

专题2 一个综合的设计任务



主讲教师:朱海龙

广东工业大学

tank12345678@126.com

本PPT的主要参考文献

- [1] STM32F1开发指南-库函数版本_V3.1
- [2] STM32中文参考手册_V10
- [3]
- [4]



跑马灯程序详解

- 0 综合设计任务：一个含待机功能的温湿度监控系统
- 1 子任务：串行接口控制（第9章）
- 2 子任务：中断控制（第10、13章）
- 3 子任务：液晶显示器控制（第18章）
- 4 子任务：RTC控制（第20章）
- 5 子任务：低功耗模式切换（第21章）
- 6 子任务：内部温度传感器控制（第23章）
- 7 子任务：DHT11温湿度传感器控制（第36章）
- 8 子任务：GPRS通信模块的控制（附加内容）



0 综合设计任务：一个含待机功能的 温湿度监控系统



设计目标

- 1 设计一个温度检测系统，同时使用STM32芯片的内部温度传感器和外部的DHT11温湿度传感器对环境参数进行监控；
- 2 所读到的数据可以使用串口助手进行监测，在PC上进行显示；
- 3 所读到的数据可以使用液晶显示屏进行显示；
- 4 系统完成传感器数据的读取和显示后，即进入低功耗工作模式（可在睡眠、停机、待机模式之中任选一种），要求每分钟唤醒（自动唤醒和自动切换）一次，并对传感器数据进行下一次测量和显示。
- 5 额外任务：传感器数据的远程发送（供参考，选做）。



1 子任务：串行接口控制（第9章）



2 子任务：中断控制（第10、13章）



3 子任务：液晶显示器控制（第18章）



4 子任务：RTC控制（第20章）



5 子任务：低功耗模式切换（第21章）



6 子任务：内部温度传感器控制（第23章）



7 子任务：DHT11温湿度传感器控制（第36章）



DHT11的使用方法

- 电阻式感湿元件+NTC测温元件，内部集成MCU。
- 单线制串行接口。
- 数据线需要接5K~10K的上拉电阻。



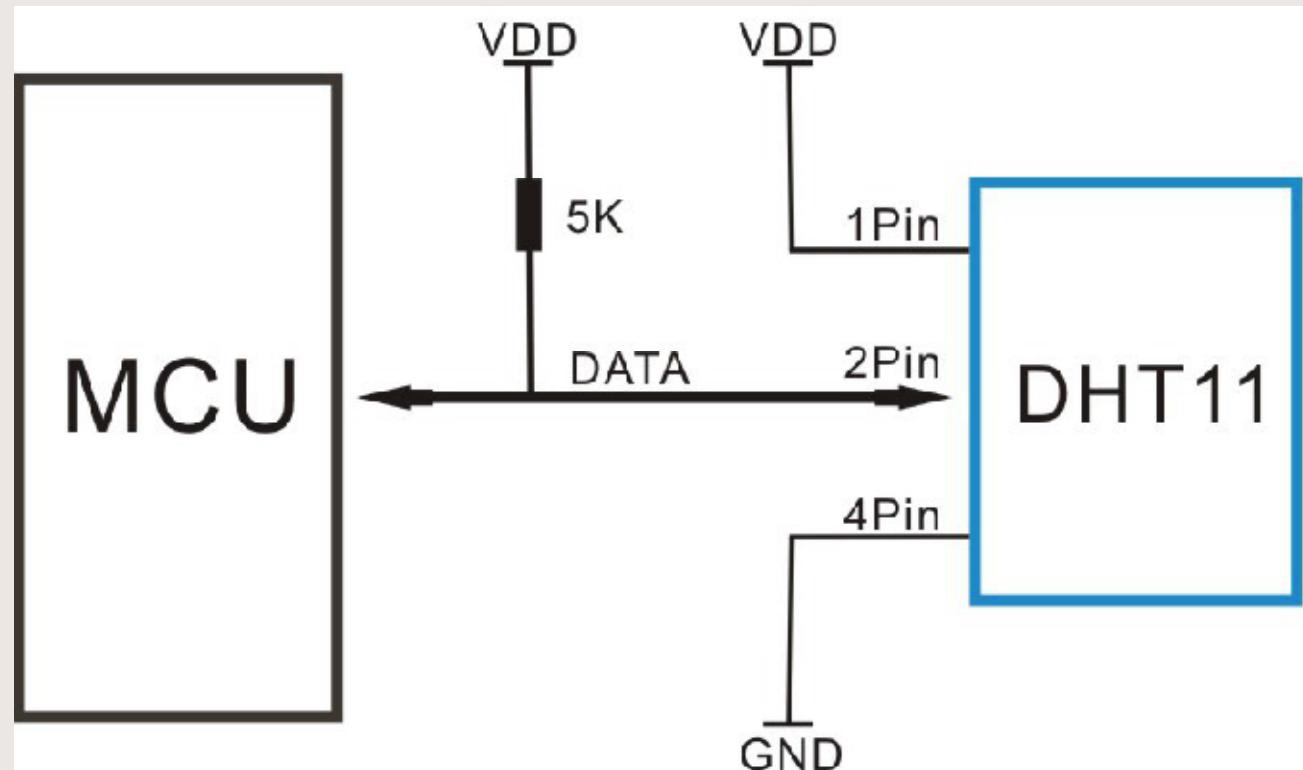
DHT11基本特性

- 一款含有含有已校准数字信号输出的温度湿度复合传感器。
- 测温范围：0~50度，测湿范围：20~90%，精度：
：测温±2度，测湿±5%。



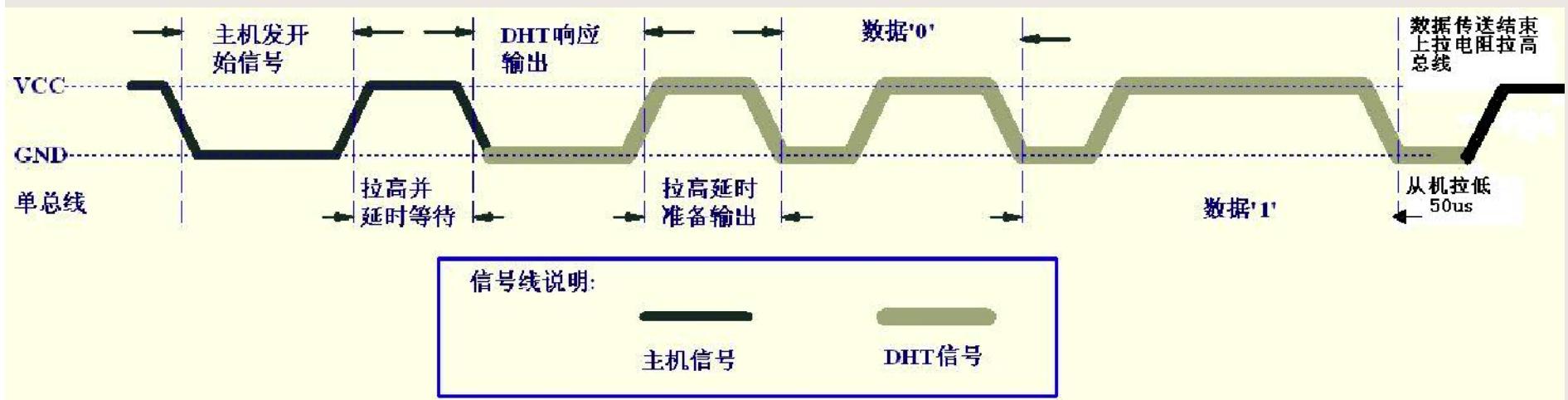
典型应用电路接法

- 注意：5K（或其它数值）的上拉电阻一定要接。电源引脚到地之间可考虑接100nF的去耦电容。



DHT11的信号传输时序

- 总线空闲状态为高电平：即总线必须接上拉电阻，在主机和DHT11均无信号输出时，总线上电压因为上拉电阻的原因，维持在高电平。



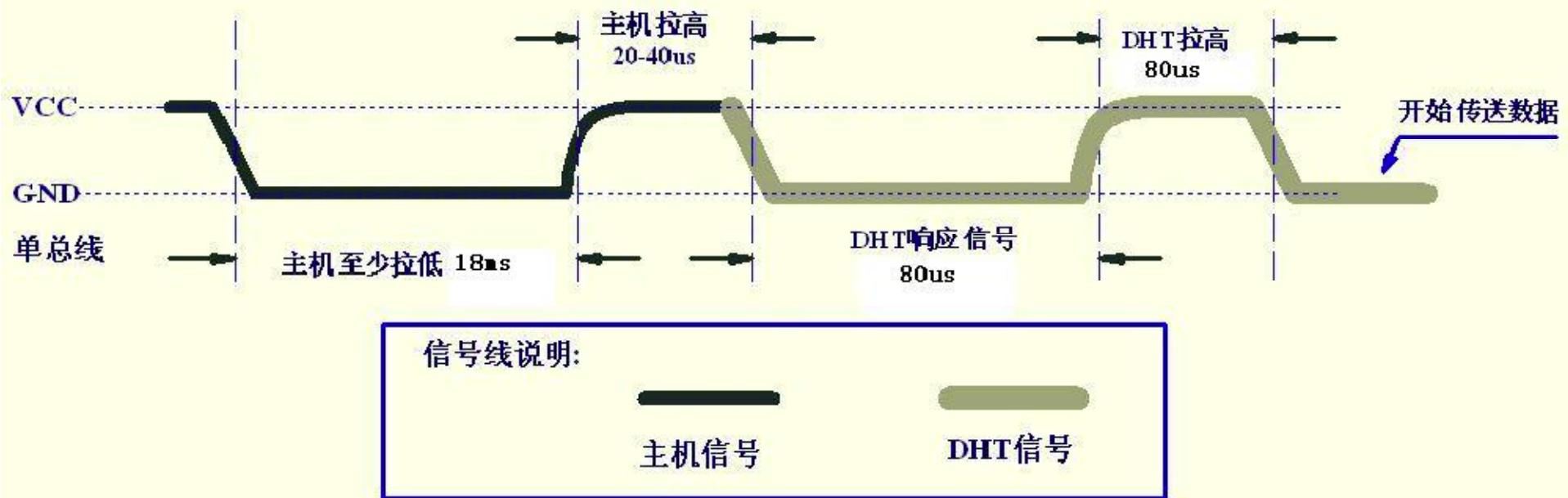
DHT11数据传送过程详解

- 1 DHT11初始工作在低功耗模式。
- 2 用户MCU发送一次开始信号（等同于激活DHT11），DHT11发送响应信号，送出40bit数据。并触发一次信号采集。
- 3 DHT11接收到开始信号后，触发一次温湿度采集。如果没有接收到主机发送的开始信号，DHT11不会主动进行温湿度采集。



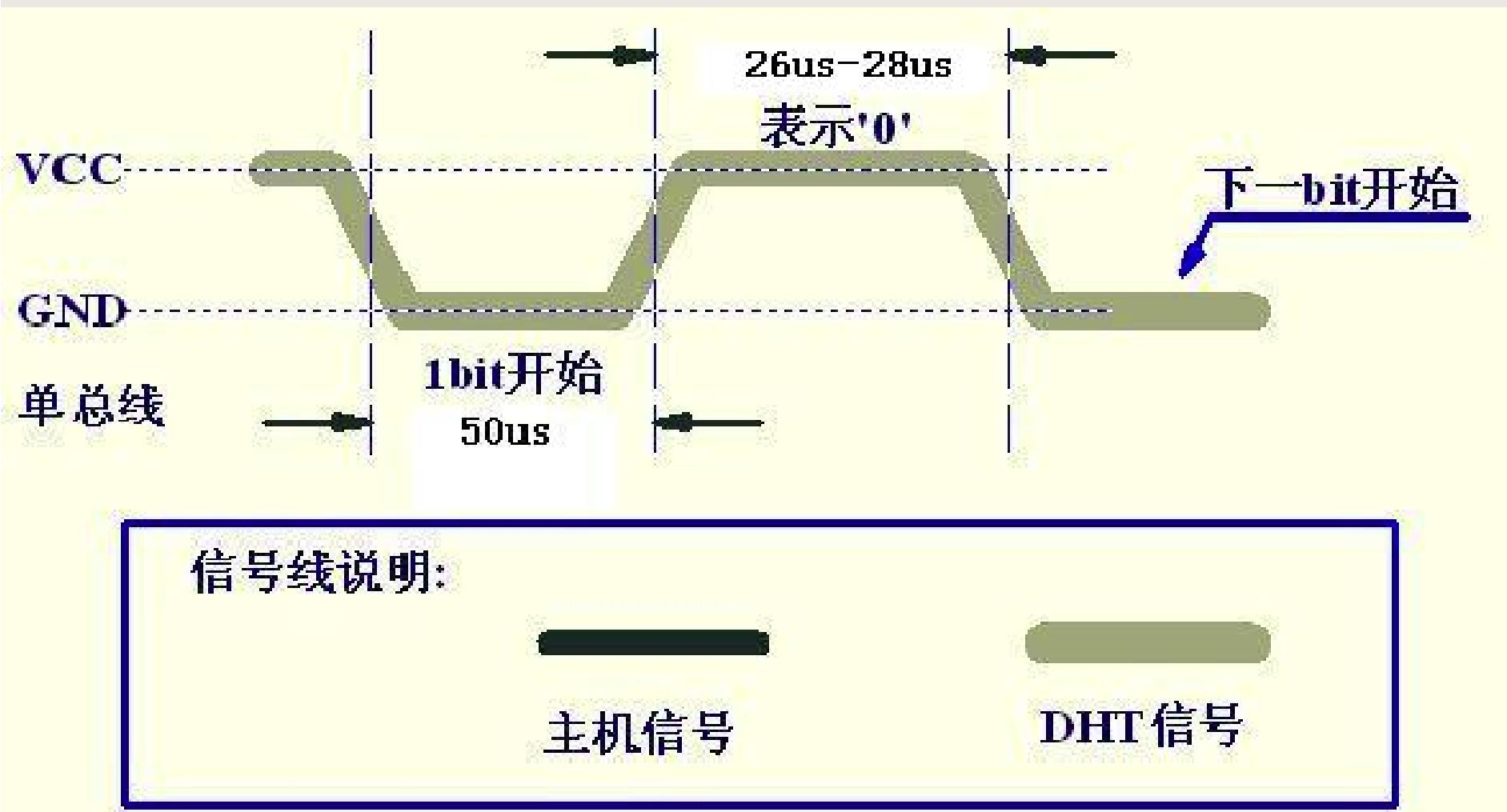
- 主机把总线拉低必须大于18毫秒,保证DHT11能检测到起始信号。DHT11接收到主机的开始信号后,等待主机开始信号结束,然后发送80us低电平响应信号.主机发送开始信号结束后,延时等待20-40us后,读取DHT11的响应信号,主机发送开始信号后,可以切换到输入模式,总线由上拉电阻拉高。
 -

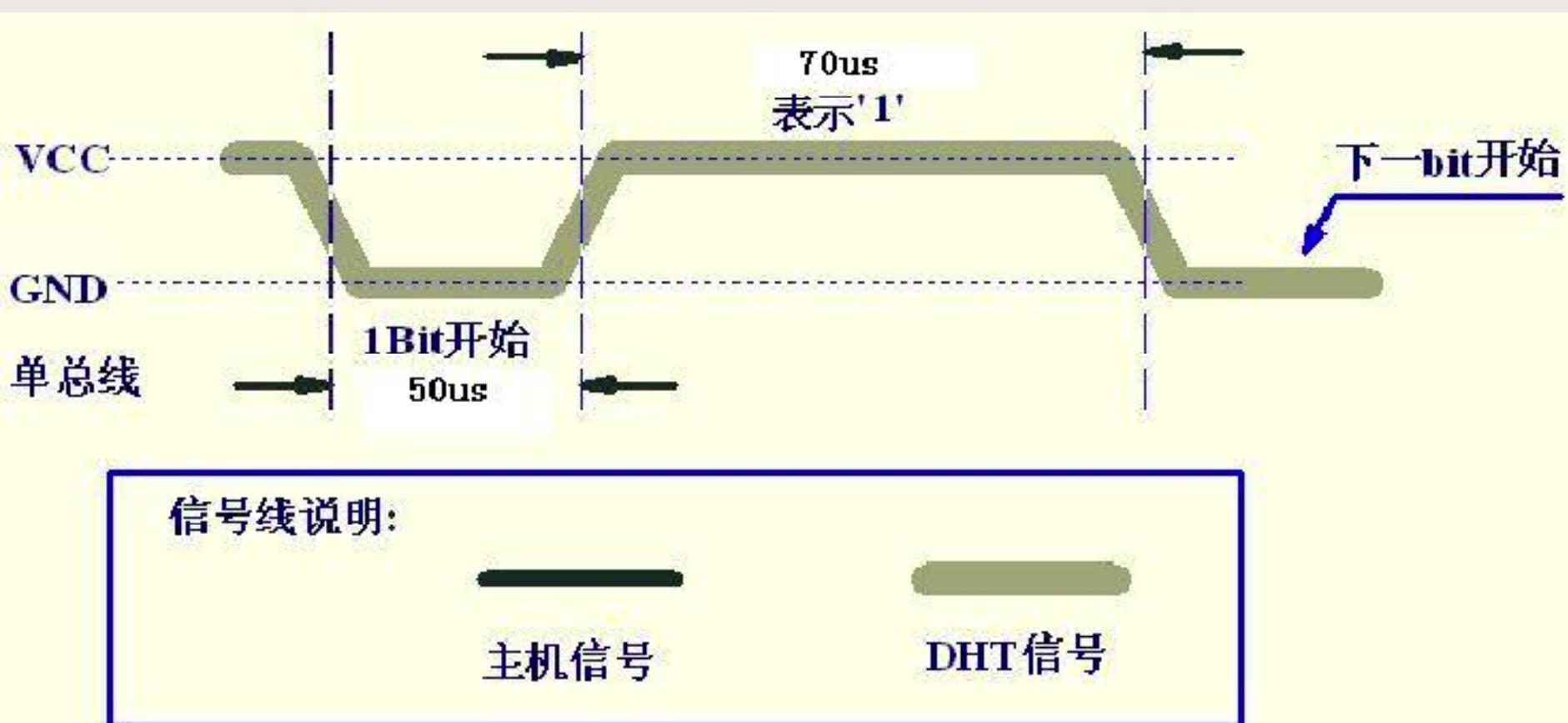


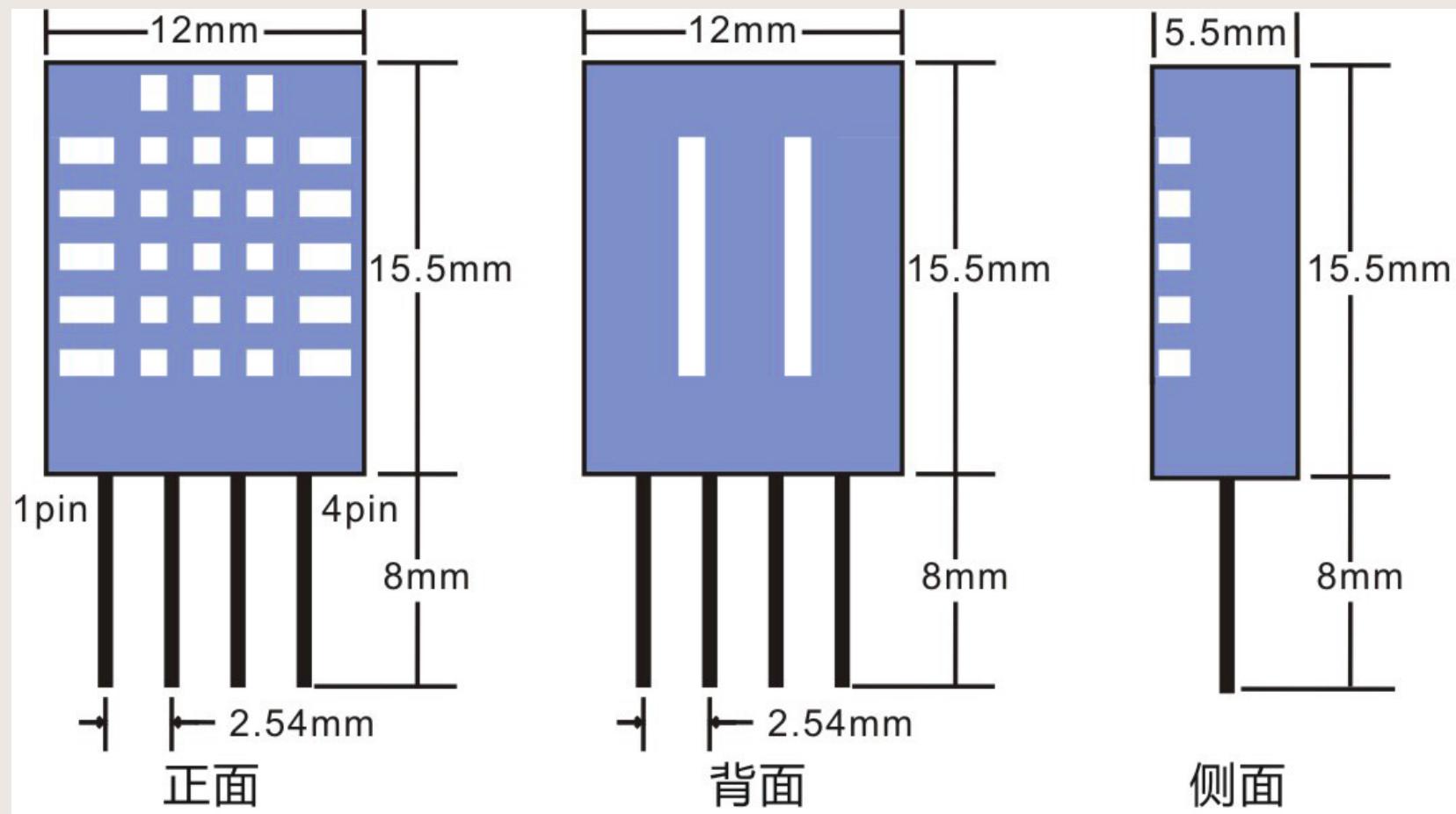


- DHT11发送响应信号后,再把总线拉高80us,准备发送数据,每一bit数据都以50us低电平时隙开始,高电平的长短决定了数据位是0还是1.格式见下面图示.
- 如果读取响应信号为高电平,则DHT11没有响应,请检查线路是否连接正常.当最后一bit数据传送完毕后, DHT11拉低总线50us,随后总线由上拉电阻拉高进入空闲状态。









Pin	名称	注释
1	VDD	供电 3–5.5VDC
2	DATA	串行数据, 单总线
3	NC	空脚, 请悬空
4	GND	接地, 电源负极

DHT11相关控制程序详解

- 编程的几个主要部分：
 - 1 DHT11的数据提取
 - 2 低功耗与正常模式的切换
 - 3 终端节点（EndDevice）与协调器（Coordinator）之间的数据通信
 - 4 通过GPRS将数据发回后台



main.c

- #include <ioCC2530.h>
- #include <string.h>
- #include "UART.H"
- #include "DHT11.H"



main.c

-
- void main(void)
 - {
 - Delay_ms(1000); //让设备稳定
 - InitUart(); //串口初始化



Delay_ms()

- void Delay_ms(uint Time)//n ms延时
- {
- unsigned char i;
- while(Time--)
- {
- for(i=0;i<100;i++)
- Delay_10us();
- }
- }



Delay_us()

-
- void Delay_10us() //10 us延时
 - {
 - Delay_us();
 - }



Delay_us

- void Delay_us() //1 us延时
- {
- asm("nop");
- }
- //如何保证这里的延时是准确的1us呢？当芯片的系统时钟改变时，如何调整这个程序？



InitUart()

- void InitUart()
- {
- CLKCONCMD &= ~0x40; // 设置系统时钟源为 32MHZ晶振
- while(CLKCONSTA & 0x40); // 等待晶振稳定
- CLKCONCMD &= ~0x47; // 设置系统主时钟频率为 32MHZ
- PERCFG = 0x00; //位置1 P0口
- P0SEL = 0x3c; //P0_2,P0_3,P0_4,P0_5用作串口,第二功能
- P2DIR &= ~0XC0; //P0 优先作为UART0 , 优先级
-
-



InitUart()

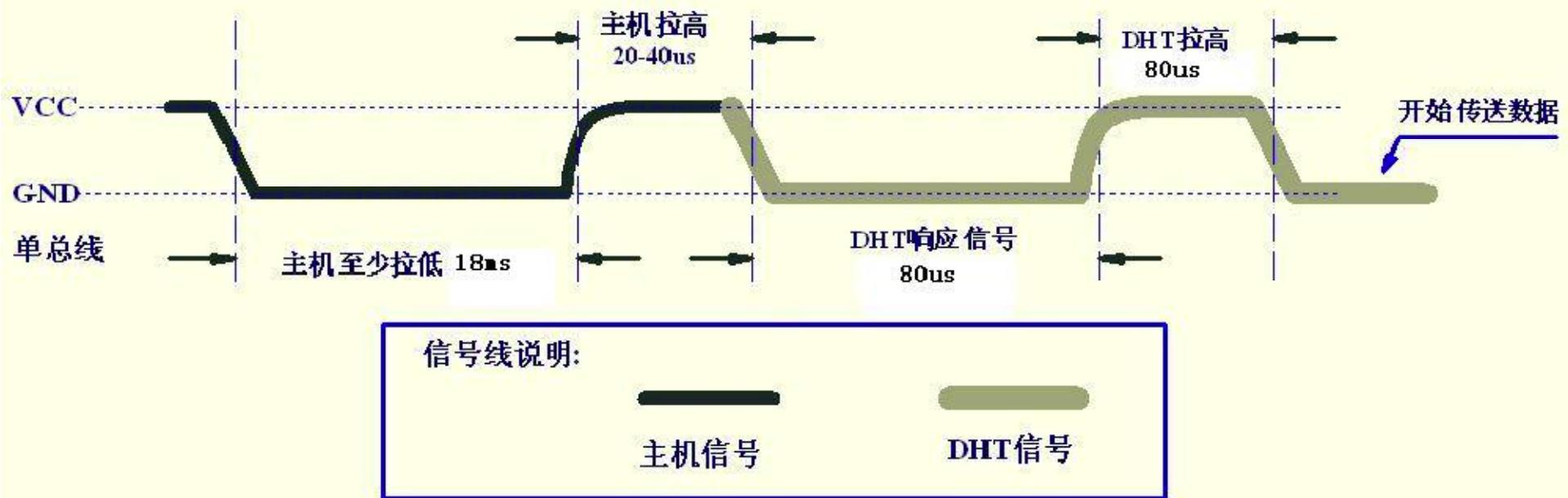
- U0CSR |= 0x80; //UART 方式
- U0GCR |= 11; //U0GCR与U0BAUD配合
- U0BAUD |= 216; // 波特率设为115200
- UTX0IF = 0; //UART0 TX 中断标志初始置位0
- }



main.c

- while(1)
- {
- DHT11(); //获取温湿度





DHT11()

- void DHT11(void) //温湿传感启动
- {
- wenshi=0;
- Delay_ms(19); //>18MS 主机把总线拉低至少18ms !
- wenshi=1;
- P0DIR &= ~0x40; //重新配置IO口方向
- // &=1011 1111, 即第6位置0, 设P0-6位的方向为输入, 在这一步设置之后,
原有的wenshi=1应该就失效了, 因为IO已经变成了输入状态, 开始输入捕获
。
- Delay_10us();
- Delay_10us();
- Delay_10us();
- Delay_10us(); //将电平拉高**40us**, 注意其拉高是通过上拉电阻实现的, 因此
上拉电阻必不可少。
-



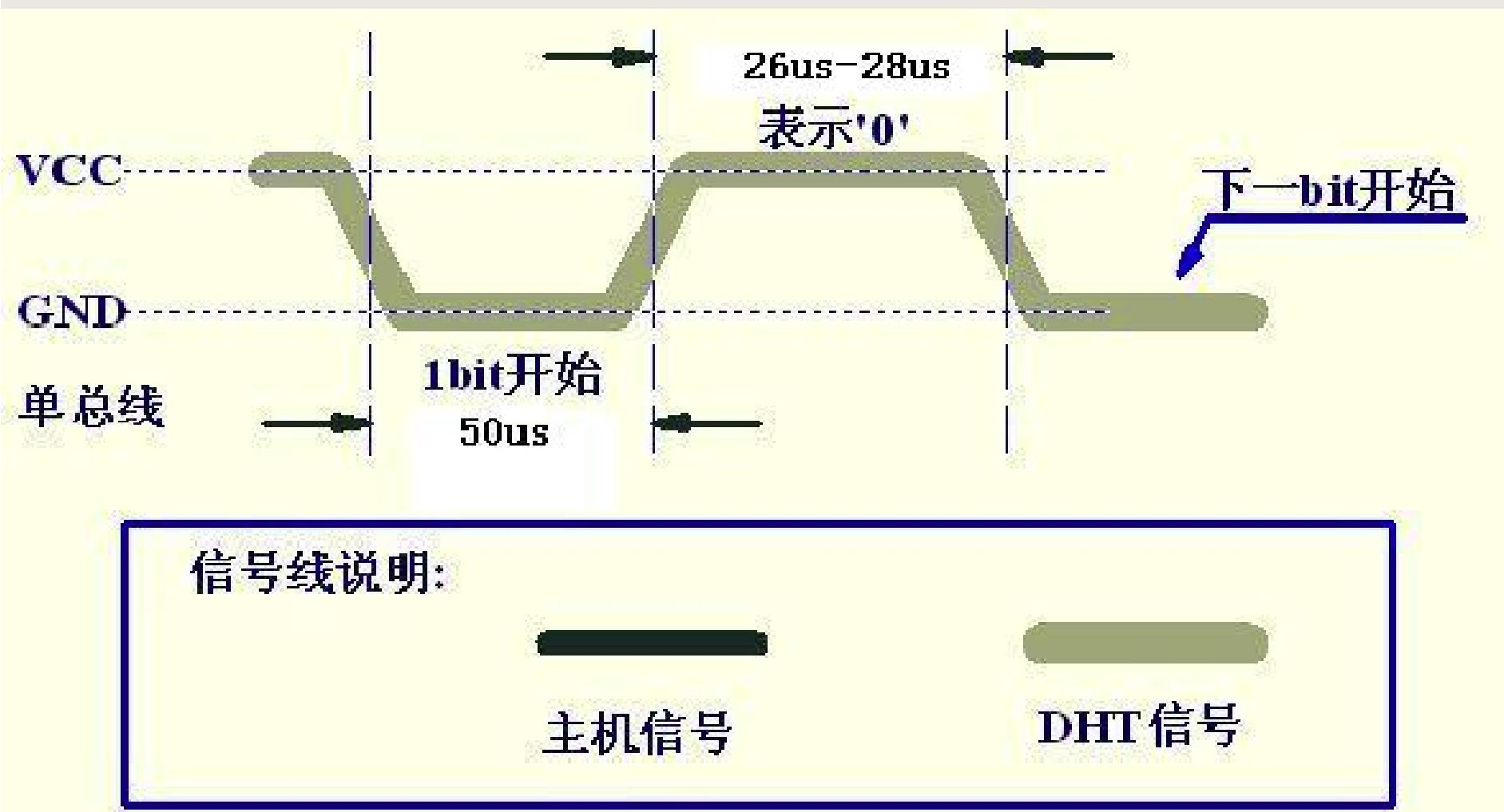
DHT11()

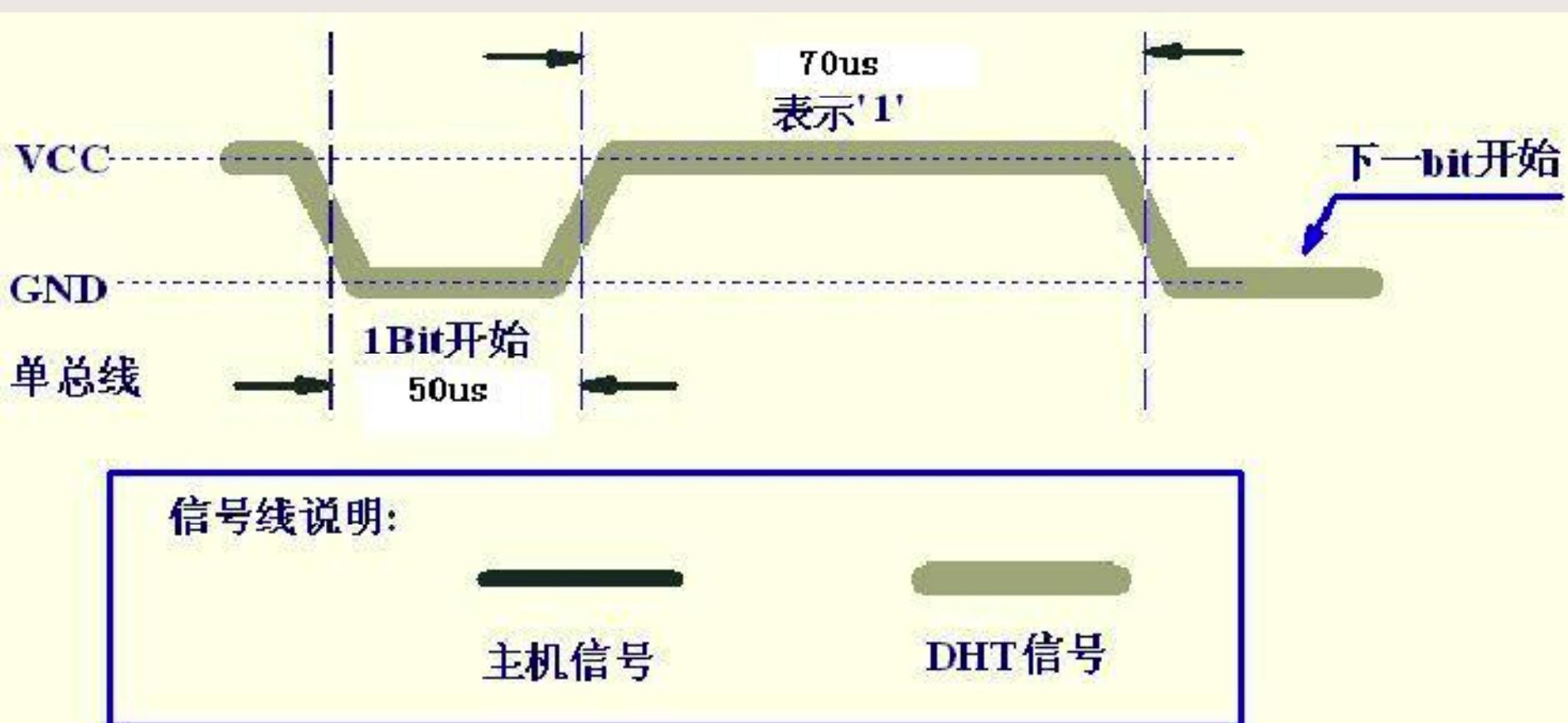
- if(!wenshi) //40us时间之内， wenshi管脚电压将因为DHT11的输出而受控， 注意与波形图比较， 判定其输入的数据。判定第一个信号拉低的波形（80us）
- {
- ucharFLAG=2;
- while((!wenshi)&&ucharFLAG++);
- //当wenshi=0时， 一直在执行上句， 但为什么要写成这样呢， ucharFLAG并没有什么用！！！
- ucharFLAG=2;
- while((wenshi)&&ucharFLAG++);
- //当wenshi=1时， 一直在执行上句， 直到wenshi由1变0时为止。
- COM(); //接收来自于DHT11的第一个字节， 正常的数据传输
-



- 注意此处的控制位：
 - 1 whshi
 - #define wenshi P0_6
 - 2 ucharFLAG应用于判断是否收满了一个字节的8个位，程序在这里是有bug的。
 - while(dht11 == 0 && count++ < NUMBER);
 - if(count >= NUMBER)
 - {
 - status = ERROR; //设定错误标志
 - return 0; //函数执行过程发生错误就退出函数
 - } //手册上的程序是这样的，此处NUMBER预设为20，避免管脚在某个电平状态下被锁定。！！！必须要这样进行修改！







COM()

- void COM(void) // 温湿写入
- {
- uchar i;
- for(i=0;i<8;i++)
- {
- ucharFLAG=2;
- while((!wenshi)&&ucharFLAG++);
- //当wenshi=0时，一直在执行上句，即在50us低电平期间，程序处在循环状态
- Delay_10us();
- Delay_10us();
- Delay_10us(); //此处的30us目的是判断波形是0还是1
- uchartemp=0; //默认波形为0（当wenshi经30us后变为0时）



COM()

- if(wenshi) uchar temp=1; //反之，如果wenshi=1，则代表脉冲为1的时间大于30us，波形为1。
- uchar FLAG=2;
- while((wenshi)&&ucharFLAG++);
- //当wenshi=1时，一直在执行上句，直到wenshi由1变0时为止。
- if(ucharFLAG==1)break; //flag没有机会等于1
- uchar comdata<<=1; //把上一循环读入的位左移
- uchar comdata|=uchar temp; //读入当前位
- }
- }



DHT11 ()

- ucharRH_data_H_temp=ucharcomdata;
COM();
- ucharRH_data_L_temp=ucharcomdata;
COM();
- ucharT_data_H_temp=ucharcomdata;
COM();
- ucharT_data_L_temp=ucharcomdata;
COM();
- ucharcheckdata_temp=ucharcomdata;
- wenshi=1; //? 这个指令是无效指令，因为此时wenshi还在输入状态！！！这个指令能将输入状态自动切换成输出状态吗？
-



DHT11()

- uchar temp=(ucharT_data_H_temp+ucharT_data_L_temp+ucharRH_data_H_temp+ucharRH_data_L_temp);

//由接收到的前4个字节计算checksum的数值，准备与后面接收到的第5个字节进行比对，以确认数据的有效性。



DHT11()

- if(uchartemp==ucharcheckdata_temp)
- //判断前4个字节的chechksum与第5个字节是否相等
- {
- ucharRH_data_H=ucharRH_data_H_temp;
- ucharRH_data_L=ucharRH_data_L_temp;
- ucharT_data_H=ucharT_data_H_temp;
- ucharT_data_L=ucharT_data_L_temp;
- ucharcheckdata=ucharcheckdata_temp;
- }
-



温湿度测量值的数据转换

- wendu_shi=ucharT_data_H/10;
- wendu_ge=ucharT_data_H%10;
-
- shidu_shi=ucharRH_data_H/10;
- shidu_ge=ucharRH_data_H%10;
- }
- //为什么要这样处理? /*可在其他的文件引用温湿度值
,实际是温度的整数的10倍
- 如dht11读回的温度是26,则temp_value = 260, 湿度同理
*/



- else //checksum检查失败， 没有成功读取， 返回0
- {
- wendu_shi=0;
- wendu_ge=0;
-
- shidu_shi=0;
- shidu_ge=0;
- }
- }



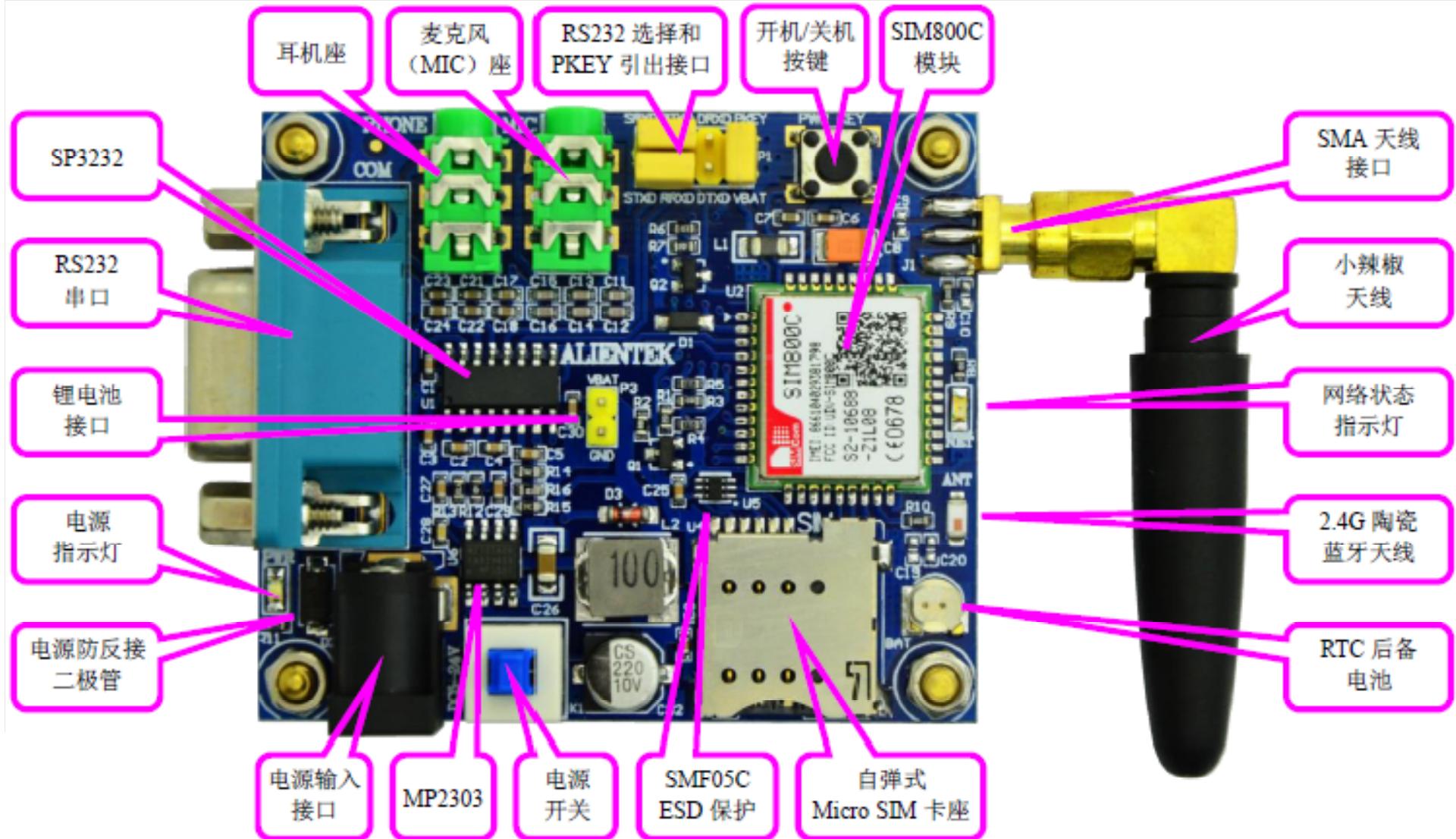
8 子任务：GPRS通信模块的控制（附加内容）



-
- 选用ATK-SIM800C-V15(V15为版本号)
GSM/GPRS模块实现传感器数据到后台的传递。
 - ATK-SIM800C模块板载SIMCOM公司的工业级
四频GSM/GPRS（Global System for Mobile
Communication/General Packet Radio Service）模
块



GPRS模块的构成



GPRS与GSM的关系

- GPRS可说是GSM的延续。GPRS和以往连续在频道传输的方式不同，是以封包（Packet）式来传输，因此使用者所负担的费用是以其传输资料单位计算，并非使用其整个频道，理论上较为便宜。GPRS的传输速率可提升至56甚至114Kbps。



- 系统四频率工作：850/900/1800/1900MHz，可以低功耗实现语音、SMS(Short Messaging Service短信)、MMS (Multimedia Messaging Service彩信)，蓝牙数据信息的传输等。
- 模块所支持的数据接口：RS232串口、**LVTTL** (**Low Voltage Transistor-Transistor Logic**) 串口？如果支持LVTTL？



数据的上行与下行

- 上行：用户端向后台传送数据。
- 下行：后台向用户端传送数据。



通过MCU控制ATK-SIM800C模块

- 通过MCU控制ATK-SIM800C模块的基本思路：
 - 由于ATK-SIM800C模块开放的数据端口是串口，因此主要考虑通过串口对模块进行控制。
 - MCU通过串口发送特定的指令给ATK-SIM800C模块，模块将对发送的指令进行解释，并根据解释的结果执行相应操作。



AT指令

- AT指令：AT 即Attention，AT指令集是从终端设备(Terminal Equipment, TE)或数据终端设备(Data Terminal Equipment, DTE)向终端适配器(Terminal Adapter, TA)或数据电路终端设备(Data Circuit Terminal Equipment, DCE)发送的。通过TA, TE发送AT指令来控制移动台(Mobile Station, MS)的功能，与GSM 网络业务进行交互。用户可以通过AT指令进行呼叫、短信、数据传送业务等功能。



GSM/GPRS模块的基础指令

- SIM800C模块提供的AT命令包含符合3GPP TS 27.005、3GPP TS 27.007和ITU-T Recommendation V.25ter的指令，以及SIMCOM自己开发的指令。
- 常用的GPRS相关AT指令包括11条：
 - AT+CGCLASS/AT+CGDCONT/
AT+CGATT/AT+CIPCSGP/AT+CIPHEAD
/AT+CLPORT/AT+CIPSTART/
 - AT+CIPSEN/AT+CIPSTATUS/AT+CIPCLOSE/ACIPSHUT等11
条AT指令。



AT指令的基本格式

- AT指令必须以"AT"或"at"开头，以回车（<CR>）结尾。模块的响应通常紧随其后，格式

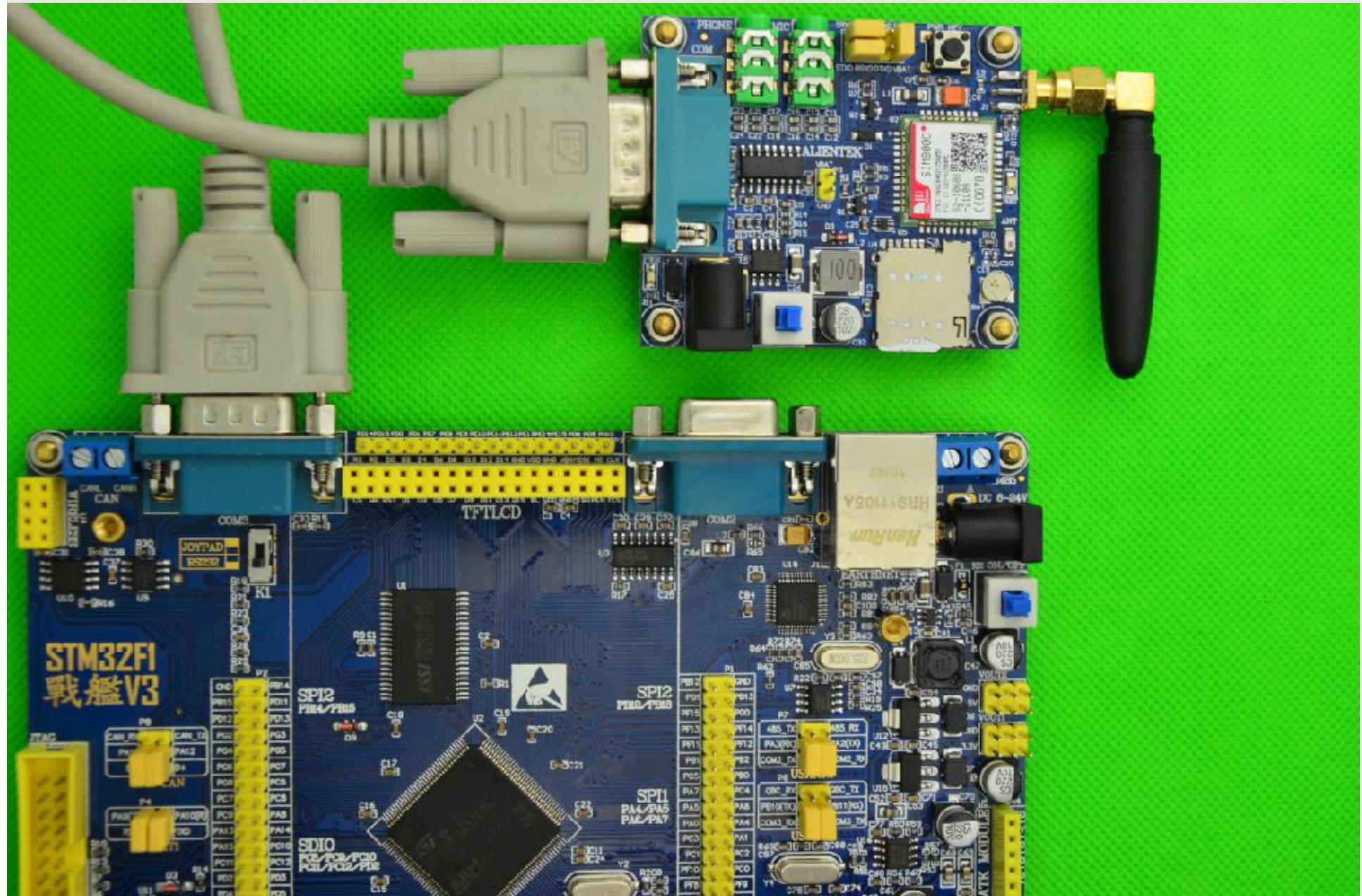


设备的硬件连接

- 1 由于GPRS采用串口进行数据的通讯，因此采用特定的UART TX&RX信号实现GPRS模块与其它设备的连接即可。
- 2 可以采用两种方法进行有效连接：
 - 2.1 不经过RS232信号的转换。RS232是±15V电平的信号，需通过SP3232或类似功能芯片进行电平转换。可将未经转换的信号直接连接，以实现数据发送。
 - 2.2 通过通用的RS232串口，经过电平转换后再连接。



开发板与模块的链接（RS232模式）



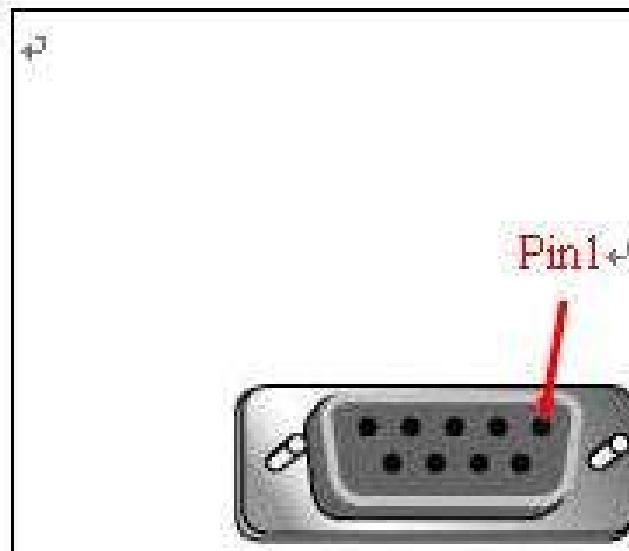
RS232模式的连接

- 在此连接模式中，将开发板的com3通过串口线（注意串口线有不同分类，此处使用的是直连线，即公口与母口的线一一对应）连接至GPRS模块的com口。



RS232管脚定义

- pin 2 为RXD（接收信号），pin3为TXD（发射信号），7为RTS（要求传送），8为CTS（清除以传送）



Pin	Contact	简写	意义	25 针 RS232
1	M.C.	CD	载波侦测	8: CD
2	RXD	RXD	接收字符	3: RXD
3	TXD	TXD	传送字符	2: TXD
4	M.C.	DTR	数据端设备就绪	20: DTR
5	GND	GND	地线	7: GND
6	M.C.	DSR	数据设备就绪	6: DSR
7	RTS	RTS	要求传送	4: RTS
8	CTS	CTS	清除以传送	5: CTS
9	M.C.	RI	响铃侦测	22: RI

公口与母口的区别

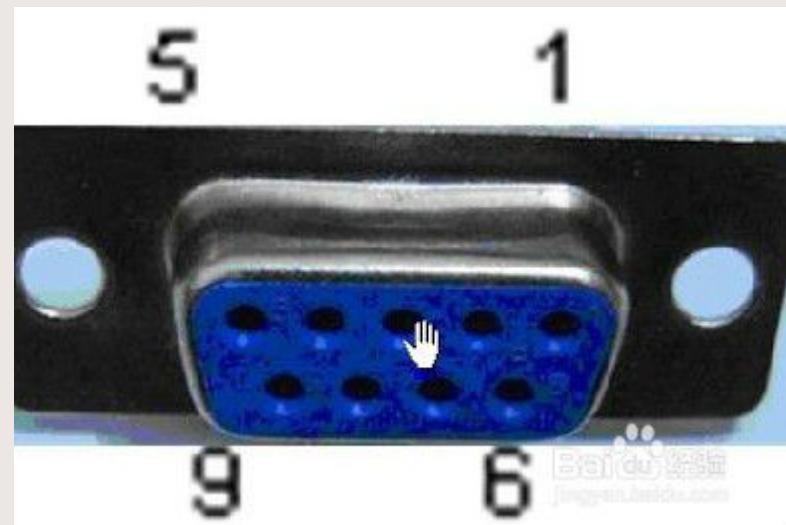


DB-9针串口引脚图



- | | |
|---------------|-------------|
| 1 DCD 载波检测 | ——方向：终端到计算机 |
| 2 RXD 接收数据 | ——方向：计算机到终端 |
| 3 TXD 发送数据 | |
| 4 DTR 数据终端准备好 | |
| 5 GND 信号地线 | |
| 6 DSR 数据准备好 | |
| 7 RTS 请求发送 | |
| 8 CTS 清除发送 | |
| 9 RI 振铃指示 | |

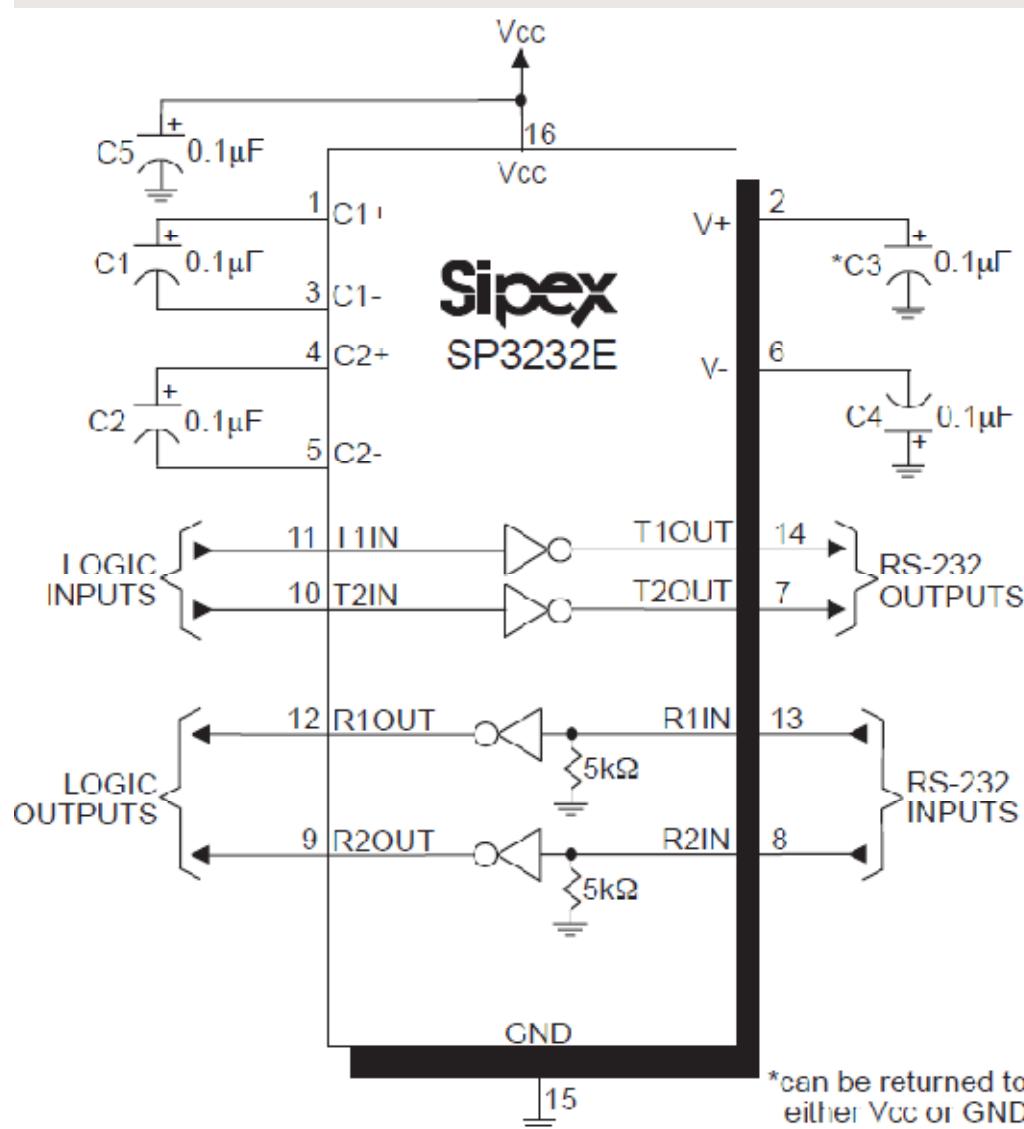




脚位	简写	意义
Pin 1	CD	载波侦测 (Carrier Detect)
Pin 2	RXD	接收字符 (Receive)
Pin 3	TXD	传送字符 (Transmit)
Pin 4	DTR	数据端被妥 (Data Terminal Ready)
Pin 5	GND	地线 (Ground)
Pin 6	DSR	数据被妥 (Data Set Ready)
Pin 7	RTS	要求传送 (Request To Send)
Pin 8	CTS	清除以传送 (Clear to Send)
Pin 9	RI	响铃侦测 (Ring Indicator)



关键芯片：SP3232

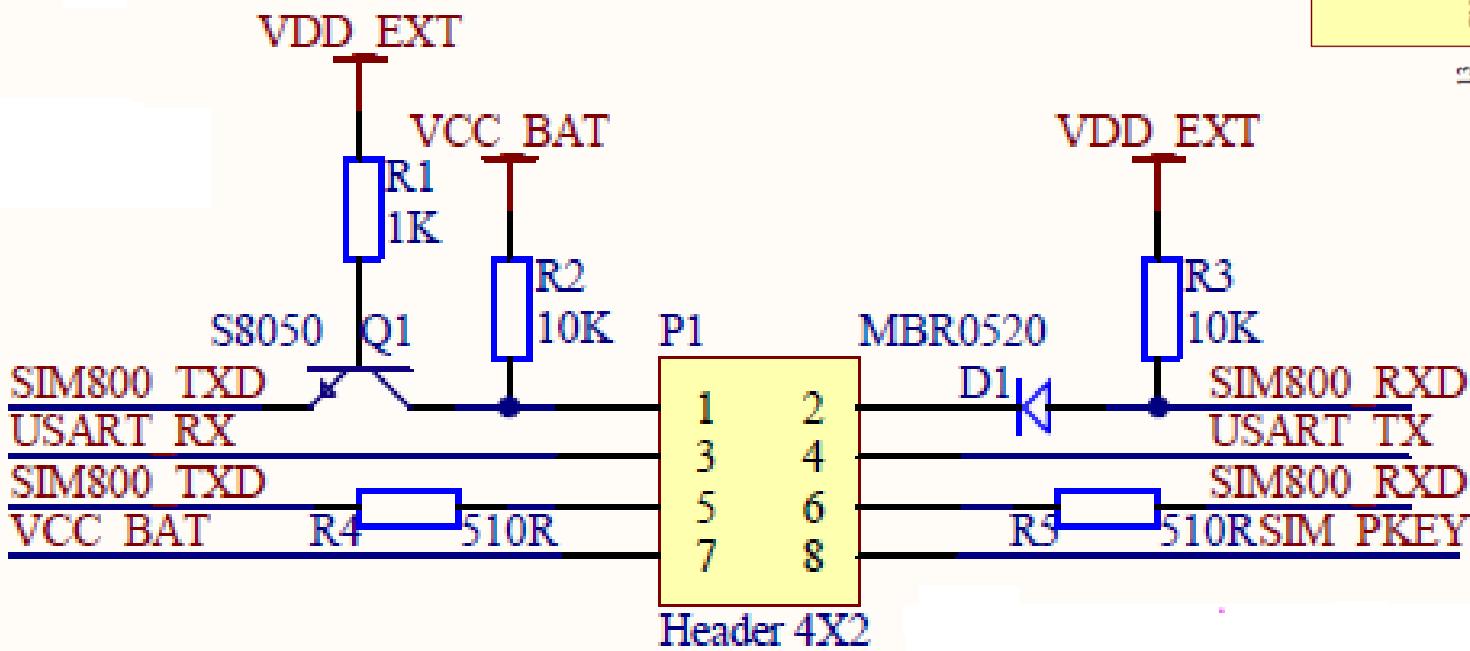
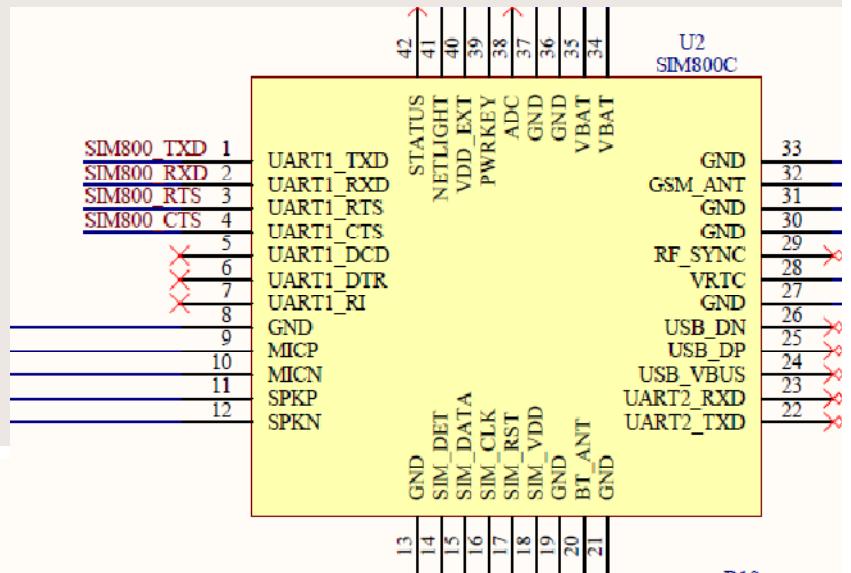


- 如左图所示：芯片有两路通讯口：T1和T2的两组发射信号，R1和R2为两组接收信号，IN代表进入SP3232芯片，OUT代表流出SP3232芯片。



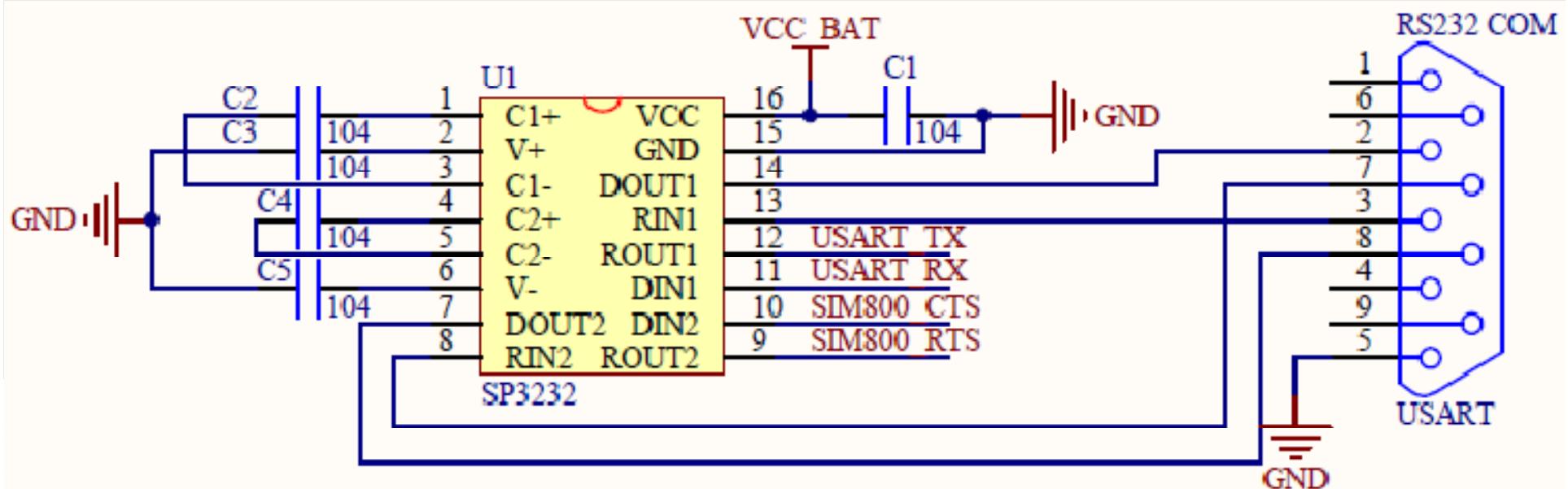
GPRS模块的TX与RX

- Tx信号：从U2 P1输出，默认通过跳线帽连至USART RX
- Rx信号：需从U2P2输入，默认通过跳线帽连至USART TX



RS232模式的硬件链接

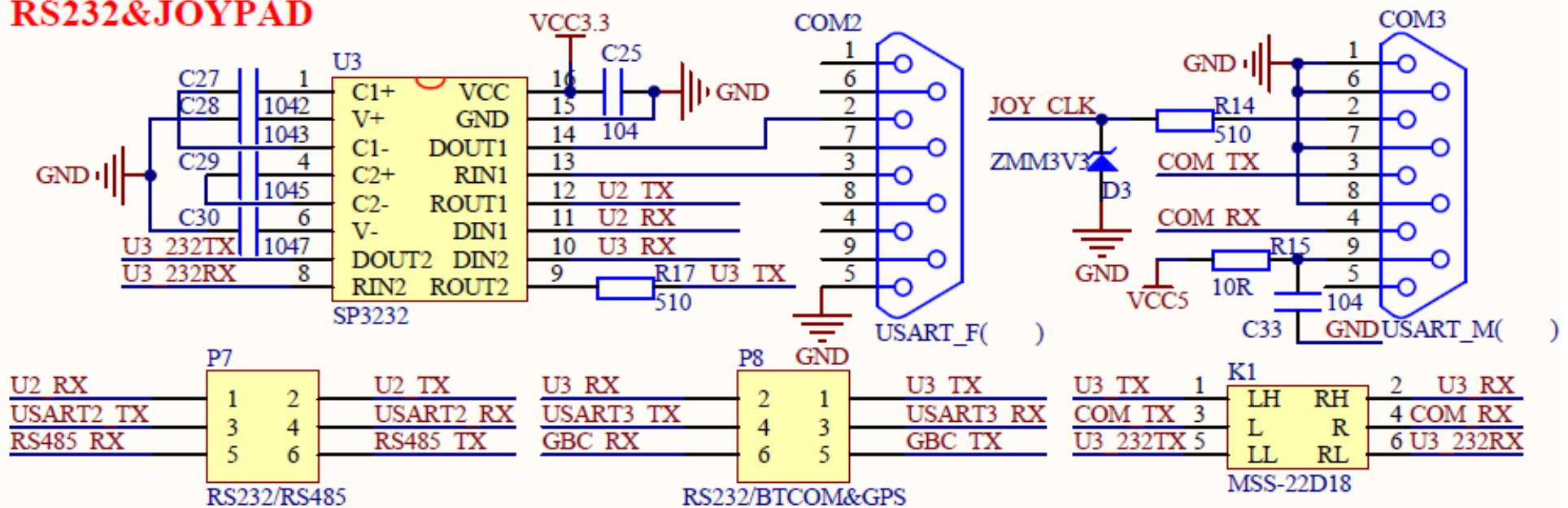
- Tx: U2 P1至U1 pin11入，由U1 pin14出，至com口pin 2。
- Rx: com口pin3 至U1 pin13入，由U1 pin12出，至U2 P2。



RS232的对接

- 尤其需要注意的是：GPRS的Rx接到STM32的开发板对应的是com3的3脚，该脚通过K1的控制接至U3232TX，其源来自于STM32的USART3 TX（通过P8转接）。GPRS的Tx同理连接。

RS232&JOYPAD



CTS与RTS

- CTS (clear to send) : 当MODEM准备好接收终端传来的数据，并向前发送时，使该信号有效，通知终端开始沿发送数据线TxD发送数据。
- RTS (request to send) : 即当终端准备要接收 MODEM传来的数据时，使该信号有效（ON状态），请求MODEM发送数据。它用来控制MODEM是否要进入发送状态。
- 同理，可知CTS在GPRS模块中接成输出模式，RTS接成输入模式。



共模干扰

- 接口使用一根信号线和一根信号返回线而构成**共地**的传输形式，这种共地传输容易产生共模干扰，所以抗噪声干扰性弱。
- 共模干扰：共模干扰指的是干扰电压在信号线及其回线（一般称为信号地线）上的幅度相同，即名义上的地线并不是真正的地，而是有一个特定大小的干扰电压。为消除此电压的影响，高性能电路中通常会使用差分信号来传输信号。
- 切记GPRS模块与STM32开发板要**共地**，否则浮动的共模电压会导致巨大的干扰。



开发板与模块的链接（直连模式）

- 亦可无需通过RS232电平的转换，直接将对应连接。
- 即将U2的pin1 (STXD, Tx)直接接至STM32的USART3_RX; 将U2的pin2 (SRXD, Rx) 直接接至STM32的USART3_TX。

ATK-SIM800C GSM 模块与开发板连接关系

ATK-SIM800C GSM 模块	GND	STXD	SRXD
战舰 V3 开发板	GND	PB11	PB10

- PB10对应STM32的USART3_TX。
- PB11对应STM32的USART3_RX

REMOTE_IN	PB9	140	PB8/TIM4_CH3/SDIO_D4
USART3_TX	PB10	69	PB9/TIM4_CH4/SDIO_D5
USART3_RX	PB11	70	PB10/I2C2_SCL/USART3_TX
F_CS	PB12	73	PB11/I2C2_SDA/USART3_RX



心跳包的测试

- 通过心跳包的测试保证GRRS模块连接正常。

