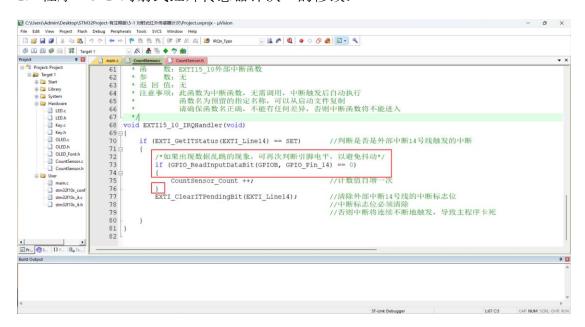
程序修改说明

本文件夹中包含了"STM32入门教程"视频里的全部程序源码,代码内容最初和视频里展示的完全一致。但是随后在实践过程中,笔者发现了代码中的一些小问题。为了保证代码的严谨,此文件夹最终提供给大家的程序源码,是笔者修改之后的版本,与视频中展示的代码有少许差异。此文档展示了这些差异,供大家参考。

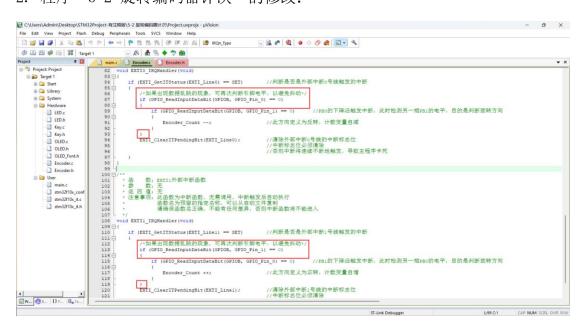
1. 程序"5-1 对射式红外传感器计次"的修改:



修改原因: 在笔者做视频的时候, 当时硬件产生的外部中断信号并没有抖动, 现象很完美。但是过了一段时间, 新购买的硬件在触发外部中断时, 会出现"给一次边沿触发多次中断"的情况, 换了几批货源均有这种情况。最终无奈, 只能通过程序来弥补这个问题了。经过笔者测试, 在进入中断后, 加入再次判断引脚电平的代码, 可以比较好地解决这个问题, 因此, 笔者对程序进行了上述修改。

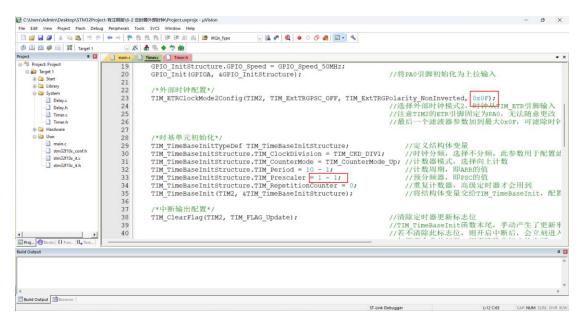
相关说明也可查看 "https://jiangxiekeji.com/problem/p1-8.html"。

2. 程序"5-2 旋转编码器计次"的修改:



修改原因:与程序"5-1 对射式红外传感器计次"的修改原因相同。相关说明也可查看"https://jiangxiekeji.com/problem/p1-8.html"。

3. 程序"6-2 定时器外部时钟"的修改:



修改原因:对射式红外传感器输出的信号有噪声,导致有时候出现"遮挡一次,计数多次"的现象。不过好在 STM32 定时器的外部时钟输入自带有滤波器,可以对信号进行过滤。所以,在上述代码中,笔者将 TIM_ETRClockMode2Config 的最后一个参数,即滤波器参数,修改为了 0x0F (视频中为 0x00),这样可以解决外部时钟信号抖动的问题。

另外,部分 STM32 芯片,给 TIM_Prescaler 赋值为"1 - 1"时,会造成整个程序卡死,具体原因不详。如果出现这个问题,可以把程序中的"1 - 1"改成"2 - 1"。

相关说明也可查看"https://jiangxiekeji.com/problem/p1-9.html"。

4. 程序"10-1 软件 I2C 读写 MPU6050"的修改:

```
File Edit View Project Flash Debug Peripherals Tools SVCS Window Help
                                                                                                                                                                                                                                                                 → II II → II Target 1
                                                                                                                        A = + 7 m
                                                                                                                 in.c Myl2Ce Myl2Ch MPU6050.c MPU6050.h MPU6050_Reg.h
                                                                                                                       void MyI2C_SendByte(uint8_t Byte)
                                                                                                                                                                                                                                                                                                   //循环8次, 主机依次发送数据的每一位
                                                                                                       101
                                                                                                                                                   /*两个!可以对数据进行两次逻辑取反,作用是把非0值统一转换为1,即:!!(0) = 0,!!(非0) = 1*/My12C_W_SDA(!!(Byte & \{0x80 >> i\}));//使用掩码的方式取出Byte的指定一位数据并写入到SDA线
                             Key.c

Key.h

OLED.c
                               OLED,H
                               OLED_Font.h
                             Myl2C.c

Myl2C.h

Mpu6050.c

Mpu6050.h

Mpu6050_Reg.l
                                                                                                                              * 函 数: 120接收一个字节
* 参 数: 无
* 返 回 值: 接收到的一个字节数据, 范围: 0x00-0xFF
                                                                                                                       uint8 t MyI2C ReceiveByte(void)
                                                                                                                                       uint8_t i, Byte = 0x00;
MyI2C_W SDA(1);
for (i = 0; i < 8; i ++)
                                                                                                                                                                                                                                                                                                 //定义接收的数据,并赎初值0x00,此处必须赋初值0x00,后面会用到
//接收前,主机先确保释放SDA,避免干扰从机的数据发送
//循环8次,主机依次接收数据的每一位
                                                                                                                                                   | MyI2C \( \text{ \te}
                                                                                                                                                                                                                                                                                                                                                                                                                                                             当SDA为0时,不做处理,指定位为默认的
                                                                                                                                                                                                                                                                                                   //返回接收到的一个字节数据
                                                                                                                                         return Byte;
■ Pro... | {} Fu... | {} U_• T
```

修改原因:原来的程序"MyI2C_W_SDA(Byte & (0x80 >> i))",函数参数的运算结果会出现大于1的值,比如Byte的最高位是1,则Byte & 0x80的结果为0x80,传递给MyI2C_W_SDA函数时,参数就是0x80。这样就要求MyI2C_W_SDA内部必须有非0即1的操作(参数传入非0值,置引脚为高电平,参数传入0,置引脚为低电平)。视频中GPIO_WriteBit函数内部有非0即1的操作代码,因此,视频里的代码并无Bug。但是,有些同学在移植这个代码时,并没有注意到这个问题,写出了非1即0的逻辑,导致程序出现Bug。

为了彻底避免这个问题,笔者对程序进行了修改,对括号内的运算结果加了两个!(逻辑取反),这样就能把传入函数的非0值全部转换为1,函数内,也只需要判断0和1两个值即可,不会出现大于1的值。

另外此处也可如下图所示修改,效果相同。if 判断时,有非 0 即真的特性。

```
98 void MyI2C SendByte (uint8 t Byte)
100
      uint8 t i;
                                         //循环8次,主机依次发送数据的每一位
101
       for (i = 0; i < 8; i ++)
102
          if (Byte & (0x80 >> i))
103
                                         //如果运算结果非0
                                         //给MyI2C W SDA函数直接传入1
105
              MyI2C W SDA(1);
106
107
                                         //否则
108
109
             MyI2C_W_SDA(0);
                                        //给MyI2C W SDA函数直接传入0
110
111
          MyI2C W SCL(1);
                                         //释放SCL, 从机在SCL高电平期间读取SDA
112
          MyI2C W SCL(0);
                                         //拉低SCL, 主机开始发送下一位数据
113
```

同理,在读取引脚电平时,原来的"if (MyI2C_R_SDA() == 1)"也被改成了"if (MyI2C_R_SDA())",自带一个非 0 即 1 的效果,避免后续出现可能的 Bug。

5. 程序"11-1 软件 SPI 读写 W25Q64"的修改:

```
| CLUMers/Admin/Debtogp/STML2Project-NIERRN/1-1 RESPIRENCE/SQR/Project-project projects from Debtog Prophenis Tools SVCS Window Medic
| Part | Part
```

修改原因: 与程序"10-1 软件 I2C 读写 MPU6050"的修改原因相同。

6. 程序 "OLED 驱动函数模块"的修改:

```
BC CLUbers Adminish Desktop (STAME Project - HELBEN A-1 OLIDE FREP Project Project File Helber Project File Helber Project From Debug Project Projec
```

修改原因: 与程序"10-1 软件 I2C 读写 MPU6050"的修改原因相同。