计算机模块

VDMC0003xP197xx5C01-计算机模块

用户手册

（版本：V1.4）

目录

1. 特征

- 本体（不含引脚）尺寸为：42mm×42mm×18mm;
- 总质量为：65±5g;
- SRAM: 256K×40bit;
- CPU:4 核心;
- FLASH: 4096K×40bit;
- 主频：400MHz；
- DSU:1 路;
- UART:2 路;
- JTAG:1 路；
- 1553B：1 路
- SPACE WIRE:4 路；
- CAN:2 路;
- SPI:1 路；
- I2C:1 路；
- 功耗：小于 5W;
- 封装：PGA197;
- 工作温度为：0℃~70℃（工程样片），-40℃~85℃（工业级和宇航级）；
- 贮存温度为：-55℃∽125℃；
- 抗单粒子闩锁效应性能： 75 MeV·cm2/mg；
- 抗电离总剂量效应性能：50 Krad(Si)；

2. 概述

计算机模块 VDMC0003xP197xx5C01-计算机模块（以下简称计算机模块）内部包含一个 4 核微处理器（计算机模块 ）、3 个 SRAM 芯片(256K×40bit)、3 个 Nor FLASH 芯片（4096K×40bit）、1 个 1553B 驱动芯片、2 个时钟芯片和阻容滤波电路等,其独特的结构,特别适用于严酷的电子应用环境。

3. 功能框图

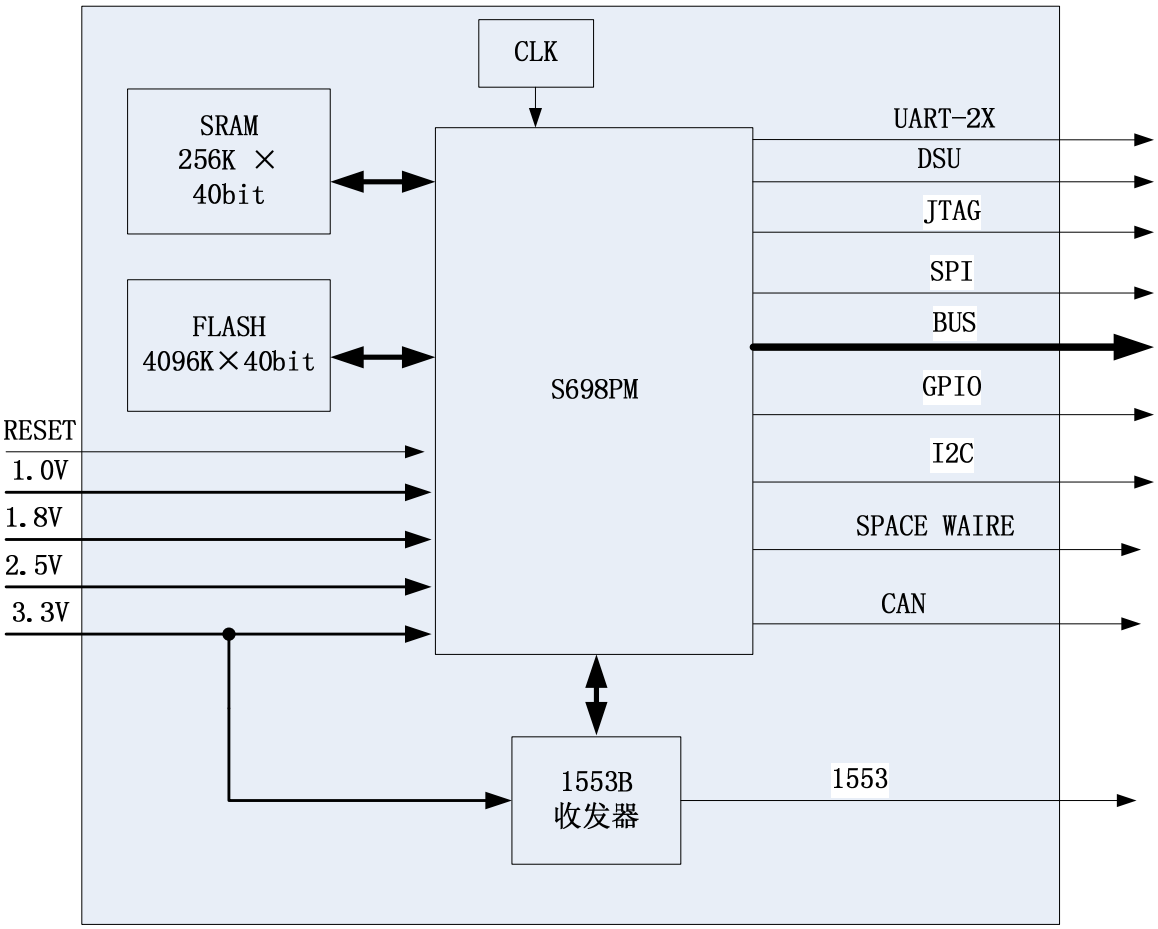


图 3-1 计算机模块功能框图

4. 计算机模块初始化

4.1 上电顺序或复位初始化过程

当系统给计算机模块上电，计算机模块便进入上电初始化过程。计算机模块处于被正常供电状态时，若计算机模块的外引脚 RESETN 连续保持 3 个外部时钟 CLOCK1 周期宽度的低电平，计算机模块便进入复位初始化过程。上电初始化和复位初始化过程相同。计算机模块的上电顺序或复位初始化过程如下图所示。

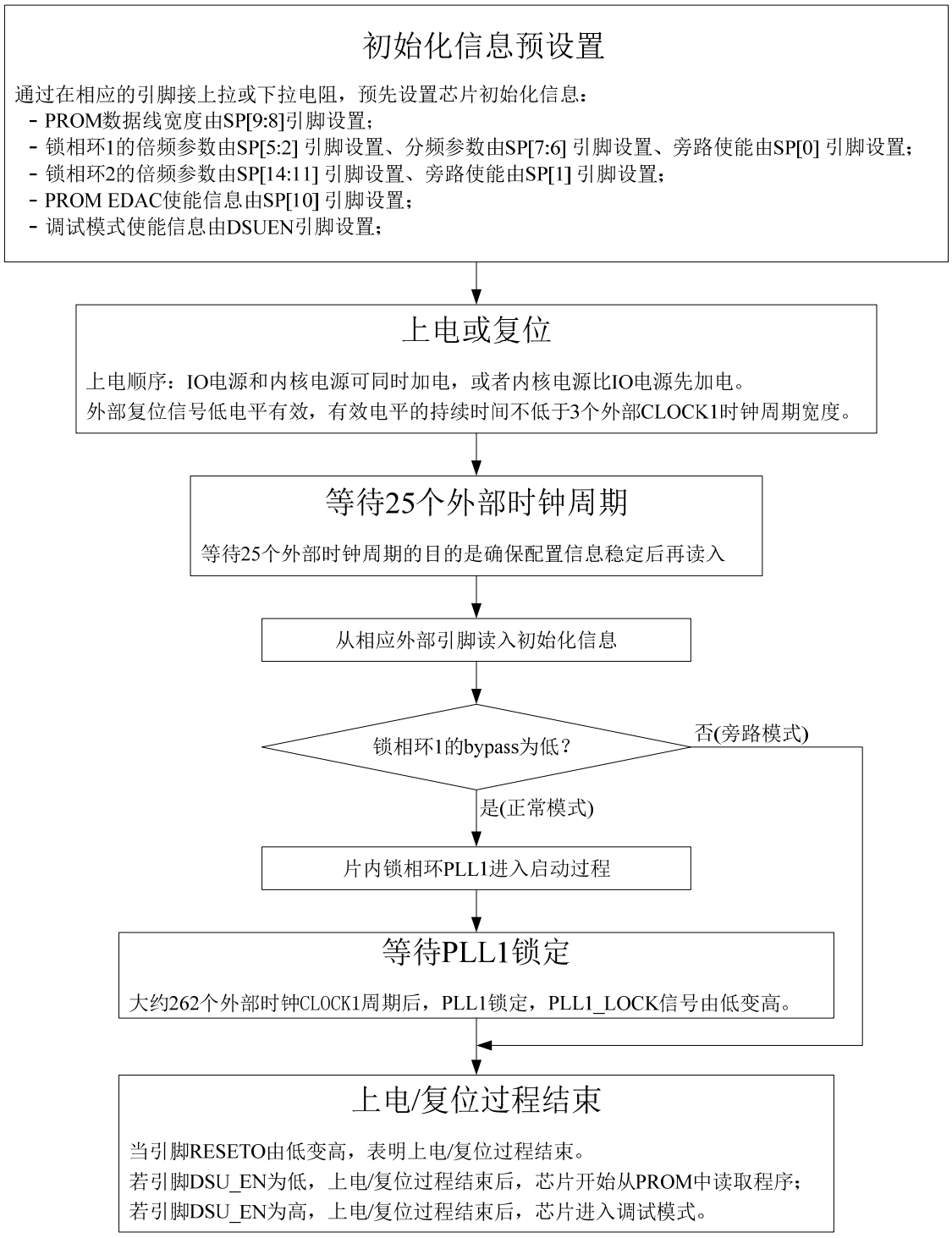


图 4-1 计算机模块的复位及启动过程图

4.2 上电或复位需初始化参数配置

计算机模块的上电或复位初始化配置参数，由初始化过程时间段内，相应引脚的电平状态确定，系统设计时，可以通过在相应的引脚上接上拉或下拉电阻方式，来设置电平状态。具体如表 4-1 所示。

表 4-1　计算机模块上电或复位需配置信号

序号	引脚名	描述
1	DSUEN	芯片的调试模式使能，高电平有效。
2	SP[0]	时钟及复位信号发生器中的锁相环 1（PLL1）旁路使能，高电平有效。
3	SP[1]	时钟及复位信号发生器中的锁相环 2（PLL2）旁路使能，高电平有效。
4	SP[2:5]	时钟及复位信号发生器中的锁相环 1（PLL1）倍频参数。
5	SP[6:7]	时钟及复位信号发生器中的锁相环 1（PLL1）分频参数。
6	SP[8:9]	PROM 数据总线宽度选择： 00：8-bit、01：16-bit、10 或 11：32-bit。
7	SP[10]	PROM EDAC功能使能，高电平有效。
8	SP[11:14]	时钟及复位信号发生器中的锁相环 2（PLL2）倍频参数。

5. 时钟信号发生模块

5.1 片内时钟域

计算机模块结构复杂，外设众多，不同的外设所需的驱动时钟信号也不尽相同。计算机模块时钟域如表 5-1 所述。表 5-2 中详细列举了计算机模块与时钟相关的引脚信号。

表 5-1　计算机模块内部时钟域

#	时钟域	描述	用途	时钟来源
1	IU_CLK	内核时钟	供处理器核心使用	由 PLL1 产生
2	SYS_CLK	系统时钟	供部分片内外设用	由 PLL1 产生

#	时钟域	描述	用途	时钟来源
3	DDR2_CLK	DDR2 时钟	供 DDR2 控制器用	由 PLL1 产生
4	SPW_CLK	SPW 时钟	供 SpaceWire 总线控制器用	由 PLL2 产生
5	1553_CLK1	1553B 时钟 1	供 1553B 总线控制器 1 用	由片外直接输入
6	1553_CLK2	1553B 时钟 2	供 1553B 总线控制器 2 用	由片外直接输入
7	ETH_CLK	以太网时钟	供以太网控制器用	由片外直接输入
8	USB_CLK	USB 时钟	供 USB 控制器用	由片外直接输入
9	JTAG_CLK	JTAG 时钟	供 JTAG 控制器用	由片外直接输入
10	TM_CLK	遥测时钟	供遥测模块用	由片外直接输入

表 5-2　计算机模块时钟相关的引脚信号

#	引脚信号	方向	描述	建议频率
1	CLOCK1	IN	PLL1 的参考时钟输入，接片外有源晶振。	10~60MHz
2	CLOCK2	IN	PLL2 的参考时钟输入，接片外有源晶振。	10~20MHz
3	M1553_CLK	IN	1553B 通道 1 时钟输入，接片外有源晶振。	16MHz@1Mbps 速率模式 80MHz@10Mbps 速率模式
4	SP[21]	IN	1553B 通道 2 时钟输入，接片外有源晶振。	16MHz@1Mbps 速率模式 80MHz@10Mbps 速率模式
5	JTAG_TCK	IN	JTAG 时钟输入	10MHz
6	SP[47]	IN	遥测时钟输入	/
7	ETH_MII_RX_CLK	IN	以太网 RX 时钟，由外部 PHY 芯片提供。	25MHz
8	ETH_MII_TX_CLK	IN	以太网 TX 时钟，由外部 PHY 芯片提供。	25MHz
9	USB_XTALIN	IN	USB 时钟，接外部无源晶振。	24MHz 或 12MHz
10	USB_XTALOUT	OUT		
11	SYSCLK	OUT	系统时钟输出	/

5.2 时钟信号发生模块

计算机模块的时钟信号发生模块主要用来产生片内所需的各种时钟信号，其结构如图 5-所示。时钟信号发生模块内包含两个锁相环 PLL1 和 PLL2，其中 PLL1 主要产生内核时钟 IU_CLK、系统时钟 SYS_CLK 和 DDR2 时钟 DDR2_CLK，PLL2 主要产生 SPW 时钟 SPW_CLK。

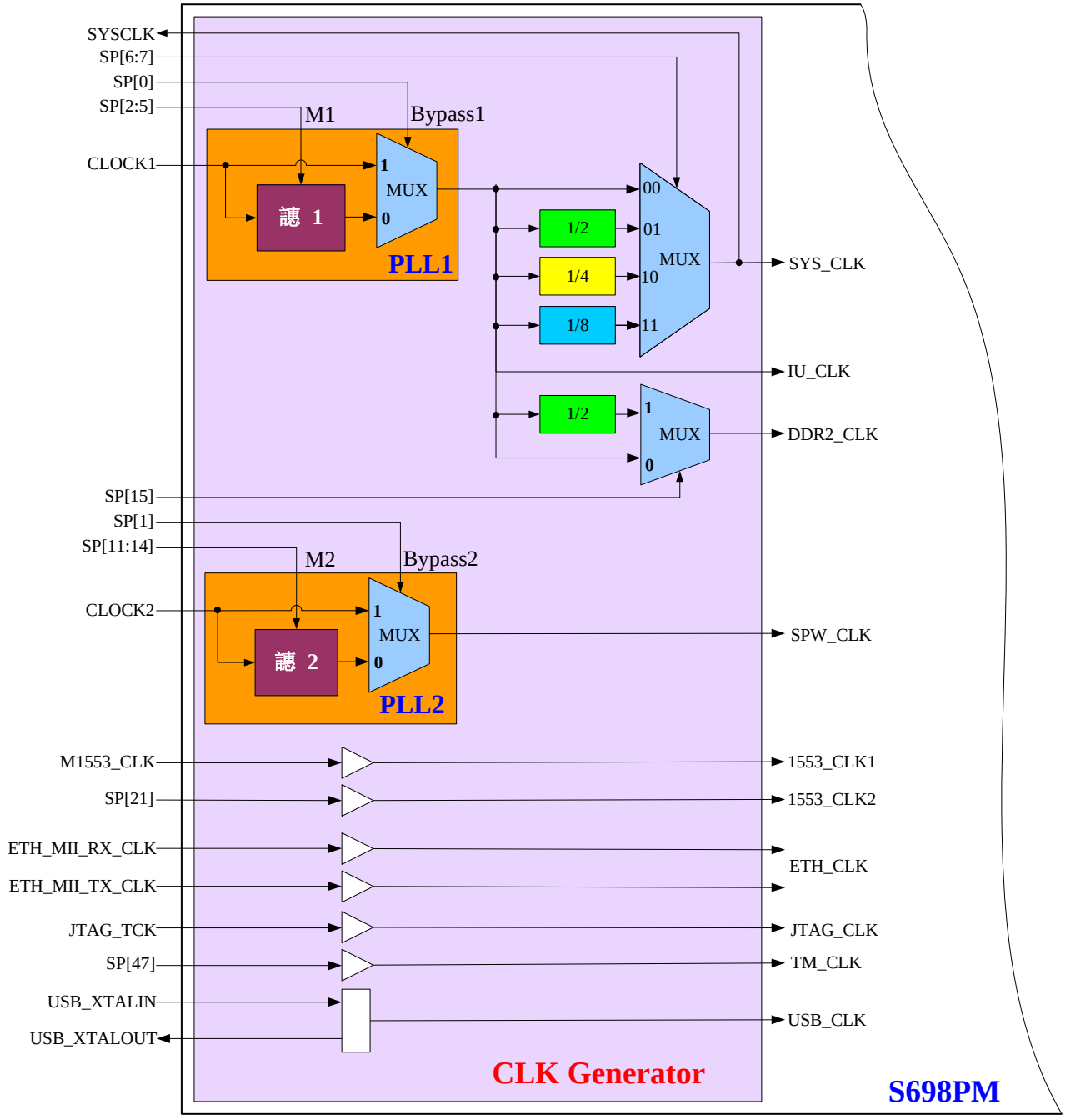


图 5-1 时钟分频结构图

参考图 5-，PLL1 的工作原理说明如下：

(1) PLL1 参考时钟由引脚 CLOCK1 输入，频率范围为 10~60MHz；

(2) PLL1 的倍频参数由引脚 SP[2:5]设定，倍频参数 M1 的计算公式为：

$$M1 = 10 + SP[2:5] \quad \text{（式 3-1）}$$

由式 3-1 可知，即当 SP[2:5] = 0001，M1=11；当 SP[2:5] = 0010，M1=12；当 SP[2:5] = 0011，M1=13……当 SP[2:5] = 1111，M1=25，由此可知，PLL1 的倍频参数 M1 范围为 10 ~ 25。

(3) PLL1 的旁路（Bypass）参数由引脚 SP[0]设定：

- 当 SP[0]=1，PLL1 处于旁路工作模式，输出时钟频率与参考时钟频率相等，即 $F_{IU\_CLK} = F_{CLOCK1}$；
- 当 SP[0]=0，PLL1 处于正常工作模式，输出时钟频率为参考时钟频率的 M1 倍，即 $F_{IU\_CLK} = M1 \times F_{CLOCK1}$；

(4) SYS_CLK 以 IU_CLK 为时钟源，通过 IU_CLK 分频得来，该分频参数 D1 由引脚 SP[6:7]设定：

- 当 SP[6:7]=“00”， D1=1，SYS_CLK 与 IU_CLK 同频率，即：

$$F_{SYS\_CLK} = F_{IU\_CLK} \quad \text{（式 3-2）}$$

- 当 SP[6:7] =“01”， D1=2，SYS_CLK 为的 IU_CLK 二分频，即：

$$F_{SYS\_CLK} = 1/2 \times F_{IU\_CLK} \quad \text{（式 3-3）}$$

- 当 SP[6:7] =“10”， D1=4，SYS_CLK 为 IU_CLK 的四分频，即：

$$F_{SYS\_CLK} = 1/4 \times F_{IU\_CLK} \quad \text{（式 3-4）}$$

- 当 SP[6:7] =“11”， D1=8，SYS_CLK 为 IU_CLK 的八分频，即：

$$F_{SYS\_CLK} = 1/8 \times F_{IU\_CLK} \quad \text{（式 3-5）}$$

(5) DDR2_CLK 以 IU_CLK 为时钟源，通过 IU_CLK 分频得来，该分频参数由引脚 SP[15]设定：

- 当 SP[15]=0，DDR2_CLK 与 IU_CLK 同频率，即：

$$F_{DDR2\_CLK} = F_{IU\_CLK} \quad \text{（式 3-6）}$$

- 当 SP[15]=1，DDR2_CLK 为 IU_CLK 的二分频，即：

$$F_{DDR2\_CLK} = 1/2 \times F_{IU\_CLK} \quad \text{（式 3-7）}$$

参考图 5-，PLL2 的工作原理说明如下：

(1) PLL2 参考时钟由引脚 CLOCK2 输入，频率范围为 10~60MHz；

(2) PLL2 的倍频参数由引脚 SP[11:14]设定，倍频参数 M2 的计算公式为：

$$M2 = 10 + SP[11:14] \quad \text{（式 3-8）}$$

由式 3-8 可知，即当 SP[11:14] = 0001，M2=11；当 SP[11:14] = 0010，M2=12；当 SP[11:14] = 0011，M2=13……当 SP[11:14] = 1111，M2=25，由此可知，PLL2 的倍频参数 M2 范围为 10 ~ 25。

(3) PLL2 的旁路（Bypass）参数由引脚 SP[1]设定：

- 当 SP[1]=1，PLL2 处于旁路工作模式，输出时钟频率与参考时钟频率相等，即 $F_{SPW\_CLK} = F_{CLOCK2}$；

- 当 SP[1]=0，PLL2 处于正常工作模式，输出时钟频率为参考时钟频率的 M2 倍，即 $F_{SPW\_CLK} = M2 \times F_{CLOCK2}$；

由图 5-还可以看出，1553B 时钟 1553_CLK1/1553_CLK2、以太网时钟 ETH_CLK、JTAG 时钟 JTAG_CLK、遥测时钟 TM_CLK 均是通过相关引脚直接从片外输入，再在片内加了驱动后，输送给相应的模块使用。USB 时钟 USB_CLK 是由片外无源晶振与片内振荡驱动电路共同产生，再输送给 USB 控制器使用。

6. 处理器核心

计算机模块内部集成4个相同的高性能处理器核心，每个处理器核心均由32位RISC整型处理单元(IU)、双精度浮点处理单元(FPU)、高速一级缓存(L1 Cache)和存储器管理单元(SRMMU)等组成。

6.1 整型单元 IU

6.1.1 主要特性

计算机模块整数单元IU采用7级流水线处理机制，指令集兼容于SPARC V8 指令集;寄存器窗口为8个，采用循环窗口方式。具有硬件乘除法器，内部实现了容错设计。IU的主要功能是执行整数运算、计算要访问的存储器的地址，和控制FPU指令的执行。

计算机模块整数单元IU具有下列的主要特性：

- 七 7 级的指令流水线：

(1) 预取指令(FE)：如果指令缓存使能，指令预取到指令缓存中。否则，预取被转送给存储器控制器。指令在这一个阶段结束的时候有效和被锁进整数单元；

(2) 译码(DE)：指令被解码，产生调用和分支对象地址；

(3) 寄存器访问(RA)： 操作数从寄存器中或者内在的数据旁路被读出；

(4) 运行(EX)：运行累加器，逻辑和移位操作；

(5) 存储器(ME)：数据缓存被访问，在执行阶段中读出要存储的数据，同时写到数据缓存中；

(6) 例外与异常 (XC)： 处理陷阱 Trap 和中断；

(7) 回写 (WR)： 任何累加器、逻辑、移位、或缓存操作的结果被回写到寄存器；

- 独立的指令和数据缓存接口；
- 支持 8 个寄存器窗口；
- 具有 16x16 位 MAC 和 40 位累加器的硬件乘法器；
- 具有指数为 2 的除法器；
- 采用单一矢量陷陷阱（Trapping），以减少的代码数量；
- 触发器采用三模冗余 TMR 设计，提高抗击单粒子效应的能力；
- 存储器模块采用 EDAC 保护措施，提高抗击单粒子效应的能力；

6.1.2 指令

计算机模块整数单元IU指令遵循SPARC V8（IEEE-1754）标准，按照功能分为6类：

(1) 存储器存取指令（load/store）

存储器存取指令是唯一用来访问存储器的指令。存储器存取指令用2个‘r’寄存器或者1个‘r’寄存器和1个13位的无符号立即数计算出1个32位、按字节排列的存储器地址，IU再在该地址后面加上“地址空间标志符（ASI）”以决定处理器是处于管理模式还是用户模式，是访问指令存储器还是数据存储器。

存储器存取指令的目标域指定是一个r寄存器，f寄存器，或者是协处理器寄存器（此寄

存器提供存储的数据或取得要载入的数据)。

整数存取指令支持字节方式(8位)、半字方式(16位)、字方式(32位)和双字方式(64位)。整数存取指令包括一些整数取数指令，用来从内存中取得8位或16位的单精度数并放入目的寄存器中。浮点和协处理器存取指令支持字方式和双字方式。

计算机模块对半字的大于一个字节类型的数据采用高地址优先存储的方式。

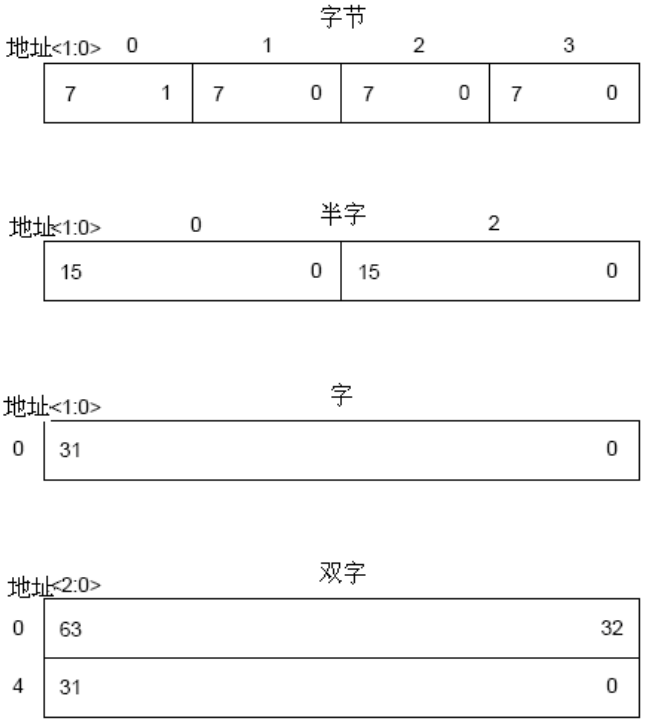


图 6-1 地址转换

(2) 算术运算/逻辑运算/移位指令（arithmetic/logical/shift）

算术运算/逻辑运算/移位指令提供了算术运算、逻辑运算和移位操作。这些指令除“SETHI”指令之外，都包括2个操作码，并由这2个操作码运算产生一个结果，这个结果或者放入目的寄存器中，或者丢弃。“SETHI”指令是一条专门的指令，用来和它后面的指令一起创建一个32位的常数并放入r寄存器中。

移位指令用来把‘r’寄存器中的内容右移或左移‘n’位，‘n’的值由指令中的常数或’r’寄存器中的值指定。

整数乘法指令提供有符号或无符号的32位×32位=64位操作。整数除法指令提供有符号或无符号的 64位÷32位=32位操作。整数乘法指令和整数除法指令的结果会影响“PSR”的相应

位。被‘0’整除会产生一个“陷阱”。

(3) 跳转控制指令 （control transfer）

跳转控制指令（CTIs）包括与PC指针相关的分支指令和调用指令、寄存器间接跳转指令和条件陷阱s。大多数跳转控制指令是延时的跳转控制指令（DCTIs）。跟在延时跳转控制指令后面的指令会在此跳转完成之前执行。

跟在跳转控制指令后的指令叫延时指令。延时指令总是预取的，即使跳转控制指令是无条件分支。然而，如果跳转控制指令不被执行，跳转控制指令的某一位可以使延时指令也不被执行。

(4) 读/写状态寄存器指令

读/写寄存器指令用来读/写用户可见的状态寄存器，也可读/写辅助状态寄存器。

(5) 浮点运算指令（floating-point）

浮点运算指令用来完成所有的浮点运算。浮点运算指令是寄存器到寄存器的指令，浮点运算操作利用浮点寄存器进行。同算术运算指令/逻辑运算指令和移位指令一样，浮点运算指令有1或2个源操作数，并得出一个结果。

(6) 辅助/杂项指令（miscellaneous）

其它的一些辅助或杂项功能的指令。

计算机模块IU的大部分指令为单周期，也有极少数指令是多周期指令。表 6-1列出了指令的周期。

表 6-1 计算机模块指令周期

序号	指令	周期(单位：时钟周期)
1	跳转类 (JMPL,RETT)	3
2	装载/存贮 (LOAD/STORE)	5
3	陷阱(Taken Trap)	5

序号	指令	周期(单位：时钟周期)
4	乘法(SMUL/UMUL)	1
5	除法(SDIV/UDIV)	35
6	其它指令 (All other instructions)	1

6.1.3 寄存器堆（Register File）

6.1.3.1 通用目的寄存器(r register)

计算机模块的整型单元的寄存器堆（Register File）包含136个32-bit的通用目的寄存器（即general purpose register，简称r register或r寄存器）。这136个r寄存器被分成8个全局寄存器（global register）、8个“16-寄存器组”(即每个16-寄存器组中含16个r寄存器)，每个16-寄存器组又进一步的分成8个输入寄存器（即in寄存器）和8个本地寄存器（即local寄存器）。

6.1.3.2 寄存器窗口

寄存器窗口（register window）是一个24-寄存器组，它包括一个16-寄存器组（由8个in 寄存器和8个local寄存器组成）、相邻的16-寄存器组中的8个in 寄存器（相邻16-寄存器组中的8个in 寄存器被当前寄存器窗口设定为自己的8个out寄存器）。

计算机模块共包括8个寄存器窗口，处理器运行时将按照0到7的顺序依次切换窗口。在特定的时间内，指令可以访问136个r寄存器中的8个global寄存器和一个寄存器窗口，即计算机模块的指令可以访问32个通用寄存器r[0]~r[31]，这些寄存器对用户可见。(见表 6-2)。

表 6-2 计算机模块的指令可以访问 32 个通用寄存器

序号	寄存器类别	寄存器	寄存器地址	备注	
1	global全局	%g0-%g7	r[0] - r[7]	/	/
2	out输出	%o0-%o7	r[8] - r[15]	/	组成一个寄存器窗口
3	local本地	%l0-%l7	r[16] - r[23]	组成一个16-寄存器组	
4	in输入	%i0-%i7	r[24] - r[31]		

处理器当前访问的寄存器窗口由处理器状态寄存器PSR中的Current Window Pointer (CWP)区（5 bits）设定，当执行“RESTORE（或RETT)”指令时，CWP加1；当执行“SAVE”指令或产生一个trap时，CWP的值减1。寄存器窗口上溢和下溢由Window Invalid Mask(WIM)寄存器监控，WIM由管理软件控制。寄存器窗口实际的数目对用户应用程序是透明的。

6.1.3.3 寄存器窗口的重叠

计算机模块通用窗口寄存器结构，如图6-所示。在程序运行时，寄存器窗口中保存着当前进程的状态（STATUS）信息。在任一时刻,程序只使用一组寄存器窗口，当发生函数调用或返回时(叶过程除外),处理器会在不同的寄存器窗口间移动(SHIFT),以保存当前上下文环境。

计算机模块的每个寄存器窗口都和与之相邻的2个寄存器窗口共享in寄存器和out寄存器。图6-中有8个寄存器窗口,分别是从 0 到 7(用 w0到w7为标记),每个窗口对应了24个寄存器,其中的16个与相邻窗口共用。这些寄存器窗口以循环圈的方式连接起来,寄存器窗口0与寄存器窗口7连接起来。进程在不同的寄存器窗口间移动(SHIFT)，处理器执行RESTORE和SAVE指令时，将导致窗口切换，CWP指针用来指向当前正在使用的窗口。

特权模式下的 RETT 指令(return from trap) 和trap 事件(中断，异常或TRAP指令)也将导致窗口切换。CWP+1寄存器窗口的out寄存器被设定为当前寄存器窗口的in寄存器；当前寄存器窗口的out寄存器则被设定为CWP-1寄存器窗口的in寄存器。Local寄存器对于每个寄存器窗口来说都是唯一的。一个地址设定为“o”的r寄存器（其中8≤o≤15）和地址为“o+16”的r寄存器被认为是同一个寄存器。同样的，一个地址设定为“i”的r寄存器（8≤i≤15）和地址为“(i+16)”的r寄存器也被认为是同一个寄存器。

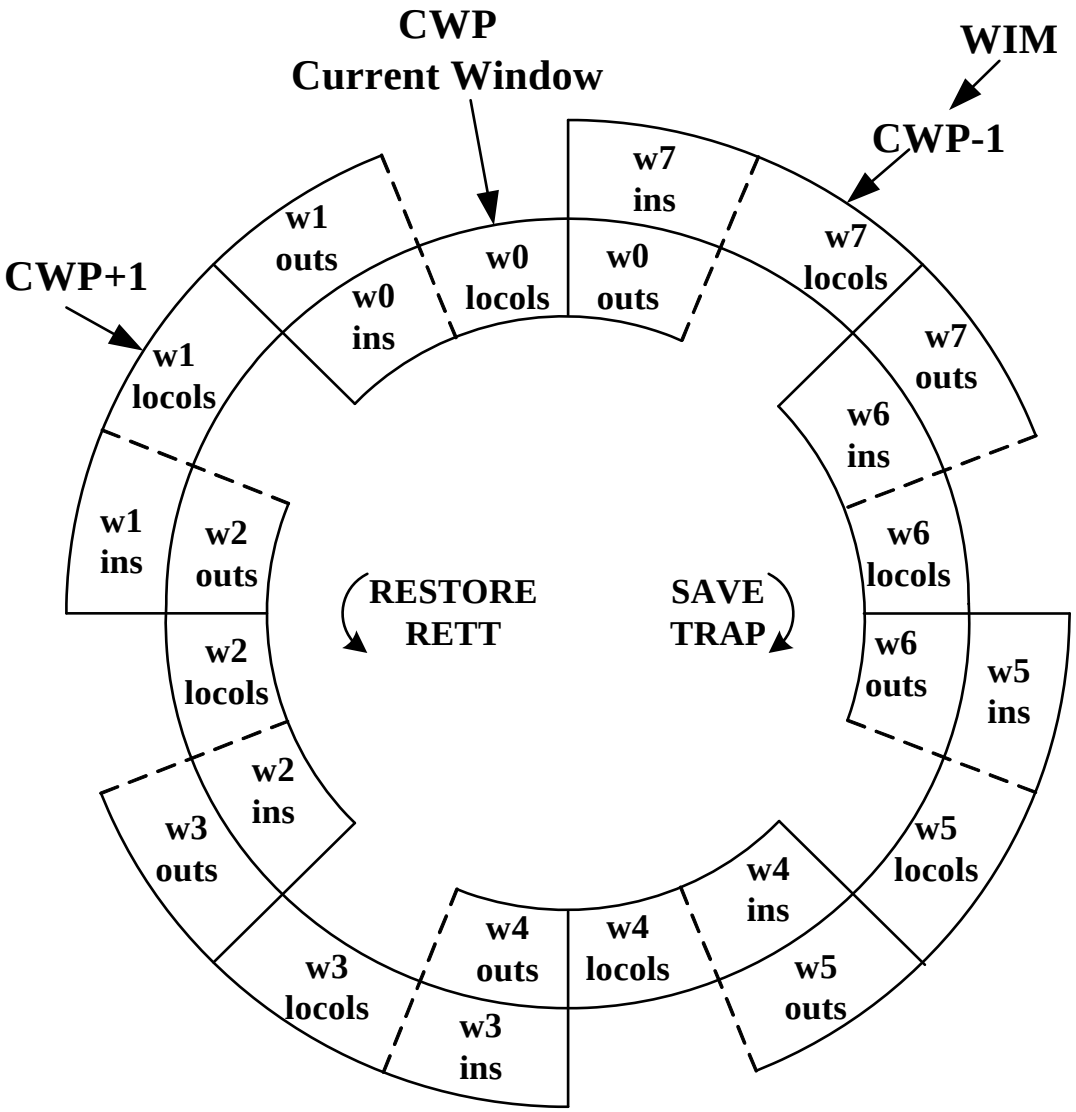


图 6-2 计算机模块窗口的寄存器结构

图6-的右上角的"WIM" 寄存器用来表示每个窗口的使用状态,每个窗口在WIM中对应1个bit位，如果这一位设置为1表示此窗口已经被使用（有效），设置为0表示此窗口未被使用（无效）。"WIM" 寄存器常被用来判断窗口是否溢出（overflow）或者下溢（underflow）。当寄存器窗口都被使用时，如果再发生一个额外的SAVE指令将会超过窗口容量,此时将触发一个窗口寄存器溢出陷阱；与之相反，当发生RESTORE或RETT指令时，如果"WIM" 寄存器对应位无效，此时将触发一个窗口寄存器下溢陷阱。

6.1.3.4 寄存器堆的编程注意事项

(1) 由于过程调用指令“调用”和“跳转”都不改变CWP的值，所以过程调用

不改变寄存器窗口。

(2) 当通过保存指令重新进入窗口时就不能确定“本地”、“输出”寄存器中的值。由于在恢复和保存之间可能会发生陷阱，执行的程序中若有恢复指令，且其后跟有保存指令，则结果窗口中的“本地”和“输出”寄存器中的数值就不能确定。然而如果陷阱被屏蔽，“本地”和“输出”寄存器中的值就有效了。

(3) 因为寄存器窗口有交迭，所以，软件可以利用的寄存器窗口比IU实际实现的寄存器窗口少1，即NWINDOWS 1。当寄存器集合被填满时，最新寄存器窗口的“输出”寄存器和仍然保留着有效数据的最旧的寄存器窗口的“输入寄存器”是同一个寄存器。

6.1.4 专用寄存器

计算机模块整型单元的控制/状态寄存器包括：

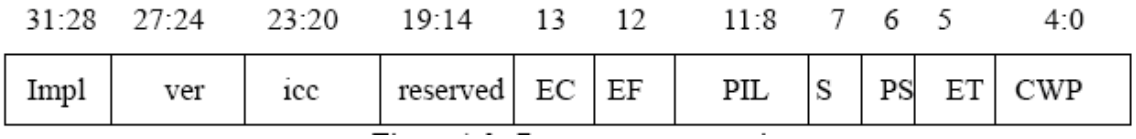
- 处理器状态寄存器(PSR：Processor state Register)：反映并控制处理器的运行状态；
- 窗口无效屏蔽寄存器(WIM：Window Invalid Mask Register)：来查看保存、恢复或返回指令是否会引起窗口上溢（overflow）或下溢（underflow）陷阱；
- 陷阱基址寄存器(TBR：Trap Base Register)：存放陷阱发生时要转向的地址；
- 乘法/除法 (Y)寄存器：在除法中，暂存被除数的高32位有效位；在乘法中，暂存乘积的高32位有效位；
- 程序指针寄存器 (PC/nPC：Program Counter/next Program Counter)：存放当前指令地址和下一条指令的地址；
- 辅助状态寄存器(ASR：Ancillary State Register)：存放监视点Watch point等信息。

6.1.4.1 处理器状态寄存器（PSR）

处理器状态寄存器PSR（Processor State Register）是一个32位的的寄存器，其中包含最近一次指令执行结果信息，该信息可被用来在条件跳转操作中改变程序执行的流程。PSR

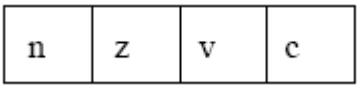
中的状态在ALU操作后被更新。此外，PSR中还包含当前寄存器堆窗口（current window）、中断状态、协处理器状态等信息。

PSR寄存器包含若干个域，这些域控制处理器的操作或保存状态信息。它们可以被保存、恢复、陷阱、或返回指令修改，特权指令RDPSR和WRPSR也可直接读写PSR寄存器。



31:28	27:24	23:20	19:14	13	12	11:8	7	6	5	4:0
Impl	ver	icc	reserved	EC	EF	PIL	S	PS	ET	CWP

- impl 域：这四位为只读位，用来在硬件上确定，分类硬件体系架构的实现。
- ver 域：这四位也为只读位，标记 S698P 的版本号。
- icc 域：这四位标记 IU 的状态，它们可被以字母串 cc 结尾的算术逻辑指令或是 WRPSR 指令修改。Bicc 和陷阱指令可以引起基于此域的控制转移。此域的格式定义如下：



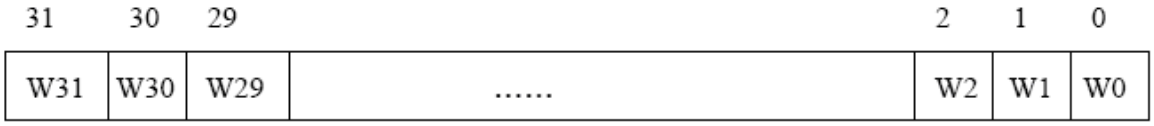
n	z	v	c

  - n 域：此位指示累加器的运算结果是否要被忽略。1＝忽略，0＝不忽略。
  - PSR_zero（z）域：对于累加器的最后一条修改了 icc 域的指令，此位指示是否累加器的运算结果为 0。1＝为零，0＝不为零。
  - v 域：对于累加器的最后一条修改了 icc 域指令，此位指示累加器的结果是否在 32 位能表示的范围内。1＝溢出；0＝无溢出。
  - c 域：对于累加器的最后一条修改了 icc 域的指令，此位指示是否指令的运算的最后一位有借位或是进位。1＝有，0＝没有。
- reserved 域：此六位是保留位。如果被 RDPSR 指令读，则返回 0。为了以后扩展兼容性，管理软件可以通过 WRPSR 向此域写入 0。
- EC 域：此位决定协处理器是否被激活。如果没被激活则会有一个与协处理器相关的陷阱。1＝支持，0＝不支持。如果硬件没实现协处理器，则此位始终为 0，这时对此位的写被忽略。
- EF 域：此位决定浮点单元(FPU)是否被激活。如果没被激活则会有一个与浮点单元(FPU)相关的陷阱。1＝支持，0＝不支持。如果硬件没实现浮点单元(FPU)，则此位始终为 0，这时对此位的写被忽略。

- PIL 域：此域指示 CPU 要接收中断的级别。
- S 域：此为指示 CPU 是处于管理状态还是用户状态。1＝管理状态，0＝用户状态。
- PS 域：此位保存当前陷阱所处的状态值。1＝管理状态，0＝用户状态。
- ET 域：此位指示是否允许有陷阱，如果当前有陷阱，则此陷阱会自动将此位置为 0。0＝忽略陷阱，这时若有中断请求则会被忽略，而且一个异常的陷阱会使 IU 执行挂起（halt）操作（此操作又触发重启陷阱）。1＝允许陷阱。
- CWP 域：此五位为当前寄存器窗口指针。当发生陷阱时或保存指令硬件会使 CWP 减少；恢复或返回指令会 CWP 增加。

6.1.4.2 窗口无效屏蔽寄存器（WIM）

窗口无效屏蔽寄存器（Window Invalid Mask Register）是一个32位的的寄存器。WIM由管理软件控制，被用来查看保存、恢复或返回指令会是否引起窗口上溢（overflow）或下溢（underflow）陷阱。



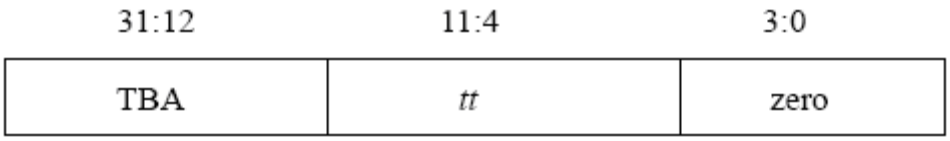
WIM[n]关联CWP为n时对应的窗口。

当保存、恢复或返回指令执行，当前CWP会和WIM比较，如果要转到的是一个非法窗口，相应的WIM位将被置为1，进而将转入相应的窗口上溢（overflow）或下溢（underflow）陷阱。

WIM可被特权指令RDWIM读取，被WRWIM指令写入。

6.1.4.3 陷阱基址寄存器（TBR）

陷阱基址寄存器TBR（Trap Base Register）是一个32位的的寄存器。TBR寄存器有三个域，每个域中都存放陷阱发生时要转向的地址。



➢ TBA 域：这 20 位存放陷阱地址表的高 20 位地址。这些位被管理软件维护，可以被 WRTBR 指令写入。

➢ tt 域：这 8 位存放陷阱的类型编码。当发生陷阱时，这 8 位被硬件写入，且一直保持此值到下一个陷阱到来。WRTBR 指令对此域没有影响。

➢ 0 域：这四位全为 0，不受 WRTBR 指令的影响。这四位留作以后扩展软件使用 WRTBR 时应向这四位写 0。

6.1.4.4 乘法/除法寄存器（Y）

Y寄存器是一个32位的的寄存器，用于乘法和除法中。在除法中，暂存被除数的高32位有效位；在乘法中，暂存乘积的高32位有效位。相应的乘法指令包括SMUL、SMULcc、UMUL、UMULcc乘法指令。Y寄存器也可保存SDIV,SDIVcc,UDIV,UDIVcc等除法指令的双精度数。

Y寄存器可以被RDY，WRY指令读或写。

6.1.4.5 程序指针（PC，nPC）

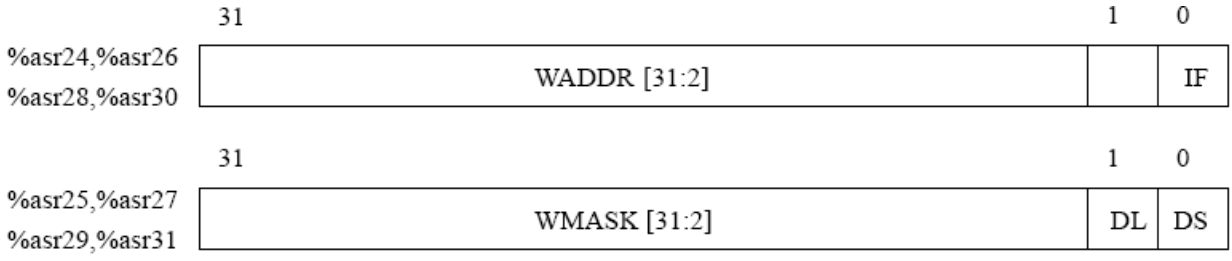
程序指针PC和nPC都是32位的的寄存器。

在正常模式下（即没有TRAP发生），PC中的内容表示的是IU当前执行的指令的地址，nPC中的内容表示的是IU要执行的下一条指令的地址。

在TRAP发生的情况下，PC中的值将保存入local register %l1中，nPC中的值将保存入local register %l2中。当从TRAP返回时，%l1中的值将被拷贝回PC中，%l2中的值将被拷贝回nPC中。

6.1.4.6 监视点寄存器（ASR）

计算机模块使用四对ASR寄存器（%asr24/25、%asr26/27、%asr28/30、%asr30/31）实现了四个监视点（watch point）：



由WADDR域定义的地址范围都可以被监视，这个地址范围也可被WMASK域掩码（WMASK[x]=1）

激活匹配。在一个监视点上，陷阱 0x0B会被生成。通过设定IF，DL和DS位监视hit可以在取指，数据存储读写的时候发生。将此三位清0会有效的禁止此项功能。

6.1.5 异常（Exception）

计算机模块使用异常（Exception）来处理在执行程序时发生的意外事件（如中断、存储器故障等）。在程序执行过程中，异常发生时，处理器将在执行完当前指令后，通过控制跳转类指令，跳转到特定的地址标号或者特定的子程序处执行异常处理，异常处理完成后，程序将返回。

异常事件改变了程序正常执行的顺序，是程序执行的非正常状态。在进入异常中断处理程序时，要保存被中断的程序的执行现场。在从异常中断处理程序退出时，要恢复被中断的程序的执行现场。每种异常中断都具有各自的备份寄存器组。

计算机模块支持中断（Interrupt）和陷阱（Trap）两种异常，中断将会在后面的中断控制器介绍。

陷阱：由与特定指令相关的硬件引起，在引起异常产生的指令运行期间发生。

当PSR寄存器的ET=0，表示一个陷阱异常会使IU执行挂起（halt），同时ERRORN脚将输出有效电平（即“0”）。

表 6-3 计算机模块陷阱及其优先级分配表

序号	Trap 名称	Trap 类型 (TT)	Trap 优先级 Priority	Trap 描述
1	reset	0x00	1	复位
2	write error	0x2b	2	写缓存错误
3	instruction_access_exception	0x01	3	取指令异常
4	illegal_instruction	0x02	5	企图执行UNIMP或其它未实现的指令
5	privileged_instruction	0x03	4	用户模式下（即PSR中的supervisor位为 0）企图执行特权指令

序号	Trap 名称	Trap 类型 (TT)	Trap 优先级 Priority	Trap 描述
6	fp_disabled	0x04	6	但FPU不存在或被关闭时，企图执行浮点处理指令
7	cp_disabled	0x24	6	但协处理器不存在或被关闭时，企图执行协处理指令
8	watchpoint_detected	0x0B	7	遇到硬件断点（即取指或Load/Store地址与watchpoint的内容匹配）
9	window_overflow	0x05	8	SAVE指令引起当前寄存器窗口指针CWP指向WIM寄存器中指定的非法窗口
10	window_underflow	0x06	8	RESTORE或RETT指令引起当前寄存器窗口指针CWP指向WIM寄存器中指定的非法窗口
11	register_hadrware_error	0x20	9	寄存器堆内容读错误
12	mem_address_not_aligned	0x07	10	指令存储阶段地址未对齐(un-aligned)
13	fp_exception	0x08	11	FPU异常
14	data_access_exception	0x09	13	load或store指令异常
15	tag overflow	0x0A	14	Tag溢出
16	divide_exception	0x2A	15	除法错误(除以0)
17	trap_instruction	0x80 -0xFF	16	软件Trap (TA)
18	Interrupt_15	0x1F	17	中断15
19	Interrupt_14	0x1E	18	中断14
20	Interrupt_13	0x1D	19	中断13
21	Interrupt_12	0x1C	20	中断12
22	Interrupt_11	0x1B	21	中断11

序号	Trap 名称	Trap 类型 (TT)	Trap 优先级 Priority	Trap 描述
23	Interrupt_10	0x1A	22	中断 10
24	Interrupt_9	0x19	23	中断 9
25	Interrupt_8	0x18	24	中断 8
26	Interrupt_7	0x17	25	中断 7
27	Interrupt_6	0x16	26	中断 6
28	Interrupt_5	0x15	27	中断 5
29	Interrupt_4	0x14	28	中断 4
30	Interrupt_3	0x13	29	中断 3
31	Interrupt_2	0x12	30	中断 2
32	Interrupt_1	0x11	31	中断 1

注：当多个trap同时发生时，优先级最高（Priority数小的优先级越高）优先处理和显示。

6.1.6 复位操作

当输入复位信号RESET在三个时钟周期内保持有效，计算机模块将进行复位操作。表 6-4 给出了受复位信号影响的IU内部部分关键寄存器的初始值，而不受复位信号影响的寄存器保持原先的值或者未定义。

表 6-4 IU 内部部分寄存器复位状态

序号	寄存器	复位值
1	当前程序计数器PC(program counter)	0x0
2	下一条程序计数器nPC(next program counter)	0x4
3	处理器状态寄存器PSR(processor status register)	ET=0，S=1
4	Cache控制寄存器CCR(cache control register)	0x0

6.1.7 休眠模式（Power-down）

计算机模块的各个处理器核心支持休眠模式(power-down mode)，处于该模式下的处理器

核心内部信号将处于静止状态，这样可以减少动态开关功耗，进而将功耗降到最低水平。

当执行指令“WRASR %asr19”时，处理器核心将进入休眠模式，其指令流水将被挂起（halted）。当有中断产生的时候，处理器核心将退出休眠模式，进入正常工作模式。

6.1.8 多核处理器支持

计算机模块处理器采用对称多处理器（symmetric multi-processing,SMP）架构。计算机模块复位后只有第一个处理器（CPU0）是开始工作的，其它的3个处理器（CPU1～CPU3）处于非工作状态。其它的3个处理器需要系统在初始化时，由CPU0通过操作“MP status register”进行唤醒。

6.2 浮点单元(FPU)

计算机模块的浮点单元(FPU)符合IEEE-754标准，专门负责浮点数处理。浮点单元(FPU)和整数单元（IU）在时间上是并行执行任务的。

计算机模块浮点单元(FPU)有三个作用：执行浮点指令操作、侦测与操作相关的数据、处理操作异常。浮点单元(FPU)支持单精度和双精度浮点运算，支持SPARC V8 浮点单元(FPU)指令。通过浮点单元(FPU) 控制器（MFC）连接，执行运算操作与IU并行。MFC执行SPARC 延迟陷阱模型。

计算机模块浮点单元(FPU)有32个32位的浮点寄存器。设计上采用了三模冗余技术，将触发器进行了抗辐照保护。双精度浮点数占用一对偶奇寄存器，四精度浮点数占用四个四字节对齐的寄存器。因此，浮点寄存器能保存32个单精度(32位)、16个双精度(64位)、8个四倍精度寄存器(128位)的数。

浮点 load/store指令用来在浮点单元(FPU)和内存之间交换数据，内存地址由IU计算。浮点数操作指令使用实际的浮点数运算。浮点数数据格式和指令集遵循IEEE标准。

计算机模块提供了一个浮点状态寄存器(FSR)，该寄存器中包含与浮点单元(FPU)相关的状态位和控制位。FSR可由用户软件访问，以检测浮点异常、舍入方向和非标准的算法模式。在PSR的EF位为0的情况下,执行一条浮点数指令就会引发一个fp_disabled的陷阱。

6.3 一级缓存（L1 Cache）

计算机模块的处理器核心含有高速一级缓存（L1 Cache）。一级缓存具体包含数据cache（Dcache）和指令cahce（Icache）。计算机模块的一级缓存使用LRU算法进行数据调度，采用透写策略（write-through）更新数据。在地址映射时，采用的是全相联映像法。

计算机模块每个处理器核心带有16K 字节数据缓存和32K字节指令缓存。

6.3.1 指令缓存(instration cache)

指令缓存可以配置成一个直接映射缓存或者一个4组缓存 ，缓存使用LRU算法。每组的大小为 2 千字节，缓存划分为每行32个字节的数据。 每行有一个缓存标签，还结合有标签区域，每 4个字节子块的有效区域有一位有效位、和锁位。当没有命中高速缓冲时，指令预取，对应的标签和数据行更新。 在多组配置中一行依照LRU法替代。

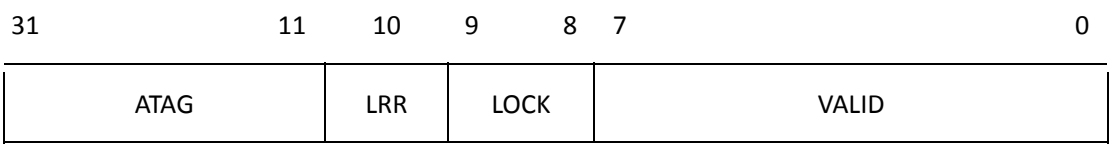
如果指令突发预取在缓存控制寄存器 (CCR)中使能，缓存行被从丢失地址开始的主存储器区域填充，直到行结束。 同时，指令被转寄到整数单元(IU) 。 如果由于内在的依赖或者多周期指令，IU没有接收数据流，则整数单元(IU) 被停止，直到行填充被完成。 如果整数单元(IU) 在行填充期间运行一个控制传输指令 (分支/调用/跳转/陷阱)，行填充将会在下次预取时结束。 如果指令突发预取使能,即使缓存被无效，指令流可以工作。在这情况，预取的指令只被转寄到整数单元(IU) ，缓存不更新。 在缓存行再填充期间，在 AHB 总线上产生逐渐增加的突发脉冲。

如果在IU停止期间，进行行填充时产生一个存储器访问错误， 在缓存标签的对应的有效位将不被设定。 如果整数单元(IU) 稍后预取来自无效地址的指令,一个缓存错误将会发生，触发一个对无效地址的访问。 如果错误保持,一个指令错误陷阱 (编号1) 将会产生。

6.3.2 数据缓存(data cache)

数据缓存由两部分构成：一部分用来存贮缓存标志（TAG）等信息，另一部分用来保存缓存的数据。

缓存标志等信息的存贮器称为标志存贮器，每个标志的结构定义如下:



各个区域的含义：

- [31:11]：地址标志(ATAG) - 主存贮器的地址信息；
- [10]：LRR -当置位时，使用 LRR 算法更新数据。“0”表示不使用；
- [9:8]：LOCK - 当置位时，这条缓存线被锁定；
- [7:0]：Valid(V) - 当置位时，表示当前缓存线的数据无效；

主存贮器地址的高21位保存在ATAG中，低11位则决定了数据在缓存中的位置。一条TAG对应了一块缓存区域，这块区域称为一条缓存线,目前1个缓存线大小为32字节。

如果CPU需要数据，它自己的缓存首先检查数据是否已经在自己的缓存中，如果是(称为缓存命中)，则直接返回数据。如果没有，则从主存中调入这个数据到自己的缓存中，同时传输给CPU。

计算机模块中的数据缓存采用透写策略（write-through）更新数据。当CPU把修改后的数据写入到缓存中时，缓存在更新自己缓存的同时，也通过AHB总线把数据写到主存贮器中。

计算机模块使用基于“写无效”的数据一致性策略。所有的缓存都会监听地址和数据总线，当总线上存在一个写操作时，缓存控制器会检查这个地址是否命中自己的缓存，如果命中，则无效自己相应的缓存区域。当CPU需要这个数据时，缓存就会从主存中读取，从而可以使自己的数据和主存中的数据保持一致。

如果在缓存无效这个缓存区域时，CPU正好需要这个数据，则缓存会把当前数据总线上的数据返回给CPU，并更新自己的缓存。

为了提高效率，缓存使用双口存贮器保存地址标志，CPU访问缓存和数据侦听是从两个不同的端口读取标志数据的。

6.3.3 缓存寄存器定义

表 6-5 L1 Cache 寄存器列表

ASI	地址(0x)	寄存器名称	有效宽度	读/写	默认值(0x)	描述
0x02	0x00	Cache 控制寄存器	32bit	R/W	0x00	cache freeze,flush, error 以及状态机状态查看

ASI	地址(0x)	寄存器名称	有效宽度	读/写	默认值(0x)	描述
0x02	0x08	ICache 配置寄存器	32bit	R/W	0x00	ICache lock，lru，相连度以及 MMU 开启配置
0x02	0x0C	DCache 配置寄存器	32bit	R/W	0x00	DCache lock，lru，相连度以及 MMU 开启配置

Cache 控制寄存器位描述

表 6-6 Cache 控制寄存器位描述

位	位名称	位描述
[28]	奇偶校验选择	如果进行诊断性读取则将返回4bit的校验位，否则返回cache标签或数据字
[27:24]	测试位	若使能，则诊断性写的数据将与测试位做异或
[23]	数据缓存监听使能	若置位则使能数据缓存监听功能
[22]	清空数据缓存	若置位则清空数据缓存
[21]	清空指令缓存	若置位则清空指令缓存
[20:19]	数据保护设置	“00” =无保护,“01” = 字节奇偶校验检查使能
[15]	指令缓存清空标志位	当指令缓存清空操作进行时此位被置位
[14]	数据缓存清空标志位	当数据缓存清空操作进行时此位被置位
[13:12]	指令缓存标签错误	指令缓存标签中检测到的奇偶校验错误个数
[11:10]	指令缓存数据错误	指令缓存数据中检测到的奇偶校验错误个数
[9:8]	数据缓存标签错误	指令缓存标签中检测到的奇偶校验错误个数

位	位名称	位描述
[7:6]	数据缓存数据错误	指令缓存数据中检测到的奇偶校验错误个数
[5]	数据缓存中断冻结	若置位，则异步中断发生时数据缓存自动被冻结
[4]	指令缓存中断冻结	若置位，则异步中断发生时指令缓存自动被冻结
[3:2]	数据缓存状态	显示当前数据缓存的状态值：X0 = dsiabled, 01 = frozen, 11 = enabled
[1:0]	指令缓存状态	显示当前指令缓存的状态值：X0 = dsiabled, 01 = frozen, 11 = enabled

ICache 配置寄存器位描述

表 6-7 ICache 配置寄存器位描述

位	位名称	位描述
[31]	缓存锁定	标示MMU的具体实现方式，由硬布线实现软件只读
[29:28]	缓存替换策略	保留
[27]	缓存监听	系统控制位。由开发者自定义功能，可以保留不用
[26:24]	缓存相联度	PSO 位决定内存结构对于 CPU 是部分存储顺序（PSO）还是全存储顺序（TSO）
[23:20]	每路大小	显示 ICache 每路的大小
[18:16]	每行大小	显示 ICache 每行的大小
[3]	MMU 使能标志位	若 MMU 已被使能则需要将此位置 1，否则置 0

DCache 配置寄存器位描述

表 6-8 DCache 配置寄存器位描述

位	位名称	位描述
[31]	缓存锁定	标示MMU的具体实现方式，由硬布线实现软件只读
[29:28]	缓存替换策略	保留
[27]	缓存监听	系统控制位。由开发者自定义功能，可以保留不用
[26:24]	缓存相联度	PSO 位决定内存结构对于 CPU 是部分存储顺序（PSO）还是全存储顺序（TSO）

位	位名称	位描述
[23:20]	每路大小	显示 DCache 每路的大小
[18:16]	每行大小	显示 DCache 每行的大小
[3]	MMU 使能标志位	若 MMU 已被使能则需要将此位置 1，否则置 0

6.4 存储器管理单元（MMU）

计算机模块的处理器核心具有存储区管理单元MMU（Memory Management Unit），该MMU遵循IEEE-1754标准中“the SPARC V8 Reference Memory Management Unit”的相关规定。

计算机模块的的MMU可以提供36位的物理地址空间与多个32位的虚拟地址空间之间的映射。当MMU被关闭，处理器核心的Cache使用正常的物理地址映射方式。当MMU被使能，Cache的Tag域存储的是虚拟地址信息。

MMU 寄存器定义，如下表所述。

表 6-9 MMU 寄存器列表

ASI	地址(0x)	寄存器名称	有效宽度	读/写	默认值(0x)	描述
0x19	0x000	MMU 控制寄存器	32bit	R/W	0x00	MMU 控制位设置
0x19	0x100	上下文指针寄存器	32bit	R/W	0x00	存储上下文指针
0x19	0x200	上下文寄存器	32bit	R/W	0x00	存储上下文
0x19	0x300	错误状态寄存器	32bit	R/W	0x00	存储错误状态
0x19	0x400	错误地址寄存器	32bit	R/W	0x00	存储错误地址

6.4.1 MMU 控制寄存器

表 6-10 MMU 控制寄存器

位	位名称	位描述
[31:28]	IMPL	标示MMU的具体实现方式，由硬布线实现软件只读
[27:24]	REV	保留
[23:8]	SC	系统控制位。由开发者自定义功能，可以保留不用
[7]	PSO	PSO 位决定内存结构对于 CPU 是部分存储顺序（PSO）还是全存储顺序（TSO）
[6:2]	REV	保留
[1]	NF	No Fault 位。NF = 0 时，任何 MMU 监测到的错误都会更新 FSR 和 FAR，并且向处理器产生错误信号。NF = 1 时，访问 ASI 9 产生的错误将采用与 NF = 0 相同的方式处理。访问其余 ASI 产生的错误将会更新 FSR 和 FAR，并且但处理器不会产生错误信号。
[0]	E	MMU 使能位，为 1 时 MMU 使能，为 0 时关闭 MMU

设置程序示例

```
static inline void srmmu_set_mmureg(unsigned long regval)
{
        asm volatile("sta %0, [%%g0] %1\n\t" : :
                             "r" (regval), "i" (ASI_M_MMUREGS) : "memory");
}
```

6.4.2 MMU 上下文指针寄存器

表 6-11 MMU 上下文指针寄存器

位	位名称	位描述
[31:2]	上下文指针	上下文列表指针指向物理内存中的上下文列表。列表由上下文寄存器中的值索引。
[1:0]	REV	保留

位	位名称	位描述

设置程序示例
static inline void srmmu_set_ctable_ptr(unsigned long paddr)
{
paddr = ((paddr >> 4) & SRMMU_CTX_PMASK);
asm volatile("sta %0, [%1] %2\n\t" : :
"r" (paddr), "r" (SRMMU_CTXTBL_PTR),
"i" (ASI_M_MMUREGS) :
"memory");
}		

6.4.3 MMU 上下文寄存器

表 6-12 MMU 上下文寄存器

位	位名称	位描述
[31:0]	上下文地址空间值	定义了当前进程的虚拟地址空间值。MMU对内存的访问都将通过上下文寄存器进行地址转换。

设置程序示例
static inline void srmmu_set_context(int context)
{
asm volatile("sta %0, [%1] %2\n\t" : :
"r" (context), "r" (SRMMU_CTX_REG),
"i" (ASI_M_MMUREGS) : "memory");
}		

6.4.4 MMU 错误状态寄存器

MMU错误状态寄存器提供由MMU产生的错误（异常）信息。错误信息一共分为三类：1.指令的访问错误 2.数据访问错误 3.转换列表访问错误。

如果前一条指令访问错误未被CPU读取而又发生了一条指令访问错误，则MMU将覆盖错误信息并且将OW位置高。数据访问错误也可以覆盖指令访问错误，但是不会置位OW位。假如MMU

访问外部系统时产生错误，则会生成转换列表错误信号，且指令访问错误和数据访问错误不会覆盖转换列表错误。

表 6-13 MMU 错误状态寄存器

位	位名称	位描述
[31:18]	reserved	保留位
[17:10]	EBE	外部总线错误标识符
[9:8]	L	MMU 错误级别标识符。提示错误发生 MMU 地址转换的第几级。定义如下：00 上下文列表 01 第一级页表 10 第二级页表 11 第三级页表
[7:5]	AT	访问类型错误标识符。提示发生错误的访存类型 定义如下： 000 读取用户数据空间 001 读取管理员数据空间 010 读取用户指令空间 011 读取管理员指令空间 100 存储用户数据空间 101 存储管理员数据空间 110 存储用户指令空间 111 存储管理员指令空间
[4:2]	FT	错误类型标识符。 提示当前发生的错误的类型 定义如下： 000 无错误 001 无效地址错误 010 保护错误 011 优先级错误 100 转换错误 101 访问总线错误 110 内部错误 111 保留
[1]	FAV	设置错误地址寄存器是否有效
[0]	OW	覆盖标识位。 提示当前错误信息是否被同一类型错误信息覆盖。 1 表示已覆盖，0 表示未被覆盖

6.4.5 MMU 错误地址寄存器

错误地址寄存器储存错误发生的虚拟地址值。同一类型的错误可以覆盖此寄存器，且此寄存器为只读。

表 6-14 MMU 错误地址寄存器

位	位名称	位描述
[31:0]	Fault Address	错误地址值

6.4.6 MMU Flush 操作

Flush操作刷新MMU中的页表项。通过写相应的地址可以实现Flush操作。示例代码如下：

```
void leon_flush_tlb_all (void)
{
```

```
        __asm__ __volatile__("sta %%g0, [%0] %1\n\t": :
                                        "r" (0x400),
                                        "i" (0x18) : "memory");
}
```

6.4.7 MMU Bypass 操作

Bypass操作可以让CPU在MMU使能的情况下不经地址转换直接读写内存。示例代码如下：

```
    /* 使用物理地址不经转换直接写内存 */
static __inline__ void leon_store_bp(unsigned long paddr,unsigned long value)
{
        __asm__ __volatile__("sta %0, [%1] %2\n\t": :
                                        "r" (value), "r" (paddr),
                                        "i" (ASI_MMU_BP) : "memory");
}

/* 使用物理地址不经转换直接读内存 */
static __inline__ unsigned long leon_load_bp(unsigned long paddr)
{
        unsigned long retval;
        __asm__ __volatile__("lda [%1] %2, %0\n\t" :
                                        "=r" (retval) :
                                        "r" (paddr), "i" (ASI_MMU_BP));
        return retval;
}
```

7. 二级缓存（L2 Cache）

计算机模块内集成二级缓存（L2 Cache，简称L2C），位于外存储器控制器与128-bit CPU AHB总线之间。CPU AHB总线是L2C的主机，L2C是外存储器控制器的主机。计算机模块二级缓存的设计，提高了处理器访问外部存储器的效率，进而提高了处理器的性能。

计算机模块内的二级缓存容量为512K字节，被配置成4个块，每个块的容量为128K字节。为了提高抗辐照能力，二级缓存中的存储器模块被EADC检错纠错功能说保护。

计算机模块复位后，二级缓存默认是关闭的。通过操作相关寄存器，软件可以使能二级缓存。

7.1 读操作

当CPU AHB总线向L2C发起一个可缓存（cachable）的读操作时，L2C首先要判断被读的数据是否在缓存中（或称作是否命中缓存）。

- 若命中（cache hit），即目标数据在缓存中，CPU AHB总线将直接从缓存中读取数据，而无需访问外部存储器；
- 若没命中（cache miss），即目标数据不在缓存中，L2C将通知外部存储器控制器，由外部存储器控制器发起连续的读数操作，从外部存储器中读取包含目标数据的数据块到缓存中。

当CPU AHB总线发起一个非缓存（non-cachable）的读操作时，L2C直接将读操作转给外部存储器控制器，由外部存储器控制器发起单次读数操作，读取目标数据。

7.2 写操作

当CPU AHB总线向L2C发起一个可缓存（cachable）的写操作时，L2C首先要判断被写目标地址是否已经在缓存中（或称作是否命中缓存）。

- 若命中（cache hit），即被写目标地址在缓存中，CPU AHB总线将直接把数据写到缓存相应的缓存块中，更新相应的缓存块，而无需直接写入外部存储器。

- 若没命中（cache miss），即被写目标地址不在缓存中，L2C将通知外部存储器控制器，由外部存储器控制器发起单次写数操作，将数写入目标地址。

当CPU AHB总线发起一个非缓存（non-cachable）的写操作时，L2C直接将写操作转给外部存储器控制器，由外部存储器控制器发起单次写数操作，将数据写入目标地址。

7.3 Flush 操作

用户软件可以通过操作cache flush寄存器，以实现flush二级缓存的操作，具体详见7.6.3节和7.6.4节相关寄存器的使用描述。

7.4 诊断接口

计算机模块的二级缓存提供诊断接口，该接口用于内部RAM模块测试，用户也可以通过该接口访问cache的Tag、Data以及EDAC check bits等内容。

7.5 地址映射

计算机模块的二级缓存相关的地址映射如表7-1所示。

表 7-1　计算机模块二级缓存相关的地址映射

地址	L2C 寄存器（Register）
0xF0000000	L2C控制寄存器(Control register)
0xF0000004	L2C状态寄存器(Status register)，只读
0xF0000008	L2C Flush寄存器1(Memory address)
0xF000000C	L2C Flush寄存器2(set, index)
0xF0000010	Access counter
0xF0000020	L2C错误状态/控制寄存器(Error status/control)
0xF0000024	L2C错误地址寄存器(Error address)
0xF0000028	L2C TAG校验位寄存器(TAG-check-bit)
0xF000002C	L2C 数据校验位寄存器(Data-check-bit)

地址	L2C 寄存器（Register）
0xF0000030	L2C scrub控制/状态寄存器(Scrub Control/Status)
0xF0000034	L2C scrub延迟寄存器(Scrub Delay)
0xF0000038	L2C 错误注入寄存器(Error injection)
0xF0000080 | 0xF00000FC	L2C 存储器类型范围寄存器(MTRR registers)
0xF0080000 | 0xF008FFFC	L2C TAG诊断接口区： 0xF0080000: Tag 1, way-1 0xF0080004: Tag 1, way-2 0xF0080008: Tag 1, way-3 0xF008000C: Tag 1, way-4 0xF0080010: Tag check-bits way-0,1,2,3 （只读） bit[27:21] = way-1的check-bits bit[20:14] = way-2的check-bits bit[13:7] = way-3的check-bits bit[6:0] = way-4的check-bits 0xF0080020: Tag 2, way-1 0xF0080024: Tag 2, way-2 0xF0080028: Tag 2, way-3 0xF008002C: Tag 2, way-4 0xF0080030: Tag check-bits way-0,1,2,3 （只读） bit[27:21] = way-1的check-bits bit[20:14] = way-2的check-bits bit[13:7] = way-3的check-bits bit[6:0] = way-4的check-bits 0xF0080040: Tag 3, way-1 0xF0080044: Tag 3, way-2 本区其他地址的定义，以此类推

地址	L2C 寄存器（Register）
	...
0xF0200000 | 0xF03FFFFC	L2C Data诊断接口区： 0xF0200000-0xF027FFFC: way-1的Data或 check-bits 0xF0280000-0xF02FFFFF: way-2的Data或 check-bits 0xF0300000-0xF037FFFC: way-3的Data或 check-bits 0xF0380000-0xF03FFFFF: way-4的Data或 check-bits

7.6 寄存器定义

7.6.1 L2C 控制寄存器

表 7-2 L2C控制寄存器

位	位名称	位描述
31	EN	CACHE 使能，1：使能；0：禁能。
30	EDAC	EDAC 使能，1：使能；0：禁能。
[29:28]	REPL	替换策略选择；00:LRU,01:随机,10:master-index(索引替换), 11:master-index(取模函数)
[27:16]	REV	保留
[15:12]	INDEX-WAY	需要替换的 way
[11:8]	LOCK	锁定 way 的数量
[7:6]	RES	保留

位	位名称	位描述
[5]	HPRHB	进行一次不缓存的读操作，即使cache命中，任然从memory中读取数据
[4]	HPB	标记缓存
[3]	UC	总线使用状态，0：环形模式 1：移位模式
[2]	HC	命中状态模式选择，0：环形模式 1：移位模式
[1]	WP	写策略。0：写通 1：复制返回
[0]	HP	HPROT 控制性能

7.6.2 L2C 状态寄存器

表 7-3 L2C状态寄存器

位	位名称	位描述
[31:25]	RES	保留
[24]	LSIZE	CACHE 行大小。1：64 字节 0：32 字节
[23]	FTTIME	方式时序是否是重建的
[22]	EDAC	EDAC是否存在
[21:16]	MTRR	MTRR 寄存器数量
[15:13]	BBUS WIDTH	总线位宽。1 = 128-bit, 2 = 64-bit, 4 = 32-bit.
[12:2]	CACHE SET SIZE	Cache 大小（KB）
[1:0]	WAY	多通道状态。 “00“: 直接映射 “01“: 2-way “10“: 3-way “11“: 4-way

7.6.3 L2C flush 寄存器

表 7-4 L2C flush寄存器 1(memory address)

位	位名称	位描述
[31:5]	Memory address	存储器地址，如果要 flush 所有的 cache 空间，该区域置 0
[4]	REV	保留

位	位名称	位描述
[3]	DI	Cache 关闭
[2:0]	Flush	Flush模式选择： “001“：对一行标记无效，“010”：将一行cache数据写回,但不进行无效标记 “011“：进行无效标记并写一行回数据. “101“:将cache空间全部标记为无效，“110”:写回所有的cache数据，但不进行无效标记，“111“：写回所有的cache数据并且进行无效标记

7.6.4 L2C flush 寄存器

表 7-5 L2C flush寄存器2(set, index)

位	位名称	位描述
[31:16]	Cache line index	对 cache 行 flush 操作时候使用的 Cache 行索引
[31：10]	TAG	对 cache 进行通道 flush 操作时候使用。进行 cache 行 flush 操作时，[15：0]应该被置‘0’
[9]	FETCH	置 1 时，执行通道 flush 操作后，数据存 memory 读取。进行 cache 行 flush 操作时，该位应该被置‘0’
[8]	VALID	通道flush操作时的有效位。进行cache行flush操作时，该位应该被置‘0’
[7]	DIRTY	通道 flush 操作时的 dirty 位。进行 cache 行 flush 操作时，该位应该被置‘0’
[6]	RES	保留
[5:4]	WAY	Cache 通道
[3]	DI	Cache 关闭
[2]	WF	进性通道 flush 操作

位	位名称	位描述
[1:0]	Flush	Flush模式选择： 行flush操作： “01“：对cache一行做无效标记 “10”：如果数据修改过，对cache行进行写回存储器操作 “11“：如果数据修改过，对cache行进行写回存储器操作，并做无效标记. 通道flush操作： “01“：根据[8:7]，更新 Valid/Dirty 位 “10”：如果数据修改过，对cache行进行写回存储器操作 “11“：如果数据修改过，对 cache 行进行写回存储器操作，并且根据[8:7]，更新 Valid/Dirty 位

7.6.5 L2C 错误状态控制寄存器

表 7-6 L2C错误状态控制寄存器

位	位名称	位描述
[31:28]	AHB master index	AHB 主机索引
[27]	SCRUB	错误是否由 scrubber 触发
[26:24]	TYPE	访问错误类型： 000：读cache 001：写cache，010:存储器读取，011:存储器写入， 100：违反了写保护，101：ahb 读总线错误，110：ahb 总线写错误
[23]	TAG/DATA	0：tag错误 1：数据错误
[22]	COR/UCOR	0：可纠正错误 1：不可纠正错误
[21]	MULTI	多次错误标记
[20]	VALID	错误状态寄存器是否有 valid 错误
[19]	DISERSESP	关闭不可纠错 EDAC 错误响应
[18:16]	CORRECTABLE ERROR COUNTER	可纠正 EDAC 错误计数
[15:12]	IRQ pending	bit3：AHB 错误 bit2：写保护错误

位	位名称	位描述
		bit1：不可纠正 EDAC 错误 bit0：可纠正 EDAC 错误
[11:8]	IRQ MASK	中断屏蔽 bit3：AHB 错误 bit2：写保护错误 bit1：不可纠正 EDAC 错误 bit0：可纠正EDAC错误
[7:6]	SELECT CB	诊断时，数据写操作校验位数据选择 00：使用正常逻辑产生的校验位 01：使用 data-check-bit 寄存器中的数据作为校验位 10：data-check-bit 寄存器中的数据于逻辑产生的校验位做XOR操作后，结果作为校验位 11：使用正常逻辑产生的校验位
[5:4]	SELECT TCB	诊断时，TAG写操作校验位数据选择 00：使用正常逻辑产生的校验位 01：使用 tag-check-bit 寄存器中的数据作为校验位 10：tag-check-bit 寄存器中的数据于逻辑产生的校验位做XOR操作后，结果作为校验位 11：使用正常逻辑产生的校验位
[3]	XCB	置1时，下一次数据或者TAG写操作时，校验位将会做XOR操作
[2]	RCB	置1时，做诊断读数据的时候，会同时读回校验位。
[1]	COMP	置1时，一次触发不可纠正错误的读操作将会触发AHB错误响应
[0]	RST	初始化错误状态寄存器

7.6.6 L2C 错误地址寄存器

表 7-7 错误地址寄存器

位	位名称	位描述
[31:0]	Error address	错误地址寄存器

7.6.7 L2C TAG 位校验位寄存器

表 7-8 TAG校验位寄存器

位	位名称	位描述
[31:7]	REV	保留
[31:0]	TCB	Tag 校验位

7.6.8 L2C 数据校验位寄存器

表 7-9 数据校验位寄存器

位	位名称	位描述
[31:28]	REV	保留
[31:0]	DCB	数据校验位

7.6.9 L2C scrub 控制状态寄存器

表 7-10 L2C scrub控制状态寄存器

位	位名称	位描述
[31:16]	INDEX	下次做 scrub 操作的行索引数值
[15:4]	REV	保留
[3:2]	WAY	下次做 scrub 操作的通道索引数值
[1]	PEN	一次 scrub 操作被挂起
[0]	EN	0：关闭自动 scrub 操作 1：打开自动 scrub 操作

7.6.10 L2C scrub 延迟寄存器

表 7-11 L2C scrub延迟寄存器

位	位名称	位描述
[31:16]	REV	保留
[15:0]	Delay	延迟 scrub 操作等待

7.6.11 L2C 错误注入寄存器

表 7-12 L2C错误注入寄存器

位	位名称	位描述
[31:2]	Address	注入错误的地址

位	位名称	位描述
[1]	RES	保留
[0]	Inject	1：注入错误在 Address 指定的地址

7.6.12 L2C 存储器类型范围寄存器

表 7-13 L2C存储器类型范围寄存器

位	位名称	位描述
[31:18]	Address field	需要和 cache 地址的[31：18]进行比较的地址设置
[17：16]	ACC	00：不缓存　01：写通
[15：2]	Address mask	地址屏蔽位。‘0’为屏蔽有效
[1]	Write-protection	0：写保护关闭　1：写保护开启
[0]	Access control field	0：访问控制关闭　1：访问控制开启

8. 地址空间分配

8.1 内部地址空间分配

表 8-1 计算机模块内部地址空间分配

地址空间	描述	容量（字节）
0x00000000—0x1FFFFFFF	外部存储器 ROM 区	512M
0x20000000—0x3FFFFFFF	外部存储器 I/O 区	512M
0x40000000—0x5FFFFFFF	外部存储器 SRAM 区	512M
0x60000000—0x7FFFFFFF	外部存储器 DDR2 区	512M
0x80000000－0x801FFFFF	APB 总线	2M
0x80200000－0x802000FF	CAN 通道 1 总线控制器	256
0x80200100－0x802001FF	CAN 通道 2 总线控制器	256
0x80210000－0x802100FF	遥测遥控的 TC 控制器	256

地址空间	描述	容量（字节）
0x802A0000－0x802A00FF	USB HOST	256
0x802FF000－0x802FFFFF	AHB I/O 总线配置区	4K
0x90000000－0x9FFFFFFF	DSU	256M
0xF0000000－0xF03FFFFF	二级缓存寄存器区	4M
0xFFEFF000－0xFFEFFFFF	AHB 存储器总线配置区	4K
其它地址	保留未用	/

8.2 APB 总线地址分配

表 8-2 计算机模块 APB 总线地址分配

地址空间	描述	容量（字节）
0x80000000—0x800000FF	FTM 控制器	256
0x80000100—0x800001FF	UART 1	256
0x80000200—0x800002FF	中断控制器	256
0x80000300—0x800003FF	通用定时器	256
0x80000400—0x800004FF	SPI	256
0x80000500—0x800005FF	I2C	256
0x80000600—0x800006FF	GPIO1	256
0x80000700—0x800007FF	UART(debug)	256
0x80000800—0x800008FF	遥测遥控的 TM 控制器	256
0x80000900—0x800009FF	UART 2	256
0x80000A00—0x80000AFF	SPW 1	256
0x80000B00—0x80000BFF	SPW 2	256
0x80000C00—0x80000CFF	SPW 3	256
0x80000D00—0x80000DFF	SPW 4	256
0x80000E00—0x80000EFF	以太网	256
0x80000F00—0x80000FFF	AHB 状态模块	256

地址空间	描述	容量（字节）
0x80100000—0x801000FF	通用带锁存同能定时器	256
0x80100100—0x801001FF	UART 3	256
0x80100200—0x801002FF	UART 4	256
0x80100300—0x801003FF	USBHC(EHC)	256
0x80100400—0x801004FF	GPIO 2	256
0x80100500—0x801005FF	GPREG	256
0x80108000—0x8010BFFF	1553 通道 1 的 Register	4k
0x8010c000—0x8010FFFF	1553 通道 1 的 Memory	4k
0x80110000—0x80113FFF	1553 通道 2 的 Register	4k
0x80114000—0x80117FFF	1553 通道 2 的 Memory	4k

8.3 AHB 总线状态寄存器

AHB总线状态寄存器记录了关于AHB总线传输过程中出错信息。其中状态寄存器记录了AHB错误状态信息，错误地址寄存器记录了AHB出错地址。

8.3.1 寄存器地址分配

表 8-3 寄存器地址分配

地址	读/写	有效位宽	默认值(HEX)	寄存器描述
0x80000F00	R/W	32	00000000	AHB 状态寄存器(AHB STATUS)
0x80000F04	R	32	00000000	AHB 出错地址寄存器(AHB FAIL ADDR)

8.3.2 AHB 状态寄存器

表 8-4 AHB 状态寄存器(AHB STATUS)

位	位名称	位描述
[31:10]	RES	保留
[9]	CE	CE,可纠正错误。当检测到可纠正错误时置 1，写 0 清除。

位	位名称	位描述
[8]	NE	NE,新错误。当检测到错误时置 1，写 0 清除。
[7]	HWRITE	AHB 总线 HWRITE 信号引起的错误
[6:3]	HMASTER	AHB 总线 HMASTER 信号引起的错误
[2:0]	HSIZE	AHB 总线 HSIZE 信号引起的错误

8.3.3 AHB 出错地址寄存器

表 8-5 AHB 出错地址寄存器(AHB FAIL ADDR)

位	位名称	位描述
[31:0]	Failing Address	AHB 总线 HADDR 信号引起的错误

9. 中断控制器

计算机模块内部集成支持多处理器的中断控制器，其是APB总线上的从设备。片内所有模块的中断请求信号均传送中断控制器，中断控制器通过优先级区分，中断屏蔽选择，把最高优先级的中断送所有处理器核心。

计算机模块的中断控制器支持15个非扩展中断（表 9-1的第1～第15号中断）和16个扩展中断（表 9-1的第16～第31号中断）。所有的扩展中断经处理合并成一个中断信号，挂接在第11号中断。

9.1 中断优先级

通过设置中断级别寄存器ILR（Interrupt Level Register），可以为计算机模块的15个非扩展中断（表 9-1的第1～第15号中断）设置中断级别（级别0或级别1）。级别1的中断优先级比级别0的中断优先级高。

同一级别的中断，中断号大的优先级较高（中断 15 有优先级最高，中断1优先级最低）。级别1的最大号中断将会优先得到响应，如果级别1的无中断激发，那么，级别0的最大号中断将会优先得到响应。

另外，计算机模块的16个扩展中断优先级相同，不支持优先级配置，它们继承第12号中断的优先级。

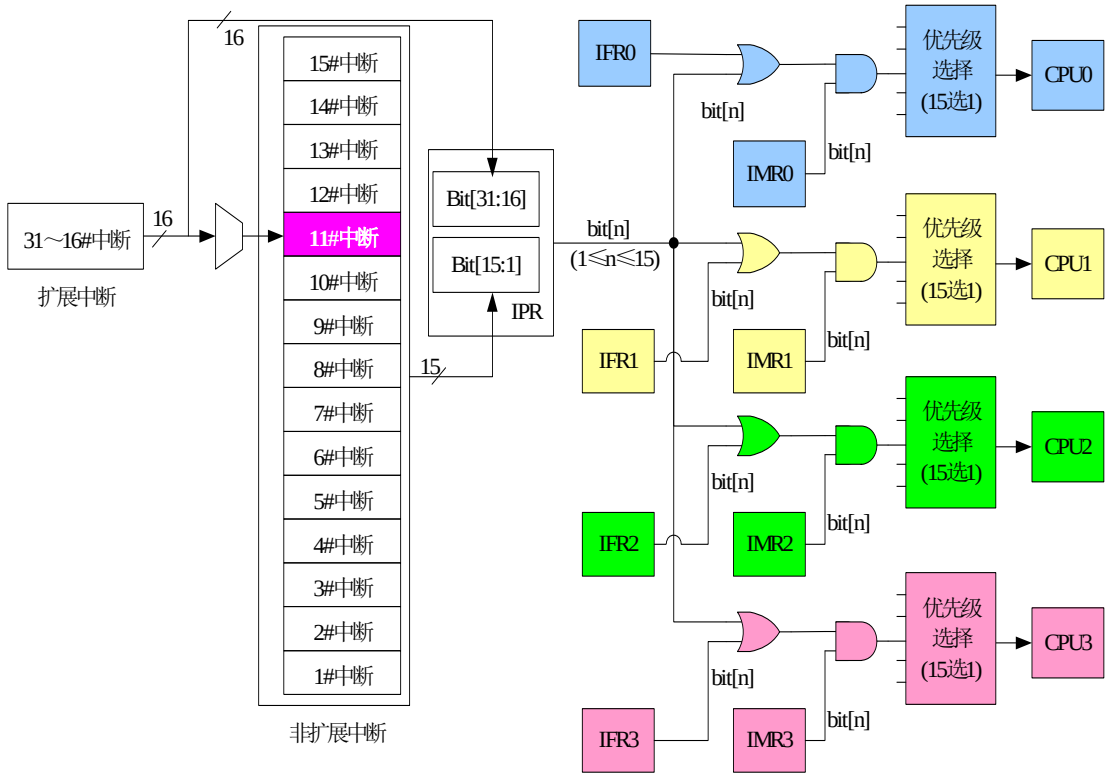
9.2 中断信号流程及中断处理过程

计算机模块的每个处理器核心均对应单独的中断屏蔽寄存器IMR（Interrupt Mask Register）和中断强制寄存器IFR（Interrupt Force Register）。中断屏蔽寄存器用来设定对应的CPU是否要响应某个中断（或称作是否要使能某个中断）。中断强制寄存器用来强行激发某个中断。

当表 9-1所示的某个中断激发有效，则中断悬挂寄存器IPR（Interrupt Pending Register）的相应位将被置1，若此时中断屏蔽寄存器的相应位为1，则中断有效信号将送给优先级筛选模块，优先级最高的中断信号将传送给该CPU，CPU将进行中断响应。CPU响应中断后，会自动将中断悬挂寄存器的相应位清零。

此外，软件可以通过将中断强制寄存器中的相应位置1的方法，来强制产生对应的中断。在这情况，CPU在响应中断后将清除中断强制寄存器中的相应位。

系统复位后，中断屏蔽寄存器全部被设定成零，屏蔽所有中断。



图错误!使用“开始”选项卡将 0 应用于要在此处显示的文字。-1 计算机模块的中断控制器功能结构图

9.3 多处理器状态监视

计算机模块通过多处理器状态寄存器MSR（Multiprocessor Status Register）可以监视各个处理器核心的状态。读MSR寄存器的bit[3:0]可以知道处理器核心是否处于运行状态，若某位为1，表示对应的处理器核心被挂起（halted），处于power-down模式，若某位为0，表示对应的处理器核心已经被激活，处于正常工作模式。往MSR寄存器的bit[3:0]的某位写入1，则可以激活对应的处理器核心。

系统复位后，除了CPU0，其它的三个CPU均处于power-down模式，CPU0可以通过操作多处理器状态寄存器MSR的bit[3:0]，激活CPU1～CPU3中的部分或全部。

9.4 中断分配表

计算机模块的中断控制器总共支持32个中断源，其具体分配如表 9-1所示：

表 9-1 计算机模块的中断列表

中断号	中断源	中断号	中断源
31	L2 CACHE 中断 / 外部 GPIO 中断	15	外部 GPIO 中断
30	UART4 中断 / 外部 GPIO 中断	14	外部 GPIO 中断
29	SPI 中断 / 外部 GPIO 中断	13	外部 GPIO 中断
28	I2C 中断 / 外部 GPIO 中断	12	外部 GPIO 中断
27	CCSDS TM 解码器中断 / 外部 GPIO 中断	11	扩展中断 / 外部 GPIO 中断
26	CCSDS TM 编码器中断 / 外部 GPIO 中断	10	M1553B-1/2 中断 / 外部 GPIO 中断
25	CAN2 中断 / 外部 GPIO 中断	9	通用定时器 4 中断 / 外部 GPIO 中断
24	CAN1 中断 / 外部 GPIO 中断	8	通用定时器 3 中断 / 外部 GPIO 中断
23	UART3 中断 / 外部 GPIO 中断	7	通用定时器 2 中断 / 外部 GPIO 中断
22	USB HOST 控制器中断 / 外部 GPIO 中断	6	通用定时器 1 中断 / 外部 GPIO 中断
21	外部 GPIO 中断	5	以太网中断 / 外部 GPIO 中断
20	CCSDS 中断(TC) / 外部 GPIO 中断	4	锁存定时器 1/2 中断 / 外部 GPIO 中断
19	SPW4 中断 / 外部 GPIO 中断	3	UART2 中断 / 外部 GPIO 中断
18	SPW3 中断 / 外部 GPIO 中断	2	UART1 中断 / 外部 GPIO 中断
17	SPW2 中断 / 外部 GPIO 中断	1	AHB 总线错误中断 / 外部 GPIO 中断
16	SPW1 中断 / 外部 GPIO 中断	-	-

注：第16-31号中断通过第11号中断扩展，故如需使用第16-31号中断，需要将第11号中断先使能。

9.5 外部中断扩展

计算机模块可以通过GPIO接口扩展外部中断，表 9-1中的每一个中断源都可以作为GPIO外部中断的接入点。至于哪一个GPIO外部中断接到哪一个中断源，可以通过GPIO中断映射配置寄存器（见表 12-8）来设定。由表 9-1可以看出，部分GPIO外部中断与片内模块中断复用同一个中断源，因此用户在进行外部中断扩展的时候，建议尽量先使用第12、13、14、15、21号中断作为GPIO外部中断接入点，若选用其它的中断源作为接入点，需要注意避免同片内

模块的中断冲突，建议采用同片内模块分时复用中断源，或关闭相应片内外设的中断输出的方法。

9.6 中断寄存器

计算机模块的中断控制器相关的寄存器如表 9-2所示。

表 9-2 中断控制寄存器列表

地址	读写	寄存器
0x80000200	R/W	中断级别寄存器 ILR（Interrupt Level Register）
0x80000204	R	中断悬挂寄存器 IPR（Interrupt Pending Register）
0x80000208	/	保留未用
0x8000020C	W	中断清除寄存器 ICR（Interrupt Clear Register）
0x80000210	R/W	多处理器状态寄存器 MSR（Multiprocessor Status Register）
0x80000214	W	中断广播寄存器 IBR（Interrupt Broadcast Register）
0x80000240	R/W	CPU0 中断屏蔽寄存器 IMR0（CPU0 Interrupt Mask Register）
0x80000244	R/W	CPU1 中断屏蔽寄存器 IMR1（CPU1 Interrupt Mask Register）
0x80000248	R/W	CPU2 中断屏蔽寄存器 IMR2（CPU2 Interrupt Mask Register）
0x8000024C	R/W	CPU3 中断屏蔽寄存器 IMR3（CPU3 Interrupt Mask Register）
0x80000280	W	CPU0 中断强制寄存器 IFR0（CPU0 Interrupt Force Register）
0x80000284	W	CPU1 中断强制寄存器 IFR1（CPU1 Interrupt Force Register）
0x80000288	W	CPU2 中断强制寄存器 IFR2（CPU2 Interrupt Force Register）
0x8000028C	W	CPU3 中断强制寄存器 IFR3（CPU3 Interrupt Force Register）
0x800002C0	R/W	CPU0 扩展中断响应寄存器（CPU0 extended interrupt acknowledge register）
0x800002C4	R/W	CPU1 扩展中断响应寄存器（CPU1 extended interrupt acknowledge register）
0x800002C8	R/W	CPU2 扩展中断响应寄存器（CPU2 extended interrupt acknowledge register）
0x800002CC	R/W	CPU3 扩展中断响应寄存器（CPU3 extended interrupt acknowledge register）

9.6.1 中断级别寄存器

表 9-3 中断级别寄存器 ILR

位	位名称	位描述
[31:16]	RES	保留，读值为 0.
[15:1]	非扩展中断的中断级别	非扩展中断的中断级别设置：0 或 1。 例如：若将 bit[5]设为 1，则表示第 5 号中断的中断级别为 1，具有高的优先级； 若将 bit[5]设为 0，则表示第 5 号中断的中断级别为 0，具有低的优先级；
[0]	RES	保留，读值为 0.

9.6.2 中断悬挂寄存器

表 9-4 中断悬挂寄存器 IPR

位	位名称	位描述
[31:0]	中断悬挂	若某位为 1，则对应表 9-1 中的相应的中断有效； 若某位为 0，则对应表 9-1 中的相应的中断无效。

9.6.3 中断清除寄存器

表 9-5 中断清除寄存器 ICR

位	位名称	位描述
[31:16]	扩展中断清除	往某位写‘1’，则将中断悬挂寄存器中的相应位清零； 写‘0’无意义。
[15:1]	非扩展中断清除	往某位写‘1’，则将中断悬挂寄存器中的相应位清零； 写‘0’无意义。
[0]	RES	保留，读值为 0.

9.6.4 多处理器状态寄存器

表 9-6 多处理器状态寄存器 MSR

位	位名称	位描述
[31:28]	NCPU	芯片中处理器核心数量-1，故该域恒为 3；
[27]	BA	该域恒为 1，表示支持中断广播；

位	位名称	位描述
[26:20]	RES	保留未用，读值为 0；
[19:16]	EIRQ	扩展中断接入到非扩展中断的入口编号。 计算机模块的扩展中断接在第 11 号中断上，故该域恒为 11。
[15:4]	RES	保留未用，读值为 0；
[3:0]	STATUS	读该域： 若第 3 位为 1，表示 CPU3 处于 power-down 模式；为 0 则为正常工作模式； 若第 2 位为 1，表示 CPU2 处于 power-down 模式；为 0 则为正常工作模式； 若第 1 位为 1，表示 CPU1 处于 power-down 模式；为 0 则为正常工作模式； 若第 0 位为 1，表示 CPU0 处于 power-down 模式；为 0 则为正常工作模式； 往该域写 1（写 0 无效）： 若往第 3 位写入 1，则可将 CPU3 从 power-down 模式唤醒，进入正常工作模式； 若往第 2 位写入 1，则可将 CPU2 从 power-down 模式唤醒，进入正常工作模式； 若往第 1 位写入 1，则可将 CPU1 从 power-down 模式唤醒，进入正常工作模式； 若往第 0 位写入 1，则可将 CPU0 从 power-down 模式唤醒，进入正常工作模式；

9.6.5 中断广播寄存器

表 9-7 中断广播寄存器 IBR

位	位名称	位描述
[31:16]	RES	保留，读值为 0.
[15:1]	BM	如果第 n（0<n<16）位被写入 1，将使能第 n（0<n<16）号非扩展中断广播模式。表示第 n 号非扩展中断广播到所有的处理器核心（CPU0～CPU3），相当于将该中断写入了所有 CPU 的中断强制寄存器的相应位中。 写入 0，无效。
[0]	RES	保留，读值为 0.

9.6.6 中断屏蔽寄存器

表 9-8 中断屏蔽寄存器 IMR

位	位名称	位描述
[31:16]	EIM	扩展中断屏蔽 将第 n（15<n<32）位置 0：表示第 n 个扩展中断被屏蔽； 将第 n（15<n<32）位置 1：表示第 n 个扩展中断被使能；
[15:1]	IM	非扩展中断屏蔽 将第 n（0<n<16）位置 0：表示第 n 个非扩展中断被屏蔽； 将第 n（0<n<16）位置 1：表示第 n 个非扩展中断被使能；
[0]	RES	保留，读值为 0.

9.6.7 处理器中断强制寄存器

表 9-9 中断强制寄存器

位	位名称	位描述
[31:17]	EIF	扩展中断强制 往第 n 位写入 0：无意义； 往第 n 位写入 1：表示强制使第 n 个扩展中断有效；
[16]	RES	保留，读值为 0.
[15:1]	IF	非扩展中断强制 往第 n 位写入 0：无意义； 往第 n 位写入 1：表示强制使第 n 个非扩展中断有效；
[0]	RES	保留，读值为 0.

9.6.8 扩展中断响应寄存器

表 9-10 扩展中断响应寄存器

位	位名称	位描述
[31:5]	RES	保留，读值为 0.
[4:0]	EID	扩展中断 ID 号，取值范围为 0 或 16～31。

位	位名称	位描述
		如果最近一次中断响应（acknowledge）是针对某个扩展中断，则该域存储的就是该扩展中断的 ID 编号。 如果该域为 0，则表示最近一次中断响应不是针对扩展中断。

10. 通用定时器

计算机模块内部集成四个32位的通用定时器TIMER1～TIMER4，每个定时器均具有各自的定时溢出中断（第6～9号中断）。其中第四个通用定时器TIMER4还可被用作看门狗定时器WDOG，其具有定时溢出状态输出引脚WDOG（低有效）。

10.1 通用定时器工作原理

如图 所示，TIMER1～TIMER4共用一个16位的预分频器PRESCALER。预分频器PRESCALER在系统时钟SYS_CLK的驱动下，进行递减计数。预分频器PRESCALER的起始计数值由预分频器计数值寄存器(PRESCALER VALUE)设定，每当递减到0时，将从预分频器重载计数值寄存器(PRESCALER RELAOD VALUE)获取计数值，继续递减计数。每当预分频器PRESCALER递减到0时，将输出一个的脉冲PRESCALER_TICK（宽度为一个SYS_CLK时钟宽度）。

若TIMERn（n=1，2，3，4）被使能（即TIMERn的控制寄存器的bit0被置1），则TIMERn将在PRESCALER_TICK的驱动下进行递减计数。TIMERn的起始计数值由定时器n定时值寄存器(TIMERn VALUE)设定。当TIMERn递减到0时：

- 若TIMERn的自动重载被使能（即控制寄存器的bit1=1），则TIMERn从将从TIMERn重载定时值寄存器(TIMERn RELOD VALUE)中获取定时值，继续递减计数。
- 若TIMERn的自动重载被关闭（即控制寄存器的bit1=0），则TIMERn停止，同时自动将TIMERn的控制寄存器的bit0被置0；
- 若TIMERn的溢出中断被使能（即控制寄存器的bit3=1），则将产生定时溢出中断，TIMERn溢出中断标志将被置有效（即控制寄存器的bit4将被置1），溢出中断标志

将保持有效，直到控制寄存器的bit4被写入1才会被清除。

- 若TIMERn的溢出中断被关闭（即控制寄存器的bit3=0），则不产生定时溢出中断。

定时器TIMERn（n=1，2，3，4）定时时间长度P等于预分频器PRESCALER的计数值（PRESCALER VALUE）乘以定时器TIMERn的定时值（TIMERn VALUE），再乘以系统时钟SYS_CLK的周期（T_{SYS_CLK}），即：

$$P = PRESCALER\ VALUE \times TIMERn\ VALUE \times T_{SYS\_CLK} \quad （式8-1）$$

通用定时器TIMER4可被用作看门狗定时器WDOG，其具有定时溢出状态输出引脚WDOG（低有效）。引脚WDOG平时为高电平，当TIMER4定时溢出时，其溢出中断标志将被置有效，而且引脚WDOG将被置为低电平，直到TIMER4控制寄存器的bit4被写入1才会被清除。

计算机模块复位后，TIMER1～TIMER3均处于关闭状态，TIMER4（看门狗）处于使能状态，其预分频器计数值和重载定时值都被设为全1。

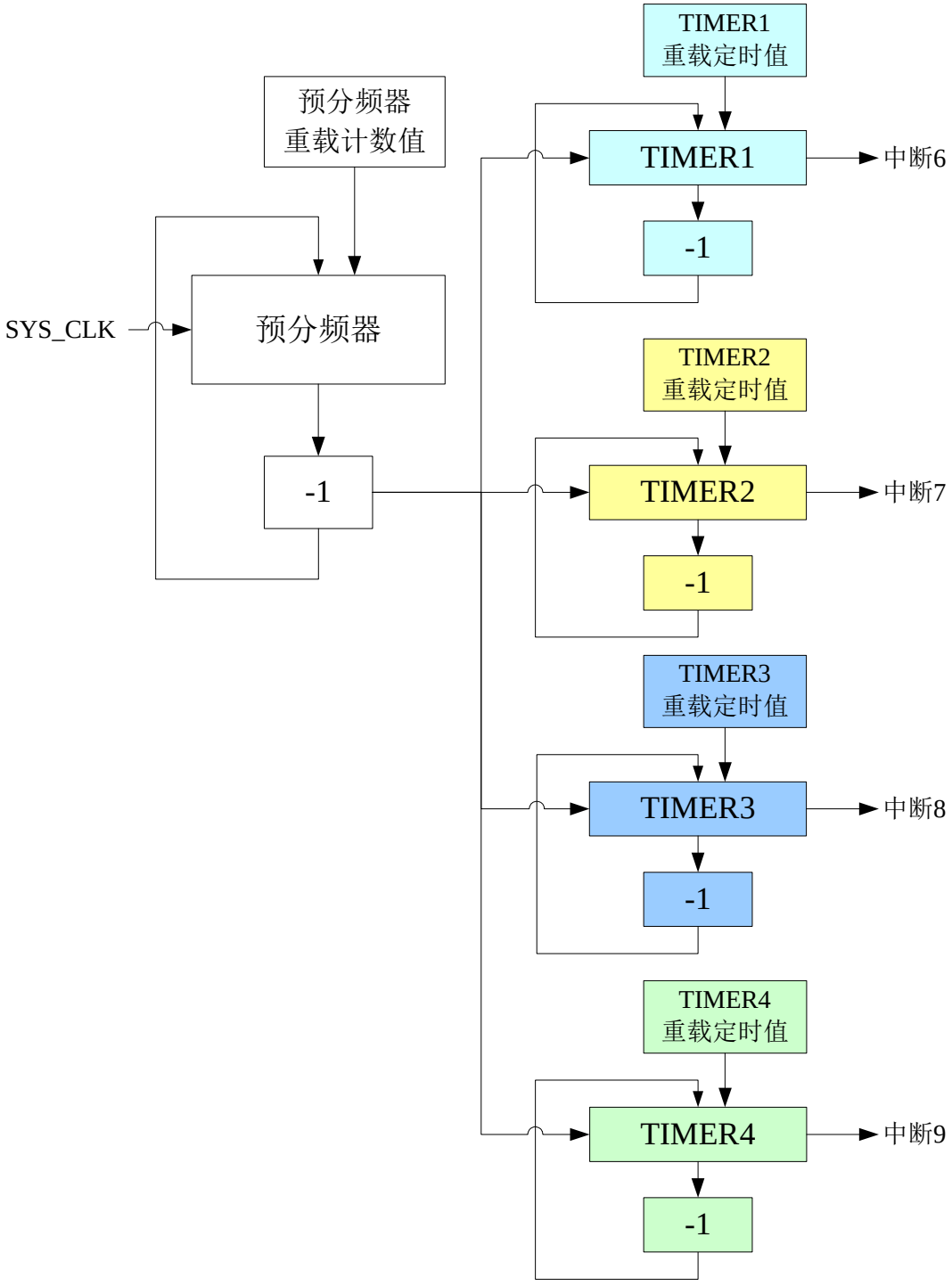


图 10-1 计算机模块通用定时器结构图

计算机模块的四个通用定时器具有“级联功能”。若TIMERn（n=2，3，4）的级联功能被使能（即控制寄存器的bit5=1），则TIMERn将在TIMERn-1的溢出信号驱动下进行递减计数。例如，若TIMER2的控制寄存器的bit5被置1，则TIMER1每发生一次定时溢出，TIMER2将递减1。

若四个定时器全部级联起来，相当于一个128位的定时器。

级联模式下，定时器TIMERn定时时间长度P等于预分频器PRESCALER的计数值（PRESCALER VALUE）乘以定时器TIMERn-1的定时值（TIMERn-1 VALUE），再乘以TIMERn的定时值（TIMERn VALUE），再乘以系统时钟SYS_CLK的周期（T_{SYS_CLK}），即：

$$P = PRESCALER\ VALUE \times TIMERn\text{-}1\ VALUE \times TIMERn\ VALUE \times T_{SYS\_CLK} \quad （式8-2）$$

10.2 通用定时器寄存器

表 10-1 寄存器地址分配

地址	读/写	有效位宽	默认值(HEX)	寄存器描述
0X80000300	R/W	32	0000FFFF	预分频器计数值寄存器(PRESCALER VALUE)
0X80000304	R/W	32	0000FFFF	预分频器重载计数值寄存器(PRESCALER RELOAD VALUE)
0x80000308	R/W	32	00000134	通用定时器配置寄存器(TIMER CONFIG)
0X80000310	R/W	32	FFFFFFFF	通用定时器 1 定时值寄存器(TIMER1 VALUR)
0X80000314	R/W	32	FFFFFFFF	通用定时器 1 重载定时值寄存器(TIMER1 RELOD VALUE)
0X80000318	R/W	32	00000000	通用定时器 1 控制寄存器(TIMER1 CONTROL)
0X80000320	R/W	32	FFFFFFFF	通用定时器 2 定时值寄存器(TIMER2 VALUE)
0X80000324	R/W	32	FFFFFFFF	通用定时器 2 重载定时值寄存器(TIMER2 RELOD VALUE)
0X80000328	R/W	32	00000000	通用定时器 2 控制寄存器(TIMER2 CONTROL)
0X80000330	R/W	32	FFFFFFFF	通用定时器 3 定时值寄存器(TIMER3 VALUR)
0X80000334	R/W	32	FFFFFFFF	通用定时器 3 重载定时值寄存器(TIMER3 RELOD VALUE)
0X80000338	R/W	32	00000000	通用定时器 3 控制寄存器(TIMER3 CONTROL)
0X80000340	R/W	32	FFFFFFFF	通用定时器 4 定时值寄存器(TIMER4 VALUR)
0X80000344	R/W	32	FFFFFFFF	通用定时器 4 重载定时值寄存器(TIMER4 RELOD VALUE)
0X80000348	R/W	32	00000009	通用定时器 4 控制寄存器(TIMER4 CONTROL)

10.2.1 预分频器计数值寄存器

表 10-2 预分频器计数值寄存器(PRESCALER VALUE)

位	位名称	位描述
[31:16]	RES	保留，读值为 0。
[15:0]	Prescaler Value	预分频器计数值

10.2.2 预分频器重载计数值寄存器

表 10-3 预分频器重载计数值寄存器(PRESCALER RELOAD VALUE)

位	位名称	位描述
[31:16]	RES	保留，读值为 0。
[15:0]	Prescaler Relod Value	预分频器重载计数值

10.2.3 通用定时器配置寄存器

表 10-4 通用定时器配置寄存器(TIMER CONFIG)

位	位名称	位描述
[31:10]	RES	保留，读值为 0。
[9]	DF	若被置‘1’，调试模式下，定时器仍将继续计数。 若被置‘0’，调试模式下，定时器将停止计数。
[8]	SI	该域恒为‘1’，表示四个通用定时器使用独立的中断。只读。
[7:3]	IRQ	该域恒为“00110”，表示四个通用定时器的起始中断号为 6。只读。
[2:0]	TIMERS	该域恒为“100”，表示计算机模块有 4 个通用定时器。只读。

10.2.4 通用定时器定时值寄存器

表 10-5 通用定时器定时值寄存器(TIMERn VALUE)

位	位名称	位描述
[31:0]	TIMER VALUR	定时器定时值。 每当 PRESCALER_TICK 有效时，该域递减 1。

10.2.5 通用定时器重载值寄存器

表 10-6 通用定时器重载值寄存器(TIMERn RELOAD VALUE)

位	位名称	位描述
[31:0]	RELOD VALUR	定时器重载定时值。 两种情况下进行计数重置： 1)、当定时器控制寄存器的 bit2 被写 1； 2)、控制寄存器中的 bit=1，且定时溢出时。

10.2.6 通用定时器控制寄存器

表 10-7 通用定时器控制寄存器(TIMER CONTROL)

位	位名称	位描述
[31:7]	RES	保留，读值为 0。
[6]	DH	调试模式，定时器挂起标志。只读。 若为‘1’，表示调试模式下，定时器仍将继续计数。 若为‘0’，表示调试模式下，定时器将停止计数。
[5]	CH	级联使能位，高有效。 若为 1，则 TIMERn 将在 TIMERn-1 溢出时递减 1。 若为 0，级联功能关闭。
[4]	IP	定时器溢出中断标志位。 若定时器溢出中断被使能(bit3=1)，当定时器溢出时，该位被置 1。 往该位写入 1 时，可将该位清零。 往该位写入写入 1，无效。
[3]	IE	定时器溢出中断使能位，高有效。
[2]	LD	定时器重载命令。 写入 1，定时器将立即从重载值寄存器中获取定时器重新定时计数。 写入 0，无效。
[1]	RS	定时器自动重载使能，高有效。

位	位名称	位描述
[0]	EN	定时器使能位，高有效。

11. 锁存定时器

计算机模块内部集成两个32位的带锁存功能的定时器LTIMER1～LTIMER2，两个锁存定时器均共用定时溢出中断（第4号中断）。

11.1 锁存定时器工作原理

如图 所示，LTIMER1和LTIMER2共用一个16位的预分频器PRESCALER。预分频器PRESCALER在系统时钟SYS_CLK的驱动下，进行递减计数。预分频器PRESCALER的起始计数值由预分频器计数值寄存器(PRESCALER VALUE)设定，每当递减到0时，将从预分频器重载计数值寄存器(PRESCALER RELAOD VALUE)获取计数值，继续递减计数。每当预分频器PRESCALER递减到0时，将输出一个的脉冲PRESCALER_TICK（宽度为一个SYS_CLK时钟宽度）。

若LTIMERn（n=1，2）被使能（即LTIMERn的控制寄存器的bit0被置1），则TIMERn将在PRESCALER_TICK的驱动下进行递减计数。TIMERn的起始计数值由定时器n定时值寄存器(TIMERn VALUR)设定。当TIMERn递减到0时：

- 若LTIMERn的自动重载被使能（即控制寄存器的bit1=1），则LTIMERn从将从LTIMERn重载定时值寄存器(LTIMERn RELOD VALUE)中获取定时值，继续递减计数。
- 若LTIMERn的自动重载被关闭（即控制寄存器的bit1=0），则LTIMERn停止，同时自动将LTIMERn的控制寄存器的bit0被置0；
- 若LTIMERn的溢出中断被使能（即控制寄存器的bit3=1），则将产生定时溢出中断，LTIMERn溢出中断标志将被置有效（即控制寄存器的bit4将被置1），溢出中断标志将保持有效，直到控制寄存器的bit4被写入1才会被清除。
- 若LTIMERn的溢出中断被关闭（即控制寄存器的bit3=0），则不产生定时溢出中断。

定时器LTIMERn（n=1，2）定时时间长度P等于预分频器PRESCALER输出脉冲周期乘以定时器定时值，再乘以系统时钟SYS_CLK的周期（T_{SYS_CLK}），即：

$$P = PRESCALER\ VALUE \times LTIMER\ VALUR \times T_{SYS\_CLK} \quad （式9-1）$$

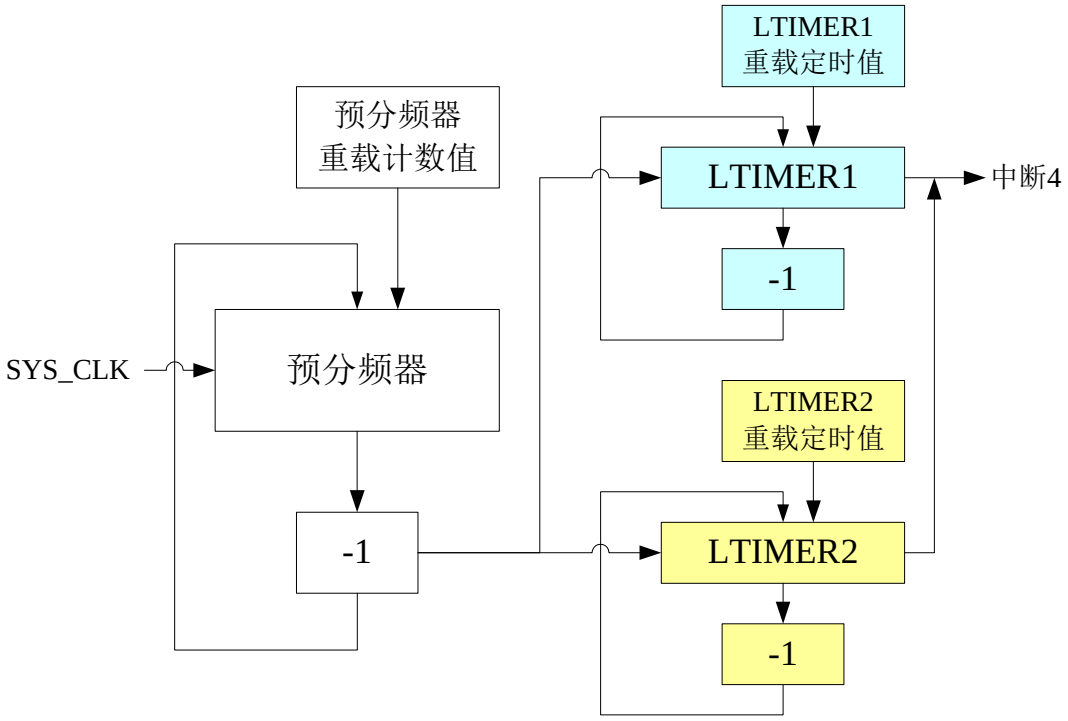


图 11-1 计算机模块锁存定时器结构图

计算机模块的两个锁存定时器具有“级联功能”。若LTIMER2的级联功能被使能（即控制寄存器的bit5=1），则LTIMER2将在LTIMER1的溢出信号驱动下进行递减计数，即LTIMER1每发生一次定时溢出，LTIMER2将递减1。若两个锁存定时器级联起来，相当于一个64位的定时器。

级联模式下，定时器LTIMER2定时时间长度P等于预分频器PRESCALER的计数值（PRESCALER VALUE）乘以定时器LTIMER1的定时值（LTIMER1 VALUE），再乘以LTIMER2的定时值（TIMER2 VALUE），再乘以系统时钟SYS_CLK的周期（T_{SYS_CLK}），即：

$$P = PRESCALER\ VALUE \times LTIMER1\ VALUE \times LTIMER2\ VALUE \times T_{SYS\_CLK} \quad （式9-2）$$

11.2 锁存定时器寄存器

表 11-1 锁存定时器寄存器地址分配

地址	读/写	有效位宽	默认值(HEX)	寄存器描述
0x80100000	R/W	32	0000FFFF	预分频器计数值寄存器(PRESCALER VALUE)
0x80100004	R/W	32	0000FFFF	预分频器重载计数值寄存器(PRESCALER RELOAD VALUE)
0x80100008	R/W	32	00000022	锁存定时器配置寄存器(LTIMER CONFIG)
0x8010000C	R/W	32	00000000	锁存触发中断选择寄存器(LTIMER LATCH INT SELECT)
0x80100010	R/W	32	FFFFFFFF	锁存定时器 1 定时值寄存器(LTIMER1 VALUR)
0x80100014	R/W	32	FFFFFFFF	锁存定时器 1 重载定时值寄存器(LTIMER1 RELOD VALUE)
0x80100018	R/W	32	00000000	锁存定时器 1 控制寄存器(LTIMER1 CONTROL)
0x8010001C	R	32	--------	锁存定时器 1 锁存值寄存器(LTIMER1 LATCH VALUE)
0x80100020	R/W	32	FFFFFFFF	锁存定时器 2 定时值寄存器(LTIMER2 VALUR)
0x80100024	R/W	32	FFFFFFFF	锁存定时器 2 重载定时值寄存器(LTIMER2 RELOD VALUE)
0x80100028	R/W	32	00000000	锁存定时器 2 控制寄存器(LTIMER2 CONTROL)
0x8010002C	R	32	--------	锁存定时器 2 锁存值寄存器(LTIMER2 LATCH VALUE)

11.2.1 预分频器计数值寄存器

表 11-2 预分频器计数值寄存器(PRESCALER VALUE)

位	位名称	位描述
[31:16]	RES	保留，读值为 0。
[15:0]	Prescaler Value	预分频器计数值

11.2.2 预分频器重载计数值寄存器

表 11-3 预分频器重载计数值寄存器(PRESCALER RELOAD VALUE)

位	位名称	位描述
[31:16]	RES	保留，读值为 0。

位	位名称	位描述
[15:0]	Prescaler Relod Value	预分频器重载计数值

11.2.3 锁存定时器配置寄存器

表 11-4 锁存定时器配置寄存器(LTIMER CONFIG)

位	位名称	位描述
[31:10]	RES	保留，读值为 0。
[11]	EL	锁存功能使能，高有效。 若为‘1’，锁存功能被使能。当锁存触发中断选择寄存器中选定的任意一个中断有效时，锁存定时器的当前计数值将被锁存到锁存值寄存器，同时该位将自动清零，直到下次被软件置 1 为止。 若为‘0’，锁存功能被关闭。
[10]	RES	保留，读值为 0。
[9]	DF	若被置‘1’，调试模式下，定时器仍将继续计数。 若被置‘0’，调试模式下，定时器将停止计数。
[8]	SI	该域恒为‘0’，表示两个锁存定时器共用中断。只读。
[7:3]	IRQ	该域恒为“00100”，表示锁存定时器的中断号为 4。只读。
[2:0]	TIMERS	该域恒为“010”，表示计算机模块有 2 个锁存定时器。只读。

11.2.4 锁存触发中断选择寄存器

表 11-5 锁存触发中断选择寄存器(LTIMER LATCH INT SELECT)

位	位名称	位描述
[31:1]	LATCH INT SELECT	若某些位被置 1，表示当对应的中断的任意一个有效，且锁存定时器配置寄存器的 bit11 为 1 是，LTIMER1 和 LTIMER2 的当前计数值将被存入各自的锁存值寄存器。
[0]	RES	保留，读值为 0。

11.2.5 锁存定时器定时值寄存器

表 11-6 锁存定时器定时值寄存器(COUNTER VALUR)

位	位名称	位描述
[31:0]	TIMER VALUR	定时器定时值。 每当 PRESCALER_TICK 有效时，该域递减 1。

11.2.6 锁存定时器重载值寄存器

表 11-7 锁存定时器重载值寄存器(RELOD VALUR)

位	位名称	位描述
[31:0]	RELOD VALUR	定时器重载定时值。 两种情况下进行计数重置： 1)、当定时器控制寄存器的 bit2 被写 1； 2)、控制寄存器中的 bit=1，且定时溢出时。

11.2.7 锁存定时器控制寄存器

表 11-8 锁存定时器控制寄存器(LTIMER CONTROL)

位	位名称	位描述
[31:7]	RES	保留，读值为 0。
[6]	DH	调试模式，定时器挂起标志。只读。 若为‘1’，表示调试模式下，定时器仍将继续计数。 若为‘0’，表示调试模式下，定时器将停止计数。
[5]	CH	级联使能位，高有效。 若为 1，则 LTIMERn 将在 LTIMERn-1 溢出时递减 1。 若为 0，级联功能关闭。
[4]	IP	定时器溢出中断标志位。 若定时器溢出中断被使能(bit3=1)，当定时器溢出时，该位被置 1。 往该位写入 1 时，可将该位清零。 往该位写入写入 1，无效。

位	位名称	位描述
[3]	IE	定时器溢出中断使能位，高有效。
[2]	LD	定时器重载命令。 写入 1，定时器将立即从重载值寄存器中获取定时器重新定时计数。 写入 0，无效。
[1]	RS	定时器自动重载使能，高有效。
[0]	EN	定时器使能位，高有效。

11.2.8 锁存定时器锁存值寄存器

表 11-9 锁存定时器的锁存值寄存器(LTIMER LATCH VALUE)

位	位名称	位描述
[31:0]	LATCH VALUE	锁存定时器的锁存置。

12. 通用输入输出接口 GPIO

计算机模块集成64位的通用输入输出接口GPIO（64位的GPIO被分成两组GPIO[63:32]和GPIO [31:0]），每一个GPIO都可以独立配置成输入口或输出口。GPIO[63:32]还可以作为外部中断的输入接口。计算机模块的GPIO与芯片的其它信号复用，具体情况请查看“封装及信号”章节。

12.1 GPIO 的工作原理

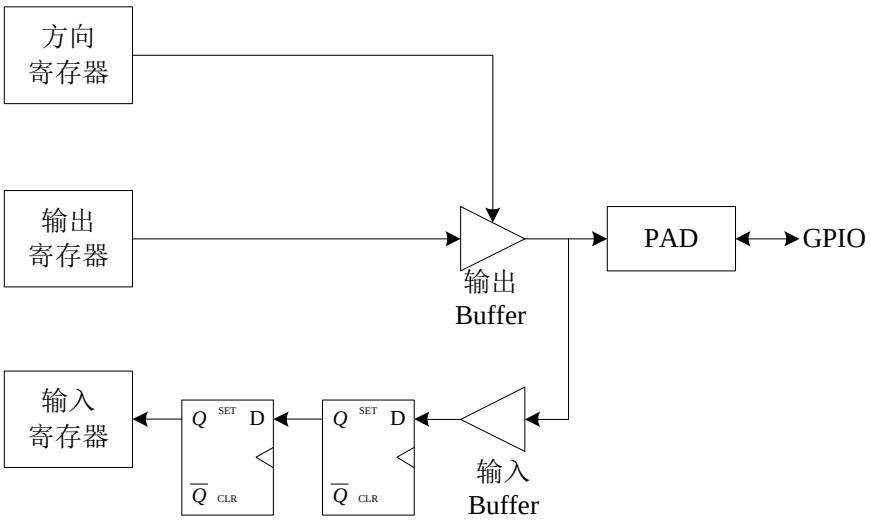


图 12-1 计算机模块 GPIO 结构示意图

GPIO口的输入信号经过两个D触发器，可以滤出一定的干扰和毛刺，然后才进入输入寄存器。

GPIO的方向设置：通过设置GPIO方向寄存器的相应位可以设置对应GPIO口的方向，1表示输出，0表示输入。所有的GPIO复位默认为输入状态。

GPIO的输入状态获取：首先要将要目标GPIO设为输入状态，然后读取GPIO输入寄存器，便可以获得目标GPIO口的输入状态。

GPIO输出设定：首先要将要目标GPIO设为输出状态，然后往GPIO输出寄存器的对应位写入期望输出的值，便可实现GPIO口输出设定。

GPIO外部中断扩展：首先设置外部中断映射配置寄存器，将GPIO外部中断接到目标中断源；第二步操作外部中断方式寄存器，决定是采用电平还是边沿来触发中断；第三步操作外部中断极性寄存器，设定是中断触发的极性；第四步操作外部中断屏蔽寄存器，使能相应的外部中断。

12.2 GPIO 寄存器

表 12-1 GPIO 寄存器地址分配

地址	读/写	有效位宽	默认值(HEX)	寄存器描述
0X80000600	R/W	32	00000000	GPIO[31:0]数据输入寄存器(INPUT)
0X80000604	R/W	32	00000000	GPIO[31:0]数据输出寄存器(OUTPUT)
0X80000608	R/W	32	00000000	GPIO[31:0]方向寄存器(DIRECTION)
0X80100400	R/W	32	00000000	GPIO[63:32]数据输入寄存器(INPUT)
0X80100404	R/W	32	00000000	GPIO[63:32]数据输出寄存器(OUTPUT)
0X80100408	R/W	32	00000000	GPIO[63:32]方向寄存器(DIRECTION)
0X8010040C	R/W	32	00000000	GPIO[63:32] 外部中断屏蔽寄存器(INT MASK)
0X80100410	R/W	32	00000000	GPIO[63:32] 外部中断极性寄存器(INT POLAR)
0X80100414	R/W	32	00000000	GPIO[63:32] 外部中断方式寄存器(INT EDGE)
0X80100420	R/W	32	00000000	外部中断映射配置寄存器 1，负责配置 GPIO[32:35]
0X80100424	R/W	32	00000000	外部中断映射配置寄存器 2，负责配置 GPIO[36:39]
0X80100428	R/W	32	00000000	外部中断映射配置寄存器 3，负责配置 GPIO[40:43]
0X8010042C	R/W	32	00000000	外部中断映射配置寄存器 4，负责配置 GPIO[44:47]
0X80100430	R/W	32	00000000	外部中断映射配置寄存器 5，负责配置 GPIO[48:51]
0X80100434	R/W	32	00000000	外部中断映射配置寄存器 6，负责配置 GPIO[52:55]
0X80100438	R/W	32	00000000	外部中断映射配置寄存器 7，负责配置 GPIO[56:59]
0X8010043C	R/W	32	00000000	外部中断映射配置寄存器 8，负责配置 GPIO[60:63]

12.2.1 GPIO 数据输入寄存器

表 12-2 GPIO 数据输入寄存器(INPUT)

位	位名称	位描述
[31:0]	Input	输入 GIPO 数据值

12.2.2 GPIO 数据输出寄存器

表 12-3 GPIO 数据输出寄存器(OUTPUT)

位	位名称	位描述
[31:0]	Output	输出 GPIO 数据值

12.2.3 GPIO 方向寄存器

表 12-4 GPIO 方向寄存器(DIRECTION)

位	位名称	位描述
[31:0]	Direction	GPIO 方向寄存器值，(0=输入，1=输出)

12.2.4 GPIO 外部中断屏蔽寄存器

表 12-5 GPIO 外部中断屏蔽寄存器(INT MASK)

位	位名称	位描述
[31:0]	Interrupt Mask	外部中断屏蔽，(0=屏蔽，1=中断允许) 该域的 bit[n]设置 GPIO[32+n]，其中 0≤n≤31；

12.2.5 GPIO 外部中断极性寄存器

表 12-6 GPIO 外部中断极性寄存器(INT POLA)

位	位名称	位描述
[31:0]	Interrupt Polarity	中断极性，(0=低点平/下降沿，1=高点平/上升沿) 该域的 bit[n] 设置 GPIO[32+n]，其中 0≤n≤31；

12.2.6 GPIO 外部中断方式寄存器

表 12-7 GPIO 外部中断方式寄存器(EDGE)

位	位名称	位描述

位	位名称	位描述
[31:0]	Interrupt Edge	中断方式，（0=电平，1=边沿） 该域的 bit[n] 设置 GPIO[32+n]，其中 0≤n≤31；

12.2.7 GPIO 外部中断映射配置寄存器

表 12-8 GPIO 外部中断映射配置寄存器 n（0≤n≤7）

位	位名称	位描述
[31:29]	RES	保留，读值为 0。
[28:24]	IRQMAP[32+4*n]	GPIO[32+4*n]外部中断映射。 若该域设为 K（1≤n≤31），则 GPIO[32+4*n]外部中断被连接到第 K 号中断源。
[23:21]	RES	保留，读值为 0。
[20:16]	IRQMAP[32+4*n+1]	GPIO[32+4*n+1]外部中断映射。 若该域设为 K（1≤n≤31），则 GPIO[32+4*n+1]外部中断被连接到第 K 号中断源。
[15:13]	RES	保留，读值为 0。
[12:8]	IRQMAP[32+4*n+2]	GPIO[32+4*n+2]外部中断映射。 若该域设为 K（1≤n≤31），则 GPIO[32+4*n+2]外部中断被连接到第 K 号中断源。
[7:5]	RES	保留，读值为 0。
[4:0]	IRQMAP[32+4*n+3]	GPIO[32+4*n+3]外部中断映射。 若该域设为 K（1≤n≤31），则 GPIO[32+4*n+3]外部中断被连接到第 K 号中断源。

13. 多功能引脚配置寄存器 GPREG

13.1 概述

该寄存器主要用来设置多功能引脚SP[17-69]，是作专用模块引脚，还是作通用GPIO引脚。

表 13-1 多功能引脚配置 GPREG 寄存器地址

地址	寄存器

0x80100500	通用寄存器 31：0

13.1.1 GPREG 通用寄存器

表 13-2 多功能引脚配置 GPREG 寄存器

位	位名称	位描述
[31:11]	RES	保留
9	tc_en	默认值为“0”，作为 GPIO 口；当该位为“1”用作 TM/TC-TC 模块的功能脚。对应 SP[69-50]引脚。
8	tm_en	默认值为“0”，作为 GPIO 口；当该位为“1”用作 TM/TC-TM 模块的功能脚。对应 SP[49-47]引脚。
7	spw_en(3)	默认值为“0”，作为 GPIO 口；当该位为“1”用作 SPW-4 模块的功能脚。对应 SP[46-43]引脚。
6	spw_en(2)	默认值为“0”，作为 GPIO 口；当该位为“1”用作 SPW-3 模块的功能脚。对应 SP[42-39]引脚。
5	spi_en	默认值为“0”，作为 GPIO 口；当该位为“1”用作 SPI 模块的功能脚。对应 SP[38-36]引脚。
4	I2C_en	默认值为“0”，作为 GPIO 口；当该位为“1”用作 I2C 模块的功能脚。对应 SP[35-34]引脚。
3	can_en	默认值为“0”，作为 GPIO 口；当该位为“1”用作 CAN-1 模块的功能脚。对应 SP[33-32]引脚。
2	m1553_en	默认值为“0”，作为 GPIO 口；当该位为“1”用作 1553-1 模块的功能脚。对应 SP[21-31]引脚。
1	uart_en(3)	默认值为“0”，作为 GPIO 口；当该位为“1”用作 UART3 模块的功能脚。对应 SP[18-17]引脚。
0	uart_en(2)	默认值为“0”，作为 GPIO 口；当该位为“1”用作 UART2 模块的功能脚。对应 SP[20-19]引脚。

14. I²C 总线主控制器

14.1 概述

I²C是一种串行数据传输的标准总线。模块中的I²C master控制器与飞利浦I²C标准兼容，支持7位和10位的地址。标准模式工作下速率为100kb/s，快速模式为400kb/s。

I²C主控制器结构框图如下：

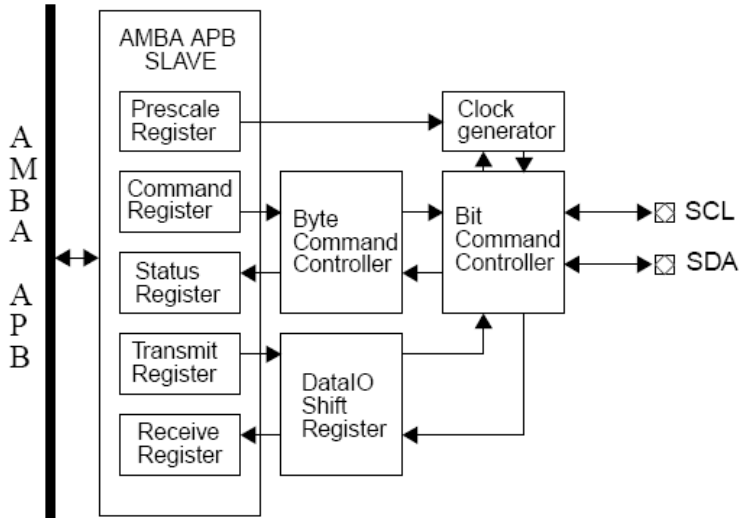


图 14-1 I²C master 结构框图

14.2 I²C 工作原理

I²C总线具备冲突检测和裁决，包括串行数据线(SDA)和串行时钟线(SCL)。I²C总线数据传输开始于START状态。当SCL为高时，SDA从高电平转换到低电平时的状态为START状态。I²C总线数据传输结束于STOP状态，当SCL为高时，SDA从低电平转换到高电平的状态为STOP状态。I2C总线在START状态后处于忙状态，在STOP状态结束后一段时间处于空闲状态。

下图为I²C数据传输示意图，master首先产生一个START状态，然后发送一个7位的slave地址，跟着slave地址之后的是1位读写控制位。当读写控制位为0时，为写操作，为1时，为读操作。当传输完地址和读写控制信号后，释放SDA信号线。接收端把SDA信号线拉低作为对

发送端的响应信号。如果接收端没有响应发送端，master会产生一个STOP状态来中止传输或者产生一个重复的START状态来开始一个新的传输。

在第一个字节传输得到响应后，master开始传输数据。如果读写控制位设置为1，master将作为这个数据传输周期中的接收端来读取数据。在数据传输完毕并且得到响应之后，master产生一个STOP状态，表示传输完成。如果对于一个slave设备来说，数据bitrate太高，则可以通过在master驱动SCL为低电平之后通过保持SCL为低电平的时间来延长时钟周期。

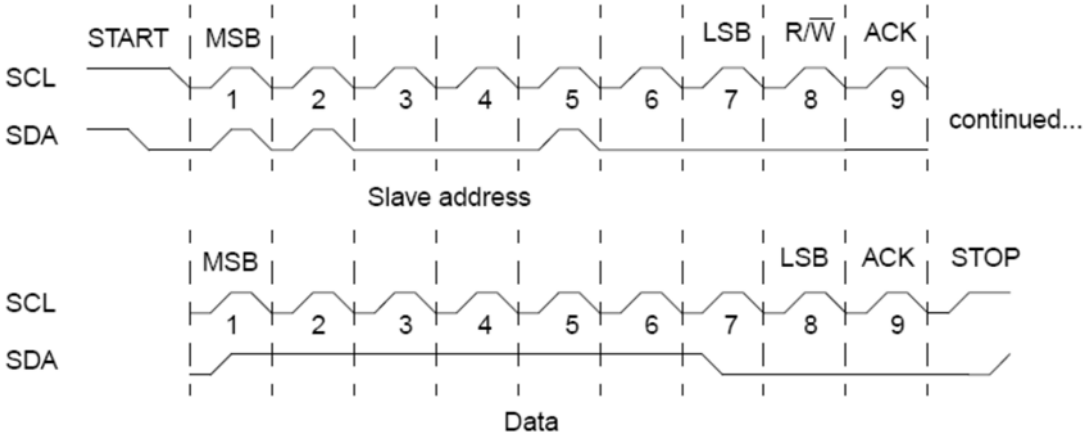


图 14-2 I2C 数据传输示意图

通过设置prescale寄存器来决定SCL时钟线的频率，采用下面的公式来计算prescale值：

$$Prescale=\frac{SYSCLK}{5*SCLfrequency}-1 \qquad （式 12－1）$$

标准模式下SCL频率为100kHz，快速模式下SCL频率为400kHz，当SYSCLK频率为60MHz时，为了使I^2C内核工作在标准模式，则需要的prescale值如下

$$Prescale=\frac{60MHz}{5*100kHz}-1=119=0x77 \qquad （式 12－2）$$

注意只有在I^2C内核被disabled后才能修改prescale寄存器值。

14.3 I^2C-master 寄存器

I^2C-master 寄存器列表如下：

表 14-1 I²C-master 寄存器

地址	读写	寄存器说明
0x80000500	R/W	时钟 prescale 寄存器
0x80000504	R/W	控制寄存器
0x80000508	W	发送器寄存器
0x80000508	R	接收器寄存器
0x8000050C	W	命令寄存器
0x8000050C	R	状态寄存器
0x80000510	R/W	动态滤波器寄存器(某些版本可用)

14.3.1 I²C -master 时钟 prescale 寄存器

表 14-2 I²C-master 时钟 prescale 寄存器

位	位名称	位描述
[31:16]	RES	保留
[15:0]	Clock Prescale	时钟 prescale 值：该值用与预设 SCL 时钟线值，不要修改该寄存器值，除非控制寄存器的 EN 使能信号为 0。最小推荐值为 0x0003。更低的值会使 master 与 I2C 时序要求向冲突。

14.3.2 I²C -master 控制寄存器

表 14-3 I²C-master 控制寄存器

位	位名称	位描述
[31:8]	RES	保留
[7]	EN	使能(EN)：使能 I2C 功能模块。该位置 1，开启
[6]	IEN	中断使能(IEN)：该位置 1 时，数据传输完成后会产生中断信号
[5:0]	RES	保留

14.3.3 I²C-master 发送寄存器

表 14-4 I²C-master 发送寄存器

位	位名称	位描述
[31:8]	RES	保留
[7:1]	TDATA	发送数据(TDATA)：I2C 总线将要发送的数据，高位为 MSB
[0]	RW	写/读(RW)：读写控制位，在传输中，1 表示读，0 表示写

14.3.4 I²C -master 接收寄存器

表 14-5 I²C-master 接收寄存器

位	位名称	位描述
[31:8]	RES	保留
[7:0]	RDATA	接收数据(RDATA)：通过 I2C 总线接收到的最后 1 个字节数据

14.3.5 I²C -master 命令寄存器

表 14-6 I²C-master 命令寄存器

位	位名称	位描述
[31:8]	RES	保留
[7]	STA	开始(STA)：在 I2C 总线上产生 START 状态。也用于产生重复 START 状态
[6]	STO	停止(STO)：产生停止状态
[5]	RD	读(RD)：从 slave 上读数据
[4]	WR	写(WR)：往 slave 写数据
[3]	ACK	响应(ACK)：I2C-master 作为接收器时使用该位，‘0’发送 ACK 信号，‘1’发送 NACK
[2:1]	RES	保留
[0]	IACK	中断响应(IACK)：清除状态寄存器中的中断标志信号

14.3.6 I²C -master 状态寄存器

表 14-7 I²C-master 状态寄存器

位	位名称	位描述
[31:8]	RES	保留
[7]	RxACK	接收响应(RxACK)：从 slave 接收响应。没有接收到响应信号，该位为‘1’。slave 响应了传输动作，则为‘0’
[6]	BUSY	I²C 总线忙(BUSY)：当检测到 start 信号，该位为‘1’。检测到 stop 信号后该位置‘0’
[5]	AL	仲裁丢失(AL)：当内核丧失仲裁，该位为‘1’。当检测到 stop 信号却没有被请求信号或者 master 驱动 SDA 信号到高，而实际 SDA 为低时，该情况发生
[4:2]	RES	保留
[1]	TIP	传输进度(TIP)：正在传输数据，该位为‘1’，完成数据传输，该位为‘0’
[0]	IF	中断标志(IF)：当完成一个字节的传输并且不需要裁决的时候，该位置‘1’。如果控制寄存器的 IEN 置‘1’，将会产生中断。即使该位没有清除，也会产生新的中断

14.3.7 I²C-master 动态滤波器寄存器

表 14-8 I²C-master 动态滤波器寄存器

位	位名称	位描述
[31:x]	RES	保留
[x-1:0]	FILT	动态滤波器重载值(FILT)：设置动态滤波器计数器的重载值。模块将忽略所有时间长度短于 2*重载值*（系统时钟周期）-1 的信号。 此位初始值为全‘1’。

$$FILT = (\frac{SYSCLK}{5 * SCLK_{frequency}} - 1)/10 \qquad （式 12－3）$$

SYSCLK 为系统时钟，如果计算结果 FILT 的值小于 1 则设为 1，如果计算结果 FILT 的值大于 1 则取整。

15. 调试支持模块 DSU

15.1 DSU 简介

为方便计算机模块的硬件调试工作，该计算机模块在设计之初，添加 DSU 调试模块，在调试模式下，DSU 模块直接控制计算机模块内核，通过串口、以太网口可以很方便读取计算机模块内核里的寄存器、设置断点、设置配置信息、读取外部 FLASH、SRAM 等工作。同时 DSU 模块支持控制多核系统，可以任意控制模块内的任一个内核。

计算机模块的 DSU 模块结构图如下图 。

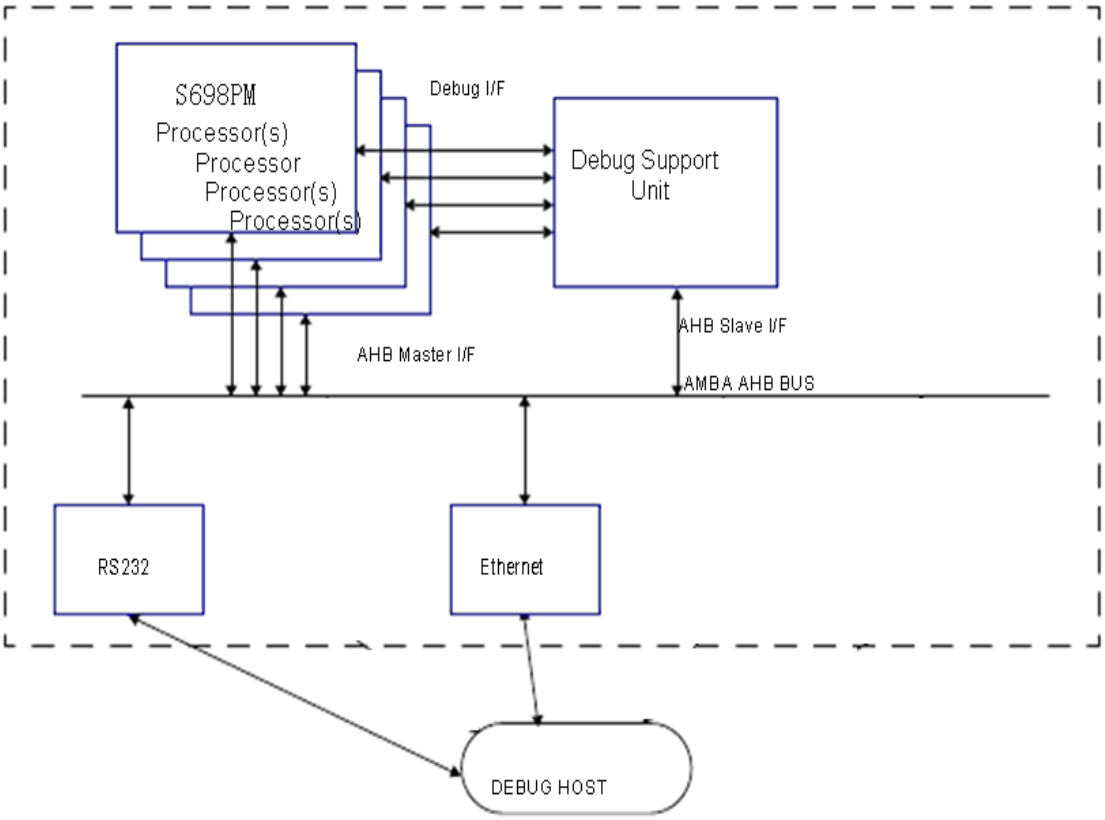


图 15-1 计算机模块内部 DSU 模块结构图

15.2 DSU 工作原理

通过 DSU AHB 从设备接口，任何 AHB 主机可以访问处理器寄存器和命令跟踪缓存区中的内容，相对处理器只有进入调试模式时，处理器寄存器，缓存和命令跟踪缓存区才能被访问的而言，DSU 控制寄存器却可以在任何时刻都能够访问。在调试模式里，处理器的流水线和处理器状态都可以被 DSU 模块访问。下列事件可以触发进入调试模式：

- 执行断点命令（ta 1）
- 整数单元硬件断点、观察点触发（trap 0xb）
- 外部断点信号（DSUBRE）的上升沿触发
- 在 DSU 控制寄存器设置断点位（BN）
- 引起处理器进入错误模式的陷阱
- 在 DSU 控制寄存器设置的陷阱产生
- 单步执行操作后
- 在多核系统中，某个单核进入调试模式

- DSU AHB 断点或观测点触发
- 通过外部信号（DSUEN）使能 DSU 单元后，才能够进入调试模式。当进入调试模式时，
- 下列动作发生：
- PC 和 nPC 被保存在临时寄存器（调试单元可以访问的）里
- 输出信号（DSUACT）被设计成指示调试状态
- 定时器单元冻结计算机模块定时器和看门狗

当处理器处于调试模式时，ASI 诊断区域的访问将指向执行访问地址为原始地址的低 20 位 DSU ASI 寄存器数值的 IU 单元。

15.3 DSU 寄存器映射表

DSU 模块寄存器的基地址为：0x90000000,偏移地址及映射表 15-1 内容参考下。在计算机模块四核系统中，每个内核的地址分别由地址线 27~24 决定，因此他们为 0x90000000、0x91000000、0x92000000、0x93000000；

15.3.1 DSU 寄存器映射表

表 15-1 DSU 寄存器映射表

偏移地址	读/写	寄存器描述
0x000000	R/W	DSU 控制寄存器
0x000008	R/W	时间标识寄存器
0x000020	R/W	断点和单步寄存器
0x000024	R/W	DSU 调试模式控制寄存器
0x000040	R/W	踪迹缓存控制寄存器
0x000044	R/W	踪迹缓存索引寄存器
0x000048	R/W	踪迹缓存过滤控制寄存器
0x00004c	R/W	踪迹缓存过滤标识寄存器
0x000050	R/W	断点 1 的地址寄存器
0x000054	R/W	AHB 标识寄存器 1
0x000058	R/W	断点 2 的地址寄存器

0x00005c	R/W	AHB 标识寄存器 2
0x000070	R/W	命令计数寄存器
0x000080	R/W	AHB 观测点控制寄存器
0x000090 – 0x00009C	R	AHB 观测点 1 数据寄存器
0x0000A0 – 0x0000AC	R	AHB 观测点 1 标识寄存器
0x0000B0 – 0x0000BC	R	AHB 观测点 2 数据寄存器
0x0000C0 – 0x0000CC	R	AHB 观测点 2 标识寄存器
0x100000 – 0x10FFFF	R	命令踪迹缓存
0x110000	R/W	命令踪迹缓存控制寄存器
0x200000 – 0x210000	R	AHB 踪迹缓存
0x300000 – 0x3007FC	R/W	IU 寄存器文件 %on: 0x300000 + (((psr.cwp * 64) + 32 + n*4) mod (NWINDOWS*64)) %ln: 0x300000 + (((psr.cwp * 64) + 64 + n*4) mod (NWINDOWS*64)) %in: 0x300000 + (((psr.cwp * 64) + 96 + n*4) mod (NWINDOWS*64)) %gn: 0x300000 + (NWINDOWS*64) + n*4 %fn: 0x301000 + n*4
0x301000 – 0x30107C	R/W	FPU 寄存器文件
0x400000	R	Y 寄存器
0x400004	R	PSA 寄存器

0x400008	R	WIM 寄存器
0x40000C	R	TBR 寄存器
0x400010	R	PC 寄存器
0x400014	R	NPC 寄存器
0x400018	R	FSR 寄存器
0x40001C	R	CPSR 寄存器
0x400020	R	DSU 陷阱寄存器
0x400024	R	DSU ASI 寄存器
0x400040 - 0x40007C	R/W	ASR16-ASR31 寄存器
0x700000 - 0x7FFFFC	R/W	ASI 寄存器访问 ASI = 0x9 : Local instruction RAM ASI = 0xB : Local data RAM ASI = 0xC : Instruction cache tags ASI = 0xD : Instruction cache data ASI = 0xE : Data cache tags ASI = 0xF : Data cache data ASI = 0x1E : Separate snoop tags

15.3.2 DSU 控制寄存器

表 15-2 DSU 控制寄存器

位	位名称	位描述
[31:12]	RES	保留
[11]	PW	掉电状态位（PW）-当内核掉电后，该值为‘1’
[10]	HL	内核关闭状态位（HL）
[9]	PE	内核错误模式状态位（PE）
[8]	EB	外部 DSUBRE 状态值（EB）
[7]	EE	外部 DSUEN 状态值（EE）
[6]	DM	调试模式状态位（DM）

位	位名称	位描述
[5]	BZ	错误 TRAP 中断使能位（BZ）
[4]	BX	TRAP 中断使能位（BX）
[3]	BS	断点中断使能位（BS）
[2]	BW	IU 观测点中断使能位（BW）
[1]	BE	错误中断使能位（BE）
[0]	TE	踪迹使能位（TE）

15.3.3 DSU 断点和单步寄存器

表 15-3 DSU 断点和单步寄存器

位	位名称	位描述
[31:16]	SSx	单步设定值（SSx）-如果该位设置后，对应的内核进入调试模式，执行单个命令
[15:0]	BNx	断点值（BNx）-强制对应的内核进入调试模式

15.3.4 DSU 调试模式控制寄存器

表 15-4 DSU 调试模式控制寄存器

位	位名称	位描述
[31:16]	DMx	调试模式控制位（DMx）-如果设定，对应的内核能够进入调试模式
[15:0]	EDx	调试模式使能位（EDx）-如果设定，对应的内核进入调试模式

15.3.5 DSU 陷阱寄存器

表 15-5 DSU 陷阱寄存器

位	位名称	位描述
[31:13]	RES	保留
[12]	EM	错误模式使能（EM）-如果设定，陷阱将使内核进入错误模式
[11:4]	TRAP TYPE	陷阱类型（TRAP TYPE）-8 位 SPARC 陷阱类型

位	位名称	位描述
[3:0]	-	默认值 0x0

15.3.6 DSU 踪迹缓存时间标识计数器

表 15-6 DSU 踪迹缓存时间标识寄存器

位	位名称	位描述
[31:30]	-	默认为 0x0
[29:0]	TIMETAG	DSU 时间标识值（TIMETAG）

15.3.7 DSU ASI 寄存器

表 15-7 DSU ASI 寄存器

位	位名称	位描述
[31:8]	RES	保留
[7:0]	ASI	设定 ASI 访问值

15.3.8 DSU 踪迹缓存控制寄存器

表 15-8 DSU 踪迹缓存控制寄存器

位	位名称	位描述
[31:16]	DCNT	踪迹缓存延迟计数（DCNT）
[15:8]	RES	保留
[7]	SF	采样设置（SF）
[6]	TE	时间使能（TE）
[5]	TF	踪迹使能（TF）
[4:3]	BW	总线位宽（BW）
[2]	BR	断点（BR）
[1]	DM	计数模式（DM）
[0]	EN	踪迹缓存使能位（EN）

15.3.9 DSU 踪迹缓存索引寄存器

表 15-9 DSU 踪迹缓存索引寄存器

位	位名称	位描述
[31:4]	INDEX	踪迹缓存模式计数（INDEX）
[3:0]	-	默认值为 0x0

15.3.10 DSU 踪迹缓存过滤控制寄存器

表 15-10 DSU 踪迹缓存过滤控制寄存器

位	位名称	位描述
[31:14]	RES	保留
[13:12]	WPF	AHB 观测点过滤（WPF）
[11:10]	RES	保留
[9:8]	BPF	AHB 断点过滤（BPF）
[7:4]	RES	保留
[3]	PF	计数过滤（PF）
[2]	AF	地址过滤（AF）
[1]	FR	过滤读（FR）
[0]	FW	过滤写（FW）

15.3.11 DSU 踪迹缓存过滤标识寄存器

表 15-11 DSU 踪迹缓存过滤标识寄存器

位	位名称	位描述
[31:16]	SMASK	从设备标识（SMASK）
[15:0]	MMASK	主设备标识（MMASK）

15.3.12 DSU 踪迹缓存断点寄存器

表 15-12 DSU 踪迹缓存断点寄存器

位	位名称	位描述
[31:2]	BRADDR	断点地址（BRADDR）
[1:0]	-	默认为：0x0

15.3.13 DSU 命令踪迹控制寄存器

表 15-13 DSU 命令踪迹控制寄存器

位	位名称	位描述
[31:28]	ITRACE CFG	踪迹过滤配置位
[27:16]	RES	保留
[15:0]	ITPOINTER	命令踪迹指针（ITPOINTER）

15.3.14 DSU 命令计数寄存器

表 15-14 DSU 命令计数寄存器

位	位名称	位描述
[31]	CE	计数使能位（CE）
[30]	IC	命令计数开始（IC）
[29]	PE	过滤使能（PE）
[28:0]	ICOUNT	命令计数（ICOUNT）

15.3.15 AHB 观测控制寄存器

表 15-15 AHB 观测控制寄存器

位	位名称	位描述
[31:7]	RES	保留
[6]	IN	观测点 2 反转（IN）
[5]	CP	观测点 2 连接（CP）

位	位名称	位描述
[4]	EN	观测点 2 使能（EN）
[3]	RES	保留
[2]	IN	观测点 1 反转（IN）
[1]	CP	观测点 1 连接（CP）
[0]	EN	观测点 1 使能（EN）

15.3.16 AHB 观测点数据寄存器

表 15-16 AHB 观测点数据寄存器

位	位名称	位描述
[31:0]	DATA	AHB 观测点数据（DATA）

16. JTAG 接口控制器

16.1 概述

JTAG TAP控制器提供了一个测试访问接口，该接口符合IEEE-1149（JTAG）标准。该模块实现了jtag测试访问端口信号，同步的测试状态机实现，以及一组jtag测试的数据寄存器。内部将JTAG指令转化为AHB的总线信息传输，可以通过jtag读写AHB总线的任何一个地址。当DSU_EN无效的时候，JTAG无法使用。

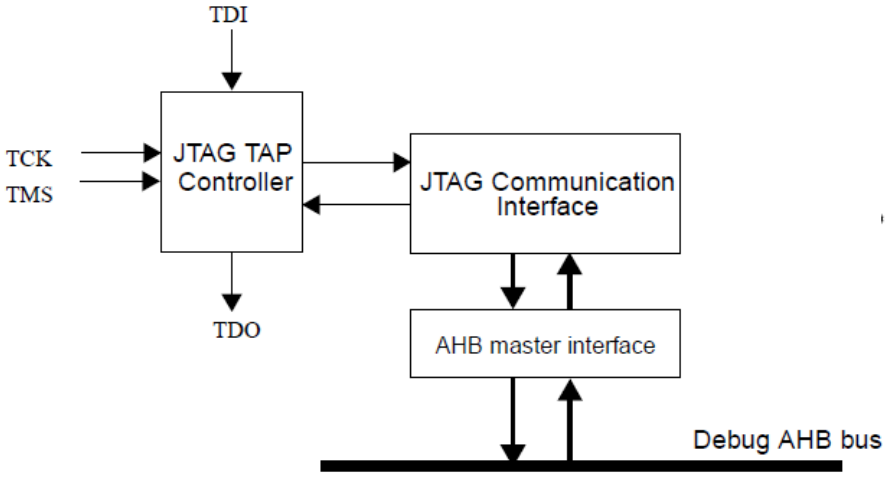


图 16-1 jtag 控制器模块图

16.2 功能说明

该控制器的测试引脚包括TCK,TMS,TDI,TDO。内部通过同步状态机进行控制，该状态机控制指令寄存器以及数据寄存器的状态。控制器能够工作在bypassmod。控制可以移入指令数据并更新指令寄存器。用户自定义的测试寄存器可以通过用户定义数据寄存器接口进行访问。

指令寄存器中的内容在移入扫描数据的时候被移出到端口，可以通过该信息判断控制器的工作状态，如捕获数据的阶段，移动数据，更新数据寄存器内容等。

需要扫描内部寄存器时，首先通过TDI逐位送入扫描指令信息，然后通过TDI再此送入相关参数，或者直接通过TMS的时钟信号直接扫描出内部数据寄存器的内容即可。

计算机模块 JTAG调试口解码两种JTAG指令，有两个JTAG数据寄存器：命令/地址寄存器和数据寄存器。一次读访问可以通过移位的方式送入带有读写信息、AHB访问大小、以及AHB地址的命令来完成，写访问同样是通过移入AHB访问大小、AHB地址到命令寄存器，然后往数据寄存器中移入需要写入的数据来完成。连续传输可以通过将起始地址移入命令/地址寄存器并且移入SEQ到数据寄存器。在随后的访问中SEQ将会增加，但连续传输不能越过1K边界，连续传输只能是字传输。

通过数据寄存器的32bit，JTAG 调试口通知一次AHB访问是否完成，如果数据寄存器的32bit是‘1’，那么表示一次AHB访问已经完成，如果是‘0’表示一次AHB访问还没有结束，如果AHB访问还没有结束，那么调试程序应该等待，并不断检查数据寄存器32bit，知道其为‘1’。如果是移入数据的时候该32bit为‘0’，同样需要等待，为‘0’时，不能移入新的数

据。

16.3 寄存器说明

16.3.1 JTAG 命令/地址寄存器

表 16-1 JTAG 命令/地址寄存器

位	位名称	位描述
[34]	W	0：读传输　1：写传输
[33:32]	SIZE	AHB 传输位宽 “00”：字节 01”：半字 10”：字 “11”：保留
[31:0]	AHB ADDRESS	AHB 访问地址

16.3.2 JTAG 数据寄存器

表 16-2 JTAG 数据寄存器

位	位名称	位描述
[32]	SEQ	当移入数据时：1 ：表示连续传输，下次数据传输的地址是当前地址加 1　0：非连续传输 当移出数据时：1：访问完成　0：访问没有完成
[31:0]	AHB DATA	AHB 数据

17. 外部存储控制器

17.1 存储控制器简介

存储控制器控制一个连有 PROM, 存储器映 I/O 设备, 静态存贮器和动态随机存贮器的存储器总线。控制器作为 AHB 总线上的从设备。存储器控制器的功能通过 APB 总线访问存储器控制寄存器 1 和 2(MCFG1, MCFG2)进行配置。存储器总线支持四种类型的设备：prom，静态存贮器, 动态随机存贮器 和 I/O。存储器总线支持 8、16、32 位工作模式。其中 PROM 和

SRAM 可以通过 BCD 编码模块进行 EDAC 保护，EDAC 保护可以提供发现两个错误和纠正一个错误的容错能力，同时，EDAC 功能可以通过使能脚来控制是否需要 EDAC 功能模块参数数据传递。

17.2 存储地址分配

表 17-1 存储器控制器地址分配表

地址空间	描述	大小
0x00000000—0x1FFFFFFF	ROM 区	512M
0x20000000—0x3FFFFFFF	I / O 区	512M
0x40000000—0x5FFFFFFF	片外 RAM 区	512M
0x60000000—0x7FFFFFFF	片外 DDR2 区	512M

17.3 存储器控制寄存器

17.3.1 存储器寄存器地址分配

表 17-2 EDAC 寄存器地址分配列表

地址	寄存器	读/写	描述
0x80000000	MCFG1	RW	存储器配置寄存器 1
0x80000004	MCFG2	RW	存储器配置寄存器 2
0x80000008	MCFG3	RW	存储器配置寄存器 3
0x80000440	DDR2FTCFG	RW	普通 DDR2 FT 配置寄存器

17.3.2 存储器配置寄存器 1（MCFG1）

表 17-3 存储器配置寄存器 1

位	位名称	位描述
[31]	RES	保留
[30]	PBRDY	PROM 总线就绪使能,当该位为“1”使能，外部 BRDYN 总线输入信号有效，否则忽略外部输入 BRDYN 信号；默认值为“0”
[29]	ABRDY	外部 BRDYN 总线输入信号有效时长设置，当该位为“1”，外部 BRDYN 总线输入信号低电平有效至少保持 3 个 sysclk 时钟周期；当该位为“0”，外部 BRDYN 总线输入信号低电平有效至少保持 2 个 sysclk 时钟周期,；默认值为“0”
[28:27]	IOBUSW	I/O 总线宽度设置 “00”=8，“01”=16，“10”=32

位	位名称	位描述
[26]	IBRDY	外部 BRDYN 输入信号使能位,当该位为“1”使能，外部 BRDYN 总线输入信号有效，否则忽略外部输入 BRDYN 信号；默认值为“0”
[25]	BEXCN	PRAM,RAM,I/O 所有总线错误信号使能；当该位为“1”使能，外部 BEXCN 总线输入信号有效，否则忽略外部输入 BRDYN 信号；默认值为“0”
[24]	RES	保留
[23:20]	IO WAITSTATES	I/O 区读写等待时间长度配置，“0000”=0，“0001”=1，“0010”=2，…“1111”=15
[19]	IOEN	使能存储器 I/O 区访问
[18]	RES	保留
[17:14]	ROMBANKSZ	PROM bank 大小信息,每个 Bank 固定为 256MB,2 个 PROM 片选信号,最大可访问 512MB PROM。
[13:12]	RES	保留
[11]	PWEN	使能 PROM 写
[10]	RES	保留
[9:8]	PROM WIDTH	设置 PROM 位宽. “00”=8，“01”=16，“10”=32,默认值由外部输入脚值决定
[7:4]	PROM WRITE WS	设置 PROM 写周期的长度，“0000”=0，“0001”=2，“0010”=4，…“1111”=30，默认值为“1111”。PROM 写等待时间公式为 WS=(PROM WRITE WS)*8,如写入数据为”0010”则写 PROM 区等待时间为 4*8*Tsysclk
[3:0]	PROM READ WS	设置 PROM 读周期的长度，“0000”=0，“0001”=2，“0010”=4，…“1111”=30，默认值为“1111”。PROM 读等待时间公式为 WS=(PROM READ WS)*8,如写入数据为”0010”则读 PROM 区等待时间为 4*8*Tsysclk

17.3.3 存储器配置寄存器 2（MCFG2）

表 17-4 存储器配置寄存器 2

位	位名称	位描述
[31]	SDRF	保留
[30]	TRP	保留
[29:27]	TRFC	保留
[26]	TCAS	保留
[25:23]	SDRAM BANKSZ	保留
[22:21]	SDRAM COLSZ	保留
[20:19]	SDRAM CMD	保留
[18]	D64	保留

位	位名称	位描述
[17]	RES	保留
[16]	MS	保留
[15]	RES	保留
[14]	SE	保留
[13]	SI	SRAM 禁止访问使能，该位为 1 禁示访问 SRAM
[12:9]	RAM BANK SIZE	RAM bank 大小设置。“0000”=8kbyte， “0001”=16kbyte，…“1111”=256Mbyte
[8]	RES	保留
[7]	RBRDY	RAM 总线读就绪使能,当该位为“1”使能，外部 BRDYN 总线输入信号有效，否则忽略外部输入 BRDYN 信号；默认值为“0”
[6]	RMW	Read-modify-write 使能,只有在 16bit,32bit 总线宽度下有效
[5:4]	RAM WIDTH	RAM 数据位宽。“00”=8，“01”=16，“1X”=32
[3:2]	RAM WRITE WS	RAM 写周期长度“00”=0，“01”=1，“10”=2，“11”=3，默认值为“11”。RAM 写等待时间公式为 WS=(RAM WRITE WS)*4,如写入数据为”10”则写 RAM 区等待时间为 2*4*Tsysclk
[1:0]	RAM READ WS	RAM 读周期长度“00”=0，“01”=1，“10”=2，“11”=3，默认值为“11”。RAM 读等待时间公式为 WS=(RAM READ WS)*4,如写入数据为”10”则读 RAM 区等待时间为 2*4*Tsysclk

17.3.4 存储器配置寄存器 3（MCFG3）

表 17-5 存储器配置寄存器 3

位	位名称	位描述
[31:27]	RES	保留
[28]	RSE	保留
[27]	ME	存储器 EDAC 使能标识位，当 EDAC 使能有效，则该位读出为 1，该位为只读位，不可写
[26:12]	RES	保留

位	位名称	位描述
[11]	WB	EDAC 写旁路诊断寄存器
[10]	RB	EDAC 读旁路诊断寄存器
[9]	RE	RAM EDAC 保护使能位(带 EDAC 保护的 SRAM 位宽要选择 32BIT)。采用的是 BCH EDAC with pipeling
[8]	PE	PROM EDAC 保护使能(带 EDAC 保护的 PROM 位宽要选择 32BIT)。采用的是 BCH EDAC with pipeling
[7:0]	TCB	EDAC 校验位测试域。当 WB 置 1 时，将测试校验位写入该域供诊断使用，当 RB 置 1 时，读取到的数据校验位将被存储在该域供诊断使用

17.4 EDAC 控制器

17.4.1 概述

计算机模块支持四种类型的存储设备：PROM、异步静态RAM（SRAM）、同步动态RAM（DDR2）及存储器空间映射的IO设备。其中PROM、SRAM和DDR2都通过EDAC进行了数据保护，可以通过该数据保护功能实现纠正一位数据错误，检出两位数据错误。其中DDR2使用reed-solomon纠错算法，可实现邻近4位错误的纠正能力，32位的数据需要16位的保护位进行保护。

EDAC不支持保护16bit位宽的数据，因此当PM运行在16位模式的时候，必须关闭EDAC功能。当运行8bit模式时，片选只能使用RAMSN[0]和PROMSN[0]。

EDAC功能可以通过相应的寄存器关闭，当关闭后访问时序和标准的存储器时序完全一致。当EDAC功能使能以后，做写操作时，数据首先进入EDAC编码模块，生产相应的校验位，然后将校验位和数据位一起写入存储器中。做读操作时候，存储器中的校验位和数据一起读出，然后进入EDAC模块做校验检查，发现一位错误时将其纠正，发现两位错误时做相应的标记处理。

17.4.2 EDAC 校验的测试方法

计算机模块中提供了存储器读写操作时，EDAC功能的测试方法。

写操作诊断使能后，当做一次写操作时，EDAC模块计算出来的校验位将不随数据一起存入存储器，存入存储器的校验数据将是存储在配置寄存器MCFG3中TCB域的数据。

当读操作诊断使能后，从存储器中读回来的校验位同样也会存储到配置寄存器MCFG3的TCB域中，供用户查看，测试。

如何检查“纠一检二”的功能是否正常？首先，打开EDAC功能，同时关闭写诊断功能，做写入操作，将数据正确写入存储器。然后打开读诊断功能，对刚才写入的存储空间做读取操作，这时会将正确的校验位存储在MCFG3存储器的TCB域中，然后打开写诊断功能，将原数据做一位错误处理，然后对同一存储空间再做写操作。然后关闭EDAC功能，读回刚才写入的数据，看是否是错误的数据，然后再打开EDAC功能，看是否能够将错误纠正，返回正确的数据。

17.4.3 EDAC 的配置

EDAC的功能可以通过MCFG3寄存器进行配置。如果WB（写旁路）使能，则存储在TCB域的测试保护位将会代替芯片内部生成的数据保护位写入存储器中，通过写入不同的测试保护位，就可以检查EDAC的纠错及检错功能。当RB（读旁路）使能时，发起一次读操作，这时存储在存储器中的保护位将会被同时保存入TCB域，通过读回TCB数据可以查看EDAC是否正常工作。

当EDAC使能的时候，在MCFG2中的RMW位必须使能。EDAC功能是否使能不影响存储的读写时序关系。

SDRAM中有关于EDAC测试用的诊断寄存器，详细信息可参考DDR2SPA模块章节。

17.5 PROM 控制器

计算机模块处理器提供两个 PROM 片选择信号 ROMSN[1：0],最大支持 512M*32BIT。Prom 的访问和 ram 的访问基本相似。

PROM的访问主要通过配置MCFG1寄存器选择各种状态，[3:0] 是设置PROM读等待周期，[7:4]是 PROM 写等待周期，[9:8] 是 PROM 的数据宽度设定，[11]是确定 PROM 的写使能。

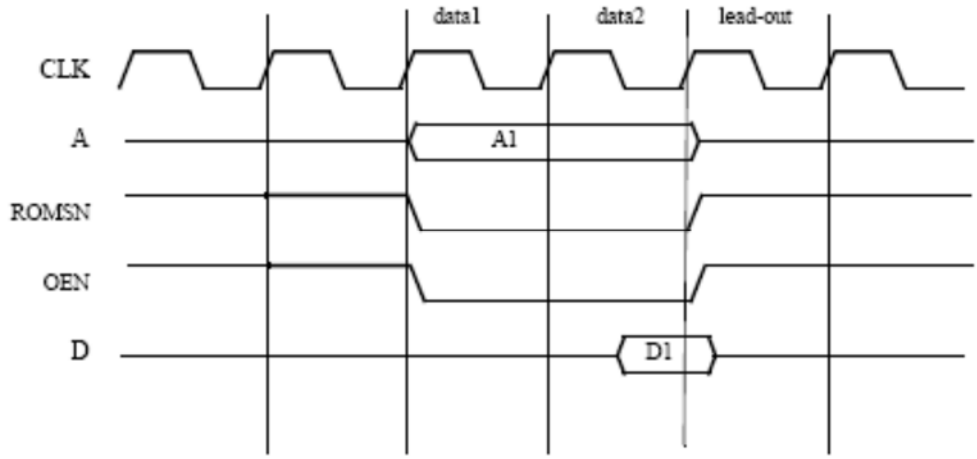


图 17-1 prom 读周期（0 等待）

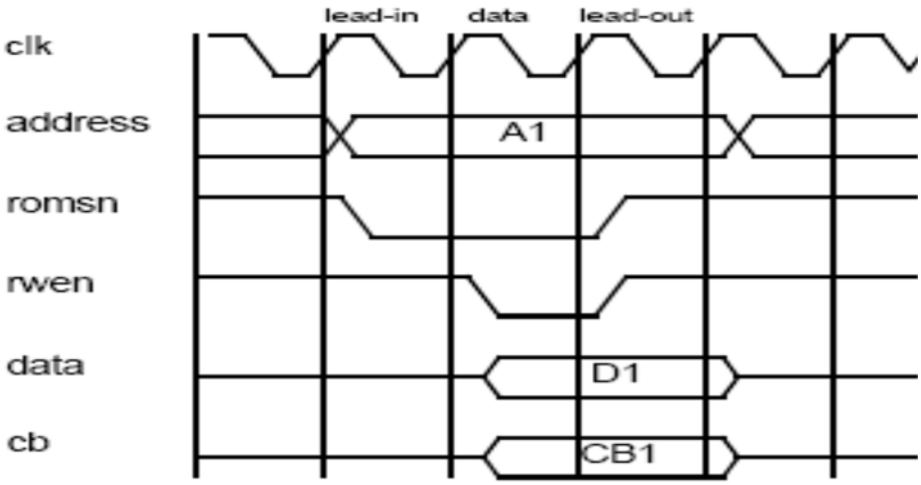


图 17-2 prom 写周期（0 等待）

17.6 SRAM 控制器

SRAM 区域最大可支持 512M 字节空间，具有 2 个 RAM 块，每块的大小在 MCFG2[12：9]配置，可以设置从 8 Kbyte 到 256 Mbyte。SRAM 读访问包括两个数据周期和 0-15 个等待周期。在非连贯的访问中，在一个读周期后增加一个 lead-out 的周期，可以阻止由于存储器或者 i/o 设备的 turn-off 时间引起的总线竞争。RAMSN[1:0]是外部 SRAM 片选信号，地址空间为 0x40000000 ~ 0x60000000；图 17-3、17-4 显示了基本的读/写周期波形（0 等待周期）。

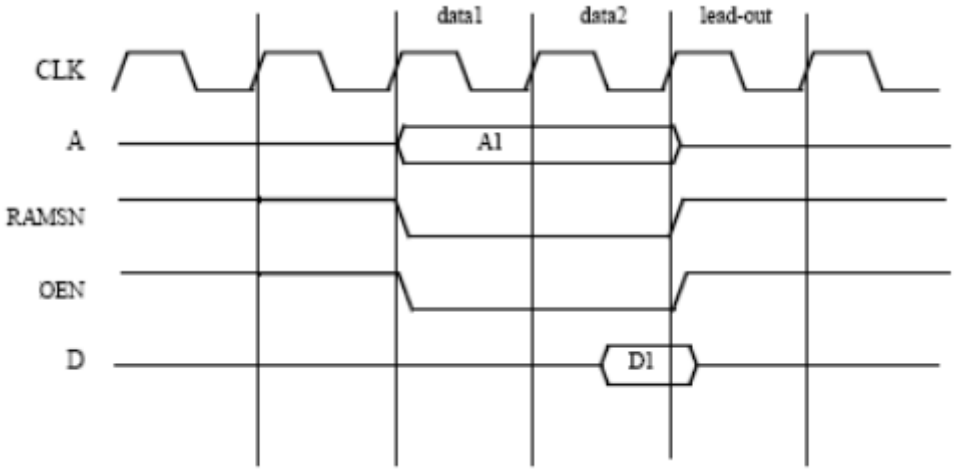


图 17-3 静态 ram 读周期（0 等待）

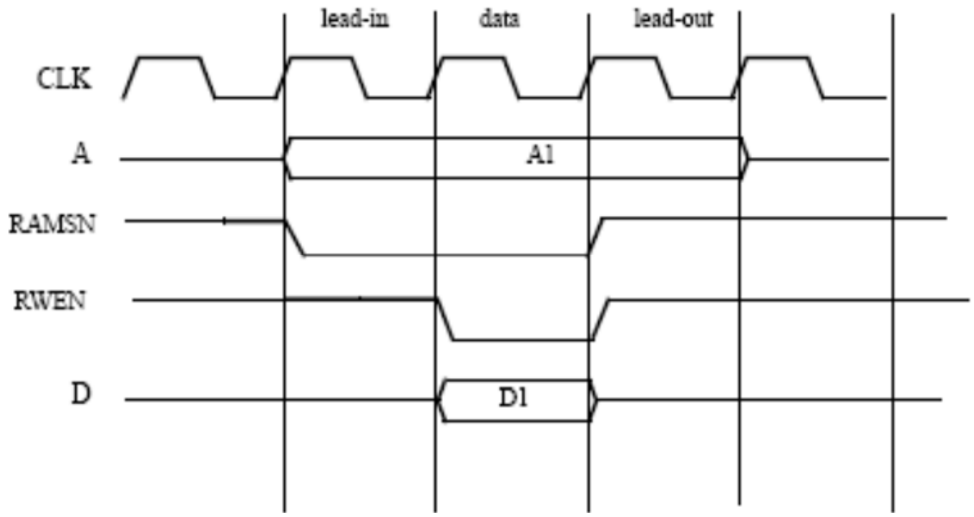


图 17-4 静态 ram 写周期（0 等待）

17.7 I/O 设备

I/O 读写等待周期也可在 MCFG1 中设定访问，可插入一个额外的等待周期(0-15 个周期)。处理器通过 MCFG1 决定 I/O 的数据总线宽度是 8-bit、16-bit、还是 32-bit。图 17-5、17-6 显示了基本的读/写周期波形（0 等待周期）。

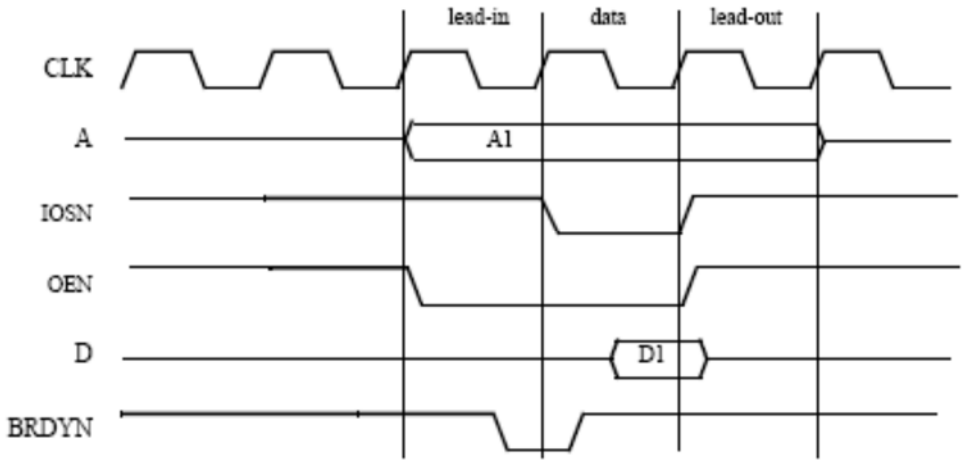


图 17-5 I/O 读周期（0 等待）

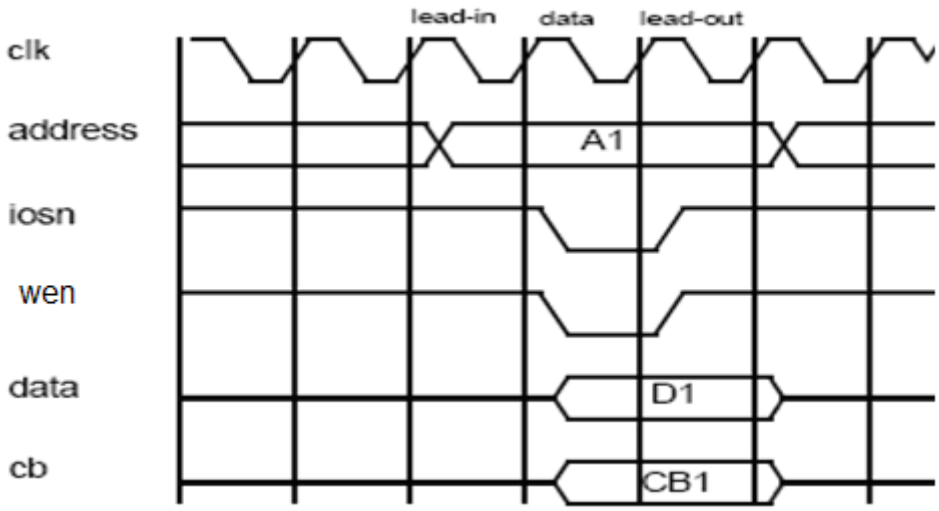


图 17-6 I/O 写周期（0 等待）

18. Spacewire 节点控制器

18.1 Spacewire 总线简介

SPW(Spacewire) 是 ESA(欧洲航天局)参考了 IEEE 1355-1995 商业接口标准和 ANSI/TIA/EIA-644标准，结合LVDS(低压差分技术)，并对IEEE-1355标准中不适合航天应用的连接器和电缆等部分进行了修改，提出的一种应用于航天领域的串行数据总线标准。SPW网络体系结构包括航天电子子系统、终端节点和SPW互联模块。

终端节点是SPW网络的重要组成部分，它嵌入在航天电子子系统中，将航天电子子系统与SPW网络连接起来，实现航天电子子系统与SPW网络收发信息的目的，保证了航天电子子系统与SPW网络上的其它子系统之间高速、实时、确定、可靠地进行数据交换。

SPW是一种高速、点对点、全双工、网络型串行总线，其通用性好，拓扑结构灵活，具有很好的抗辐射特性，特别在错误检测。异常处理及故障恢复等方面具有突出优势。SPW采用DS(Data-Srtobe)编码方式，信号采用LVDS方式传输，SPW具有良好的EMC特性，并且在错误检测、异常处理、故障保护和故障恢复等方面都有很强的支持。这些特性使得该技术对航天嵌入式应用有着非常重要的意义。

计算机模块内部集成4通道的SpaceWire总线节点控制器,其中SPW0,SPW1内置LVDS模块，SPW2,SPW3没有内置LVDS模块；

18.2 Spacewire 节点控制器主要特征

- 遵循欧空局 ECSS-E-ST-50-12C 和 ECSS-E-ST-50-52C 规范；
- 具备高速、串行、点对点、全双工通信等特点，其支持通过路由开关（Router）形成大型的通信网络；
- 可提供双向全双工、串行的高速 SpaceWire 链路网络接口，利用每个方向两对差分信号实现数据的编码及传输，最大速率可达 400Mbits/s；
- 支持配置远程存储访问（RMAP）功能；
- 编码采用 Data-Strobe 编码；
- 支持时间码；
- 支持DMA传输；
- 硬件逻辑实现物理层（PHY）、信号层、字符层、交换层以及数据包层协议；
- 支持点对点的链路以及路由交换网络，既可以通过 SpaceWire 电缆直接相连，也可通过 SPW 路由器与其它 SPW 节点或路由器相连；
- 支持电压为 350mV 的 LVDS 物理层，使 SPW 具有良好的 EMC 特性和更高的传输速率；

18.3 Spacewire 节点控制器实现的功能与工作流程

18.3.1 Spacewire 节点控制器实现的功能

- 配置管理：负责对各个节点地址进行配置，RMAP 功能配置，以及配置其他的寄存器
- 链路建立和维持：Spacewire 链路建立成功后，才能进行数据传输。无数据传输时需要发送 NULL 字符维持链路的连接性；
- 信息编码：对应用层信息首先完成并串转换，再进行 DS 编码后进行传输；
- 信息解码：对接收到的 DS 编码数据进行解码，完成串并转换；
- RMAP 包传输：支持 RMAP 数据包传输协议；
- 时间码接口：能够识别接收及发送系统时间码；

18.3.2 Spacewire 节点控制器工作流程

Spacewire发送数据是主动过程，而接收数据是一个被动的过程。计算机模块的Spacewire节点控制器要正常工作，需要进行一系列寄存器配置及发送或接收描述符的配置。Spacewire的标准协议ECSS-E-ST-50-12C规定了Spacewire具有两种频率，一种是链路建立时采用的频率(规定为10MHz)，另外一种是Spacewire收发数据时的工作频率(2-400MHz)。Spacewire节点之间要能够建立正常的通信渠道，所以在配置时，其频率必须保持一致。Spacewire的通信模式是点对点的方式，因而Spacewire的每个节点控制器都有一个自己的节点地址(node address)。如A节点往B节点发送数据，A发送数据的第一个字节就必须是B节点的节点地址，否者B节点接收到数据就将丢掉，认为是无效数据。Spacewire能够接收到的一个数据包的最大长度由接收描述符0中的Packet length配置决定(1Byte--(32M-1)Byte)。Spacewire一个数据包能发送的数据长度由发送描述符2中的Data length参数配置决定(1Byte—(16M-1)Byte)。

计算机模块的Spacewire节点控制器要进行基本的收发操作，则只需要对节点地址寄存器、时钟分频寄存器、DMA通道的接收最大长度寄存器、DMA发送描述符表地址寄存器、DMA通道接收描述符表地址寄存器、状态中断寄存器、控制寄存器进行配置即可。需要注意的是，使能Spacewire进行接收/发送数据的使能位在控制寄存器中进行配置，该位一旦使能即进行接收/发送，因而控制寄存器的配置放在最后进行配置。

计算机模块Spacewire提供了ECSS-E-ST-50-12C规定的时间码接口，可以实现将

Spacewire节点控制器的时间码值传输到Spacewire网络中去，也可以接收Spacewire网络传输来的时间码。通过操作时间寄存器、控制寄存器即可。

计算机模块中的Spacewire节点控制器支持ECSS-E-ST-50-52C的RMAP协议。RMAP协议的制定目的，在于支持SpaceWire网络中的节点通过链路对远程节点的存储器进行读写访问。这里的存储器是广义的，包括各种寄存器、存储器、缓存。RMAP可以用于配置网络、控制节点，节点间收发数据等。RMAP的基本机制是基于命令响应式的，命令类型总体分为三种：读、写、读-修改-写。进一步还可以细分为是否带应答、是否需校验、是否增址访问等。需要指出的是，RMAP定义的所有读写操作都是posted operations模式，即命令的发起者不需等待接收应答回复，不会被占用阻塞。这意味着任意时刻都可能有很多个读写命令包跑在链路上（outstanding），依靠事务标识符作区分，且没有针对响应丢失的超时机制。如果用户需要，可以在自己的应用层上另行实现超时机制。对于具体的RMAP命令格式可参考ECSS-E-ST-50-52C协议。RMAP的操作相比常规的Spacewire收发的操作，只增加了目标关键字寄存器的操作。但是进行RMAP数据通信时，需要符合ECSS-E-ST-50-52C协议的规定。

18.4 Spacewire 节点控制器结构

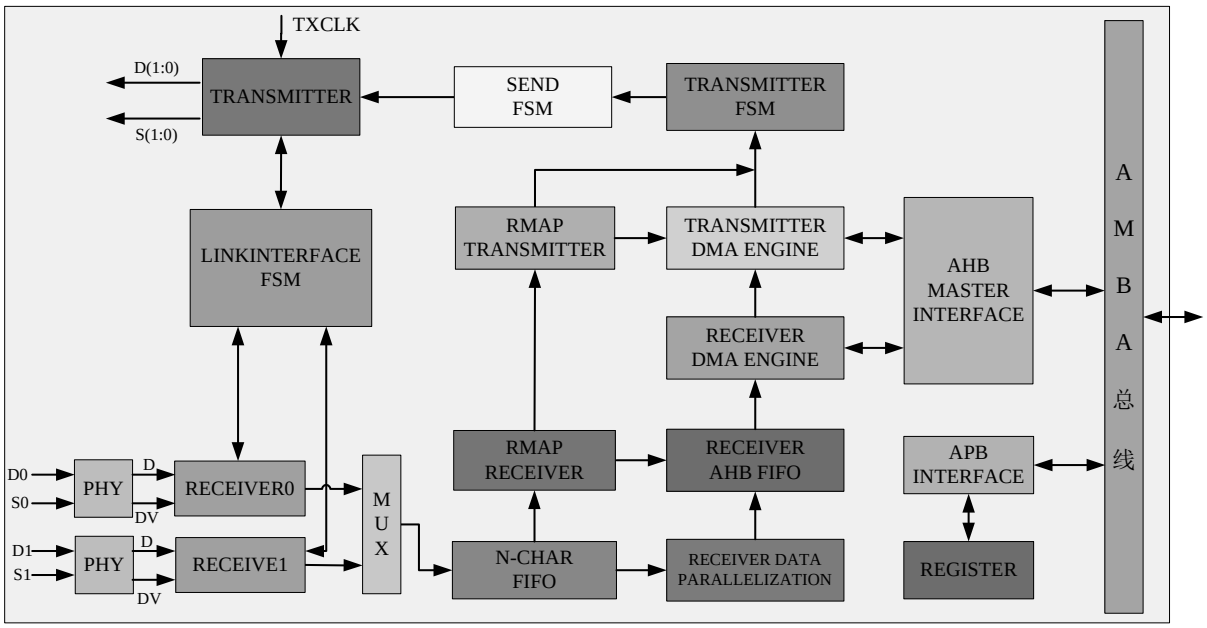


图 18-1 Spacewire 节点控制器功能结构框图

Spacewire 节点控制器结构框图如图图 所示。主要模块功能说明如下

表 18-1 所示：

表 18-1 Spacewire 节点控制器主要模块说明

序号	模块名称	模块描述
1	AMBA 总线	通过 AMBA 的 AHB 总线,实现 Spacewire 和 SRAM 的数据交换；通过 AMBA 的 APB 总线，实现 Spacewire 的寄存器配置。
2	LINKINTERFACE FSM	控制 spacewire 连接接口的状态机，处理交换层的协议，通过控制寄存器配置其状态。
3	TRANSMITTER	发送字符(包括 N-Chars 和 Time-codes)到 spacewire 网络链路上
4	RECEIVER	检测与其他节点的连接状况，接收经过 PHY 处理后的以位流方式传输的字符
5	PHY	接收其他节点传输来的数据，恢复出有效数据 D 和数据有效标志信号 DV
6	RECEIVER DMA ENGINE	处理从 spacewire 网络接收到的数据，将数据分配到相应的 DMA channel
7	RECEIVER AHB FIFO	接收端缓存数据 FIFO，FIFO 中的数据将通过 AHB 总线传输
8	TRANSMITTER DMA ENGINE	发送从 DMA channel 获得的数据到 spacewire 网络
9	RMAP RECEIVER	接收具有 RMAP 协议规范的数据
10	RMAP TRANSMITTER	发送遵循 RMAP 协议规范的数据
11	RECEIVER DATA PARALLELIZATION	把 N-Char FIFO 中的串行数据转换为并行格式

18.5 Spacewire 节点控制器寄存器描述

18.5.1 Spacewire 节点控制器寄存器地址

表 18-2 Spacewire 节点控制器寄存器地址

基地址	寄存器
0x80000a00	SPW AMBA I/F 0
0x80000b00	SPW AMBA I/F 1

0x80000c00	SPW AMBA I/F 2
0x80000d00	SPW AMBA I/F 3

偏移地址	读写	寄存器
0x0	R/W	控制寄存器(Control)
0x4	R	状态中断寄存器(Status/Interrupt-source)
0x8	R/W	节点地址寄存器(Node address)
0xC	R/W	时钟分频寄存器(Clock divisor)
0x10	R/W	目标关键字寄存器(Destination key)
0x14	R	时间寄存器(Time)
0x20	R/W	DMA 通道 1 控制/状态寄存器(DMA channel 1 control/status)
0x24	R/W	DMA 通道 1 接收最大长度寄存器(DMA channel 1 rx maximum length)
0x28	R/W	DMA 发送描述符表地址寄存器(DMA channel 1 transmit descriptor table address)
0x2C	R/W	DMA 通道 1 接收描述符表地址寄存器(DMA channel 1 receive descriptor table address)
0x30	R/W	DMA 通道 1 地址寄存器(DMA channel 1 address register)
0x40 - 0x5C	R/W	DMA 通道 2 寄存器(DMA channel 2 registers)
0x60 - 0x7C	R/W	DMA 通道 3 寄存器(DMA channel 3 registers)
0x80 - 0x9C	R/W	DMA 通道 4 寄存器(DMA channel 4 registers)

18.5.2 Spacewire 节点控制器寄存器说明

18.5.2.1 SPW 节点控制寄存器

表 18-3 SPW 控制寄存器

位	位名称	位描述
[31]	RA	RMAP 可用配置位(RA) 如果硬件具有 RMAP 命令处理模式，将该位置‘1’，只读
[30]	RX	RX 非对齐访问(RX) 如果接收器中的非对齐写功能可用，将该位置‘1’，只读
[29]	RC	RMAP CRC 可用配置位(RC) - 如果芯片中使能了 RMAP CRC 校验功能，将该位置‘1’，只读
[28：27]	NCH	DMA 通道数量(NCH) NCH 值等于可用的 DMA 通道数减 1(通道数=NCH+1)
[26]	PO	端口数量(PO)。PO 值等于 spacewire 端口数减 1
[25:23]	RES	保留
[22]	LOOP	回环使能。该位打开后，可以使 SPW 回环输出有效
[21]	PS	端口选择(PS) 当 NP 为‘0’时，选择可工作的端口。PS 为‘0’时，通过 index0 将端口链接到 data 和 strobe 端。为‘1’时，通过 index1 将端口链接到 data 和 strobe 端。只有在 VHDL 代码的 generic 中将 port 设置为‘2’时 PS 才有效。复位值为‘0’
[20]	NP	端口强制位(NP) 使端口无效位。当使 PS 无效时，不能用于选择可工作的端口。但是，通过检测相应接收链路的活跃性能够实现自动选择。只有当 VHDL generic 中将 port 设置为‘2’时，NP 才能使用。复位值为‘0’
[19:18]	RES	保留

[17]	RD	RMAP buffer 使无效位(RD) 如果仅仅设置只使用 1 个 RMAP buffer,这样就能确保 RMAP 命令能够连续的执行。只有当 VHDL generic 中设置 rmap 为‘1’时，RD 可用。复位值为‘0’
[16]	RE	RMAP 使能位(RE) 使能 RMAP 功能。只有当 VHDL generic 中设置 rmap 为‘1’时，RD 可用。复位值为‘1’
[15:12]	RES	保留
[11]	TR	R 接收时间使能位(TR) 开启后能够接收时间码。复位值为‘0’
[10]	TT	发送时间使能位(TT) 开启后能都发送时间码。复位值为‘0’
[9]	LI	Link error 中断请求位(LI) 当链路发生错误后产生中断。没有复位值
[8]	TQ	Tick-out 中断请求位(TQ) 当收到有效时间码后产生中断。没有复位值
[7]	RES	保留
[6]	RS	复位(RS) 使整个 SPW 节点复位。复位值为‘0’
[5]	PM	混杂模式(PM) 使能混杂模式。复位值为‘0’
[4]	TI	L Tick In(TI) 将 TI 设为‘1’后，主机就能够产生一个 tick 信号，相应的定时器值就增加，当前的字符发送完毕后，新值就接着被发送。也可以通过声明 tick_in_signal 信号来产生 tick 信号。复位值为‘0’
[3]	IE	中断使能(IE) 设置有效后，第 8-10 位设为 1 后，相应的时间触发后就能产生中断信号。复位值为‘0’
[2]	AS	自动开始(AS) 当收到一个 NULL 字符后，自动开始链路连接。没有复位值
[1]	LS	连接开始(LS) 开始连接，也就是允许从准备状态到开始状态的转换。如果不支持 RMAP 功能，则复位值为‘0’。支持则将复位值复位为输入的 rmapen 信号值
[0]	LD	链路无效位(LD) 使 SPW codec 无效。复位值为‘0’

18.5.2.2 SPW 节点控制器状态寄存器

表 18-4 SPW 状态寄存器

位	位名称	位描述
[31:24]	RES	保留
[23:21]	LS	链路状态(LS) 启动序列的目前状态。‘0’为错误复位状态，‘1’为错误等待状态，‘2’为准备状态，‘3’为开始状态，‘4’为连接状态，‘5’为运行状态
[20:10]	RES	保留
[9]	AP	有效端口(AP) 显示目前有效的端口。‘0’表示端口 0，‘1‘表示端口 1.端口序号与指示端口的 data 和 strobe 信号相关。只有在 generic 中 port 设置为‘2’才能使用
[8]	EE	Early EOP/EEP(EE) 当接收非 rmap 格式数据包的第一个字节的数据后收到 EOP 字符或是接收 rmap 格式数据包的第二个字节的数据后收到 EOP 字符后,将会把 EE 值置为‘1’。当写为‘1’后，这些数据包将被清除
[7]	IA	无效地址(IA) 当接收到的数据包中的地址使无效的目的地址(也就是说该地址与节点地址寄存器值不匹配)时，该位置‘1’。复位值为‘0’
[6:5]	RES	保留
[4]	PE	奇偶校验错误(PE) 发生奇偶校验错位，该位置‘1’，并将删除该数据。复位值为‘0’
[3]	DE	链接断开错误(DE) 当链路断开发生，该位设为‘1’，清除没有接收完整的数据。复位值为‘0’
[2]	ER	Escape Error(ER) 当 Escape 错误发生时，该位设为‘1’，清除没有接收完整的数据。复位值为‘0’
[1]	CE	Credit Error(ER) 发生 Credit 错误时，该位设为‘1’，清除没有接收完整的数据。复位值为‘0’

[0]	TO	Tick Out(TO) 收到新的时间计数值后，将其存入到时间计数寄存器中。当该位为‘1’时，清除计数器中的值。

18.5.2.3 SPW 节点控制器默认地址寄存器

表 18-5 SPW 节点地址寄存器

位	位名称	位描述
[31:16]	RES	保留
[15:8]	DEFMASK	Default mask(DEFMASK) Default mask 用于 spacewire 网路中节点的身份标识。这个存储区域用于在比较前标识节点地址。在检查地址前将具有 DEFMASK 反码的 DEFADDR 与收到的地址相与
[7:0]	DEFADDR	Default address(DEFADDR) Default address 用于 spacewire 网路中节点的身份标识。复位值为‘254’

18.5.2.4 SPW 节点控制器时钟分频器寄存器

表 18-6 SPW 时钟分频器寄存器

位	位名称	位描述
[31:16]	RES	保留
[15:8]	CLKDIVSTART	时钟分频启动(CLKDIVSTART) 用于链路处于启动状态时，时钟分频器的分频值。实际的分频值时钟分频器寄存值加 1
[7:0]	CLKDIVRUN	时钟分频运行(CLKDIVRUN) 当链路处于启动状态时，该值作为时钟分频器的分频值。实际的分频值时时钟分频器寄存值加 1

18.5.2.5 SPW 节点控制器目标关键字寄存器

表 18-7 SPW 目标关键字寄存器

位	位名称	位描述
[31:8]	RES	保留
[7:0]	DESTKEY	时 Destination key(DESTKEY) RMAP 目标器关键字。只有当 VHDL generic 中 rmap 设置为‘1’时可用

18.5.2.6 SPW 节点控制器时间寄存器

表 18-8 SPW 时间寄存器

位	位名称	位描述
[31:8]	RES	保留
[7:6]	TCTRL	时间控制标志(TCTRL) 时间控制标志的当前值，与从 tick-in 信号获得的时间码一起发送。收到的时间控制标志信号也存储在该寄存器中。复位值为‘0’
[5:0]	TIMECNT	时间计数器(TIMECNT) 系统时间计数器的当前值。每收到一个 tick-in 信号，计数

		器值增加，并将该值发送出去。可以直接对该寄存器进行写数据操作，但是这样的数据值就无法发送出去。收到的时间码值存储于该寄存器中。复位值为‘0’

18.5.2.7 SPW 节点控制器 DMA 控制寄存器

表 18-9 SPW DMA 控制寄存器

位	位名称	位描述
[31:17]	RES	保留
[16]	LE	链路连接错误无效(LE) 当链路连接错误发生时，使发送器无效。直到发送器被重新使能有效之前都不会再发送新的数据包。复位值为‘0’
[15]	SP	去除协议标识符(SP) 去掉每个数据包的协议标识符字节(第 2 字节)。当 SP 位独立于 SA 位进行设置时，地址字节(第 1 字节)同样也会被去掉。复位值为‘0’
[14]	SA	去除地址(SA) 去除每个数据包的地址字节(第 1 字节)。
[13]	EN	使能地址(EN) 对通道的节点地址进行单独的使能。
[12]	NS	没有溢出(NS) 当该位为‘0’，通道传输来了数据但却没有起作用的描述符时，该数据包被丢弃，SPW 继续接收新的数据。当设置为‘1’，SPW 将会等待描述符有效后再接收新数据
[11]	RD	Rx 描述符可用(RD) 设置为‘1’时，通知，SPW，在描述符表中存在有效的描述符。当遇到无效的描述符时,，SPW 将其置‘0’。复位值为‘0’
[10]	RX	RX 有效(RX) 如果 DMA 通道正在接收数据，该位置‘1’，否则置‘0’
[9]	AT	中断 TX(AT) 设置为‘1’时，正在发送的数据包将会中止，发送功能将无效，若没有在发送数据，唯一的作用将是使发送功能无效。自我清零。复位值为‘0’
[8]	RA	RX AHB 错误(RA) 当接收 DMA 通道访问 AHB 总线时，检测到错误响应信号。该位置‘1’时，将消除错误响应信号。复位值为‘0’
[7]	TA	TX AHB 错误(TA) 当发送 DMA 通道访问 AHB 总线时，检测到错误响应信号。该位置‘1’时，将消除错误响应信号。复位值为‘0’
[6]	PR	接收到数据包标示(PR) 每次收到一个数据包时该位将被置位。不能被 SPW 节点清除。通过写 1 进行清除。复位值为‘0’
[5]	PS	信息包发送(PS) 每次发送一个数据包时该位将被置位。不能被 spacewire 节点清除。通过写 1 进行清除。复位值为‘0’
[4]	AI	AHB 错误中断(AI) 置位后，当 DMA 通道访问 AHB 总线时，发生 AHB 总线错误时将产生中断信号。没有复位值
[3]	RI	接收中断(RI) 置位后，当每次收到一个数据包后将产生中断信号。当数据包以 EEP 或是 EOP 结束时，均会产生中断。没有复位值
[2]	TI	发送中断(TI) 置位后，当每次发送一个数据包后将产生中断信号。无论发送是否成功，均会产生中断。没有复位值
[1]	RE	接收中断(RI) 置位后，当每次收到一个数据包后将产生中断信号。当数据包以 EEP 或是 EOP 结束时，均会产生中断。没有复位值
[0]	TE	发送器使能(TE) 描述符表格中每次有新描述符有效时该位置‘1’。置‘1’后将使 spacewire 节点读取新的描述符用于发送数据包到相对应的地址。当 spacewire 节点遇到描述符无效时，该位自动清零。复位值为‘0’

18.5.2.8 SPW 节点控制器接收最长数据包寄存器

表 18-10 SPW 接收最长数据包寄存器

位	位名称	位描述

[31:25]	RES	保留
[24:0]	RXMAXLEN	RX 最大长度(RXMAXLEN) 以字节为单位能接收的最大数据包。只有 24-2 位可写。1-0 始终为‘0’。没有复位值

18.5.2.9 SPW 节点控制器发送描述符表地址寄存器

表 18-11 SPW 发送器描述符表地址寄存器

位	位名称	位描述
[31:10]	DESCBASE ADDR	描述符表基地址(DESCBASEADDR)设置描述符表的基地址。没有复位值
[9:4]	DESCSEL	描述符选择器(DESCSEL) 描述符表的偏移值。显示 spacewire 正在使用的描述符。当新的描述符读操作时，选择器每次按 16 增加，再翻转到 0，一直循环。复位值为‘0’
[3:0]	RES	保留

18.5.2.10 SPW 节点控制器接收描述符表地址寄存器

表 18-12 SPW 接收描述符表地址寄存器

位	位名称	位描述
[31:10]	DESCBASE ADDR	描述符表基地址(DESCBASE ADDR)设置描述符表的基地址。没有复位值
[9:3]	DESCSEL	描述符选择器(DESCSEL) 描述符表的偏移值。显示 SPW 正在使用的描述符。当新的描述符读操作时，选择器每次按 8 增加，再翻转到 0，一直循环。复位值为‘0’
[2:0]	RES	保留

18.5.2.11 SPW 节点控制器 DMA 通道地址寄存器

表 18-13 SPW 通道地址寄存器

位	位名称	位描述
[31:16]	RES	保留
[15:8]	MASK	DMA 通道地址屏蔽位
[7:0]	ADDR	DMA 通道的地址

18.5.2.12 SPW 节点控制器接收描述符寄存器 0

表 18-14 SPW 接收描述符寄存器 0

位	位名称	位描述
[31]	TR	截断(TR)，数据长度与最大数据长度冲突，会把超出最大长度的数据截掉表的基地址。没有复位值
[30]	DC	数据 CRC(DC) ，检测接收到的数据有 CRC 错误，置 1，其他情况为 0
[29]	HC	包头 CRC(HC) ，检测接收到的包头有 CRC 错误，置 1，其他情况为 0
[28]	EP	EEP termination(EP) ，数据包以错误的包尾结束标志
[27]	IE	中断使能(IE) ，如果设置 IE 有效，在 DMA 通道控制寄存器中的接收中断使能位有效的情况下，接收到数据后会产生中断信号
[26]	WR	Wrap(WR)，如果该位置 1，SPW 通道每次使用的描述符都是描述符表寄存器基地址指定的那一个。否则，描述符指针将会以 8 字节递增到更高的存储空间。描述符表限制在 1kbytes 的范围内，当达到最大边界后，自动会返回到基地址
[25]	EN	使能(EN)，设置为 1，才能激活描述符。激活后描述符包含的控制值才是有效的，指定的地址空间才能用于存储接收数据
[24:0]	PACKET LENGTH	数据长度(PACKETLENGTH)，存储在接收缓存中的数据字节数。只有在 EN 被 SPW 置为 0 后，该值才是有效的

- 地址为：接收描述符表地址寄存器中（31-10 位指定的基地址）&（9-3 位的偏移地址）&000，3 个部分拼起来的 32 位地址。

18.5.2.13 SPW 节点控制器接收描述符寄存器 1

表 18-15 SPW 接收描述符寄存器 1

位	位名称	位描述
[31:0]	PACKET ADDERSS	接收数据包地址(PACKETADDRESS),指向 buffer 的地址,buffer 用于存储接收到数据包，如果 rxunaligned 和 rmap 都设置为 0，则只有[31:2]被使用

- 地址为：接收描述符寄存器 0 地址+4

18.5.2.14 SPW 节点控制器发送描述符寄存器 0

表 18-16 SPW 发送描述符寄存器 0

位	位名称	位描述
[31:18]	RES	保留
[17]	DC	添加数据 CRC(DC)，把数据指针指定的数据发送完后，根据 RMAP spec 计算 CRC。CRC 会覆盖从指针处发出的所有数据。如果数据长度设置为 0，则不进行 CRC 计算
[16]	HC	添加包头 CRC(HC)，把包头指针指定的数据发送完后，根据 RMAP spec 计算 CRC。CRC 会覆盖除 non-crc 域覆盖外的所有从包头指针处发送出去的数据。当包头数据长度设置为 0，不会发送 CRC 码
[15]	LE	链路错误(LE)，在发送数据包过程中发生链路错误
[14]	IE	中断使能(IE)，如果置 1，在 DMA 控制寄存器中的发送中断使能位置 1 时，发送完数据包后产生会中断信号
[13]	WR	Wrap(WR)，如果该位置 1，SPW 通道每次使用的描述符都是描述符表寄存器基地址指定的那一个。否则，描述符指针将会以 16 字节递增到更高的存储空间。描述符表限制在 1kbytes 的范围内，当达到最大边界后，自动会返回到基地址
[12]	EN	使能(EN)，使能发送描述符，所有的控制域(地址、长度、Wrap、crc)设置后，EN 位应该设置。当该位被设置后，描述符不能再被修改，否者可能引起发送失败，当发送

		完成后，SPW 会清除该位
[11:8]	NONCRCLEN	Non-CRC bytes(NONCRCLEN)，设置包头开始处有多少字节的数据不包括在 CRC 计算范围内。当使用路径地址的时候，必须采用该设置，因为在使用路径地址的时候，包头最开始的数个字节需要丢弃，因而也就不需要进行 CRC 编码
[7:0]	HEADER LENGTH	包头长度(HEADERLEN)，包头长度(单位字节)。如果设置为 0，将跳过包头

- 地址为：发送描述符表地址寄存器中（31-10 位指定的基地址）&（9-4 位的偏移地址）&0000，3 个部分拼起来的 32 位地址。

18.5.2.15 SPW 节点控制器发送描述符寄存器 1

表 18-17 SPW 发送描述符寄存器 1

位	位名称	位描述
[31:0]	HEADER ADDERSS	头地址(HEADERADDRESS)，存储包头的地址。地址不需要以字为单位

- 地址为：发送描述符寄存器 0 地址+4

18.5.2.16 SPW 节点控制器发送描述符寄存器 2

表 18-18 SPW 发送描述符寄存器 2

位	位名称	位描述
[31:24]	RES	保留
[23:0]	DATA LENGTH	数据长度(DATALEN)，数据长度(单位字节)。如果设置为 0，则不发送数据。如果包头和数据长度都设置为 0，则不发送数据包

- 地址为：发送描述符寄存器 0 地址+8

18.5.2.17 SPW 节点控制器发送描述符寄存器 3

表 18-19 SPW 发送描述符寄存器 3

位	位名称	位描述
[31:0]	DATA ADDERSS	数据地址(DATAADDRESS) ，发送数据所在的存储地址，地址不需要以字为单位

- 地址为：发送描述符寄存器 0 地址+12

19. SPI 总线主控制器

19.1 SPI 简介

该模块主要为AMBA APB总线和串行总线接口（SPI）总线之间提供通道，可以被动态配置为spi主控设备（master）或者配置为spi从设备（slave）。发送字长、位顺序、时钟gap插入等都可以配置。模块主要由发送/接收FIFO、主控制器、从设备控制器、时钟产生模块及同步逻辑组成。SPI控制器结构如下图 所示：

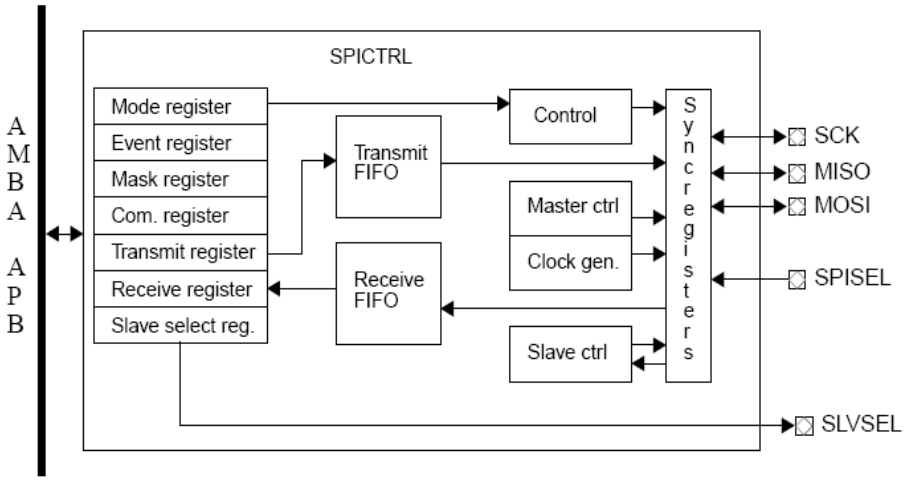


图 19-1 SPI 控制器结构框图

19.2 SPI 工作原理

19.2.1 SPI 传输协议

SPI总线是一种全双工同步串口总线。当主控机通过SLVSEL信号选中一个从设备后传输开始，时钟线SCK从理想状态开始传输，主控数据从主控机的MOSI信号线上传输出去，从机数据从从设备的MISO信号线上传输出来。当该模块被设置成主设备后，它将监测SPISEL信号线来侦测是否与其他主设备产生冲突，如果SPISEL被激活，该主控机将自动变为不使能。SPI传输模式通过时钟极性（CPOL）和时钟相位（CPHA）组合成四种不同的传输模式，分别如下图 所示：

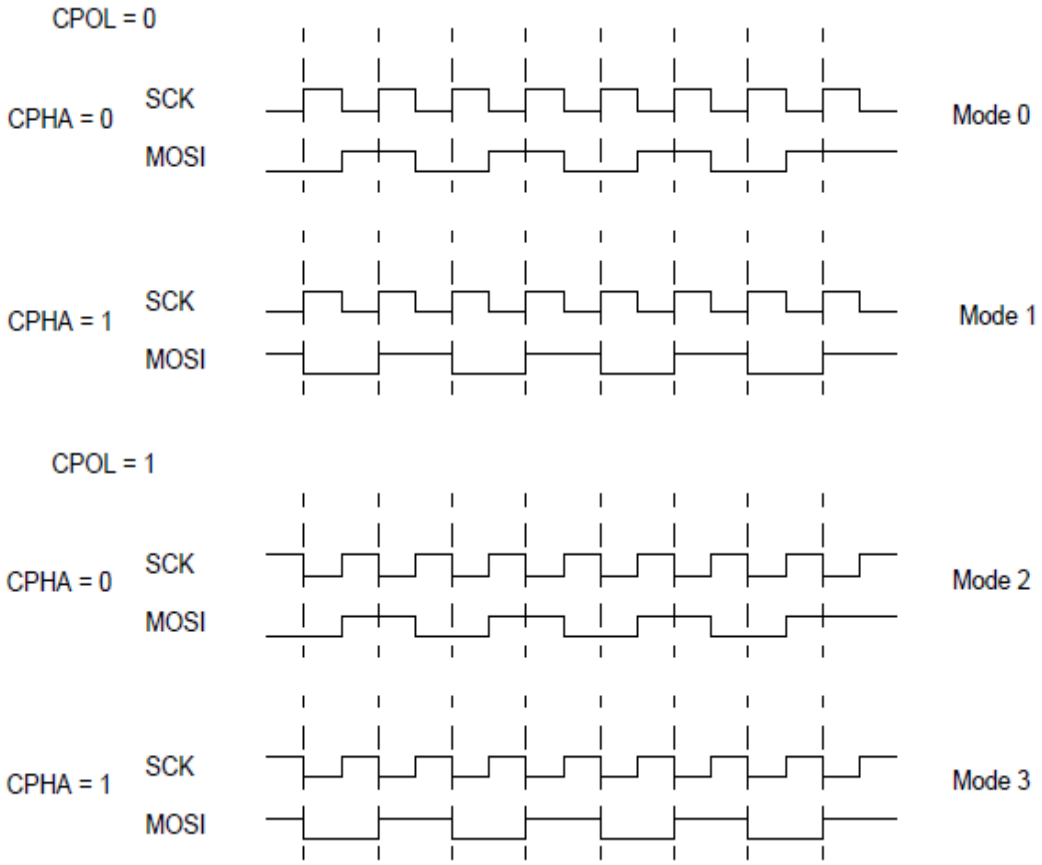


图 19-2 SPI 控制器传输字节（0x55）的所有传输模式

19.2.2 三线传输协议

当性能寄存器里的TWEN位设置为’1’时，该模块就被配置成三线传输模式，该模式用一条双向

数据信号线代替分离的输入输出数据线。在三线模式下，SPI总线是半双工串口总线。当主设备通过SLVSEL信号线选中从设备后传输开始，同时SCK信号也从理想状态开始发送波形。在三线模式下，只用MOSI信号来传输数据，而MISO信号不用。

在三线模式下，第一个传输数据的方向是由模式寄存器里的三线传输命令TTO位数值决定，当TTO设置为‘0’时，数据从主设备传到从设备上，当传输一个字的长度后，从设备开始用同样的数据线反过来传输一个字给主设备；当TTO设置为‘1’时，数据从从设备传输到主设备上，当传输一个字的长度后，主设备开始用同样的数据线反过来传输一个字给从设备。

19.2.3 SPI 时钟控制

该模块只有在主模式下才能产生时钟信号，该时钟频率是依赖于系统时钟频率和模式

寄存器下的27位（DIV16）、19~16位（PM）、13位（FACT）的设置。

当DIV16设置为0时，SCK频率计算公式如下：

$$SCK=\frac{SYSCLK}{(4-(2*FACT))*(PM+1)}-1 \qquad （式18－1）$$

当 DIV16 设置为 1 时，SCK 频率计算公式如下：

$$SCK=\frac{SYSCLK}{16*(4-(2*FACT))*(PM+1)} \qquad （式18－2）$$

注：当该模块使能后，DIV16、PM、FACT 所在的模式寄存器不可以修改配置。

19.2.4 SPI 从模式控制

当该模块被配置成从模式时，其就不能驱动任何SPI信号，直到被主机通过SPI_SEL

输入信号选中后。当该模块处于从模式时，如果SPI_SEL变为低时，该模块将把SPI_MISO配置成输出，并驱动设定好的第一位数值发送出去。

因该模块需要同步进来的时钟，只有两个时钟后，才能响应SCK信号上的传输，这将导致当数据线上改变后至少延迟三个时钟周期才能采样数据，所以SCK信号的最大时钟频率不能超过从设备系统时钟的1/8,主控设备必须允许从设备减少数据的建立时间。

19.2.5 SPI 主模式控制

通过配置模式寄存器使该模块为主模式，然后如果发送队列中有有效数据时，就会进行发送工作；当发送队列为空时，该模块就会把SPI_SCK信号置为理想状态。当处于主模式工作时，如果SPI_SEL输入为低信号时，该模块就会忽略正在传输工作并把事件寄存器里的MME状态位置1。当有MME错误发生时，该模块就会变为没使能，停止工作。

19.3 SPI 控制寄存器描述

19.3.1 SPI 寄存器地址

表 19-1 SPI 寄存器地址

地址	读写	寄存器
0x80000400	R	SPI 性能寄存器(SPI controller capability register)
0x80000420	R/W	SPI 模式寄存器(SPI controller mode register)
0x80000424	R	SPI 事件寄存器(SPI controller event register)
0x80000428	R/W	SPI 屏蔽寄存器(SPI controller mask register)
0x8000042C	R/W	SPI 命令寄存器(SPI controller command register)
0x80000430	W	SPI 发送数据寄存器 1(SPI controller transmit register)
0x80000434	R	SPI 接收数据寄存器 2(SPI controller receive register)
0x80000438	R/W	SPI slave 选择寄存器(SPI controller slave slect register)

注：在非自动传输模式下 SPI 寄存器的配置 0x80000400 - 0x80000500

19.3.2 SPI 性能寄存器

表 19-2 SPI 性能寄存器

位	位名称	位描述
[31:24]	SSSZ	Slave 选择信号有效数目，只有在 SSEN 是‘1’的时候有效当前值：1
[23:20]	MAXWLEN	SPI 控制模块支持的最大数据长度。 0x0：4~16bits，32bits 0x3~0xF：（MAXWLEN+1）bits 当前值：0
[19]	TWEN	SPI 三线模式支持使能 0：不支持；1：支持；当前值：‘0’
[18]	AMODE	是否支持自动传输模式 0：不支持；1：支持；当前值：0
[17]	ASELA	是否支持自动设置 slave 选择信号 0：支持；1：不支持；当前值：0
[16]	SSEN	是否支持 slave 选择寄存器和 slave 选择信号的映射 0：支持；1：不支持；当前值：1
[15:8]	FDEPTH	内部 FIFO 深度定义 当前值：16
[7]	SR	是否使用 SYNCRAM 实现 buffer 0：不是；1：是；当前值：1
[6:5]	FT	错误容忍支持 00：不容忍失败 01：DMR(每 32bits 可纠正 4bit 错误) 10：TMR（可纠正任意 bits 错误） 当前值：0

[4:0]	REV	版本

19.3.3 SPI 模式控制寄存器

表 19-3 SPI 模式控制寄存器

位	位名称	位描述
[31]	AMEN	0：自动传输模式使能关闭 1：自动传输模式使能
[30]	LOOP	0：spi 环回模式关闭 1：spi 环回模式打开
[29]	CPOL	时钟极性定义 0：初始为‘0’；1：初始为‘1’
[28]	CPHA	SPI 时钟相位定义 0：在第一次 SCK 变化的时候读回数据 1：在第二次 SCK 变化的时候读回数据
[27]	DIV16	系统时钟除以 16，仅在 master 模式下有效 0：系统时钟不进行 16 分频 1：系统时钟进行 16 分频
[26]	REV	数据其实传输 bit 定义 0：数据首先传输 LSB 1：数据首先传输 MSB
[25]	MS	主/从模式选择 0：slave；1：master
[24]	EN	SPI 控制模块使能位 1：SPI 控制模块使能 0：SPI 控制模块没有使能 注：软件可清零，多主机错误发生时 core 会自动将其清零
[23:20]	LEN	传输数据长度定义 0x0：32bits 数据 0x1~0x2：非法配置 0x3~0x15：LEN+1 注：不能超过 SPI 模块处理能力寄存器中 MAXLEN 的值，当前不能超过 31，即传输最长为 32bit
[19:16]	PM	在 master 模式下：对系统时钟分频产生 SCK 时钟时使用的参数。 在 slave 模式下：定义 spi 模块检测到 SCK 信号状态前，输入 SCK 信号需要稳定的周期数
[15]	TWEN	三线模式使能 0：不是能三线模式 1：使能三线模式
[14]	ASEL	slave 模式自动选择使能位 0：关闭自动选择 slave 模式 1：使能自动选择 slave 模式
[13]	FACT	系统时钟分频系数选择 0：4； 1：2

[12]	OD	引脚开驱动模式选择 0：普通模式； 1：开漏模式 注：对 slave 模式下输入信号没有影响
[11:7]	CG	连续传输时，用来设置插入 sck 时钟的数量，该值只有的 master 模式且发送队列保持非空的情况下有效。0x0 的配置值可以实现背靠背的传输
[6:5]	ASELDEL	Slave 自动选择延迟设置
[4]	TAC	在时钟 gap 期间是否使能 slave 自动选择功能 0：不是使能； 1：使能
[3]	TTO	三线模式下，传输顺序定义。 0：开始传输时，首先从 master 到 slave 1：开始传输时，首先从 slave 到 master
[2]	IGSEL	Spisel 信号是否忽略 0：不忽略；1：忽略
[1]	CITE	传输结束，是否需要空闲时间 0：不需要空闲时间，只要传输数据的最后一 bit 传输完成，则当然传输结束 1：需要空闲时间，只有数据传输完成，并且再等待空闲时钟达到空闲设置的水平，当然传输才结束 注：空闲值由 CG 域决定
[0]	RES	保留，数值为 0，向前兼容的时候需要写‘0’

19.3.4 SPI 事件寄存器

表 19-4 SPI 事件寄存器

位	位名称	位描述
[31]	TIP	SPI 传输状态寄存器 0：传输结束，困闲 1：传输正在进行
[30:15]	RES	保留
[14]	LT	传输结束并且发送队列空状态 0：软件写‘1’清零，写‘0’无效 1：上述状态发生
[13]	RES	保留
[12]	OV	0：软件写‘1’清零，写‘0’无效 1：接收缓存溢出
[11]	UN	0：软件写‘1’清零，写‘0’无效 1：发送开始时，发送队列空
[10]	MME	0：软件写‘1’清零，写‘0’无效 1：多主机错误发生
[9]	NE	0：SPI 接收缓存空 1：SPI 接收缓存非空
[8]	NF	0：SPI 发送缓存满 1：SPI 发送缓存有空间
[7:0]	RES	保留

19.3.5 SPI 控制屏蔽寄存器

表 19-5 SPI 控制屏蔽寄存器

位	位名称	位描述
[31]	TIPE	传输开始中断使能寄存器，当 SPI controller Event register 中 TIP 位的上升沿出现时产生一个传输中断。 0：屏蔽；1：不屏蔽
[30:15]	RES	保留
[14]	LTE	SPI controller Event register 中 LT 位触发中断功能的屏蔽位。 0：屏蔽；1：不屏蔽
[13]	RES	保留
[12]	OVE	SPI controller Event register 中 OV 位触发中断功能的屏蔽位。 0：屏蔽；1：不屏蔽
[11]	UNE	SPI controller Event register 中 UN 位触发中断功能的屏蔽位。 0：屏蔽；1：不屏蔽
[10]	MMEE	SPI controller Event register 中 MME 位触发中断功能的屏蔽位。 0：屏蔽；1：不屏蔽
[9]	NEE	SPI controller Event register 中 NE 位触发中断功能的屏蔽位。 0：屏蔽；1：不屏蔽
[8]	NFE	SPI controller Event register 中 NF 位触发中断功能的屏蔽位。 0：屏蔽；1：不屏蔽
[7:0]	RES	保留

19.3.6 SPI 控制命令寄存器

表 19-6 SPI 控制命令寄存器

位	位名称	位描述
[31:23]	RES	保留
[22]	LST	最后一位传输状态控制命令位 0：事件寄存器 LT 置‘1’后，自动清零 1：要求传输完最后一 bit 后产生 LT 事件状态 注：软件读回始终为‘0’
[21:0]	RES	保留

19.3.7 SPI 控制传输寄存器

表 19-7 SPI 控制传输寄存器

位	位名称	位描述
[31:0]	TDATA	传输数据寄存器，只有在 SPI controller Event register 中 NF 置位时，才能写入

		数据，数据存储的方式由 SPI controller Mode register 中的 REV 位控制

19.3.8 SPI 控制接收寄存器

表 19-8 SPI 控制接收寄存器

位	位名称	位描述
[31:0]	RDATA	NE 置位的时候，有效数据接收寄存器，接收长度和数据存储方式由 SPI controller Mode register 中的 REV 和 LEN 控制。 注：当 LEN=0X0 时，数据最高位存储在 bit[31]，最低位存储在 bit[0]

19.3.9 SPI slave 选择寄存器

表 19-9 SPI slave 选择寄存器

位	位名称	位描述
[31:SSSZ]	RES	保留，SSSZ 为 SPI 控制器处理能力配置数据。SSEN 为‘0’的时候，SSSZ 为 0 注：当然 SSSZ 为 1
[SSSZ-1:0]	ASLVSEL	SSEN 为‘1’的情况下，SPI 模块 slave 选择信号映射寄存器。 注：当然 SSSZ 为 1

20. CAN 总线控制器

20.1 简介

计算机模块-CAN 实现了 CAN 2.0B 协议，支持 BasicCAN 和 PeliCAN 模式，这两种模式可以通过时钟分频寄存器选择。在 BasicCAN 和 PeliCAN 两种模式下寄存器的映射有所不同。

计算机模块-CAN 控制器共有 32 个寄存器，地址分别为 0-31。在 CPU 看来，CAN 控制器相当于存储器地址映射的 I/O 设备，CPU 对 CAN 控制器的所有操作都是通过访问寄存器实现的。

20.2 CAN 控制器主要特征

- PCA82C200 模式（即默认的 BasicCAN 模式）
- 扩展的接收缓冲器（64 字节先进先出 FIFO）

- 和 CAN2.0B 协议兼容
- 同时支持 11 位和 29 位标识符
- 位速率可达 1Mbits/s
- PeliCAN 模式扩展功能
- 可读/写访问的错误计数器
- 可编程的错误报警限制
- 最近一次错误代码寄存器
- 对每一个 CAN 总线错误的中断
- 具体控制位控制的仲裁丢失中断
- 单次发送（无重发）
- 只听模式（无确认、无活动的出错标志）
- 支持热插拔（软件位速率检测）
- 验收过滤器扩展（4 字节代码，4 字节屏蔽）
- 自身信息接收（自接收请求）

20.3 结构框图

计算机模块-CAN 控制器主要由寄存器、位定时逻辑和位流处理器 3 个模块组成，如图图所示。

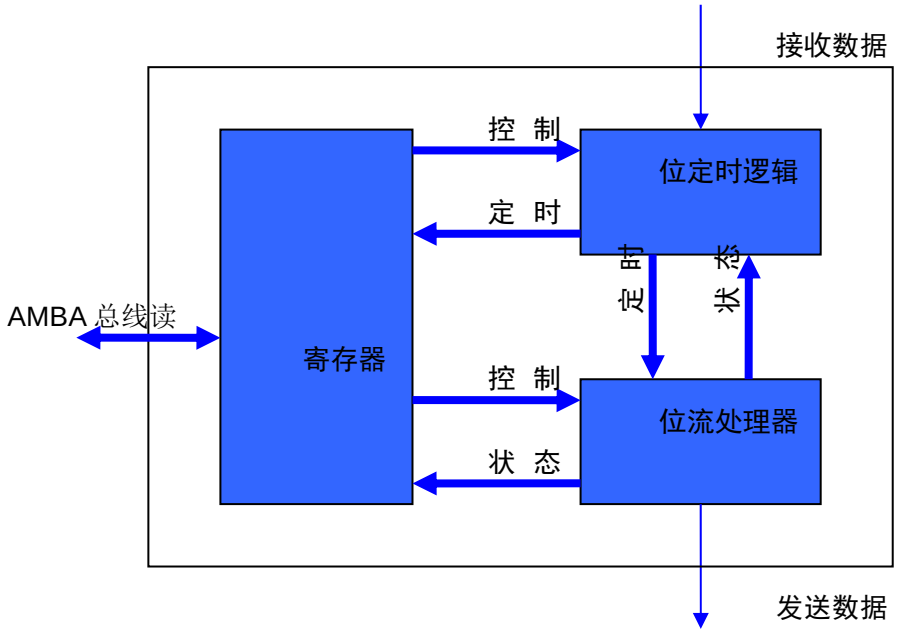


图 20-1 CAN 控制器结构框图

CAN 寄存器是与 CPU 连接的模块，CPU 对 CAN 控制器的所有操作都是通过寄存器进行的。

位定时逻辑实现的主要功能是检测 CAN 总线输入信号和来自位流处理器的 CAN 总线输出信号，输出采样点、采样位、发送点和同步信号等位定时信息。

位流处理器实现的主要功能是输入来自寄存器的寄存器信息和来自位定时逻辑的采样点、采样位、发送点和同步信号等位定时信息，对采样位的位流进行处理，并输出状态信息和 CAN 总线输出信号。

20.4 BasicCAN 模式寄存器

20.4.1 BasicCAN 模式寄存器映射

表 20-1 BasicCAN 偏移地址分配（基地址为：0x80200000）

地址	工作模式		复位模式	
	读	写	读	写
0	控制寄存器	控制寄存器	控制寄存器	控制寄存器
1	（0xFF）	命令寄存器	（0xFF）	命令寄存器
2	状态寄存器	—	状态寄存器	—
3	中断寄存器	—	中断寄存器	—
4	（0xFF）	—	验收代码寄存器	验收代码寄存器
5	（0xFF）	—	验收屏蔽寄存器	验收屏蔽寄存器

地址	工作模式		复位模式	
	读	写	读	写
6	（0xFF）	—	总线定时 0 寄存器	总线定时 0 寄存器
7	（0xFF）	—	总线定时 1 寄存器	总线定时 1 寄存器
8	（0x00）	—	（0x00）	—
9	（0x00）	—	（0x00）	—
10	发送识别码（10-3）	发送识别码（10-3）	（0xFF）	—
11	发送识别码（2-0）、RTR、DLC	发送识别码(2-0)、RTR、DLC	（0xFF）	—
12	发送数据字节 1	发送数据字节 1	（0xFF）	—
13	发送数据字节 2	发送数据字节 2	（0xFF）	—
14	发送数据字节 3	发送数据字节 3	（0xFF）	—
15	发送数据字节 4	发送数据字节 4	（0xFF）	—
16	发送数据字节 5	发送数据字节 5	（0xFF）	—
17	发送数据字节 6	发送数据字节 6	（0xFF）	—
18	发送数据字节 7	发送数据字节 7	（0xFF）	—
19	发送数据字节 8	发送数据字节 8	（0xFF）	—
20	接收识别码（10-3）	—	接收识别码（10-3）	—
21	接收识别码（2-0）、RTR、DLC	—	接收识别码（2-0）、RTR、DLC	—
22	接收数据字节 1	—	接收数据字节 1	—
23	接收数据字节 2	—	接收数据字节 2	—
24	接收数据字节 3	—	接收数据字节 3	—
25	接收数据字节 4	—	接收数据字节 4	—
26	接收数据字节 5	—	接收数据字节 5	—
27	接收数据字节 6	—	接收数据字节 6	—
28	接收数据字节 7	—	接收数据字节 7	—
29	接收数据字节 8	—	接收数据字节 8	—
30	（0x00）	—	（0x00）	—
31	时间分频寄存器	时间分频寄存器	时间分频寄存器	时间分频寄存器

20.4.2 控制寄存器

控制寄存器包含中断使能位和复位请求位。

表 20-1 控制寄存器（CR）

位	位名称	位描述
[7:5]	RES	保留
[4]	溢出中断使能	1：使能；0：禁能。
[3]	错误中断使能	1：使能；0：禁能。

[2]	发送中断使能	1：使能；0：禁能。
[1]	接收中断使能	1：使能；0：禁能。
[0]	复位请求	1：停止当前传输并进入复位模式； 0：返回工作模式。

- x：复位不影响该寄存器或位

20.4.3 命令寄存器

往寄存器的相应位写 1 将引起被支持的动作。

表 20-2 命令寄存器（CMR）

位	位名称	位描述
[7:4]	RES	保留
[3]	清除数据溢出	清除数据溢出状态位。
[2]	释放接收缓冲器	释放当前接收缓冲器以便于新的接收。
[1]	停止发送	停止尚未开始的发送。
[0]	发送请求	开始发送缓冲器中报文的发送。

- 注 1：读命令寄存器的结果总为“1111 1111”。

20.4.4 状态寄存器

状态寄存器反映模块的当前状态并且是只读的。

表 20-3 状态寄存器（SR）

位	位名称	位描述
[7]	总线状态	模块总线关闭，且此时无总线活动时，为 1。
[6]	错误状态	至少有一个错误计数器达到或超过 CPU 报警限制。
[5]	发送状态	正在发送报文时，为 1。
[4]	接收状态	正在接收报文时，为 1。
[3]	发送完毕	最后一个报文发送成功时，为 1。
[2]	发送缓冲器状态	为 1 时，CPU 可以向发送缓冲器中写入数据。
[1]	数据溢出状态	FIFO 中无空间导致报文丢失时，为 1。
[0]	接收缓冲器状态	接收 FIFO 中有可用报文时，为 1

- x：复位不影响该寄存器或位。

20.4.5 中断寄存器

中断寄存器通知 CPU 是什么引起了中断。只有在控制寄存器里相应的中断允许位置 1 时中断位才置 1。

表 20-4 中断寄存器（IR）

位	位名称	位描述
[7:4]	RES	保留
[3]	数据溢出中断	若 SR.1 由 0 变为 1，置位。
[2]	错误中断	若错误状态或总线状态发生变化，置位。
[1]	发送中断	若发送缓冲器被释放，置位。
[0]	接收中断	FIFO 不空时，置位。

- x：复位不影响该寄存器或位

20.4.6 发送缓冲寄存器

发送缓冲存储来自 CPU 的将要通过本模块发送的数据。在 BasicCAN 模式下只有标准帧格式报文可以被发送和接收，扩展帧格式报文将被忽略。

表 20-5 发送缓冲器

地址	名称	位							
		7	6	5	4	3	2	1	0
10	识别码 1	ID.10	ID.9	ID.8	ID.7	ID.6	ID.5	ID.4	ID.3
11	识别码 1	ID.2	ID.1	ID.0	RTR	DLC.3	DLC.2	DLC.1	DLC.0
12	发送数据 1	发送字节 1							
13	发送数据 2	发送字节 2							
14	发送数据 3	发送字节 3							
15	发送数据 4	发送字节 4							
16	发送数据 5	发送字节 5							
17	发送数据 6	发送字节 6							
18	发送数据 7	发送字节 7							
19	发送数据 8	发送字节 8							

20.4.7 接收缓冲寄存器

位于地址 20 至 29 的接收缓冲是 64 字节接收 FIFO 的可见部分。它的结构与发送缓冲器相同。

20.4.8 接收过滤寄存器

应用接收过滤代码和接收过滤屏蔽寄存器，报文可以根据它们的标识符（ID）被过滤。11 位的标识符的高 8 位与接收过滤代码寄存器中相应的接收过滤屏蔽寄存器中设为 0 的位比较，如果匹配则储存进 FIFO。

该寄存器不受硬件复位和软件复位的影响。

20.5 PeliCAN 模式寄存器

20.5.1 PeliCAN 模式寄存器映射

表 20-6 PeliCAN 偏移地址分配

<table>
<tr><th rowspan="2">地址</th><th colspan="4">工作模式</th><th colspan="2">复位模式</th></tr>
<tr><th colspan="2">读</th><th colspan="2">写</th><th>读</th><th>写</th></tr>
<tr><td>0</td><td colspan="2">模式寄存器</td><td colspan="2">—</td><td>模式寄存器</td><td>模式寄存器</td></tr>
<tr><td>1</td><td colspan="2">（0x00）</td><td colspan="2">命令寄存器</td><td>（0x00）</td><td>命令寄存器</td></tr>
<tr><td>2</td><td colspan="2">状态寄存器</td><td colspan="2">—</td><td>状态寄存器</td><td>—</td></tr>
<tr><td>3</td><td colspan="2">中断寄存器</td><td colspan="2">—</td><td>中断寄存器</td><td>—</td></tr>
<tr><td>4</td><td colspan="2">中断使能寄存器</td><td colspan="2">中断使能寄存器</td><td>中断使能寄存器</td><td>中断使能寄存器</td></tr>
<tr><td>5</td><td colspan="2">（0x00）</td><td colspan="2">—</td><td>（0x00）</td><td>—</td></tr>
<tr><td>6</td><td colspan="2">总线定时 0 寄存器</td><td colspan="2">—</td><td>总线定时 0 寄存器</td><td>总线定时 0 寄存器</td></tr>
<tr><td>7</td><td colspan="2">总线定时 1 寄存器</td><td colspan="2">—</td><td>总线定时 1 寄存器</td><td>总线定时 1 寄存器</td></tr>
<tr><td>8</td><td colspan="2">（0x00）</td><td colspan="2">—</td><td>（0x00）</td><td>—</td></tr>
<tr><td>9</td><td colspan="2">（0x00）</td><td colspan="2">—</td><td>（0x00）</td><td>—</td></tr>
<tr><td>10</td><td colspan="2">（0x00）</td><td colspan="2">—</td><td>（0x00）</td><td>—</td></tr>
<tr><td>11</td><td colspan="2">仲裁丢失捕捉寄存器</td><td colspan="2">—</td><td>仲裁丢失捕捉寄存器</td><td>—</td></tr>
<tr><td>12</td><td colspan="2">错误代码捕捉寄存器</td><td colspan="2">—</td><td>错误代码捕捉寄存器</td><td>—</td></tr>
<tr><td>13</td><td colspan="2">错误报警限制寄存器</td><td colspan="2">—</td><td>错误报警限制寄存器</td><td>错误报警限制寄存器</td></tr>
<tr><td>14</td><td colspan="2">接收错误计数器</td><td colspan="2">—</td><td>接收错误计数器</td><td>接收错误计数器</td></tr>
<tr><td>15</td><td colspan="2">发送错误计数器</td><td colspan="2">—</td><td>发送错误计数器</td><td>发送错误计数器</td></tr>
<tr><td rowspan="2">16</td><td>接收 SFF</td><td>接收 EFF</td><td>发送 SFF</td><td>发送 EFF</td><td rowspan="2">验收代码 0 寄存器</td><td rowspan="2">验收代码 0 寄存器</td></tr>
<tr><td>帧信息</td><td>帧信息</td><td>帧信息</td><td>帧信息</td></tr>
</table>

17	接收 识别码 1	接收 识别码 1	发送 识别码 1	发送 识别码 1	验收代码 1 寄存器	验收代码 1 寄存器
18	接收 识别码 2	接收 识别码 2	发送 识别码 2	发送 识别码 2	验收代码 2 寄存器	验收代码 2 寄存器
19	接收数据 1	接收 识别码 3	发送数据 1	发送 识别码 3	验收代码 3 寄存器	验收代码 3 寄存器
20	接收数据 2	接收 识别码 4	发送数据 2	发送 识别码 4	验收屏蔽 0 寄存器	验收屏蔽 0 寄存器
21	接收数据 3	接收数据 1	发送数据 3	发送数据 1	验收屏蔽 1 寄存器	验收屏蔽 1 寄存器
22	接收数据 4	接收数据 2	发送数据 4	发送数据 2	验收屏蔽 2 寄存器	验收屏蔽 2 寄存器
23	接收数据 5	接收数据 3	发送数据 5	发送数据 3	验收屏蔽 3 寄存器	验收屏蔽 3 寄存器
24	接收数据 6	接收数据 4	发送数据 6	发送数据 4	保留（0x00）	—
25	接收数据 7	接收数据 5	发送数据 7	发送数据 5	保留（0x00）	—
26	接收数据 8	接收数据 6	发送数据 8	发送数据 6	保留（0x00）	—
27	FIFO	接收数据 7	—	发送数据 7	保留（0x00）	—
28	FIFO	接收数据 8	—	发送数据 8	保留（0x00）	—
29	接收报文计数器		—		接收报文计数器	—
30	（0x00）		—		（0x00）	—
31	时间分频寄存器		时间分频寄存器		时间分频寄存器	时间分频寄存器

20.5.2 模式寄存器

表 20-7 模式寄存器（MOD）

位	位名称	位描述
[7:4]	RES	保留
[3]	验收滤波模式	1：单滤波模式；0：双滤波模式。
[2]	自检测模式	1：控制器进入自检测模式。
[1]	仅听模式	1：控制器进入仅听模式。
[0]	复位模式	1：停止当前传输并进入复位模式。 0：返回工作模式。

- x：复位不影响该寄存器或位

20.5.3 命令寄存器

往寄存器的相应位写 1 将引起被支持的动作。

表 20-8 命令寄存器（CMR）

位	位名称	位描述
[7:5]	RES	保留
[4]	自接收请求	1：发送并同时接收一个报文。

[3]	清除数据溢出	1：清除数据溢出状态位。
[2]	释放接收缓冲器	1：释放当前接收缓冲器以便于新的接收。
[1]	停止发送	1：停止尚未开始的发送。
[0]	发送请求	1：开始发送缓冲器中报文的发送。

- x：复位不影响该寄存器或位。

20.5.4 状态寄存器

状态寄存器反映模块的当前状态并且是只读的。

表 20-9 状态寄存器

位	位名称	位描述
[7]	总线状态	1：模块总线关闭，且此时无总线活动。
[6]	错误状态	1：至少有一个错误计数器达到或超过报警限制。
[5]	发送状态	1：正在发送报文。
[4]	接收状态	1：正在接收报文。
[3]	发送完毕	1：最后一个报文发送成功。
[2]	发送缓冲器状态	1：CPU 可以向发送缓冲器中写入数据。
[1]	数据溢出状态	1：FIFO 中无空间导致报文丢失。
[0]	接收缓冲器状态	1：接收 FIFO 中有可用报文。

- x：复位不影响该寄存器或位。SR.4 与 SR.5 在总线关闭后会产生 11 个连续的高电平

20.5.5 中断寄存器

中断寄存器通知 CPU 是什么引起了中断。只有在中断允许寄存器里相应的中断允许位置 1 时中断位才置 1。

表 20-10 中断寄存器（IR）

位	位名称	位描述
[7]	总线错误中断	若检测到总线上有错误，置位。
[6]	仲裁丢失中断	若模块已经丢失仲裁，置位。
[5]	错误被动中断	若模块处于错误主动与错误被动之间。
[4]	保留	保留（总是 0）。

[3]	数据溢出中断	若 SR.1 由 0 变为 1，置位。
[2]	错误中断	若错误状态或总线状态发生变化，置位。
[1]	发送中断	若发送缓冲器被释放，置位。
[0]	接收中断	FIFO 不空时，置位。

- x：复位不影响该寄存器或位。

20.5.6 中断允许寄存器

在中断允许寄存器里可以允许/禁止独立的中断源。如果被允许，则中断寄存器里的相应位可以被置 1，同时将产生一个中断。

表 20-11 中断允许寄存器（IER）

位	位名称	位描述
[7]	总线错误中断使能	1：使能；0：禁能。
[6]	仲裁丢失中断使能	1：使能；0：禁能。
[5]	错误被动中断使能	1：使能；0：禁能。
[4]	保留	保留。
[3]	数据溢出中断使能	1：使能；0：禁能。
[2]	错误中断使能	1：使能；0：禁能。
[1]	总线错误中断使能	1：使能；0：禁能。
[0]	仲裁丢失中断使能	1：使能；0：禁能。

- x：复位不影响该寄存器或位。

20.5.7 仲裁丢失捕捉寄存器

表 20-12 仲裁丢失捕捉寄存器（ALC）

位	位名称	位描述
[7:5]	保留	保留
[4:0]	位编号	仲裁时丢失的位编号

- x：复位不影响该寄存器或位。

20.5.8 错误代码捕捉寄存器

表 20-13 仲裁丢失捕捉寄存器（ECC）

位	位名称	位描述
[7:6]	错误代码	错误代码编号。
[5]	方向	1：接收；0：发送。
[4:0]	段	帧中出错的部分。

- x：复位不影响该寄存器或位。

表 20-14 错误代码说明（ECC.7:6）

ECC.7:6	说明	ECC.7:6	说明
0	位错误	2	填充错误
1	格式错误	3	其它

表 20-15 错误代码说明（ECC.4:0）

ECC.4:0	说明	ECC.4:0	说明
0x03	帧起始	0x0A	数据段
0x02	ID.28 - ID.21	0x08	CRC 序列
0x06	ID.20 - ID.18	0x18	CRC 界定符
0x04	SRTR 位	0x19	应答通道
0x05	IDE 位	0x1B	应答界定符
0x07	ID.17 - ID.13	0x1A	帧结束
0x0F	ID.12 - ID.5	0x12	间断
0x0E	ID.4 - ID.0	0x11	主动错误标记
0x0C	RTR 位	0x16	被动错误标记
0x0D	保留位 1	0x13	支配位误差
0x09	保留位 0	0x17	错误界定符
0x0B	数据长度代码	0x1C	过载标记

20.5.9 错误报警限制寄存器

该寄存器允许设置 CPU 错误警告的限制。默认值是 96。注意该寄存器只在复位模式下可写。

20.5.10 接收错误计数器

该寄存器显示接收错误计数器的值。它在复位模式下可写。总线关闭事件会把它复位为 0。

20.5.11 发送错误计数器

该寄存器显示发送错误计数器的值。它在复位模式下可写。总线关闭事件会把它复位为0。

20.5.12 发送缓冲寄存器

发送缓冲被映射为地址 16 至 28 并且是只写的。发送缓冲的结构取决于将要发送的是标准帧（SFF）还是扩展帧（EFF），如下所示：

表 20-16 发送缓冲器

地址	写（SFF）	写（EFF）	地址	写（SFF）	写（EFF）
16	发送帧信息	发送帧信息	23	发送数据 5	发送数据 3
17	发送识别码 1	发送识别码 1	24	发送数据 6	发送数据 4
18	发送识别码 2	发送识别码 2	25	发送数据 7	发送数据 5
19	发送数据 1	发送识别码 3	26	发送数据 8	发送数据 6
20	发送数据 2	发送识别码 4	27	—	发送数据 7
21	发送数据 3	发送数据 1	28	—	发送数据 8
22	发送数据 4	发送数据 2			

表 20-17 发送帧信息（此位段在 SFF 和 EFF 帧中相同）

位	位名称	位描述
[7]	选择帧格式	1：使能；0：禁能。
[6]	远程发送请求帧	1：使能；0：禁能。
[5:4]	保留	保留。
[3:0]	数据长度	DLC 指定数据长度代码并且应该是 0 到 8 之间的数值。如果大于 8，则 8 个字节将被发送

表 20-18 发送标识符 1（此位段在 SFF 帧和 EFF 帧中相同）

位	位名称	位描述
[7:0]	标识符	标识符的最高 8 位，EFF 的 ID28-21，SFF 的 ID10-3

表 20-19 发送标识符 2，SFF 帧

位	位名称	位描述
[7:5]	标识符	SFF 标识符的低 3 位
[4:0]	保留	保留

表 20-20 发送标识符 2，EFF 帧

位	位名称	位描述
[7:0]	标识符	29 位 EFF 标识符的第 20 到第 13 位

表 20-21 发送标识符 3，EFF 帧

位	位名称	位描述
[7:0]	标识符	29 位 EFF 标识符的第 12 到第 5 位

表 20-22 发送标识符 4，EFF 帧

位	位名称	位描述
[7:3]	标识符	29 位 EFF 标识符的第 4 到第 0 位
[2:0]	保留	保留

数据位段：

对于 SFF 帧，数据位段位于地址 19 到 26，对于 EFF 帧，位于 21 到 28。数据从位于最低地址的 MSB（最高位字节）开始发送。

20.5.13 接收缓冲寄存器

表 20-23 接收缓冲寄存器

地址	读（SFF）	读（EFF）
16	接收帧信息	接收帧信息
17	接收识别码 1	接收识别码 1

地址	读（SFF）	读（EFF）
18	接收识别码 2	接收识别码 2
19	接收数据 1	接收识别码 3
20	接收数据 2	接收识别码 4
21	接收数据 3	接收数据 1
22	接收数据 4	接收数据 2
23	接收数据 5	接收数据 3
24	接收数据 6	接收数据 4
25	接收数据 7	接收数据 5
26	接收数据 8	接收数据 6
27	FIFO 中下一个报文的接收帧信息	接收数据 7
28	FIFO 中下一个报文的接收识别码 1	接收数据 8

表 20-24 接收帧信息（此位段在 SFF 和 EFF 帧中相同）

位	位名称	位描述
[7]	已接收报文帧格式	1 = EFF；0 = SFF。
[6]	RTR	RTR 帧时为 1
[5:4]	保留	00
[3:0]	数据长度	DLC 指定数据长度代码

表 20-25 接收标识符 1（此位段在 SFF 帧和 EFF 帧中相同）

位	位名称	位描述
[7:0]	标识符	标识符的高 8 位，EFF 的 ID28-21，SFF 的 ID10-3

表 20-26 接收标识符 2，SFF 帧

位	位名称	位描述
[7:5]	标识符	SFF 标识符的低 3 位
[4]	RTR	RTR 帧时为 1

[3:0]	保留	保留

表 20-27 接收标识符 2，EFF 帧

位	位名称	位描述
[7:0]	标识符	29 位 EFF 标识符的第 20 到第 13 位

表 20-28 接收标识符 3，EFF 帧

位	位名称	位描述
[7:0]	标识符	29 位 EFF 标识符的第 12 到第 5 位

表 20-29 接收标识符 4，EFF 帧

位	位名称	位描述
[7:3]	标识符	29 位 EFF 标识符的第 4 到第 0 位
[2]	RTR	RTR 帧时为 1
[1:0]	保留	保留

数据位段：

对于接收到的 SFF 帧，数据段位于地址 19 到 26，对于 EFF 帧则位于 21 到 28。

20.5.14 验收过滤寄存器

验收过滤器可以用来过滤掉不符合特定要求的报文。如果一个报文被过滤掉，它将不被放进接收 FIFO 里面，CPU 也不必处理它。

有两种不同的过滤模式：单过滤和双过滤。模式寄存器的第 3 位控制使用哪种模式。在单过滤模式下只使用一个 4 个字节的过滤器。在双过滤模式下使用两个更小的过滤器，如果匹配其中任何一个，则报文被接收。每个过滤器由两部分组成：接收代码和接收屏蔽。代码寄存器用来指定匹配的格式而屏蔽寄存器则指定不考虑的位。总共 8 个寄存器被用作接收过滤器，如下表所示。注意它们只在复位模式下被读写。

表 20-30 验收过滤寄存器

地址	说明	地址	说明
16	验收代码 0 寄存器（ACR0）	20	验收屏蔽 0 寄存器（ACM0）
17	验收代码 1 寄存器（ACR1）	21	验收屏蔽 1 寄存器（ACM1）
18	验收代码 2 寄存器（ACR2）	22	验收屏蔽 2 寄存器（ACM2）
19	验收代码 3 寄存器（ACR3）	23	验收屏蔽 3 寄存器（ACM3）

20.5.14.1 单过滤模式，标准帧

当在单过滤模式下接收一个标准帧，寄存器 ACR0:3 将会以以下的方式与接收到的报文比较：

ACR0.7:0 和 ACR1.7:5 与 ID.28:18 比较。

ACR1.4 与 RTR 位比较。

ACR1.3:0 未使用。

ACR2 和 ACR3 与数据字节 1 和 2 比较。

AMR 寄存器里相应的位选择是否比较的结果没关系。屏蔽寄存器里一个置 1 的位表示不考虑。

20.5.14.2 单过滤模式，扩展帧

当在单过滤模式下接收一个扩展帧，寄存器 ACR0:3 将会以以下的方式与接收到的报文比较：

ACR0.7:0 和 ACR1.7:0 与 ID.28:13 比较。

ACR2.7:0 和 ACR3.7:3 与 ID.12:0 比较。

ACR3.2 与 RTR 位比较。

ACR3.1:0 未使用。

AMR 寄存器里相应的位选择是否比较的结果没关系。屏蔽寄存器里一个置 1 的位表示不考虑。

20.5.14.3 双过滤模式，标准帧

当在双过滤模式下接收一个标准帧，寄存器 ACR0:3 将会以以下的方式与接收到的报文比较：

过滤器 1：

ACR0.7:0 和 ACR1.7:5 与 ID.28:18 比较。

ACR1.4 与 RTR 位比较。

ACR1.3:0 与数据字节 1 的高半字节比较。

ACR3.3:0 与数据字节 1 的低半字节比较。

过滤器 2：

ACR2.7:0 和 ACR3.7:5 与 ID.28:18 比较。

ACR3.4 与 RTR 位比较。

AMR 寄存器里相应的位选择是否比较的结果没关系。屏蔽寄存器里一个置 1 的位表示不考虑。

20.5.14.4 双过滤模式，扩展帧

当在双过滤模式下接收一个扩展帧，寄存器 ACR0:3 将会以以下的方式与接收到的报文比较：

过滤器 1：

ACR0.7:0 和 ACR1.7:0 与 ID.28:13 比较。

过滤器 2：

ACR2.7:0 和 ACR3.7:0 与 ID.28:13 比较。

AMR 寄存器里相应的位选择是否比较的结果没关系。屏蔽寄存器里一个置 1 的位表示不考虑。

20.5.15 接收报文计数器

位于地址 29 的接收报文计数器保持当前储存在接收 FIFO 里的报文的数量。最高 3 位总为 0。

20.6 公共寄存器

在 BasicCAN 和 PeliCAN 模式下有 3 个公共寄存器，它们具有相同的地址和相同的功能。它们是时钟分频寄存器和总线定时寄存器 0 和 1。

20.6.1 时钟分频寄存器

此寄存器的功能有二个：

一、是选择 PeliCAN 和 BasicCAN 模式；

二、是进行输出时钟分频设置；

表 20-31 时钟分频寄存器（CDR）

位	位名称	位描述
[7]	CAN 模式	1：PeliCAN；0：BasicCAN
[6:4]	保留	保留
[3]	输出时钟关闭	禁止输出时钟 clkout 输出
[2:0]	clkout 时钟分频设置	设定频率分频系数 BIT[2：0] = “000” => Fclkout = Fosc/2; BIT[2：0] = “001” => Fclkout = Fosc/4; BIT[2：0] = “010” => Fclkout = Fosc/6; BIT[2：0] = “011” => Fclkout = Fosc/8; BIT[2：0] = “100” => Fclkout = Fosc/10; BIT[2：0] = “101” => Fclkout = Fosc/12; BIT[2：0] = “110” => Fclkout = Fosc/14; BIT[2：0] = “111” => Fclkout = Fosc;

20.6.2 总线定时 0 寄存器

表 20-32 总线定时 0 寄存器（BTR0）

位	位名称	位描述
[7:6]	SJW	同步跳跃宽度
[5:0]	BRP	波特速率预设值

CAN Core 的系统时钟由以下计算：

$$t_{scl} = 2 \times t_{clk} \times (BRP + 1) \quad \text{（式 20-1）}$$

其中 tscl 是系统时钟。

同步跳跃宽度定义了在一次重同步中一个位周期里有多少个时钟周期(tscl)可以调整。

20.6.3 总线定时 1 寄存器

总线定时 1 寄存器（BTR1）的位分配（地址 7）。

表 20-33 总线定时 1 寄存器（BTR1）

位	位名称	位描述
[7]	SAM	1：对总线采样 3 次。 0：对总线采样 1 次。
[6:4]	TSEG2	时间段 2。
[3:0]	TSEG1	时间段 1。

CAN 总线的位周期由 CAN 的系统时钟和时间段 1 和 2 决定，如下面的等式所示：

$$t_{tseg1} = t_{scl} \times (TSEG1+1) \quad \text{（式 20-2）}$$

$$t_{tseg2} = t_{scl} \times (TSEG2+1) \quad \text{（式 20-3）}$$

$$t_{bit} = t_{tseg1} + t_{tseg2} + t_{scl} \quad \text{（式 20-4）}$$

附加的 tscl 项来自初始的同步段。采样在位周期的 TSEG1 和 TSEG2 之间完成。

20.7 信号数据帧组成

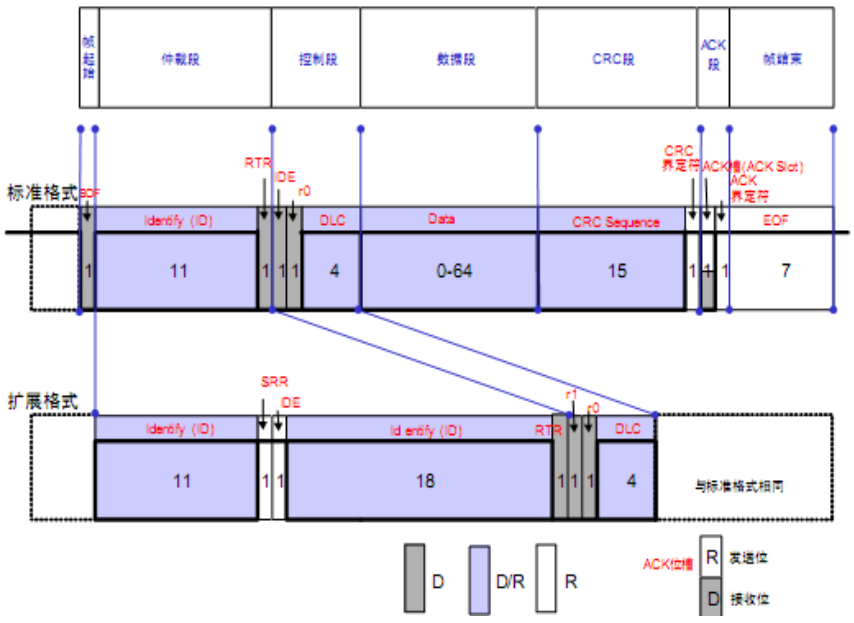


图 20-2 信号数据帧组成

21. 通用串行接口 UART

21.1 串口(UART)简介

通用异步串行接口（UART：Universal Asynchronous Receiver/Transmitter）提供串行通讯功能。

计算机模块提供了 4 个 UART 接口，其中 UART0、UART1、UART2 以及 UART3 控制器的结构和功能完全相同，是简单的 UART，只包括 TXD 和 RXD 两根信号线。UART 支持 8 数据位、一个可选的校验位和一个停止位的数据帧。为了生成不同的传输速率，每个 UART 有一个 12 位的可编程时钟分频器。有两个 FIFO 用于系统总线和 UART 之间的数据传输。有两个保持寄存器用于总线和 UART 之间的数据传输。UART 结构组成如下图 所示：

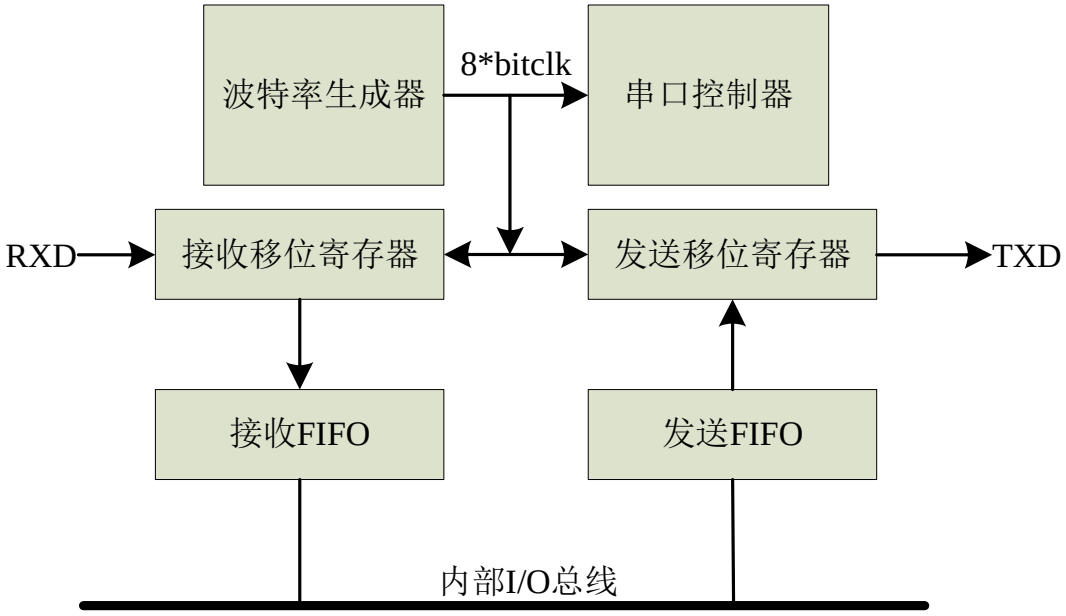


图 21-2 UART 结构框图

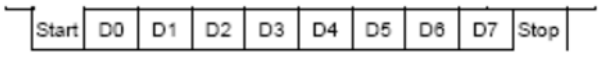
21.2 串口(UART)工作原理

21.2.1 发送操作

通过设置“UART 控制寄存器”的“TE”位来使能发送操作。通过写入数据寄存器的操作把要发送的数据写入长度为4个字节的FIFO里，当发送使能后，发送数据就会从“发送FIFO缓存区”送到“发送移位寄存器”，被转换成串行数据流，通过 TXD 引脚输出。UART 控制器

自动在8位数据的前面加上1位起始位，在其后面加上1位可选的奇偶校验位和1位停止位。如下图所示：

数据帧 ，无校验位：



数据帧 ，有校验位：

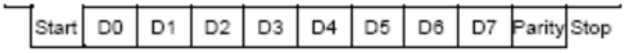


图 21-2 UART 数据帧

如果“发送 FIFO 缓存区”中没有新的字符，则输出引脚 TXD 保持高电平，“UART 控制寄存器”中的 TSRE（发送移位寄存器空）位被置‘1’。当一个新的字符被载入“发送 FIFO 缓存区”中，则发送重新开始，TSRE 位被清‘0’。当发送被禁止时，发送操作继续进行，直至当前正在发送的字符发完为止。当发送被禁止时，“发送保持寄存器”不能载入数据。

21.2.2 接收操作

通过设置“UART 控制寄存器”中的 RE（接收使能）位来使能接收操作。当接收器查找到一个从高到低的跳变，表示一个 start 开始位。”半位”周期后开始接受真正发送过来的原始数据。每接受一位信息后延时一个输入 terval，直到接收到 stop 位。每接受一位，接受移位寄存器就移位，接受完毕后其内的值会保持到新的传送到来为止。

在接收期间，最低有效位最早接收。然后数据被送入接收 FIFO 缓存区，UART 状态寄存器中的“数据准备好”位（DR 位）即数据 ready 位被置位。其他的状态位也在这时被刷新。如果接收保持寄存器或接收移位寄存器中有某一个没有保存，但这时又有了新的传送发生，那么接收移位寄存器中的数据会丢失，UART 状态寄存器中的过载状态位会被置位。如果“流控制”（flow control）被激活，则当接收到有效起始位时，并且接收 FIFO 也处在满状态的时候，RTSN 将被置‘1’；当接收保持寄存器被读取，RTSN 会被自动重新判断置位。

21.2.3 波特率设置

每个 UART 串口都有一个 20 位的倒计时计数器来产生需要的波特率，计数器的时钟来源于系统时钟且当发生溢出时会发出标志（tick）信号。溢出后计数器的值会被 UART 的重加载寄存器更新。溢出标志频率应该是所设定波特率的 8 倍。

分频值 = (((Sysclk * 10) / (波特率 * 8)) - 5) / 10 （式 20-1）

21.2.4 自环模式

当设置 UART 控制寄存器的 LB 位为‘1’时，UART 模块进入自环模式。在该模式下，传输的输出数据内部连接到接收的输入端。这时，可以进行一些自环测试来验证发送操作、接收操作、软件设置等功能。在自环模式下，输出端口保持不激活状态，防止有数据输出。

21.2.5 FIFO 调试模式

当设置 UART 控制寄存器的调试模式位为‘1’时，UART 模块进入 FIFO 调试模式，在该模式下通过 FIFO 调试寄存器可以读取发送 FIFO 和写入接收 FIFO 等功能。在调试模式下，发送输出保持无效状态；如果接收中断使能，当写入接收 FIFO 数据时，UART 模块就会产生中断信号。

21.2.6 中断机制

UART 控制器有两种中断，一种是普通中断，另外一种是 FIFO 中断。对于发送模块，

当发送中断（TI）位使能、发送使能、发送 FIFO 变为空时候，产生普通中断；当 FIFO 中断（TF）位使能、发送使能、FIFO 剩少于一半有效数据时，产生 FIFO 中断。

UART 接收中断也是一样设置。

21.3 串口寄存器

表 21-1 UART寄存器

地址	读写	寄存器
0x80000100	R/W	UART1 数据寄存器
0x80000104	R	UART1 状态寄存器
0x80000108	R/W	UART1 控制寄存器
0x8000010C	R/W	UART1 分频寄存器
0x80000900	R/W	UART2 数据寄存器
0x80000904	R	UART2 状态寄存器
0x80000908	R/W	UART2 控制寄存器
0x8000090C	R/W	UART2 分频寄存器
0x80100100	R/W	UART3 数据寄存器

0x80100104	R	UART3 状态寄存器
0x80100108	R/W	UART3 控制寄存器
0x8010010C	R/W	UART3 分频寄存器
0x80100200	R/W	UART4 数据寄存器
0x80100204	R	UART4 状态寄存器
0x80100208	R/W	UART4 控制寄存器
0x8010020C	R/W	UART4 分频寄存器

21.3.1 UART 数据寄存器

表 21-2 UART数据寄存器

位	位名称	位描述
[31:8]	RES	1：使能；0：禁能。
[7:0]	DATA	数据寄存器，读取输入的数据，写入输出的数据

21.3.2 UART 状态寄存器

表 21-3 UART状态寄存器

位	位名称	位描述
[31:26]	RCNT	接收器先进先出计数(RCNT) -显示在接收器先进先出中的数据帧数目
[25:20]	TCNT	发送器先进先出 计数(TCNT) -显示在发送机先进先出中的数据帧的数目
[19:11]	RES	保留
[10]	RF	接收器先进先出 满(RF) -声明接收器先进先出是满的
[9]	TF	发送器先进先出满(TF) -声明发送器先进先出是满的
[8]	RH	接收器先进先出半满(RH) -声明持有至少半个先进先出数据
[7]	TH	发送器 先进先出半满 1 (TH) -声明先进先出少于半满
[6]	FE	帧错误(FE) -声明帧错误被检测到
[5]	PE	奇偶错误 (PE) -声明奇偶错误被检测到
[4]	OV	溢出(OV) -声明一个或更多的字符由于溢出丢失
[3]	BR	突发接收 (BR) - 声明一个 BREAK 被检测到
[2]	TE	发送器先进先出空(TE) -声明发送器先进先出是空的

[1]	TS	发送器移位寄存器空（TS）-声明发送器移动寄存器是空的 FIFO 中无空间导致报文丢失时，为 1。
[0]	DR	数据准备位(DR) -声明新的数据在接收器保持寄存器是有效的

21.3.3 UART 控制寄存器

表 21-4 UART控制寄存器

位	位名称	位描述
[31:11]	RES	保留
[10]	RF	接收器先进先出中断使能(RF) -当被设定，接收器先进先出层中断使能
[9]	TF	发送器先进先出中断使能(TF) -当被设定，发送器先进先出层中断使能
[8]	EC	外部时钟(EC) - 当前无意义
[7]	LB	回环(LB) - 如果设定，回环模式被使能
[6]	FL	流控制(FL) -如果设定，使用 CTS/RTS 流控制使能
[5]	PE	奇偶使能(PE) -如果设定，奇偶生成和校验被使能
[4]	PS	奇偶选择(PS) -选择奇偶（0 =偶校验，1 =奇校验）
[3]	TI	发送器中断使能(TI) -如果设定，当发送完一个帧，中断生成
[2]	RI	接收器中断使能(RI) -如果设定，当接收到一个帧，中断生成
[1]	TE	发送器使能(TE) -如果设定，使能发送
[0]	RE	接收器使能(RE) -如果设定，使能接收

21.3.4 UART 分频寄存器

表 21-5 UART分频寄存器

位	位名称	位描述
[31:12]	RES	保留
[11:0]	SCALER RELOAD VALUE	分频重载值

22. 1553B 总线控制器

计算机模块内部集成二通道的 1553B 总线控制器，支持 BC、RT 和 BM 三种终端类型，支持完整的 MIL-STD-1553B 协议，数据传输速率 1Mbps 和 10Mbps 可配置，存储器布局和寄存器设置同 BU-61580 兼容。*（注：在模块中由于二通道的 1553B 总线控制器共用 10 号中断，建议用中断方式处理 1553B 请求时，建议只用一个通道。）*

22.1 主要特征

- 遵循 MIL-STD-1553B 标准(国军标 GJB289A-97 标准)；
- 操作方式、寄存器设置以及存储器布局等方面同 BU-61580 兼容；
- 支持的通讯类型包括：
  - BC → RT；
  - RT → BC；
  - RT → RT；
  - Broadcast；
  - Mode code；
- 能被配置为 BC、RT、BM 三种类型的控制器；
- 支持 1Mbps、10Mbps 两种传输速度；
- 带 4K*16Bit 的集成 DPRAM；
- 外部接口支持通用的 1553B 总线收发器：HI1567、HI1573 等
- 带 A、B 双冗余通道；
- BC 性能：
  - 支持 A/B 区域；
  - 具有自动重发功能；
  - 可编程的消息间隔时间；
  - 帧自动重复发送；
  - 可编程的超时响应时间；
- RT 性能：
  - 可编程的 RT 地址，子地址；

- 支持单缓冲存储器管理方式；
- 可编程的非法命令表；
- 可编程的方式代码中断表；
- 可编程的子地址忙表；

➢ BM 性能：

- 能够实时侦听总线上的数据流，可以将所有的数据流记录下来，也可以有选择地进行数据监听；
- 支持命令堆栈半满、全满溢出；
- 支持数据堆栈半满、全满溢出；
- 命令堆栈与数据堆栈独立；
- 对每条消息有相应的属性标志；

22.2 结构描述

计算机模块中的 1553B 主要包括通信协议模块 A、通信协议模块 B、主机信号接口模块、配置寄存器模块、存储管理模块和时钟管理与主控制模块等。其中 1553 通信协议模块部分用差分曼彻斯特编码实现时分命令响应式串行通讯，主要包括 A、B 通道编码器和解码器；主机信号接口模块实现 1553B 模块与 CPU 的接口，用来实现 CPU 对 1553B 模块的控制；配置寄存器主要实现对 1553B 模块功能的配置，能够间接反映 1553B 模块的功能；存储管理为 CPU 和 1553B 模块之间交互数据的管理方式，主要为 4K*16BIT 的双口 RAM； 时钟管理与主控制模块实现对输入时钟的管理和对 1553B 模块的基本通讯功能的实现与控制，是 1553B 模块的中心控制单元，能配置成 BC、RT、BM 三种类型的控制器。

OBT1553 模块的结构如图 所示，其内部各个子各模块与以及各个信号端口的说明分别如下表 22-1 和表 22-1 示：

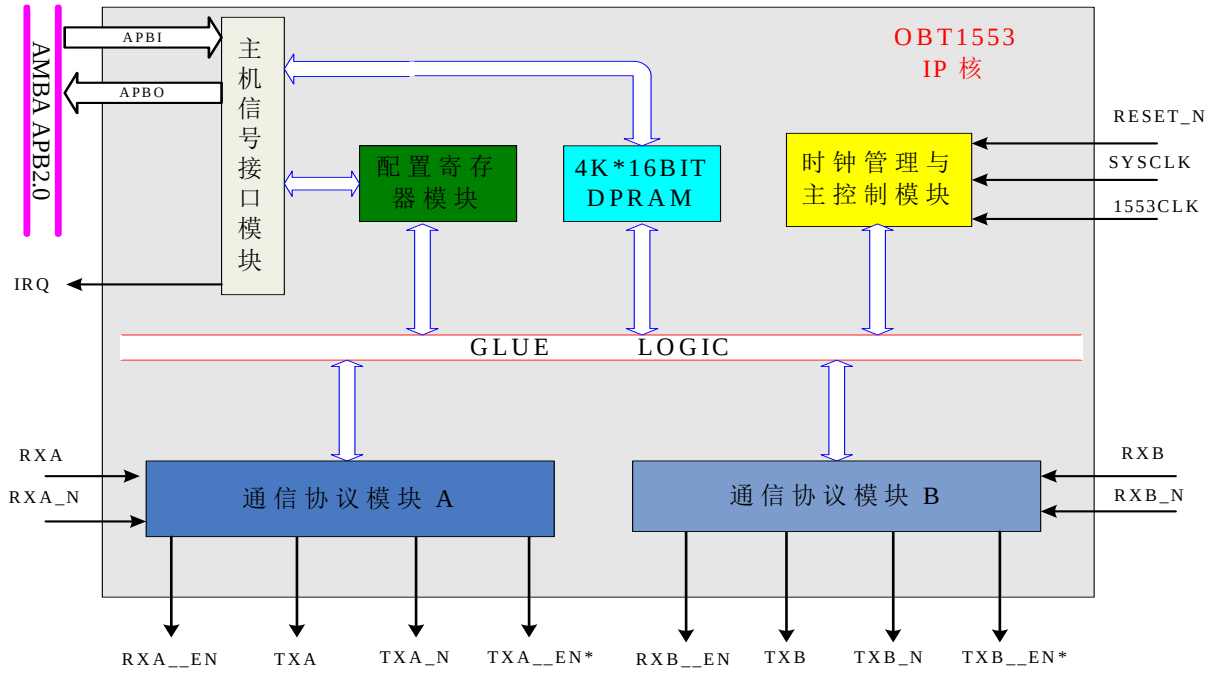


图 22-1 计算机模块中 1553B 模块结构框图

表 22-1 计算机模块中 1553B 模块各子模块说明

序号	模块名称	模块描述
1	通信协议模块 A/B	通信协议模块部分用差分曼彻斯特编码实现时分命令响应式串行通讯，主要包括 A、B 通道编码器和解码器
2	主机信号接口模块	主机信号接口模块实现 1553 IP 核与 CPU 的接口，用来实现 CPU 对 OBT1553B IP 核的控制
3	配置寄存器模块	配置寄存器主要实现对 OBT1553 IP 核功能的配置，能够间接反映 1553 IP 核的功能
4	存储管理模块（4K*16BIT DPRAM）	存储管理为 CPU 和 1553 IP 核之间交互数据的管理方式，主要为 4K*16BIT 的双口 RAM
5	时钟管理与主控制模块	时钟管理与主控制模块实现对输入时钟的管理和对 OBT1553 IP 核的基本通讯功能的实现与控制，是该 IP 核的中心控制单元能配置成 BC 或 RT 控制器

表 22-2　计算机模块中 1553B 模块的端口信号说明

序号	信号名称	信号方向	默认状态	信号描述
1	AMBA　APB2.0	I/O	均为 0	主机接口接 AMBA.APB2.0 信号，主要是地址线，数据线，读写控制线
2	RXA	I	0	通道 A 接收信号
3	RXA_N	I	0	通道 A 接收信号的反
4	RXA_EN	0	0	通道 A 接收使能信号,主要为外接 HI-1567PSI 芯片时的使能
5	TXA	0	0	通道 A 发送信号
6	TXA_N	0	0	通道 A 发送信号的反
7	TXA_EN*	0	1	通道 A 发送使能信号，主要为外接 HI-1567PSI 芯片时的使能
8	RXB	I	0	通道 B 接收信号
9	RXB_N	I	0	通道 B 接收信号的反
10	RXB_EN	0	0	通道 B 接收使能信号,主要为外接 HI-1567PSI 芯片时的使能
11	TXB	0	0	通道 B 发送信号
12	TXB_N	0	0	通道 B 发送信号的反
13	TXB_EN*	0	1	通道 B 发送使能信号，主要为外接 HI-1567PSI 芯片时的使能
14	RESET_N	I	1	外部复位信号低电平有效
15	SYSCLK	I	外部确定	主机输入时钟
16	1553CLK	I	16MHz	1553 收发模块专用时钟
17	IRQ	0	0	中断信号输出，高电平有效

22.3 功能描述

22.3.1 总线控制器（BC）

当主控制器配置为 BC 总线控制器时，则实现 BC 总线控制器功能。

BC 总线控制器控制通讯数据流的传输，是数据发送和接收的发起者和总线网络的管理者。控制计算机将数据写入 BC 总线控制器的内部存储器，并通过 开始/启动（SSR）寄存器来启动 BC 总线控制器进行数据传输。对于每个消息，BC 总线控制器通过 BC 控制字初始化 BC 总线控制器的状态从而发起数据传输（发送或者接收），并通过 BC 命令字通知 RT 响应数据传输（接收或者发送）。BC 总线控制器还可以通过模式命令对 RT 进行控制，包括读取同步和状态字等内容。

BC 总线控制器一方面通过 BC 模块状态字判断接受到的数据是否正确（包括奇偶校验）、响应是否超时等，另一方面通过读回的 RT 状态字，判断 RT 接收的数据是否正确、响应是否超时等。BC 模块状态字和读回的 RT 状态字均正常，说明数据传输正常。

如果传输过程中出现错误（BC 状态字和 RT 状态字异常），BC 总线控制器通过中断通知控制计算机进行处理，如消息重发。如果 BC 总线控制器出现灾难性故障，控制计算机（指 BC 的控制计算机）对 BC 复位。

22.3.2 远程终端（RT）

当主控制器配置为 RT 远程终端时，则实现 RT 远程终端功能。

RT 能根据协议在规定的时间内响应 BC 总线控制器发出的命令，进行数据接收或发送。RT 对输入信号进行检测，当检测到跟该 RT 地址一致的命令字后，响应数据传输。对于发送命令，RT 在发送数据之前将 RT 状态字通过 1553 总线发送给 BC；对于接收命令，RT 在接收完数据后将 RT 状态字通过 1553 总线发送给 BC。BC 通过 RT 状态字判断本次数据传输（发送/接收）是否有效。

当接受到模式命令后，RT 需要对接受到的模式命令进行响应。

22.3.3 总线监视器（BM）

BM 能够实时侦听总线上的数据流，可以将所有的数据流记录下来，也可以有选择地进行数据监听。并设有命令存储区半满、全满标志和数据存储区半满、全满标志。

22.4 地址空间分配

计算机模块中集成了 2 通道的 1553B 总线控制器，它们不仅功能、工作方式和操作方式

相同，而且寄存器和存储器的数量、格式、定义以及偏移地址等均相同，只是各自的寄存器/存储器的基地址不同。

表 22-3　计算机模块中 1553B 模块地址空间分配

偏移地址	基地址	寄存器的地址空间	存储器的地址空间
通道-1	0x80108000	0x80108000—0x8010BFFF	0x8010c000—0x8010FFFF
通道-2	0x80110000	0x80110000—0x80113FFF	0x80114000—0x80117FFF

1553B 模块中的存储器的容量为 4K*16bit，可以通过 APB 总线直接进行读写访问，且存储器的每 16bits 为一个被访问单位（即一个字），对应 APB 数据总线的低 16 位（此时，APB 数据总线的高 16 位未定义，读值总为 0）。

1553B 模块中的存储器的每一个字（16bits）占用 APB 的一个 32 位的地址空间，如 1553B 通道 1 的存储器的第 1 个字的地址为 0x80108000，第 2 个字的地址为 0x80108004，第 3 个字的地址为 0x80108008，第 4 个字的地址为 0x8010800C...以此类推。

22.5 寄存器定义及描述

表 22-4　计算机模块中 1553B 模块寄存器偏移地址分配

偏移地址	读/写	有效位宽	默认值 (HEX)	寄存器描述
0x00	RD/WR	16	0000	中断屏蔽寄存器(IMR)
0x01	RD/WR	16	0000	配置寄存器 1(CFG1)
0x02	RD/WR	16	0000	配置寄存器 2(CFG2)
0x03	WR	16	0000	启动/复位寄存器(SRR)
0x03	RD	16	0000	BC/RT/BM 命令堆栈指针寄存器(STACK_ADDR)
0x04	RD/WR	16	0000	BM 初始命令堆栈指针寄存器
0x05	RD	16	0000	时间标签寄存器 0(TTR0)
0x06	RD	16	0000	中断状态寄存器(INT_STA)
0x07	RD/WR	16	0000	配置寄存器 3 (CFG3)

偏移地址	读/写	有效位宽	默认值 (HEX)	寄存器描述
0x08	RD/WR	16	0000	配置寄存器 4(CFG4)
0x09	RD/WR	16	0000	配置寄存器 5(CFG5)
0x0A	RD	16	0000	BM 数据堆栈指针寄存器（BM_STACK_ADDR）
0x0B	RD	16	0000	1Mbps/10Mbps 配置寄存器
0x0C	RD	16	0000	保留
0x0D	RD	16	0000	RT 上一命令字寄存器(LAST_CMD)
0x0D	WR	16	0000	BC 帧时间寄存器
0x0E	RD	16	0000	RT 状态字寄存器（RT_STA）
0x0F	RD	16	0000	RT BIT 字寄存器(RT_BIT_REG)
0x10	RD	16	0000	时间标签寄存器 1(TTR1)
0x11	RD	16	0000	
0x12	RD/WR	16	0000	

表 22-5　计算机模块中 1553B 模块中断屏蔽寄存器（IMR）

位(BIT)	描述 (DESCRIPTION)
15	保留
14	RAM 奇偶校验错误（RAM PARITY ERROR）
13	BC/RT 传输器超时（BC/RT TRANSMITTER TIMEOUT）
12	BC/RT 命令堆栈溢出/ BM 命令堆栈半满溢出
11	BM 命令堆栈溢出（BM COMMAND STACK ROLLOVER）
10	BM 数据堆栈溢出（BM DATA STACK ROLLOVER）
9	保留
8	BC 重发/BM 数据半满溢出
7	RT 地址奇偶校验错误（RT ADDRESS PARITY ERROR）
6	时间标签寄存器溢出（TIME TAG ROLLOVER）
5	保留

位(BIT)	描述 (DESCRIPTION)
4	BC 消息消息结束（BC MSG EOM）
3	BC 帧结束（BC END OF FRAME）
2	格式错误(FORMAT ERROR)
1	BC 状态置位/RT 方式代码（BC STATUS SET/RT MODE CODE）
0	消息结束（END OF MESSAGE）

中断屏蔽寄存器中位如置 1 表示打开对应中断状态寄存器的中断，如置 0 表示关闭对应的中断状态寄存器的中断。

- **BIT14:** 当使能（置 1）该位则 RAM 数据奇偶校验出错时产生中断。
- **BIT13:** IP 核内置看门狗，当使能(置 1)该位则传输编码时间超过 668us 时产生中断。
- **BIT12:** IP 核命令堆栈大小为 256 字，当使能（置 1）该位则在 BC/RT 类型下命令堆栈指针超出 256 则产生中断；当在 BM 类型下则在命令堆栈半满时产生中断。
- **BIT11:** 当使能（置 1）该位则 BM 命令堆栈溢出时产生中断。
- **BIT10:** 当使能（置 1）该位则 BM 数据堆栈溢出时产生中断。
- **BIT8:** 当使能（置 1）该位则在 BC 类型下 BC 重发消息前将产生中断；如果在 BM 类型下数据堆栈半满溢出将产生中断。
- **BIT7:** 在 RT 模式下当使能（置 1）该位那么如果 RTAD4-RTAD0 与 RTADP 共六位进行奇偶的结果是 0 就产生中断。
- **BIT6:** 当使能（置 1）该位那么当时标寄存器 TT2 溢出时产生中断。
- **BIT4:** 当使能（置 1）该位那么当作 BC 总线控制器时 BC 控制字中的 BIT4 位为 1 则消息结束产生中断。
- **BIT3:** 当使能（置 1）该位那么 BC 帧发送结束产生中断。
- **BIT2:** 当使能（置 1）该位那么格式错误时产生中断。
- **BIT1:** 当使能（置 1）该位那么当作 BC 时，BC 收到的状态字中有置 1 的位，则产生中断；当作 RT 时子地址查找表中方式字对应位为 1 则产生中断。
- **BIT0:** 当使能（置 1）该位那么当 BC/RT 发送/接收消息结束产生中断。

表 22-6 计算机模块中 1553B 模块 BC 配置寄存器 1(BC-CFG1)

位（BIT）	描述 (DESCRIPTION)

位（BIT）	描述（DESCRIPTION）
15(MSB)	BC/RT/BM 模式设置（RT/BM*-BC*）
14	BC/RT/BM 模式设置（BM/RT*- BC*）
13	A*/B 区域设置（CURRENT AREA A*/B）
12-9	保留
8	帧自动重复发送使能（FRAME AUTO-REPEAT）
7-6	保留
5	消息间隔时间使能（MESSAGE GAP TIMER ENABLED）
4	消息重发使能（RETRY ENABLED）
3	消息重发一次或二次选择（DOUBLE/SINGLE* RETRY）
2	BC 使能（BC ENABLED），该位只读
1	BC 帧信息忙指示（BC FRAME IN PROGRESS），该位只读
0（LSB）	BC 消息忙指示（BC MESSAGE IN PROGRESS），该位只读

BC 配置寄存器 1 主要用作 1553 总线控制器工作模式的选择，还有是否使能消息重发、帧重复发送功能，以及 1553 总线控制器工作状态的指示等。

- **BIT15,BIT14:** 此两位组合为 00 设置为 BC 模式,10 设置为 RT 模式,01 设置为 BM 模式，11 则保持为上电初始状态。
- **BIT13：** 该位为 0 则使用 RAM 的 A 区域，该位为 1 则使用 RAM 的 B 区域。
- **BIT8:** 该位为 0 则 BC 发送完一帧数据就停止，若该位为 1 则帧重复发送直到启动/复位寄存器(SSR)中的 BIT0(RESET)、BIT5(STOP_ON_FRAME)，BIT6(STOP_ON_MESSAGE)中任意位为 1 才会停止。
- **BIT5:** 该位为 0 则消息之间间隔时间固定为近似 8 到 11us,这位为 1 则消息之间的间隔时间通过 BC 命令堆栈的第三个字指定，其指定范围为最小约 8us 到最大约 65535us，时间精度为 1us。当为 10Mbps 传输速度时则相应为以前的 0.2 倍。
- **BIT4:** 该位为 0，BC 对所有的消息都不重发， 该位为 1 且 BC 控制字的 BIT8 也为 1 那么该消息在返回状态字出错，响应时间超时则重发消息。
- **BIT3:** 在配置寄存器 1（CFG1）的 BIT4 为 1 的条件下，该位为 0 则该消息在返回状态字出错，响应时间超时则重发一次；该位为 1 则该消息在返回状态字出错，响应时间超时则重发消息两次。

- **BIT2:** 该位为只读位，含义同 BIT1。
- **BIT1:** 该位为只读位，在帧的第一个消息启动后到帧的最后一个消息结束一直被设为 1，在帧自动重复发送模式下则一直保持为 1 直到帧重复发送结束。
- **BIT0:** 该位在 BC 总线控制器每个消息开始传输时置为 1，在消息结束传输时清为 0。

表 22-7 计算机模块中 1553B 模块 RT 配置寄存器 1(RT- CFG1)

位（BIT）	描述 (DESCRIPTION)
15(MSB)	BC/RT/BM 模式设置（RT/BM*-BC*）
14	BC/RT/BM 模式设置（BM/RT*- BC*）
13-12	保留
11	动态总线控制接收*（DYNAMIC BUS CONTROL ACCEPTANCE*）
10	忙*（BUSY*）
9	服务请求*（SERVICE REQUEST*）
8	子系统标志*（SUBSYSTEM FLAG*）
7	RT 标志*（RTFLAG*）
6-1	保留
0	BC 消息忙指示（BC MESSAGE IN PROGRESS），该位只读

- **BIT15,BIT14:** 此两位组合为 00 设置为 BC 模式，10 设置为 RT 模式，01 设置为 BM 模式，11 则保持为上电初始状态。
- **BIT11:** 该位为 0 则 RT 状态字寄存器的 BIT1 位为 1。
- **BIT10:** 该位为 0 则 RT 状态字寄存器的 BIT3 位为 1。
- **BIT9:** 该位为 0 则 RT 状态字寄存器的 BIT8 位为 1。
- **BIT8:** 该位为 0 则 RT 状态字寄存器的 BIT2 位为 1。
- **BIT7:** 该位为 0 则 RT 状态字寄存器的 BIT0 位为 1。
- **BIT0:** 该位在 BC 总线控制器每个消息开始传输时置为 1，在消息结束传输时清为 0。

表 22-8 计算机模块中 1553B 模块配置寄存器 2(CFG2)

位（BIT）	描述 (DESCRIPTION)
15-14	保留

位（BIT）	描述 （DESCRIPTION）
13	忙查找表使能(BUSY LOOK UP TABLE ENABLE)
12-10	保留
9-7	时间标签最小精度设置（TIME TAG RESOLUTION2，1，0）
6	同步清除时标寄存器使能（CLEAR TIME TAG ON SYNCHRONIZE）
5	同步重载时标寄存器使能（LOAD TIME TAG ON SYNCHRONIZE）
4	中断状态自动清除（INTERRUPT STATUS AUTO CLEAR）
3	电平/脉冲中断(LEVEL/PULSE *INTERRUPT REQUEST)
2	清除服务请求(CLEAR SERVICE REQUEST)
1-0	保留（其中 BIT1 可进行读写，但没有实际意义）

- **BIT13:** 该位为 1 则使能 RT 的忙位查找表。
- **BIT9，BIT8，BIT7:** 000 则最小精度为 64us，001 则最小精度为 32us，010 则最小精度为 16us，011 则最小精度为 8us，100 则最小精度为 4us，101 则最小精度为 2us，110 则最小精度为 1us，111 则最小精度为 128us。当传输速率为 10Mbps 时则最小精度缩小 5 倍，也就是 000 则最小精度为 12.8us，001 则最小精度为 6.4us 依此类推。
- **BIT6:** 该位为 1 则当 RT 收到同步方式字(方式代码为 00001)时 RT 的时间标签寄存器清 0。
- **BIT5:** 该位为 1 则当 RT 收到同步方式字(方式代码为 10001)时 RT 的时间标签寄存器重新导入方式代码带的数据值。
- **BIT4:** 该位为 1 则 CPU 读出中断状态寄存器的值后，中断状态字寄存器自动清 0。
- **BIT3:** 该位为 0 产生脉冲中断信号，为 1 则产生电平中断信号，在该 IP 核中建议用电平中断。
- **BIT2:** 该位为 1 则当 RT 收到方式字(方式代码为 10000)时，将自动将服务请求撤消。也就是将 RT 配置寄存器 1 的 BIT9 置 1， RT 状态寄存器的 BIT8 置 0。

表 22-9 计算机模块中 1553B 模块启动/复位寄存器（SRR）

位（BIT）	描述 (DESCRIPTION)
15-7	保留

位（BIT）	描述 (DESCRIPTION)
6	BC 停止消息发送（BC STOP-ON-MESSAGE）
5	BC 停止帧发送（BC STOP-ON-FRAME）
4	保留
3	时间标签寄存器清零（TIME TAG RESET）
2	中断状态寄存器清零（INTERRUPT RESET）
1	BC/BM 启动（BC/BM START）
0	系统软复位（RESET）

启动/复位寄存器(SSR)用作“命令”类型的功能，能实现软复位，BC 启动，中断状态寄存器复位，时间标签寄存器(TTR)复位，在帧自动重复发送时还可以停止帧的自动复重发送。

- **BIT6:** 置 1 则在一个正在发送的消息发送完毕后即停止 BC 工作，如果没有消息在处理则立即停止 BC 工作。
- **BIT5:** 置 1 则在一个正在发送的帧发送完毕后即停止 BC 工作，如果没有帧在处理则立即停止 BC 工作。
- **BIT3:** 置 1 清时间标签寄存器 0,1,2 均被清为 0。
- **BIT2:** 置 1 除了中断状态寄存器 的 BIT 7 位（RT 地址奇偶位错）不被清除其余位均被清除到 0。
- **BIT1:** 置 1 时在 BC 模式下启动帧传输;在 BM 模式下启动 BM 监视。
- **BIT0:** 置 1 则进行软复位，立即停止正在进行的处理。所有的寄存器和内部状态都被复位到上电时的初始态。

表 22-10 计算机模块中 1553B 模块 BC/RT 命令堆栈指针寄存器(STACK_ADDR)

位（BIT）	描述 （DESCRIPTION）
15-0	BC/RT/BM 命令堆栈指针

BC/RT/BM 命令堆栈寄存器主要寄存 BC/RT/BM 命令堆栈指针，当作 BC 时将消息数据读出后该指针递增 4；当作 RT 时 RT 接到新的消息时该指针递增 4； 当作 BM 时 BM 接到新的消息时该指针递增 4。

表 22-11 计算机模块中 1553B 模块 BM 初始命令堆栈指针寄存器（INIT_STACK_ADDR)

位（BIT）	描述 (DESCRIPTION)
15-0	BM 命令堆栈指针初始位置

BM 初始命令堆栈指针寄存器主要用于设置最初的命令堆栈指针,默认为 0X0000H 即从 RAM 的第一个单元开始保存接收到的数据。

表 22-12　计算机模块中 1553B 模块时间标签寄存器 0（TTR）

位（BIT）	描述 （DESCRIPTION）
15-0	时间计时标签位

时间标签寄存器 0 用于寄存 OBT1553 计时结果的 BIT15-BIT0 位。

表 22-13　计算机模块中 1553B 模块中断状态寄存器（INT_STA）

位（BIT）	描述 (DESCRIPTION)
15	中断请求（MASTER INTERRUPT）
14	保留
13	BC/RT 传输器超时（BC/RT TRANSMITTER TIMEOUT）
12	BC/RT 命令堆栈溢出/BM 命令堆栈半满溢出
11	BM 命令堆栈溢出（BM COMMAND STACK ROLLOVER）
10	BM 数据堆栈溢出（BM DATA STACK ROLLOVER）
9	保留
8	BC 重发（BC RETRY）/BM 数据半满溢出
7	RT 地址奇偶校验错误（RT ADDRESS PARITY ERROR）
6	时间标签寄存器溢出（TIME TAG ROLLOVER）
5	保留
4	BC 消息结束（BC MSG EOM）
3	BC 帧结束（BC END OF FRAME）
1	BC 状态置位/RT 方式代码（BC STATUS SET/RT MODE CODE）
0	消息结束（END OF MESSAGE）

- **BIT15:** BIT14-BIT0 中的任意一位为 1 则该位为 1。
- **BIT13:** IP 核内置看门狗，当传输编码时间超过 668us 时该位置 1（即每次同步头加所

有数据个数传输时间超过 668us 时则传输器超时)。

- **BIT12:** IP 核命令堆栈大小为 256 字，当在 BC/RT 类型下命令堆栈指针超出 256 时该位置 1；当在 BM 类型下则在命令堆栈半满时该位置 1。
- **BIT11:** BM 命令堆栈溢出时则该位为 1。
- **BIT10:** BM 数据堆栈溢出时则该位为 1。
- **BIT8:** 在 BC 类型下 BC 重发消息前将时该位置 1；如果在 BM 类型下数据堆栈半满溢出时该位置 1。
- **BIT7:** 在 RT 模式下那么如果 RTAD4-RTAD0 与 RTADP 共六位进行奇偶的结果是 0 就时该位置 1。
- **BIT6:** 当时标寄存器计时到 TTR2 溢出时该位置 1,也就是说这一位跟 TTR0、TTR1，TTR2 都有关系。
- **BIT4:** 当作在中断屏蔽寄存器相应位不使能的情况下如果为 BC 总线控制器时 BC 控制字中的 BIT4 位为 1 则消息结束时该位置 1。
- **BIT3:** BC 帧发送结束时该位置 1。
- **BIT2:** 格式错误时该位置 1，格式错误是指响应超时、奇偶校验错、编码错、计数错等。也就是说它包括了响应超时错和 BC 块状态字的格式错，所以中断状态寄存器的格式错置 1 了则有可能是 BC 块状态字的格式错位(BIT10)置 1 或 BC 块状态字的响应超时位（BIT9）置 1。
- **BIT1:** 当作 BC 时，BC 收到的状态字中有置 1 的位，则产生中断；当作 RT 时子地址查找表中方式字对应位为 1 时该位置 1。
- **BIT0:** 当 BC/RT 发送/接收消息结束时该位置 1。

表 22-14 计算机模块中 1553B 模块配置寄存器 3(CFG3)

位（BIT）	描述 (DESCRIPTION)		
15-13	保留		
12-11	BM 命令堆栈大小设置位 1,0	BIT12,BIT11	全满消息条数
		00	20
		01	40
		10	80
		11	160

<table>
<tr><th>位（BIT）</th><th colspan="3">描述 (DESCRIPTION)</th></tr>
<tr><td rowspan="9">10-8</td><td rowspan="9">BM 数据堆栈大小设置位 2-0</td><td>BIT10, BIT9, BIT8</td><td>全满字个数</td></tr>
<tr><td>000</td><td>3328</td></tr>
<tr><td>001</td><td>1664</td></tr>
<tr><td>010</td><td>832</td></tr>
<tr><td>011</td><td>416</td></tr>
<tr><td>100</td><td>208</td></tr>
<tr><td>101</td><td>624</td></tr>
<tr><td>110</td><td>1248</td></tr>
<tr><td>111</td><td>2496</td></tr>
<tr><td>7</td><td colspan="3">非法命令查找表使能（ILLEGALIZATION DISABLED）</td></tr>
<tr><td>4</td><td colspan="3">接收非法命令屏蔽（ILLEGAL RX TRANSFER DISABLE）</td></tr>
<tr><td>3</td><td colspan="3">接收忙屏蔽(BUSY RX TRANSFER ENABLE)</td></tr>
<tr><td>2-1</td><td colspan="3">保留</td></tr>
<tr><td>0</td><td colspan="3">增强方式代码功能(ENHANCED MODE CODE HANDLING)</td></tr>
</table>

- **BIT12, BIT11:** 这两位为 BM 命令堆栈大小设置位，“00”时表示收到 20 条消息时全满溢出，半满溢出则是收到 10 条消息；“01”时表示收到 40 条消息时全满溢出，半满溢出则是收到 20 条消息；“10”时表示收到 80 条消息时全满溢出，半满溢出则是收到 40 条消息；“11”时表示收到 160 条消息时全满溢出，半满溢出则是收到 80 条消息。

- **BIT10-BIT8:** 这三位为 BM 数据堆栈大小设置位，当全满溢出时接收到的数据字个数如表中所示，半满溢出则少一半。如“000”时表示收到 3328 个数据字时全满溢出，半满溢出则是收到 1664 个数据字。

- **BIT7:** 该位为 1，使能 RT RAM 中的非法命令查找表。

- **BIT4:** 该位为 0，则在接收到非法命令时将接收到的数据写入 RAM 中，否则在接收到非法命令时不将接收到的数据写入 RAM 中。

- **BIT3:** 该位为 0，则在忙时将接收到的数据写入 RAM 中，否则在忙时不将接收到的数据写入 RAM 中。

- **BIT0:** 该位为 1，使能 RT RAM 中的方式代码查找表。

表 22-15　计算机模块中 1553B 模块配置寄存器 4(CFG4)

位（BIT）	描述 (DESCRIPTION)
15-9	保留
8	第一次重发通道选择（FIRST RETRY ALT/SAME* BUS）
7	第二次重发通道选择（SECOND RETRY ALT/SAME* BUS）
6-4	保留
3	RT 地址配置使能(LATCH RT ADDRESS WITH CFG REG #5)
2-0	保留

- **BIT8:** 置 0 则在最初发送的消息失败后第一次重发的消息与最初发送的消息在同一通道上传输。置 1 则在最初发送的消息失败后第一次重发的消息不再最初发送的消息的通道上传输。
- **BIT7:** 置 0 则在第一次重发的消息失败后第二次重发的消息与第一次重发的消息在同一通道上传输。置 1 则在第一次重发的消息失败后第二次重发的消息不再第一次重发的消息的通道上传输。此位只有在配置寄存器 1（CFG1）的 BIT3 位为 1 才有效。
- **BIT3:** 当该位为 1 时配置寄存器 5 的 BIT5-BIT0 才可写。

表 22-16　计算机模块中 1553B 模块配置寄存器 5(CFG5)

位（BIT）	描述 (DESCRIPTION)
15-11	保留
10-9	超时响应时间设置(RESPONSE TIMEOUT SELECT1,0)
8-6	保留（其中 BIT6 可进行读写，但没有实际意义）
5-1	RT 地址位 4-0(RT ADDRESS4-ADDRESS0)
0	RT 地址奇偶位(RT ADDRESS PARITY)

- **BIT10,BIT9:** 超时响应时间设置,如为 00是 19us，01是23us，10是51us，11是130us。当为 10Mbps 的传输速度时则均为 1M 的 0.2 倍，如 00 是 3.8us。
- **BIT5-BIT1:** 配置 RT 地址。
- **BIT0:** 配置 RT 校验位，该位与 BIT5-BIT1 的异或结果要为 1。

表 22-17　计算机模块中 1553B 模块 BM 数据堆栈指针寄存器（BM_STACK_ADDR)

位（BIT）	描述 (DESCRIPTION)

位（BIT）	描述 (DESCRIPTION)
15-0	BM 数据堆栈指针

BM 数据堆栈寄存器主要寄存 BM 数据堆栈指针，BM 接到新的数据时该指针递增 1。

表 22-18　计算机模块中 1553B 模块 1Mbps/10Mbps 配置寄存器（1M_10M_SEL）

位（BIT）	描述 (DESCRIPTION)
15-1	保留
0	1Mbps/10Mbps 设置位

- **BIT0**：该位为 0 选择的是 1Mbps 的传输率，该位为 1 选择的是 10Mbps 的传输率。

表 22-19　计算机模块中 1553B 模块 BC 帧时间/RT 上一命令字寄存器（LAST_CMD）

位（BIT）	描述 (DESCRIPTION)
15-0	BC 帧时间/RT 上一命令字寄存器

该寄存器用作 BC 帧时间设置寄存器时为可写，RT 上一命令字时为可读。

表 22-20　计算机模块中 1553B 模块 RT 状态字寄存器(RT_STA)

位（BIT）	描述 (DESCRIPTION)
15-11	均为 0
10	消息错误(MESSAGE ERROR)
9	测试手段(INSTRUMENTATION)
8	服务请求(SERVICE REQUEST)
7-5	保留
4	广播指令接收(BROADCAST COMMAND RECEIVED)
3	忙(BUSY)
2	子系统标志(SYBSYSTEM FLAG)
1	动态总线控制接收(DYNAMIC BUS CONTROL ACCEPT)
0	终端标志(TERMINAL FLAG)

- **BIT10**：如该位为 1 则表明有消息错误。消息错误是指响应超时、奇偶校验错、编码错、计数错，非法命令等。
- **BIT9**：该位在 OBT1553B IP 中一直为 0。

- **BIT8:** 如该位为 1 则表明 RT 有服务请求。
- **BIT4:** 如该位为 1 则表明通讯方式是广播通讯方式。
- **BIT3:** 如该位为 1 则表明系统正忙。
- **BIT2:** 子系统标志位。
- **BIT1:** 动态总线控制接收标志位。
- **BIT0:** 终端标志位。

表 22-21 计算机模块中 1553B 模块 RT BIT 字寄存器(RT_BIT_REG)

位（BIT）	描述 (DESCRIPTION)
15	传输器超时（TRANSMITTER TIMEOUT）
14-12	保留
11	通道 B 发送器关闭（TRANSMITTER SHUTDOWN B）
10	通道 A 发送器关闭（TRANSMITTER SHUTDOWN A）
9	终端标志禁止(TERMINAL FLAG INHIBITED)
8	总线传输通道 B/A*(CHANNEL B/A*)
7	保留
6	字计数错（WORD COUNT ERROR）
5	错误数据同步头（INCORRECT SYNC RECEIVED）
4	奇偶/位计数错(PARITY/BIT COUNT ERROR)
3	RT-RT 同步头/地址错(RT-RT SYNC/ADDRESS ERROR)
2	RT-RT 响应超时（RT-RT NO RESPONSE ERROR)
1	RT-RT 第二个命令字错(RT-RT 2ND COMMAND WORD ERROR)
0	命令字内容错(COMMAND WORD CONTENTS ERROR)

- **BIT15:** 当每次同步头加所有数据个数传输时间超过 668us 时则传输器超时，此时该位置 1。
- **BIT11:** 当接收到关闭通道方式代码时如果通道 B 发送器被关闭则该位置 1。
- **BIT10:** 当接收到关闭通道方式代码时如果通道 A 发送器关闭则该位置 1。
- **BIT9:** 当接收到终端标志禁止方式代码时则该位置 1。
- **BIT8:** 消息传输在 A 通道进行为 0，消息传输在 B 通道进行为 1。
- **BIT6:** RT 接收到的字计数错则该位置 1。

- **BIT5:** RT 接收到的数据同步头出错时该位置 1。
- **BIT4:** RT 接收到的数据有奇偶校验错或位计数错则该位置 1。
- **BIT3:** RT-RT 同步头/地址错则该位置 1，包括了第二个命令字同步头错，RT 地址错。
- **BIT2:** RT-RT 通讯时响应超时则该位置 1。
- **BIT1:** RT-RT 通讯时第二个命令字错则该位置 1，包括了第二个命令字的位计数错、奇偶校验错。
- **BIT0:** RT-RT 第二个命令字内容出错则该位置 1，它指的是 RT 地址错。

表 22-22 计算机模块中 1553B 模块时间标签寄存器 1（TTR1）

位（BIT）	描述 （DESCRIPTION）
15-0	时间计时标签位

时间标签寄存器用于寄存 OBT1553 计时结果的 BIT31-BIT16 位。

表 22-23 计算机模块中 1553B 模块 BC 控制字（BC_CTRL）

位（BIT）	描述 (DESCRIPTION)
15	保留
14	消息格式错误屏蔽（M.E.MASK）
13	服务请求位屏蔽(SERVICE REQUEST BIT MASK)
12	忙位屏蔽(SUBSYS BUSY BIT MASK)
11	子系统标志位屏蔽(SUBSYS FLAG BIT MASK)
10	终端标志位屏蔽(TERMINAL FLAG BIT MASK)
9	保留
8	重试使能(RETRY ENABLED)
7	总线通道选择 A/B*(bus channel a/b*)
6	保留
5	保留
4	EOM 中断使能(EOM INTERRUPT ENABLE)
3	保留
2	模式命令(MODE CODE FORMAT)

位（BIT）	描述 (DESCRIPTION)
1	广播命令(BROADCAST FORMAT)
0	RT2RT(RT-TO-RT FORMAT)

- **BIT14-BIT10:** 这 5 位均是为 1 则屏蔽相应的 RT 状态字位，如果 RT 状态字位某位被屏蔽则该位不影响中断状态寄存器的（INT_STA）的 BIT1 位和 BC 块状态字的 BIT7 位。
- **BIT8:** 置 1 且配置寄存器 1(CFG1)的 bit4 为 1 则在响应超时和格式错误时消息重发。
- **BIT7:** 置 1 消息传输通过 A 通道，置 0 消息传输通过 B 通道。
- **BIT4:** 该位置 1 且中断屏蔽寄存器（IMR）的 BIT4 也置 1 那么则消息结束中断状态寄存器(INT_STA)的 BIT4 为 1。
- **BIT2,BIT1,BIT0:** 置为 000 且命令字的 T/R*位是 0 则是 BC-> RT 通讯方式，置为 000 且命令字的 T/R*位是 1 则是 RT->BC 的通讯方式； 置为 001 是 RT-RT 通讯方式；置为 010 是 Broadcast 通讯方式；置为 100 是 Mode Code 通讯方式； 置为 110 是 Broadcast Mode Code 通讯方式。 其它两种组合没用到。

表 22-24 计算机模块中 1553B 模块 BC 命令字(BC_CMD)

位（BIT）	描述 (DESCRIPTION)
15-11	远程终端地址（REMOTE TERMINAL ADDRESS）
10	发送/接收* (T/R*)
9-5	子地址/方式(SUBADDRESS/MODE)
4-0	数据字计数/方式代码(DATA WORD COUNT/MODE CODE)

表 22-25 计算机模块中 1553B 模块 BC 块状态字(BC_BLK)

位（BIT）	描述 (DESCRIPTION)
15	消息传输结束标志位（EOM）
14	保留
13	传输通道指示(CHANNEL B/A*)
12	出错标志(ERROR FLAG)
11	状态设置（STATUS SET）
10	格式错误(FORMAT ERROR)

位（BIT）	描述 (DESCRIPTION)
9	响应超时(NO RESPONSE TIMEOUT)
8	保留
7	屏蔽状态设置（MASKED STATUS SET）
6-5	重发消息次数(RETRY COUNT 1,0)
4	数据传输正常（GOOD DATA BLOCK TRANSFER）
3	错误的状态地址（WRONG STATUS ADDRESS）
2	字计数错误（WORD COUNT ERROR）
1	同步头出错(INCORRECT SYNC TYPE)
0	无效字（INVALID WORD）

该字主要用来说明消息发送/接收后的状态。该存储器字如果有重发消息，只表明的是最后一个消息的状态。

- **BIT15:** 消息转输结束该位置 1。
- **BIT13:** 消息传输在 A 通道进行为 0，消息传输在 B 通道进行为 1。
- **BIT12:** 有格式错误或响应超时产生该位为 1。
- **BIT11:** 接受的 RT 返回字中 BIT10-BIT0 中只要有一位为 1 则该位为 1。
- **BIT10:** 格式错误包括字计数个数错、同步头错、奇偶校验错，如果其中任意一位为 1 则该位为 1。
- **BIT9:** RT 响应超时则该位置 1。
- **BIT7:** 在 BC 控制字中如果 BIT14-BIT10 中任意一位为 0（不屏蔽），则该位对应的 RT 状态字为 1 则该位置 1。
- **BIT6,5:** 记录 BC 重发消息的个数，00 表示没有重发，01 表示重发了一次，11 表示重发了两次。
- **BIT4:** 当 RT2BC,RT2RT,或者传输带数据方式代码，没有消息格式错则为 1，有消息格式错为 0；当 BC2RT，带数据方式代码收和不带数据方式代码，为 0。
- **BIT3:** RT 返回的状态字中 RT 地址有错则该位为 1。
- **BIT2:** 当 RT2BC,RT2RT,或者传输带数据方式代码，字计数有错则为 1，没错为 0；当 BC2RT，带数据方式代码收和不带数据方式代码，为 0。
- **BIT1:** 同步头出错该位置 1。

- **BIT0:** 当数据传输中有奇偶错、位计数错和同步头错则该位置 1。

表 22-26 计算机模块中 1553B 模块 RT 子地址控制字(RT_SUB_CTRL)

位（BIT）	描述 (DESCRIPTION)
15	RX: 接收双缓存使能（DOUBLE BUFFER ENABLE）
14	保留
13-10	保留
9	保留
8-5	保留
4	保留
3-0	保留

- **BIT15:** 该位为 RT 接收双缓存使能位，当为 1 时使能双缓存模式，为 0 则是单

表 22-27 计算机模块中 1553B 模块 RT 块状态字(RT_BLK)

位（BIT）	描述 (DESCRIPTION)
15	消息传输结束标志（EOM）
14	消息传输开始标志（SOM）
13	传输通道指示（CHANNEL B/A*）
12	出错标志(ERROR FLAG)
11	RT-RT 通讯方式标志(RT-RT FORMAT)
10	格式错误标志(FORMAT ERROR)
9	响应超时标志(NO RESPONSE TIMEOUT)
8	保留
7	保留
6	非法命令字(ILLEGAL COMMAND WORD)
5	字计数个数错(WORD COUNT ERROR)
4	数据同步头出错(INCORRECT DATA SYNC)
3	无效字（INVALID WORD）
2	RT-RT 同步头/地址错（RT-RT SYNCH/ADDRESS ERROR）

位（BIT）	描述 (DESCRIPTION)
1	RT-RT 第二个命令字错（RT-RT 2ND COMMAND ERROR）
0	命令字内容错（COMMAND WORD CONTENTS ERROR）

- **BIT15:** 消息传输结束该位置 1。
- **BIT14:** 消息传输开始该位置 1。
- **BIT13:** 消息传输在 A 通道进行为 0，消息传输在 B 通道进行为 1。
- **BIT12:** 有格式错误或响应超时产生该位为 1。
- **BIT11:** 在 RT-RT 通讯时接收 RT 时被设置为 1，发送 RT 时与该位无关。
- **BIT10:** 格式错误包括表中的 BIT6-BIT0 中任何一种错,如果 BIT6-BIT0 中出现其中任何一种错误则该位为 1。
- **BIT9:** RT-RT 通讯时响应超时则该位置 1。
- **BIT6:** RT 接收到非法命令字则该位置 1。
- **BIT5:** RT 接收到的字计数个数错时该位置 1。
- **BIT4:** RT 接收到的数据同步头出错时该位置 1。
- **BIT3:** 当数据传输中有奇偶错、位计数错则该位置 1。
- **BIT2:** RT-RT 同步头/地址错则该位置 1，包括了第二个命令字同步头错，RT 地址错。
- **BIT1:** RT-RT 通讯时第二个命令字错则该位置 1，包括了第二个命令字的位计数错、奇偶校验错。
- **BIT0:** RT-RT 第二个命令字内容出错则该位置 1，它指的是 RT 地址错。

表 22-28 计算机模块中 1553B 模块 BM 块状态字(BM_BLK)

位（BIT）	描述 (DESCRIPTION)
15	消息传输结束标志（EOM）
14	消息传输开始标志（SOM）
13	传输通道指示（CHANNEL B/A*）
12	出错标志(ERROR FLAG)
11	RT-RT 通讯方式标志(RT-RT FORMAT)
10	格式错误标志(FORMAT ERROR)
9	响应超时标志(NO RESPONSE TIMEOUT)

位（BIT）	描述 (DESCRIPTION)
8	数据没有错（GOOD DATA BLOCK TRANSFER）
7	保留
6	保留
5	字计数个数错(WORD COUNT ERROR)
4	数据同步头出错(INCORRECT DATA SYNC)
3	无效字（INVALID WORD）
2	RT-RT 同步头/地址错（RT-RT SYNCH/ADDRESS ERROR）
1	RT-RT 第二个命令字错（RT-RT 2ND COMMAND ERROR）
0	命令字内容错（COMMAND WORD CONTENTS ERROR）

- **BIT15:** 消息传输结束该位置 1。
- **BIT14:** 消息传输开始该位置 1。
- **BIT13:** 消息传输在 A 通道进行为 0，消息传输在 B 通道进行为 1。
- **BIT12:** 有格式错误或响应超时产生该位为 1。
- **BIT11:** 在 RT-RT 通讯时接收 RT 时被设置为 1，发送 RT 时与该位无关。
- **BIT10:** 格式错误包括表中的 BIT5-BIT0 中任何一种错,如果 BIT5-BIT0 中出现其中任何一种错误则该位为 1。
- **BIT9:** RT-RT 通讯时响应超时则该位置 1。
- **BIT8:** 接收到的数据字中没有格式错则该位置 1，有格式错则该位置 0，如是不带数据的方式命令字则该位一直为 0。
- **BIT5:** RT 接收到的字计数个数错时该位置 1。
- **BIT4:** RT 接收到的数据同步头出错时该位置 1。
- **BIT3:** 当数据传输中有奇偶错、位计数错则该位置 1。
- **BIT2:** RT-RT 同步头/地址错则该位置 1，包括了第二个命令字同步头错，RT 地址错。
- **BIT1:** RT-RT 通讯时第二个命令字错则该位置 1，包括了第二个命令字的位计数错、奇偶校验错。
- **BIT0:** RT-RT 第二个命令字内容出错则该位置 1，它指的是 RT 地址错。

22.6 模块工作方式描述

22.6.1 BC 总线控制器工作方式

22.6.1.1 BC 存储器地址分配

表 22-29　计算机模块中 1553B 模块 BC 存储器地址分配(4K 双口 RAM)

地址（HEX）	描述
0000-00FF	堆栈 A（STACK A）
0100	堆栈指针 A (STACK POINTER A)
0101	消息个数设置 A (MESSAGE COUNT A)
0102-0103	保留
0104	堆栈指针 B (STACK POINTER B)
0105	消息个数设置 B (MESSAGE COUNT B)
0106-0107	保留
0108-012D	消息块 0(MESSAGE BLOCK0)
012E-0153	消息块 1(MESSAGE BLOCK1)
…	…
0ED6-0EFB	消息块 93(MESSAGE BLOCK 93)
…	…
0F00-0FFF	堆栈 B（STACK B）

22.6.1.2 BC 存储器管理

BC 存储器管理如图 所示。该图说明了命令堆栈区包含四个描述符，即块状态字，时间标签字，消息间隔时间字和消息块地址字。块状态字包括消息状态、完成、有效性及总线通道信息；时间标签字寄存了当前消息结束时时间标签寄存器的值；消息间隔时间字存储的是设定的消息间隔时间；消息块地址字寄存的是指向消息块第一个字的地址。程序通过 RAM 的 0X0100H 地址取命令堆栈指针，0X0101 地址取消息个数值。

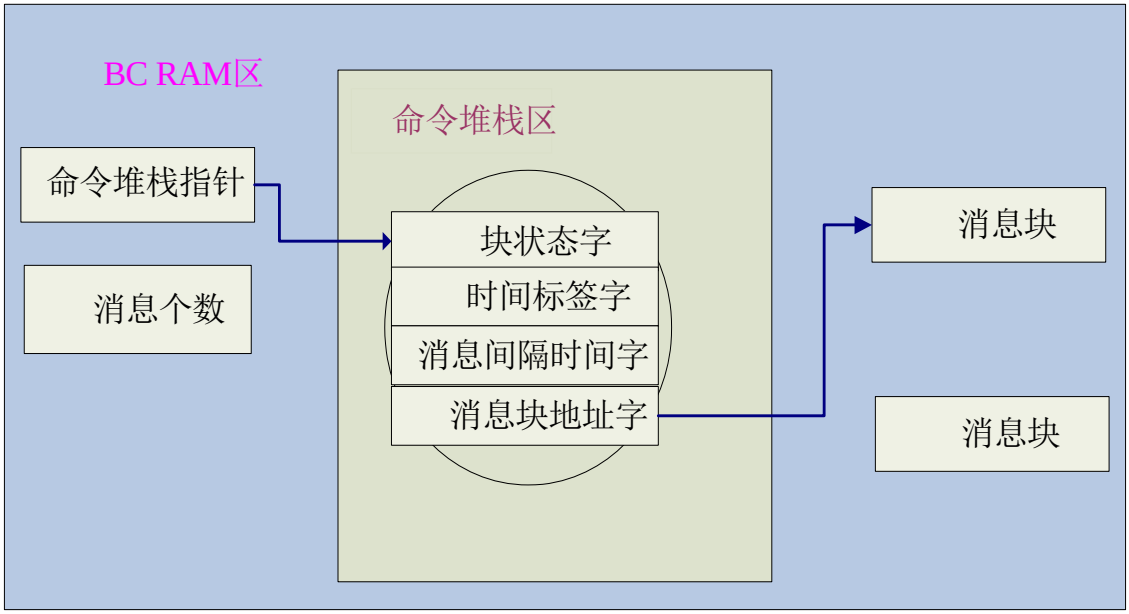


图 22-2　计算机模块中 1553B 模块 BC 存储器管理

22.6.1.3 BC 消息格式

表 22-30　计算机模块中 1553B 模块 BC 消息格式

BC 到 RT 的传输	RT 到 BC 的传输	RT 到 RT 的传输	不带字的方式命令
控制字	控制字	控制字	控制字
接收命令字	发送命令字	接收命令字	方式命令字
数据字 1	发送命令字 的回应字	发送命令字	方式回应字
数据字 2	状态字	发送命令字的回应字	状态字
…	数据字 1	发送终端的状态字	
	数据字 2	数据字 1	
最后一个数据字	…	…	
最后数据字的回应字		最后一个数据字	
状态字	最后一个数据字	接收终端的状态字	

表 22-31　计算机模块中 1553B 模块 BC 消息格式（接上表）

带字的发送方式命令	带字的接收方式命令	广播命令	不带字的广播方式命令	带字的广播方式命令
控制字	控制字	控制字	控制字	控制字
发送方式命令字	接收方式命令字	广播命令字	广播方式命令字	广播方式命令字
方式命令回应字	数据字	数据字 1	广播方式命令字的回应字	数据字
接收状态字	接收命令字的回应字	数据字 2 …		最后数据字的回应字
数据字	接收状态字	最后的数据回应字		

22.6.2 RT 远程终端工作方式

22.6.2.1 RT 存储器地址分配

表 22-32　计算机模块中 1553B 模块 RT 存储器地址分配(4K 双口 RAM)

地址（HEX）	描述
0000-00FF	堆栈 (STACK)
0100	堆栈指针(STACK POINTER)
0101-0107	保留
0108-010F	方式代码选择中断表（MODE CODE SELECTIVE INTERRUPT TABLE）
0110-013F	方式代码数据(MODE CODE DATA)
0140-01BF	查找表(LOOKUP TABLE)
01C0-023F	保留
0240-0247	忙位查找表（BUSY BIT LOOKUP TABLE）
0248-025F	（没有使用）
0260-027F	数据块 0 (DATA BLOCK 0)
0280-02FF	数据块 1-4 (DATA BLOCK 1-4)

地址（HEX）	描述
0300-03FF	非法命令表 (CINNABD UKKEGAKUZUBG TABKE)
0400-041F	数据块 5 (DATA BLOCK 5)
0420-043F	数据块 6 (DATA BLOCK 6)
…	…
0FE0-0FFF	数据块 100 (DATA BLOCK 100)

22.6.2.2 RT 存储器查找表

表 22-33　计算机模块中 1553B 模块 RT 存储器查找表(LOOK_UP TABLE)

地址（HEX）	对应子地址	描述
0140 … 015F	Rx_SA0 … Rx _SA31	接收查找表
0160 … 017F	Tx_SA0 … Tx _SA31	发送查找表
0180 … 19F	Bcst_SA0 … Bcst _SA31	广播查找表
01A0 … 01BF	SACW_SA0 … SACW_SA31	子地址控制字查找表

22.6.2.3 RT 存储器非法命令表地址分配

表 22-34　计算机模块中 1553B 模块 RT 存储器非法命令地址分配表(COMMAND ILLEGALIZING TABLE)

位(BIT)	描述 (DESCRIPTION)
15-10	均为逻辑 0
9-8	均为逻辑 1
7	广播*/本身地址（BROADCAST*/OWN ADDRESS）
6	发送/接收*（T/R*）
5-1	子地址 4－子地址 0（SA4-SA0）
0	子计数 4/方式字 4（WC4/MC4）

RT 非法命令表，在 RT 中占用 0x300～0x3FF 的地址空间。当 RT 接收到命令字后，如果使能(为 1)非法化命令检测。通过广播/RT 地址、T/R*、SA4～SA0 和 WC4/MC4 共 8 位在 0x300～0x3FF 中，查找 WC3～WC0（MC3~MC0）收到的给 RT 某一子地址、某些个数的命令字是否非法。

22.6.2.4 RT 存储器忙位查找表地址分配

表 22-35　计算机模块中 1553B 模块 RT 存储器忙位查找表地址分配表(BUSY BIT LOOKUP TABLE)

位(BIT)	描述 (DESCRIPTION)
15-10	均为逻辑 0
9	逻辑 1
8	逻辑 0
7	逻辑 0
6	逻辑 1
5-3	均为逻辑 0
2	广播/本身地址*（BROADCAST/OWN ADDRESS*）
1	发送/接收*（T/R*）
0	子地址 4（SA4）

RT 忙位查找表，在 RT 中占用 0x240～0x247 的地址空间。当 RT 接收到命令字后，如果使能（为 1）忙位查找表检测。通过广播/RT 地址、T/R*、SA4 共 3 位在 0x240～0x247 中，查找 SA3～SA0 收到的给 RT 某一子地址的命令字是否忙。

22.6.2.5 RT 存储器方式代码选择中断表

表 22-36　计算机模块中 1553B 模块 RT 存储器方式代码选择中断表

地址（HEX）	描述 (DESCRIPTION)
0108	接收方式命令 0-15(undefined)
0109	发送方式命令 16-31(WITH DATA)
010A	发送方式命令 0-15(WITHOUT DATA)
010B	发送方式命令 16-31(WITH DATA)
010C	广播接收方式命令 0-15(undefined)
010D	广播接收方式命令 16-31(WITH DATA)
010E	广播发送方式命令 0-15(WITHOUT DATA)
010F	广播发送方式命令 16-31(UNDEFINED)

22.6.2.6 RT 存储器方式代码选择中断地址分配

表 22-37　计算机模块中 1553B 模块 RT 存储器方式代码选择中断表地址分配表

位(BIT)	描述 (DESCRIPTION)
15-9	均为逻辑 0
8	逻辑 1
7	逻辑 0
6	逻辑 0
5	逻辑 0
4	逻辑 0
3	逻辑 1
2	广播/本身地址*（BROADCAST/OWN ADDRESS*）
1	发送/接收*（T/R*）
0	方式代码 4（MC4）

RT 方式代码选择中断表，在 RT 中占用 0x108～0x10F 的地址空间。当 RT 接收到命令字

后，如果使能（为 1）式代码选择中断表检测。通过广播/RT 地址、T/R*、MC4 共 3 位在 0x108～0x10F 中，查找 MC3～MC0 收到的给 RT 某一方式代码是否有中断。

22.6.2.7 RT 方式代码数据表

表 22-38　计算机模块中 1553B 模块 RT 存储器方式代码数据表

地址（HEX）	方式代码（MODE CODE）
0110	没有定义
0111	同步带数据
0112-11F	没有定义
0120	发送矢量字
0121-13F	没有定义

22.6.2.8 已实现的方式代码

表 22-39　计算机模块中 1553B 模块已实现的方式代码

发/收*	方式代码	功能	是否带数据字	是否允许广播指令
1	00000	动态总线控制	否	否
1	00001	同步	否	是
1	00010	发送上一状态字	否	否
1	00011	启动自测试[1]	否	是
1	00100	发送器关闭	否	是
1	00101	取消发送器关闭	否	是
1	00110	禁止终端标志位	否	是
1	00111	取消禁止终端标志位	否	是
1	01000	复位远程终端[2]	否	是
1	01001	备用	否	待定
…	…	…	…	…

发/收*	方式代码	功能	是否带数据字	是否允许广播指令
1	01111	备用	否	待定
1	10000	发送矢量字	是	否
0	10001	同步[3]	是	是
1	10010	发送上一指令字	是	否
1	10011	发送自检测字	是	否

备注:

【1】启动自测试执行的操作是:

当RT接受到该命令后,软件进行相应的测试，并判断RT BIT字如果没有异常则认为自测试OK。

【2】复位远程终端执行的操作是:

1) 对RT状态字将清BIT10,BIT4这两位;

2) 对RT位字寄存器将清BIT15, BIT11, BIT10, BIT9;

3) 如果使用了方式命令 00110(禁止终端标志位)或 00100（发送器关闭）哪么在执行完复位远程终端这一方式命令后，就将方式命令00110,00100的结果予以清除;

【3】带数据的同步执行的操作是:

TAG REG0,接收最新的重载值, TAG REG1 TAG REG2被清为0。

【4】不带数据的同步执行的操作是:

TAG REG0, TAG REG1 TAG REG2均被清为0。

【5】当接收到的方式字是发送上一命令字或发送状态字时RT的状态字不会被改变。

【6】当接收到的方式字是发送自检测字时RT的位状态字不会被改变。

22.6.2.9 RT 单缓冲存储器管理

RT 单缓冲存储器管理如图 所示。该图说明了命令堆栈区包含四个描述符，即块状态字，时间标签字，数据块指针字和接收命令字。块状态字包括消息状态、完成、有效性及总线通道信息；时间标签字寄存了当前消息结束时时间标签寄存器的值；数据块指针字存储指向数据块的起始地址；接收命令字存储 RT 接收到的命令字。程序通过 RAM 的 0X0100H 地址取命令堆栈指针。

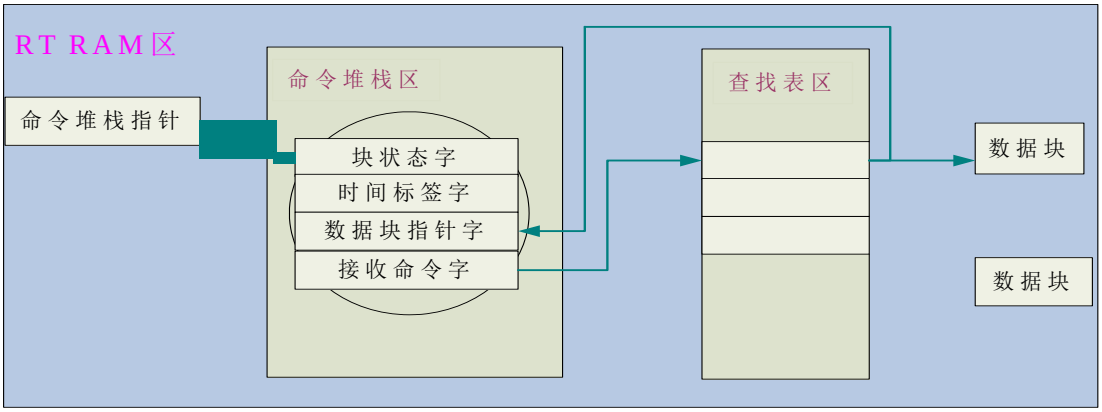


图 22-3 计算机模块中 1553B 模块 RT 单缓冲存储器管理

22.6.2.10 RT 循环缓冲存储器管理

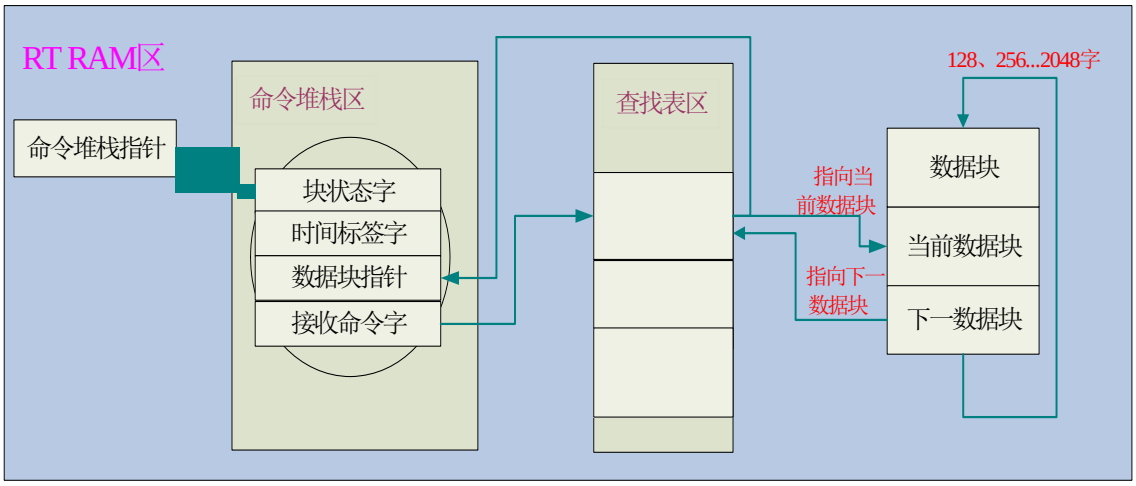


图 22-4 计算机模块中 1553B 模块 RT 循环缓冲存储器管理

22.6.2.11 RT 双缓冲存储器管理

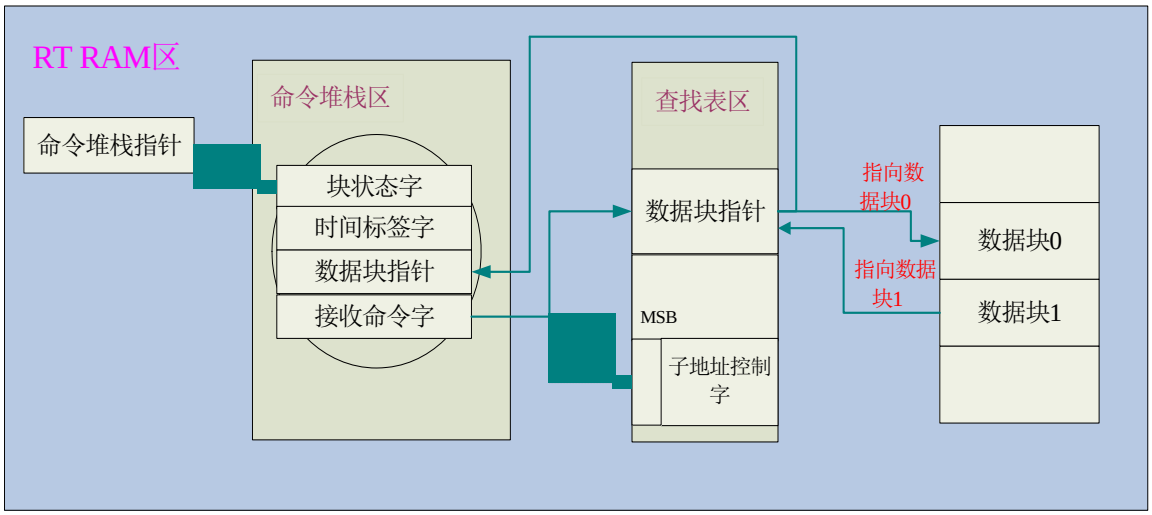


图 22-5 计算机模块中 1553B 模块 RT 双缓冲存储器管理

22.6.3 BM 总线监视器工作方式

22.6.3.1 BM 存储器地址分配

表 22-40 计算机模块中 1553B 模块 BM 存储器地址分配

地址（HEX）	描述
0000-027F	命令堆栈区域 (STACK AREA)
0280-02FF	子地址选择设置区域(SUBADDRESS SELECT AREA)
0300-0FFF	数据块区域(DATA BLOCK AREA)

22.6.3.2 BM 存储器管理

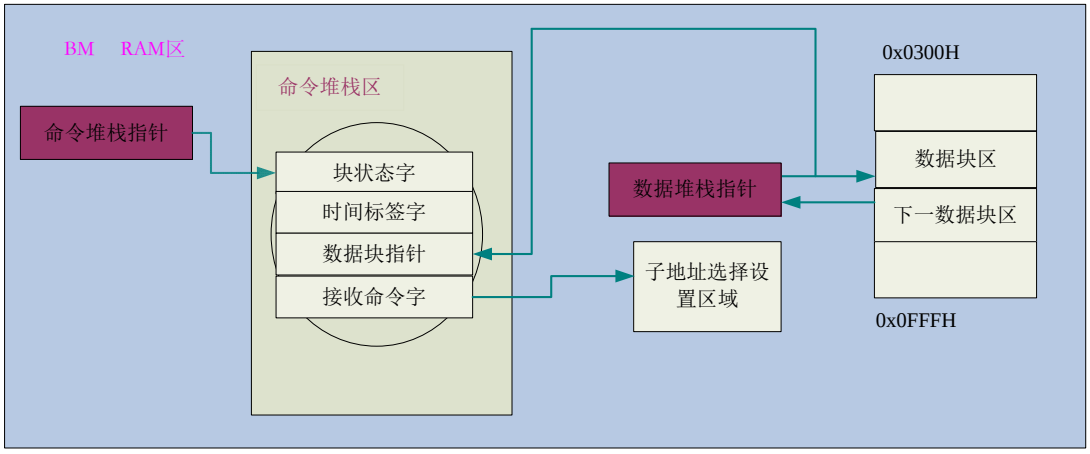


图 22-6 计算机模块中 1553B 模块 BM 存储器管理

22.6.3.3 BM 子地址选择设置区地址分配

表 22-41 计算机模块中 1553B 模块 BM 子地址分配表

位 (BIT)	描述 (DESCRIPTION)
15-11	逻辑 0
10	逻辑 0

位 （BIT）	描述 (DESCRIPTION)
9	逻辑 1
8	逻辑 0
7	逻辑 1
6	RT 地址 4（RT ADDRESS 4）
5	RT 地址 3（RT ADDRESS 3）
4	RT 地址 2（RT ADDRESS 2）
3	RT 地址 1（RT ADDRESS 1）
2	RT 地址 0（RT ADDRESS 0）
1	发送/接收*（T/R*）
0	子地址 4（SA4）

BM 子地址选择设置区地址分配表，在 BM 中占用 0x280～0x2FF 的地址空间。当 BM 接收到命令字后，如果使能（为 1）子地址选择设置检测。通过 RT 地址、T/R*、SA4 共 7 位在 0x280～0x2FF 中，查找 SA3～SA0 收到的给 BM 某一子地址的内容接收，否则不做接收。

22.7 时序图

发送波形如图 所示，接收波形如图 所示，数据频率编码关系如图 所示：

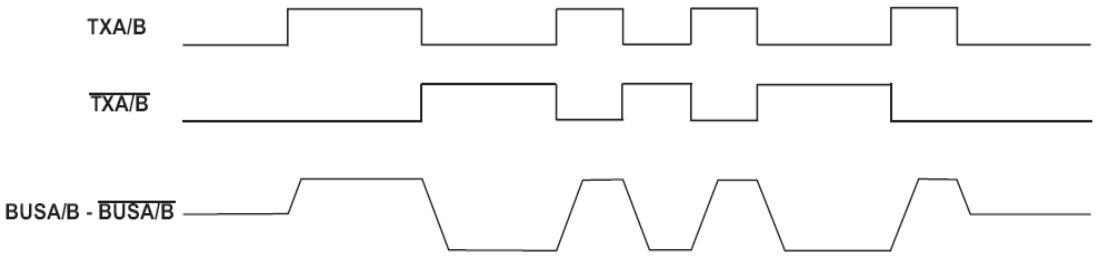


图 22-7 计算机模块中 1553B 模块发送波形

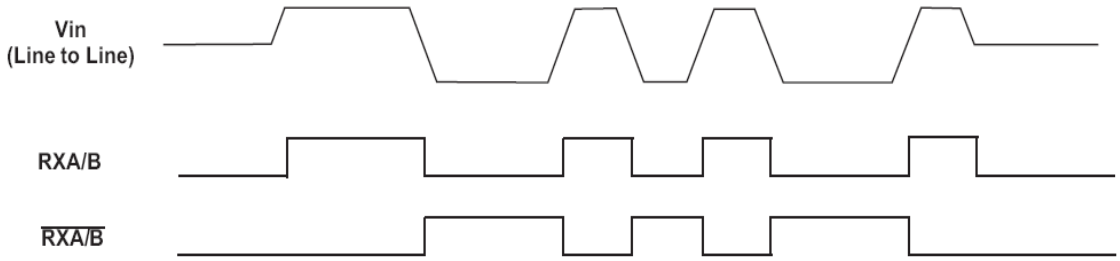


图 22-8 计算机模块中 1553B 模块接收波形

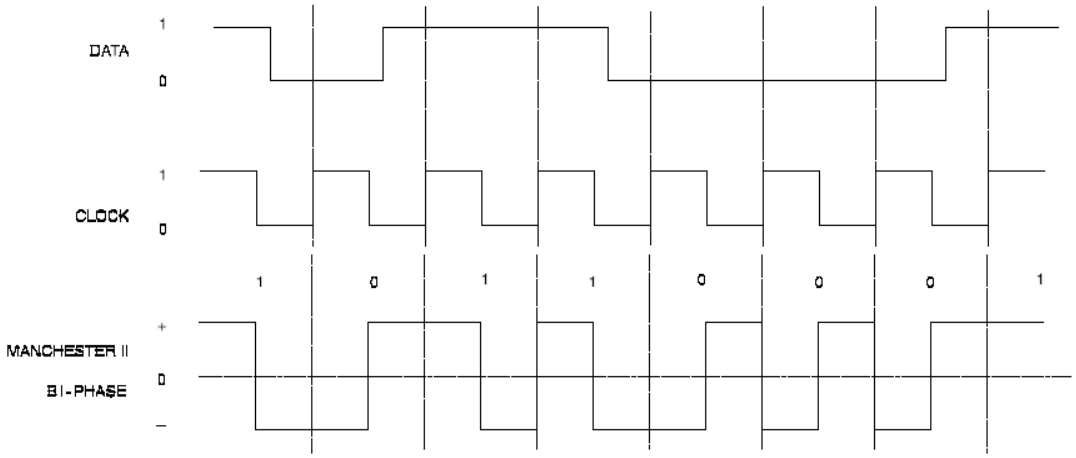


图 22-9 计算机模块中 1553B 模块数据频率编码之间的关系

22.8 应用说明

22.8.1 1Mbps 外围接口

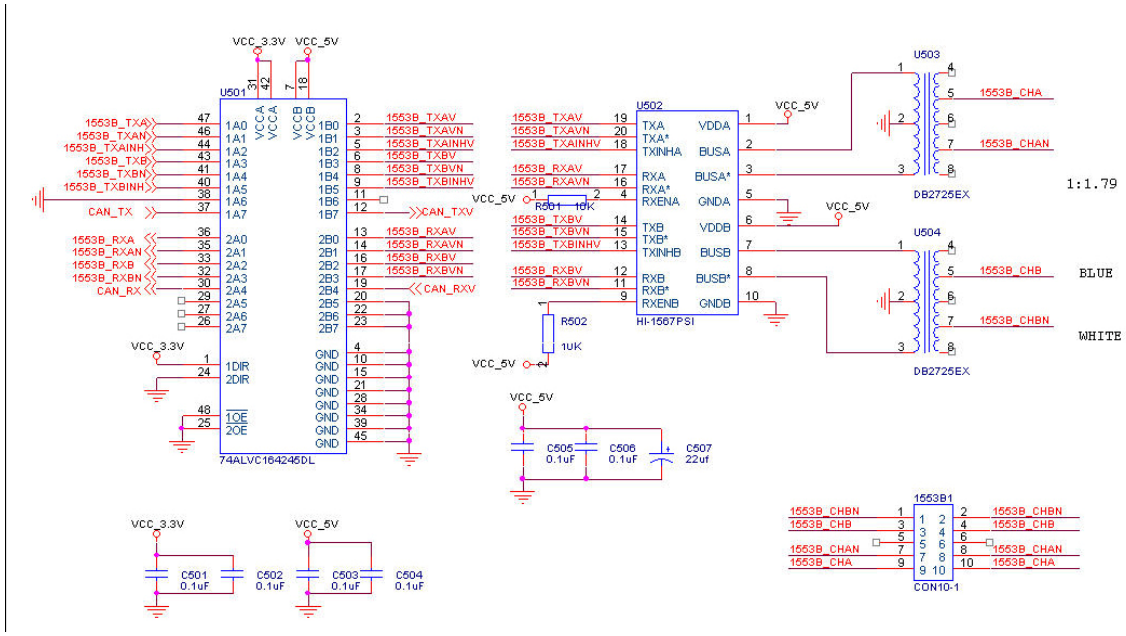


图 22-10　计算机模块中 1553B 模块 1M 外围接口原理图

计算机模块中 1553B 模块 1M 外围接口原理图主要说明 1553 总线外部接线关系。由于计算机模块输出电平为 CMOS 电平，而 1553B 电平驱动芯片 HI-1567 芯片为 TTL 电平，中间需要通过一个电平转换芯片如 74ALVC164245DL 芯片进行电平转换。HI-1567 芯片在连接 1553B 变压器。

22.8.2 10Mbps 外围接口

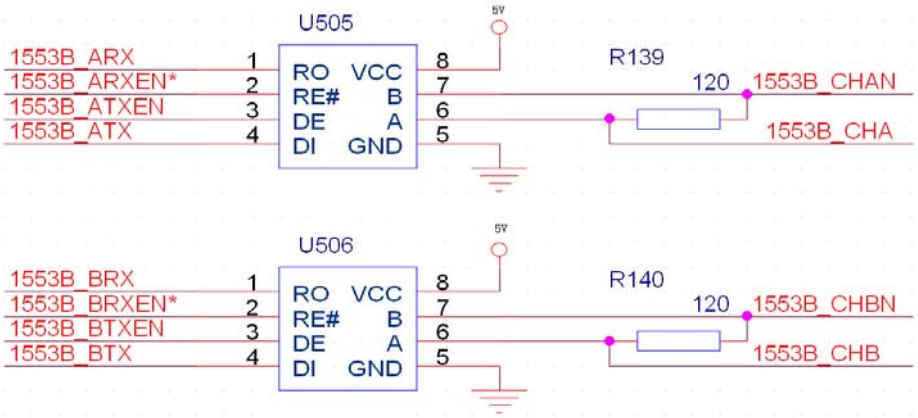


图 22-11　计算机模块中 1553B 模块 10M 接口原理图

计算机模块中 1553B 模块 10M 接口原理图主要说明 1553-10M 总线外部接线关系。通过外接 RS485 芯片（图中的 U505，U506），进行差分信号传输。1553-10M 对外接口不接变压器。

22.8.3 BC 总线控制器应用案例

对于 BC 编程，首先要初始化相应的寄存器以及堆栈指针、消息计数器；然后定义消息的控制字、命令字等；最后启动 BC。需要注意的是 BC 控制字不会在 1553 总线上传输。BC 的消息格式通过编程 BC 控制字的最低 3 位来控制。

BC 消息帧可以通过查询和中断来进行处理。如果采用查询模式，那么可以查询配置寄存器 1、中断状态寄存器、堆栈指针和消息计数器寄存器。另外，每一条 BC 消息结束后堆栈指针加 4。在嵌入式系统软件处理中，我们应尽量采用中断方式。

流程图如图 22-12 所示。程序把 OBT1553 设置成 BC 总线控制器类型，并且设置消息类型为 BC-RT，RT-BC；一帧两条消息，第一条消息为 BC 往地址为 0，子地址为 5 发送 32 个数据；第二条消息为地址 0，子地址 4 往 BC 发送 32 个数据。

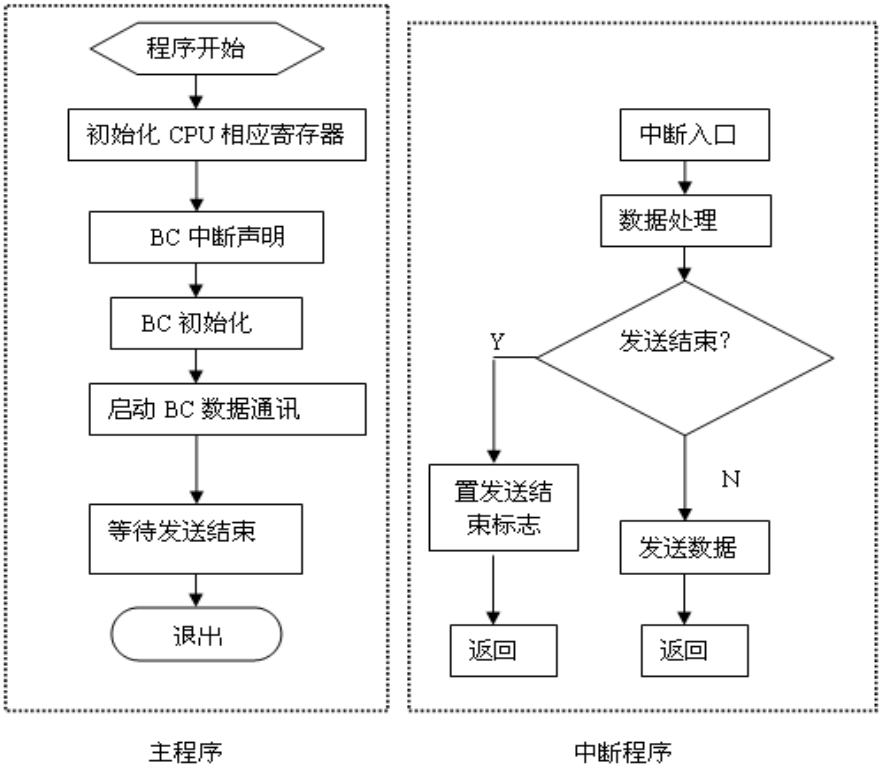


图 22-12　计算机模块中 1553B 模块 BC 程序流程图

22.8.4 RT 远程终端应用案例

对于 RT 编程，首先初始化相应的寄存器；然后设置非法区、初始化相应子地址的查询表

及子地址控制字；最后设置配置寄存器 1 使设备处于 RT 模式。此后该设备就处于在线，只要 BC 发送一条消息命令与该设备相关，那么该设备就会做出反映。处理 RT 消息时，这里也有四个字的块描述符，即块状态字、时间标志字、数据块起始地址指针和接收到的 16 位命令字。与 BC 模式一样，要读取接收到的消息，我们应该首先从堆栈指针中读取当前消息的堆栈指针，来分别读出块状态字、时间标志字、上一条消息的块地址和命令字。

流程图如图 22-13 所示。程序把 OBT1553 设置成 RT 远程终端类型，并且设置消息类型为 BC-RT；BC 往 RT 地址 0，子地址为 1，MC32 发送消息，消息的数据长度为 1 个，数据为互为反码的一组数：0x0000 和 0xffff。

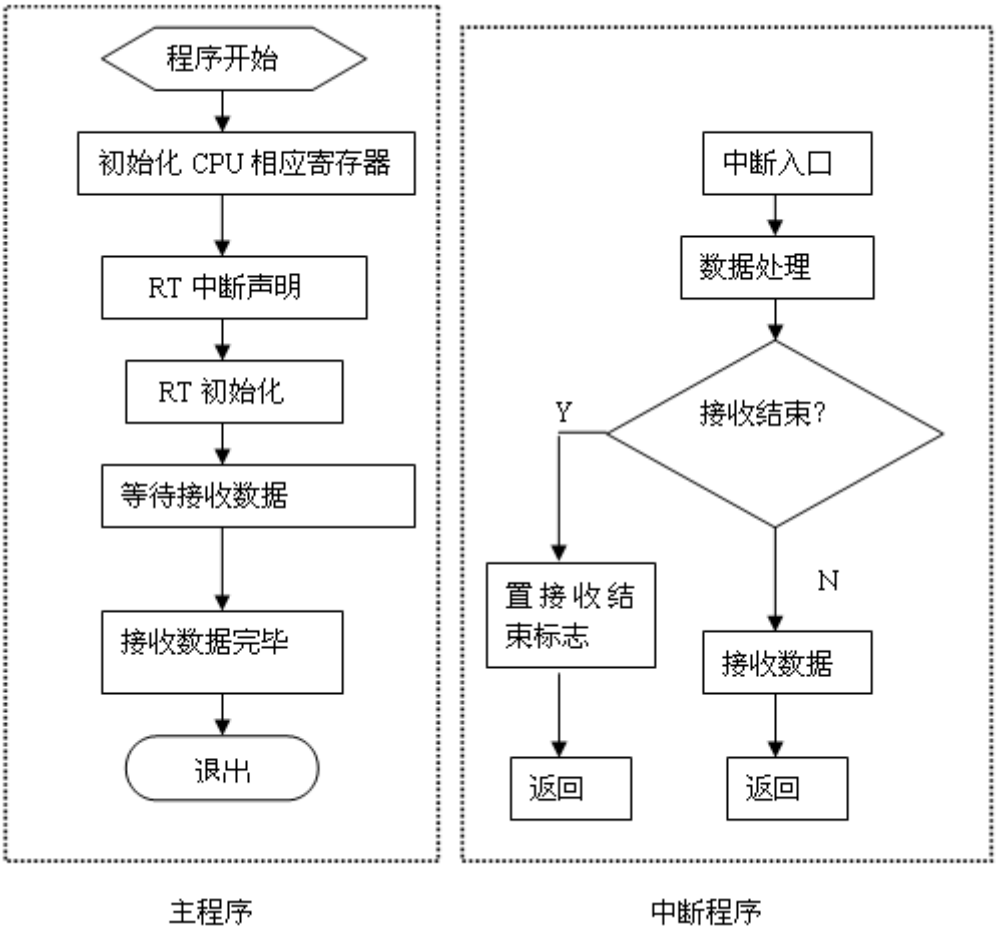


图 22-13　计算机模块中 1553B 模块 RT 程序流程图

22.8.5 BM 总线监视器应用案例

对于 BM 编程，首先初始化相应的寄存器；然后设置子地址选择设置区；最后设置启动/复位寄存器启动 BM。此后该设备就处于在线。接收到的数据里也有四个字的块描述符，即块状态字、时间标志字、数据块起始地址指针和接收到的 16 位命令字。与 RT 模式一样，要读

取接收到的消息，我们应该首先从堆栈指针中读取当前消息的堆栈指针，来分别读出块状态字、时间标志字、上一条消息的块地址和命令字；从数据堆栈指针中读取接收到的数据。

流程图如图22-14所示。程序把OBT1553B设置成BM类型，接收消息类型BC→RT0 SA1 MC32，判断BM接收的数据是否是BC发送的数据。

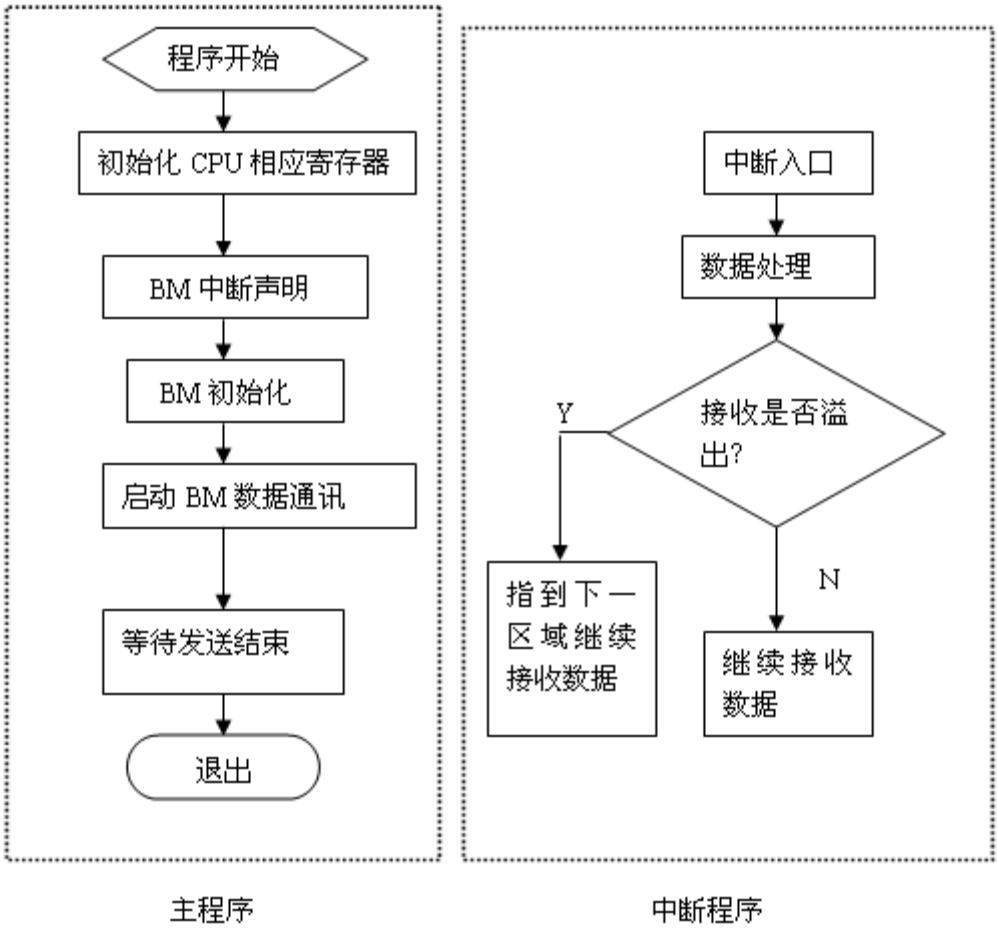


图 22-14　计算机模块中 1553B 模块 BM 程序流程图

23. 引脚分配

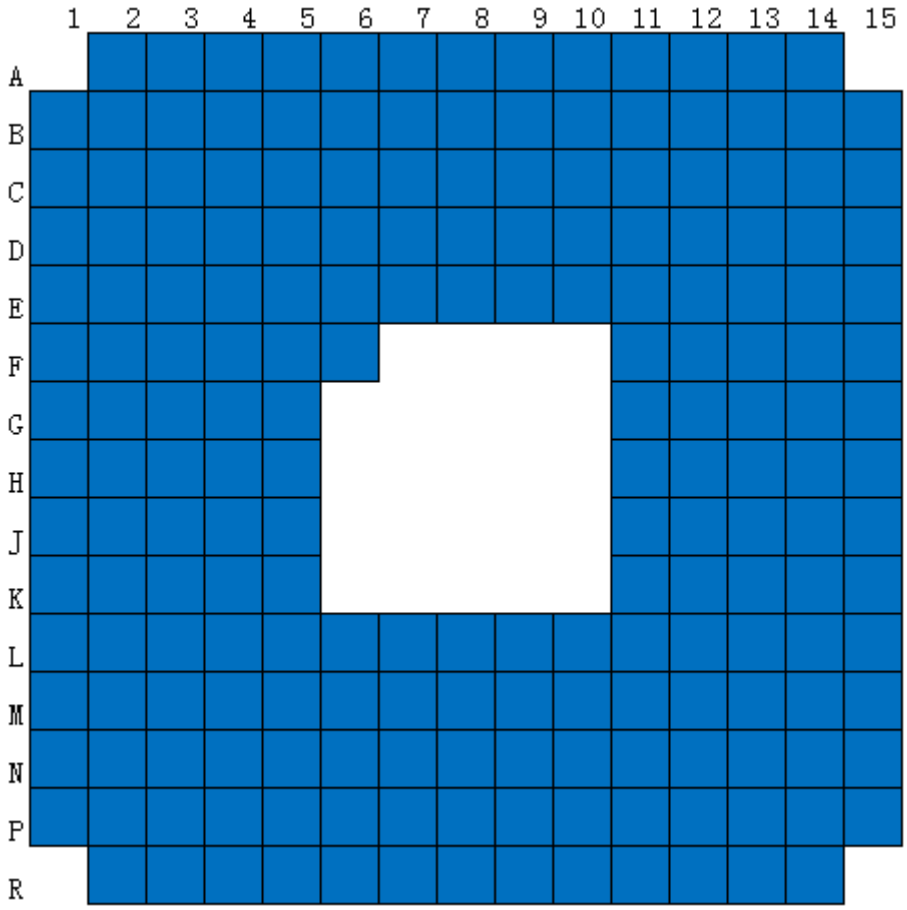


图 23-1 计算机模块引脚示意图

表 23-1 计算机模块引脚说明

序号	引脚号	引脚名称	方向	信号说明	有效电平
1	L5	ADDRESS[0]	OUT	ram&rom 的地址信号	-
2	P4	ADDRESS[1]	OUT	ram&rom 的地址信号	-
3	C7	ADDRESS[2]	OUT	ram&rom 的地址信号	-
4	M5	ADDRESS[3]	OUT	ram&rom 的地址信号	-
5	N5	ADDRESS[4]	OUT	ram&rom 的地址信号	-
6	L4	ADDRESS[5]	OUT	ram&rom 的地址信号	-
7	R4	ADDRESS[6]	OUT	ram&rom 的地址信号	-
8	P5	ADDRESS[7]	OUT	ram&rom 的地址信号	-

序号	引脚号	引脚名称		方向	信号说明	有效电平
9	L6	ADDRESS[8]		OUT	ram&rom 的地址信号	-
10	P2	ADDRESS[9]		OUT	ram&rom 的地址信号	-
11	B7	ADDRESS[10]		OUT	ram&rom 的地址信号	-
12	R5	ADDRESS[11]		OUT	ram&rom 的地址信号	-
13	A7	ADDRESS[12]		OUT	ram&rom 的地址信号	-
14	D10	ADDRESS[13]		OUT	ram&rom 的地址信号	-
15	P6	ADDRESS[14]		OUT	ram&rom 的地址信号	-
16	M6	ADDRESS[15]		OUT	ram&rom 的地址信号	-
17	B8	ADDRESS[16]		OUT	ram&rom 的地址信号	-
18	A8	SP[10]**	PROMEDAC_EN	IN	PROM EDAC 使能信号	-
			ADDRESS[17]	OUT	ram&rom 的地址信号	-
19	R6	SP[9]**	PROMWIDTH[1]	IN	PROM 数据位宽选择信号 bit[0]	-
			ADDRESS[18]	OUT	ram&rom 的地址信号	-
20	C8	SP[8]**	PROMWIDTH[0]_	IN	PROM 数据位宽选择信号 bit[1]	-
			ADDRESS[19]	OUT	ram&rom 的地址信号	-
21	M7	SP[7]**	PLLDIV[0]	IN	PLL1 分频参数配置 bit[0]	-
			ADDRESS[20]	OUT	ram&rom 的地址信号	-
22	N6	SP[6]**	PLLDIV[1]	IN	PLL1 分频参数配置 bit[1]	-
			ADDRESS[21]	OUT	ram&rom 的地址信号	
23	L7	SP[5]**	PLLMLT[0]	IN	PLL1 倍频参数配置 bit[0]	-
			ADDRESS[22]	OUT	ram&rom 的地址信号	-
24	C9	SP[4]**	PLLMLT[1]	IN	PLL1 倍频参数配置 bit[1]	-
			ADDRESS[23]	OUT	ram&rom 的地址信号	-
25	B9	SP[3]**	PLLMLT[2]	IN	PLL1 倍频参数配置 bit[2]	-
			ADDRESS[24]	OUT	ram&rom 的地址信号	-

序号	引脚号	引脚名称		方向	信号说明	有效电平
26	N7	SP[2]**	PLLMLT[3]	IN	PLL1 倍频参数配置 bit[3]	-
			ADDRESS[25]	OUT	ram&rom 的地址信号	-
27	P7	SP[1]**	PLLBYPS2	IN	PLL2 旁路使能信号	H
			ADDRESS[26]	OUT	ram&rom 的地址信号	-
28	L8	SP[0]**	PLLBYPS1	IN	PLL1 旁路使能信号	H
			ADDRESS[27]	OUT	ram&rom 的地址信号	-
29	R2	WRITEN		OUT	外部存储器(flash、sram)读写控制信号	L
30	L2	OEN		OUT	外部存储器输出使能信号	L
31	R3	RAMSN[0]		OUT	ram 片选信号 0	L
32	N3	ROMSN[0]		OUT	rom 片选信号 0	L
33	H14	CB[0]		IN/OUT	SRAM 校验数据	
34	K13	CB[1]		IN/OUT	SRAM 校验数据	-
35	J12	CB[2]		IN/OUT	SRAM 校验数据	-
36	J15	CB[3]		IN/OUT	SRAM 校验数据	-
37	H12	CB[4]		IN/OUT	SRAM 校验数据	-
38	J14	CB[5]		IN/OUT	SRAM 校验数据	-
39	H11	CB[6]		IN/OUT	SRAM 校验数据	-
40	H13	CB[7]		IN/OUT	SRAM 校验数据	-
41	R8	DATA[0]		IN/OUT	ram&rom 的数据信号	-
42	M9	DATA[1]		IN/OUT	ram&rom 的数据信号	-
43	A10	DATA[2]		IN/OUT	ram&rom 的数据信号	-
44	B10	DATA[3]		IN/OUT	ram&rom 的数据信号	-
45	N9	DATA[4]		IN/OUT	ram&rom 的数据信号	-
46	P9	DATA[5]		IN/OUT	ram&rom 的数据信号	-
47	L10	DATA[6]		IN/OUT	ram&rom 的数据信号	-
48	R9	DATA[7]		IN/OUT	ram&rom 的数据信号	-

序号	引脚号	引脚名称	方向	信号说明	有效电平
49	D11	DATA[8]	IN/OUT	ram&rom 的数据信号	-
50	M10	DATA[9]	IN/OUT	ram&rom 的数据信号	-
51	N10	DATA[10]	IN/OUT	ram&rom 的数据信号	-
52	P10	DATA[11]	IN/OUT	ram&rom 的数据信号	-
53	R10	DATA[12]	IN/OUT	ram&rom 的数据信号	-
54	L11	DATA[13]	IN/OUT	ram&rom 的数据信号	-
55	A11	DATA[14]	IN/OUT	ram&rom 的数据信号	-
56	M11	DATA[15]	IN/OUT	ram&rom 的数据信号	-
57	B11	DATA[16]	IN/OUT	ram&rom 的数据信号	-
58	N11	DATA[17]	IN/OUT	ram&rom 的数据信号	-
59	P11	DATA[18]	IN/OUT	ram&rom 的数据信号	-
60	R11	DATA[19]	IN/OUT	ram&rom 的数据信号	-
61	L12	DATA[20]	IN/OUT	ram&rom 的数据信号	-
62	D12	DATA[21]	IN/OUT	ram&rom 的数据信号	-
63	M12	DATA[22]	IN/OUT	ram&rom 的数据信号	-
64	N12	DATA[23]	IN/OUT	ram&rom 的数据信号	-
65	M13	DATA[24]	IN/OUT	ram&rom 的数据信号	-
66	P12	DATA[25]	IN/OUT	ram&rom 的数据信号	-
67	L13	DATA[26]	IN/OUT	ram&rom 的数据信号	-
68	D13	DATA[27]	IN/OUT	ram&rom 的数据信号	-
69	R12	DATA[28]	IN/OUT	ram&rom 的数据信号	-
70	P13	DATA[29]	IN/OUT	ram&rom 的数据信号	-
71	A14	DATA[30]	IN/OUT	ram&rom 的数据信号	-
72	N13	DATA[31]	IN/OUT	ram&rom 的数据信号	-
73	L9	SPW_CLKIN	IN	spacewire 时钟	
74	R7	SYSCLK_OUT	OUT	系统时钟输出	
75	L1	RESETN	IN	外部复位输入信号	L

序号	引脚号	引脚名称	方向	信号说明	有效电平
				注：欲将系统可靠复位，需给该引脚一个宽度不低于 3 个外部时钟周期的低电平，否则，将无法保证系统被可靠复位。	
76	L3	RESETO	OUT	内部复位信号	-
77	P1	ERRORN	OUT	内部处理错误状态信号	L
78	R14	DSUTX	OUT	DSU 数据发送信号	
79	G15	DSURX	IN	DSU 数据接收信号	
80	R13	DSUEN	IN	DSU 使能信号,Debug 模式下为高，Normal 模式下为低	H
81	N15	DSUBRE	IN	DSU Break 输入信号： 当此信号由 0 变为 1 时，将使得处理器进入调试模式； 平时情况下，需将此信号需保持低电平；	H
82	P15	DSUACT	OUT	DSU 有效状态	H
83	N14	JTAG_TCK	IN	JTAG 扫描时钟	-
84	M14	JTAG_TDI	IN	JTAG 扫描数据输入	-
85	H15	JTAG_TDO	OUT	JTAG 扫描数据输出	-
86	P14	JTAG_TMS	IN	JTAG 扫描控制信号	H
87	D1	NC			-
88	E2	BUSB_N	INOUT	1553B-0 通道 B 总线信号	-
89	F3	BUSA_N	INOUT	1553B-0 通道 A 总线信号	-
90	F2	BUSA	INOUT	1553B-0 通道 A 总线信号	-
91	C3	NC			-

序号	引脚号	引脚名称		方向	信号说明	有效电平
92	E1	BUSB		INOUT	1553B-0 通道 B 总线信号	-
93	F1	ROMSN1		OUT	ROM1 片选信号	
94	G5	RAMSN1		OUT	RAM1 片选信号	-
95	G4	FLASH_CS		IN	内部 Nor flash 片选控制信号	L
96	B3	NC				
97	N8	SP[11] **	PLLMLT2[3]	IN	PLL2 倍频参数配置 bit[3]	-
			GPIO[63]	INOUT	通用输入/输出口 63	-
98	P8	SP[12] **	PLLMLT2[2]	IN	PLL2 倍频参数配置 bit[2]	-
			GPIO[62]	INOUT	通用输入/输出口 62	-
99	M8	SP[13] **	PLLMLT2[1]	IN	PLL2 倍频参数配置 bit[1]	-
			GPIO[61]	INOUT	通用输入/输出口 61	-
100	A9	SP[14] **	PLLMLT2[0]	IN	PLL2 倍频参数配置 bit[0]	-
			GPIO[60]	INOUT	通用输入/输出口 60	-
101	A5	WP#		IN	内部 nor flash 写保护信号，内部已接 1K 上拉电阻	L
102	D4	SPW_RXD_N[0]		IN	spacewire0 DS 编码中接收数据差分信号	
103	B2	SPW_RXD_N[1]		IN	spacewire1 DS 编码中接收数据差分信号	
104	D5	SPW_RXD_P[0]		IN	spacewire0 DS 编码中接收数据差分信号	
105	A3	SPW_RXD_P[1]		IN	spacewire1 DS 编码中接收数据差分信号	

序号	引脚号	引脚名称		方向	信号说明	有效电平
106	D6	SPW_RXS_N[0]		IN	spacewire0 DS 编码中接收选通差分信号	
107	A2	SPW_RXS_N[1]		IN	spacewire1 DS 编码中接收选通差分信号	
108	C1	SPW_RXS_P[0]		IN	spacewire0 DS 编码中接收选通差分信号	
109	C4	SPW_RXS_P[1]		IN	spacewire1 DS 编码中接收选通差分信号	
110	C2	SPW_TXD_N[0]		OUT	spacewire0 DS 编码中发送数据差分信号	
111	D3	SPW_TXD_N[1]		OUT	spacewire1 DS 编码中发送数据差分信号	
112	B4	SPW_TXD_P[0]		OUT	spacewire0 DS 编码中发送数据差分信号	
113	E5	SPW_TXD_P[1]		OUT	spacewire1 DS 编码中发送数据差分信号	
114	D2	SPW_TXS_N[0]		OUT	spacewire0 DS 编码中发送选通差分信号	
115	E6	SPW_TXS_N[1]		OUT	spacewire1 DS 编码中发送选通差分信号	
116	E4	SPW_TXS_P[0]		OUT	spacewire0 DS 编码中发送选通差分信号	
117	B1	SPW_TXS_P[1]		OUT	spacewire1 DS 编码中发送选通差分信号	
118	N1	SP[39] *	SPW_RXD[2]	IN	SpaceWire-2 接收数据信号	
			GPIO[35]	IN/OUT	通用输入/输出口 35	
119	M1	SP[40] *	SPW_RXD[1]	IN	SpaceWire-3 接收数据选	

序号	引脚号	引脚名称		方向	信号说明	有效电平
					通信号	
			GPIO[31]	IN/OUT	通用输入/输出口 34	
120	N2	SP[41] *	SPW_TXD[2]	OUT	SpaceWire-2 发送数据信号	
			GPIO[33]	IN/OUT	通用输入/输出口 33	
121	B6	SP[42] *	SPW_TXD[1]	OUT	SpaceWire-3 发送选通信号	
			GPIO[29]	IN/OUT	通用输入/输出口 32	
122	K3	SP[43]*	SPW_TXS[2]	IN	SpaceWire-2 接收数据信号	
			GPIO[32]	IN/OUT	通用输入/输出口 31	
123	M4	SP[44]*	SPW_TXS[1]	IN	SpaceWire-3 发送选通信号 通信号	
			GPIO[28]	IN/OUT	通用输入/输出口 30	
124	M3	SP[45]*	SPW_RXS[2]	OUT	SpaceWire-2 接收数据信号	
			GPIO[34]	IN/OUT	通用输入/输出口 29	
125	M2	SP[46]*	SPW_RXS[1]	OUT	SpaceWire-3 接收数据信号	
			GPIO[30]	IN/OUT	通用输入/输出口 28	
126	E3	CAN_RX[0]		IN	CAN0 接收信号	-
127	F4	CAN_TX[0]		OUT	CAN0 发送信号	-
128	B5	SP[32]*	CAN_RX[1]	IN	CAN-1 接收	-
			GPIO[42]	IN/OUT	通用输入/输出口 42	-
129	C6	SP[33]*	CAN_TX[1]	OUT	CAN-1 发送	-
			GPIO[41]	IN/OUT	通用输入/输出口 41	-
130	L14	UART_RXD[0]		IN	UART 数据接收	-
131	L15	UART_RXD[1]		IN	UART 数据接收	-
132	K14	UART_TXD[0]		OUT	UART 数据发送	-
133	J13	UART_TXD[1]		OUT	UART 数据发送	-
134	J3	SP[34]*	I2C_CLK	IN/OUT	I2C 时钟	-
			GPIO[40]	IN/OUT	通用输入/输出口 40	-

序号	引脚号	引脚名称		方向	信号说明	有效电平
135	K1	SP[35]*	I2C_DATA	IN/OUT	I2C 数据	-
			GPIO[39]	IN/OUT	通用输入/输出口 39	-
136	J5	SP[36]*	SPI_CLK	OUT	SPI 时钟	-
			GPIO[38]	IN/OUT	通用输入/输出口 38	-
137	J2	SP[37]*	SPI_MOSI	OUT	SPI 主出从入	-
			GPIO[37]	IN/OUT	通用输入/输出口 37	-
138	C5	SP[38]*	SPI_MISO	IN	SPI 主入从出	-
			GPIO[36]	IN/OUT	通用输入/输出口 36	-
139	A4	GPIO[53]		IN/OUT	通用输入/输出口 53	
140	J1	GPIO[52]		IN/OUT	通用输入/输出口 52	
141	J4	GPIO[51]		IN/OUT	通用输入/输出口 51	
142	H5	GPIO[50]		IN/OUT	通用输入/输出口 50	
143	H4	GPIO[49]		IN/OUT	通用输入/输出口 49	
144	H3	GPIO[48]		IN/OUT	通用输入/输出口 48	
145	H2	GPIO[47]		IN/OUT	通用输入/输出口 47	
146	H1	GPIO[46]		IN/OUT	通用输入/输出口 46	
147	G1	GPIO[45]		IN/OUT	通用输入/输出口 45	
148	G2	GPIO[44]		IN/OUT	通用输入/输出口 44	
149	G3	GPIO[43]		IN/OUT	通用输入/输出口 43	
150	A6	WDOGN		OUT	看门狗信号	L
151	N4	IOSN		OUT	IO 区域片选信号	L
152	P3	READ		OUT	外部存储器读状态信号	H
153	K2	GPIO[7]		IN/OUT	通用输入/输出口 7	-
154	K4	GPIO[6]		IN/OUT	通用输入/输出口 6	-
155	K5	GPIO[5]		IN/OUT	通用输入/输出口 5	-
156	K15	GPIO[4]		IN/OUT	通用 IO 接口	-

序号	引脚号	引脚名称	方向	信号说明	有效电平
157	K12	GPIO[3]	IN/OUT	通用 IO 接口	-
158	K11	GPIO[2]	IN/OUT	通用 IO 接口	-
159	J11	GPIO[1]	IN/OUT	通用 IO 接口	-
160	M15	GPIO[0]	IN/OUT	通用 IO 接口	-
161	E15	VDD_5V0	IN	输入电源 5.0V	
162	F14	VDD_5V0	IN	输入电源 5.0V	
163	F15	VDD_5V0	IN	输入电源 5.0V	
164	E14	VDD_5V0	IN	输入电源 5.0V	
165	D7	VDD_3V3	IN	输入电源 3.3V	
166	D8	VDD_3V3	IN	输入电源 3.3V	
167	E7	VDD_3V3	IN	输入电源 3.3V	
168	E8	VDD_3V3	IN	输入电源 3.3V	
169	F5	VDD_2V5	IN	输入电源 2.5V	
170	F6	VDD_2V5	IN	输入电源 2.5V	
171	C15	VDD_1V8	IN	输入电源 1.8V	
172	B15	VDD_1V8	IN	输入电源 1.8V	
173	D15	VDD_1V8	IN	输入电源 1.8V	
174	A13	VDD_1V0	IN	输入电源 1.0V	
175	A12	VDD_1V0	IN	输入电源 1.0V	
176	C13	VDD_1V0	IN	输入电源 1.0V	
177	B12	VDD_1V0	IN	输入电源 1.0V	
178	C10	VDD_1V0	IN	输入电源 1.0V	
179	C12	VDD_1V0	IN	输入电源 1.0V	
180	C11	VDD_1V0	IN	输入电源 1.0V	
181	B13	VDD_1V0	IN	输入电源 1.0V	
182	C14	VSS	IN	数字地	
183	B14	VSS	IN	数字地	

序号	引脚号	引脚名称	方向	信号说明	有效电平
184	F13	VSS	IN	数字地	
185	D9	VSS	IN	数字地	
186	G11	VSS	IN	数字地	
187	F12	VSS	IN	数字地	
188	F11	VSS	IN	数字地	
189	D14	VSS	IN	数字地	
190	E11	VSS	IN	数字地	
191	E9	VSS	IN	数字地	
192	E12	VSS	IN	数字地	
193	E13	VSS	IN	数字地	
194	G12	VSS	IN	数字地	
195	G14	VSS	IN	数字地	
196	E10	VSS	IN	数字地	
197	G13	VSS	IN	数字地	

备注：

1) NC表示此脚只能悬空，外面禁止接其它信号；

2) *标识信号为功能复用引脚，可通过GPREG寄存器选择该pin工作在功能模块引脚或者GPIO，详见《高性能32位多核处理器SOC芯片计算机模块用户手册》第11节通用寄存器GPREG介绍；

3) ** 标识信号在芯片初始化阶段，作为内部模块配置信号使用，要求接10K欧姆的上拉或下拉电阻，初始化完成后，用作第二功能，初始化配置请参考《高性能32位多核处理器SOC芯片计算机模块用户手册》2.2节“复位需预配置的外部信号”；第二功能是：Sp[10:0]用作ROM, SRAM, IO memory 地址信号。

4) 所有I/O引脚在上电后默认为INPUT脚，所有多功能脚在上电后默认作GPIO脚。

24. 计算机模块尺寸

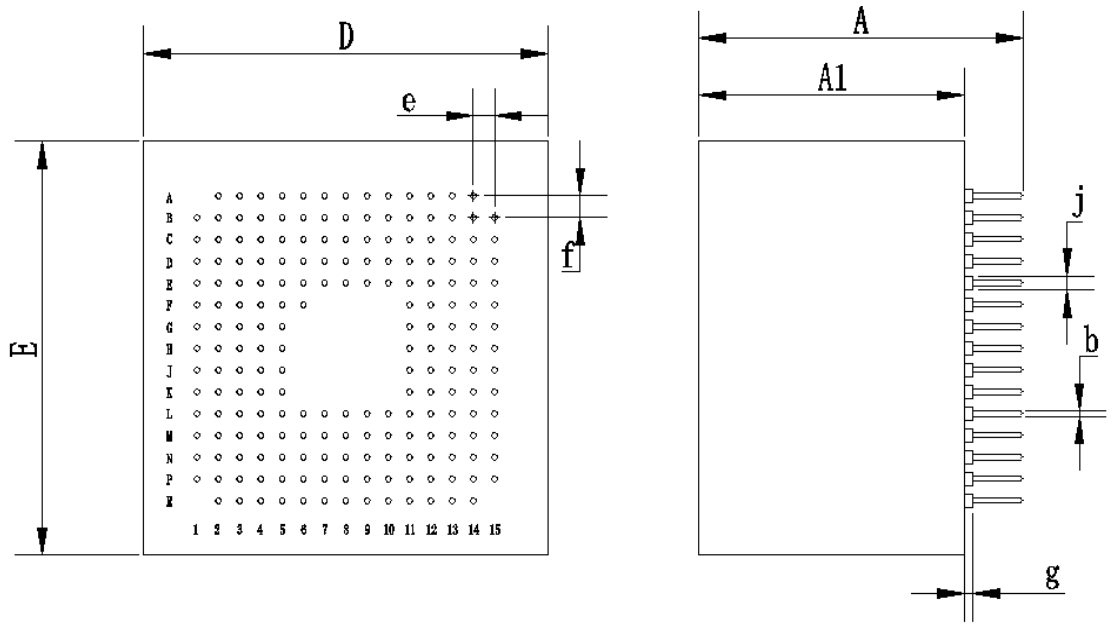


图 24-1 计算机模块尺寸图（TOP 面）

表 24-1 器件尺寸说明

	Min	Max
A	22.5	22.9
A1	17.0	17.4
D	41.8	42.2
E	41.8	42.2
b	0.46±0.03	
e	2.0	
f	2.0	
g	0.8	
j	1.27	
单位：mm		

25. 工作条件及电气特性

表 25-1 计算机模块关键参数极限范围

#	关键参数	极限范围
1	工作环境温度	-40℃~+85℃
2	存储温度	-65℃~+150℃
3	VDD_3V3 对 VSS 的电压差	-0.5~+3.6
	VDD_2V5 对 VSS 的电压差	-0.5~+2.8
	VDD_1V8 对 VSS 的电压差	-0.5~+2.3
4	VDD_1V0 对 VSS 的电压差	-0.5V~+1.32V
5	IO 引脚的输入电平	-0.5V~+5.5V
6	引脚的驱动能力	-8mA~8mA
7	功耗	≤5W

表 25-2 推荐直流(DC)工作条件（Ta= -40℃to+85℃）

特性	符号	最小值	典型值	最大值	单位
3.3V 输入电源	VDD_3V3	2.7	3.3	3.6	V
2.5V 输入电源	VDD_2V5	2.3	2.5	2.7	V
1.8V 输入电源	VDD_1V8	1.6	1.8	2.0	V
1.0V 输入电源	VDD_1V0	0.9	1.0	1.1	V
输入低电平电压	VIL	-0.3	——	VDD_3V3×0.2	V
输入高电平电压	VIH	VDD_3V3×0.8	——	VDD_3V3+0.3	V

备注：1.VCC 为计算机模块内部计算机模块处理器 IO 端口的工作电压，VCC=3.3V。

表 25-3 直流特性参数

技术参数	符号	测试条件:除另有规定外 -40℃ ≤ T_C≤85℃， VDD_3V3=3.3V, VDD_2V5= 2.5V, VDD_1V8= 1.8V, VDD_1V0= 1.0V,	最小值	最大值	单位
VDD_3V3 工作电流	Idd4	iuclk=400MHz, sysclk=50MHz	—	200	mA
VDD_2V5 工作电流	Idd3	iuclk=400MHz, sysclk=50MHz	—	15	mA

技术参数	符号	测试条件:除另有规定外 -40℃ ≤ T_C≤85℃， VDD_3V3=3.3V, VDD_2V5= 2.5V, VDD_1V8= 1.8V, VDD_1V0= 1.0V,	最小值	最大值	单位
VDD_1V8 工作电流	Idd2	iuclk=400MHz, sysclk=50MHz	—	150	mA
内核 1.0V 工作电流	Idd1	iuclk=400MHz, sysclk=50MHz	—	1780	mA
IO 3.3V 静态电流	Is4	RESET_IN = 0V	—	200	mA
SPW 2.5V 静态电流	Is3	RESET_IN = 0V	—	15	mA
DDR 1.8V 静态电流	Is2	RESET_IN = 0V	—	150	mA
内核 1.0V 静态电流	Is1	RESET_IN = 0V	—	1930	mA
输入漏电电流	I_{LIL}	V_{IN}=0V	-10	10	uA
	I_{LIH}	V_{IN}=3.6V	-10	10	uA
输出漏电电流	I_{LOL}	V_{OUT}=0V	-10	10	uA
	I_{LOH}	V_{OUT}=3.6V	-10	10	uA

备注：1，iuclk 为内核时钟；

2，sysclk 为系统时钟。

26. 产品内部器件信息

序号	器件名称	型号/规格	厂家	用量	抗辐射指标		
					TID (Krad(Si))	SEL (MeV·cm²/mg)	SEU (MeV·cm²/mg)
1	CPU	S698PM-PI	ORBITA	1	300	99.8	4
2	SRAM	R1RW0416DSB-2PI	RENESAS	3	100	99.8	0.7
3	FLASH	JFM29LV64IRH	复旦微电子	3	100	75	37
4	时钟	ZA517/C 20MHz 3.3V	203 所	1	300	无	无
5	时钟	ZA517/C 16MHz 3.3V	203 所	1	300	无	无
6	1553B	JSR1573	58 所	1	100	无	37

备注：序号 4~6 器件的抗辐射指标由供应商提供。

27. 产品订货信息

型号	质量等级	供货周期	备注
VDMC0003VP197EE5C02-计算机模块	工程样片	12 周	提供产品常温功能检测报告
VDMC0003RP197SS5C02-计算机模块	宇航级	18 周	提供产品筛选报告、内部芯片抗辐射报告

28. 产品命名

VD . MC . 0003 . x . P . 197 . x . x . 5 . C . 01

厂商代号

模块类型：
MC=计算机模块

产品编号

辐照能力：
R:有辐照测试;V=无辐照测试

封装形式：
P:PGA

引脚数量：
197:197 PIN

温度等级：
E:0～70℃；I:-40～85℃；M:-55～125℃；S:特殊

筛选等级：
E:工程样片；B:工业级；M:军用级；S:宇航级

叠装层数：
5:5 层

工作电压：
C:5V；S:2.8V；U:1.8V；Y:1.35V；V:3.3V；T:2.5V；W:1.5V

固件版本号：

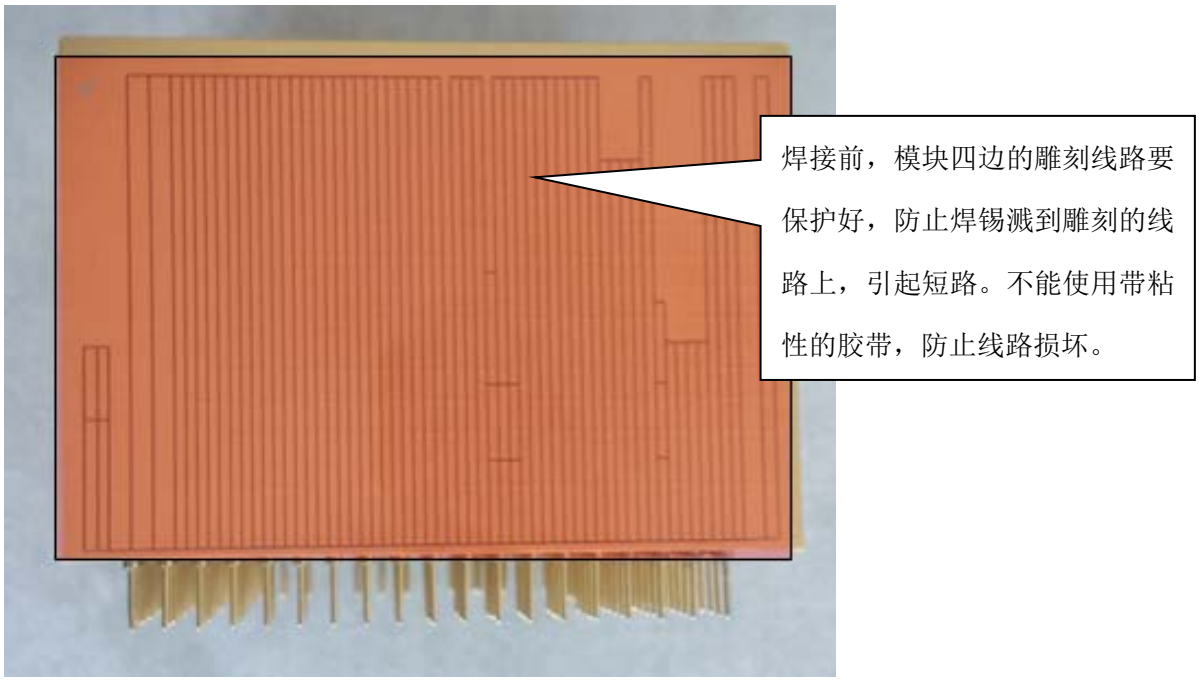
附录 A
（资料性附录）
器件焊接与加固建议

1.0 器件焊接

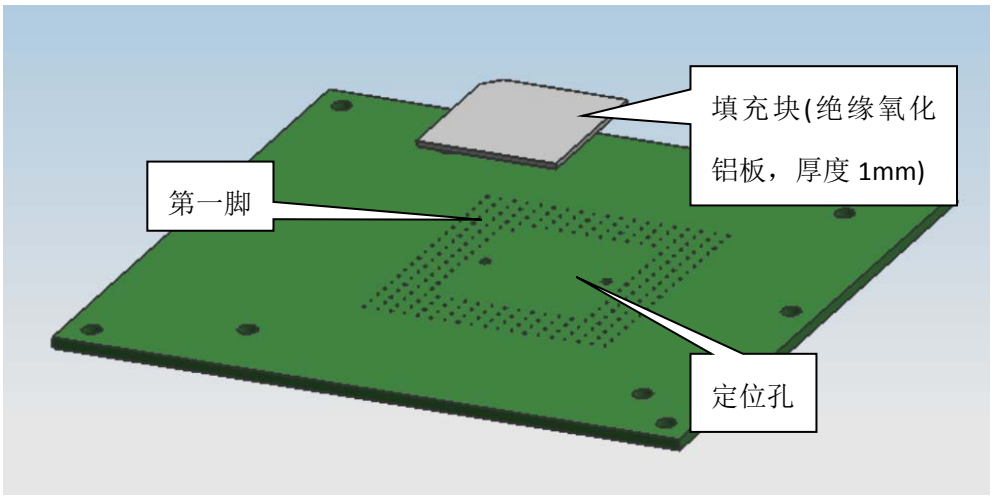
- 器件焊接仅适用于手工焊；
- 焊接材料应选择机械性能可靠，抗温变的，湿润性好，并且能与多种锡基和表面涂层相容的合金。焊接材料建议选用 Sn63Pb37（熔点+183℃）或 Sn62Pb36Ag2（熔点+179℃ ）；
- 器件在焊接前必须进行烘烤，烘烤温度 100℃，烘烤时间 8 小时；
- 焊接时，烙铁实际最高温度建议控制在+250℃到+280℃（最高不超过 320℃） ，每个点受热时间不超过 5 秒；
- 器件焊接后进行清洗，应选用对电路板和器件无腐蚀的清洗剂（无水乙醇）。

2.0 PGA 类模块加固（建议：重量 50g 以上的模块使用此类加固方法）

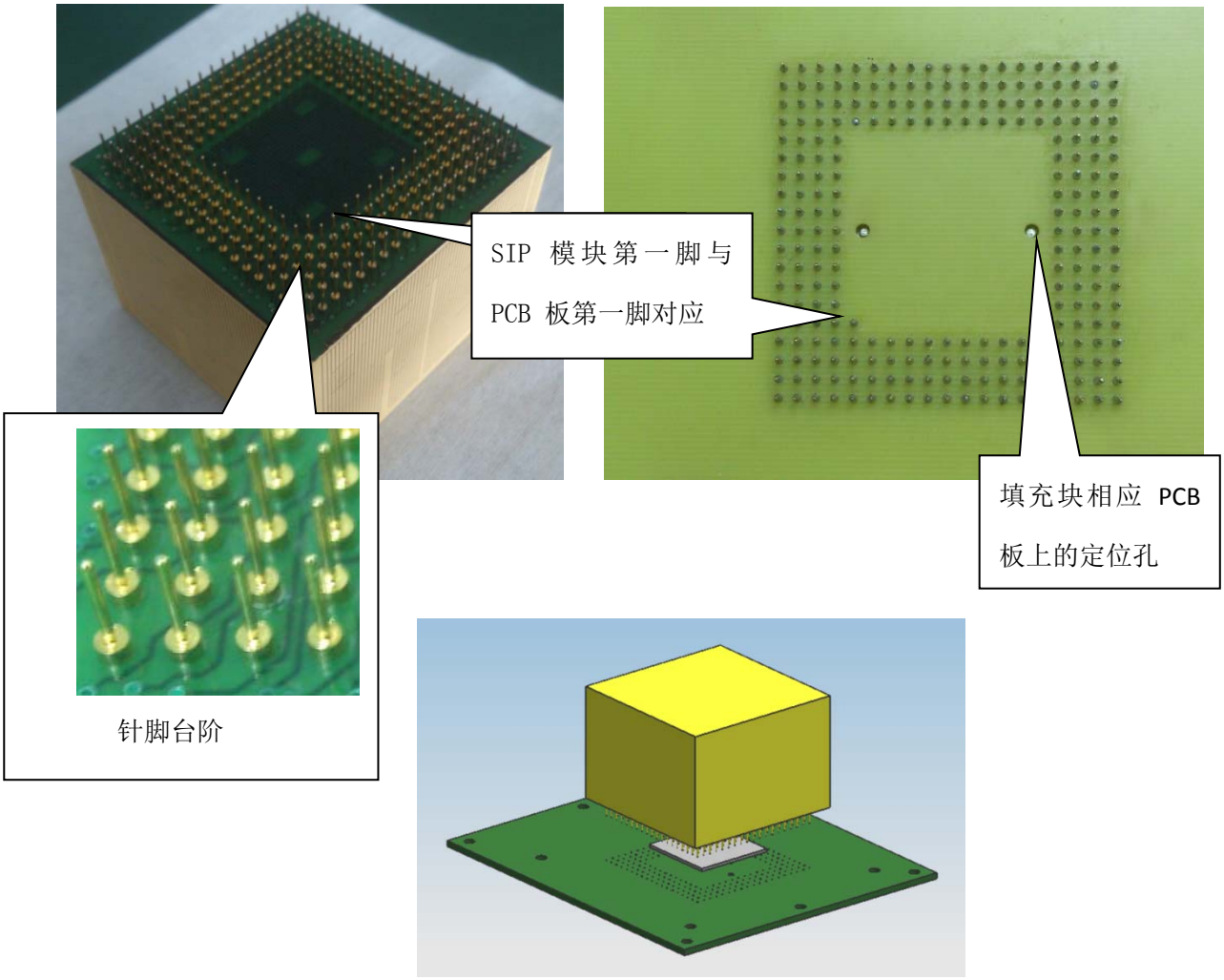
- 用高温胶制作一个模块保护套将模块保护好，防止焊接时表面线路上锡短路；



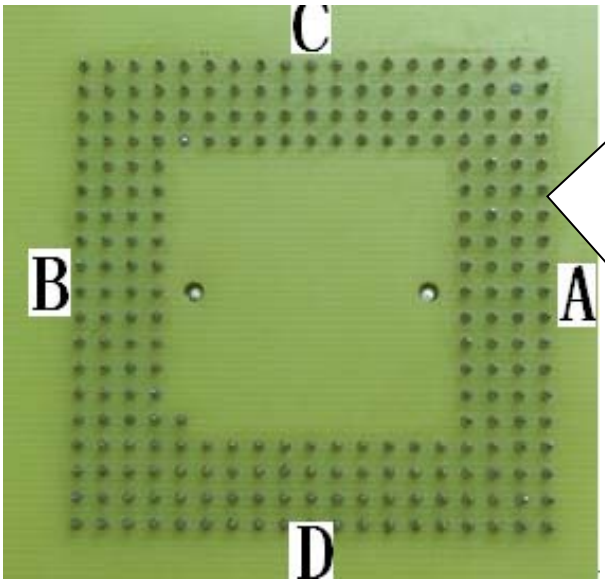
- 通过 PCB 板的定位孔与第一脚定位放置填充块（填充块设计应与模块相适配）

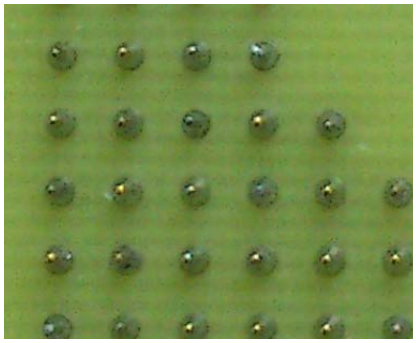


➢ 模块对准：把 SIP 模块按正确方向放置到 PCB 的焊接位置上（使用第一脚确定位置），将模块针脚与 PCB 焊接位置对准，轻轻将模块压平整（针脚台阶底部与 PCB 焊盘紧密平整接触，模块方向要正确）；



➢ 按照手工焊接要求将模块针脚焊接好，焊锡饱满，无虚焊、短路等，建议在显微镜下或放大镜下进行操作；



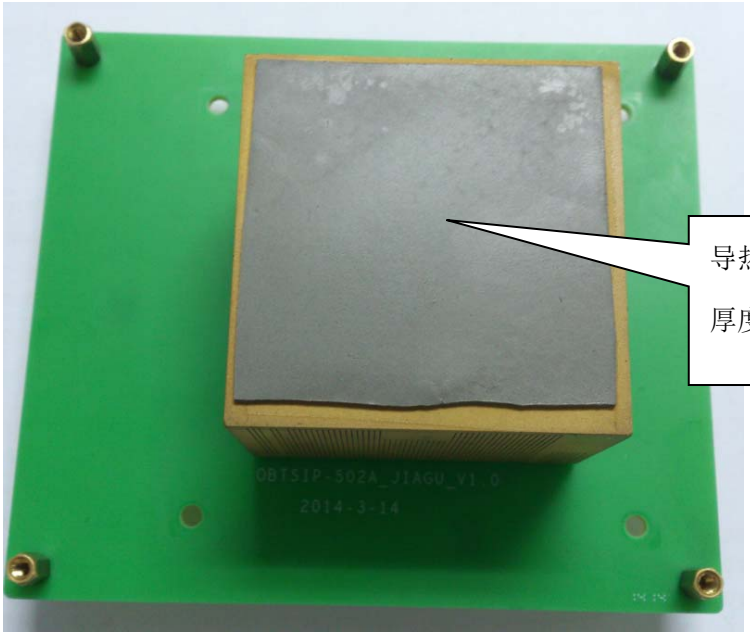


使用手工焊接将引脚焊接好（焊接温度 250～280℃），每个点的焊接时间 3～5 秒。相邻焊点焊接连续焊接 5 焊点后应转移焊接对面，防止局部过热，例如焊接 A 边相邻焊点焊接 5 焊点后，应转移焊接 B 边，如左图。

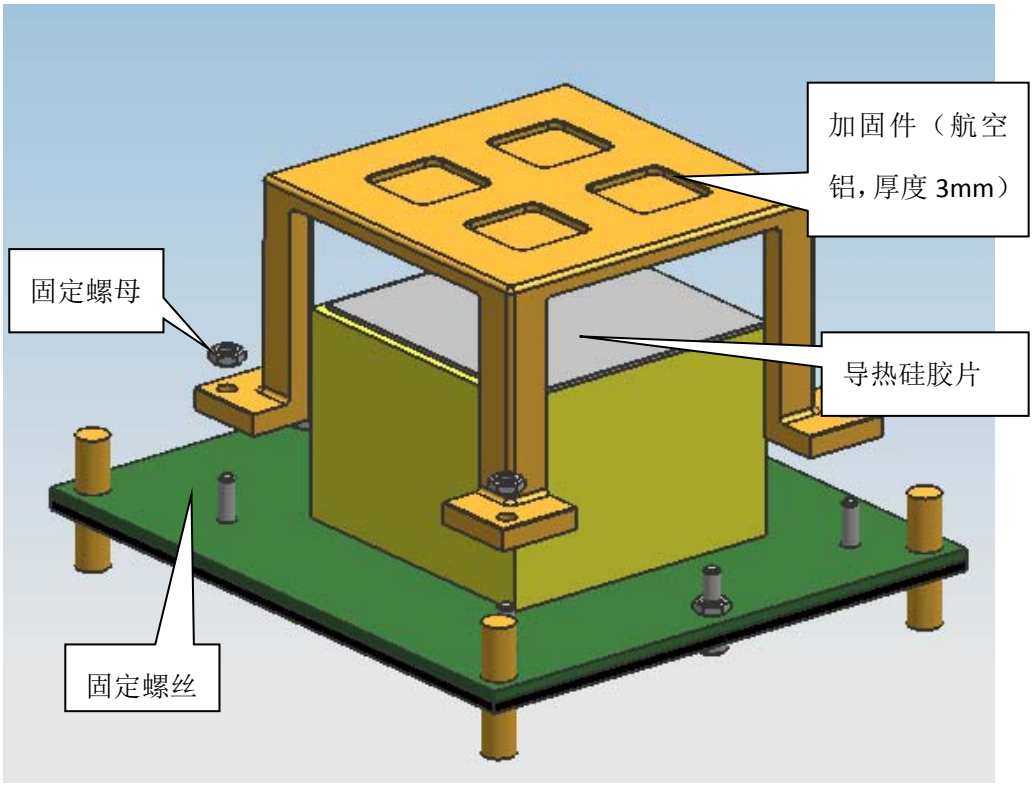
➢ 焊接完成后，除去表面的保护高温胶或保护物（使用力度应适中，不宜过猛造成雕刻线路损坏）。

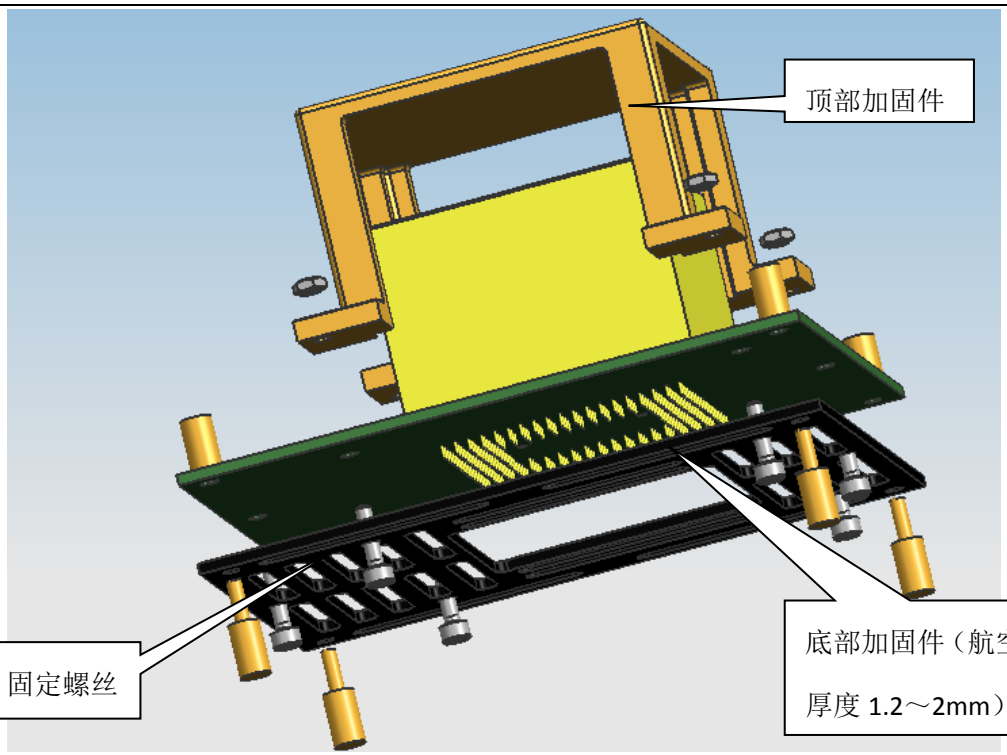
➢ 模块焊接后需要清洗，清洗方法详见欧比特公司提供的《立体封装模块焊接后清洗建议》

➢ 在焊接清洗处理完毕的模块顶部放置导热硅胶(导热尺寸应与模块尺寸相适配)，导热硅胶与模块紧密平整接触。

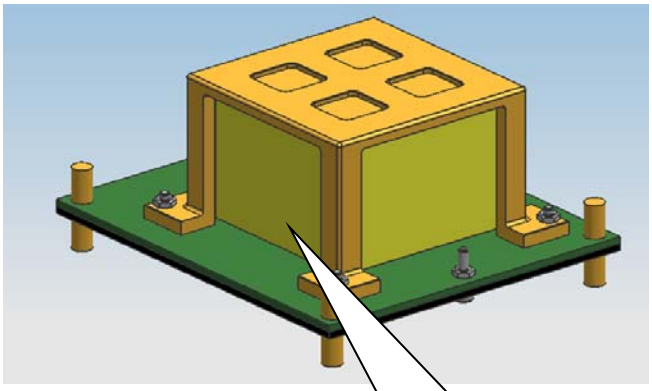


➢ 给模块与 PCB 板底部装上相适配的加固构件，装配图如下。（所有加固件都应与模块紧配）

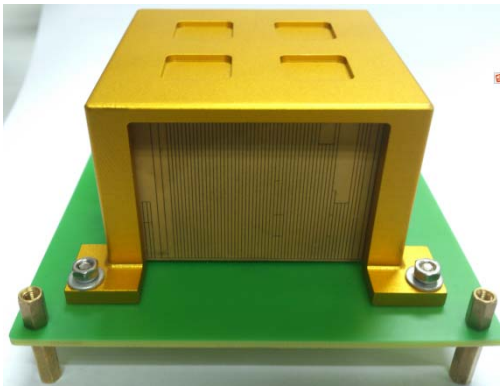




装配图



装配后模拟图



装配后实际图

模块顶部和 PCB 底部的加固件材料为航空铝，铝板厚度及加固方式可根据实际要求进行修改

3.0 注意事项

- 拿取模块前要戴上手指套和佩戴防静电手环；
- 焊接 SIP 模块操作时，应避免虚焊、引脚与 PCB 焊接位置对不准等现象，焊接之前检查方向正确后才能焊接；
- 焊锡适量、饱满光亮、焊接牢固，不能有虚焊、焊脚之间相互短路、错焊、漏焊等现象；