第 3 章　实验 2——串口电子钟

通过第 2 章的学习，初步掌握了 STM32 工程的创建、编译和下载验证方法。本章将通过串口电子钟实验，带领读者进入微控制器程序设计的世界。

3.1　实 验 内 容

本实验主要包括以下内容：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；④打开串口助手软件，查看电子钟运行是否正常。

3.2　实 验 原 理

3.2.1　RunClock 模块函数

RunClock 模块由 RunClock.h 和 RunClock.c 文件实现，这两个文件位于本书配套资料包的“04.例程资料\Material\02.串口电子钟实验\App\RunClock”文件夹中。RunClock 模块有 6 个 API 函数，分别是 InitRunClock、RunClockPer2Ms、PauseClock、GetTimeVal、SetTimeVal 和 DispTime，下面对这 6 个 API 函数进行讲解。

1．InitRunClock

InitRunClock 函数的功能是初始化 RunClock 模块，通过对 s_iHour、s_iMin 和 s_iSec 共 3 个内部变量赋值 0 来实现。该函数的描述如表 3-1 所示。

表 3-1　InitRunClock 函数的描述

函数名	InitRunClock
函数原型	void InitRunClock(void)
功能描述	初始化 RunClock 模块
输入参数	void
输出参数	无
返回值	void

2．RunClockPer2Ms

RunClockPer2Ms 函数的功能是以 2ms 为最小单位运行时钟系统，该函数每执行 500 次，

变量 s_iSec 递增一次。该函数的描述如表 3-2 所示。

表 3-2　RunClockPer2Ms 函数的描述

函数名	RunClockPer2Ms
函数原型	void RunClockPer2Ms(void)
功能描述	时钟计数，每 2ms 调用一次
输入参数	void
输出参数	无
返回值	void

3．PauseClock

PauseClock 函数的功能是启动和暂停时钟。该函数的描述如表 3-3 所示。

表 3-3　PauseClock 函数的描述

函数名	PauseClock
函数原型	void PauseClock(u8 flag)
功能描述	实现时钟的启动和暂停
输入参数	flag：时钟启动或暂停标志位。1—暂停时钟；0—启动时钟
输出参数	无
返回值	void

例如，通过 PauseClock 函数暂停时钟运行的代码如下：

```
PauseClock(1);
```

4．GetTimeVal

GetTimeVal 函数的功能是获取当前时间值，时间值的类型由 type 决定。该函数的描述如表 3-4 所示。

表 3-4　GetTimeVal 函数的描述

函数名	GetTimeVal
函数原型	i16 GetTimeVal(u8 type)
功能描述	获取当前的时间值
输入参数	type：时间值的类型
输出参数	无
返回值	获取到的当前时间值（小时、分钟或秒），类型由参数 type 决定

例如，通过 GetTimeVal 函数获取当前时间值的代码如下：

```
u8 hour;
u8 min;
u8 sec;
hour = GetTimeVal(TIME_VAL_HOUR);
```

```
min = GetTimeVal(TIME_VAL_MIN);
sec = GetTimeVal(TIME_VAL_SEC);
```

5．SetTimeVal

SetTimeVal 函数的功能是根据参数 timeVal 设置当前的时间值，时间值的类型由 type 决定。该函数的描述如表 3-5 所示。

表 3-5　SetTimeVal 函数的描述

函数名	SetTimeVal
函数原型	void SetTimeVal(u8 type, i16 timeVal)
功能描述	设置当前的时间值
输入参数	type：时间值的类型；timeVal：要设置的时间值类型
输出参数	无
返回值	void

例如，通过 SetTimeVal 函数将当前时间设置为 23:59:50，代码如下：

```
SetTimeVal(TIME_VAL_HOUR, 23);
SetTimeVal(TIME_VAL_MIN, 59);
SetTimeVal(TIME_VAL_SEC, 50);
```

6．DispTime

DispTime 函数的功能是根据参数 hour、min 和 sec 显示当前的时间，通过 printf 函数来实现。该函数的描述如表 3-6 所示。

表 3-6　DispTime 函数的描述

函数名	DispTime
函数原型	void DispTime(i16 hour, i16 min, i16 sec)
功能描述	显示当前的时间
输入参数	hour：当前的小时值；min：当前的分钟值；sec：当前的秒值
输出参数	无
返回值	void

例如，当前时间是 23:59:50，通过 DispTime 函数显示当前时间，代码如下：

```
DispTime(23, 59, 50);
```

3.2.2　函数调用框架

图 3-1 为本实验的函数调用框架，Timer 模块的 TIM2 用于产生 2ms 标志，TIM5 用于产生 1s 标志，Main 模块通过获取和清除 2ms、1s 标志，实现 Proc2msTask 函数中的核心语

句块每 2ms 执行一次，Proc1SecTask 函数中的核心语句块每 1s 执行一次。Main 模块调用 RunClock 模块的 PauseClosk 函数启动时钟运行，通过 SetTimeVal 函数设置初始时间值；Proc2msTask 函数调用 RunClock 模块的 RunClockPer2ms 函数，实现 RunClock 模块内部静态变量 s_iHour/s_iMin/s_iSec 的计数功能，进而实现时钟的运行；时间显示是由 RunClock 模块的 DispTime 函数调用 printf 语句输出实现的，Proc1SecTask 函数每秒调用一次 DispTime 函数。

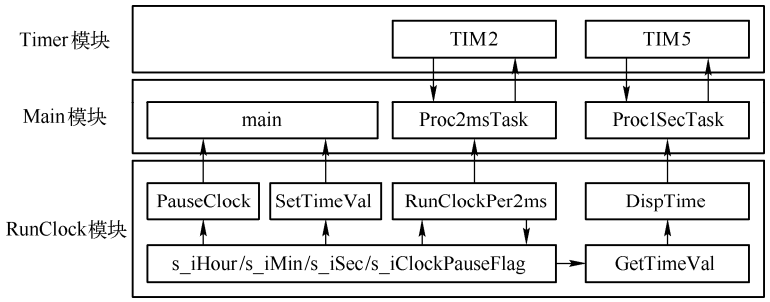

图 3-1　串口电子钟实验函数调用框架

3.2.3　Proc2msTask 与 Proc1SecTask

Proc2msTask 和 Proc1SecTask 是本书经常用到的函数，它们的工作机制类似，下面以 Proc2msTask 函数为例说明。程序清单 3-1 是 Proc2msTask 函数的实现，注意，需要每 2ms 执行一次的代码一定要放在 if 语句中。

程序清单 3-1

```
/*******************************************************************************************/
static   void   Proc2msTask(void)
{
  if(Get2msFlag())   //检查 2ms 标志状态
  {
    //用户代码，此处代码 2ms 执行一次
    Clr2msFlag();    //清除 2ms 标志
  }
}
```

Proc2msTask 函数在 main 函数的 while 语句中被调用，每隔几微秒执行一次，具体间隔取决于各中断服务函数及 Proc1SecTask 函数的执行时间。如果 Proc2msTask 函数约每 10μs 执行一次，Get2msFlag 函数用于读取 2ms 标志位的值并判断是否为 1，该标志位在 TIM2 的中断服务函数中被置为 1，TIM2 的中断服务函数每 2ms 执行一次，因此 2ms 标志位也是每 2ms 被置为 1 一次。如果 2ms 标志位为 1，则执行用户代码，执行完毕，清除 2ms 标志位，然后执行 Proc1SecTask 函数，接着继续判断 2ms 标志位；如果 2ms 标志位不为 1，则执行 Proc1SecTask 函数，然后继续判断 2ms 标志位。main 函数的 while 语句具体执行过程如图 3-2 所示。

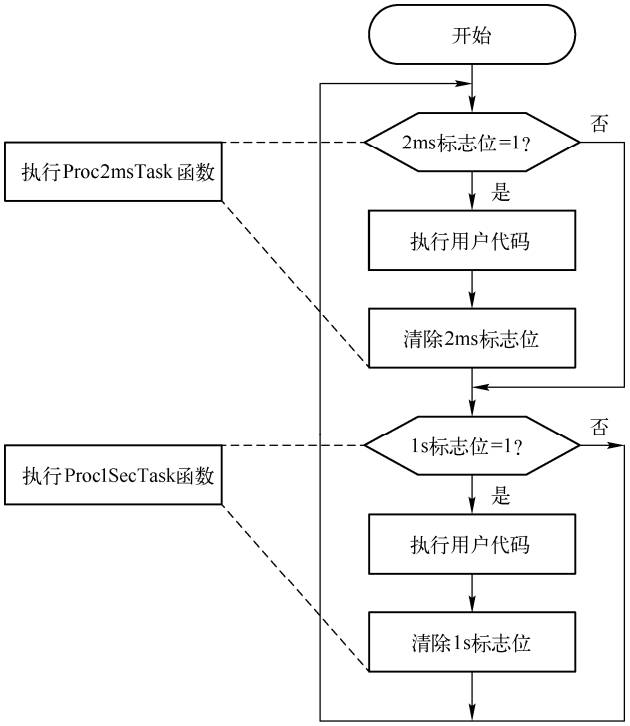

图 3-2　main 函数的 while 语句具体执行过程

3.3　实 验 步 骤

步骤 1：复制并编译原始工程

首先，将“D:\STM32KeilTest\Material\02.串口电子钟实验”文件夹复制到“D:\STM32KeilTest\Product”文件夹中。然后，双击运行“D:\STM32KeilTest\Product\02.串口电子钟实验\Project”文件夹中的 STM32KeilPrj.uvprojx，单击工具栏中的按钮，进行编译。当 Build Output 栏中出现 FromELF：creating hex file...时，表示已经成功生成.hex 文件，出现 0 Error(s), 0 Warning(s)表示编译成功。最后，将.axf 文件下载到 STM32 的内部 Flash，观察 STM32 核心板上的两个 LED 是否交替闪烁，同时打开串口助手，观察是否每秒输出一次 This is the first STM32F103 Project, by Zhangsan。如果两个 LED 交替闪烁、串口正常输出字符串，表示原始工程正确，可以进入下一步操作。

步骤 2：添加 RunClock 文件对

首先，将“D:\STM32KeilTest\Product\02.串口电子钟实验\App\RunClock”文件夹中的 RunClock.c 添加到 App 分组，具体操作可参见 2.3 节步骤 8。然后，将“D:\STM32KeilTest\Product\02.串口电子钟实验\App\RunClock”路径添加到 Include Paths 栏，具体操作可参见 2.3 节步骤 11。

步骤 3：完善串口电子钟应用层

在 Project 面板中，双击打开 Main.c 文件，在 Main.c 文件的“包含头文件”区的最后，

添加代码#include "RunClock.h"，如程序清单 3-2 所示。这样就可以在 Main.c 文件中调用 RunClock 模块的枚举定义和 API 函数等，实现对 RunClock 模块的操作。

程序清单 3-2

```
/*********************************************************************************************
*                                          包含头文件
*********************************************************************************************/
#include "Main.h"
#include "stm32f10x_conf.h"
#include "DataType.h"
#include "NVIC.h"
#include "SysTick.h"
#include "RCC.h"
#include "Timer.h"
#include "UART1.h"
#include "LED.h"
#include "RunClock.h"
```

在 Main.c 文件的 InitSoftware 函数中，添加调用 InitRunClock 函数的代码，如程序清单 3-3 所示，这样就实现了对 RunClock 模块的初始化。

程序清单 3-3

```
/*********************************************************************************************
* 函数名称：InitSoftware
* 函数功能：所有与软件相关的模块初始化函数都放在此函数中
* 输入参数：void
* 输出参数：void
* 返 回 值：void
* 创建日期：2018 年 01 月 01 日
* 注　　意：
*********************************************************************************************/
static  void  InitSoftware(void)
{
  InitRunClock();    //初始化 RunClock 模块
}
```

在 Main.c 文件的 Proc2msTask 函数中，添加调用 RunClockPer2Ms 函数的代码，如程序清单 3-4 所示。再次强调，一定要将调用 RunClockPer2Ms 函数的代码放在 if 语句中，这样才表示 RunClockPer2Ms 函数每 2ms 执行一次。

程序清单 3-4

```
/*********************************************************************************************
* 函数名称：Proc2msTask
* 函数功能：2ms 处理任务
* 输入参数：void
* 输出参数：void
```

```
* 返 回 值：void
* 创建日期：2018 年 01 月 01 日
* 注　　意：
*********************************************************************************************/
static  void  Proc2msTask(void)
{
  if(Get2msFlag())                    //判断 2ms 标志位状态
  {
    RunClockPer2Ms();                 //每 2ms 运行一次该函数

    LEDFlicker(250);                  //调用闪烁函数
    Clr2msFlag();                     //清除 2ms 标志位
  }
}
```

实验要求每秒输出一次时间，因此，需要在 Main.c 文件的 Proc1SecTask 函数中添加调用 DispTime 函数的代码。DispTime 函数的参数包括小时、分钟、秒，需要先定义 hour、min 和 sec 时间值变量，然后通过 GetTimeVal 函数获取这 3 个时间值，代码如程序清单 3-5 所示。这样即可实现每秒获取一次时间值（包括小时、分钟、秒），并通过 STM32 的串口发送到计算机的串口助手显示出来。由于 DispTime 函数是通过串口输出时间的，因此，需要注释掉 if 语句中的 printf 语句。

程序清单 3-5

```
/*********************************************************************************************
* 函数名称：Proc1SecTask
* 函数功能：1s 处理任务
* 输入参数：void
* 输出参数：void
* 返 回 值：void
* 创建日期：2018 年 01 月 01 日
* 注　　意：
*********************************************************************************************/
static  void  Proc1SecTask(void)
{
  i16 hour;
  i16 min;
  i16 sec;

  if(Get1SecFlag())      //判断 1s 标志位状态
  {
    //printf("This is the first STM32F103 Project, by Zhangsan\r\n");
    hour = GetTimeVal(TIME_VAL_HOUR);
    min = GetTimeVal(TIME_VAL_MIN);
    sec = GetTimeVal(TIME_VAL_SEC);

    DispTime(hour, min, sec);
```

```
    Clr1SecFlag();        //清除 1s 标志位
  }
}
```

在 RunClock 模块的 API 函数中，添加调用 PauseClock 和 SetTimeVal 函数的代码，如程序清单 3-6 所示。PauseClock 函数用于启动和暂停时钟，SetTimeVal 函数用于设置初始时间值。下面根据实验要求，将初始时间设定为 23:59:50，然后通过 PauseClock 函数启动时钟。

程序清单 3-6

```
/*********************************************************************************************
* 函数名称：main
* 函数功能：主函数
* 输入参数：void
* 输出参数：void
* 返 回 值：int
* 创建日期：2018 年 01 月 01 日
* 注　　意：
*********************************************************************************************/
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
  }
}
```

步骤 4：编译及下载验证

代码编写完成后，单击按钮进行编译。编译结束后，Build Output 栏中出现 0 Error(s)，0 Warning(s)，表示编译成功。然后，参见图 2-33，通过 Keil μVision5 软件将.axf 文件下载到 STM32 核心板。下载完成后，打开串口助手，可以看到时间值每秒输出一次，格式为 Now is xx:xx:xx，如图 3-3 所示。同时，可以看到 STM32 核心板上的 LED1 和 LED2 交替闪烁，表示实验成功。

图 3-3　串口电子钟实验结果

本 章 任 务

2018 年共有 365 天，将 2018 年 1 月 1 日作为计数起点，即计数 1，将 2018 年 12 月 31 日作为计数终点，即计数 365。计数 1 代表“2018 年 1 月 1 日-星期一”，计数 10 代表“2018 年 1 月 10 日-星期三”。根据串口电子钟实验原理，基于 STM32 核心板设计一个实验，实现每秒计数递增一次，计数范围为 1～365，并通过 printf 每秒输出一次计数对应的年、月、日、星期，结果通过计算机上的串口助手显示。此外，可以设置日期的初始值，例如，将初始日期设置为“2018 年 1 月 10 日-星期三”，第 1 秒输出“2018 年 1 月 10 日-星期三”、第 2 秒输出“2018 年 1 月 11 日-星期四”，以此类推。

本 章 习 题

1．Proc2msTask 函数的核心语句块如何实现每 2ms 执行一次？
2．Proc1SecTask 函数的核心语句块如何实现每秒执行一次？
3．PauseClock 函数如何实现电子钟的运行和暂停？
4．RunClockPer2Ms 函数为什么要每 2ms 执行一次？