一、串口

1.串口概述

串口是单片机中最常用也是最简单的一种通信方式

通信：两个或两个以上的设备进行数据交换

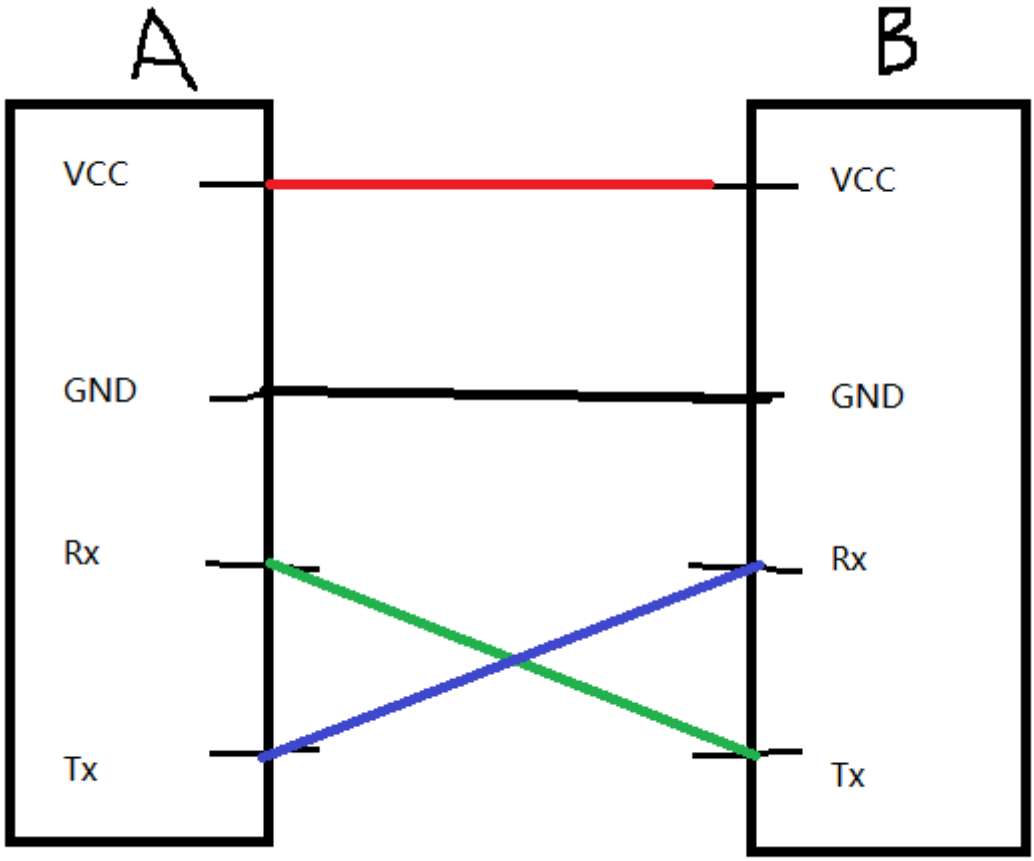
串口是用于两个设备之间的异步全双工通信

异步-》两个设备不需要共时钟

全双工-》两个设备之间服务于数据交换的“线”有两根

Tx：数据发送端，用于发送数据

Rx：数据接收端，用于接收数据



在使用串口进行通信时，要求通信双方必须在“同频道”

“同频道” =》 相同的通信协议

串口（USART)约定：通信时数据必须以“帧”的形式传递

串口的一帧数据包括：起始位 + 数据位 + 校验位 + 停止位

其中：

1）起始位：固定是1个周期的低电平信号

2）数据位：可由通信双方自行约定是 5 ~ 9 bits

3）校验位：串口采用的是奇偶校验，可由通信双方自行约定

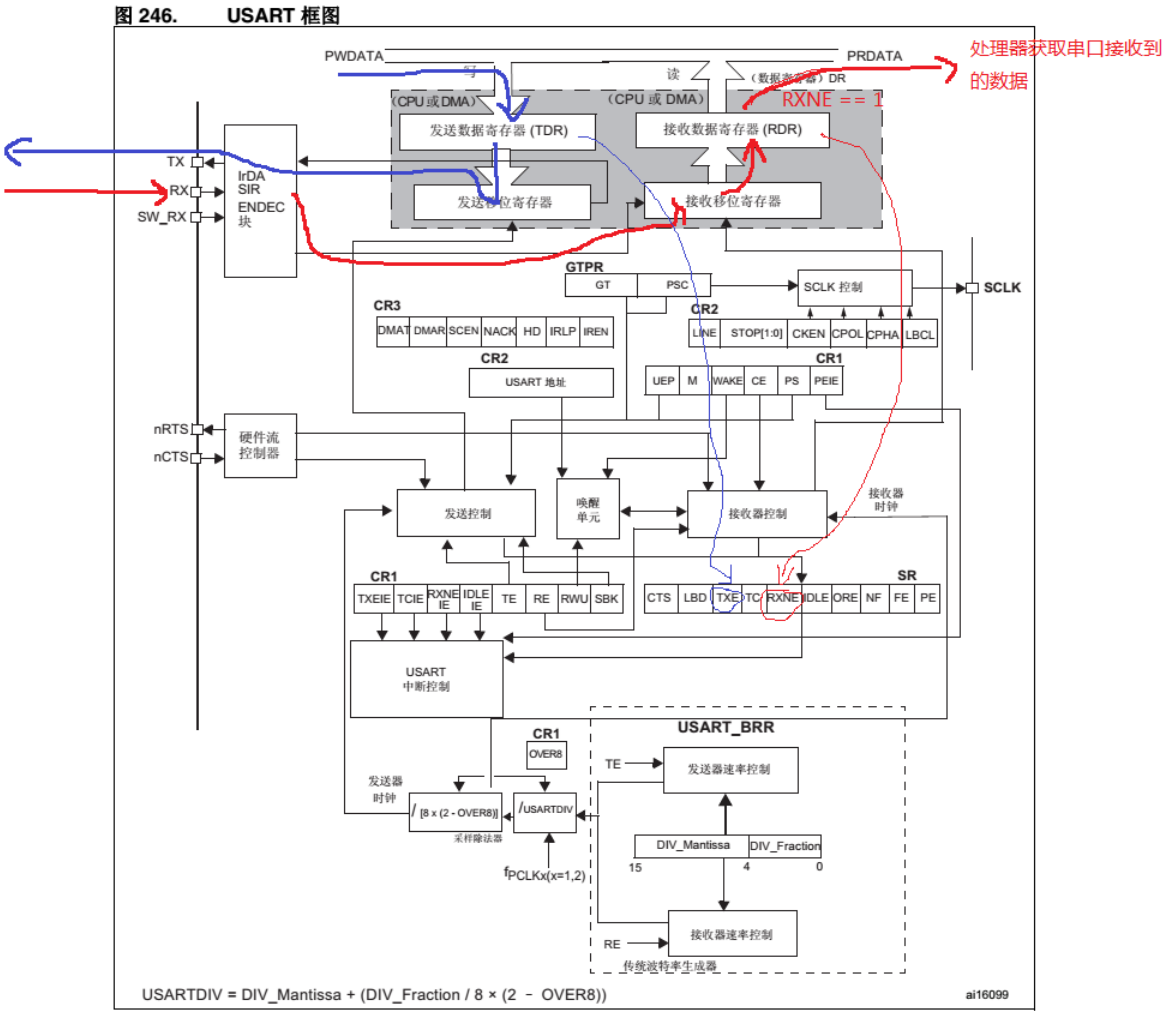
4）停止位：可选的 0.5 ~ 2 个周期的高电平

同时，为了同步通信双发的收发速度，还需要约定每秒钟传输的数据帧的数量，称为 波特率，典型波特率有 9600 115200 57600 ……

通信双发的帧格式 和 波特率必须一致

2.STM32F4xx 串口控制器

单片机中通常会集成有串口的控制器，用户通常只需要通过软件配置串口控制器就可以利用串口进行通信了！！



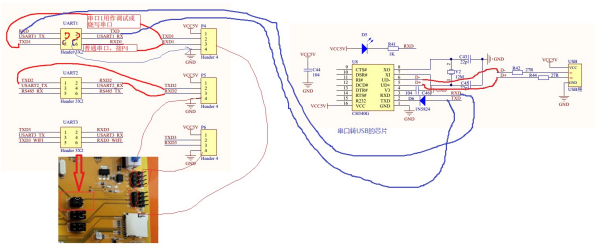
SR：状态寄存器，每个比特位标志了串口控制器中不同的状态变化

RXNE：接收数据寄存器非空标志 1表示RDR寄存器中有数据，可以读取，0表示RDR寄存器中没有数据

TXE：发送数据寄存器为空标志 1表示TDR寄存器中没有数据，可以发送，0表示TDR寄存器中有数据不能发送（覆盖上一次发送的数据）

3.STM32F4xx 中的串口实现

以STM32F4xx USART1（串口1）与 PC通信为例

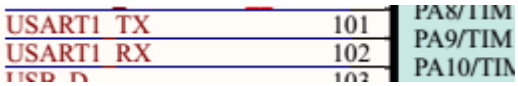


也就是说，当UART1的跳线帽接 1-3 和 2-4时，STM32的USART1 与 PC机就可以通过USB线通信（必须 烧写或 做调试串口）

配置USART1作为调试串口与PC通信

1）配置Rx和Tx引脚

STM32中串口的Tx和Rx是由GPIO复用功能而来



PA9 –>USART1_Tx

PA10 –> USART1_RX

```
GPIO_InitTypeDef  GPIO_InitStruct;

/* 配置GPIO引脚复用为 Rx Tx */
RCC_AHB1PeriphClockCmd(RCC_AHB1Periph_GPIOA,ENABLE);     //使能GPIOA组时钟

GPIO_InitStruct.GPIO_Pin    = GPIO_Pin_9 | GPIO_Pin_10;
GPIO_InitStruct.GPIO_Mode   = GPIO_Mode_AF;//复用功能模式
GPIO_InitStruct.GPIO_OType  = GPIO_OType_PP;
GPIO_InitStruct.GPIO_PuPd   = GPIO_PuPd_NOPULL;
GPIO_InitStruct.GPIO_Speed  = GPIO_Speed_100MHz;
GPIO_Init(GPIOA,&GPIO_InitStruct);

```

```
13 GPIO_PinAFConfig(GPIOA,GPIO_PinSource9,GPIO_AF_USART1); //PA9 -> USART1_Tx
14 GPIO_PinAFConfig(GPIOA,GPIO_PinSource10,GPIO_AF_USART1);//PA10-> USART1_Rx
15
```

2）配置串口初始化

```
/* 配置USART1 */
RCC_APB2PeriphClockCmd(RCC_APB2Periph_USART1,ENABLE);   //使能USART1的时钟（USART1在APB2总线上）

USART_InitStruct.USART_BaudRate = 9600;   //指定波特率
USART_InitStruct.USART_WordLength= USART_WordLength_8b;    //指定数据位长度(通常是8bits)
USART_InitStruct.USART_Parity   = USART_Parity_No;    //指定校验方式(通常不校验)
USART_InitStruct.USART_StopBits = USART_StopBits_1;   //指定停止位(通常是1个停止)
USART_InitStruct.USART_Mode     = USART_Mode_Tx|USART_Mode_Rx;    //指定收发模式
USART_InitStruct.USART_HardwareFlowControl = USART_HardwareFlowControl_None;//指定硬件控制流（通常不要）
USART_Init(USART1,&USART_InitStruct);

/* 开启串口，就可以开始通信 */
USART_Cmd(USART1,ENABLE);
```

3）串口收发函数

```
1  //通过USART1发送1个字节
2  void usart1_send_byte(char data)
3  {
4      //USART_GetFlagStatus 用来获取串口SR寄存器中的指定标志位
5      //获取TXE标志，判断其是否被设置（SET）
6
   while(USART_GetFlagStatus(USART1,USART_FLAG_TXE) !=  SET);
7      //USART_SendData 用来通过指定串口发送数据
8      USART_SendData(USART1,data);
9  }
10
11 char usart1_recv_byte(void)
12 {
13     char ch = 0;
14
15
   while(USART_GetFlagStatus(USART1,USART_FLAG_RXNE) != SET);
16     ch = USART_ReceiveData(USART1);
17
18     return ch;
19 }
```

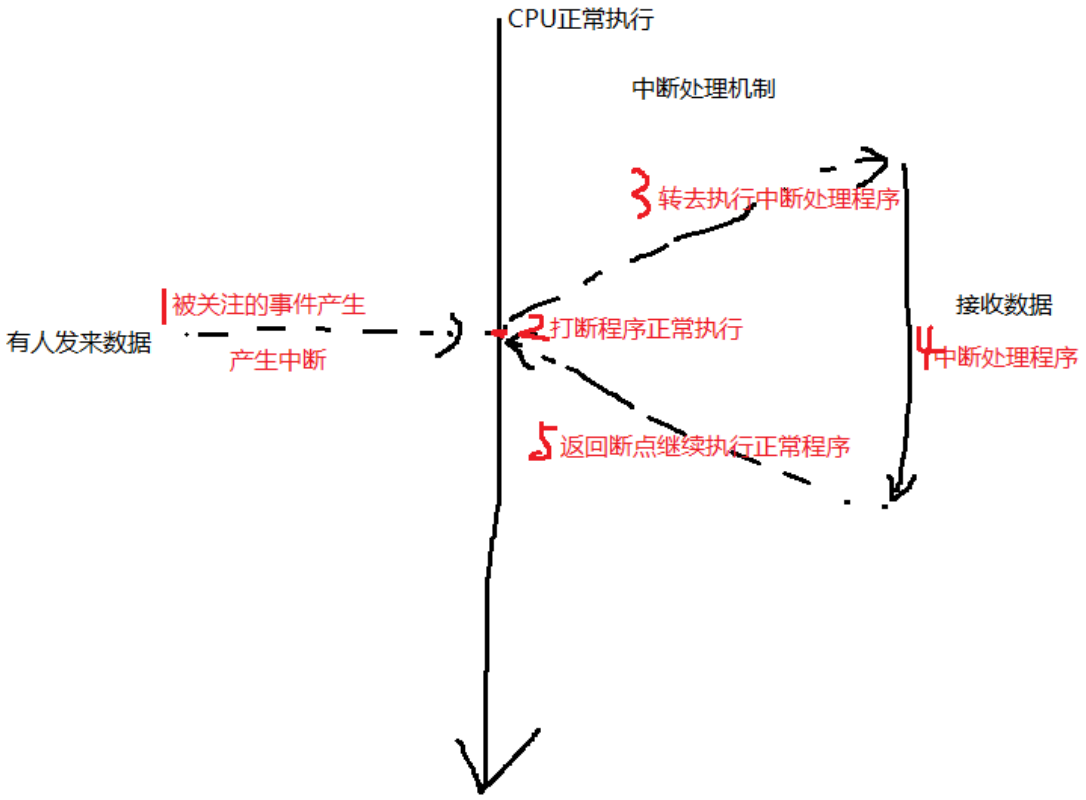
测试代码：

```
    char str[] = "oyhldsg";
    int i = 0;

    led_init();
    debug_usart();

    /* Infinite loop */
    while (1)
    {

GPIO_ResetBits(GPIOF,GPIO_Pin_9);
        Delay(1000);
        GPIO_SetBits(GPIOF,GPIO_Pin_9);
        Delay(1000);

        for(i=0;i<7;i++)
        {
            usart1_send_byte(str[i]);
        }
    }
```

由于USART1的跳线帽接 1-3和2-4，也就是通过USART1发送的数据经由USB线发送给了PC机

此时，在PC上运行串口调试助手，则可以接收这些数据

4）串口接收中断



中断是指：当某件紧急的事件产生后，会打断CPU的正常执行顺序，转去执行中断处理程序，当中断处理程序执行完后，又回到原来被打断的位置继续执行的过程，被称为中断

在串口应用中，我们不知道对方什么时候会发数据过来

所以上午讲的接收函数，并不适用 =》可能导致程序一直阻塞在ｗｈｉｌｅ

因此，我们需要借助中断 来实现串口的数据接收

串口中断配置：

```
1
2 /* 配置串口1的接收中断 */
3 USART_ITConfig(USART1,USART_IT_RXNE,ENABLE); //USART_IT_RXNE接收数据寄存器不为空时产生中断
```

配置了中断后，必须要配置 NVIC(中断控制器)

```
1 /* 配置NVIC中断控制器 */
2 NVIC_InitTypeDef  NVIC_InitStruct;
3
4 NVIC_InitStruct.NVIC_IRQChannel = USART1_IRQn;    //指定中断通道 xxx_IRQn
5 NVIC_InitStruct.NVIC_IRQChannelPreemptionPriority   =   2;//抢占优先级
6 NVIC_InitStruct.NVIC_IRQChannelSubPriority          =   2;//子优先级
7 NVIC_InitStruct.NVIC_IRQChannelCmd  =  ENABLE;
8 NVIC_Init(&NVIC_InitStruct);
```

上述配置完成后，一旦对法发送数据到STM32就会触发串口1的中断

此时还需要一个串口1的中断处理函数

```c
char ch = 0;
void USART1_IRQHandler(void)
{
    //判断是由 RXNE接收数据寄存器非空 产生的中断

if(USART_GetITStatus(USART1,USART_IT_RXNE) == SET)
    {
        ch = USART_ReceiveData(USART1); //接收1个字节的数据

USART_ClearITPendingBit(USART1,USART_IT_RXNE);//清除中断标志
    }
}
```

任务：

请根据USART1串口的示例代码，根据HC-05蓝牙模块的使用说明，完成USART2串口代码的编写，并将HC-05蓝牙模块连接在串口2，通过HC-05实现手机蓝牙对STM32小车的控制

要求

HC-05的串口协议为：

波特率： 9600

数据位： 8

停止位： 1

校验位： 不要

硬件控制流： 不要

通信协议约定如下：

通信字符	**含义**
‘0’	停止
‘2’	前进
‘8’	后退
‘4’	左转
‘6’	右转
……	

注意：

使用串口2连接HC-05时，UART2跳线帽接 3-5 和 4-6

USART2串口的接口为 P6组引脚