应用笔记

Application Note

文档编号：**AN1104**

APM32F4xx 系列软件模拟 USART

版本：**V1.0**

© 珠海极海半导体有限公司

1 引言

在系统使用过程中，可能会遇到系统资源（如串口）不够的情况，此时可以使用 IO 口模拟串口，实现同样的功能。本应用笔记介绍了在 APM32F4xx 系列上通过外部中断和定时器如何实现 IO 口模拟串口。

目录

2 APM32F4xx USART 简介

USART（通用同步异步收发器）是一个可以灵活地与外部设备进行全双工、半双工数据交换的串行通信设备，且同时满足外部设备对工业标准 NRZ 异步串行数据格式的要求。USART 还提供宽范围的波特率选择，且支持多处理器通信。 USART 不仅支持标准的异步收发模式，也支持一些其他的串行数据交换模式，如 LIN 协议、智能卡协议、IrDA SIR ENDEC 规范和硬件流控制模式。USART 还支持使用 DMA 功能，以实现高速数据通信。

2.1 串口模式

串行通讯是指将数据逐位顺序传送的通信方式。串口分为同步串行接口和异步串行接口。

同步模式：一次传输的数据块中包含的数据较多，所以接收时钟与发送时钟严格同步，比异步模式多了一个可以输出同步时钟的信号线 USART_CK。适用于大批量数据需要传输的情况。

下图为 USART 同步发送时序图，分别给出极性和相位自由组合成的四种情况，图 1 USART_CTRL1 寄存器的 DBLCFG=0，对应 1 位起始位，8 位数据位，一位停止位的情况，图 2 USART_CTRL1 寄存器的 DBLCFG=1，对应 1 位起始位，9 位数据位，1 位停止位的情况。其中 USART_CK 的时钟极性由 USART_CTRL2 寄存器的 CPOL 位决定：CPOL 为 0 时，CK 引脚的空闲状态为低电平；CPOL 为 1 时，CK 引脚的空闲状态为高电平。USART_CK 的相位由 USART_CTRL2 寄存器的 CPHA 位决定：CPHA 为 0 时，表明在第 1 个时钟边沿进行采样；CPOL 为 1 时，表明在第 2 个时钟边沿进行采样。

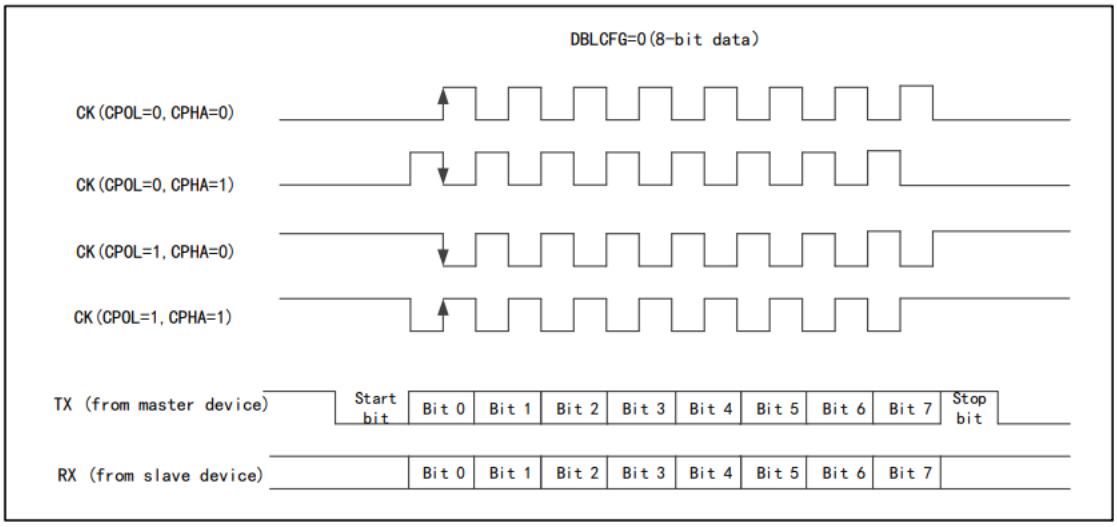

图 1 USART 同步发送时序图（DBLCFG=0）

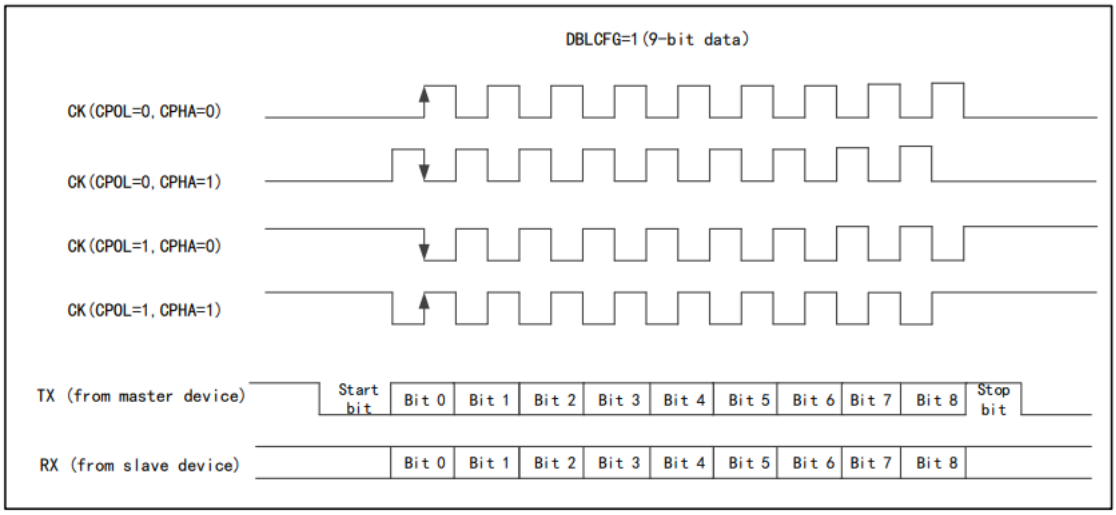

图 2　USART 同步发送时序图（DBLCFG=1）

异步模式是串行，异步，全双工的通信。通过收发双方约定相同波特率实现同步通信，数据以相同的帧格式进行发送。通常运用在短距离、速率不高的工业实际应用中。本文主要讨论异步串口通信情况。

2.2　UART 通信协议概述

异步通信需要收发双方约定波特率，确定每位比特位的持续时间，以保证双方的时序同步。其中，波特率是指单位时间内传送的码元的符号个数。

如图 3，图 4 所示，串口的报文格式为：起始位（1bit）+ 数据位（5~8 bits）+ 奇偶校验位（0/1 bit）+停止位（0.5~1.5 bits）。

起始位：使用串口时，在发送有效，自动产生 1bit 的低电平起始位。

数据位：通信时有效数据的长度，通常为 5~8 位。在进行数据收发前，应先完成相应的配置。

奇偶校验位：增加校验位以检验传输过程中是否因干扰出现数据传输错误的情况。设置为奇校验，确保传输数据逻辑高位的个数为奇数；设置为偶校验，确保传输数据逻辑高位的个数为偶数。

停止位：有效数据发送完毕后，发送设定位数的高电平，表示一帧数据传输结束。

空闲位：在下一个起始位逻辑低位到来之前，数据线一直保持高位状态。空闲位不属于报文内容。

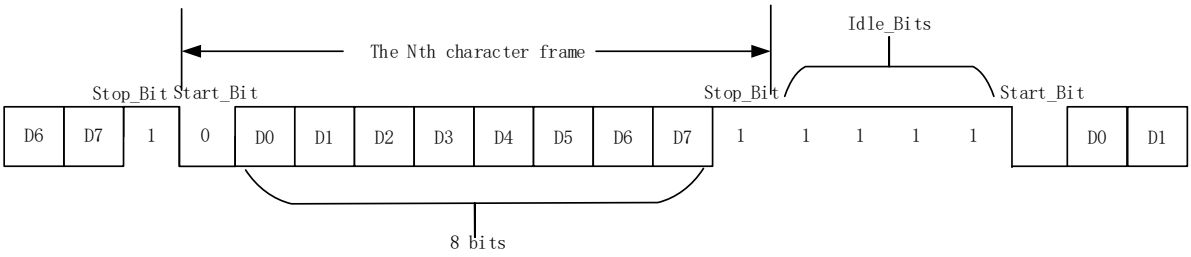

图 3 串口帧结构图（含空闲位）

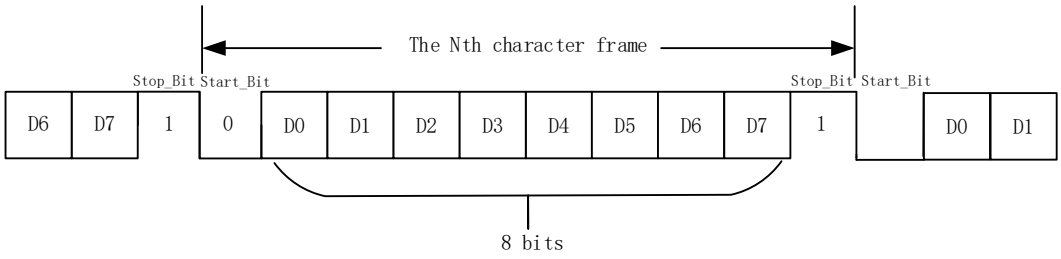

图 4 串口帧结构图（不含空闲位）

3 软件模拟串口

3.1 硬件设计

本次设计使用 APM32F407ZGT6 的 GPIO PB9,PB10 分别模拟串口的发送线 TX 和接收线 RX。

借助 USB 转 TTL 线，将 RX 连接开发板的 PB9，TX 连接 PB10，同时将两边地线相连。

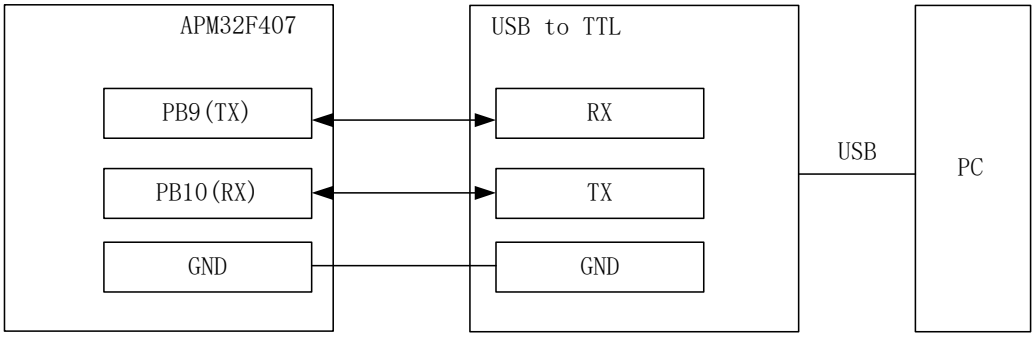

图 5 硬件连接图

3.2 软件设计

本次设计实现功能为接收串口助手发送的数据后，发送相同数据进行回显。设置波特率为 9600 bps，起始位 1 bit，数据位 8 bit，不设校验位，停止位 1 bit。

设计思路为：将功能实现分为接收和发送两个部分。

接收功能的实现，需要以精确的延时为前提；从而获得正确的数据。例程使用通用定时器定时 104us(t = 1/9600 s = 104us)进入中断，在中断服务函数中按位接收数据。

将接收到的数据存入缓存区，并将其发送到串口助手进行回显。

发送功能实现相对简单：将数据线置 0，模拟起始位；利用滴答定时器进行延时 104us；利用 for 循环按位发送并延时。完成 8 位数据位传输后，将数据线置 1，模拟停止位。

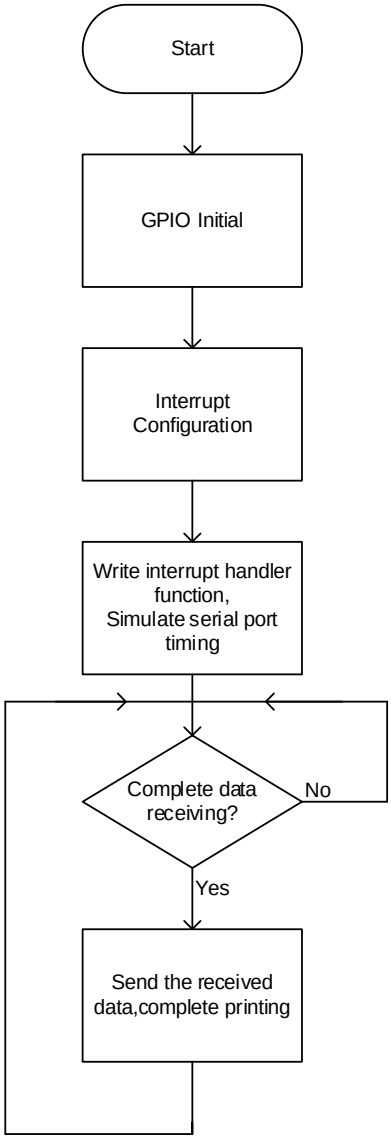

图 6 软件实现流程图

3.2.1 GPIO 配置

函数完成了 GPIO PB9,PB10 的初始化配置。

将发送引脚 TX PB9 配置为推挽输出模式，输出速度为 50MHz。串口在空闲时，数据位为高电平状态，因此，完成配置后，需将 PB9 置 1。

将接收引脚 RX PB10 配置为上拉输入模式，同时配置外部中断，数据开始传输前，一定会产生一个下降沿。因此将 PB10 中断触发方式设定为下降沿触发。

```
void Soft_Usart_Init(void)
{
    GPIO_Config_T    gpioConfig;
    EINT_Config_T    eintConfig;

    RCM_EnableAHB1PeriphClock(RCM_AHB1_PERIPH_GPIOB);
    RCM_EnableAPB2PeriphClock(RCM_APB2_PERIPH_SYSCFG);

    /* Tx GPIOB PIN9 */
    GPIO_ConfigStructInit(&gpioConfig);
    gpioConfig.mode = GPIO_MODE_OUT;
    gpioConfig.otype = GPIO_OTYPE_PP;
    gpioConfig.pin = GPIO_PIN_9;
    gpioConfig.speed = GPIO_SPEED_50MHz;
    GPIO_Config(GPIOB,&gpioConfig);
    GPIO_SetBit(GPIOB,GPIO_PIN_9);

    /* Rx GPIOB PIN10 */
    GPIO_ConfigStructInit(&gpioConfig);
    gpioConfig.mode = GPIO_MODE_IN;
    gpioConfig.pin = GPIO_PIN_10;
    gpioConfig.pupd = GPIO_PUPD_UP;
    gpioConfig.speed = GPIO_SPEED_50MHz;
    GPIO_Config(GPIOB,&gpioConfig);

    SYSCFG_ConfigEINTLine(SYSCFG_PORT_GPIOB,SYSCFG_PIN_10);

    eintConfig.line = EINT_LINE_10;
    eintConfig.mode = EINT_MODE_INTERRUPT;
    eintConfig.lineCmd = ENABLE;
    eintConfig.trigger = EINT_TRIGGER_FALLING;
    EINT_Config(&eintConfig);

    NVIC_EnableIRQRequest(EINT15_10_IRQn,2,3);
}
```

3.2.2 TMR 配置

例程使用 TMR4 通用定时器。根据波特率 9600 bps 可知，每一位数据的持续时间为 104.16us。配置一次计数时间为 1us，周期设定为 104 可达到所需定时效果。

TMR4 时钟线为 APB1，如图 7，图 8 和图 9 所示，根据用户手册和系统时钟初始化函数可知 APB1 为 AHB1 进行 4 分频后得到，最大频率为 42MHz。TMR 时钟频率为 42 * 2 = 84MHz。根据计算公式将预分频系数设置为 83，得到计数器驱动时钟为 84 /（83+1）= 1MHz，即计数一次时间为 1/1M s = 1 us。一次中断产生的时间间隔为自动重装载值 104 * 计数一次所需时间 1us =104us。

表格 1 计算公式说明及使用

相关公式	说明	计算
t = 1 / Baud	脉宽 = 1 / 波特率	t = 1 / 9600 s ≈ 104 us
CK_CNT = CK_INT/(Division + 1)	驱动计数器时钟=内部时钟/（预分频+1）	CK_CNK = 84 /（83+1）= 1MHz
t1 = 1 / CK_CNT	一次计数时间 = 1 / 驱动计数器时钟	t1 = 1/1M s = 1 us
t2 = t1 * period	产生中断的时间间隔 = 一次计数所需时间 * 周期	t2 = t = 1 us * 104 = 104us

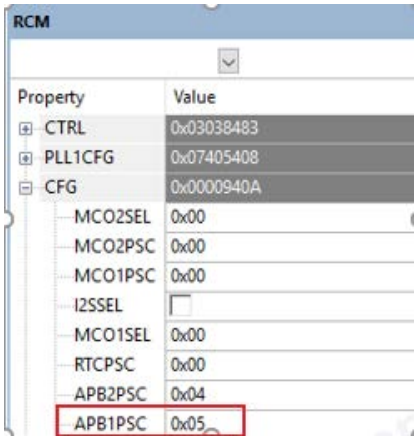

图 7 RCM 寄存器配置

12:10	APB1PSC	R/W	APB1 Clock Prescaler Factor Configure To control APB low-speed clock division factor. 0xx:HCLK not divided 100:HCLK divided by 2 101:HCLK divided by 4 110:HCLK divided by 8 111:HCLK divided by 16 Caution:PCLK1 not to exceed 45 MHz

图 8 APB1PSC 位域说明

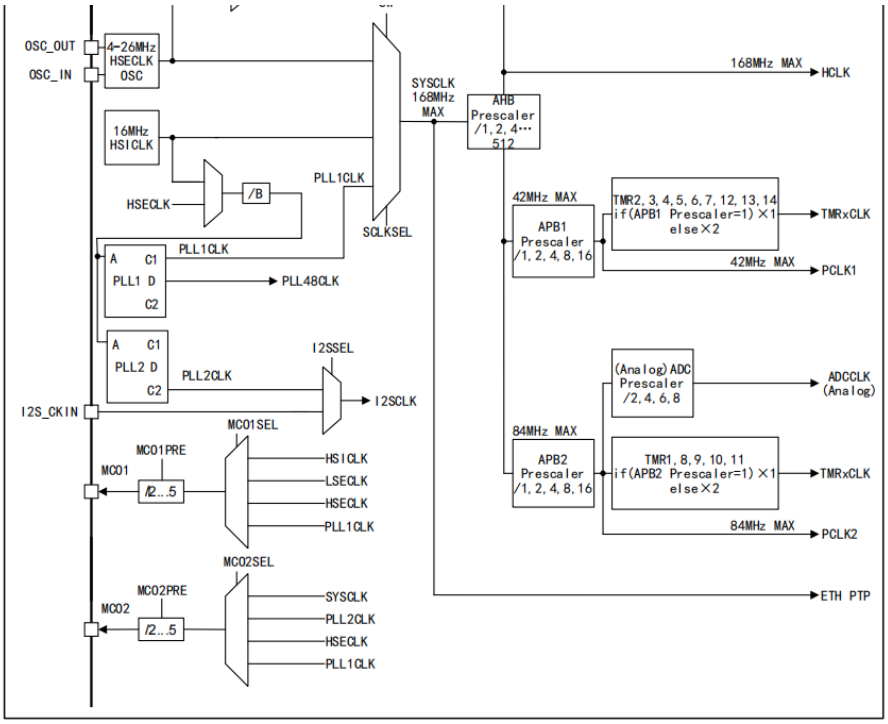

图 9 时钟树

```
void TMR4_Init(void)
{
    TMR_BaseConfig_T TMRBaseConfig;

    RCM_EnableAPB1PeriphClock(RCM_APB1_PERIPH_TMR4);

    TMRBaseConfig.clockDivision = TMR_CLOCK_DIV_1;
    TMRBaseConfig.countMode = TMR_COUNTER_MODE_UP;
    TMRBaseConfig.division = 84 - 1;
    TMRBaseConfig.period = 104;
    TMR_ConfigTimeBase(TMR4,&TMRBaseConfig);
    TMR_ClearStatusFlag(TMR4,TMR_FLAG_UPDATE);
    TMR_Enable(TMR4);

    /* Configure NVIC_IRQRequest */
    TMR_EnableInterrupt(TMR4,TMR_INT_UPDATE);
    NVIC_EnableIRQRequest(TMR4_IRQn,2,1);

}
```

3.2.3 GPIO 模拟串口发送

根据图 3 可知，空闲时数据线处于高电平状态，由高电平跳变为低电平表示通信开始，因此在有效数据传输前将 PB9 置 0；发送起始位后，根据设置的波特率，延时 104us，利用 for 循环开始

发送数据位。需要注意的是，串口由最低位开始发送。数据位发送结束，将引脚置 1，模拟停止位。

```
void Soft_Usart_TXData(u8 ch)
{
    uint8_t i = 0;
    GPIO_ResetBit(GPIOB,GPIO_PIN_9);
    Delay_us(BaudRate_9600);
    for(i = 0; i < 8; i++)
    {
        if(ch & 0x01)
        {
            GPIO_SetBit(GPIOB,GPIO_PIN_9);
        }
        else
        {
            GPIO_ResetBit(GPIOB,GPIO_PIN_9);
        }
        Delay_us(BaudRate_9600);
        ch = ch >> 1;
    }
    GPIO_SetBit(GPIOB,GPIO_PIN_9);
    Delay_us(BaudRate_9600);
}

void Soft_Usart_Send(u8 *buf,u8 len)
{
    uint8_t t;
    for(t = 0;t < len;t++)
    {
        Soft_Usart_TXData(buf[t]);
        Delay_ms(1);
    }
}
```

3.2.4 GPIO 中断服务函数

当接收到低电平并且接收位超过或等于停止位时，此时重置接收状态为起始位，使能定时器，开始接收新一帧的数据。若接收状态未达到停止位，表示此时的下降沿由有效数据产生，并不表示新数据的到来，可直接跳出中断。

```
void EINT15_10_IRQHandler(void)
{
    if(EINT_ReadStatusFlag(EINT_LINE_10) != RESET)
    {
        if(Soft_Usart_RXData() == 0)
        {
          if(recvState >= COM_STOP_BIT)
          {
              recvState = COM_START_BIT;
              TMR_Enable(TMR4);
              flag = 1;
          }
        }
       EINT_ClearStatusFlag(EINT_LINE_10);
    }
}
```

3.2.5 TMR 中断服务函数

开启定时器后，每隔 104us 触发中断。进入中断服务函数，完成数据的按位接收。由于串口发送数据时，由最低位开始发送，在接收一位数据后需通过移位，将其放置在正确的位置，保证接收数据的准确性。当读取到停止位时，将接收值存入缓存区。

```
void TMR4_IRQHandler(void)
{
    if(TMR_ReadStatusFlag(TMR4,TMR_FLAG_UPDATE) != RESET)
    {
        TMR_ClearStatusFlag(TMR4,TMR_FLAG_UPDATE);
        recvState++;
        if(recvState >= COM_STOP_BIT)
        {
            TMR_Disable(TMR4);
            USART_buf[len++] = recvData;
            return;
        }
        if(Soft_Usart_RXData())
        {
          recvData |= (1 << (recvState - 1));
        }
        else
        {
          recvData &= ~(1 << (recvState - 1));
        }
    }
}
```

3.2.6 主函数

完成初始化后，持续发送接收到的数据。

```
int main(void)
{
    NVIC_ConfigPriorityGroup(NVIC_PRIORITY_GROUP_2);
    Delay_Init();
    Soft_Usart_Init();
    TMR4_Init();
    while (1)
    {
       if(len > 10)
        {
          len = 0;
          Soft_Usart_Send(USART_buf,11);
        }
    }

}
```

3.3 实验现象

打开串口调试助手，设置波特率，起始位，数据位和停止位后，发送字符，成功回显。

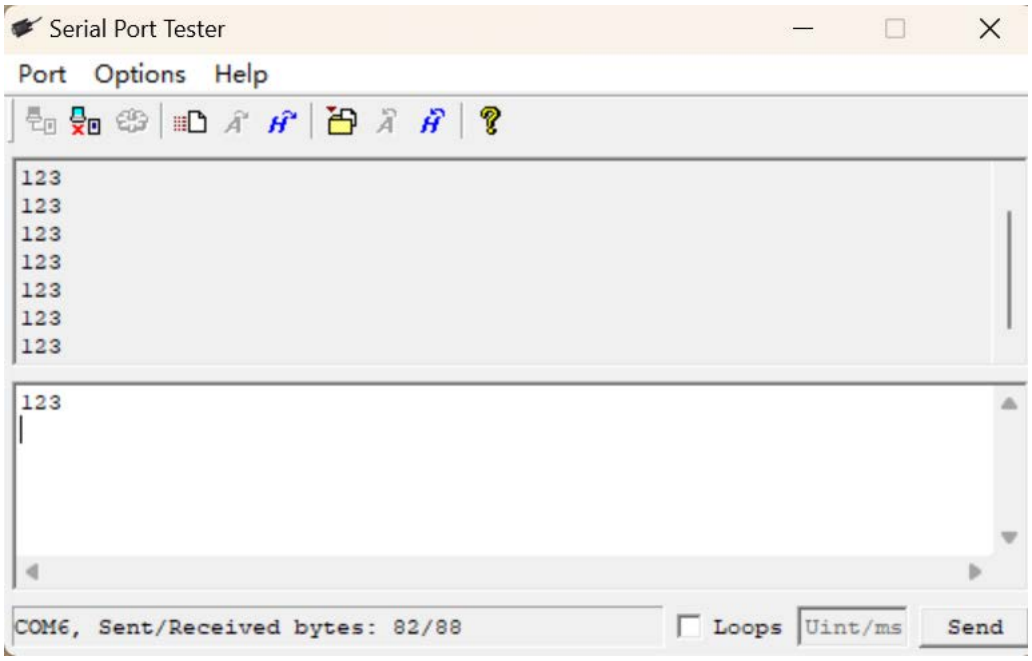

图 10 实验现象

4 版本历史

表格 2 文件版本历史

日期	版本	变更历史
2023.08.22	1.0	新建

声明

本手册由珠海极海半导体有限公司（以下简称“极海”）制订并发布，所列内容均受商标、著作权、软件著作权相关法律法规保护，极海保留随时更正、修改本手册的权利。使用极海产品前请仔细阅读本手册，一旦使用产品则表明您（以下称“用户”）已知悉并接受本手册的所有内容。用户必须按照相关法律法规和本手册的要求使用极海产品。

1、权利所有

本手册仅应当被用于与极海所提供的对应型号的芯片产品、软件产品搭配使用，未经极海许可，任何单位或个人均不得以任何理由或方式对本手册的全部或部分内容进行复制、抄录、修改、编辑或传播。

本手册中所列带有“®”或“TM”的“极海”或“Geehy”字样或图形均为极海的商标，其他在极海产品上显示的产品或服务名称均为其各自所有者的财产。

2、无知识产权许可

极海拥有本手册所涉及的全部权利、所有权及知识产权。

极海不应因销售、分发极海产品及本手册而被视为将任何知识产权的许可或权利明示或默示地授予用户。

如果本手册中涉及任何第三方的产品、服务或知识产权，不应被视为极海授权用户使用前述第三方产品、服务或知识产权，除非在极海销售订单或销售合同中另有约定。

3、版本更新

用户在下单购买极海产品时可获取相应产品的最新版的手册。

如果本手册中所述的内容与极海产品不一致的，应以极海销售订单或销售合同中的约定为准。

4、信息可靠性

本手册相关数据经极海实验室或合作的第三方测试机构批量测试获得，但本手册相关数据难免会出现校正笔误或因测试环境差异所导致的误差，因此用户应当理解，极海对本手册中可能出现的该等错误无需承担任何责任。本手册相关数据仅用于指导用户作为性能参数参照，不构成极海对任何产品性能方面的保证。

用户应根据自身需求选择合适的极海产品，并对极海产品的应用适用性进行有效验证和测试，以确认极海产品满足用户自身的需求、相应标准、安全或其它可靠性要求；若因用户未充分对极海产品进行有效验证和测试而致使用户损失的，极海不承担任何责任。

5、合规要求

用户在使用本手册及所搭配的极海产品时，应遵守当地所适用的所有法律法规。用户应了解产品可能受到产品供应商、极海、极海经销商及用户所在地等各国有关出口、再出口或其它法律的限制，用户（代表其本身、子公司及关联企业）应同意并保证遵守所有关于取得极海产品及／或技术与直接产品的出口和再出口适用法律与法规。

6、免责声明

本手册由极海“按原样”（as is）提供，在适用法律所允许的范围内，极海不提供任何形式的明示或暗示担保，包括但不限于对产品适销性和特定用途适用性的担保。

对于用户后续在针对极海产品进行设计、使用的过程中所引起的任何纠纷，极海概不承担责任。

7、责任限制

在任何情况下，除非适用法律要求或书面同意，否则极海和/或以“按原样”形式提供本手册的任何第三方均不承担损害赔偿责任，包括任何一般、特殊因使用或无法使用本手册相关信息而产生的直接、间接或附带损害（包括但不限于数据丢失或数据不准确，或用户或第三方遭受的损失）。

8、适用范围

本手册的信息用以取代本手册所有早期版本所提供的信息。

©2023 珠海极海半导体有限公司 - 保留所有权利