

BPI-UNO32 Programing by PlatformIO

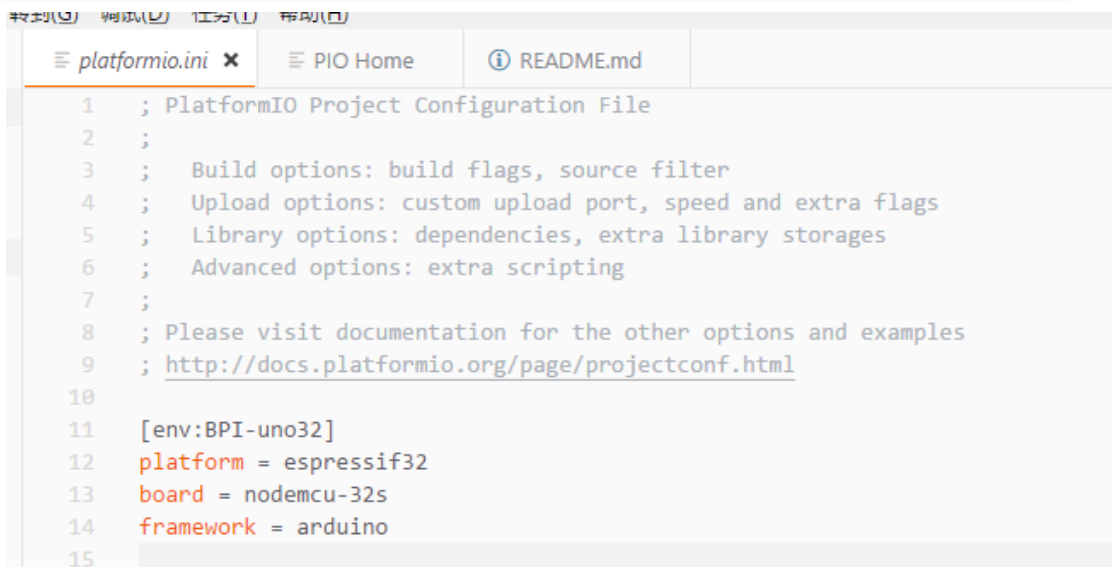
1. First, open the PlatformIO Home page, then click Open Project to open the project. (You can choose the test code.) If you write your own code to burn, then the platformio.ini file should write the following code

```
[env:BPI-uno32]
```

```
platform = espressif32
```

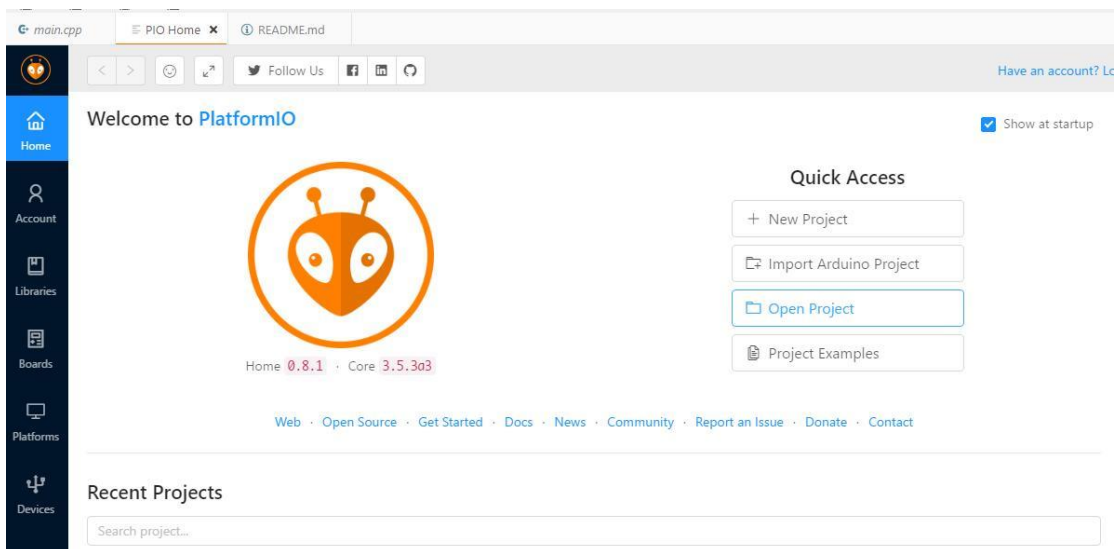
```
board = nodemcu-32s
```

```
framework = arduino
```

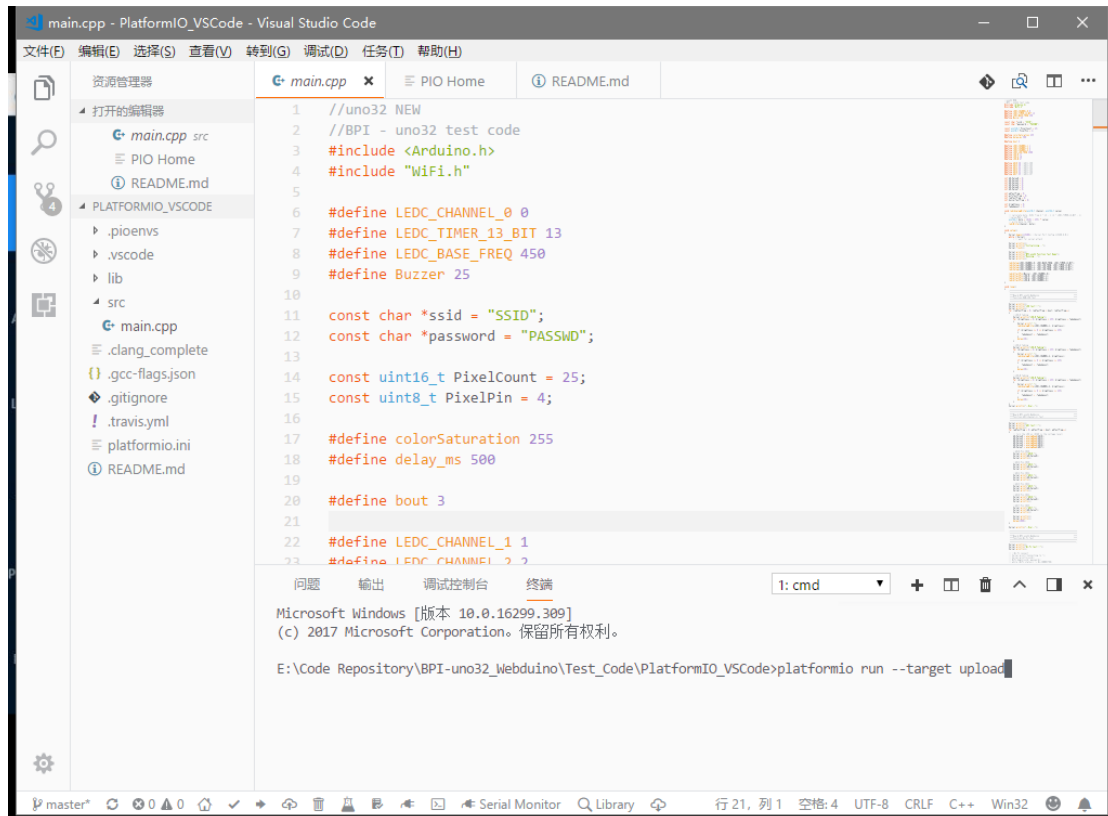


The screenshot shows the PlatformIO IDE interface. The top bar includes tabs for 'platformio.ini', 'PIO Home', and 'README.md'. The 'platformio.ini' tab is active, displaying a configuration file with the following content:

```
1 ; PlatformIO Project Configuration File
2 ;
3 ; Build options: build flags, source filter
4 ; Upload options: custom upload port, speed and extra flags
5 ; Library options: dependencies, extra library storages
6 ; Advanced options: extra scripting
7 ;
8 ; Please visit documentation for the other options and examples
9 ; http://docs.platformio.org/page/projectconf.html
10
11 [env:BPI-uno32]
12 platform = espressif32
13 board = nodemcu-32s
14 framework = arduino
15
```



- The code path under PlatformIO project is generally src/main.cpp. Open the code and use Ctrl+` to open the terminal debugger. Enter platformio run --target upload.



The screenshot shows the Visual Studio Code interface with the PlatformIO extension. The left sidebar displays the file explorer with the project structure. The main editor window shows the code in main.cpp. The bottom panel contains the terminal window with the command to upload the code.

main.cpp - PlatformIO_VSCode - Visual Studio Code

文件(F) 编辑(E) 选择(S) 查看(V) 转到(G) 调试(D) 任务(T) 帮助(H)

资源管理器

- 打开的编辑器
 - main.cpp src
 - PIO Home
 - README.md
- PLATFORMIO_VSCODE
 - .pioenvs
 - .vscode
 - lib
 - src
 - main.cpp
 - .clang_complete
 - .gcc-flags.json
 - .gitignore
 - .travis.yml
 - platformio.ini
 - README.md

```
1 //uno32 NEW
2 //BPI - uno32 test code
3 #include <Arduino.h>
4 #include "Wifi.h"
5
6 #define LEDC_CHANNEL_0 0
7 #define LEDC_TIMER_13_BIT 13
8 #define LEDC_BASE_FREQ 450
9 #define Buzzer 25
10
11 const char *ssid = "SSID";
12 const char *password = "PASSWD";
13
14 const uint16_t PixelCount = 25;
15 const uint8_t PixelPin = 4;
16
17 #define colorSaturation 255
18 #define delay_ms 500
19
20 #define bout 3
21
22 #define LEDC_CHANNEL_1 1
23 #define LEDC_CHANNEL_2 2
```

问题 输出 调试控制台 终端

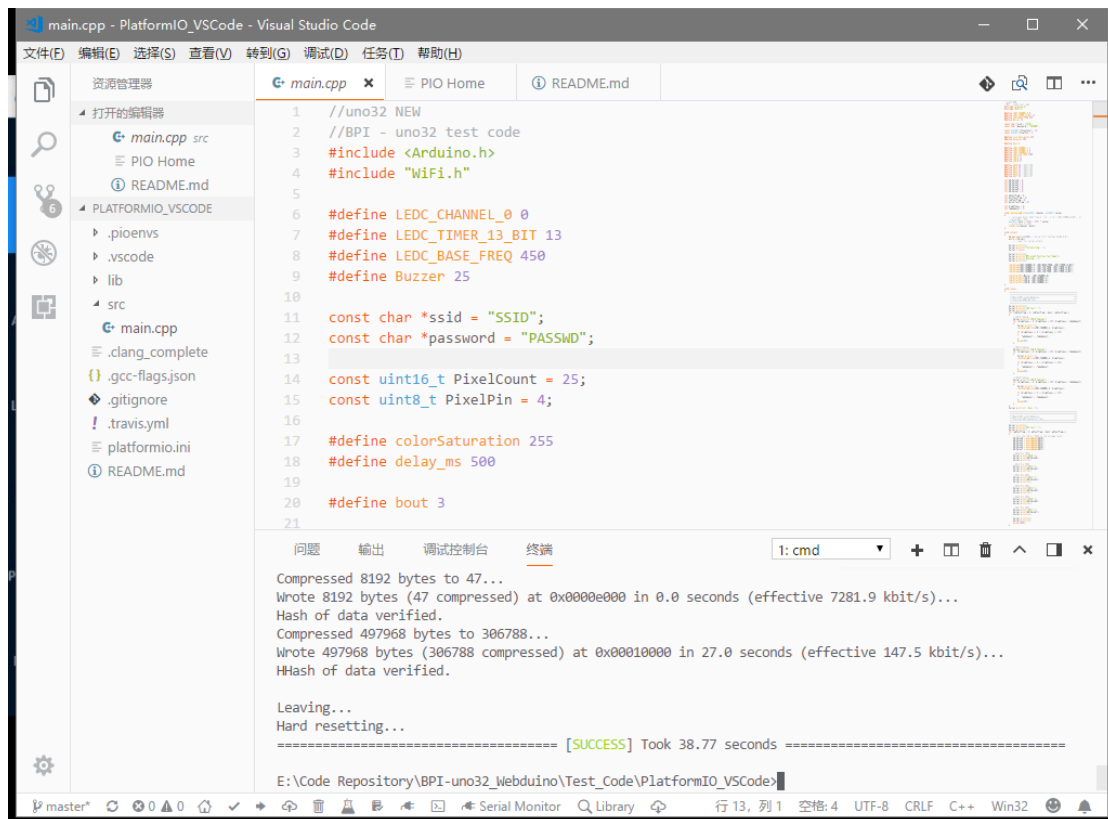
1: cmd

Microsoft Windows [版本 10.0.16299.309]
(c) 2017 Microsoft Corporation。保留所有权利。

E:\Code Repository\BPI-uno32_Webduino\Test_Code\PlatformIO_VSCode>platformio run --target upload

master 行 21, 列 1 空格: 4 UTF-8 CRLF C++ Win32

3. Make sure the board is connected. The program will be compiled first and then burned into the BPI-uno32 board. There are some points to note. If the above code does not specify the port, please try to avoid other serial devices connected on the computer. For serial port numbers, run `platformio --help`.



```
1 //uno32 NEW
2 //BPI - uno32 test code
3 #include <Arduino.h>
4 #include "WiFi.h"
5
6 #define LEDC_CHANNEL_0 0
7 #define LEDC_TIMER_13_BIT 13
8 #define LEDC_BASE_FREQ 450
9 #define Buzzer 25
10
11 const char *ssid = "SSID";
12 const char *password = "PASSWORD";
13
14 const uint16_t PixelCount = 25;
15 const uint8_t PixelPin = 4;
16
17 #define colorSaturation 255
18 #define delay_ms 500
19
20 #define bout 3
21
```

问题 输出 调试控制台 终端 1: cmd

Compressed 8192 bytes to 47...
Wrote 8192 bytes (47 compressed) at 0x0000e000 in 0.0 seconds (effective 7281.9 kbit/s)...
Hash of data verified.
Compressed 497968 bytes to 306788...
Wrote 497968 bytes (306788 compressed) at 0x00010000 in 27.0 seconds (effective 147.5 kbit/s)...
HHash of data verified.

Leaving...
Hard resetting...
===== [SUCCESS] Took 38.77 seconds =====

E:\Code Repository\BPI-uno32_Webduino\Test_Code\PlatformIO_VSCode>