# 智慧家庭:PM2.5 空氣感測器(上網篇：啟動網路校時功能)

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(電路設計上、下篇)』(曹永忠, 許智誠, & 蔡英德, 2015a, 2015b)，本文主要是讓 PM2.5 空氣感測器可以連上網路，並且可以啟動網路校時功能，讓 PM2.5 空氣感測器的可以正確使用時間，並且可以透過網路校時來啟動自動化校時功能(曹永忠, 2016a, 2016b, 2016c, 2016d)，簡化手動校時的工作量。

## 連上網路顯示裝置 ID

為了可以上網，我們先了解 MAC[1]媒體存取控制位址，因為很多網路地方，我們需要知道 MAC 位址 (Media Access Control Address)：媒體存取控制位址的資訊(曹永忠, 許智誠, & 蔡英德, 2015c)。

我們將下表之 PMS3003 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 1 PMS3003 空氣懸浮粒子感測器測試程式一

| PMS3003 空氣懸浮粒子感測器測試程式一(PMS3003AirQualityV21) |
|---|

[1] **MAC 位址（Media Access Control Address）**，媒體存取控制位址，或稱為實體位址，是用來定義網路裝置的位置。在 OSI 模型中，第三層網路層負責 IP 位址，第二層資料鏈結層則負責 MAC 位址。MAC 位址用於在網路中唯一標示一個網卡，一台電腦會有一或多個網卡，每個網卡都需要有一個唯一的 MAC 位址。

PMS3003 空氣懸浮粒子感測器測試程式一(PMS3003AirQualityV21)

```
/*
 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)

 */
#include "PMType.h"
#include <WiFi.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 <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[] = "1234";      // your network password
```

PMS3003 空氣懸浮粒子感測器測試程式一(PMS3003AirQualityV21)

```
    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
    int status = WL_IDLE_STATUS;

    #define pmsDataLen 24
    uint8_t buf[pmsDataLen];
    int idx = 0;
    int pm25 = 0;
    LiquidCrystal_I2C lcd(0x27, 2, 1, 0, 4, 5, 6, 7, 3, POSITIVE);  //
設定 LCD I2C 位址
    String MacAddress ;
      IPAddress Meip ;
      IPAddress Mesubnet ;
      IPAddress Megateway ;
    void setup() {
      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
    WiFi.status();
    MacAddress = GetWifiMac() ;
      ShowMac() ;
        initializeWiFi();
      ShowInternetStatus() ;


    }

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

PMS3003 空氣懸浮粒子感測器測試程式一(PMS3003AirQualityV21)

```
  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) ;
      }

}

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())
          {
               lcd.print("Connected:");
              lcd.print(Meip);
          }

}
```

PMS3003 空氣懸浮粒子感測器測試程式一(PMS3003AirQualityV21)

```
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);

}

 String GetWifiMac()
{
   String tt ;
    String t1,t2,t3,t4,t5,t6 ;
  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);
```

PMS3003 空氣懸浮粒子感測器測試程式一(PMS3003AirQualityV21)

```
Serial.print("\n");

  return tt ;
}

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

PMS3003 空氣懸浮粒子感測器測試程式一(PMS3003AirQualityV21)

```
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);
    // 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:
```

PMS3003 空氣懸浮粒子感測器測試程式一(PMS3003AirQualityV21)

```
    delay(10000);
  }


}


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:
```

PMS3003 空氣懸浮粒子感測器測試程式一(PMS3003AirQualityV21)

```
  Megateway = WiFi.gatewayIP();
  Serial.print("Gateway: ");
  Serial.println(Megateway);
}
```

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

上述程式執行後，可以見到下圖之 PMS3003 空氣懸浮粒子感測器測試程式一畫面結果，也可以顯示裝置的 MAC 位址於 LCD2004 顯示模組上。

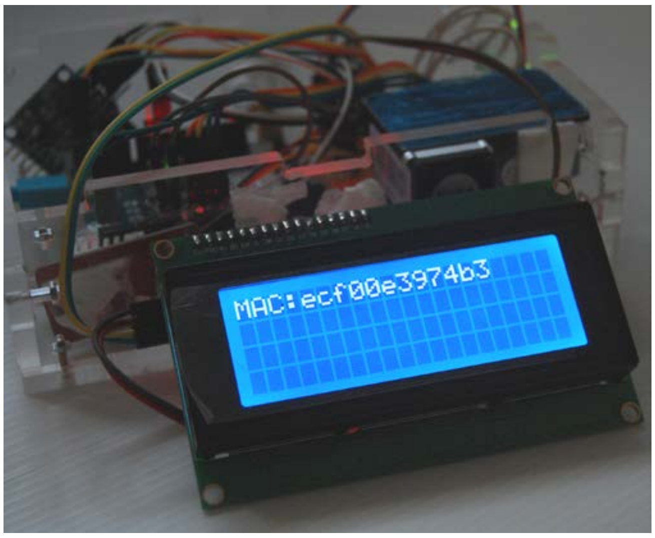


圖 1 PMS3003 空氣懸浮粒子感測器測試程式一畫面結果

### 擴充 RTC 時鐘模組網路校時能力

由於 RTC 時鐘模組是一個可以信賴的即時時鐘模組，但是如果新安裝裝置、更換電池或地區變更等等，都必須要將空氣粒子感測裝置帶回原開發者的研究室方可以更正於 RTC 時鐘模組的時間，雖然 Ameba 開發版有強大的無線網路連接網際網路的能力，但是在六秒中，除了重新讀取空氣粒子感測裝置的資料，還必須完成許多其他的工作，這些都必須耗費 Ameba 開發版的時間，與無線網路連接網際網路取得時間的成本，這樣對一個完善的空氣粒子感測裝置，太耗費在無線網路連接網際網路取得時間的成本。

所以我們如果能將系統修正，在每一次空氣粒子感測裝置開機後，即使用無線網路連接網際網路取得時間，並動態校正 RTC 時鐘模組，往後的時間就完全依

靠內部的 RTC 時鐘模組運作，如此空氣粒子感測裝置可以更加完備，所以我們修改上述程式來達到此功能。

我們將下表之整合空氣懸浮粒子感測器測試程式二攥寫好之後，編譯完成後上傳到 Ameba 開發板，

表 2 整合空氣懸浮粒子感測器測試程式二

整合空氣懸浮粒子感測器測試程式二(PMS3003AirQualityV22)

```
/*
 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)

 */
 /*
  This example demonstrate how to upload sensor data to MQTT server of LASS.
  It include features:
      (1) Connect to WiFi
      (2) Retrieve NTP time with WiFiUDP
      (3) Get PM 2.5 value from PMS3003 air condition sensor with UART
      (4) Connect to MQTT server and try reconnect when disconnect
```

整合空氣懸浮粒子感測器測試程式二(PMS3003AirQualityV22)

```
  You can find more information at this site:

      https://lass.hackpad.com/LASS-README-DtZ5T6DXLbu

*/

//
http://nrl.iis.sinica.edu.tw/LASS/show.php?device_id=FT1_074B3
#define turnon HIGH
#define turnoff LOW
#define DHTSensorPin 8
#define ParticleSensorLed 9
#define InternetLed 10
#define AccessLed 11


#include "PMType.h"
#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 = RTC.now();

#include <LiquidCrystal_I2C.h>
#include <SoftwareSerial.h>

uint8_t MacData[6];
```

整合空氣懸浮粒子感測器測試程式二(PMS3003AirQualityV22)

```
    SoftwareSerial mySerial(0, 1); // RX, TX
    char ssid[] = "TSAO";      // your network SSID (name)
    char pass[] = "1234";     // 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];

    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
```

整合空氣懸浮粒子感測器測試程式二(PMS3003AirQualityV22)

```
    };
    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 pm10 = 0;
    int pm25 = 0;
    int pm100 = 0;
     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

      MacAddress = GetWifiMac() ;
      ShowMac() ;
        initializeWiFi();
```

整合空氣懸浮粒子感測器測試程式二(PMS3003AirQualityV22)

```
  initRTC() ;
  ShowDateTime() ;
  showLed() ;
  ShowInternetStatus() ;

     delay(1500);
}

void loop() { // run over and over
    ShowDateTime() ;
  showLed() ;
    retrievePM25Value() ;

  delay(6000); // delay 1 minute for next measurement


}



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) ;
          }
```

整合空氣懸浮粒子感測器測試程式二(PMS3003AirQualityV22)

```
          else
                    {
               lcd.print("DisConnected:");
              digitalWrite(InternetLed,turnoff) ;
          }

}

void ShowPM(int pp25, int pp10, int pp100)
{
    lcd.setCursor(0, 3); // 設定游標位置在第一行行首
     lcd.print("Dust:");
     lcd.print(pp25);
     lcd.print("/");
     lcd.print(pp10);
     lcd.print("/");
     lcd.print(pp100);

}

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;
DateTime now = RTC.now();
 ttt = print4digits(now.year()) + "-" + print2digits(now.month())
```

整合空氣懸浮粒子感測器測試程式二(PMS3003AirQualityV22)

```
+ "-" + print2digits(now.day()) ;
     //ttt = print4digits(NDPyear) + "/" + print2digits(NDPmonth) + "/"
+ print2digits(NDPday) ;
      return ttt ;
    }

    String  StringDate(int yyy,int mmm,int ddd) {
      String ttt ;
    //nowT  = now;
     ttt = print4digits(yyy) + "-" + print2digits(mmm) + "-" +
print2digits(ddd) ;
      return ttt ;
    }


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

    String  StringTime(int hhh,int mmm,int sss) {
      String ttt ;
      ttt = print2digits(hhh) + ":" + print2digits(mmm) + ":" +
print2digits(sss) ;
    return ttt ;
    }


    String GetWifiMac()
    {
       String tt ;
```

整合空氣懸浮粒子感測器測試程式二(PMS3003AirQualityV22)

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

整合空氣懸浮粒子感測器測試程式二(PMS3003AirQualityV22)

```
  {
      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 ;
}



// 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
```

整合空氣懸浮粒子感測器測試程式二(PMS3003AirQualityV22)

```
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;

      *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)) {
```

整合空氣懸浮粒子感測器測試程式二(PMS3003AirQualityV22)

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


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) {
```

整合空氣懸浮粒子感測器測試程式二(PMS3003AirQualityV22)

```
        pm25 = ( buf[12] << 8 ) | buf[13];
        pm10 = ( buf[10] << 8 ) | buf[11];
        pm100 = ( buf[14] << 8 ) | buf[15];
        Serial.print("pm2.5: ");
        Serial.print(pm25);
        Serial.print(" ug/m3");
        Serial.print("pm1.5: ");
        Serial.print(pm10);
        Serial.print(" ug/m3");
        Serial.print("pm100: ");
        Serial.print(pm100);
        Serial.print(" ug/m3");
        Serial.println("");
        hasPm25Value = true;
        ShowPM(pm25,pm10,pm100) ;
      }
      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);
      // 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);
    }
```

整合空氣懸浮粒子感測器測試程式二(PMS3003AirQualityV22)

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


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: ");
```

整合空氣懸浮粒子感測器測試程式二(PMS3003AirQualityV22)

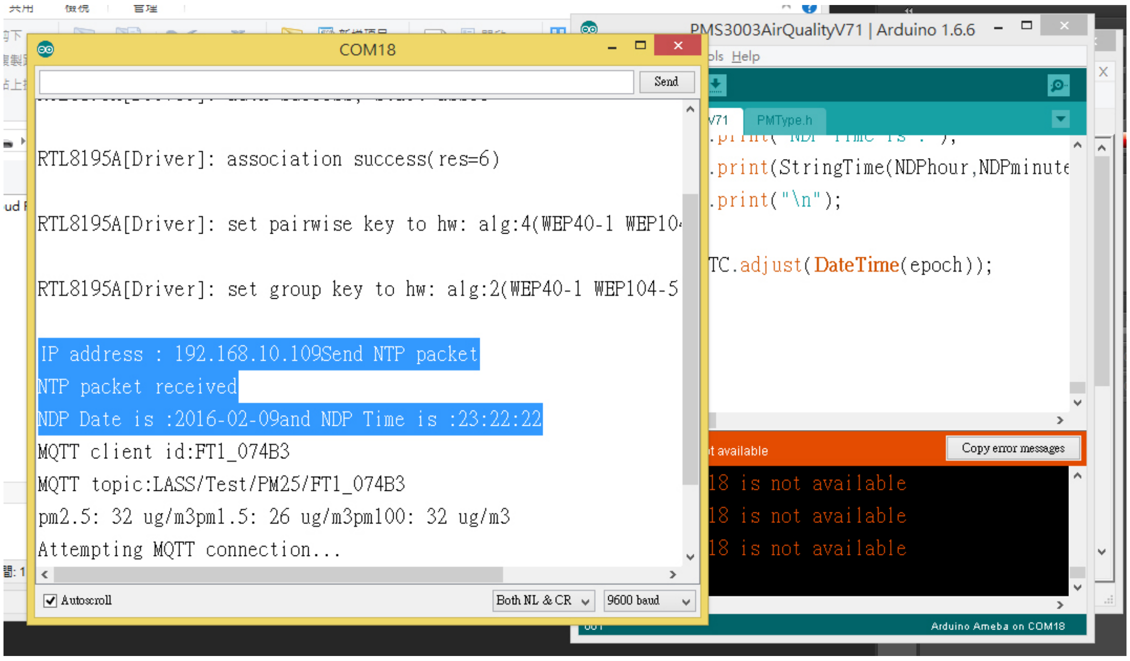
```
  Serial.println(Megateway);
}

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


}


void initRTC()
{
     Wire.begin();
    RTC.begin();
    SetRTCFromNtpTime() ;
  if (! RTC.isrunning()) {
    Serial.println("RTC is NOT running!");

  }
}
void initPins()
```

整合空氣懸浮粒子感測器測試程式二(PMS3003AirQualityV22)

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


    digitalWrite(ParticleSensorLed,turnoff) ;
    digitalWrite(InternetLed,turnoff) ;
    digitalWrite(AccessLed,turnoff) ;

    }
    void SetRTCFromNtpTime()
    {
      retrieveNtpTime();
      //DateTime ttt;
        getCurrentTime(epoch, &NDPyear, &NDPmonth, &NDPday, &NDPhour,
&NDPminute, &NDPsecond);
        //ttt->year = NDPyear ;
        Serial.print("NDP Date is :");
        Serial.print(StringDate(NDPyear,NDPmonth,NDPday));
        Serial.print("and ");
        Serial.print("NDP Time is :");
        Serial.print(StringTime(NDPhour,NDPminute,NDPsecond));
        Serial.print("\n");

            RTC.adjust(DateTime(epoch));



    }
```

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

由於空氣粒子感測裝置每次開機時，我們可以連上網際網路，使用無線網路連接網際網路取得時間，並動態校正 RTC 時鐘模組，往後的時間就完全依靠內部的 RTC 時鐘模組運作。

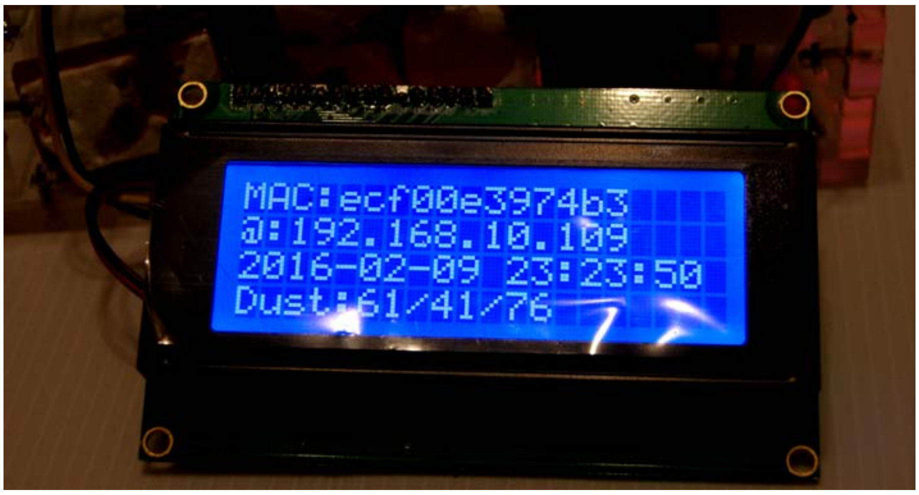


```
RTL8195A[Driver]: association success(res=6)

RTL8195A[Driver]: set pairwise key to hw: alg:4(WEP40-1 WEP10

RTL8195A[Driver]: set group key to hw: alg:2(WEP40-1 WEP104-5

IP address : 192.168.10.109Send NTP packet
NTP packet received
NDP Date is :2016-02-09and NDP Time is :23:22:22
MQTT client id:FT1_074B3
MQTT topic:LASS/Test/PM25/FT1_074B3
pm2.5: 32 ug/m3pm1.5: 26 ug/m3pm100: 32 ug/m3
Attempting MQTT connection...
```

```
.print(StringTime(NDPhour,NDPminute
.print("\n");

TC.adjust(DateTime(epoch));
```

```
18 is not available
18 is not available
18 is not available
```

(a). 開發 IDE 監控畫面



(b). 校時之後空氣粒子感測裝置 LCD 顯示畫面

圖 2 整合空氣懸浮粒子感測器測試程式二畫面結果

本文為『PM2.5空氣感測器』系列第六篇：上網篇：啟動網路校時功能；主要介紹如何將PM2.5空氣感測器加入RTC時鐘模組，取得MAC位址（Media Access Control Address)：媒體存取控制位址，透過WIFI連接上網，並且可以透過網路校時來啟動自動化校時功能(曹永忠, 2016a, 2016b, 2016c, 2016d)，簡化手動校時的工作量，為系上網篇第一篇文章，逐一完成PM2.5空氣感測器的WIFI連接上網、校時、顯示資訊等功能。

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_u0mlaotcA4hrRgQ
程式下載網址：https://github.com/brucetsao/makerdiwo

## 參考文獻：

曹永忠. (2016a). AMEBA 透過網路校時 RTC 時鐘模組. *智慧家庭.* Retrieved from http://makerpro.cc/2016/03/using-ameba-to-develop-a-timing-controlling-device-via-internet/

曹永忠. (2016b). 用 RTC 時鐘模組驅動 Ameba 時間功能. *智慧家庭.* Retrieved from http://makerpro.cc/2016/03/drive-ameba-time-function-by-rtc-module/

曹永忠. (2016c). 智慧家庭實作：ARDUINO 永遠的時間靈魂－RTC 時鐘模組. *智慧家庭.* Retrieved from http://www.techbang.com/posts/40838

曹永忠. (2016d). 實戰 ARDUINO 的 RTC 時鐘模組，教你怎麼進行網路校時. Retrieved from http://www.techbang.com/posts/40869-smart-home-arduino-internet-soul-internet-school

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.

曹永忠, 許智誠, & 蔡英德. (2015c).  邁入『物聯網』的第一步：如何使用無線傳輸：基本篇. *物聯網.*  Retrieved from http://www.techbang.com/posts/25459-technology-in-the-future-internet-of-things-business-model-how-to-use-wireless-transmission-the-basic-text