智慧家庭:PM2.5 空氣感測器(上網篇：連上 MQTT)

五月, 2016
文\曹永忠

本篇是接續上篇文章『智慧家庭：PM2.5 空氣感測器(上網篇：啟動網路校時功能)』,已經可以讓 PM2.5 空氣感測器(曹永忠, 2016a, 2016b, 2016c, 2016d, 2016e, 2016f)可以連上網路，並啟動網路校時功能。

但是如何雲端儲存資料，並且可以視覺化資訊，這就是關鍵性的問題，所以作者與 LASS 社群合作，將 PM2.5 空氣感測器的資料可以連上 LASS 平台(Hsu, 2016)，並且可以視覺化呈現。

LASS 是甚麼

網路強人『哈爸[1]』在當紅網路媒體：MAKEPRO 中，開啟一個『【開源公益專案】LASS 環境感測網路系統』，網址：http://makerpro.cc/2015/09/projectplus-lass/ ，FB 官網為：LASS-開源公益的環境感測器網路(https://www.facebook.com/groups/1607718702812067/)，提倡開源和公益的環境感測器網路系統(曹永忠, 許智誠, & 蔡英德, 2015a, 2015b)。

網路強人『哈爸』在當紅網路媒體：MAKEPRO[2]中，提出 LASS[3]是一套具備環

[1]哈爸：現職為晶片設計公司架構師，熱愛 Linux，熟悉嵌入式系統設計、通訊系統以及軟體設計。曾任 Linux 認證講師，Java 認證講師。希望集結台灣 PRO Maker 社群力量，共創讓世人注目的開放硬體計畫，並發展出創新的 Maker 經濟模式。現兼任 MakerPRO 社群顧問，主持【哈爸陪你問】online QA、LASS 開源環境監測網路專案、Maker 零件包計畫等。

[2] MAKEPRO：http://makerpro.cc/

[3] LASS（Location Aware Sensor System），顧名思義可知，這是一套「環境感測器網路系統」，而且是開源和公益的定位

境感測器的網路系統架構，而且要讓 Maker 有能力自製所需要、且能與系統相容的感測器套件，再將感測到的資料提供給任何使用者。為此，我們需要提供一套完成度夠高的系統雛型，以賦予所有使用者俱備監測全部感測器的能力。

如下圖所示，以下網路強人『哈爸』在當紅網路媒體：MAKEPRO 中，提出的上層系統架構及感測端設備架構。

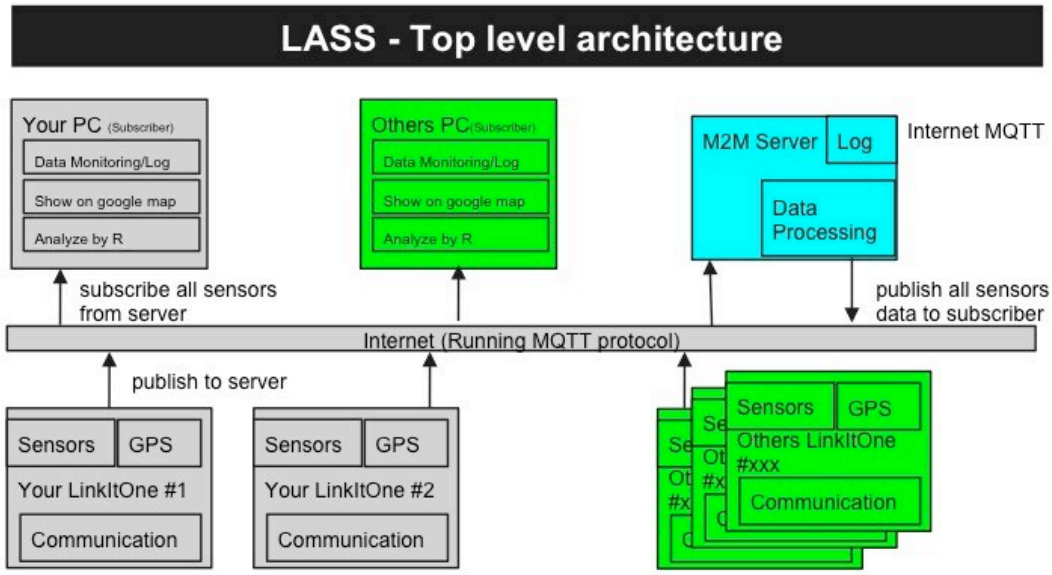

圖 1 LASS 上層系統架構示意圖

參考資料：哈爸：【開源公益專案】LASS 環境感測網路系統 (http://makerpro.cc/2015/09/projectplus-lass/)

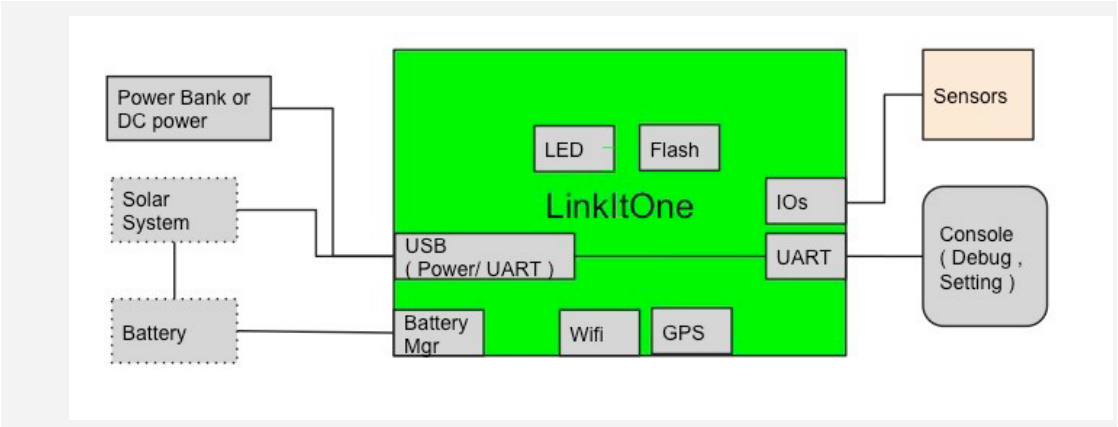

圖 2 LASS 感測設備端架構

參考資料：哈爸：【開源公益專案】LASS 環境感測網路系統 (http://makerpro.cc/2015/09/projectplus-lass/)

如下表所示，網路強人『哈爸』在當紅網路媒體：MAKEPRO 中，提出三層網路系統架構，為了讓不同開發版、感測器、傳輸介面等等可以依據各自規範，各自運作並整合。如下表所示，網路強人『哈爸』說明其網路系統架構可分三層，分別是感知層、網路層和應用層，各層運作方式及特色。

表 1 系統架構運作方式與特色

三層架構	系統運作方式	特色
感知層	● 可支援各種感測器 ● 軟體架構設計成可輕易經由簡單的修改達成客製化要求 ● 已整合 Dust/UV/sound 感測器	● 持續支援更多感測器 ● 方便使用者建構自己需要的感測器
網路層	● 使用 WiFi 連上 Internet(AWS Linux) ● 使用分散式 MQTT 協定 ● Server 使用 Mosquitto ● 支援離線持續量測，自動等待連線	● 分散式，降低網路負載 ● 設備跟設備之間邏輯上可以直接對話
應用層	● 伺服器留存 Log，提供下載 ● 當場資料轉傳給所有監測的軟體 ● 動態感測器圖 ● 完整匯出 CSV, KML, R ● 範例支援 Google Map 顯示 ● 支援多型態的警報機制 ● 支援手機 GUI 即時監測(By Blynk) ● 資料架構支援異質系統共存	● 資料為長久可儲存的簡單架構 ● 完整保留資料的主控權給使用者 ● 資料型態方便進階分析 (R) ● 雲端精簡化，隨時可改用自己的 ● 創意的夥伴警報機制 ● 針對異質系統共存設計，讓不同研究人員可以無痛分享彼此資料

參考資料：哈爸：【開源公益專案】LASS 環境感測網路系統
(http://makerpro.cc/2015/09/projectplus-lass/)

表 2 LASS 警報架構

警報型態	使用情境	簡易可行的使用方式（舉例）
自我偵測 Self detect mode	感測器值超過限定範圍	● 發出聲響 ● 改變燈號 ...
中控模式 Central mode	區域感測狀況警報：根據統計資訊智能判斷，於管理前端操作，可以隨時加強演算法	當半小時內平均超標，或附近區域一起發生異常，可發送 Email, 簡訊...
夥伴模式 Partner mode	監測對象超過限定範圍，需要通知的對象得到警報	老家下雨，桌上點燈 重要觀測對象警報

參考資料：哈爸：【開源公益專案】LASS 環境感測網路系統
(http://makerpro.cc/2015/09/projectplus-lass/)

透過 MQTT 連上 LASS

所以我們鑒於 LASS 格式
(https://lass.hackpad.com/LASS-Data-Platform-tYTI2roiKW2)，由下圖所示，我們將感測器的資料讀出後，透過 internet 網路連接，連到 gpssensor.ddns.net，這是 LASS 的 MQTT 伺服器，將資料送到雲端。

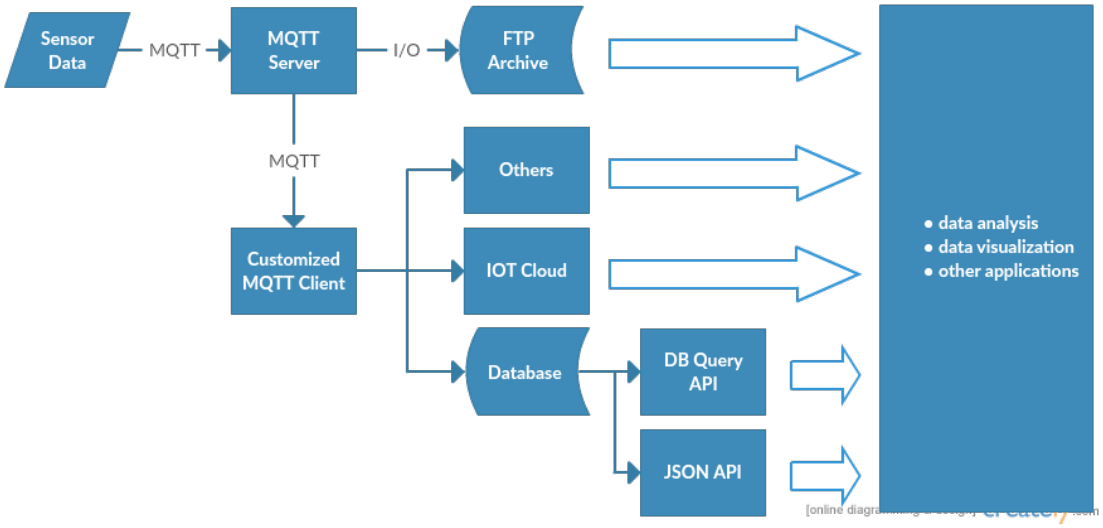

圖 3 LASS 資料流程圖

資料出處：lass ：

https://lass.hackpad.com/LASS-Data-Platform-tYTI2roiKW2

我們先讓 PM2.5 空氣感測器可以連上網路，並簡單透過網頁方式取得們將下表之整合空氣懸浮粒子感測器測試程式一攥寫好之後，編譯完成後上傳到 Ameba 開發板，

表 3 整合空氣懸浮粒子感測器測試程式一

整合空氣懸浮粒子感測器測試程式一(PMS3003AirQualityV52)

```
/*
 This example demonstrate how to read pm2.5 value on PMS 3003 air condition sensor

 PMS 3003 pin map is as follow:
    PIN1  :VCC, connect to 5V
    PIN2  :GND
    PIN3  :SET, 0:Standby mode, 1:operating mode
    PIN4  :RXD :Serial RX
    PIN5  :TXD :Serial TX
    PIN6  :RESET
    PIN7  :NC
    PIN8  :NC

 In this example, we only use Serial to get PM 2.5 value.

 The circuit:
 * RX is digital pin 0 (connect to TX of PMS 3003)
 * TX is digital pin 1 (connect to RX of PMS 3003)

 */
#define turnon HIGH
#define turnoff LOW
#define DHTSensorPin 8
#define ParticleSensorLed 9
#define InternetLed 10
#define AccessLed 11


#include "PMType.h"
```

整合空氣懸浮粒子感測器測試程式一(PMS3003AirQualityV52)

```
#include <WiFi.h>
#include <PubSubClient.h>
#include <WiFiUdp.h>



#include <Wire.h>  // Arduino IDE 內建
// LCD I2C Library，從這裡可以下載：
// https://bitbucket.org/fmalpartida/new-liquidcrystal/downloads

#include "RTClib.h"
RTC_DS1307 RTC;
 DateTime nowT;
// DateTime nowT = RTC.now();
#include <LiquidCrystal_I2C.h>
#include <SoftwareSerial.h>

uint8_t MacData[6];

SoftwareSerial mySerial(0, 1); // RX, TX
char ssid[] = "TSAO";      // your network SSID (name)
char pass[] = "TSAO1234";     // your network password
int keyIndex = 0;               // your network key Index number
(needed only for WEP)

char gps_lat[] = "23.954710";  // device's gps latitude 清心福全(中
正店) 510 彰化縣員林市中正路 254 號
char gps_lon[] = "120.574482"; // device's gps longitude 清心福全(中
正店) 510 彰化縣員林市中正路 254 號

char server[] = "gpssensor.ddns.net"; // the MQTT server of LASS

#define MAX_CLIENT_ID_LEN 10
#define MAX_TOPIC_LEN     50
char clientId[MAX_CLIENT_ID_LEN];
char outTopic[MAX_TOPIC_LEN];
```

整合空氣懸浮粒子感測器測試程式一(PMS3003AirQualityV52)

```
    WiFiClient wifiClient;
    PubSubClient client(wifiClient);
    IPAddress  Meip ,Megateway ,Mesubnet ;
    String MacAddress ;
    int status = WL_IDLE_STATUS;
    boolean ParticleSensorStatus = true ;
    WiFiUDP Udp;

    const char ntpServer[] = "pool.ntp.org";
    const long timeZoneOffset = 28800L;
    const int NTP_PACKET_SIZE = 48; // NTP time stamp is in the first 48
bytes of the message
    const byte nptSendPacket[ NTP_PACKET_SIZE] = {
      0xE3, 0x00, 0x06, 0xEC, 0x00, 0x00, 0x00, 0x00,  0x00, 0x00, 0x00,
0x00, 0x31, 0x4E, 0x31, 0x34,
      0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,  0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00,
      0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,  0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00
    };
    byte ntpRecvBuffer[ NTP_PACKET_SIZE ];

    #define LEAP_YEAR(Y)     ( ((1970+Y)>0) && !((1970+Y)%4) &&
( ((1970+Y)%100) || !((1970+Y)%400) ) )
    static  const uint8_t
monthDays[]={31,28,31,30,31,30,31,31,30,31,30,31}; // API starts months
from 1, this array starts from 0
    uint32_t epochSystem = 0; // timestamp of system boot up



    #define pmsDataLen 24
    uint8_t buf[pmsDataLen];
    int idx = 0;
    int pm25 = 0;
    uint16_t PM01Value=0;          //define PM1.0 value of the air
detector module
    uint16_t PM2_5Value=0;         //define PM2.5 value of the air
```

整合空氣懸浮粒子感測器測試程式一(PMS3003AirQualityV52)

```
detector module
    uint16_t PM10Value=0;         //define PM10 value of the air
detector module
      int NDPyear, NDPmonth, NDPday, NDPhour, NDPminute, NDPsecond;
      unsigned long epoch  ;
    LiquidCrystal_I2C lcd(0x27, 2, 1, 0, 4, 5, 6, 7, 3, POSITIVE);  //
設定 LCD I2C 位址



    void setup() {
      initPins() ;
      Serial.begin(9600);

      mySerial.begin(9600); // PMS 3003 UART has baud rate 9600
      lcd.begin(20, 4);      // 初始化 LCD，一行 20 的字元，共 4 行，
預設開啟背光
           lcd.backlight(); // 開啟背光
         //  while(!Serial) ;
         initRTC() ;

      MacAddress = GetWifiMac() ;
      ShowMac() ;
        initializeWiFi();
      retrieveNtpTime();
      ShowDateTime() ;
      showLed() ;
      ShowInternetStatus() ;

      initializeMQTT();

    }

    void loop() { // run over and over
        ShowDateTime() ;
      showLed() ;
      idx = 0;
```

整合空氣懸浮粒子感測器測試程式一(PMS3003AirQualityV52)

```
  memset(buf, 0, pmsDataLen);

  while (mySerial.available())
    {
      buf[idx++] = mySerial.read();
    }

  // check if data header is correct
  if (buf[0] == 0x42 && buf[1] == 0x4d)
      {
        pm25 = ( buf[12] << 8 ) | buf[13];
        Serial.print("pm2.5: ");
        Serial.print(pm25);
        Serial.println(" ug/m3");
        ShowPM(pm25) ;
      }
        if (client.connected())
        {
          reconnectMQTT();
         }
        client.loop();

       delay(60000); // delay 1 minute for next measurement
/*
  int checkSum=checkValue(buf,pmsDataLen);
  if(pmsDataLen&&checkSum)
  {
    PM01Value=transmitPM01(buf);
    PM2_5Value=transmitPM2_5(buf);
    PM10Value=transmitPM10(buf);
  }
    static unsigned long OledTimer=millis();
  if (millis() - OledTimer >=1000)
  {
    OledTimer=millis();
      Serial.print("PM1.0: ");
      Serial.print(PM01Value);
```

整合空氣懸浮粒子感測器測試程式一(PMS3003AirQualityV52)

```
        Serial.println("  ug/m3");
        Serial.print("PM2.5: ");
        Serial.print(PM2_5Value);
        Serial.println("  ug/m3");
        Serial.print("PM10:  "); //send PM1.0 data to bluetooth
        Serial.print(PM10Value);
        Serial.println("ug/m3");
    }
*/

}

void ShowMac()
{
     lcd.setCursor(0, 0); // 設定游標位置在第一行行首
     lcd.print("MAC:");
     lcd.print(MacAddress);

}

void ShowInternetStatus()
{
     lcd.setCursor(0, 1); // 設定游標位置
        if (WiFi.status())
          {
               Meip = WiFi.localIP();
               lcd.print("@:");
               lcd.print(Meip);
               digitalWrite(InternetLed,turnon) ;
          }
          else
                    {
               lcd.print("DisConnected:");
              digitalWrite(InternetLed,turnoff) ;
          }

}
```

整合空氣懸浮粒子感測器測試程式一(PMS3003AirQualityV52)

```
    void ShowPM(int pp25)
    {
        lcd.setCursor(0, 3); // 設定游標位置在第一行行首
         lcd.print("  PM2.5:");
         lcd.print(pp25);

    }

    void ShowDateTime()
    {
        getCurrentTime(epoch, &NDPyear, &NDPmonth, &NDPday, &NDPhour,
&NDPminute, &NDPsecond);
        lcd.setCursor(0, 2); // 設定游標位置在第一行行首
         lcd.print(StrDate());
        lcd.setCursor(11, 2); // 設定游標位置在第一行行首
         lcd.print(StrTime());
       //  lcd.print();

    }

    String  StrDate() {
      String ttt ;
    //nowT  = now;
    // ttt = print4digits(nowT.year()) + "/" +
print2digits(nowT.month()) + "/" + print2digits(nowT.day()) ;
     ttt = print4digits(NDPyear) + "/" + print2digits(NDPmonth) + "/" +
print2digits(NDPday) ;
      return ttt ;
    }

    String  StrTime() {
      String ttt ;
     // nowT  = RTC.now();
     // ttt = print2digits(nowT.hour()) + ":" +
print2digits(nowT.minute()) + ":" + print2digits(nowT.second()) ;
        ttt = print2digits(NDPhour) + ":" + print2digits(NDPminute) + ":"
```

整合空氣懸浮粒子感測器測試程式一(PMS3003AirQualityV52)

```
+ print2digits(NDPsecond) ;
    return ttt ;
    }
    String GetWifiMac()
    {
       String tt ;
        String t1,t2,t3,t4,t5,t6 ;
        WiFi.status();    //this method must be used for get MAC
      WiFi.macAddress(MacData);

      Serial.print("Mac:");
       Serial.print(MacData[0],HEX) ;
       Serial.print("/");
       Serial.print(MacData[1],HEX) ;
       Serial.print("/");
       Serial.print(MacData[2],HEX) ;
       Serial.print("/");
       Serial.print(MacData[3],HEX) ;
       Serial.print("/");
       Serial.print(MacData[4],HEX) ;
       Serial.print("/");
       Serial.print(MacData[5],HEX) ;
       Serial.print("~");

        t1 = print2HEX((int)MacData[0]);
        t2 = print2HEX((int)MacData[1]);
        t3 = print2HEX((int)MacData[2]);
        t4 = print2HEX((int)MacData[3]);
        t5 = print2HEX((int)MacData[4]);
        t6 = print2HEX((int)MacData[5]);
     tt = (t1+t2+t3+t4+t5+t6) ;
    Serial.print(tt);
    Serial.print("\n");

      return tt ;
    }
    String  print2HEX(int number) {
```

整合空氣懸浮粒子感測器測試程式一(PMS3003AirQualityV52)

```
  String ttt ;
  if (number >= 0 && number < 16)
  {
    ttt = String("0") + String(number,HEX);
  }
  else
  {
      ttt = String(number,HEX);
  }
  return ttt ;
}
String  print2digits(int number) {
  String ttt ;
  if (number >= 0 && number < 10)
  {
    ttt = String("0") + String(number);
  }
  else
  {
    ttt = String(number);
  }
  return ttt ;
}

String  print4digits(int number) {
  String ttt ;
  ttt = String(number);
  return ttt ;
}




void callback(char* topic, byte* payload, unsigned int length) {
  Serial.print("Message arrived [");
  Serial.print(topic);
  Serial.print("] ");
  for (int i=0;i<length;i++) {
```

```
        Serial.print((char)payload[i]);
      }
      Serial.println();
    }

    // send an NTP request to the time server at the given address
    void retrieveNtpTime() {
      Serial.println("Send NTP packet");

      Udp.beginPacket(ntpServer, 123); //NTP requests are to port 123
      Udp.write(nptSendPacket, NTP_PACKET_SIZE);
      Udp.endPacket();

      if(Udp.parsePacket()) {
        Serial.println("NTP packet received");
        Udp.read(ntpRecvBuffer, NTP_PACKET_SIZE); // read the packet
into the buffer

        unsigned long highWord = word(ntpRecvBuffer[40],
ntpRecvBuffer[41]);
        unsigned long lowWord = word(ntpRecvBuffer[42],
ntpRecvBuffer[43]);
        unsigned long secsSince1900 = highWord << 16 | lowWord;
        const unsigned long seventyYears = 2208988800UL;
         epoch = secsSince1900 - seventyYears + timeZoneOffset ;

        epochSystem = epoch - millis() / 1000;
      }
    }

    void getCurrentTime(unsigned long epoch, int *year, int *month, int
*day, int *hour, int *minute, int *second) {
      int tempDay = 0;

      *hour = (epoch  % 86400L) / 3600;
      *minute = (epoch  % 3600) / 60;
      *second = epoch % 60;
```

整合空氣懸浮粒子感測器測試程式一(PMS3003AirQualityV52)

```
  *year = 1970;
  *month = 0;
  *day = epoch / 86400;

  for (*year = 1970; ; (*year)++) {
    if (tempDay + (LEAP_YEAR(*year) ? 366 : 365) > *day) {
      break;
    } else {
      tempDay += (LEAP_YEAR(*year) ? 366 : 365);
    }
  }
  tempDay = *day - tempDay; // the days left in a year
  for ((*month) = 0; (*month) < 12; (*month)++) {
    if ((*month) == 1) {
      if (LEAP_YEAR(*year)) {
        if (tempDay - 29 < 0) {
          break;
        } else {
          tempDay -= 29;
        }
      } else {
        if (tempDay - 28 < 0) {
          break;
        } else {
          tempDay -= 28;
        }
      }
    } else {
      if (tempDay - monthDays[(*month)] < 0) {
        break;
      } else {
        tempDay -= monthDays[(*month)];
      }
    }
  }
  (*month)++;
```

整合空氣懸浮粒子感測器測試程式一(PMS3003AirQualityV52)

```
  *day = tempDay+2; // one for base 1, one for current day
}

void reconnectMQTT() {
  // Loop until we're reconnected
  char payload[300];

  unsigned long epoch = epochSystem + millis() / 1000;

  getCurrentTime(epoch, &NDPyear, &NDPmonth, &NDPday, &NDPhour,
&NDPminute, &NDPsecond);
  digitalWrite(AccessLed,turnon) ;

  while (!client.connected()) {
    Serial.print("Attempting MQTT connection...");
    // Attempt to connect
    if (client.connect(clientId)) {
      Serial.println("connected");

      sprintf(payload,
"|ver_format=3|fmt_opt=1|app=Pm25Ameba|ver_app=0.0.1|device_id=%s|tic
k=%d|date=%4d-%02d-%02d|time=%02d:%02d:%02d|device=Ameba|s_d0=%d|gps_
lat=%s|gps_lon=%s|gps_fix=1|gps_num=9|gps_alt=2",
        clientId,
        millis(),
        NDPyear, NDPmonth, NDPday,
        NDPhour, NDPminute, NDPsecond,
        pm25,
        gps_lat, gps_lon
      );

      // Once connected, publish an announcement...
      client.publish(outTopic, payload);
    } else {
      Serial.print("failed, rc=");
      Serial.print(client.state());
      Serial.println(" try again in 5 seconds");
```

整合空氣懸浮粒子感測器測試程式一(PMS3003AirQualityV52)

```
      // Wait 5 seconds before retrying
      delay(5000);
    }
  }
  digitalWrite(AccessLed,turnoff) ;
}

void retrievePM25Value() {
  int idx;
  bool hasPm25Value = false;
  int timeout = 200;
  while (!hasPm25Value) {
    idx = 0;
    memset(buf, 0, pmsDataLen);
    while (mySerial.available()) {
      buf[idx++] = mySerial.read();
    }

    if (buf[0] == 0x42 && buf[1] == 0x4d) {
      pm25 = ( buf[12] << 8 ) | buf[13];
      Serial.print("pm2.5: ");
      Serial.print(pm25);
      Serial.println(" ug/m3");
      hasPm25Value = true;
    }
    timeout--;
    if (timeout < 0) {
      Serial.println("fail to get pm2.5 data");
      break;
    }
  }
}

void initializeWiFi() {
  while (status != WL_CONNECTED) {
    Serial.print("Attempting to connect to SSID: ");
    Serial.println(ssid);
```

整合空氣懸浮粒子感測器測試程式一(PMS3003AirQualityV52)

```
        // Connect to WPA/WPA2 network. Change this line if using open
or WEP network:
       status = WiFi.begin(ssid, pass);
      //  status = WiFi.begin(ssid);

        // wait 10 seconds for connection:
        delay(10000);
      }

      // local port to listen for UDP packets
      Udp.begin(2390);
    }

    void initializeMQTT() {

     // byte mac[6];
     // WiFi.macAddress(MacData);
      memset(clientId, 0, MAX_CLIENT_ID_LEN);
      sprintf(clientId, "FT1_0%02X%02X", MacData[4], MacData[5]);
      sprintf(outTopic, "LASS/Test/Pm25Ameba/%s", clientId);

      Serial.print("MQTT client id:");
      Serial.println(clientId);
      Serial.print("MQTT topic:");
      Serial.println(outTopic);

      client.setServer(server, 1883);
      client.setCallback(callback);


    }

    void printWifiData()
    {
      // print your WiFi shield's IP address:
      Meip = WiFi.localIP();
      Serial.print("IP Address: ");
```

```
  Serial.println(Meip);


  // print your MAC address:
  byte mac[6];
  WiFi.macAddress(mac);
  Serial.print("MAC address: ");
  Serial.print(mac[5], HEX);
  Serial.print(":");
  Serial.print(mac[4], HEX);
  Serial.print(":");
  Serial.print(mac[3], HEX);
  Serial.print(":");
  Serial.print(mac[2], HEX);
  Serial.print(":");
  Serial.print(mac[1], HEX);
  Serial.print(":");
  Serial.println(mac[0], HEX);

  // print your subnet mask:
  Mesubnet = WiFi.subnetMask();
  Serial.print("NetMask: ");
  Serial.println(Mesubnet);

  // print your gateway address:
  Megateway = WiFi.gatewayIP();
  Serial.print("Gateway: ");
  Serial.println(Megateway);
}

void showLed()
{
    if (ParticleSensorStatus)
        {
          digitalWrite(ParticleSensorLed,turnon) ;
        }
        else
```

整合空氣懸浮粒子感測器測試程式一(PMS3003AirQualityV52)

```
            {
              digitalWrite(ParticleSensorLed,turnoff) ;
            }
          if (status == WL_CONNECTED)
            {
              digitalWrite(InternetLed,turnon) ;
            }
            else
            {
              digitalWrite(InternetLed,turnoff) ;
            }


    }


    void initRTC()
    {

         Wire.begin();
        RTC.begin();
      if (! RTC.isrunning()) {
        Serial.println("RTC is NOT running!");
        // following line sets the RTC to the date & time this sketch was
compiled
        RTC.adjust(DateTime(__DATE__, __TIME__));
      }

    }
    void initPins()
    {
    pinMode(DHTSensorPin,INPUT) ;
    pinMode(ParticleSensorLed,OUTPUT) ;
    pinMode(InternetLed,OUTPUT) ;
    pinMode(AccessLed,OUTPUT) ;
```

```
整合空氣懸浮粒子感測器測試程式一(PMS3003AirQualityV52)
digitalWrite(ParticleSensorLed,turnoff) ;
digitalWrite(InternetLed,turnoff) ;
digitalWrite(AccessLed,turnoff) ;

}
```

資料下載：
https://github.com/brucetsao/makerdiwo/tree/master/201605

由於，我們可以連上網際網路，將空氣懸浮粒子感測器讀取的值，透過 MQTT 方式連接到網際網路的 LASS 伺服器。

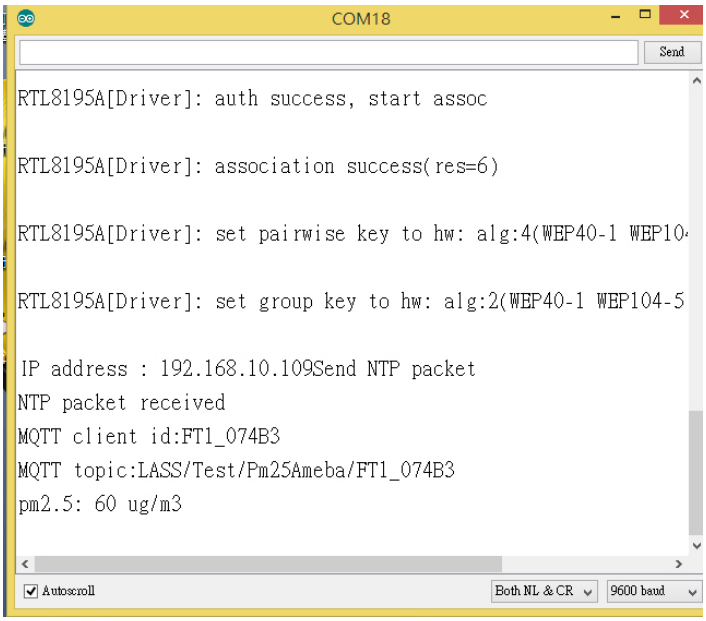

圖 4 整合空氣懸浮粒子感測器測試程式一畫面結果

LASS 平台服務功能

LASS 社群整合了 GOV 的資源，由 https://g0v.tw/zh-TW/index.html 協助幫忙，並且整合了中央研究院的資源，中央研究院研究員陳玲志博士(https://www.facebook.com/lingjyh.chen?fref=ts)也參予其中，對 LASS 雲端平台的資料蒐集、分析、視覺化、整合提供莫大貢獻，所以我們可以到 GOV 零時空污資訊網(http://g0vairmap.3203.info/map.html)察看到所有連上 LASS 雲端平台的 PM2.5 裝置，依 PM2.5 裝置 GPS 資料進行地圖化顯示。

圖 5 GOV 零時空污資訊網

作者的裝置，可以在地圖上看到，裝置名稱為 FT1_07551，位置在員林市，讀者可以到地圖上搜尋之，不過因為作者寫書、測試之用，該裝置有時開機，有時候關機，若讀者沒找到，請見諒。

讀者可以在 http://nrl.iis.sinica.edu.tw/LASS/show.php?device_id=FT1_07551，由下圖所示，可以使用瀏覽器，輸入上面網址，就可以看到作者裝置的視覺化顯示空污資訊。

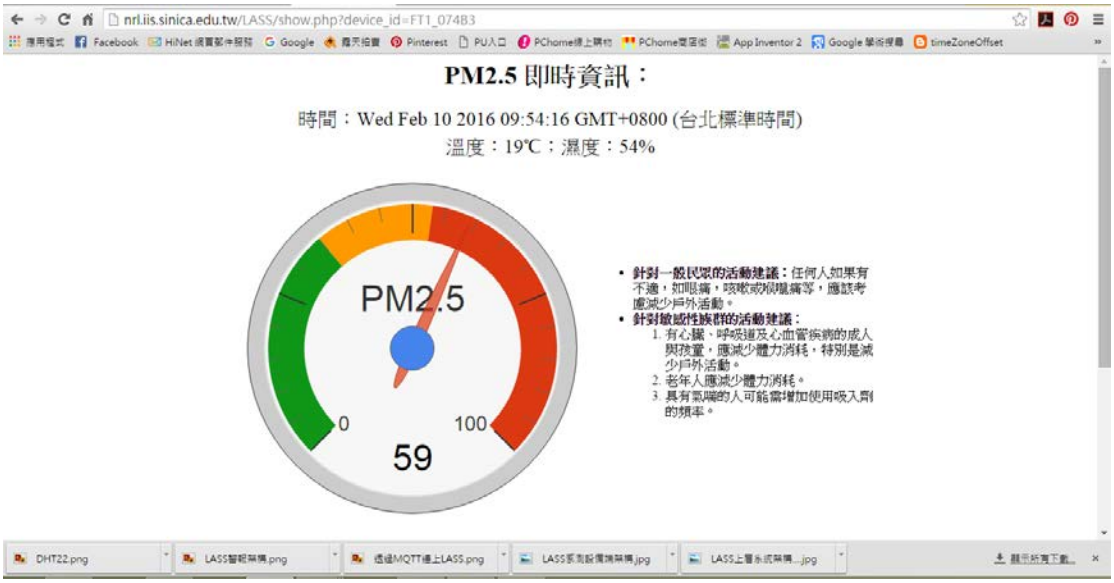

圖 6 視覺化顯示空污資訊

本文為『PM2.5空氣感測器』系列第七篇：上網篇：連上MQTT篇，主樣承接上篇『智慧家庭：PM2.5空氣感測器(上網篇：啟動網路校時功能)』可以連上網路，啟動網路校時功能之後，進而將空污資訊送上雲端，並透過雲端平台的服務，進行視覺化顯示空污資訊。

後續筆者還會繼續發表『PM2.5空氣感測器』系列的文章，讓我們在未來可以創造出更優質、智慧化的家庭。

敬請期待更多的文章。

筆者介紹

曹永忠 (Yung-Chung Tsao)：目前為自由作家，專注於軟體工程、軟體開發與設計、物件導向程式設計、Arduino 開發、嵌入式系統開發，商品攝影及人像攝影。長期投入資訊系統設計與開發、企業應用系統開發、軟體工程、新產品開發管理、商品及人像攝影等領域，並持續發表作品及相關專業著作。

Email:prgbruce@gmail.com ，Line ID：dr.brucetsao
Arduino 部落格：http://taiwanarduino.blogspot.tw/
臉書社群(Arduino.Taiwan)：https://www.facebook.com/groups/Arduino.Taiwan/
活動官網：http://arduino.kktix.cc/
Youtube：https://www.youtube.com/channel/UCcYG2yY_u0m1aotcA4hrRgQ
程式下載網址：https://github.com/brucetsao/makerdiwo

參考文獻：

Hsu, R. (2016). 運用 ESP8266 及 MQTT 完成 IoT 數據傳輸. *智慧家庭*. Retrieved from http://makerpro.cc/2016/02/use-esp8266-and-mqtt-to-transfer-iot-data/

曹永忠. (2016a). 智慧家庭：PM2.5 空氣感測器（感測器篇）. *智慧家庭*. Retrieved from http://vmaker.tw/project/view/695

曹永忠. (2016b). 智慧家庭：PM2.5 空氣感測器（硬體組裝上篇）. *智慧家庭*. Retrieved from http://vmaker.tw/project/view/749

曹永忠. (2016c). 智慧家庭：PM2.5 空氣感測器（硬體組裝下篇）. *智慧家庭*. Retrieved from http://vmaker.tw/project/view/772

曹永忠. (2016d). 智慧家庭：PM2.5 空氣感測器（電路設計上篇）. *智慧家庭*. Retrieved from http://vmaker.tw/project/view/817

曹永忠. (2016e). 智慧家庭：PM2.5 空氣感測器（電路設計下篇）. *智慧家庭*. Retrieved from http://vmaker.tw/project/view/870

曹永忠. (2016f). 智慧家庭：如何安裝各類感測器的函式庫. *智慧家庭*. Retrieved from http://vmaker.tw/project/view/651

曹永忠, 許智誠, & 蔡英德. (2015a). *Ameba 空气粒子感测装置设计与开发 (MQTT 篇):Using Ameba to Develop a PM 2.5 Monitoring Device to MQTT* (初版 ed.). 台湾、彰化: 渥瑪數位有限公司.

曹永忠, 許智誠, & 蔡英德. (2015b). *Ameba 空氣粒子感測裝置設計與開發 (MQTT 篇)):Using Ameba to Develop a PM 2.5 Monitoring Device to MQTT* (初版 ed.). 台湾、彰化: 渥瑪數位有限公司.