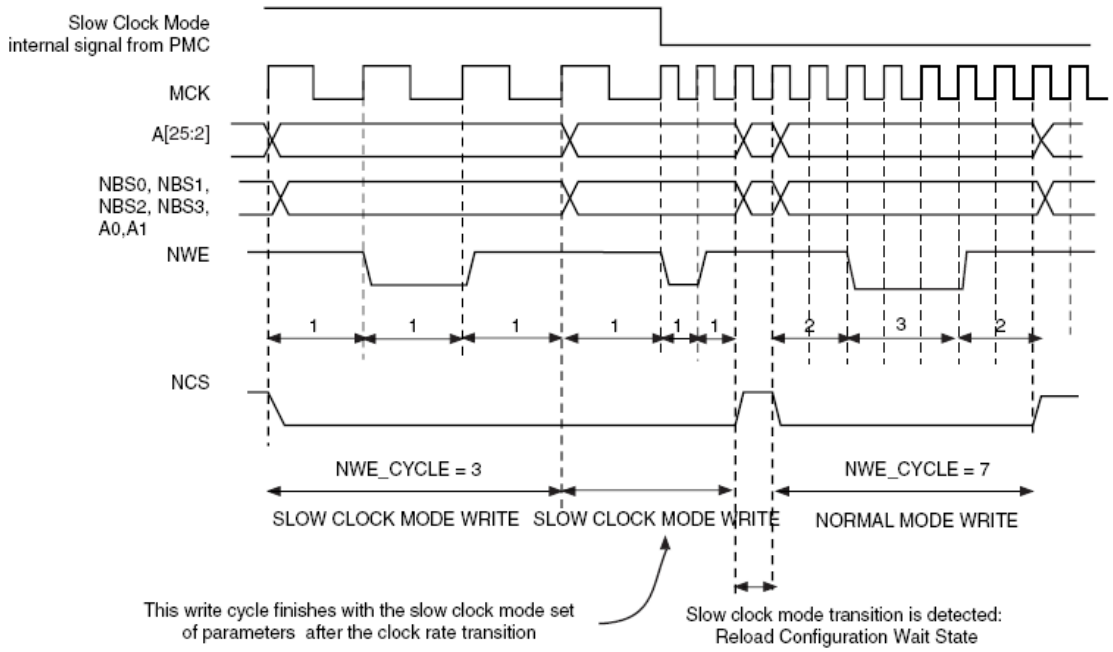
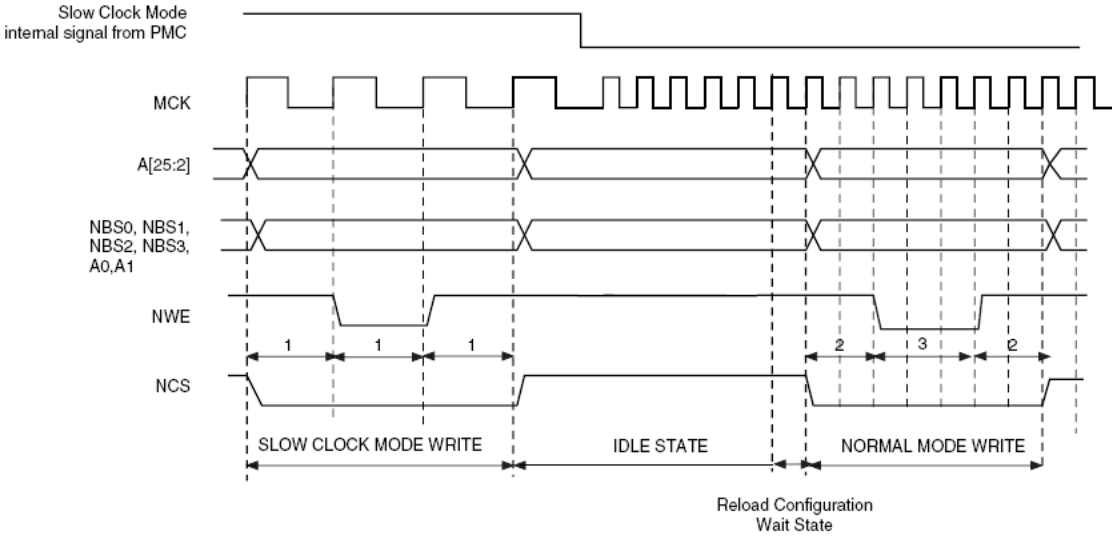


图21-33 从慢时钟变换到正常模式/从正常模式变换到慢时钟模式的推荐程序



21.13 异步页模式

SMC支持在页面模式异步脉冲串读，假如页面模式在SMC_MODE寄存器(PMEN 域)中被使能。页面大小必须在SMC_MODE寄存器中配置为4，8，16或32字节。

页面定义一套连续字节进入存储器。一4字节页面（分别是，8，16，32字节页面）总是和存储器的4字节边界（8，16，32字节边界）对齐。数据字节的MSB在存储器中定义页面地址，地址的LSB在页面中定义数据的地址，细节见表21-7。

页面模式存储设备，首次访问一页(t_{pa})比随后访问页面(t_{sa})所用时间长，如图21-34。当在页面模式，SMC使用户能对首次访问一页以内，随后访问页面内定义不同的读定时。

表21-7　一页中的页面地址和数据地址

页面大小	页面地址	页面中的数据地址
4字节	A[25:2]	A[1:0]
8字节	A[25:3]	A[2:0]
16字节	A[25:4]	A[3:0]
32字节	A[25:5]	A[4:0]

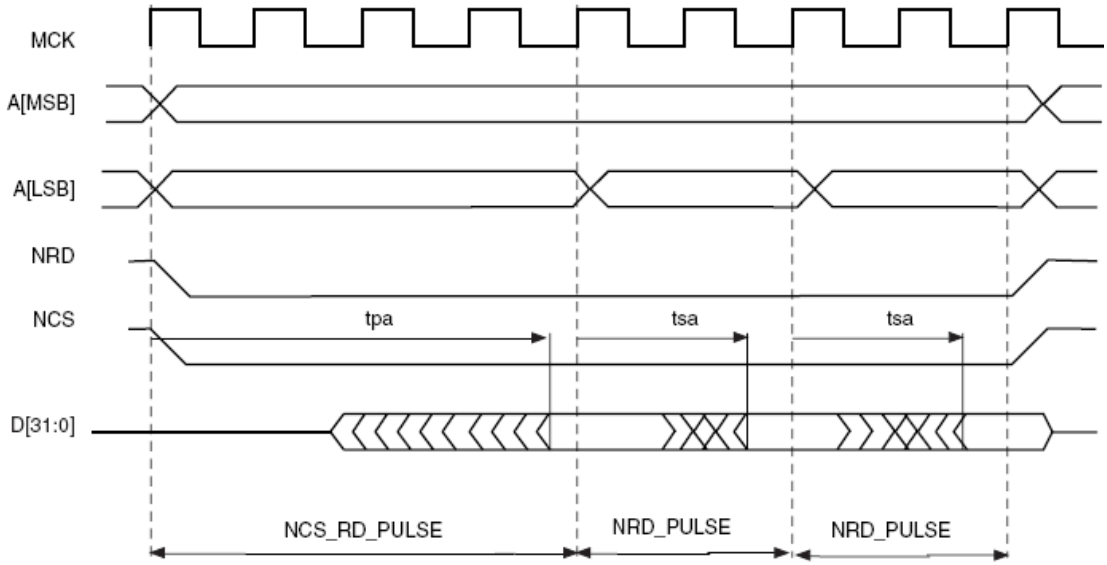
注意：1。A代表存储设备的地址总线

2．对16位设备，地址的0位被忽略。对32位设备，位[1:0]被忽略。

21.13.1　页面模式中的协议和定时

图21-34展示了页面模式访问下的NRD和NCS定时。

图21-34　页面模式读协议（地址MSB和LSB被定义在表21-7）



无论用户模式下设定和保持定时的编程值可能是什么，NRD和NCS信号在所有读传送期间被保持在低电平。此外，NRD和NCS定时相同。首次访问页面的脉冲长度用SMC_PULSE寄存器的NCS_RD_PULSE域定义。随后访问页面内的脉冲长度用NRD_PULSE参数定义。

在页面模式，读定时的编程如表21-8中描述。

表21-8页面模式中读定时的编程

参数	值	定义
READ_MODE	‘x’	无影响
NCS_RD_SETUP	‘x’	无影响
NCS_RD_PULSE	t_{pa}	首次访问页面的访问时间
NRD_SETUP	‘x’	无影响
NRD_PULSE	t_{sa}	随后访问页面的访问时间
NRD_CYCLE	‘x’	无影响

SMC不检查定时的一致性。它将总是运用NCS_RD_PULSE定时作为页面访问定时(t_{pa})并且用NRD_PULSE访问页面(t_{sa})，即使对t_{pa} 的编程值比对t_{sa} 的编程值短。

21.13.2　页面模式下字节访问类型

字节访问类型配置在页面模式保持有效。对16位或32位页面模式设备需要字

节选择信号，配置SMC_REGISTER 的BAT域为0（字节选择访问类型）。

21.13.3 页面模式限定

页面模式和NWAIT信号的使用不兼容。使用页面模式和NWAIT信号可能导致不可预期的动作。

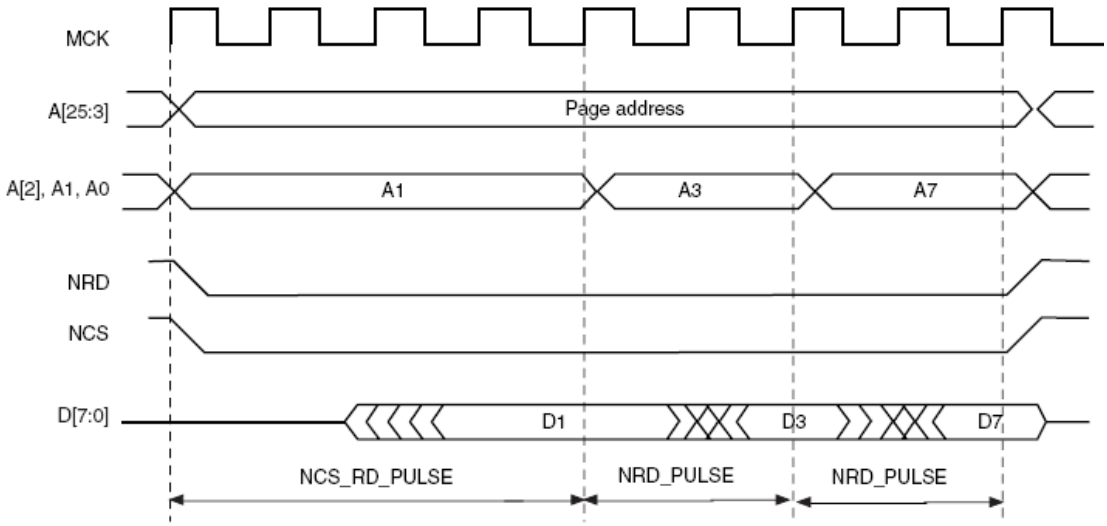
21.13.4 顺序和非顺序访问

如果片选和如表21-7中定义的地址的MSB相同，则当前访问和前一个一样放置在同一个页面，并且无页面损坏发生。

使用此信息，所有同一页面内的数据，顺序的和非顺序的，用一个最小的访问时间(t_{sa})被访问。图21-35说明了用8字节页面在页面模式访问8位存储设备。访问D1导致用一个长访问时间(t_{pa})访问页面。访问D3和D7，虽然不是顺序访问，仅需要一个短访问时间(t_{sa})。

如果地址的MSB不同，SMC执行一新页面的访问。相同的方式下，如果片选和前面访问的不同，将导致页面损坏。如果2个顺序访问到页面模式存储器，但是被另外一个内部和外部外围访问分离，将在第二次访问产生页面损坏，因为设备的片选在两次访问间被重激活。

图21-35 同一页面中访问非顺序数据



21.14 静态存储控制器（SMC）用户接口

SMC用表21-9中所列寄存器编程。对每个片选，4寄存器被用于编程连接寄存器的外部设备的参数。表21-9中，“CS_number”指的是片选数。每个片选需要16字节(0x10)。

用户必须通过写SMC_MODE寄存器中的任何一个来完成写配置。

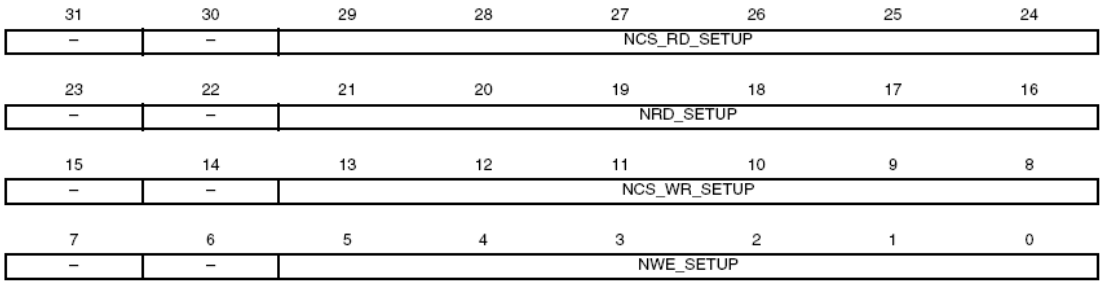
表21-9 SMC寄存器映射

偏移量	寄存器	名称	访问	复位状态
0x10 x CS_number + 0x00	SMC设定寄存器	SMC_SETUP	读/写	0x00000000
0x10 x CS_number + 0x04	SMC脉冲寄存器	SMC_PULSE	读/写	0x01010101
0x10 x CS_number + 0x08	SMC周期寄存器	SMC_CYCLE	读/写	0x00010001
0x10 x CS_number + 0x0C	SMC模式寄存器	SMC_MODE	读/写	0x10001000

21.14.1 SMC设定寄存器

寄存器名称：SMC_SETUP[0 ..7]

访问类型：读/写



1． NWE_SETUP:NWE设定长度

NWE设定长度被定义为：

NWE设定长度= (128* NWE_SETUP[5] + NWE_SETUP[4:0])时钟周期

2． NCS_WR_SETUP：NCS写访问中设定长度

写访问中，NCS信号设定长的被定义为：

NCS设定长度= (128* NCS_WR_SETUP[5] + NCS_WR_SETUP[4:0])时钟周期

3． NRD_SETUP：NRD设定长度

NRD信号设定长度在时钟周期被定义为：

NRD设定长度= (128* NRD_SETUP[5] + NRD_SETUP[4:0])时钟周期

4． NCS_RD_SETUP：读访问中NCS设定长度

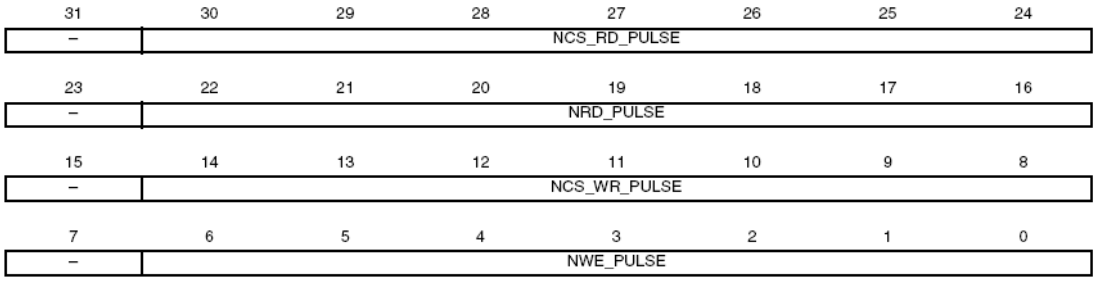
读访问中，NCS信号设定长度被定义为：

NCS设定长度= (128* NCS_RD_SETUP[5] + NCS_RD_SETUP[4:0])时钟周期

SMC脉冲寄存器

寄存器名称：SMC_PULSE[0..7]

访问类型：读/写



1． NWE_PULSE：NWE脉冲长度

NWE信号脉冲长度被定义为：

NWE脉冲长度= (256* NWE_PULSE[6] + NWE_PULSE[5:0])时钟周期

NWE脉冲长度必须至少1个时钟周期

2． NCS_WR_PULSE：写访问中NCS脉冲长度

写访问中，NCS信号脉冲长度被定义为：

NCS脉冲长度= (256* NCS_WR_PULSE[6] + NCS_WR_PULSE[5:0])时钟周期

NCS脉冲长度必须至少1个时钟周期

3． NRD_PULSE：NRD脉冲长度

标准读访问中，NRD脉冲长度在时钟周期中被定义为：

NRD脉冲长度= (256* NRD_PULSE[6] + NRD_PULSE[5:0])时钟周期

NRD脉冲长度必须至少1个时钟周期。

页面模式读访问，NRD_PULSE参数定义了页面中随后访问的持续时间。

4．NCS_RD_PULSE：读访问中NCS脉冲长度

标准读访问中，NCS信号脉冲长度被定义为：

NCS脉冲长度= (256* NCS_RD_PULSE[6] + NCS_RD_PULSE[5:0])时钟周期

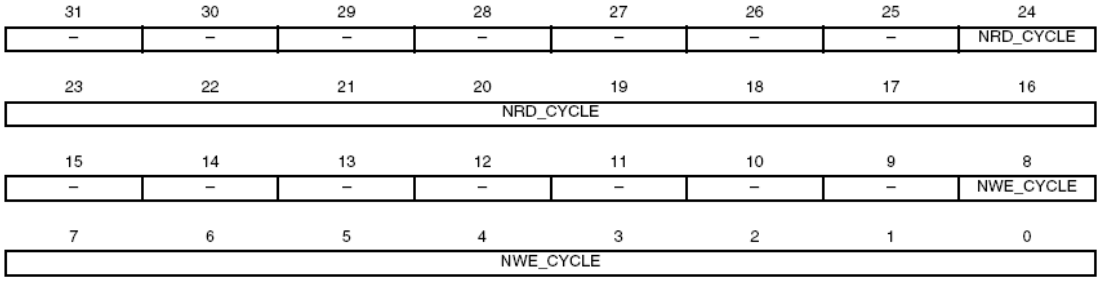
NCS脉冲长度必须至少1个时钟周期

页面模式读访问，NCS_RD_PULSE参数定义首次访问一页的持续时间。

21.14.2　SMC周期寄存器

寄存器名称：SMC_CYCLE[0..7]

访问类型：读/写



1．NWE_CYCLE：总的写周期长度

总的写周期长度是写周期的时钟周期中总的持续时间。它等于NWE和NCS信号设定，脉冲和保持阶段的总和。它被定义为：

写周期长度= (NWE_CYCLE[8:7]*256 + NWE_CYCLE[6:0])时钟

2．NRD_CYCLE：总的读周期长度

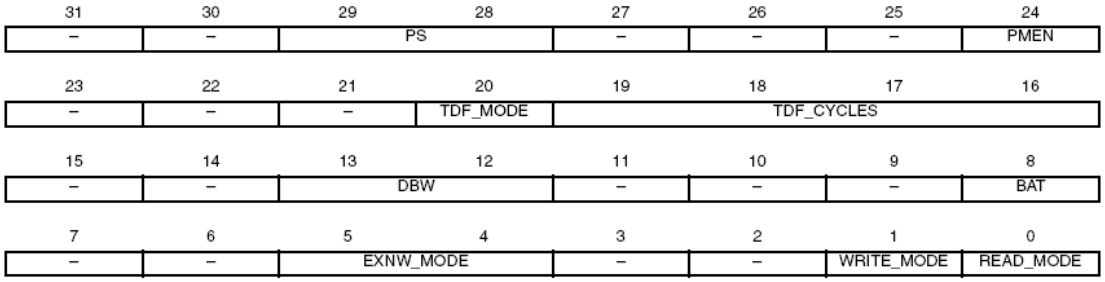
总的读周期长度是读周期的时钟周期中总的持续时间。它等于NWE和NCS信号设定，脉冲和保持阶段的总和。它被定义为：

读周期长度= (NRD_CYCLE[8:7]*256 + NRD_CYCLE[6:0])时钟周期

21.14.4　SMC MODE 寄存器

寄存器名称：SMC_MODE[0..7]

访问类型：读/写



1．READ_MODE：

1 ：读操作由 NRD 信号控制。

-如果 TDF 周期被编程，外部总线在 NRD 上升沿后被标记为忙。

-如果使能(TDF_MODE =1)TDF 最优化，在 NRD 设定后插入 TDF 等待状态。

0 ：读操作由 NCS 信号控制

-如果 TDF 周期被编程，外部总线在 NCS 上升沿后被标记为忙

-如果使能(TDF_MODE =1)TDF 最优化，TDF 等待状态在 NCS 设定后被插入

2．WRITE_MODE

1：写操作由 NWE 信号控制。

-如果使能(TDF_MODE =1)TDF 最优化，TDF 等待状态在 NWE 设定后将被插入。

0：写操作由 NCS 信号控制

-如果使能(TDF_MODE =1)TDF 最优化，TDF 等待状态在 NCS 设定后将被插入。

3．EXNW_MODE：NWAIT 模式

NWAIT 信号被用于扩展当前读或写信号。仅考虑读和写控制信号脉冲阶段。当 NWAIT 的使用使能，至少一个周期的保持时间必须为读和写控制信号编程。

EXNW_MODE		NWAIT 模式
0	0	使无效
0	1	保留
1	0	冻结模式
0	1	就绪模式

a.无效模式(disable mode)：NWAIT 输入信号在对应片选上被忽略。

b.冻结模式(frozen mode)：如果被激活，NWAIT 信号冻结当前读或写周期。重激活后，读/写周期从停止点重新开始。

c.预备模式(ready mode)：NWAIT 信号指示外部设备在控制读或写信号脉冲结束时的可用性，来完成访问。如果高电平，访问正常完成。如果低电平，访问被扩展直到 NWAIT 返回高电平。

4．BAT：字节访问类型

只要 DBW 定义一个 16 或 32 位数据总线则此域就被使用。

a.1：字节写访问类型：

-写操作用 NCS, NWR0, NWR1, NWR2, NWR3 控制。

-读操作用 NCS 和 NRD 控制

b.0：字节选择访问类型：

-写操作用 NCS, NWE, NBS0, NBS1, NBS2 and NBS3 控制

-读操作用 NCS, NRD, NBS0, NBS1, NBS2 and NBS3 控制

5．DBW：数据总线宽度

DBW		数据总线宽度
0	0	8 位总线
0	1	16 位总线
1	0	32 位总线
1	1	保留

6．TDF_CYCLES：数据浮动时间

此域给出了外部设备需要的整数的时钟周期数来在读控制信号的上升沿后释放数据。SMC 总是在 TDF_CYCLES 周期后提供一个转变得全周期。外部总线不能在 TDF_CYCLES + 1 周期被另外的片选使用。从 0 到 15 TDF_CYCLES 可被置位。

7．TDF_MODE：TDF 最优化

1：使能 TDF 最优化

-TDF 等待状态数用下一此读/写访问的设定周期优化。

0：使无效 TDF 最优化

-TDF 等待状态数在下一次访问开始前被插入

8．PMEN：页面模式使能

1：页面模式中异步脉冲串读运用在对应的片选上

0：运用标准读

9．PS：页面大小

如果页面模式被使能，此域表示页面的字节大小。

PS		页面大小
0	0	4 字节页面
0	1	8 字节页面
1	0	16 字节页面
1	1	32 字节页面

22．SDRAM 控制器（SDRAMC）

22.1 描述

SDRAM 控制器（SDRAMC）通过提供一个外部 16 位或 32 位

SDRAM 设备的接口扩展芯片的存储能力。页面大小支持从 2048 到 8192，从 256 到 2048 列。支持字节（8 位），半字（16 位）和字（32 位）访问。

SDRAM 控制器支持一个存储单元的读或写突发数据长度。控制器记录每个 bank 中的有效 row，以此方式提升 SDRAM 的性能，例如，应用程序可以放在一个 bank，数据放在另外的 bank。为了优化性能，避免在同一 bank 中访问不同的 row 是明智的。

SDRAM 控制器支持 1，2 或 3 的 CAS 延迟时间并依赖于频率优化读访问。

不同可用的模式-自刷新，掉电和深度掉电模式-最小化 SDRAM 设备上的功率损耗。

22.2 I/O 口线描述

表 22-1 I/O 口线描述

名称	描述	类型	有效电平
SDCK	SDRAM 时钟	Output	
SDCKE	SDRAM 时钟使能	Output	High
SDCS	SDRAM 控制器片选	Output	Low
BA[1:0]	Bank 选择信号	Output	
RAS	Row 信号	Output	Low
CAS	Column 信号	Output	Low
SDWE	SDRAM 写使能	Output	Low
NBS[3:0]	数据屏蔽使能信号	Output	Low
SDRAMC_A[12:0]	地址总线	Output	
D[31:0]	数据总线	I/O	

22.3 应用举例

22.3.1　软件接口

SDRAM 地址空间由 bank，row，和 column 组成。SDRAM 控制器允许依据设置在 SDRAMC 配置寄存器中的值映射不同的存储类型。

SDRAM 控制器的功能使得 SDRAM 设备访问协议对用户透明。表 22-2 到表 22-7 说明了 SDRAM 设备存储器映射和相关的设备结构被用户看见。不同的配置都有说明。

22.3.2　32 位存储器数据总线宽度

表 22-2　SDRAM 配置映射：2K 行，256/512/1024/2048 列

CPU Address Line																									
27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
					Bk[1:0]		Row[10:0]											Column[7:0]								M[1:0]	
				Bk[1:0]			Row[10:0]											Column[8:0]								M[1:0]	
			Bk[1:0]				Row[10:0]											Column[9:0]								M[1:0]	
		Bk[1:0]					Row[10:0]											Column[10:0]								M[1:0]	

微控电子 www.mcuzone.com www.ATARM.com

表 22-3　SDRAM 配置映射：4K 行，256/512/1024/2048 列

CPU Address Line																									
27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
				Bk[1:0]		Row[11:0]												Column[7:0]								M[1:0]	
			Bk[1:0]			Row[11:0]												Column[8:0]								M[1:0]	
		Bk[1:0]				Row[11:0]												Column[9:0]								M[1:0]	
	Bk[1:0]					Row[11:0]												Column[10:0]								M[1:0]	

表 22-4　SDRAM 配置映射：8K 行，256/512/1024/2048 列

CPU Address Line																									
27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
			Bk[1:0]			Row[12:0]												Column[7:0]								M[1:0]	
		Bk[1:0]				Row[12:0]												Column[8:0]								M[1:0]	
	Bk[1:0]					Row[12:0]												Column[9:0]								M[1:0]	
Bk[1:0]						Row[12:0]												Column[10:0]								M[1:0]	

注意：1。M[1:0]是字节地址里一个 32 位字

2．Bk[1] = BA1, Bk[0] = BA0.

22.3.3　16 位存储器数据总线宽度

表 22-5　SDRAM 配置映射：2K 行，256/512/1024/2048 列

CPU Address Line																									
27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
						Bk[1:0]					Row[10:0]							Column[7:0]									M0
					Bk[1:0]						Row[10:0]							Column[8:0]									M0
				Bk[1:0]							Row[10:0]							Column[9:0]									M0
			Bk[1:0]								Row[10:0]							Column[10:0]									M0

表 22-6　SDRAM 配置映射：4K 行，256/512/1024/2048 列

CPU Address Line																									
27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
					Bk[1:0]					Row[11:0]								Column[7:0]									M0
				Bk[1:0]						Row[11:0]								Column[8:0]									M0
			Bk[1:0]							Row[11:0]								Column[9:0]									M0
		Bk[1:0]								Row[11:0]								Column[10:0]									M0

表 22-7 SDRAM 配置映射：8K 行，256/512/1024/2048 列

CPU Address Line																									
27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
				Bk[1:0]						Row[12:0]								Column[7:0]									M0
			Bk[1:0]							Row[12:0]								Column[8:0]									M0
		Bk[1:0]								Row[12:0]								Column[9:0]									M0
	Bk[1:0]									Row[12:0]								Column[10:0]									M0

微控电子 www.mcuzone.com www.ATARM.com

注意：1，M0 是字节地址内的一格 16 位半字

2．Bk[1] = BA1, Bk[0] = BA0.

22.4 产品相关性

22.4.1 SDRAM 设备初始化

初始化顺序由软件产生。SDRAM 设备由以下顺序被初始化：

1. SDRAM 部件必须在配置寄存器中被配置：异步时序（TRC，TRAS，等），行数，列数，CAS 延迟时间，以及数据总线宽度。
2. 对移动 SDRAM，温度补偿自刷新（TCSR），驱动强度(PS)和局部矩阵自刷新(PASR)必须在低功耗寄存器中被置位。
3. SDRAM 存储类型必须在存储设备寄存器中被置位。
4. 在任何信号转换前，提供一个最小 200 微秒的暂停。
5. 一个空操作指令命令发给 SDRAM 设备。应用程序在模式寄存器中置位模式到 1 并执行一个到任何 SDRAM 地址写访问。
6. 一个所有 bank 预充电命令发到 SDRAM 设备。应用程序必须在模式寄存器中置位模式到 2 并执行写访问到任何 SDRAM 地址。
7. 提供八个自动刷新（CBR）周期。应用程序必须在模式寄存器中置位到 4 并执行八次任何 SDRAM 存储单元的写访问。
8. 模式寄存器（MRS）周期提供 SDRAM 设备的编程参数，特别是 CAS 延迟时间和突发数据长度。应用程序必须在模式寄存器中置位模式到 3 并执行一个 SDRAM 写访问。必须选择写地址使 BA[1:0]被设置为 0。例如，用一个 16 位 128MB SDRAM（12 行，9 列，4 个存储体）存储体地址，SDRAM 写访问应在地址 0x20000000 被执行。
9. 对移动 SDRAM 的初始化，扩展模式寄存器(EMRS)周期可用于编程 SDRAM 参数(TCSR, PASR, DS)。应用程序在模式寄存器中置位模式到 5 并执行 SDRAM 写访问。必须选择写地址使 BA[1] 或 BA[0]被设置为 1。例如，用一个 16 位 128 MB SDRAM，(12 行, 9 列, 4 存储体)存储体寻址 SDRAM 写访问应在地址 0x20800000 或 0x20400000 被执行。
10. 应用程序必须进入正常模式，在模式寄存器中设置模式为 0 并执行 SDRAM 中任何存储单元写访问。
11. 写刷新速率到 SDRAMC 刷新定时器寄存器中的计数域。

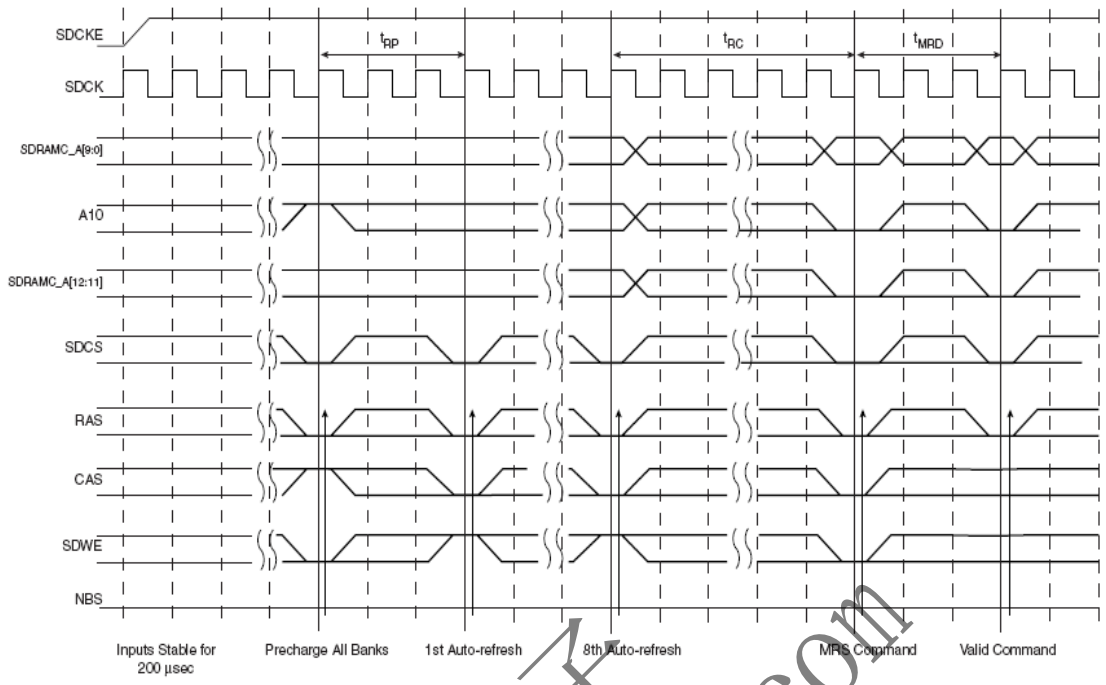
（刷新速率=刷新周期间的延迟）。SDRAM 设备需要每 15.625 微秒或 7.81 微秒刷新一次。在 100MHz 频率下，刷新定时器计数器寄存器必须用值 1562（15.652 µs x 100 MHz)或 781(7.81 µs x 100MHz)。

初始化后，SDRAM 设备将完全可用。

注意：1．强烈推荐重视初始化进程的第 5 步中描述的指令,以确保随后 SDRAMC 发出的命令可以生效。

图 22-1 SDRAM 设备初始化顺序

微控电子 www.mcuzone.com www.ATARM.com



22.4.2 I/O 口线

用于连接 SDRAM 控制器的引脚可以和 PIO 口线多路复用。编程者必须首先编程 PIO 控制器来指定 SDRAM 控制器引脚到他们的外围功能。如果 SDRAM 控制器的 I/O 口线未被应用程序使用，他们可通过 PIO 控制器被用于其它目的。

22.4.3 中断

SDRAM 控制器中断（刷新错误）连接于存储控制器。此中断可以和其它系统外围中断口线相或并最终作为系统中断源（中断源 1）提供给 AIC（先进中断控制器）。

需要首先编程 AIC 来使用 SDRAM 控制器中断。

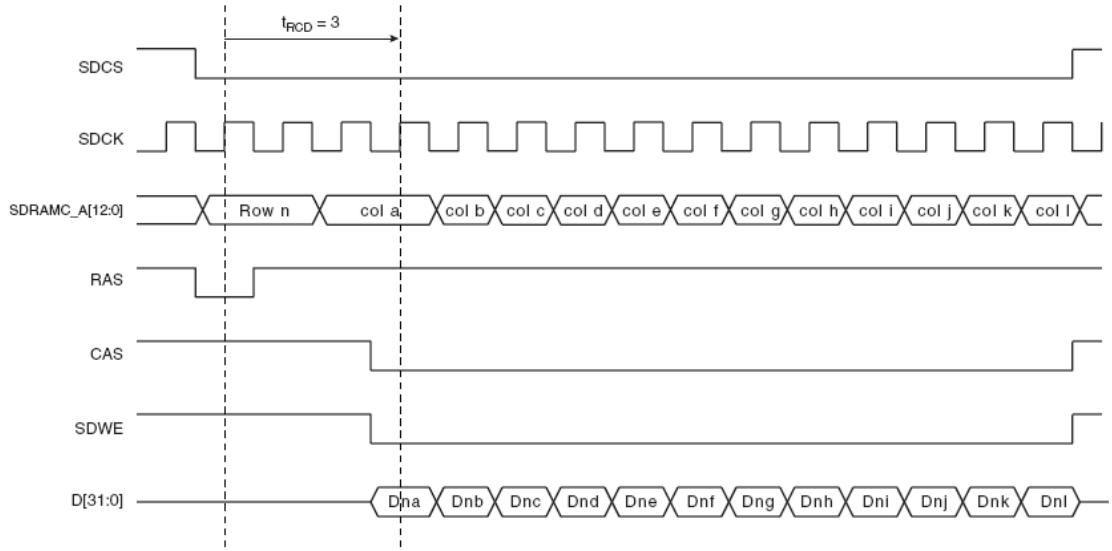
22.5 功能描述

22.5.1 SDRAM 控制器写周期

SDRAM 控制器允许突发数据访问或单独访问。两种情况中，SDRAM 控制器记录每个 bank 中的有效 row，以此方式提升性能。为初始化突发数据访问，SDRAM 控制器由主控请求访问提供的传送类型信号。如果下一次访问是顺序写访问，那么 SDRAM 的写操作将被执行。但当前访问是一个边界页面，或如果下一次访问在另外的行，接着 SDRAM 产生一个预充电命令，激活新行并初始化写命令。为满足 SDRAM 时序参数，在预充电/激活(t_{RP})命令和激活/写(t_{RCD})命令之间插入附加的时钟周期。对于这些时序参数的定义，参考 223 页“SDRAM 配置寄存器”。如图 22-2 所示。

图 22-2 写突发数据，32 位 SDRAM 访问

微控电子 www.mcuzone.com www.ATARM.com



22.5.2 SDRAM 控制器读周期

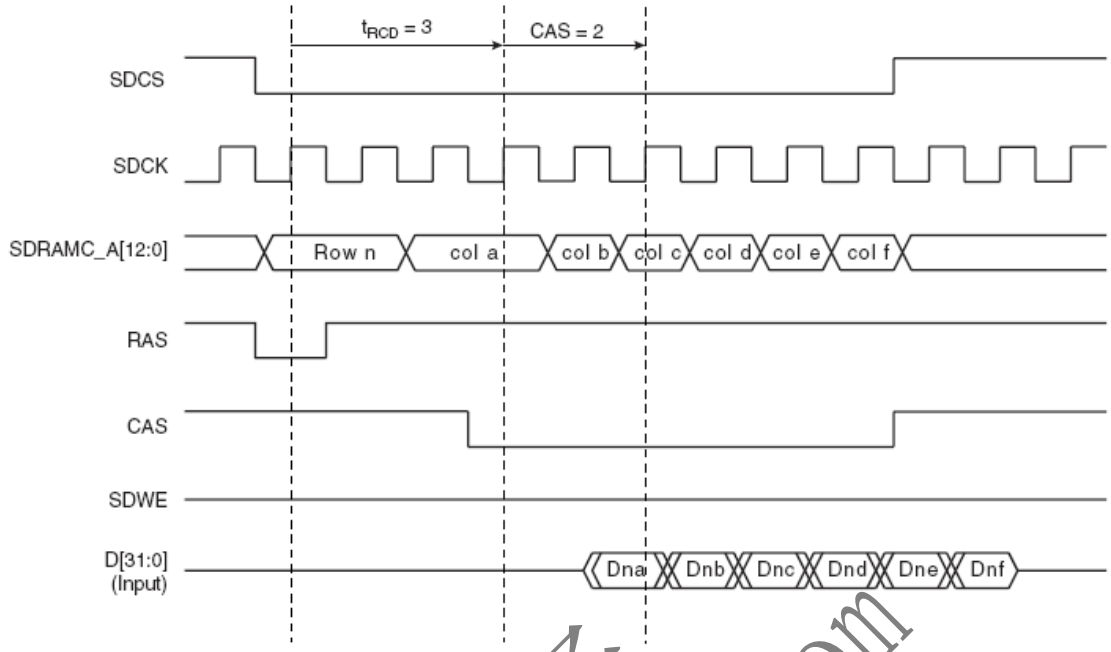
SDRAM 控制器允许突发数据访问，非特定长度增量突发访问或单个访问。所有情况中，SDRAM 控制器记录每个 bank 中的活动的 row，以此来最大化 SDRAM 的性能。如果行和存储体地址和先前的行/存储体地址不匹配，接着 SDRAM 控制器自动地产生一个预充电命令，激活新行和启动读命令。为满足 SDRAM 时序参数，在预充电和激活命令(tRP)间与激活和读命令(tRCD)间插入 SDCK 上的附加的时钟周期。这两个参数被设置在 SDRAM 控制器的配置寄存器中。一个读命令后，产生附加的等待状态服从于 CAS 延迟时间（在配置寄存器中特定的 1，2 或 3 时钟延迟）。

对一个单独访问或一个非特定长度的增量突发访问，SDRAM 控制器预先处理下一次访问。当列的最后一个值被 SDRAM 控制器返回到总线上，SDRAM 控制器预先处理读下一列并因此预先处理 CAS 延迟。这就减小了内部总线上 CAS 延迟。

对特定长度的（4，8，16 字）突发数据访问，无预处理访问。这种情况导致最好的性能。如果突发数据被破坏（边界，繁忙模式，等），下一次访问被处理为非特定长度的增量突发访问。

图 22-3 读突发数据，32 位 SDRAM 访问

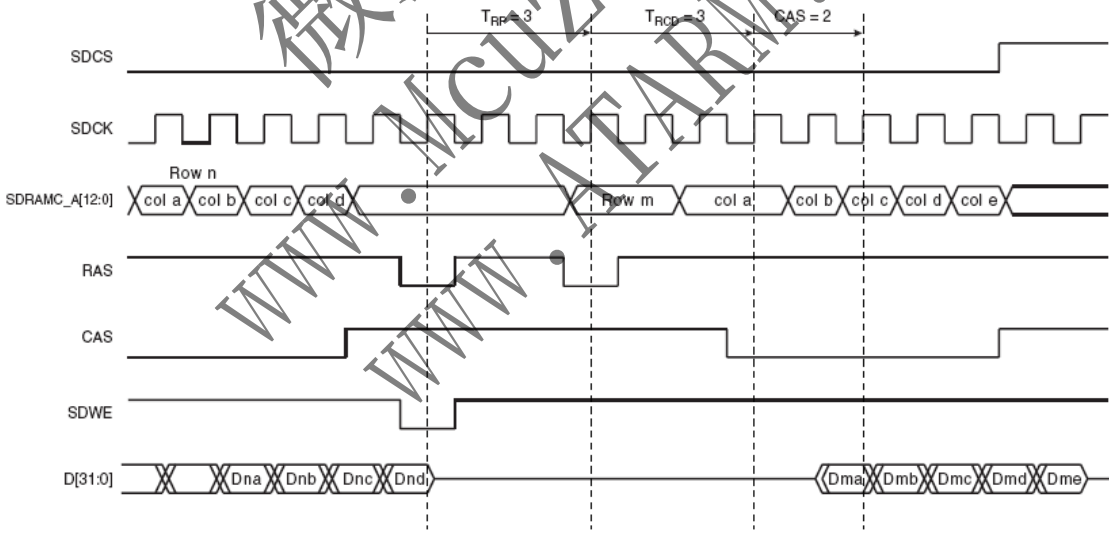
微控电子 MCUZONE.COM WWW.ATARM.COM



22.5.3 边界管理

当存储器行边界到达，将插入一个自动页面中断。在此情况下，SDRAM 控制器产生一个预充电命令，激活新行并初始化读或写命令。为满足 SDRAM 时序参数，在预充电和激活命令(tRP)间与激活和读命令(tRCD)间插入 SDCK 上的附加的时钟周期。如下图 22-4 描述。

图 22-4 边界行访问的数据突发读



22.5.4 SDRAM 控制器刷新周期

自动刷新命令被用于刷新 SDRAM 设备。刷新地址由 SDRAM 设备内部产生并在每次自动刷新后增加。SDRAM 控制器周期性的产生这些自动刷新命令。内部定时器用指示刷新周期间时钟周期数的 SDRAMC_TR 寄存器中的值装载。

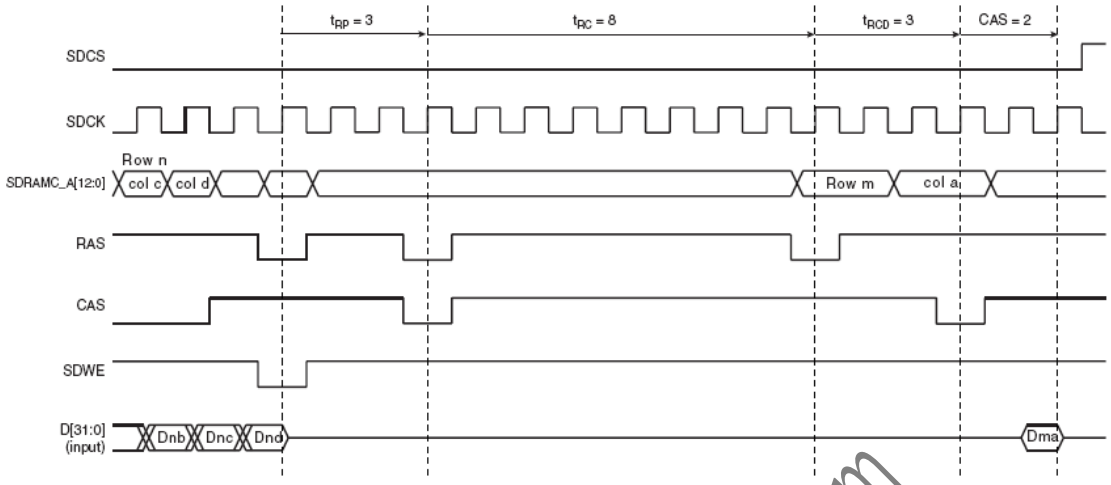
当前一个自动刷新命令未执行时产生一个刷新错误中断。通过读中断状态寄存器(SDRAMC_ISR)获取。

当 SDRAM 控制器初始化 SDRAM 设备的一个刷新，内部存储器访问不被延

微控电子 www.mcuzone.com www.ATARM.com

迟。然而，如果 CPU 试图访问 SDRAM，从控指示设备繁忙，主控由一个等待信号保持。见图 22-5。

图 22-5　刷新周期紧跟一个读访问



22.5.5　电源管理

三个可用的低功率模式：

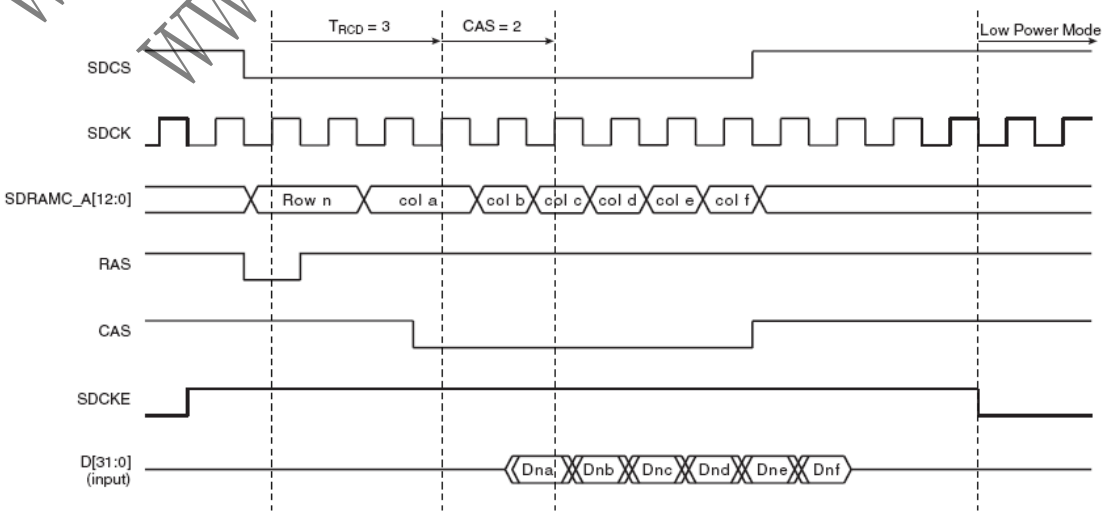
1. 自刷新模式：SDRAM 在没有 SDRAM 控制器控制下执行其自动刷新周期。被 SDRAM 消耗的电流很低。

2. 掉电模式：自动刷新周期由 SDRAM 控制器控制。在自动刷新周期间，SDRAM 处于掉电状态。掉电模式中消耗的电流比在自刷新模式中的高。

3. 深度掉电模式：（仅在移动 SDRAM 可用）SDRAM 内容丢失，但 SDRAM 不消耗任何电流。

只要 SDRAM 设备未被选择，SDRAM 控制器就激活一个低功率模式。最近一次访问后通过编程低功率寄存器中的一个超时值，可以延迟进入自刷新和掉电模式。

22.5.6　自刷新模式



该模式通过向 SDRAMC 的低功耗寄存器中的 LPCB 域写入 1 来选择。自刷

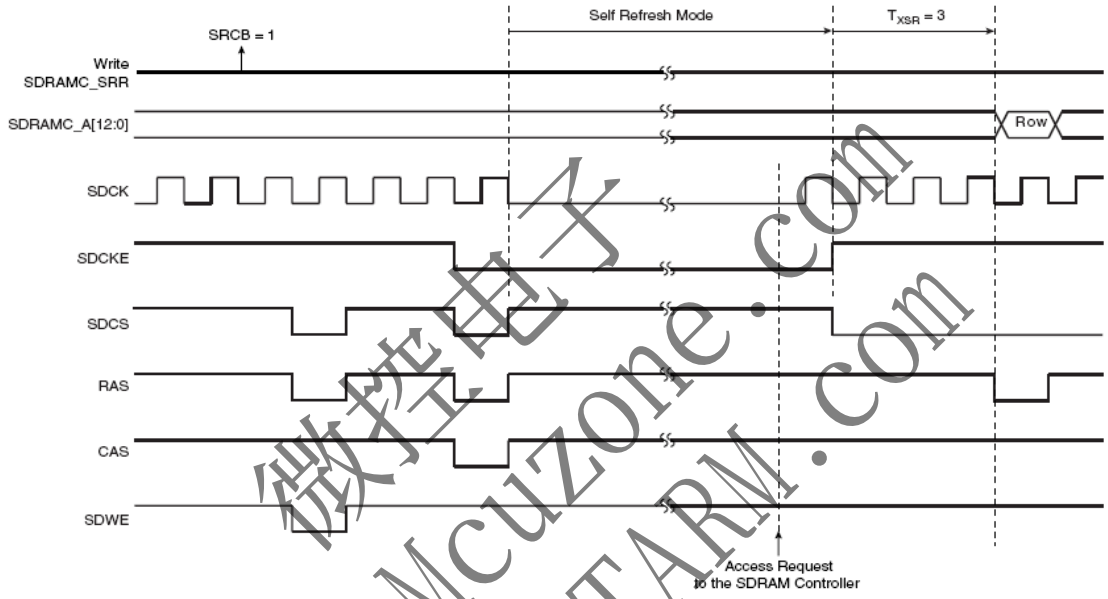
微控电子 www.mcuzone.com www.ATARM.com

新模式中，SDRAM 设备不用外部时钟保留数据而提供其自身的内部时钟，因此执行其自身的自动刷新。所有到 SDRAM 设备的信号除了必须保持低电平的 SDCKE，其它的变得无关紧要。只要 SDRAM 设备被选择，SDRAM 控制器提供一系列命令使其退出自刷新模式。

一些低功率 SDRAM（例如，移动 SDRAM）仅能刷新四分之一或八分之一或 SDRAM 行列的所有的存储体。此特性减小了自刷新电流。为配置此特性，温度补偿自刷新（TCSR），局部行列自刷新(PASR)和驱动强度（DS）参数必须在低功率寄存器被设置并在初始化期间被传输到低功率 SDRAM。

SDRAM 设备必须维持在自刷新模式至少 t_{RAS}。可以保持在自刷新模式并持续不确定的时间。如图 22-6 中描述。

图 22-6　自刷新模式过程



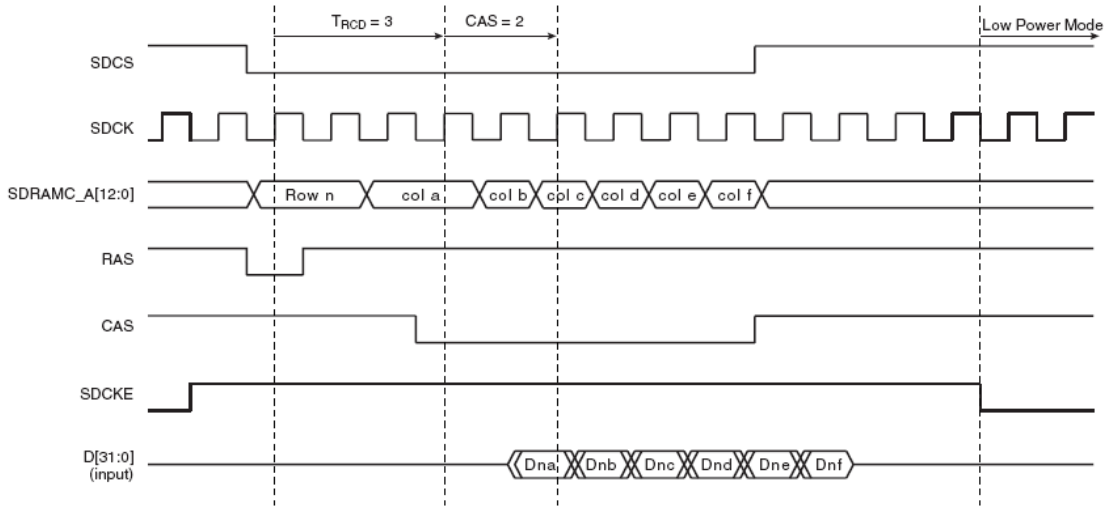
22.5.7　低功率模式

在 SDRAM 低功率寄存器通过编程 LPCB 域为 2 来选择此模式。功耗比在自刷新模式大。SDRAM 的所有输入和输出缓冲器除保持低电平的 SDCKE 外均被禁用。与自刷新模式不同，SDRAM 设备保持在低功率模式的时间不能比刷新周期（对整个设备刷新操作的 64ms）长。因为无自动刷新操作被 SDRAM 自身执行，SDRAM 控制器实现刷新操作。退出过程比自刷新模式快。

如图 22-7 中描述。

图 22-7　低功率模式过程

微控电子 www.mcuzone.com www.ATARM.com



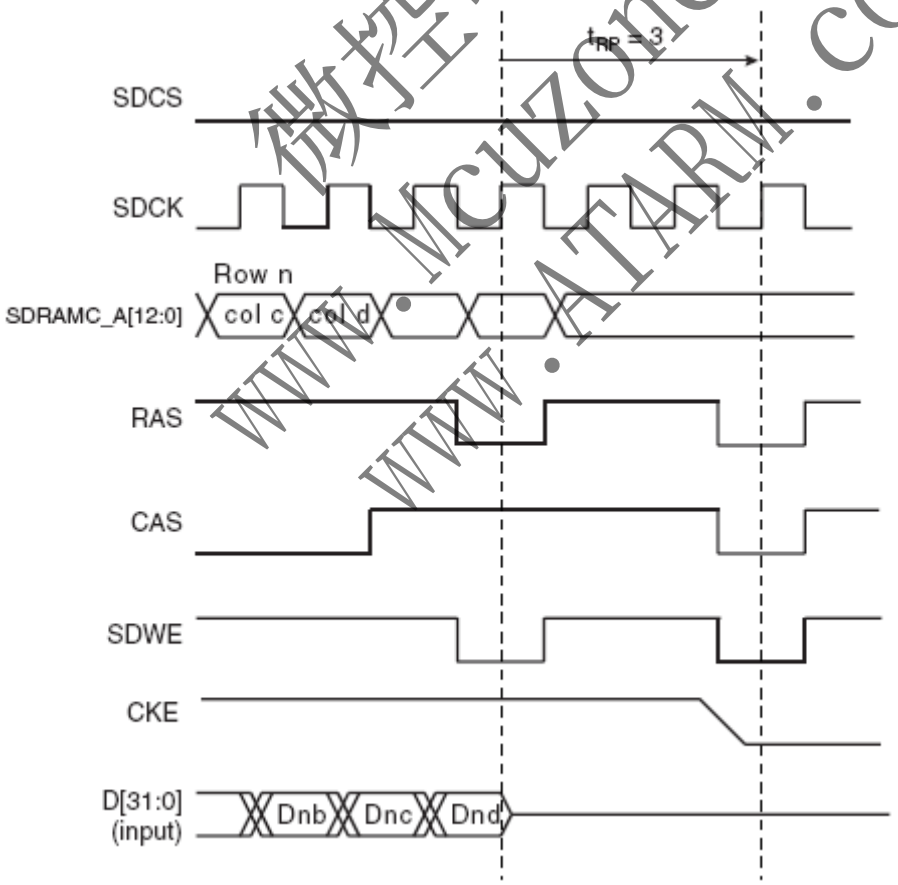
22.5.8 深度掉电模式

在 SDRAM 低功率寄存器通过编程 LPCB 域为 3 来选择此模式。当此模式被激活，所有 SDRAM 的内部电压发生器被停止并且所有数据丢失。

当此模式被使能，应用程序不能访问 SDRAM，直到新的初始化顺序完成。（见 210 页“SDRAM 设备初始化”）。

如图 22-8 描述

图 22-8 深度掉电模式过程



22.6 SDRAM 控制器用户接口

微控电子 www.mcuzone.com www.ATARM.com

表 22-8　SDRAM 控制器存储器映射

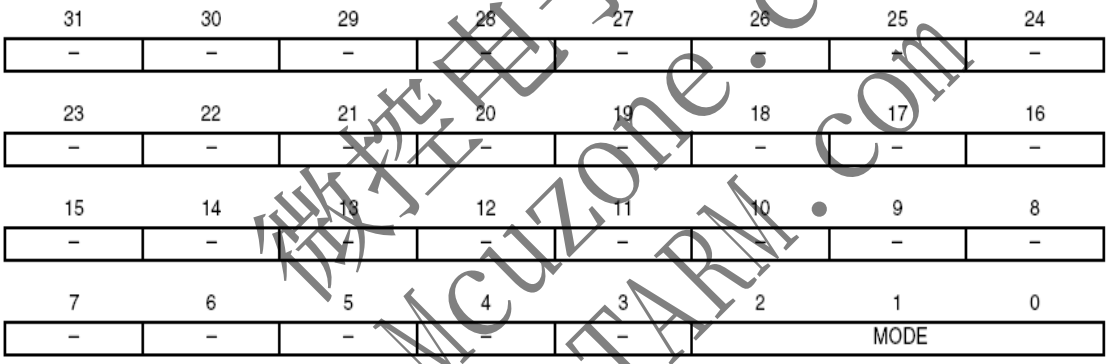
偏移量	寄存器	名称	访问	复位状态
0x00	SDRAMC 模式寄存器	SDRAMC_MR	读/写	0x00000000
0x00	SDRAMC 刷新定时器寄存器	SDRAMC_TR	读/写	0x00000000
0x04	SDRAMC 配置寄存器	SDRAMC_CR	读/写	0x852372C0
0x08	SDRAMC 低功率寄存器	SDRAMC_LPR	读/写	0x0
0x10	SDRAMC 中断使能寄存器	SDRAMC_IER	只写	-
0x14	SDRAMC 中断使无效寄存器	SDRAMC_IDR	只写	-
0x18	SDRAMC 中断屏蔽寄存器	SDRAMC_IMR	只读	0x0
0x1C	SDRAMC 中断状态寄存器	SDRAMC_ISR	只读	0x0
0x20	SDRAMC 存储设备寄存器	SDRAMC_MDR	读	0x0
0x24	保留	-	-	-

22.6.1　SDRAM 模式寄存器

寄存器名称：SDRAMC_MR

访问类型：读/写

复位值：0x00000000



MODE：SDRAM 命令模式

此域定义了当 SDRAM 设备被访问时由 SDRAM 控制器发出的命令。

MODE			描述
0	0	0	正常模式。任何访问 SDRAM 被正常解码
0	0	1	SDRAM 控制器在访问 SDRAM 设备时发出一个空操作命令而不管周期。
0	1	0	SDRAM 控制器在访问 SDRAM 设备时发出一个“所有存储体预充电”命令而不管周期。
0	1	1	SDRAM 控制器在访问 SDRAM 设备时发出一个“装载模式寄存器”命令而不管周期。关于 SDRAM 设备基本地址的地址偏移量被用于编程模式寄存器。例如，当此模式被激活，访问“SDRAM_Base +偏移量”地址产生一个“装载模式寄存器”命令，将会将“偏移量”值写入 SDRAM 设备模式寄存器
1	0	0	SDRAM 控制器在访问 SDRAM 设备时发出一个“自动刷新”命令而不管周期。在此之前，一个“所有存储体预充电”命令必须被发出。
1	0	1	SDRAM 控制器在 SDRAM 设备被访问时发出一个扩展的装载模式寄存器命令而不管周期。相对于 SDRAM 设备基地址的地址偏移量被用于编程模式寄存器。例如，当此模式被激活，访问“SDRAM_Base +偏移量”地址产生“扩展的装载模式寄存器”命令，将写偏移量值到 SDRAM 设备模式寄存器。
1	1	0	深度掉电模式。进入深度掉电模式。

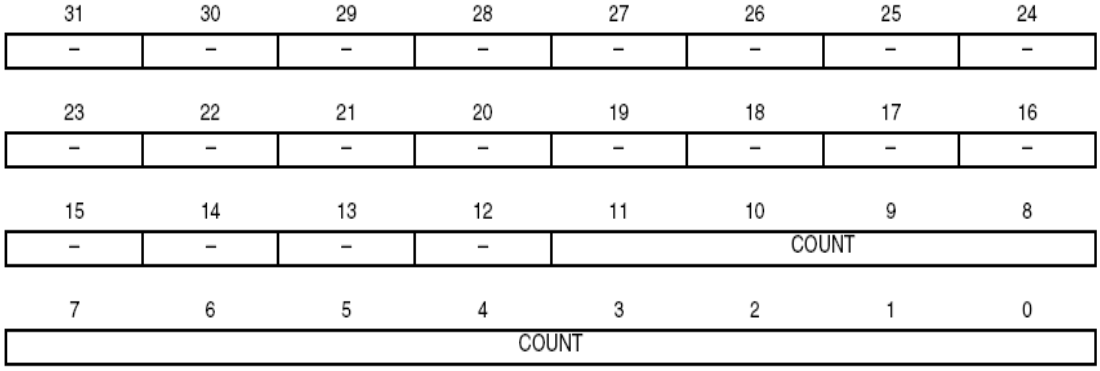
微控电子 www.mcuzone.com www.ATARM.com

22.6.2 SDRAMC 刷新定时器寄存器

寄存器名称：SDRAMC_TR

访问类型：读/写

复位值：0x00000000



COUNT：SDRAMC 刷新定时器寄存器

此 12 位域被装载入一个产生刷新脉冲的定时器。每次产生刷新脉冲，初始化一次刷新突发数据。被装载的值依赖于 SDRAM 时钟频率（MCK：主控时钟），SDRAM 设备的刷新速率和 15.6 μs 每行的刷新突发数据长度对一个突发数据长度是一个典型的值。

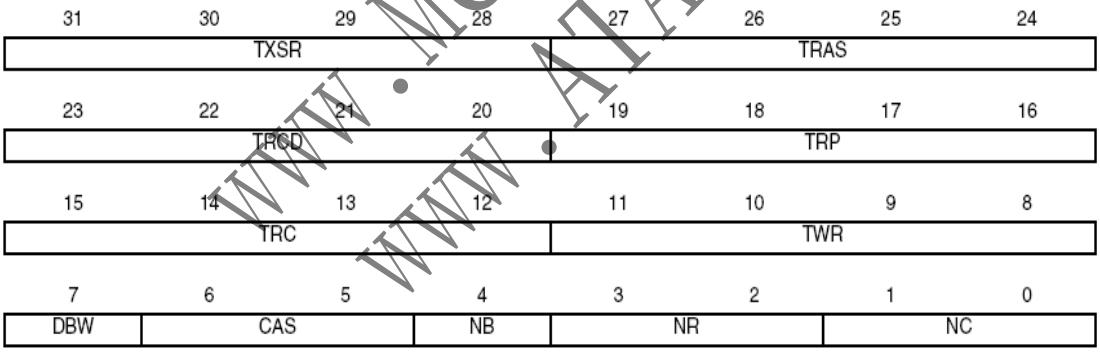
为了刷新 SDRAM 设备，必须写此 12 位域。如果未满足此条件，将不会发出刷新命令并且不会执行 SDRAM 设备的刷新。

22.6.3 SDRAM 配置寄存器

寄存器名称：SDRAMC_CR

访问类型：读/写

复位值：0x852372C0



NC：列位数

复位值是 8 列

NC		列
0	0	8
0	1	9
1	0	10
1	1	11

NR：行位数

复位值是 11 行

微控电子 www.Mcuzone.com www.ATARM.com

NR		行
0	0	11
0	1	12
1	0	13
1	1	保留

NB：存储体数

复位值是两个存储体

NB	存储体数
0	2
1	4

CAS：CAS 延迟时间

复位值是两个周期

SDRAMC 中，仅管理一，二和三个周期的 CAS 延迟时间。任何情况下，必须编程另外的值。

CAS		CAS 延迟时间（周期）
0	0	保留
0	1	1
1	0	2
1	1	3

- DBW：数据总线宽度

复位值是 16 位

0：数据总线宽度是 32 位

1：数据总线宽度是 16 位

- TWR：写恢复延迟

复位值是两个周期

此域定义写恢复时间的周期数。周期数在 0 到 15 之间。

- TRC：行周期延迟

复位值是七个周期。

此域定义刷新和激活命令间延迟的周期数。周期数在 0 到 15 之间。

- TRP：行预充电延迟

复位值是三周期

此域定义了一个预充电命令和另外的命令之间延迟的周期数。周期数在 0 到 15 之间。

- TRCD：行到列延迟

复位值是两个周期

此域定义一个激活命令和一个读/写命令之间延迟的周期数。周期数在 0 到 15 之间。

- TRAS：激活到预充电延迟

复位值是五个周期。

此域定义一个激活命令和一个预充电命令之间延迟的周期数。周期数在 0 到 15 之间。

- TXSR：退出自刷新到激活延迟

复位值是八个周期

此域定义一个 SCKE 置位高电平和一个激活命令之间延迟的周期数。周期数在 0 到 15 之间。

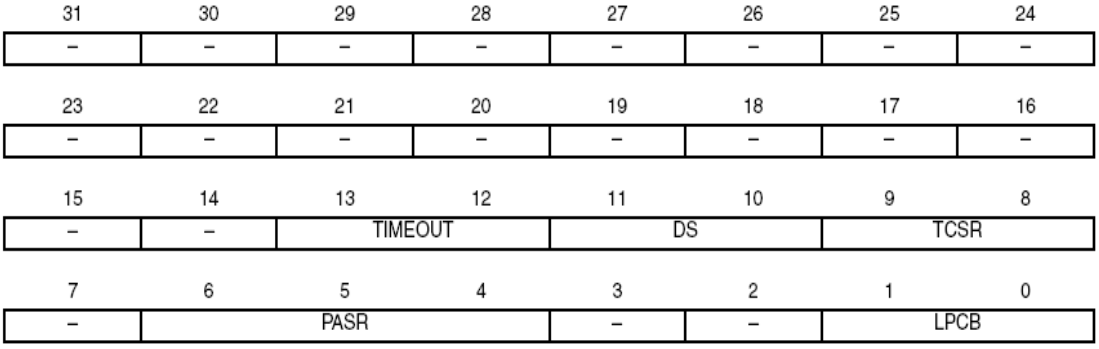
22.6.4 SDRAM 低功率寄存器

寄存器名称：SDRAMC_LPR

微控电子 www.mcuzone.com www.ATARM.com

访问类型：读/写

复位值：0x0



- LPCB：低功率配置位

00	低功率特性被禁止：无掉电，自刷新或深度掉电命令发给 SDRAM 设备
01	SDRAM 控制器发一个自刷新命令给 SDRAM 设备，SDCLK 时钟被禁用并且 SDCKE 信号被设置为低电平。当访问 SDRAM 设备退出自刷新模式并在访问后进入自刷新模式。
10	SDRAM 控制器在每次访问后发掉电命令给 SDRAM 设备，SDCKE 信号被设置为低电平。SDRAM 设备当访问时退出掉电模式并在访问后进入掉电模式。
11	SDRAM 控制器发一个深度掉电命令给 SDRAM 设备。此模式对于低功率 SDRAM 是唯一的。

- PASR：局部行列自刷新（仅对低功率 SDRAM）

  PASR 参数在初始化期间传送到 SDRAM 来规定是否仅 SDRAM 阵列的四分之一，二分之一或所有存储体被使能。使无效的存储体在自刷新模式不会刷新。此参数必须根据 SDRAM 设备说明被设置。

- TCSR：温度补偿自刷新（仅对低功率 SDRAM）

TCSR 参数在初始化期间传输给 SDRAM，根据低功耗 SDRAM 的温度来设置自刷新模式下刷新间隔。此参数必须根据 SDRAM 设备说明被设置。

- DS：驱动强度（仅对低功率 SDRAM）

DS 参数在初始化期间传输给 SDRAM 来选择 SDRAM 数据输出强度。此参数必须根据 SDRAM 设备说明被设置。

- TIMEOUT：用于确定何时进入低功耗模式

00	SDRAM 在最后一次传输结束后立即激活 SDRAM 低功率模式
01	SDRAM 在最后一次传输结束 64 个时钟周期后立即激活 SDRAM 低功率模式
10	SDRAM 在最后一次传输结束 128 个时钟周期后立即激活 SDRAM 低功率模式
11	保留

22.6.5 SDRAM 中断使能寄存器

寄存器名称：SDRAMC_IER

访问类型：只写

微控电子 www.mcuzone.com www.ATARM.com

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

- RES：刷新错误状态

0：无效

1：使能刷新错误中断

22.6.6 SDRAM 中断使无效寄存器

寄存器名称：SDRAMC_IDR

访问类型：只写

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

- RES：刷新错误状态

0：无效

1：使无效刷新错误中断

22.6.7 SDRAMC 中断屏蔽寄存器

寄存器名称：SDRAMC_IMR

访问类型：只读

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

- RES：刷新错误状态

0：刷新错误中断被使无效

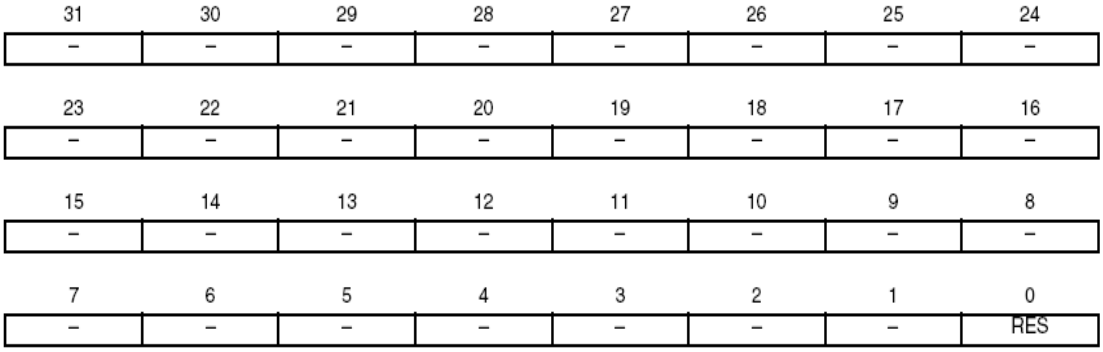
1：刷新错误中断被使能

微控电子 www.Mcuzone.com www.ATARM.com

22.6.8 SDRAMC 中断状态寄存器

寄存器名称：SDRAMC_ISR

访问类型：只读



- RES： 刷新错误状态

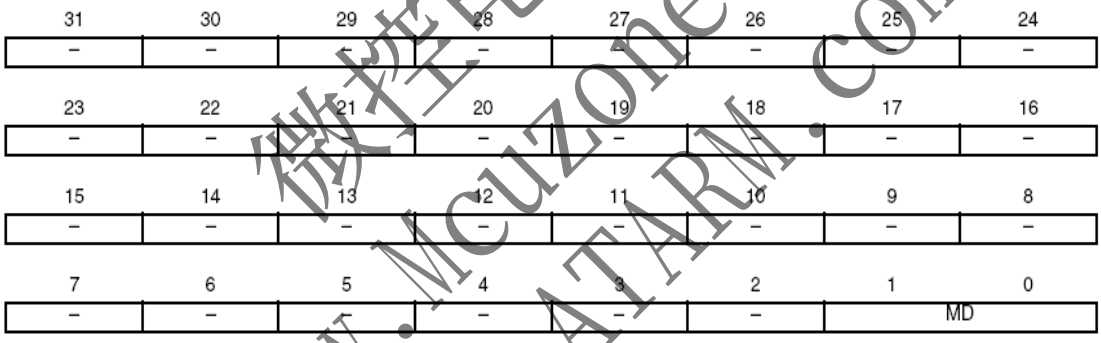
0：从最后一次读到当前无刷新错误被检测到

1：从最后一次读到当前一个刷新错误被检测到

22.6.9 SDRAMC 存储设备寄存器

寄存器名称：SDRAMC_MDR

访问类型：读/写



- MD：存储设备类型

00	SDRAM
01	低功率 SDRAM
10	保留
11	保留

微控电子 www.mcuzone.com www.ATARM.com

23. 外设 DMA 控制器（PDC）

23.1 描述

外设 DMA 控制器（PDC）在片上串行外设和片上或片外存储器之间传送数据。PDC 和一个串行外设间的连接被 AHB 到 ABP 的桥操作。

PDC 包含九个通道。全双工外设包含十八个成对（仅传送或仅接收）使用的单向通道。半双工外设使用一个双向通道。

每个 PDC 通道的用户接口集成进了其服务外设的用户接口。单向通道（仅接收或仅发送）的用户接口，包含两个 32 位存储器指针和两个 16 位计数器，一组用于当前传送（指针，计数器），另一组用于下一次传送。双向通道用户接口包含四个 32 位存储器指针和四个 16 位计数器，每组（指针，计数器）被用于当前传送，下一次传送，当前接收和下一次接收。

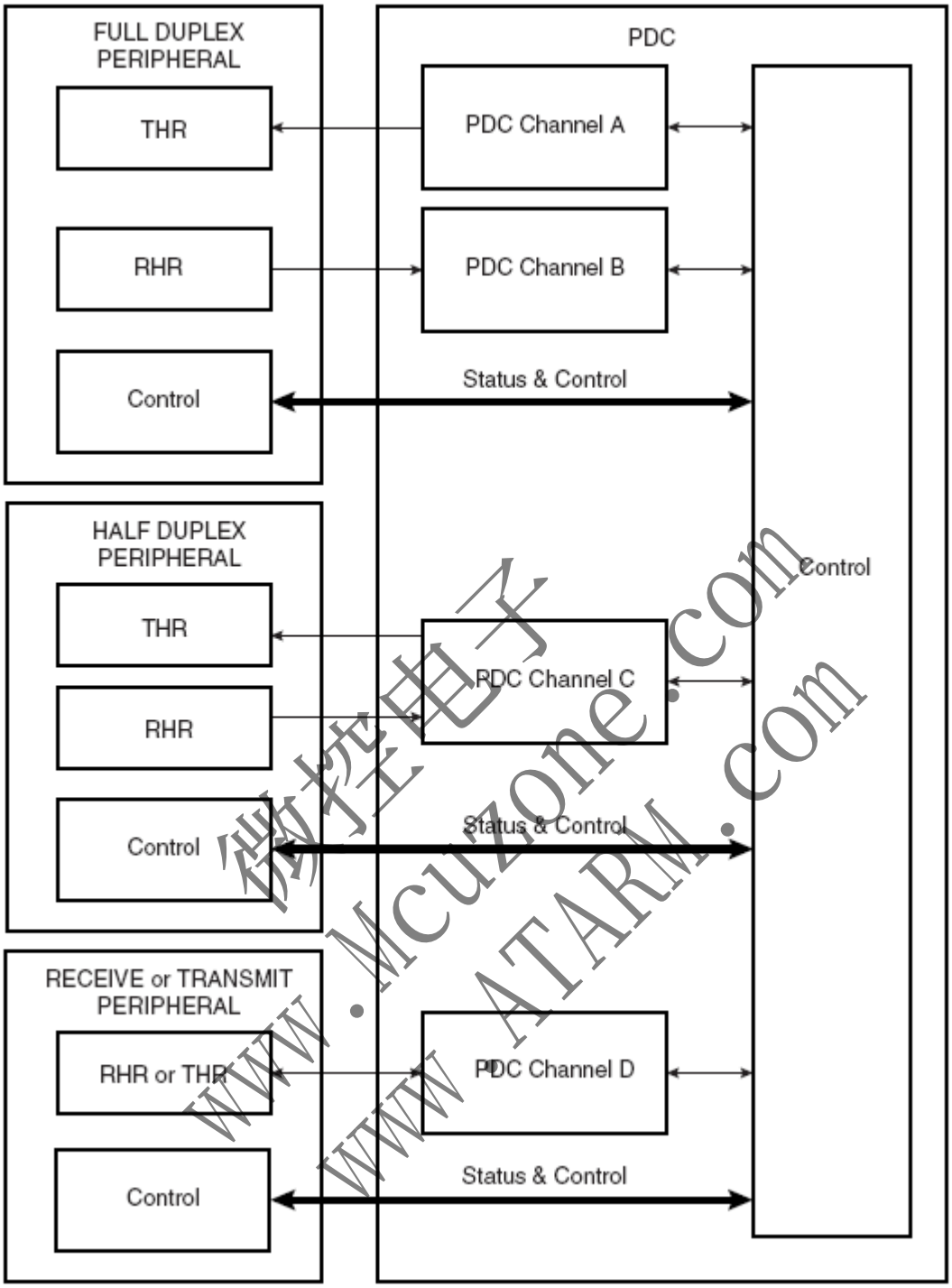
通过使用 PDC 进行传输，可以避免处理器的参与，从而减少处理器的负荷。显著减小了一个数据传送所需的周期数，从而提高了微控制器的性能。

为启动一个传送，外设通过使用发送和接收信号触发其关联的 PDC 通道。当编程的数据被传送完成，外设自身产生传送完成的中断。

23.2 方块图

图 23-1 方块图

微控电子 www.Mcuzone.com www.ATARM.com



微控电子 www.mcuzone.com www.ATARM.com

23.3 功能描述

23.3.1 配置

PDC 通道用户接口使用户能对每个通道进行配置并控制数据传送。每个 PDC 通道的用户接口都集成进了关联的外设用户接口。

串行外设的用户接口，无论是全双工或半双工，包含四个 32 位指针(RPR, RNPR, TPR, TNPR)和四个 16 位计数器寄存器(RCR, RNCR, TCR, TNCR)。然而，每种类型的发送和接收部分被不同的编程：全双工外设的发送和接收部分可在同一时间被编程，而半双工外设仅一部分（发送或接收）可在某时被编程。

32 位指针为当前和下一次传送在存储器中定义访问位置，无论其读（发送）或写（接收）访问。16 位计数器定义当前和下一次传送的数据长度。在任何时刻，读取每个通道的剩余传送数是可能的。

PDC 有专用的指示每个通道传送是否被使能的状态寄存器。每个通道的状态被设置在关联的外设状态寄存器。传送器可通过置位外设的传送控制寄存器中的 TXTEN/TXTDIS 和 RXTEN/RXTDIS 使能和/或使无效。

在传送完成时，PDC 通道发送状态标志位到其关联的外设。这些标志位在外设状态寄存器(ENDRX, ENDTX, RXBUFF,和 TXBUFE)中是可见的。参考 23.3.3 段和关联的外设用户接口。

23.3.2 存储器指针

每个全双工外设被一个接收通道和一个发送通道连接到 PDC。两个通道都有 32 位存储器指针，指针分别指向片上或片外存储器中的一个接收区和一个发送区。

每个半双工外设被一个双向通道连接到 PDC。此通道有两个 32 位存储器指针，一个用于当前传送，另一个用于下一次传送。这些指针指向发送数据还是接收数据依赖于外设的操作模式。

依赖于传送的类型（字节，半字或子），存储器指针分别增长 1，2 或 4 字节。

如果一个存储器指针地址在传送中间改变了，PDC 通道继续用新地址操作。

23.3.3 传送计数器

每个通道有两个 16 位计数器，一个对当前传送计数，另一个用于下一次传送。这两个计数器定义由通道传送的数据容量。当前传送计数器随着由当前存储器指针寻址的数据开始被传送而首先减小。当当前传送计数器到达 0 时，通道检查它的下一个计数器。如果下一个计数器的值是 0，通道停止传送数据并置位正确的标志位。如果下一个计数器的值比 0 大，下一个指针/下一个计数器被复制到当前指针/当前计数器并且通道重新开始传送而下一个指针/下一个计数器变为 0 值。在此次传送的结尾，PDC 通道在外设状态寄存器中置位正确的标志位。

下面给出了状态寄存器标志位依赖于计数器值的如何动作的概述：

1. 当 PERIPH_RCR 寄存器到达 0 时置位 ENDRX 标志位
2. 当 PERIPH_RCR 和 PERIPH_RNCR 都到达 0 时置位 RXBUFF 标志位
3. 当 PERIPH_RCR 寄存器到达 0 时置位 ENDTX 标志位
4. 当 PERIPH_RCR 和 PERIPH_RNCR 都到达 0 时置位 TXBUFE 标志位

这些状态标志位在外设状态寄存器中有描述。

23.3.4 数据传送

串行外设用传送控制寄存器中的发送使能和接收使能标志位触发其关联的 PDC 通道的传输，此串行外设集成于外设的用户接口。

当外设接收一个外部数据，将发送一个接收就绪信号给其对应的 PDC 接收通道，后者将请求访问矩阵。当访问被授权，PDC 接收通道开始读取外设接收保持寄存器（RHR）。读数据被存储于一个内部缓冲器并且接着被写入存储器。

当外设要发送数据，先发送一个发送就绪信号给对应的 PDC 发送通道，后者将请求访问矩阵。当访问被授权，PDC 发送通道从存储器读取数据并把它们

放入其关联外设的发送保持寄存器（THR）。相同的外设根据其途径发送数据。

23.3.5 PDC 标志位和外设状态寄存器

连接于 PDC 的每个外设发送接收就绪和发送就绪标志位， PDC 返回标志位给外设。所有这些标志位仅在外设状态寄存器中可见。

依赖于外设的类型，半双工或全双工，标志位属于一个单通道或两个不同的通道。

23.3.5.1 接收传送结束

当 PERIPH_RCR 寄存器到达 0 时此标志位被置位并且最后一次数据被传送给存储器。

通过在 PERIPH_TCR 或 PERIPH_TNCR 中写一个非零值来复位。

23.3.5.2 发送传送结束

当 PERIPH_TCR 寄存器到达 0 时此标志位被置位并且最后一次数据被写入外设 THR。

通过在 PERIPH_TCR 或 PERIPH_TNCR 中写一个非零值来复位。

23.3.5.3 接收缓冲器满

当 PERIPH_RCR 寄存器到达 0， PERIPH_RNCR 也被设置为 0 时此标志位被置位，最后一次数据被传送给存储器。

通过在 PERIPH_TCR 或 PERIPH_TNCR 中写一个非零值来复位。

23.3.5.4 发送缓冲器空

当 PERIPH_TCR 寄存器到达 0， PERIPH_TNCR 也被置位 0 时此标志位被置位，最后一次数据被写入外设 THR。

通过在 PERIPH_TCR 或 PERIPH_TNCR 中写一个非零值来复位。

23.4 外设 DMA 控制器（PDC）用户接口

表 23-1 存储器映射

偏移量	寄存器	名称	访问	复位值
0x100	接收指针寄存器	PERIPH(1)_RPR	读/写	0
0x104	接收计数器寄存器	PERIPH_RCR	读/写	0
0x108	发送指针寄存器	PERIPH_TCR	读/写	0
0x10C	发送计数器寄存器	PERIPH_TCR	读/写	0
0x110	接收下一个指针寄存器	PERIPH_RNPR	读/写	0
0x114	接收下一个计数器寄存器	PERIPH_RNPR	读/写	0
0x118	发送下一个指针寄存器	PERIPH_TNPR	读/写	0
0x11C	发送下一个计数器寄存器	PERIPH_TNPR	读/写	0
0x120	传送控制寄存器	PERIPH_PTCR	写	0
0x124	传送状态寄存器	PERIPH_PTCR	读	0

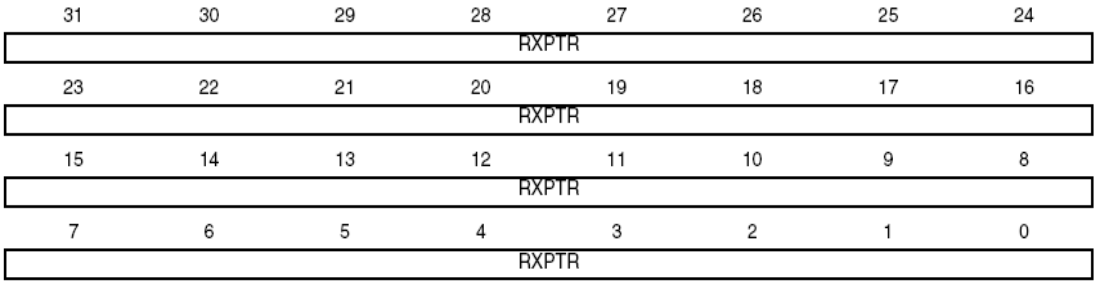
注意：1。PERIPH：十个寄存器以同一偏移量被映射到外设存储器空间。这些可以被用户根据功能和外设需要(DBGU, USART, SSC, SPI, MCI, 等)定义。

23.4.1 接收指针寄存器

寄存器名称：PERIPH_RPR

访问类型：读/写

微控电子 www.mcuzone.com www.ATARM.com



- RXPTR：接收指针寄存器

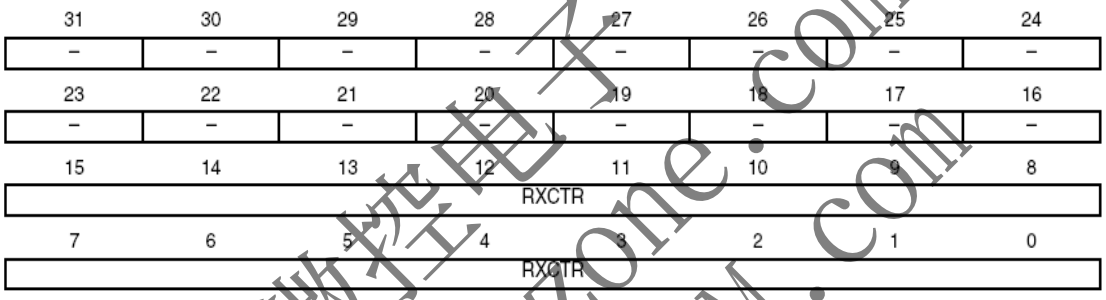
RXPTR 必须设置到接收缓冲地址。

当一个半双工外设被连接于 PDC, RXPTR = TXPTR。

23.4.2 接收计数器寄存器

寄存器名称：PERIPH_RCR

访问类型：读/写



- RXCTR：接收计数器寄存器

RXCTR 必须被设置为接收缓冲器大小。

当一个半双工外设被连接于 PDC, RXCTR = TXCTR。

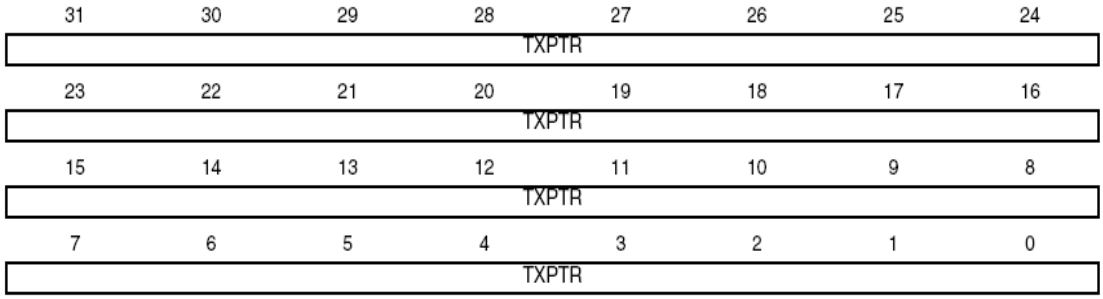
0=停止给接收器的数据传送

1-65535=如果对应通道有效，开始外设数据传送

23.4.3 发送指针寄存器

寄存器名称：PERIPH_TPR

访问类型：读/写



- TXPTR：发送计数器寄存器

TXPTR 必须被设置到发送缓冲器地址。

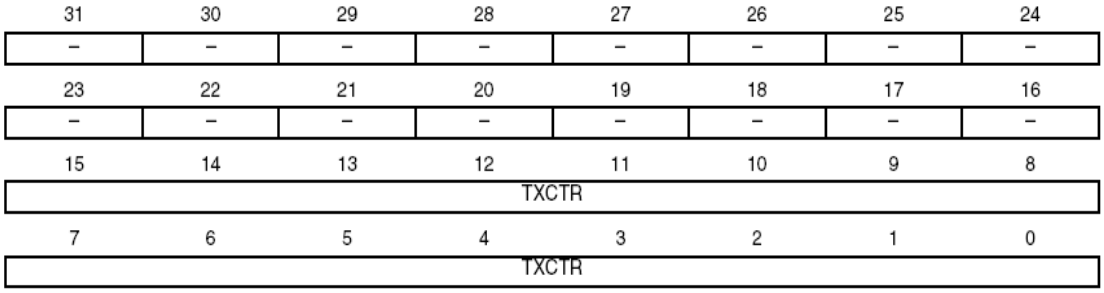
当一个半双工外设被连接到 PDC, RXPTR = TXPTR。

微控电子 www.mcuzone.com www.ATARM.com

23.4.4 发送计数器寄存器

寄存器名称：PERIPH_TCR

访问类型：读/写



- TXCTR：发送计数器寄存器

TXCTR 必须被设置为发送缓冲器大小

当一个半双工外设被连接于 PDC, RXCTR = TXCTR。

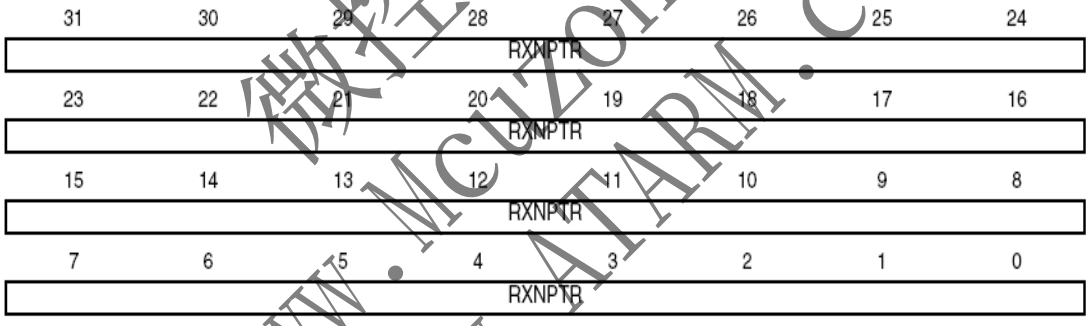
0=停止外设给发送器的数据传送

1-65535=如果对应通道有效，开始外设数据传送

23.4.5 接收下一个指针寄存器

寄存器名称：PERIPH_RNPR

访问类型：读/写



- RXNPTR：接收下一个指针

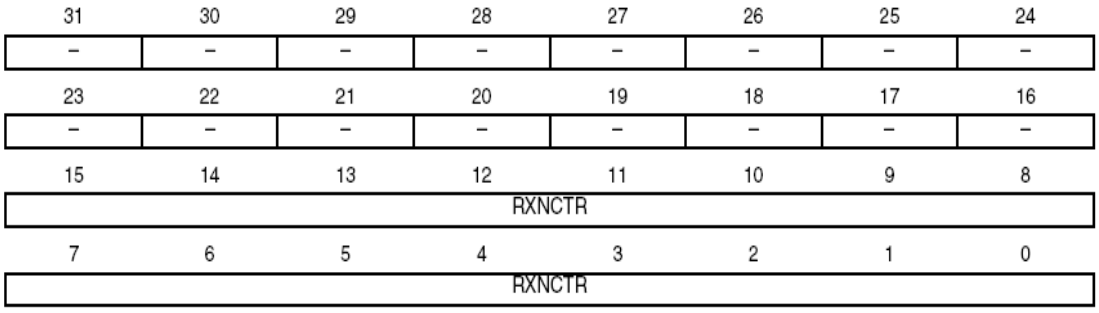
RXNPTR 包含下一次接收缓冲器地址。

当一个半双工外设连接于 PDC, RXNPTR = TXNPTR。

23.4.6 接收下一个计数器寄存器

寄存器名称：PERIPH_RNCR

访问类型：读/写



微控电子 www.mcuzone.com www.ATARM.com

- RXNCTR：接收下一个计数器

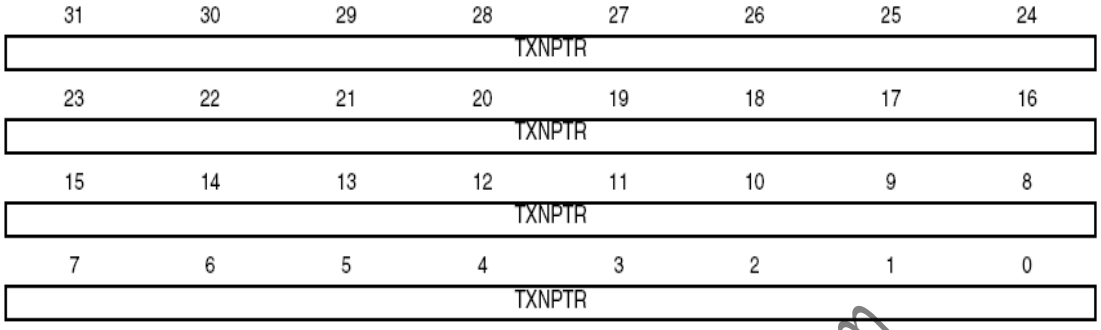
RXNCTR 包含下一个接收缓冲器大小

当一个半双工外设被连接于 PDC, RXNCTR = TXNCTR。

23.4.7 发送下一个计数器寄存器

寄存器名称：PERIPH_TNPR

访问类型：读/写



- TXNPTR：发送下一个指针

TXNPTR 包含下一个发送缓冲器地址

当一个半双工外设备连接于 PDC, RXNPTR = TXNPTR。

23.4.8 发送下一个计数器寄存器

寄存器名称：PERIPH_TNCR

访问类型：读/写

- TXNCTR：发送下一个计数器

TXNCTR 包含下一个发送缓冲器大小

当一个半双工外设被连接于 PDC, RXNCTR = TXNCTR。

23.4.9 传送控制寄存器

寄存器名称：PERIPH_PTCR

访问类型： 写

微控电子 www.mcuzone.com www.ATARM.com

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

- RXTEN：接收器传送使能

0=无效

1=如果 RXTDIS 未被置位，使能 PDC 外设通道请求

当一个半双工外设被连接于 PDC，使能接收通道请求自动的禁止发送器通道请求 q。对一个半双工外设置位 TXTEN 和 RXTEN 来禁用它。

- RSTDIS：接受器传送使无效

0=无效

1=使无效 PDC 接收器通道请求

当一个半双工外设被连接于 PDC，使无效接收器通道请求还使无效发送器通道请求

- TXTEN：发送器传送使能

0=无效

1=使能 PDC 发送器通道请求

当一个半双工外设被连接于 PDC，只要 RXTEN 未置位它就使能发送器通道请求。对一个半双工外设置位 TXTEN 和 RXTEN 来禁用它。

- TXTDIS：发送器传送使无效

0=无效

1=使无效 PDC 发送器通道请求

当一个半双工外设被连接于 PDC，使无效发送器通道请求，使无效接收器通道请求

23.4.10 传送状态寄存器

寄存器名称：PERIPH_PTSR

访问类型：读

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

- RXTEN：接收器传送使能

0=PDC接收器通道请求被使无效

1= PDC接收器通道请求被使能

- TXTEN：发送器传送使能

0=PDC发送器通道请求被使无效

1= PDC发送器通道请求被使能

微控电子 www.Mcuzone.com www.ATARM.com

24 时钟发生器

24.1 描述

时钟发生器由2个PLL，一个主振荡器，还有一个RC振荡器和一个32，768Hz低功耗振荡器组成。

它提供如下时钟：

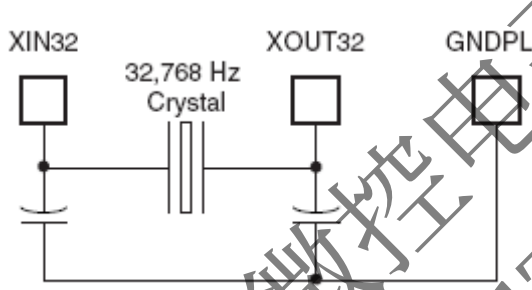
- SLCK，慢时钟，慢时钟是系统内唯一的永久时钟
- MAINCK是主振荡器的输出

  时钟发生器用户接口被嵌入在电源管理控制器中，如25.10段中描述。但是时钟发生器寄存器被命名为CKGR_。
- PLLACK是分频器和PLL A块的输出
- PLLBCK是分频器和PLL B块的输出

24.2 慢时钟晶振

时钟发生器集成一个32，768Hz的低功耗振荡器。XIN32 和 XOUT32引脚必须被连接到一个32，768Hz的晶体。如图24-1所示，必须接两个外部电容。

图24-1　典型慢时钟晶振接法



24.3 慢时钟RC振荡器

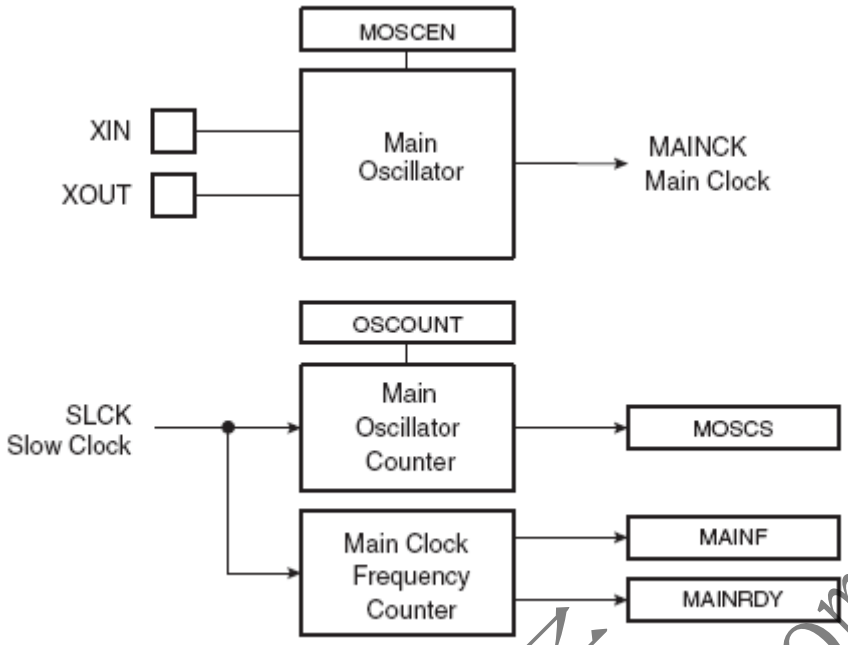
用户必须考虑RC振荡器可能的漂移。更多细节参见产品datasheet“DC特性”段。

24.4 主振荡器

图24-2所示为主振荡器方块图

图24-2　主振荡器方块图

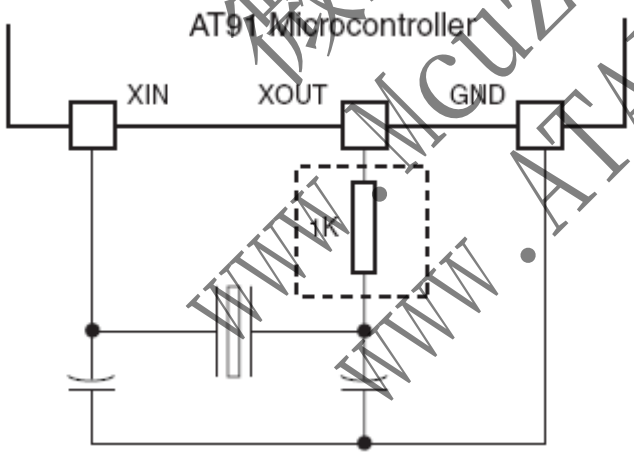
微控电子 www.mcuzone.com www.ATARM.com



24.4.1 主振荡器的连接

时钟发生器集成了一个主振荡器，主振荡器设计为可用于一个3到20MHz的晶体。此晶体的典型连接如图24-3说明。1 k欧的电阻仅用于频率低于8MHz的晶体。更多关于主振荡器的电特性，参见产品datasheet的“DC特性”段。

图24-3 典型晶振的连接



24.4.2 主振荡器启动时间

产品datasheet的DC特性段给出了主振荡器的启动时间。启动时间依赖于晶体频率并当频率升高时减小。

24.4.3 主振荡器控制

为了降低系统启动的功耗，主振荡器在复位后被使无效而选择慢时钟。

软件使能或使无效主振荡器可通过清零主振荡器寄存器(CKGR_MOR)中的MOSCEN位来实现，并可以减小功耗。

当通过清零CKGR_MOR中的MOSCEN位禁用主振荡器，PMC_SR中的MOSCS位被自动的清零，指示主时钟关闭。

微控电子 www.mcuzone.com www.ATARM.com

当使能主振荡器时，用户必须用对应于振荡器启动时间的一个值来初始化主振荡器计数器。启动时间依赖于连接于主振荡器的晶体频率。

通过写CKGR_MOR中的MOSCEN位和OSCOUNT来使能主振荡器，PMC_SR（状态寄存器）中的MOSCS位被清零并且计数器从OSCOUNT的值以慢时钟的8分频开始递减计数。因为OSCOUNT值用8位编码，最大启动时间大约62ms。(256*8*(1/32768) = 62.5ms)

当计数器到达0，MOSCS位置位，指示主时钟有效。置位PMC_IMR中的MOSCS位可以触发一个中断到处理器。

24.4.4 主时钟频率计数器

主振荡器有一个主时钟频率计数器可提供连接于主振荡器的晶振频率。一般地，该值可被系统设计者所知；然而，对于启动程序该值就很有用，它可以利用这个值来正确地配置设备的时钟速率，而与应用程序无关。

只要主振荡器稳定，也就是说，只要MOSCS位置位，主时钟频率计数器在慢时钟下一次上升沿后以主时钟速度开始增长。接着，在慢时钟的第16个下降沿，CKGR_MCFR中的MAINRDY位被置位并且计数器停止计数。其值可在CKGR_MCFR的MAINF域中被读取，表征在慢时钟的16个周期期间主时钟周期数，所以可以确定连接在主振荡器上的晶体频率。

24.4.5 主振荡器旁路

用户可在设备上输入一个时钟代替连接晶体。此情况下，用户必须在XIN引脚上提供外部时钟信号。此条件下XIN引脚的输入特性在产品的电特性段给出。为使外部时钟操作正确，编程者必须确保设置主振荡器寄存器(CKGR_MOR)中的OSCBYPASS位为1，MOSCEN位为0。

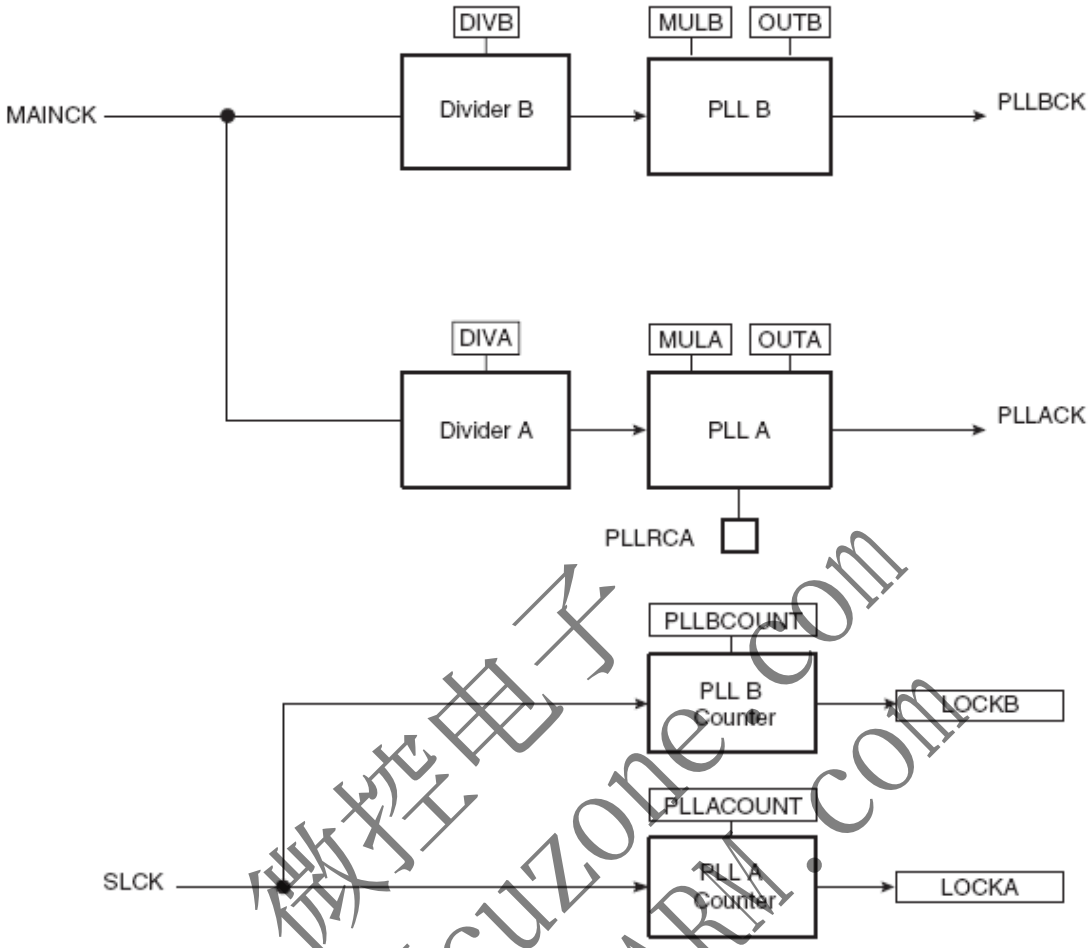
24.5 分频器和PLL

PLL包含一个输入分频器来增加所产生的时钟信号的准确性。然而，当用户编程分频器时，必须考虑PLL最小的输入频率。

图24-4 展示了分频器和PLL的方块图

微控电子 www.Mcuzone.com www.ATARM.com

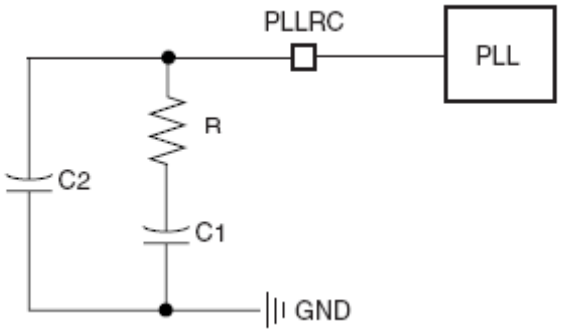
图24-4　分频器和PLL方块图



24.5.1　PLL滤波器

PLL需要通过PLLRCA和/或PLLRCB引脚连接外部二阶滤波器。图24-5展示了这些滤波器的略图。

图24-5　PLL电容和电阻



被连接到PLLRC引脚的R, C1 和 C2值由PLL输入频率，PLL输出频率和相位裕度的公式推算。必须在输出信号过冲和启动时间之间找到一个平衡。

24.5.2　分频器和锁相环编程

分频器可在第一步时设置在1到255之间。当一个分频器域(DIV)被设置为0，

微控电子 www.Mcuzone.com www.ATARM.com

对应分频器的输出和PLL输出是一个0电平的连续信号。复位时，每个DIV域均被设置为0，因此对应的PLL输入被设置为0。

PLL允许分频器输出相乘。PLL时钟信号依赖于各自信号源的频率及参数DIV和MUL。相当于信号源频率的(MUL + 1)/DIV。

当MUL被写为0，将禁用对应的PLL并且可节省其功耗。可通过在MUL域写入一个大于0的值来重新使能PLL。

无论何时重新使能PLL或其参数之一被改变，PMC_SR中的LOCK 位(LOCKA 或LOCKB)会被自动清零。写入 CKGR_PLLR (CKGR_PLLAR 或CKGR_PLLBR)中的PLLCOUNT 域中(PLLA-COUNT或PLLBCOUNT)的值，被装载进PLL计数器。PLL计数器接着以慢时钟速度减小直到其到达0。这时候，LOCK位被设置在PMC_SR中并可以触发一个中断到处理器。用户必须装载所需要的慢时钟周期数到PLLCOUNT域以满足PLL暂态时间的要求。暂态时间取决于PLL滤波器。PLL的初始状态和目标频率可用一个由Atmel提供的特殊工具计算。

25 电源管理控制器（PMC）

25.1 描述

电源管理控制器(PMC)通过控制所有系统和用户外设时钟来优化功耗。PMC使能/禁用多个外设和ARM处理器的时钟输入。

电源管理控制器提供以下时钟：

1. MCK，主控时钟，可编程为从几百Hz到设备的最大操作频率。对永久运行的模块是可用的，像AIC和存储控制器。
2. 处理器时钟(PCK)，当进入处理器空闲模式时PCK被关闭。
3. 外设时钟，典型的就是MCK，提供给嵌入式外设(USART, SSC, SPI, TWI, TC, MCI, 等)并且可以独立控制。为了减少产品中的时钟名称数，外设时钟在产品数据手册上被命名为MCK。
4. HClocks (HCKx)，提供给AHB/ASB高速外设并且可以独立控制。
5. UHP 时钟 (UHPCK)，USB主机端口操作所需的时钟。
6. 可编程的时钟输出可由时钟发生器提供并驱动PCKx引脚。

25.2 主时钟控制器

主时钟控制器提供主时钟(MCK)的选择和分频。MCK是提供给所有外设和存储控制器的时钟。

主时钟是时钟发生器提供的时钟之一。选择慢时钟提供一个慢时钟信号给整个外设。选择其为主时钟可节省PLL的功耗。

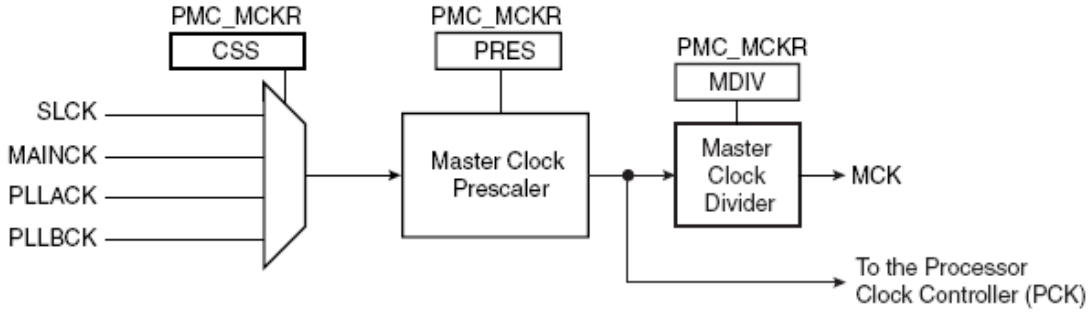
主时钟控制器由时钟选择器和一个预分频器组成。它还包括一个主时钟分频器，此分频器允许处理器时钟比主时钟快。

主时钟通过写PMC_MCKR中（主控时钟寄存器）的CSS域来选择。预分频器支持将被选择的时钟进行分频，分频因子在2的1到64的幂之间。

PMC_MCKR中PRES域编程预分频器。可通过PMC_MCKR中MDIV域编程主时钟分频器。

每次写PMC_MCKR定义一个新主时钟，PMC_SR中的MCKRDY位被清零。且一直读取为0，直到主时钟被建立。接着，MCKRDY被置位并可触发一个处理器中断。此功能在从高速时钟切换到低速时钟时是很有用的，可以用于通知应用程序切换完成。

图25-1　主控时钟控制器



25.3 处理器时钟控制器

PMC有一个处理器时钟控制器(PCK)以实现处理器空闲模式。处理器时钟可通过写系统时钟禁用寄存器(PMC_SCDR)来禁用。此时钟的状态（至少可用于调试目的）在系统时钟状态寄存器(PMC_SCSR)中被读取。

处理器时钟PCK在复位后被使能并被任何使能中断自动地重新使能。通过禁用处理器时钟来达到处理器空闲模式，通过任何使能快速或正常中断自动的重新使能，或通过产品复位键。

当处理器时钟被禁用，当前指令在时钟停止前完成，但是这并不阻止数据从系统总线的其他主控传送。

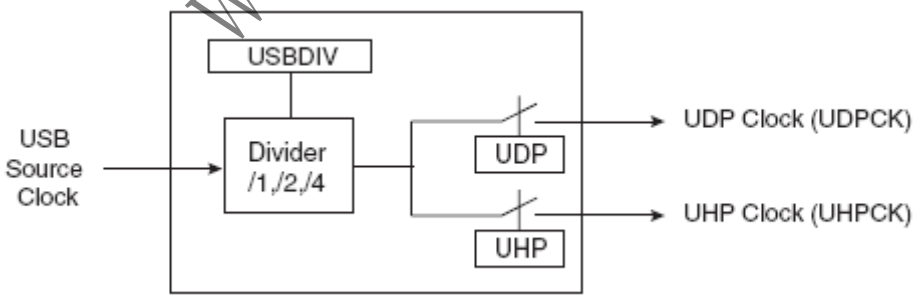
25.4 USB时钟控制器

总是从PLL B输出产生USB时钟源。如果使用USB，用户必须编程PLL来产生准确度为±0.25%的一个48MHz，一个96 MHz，或一个 192 MHz信号，准确度依赖于CKGR_PLLBR中的USBDIV位（见图25-2）。

当PLL B输出稳定，就是说，LOCKB置位：

- USB主机时钟通过置位PMC_SCER中的UHP位使能。当此外设未用时为节省其上的功耗，用户可以置位PMC_SCDR中的UHP位。PMC_SCSR中的UHP位给出了此时钟的使能状态。USB主机端口需要12/48MHz信号和主时钟。可通过主控时钟控制器控制主控时钟。

图25-2　USB时钟控制器



25.5 外设时钟控制器

电源管理控制器通过外设时钟控制器来控制每个嵌入式外设的时钟。

用户可通过写入外设时钟使能寄存器(PMC_PCER)和外设时钟禁用寄存器(PMC_PCDR)来独立的使能和禁用外设上的主时钟。外设时钟使能状态可在外设时钟状态寄存器(PMC_PCSR)中读取。

www.mcuzone.com www.ATARM.com

当一个外设时钟被禁用，时钟立即被停止。外设时钟在复位后被自动的禁用。停止一个外设时，推荐系统软件先等待，一直到禁用时钟前外设已执行其最后编程的操作。这是为了避免数据讹误或系统的误动作。

外设时钟控制寄存器(PMC_PCER, PMC_PCDR,和PMC_PCSR)中的位数是依据产品定义的外设标识符。总之，位数对应于指定外设的中断源数。

25.6 HClock 控制器

PMC通过HClock控制器使每个特定AHB/ASB外设的时钟控制变得容易。用户可以通过写入寄存器：系统时钟使能寄存器(PMC_SCER)和系统时钟禁用寄存器(PMC_SCDR)独立的使能和禁用Hclocks。HClock使能状态可在系统时钟状态寄存器(PMC_SCSR)中读取。

禁用一个HClock时，时钟立即被停止。当HClock被重新使能，外设在其停止的地方重新开始运行。HClocks在复位后被自动的禁用。

1 HClock可以被控制。

25.7 可编程的时钟输出控制器

PMC控制输出到外部引脚PCKx的4个信号。每个信号通过PMC_PCKx寄存器被独立的编程。

PCKx可通过写PMC_PCKx中的CSS域在慢时钟，PLL A输出，PLL B输出和主时钟之间独立的选择。每个输出信号还可通过写PMC_PCKx中的PRES（预分频器）域加以分频，分频因子可为2的1到64次幂之间。

每个输出信号可通过在对应的位，PMC_SCER 和 PMC_SCDR的PCKx，写1来被使能或禁用。有效的可编程输出时钟的状态在PMC_SCSR（系统时钟状态寄存器）的PCKx位中被给出。

另外，像PCK，PMC_SR中的状态位指示可编程的时钟实际上是在可编程寄存器中已被编程。

因为当变换时钟时，可编程时钟控制器不处理信号毛刺，强烈推荐在任何配置改变前禁用可编程时钟并在改变确实执行后重新使能可编程时钟。

25.8 编程顺序

1. 使能主振荡器：

通过置位CKGR_MOR寄存器中的MOSCEN域使能主振荡器。有些情况下，定义一个启动时间可能更有利。这可通过在CKGR_MOR寄存器中的OSCOUNT域写一个值来实现。

一旦此寄存器被正确的配置，用户必须等待PMC_SR寄存器中的MOSCS域被置位。这可通过轮询状态寄存器或通过等待中断口线有效来完成，如果与MOSCS关联的中断在PMC_IER寄存器中已被使能。

编码举例：

```
write_register(CKGR_MOR,0x00000701)
```

启动时间=8 * OSCOUNT / SLCK =56个慢时钟周期。

所以，主振荡器会在56个慢时钟周期后被使能（MOSCS 位置位）。

2. 检查主振荡器频率（可选）：

有些条件下，用户可能需要主振荡器频率的一个准确测量。此测量可以通过CKGR_MCFR寄存器来完成。

只要MAINRDY域在CKGR_MCFR寄存器中被置位，用户可以读取CKGR_MCFR寄存器中的MAINF域。这可以提供16个慢时钟周期内的主时钟周期数。

3． 设置PLL A和分频器A：

需要配置PLL A和分频器A的所有参数被设置在CKGR_PLLAR寄存器中。

当编程CKGR_PLLAR寄存器时，注意必须置位位29。

DIVA域被用于控制分频器A本身。用户可编程一个0到255之间的值 。分频器A的输出是被DIVA分频的分频器A的输入。缺省状态下，DIVA参数被设置为0来关闭分频器A。

OUTA域被用来选择PLL A的输出频率范围。

MULA域是PLL A乘数。此参数可被编程在0到2047之间。如果MULA被设置为0，PLL A将被关闭。否则PLL A的输出频率是(MULA + 1)与PLL A输入相乘的值。

CKGR_PLLAR寄存器被写后，PLLACOUNT域指定经过多少慢时钟周期后，在PMC_SR寄存器中的LOCKA位被置位。

只要CKGR_PLLAR寄存器一被写，用户必须等待PMC_SR寄存器中LOCKA位被置位。这可以通过轮询状态寄存器或通过等待中断口线有效来完成，后者要求与LOCKA关联的中断在PMC_IER寄存器中被使能。

CKGR_PLLAR中的所有参数可在一个单一的写操作中完成。如果在某些情况下，以下参数之一，SRCA, MULA, DIVA被修改，LOCKA位将变低来指示PLL A未就绪。当PLL A被锁定，LOCKA将被再次置位。用户在使用PLL A输出时钟前，必须等待LOCKA位被置位。

编码举例：

```
write_register(CKGR_PLLAR,0x20030605)
```

PLL A和分频器A被使能，PLL A输入时钟是被5分频的主时钟。PLL 的一个输出时钟是4与PLL A输入时钟相乘的值。只要CKGR_PLLAR一被写，LOCKA位将在6个慢时钟周期后被置位。

4． 设置PLL B和除法器B：

所有配置PLL B和分频器B所需要的参数均在CKGR_PLLBR寄存器中。

DIVB域被用于控制分频器B本身。可编程0到255之间的值。分频器B输出是被DIVB分频的分频器B的输入值。缺省情况下，DIVB参数被设置为0来关闭分频器。

OUTB域被用于选择PLL B输出频率范围。

MULB域是PLL B乘数。此参数可在0到2047之间被编程。如果MULB被设置为0，PLL B将被关闭，否则PLL B输出频率是(MULB + 1)与PLL B输入频率的乘积。

CKGR_PLLBR寄存器被写后，PMC_SR寄存器中的LOCKB位在经过PLLBCOUNT域所确定的慢时钟周期数后被置位。

只要PMC_PLLB寄存器被写，用户必须等待在PMC_SR寄存器中的LOCKB位被置位。这可以通过轮询状态寄存器或通过等待中断口线有效来完成，如果与LOCKB关联的中断在PMC_IER寄存器中已被使能。CKGR_PLLBR中的所有参数可在一个单独的写操作中被编程。如果在某些情况下以下参数之一，如MULB, DIVB被修改，LOCKB位将变低来指示PLL B未就绪。当PLL B被锁定，LOCKB将被再次置位。用户使用PLL B输出时钟前必须等待LOCKB位被

微控电子 WWW.MCUZONE.COM WWW.ATARM.COM

置位。

USBDIV域被用于产生USB时钟所需要的附加1，2或4分频。

编码举例：

```
write_register(CKGR_PLLBR,0x00040805)
```

如果PLL B和分频器B被使能，PLL B输入时钟是主时钟。PLL B输出时钟是5与PLL B输入时钟的乘积。只要CKGR_PLLBR一被写，LOCKB位将在8个慢时钟周期后被置位。

5． 主控时钟和处理器时钟的选择。

主控时钟和处理器时钟通过PMC_MCKR寄存器可被配置。

CSS域被用于选择主时钟分频器源。缺省状态下，被选择的时钟源是慢时钟。

PRES域被用于控制主控时钟预分频器。用户可在不同值(1, 2, 4, 8, 16, 32, 64)之间选择。主时钟输出是被PRES参数分频的预分频器的输入值。缺省状态下，PRES参数被置为1，此时主时钟等于慢时钟。

MDIV域被用于控制主时钟预分频器。可以在不同值(0, 1, 2)之间选择。

主时钟输出是被1，2或4分频的处理器时钟，取决于在MDIV中编程的值。缺省状态下，MDIV被设置为0，处理器时钟等于主时钟。

只要PMC_MCKR寄存器一被写，用户就必须等待在PMC_SR寄存器中的MCKRDY位被置位。这可以通过轮询状态寄存器或通过等待中断口线有效来完成，如果与MCKRDY关联的中断在PMC_IER寄存器中被使能。

PMC_MCKR寄存器不可以在一个单独写操作中被编程。推荐对PMC_MCKR寄存器按下面顺序编程：

- 如果CSS域的一个新值对应于PLL时钟，
  -在PMC_MCKR寄存器中编程PRES域
  -等待在PMC_SR寄存器中的MCKRDY位被置位。
  -在PMC_MCKR寄存器中编程CSS域
  -等待在PMC_SR寄存器中的MCKRDY位被置位。
- 如果CSS域的一个新值对应于主时钟或慢时钟，
  -在PMC_MCKR寄存器中编程CSS域
  -等待在PMC_SR寄存器中的MCKRDY位被置位。
  -在PMC_MCKR寄存器中编程PRES域
  -等待在PMC_SR寄存器中的MCKRDY位被置位。

如果在某些时候，以下参数之一，CSS 或 PRES，被修改，MCKRDY位将变低来指示主时钟和处理器时钟未就绪。用户在使用主控和处理器时钟前必须等待MCKRDY位被置位。

注意：如果PLLx时钟被选为主时钟并且用户决定通过写CKGR_PLLR (CKGR_PLLAR 或 CKGR_PLLBR)来修改它，MCKRDY标志位在PLL未被锁定时将变低。只要PLL再次被锁定，LOCK (LOCKA 或 LOCKB)变高并且MCKRDY被置位。当PLLA未被锁定，主控时钟选择被自动的改变为慢时钟。当PLLB未被锁定，主控时钟选择被自动的改变为主振荡器时钟。更多信息，见25.9.2段，259页“时钟变换波形”。

编码举例：

```
write_register(PMC_MCKR,0x00000001)
wait (MCKRDY=1)
write_register(PMC_MCKR,0x00000011)
wait (MCKRDY=1)
```

微控电子 www.mcuzone.com www.ATARM.com

主控时钟是被16分频的主振荡器时钟。
处理器时钟是主控时钟。
6． 可编程时钟的选择
通过以下寄存器控制可编程时钟：PMC_SCER, PMC_SCDR 和 PMC_SCSR。
可通过PMC_SCER 和 PMC_SCDR使能和/或禁用可编程时钟。依赖于使用的系统，4个可编程时钟可以被使能或禁用。PMC_SCSR提供一个清楚的指示来指明哪个可编程时钟是使能的。缺省情况下，禁用所有可编程时钟。
PMC_PCKx寄存器被用于配置可编程时钟。
CSS域被用于选择可编程时钟分频器源，4个时钟选项可用：主时钟，慢时钟，PLLACK, PLLBCK。缺省时，选择的时钟源是慢时钟。
PRES域被用于控制可编程时钟预分频器。可以在不同值(1, 2, 4, 8, 16, 32, 64)间选择。可编程时钟输出是被PRES参数分频的预分频器的输入值。缺省时，PRES参数被设置为1表明主控时钟等于慢时钟。
只要PMC_PCKx寄存器一被编程，对应的可编程时钟必须被使能并且用户必须等待在PMC_SR寄存器中的PCKRDYx位被置位。这可以通过轮询状态寄存器或通过等待中断口线有效来完成，如果与PCKRDYx的关联中断在PMC_IER寄存器中已被使能。PMC_PCKx中的所有参数可以在一个单独的写操作中被编程。
如果CSS和PRES参数要被修改，对应的可编程时钟必须首先被禁用。然后参数可被修改。一旦这个已被完成，用户必须重新使能可编程时钟并等待PCKRDYx位被置位。
编码举例：

```
write_register(PMC_PCK0,0x00000015)
```

可编程时钟0是被32分频的主时钟。
7． 使能外设时钟
一旦所有前面的步骤已被完成，外设时钟可通过寄存器PMC_PCER 和 PMC_PCDR来使能和/或禁用。
依赖于使用的系统，17个外设时钟可被使能或禁用。PMC_PCSR提供清楚的信息，关于哪个外设时钟被使能。
注意：每个被使能的外设时钟等于于主控时钟
编码举例：

```
write_register(PMC_PCER,0x00000110)
```

外设时钟4和8被使能。

```
write_register(PMC_PCDR,0x00000010)
```

外设时钟4被禁用。
8． 使能HClock
一旦所有前面的步骤被完成，HClock可通过寄存器：PMC_SCER and PMC_SCDR被使能和/或禁用。
依赖于使用的系统，1 HClock可被使能或禁用。
PMC_SCSR寄存器指示哪个HClock被使能。
注意：每个使能的HClock等同于主控时钟。
编程举例：

```
write_register(PMC_SCER,0x00110000)
```

HClock 0 和 4 被使能。

25.9 时钟转换细节

25.9.1 主控时钟转换时间

表25-1和表25-2给出了需要主控时钟从一个已选择的时钟转换到另外的一个时钟的最坏情况下的时间需求。这里假设预分频器被禁用。当预分频器被激活，必须被添加一个额外时间，其值为64个新选择的时钟的周期。

表25-1 时钟变换时间（最坏情况）

从 到	主时钟	SLCK	PLL Clock
主时钟	-	4 x SLCK +2.5 x主时钟	3 x PLL Clock +4 x SLCK +1 x主时钟
SLCK	0.5 x主时钟+4.5 x SLCK	-	3 x PLL Clock +5 x SLCK
PLL Clock	0.5 x主时钟+4 x SLCK +PLLCOUNT x SLCK +2.5 x PLLx Clock	2.5 x PLL Clock +5 x SLCK +PLLCOUNT x SLCK	2.5 x PLL Clock +4 x SLCK +PLLCOUNT x SLCK

注意：1. PLL选定PLL A或PLL B时钟。

2. PLLCOUNT选定PLLACOUNT或PLLBCOUNT.

表25-2 两个PLL之间的时钟转换时间（最坏情况）

从 到	PLLA时钟	PLLB时钟
PLLA时钟	2.5 x PLLA Clock + 4 x SLCK + PLLACOUNT x SLCK	3 x PLLA Clock + 4 x SLCK + 1.5 x PLLA Clock
PLLB时钟	3 x PLLB Clock +4 x SLCK + 1.5 x PLLB Clock	2.5 x PLLB Clock + 4 x SLCK + PLLBCOUNT x SLCK

25.9.2 时钟转换波形

图25-3 主控时钟从慢时钟到PLL时钟

微控电子 www.mcuzone.com www.ATARM.com

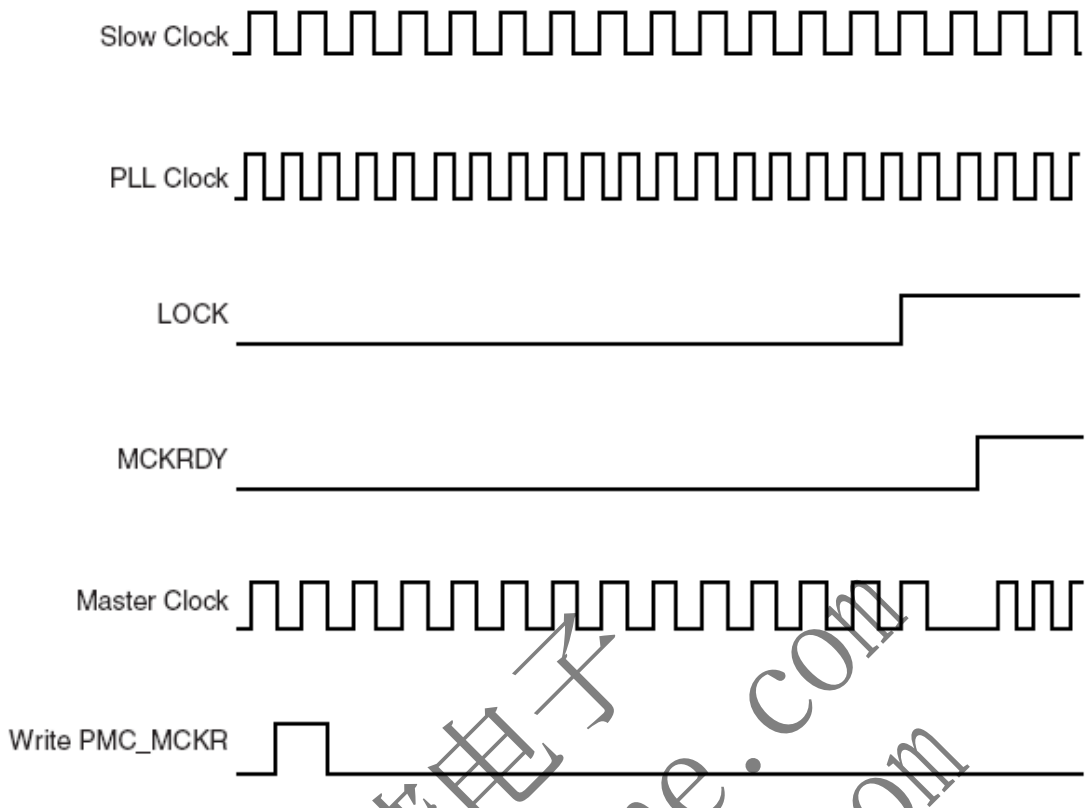


图25-4　主控时钟从主振荡器时钟转换到慢时钟

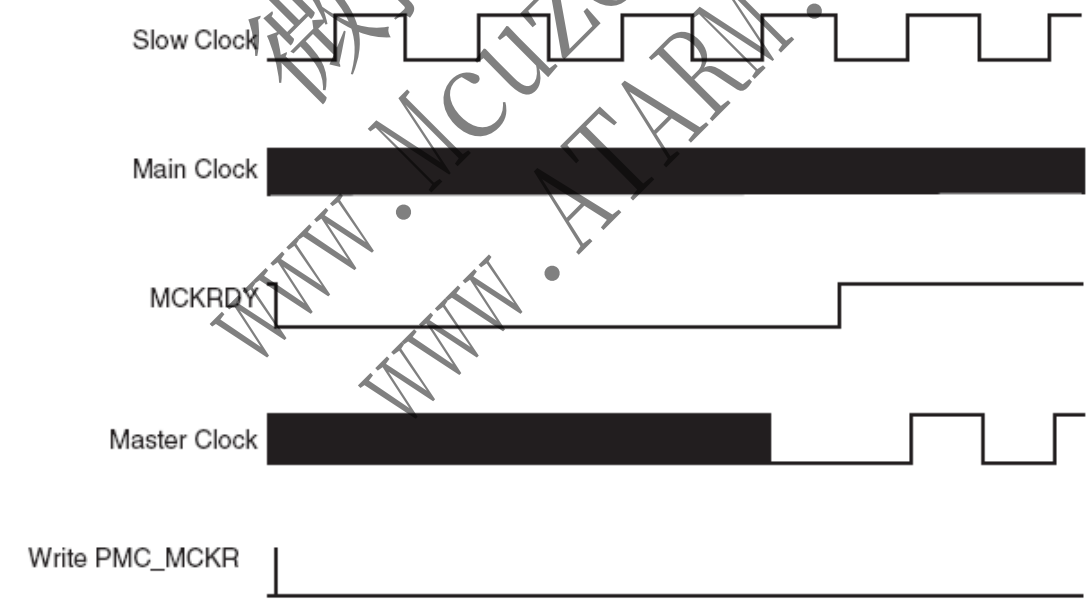


图25.5　改变PLLA编程

微控电子 www.mcuzone.com www.ATARM.com

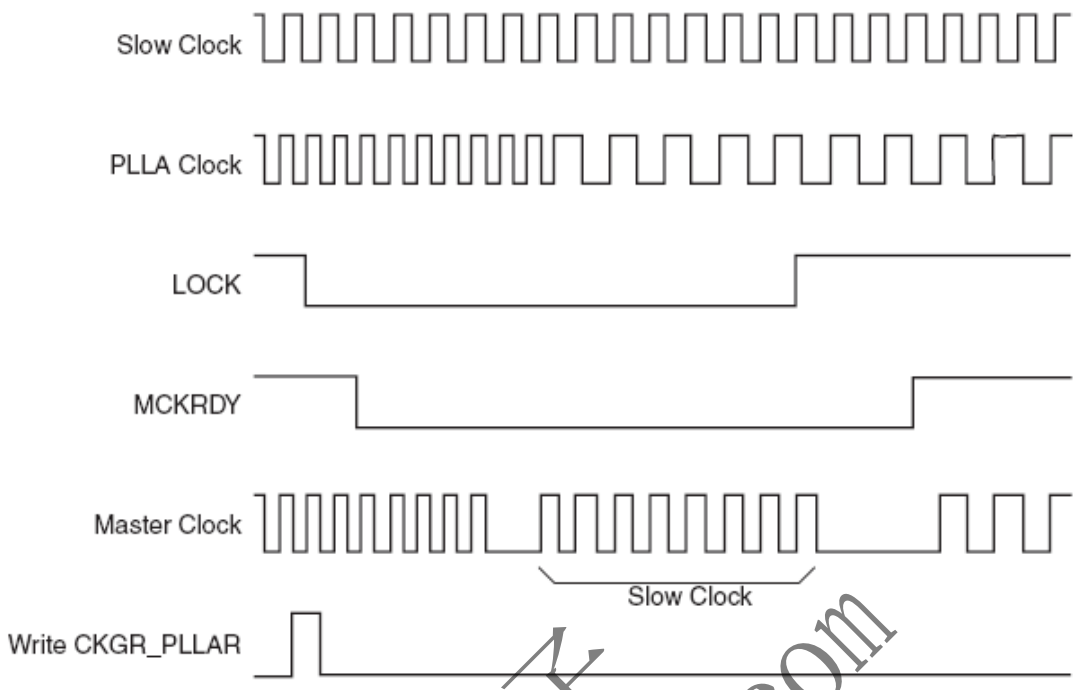


图25-6　改变PLLB编程

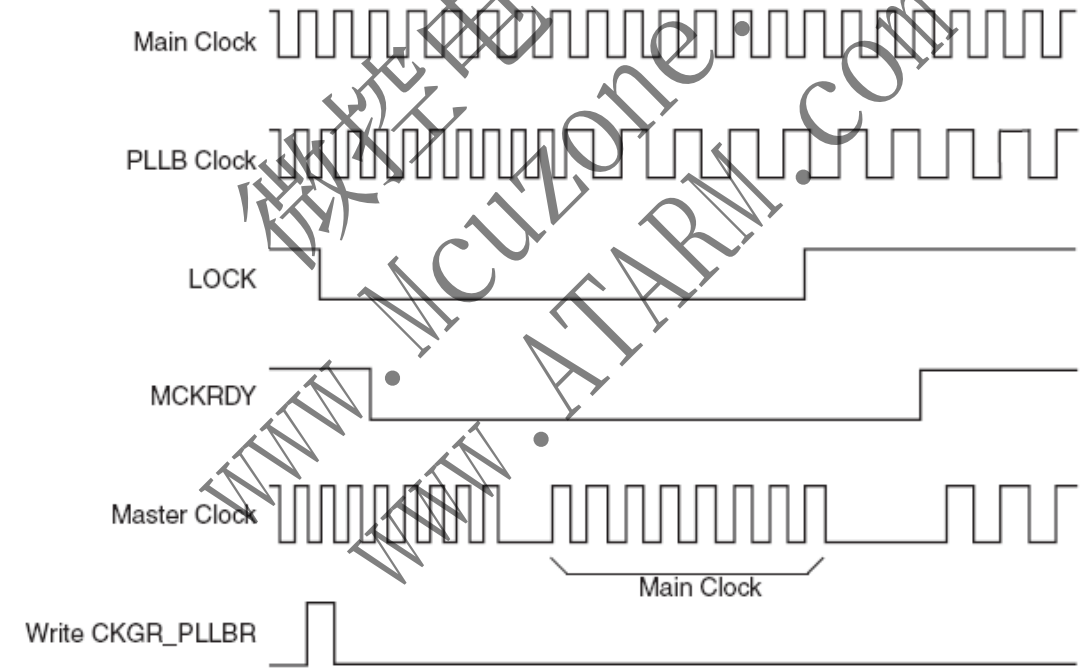
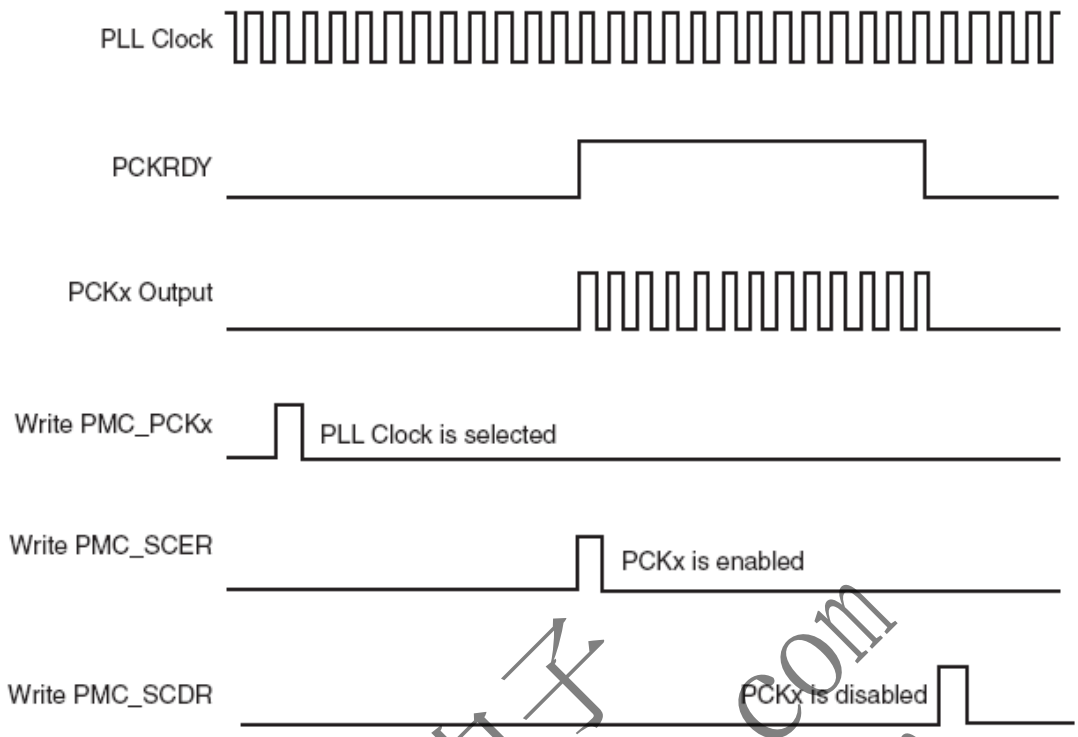


图25-7　可编程时钟输出编程

微控电子 www.Mcuzone.com www.ATARM.com



25.10 电源管理控制器(PMC)用户接口

表25-3 寄存器映射

偏移量	寄存器	名称	访问	复位值
0x0000	系统时钟使能寄存器	PMC_SCER	只写	-
0x0004	系统时钟禁用寄存器	PMC_SCDR	只写	-
0x0008	系统时钟状态寄存器	PMC _SCSR	只读	0x03
0x000C	保留	-	-	-
0x0010	外设时钟使能寄存器	PMC _PCER	只写	-
0x0014	外设时钟禁用寄存器	PMC_PCDR	只写	-
0x0018	外设时钟状态寄存器	PMC_PCSR	只读	0x0
0x001C	保留	-	-	-
0x0020	主振荡器寄存器	CKGR_MOR	读/写	0x0
0x0024	主时钟频率寄存器	CKGR_MCFR	只读	0x0
0x0028	PLL A寄存器	CKGR_PLLAR	读写	0x3F00
0x002C	PLL B寄存器	CKGR_PLLBR	读写	0x3F00
0x0030	主控时钟寄存器	PMC_MCKR	读/写	0x0
0x0038	保留	-	-	-
0x003C	保留	-	-	-
0x0040	可编程时钟0寄存器	PMC_PCK0	读/写	0x0

微控电子 www.Mcuzone.com www.ATARM.com

0x0044	可编程时钟1寄存器	PMC_PCK1	读/写	0x0
…	…	…	…	…
0x0060	中断使能寄存器	PMC_IER	只写	-
0x0064	中断禁用寄存器	PMC_IDR	只写	-
0x0068	状态寄存器	PMC_SR	只读	0x08
0x006C	中断屏蔽寄存器	PMC_IMR	只读	0x0
0x0070-0x007C	保留	-	-	-
0x0080	电荷泵电流寄存器	PMC_PLLICPR	只写	-
0x0084-0x00FC	保留	-	-	-

25.10.1 PMC系统时钟使能寄存器

寄存器名称：PMC_SCER

访问类型：只写

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	HCK1	HCK0
15	14	13	12	11	10	9	8
–	–	–	–	PCK3	PCK2	PCK1	PCK0
7	6	5	4	3	2	1	0
UDP	UHP	–	–	–	–	–	–

- UHP：USB主机端口时钟使能
  0=无效
  1=使能USB主机接口12和48MHz的时钟
- UDP：USB设备端口时钟使能
  0=无效
  1=使能USB设备端口的48MHz时钟
- PCKx：可编程时钟x输出使能
  0=无效
  1=使能对应的可编程时钟输出
- HCKx：**HClock** x输出使能
  0=无效
  1=使能对应HClock输出

25.10.2 PMC系统时钟禁用寄存器

寄存器名称：PMC_SCDR

访问类型：只写

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	HCK1	HCK0
15	14	13	12	11	10	9	8
–	–	–	–	PCK3	PCK2	PCK1	PCK0
7	6	5	4	3	2	1	0
UDP	UHP	–	–	–	–	–	PCK

微控电子 www.mcuzone.com www.ATARM.com

- PCK：处理器时钟禁用

  0=无效

  1=禁用处理器时钟。这被用于进入处理器空闲模式。

- UHP：USB主机端口时钟禁用

  0=无效

  1=禁用USB主机端口的12 和48MHz时钟

- UDP：USB设备端口时钟禁用

  0=无效

  1=禁用USB设备端口的48MHz时钟

- PCKx：可编程时钟x输出禁用

  0=无效

  1=禁用对应的可编程时钟输出

- HCKx：Hclock x 输出禁用

  0=无效

  1=禁用对应的HClock输出

25.10.3 PMC系统时钟状态寄存器

寄存器名称：PMC_SCSR

访问类型：只读

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	HCK1	HCK0
15	14	13	12	11	10	9	8
–	–	–	–	PCK3	PCK2	PCK1	PCK0
7	6	5	4	3	2	1	0
UDP	UHP	–	–	–	–	–	PCK

- PCK：处理器时钟状态

  0=处理器时钟被禁用

  1=处理器时钟被使能

- UHP：USB主机端口时钟状态

  0=USB主机端口的12和48MHz时钟(UHPCK)被禁用

  1=USB主机端口的12和48MHz时钟(UHPCK)被使能

- UDP：USB设备端口时钟状态

  0=USB设备端口的48MHz时钟(UDPCK)被禁用

  1=USB设备端口的48MHz时钟(UDPCK)被使能

- PCKx：可编程时钟x输出状态

  0=对应的可编程时钟输出被禁用

  1=对应的可编程时钟输出被使能

- HCKx：HClock输出x状态

  0=对应的HClock输出被禁用

  1=对应的HClock输出被使能

25.10.4 PMC外设时钟使能寄存器

寄存器名称：PMC_PCER

微控电子 www.mcuzone.com www.ATARM.com

访问类型：只写

31	30	29	28	27	26	25	24
PID31	PID30	PID29	PID28	PID27	PID26	PID25	PID24
23	22	21	20	19	18	17	16
PID23	PID22	PID21	PID20	PID19	PID18	PID17	PID16
15	14	13	12	11	10	9	8
PID15	PID14	PID13	PID12	PID11	PID10	PID9	PID8
7	6	5	4	3	2	1	0
PID7	PID6	PID5	PID4	PID3	PID2	-	-

- PIDx：外设时钟x使能

  0=无效

  1=使能对应的外设时钟

  注意：PID2到PID31参考在产品数据手册“外设标识符”段中定义的标识符。

  注意：编程未实现的外设ID的控制位不影响PMC的动作。

25.10.5 PMC外设时钟禁用寄存器

寄存器名称：PMC_PCDR

访问类型：只写

31	30	29	28	27	26	25	24
PID31	PID30	PID29	PID28	PID27	PID26	PID25	PID24
23	22	21	20	19	18	17	16
PID23	PID22	PID21	PID20	PID19	PID18	PID17	PID16
15	14	13	12	11	10	9	8
PID15	PID14	PID13	PID12	PID11	PID10	PID9	PID8
7	6	5	4	3	2	1	0
PID7	PID6	PID5	PID4	PID3	PID2	-	-

- PIDx：外设时钟x禁用

  0=无效

  1=禁用对应的外设时钟

  注意：PID2到PID31参考在产品数据手册“外设标识符”段中定义的标识符。

25.10.6 PMC外设时钟状态寄存器

寄存器名称：PMC_PCSR

访问类型：只读

31	30	29	28	27	26	25	24
PID31	PID30	PID29	PID28	PID27	PID26	PID25	PID24
23	22	21	20	19	18	17	16
PID23	PID22	PID21	PID20	PID19	PID18	PID17	PID16
15	14	13	12	11	10	9	8
PID15	PID14	PID13	PID12	PID11	PID10	PID9	PID8
7	6	5	4	3	2	1	0
PID7	PID6	PID5	PID4	PID3	PID2	–	–

- PIDx：外设时钟x状态

  0=对应的外设时钟被禁用

  1=对应的外设时钟被使能

  注意：PID2到PID31参考在产品数据手册“外设标识符”段中定义的标识符。

25.10.7 PMC时钟发生器主振荡器寄存器

寄存器名称：CKGR_MOR

微控电子 www.mcuzone.com www.ATARM.com

访问类型：读/写

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

- MOSCEN：主振荡器使能

  晶体被连接于XIN和XOUT之间

  0=主振荡器被禁用

  1=主振荡器被使能。OSCBYPASS必须被设置为0

  当MOSCEN被置位，只要主振荡器启动时间到达，MOSCS标志位就被置位。
- OSCBYPASS：振荡器旁路

  0=无效

  1=主振荡器旁路。MOSCEN必须被设置为0。外部时钟必须被连接于XIN上。

  当OSCBYPASS被置位，PMC_SR中的MOSCS标志位被自动的置位。

  清零MOSCEN 和 OSCBYPASS位允许复位MOSCS标志位。
- OSCOUNT：主振荡器启动时间

  为主振荡器确定启动时间，数值指定为被8分频的慢时钟周期数。

25.10.8 PMC时钟发生器主时钟频率寄存器

寄存器名称：CKGR_MCFR

访问类型：只读

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

- MAIN：主时钟频率

  给出16个慢时钟周期内的主时钟周期数
- MAINRDY：主时钟就绪

  0=MAINF值无效或主振荡器被禁用

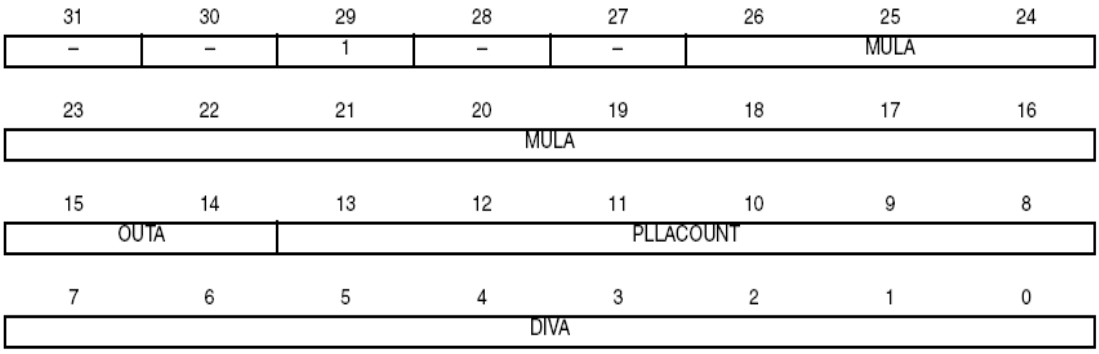
  1=主振荡器先前已被使能并MAINF值可用

25.10.9 PMC时钟发生器PLL A寄存器

寄存器名称：CKGR_PLLAR

访问类型：读/写

微控电子 www.mcuzone.com www.ATARM.com



使用PMC前应该检查PLL A输入频率和倍频上可能的限制。

警告：当编程CKGR_PLLAR寄存器时，位29必须总是被置为1。

- DIVA：分频器A

DIVA	选择的分频器
0	分频器输出是0
1	分频器被旁路
2-255	分频器输出是被DIVA分频的主振荡器时钟

- PLLACOUNT：PLL A计数器

CKGR_PLLAR被写后，PMC_SR寄存器中的POCKA位被置位前经过的慢时钟周期数。

- OUTA：PLL A时钟频率范围

为优化时钟性能，必须如产品数据手册电特性段的“PLL 特性”中的规定来编程此域。

- MULA：PLL A倍频因子

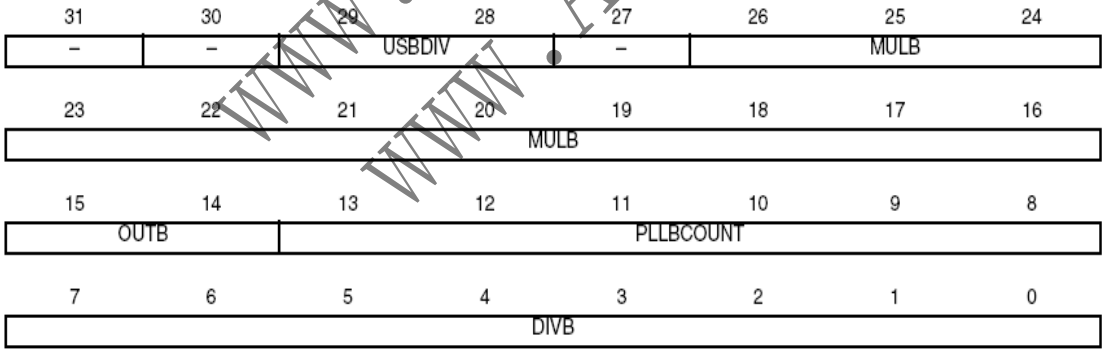
0=PLL A被禁用

1到2047=PLL A时钟频率是(MULA+1)与PLL A输入频率的乘积

25.10.10 PMC时钟发生器PLL B寄存器

寄存器名称：CKGR_PLLBR

访问类型：读/写



使用PMC前应该检查PLL B输入频率和乘数上的限制。

- DIVB：分频器B

DIVB	选择的分频器
0	分频器输出是0
1	分频器被旁路
2-255	分频器输出是被DIVB分频的已选择的时钟

- PLLBVOUNT：PLL B计数器

CKGR_PLLBR被写后，在PMC_SR中LOCKB位被置位前所需等待的慢时钟周期数。

微控电子 www.mcuzone.com www.ATARM.com

- OUTB：PLLB时钟频率范围

为优化时钟性能，必须如产品数据手册电特性段的“PLL 特性”中的规定来编程此域。

- MULB：PLL 乘数

0=PLL B被禁用

1到2047=PLL B时钟频率是(MULB+1)与PLL B输入频率的乘积

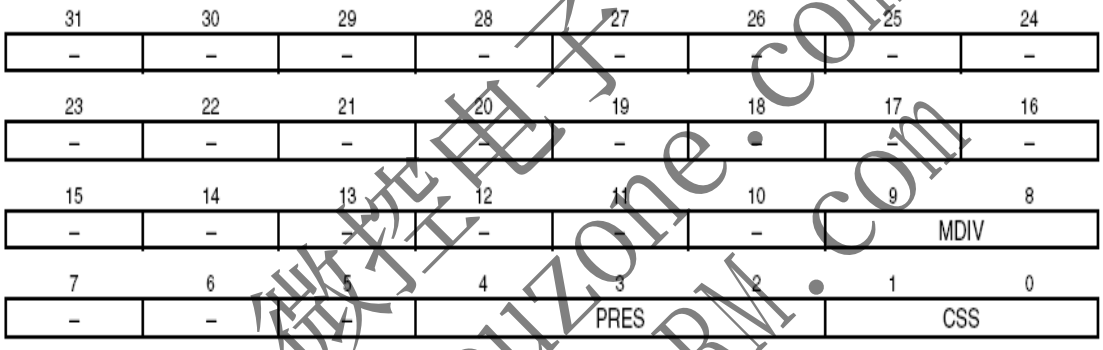
- USBDIV：对USB时钟的分频器

USBDIV		对USB时钟的分频器
0	0	分频器输出是PLL B时钟输出
0	1	分频器输出是PLL B时钟输出的二分之一
1	0	分频器输出是PLL B时钟输出的四分之一
1	1	保留

25.10.11 PMC主控时钟寄存器

寄存器名称：PMC_MCKR

访问类型：读/写



- CSS：主控时钟选择

CSS		时钟源选择
0	0	选择慢时钟
0	1	选择主振荡器时钟
1	0	选择PLL A时钟
1	1	选择PLL B时钟

- PRES：处理器时钟预分频器

PRES			处理器时钟
0	0	0	选择的时钟
0	0	1	选择时钟的二分之一
0	1	0	选择时钟的四分之一
0	1	1	选择时钟的八分之一
1	0	0	选择时钟的十六分之一
1	0	1	选择时钟的三十二分之一
1	1	0	选择时钟的六十四分之一
1	1	1	保留

- MDIV：主控时钟分频

MDIV		主控时钟分频
0	0	主控时钟是处理器时钟

微控电子 www.mcuzone.com www.ATARM.com

0	1	主控时钟是处理器时钟的二分之一
1	0	主控时钟是处理器时钟的四分之一
1	1	保留

25.10.12 PMC可编程时钟寄存器

寄存器名称：PMC_PCKx

访问类型：读/写

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

- CSS：可编程时钟源选择

CSS		时钟源选择
0	0	选择慢时钟
0	1	选择主时钟
1	0	选择PLL A时钟
1	1	选择PLL B时钟

- PRES：可编程时钟预分频器

PRES			可编程时钟
0	0	0	选择的时钟
0	0	1	选择时钟的二分之一
0	1	0	选择时钟的四分之一
0	1	1	选择时钟的八分之一
1	0	0	选择时钟的十六分之一
1	0	1	选择时钟的三十二分之一
1	1	0	选择时钟的六十四分之一
1	1	1	保留

25.10.13 PMC中断使能寄存器

寄存器名称：PMC_IER

访问类型：只写

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	PCKRDY3	PCKRDY2	PCKRDY1	PCKRDY0
7	6	5	4	3	2	1	0
–	–	–	–	MCKRDY	LOCKB	LOCKA	MOSCS

- MOSCS：主振荡器状态中断使能
- LOCKA：PLL A时钟中断使能

微控电子 www.mcuzone.com www.ATARM.com

- LOCKB：PLL B时钟中断使能
- MCKRDY：主控时钟就绪中断使能
- PCKRDYx：可编程时钟就绪x中断使能

0=无效

1=使能对应的中断

25.10.14 PMC中断禁用寄存器

寄存器名称：PMC_IDR

访问类型：只写

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	PCKRDY3	PCKRDY2	PCKRDY1	PCKRDY0
7	6	5	4	3	2	1	0
–	–	–	–	MCKRDY	LOCKB	LOCKA	MOSCS

- MOSCS：主振荡器状态中断禁用
- LOCKA：PLL A时钟中断使能
- LOCKB：PLL B时钟中断使能
- MCKRDY：主控时钟就绪中断禁用
- PCKRDYx：可编程时钟就绪x中断禁用

0=无效

1=禁用对应的中断

25.10.15 PMC状态寄存器

寄存器名称：PMC_SR

访问类型：只读

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	PCKRDY3	PCKRDY2	PCKRDY1	PCKRDY0
7	6	5	4	3	2	1	0
–	–	–	–	MCKRDY	LOCKB	LOCKA	MOSCS

- MOSCS：MOSCS状态标志位

0=主振荡器未稳定

1=主振荡器已稳定

- LOCKA：PLL A锁定状态

0=PLL A未被锁定

1=PLL A被锁定

- LOCKB：PLL A锁定状态

0=PLL B未被锁定

1=PLL B被锁定

- MCKRDY：主控时钟状态

微控电子 www.Mcuzone.com www.ATARM.com

0=主控时钟未就绪

1=主控时钟就绪

- PCKRDYx：可编程时钟就绪状态

0=可编程时钟x未就绪

1=可编程时钟x就绪

25.10.16 PMC中断屏蔽寄存器

寄存器名称：PMC_IMR

访问类型：只读

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	PCKRDY3	PCKRDY2	PCKRDY1	PCKRDY0
7	6	5	4	3	2	1	0
–	–	–	–	MCKRDY	LOCKB	LOCKA	MOSCS

- MOSCS：主振荡器状态中断掩码
- LOCKA：PLL A时钟中断掩码
- LOCKB：PLL B时钟中断掩码
- MCKRDY：主时钟就绪中断掩码
- PCKRDYx：可编程时钟就绪x中断掩码

0=对应中断被使能

1=对应中断被禁用

25.10.17 PLL 电荷泵电流寄存器

寄存器名称：PMC_PLLICPR

访问类型：只写

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

- ICPPLLA：电荷泵电流

必须被设置为1

- ICPPLLB：电荷泵电流

必须被设置为1

微控电子 www.Mcuzone.com www.ATARM.com

26 先进中断控制器(AIC)

26.1 描述

先进中断控制器(AIC)是一个8级优先级，可独立屏蔽屏蔽的，向量中断控制器，可处理多达32个中断源。被设计用来从本质上减小在处理内部和外部中断时的软件和实时系统开销。

AIC驱动ARM处理器的nFIQ（快速中断请求）和nIRQ（标准中断请求）。AIC的输入来自内部外设中断或产品引脚的外部中断。

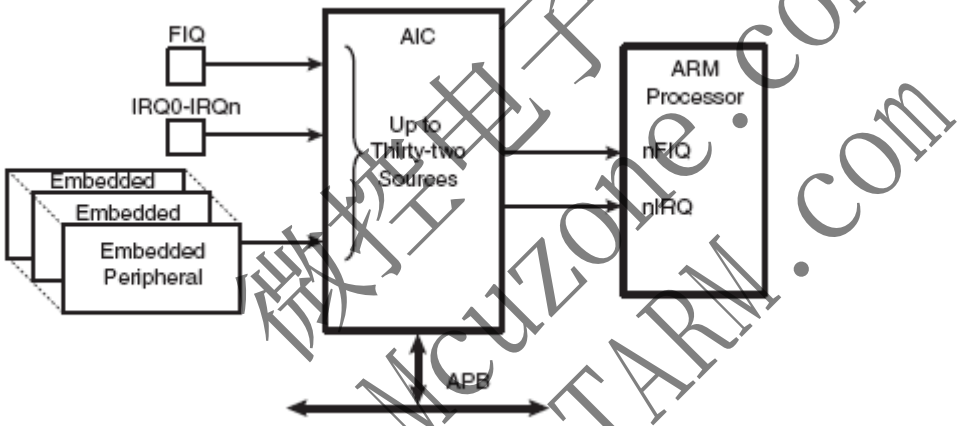
8级优先级控制器允许用户对每个中断源定义优先级。即使一个低优先级中断正在被处理，也允许高优先级中断被服务。

内部中断源可被编程为电平有效或边沿触发。外部中断源可被编程为上升沿或下降沿触发或者高电平或低电平有效。

快速强制特性可重定向任何内部或外部中断源为一个快速中断而不是一个普通中断。

26.2 方块图

图26-1 方块图



26.3 应用方块图

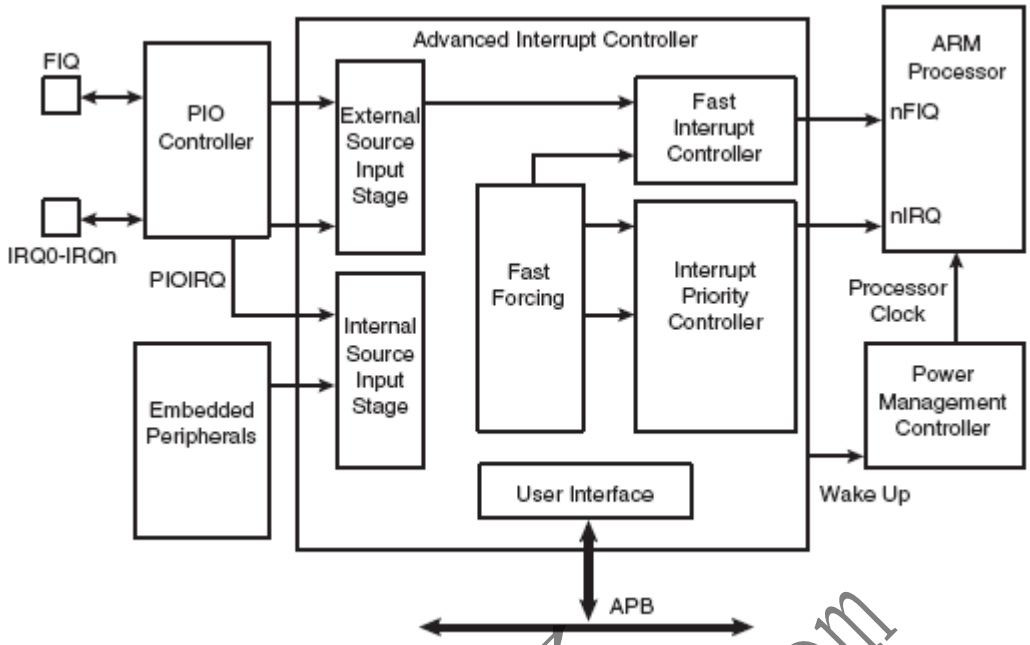
图26-2 应用方块图描述

独立应用	基于OS的应用		
	OS驱动器	RTOS驱动器	硬实时任务
	通用OS中断处理程序		
先进的中断控制器			
嵌入式外设	外部外设（外部中断）		

26.4 详细的AIC方块图

图26-3 详细的AIC方块图

微控电子 www.mcuzone.com www.ATARM.com



26.5 I/O口线描述

表26-1 I/O口线描述

引脚名称	引脚描述	类型
FIQ	快速中断	输入
IRQ0-IRQn	中断0-中断n	输入

26.6 产品相关性

26.6.1 I/O口线

中断信号FIQ和IRQ0到IRQn是通过PIO控制器多路复用的。取决于产品中使用的PIO控制器的特性，必须依照指定的中断功能编程引脚。当产品中使用的PIO控制器在输入路径上是透明的时，这将不适用。

26.6.2 电源管理

先进中断控制器被永久提供时钟。电源管理控制器的行为对先进中断控制器无影响。

先进中断控制器的输出，nIRQ 或 nFIQ，在有效时，可以当ARM处理器在空闲模式时唤醒ARM处理器。通用中断屏蔽特性可使能AIC来唤醒处理器而不用激活处理器的中断口线，由此可使得处理器和特定事件同步。

26.6.3 中断源

中断源0总是被分配给FIQ。如果产品没有FIQ引脚，则不能使用中断源0。

中断源1总是被分配给系统中断。这是将系统定时器，实时时钟，电源管理控制器和存储控制器等系统外设中断口线‘线或’的结果。当一个系统中断产生，服务例程必须首先判别中断的原因，可通过连续的读取以上提到的系统外设的状态寄存器来执行。

中断源2到31可以被连接到嵌入式用户外设的中断输出或外部中断引脚。外部中断引脚可被直接的连接，或通过PIO控制器连接。

PIO控制器在中断处理时被认为是用户外设。相应的，PIO控制器中断口线被连接于中断源2到31。

定义在产品等级上的外设标识相当于中断源号（也是控制外设时钟的位号）。

微控电子 www.mcuzone.com www.ATARM.com

所以，为简化功能操作和用户接口的描述，中断源被命名为FIQ, SYS, 和 PID2 到 PID31。

26.7 功能描述

26.7.1 中断源控制

26.7.1.1 中断源模式

先进中断控制器可独立的编程每个中断源。相应AIC_SMR（中断源模式寄存器）中的SRC-TYPE域选择每个中断源的中断条件。

连接在嵌入式外设的中断输出上的内部中断源可被编程为电平有效模式或边沿触发模式。内部中断的有效电平对用户并不重要。

外部中断源可被编程为高电平有效或低电平有效模式，或者上升沿触发或下降沿触发模式。

26.7.1.2 中断源使能

每个中断源，包括中断源0中的FIQ，可通过使用命令寄存器：AIC_IECR（中断使能命令寄存器）和AIC_IDCR（中断禁用命令寄存器）被使能或被禁用。此套寄存器使得可以使用一个指令来使能和禁用中断。中断屏蔽状态可在AIC_IMR寄存器中被读取。一个被禁用的中断不影响其它中断的服务。

26.7.1.3 中断清零和置位

所有编程为边沿触发（包括中断源0中的FIQ）的中断源可通过写AIC_ISCR和 AIC_ICCR寄存器中的相应的位来分别置位或清零。清零或置位对编程为电平有效模式的中断源无影响。

清零操作是很基本的，当中断源被编程边沿触发模式时软件必须执行一个操作来重新初始化“记忆”电路。然而，置位操作对自动测试或软件调试目的是有用的。还可被用于实现一个AIC实现的软件中断。

AIC在AIC_IVR（中断向量寄存器）被读取时执行一个当前中断的自动清零操作。仅中断源被AIC检测到并作为当前中断时其才会被此操作影响（见285页“优先级控制器”）。自动清零减少了中断服务例程入口代码读AIC_IVR的操作。注意如果中断源的快速强制特性被使能，自动中断清零会被禁用，因为其被认为像一个FIQ。（更多细节，见289页“快速强制”）。

26.7.1.4 中断状态

对每个中断，AIC操作发自AIC_IPR（中断待定寄存器）和其AIC_IMR（中断屏蔽寄存器）中的屏蔽码。AIC_IPR反映中断源的实际的活动情况，而无论其被使能与否。

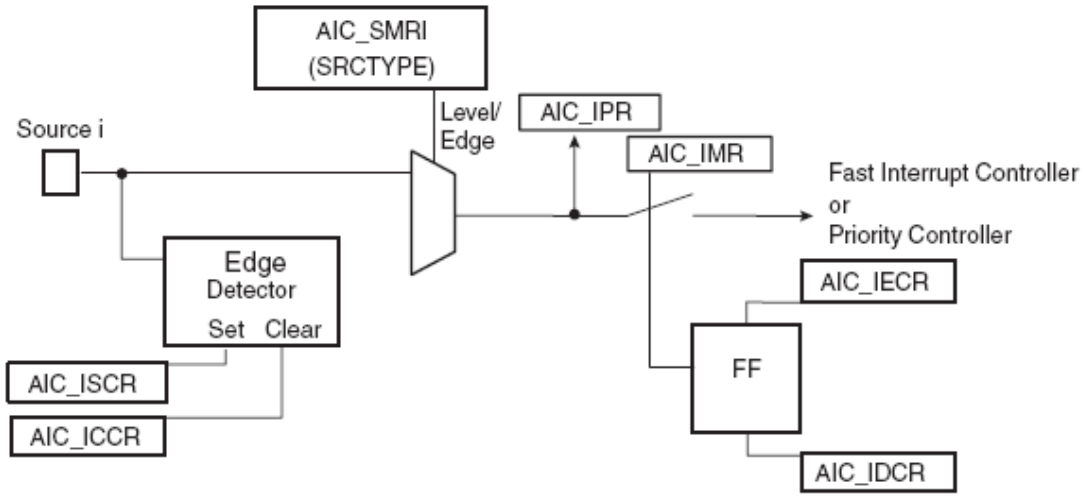
AIC_ISR寄存器读取当前中断源号码（见285页“优先级控制器”）并且寄存器AIC_CISR给出了一个驱动处理器上nIRQ 和 nFIQ信号的映像。

以上提到的每种状态可被用于优化系统的中断处理。

26.7.1.5 内部中断源输入

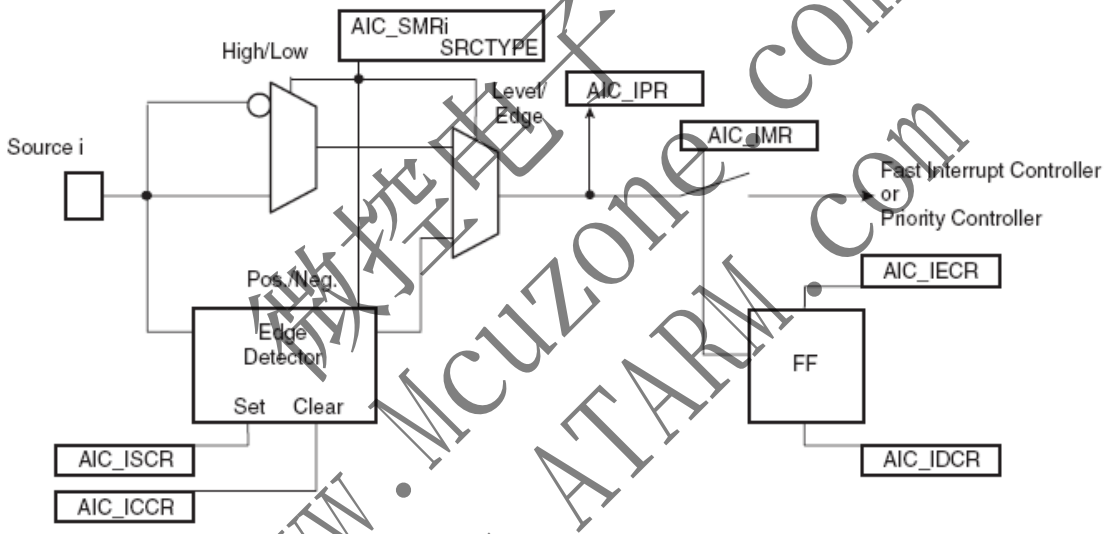
图26-4 内部终端源输入

微控电子 www.mcuzone.com www.ATARM.com



26.7.1.6 外部中断源输入

图26-5 外部中断源输入



26.7.2 中断延迟

全局中断延迟取决于若干参数，包括：

1. 软件屏蔽中断的时间
2. 处理器级或AIC级的事件
3. 当中断发生时，过程中的指令执行时间
4. 高优先级中断的处理和硬件信号的重新同步

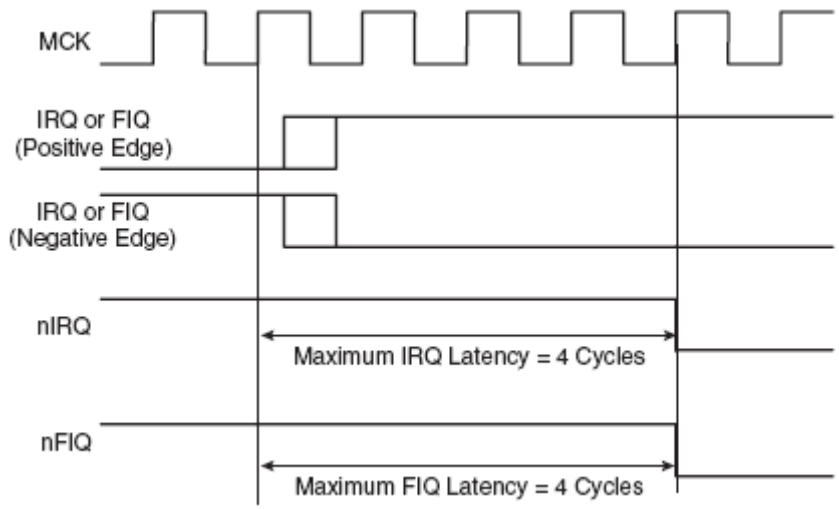
此段仅陈述硬件重新同步。它给出了一个外部中断造成一个有效中断（边沿或电平）或内部中断源的激活和处理器上nIRQ 或 nFIQ口线被激活之间的延迟时间的细节。重同步时间取决于中断源的编程和其类型（内部或外部）。对于标准中断，在假设进程中无更高优先级的情况下给出重同步时间。

PIO控制器多路复用对外部中断源的中断延迟时间无影响。

26.7.2.1 外部中断边沿触发源

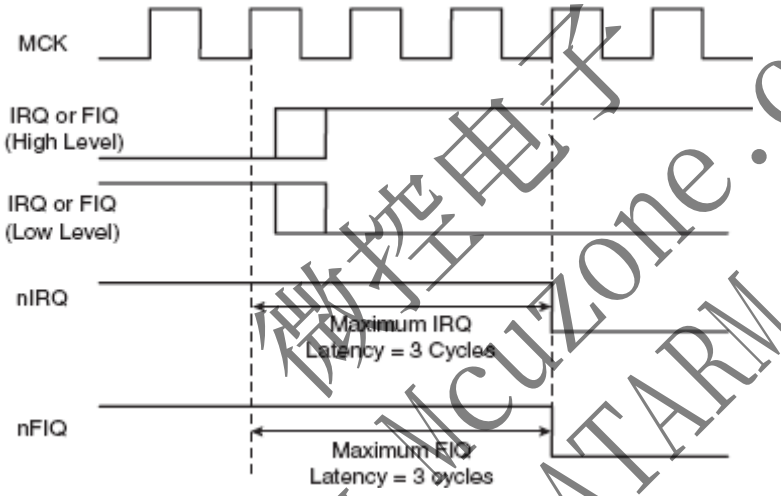
图26-2 外部中断边沿触发源

微控电子 www.Mcuzone.com www.ATARM.com



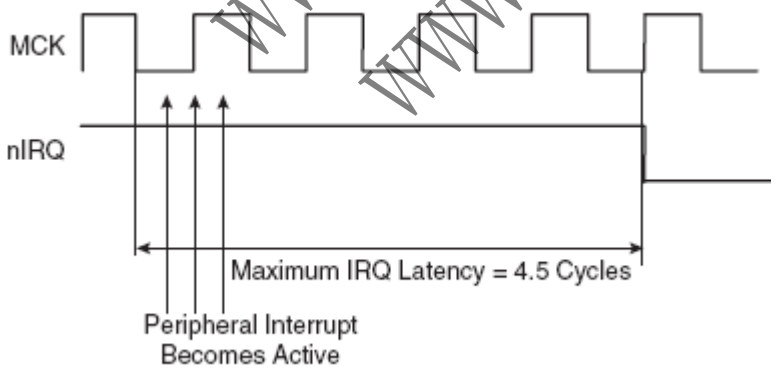
26.7.2.2 外部中断电平敏感中断源

图26-7 外部中断电平敏感中断源



26.7.2.3 内部中断边沿触发源

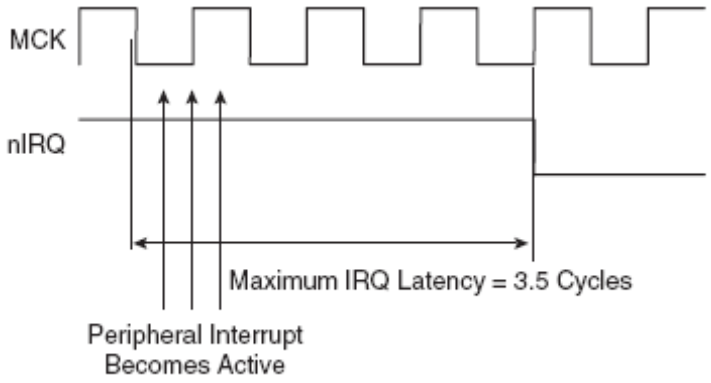
图26-8 内部中断边沿触发源



26.7.2.4 内部中断电平敏感中断源

图26-9 内部中断电平敏感中断源

微控电子 www.mcuzone.com www.ATARM.com



26.7.3 一般中断

26.7.3.1 优先级控制器

8级优先级控制器驱动处理器的nIRQ口线，取决于中断源1到中断源31（除在快速强制中被编程了的）上产生的中断条件。

每个中断源有一个从7到0级的可编程优先级，并可通过写对应的AIC_SMR（中断源模式寄存器）的PRIOR域由用户定义。第7级是最高级，第0级是最低级。

只要一个中断条件产生，此条件由AIC_SMR（中断源模式寄存器）中的SRCTYPE域定义，nIRQ口线被激活。从nIRQ被激活开始，由于一个新中断条件可能在其它中断源上发生，优先级控制器在读取AIC_IVR（中断向量寄存器）时决定当前需要处理的中断。AIC_IVR的读取是中断处理的入口点，此入口点使得AIC认为有软件开始来处理中断。

当前优先级被定义为当前中断的优先级。

如果同等优先级的若干中断源被使能且都处于中断状态，当AIC_IVR被读取时，具有最低中断源号码的中断首先被服务。

nIRQ口线在一个高优先级的中断源上产生中断条件时可被再次激活。如果新的非高优先级中断条件在中断处理期间产生(或等待)，直到软件通过写AIC_EOICR（中断处理结束寄存器）指示AIC当前服务结束才被处理。写AIC_EOICR是中断处理的出口点。

26.7.3.2 中断嵌套

优先级控制器利用中断嵌套使高优先级中断在低优先级中断服务期间被处理。这需要低优先级中断的中断服务例程在处理器级别重新使能中断。

当在处理一个中断服务例程期间发生一个高优先级中断，则nIRQ口线被重新有效。如果中断在内核级别被使能，当前中断服务被中断并且新的中断服务例程将读AIC_IVR。此时，当前中断号和其优先级被放入一个嵌入式的硬件堆栈，因此当高优先级中断服务被完成并且AIC_EOICR被写时他们被取回以完成被中断的处理。

AIC有一个8级深度的硬件堆栈以按照8个优先级来支持多达8级中断嵌套。

26.7.3.3 向量化中断

相当于每个中断源的中断处理程序的地址可被存储在寄存器AIC_SVR1到AIC_SVR31（源向量寄存器1到31）。当处理器读AIC_IVR（中断向量寄存

器），其返回当前中断被写入到对应的AIC_SVR的值。

此功能提供了用单条指令转移到中断处理程序的一个方法，因为AIC_IVR被映射在绝对地址0xFFFF F100处，因此可以在ARM中断向量地址0x0000 0018处放置如下指令：

```
LDR PC,[PC,# -&F20] （译者注：0x18 + 0x08 – 0xF20 = 0xFFFFF100）
```

当处理器执行此指令，则将读取AIC_IVR并装载其值到程序计数器，以此方式跳转到当前中断处理程序上执行操作。

当应用程序是基于操作系统（实时或非实时）时此功能通常未被使用。操作系统对所有中断通常有一个单一入口点并且首先执行的任务是识别中断源。然而，强烈推荐移植操作系统到AT91产品时支持向量中断。这可以通过定义中断源的所有AIC_SVR为操作系统的中断处理地址来实现。当执行此操作时，向量中断允许一个关键中断转换到处理特定的非常快速的处理程序而不是操作系统的通用中断处理程序。这使得硬实时任务（声音/音频缓冲和软件外设处理的输入/输出）的支持更容易，运行在操作系统下的应用程序被有效和独立的处理。

26.7.3.4 中断处理程序

此段大致给出了当使用AIC时快速中断处理顺序。假设编程者了解ARM处理器的架构，特别是处理器的中断模式和相关的状态位。

假设：

1. 先进中断控制器已被编程，AIC_SVR寄存器被用对应中断服务例程地址装载并且中断被使能。
2. ARM中断异常向量地址处的指令需要支持向量中断

```
LDR PC, [PC, # -&F20]
```

当nIRQ被激活，如果CPSR的位“I”是0，则顺序如下：

1. CPRS被存储在SPSR_irq中，编程计数器的当前值被装载入中断模式的连接寄存器(R14_irq)并且程序计数器被用0x18装载。在一下周期在地址0x1C 取指期间，ARM内核调节R14_irq，减4。
2. ARM内核将进入中断模式，如果还未进入。
3. 当装载在地址0x18处的指令被执行，程序计数器被用从AIC_IVR中读取的值装载。读取AIC_IVR有以下效果：
   -设置当前中断为等待并使能的中断，并有最高级优先级。当前优先级是当前中断的优先级。
   -释放处理器上的nIRQ口线。即使向量化未被使用，AIC_IVR也必须被读取以释放nIRQ。
   -自动清零中断，如果其已被编程为边沿触发。
   -将当前优先级和当前中断号压入堆栈。
   -返回写入AIC_SVR中对应当前中断的值。
4. 前一步骤的效果是转移到对应中断服务例程。应该首先保存连接寄存器(R14_irq)和SPSR_IRQ。如果连接寄存器在中断处理结束时被直接加载到进程序计数器，连接寄存器被保存时必须减4。例如，指令`SUB PC, LR, #4`可被使用
5. 更多中断可通过清零CPSR中的“I”位被使能，允许重激活nIRQ以被内核处理。如果一个优先级比当前中断优先级高的中断产生，此操作会发生。

微控电子 www.mcuzone.com www.ATARM.com

6. 中断处理程序随需要可接着进行，保存将被使用的寄存器并在结束时恢复。此阶段中，若发生比当前优先级高的中断，将从第1步执行。

注意：如果中断被编程为电平敏感，中断源必须在此阶段被清零。

7. CPSR中的“I”位必须在中断返回前被置位，以确保中断被正确完成。
8. 中断结束命令寄存器(AIC_EOICR)必须被写以指示AIC当前中断已完成。这导致当前级从堆栈被弹出，如果栈有前一个中断级，其将被恢复。如果另一个中断处于等待状态，其优先级低于或等于前面的当前级，但其优先级比新的当前级高，nIRQ口线被重新激活，但中断处理未立即启动因为“I”位在内核中被置位。SPSR_irq被恢复。最终，连接寄存器中的保存值被直接存储到PC。这使得中断返回到前面的操作，用存储的SPSR装载CPSR，使能或禁用中断的操作，取决于保存在SPSR_irq中的状态。

注意：SPSR中的“I”位是有意义的。如果被置位，它将指示当ARM处理器执行一个中断屏蔽指令时，该指令被中断。因此，当SPSR被恢复，禁用指令已完成（中断被禁用）。

26.7.4 快速中断

26.7.4.1 快速中断源

中断源0是唯一可引起一个快速中断请求到处理器中断源，除非快速强制被使用。中断源0一般连接于产品的一个FIQ引脚，直接连接或通过一个PIO控制器。

26.7.4.2 快速中断控制

AIC的快速中断逻辑无优先级控制器。中断源0的模式用AIC_SMR0并且此寄存器未使用 PRIOR 域，即使其读取为被写的数据。AIC_SMR0的SRCTYPE域使能快速中断源被编程为正边沿触发或负边沿触发，高电平敏感或低电平敏感。

写0x1在AIC_IECR（中断使能命令寄存器）和AIC_IDCR（中断禁用命令寄存器）中使能和禁用快速中断。AIC_IMR（中断屏蔽寄存器）的0位指示快速中断是否被使能或禁用。

26.7.4.3 快速中断向量化

快速中断处理程序地址可被存储在AIC_SVR0（源向量寄存器0）中。

写入此寄存器的值在处理器读取AIC_FVR（快速向量寄存器）时被返回。

这提供了用单一指令转移到快速中断处理程序的一个方法，因为AIC_FVR被映射在绝对地址0xFFFF F104处，因此可通过将以下指令放置在ARM中断向量地址0x0000 001C处来实现：

```
LDR PC,[PC,# -&F20]
```

当处理器执行此指令，装载从AIC_IVR中读取的值到程序计数器，以此方式在跳转到快速中断处理程序执行。还将自动执行快速中断源的清零，如果其被编程为边沿触发模式。

26.7.4.4 快速中断处理程序

此段大致给出了当使用AIC时，快速中断处理顺序。假设编程者了解ARM处理器的架构，特别是处理器中断模式和相关状态位。

假设：

1. 先进的中断控制器已被编程，AIC_SVR0被用快速中断服务例程地址装载，中断源0被使能。
2. 地址0x1C（FIQ异常向量地址）处的指令需要满足向量中断要求：

微控电子 www.mcuzone.com WWW.ATARM.com

```
LDR PC, [PC, # -&F20]
```

3. 用户不需要快速中断嵌套。

当nFIQ被激活时，如果CPSR的“F”位是0，则顺序是：

1. CPSR被存储在SPSR_fiq中，程序计数器的当前值被装载入FIQ连接寄存器(R14_FIQ)中并且程序计数器（R15）被用0x1C装载。在一下周期，从地址0x20处取指期间，ARM内核调节R14_fiq，减4。
2. ARM内核进入FIQ模式。
3. 当装载在地址0x1C的指令被执行，程序计数器被用在AIC_FVR中读取的值装载。读取AIC_FVR可以自动清零快速中断，如果其已被编程为边沿触发。仅在此情况下，释放处理器上的nFIQ口线。
4. 前一步骤跳转到对应中断服务例程。如果不需要快速中断嵌套则不必保存连接寄存器R14_fiq和SPSR_fiq。
5. 中断处理程序按需要接着进行。不必保存寄存器R8 到 R13，因为FIQ模式有其自己的专用寄存器并且用户R8 到 R13是单独的。其他寄存器，R0 到 R7，必须在使用前被保存，并在结束时（下一步骤前）恢复。注意如果快速中断被编程为电平敏感，中断源必须在此阶段被清零以释放中断源0。
6. 最后，连接寄存器R14_fiq减4后（例如，用指令SUB PC, LR, #4）被恢复到PC。这使得从FIQ中断返回到到被中断的代码，用SPSR装载CPSR，屏蔽或使能快速中断的任何操作，取决于SPSR中保存的状态。

注意：SPSR中的“F”位是有意义的。如果其被置位，则指示ARM内核在屏蔽FIQ中断时，该指令被中断。因此当SPSR被恢复，中断屏蔽指令被完成（FIQ被屏蔽）。

另一种处理快速中断的方法是映射中断服务例程在ARM向量地址0x1C处。此方法不使用向量中断，因此读取AIC_FVR必须在处理程序操作的刚开始处被执行。然而，此方法节省一个转移指令的执行时间。

26.7.4.5 快速强制

先进中断控制器的快速强制功能提供重定向任何一般中断源到快速中断控制器上。

快速强制通过写快速强制使能寄存器(AIC_FFER)和快速强制禁用寄存器(AIC_FFDR)被使能或禁用。写这些寄存器导致快速强制状态寄存器(AIC_FFSR)的更新，此寄存器为每个内部和外部中断源控制此功能。

当快速中断强制被禁用，中断源如前页所描述的那样被处理。

当快速中断被使能，边沿/电平编程，在某特定情况下，中断源的边沿检测有效但中断源不能触发一个一般的中断到处理器并且不能通被优先级处理硬件处理。

如果中断源被编程为电平敏感模式并且有效电平被采样到，快速强制导致内核的nFIQ口线被激活。

如果中断源被编程为边沿触发模式并且有效边沿被检测到，快速强制导致内核的nFIQ口线被激活。

快速中断功能不影响中断源0在中断等待寄存器(AIC_IPR)中的等待位。

FIQ向量寄存器(AIC_FVR)读取源向量寄存器0 (AIC_SVR0)的内容，无论快速中断源可能是什么。当快速强制功能被使用时FVR的读取不清零中断源0，清零中断源应该通过写中断清零命令寄存器(AIC_ICCR)来实现。

所有使能了快速强制功能并且被编程为边沿触发模式的处于等待状态的中断

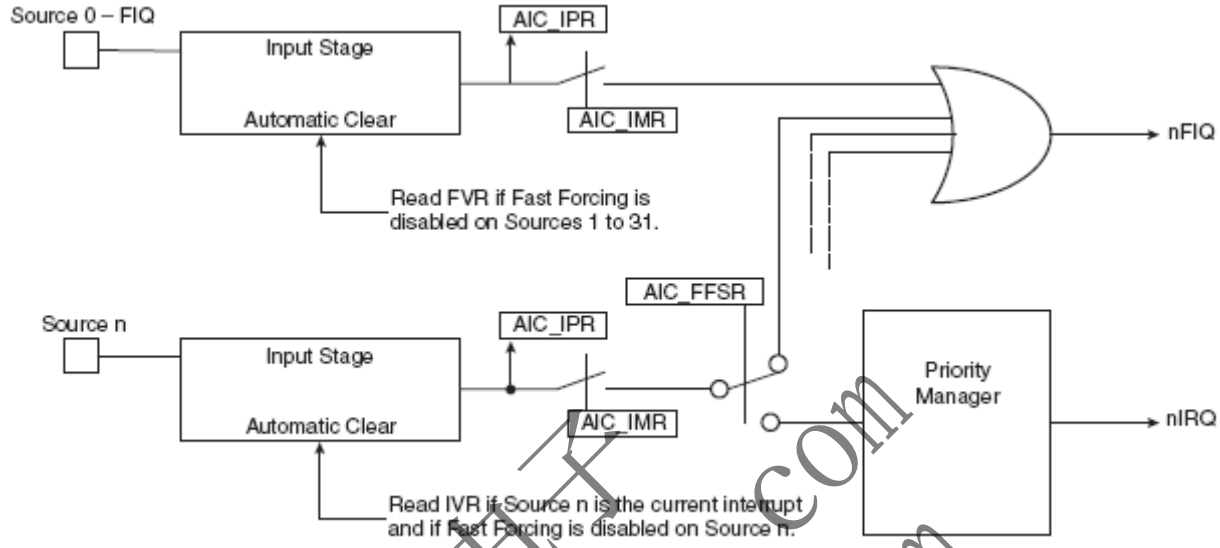
微控电子 www.MCUZone.com www.ATARM.com

源必须通过写中断清零命令寄存器被清零。这样做，它们将被独立清零并能防止中断丢失。

读取AIC_IVR不清零使能了快速强制功能中断源。

中断源0，被快速中断保留，将继续正常操作并变成快速中断源之一。

图26-10 快速强制



26.7.5 保护模式

保护模式允许读取中断向量寄存器而不执行相关的自动化操作。当调试系统时是有必要的。当一个调试器，与调试监视器或ARM处理器ICE一起运行，在停止应用程序并更新打开的窗口时，可能读取AIC用户接口从而读取IVR。这可能会有不可预期的结果：

1. 如果一个被使能的中断处于等待状态，且其优先级比当前高，将被放入堆栈。
2. 如果没有使能的中断处于等待状态，伪中断向量被返回。

任何情况下，中断命令的结束需要确认恢复AIC的上下文。此操作通常不被调试系统执行，因为调试系统的强侵入性的导致应用程序进入非期望的状态。这可以通过使用保护模式避免。写AIC_DCR（调试控制寄存器）中的DBGM为0x1使能保护模式。

当保护模式被使能，仅当写访问在AIC_IVR上被执行时AIC执行中断的堆栈操作。因此，中断服务例程必须在读取后写（任意数据）到AIC_IVR。AIC的新的上下文，包括中断状态寄存器(AIC_ISR)的值，仅当AIC_IVR被写时用当前中断更新。

读取AIC_IVR自身（例如，通过一个调试器），AIC上下文和AIC_ISR两者都不修改。额外的AIC_IVR读取执行同样的操作。然而，推荐不要在中断服务例程的AIC_IVR的读和写之间停止处理器，以确保调试器不修改AIC上下文。

总之，在正常操作模式下，AIC_IVR的读取在AIC中执行以下操作：

1. 计算有效的中断（比当前或伪中断优先级高）
2. 确定并返回有效中断的向量
3. 记录中断
4. 把当前优先级压入内部堆栈
5. 相应中断

微控电子 WWW.MCUZONE.COM WWW.ATARM.COM

然而，在保护模式被激活时，当AIC_IVR被读取时仅执行操作1到3。操作4和5 仅在AIC_IVR被写时被AIC执行。

使用保护模式编写和调试的软件无需修改即可正确的运行在正常模式。然而，正常模式下AIC_IVR写并无效果，因此可被删除以优化代码。

26.7.6 伪中断

先进中断控制器有保护功能以处理伪中断。伪中断被定义为中断源激活足够长的时间，使得AIC激活了nIRQ，但当AIC_IVR被读取时中断源却不再存在。这在下列情况下更容易发生：

1. 外部中断源被编程为电平敏感模式并且有效电平仅持续一个短的时间。
2. 内部中断源被编程为电平敏感模式并且对应的嵌入式外设的输出信号仅持续一个短的时间。（如看门狗的情况）
3. 软件开始屏蔽中断前几个周期产生中断，因此导致中断源上的一个脉冲 。

当无使能的中断源等待，AIC在AIC_IVR被读取时检测到伪中断。当检测到伪中断，AIC返回被编程者存储在AIC_SPU（伪向量寄存器）中的值。编程者必须将伪中断处理程序的地址存储在AIC_SPU中作为应用程序的一部分，使得尽可能快地返回正常执行流程。此处理程序写入AIC_EOICR并执行一个中断返回。

26.7.7 通用中断屏蔽

AIC有一通用中断屏蔽位来阻止中断到达处理器的中断引脚。如果AIC_DCR（调试控制寄存器）中的GMSK位被置位，nIRQ 和 nFIQ两者都被驱动为其非活动状态。然而，如果处理器进入空闲模式，此屏蔽不阻止唤醒处理器。此功能使在下一次事件上同步化处理器变得容易，只要事件发生，不必处理中断就可以执行随后的操作。强烈建议谨慎使用此屏蔽位。

26.8 先进的中断控制器(AIC)用户接口

26.8.1 基地址

AIC被映射在地址0xFFFF F000。共有4K字节的地址空间。允许向量化功能，因为ARM处理器的PC相对装载/存储指令仅支持± 4K字节的偏移量。

26.8.2 寄存器映射

表26-2 寄存器映射

偏移量	寄存器	名称	访问	复位值
0000	源模式寄存器0	AIC_SMR0	读/写	0x0
0x04	源模式寄存器1	AIC_SMR1	读/写	0x0
---	---	---	---	---
0x7C	源模式寄存器31	AIC_SMR31	读/写	0x0
0x80	源向量寄存器0	AIC_SVR0	读/写	0x0
0x84	源向量寄存器1	AIC_SVR1	读/写	0x0
---	---	---	---	---
0xFC	源向量寄存器31	AIC_SVR31	读/写	0x0
0x100	中断向量寄存器	AIC_IVR	只读	0x0

微控电子 www.mcuzone.com www.ATARM.com

0x104	FIQ中断向量寄存器	AIC_FVR	只读	0x0
0x108	中断状态寄存器	AIC_ISR	只读	0x0
0x10C	中断待定寄存器（2）	AIC_IPR	只读	0x0（1）
0x110	中断屏蔽寄存器（2）	AIC_IMR	只读	0x0
0x114	内核中断状态寄存器	AIC_CISR	只读	0x0
0x118	保留	---	---	---
0x11C	保留	---	---	---
0x120	中断使能命令寄存器（2）	AIC_IECR	只写	---
0x124	中断禁用命令寄存器（2）	AIC_IDCR	只写	---
0x128	中断清零命令寄存器（2）	AIC_ICCR	只写	---
0x12C	中断置位命令寄存器（2）	AIC_ISCR	只写	---
0x130	中断结束命令寄存器	AIC_EOICR	只写	---
0x134	伪中断向量寄存器	AIC_SPU	读/写	0x0
0x138	调试控制寄存器	AIC_DCR	读/写	0x0
0x13C	保留	---	---	---
0x140	快速强制使能寄存器（2）	AIC_FFER	只写	---
0x144	快速强制禁用寄存器（2）	AIC_FFDR	只写	---
0x148	快速强制状态寄存器（2）	AIC_FFSR	只读	0x0

注意：1。此寄存器的复位值取决于外部中断源的电平。所有其它中断源在复位时被清零，因此处于非等待状态。

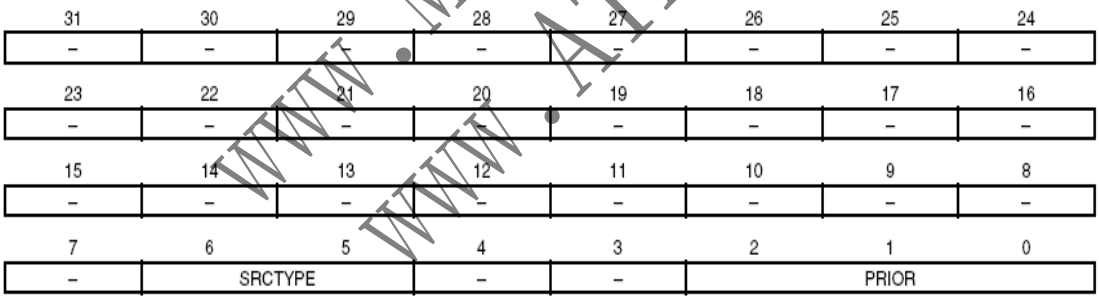
2．PID2...PID31位域提到的标识符如定义在产品数据手册的外设标识符段。

26.8.3 AIC源模式寄存器

寄存器名称：AIC_SMR0..AIC_SMR31

访问类型：读/写

复位值：0x0



- PRIOR：优先级

对除FIQ源（中断源0）的所有源编程优先级。

优先级可以在0（最低）和7（最高）之间。

优先级未用于AMR FIQ中断相关的AIC_SMR寄存器。

- SRCTYPE：中断源类型

有效电平或边沿对内部中断源是不可编程的。

SRCTYPE		内部中断源	外部中断源
0	0	高电平敏感	低电平敏感
0	1	正边沿触发	负边沿触发
1	0	高电平敏感	高电平敏感

微控电子 www.Mcuzone.com www.ATARM.com

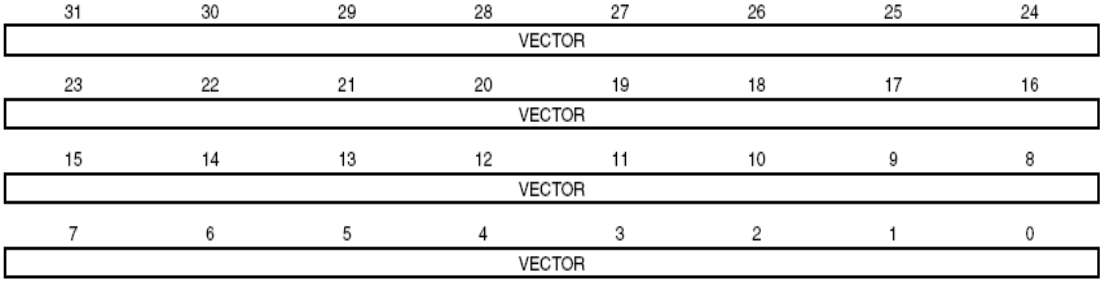
1	1	正边沿触发	正边沿触发

26.8.4 AIC源向量寄存器

寄存器名称：AIC_SVR0..AIC_SVR31

访问类型：读/写

复位值：0x0



- VECTOR：源向量

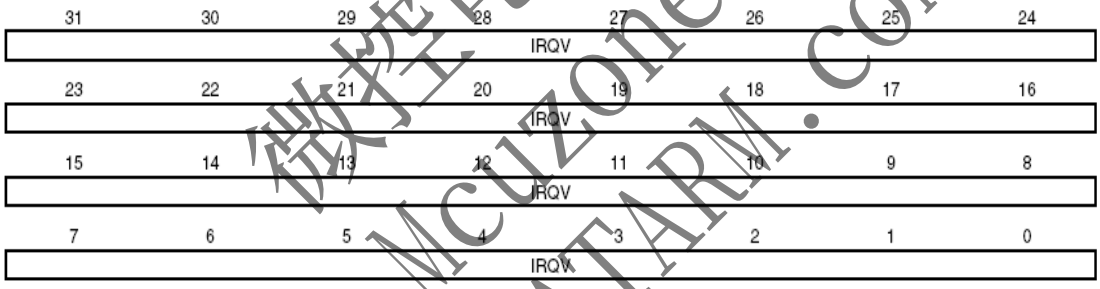
用户可以存储每个中断源的对应处理程序的地址到这些寄存器。

26.8.5 AIC中断向量寄存器

寄存器名称：AIC_IVR

访问类型：只读

复位值：0x0



- IRQV：中断向量寄存器

中断向量寄存器包含当前中断对应的源向量寄存器中编程的向量。

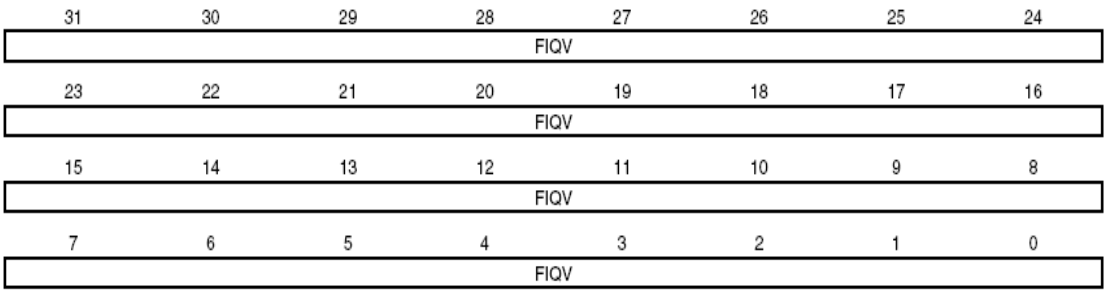
当中断向量寄存器被读取，将使用当前中断号索引源向量寄存器。当无当前中断时，中断向量寄存器读取存储在AIC_SPU中的值。

26.8.6 AIC FIQ 向量寄存器

寄存器名称：AIC_FVR

访问类型：只读

复位值：0x0



- FIQV：FIQ向量寄存器

微控电子 www.mcuzone.com www.ATARM.com

FIQ向量寄存器包含用户在源向量寄存器0中编程的向量。当无快速中断，FIQ向量寄存器读取存储在AIC_SPU中的值。

26.8.7 AIC中断状态寄存器

寄存器名称：AIC_ISR

访问类型：只读

复位值：0x0

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

- IRQID：当前中断标识符

中断状态寄存器返回当前中断源号

26.8.8 AIC中断等待寄存器

寄存器名称：AIC_IPR

访问类型：只读

复位值：0x0

31	30	29	28	27	26	25	24
PID31	PID30	PID29	PID28	PID27	PID26	PID25	PID24
23	22	21	20	19	18	17	16
PID23	PID22	PID21	PID20	PID19	PID18	PID17	PID16
15	14	13	12	11	10	9	8
PID15	PID14	PID13	PID12	PID11	PID10	PID9	PID8
7	6	5	4	3	2	1	0
PID7	PID6	PID5	PID4	PID3	PID2	SYS	FIQ

- FIQ,SYS,PID2-PID31：中断等待状态

0=对应中断未等待

1=对应中断处于等待状态

26.8.9 AIC中断屏蔽寄存器

寄存器名称：AIC_IMR

访问类型：只读

复位值：0x0

31	30	29	28	27	26	25	24
PID31	PID30	PID29	PID28	PID27	PID26	PID25	PID24
23	22	21	20	19	18	17	16
PID23	PID22	PID21	PID20	PID19	PID18	PID17	PID16
15	14	13	12	11	10	9	8
PID15	PID14	PID13	PID12	PID11	PID10	PID9	PID8
7	6	5	4	3	2	1	0
PID7	PID6	PID5	PID4	PID3	PID2	SYS	FIQ

- FIQ,SYS,PID2-PID31：中断屏蔽

微控电子 www.mcuzone.com www.ATARM.com

0=对应中断被禁用

1=对应中断被使能

26.8.10 AIC内核中断状态寄存器

寄存器名称：AIC_CISR

访问类型：只读

复位值：0x0

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

- NFIQ：NFIQ状态

0= nFIQ口线被禁用

1= nFIQ口线被使能

- NIRQ：NIRQ状态

0= nIRQ口线被禁用

1=nIRQ口线被使能

26.8.11 AIC中断使能命令寄存器

寄存器名称：AIC_IECR

访问类型：只写

31	30	29	28	27	26	25	24
PID31	PID30	PID29	PID28	PID27	PID26	PID25	PID24
23	22	21	20	19	18	17	16
PID23	PID22	PID21	PID20	PID19	PID18	PID17	PID16
15	14	13	12	11	10	9	8
PID15	PID14	PID13	PID12	PID11	PID10	PID9	PID8
7	6	5	4	3	2	1	0
PID7	PID6	PID5	PID4	PID3	PID2	SYS	FIQ

- FIQ,SYS,PID2-PID3：中断使能

0=无效

1=使能对应中断

26.8.12 AIC中断禁用命令寄存器

寄存器名称：AIC_IDCR

访问类型：只写

微控电子 www.Mcuzone.com www.ATARM.com

31	30	29	28	27	26	25	24
PID31	PID30	PID29	PID28	PID27	PID26	PID25	PID24
23	22	21	20	19	18	17	16
PID23	PID22	PID21	PID20	PID19	PID18	PID17	PID16
15	14	13	12	11	10	9	8
PID15	PID14	PID13	PID12	PID11	PID10	PID9	PID8
7	6	5	4	3	2	1	0
PID7	PID6	PID5	PID4	PID3	PID2	SYS	FIQ

- FIQ,SYS,PID-PID31：中断禁用

0=无效

1=禁用对应中断

26.8.13 AIC中断清零命令寄存器

寄存器名称：AIC_ICCR

访问类型：只写

31	30	29	28	27	26	25	24
PID31	PID30	PID29	PID28	PID27	PID26	PID25	PID24
23	22	21	20	19	18	17	16
PID23	PID22	PID21	PID20	PID19	PID18	PID17	PID16
15	14	13	12	11	10	9	8
PID15	PID14	PID13	PID12	PID11	PID10	PID9	PID8
7	6	5	4	3	2	1	0
PID7	PID6	PID5	PID4	PID3	PID2	SYS	FIQ

- FIQ,SYS,PID2-PID31：中断清零

0=无效

1=清零对应中断

26.8.14 AIC中断置位命令寄存器

寄存器名称： AIC_ISCR

访问类型：只写

31	30	29	28	27	26	25	24
PID31	PID30	PID29	PID28	PID27	PID26	PID25	PID24
23	22	21	20	19	18	17	16
PID23	PID22	PID21	PID20	PID19	PID18	PID17	PID16
15	14	13	12	11	10	9	8
PID15	PID14	PID13	PID12	PID11	PID10	PID9	PID8
7	6	5	4	3	2	1	0
PID7	PID6	PID5	PID4	PID3	PID2	SYS	FIQ

- FIQ,SYS,PID2-PID31：中断置位

0=无效

1=置位对应中断

26.8.15 AIC中断结束命令寄存器

微控电子 www.mcuzone.com www.ATARM.com

寄存器名称：AIC_EOICR
访问类型：只写

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

被中断例程通过写中断结束命令寄存器指示中断处理已完成
任何值可被写入，因为仅需要写此寄存器来指示中断处理完成。

26.8.16 AIC伪中断向量寄存器
寄存器名称：AIC_SPU
访问类型：读/写
复位值：0x0

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

- SIQV：伪中断向量寄存器

用户可以在此寄存器中存储伪中断处理程序的地址。已写值的通过AIC_IVR返回以处理伪中断，通过AIC_FVR返回以处理伪快速中断。

26.8.17 AIC调试控制寄存器
寄存器名称：AIC_DEBUG
访问类型：读/写
复位值：0x0

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

- PROT：保护模式

0=保护模式被禁用

微控电子 www.mcuzone.com www.ATARM.com

1=保护模式被使能

- GMSK：通用屏蔽

0= nIRQ 和 nFIQ口线被AIC正常的控制

1= nIRQ 和 nFIQ口线被接于非活动状态。

26.8.18 AIC快速强制使能寄存器

寄存器名称：AIC_FFER

访问类型：只写

31	30	29	28	27	26	25	24
PID31	PID30	PID29	PID28	PID27	PID26	PID25	PID24
23	22	21	20	19	18	17	16
PID23	PID22	PID21	PID20	PID19	PID18	PID17	PID16
15	14	13	12	11	10	9	8
PID15	PID14	PID13	PID12	PID11	PID10	PID9	PID8
7	6	5	4	3	2	1	0
PID7	PID6	PID5	PID4	PID3	PID2	SYS	–

- SYS,PID2-PID31：快速强制使能

0=无效

1=使能对应中断上的快速强制功能

26.8.19 AIC快速强制禁用寄存器

寄存器名称：AIC_FFDR

访问类型：只写

31	30	29	28	27	26	25	24
PID31	PID30	PID29	PID28	PID27	PID26	PID25	PID24
23	22	21	20	19	18	17	16
PID23	PID22	PID21	PID20	PID19	PID18	PID17	PID16
15	14	13	12	11	10	9	8
PID15	PID14	PID13	PID12	PID11	PID10	PID9	PID8
7	6	5	4	3	2	1	0
PID7	PID6	PID5	PID4	PID3	PID2	SYS	–

- SYS,PID2-PID31：快速强制禁用

0=无效

1=禁用对应中断上的快速强制功能

26.8.20 AIC快速强制状态寄存器

寄存器名称：AIC_FFSR

访问类型：只读

31	30	29	28	27	26	25	24
PID31	PID30	PID29	PID28	PID27	PID26	PID25	PID24
23	22	21	20	19	18	17	16
PID23	PID22	PID21	PID20	PID19	PID18	PID17	PID16
15	14	13	12	11	10	9	8
PID15	PID14	PID13	PID12	PID11	PID10	PID9	PID8
7	6	5	4	3	2	1	0
PID7	PID6	PID5	PID4	PID3	PID2	SYS	–

微控电子 www.mcuzone.com www.ATARM.com

- SYS,PID2-PID31：快速强制状态

0=对应中断上的快速强制功能被禁用

1=对应中断上的快速强制功能被使能

微控电子
www.Mcuzone.com
www.ATARM.com

27．调试单元（DBGU）

27.1 描述

调试单元提供了一个访问基于 Atmel 所有具备调试功能的 ARM 系统的处理器的单入口点。

调试单元有一个两引脚的 UART，可用于调试，跟踪目标，就地编程方案并且调试监视器通信。此外，和两外设数据控制器关联的通道允许对减小到最小的处理器时间的任务的数据包处理。

调试单元使得由 ARM 处理器的在线仿真器提供的调试通信通道（DCC）对软件可见。这些信号指示 DCC 读和写寄存器的状态并产生一个到 ARM 处理器的中断，使得在中断控制下 DCC 的处理成为可能。

芯片标识符寄存器允许设备和其版本的识别。这些寄存器通知片上存储器的容量和类型，和嵌入式外设系统。

最后，调试单元有一个强制 NTRST 功能，可以使软件能决定是否阻止通过在线仿真器访问系统。这样可以保护存储于 ROM 的代码。

27.2 方块图

图 27-1 调试单元功能方块图

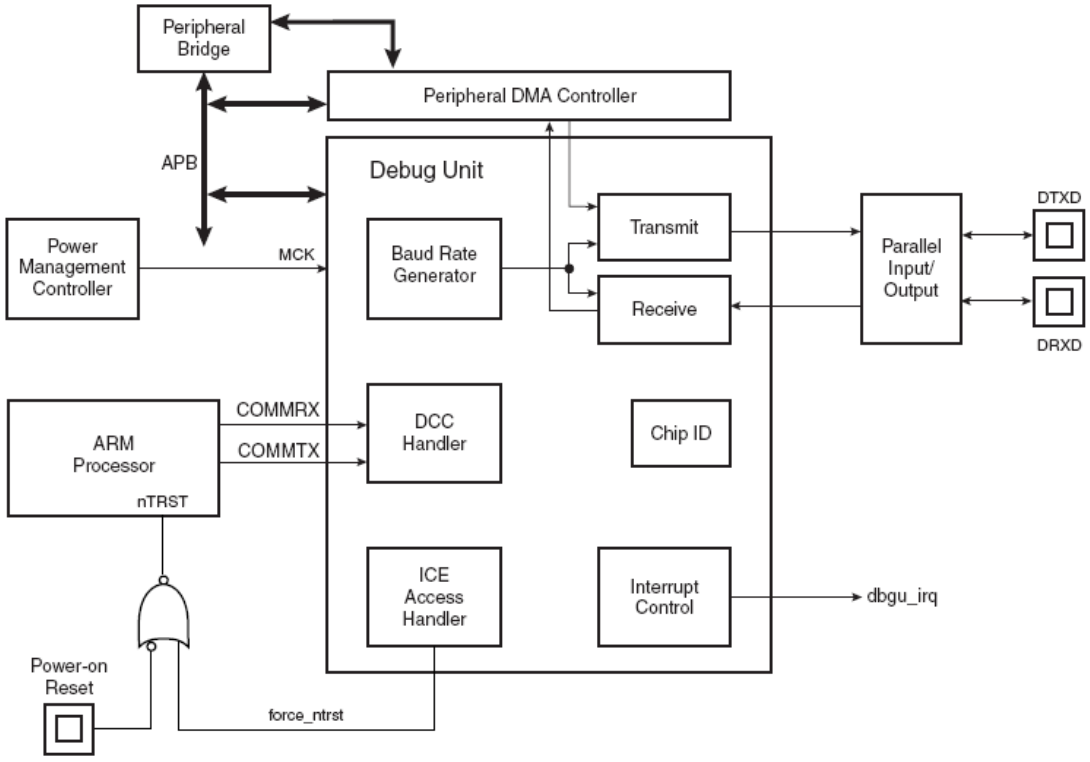
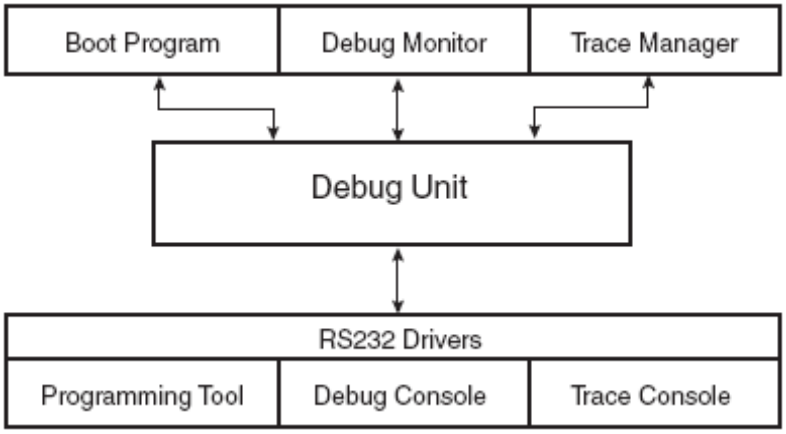


表 27-1 调试单元引脚描述

引脚名称	描述	类型
DRXD	调试接收数据	输入
DTXD	调试发送数据	输出

图 27-2　调试单元应用举例



27.3　产品相关

27.3.1　I/O 口线

取决于产品的集成度，调试单元引脚可以和 PIO 口线分路复用。在此情况下，编程者必须首先配置对应的 PIO 控制器来使能调试单元的 I/O 口线操作。

27.3.2　电源管理

取决于产品的集成度，调试单元时钟可通过电源管理控制器控制。此情况下，编程者必须首先配置 PMC 来使能调试单元时钟。通常，用于此目的外设标识符是 1。

27.3.3　中断源

取决于产品的集成度，调试单元中断口线被连接于高级中断控制器（AIC）的中断源之一。中断处理需要在配制调试单元前编程 AIC。通常，调试单元中断口线连接于 AIC 的中断源 1，此中断源可以和实时时钟，系统定时器中断口线和其他系统外设中断共享，如图 27-1 中所示。此共享需要编程者在中断源 1 被触发时来决定中断源。

27.4　UART 操作

调试单元作为一个 UART 操作，（仅异步模式）并仅支持 8 位字符处理（带奇偶校验）。无时钟引脚。

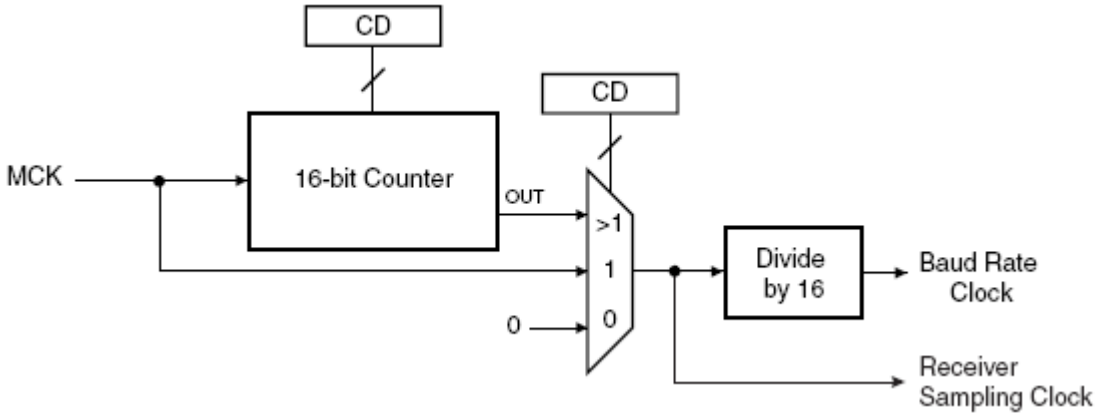
调试单元的 UART 由独立操作的接收器和发送器，和一个通用波特率发生器组成。接收器超时和发送器定时保护未被实现。然而，所有实现了的特性和标准 USART 的兼容。

27.4.1　波特率发生器

波特率发生器提供名为波特率时钟的周期时钟位给接收器和发送器两者。波特率时钟是被 16 分频的主控时钟和被写入 DBGU_BRGR（波特率发生器寄存器）的值相乘的值。如果 DBGU_BRGR 被置为 0，波特率时钟被禁用并且调试单元的 UART 保持待用。最大允许波特率是被 16 分频的主控时钟。最小允许波特率是被(16 x65536)分频的主控时钟。

$$\text{波特率} = \frac{MCK}{16 \times CD}$$

图 27-3　波特率发生器



27.4.2　接收器

27.4.2.1 接收器复位，使能和禁用

设备复位后，调试单元接收器被禁用并且必须在被使用前被禁用。接收器可通过用 RXEN 写控制寄存器 DBGU_CR 为 1 被禁用。在此命令状态，接收器开始寻找起始位。

编程者可通过用 RXDIS 位写 DBGU_CR 为 1 来禁用接收器。如果接收器正等待起始位，将立即停止。然而，如果接收器已检测到起始位并且接收数据，接收器在实际停止其操作前等待停止位。

编程者还可通过用RSTRX位写DBGU_CR为1来将接收器置为其复位状态。这样的话，接收器立即停止其当前操作并被禁用，无论其当前是何状态。如果 RSTRX 当数据被处理时被应用，此数据会丢失。

27.4.2.2 起始位检测和数据采样

调试单元仅支持异步操作，并只影响其接收器。调试单元接收器直到其检测到一个有效的起始位才通过采样 DRXD 信号检测接收到的字符的起始位。如果检测到对应多于 7 个采样时钟周期（16 和波特率的乘积），DRXD 上的低电平（空字符）被解释为一个有效的起始位。因此，空字符比被检测为有效起始位的位周期的 7/16 长。位周期的 7/16 或更短的空字符会被忽略并且接收器继续等待有效的起始位。

当有效起始位被检测到，接收器在每位的理论中点采样 DRXD。假设每位持续采样时钟（1 位周期）的 16 个周期。所以位采样点在起始位后是八周期（0.5 位周期）。首次采样点因此在检测到起始位的下降沿后是 24 个周期。

每个随后的位在前一位后被采样为 16 个周期。

图 27-4　起始位检测

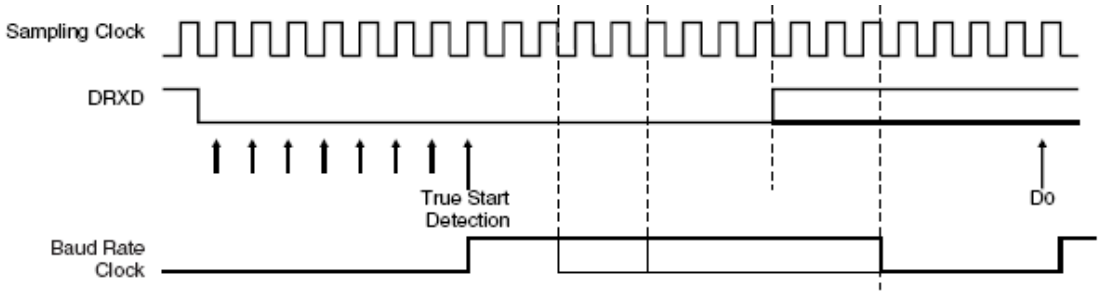
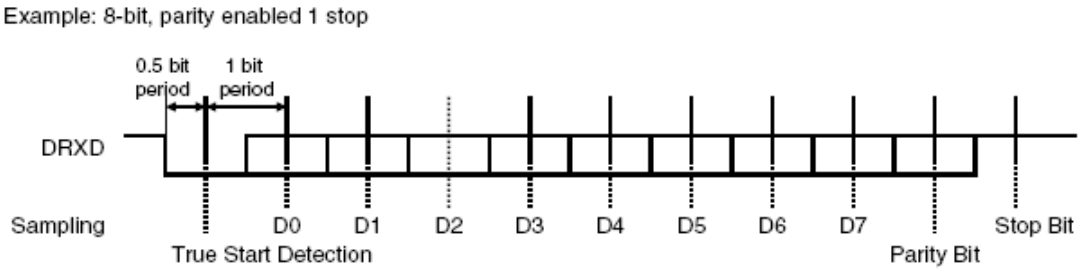


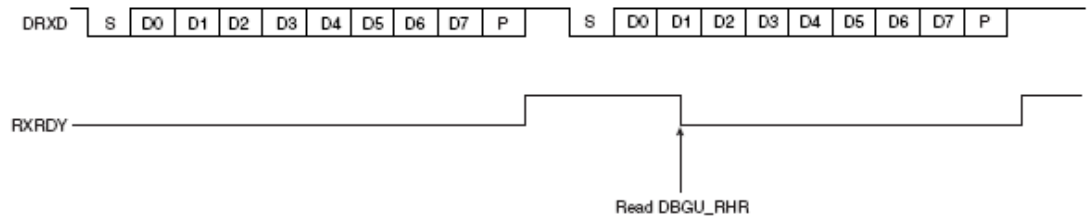
图 27-5　字符接收



27.4.2.3 接收器就绪

当接收到一个完整的字符，将被传送到 DBGU_RHR 和 RXRDY，DBGU_SR（状态寄存器）中的状态位被置位。当读取接收保持寄存器 DBGU_RHR 时 RXRDY 位被自动的清零。

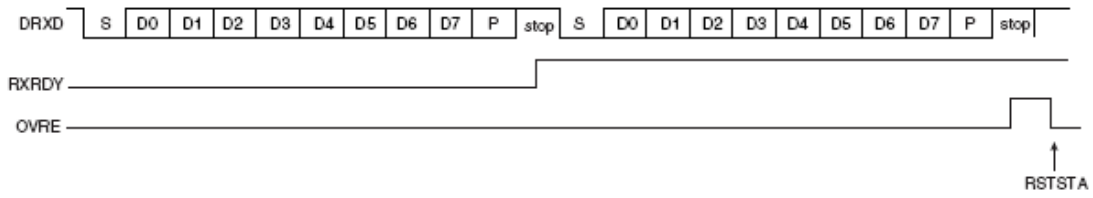
图 27-6 接收器就绪



27.4.2.4 接收器溢出

如果 DBGU_RHR 从最后一次传送为止，未被软件（或外设数据控制器）读取，仍然置位 RXRDY 位并且接收一个新字符，DBGU_SR 中的 OVRE 状态位被置位。当软件用 RSTSTA（复位状态）位写控制寄存器 DBGU_CR 为 1 时，OVRE 被清零。

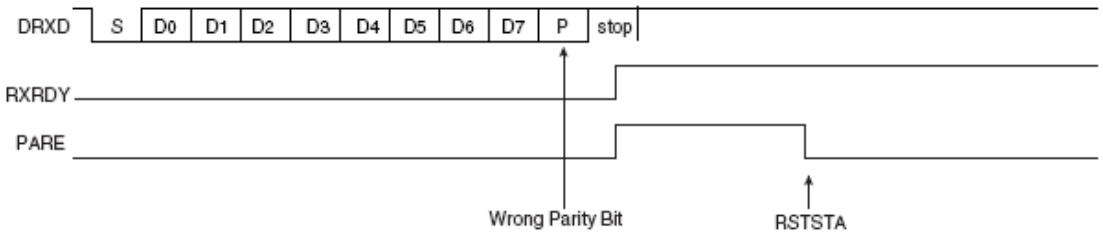
图 27-7　接收器溢出



27.4.2.5 奇偶校验出错

每次接收一个字符，根据 DBGU_MR 终端 PAR 域，接收器计算接收数据位的奇偶校验。然后将结果和接收到的奇偶校验位做比较。如果不同，DBGU_SR 终端奇偶校验出错位 PARE 在 RXRDY 被置位的同时被置位。当控制寄存器 DBGU_CR 被用 RSTSTA（复位状态）位写 1 时奇偶校验位被清零。如果在复位状态命令被写之前接收到一个新字符，PARE 位保持 1。

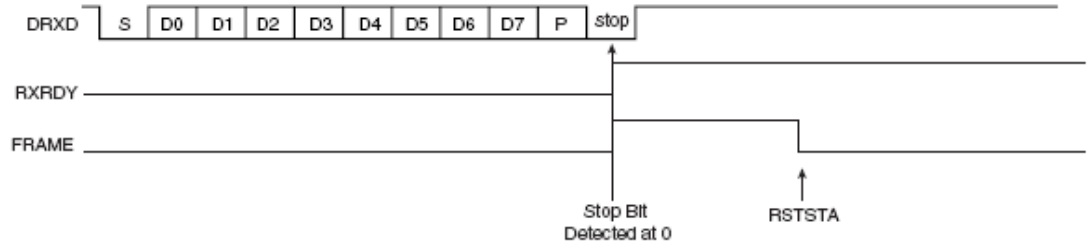
图 27-8　奇偶校验出错



27.4.2.6 接收器帧错误

当检测到起始位，当所有数据位已采样完毕将产生一个字符接收信号。还采样停止位并且当检测到 0 时，DBGU_SR 中的 FRAMW（帧错误）位在 RXRDY 位被置位的同时被置位。FRAME 位保持高电平直到控制寄存器 DBGU_CR 被用 RSTSTA 位写 1 为止。

图 27-9 接收器帧错误



27.4.3 发送器

27.4.3.1 发送器复位，使能和禁用

设备复位后，调试单元发送器被禁用并且必须在被使用前禁用。通过用 TXEN 位写控制寄存器 DBGU_CR 为 1 来禁用发送器。此命令下，发送器实际开始发送前等待字符被写入发送保持寄存器 DBGU_THR。

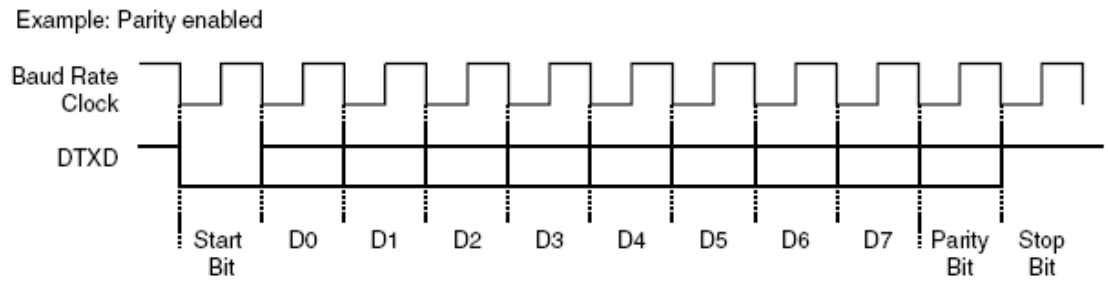
编程者可通过用 TXDIS 位写 DBGU_CR 为 1 来禁用。如果发送器未运行，将立即停止。然而，如果字符被移入移位寄存器和/或字符已被写入发送保持寄存器，字符在发送器实际停止前已完成。

编程者还可通过用 RSTTX 位写 DBGU_CR 为 1 来置发送器为其复位状态。这样可立即停止发送器，无论是否正在移动字符。

27.4.3.2 发送格式

调试单元发送器以波特率的时钟速率驱动 DTXD 引脚。口线驱动取决于在模式寄存器中定义的格式和在移位寄存器中存储的数据。起始位是 0，接着是 8 位数据，从最低位到最高位，一个正确的奇偶校验位和一个停止位 1 如下图所示顺序的移出。模式寄存器 DBGU_MR 中的 PARE 域定义奇偶校验位是否要移出。当奇偶校验位被使能，可在奇校验和偶校验间选择，或在固定空间和标记位间选择。

图 27-10 字符发送

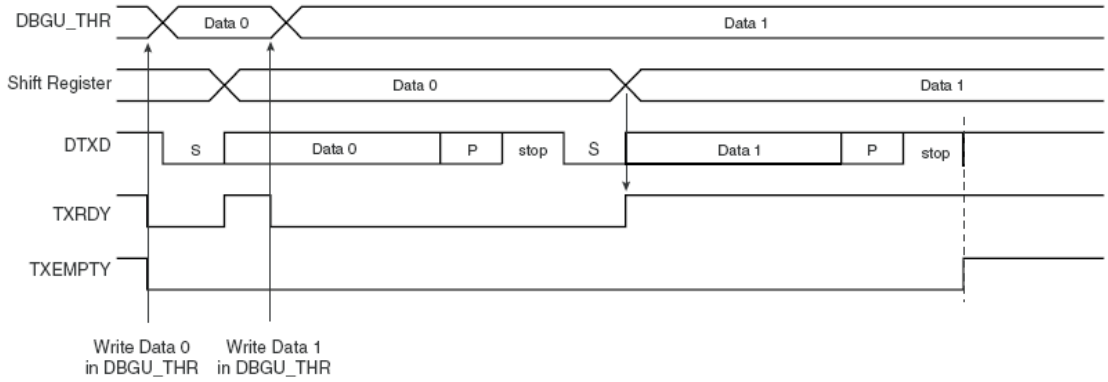


27.4.3.3 发送器控制

当发送器被使能，TXRDY（发送器就绪）位在状态寄存器 DBGU_SR 中被置位。当编程者写入发送保持寄存器 DBGU_THR，并且写入的字符被从 DBGU_THR 传送到移位寄存器时发送开始。TXRDY 位保持高电平直到第二个字符被写入 DBGU_THR。只要首字符已完成，写入 DBGU_THR 中的最后一个字符被传送到移位寄存器并且 TXRDY 再次升为高电平，表示保持寄存器为空。

当移位寄存器和 DBGU_THR 都空时，就是说，所有写入 DBGU_THR 的字符已被移动，TXEMPTY 在最后一个停止位完成后升为高点平。

图 27-11 发送器控制



27.4.4 外设数据控制器

调试单元 UART 的接收器和发送器都通常被连接到外设数据控制器（PDC）通道。

通过从偏移量 0x100 映射在调试单元用户接口的寄存器编程外设数据控制器通道。状态位被报告给调试单元状态寄存器 DBGU_SR 并可产生一个中断。RXRDY 位触发接收器的 PDC 通道数据传送。这导致读取 DBGU_RHR 中的数据。TXRDY 位触发发送器的 PDC 通道数据传送。将导致向 DBGU_THR 中写入数据。

27.4.5 测试模式

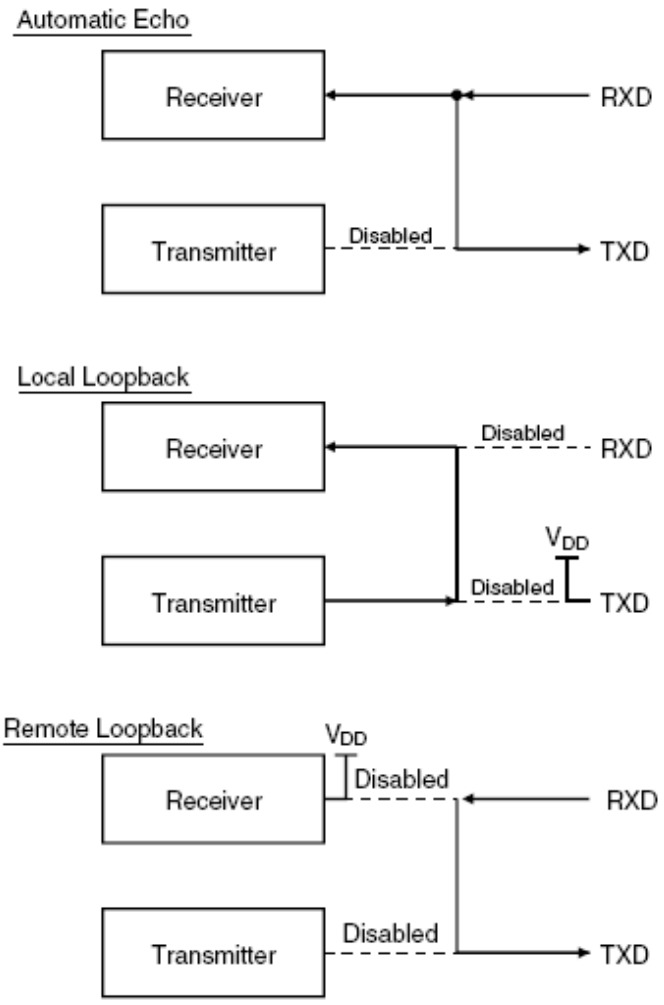
调试单元支持三种测试模式。这些操作模式通过使用模式寄存器中的 CHMODE（通道模式）域来编程。

自动回应模式允许逐位重新发送。当在 DRXD 口线上接收到一位时，此位被送往 DTXD 口线。发送器正常的运行，但是不影响 DTXD 口线。

本地回环模式允许发送接收到的字符。未使用 DTXD 和 DRXD 引脚并且发送器的输出内部的连接于接收器的输入。DRXD 引脚电平无效并且 DTXD 口线保持高电平，如在空闲状态。

远程回环模式直接连接 DRXD 引脚到 DTXD 口线。发送器和接收器被禁用并无效。此模式允许逐位的重新发送。

图 27-12 测试模式



27.4.6 调试通信通道支持

调试单元处理从 ARM 处理起得调试通信通道来的 COMMRX 和 COMMTX 信号，并且由在线仿真器(ICE)驱动。

调试通信通道包括可通过边 JTAG 上的 ICE 断路器寄存器和通过 ARM 处理器中协处理器 0 访问的两个寄存器。

作为提示，以下指令被用于读写调试通信通道：

```
MRC p14, 0, Rd, c1, c0, 0
```

返回调试通信数据读寄存器进入 Rd

```
MCR p14, 0, Rd, c1, c0, 0
```

写值入 Rd 到调试通信数据写寄存器。

COMMRX 和 COMMTX 位，分别的指示读寄存器已被调试器写入但还未被处理器读取，而写寄存器已被处理器写入但未被调试器读取，位于状态寄存器 DBGU_SR 的最高两位。这两位可产生中断。此特性允许中断下处理运行于目标系统的调试监示器和调试器间的调试链接。

27.4.7 芯片标识符

调试单元有两个芯片别识符寄存器，DBGU_CIDR（芯片 ID 寄存器）和 DBGU_EXID（扩展 ID）。两个寄存器都是只读的硬件固化值。首个寄存器包括以下域：

• EXT –表示使用了扩展标识符寄存器

• NVPTYP 和 NVPSIZ –标识内置非易失性存储器类型及其大小
• ARCH –标识内置外设
• SRAMSIZ –指示内置 SRAM 大小
• EPROC –指示内置 ARM 处理器
• VERSION –给出芯片型号

第二个寄存器是由设备决定的，若 EXT 为 0，则其值为 0。

27.4.8 ICE 访问限制

调试单元允许阻止通过 ARM 处理器的 ICE 接口对系统的访问。通过强制 NTRST (DBGU_FNR)寄存器中的 ICE 接口插入信号 NTRST 信号实现该特性。对 FNTRST (强制 NTRST)位写 1 将阻止 TAP 控制器的任何操作。标准设备上，FNTRST 位复位为 0，因此不会阻止 ICE 访问。

该特性主要用于自定义 ROM 设备给不愿其片上代码可见的用户。

27.5 调试单元用户接口

表 27-2 调试单元存储器映射

偏移量	寄存器	名称	访问类型	复位值
0x0000	控制寄存器	DBGU_CR	只写	-
0x0004	模式寄存器	DBGU_MR	读/写	0x0
0x0008	中断使能寄存器	DBGU_IER	只写	-
0x000C	中断禁用寄存器	DBGU_IDR	只写	-
0x0010	中断屏蔽寄存器	DBGU_IMR	只读	0x0
0x0014	状态寄存器	DBGU_SR	只读	-
0x0018	接收保持寄存器	DBGU_RHR	只读	0x0
0x001C	发送保持寄存器	DBGU_THR	只写	-
0x0020	波特率发生器寄存器	DBGU_BRGR	读/写	0x0
0x0024-0x003C	保留	–	-	-
0x0040	芯片 ID 寄存器	DBGU_CIDR	只读	-
0x0044	芯片 ID 扩展寄存器	DBGU_EXID	只读	-
0x0048	强制 NTRST 寄存器	DBGU_FNR	读/写	0x0
0x004C-0x00FC	保留	-	-	-
0x0100-0x0124	PDC 区	-	-	-

27.5.1 调试单元控制寄存器

名称：DBGU_CR

访问类型：只写

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
–	–	–	–	–	–	–	–
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
–	–	–	–	–	–	–	RSTSTA
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TXDIS	TXEN	RXDIS	RXEN	RSTTX	RSTRX	–	–

- RSTRX：复位接收器

0 ＝无效

1 ＝接收器逻辑复位并禁用。如果正接收字符，则停止接收。

- RSTTX：复位发送器

0=无效

1= 发送器逻辑复位并禁用。如果正在发送，则停止发送。

- RXEN：接收器使能

0=无效

1=如果 RXDIS 是 0，则接收器被使能。

- RXDIS：接收器禁用

0 =无效

1 =接收器禁用。如果正在处理字符且 RSTRX 未置位，则在接收器停止前完成字符处理 。

- TXEN：发送器使能

0 =无效

1 =如果 TXDIS 是 0，则发送器被使能

- TXDIS：发送器禁用

0 =无效

1 =发送器禁用。如果正在处理一个字符，另一个字符已被写入 DBGU_THR 并且 RSTTX 未置位，两个字符处理在发送器停止前都要完成。

- RSTSTA：复位状态位

0 =无效

1 =复位 DBGU_SR 中的状态位 PARE, FRAME 和 OVRE。

27.5.2 调试单元模式寄存器

名称：DBGU_MR

访问类型：读/写

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

- PAR：奇偶校验类型

PAR			奇偶校验类型
0	0	0	偶校验
0	0	1	奇校验
0	1	0	空字符：校验位强制为 0
0	1	1	标记：校验位强制为 1
1	x	x	无校验位

- CHMODE：通道模式

CHMODE		模式描述
0	0	正常模式
0	1	自动回应
1	0	本地回环
1	1	远程回环

27.5.3 调试单元中断使能寄存器

名称：DBGU_IER

访问类型：只写

31	30	29	28	27	26	25	24
COMMRX	COMMTX	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	RXBUFF	TXBUFE	–	TXEMPTY	–
7	6	5	4	3	2	1	0
PARE	FRAME	OVRE	ENDTX	ENDRX	–	TXRDY	RXRDY

- RXRDY：使能 RXRDY 中断
- TXRDY：使能 TXRDY 中断
- ENDRX：使能接收传输结束中断
- ENDTX：使能发送结束中断
- OVRE：使能溢出错误中断
- FRAME：使能帧错误中断
- PARE：使能奇偶校验错误中断
- TXEMPTY：使能 TXEMPTY 中断
- TXBUFE：使能缓冲区空中断
- RXBUFF：使能缓冲区满中断
- COMMTX：使能 COMMTX（来自 ARM）中断
- COMMRT：使能 COMMRT（来自 ARM）中断

0 ＝无效

1 ＝使能对应的中断

27.5.4 调试单元中断禁用寄存器

名称：DBGU_IDR

访问类型：只写

31	30	29	28	27	26	25	24
COMMRX	COMMTX	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	RXBUFF	TXBUFE	–	TXEMPTY	–
7	6	5	4	3	2	1	0
PARE	FRAME	OVRE	ENDTX	ENDRX	–	TXRDY	RXRDY

- RXRDY：禁用 RXRDY 中断
- TXRDY：禁用 TXRDY 中断
- ENDRX：禁用接收传输结束中断
- ENDTX：禁用发送结束中断
- OVRE：禁用溢出错误中断
- FRAME：禁用帧错误中断
- PARE：禁用奇偶校验错误中断
- TXEMPTY：禁用 TXEMPTY 中断
- TXBUFE：禁用缓冲区空中断
- RXBUFF：禁用缓冲区满中断

- COMMTX：禁用 COMMTX（来自 ARM）中断
- COMMRT：禁用 COMMRT（来自 ARM）中断

0 ＝无效

1 ＝禁用对应的中断

27.5.5 调试单元中断屏蔽寄存器

名称：DBGU_IMR

访问类型：只读

31	30	29	28	27	26	25	24
COMMRX	COMMTX	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	RXBUFF	TXBUFE	–	TXEMPTY	–
7	6	5	4	3	2	1	0
PARE	FRAME	OVRE	ENDTX	ENDRX	–	TXRDY	RXRDY

- RXRDY：屏蔽 RXRDY 中断
- TXRDY：屏蔽 TXRDY 中断
- ENDRX：屏蔽接收传输结束中断
- ENDTX：屏蔽发送结束中断
- OVRE：屏蔽溢出错误中断
- FRAME：屏蔽帧错误中断
- PARE：屏蔽奇偶校验错误中断
- TXEMPTY：屏蔽 TXEMPTY 中断
- TXBUFE：屏蔽 TXBUFE 中断
- RXBUFF：屏蔽 RXBUFF 中断
- COMMTX：屏蔽 COMMTX 中断
- COMMRT：屏蔽 COMMRT 中断

0 ＝禁用对应中断

1 ＝使能对应中断

27.5.6 调试单元状态寄存器

名称：DBGU_SR

访问类型：只读

31	30	29	28	27	26	25	24
COMMRX	COMMTX	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	RXBUFF	TXBUFE	–	TXEMPTY	–
7	6	5	4	3	2	1	0
PARE	FRAME	OVRE	ENDTX	ENDRX	–	TXRDY	RXRDY

- RXRDY：接收器就绪

0= 最近一次读 DBGU_RHR 后未接收到字符或接收器已禁用

1=至少已收到一个完整的字符，被传输到 DBGU_RHR 而未被读取

- TXRDY：发送器就绪

0 =字符已被写入 DBGU_THR 还未传送到移位寄存器，或发送器被禁用。

1=无被写入 DBGU_THR 还未传送到移位寄存器。

- ENDRX：接收器传输结束

0 =来自接收器外设数据控制器通道的传输结束信号无效

1 =来自接收器外设数据控制器通道的传输结束信号有效

- RNDTX：发送器传输结束

0=来自发送器外设数据控制器通道的传输结束信号无效

1 =来自发送器外设数据控制器通道的传输结束信号有效

- OVRE：溢出错误

0 =上次 RSTSTA 后无溢出发生

1=上次 RSTSTA 后至少一次溢出发生

- FRAME：帧错误

0 =上次 RSTSTA 后无帧错误

1 =上次 RSTSTA 后至少一次帧错误

- PARE：奇偶校验错误

0 =上次 RSTSTA 后无奇偶校验错误发生

1 =上次 RSTSTA 后至少一次奇偶校验错误发生

- TXEMPTY：发送器空

0=DBGU_THR 中有字符，或发送器正在处理字符，或发送器被禁用

1= DBGU_THR 中无字符并且发送器没有正在处理字符。

- TXBUFE：发送缓冲器空

0=来自发送器 PDC 通道的缓冲期空的信号无效

1=来自发送器 PDC 通道的缓冲期空的信号有效

RXBUFF：接收缓冲期满

0=来自接收 PDC 通道的缓冲期满的信号无效

1=来自接收 PDC 通道的缓冲期满的信号有效

- COMMTX：调试通信通道写状态

0=来自 ARM 处理器的 COMMTX 无效

1=来自 ARM 处理器的 COMMTX 有效

- COMMRX：调试通信通道读状态

0=来自 ARM 处理器的 COMMRX 无效

1=来自 ARM 处理器的 COMMRX 有效

27.5.7 调试单元接收器保持寄存器

名称：DBGU_RHR

访问类型：只读

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

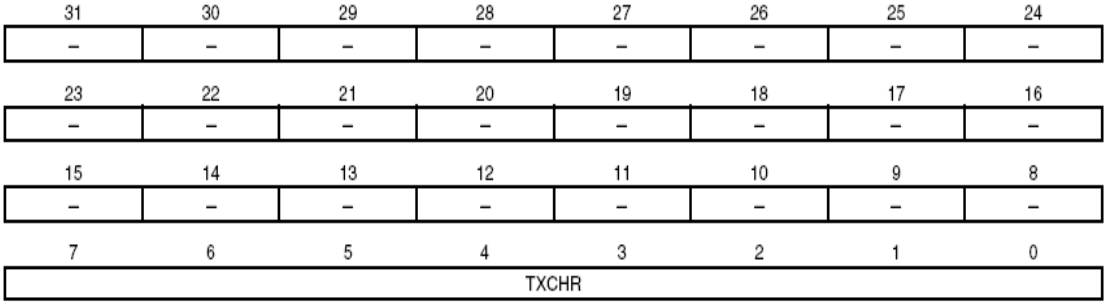
- RXCHR：接收的字符

如果 RXRDY 置位，则为上次接收的字符

27.5.8 调试单元发送保持寄存器

名称：DBGU_THR

访问类型：只写



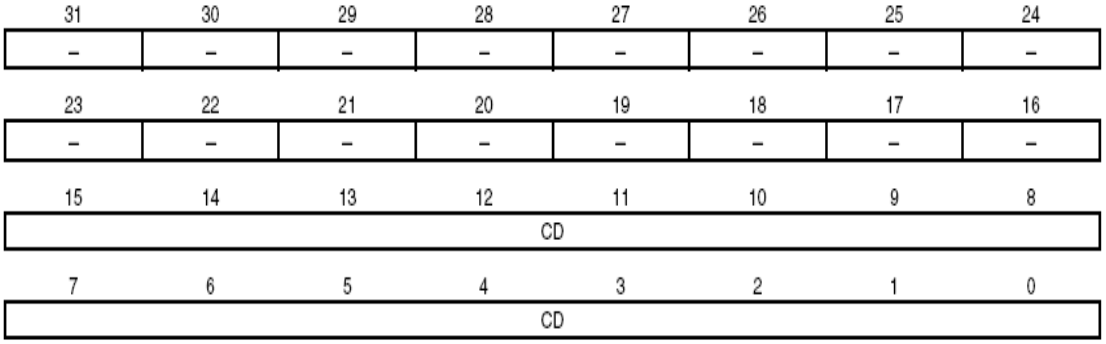
- TXCHR：待发送的字符

如果 TXRDY 未置位，则为当前字符发送后下一个待发送的字符

27.5.9 调试单元波特率发生器寄存器

名称：DBGU_BRGR

访问类型：读/写



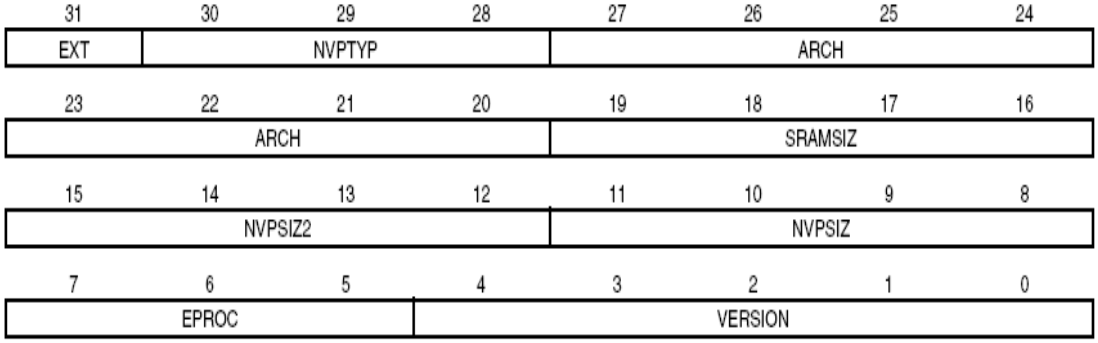
- CD：时钟分频器

CD	波特率时钟
0	禁用
1	MCK
2 到 65536	MCK/(CD x 16)

27.5.10 调试单元芯片 ID 寄存器

名称：DBGU_CIDR

访问类型：只读



- 版本：设备版本
- EPROC：嵌入式处理器

EPROC			处理器
0	0	1	ARM946ES
0	1	0	ARM7TDMI
1	0	0	ARM920T
1	0	1	ARM926EJS

- NVPSIZ：非易失性程序存储器大小

NVPSIZ				大小
0	0	0	0	无
0	0	0	1	8K 字节
0	0	1	0	16K 字节
0	0	1	1	32K 字节
0	1	0	0	保留
0	1	0	1	64K 字节
0	1	1	0	保留
0	1	1	1	128K 字节
1	0	0	0	保留
1	0	0	1	256K 字节
1	0	1	0	512K 字节
1	0	1	1	保留
1	1	0	0	1024K 字节
1	1	0	1	保留
1	1	1	0	2048K 字节
1	1	1	1	保留

- NVPSIZ2 第二个非易失性程序存储器大小

NVPSIZ2				大小
0	0	0	0	无
0	0	0	1	8K 字节
0	0	1	0	16K 字节
0	0	1	1	32K 字节
0	1	0	0	保留
0	1	0	1	64K 字节
0	1	1	0	保留
0	1	1	1	128K 字节
1	0	0	0	保留
1	0	0	1	256K 字节
1	0	1	0	512K 字节
1	0	1	1	保留
1	1	0	0	1024K 字节
1	1	0	1	保留
1	1	1	0	2048K 字节
1	1	1	1	保留

- SRAMSIZ：内部 SRAM 大小

SRAMSIZ				大小
0	0	0	0	保留
0	0	0	1	1K 字节
0	0	1	0	2K 字节
0	0	1	1	6K 字节
0	1	0	0	112K 字节
0	1	0	1	4K 字节
0	1	1	0	80K 字节
0	1	1	1	160K 字节
1	0	0	0	8K 字节
1	0	0	1	16K 字节
1	0	1	0	32K 字节
1	0	1	1	64K 字节
1	1	0	0	128K 字节
1	1	0	1	256K 字节
1	1	1	0	96K 字节
1	1	1	1	512K 字节

- ARCH：体系结构标识符

ARCH		体系结构
Hex	Bin	
0x19	0001 1001	AT91SAM9xx 系列
0x29	0010 1001	AT91SAM9XExx 系列
0x34	0011 0100	AT91x34 系列
0x37	0011 0111	CAP7 系列
0x39	0011 1001	CAP9 系列
0x3B	0011 1011	CAP11 系列
0x40	0100 0000	AT91x40 系列
0x42	0100 0010	AT91x42 系列
0x55	0101 0101	AT91x55 系列
0x60	0110 0000	AT91SAM7Axx 系列
0x61	0110 0001	AT91SAM7AQxx 系列
0x63	0110 0011	AT91x63 系列
0x70	0111 0000	AT91SAM7Sxx 系列
0x71	0111 0001	AT91SAM7XCxx 系列
0x72	0111 0010	AT91SAM7SExx 系列
0x73	0111 0011	AT91SAM7Lxx 系列
0x75	0111 0101	AT91SAM7Xxx 系列
0x92	1001 0010	AT91x92 系列
0xF0	1111 0000	AT75Cxx 系列

- NVPTYP：非易失性程序存储器类型

NVPTYP			存储器
0	0	0	ROM
0	0	1	ROMless 或片上 Flash
1	0	0	SRAM 仿真 ROM
0	1	0	嵌入式 Flash 存储器
0	1	1	ROM 和嵌入式 Flash 存储器 NVPSIZ 是 ROM 大小 NVPSIZ2 是 Flash 大小

- EXT：扩展标志

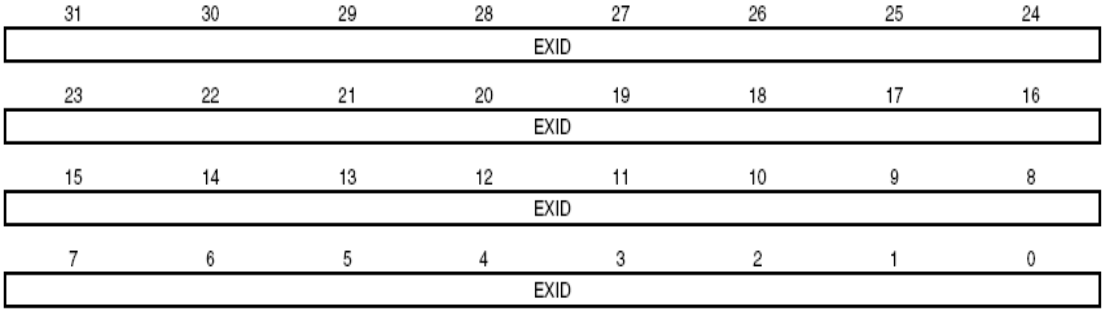
0=芯片 ID 在一无扩展的单寄存器中定义

1=存在扩展的芯片 ID

27.5.11 调试单元芯片 ID 扩展寄存器

名称：DBGU_EXID

访问类型：只读



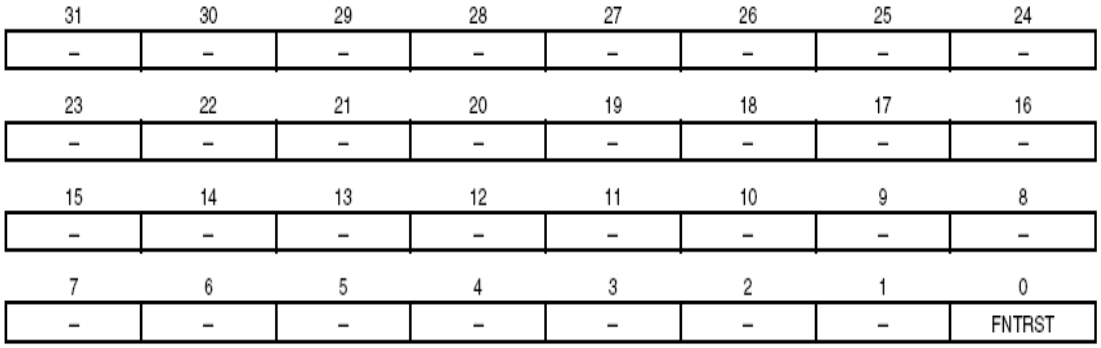
- EXID：芯片 ID 扩展

如果 DBGU_CIDR 中的 EXT 位是 0 则读取 0

27.5.12 调试单元强制 NTRST 寄存器

名称：DBGU_FNR

访问类型：读/写



- FNTRST：强制 NTRST

0=ARM 处理器的 TAP 控制器的 NTRST 由上电复位信号驱动

1=ARM 处理器的 TAP 控制器的 NTRST 保持低电平

28. 并行输入/输出控制器（PIO）

28．1 描述

并行输入/输出控制器（PIO）管理多达 32 个全可编程的输入/输出口线。每个 I/O 口线可被用作通用 I/O 或被指定为嵌入式外设功能。这就保证了产品引脚的有效优化。

PIO 控制器的每个 I/O 口线的特性：

- 输入变化中断使能任何 I/O 口线上的电平变化检测
- 毛刺滤波器能除去所有低于一个半时钟周期的脉冲
- 类似于开漏 I/O 口线的多驱动能力
- I/O 口线上拉控制
- 输入可见性和输出控制

PIO 控制器的异步输出提供多达 32 位的单写入操作中的数据输出

28.2 方块图

图 28-1 方块图

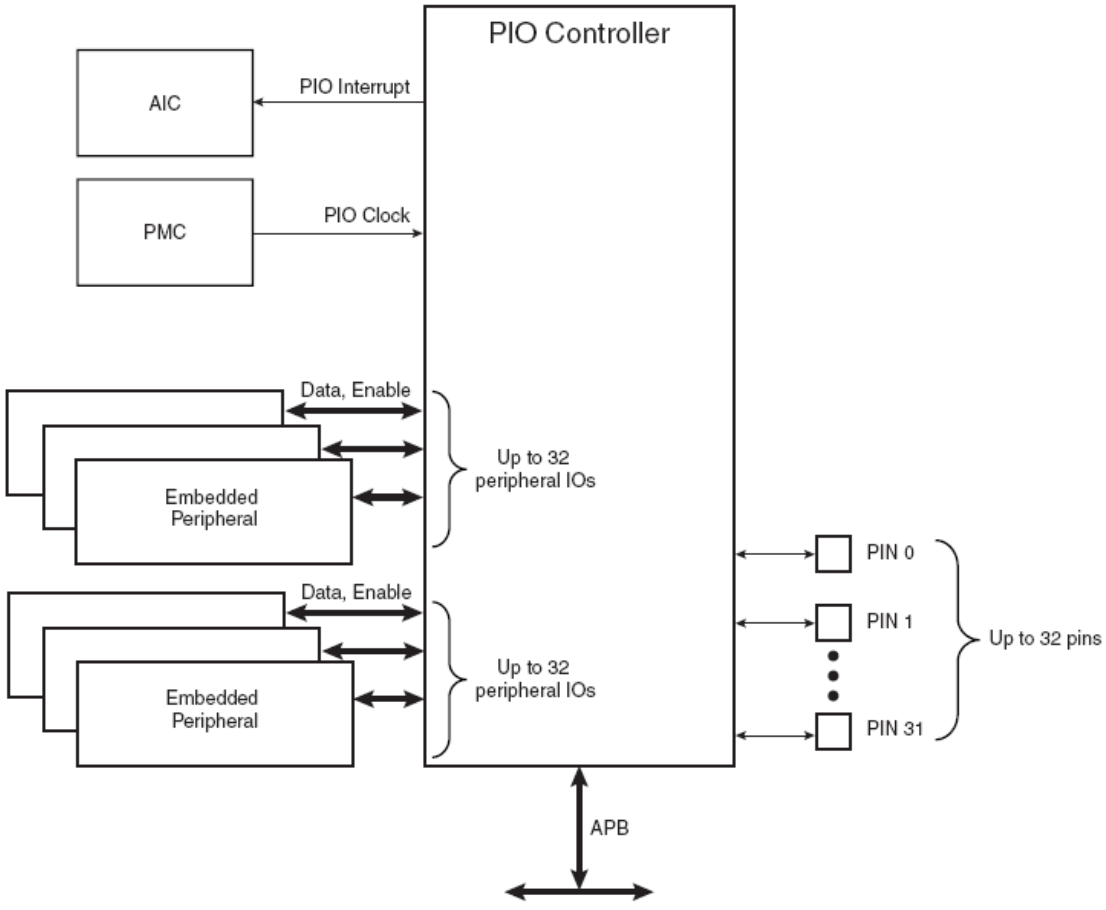
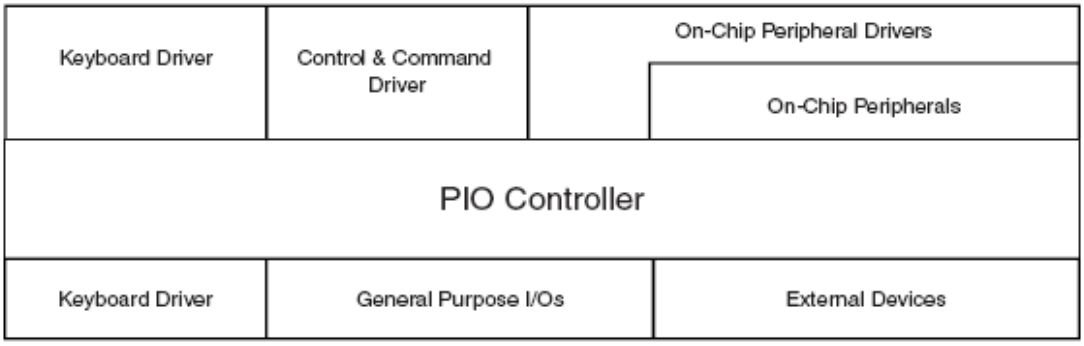


图 28-2 应用方块图



28.3 附属产品

28.3.1 引脚多路复用

每个引脚根据产品定义仅可配置为通用 I/O 口线，或一个 I/O 口线复用为一个或两个外设 I/O。由于复用是硬件定义的并因此以来于产品，所以硬件设计者和编程者必须根据应用谨慎决定 PIO 控制器的配置。当仅是通用 I/O 口线，就是说，无任何外设 I/O 复用，关于指定给外设的 PIO 控制器的编程无影响，并且仅 PIO 控制器可以控制产品驱动引脚。

28.3.2 外部中断口线

中断信号 FIQ 和 IRQ0 到 IRQn 通常通过 PIO 控制器多路复用。然而，由于 PIO 控制器对输入无效并且中断口线仅被用为输入，所以没必要分配 I/O 口线给中断功能。

28.3.3 电源管理

电源管理控制器控制 PIO 控制器时钟来节省功率。写入用户接口的任何寄存器都不需要 PIO 控制器时钟来使能。这就意味着 I/O 口线的配置不需要 PIO 控制器时钟来使能。

然而，当时钟被禁用，不是所有的 PIO 控制器特性都是可用的。注意输入变化中断和引脚点评的读取需要时钟来激活。

硬件复位后，PIO 时钟缺省为禁用

用户必须在任何访问输入口线信息前配置电源管理控制器

28.3.4 中断发生

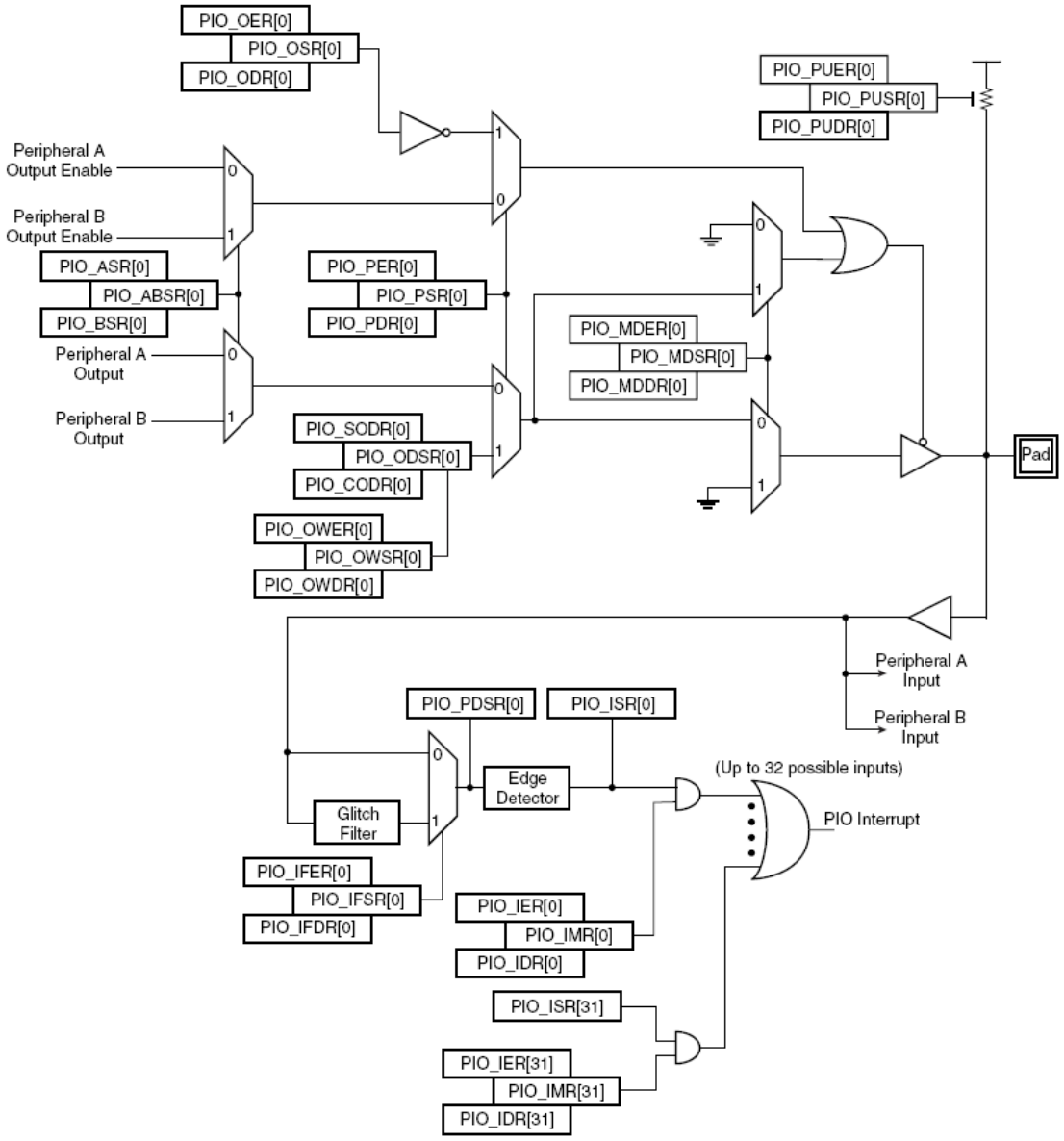
对于中断处理，认为 PIO 控制器是用户外设。这就是说 PIO 控制器中断口线被连接于中断源 2 到 31 之中。参考产品描述中的 PIO 控制器外设标识符来确认分配给 PIO 控制器的中断源。

只有 PIO 控制器时钟被使能时才产生 PIO 控制器中断

28.4 功能描述

PIO 控制器有多达 32 个全可编程 I/O 口线。图 28-3 給出了和每个 I/O 关联的大部分控制逻辑。除了 32 个之一可能编索引每个信号都有描述。

图 28-3 I/O 口线控制逻辑



28.4.1 上拉电阻控制

每个 I/O 口线都设计了一个内置上拉电阻。可通过分别写 PIO_PUER（上拉电阻使能寄存器）和 PIO_PUDR（上拉电阻禁用寄存器）来使能或禁用上拉电阻。写入这些寄存器导致 PIO_PUSR（上拉电阻状态寄存器）中对应位置位或清零。读取 PIO_PUSR 为 1 就是上拉电阻被禁用，读取 0 是上拉电阻被使能。

上拉电阻的控制可能和 I/O 口线的配置无关。

复位后，所有的上拉电阻被使能，即，PIO_PUSR 复位值是 0x0。

28.4.2 I/O 口线或外设功能选择

当引脚复用一个或两个外设功能，通过寄存器 PIO_PER（PIO 使能寄存器）和 PIO_PDR（PIO 禁用寄存器）控制功能选项。寄存器 PIO_PSR（PIO 状态寄存器）为置位和清零寄存器的结果，并且指示引脚是由对应外设控制还是由 PIO 控制器控制。0 值表示引脚由在 PIO_ABSR（AB 选择状态寄存器）中已选择的对应片上外设控制。1 值表示引脚由 PIO 控制器控制。

如果引脚被用作通用 I/O 口线（没有和片上外设复用），则 PIO_PER 和 PIO_PDR 无效并且 PIO_PSR 给对应位返回 1 值。

复位后，通常，I/O 口线由 PIO 控制器控制，即，PIO_PSR 复位为 1。然而，某些时候，PIO 口线要由外设控制（因为存储器片选口线在复位后必须被置为无效或外部存储器导出时地址口线必须被置为低电平）。因此，PIO_PSR 的复位值有产品定义，取决于设备的复用。

28.4.3 外设 A 或 B 选择

PIO 控制器在单引脚上提供多达两个外设功能的复用。通过写 PIO_ASR（A 选择寄存器）和 PIO_BSR（B 选择寄存器）来执行选择。PIO_ABSR（AB 选择寄存器）指示当前选择的是哪个外设口线。对每一个引脚，对应位是 0 点平意味着选择外设 A 而对应位是 1 表示选择外设 B。

注意外设口线 A 和 B 的复用仅影响输出口线。外设输入口线总是连接于输入引脚。

复位后，PIO_ABSR 是 0，因此表示所有的 PIO 口线被分配在外设 A 上。然而，外设 A 通常不驱动引脚，因为 PIO 控制器在 I/O 口线模式复位。不管引脚配置，写入 PIO_ASR 和 PIO_BSR 控制 PIO_ABSR。然而，给引脚分配外设功能时需要写对应的外设选择寄存器(PIO_ASR 或 PIO_BSR)还需写 PIO_PDR。

28.4.4 输出控制

当 I/O 口线分配外设功能时，即，PIO_PSR 中的对应位是 0，外设控制 I/O 口线的驱动。外设 A 或 B，取决于 PIO_ABSR 中的值，决定是否驱动引脚。

当 PIO 控制器控制 I/O 口线时，引脚可被配置为驱动，通过写 PIO_OER（输出使能寄存器）和 PIO_ODR（输出禁用寄存器）完成。

这些写操作的结果可在 PIO_OSR 中检测到。当此寄存器中的位是 0，对应 I/O 口线仅被用作输入。当此位是 1 时，对应 I/O 口线由 PIO 控制器控制。

通过写入 PIO_SODR（置位输出数据寄存器）和 PIO_CODR（清零输出数据寄存器）可确定驱动 I/O 口线的电平。这些写操作分别置位和清零 PIO_ODSR（输出数据状态寄存器），表示 I/O 口线上的数据驱动。无论引脚配置为由 PIO 控制器控制还是分配外设功能，写入 PIO_OER 和 PIO_ODR 来控制 PIO_OSR。这使得通过 PIO 控制器优先设置 I/O 口线的配置。

类似的，写入 PIO_SODR 和 PIO_CODR 影响 PIO_ODSR。这对于定义 I/O 口线上的第一级驱动很重要。

28.4.5 同步数据输出

用若干 PIO 控制所有并行总线需要对 PIO_SODR 和 PIO_CODR 寄存器执行两个连续的写操作。这将导致不可预期的暂态值。PIO 控制器通过单写访问 PIO_ODSR（输出数据状态寄存器）提供 PIO 输出的直接控制。仅写 PIO_OWSR(输出写状态寄存器)未屏蔽的位。通过写入 PIO_OWER（输出写使能寄存器）置位 PIO_OWSR 中的屏蔽位并通过写入 PIO_OWDR（输出写禁用寄存器）清零。

复位后，PIO_OWSR 复位值为 0x0，所有 I/O 口线上禁用同步数据输出。

28.4.6 多驱动控制（开漏）

通过使用多驱动功件在开漏时独立的编程每个 I/O。此特性允许若干驱动器连接于仅由每个设备置为低电平的 I/O 口线上。通常需要一个外部上拉电阻（或使能一个内部上拉电阻）来保证口线上的高电平。

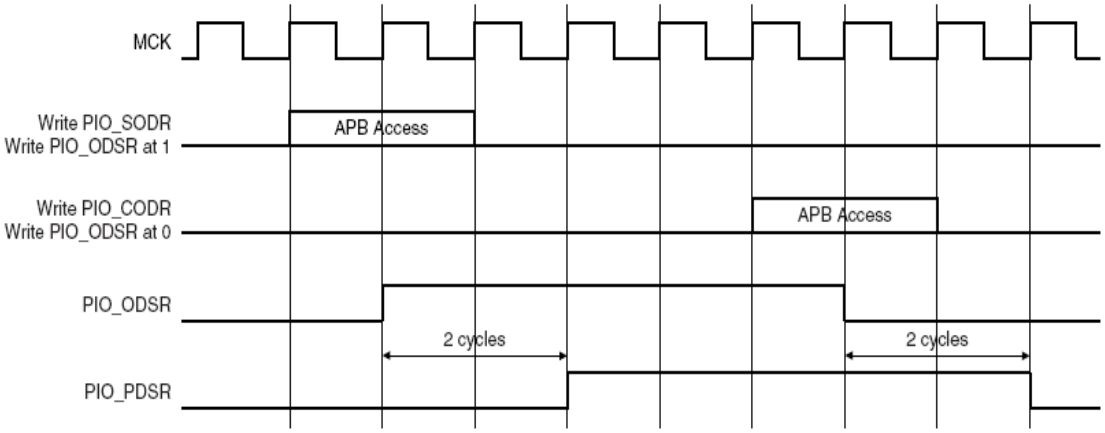
通过 PIO_MDER（多驱动使能寄存器）和 PIO_MDDR（多驱动禁用寄存器）控制多驱动特性。无论 PIO 控制器控制 I/O 口线还是分配一个外设功能都可选择多驱动。PIO_MDSR（多驱动状态寄存器）指示配置引脚支持外部驱动器。

复位后，禁用所有引脚上的多驱动特性，即，PIO_MDSR 复位值为 0x0。

28.4.7 输出口线时序

图 28-4 展示了怎样通过写 PIO_SODR 或 PIO_CODR，或通过直接写 PIO_ODSR 驱动输出。最后一种情况只有在 PIO_OWSR 中对应位置位时有效。图 28-4 展示了 PIO_PDSR 中的反馈有效的时序。

图 28-4 输出口线时序



28.4.8 寄存器

通过 PIO_PDSR（引脚数据状态寄存器）读取每个 I/O 口线上的电平。

存器指示 I/O 口线的电平，无论其配置是仅作为输入或由 PIO 控制器驱动或由外设驱动。

读取 I/O 口线电平需要使能 PIO 控制器的时钟，另外 PIO_PDSR 在禁用时钟的时候读取 I/O 口线上当前电平。

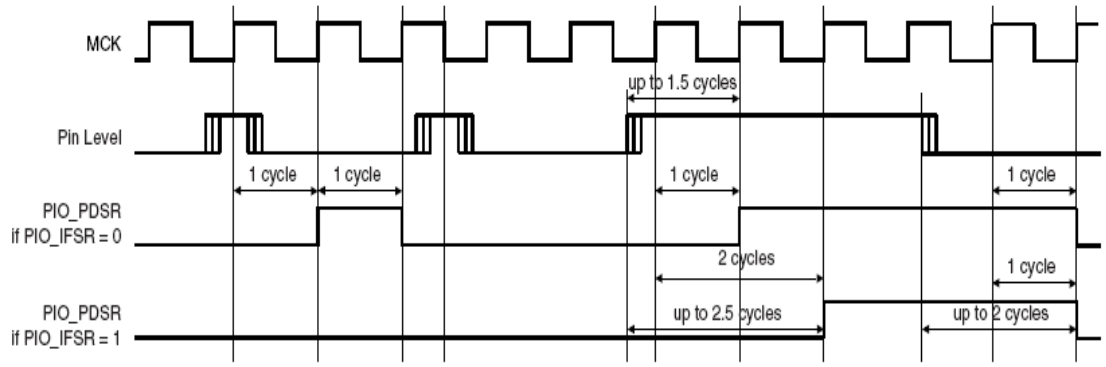
28.4.8 输入毛刺滤波

每个 I/O 口线上可选的输入毛刺滤波器可独立编程。当使能毛刺滤波器时，自动屏蔽小于 1/2 主控时钟周期（MCK）的毛刺，而通过大于等于 1 个主控时钟周期的脉冲。对于 1/2 主控时钟周期和 1 个主控时钟周期之间的脉冲取决于其出现的时间。因此对一个可见的脉冲其必超过一个主控时钟周期，而对于确定滤除的毛刺，其持续时间必超过 1/2 主控时钟周期。如果在下降沿到来前引脚电平发生变化，滤波器将引入一个主控时钟周期延迟。如图 28-5 所示。

毛刺滤波器由寄存器组：PIO_IFER（输入滤波器使能寄存器），PIO_IFDR（输入滤波器禁用寄存器）和 PIO_IFSR（输入滤波器状态寄存器）控制。分别写 PIO_IFER 和 PIO_IFDR 将置位和清零 PIO_IFSR 中位。最后一个滤波器使能 I/O 口线上的毛刺滤波器。

当毛刺滤波器被使能，并不修改外设上输入操作。仅对读取 PIO_PDSR 中值和输入变化中断检测起作用。毛刺滤波器需要 PIO 控制器时钟使能。

图 28-5 输入毛刺滤波器时序

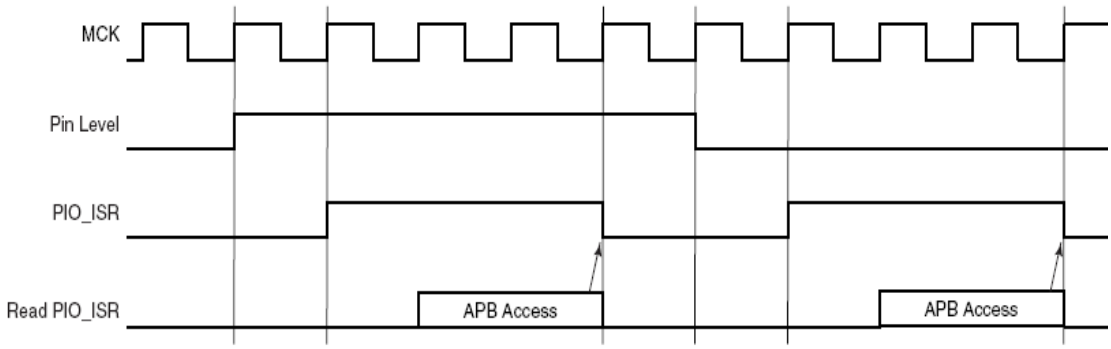


28.4.9 输入变化中断

PIO 控制器可被编程用来发生中断，当其检测到 I/O 口线上的输入变化时。通过写入 PIO_IER（中断使能寄存器）和 PIO_IDR（中断禁用寄存器）控制输入变化中断，两寄存器分别通过置位和清零 PIO_IMR（中断屏蔽寄存器）中的对应位来使能和禁用输入变化中断。因为输入变化检测仅有可能通过比较 I/O 口线上输入的两个连续采样来完成，所以 PIO 控制器时钟必须使能。无论 I/O 口线的配置，输入变化中断都可用，即仅配置为输入，由 PIO 控制器控制或分配外设功能。

当在 I/O 口线上检测到输入变化，PIO_ISR（中断状态寄存器）中的对应位置位。如果 PIO_IMR 中对应位置位，则激活 PIO 控制器中断口线。32 个通道的中断信号“线或”起来产生一个到高级中断控制器的中断信号。当软件读取 PIO_ISR 时，所有中断自动清零。这表明当 PIO_ISR 被读取时必须处理所有待定的中断。

图 28-6 输入变化中断时序



28.5 I/O 口线编程举例

编程举例如表 28-1 所示，用于定义以下配置：

- I/O 口线 0-3 上的 4 位输出端口，（应该用单写操作写入），开漏，带上拉电阻
- I/O 口线 4-7 上的 4 个输出信号（例如驱动 LED），驱动高和低电平，无上拉电阻
- I/O 口线 8-11 上的 4 个输入信号（例如读取按钮状态），带上拉电阻，毛刺滤波器和输入变化中断
- I/O 口线 12-15 上的 4 个输入信号读取外部设备状态（轮询，因此无输入变化中断），无上拉电阻，无毛刺

- I/O 口线 16-19 分配为外设 A 功能，带上拉电阻
- I/O 口线 20-23 分配为外设 B 功能，无上拉电阻
- I/O 口线 24-27 分配为带输入变化终端和上拉电阻的外设 A

表 28-1　　编程举例

寄存器	写入值
PIO_PER	0x0000 FFFF
PIO_PDR	0x0FFF 0000
PIO_OER	0x0000 00FF
PIO_ODR	0x0FFF FF00
PIO_IFER	0x0000 0F00
PIO_IFDR	0x0FFF F0FF
PIO_SODR	0x0000 0000
PIO_CODR	0x0FFF FFFF
PIO_IER	0x0F00 0F00
PIO_IDR	0x00FF F0FF
PIO_MDER	0x0000 000F
PIO_MDDR	0x0FFF FFF0
PIO_PUDR	0x00F0 00F0
PIO_PUER	0x0F0F FF0F
PIO_ASR	0x0F0F 0000
PIO_BSR	0x00F0 0000
PIO_OWER	0x0000 000F
PIO_OWDR	0x0FFF FFF0

28.6　　用户接口

PIO 控制器控制的每个 I/O 口线和每个 PIO 控制器用户接口寄存器一位相关。每个寄存器 32 位宽。如果未定义并行 I/O 口线，写入对应位无效。为定义位 读为 0。如果 I/O 口线未和任何外设复用，PIO 控制器和 PIO_PSR 控制 I/O 口线系统的返回 1。

表　28-2　　寄存器映射

偏移量	寄存器	名称	访问类型	复位值
0x0000	PIO 使能寄存器	PIO_PER	只写	-
0x0004	PIO 禁用寄存器	PIO_PDR	只写	-
0x0008	PIO 状态寄存器	PIO_PSR	只读	(1)
0x000C	保留			
0x0010	输出使能寄存器	PIO_OER	只写	-
0x0014	输出禁用寄存器	PIO_ODR	只写	-
0x0018	输出状态寄存器	PIO_OSR	只读	0x0000 0000
0x001C	保留			
0x0020	毛刺输入滤波器使能寄存器	PIO_IFER	只写	-
0x0024	毛刺输入滤波器禁用寄存器	PIO_IFDR	只写	-
0x0028	毛刺输入滤波器状态寄存器	PIO_IFSR	只读	0x0000 0000
0x002C	保留			

0x0030	置位输出数据寄存器	PIO_SODR	只写	-
0x0034	请零输出数据寄存器	PIO_CODR	只写	
0x0038	输出数据状态寄存器	PIO_ODSR	只读或读写	-
0x003C	引脚数据状态寄存器	PIO_PDSR	只读	(3)
0x0040	中断使能寄存器	PIO_IER	只写	-
0x0044	中断禁用寄存器	PIO_IDR	只写	-
0x0048	中断屏蔽寄存器	PIO_IMR	只读	0x0000 0000
0x004C	中断状态寄存器	PIO_ISR	只读	0x0000 0000
0x0050	多驱动使能寄存器	PIO_MDER	只写	-
0x0054	多驱动禁用寄存器	PIO_MDDR	只写	-
0x0058	多驱动状态寄存器	PIO_MDSR	只读	0x0000 0000
0x005C	保留			
0x0060	上拉电阻禁用寄存器	PIO_PUDR	只写	-
0x0064	上拉电阻使能寄存器	PIO_PUER	只写	-
0x0068	焊盘上拉阻抗状态寄存器	PIO_PUSR	只读	0x0000 0000
0x006C	保留			
0x0070	外设 A 选择寄存器	PIO_ASR	只写	-
0x0074	外设 B 选择寄存器	PIO_BSR	只写	-
0x0078	AB 状态寄存器	PIO_ABSR	只读	0x0000 0000
0x007C-0x009C	保留			
0x00A0	输出写使能	PIO_OWER	只写	-
0x00A4	输出写禁用	PIO_OWDR	只写	-
0x00A8	输出写状态寄存器	PIO_OWSR	只读	0x0000 0000
0x00AC	保留			

注意：1.PIO_PSR 的复位值取决于产品

2.PIO_ODSR 是只读或读写取决于 PIO_OWSR I/O 口线

3.PIO_PDSR 的复位值取决于 I/O 口线上的电平。读取 I/O 口线电平需要使能 PIO 控制器，另外 PIO_PDSR 在时钟禁用的时候读取 I/O 口线上当前电平

4.PIO_ISR 复位值为 0x0。然而，首次读取寄存器可能读取一个不同的值 ，因为可能发生输入变化。

5.仅此系列寄存器通过对第一个寄存器写 1 请零状态，并通过对第二个寄存器写 1 置位状态。

28.6.1 PIO 控制器 PIO 使能寄存器

名称：PIO_PER

访问类型：只写

31	30	29	28	27	26	25	24
P31	P30	P29	P28	P27	P26	P25	P24
23	22	21	20	19	18	17	16
P23	P22	P21	P20	P19	P18	P17	P16
15	14	13	12	11	10	9	8
P15	P14	P13	P12	P11	P10	P9	P8
7	6	5	4	3	2	1	0
P7	P6	P5	P4	P3	P2	P1	P0

- P0-P31：PIO 使能

0=无效

1=使能 PIO 来控制对应引脚（禁用引脚的外设控制）

28.6.2 P0-P31：PIO 禁用寄存器

名称：PIO_PDR

访问类型：只写

31	30	29	28	27	26	25	24
P31	P30	P29	P28	P27	P26	P25	P24
23	22	21	20	19	18	17	16
P23	P22	P21	P20	P19	P18	P17	P16
15	14	13	12	11	10	9	8
P15	P14	P13	P12	P11	P10	P9	P8
7	6	5	4	3	2	1	0
P7	P6	P5	P4	P3	P2	P1	P0

- P0-P31：PIO 禁用

0=无效

1=禁用 PIO 控制对应引脚（使能引脚的外设控制）

28.6.3 PIO 控制器 PIO 状态寄存器

名称：PIO_PSR

访问类型：只读

31	30	29	28	27	26	25	24
P31	P30	P29	P28	P27	P26	P25	P24
23	22	21	20	19	18	17	16
P23	P22	P21	P20	P19	P18	P17	P16
15	14	13	12	11	10	9	8
P15	P14	P13	P12	P11	P10	P9	P8
7	6	5	4	3	2	1	0
P7	P6	P5	P4	P3	P2	P1	P0

- P0-P31：PIO 状态

0=对应 I/O 口线上的 PIO 无效（外设有效）

1=对应 I/O 口线上的 PIO 有效（外设无效）

28.6.4 PIO 控制器输出使能寄存器

名称：PIO_OER

访问类型：只写

31	30	29	28	27	26	25	24
P31	P30	P29	P28	P27	P26	P25	P24
23	22	21	20	19	18	17	16
P23	P22	P21	P20	P19	P18	P17	P16
15	14	13	12	11	10	9	8
P15	P14	P13	P12	P11	P10	P9	P8
7	6	5	4	3	2	1	0
P7	P6	P5	P4	P3	P2	P1	P0

- P0-P31：输出使能

0=无效

1=使能 I/O 口线上的输出

28.6.5 PIO 控制器输出禁用寄存器

名称：PIO_ODR

访问类型：只写

31	30	29	28	27	26	25	24
P31	P30	P29	P28	P27	P26	P25	P24
23	22	21	20	19	18	17	16
P23	P22	P21	P20	P19	P18	P17	P16
15	14	13	12	11	10	9	8
P15	P14	P13	P12	P11	P10	P9	P8
7	6	5	4	3	2	1	0
P7	P6	P5	P4	P3	P2	P1	P0

- P0-P31：输出禁用

0=无效

1=禁用 I/O 口线上的输出

28.6.6 PIO 控制器输出状态寄存器

名称：PIO_OSR

访问类型：只读

31	30	29	28	27	26	25	24
P31	P30	P29	P28	P27	P26	P25	P24
23	22	21	20	19	18	17	16
P23	P22	P21	P20	P19	P18	P17	P16
15	14	13	12	11	10	9	8
P15	P14	P13	P12	P11	P10	P9	P8
7	6	5	4	3	2	1	0
P7	P6	P5	P4	P3	P2	P1	P0

- P0-P31：输出状态

0=I/O 口线是一个纯输入

1=I/O 口线是已使能的输出

28.6.7 PIO 控制器输入滤波器使能寄存器

名称：PIO_IFER

访问类型：只写

31	30	29	28	27	26	25	24
P31	P30	P29	P28	P27	P26	P25	P24
23	22	21	20	19	18	17	16
P23	P22	P21	P20	P19	P18	P17	P16
15	14	13	12	11	10	9	8
P15	P14	P13	P12	P11	P10	P9	P8
7	6	5	4	3	2	1	0
P7	P6	P5	P4	P3	P2	P1	P0

- P0-P31：输入滤波器使能

0=无效

1=使能 I/O 口线上的输入毛刺滤波器

28.6.8 PIO 控制器输入滤波器禁用寄存器

名称：PIO_IFDR

访问类型：只写

31	30	29	28	27	26	25	24
P31	P30	P29	P28	P27	P26	P25	P24
23	22	21	20	19	18	17	16
P23	P22	P21	P20	P19	P18	P17	P16
15	14	13	12	11	10	9	8
P15	P14	P13	P12	P11	P10	P9	P8
7	6	5	4	3	2	1	0
P7	P6	P5	P4	P3	P2	P1	P0

- P0-P31：输入滤波器禁用

0=无效

1=禁用 I/O 口线上的输入毛刺滤波器

28.6.9 PIO 控制器输入滤波器状态寄存器

名称：PIO_IFSR

访问类型：只读

31	30	29	28	27	26	25	24
P31	P30	P29	P28	P27	P26	P25	P24
23	22	21	20	19	18	17	16
P23	P22	P21	P20	P19	P18	P17	P16
15	14	13	12	11	10	9	8
P15	P14	P13	P12	P11	P10	P9	P8
7	6	5	4	3	2	1	0
P7	P6	P5	P4	P3	P2	P1	P0

- P0-P31：输入滤波器状态

0=禁用 I/O 口线上的输入毛刺滤波器

1=使能 I/O 口线上的输入毛刺滤波器

28.6.10 PIO 控制器置位输出数据寄存器

名称：PIO_SODR

访问类型：只写

31	30	29	28	27	26	25	24
P31	P30	P29	P28	P27	P26	P25	P24
23	22	21	20	19	18	17	16
P23	P22	P21	P20	P19	P18	P17	P16
15	14	13	12	11	10	9	8
P15	P14	P13	P12	P11	P10	P9	P8
7	6	5	4	3	2	1	0
P7	P6	P5	P4	P3	P2	P1	P0

- P0-P31：置位输出数据

0=无效

1=置位 I/O 口线上驱动的数据

28.6.11 PIO 控制器清零输出数据寄存器

名称：PIO_CODR

访问类型：只写

31	30	29	28	27	26	25	24
P31	P30	P29	P28	P27	P26	P25	P24
23	22	21	20	19	18	17	16
P23	P22	P21	P20	P19	P18	P17	P16
15	14	13	12	11	10	9	8
P15	P14	P13	P12	P11	P10	P9	P8
7	6	5	4	3	2	1	0
P7	P6	P5	P4	P3	P2	P1	P0

- P0-P31：置位输出数据

0=无效

1=清零 I/O 口线上驱动的数据

28.6.12 PIO 控制器输出数据状态寄存器

名称：PIO_ODSR

访问类型：只读或读写

31	30	29	28	27	26	25	24
P31	P30	P29	P28	P27	P26	P25	P24
23	22	21	20	19	18	17	16
P23	P22	P21	P20	P19	P18	P17	P16
15	14	13	12	11	10	9	8
P15	P14	P13	P12	P11	P10	P9	P8
7	6	5	4	3	2	1	0
P7	P6	P5	P4	P3	P2	P1	P0

- P0-P31：输出数据状态

0=I/O 口线上驱动的数据是 0

1=I/O 口线上驱动的数据是 1

28.6.13 PIO 控制器引脚数据状态寄存器

名称：PIO_PDSR

访问类型：只读

31	30	29	28	27	26	25	24
P31	P30	P29	P28	P27	P26	P25	P24
23	22	21	20	19	18	17	16
P23	P22	P21	P20	P19	P18	P17	P16
15	14	13	12	11	10	9	8
P15	P14	P13	P12	P11	P10	P9	P8
7	6	5	4	3	2	1	0
P7	P6	P5	P4	P3	P2	P1	P0

- P0-P31：输出数据状态

0=I/O 口线是 0 电平

1=I/O 口线是 1 电平

28.6.14　PIO 控制器中断使能寄存器

名称：PIO_IER

访问类型：只写

31	30	29	28	27	26	25	24
P31	P30	P29	P28	P27	P26	P25	P24
23	22	21	20	19	18	17	16
P23	P22	P21	P20	P19	P18	P17	P16
15	14	13	12	11	10	9	8
P15	P14	P13	P12	P11	P10	P9	P8
7	6	5	4	3	2	1	0
P7	P6	P5	P4	P3	P2	P1	P0

- P0-P31：输入变化中断使能

0=无效

1=使能 I/O 口线上的输入变化中断

28.6.15　PIO 控制器中断禁用寄存器

名称：PIO_IDR

访问类型：只写

31	30	29	28	27	26	25	24
P31	P30	P29	P28	P27	P26	P25	P24
23	22	21	20	19	18	17	16
P23	P22	P21	P20	P19	P18	P17	P16
15	14	13	12	11	10	9	8
P15	P14	P13	P12	P11	P10	P9	P8
7	6	5	4	3	2	1	0
P7	P6	P5	P4	P3	P2	P1	P0

- P0-P31：输入变化中断禁用

0=无效

1=禁用 I/O 口线上的输入变化中断

28.6.16　PIO 控制器中断屏蔽寄存器

名称：PIO_IMR

访问类型：只读

31	30	29	28	27	26	25	24
P31	P30	P29	P28	P27	P26	P25	P24
23	22	21	20	19	18	17	16
P23	P22	P21	P20	P19	P18	P17	P16
15	14	13	12	11	10	9	8
P15	P14	P13	P12	P11	P10	P9	P8
7	6	5	4	3	2	1	0
P7	P6	P5	P4	P3	P2	P1	P0

- P0-P31：输入变化中断屏蔽

0=禁用 I/O 口线上的输入变化中断

1=使能 I/O 口线上的输入变化中断

28.6.17　PIO 控制器中断状态寄存器

名称：PIO_ISR

访问类型：只读

31	30	29	28	27	26	25	24
P31	P30	P29	P28	P27	P26	P25	P24
23	22	21	20	19	18	17	16
P23	P22	P21	P20	P19	P18	P17	P16
15	14	13	12	11	10	9	8
P15	P14	P13	P12	P11	P10	P9	P8
7	6	5	4	3	2	1	0
P7	P6	P5	P4	P3	P2	P1	P0

- P0-P31：输入变化中断状态

0=上次读取 PIO_ISR 后或复位后，I/O 口线上未检测到输入变化

1=上次读取 PIO_ISR 后或复位后，I/O 口线上至少检测到一次输入变化

28.6.18　PIO 多驱动使能寄存器

名称：PIO_MDER

访问类型：只写

31	30	29	28	27	26	25	24
P31	P30	P29	P28	P27	P26	P25	P24
23	22	21	20	19	18	17	16
P23	P22	P21	P20	P19	P18	P17	P16
15	14	13	12	11	10	9	8
P15	P14	P13	P12	P11	P10	P9	P8
7	6	5	4	3	2	1	0
P7	P6	P5	P4	P3	P2	P1	P0

- P0-P31：多驱动使能

0=无效

1=使能 I/O 口线上的多驱动

28.6.19　PIO 多驱动禁用寄存器

名称：PIO_MDDR

访问类型：只写

31	30	29	28	27	26	25	24
P31	P30	P29	P28	P27	P26	P25	P24
23	22	21	20	19	18	17	16
P23	P22	P21	P20	P19	P18	P17	P16
15	14	13	12	11	10	9	8
P15	P14	P13	P12	P11	P10	P9	P8
7	6	5	4	3	2	1	0
P7	P6	P5	P4	P3	P2	P1	P0

- P0-P31：多驱动禁用

0=无效

1=禁用 I/O 口线上的多驱动

28.6.20　PIO 多驱动状态寄存器

名称：PIO_MDSR

访问类型：只读

31	30	29	28	27	26	25	24
P31	P30	P29	P28	P27	P26	P25	P24
23	22	21	20	19	18	17	16
P23	P22	P21	P20	P19	P18	P17	P16
15	14	13	12	11	10	9	8
P15	P14	P13	P12	P11	P10	P9	P8
7	6	5	4	3	2	1	0
P7	P6	P5	P4	P3	P2	P1	P0

- P0-P31：多驱动状态

0=禁用 I/O 口线上的多驱动。以高低电平驱动引脚。

1=使能 I/O 口线上的多驱动。仅以低电平驱动引脚

28.6.21 PIO 上拉电阻禁用寄存器

名称：PIO_PUDR

访问类型：只写

31	30	29	28	27	26	25	24
P31	P30	P29	P28	P27	P26	P25	P24
23	22	21	20	19	18	17	16
P23	P22	P21	P20	P19	P18	P17	P16
15	14	13	12	11	10	9	8
P15	P14	P13	P12	P11	P10	P9	P8
7	6	5	4	3	2	1	0
P7	P6	P5	P4	P3	P2	P1	P0

- P0-P31：禁用上拉电阻

0=无效

1=禁用 I/O 口线上的上拉电阻寄存器

28.6.22 PIO 上拉电阻使能寄存器

名称：PIO_PUER

访问类型：只写

31	30	29	28	27	26	25	24
P31	P30	P29	P28	P27	P26	P25	P24
23	22	21	20	19	18	17	16
P23	P22	P21	P20	P19	P18	P17	P16
15	14	13	12	11	10	9	8
P15	P14	P13	P12	P11	P10	P9	P8
7	6	5	4	3	2	1	0
P7	P6	P5	P4	P3	P2	P1	P0

- P0-P31：使能上拉电阻

0=无效

1=使能 I/O 口线上的上拉电阻寄存器

28.6.23 PIO 上拉电阻状态寄存器

名称：PIO_PUSR

访问类型：只读

31	30	29	28	27	26	25	24
P31	P30	P29	P28	P27	P26	P25	P24
23	22	21	20	19	18	17	16
P23	P22	P21	P20	P19	P18	P17	P16
15	14	13	12	11	10	9	8
P15	P14	P13	P12	P11	P10	P9	P8
7	6	5	4	3	2	1	0
P7	P6	P5	P4	P3	P2	P1	P0

- P0-P31：上拉电阻状态

0=使能 I/O 口线上的上拉电阻

1=禁用 I/O 口线上的上拉电阻

28.6.24 PIO 外设 A 选择寄存器

名称：PIO_ASR

访问类型：只写

31	30	29	28	27	26	25	24
P31	P30	P29	P28	P27	P26	P25	P24
23	22	21	20	19	18	17	16
P23	P22	P21	P20	P19	P18	P17	P16
15	14	13	12	11	10	9	8
P15	P14	P13	P12	P11	P10	P9	P8
7	6	5	4	3	2	1	0
P7	P6	P5	P4	P3	P2	P1	P0

- P0-P31：外设 A 选择

0=无效

1=分配 I/O 口线为外设 A 功能

28.6.25 PIO 外设 B 选择寄存器

名称：PIO_BSR

访问类型：只写

31	30	29	28	27	26	25	24
P31	P30	P29	P28	P27	P26	P25	P24
23	22	21	20	19	18	17	16
P23	P22	P21	P20	P19	P18	P17	P16
15	14	13	12	11	10	9	8
P15	P14	P13	P12	P11	P10	P9	P8
7	6	5	4	3	2	1	0
P7	P6	P5	P4	P3	P2	P1	P0

- P0-P31：外设 B 选择

0=无效

1=分配 I/O 口线为外设 B 功能

28.6.26 PIO 外设 A B 状态寄存器

名称：PIO_ABSR

访问类型：只读

31	30	29	28	27	26	25	24
P31	P30	P29	P28	P27	P26	P25	P24
23	22	21	20	19	18	17	16
P23	P22	P21	P20	P19	P18	P17	P16
15	14	13	12	11	10	9	8
P15	P14	P13	P12	P11	P10	P9	P8
7	6	5	4	3	2	1	0
P7	P6	P5	P4	P3	P2	P1	P0

- P0-P31：外设 A B 状态

0=I/O 口线分配为外设 A

1=I/O 口线分配为外设 B

28.6.27 PIO 输出写使能寄存器

名称：PIO_OWER

访问类型：只写

31	30	29	28	27	26	25	24
P31	P30	P29	P28	P27	P26	P25	P24
23	22	21	20	19	18	17	16
P23	P22	P21	P20	P19	P18	P17	P16
15	14	13	12	11	10	9	8
P15	P14	P13	P12	P11	P10	P9	P8
7	6	5	4	3	2	1	0
P7	P6	P5	P4	P3	P2	P1	P0

- P0-P31：输出写使能

0=无效

1=对 I/O 口线使能写 PIO_ODSR

28.6.28　　PIO 输出写禁用寄存器

名称：PIO_OWDR

访问类型：只写

31	30	29	28	27	26	25	24
P31	P30	P29	P28	P27	P26	P25	P24
23	22	21	20	19	18	17	16
P23	P22	P21	P20	P19	P18	P17	P16
15	14	13	12	11	10	9	8
P15	P14	P13	P12	P11	P10	P9	P8
7	6	5	4	3	2	1	0
P7	P6	P5	P4	P3	P2	P1	P0

- P0-P31：输出写禁用

0=无效

1=对禁用 I/O 口线写 PIO_ODSR

28.6.29　PIO 输出写状态寄存器

名称：PIO_OWSR

访问类型：只读

31	30	29	28	27	26	25	24
P31	P30	P29	P28	P27	P26	P25	P24
23	22	21	20	19	18	17	16
P23	P22	P21	P20	P19	P18	P17	P16
15	14	13	12	11	10	9	8
P15	P14	P13	P12	P11	P10	P9	P8
7	6	5	4	3	2	1	0
P7	P6	P5	P4	P3	P2	P1	P0

- P0-P31：输出写状态

0=写入 PIO_ODSR 不影响 I/O 口线

1=写入 PIO_ODSR 影响 I/O 口线

29 串行外设接口（SPI）

29.1 描述

串行外设接口（SPI）电路提供一个可在主/从模式下与外部设备通信的同步数据链路。如果系统连接了一个外部处理器则还可以实现处理器间的通信。

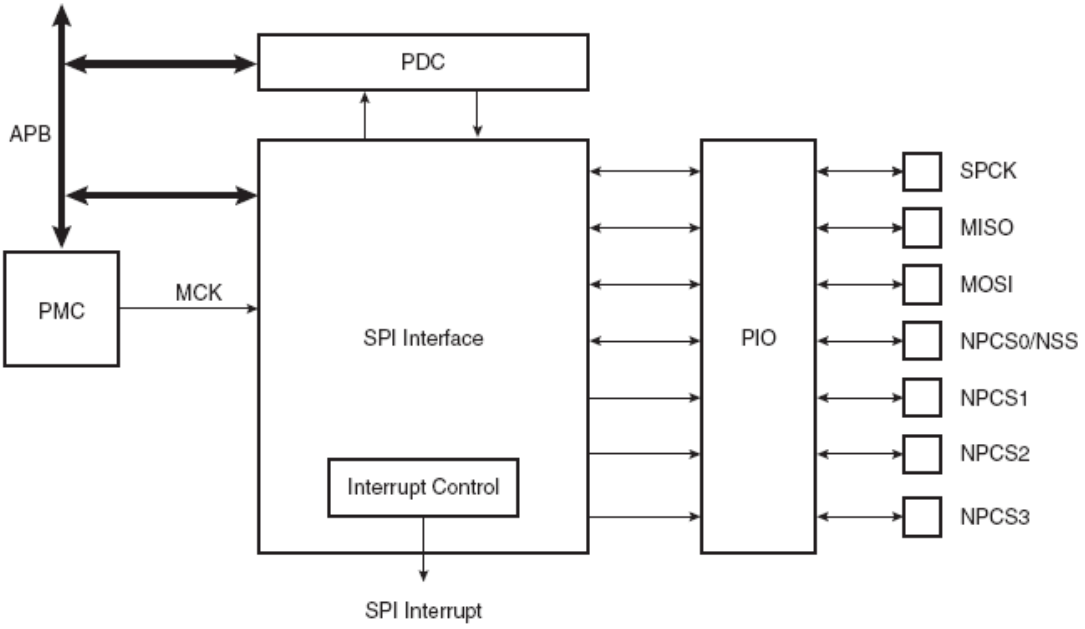
串行外设接口（SPI）本质上是一个串行传送数据位到其他 SPI 的移位寄存器。在一个数据传送期间，一个 SPI 系统作为控制数据流的“主控”运行，而其他设备作为通过主控将数据移入或移出的“从设备”在运行。不同的 CPU 可以轮流作为主控（多主控协议和单主控协议相反，单主控协议中一个 CPU 始终是主控而其他的始终是从设备）并且一个主控可同时将数据移入多个从设备。然而，仅单个从设备可以在任何给定时间驱动其输出来将数据写回到主控。

SPI 系统由两个数据口线和两个控制口线组成：

- 主控输出从设备输入（MOSI）：此数据口线用来将主控输出的数据移入从设备。
- 主控输入从设备输出(MISO)：此数据口线用来将从设备输出的数据移入主控。在任何特定的传送期间只能有单个从设备在发送数据。
- 串行时钟(SPCK)：此控制口线由主控驱动，校验数据位流。主控可以以一种波特率发送数据；SPCK 口线对已发送的每位循环一次。
- 从设备选择(NSS)：此控制口线允许由硬件开启和关断

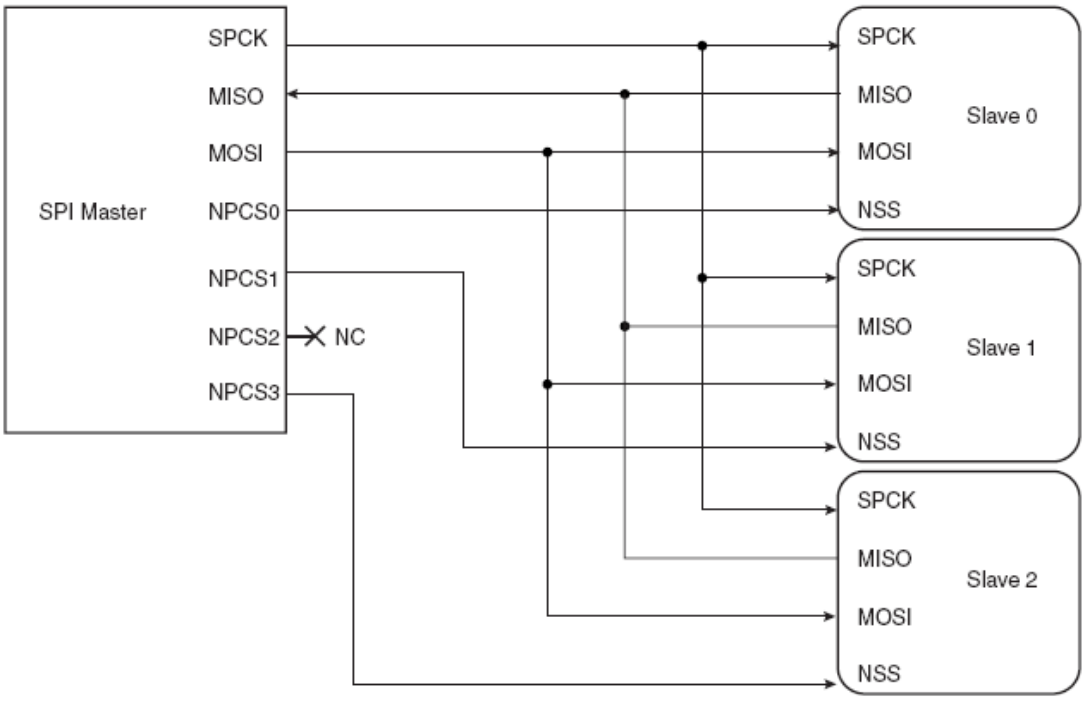
29.2 方块图

图 29-1 方块图



29.3 应用方块图

图 29-2 应用方块图：单主控/多从设备的实现



29.4 信号描述

引脚名称	引脚描述	类型	
		主控	从设备
MISO	主控输入从设备输出	输入	输出
MOSI	主控输出从设备输入	输出	输入
SPCK	串行时钟	输出	输入
NPCS1-NPCS3	外设芯片选择	输出	未用
NPCS0/NSS	外设芯片选择/从设备选择	输出	输入

29.5 产品依赖性

29.5.1 I/O 口线

用于连接相应外部设备的引脚可与 PIO 口线复用。编程者必须首先编程 PIO 控制器来分配 SPI 引脚为其外设功能。

29.5.2 电源管理

SPI 可通过电源管理控制器(PMC)同步，因此编程者必须首先配置 PMC 来使能 SPI 时钟。

29.5.3 中断

SPI 接口有一个连接于高级中断控制器(AIC)的中断口线。

处理 SPI 中断需要在配置 SPI 前编程 SIC。

29.6 功能描述

29.6.1 操作模式

SPI 操作在主/从模式下。

通过向模式寄存器中的 MSTR 位写 1 来编程主控模式下的操作。配置引脚 NPCS0 到 NPCS3 为输出，驱动 SPCK 引脚，MISO 口线连接于接收器输入并且被发送器驱动为输出。如果对 MSTR 位写 0，则 SPI 操作在从模式。发送器输出驱动 MISO 口线，MOSI 口线连接于接收器输入，发送器驱动 SPCK 引脚来同步接收器。NPCS0 变为输入，并被用作从设备选择信号(NSS)。引脚 NPCS1 到

NPCS3 未被驱动而可被用作其他用途。

数据传输同样可对两种操作模式编程。仅在主控模式激活波特率发生器。

29.6.2 数据传输

对数据传输有四个极性和相位的组合可用。用片选寄存器中的 CPOL 位编程时钟极性。用 NCPHA 位编程时钟相位。这两个参数决定已驱动和采样数据的时钟信号边沿。每两个参数有两个可能的状态，产生四个互相兼容的可能的组合。因此，一个主/从对必须用同样的参数对值来通信。若使用多个从设备并且固定为不同的配置，则主控每次需要和不同从设备通信时必须重新配置主控自身。

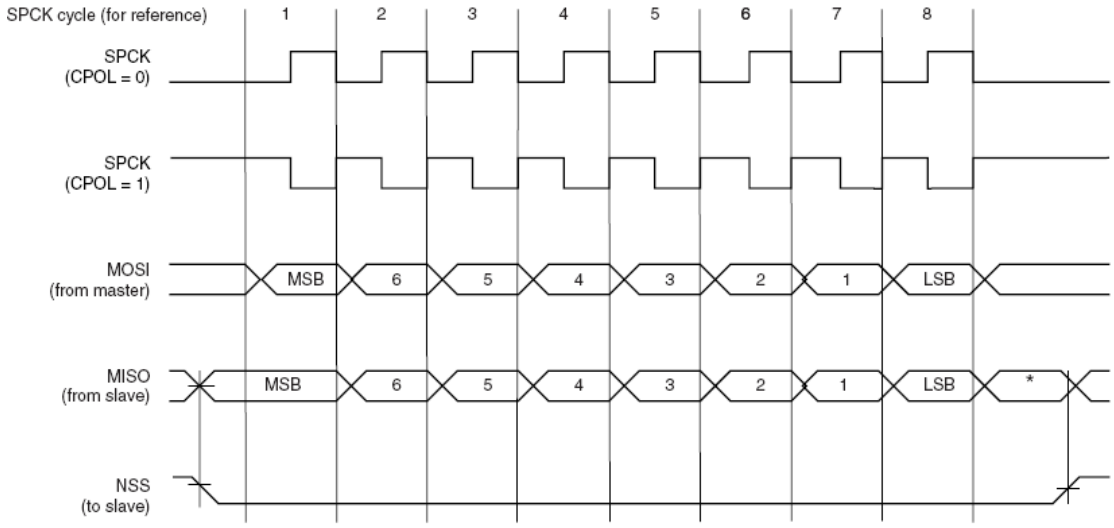
表 29-2 展示了四个模式和对应参数设置。

表 29-2 SPI 总线协议模式

SPI 模式	CPOL	NCPHA
0	0	1
1	0	0
2	1	1
3	1	0

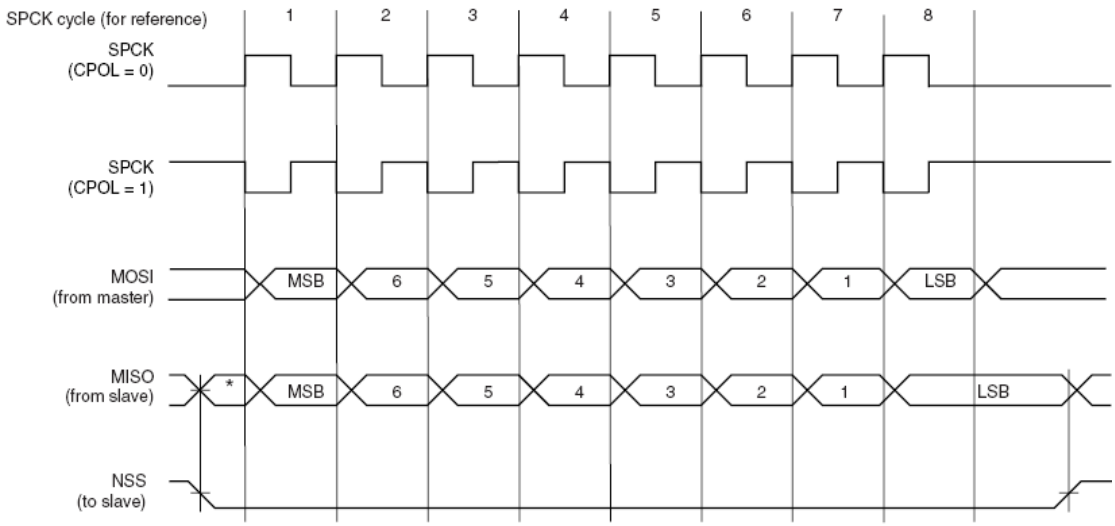
图 29-3 和图 29-4 为数据传输举例

图 29-3 SPI 传输格式（NCPHA = 1，每次传送 8 位）



*未定义，但通常已接收前一字符的 MSB

图 29-4 SPI 传输格式（NCPHA = 0，每次传送 8 位）



*未定义，但通常已发送前一字符的 LSB

29.6.3　主控模式操作

当配置为主控模式，SPI 运行于由内部可编程波特率发生器产生的时钟。完全控制来自和到达连接于 SPI 总线的从设备的数据传输。SPI 驱动片选口线到从设备和串行时钟信号(SPCK)。

SPI 有两个保持寄存器，发送数据寄存器和接收数据寄存器还有一个单移位寄存器。保持寄存器以一个固定速率维持数据流。

使能 SPI 后，当处理器写入 SPI_TDR，开始传输数据。已写数据立即被传输到移位寄存器并且启动 SPI 总线传送。当移位寄存器中的数据被转移到 MOSI 口线上，则 MISO 口线被采样和转移到移位寄存器。无接收则不能产生发送。

写 TDR 前，必须置位 PCS 域来选择一个从设备。

若在传输过程中有新数据写入 SPI_TDR，继续保存直到当前传输完成。接着，接收到的数据从移位寄存器传输到 SPI_RDR，SPI_TDR 中的数据被装载到移位寄存器并且一次新的传输开始。

通过状态寄存器中(SPI_SR)的 TDRE 位（发送数据寄存器空）指示移位寄存器中写入 SPI_TDR 的数据的传输。当新数据被写入 SPI_TDR，此位将被清零。TDRE 位被用来触发发送 PDC 通道。

通过 SPI_SR 寄存器中的 TXEMPTY 标志指示传输结束。若对传输延迟比上次(DLYBCT)，则在上述延迟完成后置位 TXEMPTY。此时可关断主控时钟(MCK)。

通过状态寄存器(SPI_SR)中的 RDRF 位（接收数据寄存器满）指示从 SPI_RDR 中移位寄存器接收到的数据的传输。当读取接收到的数据时，清零 RDRF 位。

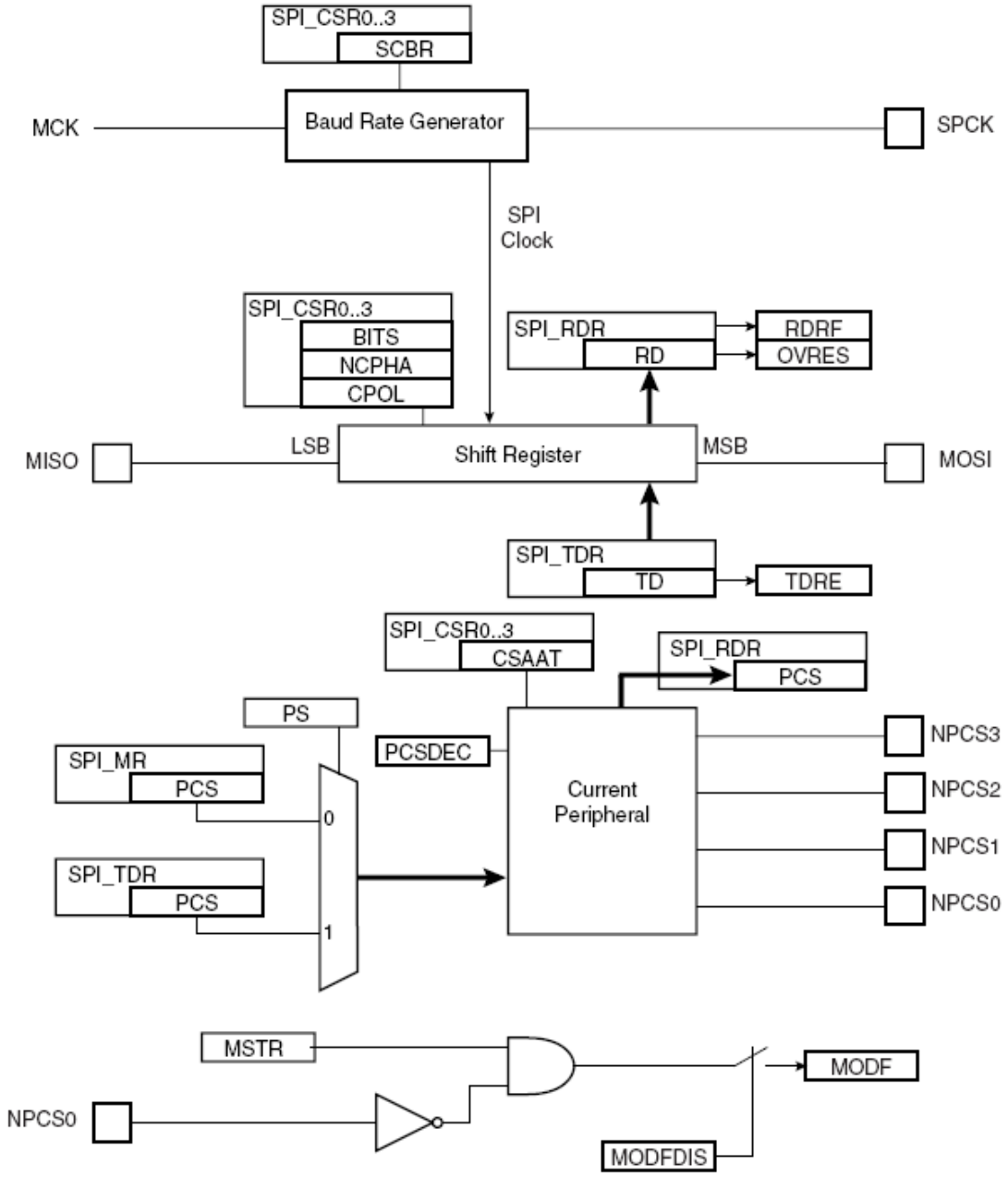
若 SPI_RDR（接收数据寄存器）在新数据被接收到前未被读取，则置位 SPI_SR 中的超限错误位(OVRES)。只要置位此标志位，则将数据装载到 SPI_RDR 中。用户必须读取状态寄存器来清零 OVRES 位。

362 页图 29-6 显示了当运行于主控模式时 SPI 的方块图

362 页图 29-6 显示了描述怎样处理传输的流程图

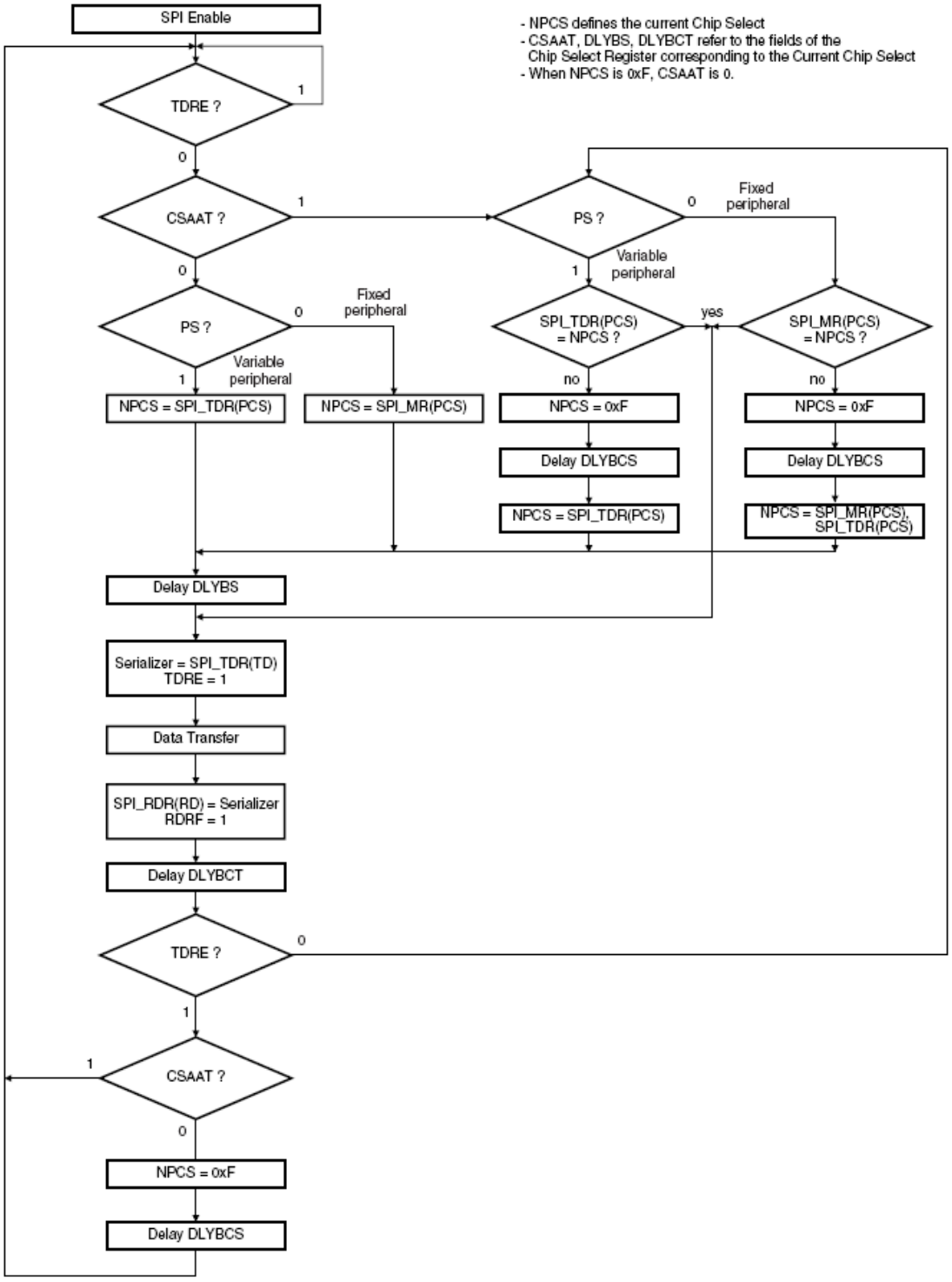
29.6.3.1　主控模式方块图

图 29-5　主控模式方块图



29.6.3.2　主控模式流程图

图 29-6　主控模式流程图



29.6.3.3 时钟的产生

SPI 波特率时钟由主控时钟（MCK）分频产生，被 1 到 255 之间的值分频。

这允许以最大可达主控时钟的波特率运行，以及被 255 分频的 MCK 的最小波特率运行。

禁止编程 SCBR 域为 0。当 SCBR 为 0 时触发传输可导致不可预期的结果。

复位时，SCBR 为 0，用户必须在执行首次传输前以一个有效值编程它。

对每个片选可设置独立的分频器，并必须在片选寄存器的 SCBR 域中编程。这允许 SPI 在未编程的情形下对每个连接的外设自动适应波特率。

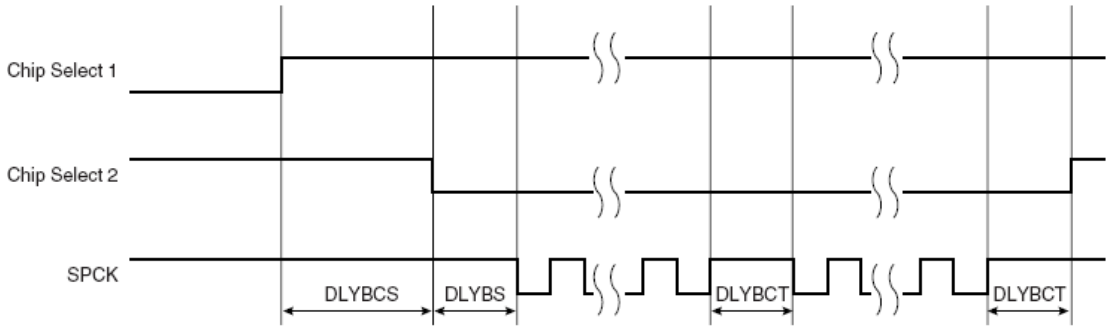
29.6.3.4 传输延迟

图 29-7 显示了片选传输变化和相同片选上的连续传输。可编程三个延迟来修改传输波形：

- 片选间的延迟，仅可对所有片选通过写模式寄存器中的 DLYBCS 域编程一次。允许在一个片选的释放和一个新片选的激活之间插入一个延迟。
- SPCK 前的延迟，可对每个片选通过写 DLYBS 域独立编程。允许在片选激活后启动 SPCK 延迟
- 连续传输间的延迟，可对每个片选通过写 DLYBCT 域独立编程。允许在同一片选上的产生的两次传输间插入延迟。

这些延迟允许 SPI 适应已连接的外设和其速率还有总线释放时间。

图 29-7 可编程延迟



29.6.3.5 外设选择

通过激活 NPCS0 到 NPCS3 信号选择串行外设。通过缺省，每次传输前后所有的 NPCS 信号都是高电平。

可用两种不同的方式执行外设选择：

- 固定外设选择：SPI 仅用一个外设交换数据
- 可变外设选择：可用多个外设交换数据

通过在 SPI_MR（模式寄存器）中对 PS 位写 0 来激活固定外设选择。此情形下，通过 SPI_MR 中的 PCS 域定义当前外设而 SPI_TDR 中的 PCS 域无效。

通过置位 PS 位激活可变外设选择。SPI_TDR 中的 PCS 域用来选择当前外设。这就是说可对每个新数据定义外设选择。

固定外设选择允许缓冲器用一个单外设传输。使用 PDC 是最佳的方法，因为存储器和 SPI 间的数据传输的大小为 8 位或 16 位。然而，改变外设选择需要编程模式寄存器。

可变外设选择在未编程模式寄存器情形下允许缓冲器用多个外设传输。写入 SPI_TDR 中的数据是 32 位宽并定义待发送的实数据和已定义的外设。此模式下使用 PDC 需要 32 位宽的缓冲器，LSB 中的数据，MSB 中的 PCS 和 LASTXFER 域，然而 SPI 仍然用片选配置寄存器通过 MISO 和 MOSI 口线控制传输位数（8 到 16）。这对缓冲器的存储容量来说不是最佳方法，但是它提供一个非常有效的方法在无任何处理器参与的情况下同若干外设交换数据。

29.6.3.6 外设片选解码

用户可通过用外部逻辑解码四个片选口线，NPCS0 到 NPCS3 编程 SPI 运行多达 15 个外设。可通过对模式寄存器（SPI_MR）中的 PCSDEC 位写 1 使能此功能。

当未解码运行时，SPI 确保在任何情况下仅一个片选口线被激活，即每次

驱动为低电平。

当解码运行时，SPI 直接输出由模式寄存器或发送数据寄存器（取决于 PS）中的 PCS 域定义的值。

因为 SPI 在片选口线上设置一个缺省的值 0xF（即所有片选口线为 1）

当未处理任何传输时，仅可解码 15 个外设。

SPI 仅有四个片选寄存器，不是 15 个。所以，当激活解码时，每个片选定义达四个外设的特性。例如，SPI_CRS0 定义外部解码的外设 0-3 的特性，对应 PCS 值 0x0 到 0x3。因此，用户必须确保在解码的片选口线 0 到 3，4 到 7，8 到 11 和 12 到 14 连接兼容的外设。

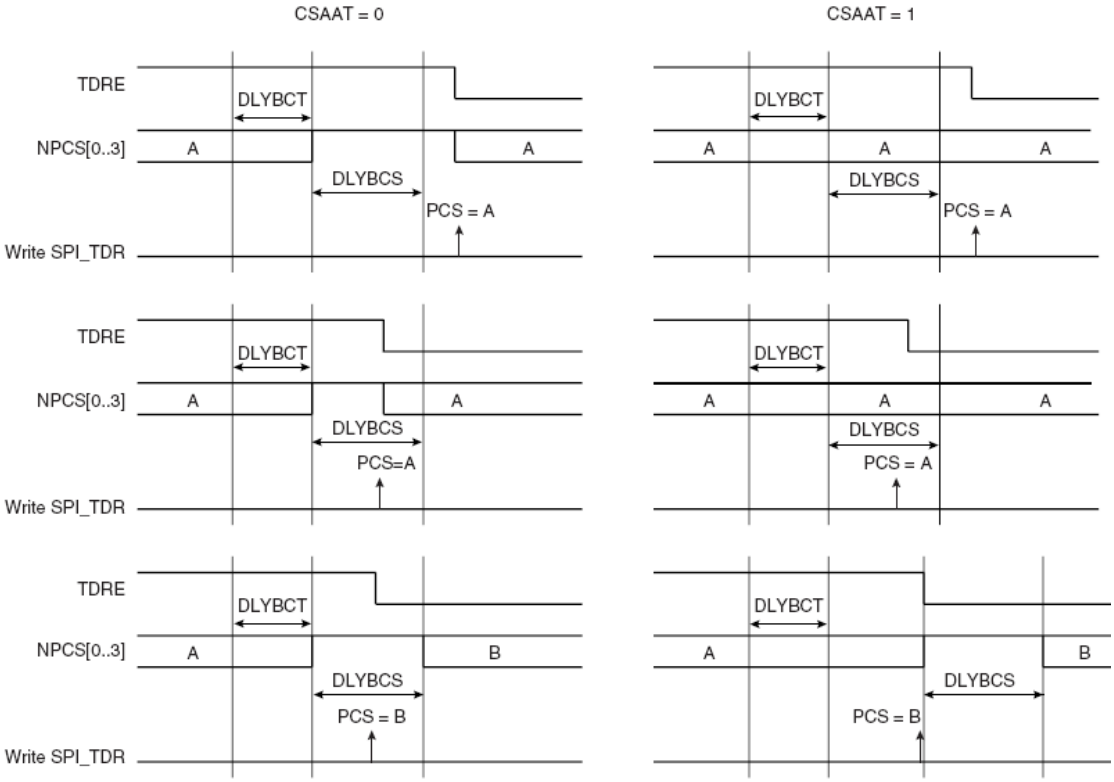
29.6.3.7 外设取消选择

当正常运行时，只要上次写入 SPI_TDR 的数据的传输已完成，则 NPCS 口线都升高。若处理器响应中断太长则可能导致例程错误，并因此在满组传输期间可能导致连接若干串行外设困难，需要片选口线维持有效。

为方便和这样的设备的连接，可用 CSAAT 位（传输后片选有效）置 1 编程片选寄存器。这允许片选口线维持其当前状态（低电平有效）直到需要传输到另一个外设。

图 29-8 显示了不同外设取消选定的情形和 CSAAT 位的结果

图 29-8 外设取消选定



29.6.3.8 模式错误检测

当 SPI 被编程为主控模式并且 NPCS0/NSS 信号上被外部主控拉为低电平时检测到模式错误。NPCS0，MOSI，MISO 和 SPCK 必须通过 PIO 控制器配置为开漏，所以需要保证外部上拉电阻外高电平。

当检测 到一个模式错误，置位 SPI_SR 中的 MODF 位直到读取 SPI_SR，自动禁用 SPI 直到通过对 SPI_CR 中的 SPIEN 位写 1 重新被使能。

通过缺省，使能模式错误检测电路。用户可通过置位 SPI 模式寄存器

(SPI_MR)中的 MODFDIS 位禁用模式错误检测。

29.6.4 SPI 从模式

当运行在从模式下，SPI 处理提供给 SPI 时钟引脚(SPCK)上时钟上的数据位。

SPI 等待 NSS 在从外部主控接收串行时钟前激活。当 NSS 下降，时钟在并串行转换器上生效，并串行转换器处理由片选寄存器 0(SPI_CSR0)的 BITS 域定义的位数。处理了这些位紧接着处理由 SPI_CSR0 的 NCPHA 和 CPOL 位分别定义的一个相位和极性。注意当编程 SPI 为从模式时，其他片选寄存器的 BITS, CPOL 和 NCPHA 无效。

MISO 口线上移出这些位并在 MOSI 口线上采样。

当所有位被处理，传输接收到的数据到接收数据寄存器并且 RDRF 位升高。如果当数据移传输时 RDRF 已经是高电平，则超限位升高并且中止到 SPI_RDR 数据传输。

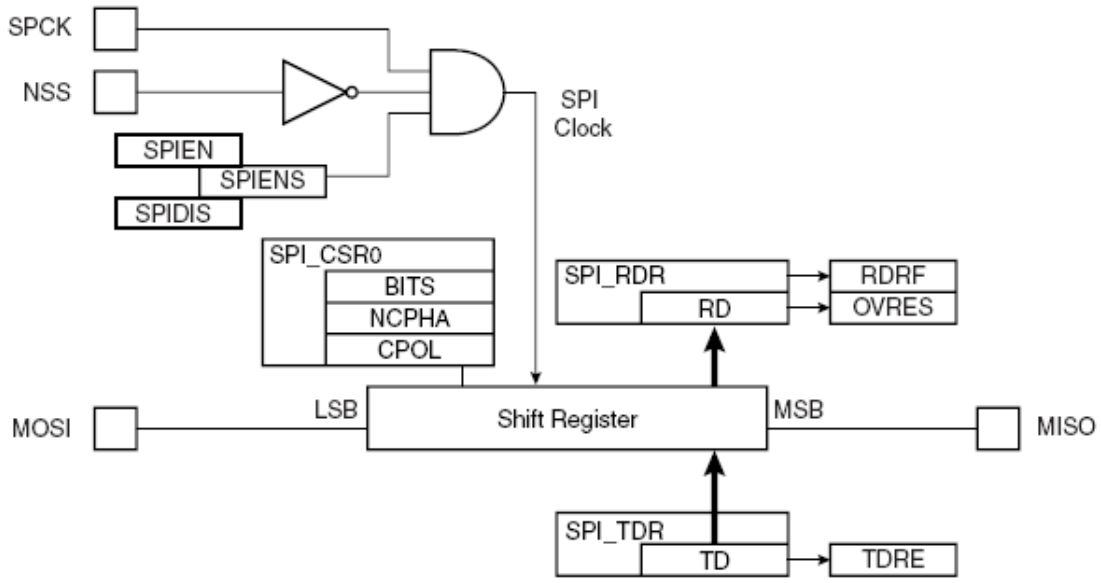
当启动一次传输时，移出的数据是当前移位寄存器中的数据。若无数据被写入发送数据寄存器中(SPI_TDR)，则已传输上次接收到的数据。若自上次复位未接收到数据，则所有位发送位低电平，因为移位寄存器复位值是 0。

当首次数据被写入 SPI_TDR，则立即被传输到移位寄存器中并且 TDRE 位升高。若写入新数据，则保存在 SPI_TDR 中直到开始传输，即，NSS 下降并且在 SPCK 引脚上有一个有效时钟。当传输开始，上次写入 SPI_TDR 的数据被传输到移位寄存器并且 TDRE 位升高。这使得单传输关键变量频率更新可行。

接下来，从发送数据寄存器装载一个新数据到移位寄存器。万一无就绪的字符待发送，即，自上次从 SPI_TDR 装载到移位寄存器，未向 SPI_TDR 写入字符，未修改移位寄存器并且重发送上次接收到的字符。

图 29-9 显示了当 SPI 运行在从模式下时的方框图

图 29-9 从模式功能方框图



29.7　　串行外设接口（SPI）用户接口

表 29-3　SPI 寄存器映射

偏移量	寄存器	寄存器名称	访问类型	复位值
0x00	控制寄存器	SPI_CR	只写	---
0x04	模式寄存器	SPI_MR	读/写	0x0
0x08	接收数据寄存器	SPI_RDR	只读	0x0
0x0C	发送数据寄存器	SPI_TDR	只写	---
0x10	状态寄存器	SPI_SR	只读	0x000000F0
0x14	中断使能寄存器	SPI_IER	只写	---
0x18	中断禁用寄存器	SPI_IDR	只写	---
0x1C	中断屏蔽寄存器	SPI_IMR	只读	0x0
0x20-0x2C	保留			
0x30	片选寄存器 0	SPI_CSR0	读/写	0x0
0x34	片选寄存器 1	SPI_CSR1	读/写	0x0
0x38	片选寄存器 2	SPI_CSR2	读/写	0x0
0x3C	片选寄存器 3	SPI_CSR3	读/写	0x0
0x004C-0x00F8	保留	-	-	-
0x004C-0x00FC	保留	-	-	-
0x100-0x124	为 PDC 保留			

29.7.1　　SPI 控制寄存器

名称：SPI_CR

访问类型：只写

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

- SPIEN：使能 SPI

0=无效

1=使能 SPI 传输和接收数据

- SPIDIS：禁用 SPI

0=无效

1=禁用 SPI

只要 SPIDIS 被置位，则 SPI 就完成其传输

所有引脚在输入模式被置位并且不接收或发送数据。

若正在传输,则在禁用 SPI 前要完成传输。

若当写入控制寄存器时 SPIEN 和 SPIDIS 都是 1，则 SPI 被禁用。

- SWRST：软件复位 SPI

0=无效

1=复位 SPI。执行一个软件触发的 SPI 的硬件复位。

软件复位后，SPI 为从模式。

软件复位不影响 PDC 通道。

- LASTXFER：上次传输

0=无效

1=传输完写入 TD 的字符后激活当前 NPCS。当 CSAAT 置位，就允许用当前串行外设通过拉高对应 NPCS 口线关闭通信，只要 TD 传输已完成。

29.7.2 SPI 模式寄存器

名称：SPI_MR

访问类型：读/写

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

- MSTR：主/从模式

0=SPI 在从模式

1=SPI 在主模式

- PS：外设选择

0=固定外设选择

1=可变外设选择

- PCSDEC：片选解码

0=将片选直接连接到外围设备

1=将四个片选口线连接到一个 4 到 16 位的解码器

当 PCSDEC 为 1 时，则使用一个外部 4 到 16 位的解码器用四个口线产生多达 15 个片选信号。片选寄存器根据如下规则定义 15 个片选的特性：

SPI_CSR0 定义外设片选信号 0-3

SPI_CSR1 定义外设片选信号 4-7

SPI_CSR2 定义外设片选信号 8-11

SPI_CSR0 定义外设片选信号 12-14

- MODFDIS：模式错误检测

0=使能模式错误检测

1=禁用模式错误检测

- LLB：使能本地回送

0=禁用本地回送路径

1=使能本地回送路径

LLB 控制数据并串行转换器上本地回送，仅为测试主控模式。（MISO 内部的连接于 MOSI）

- PCS：外设片选

此域仅在固定外设选择有效（PS=0）时使用。

若 PCSDEC = 0:

PCS = xx01 NPCS[3:0] = 1101

PCS = xxx0 NPCS[3:0] = 1110

PCS = x011 NPCS[3:0] = 1011
PCS = 0111 NPCS[3:0] = 0111
PCS = 1111 禁用（未选择外设）
（x=无关）
若 PCSDEC=1：
NPCS[3:0]输出信号= PCS。

- DLYBCS：片选间的延迟

此域定义从无效的 NPCS 到激活另一个 NPCS 的延迟。DLYBCS 时间确保非重叠片选并且万一外设出现很长的数据浮动时间时解决总线冲突。
若 DLYBCS 小于等于 6 个，通过缺省插入 6 个 MCK 周期。
否则，一下等式决定延迟：

$$\text{片选间延迟} = \frac{DLYBCS}{MCK}$$

29.7.3 SPI 接收数据寄存器

名称：SPI_RDR
访问类型：只读

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

- RD：接收数据

由 SPI 接口接收的数据被右对齐的存储到此寄存器，未使用位读取值为 0

- PCS：外设片选

仅在主控模式下，在一次传输的结尾这些位指示 NPCS 引脚上的值。否则，这些位读取值为 0

29.7.4 SPI 发送数据寄存器

名称：SPI_TDR
访问类型：只写

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

- TD：发送数据

要由 SPI 接口发送的数据被存储到此寄存器中。必须将要发送的信息以右对齐格式写入发送数据寄存器。

PCS：外设片选
此域仅在可变外设选择有效（PS = 1）时使用。
若 PCSDEC = 0:
PCS = xxx0 NPCS[3:0] = 1110
PCS = xx01 NPCS[3:0] = 1101
PCS = x011 NPCS[3:0] = 1011
PCS = 0111 NPCS[3:0] = 0111
PCS = 1111 禁用（未选择外设）
（x=无关）
若 PCSDEC = 1:
NPCS[3:0]输出信号=PCS

- LASTXFER：上次传输

0=无效
1=传输完写入 TD 的字符后激活当前 NPCS。当 CSAAT 被置位，则允许用当前串行外设通过拉高对应 NPCS 口线关闭通信，只要 TD 传输已完成。
仅在可变外设选择有效(PS = 1)时使用此域。

29.7.5 SPI 状态寄存器
名称：SPI_SR
访问类型：只读

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	SPIENS
15	14	13	12	11	10	9	8
–	–	–	–	–	–	TXEMPTY	NSSR
7	6	5	4	3	2	1	0
TXBUFE	RXBUFF	ENDTX	ENDRX	OVRES	MODF	TDRE	RDRF

- RDRF：接收数据寄存器满

0=自上次读取 SPI_RDR 后，未接收数据
1=自上次读取 SPI_RDR 后，已接收数据并且已将接收到的数据从并串行转换器传输到了 SPI_RDR。

- TDRE：发送数据寄存器空

0=数据已写入 SPI_RDR 但还未传输到并串行转换器
1=上次写入发送数据寄存器的数据已传输到并串行转换器
当禁用 SPI 或在复位时，SPITDRE 为 0。SPI 使能命令置此位为 1

- MODF：模式错误出错

0=自上次读取 SPI_SR，未检测模式错误
1=自上次读取 SPI_SR，检测到一个模式错误

- OVRES：超限出错状态

0=自上次读取 SPI_SR，未检测到超限
1=自上次读取 SPI_SR，产生一次超限
自上次读取 SPI_RDR，当从并串行转换器至少两次装载 SPI_RDR 时产生一次超限。

- ENDRX：RX 缓冲器结束

0=自上次写入 SPI_RCR(1) 或 SPI_RNCR(1)，接收计数器寄存器未达到 0
1=自上次写入 SPI_RCR(1) 或 SPI_RNCR(1)，接收计数器寄存器已达到 0

- ENDTX：TX 缓冲器结束

0=自上次写入 SPI_TCR(1) 或 SPI_TNCR(1)，发送计数器寄存器未达到 0
1=自上次写入 SPI_TCR(1) 或 SPI_TNCR(1)，发送计数器寄存器已达到 0

- RXBUFF：RX 缓冲器满

0=SPI_RCR(1) 或 SPI_RNCR(1)除了 0 还有其它值
1=SPI_RCR(1) 和 SPI_RNCR(1)都有一个 0 值

- TXBUFE：TX 缓冲器空

0 = SPI_TCR(1) 或 SPI_TNCR(1) 除了 0 还有其它值
1=SPI_TCR(1) 和 SPI_TNCR(1)有一个 0 值

- NSSR：NSS 上升沿

0=自上次读取，未检测到上升沿
1=自上次读取，产生一次上升沿

- TXEMPTY：发送寄存器空

0=只要将数据写入 SPI_TDR
1=SPI_TDR 和内部移位器空。若已定义传输延迟，在这样的延迟完成后置位 TXEMPTY。

- SPIENS：SPI 使能状态

0=禁用 SPI
1=使能 SPI

注意：1. 物理分配 SPI_RCR, SPI_RNCR, SPI_TCR, SPI_TNCR 到 PDC。

29.7.6 SPI 中断使能寄存器
名称：SPI_IER
访问类型： 只写

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	TXEMPTY	NSSR
7	6	5	4	3	2	1	0
TXBUFE	RXBUFF	ENDTX	ENDRX	OVRES	MODF	TDRE	RDRF

- RDRF：接收数据寄存器满中断使能
- TDRE：SPI 发送数据寄存器空中断使能
- MODF：模式错误出错中断使能
- OVRES：超限出错中断使能
- ENDRX：接收缓冲器结束中断使能
- ENDTX：发送缓冲器结束中断使能
- RXBUFF：接收缓冲器满中断使能
- TXBUFE：发送缓冲器空中断使能
- TXEMPTY：发送寄存器空使能
- NSSR：NSS 上升中断使能

0=无效

1=使能对应中断

29.7.7 SPI 中断禁用寄存器

名称：SPI_IDR

访问类型 ：只写

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	TXEMPTY	NSSR
7	6	5	4	3	2	1	0
TXBUFE	RXBUFF	ENDTX	ENDRX	OVRES	MODF	TDRE	RDRF

- RDRF：接收数据寄存器满中断禁用
- TDRE：SPI 发送数据寄存器空中断禁用
- MODF：模式错误出错中断禁用
- OVRES：超限出错中断禁用
- ENDRX：接收缓冲器结束中断禁用
- ENDTX：发送缓冲器结束中断禁用
- RXBUFF：接收缓冲器满中断禁用
- TXBUFE：发送缓冲器空中断禁用
- TXEMPTY：发送寄存器空禁用
- NSSR：NSS 上升中断禁用

0=无效

1=禁用对应中断

29.7.8 SPI 中断屏蔽寄存器

名称 ：SPI_IMR

访问类型：只读

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	TXEMPTY	NSSR
7	6	5	4	3	2	1	0
TXBUFE	RXBUFF	ENDTX	ENDRX	OVRES	MODF	TDRE	RDRF

- RDRF：接收数据寄存器满中断屏蔽
- TDRE：SPI 发送数据寄存器空中断屏蔽
- MODF：模式错误出错中断屏蔽
- OVRES：超限出错中断屏蔽
- ENDRX：接收缓冲器结束中断屏蔽
- ENDTX：发送缓冲器结束中断屏蔽
- RXBUFF：接收缓冲器满中断屏蔽

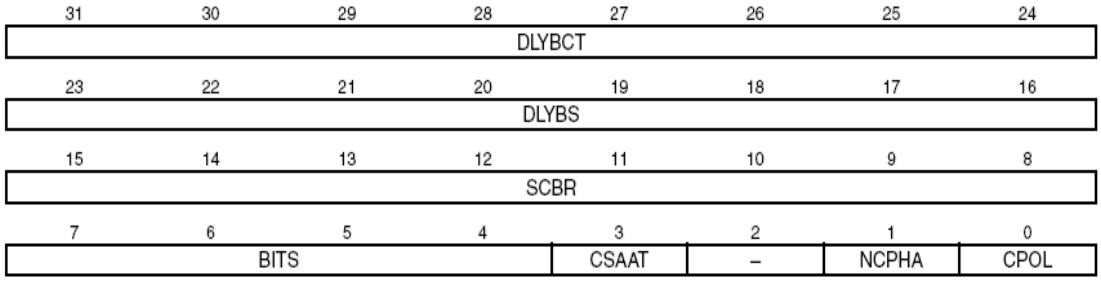
- TXBUFE：发送缓冲器空中断屏蔽
- TXEMPTY：发送寄存器空屏蔽
- NSSR：NSS 上升中断屏蔽

0=对应中断未使能
1=对应中断已使能

29.7.9 SPI 片选寄存器

名称：SPI_CSR0... SPI_CSR3

访问类型：读/写



- CPOL：时钟极性

0=SPCK 的待用状态值是逻辑电平 0
1=SPCK 的待用状态值是逻辑电平 1
使用 CPOL 确定串行时钟(SPCK)的待用状态值。用 CPOL 和 NCPHA 在主控和从设备之间引入需要的时钟/数据关系。

- NCPHA：时钟相位

0=SPCK 前沿上改变数据并且在 SPCK 后沿捕捉到
1=SPCK 前沿上捕捉到数据并且在 SPCK 后沿改变
NCPHA 决定 SPCK 的哪个边沿导致数据改变，哪个边沿导致数据被捕捉到。用 NCPHA 和 CPOL 在主控和从设备之间引入需要的时钟/数据关系。

- CSAAT：传输后片选有效

0=只要上次传输到达则外设片选口线升高
1=上次传输到达后外设片选口线未升高。仍然维持有效直到其它的片选上请求新的传输。

- BITS：每次传输位数

BITS 域决定传输数据的位数，不应使用保留值。

BITS	每次传输的位数
0000	8
0001	9
0010	10
0011	11
0100	12
0101	13
0110	14
0111	15
1000	16
1001	保留
1010	保留
1011	保留

1100	保留
1101	保留
1110	保留
1111	保留

- SCBR：串行时钟波特率

主控模式下，SPI 接口使用模数计数器从主控时钟 MCK 取得 SPCK 波特率。通过向 SCBR 写入一个 1 到 255 之间的值来选择波特率。一下方程决定 SPCK 波特率：

$$\text{SPCK 波特率} = \frac{MCK}{SCBR}$$

禁止将 SCBR 域编程为 0。当 SCBR 为 0 时触发一此传输可能导致不可预期的结果。复位后，SCBR 为 0 并且用户比素在执行首次传输前以一个有效值编程 SCBR。

- DLYBS：SPCK 前延迟

此域定义了从 NPCS 有效到首次有效的 SPCK 传输的延迟

当 DLYBS 等于 0 时，从 NPCS 有效到首次有效的 SPCK 传输为 1/2 个 SPCK 时钟周期。

否则，以下方程决定延迟时间：

$$\text{Delay Before SPCK} = \frac{DLYBS}{MCK}$$

- DLYBCT：连续传输间的延迟

此域定义未调动片选的情形下同一外设两次连续传输间的延迟。

通常在每次传输后和调动片选（若需要）前插入延迟。

当 DLYBCT 为 0，则在连续传输间不插入延迟并且时钟在整个字符传输期间保持其本来的周期。

否则，下式决定此延迟：

连续传输间的延迟=32*DLYBCT/MCK

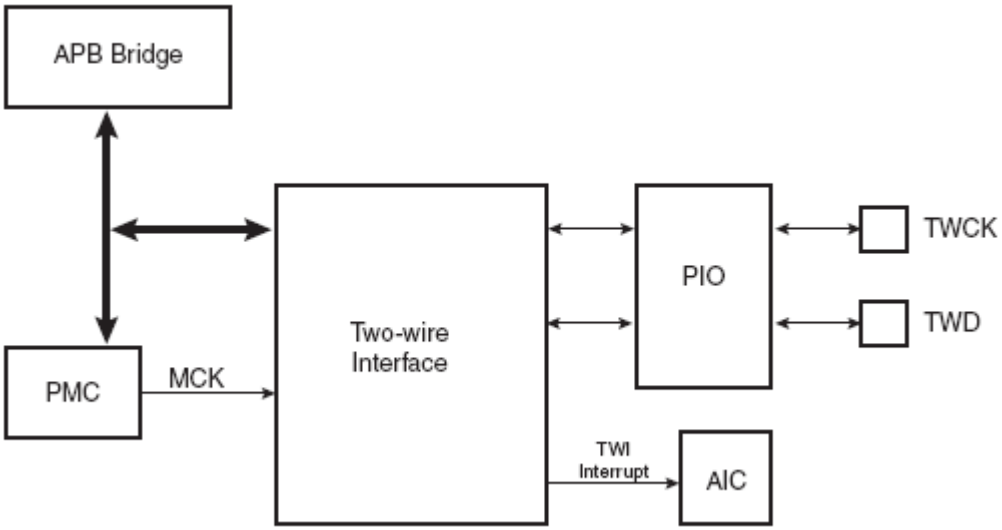
30. 两线接口（TWI）

30.1 描述

两线接口(TWI)将独特的两线总线上的元件互连，基于字节传输格式，由一个时钟线和一个速度达 400K 位每秒的数据线组成。可以和 Atmel 的任何两线总线串行 EEPROM 兼容使用。TWI 编程为可用顺序或单字节访问的主控。一个可配置的波特率发生器允许输出数据速率去适应较宽范围内的内核时钟频率。

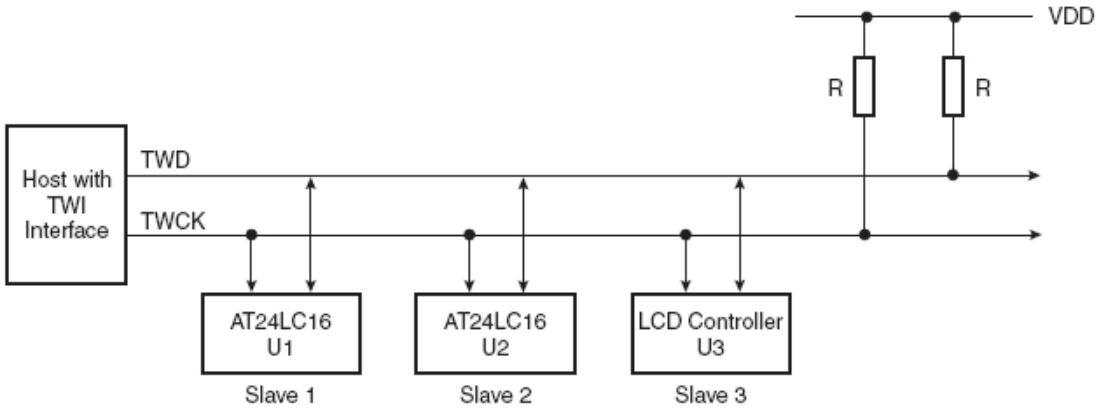
30.2 方框图

图 30-1 方框图



30.3 应用方框图

图 30-2 应用方框图



30.3.1 I/O 口线描述

表 30-1 I/O 口线描述

引脚名称	引脚描述	类型
TWD	两线串行数据	输入/输出
TWCK	两线串行时钟	输入/输出

30.4 相关产品

30.4.1 I/O 口线

TWD 和 TWCK 都是双向口线，通过一个电流源或一个上拉电阻连接于一个正极电压（见 381 页的图 30-2）。当总线空闲，两线都是高电平。连接于总线

的设备输出级必须要有一个开漏或开集电极来执行“线与”功能。

TWD 和 TWCK 引脚可以和 PIO 口线复用。编程者必须执行以下步骤来使能 TWI：

- 编程 PIO 控制器到：
  -指定 TWD 和 TWCK 作为外设口线。
  -定义 TWD 和 TWCK 为开漏

30.4.2 电源管理

- 使能外设时钟

TWI 接口可通过电源管理控制器(PMC)协调同步，因此编程者必须首先配置 PMC 去使能 TWI 时钟。

30.4.3 中断

TWI 接口有一个连接于高级中断控制器(AIC)的中断口线。为了处理中断，必须在配置 TWI 前编程 AIC。

30.5 功能描述

30.5.1 传输格式

放在 TWD 上的数据必须是 8 位长。数据从 MSB 开始传输；每个字节必须跟随一个应答信号。每次传输的字节数是不受限制的（见 383 页的图 30-4）。

每次传输用一个 START 状态开始，用一个 STOP 状态结束（见 383 页图 30-3）。

- 当 TWCK 为高时 TWD 口线上由高到低的转换定义为 START 状态。
- 当 TWCK 为高时 TWD 口线上由低到高的转换定义为 STOP 状态。

图 30-3 START 和 STOP 状态

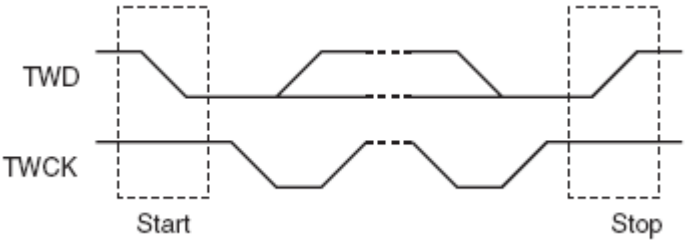
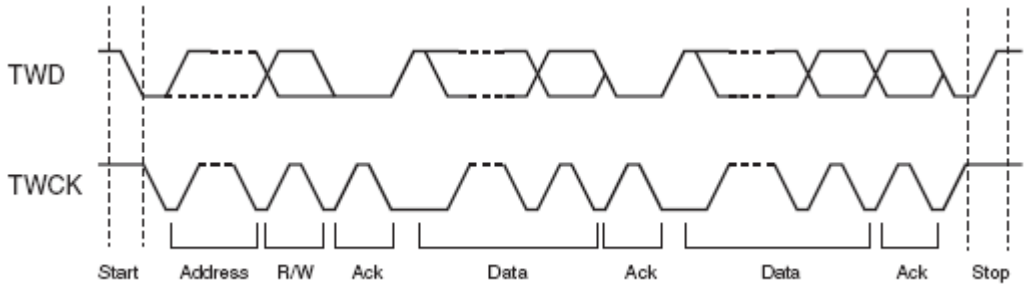


图 30-4 传输格式



30.5.2 操作模式

TWI 由两个操作模式：

- 主控发送模式
- 主控接收模式

主控模式下，允许配置 TWI 控制寄存器(TWI_CR)为接口。此模式下，根据在时钟波形发生器寄存器中编程的值来产生时钟。此寄存器完整的定义了 TWCK

信号，使能接口来适应宽范围的时钟。

30.5.3 数据发送

在主控初始化一个开始状态后，将发送一个在主控模式寄存器(TWI_MMR 中的 DADR)中配置的 7 位的从地址，来通知从设备。紧随从地址的位指示传输方向（读或写）。若此位为 0，则表示一个写操作（发送操作）。若此位为 1，则表示请求读取数据（接收操作）。TWI 传输需要从设备应答每个已接收得字节。在应答时钟脉冲期间，主控释放数据线(HIGH)，使从设备拉低来产生应答。主控在此时钟脉冲期间轮询数据线并且若从设备未应答该字节则置位状态寄存器中的 NAK 位。和其它状态位一样，若使能终端使能寄存器(TWI_IER)则可产生中断。写入发送保持寄存器(TWI_THR)后，置位控制寄存器中的 START 位来启动发送。当监测到应答，将数据移入内部移位器，TXRDY 位为置位状态直到新的数据写入 TWI_THR（见下面的图 30-6）。主控产生一个停止状态结束传输。

通过置位 START 位启动读取顺序。当状态寄存器中 RXRDY 位置位，则在接收保持寄存器(TWI_RHR)接收到一个字符。当读取 TWI_RHR 时将重新复位 RXRDY 位。

TWI 接口执行各种传输格式（7 位从地址，10 位从地址）。可通过主控模式寄存器(TWI_MMR)配置三个内部地址字节。若从设备仅支持 7 位地址，则必须置 IADRSZ 为 0。对高于 7 位的从地址，用户必须配置地址大小(IADRSZ)并置位内部地址寄存器(TWI_IADR)中的其他从地址。

图 30-5 对写主控一个，两个或三个字节内部地址和一个数据字节

图 30-6 对主控写一个字节的内部地址和多个数据字节

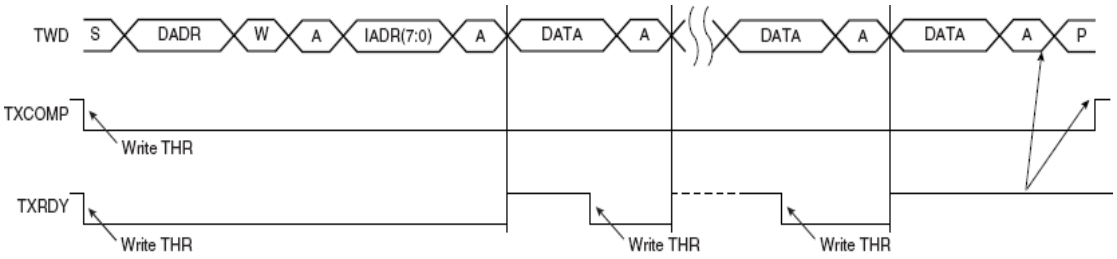


图 30-7 主控读取一个，两个或三个字节的内部地址和一个数据字节

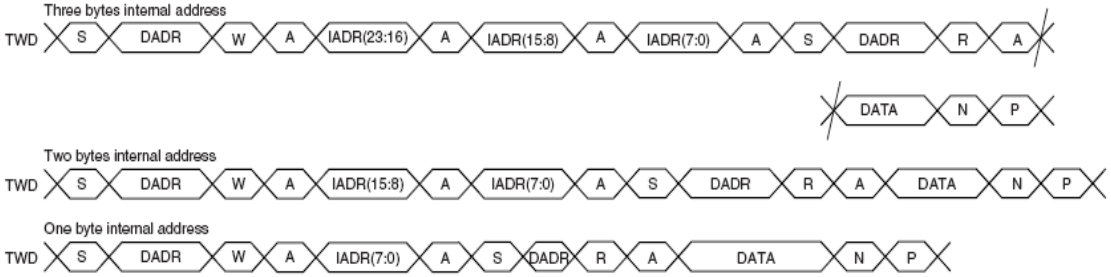
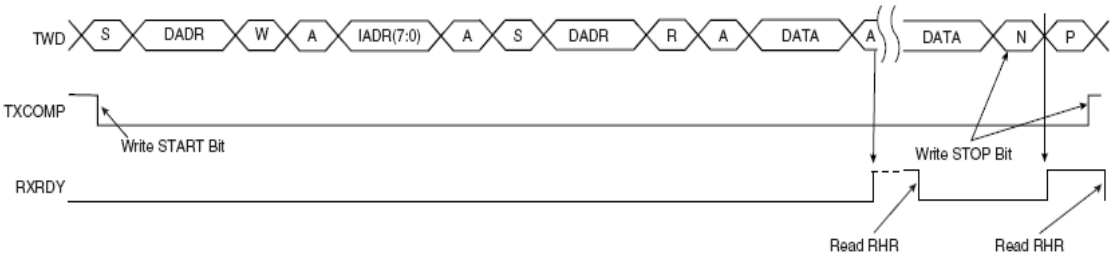


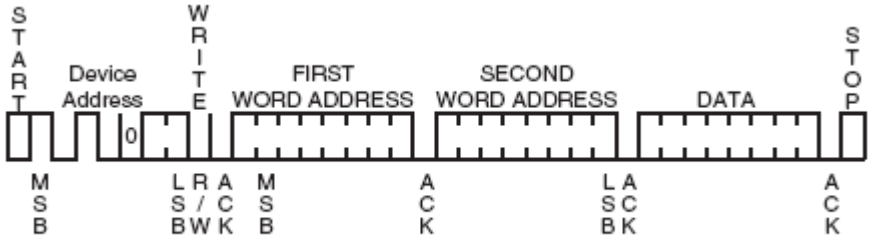
图 30-8　主控读取一个字节的内部地址和多个数据字节



- S = Start（开始）
- P = Stop（停止）
- W = Write（写）
- R = Read（读）
- A = Acknowledge（应答）
- N = Not Acknowledge（未应答）
- DADR= Device Address（设备地址）
- IADR = Internal Address（内部地址）

下图 30-9 示意了向 Atmel AT24LC512 EEPROM 写入一个字节。示范了使用内部地址访问设备。

图 30-9 内部地址的使用



30.5.4　读/写流程图

386 页的图 31-10 和 387 页的图 30-11 流程图给出了主控模式下读写操作的例子。可用轮询或中断方式检查状态位。中断方式需要首先配置中断使能寄存器(TWI_IER)。

图 30-10 主控模式下写 TWI

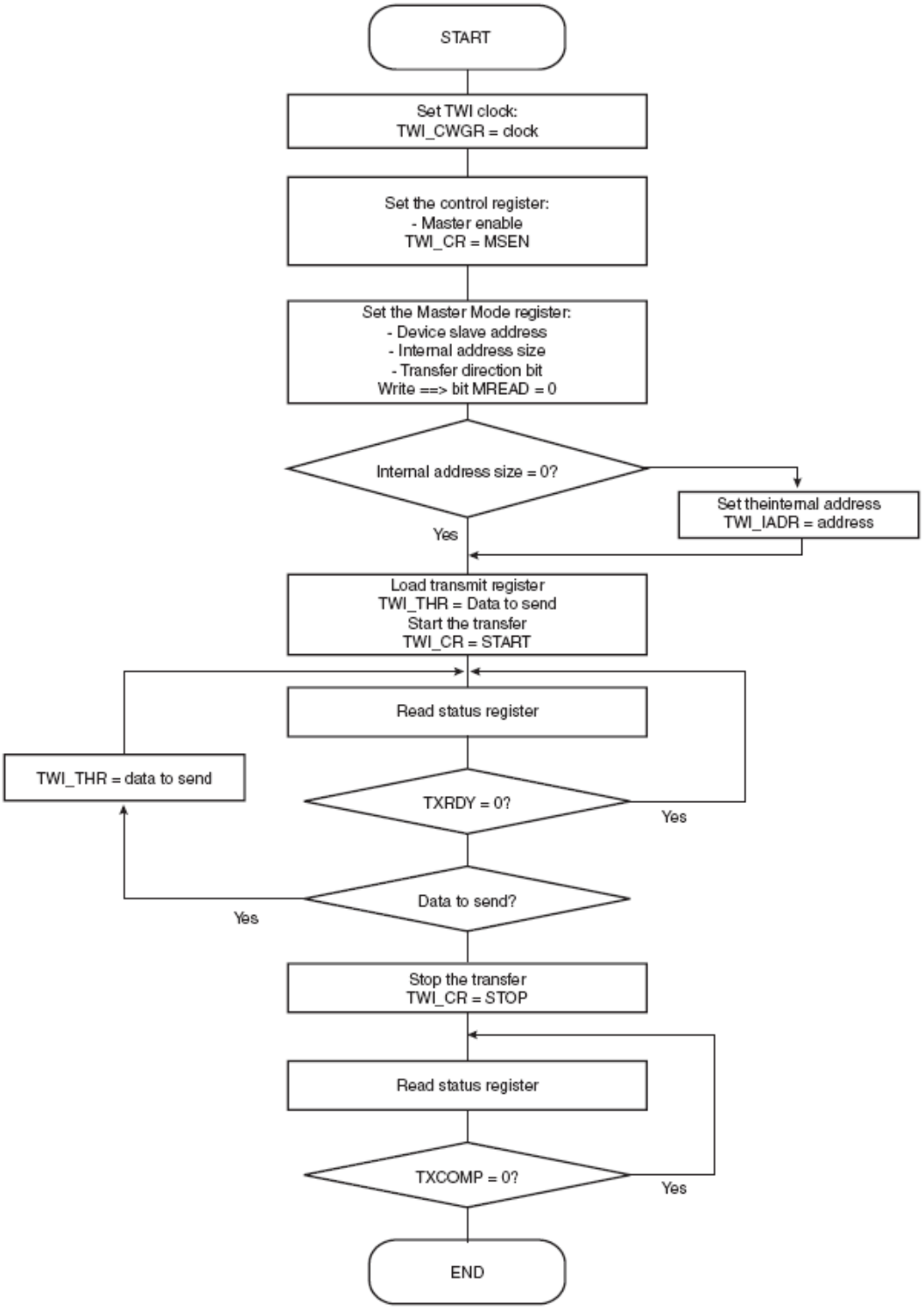
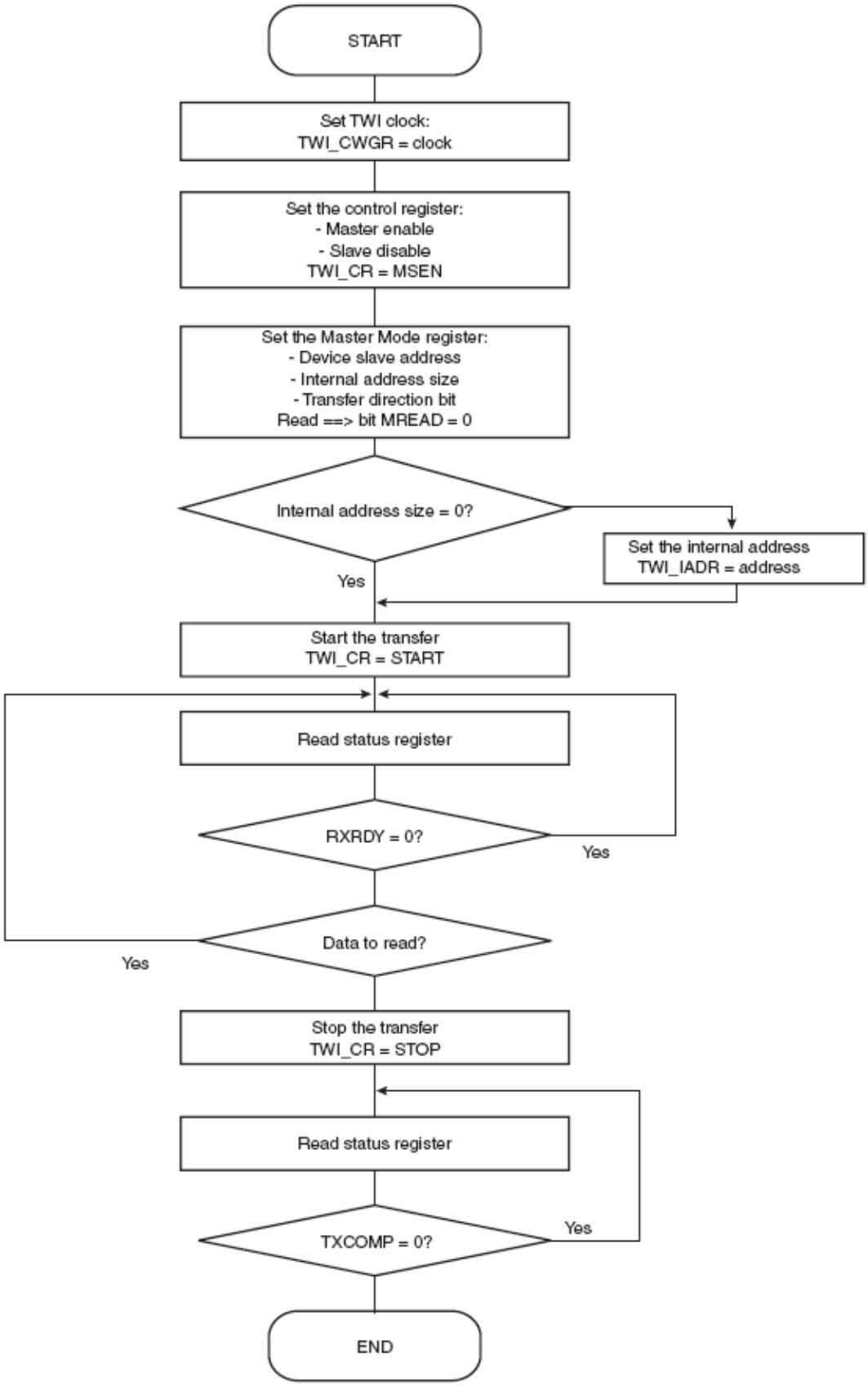


图 30-11 主控模式下读 TWI


START
Set TWI clock:
TWI_CWGR = clock
Set the control register:
- Master enable
- Slave disable
TWI_CR = MSEN
Set the Master Mode register:
- Device slave address
- Internal address size
- Transfer direction bit
Read ==> bit MREAD = 0
Internal address size = 0?
Set the internal address
TWI_IADR = address
Yes
Start the transfer
TWI_CR = START
Read status register
RXRDY = 0?
Yes
Data to read?
Yes
Stop the transfer
TWI_CR = STOP
Read status register
TXCOMP = 0?
Yes
END

30.6　TWI 用户接口

30.6.1　寄存器映射

表 30-2　两线接口(TWI)用户接口

偏移量	寄存器	名称	访问方式	复位值
0x0000	控制寄存器	TWI_CR	只写	N/A
0x0004	主控模式寄存器	TWI_MMR	读/写	0x0000
0x0008	保留	–	–	–
0x000C	内部地址寄存器	TWI_IADR	读/写	0x0000
0x0010	时钟波形发生寄存器	TWI_CWGR	读/写	0x0000
0x0020	状态寄存器	TWI_SR	只读	0x0008
0x0024	中断使能寄存器	TWI_IER	只写	N/A
0x0028	中断禁用寄存器	TWI_IDR	只写	N/A
0x002C	中断屏蔽寄存器	TWI_IMR	只读	0x0000
0x0030	接收保持寄存器	TWI_RHR	只读	0x0000
0x0034	发送保持寄存器	TWI_THR	读/写	0x0000
0x0038-0x00FC	保留	–	–	–

30.6.2　TWI 控制寄存器

寄存器名称：TWI_CR

访问类型：只写

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

- START：发送一个 START 状态

0=无效

1=根据在模式寄存器中定义的特性发送以一个 START 位开始的帧

当 TWI 外设要从一个从设备读取数据时此操作是必要的。当在主控模式下配置一个写操作时，只要在保持寄存器写一个字符，则用模式寄存器发送一帧数据。

- STOP：发送一个 STOP 状态

0=无效

1=在读或写模式下刚完成当前字节发送后发送 STOP 状态

单数据字节主控读或写时，必须设置 START 和 STOP。

多数据字节主控读或写时，在发送 ACK/NACK 位前必须设置 STOP。

在主控读模式，若接收到 NACK 位，则自动的执行 STOP。

多数据写操作时，当 THR 和移位寄存器都空的时候，自动的发送一个 STOP 状态。

- MSEN：TWI 主控传输使能

0=无效

1=若 MSDIS=0，则使能主控数据传输。

- MSDIS：TWI 主控传输禁用

0=无效

1=主控数据传输禁用，所有挂起的数据已发送。执行写操作时发送移位器和保持字符（若他们包含数据）。读操作时，在禁用前必须完整的接收正在传输的字符。

- SWRST：软件复位

0=无效

1=等价于系统复位

30.6.3 TWI 主控模式寄存器

寄存器名称：TWI_MMR

访问类型：读/写

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

- IADRSZ：内部设备地址大小

IADRSZ[9:8]		
0	0	无内部设备地址（字节命令协议）
0	1	单字节内部设备地址
1	0	双字节内部设备地址
1	1	三字节内部设备地址

- MREAD：主控读方向

0=主控写方向

1=主控读方向

- DADR：设备地址

在主控模式下使用设备地址以读或写模式访问从设备。

30.6.4 TWI 内部地址寄存器

寄存器名称：TWI_IADR

访问类型：读/写

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

- IADR：内部地址

0，1，2 或 3 字节取决于 IADRSZ.

-10 位模式地址下，最低位有效字节地址

30.6.5 TWI 时钟波形发生器寄存器

寄存器名称：TWI_CWGR

访问类型：读/写

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

- CLDIV：时钟低分频器

SCL 低周期定义如下：

$$T_{low} = ((\mathrm{CLDIV} \times 2^{\mathrm{CKDIV}}) + 3) \times T_{MCK}$$

- CHDIV：时钟高分频器

SCL 高周期定义如下：

$$T_{high} = ((\mathrm{CHDIV} \times 2^{\mathrm{CKDIV}}) + 3) \times T_{MCK}$$

- CKDIV：时钟分频器

用 CKDIV 来增加 SCL 高周期和低周期

30.6.6 TWI 状态寄存器

寄存器名称：TWI_SR

访问类型：只读

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

- TXCOMP：发送完成

0=主控模式，在当前帧长度期间。从模式下，从接收 START 到接收 STOP。

1=当保持和移位寄存器都空并且 STOP 状态已被发送时（主模式），或当 MSEN 置位（使能 TWI）

- RSRDY：接收保持寄存器就绪

0=从上次 TWI_RHR 读操作未接收到字符。

1=从上次 TWI_RHR 读操作接收到一个字节

- TXRDY：发送保持寄存器就绪

0=发送保持寄存器未被传输到移位寄存器。当写入 TWI_THR 寄存器时置为 0.

1=只要数据字节从 TWI_THR 传输到内部移位器或者若检测到一个 NACK 错误，同时设置 TXRDY 为 TXCOMP 和 NACK。当 MSEN 置位（使能 TWI）时置位 TXRDY。

- NACK：未应答

0=每个数据字节都被远端 TWI 从元件正确的接收到。

1=数据字节未被从元件应答。同时置位为 TXCOMP。读取后复位。

30.6.7 TWI 中断使能寄存器

寄存器名称：TWI_IER

访问类型：只写

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

- TXCOMP：发送完成
- RXRDY：接受保持寄存器就绪
- TXRDY：发送保持寄存器就绪
- NACK：未应答

0=无效

1=使能对应中断

30.6.8 TWI 中断禁用寄存器

寄存器名称：TWI_IDR

访问类型：只写

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

- TXCOMP：发送完成
- RXRDY：接受保持寄存器就绪
- TXRDY：发送保持寄存器就绪
- NACK：未应答

0=无效

1=禁用对应中断

30.6.9 TWI 中断屏蔽寄存器

寄存器名称：TWI_IMR

访问类型：只读

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

- TXCOMP：发送完成
- RXRDY：接受保持寄存器就绪
- TXRDY：发送保持寄存器就绪
- NACK：未应答

0=禁用对应中断

1=使能对应中断

30.6.10 TWI 接收保持寄存器

寄存器名称：TWI_RHR

访问类型：只读

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

- RXDATA：主控或从设备接收保持数据

30.6.11 TWI 发送保持寄存器

寄存器名称：TWI_THR

访问类型：读/写

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

- TXDATA：主控或从设备发送保持数据

31 通用同步/异步收发器

31.1 描述

通用同步/异步收发器(USART)提供一个全双工通用同步异步串行连接。数据帧格式可广泛的编程（数据长度，奇偶校验，停止位数）以适应尽可能多的标准。接收器执行奇偶校验错误，成帧错误和超时错误检测。接收器超时使能可变长帧处理，发送器时间保障方便和慢速远程器件的通信。接收与发送地址位也支持多点通信。

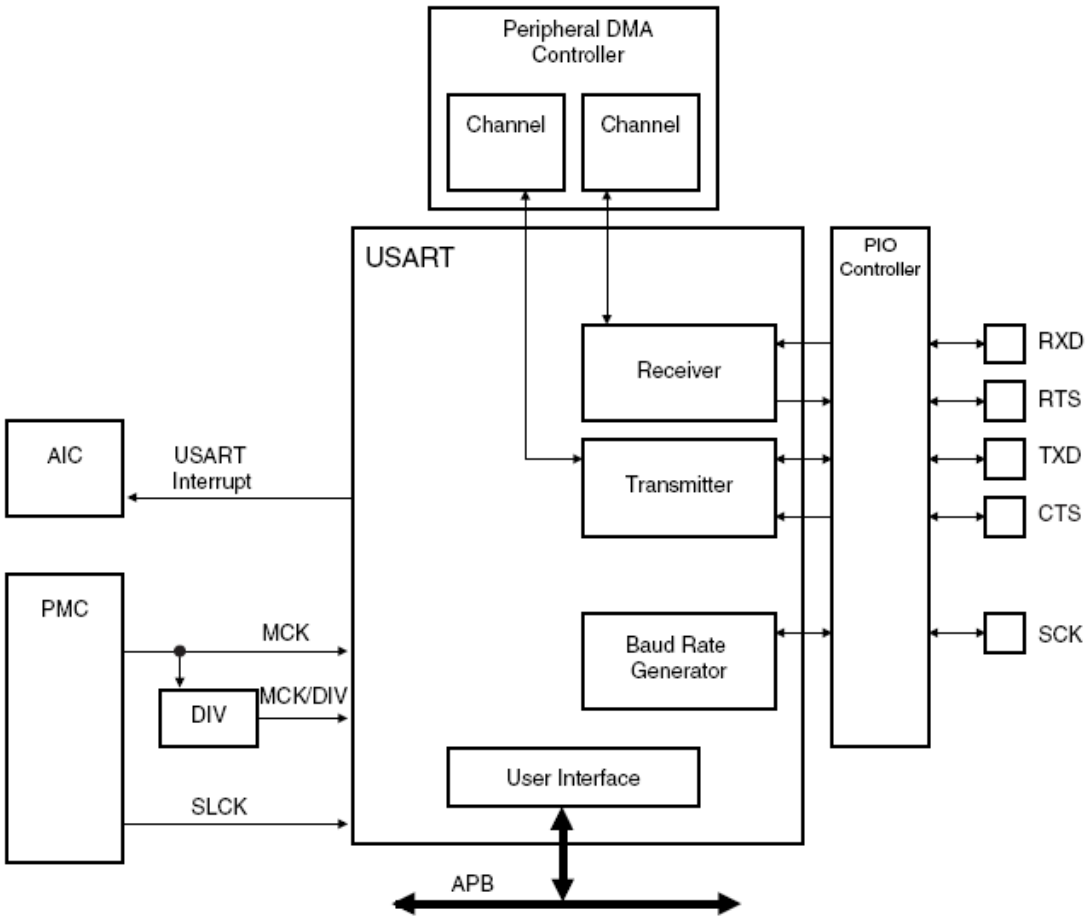
USART 有三个测试模式：远程回环，本地回环，自动回应。

USART 支持提供 RS485 总线接口的特殊操作模式，用 ISO7816T=0 或 T=1 智能卡插槽和红外收发器连接。硬件握手通信通过 RTS 和 CTS 引脚自动管理溢出控制。

USART 支持连接外设 DMA 控制器，外设 DMA 控制器将数据从接收器传输到发送器。PDC 提供无任何处理器干预的链式缓冲器管理。

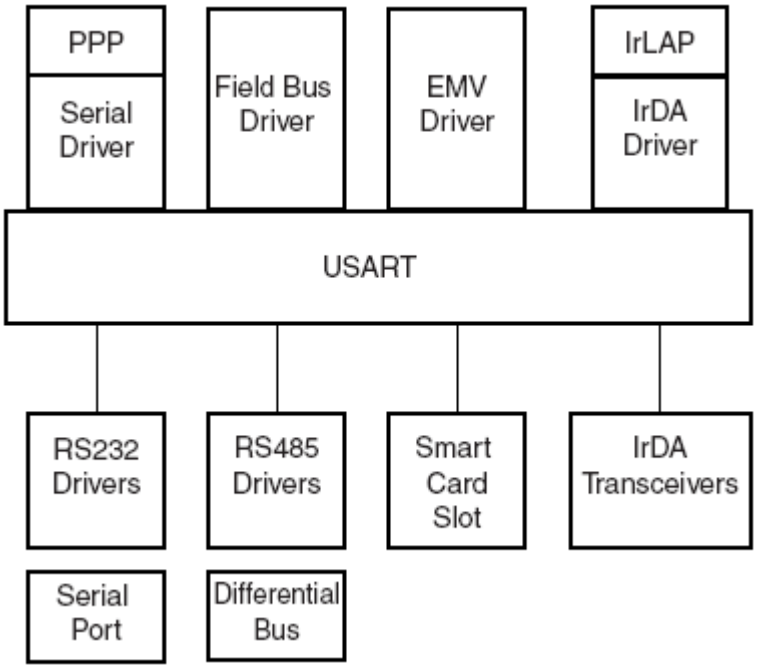
31.2 方框图

图 31-1 USART 方框图



31.3 应用方框图

图 31-2 应用方框图



31.4　I/O 口线描述

表 31-1　I/O 口线描述

名称	描述	类型	有效电平
SCK	串行时钟	I/O	
TXD	发送串行数据	I/O	
RXD	接收串行数据	输入	
CTS	清除发送	输入	低
RTS	请求发送	输出	低

31.5　附属产品

31.5.1　I/O 口线

连接 USART 的引脚可以和 PIO 口线复用。编程者必须首先编程 PIO 控制器来指定想要的 USART 引脚为其外设功能。若应用程序未使用 USART 的 I/O 口线，则它们可被 PIO 控制器用于其它用途。

当 USART 被禁用时，为了从下降沿阻止 TXD 口线，内部上拉电阻的使用是必须的。

31.5.2　电源管理

USART 时钟不连续。编程者必须首先在使用 USART 前使能电源管理控制器中的 USART 时钟。然而，若应用程序不需要 USART 操作，则可以停止 USART 时钟，并在需要的时候重新启动。此时，USART 将继续其停止的操作。

配置 USART 不需要使能 USART 时钟。

31.5.3　中断

USART 中断口线连接到高级中断控制器的内部中断源之一。使用 USART 需要首先编程 AIC。注意不推荐在边沿敏感模式下使用 USART 中断口线。

31.6　功能描述

USART 可以管理若干类型的串行同步或异步通信。

它支持以下通信模式：

- 5 到 9 位全双工异步串行通信
  -高位或低位在先
  -1，1.5 或 2 个停止位
  -偶校验，奇校验，标志，间隔或无
  -接收器频率 8 或 16 倍重采样
  -可选的硬件握手
  -可选的间断管理
  -可选的多点串行通信
- 高速 5 到 9 位全双工同步串行通信
  -高位或低位在先
  -1，或 2 个停止位
  -偶校验，奇校验，标志，间隔或无
  -接收器频率 8 或 16 倍过采样
  -可选的硬件握手
  -可选的间断管理
  -可选的多点串行通信
- 带驱动控制信号的 RS485
- 连接智能卡的 ISO7816,T0 或 T1 协议
  -NACK 处理，带复制和重复限制的错误计数器
- 红外 IrDA 调制和解调
- 测试模式
  -远程回环，本地回环，自动回应

31.6.1 波特率发生器

波特率发生器提供名叫波特率时钟的位周期时钟给接收器和发送器。

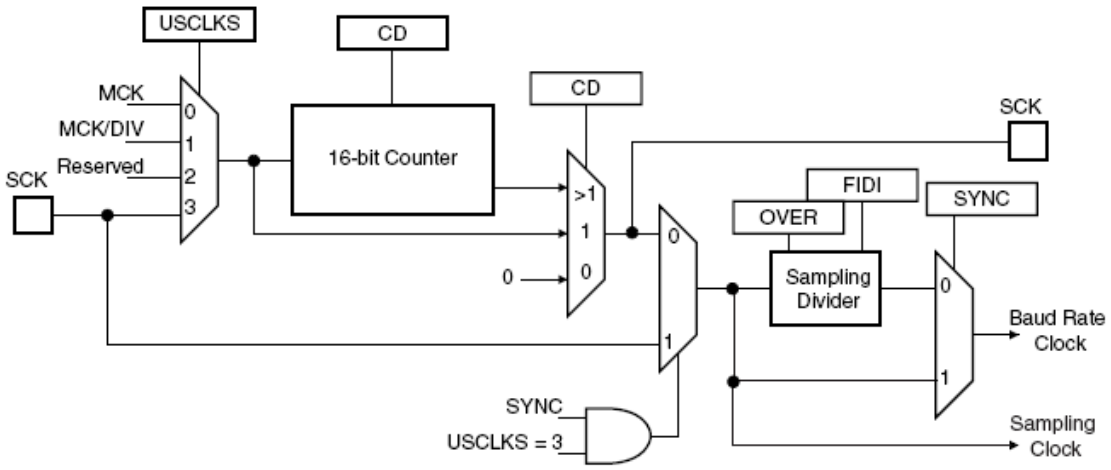
可通过设置模式寄存器(US_MR)中的 USCLKS 域在以下时钟间来选择波特率发生器时钟源：

- 主控时钟 MCK
- 主控时钟分频，分频器和产品相关，通常设置为 8
- 外部时钟，SCK 引脚有效。

波特率发生器基于一个 16 位分频器，用波特率发生器寄存器(US_BRGR)的 CD 域编程此分频器。若编程 CD 为 0，则波特率发生器不产生任何时钟。若编程 CD 为 1，分频器被绕过变为无效。

若选择外部 SCK 时钟，则 SCK 引脚上提供的信号的低电平和高电平的持续时间必须比主控时钟(MCK)周期长。SCK 引脚上提供的信号的频率必须至少比 MCK 的低 4.5 倍。

图 31-3 波特率发生器



31.6.1.1 异步模式下的波特率

若将 USART 编程为异步模式下工作，则选择的时钟首先要除以波特率发生器寄存器(US_BRGR)中已编程的 CD 域的值。合成的时钟被提供给接收器作为采样时钟并接着除以 16 或 8，这取决于对 US_MR 中 OVER 位的编程。

若 OVER 被置为 1，接收器采样时钟为波特率时钟的 8 倍。若 OVER 被清零，采样频率为波特率时钟的 16 倍。

下面的公式计算波特率

$$Baudrate = \frac{SelectedClock}{(8(2-Over)CD)}$$

假设 MCK 为可能的最高的时钟并且 OVER 被编程为 1，这就给出了最大的波特率为 8 分频的 MCK。

波特率计算举例

表 31-2 给出了对不同的源时钟频率计算 CD 值来获得 38400 波特的波特率。此表还给出实际合成的波特率和误差。

表 31-2 波特率举例(OVER = 0)

源时钟	预期的波特率	计算结果	CD	实际波特率	误差
MHz	bit/s			Bit/s	
3 686 400	38400	6.00	6	38 400.00	0.00%
4 915 200	38400	8.00	8	38 400.00	0.00%
5 000 000	38400	8.14	8	39 062.50	1.70%
7 372 800	38400	12.00	12	38 400.00	0.00%
8 000 000	38400	13.02	13	38 461.54	0.16%
12 000 000	38400	19.53	20	37 500.00	2.40%
12 288 000	38400	20.00	20	38 400.00	0.00%
14 318 180	38400	23.30	23	38 908.10	1.31%
14 745 600	38400	24.00	24	38 400.00	0.00%
18 432 000	38400	30.00	30	38 400.00	0.00%
24 000 000	38400	39.06	39	38 461.54	0.16%
24 576 000	38400	40.00	40	38 400.00	0.00%
25 000 000	38400	40.69	40	38 109.76	0.76%
32 000 000	38400	52.08	52	38 461.54	0.16%
32 768 000	38400	53.33	53	38 641.51	0.63%

33 000 000	38400	53.71	54	38 194.44	0.54%
40 000 000	38400	65.10	65	38 461.54	0.16%
50 000 000	38400	81.38	81	38 580.25	0.47%
60 000 000	38400	97.66	98	38 265.31	0.35%
70 000 000	38400	113.93	114	38 377.19	0.06%

用以下公式计算波特率

$$BaudRate = MCK / CD \times 16$$

用以下公式计算波特率误差。建议误差不要高于 5%.

$$Error = 1 - \left(\frac{ExpectedBaudRate}{ActualBaudRate}\right)$$

31.6.1.2 同步模式下的波特率

若 USART 被编程为同步模式下工作，则选择的时钟简单的除以 US_BRGR 中 CD 域的值

$$BaudRate = \frac{SelectedClock}{CD}$$

同步模式下，若选择外部时钟(USCLKS = 3)，则由 USART SCK 引脚上的信号直接提供时钟。无需分频。写入 US_BRGR 中的值无效。外部时钟频率必须至少比系统时钟低 4.5 倍。

当选择外部时钟 SCK 或分频的(MCK/DIV)内部时钟，若用户必须确保 SCK 引脚上信号占空比为 50:50，则 CD 中的编程值必须为偶数。若选择内部时钟 MCK，即使 CD 中的编程值为奇数，波特率发生器仍确保 SCK 引脚信号占空比为 50:50。

31.6.1.3 ISO7816 模式下的波特率

ISO7816 说明用以下公式定义了比特率：

$$B = \frac{Di}{Fi} \times f$$

其中：

- B 为比特率
- Di 是比特率调整因数
- Fi 是时钟频率分频因数
- F 是 ISO7816 时钟频率(Hz)

Di 为 4 位二进制值，称作 DI，如表 31-3 中所示。

表 31-3 Di 中二进制和十进制值

DI 域	0001	0010	0011	0100	0101	0110	1000	1001
Di（十进制）	1	2	4	8	16	32	12	20

Fi 为 4 位二进制值，称作 FI，如表 31-4 中所示。

表 31-4 Fi 中二进制和十进制值

FI 域	0000	0001	0010	0011	0100	0101	0110	1001	1010	1011	1100	1101
Fi（十进制）	372	372	558	744	1116	1488	1860	512	768	1024	1536	2048

表 31-5 给出了合成的 Fi/Di 比，是 ISO7816 时钟和波特率时钟的比值。

表 31-5 可能的 Fi/Di 比值

Fi/Di	372	558	744	1116	1488	1860	512	768	1024	1536	2048
1	372	558	744	1116	1488	1860	512	768	1024	1536	2048
2	186	279	372	558	744	930	256	384	256	768	1024
4	93	139.5	186	279	372	465	128	192	128	384	512
8	46.5	69.75	93	139.5	186	232.5	64	96	64	192	256
16	23.25	34.87	46.5	69.75	93	116.2	32	48	32	96	128
32	11.62	17.43	23.25	34.87	46.5	58.13	16	24	85.33	48	64
12	31	46.5	62	93	124	155	42.66	64	51.2	128	173.6
20	18.6	27.9	37.2	55.8	74.4	93	25.6	38.4		76.8	102.4

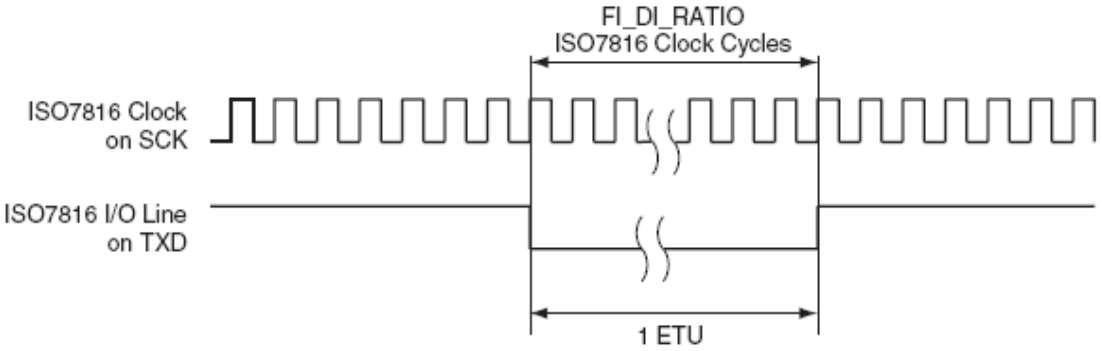
若 USART 被配置为 ISO7816 模式，通过模式寄存器(US_MR)中 USCLKS 域选择的时钟首先除以波特率发生器寄存器(US_BRGR)中 CD 域的编程值。可将合成的时钟提供给 SCK 引脚来供给智能卡时钟输入。这就是说可在 US_MR 中置位 CLKO 位。

可在 FI_DI_比寄存器(US_FIDI)中的 FI_DI_RATIO 域编程来分频此时钟。采样分频器执行此工作，在 ISO7816 模式采样分频器可执行达 2047 的分频。不支持非整数值的 Fi/Di 比并且用户必须编程 FI_DI_RATIO 域为一个尽可能接近期望值的值。

FI_DI_RATIO 域复位值为 0x174（十进制为 372），是 ISO7816 时钟和比特率(Fi = 372, Di = 1)之间最普通的除数。

图 31-4 根据一个比特时间，和 ISO 7816 时钟，展示了基本时间单元间的联系

图 31-4 基本时间单元(ETU)



31.6.2 接收器和发送器控制

复位后，禁用接收器。用户必须通过设置控制寄存器(US_CR)中的 RXEN 位使能接收器。然而，可在接收器时钟被使能前编程接收器寄存器。

复位后，禁用发送器。用户必须设置控制寄存器(US_CR)中的 TXEN 位使能

接收器。然而，可在发送器时钟被使能前编程发送器寄存器。

可一起或独立的使能接收器和发送器

任何时候，软件可分别通过设置控制寄存器中的比特，RSTRX 和 RSTTX 来执行 USART 的接收器或发送器的复位。复位命令作为对应逻辑上的硬件复位有同样的效果。无论接收器或发送器执行了什么，通信将立即被停止。

用户还可通过分别的设置 US_CR 中的 RXDIS 和 TXDIS 独立的禁用接收器或发送器。若在接收一个字符期间禁用了接收器，则 USART 等待直到当前字符的接收结束，然后接受停止。若当发送器正在工作时被禁用，则 USART 等待当前字符和正在被存储到发送保持寄存器(US_THR)中的字符的发送结束。若编程了时间保护器，则可以正常的处理。

31.6.3 同步和异步模式

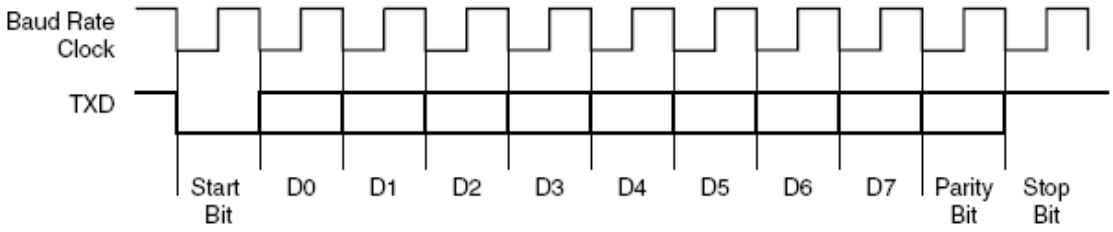
31.6.3.1 发送器操作

发送器在同步和异步操作模式(SYNC = 0 or SYNC = 1)执行同样的工作。在已编程的串行时钟的每个下降边沿，一个起始位，多达 9 个数据位，一个可选择的奇偶校验位和一个达 2 位的停止位被连续的移出 TXD 引脚。

通过模式寄存器(US_MR)中的 CHRL 域和 MODE 9 位选择数据位数。不管 CHRL 域，通过设置 MODE 9 位选择九位。根据 US_MR 中的 PAR 位设置奇偶校验位。可配置偶校验，奇校验，空号，传号或无校验位。US_MR 中的 MSBF 域配置首先发送哪一位。若写 1，则首先发送最高位。写 0，则首先发送最低位。可通过 US_MR 中的 NBSTOP 域选择停止位的位数。在异步模式仅支持 1.5 个停止位。

图 31-5 字符发送

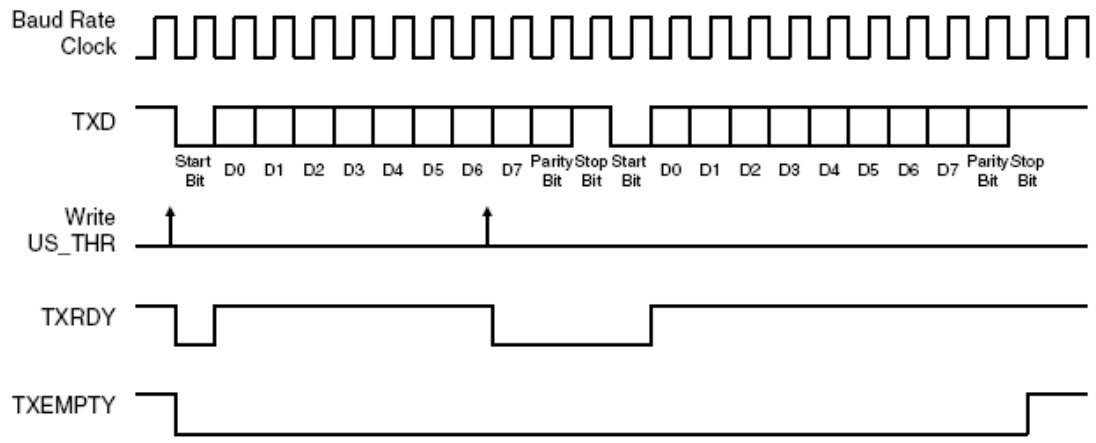
举例：8 位，校验使能，一个停止位



通过写发送保持寄存器(US_THR)发送字符。发送器报告通道状态寄存器(US_CSR)中的两个状态位：TXRDY(发送器就绪)和 TXEMPTY。TXRDY 指示 US_THR 为空，TXEMPTY 指示所有已写入 US_THR 中的字符对已被处理。当当前字符处理已完成，则上次写入 US_THR 的字符被传入发送器的移位寄存器并且 US_THR 变空，因此 TXRDY 升高。

从禁用发送器后，TXRDY 和 TXEMPTY 位都为低电平。当 TXRDY 有效时向 US_THR 写一个字符无效并且写入的字符丢失。

图 31-6 发送器状态



31.6.3.2 异步接收器

若将 USART 编程为异步工作模式(SYNC = 0)，接收器过采样 RXD 输入口线。过采样频率是波特率时钟的 16 或 8 倍，取决于模式寄存器(US_MR)中的 OVER 位。

接收器采样 RXD 位。若在一个半的比特时间期间采样口线值为 0，则将检测到起始位，在比特率时钟上连续的采样到数据，奇偶校验位和停止位。

若采样频率为 16 倍，(OVER 为 0)，在第八次采样时检测到起始位为 0。接着在每 16 个采样时钟周期上连续的采样到数据位，奇偶校验位和停止位。若过采样频率为 8 倍，(OVER 为 1)，在第四次采样时检测到起始位为 0. 接着，在每 8 个采样时钟周期上采样到数据位，奇偶校验位和停止位。

通过相同域和位选择数据位数，第一位发送和奇偶校验方式作为发送器，即，分别为 CHRL, MODE9, MSBF 和 PAR。仅对同步化方法，停止位数对接收器影响因为考虑到仅一个停止位，无论 NBSTOP 域，因此可能发生接收器和发送器间的重新同步化。此外，只要采样到停止位，接收器开始寻找一个新的起始位因此还可能当发送器运行在一个停止位时完成重新同步化。

图 31-7 和图 31-8 阐明了当 USART 运行在异步模式时，开始检测和字符接收

图 31-7 异步开始检测

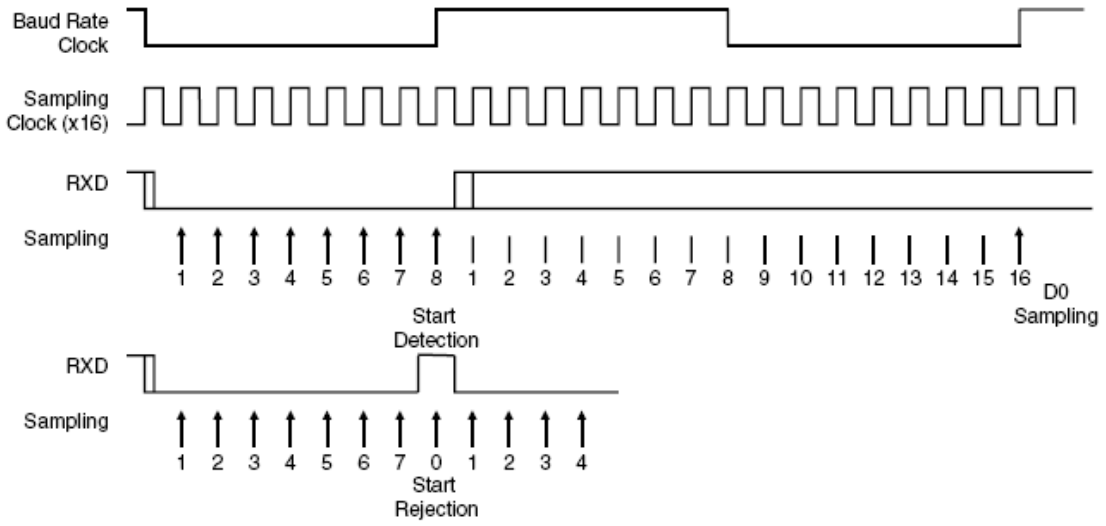
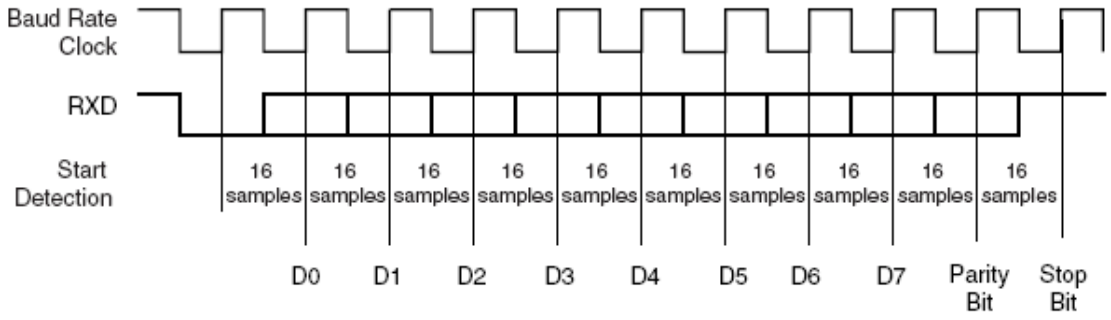


图 31-8　异步字符接收

举例：8 位，奇偶校验使能



31.6.3.3　同步接收器

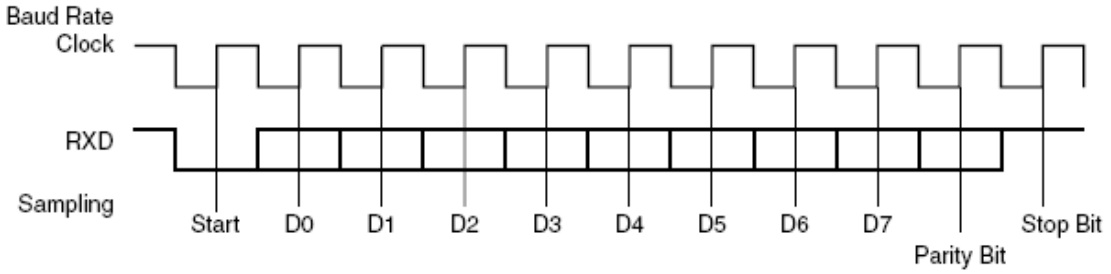
同步模式下(SYNC = 1)，接收器在波特率时钟的每个上升沿采样 RXD 信号。若检测到一个低电平信号，则认为是一个起始位。采样到所有的数据位，奇偶校验位和停止位并且接收器等待下一个起始位。同步模式运行提供了一个高速传输的可能性。

配置域和位和异步模式相同。

图 31-9　阐明了同步模式下一个字符的接收

图 31-9　同步模式字符接收

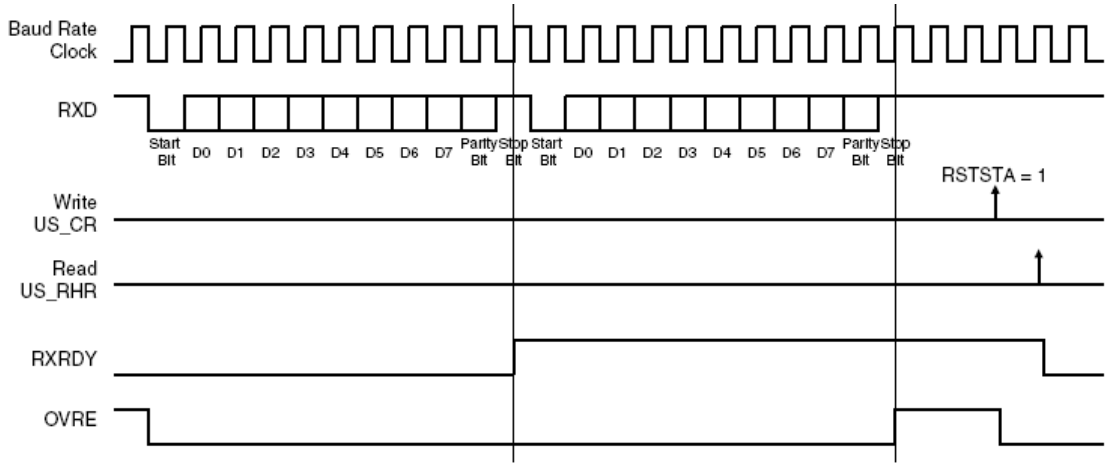
举例：8 位，奇偶校验使能，1 个停止位



31.6.3.4　接收器运行

当完成一个字符的接收，将被传输到接收保持寄存器(US_RHR)并且状态寄存器(US_CSR)中的 RXRDY 位升高。若当 RXRDY 置位时完成一个字符，则 OVRE（超限错误）位被置位。上次的字符被传输到 US_RHR 并且覆盖前一个。通过用 RSTSTA（复位状态）位对控制寄存器(US_CR)写 1 清零 OVRE 位。

图 31-10 接收器状态



31.6.3.5 奇偶校验

USART 支持通过编程模式寄存器(US_MR)中的 PAR 域选择五个奇偶校验模式。PAR 域还使能多点模式，见 410 页“多点模式”。支持奇校验和偶校验位产生和错误检测。

若选择偶校验，如果字符数据位中 1 的个数为偶数，则发送器的校验发生器使校验位为 0，如果 1 的个数为奇数，则校验位为 1。于是，接收器奇偶校验程序计数接收到的 1 的个数并且若采样奇偶位未响应则报告一个奇偶校验错误。若选择奇校验，如果字符数据位中 1 的个数为偶数，则发送器的奇偶校验发生器是奇偶校验位为 1，如果 1 的个数为奇数时校验位为 0.于是，接收器奇偶校验程序计数接收到的 1 的个数并且若采样奇偶位未响应则报告一个奇偶校验错误。如果使用传号奇偶校验，对所有字符发送器的奇偶校验发送器使奇偶位为 1。若奇偶校验位采样值为 0，则接收器奇偶校验程序报告错误。若使用空号奇偶校验，则对所有字符发送器的奇偶校验发生器使奇偶校验位为 0.若奇偶校验位采样值为 1 则接收器奇偶校验检验程序报告错误。若禁用奇偶校验，发送器不产生任何奇偶校验位并且接收器不报告任何校验错误。

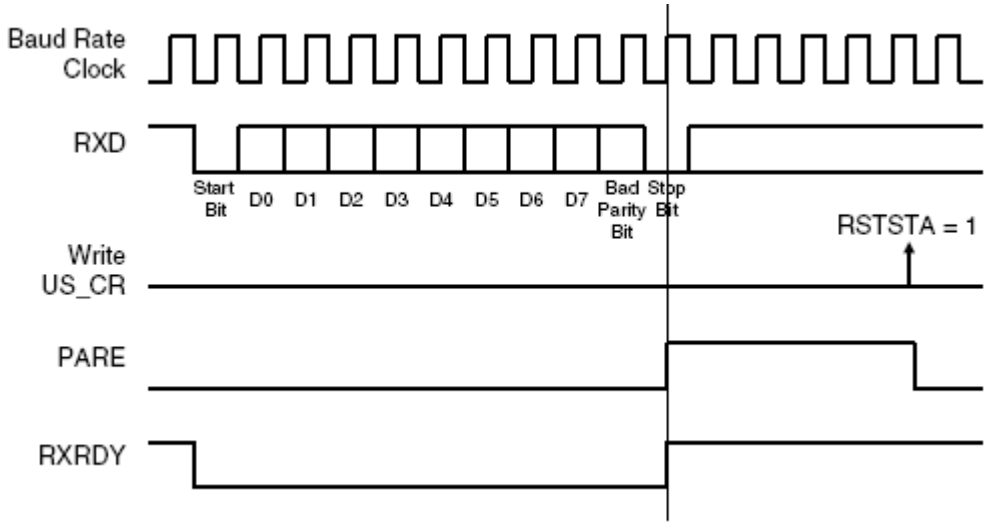
表 31-6 展示了对字符 0x41（字符 ASCII “A”）奇偶校验位取决于 USART 配置的例子。因为有两位为 1，当奇偶校验位为奇数则加一位 1，或当奇偶校验位为偶数则加一个 0

表 31-6 奇偶校验位举例

字符	十六进制值	二进制值	奇偶校验位	校验方式
A	0x41	0100 0001	1	奇校验
A	0x41	0100 0001	0	偶校验
A	0x41	0100 0001	1	传号校验
A	0x41	0100 0001	0	空号校验
A	0x41	0100 0001	无	无

当接收器检测到校验错误，则置位通道状态寄存器(US_CSR)中的 PARE（校验错误）位。可通过用 RSTSTA 位向控制寄存器(US_CR)写 1 来清零 PARE 位。图 31-11 阐明了校验位状态的置位和清零。

图 31-11 校验错误



31.6.3.6 多点模式

若模式寄存器(US_MR)中的 PAR 域编程为 0x6 或 0x07，则 USART 运行在多点模式。此模式区分数据字符和地址字符。发送数据时奇偶校验位为 0，发送地址时奇偶校验位为 1.

若配置 USART 为多点模式，则当奇偶校验位高电平时接收器置位 PARE 奇偶校验错误位，当向控制寄存器的 SENDA 位写 1 时，发送器可以同奇偶校验位高电平发送一个字符。

为处理奇偶校验错误，当向控制寄存器的 RSTSTA 位写 1 时清零 PARE 位。

当 SENDA 被写入 US_CR 时，发送器发送一个地址字节（奇偶校验位置位）。此时，写入 US_THR 的下一个字节作为地址被发送。正常发送写入 US_THR 中未写 SENDA 命令的任何字节时奇偶校验位为 1.

31.6.3.7 发送器时间保护器

时间保护器可用慢速远程设备使能 USART 接口。

时间保护器功能可使能发送器在 TXD 口线上两个字符间插入一个空闲状态。此空闲状态实际上作为一个长停止位运行。

在发送器时间保护起寄存器(US_TTGR)的 TG 域编程空闲状态的延迟时间。当将此域编程为零值时不产生时间保护。否则，在 TG 中编程的比特周期数中除了停止位数外，发送器在 TXD 上每个已发送的字节后保持一个高电平。

如图 31-12 所示，通过编程时间保护器修改 TXRDY 和 TXEMPTY 状态位的状态。TXRDY 仅当下一个字符的起始位已发送时升高，因此若已向 US_THR 写入一个字符则在时间保护发送期间 TXRDY 维持 0 值。TXEMPTY 维持低电平直到完成时间保护发送，因为时间保护是当前正在被发送字符的一部分。

图 31-12 时间保护器操作

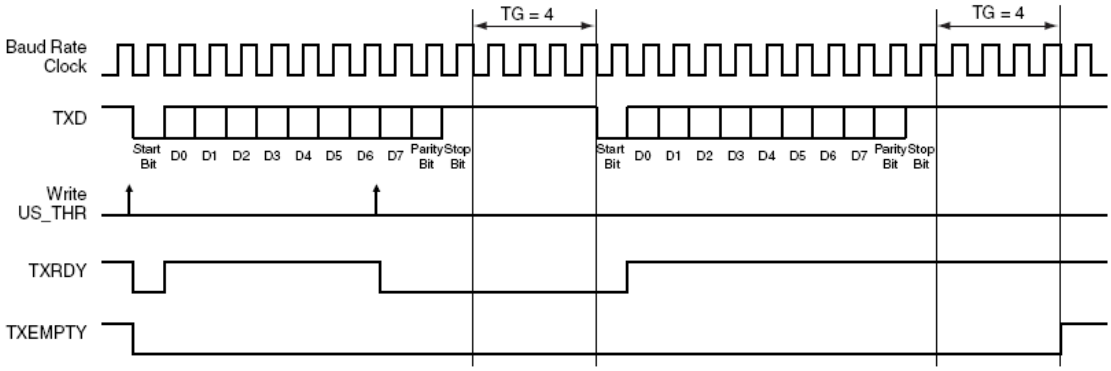


表 31-7 指示了时间保护器周期的最大长度，发送器可以处理相关波特率的功能

表 31-7 最大时间保护长度取决于波特率

波特率	比特时间	时间保护
Bit/sec	μs	ms
1200	833	212.50
9600	104	26.56
14400	69.4	17.71
19200	52.1	13.28
28800	34.7	8.85
33400	29.9	7.63
56000	17.9	4.55

57600	17.4	4.43
115200	8.7	2.21

31.6.3.8 接收器超时

接收器超时提供支持处理各种长度的帧。此功能检测 RXD 线上的空闲状态。当检测到超时，通道状态寄存器(US_CSR)中的 TIMEOUT 位升高并且可产生一个中断，以此向驱动指示一帧数据的结束。

在接收超时寄存器(US_RTOR)的 TO 域中编程超时延迟周期(在此期间接收器等待一个新字符)。若将 TO 域编程为 0，则禁用接收器超时并且检测不到超时。US_CSR 中的 TIMEOUT 位维持 0 值。否则，接收器用 TO 中的编程值来装载 16 位寄存器。此计数器以每个比特周期递减，并每次重新装载接收到的一个新字符。若计数器值到达 0，则状态寄存器中的 TIMEOUT 位升高。接着，用户可以：

- 停止计数器时钟直到接收到一个新字符。可通过向控制寄存器(US_CR)中的 STTTO（启动超时）写 1 执行。此时，接收到一个新字符前 RXD 上的空闲状态将不提供超时。这就阻止了必须在接收到一个新字符前处理以此中断，并且允许在接收到一帧数据后，等待 RXD 上的下一个空闲状态。
- 当未接收到字符时获得一个中断。通过向 US_CR 的 RETTO（重装载和启动超时）位写 1 执行此操作。若执行 RETTO，计数器立即开始从 TO 值向下计数。这就使得能产生周期性的中断所以可以处理用户超时，例如当在键盘上无按键按下时。

若执行 STTTO，则停止计数器时钟直到接收到第一个字符。RXD 上帧启动前的空闲状态不提供超时。这就避免了必须获得一个周期性中断并且当检测到 RXD 上的空闲状态时使能帧尾的等待。

若执行 RETTO，计数器立即开始从 TO 值向下计数。这就使得能产生周期性中断因此可以处理用户超时，例如当键盘上无按键按下时。

图 31-13 展示了接收器超时功能部件的框图

图 31-13 接收器超时框图

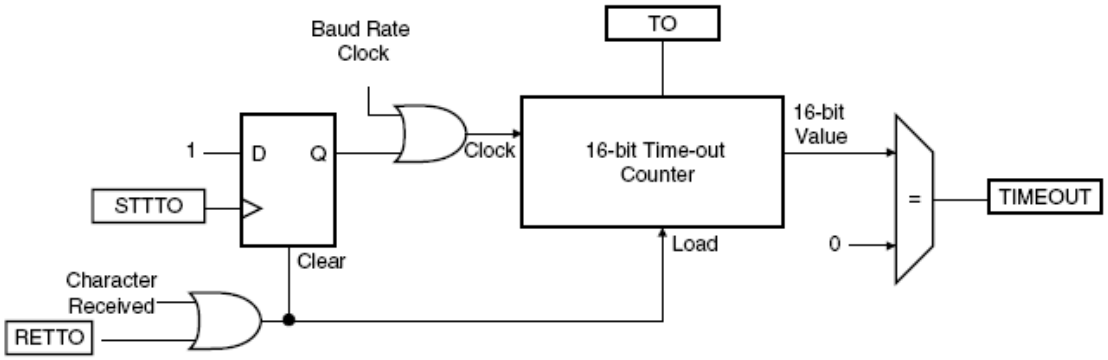


表 31-8　给出了对若干标准波特率的最大超时周期

表 31-8　最大超时周期

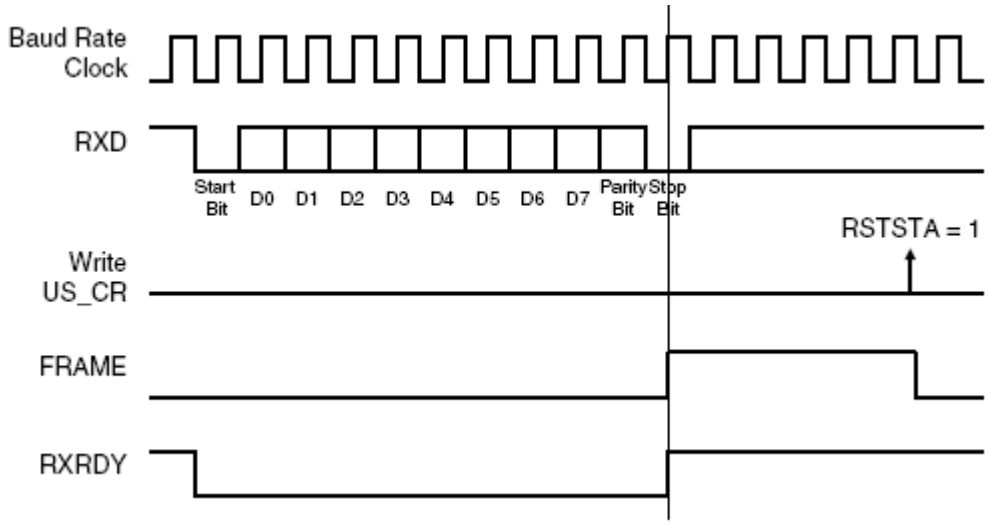
波特率	比特时间	超时
bit/sec	μs	ms
600	1 667	109 225
1 200	833	54 613
2 400	417	27 306
4 800	208	13 653
9 600	104	6 827
14400	69	4 551
19200	52	3 413
28800	35	2 276
33400	30	1 962
56000	18	1 170
57600	17	1 138
200000	5	328

31.6.3.9　成帧错误

接收器能检测到成帧错误。当检测到一个已接收字符的停止位为 0 时产生一个成帧错误。若完全重新同步化接收器和发送器则可能发生这种情况。

将在通道状态寄存器(US_CSR)的 FRAME 位报告成帧错误。只要检测到成帧错误则在停止位的中间激活 FRAME 位。通过向控制寄存器(US_CR)的 RSTSTA 位写 1 来清零 FRAME 位。

图 31-14　成帧错误状态



31.6.3.10　发送停顿

用户可以请求发送器在 TXD 线上产生一个间断状态。间断状态在至少一个完全的字符期间使 TXD 线变低。当一个 0x00 字符和奇偶校验位和停止位值 0 一起发送时也一样。然而，发送器至少在一个字符期间保持 TXD 线直到用户请求删除间断状态。

通过向控制寄存器(US_CR)的 STTBRK 位写 1 来发送一个间断状态。无论当发送器空（在移位寄存器或在 US_THR 中无字符）或当正在发送一个字符时，

可在任何时间执行。若当正在移出一个字符时请求到一个间断，则在 TXD 线保持低前首先完成该字符。

一旦请求 STTBRK 命令，则忽略更远的 STTBRK 命令直到间断末尾完成。

通过向 US_CR 的 STPBRK 位写 1 来删除间断状态。若在小间断持续时间(一个字符，包括起始位，数据，奇偶校验位和停止位）结束前请求 STPBRK，则发送器确保完成间断状态。

即使是一个字符发送器也当作间断，即，仅当 US_CSR 中 TXRDY 位为 1 时考虑 STTBRK 和 STPBRK 命令，并且即使一个已处理的字符，间断状态的起始清零 TXRDY 和 TXEMPTY 位。

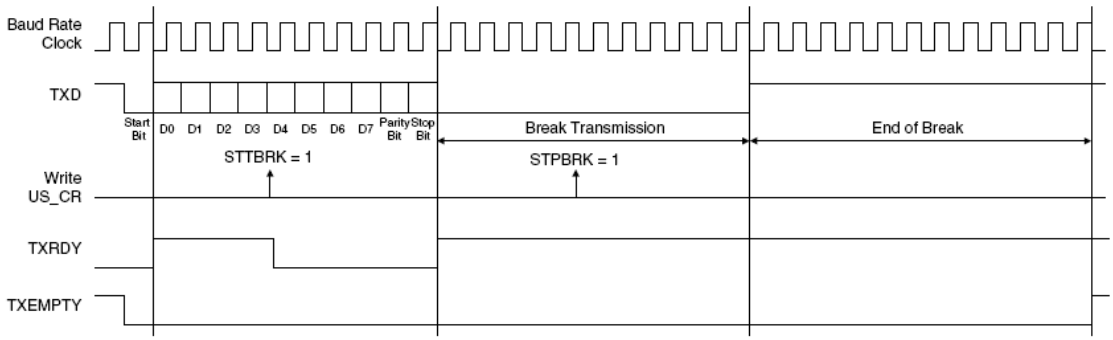
向 US_CR 的 STTBRK 和 STPBRK 位写 1 可导致不可预期的结果。忽略所有无先前的 STTBRK 命令的以请求的 STPBRK 命令。当挂起一个间断时一个写入发送保持寄存器的字节，但未启动启动，则忽略。

间断状态后，发送器对最小的 12 位时间向 TXD 线返回 1。因此，发送器确保远程接收器正确的检测到间断的结尾和下一个字符的起始。若用一个高于 12 的值编程时间保护器，则 TXD 线对时间保护周期保持高电平。

对此周期保持 TXD 线后，发送器回复正常操作。

图 31-15 阐述了 TXD 线上起始间断(STTBRK)和停止间断(STPBRK)命令的效果

图 31-15 间断发送



31.6.3.11 接收间断

当所有数据，奇偶校验位和停止位为低时，接收器检测到间断状态。这相当于用数据 0x00 检测成帧错误，但时 FRAME 维持低电平。

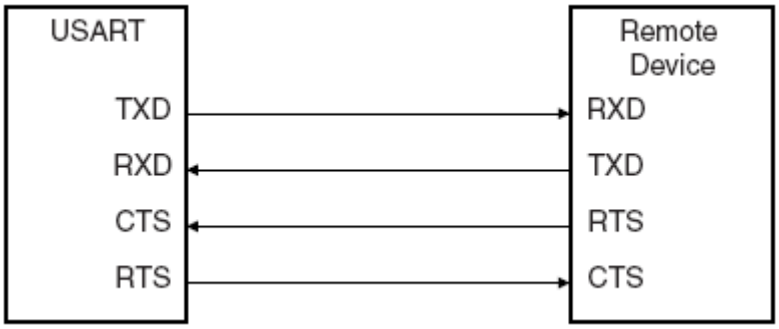
当检测到低电平停止位，接收器激活 US_CSR 中的 RXBRK 位。可通过向控制寄存器(US_CR)的 RSTSTA 写 1 清零该位。

在异步操作模式对至少 2/16 比特周期的一个高电平或在同步操作模式下一个高电平的采样值检测到接收间断结束。间断结束的检测还激活了 RXBRK 位。

31.6.3.12 硬件握手

USART 有一个带外流控制硬件握手。RTS 和 CTS 引脚被用于连接远程设备，如图 31-16 所示。

图 31-16 连接远程设备的硬件握手



通过向模式寄存器(US_MR)中的 USART_MODE 域写 0x2 值执行设置 USART 来同硬件握手运行。

当使能硬件握手时 USART 的动作和标准同步或异步模式下的动作相同，除了接收器驱动 RTS 引脚（如以下描述）和 CTS 上电平修改发送器的动作（如以下描述）。使用此模式需要用 PDC 通道接收。发送器可在任何情况下处理硬件握手。

图 31-17 显示了若使能硬件握手则接收器怎样运行。若禁用接收器并且来自 PDC 通道的 RXBUFF（接收缓冲器满）状态为高则将 RTS 引脚拉高。通常，当其 CTS 引脚（由 RTS 驱动）为高时远程设备不启动发送。只要使能接收器，则 RTS 降低，向远程设备指示可以启动发送。定义一个新缓冲区给 PDC 清零 RXBUFF 状态位，并因此将 RTS 引脚拉低。

图 31-17 当用硬件握手操作时接收器的动作

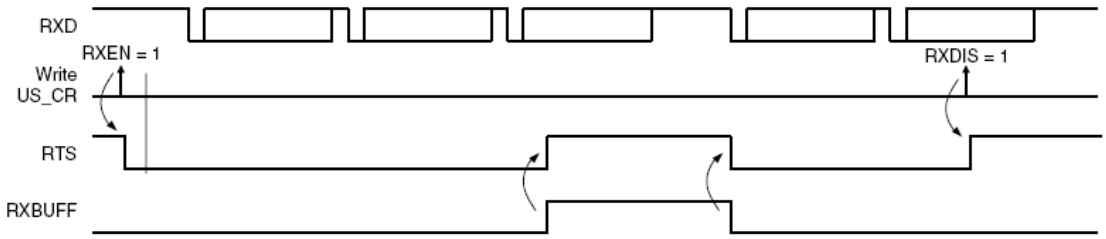
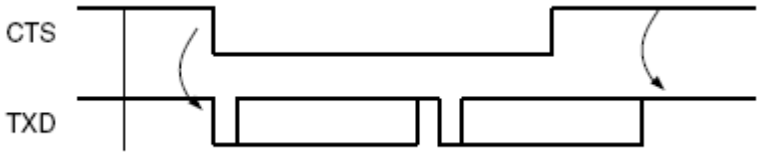


图 31-18 展示了人类使能硬件握手则发送器怎样运行。CTS 禁用发送器。若正在处理一个字符，则仅在当前字符完成后禁用发送器并且只要 CTS 引脚下降则发送下一个字符。

图 31-18 当用硬件握手操作时发送器的动作



31.6.4 ISO7816 模式

USART 有一个兼容 ISO7816 的操作模式。此模式允许接智能卡和通过一个 ISO7816 链接通信的安全访问模块(SAM)。支持由 ISO7816 规定定义的 T=0 和 T=1 两协议。

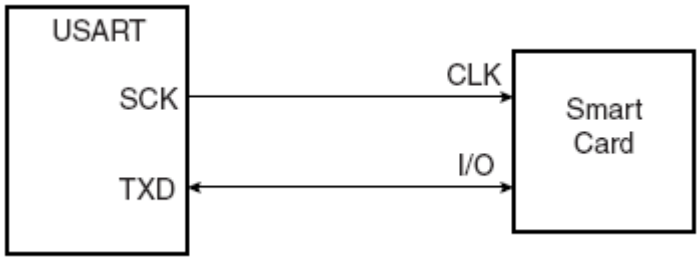
通过向模式寄存器(US_MR)中的 USART_MODE 域写值 0x4（对协议 T=0）和值 0x5（对协议 T=1）来执行 ISO7816 模式下的 USART 设定.

31.6.4.1 ISO7816 模式概述

ISO7816 为仅有的一双向口线上的半双工通信。波特率由为远程设备提供的时钟的分频来确定。（见 401 页“波特率发生器”）

USART 连接一个智能卡，如图 31-19。TXD 口线变为双向并且波特率发生器供给 SCK 引脚上的 ISO7816 时钟。由于 TXD 引脚变为双向，所以其输出仍由发送器输出驱动，但是仅当发送器有效同时其输入指向接收器的輸入。因为 USART 产生时钟所以被认为时通信的主控。

图 31-19 连接智能卡到 USART



当运行在 ISO7816 模式，T=0 或 T=1 的方式，字符格式是固定的。不管在 CHRL,MODE9,PAR 和 CHMODE 域中的编程值，其配置为 8 个数据位，偶校验和 1 或 2 个停止位。MSBF 决定先发送 LSB 或 MSB。校验位(PAR)决定正常模式还是反码发送。参考 427 页“USART 模式寄存器”和 428 页“PAR：校验方式”。

USART 不能并行运行接收器和发送器模式，因为每次通信时单向的。不得不根据需要的模式随需求通过使能或禁用接收器或发送器配置 USART。在 ISO7816 模式下同时使能接收器和发送器可能导致不可预期的结果。

ISO7816 规范定义了一个反码发送格式。必须以其“非”值在 I/O 口线上发送字符的数据位。USART 不支持此模式，用户必须在数据写入发送保持寄存器(US_THR)前或在接收保持寄存器(US_RHR)中读取后对数据异或。

31.6.4.2 Protocol=0

在 T=0 的协议下，一个字符由 1 个起始位，8 个数据位，一个校验位和一个持续两位时间的保护时间组成。在保护时间内发送器移出这些位并且不驱动 I/O 口线。

若未检测到校验错误，则 I/O 口线在保护时间内维持 1 并且发送器下一个字符的发送，如图 31-20 所示。

若接收器检测到一个校验错误，则在保护时间内驱动 I/O 口线为 0 如图 31-21 所示。此错误位对无（Non Acknowledge）应答还叫做 NACK。此时，字符持续 1 的比特时间更长，因为保护时间长度一样并且持续 1 的比特时间加到了错误比特时间上。

当 USART 为接收器并且检测到一个错误，则不会将错误的字符装载到接收保持寄存器(US_RHR)中。将适时的置位状态寄存器中的 PSRE 位使得软件能处理错误。

图 31-20 无校验错误的 T=0 协议

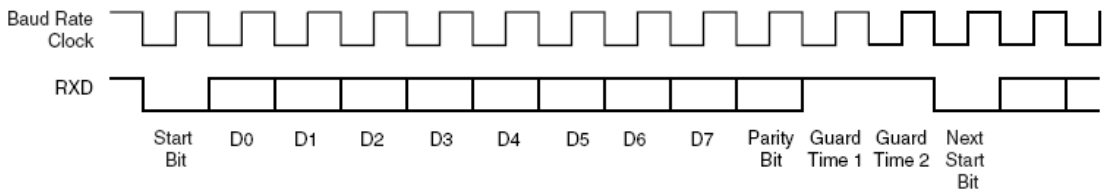
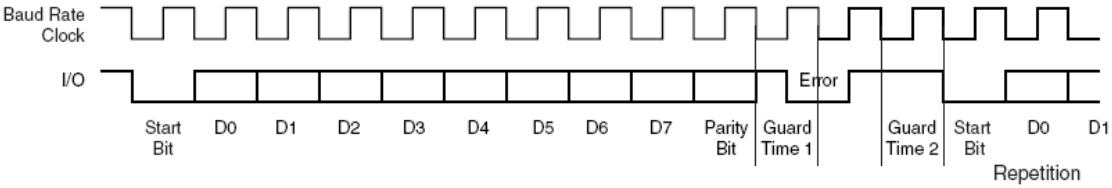


图 31-21 带校验错误的 T=0 协议



接收错误计数器

USART 接收器还记录总错误数。可通过读取寄存器中的错误数实现。NB_ERRORS 域可记录多达 255 个错误。自动的读取 US_NER 来清零 NB_ERRORS 域。

接收 NACK 禁止

还可配置 USART 来禁止错误。可通过置位模式寄存器(US_MR)中的 INACK 位来实现。若 INACK 为 1，则 I/O 口线上无错误信号即使检测到一个校验错误，但是置位状态寄存器(US_SR)中的 INACK 位。可通过向控制寄存器(US_CR)的 RSTNACK 位写 1 来清零 INACK 位。

此外，若置位 INACK，则接收到的错误字符将存储在接收保持寄存器中，就像未发生错误。然而，RXRDY 位不升高。

发送字符复制

当 USART 正在发送一个字符并得到 NACK，则可在移动下一个字符前复制该字符。可通过向模式寄存器(US_MR)的 MAX_ITERATION 域写入一个大于 0 的值来了使能复制。每个字符可重复发送多达 8 次；首次发送加七次副本。

若 MAX_ITERATION 不等于 0，USART 将复制和 MAX_ITERATION 中的装载值一样多的字符。

当 USART 副本数达到 MAX_ITERATION 值，则置位通道状态寄存器(US_CSR)的 ITERATION 位。若接收器应答字符复制，则停止复制并清零重复计数器。

可通过向控制寄存器的 RSIT 位写 1 来清零 US_CSR 中的 ITERATION 位。

禁用连续接收 NACK

接收器可限制发回远程发送器的连续 NACK 数。可通过设置模式寄存器中的 DSNACK 位编程。在 MAX_ITERATION 域中编程发送的最大 NACK 数。只要达到 MAX_ITERATION 值，则认为字符时正确的，向线上发送一个应答并且置位通道状态寄存器中的 ITERATION 位。

31.6.4.3 T=1 协议

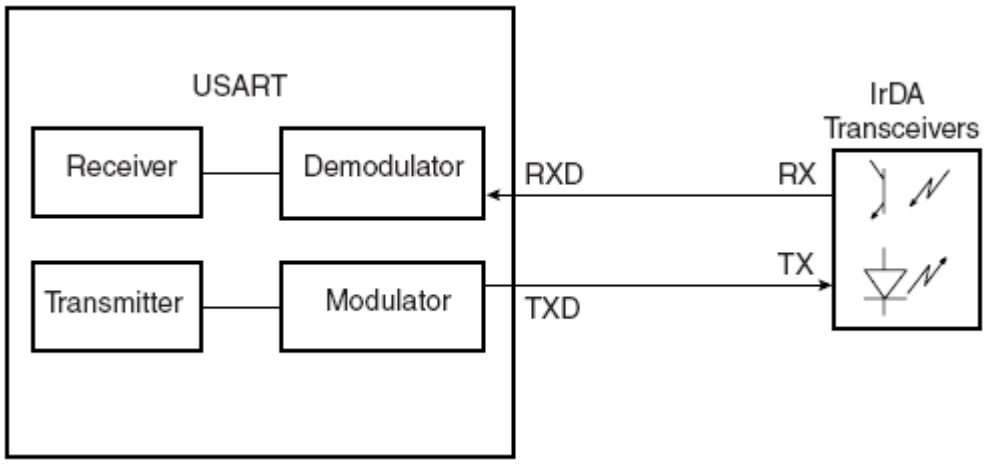
当运行在 ISO7816 协议 T=1，则发送类似于仅一个停止位的异步格式。发送时产生校验接收时检查。校验错误检测置位通道状态寄存器(US_CSR)中的 PARE 位。

31.6.5 IrDA 模式

USART 有支持半双工点对点无线通信的 IrDA 模式。嵌入了允许到红外收发器的无缝连接的调制器和解调器，如图 31-22 所示。调制器和解调器兼容 IrDA 规范 1.1 版并支持范围从 2.4 Kb/s 到 115.2 Kb/s 的数据传输速度。

可通过将模式寄存器(US_MR)中的 USART_MODE 域设置为 0x8 来使能 USART IrDA 模式。IrDA 滤波器寄存器(US_IF)允许配置解调器。USART 发送器和接收器运行在正常模式并且所有参数都是可访问的。注意调制起和解调器要处于有效状态。

图 31-22 IrDA 收发器的连接



必须根据管理的发送方向使能和禁用接收器和发送器。

31.6.5.1 IrDA 调制

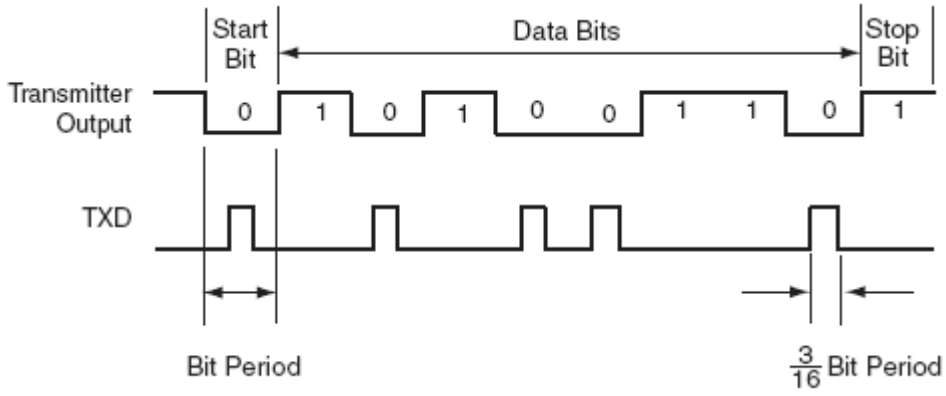
因为波特率达到 115.2 Kbits/sec（包含 115.2 Kbits/sec），所以使用 RZI 调制方案。一比特时间 3/16 的一个轻脉冲代表“0”。信号脉冲持续时间的一些例子如表 31-9

表 31-9 IrDA 脉冲持续时间

波特率	脉冲持续时间（3/16）
2.4 Kb/s	78.13μs
9.6Kb/s	19.53μs
19.2Kb/s	9.77μs
38.4Kb/s	4.88μs
57.6Kb/s	3.26μs
115.2Kb/s	1.63μs

图 31-23 显示了字符发送的一个例子

图 31-23 IrDA 调制



31.6.5.2 IrDA 波特率

表 31-10 给出了 CD 值，波特率错误和脉冲持续时间的一些例子。注意必须满足设备最大的容许误差±1.87%。

表 31-10 IrDA 波特率误差

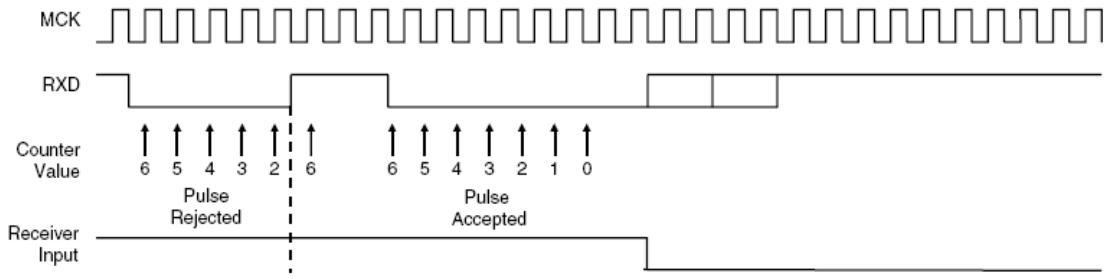
外设时钟	波特率	CD	波特率误差	脉冲时间
3686400	115200	2	0.00%	1.63
20000000	115200	11	1.38%	1.63
32768000	115200	18	1.25%	1.63
40000000	115200	22	1.38%	1.63
3686400	57600	4	0.00%	3.26
20000000	57600	22	1.38%	3.26
32768000	57600	36	1.25%	3.26
40000000	57600	43	0.93%	3.26
3686400	38400	6	0.00%	4.88
20000000	38400	33	1.38%	4.88
32768000	38400	53	0.63%	4.88
40000000	38400	65	0.16%	4.88
3686400	19200	12	0.00%	9.77
20000000	19200	65	0.16%	9.77
32768000	19200	107	0.31%	9.77
40000000	19200	130	0.16%	9.77
3686400	9600	24	0.00%	19.53
20000000	9600	130	0.16%	19.53
32768000	9600	213	0.16%	19.53
40000000	9600	260	0.16%	19.53
3686400	2400	96	0.00%	78.13
20000000	2400	521	0.03%	78.13
32768000	2400	853	0.04%	78.13

31.6.5.3 IrDA 解调器

解调器基于由一个 8 位递减计数器组成的 IrDA 接收滤波器，递减计数器由 US_IF 中的编程值装载。当在 RXD 上检引脚测到一个下降沿，滤波器计数器开始以主控时钟速(MCK)率递减计数。若当计数器到达 0 时在 RXD 上检引脚测到一个上升沿，则在一个比特时间内将接收器输入驱动为低电平。

图 31-24 阐述了 IrDA 解调器的运行

图 31-24 IrDA 解调器的运行

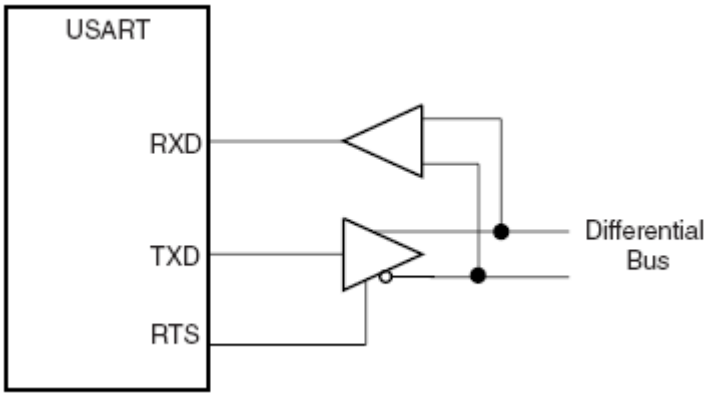


IrDA 模式和 ISO7816 使用相同的逻辑，注意必须将 US_FIDI 中的 FI_DI_RATIO 域设置为大于 0 的值以确保 IrDA 通信正常工作。

31.6.6　RS485 模式

USART 拥有 RS485 模式来使能线启动程序控制。当运行在 RS485 模式，USART 和运行在异步或同步模式相似并且所有参数的配置都是可能的。不同之处在于当发送器运行时 RTS 引脚为高电平。TXEMPTY 位控制 RTS 引脚的动作。一个 USART 上 RS485 总线的典型连接如图 31-25

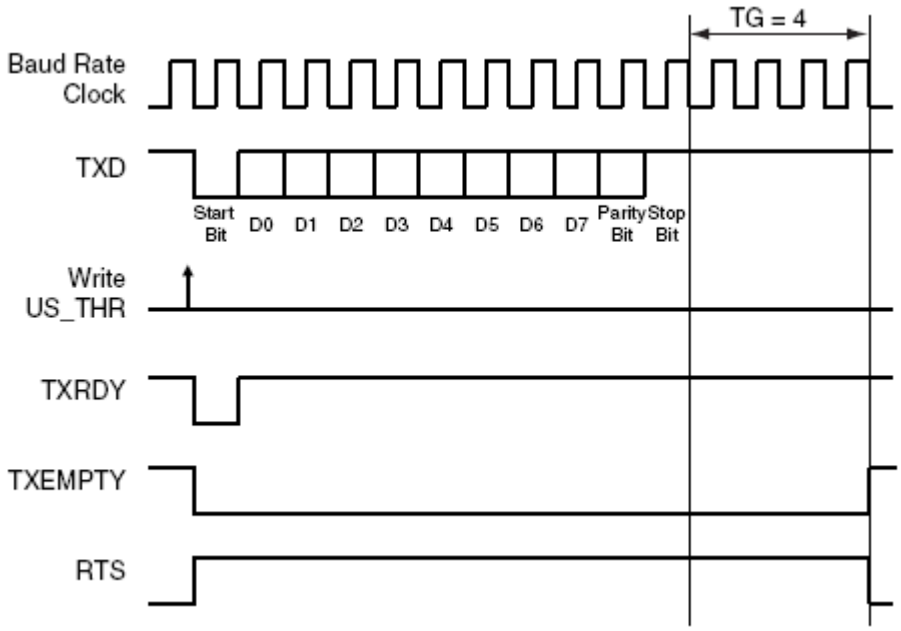
吐 1-25　RS485 总线的典型连接



通过将模式寄存器(US_MR)中 USART_MODE 域编程为 1 来设置 USART 为 RS485 模式。

RTS 引脚电平和 TXEMPTY 位相反。重要的是，当编程了一个时间保护时 RTS 引脚维持高电平，因此在上一字符完成后仍可驱动口线。图 31-26 给出了当使能时间保护时在一个字符发送期间 RTS 波形的例子。

图 31-26 带时间保护的 RTS 驱动举例



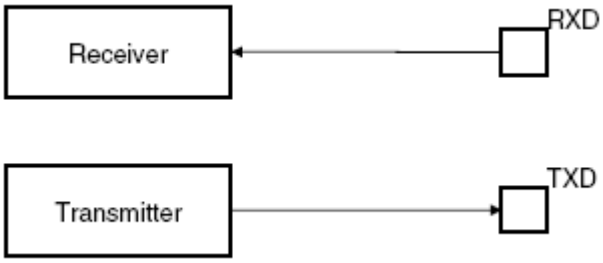
31.6.7　测试模式

可编程 USART 运行在三种不同的测试模式。内部回环功能允许板上诊断。回环模式下断开或未断开 USART 接口引脚并对回环内部的或外部的重新配置。

31.6.7.1 正常模式

正常模式将 RXD 引脚连接接收器输入将 TXD 引脚连接发送器输出。

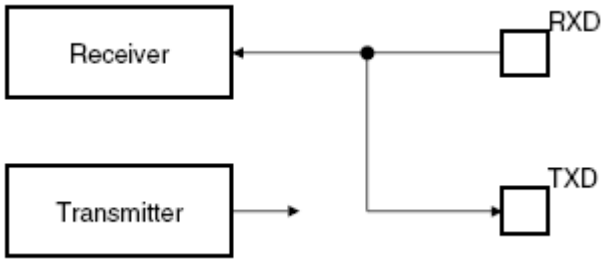
图 31-27 正常模式配置



31.6.7.2 自动回应模式

自动回应模式允许逐位重新发送。当在 RXD 引脚上接收到一个比特，则将其发送到 TXD 引脚，如图 31-28 所示。编程发送器对 TXD 引脚无影响。RXD 引脚仍然连接接收器输入，因此接收器维持有效。

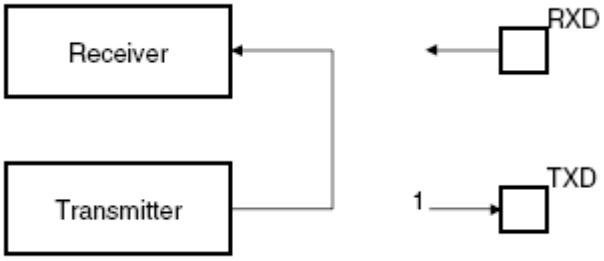
图 31-28 自动回波模式的配置



31.6.7.3 本地回环模式

本地回环模式将发送器的输出直接连接到接收器的输入，如图 31-29。未使用 TXD 和 RXD 引脚。RXD 引脚对接收器无影响，TXD 引脚被连续的拉高，如空闲模式。

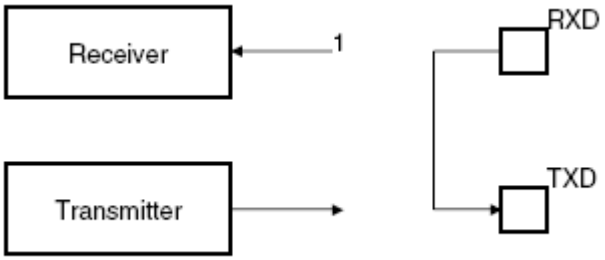
图 31-29 本地回环模式的配置



31.6.7.4 远程回环模式

远程回环模式直接连接 RXD 引脚和 TXD 引脚，如图 31-30 所示。禁用发送器和接收器并且无任何影响。此模式允许逐位重新发送。

图 31-30 远程回环模式的配置



31.7　USART 用户接口

表 31-11 USART 存储器映射

偏移量	寄存器	名称	访问类型	复位值
0x0000	控制寄存器	US_CR	只写	-
0x0004	模式寄存器	US_MR	读/写	-
0x0008	中断使能寄存器	US_IER	只写	-
0x000C	中断禁用寄存器	US_IDR	只写	-
0x0010	中断屏蔽寄存器	US_IMR	只读	0x0
0x0014	通道状态寄存器	US_CSR	只读	-
0x0018	接收器保持寄存器	US_RHR	只读	0x0
0x001C	发送器保持寄存器	US_THR	只写	-
0x0020	波特率发生器寄存器	US_BRGR	读/写	0x0
0x0024	接收器超时寄存器	US_RTOR	读/写	0x0
0x0028	发送器时间保护寄存器	US_TTGR	读/写	0x0
0x2C-0x3C	保留	-	-	-
0x0040	FI DI 比寄存器	US_FIDI	读/写	0x174
0x0044	错误数寄存器	US_NER	只读	-
0x0048	保留	-	-	-
0x004C	IrDA 滤波器寄存器	US_IF	读/写	0x0
0x5C - 0xFC	保留	-	-	-
0x100 - 0x128	位 PDC 寄存器保留	-	-	-

31.7.1　USART 控制寄存器

名称：US_CR

访问类型：只写

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	RTSDIS	RTSEN	–	–
15	14	13	12	11	10	9	8
RETTO	RSTNACK	RSTIT	SENDA	STTTO	STPBRK	STTBRK	RSTSTA
7	6	5	4	3	2	1	0
TXDIS	TXEN	RXDIS	RXEN	RSTTX	RSTRX	–	–

- RSTRX：复位接收器

0：无效

1：复位接收器

- RSTTX：复位发送器

0：无效

1：复位发送器

- RXEN：使能接收器

0：无效

1：若 RXDIS 为 0，使能接收器

- RXDIS：禁用接收器

0：无效

1：禁用接收器

- TXEN：使能发送器

0：无效

1：若 TXDIS 为 0，使能发送器

- TXDIS：禁用发送器

0：无效

1：禁用发送器

- RSTSTA：复位状态位

0：无效

1：复位 US_CSR 中的 PARE，FRAME，OVRE，和 RXBRK 状态位

- STTBRK：启动间断

0：无效

1：在 US_THR 中当前字符已由发送移位寄存器发送则在字符后启动一个间断。若间断正在被发送则无效

- STPBRK：停止间断

0：无效

1：最少一个字符长度时间并在 12 位周期内发送一个高电平后停止间断发送。若无间断正在被发送则无影响。

- STTTO：启动超时

0：无效

1：协调超时计数器前开始等待字符。复位 US_CSR 中的 TIMEOUT 状态位

- SENDA：发送地址

0：无效

1：仅在多点模式下，发送写入 US_THR 的下一个字符同时置位地址位。

- RSTIT：重复次数复位

0：无效

1：复位 US_CSR 中的 ITERATION，若未使能 ISO7816 则无效

- RSTNACK：复位无应答

0：无效

1：复位 US_CSR 中的 NACK

- RTSEN：请求发送使能

0：无效

1：将 RTS 引脚驱动为 0

- RTSDIS：请求发送禁用

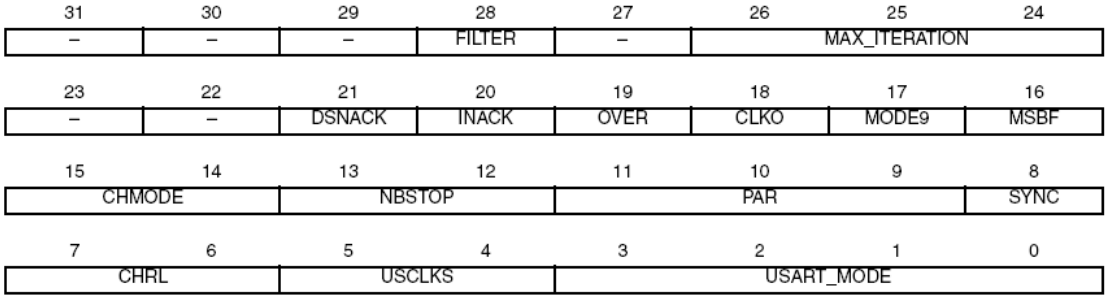
0：无效

1：将 RTS 引脚驱动为 1

31.7.2 USART 模式寄存器

名称：US_MR

访问类型：读/写



- USART_MODE

USART_MODE				USART 的模式
0	0	0	0	正常
0	0	0	1	RS485
0	0	1	0	硬件握手
0	0	1	1	保留
0	1	0	0	ISO7816 协议：T=0
0	1	0	1	保留
0	1	1	0	ISO7816 协议：T=1
0	1	1	1	保留
1	0	0	0	IrDA
1	1	x	x	保留

- USCLKS：时钟选择

USCLKS		选择的时钟
0	0	MCK
0	1	MCK/DIV (DIV = 8)
1	0	保留
1	1	SCK

- CHRL：字符长度

CHRL		字符长度
0	0	5bits
0	1	6bits
1	0	7bits
1	1	8bits

- SYNC：同步模式选择

0：USART 运行在异步模式下

1：USART 运行在同步模式下

- PAR：校验类型

PAR			校验类型
0	0	0	偶校验
0	0	1	奇校验
0	1	0	校验位强迫为 0（空号）
0	1	1	校验位强迫为 1（传号）
1	0	x	无校验
1	1	x	多点模式

- NBSTOP：停止位数

NBSTOP		异步（SYNC=0）	同步（SYNC=1）
0	0	1 个停止位	1 个停止位
0	1	1.5 个停止位	保留
1	0	2 个停止位	2 个停止位
1	1	保留	保留

- CHMODE：通道模式

CHMODE		模式描述
0	0	正常模式
0	1	自动回波。接收器输入连接到 TXD 引脚
1	0	本地回环。发送器输出连接到接收器输入
1	1	远程回环。RXD 引脚内部的连接到 TXD 引脚

- MSBF：位顺序

0：首先发送 LSB 位

1：首先发送 MSB 位

- MODE9：9 位字符长度

0：CHRL 定义字符长度

1：9 位字符长度

- CLKO：时钟输出选择

0：USART 不驱动 SCK 引脚

1：若 USCLKS 不选择外部时钟 SCK 则 USART 驱动 SCK

- OVER：过采样模式

0：16x 倍过采样

1：8x 被过采样

- INACK：禁止无应答

0：产生 NACK

1：不产生 NACK

- DSNACK：禁用连续的 NACK

0：只要在接收到的字符（除非 INACK 置位）中产生一个校验错误则发送 NACK 到 ISO 口线上。

1：连续的校验错误加起来达到在 MAX_ITERATION 域中规定的值。这些校验错误在 ISO 口线上产生一个 NACK。只要达到此值，不向 ISO 口线上发送附加的 NACK。ITERATION 标志位有效。

- MAX_ITERATION

定义 ISO7816 模式下，protocol T= 0 时最大重复数

- FILTER：红外接收口线滤波器

0：USART 不对接收口线滤波

1：USART 使用三采样滤波器(1/16-位时钟) (2/3 多数逻辑)对接收口线滤波。

31.7.3 USART 接口使能寄存器

名称：US_IER

访问类型：只写

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	CTSIC	–	–	–
15	14	13	12	11	10	9	8
–	–	NACK	RXBUFF	TXBUFE	ITERATION	TXEMPTY	TIMEOUT
7	6	5	4	3	2	1	0
PARE	FRAME	OVRE	ENDTX	ENDRX	RXBRK	TXRDY	RXRDY

- RXRDY：RXRDY 中断使能
- TXRDY：TXRDY 中断使能
- RXBRK：接收器间断中断使能
- ENDRX：接收传输结束中断使能
- ENDTX：发送结束中断使能
- OVER：超限错误中断使能
- FRAME：成帧错误中断使能
- PARE：校验错误中断使能
- TIMEOUT：超时中断使能
- TXMEPTY：TXMEPTY 中断使能
- ITERATION：迭代中断使能
- TXBUFE：缓冲区空中断使能
- RXBUFF：缓冲区满中断使能
- NACK：无应答中断使能
- CTSIC：发送输入变化清除中断使能

31.7.4 USART 中断禁用寄存器

名称：US_IDR

访问类型：只写

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	CTSIC	–	–	–
15	14	13	12	11	10	9	8
–	–	NACK	RXBUFF	TXBUFE	ITERATION	TXEMPTY	TIMEOUT
7	6	5	4	3	2	1	0
PARE	FRAME	OVRE	ENDTX	ENDRX	RXBRK	TXRDY	RXRDY

- RXRDY：RXRDY 中断禁用
- TXRDY：TXRDY 中断禁用
- RXBRK：接收器间断中断禁用
- ENDRX：接收传输结束中断禁用
- ENDTX：发送结束中断禁用

- OVER：超限错误中断禁用
- FRAME：成帧错误中断禁用
- PARE：校验错误中断禁用
- TIMEOUT：超时中断禁用
- TXMEPTY：TXMEPTY 中断禁用
- ITERATION：迭代中断禁用
- TXBUFE：缓冲区空中断禁用
- RXBUFF：缓冲区满中断禁用
- NACK：无应答中断禁用
- CTSIC：发送输入变化清除中断禁用

31.7.5 USART 中断屏蔽寄存器

名称：US_IMR

访问类型：只读

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	CTSIC	–	–	–
15	14	13	12	11	10	9	8
–	–	NACK	RXBUFF	TXBUFE	ITERATION	TXEMPTY	TIMEOUT
7	6	5	4	3	2	1	0
PARE	FRAME	OVRE	ENDTX	ENDRX	RXBRK	TXRDY	RXRDY

- RXRDY：RXRDY 中断屏蔽
- TXRDY：TXRDY 中断屏蔽
- RXBRK：接收器间断中断屏蔽
- ENDRX：接收传输结束中断屏蔽
- ENDTX：发送结束中断屏蔽
- OVER：超限错误中断屏蔽
- FRAME：成帧错误中断屏蔽
- PARE：校验错误中断屏蔽
- TIMEOUT：超时中断屏蔽
- TXMEPTY：TXMEPTY 中断屏蔽
- ITERATION：迭代中断屏蔽
- TXBUFE：缓冲区空中断屏蔽
- RXBUFF：缓冲区满中断屏蔽
- NACK：无应答中断屏蔽
- CTSIC：发送输入变化清除中断屏蔽

31.7.6 USART 通道状态寄存器

名称：US_CSR

访问类型：只读

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
CTS	–	–	–	CTSIC	–	–	–
15	14	13	12	11	10	9	8
–	–	NACK	RXBUFF	TXBUFE	ITERATION	TXEMPTY	TIMEOUT
7	6	5	4	3	2	1	0
PARE	FRAME	OVRE	ENDTX	ENDRX	RXBRK	TXRDY	RXRDY

- RXRDY：接收器就绪

0：自上次读取 US_RHR 未接收到完整的字符或已禁用接收器。若当接收器被禁用时正在接收字符，则当接收器使能时 RXRDY 变为 1.

1：至少接收到一个完整的字符并且还未读取 US_RHR

- TXRDY：发送器就绪

0：一个字符在 US_THR 正等待被发送到发送移位寄存器，或请求一个 STTBRK 命令，或发送器禁用。只要发送器使能，则 TXRDY 变为 1.

1：US_THR 中无字符

- RXBRK：间断接收/间断结束

0：自上次 RSTSTA 未检测到间断接收或间断结束

1：自上次 RSTSTA 检测到间断接收或间断结束

- ENDRX：接收器传输结束

0：来自接收 PDC 通道的传输结束信号无效

1：来自接收 PDC 通道的传输结束信号有效

- ENDRX：发送器传输结束

0：来自发送 PDC 通道的传输结束信号无效

1：来自发送 PDC 通道的传输结束信号有效

- OVER：超限错误

0：自上次 RSTSTA 未发生超限错误

1：自上次 RSTSTA 至少发生一次超限错误

- FRAME：成帧错误

0：自上次 RSTSTA 未检测到停止位

1：自上次 RSTSTA 至少检测到一个停止位

- PARE：校验错误

0：自上次 RSTSTA 未检测到校验错误

1：自上次 RSTSTA 至少检测到一次校验错误

- TIMEOUT：接收器超时

0：自上次启动超时命令(US_CR 中 STTTO)无超时或超时寄存器为 0

1：自上次启动超时命令(US_CR 中 STTTO)有一次超时

- TXEMPTY：发送器空

0：US_THR 或发送移位寄存器中有字符，或发送器禁用

1：US_THR 或发送移位寄存器中无字符

- ITERATION：达到最大复制数

0：自上次 RSIT 未达到最大复制数

1：自上次 RSIT 已达到最大复制数

- TXBUFE：发送缓冲区空

0：来自发送 PDC 通道的信号缓冲区空无效
1：来自发送 PDC 通道的信号缓冲区空有效

- RXBUFF：复制缓冲区满

0：来自接收 PDC 通道的信号缓冲区满无效
1：来自接收 PDC 通道的信号缓冲区满有效

- NACK：无应答

0：自上次 RSTNACK 未检测到无应答
1：自上次 RSTNACK 至少检测到一次无应答

- CTSIC：发送输入变化清除标志位

0：自上次读取 US_CSR 在 CTS 引脚上未检测到输入变化
1：自上次读取 US_CSR 在 CTS 引脚上至少检测到一次输入变化

- CTS：CTS 输入映像

0：CTS 为 0
1：CTS 为 1

31.7.7 USART 接收保持寄存器

名称：US_RHR
访问类型：只读

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

- RXCHR：接收到的字符

若 RXRDY 置位则为上次接收的字符

- RXSYNH：接收同步

0：上次接收到的字符为数据
1：上次接收到的字符为命令

31.7.8 USART 发送保持寄存器

名称：US_THR
访问类型：只写

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

- TXCHR：待发送字符

若未置位 TXRDY 在当前字符后下一个字符待发送

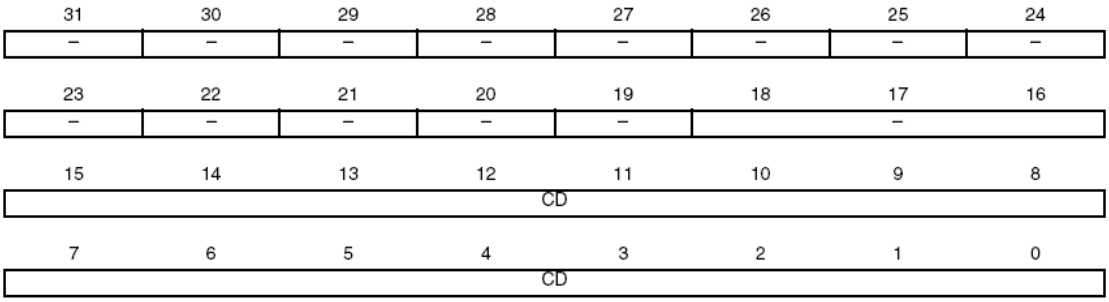
- TXSYNH：带发送 Sync 域

0：下一个发送字符被解码为一个数据。启动帧定界器为 DATA SYNC
1：下一个发送字符被解码为一个命令。启动帧定界器为 COMMAND SYNC

31.7.9 USART 波特率发生器寄存器

名称：US_BRGR

访问类型：读/写



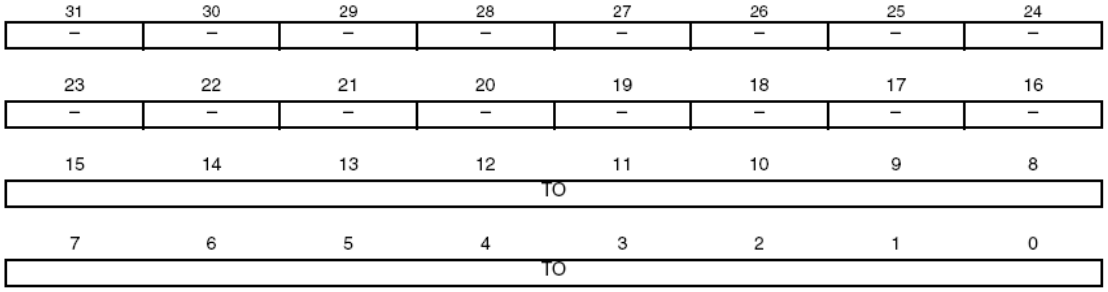
- CD：时钟分频器

CD	USART_MODE≠IS07816			USART_MODE =IS07816
	SYNC=0		SYNC=1	
	OVER=0	OVER=1		
0	波特率时钟禁用			
1 到 65535	波特率=选择的时钟/16/CD	波特率=选择的时钟/8/CD	波特率=选择的时钟/CD	波特率=选择的时钟/CD/FI_DI_RATIO

31.7.10 USART 接收起超时寄存器

名称：US_RTOR

访问类型：读/写



- TO：超时值

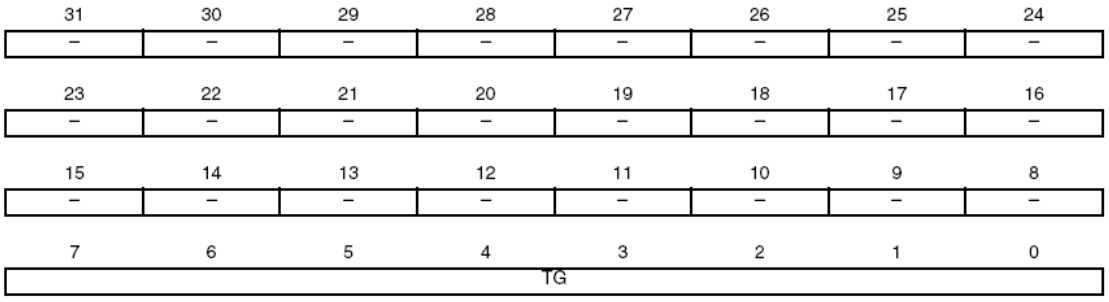
0：接收器超时禁用

1-65535：接收器超时使能并且超时延迟位 TO*Bit 周期

31.7.11 USART 发送器时间保护寄存器

名称：US_TTGR

访问类型：读/写



- TG：时间保护值

0：发送器时间保护禁用

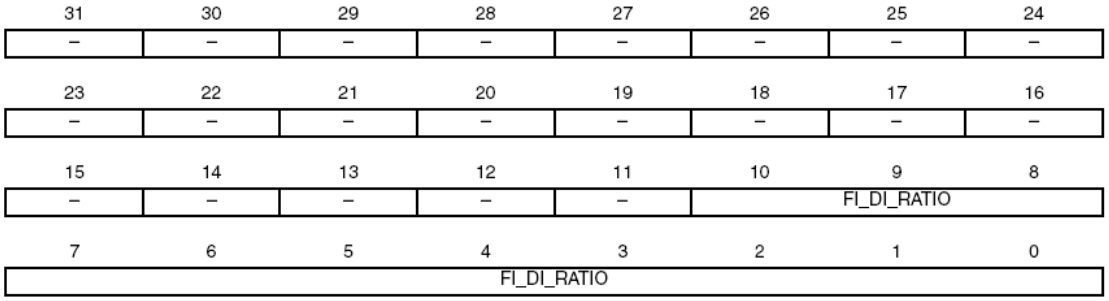
1-255：发送器时间保护使能并且时间保护延迟为 TG*Bit 周期

31.7.12 USART FI DI RADIO 寄存器

名称：US_FIDI

访问类型：读/写

复位值：0x174



- FI_DI_RADIO：FI/DI 比值

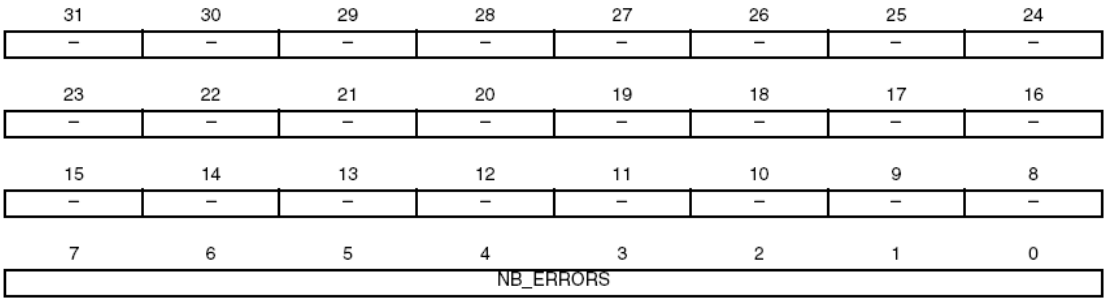
0：若选择 ISO7816 模式，则波特率发生器不产生信号

1-2047：若选择 ISO7816 模式，波特率为 SCK 上提供被 FI_DI_RATIO 分频的时钟。

31.7.13 USART 错误数寄存器

名称：US_NER

访问类型：只读



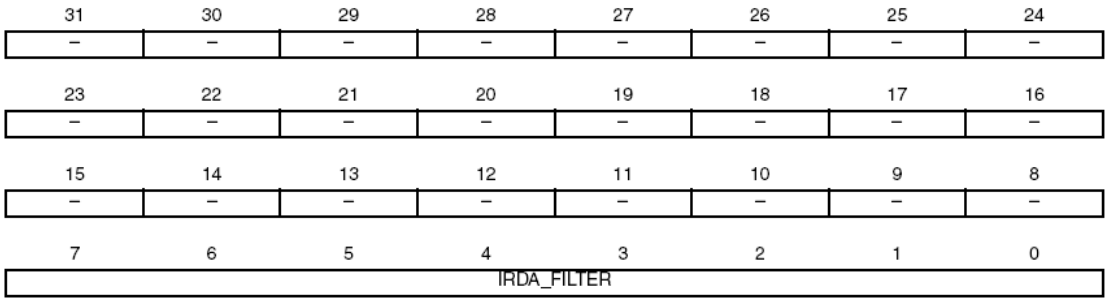
- NB_ERRORS：错误数

在一个 ISO7816 传输期间产生的错误总数。当读取时此寄存器自动清零

31.7.14 USART IrDA FILTER 寄存器

名称：US_IF

访问类型：读/写



- IRDA_FILTER：IrDA 滤波器

置位 IrDA 解调器的滤波器

32. 同步串行控制器（SSC）

32.1 描述

Atmel 同步串行控制器(SSC)提供与外部设备的同步通信。支持多种用于语音和电信应用中常用的串行同步通信协议，如 I2S，短帧同步，长帧同步等。

SSC 包含一个独立的接收器和发送器和一个共用时钟分频器。接收器和发送器每个接口由三个信号：针对数据的 TD/RD 信号，针对时钟的 TK/RK 信号和针对帧同步的 TF/RF 信号。可编程设定为自动启动或在帧同步信号上检测到不同事件时启动。

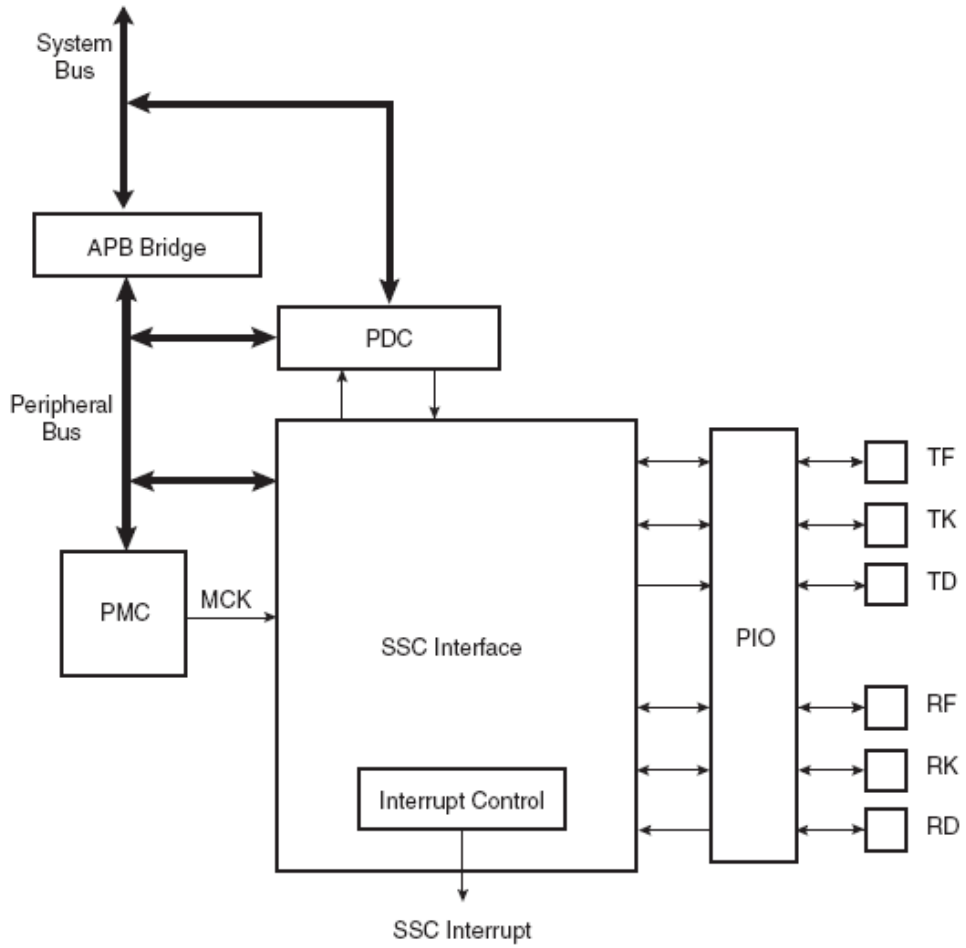
SSC 的可编程高电平和两个 32 位专用 PDC 通道，可在无处理器干涉时进行连续的高比特率的数据传输。

基于与两个 PDC 通道连接的特性，SSC 允许在低处理器开销下连接以下设备：

- 主/从模式下 CODEC
- 专用串行接口的 DAC，特别是 I2S
- 磁卡读取器

32.2 框图

图 32-1 框图



32.3 应用框图

图 32-2 应用框图

OS 或 RTOS 驱动		电源管理	中断管理	测试管理
SSC				
串行 AUDIO	CODEC	时间槽管理	帧管理	线接口

32.4 引脚名称列表

表 32-1 I/O 口线描述

引脚名称	引脚描述	类型
RF	接收器帧同步	输入/输出
RK	接收器时钟	输入/输出
RD	接收器数据	输入
TF	发送器帧同步	输入/输出
TK	发送器时钟	输入/输出
TD	发送器数据	输出

32.5 附属产品

32.5.1 I/O 口线

用于连接兼容的外部设备的引脚可与 PIO 口线复用。

使用 SSC 接收器之前，必须将 PIO 控制器配置为专用的 SSC 接收器 I/O 口线的 SCC 外设模式。

使用 SSC 发送器之前，必须将 PIO 控制器配置为专用的 SSC 发送器 I/O 口线的 SCC 外设模式。

32.5.2 电源管理

SSC 时钟不是连续的。可以通过电源管理控制器(PMC)协调同步 SSC 接口，因此编程者必须首先配置 PMC 来使能 SSC 时钟。

32.5.3 中断

SSC 接口连接于高级中断控制器(AIC)的一个中断口线。处理中断需要在配置 SSC 前编程 AIC。

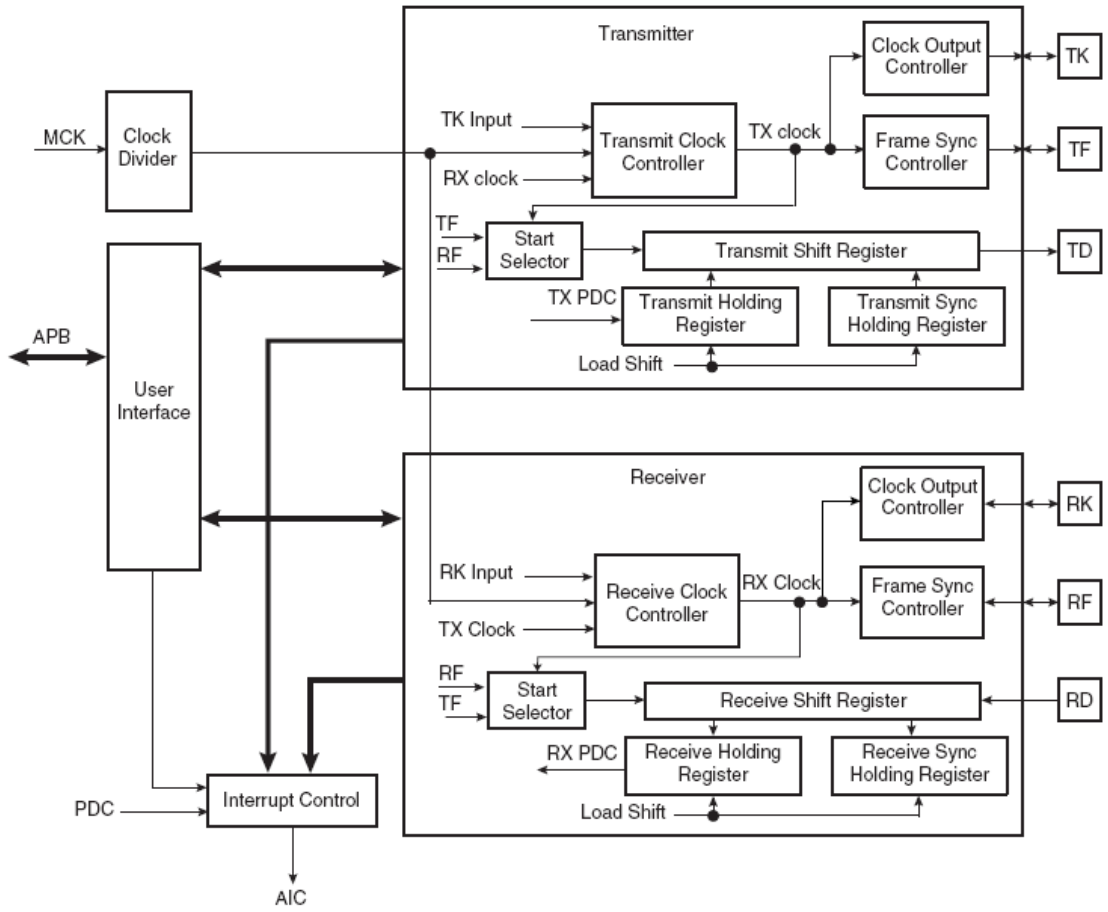
可通过配置 SSC 中断屏蔽寄存器使能和禁用所有的 SSC 中断。每个挂起的或未屏蔽的 SSC 中断源将出现在 SSC 中断线上。SSC 中断服务子程序可通过读取 SSC 中断状态寄存器来得到中断源。

32.6 功能描述

这一章包括以下说明：SSC 功能框图，时钟管理器，数据格式，启动，发送器，接收器和帧同步。

接收器和发送器独立运行。然而，可以通过编程接收器来使用发送时钟或当发送器启动时开始数据传输，使得接收器和发送器同步工作。另外，可通过编程发送器使用接收时钟或当接收启动时开始数据传输实现同步。可编程发送器和接收器使其运行于 TK 或 RK 提供的时钟信号。这就允许 SSC 支持很多的从模式数据传输。TK 和 RK 引脚上允许的最大时钟速率为主控时钟的 2 分频。

图 32-2 SSC 功能框图



32.6.1　时钟管理

发送器时钟可由以下产生：

- TK I/O 上接收的外部时钟
- 接收器时钟
- 内部时钟分频器

接收器时钟可以由以下产生：

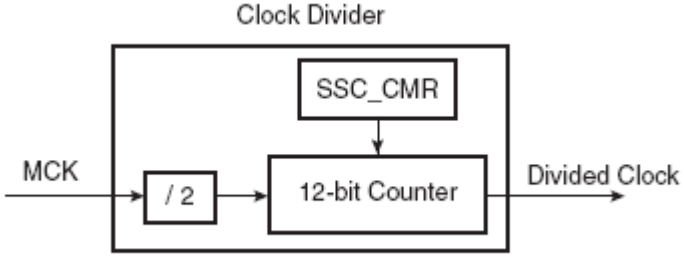
- RK I/O 上接收的外部时钟
- 发送器时钟
- 内部时钟分频器

此外，发送器时钟可以在产生 TK I/O 焊盘上产生外部时钟，接收器时钟可在 RK I/O 上产生外部时钟

这就使得 SSC 支持很多的主从模式数据传输。

32.6.1.1　时钟分频器

图 32-4　分频时钟框图



时钟模式寄存器 SSC_CMR 中的 12 位域 DIV 计数器和比较器（所以起最大值为 4095）决定主控时钟分频器，允许主控时钟多达 8190 分频。分频时钟提供给接收器和发送器。当将此域编程为 0，则不使用时钟分频器并保持无效。

当 DIV 设置为大于等于 1 的值，分频时钟为 2 倍 DIV 分频的主控时钟。分频时钟的每级电平持续时间为 DIV 倍的主控时钟。这就确保了 50%占空比的周期分频时钟，不管 DIV 值是奇数或偶数。

图 32-5 分频时钟的产生

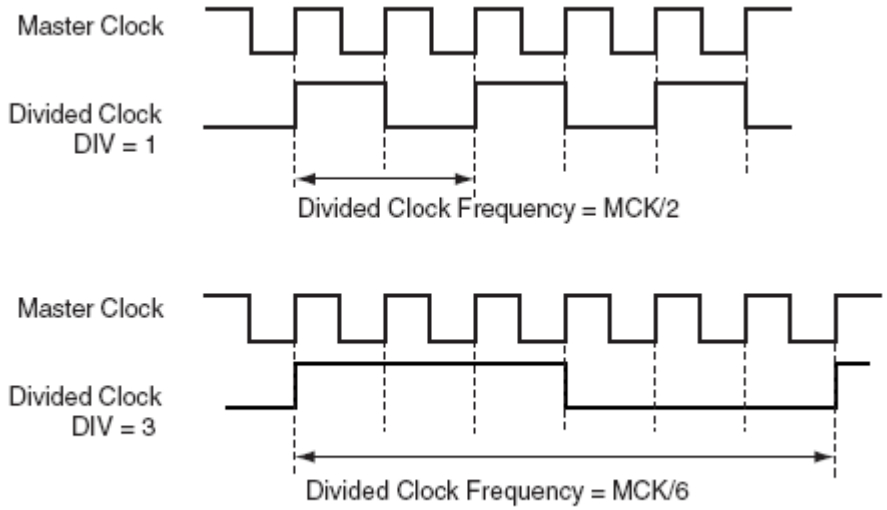


表 32-2

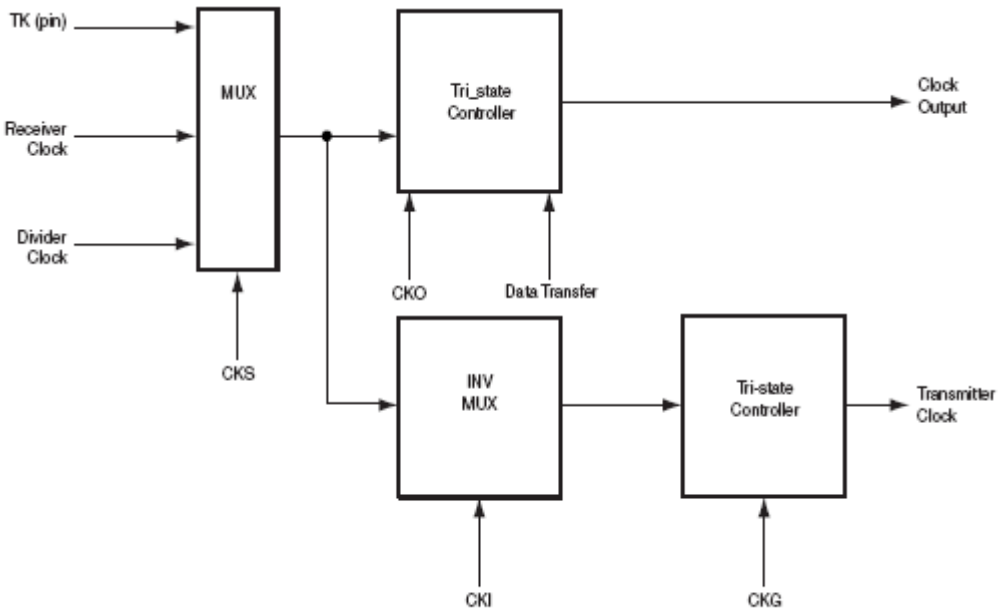
最大值	最小值
MCK / 2	MCK / 8190

32.6.1.2 发送器时钟管理

发送器时钟来自接收器时钟或分频器时钟或 TK I/O 上扫描的外部时钟。通过 SSC_TCMR（发送时钟模式寄存器）中的 CKS 域选择发送器时钟。可通过 SSC_TCMR 中的 CKI 位独立的反转发送时钟。

发送器还可连续的驱动 TK I/O 或受限于实际数据传输。通过 SSC_TCMR 寄存器配置时钟输出。发送时钟反转位（CKI）对时钟输出无影响。编程 TCMR 寄存器选择 TK 引脚（CKS 域）并且同时连续发送时钟（CKO 域）可能导致不可预期的结果。

图 32-6 发送器时钟管理

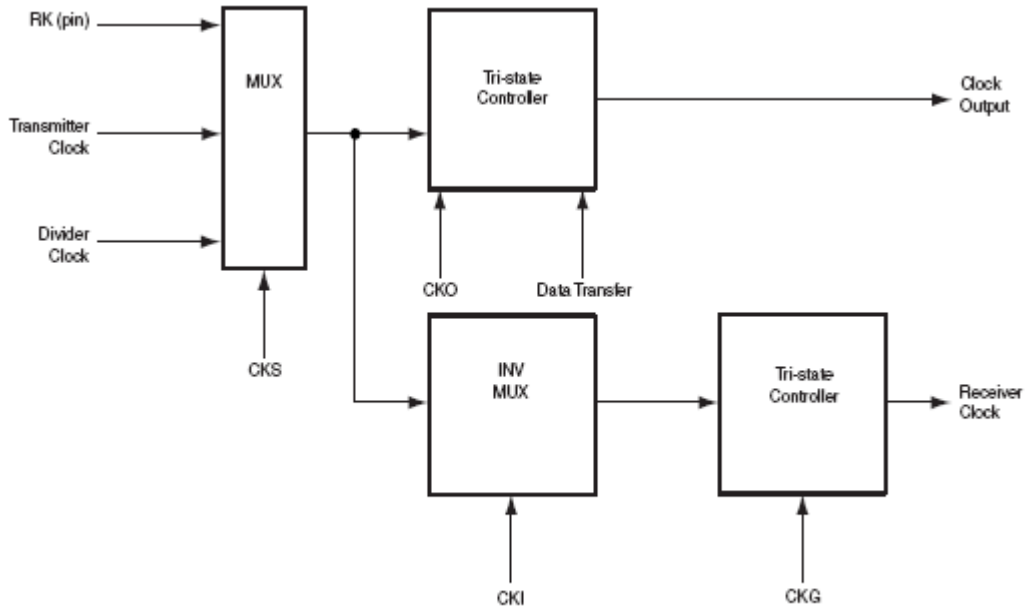


32.6.1.3 接收器时钟管理

接收器时钟来自发送器时钟或分频器时钟或 RK I/O 上的外部时钟。通过 SSC_RCMR(接收时钟模式寄存器)中的 CKS 域选择接收时钟。SSC_RCMR 中的 CKI 位可独立的反转接收时钟。

接收器还可连续的驱动 RK I/O 或受限于实际的数据传输。通过 SSC_RCMR 寄存器配置时钟输出。接收时钟反转位（CKI）不影响时钟输出。编程 RCMR 寄存器来选择 RK 引脚（CKS 域），同时连续接收时钟（CKO 域）可导致不可预期的结果。

图 32-7 接收器时钟管理



32.6.1.4 串行时钟率事宜

可编程发送器和接收器使其工作于 TK 或 RK 引脚上提供的时钟信号。这就允许 SSC 支持很多从模式的数据传输。此时，RK 引脚上允许的最高时钟速

率为：

-若接收器帧同步为输入则为 2 分频的主控时钟

-若接收器正同步为输出则为 3 分频的主控时钟

另外，TK 引脚上允许的最大时钟速率为：

-若发送帧同步为输入则为 6 分频的主控时钟

-若发送帧同步为输出则为 2 分频的主控时钟

32.6.2 发送器操作

通过一个启动事件触发一个发送帧，并可在数据发送前加入同步数据。

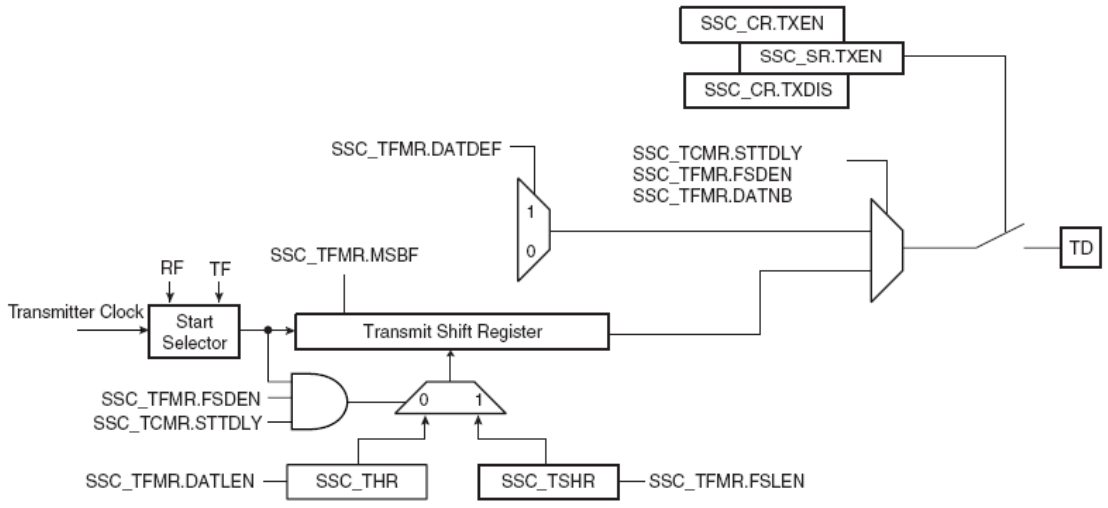
通过设置发送时钟模式寄存器(SSC_TCMR)配置启动事件。见 452 页“启动”

通过设置发送帧同步模式寄存器(SSC_TFMR)配置帧同步。见 454 页“帧同步”

发送数据时，发送器使用由发送器时钟信号协调同步的移位寄存器和在 SSC_TCMR 中选择的启动模式。应用程序将数据写入 SSC_THR 寄存器接着根据选择的数据格式将数据传输到移位寄存器。

当 SSC_THR 和发送移位寄存器都空的时候，置位 SSC_SR 中的状态标志位 TXEMPTY。当把发送保持寄存器数据传输到发送移位寄存器时，置位 SSC_SR 中的状态标志位 TXRDY 并且可将附加数据装载到保持寄存器。

图 32-8 发送器框图



32.6.3 接收器操作

可通过一个启动事件触发一个接收帧并可在数据发送前加入同步数据。

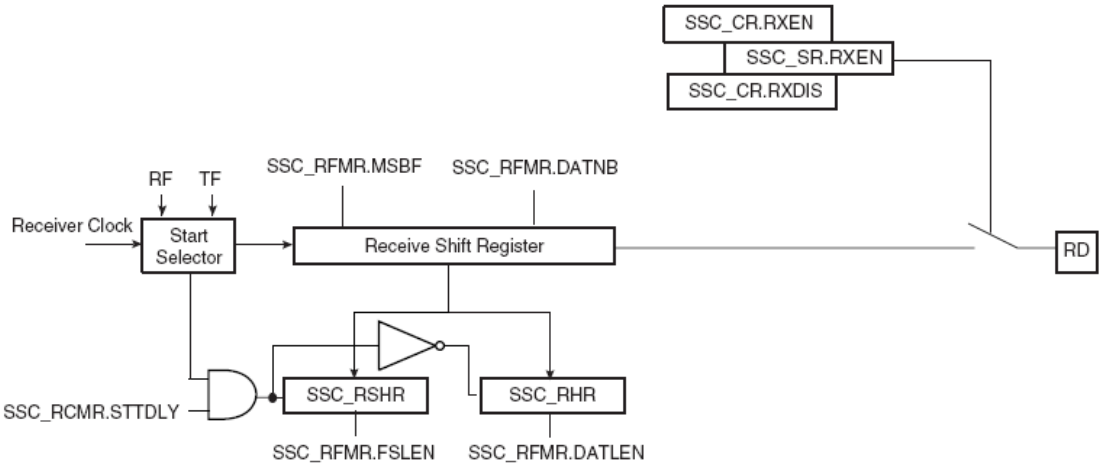
通过设置接收时钟模式寄存器(SSC_RCMR)配置启动事件。见 452 页“启动”

通过设置接收帧模式寄存器(SSC_RFMR)配置帧同步。见 454 页“帧同步”

接收器使用由接收器时钟信号协调同步的移位寄存器和 SSC_RCMR 中选择的启动模式。从移位寄存器传输数据取决于选择的数据格式。

当接收器移位寄存器满时，SSC 将此数据传输到保持寄存器，置位 SSC_SR 中的状态标志位 RXRDY 并且可在接收器保持寄存器中读取此数据。若在读取 RHR 寄存器前发生其他的传输，则置位 SSC_SR 中的状态标志位 OVERUN 并且将接收器移位寄存器数据传输到 RHR 寄存器。

图 32-9 接收器框图



32.6.4 启动

当一事件发生，可分别在 SSC_TCMR 的发送启动选择(START)域和 SSC_RCMR 的接收启动选择域中编程发送器和接收器来启动其操作。

以下情况中，启动事件可独立编程：

- 连续。此时，只要向 SSC_THR 写入一个字就启动发送，只要接收器使能就启动接收。
- 与发送器/接收器同步
- TF/RF 上检测到一个下降/上升沿
- TF/RF 上检测到一个低电平/高电平
- TF/RF 上检测到一个电平变化或一个边沿

可在发送/接收时钟寄存器(RCMR/TCMR)的任意边以同样方式编程启动。因此，可以在 TF（发送）上或 RF（接收）上启动

此外，当在比特流中比较检测到数据则接收器可以启动

发送/接收帧模式寄存器(TFMR/RFMR)的 FSOS 域可完成 TF/RF 上输入/输出的检测。

图 32-10 发送启动模式

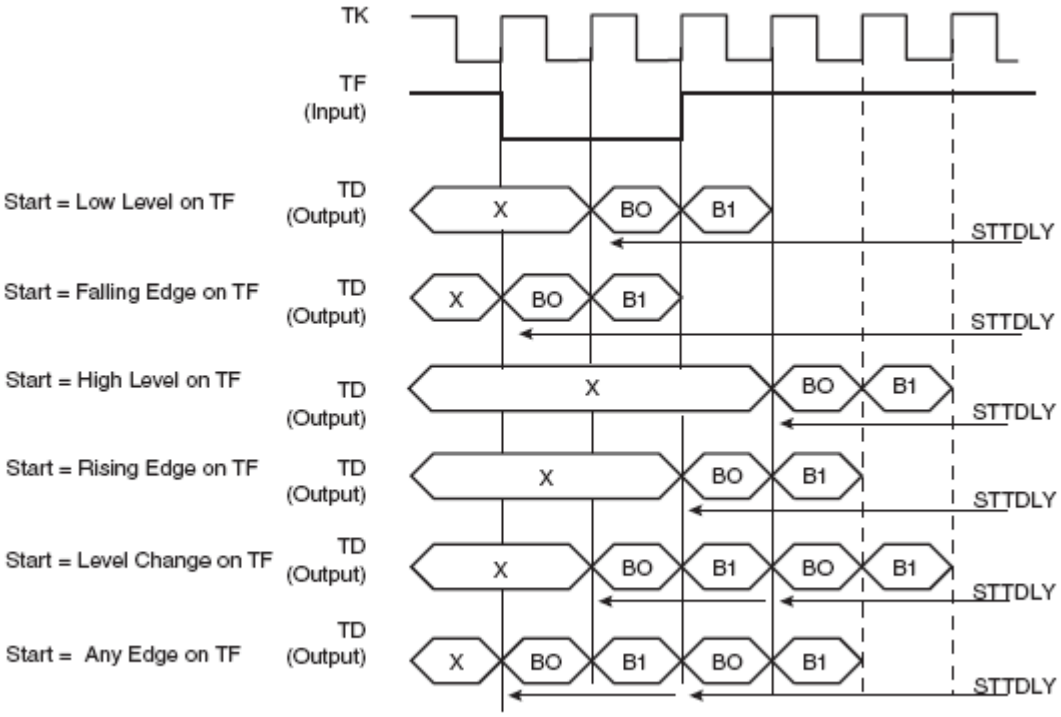
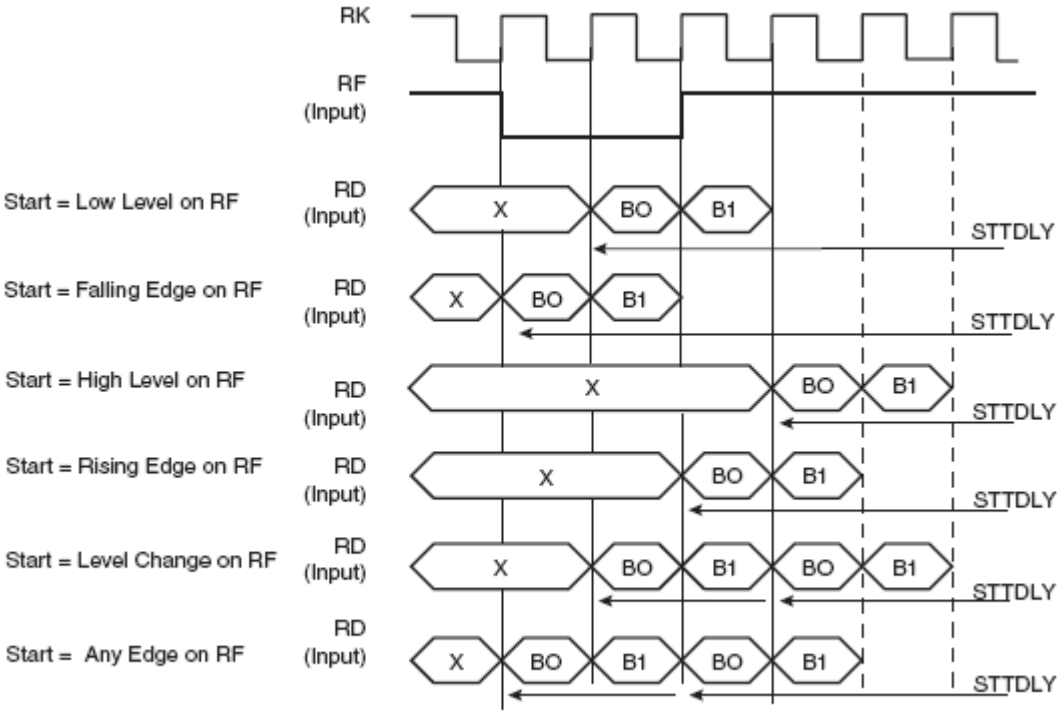


图 32-11 接收脉冲/边沿启动模式



32.6.5 帧同步

发送器和接收器帧同步引脚，TF 和 RF，可配置其用于产生各种不同的帧同步信号。接收帧模式寄存器(SSC_RFMR)和发送帧模式寄存器(SSC_TFMR)中的帧同步输出选择(FSOS)域用来选择所需波形。

- 支持数据传输期间可编程的低电平/高电平
- 支持数据传输前可编程的高电平或电平切换

若选择一个脉冲波形，SSC_RFMR 和 SSC_TFMR 中的帧同步长度(FSLEN)域编程脉冲的长度，从 1 比特时间到 16 比特时间

可通过 SSC_RCMR 和 SSC_TCMR 中的周期分频选择(PERIOD)域编程接收和发送帧同步脉冲输出的周期。

32.6.5.1 帧同步数据

帧同步数据在帧同步信号内发送或接收一个特定的标记。

帧同步信号内，接收器可采样 RD 口线并将数据存入接收同步保持寄存器并且发送器可将发送同步保持寄存器数据传输到移位寄存器。通过 SSC_RFMR/SSC_TFMR 中的 FSLEN 域来编程帧同步信号内采样/移出的数据长度并且起最大值为 16。

关于接收帧同步数据操作，若帧同步长度等于或小于启动时间和实际数据接收间的延迟，则通过接收移位寄存器在接收同步保持寄存器中执行数据采样操作。

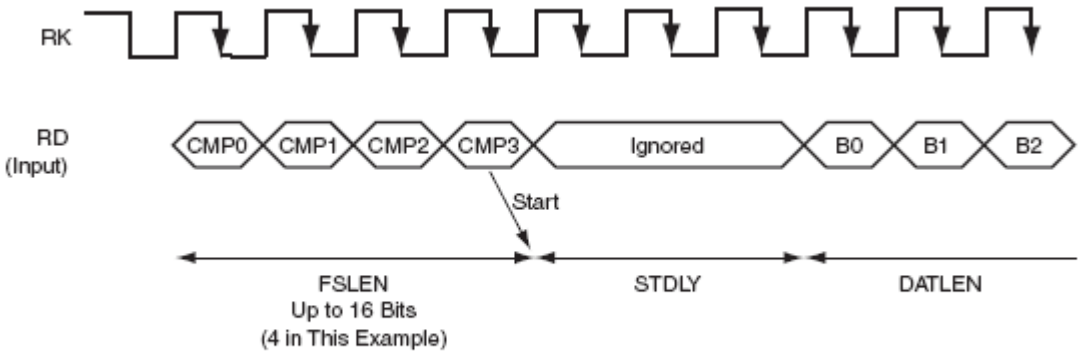
只要 SSC_TFMR 中的帧同步数据使能位置位则发送器就发送帧同步操作。若帧同步长度等于或低于启动事件和实际数据发送间的延迟，则正常发送优先并且将发送同步保持寄存器中的数据传输到发送寄存器，然后移出。

32.6.5.2 帧同步边沿检测

由 SSC_RFMR/SSC_TFMR 中的 FSEDGE 域执行帧同步边沿检测。将 SSC 状态寄存器(SSC_SR)中相应的标志位 RXSYN/TXSYN 设置为帧同步边沿检测（RF/TF 信号）

32.6.6 接收比较模式

图 32-12 接收比较模式



32.6.6.1 比较功能

FSLEN 定义比较模式(Compare0,Compare1)的长度和相比较的位数，但其最大值为 16 位。通常比较上次接收到的位和比较模式，Compare 0 可以是接收器的一个启动事件。此时接收器以上次接收的位的每个新采样值和 Compare 0 寄存器中(SSC_RC0R)Compare 0 模式做比较。当检测到此启动事件，则用户可通过写一个新的 Compare 0，或通过连续的接收编程接收器来启动一个新的数据传输直到发生 Compare 1。SSC_RCMR 中的(STOP)位完成此选择。

32.6.7 数据格式

发送器和接收器的数据成帧格式都可通过发送器帧模式寄存器(SSC_TFMR)

和发送器帧模式寄存器(SSC_RFMR)编程。

无论何种情况，用户可独立的选择：

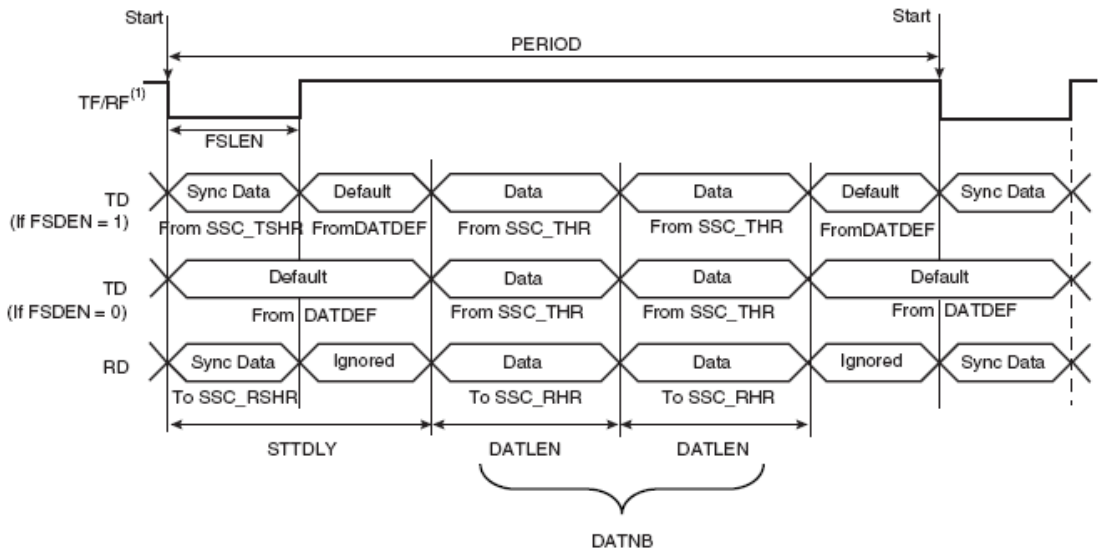
- 启动数据传输的事件(START)
- 启动事件和首个数据位之间的比特周期数延迟(STTDLY)
- 数据长度(DATLEN)
- 每次启动事件传输的数据数(DATNB)
- 每次启动事件同步传输长度(FSLEN)
- 比特意义：高位或低位在先(MSBF)

另外，当未数据传输时，可以使用发送器来传输同步并且选择 TD 引脚启动电平。可分别通过 SSC_TFMR 中的帧同步数据使能(FSDEN)和数据缺省值(DATDEF)位完成。

表 32-3 数据帧寄存器

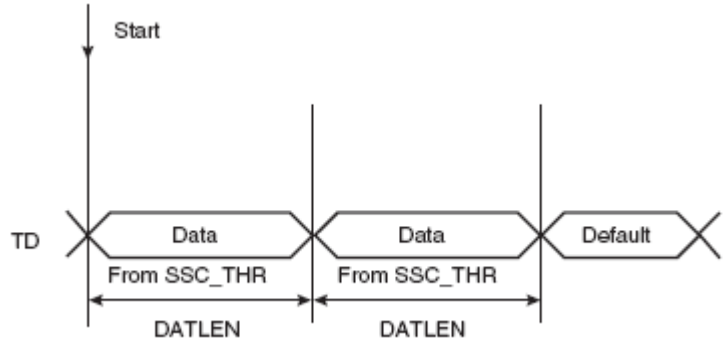
发送器	接收器	域	长度	备注
SSC_TFMR	SSC_RFMR	DATLEN	达 32	字长
SSC_TFMR	SSC_RFMR	DATNB	达 16	发送帧字数
SSC_TFMR	SSC_RFMR	MSBF		最高位在先
SSC_TFMR	SSC_RFMR	FSLEN	达 16	同步数据寄存器大小
SSC_TFMR		DATDEF	0 或 1	数据缺省值结束
SSC_TFMR		FSDEN		使能发送 SSC_TSHR
SSC_TCMR	SSC_RCMR	PERIOD	达 512	帧长度
SSC_TCMR	SSC_RCMR	STTDLY	达 255	发送启动延迟大小

图 32-13 边沿/脉冲启动模式下发送和接收帧格式



注意：1.TF/RF 下降沿输入举例

图 32-14 连续模式下发送帧格式

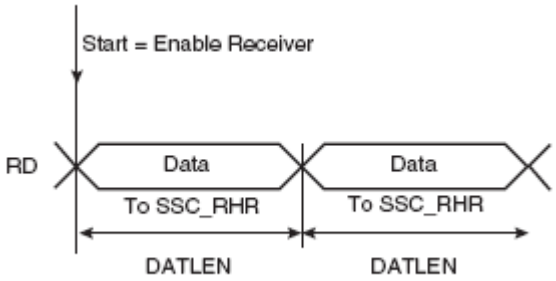


启动：1.TXEMPTY 置为 1

2.写入 SSC_THR

注意：1.STTDLY 置为 0.此例中 SSC_THR 被装载了两次。FSDEN 值不影响发送。连续模式下同步数据不能为输出。

图 32-15　　连续模式下接收帧格式



注意：1.STTDLY 置为 0

32.6.8　循环模式

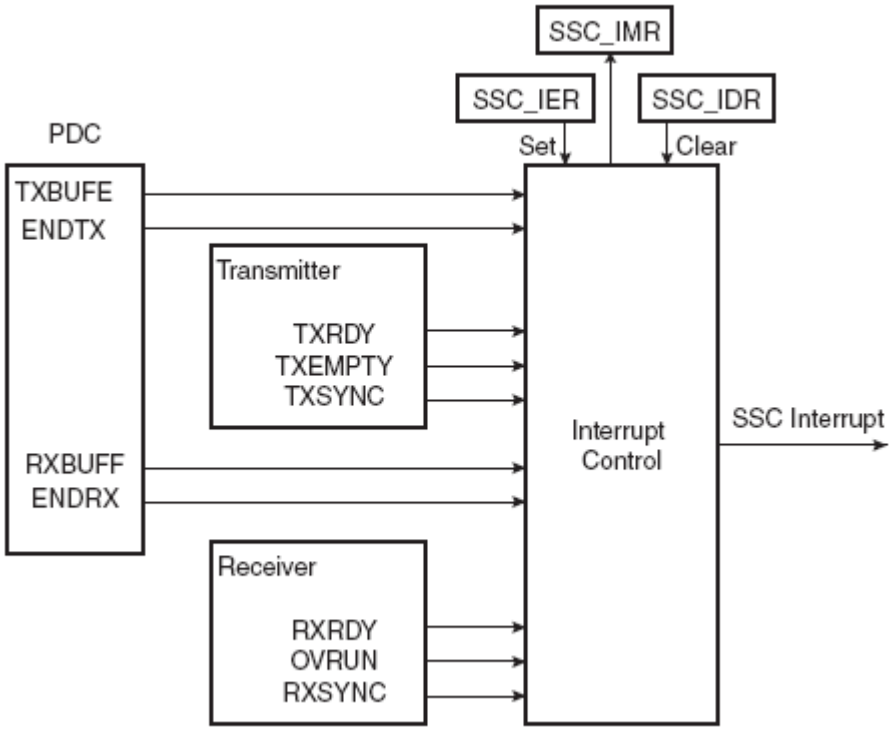
可编程接收器来从发送器接收发送。通过设置 SSC_RFMR 中的循环模式(LOOP)位完成。此时 RD 连接 TD，RF 连接 TF，RK 连接 TK。

32.6.9　中断

SSC_SR 中大多数位有中断管理寄存器中相应的位

SSC 可被编程来当检测到一个事件时产生中断。通过写 SSC_IER（中断使能寄存器）和 SSC_IDR（中断禁用寄存器）来控制中断。通过分别置位和清零 SSC_IMR（中断屏蔽寄存器）中相应位来使能和和禁用这些寄存器，通过激活连接于 AIC 的 SSC 中断口线来使得 SSC_IMR 控制中断的产生。

图 32-16 中断框图



32.7　　应用举例

SSC 可支持若干使用音频或高速串行链接的串行通信模式。一些标准应用如下图所示。所有通过 SSC 支持的串行链接应用未列出。

图 32-17　　音频应用框图

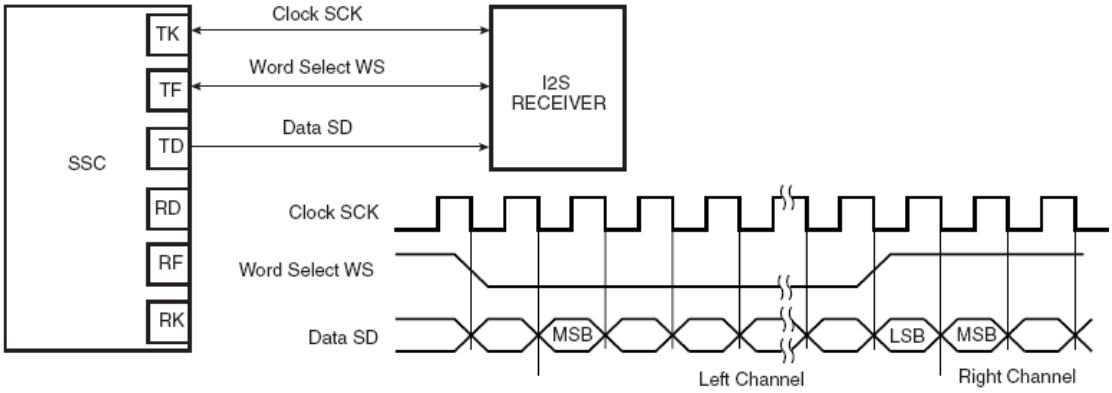


图 32-18　　Codec 应用框图

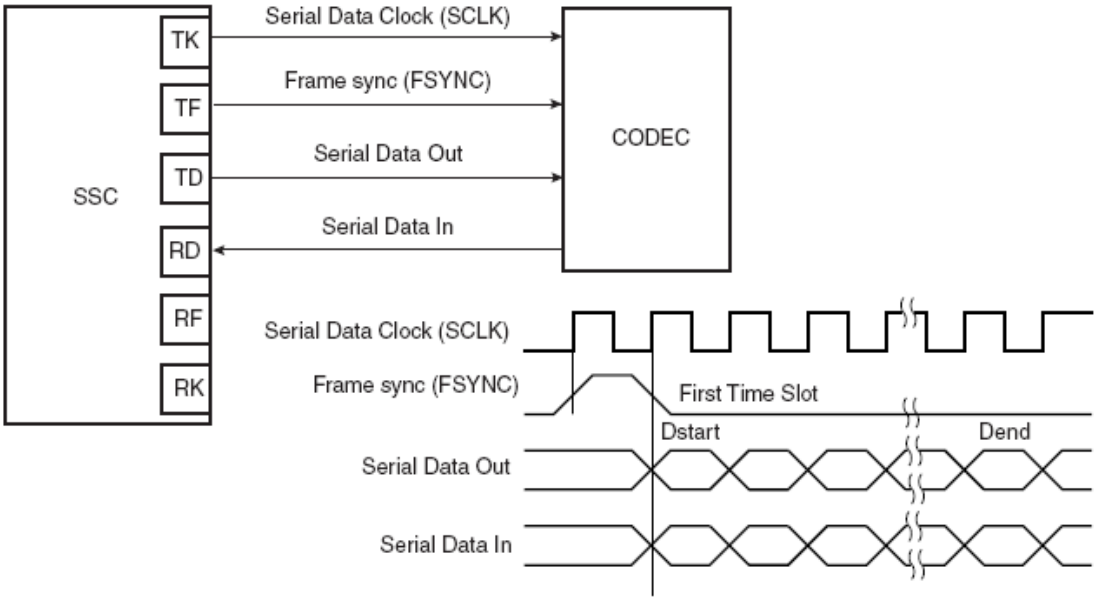
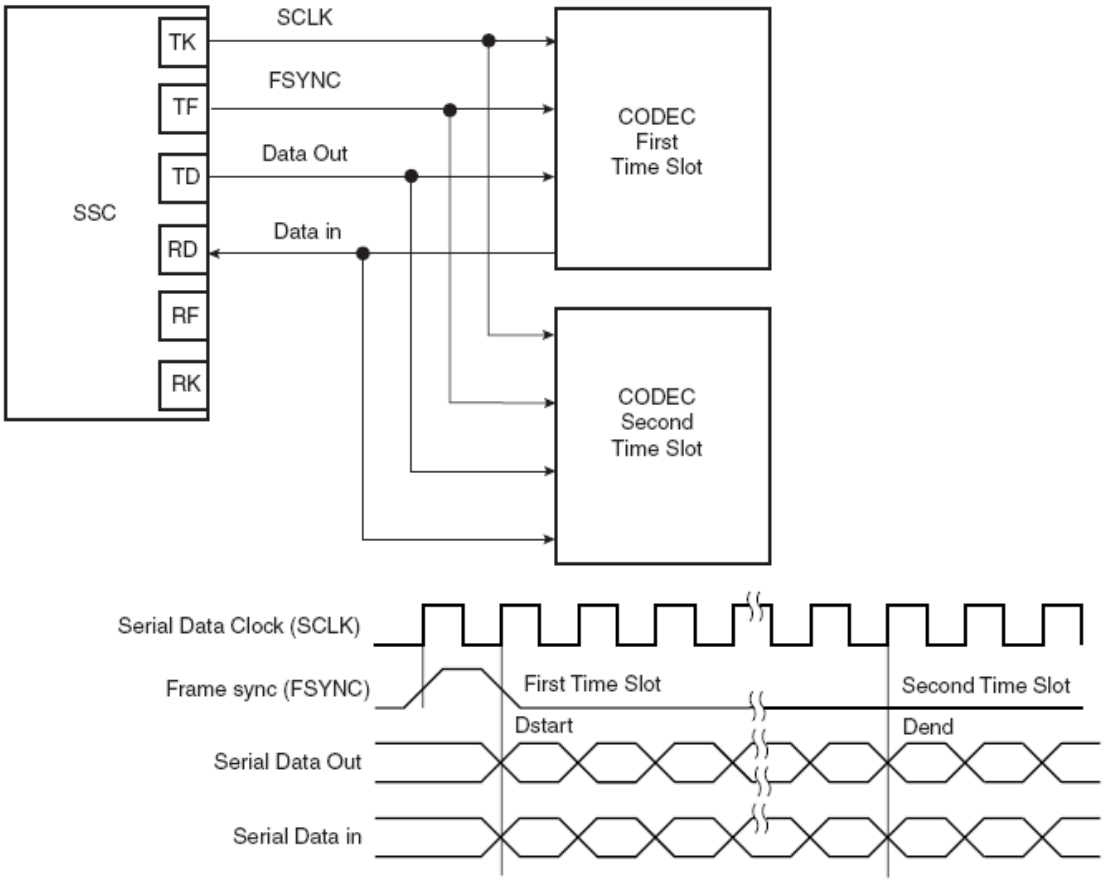


图 32-19　　时间槽应用框图



32.8 同步串行控制器（SSC）用户接口

表 32-4 寄存器映射

偏移量	寄存器	寄存器名称	访问类型	复位值
0x0	控制寄存器	SSC_CR	只写	-
0x4	时钟模式寄存器	SSC_CMR	读/写	0x0
0x8	保留	-	-	-
0xC	保留	-	-	-
0x10	接收时钟模式寄存器	SSC_RCMR	读/写	0x0
0x14	接收帧模式寄存器	SSC_RFMR	读/写	0x0
0x18	发送时钟模式寄存器	SSC_TCMR	读/写	0x0
0x1C	发送帧模式寄存器	SSC_TFMR	读/写	0x0
0x20	接收保持寄存器	SSC_RHR	只读	0x0
0x24	发送保持寄存器	SSC_THR	只写	-
0x28	保留	-	-	-
0x2C	保留	-	-	-
0x30	接收同步保持寄存器	SSC_RSHR	只读	0x0
0x34	发送同步保持寄存器	SSC_TSHR	读/写	0x0
0x38	接收 Compare0 寄存器	SSC_RC0R	读/写	0x0
0x3C	接收 Compare1 寄存器	SSC_RC1R	读/写	0x0
0x40	状态寄存器	SSC_SR	只读	0x000000CC
0x44	中断使能寄存器	SSC_IER	只写	-
0x48	中断禁用寄存器	SSC_IDR	只写	-
0x4C	中断屏蔽寄存器	SSC_IMR	只读	0x0
0x50-0xFC	保留	-	-	-
0x100-0x124	为外设数据控制器（PDC）保留	-	-	-

32.8.1 SSC 控制寄存器

名称：SSC_CR

访问类型：只写

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

- RXEN：接收使能

0：无效

1：若 RXDIS 未置位则使能接收

- RXDIS：接收禁用

0：无效

1：禁用接收。若当前正在传输一个字符，在当前接收字符的结尾禁用。

- TXEN：发送使能

0：无效

1：若 TXDIS 未置位则使能发送

- TXDIS：发送禁用

0：无效

1：禁用发送。若当前正在发送一个字符，在发送当前字符的结尾禁用

- SWRST：软件复位

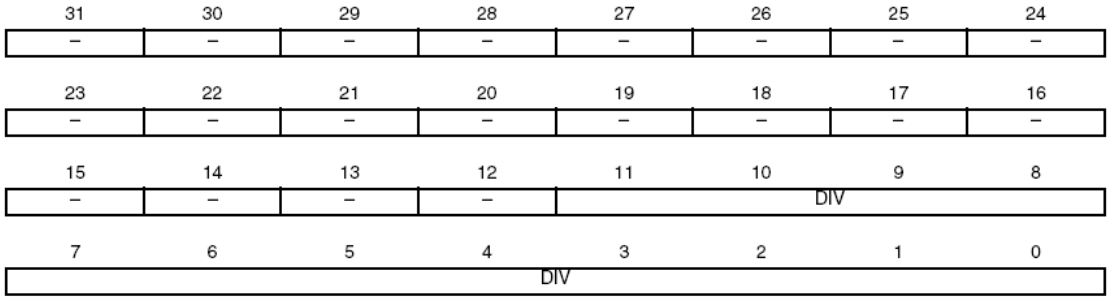
0：无效

1：执行一个软件复位。比 SSC_CR 中任何其它位优先级高

32.8.2 SSC 时钟模式寄存器

名称：SSC_CMR

访问类型：读/写



DIV：时钟分频器

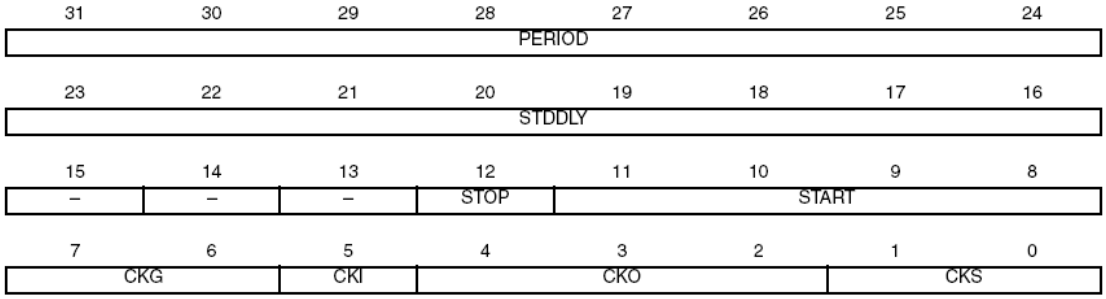
0：时钟分频起无效

任何其它值：分频时钟等于 2 倍 DIV 分频的主控时钟。最大比特率为 MCK/2。最小比特率为 MCK/2 x 4095 = MCK/8190

32.8.3 SSC 接收时钟模式寄存器

名称：SSC_RCMR

访问类型：读/写



- CKS：接收时钟选择

CKS	选择的接收时钟
0x0	分频的时钟
0x1	TK 时钟信号
0x2	RX 引脚
0x3	保留

- CKO：接收时钟输出模式选择

CKO	接收时钟输出模式	RK 引脚
0x0	无	只输入
0x1	连续接收时钟	输出
0x2	仅在数据传输期间的接收时钟	输出
0x3-0x7	保留	输出

- CKI：接收时钟反转

0：在接收时钟下降沿采样数据输入（数据和帧同步信号）。帧同步信号输出在接收时钟上升沿时被移出。

1：在接收时钟上升沿采样数据输出（数据和帧同步信号）。帧同步信号输出在接收时钟下降沿时被移出。

CKI 仅影响接收时钟而不影响输出时钟信号

- CKG：接收时钟选通选择

CKG	接收时钟选通
0x0	无，连续时钟
0x1	只有在 RF 低电平时接收时钟使能
0x2	只有在 RF 高电平时接收时钟使能
0x3	保留

- START：接收启动选择

START	接收启动
0x0	连续，只要接收器使能，在前一次数据传输结束后立即启动
0x1	发送启动
0x2	检测 RF 信号上的低电平
0x3	检测 RF 信号上的高电平
0x4	检测 RF 信号上的下降沿
0x5	检测 RF 信号上的上升沿
0x6	检测 RF 信号上的任何电平变化
0x7	检测 RF 信号上的任何边沿
0x8	Compare 0
0x9-0xF	保留

- STOP：接收停止选择

0：数据传输完成后当启动 Compare0 时，接收器停止数据传输并且等待一个新的 compare0

1:启动接收 Compare0 后，接收器运行在连续模式直到检测到 Compare1。

- STTDLY：接收启动延迟

若 STTDLY 不为 0，将在启动事件和实际接收启动之间插入一个 STTDLY 时钟周期的延迟。当编程接收器使和发送器同步启动时，延迟仍然适用。

注意：仔细设置 STTLY 非常重要。若必须设置 STTDLY，则应该关于 TAG（接收同步数据）接收。

- PERIOD：接收周期分频器选择

此域选择分频器来应用于选择的接收以产生新的帧同步信号。若为 0，则不产生 PERIOD 信号。若不为 0，则每 2 x (PERIOD+1)个接收周期产生一个 PERIOD 信号

32.8.4 SSC 接收帧模式寄存器

名称：SSC_RFMR

访问类型：读/写

- DATLEN：数据长度

0：禁用值（不支持 1 位数据长度）

任何其它值：比特流包括 DATLEN+1 数据位。此外，它定义了接收器 PDC2 执行的传输大小。若 DATLEN 低于或等于 7，则数据以字节传输。若 DATLEN 在 8 和 15 之间（包含 15），以半字传输，其它任何值时，以 32 位字传输。

- LOOP：循环模式

0：正常运行模式

1：TD 驱动 RD，TF 驱动 RF，TK 驱动 RK

- MSBF：最高位在先

0：首先采样比特流中数据寄存器的最低位

1：首先采样比特流中数据寄存器的最高位

- DATNB：每帧数据数

此域定义了每次传输启动后待发送数据字数，其值等于(DATNB + 1)

- FSLEN：接收帧同步长度

此域定义了采样和存储在接收同步数据寄存器中的位数。当通过接收时钟模式寄存器中的 START 域选择此模式时，他还决定了要和 Compare 0 或 Compare 1 寄存器比较的采样数据的长度。

此域和 FSLEN_EXT 一起使用来决定接收帧同步信号的脉冲长度。

脉冲长度等于 FSLEN + 1 个接收时钟周期

- FSOS：接收帧同步输出选择

FSOS	选择的接收帧同步信号	RF 引脚
0x0	无	仅输入
0x1	负脉冲	输出
0x2	正脉冲	输出
0x3	数据传输期间驱动为低	输出
0x4	数据传输期间驱动为高	输出
0x5	每次数据传输启动时切换	输出
0x6-0x7	保留	未定义

- FSDGE：帧同步边沿检测

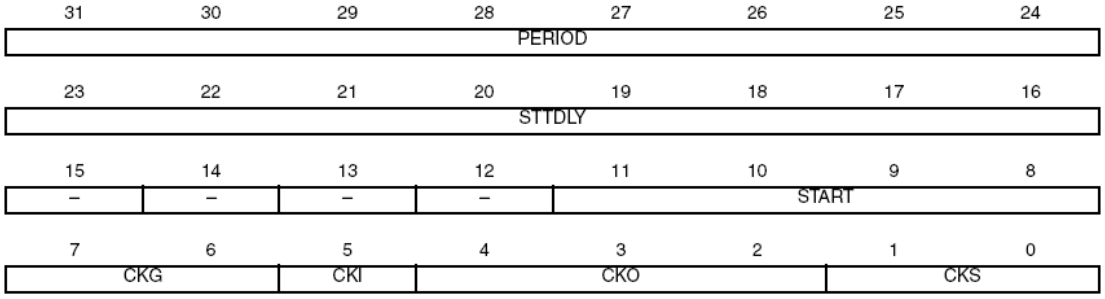
决定帧同步哪个边沿产生 SSC 状态寄存器中的中断 RXSYN

FSEDGE	帧同步边沿检测
0x0	正边沿检测
0x1	负边沿检测

32.8.5 SSC 发送时钟模式寄存器

名称：SSC_TCMR

访问类型：读/写



- CKS：发送时钟选择

CKS	选择的发送时钟
0x0	分频时钟
0x1	RK 时钟信号
0x2	TK 引脚
0x3	保留

- CKO：发送时钟输出模式选择

CKO	发送时钟输出模式	TK 引脚
0x0	无	仅输入
0x1	连续发送时钟	输出
0x2	仅在数据传输期间的发送时钟	输出
0x3-0x7	保留	

- CKI：发送时钟反转

0：在发送时钟下降沿移出数据输出（数据和帧同步信号），在发送时钟上升沿采样帧同步信号输入

1：在发送时钟上升沿移出数据输出（数据和帧同步信号），在发送时钟下降沿采样帧同步信号输入

CKI 仅影响发送时钟而不影响输出时钟信号。

- CKG：发送时钟选通选择

CKG	发送时钟选通
0x0	无，连续时钟
0x1	只有 TF 低电平时发送时钟才使能
0x2	只有 TF 高电平时发送时钟才使能
0x3	保留

- START：发送启动选择

START	发送启动
0x0	连续，只要向 SSC_THR 寄存器写入一个字（若发送使能），则在前一个数传输结束后立即启动
0x1	接收启动
0x2	检测 TF 低电平信号
0x3	检测 TF 高电平信号
0x4	检测 TF 下降沿信号

0x5	检测 TF 上升沿信号
0x6	检测 TF 任何电平变化信号
0x7	检测 TF 任何边沿信号
0x8-0xF	保留

- STTDLY：发送启动延迟

若 STTDLY 不为 0，则在启动事件和实际数据发送启动之间插入一个 STTDLY 时钟周期延迟。当编程发送器和接收器同步启动时，延迟仍然适用。

注意：必须仔细设置 STTDLY。若 STTDLY 对于 TAG（发送同步数据）发射太短，则数据代替 TAG 结束发送

- PERIOD：发送周期分频器选择

此域选择应用于所选发送时钟的分频器以产生一个新的帧同步信号。若为 0.不产生周期信号。若不为 0，在每 2 x (PERIOD+1)个发送周期时产生一个周期信号。

32.8.6 SSC 发送帧模式寄存器

名称：SSC_TFMR

访问类型：读/写

- DATLEN：数据长度

0：禁用值（不支持 1 位数据长度）

任何其它值：比特流含 DATLEN + 1 个数据位。此外，它定义了发送器的 PDC2 执行的传输大小。若 DATIEN 低于或等于 7，则以字节传输数据，若 DATLEN 在 8 到 15 之间（包含 15），则以半字传输，对任何其它值，以 32 位字传输。

- DATDEF：数据缺省值

此位定义发送输出时 TD 引脚上的驱动电平。注意若 PIO 控制器将此引脚定义为多驱动，则只有在 SCC TD 输出为 1 时该引脚使能

- MSBF：最高位在先

0：首先移出数据寄存器的最低位

1：首先移出数据寄存器的最高位

- DATNB：每帧数据数

此域定义每次传输启动后待传输的数据字数，其值等于(DATNB +1)

- FSLEN：发送帧同步长度

若 FSDEN 为 1，此域定义发送帧同步信号的长度和来自发送同步数据寄存器移出的比特数。此域和 FSLEN_EXT 被用来决定发送帧同步信号的脉冲长度。

脉冲长度等于 FSLEN + 1 个发送时钟周期

- FSOS：发送帧同步输出选择

FSOS	所选发送帧同步信号	TF 引脚
0x0	无	仅输入
0x1	负脉冲	输出
0x2	正脉冲	输出
0x3	数据传输期间驱动为低电平	输出
0x4	数据传输期间驱动为高电平	输出
0x5	每次数据传输启动时切换	输出
0x6-0x7	保留	未定义

- FSDEN：帧同步数据使能

0：发送帧同步信号内用缺省值驱动 TD 口线

1：发送帧同步信号的发送期间移出 SSC_TSHR 值

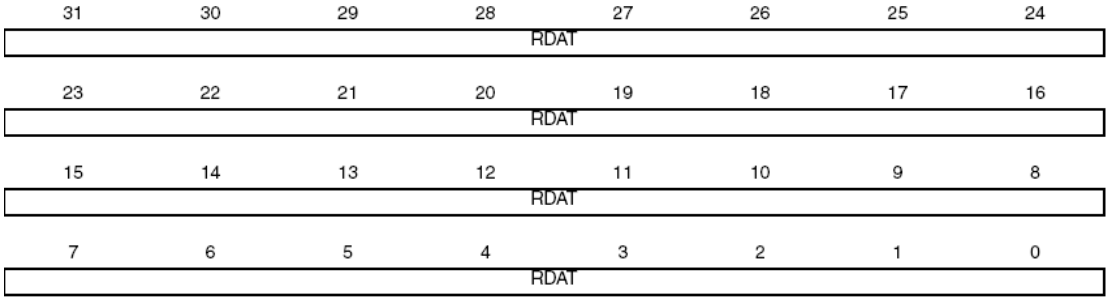
- FSEDGE：帧同步边沿检测

定义哪个边沿将产生中断 TXSYN（状态寄存器）

32.8.7 SSC 接收保持寄存器

名称：SSC_RHR

访问类型：只读



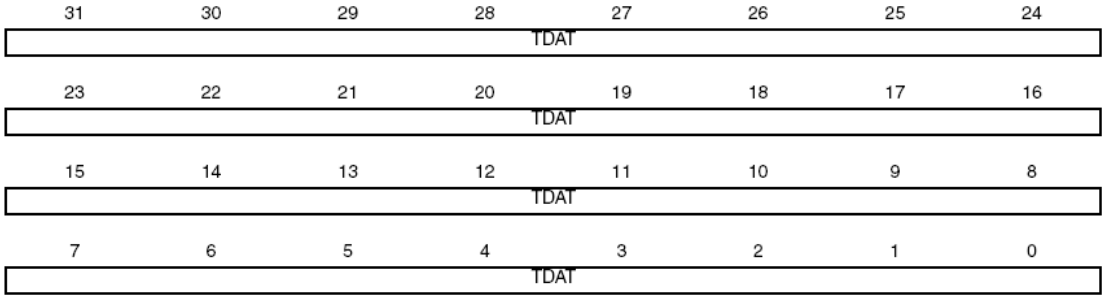
- RDAT：接收数据

不管 SSC_RFMR 中 DATLEN 定义的数据位数，数据右对齐

32.8.8 SSC 发送保持寄存器

名称：SSC_THR

访问类型：只写



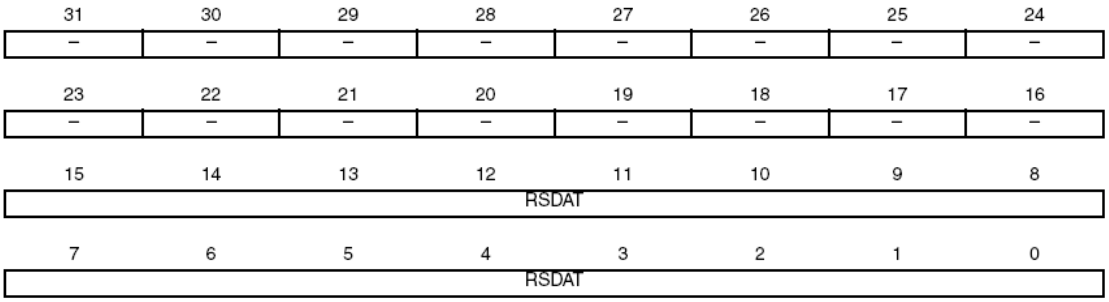
- TDAT：发送数据

不管 SSC_TFMR 中 DATLEN 定义的数据位数，数据右对齐

32.8.9 SSC 接收同步保持寄存器

名称：SSC_RSHR

方外类型：只读

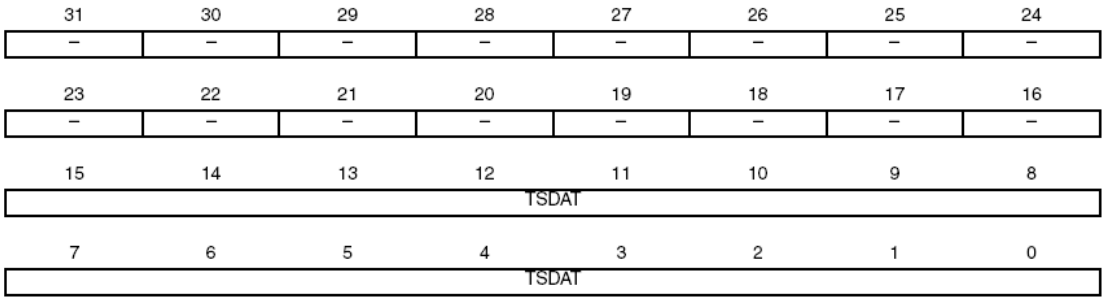


- RSDAT：接收同步数据

32.8.10 SSC 发送同步保持寄存器

名称：SSC_TSHR

访问类型：读/写

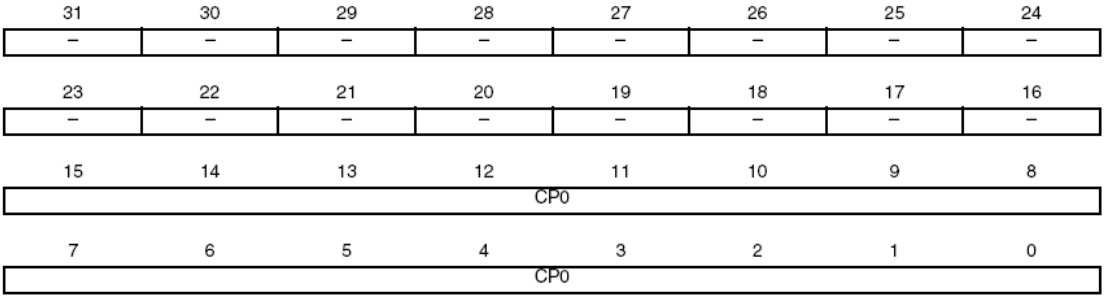


- TSDAT：发送同步数据

32.8.11 SSC 接收 Compare0 寄存器

名称：SSC_RC0R

访问类型：读/写



- CP0：接收比较数据 0

32.8.12 SSC 接收 Compare1 寄存器

名称：SSC_RC1R

访问类型：读/写

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
–	–	–	–	–	–	–	–
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
CP1							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CP1							

- CP1：接收比较数据 1

32.8.13 SSC 状态寄存器

名称：SSC_SR

访问类型：只读

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
–	–	–	–	–	–	RXEN	TXEN
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
–	–	–	–	RXSYN	TXSYN	CP1	CP0
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
RXBUFF	ENDRX	OVRUN	RXRDY	TXBUFE	ENDTX	TXEMPTY	TXRDY

- TXRDY：发送就绪

0：数据被装入 SSC_THR 并且等待被装入发送移位寄存器(TSR)

1：SSC_THR 空

- TSEMPTY：发送空

0：数据保持在 SSC_THR 或正从 TSR 发送

1：上次写入 SSC_THR 的数据已被装载到 TSR 并且上次被装载到 TSR 的数据已被发送

- ENDTX：发送结束

0：自上次写 SSC_TCR 或 SSC_TNCR 寄存器 SSC_TCR 未达到 0

1：自上次写 SSC_TCR 或 SSC_TNCR 寄存器 SSC_TCR 已达到 0

- TXBUFE：发送缓冲区空

0：SSC_TCR 或 SSC_TNCR 值不为 0

1：SSC_TCR 和 SSC_TNCR 的值都为 0

- RXRDY：接收就绪

0：SSC_RHR 为空

1：数据已接收并装载到 SSC_RHR

- OVRUN：接收超限

0：自上次读取状态寄存器，当前一次数据未读取时不向 SSC_RHR 装载数据

1：自上次读取状态寄存器，当前一次数据未读取时向 SSC_RHR 装载数据

- ENDRX：接收结束

0：数据写入接收计数器寄存器或接收下一次计数器寄存器

1：当接收计数器寄存器达到 0 时 PDC 传输结束

- RXBUFF：接收缓冲满

0：SSC_RCR 或 SSC_RNCR 值不为 0

1:SSC_RCR 和 SSC_RNCR 值都为 0

- CP0：Compare0

0：自上次读取状态寄存器未产生 Compare0

1：自上次读取状态寄存器未产生 Compare0

- CP1：Compare1

0：自上次读取状态寄存器未产生 Compare1

1：自上次读取状态寄存器未产生 Compare1

- TXSYN：发送同步

0：自上次读取状态寄存器未产生 Tx 同步

1：自上次读取状态寄存器产生 Tx 同步

- RXSYN：接收同步

0：自上次读取状态寄存器未产生 Rx 同步

1：自上次读取状态寄存器产生 Rx 同步

- TXEN：发送使能

0：发送禁用

1：发送使能

- RXEN：接收使能

0：接收禁用

1：接收使能

32.8.14 SSC 中断使能寄存器

名称：SSC_IER

访问类型：只写

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	RXSYN	TXSYN	CP1	CP0
7	6	5	4	3	2	1	0
RXBUFF	ENDRX	OVRUN	RXRDY	TXBUFE	ENDTX	TXEMPTY	TXRDY

- TXRDY：发送就绪中断使能

0：无效

1：时钟发送就绪中断

- TXEMPTY：发送空中断使能

0：无效

1：时钟发送空中断

- ENDTX：发送结束中断使能

0：无效

1：时钟发送结束中断

- TXBUFE：发送缓冲空中断使能

0：无效
1：时钟发送缓冲空中断

- RXRDY：接收就绪中断使能

0：无效
1：时钟接收就绪中断

- OVRUN：接收超限中断使能

0：无效
1：使能接收超限中断

- ENDRX：接收结束中断使能

0：无效
1：时钟接收结束中断

- RXBUFF：接收缓冲区满中断使能

0：无效
1：使能接收缓冲器满中断

- CP0：Compare0 中断使能

0：无效
1：使能 Compare0 中断

- CP1：Compare1 中断使能

0：无效
1：使能 Compare1 中断

- TXSYN：Tx 同步中断使能

0：无效
1：使能 Tx 同步中断

- RXSYN：RX 同步中断使能

0：无效
1：使能 RX 同步中断

32.8.15 SSC 中断禁用寄存器

名称：SSC_IDR
访问类型：只写

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	RXSYN	TXSYN	CP1	CP0
7	6	5	4	3	2	1	0
RXBUFF	ENDRX	OVRUN	RXRDY	TXBUFE	ENDTX	TXEMPTY	TXRDY

- TXRDY：发送就绪中断禁用

0：无效
1：时钟发送就绪中断

- TXEMPTY：发送空中断禁用

0：无效
1：时钟发送空中断

- ENDTX：发送结束中断禁用

0：无效

1：时钟发送结束中断

- TXBUFE：发送缓冲空中断禁用

0：无效

1：时钟发送缓冲空中断

- RXRDY：接收就绪中断禁用

0：无效

1：时钟接收就绪中断

- OVRUN：接收超限中断禁用

0：无效

1：禁用接收超限中断

- ENDRX：接收结束中断禁用

0：无效

1：时钟接收结束中断

- RXBUFF：接收缓冲区满中断禁用

0：无效

1：禁用接收缓冲器满中断

- CP0：Compare0 中断禁用

0：无效

1：禁用 Compare0 中断

- CP1：Compare1 中断禁用

0：无效

1：禁用 Compare1 中断

- TXSYN：Tx 同步中断禁用

0：无效

1：禁用 Tx 同步中断

- RXSYN：RX 同步中断禁用

0：无效

1：禁用 RX 同步中断

32.8.16 SSC 中断屏蔽寄存器

名称：SSC_IMR

访问类型：只读

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	RXSYN	TXSYN	CP1	CP0
7	6	5	4	3	2	1	0
RXBUF	ENDRX	OVRUN	RXRDY	TXBUFE	ENDTX	TXEMPTY	TXRDY

- TXRDY：发送就绪中断屏蔽

0：无效

1：时钟发送就绪中断

- TXEMPTY：发送空中断屏蔽

0：无效

1：时钟发送空中断

- ENDTX：发送结束中断屏蔽

0：无效

1：时钟发送结束中断

- TXBUFE：发送缓冲空中断屏蔽

0：无效

1：时钟发送缓冲空中断

- RXRDY：接收就绪中断屏蔽

0：无效

1：时钟接收就绪中断

- OVRUN：接收超限中断屏蔽

0：无效

1：屏蔽接收超限中断

- ENDRX：接收结束中断屏蔽

0：无效

1：时钟接收结束中断

- RXBUFF：接收缓冲区满中断屏蔽

0：无效

1：屏蔽接收缓冲器满中断

- CP0：Compare0 中断屏蔽

0：无效

1：屏蔽 Compare0 中断

- CP1：Compare1 中断屏蔽

0：无效

1：屏蔽 Compare1 中断

- TXSYN：Tx 同步中断屏蔽

0：无效

1：屏蔽 Tx 同步中断

- RXSYN：RX 同步中断屏蔽

0：无效

1：屏蔽 RX 同步中断

33 定时器/计数器（TC）

33.1 描述

定时器/计数器（TC）包含三个相同的 16 位定时器/计数器通道。

每个通道可独立编程，以执行不同功能包括频率测量，事件计数，间断测量，脉冲发生，延迟定时和脉宽调制。

每个通道有由用户可配置的三个外部时钟输入，五个内部时钟输入和两个多用输入/输出信号。每个通道驱动一个内部中断信号，此信号可被配置用来发生处理器中断。

定时器/计数器部件有两个作用于这三个 TC 通道的全局寄存器。

块控制寄存器允许用同样的指令同时启动这三个通道。

块模式寄存器为每个通道定义了外部时钟输入，允许将它们链接。

表 33-1 给出了设备定时器/计数器部件输入共同赋值给定时器计数器 0 到 2

表 33-1　定时器计数器部件赋值

名称	定义
TIMER_CLOCK1	MCK/2
TIMER_CLOCK2	MCK/8
TIMER_CLOCK3	MCK/32
TIMER_CLOCK4	MCK/128
TIMER_CLOCK5	SLCK

33.2 框图

图 33-1　定时器/计数器框图

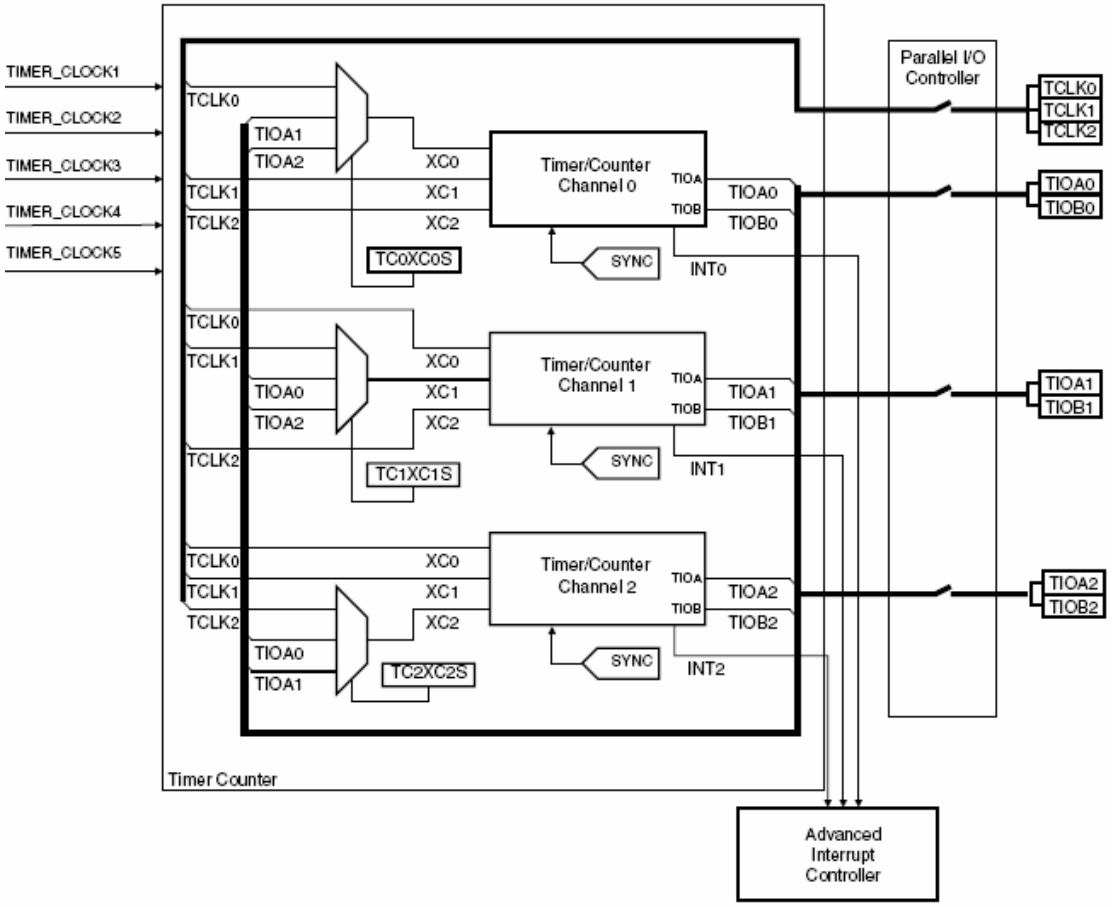


表 33-2　信号名称描述

部件/通道	信号名称	描述
通道信号	XC0，XC1，XC2	外部时钟输入
	TIOA	捕捉模式：定时器/计数器输入定时器 波形模式：定时器/计数器输出
	TIOB	捕捉模式：定时器/计数器输入定时器 波形模式：定时器/计数器输入/输出
	INT	中断信号输出
	SYNC	同步化输入信号

33.3　　引脚名称列表

表 33-3　TC 引脚列表

引脚名称	描述	类型
TCLK0-TCLK2	外部时钟输入	输入
TIOA0-TIOA2	I/O 口线 A	输入/输出
TIOB0-TIOB2	I/O 口线 B	输入/输出

33.4　　附属产品

33.4.1　I/O 口线

引脚被用来连接可与 PIO 口线复用的兼容的外部设备。

编程者必须首先编程 PIO 控制器来给 TC 引脚指定外设功能。

33.4.2　电源管理

TC 通过电源管理控制器（PMC）协调同步，因此编程者必须首先配置 PMC

来了使能定时器/计数器

33.4.3 中断

TC 有一个连接于高级中断控制器（AIC）的中断口线。处理 TC 中断需要在配置 TC 前编程 AIC

33.5 功能描述

33.5.1 TC 描述

定时器/计数器的三个通道相互独立，但操作相同。通道寄存器编程见 501 页表 33-5 列表

33.5.2 16 位计数器

每个通道有一个 16 位计数器。计数器值在所选时钟的每个正边沿递增。当计数器达到 0xFFFF 值并且变为 0x0000 时，发生溢出并且 TC_SR（状态寄存器）中的 COVFS 位置位。

计数器的当前值可通过读取计数器值寄存器 TC_CV 实时访问。可通过一个触发复位计数器。此时，计数器值在所选时钟的下一个有效边沿上变为 0x0000.

33.5.3 时钟选择

在 TC 块级上，每个通道的输入时钟信号通过编程 TC_BMR（部件模式）可以与外部时钟 TCLK0, TCLK1 或 TCLK2 连接,或与内部 I/O 信号 TIOA0, TIOA1 或 TIOA2 连接。见 489 页的图 33-2

每个通道可独立的为其计数器选择一个内部或外部时钟源：

- 内部时钟信号：TIMER_CLOCK1, TIMER_CLOCK2, TIMER_CLOCK3, TIMER_CLOCK4, TIMER_CLOCK5
- 外部时钟信号：XC0, XC1 or XC2

可通过 TC 通道模式寄存器的 TCCLKS 位完成选择。

可用 TC_CMR 中的 CLKI 位反转所选时钟。这就允许在时钟的负边沿计数。

脉冲功能使外部信号为高时时钟有效。模式寄存器中的 BURST 参数定义此信号（无，XC0, XC1, XC2）。见 489 页图 33-3

注意：此时，若使用外部时钟，TC 部件每个级的持续时间必须比主控时钟周期长。外部时钟频率必须至少比主控时钟低 2.5 倍。

图 33-2 时钟选择

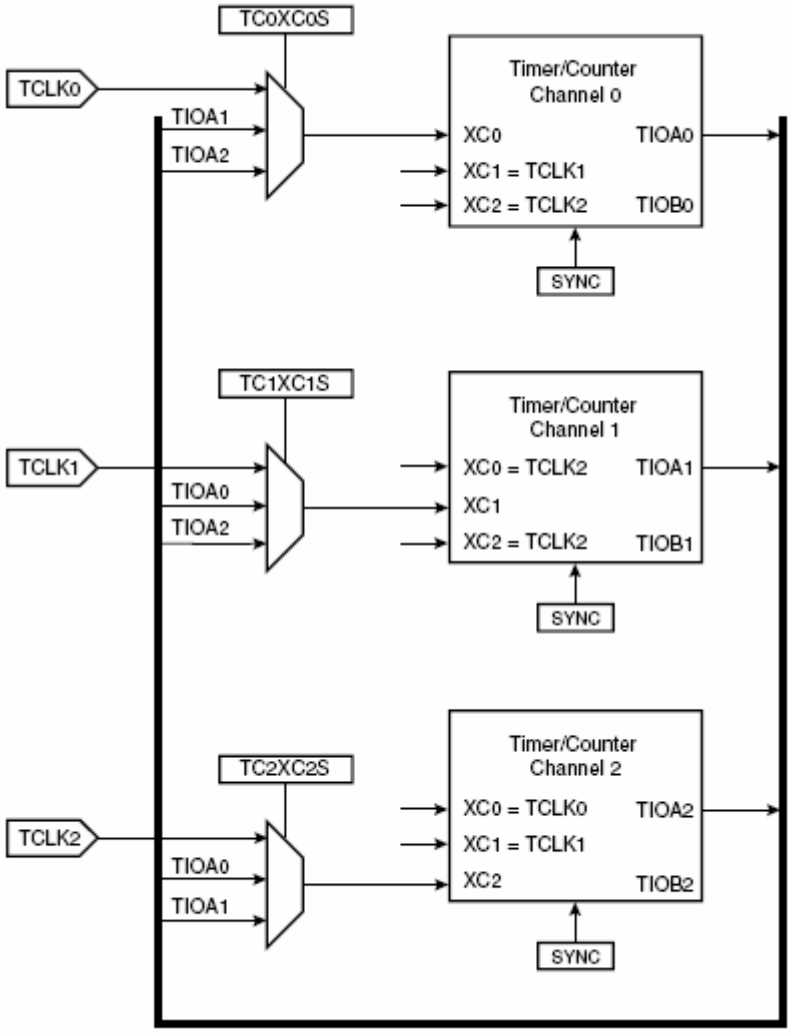
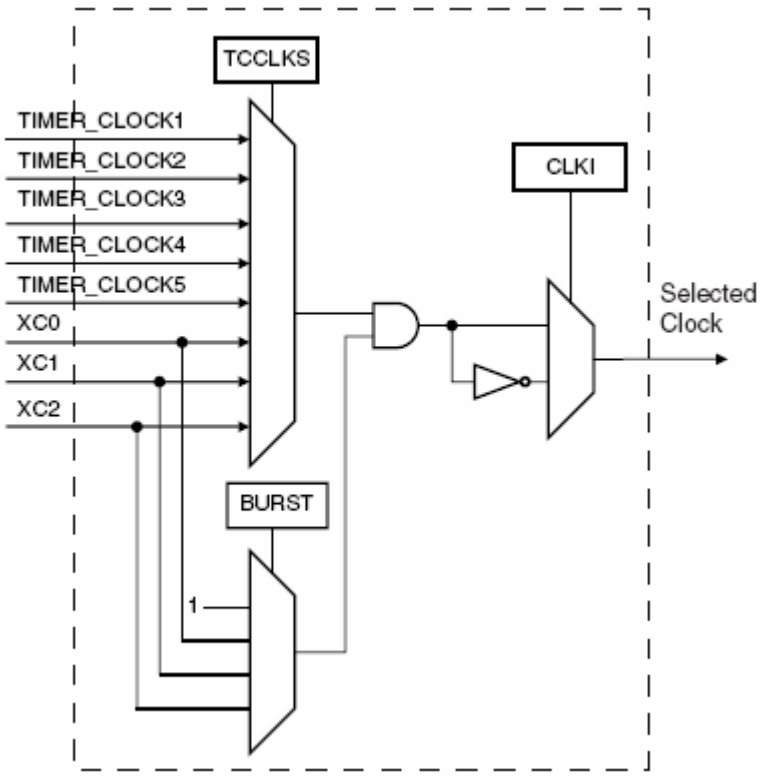

TC0XC0S
TCLK0
TIOA1
TIOA2
Timer/Counter
Channel 0
XC0
TIOA0
XC1 = TCLK1
XC2 = TCLK2
TIOB0
SYNC
TC1XC1S
TCLK1
TIOA0
TIOA2
Timer/Counter
Channel 1
XC0 = TCLK2
TIOA1
XC1
XC2 = TCLK2
TIOB1
SYNC
TC2XC2S
TCLK2
TIOA0
TIOA1
Timer/Counter
Channel 2
XC0 = TCLK0
TIOA2
XC1 = TCLK1
XC2
TIOB2
SYNC

图 33-3　时钟选择

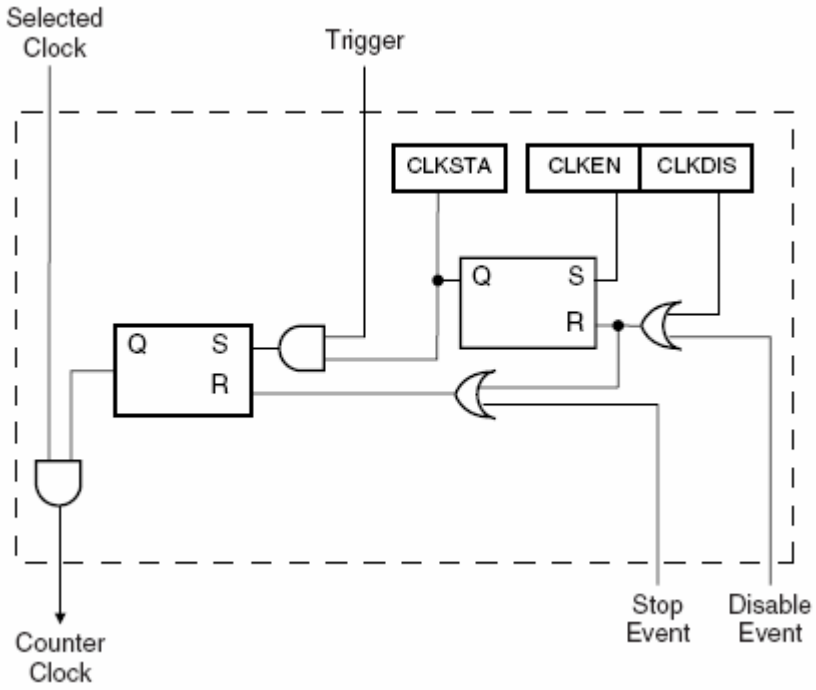


33.5.4　时钟控制

可用两个不同的方法控制每个计数器的时钟：可以使能/禁用和启动/停止。见图 33-4

- 用户可用控制寄存器中的 CLKEN 和 CLKDIS 命令使能或禁用时钟。在捕捉模式若 TC_CMR 中的 LDBDIS 置 1 则可由 RB 装载事件禁用时钟。在波形模式，若 TC_CMR 中的 CPCDIS 置 1 则由 RC 比较事件禁用。当禁用时，启动或停止作用无效：仅控制寄存器中的 CLKEN 命令可重新使能时钟。当时钟使能时，状态寄存器中的 CLKSTA 置位。
- 还可启动或停止时钟：一个触发（软件，同步，外部或比较）通常启动时钟。在捕捉模式(TC_CMR 中 LDBSTOP = 1)由 RB 装载事件或在波形模式(TC_CMR 中 CPCSTOP = 1）由 RC 比较时间停止时钟。启动和停止命令尽在时钟使能的时候有效。

图 33-4　时钟控制



33.5.5 TC 操作模式

每个通道可以两种不同的模式独立的运行：

- 捕捉模式提供信号测量
- 波形模式用来产生波形

用 TC 通道模式寄存器中的 WAVE 位编程 TC 运行模式

在捕捉模式，TIOA 和 TIOB 被配置为输入。

在波形模式，TIOA 通常被配置为输出而若 TIOB 未被选作外部触发时为输出。

33.5.6 触发器

触发器复位计数器并启动计数器时钟。两种模式下有通用三种触发器，第四种外部触发器适用于每种模式。

以下触发器通用于两种模式：

- 软件触发器：每个通道有一个软件触发器，通过设置 TC_CCR 中的 SWTRG 有效。
- SYNC：每个通道有一个同步信号 SYNC。当有效时，此信号和软件触发器有同样的效果。通过用 SYNC 置位写 TC_BCR（部件控制）同时激活所有通道的 SYNC 信号
- 比较 RC 触发器：RC 在每个通道中执行，并且若 TC_CMR 中的 CPCTRG 置位，当计数器值和 RC 值相等则提供一个触发。

还可配置通道为有一个外部触发器。在捕获模式下，外部触发器信号可在 TIOA 和 TIOB 选择。在波形模式，可在以下信号上编程外部事件：TIOB, XC0, XC1 或 XC2。可通过写 TC_CMR 中的 ENETRG 编程此外部事件了执行一个触发器

若使用外部触发完成，脉冲持续时间必须比指控周期长，以便对其检测。

不管是否使用触发器，它将在后续选定时钟的有效边沿记录。这就意味着计数器值在触发后是不为零的其它值，特别是当选择低频率信号作为时钟的时候。

33.5.7 捕获工作模式

通过清零 TC_CMR（通道模式寄存器）中的 WAVE 参数进入此模式。

捕获模式允许 TC 通道对输入的 TIOA 和 TIOB 信号上的脉冲时序，频率，周期，占空比和相位进行测量。

图 33-5 展示了当编程为捕获模式时 TC 通道的配置。

33.5.8 捕获寄存器 A 和 B

寄存器 A 和 B（RA 和 RB)被用作捕获寄存器。这就是说当在 TIOA 信号上产生一个可编程事件时可用计数器值装载 RA 和 RB。TC_CMR 中的 LDTA 参数定义装载寄存器 A 的 TIOA 边沿，LDTB 参数定义装载寄存器 B 的 TIOA 边沿。

只有在自上次触发后还未装载 RA 或自上次装载 RA 已装载 RB 时才装载 RA。

只有在自上次触发后已装载 RA 或最后装载 RB 时才装载 RB。

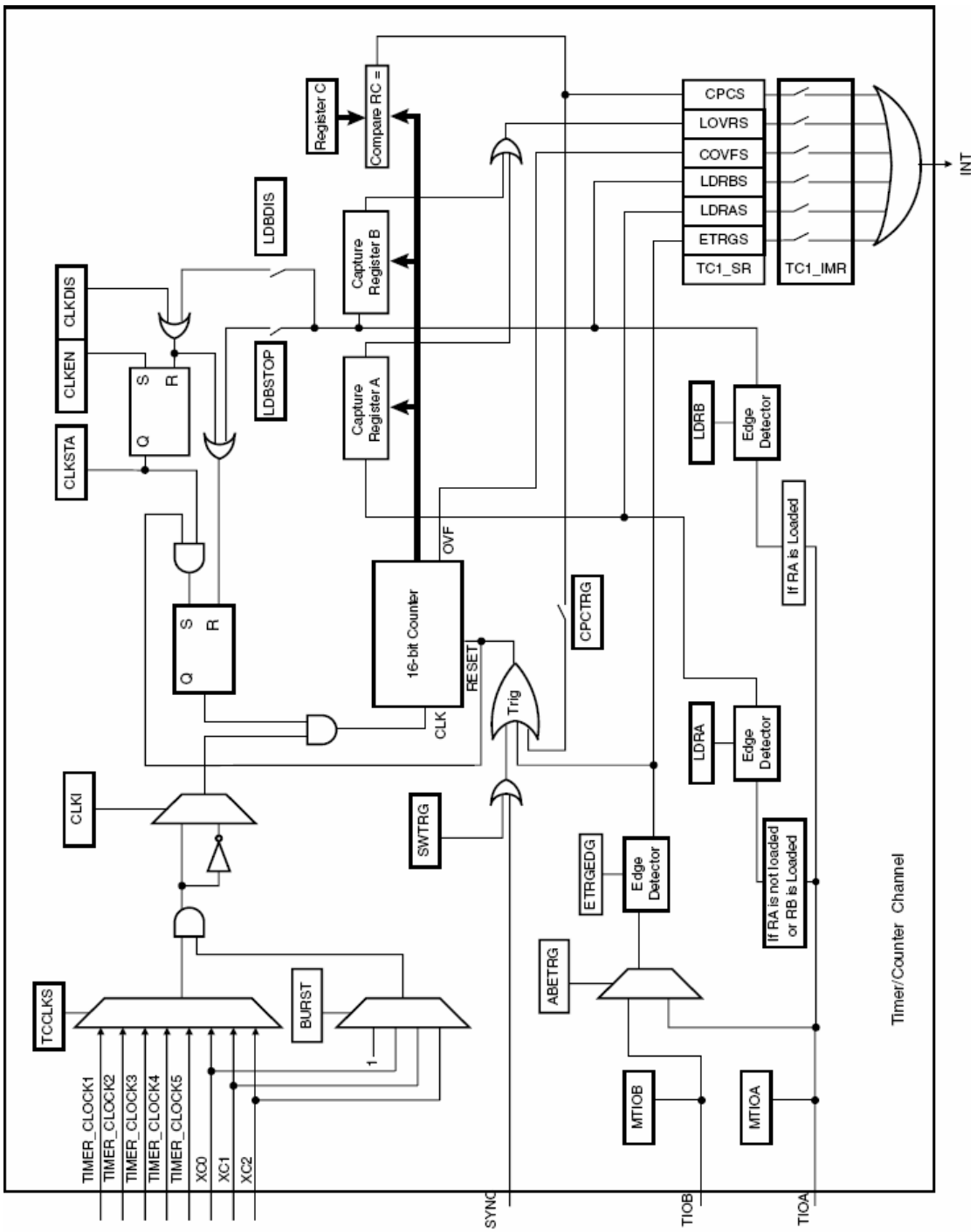
在读取上次装载值前装载 RA 或 RB 将设置 TC_SR（状态寄存器）中的溢出错误标志位(LOVRS)。此时，将覆盖旧值。

33.5.9 触发条件

除了 SYNC 信号，软件触发器和 RC 比较触发器，还可定义外部触发。

TC_CMR 中的 ABETRG 位选择 TIOA 或 TIOB 输入信号作为外部触发。ETRGEDG 参数定义产生外部触发的检测边沿。若 ETRGEDG = 0（无），则禁用外部触发。

图 33-5 捕捉模式



33.5.10　波形工作模式

通过设置 TC_CMR（通道模式寄存器）中的 WAVE 参数进入波形工作模式。

波形工作模式下，TC 通道产生 1 个或 2 个相同频率和独立可编程占空比的 PWM 信号，或产生不同类型的单发射或重复脉冲

在此模式下，TIOA 被配置为输出而若 TIOB 未被用作外部事件（TC_CMR 中的 EEVT 参数）时则定义 TIOB 为输出。

图 33-6 展示了当在波形工作模式下编程的 TC 通道配置。

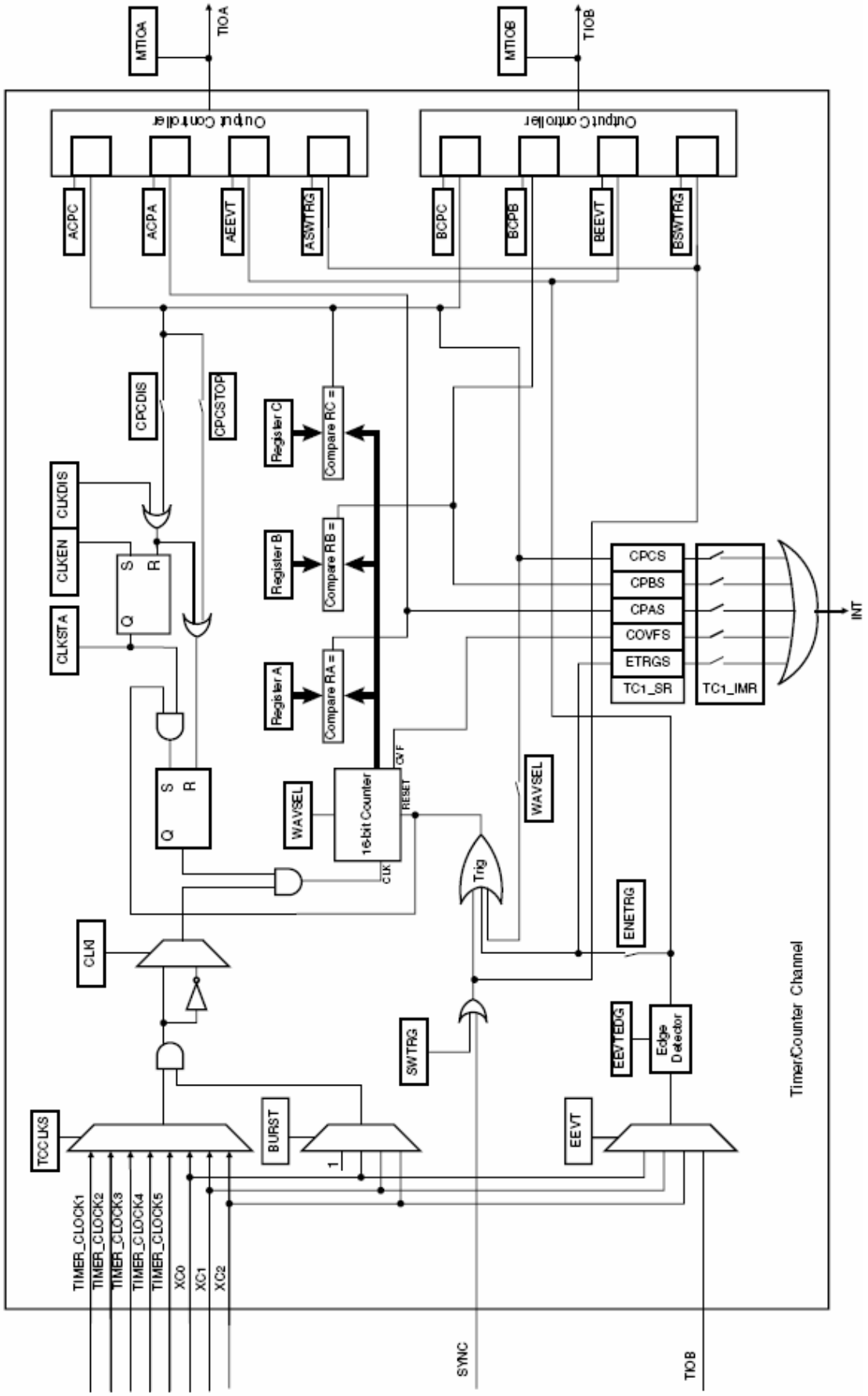
33.5.11 波形选择

TC_CV 的动作随 TC_CMR（通道模式寄存器）中的 WAVSEL 参数而变

不管如何选择，RA，RB 和 RC 都可被用作比较寄存器。

RA 比较器被用于控制 TIOA 输出，RB 比较器被用于控制 TIOB 输出（若正确配置）而 RC 比较器被用于控制 TIOA 和/或 TIOB 输出

图 33-6 波形模式


MTIOA
TIOA
MTIOB
TIOB
Output Controller
Output Controller
ACPC
ACPA
AEEVT
ASWTRG
BCPC
BCPB
BEEVT
BSWTRG
CPCDIS
CPCSTOP
Register C
Compare RC =
CLKDIS
CLKEN
CLKSTA
Register B
Compare RB =
Register A
Compare RA =
S
R
Q
CPCS
CPBS
CPAS
COVFS
ETRGS
TC1_SR
TC1_IMR
INT
WAVSEL
16-bit Counter
CLK
RESET
OVF
WAVSEL
Trig
ENETRG
CLKI
SWTRG
EEVTEDG
Edge Detector
TCCLKS
BURST
1
EEVT
TIMER_CLOCK1
TIMER_CLOCK2
TIMER_CLOCK3
TIMER_CLOCK4
TIMER_CLOCK5
XC0
XC1
XC2
SYNC
TIOB
Timer/Counter Channel

33.5.11.1 WAVSEL=00

当 WAVSEL=00，TC_CV 值从 0 递增到 0xFFFF。一旦达到 0xFFFF，则 TC_CV 值复位。TC_CV 重新开始递增并且继续循环。见图 33-7

一个外部事件触发或软件触发可复位 TC_CV 值。要注意随时可能产生触发，这点很重要。见图 33-8

该配置下不能编程 RC 比较器来产生触发。同时，RC 比较器可以停止计数器时钟(TC_CMR 中的 CPCSTOP = 1)与/或禁用计数器时钟(TC_CMR 中的 CPCDIS = 1)。

图 33-7 无触发的 WAVSEL= 00

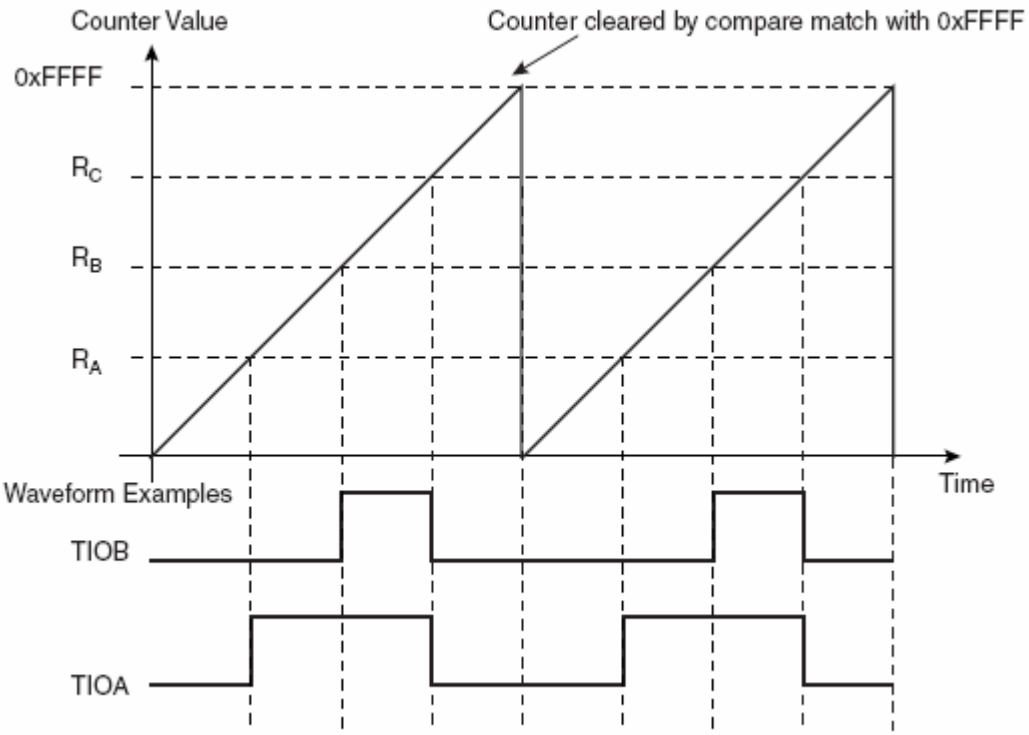
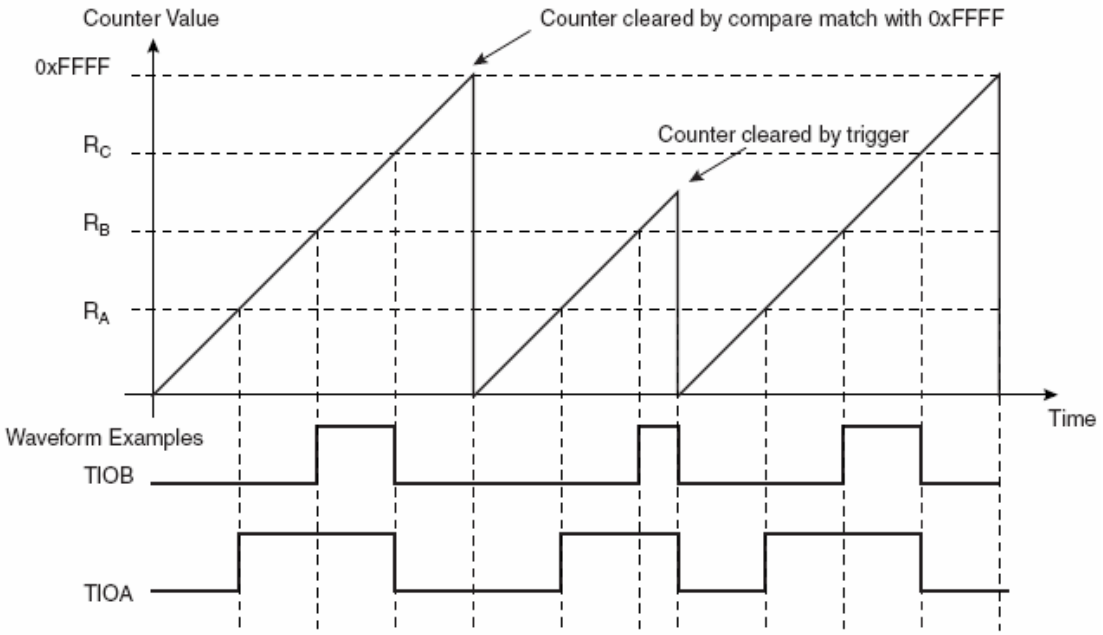


图 33-8 带触发的 WAVSEL= 00



33.5.11.2 WAVSEL = 10

当 WAVSEL = 10, TC_CV 值从 0 递增到 RC 值，接着自动复位为 RC 比较器值。一旦 TC_CV 值已复位，接着继续递增。见图 33-9

注意 TC_CV 在任何时候可被外部事件或软件触发复位（若两者都正确编程），这点很重要。见图 33-10

另外，RC 比较器可停止计数器时钟(TC_CMR 中的 CPCSTOP = 1）与/或禁用计数器时钟（TC_CMR 中的 CPCDIS = 1）

图 33-9 无触发的 WAVSEL = 10

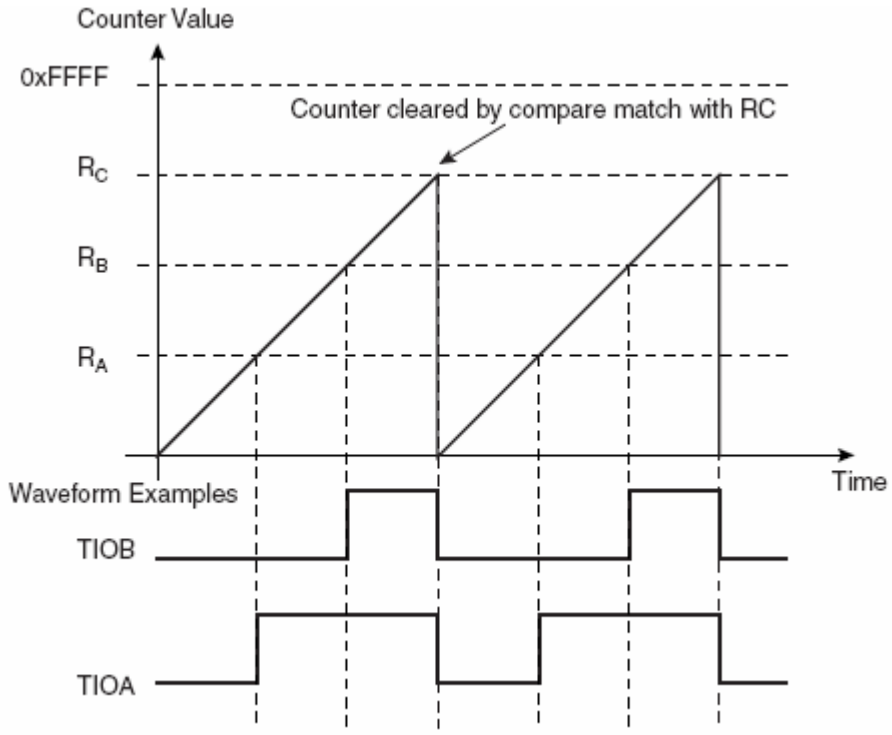
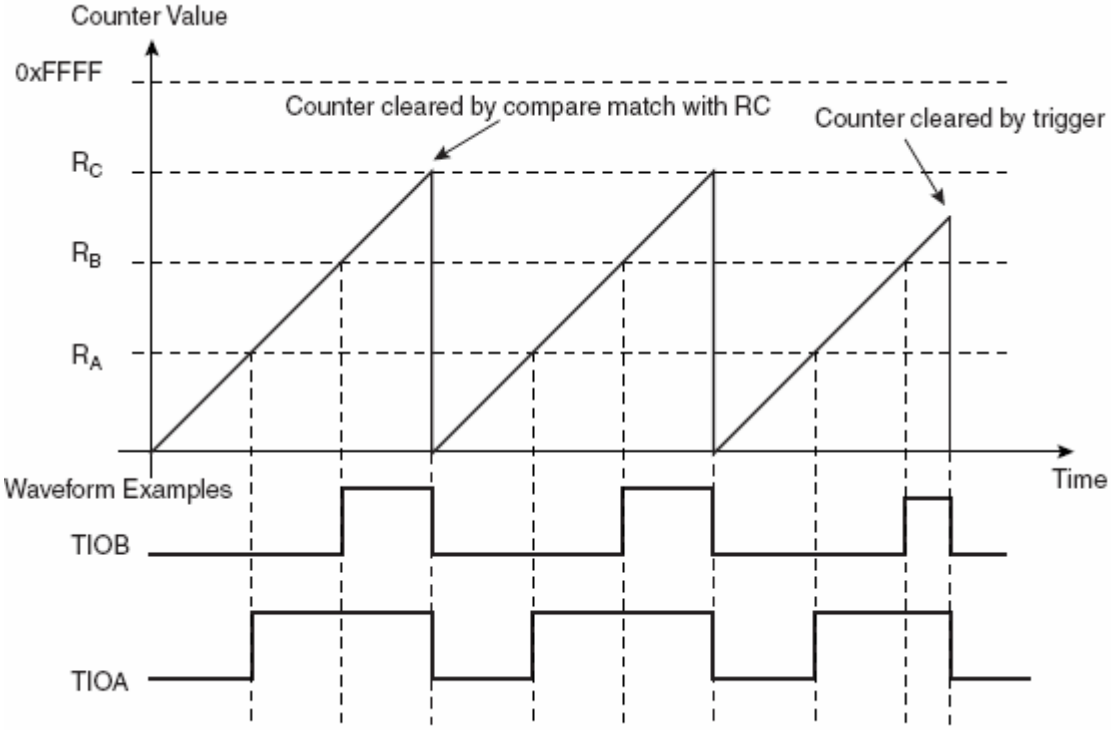


图 33-10 带触发的 WAVSEL = 10



33.5.11.3 WAVSEL = 01

当 WAVSEL = 01,TC_CV 值从 0 递增到 0xFFFF。一旦达到 0xFFFF，TC_CV 值递减到 0，接着重新递增到 0xFFFF，一直继续下去。见图 33-11

像外部事件或软件触发这样的触发可在任何时间修改 TC_CV。若当 TC_CV 正在递增时产生触发，则 TC_CV 接下来递减。若当 TC_CV 正在递减时接收到触发，则 TC_CV 接着递增。见图 33-12

该配置下不能编程 RC 比较器来产生触发。

同时，RC 比较器可以停止计数器时钟(CPCSTOP = 1)与/或禁用计数器时钟(CPCDIS = 1).

图 33-11 无触发的 WAVSEL = 01

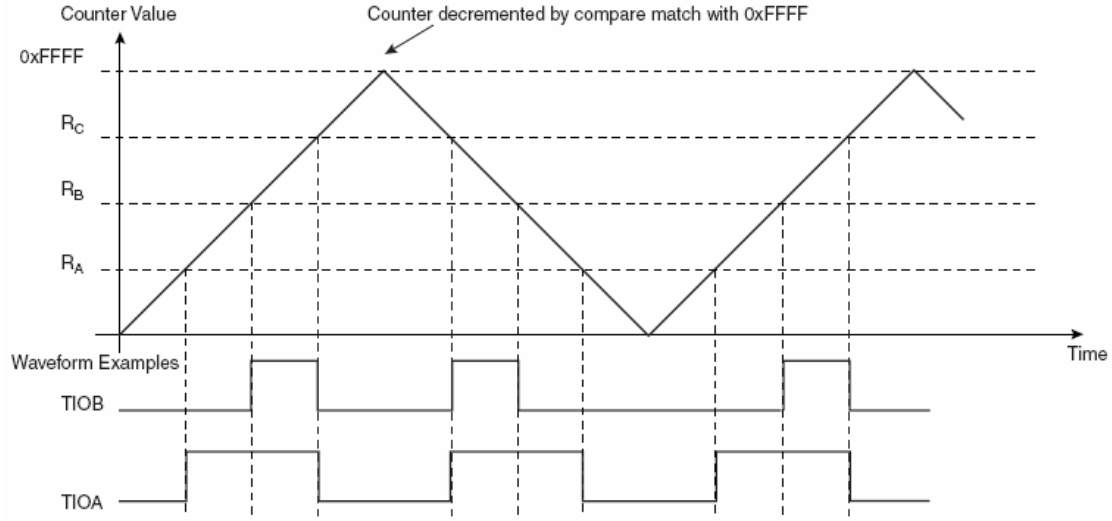
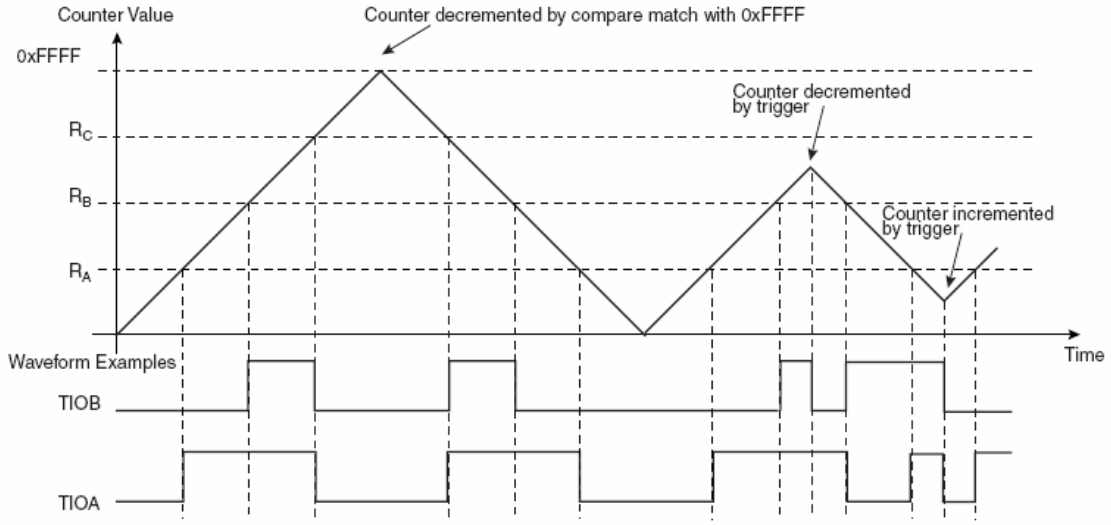


图 33-12 带触发的 WAVSEL = 01



33.5.11.44 WAVSEL = 11

当 WAVSEL = 11,TC_CV 值从 0 递增到 RC 值。一旦达到 RC 值，则 TC_CV 值递减到 0，接着重新递增到 RC，这样继续下去。见图 33-13

像外部事件或软件触发这样的触发可在任何时间修改 TC_CV 值。当 TC_CV 正在递增时产生触发，则 TC_CV 接下来递减。若当 TC_CV 正在递减时接收到触发，则 TC_CV 接下来递增。见图 33-14

RC 比较可停止计数器时钟(CPCSTOP = 1)与/或禁用计数器时钟(CPCDIS=1)

图 33-13 无触发时 WAVSEL = 11

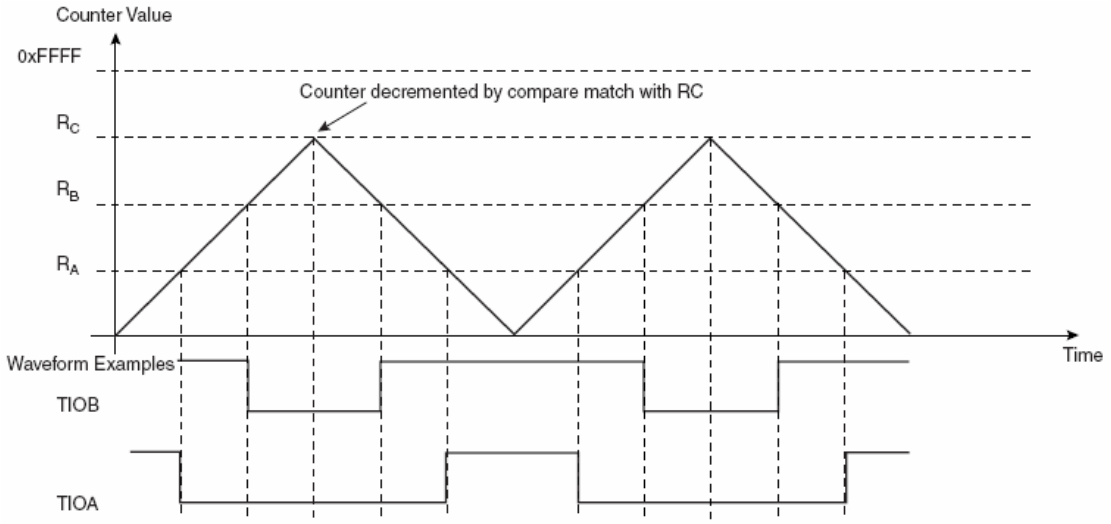
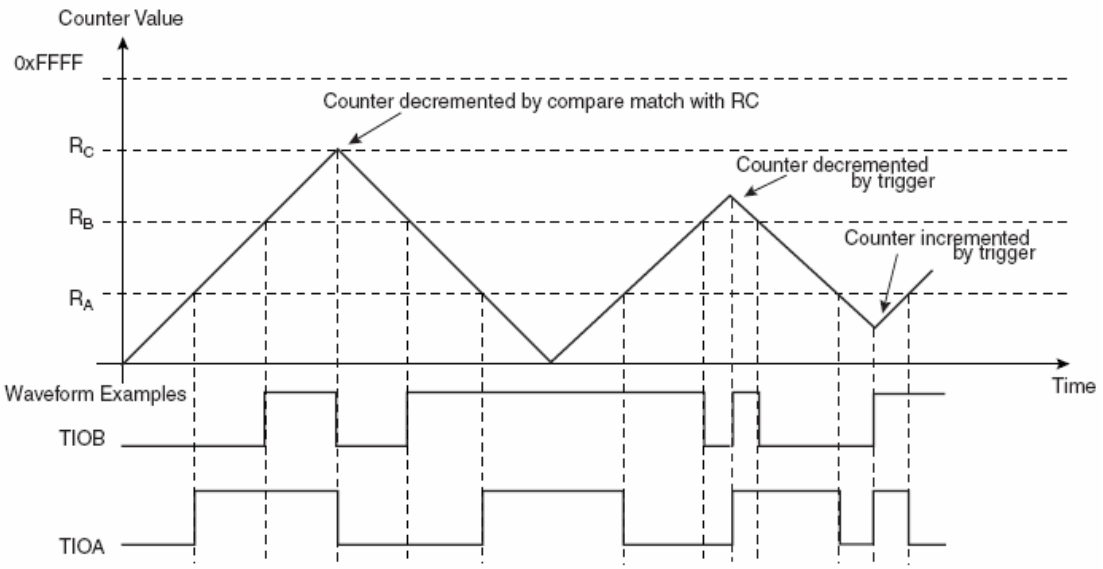


图 33-14 带触发时 WAVSEL = 11



33.5.12 外部事件/触发条件

可编程外部事件来检测时钟源（XC0, XC1,XC2）或 TIOB。接下来所选外部事件可用作触发。

TC_CMR 中的 EEVT 参数用来选择外部触发器。EEVTEDG 参数定义每个可能的外部触发（上升，下降或两者都）定义触发边沿。若 EEVTEDG 清零，则未定义外部事件。

若定义 TIOB 为外部事件信号(EEVT = 0)，则 TIOB 不再用作输出并且比较寄存器 B 不再用于产生波形，其次无 IRQ。此时 TC 通道仅能产生 TIOA 波形

当定义了外部事件,则可通过设置 TC_CMR 中的 ENETRG 位将外部事件用作触发。还可根据 WAVSEL 参数将 RC 比较器用作触发。

33.5.13 输出控制器

输出控制器定义了下一事件 TIOA 和 TIOB 上的输出电平变化。

只有在 TIOB 被定义为输出（不是外部事件）时使用 TIOB 控制

下列事件控制 TIOA 和 TIOB：软件触发，外部事件和 RC 比较器。

RA 比较控制 TIOA，RB 比较控制 TIOB。根据 TC_CMR 中相应参数的定义每个时间可编程置位，清零或切换输出。

33.6 定时器/计数器（TC）用户接口

表 33-4 TC 全局存储器映射

偏移量	通道/寄存器	名称	访问类型	复位值
0x00	TC Channel 0	见表 33-5		
0x40	TC Channel 1			
0x80	TC Channel 2			
0xC0	TC 部件控制寄存器	TC_BCR	只写	-
0xC4	TC 部件模式寄存器	TC_BMR	读写	0

TC_BCR（部件控制寄存器）和 TC_BMR（部件模式寄存器）控制整个 TC 部件。TC 通道由表 333-5 所列寄存器控制。表 33-5 中每个通道寄存器的偏移量和表 33-5 中提到相应通道的偏移量相关。

表 33-5 TC 通道存储映射

偏移量	寄存器	名称	访问类型	复位值
0x00	通道控制寄存器	TC_CCR	只写	–
0x04	通道模式寄存器	TC_CMR	读/写	0
0x08	保留			–
0x0C	保留			–
0x10	计数器值	TC_CV	只读	0
0x14	寄存器 A	TC_RA	读/写（1）	0
0x18	寄存器 B	TC_RB	读/写（1）	0
0x1C	寄存器 C	TC_RC	读/写	0
0x20	状态寄存器	TC_SR	只读	0
0x24	中断使能寄存器	TC_IER	只写	–
0x28	中断禁用寄存器	TC_IDR	只写	–
0x2C	中断屏蔽寄存器	TC_IMR	只读	0
0xFC	保留	–	–	–

33.6.1 TC 块控制寄存器

寄存器名称：TC_BCR

访问类型：只写

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

注意：1.若 WAVE=0 则只读

- SYNC：同步命令

0=无效

1=出现 SYNC 信号，给每个通道同时产生一个软件触发。

33.6.2 TC 块模式寄存器

寄存器名称：TC_BMR

访问类型：读/写

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	TC2XC2S		TCXC1S		TC0XC0S	

- TC0XC0S：外部时钟信号 0 选择

TC0XC0S		连接到 XC0 的信号
0	0	TCLK0
0	1	无

1	0	TIOA1
1	1	TIOA2

- TC1XC1S：外部时钟信号 1 选择

TC1XC1S		连接到 XC1 的信号
0	0	TCLK1
0	1	无
1	0	TIOA0
1	1	TIOA2

- TC2XC2S：外部时钟信号 2 选择

TC2XC2S		连接到 XC2 的信号
0	0	TCLK2
0	1	无
1	0	TIOA0
1	1	TIOA1

33.6.3 TC 通道控制寄存器

寄存器名称：TC_CCR

访问类型：只写

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

- CLKEN：计数器时钟使能命令

0=无效

1=若 CLKDIS 不为 1 则使能时钟

- CLKDIS：计数器时钟禁用命令

0=无效

1=禁用时钟

- SWTRG：软件触发命令

0=无效

1=软件触发执行：计数器复位并且启动时钟

33.6.4 TC 通道模式寄存器：捕获模式

寄存器名称：TC_CMR

访问类型：读/写

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	LDRB		LDRA	
15	14	13	12	11	10	9	8
WAVE = 0	CPCTRG	–	–	–	ABETRG	ETRGEDG	
7	6	5	4	3	2	1	0
LDBDIS	LDBSTOP	BURST		CLKI	TCCLKS		

- TCCLKS：时钟选择

TCCLKS			所选时钟
0	0	0	TIMER_CLOCK1
0	0	1	TIMER_CLOCK2
0	1	0	TIMER_CLOCK3
0	1	1	TIMER_CLOCK4
1	0	0	TIMER_CLOCK5
1	0	1	XC0
1	1	0	XC1
1	1	1	XC2

- CLKI：时钟反转

0=在时钟上升沿计数器递增

1=在时钟下降沿计数器递增

- BURST：突发信号选择

BURST		
0	0	时钟不是由外部信号选通
0	1	XC0 和所选时钟相与
1	0	XC1 和所选时钟相与
1	1	XC2 和所选时钟相与

- LDBSTOP：当装入 RB，计数器时钟停止

0=当发生 RB 装载时未停止计数器时钟

1=当发生 RB 装载时停止计数器时钟

- LDBDIS：当 RB 装载时计数器时钟禁用

0=当发生 RB 装载时未禁用计数器时钟

1=当发生 RB 装载时禁用计数器时钟

- ETRGEDG：外部触发边沿选择

ETRGEDG		边沿
0	0	无
0	1	上升边沿
1	0	下降边沿
1	1	每个边沿

- ABETRG：TIOA 或 TIOB 外部触发选择

0=TIOB 被用作外部触发器

1=TIOA 被用作外部触发器

- CPCTRG：RC 比较触发使能

0=RC 比较对计数器和其时钟无效

1=RC 比较复位计数器并且启动计数器时钟

- WAVE

0=捕获模式使能

1=捕获模式禁用（波形模式使能）

- LDRA：RA 装载选择

LDRA		边沿
0	0	无
0	1	TIOA 的上升沿
1	0	TIOA 的下降沿
1	1	TIOA 的每个沿

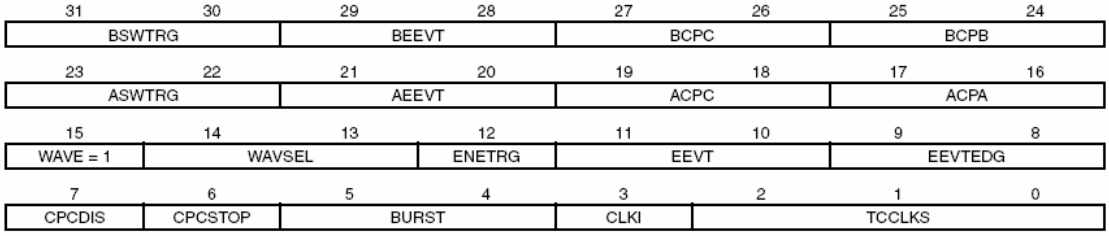
- LDRB：RB 装载选择

LDRB		边沿
0	0	无
0	1	TIOB 的上升沿
1	0	TIOB 的下降沿
1	1	TIOB 的每个沿

33.6.5 TC 通道模式寄存器：波形模式

寄存器名称：TC_CMR

访问类型：读/写



- TCCLKS：时钟选择

TCCLKS			所选时钟
0	0	0	TIMER_CLOCK1
0	0	1	TIMER_CLOCK2
0	1	0	TIMER_CLOCK3
0	1	1	TIMER_CLOCK4
1	0	0	TIMER_CLOCK5
1	0	1	XC0
1	1	0	XC1
1	1	1	XC2

- CLKI：时钟反转

0=在时钟上升沿计数器递增

1=在时钟下降沿计数器递增

- BURST：突发信号选择

BURST		
0	0	时钟不是由外部信号选通
0	1	XC0 和所选时钟相与
1	0	XC1 和所选时钟相与
1	1	XC2 和所选时钟相与

- CPCSTOP：RC 比较时计数器时钟停止

0=当计数器发送到 RC 值时不停止计数器时钟

1=当计数器发送到 RC 值时停止计数器时钟

- CPCDIS：RC 比较时计数器时钟禁用

0=当计数器达到 RC 值时不禁用计数器时钟
1=当计数器达到 RC 值时禁用计数器时钟

- EEVTEDG：外部事件边沿选择

EEVTEDG		边沿
0	0	无
0	1	上升沿
1	0	下降沿
1	1	每个边沿

- EEVT：外部事件选择

EEVT		信号被选做外部事件	TIOB 方向
0	0	TIOB	输入（1）
0	1	XC0	输出
1	0	XC1	输出
1	1	XC2	输出

注意：1.若 TIOB 被选做外部事件信号，则配置 TIOB 为输入并且不再产生波形并且随后无 IRQ

- ENETRG：外部事件触发使能

0=外部事件对计数器和其时钟无效。此时，所选外部事件仅控制 TIOA 输出。
1=外部事件复位计数器并且启动计数器时钟

- WAVSEL：波形选择

WAVSEL		效果
0	0	无 RC 比较自动触发的 UP 模式
0	1	带 RC 比较自动触发的 UP 模式
1	0	无 RC 比较自动触发的 UPDOWN 模式
1	1	带 RC 比较自动触发的 UPDOWN 模式

- WAVE=1

0=波形模式禁用（捕获模式使能）
1=波形模式使能

- ACPA：TIOA 上 RA 比较效果

ACPA		效果
0	0	无
0	1	置位
1	0	清零
1	1	切换

- ACPC：TIOA 上 RC 比较效果

ACPC		效果
0	0	无
0	1	置位
1	0	清零
1	1	切换

- AEEVT：TIOA 上外部事件效果

AEEVT		效果
0	0	无
0	1	置位
1	0	清零
1	1	切换

- ASWTRG：TIOA 上软件触发效果

ASWTRG		效果
0	0	无
0	1	置位
1	0	清零
1	1	切换

- BCPB：TIOB 上 RB 比较效果

BCPB		效果
0	0	无
0	1	置位
1	0	清零
1	1	切换

- BCPC：TIOB 上 RC 比较效果

BCPC		效果
0	0	无
0	1	置位
1	0	清零
1	1	切换

- BEEVT：TIOB 上外部事件效果

BEEVT		效果
0	0	无
0	1	置位
1	0	清零
1	1	切换

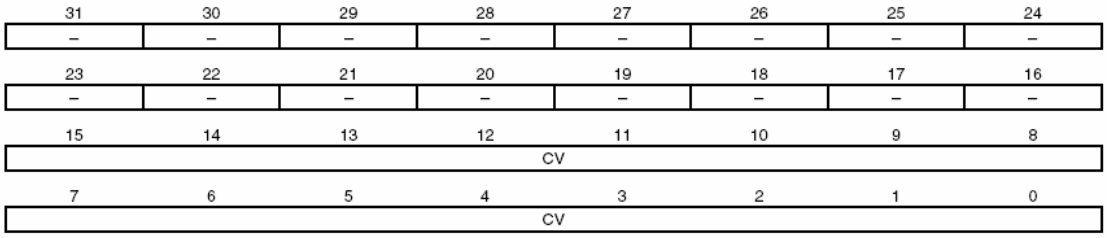
- BSWTRG：TIOB 软件触发效果

BSWTRG		效果
0	0	无
0	1	置位
1	0	清零
1	1	切换

33.6.6 TC 计数器值寄存器

寄存器名称：TC_CV

访问类型：只读



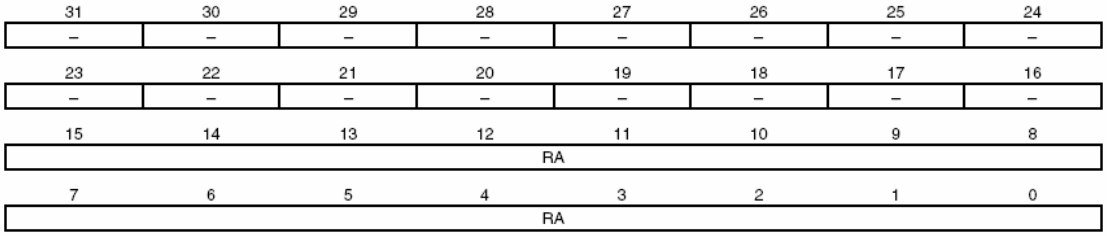
- CV：计数器值

CV 包括实时计数器值

33.6.7 TC 寄存器 A

寄存器名称：TC_RA

访问类型：若 WAVE=0 只读 ， 若 WAVE=1 读/写



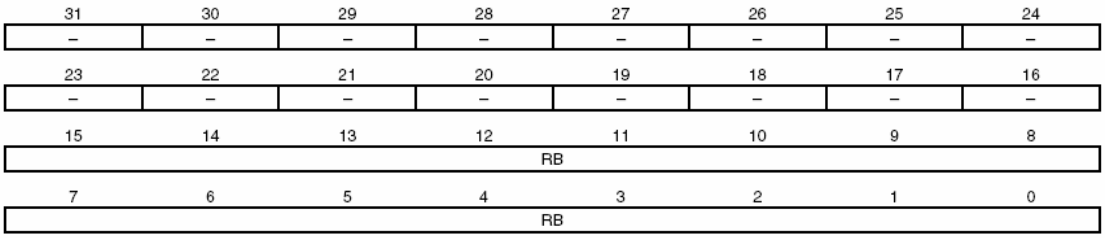
- RA：寄存器 A

RA 包括实时寄存器 A 值

33.6.8 TC 寄存器 B

寄存器名称：TC_RB

访问类型：若 WAVE=0 只读 ， 若 WAVE=1 读/写



- RB：寄存器 B

RB 包括实时寄存器 B 值

33.6.9 TC 寄存器 C

寄存器名称：TC_RC

访问类型：读/写

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

- RC：寄存器 C

RC 包括实时寄存器 C 值

33.6.10 TC 状态寄存器

寄存器名称：TC_SR

访问类型：只读

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	MTIOB	MTIOA	CLKSTA
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
ETRGS	LDRBS	LDRAS	CPCS	CPBS	CPAS	LOVRS	COVFS

- COVFS：计数器溢出状态

0=自上次读取状态寄存器未发生计数器溢出

1=自上次读取状态寄存器至少发生一次计数器溢出

- LOVRS：载入溢出状态

0=自上次读取状态寄存器或 WAVE=1 未发生载入溢出

1=若 WAVE=0，自上次读取状态寄存器在未读取任何相应寄存器的情况下至少装载 RA 或 RB 两次。

- CPAS：RA 比较状态

0=自上次读取状态寄存器或 WAVE=0 未发生 RA 比较

1=若 WAVE=1，自上次读取状态寄存器发生 RA 比较

- CPBS：RB 比较状态

0=自上次读取状态寄存器或 WAVE=0 未发生 RB 比较

1=若 WAVE=1，自上次读取状态寄存器发生 RB 比较

- CPCS：RC 比较状态

0=自上次读取状态寄存器未发生 RC 比较

1=自上次读取状态寄存器发生 RC 比较

- LDRAS：RA 装载状态

0=自上次读取状态寄存器或 WAVE=1 未发生 RA 装载

1=若 WAVE=0，自上次读取状态寄存器发生 RA 装载

- LDRBS：RB 装载状态

0=自上次读取状态寄存器或 WAVE=1 未发生 RB 装载

1=若 WAVE=0，自上次读取状态寄存器发生 RB 装载

- ETRGS：外部触发状态

0=自上次读取状态寄存器未发生外部触发

1=自上次读取状态寄存器发生外部触发

- CLKSTA：时钟使能状态

0=时钟禁用

1=时钟使能

- MTIOA：TIOA 镜像

0=TIOA 为低。若 WAVE = 0, 就是 TIOA 引脚为低。若 WAVE = 1, 就是将 TIOA 拉低

1=TIOA 为高。若 WAVE = 0, 就是 TIOA 引脚为高。若 WAVE = 1, 就是将 TIOA 拉高

- MTIOB：TIOB 镜像

0=TIOB 为低。若 WAVE = 0, 就是 TIOB 引脚为低。若 WAVE = 1, 就是将 TIOB 拉低

1=TIOB 为高。若 WAVE = 0, 就是 TIOB 引脚为高。若 WAVE = 1, 就是将 TIOB 拉高

33.6.11 TC 中断使能寄存器

寄存器名称：TC_IER

访问类型：只写

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
ETRGS	LDRBS	LDRAS	CPCS	CPBS	CPAS	LOVRS	COVFS

- COVFS：计数器溢出

0=无效

1=使能计数器溢出中断

- LOVRS：载入溢出

0=无效

1=使能载入溢出中断

- CPAS：RA 比较

0=无效

1=使能 RA 比较中断

- CPBS：RB 比较

0=无效

1=使能 RB 比较中断

- CPCS：RC 比较

0=无效

1=使能 RC 比较中断

- LDRAS：RA 装载

0=无效

1=使能 RA 装载中断

- LDRBS：RB 装载

0=无效

1=使能 RB 装载中断

- ETRGS：外部触发

0=无效

1=使能外部触发中断

33.6.12　TC 中断禁用寄存器

寄存器名称：TC_IDR

访问类型：只写

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
ETRGS	LDRBS	LDRAS	CPCS	CPBS	CPAS	LOVRS	COVFS

- COVFS：计数器溢出

0=无效

1=禁用计数器溢出中断

- LOVRS：载入溢出

0=无效

1=禁用载入溢出中断（若 WAVE = 0）

- CPAS：RA 比较

0=无效

1=禁用 RA 比较中断（若 WAVE = 1）

- CPBS：RB 比较

0=无效

1=禁用 RB 比较中断（若 WAVE = 1）

- CPCS：RC 比较

0=无效

1=禁用 RC 比较中断

- LDRAS：RA 装载

0=无效

1=禁用 RA 装载中断

- LDRBS：RB 装载

0=无效

1=禁用 RB 装载中断

- ETRGS：外部触发

0=无效

1=禁用外部触发中断

33.6.13　TC 中断屏蔽寄存器

寄存器名称：TC_IMR

访问类型：只读

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
ETRGS	LDRBS	LDRAS	CPCS	CPBS	CPAS	LOVRS	COVFS

- COVFS：计数器溢出

0=计数器溢出中断禁用

1=计数器溢出中断使能

- LOVRS：载入溢出

0=载入溢出中断禁用

1=载入溢出中断使能

- CPAS：RA 比较

0=RA 比较中断禁用

1=RA 比较中断使能

- CPBS：RB 比较

0=RB 比较中断禁用

1=RB 比较中断使能

- CPCS：RC 比较

0=RC 比较中断禁用

1=RC 比较中断使能

- LDRAS：RA 装载

0=RA 装载中断禁用

1=RA 装载中断使能

- LDRBS：装载 RB

0=装载 RB 中断禁用

1=装载 RB 中断使能

- ETRGS：外部触发

0=外部触发中断禁用

1=外部触发中断使能

34 多媒体卡接口（MCI）

34.1 描述

多媒体卡接口(MCI)支持多媒体卡(MMC)规范 V2.2 和 SD 存储卡规范 V1.0。

MCI 包括一个命令寄存器，响应寄存器，数据寄存器，暂停计数器和错误检测逻辑，当需要时能自动处理命令发送，接收响应和有处理器开销限制的数据

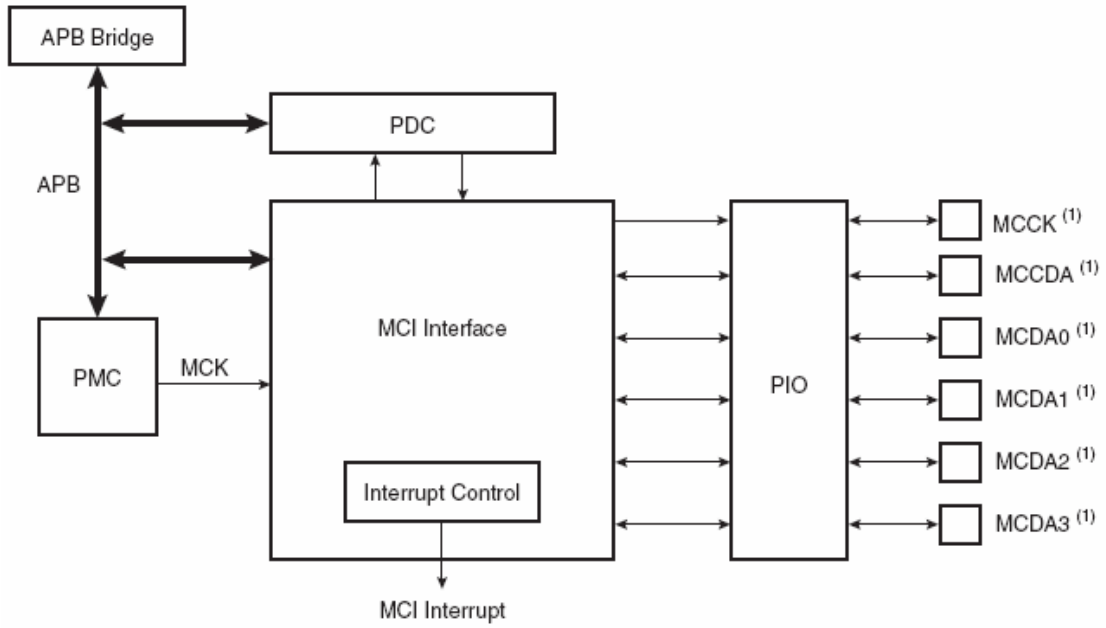
MCI 的最高工作频率为主控时钟 2 分频并支持 1 插槽连接。每个插槽可用于连接多媒体卡总线(多达 30 个卡)或 SD 存储卡。每次仅可选择一个插槽(插槽为复用)。SD 卡寄存器中的一个位域执行此选择

SD 存储卡通信是基于一个 9 针接口（时钟，命令，四个数据和三个电源线），多媒体卡基于一个 7 针接口（时钟，命令，一个数据，三个电源线和一个保留线日后使用）

SD 存储卡接口还支持多媒体卡操作。SD 和多媒体卡之间的主要区别是初始化过程和总线拓扑结构

34.2 框图

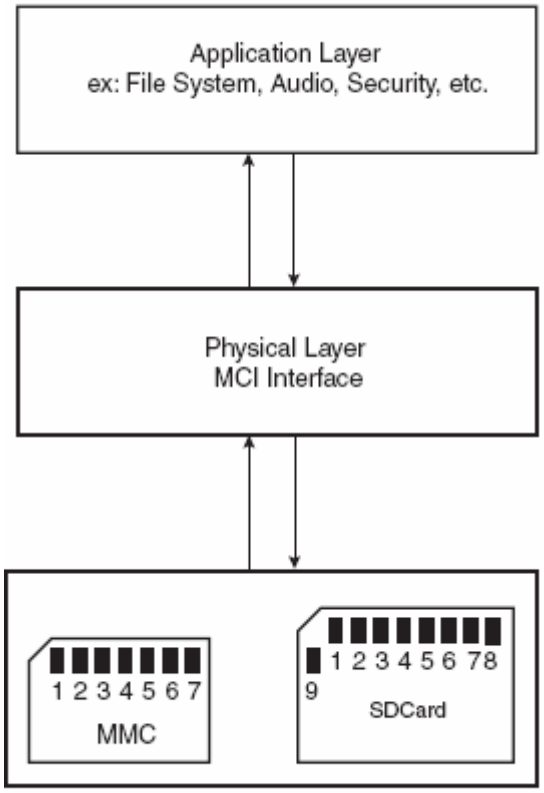
图 34-1 框图



注意：当在产品中嵌入若干 MCI（x MCI），MCCK 参考 MCIx_CK，MCCDA 参考 MCIx_CDA,MCDAy 参考 MCIx_DAy.

34.3 应用框图

图 34-2 应用框图



34.4　　引脚名称列表

表 34-1　I/O 口线描述

引脚名称（2）	引脚描述	类型（1）	备注
MCCDA	命令/响应	I/O/PP/OD	MMC 或 SD 卡的 CMD
MCCK	时钟	I/O	MMC 或 SD 卡的 CLK
MCDA0 - MCDA3	插槽 A 数据 0...3	I/O/PP	MMC 的 DAT0 SD 卡的 DAT[0...3]

注意：1.I：输入，O：输出，PP：推/拉,OD：开漏

2.当在产品中嵌入若干 MCI（x MCI），MCCK 参考 MCIx_CK，MCCDA 参考 MCIx_CDA，MCDAy 参考 MCIx_DAy.

34.5　　附属产品

34.5.1　I/O 口线

用于连接多媒体卡或 SD 卡的引脚可以和 PIO 口线复用。编程者必须首先编程 PIO 控制器来为 MCI 引脚指定外设功能

34.5.2　电源管理

MCI 可通过电源管理控制器（PMC）协调同步，所以编程者必须首先配置 PMC 来使能 MCI 时钟。

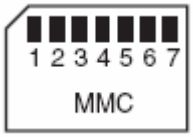
34.5.3　中断

MCI 接口有一个连接于高级中断控制器（AIC）的中断口线

处理 MCI 中断需要在配置 MCI 前编程 AIC

34.6　　总线拓扑结构

图 34-3　多媒体存储卡总线结构



多媒体卡通信基于一个 7 针串行总新接口。有三个通信口线和四个电源口线。

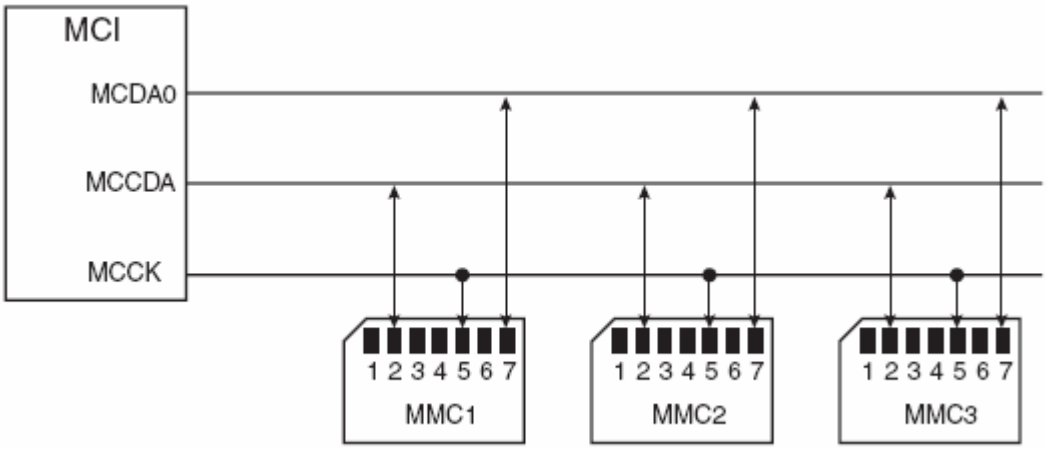
表 34-2 总线结构

引脚编号	名称	类型（1）	描述	MCI 引脚名称（2）（Slot z）
1	RSV	NC	未连接	-
2	CMD	I/O/PP/OD	命令/响应	MCCDz
3	VSS1	S	电源地	VSS
4	VDD	S	电源电压	VDD
5	CLK	I/O	时钟	MCCK
6	VSS2	S	电源地	VSS
7	DST[0]	I/O/PP	数据 0	MCDz0

注意：1。I：输入，O：输出，PP：推/拉,OD：开漏

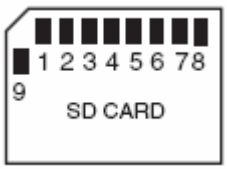
2.当在产品中嵌入若干 MCI（x MCI），MCCK 参考 MCIx_CK，MCCDA 参考 MCIx_CDA，MCDAy 参考 MCIx_DAy.

图 34-4 MMC 总线连接（一个插槽）



注意：当在产品中嵌入若干 MCI（x MCI），MCCK 参考 MCIx_CK，MCCDA 参考 MCIx_CDA，MCDAy 参考 MCIx_DAy.

图 34-5 SD 存储卡总线结构



SD 存储卡总线包括表 34-3 中所列信号

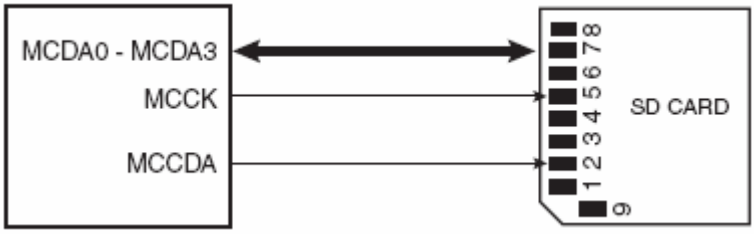
表 34-3　SD 存储卡总线信号

引脚编号	名称	类型（1）	描述	MCI 引脚名称（2）（Slot z）
1	CD/DAT[3]	I/O/PP	卡检测/数据口线位 3	MCKz3
2	CMD	PP	命令/响应	MCCDz
3	VSS1	S	电源地	VSS
4	VDD	S	电源电压	VDD
5	CLK	I/O	时钟	MCCK
6	VSS2	S	电源地	VSS
7	DAT[0]	I/O/PP	数据口线位 0	MCDz0
8	DAT[1]	I/O/PP	数据口线位 1	MCDz1
9	DAT[2]	I/O/PP	数据口线位 2	MCDz2

注意：1。I：输入，O：输出，PP：推/拉,OD：开漏

2.当在产品中嵌入若干 MCI（x MCI），MCCK 参考 MCIx_CK，MCCDA 参考 MCIx_CDA，MCDAy 参考 MCIx_DAy.

图 34-6　用一个插槽连接 SD 卡总线



注意：当在产品中嵌入若干 MCI（x MCI），MCCK 参考 MCIx_CK，MCCDA 参考 MCIx_CDA，MCDAy 参考

当配置 MCI 为 SD 存储卡操作时，可在 MCI_SDCR 寄存器中选择数据总线宽度。清零此寄存器的 SDCBUS 位表示一位宽度；置位表示四位宽度。对多媒体卡来说，仅使用数据口线 0.其它数据口线可用作独立的 PIO

34.7　多媒体卡操作

上电复位后，一个专用的基于多媒体卡总线协议的信息将卡初始化。每个信息由以下任一记号表示：

- 命令：命令用来启动操作。从主机发送命令到单卡（寻址命令）或到所有已连接的卡（广播命令）。命令在 CMD 口线上串行发送。
- 响应：响应是从定址卡或（同步的）从所有已连接的卡到主机作为对前一个已接收命令做出回答。响应在 CMD 口线上串行传输。
- 数据：数据在卡和主机之间传输。数据由数据口线传输。

在初始化阶段由总线控制器对当前连接的卡进行地址分配，实现卡定址。每个卡都有其唯一的 CID 编号。

在多媒体卡系统规范中有对命令，响应和数据部件结构的描述。见 525 页的表 34-4。

多媒体卡总线传输由这些内容组成。

多媒体卡有不同类型的操作。定址操作通常包括命令和响应记号。另外，有些操作包含数据记号；其它的操作在命令或响应结构内直接传输信息。此时，操作中不出现数据记号。DAT 和 CMD 口线上的位以 MCI 时钟同步传输。

定义两种类型的数据传输命令：

- 序列命令：这些命令初始化一个连续的数据流。它们仅当 CMD 口线上跟随一个停止命令时终止。此模式将命令开销减少到一个极小范围。
- 块定向命令：这些命令由 CRC 位连续的发送数据块。

读/写操作都允许单个或多块发送。类似于序列读，当 CMD 口线上跟随一个停止命令时多块发送终止

MCI 提供一套寄存器来执行所有的多媒体卡操作。

34.7.1 命令-响应操作

复位后，MCI 禁用并且只有在置位 MCI_CR 控制寄存器中的 MCIEN 位后生效。

当总线待用时 PWSEN 位通过$2^{PWSDIV} + 1$分频 MCI 时钟来降低功耗。

卡的命令和响应由 MCI 时钟的上升沿协调同步。

多媒体卡的所有时序都定义在多媒体卡系统规范中。

需要两种总线模式（开漏和推/拉）来处理所有定义在 MCI 命令寄存器中的操作。MCI_CMDR 允许执行命令。

例如，执行 ALL_SEND_CID 命令：

CMD

主机命令					Nid 周期					CID			
S	T	内容	CRC	E	Z	******	Z	S	T	内容	Z	Z	Z

ALL_SEND_CID 命令和 MCI_CMDR 控制寄存器的域和值见表 34-4 和表 34-5 描述。

表 34-4 ALL_SEND_CID 命令描述

CMD 索引	类型	变量	Resp	缩写	命令描述
CMD2	Bcr	[31：0]填充位	R2	ALL_SEND_CID	要求所有卡在 CMD 口线上发送其 CID 编号

注意：bcr 就是带响应的广播命令

表 34-5 MCI_CMDR 命令寄存器的域和值

域	值
CMDNB（命令编号）	2（CMD2）
RSPTYP（响应类型）	2（R2：136 位响应）
SPCMD（专用命令）	0（非专用命令）
OPCMD（开漏命令）	1
MAXLAT（命令响应的最大延迟时间）	0（NID 周期==》5 周期）
TRCMD（传输命令）	0（不传输）
TRDIR（传输方向）	X(仅在传输命令可用)
TRTYP（传输类型）	X(仅在传输命令可用)

MCI_ARGR 包含命令的变量域。

要发送命令，用户必须执行以下步骤：

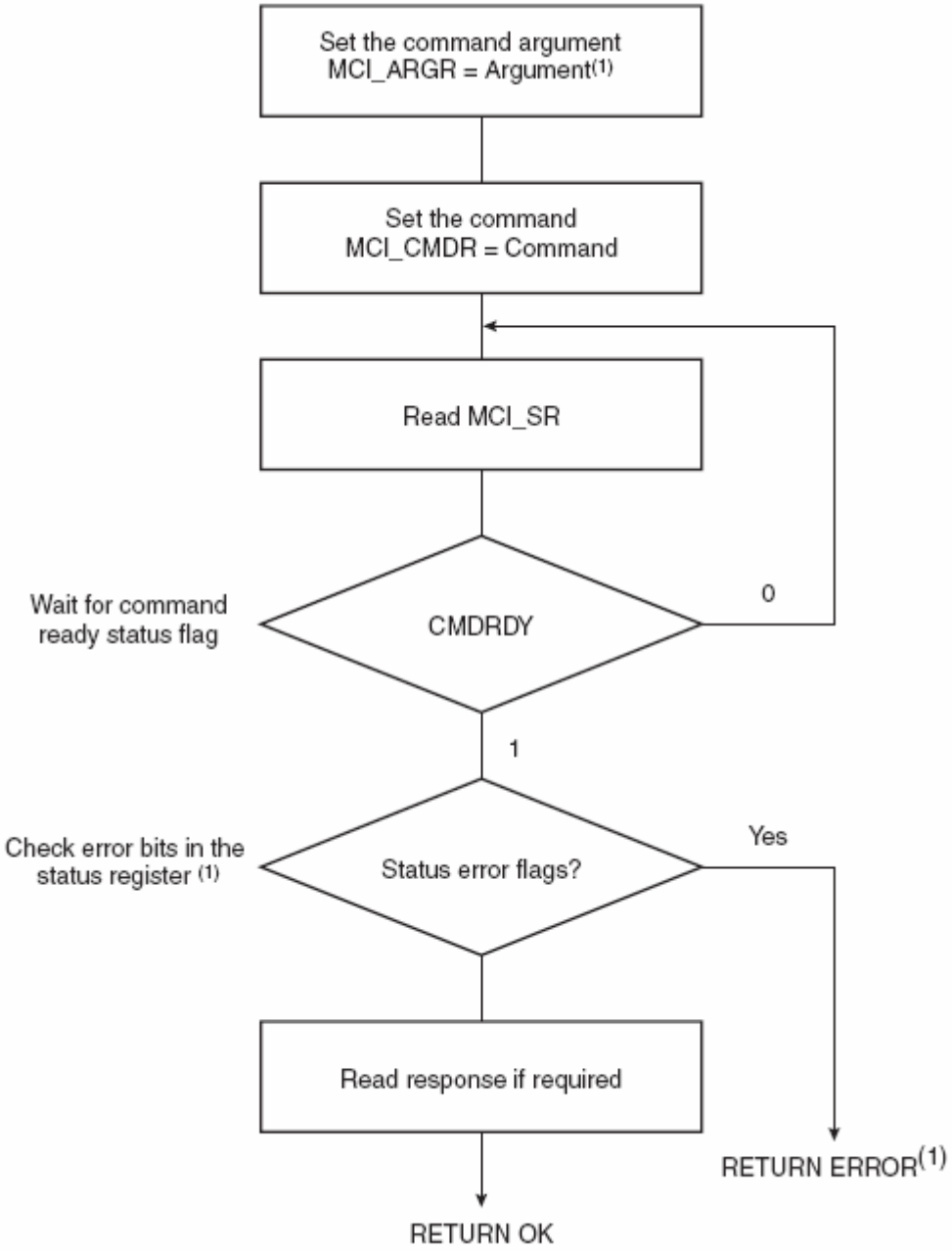
- 将命令变量填入变量寄存器(MCI_ARGR)
- 设置命令寄存器(MCI_CMDR)（见表 34-5）

写命令寄存器后立即发送该命令。当命令完成时激活状态寄存器(MCI_SR)中的状态位 CMDRDY。若此命令需要响应，可在 MCI 响应寄存器(MCI_RSPR)中读取到。根据命令响应大小从 48 位到 136 位。MCI 内置了一个错误检测来防止传

输期间的数据破坏。

下面的流程图展示了怎样发送命令到卡及在需要时读取响应。该例中，对状态寄存器位进行轮询，但是要设置中断使能寄存器中的相应位以允许使用中断方式。

图 34-7　命令/响应功能流程图



注意：1.若为 SEND_OP_COND 命令，则 CRC 错误标志位总会出现（参考多媒体卡规范中的 R3 响应）

34.7.2　数据传输操作

多媒体卡允许多种读/写操作（单块，多块，流等）。通过设置 MCI 命令寄存器(MCI_CMDR)中的传输类型(TRTYP)域来选择这些传输。

可用外围 DMA 控制器（PDC）功能完成这些操作。若 MCI_MR 中的 PDCMODE 位置位，则接下来所有的读/写都使用 PDC 设备

所有情况下，块长度必须在模式寄存器 MCI_MR 中定义。

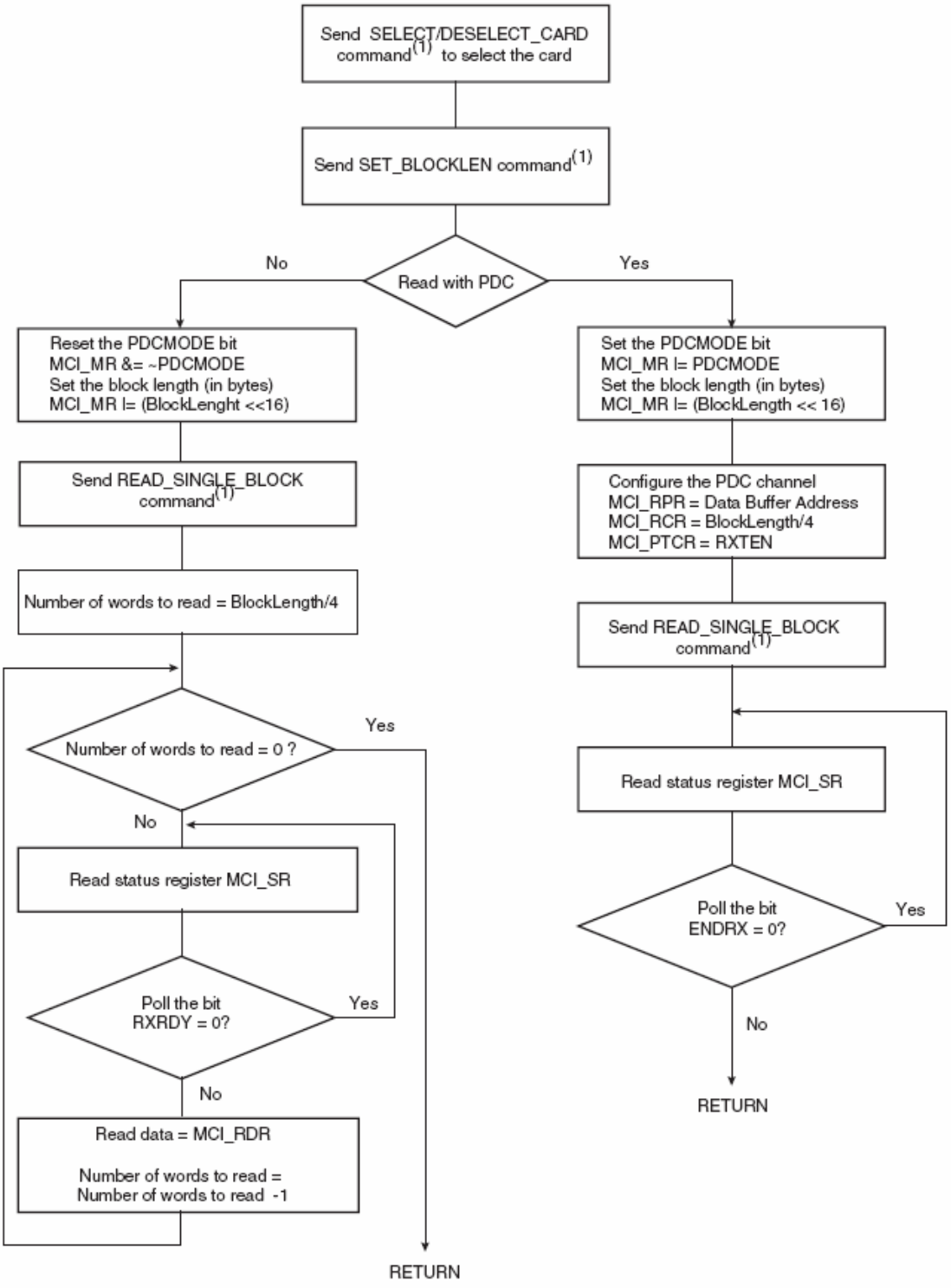
此域决定数据块的大小

34.7.3 读操作

下面的流程图展示了使用或不用 PDC 设备时怎样读取单时钟。此例中（见图 34-8），使用轮询发等待读取结束。

类似的，用户可配置中断使能寄存器(MCI_IER)在读取接收时触发一个中断

图 34-8 读功能流程图



注意：1.假设此命令已正确发送（见图 34-7）

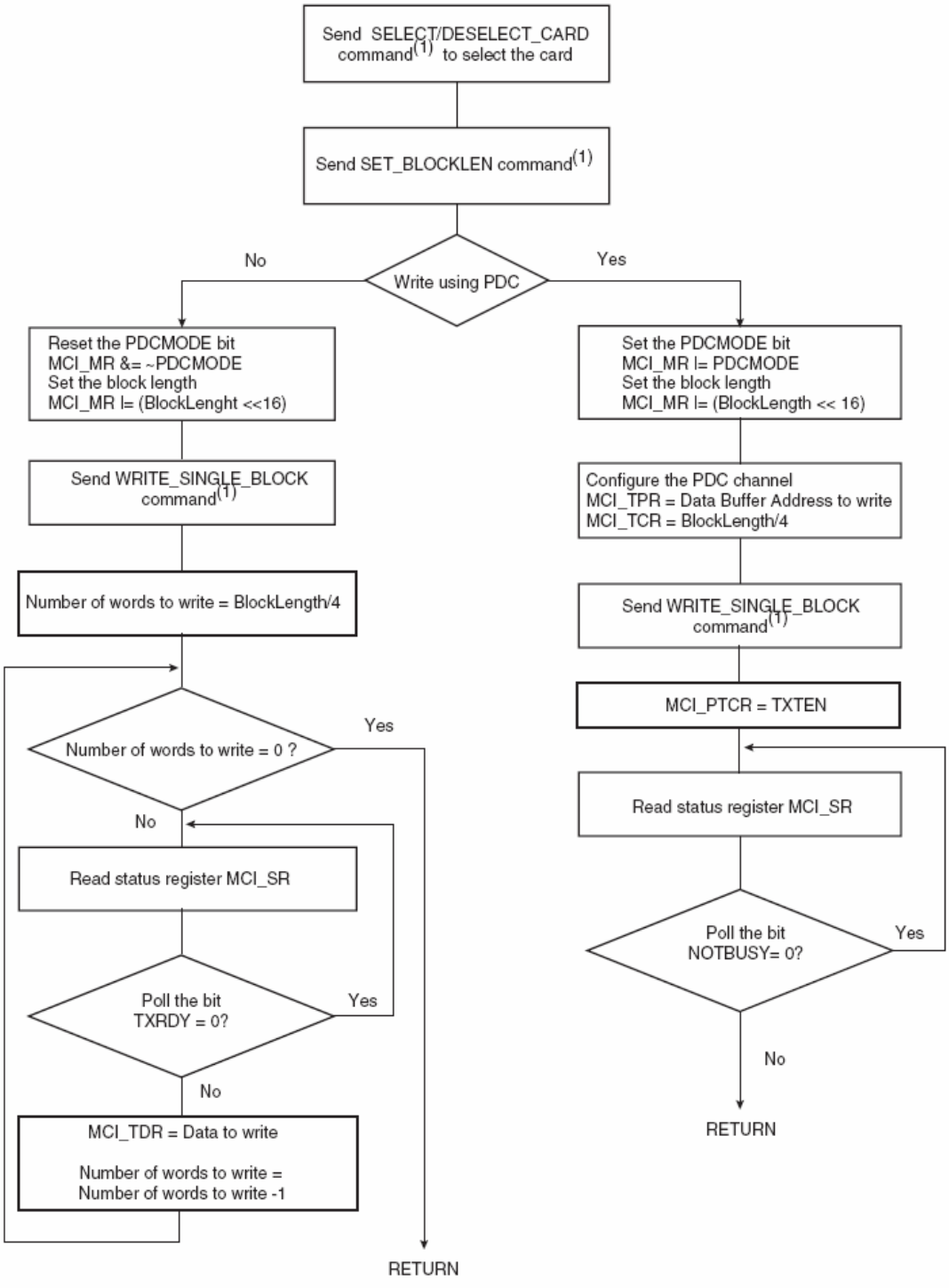
34.7.4　写操作

写操作中，MCI 模式寄存器(MCI_MR)用来定义写非多块大小时的填充值。若 PDCPADV 位为 0，则当填充数据时使用 0x00 值，否则使用 0xFF

若置位，PDCMODE 位使能 PDC 传输。

下面的流程图展示了使用或不用 PDC 工具时怎样写一个单块(见图 37-9)。可使用轮询或中断方法根据中断屏蔽寄存器(MCI_IMR)中的内容等待写结束

图 34-9　写功能流程图

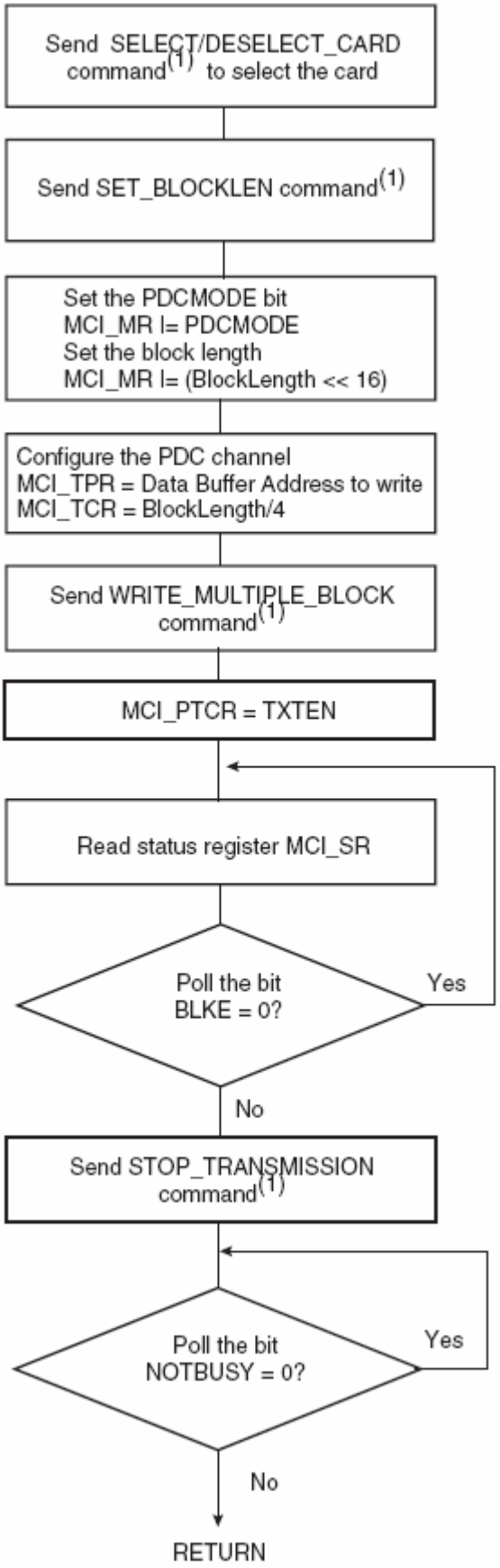


注意：1.假设此命令已正确发送（见图 34-7）

下面的流程图展示了使用 PDC 时怎样管理一个多写块传输(见图 34-10)。

根据中断屏蔽寄存器(MCI_的内容可使用轮询或中断方法等待写结束

图 34-10 多写功能流程图



注意：1。假设此命令已正确发送（见图 34-7）

34.8 SD 操作

多媒体卡接口能执行 SD 存储（安全数字存储卡）卡命令。

SD 卡基于多媒体卡（MMC）格式，但尺寸稍微大一些并且数据传输速率性能高一些，边上有一个锁定开关来防止意外改写和安全特性。其物理形状因素，引脚分配和数据传输协议附加一些条件就与多媒体卡前向兼容。SD 受众多专利和商标保护，只有通过安全数字卡协会许可才可使用。

SD 卡通信基于 9 针接口（时钟，命令，4×数据和 3×电源线）。规范中定义了通信协议。SD 卡和多媒体卡之间的主要差别是在初始化过程。

SD 卡总线允许动态配置数据口线数量。上电后，对数据传输 SD 卡默认仅使用 DAT0 作为数据传输。初始化后，主机可以改变总线宽度（有效数据口线数）。

34.9 多媒体卡接口（MCI）用户接口

表 34-6　寄存器映射

偏移量	寄存器	寄存器名称	读/写	复位值
0x00	控制寄存器	MCI_CR	写	-
0x04	模式寄存器	MCI_MR	读/写	0x0
0x08	数据超时寄存器	MCI_DTOR	读/写	0x0
0x0C	SD 卡寄存器	MCI_SDCR	读/写	0x0
0x10	变量寄存器	MCI_ARGR	读/写	0x0
0x14	命令寄存器	MCI_CMDR	写	-
0x18-0x1C	保留	-	-	-
0x20	响应寄存器（1）	MCI_RSPR	读	0x0
0x24	响应寄存器（1）	MCI_RSPR	读	0x0
0x28	响应寄存器（1）	MCI_RSPR	读	0x0
0x2C	响应寄存器（1）	MCI_RSPR	读	0x0
0x30	接收数据寄存器	MCI_RDR	读	0x0
0x34	发送数据寄存器	MCI_TDR	写	-
0x38-0x3C	保留	-	-	-
0x40	状态寄存器	MCI_SR	读	0xC0E5
0x44	中断使能寄存器	MCI_IER	写	-
0x48	中断禁用寄存器	MCI_IDR	写	-
0x4C	中断屏蔽寄存器	MCI_IMR	读	0x0
0x50-0xFC	保留	-	-	-
0x100-0x124	保留给 PDC	-	-	-

注意：1.响应寄存器可由 MCI_RSPR 读 N 次或在连续地址（0x20 到 0x2C），N 由响应大小决定

34.9.1 MCI 控制寄存器

名称：MCI_CR

访问类型：只写

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

- MCIEN：多媒体接口使能

0=无效

1=若 MCDIS 为 0，使能多媒体接口

- MCIDIS：多媒体接口禁用

0=无效

1=禁用多媒体接口

- PWSEN：节能模式使能

0=无效

1=若 PWSDIS 为 0，使能节能模式

警告：使能此模式前，用户必须在 PWSDIV 域（模式寄存器 MCI_MR）设置一个非零值

- PWSDIS：节能模式禁用

0=无效

1=禁用节能模式

- SWRST：软件复位

0=无效

1=复位 MCI。执行软件硬件复位 MCI 接口

34.9.2 MCI 模式寄存器

名称：MCI_MR

访问类型：读/写

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

- CLKDIV：时钟分频器

多媒体卡接口时钟(MCCK 或 MCI_CK)为(2*(CLKDIV+1))分频的主控时钟（MCK)。

- PWSDIV：节能分频器

当进入节能模式时，$2^{(PWSDIV)} + 1$分频多媒体卡接口时钟

警告：在使能 MCI_CR (MCI_PWSEN 位)中的节能模式前该值必须为非零。

- PDCPADV：PDC 填充值

0=当填充数据到写传输时使用 0x00 值（不仅 PDC 传输）

1=当填充数据到写传输时使用 0xFF 值（不仅 PDC 传输）

- PDCMODE：PDC 定向模式

0=禁用 PDC 传输

1=使能 PDC 传输。此情况下，在 PDC 传输已完成后重新激活 MCI 模式寄存器(MCI_SR)中的 UNRE 和 OVRE 标志位

- BLKLEN：数据块长度

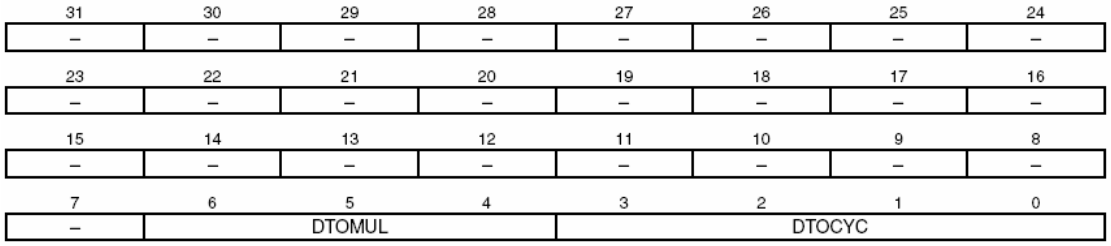
此域决定数据块的大小

位 16 和位 17 必须置为 0

34.9.3 MCI 数据超时寄存器

名称：MCI_DTOR

访问类型：读/写



- DTOCYC：数据超时周期数
- DTOMUL：数据超时乘数

此域决定 MCI 在两个数据块传输之间等待的最大主控周期数

它等于(DTOCYC ×乘数).

乘数如下表所示由 DTOMUL 定义

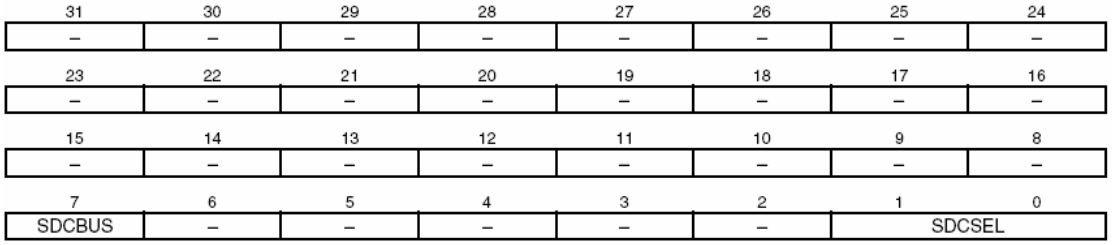
DTOMUL			乘数
0	0	0	1
0	0	1	16
0	1	0	128
0	1	1	256
1	0	0	1024
1	0	1	4096
1	1	0	65536
1	1	1	1048576

若已超越由 DTOCYC 和 DTOMUL 设置的数据时限，则 MCI 状态寄存器(MCI_SR)中的数据超时错误标志位(DTOE)升高

34.9.4 MCI SD 卡寄存器

名称：MCI_SDCR

访问类型：读/写



- SDCSEL：SD 卡选择

SDCSEL		SD 卡插槽
0	0	选择插槽 A
0	1	保留
1	0	保留
1	1	保留

- SDCBUS：SD 卡总线宽度

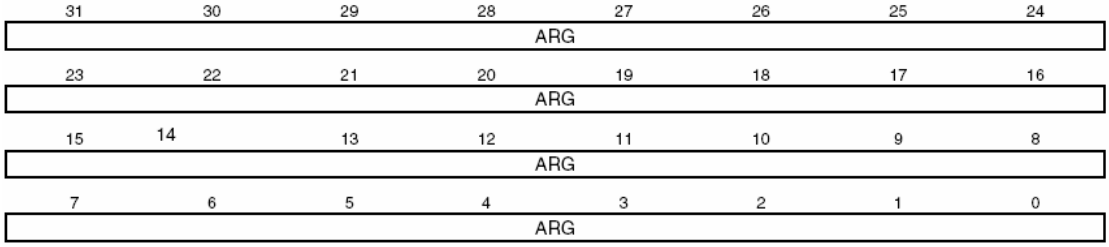
0=1 位数据总线

1=4 位数据总线

34.9.5　MCI 变量寄存器

名称：MCI_ARGR

访问类型：读/写

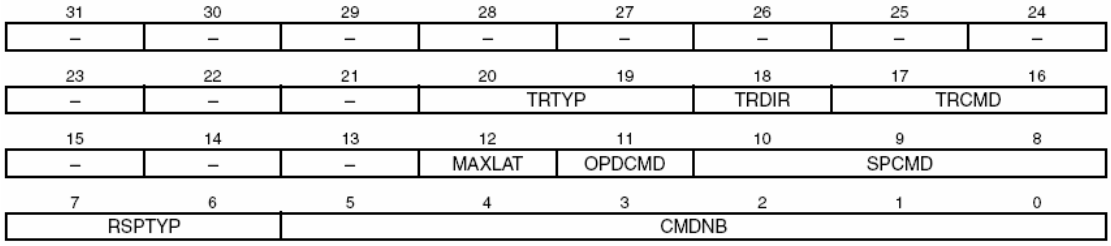


- ARG：命令变量

34.9.6　MCI 命令寄存器

名称：MCI_CMDR

访问类型：只写



当 MCI_SR 中 CMDRDY 为 0 时此寄存器写保护。若发送一个中断命令，此寄存器仅可被中断响应（SPCMD 域）写。这就是说当前命令不能被中断或修改。

- CMDNB：命令号
- RSPTYP：响应类型

RSP		响应类型
0	0	无响应
0	1	48 位响应
1	0	136 位响应
1	1	保留

- SPCMD：专用命令

SPCMD			命令
0	0	0	无专用 CMD
0	0	1	初始化 CMD： 初始化序列须 74 个时钟周期
0	1	0	同步 CMD： 发送挂起命令前等待当前数据块传输结束
0	1	1	保留
1	0	0	中断命令： 对应中断模式（CMD40）
1	0	1	中断响应： 对应中断模式（CMD40）

- OPDCMD：开漏命令

0=推拉命令

1=开漏命令

- MAXLAT：命令到响应最大等待时间

0=5 周期最大等待

1=64 周期最大等待

- TRCMD：传输命令

TRCMD		传输类型
0	0	无数据传输
0	1	启动数据传输
1	0	停止数据传输

1	1	保留

- TRDIR：传输检测

0=写

1=读

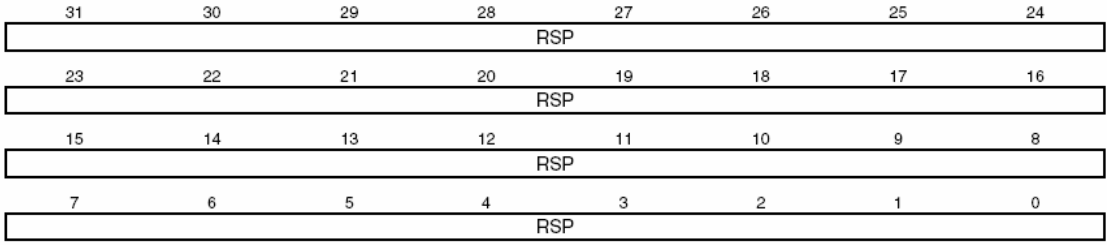
- TRTYP：传输类型

TRTYP			传输类型
0	0	0	MMC/SD 卡单块
0	0	1	MMC/SD 卡多块
0	1	0	MMC 流
0	0	1	保留

34.9.7 MCI 响应寄存器

名称：MCI_RSPR

访问类型：只读



- RSP：响应

注意：1.可通过 N 次访问在相同的 MCI_RSPR 或在连续地址(0x20 到 0x2C)上读取响应寄存器。
N 取决于响应的大小

34.9.8 MCI 接收数据寄存器

名称：MCI_RDR

访问类型：只读

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

- DATA：读取数据

34.9.9 MCI 发送数据寄存器

名称：MCI_TDR

访问类型：只写

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

- DATA：写数据

34.9.10 MCI 状态寄存器

名称：MCI_SR

访问类型：只读

31	30	29	28	27	26	25	24
UNRE	OVRE	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	DTOE	DCRCE	RTOE	RENDE	RCRCE	RDIRE	RINDE
15	14	13	12	11	10	9	8
TXBUFE	RXBUFF	–	–	–	–	–	–
7	6	5	4	3	2	1	0
ENDTX	ENDRX	NOTBUSY	DTIP	BLKE	TXRDY	RXRDY	CMDRDY

- CMDRDY：命令就绪

0=命令正在处理

1=上一个命令已发送。当写 MCI_CMDR 时清零

- RXRDY：接收器就绪

0=自上次读取 MCI_RDR 还未接收到数据

1=自上次读取 MCI_RDR 还已接收到数据

- TXRDY：发送就绪

0=上次写入 MCI_TDR 的数据还未传输到移位寄存器

1=上次写入 MCI_TDR 的数据已传输到移位寄存器

- BLKE：数据块结束

此标志必须仅对写操作使用

0=还未完成数据块传输。当读取 MCI_SR 时清零

1=数据块传输结束，包括 CRC16 状态发送

在 PDC 模式(PDCMODE=1)，当上一数据块的 CRC 状态已发送（已置位 TXBUFE）则标志位置位否则(PDCMODE=0)，对每一个发送 CRC 状态标志位置位。

对更多关于 CRC 状态的细节参考 MMC 或 SD 规范

- DTIP：数据传输正在进行

0=无传输数据正在进行

1=当前数据传输正在进行，包括 CRC16 计算。在 CRC16 结束时清零

NOTBUSY：MCI 不忙

此标志必须仅对写操作使用

数据块写操作使用数据口线(DAT0)上写操作持续的采样忙信号：在一个数据块传输期间，若卡没有空闲的数据接收缓冲区，则该卡通过轮询数据口线(DAT0)的 LOW 信号指示此状态。只要此卡至少有一个对已定义数据传输块长度的接收缓冲区变为空闲则该卡就停止轮询数据口线。

NOTBUSY 标志允许处理这些不同状态

0=对新的数据传输 MCI 未就绪。在卡响应结束时清零

1=对新的数据传输 MCI 未就绪。当数据口线上的忙状态结束时置位。这对应于卡的空闲间断数据接收缓冲

对更多关于忙动作的细节参考 MMC 或 SD 规范

- ENDRX：RX 缓冲结束

0=自上次写 MCI_RCR 或 MCI_RNCR 接收计数器寄存器未达到 0

1=自上次写 MCI_RCR 或 MCI_RNCR 接收计数器寄存器已达到 0

- ENDTX：TC 缓冲结束

0=自上次写 MCI_RCR 或 MCI_RNCR 发送计数器寄存器未达到 0

1=自上次写 MCI_RCR 或 MCI_RNCR 发送计数器寄存器已达到 0

注意：可使用 BLKE 和 NOTBUSY 标志来检查数据口线上数据已成功发送或不仅仅从 PDC 传输到 MCI 控制器

- RXBUFF：RX 缓冲区满

0=MCI_RCR 或 MCI_RNCR 为非零值

1=MCI_RCR 和 MCI_RNCR 都为零值

注意：可使用 BLKE 和 NOTBUSY 标志来检查数据口线上数据已成功发送或不仅仅从 PDC 传输到 MCI 控制器

- RINDE:响应索引错误

0=无错误

1=在命令索引发送和响应索引接收之间检测到误匹配。当写 MCI_CMDR 时清零

- RDIRE：响应方向错误

0=无误

1=在响应时还未检测到从卡到主机的方向位

- RCRCE：响应 CRC 错误

0=无误

1=在响应时检测到一个 CRC7 错误。当写 MCI_CMDR 时清零

- RENDE：响应结束位错误

0=无误
1=还未检测到响应结束位。当写 MCI_CMDR 时清零

- RTOE：响应超时错误

0=无误
1=通过 MCI_CMDR 中的 MAXLAT 已超越时限来置位响应超时。当写 MCI_CMDR 时清零

- DCRCE：数据 CRC 错误

0=无误
1=在上一数据块中检测到 CRC16 错误。通过读 MCI_SR 寄存器复位

- DTOE：数据超时错误

0=无误
1=通过 MCI_DTOR 中的 DTOCYC 和 DTOMUL 已超越时限来置位数据超时。通过读 MCI_SR 寄存器复位

- OVRE：超限

0=无误
1=至少已丢失一个 8 位已接收数据（未读）。当发送一个新数据传输命令时清零

- UNRE：欠载运行

0=无误
1=至少一个未带有效信息（未写）的 8 位数据已发送。当发送一个新数据传输命令时清零

- RXBUFF：RX 缓冲区满

0=MCI_RCR 或 MCI_RNCR 为非零值
1=MCI_RCR 和 MCI_RNCR 都为零值

- TXBUFE：TX 缓冲区空

0=MCI_TCR 或 MCI_TNCR 为非零值
1=MCI_TCR 和 MCI_TNCR 都为零值

34.9.11 MCI 中断使能寄存器

名称：MCI_IER

访问类型：只写

31	30	29	28	27	26	25	24
UNRE	OVRE	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	DTOE	DCRCE	RTOE	RENDE	RCRCE	RDIRE	RINDE
15	14	13	12	11	10	9	8
TXBUFE	RXBUFF	–	–	–	–	–	–
7	6	5	4	3	2	1	0
ENDTX	ENDRX	NOTBUSY	DTIP	BLKE	TXRDY	RXRDY	CMDRDY

- CMDRDY：命令就绪中断使能
- RXRDY：接收器就绪中断使能
- TXRDY：发送就绪中断使能
- BLKE：数据块结束中断使能
- DTIP：数据传输正在进行中断使能
- NOTBUSY：数据不忙中断使能
- ENDRX：接收缓冲结束中断使能
- ENDTX：发送缓冲结束中断使能
- RXBUFF：接收缓冲区满中断使能
- TXBUFE：发送缓冲区空中断结束
- RINDE：响应索引错误中断使能
- RDIRE：响应方向错误中断使能
- RCRCE：响应 CRC 错误中断使能
- RENDE：响应结束位错误中断使能
- RTOE：响应超时错误中断使能
- DCRCE：数据 CRC 错误中断使能
- DTOE：数据超时错误中断使能
- OVRE：超限中断使能
- UNRE：欠载运行中断使能

0=无效

1=使能相应中断

34.9.12　MCI 中断禁用寄存器

名称：MCI_IDR

访问类型：只写

31	30	29	28	27	26	25	24
UNRE	OVRE	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	DTOE	DCRCE	RTOE	RENDE	RCRCE	RDIRE	RINDE
15	14	13	12	11	10	9	8
TXBUFE	RXBUFF	–	–	–	–	–	–
7	6	5	4	3	2	1	0
ENDTX	ENDRX	NOTBUSY	DTIP	BLKE	TXRDY	RXRDY	CMDRDY

- CMDRDY：命令就绪中断禁用
- RXRDY：接收器就绪中断禁用
- TXRDY：发送就绪中断禁用
- BLKE：数据块结束中断禁用
- DTIP：数据传输正在进行中断禁用
- NOTBUSY：数据不忙中断禁用
- ENDRX：接收缓冲结束中断禁用
- ENDTX：发送缓冲结束中断禁用
- RXBUFF：接收缓冲区满中断禁用
- TXBUFE：发送缓冲区空中断禁用
- RINDE：响应索引错误中断禁用
- RDIRE：响应方向错误中断禁用
- RCRCE：响应 CRC 错误中断禁用
- RENDE：响应结束位错误中断禁用
- RTOE：响应超时错误中断禁用
- DCRCE：数据 CRC 错误中断禁用
- DTOE：数据超时错误中断禁用
- OVRE：溢出中断禁用
- UNRE：欠载运行中断禁用

0=无效

1=禁用相应中断

34.9.13　MCI 中断屏蔽寄存器

名称：MCI_IMR

访问类型：只读

31	30	29	28	27	26	25	24
UNRE	OVRE	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	DTOE	DCRCE	RTOE	RENDE	RCRCE	RDIRE	RINDE
15	14	13	12	11	10	9	8
TXBUFE	RXBUFF	–	–	–	–	–	–
7	6	5	4	3	2	1	0
ENDTX	ENDRX	NOTBUSY	DTIP	BLKE	TXRDY	RXRDY	CMDRDY

- CMDRDY：命令就绪中断屏蔽
- RXRDY：接收器就绪中断屏蔽
- TXRDY：发送就绪中断屏蔽
- BLKE：数据块结束中断屏蔽
- DTIP：数据传输正在进行中断屏蔽
- NOTBUSY：数据不忙中断屏蔽
- ENDRX：接收缓冲结束中断屏蔽
- ENDTX：发送缓冲结束中断屏蔽
- RXBUFF：接收缓冲区满中断屏蔽
- TXBUFE：发送缓冲区空中断结束
- RINDE：响应索引错误中断屏蔽
- RDIRE：响应方向错误中断屏蔽
- RCRCE：响应 CRC 错误中断屏蔽
- RENDE：响应结束位错误中断屏蔽
- RTOE：响应超时错误中断屏蔽
- DCRCE：数据 CRC 错误中断屏蔽
- DTOE：数据超时错误中断屏蔽
- OVRE：超限中断屏蔽
- UNRE：欠载运行中断屏蔽

0=相应中断未使能

1=相应中断使能

35. USB 主机端口（UHP）

35.1 描述

USB 主机端口(UHP)在主机应用中和 USB。UHP 处理 USB v2.0 全速和低速协议还处理开放的 HCI 协议（开放主机控制器接口）

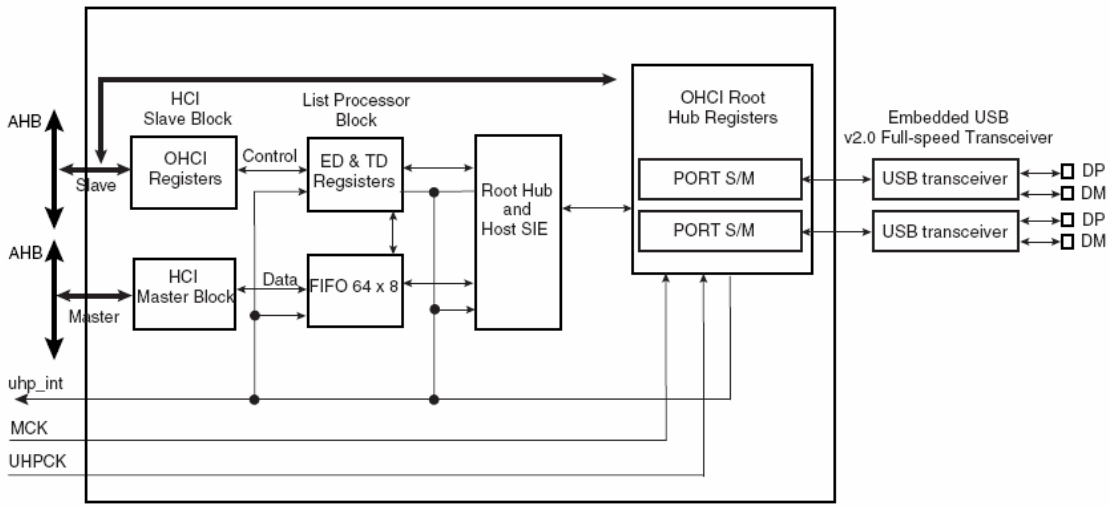
USB 主机端口集成一个根集线器，并在下游端口集成收发器。它提供若干速率为 12Mbit/s 的高速半双工高速串行通信端口。在 USB“分层星形”拓扑结构中可以连接多达 127 个 USB 设备和 USB 集线器到 USB 主机。

USB 主机端口控制器和开 HCI 规范全兼容。可以容易的以同样的方法将标准的 OHCI USB 堆栈启动器连接到 ATMEL 的架构。所有现存的类驱动器可在无硬件规范的情况下运行

这就是说标准分类驱动是自动检测并可在应用程序可用。例如，集成一个 HID（人机接口设备）类驱动器为所有 USB 键盘和鼠标提供一个即插即用功能

35.2 框图

图 35-1 框图



通过 AHB 总线从设备接口完成 USB 主机操作寄存器的访问。OHCI 主机控制器通过以下 ASB 总线主控接口初始化主控 DMA 传输：

- 取得端点描述符和传输描述符
- 从系统存储器访问端点数据
- 访问 HC 通信域
- 写状态并退出传输描述符

存储器访问错误（中止，未对准）导致一个“不可恢复的错误”，由主机控制器操作寄存器中的相应标志位指示

USB 根集线器集成在 USB 主机端口。若干 USB 下游端口可用。下游端口数由软件驱动器读取根集线器的操作寄存器来决定。由 USB 主机端口逻辑电路自动检测设备连接。

警告：必须在板上连接一个上拉电阻。否则 USB 主机将一直检测此端口的器件连接。

USB 物理收发器集成在产品中并且由根集线器端口驱动。

由 USB 主机控制器激活端口上的过电流保护。Atmel 的标准产品不给外部过电流保护提供保护垫。

35.3 附属产品

35.3.1 I/O 口线

DP 和 DM 不由任何 PIO 控制器控制。嵌入式 USB 物理收发器由 USB 主机控制器控制

35.3.2 电源管理

USB 主机控制器需要 48MHz 的时钟。必须由精度位±0.25%的 0PLL 产生该时钟。

因此 USB 器件外设从电源管理控制器（PMC）接收两个时钟：用于驱动外设用户接口的主控时钟 MCK（MCK 域）和用于连接总线 USB 信号的 UHPCLK 48 MHz 的时钟（恢复 12MHz 域）

35.3.3 中断

USB 主机接口有一个连接到高级中断控制器（AIC）的中断口线。

处理 USB 主机中断需要在配置 UHP 前编程 AIC

35.4 功能描述

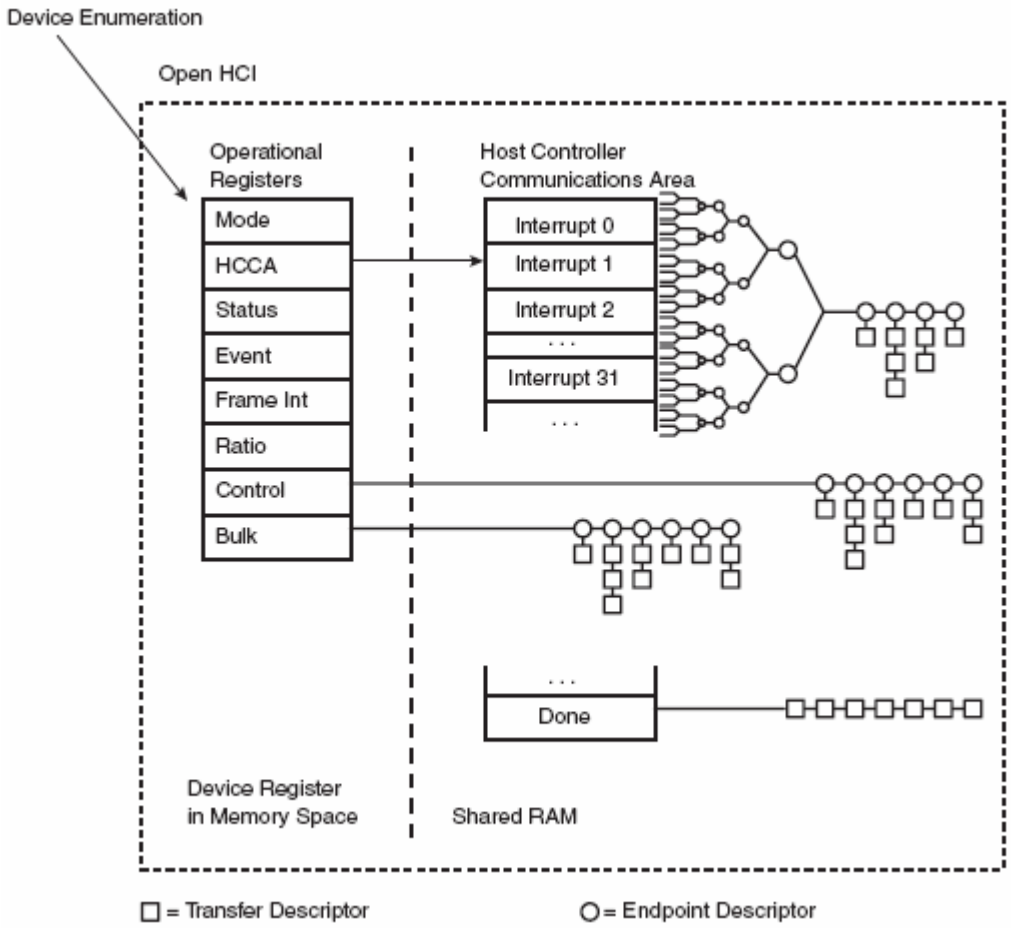
请参考 USB 发行 1.0.a 的 OHCI 规范

35.4.1 主机控制器接口

在主机控制器和主机控制器驱动器之间有两个通信通道。第一个通道使用分配在 USB 主机控制器上的一套工作寄存器。主机控制器为此通道上所有通信的目标。操作寄存器包括控制，状态和表指针寄存器。它们映射在映射区存储器。在操作寄存器集内有一个指向叫主机控制器通信区域（HCCA）的处理器空间的位置的指针。HCCA 是第二个通道。主机控制器时此通道上所有通信的主控。HCCA 包括指向中断终点描述符表的头指针，指向和帧首处理关联的完成队列和状态信息的头指针。

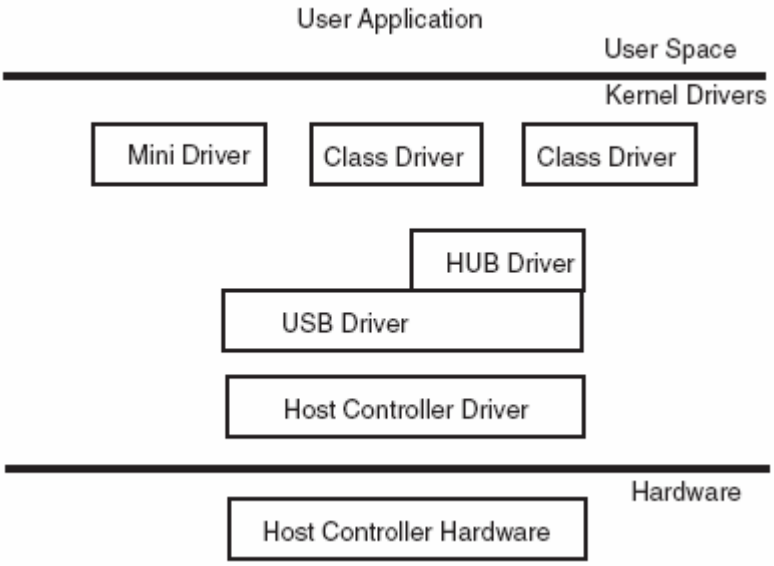
接口间通信的基本模块为端点描述符（ED，4 个双字）和传输描述符（TD，4 个或 8 个双字）。主机控制器给系统的每个端点指定一个端点描述符。对于专用端点将传输描述符队列连接到端点描述符

图 35-2 USB 主机通信通道



35.4.2　主机控制器驱动

图 35-3　USB 主机驱动

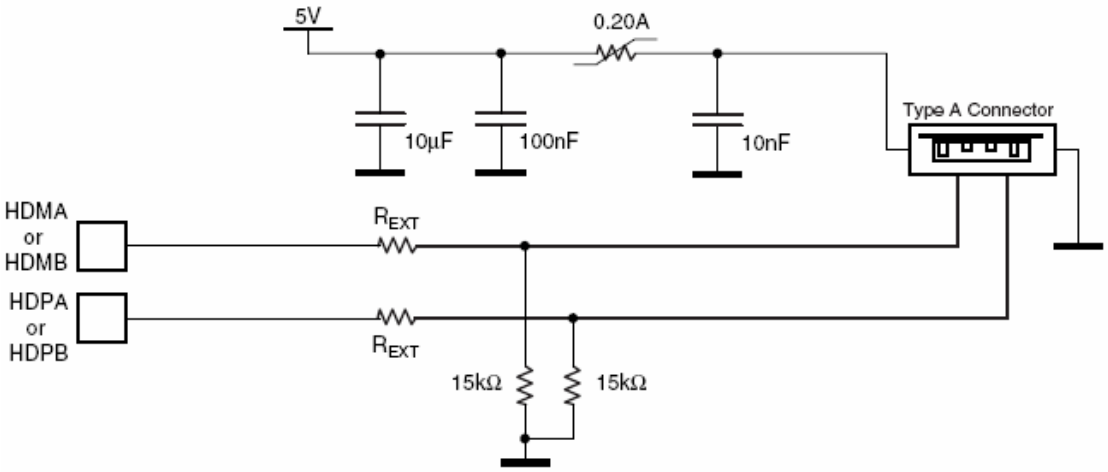


通过以下若干 USB 层完成 USB 处理：

- 主机控制器硬件和串行引擎：发送和接收总线上的 USB 数据
- 主机控制器驱动器：驱动主机控制器硬件并且处理 USB 协议
- USB 总线驱动和集线器驱动：处理 USB 命令和枚举。提供一个硬件独立接口
- 小驱动器：处理设备特定命令
- 类驱动器：处理标准器件。对于每一级设备，此驱动作为一个通用驱动器，例如 HID 驱动。

35.5 典型连接

图 35-4 连接 UHP 器件控制器的电路原理图



因为由 USB 主机端口逻辑自动的检测设备连接，所以下拉电阻必须连接到板上的 DP 和 DM。否则 USB 主机一直检测此端口上器件连接。

必须将终端串行电阻连接到 HDP 和 HDM。电阻值在产品(R_{EXT})的电气规范中定义。

36.USB 设备端口（UDP）

36.1 概述

USB 设备端口（UDP）适用于通用串行总线（USB）V2.0 全速设备规范。

每个端点可配置为几种 USB 传输类型中的一种。可以和双端 RAM 的一段或两段联合用于存储当前数据有效负载。如果使用两段，则处理器读写一个 DPR 段，而 USB 器件外设读写另一个。同步端点必须遵循此特性。因此器件工作于有两 DPR 段端点时维持最大带宽（1M 字节/秒）。

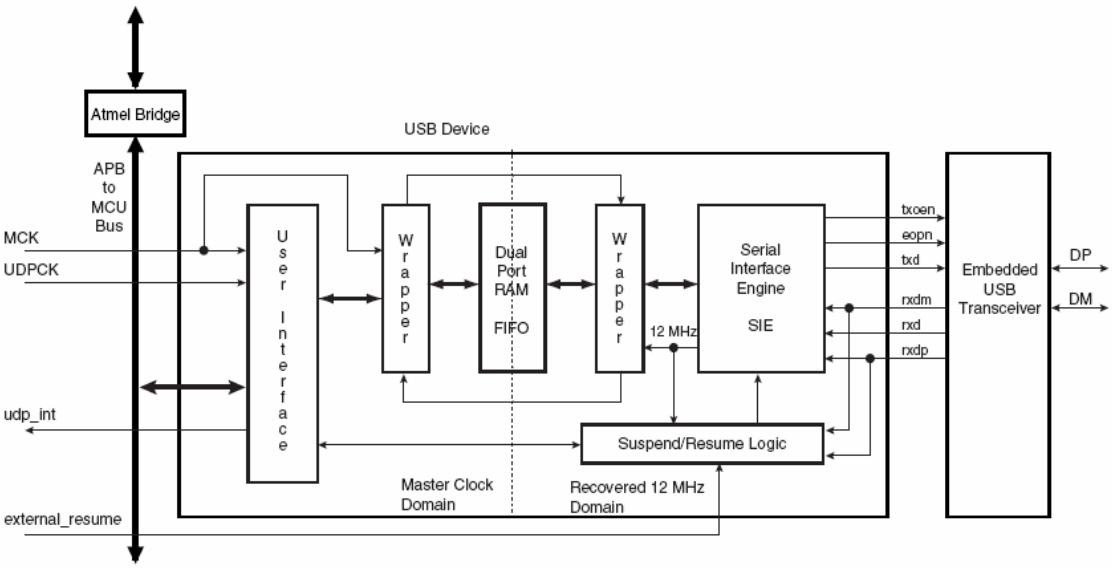
表 36-1。USB 端点说明

端点数	助记符	Dual-Bank	最大端点容量	端点类型
0	EP0	否	8	Control/Bulk/Interrupt
1	EP1	是	64	Bulk/Iso/Interrupt
2	EP2	是	64	Bulk/Iso/Interrupt
3	EP3	否	64	Control/Bulk/Interrupt
4	EP4	是	256	Bulk/Iso/Interrupt
5	EP5	是	256	Bulk/Iso/Interrupt

USB 器件自动检测挂起与恢复，通过出现中断来停止处理器。某些产品可利用外部信号唤醒 USB 主机控制器发送。

36.2 方框图

图 36-1　方框图



通过 APB 总线接口访问 UDP。通过对 APB 寄存器 8 位值读写来对数据 FIFO 进行读写。UDP 外设需要两个时钟：用于 MCK 域的外设时钟和用于 12MHz 域的 48MHz 时钟。

USB2.0 全速垫内置并且由串行接口引擎（SIE）控制。

external_resume 信号可选。它允许在系统模式下唤醒 UDP 外设。接着主机通知请求恢复的器件。列举时，此特性必须由主机处理。

36.3　附属产品

更多细节请看 USB 设备硬件实现，见产品属性文件说明。

USB 物理收发器集成在产品中。双向差分信号 DP 和 DM 对于产品边界有效。

一个 I/O 口可被用于检查来自主机的 VBUS 是否仍然有效。可使用此入口通知自供电设备主机断电。此时，必须禁用 DP 上拉电阻来防止电流流入主机。此应用应该断开收发器，然后去掉上拉电阻。

36.3.1 I/O 口

PIO 控制器不控制 DP 和 DM。内置 USB 物理收发器由 USB 器件外设控制。

为保留一个检查 VBUS 的 I/O 口，必须先对 PIO 控制器编程，将该 I/O 配置为输入 PIO 模式。

36.3.2 电源管理

USB 设备外设需要 48MHz 时钟。此时钟必须由精度为±0.25%的 PLL 产生。

因此，USB 设备从电源控制器（PMC）接收两个时钟：主控时钟，MCK，用于驱动外围用户接口，UDPCK，用于连接总线 USB 信号（恢复的 12MHz 域）。

WARNING：电源管理器（PMC）中的 UDP 外围时钟必须在对 UDP 寄存器（包括 UDP_TXCV 寄存器）任意读写操作前使能。

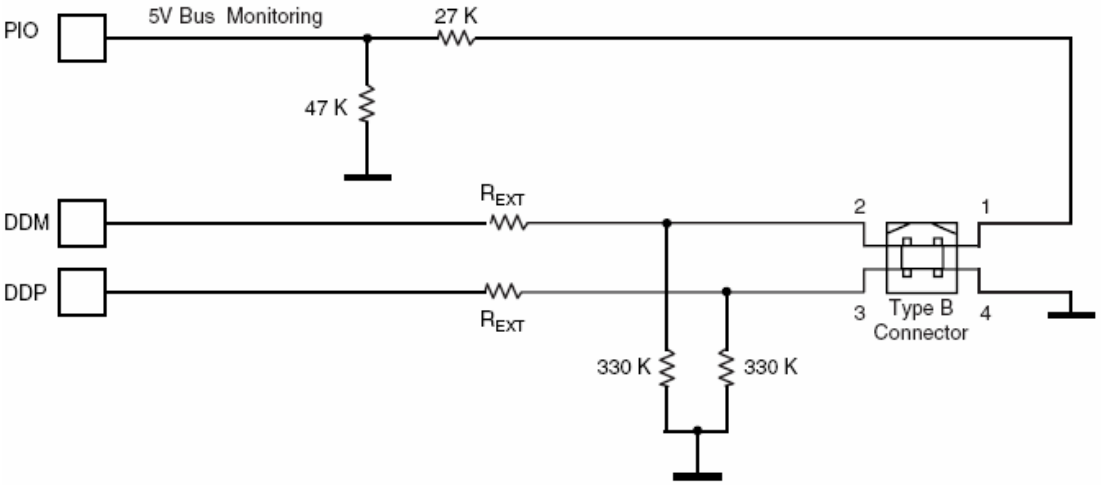
36.3.3 中断

USB 设备接口有一个连接于高级中断控制器（AIC）的中断线。

处理 USB 设备中断需要在配置 UDP 前编程 AIC。

36.4 典型连接图

图 36-2 连接器件外设的连接图



36.4.1 USB 设备收发器

USB 设备收发器内置在产品中。一些需要的分离元件如下：

- 该应用检测所有设备在第 9 章 USB 规范中定义的状态；
  ——VBUS 监控
- 为了减少功率损耗断开主机
- 口线终端

36.4.2 VBUS 监控

VBUS 监控需要检测主机连接。VBUS 监控通过使用一个可禁用的内部上拉电阻的标准 PIO。当关掉主机，则主机应该看作是断开的，必须禁用上拉电阻来防止主机通过上拉电阻寄存器耗电。

当主机断开并且使能收发器，则 DDP 和 DDM 悬空。

这可能导致过损耗。一个解决方案是在 DP 和 DM 上连接 330KΩ的下拉电阻。这些下拉电阻不改变 DDP 和 DDM 信号的完整性。

终端串行电阻必须连接到 DP 和 DM。寄存器值定义在产品(R_{EXT})的电气规范

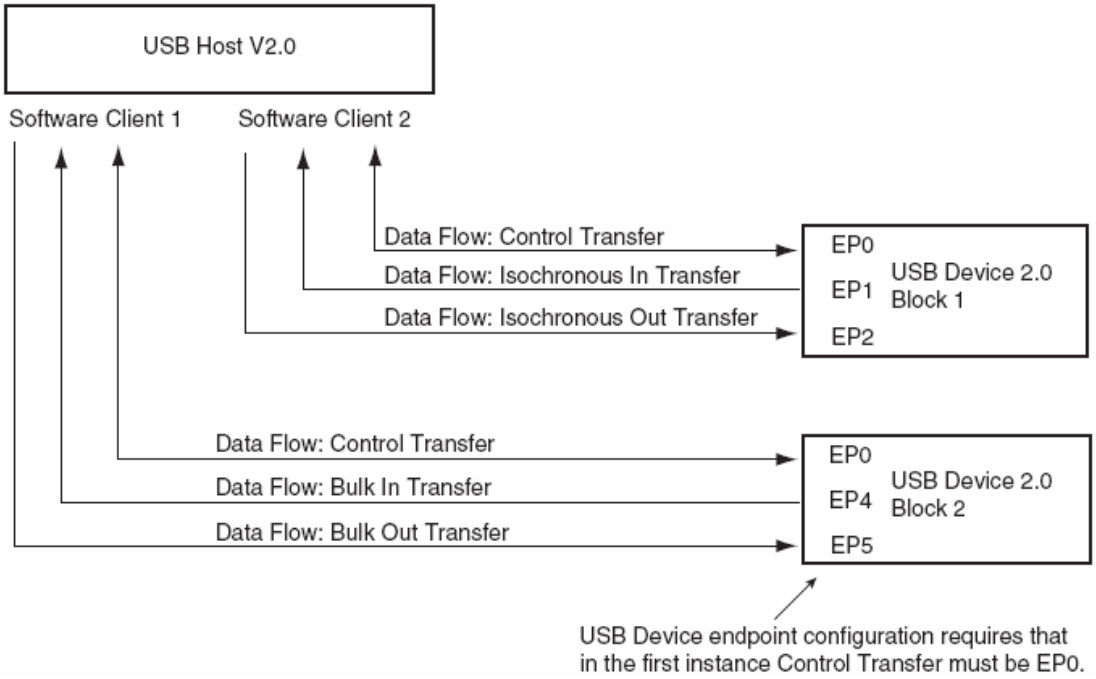
中。

36.5 功能说明

36.5.1 USBV2.0 全速介绍

USBV2.0 全速提供主机和附属 USB 器件间的通信服务。每个器件提供与每个端点相关的通信流。软件通过通信流实现主机与 USB 器件的通信。

图 36-3 USBV2.0 全速通信控制举例



当首先配置 USB 器件（USBV2.0 规范）时总是使用控制传输端点 EP0。

36.5.1.1 USBV2.0 全速传输类型

由 USB 器件自身定义的传输类型携带通信流。

表 36-2 USB 通信流

传输	方向	带宽	端点大小	错误检测	重试
控制	双向	无保证	8，16，32，64	是	自动
同步	单向	保证	256	是	否
中断	单向	无保证	8，16，32，64	是	是
批	单向	无保证	8，16，32，64	是	是

36.5.1.2 USB 总线处理

每次传送将在 USB 总线上进行一次或多次处理。总线上有以下三种数据包处理：

1.设置处理

2.数据传入处理

3.数据传出处理

36.5.1.3 USB 传输事件定义

如下表，传输在 USB 总线上顺序进行。

表 36-3 USB 传输事件

控制传输（1）（3）	设置处理>数据传入处理>状态传出处理 设置处理>数据传出处理>状态传入处理 设置处理>状态传入处理
中断进入传输（设备到主机）	数据传入处理>数据传入处理
中断输出传输（主机到设备）	数据传出处理>数据传出处理
同步传入传输（2）（主机到设备）	数据传入处理>数据传入处理
同步传出传输（2）（设备到主机）	数据传出处理>数据传出处理
批传入传输（设备到主机）	数据传入处理>数据传入处理
批传出传输（主机到设备）	数据传出处理>数据传出处理

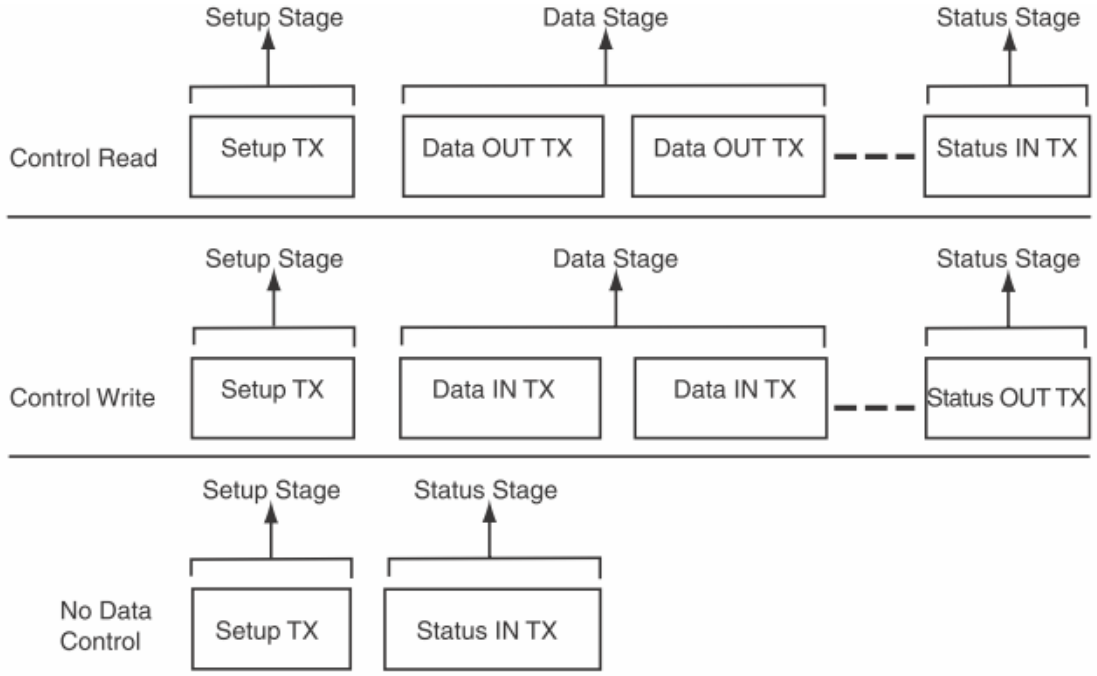
注意：1.控制传输必须使用无 ping-pang 属性的端点

2.同步传输必须使用 ping-pang 属性的端点

3.可使用延迟握手中止控制传输

状态处理是一个只用一个控制传输的主机到设备处理的特殊类型。此控制传输必须使用无 ping-pong 属性的端点。根据控制顺序（读或写），USB 设备发送或接收一次状态处理。

图 36-4 控制读写顺序



注意：1.在 Status IN 阶段，主机等待一个从设备来的使用 DATA1 PID 的长度为 0 的数据包（无数据的 Data IN 处理）。更多关于协议层的细节参考 Universal Serial Bus Specification, Rev. 2.0 第 8 章

2.在 Status OUT 阶段，主机发射一个长度为 0 的数据包到设备（无数据的 Data OUT 处理）。

36.5.2 与 USB V2.0 器件外设交互处理

36.5.2.1 设置处理

设置是在控制传输期间使用的主机到器件之间的一个特殊类型。必须使用无 ping-pong 特性的端点执行控制传输。设置处理必须尽可能快的由固件处理。它使用从主机到设备的传送请求。这些请求由 USB 设备处理并可能需要更多的变量。

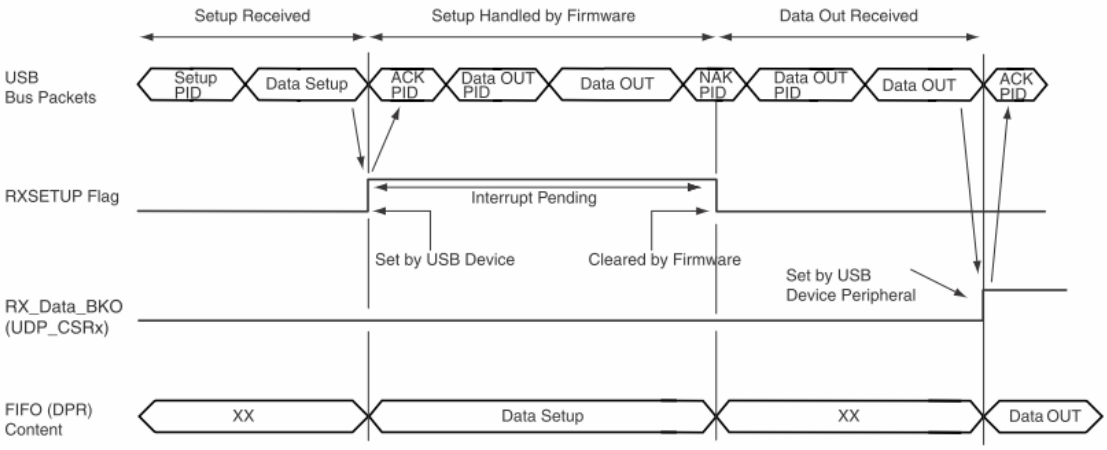
这些变量由设置处理后的 Data OUT 处理发送到器件。这些请求也可能返回数据。返回的数据被设置处理后的 Data IN 处理送入主机。状态处理结束整个控制传输。

当 USB 端点接收到一个设置传输：

- USB 器件自动应答此设置数据包
- 在 UDP_CSRx 寄存器设置 RXSETUP
- 当 RXSETUP 未被清零时产生端点中断，如果该设备中断使能，则该中断被送到微控制器执行。

因此，固件必须轮询 UDP_CSRx 检测 RXSETUP 或获取中断，读取 FIFO 中的设置数据包，然后清零 RXSETUP。在未读取 FIFO 中的设置数据包前不能清零 RXSETUP。否则，USB 设备将接收下一次 Data OUT 传输并覆盖 FIFO 中的设置数据包。

图 36-5 设置处理和 Data OUT 处理



36.5.2.2 Data IN 处理

Data IN 处理被用在控制传输，同步传输，批传输和中断传输，并引导数据从设备到主机的传输。同步传输中的 Data IN 处理必须使用 ping-pong 属性的端点。

36.5.2.3 使用无 ping-pong 属性的端点

使用无 ping-pong 属性执行 Data IN 处理：

1.应用程序检查是否通过轮询端点 UDP_CSRx 寄存器的 TXPKTRDY 对 FIFO 写入 (TXPKTRDY 必须清零)。

2.应用程序将准备发送的第一个数据写入端点的 FIFO 中，给端点的 UDP_FDRx 寄存器写入 0 或多字节值。

3.应用程序通过设置端点的 UDP_CSRx 寄存器中的 TXPKTRDY 通知 USB 外设它已完成。

4.当端点的 UDP_CSRx 寄存器中的 TXCOMP 被置位时应用程序通知端点的 FIFO 已经被 USB 器件释放。然后当 TXCOMP 置位时相应端点的中断被挂起。

5.微处理器将准备发送的第二个数据包写入到端点 FIFO，在端点 UDP_

FDRx 寄存器写入 0 或多字节值。

6.微控制器通过设置端点的 UDP_CSRx 寄存器中的 TXPKTRDY 通知 USB 外设已完成。

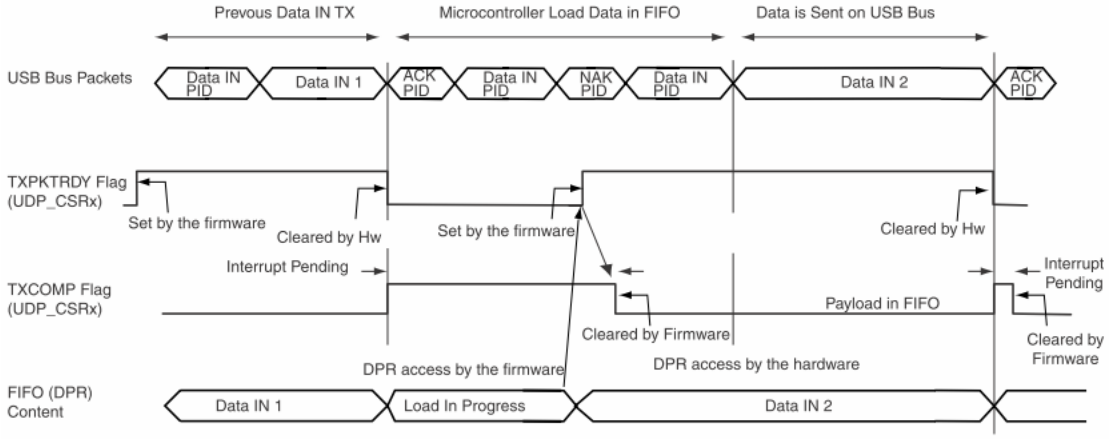
7.应用程序清零端点的 UDP_ CSRx 中的 TXCOMP。

最后一个数据包被发送后，只要 TXCOMP 被置位则应用程序必须清零 TXCOMP。当接收到一个 Data IN 数据包的 ACK PID 信号时由 USB 器件置位 TXCOMP。当 TXCOMP 置位时中断挂起。

警告：TX_PKTRDY 被置位后必须清零 TX_COMP。

注意：关于 Data IN 协议层的更多细节请参考 Universal Serial Bus Specification, Rev 2.0 的第 8 章。

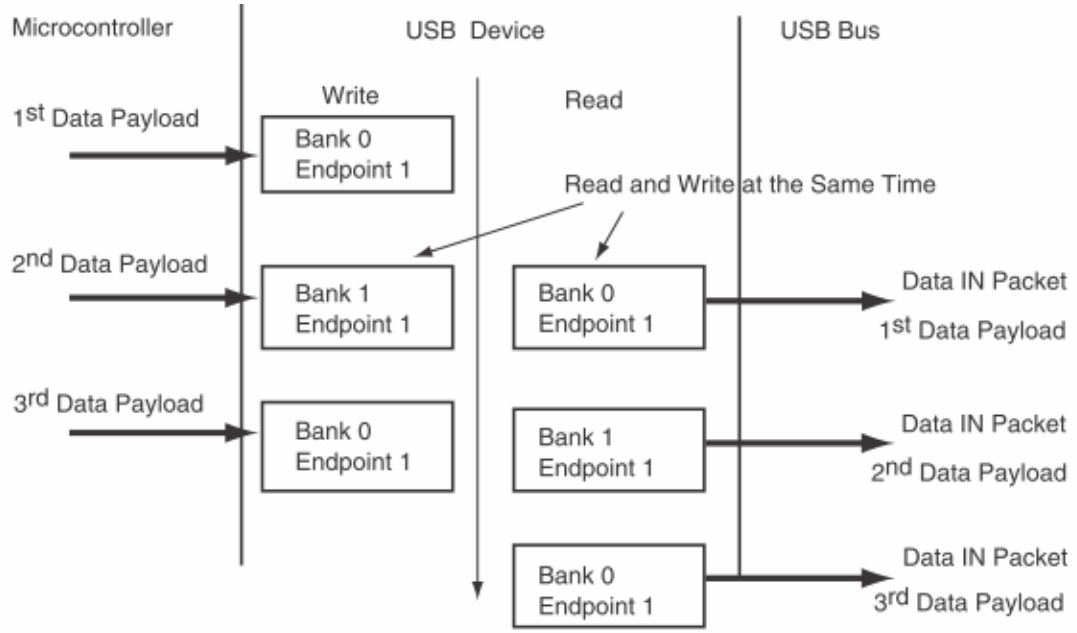
图 36-6 无 Ping- pong 端点的 Data IN 传输



36.5.2.4 使用 ping_pong 属性的端点

在同步传输期间需要使用到 ping_pong 属性的端点。也允许在此传输期间处理 USB 规范定义的最大带宽。为保证带宽为常数或最大带宽，在当前数据被 USB 器件发送的同时微控制器必须准备下一个要发送的数据的有效负载。因此要使用两个存储器段。一个给微控制器，另一个由 USB 器件锁定。

图 36-7 ping_pong 端点 Data IN 传输的段交换



当使用一个 ping-pong 端点，执行 Data IN 处理时需要以下程序：

1.微控制器通过轮询端点 UDP_CSRx 寄存器的 TXPKTRDY 是否清零来检查是否能对 FIFO 写入 。

2.微控制器将准备发送的第一个数据的有效负载写入端点的 FIFO 中，给端点 UDP_FDRx 寄存器写入 0 或多字节值。

3.微控制器通过设置端点的 UDP_CSRx 寄存器中的 TXPKTRDY 通知 USB 外设，它已完成对 FIFO(bank 0）的写入。

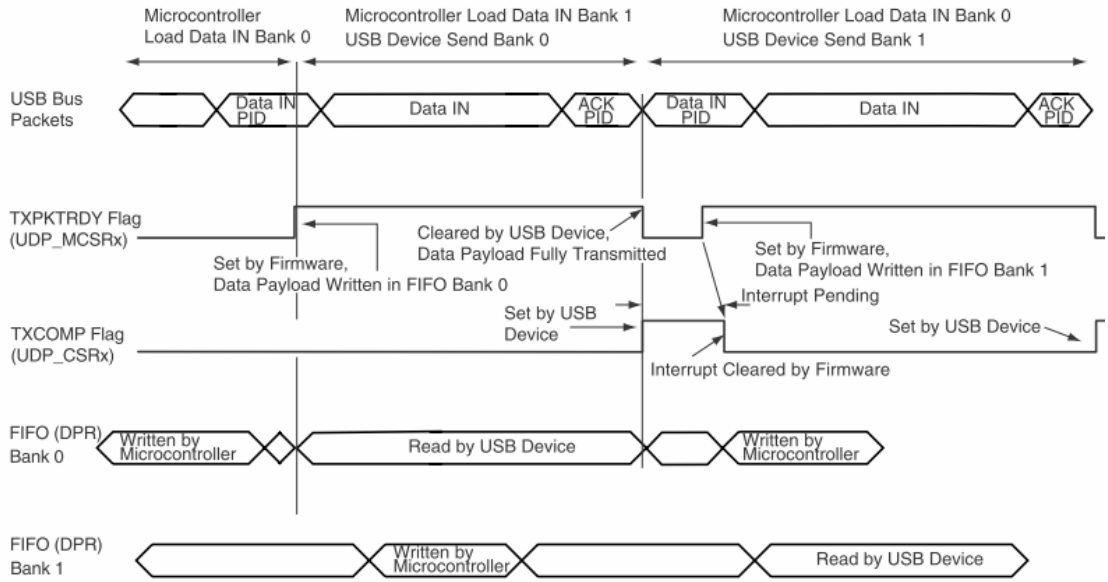
4.不必等待 TXPKTRDY 被清零，微控制器将准备发送的第二个数据的有效负载写入到 FIFO（bank 1），对端点 UDP_ FDRx 寄存器写入 0 或多个字节值。

5.当端点 UDP_ CSRx 寄存器中的 TXCOMP 置位时，由 USB 器件通知微控制器已经释放第一个存储器段。当 TXCOMP 被置位则中断挂起。

6.一旦微控制器接收到第一个存储器段的 TXCOMP 时，则拉高端点 UDP_ CSRx 寄存器的 TXPKTRDY 通知 USB 器件已准备好将要发送的第二个存储段。

7.此步骤中，Bank 0 可用并且微控制器可以准备将要发送的第三个数据的有效负载。

图 36-8 ping-pong 端点的 Data IN 传输



警告：这是软件关键路径，由于一旦第二个段已满，则驱动器不得不等待 TX_COMP 置位 TX_PKTRDY。如果在接收 TX_COMP 置位 和 TX_PKTRDY 置位之间的延迟太长，则一些 Data IN 数据包可能无应答，减小带宽。

警告： TX_PKTRDY 被置位后必须清零 TX_COMP。

36.5.2.5 Data OUT 处理

Data OUT 处理被用在控制，同步，批和中断传输中并引导数据从主机到器件的传输。同步传输中的 Data OUT 处理必须使用 ping-pong 属性的端点。

36.5.2.6 无 ping-pong 属性的 Data OUT 处理

使用无 ping-pong 属性的端点执行 Data OUT 处理：

1.主机产生一个 Data OUT 数据包。

2.此数据包由 USB 器件端点接收。当与该端点有关的 FIFO 正在被微控制器使用时，则返回一个 NAK PID 到主机。一旦 FIFO 可用，则由 USB 器件将数据写入到 FIFO 并且自动发送 ACK 到主机。

3.通过轮询端点 UDP_ CSRx 寄存器的 RX_DATA_BK0 通知微控制器 USB 器件已接收到数据的有效负载。当 RX_DATA_BK0 置位则挂起该端点的中断。

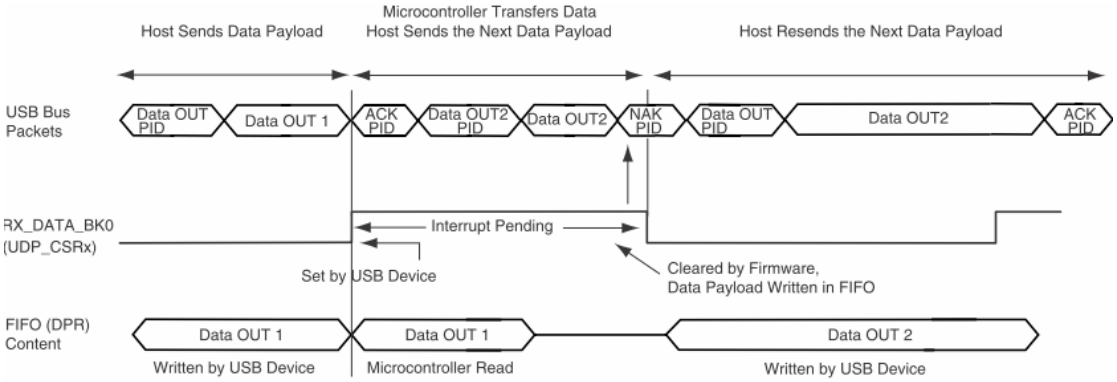
4.通过读取端点 UDP_ CSRx 寄存器的 RXBYTECNT 得到 FIFO 中的字节数。

5.微控制器接收端点存储器的数据到其存储器中。通过读取端点 UDP_ FDRx 寄存器得到接收的数据。

6.微控制器通过清零端点 UDP_ CSRx 寄存器中的 RX_DATA_BK0 通知 USB 器件已完成传输。

7.USB 器件可接收一个新的 Data OUT 数据包。

图 36-9 无 ping-pong 端点的 Data OUT 传输

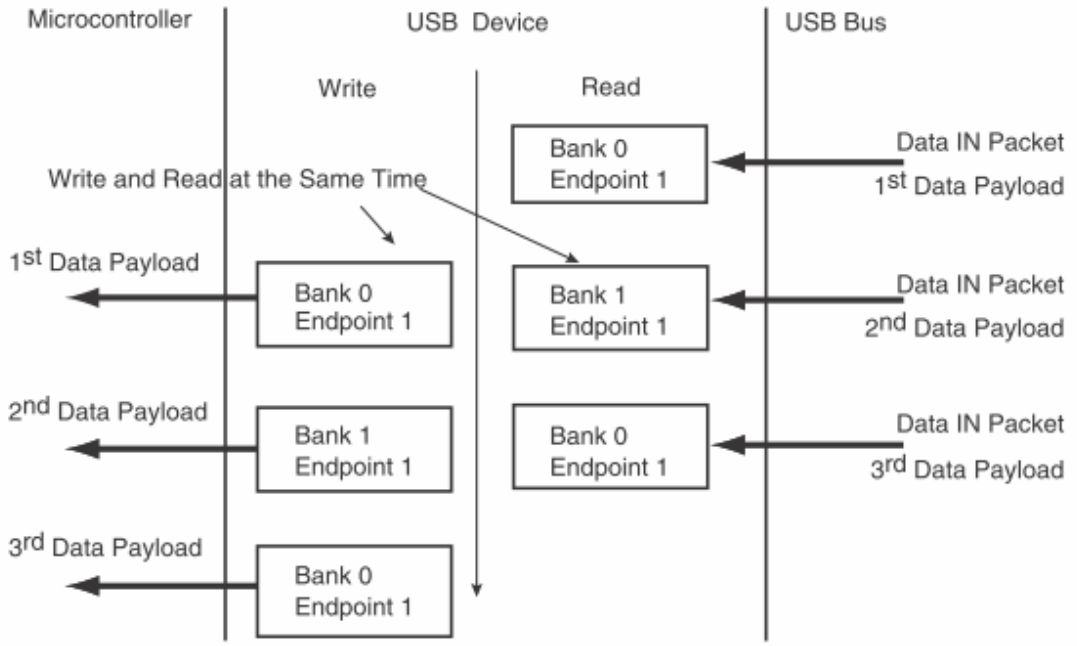


当 RX_DATA_BK0 置位时中断挂起。在 RX_DATA_BK0 被清零后，USB 器件，FIFO 和微控制器存储器间的存储器传输不能进行。否则，USB 器件将接收下一个 Data OUT 传输并且覆盖 FIFO 中当前 Data OUT 数据包。

36.5.2.7 使用 ping-pong 属性的端点

同步传输期间，必须使用 ping-pong 属性的端点。为保证一个常数带宽，当 USB 器件接收到当前数据有效负载时，微控制器必须读取由主机发送的前一个数据有效负载。所以使用两个存储器段。一个给微控制器，另一个由 USB 器件锁定。

图 36-10 ping-pong 端点 Data OUT 传输中的段交换



当使用 ping-pong 端点，执行 Data OUT 处理需要以下步骤：

1.主机产生一个 Data OUT 数据包。

2.该数据包由 USB 器件端点接收。写入到端点的 FIFO 的 Bank0.

3.USB 器件发送一个 ACK PID 数据包到主机。主机可立即发送第二个数据包。由器件接收并复制到 FIFO 的 Bank1.

4.通过轮询端点 UDP_ CSRx 寄存器的 RX_DATA_BK0 通知微控制器 USB 器件已接收到一个数据有效负载。当 RX_DATA_BK0 置位时则挂起该端点的中断。

5.通过读取端点 UDP_ CSRx 寄存器的 RXBYTECNT 得到 FIFO 中的字节数。

6.微控制器将接收到的数据从端点存储器传输到微控制器的存储器。通过读取端点 UDP_ FDRx 寄存器获得接收到的数据。

7.微控制器通过清零端点 UDP_ CSRx 寄存器的 RX_DATA_BK0 通知 USB 外设已完成传输。

8.可以由 USB 外设接收第三个 Data OUT 数据包并且复制到 FIFO 的 Bank0

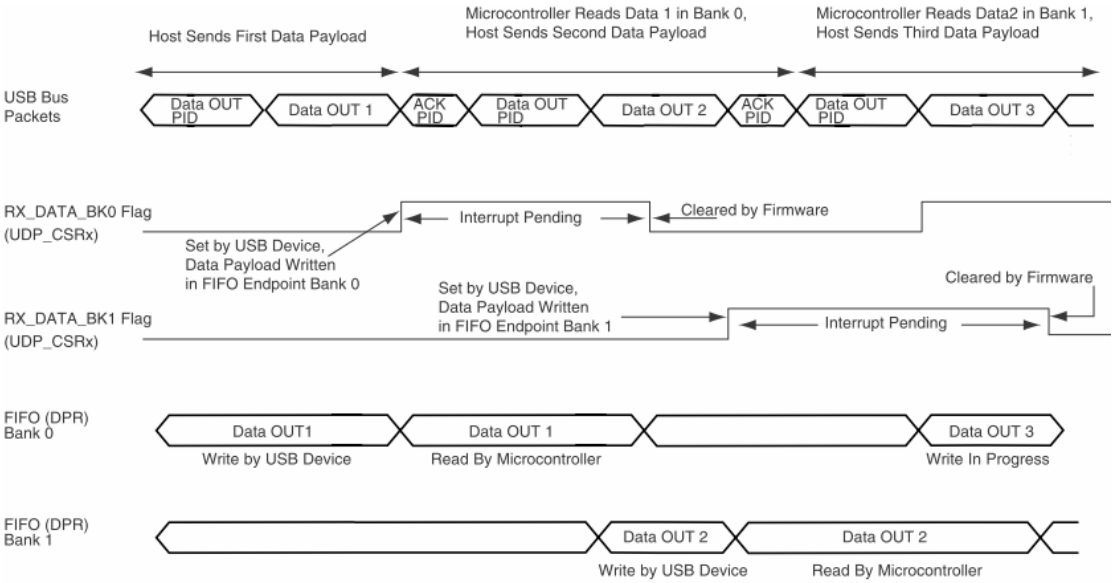
9.如果已经接收到第二个 Data OUT 数据包，通过置位端点 UDP_ CSRx 寄存器的 RX_DATA_BK1 通知微控制器。当 RX_DATA_BK1 置位时则挂起该端点的中断。

10.微控制器将接收到的数据从端点存储器传输到微控制器的存储器。可通过读取端点 UDP_ FDRx 寄存器获得接收到的数据。

11.微控制器通过清零端点 UDP_ CSRx 寄存器的 RX_DATA_BK1 通知 USB 器件已完成传输。

12.USB 器件可接收第四个 Data OUT 数据包并且复制到 FIFO 的 Bank0.

图 36-11 ping-pong 端点的 Data OUT 传输



注意：当 RX_DATA_BK0 或 RX_DATA_BK1 置位则挂起中断。

警告：当 RX_DATA_BK0 和 RX_DATA_BK1 都被置位，则无法确定首先对哪一个清零。因此必须用软件设立一个内部计数器确保能先清零 RX_DATA_BK0 然后清零 RX_DATA_BK1 。该状况可能在应用软件在别处忙且两个存储段被 USB 主机填满时出现。一旦应用程序返回 USB 驱动器，则两个标志位置位。

36.5.2.8 停止握手

停止握手可用在以下两种情形之一。(更多关于停止握手，参考 Universal Serial Bus Specification, Rev 2.0 第 8 章)

- 当和端点相关的停止特性设置时使用功能停止（更多关于停止特性，参考 Universal Serial Bus Specification, Rev 2.0 第 9 章）。
- 中端当前请求，使用协议停止，仅用于控制传输。

以下步骤产生一个停止数据包：

1.微控制器设置端点 UDP_ CSRx 寄存器的 FORCESTALL 标志位。

2.主机接收停止数据包

3.通过轮询 STALLSENT 设置来通知微控制器，器件已停止发送数据。当 STALLSENT 置位，挂起端点中断。微控制器必须清零 STALLSENT 来清除中断。

当停止握手后接到一个设置处理，必须清零 STALLSENT 以防止由于 STALLSENT 被置位产生的中断。

图 36-12 停止握手（Data IN 传输）

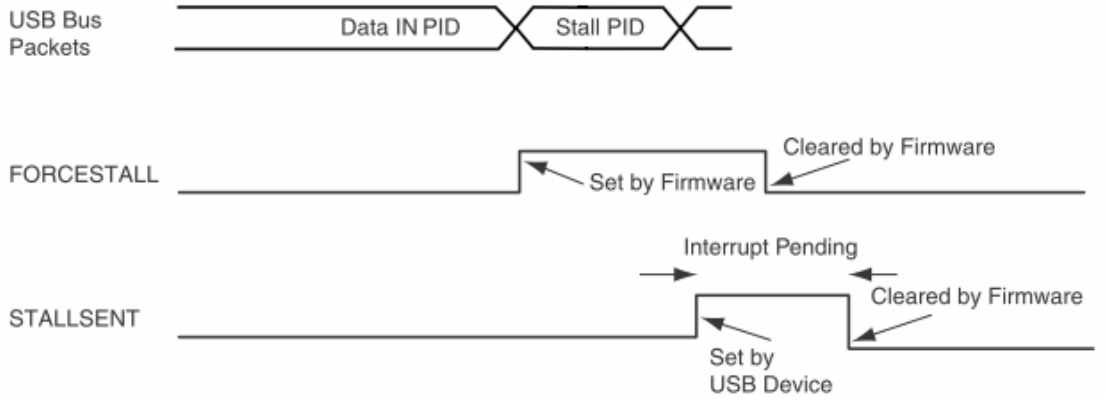
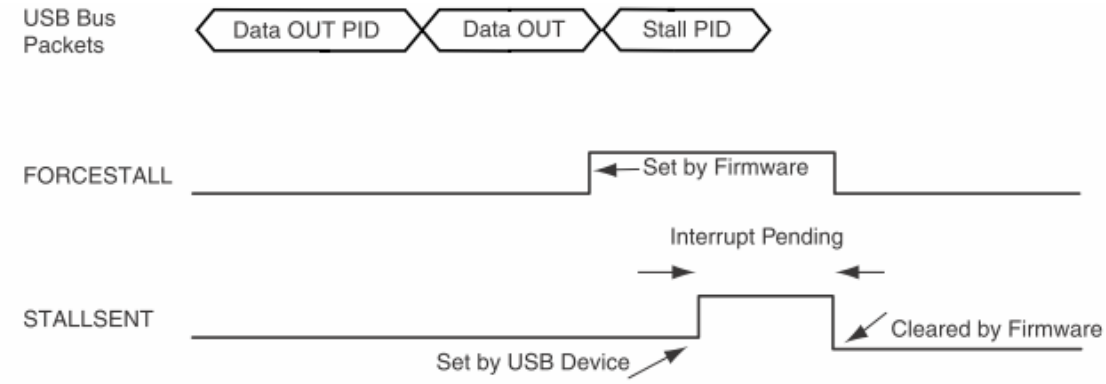


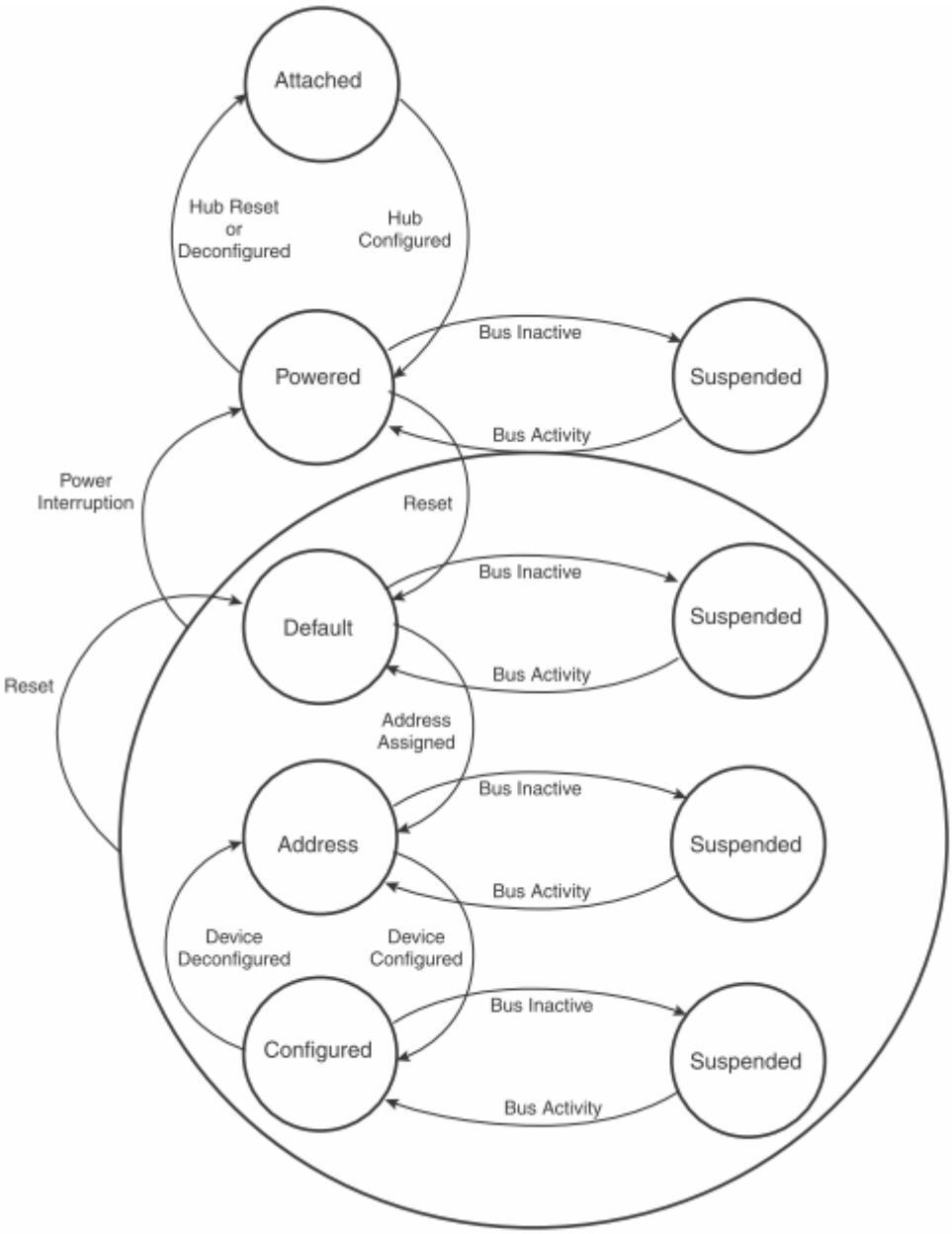
图 36-13 停止握手（Data OUT 传输）



36.5.3 控制器件状态

一个 USB 器件有几种可能的状态。参考 Universal Serial Bus Specification, Rev 2.0 第 9 章。

图 36-14 USB 器件状态框图



一种状态到另一种状态的转换取决于 USB 总线状态或通过默认端点（端点 0）的控制处理发送标准请求。

在一个总线无效周期后，USB 设备进入悬挂模式。强制从 USB 主机接受挂起/恢复请求。挂起模式对总线供电应用的要求非常严格；器件在 USB 总线上消耗不超过 500 微安。

在挂起模式，主机可通过发送一个恢复信号（总线激活）来唤醒一个器件或者 USB 器件可以发送一个唤醒请求给主机，例如，通过移动鼠标唤醒 PC。

唤醒特性并不是对所有器件都强制执行，并必须与主机协商。

36.5.3.1　无供电状态

自供电设备可使用一个称作典型连接单元的 PIO 检测到 5V VBUS。当设备未连接到主机，可通过对 UDP 禁用 MCK，禁用 UDPCK 和禁用收发器减小设备功耗。DDP 和 DDM 接口需要 330 千欧电阻下拉电阻。

36.5.3.2 进入连接状态

当未连接设备时，USB DP 和 DM 信号被通过集成在集线器下游端口的 15 千欧下拉电阻连接到 GND。当器件连接到集线器下游端口，则器件在 DP 上连接了一个 1.5 千欧的上拉电阻。USB 总线接口进入 IDLE 状态，DP 被器件的 1.5 千欧电阻上拉到 3.3V，DM 被主机的 15 千欧电阻拉低。为使能集成的上拉功能，UDP_TXVC 寄存器中的 PUON 位必须置位。要使能集成的上拉，USB_PUCR 总线矩阵寄存器的 UDP_PUP_ON 位必须置位。

警告：为能对 UDP_TXVC 寄存器写入数据，必须在 UDP 上使能 MCK 时钟。在电源管理控制器中实现。

上拉连接后，设备进入供电状态。该状态下，必须在电源管理控制器中使能 UDPCK 和 MCK。收发器可保持禁用。

36.5.3.3 从上电状态到默认状态

连接到一个 USB 主机后，USB 器件要等待一个 end-of-bus 的复位。在 UDP_ISR 寄存器中置位不可屏蔽标志位 ENDBUSRES 并且触发一个中断。

一旦 ENDBUSRES 中断已被触发，则设备进入默认状态。该状态下，UDP 软件必须：

- 使能默认端点，置位 UDP_CSR[0]寄存器中的 EPEDS 标志位，并通过向 UDP_IER 寄存器写入 1 有选择的使能端点 0 的中断。
- 配置已被 USB 复位检测器复位的中断屏蔽寄存器 。
- 清零 UDP_TXVC 寄存器中的 TXVDIS 标志位使能收发器

该状态必须使能 UDPCK 和 MCK。

警告：每次触发 ENDBUSRES 中断时，确保已复位中断屏蔽寄存器和 UDP_CSR 寄存器

36.5.3.4 从默认状态到地址状态

在设置地址标准器件请求后，USB 主机外设进入地址状态。

警告：在器件进入地址状态前，必须完成控制传输的 Status IN 处理，也就是，一旦已接收到 UDP_CSR[0] 寄存器中的 TXCOMP 标志位并被清零则 UDP 器件要设置新的地址。

为了转换到地址状态，驱动软件在 UDP_GLB_STAT 寄存器中设置 FADDEN 标志位，设置其寻址，并设置 UDP_FADDR 寄存器中的 FEN 位。

36.5.3.5 从地址状态到配置状态

一旦已接收到一个有效的设置地址标准器件请求并应答后，则器件使能相应当前配置的端点。通过设置 UDP_CSRx 寄存器中的 EPEDS 和 EPTYPE 域完成，并可使能 UDP_IER 寄存器中的相关中断（可选）。

36.5.3.6 进入挂起状态

当检测到挂起状态（在 USB 总线上无总线动作），则 UDP_ISR 寄存器中的 RXSUSP 信号置位。如果 UDP_IMR 寄存器中的相关位置位则触发中断。通过向 UDP_ICR 寄存器写入数据清零该标志位。然后设备进入挂起状态。

该状态下总线供电设备从 5V VBUS 的漏电流必须小于 500 微安。例如微控制器转换到低速时钟，禁用 PLL 和主振荡器，并进入空闲状态。还可能关闭板上其他器件。

可关闭 USB 器件外设时钟。异步检测 Resume 事件。可在电源管理控制器中关闭 MCK 和 UDPCK 并且可通过 UDP_TXVC 寄存器中的 TXVDIS 域禁用 USB 收发器。

警告：仅在 MCK 对外设使能时允许向 UDP 寄存器读，写。关闭 UDP 外设的 MCK 必须是在向 UDP_TXVC 写入并应答 RXSUSP 后的最后一个操作。

36.5.3.7　接收主机恢复

挂起模式下，异步检测 USB 总线上的恢复事件，收发器和时钟是禁用的（但是上拉仍有效）。

一旦在总线上检测到恢复事件，UDP_ISR 中的 WAKEUP 信号置位。如果 UDP_IMR 寄存器的相关位置位则可能产生中断。该中断可用于唤醒内核，使能 PLL 和主振荡器并配置时钟。

警告：仅当 UDP 外设的 MCK 使能时允许向 UDP 寄存器读写。必须在清零 UDP_ICR 寄存器中的 WAKEUP 位和 UDP_TXVC 寄存器中的 TXVDIS 位前使能 UDP 的 MCK。

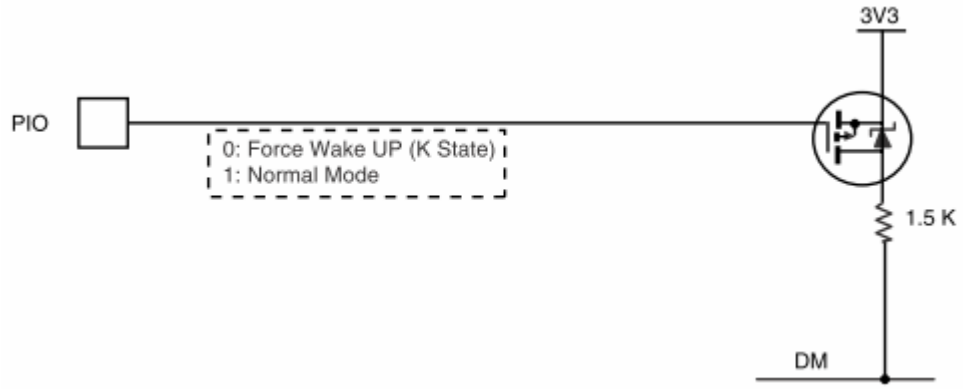
36.5.3.8　发送设备远程唤醒

挂起模式下发送一个外部恢复事件唤醒主机。

- 在器件进入挂起状态后，至少等待 5ms 才可发送外部恢复事件。
- 器件有 10ms 时间开始灌入电流并驱动 K 状态来恢复主机。
- 器件必须在 1 到 15ms 内强制一个 K 状态来恢复主机

为了向总线驱动 K 状态（DM3.3V，DP 连接到 GND），可连接一个晶体管上拉到 DM。可通过禁用 DP 的上拉和使能 DM 的上拉获得 K 状态。可在应用程序控制。

图 36-15 驱动 K 状态的原理图



36.6 USB 器件端口（UDP）用户接口

警告：必须在向 UDP 寄存器包括 UDP_TXCV 寄存器的任何读写前使能电源管理控制器中的 UDP 外设时钟。

表 36-4 UDP 存储器分配

偏移量	寄存器	名称	访问类型	复位状态
0x000	帧数寄存器	UDP_ FRM_NUM	读	0x0000_0000
0x004	全局状态寄存器	UDP_ GLB_STAT	读写	0x0000_0000
0x008	功能地址寄存器	UDP_ FADDR	读写	0x0000_0100
0x00c	保留	-	-	
0x010	中断使能寄存器	UDP_ IER	写	
0x014	中断禁用寄存器	UDP_ IDR	写	
0x018	中断屏蔽寄存器	UDP_ IMR	读	0x0000_1200
0x01c	中断状态寄存器	UDP_ ISR	读	0x0000_XX00
0x020	中断清除寄存器	UDP_ ICR	写	
0x024	保留	-	-	
0x028	复位端点寄存器	UDP_ RST_EP	读写	
0x02c	保留	-		
0x030	端点 0 控制和状态寄存器	UDP_CSR0	读写	0x0000_0000
见注 1	端点 5 控制和状态寄存器	UDP_CSR5	读写	0x0000_0000
0x050	端点 0 FIFO 数据寄存器	UDP_ FDR0	读写	0x0000_0000
见注 2	端点 5 FIFO 数据寄存器	UDP_ FDR5	读写	0x0000_0000
0x070	保留	-		
0x074	收发器控制寄存器	UDP_ TXVC (3)	读写	0x0000_0000
0x078-0x FC	保留			

注意：1.UDP_ CSRx 寄存器地址算法： 0x030 + 4(端点数 - 1)

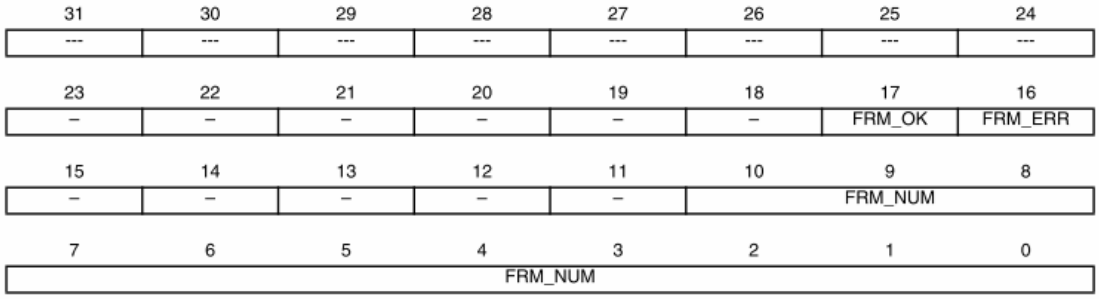
2.UDP_ FDRx 寄存器地主算法： 0x050 + 4(端点数 - 1)

3.见此页“UDP 存储器分配”上面的警告。

36.6.1 UDP 帧数寄存器

寄存器名称：UDP_ FRM_NUM

访问类型：只读



- FRM_NUM[10:0]:在数据包域格式中定义的帧数

每有一帧，主机将这 11 位值加一。每帧开始时更新此值。

在 SOF_EOP（帧开始，数据包结束） 时更新此值。

- FRM_ERR: 帧错误

当接收到的 SOF 包中有错时，在 SOF_EOP 中的该位置位。

收到 SOF_PID 后该位复位。

- FRM_OK: 帧正确

当接收到的 SOF 包中无错时，在 SOF_EOP 中的该位置位。

收到 SOF_PID 后该位复位 (包标识)。

中断状态寄存器中，收到 SOF_PID 时更新 SOF 中断。该位置位不必等待 EOP。

注意:在 8 位寄存器接口中， FRM_OK 为 FRM_NUM_H 的位 4， FRM_ERR 为 FRM_NUM_L 的位 3。

36.6.2 UDP 全局状态寄存器

寄存器名称 :UDP_GLB_STAT

访问类型 :读 / 写

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

该寄存器用于按照 USB Serial Bus Specification, Rev.2.0 第 9 章来获得并设置器件状态。

- FADDEN: 功能地址使能

读：

0 = 器件不处于地址状态。

1 = 器件处于地址状态。

写：

0 = 无效，只有复位能将器件带回默认状态。

1 = 将器件设置为地址状态。在成功设置地址请求后出现。在此之前，必须用设置地址参数将 UDP_FADDR 寄存器初始化。在设置 FADDEN 前设置地址必须完成状态阶段，详见 Universal Serial Bus Specification, Rev. 2.0 第 9 章。

- CONFG: 配置

读：

0 = 器件不处于配置状态。

1 = 器件处于配置状态。

写：

0 = 设置器件到非配置状态。

1 = 设置器件到配置状态。

当处于地址状态且接收到成功设置配置请求时将器件置于配置状态，详见 Universal Serial Bus Specification, Rev. 2.0 第 9 章。

36.6.3 UDP 功能地址寄存器

寄存器名称 : UDP_FADDR

访问类型 :读 / 写

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

- FADD[6:0]: 功能地址值

一旦从主机接收到设置地址请求，并处于无数据控制序列的状态阶段，固件必须对功能地址值编程，参考 Universal Serial Bus Specification，Rev. 2.0 。上电或复位后，功能地址值设置为 0。

- FEN: 功能使能

读：

0 = 功能端点禁用。

1 = 功能端点使能。

写：

0 = 禁用功能端点。

1 = 默认值。

功能使能位 (FEN) 允许微控制器使能或禁用功能端点。从主机接收到复位后微控制器设置该位。一旦该位被设置， USB 器件被主机接受并可与主机传输数据包。

36.6.4 UDP 中断使能寄存器

寄存器名称 :UDP_IER

访问类型 :只写

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	WAKEUP	–	SOFINT	–	RXRSM	RXSUSP
7	6	5	4	3	2	1	0
–	–	EP5INT	EP4INT	EP3INT	EP2INT	EP1INT	EP0INT

- EP0INT: 使能端点 0 中断
- EP1INT: 使能端点 1 中断
- EP2INT: 使能端点 2 中断
- EP3INT: 使能端点 3 中断
- EP4INT: 使能端点 4 中断
- EP5INT: 使能端点 5 中断

0 = 无效。

1 = 使能相应端点中断。

- RXSUSP: 使能 UDP 挂起中断

0 = 无效。

1 = 使能 UDP 挂起中断。

- RXRSM: 使能 UDP 恢复中断

0 = 无效。

1 = 使能 UDP 恢复中断。

- SOFINT: 使能帧起始中断

0 = 无效。

1 = 使能帧起始中断。

- WAKEUP: 使能 UDP 总线唤醒中断

0 = 无效。

1 = 使能 USB 总线中断。

36.6.5 UDP 中断禁用寄存器

寄存器名称 :UDP_IDR

访问类型 :只写

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	WAKEUP	–	SOFINT	–	RXRSM	RXSUSP
7	6	5	4	3	2	1	0
–	–	EP5INT	EP4INT	EP3INT	EP2INT	EP1INT	EP0INT

- EP0INT: 禁用端点 0 中断
- EP1INT: 禁用端点 1 中断
- EP2INT: 禁用端点 2 中断
- EP3INT: 禁用端点 3 中断
- EP4INT: 禁用端点 4 中断
- EP5INT: 禁用端点 5 中断

0 = 无效。

1 = 禁用相应端点中断。

- RXSUSP: 禁用 UDP 挂起中断

0 = 无效。

1 = 禁用 UDP 挂起中断。

- RXRSM: 禁用 UDP 恢复中断

0 = 无效。

1 = 禁用 UDP 恢复中断。

- SOFINT: 禁用帧起始中断

0 = 无效。

1 = 禁用帧起始中断。

- WAKEUP: 禁用 USB 总线唤醒中断

0 = 无效。

1 = 禁用 USB 总线中断。

36.6.6 UDP 中断屏蔽寄存器

寄存器名称 :UDP_IMR

访问类型 :只读

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12[1]	11	10	9	8
–	–	WAKEUP	–	SOFINT	–	RXRSM	RXSUSP
7	6	5	4	3	2	1	0
–	–	EP5INT	EP4INT	EP3INT	EP2INT	EP1INT	EP0INT

注意：1.不允许屏蔽 UDP_IMR 的 Bit 12 并且总是读到 1

- EP0INT: 屏蔽端点 0 中断
- EP1INT: 屏蔽端点 1 中断
- EP2INT: 屏蔽端点 2 中断
- EP3INT: 屏蔽端点 3 中断
- EP4INT: 屏蔽端点 4 中断
- EP5INT: 屏蔽端点 5 中断

0 = 禁用相应端点中断。

1 = 使能相应端点中断。

- RXSUSP: 屏蔽 UDP 挂起中断

0 = 禁用 UDP 挂起中断。

1 = 使能 UDP 挂起中断。

- RXRSM: 屏蔽 UDP 恢复中断

0 = 禁用 UDP 恢复中断。

1 = 使能 UDP 恢复中断。

- SOFINT: 屏蔽帧起始中断

0 = 禁用帧起始中断。

1 = 使能帧起始中断。

- WAKEUP: USB 总线唤醒中断

0 = 禁用 USB 总线唤醒中断。

1 = 使能 USB 总线唤醒中断。

注意: 当 USB 块处于挂起模式时，应用程序可能关闭 USB 逻辑。此时，所有 USB 主机恢复请求必须记录，因此，UDP_IMR 寄存器的 RXRSM 位复位值必须使能。

36.6.7 UDP 中断状态寄存器

寄存器名称 :UDP_ISR

访问类型 :只读

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	WAKEUP	ENDBUSRES	SOFINT	–	RXRSM	RXSUSP
7	6	5	4	3	2	1	0
–	–	EP5INT	EP4INT	EP3INT	EP2INT	EP1INT	EP0INT

- EP0INT: 中断 0 中断状态
- EP1INT: 中断 1 中断状态
- EP2INT: 中断 2 中断状态
- EP3INT: 中断 3 中断状态
- EP4INT: 中断 4 中断状态
- EP5INT: 中断 5 中断状态

0=无端点 0 中断挂起

1 = 端点 0 中断已发生。

几个信号可产生该中断，通过读 UDP_ CSR0:可得：

RXSETUP 置为 1

RX_DATA_BK0 置为 1

RX_DATA_BK1 置为 1

TXCOMP 置为 1

STALLSENT 置为 1

EP0INT 为 粘着位。写相应 UDP_ CSR0 位对 EP0INT 清零前中断有效。

- RXSUSP: UDP 挂起中断状态

0 = 无 UDP 挂起中断等待。

1 = UDP 挂起中断已发生。

当 USB 器件检测到 3ms 无工作时设置该位。 USB 器件进入挂起模式。

- RXRSM: UDP 恢复中断状态

0 = 无 UDP 恢复中断等待。

1 =UDP 恢复中断已发生。

当 USB 器件端口检测到 USB 恢复信号时设置该位。

复位后，该位的状态未定义，应用程序必须清零该位，可通过设置 UDP_ ICR 寄存器中的 RXRSM 标志位。

- SOFINT: 帧开始中断状态

0 = 无帧开始中断等待。

1 = 帧开始中断已发生。

每次检测到 SOF 时发生中断。它可被同步端点作为同步信号使用。

- ENDBUSRES: 总线结束复位中断状态

0 = 无总线结束复位中断等待。

1 = 总线结束复位中断已发生。

UDP 复位序列结束后发生中断。 USB 器件必须在端点 0 准备接收请求。主机开始列举，

然后开始进行配置。

- WAKEUP: UDP 恢复中断状态

0 = 无唤醒中断等待。

1 = 上次清零后出现唤醒中断 (USB 主机发送 RESUME 或 RESET)。

复位后，该位的状态未定义，应用程序必须清零该位，可通过设置 UDP_ ICR 寄存器中的 WAKEUP 标志位。

36.6.8 UDP 中断清除寄存器

寄存器名称 :UDP_ICR

访问类型 :只写

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

- RXSUSP: 清除 UDP 挂起中断

0 = 无效。

1 = 清除 UDP 挂起中断。

- RXRSM: 清除 UDP 恢复中断

0 = 无效。

1 = 清除 UDP 恢复中断。

- SOFINT: 清除帧开始中断

0 = 无效。

1 = 清除帧开始中断。

- ENDBURST: 清除总线复位结束中断

0 = 无效。

1 = 清除总线复位结束中断。

- WAKEUP: 清除唤醒中断

0 = 无效。

1 = 清除唤醒中断。

36.6.9 UDP 复位端点寄存器

寄存器名称 :UDP_RST_EP

访问类型 :读 / 写

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	EP5	EP4	EP3	EP2	EP1	EP0

- EP0: 复位端点 0
- EP1: 复位端点 1
- EP2: 复位端点 2
- EP3: 复位端点 3
- EP4: 复位端点 4
- EP5: 复位端点 5

该标志用于复位与端点相关的 FIFO 及 UDP_CSRx 寄存器中的 RXBYTECOUNT 位。它还复位数据切换到 DATA0。它在删除 BULK 端点 HALT 状态后有用。参见 USB Serial Bus Specification, Rev.2.0. 的 5.8.5 节。

警告：复位结束时该标志必须清除。不清除 UDP_CSRx 标志。

0 = 无复位。

1 = 强制将相应端点 FIF0 指向 0，因此 UDP_CSRx 寄存器 RXBYTECNT 域为 0。

36.6.10 UDP 端点控制与状态寄存器

寄存器名称 :UDP_CSRx [x = 0. 5]

访问类型 :读 / 写

31	30	29	28	27	26	25	24
–	–	–	–	–	RXBYTECNT		
23	22	21	20	19	18	17	16
RXBYTECNT							
15	14	13	12	11	10	9	8
EPEDS	–	–	–	DTGLE	EPTYPE		
7	6	5	4	3	2	1	0
DIR	RX_DATA_ BK1	FORCE STALL	TXPKTRDY	STALLSENT ISOERROR	RXSETUP	RX_DATA_ BK0	TXCOMP

警告：由于 MCK 和 UDPCK 之间的同步，通过轮询必须被置位/清零的位，应用软件必须在执行另一次写之前等待写操作结束。

```
//! Clear flags of UDP UDP_CSR register and waits for synchronization
#define Udp_ep_clr_flag(pInterface, endpoint, flags) { \
while (pInterface->UDP_CSR[endpoint] & (flags)) \
pInterface->UDP_CSR[endpoint] &= ~(flags); \
}
//! Set flags of UDP UDP_CSR register and waits for synchronization
#define Udp_ep_set_flag(pInterface, endpoint, flags) { \
while ( (pInterface->UDP_CSR[endpoint] & (flags)) != (flags) ) \
pInterface->UDP_CSR[endpoint] |= (flags); \
}
```

- TXCOMP: 产生一个有前数据写入 DPR 的入包

该标志设置为 1 时产生中断。

写 (由固件清零) ;

0 = 清除标志，清除中断。

1 = 无效。

读 (由 USB 外设设置):

0 = 主机未应答数据入处理。

1 = 数据入处理完成，主机应答。

数据入处理设置 TXPKTRDY 后，器件固件等待 TXCOMP 以确保主机已应答处理。

- RX_DATA_BK0: 接收数据段 0

该标志设置为 1 时产生中断。

写 (由固件清零) ;

0 = 通知 USB 外设器件， FIFO 段 0 数据已被读取。

1 = 无效。

读 (由 USB 外设设置):

0 = FIFO 段 0 未收到数据包。

1 = 已收到数据包，并已存入 FIFO 段 0。

当器件固件已轮询该位或由该信号中断，必须将数据由 FIFO 送到微控制器存储器。RXBYTCENT 域中收到的字节数有效。通过 UDP_FDRx 寄存器读 FIFO 段 0 值。一旦

发送完成，器件固件必须通过清除 RX_DATA_BK0 将 USB 外设器件段 0 释放。

- RXSETUP: 接收设置

该标志设置为 1 时产生中断。

读：

0 = 无有效设置包。

1 = 主机已发送设置包并在 FIFO 中有效。

写：

0 = 器件固件通知 USB 外设器件，它已读取 FIFO 中设置数据。

1 = 无效。

该标志用来通知 USB 器件固件，主机已收到有效设置数据包并由 USB 器件成功接收。USB 器件固件可通过读 UDP_FDRx 寄存器将 FIFO 中设置数据发送到微控制器处理器。一旦发送完成， RXSETUP 必须由器件固件清除。

当 RXSETUP 置位时，不接收随后的数据出数据。

- STALLSENT: 发送停止 (控制、批中断端点) / ISOERROR (同步端点)

该标志设置为 1 时产生中断。

STALLSENT：结束停止握手。

读：

0 = 主机未应答 STALL。

1 = 主机应答 STALL。

写：

0 = 复位 STALLSENT 标志，清除中断。

1 = 无效。

强制器件固件清除该标志，否则中断保持。

关于 STALL 握手，参见通用串行总线规范，Rev. 2.0 8.4.5 及 9.4.5 节。

ISOERROR：同步发送中检测到 CRC 错误。

读：

0 = 之前的同步传输中无错误。

1 = 检测到 CRC 错误， FIFO 中可用数据被破坏。

写：

0 = ISOERROR 标志复位，清除中断。

1 = 无效。

- TXPKTRDY: 发送包就绪

该标志由 USB 器件清除。

该标志由 USB 器件固件置位。

读：

0 = 数据值可写入 FIFO。

1 = 数据值不能写入 FIFO。

写：

0 = 无效。

1 = 新数据有效负载通过固件写入 FIFO 并即将发送。

该标志用来产生数据入处理 (器件到主机)。器件固件检查它能否在 FIFO 中写入数据有效负载，检查 TXPKTRDY 是否清零。通过写 UDP_FDRx 寄存器实现数据到 FIFO 的发送。一旦数据有效否则发送到 FIFO，固件通知 USB 器件将 TXPKTRDY 设置为 1 。USB 总线处理开始。一旦主机收到数据有效负载， TXCOMP 置位。

- FORCESTALL: 强制停止 (用于控制、批及同步端点)

读：
0=普通状态
1=停止状态
写：
0 = 返回到普通状态。
1 = 向主机发送 STALL。
关于 STALL 握手，参见通用串行总线规范，Rev. 2.0 8.4.5 及 9.4.5 节。
控制端点：数据筹备与状态筹备过程中，表明微控制器不能完成请求。
批与中断端点：通知主机端点已停止。
主机应答 STALL， STALLSENT 标志通知器件固件。

- RX_DATA_BK1: 接收数据段 1 (只用于有 ping-pong 特性的端点)

该标志设置为 1 时产生中断。
写 (由固件清零) ；
0 = 通知 USB 器件， FIFO 段 1 数据已被读取。
1 = 无效。
读 (由 USB 外设设置)：
0 = FIFO 段 1 未收到数据包。
1 = 已收到数据包，并已存入 FIFO 段 1。
当器件固件已轮询该位或由该信号中断，必须将数据由 FIFO 送到微控制器存储器。RXBYTCENT 域中收到的字节数有效。通过 UDP_FDRx 寄存器读 FIFO 段 1 值。一旦发送完成，器件固件必须通过清除 RX_DATA_BK1 将 USB 外设器件段 1 释放。

- DIR: 发送方向 (仅对控制端点有效)

读 / 写
0 = 在控制数据筹备时允许数据出处理。
1 = 控制数据筹备时使能数据入处理。
关于控制数据筹备，参见通用串行总线规范，Rev. 2.0 8.5.3 节。
在设置筹备结束清除 UDP_CSRx/RXSETUP 前该位必须置位。根据设置数据包发送请求，数据筹备为器件到主机 (DIR = 1)或主机到器件 (DIR = 0) 数据发送。状态筹备时不需检验该位反向。

- EPTYPE[2:0]: 端点类型

读写

000	控制
001	同步 OUT
101	同步 IN
010	批 OUT
110	批 IN
011	中断 OUT
111	中断 IN

- DTGLE: 数据切换

只读
0 = 识别 DATA0 包。
1 = 识别 DATA1 包。

更多 DATA0、 DATA1 包定义见通用串行总线规范，Rev. 2.0 第八章。

- EPEDS: 端点使能禁用

读：

0 = 端点禁用。

1 = 端点使能。

写：

0 = 禁用端点。

1 = 使能端点。

- RXBYTECNT[10:0]: FIFO 中可用字节数

只读

当主机向器件发送数据包时， USB 器件将数据存入 FIFO 并通知微控制器。微控制器可通过读取 UDP_FDRx 寄存器的 RXBYTECENT 字节载入 FIFO 中数据。

36.6.11 UDP FIFO 数据寄存器

寄存器名称：UDP_ FDRx [x = 0..5]

访问类型：读写

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

- FIFO_DATA[7:0]: FIFO 数据值

微控制器可通过该寄存器将数据推入或弹出 FIFO。

相应 UDP_CSRx 寄存器的 RXBYTECNT 中是由 FIFO 中读取的字节数（主机发送）。能写入的最大字节数由标准端点描述符的最大包尺寸确定。它不能大于相关端点物理存储器大小，详见通用串行总线规范，Rev. 2.0 。

36.6.12 UDP 收发器控制寄存器

寄存器名称： UDP_ TXVC

访问类型：读写

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

警告：必须在向 UDP 寄存器包括 UDP_TXCV 寄存器的任何读写操作前使能电源管理控制器（PMC）中的 UDP 外设时钟。

- TXVDIS: 禁用收发器

当禁用 UDP，可通过禁用内置的收发器来有效的降低功耗。可通过设置 TXVDIS 域实现。

注意：如果 USB 上拉未连接到 DP，用户不能向任何 UDP 寄存器写入，除了 UDP_ TXVC 寄存器。

- PUON：上拉

0：断开集成在 DP 上的 1.5 千欧上拉电阻

1：连接集成在 DP 上的 1.5 千欧上拉电阻

37 LCD 控制器（LCDC）

37.1 概述

LCD 控制器由从外部显示缓存(以下简称为显存)到集成有 common 和 segment 驱动的 LCD 组件传输数据的逻辑构成。

LCD 控制器支持单，双扫描的单色或彩色被动 STN LCD 模组和单扫描的主动 TFT LCD 模组。对于单色 STN 显示，通过时基抖动算法和帧速率控制（FRC）方法支持多达 16 个灰度级。该方法也被用于彩色 STN 显示，用来产生 4096 色彩。

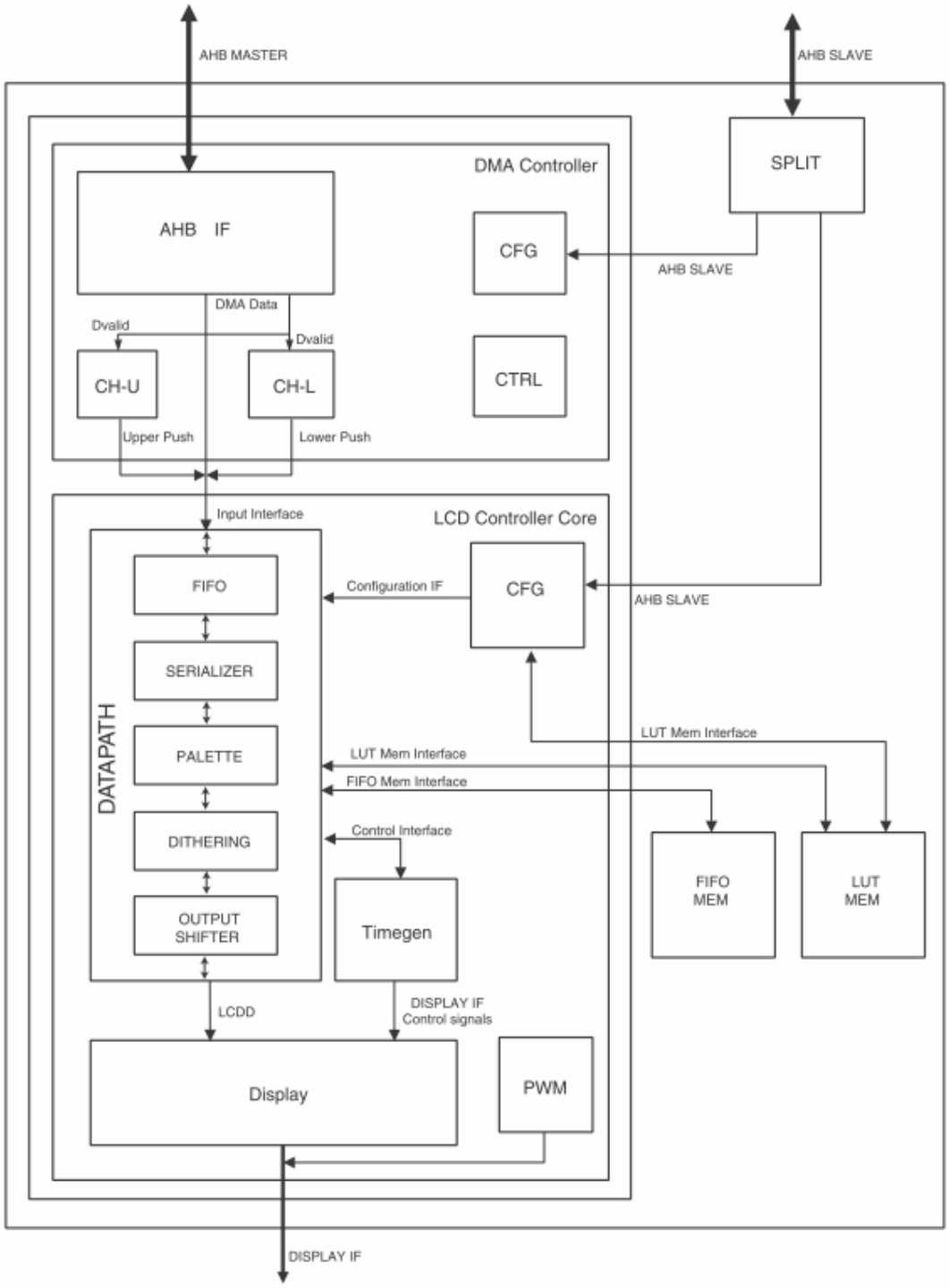
LCD 控制器的输入 FIFO 允许与外部 AHB 主设备接口进行灵活连接，其调色板查询表允显示配置为调色板模式。

可编程 LCD 控制器以实现各种不同的需求，如分辨率可达 2048*2048；像素色深（1，2，4，8，16，24 位每像素）；数据线宽度（4，8，16 或 24 位）和接口时序。

LCD 控制器作为连接到 ARM 先进高性能总线（AHB）的主控读取像素数据。同时，LCD 控制器连接 AHB 的接口在配置其寄存器时作为从机。

37.2 框图

图 37-1 LCD 宏单元框图



37.3 I/O 口说明

表 37-1 I/O 口说明

名称	说明	类型
LCDCC	对比度控制信号	输出
LCDHSYNC	行同步信号（STN）或水平同步信号（TFT）	输出
LCDDOTCK	LCD 时钟信号（STN/TFT）	输出
LCDVSYNC	帧同步信号（STN）或垂直同步信号（TFT）	输出
LCDDEN	STN 驱动 AC 偏置信号或数据使能信号（TFT）	输出

LCDD[23 ： 0]	LCD 数据总线输出	输出

37.4 产品相关属性

37.4.1I/O 口

用于连接 LCD 控制器的引脚可以和 PIO 口复用。编程者必须首先编程 PIO 控制器来指定这些引脚的外设功能。如果应用程序未使用 LCD 控制器的 I/O 口，则这些 IO 口可被 PIO 控制器用于其他目的。

37.4.2 电源管理

LCD 控制器不是一直在工作。由于 LCD 控制器连在 AHB 总线上，其时钟是通过设置 PMC_SCER 寄存器中的 HCKx 位使能的。

37.4.3 中断源

LCD 控制器中断口连接于高级中断控制器（AIC）的内部中断源之一。使用 LCD 控制器中断需要首先编程 AIC。

37.5 功能说明

LCD 控制器由两部分组成（图 37-1），DMA 控制器和 LCD 控制器内核（LCDC 核）。DMA 控制器通过 AHB 主控接口从外部存储器读取显示数据。LCD 控制器内核规范化这些显示数据。LCD 控制器内核通过 LCD 数据总线（LCDD[23：0]）连续的把像素数据输送到 LCD 模组；此总线由 LCDDOTCK，LCDDEN，LCDHSYNC 和 LCDVSYNC 信号同步。

37.5.1DMA 控制器

37.5.1.1 配置模块

配置模块是一组用于配置 DMA 控制器工作的可编程寄存器。通过 AHB 从机接口写这些寄存器。仅允许按字访问。

配置寄存器的更多细节，见“LCD 控制器（LCDC）用户接口”

37.5.1.2 AHB 接口

该模块产生 AHB 事务。产生未定义长度自增的脉冲以及 4-，8-或 16-拍自增脉冲。传输长度可在 DMAFRMCFG 寄存器中的 BRSTLN 域配置。该寄存器详情见“DMA 帧配置寄存器”

37.5.1.3 U 通道

该模块记录从帧起始时存储基地址计算的偏移地址和传输到该通道（单扫描模式的帧和双扫描模式的上部画面）的字数。同时产生帧结束信号。

它有两个指针，基地址和传输字数。当模组接收到一个新帧，将重新装载带帧/画面大小的传输字数指针。当模组接收到新帧信号，也重新装载由主机编程的基地址。

帧/画面的大小可通过 DMAFRMCFG 寄存器的 FRMSIZE 域编程。计算方法如下：

- TFT 模式：

$$\text{帧大小} = \frac{\text{Display_size} \times \text{Bpp}}{32}$$

- STN 单色模式：

$$\text{帧大小} = \left\lceil \frac{(\text{LINEVAL} + 1) \times (\text{HOZVAL} + 1) \times \text{E_ifwidth} \times \text{Bpp}}{32} \right\rceil$$

- STN 彩色模式：

$$\text{帧大小}=\frac{(\text{LINEVAL}+1)\left(\frac{(\text{HOZVAL}+1)\times \text{E_ifwidth}}{3}\right)\text{Bpp}}{32}$$

注：

- LINEVAL 是 LCD 控制器 LCDFRMCFG 寄存器的 LINEVAL 域
- HOZVAL 是 LCD 控制器 LCDFRMCFG 寄存器的 HOZVAL 域
- E_ifwidth 是 LCD 接口的数据位数
- Bpp 是每像素配置位。

37.5.1.4 L 通道

该模块有如 U 通道同样的功能，但仅用于双扫描模式的 lower 画面。

37.5.1.5 控制

该模块从 LCDC 内核接收请求信号并对通道发出请求。

37.5.2 LCD 控制器内核

37.5.2.1 配置块

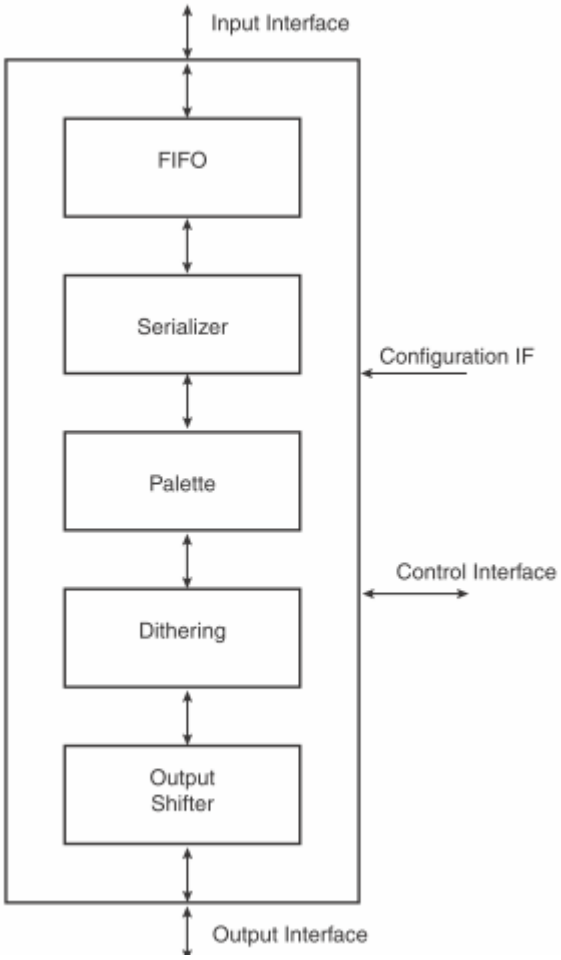
配置块是一组用于配置 LCDC 内核工作的可编程寄存器。可通过 AHB 从接口写这些寄存器。仅允许按字访问。

这些配置寄存器的说明见“LCD 控制器用户接口”

37.5.2.2 数据路径

数据路径方块包括五个子模块：FIFO，序列器，调色板，抖动和移位器。数据路径的结构见图 37-2

图 37-2 数据路径结构



该模块将从存储器读取的数据转换到 LCD 模组使用的格式。有四个不同的接口：输入接口，输出接口，配置接口和控制接口。

- 输入接口连接数据路径到 DMA 控制器。这是带一个数据总线和两个注入口的双 FIFO 接口，用于 DMA 控制器降数据填入 FIFO
- 输出接口是个 24 位数据总线。该接口的配置取决于使用的 LCD 类型（TFT 或 STN，单或双扫描，4 位，8 位，16 位或 24 位接口）
- 配置接口连接数据路径和配置块。用于选择不同数据路径的配置。
- 控制接口连接数据路径和时序发生块。主控制信号是数据请求信号，时序发生模块用来从数据路径请求新数据。

数据路径可由两个参数描述：initial_latency 和 cycles_per。initial_latency 参数定义为到数据路径输出第一个可用数据时 LCDC 内核时钟周期。cycles_per 参数是输出接口两个连续数据见 LCDC 内核时钟周期的最小数。

这些参数 LCD 控制器的不同配置的差异见表 37-2

表 37-2 数据路径参数

配置			initial_latency	cycles_per_data
DISTYPE	SCAN	IFWIDTH		
TFT			9	1
STN 单色	单	4	13	4
STN 单色	单	8	17	8
STN 单色	双	8	17	8

STN 单色	双	16	25	16
STN 彩色	单	4	11	2
STN 彩色	单	8	12	3
STN 彩色	双	8	14	4
STN 彩色	双	16	15	6

FIFO

FIFO 块缓冲从 DMA 模块读取的输入数据。包括用于双扫描配置的两个输入 FIFO，当用在单扫描配置时配置为单 FIFO。

FIFO 的大小允许支持宽范围的结构。

FIFO 的上限可在 LCDFIFO 寄存器的 FIFOTH 域配置。当每个 FIFO 中的字数少于 FIFOTH 字时 LCDC 内核将请求 DMA 传输。为避免覆盖 FIFO 并最大化 FIFO 的应用，FIFOTH 应该编程为 FIFOTH = 512 words - (2 x DMA_BURST_LENGTH + 3)

其中：

- 512 字是 FIFO 的有效字长。是单扫描模式的全 FIFO 存储长度，双扫描模式的半长。
- DMA_burst_length 是由 DMA 生成的传输脉冲长度

序列器

该方块串行化从存储器读取的数据。它从 FIFO 读取字并且依照 LCDCON2 寄存器的 PIXELSIZE 域中规范的格式输出像素（1 位，2 位，4 位，8 位，16 位或 24 位宽）。也适应存储器定制格式。大端小端格式都支持。在 LCDCON2 寄存器的 MEMOR 域配置。

存储器中的像素数据的组织取决于配置。见表 37-3 和表 37-4

表 37-3 小端存储结构

Mem Addr	0x3								0x2								0x1								0x0							
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Pixel 1bpp	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Pixel 2bpp	15		14		13		12		11		10		9		8		7		6		5		4		3		2		1		0	
Pixel 4bpp	7				6				5				4				3				2				1				0			
Pixel 8bpp	3								2								1								0							
Pixel 16bpp	1																0															
Pixel 24bpp	1								0																							
Pixel 24bpp	2																1															
Pixel 24bpp	3																								2							
Pixel 24bpp	5								4																							

表 37-4 大端存储结构

Mem Addr	0x3								0x2								0x1								0x0							
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Pixel 1bpp	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Pixel 2bpp	0		1		2		3		4		5		6		7		8		9		10		11		12		13		14		15	
Pixel 4bpp	0				1				2				3				4				5				6				7			
Pixel 8bpp	0								1								2								3							
Pixel 16bpp	0																1															
Pixel 24bpp	0																								1							
Pixel 24bpp	1																2															
Pixel 24bpp	2								3																							
Pixel 24bpp	4																								5							

调色板

此块用于产生像素灰度或调色配置中的彩色数据。palletized/non-palletized 配置的不同模式见表 37-5。这些模式中，1，2，4 或 8 输入位索引到查询表的入口。查询表对应的入口包括像素的彩色或灰度信息

表 37-5 调色配置

配置		调色
DISTYPE	PIXELSIZE	
TFT	1，2，4，8	Palletized
TFT	16，24	Non-palletized
STN 单色	1，2	Palletized
STN 单色	4	Non-palletized
STN 彩色	1，2，4，8	Palletized
STN 彩色	16	Non-palletized

主机可以 R/W 模式访问查询表，所以允许主机编程和检查存储在调色板中的值。映射到 LCD 控制器配置存储器。LUT 映射为以字边界对齐的 16 位半字，仅允许字写入访问（总线的 16MSB 未用）。更多存储器映射的细节见表 37-12

查询表包括 256 个 16 位宽的入口。编程者从 2 的 16 次方个组合中选择 256 个入口。

每个 LUT 入口的结构 见表 37-6

表 37-6 存储器中的查询表结构

地址	数据输出[15：0]			
00	Intensity_bit_0	Blue_value_0[4:0]	Green_value_0[4:0]	Red_value_0[4:0]
01	Intensity_bit_2	Blue_value_1[4:0]	Green_value_1[4:0]	Red_value_1[4:0]
...				
FE	Intensity_bit_254	Blue_value_254[4:0]	Green_value_254[4:0]	Red_value_254[4:0]
FF	Intensity_bit_255	Blue_value_255[4:0]	Green_value_255[4:0]	Red_value_255[4:0]

在 STN 单色模式，仅使用红色值的四个最高位（16 个灰度）。在 STN 彩色模式，仅使用蓝

色，绿色和红色值的四个最高位（4096 色）。
在 TFT 模式，使用蓝色，绿色和红色的所有位（32768 色）。该模式中，还可使用共有的一个亮度位来倍增可能的色彩。该位是 LCDD 接口(LCDD[18], LCDD[10],LCDD[2])中每种色彩元素的最低位。当使用 TFT 调色板配置(LCDD[17:16], LCDD[9:8], LCDD[1:0])时 LCDD 中未用位固定位 0。

抖动

当 LCD 控制器驱动 STN LCD 模组时，抖动模块用来产生灰度或彩色度。用时基抖动算法和帧速率控制方法。
帧速率控制改变设定的打开状态的像素的占空比，给出多级色度性能的显示。为了减少由相邻像素的同时开关导致的闪动噪声，用时基抖动算法改变每帧相邻像素的类型。该算法用抖动类型寄存器（DP_i)）表示。不仅仅考虑像素的灰度级数，还考虑其水平坐标。
表 37-7 抖动占空比

灰度级	占空比	图形寄存器
15	1	-
14	6/7	DP6_7
13	4/5	DP4_5
12	3/4	DP3_4
11	5/7	DP5_7
10	2/3	DP2_3
9	3/5	DP3_5
8	4/7	DP4_7
7	1/2	-DP1_2
6	3/7	-DP4_7
5	2/5	-DP3_5
4	1/3	-DP2_3
3	1/4	-DP3_4
2	1/5	-DP4_5
1	1/7	-DP6_7
0	0	-

灰度 0 和灰度 15 的占空比分别是 0 和 1。
其占空比和为 1（如，1/7 和 6/7）的相同的 DP_i 寄存器可成对使用。第一对元素的抖动图形是第二对元素的抖动图形的反相。

DP_i 寄存器包括一系列的 4 位类型。类型的$(3-m)^{th}$位决定当前帧中水平坐标 x=4n+m（n 为整数，m 为 0 到 3 的数）的像素是否应该打开或关闭。此操作见下例。
分别考虑水平坐标位为 4*n+0, 4*n+1, 4*n+2 和 4*n+3 的 a,b,c 和 d 像素。四个像素应该以 9 级灰度显示所以应使用寄存器 DP3_5=”1010 0101 1010 0101 1111”。
单色模式数据输出顺序见表 37-8。
表 37-8 单色模式抖动算法

帧序	类型	像素 a	像素 b	像素 c	像素 d
N	1010	ON	OFF	ON	OFF
N+1	0101	OFF	ON	OFF	ON

N+2	1010	ON	OFF	ON	OFF
N+3	0101	OFF	ON	OFF	ON
N+4	1111	ON	ON	ON	ON
N+5	1010	ON	OFF	ON	OFF
N+6	0101	OFF	ON	OFF	ON
N+7	1010	ON	OFF	ON	OFF
...					

考虑当前彩色显示模式和水平坐标为 4*n+0, 和 4*n+1 的两个像素 p0 和 p1。一个彩色像素由三个元素构成：{R,G,B}。发送彩色元素{R0, G0, B0} 到显示器将显示像素 p0。发送彩色元素{R1, G1, B1} 到显示器将显示像素 p1。假设从存储器读取的和映射到查询表的数据对应两个像素的三色的灰度 10，则抖动图形将所有数据应用位 DP2_3 = “1101 1011 0110”。表 37-9 表示了单扫描配置数据输出总线的输出顺序。（双扫描配置中，每个画面数据总线作用和三扫面配置等效。）

表 37-9 彩色模式抖动算法

帧	信号	灰度	使用位	抖动	4 位 LCDD	8 位 LCDD	输出
N	red_data_0	1010	3	1101	LCDD[3]	LCDD[7]	R0
N	green_data_0	1010	2	1101	LCDD[2]	LCDD[6]	G0
N	blue_data_0	1010	1	1101	LCDD1]	LCDD[5]	b0
N	red_data_1	1010	0	1101	LCDD[0]	LCDD[4]	R1
N	green_data_1	1010	3	1101	LCDD[3]	LCDD[3]	G1
N	blue_data_1	1010	2	1101	LCDD[2]	LCDD[2]	B1
...							
N+1	red_data_0	1010	3	1011	LCDD[3]	LCDD[7]	R0
N+1	green_data_0	1010	2	1011	LCDD[2]	LCDD[6]	g0
N+1	blue_data_0	1010	1	1011	LCDD[1]	LCDD[5]	B0
N+1	red_data_1	1010	0	1011	LCDD[0]	LCDD[4]	R1
N+1	green_data_1	1010	3	1011	LCDD[3]	LCDD[3]	G1
N+1	blue_data_1	1010	2	1011	LCDD[2]	LCDD[2]	b1
...							
N+2	red_data_0	10103	3	0110	LCDD[3]	LCDD[7]	r0
N+2	green_data_0	1010	2	0110	LCDD[2]	LCDD[6]	G0
N+2	blue_data_0	1010	1	0110	LCDD[1]	LCDD[5]	B0
N+2	red_data_1	1010	0	0110	LCDD[0]	LCDD[4]	r1
N+2	green_data_1	1010	3	0110	LCDD[3]	LCDD[3]	g1
N+2	blue_data_1	1010	2	0110	LCDD[2]	LCDD[2]	B1
...							

注意：Ri=红色像素单元 ON，Gi=绿色像素单元 ON，Bi=蓝色像素单元 ON。Ri=红色像素单元 OFF，gi=绿色像素单元 OFF，bi=蓝色像素单元 OFF。

移位寄存器

FIFO，串行转换器，调色板和抖动模块在单色模式下每次处理一个像素，在彩色(R,G,B 单元)模式下每次处理三个子像素。该模块根据输出接口打包数据。可用 LDCCON3

寄存器的 DISTYPE，SCANMOD，和 IFWIDTH 域编程输出接口。

DISTYPE 域选择 TFT，STN 单色和 STN 彩色显示。SCANMODE 域选择单双扫描模式。（IFWIDTH 域配置 STN 模式下接口宽度：4 位（仅单色扫描模式），8 位和 16 位（仅双扫描模式）。

关于域的更多说明见“LCD 控制器用户接口”

关于 LCD 接口的更多说明见“LCD 接口”

37.5.2.3Timegen

LCD 模组用时序发生器模块产生控制信号 LCDDOTCK, LCDHSYNC, LCDVSYNC,LCDDEN。该模块可编程，支持不同类型的 LCD 模组，并获得来自 LCDC 内核时钟的输出时钟信号。

用 LCDDOTCK 信号协调数据进入 LCD 驱动移位寄存器。默认 LCDDOTCK 下降沿（可选上升沿）同步的 LCDD[23：0]发送数据。LCDCON1 寄存器的 CLKVAL 域控制该信号速率。LCDCON1 寄存器中的 BYPASS 位可绕开分频器。该情况下，LCDDOTCK 的速率等于 LCDC 内核时钟频率。LCDDOTCK 信号的最小周期取决于配置。见表 37-10。

$$f_{LCDDOTCK} = \frac{f_{LCDC_clock}}{2 \times CLKVAL}$$

可通过 LCDCON2 寄存器的 CLKMOD 域选择 LCDDOTCK 信号的两种不同的时序；

- 总是有效（使用 TFT LCD 模组）
- 仅当数据可用时有效（STN LCD 模组）

表 37-10 LCDC 内核时钟周期下最小的 LCDDOTCK 周期

配置			LCDDOTCK 周期
DISTYPE	SCAN	IFWIDTH	
TFT			1
STN 单色	单	4	4
STN 单色	单	8	8
STN 单色	双	8	8
STN 单色	双	16	16
STN 彩色	单	4	2
STN 彩色	单	8	2
STN 彩色	双	8	4
STN 彩色	双	16	6

LCDDEN 信号指示 LCD 接口上的有效数据。

在每次水平扫描数据被移位到 LCD 后，LCDHSYNC 有效并使接口的数据显示到面板上。

以下时序参数可配置：

- 垂直到水平延迟(VHDLY)： begin_of_line 和 LCDHSYNC 的产生之间的延迟可在 LCDTIM1 寄存器中的 VHDLY 域配置。延迟等于(VHDLY+1) LCDDOTCK 周期。
- 水平脉冲宽度（HPW）：在 LCDTIM2 寄存器中的 HPW 域配置 LCDHSYNC 脉冲宽度。宽度等于(HPW + 1) LCDDOTCK 周期
- 水平后边沿（HBP）：LCDHSYNC 下降沿和 LCD 接口有效数据的第一个 LCDDOTCK 的上升沿之间的延迟在 LCDTIM2 寄存器中的 HBP 中配置。该延迟等于(HBP+1) LCDDOTCK 个周期。

- 水平前沿（HFP）：有效数据结尾和行结尾之间的延迟在 LCDTIM2 寄存器中的 HFP 域中配置。该延迟等于 (HFP+1)LCDDOTCK 个周期。

数据路径的初始等待时间对参数 VHKLY，HPW 和 HBP 的最小值有限制。LCDC 时钟周期中的总延迟必须大于等于表 37-2 中的等待时间。以下公式给出该限制：

等式一：(VHDLY + HPW + HBP + 3) × PCLK_PERIOD ≥DPATH_LATENCY

其中：

- VHDLY, HPW, HBP 是 LCDTIM1 和 LCDTIM2 寄存器中的域值。
- PCLK_PERIOD 是 LCDC 时钟周期中测量的 LCDDOTCK 信号周期
- DPATH_LATENCY 是表 37-2 中给出的配置的数据路径延迟

LCDVSYNC 每帧被激活一次。激活该信号导致回扫。该信号的时序取决于 LCD 的类型：STN 或 TFT LCD。

在 STN 模式，高相位对应帧的完整的第一行。在 STN 模式，该信号和一行中第一个有效的 LCDDOTCK 上升沿同步。

在 TFT 模式，该信号的高相位在第一行的起始位置开始。可选以下时序参数：

- 垂直脉冲宽度(VPW)：LCDTIM1 寄存器中的 VPW 域配置 LCDVSYNC 脉冲宽度。该脉冲宽带等于(VPW+1) 行。
- 垂直后边沿：在 LCDTIM1 寄存器的 VBP 域中配置帧起始的当前行数。非当前行数等于 VBP。STN 模式该域应该编程为 0。
- 垂直前边沿：在 LCDTIM2 寄存器中的 VFP 域配置帧结束的无效行数。无效行数等于 VFP。STN 模式该域应该编程为 0。

该模块中有两个另外可配置的参数， LCDFRMCFG:中的 HOZVAL 和 LINEVAL 域：

- HOZVAL 配置每行中有效 LCDDOTCK 周期数。每行中的有效周期数等于(HOZVAL+1)个周期。该参数的最小值为 1.
- LINEVAL 配置每帧中有效行数。有效行数等于(LINEVAL+1)行。该参数的最小值为 1.

图 37-3 和图 37-4 和图 37-5 展示了 LCDDOTCK, LCDDEN, LCDHSYNC 和 LCDVSYNC 信号。

图 37-3 STN 画面时序，CLKMOD0

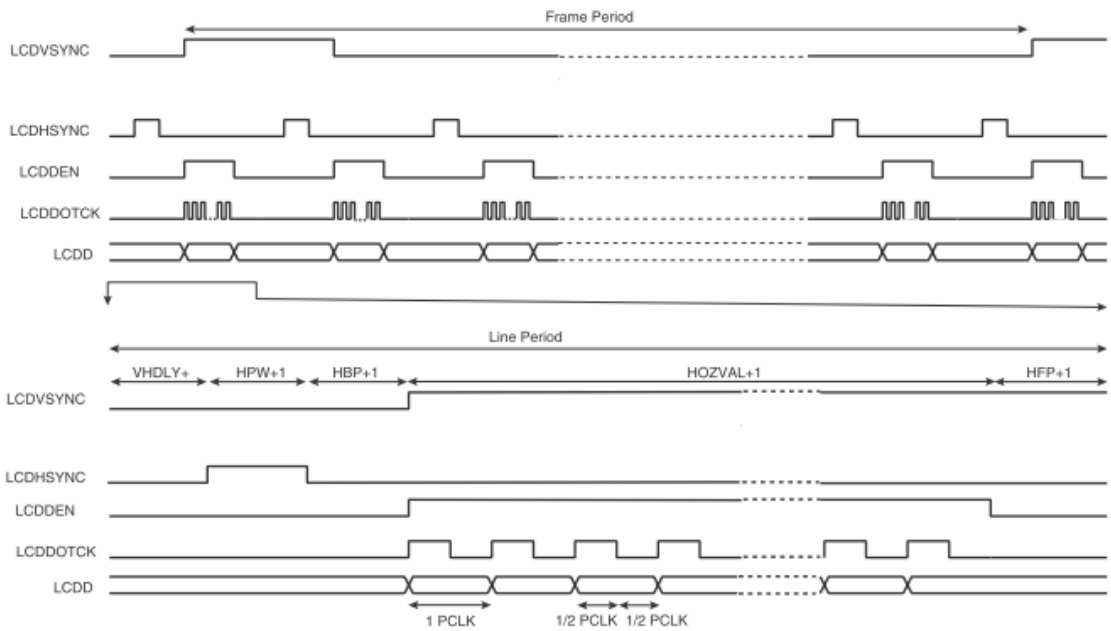


图 37-4　TFT 面板时序，CLKMOD=0，VPW=2，VFP=1

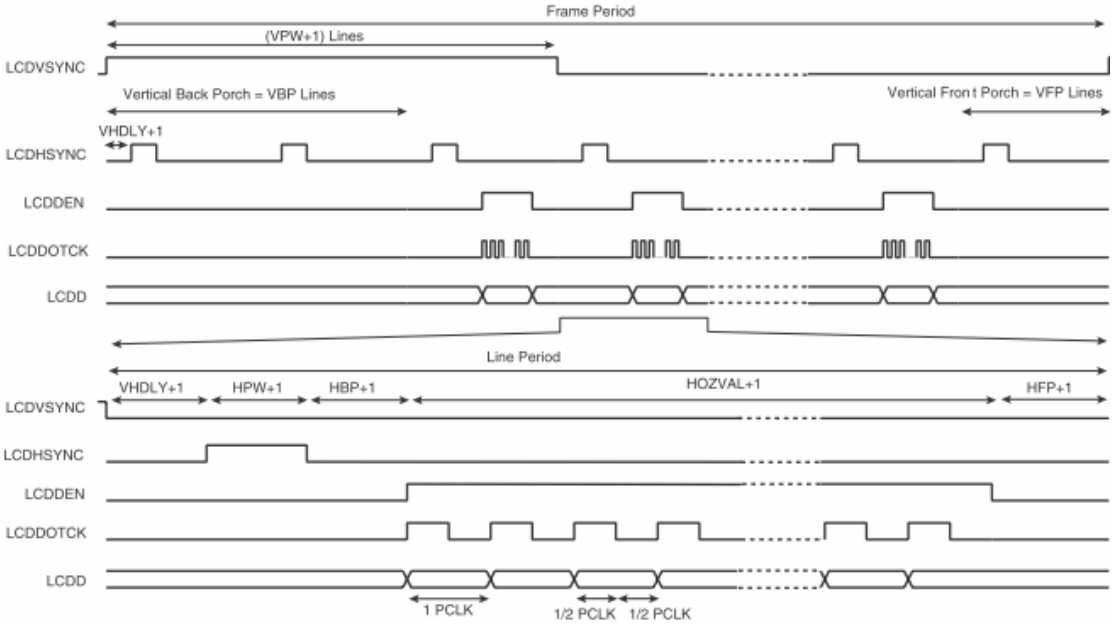
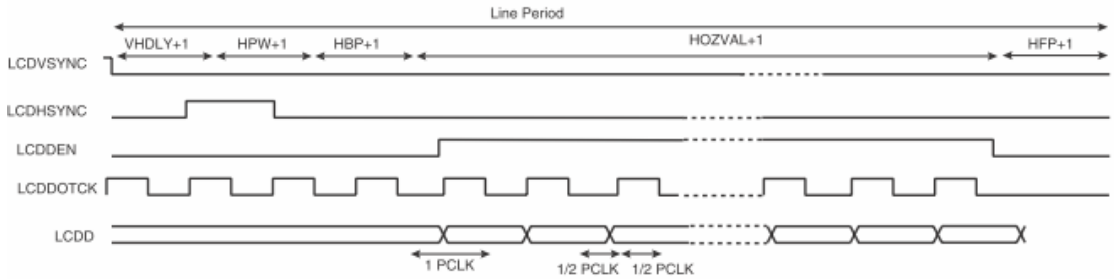


图 37-5 TFT 面板时序（行扩展图），CLKMOD=1



通常 LCD_FRM 速率大约 70 赫兹到 75 赫兹。由以下等式给出：

$$\frac{1}{f_{LCDVSYNC}} = \left(\frac{VHDLY + HPW + HBP + HOZVAL + HFP + 5}{f_{LCDDOTCK}}\right)(VBP + LINEVAL + VFP + 1)$$

其中：

- HOZVAL 决定每行的 LCDDOTCK 周期数
- LINEVAL 决定每帧的 LCDHSYNC 周期数。根据以下表达式：

STN 模式：

$$HOZVAL = \frac{Horizontal_display_size}{Number_data_lines} - 1$$

$$LINEVAL = Vertical_display_size - 1$$

单色模式，Horizontal_display_size 等于水平像素数。单扫描模式下 number_data_lines 等于接口的位数；双扫描模式下等于接口位数的一半

彩色模式下，Horizontal_display_size 等于水平像素数的三倍。

TFT 模式：

HOZVAL = Horizontal_display_size - 1

LINEVAL = Vertical_display_size - 1

不考虑加在起始结束或每行结束的时钟周期，通过帧速率方程决定，大概的 LCDDOTCK 速率：$f_{lcd_pclk} = (HOZVAL + 5) \times (f_{lcd_vsync} \times (LINEVAL + 1))$

有了此值则 CLKVAL 固定，对应的 LCDDOTCK 速率也确定。

然后根据所用的 LCD 类型和方程式 1 选择 VHDLY，HPW 和 HBP.

最后，帧速率被 HFP 值调整为 70 赫兹到 75 赫兹：

$$HFP = f_{LCDDOTCK} \times \left[\frac{1}{f_{LCDVSYNC} \times (LINEVAL + VBP + VFP + 1)}\right] - (VHDLY + VPW + VBP + HOZVAL + 5)$$

LCDCON1 寄存器中的只读域 LINECNT 控制行计数。LCDHYSNC 的每次下降沿时某装置递减 LINECNT 域。

37.5.2.4 显示

该模块用来配置数据和控制信号的极性。 LCDCON2[12:8] 寄存器设置可配置所有时钟信号的极性。该模块还内部产生 lcd_pwr 信号用于控制 LCD 引脚的状态，用于由软件打开或关闭 LCD 模块。

PWRCON 寄存器控制该信号。该信号考虑 LCD_PWR 域(PWRCON[0]) 写访问和 lcd_pwr 信号的激活禁用之间 PWRCON 寄存器的 GUARD_TIME 域(PWRCON[7:1])中配置的帧数。 GUARD_TIME 域的最小值是一帧。这给 DMA 控制器足够的时间在数据开始传输到 LCD 之前填满 FIFO。

37.5.2.5 PWM

该模块产生 LCD 对比度控制信号（LCDCC）使得显示对比度可由软件控制。这是一个 8 位 PWM（脉冲宽度调制）信号，可用一个简单的被动过滤器转换为模拟电压值。

PWM 模块有一个自由运行的计数器，计数器的值和对比寄存器（CONSTRAST_VAL 寄存器）对比。如果计数器中的值小于寄存器中的，则输出将极性位（POL）的值带到 PWM 控制寄存器：CONTRAST_CTR.。否则，输出相反的值。因此，产生一个和比较寄存器中的值成比例的脉冲宽度的周期波形。

由于比较机理，输出脉冲宽度为 0 到 255PWM 计数器周期。因此通过芯片外增加一个简单的无源过滤器，获得一个 0 和(255/256) × VDD 之间的模拟电压（对正极性情况，负极性时(1/256) × VDD 和 VDD 之间）。其他电压值可通过增加外部有源电路获得。

PWM 模式，可用 CONTRAST_CTR 寄存器的 PS 域调整计数器频率为四个不同的值。

37.5.3 LCD 接口

LCD 控制器通过 LCD 接口（表 37-11）和 LCD 模块连接。控制器支持以下接口配置：24 位 TFT 单扫描，16 位 STN 双扫描单色（彩色），8 位 STN 双（单）扫描单色（彩色），4 位单扫描单色（彩色）。

4 位单扫描 STN 显示使用 4 个并行数据口转移数据以每次连续的单水平扫描一次直到整个帧已被转移并传输完毕。LCD 数据总线的 4LSB 引脚(LCDD [3:0]) 可直接连接到 LCD 驱动器；未使用 20MSB 引脚(LCDD [23:4]) 。

8 位单扫描 STN 显示使用 8 个并行数据口转移数据以每次连续的单水平扫描一次直到整个帧已被转移并传输完毕。LCD 数据总线的 8 个 LSB 引脚(LCDD [7:0]) 可直接连接到 LCD 驱动器；未使用 16MSB 引脚(LCDD [23:8])。

8 位双扫描 STN 显示使用 2 组 4 个并行数据口转移数据使每次连续的上下画面水平扫描一次直到整个帧已完成转移和传输。总线 LCDD[3:0] 连接到上部面板数据口，总线 LCDD[7:4] 连接到下部面板数据口。LCD 数据总线其余的接口(LCDD[23:8]) 未被使用。

16 位双扫描 STN 显示使用两组 8 个并行数据口传输数据使得上部和下部面板水平扫描连续进行，直到整帧传输完毕。总线 LCDD[7:0] 连接到上面板数据接口，LCDD[15:8] 连接到下面板数据接口。其余的 LCD 数据总线接口(LCDD[23:16]) 未被使用。

STN 单色显示每像素需要一个图像数据位。STN 彩色显示每像素需要三个图像数据位（红绿蓝），导致每个水平接口的长度水平移位寄存器数是像素数的三倍。RGB 或单色数据通过并行数据接口按照顺序位转移到 LCD 驱动器。

TFT 单扫描显示使用多达 24 个并行数据接口转移数据使每次连续的水平扫描一次直至整帧已转移并传输完毕。24 个数据接口被分成三个字节来定义每像素的每个彩色单元的色深。LCDD 总线被分割，LCDD[23:16]被分为蓝色单元，LCDD[15:8] 为绿色单元，LCDD[7:0]是红色单元。如果 LCD 模组有更低的分辨率（每个彩色单元更少的位），仅使用每个单元的最高位。

所有这些接口如图 37-6 到图 37-10 所示。图 37-6 所展示的是 24 位单扫描 TFT 显示时序；图 37-7 所示为 4 位单扫描 STN 单色和彩色显示时序；图 37-8 所示为 8 位单扫描 STN 单色和彩色显示时序；图 37-9 所示位 8 位双扫描 STN 单色和彩色显示时序；图 37-10 所示为 16 位双扫描单色和彩色显示时序。

图 37-6 TFT 时序（首行扩展图）

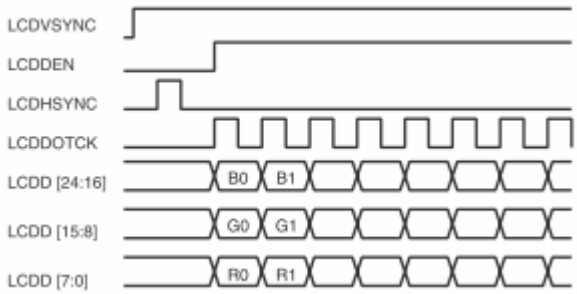


图 37-7 单色和彩色单扫描 4 位面板时序（首行扩展图）

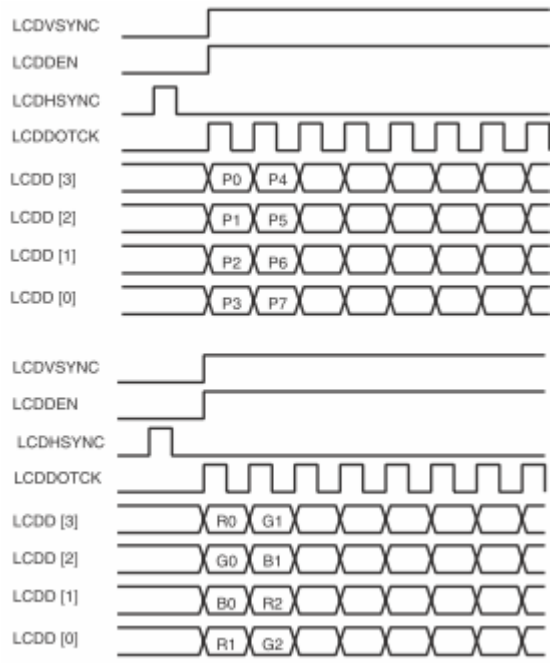


图 37-8　单色和彩色单扫描 8 位面板时序（首行扩展图）、

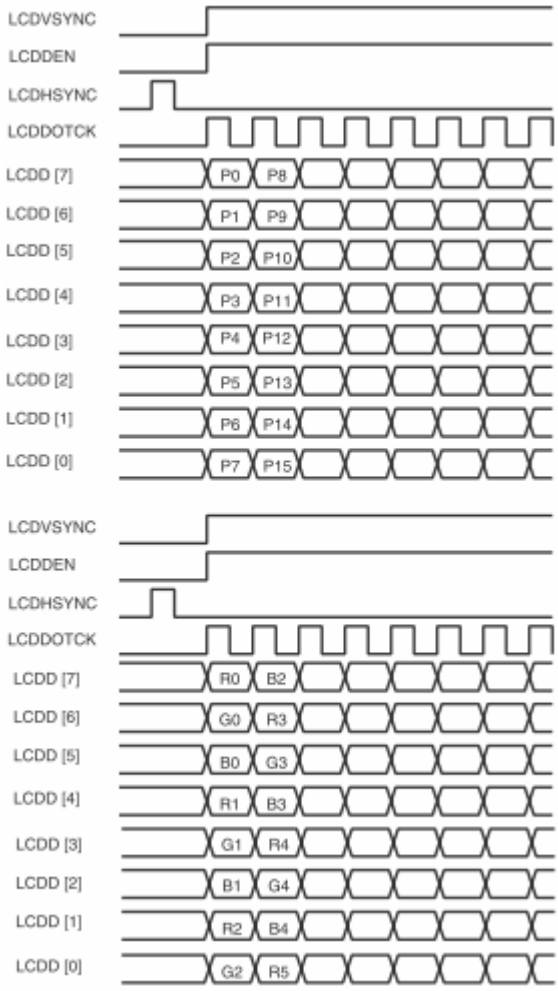


图 37-9 单色和彩色双扫描 8 位面板时序（首行扩展图）

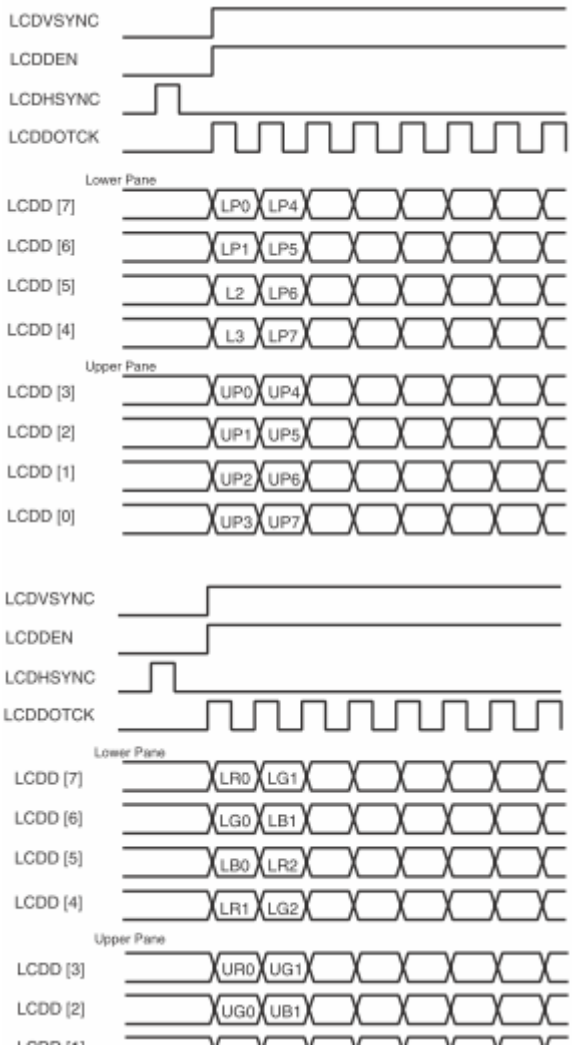
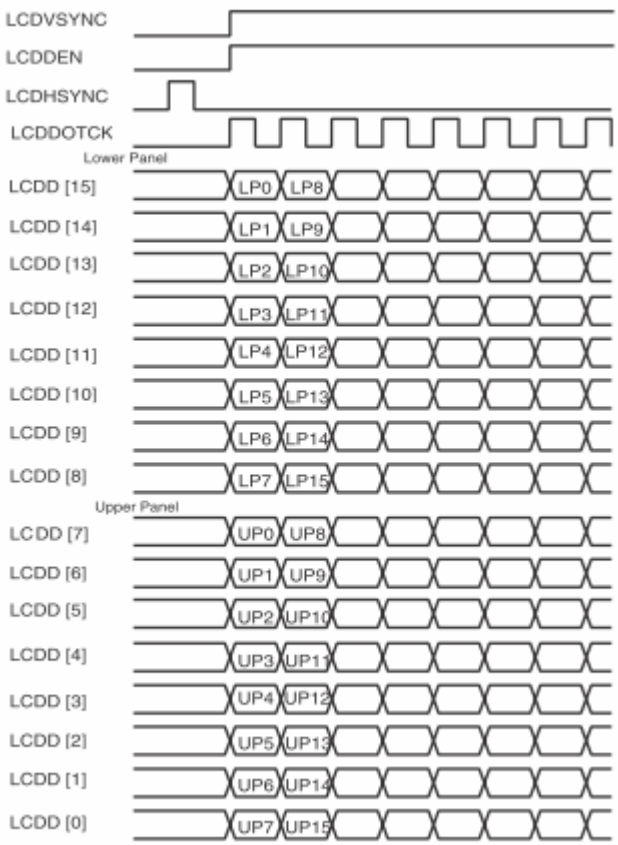


图 37-10 单色和彩色双扫描 16 位画面时序（首行扩展图）


LCDVSYNC
LCDDEN
LCDHSYNC
LCDDOTCK
Lower Panel
LCDD [15]
LCDD [14]
LCDD [13]
LCDD [12]
LCDD [11]
LCDD [10]
LCDD [9]
LCDD [8]
LP0
LP8
LP1
LP9
LP2
LP10
LP3
LP11
LP4
LP12
LP5
LP13
LP6
LP14
LP7
LP15
Upper Panel
LCDD [7]
LCDD [6]
LCDD [5]
LCDD [4]
LCDD [3]
LCDD [2]
LCDD [1]
LCDD [0]
UP0
UP8
UP1
UP9
UP2
UP10
UP3
UP11
UP4
UP12
UP5
UP13
UP6
UP14
UP7
UP15

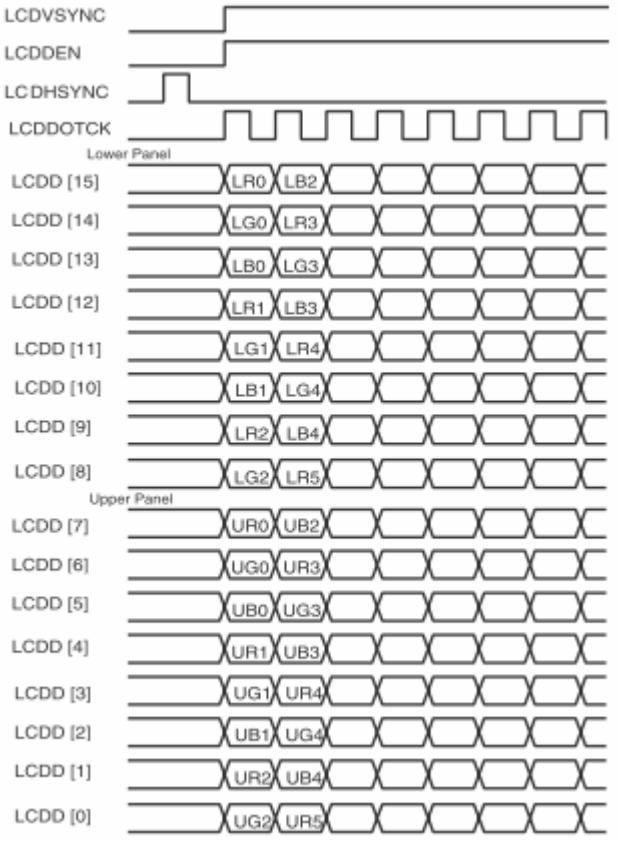

LCDVSYNC
LCDDEN
LCDHSYNC
LCDDOTCK
Lower Panel
LCDD [15]
LCDD [14]
LCDD [13]
LCDD [12]
LCDD [11]
LCDD [10]
LCDD [9]
LCDD [8]
LR0
LB2
LG0
LR3
LB0
LG3
LR1
LB3
LG1
LR4
LB1
LG4
LR2
LB4
LG2
LR5
Upper Panel
LCDD [7]
LCDD [6]
LCDD [5]
LCDD [4]
LCDD [3]
LCDD [2]
LCDD [1]
LCDD [0]
UR0
UB2
UG0
UR3
UB0
UG3
UR1
UB3
UG1
UR4
UB1
UG4
UR2
UB4
UG2
UR5

表 37-11 LCD 信号多路复用

LCD 数据总线	扫描（单，彩）	扫描（单，彩）	扫描（单，彩）	扫描（单，彩）	24 位 TFT	16 位 TFT
LCDD[23]					LCD_BLUE7	LCD_BLUE4
LCDD[22]					LCD_BLUE6	LCD_BLUE3
LCDD[21]					LCD_BLUE5	LCD_BLUE2
LCDD[20]					LCD_BLUE4	LCD_BLUE1
LCDD[19]					LCD_BLUE3	LCD_BLUE0
LCDD[18]					LCD_BLUE2	Intensity Bit
LCDD[17]					LCD_BLUE1	
LCDD[16]					LCD_BLUE0	
LCDD[15]				LCDLP7	LCD_GREEN7	LCD_GREEN4
LCDD[14]				LCDLP6	LCD_GREEN6	LCD_GREEN3
LCDD[13]				LCDLP5	LCD_GREEN5	LCD_GREEN2
LCDD[12]				LCDLP4	LCD_GREEN4	LCD_GREEN1
LCDD[11]				LCDLP3	LCD_GREEN3	LCD_GREEN0
LCDD[10]				LCDLP2	LCD_GREEN2	Intensity Bit
LCDD[9]				LCDLP1	LCD_GREEN1	
LCDD[8]				LCDLP0	LCD_GREEN0	
LCDD[7]		LCD7	LCDLP3	LCDUP7	LCD_RED7	LCD_RED4
LCDD[6]		LCD6	LCDLP2	LCDUP6	LCD_RED6	LCD_RED3
LCDD[5]		LCD5	LCDLP1	LCDUP5	LCD_RED5	LCD_RED2
LCDD[4]		LCD4	LCDLP0	LCDUP4	LCD_RED4	LCD_RED1
LCDD[3]	LCD3	LCD3	LCDUP3	LCDUP3	LCD_RED3	LCD_RED0
LCDD[2]	LCD2	LCD2	LCDUP2	LCDUP2	LCD_RED2	Intensity Bit
LCDD[1]	LCD1	LCD1	LCDUP1	LCDUP1	LCD_RED1	
LCDD[0]	LCD0	LCD0	LCDUP0	LCDUP0	LCD_RED0	

37.6 中断

LCD 控制器产生六种不同的中断请求。所有中断请求由内部 LCD 内核时钟同步。这些中断请求是：

- DMA 存储器错误 IRQ。当 DMA 正在传输数据时从 AHB 从机接收到一个错误响应的时候产生。
- FIFO 下溢 IRQ。当 FIFO 空时并行转换器试图从 FIFO 读取到一个字，这时产生 IRQ。
- FIFO 覆盖 IRQ。当 FIFO 满时 DMA 控制器试图向 FIFO 写入字时产生 IRQ。
- DMA 帧结束 IRQ。当 DMA 控制器更新帧基地址指针时产生 IRQ。该 IRQ 可被用来实现双缓冲技术。更多详情见“双缓冲技术”
- 行结束 IRQ。当每行的 LINEBLANK 周期达到并且 DMA 控制器处于无效状态时产生该 IRQ。
- 最后行结束 IRQ。当前帧的最后一行的 LINEBLANK 周期达到并且 DMA 控制器无效状态时产生该 IRQ。

每个 IRQ 可在 LCD_IER（中断使能寄存器），LCD_IDR（中断禁用寄存器），LCD_ICR（中断清除寄存器）独立的使能，禁用或清零。LCD_IMR 寄存器包括对每个 IRQ 源的屏蔽值，LDC_ISR 包括每个 IRQ 源的状态。这些寄存器的更多细节见“LCD 控制器用户接口”

37.7 配置顺序

当 LCDC 内核被激活（向 PWRCON 寄存器中的 LCD_PWR 域写入数据），DMA 控制器开始传输图片数据。因此，用户应该配置 LCDC 内核并且配置和使能 DMA 控制器优先于 LCD 控制器。另外，当 LCDC 内核被激活的时候图像数据可显示，不管 PWRCON 寄存器中的 GUARD_TIME 域中的值。

要禁用 LCD 控制器，用户应该禁用 LCDC 内核然后禁用 DMA 控制器。用户不能再使能 LIP 直到 LCDC 内核处在 IDLE 状态。可通过读取 PWRCON 寄存器中的 LCD_BUSY 位检查。

用户要使 LCDC 工作，应该遵循的初始化顺序：

- 创建或复制第一张图像，在显示缓冲存储器中显示
- 如果使用 palletized 模式，在内部 LCD Pallette 存储器中创建并存储一个调色板。
- 配置 LCD 控制器内核，而不使能它：

-LCDCON1 寄存器：编程 CLKVAL 和 BYPASS 域：这些域控制像素时钟分频器。分频器用来产生像素时钟 LCDDOTCK。编程值取决于 LCD 内核时钟，LCD 模组的尺寸和类型。LCDDOTCK 时钟周期的最小值取决于 LCD 控制器配置，该最小值可在表 37-10 中找到。方程式用于计算像素时钟分频，可在"Timegen"段尾找到。

-LCDCON2 寄存器：按照下面 LCD 控制器用户接口段的说明并且考虑 LCD 模组的类型和想要的工作模式编程其域。考虑并非所有组合都可以。

-LCDTIM1 和 LCDTIM2 寄存器：根据 LCD 模组的的数据手册和 Timegen 段的帮助编程寄存器的域。注意一些域并不适用于 STN 模组，并且必须用 0 值编程。也要注意根据 LCDC 配置，VHDLY，HPW，HBP 的最小值有限制。

-LCDFRMCFG 寄存器：编程 LCD 的尺寸。

-LCDFIFO 寄存器：用"FIFO"段中的公式编程

-DP1_2 到 DP6_7 寄存器：仅用于 STN 显示。它们包括用于产生灰阶或彩色的抖动类型 。在复位时用将会用推荐的值装载这些寄存器，所以可以不用写入。在不同的应用中可以通过转变类型来提高画面质量。

-PWRCON 寄存器：此寄存器控制 LCD 的上电顺序，所以要注意正确使用。不要使能 LCD（向 LCD_PWR 域写 1）直到前一步和 DMA 的配置已经完成。

-CONTRAST_CTR 和 CONTRAST_VAL：当 LCDCC 接口被使用时，用该寄存器调整显示的对比度。

- 配置 DMA 控制器。用户应该配置显示缓冲存储器的基地址，AHB 处理的大小和存储器中显示图像的大小。当配置 DMA 的时候用户应该使能 DMA。这样做的话用户应该配置以下寄存器：

-DMABADDR1 和 DMABADDR2 寄存器：单扫描模式下仅 DMABADDR1 寄存器必须用存储器中显示缓冲的基地址配置。双扫描模式下 DMABADDR1 应该用上画面显示缓冲基地址配置，DMABADDR2 应该用下画面显示缓冲基地址配置

-DMAFRMCFG 寄存器：编程 FRMSIZE 域。双扫描模式下每个画面的垂直尺寸。关于 BRSTLN 域，推荐值为一个 4 字的脉冲。

-DMACON 寄存器：一旦 LCD 控制器内核和 DMA 控制器已被配置，通过向该寄

存器中的 DMAEN 域写 1 来使能 DMA 控制器。

- 最后，通过向 PWRCON 寄存器的 LCD_PWR 域写 1 来使能 LCD 控制器内核，其他任何动作都需要 LCD 模组已开启。

37.8 双缓冲技术

双缓冲技术被用于避免修改当前显示帧造成的闪烁。不同于使用一个缓冲，这里有两个不同的缓冲，后缓冲（后台缓冲）和前景缓冲（正在显示的缓冲）

当 LCD 控制器正在显示前景时主机更新后缓冲。当后缓冲已被更新则主机更新 DMA 基地址寄存器。

当使用双画面 LCD 模组时，两个基地址指针都应该在同一帧被更新：有两种可能：

- 检查 DMAFRMPTx 寄存器确保有足够的时间在帧结束时更新 DMA 基地址寄存器
- 产生当帧结束 IRQ 时更新帧基地址寄存器

一旦主机更新帧基地址寄存器并且下一个 DMA 帧结束 IRQ 已到达，则交换后缓冲和首缓冲，并且主机可使用新的后缓冲。

37.9 寄存器配置指南

编程 PIO 控制器使能 LCD 信号。

在电源管理控制器中使能 LCD 控制器时钟

```
PMC_SCER = 1 << 17；//LCDC HCLK = HCK1
```

37.9.1 STN 模式示例

STN 彩色（R，G，B）320*240，8 位单扫描，70 帧每秒，主控时钟=60M 赫兹

数据速率：320*240*70*3/8=2.016MHz

```
HOZVAL=((3*320)/8 ) - 1
LINEVAL=240-1
CLKVAL=(60 MHz/ (2*2.016 MHz)) - 1= 14
LCDCON1= CLKVAL << 12
LCDCON2 = LITTLEENDIAN | SINGLESCAN | STNCOLOR | DISP8BIT| PS8BPP
LCDTIM1 = 0
LCDTIM2 = 10 | (10 << 21)
LCDFRMCFG = (HOZVAL << 21) | LINEVAL
DMAFRMCFG   = (7 << 24) + (320 * 240 * 8) / 32
```

37.9.2 TFT 模式示例

此例是基于 NEC TFT 彩色 LCD 模组 NL6448BC20-08。

TFT 640*480，16 位单扫描，60 帧每秒，像素时钟频率 = [21MHz..29MHz]

典型值 = 25.175 MHz。

主控时钟必须是 (2*(n + 1))*像素时钟频率

```
HOZVAL = 640 - 1
LINEVAL = 480 - 1
```

如果主控时钟位 50MHz

```
CLKVAL = (50 MHz/ (2*25.175 MHz)) - 1= 0
VFP = (12 -1), VBP = (31-1), VPW = (2-1), VHDLY= (2-1)
HFP = (16-1), HBP = (48 -1), HPW= (96-1)
```

```
LCDCON1= CLKVAL << 12
LCDCON2 = LITTLEENDIAN | CLKMOD |    INVERT_CLK | INVERT_LINE |
INVERT_FRM |PS16BPP | SINGLESCAN | TFT
 LCDTIM1 = VFP | (VBP << 8) | (VPW << 16) | (VHDLY << 24)
 LCDTIM2 = HBP | (HPW << 8) | (HFP << 21)
 LCDFRMCFG = (HOZVAL << 21) | LINEVAL
 DMAFRMCFG    = (7 << 24) + (640 * 480* 16) / 32
```

37.10　LCD 控制器（LCDC）用户接口

表 37-12　LCD 控制器（LCDC）用户接口

偏移量	寄存器	寄存器名	访问类型	复位值
0x0	DMA 基地址寄存器 1	DMABADDR1	R/W	0x00000000
0x4	DMA 基地址寄存器 2	DMABADDR2	R/W	0x00000000
0x8	DMA 帧指针寄存器 1	DMAFRMPT1	只读	0x00000000
0xC	DMA 帧指针寄存器 2	DMAFRMPT2	只读	0x00000000
0x10	DMA 帧地址寄存器 1	DMAFRMADD1	只读	0x00000000
0x14	DMA 帧地址寄存器 2	DMAFRMADD2	只读	0x00000000
0x18	DMA 帧配置寄存器	DMAFRMCFG	R/W	0x00000000
0x1C	DMA 控制寄存器	DMACON	R/W	0x00000000
0x800	LCD 控制寄存器 1	LCDCON1	R/W	0x00000000
0x804	LCD 控制寄存器 2	LCDCON2	R/W	0x00000000
0x808	LCD 时序寄存器 1	LCDTIM1	R/W	0x00000000
0x80C	LCD 时序寄存器 2	LCDTIM2	R/W	0x00000000
0x810	LCD 帧配置寄存器	LCDFRMCFG	R/W	0x00000000
0x814	LCD FIFO 寄存器	LCDFIFO	R/W	0x00000000
0x818	保留			
0x81C	抖动类型 DP1_2	DP1_2	R/W	0xA5
0x820	抖动类型 DP4_7	DP4_7	R/W	0x5AF0FA5
0x824	抖动类型 DP3_5	DP3_5	R/W	0xA5A5F
0x828	抖动类型 DP2_3	DP2_3	R/W	0xA5F
0x82C	抖动类型 DP5_7	DP5_7	R/W	0xFAF5FA5
0x830	抖动类型 DP3_4	DP3_4	R/W	0xFAF5
0x834	抖动类型 DP4_5	DP4_5	R/W	0xFAF5F
0x838	抖动类型 DP6_7	DP6_7	R/W	0xF5FFAFF
0x83C	电源控制寄存器	PWRCON	R/W	0x0000000e
0x840	对比度控制寄存器	CONTRAST_CTR	R/W	0x00000000
0x844	对比度值寄存器	CONTRAST_VAL	R/W	0x00000000
0x848	LCD 中断使能寄存器	LCD_IER	只写	0x0
0x84C	LCD 中断禁用寄存器	LCD_IDR	只写	0x0
0x850	LCD 中断屏蔽寄存器	LCD_IMR	只读	0x0
0x854	LCD 中断状态寄存器	LCD_ICR	只读	0x0
0x858	LCD 中断清楚寄存器	LCD_ICR	只写	0x0
0xC00	调色板入口 0	LUT ENTRY 0	R/W	
0xC04	调色板入口 1	LUT ENTRY 1	R/W	
0xC08	调色板入口 2	LUT ENTRY 2	R/W	
0xC0C	调色板入口 3	LUT ENTRY 3	R/W	
…				
0xFFC	调色板入口 256	LUT ENTRY 256	R/W	

37.10.1　DMA 基地址寄存器 1

名称：DMABADDR1

访问：读写

复位值：0x00000000

- BADDR-U

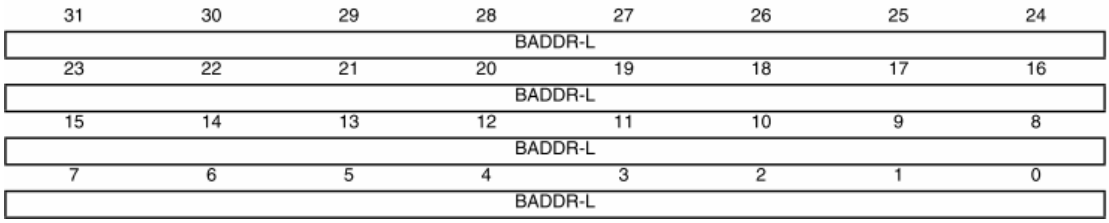
双扫描模式中上部画面的基地址。单扫描模式中整帧的基地址、

37.10.2 DMA 基地址寄存器 2

名称：　DMABADDR2

访问：读写

复位值：0x00000000



- BADDR-L

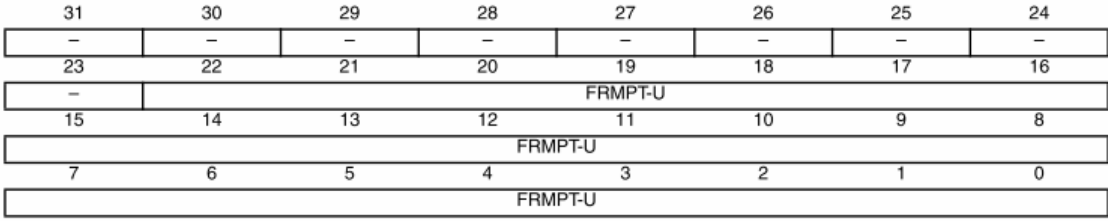
仅双扫描模式中下部画面的基地址

37.10.3　DMA 帧指针寄存器 1

名称：DMAFRMPT1

访问：只读

复位值：0x00000000



- FRMPT-U

双扫描模式中上部画面的帧指针当前值。单扫描模式中整帧帧指针当前值。从 FRMSIZE 到 0 向下计数。

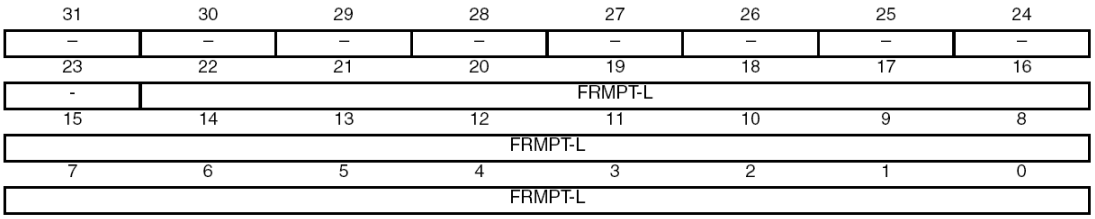
注意：该寄存器为只读。包括帧指针的当前值（到帧结束的字长）。可被用作从当前帧存储器传输字长的估算。

37.10.4　DMA 帧指针寄存器 2

名称：DMAFRMPT2

访问：只读

复位值：0x00000000



- FRMPT-L

双扫描模式中下部画面的帧指针当前值。单扫描模式中整帧帧指针当前值。从 FRMSIZE 到 0 向下计数。

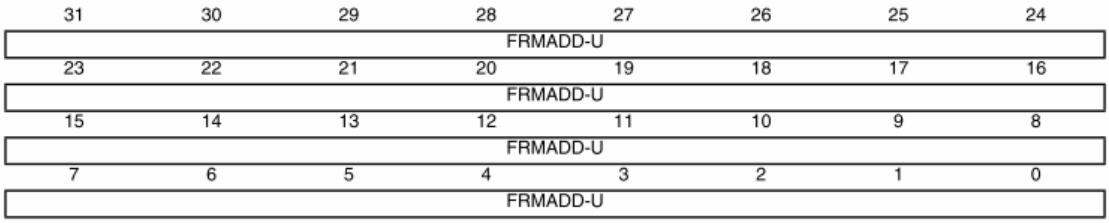
注意：该寄存器为只读。包括帧指针的当前值（到帧结束的字长）。可被用作从当前帧存储器传输字长的估算。

37.10.5 DMA 帧地址寄存器 1

名称：DMAFRMADD1

访问：只读

复位值：0x00000000



- FRMADD-U

双扫描模式下上部画面帧地址的当前值。单扫描模式中整帧地址的当前值。

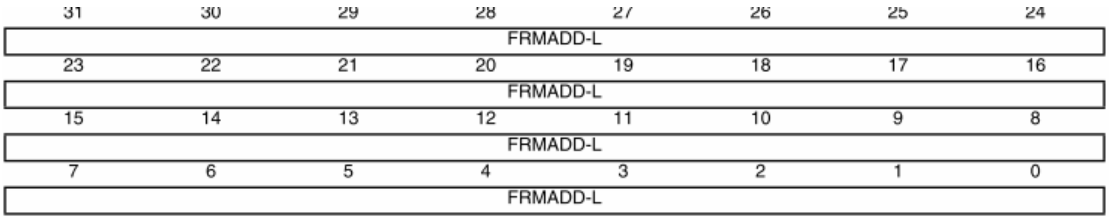
注意：该寄存器为只读，包括画面、帧总线上最后的 DMA 处理的当前值。

37.10.6 DMA 帧地址寄存器 2

名称：DMAFRMADD2

访问：只读

复位值：0x00000000



- FRMADD-L

仅单扫描模式中下部画面帧地址的当前值

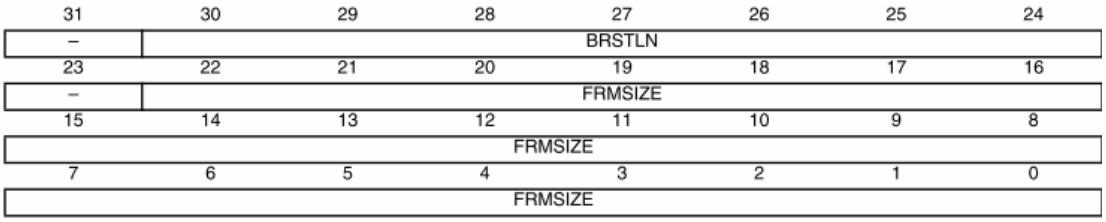
注意：该寄存器为只读，高阔画面总线中最后的 DMA 处理的当前值。

37.10.7 DMA 帧配置寄存器

名称：DMAFRMCFG

访问：读写

复位值：0x00000000



- FRMSIZE：帧长度

单扫描模式中，是字帧长度。双扫描模式中，是每个画面的大小

- BRSTLN：脉冲串长度

用需要的 burst length - 1 编程

37.10.8 DMA 控制寄存器

名称：DMACON

访问：读写

复位值：0x00000000

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

- DMAEN：DMA 使能

0：DMA 禁用

1：DMA 使能

- DMARST：DMA 复位（只写）

0：无效

1：复位 DMA 模块。仅当在禁用和空闲状态时复位 DMA 模块

- DMABUSY：DMA 忙

0：DMA 模块空闲

1：DMA 模块忙（正在 AHB 总线上做处理）

37.10.9　LCD 控制寄存器 1

名称：LCDCON1

访问：读写，除了 LINECNT：只读

复位值：　0x00002000

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

- BYPASS:：旁路 LCDDOTCK 分频器

0：分频器未被旁路。LCDDOTCK 频率由 CLKVAL 域定义。

1：LCDDOTCK 分频器被旁路。LCDDOTCK 频率等于 LCDC 时钟频率

- CLKVAL：时钟分频器

像素时钟（LCDDOTCK)频率的 9 位分频器。

Pixel_clock　=　system_clock／(CLKVAL +1) × 2

- LINECNT：行计数器（只读）

11 位行计数器的当前值。从 LINEVAL 到 0 向下计数

37.10.10　LCD 控制寄存器 2

名称：　LCDCON2

访问：读写

复位值：0x0000000

31	30	29	28	27	26	25	24
MEMOR		–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
CLKMOD	–	–	INVDVAL	INVCLK	INVLINE	INVFRAME	INVVD
7	6	5	4	3	2	1	0
PIXELSIZE			IFWIDTH		SCANMOD	DISTYPE	

- DISTYPE：显示类型

表 37.11

DISTYPE		
0	0	STN 单色
0	1	STN 彩色
1	0	TFT
1	1	保留

- SCANMOD：扫描模式

0：单扫描

1：双扫描

- IFWIDTH：接口宽度（STN）

表 37.12

IFWIDTH		
0	0	4 位（仅在 STN 单色或彩色单扫描模式中有效）
0	1	8 位（仅在 STN 单色或彩色中有效）
1	0	16 位（仅在 STN 单色或彩色双扫描模式中有效）
1	1	保留

- PIXELSIZE：每像素的位

表 37.13

PIXELSIZE			
0	0	0	每像素 1 位
0	0	1	每像素 2 位
0	1	0	每像素 4 位
0	1	1	每像素 8 位
1	0	0	每像素 16 位

1	0	1	每像素 24 位（仅 TFT 模式有效）
1	1	0	保留
1	1	1	保留

- INVVD：LCDD 极性

0：普通

1：反向

- INVFRAME：LCDVSYNC 极性

0：普通（高电平有效）

1：反向（低电平有效）

- INVLINE：LCDHSYNC 极性

0：普通（高电平有效）

1：反向（低电平有效）

- INVCLK:：LCDDOTCK 极性

0：普通（在 LCDDOTCK 下降边沿获取 LCDD ）

1:反向（在 LCDDOTCK 上升边沿获取 LCDD）

- INVDVAL：LCDDEN 极性

0：普通（高电平有效）

1：反向（低电平有效）

- CLKMOD：LCDDOTCK 模式

0：LCDDOTCK 仅在有效显示周期有效

1：LCDDOTCK 总是有效

- MEMOR：存储器顺序格式

00：大尾

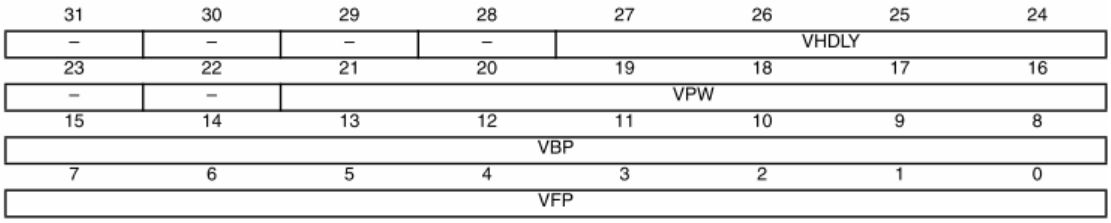
10：小尾

37.10.11　LCD 时序配置寄存器 1

名称：LCDTIM1

访问：读写

复位值：0x0000000



- VFP：垂直前边沿

TFT 模式，这些位等于帧结尾的空闲行数。

STN 模式，这些位应设置为 0

- VBP：垂直后边沿

TFT 模式，这些位等于帧起始的空闲行数

STN 模式，这些位应设置为 0

- VPW：垂直同步脉冲宽度

TFT 模式，这些位等于行数中给出的垂直同步脉冲宽度，LCDVSYNC 宽度等于(VPW+1) 行

STN 模式，这些位应置 0

- VHDLY：垂直到水平的延迟

TFT 模式，是 LCDHSYNC 上升边沿和 LCDVSYNC 上升或下降边沿之间的延迟。延迟为(VHDLY+1) LCDDOTCK 周期

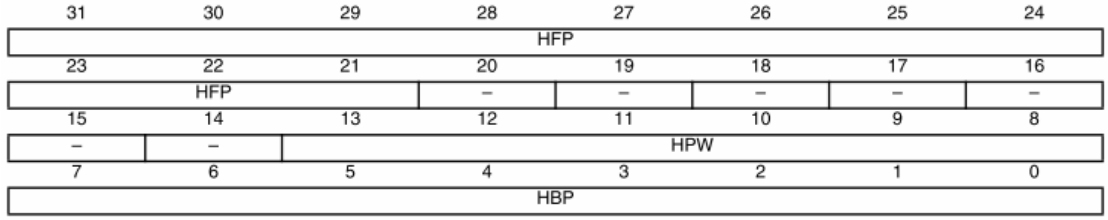
STN 模式这些位应置 0

37.10.12 LCD 时序配置寄存器 2

名称：LCDTIM2

访问：读写

复位值：0x0000000



- HBP：水平后边沿

行起始的空闲 LCDDOTCK 周期数。空闲周期为(HBP+1) LCDDOTCK 周期

- HPW：水平同步脉冲宽度

LCDDOTCK 周期中给出的 LCDHSYNC 脉冲宽度。宽度为 HPW+1) LCDDOTCK 周期

- HFP：水平前边沿

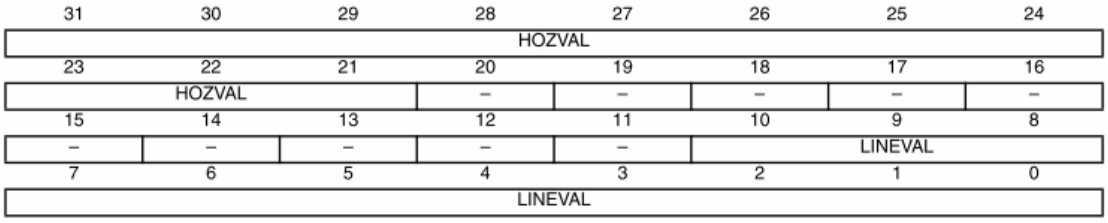
行结尾的空闲 LCDDOTCK 周期数。空闲周期为(HFP+1) LCDDOTCK 周期

37.10.13 LCD 帧配置寄存器

名称：LCDFRMCFG

访问：读写

复位值：0x0000000



- LINEVAL：LCD 模组的垂直尺寸

LINEVAL=（垂直显示尺寸）-1

双扫描模式，垂直显示尺寸参考每画面的尺寸

- HOZVAL：LCD 模组的水平尺寸

STN 模式：

-HOZVAL=(水平显示尺寸/有效 LCDD 数据行数)-1

-STN 单色模式，水平显示尺寸=水平像素数

-STN 彩色模式， 水平显示尺寸=3 倍的水平像素数

-4 位单扫描或 8 位双扫描 STN 显示模式，有效 LCDD 数据行数=4

-8 位单扫描或 16 位双扫描 STN 显示模式，有效 LCDD 数据行数=8

-如果用以上公式为 HOZVAL 计算的值不是整数，必须把它调高为整数。

TFT 模式：

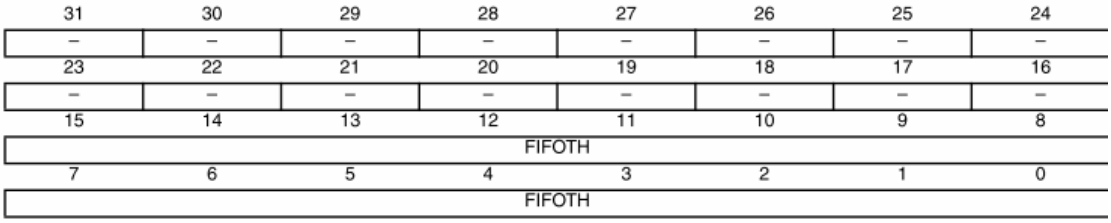
-HOZVAL=水平显示尺寸

37.10.14 LCD FIFO 寄存器

名称：LCDFIFO

访问：读写

复位值： 0x0000000



- FIFOTH:：FIFO 阈值

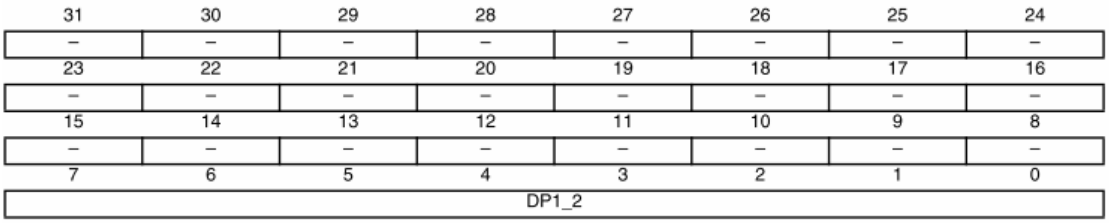
必须用下面的公式编程

FIFOTH = 512 words - (2 x DMA_BURST_LENGTH + 3)

其中：

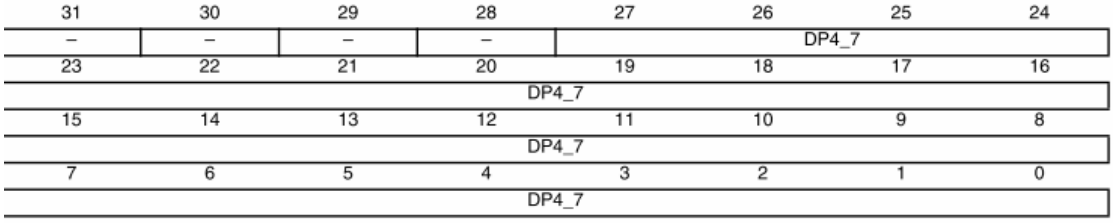
- 512 字是 FIFO 的有效长度。是单扫描模式中整个 FIFO 存储器的长度，双扫描模式一半 FIFO 存储器的长度。
- DMA_burst_length 是由 DMA 生成的传输的脉冲长度。参考“BRSTLN: Burst Length”

37.10.15 抖动图形 DP1_2 寄存器
名称：DP1_2
访问：读写
复位值：0xA5



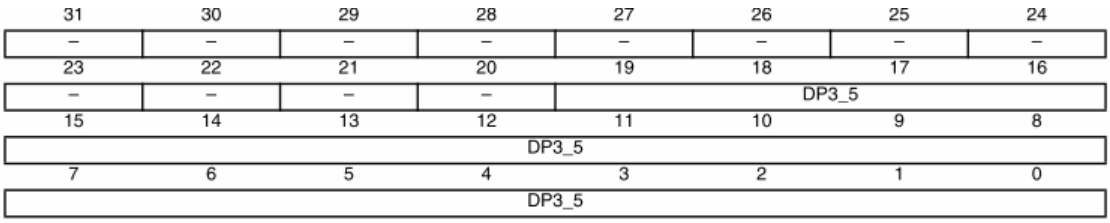
- DP1_2：二分之一占空比的图形值

37.10.16 抖动图形 DP4_7 寄存器
名称：DP4_7
访问：读写
复位值 0x5AF0FA5



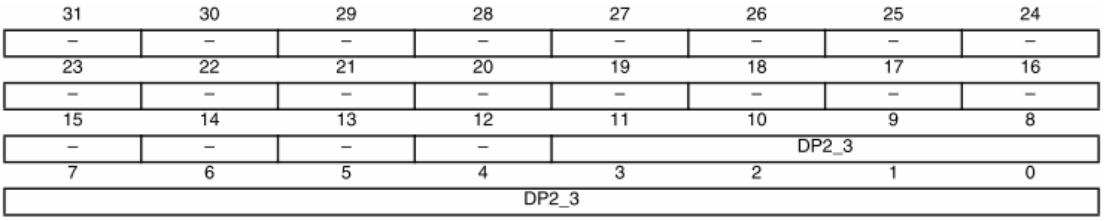
- DP4_7：七分之四占空比的图形值

37.10.17 抖动图形 DP3_5 寄存器
名称：DP3_5
访问：读写
复位值： 0xA5A5F



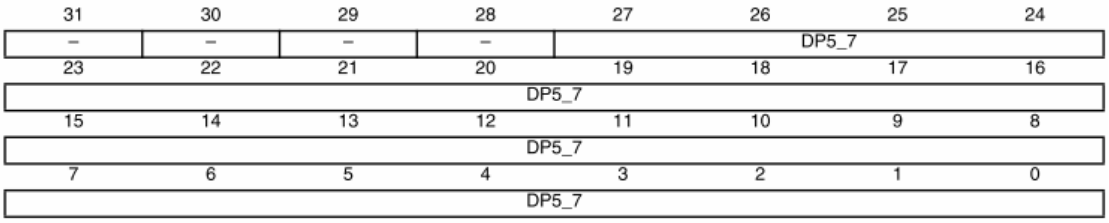
- DP3_5：五分之三占空比的图形值

37.10.18 抖动图形 DP2_3 寄存器
名称：DP2_3
访问：读写
复位值：
0xA5F



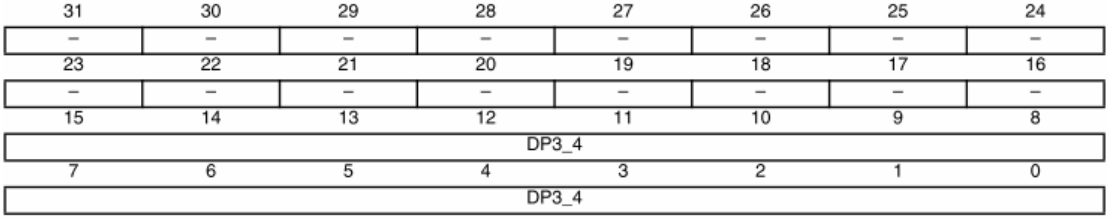
- DP2_3：占空比三分之二的图形值

37.10.19 抖动图形 DP5_7 寄存器
名称：DP5_7
访问：读写
复位值：0xFAF5FA5



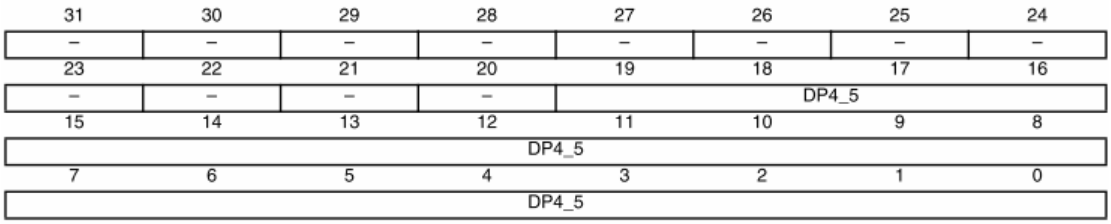
- DP5_7：占空比七分之五的图形值

37.10.20 抖动图形 DP3_4 寄存器
名称： DP3_4
访问：读写
复位值：0xFAF5



- DP3_4：占空比四分之三的图形值

37.10.21 抖动图形 DP4_5 寄存器
名称：DP4_5
访问：读写
复位值：0xFAF5F



- DP4_5：占空比五分之四的图形值

37.10.22　抖动图形 DP6_7 寄存器

名称：DP6_7

访问：读写

复位值：0xF5FFAFF

31	30	29	28	27	26	25	24
–	–	–	–	DP6_7			
23	22	21	20	19	18	17	16
DP6_7							
15	14	13	12	11	10	9	8
DP6_7							
7	6	5	4	3	2	1	0
DP6_7							

- DP6_7：占空比七分之六的图形值

37.10.23 电源控制寄存器

名称：PWRCON

访问：读写

复位值： 0x0000000e

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

- LCD_PWR：LCD 模组电源控制

0=lcd_pwr 引脚低，其他 LCD_* 引脚低电平

0->1 = lcd_*引脚有效状态，用 GUARD_TIME 帧周期的延迟设置 lcd_pwr 为高

1 = lcd_pwr 引脚高，其他 lcd_* 引脚有效

1->0 = lcd_pwr 引脚低，其他 lcd_* 引脚有效，但是在 GUARD_TIME 帧周期后置低

- GUARD_TIME

到 LCD 模组的应用控制信号和设置 LCD_PWR 高之间，设置 LCD_PWR 低和从 LCD 模组中移除控制信号之间的帧周期中的延迟

- LCD_BUSY

只读域。如果是 1，表示 LCD 忙（有效并显示闪烁，顺序供电或顺序断电）

37.10.24 对比度控制寄存器

名称：CONTRAST_CTR

访问：读写

复位值：0x00000000

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

- PS

此 2 位值选择计数器预标定器的配置。每个组合的含义如下：

表 37.14

PS		
0	0	fCOUNTER = fLCDC_CLOCK 速率时计数器预标定
0	1	fCOUNTER = fLCDC_CLOCK/2 速率时计数器预标定
1	0	fCOUNTER = fLCDC_CLOCK/4 速率时计数器预标定
1	1	fCOUNTER = fLCDC_CLOCK/8 速率时计数器预标定

- POL

该位定义输出极性。如果是 1，输出脉冲为高电平（无论何时计数器中的值小于比较寄存器 CONSTRAST_VAL 中的值，输出都是高）。如果是 0，输出脉冲为低电平

- ENA

当是 1 的时候，该位使能 PWM 发生器的工作。为 0 时，PWM 计数器停止

37.10.25 对比度值寄存器

名称：CONSTRAST_VAL

访问：读写

复位值：0x00000000

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

- CVAL

PWM 比较值。用于调整外部过滤器控制显示对比度后获得的模拟值。

37.10.26　LCD 中断使能寄存器

名称：LCD_IER

访问：只写

复位值：0x0

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

- LNIE：行中断使能

0：无效

1：使能每个行中断

- LSTLNIE：最后一行中断使能

0 无效

1：使能最后一行的中断

- EOFIE：帧结束 DMA 中断使能

0：无效

1：使能帧结束中断

- UFLWIE：FIFO 下溢中断使能

0：无效

1：使能 FIFO 下溢中断

- OWRIE:：FIFO 覆盖中断使能

0：无效

1：使能 FIFO 覆盖中断

- MERIE:：DMA 存储器错误中断使能

0：无效

1：使能 DMA 存储器错误中断

37.10.27　LCD 中断禁用寄存器

名称：LCD_IDR

访问：只写

复位值：0x0

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

- LNID：行中断禁用

0：无效

1：禁用每个行中断

- LSTLNID：最后一行中断禁用

0：无效

1：禁用最后一行中断

- EOFID：帧结束 DMA 中断禁用

0：无效

1：禁用帧结束中断

- UFLWID：FIFO 下溢中断禁用

0：无效

1：禁用 FIFO 下溢中断

- OWRID：FIFO 覆盖中断禁用

0：无效

1：禁用 FIFO 覆盖中断

- MERID：DMA 存储器错误中断禁用

0：无效

1：禁用 DMA 存储器错误中断

37.10.28　LCD 中断屏蔽寄存器

名称：LCD_IMR

访问：只读

复位值 0x0

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

- LNIM：行中断屏蔽

0：行中断禁用

1：行中断使能

- LSTLNIM：最后一行的中断屏蔽

0：最后一行的中断禁用

1：最后一行的中断使能

- EOFIM：帧结束 DMA 中断屏蔽

0：帧结束中断禁用

1：帧结束中断使能

- UFLWIM：FIFO 下溢中断屏蔽

0：FIFO 下溢中断禁用

1：FIFO 下溢中断使能

- OWRIM：FIFO 覆盖中断屏蔽

0：FIFO 覆盖中断禁用

1：FIFO 覆盖中断使能

- MERIM：DMA 存储器错误中断屏蔽

0：DMA 存储器错误中断禁用

1：DMA 存储器错误中断使能

37.10.29　LCD 中断状态寄存器

名称：LCD_ISR

访问：只读

复位值 0x0

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

- LNIS：行中断状

0：行中断无效

1：行中断有效

- LSTLNIS：最后一行的中断状态

0：最后一行的中断无效

1：最后一行的中断有效

- EOFIS：帧结束 DMA 中断状态

0：帧结束 DMA 中断无效

1：帧结束 DMA 中断有效

- UFLWIS：FIFO 下溢中断状态

0：FIFO 下溢中断无效

1：FIFO 下溢中断有效

- OWRIS：FIFO 覆盖中断状态

0：FIFO 覆盖中断无效

1：FIFO 覆盖中断有效

- MERIS：DMA 存储器错误中断状态

0：DMA 存储器错误中断无效

1：DMA 存储器错误中断有效

37.10.30　LCD 中断清零寄存器

名称：LCD_ICR

访问：只写

复位值 0x0

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

- LNIC：行中断清零

0：无效

1：清零每个行中断

- LSTLNIC：最后一行的中断清零

0：无效

1：清零最后一行的中断

- EOFIC:：帧结束 DMA 中断清零

0：无效

1:清零帧选举是中断

- UFLWIC：FIFO 下溢中断清零

0：无效

1：清零 FIFO 下溢中断

- OWRIC：FIFO 覆盖中断清零

0：无效

1：清零 FIFO 覆盖中断

- MERIC：DMA 存储器错误中断清零

0：无效

1：清零 DMA 存储器错误中断

38. AT91SAM9261 电气特性

38.1 绝对最大值

表 38-1 绝对最大值

工作温度（工业级）	-40° C 到 +125° C
仓储温度	-60° C 到 +150° C
输入引脚到地电压	-0.3V 到 +4.0V
最大工作电压(VDDCORE 和 VDDBU).	1.5V
最大工作电压(VDDOSC, VDDPLL, VDDIOM 和 VDDIOP)	4V
所有 IO 口总直流输出电流	350mA

38.2 DC 特性

以下特性的工作温度范围：Ta=-40° C 到 85° C，除非另外规范并被证明节点温度达到 Tj=100° C

表 38-2 DC 特性

符号	参数	条件	最小值	典型值	最大值	单位
Vvddcore	内核直流电压		1.08		1.32	V
Vvddbu	备用直流电压		1.08		1.32	
Vvddosc	振荡器直流电压		3.0		3.6	V
Vvddpll	PLL 直流电压		3.0		3.6	V
Vvddiom	存储器 I/ODC 电压		1.65		1.95	V
			3.0		3.6	
Vvddiop	并口 I/ODC 电压		2.7		3.6	V
Vil	输入低电平电压		-0.3		0.8	V
Vih	输入高电平电压	Vvddio=Vvddiom 或 Vvddiop	2		Vvddio +0.3	V
Vol	输出低电平电压				0.4	V
Voh	输出高电平电压	Vvddio=Vvddiom 或 Vvddiop	Vvddio -0.4			V
Ileak	输入漏电流	上拉电阻禁用			+-1	uA
Cin	输入电容	217 球形 LFBGA 封装			5.0	pF
Rpullup	上拉电阻	PA0-PA31, PB0-PB31, PC0-PC31	70	100	175	kOhm
Io	输出电流	PA0-PA31, PB0-PB31, PC0-PC31			8	mA

Bc38-2 DC 特性（续）

Isc	静态电流	Vvddcore=1.2V, MCK=0MHz,除了 POR 所有输入引脚驱动 TMS, TDI,TCK, NRST = 1	Ta=25° C		245		uA
			Ta=85° C			3000	
		Vvddbu=1.2V 逻辑电源功耗，除了 POR 所有输入驱动 WKUP = 0	Ta=25° C		1.13		uA
			Ta=85° C			9.8	

38.3 功耗

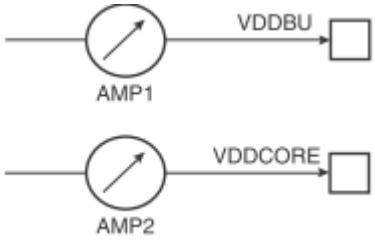
- 四种不同模式下的电源功耗：全速，空闲，静态和备份
- 外围功耗：使能然后禁用对应时钟后，测量出来的电流差。

38.3.1 功耗 VS 模式

表 38-3 和表 38-4 中的值是以下工作条件下测量的功耗值。

- Vddiom=Vddiop=3.3V
- Vddpll=Vddosc=3.3V
- I/O 口上无功耗

图 38-1 测量图



这些数字表示电源上测量的功耗。

表 38-3 不同模式的功耗

模式	条件	功耗	单位
全速	ARM 内核时钟 188MHz MCK94MHz Dhrystone 运行在 Icache。 Vddcore= 1.08V,Ta = 85° C,onto AMP2	52.7	mA
	ARM 内核时钟 240MHz MCK120MHz Dhrystone 运行在 Icache。 Vddcore= 1.2V,Ta = 85° C,onto AMP2	78.3	
	ARM 内核时钟 240MHz MCK120MHz Dhrystone 运行在 Icache。 Vddcore= 1.2V,Ta = 25° C,onto AMP2	75.3	
闲置	ARM 内核空闲状态，等待中断处理器时钟禁用 MCK96MHz Vddcore= 1.08V,Ta = 85° C,onto AMP2	12	mA
	ARM 内核空闲状态，等待中断处理器时钟禁用 MCK96MHz	14.6	

	Vddcore= 1.2V,Ta = 85° C,onto AMP2		
	ARM 内核空闲状态，等待中断处理器时钟禁用 MCK96MHz Vddcore= 1.2V,Ta = 25° C,onto AMP2	1.9	
准静态	ARM 内核时钟 500Hz MCK500Hz Vddcore= 1.08V,Ta = 85° C,onto AMP2	2300	uA
	ARM 内核时钟 500Hz MCK500Hz Vddcore= 1.2V,Ta = 85° C,onto AMP2	2450	
	ARM 内核时钟 500Hz MCK500Hz Vddcore= 1.2V,Ta = 25° C,onto AMP2	260	
备份	关断状态 Vddcore= 1.08V,Ta = 85° C,onto AMP1	6	uA
	关断状态 Vddcore= 1.2V,Ta = 85° C,onto AMP1	6.7	
	关断状态 Vddcore= 1.2V,Ta = 25° C,onto AMP1	2.5	

注意：在空闲模式无 SRAM 访问

表 38-4 外设功耗（(Ta= 25° C, VDDCORE = 1.2V）

外设	功耗	单位
PIO 控制器	4.5	µA/MHz
USART	1.7	
UHP	12.1	
UDP	8.9	
LCDC	40.2	
TWI	2.1	
SPI	9.5	
MCI	12.9	
SSC	15.3	
定时器计数器通道	3.0	

38.4 时钟特性

38.4.1 处理器时钟特性

表 38.5 处理器时钟波形参数

符号	参数	条件	MIN	MAX	单位
1/(tCPPCK)	处理器时钟频率	VDDCORE = 1.08V T = 85° C		188	MHx
1/(tCPPCK)	处理器时钟频率	VDDCORE = 1.2V T = 85° C		240	MHz

38.4.2 XIN 时钟特性

表 38-6 XIN 时钟电气特性

符号	参数	条件	Min	Max	单位
1/(tCPXIN)	XIN 时钟频率			50.0	MHx
tCPXIN	XIN 时钟周期		20.0		ns
tCHXIN	XIN 时钟高半周期		0.4 x tCPXIN	0.6 x tCPXIN	
tCLXIN	XIN 时钟低半周期		0.4 x tCPXIN	0.6 x tCPXIN	
CIN	XIN 输入电容	（1）		25	pF
RIN	XIN 下拉电阻	（1）		500	kOhm

注意：1.这些特性仅当在主振荡器旁路模式时适用（如，当 CKGR_MOR 寄存器中的 MOSCEN = 0，OSCBYPASS =1）

38.5 晶体振荡器特性

以下特性工作电压范围：Ta= -40° C 到 85° C 并且是电源最坏的情况，除非另外规范。

38.5.1 32KHz 振荡器特性

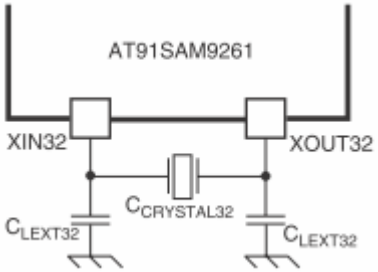
表 38-7 32KHz 振荡器特性

符号	参数	条件	Min	Typ	Max	单位
1/(tCP32KHz)	晶体振荡器频率			32.768		kHz
CCRYSTAL32	晶体负载电容	晶振为 32.768 kHz	6		12.5	pF
CLEXT32 (2)	外部负载电容	Ccrystal32 = 6pF(3)		4		pF
		Ccrystal32 = 12.5pF(3)		17		pF
	占空比		40		60	%
tST	启动时间	Vddosc = 3.3V Rs= 50 kΩ, Cl = 6pF(1)			400	ms
		Vddosc = 3.3V Rs= 50 kΩ, Cl = 12.5pF(1)			900	ms
		Vddosc = 3.3V Rs= 100 kΩ, Cl = 6pF(1)			600	ms
		Vddosc = 3.3V Rs= 100 kΩ, Cl = 12.5pF(1)			1200	ms

注意：1.Rs 等效串电阻，Cl 等效负载电容

2.Clext32 通过计算内部寄生和封装负载电容确定

3.另外用户负载电容应该从 Clext32 中减去。



38.5.2　主振荡器特性

表 38-8　主振荡器特性

符号	参数	条件	Min	Typ	Max	单位
1/(tCPMAIN)	晶体振荡器频率		3	16	20	MHz
CCRYSTAL	晶体负载电容		12.5	15	17.5	pF
CLEXT(7)	外部负载电容	Ccrystal = 12.5 pF(6)		15		pF
		Ccrystal = 15 pF(6)		18		
		Ccrystal = 17.5 pF(6)		22		
	占空比		40	50	60	%
tST	启动时间	VDDPLL = 3 to 3.6V Cs=3pF(1) 1/(tCPMAIN)=3MHz Cs=7pF(1) 1/(tCPMAIN) =8MHz Cs=7pF(1) 1/(tCPMAIN)=16MHz Cs=7pF(1) 1/(tCPMAIN)=20MHz			 20 4 2 2	ms
IDDST	待机电流功耗	待机模式			1	uA
PON	驱动电平	@ 3 MHz @ 8 MHz @16MHz @ 20 MHz			15 30 50 50	uW
IDD ON	电流消耗	@ 3 MHz(2) @ 8 MHz(3) @ 16 MHz(4) @ 20 MHz（5）		150 300 300 450	250 530 530 650	uA

注意：1.Cs 是旁路电容

2.Rs=100 到 200 欧；Cs=2.0 到 2.5pF；XOUT 上接 1 千欧串联电阻时 Cm=2 到 1.5fF（典型值，最坏情况）

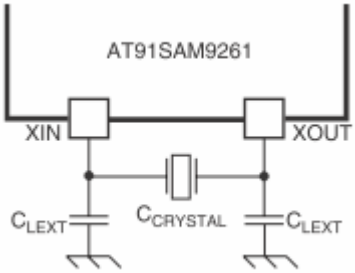
3.Rs=50 到 100 欧；Cs=2.0 到 2.5pF：Cm=4 到 3fF（典型值，最坏情况）

4.Rs=25 到 50 欧；Cs=2.5 到 3.0pF：Cm=7 到 5fF（典型值，最坏情况）

5.Rs=20 到 50 欧；Cs=3.2 到 4.0pF：Cm=10 到 8fF（典型值，最坏情况）

6.用户外加的负载电容应该从 Clext 中减去

7.Clext 通过计算内部寄生和封装负载电容确定



38.5.3 晶体特性

表 38-9 晶体特性

符号	参数	条件	Min	Typ	Max	单位
ESR	等效串联电阻 Rs	Fundamental @ 3 MHz Fundamental @ 8 MHz Fundamental @ 16 MHz Fundamental @ 20 MHz			200 100 80 50	欧
Cm	Motional 电容				8	fF
Cs	旁路电容				7	pF

38.5.4 PLL 特性

表 38-10 锁相环特性

符号	参数	条件	Min	Typ	Max	单位
Fout	输出频率	CKGR_PLL 的 OUT 域为 00	80		200	MHz
		CKGR_PLL 的 OUT 域为 10	190		240	MHz
Fin	输入频率		1		32	MHz
Ipll	电流消耗	激活模式			3	mA
		待机模式			1	uA

注意：1.启动时间取决于 PLL RC 滤波。Atmel 提供计算工具。

38.6 USB 收发器特性

38.6.1 电气特性

表 38-11 USB 收发器电气参数

符号	参数	条件	Min	Typ	Max	单位
输入电平						
Vil	低电平				0.8	V
Vih	高电平		2.0			V
Vdi	差分输入灵敏度	|(D+) - (D-)|	0.2			V
Vcm	差分输出共模范围		0.8		2.5	V
Cin	收发器电容	每个接口到地的电容			9.18	pF
I	Hi-Z 状态数据接口漏电流	0V < VIN < 3.3V	-10		+10	uA
Rext	推荐外部 USB 串联电阻	和每个 USB 引脚串接误差 5%		27		欧
输出电平						
Vol	输出低电平	通过 1.425k 欧接到 3.6V 测量	0.0		0.3	V
Voh	输出高电平	通过 14.25k 欧电阻接地测量	2.8		3.6	V
Vcrs	输出信号 crossover	测量条件见图 38-2	1.3		2.0	V
上拉电阻						
Rpui	上游端口（空闲总线）总线上拉电阻		0.900		1.575	kOhm
Rpua	上游端口（上游端口接收）总线上拉电阻		1.425		3.090	kOhm

38.6.2 开关特性

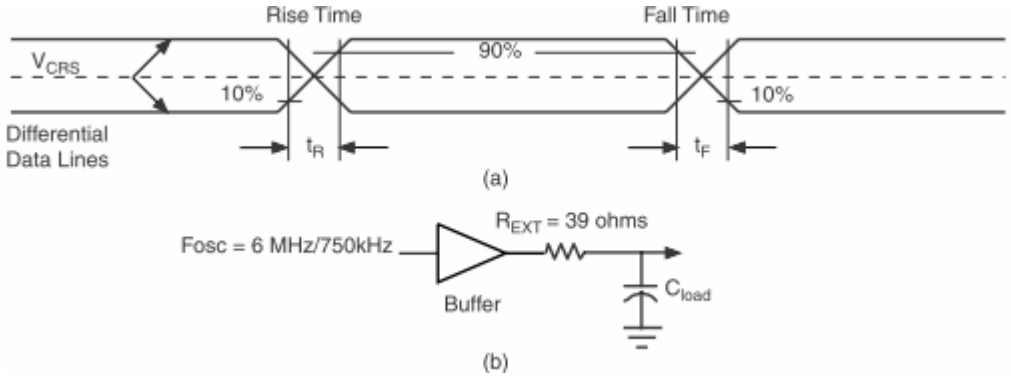
表 38-12 全速模式

符号	参数	条件	Min	Typ	Max	单位
tFR	转换上升时间	CLOAD = 400 pF	75		300	ns
tFE	转换下降时间		75		300	ns
tFRFM	上升下降时间匹配		80		125	%

表 38-13 全速模式

符号	参数	条件	Min	Typ	Max	单位
tFR	转换上升时间	CLOAD = 50 pF	4		20	ns
tFE	转换下降时间	CLOAD = 50 pF	4		20	ns
tFRFM	上升下降时间匹配		90		111.11	%

图 38-2 USB 数据信号上升和下降时间


Rise Time
Fall Time
V_{CRS}
90%
10%
10%
Differential
Data Lines
t_R
t_F
(a)
R_{EXT} = 39 ohms
Fosc = 6 MHz/750kHz
Buffer
C_{load}
(b)

38.7 EBI 时序

这些时序是在最坏的情况下给出的，T = 85° C ， VDDCORE=1.65V。
第一栏中 VDDIOM 在 1.8V 电压范围(1.65V 到 1.95V)和 30pF 负载电容
第二栏中 VDDIOM 在 3.3V 电压范围(3.0V 到 3.6V)和 50pF 负载电容

表 38-14 SMC 读信号（保持设置）

符号	参数	Min		单位
		1.8V 电源	3.3V 电源	
NRD 控制(READ_MODE = 1)				
SMC1	NRD 变高前数据设定	-0.1	-2.7	ns
SMC2	NRD 变高后数据保持	-2.6	-2.1	ns
SMC3	NRD 高到 NBS0/A0 变化	nrd hold length * tCPMCK - 0.7	nrd hold length * tCPMCK - 0.8	ns
SMC4	NRD High to NBS1 Change (1)	nrd hold length * tCPMCK - 0.7	nrd hold length * tCPMCK - 0.8	ns
SMC5	NRD High to NBS2/A1 Change (1)	nrd hold length * tCPMCK - 0.7	nrd hold length * tCPMCK - 0.8	ns
SMC6	NRD High to NBS3 Change (1)	nrd hold length * tCPMCK - 0.7	nrd hold length * tCPMCK - 0.8	ns
SMC7	NRD High to A2 - A25 Change (1)	nrd hold length * tCPMCK - 0.9	nrd hold length * tCPMCK - 1.0	ns
SMC8	NRD High to NCS Inactive (1)	(nrd hold length - ncs rd hold length) * tCPMCK - 0.4	(nrd hold length - ncs rd hold length) * tCPMCK - 0.7	ns
SMC9	NRD 脉冲宽度	nrd pulse length * tCPMCK - 0.2	nrd pulse length * tCPMCK + 0.1	ns
NCS 控制(READ_MODE = 0)				
SMC10	Data Setup before NCS High	2.3	-0.3	ns
SMC11	Data Hold after NCS High	-2.5	-1.9	ns
SMC12	NCS High to NBS0/A0 Change (1	ncs rd hold length * tCPMCK - 3.1	ncs rd hold length * tCPMCK - 3.2	ns
SMC13	NCS High to NBS1 Change (1)	ncs rd hold length * tCPMCK - 3.1	ncs rd hold length * tCPMCK - 3.2	ns
SMC14	NCS High to NBS2/A1 Change (1)	ncs rd hold length * tCPMCK - 3.1	ncs rd hold length * tCPMCK - 3.2	ns
SMC15	NCS High to NBS3 Change(1)	ncs rd hold length * tCPMCK - 3.1	ncs rd hold length * tCPMCK - 3.2	ns
SMC16	NCS High to A2 - A25 Change (1)	ncs rd hold length * tCPMCK + 1.8	ncs rd hold length * tCPMCK + 1.6	ns
SMC17	NCS High to NRD Inactive (1)	(ncs rd hold length - nrd hold length)* tCPMCK - 2.4	(ncs rd hold length - nrd hold length)* tCPMCK - 2.4	ns

SMC18	NCS 脉冲宽度	ncs rd pulse length * tCPMCK - 2.8	ncs rd pulse length * tCPMCK - 2.5	ns

注意：1.保持长度=总周期持续时间-设定时间-脉冲持续时间。“保持长度”是对于“ncs rd 保持长度”或“nrd 保持长度”

表 38-15 SMC 读信号（无保持设定）

符号	参数	Min		单位
		1.8V 电源	3.3V 电源	
NRD 控制(READ_MODE = 1)				
SMC19	NRD 高之前数据设定	2.3	0.1	ns
SMC20	NRD 高后数据保持	-2.4	-1.9	ns
NCS 控制(READ_MODE = 0)				
SMC21	NRD 高之前数据设定	4.7	1.9	ns
SMC22	NRD 高后数据保持	-2.3	-1.7	ns

表 38-16 SMC 写信号（保持设定）

符号	参数	Min		单位
		1.8V 电压	3.3V 电源	
NWE 控制 (WRITE_MODE = 1)				
SMC23	Data Out Valid before NWE High	(nwe pulse length - 1) * tCPMCK - 5.4	(nwe pulse length - 1) * tCPMCK - 4.3	ns
SMC24	Data Out Valid after NWE High (1)	nwe hold length * tCPMCK - 4.9	nwe hold length * tCPMCK - 3.6	ns
SMC25	NWE High to NBS0/A0 Change (1)	nwe hold length * tCPMCK -1.1	nwe hold length * tCPMCK - 1.2	ns
SMC26	NWE High to NBS1 Change (1)	nwe hold length * tCPMCK - 1.1	nwe hold length * tCPMCK - 1.2	ns
SMC29	NWE High to NBS2/A1 Change (1)	nwe hold length * tCPMCK - 1.1	nwe hold length * tCPMCK - 1.2	ns
SMC30	NWE High to NBS3 Change (1)	nwe hold length * tCPMCK - 1.1	nwe hold length * tCPMCK - 1.2	ns
SMC31	NWE High to A2 - A25 Change (1)	nwe hold length * tCPMCK - 1.3	nwe hold length * tCPMCK - 1.4	ns
SMC32	NWE High to NCS Inactive(1)	(nwe hold length - ncs wr hold length)* tCPMCK - 0.8	(nwe hold length - ncs wr hold length)* tCPMCK - 1.1	ns
SMC33	NWE 脉冲宽度	nwe pulse length * tCPMCK + 0.1	nwe pulse length * tCPMCK + 0.2	ns
SMC34	Data Out Valid before NCS High	(ncs wr pulse length - 1)* tCPMCK - 3.4	(ncs wr pulse length - 1)* tCPMCK - 2.3	ns
SMC35	Data Out Valid after NCS High (1)	ncs wr hold length * tCPMCK - 4.2	ncs wr hold length * tCPMCK - 2.8	ns

SMC36	NCS High to NWE Inactive (1)	(ncs wr hold length - nwe hold length)* tCPMCK - 2.3	(ncs wr hold length - nwe hold length)* tCPMCK - 2.5	ns

注意.保持长度=总周期持续时间-设定时间-脉冲持续时间。“保持长度”是对于“ncs wr 保持长度”或“nwe 保持长度”

表 38-17 SMC 写信号（无保持设定）仅 NWE 控制

符号	参数	Min		单位
		1.8V 电源	3.3V 电压	
SMC37	NWE Rising to A2-A25 Valid	2.3	1.7	ns
SMC38	NWE Rising to NBS0/A0 Valid	2.4	1.9	ns
SMC39	NWE Rising to NBS1 Change	2.4	1.9	ns
SMC40	NWE Rising to A1/NBS2 Change	2.4	1.9	ns
SMC41	NWE Rising to NBS3 Change	2.4	1.9	ns
SMC42	NWE Rising to NCS Rising	2.2	1.7	ns
SMC43	Data Out Valid before NWE Rising	(nwe pulse length - 1) * tCPMCK - 5.4	(nwe pulse length - 1) * tCPMCK - 4.3	ns
SMC44	Data Out Valid after NWE Rising	2.2	1.6	ns
SMC45	NWE Pulse Width	nwe pulse length * tCPMCK + 0.1	nwe pulse length * tCPMCK + 0.2	ns

图 38-3 NCS 控制访问的 SMC 信号

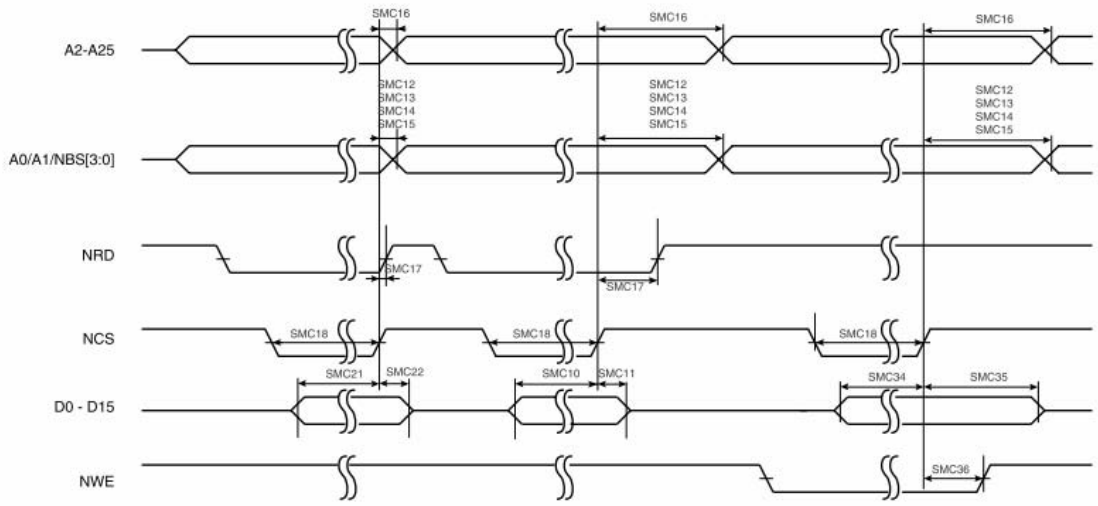
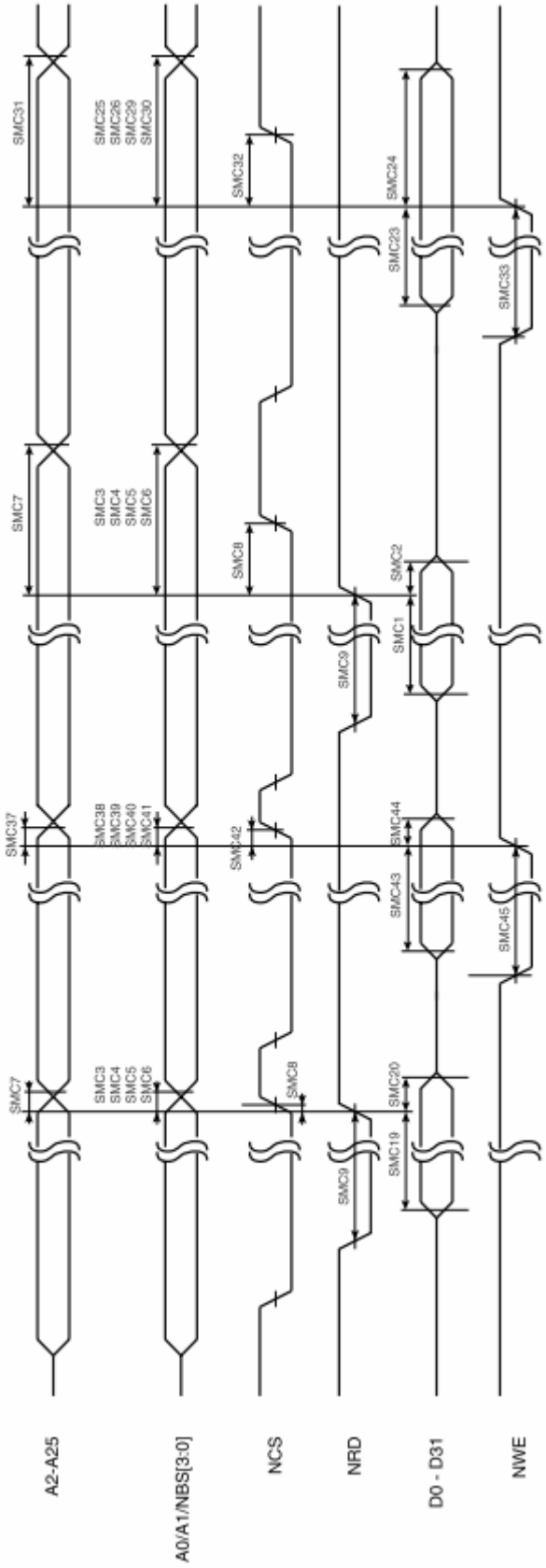


图 38-4　NRD 和 NWR 控制访问的 SMC 信号



38.7.1 SDRAMC 信号

这些时序是在 SDCK 上 10pF 负载和数据总线上 50pF 负载得到的

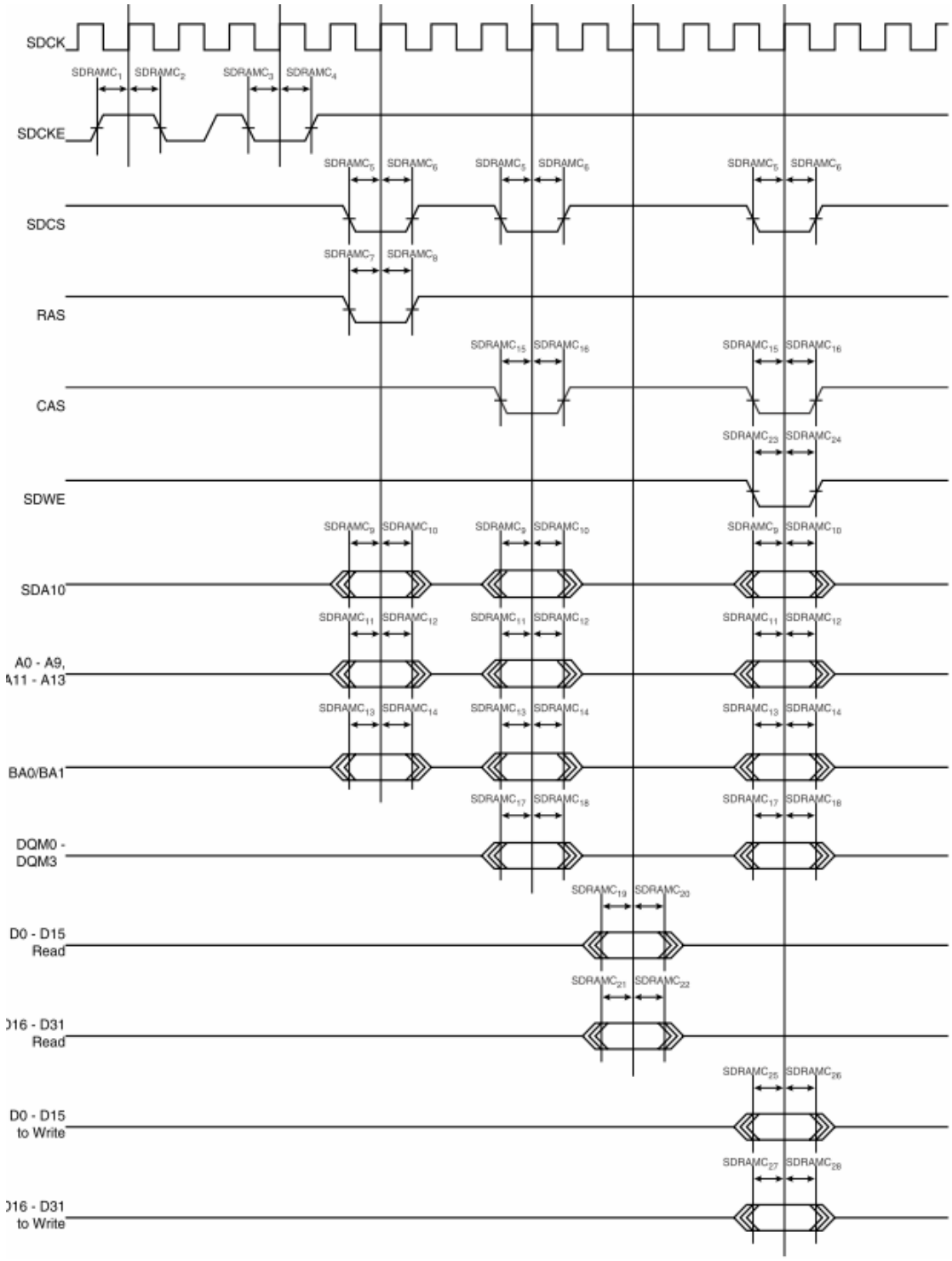
表 38-18 SDRAMC 时钟信号

符号	参数	Max		单位
		1.8V 电源	3.3V 电源	
1/(tCPSDCK)	SDRAM 控制器时钟频率	100	100	MHz

表 38-9 SDRAMC 信号

符号	参数	Min		单位
		1.8V 电压	3.3V 电源	
SDRAMC1	SDCK 上升沿之前 SDCKE 变高	4.1	4.1	ns
SDRAMC2	SDCK 上升沿后 SDCKE 变低	4.7	4.6	ns
SDRAMC3	SDCK 上升沿之前 SDCKE 变低	4.6	4.8	ns
SDRAMC4	SDCK 上升沿后 SDCKE 变高	4.8	4.8	ns
SDRAMC5	SDCK 上升沿之前 SDCS 变低	3.6	3.8	ns
SDRAMC6	SDCK 上升沿后 SDCS 变高	4.7	4.7	ns
SDRAMC7	SDCK 上升沿之前 RAS 变低	4.7	4.8	ns
SDRAMC8	SDCK 上升沿后 RAS 变高	4.6	4.6	ns
SDRAMC9	SDCK 上升沿之前 SDA10 改变	4.7	4.6	ns
SDRAMC10	SDCK 上升沿后 SDA10 改变	4.6	4.5	ns
SDRAMC11	SDCK 上升沿之前地址改变	1.4	1.3	ns
SDRAMC12	SDCK 上升沿后地址改变	4.4	4.3	ns
SDRAMC13	SDCK 上升沿之前 bank 改变	1.6	1.6	ns
SDRAMC14	SDCK 上升沿后 bank 改变	4.5	4.4	ns
SDRAMC15	SDCK 上升沿之前 CAS 变低	4.6	4.8	ns
SDRAMC16	SDCK 上升沿后 CAS 变高	4.8	4.7	ns
SDRAMC17	SDCK 上升沿之前 DQM 改变	1.7	1.8	ns
SDRAMC18	SDCK 上升沿后 DQM 改变	4.5	4.5	ns
SDRAMC19	SDCK 上升沿之前 D0-D15 设定	1.4	1.4	ns
SDRAMC20	SDCK 上升沿后 D0-D15 保持	-0.1	-0.1	ns
SDRAMC21	SDCK 上升沿之前 D16-D31 设定	1.9	2.0	ns
SDRAMC22	SDCK 上升沿后 D16-D31 保持	-0.2	-0.1	ns
SDRAMC23	SDCK 上升沿之前 SDWE 变低	4.8	4.9	ns
SDRAMC24	SDCK 上升沿后 SDWE 变高	4.6	4.6	ns
SDRAMC25	SDCK 上升沿之前 D0-D15 外部有效	3.7	3.9	ns
SDRAMC26	SDCK 上升沿后 D0-D15 外部有效	3.1	4.4	ns
SDRAMC27	SDCK 上升沿之前 D16-D31 外部有效	1.2	1.2	ns
SDRAMC28	SDCK 上升沿后 D16-D31 外部有效	2.0	3.4	ns

图 38-5 和 SDCK 相关的 SDRAMC 信号



38.8　外设时序

38.8.1 SPI

图 38-6　SPI 主控模式（CPOL = NCPHA = 0）或（CPOL= NCPHA= 1）

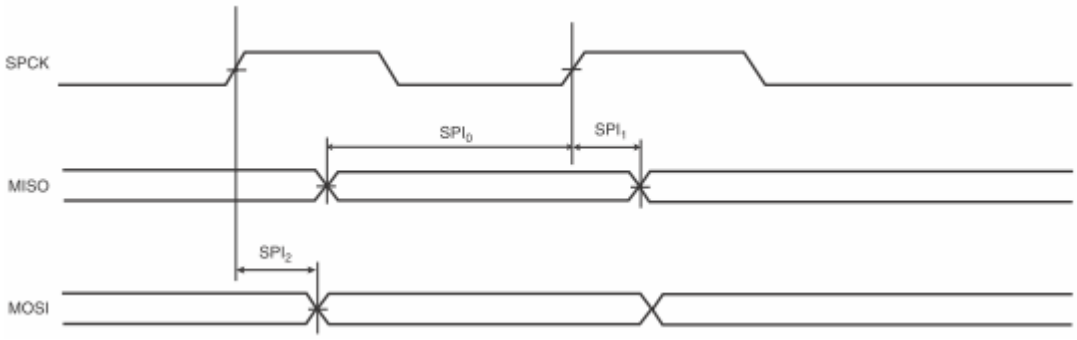


图 38-7　SPI 主控模式（CPOL=0 and NCPHA=1)或(CPOL=1 and NCPHA=0）

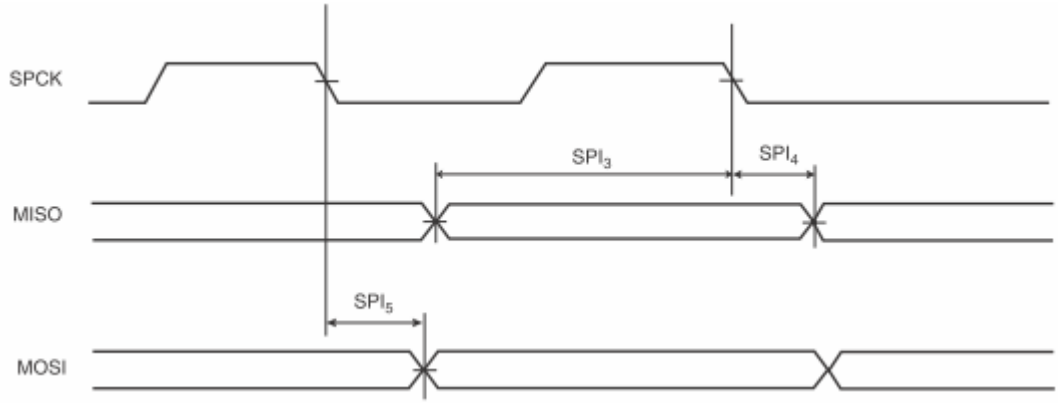


图 38-8 SPI 从控模式（CPOL=0 and NCPHA=1）或(CPOL=1 and NCPHA=0)

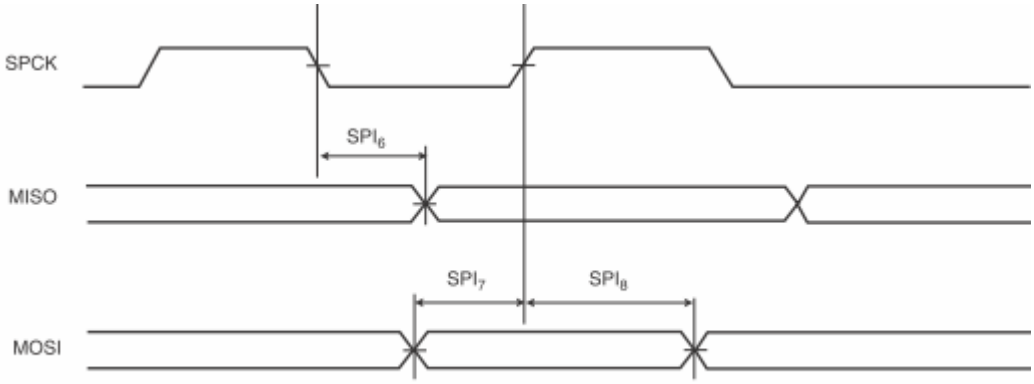


图 38-9 SPI 从控模式（CPOL = NCPHA = 0)或（CPOL= NCPHA= 1)

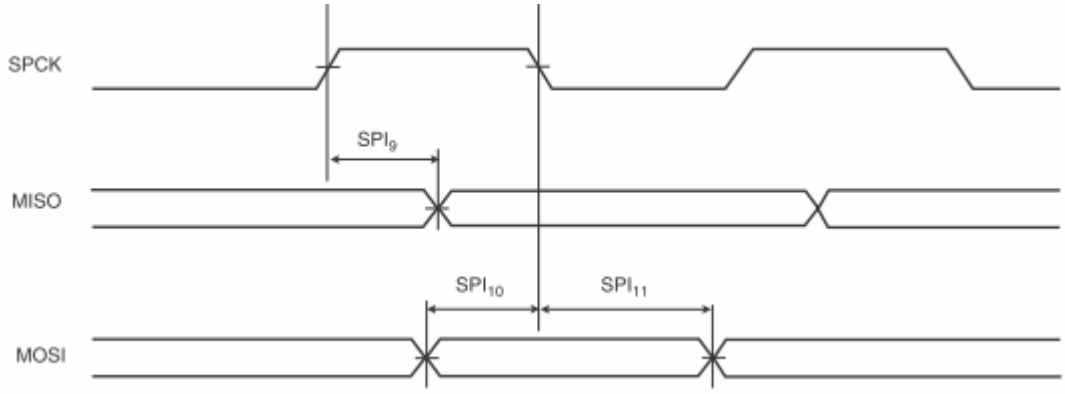


表 38-20 SPI 时序

符号	参数	条件	Min	Max	单位
SPI0	SPCK 上升前 MISO 设定时间（主）	（1）	5.6		ns
SPI1	SPCK 上升后 MISO 保持时间（主）	（1）	4.9		ns
SPI2	SPCK 上升到 MOSI 延迟（主）	（1）		0.2	ns
SPI3	SPCK 下降前 MISO 设定时间（主）	（1）	11.8		ns
SPI4	SPCK 下降后 MISO 保持时间（主）	（1）	11.1		ns
SPI5	SPCK 下降到 MOSI 延迟（主）	（1）		-5.0	ns
SPI6	SPCK 下降到 MISO 延迟（从）	（1）		6.0	ns
SPI7	SPCK 上升前 MOSI 设定时间（从）	（1）	-0.2		ns
SPI8	SPCK 上升后 MOSI 保持时间（从）	（1）	0.8		ns
SPI9	SPCK 上升到 MOSI 延迟（从）	（1）		6.0	ns
SPI10	SPCK 下降前 MOSI 设定时间（从）	（1）	0.2		ns
SPI11	SPCK 下降后 MOSI 保持时间（从）	（1）	0.8		ns

注意:1. 对 MISO　Cload 为 8pF,对 SPCK 和 MOSI Cload 为 6pF,

38.8.2　MCI

PDC 接口控制外部数据总线上，内部 MMC/SD 模块数据总线，和通过一个专用的监视 FIFO 内容状态（空或满）的状态机访问的内部系统 FIFO，FIFO 地址，和对 MMC、SD 模块的字节、区块计数器，以及应用程序（用户编程）的所有数据路线。

这些时序是在 25pF 负载下给出的，对应 1 个 MMC、SD 卡

图 38-10MCI 时序图

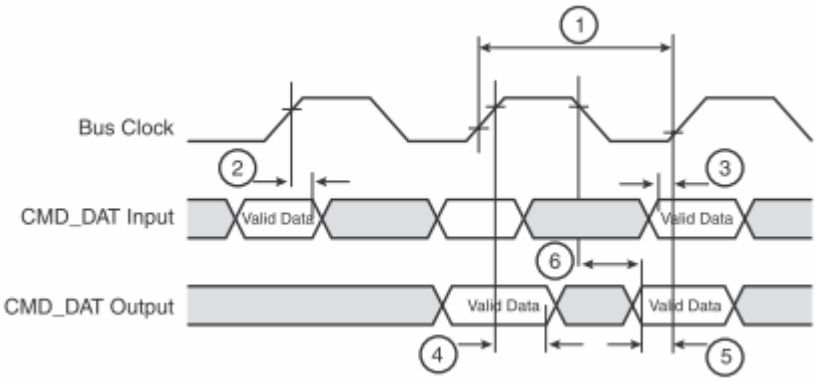


表 38-21　MCI 时序

符号	参数	Min	Max	
1	数据传输模式下（PP）CLK 频率	0	50	MHz
2	输入保持时间	7.5		ns
3	输入设定时间	4.1		ns
4	输出保持时间	-8.2		ns
5	输出设定时间	-0.3		ns

39 AT91SAM9261 机械特性

39.1 温度相关

39.1.1 温度数据

表 39-1 是根据封装总结的热阻数据

表 39-1 热阻数据

符号	参数	条件	封装	Typ	单位
θ JA	结点周围热阻	静止空气	LFBGA217	39.1	° C/W
θ JC	结点到外壳热阻		LFBGA217	7.1	

39.1.2 结点温度

平均芯片结点摄氏温度，Tj，可从下式获得：

1. Tj=Ta+(Pd* θ JA)

2.Tj=Ta+(Pd*(θ heatsink+ θ jc)

其中：

- θ JA=封装热阻。结点周围(° C/W), 在表 39-4 中提供。
- θ JC=封装热阻。结点到外壳(° C/W),在表 39-4 中提供。
- ⊖heatsink=冷却设备热阻(° C/W)，设备数据手册中提供
- Pd=设备功耗（W），从“功耗”章节中提供的数据评估得到
- Ta=环境温度(° C).

从第一个等式，用户可以得到芯片的估算使用限制并决定是否有必要使用冷却设备。如果在芯片上安装了冷却设备，则用第二个方程计算产生的平均芯片结点的摄氏温度 Tj。

39.2 封装图

图 39-1 217 球 LFBGA 封装图

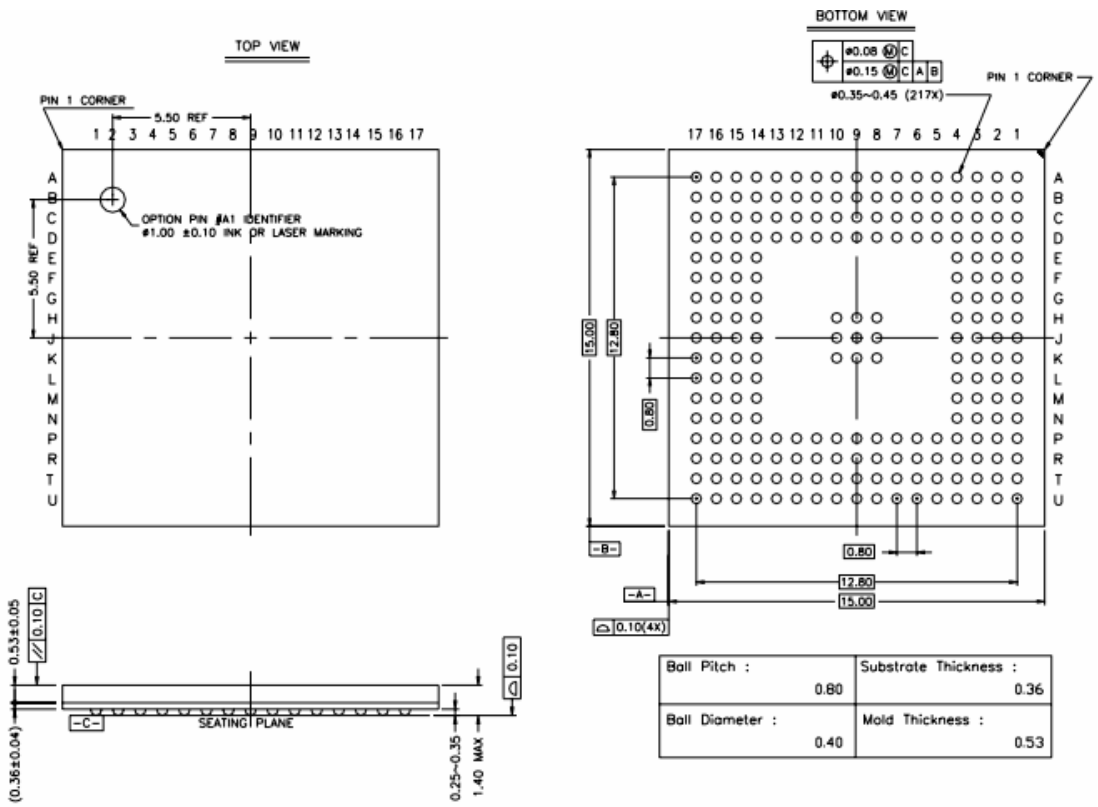


表 39-2　锡焊说明

球形焊接区	0.43 mm ± 0.05
阻焊孔	0.30 mm ± 0.05

表 39-3　设备和 217 球 LFBGA 封装最大重量

450	mg

表 39-4　217 球 LFBGA 封装特性

防潮灵敏度	3

表 39-5 封装参考

JEDEC 图参考	MO-205
JESD97 类	e 1

39.3　焊接参数

表 39-6 是 J-STD-20 推荐的焊接参数，

表 39-6　参数

参数	Green Package
平均斜坡上升率(217°C 到峰值)	最大 3°　C/sec.
预热温度 175°C ±25°C	最大 180 秒
保持温度 217°C 以上	60 秒到 150 秒
实际峰值温度高 5°C 的时间	20 秒到 40 秒
峰值温度范围	260 +0 °　C
斜坡下降速度	最大 6°　C/sec
25°　C 到峰值温度的时间	最大 8 分钟

注意：推荐锡焊温度大于 250 摄氏度

每个元件允许最多三次回流焊。

40　AT91SAM9261 订购说明

表 40-1 AT91SAM9261 订购说明

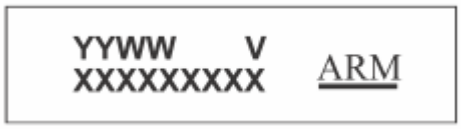
订购编码	封装	封装类型	工作温度范围
AT91SAM9261-CJ	BGA217	RoHS 兼容	工业级-40°C to 85°C

41 AT91SAM9261 勘误表

41.1 印记

所有器件都印有 Atmel 商标和订购编码

附加印记如下：



其中：

- “YY”：出厂年份
- “WW”：当年第几周
- “V”：版本
- “XXXXXXXXX”：生产批号

41.2 AT91SAM9261 勘误表 A 版部分

参考 41.1 段“印记”

41.2.1 备用电池

41.2.1.1 AHB 主控工作器件的备用电池的过度消耗

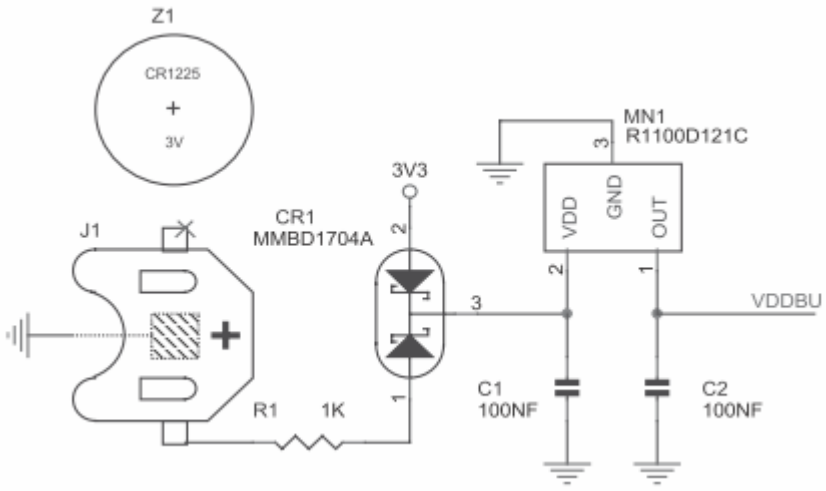
条件：

AHB 主控工作器件（LCD DMA, USB 主机 DMA,等 ），备用电池电流可以上升到 12uA， 25° C 时。

问题的永久对策、临时对策

图 41-1 展示了当主电源关闭的时候怎样仅从电池供给芯片的备份。在有源模式，未用外设的时钟应该通过电源管理控制器禁用以省电。

图 41-1 原理图



41.2.2 Boot ROM

41.2.2.1 Boot ROM 禁用看门狗

当 AT91SAM9261 在内部 ROM 上启动的时候看门狗被软件禁用。因为看门狗模式寄存器是单次写入，设计为在 SPI DataFlash 上启动的系统，不能重复使用看门狗功能。

问题的临时永久对策

如果看门狗功能对应用程序是强制性的，则系统必须在连接于 CS0（BMS=0）的外部存储器上启动.

41.2.2.2 Boot ROM：温度范围

Boot ROM 的温度范围是 0° C / 70° C。

在该温度范围外启动器件可能导致不可预测的情况。如果 AT91SAM9261 系统可能必须在 0 摄氏度以下和 70 摄氏度以上启动，则推荐连接 NCS0 的外部存储器上启动

问题的临时永久对策

无

41.2.3 总线矩阵

41.2.3.1 总线矩阵：传输锁定问题

传输锁定未被总线矩阵正确处理会导致系统冻结。这并不涉及 ARM 的传输锁定

问题的临时永久对策

避免其他总线矩阵主控传输锁定

41.2.4 MCI

41.2.4.1 MCI：未考虑 R1b 响应的忙信号

对命令 CMD7,CMD28, CMD29, CMD38, CMD42, CMD56，R1b 响应期间卡的忙状态被忽略。另外，对命令 CMD42 和 CMD56，当卡正忙的时候如果 MCI 发送数据到卡，则可能在数据接口 0 上发生冲突。

这种情况对 CMD12 命令（STOP_TRANSFER）是正确的。

临时永久对策

无

41.2.4.2 MCI：数据超时错误标志

由于数据超时错误标志未出现，MCI 无限期的拖延等待数据起始位。

临时永久对策

用软件超时发送一个 STOP 命令

41.2.4.3 MCI：不支持 STREAM 命令

MCI 不支持 STREAM READ/WRITE 命令

临时永久对策

无

41.2.4.4 MCI：WRITE_MULTIPLE_BLOCK 命令期间的 STOP

STOP 命令不终止非区段长度倍数的传输大小（PDC）的 WRITE_MULTIPLE_BLOCK。

临时永久对策

为区段长度选择一个合适的大小

41.2.5 NTRST

41.2.5.1 NTRST：未正确启动器件导致上电顺序问题

NTRST 信号由 VDDIOP 电源（3.3V）供电而 ARM 处理器由 CDDCORE 电源（1.2V）供电。

在顺序上电期间，如果 VDDIOP 电源未建立而 VDDCORE 电源复位输出建立，则 NTRST 信号未正确激活。

这将导致对内置的跟踪宏单元（ETM9）的错误复位。ARM 处理器接着进入调试状态并器件未正确启动。

临时永久对策

1.连接 NTRST 引脚到 NRST 引脚确保在所有情况产生正确的上电顺序

2.如果不需要调试功能则连接 NTRST 到 GND

41.2.6 SDRAM 控制器

41.2.6.1 SDRAM：复位后 SDCLK 时钟有效

复位后 SDRAM 时钟总是有效导致焊盘过损耗。

临时永久对策

以下顺序允许终止 SDRAM 时钟。

1.在 SDRAMC 低功耗寄存器中设置 LPCB 位为 01（自刷新）

2.向 SDRAMC 模式寄存器写 0 并向 SDRAM 中执行一次“写”伪程序。

41.2.7 串行外设接口（SPI）

41.2.7.1 SPI：SPCK 上的脉冲产生

主控模式，当 SPI 被配置为如下所示时，在 SPCK 上将产生一个附加的脉冲。

-波特率是奇数，并不同于 1

-极性被置 1

-相位被置 0

临时永久对策

不使用该配置

41.2.7.2 SPI： 当 CSAAT=1 并 SCBR = 1 时错误的 PDC 行为

如果用 CSAAT=1，SCBR（波特率）=1 编程 SPI2，并且在相同的从设备上执行两次连续的传输，两次传输之间有一个 IDLE 状态，第二个数据被发送两次。

临时永久对策

无。不使用 CSAAT=1 和 SCBR = 1 组合。

41.2.7.3SPI：LASTXFER（最后一次传输）动作

在 FIXED 模式，CSAAT 置位，“PDC 模式”下当 TX_EMPTY 标志位置位时，片选可依据写在 SPI_TDR 中的数据升高。例如，如果 PDC 向 SPI_TDR 的 24 位（LASTXFER 位）写 1，只要 TXEMPTY 标志置位则片选置位。

临时永久对策

当需要 PDC 模式时使用 PIO 模式中的 CS，CS 必须保持在传输之间一直有效。

41.2.7.4 SPI：片选和固定模式

在 FIXED 模式，如果在片选上的 PDC（不同于片选 0）上执行一次传输，则由 PDC 采样的输出 spi_size 将取决于 SPI_CSR0 寄存器的 BITS 域，无论选择什么片选。例如，如果 CSR0 被配置给一个 10 位传输而 CSR1 被配置给一个 8 位传输，当在 FIXED 模式通过片选 1 的 PDC 执行一次传输，则此次传输被当作是一个半字传输

临时永久对策

如果片选 y（不同于 0）上执行 10 位 PDC 传输，那么必须以配置 CSRy 寄存器中的 BITS 域相同的方法配置 CSR0 中的 BITS 域。

41.2.7.5 SPI：波特率置 1

当波特率被置 1（例如当串行时钟频率等于系统时钟频率），并且当 BITS 域（要发送的位数）为一个奇数值（9，11，13 或 15 的情况），则在输出 SPCK 上产生一个附加的脉冲。

如果 BITS 域等于 8，10，12，14 或 16 并波特率等于 1 时无此问题发生。

临时永久对策

41.2.7.6 SPI：软件复位

如果当因为 TXRDY 的事件发生导致在相同的时钟周期执行软件复位命令，则复位无效。

临时永久对策

执行其他的软件复位

41.2.8 串行同步控制器

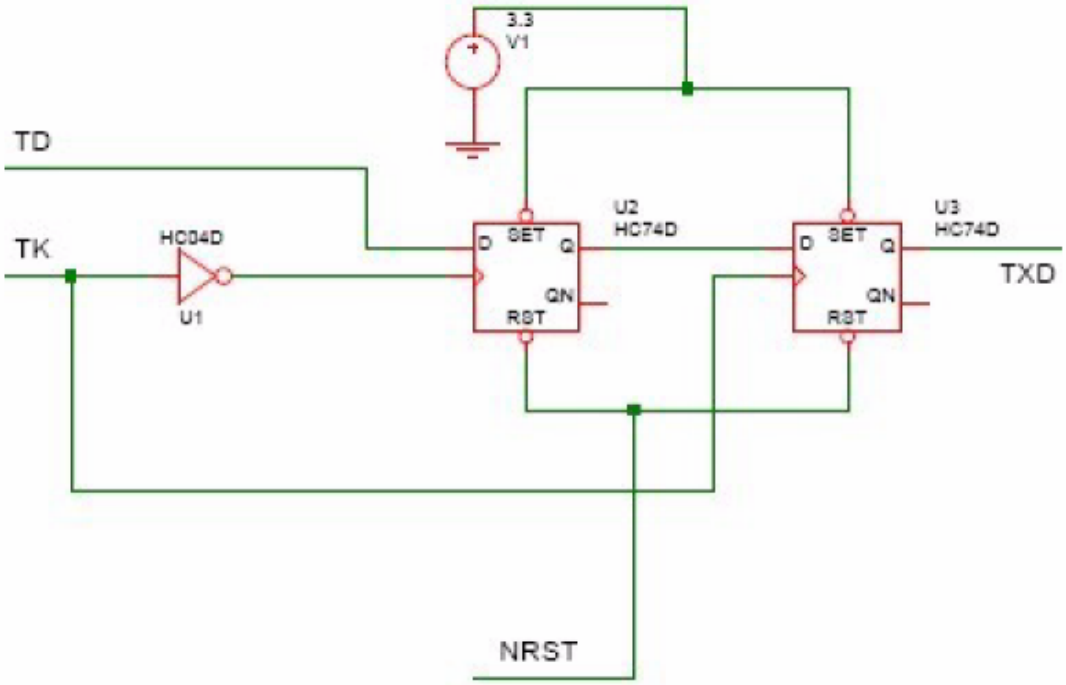
41.2.8.1 SSC：从设备模式下的发送器限制

如果 TK 被编程为输入，TF 被编程为输出并在数据发送期间请求设置高或低，则在数据开始后的一位时钟周期产生帧同步信号，一个数据位丢失。当产生周期性的同步是不存在该问题

临时永久对策

必须用一个外部装置延迟该数据一位时钟周期

下列的原理图中，TD，TK 和 NRST 是 AT91SAM9261 信号，TXD 是连接到设备的延迟数据。



41.2.8.2 SSC：主控模式下的发送限制

如果最后的有效位首先发送 (MSBF = 0) ，则帧同步期间的第一个 TAG 未被发送

临时永久对策

无

41.2.9 系统控制器

41.2.9.1 SYSC：当读取 RTT_SR 时可能遗失事件

如果由于读取 RTT_SR 时在同一慢时钟周期内发生一事件（RTTINC 或 ALMS），则相应位必须清零。这将导致该事件的遗失。

临时永久对策

软件必须处理由于中断和中断优先级 1 的唯一的中断源产生的 RTT 事件

41.2.10 双线接口（TWI）

41.2.10.1 TWI：时钟分频器

$CLDIV \times 2^{CKDIV}$的值必须小于或等于 8191，$CHDIV \times 2^{CKDIV}$的值必须小于或等于 8191

临时永久对策

无

41.2.10.2 TWI：禁用未正确执行

如果向控制寄存器(TWI_CR) 中的 MSDIS 位写 1，则进程中的任何传输立即被冻结。而且，状态寄存器(TWI_SR) 中的状态位 TXCOMP 和 TXRDY 并未复位。

临时及永久对策

用户必须在禁用 TWI 前等待传输结束。另外，必须在禁用 TWI 前禁用中断。

41.2.10.3 TWI： NACK 状态位丢失

主控帧期间，如果在无应答状态检测和 TWI_SR 中的 TXCOMP 位上升期间读取 TWI_SR，则 NACK 位未设置。

临时永久对策

用户必须通过中断等待 TXCOMP 状态位并且必须在发送未完成前不读取 TWI_SR。

注意： TXCOMP 和 NACK 域是同时设置的，NACK 域在读取 TWI_SR 后复位。

41.2.10.4 TWI：可能收到保持寄存器破坏

当装载 TWI_RHR 时，传输方向被忽略。TWI_RHR 中接收到的最后的数据字节在第一个随后发送数据字节结束时破坏。如果发生此情况，RXRDY 和 OVERRUN 两状态位都不设置。

临时、永久对策

用户必须确保接收到的数据在发送任何新数据前读取。

41.2.10.5 TWI： 软件复位

当在帧期间执行软件复位并且当 TWCK 为低时，在 READ 或 WRITE 模式初始化一次新的传输是不可能的。

临时、永久对策

无

41.2.10.6 TWI： STOP 未产生

如果以下顺序发生：

1. 在给定的地址 WRITE 1 或更多字节
2. 发送一个 STOP
3. 等待 TXCOMP 标志
4. 在相同的地址 READ 或 WRITE 1 或更多字节

然后 STOP 不会产生。

该行将显示： DADR BYTE 1, ..., BYTE n, NO STOP generated, BYTE 1, ..., BYTE n.

临时、永久对策

在上述顺序中的第四步前插入一个 TWI 时钟周期的延迟。

41.2.11 USART

41.2.11.1 USART：硬件握手中的 CTS 信号

当使用硬件握手并如果在发送器的起始位结束时间的附近 CTS 变低，则丢失一个字符。

临时、永久对策

在包括起始位上升边沿前的 2 个主控时钟周期和起始位上升边沿后的 16 个主控时钟周期之间的的时间间隔期间，CTS 必须变低。

41.2.11.2 USART： RTS 的异常情况

1.设置接收器为硬件握手模式将 RTS 接口降为低电平，即使接收器仍然关闭。USART 需要完全的配置并在设置接收器位硬件握手模式前启动。

2.当 RXBUFF 标志为 0 时在 PDC 传输期间禁用接收器，在 RTS 上无作用。将 RTS 接口升高为高电平的唯一方法是通过向两个寄存器中写 0 来复位两个 PDMA 缓冲。

临时、永久对策

无

41.2.12 UHP

41.2.12.1 UHP：Non-ISO IN 传输

条件：

考虑以下顺序

1.主机控制器发出一个 IN 记号

2.设备在短数据包中提供 IN 数据

3.主机控制器向系统存储器写接收到的数据

4.假定主机控制器正在传送两次写处理（TD 状态写和 TD 返回写）到系统存储器以完成状态更新

5.主机控制器对第一次写处理发起请求。届时处理已完成，则帧边界将混乱。

6.完成第一次写处理后，主机控制器跳过第二次写处理

结论：当这检测到其表现，主机控制器将重试相同的 IN 记号。

问题的临时、永久对策

如果系统保证状态更新可在同一帧内完成则可避免此问题。

41.2.12.2 UHP： ISO OUT 传输

条件：

考虑以下顺序：

1.在从系统存储器取 16 字节数据后，主机控制器发送一个 ISO OUT 记号

2.当主机控制器正发送 ISO OUT 数据，由于系统延迟，剩余的包字节不能发送。这导致缓冲器欠载状态

3.当处在欠载状态，如果主机控制器处在位填充进程，则导致主机控制器挂起。

结论：失败状态后，主机控制器停止发送 SOF。这导致连接的设备进入挂起状态

临时、永久对策

如果系统能保证在传输期间不会产生欠载状态的缓冲器，则可避免该问题

41.2.12.3 UHP：远程唤醒事件

条件：

当在下游端口产生一个远程唤醒事件时，OHCI 主机控制器开始发送恢复信号给设备。假设主机控制器用 20ms 发送此恢复信号。然而，如果驱动在恢复事件期间设置 HcControl.HCFS 进入 USBOPERATIONAL 状态，然后主机控制器用 EOP 终止发送恢复信号到设备。

结论：如果设备未识别恢复事件(<20 ms) ，然后设备将保持在挂起状态。

临时、永久对策

在设置 HcControl.HCFS 进入 USBOPERATIONAL 状态后主机堆栈可以做一个端口恢复。

41.2.13 UDP

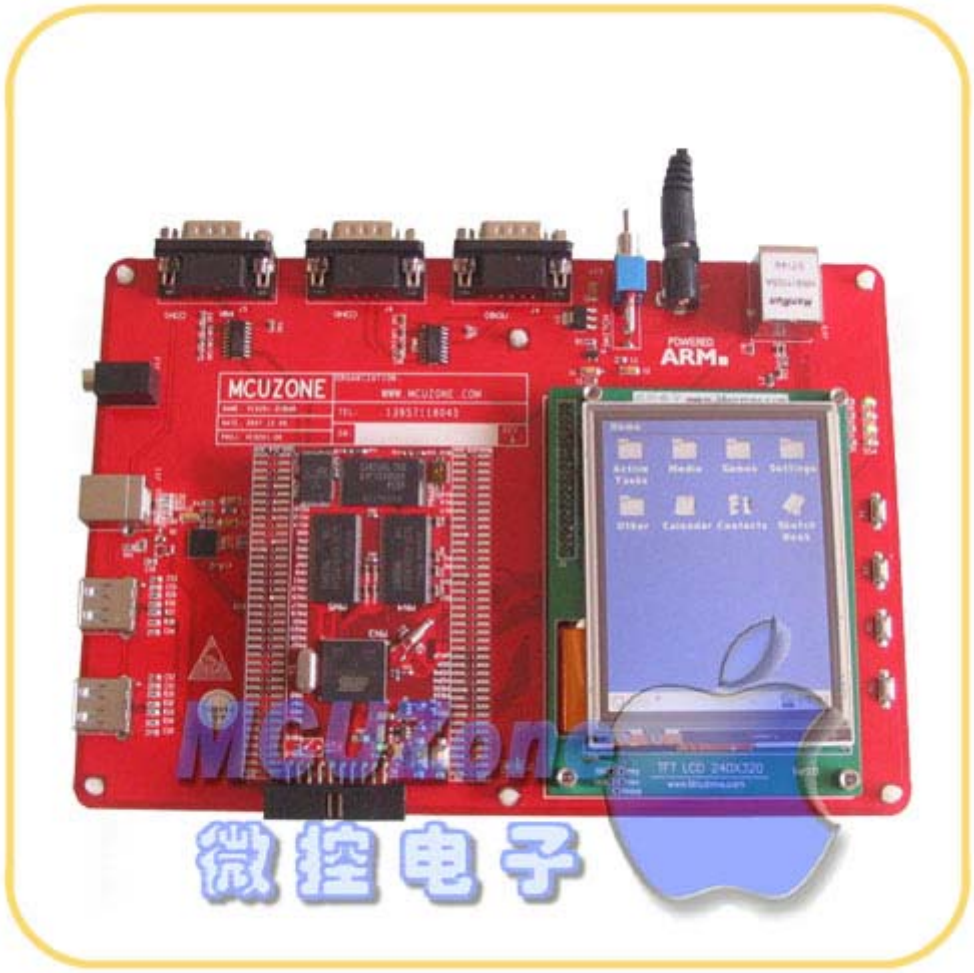
41.2.13.1 UDP： 第一个 IN 数据阶段的坏数据

第一个 IN 数据阶段的所有或部分数据未被发送。可能变为一个 0 长度的数据包。CRC 是正确的。所以 HOST 可能仅看见接收到的数据长度和请求的长度不匹配。但是即使再执行一次，控制传输还可能失败。

临时、永久对策

这些控制传输主要用在设备配置上，清零 RXSETUP 后，向 FIFO 写入数据前，若需要则软件必须设定处理请求。这个时间比上面所述需要的最大安全延迟时间还要大。如果不是这样，在 RXSETUP 清零后可能以最小的资源添加一个软件等待循环。

VC9261-EK



【VC9261-EK 硬件资源】

1、CPU

● AT91SAM9261S，工业级，无铅，217BGA（需 AT91SAM9261，加 35 元）

2、存储器

● 64MB SDRAM ，为 K4S561632 或 HY57V561620

● 4MB DataFash，AT45DB321，存储系统启动代码；可另配 8M DATAFLASH CARD

● 128MB NandFlash,存储系统内核和应用程序 ；可升级为 256MB

3、通信接口

● 1 个调试串口，1 个三线串口

● JTAG 调试接口 ，标准 20 芯接口

● 一个 10/100Mbps 网络,带网络变压器和状态指示灯

● 高质量音频输出(AT73C213)

● 一个 SD 卡接口(SPI 模式),无容量限制,AT91SAM9261 自带 SD 控制器

● 一个五线串口,支持 RS485 和 ISO7816 协议

- 两组 USB 2.0 F-S Host 端口,全兼容 OHCI USB 2.0 F-S 规格
- 一组 USB 2.0 F-S Device 端口,完全兼容 OHCI USB 2.0 F-S 规格

4、人机接口

- LCD 插槽上引出了 SAM9261 的全部 LCD 信号线，支持 TFT-LCD 和 STN-LCD
- 四线触摸屏接口(ADS7843，位于 LCD 驱动背板)
- 四个键盘和四个状态指示灯

5、其它

- 6V-9V DC 电源输入

【PCB 规格】

底板：145×205mm

核心板：60×82mm

【VC9261-EK 软件资源】

一，裸机程序

- 提供 LCD 测试程序，包括 IAR 版本、ADS 版本；
- 提供声卡测试程序，ADS 版本；
- 提供 LED（GPIO）测试程序，包括 KEIL 版本，IAR 版本；
- 提供网络测试程序，IAR 版本；
- 提供 ATMEL 软件包，主要包含以下例子（ATMEL 提供，基于 IAR 平台）：
  - DataFlash 读写程序
  - Dhrystone 测试程序
  - LCD 演示程序
  - NANDFLASH 读写程序
  - RTT 演示程序
  - SD 卡读写程序
  - LED 控制程序
  - Slideshow LCD 演示程序（类数码相框应用）
  - USB Device 演示程序
  - HID 键盘演示程序
  - Massstorage 演示程序

二，linux

- 提供 U-Boot 源码；
- 提供 busybox 源码；
- 提供 Linux 编译所需的工具链；
- 提供移植好的 Linux 2.6.xx；
- 提供以下驱动：
  - 10/100M 以太网驱动

- ◆ 2 路 USB HOST 驱动，支持 U 盘，USB mass storage，USB HID，USB2UART
- ◆ USB Device 驱动（USB gadget，详情请联系）
- ◆ LCD 驱动
- ◆ 触摸屏驱动
- ◆ 通用串口驱动
- ◆ 按键驱动
- ◆ Nand Flash 对应的 MTD 驱动
- ◆ jffs2 文件系统支持
- ◆ FAT 文件系统支持
- ◆ SPI 驱动
- ◆ TWI(I2C)驱动
- ◆ 片上 RTC 驱动
- ◆ 板载声卡驱动
- ◆ USB 声卡驱动
- ◆ DataFlash 驱动
- ◆ SD/MMC 卡驱动
- ◆ 无线网卡驱动(详情请联系)
- ◆ 摄像头驱动(详情请联系)

- 提供 Linux 网络 BT 下载应用范例，微控电子作品；
- 提供 Linux 下 webserver 应用，基于 boa；
- 提供 Linux 下网络共享应用，基于 samba；
- 提供 Linux 下 UI 应用，基于 miniGUI；
- 提供 Linux 下网络应用，基于 tftp，telnetd；

三，wince

- 提供 wince5 源码 BSP；
- 提供 wince6 源码 BSP；
- 提供以下驱动：
  - ◆ 10/100M 以太网驱动
  - ◆ 2 路 USB HOST 驱动
  - ◆ USB Device 驱动（ActiveSync）
  - ◆ LCD 驱动
  - ◆ 音频驱动
  - ◆ 触摸屏驱动
  - ◆ 通用串口驱动
  - ◆ 按键驱动
  - ◆ Nand Flash 驱动
  - ◆ TWI(I2C)驱动
  - ◆ 键盘驱动
  - ◆ SPI 驱动
  - ◆ SD/MMC 卡驱动
  - ◆ 无线网卡驱动(详情请联系)
- 提供 TCPMP 播放器软件；
- 提供手写输入软件；

- 提供截图软件；
- 提供 FTP 软件；
- 提供压缩软件；
- 提供基于 ActiveSync 的注册表编辑器；
- 提供远程控制软件；

四，RTOS

- 提供 uc/OS II（IAR 版本，Micrium 提供）；
- 提供 ucGUI（RV2.2 版本）；
- 提供 FreeRTOS（RV2.2 版本）；
- 提供 uc/OS II（IAR 版本，Micrium 提供）；
- 提供 uc/OS II（KEIL MDK 版本，Mcuzone 移植，详情请联系）；

五，配套软件

- 提供 ADS 开发环境；
- 提供 KEIL 开发环境；
- 提供 IAR 开发环境；
- 提供 RV2.2 开发环境；
- 提供 AT91-ISP 软件；
- 提供 HJTAG 调试软件；

六，文档

- 提供 EK 相关器件数据手册；
- 提供核心板和提供的原理图（pdf 格式）；
- 提供 yaffs2 的移植文档
- 提供 600 页左右的用户手册
- 提供 AT91SAM9261 中文数据册，微控电子作品

AT9261-EK

【AT9261-EK 硬件资源】

1、CPU

● AT91SAM9261S(换 9261 芯片需加 35 元），工业级，无铅，217BGA

2、存储器

● 64MB SDRAM ，为 K4S561632 或 HY57V561620

● 4MB DataFash，AT45DB321，存储系统启动代码；可另配 8M DATAFLASH CARD

● 128MB NandFlash,存储系统内核和应用程序 ；可升级为 256MB

3、通信接口

● 一个专用调试串口，即 DBGU 串口

● JTAG 调试接口 ，标准 20 芯接口

● 一个 10/100Mbps 网络,带网络变压器和状态指示灯

● 高质量音频输入，输出接口(AT73C213)

● 一个 SD 卡接口(SPI 模式),无容量限制,AT91SAM9261 自带 SD 控制器

● 一个五线串口,支持 RS485 和 ISO7816 协议

● 两组 USB 2.0 F-S Host 端口,全兼容 OHCI USB 2.0 F-S 规格

● 一组 USB 2.0 F-S Device 端口,完全兼容 OHCI USB 2.0 F-S 规格

4、人机接口

● LCD 插槽上引出了 SAM9261 的全部 LCD 信号线，支持 TFT-LCD 和 STN-LCD

- 四线触摸屏接口(ADS7843)
- 四个键盘和二个状态指示灯

5、其它

- +5V Only 电源输入
- 可选备份电池电路

【PCB 规格】

90×152mm

Powered by Team Mcuzone

QQ:8204136

Website: www.mcuzone.com

2009

MCUZone 微控电子

www.mcuzone.com