```
        uRegValue = (u32)(((u32)(0x1<<31))|(uMdiv<<16)|(uPdiv<<8)|(uSdiv<<0));
        Outp32(rMPLL_CON, uRegValue);
        break;

case eEPLL:
        // 检测分频器的值
        if( uMdiv < 13 || uMdiv > 255)
        {
            printf(" Wrong    Mdiv Value, Correct Value Range = (56 ~ 1023) (%d)\n", uMdiv);
        }
        if( uPdiv < 1 || uPdiv > 63)
        {
            printf(" Wrong    Pdiv Value, Correct Value Range = (1 ~ 63) (%d)\n", uPdiv);
        }
        if( uSdiv > 5)
        {
            printf(" Wrong    Sdiv Value, Correct Value Range = (0 ~ 5) (%d)\n", uSdiv);
        }
        if( uKdiv >65535)
        {
            printf(" Wrong    Kdiv Value, Correct Value Range = (0 ~ 65535) (%d)\n", uKdiv);
        }
          // 检测 Fvco 范围

           temp = ((FIN/uPdiv)*(uMdiv+uKdiv>>16))/1000000;
        if( temp <60 || temp > 250)
        {
            printf(" Please select the proper M,P,S divider value\n");
```

```
                printf(" Fvco Range = (60MHz ~ 250MHz),    Current Value is (%d)MHz\n", temp);
            }

            Outp32(rEPLL_CON1, uKdiv);
            uRegValue = (u32)(((u32)(0x1<<31))|(uMdiv<<16)|(uPdiv<<8)|(uSdiv<<0));
            Outp32(rEPLL_CON0, uRegValue);
            break;
        }

    // 得到信息
    SYSC_GetClkInform();
}
//读取 MEM_CFC_STAT 寄存器
void SYSC_RdMEMCFGSTAT( void )
{
    u32 uRegValue;
    u32 uTemp;

    uRegValue = Inp32(rMEM_CFG_STAT);

    // EBI  优先权配置
    uTemp = ( uRegValue >>15 ) & 0x1;
    printf(" Current EBI Priority Scheme     (0: Fixed, 1: Circular) :        %d \n",    uTemp);
    uTemp = (uRegValue >> 12) & 0x7 ;
    printf(" Current EBI Fixed Priority setting                                  :        %d \n",    uTemp);

    // CF I/F
    uTemp = ( uRegValue >>10 ) & 0x1;
    printf(" Current CF I/F Setting    (0: EBI, 1: Independet)        :        %d \n",    uTemp);
```

```
    // NAND 类型设置
    uTemp = ( uRegValue >>9 ) & 0x1;
    printf(" Current NAND Type (0:OneNAND, 1: NAND)              :      %d \n",    uTemp);
    uTemp = ( uRegValue >>3 ) & 0x1;
    printf(" Current NAND Init Setting    (0: Normal NAND, 1: Advanced NAND)   :      %d \n",    uTemp);
    uTemp = ( uRegValue >>2 ) & 0x1;
    printf(" Show address cycle init. setting of NAND        :      %d \n",    uTemp);
    uTemp = ( uRegValue >>0 ) & 0x1;
    printf(" Show NAND Page Size      :      %d \n",    uTemp);

    uTemp = ( uRegValue >>8 ) & 0x1;
    printf(" Current CS0 Bus Width    (0: 8-bit, 1: 16-bit)          :      %d \n",    uTemp);
    uTemp = ( uRegValue >>1 ) & 0x1;
    printf(" Show CS0 Bus width (init. Setting)    (0:8bit, 1: 16bit)     :      %d \n",    uTemp);

    uTemp = ( uRegValue >>7 ) & 0x1;
    printf(" NAND Booting      (0: not Used, 1: used)              :      %d \n",    uTemp);

    uTemp = ( uRegValue >>5 ) & 0x3;
    printf(" Current Booting Type    (0: NFCON, 1: SROMC, 2: ONDC,    3: Internal ROM)    :           %d    \n",
uTemp);

    uTemp = ( uRegValue >>4 ) & 0x1;
    printf(" Current ADDR Expand    (0: Used MEM1 Data, 1:Used MEM0 Addr)    :      %d \n",    uTemp);

}
```

4 存储器子系统

S3C6410 存储器包括七个存储控制器，一个 SROM 控制器，两个 OneNAND 控制器，一个 NAND 闪存控制器，一个 CF 控制器，和两个 DRAM 控制器。通过使用 EBI，静态存储控制器和 16 位 DRAM 控制器共享存储器端口 0。

4.1 存储器子系统的特性

S3C6410 存储器子系统的特性如下：

（1）存储器子系统有一个64位AXI从属器接口，一个32位AXI从属器接口，一个32位AHB主控器接口，两个32位从属器接口，其中一个用于数据传输，另一个用于SFR设置和一个用于DMC SFR设置的APB接口。

（2）存储器子系统从系统控制器获得导入方法和CS选择信息。

（3）内部AHB数据总线将32位AHB从属器数据总线和SROMC,两个 OneNANDC 和 NFCON连接起来。

（4）内部AHB SFR总线将32位AHB从属器SFR总线和SROMC、两个 OneNANDC、CFCON 和NFCON连接起来。

（5）内部 AHB 主控器总线用于 CFCON。

（6）DMC0用32位AXI从属器接口和APB接口。

（7）DMC1用64位AXI从属器接口和APB接口。

（8）存储器端口0通过使用EBI (外部总线接口)共享。

（9）仅DMC1启动用存储器端口1。

（10）支持使用NAND闪存或者OneNAND。

（11）对于CFCON，支持独立端口。

（12）存储器端口0中XmOCSn[1:0]专用于SROMC。

（13）存储器端口 0 中 XmOCSn[7:6]专用于 DMC0。

（14）选择NAND闪存或OneNAND导入设备，nCS2用于访问导入的媒体。

（15）EBI模块支持AMBA AXI 3.0低电源接口(CSYSREQ、CACTIVE、和CSYSACK)来阻止存储控制器来访问内存。

（16）通过在系统控制器中设置，存储器端口1的数据引脚[31:16]能用做端口0的地址引脚[31:16]。

（17）EBI模块支持通过存储控制器使用pad接口(DMC0, SROMC, 两个OneNANDC, CFCON, NFCON)。

（18）通过改变优先权来决定哪个能拥有pad接口。

（19）EBI和包含一个三线接口、EBIREQ、EBIGNT和EBIBACKOFF，和所有活动的高位的存储器之间相互通信：

- EBIREQ信号被存储控制器访问来指示它们需要外部总线访问。
- 各自的EBIGNT被发送到高优先权的存储控制器。
- EBI 输出EBIBACKOFF来发送信号，存储控制器必须完成当前传输释放总线。

（20）EBI保持被授权的存储控制器的跟踪，并且在它授权给下一个存储控制器之前，等待从存储控制器到闪存的传输。如果高优先权的存储控制器请求总线，那么EBIBACKOFF通知当前被授权的控制器来尽快结束当前传输。

EBI模块框图如图4-1所示，通过EBI的存储器接口如图4-2所示。

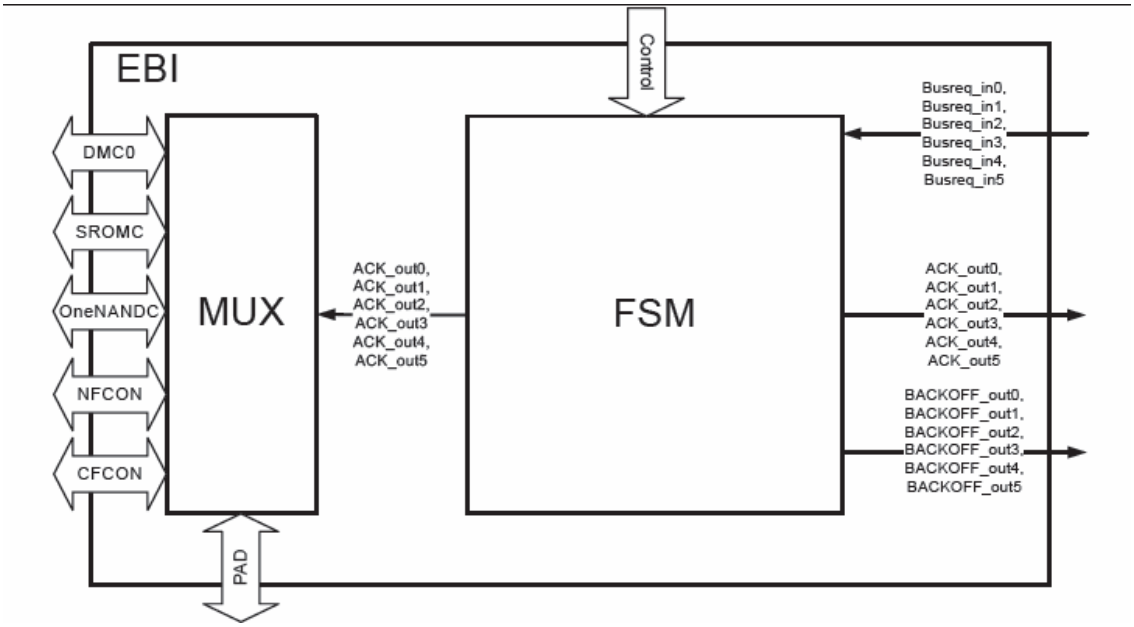

图4-1　EBI模块图

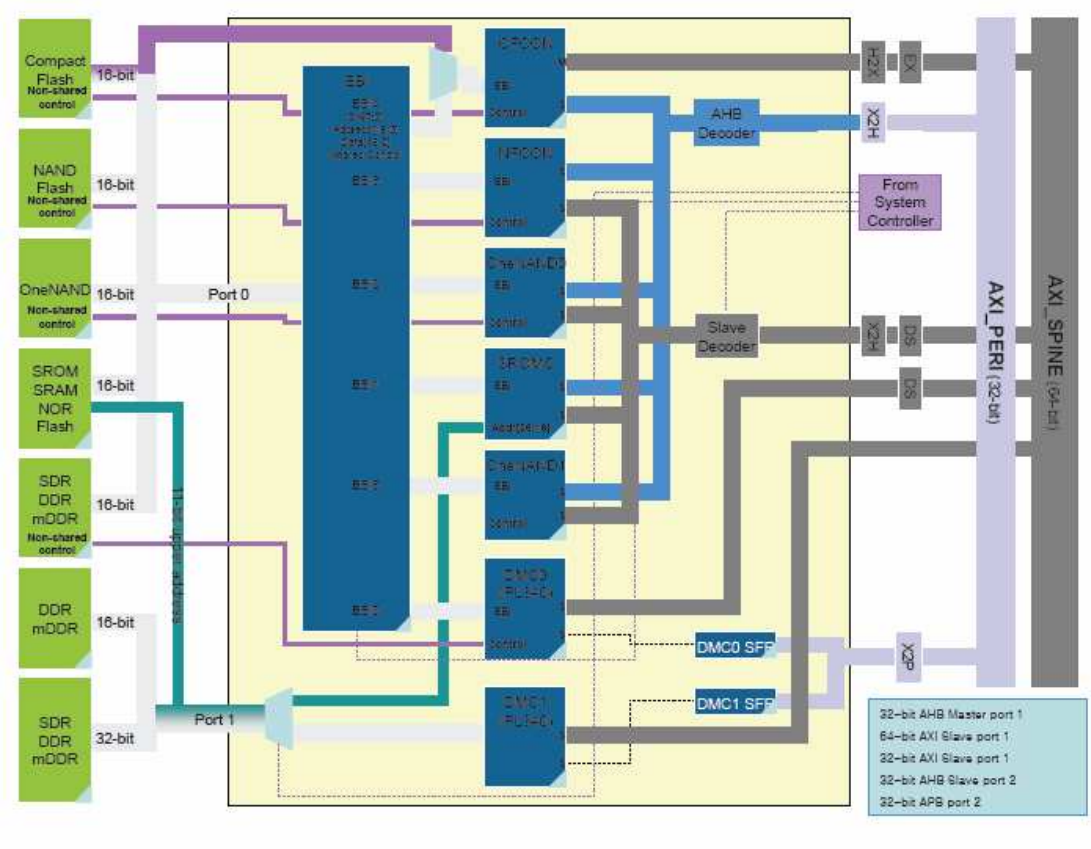

图4-2 通过EBI的存储器接口

4.2 功能描述

存储器子系统能通过系统控制器进行配置。系统控制器中，存储器构造寄存器名称是MEM_SYS_CFG。该寄存器的地址是0x7E00F120。

系统控制器发送信息如下：

4.2.1. 闪存信息

（1）NAND闪存导入启动

（2）高级NAND闪存

0= 正常 NAND / 1= 高级 NAND。

（3）地址周期。

- 正常NAND，0= 3个周期/ 1= 4个周期。
- 高级NAND，0= 4个周期/ 1= 5个周期。

（4）NAND闪存数据总线宽度

8位。

（5）页大小选择

- 正常NAND，0= 256字节/ 1= 512字节。
- 高级NAND，0= 1KB/ 1= 2KB。

（6）端口信息

①SROM 数据总线宽度

- 可以通过 SROM控制器 SFR设定改变。
- 0 ： 8位， 1 ： 16位。

②地址扩展

- 0= 用 Xm1DATA[31:16] 作为高半字数据来自存储器端口1。
- 1= 用Xm1DATA[31:16] 作为高半字地址来自存储器端口0。

  从端口1的数据引脚[31:16]为端口0借用地址位[26:16]。

③导入位置

导入位置信息能被在MEM_CFG_STAT[6:5]检查，如表4-1所示。

表4-1 导入位置

CfgBootLoc[1:0]	导入位置
2’b00	NFCON的Stepping Stone区域
2’b01	SROMC CS0
2’b10	OneNANDC CS0
2’b11	内部 ROM

④存储器端口0 CS选择

- 设置存储器端口0的静态存储芯片选择多路复用。
- 忽略MP0_CS_SEL[0]和MP0_CS_SEL[2]的设置。通过XSELNAND引脚值区别OneNANDC和NFCON。当XSELNAND是0，则OneNANDC被选择；当XSELNAND是1，则NFCON被选中。
- 当选择NAND导入(XOM[4:3] = 00)时，MP0_CS_SEL[1]和MP0_CS_SEL[3] 的设置值被忽略。XmOCSn[2]

和XmOCSn[3]用做NFCON CS0和NFCON CS1。这种情况下，XSELNAND必须设置为1。

- 当OneNAND导入(XOM[4:1] = 0110)时，MP0_CS_SEL[1]和MP0_CS_SEL[3]的设置值被忽略。XmOCSn[2]和XmOCSn[3]被用作OneNANDC CS0和OneNANDC CS1。这种情况下，XSELNAND必须设置为0。如表4-2所示。

表 4-2 存储器端口 0 CS 选择

MP0_CS_SEL[5:0]	=0	=1	=2	=3
{[1],XSELNAND}	SROMC CS2	SROMC CS2	OneNANDC CS0	NFCON CS0
{[3],XSELNAND}	SROMC CS3	SROMC CS3	OneNANDC CS1	NFCON CS1
[4]	SROMC CS4	CFCON CS0		
[5]	SROMC CS5	CFCON CS1		

⑤为CFCON选择独立的端口

- 0= CFCON 使用 EBI的共享端口。
- 1= CFCON 使用独立端口。

⑥CKE初始值 (SPCONSLP[4](0x7F0088B0))

- 0 = 复位后，存储器端口0的XmOCKE 和端口1的Xm1CKE的初始化值为0。
- 1 = 复位后，存储器端口0的XmOCKE 和端口1的Xm1CKE的初始化值为1。

⑥EBI

- 优先类型

  0=固定优先类型 / 1= 循环优先。

- 固定优先顺序

如表4-3所示，显示了固定的优先顺序。

表 4-3 固定优先顺序

CfgFixPriTyp[2:0]	1st	2nd	3rd	4th	5th	6th
	DMC0	SROMC	OneNANDC CS0	OneNANDC CS1	NFCON	CFCON
1	DMC0	OneNANDC	OneNANDC	SROMC	NFCON	CFCON

		CS0	CS1			
2	DMC0	OneNANDC CS1	NFCON	SROMC	OneNANDC CS0	CFCON
3	DMC0	NFCON	SROMC	OneNANDC CS0	OneNANDC CS1	CFCON
4	DMC0	CFCON	SROMC	OneNANDC CS0	OneNANDC CS1	NFCON
5	SROMC	DMC0	OneNANDC CS0	OneNANDC CS1	NFCON	CFCON

（7）总线信息

①MP0_QOS_OVERRIDE[15:0]

- 设置QoS 取代ID到DMC0。
- 系统控制器中，QOS_OVERRIDE0 控制该信息。
- 该寄存器地址是 0x7E00F124。

②MP1_QOS_OVERRIDE[15:0]

- 设置 QoS 取代 ID到DMC1。
- 系统控制器中QOS_OVERRIDE1控制该信息。
- 该寄存器地址是 0x7E00F128。

4.3 EBI 接口

EBI，作为外设，当它们空闲时，依靠存储控制器来释放它们的外部请求。如图4-3所示，一个简单的相互通信的例子。在这个例子里，一个设备请求外部总线，因为当先没有其它设备请求总线，所以立即产生，

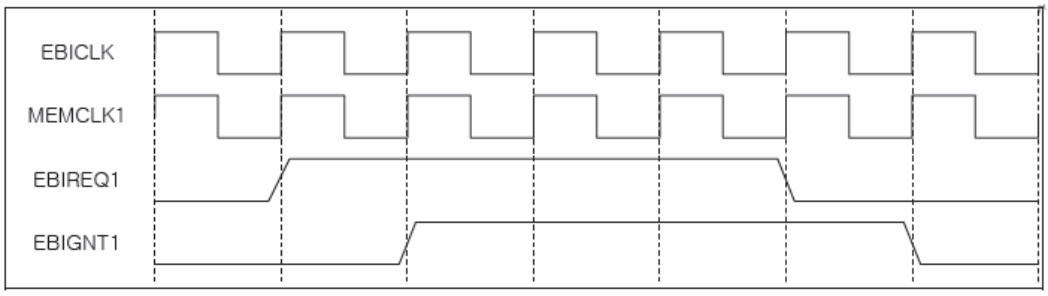

图4-3 EBIREQ, EBIGANT 信号

如果一个高优先权的设备请求总线，且是一个低优先权的设备控制该外部总线时，则EBIBACKOFF通知低优先权的设备尽快释放。

在图4-4中，一个高优先权的设备要求总线，不久这个设备就给予总线的权限。EBIBACKOFF通知设备尽早结束访问。设备一被给予总线权限并完成传输。当传输完成，设备二被给予权限来完成被中断的传输。EBIREQ2信号必须降低至少一个时钟周期，之后被释放。

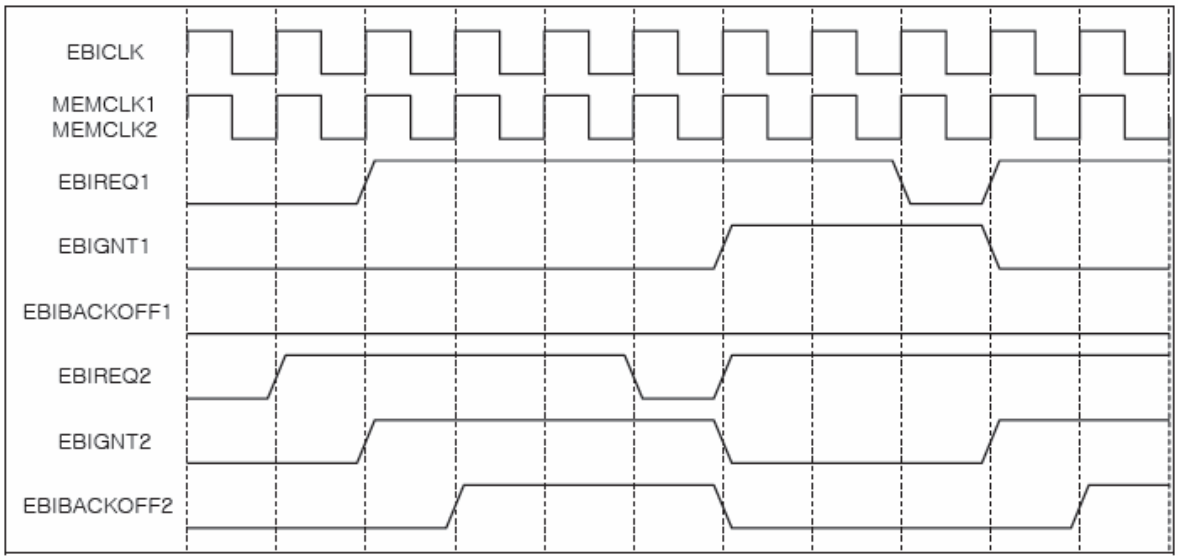

图4-4 EBIBACKOFF 信号