ESP8266——固件烧录与连接

【硬件准备】

- NodeMCU/WEMOS D1 mini/PiliBoard（多选一）



- USB 数据线

【操作步骤】

1. 安装 esptool 工具
2. 下载 MicroPython 固件
3. 在 NodeMCU/D1 mini 上烧录 MicroPython
4. 使用 Putty 进行连接
5. 在 NodeMCU/D1 mini 上点亮板载 LED
6. 在 PiliBoard 上进行操作

【参考】

- MicroPython 固件下载地址
  http://micropython.org/download#esp8266
- NodeMCU 通讯驱动（cp2102）
  https://cn.silabs.com/products/development-tools/software.page=0#interface
- D1 mini/PiliBoard 通讯驱动（ch340g）
  http://www.wch.cn/download/CH341SER_ZIP.html
- 烧入固件命令
  安装 esptool 工具（linux 上加上 sudo 执行）:
  ```
  pip install esptool
  ```
  清空固件（在 linux 上，使用 esptool.py，而不是 esptool）:
  ```
  esptool --port COMx erase_flash
  ```
  写入固件:
  ```
  esptool --port COMx --baud 115200 write_flash 0 esp8266-20180511-v1.9.4.bin
  ```
  （部分型号的 NodeMCU，需增加命令参数 -fm dio）

- 点亮 LED 灯

  ```
  import machine
  p2 = machine.Pin(2,machine.Pin.OUT)
  p2.value(1)     # 熄灭
  p2.value(0)     # 点亮
  ```

- MicroPython 官网

  http://www.micropython.org/

- PiliBoard 与 MicroPython

  https://www.hachina.io/docs/6807.html

ESP8266 上的 MicroPython 使用

【操作步骤】

1. REPL
2. 文件系统
3. 连接 wifi 网络
4. WebREPL
5. 常用的 REPL 操作技巧
6. 8266 上的 AP 网卡

【参考】

- 文件系统常用命令

```
import os
os.statvfs('/')                    # 查看文件系统状态
os.listdir()                       # 列出当前目录下文件
os.mkdir('xxxx')                   # 创建目录
os.rmdir('xxxx')                   # 删除目录
os.stat('test.py')                 # 列出文件状态
os.rename('test.py','test.py.bak') # 修改文件名
os.remove('test.py.bak')           # 删除文件
# 打印文件内容
f=open('boot.py','r')
f.read()
f.close()
```

- 网卡操作常用命令

```
import network
# 工作站网卡
sta_if = network.WLAN(network.STA_IF)  # 设置工作站网卡 sta_if
sta_if.active(True)                    # 开启网卡
sta_if.connect('xxx','yyy')            # 连接 wifi 网络，名称为 xxx，密码为 yyy
sta_if.active()                        # 查询网卡是否激活
sta_if.isconnected()                   # 查询网络是否连接
sta_if.ifconfig()                      # 查询网络连接信息
sta_if.active(False)                   # 关闭网卡
# AP 热点
ap_if = network.WLAN(network.AP_IF)                # 设置热点网卡 ap_if
ap_if.config(essid="abc", password="12345678")     # 设置热点与密码
ap_if.config("essid")                              # 查看设置的热点
# active()与 ifconfig()，用法与工作站网卡相同
```

- WEBRepl

  WEBRepl 开启与设置命令：`import webrepl_setup`

  WEBRepl 页面：http://micropython.org/webrepl/

- 闪烁 LED 灯

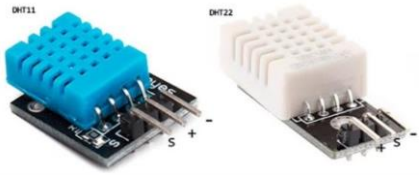
```
import machine
```

```
import time
p2 = machine.Pin(2, machine.Pin.OUT)
while True:
    print("flashing……")
    p2.value(0)
    time.sleep(1)
    p2.value(1)
    time.sleep(1)
```

连接 ESP8266 的 DHT 温湿度传感器

【硬件准备】

温湿度传感器：dht11/dht22

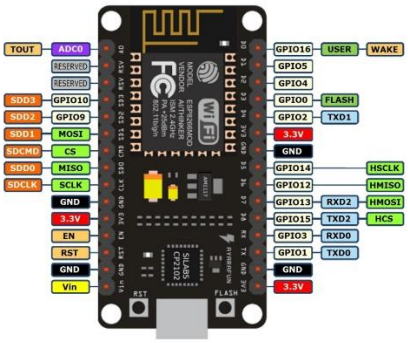


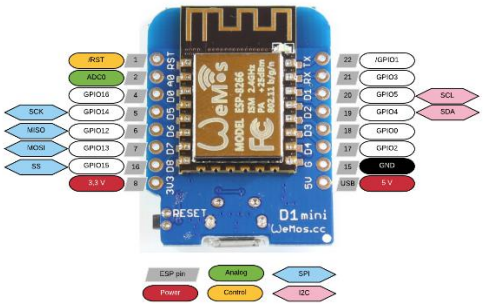
【操作步骤】

1. 硬件连接
2. 读取温湿度
3. 向 MQTT 服务器发送温度值
4. HomeAssistant 中的配置
5. 8266 中的自启动运行

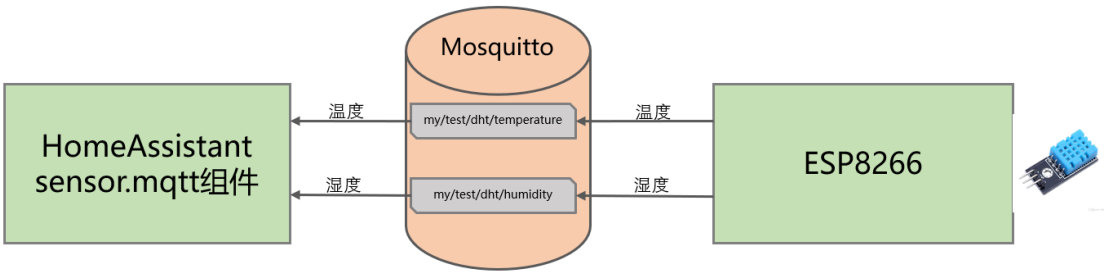
【参考】

- 开发板管脚





- mqtt 连接逻辑



- 程序

```
import dht
import machine
import time
from ubinascii import hexlify
from umqtt.simple import MQTTClient

# 初始化一个温湿度传感器，温湿度传感器连接 4 号 GPIO
DHT = dht.DHT11(machine.Pin(4))

# CLIENT_ID: 每个 mqtt 客户端有自己独立的 client_id 标识
# 无所谓是什么，但不同客户端不能相同。此处我们使用 8266 的 unique_id
client_id = hexlify(machine.unique_id()).decode()
```

```
mqtt_broker = "192.168.31.193"
mqtt_user = 'pi'
mqtt_password = 'hachina'

# 温度和湿度在 mqtt 服务器上的主题位置
TEMPERATURE_TOPIC = "my/test/dht/temperature"
HUMIDITY_TOPIC = "my/test/dht/humidity"

# 等待 15 秒，wifi 自动连接上再运行
time.sleep(15)

mqtt = MQTTClient( client_id, mqtt_broker, 1883, mqtt_user, mqtt_password )
mqtt.connect()
print("连接到服务器：{s}".format(s=mqtt_broker))

while True:
    # 不断循环，每 10 秒测量一次温度和湿度，发布到 mqtt 服务器上
    try:
        DHT.measure()
    except:
        print("No dht sensor connected to Pin(%d)"%(PIN_NO))
        break
    print("测量到温度：%d；湿度：%d"%(DHT.temperature(),DHT.humidity()))

    # mqtt 发布信息
    mqtt.publish( TEMPERATURE_TOPIC, str(DHT.temperature()).encode(), retain=True)
    mqtt.publish( HUMIDITY_TOPIC, str(DHT.humidity()).encode(), retain=True)
    time.sleep(10)
```

- HomeAssistant 中配置

```
sensor mqtt:
  - platform: mqtt
    name: "dht-T"
    state_topic: "my/test/dht/temperature"
    unit_of_measurement: "°C"
  - platform: mqtt
    name: "dht-H"
    state_topic: "my/test/dht/humidity"
    unit_of_measurement: "%"
```

ESP8266 完成各种功能

【操作步骤】

1. 从 INTERNET 同步时间
2. WEB 服务器
3. PWM 舵机控制
4. ds18b20 温度传感器
5. tsl2561 光照传感器
6. ws2812 灯带控制

【参考】

- ESP8266 功能与常用应用领域

功能	常用应用领域
网络	NTP 客户端、MQTT 客户端、WEB 服务器端/客户端
常规 GPIO 输入输出	状态量读取与输出、继电器控制
GPIO PWM	可调亮度 LED、可控硅控制、舵机控制
1-wire 总线通讯	连接 ds18b20 等各种 1-wire 总线器件
I2C 总线通讯	连接 am2320 等各种 i2c 总线器件
SPI 总线通讯	连接各种 SPI 总线器件
ADC 模拟信号输入	测量输入电压、模拟传感器连接
apa102、neopixel 驱动	各种灯带控制
中断	按钮输入
计时器	定时控制
……	……

- MicroPython 参考文档网站
  http://docs.micropython.org/en/latest/
- PiliBoard 各种用例与程序
  https://github.com/zhujisheng/piliboard/tree/master/examples
- NTP 访问

```
import ntptime
import time
ntptime.settime()
t = time.localtime()
print("当前国际标准时间：%d 年%d 月%d 日 %d:%d:%d"%(t[0],t[1],t[2],t[3],t[4],t[5]))
```

- 一个最简单的 WEB 服务器

```
response = """<!DOCTYPE html>
<html>
    <head> <title>ESP 8266</title>
    <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
    </head>
    <body><h1>欢迎来到 MicroPython 的世界</h1></body>
</html>
"""

import socket
addr = socket.getaddrinfo('0.0.0.0', 80)[0][-1]

s = socket.socket()
s.bind(addr)
s.listen(1)
```

```
print('listening on', addr)

while True:
    cl, addr = s.accept()
    print('client connected from', addr)
    cl_file = cl.makefile('rwb', 0)
    while True:
        line = cl_file.readline()
        if not line or line == b'\r\n':
            break
    cl.send(response)
    cl.close()
```

- PWM 舵机控制

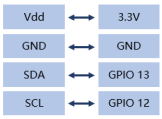

```
import machine
pin_no = 13
servo = machine.PWM(machine.Pin(pin_no), freq=50)
servo.duty(30)
```

- DS18b20 温度传感器


```
import time
import machine
import onewire, ds18x20
dat = machine.Pin(13)

ds = ds18x20.DS18X20(onewire.OneWire(dat))

roms = ds.scan()
print('找到设备：', roms)

# 循环 10 次，每次打印所有设备测量的温度
while True:
    print('温度：', end=' ')
    ds.convert_temp()
    time.sleep_ms(750)
    for rom in roms:
        print(ds.read_temp(rom), end=' ')
    print()
```

- tsl2561 光照传感器

https://github.com/adafruit/micropython-adafruit-tsl2561/blob/master/docs/tsl2561.rst




```
import tsl2561
import time
from machine import I2C, Pin
i2c = I2C(scl=Pin(12), sda=Pin(13))
sensor = tsl2561.TSL2561(i2c)
while True:
    print(sensor.read())
    time.sleep_ms(750)
```

- ws2812 灯带控制

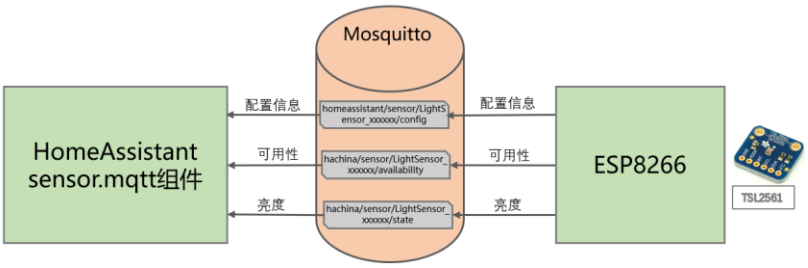

```
from machine import Pin
```

```
from neopixel import NeoPixel

leds_num = 30
dp = Pin(13, Pin.OUT)
np = NeoPixel(dp, leds_num)
for i in range(leds_num):
    np[i] = (0, 0, 0)
    np.write()

# 按红、绿、蓝、白显示灯带
color = [(255,0,0),(0,255,0),(0,0,255),(255,255,255)]    #红、绿、蓝、白
for i in range(0,np.n):
    np[i] = color[i%4]
np.write()
```

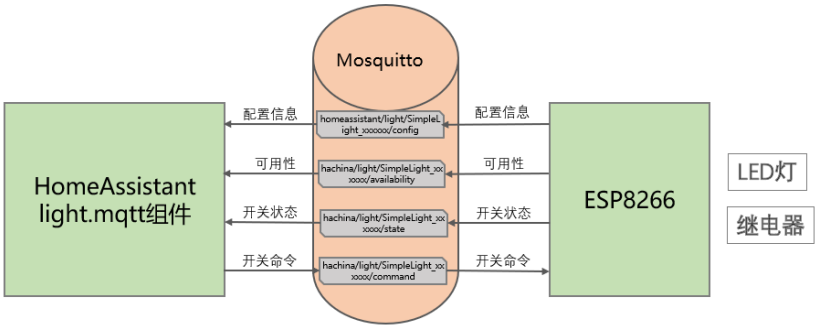
典型样例讲解：光照传感器与智能灯

【操作步骤】

1. 光照传感器

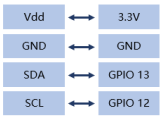


2. 智能灯



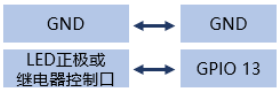
【参考】

- 光照传感器：样例 example_17_5_1.zip



文件	说明	备注
config.json	wifi 和 mqtt 的配置信息	你需要修改为你自己的环境
main.py	启动时自动运行的文件	也可以不上传这个文件，手工运行其中命令
tsl2561.py	tsl2561 驱动程序	如果使用其它传感器，不需要此文件
sensor_tsl2561.py	主程序	可以修改 GPIO 口为你实际的连接口；连接其它传感器，修改对应硬件操作部分程序

- 智能灯：样例 example_17_5_2.zip



文件	说明	备注
config.json	同上	
main.py	同上	
StateMQTTClient	基于 umqtt.simple 的一个自己实现的 mqtt 类	其中增加了对 ping 返回的判断，并增加了是否连接的状态
light_gpio.py	主程序	可以修改 GPIO 口为你实际的连接口

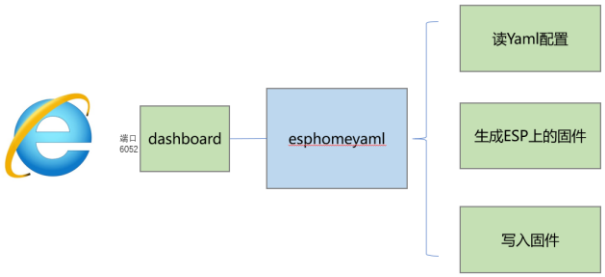
ESPHome——不编程，集成 ESP8266

【操作步骤】

1. 安装 esphomeyaml
2. 打开 web 配置界面
3. 配置与上传
4. 配置一些常见设备

【参考】

- ESPHome 的功能



- ESPHome 官网
  https://esphomelib.com/
- 相关命令
  安装：sudo pip2 install esphome
  打开 web 端配置：esphome esphome_config/ dashboard
- ESPHome 组件配置
  https://esphome.io/index.html#core-components
- 板载 GPIO2 号口 LED 的配置

```
output:
  - platform: esp8266_pwm
    pin: GPIO2
    frequency: 200 Hz
    id: led_onboard
    inverted: True

light:
  - platform: monochromatic
    name: "LED On NodeMCU Board"
    output: led_onboard
```

- 连接多个外围设备的配置样例


```
output:
  # 输出，板载 LED
  - platform: esp8266_pwm
    pin: GPIO2
    frequency: 200 Hz
    id: led_onboard
    inverted: True
  # 输出，外接 LED
  - platform: gpio
    pin: GPIO5
    id: output_gpio5
```

```
# ds18b20
dallas:
  - pin: GPIO14

i2c:
  sda: 13
  scl: 12

light:
  #  板载 LED
  - platform: monochromatic
    name: "LED On NodeMCU Board"
    output: led_onboard
  #  外接 LED
  - platform: binary
    name: "LED Link to GPIO5"
    output: output_gpio5

sensor:
  # DHT22 温湿度传感器
  - platform: dht
    pin: GPIO4
    model: DHT22
    temperature:
      name: "Temperature DHT22"
    humidity:
      name: "Humidity DHT22"
    update_interval: 30s
  # DS18b20 温度传感器
  - platform: dallas
    index: 0
    name: "Temperature ds18b20"
  # tsl2561 光亮度传感器
  - platform: tsl2561
    name: "Lux TSL2561"
    update_interval: 30s
```