

# ESP32-S3 MicroPython

## 开发环境搭建

# 目 录

1. Thonny 软件安装包下载.....	3
2. Thonny 软件安装.....	5
3. Thonny 软件介绍.....	11
3.1. 菜单栏.....	11
3.1.1. 文件菜单.....	11
3.1.2. 编辑菜单.....	12
3.1.3. 视图菜单.....	13
3.1.4. 运行菜单.....	13
3.1.5. 工具菜单.....	14
3.1.6. 帮助菜单.....	20
3.2. 工具栏.....	22
4. 下载与烧录 ESP32-S3 MicroPython 固件.....	23
4.1. 下载 ESP32-S3 MicroPython 固件.....	24
4.2. 烧录 ESP32-S3 MicroPython 固件.....	26
4.2.1. 使用 Thonny 软件烧录固件.....	27
4.2.2. 使用 flash_download_tool 工具烧录.....	35
5. 编辑、保存并运行 ESP32-S3 MicroPython 程序.....	38
5.1. 配置 MicroPython 解释器.....	38
5.2. 编辑 ESP32-S3 MicroPython 程序.....	38
5.3. 保存并运行 ESP32-S3 MicroPython 程序.....	40

## 1. Thonny 软件安装包下载

选择 MicroPython 方式进行 ESP32-S3 开发，首先得选择一款开发工具软件，在 Windows 系统下可以选择的开发工具软件有：Thonny、VS Code、PyDev、Pycharm 等，其中 Thonny 是一款适合初学者的 IDE，它易于上手，功能基本够用。其他的开发工具软件虽然功能强大，但是对于初学者来说，上手难度较大，当然如果是有一定开发基础的学者，可以选择这些功能强大，上手难度大的 IDE。

Thonny 软件安装包可以直接从官网下载。

官网地址：<https://thonny.org/>

进入官网页面后，可以看到最新的版本号，还提供各种电脑系统版本供选择，如下图所示：

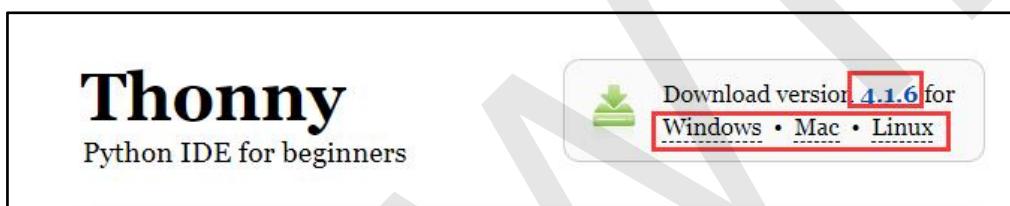


图 1.1 Thonny 软件安装包下载界面 1

根据自己的电脑系统选择相应的版本下载（将鼠标移到相应的电脑系统名称上就会弹出软件版本供选择）。这里使用 Windows 系统，那么将鼠标移到“Windows”上，弹出版本下载界面，如下图所示。

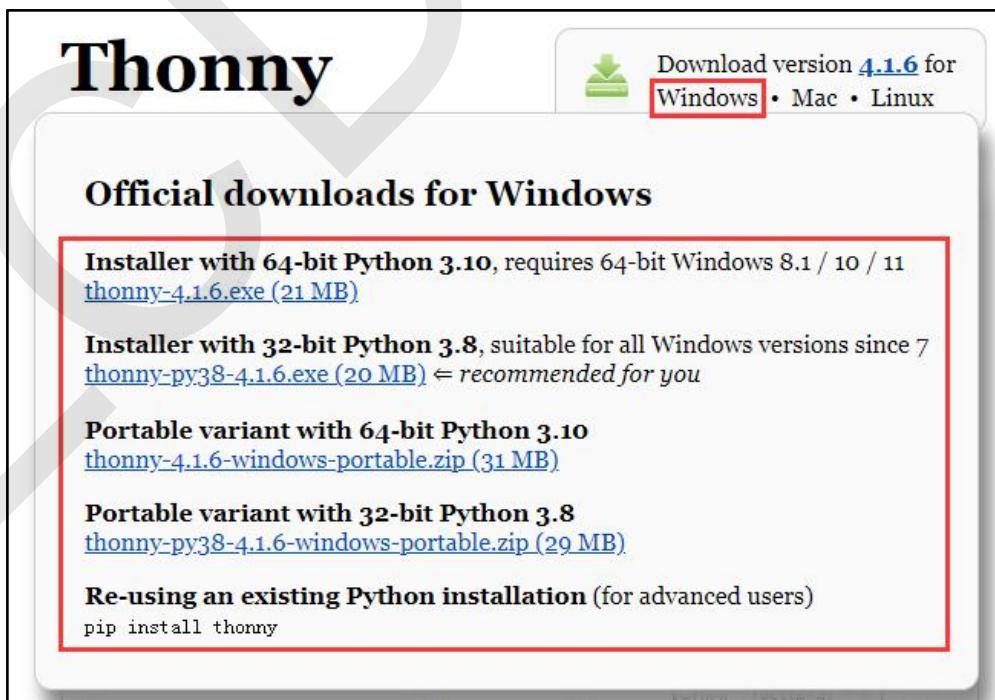


图 1.2 Thonny 软件安装包下载界面 2

各种安装包说明如下：

以下两个安装包需使用 **Installer**一步一步安装，包括安装 Python 环境和 Thonny 软件。

**Installer with 64-bit Python 3.10** (用于 Win8 及以上系统)

**Installer with 32-bit Python 3.8** (用于 Win7 及以下系统)

以下两个安装包已经包含 Python 环境和 Thonny 软件，不需安装，下载后解压就可使用。

**Portable variant with 64-bit Python 3.10** (用于 Win8 及以上系统)

**Portable variant with 32-bit Python 3.8** (用于 Win7 及以下系统)

以下方式为只安装 Thonny 软件（系统已经安装 Python 环境），通过执行“**pip install thonny**”命令安装。

**Re-using an existing Python installation**

这里只介绍使用 **Installer** 安装方法，点击“**Installer with 32-bit Python 3.8**”下载安装包，如下图所示，自行选择安装包存放路径。



图 1.3 Thonny 软件安装包下载任务

## 2. Thonny 软件安装

软件安装包下载完成后，打开保存文件夹，然后双击 exe 文件，进入程序安装（如果弹出询问是否运行文件窗口，直接点击“运行”按钮），如下图所示：

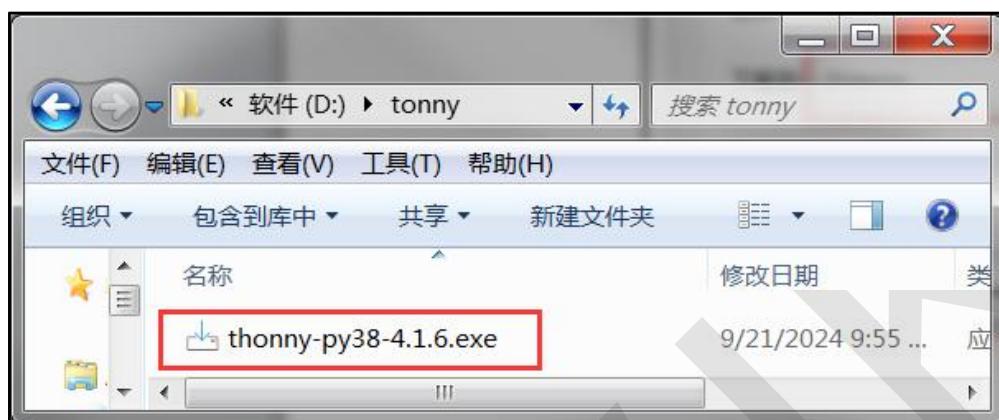


图 2.1 Thonny 软件安装包 exe 文件

最开始弹出安装模式选择界面，这里选择只单独为个人用户安装，如下图所示：

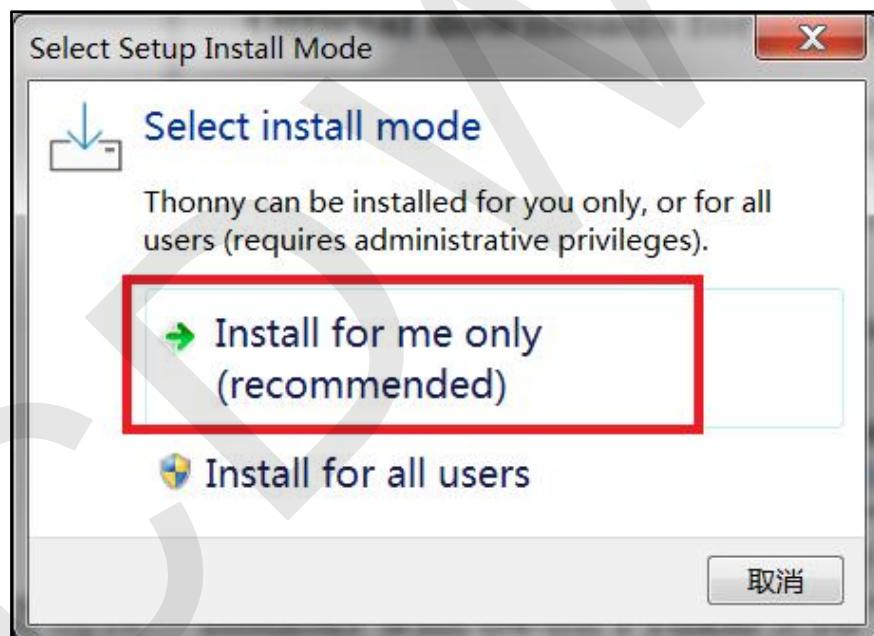


图 2.2 Thonny 软件安装模式选择

接下来进入欢迎使用 Thonny 界面，直接点击“Next”按钮，如下图所示：

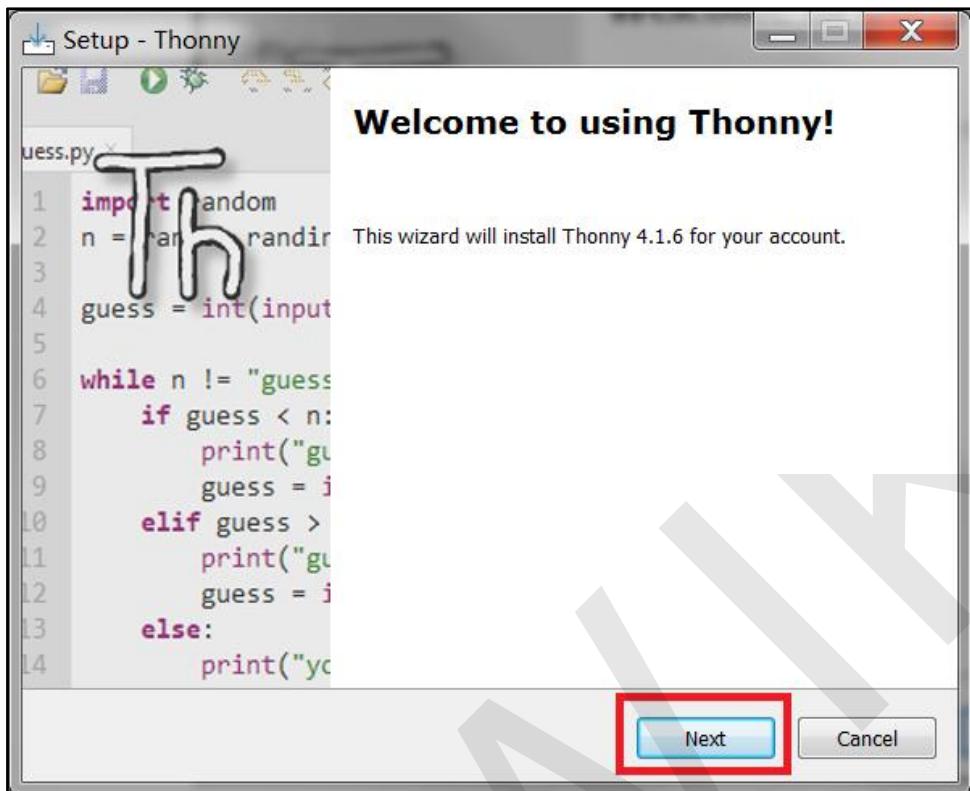


图 2.3 Thonny 软件欢迎使用界面

接下来进入是否许可协议界面，选择“*I accept the agreement*”，然后点击“*Next*”按钮，如下图所示：

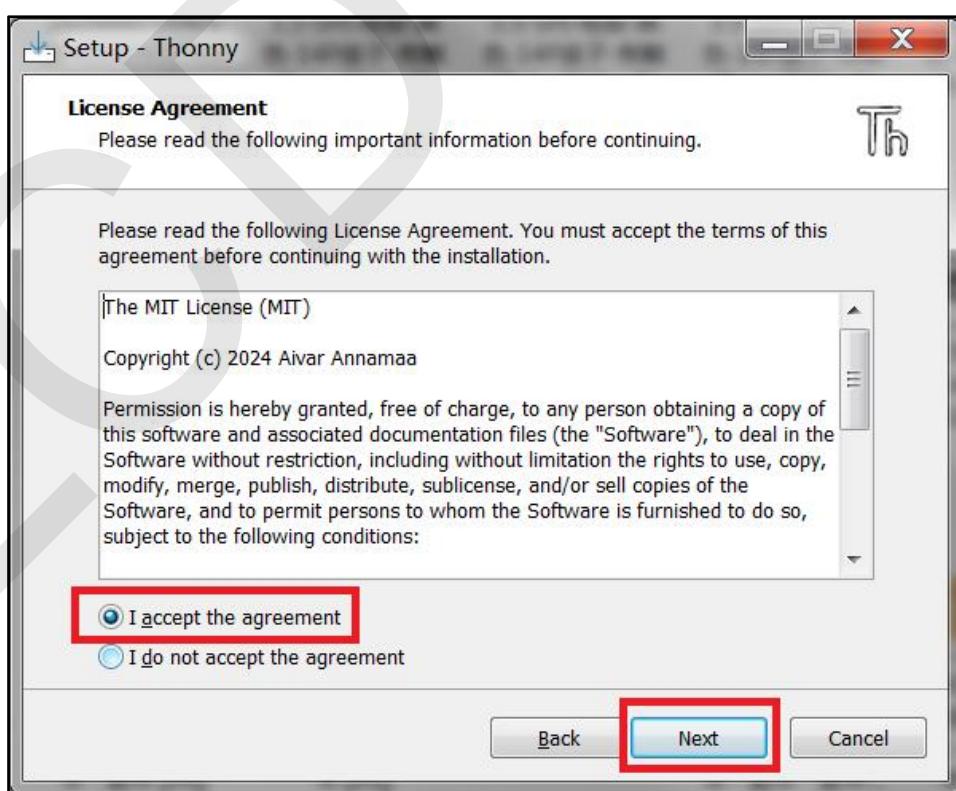


图 2.4 Thonny 软件许可协议选择

接下来进入选择安装目录的界面，点击“**Browse...**”按钮选择安装目录（也可使用默认目录），然后点击“**Next**”按钮，如下图所示：

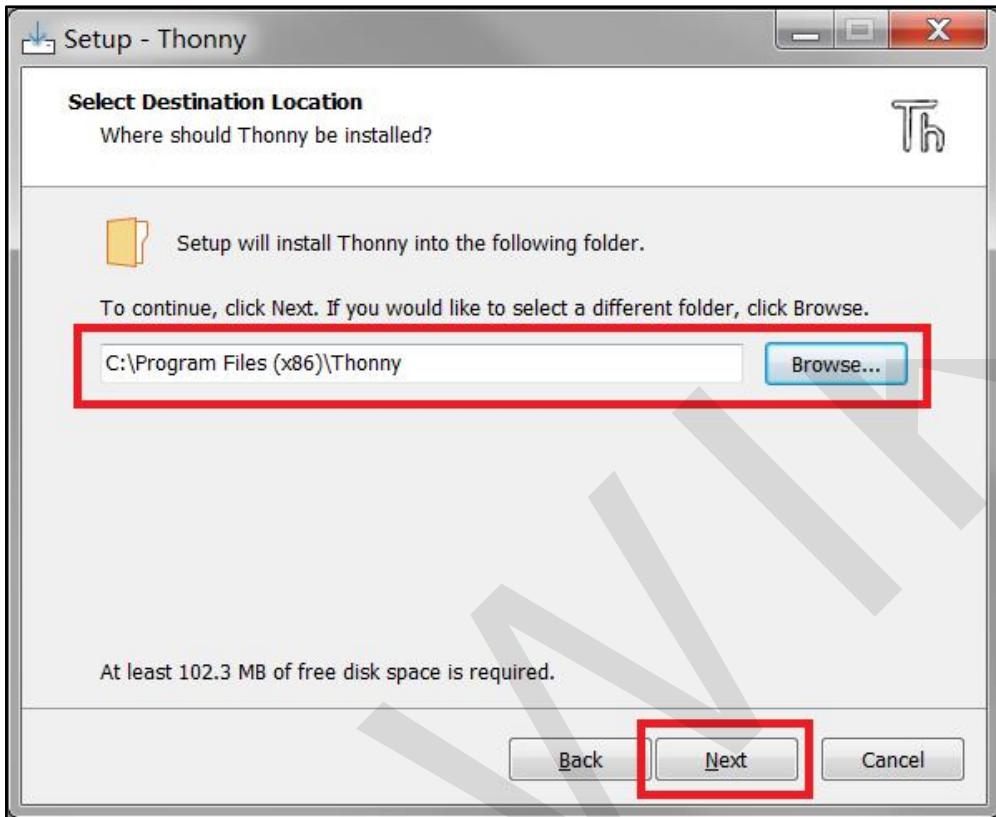


图 2.5 Thonny 软件安装目录选择

接下来进入开始菜单文件夹选择界面，此文件夹用来存放开始菜单栏里的快捷图标，点击“**Browse...**”按钮选择（也可使用默认文件夹），然后点击“**Next**”按钮，如下图所示：

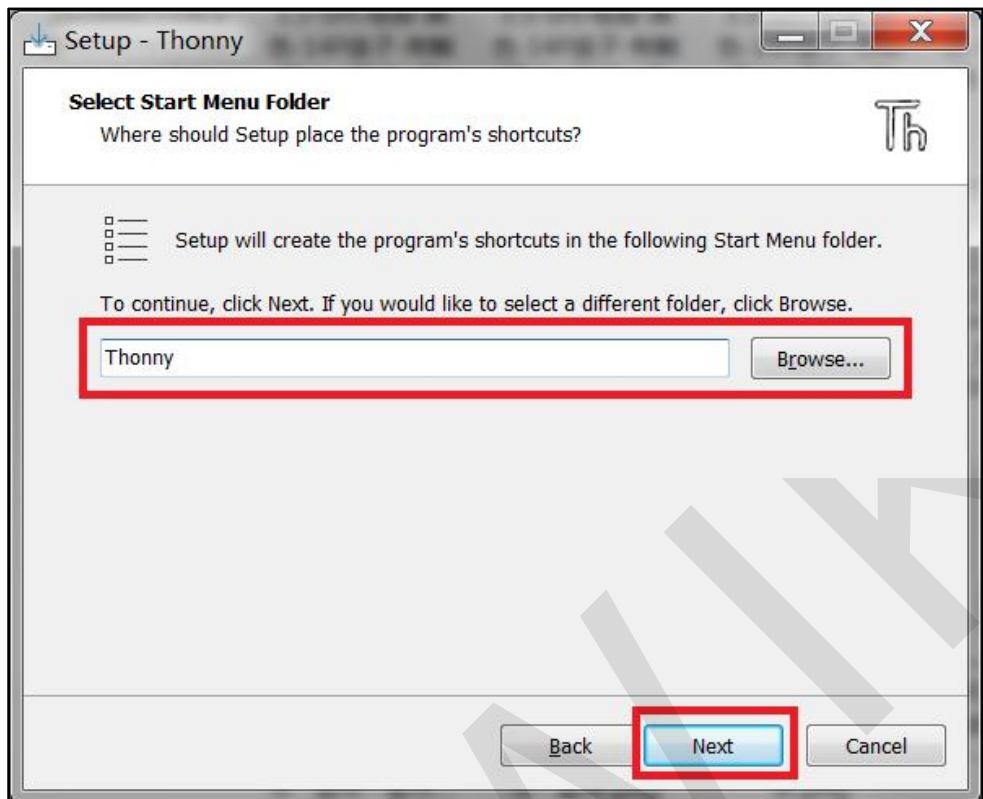


图 2.6 Thonny 软件开始菜单目录选择

接下来进入选择是否创建桌面图标界面，勾选“Create desktop icon”就会创建桌面图标，不勾选则不会。为了方便打开，一般选择勾选，然后点击“Next”按钮，如下图所示：

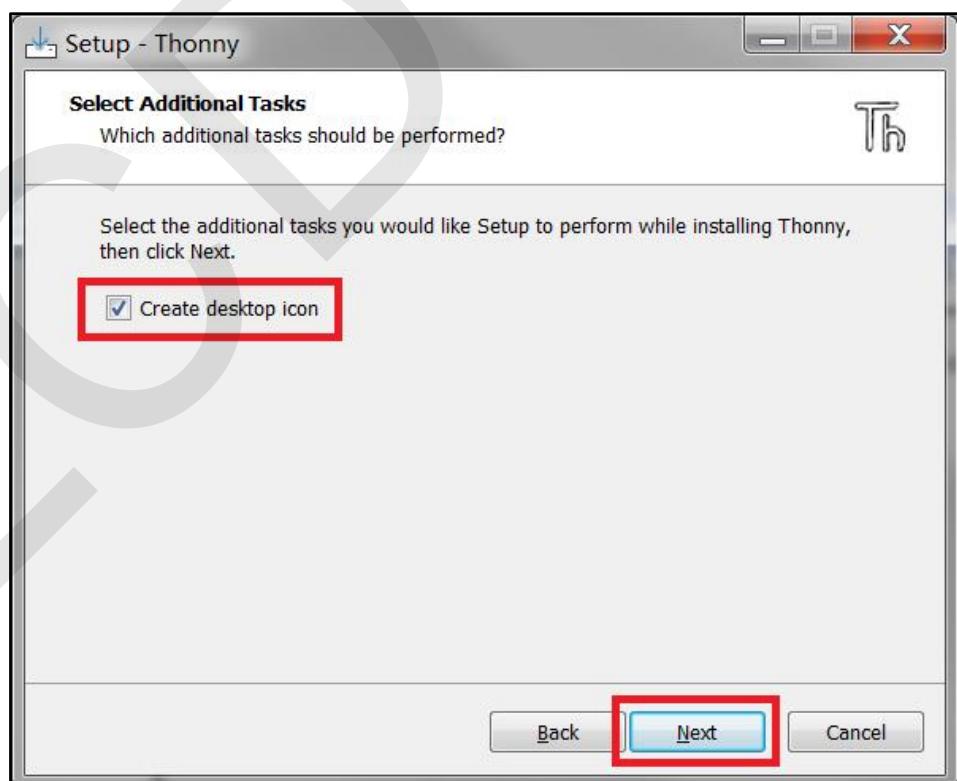


图 2.7 Thonny 软件创建桌面图标选择

接下来进入准备安装界面，点击“Install”按钮进行安装，如下图所示：

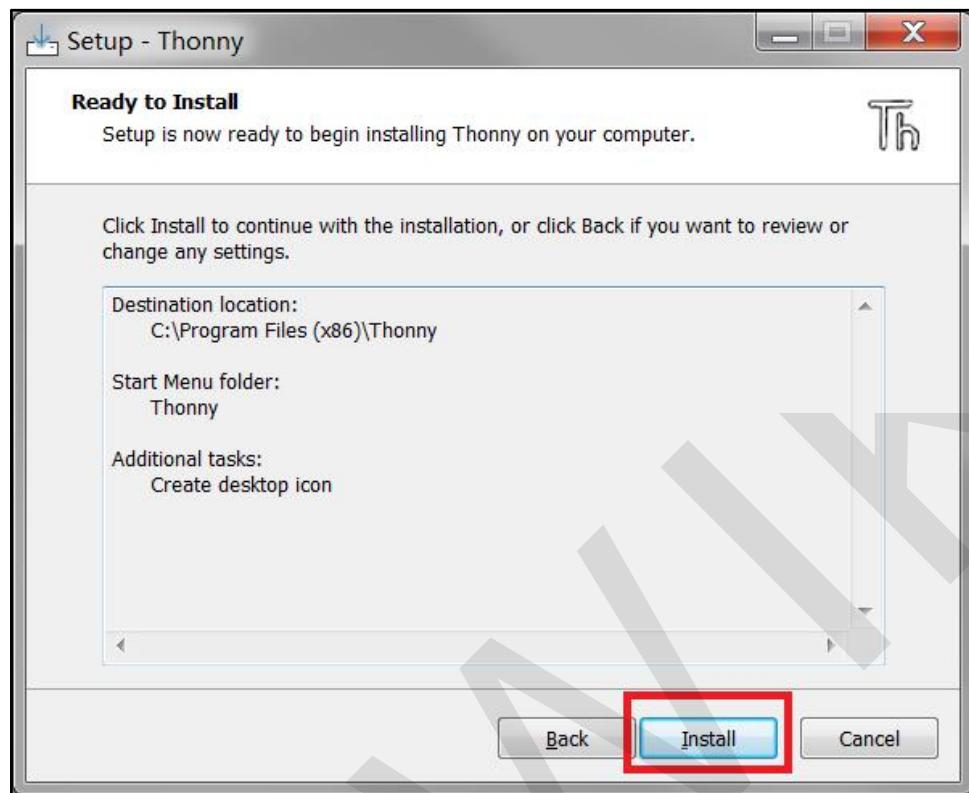


图 2.8 准备安装 Thonny 软件

接下来进入软件安装阶段，待进度条滚动完成，软件就安装完成，如下图所示：

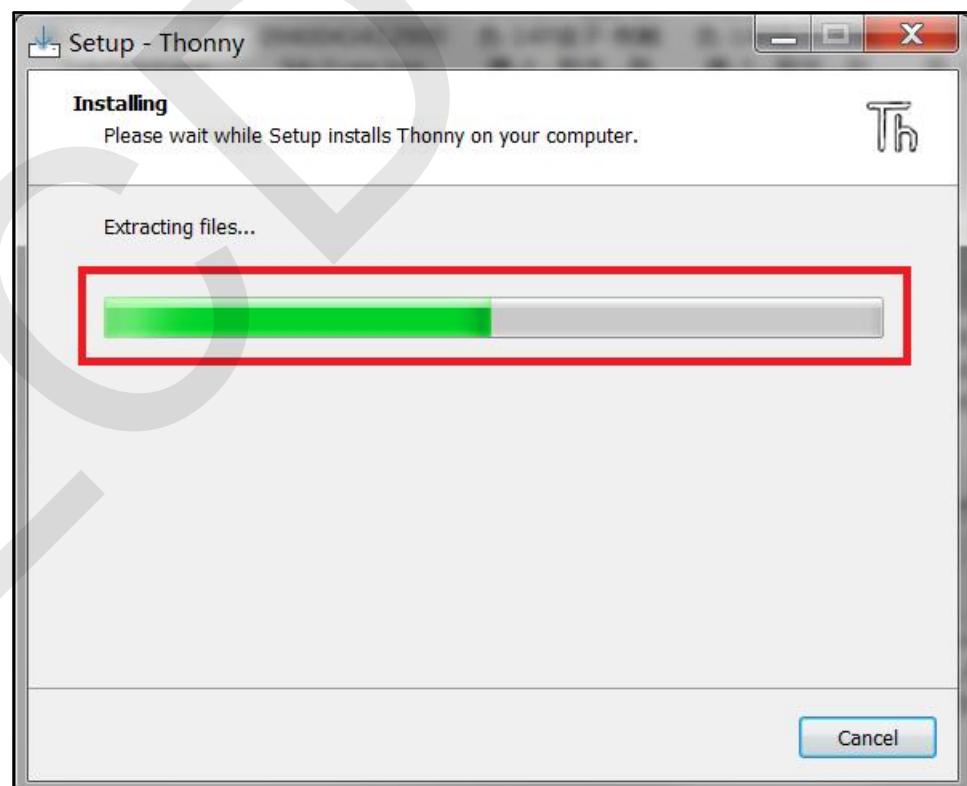


图 2.9 Thonny 软件安装

接下来进入软件安装完成界面，点击“Finish”按钮关闭界面，如下图所示：



图 2.10 Thonny 软件安装完成

通过以上步骤，可以看到电脑桌面和开始菜单栏都生成了 Thonny 快捷方式图标。点击图标打开 Thonny 软件（第一次运行时，会提示选择语言和初始化设置，这里选择中文和标准设置即可），在编辑栏里输入如下代码，然后点击运行图标，可以看到 Shell 栏里有内容输出，如下图所示。至此在 Windows 下安装 Thonny 软件就正常完成了。

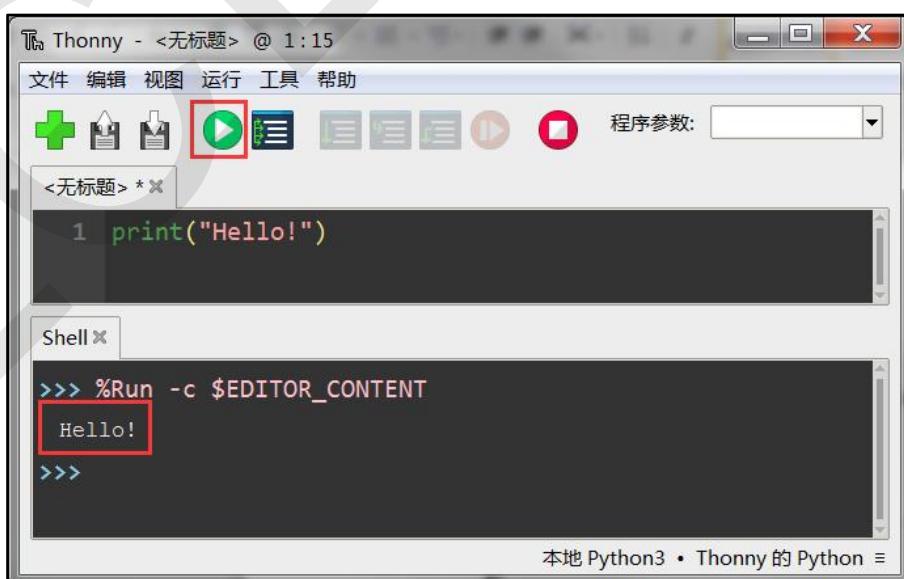


图 2.11 Thonny 软件正常运行

### 3. Thonny 软件介绍

Thonny 软件具有简单易用、程序代码编辑、自动补齐、调试、编译、上传、安装方便等特点，主界面如下图所示：

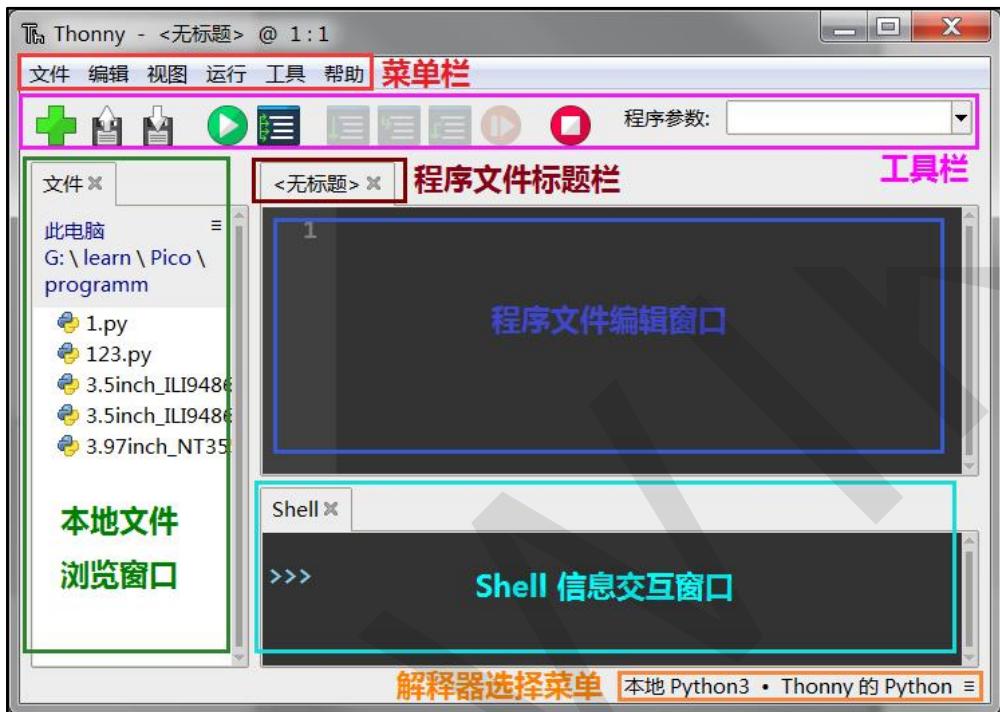


图 3.1 Thonny 主界面

**菜单栏:** Thonny 软件主要的设置选项，绝大多数设置都在这里面。

**工具栏:** 一些常用操作的按钮，方便操作（在菜单栏里也可以执行）

**本地文件浏览窗口:** 查看电脑上保存的 Python 文件

**程序文件标题栏:** 显示 Python 文件的名称，显示多个文件时，方便切换。

**程序文件编辑窗口:** 编辑 Python 文件。

**Shell 信息交互窗口:** 查看 Python 程序运行的结果，同时支持输入命令。

**解释器选择菜单:** 设置 Python 解释器。

#### 3.1. 菜单栏

##### 3.1.1. 文件菜单

菜单栏文件菜单里操作主要是对文件进行处理，包括新建文件、打开已存在的文件、打开最近操作的文件、关闭当前或全部文件、保存当前或全部文件、文件另存为、保存文件为副本、移动或重命名文件、打印文件，还有关闭 Thonny 软件功能。文件菜单栏界面如下图所示：

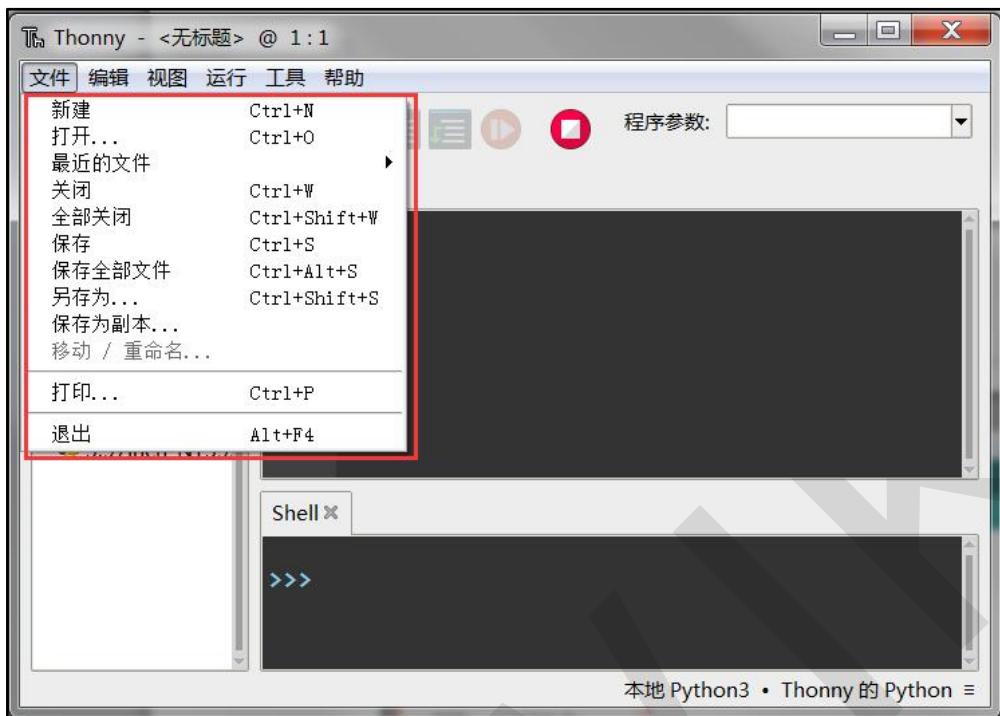


图 3.2 Thonny 文件菜单栏

### 3.1.2 . 编辑菜单

菜单栏编辑菜单里提供对文件进行编辑的操作，包括撤销、重做、剪切、复制、粘贴、全选、代码行缩进调整、代码注释调整、跳转到指定代码行、代码自动补齐、代码参数显示、查找和替换、清空 Shell 信息等操作，编辑菜单界面如下图所示：

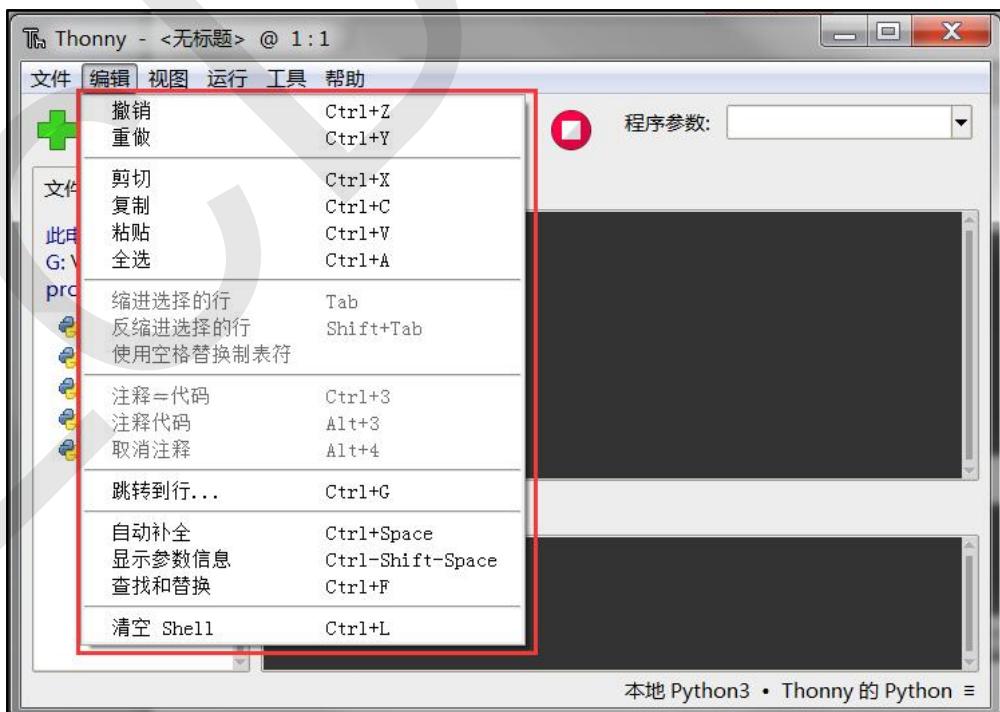


图 3.3 Thonny 编辑菜单栏

### 3.1.3. 视图菜单

菜单栏视图菜单提供关闭或打开某个视图窗口的操作，Thonny 提供了十几种信息显示窗口，信息显示十分直观。点击相应的窗口名称，名称前面出现“√”则表示窗口已打开，再点击一次，则关闭窗口。另外还提供界面字体尺寸调整功能和编辑器/Shell 窗口切换功能。视图菜单界面如下图所示：

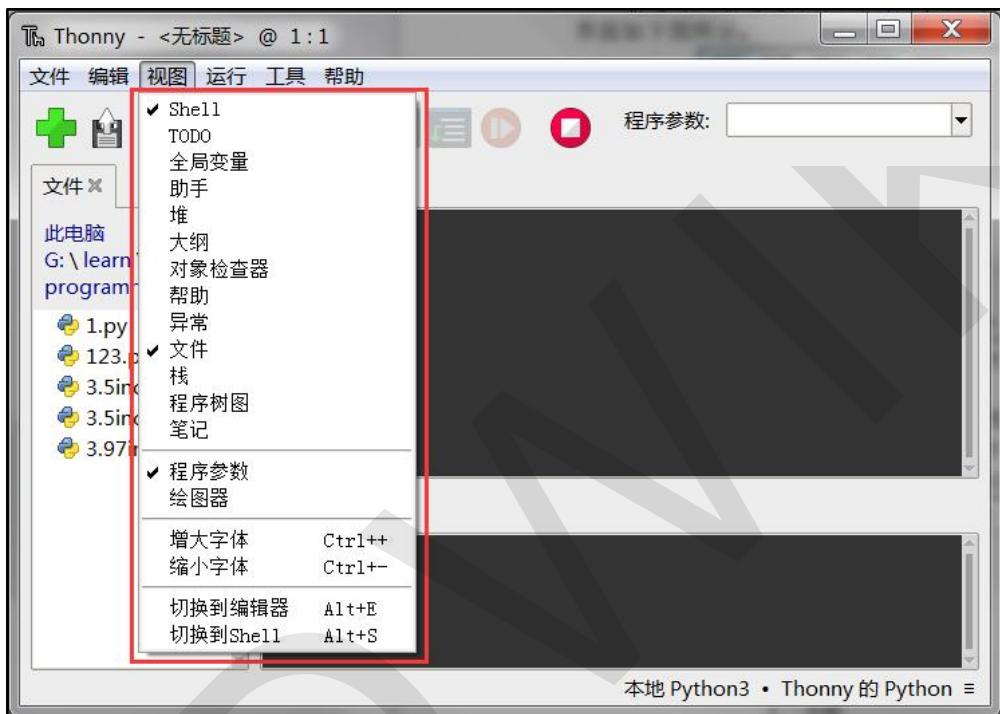


图 3.4 Thonny 视图菜单栏

### 3.1.4. 运行菜单

菜单栏运行菜单提供了配置解释器、运行 Python 程序、调试 Python 程序、停止/重启后端进程、中断执行、发送 EOF/软重启等操作。其中调试方式有 nicer、Faster、birdseye 三种，还可以对程序进行单步运行，方便用户查看程序运行的过程。运行菜单界面如下图所示：

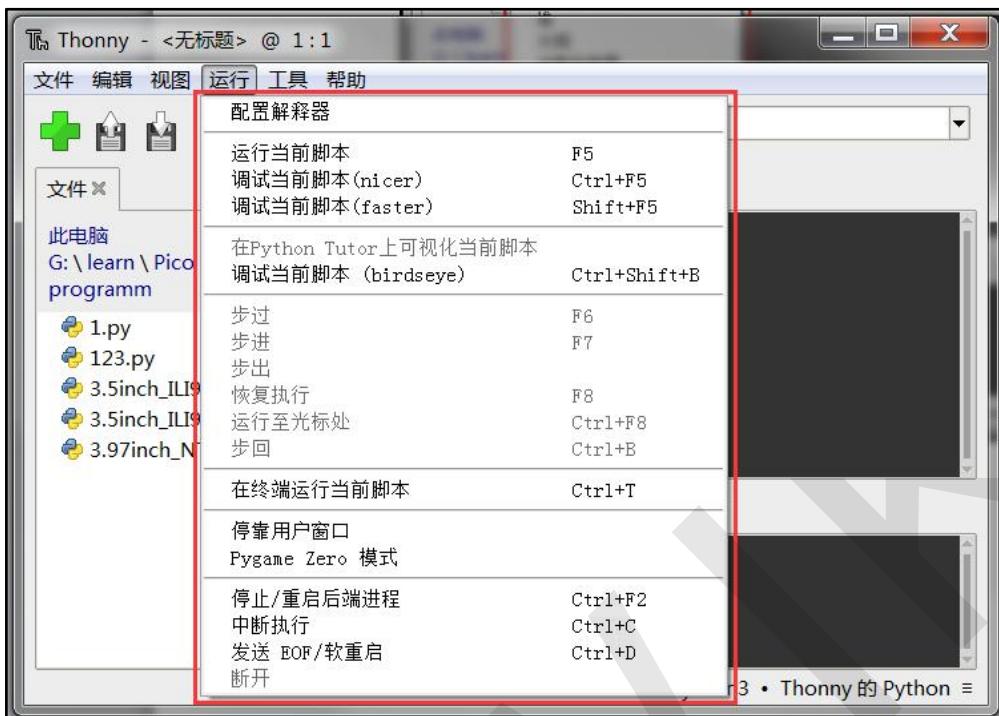


图 3.5 Thonny 运行菜单栏

### 3.1.5. 工具菜单

工具菜单栏主要提供软件包管理、打开系统 Shell 窗口、打开 Thonny 安装目录和数据目录、插件管理、选项设置等操作，工具菜单界面如下图所示：

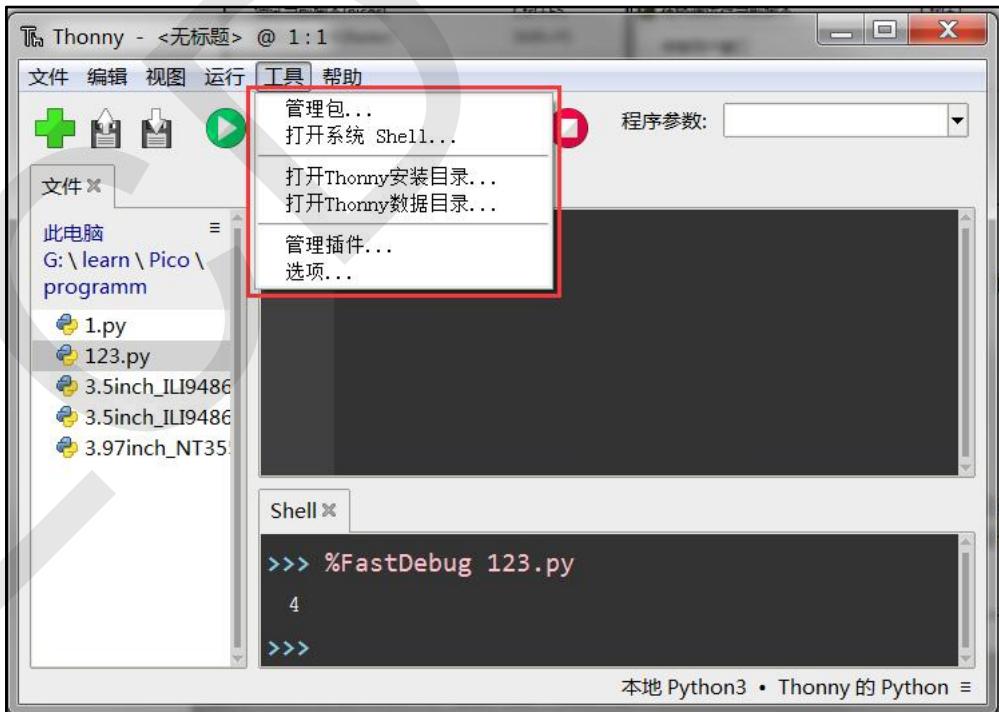


图 3.6 Thonny 工具菜单栏

点击工具菜单的“管理包...”选项，进入软件包安装管理界面，左边列出了已经

安装的软件包，点击相应的包名称，可以查看详细信息。界面提供了三种软件包的安装方式：第一种是从搜索栏输入包名称，点击“在 PyPI 上搜索”按钮搜索，然后根据搜索结果安装，第二种是根据文件中指定的包安装，第三种是从本地文件中安装。



图 3.7 Thonny 软件包管理

点击工具菜单的“管理插件...”选项，进入插件安装管理界面，从搜索栏输入插件名称，点击“在 PyPI 上搜索”按钮搜索，然后根据搜索结果安装。



图 3.8 Thonny 插件管理

点击工具菜单的“选项...”选项，进入 Thonny 选项界面。首先是常规设置界面，这里面提供运行程序时的一些设置，还有界面语言设置、UI 相关的设置以及环境变量设置，设置完成后需要重启生效。界面如下图所示：

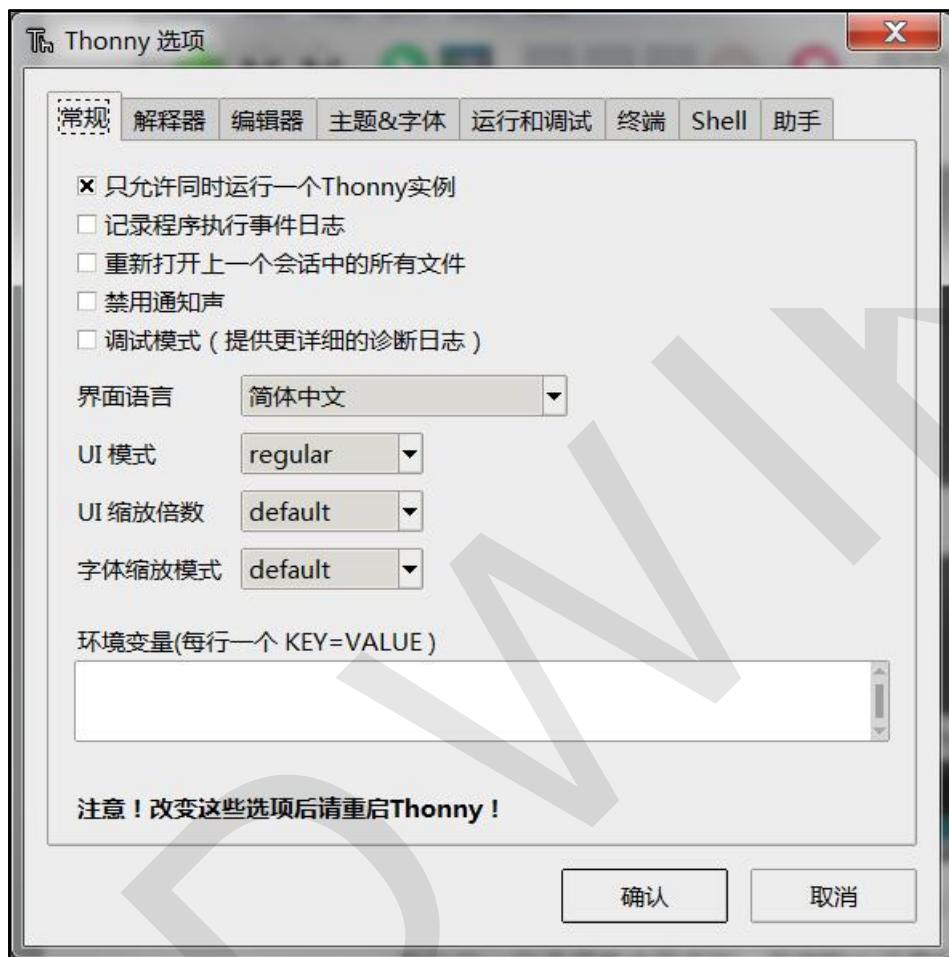


图 3.9 Thonny 选项常规设置

点击“解释器”按钮，进入 Python 解释器设置界面。Python 解释器是 Python 程序运行的核心，负责对 Python 程序进行解释，将其转换为机器语言，从而使计算机能够执行。不同的硬件平台需要选择相应的 Python 解释器，否者运行 Python 会出现异常。界面如下图所示（这里解释器设置和运行菜单里配置解释器一致）：

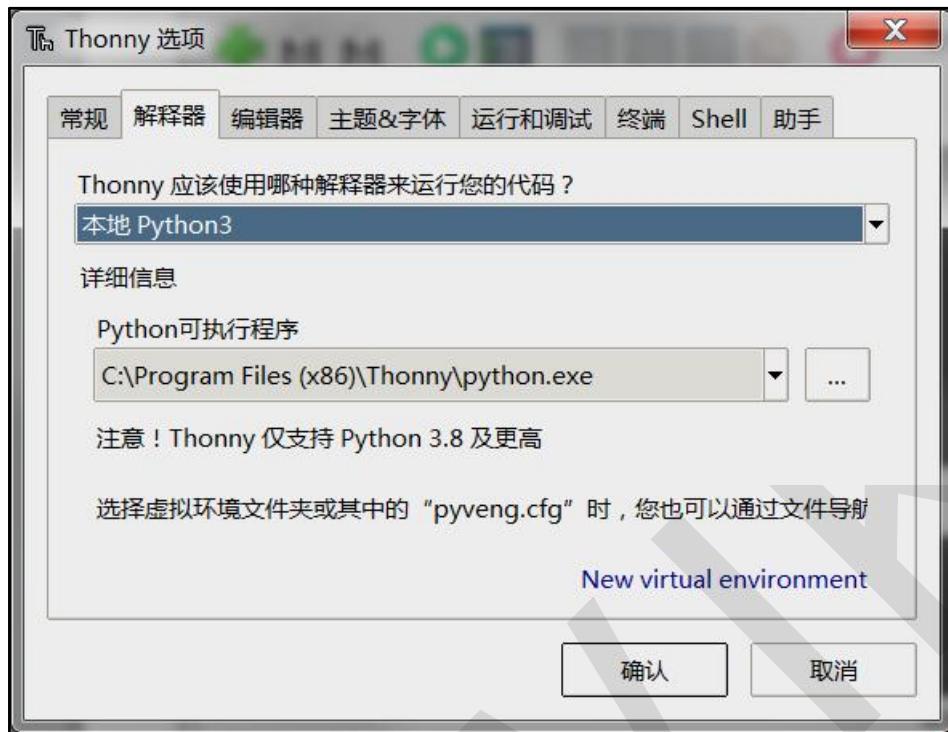


图 3.10 Thonny 选项解释器设置

点击“编辑器”按钮，进入 Python 编辑器设置界面，这里主要是对代码编辑器的输入内容进行显示设置，包括高亮显示，自动提示完成等设置，界面如下图所示：

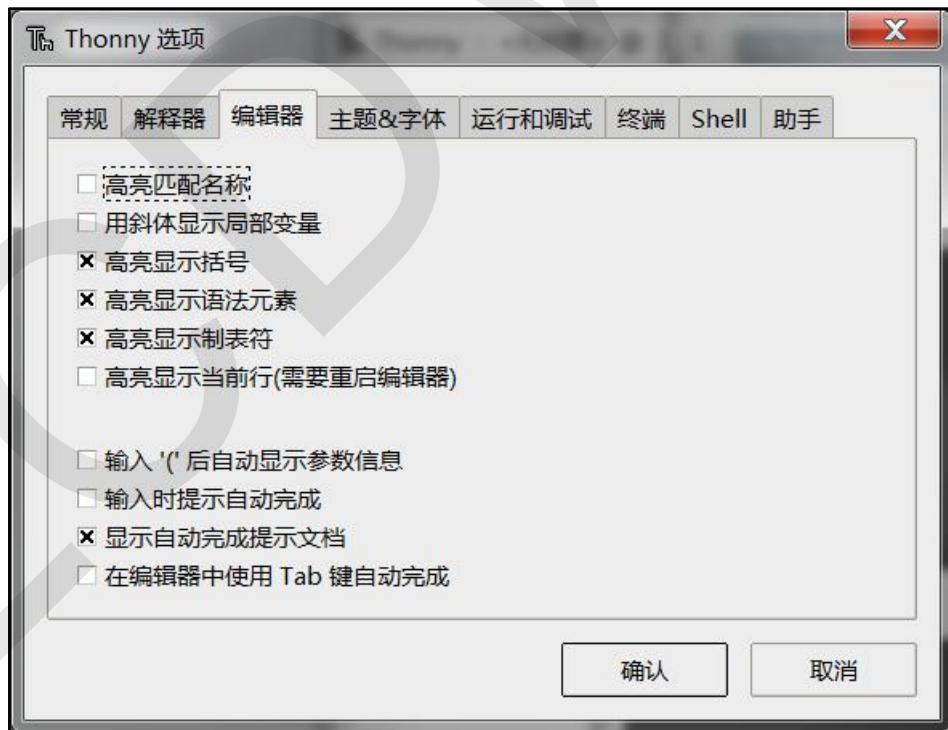


图 3.11 Thonny 选项编辑器设置

点击“主题&字体”按钮，进入软件主题和字体设置界面，这里主要是对软件显示的主题风格，字体样式和尺寸进行设置，给用户提供个性化的选择。界面如下图所示：



图 3.12 Thonny 选项主题和字体设置

点击“运行和调试”按钮，进入程序运行和调试设置界面。界面如下图所示：

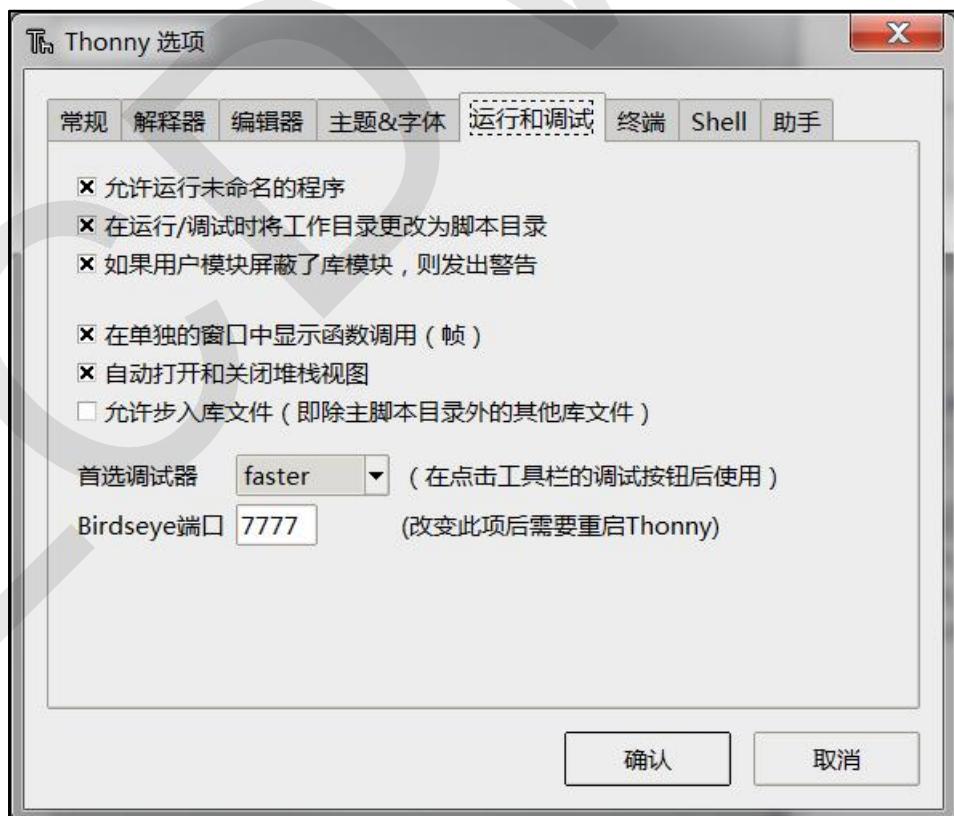


图 3.13 Thonny 选项运行和调试设置

点击“终端”按钮，进入程序程序执行时终端设置界面。界面如下图所示：

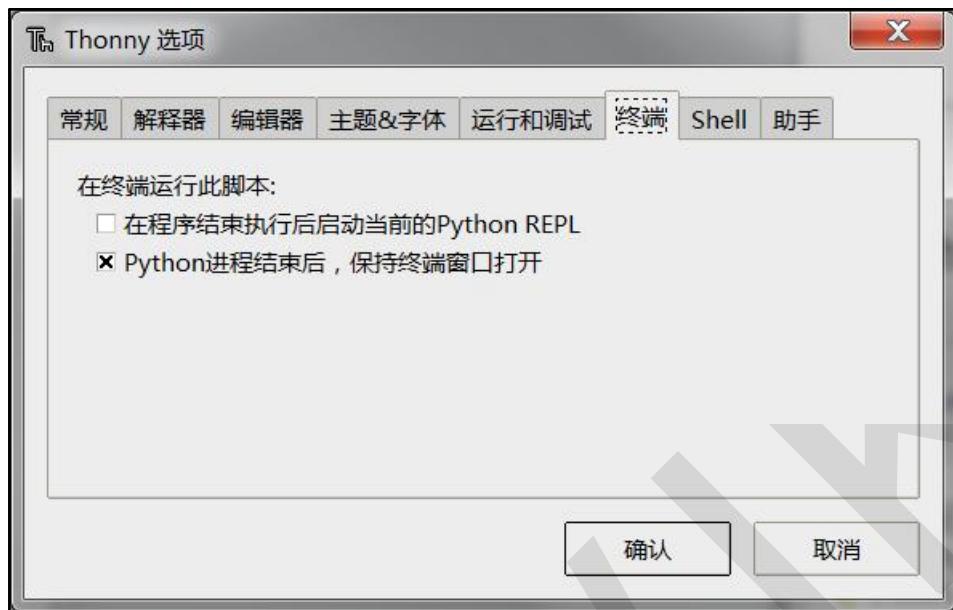


图 3.14 Thonny 选项终端设置

点击“Shell”按钮，进入 Shell 窗口设置界面，主要设置 Shell 的显示内容，包括是否清空 Shell 显示内容，显示的最大行设置，界面如下图所示。



图 3.15 Thonny 选项 Shell 设置

点击“助手”按钮，进入助手设置界面，主要检查代码运行情况，界面如下图所示。一旦设置好相应的选项，点击“确认”按钮。

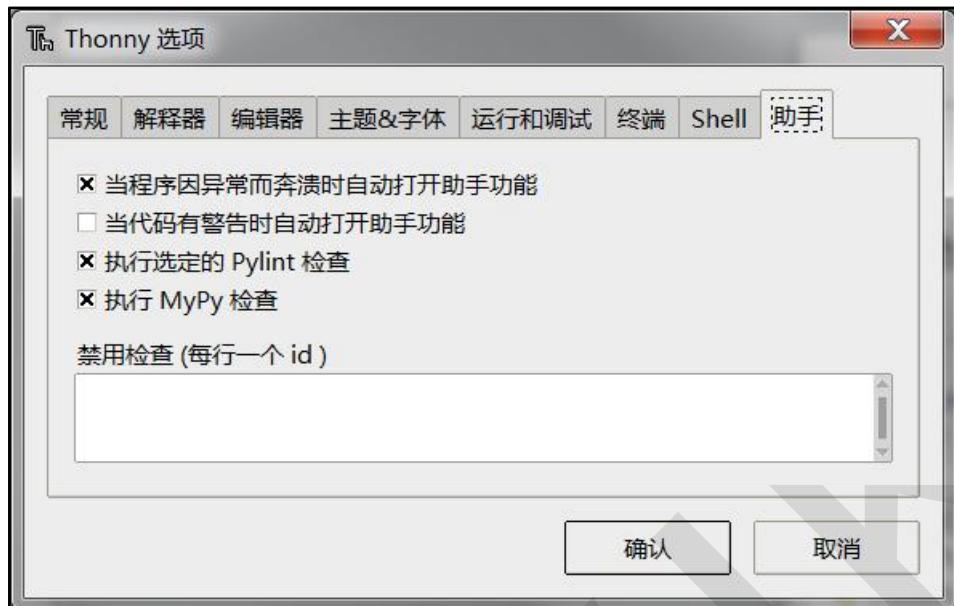


图 3.16 Thonny 选项助手设置

### 3.1.6. 帮助菜单

帮助菜单栏主要提供一些软件使用帮助，查看历史版本，反馈软件问题以及 Thonny 软件的简单介绍，帮助菜单界面如下图所示：

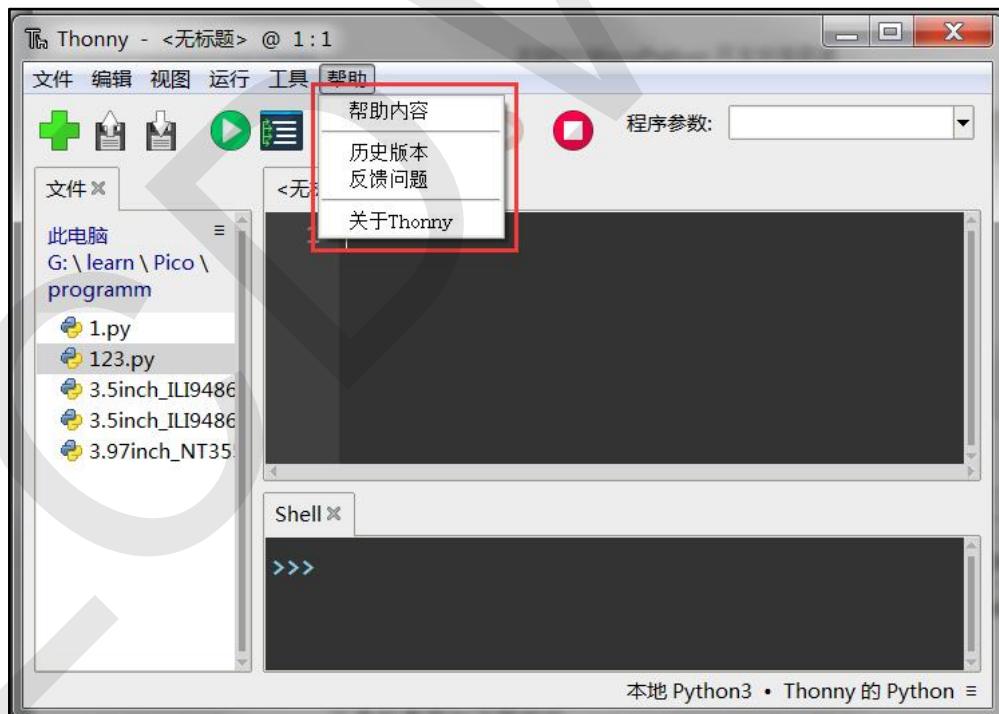


图 3.17 Thonny 帮助菜单栏

### 3.2. 工具栏

工具栏提供了一些常用操作的快捷按钮，方便执行（这些操作在菜单栏里也可以执行）。工具栏界面如下图所示：

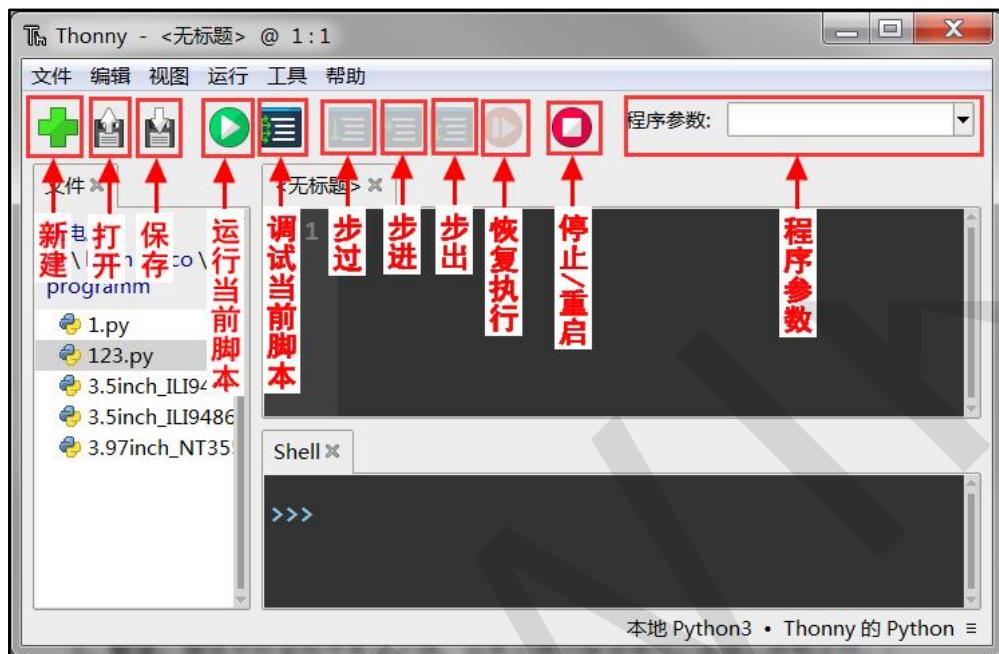


图 3.18 Thonny 工具栏

**新建:** 重新创建一个 Python 文件。

**打开:** 打开已存在的 Python 文件。

**保存:** 保存当前打开的 Python 文件。

**运行当前脚本:** 执行当前打开的 Python 文件。

**调试当前脚本:** 调试当前打开的 Python 文件，查看程序运行情况。

**步过:** 每执行一步运行一个程序函数。

**步进:** 每执行一步运行一行代码（单步调试）。

**步出:** 每执行一步整个程序运行至结束。

**恢复执行:** 调试暂停时，执行该操作，直接运行到程序结束。

以上 4 个操作需要进入调试阶段才能执行。

**停止/重启:** 停止或重启 Shell 后端进程。

**程序参数:** 显示程序里的参数。

## 4. 下载与烧录 ESP32-S3 MicroPython 固件

在 ESP32-S3 上进行 MicroPython 开始，首选需要烧录 ESP32-S3 MicroPython 固件。

ESP32-S3 MicroPython 固件是底层硬件和上层应用层的桥梁，它将底层硬件的操作方法进行封装，然后提供接口和方法给上层应用层调用，这大大降低了开发难度。

### 4.1. 下载 ESP32-S3 MicroPython 固件

ESP32-S3 MicroPython 固件可以从 MicroPython 官网直接下载。

官网下载网址: <https://micropython.org/download/>

进入下载页面后，点击“Port”栏目下的 esp32 选项，如下图所示：

The screenshot shows the MicroPython Downloads page. At the top, it says "MicroPython downloads". Below that, there's a note about the repository being managed via git and available on GitHub. It then lists the full source-code distribution for the latest version, which includes a link to "micropython-1.25.0.tar.xz (104MiB)". Following this, it shows daily snapshots from the GitHub repository, listing "micropython-master.zip" and "pyboard-master.zip". Further down, it mentions firmware for various microcontroller ports and boards built automatically daily, with a note to "Filter by:" followed by a list of supported ports, features, vendors, and MCUs. The "esp32" option is highlighted in red.

MicroPython is developed using git for source code management, and the master repository can be found on GitHub at [github.com/micropython/micropython](https://github.com/micropython/micropython).

The full source-code distribution of the latest version is available for download here:

- [micropython-1.25.0.tar.xz \(104MiB\)](#)

Daily snapshots of the GitHub repository (not including submodules) are available from this server:

- [micropython-master.zip](#)
- [pyboard-master.zip](#)

Firmware for various microcontroller ports and boards are built automatically on a daily basis and can be found below.

Filter by:

**Port:** alif, cc3200, **esp32**, esp8266, mimxrt, nrf, renesas-ra, rp2, samd, stm32

**Feature:** Audio Codec, BLE, Battery Charging, CAN, Camera, DAC, Display, Dual-core, Environment Sensor, Ethernet, External Flash, External RAM, Feather, IMU, JST-PH, JST-SH, LoRa, Microphone, PoE, RGB LED, SDCard, Secure Element, USB, USB-C, WiFi, microSD, mikroBUS

**Vendor:** Actinius, Adafruit, Alif Semiconductor, Arduino, BBC, Espressif, Espruino, Ezurio, Fez, George Robotics, HydraBus, I-SYST, LEGO, LILYGO, LimiFrog, M5Stack, Machdyne, Makery, McHobby, Microchip, MikroElektronika, MiniFig Boards, NXP, Netduino, Nordic Semiconductor, Olimex, PJRC, Particle, Pimoroni, Pololu, Pycom, Raspberry Pi, Renesas Electronics, ST Microelectronics, Seeed Studio, Silicognition, Silicognition LLC, SparkFun, Unexpected Maker, VCC-GND Studio, Vekatech, WIZnet, WeAct, WeAct Studio, Wemos, Wireless-Tag, nullbits, u-blox

**MCU:** AE722F80F55D5XX, RA6M5, cc3200, esp32, esp32c3, esp32c6, esp32s2, esp32s3, esp8266, mimxrt, nrf51, nrf52, nrf91, ra4m1, ra4w1, ra6m1, ra6m2, ra6m5, rp2040, rp2350, samd21, samd51, stm32f0, stm32f4, stm32f411, stm32f7, stm32g0, stm32g4, stm32h5, stm32h7, stm32l0, stm32l1, stm32l4, stm32wb, stm32wl

图 4.1 ESP32-S3 MicroPython 固件下载 1

接下来往下滑动网页找到“ESP32-S3 Espressif”选项，如下图所示：



图 4.2 ESP32-S3 MicroPython 固件下载 2

点击 ESP32-S3 Espressif 选项进入固件下载页面，可以看到页面有好几种 ESP32 MicroPython 固件，每种固件的说明如下：

**Firmware:** 用于大部分拥有 4MB Flash 的 ESP32-S3 设备

**Firmware (Support for Octal-SPIRAM):** 用于带有 Octal-SPIRAM 的 ESP32-S3 设备  
这里使用内部带有 8MB 的 Octal-SPIRAM 的 ESP32-S3 芯片，所以选择 **Firmware (Support for Octal-SPIRAM)** 固件，点击“V1.25.0(2025-04-15)”版本里的“[.bin]”选项下载 bin 文件，如下图所示：

## Firmware (Support for Octal-SPIRAM)

Releases

v1.25.0 (2025-04-15) .uf2 / [.app-bin] / [.bin] / [.elf] / [.map] / [Release notes] (latest)  
v1.24.1 (2024-11-29) .uf2 / [.app-bin] / [.bin] / [.elf] / [.map] / [Release notes]  
v1.24.0 (2024-10-25) .uf2 / [.app-bin] / [.bin] / [.elf] / [.map] / [Release notes]  
v1.23.0 (2024-06-02) .uf2 / [.app-bin] / [.bin] / [.elf] / [.map] / [Release notes]  
v1.22.2 (2024-02-22) .uf2 / [.app-bin] / [.bin] / [.elf] / [.map] / [Release notes]  
v1.22.1 (2024-01-05) .uf2 / [.app-bin] / [.bin] / [.elf] / [.map] / [Release notes]  
v1.22.0 (2023-12-27) .uf2 / [.app-bin] / [.bin] / [.elf] / [.map] / [Release notes]  
v1.21.0 (2023-10-05) .uf2 / [.app-bin] / [.bin] / [.elf] / [.map] / [Release notes]  
v1.20.0 (2023-04-26) .uf2 / [.bin] / [.elf] / [.map] / [Release notes]

Preview builds

v1.26.0-preview.277.g4bd99260d (2025-06-26) .uf2 / [.app-bin] / [.bin] / [.elf] / [.map]  
v1.26.0-preview.266.g6fee099ca (2025-06-25) .uf2 / [.app-bin] / [.bin] / [.elf] / [.map]  
v1.26.0-preview.265.ge57aa7e70 (2025-06-23) .uf2 / [.app-bin] / [.bin] / [.elf] / [.map]  
v1.26.0-preview.257.gc16a4db15 (2025-06-19) .uf2 / [.app-bin] / [.bin] / [.elf] / [.map]  
(These are automatic builds of the development branch for the next release)

图 4.3 ESP32-S3 MicroPython 固件下载 3

自行选择固件存放目录并下载，如下图所示：

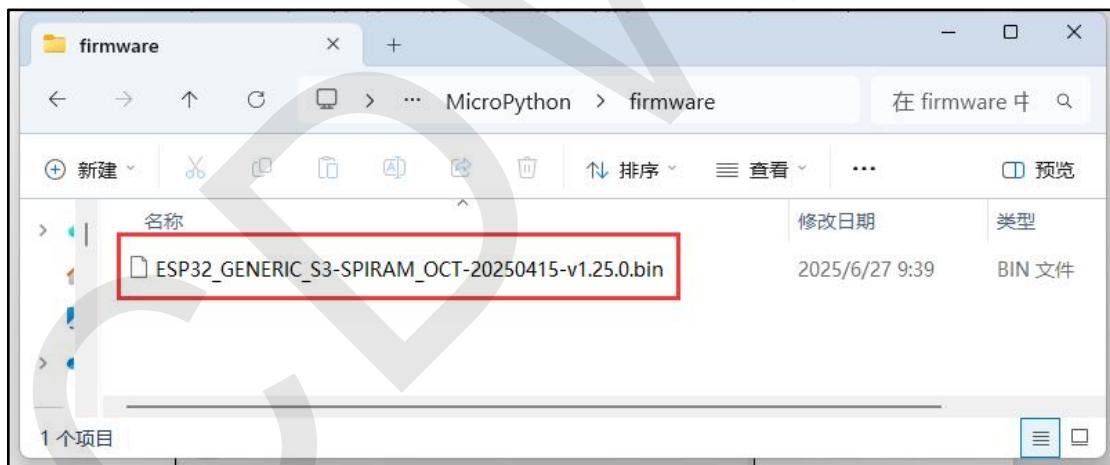


图 4.4 固件存下载完成

## 4.2. 烧录 ESP32-S3 MicroPython 固件

ESP32-S3 MicroPython 固件下载完成后，接下来进行烧录。

**注意：烧录之前先得将 ESP32-S3 设备连接到电脑的 USB 口且 ESP32-S3 设备的串口不能被任何程序占用，否则会烧录失败。**

烧录方法有两种：使用 Thonny 软件烧录、使用 flash\_download\_tool 工具烧录。

下面详细介绍这两种方法：

#### 4. 2. 1. 使用 Thonny 软件烧录固件

首先点击软件菜单栏运行->配置解释器，或者点击菜单栏工具->选项->解释器，或者点击 Thonny 软件界面底部的按钮，选择配置解释器。在解释器界面选择 MicroPython(ESP32) 解释器，端口号选择实际使用的端口号，然后点击“安装或者更新 microPython (esptool)”，如下图所示：

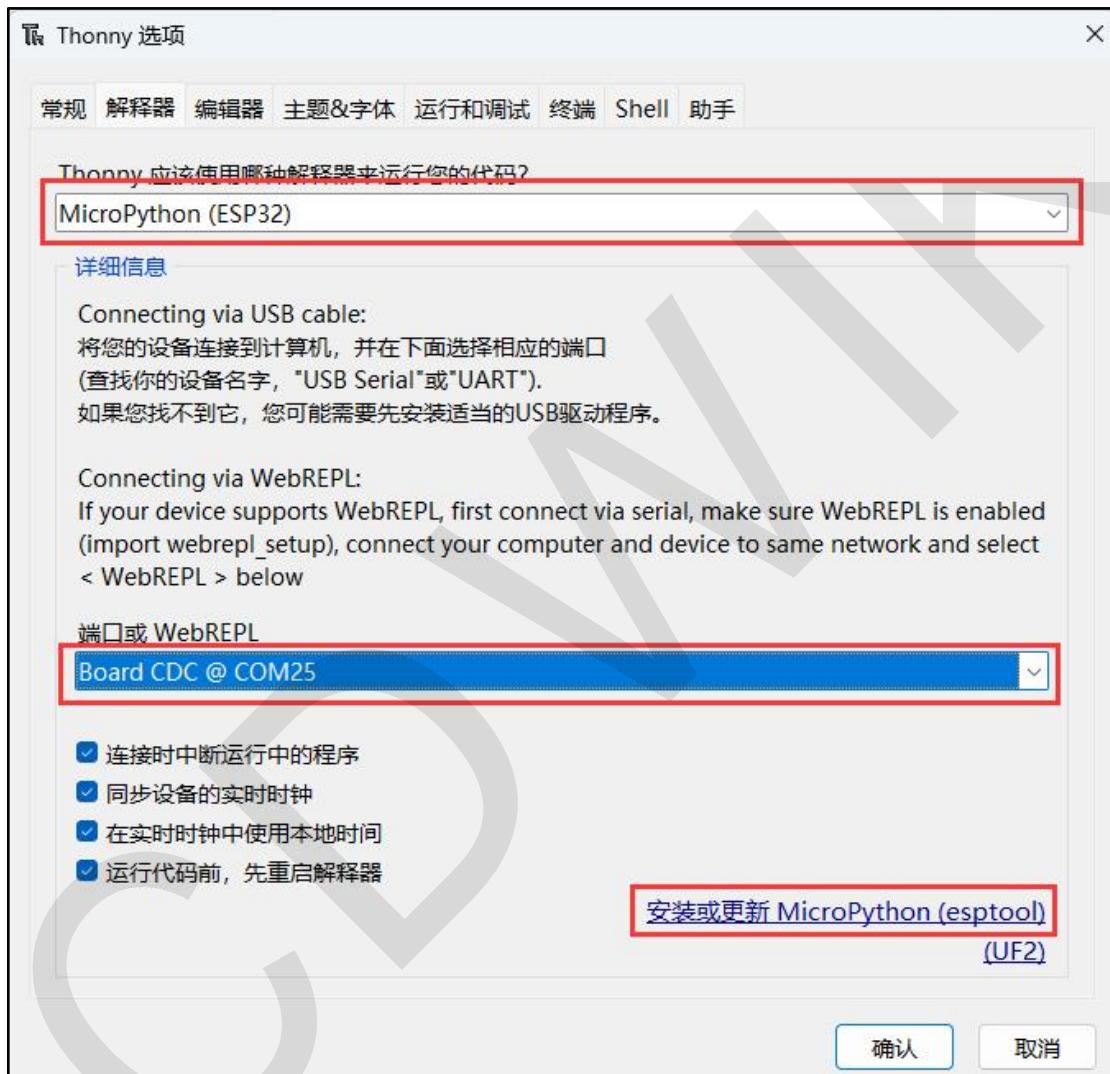


图 4.5 ESP32-S3 MicroPython 固件烧录 1

如果出现找不到 esptool 的错误，则需分别点击工具->管理包... 和工具->管理插件... 安装 esptool。如下图所示：

如果没有出现错误，则忽略该步骤。

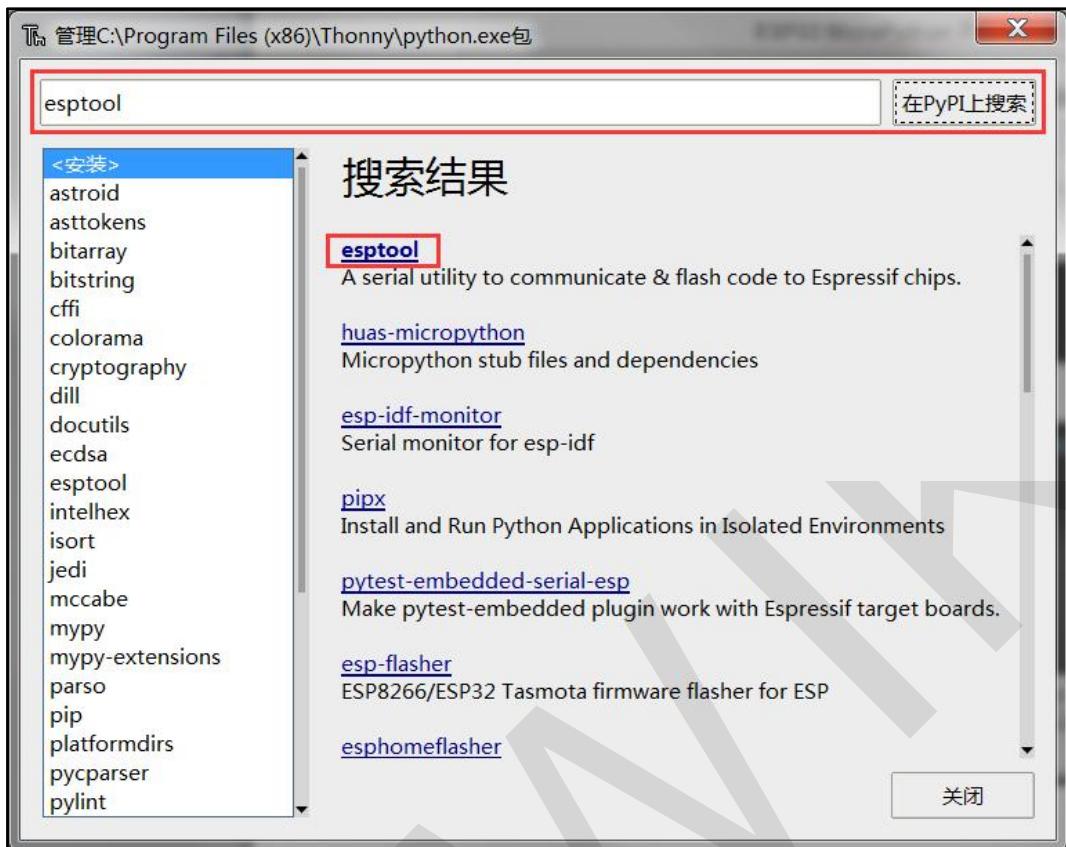


图 4.6 安装 esptool 软件包

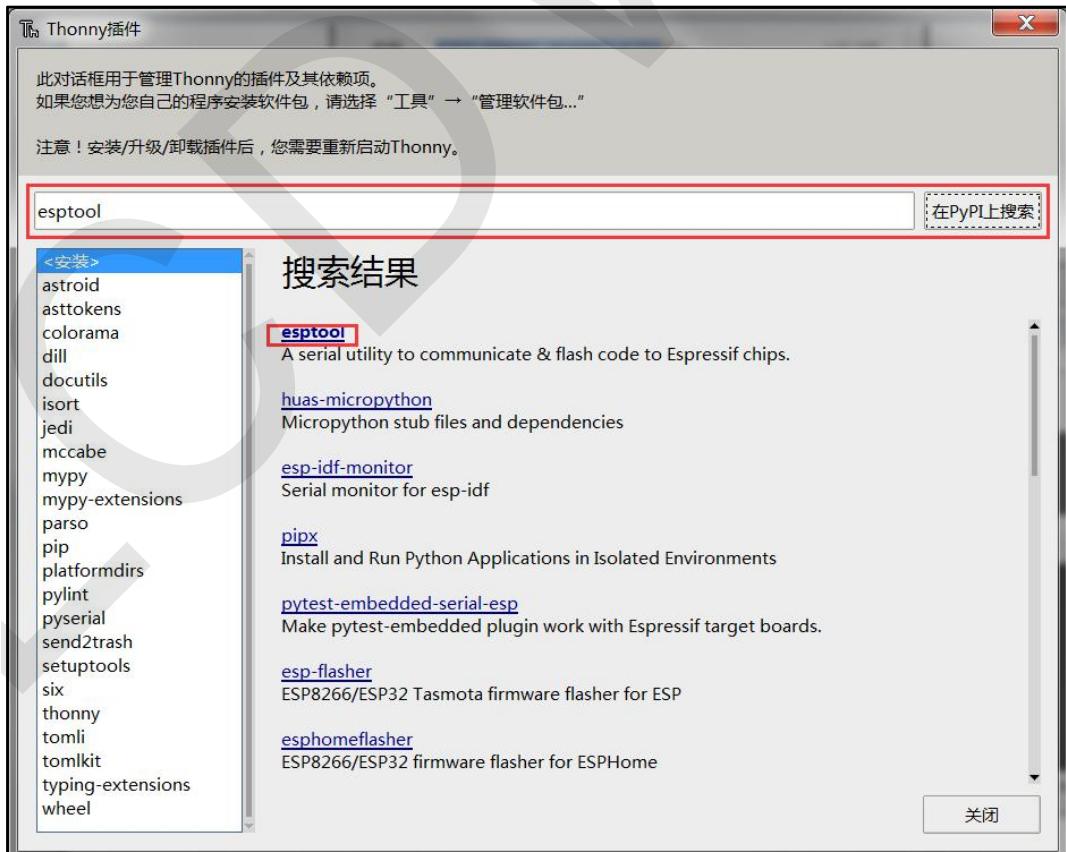


图 4.7 安装 esptool 插件

如果 esptool 安装完成，则重新进入配置解释器界面，然后点击“**安装或者更新 microPython (esptool)**”。

Thonny 软件烧录 ESP32-S3 MicroPython 固件有两种方式：本地烧录和在线烧录。

### A、本地烧录

本地烧录是指将保存在本地电脑的固件文件烧录到 ESP32-S3 设备。

进入烧录界面后，先选择正确的端口，然后点击底部最左边的按钮，在弹出的下拉菜单里点击“**Select local MicroPython image...**”选项，如下图所示：

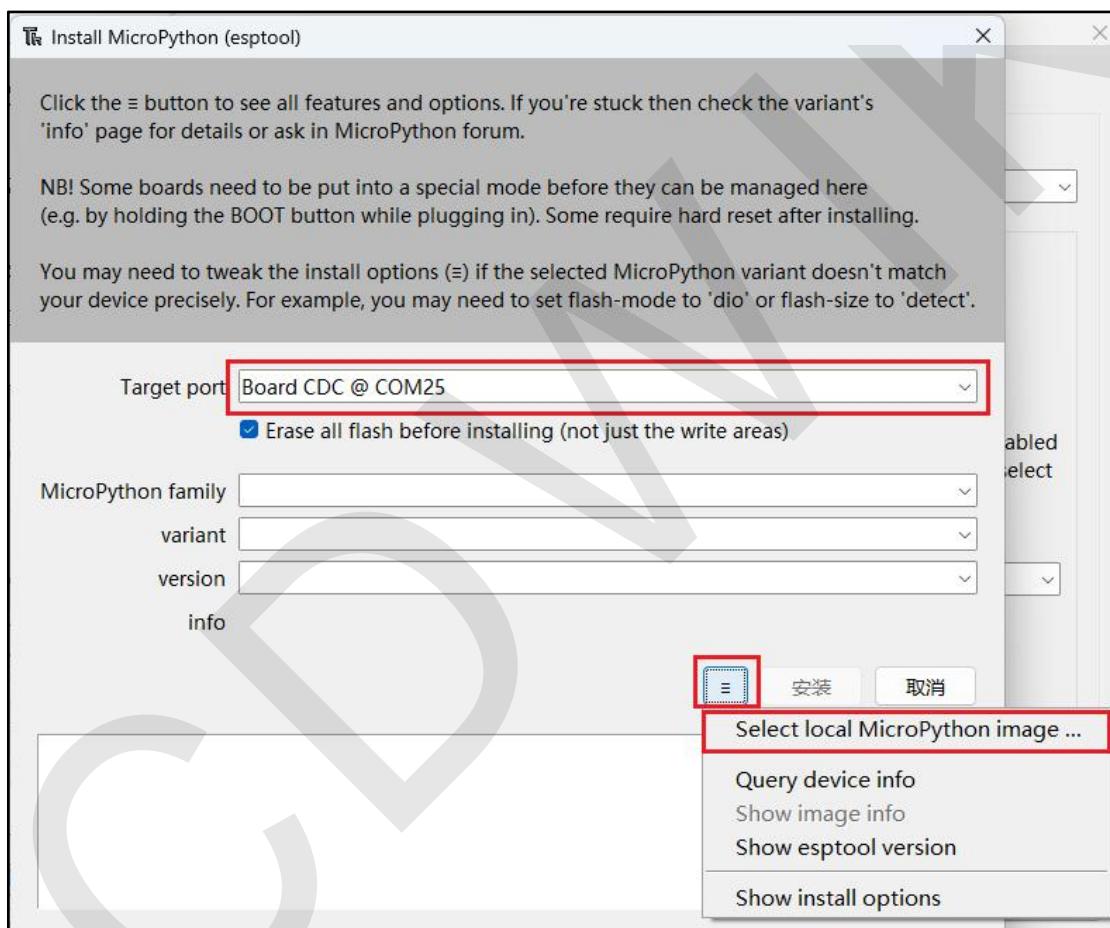


图 4.8 选择本地烧录固件

在弹出的目录里选择需要烧录的本地固件，此时烧录的配置信息会自动填充，如果不进行更加详细的配置，点击“**安装**”就可以烧录固件了。如下图所示：

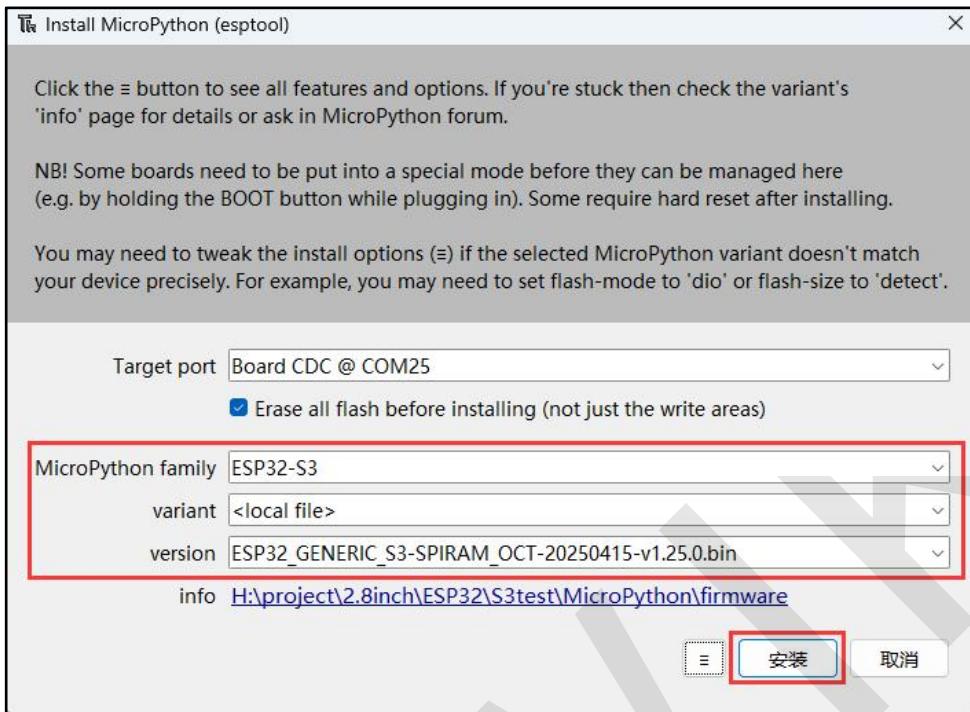


图 4.9 烧录本地固件

如果需要进行更加详细的配置，则先点击底部最左边的按钮，在弹出的下拉菜单里点击“Show install options”选项，如下图所示：

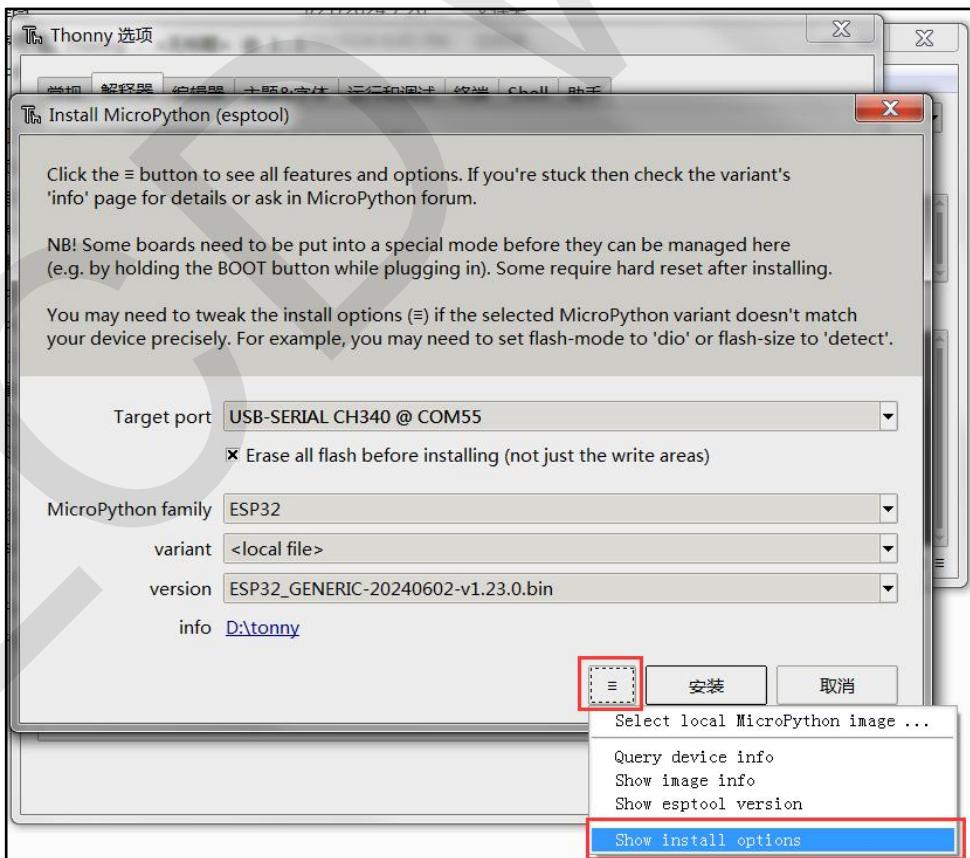


图 4.10 打开烧录配置信息

弹出配置选项后，可对相应的配置选项进行设置，如果不清楚所用的 ESP32-S3 模块是什么配置，那就保持默认配置即可。配置界面如下图所示：

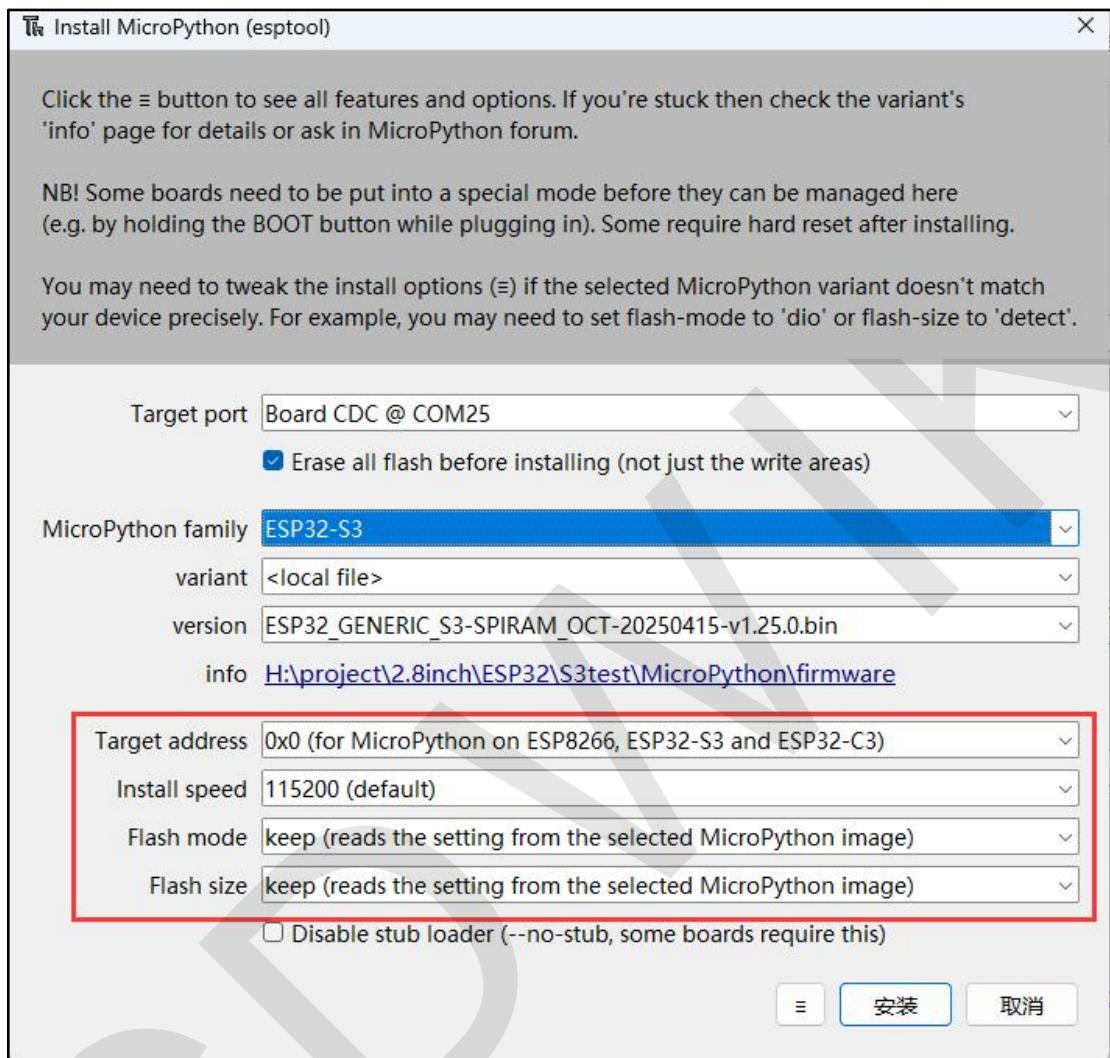


图 4.11 配置 ESP32-S3 模块

这里介绍一下各个配置选项：

**Target address:** 固件烧录地址，有两个选项：0x0 和 0x1000，这里使用 ESP32-S3，所以只能选择 0x0

**Install speed:** 固件烧录的速率，可选参数有：460800、230400、115200、38400、9600。根据 ESP32 模块上 USB 转串口所支持的最大速率选择。

**Flash mode:** ESP32 上挂载的 Flash 通信模式，可选参数有：Keep、DIO、QIO、DOUT、QOUT。其中 keep 是从烧录的固件中读取配置；DIO 和 DOUT 都为双线 SPI 模式，DIO 在地址和数据阶段都使用 2 根数据线通信，而 DOUT 只在数据阶段使用 2 根数据线通信；QIO 和 QOUT 都为四线 SPI 模式，

DIO 在地址和数据阶段都使用 4 根数据线通信，而 DOUT 只在数据阶段使用 4 根数据线通信。根据 ESP32 模块的 Flash 实际连接方式选择。

**Flash size:** ESP32 上挂载的 Flash 容量，可选参数有：keep、detect、256KB、512KB、1MB、2MB、4MB、8MB、16MB、32MB。其中 keep 是从烧录的固件中读取配置，detect 为根据 Flash ID 获取容量。根据 Flash 的实际容量选择。

其他配置项保持默认。

配置选项设置完成后，点击“安装”按钮，接下来进入烧录的进度，如下图所示：

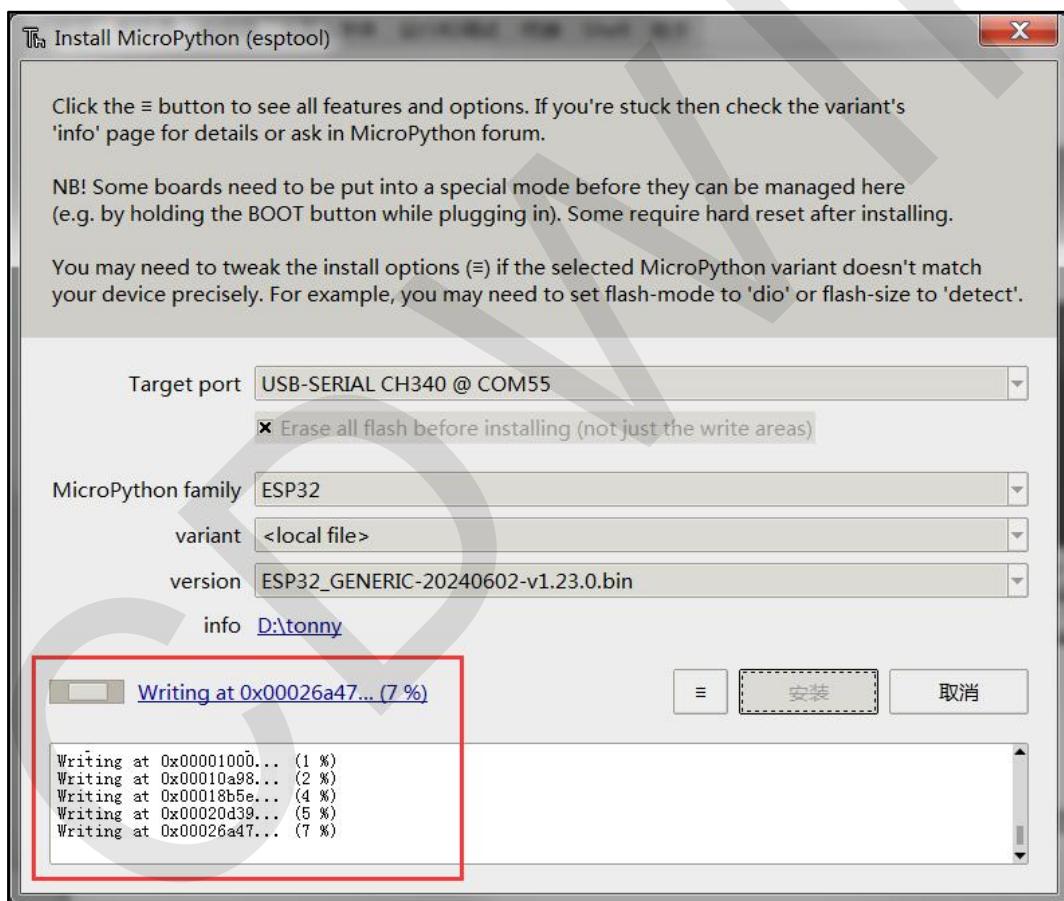


图 4.12 烧录 ESP32-S3 MicroPython 固件

当出现“Done!”提示后，说明烧录已完成，点击“关闭”按钮即可，如下图所示：

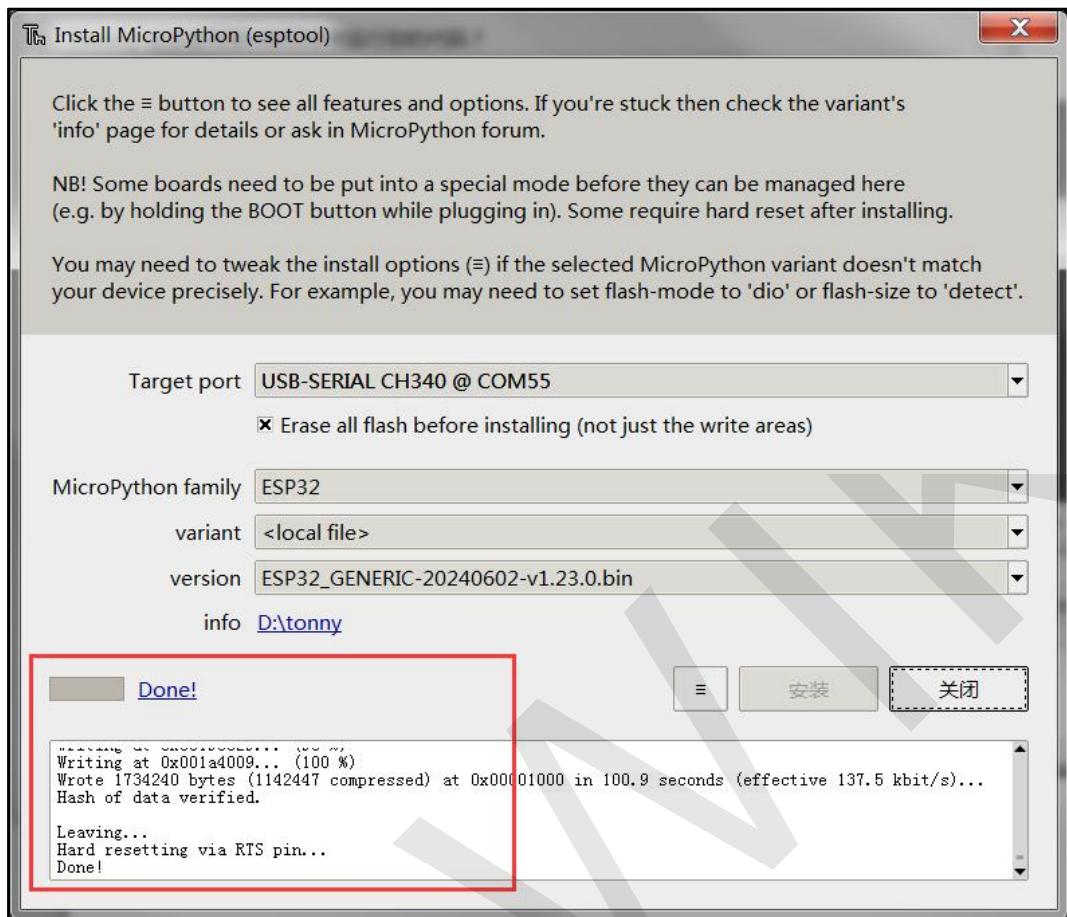


图 4.13 ESP32-S3 MicroPython 固件烧录完成

## B、在线烧录

在线烧录是指通过网络将 MicroPython 官网上最新的 ESP32-S3 MicroPython 固件下载到本地电脑，再进行烧录。此方式需要电脑连接互联网。

进入烧录界面后，先选择正确的端口，再进行 **MicroPython family**、**variant**、**version** 等选项配置，一般只需配置前面两个选项，version 会自动获取。

如果需要进行更加详细的配置，步骤请参考本地烧录方式里的描述。

配置完成后，点击“安装”按钮，如下图所示：

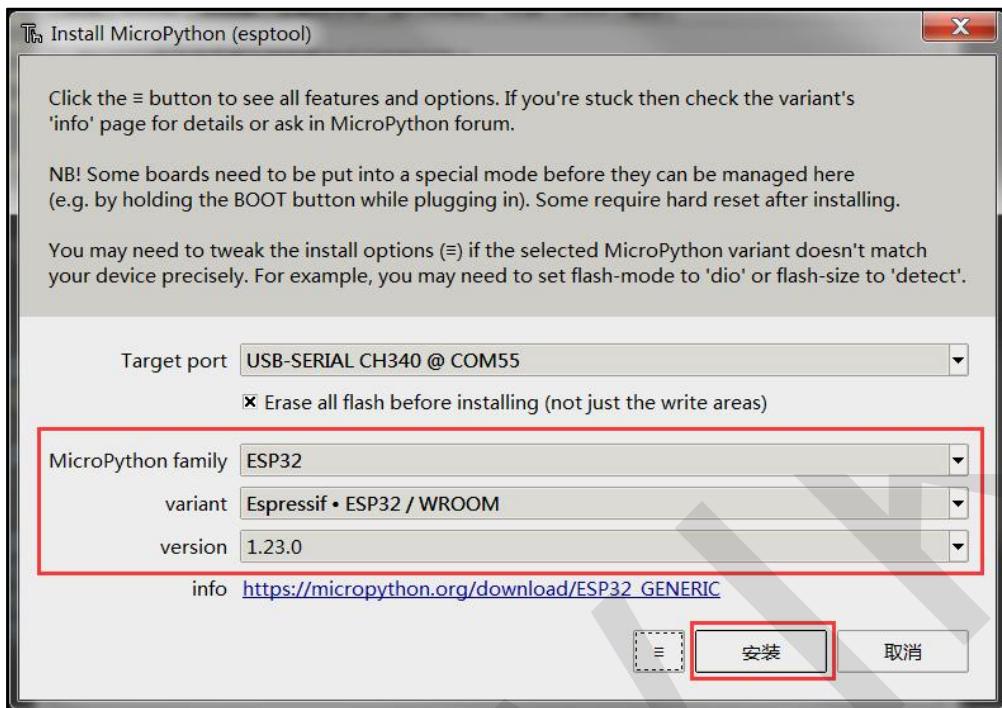


图 4.14 ESP32-S3 MicroPython 固件在线烧录配置

烧录阶段，可以看到烧录的进度，如下图所示：

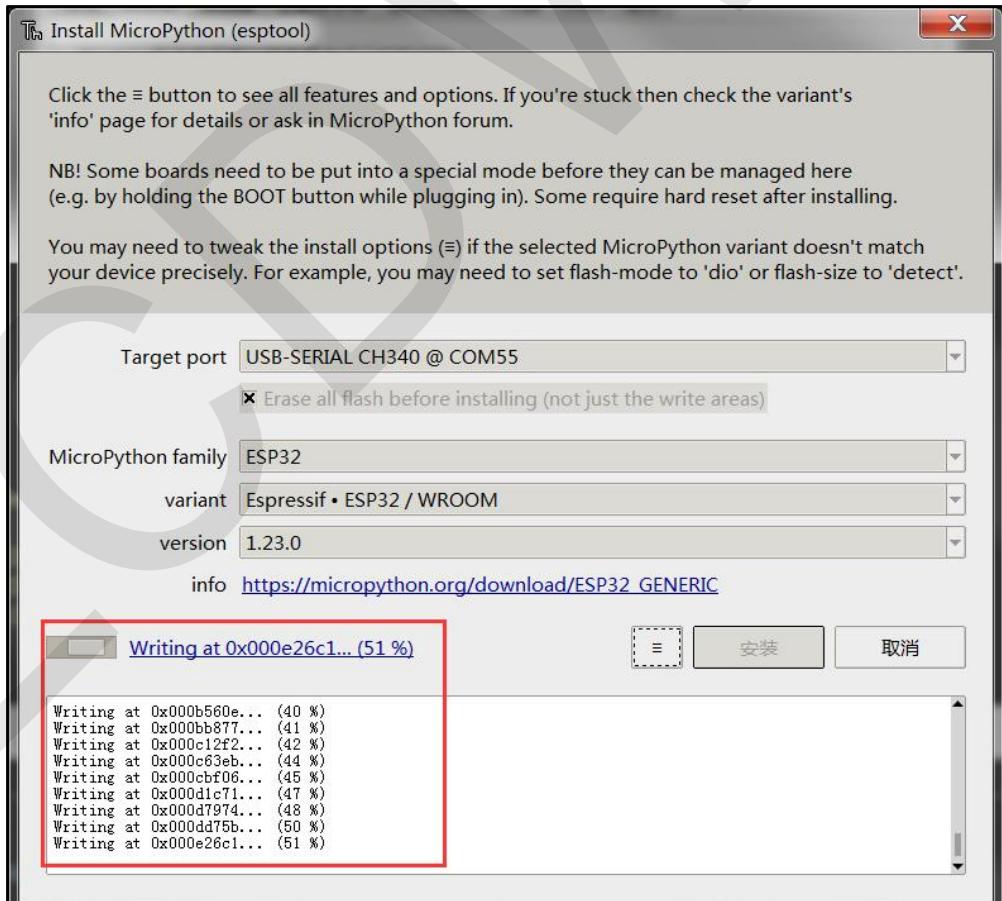


图 4.15 ESP32-S3 MicroPython 固件在线烧录

当出现“Done!”提示后，说明烧录已完成，点击“关闭”按钮即可，如下图所示：

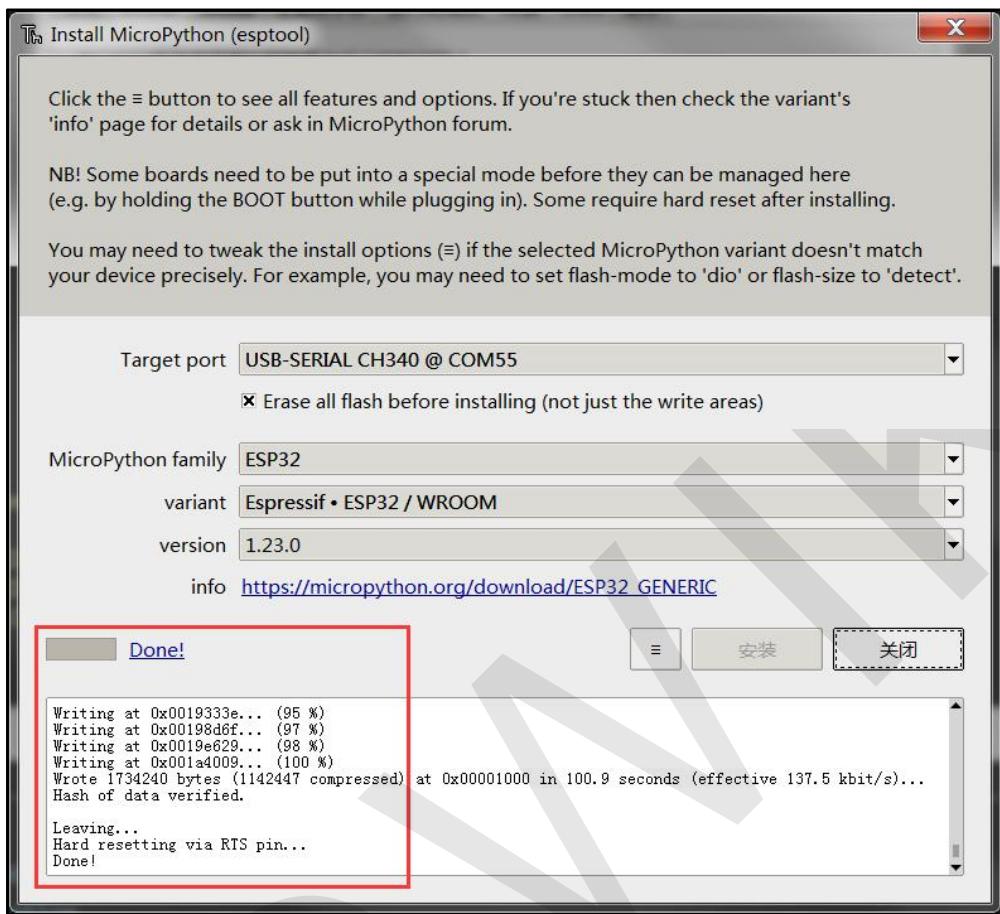


图 4.16 ESP32-S3 MicroPython 固件在线烧录完成

#### 4.3.2. 使用 flash\_download\_tool 工具烧录

flash\_download\_tool 工具可以从官网下载。

官网下载地址：

<https://www.espressif.com.cn/zh-hans/support/download/other-tools>

进入下载网页后，找到 Flash 下载工具栏目，点击下载，如下图所示：



图 4.17 官网下载 flash\_download\_tool 工具

如果不方便网上下载，还可以从资料包的“7-工具软件\_Tool\_software”文件夹获取，在该文件夹下找到“Flash\_Download.zip”压缩包，解压后就可使用。

打开 flash\_download\_tool 文件夹，找到 exe 文件，双击打开，按如下图所示进行配置，然后点击“OK”按钮。

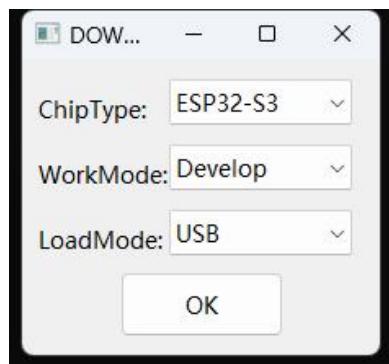


图 4.18 配置 flash\_download\_tool 工具

进入 flash\_download\_tool 工具界面后，按如下步骤进行设置：

- A、选择本地电脑保存的烧录固件，设置烧录地址，ESP32-S3 烧录地址必须设为 0x0，否则烧录后，ESP32-S3 无法正常工作。
- B、设置 SPI Flash 的 SPI 模式，根据实际情况设置，这里设为 DIO。
- C、设置 SPI Flash 的 SPI 速度，根据实际情况设置，这里设为 80MHz。
- D、设置 COM 口和 BAUD 率，根据实际情况设置。
- E、点击“START”按钮，开始启动烧录。

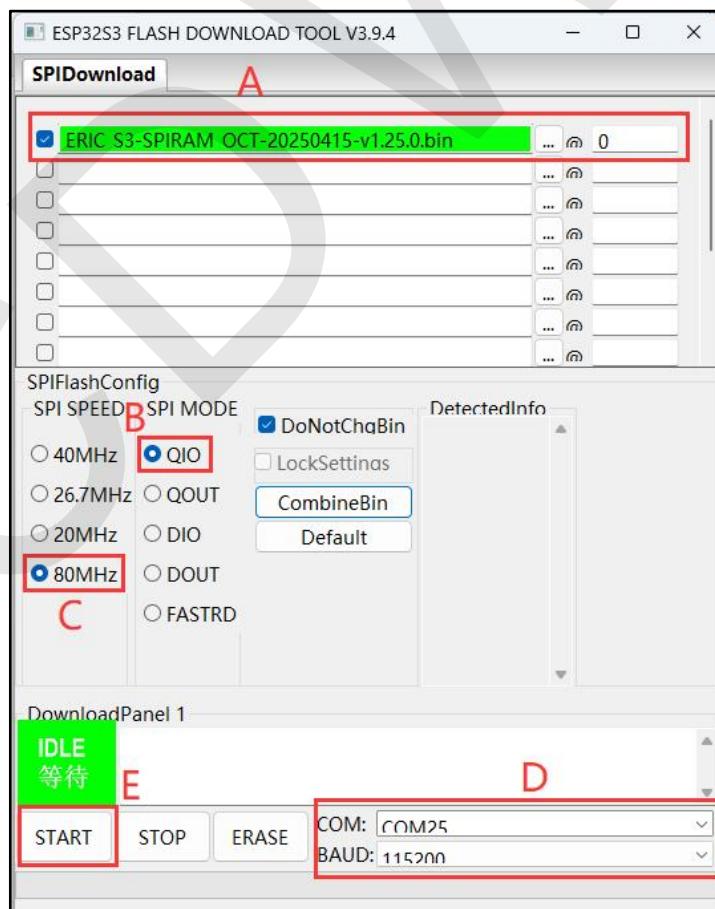


图 4.19 使用 flash\_download\_tool 工具烧录固件

启动烧录后，可以看到烧录提示和进度条变化，待进度条走完且出现“完成”的提示，则表示烧录完成，如下图所示：

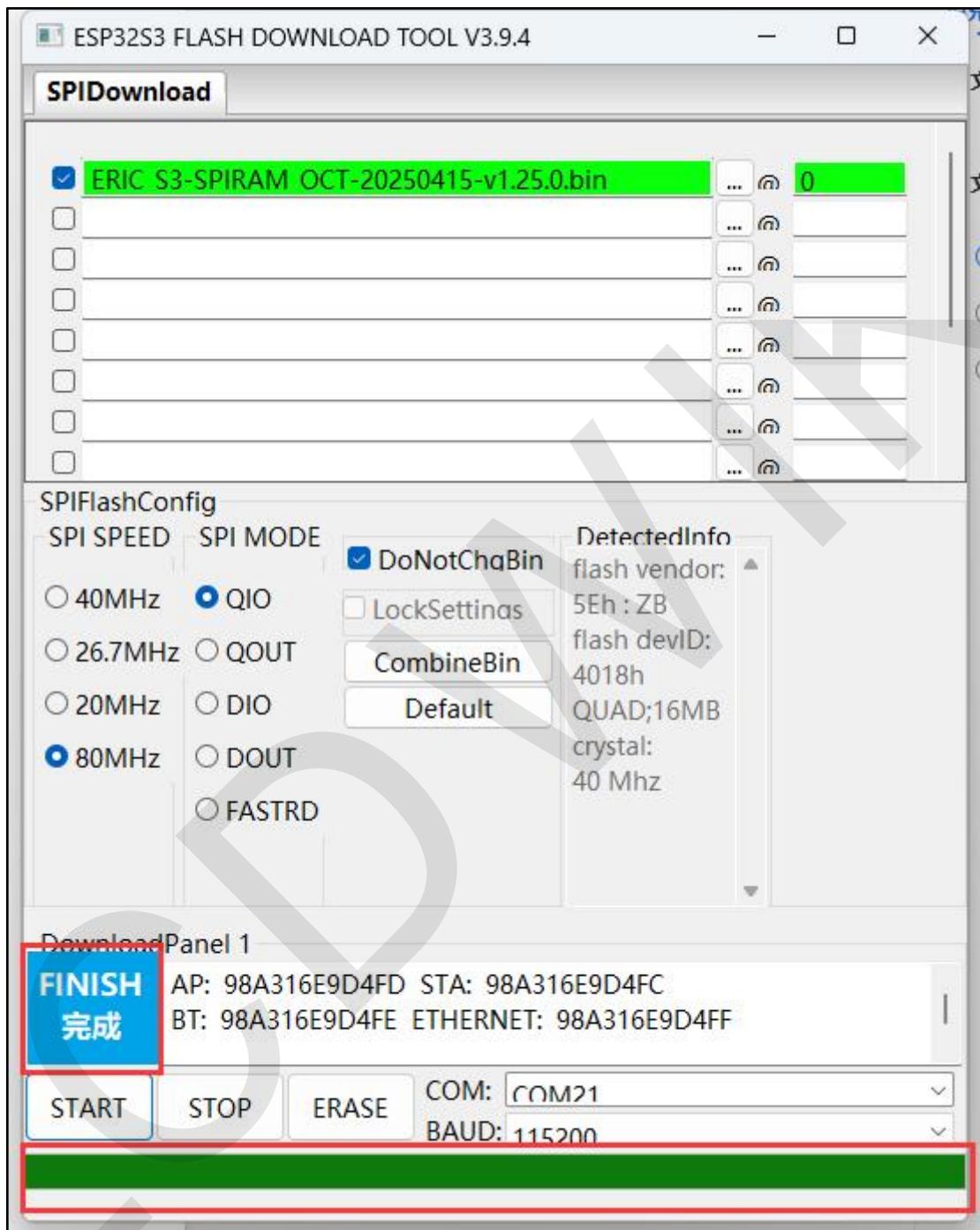


图 4.20 flash\_download\_tool 工具烧录固件完成

## 5. 编辑、保存并运行 ESP32-S3 MicroPython 程序

ESP32-S3 的 MicroPython 固件烧录成功后，接下来就可以进行 MicroPython 程序开发了。首选得将 ESP32-S3 模块连接到电脑的 USB 口上电。

### 5.1. 配置 MicroPython 解释器

点击软件菜单栏运行->配置解释器，或者点击菜单栏工具->选项->解释器，或者点击 Thonny 软件界面底部的按钮，选择配置解释器。在解释器界面选择 MicroPython(ESP32) 解释器，端口号选择实际使用的端口号，然后点击“确认”按钮保存退出。如下图所示：



图 5.1 配置 MicroPython 解释器

### 5.2. 编辑 ESP32-S3 MicroPython 程序

这里介绍的是新建一个程序文件并编辑，如果程序文件已经存在，那么可以直接打开存在的文件进行编辑并使用。

点击文件->新建按钮，或者点击工具栏上 图标，或者按“Ctrl+N”快捷键，来新建一个程序文件，如下图所示：

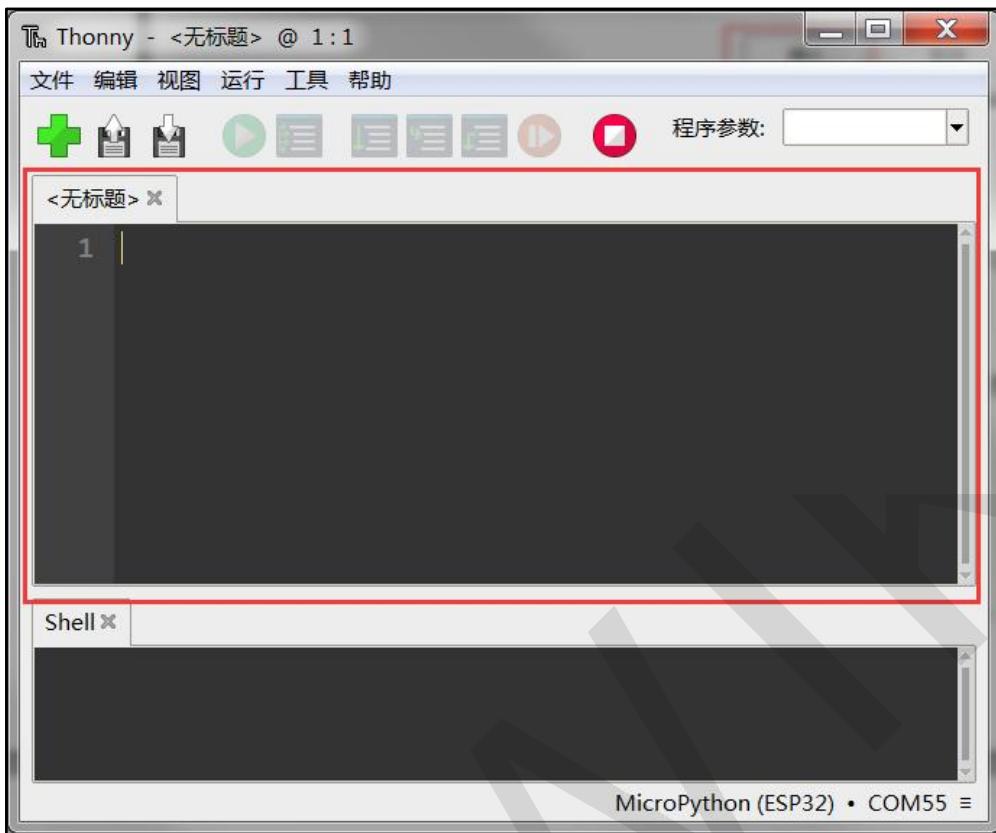


图 5.2 新建程序文件

在程序编辑窗口输入如下内容：

```
import machine
import esp

esp32_id = machine.unique_id()

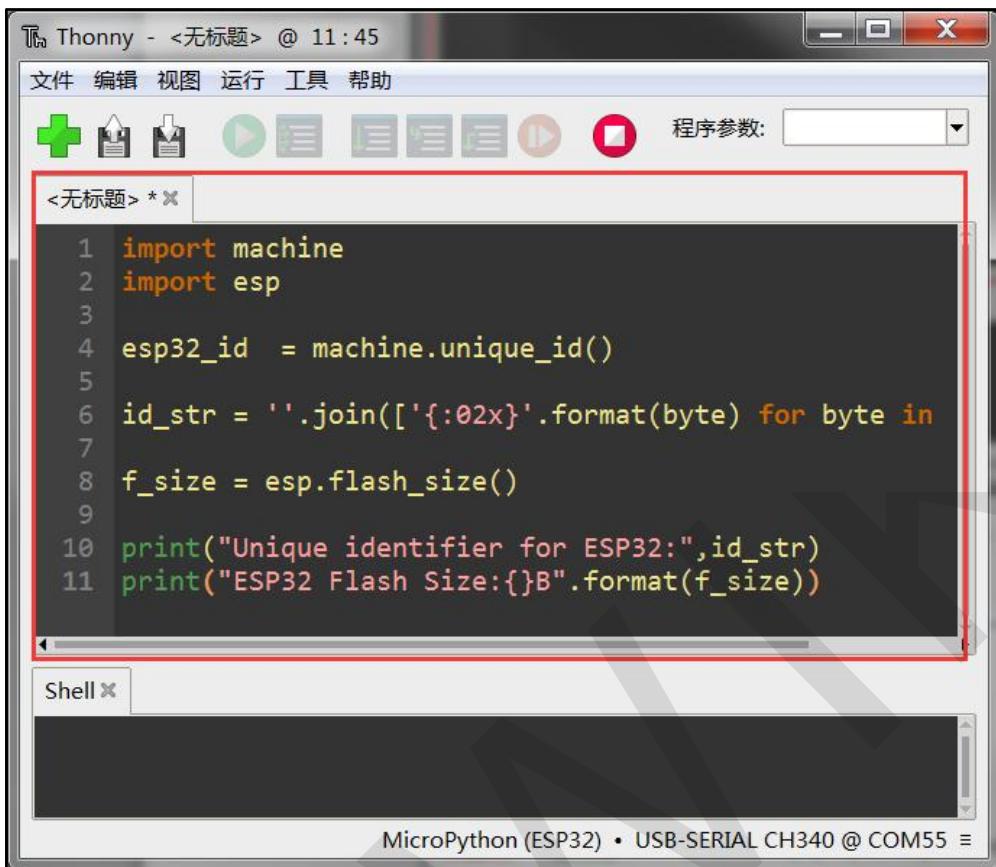
id_str = ''.join(['{:02x}'.format(byte) for byte in esp32_id])

f_size = esp.flash_size()

print("Unique identifier for ESP32:", id_str)
print("ESP32 Flash Size:{}B".format(f_size))
```

图 5.3 输入程序内容

输入程序的编辑窗口，如下图所示：



```

1 import machine
2 import esp
3
4 esp32_id = machine.unique_id()
5
6 id_str = ''.join(['{:02x}'.format(byte) for byte in
7
8     f_size = esp.flash_size()
9
10 print("Unique identifier for ESP32:",id_str)
11 print("ESP32 Flash Size:{}B".format(f_size))

```

图 5.4 程序编辑窗口内容

### 5.3. 保存并运行 ESP32-S3 MicroPython 程序

程序编辑完成后，接下来需运行程序，以便检查程序是否有错误和异常。如果工具  
->选项...->运行和调试菜单下“允许运行未命名的程序”选项被选择，则可直接运行  
编辑好的程序，进行临时检查，提高开发效率，否则需保存后才能运行。如下图所示：

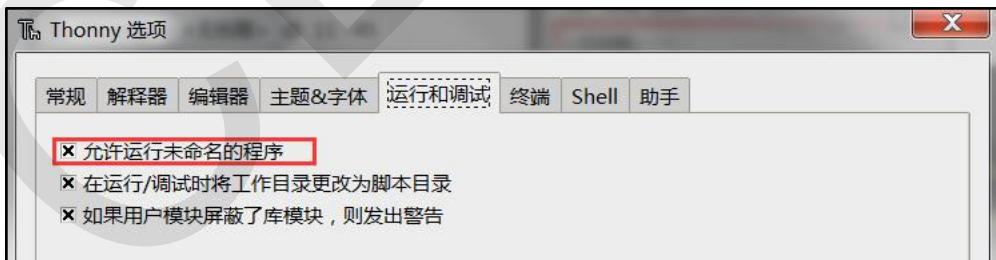


图 5.5 程序运行配置

注意：如果工具栏里的运行按钮  为灰色（不能点击），则需点击停止/重启按钮  启动后端进程。

ESP32-S3 MicroPython 程序编辑完成后，需要保存，可以保存在本地电脑，也可以保存在 ESP32-S3 模块里。

### A、保存在本地电脑

如果 Thonny 没有连接 ESP32-S3 模块，则只能保存在本地电脑。点击文件->保存，

或者点击工具栏保存按钮 ，或者按“Ctrl+S”快捷键，然后输入文件名称，再选择选择目标文件夹保存。如果 Thonny 连接了 ESP32-S3 模块，进行了上述操作后，还会弹出保存位置选择界面，选择“此电脑”即可，如下图所示：



图 5.6 选择保存位置 1

### B、保存在 ESP32-S3 模块

ESP32-S3 模块必须和 Thonny 软件连接。保存分两种情况：一种是新建文件保存，一种是从本地电脑上传。

新建文件保存时，点击文件->保存，或者点击工具栏保存按钮 ，或者按“Ctrl+S”快捷键，在弹出的保存位置选择界面，选择“MicroPython 设备”，然后输入文件名称，点击“确认”按钮保存。如下图所示：



图 5.7 选择保存位置 2

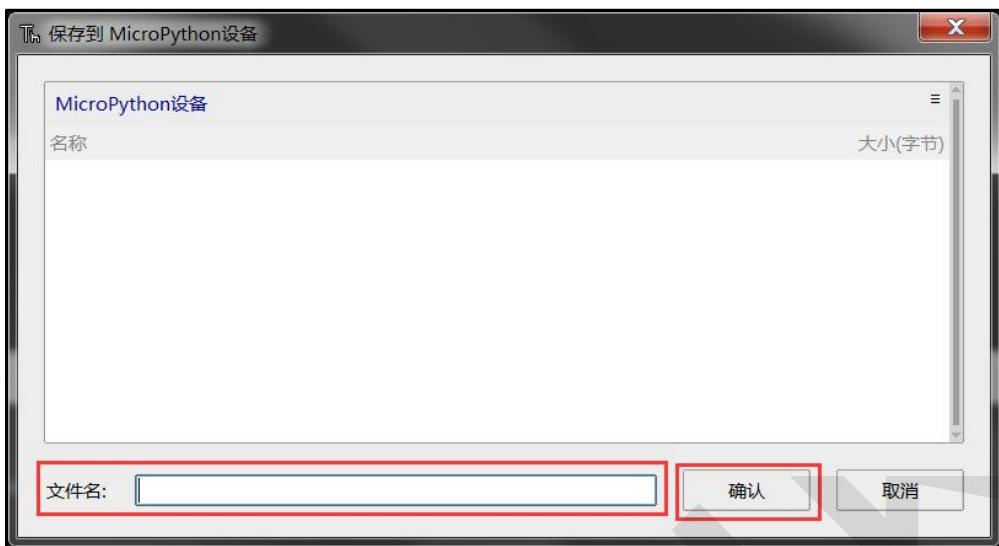


图 5.8 保存新建文件到 MicroPython 设备

从本地电脑上传时，先点击菜单栏里视图->文件，打开本地文件浏览窗口，然后找到需要上传的文件，选择文件单击鼠标右键，在弹出的菜单里选择“上传到/”选项，此时进入上传文件阶段。

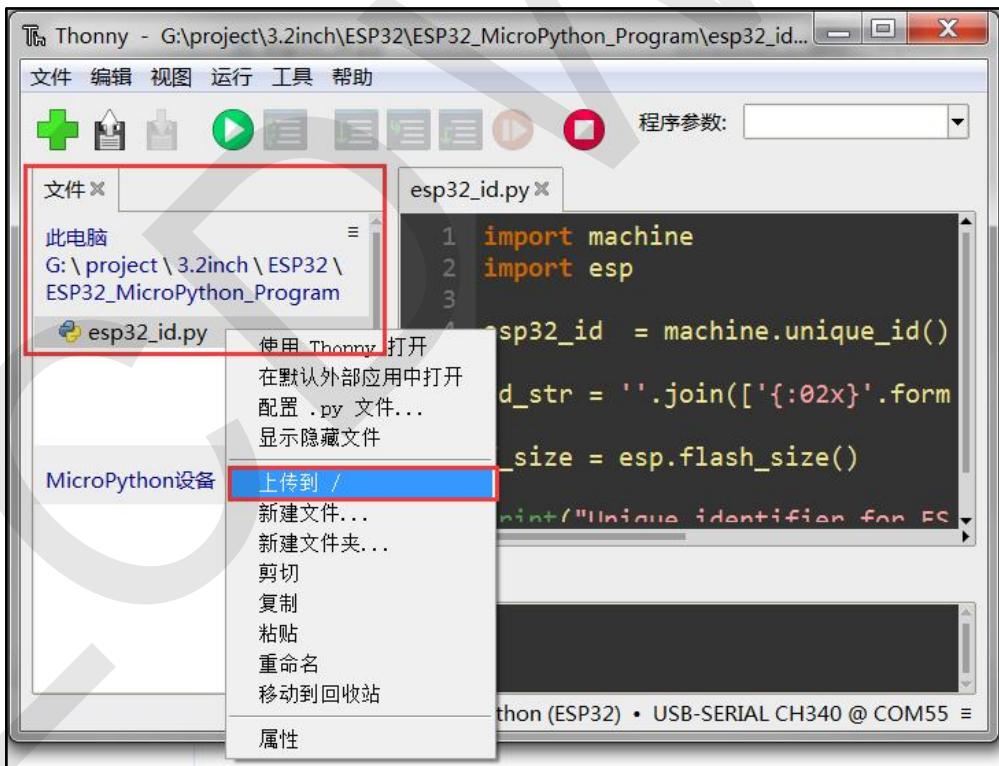


图 5.9 保存本地文件到 MicroPython 设备

上传完毕后，可以看到 MicroPython 设备栏里出现文件名称，如下图所示：

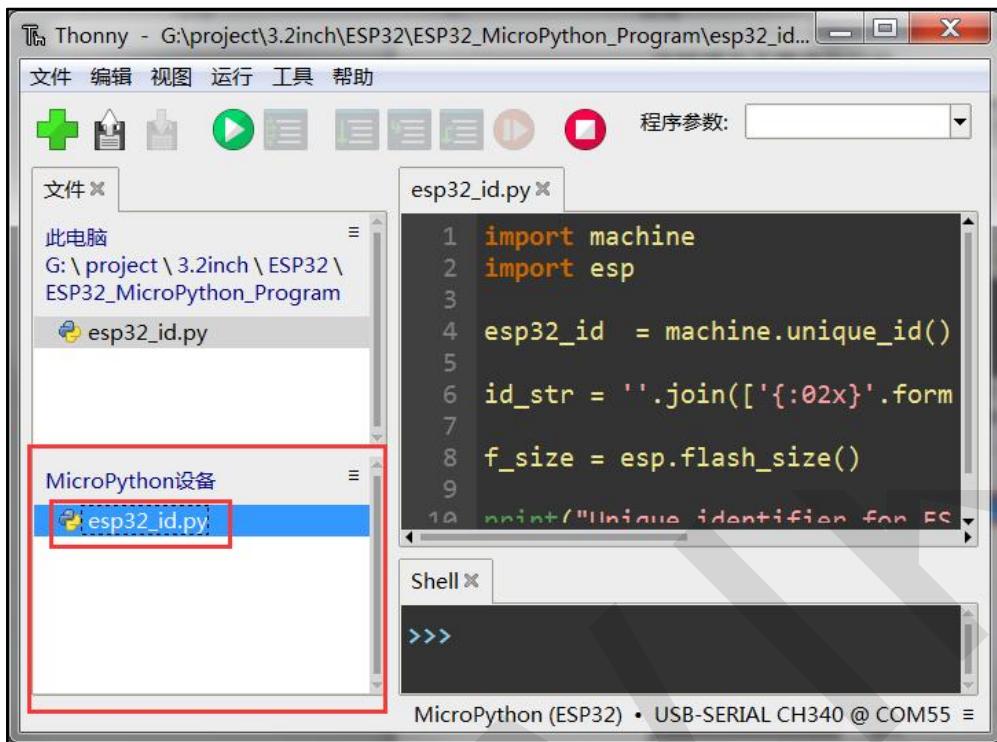


图 5.10 本地文件已保存到 MicroPython 设备

文件保存完毕后，接下来运行文件。首先打开文件（打开本地电脑文件或者 MicroPython 设备里文件都可以），然后点击运行→运行当前脚本，或者点击工具栏里运行按钮 ，或者按“F5”快捷键，运行程序文件，在 Shell 窗口可以看到运行结果，如下图所示：

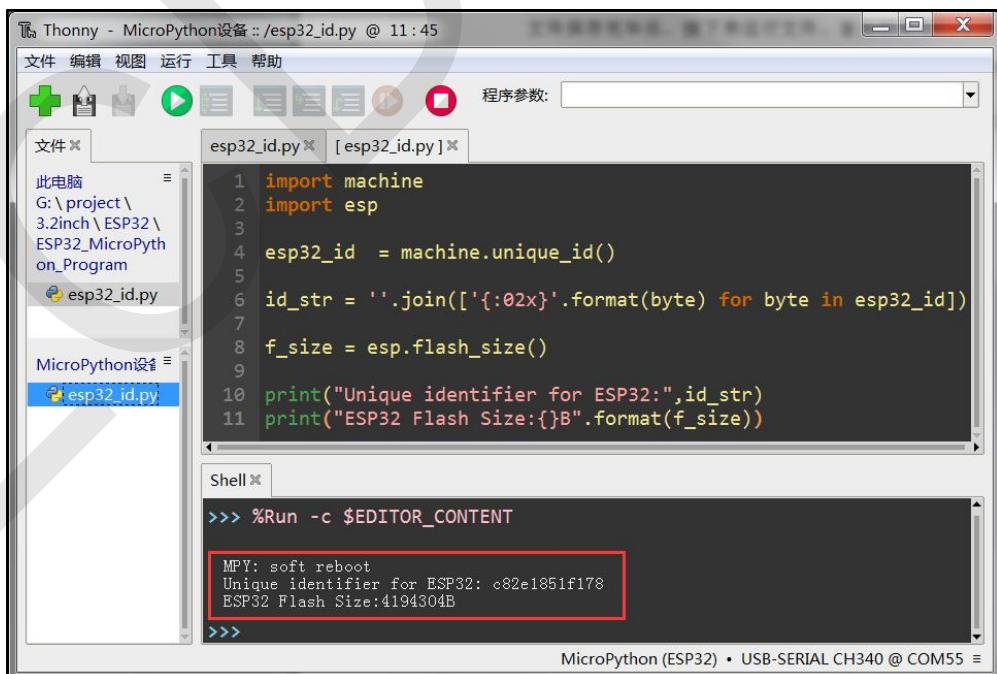


图 5.11 运行程序文件

**注意：**以上操作运行程序文件时，是将临时调试运行，需要用到 Thonny 软件。如果想要 ESP32-S3 设备上电直接运行程序，需要将目标程序文件重命名为“**main.py**”，然后保存到 ESP32-S3 设备里。

