

从传感器到通信总线

2017寒假单片机培训

各种各样的传感器

- 获取外界信息
- “输入系统”的一部分
- 与控制器的两种基本通信方式
 - 模拟/数字量直接输入
 - 经由数据总线

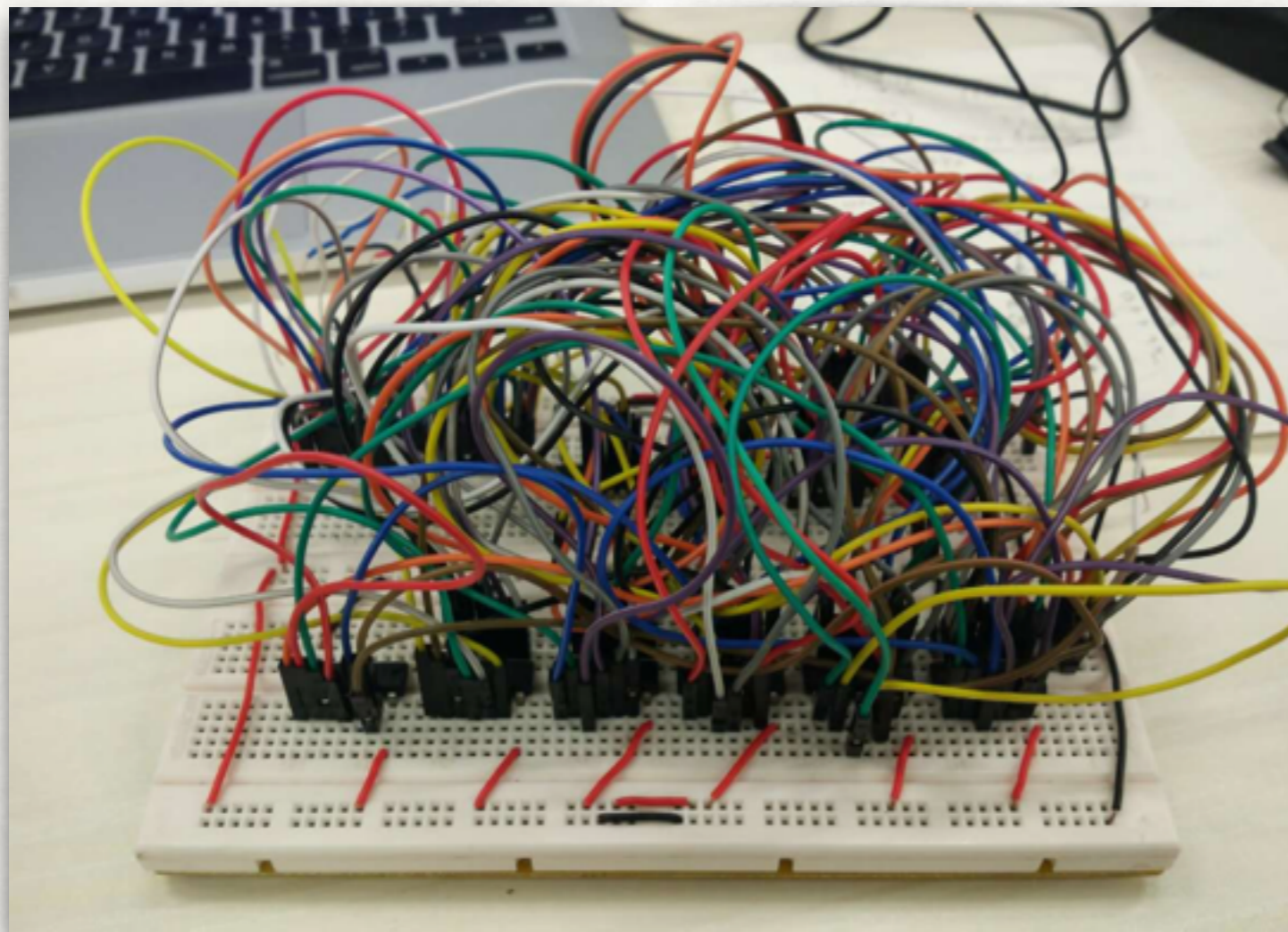


常见传感器及其接口

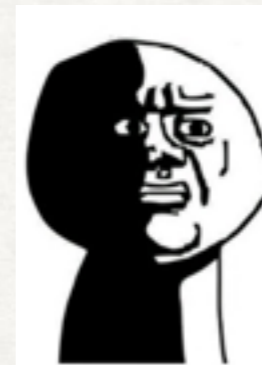
| 名称 | 接口 | 图片 |
|-------------|----------------------|---|
| MPU9250 | SPI/I2C |  |
| 超声波传感器（大眼睛） | 数字开关量输入（脉宽） |  |
| 湿度传感器 | 单工私有协议 |  |
| 雨量传感器 | 模拟（雨量）/开关量（下雨） 输入 |  |
| 光电传感器 | 开关量输入 |  |

如何上手?

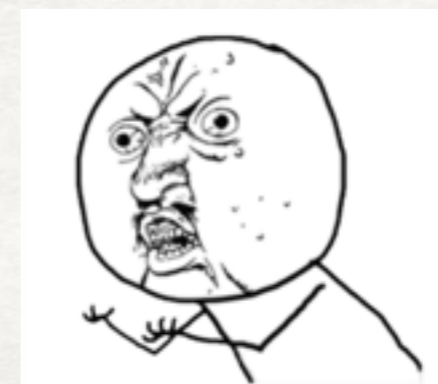
- 拿到一款传感器后，主要应当关注以下三个方面：



... ? 开关量输入?



你TM在逗我



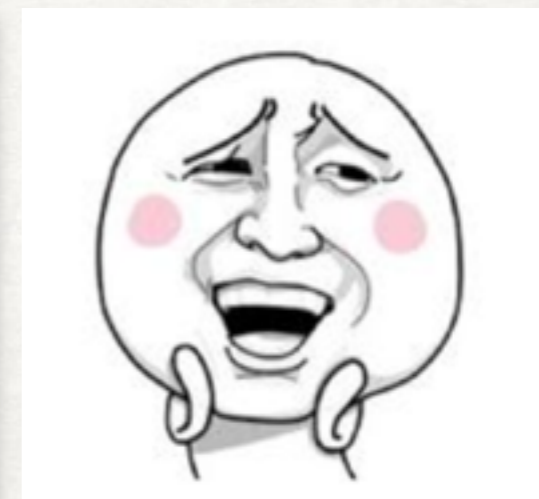
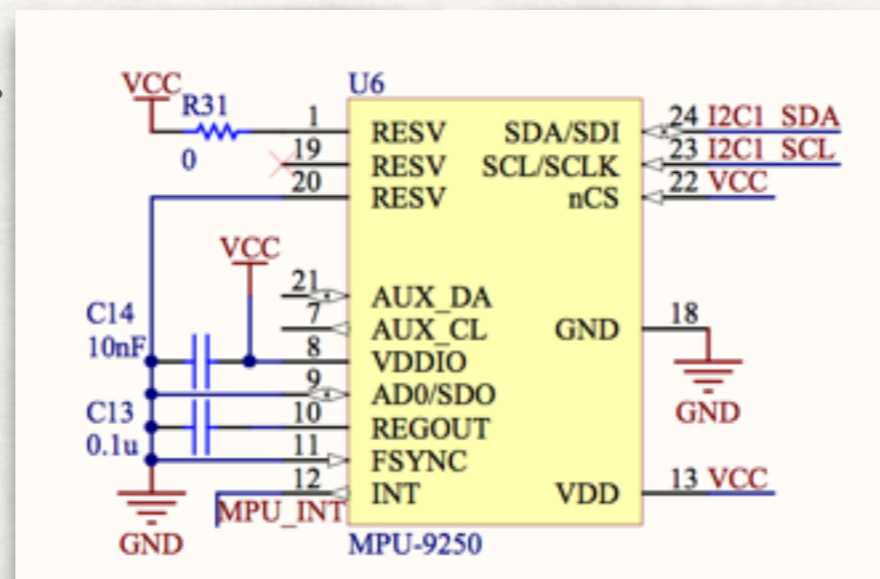
通信接口的使用

兵来将挡

- 充分单片机的外设资源解决传感器通信问题
 - 模拟输入->输入电平的高低携带信息->使用 ADC 采样
 - 数字开关量输入->输入电平的0/1态携带信息->IO 轮询或外部中断
 - 数字脉宽输入->输入脉宽携带信息->使用定时器的输入捕获功能
 - SPI 通信总线->使用硬件SPI外设
 - I2C通信总线->使用模拟 I2C/I2C 硬件外设
 - 串口通信->使用硬件USART外设

举个栗子

- 以“小老鼠”板载的 MPU9250 传感器的使用为例：
- 理想的开发流程：
 - 拿到传感器手册，确认通信方式和时序
 - 拿到电路板原理图，确认引脚连接
 - 如神附体，编写程序，几百个教授(测例)一致(次)通过！
- 走向人生巅峰...



然而，真实情况...

woc, 引脚怎么是错的!

尼玛, 怎么是么都不输出??

MDZZ, 编译都不过!!!

怎么冒烟了???



正确? 的打开方式

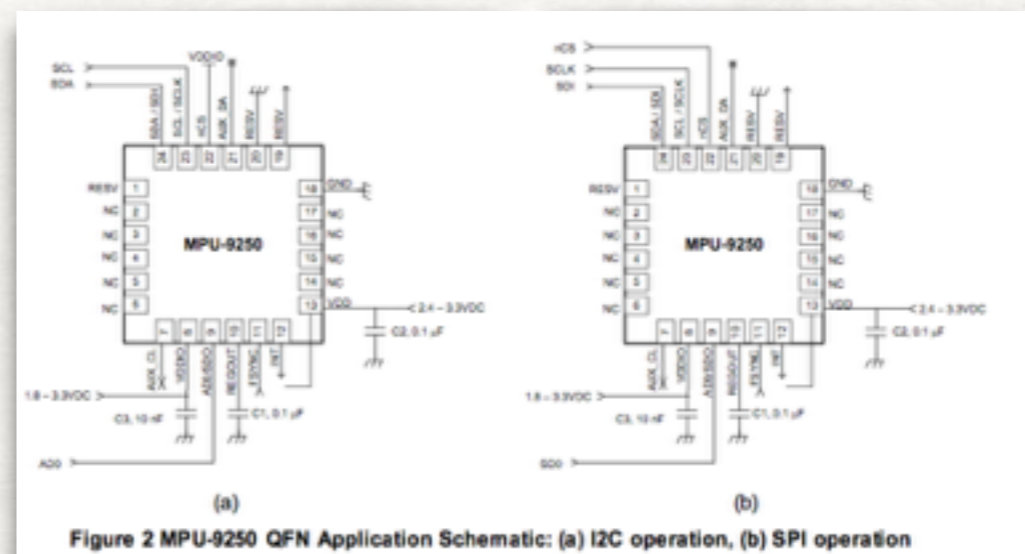
- (大雾 直接百度一段代码并照抄
 - 出错可能性很大
 - 囫圇吞枣，没有收获
 - 养成把原因归咎给他人的坏习惯

正确的打开方式—1. 阅读手册

- 阅读手册的正确方法：
 - 重点关注
 - 电气特性-> 防止“冒烟”
 - 通信方式-> SPI?I2C?..
 - 时序特性
 - 引脚定义

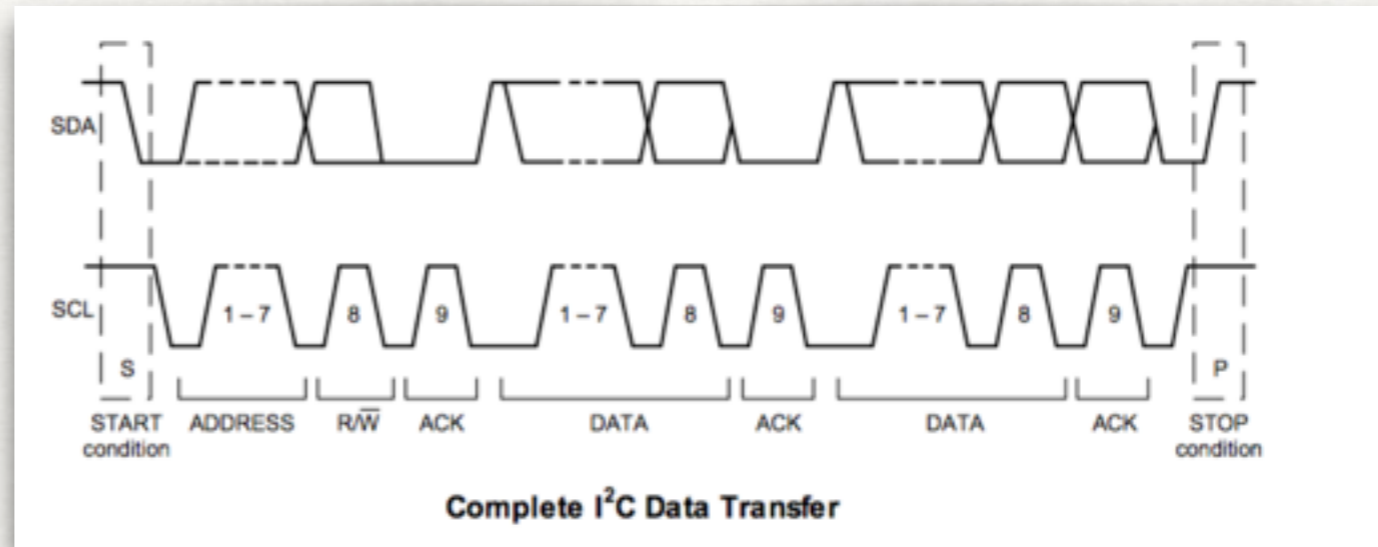
| 3.4.1 D.C. Electrical Characteristics | | | | | | |
|--|---|--|------|-----|-------|-------|
| Typical Operating Circuit of section 4.2, VDD = 2.5V, VDDIO = 2.5V, T _A = 25°C, unless otherwise noted. | | | | | | |
| PARAMETER | CONDITIONS | MIN | TYP | MAX | Units | Notes |
| SUPPLY VOLTAGES | | | | | | |
| VDD | | 2.4 | 2.5 | 3.6 | V | |
| VDDIO | | 1.71 | 1.8 | VDD | V | |
| SUPPLY CURRENTS | | | | | | |
| Normal Mode | 9-axis (no DMP), 1 kHz gyro ODR, 4 kHz accel ODR, 8 Hz mag. repetition rate | | 3.7 | | mA | |
| | 6-axis (accel + gyro, no DMP), 1 kHz gyro ODR, 4 kHz accel ODR | | 3.4 | | mA | |
| | 3-axis Gyroscope only (no DMP), 1 kHz ODR | | 3.2 | | mA | |
| | 6-axis (accel + magnetometer, no DMP), 4 kHz accel ODR, mag. repetition rate = 8 Hz | | 730 | | µA | |
| | 3-Axis Accelerometer, 4 kHz ODR (no DMP) | | 450 | | µA | |
| | 3-axis Magnetometer only (no DMP), 8 Hz repetition rate | | 280 | | µA | |
| Accelerometer Low Power Mode (DMP, Gyroscope, Magnetometer disabled) | 0.98 Hz update rate | | 8.4 | | µA | 1 |
| | 31.25 Hz update rate | | 19.8 | | µA | 1 |
| Full Chip Idle Mode Supply Current | | | 8 | | µA | |
| TEMPERATURE RANGE | | | | | | |
| Specified Temperature Range | | Performance parameters are not applicable beyond Specified Temperature Range | | -40 | +85 | °C |

Table 3 D.C. Electrical Characteristics



正确的打开方式—2.了解总线

- I2C 总线
 - 分时半双工总线
 - 主从结构
 - 两条线组成：SCL，SDA



- 一次完整的 I2C 传输过程
 1. 主设备拉低 SCL，保持一段时间后拉高，表示总线使能
 2. 主设备在 SDA 上发送待读写的地址
 3. 主设备等待从设备响应 ACK
 4. 主设备发出待写入从设备的数据/从设备发送待主设备读取的数据
 5. 主设备释放 SCL（拉高）传输结束

正确的打开方式—3. 查看时序

- 通信时序一般定义了：
 - 先后顺序
 - 谁先做什么
 - 然后做什么

To write the internal MPU-9250 registers, the master transmits the start condition (S), followed by the I²C address and the write bit (0). At the 9th clock cycle (when the clock is high), the MPU-9250 acknowledges the transfer. Then the master puts the register address (RA) on the bus. After the MPU-9250 acknowledges the reception of the register address, the master puts the register data onto the bus. This is followed by the ACK signal, and data transfer may be concluded by the stop condition (P). To write multiple bytes after the last ACK signal, the master can continue outputting data rather than transmitting a stop signal. In this case, the MPU-9250 automatically increments the register address and loads the data to the appropriate register. The following figures show single and two-byte write sequences.

Single-Byte Write Sequence

| | | | | | | | | |
|--------|---|------|-----|----|-----|------|-----|---|
| Master | S | AD+W | | RA | | DATA | | P |
| Slave | | | ACK | | ACK | | ACK | |

Burst Write Sequence

| | | | | | | | | | | |
|--------|---|------|-----|----|-----|------|-----|------|-----|---|
| Master | S | AD+W | | RA | | DATA | | DATA | | P |
| Slave | | | ACK | | ACK | | ACK | | ACK | |

To read the internal MPU-9250 registers, the master sends a start condition, followed by the I²C address and a write bit, and then the register address that is going to be read. Upon receiving the ACK signal from the MPU-9250, the master transmits a start signal followed by the slave address and read bit. As a result, the MPU-9250 sends an ACK signal and the data. The communication ends with a not acknowledge (NACK) signal and a stop bit from master. The NACK condition is defined such that the SDA line remains high at the 9th clock cycle. The following figures show single and two-byte read sequences.

Single-Byte Read Sequence

| | | | | | | | | | | | |
|--------|---|------|-----|----|-----|---|------|-----|------|------|---|
| Master | S | AD+W | | RA | | S | AD+R | | | NACK | P |
| Slave | | | ACK | | ACK | | | ACK | DATA | | |

Burst Read Sequence

| | | | | | | | | | | | | | |
|--------|---|------|-----|----|-----|---|------|-----|------|-----|------|------|---|
| Master | S | AD+W | | RA | | S | AD+R | | | ACK | | NACK | P |
| Slave | | | ACK | | ACK | | | ACK | DATA | | DATA | | |

正确的打开方式—4.快速开发

- 两个思路
 - 使用 CubeMX + HAL 库，自行实现手册的时序
 - 无需关注底层通讯总线行为
- 网上查找已有的代码，进行相应的修改，调试
 - 由于 STM32 I2C 外设的问题，大部分是采用模拟方式实现的

- `HAL_I2C_Master_Transmit()`
- `HAL_I2C_Master_Receive()`
- `HAL_I2C_Slave_Transmit()`
- `HAL_I2C_Slave_Receive()`
- `HAL_I2C_Master_Transmit_IT()`
- `HAL_I2C_Master_Receive_IT()`
- `HAL_I2C_Slave_Transmit_IT()`
- `HAL_I2C_Slave_Receive_IT()`

```
/*
 * Function Name : I2C_SendByte
 * Description   : Master Send a Byte to Slave
 * Input        : Will Send Date
 * Output       : None
 * Return       : None
 */
void I2C_SendByte(u8 SendByte) //数据从高位到低位//
{
    u8 i=8;
    while(i--)
    {
        SCL_L;
        I2C_delay();
        if(SendByte&0x80)
            SDA_H;
        else
            SDA_L;
        SendByte<<=1;
        I2C_delay();
        SCL_H;
        I2C_delay();
    }
    SCL_L;
}
```

动手时间

几个忠告

- 硬件没那么容易坏
- 大部分问题都出在自己身上
- 越是诡异的问题，原因越愚蠢

谢谢观看
欢迎提问