



南方科技大学
SOUTHERN UNIVERSITY OF SCIENCE AND TECHNOLOGY

Embedded System and Microcomputer Principle

LAB11 Resistive touch screen

2022 Fall
wangq9@mail.sustech.edu.cn



CONTENTS

- 1 Principle Description
- 2 Control principle Description
- 3 How to Program
- 4 Practice



01

Principle Description



1. Principle Description

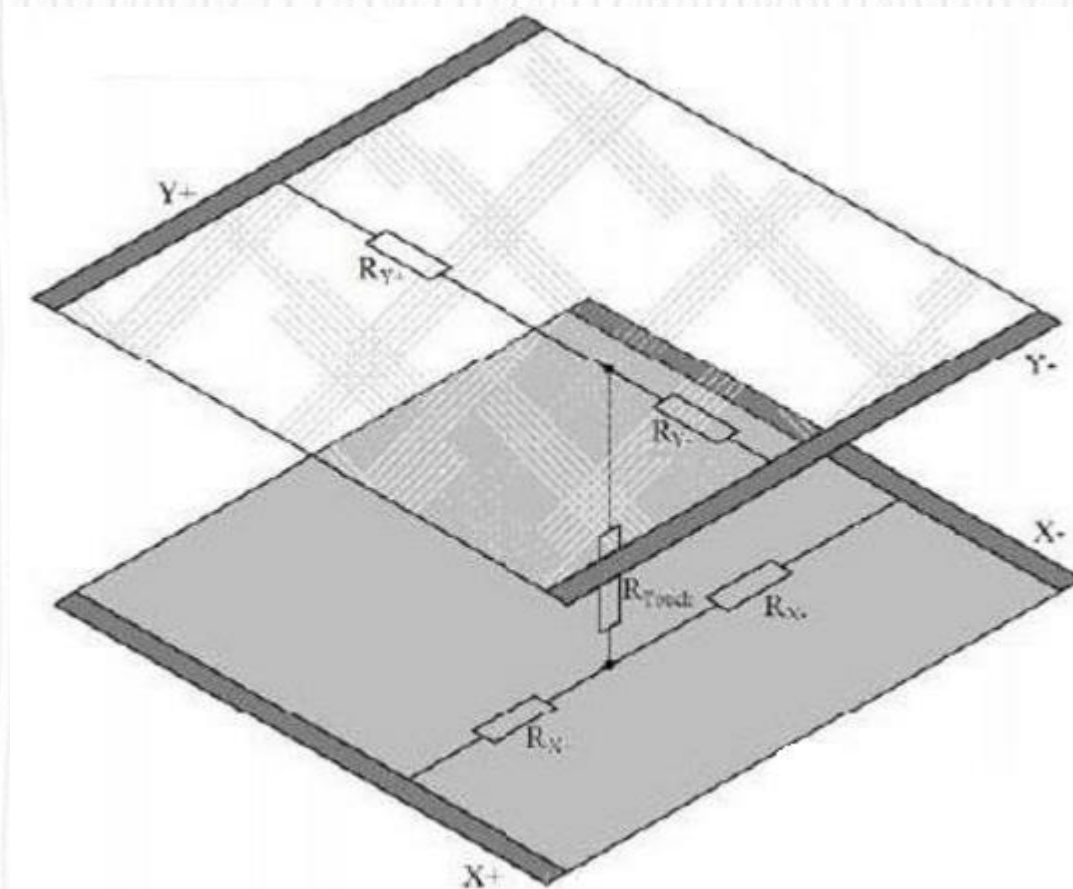
-- Classify of touch screen

- 按照触摸屏的工作原理和传输信息的介质分类
 - 电阻式：定位准确，单点触摸（Mini开发板使用）
 - 电容式：多点触控
 - 红外线式：价格低廉，曲面情况下失真
 - 表面声波式：解决各种缺点，但是屏幕表面如果有水滴和尘土会使触摸屏变的迟钝

1. Principle Description

-- Working principle of touch screen

- 电阻屏分为上下两层（为X层和Y层），中间由隔离支点隔开。
- 上下两层相邻面均涂有ITO涂层，为导电涂层。
- 在X层的两侧和Y层的两侧分别连接电路。
- X层上X-到X+，Y层上Y-到Y+的电阻是均匀分布的。

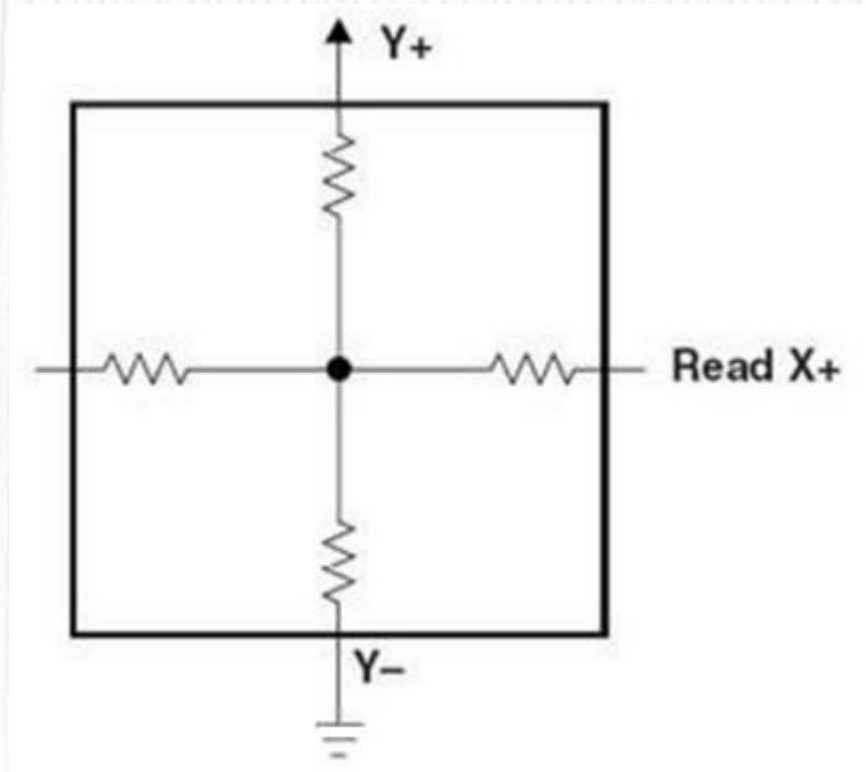


Schematic diagram

1. Principle Description

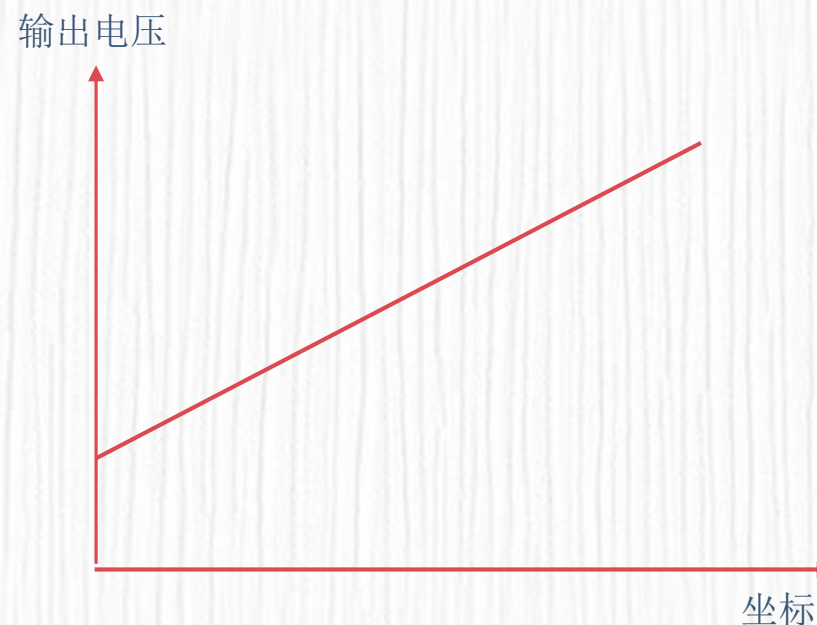
-- Measuring touch points

- 计算Y坐标时，在Y+电极施加驱动电压V，Y-接地，芯片通过X+测量接触点的电压。



等效示意图

- 由于ITO层均匀导电，触点电压与V电压之比等于触点Y坐标与屏高度之比。因此实际测量时，可以求得电压与距离的关系方程。





02

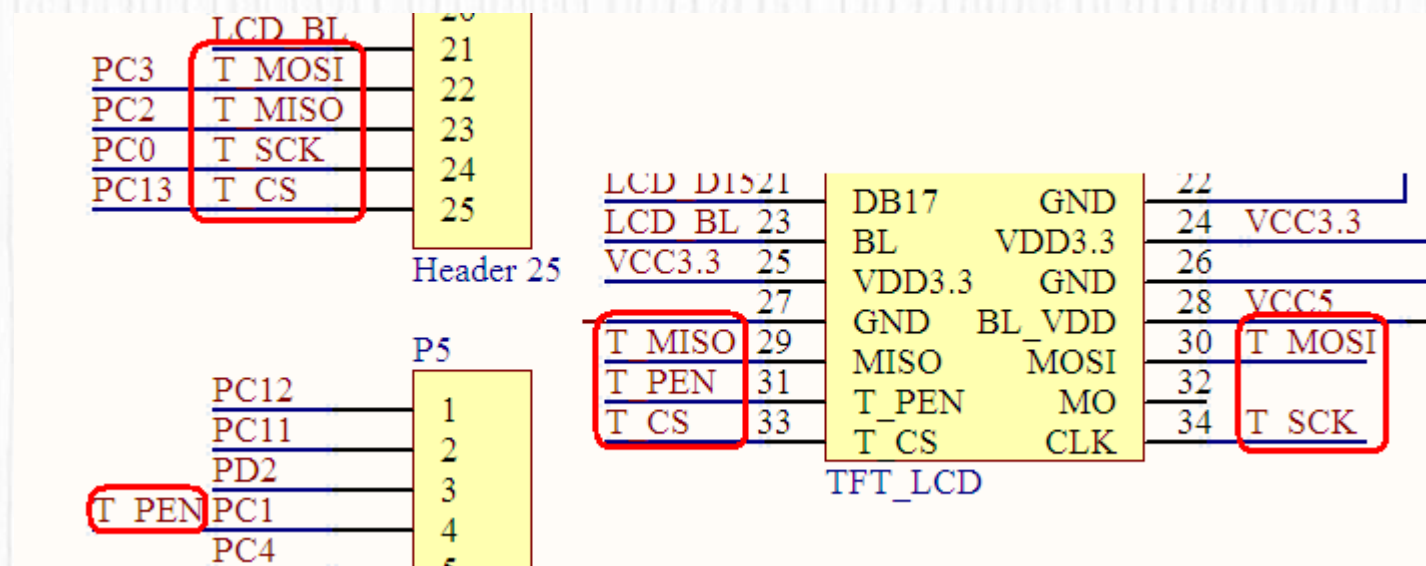
Control Principle Description



2. Control Principle Description

-- STM32 touch screen

- ALIENTEK TFTLCD 模块选择的是四线电阻式触摸屏。
- ALIENTEK TFTLCD 模块自带的触摸屏控制芯片为 XPT2046。
- T_CS: 片选
- T_CLK: 时钟信号
- T_MISO: 触屏数据读入
- T_MOSI: 数据输出
- T_PEN: 中断输出端



Circuit connection diagram between touch screen and STM32



2. Control Principle Description

-- XPT2046 chip

- 4导线制触摸屏控制器。
- 内含12位分辨率125KHz转换速率逐步逼近型A/D转换器。
- 支持从 1.5V 到 5.25V 的低电压 I/O 接口。
- 能通过执行两次 A/D 转换查出被按的屏幕位置。
- 还可以测量加在触摸屏上的压力。
- 内部自带 2.5V 参考电压可以作为辅助输入、温度测量和电池监测模式之用，电池监测的电压范围可以从 0V 到 6V。
- 片内集成有一个温度传感器。



2. Control Principle Description

-- SPI protocol

- Serial Peripheral interface , 串行外围设备接口。
- 一种高速的, 全双工, 同步的通信总线。
- 在芯片的管脚上占用四根线。
- SPI 物理接口含义:
 - SCLK: 时钟信号, 由主设备产生, 主从设备根据时钟传输数据。
 - CS: 从设备片选信号, 由主设备控制。多个从设备, 主设备选择哪一个从设备通信, 从设备相对应的片选拉低。
 - MISO: 主设备数据输入, 从设备数据输出。
 - MOSI: 主设备数据输出, 从设备数据输入。
- SPI 接口主要应用在 EEPROM, FLASH, 实时时钟, AD 转换器, 还有数字信号处理器和数字信号解码器之间。



03

How to Program



3. How to program

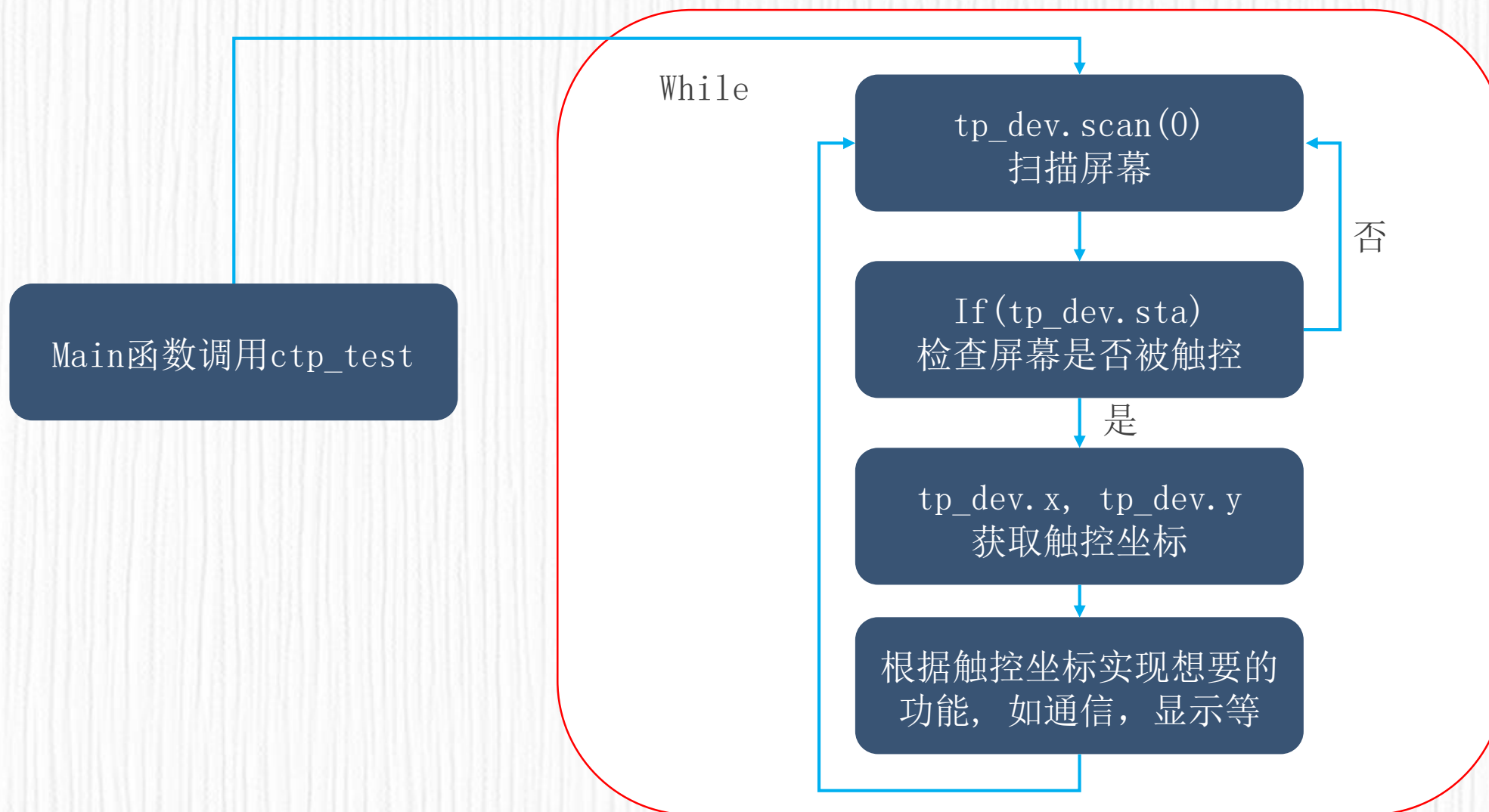
-- Touch screen control structure

- 电阻屏的驱动代码实现均在touch.h和touch.c中
- 在touch.h中定义了结构体_m_tp_dev

```
//触摸屏控制器
typedef struct
{
    u8 (*init)(void);           //初始化触摸屏控制器
    u8 (*scan)(u8);             //扫描触摸屏. 0, 屏幕扫描; 1, 物理坐标;
    void (*adjust)(void);       //触摸屏校准
    u16 x[CT_MAX_TOUCH];        //当前坐标
    u16 y[CT_MAX_TOUCH];        //电容屏有最多5组坐标, 电阻屏则用x[0], y[0]代表: 此次扫描时, 触屏的坐标, 用
                                //x[4], y[4]存储第一次按下时的坐标.
    u8 sta;                     //笔的状态
                                //b7: 按下1/松开0;
                                //b6: 0, 没有按键按下; 1, 有按键按下.
                                //b5: 保留
                                //b4~b0: 电容触摸屏按下的点数 (0, 表示未按下, 1表示按下)
    //////////////////////////////////// 触摸屏校准参数 (电容屏不需要校准) ////////////////////////////////////
    float xfac;
    float yfac;
    short xoff;
    short yoff;
    //新增的参数, 当触摸屏的左右上下完全颠倒时需要用到.
    //b0: 0, 竖屏 (适合左右为X坐标, 上下为Y坐标的TP)
    //    1, 横屏 (适合左右为Y坐标, 上下为X坐标的TP) |
    //b1~6: 保留
    //b7: 0, 电阻屏
    //    1, 电容屏
    u8 touchtype;
} _m_tp_dev;
```




3. How to program -- Working process





3. How to program

-- Some functions

- TP_Scan
 - 触摸屏按键扫描
 - 参数:tp (0:屏幕坐标; 1:物理坐标)
 - 返回值: 0:触屏无触摸; 1:触屏有触摸
- TP_Adjust
 - 触摸屏校准代码
 - 得到四个校准参数
 - 校准参数保存在AT24CXX芯片中
- TP_Init
 - 触摸屏初始化
 - 返回值: 0:触屏无校准; 1:触屏有校准



3. How to program

-- Main function

```
int main(void)
{
    HAL_Init();
    /* USER CODE BEGIN Init */
    Stm32_Clock_Init(RCC_PLL_MUL9);           //设置时钟, 72M
    delay_init(72);                           //初始化延时函数
    // uart_init(115200);                      //初始化串口
    // usmart_dev_init(84);                   //初始化USMART
    LED_Init();                               //初始化LED
    KEY_Init();                               //初始化按键
    LCD_Init();                               //初始化LCD
    tp_dev.init();                            //触摸屏初始化

    SystemClock_Config();
    MX_GPIO_Init();
    MX_SPI1_Init();
    MX_USART1_UART_Init();
    /* USER CODE BEGIN 2 */
    POINT_COLOR=RED;
    LCD_ShowString(30, 50, 200, 16, 16, "Mini STM32");
    LCD_ShowString(30, 70, 200, 16, 16, "TOUCH TEST");
    LCD_ShowString(30, 90, 200, 16, 16, "ATOM@ALIENTEK");
    LCD_ShowString(30, 110, 200, 16, 16, "2019/11/15");
    if(tp_dev.touchtype!=0XFF)
    {
        LCD_ShowString(30, 130, 200, 16, 16, "Press KEY0 to Adjust");//电阻屏才显示
    }
    delay_ms(1500);
    Load_Drow_Dialog();

    if(tp_dev.touchtype&0X80) ctp_test();//电容屏
    else rtp_test();                  //电阻屏
    /* USER CODE END 2 */
}
```



3. How to program -- rtp_test function

```
void rtp_test(void) { //电阻触摸屏测试函数
    u8 key;
    u8 i=0;
    while(1){
        key=KEY_Scan(0);
        tp_dev.scan(0);
        screen_norm_print();
        if(tp_dev.sta&TP_PRES_DOWN) { //触摸屏被按下
            if(tp_dev.x[0]<lcddev.width&&tp_dev.y[0]<lcddev.height){
                if(tp_dev.x[0]>(lcddev.width-24)&&tp_dev.y[0]<16){
                    screen_print();//清除
                }
                else if(tp_dev.x[0]<80&&tp_dev.y[0]<24){
                    LCD_ShowImage2(40,80);
                }
                else if(tp_dev.x[0]>60&&tp_dev.y[0]>60&&tp_dev.x[0]<180&&tp_dev.y[0]<100){
                    sprintf(DATA_TO_SEND, "SEND DATA");
                    HAL_UART_Transmit(&huart1, (uint8_t*)DATA_TO_SEND, strlen(DATA_TO_SEND), HAL_MAX_DELAY);
                }
                else if(tp_dev.x[0]>0&&tp_dev.y[0]>lcddev.height-24&&tp_dev.x[0]<80&&tp_dev.y[0]<lcddev.height){
                    change_state();
                }
                else {
                    TP_Draw_Big_Point(tp_dev.x[0], tp_dev.y[0], RED); //画图
                }
            }
        }
        else delay_ms(10); //没有按键按下的时候
        if(key==KEY0_PRES) { //KEY0按下, 则执行校准程序
            LCD_Clear(WHITE); //清屏
            TP_Adjust(); //屏幕校准
            TP_Save_Adjdata();
            Load_Drow_Dialog();
        }
        i++;
        if(i%20==0) LED0=!LED0;
    }
}
```


3. How to program

-- References

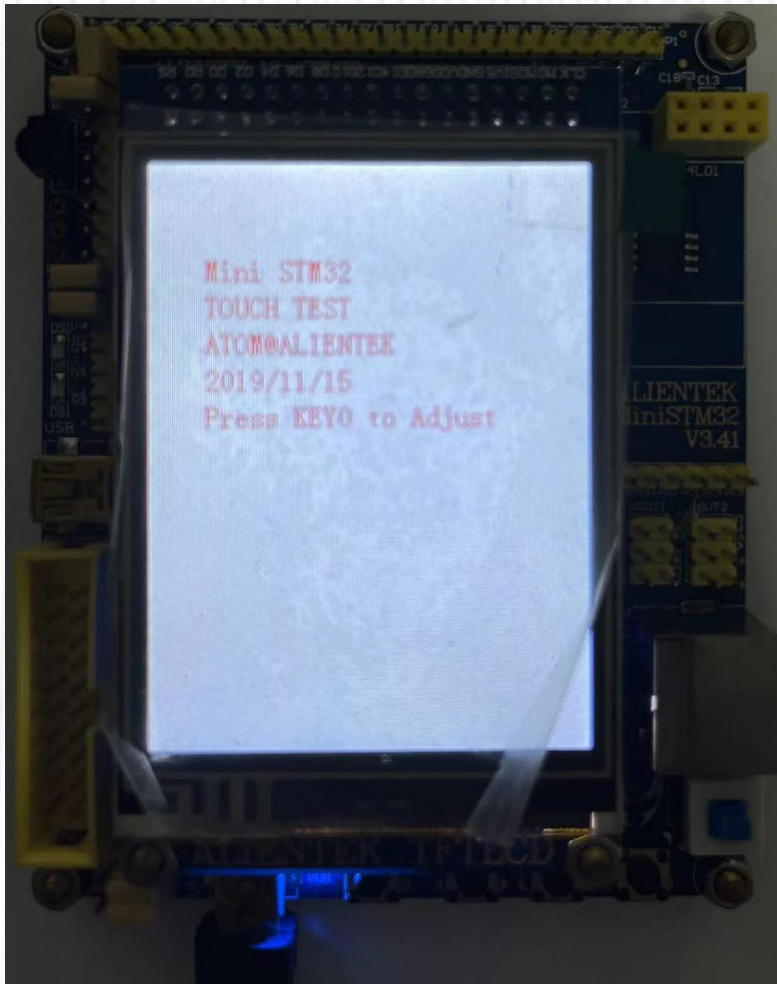
- Video:

https://www.bilibili.com/video/BV1Lx411Z7Qa?p=59&vd_source=2075c4a1a1877adf0c176fd94d5cf0f1

- Information download:

http://www.openedv.com/docs/boards/stm32/zdyz_stm32f103_mini.html

3. How to program -- Results 1



Adjust interface



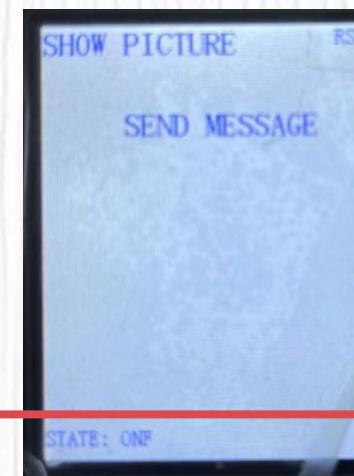
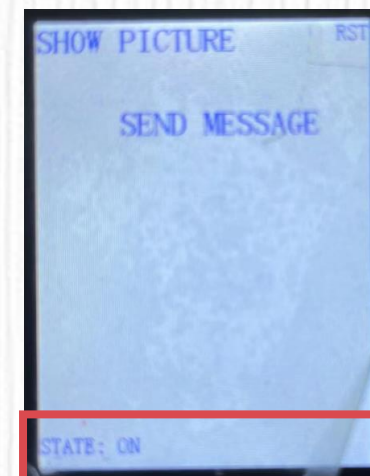
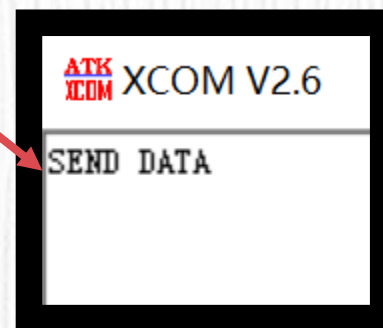
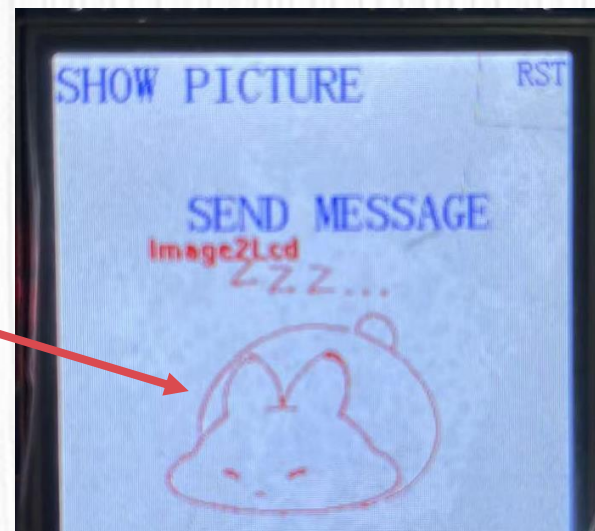
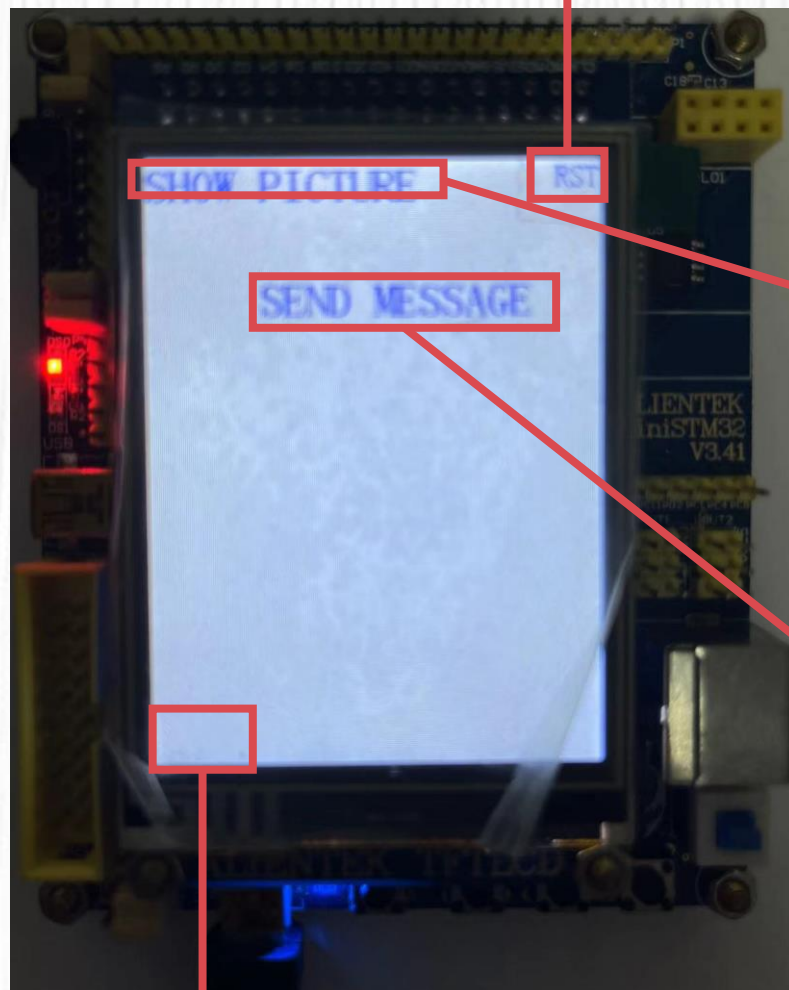
Main interface



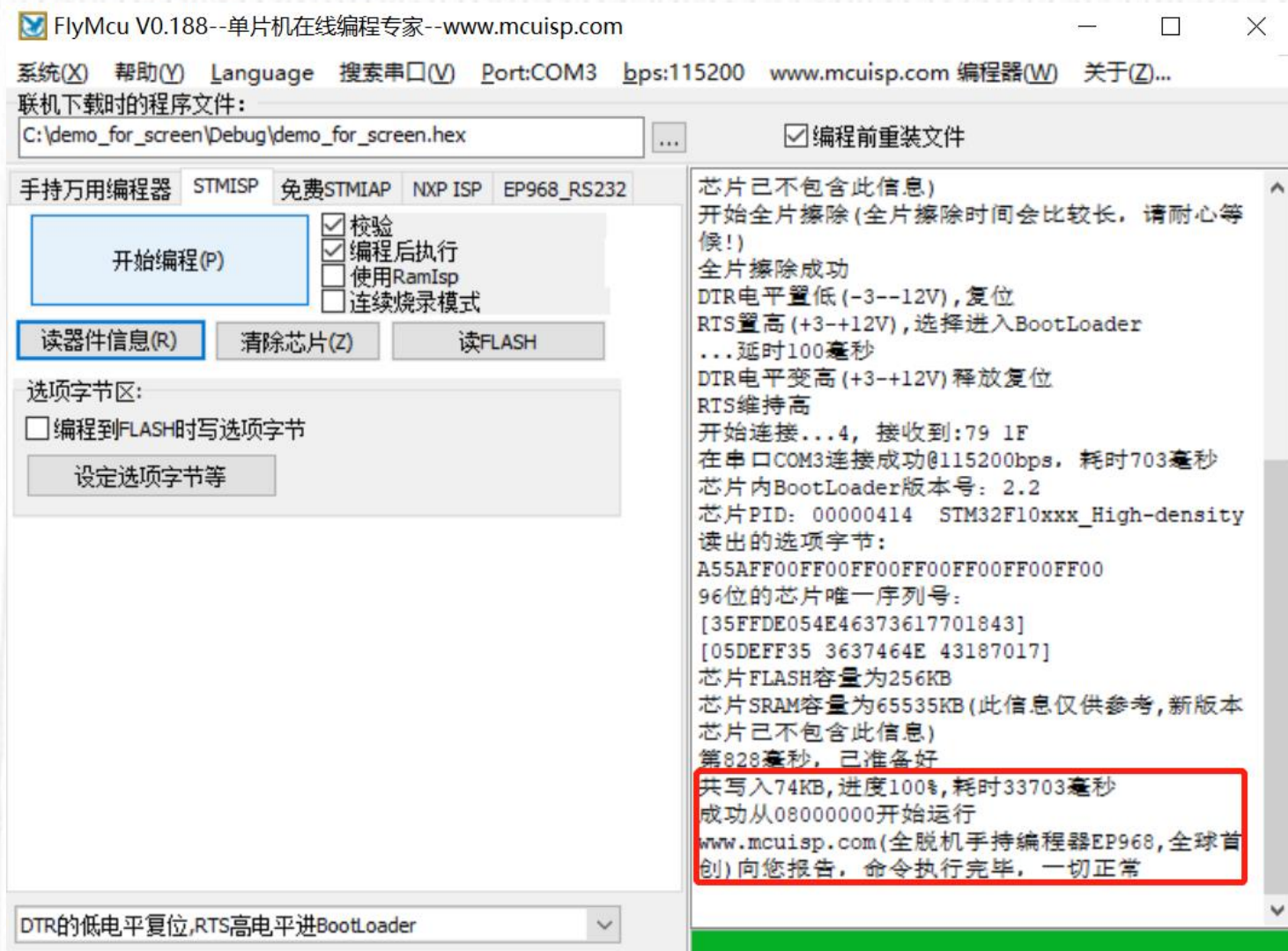
Writing on screen



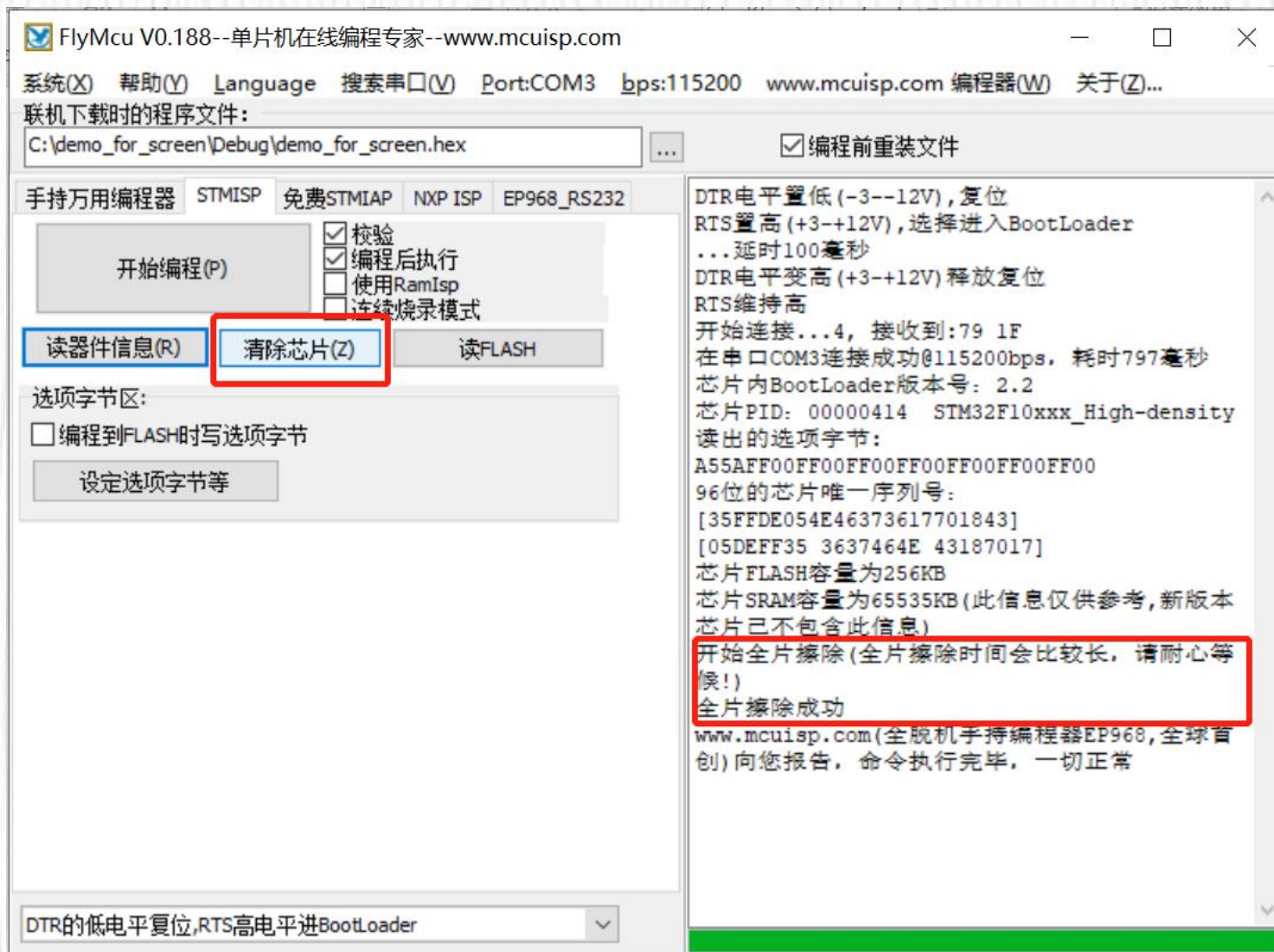
3. How to program -- Results 2



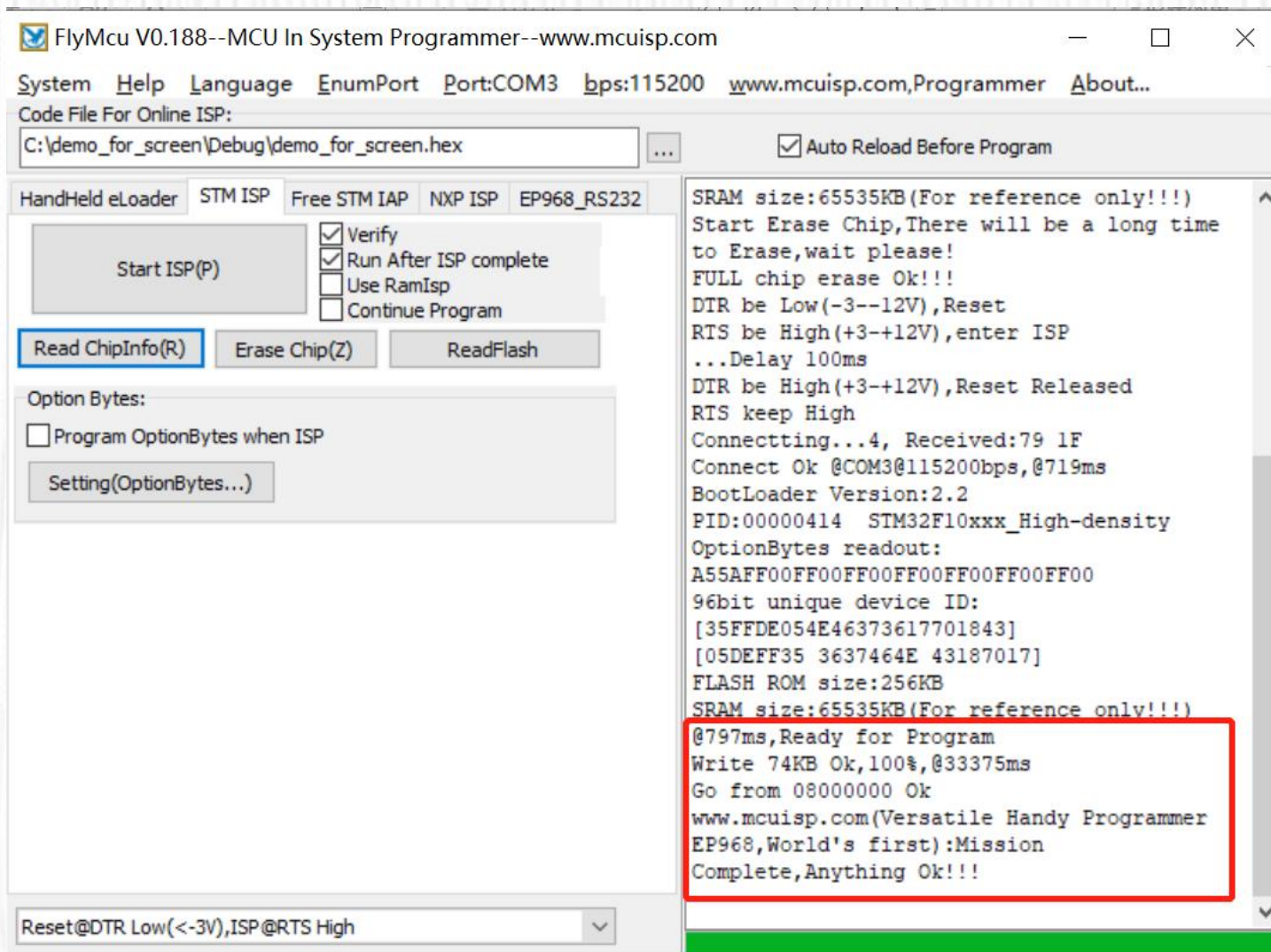
3. How to program -- Programming complete



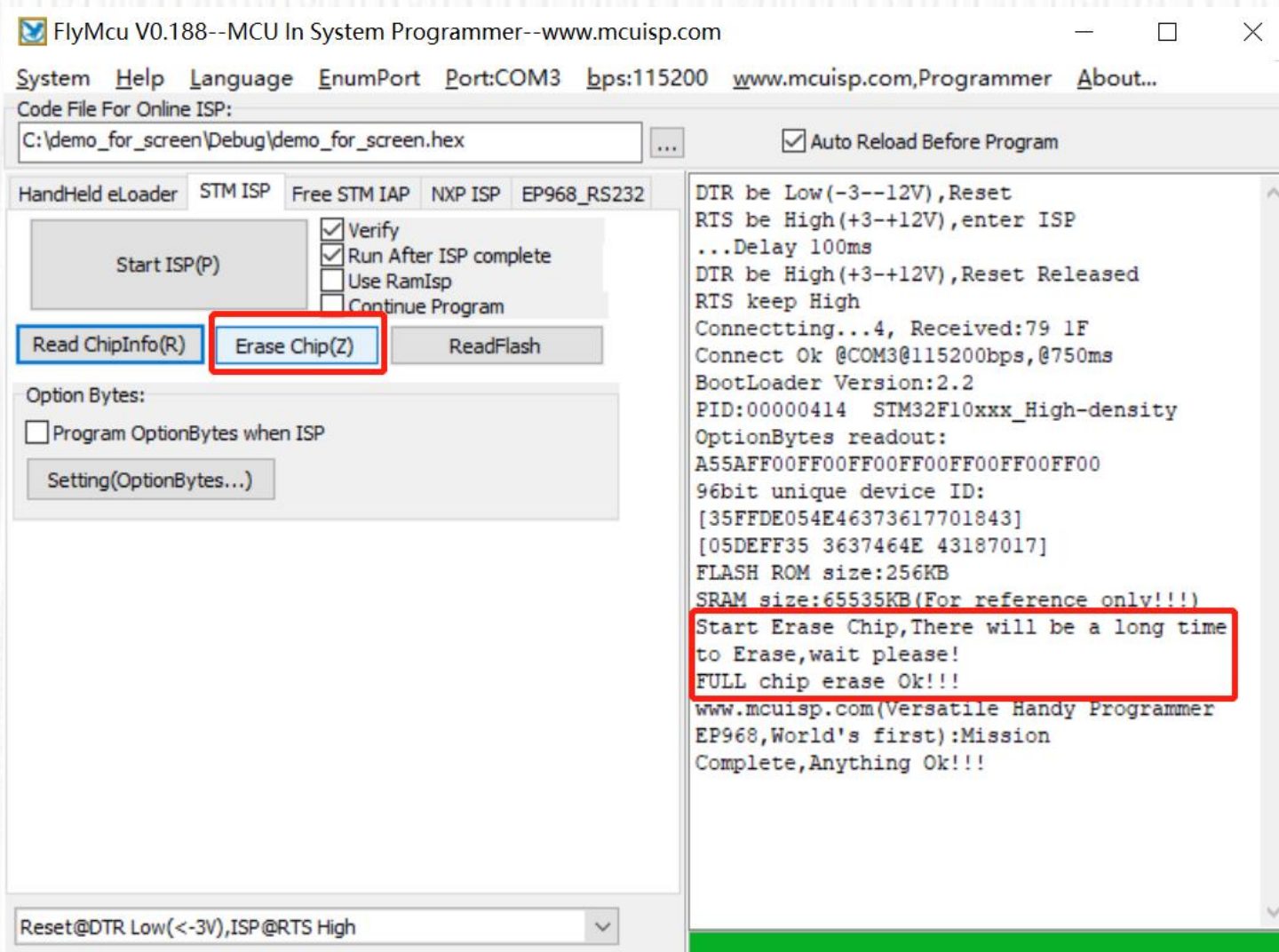
3. How to program -- Chip erase



3. How to program -- Programming complete



3. How to program -- Chip erase





04

Practice

4. Practice



- Run the demo on MiniSTM32 board.
- Complete a new buttons to realize an independently designed function.