```
printf("[ RTC Alarm Test for S3C6410 ]\n");  //输出[ RTC Alarm Test for S3C6410 ]信息。

RTC_SetCON(0,0,0,0,0,1);     //没有复位, 合并 BCD 计数器, 1/32768, RTC 控制使能。

RTC_SetAlmTime(AlmYear,AlmMon,AlmDate,AlmHour,AlmMin,AlmSec);
printf("Select alarm interrupt source \n");
printf("1:sec 2:min 3:hour 4:date 5:month 6:year 7:All components\n");
uSelect = GetIntNum();
switch(uSelect)
{
    case 1:
        RTC_SetAlmEn(1,0,0,0,0,0,1);
        break;
    case 2:
        RTC_SetAlmEn(1,0,0,0,0,1,0);
        break;
    case 3:
        RTC_SetAlmEn(1,0,0,0,1,0,0);
        break;
    case 4:
        RTC_SetAlmEn(1,0,0,1,0,0,0);
        break;
    case 5:
        RTC_SetAlmEn(1,0,1,0,0,0,0);
        break;
    case 6:
        RTC_SetAlmEn(1,1,0,0,0,0,0);
        break;
    case 7:
```

```
        RTC_SetAlmEn(1,1,1,1,1,1,1);
        break;
    default :
        RTC_SetAlmEn(1,1,1,1,1,1,1);
        break;
}
printf("After 5 sec, alarm interrupt will occur.. \n");
RTC_PrintAlm();
RTC_Print();
uCntAlarm = 0;

INTC_SetVectAddr(NUM_RTC_ALARM,Isr_RTC_Alm);
INTC_Enable(NUM_RTC_ALARM);

RTC_SetCON(0,0,0,0,0,0);  //没有复位, 合并 BCD 计数器, 1/32768, RTC 控制禁用。

while(uCntAlarm==0)
{

};
RTC_Print();
INTC_Disable(NUM_RTC_ALARM);
RTC_SetCON(0,0,0,0,0,0); //RTC 控制禁用(用于电源消耗), 1/32768, 正常的(合并), 没有复位}
}
```

34　看门狗定时器

当控制器操作被噪音或系统错误等故障打断时，S3C6410 RISC 微处理器的看门狗定时器恢复控制器的操作。它用于正常的 16 位间隔定时器来要求中断服务。看门狗定时器产生中断信号。用 WDT 代替 PWM 定时器的优点是 WDT 产生复位信号。

34.1 看门狗定时器的特性

看门狗定时器包含下面的特性：

- 具有中断请求的正常间隔定时器模式。
- 当定时器计数值达到0（超时），内部复位信号有效。
- 电平触发器中断机制。

34.2 功能描述

如图 34-1 所示，显示看门狗定时器的功能模块图。看门狗定时器用 PCLK 作为它的源时钟。PCLK 频率被预分频来产生相应的看门狗定时器时钟，并将所得的频率再次被分频。

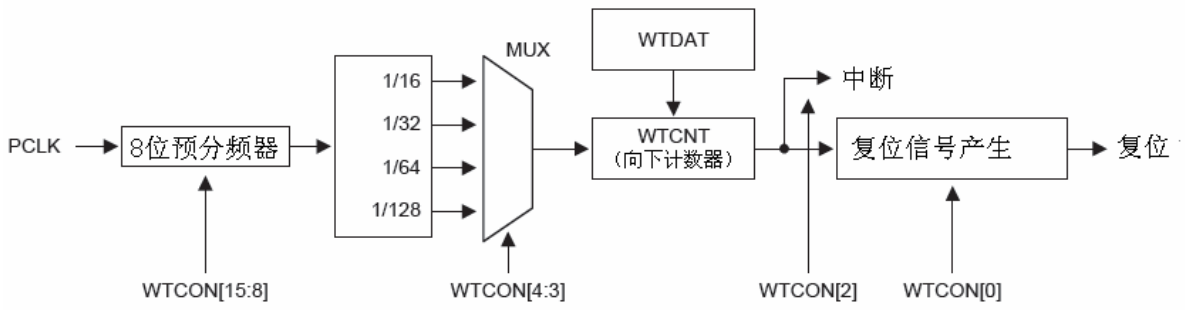

图 34-1　看门狗定时器模块图

在看门狗定时器控制(WECON)寄存器，预分频器值和分频因子被指定。有效的预分频器值的范围是 0～

28-1。分频因子能选择 16、32、64 或 128。

用下面的等式来计算看门狗定时器时钟频率，每一个定时器的时钟周期：

看门狗定时器时钟频率 = 1/(PCLK / (预分频值 + 1) / 分频因子)

一旦看门狗定时器有效，看门狗定时器数据（WTDAT）寄存器的值将不能被自动重新载入定时器计数器（WTCNT）。在看门狗定时器开始前，一个初始值必须写入看门狗定时器计数（WTDAT）寄存器。

在 ARM11 处理器中，看门狗操作的具体代码如下:

```
// WDT_operate 函数：主要功能通过输入操作看门狗定时器。
// 输入: uEnReset, uEnInt, uEnWDT [0:禁用    1:使能]
//       uSelectCLK (时钟分频因子) [0:16    1:32   2:64   3:128]
//       uPrescaler [1~256]
//       uWTDAT [0~2^15]
//       uWTCNT [0~2^15]
// 输出: NONE
void WDT_operate(u32 uEnReset, u32 uEnInt, u32 uSelectCLK, u32 uEnWDT, u32 uPrescaler,u32 uWTDAT, u32 uWTCNT)
{
    float fWDTclk;
    Outp32(rWTCON,0);
    Outp32(rWTDAT,0);
    Outp32(rWTCNT,0);
    Outp32(rWTDAT,uWTDAT);
    Outp32(rWTCNT,uWTCNT);
Outp32(rWTCON,(uEnReset<<0)|(uEnInt<<2)|(uSelectCLK<<3)|(uEnWDT<<5)|((uPrescaler)<<8));
fWDTclk = (1/(float)((float)g_PCLK/((float)uPrescaler+1)/
(1<<(uSelectCLK+4))))*uWTDAT;
    printf("WDT_clk = %f sec\n",fWDTclk);
}
```

34.3 考虑调试环境

当S3C6410在调试模式（使用嵌入的ICE）时，看门狗定时器不能进行操作。

在调试模式下，看门狗定时器能决定来自 CPU 内核的信号（DBACK 信号）是否是当前的信号。一旦 DBGACK 信号无效，看门狗定时器的复位输出禁止，并终止看门狗定时器终止。

34.4 特殊功能的寄存器

存储器映射如表34-1所示。

表 34-1 存储器映射

寄存器	地址	读/写	描述	复位值
WTCON	0x7E004000	读/写	看门狗定时器控制寄存器。	0x8021
WTDAT	0x7E004004	读/写	看门狗定时器数据寄存器。	0x8000
WTCNT	0x7E004008	读/写	看门狗定时器计数寄存器。	0x8000
WTCLRINT	0x7E00400c	写	看门狗定时器中断清除寄存器。	-

34.4.1. 看门狗定时器控制（WTCON）寄存器

WTCON 寄存器允许用户启动/禁止看门狗定时器，从四个不同的时钟源选择时钟信号，启动/禁止看门狗定时器输出。

看门狗定时器用于恢复 S3C6410 重启故障。如果不希望控制器重启，则看门狗定时器必须禁止。如果用户想使用看门狗定时器提供的正常定时器，需要启动中断，并且禁止看门狗定时器。如表 34-2 所示。

表 34-2 WTCON 寄存器

寄存器	位	读/写	描述	复位值
WTCON	0x7E004000	读/写	看门狗定时器控制寄存器。	0x8021

WTCON	位	描述	初始状态
Prescaler value	[15:8]	预分频值。	0x80

		该值范围是 0～（28-1）。	
Reserved	[7:6]	保留。 正常操作这两位必须是 00。	00
Watchdog timer	[5]	启动或禁止看门狗定时器的位。 0=禁止 1=有效	1
Clock select	[4:3]	决定时钟分频因子。 00: 16　　01 : 32 10: 64　　11 : 128	00
Interrupt generation	[2]	中断的启动或禁止位。 0=禁止 1=启动	0
Reserved	[1]	保留。 正常操作时，该位必须设置为 0。	0
Reset enable/disable	[0]	对于复位信号，启动或禁止看门狗定时器的输出位。 1:看门狗定时器超时，发出复位信号 0:禁止看门狗定时器的复位功能	1

34.4.2. 看门狗定时器数据（WTDAT）寄存器

WTDAT寄存器用于指定超时时间。看门狗定时器初始化时，WTDAT的内容不能被自动载入到定时器计数器。然而，使用0x8000（初始值）可以驱使第一个超时。这种情况下，WTDAT的值将自动重载入WTCNT。如表34-3所示。

表 34-3WTDAT 寄存器

寄存器	地址	读/写	描述	复位值
WTDAT	0x7E004004	读/写	看门狗定时器数据寄存器。	0x8000

WTDAT	位	描述	初始状态
Count reload value	[15:0]	看门狗定时器重载的计数值。	0x8000

34.4.3. 看门狗定时器计数（WTCNT）寄存器

正常操作情况下，WTCNT 寄存器包含看门狗定时器的当前计数值。

注：当看门狗定时器初始化时，WTDAT 寄存器的内容不能被自动载入到定时器计数寄存器中，因此在激活它前，WTCNT 寄存器必须设置一个初始值。如表 34-4 所示。

表 34-4 WTCNT 寄存器

寄存器	地址	读/写	描述	复位值
WTCNT	0x7E004008	读/写	看门狗定时器计数寄存器。	0x8000

WTCNT	位	描述	初始状态
Count value	[15:0]	看门狗定时器的当前计数值。	0x8000

34.4.4. 看门狗定时器中断清除（WTCLRINT）寄存器

WTCLRINT 寄存器用于清除中断。中断服务完成后，中断服务程序清除相关中断。可以在该寄存器写入任意值清除中断。不允许读该寄存器。如表 34-5 所示。

表 34-5 WTCLRINT 寄存器

寄存器	地址	读/写	描述	复位值
WTCLRINT	0x7E00400c	写	看门狗定时器中断清除寄存器。	-

WTCLRINT	位	描述	初始状态
中断清除	[0]	写任意值来清除中断。	-

35 AC97 控制器

本节主要介绍 AC97 控制器在 S3C6410X RISC 微处理器上的功能及其使用。

35.1 AC97 控制器概述

S3C6410X 的 AC97 控制器单位支持 AC97 2.0 版的功能。带有 AC97 解码器的 AC97 控制器用于连接音频控制器(AC-link)，控制器发送立体声 PCM 数据给多媒体数字信号编解码器。在多媒体数字信号编解码器里，外部数字式模拟转换器（DAC）由音频采样转换为一个模拟音频波形。控制器多媒体数字信号编解码器里接收立体声 PCM 数据和单声道麦克风的数据，然后存入存储器中。这一节主要介绍了 AC97 控制器单元的设计模型。

1. AC97 控制器

AC97 控制器包括以下特性：

（1）独立通道，用于立体声 PCM 输入，立体声 PCM 输出和单声道 MIC 输入。

（2）基于 DMA 操作和基于中断操作。

（3）变量取样速率 AC97 多媒体数字信号编解码器接口（48KHz 和低于 48KHz）。

（4）16 位，16 个 FIFO 每通道。

（5）只支持基本多媒体数字信号编解码器。

2.信号

显示信号，如表 35-1 所示。

表 35-1 显示信号

名称	方向	说明
AC_nRESE	输出	低级多媒体数字信号编解码器复位
AC_BIT_CL	输入	12.288MHz 位速率时钟
AC_SYNC	输出	48KHz 指示器结构和同步装置
AC_SDO	输出	连续音频输出数据
AC_SDI	输入	连续音频输入数据

35.2 AC97 工作流程

本节主要说明 AC97 控制器的操作，如 AC-Link，省电序列和唤醒序列。

1.结构框图

这是一个点对点的同步串行连接，支持全双工程信制数据传输。所有的数字音频流和命令/状态信息是通过 AC-link 相通起来的。如图 35-1 所示，显示了 S3C6410 AC97 控制器的功能结构框图。

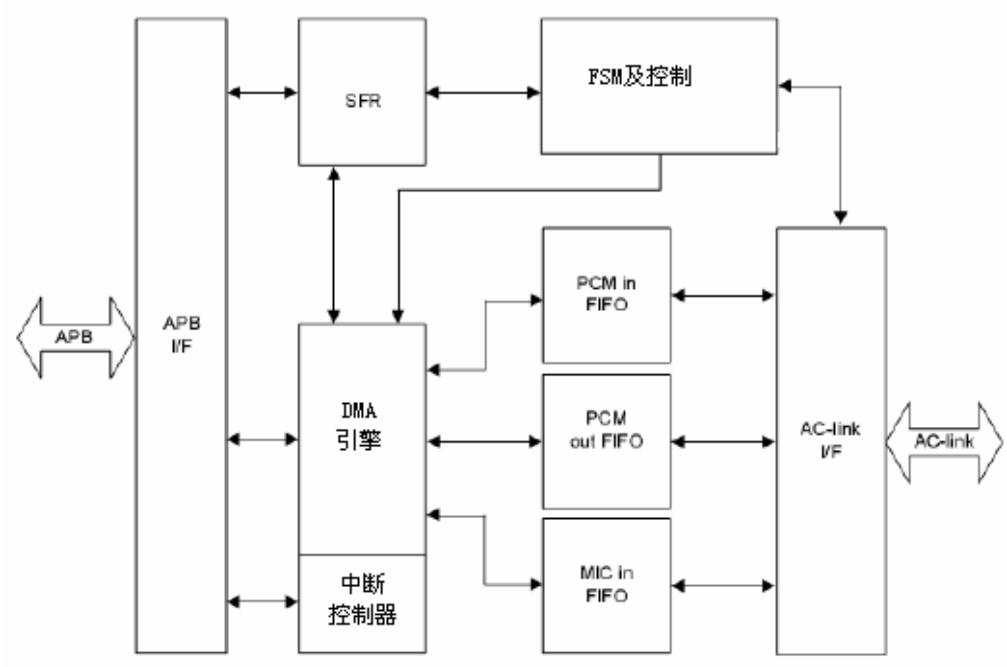

图 35-1 S3C6410 AC97 控制器的功能结构框图

2.内部数据通道

它有立体声编码调制（PCM）输入，立体声 PCM 输出和单声道 Mic-in 缓冲器，其中包括 16 位和 16 个缓冲器。它也有 20 位 I/O 移位寄存器通过 AC-link。如图 35-2 所示，显示了 S3C6410 AC97 控制器的内部数据通道。

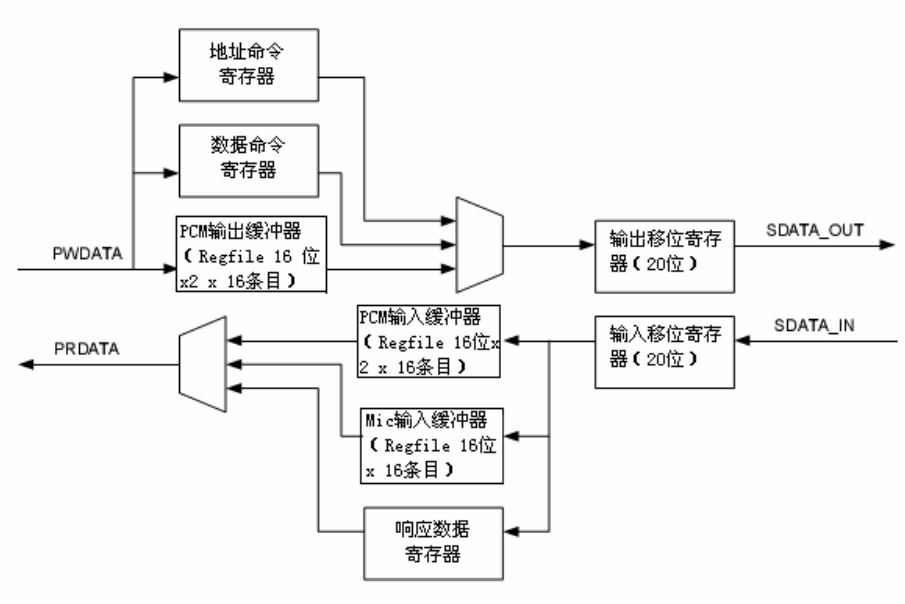

图 35-2 内部数据通道

3. 操作流程图

AC97 控制器在 ARM11 中，支持 AC97 2.0 版的功能。它带有 AC97 解码器，主要用于连接音频控制器(AC-link)。

当初始化 AC97 控制器时，首先必须进行系统复位或冷复位，因为我们不知道以前的外部 AC97 音频-编解码器的状态。

ARM11 中初始化 AC97 控制器的代码如下：

```
//AC97 控制器初始化
bool AC97_Init(void)
{
    u32 i=0;
    Disp("\nAC97 Initialization...\n");
//编解码器准备就绪，检查是否使用了编解码器准备中断
    //AC97 冷复位
    AC97_ColdReset();
    uCodecReadyIrq =0;
    INTC_SetVectAddr(NUM_AC97, Isr_AC97_CodecReady);
```