**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** |

# 实验目的与要求

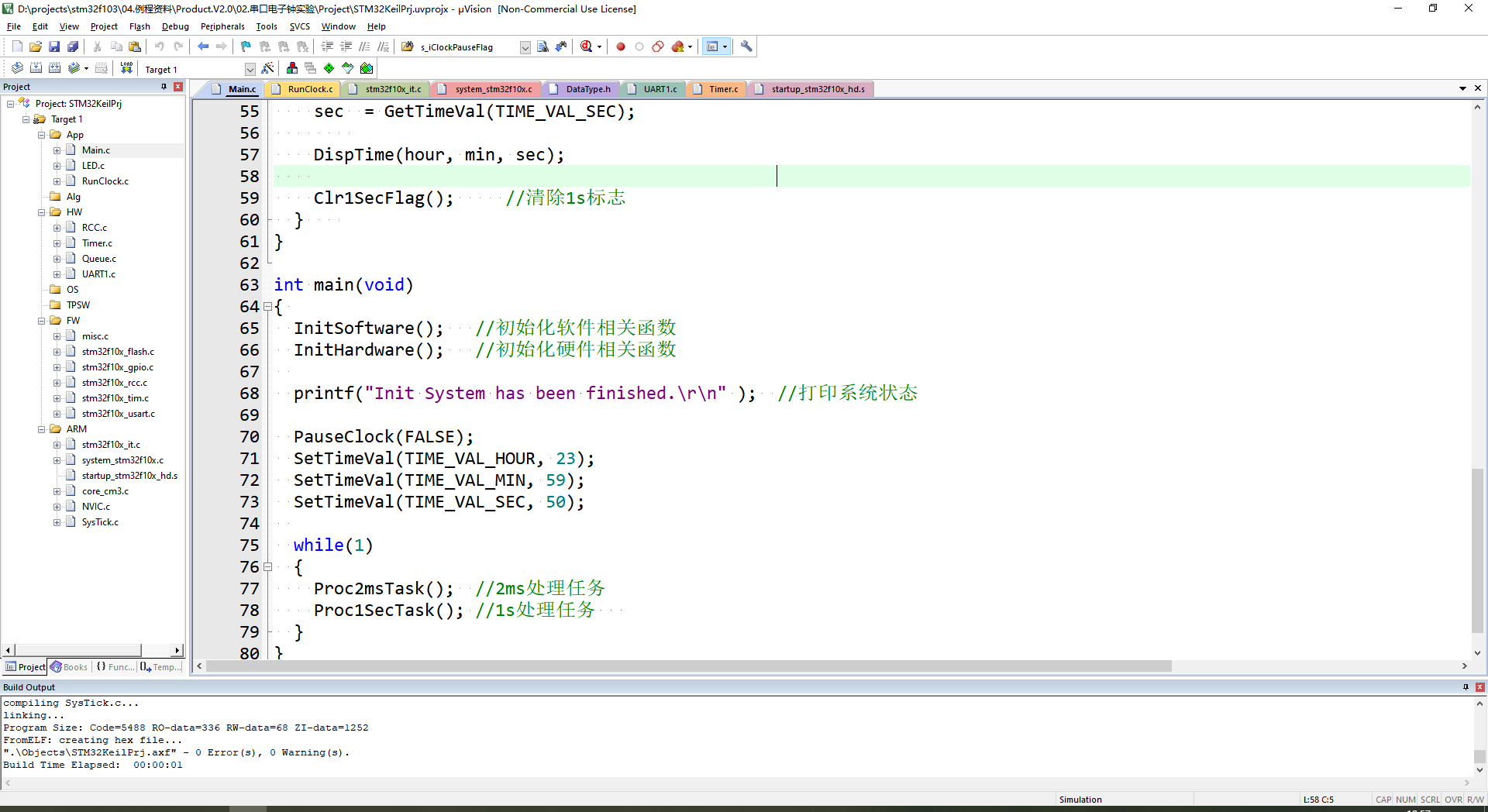
1. 将RunClock模块添加至STM32工程，并在应用层调用RunClock模块的API函数，实现基于STM32串口的电子钟功能；
2. 将时钟的初始值设置为23:59:50，通过计算机上的串口助手每秒输出一次时间值，格式为Now is xx:xx::xx；
3. 将编译生成的.hex或.axf文件下载到STM32核心板上；
4. 打开串口助手软件，查看电子钟运行是否正常。

# 实验原理

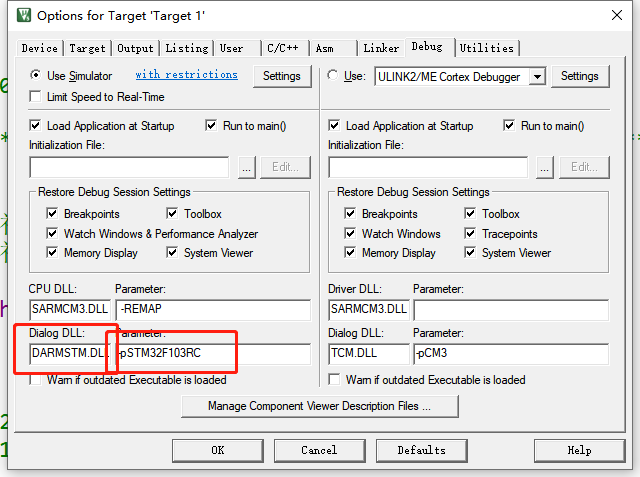
1. RunClock模块有6个接口函数，分别是InitRunClock、RunClockPer2Ms、PauseClock、GetTimeVal、SetTimeVal、DispTime。
2. 函数调用框架：



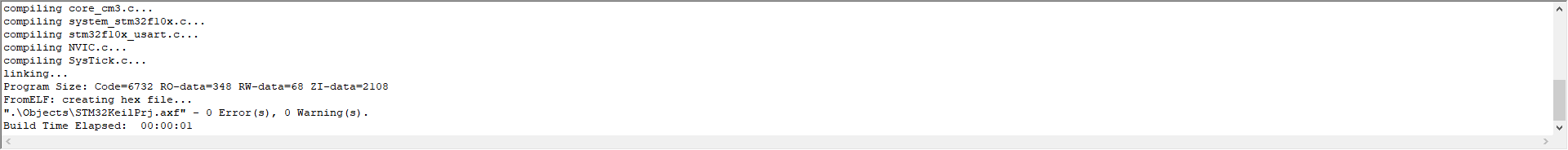
1. Proc2msTask和Proc1SecTask：
2. static  void  Proc2msTask(void)
3. {
4. if(Get2msFlag())  *//检查2ms标志状态*
5. {
6. *//用户代码，此处代码2ms执行一次*
7. Clr2msFlag();   *//清除2ms标志*
8. }
9. }
10. static  void  Proc1SecTask(void)
11. {
12. if(Get1SecFlag())  *//检查1s标志状态*
13. {
14. *//用户代码，此处代码1s执行一次*
15. Clr1SecFlag();   *//清除1s标志*
16. }
17. }
18. **实验步骤**
19. 下载并安装最新版Keil。
20. 导入项目：



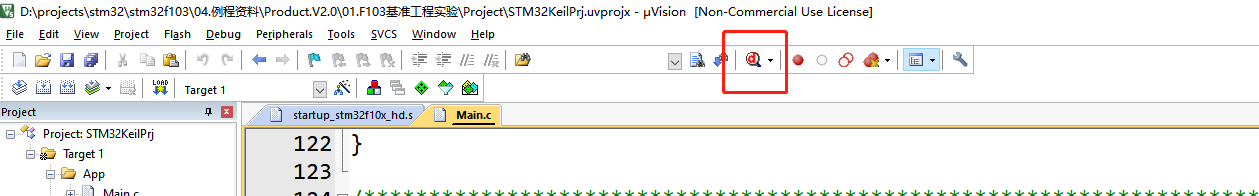
1. 本次使用Keil的仿真检验实验结果，故须如下设置：（注意，最新版的Keil仿真时的晶振只能通过命令行设置）



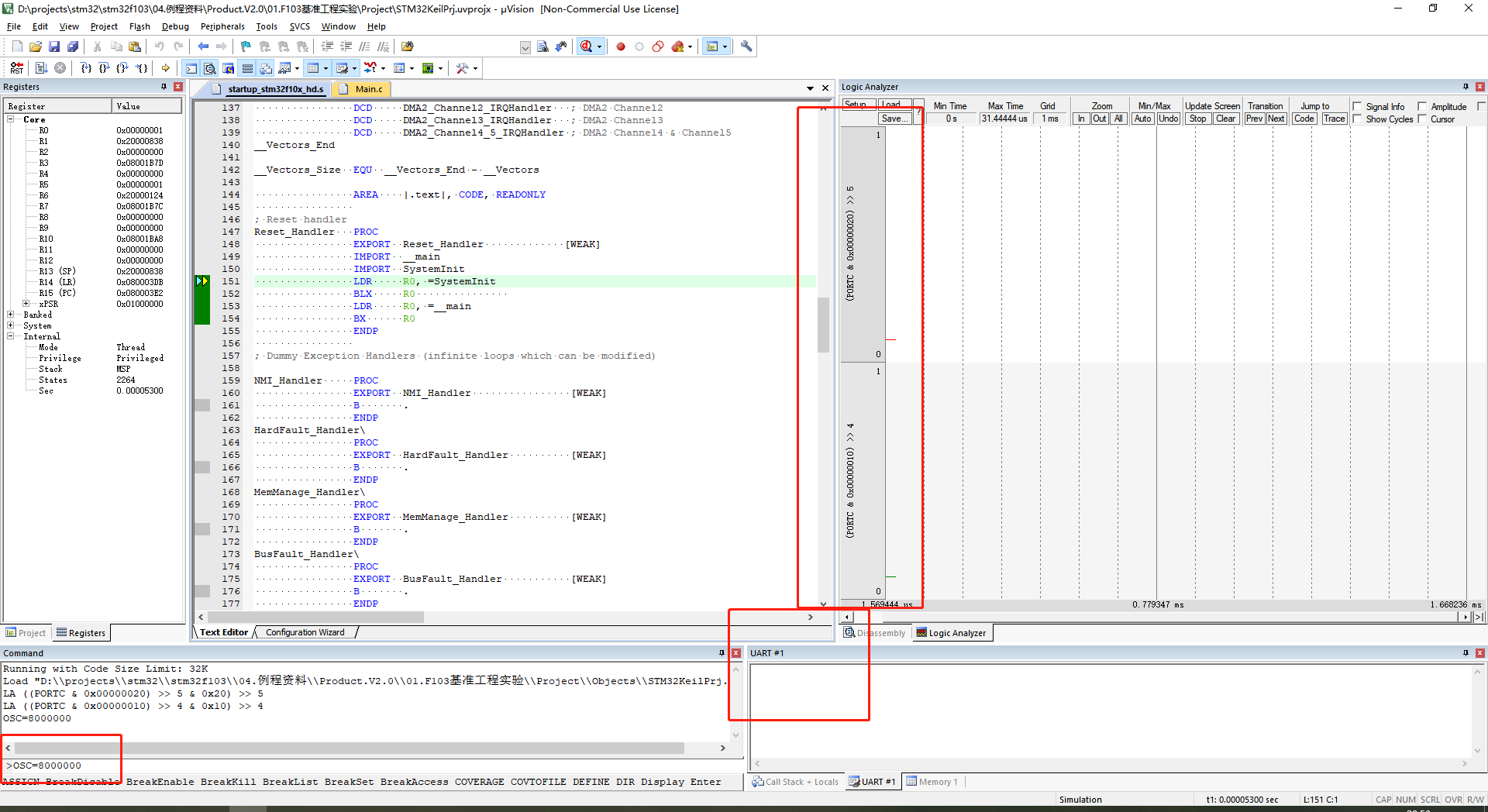
1. 通过编译：



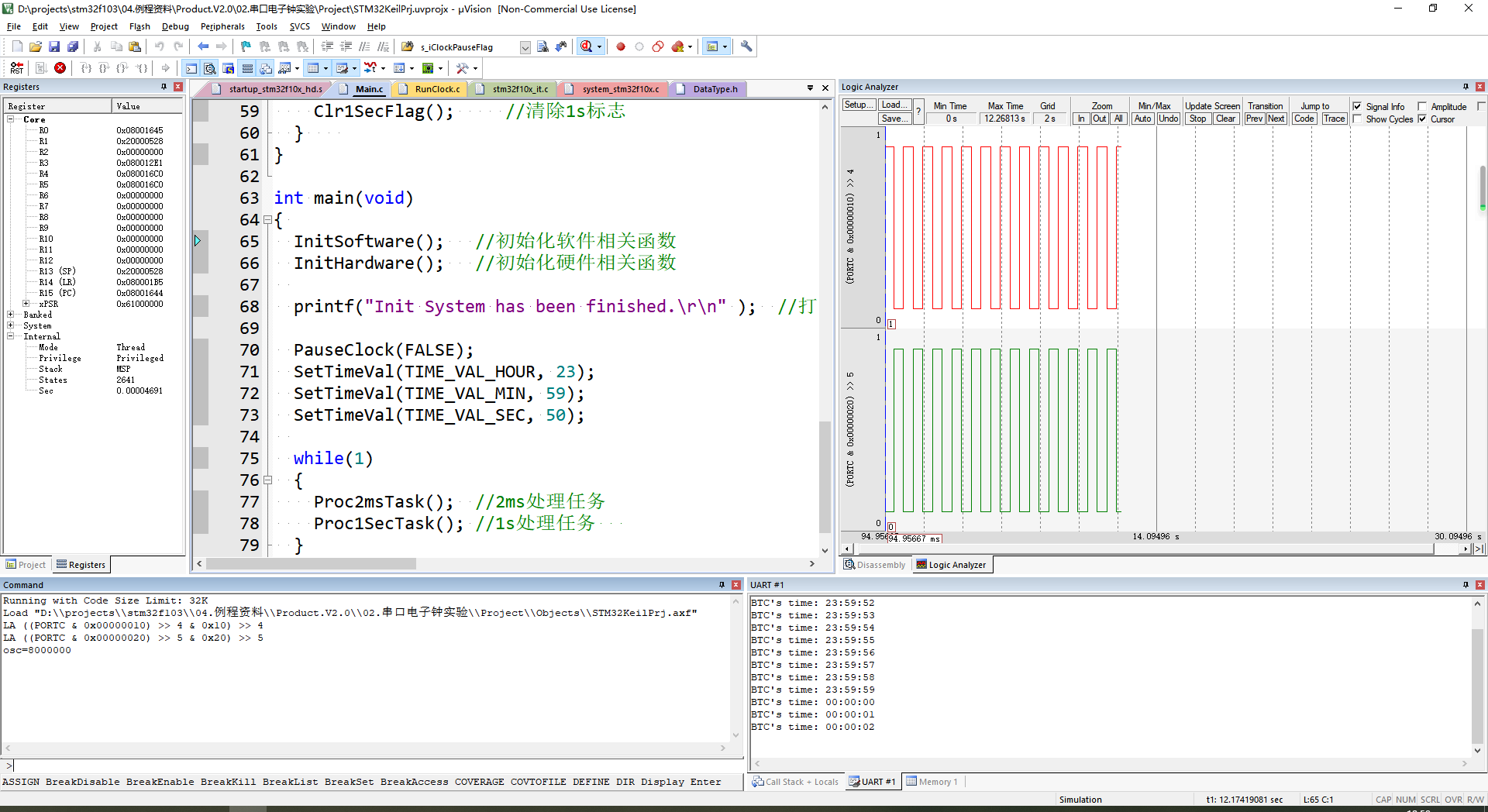
5、点击下图按钮开始调试模式：



1. 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：



1. 开始调试，输出如下波形和文字说明项目成功运行：



1. **实践感悟或疑惑**

无

1. **本章习题**
2. **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：

1. int main(void)
2. {
3. InitSoftware();
4. InitHardware();
6. printf("Init System has been finished.\r\n" );
7. PauseClock(FALSE);
8. SetTimeVal(TIME\_VAL\_HOUR, 23);
9. SetTimeVal(TIME\_VAL\_MIN, 59);
10. SetTimeVal(TIME\_VAL\_SEC, 50);
12. while(1)
13. {
14. Proc2msTask();
15. Proc1SecTask();
16. }
17. }

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：

1. static  void  Proc2msTask(void)
2. {
3. if(Get2msFlag())
4. {
5. RunClockPer2Ms();
7. LEDFlicker(250);
8. Clr2msFlag();
9. }
10. }

跳转到Time.c可以看到Get2msFlag的实现：

1. u8  Get2msFlag(void)
2. {
3. return(s\_i2msFlag);
4. }

而s\_i2msFlag标志由定时器2定时更改：

1. void TIM2\_IRQHandler(void)
2. {
3. static  u16 s\_iCnt2 = 0;
4. if(TIM\_GetITStatus(TIM2, TIM\_IT\_Update) == SET)
5. {
6. TIM\_ClearITPendingBit(TIM2, TIM\_FLAG\_Update);
7. }
8. s\_iCnt2++;
10. if(s\_iCnt2 >= 2)
11. {
12. s\_iCnt2 = 0;
13. s\_i2msFlag = TRUE;
14. }
15. }

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

1. void InitTimer(void)
2. {
3. ConfigTimer2(999, 71);
4. ConfigTimer5(999, 71);
5. }
6. **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...)。

1. **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：

1. void  PauseClock(u8 flag)
2. {
3. s\_iClockPauseFlag = flag;
4. }

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

1. void  RunClockPer2Ms(void)
2. {
3. static i16 s\_iCnt500 = 0;
4. if(499 <= s\_iCnt500 && 0 == s\_iClockPauseFlag)
5. {
6. if(59 <= s\_iSec)
7. {
8. if(59 <= s\_iMin)
9. {
10. if(23 <= s\_iHour)
11. {
12. s\_iHour = 0;
13. }
14. else
15. {
16. s\_iHour++;
17. }
18. s\_iMin = 0;
19. }
20. else
21. {
22. s\_iMin++;
23. }
24. s\_iSec = 0;
25. }
26. else
27. {
28. s\_iSec++;
29. }
30. s\_iCnt500 = 0;
31. }
32. else
33. {
34. s\_iCnt500++;
35. }
36. }
37. **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。