Application Note

ATK-MW1278D 模块使用说明

高性能无线串口模块

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	2022/06/25	第一次发布
V1.1	2023/03/07	新增阿波罗 F429 与 F767 硬件连接描述

目 录

1，硬件连接

1.1 正点原子 MiniSTM32F103 开发板

ATK-MW1278D 模块可直接与正点原子 MiniSTM32F103 开发板板载的 ATK 模块接口（ATK MODULE）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-MW1278D 模块	VCC	GND	TXD	RXD	AUX	MD0
MiniSTM32F103 开发板	3.3V/5V	GND	PD2	PC12	PC4	PA4

表 1.1.1 ATK-MW1278D 模块与 MiniSTM32F103 开发板连接关系

1.2 正点原子精英 STM32F103 开发板

ATK-MW1278D 模块可直接与正点原子精英 STM32F103 开发板板载的 ATK 模块接口（ATK MODULE）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-MW1278D 模块	VCC	GND	TXD	RXD	AUX	MD0
精英 STM32F103 开发板	3.3V/5V	GND	PB11	PB10	PA4	PA15

表 1.2.1 ATK-MW1278D 模块与精英 STM32F103 开发板连接关系

1.3 正点原子战舰 STM32F103 开发板

ATK-MW1278D 模块可直接与正点原子战舰 STM32F103 开发板板载的 ATK 模块接口（ATK MODULE）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-MW1278D 模块	VCC	GND	TXD	RXD	AUX	MD0
战舰 STM32F103 开发板	3.3V/5V	GND	PB11	PB10	PA4	PA15

表 1.3.1 ATK-MW1278D 模块与战舰 STM32F103 开发板连接关系

注意，若要使用正点原子战舰 STM32F103 开发板的 ATK MODULE 接口连接 ATK-MW1278D 模块，需要用跳线帽将开发板板载的 P8 接线端子的 PB10(TX)和 GBC_RX 以及 PB11(RX)和 GBC_TX 用跳线帽进行短接，如下图所示：

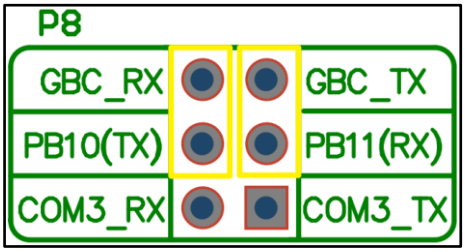

图 1.3.1 战舰 STM32F103 开发板 P8 接线端子

1.4 正点原子探索者 STM32F407 开发板

ATK-MW1278D 模块可直接与正点原子探索者 STM32F407 开发板板载的 ATK 模块接口（ATK MODULE）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-MW1278D 模块	VCC	GND	TXD	RXD	AUX	MD0
探索者 STM32F407 开发板	3.3V/5V	GND	PB11	PB10	PF6	PC0

表 1.4.1 ATK-MW1278D 模块与探索者 STM32F407 开发板连接关系

注意，若要使用正点原子探索者 STM32F407 开发板的 ATK MODULE 接口连接 ATK-MW1278D 模块，需要用跳线帽将开发板板载的 P2 接线端子的 PB10(TX)和 GBC_RX 以及 PB11(RX)和 GBC_TX 用跳线帽进行短接，如下图所示：

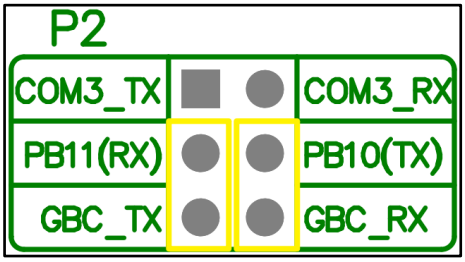

图 1.4.1 探索者 STM32F407 开发板 P2 接线端子

1.5 正点原子 F407 电机控制开发板

ATK-MW1278D 模块可直接与正点原子 F407 电机控制开发板板载的 ATK 模块接口（ATK MODULE）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-MW1278D 模块	VCC	GND	TXD	RXD	AUX	MD0
F407 电机控制开发板	3.3V/5V	GND	PC11	PC10	PI10	PI11

表 1.5.1 ATK-MW1278D 模块与 F407 电机控制开发板连接关系

1.6 正点原子 MiniSTM32H750 开发板

ATK-MW1278D 模块可直接与正点原子 MiniSTM32H750 开发板板载的 ATK 模块接口（ATK MODULE）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-MW1278D 模块	VCC	GND	TXD	RXD	AUX	MD0
MiniSTM32H750 开发板	3.3V/5V	GND	PA3	PA2	PC2	PC3

表 1.6.1 ATK-MW1278D 模块与 MiniSTM32H750 开发板连接关系

1.7 正点原子阿波罗 STM32F429 开发板

ATK-MW1278D 模块可直接与正点原子阿波罗 STM32F429 开发板板载的 ATK 模块接口（ATK MODULE）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-MW1278D 模块	VCC	GND	TXD	RXD	AUX	MD0
阿波罗 STM32F429 开发板	3.3V/5V	GND	PB11	PB10	PI11	PA4

表 1.7.1 ATK-MW1278D 模块与阿波罗 STM32F429 开发板连接关系

注意，若要使用正点原子阿波罗 STM32F429 开发板的 ATK MODULE 接口连接 ATK-MW1278D 模块，需要用跳线帽将开发板板载的 P9 接线端子的 PB10(TX)和 GBC_RX 以及 PB11(RX)和 GBC_TX 用跳线帽进行短接，如下图所示：

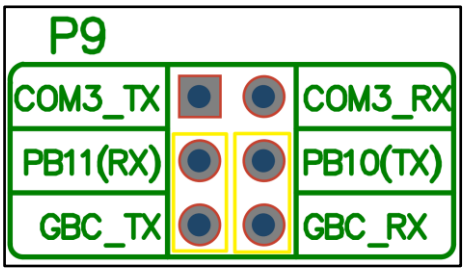

图 1.7.1 阿波罗 STM32F429 开发板 P9 接线端子

1.8 正点原子阿波罗 STM32F767 开发板

ATK-MW1278D 模块可直接与正点原子阿波罗 STM32F767 开发板板载的 ATK 模块接口（ATK MODULE）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-MW1278D 模块	VCC	GND	TXD	RXD	AUX	MD0
探索者 STM32F407 开发板	3.3V/5V	GND	PB11	PB10	PI11	PA4

表 1.8.1 ATK-MW1278D 模块与阿波罗 STM32F767 开发板连接关系

注意，若要使用正点原子阿波罗 STM32F767 开发板的 ATK MODULE 接口连接 ATK-MW1278D 模块，需要用跳线帽将开发板板载的 P9 接线端子的 PB10(TX)和 GBC_RX 以及 PB11(RX)和 GBC_TX 用跳线帽进行短接，如下图所示：

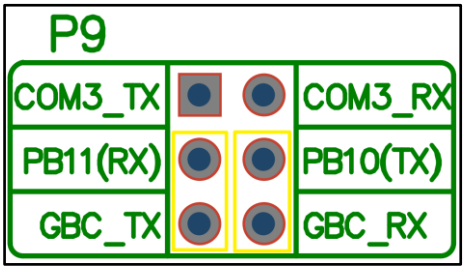

图 1.8.1 阿波罗 STM32F767 开发板 P9 接线端子

2，实验功能

2.1 ATK-MW1278D 模块测试实验

2.1.1 功能说明

在本实验中，开发板主控芯片通过串口与 ATK-MW1278D 模块进行通讯，并在上电后自动配置 ATK-MW1278D 模块，并能够与另一个 ATK-MW1278D 模块（连接至上位机）进行连接和数据传输。

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_MW1278D 子文件夹，该文件夹中就包含了 ATK-MW1278D 模块的驱动文件，如下图所示：

```
./Drivers/BSP/ATK_MW1278D/
|-- atk_mw1278d.c
|-- atk_mw1278d.h
|-- atk_mw1278d_uart.c
`-- atk_mw1278d_uart.h
```

图 2.1.2.1 ATK-MW1278D 模块驱动代码

2.1.2.1 ATK-MW1278D 模块接口驱动

在图 2.1.2.1 中，atk_mw1278d_uart.c 和 atk_mw1278d_uart.h 是开发板与 ATK-MW1278D 模块通讯而使用的 UART 驱动文件，关于 UART 的驱动介绍，请查看正点原子各个开发板对应的开发指南中 UART 对应的章节。

值得一提的是，由于 ATK-MW1278D 模块通过 UART 发送给主控芯片的数据的长度是不固定的，因此主控芯片就无法直接通过接收到数据的长度来判断 ATK-MW1278D 模块传来的一帧数据是否完成。对于这种通过 UART 接收不定长数据的情况，可以通过 UART 总线是否空闲来判断一帧的传输是否完成，恰巧 STM32 的 UART 提供了总线空闲中断功能，因此可以开启 UART 的总线空闲中断，并在中断中做相应的处理，具体的实现过程可以查看 ATK-MW1278D 模块的模块接口驱动代码，这里不做过多的描述。

2.1.2.2 ATK-MW1278D 模块驱动

在图 2.1.2.1 中，atk_mw1278d.c 和 atk_mw1278d.h 是 ATK-MW1278D 模块的驱动文件，包含了 ATK-MW1278D 模块初始化、发送 AT 指令的相关 API 函数和部分 AT 指令的封装函数。函数比较多，下面仅介绍几个重要的 API 函数。

1. 函数 atk_mw1278d_init()

该函数用于初始化 ATK-MW1278D 模块，具体的代码，如下所示：

```
/**
 * @brief   ATK-MW1278D 模块初始化
 * @param   baudrate: ATK-MW1278D 模块 UART 通讯波特率
 * @retval  ATK_MW1278D_EOK     : ATK-MW1278D 模块初始化成功，函数执行成功
 *          ATK_MW1278D_ERROR   : ATK-MW1278D 模块初始化失败，函数执行失败
 */
uint8_t atk_mw1278d_init(uint32_t baudrate)
```

```
{
    uint8_t ret;

    atk_mw1278d_hw_init();                  /* 硬件初始化 */
    atk_mw1278d_uart_init(baudrate);        /* UART 初始化 */
    atk_mw1278d_enter_config();             /* 进入配置模式 */
    ret = atk_mw1278d_at_test();            /* AT 指令测试 */
    atk_mw1278d_exit_config();              /* 退出配置模式 */
    if (ret != ATK_MW1278D_EOK)
    {
        return ATK_MW1278D_ERROR;
    }

    return ATK_MW1278D_EOK;
}
```

从上面的代码中可以看出，函数 atk_mw1278d_init()会初始化与 ATK-MW1278D 模块通讯的 UART 接口，然后通过 AT 测试指令测试与 ATK-MW1278D 模块的通讯是否正常，若通讯正常，那么接下来就能够通过 UART 配置 ATK-MW1278D 模块，并与其进行通讯了。

2. 函数 atk_mw1278d_send_at_cmd()

该函数主要实现主控芯片与 ATK-MW1278D 模块的 AT 指令传输，本驱动代码中的大部分驱动函数都是基于该函数实现的。函数 atk_mw1278d_send_at_cmd()的具体代码，如下所示：

```
/**
 * @brief   向 ATK-MW1278D 模块发送 AT 指令
 * @param   cmd     : 待发送的 AT 指令
 *          ack     : 等待的响应
 *          timeout : 等待超时时间
 * @retval  ATK_MW1278D_EOK         : 函数执行成功
 *          ATK_MW1278D_ETIMEOUT    : 等待期望应答超时，函数执行失败
 */
uint8_t atk_mw1278d_send_at_cmd(char *cmd, char *ack, uint32_t timeout)
{
    uint8_t *ret = NULL;

    atk_mw1278d_uart_rx_restart();
    atk_mw1278d_uart_printf("%s\r\n", cmd);

    if ((ack == NULL) || (timeout == 0))
    {
        return ATK_MW1278D_EOK;
    }
    else
    {
        while (timeout > 0)
```

```
        {
            ret = atk_mw1278d_uart_rx_get_frame();
            if (ret != NULL)
            {
                if (strstr((const char *)ret, ack) != NULL)
                {
                    return ATK_MW1278D_EOK;
                }
                else
                {
                    atk_mw1278d_uart_rx_restart();
                }
            }
            timeout--;
            delay_ms(1);
        }

        return ATK_MW1278D_ETIMEOUT;
    }
}
```

从上面的代码中可以看出，函数 atk_mw1278d_send_at_cmd()函数会将待发送的 AT 指令加上换行符后通过 UART 发送至 ATK-MW1278D 模块，随后等待 ATK-MW1278D 模块的响应，并判断响应中是否包含期望等待的响应，如果有，则说明本次 AT 指令传输成功。

2.1.2.3 实验测试代码

实验的测试代码为文件 demo.c，在工程目录下的 User 子目录中。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 ret;
    uint8_t key;
    uint8_t *buf;

    /* 初始化 ATK-MW1278D 模块 */
    ret = atk_mw1278d_init(115200);
    if (ret != 0)
    {
        printf("ATK-MW1278D init failed!\r\n");
        while (1)
        {
```

```
            LED0_TOGGLE();
            delay_ms(200);
        }
    }

    /* 配置 ATK-MW1278D 模块 */
    atk_mw1278d_enter_config();
    ret  = atk_mw1278d_addr_config(DEMO_ADDR);
    ret += atk_mw1278d_wlrate_channel_config(DEMO_WLRATE, DEMO_CHANNEL);
    ret += atk_mw1278d_tpower_config(DEMO_TPOWER);
    ret += atk_mw1278d_workmode_config(DEMO_WORKMODE);
    ret += atk_mw1278d_tmode_config(DEMO_TMODE);
    ret += atk_mw1278d_wltime_config(DEMO_WLTIME);
    ret += atk_mw1278d_uart_config(DEMO_UARTRATE, DEMO_UARTPARI);
    atk_mw1278d_exit_config();
    if (ret != 0)
    {
        printf("ATK-MW1278D config failed!\r\n");
        while (1)
        {
            LED0_TOGGLE();
            delay_ms(200);
        }
    }

    printf("ATK-MW1278D config succedded!\r\n");
    atk_mw1278d_uart_rx_restart();

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

        if (key == KEY0_PRES)
        {
            if (atk_mw1278d_free() != ATK_MW1278D_EBUSY)
            {
                atk_mw1278d_uart_printf("This is from ATK-MW1278D.\r\n");
            }
        }

        buf = atk_mw1278d_uart_rx_get_frame();
        if (buf != NULL)
        {
            printf("%s", buf);
```

```
            atk_mw1278d_uart_rx_restart();
        }

        delay_ms(10);
    }
}
```

从上面的代码中可以看出，整个实验代码的逻辑还是比较简单的，一开始先通过函数 atk_mw1278d_init() 初始化与 ATK-MW1278D 模块的 UART 接口，接着就是对 ATK-MW1278D 模块进行一系列的配置，最后就能够与另一个相同配置的 ATK-MW1278D 模块进行通讯了。

2.1.3 实验现象

将 ATK-MW1278D 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，如果此时开发板连接 LCD，那么 LCD 显示的内容，如下图所示：

图 2.1.3.1 LCD 显示内容一

同时，通过串口调试助手输出实验信息，如下图所示：

图 2.1.3.2 串口调试助手显示内容一

接下来程序会自动初始化与 ATK-MW1278D 模块的 UART 接口，并配置 ATK-MW1278D 模块，配置成功后，如下图所示：

图 2.1.3.3 ATK-MW1278D 模块配置成功

接下来 ATK-MW1278D 模块进能够与另一个相同设备地址、网络地址、空中速率、信道配置的 ATK-MW1278D 模块进行通讯了。

这里通过上位机软件配置另一个 ATK-MW1278D 模块，然后进行实验演示。

按下按键 0，可以发送数据至另一个 ATK-MW1278D 模块，如下图所示：

图 2.1.3.4 发送数据至另一个模块

同时也能实时通过串口将接收自另一个 ATK-MW1278D 模块的数据显示至串口调试助手，如下图所示：

图 2.1.3.5 接收到另一个模块的数据

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/modules/iot/atk-lora-01.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