MICRO-PYTHON

使用NodeMCU ESP8266與ESP32S

INTRODUCTION-1

- MicroPython
  - designed to be capable of running on microcontrollers

  (可以在微控制器上執行的python程式語言)

- 使用樹莓派與ESP32、ESP8266 NodeMCU以micro USB連線

- 可以互動方式或上傳程式控制ESP32、ESP8266 NodeMCU

- 類似 Arduino IDE

.

INTRODUCTION-2

- Bytecode
  - when a module is imported, MicroPython compiles the code to bytecode
  - then executed by the MicroPython virtual machine (VM)
  - the bytecode is stored in RAM

- More information
  - micropython.org

INTRODUCTION - 3

Quick reference for the ESP32

不安全 | docs.micropython.org/en/latest/esp32/quickref.html

應用程式 Bookmarks 興大 其他學校 計畫 期刊 固控&控制 廠商 從 IE 匯入 建議的網站 About Us | Techno...

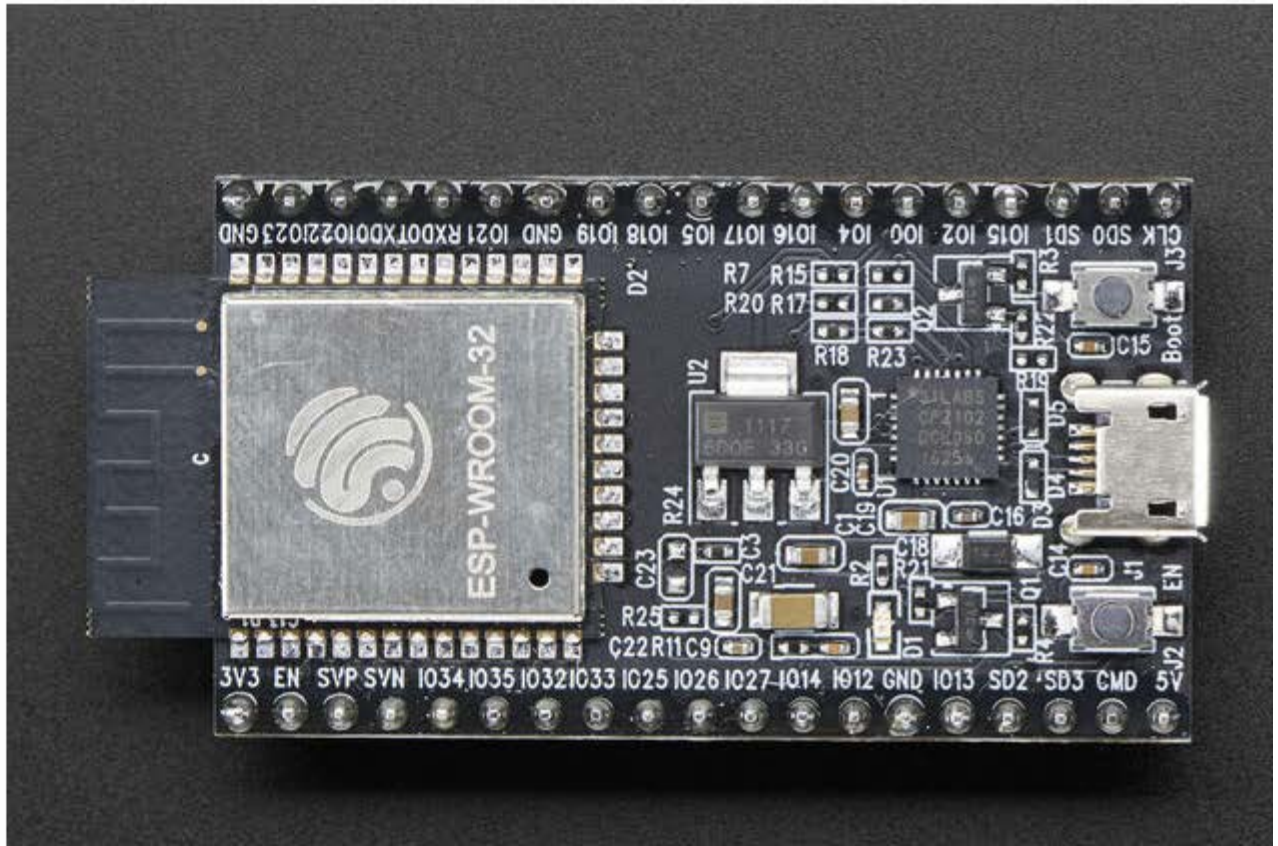
MicroPython
1.14

Search docs

- MicroPython libraries
- MicroPython language and implementation
- MicroPython differences from CPython
- MicroPython Internals
- MicroPython license information
- Quick reference for the pyboard
- Quick reference for the ESP8266
- Quick reference for the ESP32
  - General information about the ESP32 port
  - Getting started with MicroPython on the ESP32
  - Installing MicroPython
  - General board control
  - Networking

Docs » Quick reference for the ESP32 View pag

Quick reference for the ESP32



DOWNLOADS

- Micropython.org

- micropython-1.14.tar.xz

- Firmwares (版本眾多)
  - for ESP32: esp32-idf4-20210202-v1.14.bin
  - for ESP8266 NodeMCU: esp8266-20190529-v1.11.bin

INSTALL

- 樹莓派連上ESP32或ESP8266 NodeMCU：ttyUSB0或ttyUSB1(若連兩部)

- pip3 install esptool
  - A Python-based, open source, platform independent
  - utility to communicate with the ROM bootloader in Espressif ESP8266 & ESP32 series chips

- sudo pip3 install rshell
  - Remote MicroPython shell
  - a simple shell which runs on the host and uses MicroPython's raw-REPL to send python snippets to the pyboard
    - get filesystem information
    - copy files to and from MicroPython's filesystem

FIRMWARE

- 清空flash記憶體
  - esptool.py --port /dev/ttyUSB0 erase_flash

- 寫入韌體(確認檔案路徑)
  - ESP32
    - esptool.py --chip esp32 --port /dev/ttyUSB0 write_flash -z 0x1000 ~/Downloads/esp32-idf4-20210202-v1.14.bin
  - ESP8266 NodeMCU
    - esptool.py --port /dev/ttyUSB0 --baud 460800 write_flash --flash_size=detect -fm dio 0 ~/Downloads/esp8266-20190529-v1.11.bin

線上測試

- 直接下指令

- rshell –p /dev/ttyUBS0
  - ls (顯示本機RPi目錄)
  - ls /pyboard (顯示目標板ESP32目錄)

- Repl
  - read-Eval-Print Loop
  - 可與目標板互動
  - 模擬終端機

ESP獨立作業

- 測試完畢，可將程式存成main.py上傳至微控制器
- 程式上傳工具程式ampy
  - pip install adafruit-ampy
- 利用ampy上傳程式至ESP32 (使用thonny python ide，省略此步驟)
  - ampy –p /dev/ttyUSB0 put main.py
- 在微控制器上，按下RESET會先執行boot.py，接著執行main.py

NETWORK - 1

- NodeMCU ESP8266與ESP32S可以連上無線網路
- 網路模組：network
- 網路介面
  - STA_IF：使用者(client)(STA=station)
  - AP_IF：熱點(AP=access point)
  - 兩者混和

NETWORK - 2

- import network
- wlan = network.WLAN(network.STA_IF)
  - network.WLAN(network.AP_IF)
- wlan.active(True)：啟用網路介面
  - wlan.active(False)：停用網路介面
- wlan.ifconfig()：顯示IP
- wlan.connect(ssid, password)：連網

NETWORK - 3

- wlan.isconnected()：是否連上網路

- 管理 IoT聯網資料(以ssid='AACenter', password='22850946'為例)
  - 建立config.json檔案 (在repl介面執行)

```
import json
ss = {'ssid'='AACenter','password'='22850946'}
with open('config.json','w') as f:
    json.dump(ss, f)
```

NETWORK - 4

- 載入聯網資料
  - 匯入config.json檔案(可先在repl介面執行測試)

```
import json
with open('config.json') as f:
    config = json.load( f)
```

NETWORK - 4

- wlan.isconnected()：是否連上網路

- 管理 IoT聯網資料(以ssid='AACenter', password='22850946'為例)
- 建立config.json檔案 (在repl介面執行)
- import json
- ss = {'ssid'='AACenter','password'='22850946'}
- with open('config.json','w') as f:
-     json.dump(ss, f)

- 載入聯網資料