



BPI:bit 轻松上手教程

BPI:bit 轻松上手教程

0. 认识 BPI:bit
 1. 外设一览
 2. 引脚定义
 3. LED阵列编号
1. 有线连接板子
 1. 连接板子
 2. 查看驱动
 3. 安装驱动
 4. 确认串口
2. 获取开发套件
 1. 获取 开发套件
 2. 尝试 烧写板子
 1. 擦除板子固件
 2. 更新板子固件
3. Hello, World!
 1. 使用 MpfShell 编程
 1. 通过 repl 运行程序
 2. 通过 main.py 运行程序
 2. 使用 uPyCraft 编程
 1. 通过 repl 运行程序
 2. 通过 main.py 运行程序

- 3. 使用 PyCharm 编程
- 4. 详解开发环境和工具
 - 1. 什么是 repl?
 - 2. 如何选择开发工具?
- 4. 控制硬件外设
 - 1. 点亮 LED 灯
 - 2. 控制全彩LED阵列螺旋闪烁
- 5. 实现基础算法

 - 1. 输入年份判断闰年
 - 2. 生成斐波那契数列

- 6. 启动无线模式
 - 1. 连接 WIFI 热点
 - 2. 无线使用REPL
 - 3. 无线使用WebDAV
 - 4. 无线编程环境
- 7. 学习网络爬虫
 - 1. 爬取城市天气数据
- 7. 软硬结合应用
 - 1. 自定义点数的骰子
- 8. 专业学习之路
 - 1. 软件学习教程
 - 1. 硬件学习教程

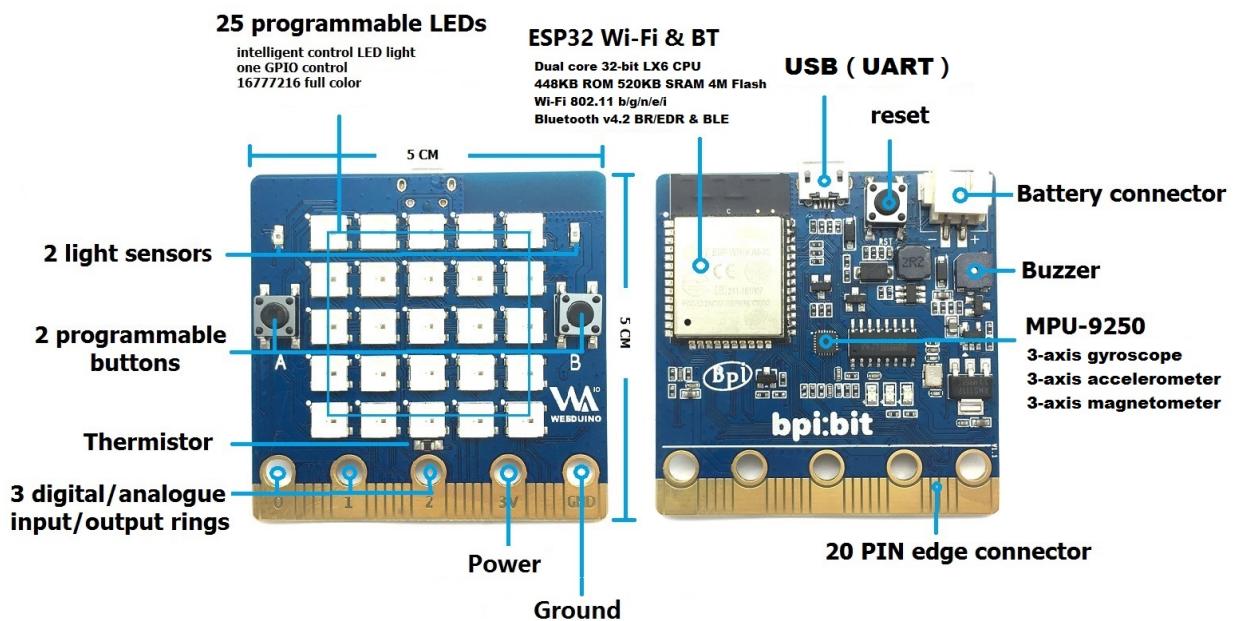
0. 认识 BPI:bit

本产品采用 ESP-WROOM-32 作为核心进行设计，支持蓝牙、BLE等多种通信方式。BPI:bit将兼容Micro:bit。

- ESP-WROOM-32
- 448KB ROM
- 520KB RAM
- Wi-Fi & BLE

正面具有25颗（5 * 5）可编程 LED 灯，单GPIO控制，单颗 16777216 色显示；并且正面还具有 2 个用户可编程按键。

1. 外设一览



2. 引脚定义

Pin Name	Analog Function1	Analog Function2	Function1	Function2	Power
P3	ADC2_CH4		GPIO13		
P0	ADC2_CH8	DAC_1	GPIO25		
P4	ADC2_CH3		GPIO16		
P5	ADC1_CH7		GPIO35		
P6	ADC2_CH5		GPIO12		
P7	ADC2_CH6		GPIO14		
P1	ADC1_CH4		GPIO32		
P8			GPIO16		
P9			GPIO17		
P10	ADC2_CH9	DAC_2	GPIO26		
P11	ADC2_CH7		GPIO27		
P12	ADC2_CH2		GPIO02		
P2	ADC1_CH5		GPIO33		
P13			GPIO18	SPI_SCK	
P14			GPIO19	SPI_MISO	
P15			GPIO23	SPI_MOSI	
P16			GPIO05	SPI_SS	
3V3					POWER:3V3
3V3					POWER:3V3
3V3					POWER:3V3
P19			GPIO22	I2C_SCL	
P20			GPIO21	I2C_SDA	
GND					GROUND
GND					GROUND
GND					GROUND

3. LED阵列编号

BPI:bit 板按照 5 * 5 排列方式焊接了25颗(编号0~24)1600万色全彩LED(WS2812)，所有的LED的控制仅使用一个引脚(GPIO4)即可完成全部控制，同时BPI:bit板还预留LED电源控制引脚(GPIO2)，可以通过拉低GPIO2的点位来关闭全部LED的电源。

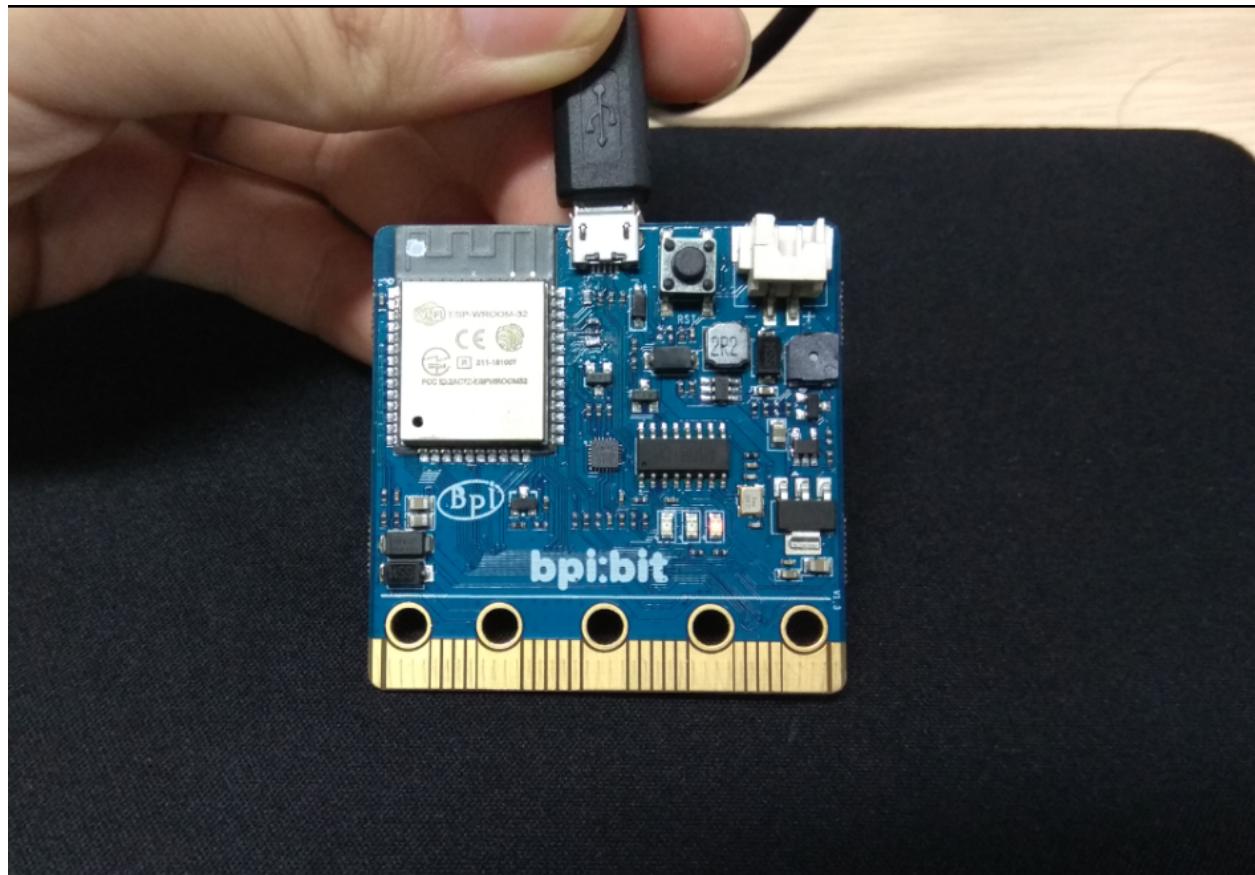
LED序号排布方式如下(5 * 5):

20	15	10	5	0
21	16	11	6	1
22	17	12	7	2
23	18	13	8	3
24	19	14	9	4

1. 有线连接板子

1. 连接板子

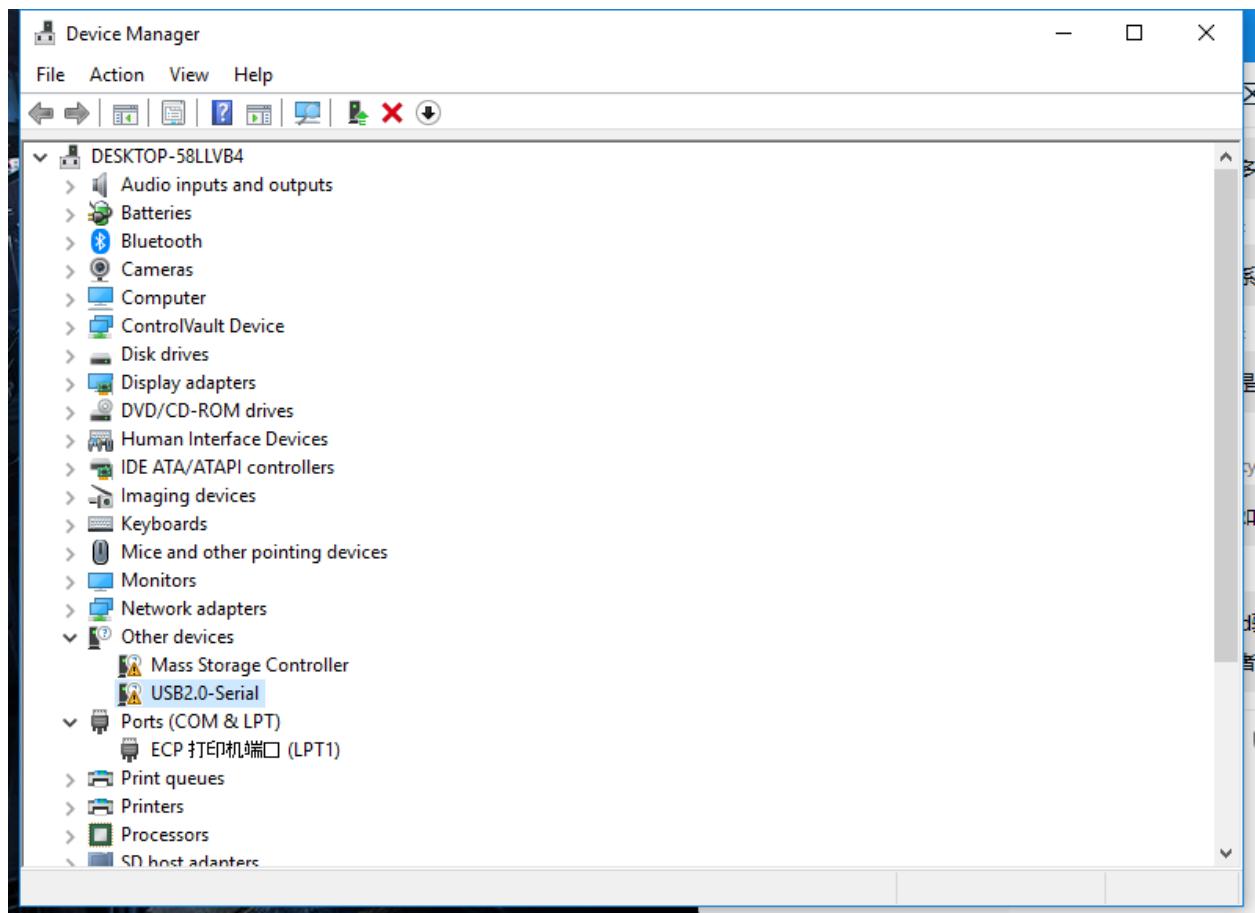
- 将板子通过 MicroUSB 线连接到你的电脑里，以下以 Windows10 为例。



2. 查看驱动

- 进入设备管理器 确认串口驱动（Serial）是否安装，进入方法如下。

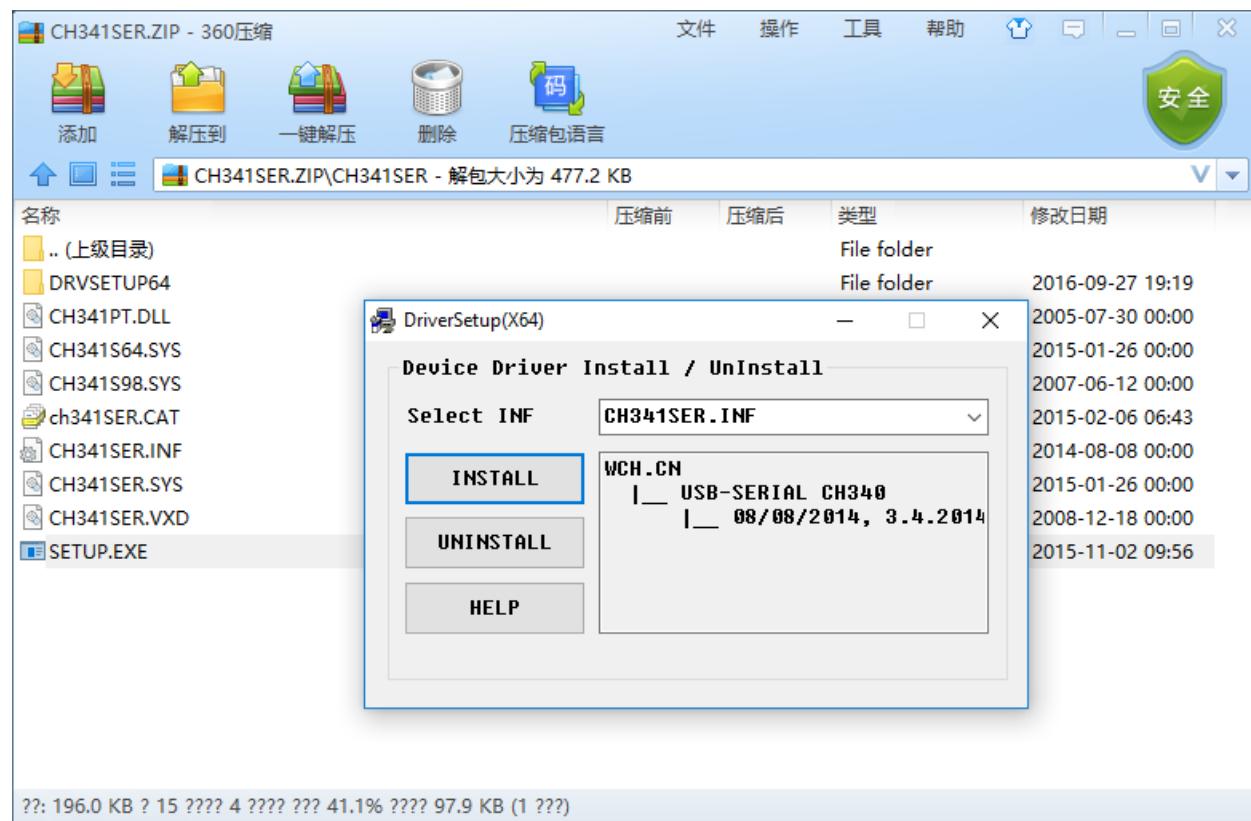
- (右键) 此电脑 -> 属性 -> 设备管理器
- 开始菜单 -> (输入) 设备管理器
- 控制面板 -> (搜索) 设备管理器



- 可以看到设备显示 **USB2.0-Serial** , 说明未安装驱动, 若此前已安装驱动, 可以跳至步骤 5 。

3. 安装驱动

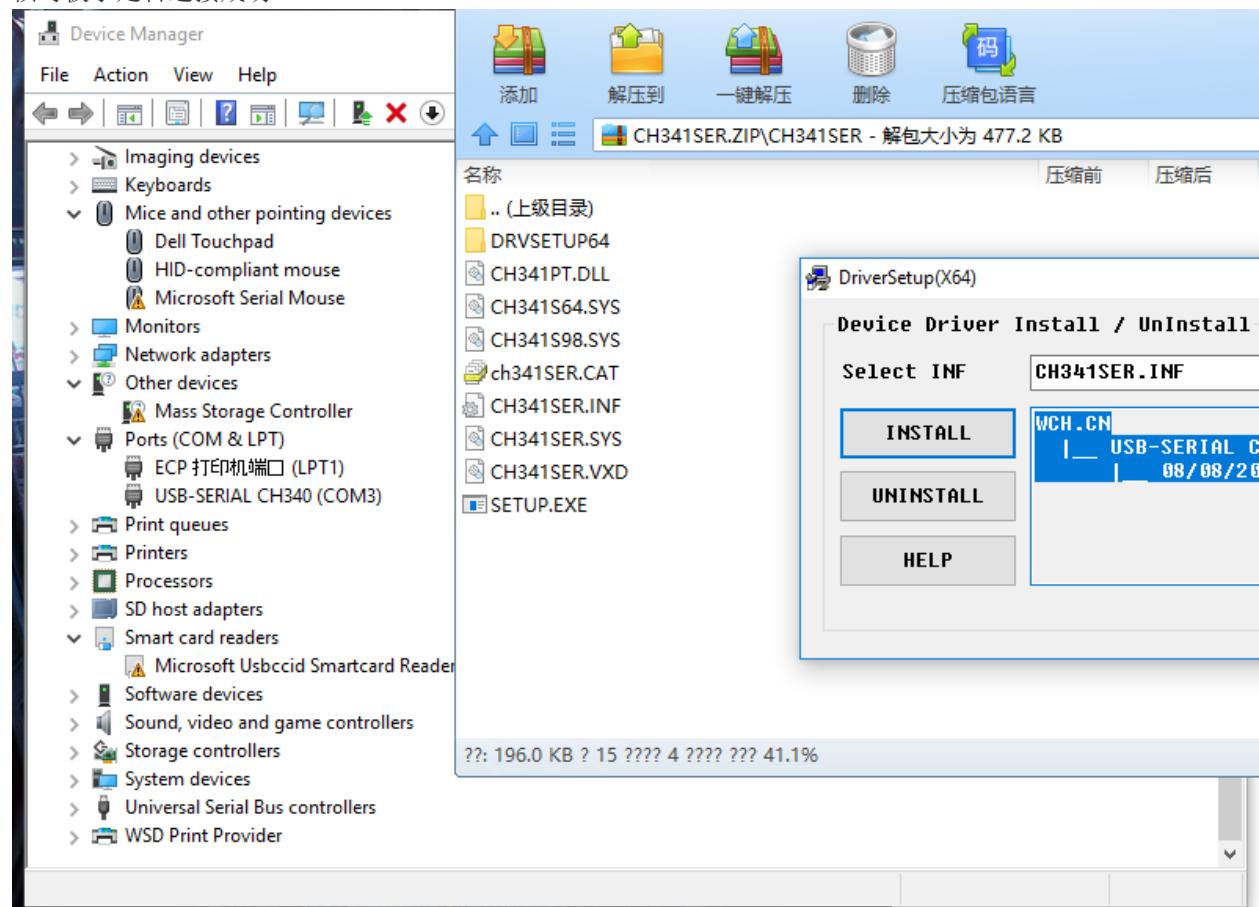
- 点此获取 [Serial CH341](#) 驱动, 并按如下说明操作安装驱动
- 打开下载的 **CH341SER.ZIP** 压缩包, 进入 **CH341SER** 文件夹, 打开 **SETUP.EXE**, 即可看到如下图。



- 点击 **INSTALL** (安装)，等待片刻即可完成安装。

4. 确认串口

- 核对板子是否连接成功



- 可以看到原来的 **USB2.0-Serial** 消失了，取而代之的是 **USB-SERIAL CH340(COM3)**，这意味着你已经成功安装驱动，并且得到板子串口名称为（**COM3**），你可以通过各种串口工具来查看串口名（**COM3**）的板子传出的信息。

2. 获得开发套件

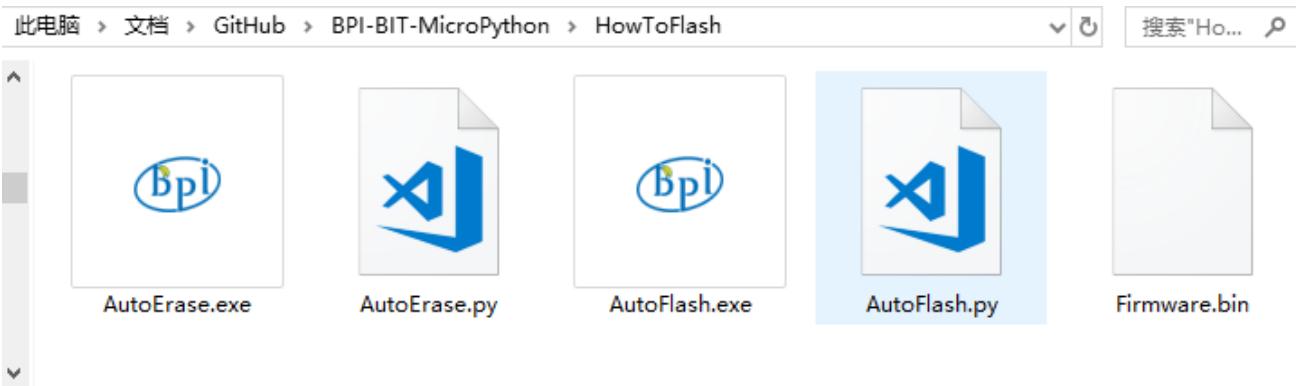
1. 获得开发套件

- [点击此处](#)直接下载目前最新的开发套件（约30M）。
- 后续更新在此获取 [BPI-BIT-MicroPython](#)，推荐使用 [GitHub-DeskTop](#) 工具随时同步更新套件。
-

名称		修改
📁 HowToCode	201	
📁 HowToFlash	201	
📁 Tools	201	
📄 .gitignore	201	
📄 LICENSE	201	
📄 main.py	201	
📄 README.md	201	
📄 webdav.json	201	

文件（夹）名称	其用途说明
HowToCode	提供了许多示例代码，可以让你参考学习或直接使用。
HowToFlash	提供了烧写或修复的工具或脚本，可以简单维护硬件。
Tools	提供了一些推荐或常用的工具，一般是不容易获取的。
main.py	板子标准启动脚本，一切开始的起点。
readme.py	标准使用说明文档引导。
webdav.json	VSCODE + WebDAV 开发模式。
LICENSE	版权声明文件

2. 尝试烧写板子



1. 擦除板子固件

- 遇到以下情况时，建议擦除一次板子。
 - 第一次使用该固件
 - 此前存在其他固件
 - 板子运行操作异常
 - 当不知所措的时候
- 以 Windows 为例，双击运行 **AutoErase.exe**，它会识别最新插入的板子的串口，并自动擦除该板子。
-

```
C:\Users\Junhuan\Documents\GitHub\BPI-BIT-MicroPython\HowToFlash\AutoErase.exe  
Looking for upload port...  
Auto-detected:COM5  
esptool.py v2.5.0  
Serial port COM5  
Connecting....  
Chip is ESP32D0WDQ6 (revision 1)  
Features: WiFi, BT, Dual Core  
MAC: 30:ae:a4:3f:a0:10  
Uploading stub...  
Running stub...  
Stub running...  
Changing baud rate to 921600  
Changed.  
Erasing flash (this may take a while)...  
Chip erase completed successfully in 4.2s  
Hard resetting via RTS pin...  
请按任意键继续. . .
```

- 以上即为成功擦除，如果失败请重试，仍然失败，请检查或设备的串口。

2. 更新板子固件

- 遇到以下情况时，建议更新一次板子。
 - 擦除板子后
 - 需升级固件
- 以 Windows 为例，双击运行 **AutoFlash.exe**，它会识别最新插入的板子的串口，并自动将同一目录下的 **Firmware.bin** 固件文件烧写到板子当中。
-

```
C:\Users\Junhuan\Documents\GitHub\BPI-BIT-MicroPython\HowToFlash\AutoFlash.exe
Looking for upload port...
Auto-detected:COM5
espptool.py v2.5.0
Serial port COM5
Connecting....
Chip is ESP32D0WDQ6 (revision 1)
Features: WiFi, BT, Dual Core
MAC: 30:ae:a4:3f:a0:10
Uploading stub...
Running stub...
Stub running...
Changing baud rate to 921600
Changed.
Configuring flash size...
Compressed 1178368 bytes to 750542...
Wrote 1178368 bytes (750542 compressed) at 0x00001000 in 10.0 seconds (effective 944.7 kbit/s)...
Hash of data verified.
rl
Leaving...
Hard resetting via RTS pin...
请按任意键继续. . .
!
```

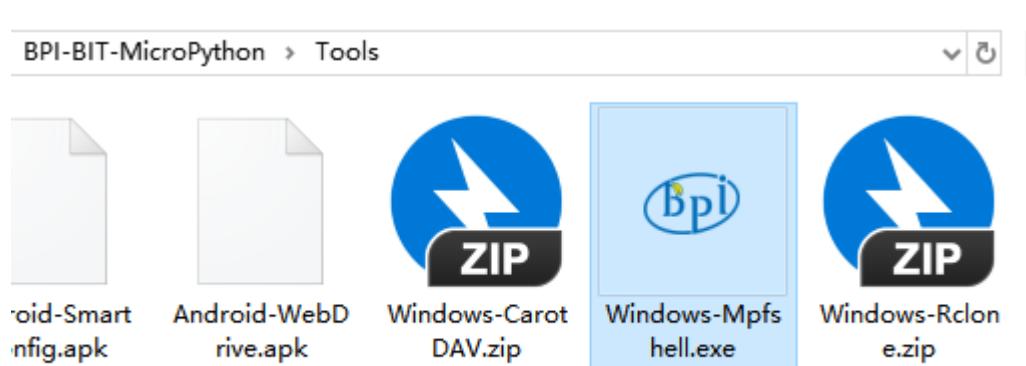
- 以上即为成功烧写，如果失败请重试，仍然失败，请检查或设备的串口。

3. Hello, World!

Hello World是一个最著名的程序。对每一位程序员来说，这个程序几乎是每一门编程语言中的第一个示例程序。实际上，这个程序的功能只是告知计算机显示Hello World这句话。传统意义上，程序员一般用这个程序测试一种新的系统或编程语言。对程序员来说，看到这两个单词显示在电脑屏幕上，往往表示他们的代码已经能够编译、装载以及正常运行了，这个输出结果就是为了证明这一点，因此以下说明以此为例。

1. 使用 **MpfShell** 编程

1. 通过 **repl** 运行程序



- (以 Windows 为例) 双击运行 **Windows-Mpfshell.exe**。
-

```
C:\Users\Junhuan\Documents\GitHub\BPI-BIT-MicroPython\Tools\Windows-Mpfshell.exe

Documented commands (type help <topic>):
=====
EOF cd exec get lcd lpwd md mput mrm put quit rm
cat close execfile help ll ls mget mpyc open pwd repl runfile

Undocumented commands:
=====
!o e ef o q r rf

All support commands, can input help ls or other command if you don't know how to use it(ls).

looking for computer port...
serial_name : com5
input 'open com5' and enter connect your board.

** Micropython File Shell v0.8.1, sw@kaltpost.de & juwan@banana-pi.com **
-- Running on Python 3.5 using PySerial 3.4 --
mpfs [/]>
```

依次出现的颜色	解释其说明
白色	显示所有支持的命令操作，上排命令表示有帮助说明，下排命令反之。
黄色	以上为全部支持的命令，如果想知道 ls 的用法，输入 help ls 即可得知。
蓝色	显示当前插入的板子的串口名，如图所示为 com5，可以输入 open com5 连接设备。

- 尝试连接设备，输入 `open com5`（可以简写为 `o com5`），如果没有反应，使用 `CTRL + C` 或右上角点红叉。

```
looking for computer port...
serial_name : com5
input 'open com5' and enter connect your board.

** Micropython File Shell v0.8.1, sw@kaltpost.de & juwan@banana-pi.com **
-- Running on Python 3.5 using PySerial 3.4 --

mpfs [/]> open com5
```

- 提示：首次启动板子，第一次会连接失败，根据提示 `try again` 重试即可，因此要注意红色的提示信息。

```
** MicroPython File Shell v0.8.1, sw@kaltpost.de & juwan@banana-pi.com **  
-- Running on Python 3.5 using PySerial 3.4 --  
  
mpfs [/]> open com5  
  
could not enter raw repl, please try again.  
  
mpfs [/]> open com5  
Connected to esp32  
mpfs [/]> -
```

- 当显示 `Connected to esp32` 时，就可以确认板子已成功连接。
- 此时再次输入 `repl`，进入基础编程环境，并写下你的第一段程序，例如：`print('Hello, world!')`。
-

```
mpfs [/]> open com5  
Connected to esp32  
mpfs [/]> repl  
>  
*** Exit REPL with Ctrl+Q ***  
  
MicroPython v1.9.4-518-g519bc39fa-dirty on 2018-09-07; ESP32 module with ESP32  
Type "help()" for more information.  
>>> print('Hello, World!')  
Hello, World!  
>>>
```

- 回显 `Hello, world!`，即说明你成功的运行了你的第一段程序，你也可以继续输入代码来测试。
- 更多的编程用法将在之后的章节中逐步涉及。

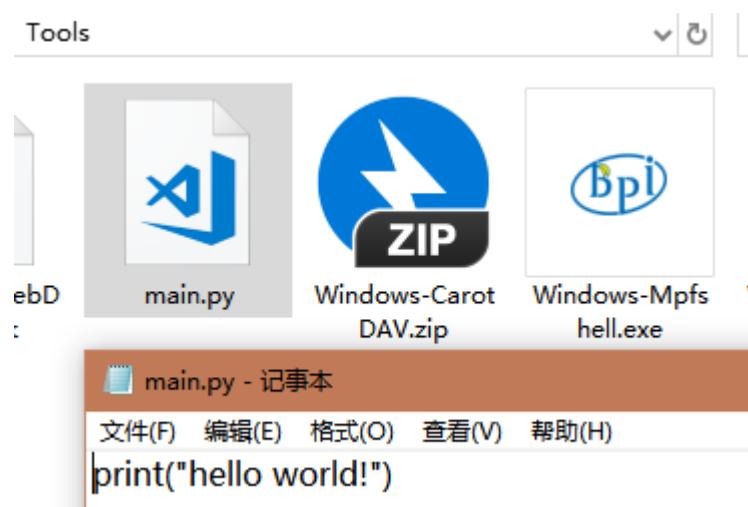
2. 通过 main.py 运行程序

- 新建一个 `main.py` 文件，并写入以下内容。

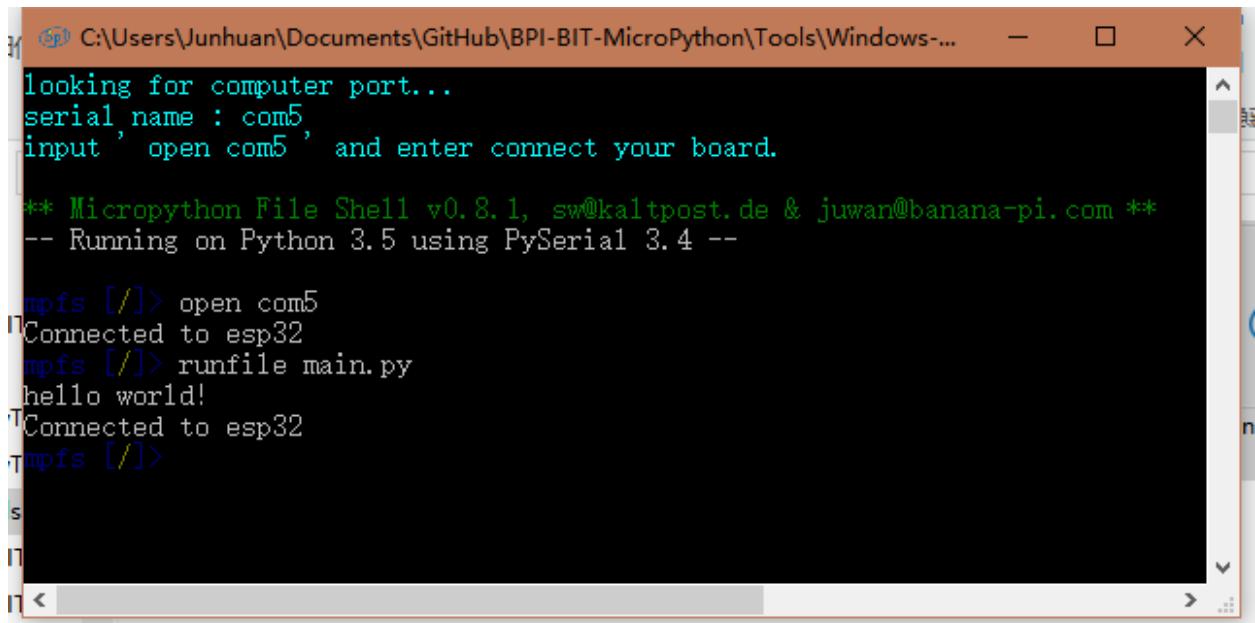
```
print('Hello, world!')
```

- 若是想使用 MpfsShell 执行该文件，请将文件放到与 MpfsShell 同一目录下。

-



- 接着再切换到 `MpfShell` 的程序区域，输入 `runfile main.py`，或输入 `rf main.py`，`runfile` 简写成 `rf`。



```

C:\Users\Junhuan\Documents\GitHub\BPI-BIT-MicroPython\Tools\Windows-...
looking for computer port...
serial_name : com5
input 'open com5', and enter connect your board.

** Micropython File Shell v0.8.1, sw@kaltpost.de & juwan@banana-pi.com **
-- Running on Python 3.5 using PySerial 3.4 --

mpfs [/]> open com5
Connected to esp32
mpfs [/]> runfile main.py
hello world!
Connected to esp32
mpfs [/]

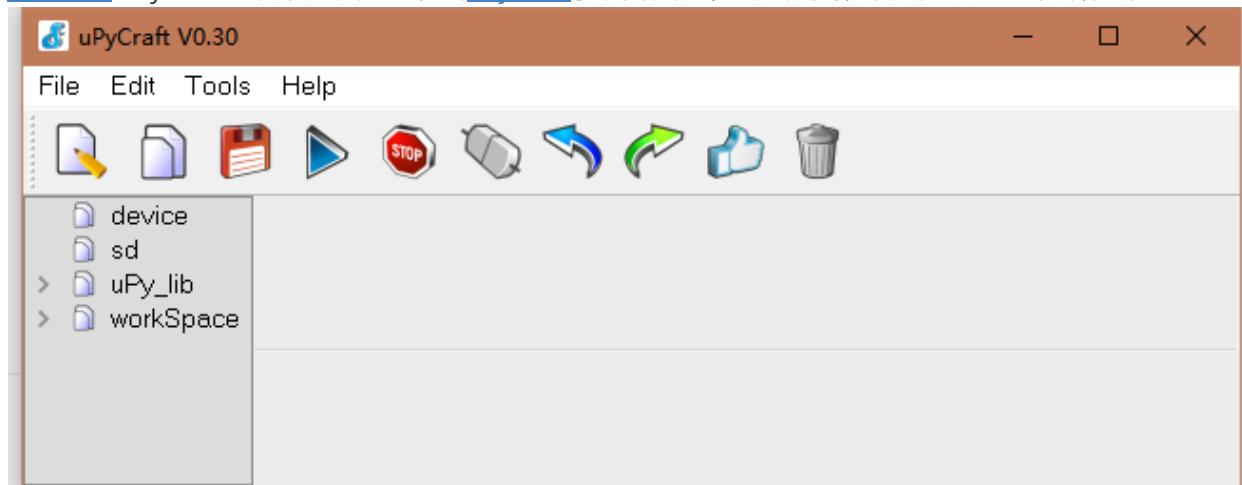
```

- 可以看到回显了 `hello world`，说明你成功运行了程序。

2. 使用 uPyCraft 编程

1. 通过 repl 运行程序

- [点此获取](#) uPyCraft3.0开发工具，可以到[uPyCraft](#)获取其他版本，启动软件的提示选 是 或 否 都无影响。



- 工具栏依次为 新建、打开、保存、运行、停止、连接、撤销、重做、语法检查、清理。

- 我们点击 连接 ，可以看到，底下框出现 `>>>` 输入提示，这表示连接成功。



- 在 `>>>` 中写下你的第一段程序，例如: `print('Hello, world!')`。



- 回显 `Hello, world!`，即说明你成功的运行了你的第一段程序，你也可以继续输入代码来测试。

2. 通过 `main.py` 运行程序

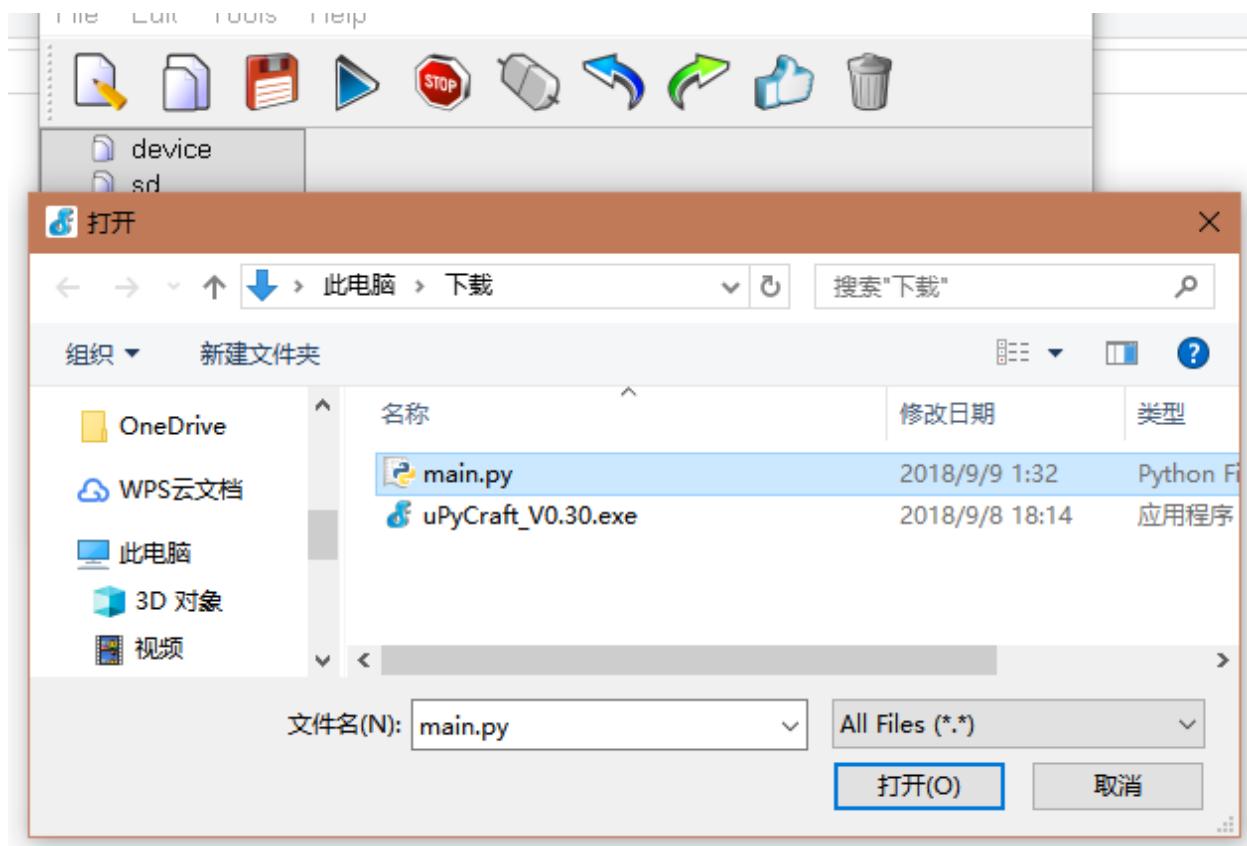
- 新建一个 `main.py` 文件，并写入以下内容。

```
print('Hello, world!')
```

- 使用 `uPyCraft` 执行该文件，将文件放到与 `uPyCraft` 同一目录下。



- 然后回到 `uPyCraft` 工具中，点击工具栏左数第二项 打开 ，选择你的代码文件，点击打开。
-



- 软件将会载入该文件到编辑器中，点击工具栏左数第四项 运行  即可。



- 可以看到 >>> 之上回显了 hello world!，说明已经成功的运行了 main.py，注意 运行 前需要先 连接 板子。

3. 使用 PyCharm 编程

使用该编程方式需要有一定的编程基础，这适合专业的开发人员，安装教程在此 [PyCharm MicroPython](#)。

4. 详解开发环境和工具

1. 什么是 repl?

REPL(Read-eval-print-loop): 交互式解析器

在REPL环境下，可以定义和运行变量、函数、对象。

REPL的常用命令：

- 启动 `MicroPython`，即进入了 REPL 环境。
- 退出：按 `Ctrl + C` 中断 或 按 `CTRL + D` 软复位。
- 点击 `tab` 可以打印出 Python 中的所有模块，或补全不完整的命令（输入过的历史命令）。
- 点击方向键 向上 或 向下 可以查看 历史命令。

常见使用场景：

- 在repl中想要输入 import 命令，只需要 输入 i + 按下 tab 键即可补全成 import，其他同理，使用的前提是此前输入过，因此不是所有命令都可以被补全。
- 在运行 python 代码时，希望能让它停下来，按下 `CTRL + C` 即可向解释器发送一个 KeyboardInterrupt 异常，此时解释器运行过程中如果未使用异常捕获处理该异常，则程序将会停止。

2. 如何选择开发工具？

名称	描述	优势	劣势
notepad	系统自带记事本	足够简单	远古高手才能用得起 XD
notepad++	系统自带记事本升级版	支持代码识别和高亮	现代高手看代码用的 XD
uPyCraft	跨平台简易开发环境	使用简单，适合新手	问题多，且缺少许多必要的编程功能，例如各种操作快捷键。
MpfShell	跨平台简易交互工具	功能完整，拓展性强	缺乏编辑器支持，代码靠外部编辑器编写，例如：VS CODE 或 PyCharm。
VS CODE	跨平台代码编辑器	专业代码标准编程环境	缺乏板子的支持，需要配合工具运行代码，例如：MpfShell、RShell、PyLoader。
PyCharm	跨平台代码编辑器	专业代码标准编程环境	支持板子的直接操作，但配置稍微复杂了一些，要使用起来没有那么简单，只适合专业开发。

个人推荐 `MpfShell` + VS CODE，理由是编写简单的代码用 `MpfShell`，编写复杂的代码用 VS CODE，无需额外配置任何东西，运行流畅，简单快捷。

4. 控制硬件外设

如果说 Hello World 对软件程序员来说是一种宗教般的编程开始，那么对在硬件程序员的人来说，Blink Led，亦如此，所有的硬件编程都是从点亮一盏灯开始的。

如此便开始准备我们的硬件了，我使用的是一个无比古老的初代 1.0 版本。它足够满足本章的要求，并且对于后续的板子来说，操作都是通用的。因此我们讲究循序渐进，先是一盏灯，再是一排灯。



1. 点亮 LED 灯

1. 使用 repl

- 进入 repl 模式
-

```
C:\Users\Junhuan\Downloads\Windows-Mpfshell.exe
serial, name : com5,
input ' open com5 ' and enter connect your board.

** MicroPython File Shell v0.8.1, sw@kaltpost.de & juwan@banana-pi.com *
*
-- Running on Python 3.5 using PySerial 3.4 --

mpfs [/]> o com5
Connected to esp32
mpfs [/]> repl
>
*** Exit REPL with Ctrl+Q ***

MicroPython v1.9.4-518-g519bc39fa-dirty on 2018-09-07; ESP32 module with
ESP32Type "help()" for more information.
>>> -
```

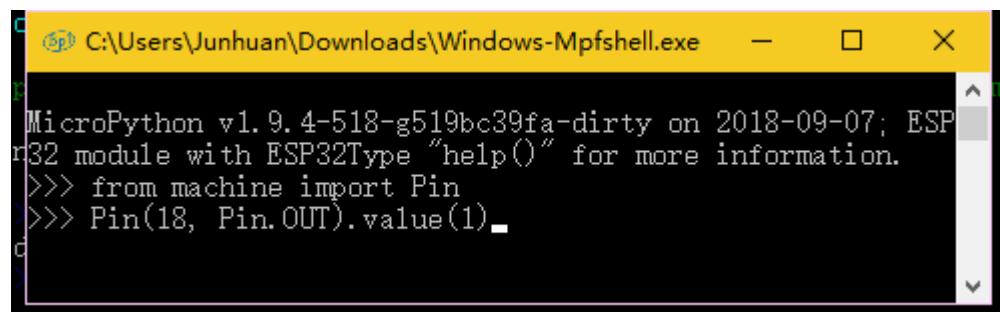
- 输入（选取文本复制后，在黑框里右键粘贴）

```
from machine import Pin
```

- 再次输入

```
Pin(18, Pin.OUT).value(1)
```

-



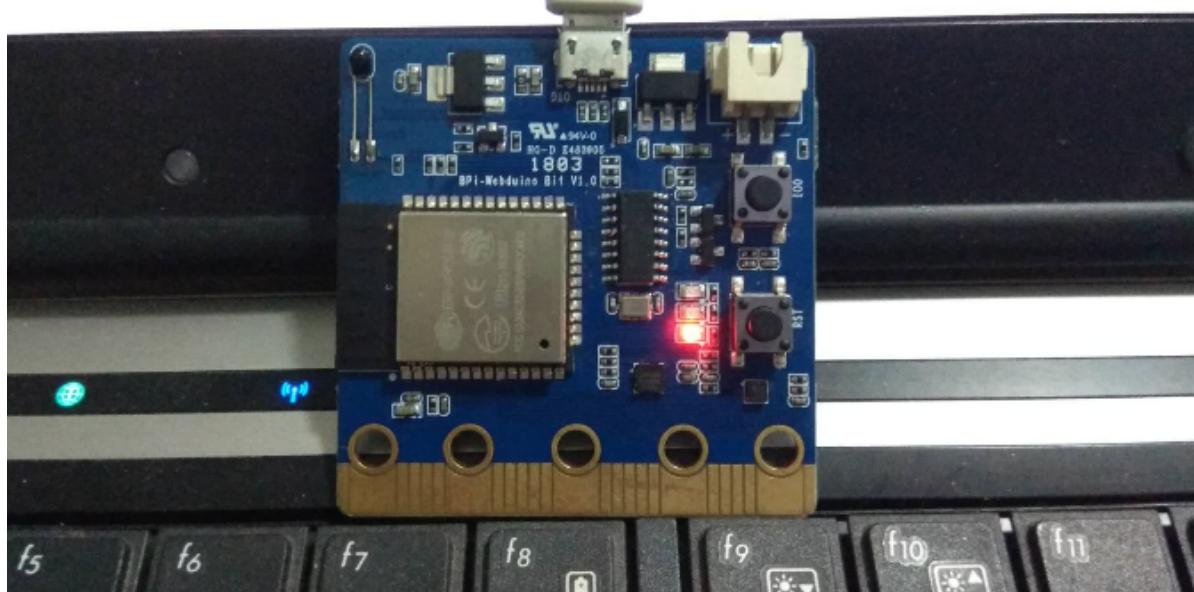
```
C:\Users\Junhuan\Downloads\Windows-Mpfshell.exe

MicroPython v1.9.4-518-g519bc39fa-dirty on 2018-09-07; ESP32 module with ESP32Type "help()" for more information.
>>> from machine import Pin
>>> Pin(18, Pin.OUT).value(1)
```

- 确定即可看到面板上的灯有一盏亮起

-

```
print('sleep 1s')
```

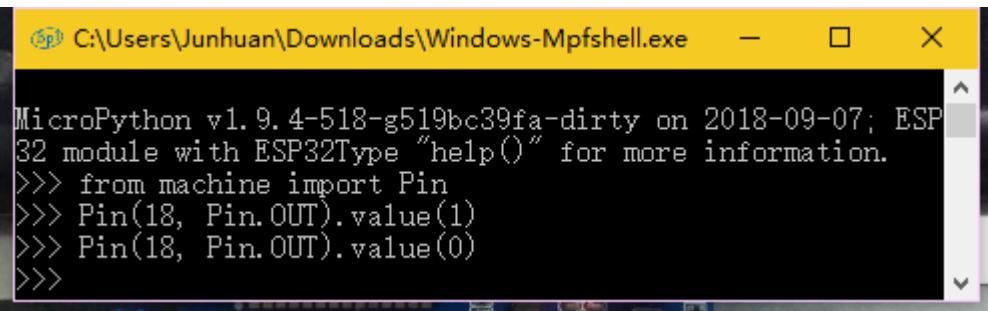


- 为了进一步确认它是我们控制的灯

- 输入

```
Pin(18, Pin.OUT).value(0)
```

-



```
C:\Users\Junhuan\Downloads\Windows-Mpfshell.exe

MicroPython v1.9.4-518-g519bc39fa-dirty on 2018-09-07; ESP32 module with ESP32Type "help()" for more information.
>>> from machine import Pin
>>> Pin(18, Pin.OUT).value(1)
>>> Pin(18, Pin.OUT).value(0)
>>>
```

- 这时就可以看到它灭了

-

```
>>> Pin(18, Pin.OUT).value(1)
>>> Pin(18, Pin.OUT).value(0)
>>>
```

○ 这时就可以看到它灭了



2. 使用 main.py

- 准备以下代码到 main.py 文件中，与上一个不同的是，这个效果是连续的，它将让灯亮起来后，等待一秒钟后灭了，由于展现的效果是连续的，没有办法通过图示来说明情况，所以自己动手试一试吧。

-

```
from machine import Pin
import time
led = Pin(18, Pin.OUT) # get a led on gpio 18.
print('turn on')
led.value(1) # turn on
print('sleep 1s')
time.sleep(1) # sleep 1s
print('turn off')
led.value(0) # turn off
```

```
mpfs [/]> o com5
Connected to esp32
mpfs [/]> runfile main.py
turn on
sleep 1s
turn off
sleep 1s
Connected to esp32
mpfs [/]>
```

```
1 from machine import Pin
2 import time
3 led = Pin(18, Pin.OUT) # get a led on gpio 18.
4 print('turn.on')
5 led.value(1) # turn on
6 print('sleep.1s')
7 time.sleep(1) # sleep 1s
8 print('turn.off')
9 led.value(0) # turn off
10 print('sleep.1s')
11 time.sleep(1) # sleep 1s
12
```

length : 262 lines Ln : 9 Col : 18 Sel : 0

- 若是觉得效果不明显，可以写成死循环来查看效果，注意使用 `ctrl + c` 停下来，否则无法继续操作。

```
from machine import Pin
import time
led = Pin(18, Pin.OUT) # get a led on gpio 18.
while True:
    print('turn on')
    led.value(1) # turn on
    print('sleep 1s')
    time.sleep(1) # sleep 1s
    print('turn off')
    led.value(0) # turn off
    print('sleep 1s')
    time.sleep(1) # sleep 1s
```

- o

```
from machine import Pin
import time
led = Pin(18, Pin.OUT) #.get()
while True:
    print('turn.on')
    led.value(1) #.turn.on
    print('sleep.1s')
    time.sleep(1) #.sleep.1s
    print('turn.off')
    led.value(0) #.turn.off
    print('sleep.1s')
    time.sleep(1) #.sleep.1s
```

mpfs [/]> o com5
Connected to esp32
mpfs [/]> runfile main.py
turn on
sleep 1s
turn off
sleep 1s
Connected to esp32
mpfs [/]> runfile main.py
turn on
sleep 1s
turn off
sleep 1s
turn on
sleep 1s
turn off
sleep 1s
turn on
sleep 1s
turn off
sleep 1s
Connected to esp32
mpfs [/]>

2. 控制全彩LED阵列螺旋闪烁

1. 学习点亮面板上的LED灯阵列（NeoPixel）

- 准备以下代码到 main.py 中（\HowToCode\01.LEDs\rgb_lattice.py）

```
from display import BpiBitNeoPixel, NeoPixelPower
NeoPixelPower(True)
View = BpiBitNeoPixel()

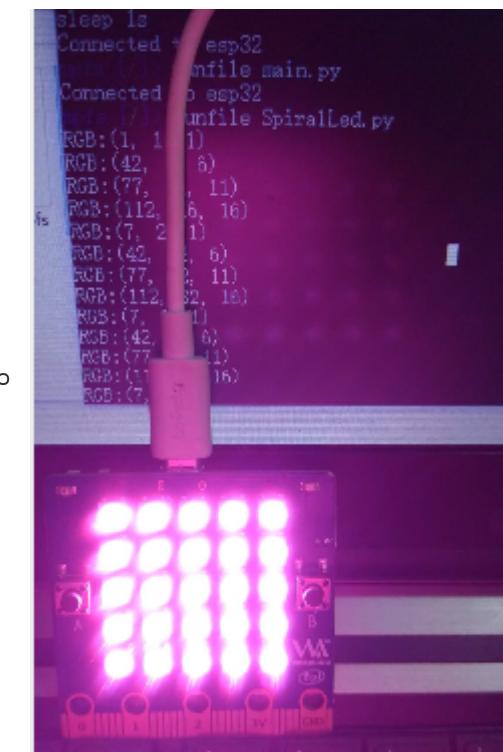
RGB = (10, 10, 10)
View.LoadXY(2, 2, RGB)
View.Show()
```

- 使用 `runfile main.py` 或 `uPyCraft` 点击运行执行均可。

- o



2. 标题效果的代码实现在 `HowToCode\08.App\SpiralLed.py`，调用试试看效果吧。



5. 实现基础算法

上章我们学会了一些基础的硬件控制后，就来补补我们的软件基础课程吧。

1. 输入年份判断闰年

- 闰年是公历中的名词。普通年:能被4整除但不能被100整除的年份为普通闰年。（如2004年就是闰年，1999年不是闰年）；世纪年:能被400整除的为世纪闰年。
- 则有如下代码：（可以放在 main.py 中）

```
def is_leap_year(year):  
    if (year % 4) == 0 and (year % 100) != 0 or (year % 400) == 0:  
        print("{0} is leap year".format(year))  
    else:  
        print("{0} not is leap year".format(year))
```

- 先使用 `runfile main.py` 将其运行到解释器中，再到 repl 中调用 `is_leap_year(int(input("input a leap year ")))`。

The screenshot shows a terminal window with the following content:

```
C:\Users\Junhuan\Downloads\main.py - No. 1  
C:\Users\Junhuan\Downloads\Windows-Mpfsh...  
文件(F) 编辑(E) 搜索(S) 视图(V) 编码(N) | mpfs [/] > open com5  
| Connected to esp32  
| mpfs [/] > runfile main.py  
| Connected to esp32  
| mpfs [/] > repl  
>  
1 def.is_leap_year(year):  
2     if (year % 4) == 0 and (year % 100) != 0 or (year % 400) == 0:  
3         print("{0} is leap year".format(year))  
4     else:  
5         print("{0} not is leap year".format(year))  
6  
7  
8  
MicroPython v1.9.4-519-g0a39aa15d-dirty on 2018-09-09; E  
SP32 module with ESP32  
Type "help()" for more information.  
>>> is_leap_year(int(input("input a leap year ")))  
input a leap year 2004  
2004 is leap year  
>>> is_leap_year(int(input("input a leap year ")))  
input a leap year 1999  
1999 not is leap year  
>>> is_leap_year(int(input("input a leap year ")))  
input a leap year 2000  
2000 is leap year  
>>> is_leap_year(int(input("input a leap year ")))  
input a leap year 2001  
2001 not is leap year  
Pj length : 190  lines : 8  Ln : 8  Cc
```

- 可以看到 2004 is leap year，说明 2004 年是闰年。
- 可以看到 1999 not is leap year，说明 1999 年不是闰年。
- 可以看到 2000 is leap year，说明 2000 年是闰年。
- 你也可以继续输入更多，让它回答你哪一年是闰年。
- 当在 REPL 时输入命令后，会提示你输入一个数值，输入后并按下确定键确认输入。

2. 生成斐波那契数列

- 什么是斐波那契数列？它指的是，有这样一个数列 0, 1, 1, 2, 3, 5, 8, 13，特别指出：第0项是0，第1项是第一个1。从第三项开始，每一项都等于前两项之和。
- 则有如下代码：（可以放在 main.py 中）

```

def fab(n):
    if n == 1:
        return 0
    if n == 2:
        return 1
    if n > 2:
        return fab(n-1) + fab(n-2)

def printfablist(n):
    for i in range(1, n+1):
        print(fab(i), end = ' ')
    print('')

```

- 先使用 `runfile main.py` 将其运行到解释器中，再到 `repl` 中调用 `printfablist(int(input('please input a number:')))`。

```

C:\Users\Junhuan\Downloads\main.py - Note C:\Users\Junhuan\Downloads\Windows-Mpfsh...
文件(E) 编辑(E) 搜索(S) 视图(V) 编码(N) 语
main.py
1 def.fab(n):
2     if.n==1:
3         return.0
4     if.n==2:
5         return.1
6     if.n>2:
7         return.fab(n-1)+fab(n-2)
8
9 def.printfablist(n):
10    for.i.in.range(1,.n+1):
11        print(fab(i),end=' ')
12    print('')

mpfs [/]> o com5
Connected to esp32
mpfs [/]> rf main.py
Connected to esp32
mpfs [/]> r
>
*** Exit REPL with Ctrl+Q ***
MicroPython v1.9.4-519-g0a39aa15d-dirty on 2018-09-09; ESP32 module with ESP32
Type "help()" for more information.
>>> printfablist(int(input('please input a number:')))
please input a number:5
0 1 1 2 3
>>> printfablist(int(input('please input a number:')))
please input a number:8
0 1 1 2 3 5 8 13
>>> printfablist(int(input('please input a number:')))
please input a number:10
0 1 1 2 3 5 8 13 21 34
>>> printfablist(int(input('please input a number:')))
please input a number:20

```

- 可以自己试着运算第一项和第二项的关系。
- 当在REPL时输入命令后，会提示你输入一个数值，并按下确定输入，比如我输入了的5，则依次输出五个项的斐波那契数列（0 1 1 2 3）。

6. 启动无线模式

1. 连接 WIFI 热点

- 在这之前先进入 `repl` 输入 `import webrepl_setup` 启动网络配置流程。
- 根据步骤依次为 (E、1234、y)
 - 启动网络服务配置（启动输入 E，停止输入 D）
 - 设置网络连接密码（不少于4位，需输入两遍）
 - 是否需要重启板子（复位输入y，否则输入n）

```
④ X:\Users\pi\Documents\GitHub\BPI-BIT-MicroPython\Tools\Windows-Mpfshell.exe - □ × ↻
mpfs [/]> o com3
Connected to esp32
mpfs [/]> r
>
*** Exit REPL with Ctrl+Q ***

MicroPython v1.9.4-519-g0a39aa15d-dirty on 2018-09-09; ESP32 module with ESP32
Type "help()" for more information.
>>> import webrepl_setup
WebREPL daemon auto-start status: disabled

Would you like to (E)nable or (D)isable it running on boot?
(Empty line to quit)
> E
To enable WebREPL, you must set password for it
New password (4-9 chars): 1234
Confirm password: 1234
Changes will be activated after reboot
Would you like to reboot now? (y/n) y


```

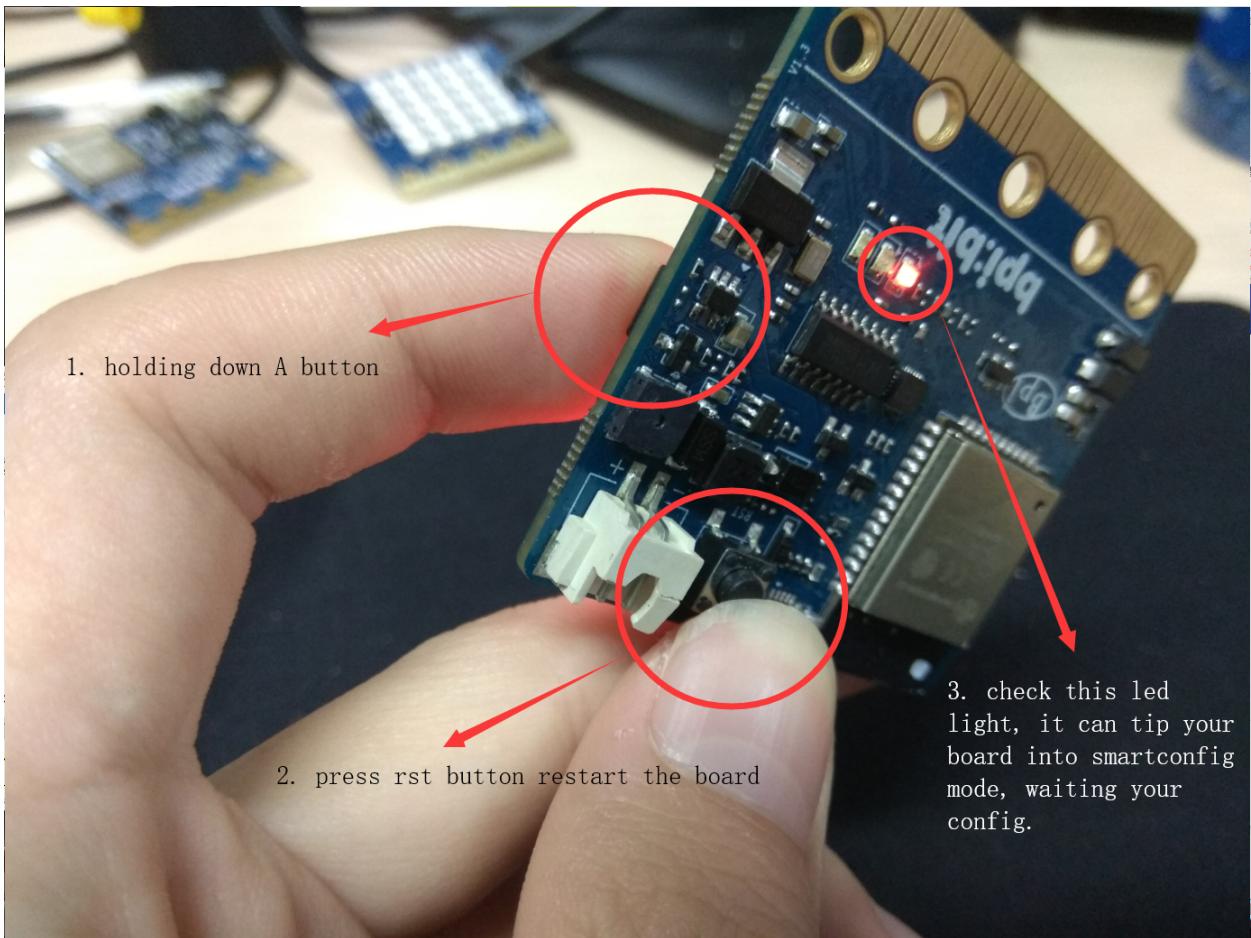
- 如果没有配网过，板子会自动连接 WiFi 名称 `webduino.io` 密码 `webduino` 的 WiFi 热点，如果附近没有将会输出 `no AP found`，而我的环境里提前准备了这个热点，因此会得到一个IP地址 `192.168.1.214`。

```
I (194) user_smartconfig: default smartconfig config

W (194) phy_init: failed to load RF calibration data (0x1102), falling back to full calibration
I (304) phy: phy_version: 3960, 5211945, Jul 18 2018, 10:40:07, 0, 2
I (304) wifi: mode : sta (b4:e6:2d:97:3d:e1)
I (304) user_smartconfig: tcpip_adapter_set_hostname = bit3de1
I (314) gpio: GPIO[18]| InputEn: 0| OutputEn: 0| OpenDrain: 0| Pullup: 1| PULLDOWN: 0| Intr:0
10 seconds waiting network connected.....
I (1194) wifi: n:11 0, o:1 0, ap:255 255, sta:11 0, prof:1
I (1764) wifi: state: init -> auth (b0)
I (1764) wifi: state: auth -> assoc (0)
I (1764) wifi: state: assoc -> run (10)
I (1774) wifi: connected with webduino.io, channel 11
I (1784) wifi: pm start, type: 1

I (3264) event: sta ip: 192.168.1.214, mask: 255.255.255.0, gw: 192.168.1.1
I (3264) network: GOT_IP
I (3264) modsocket: Initializing
WebREPL daemon started on ws://192.168.1.214:8266
Started webrepl in normal mode
OSError: [Errno 2] ENOENT
MicroPython v1.9.4-519-g0a39aa15d-dirty on 2018-09-09; ESP32 module with ESP32
Type "help()" for more information.
>>>
```

- 当然，你的WiFi热点肯定不是这个，所以你现在可以按住 **A**键 后按下 复位键 启动进入配网模式，重新给板子连入其他WiFi，帮助板子连上指定 WiFi，进入 `SmartConfig` 的配网模式，LED (18) 将会亮起，LED (18) 是前面章节点亮过的 LED 灯。



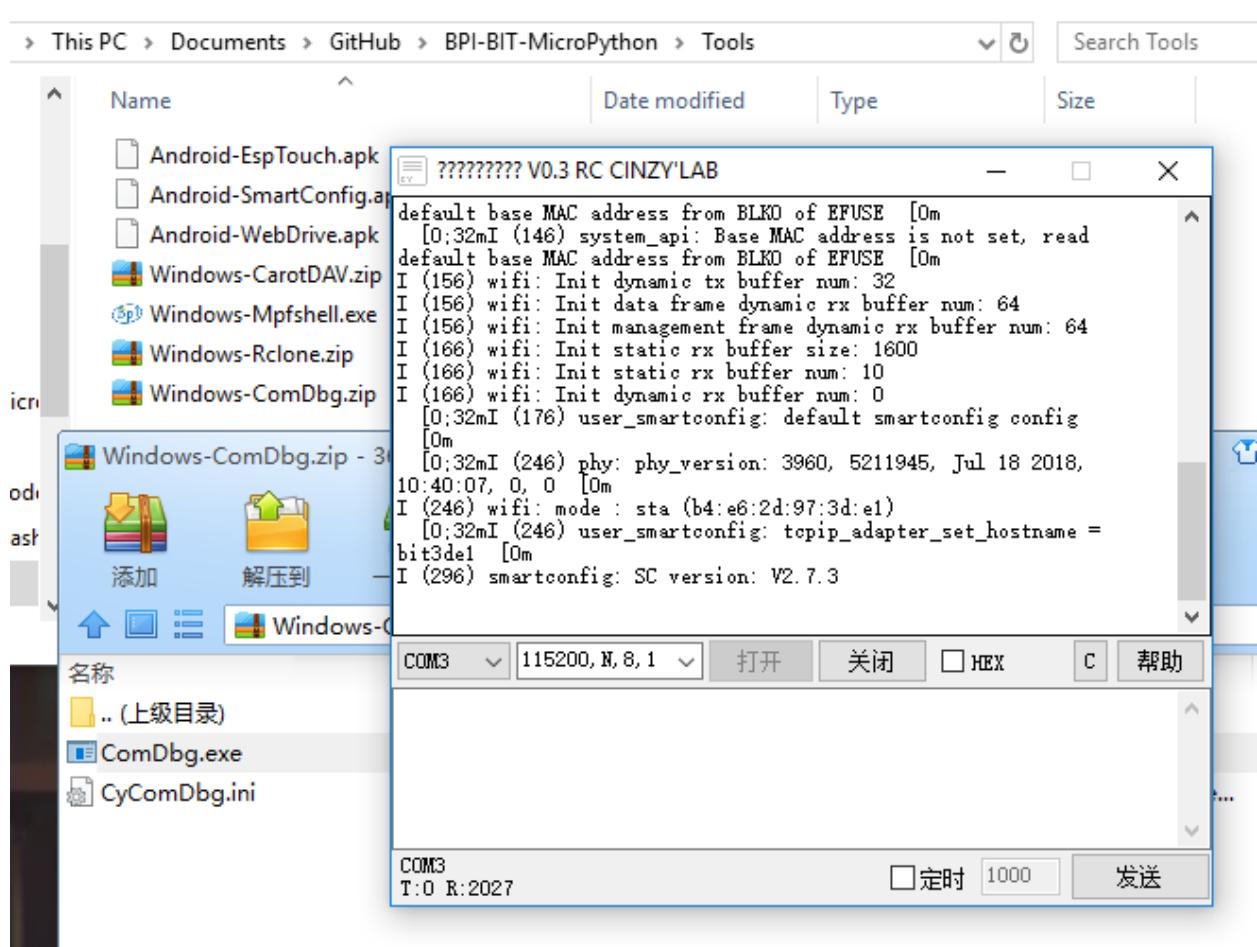
- 复位后看板子的 LED 灯 长亮 即可确认进入了配网模式，此时串口也会输出对应的信息，注意在这个模式将无法正常使用 `Mpfshell` 的 `open`，这是因为此时板子已经无法响应串口的 REPL 操作了，因此你需要配网完成后再次使用 `Mpfshell`。
-

```
X:\Users\pi\Documents\GitHub\BPI-BIT-MicroPython\Tools\Windows-Mpf... - X

I (53) cpu_start: Starting scheduler on PRO CPU.
network smartconfig
I (136) wifi: wifi driver task: 3ffd560c, prio:23, stack:3584, core=0
I (136) wifi: wifi firmware version: 633012a
I (136) wifi: config NVS flash: enabled
I (136) wifi: config nano formating: disabled
I (146) system_api: Base MAC address is not set, read default base MAC
address from BLK0 of EFUSE
I (146) system_api: Base MAC address is not set, read default base MAC
address from BLK0 of EFUSE
I (156) wifi: Init dynamic tx buffer num: 32
I (156) wifi: Init data frame dynamic rx buffer num: 64
I (156) wifi: Init management frame dynamic rx buffer num: 64
I (166) wifi: Init static rx buffer size: 1600
I (166) wifi: Init static rx buffer num: 10
I (166) wifi: Init dynamic rx buffer num: 0
I (176) user_smartconfig: default smartconfig config

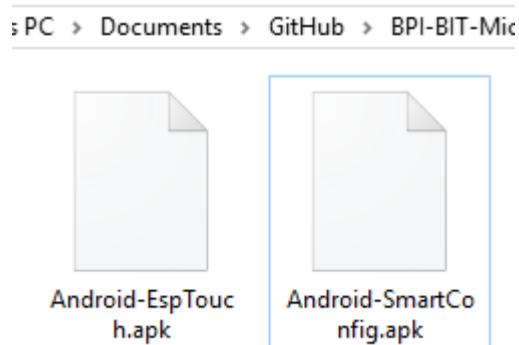
I (256) phy: phy_version: 3960, 5211945, Jul 18 2018, 10:40:07, 0, 0
I (256) wifi: mode : sta (b4:e6:2d:97:3d:e1)
I (256) user_smartconfig: tcpip_adapter_set_hostname = bit3de1
I (306) smartconfig: SC version: V2.7.3
I (3516) wifi: ic_enable_sniffer
```

- 如果期间工具运行出错，请使用其他串口工具查看输出信息，例如 Tools 目录下的 `Windows-ComDbg.zip`。
-



- 确认进入了配网模式后，此时你需要使用一台安卓手机来安装套件提供的配网软件

-



- 以 `Android-smartConfig.apk` 为例，先将手机连入WIFI，然后再将让板子也连入同一个WIFI，再到软件中输入所连WIFI的密码，这将告知板子，如何连接到该WIFI。

-

17:15

11.8K/s ◻ ◻ ◻ VoLTE ◻ 99



SmartConfig

SSID : tgoffice

Password :

tu9u2017

CONFIRM

- 点击唯一的按钮启动配网，可以看到 `repl` 有对应信息输出，同时板子的 LED 灯也会跟着变化。
-

17:44

32.4K/s ⏺ ⏻ ⏸ VoLTE 98



SmartConfig

SSID : tgoffice

Password :

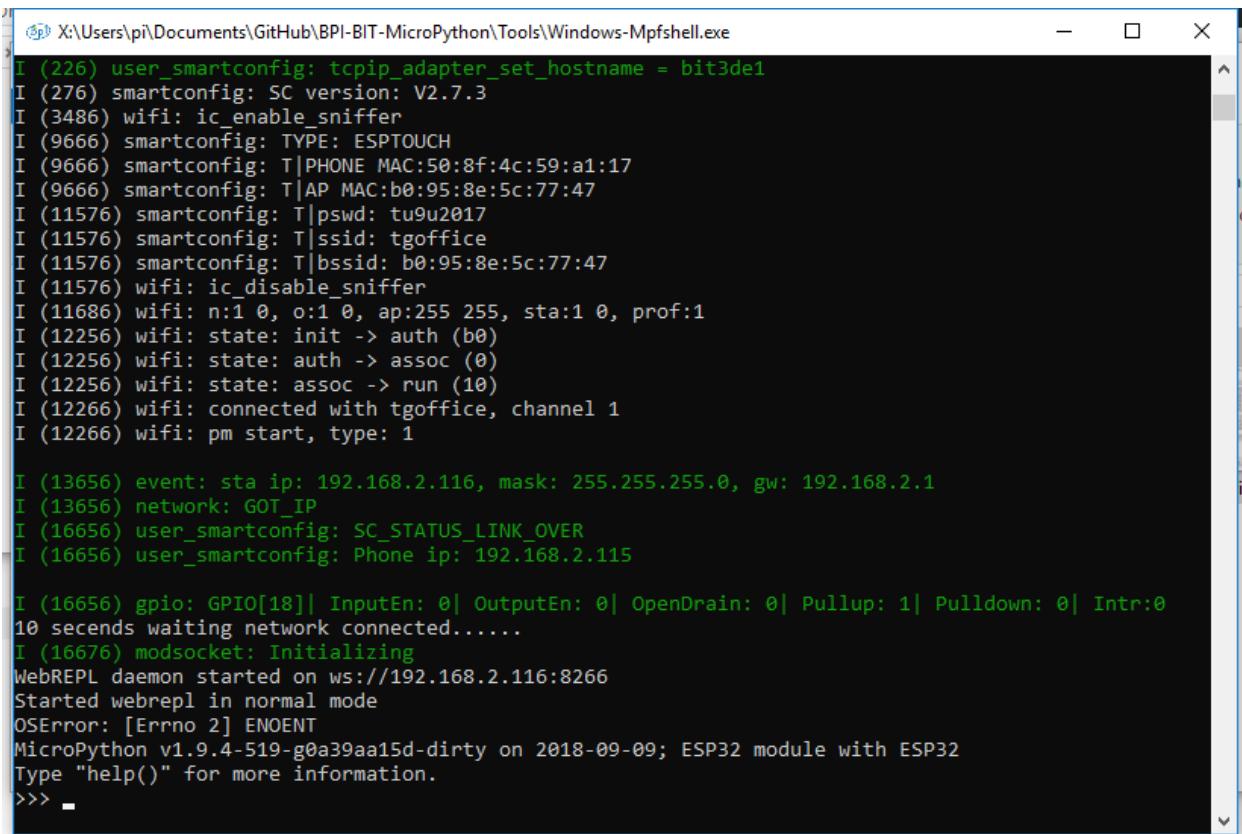


Esptouch is configuring, please wait
for a moment...

WAITING...

CONFIRM

- 等待一会，如果卡在了配网模式没有成功，则会在两分钟内会自动重启。而当配网成功后，LED 灯会变成微亮，此时 `repl` 会输出板子连上 WIFI 得到的 IP 地址，如下图为：192.168.2.116，并且值得注意的是 3de1 就对应的是板子的名称，这个名称以后会用到。



```
(226) user_smartconfig: tcpip_adapter_set_hostname = bit3de1
I (276) smartconfig: SC version: V2.7.3
I (3486) wifi: ic_enable_sniffer
I (9666) smartconfig: TYPE: ESPTOUCH
I (9666) smartconfig: T|PHONE MAC:50:8f:4c:59:a1:17
I (9666) smartconfig: T|AP MAC:b0:95:8e:5c:77:47
I (11576) smartconfig: T|pswd: tu9u2017
I (11576) smartconfig: T|ssid: tgoffice
I (11576) smartconfig: T|bssid: b0:95:8e:5c:77:47
I (11576) wifi: ic_disable_sniffer
I (11686) wifi: n:1 0, o:1 0, ap:255 255, sta:1 0, prof:1
I (12256) wifi: state: init -> auth (b0)
I (12256) wifi: state: auth -> assoc (0)
I (12256) wifi: state: assoc -> run (10)
I (12266) wifi: connected with tgoffice, channel 1
I (12266) wifi: pm start, type: 1

I (13656) event: sta ip: 192.168.2.116, mask: 255.255.255.0, gw: 192.168.2.1
I (13656) network: GOT_IP
I (16656) user_smartconfig: SC_STATUS_LINK_OVER
I (16656) user_smartconfig: Phone ip: 192.168.2.115

I (16656) gpio: GPIO[18]| InputEn: 0| OutputEn: 0| OpenDrain: 0| Pullup: 1| Pulldown: 0| Intr:0
10 secends waiting network connected.....
I (16676) modsocket: Initializing
WebREPL daemon started on ws://192.168.2.116:8266
Started webrepl in normal mode
OSError: [Errno 2] ENOENT
MicroPython v1.9.4-519-g0a39aa15d-dirty on 2018-09-09; ESP32 module with ESP32
Type "help()" for more information.
>>> -
```

- 并且在手机上，也会看到板子的 IP 地址，此时板子已经完成了网络配置。

17:44

17.3K/s VoLTE 98



SmartConfig

SSID : tgoffice

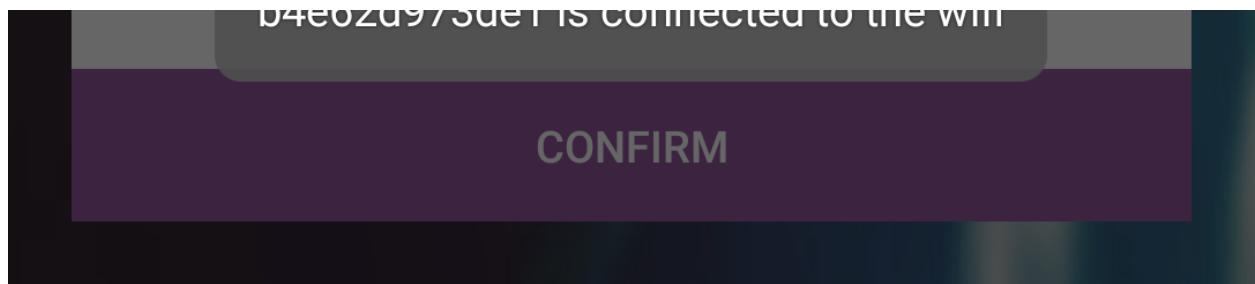
Password :



Esptouch success, bssid = b4e62d973de1,InetAddress = 192.168.2.116

CONFIRM

b4e62d072de1 is connected to the wifi



- 小提示：如果配网失败，请按以下流程排除问题。

- 确认进入了配网模式
- 确认 WIFI 热点密码无误
- 确认配网过程中 REPL 输出的信息与图示相近
- 确认 WIFI 射频是 2.4Ghz
- 更换配网环境，尤其是封闭式空间

2. 无线使用REPL

- 注意，使用前确保允许应用通过网络防火墙，且电脑与板子连接处于同一网络下（同一个WIFI下）。
- 此前我已经知道了板子现在的 IP 为 192.168.2.116，如果不知道可以重新上电查看，接着使用 mpfshell，输入 ws:192.168.2.116,1234，其中 ,1234 是我此前设定的连接密码（第一章），你也可以现在不输入，但待会也一样会提示你输入密码的。（注意是英文输入法的逗号）

```
X:\Users\pi\Documents\GitHub\BPI-BIT-MicroPython\Tools\Windows-Mpfshell.exe
I (217) wifi: mode : sta (b4:e6:2d:97:3d:e1)
I (217) user_smartconfig: tcpip_adapter_set_hostname = bit3de1
I (227) gpio: GPIO[18]: InputEn: 0| OutputEn: 0| OpenDrain: 0| Pullup: 1| Pulldown: 0| Intr:0
y10 seconds waiting network connected.....
I (307) wifi: n:1 0, o:1 0, ap:255 255, sta:1 0, prof:1
I (867) wifi: state: init -> auth (b0)
I (867) wifi: state: auth -> assoc (0)
I (877) wifi: state: assoc -> run (10)
I (887) wifi: connected with tgooffice, channel 1
I (887) wifi: pm start, type: 1

I (1677) event: sta ip: 192.168.2.116, mask: 255.255.255.0, gw: 192.168.2.1
I (1677) network: GOT_IP
I (1677) modsocket: Initializing
WebREPL daemon started on ws://192.168.2.116:8266
Started webrepl in normal mode
nOSErrno: [Errno 2] ENOENT
MicroPython v1.9.4-519-g0a39aa15d-dirty on 2018-09-09; ESP32 module with ESP32
Type "help()" for more information.
>>>
mpfs [/]> o ws:192.168.2.116,1234
Connected to esp32
mpfs [/]> repl
v>
*** Exit REPL with Ctrl+Q ***
ls
MicroPython v1.9.4-519-g0a39aa15d-dirty on 2018-09-09; ESP32 module with ESP32
Type "help()" for more information.
>>> -
```

- 可以看到已经连接成功，此时板子也可以透过无线来操作了，你也可以重启复位再试一次。
- 连接失败会有以下两种提示：
 - 连接远端无响应，提示 WebREPL Remote IP Does not respond，分析的情况是一种可能是与板子不同属一个网络，另一种可能是各种软件或硬件防火墙挡住了。
 - 连接密码错误，提示 WebREPL Password Error，重新输入密码即可，也许你连到别人的板子去了呢。

- 出现问题时的操作，假设连不上，先用有线进去按Ctrl + D软复位核对连接，接着退出来换成无线连接。

```

X:\Users\pi\Documents\GitHub\BPI-BIT-MicroPython\Tools\Windows-Mpfshell.exe
** Micropython File Shell v0.8.1, sw@kaltpost.de & juwan@banana-pi.com **
-- Running on Python 3.6 using PySerial 3.4 --

mpfs [/]> open ws:192.168.2.116,1234
WebREPL Remote IP does not respond, check belong to the same network.

Failed to open: ws:192.168.2.116,1234

mpfs [/]> o com3
Connected to esp32
mpfs [/]> r
>
*** Exit REPL with Ctrl+Q ***

MicroPython v1.9.4-519-g0a39aa15d-dirty on 2018-09-09; ESP32 module with ESP32
Type "help()" for more information.
>>>
PYB: soft reboot
dupterm: Exception in write() method, deactivating: OSError: [Errno 9] EBADF
network smartconfig
10 seconds waiting network connected.....
WebREPL daemon started on ws://192.168.2.116:8266
Started webrepl in normal mode
 OSError: [Errno 2] ENOENT
MicroPython v1.9.4-519-g0a39aa15d-dirty on 2018-09-09; ESP32 module with ESP32
Type "help()" for more information.
>>>
mpfs [/]> open ws:192.168.2.116,1234
Connected to esp32
mpfs [/]> open ws:192.168.2.116,123
WebREPL Password Error

Failed to open: ws:192.168.2.116,123
mpfs [/]> open ws:192.168.2.110,1234
WebREPL Remote IP does not respond, check belong to the same network.

Failed to open: ws:192.168.2.110,1234
mpfs [/]>

```

3. 无线使用WebDAV

- 在此之前我们需要先启动WebDAV服务，确认返回 True 即可，如果出现问题可以复位重试。

```

** Micropython File Shell v0.8.1, sw@kaltpost.de & juwan@banana-pi.com **
-- Running on Python 3.6 using PySerial 3.4 --

mpfs [/]> o ws:192.168.2.116,1234
Connected to esp32
mpfs [/]> repl
>
*** Exit REPL with Ctrl+Q ***

MicroPython v1.9.4-519-g0a39aa15d-dirty on 2018-09-09; ESP32 module with ESP32
Type "help()" for more information.
>>> import webdav
>>> webdav.start()
True

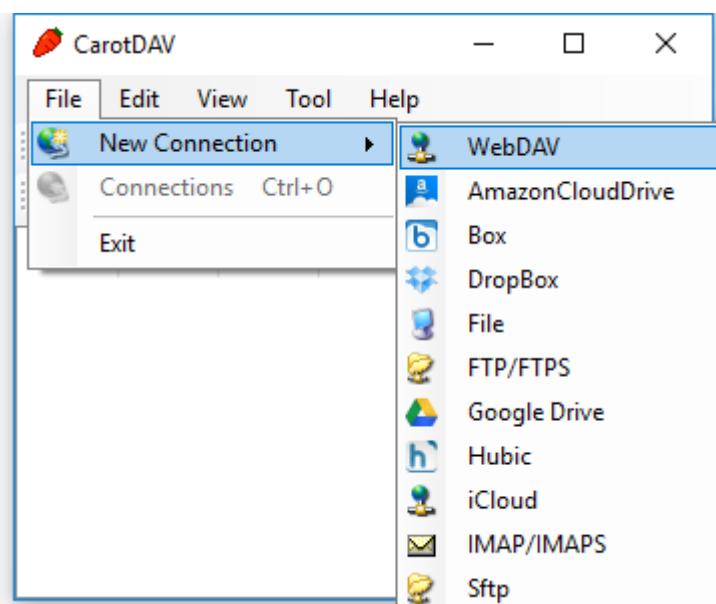
```

- 接着使用一些 WebDAV 工具来访问到板子的空间当中，这里使用 CarotDAV 来测试 webdav 是否正常工作。

●

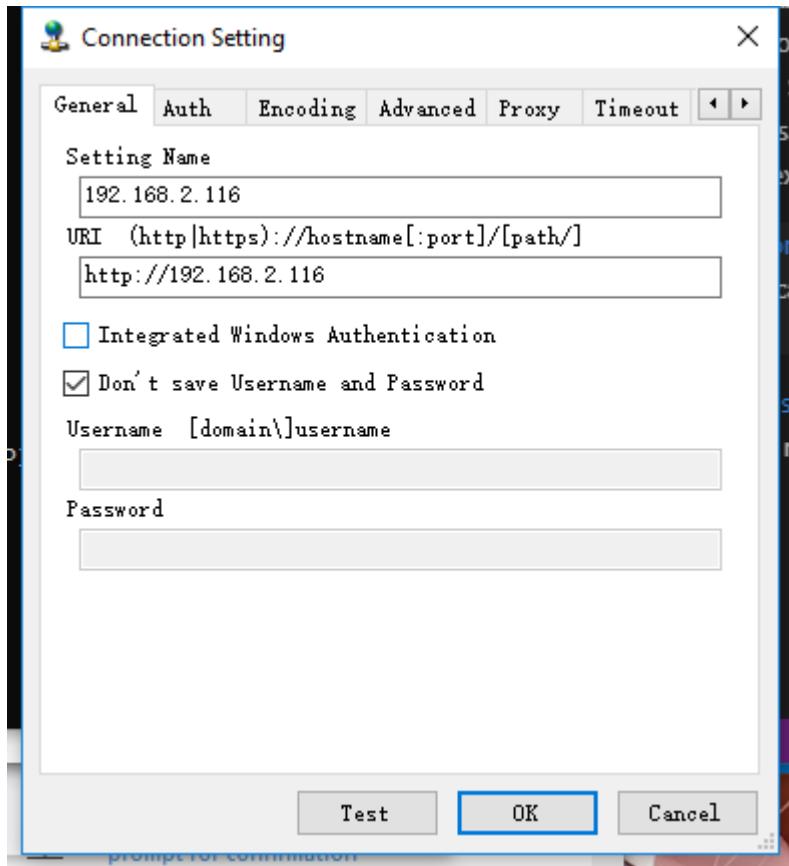
名称	压缩前	压缩后	类型
.. (上级目录)			File folder
04WebServer.xml	1.8 KB	1 KB	XML Document
CarotDAV.exe	101.4 KB	Application	
CarotDAV.exe.c	1 KB	CONFIG File	
CarotDAVC.exe	58.4 KB	Application	
CarotDAVC.exe	1 KB	CONFIG File	
IIS.xml	1 KB	XML Document	
nginx 1.4.6.xml	1 KB	XML Document	
readme_en.txt	1 KB	Text Document	
readme_ja.txt	1.7 KB	Text Document	
Rei.Com.dll	14.8 KB	Application exte...	
Rei.Fs.AmazonC	48.1 KB	Application exte...	
Rei.Fs.Box.dll	06.7 KB	Application exte...	
Rei.Fs.dll	38.5 KB	Application exte...	

- 点击 **File** -> **New Connection** -> **WebDAV**。

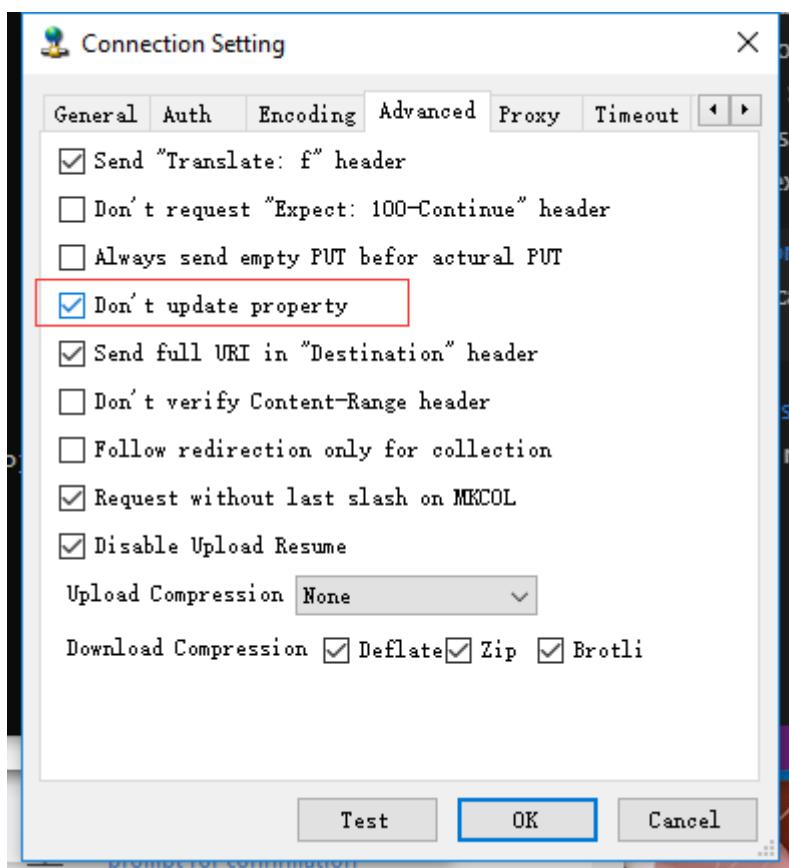


- 此前已知我手上的板子 IP 是 192.168.2.116，而你要在此填入你的板子IP，不是我的。

-



- 然后勾选一个选项，这个可以减少不必要的提示信息，当然你也可以不管。



- 最后你会看到你的板子的文件信息，并且可以将它当作一个网盘来操作。

The terminal window shows the MicroPython REPL connected to an ESP32 module via PySerial 3.4. It lists commands like cat, close, execfile, help, lls, ls, mget, mpyc, open, pwd, repl, and runfile. It also shows undocumented commands: c, e, ef, o, q, r, rf. A note says "All support commands, can input help ls or other command if you don't know how to use it(ls)." It then connects to port com3 and enters the REPL. The code then runs a script to start a WebDAV server at 192.168.2.116. The browser window shows the CaroDAV interface at http://192.168.2.116/. It lists two files: boot.py and webrepl_cfg.py, both with HTTP/1.1 200 OK status codes.

- 关于 WebDAV 访问有很多更轻松、更好玩的方式，在此就不展开了，想知道更多可以关注我的 GIT 。
 - 使用 `Rclone` 实现的板子的磁盘映射，可以把板子当做一个磁盘来操作。
 - 通过 `webDav` 实现的在线网页代码编辑器，在网页上随时开始你的编程。
- 如果你想在无线环境下开始新的编程方式，你还需要启动一个功能，它可以设定指定的文件在修改后，会自动重新执行该文件，最终效果就是，你不需要再输入任何指令，只需要保存代码文件，代码就会自动重载执行。
-
- 如图
- 使用代码
- 最后你可以将上述代码写到 `boot.py` 文件之中，使得开机默认上电就能拥有 启动 `webDav` 服务 和 检查文件修改后执行 的功能。
- 代码

4. 无线编程环境

- 现在你可以使用新的开发方式，不同于过去的输入命令，只需要点击上传即可完成程序的运行。
- 注意，使用该方法编程，建议使用VS CODE + WebDAV Upload插件来运行。

7. 学习网络爬虫

1. 爬取城市天气数据

7. 软硬结合应用

1. 自定义点数的骰子

8. 专业学习之路

至此，你已经掌握了许多简单的使用方法，但这样的你还只是“知其然而不知其所以然”，因此你需要静下来，好好体会一些基础知识的学习，才能做到知其所以然者也。

1. 软件学习教程

1. 硬件学习教程