Application Note

ATK-MB024 485 模块使用说明

RS485 模块

使用说明

正点原子

广州市星翼电子科技有限公司

修订历史

版本	日期	原因
V1.0	2024/11/01	第一次发布

目 录

1，硬件连接

这里以正点原子 M48Z-M3 最小系统板 STM32F103 版为例，给大家介绍一下模块和板卡的连接方法。其它板卡与模块的硬件连接方法，请大家在“**ATK-MB024 485 模块\3，程序源码\相应板卡例程文件夹\readme.txt**”路径下查看。

ATK-MB024 485 模块可通过杜邦线与正点原子 M48Z-M3 最小系统板 STM32F103 版进行连接，具体的连接关系，如下表所示：

模块对应开发板	连接关系			
ATK-MB024 485 模块	VCC	GND	TX	RX
M48Z-M3 最小系统板 STM32F103 版	3.3V/5V	GND	PA3	PA2

表 1.1 485 模块与 M48Z-M3 最小系统板 STM32F103 版连接关系

值得注意的是，要实现 RS485 通信，则需两套相关的设备，例如：两块 STM32F103 最小系统板+两个 ATK-MB024 485 模块。通信的时候，两个 485 模块的连接关系如下表所示：

485 模块	连接关系		
模块 1	A	B	GND
模块 2	A	B	GND

表 1.2 两个 485 模块之间的连接关系

2，实验功能

2.1 ATK-MB024 485 模块测试实验

2.1.1 功能说明

在本实验中，串口会打印 485 模块发送或接收到的数据。需要查看这部分实验信息的用户，可用杜邦线将最小系统板 STM32F103 的 PA9 引脚和 GND 连接至外部的 USB 转串口设备，这样就可以通过 XCOM 上位机查看串口打印的信息了。

当 KEY0 按键被按下时，485 模块会往外发送一次数据，数据量为 200 个字节，数据内容随机。当 485 模块接收到数据时，会自动将数据打印到串口。

用户可以通过拨动开关选择是否开启 120Ω终端电阻，ON 档即开启，OFF 档即关闭。

开发板的 LED0 闪烁，提示程序运行。

2.1.2 源码解读

打开本实验的工程文件夹，能够在./Drivers/BSP 目录下看到 ATK_RS485 文件夹，其包含了 ATK-MB024 485 模块的驱动文件，如下图所示：

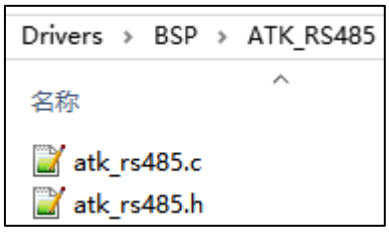

图 2.1.2.1 ATK-MB024 485 模块驱动代码

2.1.2.1 ATK-MB024 485 模块驱动

下面简要介绍 atk_rs485.c 中几个重要的 API 函数。

1. 函数 atk_rs485_init()

该函数用于初始化 ATK-MB024 485 模块，具体的代码，如下所示：

```
/**
 * @brief       ATK_RS485 模块初始化函数
 *   @note      该函数主要是初始化串口
 * @param       baudrate: 波特率，根据自己需要设置波特率值
 * @retval      无
 */
void atk_rs485_init(uint32_t baudrate)
{
    GPIO_InitTypeDef gpio_init_struct;

    ATK_RS485_TX_GPIO_CLK_ENABLE();             /* 使能 串口 TX 脚 时钟 */
    ATK_RS485_RX_GPIO_CLK_ENABLE();             /* 使能 串口 RX 脚 时钟 */
    ATK_RS485_UX_CLK_ENABLE();                  /* 使能 串口 时钟 */

    /* 串口 TX 引脚初始化 */
```

```
    gpio_init_struct.Pin = ATK_RS485_TX_GPIO_PIN;
    gpio_init_struct.Mode = GPIO_MODE_AF_PP;
    gpio_init_struct.Pull = GPIO_PULLUP;
    gpio_init_struct.Speed = GPIO_SPEED_FREQ_HIGH;
    HAL_GPIO_Init(ATK_RS485_TX_GPIO_PORT, &gpio_init_struct);

    /* 串口 RX 引脚初始化 */
    gpio_init_struct.Pin = ATK_RS485_RX_GPIO_PIN;
    gpio_init_struct.Mode = GPIO_MODE_AF_INPUT;
    HAL_GPIO_Init(ATK_RS485_RX_GPIO_PORT, &gpio_init_struct);

    /* USART 初始化设置 */
    g_rs458_handler.Instance = ATK_RS485_UX;    /* 选择 485 对应的串口 */
    g_rs458_handler.Init.BaudRate = baudrate;   /* 波特率 */
    /* 字长为 8 位数据格式 */
    g_rs458_handler.Init.WordLength = UART_WORDLENGTH_8B;
    g_rs458_handler.Init.StopBits = UART_STOPBITS_1;        /* 一个停止位 */
    g_rs458_handler.Init.Parity = UART_PARITY_NONE;         /* 无奇偶校验位 */
    g_rs458_handler.Init.HwFlowCtl = UART_HWCONTROL_NONE;   /* 无硬件流控 */
    g_rs458_handler.Init.Mode = UART_MODE_TX_RX;            /* 收发模式 */
    HAL_UART_Init(&g_rs458_handler);        /* HAL_UART_Init()会使能 UART2 */

#if RS485_EN_RX                             /* 如果使能了接收 */

    __HAL_UART_ENABLE_IT(&g_rs458_handler, UART_IT_RXNE);  /* 开启接收中断 */
    HAL_NVIC_EnableIRQ(ATK_RS485_UX_IRQn);          /* 使能 USART2 中断 */
    HAL_NVIC_SetPriority(ATK_RS485_UX_IRQn, 3, 3); /* 抢占优先级 3，子优先级 3 */

#endif
}
```

atk_rs485_init()函数并不复杂，它就是对 485 模块相关串口及引脚进行初始化配置，这里使用的串口为串口 2，TX 引脚为 PA2，RX 引脚为 PA3，波特率由函数的入口参数决定。

2. 函数 atk_rs485_send_data()

该函数用于 485 模块发送数据，具体代码，如下所示：

```
/**
 * @brief       RS485 发送 len 个字节数据
 * @param       buf     : 缓冲区指针
 * @param       len     : 发送的字节数(为了和本代码的接收匹配,这里建议不要超过
RS485_REC_LEN 个字节)
 * @retval      无
 */
void atk_rs485_send_data(uint8_t *buf, uint8_t len)
{
    HAL_UART_Transmit(&g_rs458_handler, buf, len, 1000);    /* 串口发送数据 */
```

```
    g_RS485_rx_cnt = 0;                                         /* 清空计数 */
}
```

该函数通过调用 HAL_UART_Transmit()函数来发送串口数据，具体的数据内容和长度由函数入口参数决定。

3. 函数 atk_rs485_receive_data()

该函数用于 485 模块接收数据，具体代码，如下所示：

```
/**
 * @brief       RS485 查询接收到的数据
 * @param       buf : 接收缓冲区首地址
 * @param       len : 接收到的数据长度
 *   @arg             0,表示没有接收到任何数据
 *   @arg             其他，表示接收到的数据长度
 * @retval      无
 */
void atk_rs485_receive_data(uint8_t *buf, uint8_t *len)
{
    uint8_t rxlen = g_RS485_rx_cnt;
    uint8_t i = 0;
    *len = 0;                                   /* 默认为 0 */

    delay_ms(5);     /* 等待 5ms,连续超过 5ms 没有接收到一个数据,则认为接收结束 */

    if (rxlen == g_RS485_rx_cnt && rxlen)   /* 接收到了数据,且接收完成了 */
    {
        for (i = 0; i < rxlen; i++)
        {
            buf[i] = g_RS485_rx_buf[i];         /* 取出数据，保存到 buf 中 */
        }

        *len = g_RS485_rx_cnt;                  /* 记录本次数据长度 */
        g_RS485_rx_cnt = 0;                     /* 清零 */
    }
}
```

进入到上述函数中，先读取并记录 g_RS485_rx_cnt 的值，然后延时 5ms，接着再次读取 g_RS485_rx_cnt 的值，如果前后两次的值相同，则说明没有新的数据进来了。与此同时，rxlen 不为 0，则可以开始读出数据了。

读出的数据会被保存到接收缓冲区中，该缓冲区的地址由函数入口参数决定。

2.1.2.3 实验测试代码

实验的测试代码在 demo.c 文件中，该文件在工程根目录下的 User 文件夹。测试代码的入口函数为 demo_run()，具体的代码，如下所示：

```
/**
 * @brief       例程演示入口函数
 * @param       无
 * @retval      无
```

```
 */
void demo_run(void)
{
    uint8_t key;

    uint8_t i = 0, t = 0;
    uint8_t cnt = 0;
    uint8_t rs485buf[200];

    atk_rs485_init(256000);                 /* 初始化 RS485，波特率 256000 */

    while (1)
    {
        key = key_scan(0);

        if (key == KEY0_PRES)               /* KEY0 按下,发送一次数据 */
        {
            printf("RS485 模块 发送的数据为：");

            for (i = 0; i < 200; i++)
            {
                rs485buf[i] = cnt + i;      /* 填充发送缓冲区 */
                printf("%02X", rs485buf[i]); /* 打印输出数据 */
            }

            printf("\r\n");

            atk_rs485_send_data(rs485buf, 200); /* 发送 200 个字节 */
        }

        atk_rs485_receive_data(rs485buf, &key);

        if (key)                            /* 接收到数据 */
        {
            if (key > 200)key = 200;        /* 最大有效字节数：200 个 */

            printf("RS485 模块 接收的数据为：");

            for (i = 0; i < key; i++)
            {
                printf("%02X", rs485buf[i]); /* 打印输出数据 */
            }

            printf("\r\n");
```

```
        }

        t++;
        delay_ms(10);

        if (t == 20)
        {
            LED0_TOGGLE();                    /* LED0 闪烁，提示系统正在运行 */
            t = 0;
            cnt++;
        }
    }
}
```

从上面代码可以看出，整个测试代码的逻辑相对简单。首先初始化 485 模块相关的串口，波特率为 256000，接着在 while 循环中不断地扫描按键状态以及获取接收的数据。

当 KEY0 按键被按下后，485 模块将会发送 200 个字节的随机数据，与此同时，串口会打印这些已发送的数据；当 485 模块接收到数据后，也会通过串口进行打印。LED0 闪烁表示程序正常运行。

2.1.3 实验现象

将 ATK-MB024 485 模块按照第一节“硬件连接”中介绍的连接方式与开发板连接，并将实验代码编译烧录至开发板中。**值得注意的是，要实现 RS485 通信，则需两套相关的设备，例如：两块 STM32F103 最小系统板+两个 ATK-MB024 485 模块。两块开发板都烧录本实验代码，两个 485 模块的接线端子 A 接 A，B 接 B，GND 接 GND。**

本实验使用串口输出调试信息，因此需将开发板的 PA9 连接至 DAP 虚拟串口（或 USB 转 TTL 模块）的 RX 引脚。完成连接后，可通过串口调试助手 XCOM 查看实验信息输出，如下图所示：

图 2.1.3.1 串口调试助手显示内容

注：其它现象请看 2.1.1 功能说明。

3，其他

1、 购买地址：

天猫：https://zhengdianyuanzi.tmall.com

淘宝：https://openedv.taobao.com

2、 资料下载

模块资料下载地址：http://www.openedv.com/docs/index.html

3、 技术支持

公司网址：www.alientek.com

技术论坛：http://www.openedv.com/forum.php

在线教学：www.yuanzige.com

B 站视频：https://space.bilibili.com/394620890

传真：020-36773971

电话：020-38271790

正点原子