E32R28T-1&E32N28T-1 2.8 寸 MicroPython 示例程序说明

目 录

1. 软件和硬件平台说明

模块：2.8寸ESP32-32E显示模块，拥有240x320分辨率，采用ST7789P3屏驱IC。

模块主控：ESP32-32E模组，最高主频240MHz，支持2.4G WIFI+蓝牙。

Thonny 版本：4.1.6。

ESP32 MicroPython固件版本：1.24.1。

2. 引脚分配说明

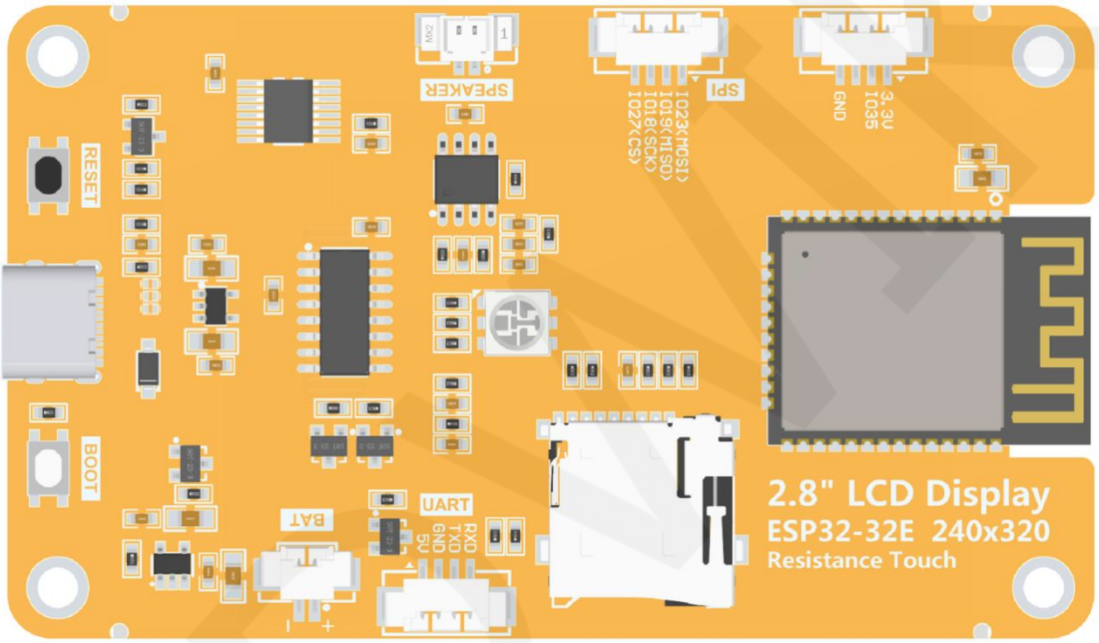

图2.1 2.8寸ESP32-32E显示模块背面图

2.8寸ESP32显示模块主控为ESP32-32E，其连接板载外设的GPIO分配如下表格所示：

ESP32-32E引脚分配说明			
板载设备	**板载设备引脚**	**ESP32-32E连接引脚**	**说明**
LCD	**TFT_CS**	IO15	液晶屏片选控制信号，低电平有效
	TFT_RS	IO2	液晶屏命令/数据选择控制信号 高电平：数据，低电平：命令
	TFT_SCK	IO14	液晶屏SPI总线时钟信号
	TFT_MOSI	IO13	液晶屏SPI总线写数据信号
	TFT_MISO	IO12	液晶屏SPI总线读数据信号

	TFT_RST	EN	液晶屏复位控制信号，低电平复位（和ESP32-32E主控共用复位引脚）	
	TFT_BL	IO21	液晶屏背光控制信号（高电平点亮背光，低电平关闭背光）	
RTP	TP_SCK	IO25	电阻触摸屏SPI总线时钟信号	
	TP_DIN	IO32	电阻触摸屏SPI总线写数据信号	
	TP_DOUT	IO39	电阻触摸屏SPI总线读数据信号	
	TP_CS	IO33	电阻触摸屏片选控制信号，低电平有效	
	TP_IRQ	IO36	电阻触摸屏触摸中断信号，产生触摸时，输入低电平到主控	
LED	LED_RED	IO22	红色LED灯	RGB三色LED灯，共阳极，低电平点亮，高电平关闭。
	LED_GREEN	IO16	绿色LED灯	
	LED_BLUE	IO17	蓝色LED灯	
SDCARD	SD_CS	IO5	SD卡片选信号，低电平有效	
	SD_MOSI	IO23	SD卡SPI总线写数据信号	
	SD_SCK	IO18	SD卡SPI总线时钟信号	
	SD_MISO	IO19	SD卡SPI总线读数据信号	
BATTERY	BAT_ADC	IO34	电池电压ADC值获取信号（输入）	
Audio	Audio_ENABLE	IO4	音频使能信号，低电平使能，高电平禁止	
	Audio_DAC	IO26	音频信号DAC输出信号	
KEY	BOOT_KEY	IO0	下载模式选择按键（按住该按键上电，然后松开就会进入下载模式）	
	RESET_KEY	EN	ESP32-23E复位按键，低电平复位（和液晶屏复位共用）	
Serial Port	RX0	RXD0	ESP32-32E串口接收信号	
	TX0	TXD0	ESP32-32E串口发送信号	
POWER	TYPE-C_POWER	/	Type-C电源接口，接入5V电压。	

表2.1 ESP32-32E板载外设引脚分配说明

3. 示例程序说明

3.1. 搭建ESP32 MicroPython开发环境

ESP32 MicroPython开发环境搭建详细说明见资料包里“**ESP32_MicroPython开发环境搭建**”说明文档。

3.2. 上传文件

开发环境搭建好之后，需将相关的文件上传到ESP32设备，才能运行测试程序。

上传文件之前，先了解一下MicroPython示例程序的目录内容。打开资料包里“**1-示例程序_Demo\MicroPython**”目录，如下图所示：

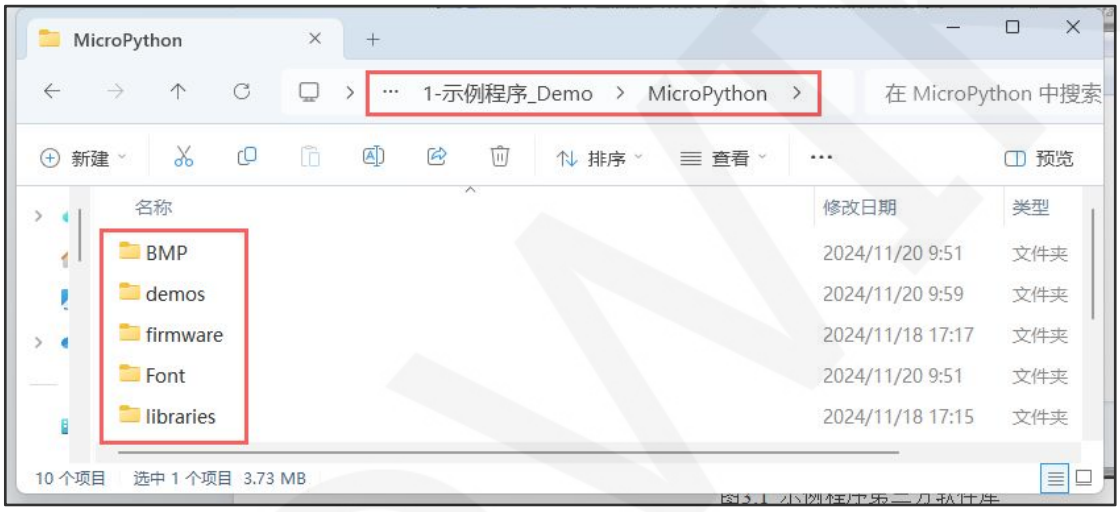

图3.1 MicroPython示例程序目录

各文件夹内容介绍如下：

BMP：存放有示例程序需要使用的BMP格式图片。

demos：存放有示例程序

firmware：存放有MicroPython固件（搭建开发环境时需要烧录）

Font：存放有示例程序需要使用的中英文字符取模数据。

libraries：存放有示例程序需要使用的 MicroPython 库文件

了解完 MicroPython 示例程序的目录内容后，接下来上传程序文件到 ESP32 设备，步骤如下：

A、使用USB线将ESP32显示模块连接电脑上电。

B、打开Thonny软件，配置ESP32的MicroPython解释器，如下图所示：

(如果已经配置，该步骤可省略)

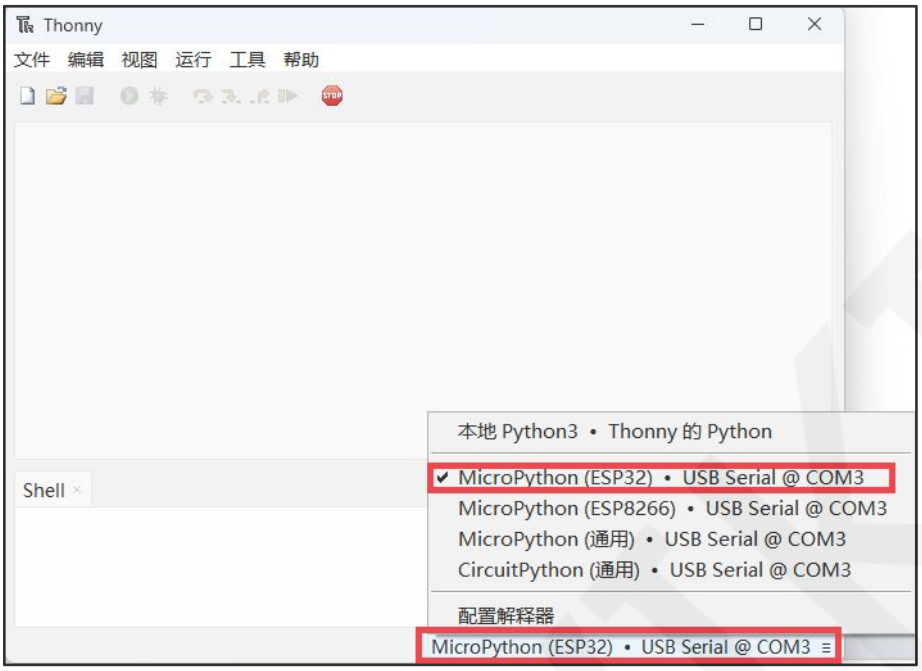

图3.2 选择MicroPython解释器

C、点击工具栏  按钮连接ESP32设备，待shell信息栏出现如下提示，则说明设备连接成功。

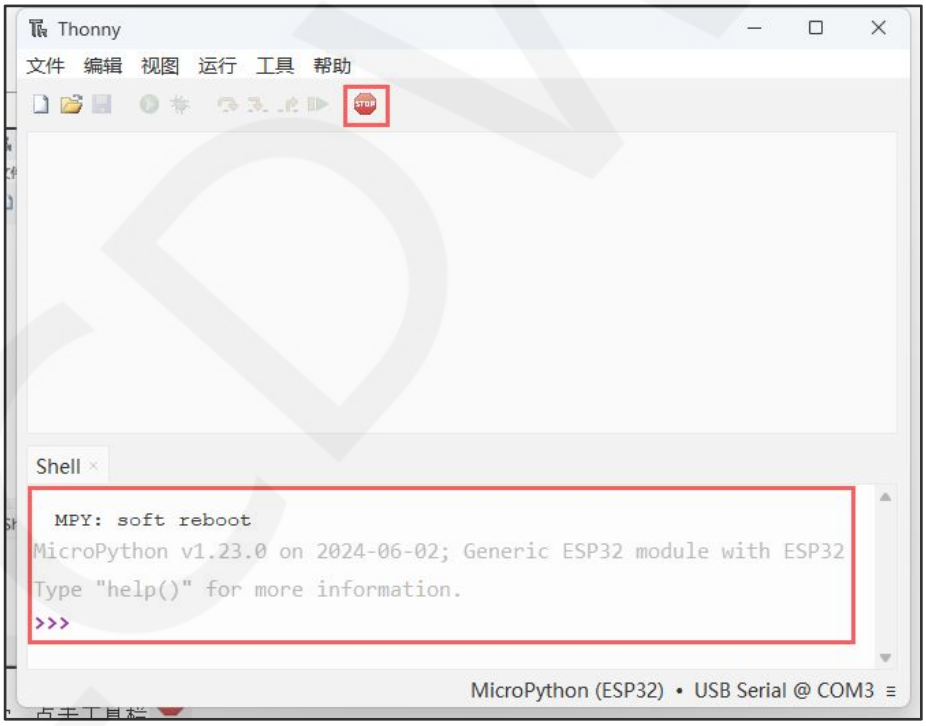

图3.3 连接ESP32设备

D、点击“**视图->文件**”按钮，打开文件窗口（如已打开，则忽略该操作），在窗口中找到资料包里的“**1-示例程序_Demo\MicroPython**”目录，单击鼠标左键选择目录里目标文件，在单机鼠标右键选择“**上传到/**”，就可以将目标文件

上传了。如下图所示：

需要注意，上传文件时，ESP32不能运行任何程序，否则会上传失败

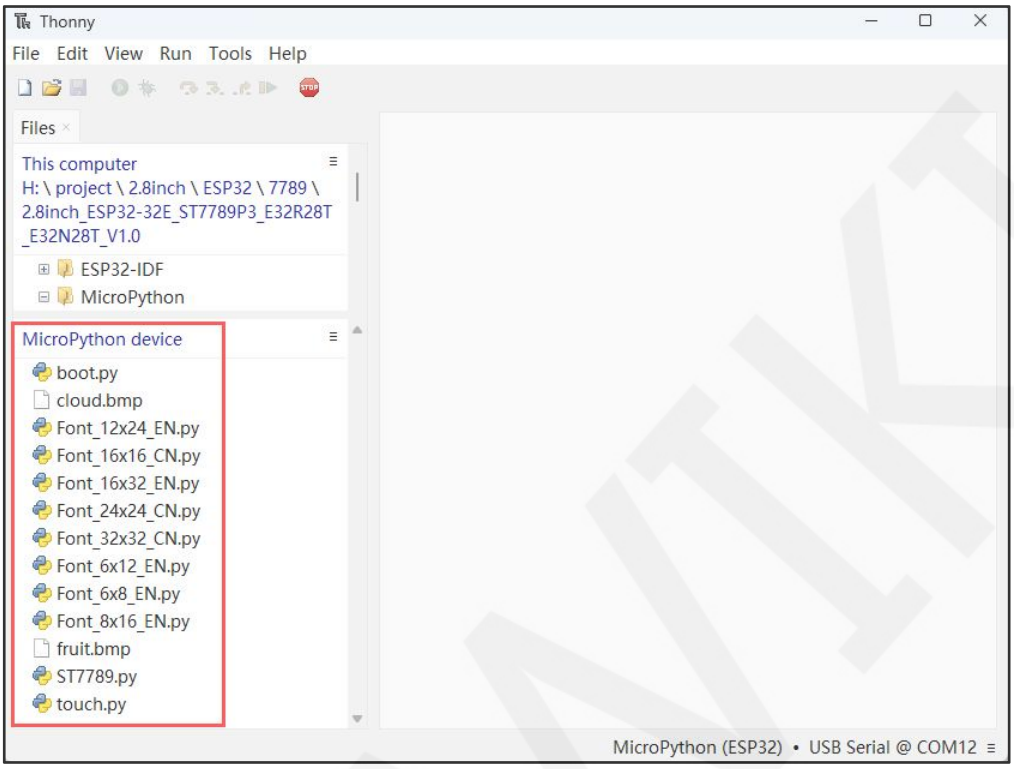

图3.4 上传文件到ESP32设备

E、按上述方法分别将“BMP”、“Font”、“libraries”目录下的文件上传到ESP32设备。“demos”目录下的文件可传可不传。如下图所示：

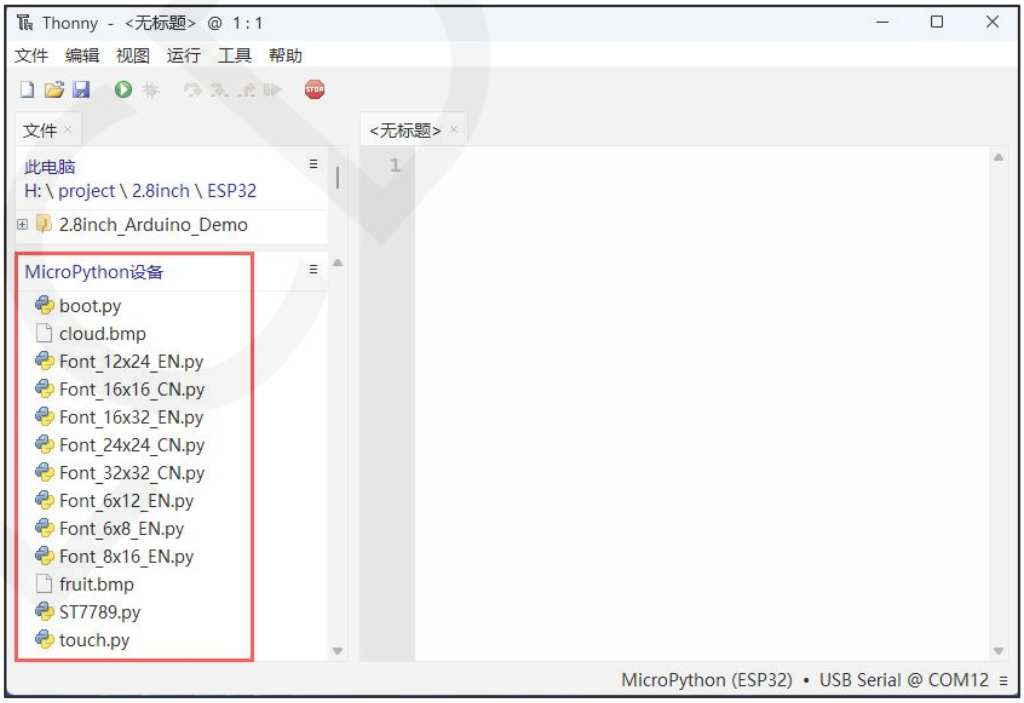

图3.5 完成文件上传

3.3. 示例程序使用说明

示例程序位于资料包的“**1-示例程序_Demo\MicroPython\demos**”目录下，如下图所示：

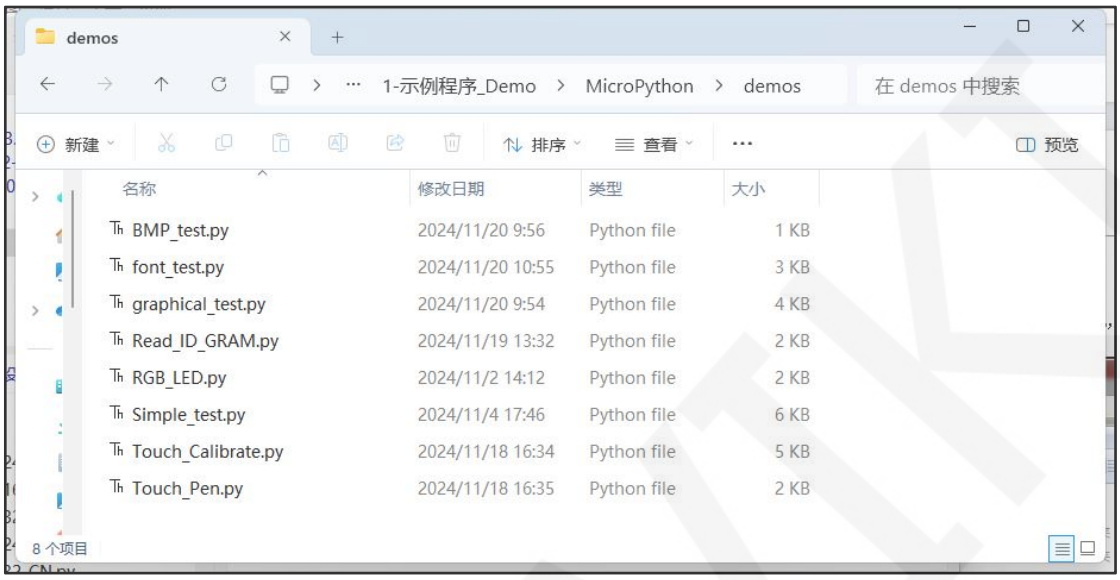

图 3.6 示例程序

示例程序可以上传到 ESP32 设备打开运行，也可以在本地电脑里打开运行。如果需要在 ESP32 显示模块里上电自动运行，需要将示例程序名称改为“**main.py**”,然后上传到 ESP32 显示模块。

在 thonny 软件里，打开目标示例程序，点击菜单栏  按钮，就可以运行了。如果运行失败，需要重新连接 ESP32 设备。

各示例程序介绍如下：

BMP_test.py

此示例程序需要依赖 ST7789.py 库，显示内容为 BMP 格式的图片

font_test.py

此示例程序需要依赖 ST7789.py 库，显示内容为各种尺寸的中英文字符。字体取模数据需要按照相关的格式保存在字体文件里。关于字符取模说明，请查阅如下网址：

http://www.lcdwiki.com/zh/%E3%80%90%E6%95%99%E7%A8%8B%E3%80%91%E4%B8%AD%E8%8B%B1%E6%96%87%E6%98%BE%E7%A4%BA%E5%8F%96%E6%A8%A1%E8%AE%BE%E7%BD%AE

graphical_test.py

此示例程序需要依赖 ST7789.py 库，显示内容为点、线、矩形、圆角矩形、三角形、

圆形，椭圆形等图形绘制和填充以及显示方向设置。

Read_ID_GRAM.py

此示例程序需要依赖 ST7789.py 库，显示内容为 LCD 的 ID 和 RGAM 颜色值读取。

RGB_LED.py

此示例硬件需要用到 RGB 三色灯，用来展示 RGB 三色灯亮灭以及亮度调节。

Simple_test.py

此示例不依赖任何软件库，显示内容为简单的刷屏。

Touch_Calibrate.py

此示例需要依赖 ST7789.py 库和 touch.py 库，显示内容为电阻触摸屏校准。按照屏幕显示的提示操作。校准完成后，校准参数通过串口输出，将校准参数拷贝到示例程序的初始化里。需要注意，要根据显示方向来校准触摸屏，可修改该程序里的显示方向，如下图所示：

```
if __name__=='__main__':
    coord = [0xFFFF, 0xFFFF]
    val = [0, 0, 0, 0, 0, 0, 0, 0]
    mylcd.LCD_Set_Rotate(1)
    mylcd.LCD_Clear(0)
    mylcd.Show_String((mylcd.lcd_width - 208) // 2, (mylcd.lcd_height-16) /
    for i in range(4):
        mylcd.Fill_Rect(0, 0, mylcd.lcd_width, 16, 0)
        mylcd.Fill_Rect(0, mylcd.lcd_height-16, mylcd.lcd_width, 16, 0)
        switch = {
            0: case_0,
            1: case_1,
            2: case_2,
```

图 3.7 修改触摸校准显示方向

Touch_Pen.py

此示例需要依赖 ST7789.py 库和 touch.py 库，显示内容为触摸屏画点画线操作。