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]	

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

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：

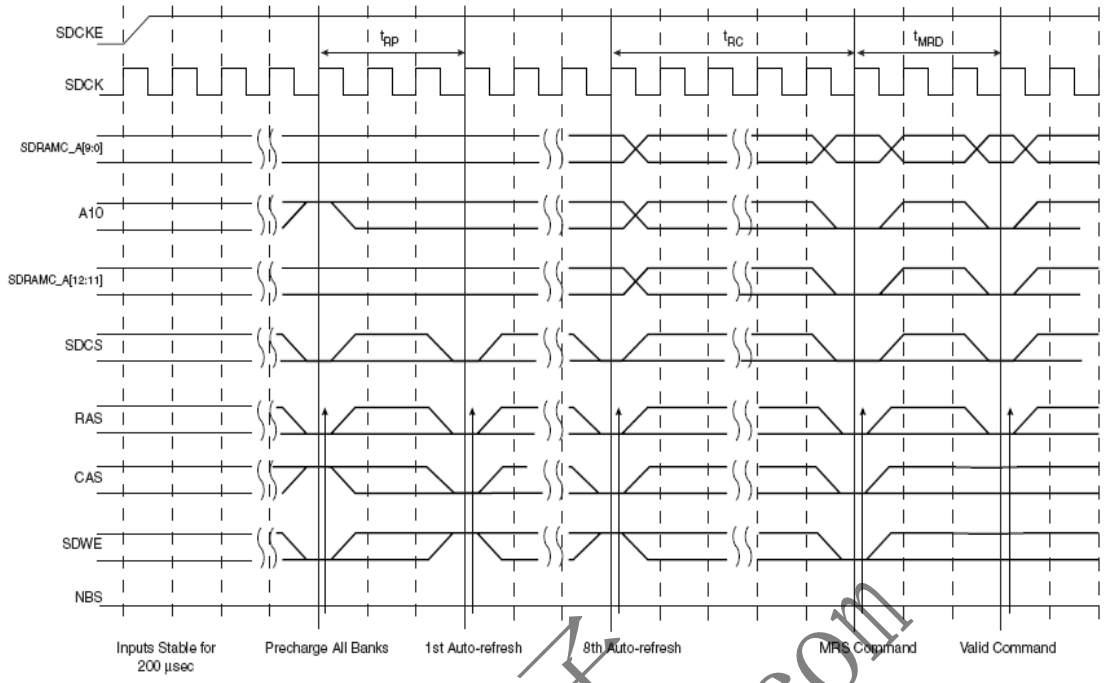
1. SDRAM 部件必须在配置寄存器中被配置：异步时序（TRC，TRAS，等），行数，列数，CAS 延迟时间，以及数据总线宽度。
2. 对移动 SDRAM，温度补偿自刷新（TCSR），驱动强度(PS)和局部矩阵自刷新(PASR)必须在低功耗寄存器中被置位。
3. SDRAM 存储类型必须在存储设备寄存器中被置位。
4. 在任何信号转换前，提供一个最小 200 微秒的暂停。
5. 一个空操作指令命令发给 SDRAM 设备。应用程序在模式寄存器中置位模式到 1 并执行一个到任何 SDRAM 地址写访问。
6. 一个所有 bank 预充电命令发到 SDRAM 设备。应用程序必须在模式寄存器中置位模式到 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



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）。

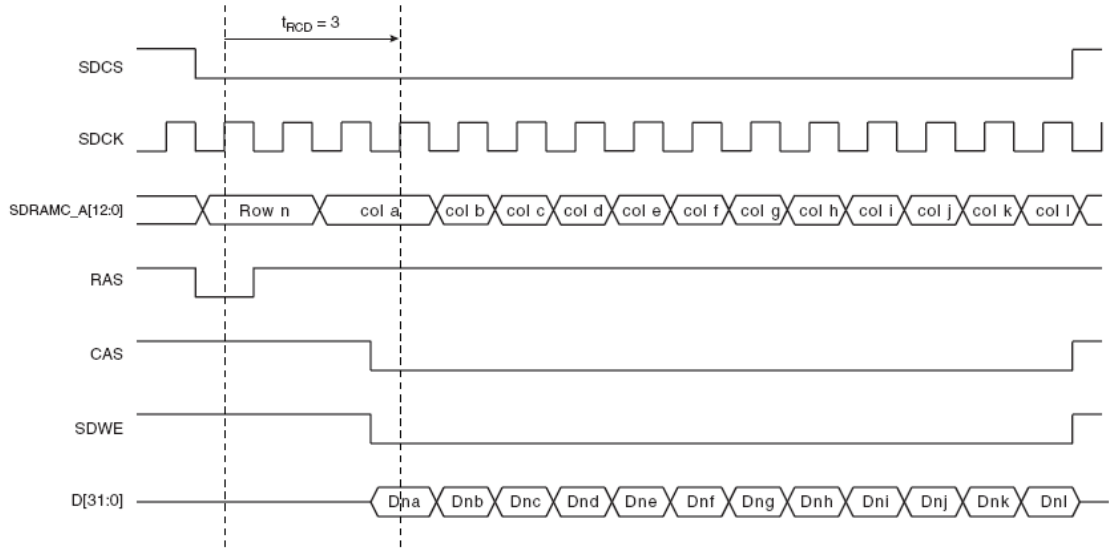
需要首先编程 AIC 来使用 SDRAM 控制器中断。

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



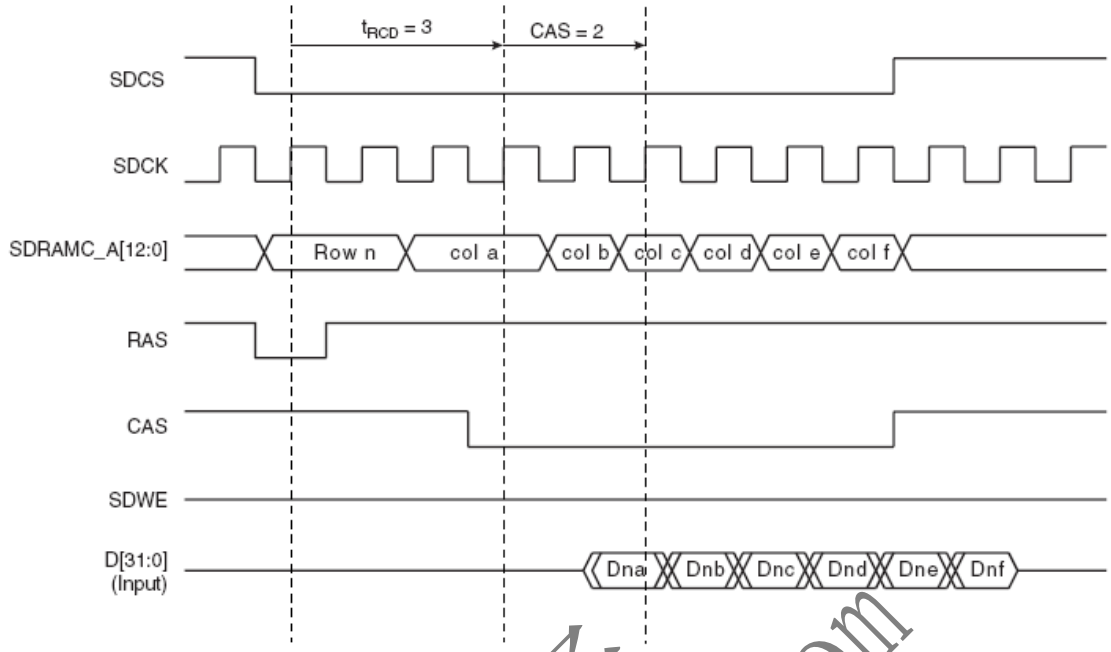
22.5.2 SDRAM 控制器读周期

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。

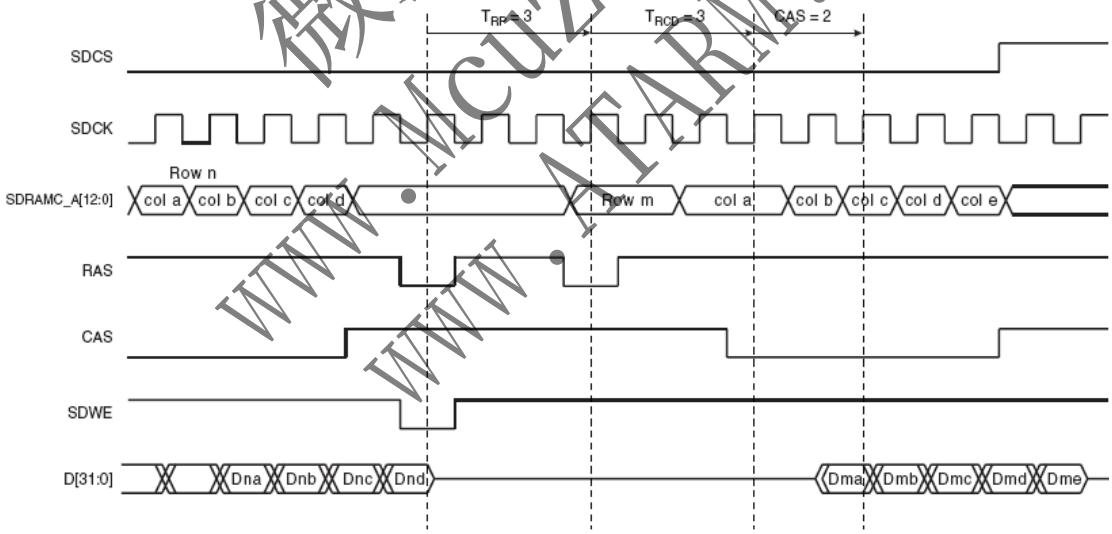
图 22-3 读突发数据，32 位 SDRAM 访问



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

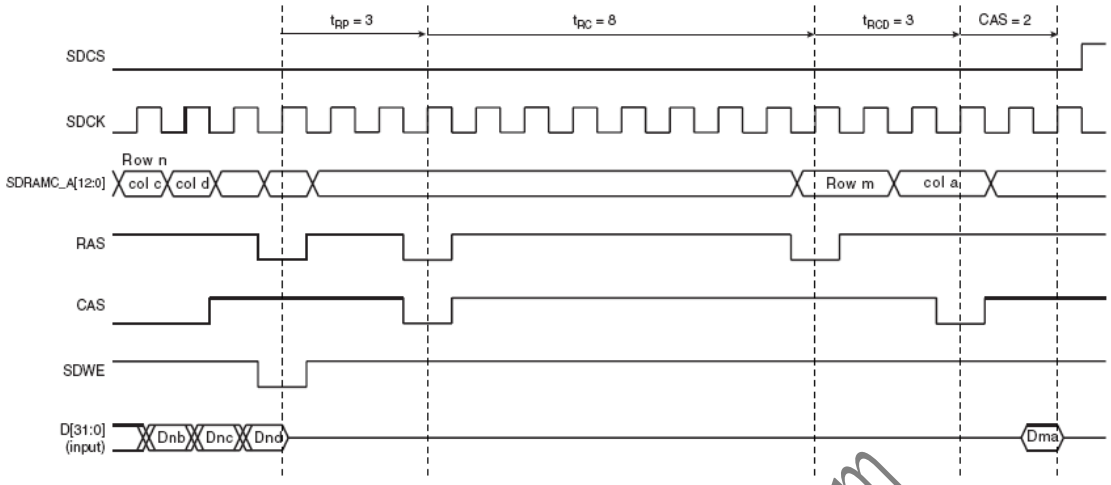
自动刷新命令被用于刷新 SDRAM 设备。刷新地址由 SDRAM 设备内部产生并在每次自动刷新后增加。SDRAM 控制器周期性的产生这些自动刷新命令。内部定时器用指示刷新周期间时钟周期数的 SDRAMC_TR 寄存器中的值装载。

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

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：

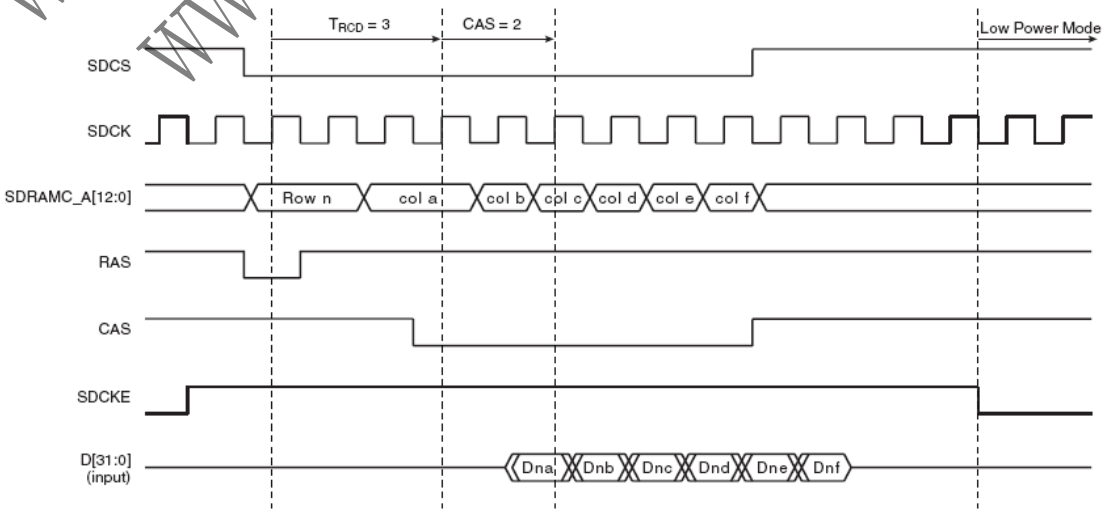
1. 自刷新模式：SDRAM 在没有 SDRAM 控制器控制下执行其自动刷新周期。被 SDRAM 消耗的电流很低。

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



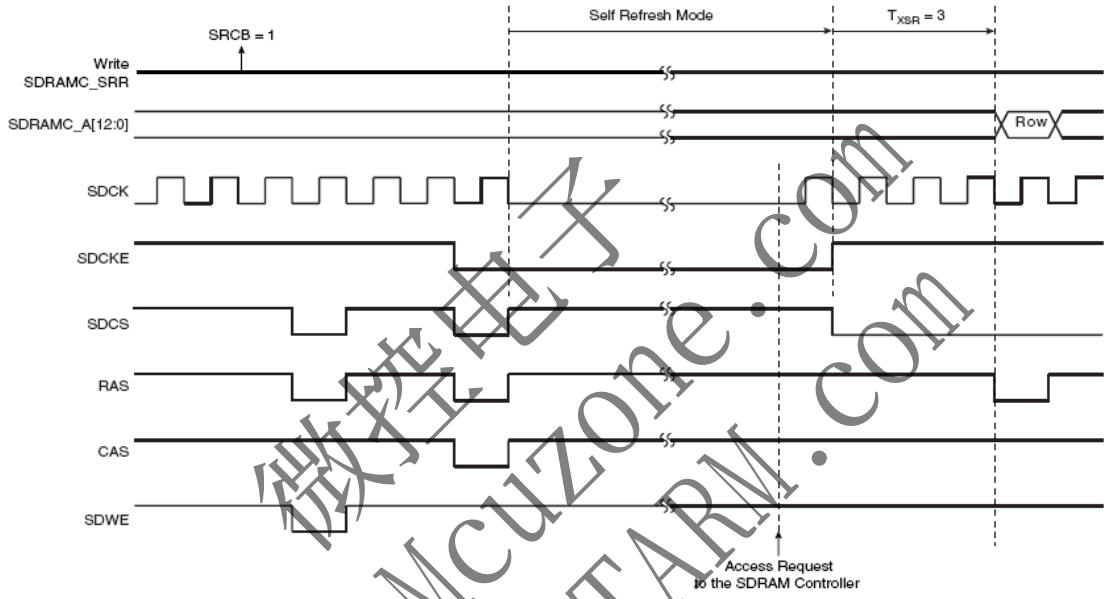
该模式通过向 SDRAMC 的低功耗寄存器中的 LPCB 域写入 1 来选择。自刷

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

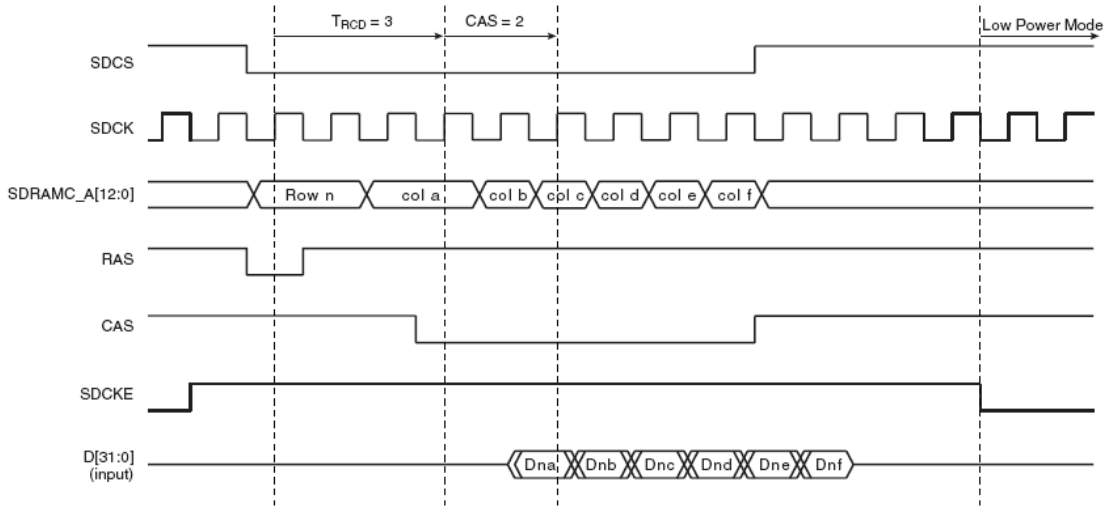


22.5.7　低功率模式

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



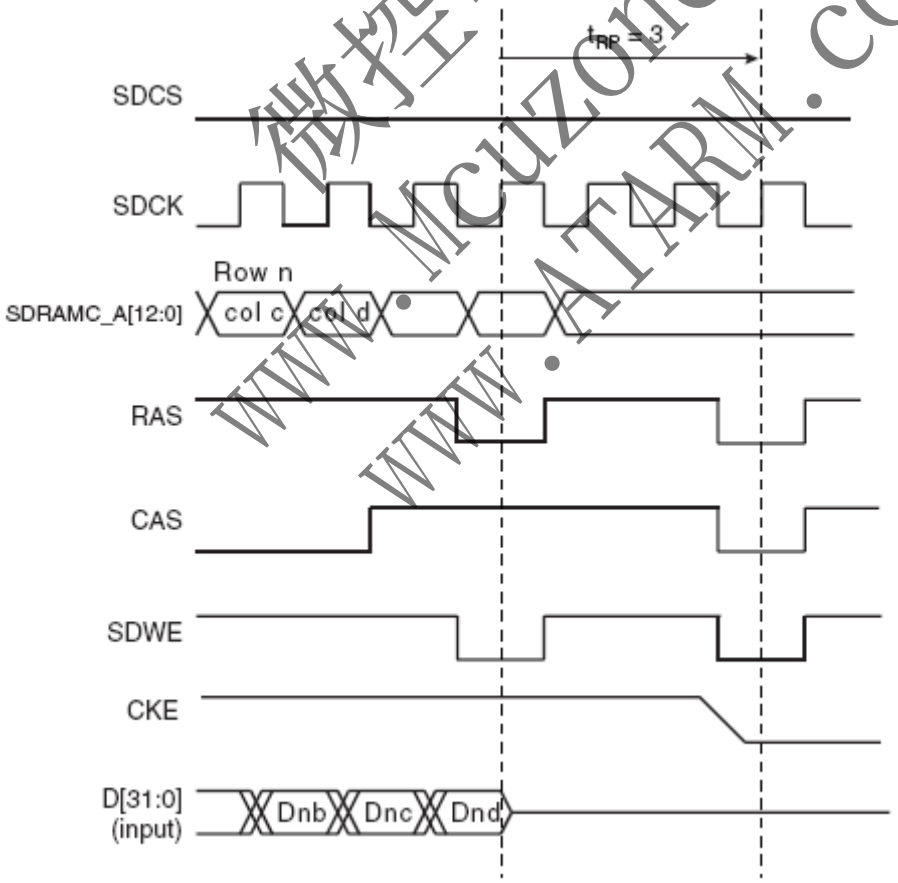
22.5.8 深度掉电模式

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

表 22-8　SDRAM 控制器存储器映射

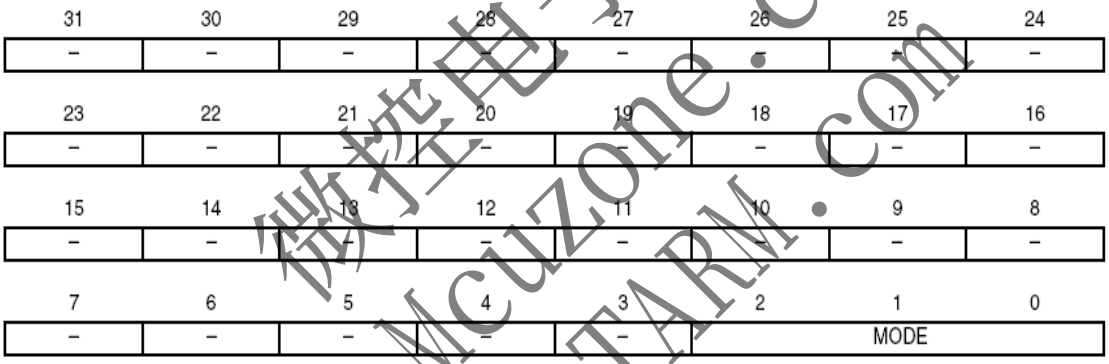
偏移量	寄存器	名称	访问	复位状态
0x00	SDRAMC 模式寄存器	SDRAMC_MR	读/写	0x00000000
0x00	SDRAMC 刷新定时器寄存器	SDRAMC_TR	读/写	0x00000000
0x04	SDRAMC 配置寄存器	SDRAMC_CR	读/写	0x852372C0
0x08	SDRAMC 低功率寄存器	SDRAMC_LPR	读/写	0x0
0x10	SDRAMC 中断使能寄存器	SDRAMC_IER	只写	-
0x14	SDRAMC 中断使无效寄存器	SDRAMC_IDR	只写	-
0x18	SDRAMC 中断屏蔽寄存器	SDRAMC_IMR	只读	0x0
0x1C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

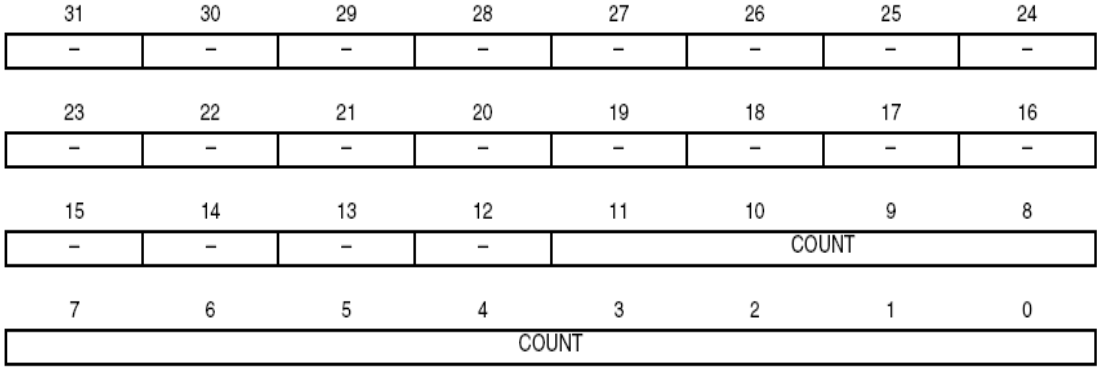
MODE			描述
0	0	0	正常模式。任何访问 SDRAM 被正常解码
0	0	1	SDRAM 控制器在访问 SDRAM 设备时发出一个空操作命令而不管周期。
0	1	0	SDRAM 控制器在访问 SDRAM 设备时发出一个“所有存储体预充电”命令而不管周期。
0	1	1	SDRAM 控制器在访问 SDRAM 设备时发出一个“装载模式寄存器”命令而不管周期。关于 SDRAM 设备基本地址的地址偏移量被用于编程模式寄存器。例如，当此模式被激活，访问“SDRAM_Base +偏移量”地址产生一个“装载模式寄存器”命令，将会将“偏移量”值写入 SDRAM 设备模式寄存器
1	0	0	SDRAM 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。

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。

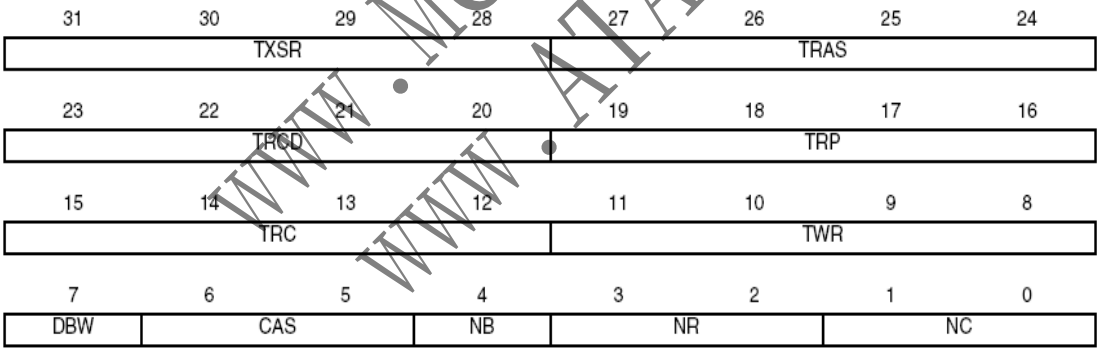
为了刷新 SDRAM 设备，必须写此 12 位域。如果未满足此条件，将不会发出刷新命令并且不会执行 SDRAM 设备的刷新。

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

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

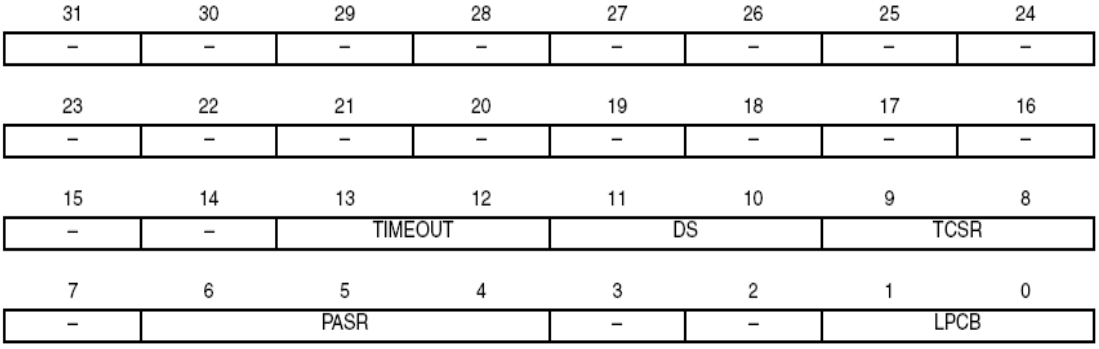
此域定义一个 SCKE 置位高电平和一个激活命令之间延迟的周期数。周期数在 0 到 15 之间。

22.6.4 SDRAM 低功率寄存器

寄存器名称：SDRAMC_LPR

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

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
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

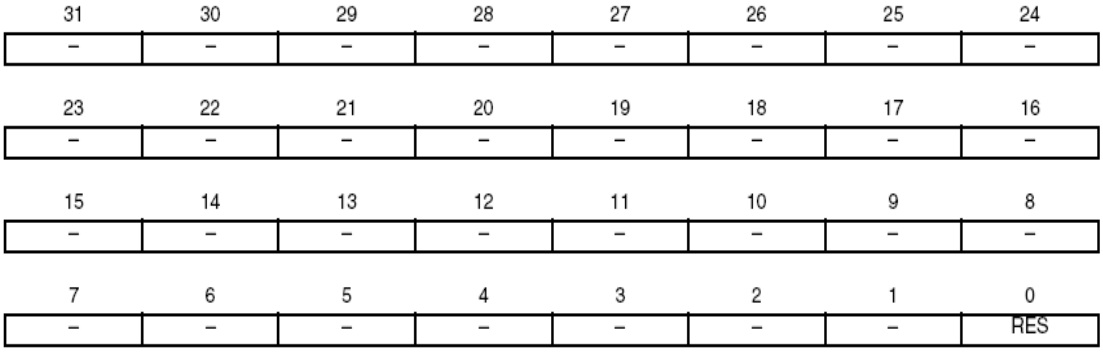
0：刷新错误中断被使无效

1：刷新错误中断被使能

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

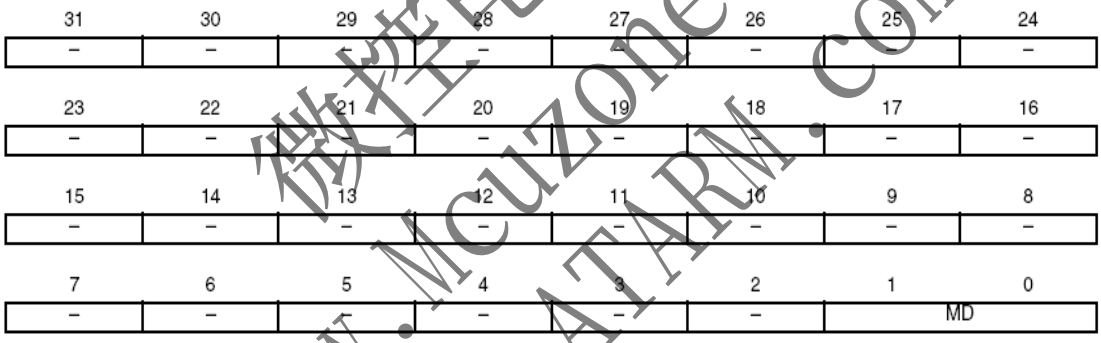
0：从最后一次读到当前无刷新错误被检测到

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

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。

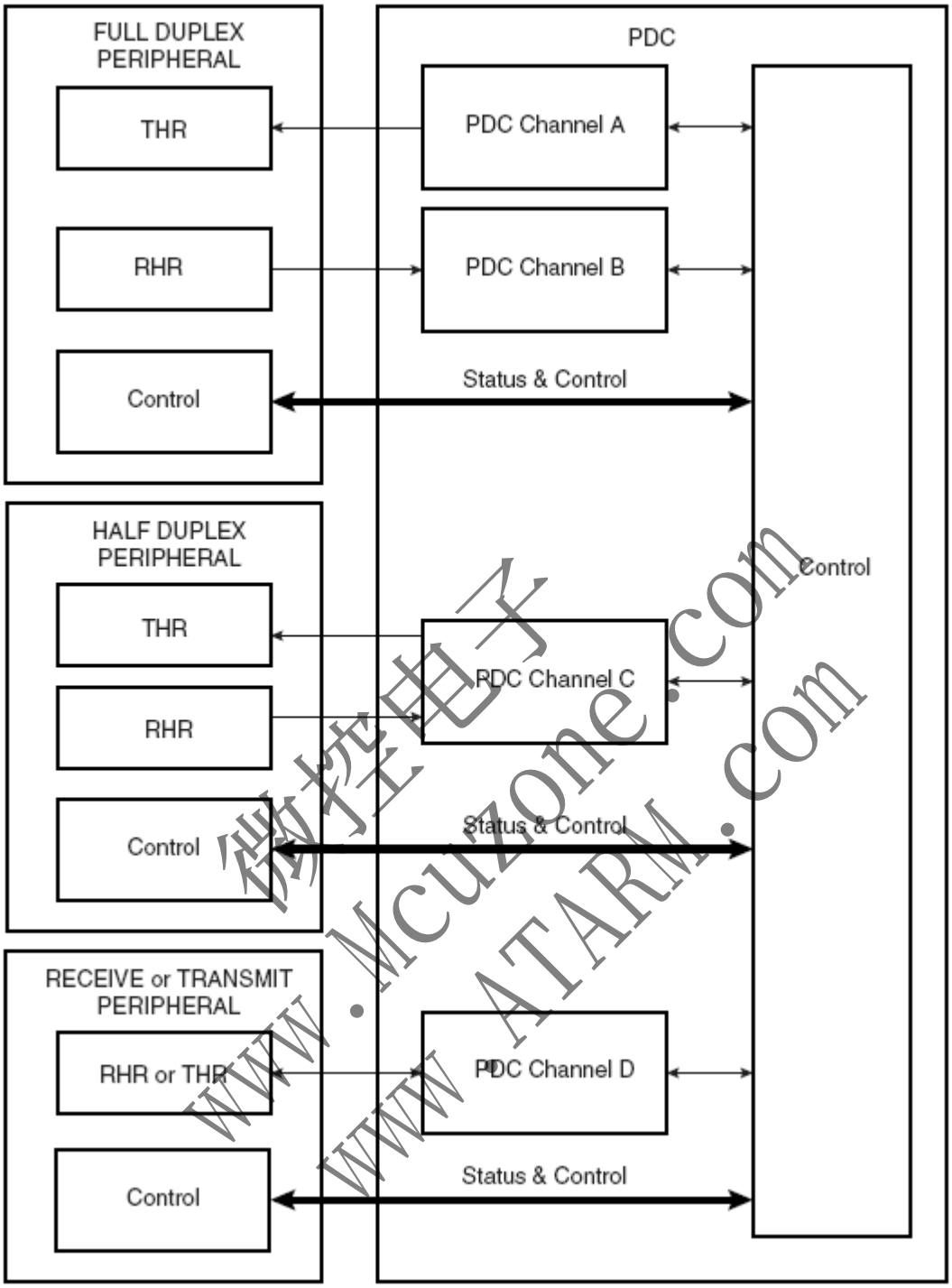
每个 PDC 通道的用户接口集成进了其服务外设的用户接口。单向通道（仅接收或仅发送）的用户接口，包含两个 32 位存储器指针和两个 16 位计数器，一组用于当前传送（指针，计数器），另一组用于下一次传送。双向通道用户接口包含四个 32 位存储器指针和四个 16 位计数器，每组（指针，计数器）被用于当前传送，下一次传送，当前接收和下一次接收。

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



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

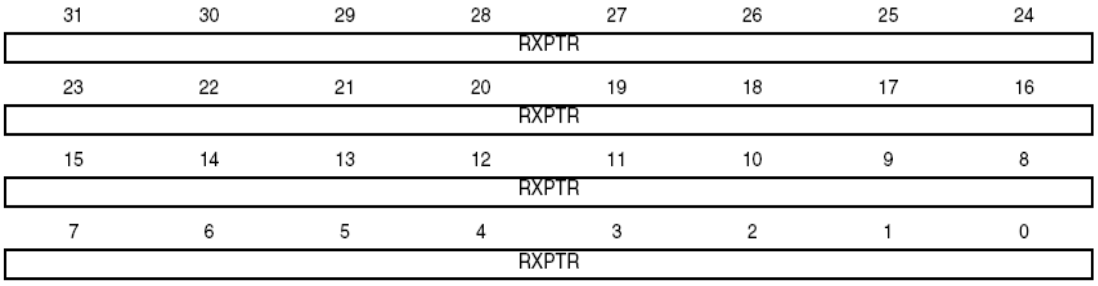
偏移量	寄存器	名称	访问	复位值
0x100	接收指针寄存器	PERIPH(1)_RPR	读/写	0
0x104	接收计数器寄存器	PERIPH_RCR	读/写	0
0x108	发送指针寄存器	PERIPH_TCR	读/写	0
0x10C	发送计数器寄存器	PERIPH_TCR	读/写	0
0x110	接收下一个指针寄存器	PERIPH_RNPR	读/写	0
0x114	接收下一个计数器寄存器	PERIPH_RNPR	读/写	0
0x118	发送下一个指针寄存器	PERIPH_TNPR	读/写	0
0x11C	发送下一个计数器寄存器	PERIPH_TNPR	读/写	0
0x120	传送控制寄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



- RXPTR：接收指针寄存器

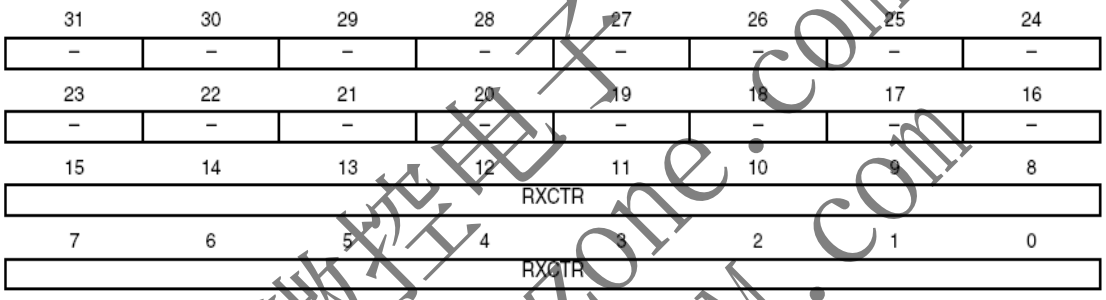
RXPTR 必须设置到接收缓冲地址。

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。

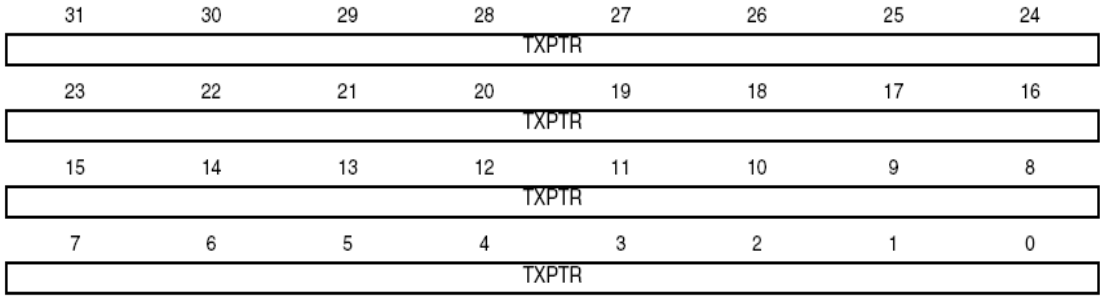
0=停止给接收器的数据传送

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。

23.4.4 发送计数器寄存器

寄存器名称：PERIPH_TCR

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
TXCTR							
7	6	5	4	3	2	1	0
TXCTR							

- 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

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

- 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

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

- 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

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

- 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

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

- 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

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

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：

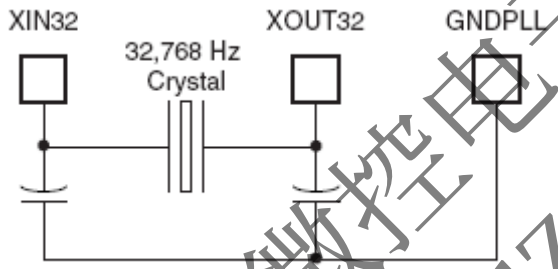
- SLCK，慢时钟，慢时钟是系统内唯一的永久时钟
- MAINCK是主振荡器的输出

  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



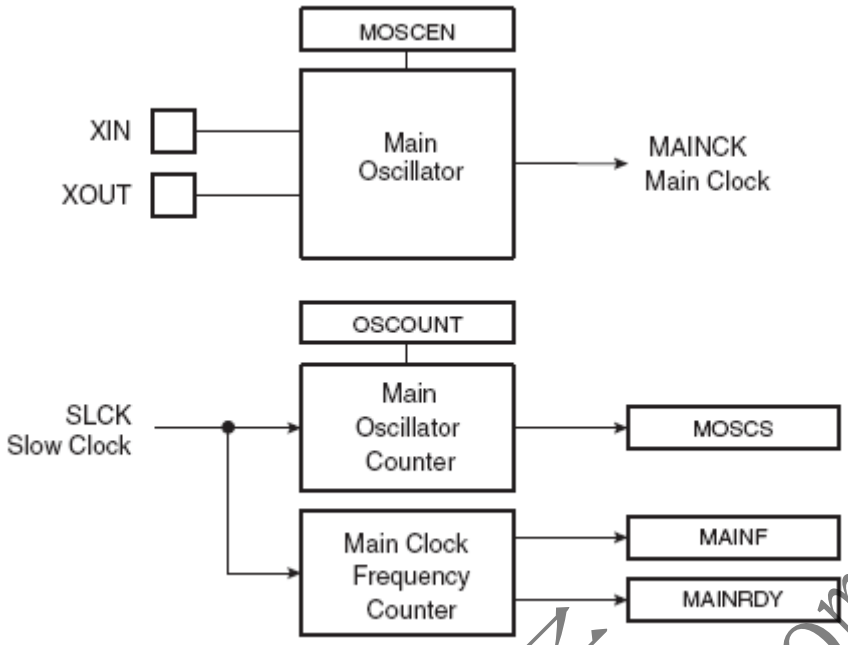
24.3 慢时钟RC振荡器

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

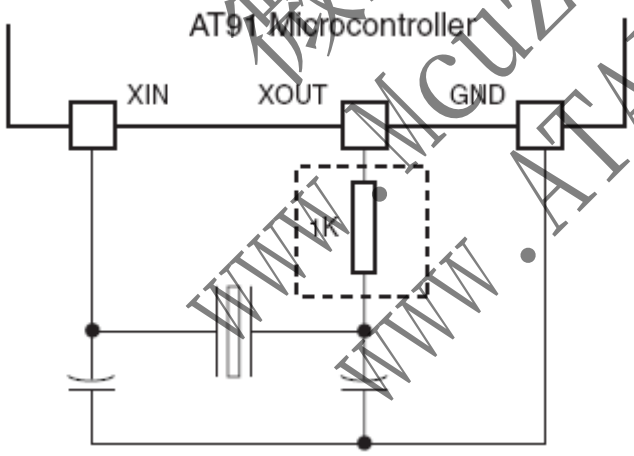
图24-2　主振荡器方块图



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。

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

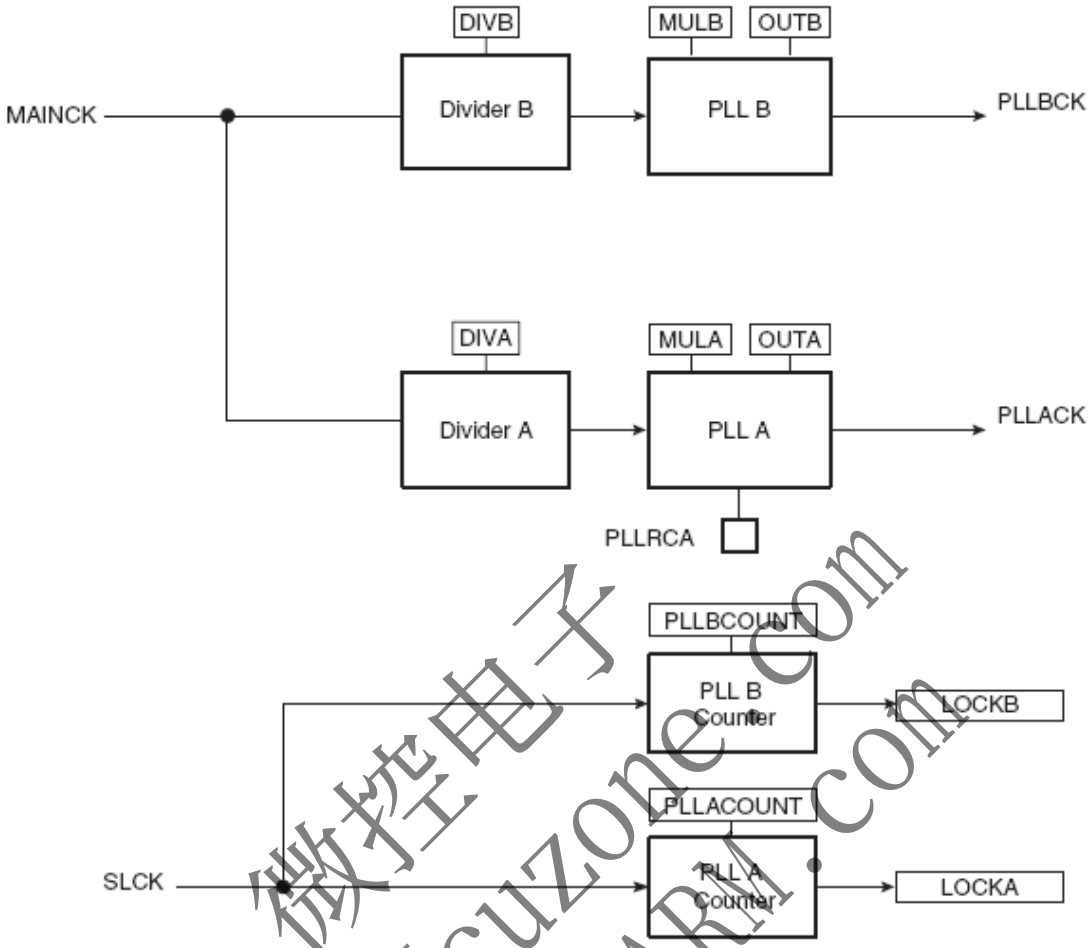
用户可在设备上输入一个时钟代替连接晶体。此情况下，用户必须在XIN引脚上提供外部时钟信号。此条件下XIN引脚的输入特性在产品的电特性段给出。为使外部时钟操作正确，编程者必须确保设置主振荡器寄存器(CKGR_MOR)中的OSCBYPASS位为1，MOSCEN位为0。

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

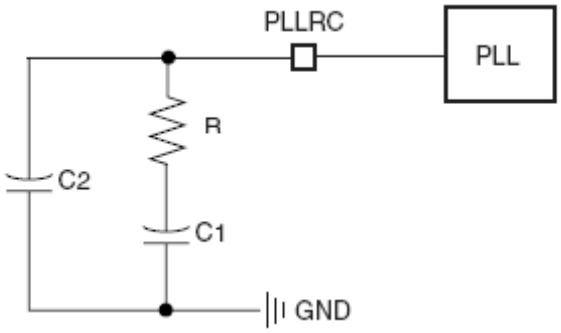
图24-4　分频器和PLL方块图



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，

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。

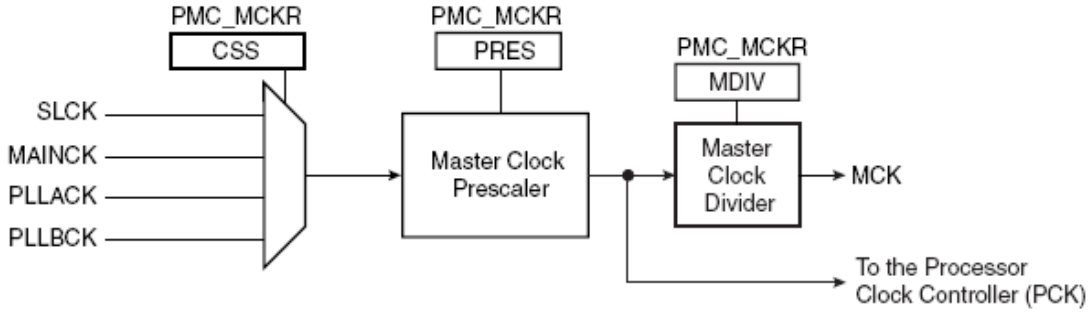
主时钟控制器由时钟选择器和一个预分频器组成。它还包括一个主时钟分频器，此分频器允许处理器时钟比主时钟快。

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。

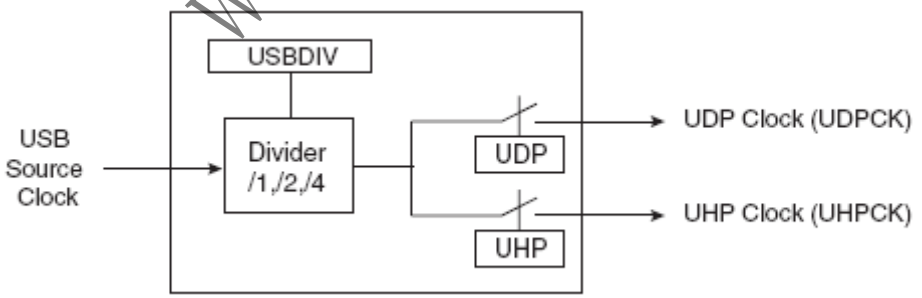
25.4 USB时钟控制器

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。

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

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)
```

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

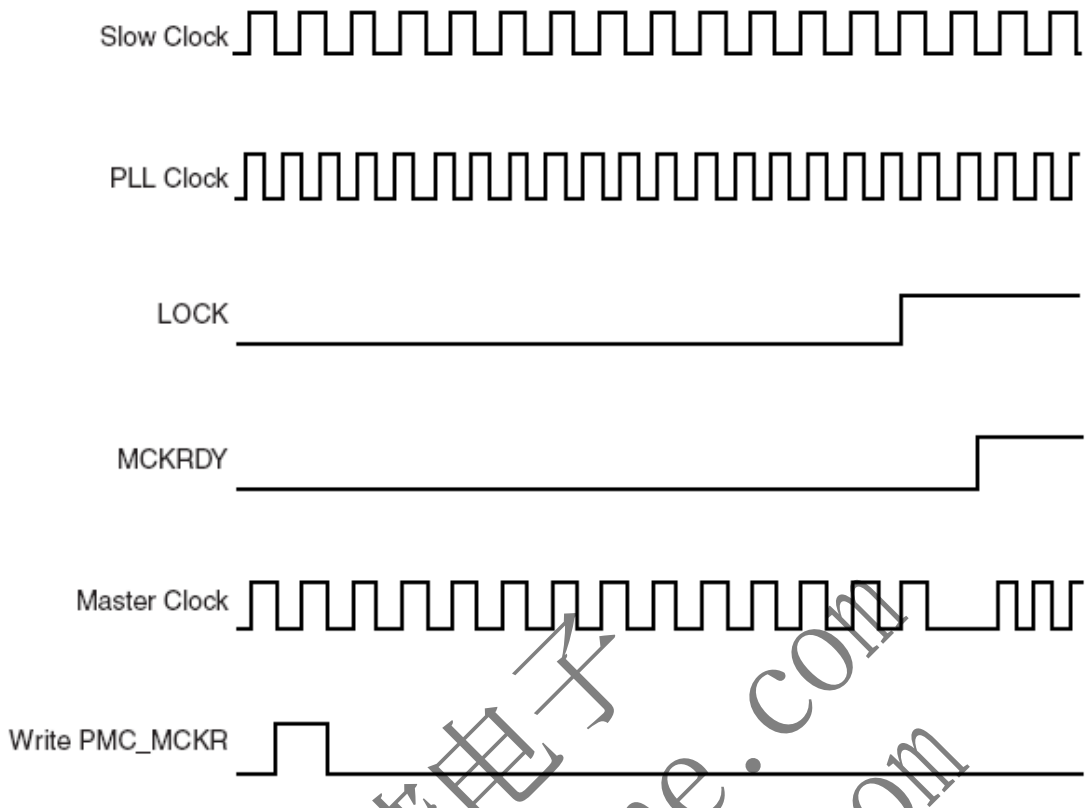


图25-4　主控时钟从主振荡器时钟转换到慢时钟

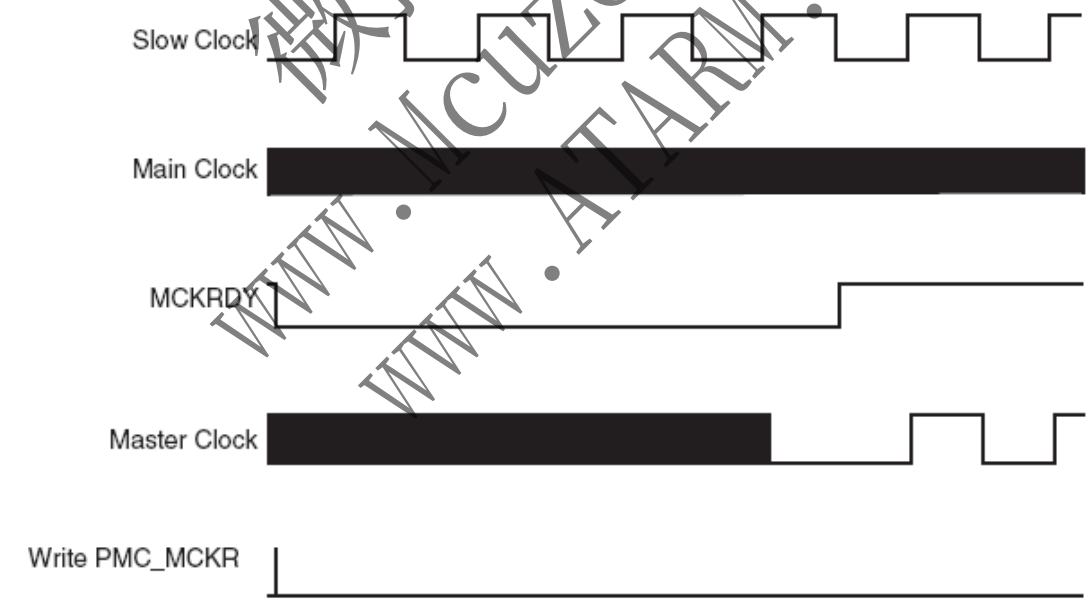


图25.5　改变PLLA编程

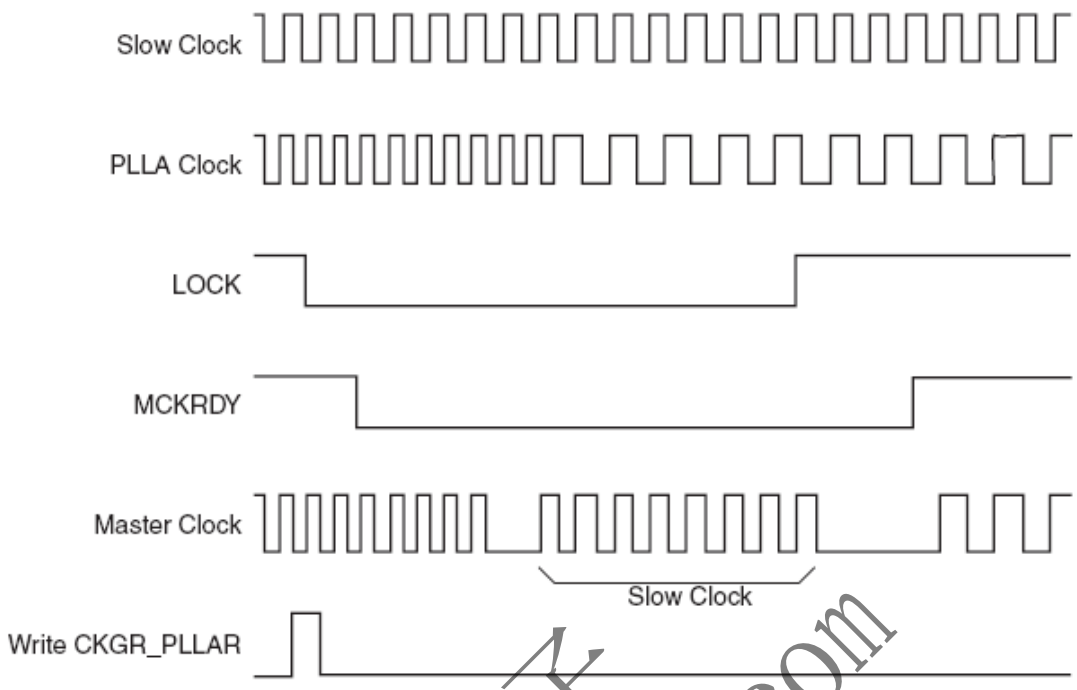


图25-6　改变PLLB编程

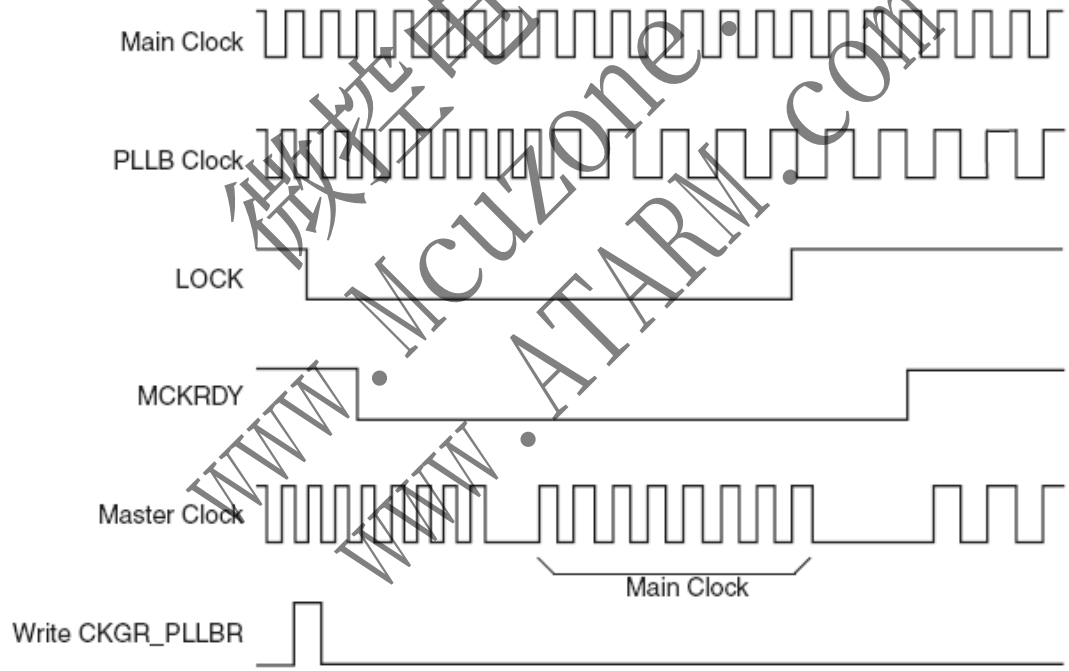
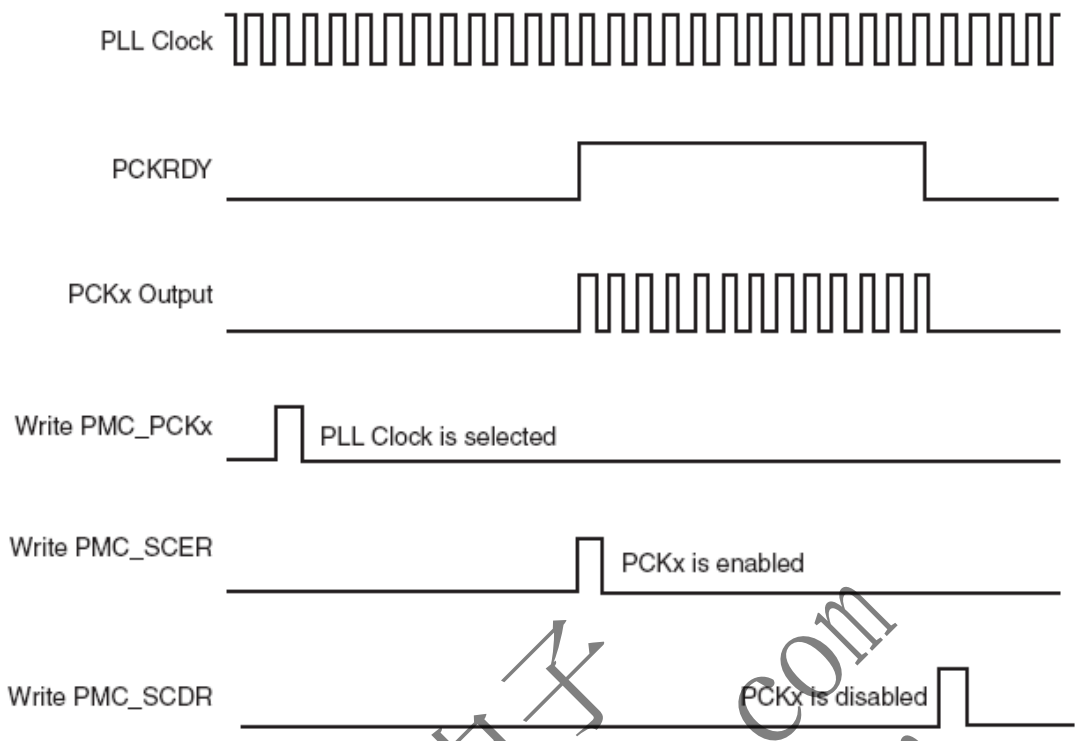


图25-7　可编程时钟输出编程



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

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

- 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

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

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

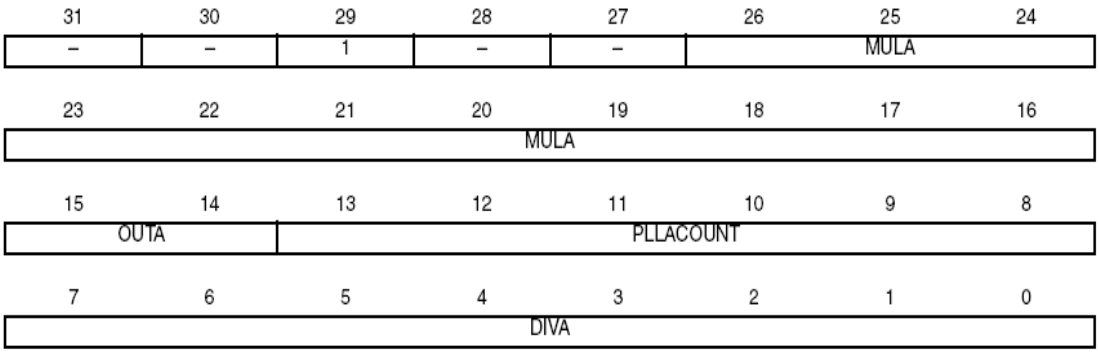
  0=MAINF值无效或主振荡器被禁用

  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



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

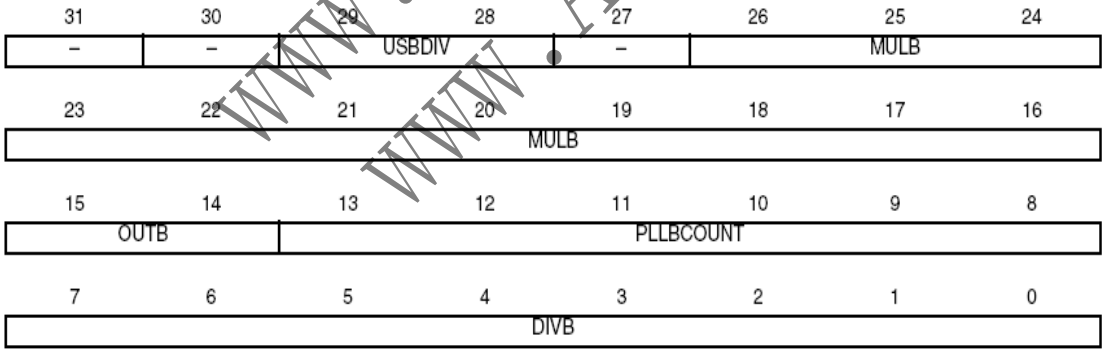
0=PLL A被禁用

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。

- 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

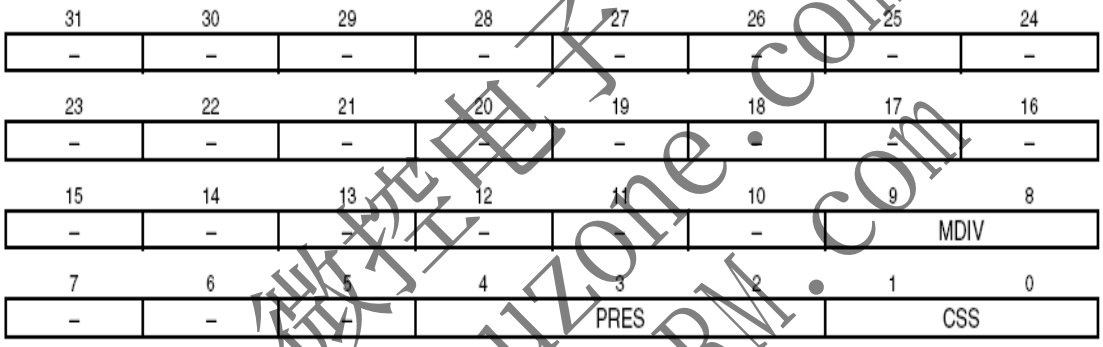
- USBDIV：对USB时钟的分频器

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

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

- 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

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

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。

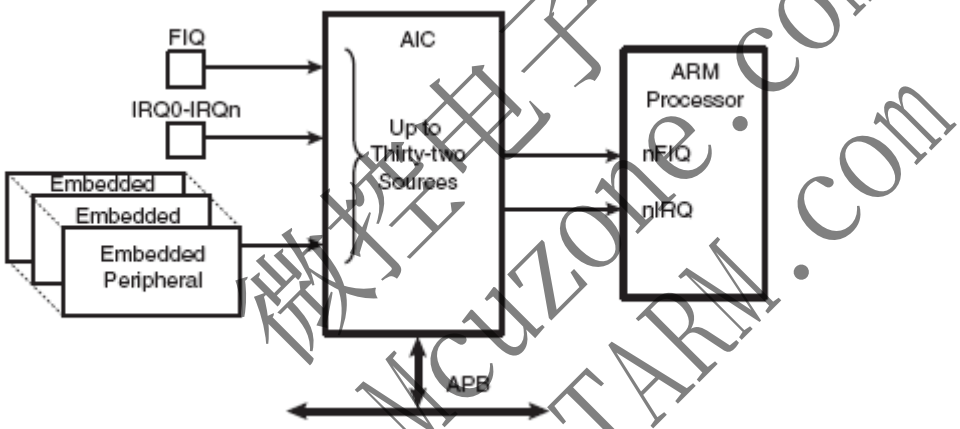
8级优先级控制器允许用户对每个中断源定义优先级。即使一个低优先级中断正在被处理，也允许高优先级中断被服务。

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



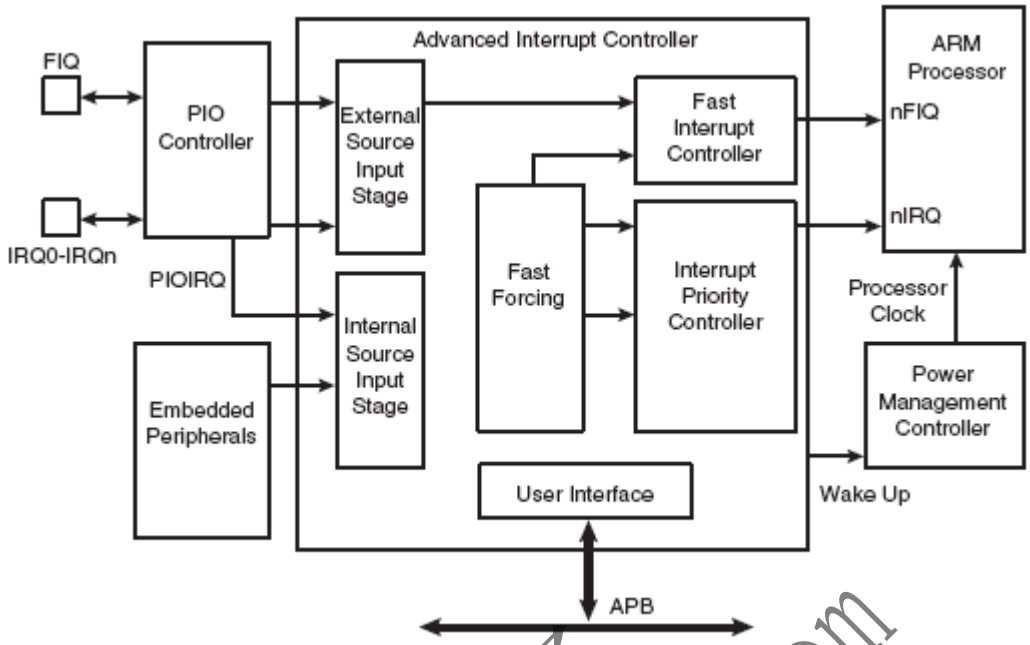
26.3 应用方块图

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



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）。

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

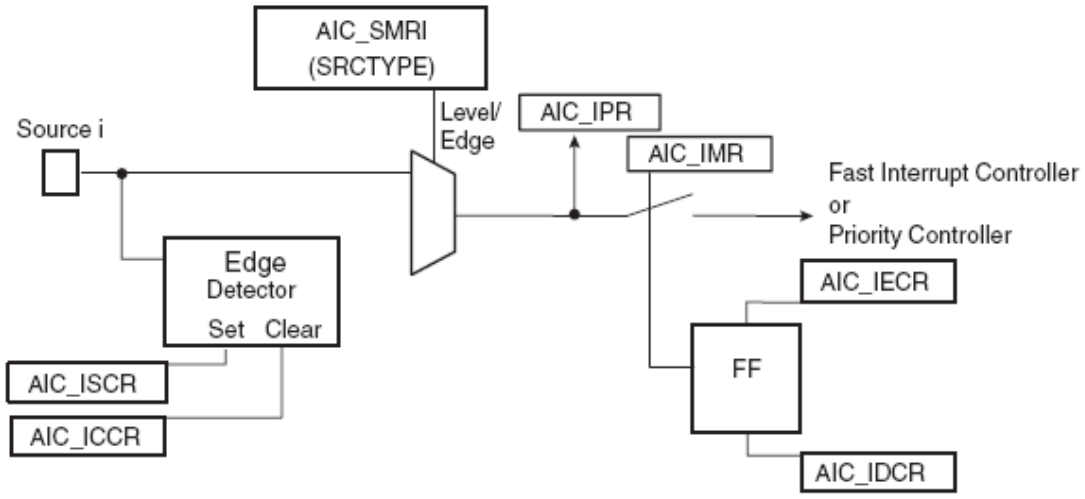
对每个中断，AIC操作发自AIC_IPR（中断待定寄存器）和其AIC_IMR（中断屏蔽寄存器）中的屏蔽码。AIC_IPR反映中断源的实际的活动情况，而无论其被使能与否。

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。

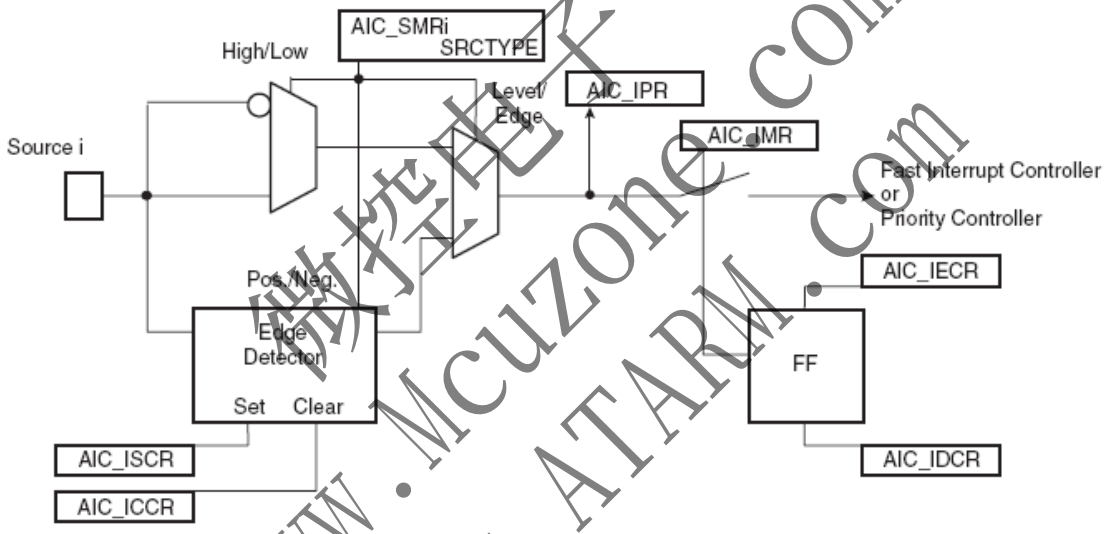
26.7.1.5 内部中断源输入

图26-4 内部终端源输入



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：

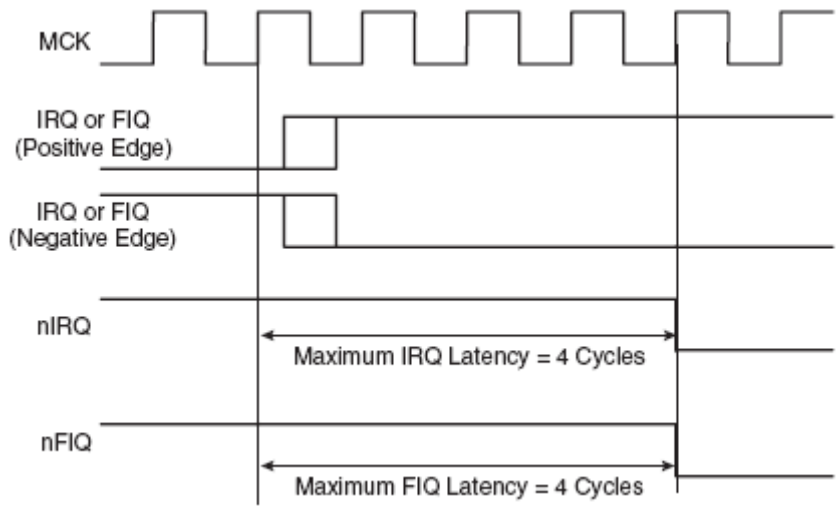
1. 软件屏蔽中断的时间
2. 处理器级或AIC级的事件
3. 当中断发生时，过程中的指令执行时间
4. 高优先级中断的处理和硬件信号的重新同步

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。

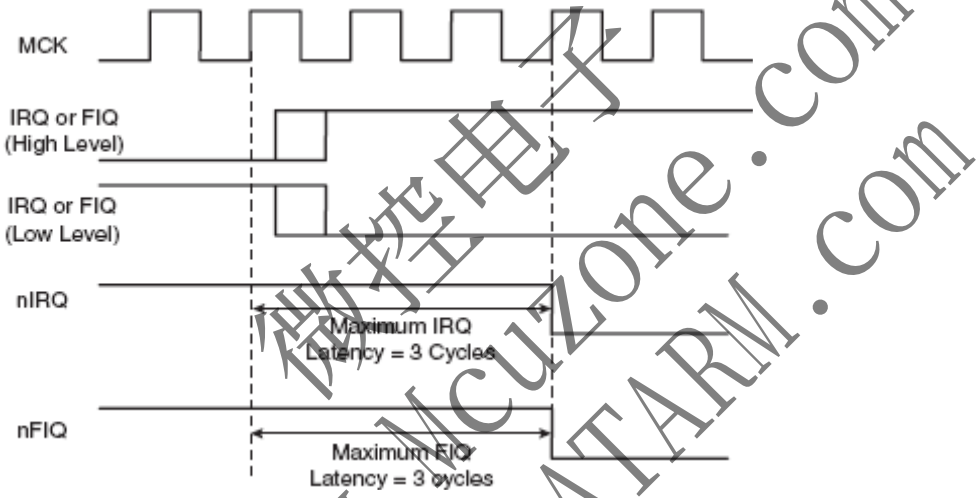
26.7.2.1 外部中断边沿触发源

图26-2 外部中断边沿触发源



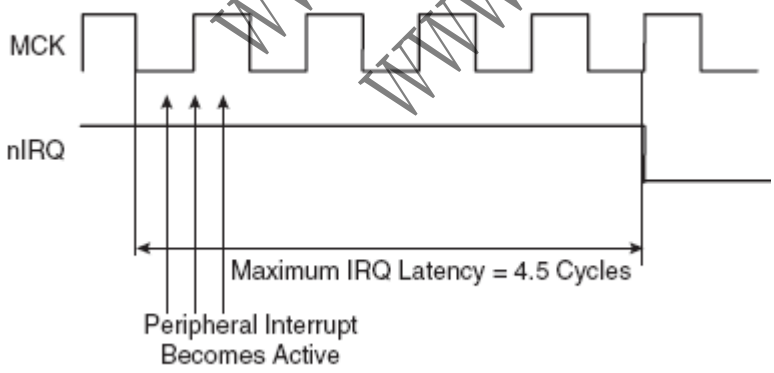
26.7.2.2　外部中断电平敏感中断源

图26-7　外部中断电平敏感中断源



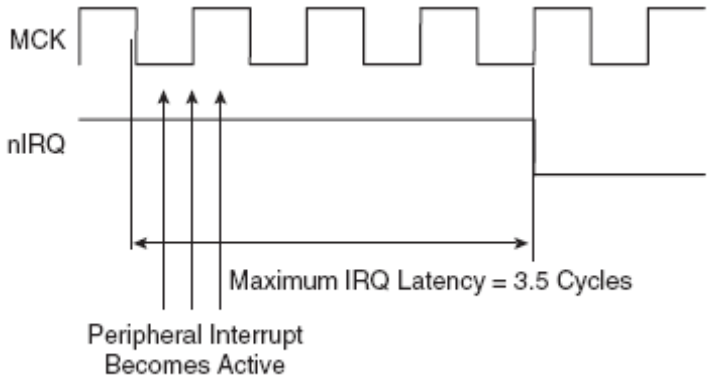
26.7.2.3　内部中断边沿触发源

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



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。

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：

```
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。

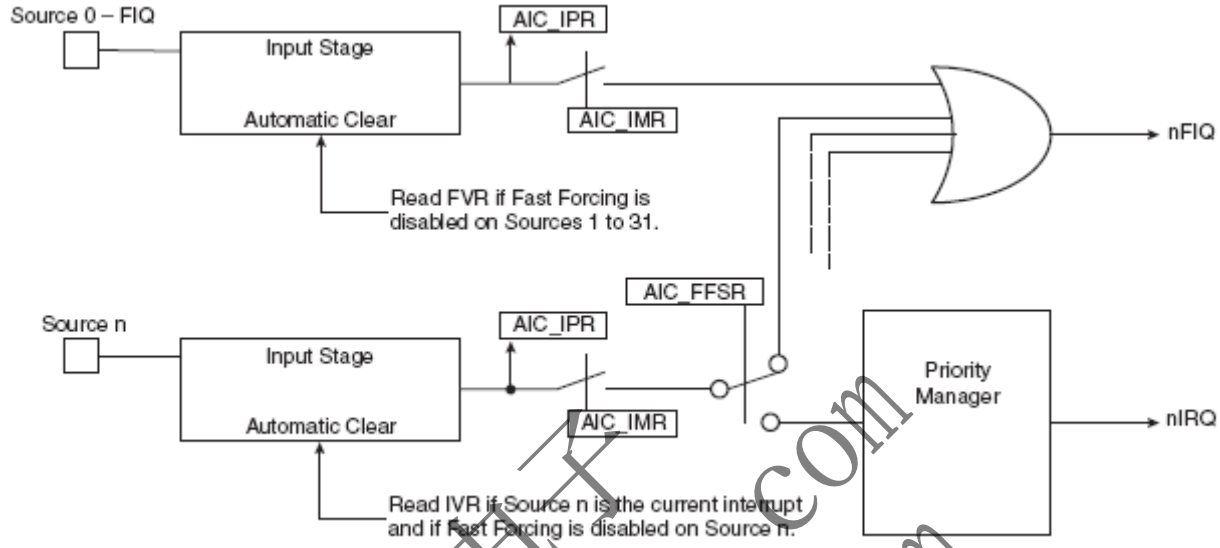
所有使能了快速强制功能并且被编程为边沿触发模式的处于等待状态的中断

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补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

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

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。

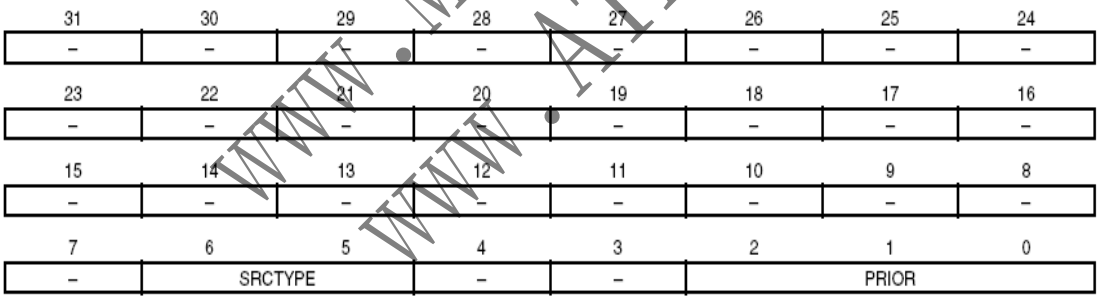
2．PID2...PID31位域提到的标识符如定义在产品数据手册的外设标识符段。

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

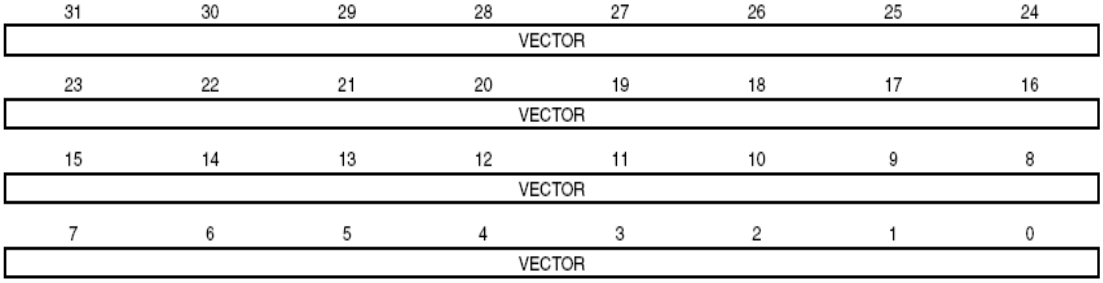
1	1	正边沿触发	正边沿触发

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

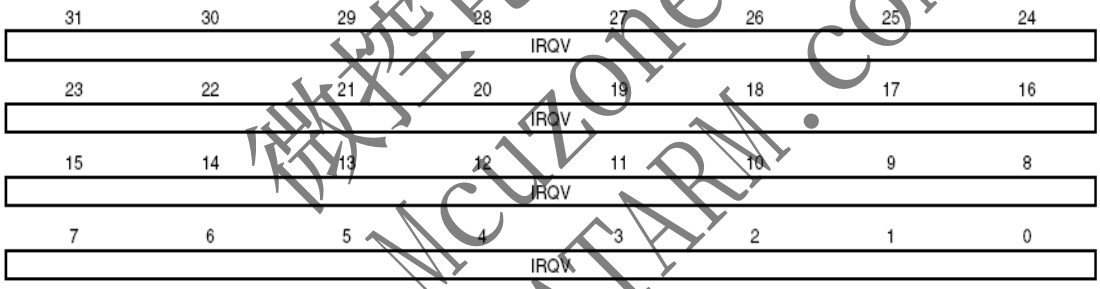
用户可以存储每个中断源的对应处理程序的地址到这些寄存器。

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。

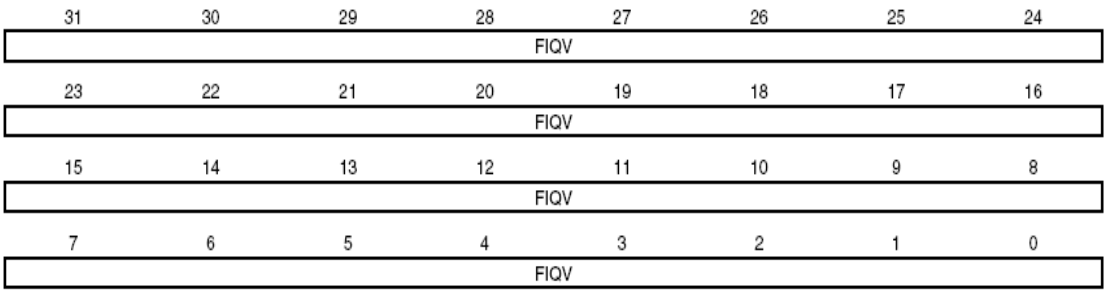
当中断向量寄存器被读取，将使用当前中断号索引源向量寄存器。当无当前中断时，中断向量寄存器读取存储在AIC_SPU中的值。

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

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

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

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

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

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	–

- 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