

✓ 1.通信接口背景知识



◆ 处理器与外部设备通信的两种方式:

● 并行通信

- 传输原理：数据各个位同时传输。
- 优点：速度快
- 缺点：占用引脚资源多

● 串行通信

- 传输原理：数据按位顺序传输。
- 优点：占用引脚资源少
- 缺点：速度相对较慢

✓ 1.通信接口背景知识



◆ 串行通信:

按照数据传送方向，分为:

◆ 单工:

数据传输只支持数据在一个方向上传输

◆ 半双工:

允许数据在两个方向上传输，但是，在某一时刻，只允许数据在一个方向上传输，它实际上是一种切换方向的单工通信;

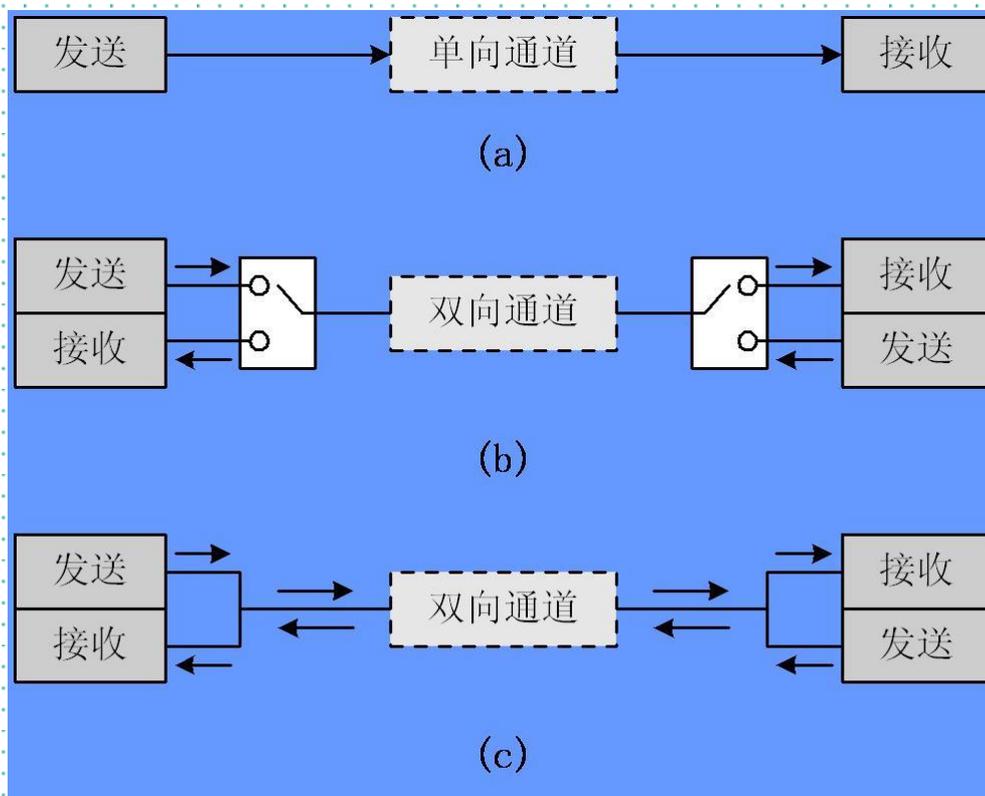
◆ 全双工:

允许数据同时在两个方向上传输，因此，全双工通信是两个单工通信方式的结合，它要求发送设备和接收设备都有独立的接收和发送能力。

✓ 1.通信接口背景知识



◆ 串行通信三种传送方式:



✓ 1.通信接口背景知识



◆ 串行通信的通信方式

- **同步通信**：带时钟同步信号传输。

-SPI, IIC通信接口

- **异步通信**：不带时钟同步信号。

-UART(通用异步收发器),单总线

✓ 1.通信接口背景知识



◆常见的串行通信接口：

通信标准	引脚说明	通信方式	通信方向
UART (通用异步收发器)	TXD:发送端 RXD:接受端 GND:公共地	异步通信	全双工
单总线 (1-wire)	DQ:发送/接受端	异步通信	半双工
SPI	SCK:同步时钟 MISO:主机输入, 从机输出 MOSI:主机输出, 从机输入	同步通信	全双工
I2C	SCL:同步时钟 SDA:数据输入/输出端	同步通信	半双工

✓ 2.STM32串口通信基础



◆ STM32的串口通信接口

- **UART:通用异步收发器**
- **USART:通用同步异步收发器**

大容量**STM32F10x**系列芯片，包含**3个USART**和**2个UART**

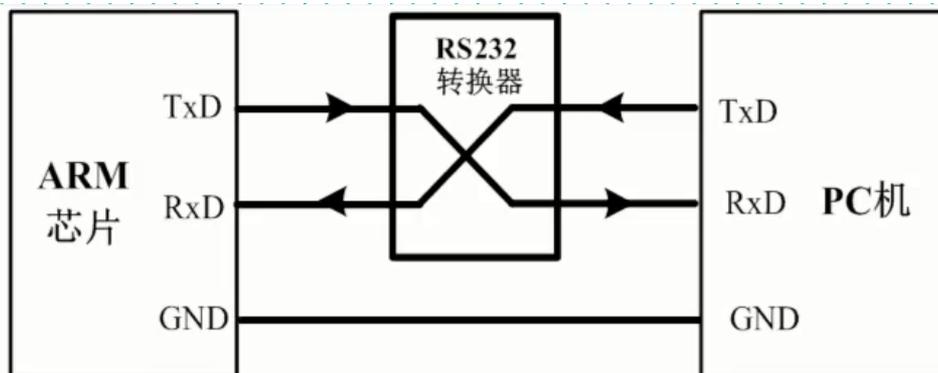
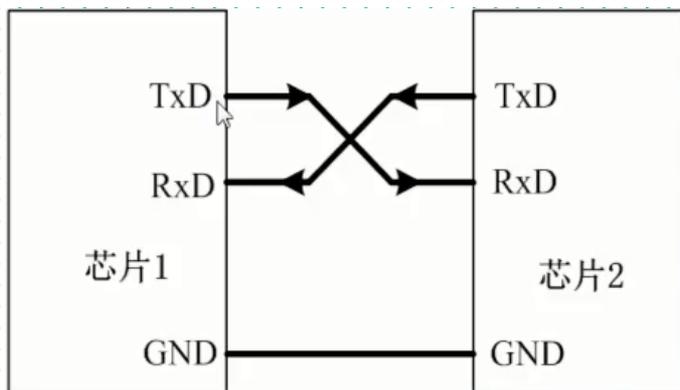
✓ 2.STM32串口通信基础



◆ UART异步通信方式引脚连接方法:

-RXD:数据输入引脚。数据接受。

-TXD:数据发送引脚。数据发送。



✓ 2.STM32串口通信基础



◆ UART异步通信方式引脚:

-RXD:数据输入引脚。数据接受。

-TXD:数据发送引脚。数据发送。

串口号	RXD	TXD
1	PA10	PA9
2	PA3	PA2
3	PB11	PB10
4	PC11	PC10
5	PD2	PC12

✓ 2.STM32串口通信基础



◆UART异步通信方式特点:

- 全双工异步通信。
- 分数波特率发生器系统，提供精确的波特率。
 - 发送和接受共用的可编程波特率，最高可达**4.5Mbits/s**
- 可编程的数据字长度（**8位或者9位**）；
- 可配置的停止位（支持**1或者2位**停止位）；
- 可配置的使用**DMA**多缓冲器通信。
- 单独的发送器和接收器使能位。
- 检测标志：① 接受缓冲器 ②发送缓冲器空 ③传输结束标志
- 多个带标志的中断源。触发中断。
- 其他：校验控制，四个错误检测标志。

✓ 2.STM32串口通信基础



◆ 串口通信过程

数据接收过程:



数据发送过程:



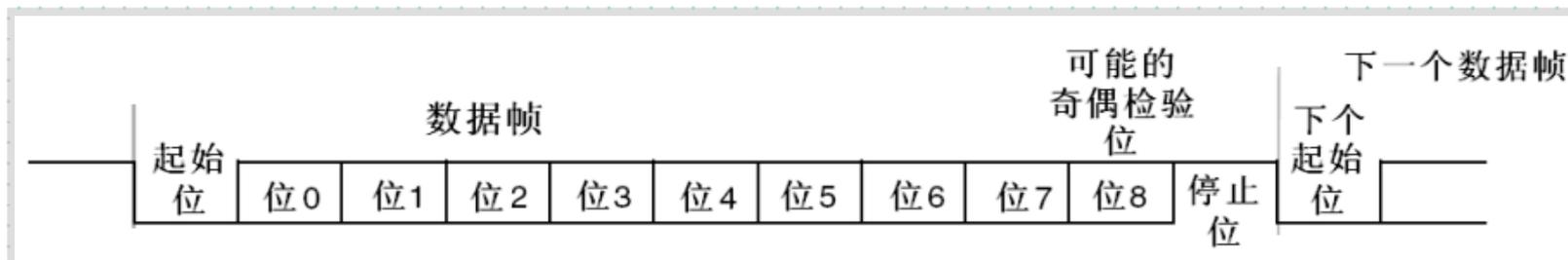
✓ 2.STM32串口通信基础



◆ STM32串口异步通信需要定义的参数:

- ① 起始位
- ② 数据位 (8位或者9位)
- ③ 奇偶校验位 (第9位)
- ④ 停止位 (1,15,2位)
- ⑤ 波特率设置

■ 范例:

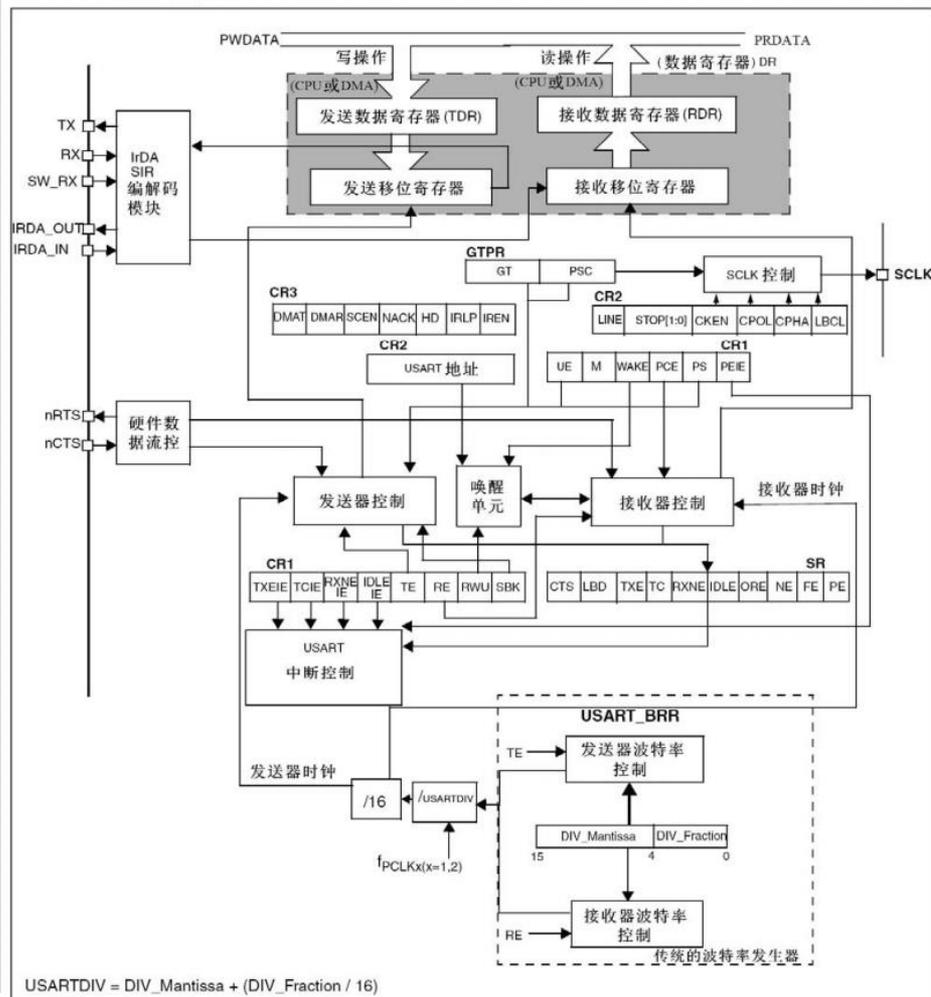




2.STM32串口通信基础



图248 USART框图



第3章 STM32单片机 串口编程



3.1 STM32单片机的 USARTx 串口和管脚

3.2 STM32单片机的 USARTx 串口配置

3.3 STM32单片机的 USARTx 串口编程步骤

3.1 STM32单片机的 USART_x串口和管脚



USART ?

**USART: Universal
Synchronous/Asynchronous
Receiver/Transmitter,**
通用同步/异步接收和发送器

比较:



MCS51单片机: 拥有1个UART(通用异步接收和发送器)

ARM9 S3C2410: 拥有3个UART(通用异步接收和发送器)

Cortex-M3 STM32单片机: 拥有3个USART(通用同步/异步接收和发送器)



开始

终止



图7-5 同步通信数据格式

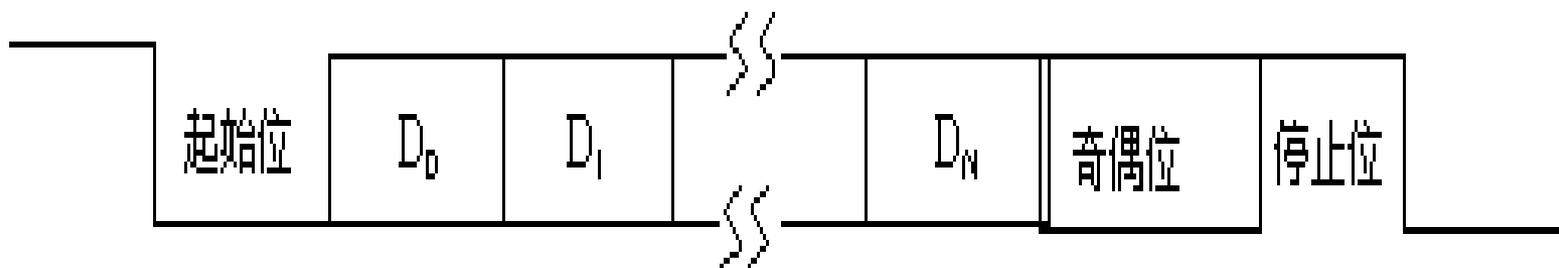


图7-4 异步串行通信数据格式



“异步通信”是一种很常用的通信方式。异步通信在发送字符时，所发送的字符之间的时间间隔可以是任意的。当然，接收端必须时刻做好接收的准备（如果接收端主机的电源都没有加上，那么发送端发送字符就没有意义，因为接收端根本无法接收）。发送端可以在任意时刻开始发送字符，因此必须在每一个字符的开始和结束的地方加上标志，即加上开始位和停止位，以便使接收端能够正确地将每一个字符接收下来。异步通信的好处是通信设备简单、便宜，但传输效率较低（因为开始位和停止位的开销所占比例较大）。

“同步通信”的通信双方必须先建立同步，即双方的时钟要调整到同一个频率。收发双方不停地发送和接收连续的同步比特流。





1、单工

单工就是指 A 只能发信号，而 B 只能接收信号，通信是单向的，就象灯塔之于航船——灯塔发出光信号而航船只能接收信号以确保自己行驶在正确的航线上。

2、半双工

半双工就是指 A 能发信号给 B，B 也能发信号给 A，但这两个过程不能同时进行。最典型的例子就象我们在影视作品中看到的对讲机一样：

007：呼叫总部，请求支援，OVER

总部：收到，增援人员将在 5 分钟内赶到，OVER

007：要 5 分钟这么久？！要快呀！OVER

总部：……

GAME OVER

在这里，每方说完一句话后都要说个 OVER，然后切换到接收状态，同时也告之对方——你可以发言了。如果双方同时处于收状态，或同时处于发状态，便不能正常通信了。

3、全双工

全双工比半双工又进了一步。在 A 给 B 发信号的同时，B 也可以给 A 发信号。典型的例子就是打电话。

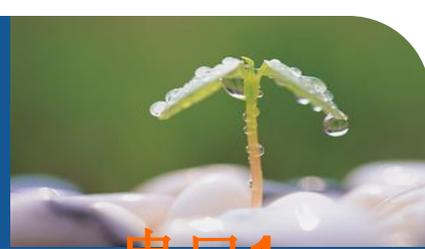
A：我跟你说呀……

B：你先听我说，情况是这样的……

A 和 B 在说的同时也能听到对方说的内容，这就是全双工。



STM32F103Cx: 拥有3个USART



串口1

PC口
3脚

PD口
2脚

PA口
16脚

USART2_TX:

串口2

USART2_RX:

USART3_TX:
USART3_RX:

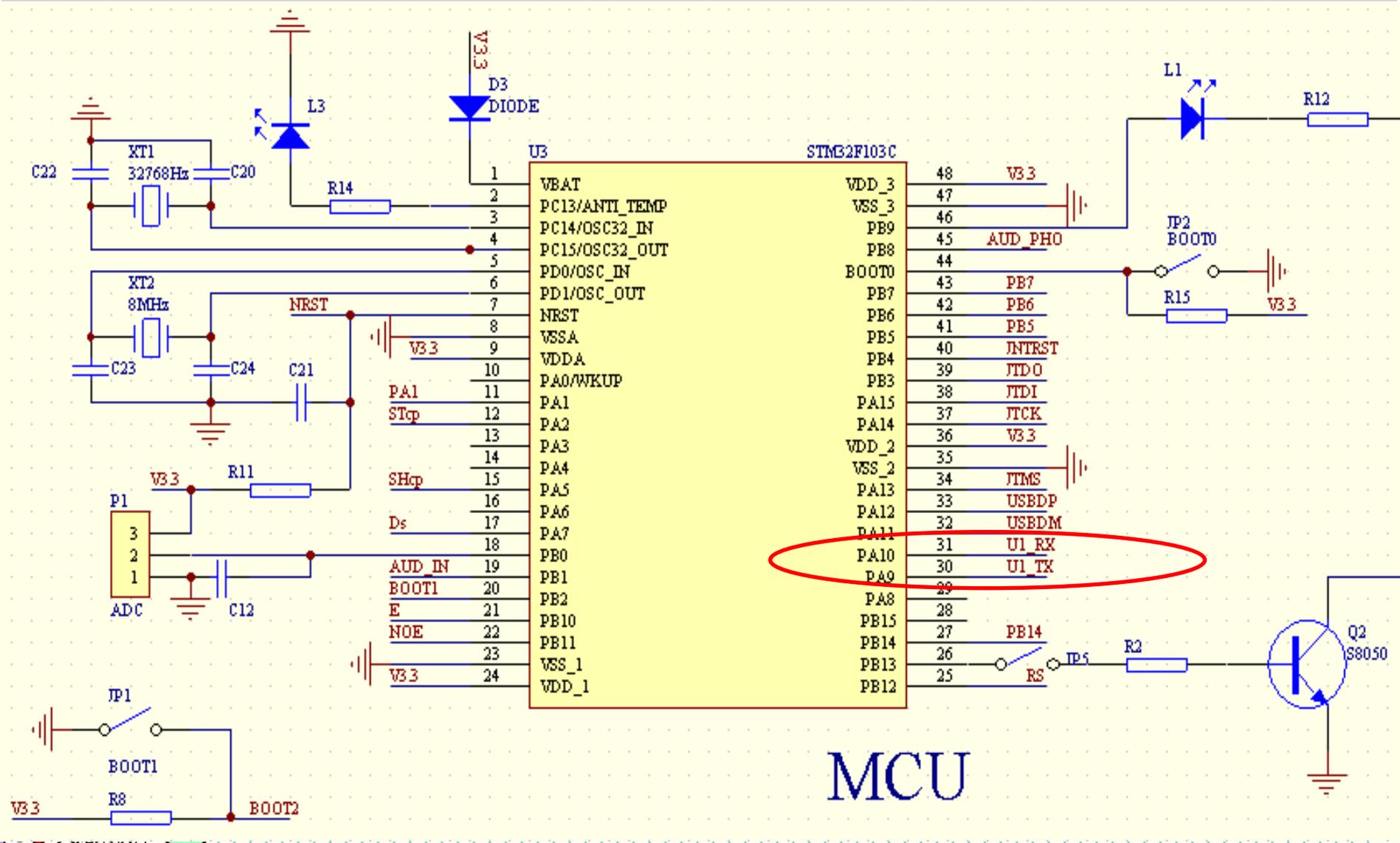
串口3

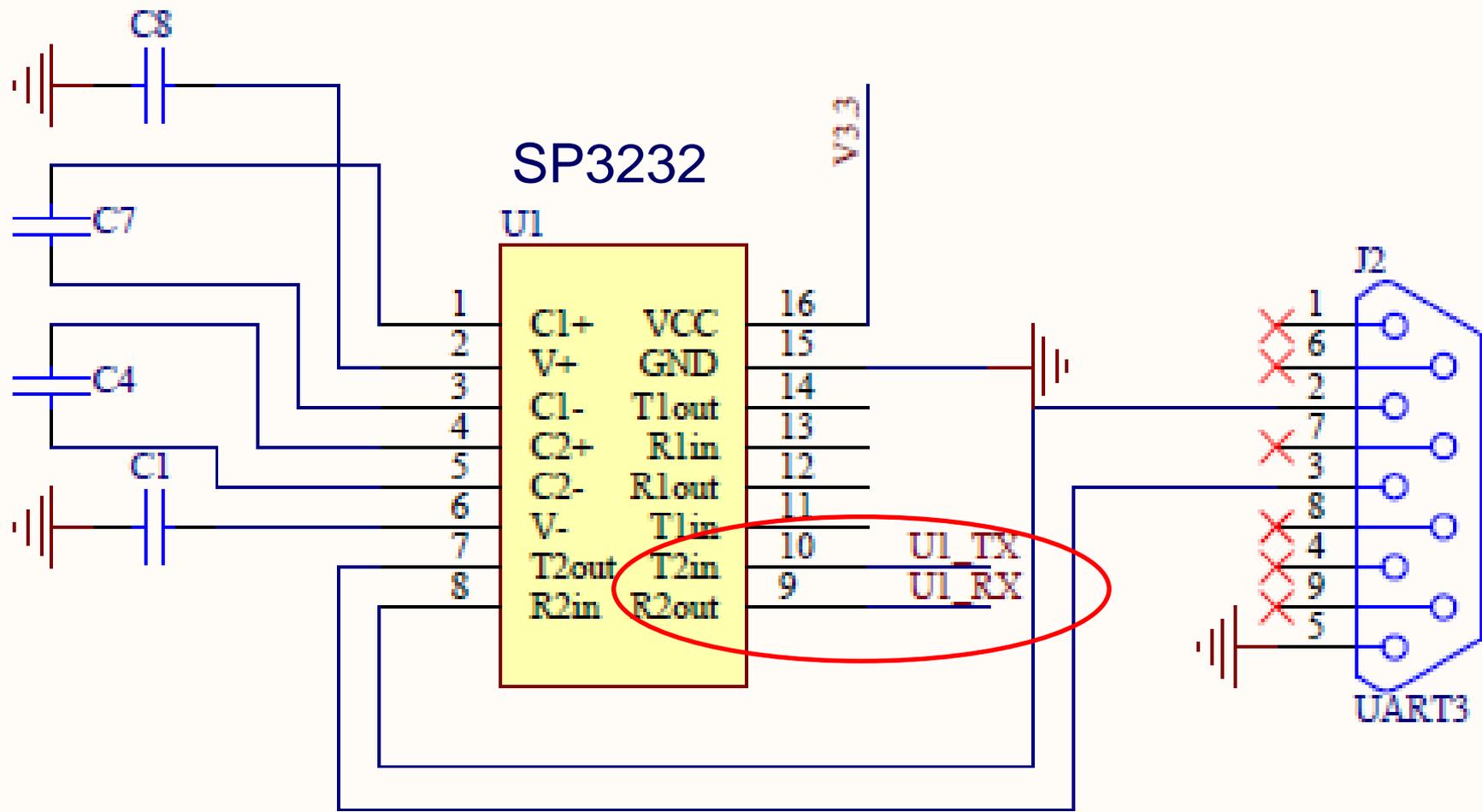
:USART1_RX
:USART1_TX

PB口
16脚



STM32的CPU 引脚，通过PA 端口的两个引脚PA9 和PA10，连接到一个SP3232 芯片，或者MAX232 芯片。然后再连接到DB9 串口座上。





每个USART口:共有7个设置寄存器



- ① 状态寄存器(USART_SR)
- ② 数据寄存器(USART_DR):它是由两个寄存器组成的,一个给发送用(发送寄存器 TDR),一个给接收用(接收寄存器 RDR)
- ③ 一个波特率寄存器(USART_BRR)
- ④ 一个控制寄存器1(USART_CR1)
- ⑤ 一个控制寄存器2(USART_CR2)
- ⑥ 一个控制寄存器3(USART_CR3)
- ⑦ 一个保护时间和预分频寄存器(USART_GTPR)

寄存器	描述
SR	USART 状态寄存器
DR	USART 数据寄存器
BRR	USART 波特率寄存器
CR1	USART 控制寄存器 1
CR2	USART 控制寄存器 2
CR3	USART 控制寄存器 3
GTPR	USART 保护时间和预分频寄存器



数据缓冲器SBUF：实际是由发送SBUF和接收SBUF组成。

发送SBUF和接收SBUF共用一个地址99H。

1) 发送SBUF存放待发送的8位数据，**写入SBUF将同时启动发送**

发送指令：`MOV SBUF, A`

2) 接收SBUF存放已接收成功的8位数据，供CPU读取。

读取串口接收数据指令：

`MOV A, SBUF`

在固件函数库的“stm32f10x_map.h”文件中，对应的定义



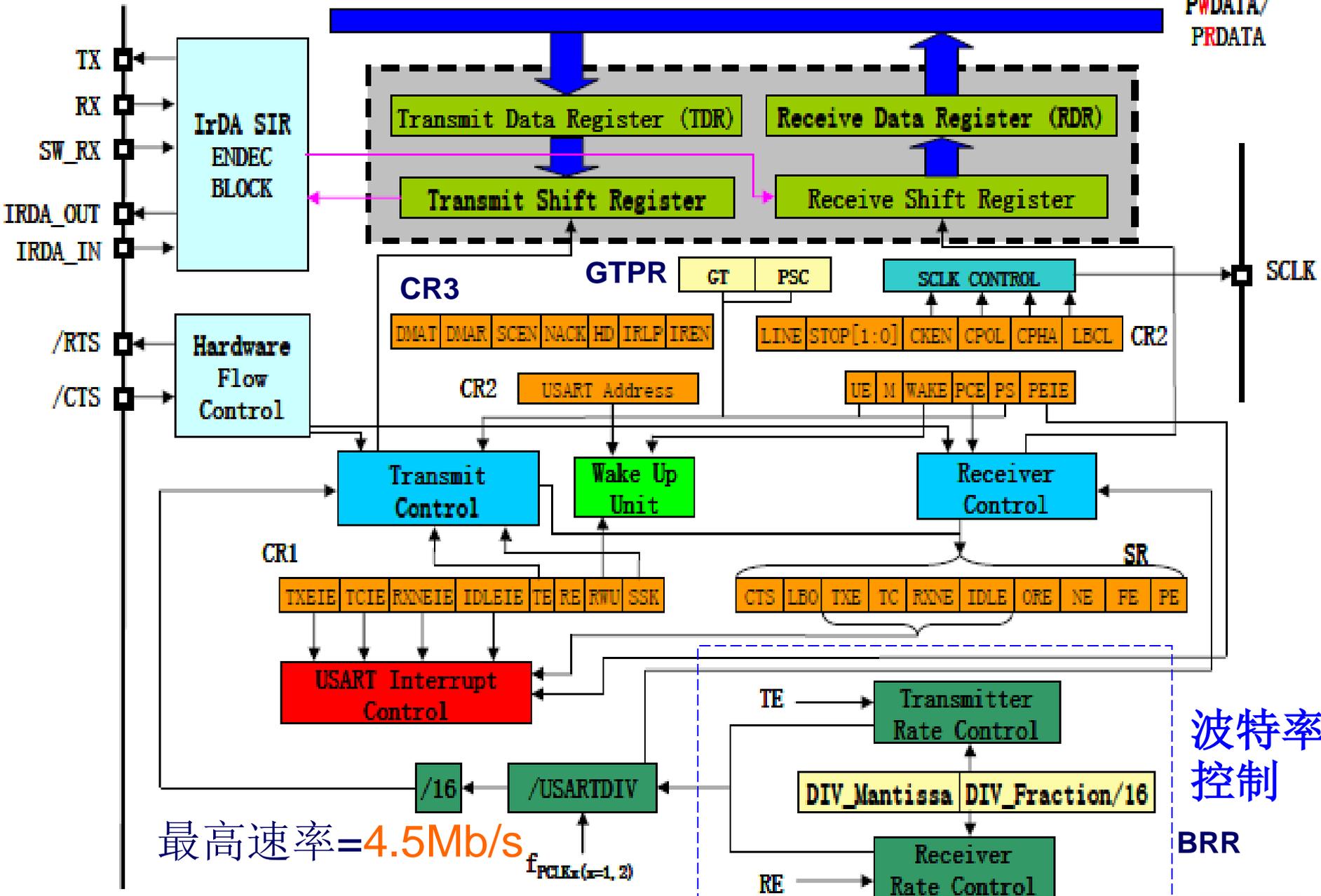
```
/*----- Universal Synchronous Asynchronous Receiver Transmitter -----*/  
typedef struct  
{  
    vu16 SR;  
    u16 RESERVED0;  
    vu16 DR;  
    u16 RESERVED1;  
    vu16 BRR;  
    u16 RESERVED2;  
    vu16 CR1;  
    u16 RESERVED3;  
    vu16 CR2;  
    u16 RESERVED4;  
    vu16 CR3;  
    u16 RESERVED5;  
    vu16 GTPR;  
    u16 RESERVED6;  
} USART_TypeDef; //用结构体USART_TypeDef定义USARTx串
```

（即定义 USARTx串口的7个设置寄存器）

STM32的USART功能模块图

CPU

PWDATA/
PRDATA



STM32单片机的USART串口：采用分数波特率发生器，
串行发送、接收数据的最高速率= $72\text{M}/16=4.5\text{Mb/s}$



MCS-51单片机的串行通讯：适用于传送距离不大于15m，
速度不高于20kb/s的本地设备之间通信的场合。

3.2 STM32单片机的 USARTx串口配置



1、USARTx

串口 定义

2、USART_InitTypeDef

初始化串口参数 定义

3、USART_Init

初始化串口 定义

4、在使用USART串口时，首先要使能该外设对应的时钟



1、编程时，USARTx串口的具体配置是从**USARTx寄存器组**开始。首先，用结构体**USART_TypeDef** 定义 **USARTx寄存器组**：

在文件“stm32f10x_map.h”中，定义如下：

```
/*----- Universal Synchronous Asynchronous Receiver Transmitter -----*/
typedef struct
{
    vu16 SR;
    u16 RESERVED0;
    vu16 DR;
    u16 RESERVED1;
    vu16 BRR;
    u16 RESERVED2;
    vu16 CR1;
    u16 RESERVED3;
    vu16 CR2;
    u16 RESERVED4;
    vu16 CR3;
    u16 RESERVED5;
    vu16 GTPR;
    u16 RESERVED6;

```

USARTx串口的**7个**设置寄存器

} **USART_TypeDef**; //用结构体**USART_TypeDef**定义**USARTx**串口，
资料整理 火龙果软件  //或称用结构体**USART_TypeDef**定义 **USARTx**寄存器组

```
/* Peripheral base address in the bit-band region */
```

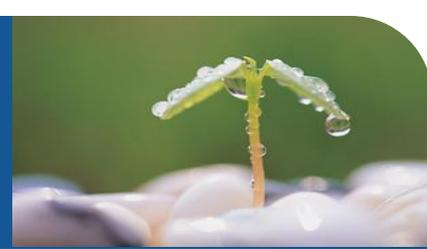
```
#define PERIPH_BASE      ((u32)0x40000000)
```

```
/* Peripheral memory map */
```

```
#define APB1PERIPH_BASE  PERIPH_BASE
```

```
#define APB2PERIPH_BASE  (PERIPH_BASE + 0x10000)
```

```
#define AHBPERIPH_BASE   (PERIPH_BASE + 0x20000)
```



USARTx寄存器组 的首地址:

```
#define USART1_BASE      (APB2PERIPH_BASE + 0x3800) 0x40013800  
#define USART2_BASE      (APB1PERIPH_BASE + 0x4400) 0x40014400  
#define USART3_BASE      (APB1PERIPH_BASE + 0x4800) 0x40014800
```

```
#ifdef _USART1
#define USART1
#endif /*_USART1 */
```

```
((USART_TypeDef *) USART1_BASE)
```



```
#ifdef _USART2
#define USART2
#endif /*_USART2 */
```

```
((USART_TypeDef *) USART2_BASE)
```

```
#ifdef _USART3
#define USART3
#endif /*_USART3 */
```

```
((USART_TypeDef *) USART3_BASE)
```



2、USART_InitTypeDef 初始化串口参数 定义

在文件“stm32f10x_usart.h”中，定义：



```
/* USART Init Structure definition */
```

```
typedef struct
```

```
{
```

```
    u32 USART_BaudRate;
```

```
    u16 USART_WordLength;
```

```
    u16 USART_StopBits;
```

```
    u16 USART_Parity;
```

```
    u16 USART_Mode;
```

```
    u16 USART_HardwareFlowControl;
```

```
} USART_InitTypeDef; //用于初始化USARTx串口的参数（  
包括波特率、字长—即数据位、停止位、奇偶效验位、工作模  
式、硬件流控制）
```



Table 708. 描述了结构体USART_InitTypeDef 在同步和异步模式下使用的不同成员。

Table 708. USART_InitTypeDef 成员 USART 模式对比

成员	异步模式	同步模式
USART_BaudRate	X	X
USART_WordLength	X	X
USART_StopBits	X	X
USART_Parity	X	X
USART_HardwareFlowControl	X	X
USART_Mode	X	X
USART_Clock		X
USART_CPOL		X
USART_CPHA		X
USART_LastBit		X

比较 结构体 `USART_TypeDef` 和 `USART_InitTypeDef` :



```
typedef struct
{
    vu16 SR;
    u16 RESERVED0;
    vu16 DR;
    u16 RESERVED1;
    vu16 BRR;
    u16 RESERVED2;
    vu16 CR1;
    u16 RESERVED3;
    vu16 CR2;
    u16 RESERVED4;
    vu16 CR3;
    u16 RESERVED5;
    vu16 GTPR;
    u16 RESERVED6;
}
```

2 USARTx串口的7个设置寄存器

1

} `USART_TypeDef`; //用结构体 `USART_TypeDef` 定义 USARTx 串口,
//或称用结构体 `USART_TypeDef` 定义 USARTx 寄存器组

```
typedef struct
{
    u32 USART_BaudRate;
    u16 USART_WordLength;
    u16 USART_StopBits;
    u16 USART_Parity;
    u16 USART_Mode;
    u16 USART_HardwareFlowControl;
}
```

} `USART_InitTypeDef`; //用于初始化 USARTx 串口的参数



3、USART_Init

初始化串口 定义



功能：根据 **USART_InitTypeDef** 中指定的参数，初始化外设

USARTx 串口

函数名	USART_Init
函数原形	void USART_Init(USART_TypeDef* USARTx, USART_InitTypeDef* USART_InitStruct)
功能描述	根据 USART_InitStruct 中指定的参数初始化外设 USARTx 寄存器
输入参数 1	USARTx: x 可以是 1, 2 或者 3, 来选择 USART 外设
输入参数 2	USART_InitStruct: 指向结构 USART_InitTypeDef 的指针, 包含了外设 USART 的配置信息。参阅 Section: USART_InitTypeDef 查阅更多该参数允许取值范围
输出参数	无
返回值	无
先决条件	无
被调用函数	无

例如，`USART_InitTypeDef USART_InitStructure;` //定义结构体变量 `USART_InitStructure`，用于初始化外设 **USARTx** 串口的参数

`USART_Init(USART1, &USART_InitStructure);` //初始化 **USARTx** 串口

在文件“stm32f10x_usart.c”中，定义：



```
*****  
* Function Name : USART_Init  
* Description : Initializes the USARTx peripheral according to the specified  
* parameters in the USART_InitStruct .  
* Input : - USARTx: Select the USART or the UART peripheral.  
* This parameter can be one of the following values:  
* - USART1, USART2, USART3, UART4 or UART5.  
* - USART_InitStruct: pointer to a USART_InitTypeDef structure  
* that contains the configuration information for the  
* specified USART peripheral.  
* Output : None  
* Return : None  
*****/  
void USART_Init(USART_TypeDef* USARTx, USART_InitTypeDef*  
USART_InitStruct)  
{  
    u32 tmpreg = 0x00, apbclock = 0x00;  
    u32 integerdivider = 0x00;  
    u32 fractionaldivider = 0x00;  
    u32 usartxbase = 0;  
    RCC_ClocksTypeDef RCC_ClocksStatus;
```

3.3 STM32单片机的 USARTx串口 编程步骤



USARTx串口 编程步骤

第3章 STM32单片机 串口编程

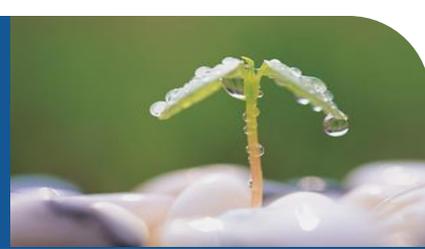


3.1 STM32单片机的 USARTx 串口和管脚

3.2 STM32单片机的 USARTx 串口配置

3.3 STM32单片机的 USARTx 串口编程步骤

作业：



- 1、**UART**和**USART**有何区别？单工、半双工、全双工有何区别？
- 2、请写出使能**EXTI**和**AFIO**外设对应的时钟命令。
- 3、简述你对**STM32**单片机的**USART**串口功能模块图的理解。
- 4、编写程序，实现**STM32**单片机通过串口1 发送一个字符串“**STM32F103CB USART1 TEST!**”，并在**PC**机的超级终端上显示。要求用**printf**函数输出。