做一個小項目: 物聯網屏蔽

https://randomnerdtutorials.com/esp32-iot-shield-pcb-dashboard/

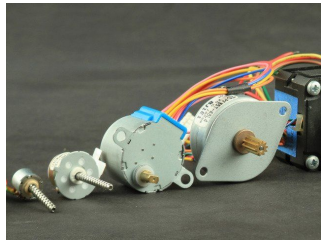
附加軟件庫: DHT Sensors, ESPUI, ESP32servo

什麼是物聯網屏蔽?

- 有輸出 - LED/蜂鳴器/

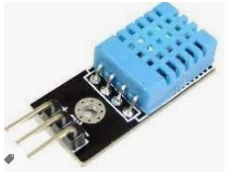
  直流馬達/伺服馬達/步進電機



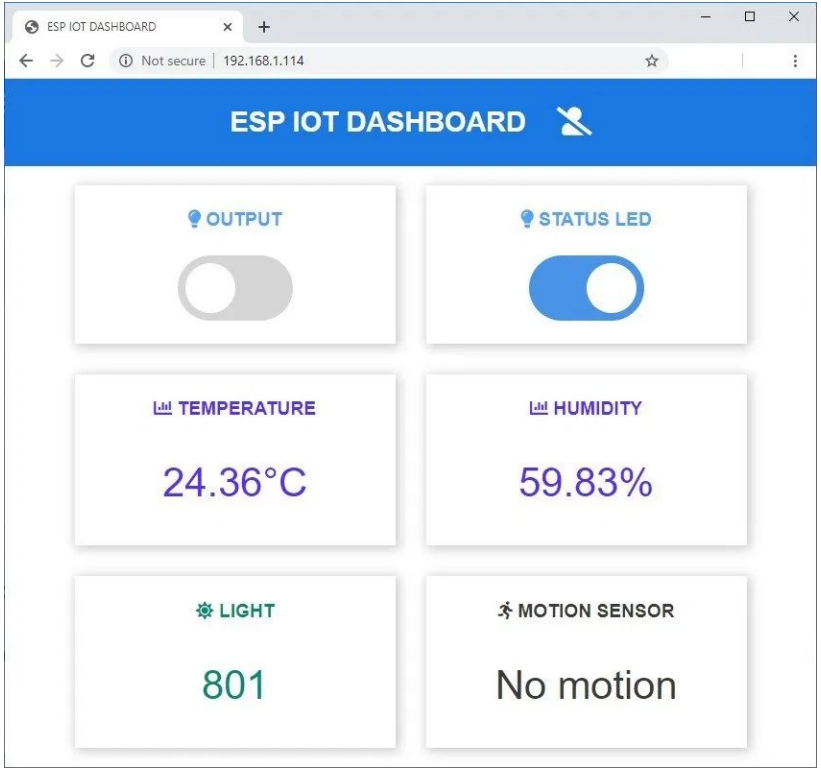
- 有輸入 - 按鈕/觸摸輸入


- 感應器 - 濕度,溫度計/光量計




什麼是物聯網屏蔽?

- 界面 - 網頁儀表板



物聯網屏蔽怎麼做?

- 使用Arduino開發環境為ESP32編寫軟件
  - 輸入/輸出 -
    - 輸入 - 按鈕/輕觸開關/直流馬達/伺服馬達/步進電機/無刷電機
    - 輸出 - LED/
  - 感應器控制
  - 網頁界面
- 電子設計
  - 什麼是電路原理圖
  - 印刷電路板製作
  - 電子設計自動化工具
  - 如何焊接

電機類型

- 直流馬達
- 伺服馬達
- 步進電機
- 無刷電機

伺服馬達

伺服馬達的構造:

- 直流馬達
- 控制電路模組
- 齒輪組
- 可變電阻器
- 軸柄

圖 14-1　伺服馬達構造圖

伺服馬達

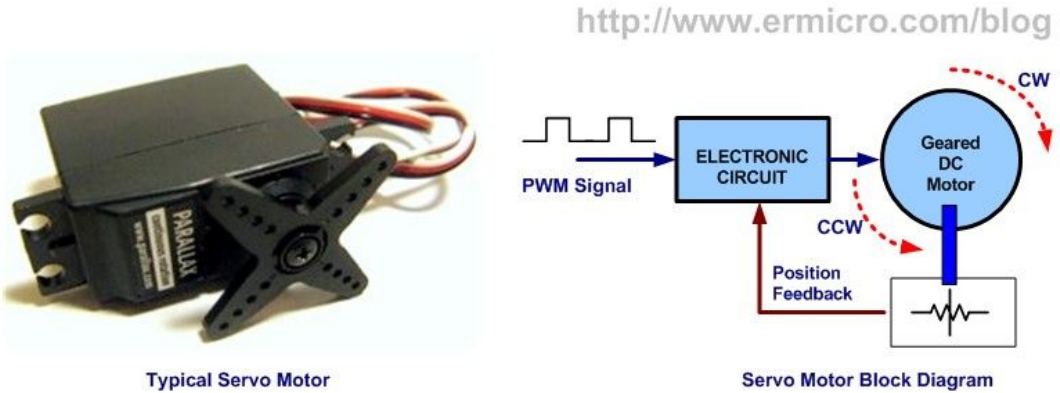
特性:

- 透過訊號來控制直流馬達運轉
- 經減速齒輪組後，轉換成適當的轉速來控制軸柄的停止角度
- 同時也提供更高的轉矩(扭轉力)
- 依旋轉角度來分
- 旋轉角度0 到180 度
- 定位型及能連續轉動的連續旋轉型;從外觀上是無法辨別其型式的

伺服馬達

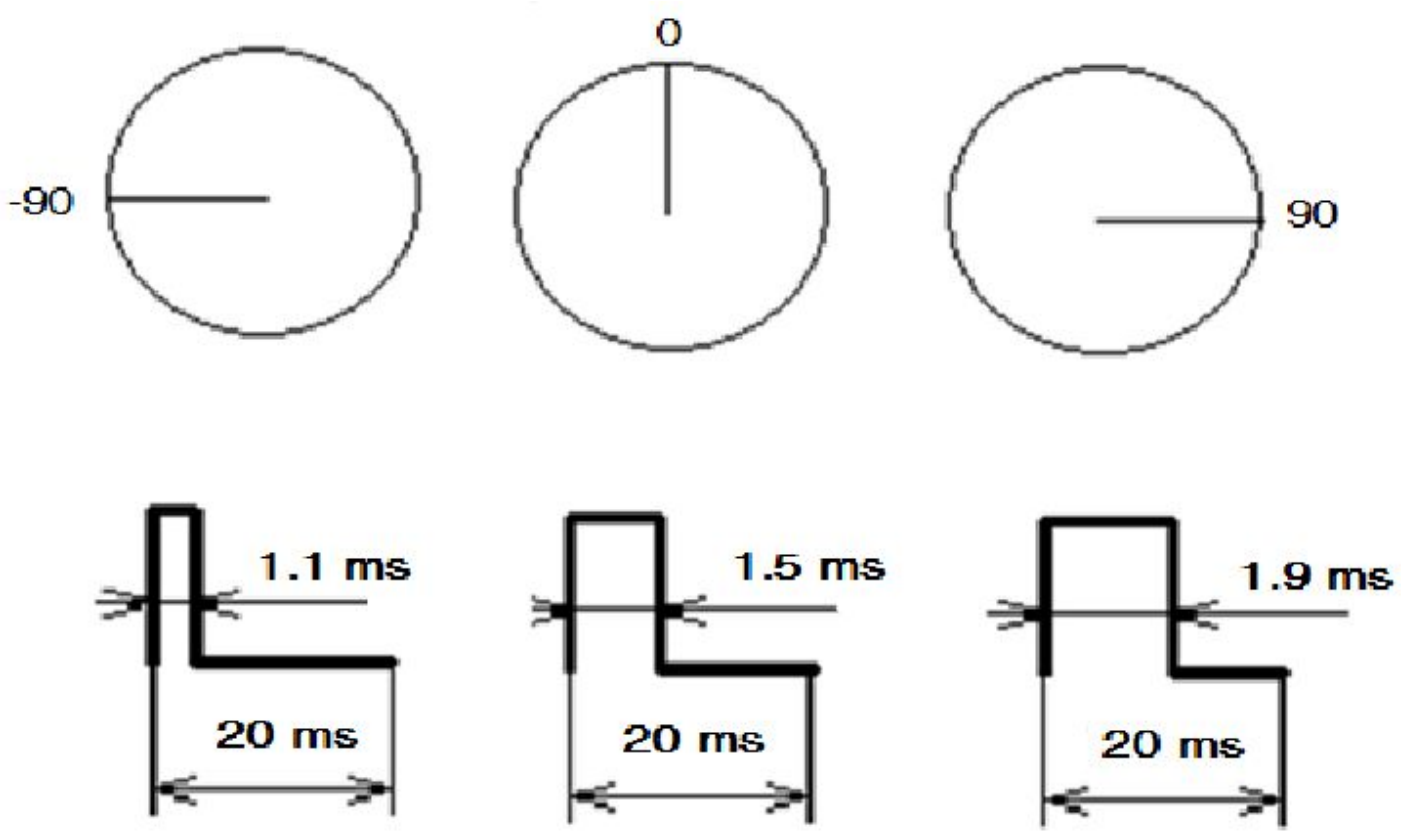
原理:

- 伺服馬達的轉動角度是藉由調製PWM(脈衝寬度)信號的占空比來實現
- 經由內部的控制電路,獲得的直流偏壓來驅動馬達開始轉動
- 並透過減速齒輪將動力傳至擺臂
- 同時與可變電阻器上的電壓做比較,來判斷是否已經到達定位



伺服馬達

PWM(脈衝寬度)信號:



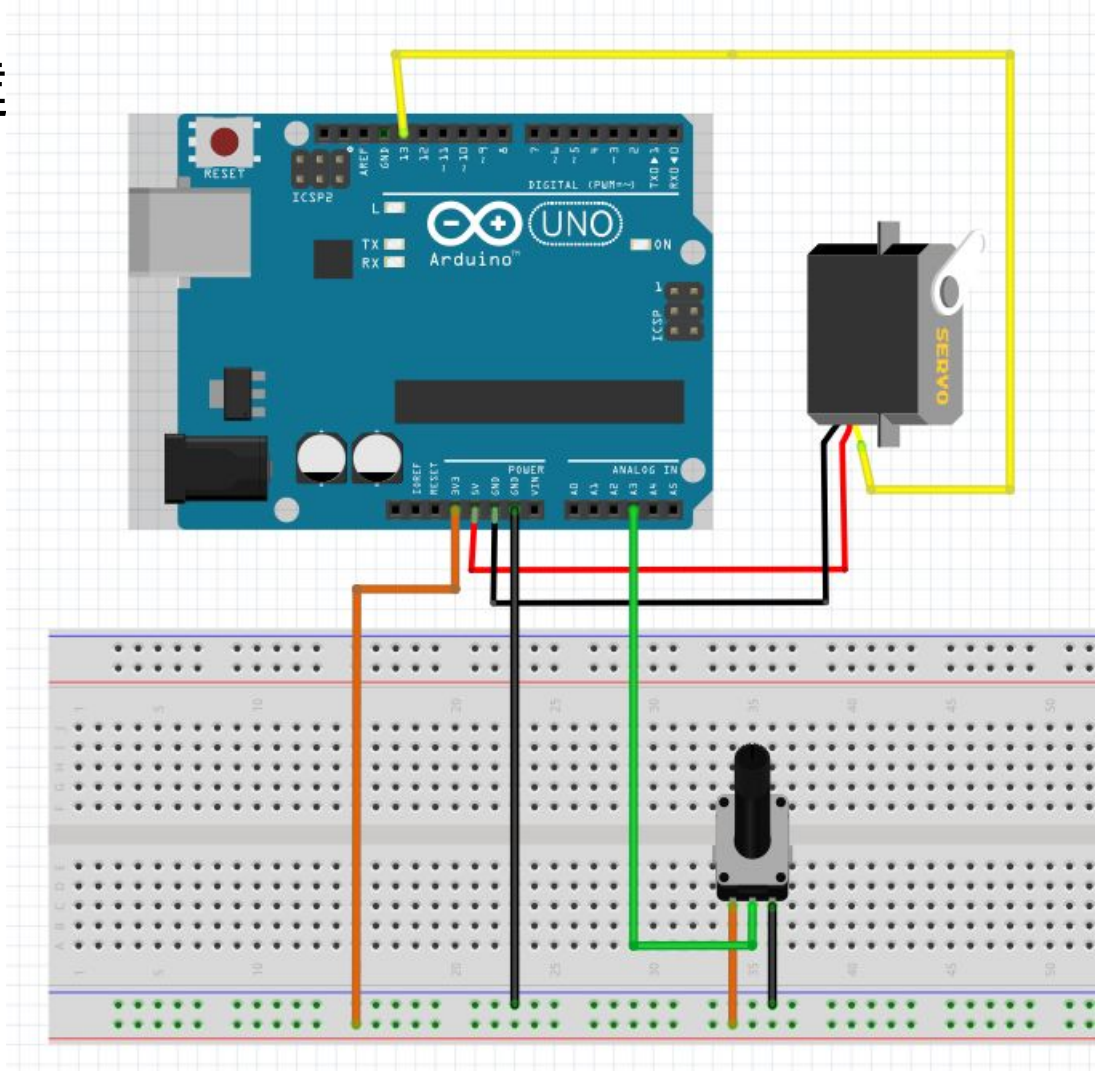
伺服馬達

脈衝寬度		軸柄位置
0.8 毫秒	安全區	CW（正轉）
0.9 毫秒	+60 度 ±10°	CW（正轉）
1.5 毫秒	0 度	中心位置
2.1 毫秒	60 度 ±10°	CCW（反轉）
2.2 毫秒	安全區	CCW（反轉）

SG-90伺服馬達

- 重量:9g
- 尺寸:23*12.2*29mm
- 工作電壓:4.8V
- 轉矩:1.8kg-cm，當工作電壓為4.8V 時
- 運轉速度:0.1 秒/60 度，當工作電壓為4.8V 時
- 脈衝寬度範圍:500~2400μs
- 死頻帶寬度（dead band width）:10μ

範例 伺服馬達


RESET
ICSP2
UNO
Arduino
TX
RX
ON
ICSP
DIGITAL (PWM~)
POWER
ANALOG IN
SERVO

程式碼解

```
// Include the ESP32 Arduino Servo Library instead of the original Arduino Servo Library
#include <ESP32Servo.h>

Servo myservo;  // create servo object to control a servo

// Possible PWM GPIO pins on the ESP32: 0(used by on-board button),2,4,5(used by on-board
LED),12-19,21-23,25-27,32-33

int servoPin = 18;      // GPIO pin used to connect the servo control (digital out)
// Possible ADC pins on the ESP32: 0,2,4,12-15,32-39; 34-39 are recommended for analog input

int potPin = 34;        // GPIO pin used to connect the potentiometer (analog in)

int ADC_Max = 4096;     // This is the default ADC max value on the ESP32 (12 bit ADC width);
                        // this width can be set (in low-level oode) from 9-12 bits, for a
                        // a range of max values of 512-4096

int val;    // variable to read the value from the analog pin
```

程式碼解

```
void setup()
{
        // Allow allocation of all timers
  Serial.begin(115200);
        ESP32PWM::allocateTimer(0);
        ESP32PWM::allocateTimer(1);
        ESP32PWM::allocateTimer(2);
        ESP32PWM::allocateTimer(3);
  myservo.setPeriodHertz(50); // Standard 50hz servo
  myservo.attach(servoPin, 500, 2400);   // attaches the servo on pin 18 to the servo object
                                         // using SG90 servo min/max of 500us and 2400us
                                         // for MG995 large servo, use 1000us and 2000us,
                                         // which are the defaults, so this line could be
                                          // "myservo.attach(servoPin);"
}
```

程式碼解

```
void loop() {
  val = analogRead(potPin);            // read the value of the potentiometer (value between 0 and 1023)

  val = map(val, 0, ADC_Max, 0, 180);     // scale it to use it with the servo (value between 0 and 180)
  myservo.write(val);                     // set the servo position according to the scaled value
    Serial.println(val);
  delay(200);                             // wait for the servo to get there
}
```

印刷電路板製作