AT91SAM7X-EK softpack 1.5 代码解读之 RTT

——基于 SAM7X-EK 和 IAR EWARM

文档编号	MAN2011A_CH				
文档版本	Rev. A				
文档摘要	基于 AT91SAM7X-EK 开发板的代码解读，RTT 代码解读				
关键词	AT91SAM7X256 SAM7X-EK IAR EWARM J-LINK RTT				
创建日期	2010-06-08	创建人员	Cust126	审核人员	Robin
文档类型	公开发布/开发板配套文件				
版权信息	Mcuzone 原创文档，转载请注明出处				

更新历史

版本	时间	更新	作者
Rev. A	2010-06-10	初始创建	Cust126

微控电子 乐微电子
杭州市登云路 639 号 2B143
销售 TEL：86-571-89908193 13957118045
支持 TEL：18913989166 13957118045
FAX：86-571-89908193
www.mcuzone.com www.atarm.com

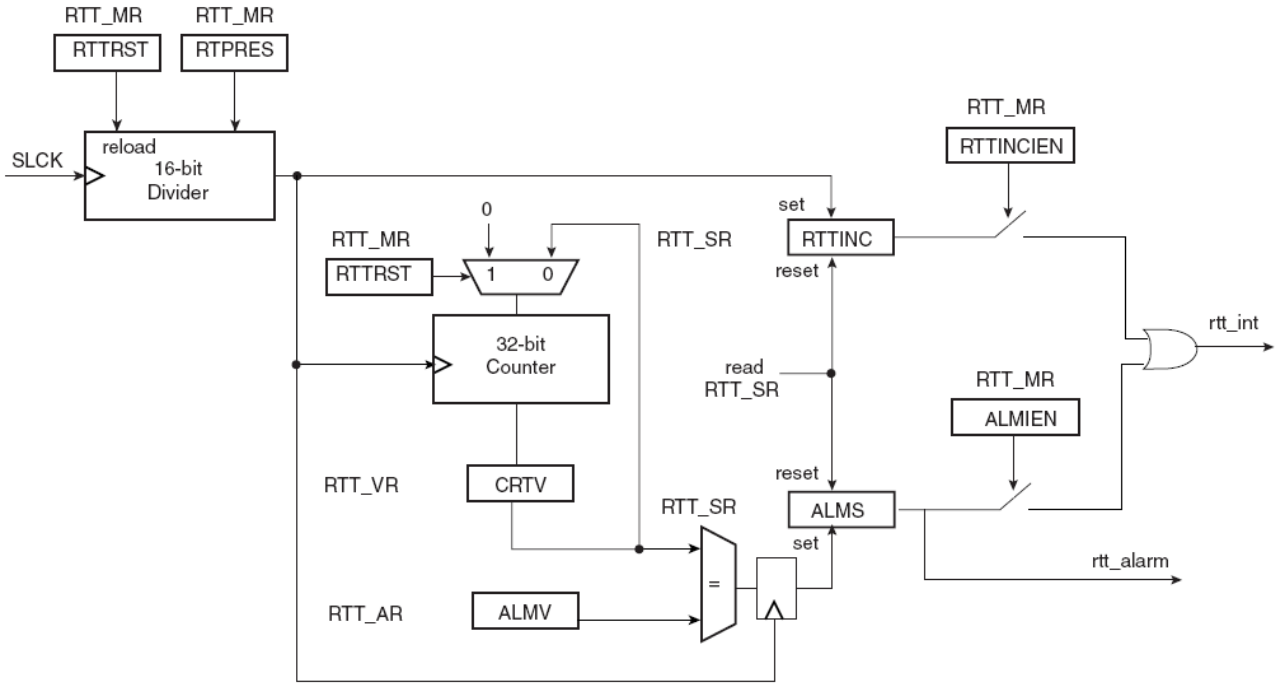
1.概述

本文档以 SAM7X-EK 为硬件平台，IAR EWARM 为编译器平台，使用 J-Link 作为调试工具，演示并解读 AT91SAM7X256 的 RTT 操作流程。

2. RTT 操作

2.1 RTT 操作流程介绍

RTT 主要有 RTT_MR、RTT_AR、RTT_VR、RTT_SR 4 个寄存器；
RTT 的初始化流程如下：
先配置计数器的时钟来源为慢速时钟，并经过 16 位数值的预分频。这个数值要写入实时模式寄存器 RTT_MR 的 RTPRES 域。配制 RTT 中断，开中断，开启实时定时器加 1 中断。
实时定时器方框图：



2.2 RTT 操作的目的与功能描述

AT91 softpack 1.5 的 SAM7X-EK 包内的 basic-rtt-project 工程目的主要是帮助初学者熟悉 RTT 在 AT91SAM7X 系列上的用法。RTT 计数器能够设置报警，当计数器到达用户预设定的值时触发报警。

2.3 示例代码

以下代码截取自 AT91 softpack 1.5 的 SAM7X-EK 包内的 basic-rtt-project ，基于 IAR EWARM 平台

2.3.1 main 函数代码注释解读

下面对 basic-rtt-project 的主要代码进行注释解读，首先是 main 函数内容：

```
//------------------------------------------------------------------------------
/// Initializes the RTT, displays the current time and allows the user to
/// perform several actions: clear the timer, set an alarm, etc.

//初始化 RTT，显示当前时间，允许用户设置以下参数：清除计数器、设置报警等
//------------------------------------------------------------------------------
int main(void)
{
    unsigned char c;
```

```
// Enable DBGU     //配置 DBGU
TRACE_CONFIGURE(DBGU_STANDARD, 115200, BOARD_MCK);
printf("-- Basic RTT Project %s --\n\r", SOFTPACK_VERSION);
printf("-- %s\n\r", BOARD_NAME);
printf("-- Compiled: %s %s --\n\r", __DATE__, __TIME__);

// Configure RTT    //配置 RTT
ConfigureRtt();

// Initialize state machine
state = STATE_MAINMENU;
alarmed = 0;        //报警标志位
RefreshDisplay(); //刷新显示

// User input loop
while (1) {

    // Wait for user input
    c = DBGU_GetChar();    //等待用户输入字符

    // Main menu mode     //主菜单模式
    if (state == STATE_MAINMENU) {

        // Reset timer       //重启定时器
        if (c == 'r') {

            ConfigureRtt();
            RefreshDisplay();
        }
        // Set alarm         //设置报警
        else if (c == 's') {

            state = STATE_SETALARM;
            newAlarm = 0;
            RefreshDisplay();
        }
        // Clear alarm       //清除报警
        else if ((c == 'c') && alarmed) {

            alarmed = 0;
            RefreshDisplay();
        }
    }
```

```
            // Set alarm mode          //报警模式
            else if (state == STATE_SETALARM) {

                // Number               //'字符'0'-'9 转换为数字 0-9
                if ((c >= '0') && (c <= '9')) {

                    newAlarm = newAlarm * 10 + c - '0';
                    RefreshDisplay();
                }
                // Backspace          //输入为退格
                else if (c == 8) {

                    newAlarm /= 10;
                    RefreshDisplay();
                }
                // Enter key          ////输入为回车
                else if (c == 13) {

                    // Avoid newAlarm = 0 case
                    if (newAlarm != 0) {

                        RTT_SetAlarm(AT91C_BASE_RTTC, newAlarm);
                    }

                    state = STATE_MAINMENU;
                    RefreshDisplay();
                }
            }
        }
}
```

2.3.2 重要子函数代码注释解读

上面是对 main 主函数的代码解读，下面是对 mian 函数里几个重要的子函数进行代码解读：

RTT 配置子函数 ConfigureRtt()：

```
void ConfigureRtt(void)
{
    unsigned int previousTime;     //设置实时定时器预分频数值

    // Configure RTT for a 1 second tick interrupt      //配置 RTT 为 1s 中断
    RTT_SetPrescaler(AT91C_BASE_RTTC, 32768);
    previousTime = RTT_GetTime(AT91C_BASE_RTTC);
```

```
    while (previousTime == RTT_GetTime(AT91C_BASE_RTTC));

    // Enable RTT interrupt
    AIC_ConfigureIT(AT91C_ID_SYS, 0, ISR_Rtt);
    AIC_EnableIT(AT91C_ID_SYS);
    RTT_EnableIT(AT91C_BASE_RTTC, AT91C_RTTC_RTTINCIEN);
}
```

RefreshDisplay()屏幕刷新子函数：

```
void RefreshDisplay(void)
{
    printf("%c[2J\r", 27);
    printf("Time: %u\n\r", RTT_GetTime(AT91C_BASE_RTTC));

    // Display alarm        //显示 ALARM
    if (alarmed) {

        printf("!!! ALARM !!!\n\r");
    }

    // Main menu
    if (state == STATE_MAINMENU) {

        printf("Menu:\n\r");
        printf(" r - Reset timer\n\r");
        printf(" s - Set alarm\n\r");
        if (alarmed) {                    //如果设置报警标志位，显示 c - Clear alarm notification 菜单

            printf(" c - Clear alarm notification\n\r");
        }
        printf("\n\rChoice? ");          //显示“Choice?”菜单
    }
    // Set alarm
    else if (state == STATE_SETALARM) {

        printf("Enter alarm time: ");
        if (newAlarm != 0) {

            printf("%u", newAlarm);
        }
    }
}
```

2.2.3 运行结果

代码在终端上运行的结果如下：
开始菜单

```
Time: 54
Menu:
 r - Reset timer
 s - Set alarm

Choice?
```

设置 20s 报警后的菜单

```
Time: 26
!!! ALARM !!!
Menu:
 r - Reset timer
 s - Set alarm
 c - Clear alarm notification

Choice?
```