功能测试

1 测试平台

测试平台通常分为两类：硬平台和软平台。硬平台可以是项目所用的硬件平台，也可以是通用的开发板。软平台则是具有相同或相近的功能的仿真软件搭建的仿真验证系统。

通常在开发硬件同时也进行软件的开发和测试。由于硬件平台没有搭建好或其它原因暂时无法使用硬件，可以采用仿真软件进行功能开发和测试验证。

Proteus 软件是英国 Lab Center Electronics 公司出版的 EDA 工具软件。它不仅具有其它 EDA 工具软件的仿真功能，还能仿真微处理器及外围器件。它内部含有 STM32F401 器件，可以采用专用的 IDE 进行程序编制，并利用软件上提供的工具进行仿真运行调试。

通过软件仿真可以验证驱动程序是否正确，为上板调试消除程序中的漏洞和处理错误。

建立基于 STM32F401RE 的工程，如图 1 所示，选择自动生成启动文件及固件工程。仿真平台提供的 IDE 仅支持 GCC 编译器，所以需要安装相应的编译软件。也可以在 Keil 下编制，通过编译链接最后生成 HEX 文件，将 HEX 文件加载至模拟仿真系统中进行运行调试。

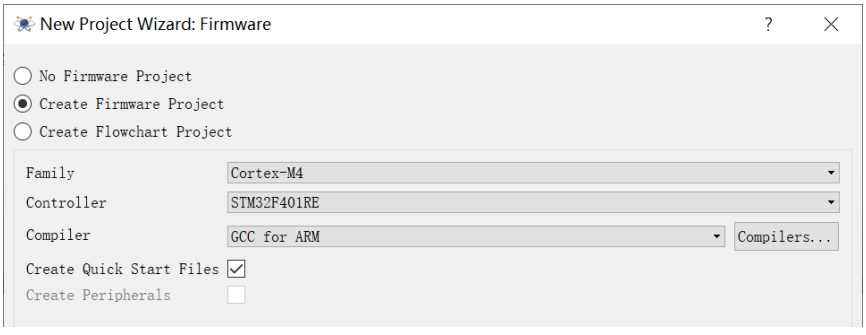

图 1 STM32F401RE 的 Proteus 工程

建立基于 STM32F401RE 的工程，选择自动生成启动文件及固件工程。仿真电路图如图 2 所示，主要进行功能仿真，因此串口没用 RS232 电平。采用仿真平台本身提供的串口终端、直流电机、数码管、热敏电阻和轻触开关。

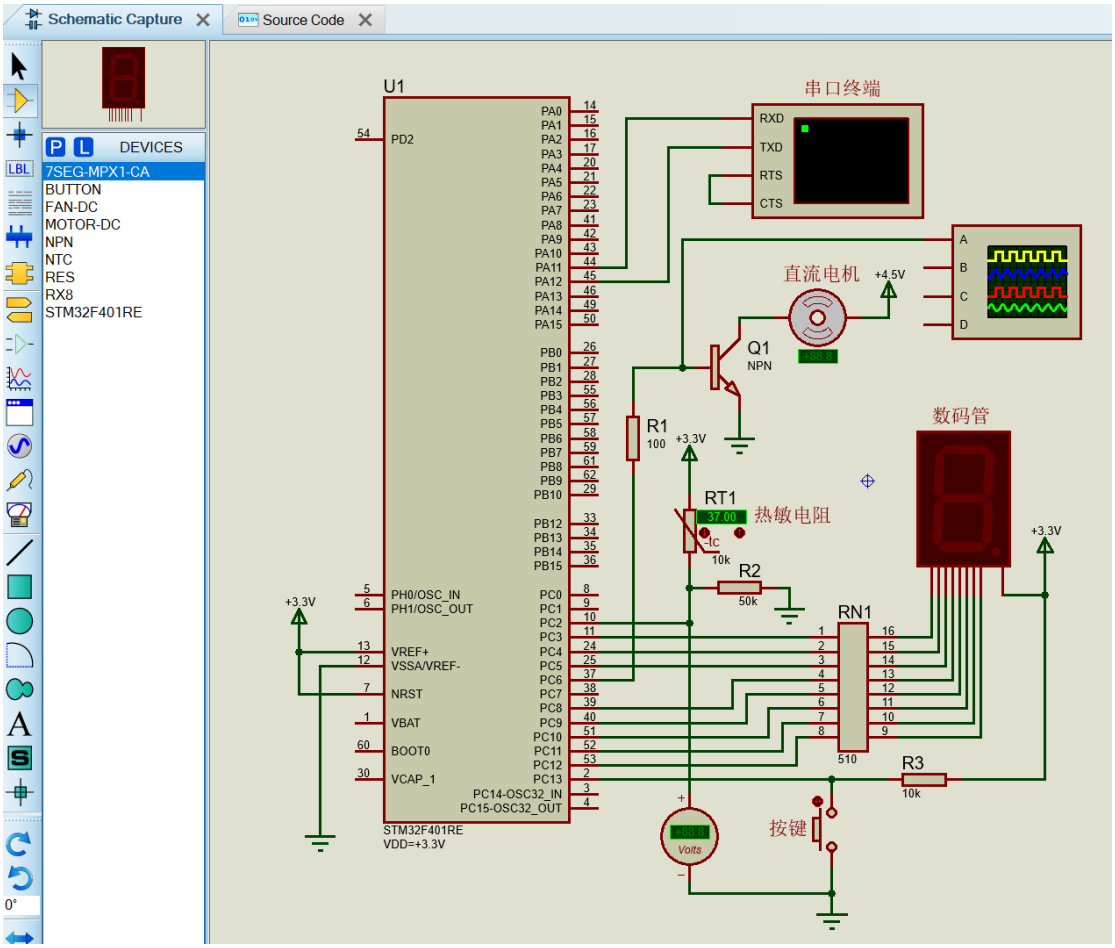

图 2　系统仿真电路图

串口终端提供串口波特率和位数等参数配置，并可以以字符或十六进制显示接收到的字符，也可以发送键盘输入的字符，可以选择是否回显。它可以模拟外部远端的串口收发设备进行数据通信。

直流电机提供工作电压、电感电阻及电感等参数配置，并显示正转或反转的速率。

热敏电阻提供热敏电阻参数配置、温度显示及手工调节功能，模拟温度变化。

轻触开关模拟轻触按钮进行显示控制。

工程中包含自动产生的各种头文件和部分启动源文件，通过新加或添加的方式将前面所编制的程序文件中加入工程中。

2 单功能模块测试

在主文件 main.c 中加入相应的单功能模块测试代码即可完成单项功能测试。

1. LED 驱动测试

实现延时一段时间指示灯亮灭切换一次，代码如下：

```
#include <stm32f4xx.h>
#include "segdisplay.h"
void test_led_drv()
{
    int i;
    LedInit();
    while(1)
    {
        for(i=0;i<2000000;i++);//延时
        LedSw();
    }
}
int main()
{
    test_led_drv();
    return 0;
}
```

2.数码显示测试

定时循环显示数值 0-9，代码如下：

```
#include <stm32f4xx.h>
#include "segdisplay.h"
void test_seg_display()
{
    unsigned char ch;
    int i;
    SegInit();
    while(1)
    {
        for(ch=0; ch<=9; ch++)
        {
            SegDisp(ch);
            for(i=0;i<2000000;i++);
        }
    }
}
int main()
{
    test_seg_display();
    return 0;
}
```

3.定时器 4 工作测试

利用定时器 4 中断控制 LED 的亮灭，代码如下：

```
#include <stm32f4xx.h>
#include "segdisplay.h"
#include "timer.h"
void test_tim4_int()
{
    LedInit();
    Tim4Init();
    Tim4IntEn(LedSw);
    while(1);
}
int main(void)
{
    test_tim4_int();
    return 0;
}
```

4.定时器 2 工作测试

利用定时器 2 中断控制 LED 的亮灭，代码如下：

```
#include <stm32f4xx.h>
#include "segdisplay.h"
#include "timer.h"
void test_tim2_int()
{
    LedInit();
    Tim2Init();
    Tim2IntEn(LedSw);
    while(1);
}
int main(void)
{
    test_tim2_int();
    return 0;
}
```

5.UART 波特率检测

在实际调试中，有时 UART 的波特率产生电路存在配置不成功或错误的情况，从而导致波特率不对，使对方无法正确接收。因此，在 UART 进行通信前，通常要进行波特率检测。

通过不断发送 0x55 产生高低电平交替的波形，测量任意一个高电平或低电

平的时长，计算其倒数即可得到波特率，代码如下：

```
#include <stm32f4xx.h>
#include "uart.h"

void test_uart_band()
{
    UartInit();
    while(1)
    {
    UartTx(0x55);
    }
}
int main(void)
{
    test_uart_band();
    return 0;
}
```

6.串口发送测试

延时循环发送‘0’~‘9’至终端，代码如下：

```
#include <stm32f4xx.h>
#include "uart.h"
void test_uart_tx()
{
    unsigned char ch;
    int i;
    UartInit();
    while(1)
    {
        for(ch='0'; ch<='9'; ch++)
        {
            while(!UartTx(ch));
            for(i=0;i<2000000;i++);
        }
    }
}
int main()
{
    test_uart_tx();
    return 0;
}
```

7.串口接收测试

将收到的终端发送的数据发回终端，代码如下：

```
#include <stm32f4xx.h>
#include "uart.h"
void test_uart_rx()
{
    unsigned char ch;
    UartInit();
    while(1)
    {
        while(!UartRx(&ch));
        while(!UartTx(ch));
    }
}
int main()
{
    test_uart_rx();
    return 0;
}
```

8.串口中断接收测试

将收到的终端发送的数据发回终端，代码如下：

```
#include <stm32f4xx.h>
#include "uart.h"
void uart_rx_proc( unsigned char ch )
{
    while(!UartTx(ch));
}
void test_uart_rx_int()
{
    UartInit();
    UartRxIntEn(uart_rx_proc);
    while(1);
}
int main()
{
    test_uart_rx_int();
    return 0;
}
```

9.按钮接口测试

按一下 LED 灯翻转一次，代码如下：

```
#include <stm32f4xx.h>
#include "segdisplay.h"
#include "button.h"
void btn_proc()
{
    LedSw();
}
void test_btn_int()
{
    LedInit();
    BtnInit();
    BtnIntEn(btn_proc);
    while(1);
}
int main()
{
    test_btn_int();
    return 0;
}
```

10.PWM 控制测试

延时一段时间改变速度，通过示波器发现 PWM 占空比变化或模拟工具发现转速变化，代码如下：

```
#include <stm32f4xx.h>
#include "motor.h"
void test_pwm_ctrl()
{
    int i, j;
    MotorInit();
    while(1)
    {
        for(i=0; i<9; i++)
        {
            MotorSpeedSet(i);
            for(j=0; j<1000000; j++);
        }
    }
}
int main()
{
    test_pwm_ctrl();
    return 0;
}
```

11.ADC 测量测试

采数据并通过串口发送到终端，调节外部电阻或仿真元件，终端显示的数值随调节变化而变化，代码如下：

```
#include <stm32f4xx.h>
#include "tempmeasure.h"
void test_temp_measure()
{
    int i;
    unsigned char temp;
    UartInit();
    TempMeasInit();
    while(1)
    {
        TempMeasRun();
        while(!TempMeasGet(&temp));
        while(!UartTx(temp));
        for(i=0; i<8000000; i++);
    }
}
int main()
{
    test_temp_measure();
    return 0;
}
```

3 系统功能模块测试

系统测试主要完成接口驱动联合测试：

①工作指示灯周期性闪亮切换；

②获取温度测量采样值后，通过串口把测量值发送给远程终端，并启下一次采样；

③串口接收远程终端发来的速度等级（数字），配置相应的速度，并根据显示允许状况确定是否在数码管上显示接收到的数字；

④按钮控制数码管是否显示当前的速度等级，是则灯亮显示，否则灯灭不显示。

在主文件 main.c 中加入测试程序所用的数据，采用结构体方式来定义测试所用的参数。

```
#include <stm32f4xx.h>
```

```
#include "segdisplay.h"
#include "uart.h"
#include "tempmeasure.h"
#include "timer.h"
#include "motorctrl.h"
#include "button.h"
typedef struct {
    unsigned char temp_curr,speed, disp_on;
} AutoCtrl;
AutoCtrl auto_ctrl={0,0,1};
void LedProc()
{
    LedSw();
}
void CmdProc(char cmd)
{
    if(cmd>='0' && cmd <='9' )
    {
        auto_ctrl.speed = cmd - '0';
        MotorSpeedSet(auto_ctrl.speed);
        if( auto_ctrl.disp_on )
            SegDisp(auto_ctrl.speed);
    }
}
void TempMeasProc( )
{
    if(TempMeasGet(&auto_ctrl.temp_curr))
    {
        TempMeasRun();
        UartTx(auto_ctrl.temp_curr);
    }
}
void BtnProc()
{
    if(auto_ctrl.disp_on)
    {
        auto_ctrl.disp_on=0;
        SegDisp(0xff);
    }
    else
    {
        auto_ctrl.disp_on=1;
        SegDisp(auto_ctrl.speed);
    }
```

```
}
int main(void)
{
    LedInit();
    SegInit();
    UartInit();
    TempMeasInit();
    Tim2Init();
    BtnInit();
    Tim4Init();
    MotorInit();
    UartRxIntEn(CmdProc);
    Tim2IntEn(TempMeasProc);
    BtnIntEn(BtnProc);
    Tim4IntEn(LedProc);
    SegDisp(auto_ctrl.speed);
    MotorSpeedSet(auto_ctrl.speed);
    TempMeasRun();
    while(1);
    return 0;
}
```

软件仿真功能正确后，再将程序用到工程所用的硬件平台上，根据实际测试微调一些参数即可完成硬件系统的功能测试。