深 圳 大 学 实 验 报 告

课程名称 单片机与嵌入式系统

作业名称 第一章任务实验报告

学　　院 电子与信息工程学院

专　　业 微电子

指导教师 潘志铭　吴国城

报 告 人 唐启斌　学号 2019285069

实验时间 2021.4.15

提交时间 2021.4.20

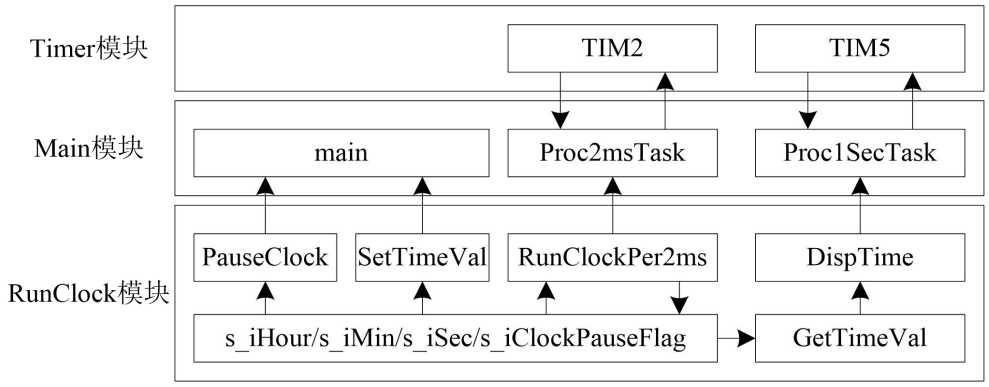
一、实验目的与要求

（1）将 RunClock 模块添加至 STM32 工程，并在应用层调用 RunClock 模块的 API 函数，实现基于 STM32 串口的电子钟功能；
（2）将时钟的初始值设置为 23:59:50，通过计算机上的串口助手每秒输出一次时间值，格式为 Now is xx:xx::xx；
（3）将编译生成的.hex 或.axf 文件下载到 STM32 核心板上；
（4）打开串口助手软件，查看电子钟运行是否正常。

二、实验原理

1. RunClock 模块有 6 个接口函数，分别是 InitRunClock、RunClockPer2Ms、PauseClock、GetTimeVal、SetTimeVal、DispTime。

2. 函数调用框架：



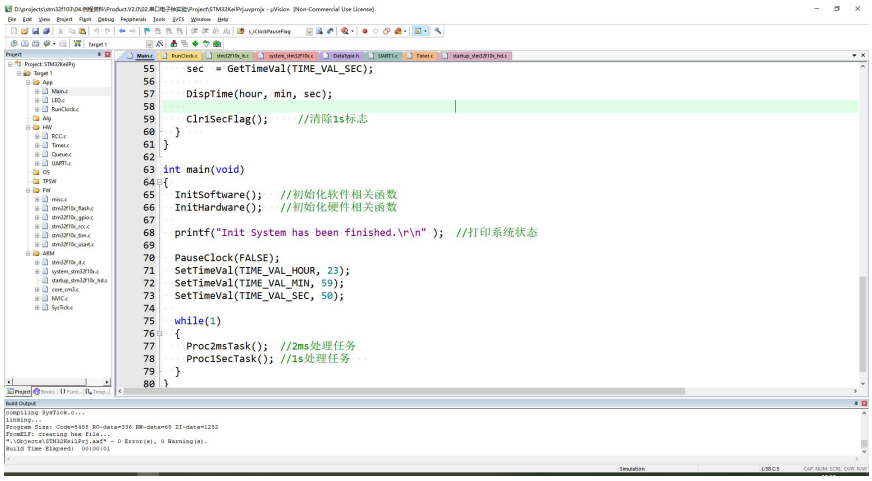
3. Proc2msTask 和 Proc1SecTask：

```
1.  static  void  Proc2msTask(void)
2.  {
3.    if(Get2msFlag())  //检查 2ms 标志状态
4.    {
5.      //用户代码，此处代码 2ms 执行一次
6.      Clr2msFlag();   //清除 2ms 标志
7.    }
8.  }
9.  static  void  Proc1SecTask(void)
10. {
11.   if(Get1SecFlag())  //检查 1s 标志状态
12.   {
13.     //用户代码，此处代码 1s 执行一次
14.     Clr1SecFlag();   //清除 1s 标志
15.   }
16. }
```

三、实验步骤

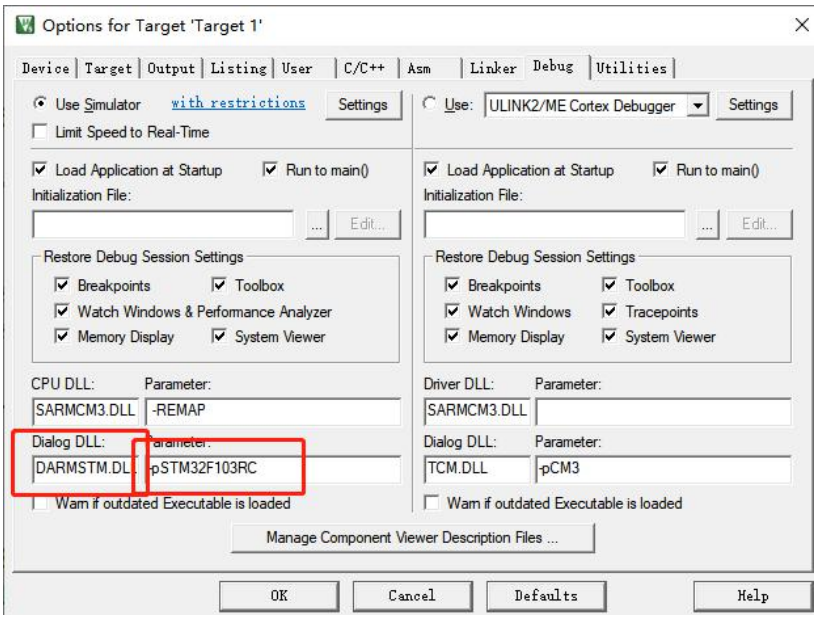
1、下载并安装最新版 Keil。

2、导入项目：



```
  sec  = GetTimeVal(TIME_VAL_SEC);

  DispTime(hour, min, sec);

  Clr1SecFlag();     //清除1s标志
}

int main(void)
{
  InitSoftware();   //初始化软件相关函数
  InitHardware();   //初始化硬件相关函数

  printf("Init System has been finished.\r\n" );  //打印系统状态

  PauseClock(FALSE);
  SetTimeVal(TIME_VAL_HOUR, 23);
  SetTimeVal(TIME_VAL_MIN, 59);
  SetTimeVal(TIME_VAL_SEC, 50);

  while(1)
  {
    Proc2msTask();  //2ms处理任务
    Proc1SecTask(); //1s处理任务
  }
}
```

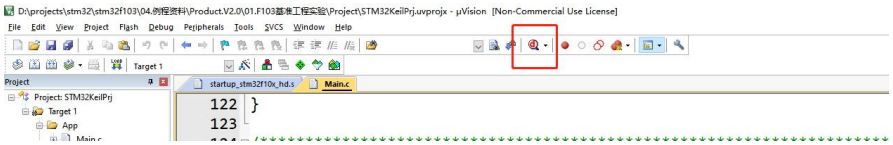
3、本次使用 Keil 的仿真检验实验结果，故须如下设置：（注意，最新版的 Keil 仿真时的晶振只能通过命令行设置）



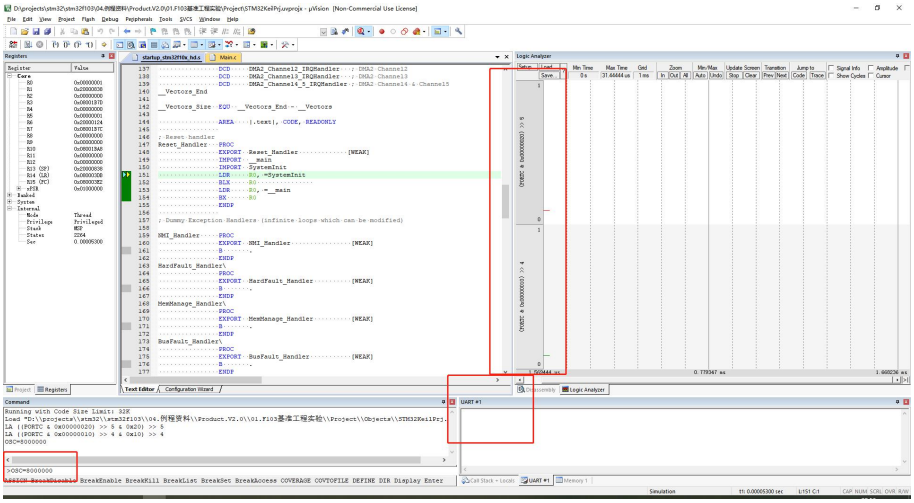
4、通过编译：

```
compiling core_cm3.c...
compiling system_stm32f10x.c...
compiling stm32f10x_usart.c...
compiling NVIC.c...
compiling SysTick.c...
linking...
Program Size: Code=6732 RO-data=348 RW-data=68 ZI-data=2108
FromELF: creating hex file...
".\Objects\STM32KeilPrj.axf" - 0 Error(s), 0 Warning(s).
Build Time Elapsed:  00:00:01
```

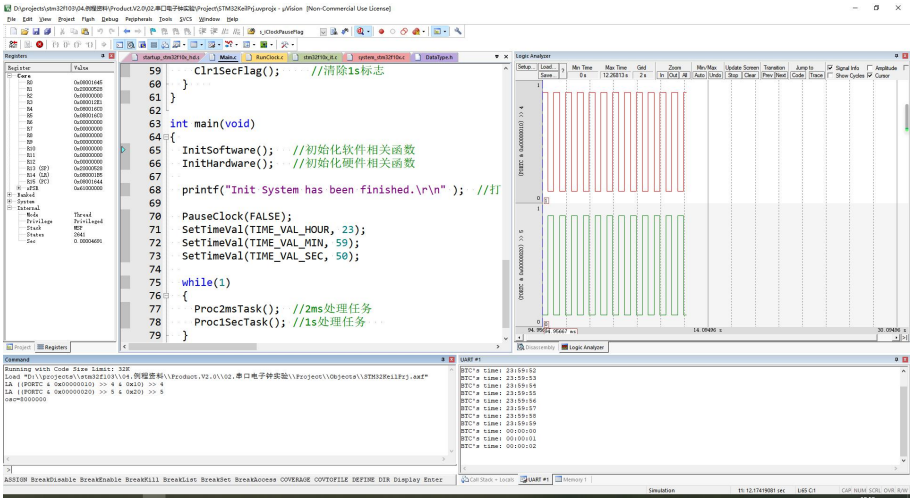
5、点击下图按钮开始调试模式：



5、通过如下命令设置晶振频率为 8M，并调出逻辑分析仪设置相应的端口和 UART#1 输出台：



9、开始调试，输出如下波形和文字说明项目成功运行：



四、实践感悟或疑惑

无

五、本章习题

1. Proc2msTask 函数的核心语句模块如何实现每 2ms 执行一次？

由如下 main 函数代码可知，本例中的程序在初始完相关变量后直接进入 while 死循环，不断阻塞调用 Proc2msTask 和 Proc1SecTask 函数：

```c
int main(void)
{
  InitSoftware();
  InitHardware();

  printf("Init System has been finished.\r\n" );

  PauseClock(FALSE);
  SetTimeVal(TIME_VAL_HOUR, 23);
  SetTimeVal(TIME_VAL_MIN, 59);
  SetTimeVal(TIME_VAL_SEC, 50);

  while(1)
  {
    Proc2msTask();
    Proc1SecTask();
  }
}
```

而在 Proc2msTask 函数定义中，很显然是通过 Get2msFlag 的返回值决定是否执行的：

```c
static  void  Proc2msTask(void)
{
  if(Get2msFlag())
  {
    RunClockPer2Ms();

    LEDFlicker(250);
    Clr2msFlag();
  }
}
```

跳转到 Time.c 可以看到 Get2msFlag 的实现：

```c
u8  Get2msFlag(void)
{
  return(s_i2msFlag);
}
```

而 s_i2msFlag 标志由定时器 2 定时更改：

```c
void TIM2_IRQHandler(void)
{
  static  u16 s_iCnt2 = 0;

```

```
    if(TIM_GetITStatus(TIM2, TIM_IT_Update) == SET)
    {
      TIM_ClearITPendingBit(TIM2, TIM_FLAG_Update);
    }

    s_iCnt2++;

    if(s_iCnt2 >= 2)
    {
      s_iCnt2 = 0;
      s_i2msFlag = TRUE;
    }
}
```

而定时器 2 在 InitTimer 函数中通过 ConfigTimer2(999, 71)配置：

```
void InitTimer(void)
{
  ConfigTimer2(999, 71);
  ConfigTimer5(999, 71);
}
```

2. Proc1SecTask 函数的核心语句块如何实现每秒执行一次？

与 Proc2msTask 逻辑相同，具体是使用 Timer4，并设置定时器不同的初值（定时器的本质是对晶振进行计数，设置不同初始值后，计数到置顶的值花费的时间不同...)。

3. PauseClock 函数如何实现电子钟的运行和暂停？

PauseClock 在 RunClock.c 中实现，仅仅只是将 s_iClockPauseFlag 赋一个新值：

```
void  PauseClock(u8 flag)
{
  s_iClockPauseFlag = flag;
}
```

然后运行电子钟的函数，即 RunClockPer2Ms 根据 s_iClockPauseFlag 决定是否执行：

```
void  RunClockPer2Ms(void)
{
```

```
3.    static i16 s_iCnt500 = 0;
4.
5.    if(499 <= s_iCnt500 && 0 == s_iClockPauseFlag)
6.    {
7.      if(59 <= s_iSec)
8.      {
9.        if(59 <= s_iMin)
10.       {
11.         if(23 <= s_iHour)
12.         {
13.           s_iHour = 0;
14.         }
15.         else
16.         {
17.           s_iHour++;
18.         }
19.         s_iMin = 0;
20.       }
21.       else
22.       {
23.         s_iMin++;
24.       }
25.       s_iSec = 0;
26.     }
27.     else
28.     {
29.       s_iSec++;
30.     }
31.     s_iCnt500 = 0;
32.   }
33.   else
34.   {
35.     s_iCnt500++;
36.   }
37. }
```

4. RunClockPer2Ms 函数为什么要每 2ms 执行一次？

3ms 执行一次也行，4ms 执行一次也行，只要你更改晶振频率或者更改定时器设置或者更改 RunClockPer2Ms 里的代码实现。