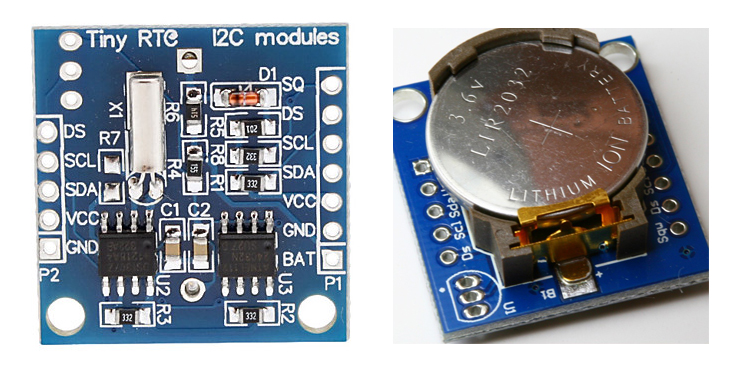
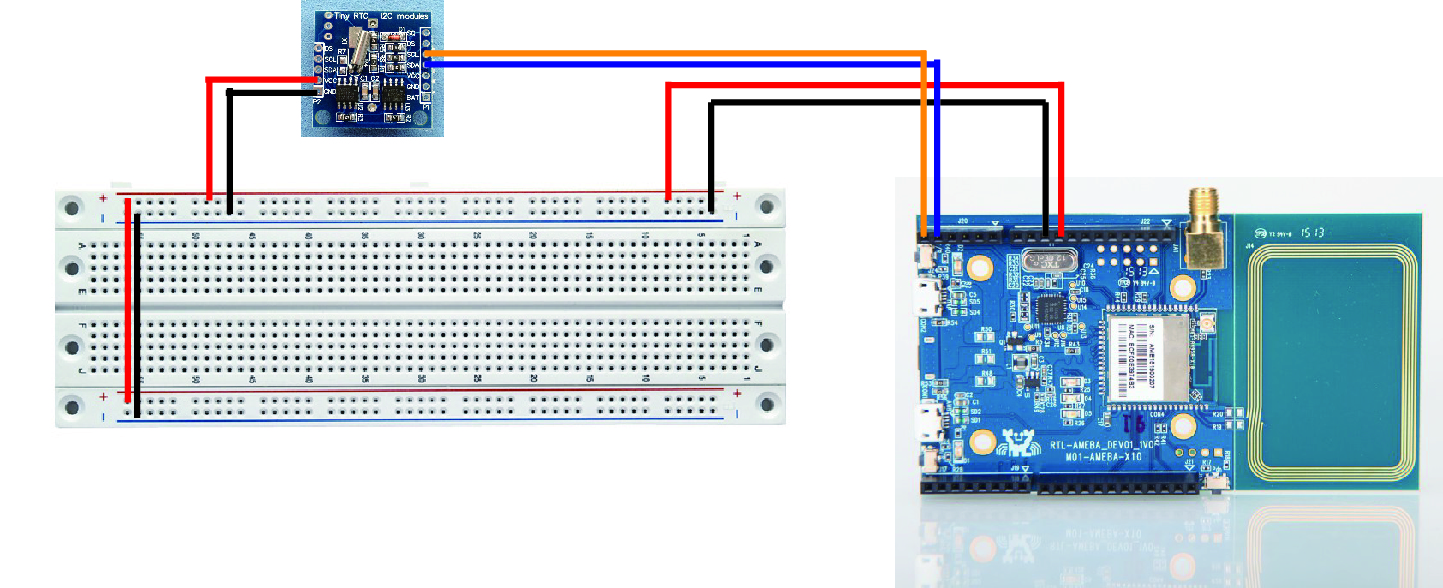
程式：SetTime\_fromNet 網路校時

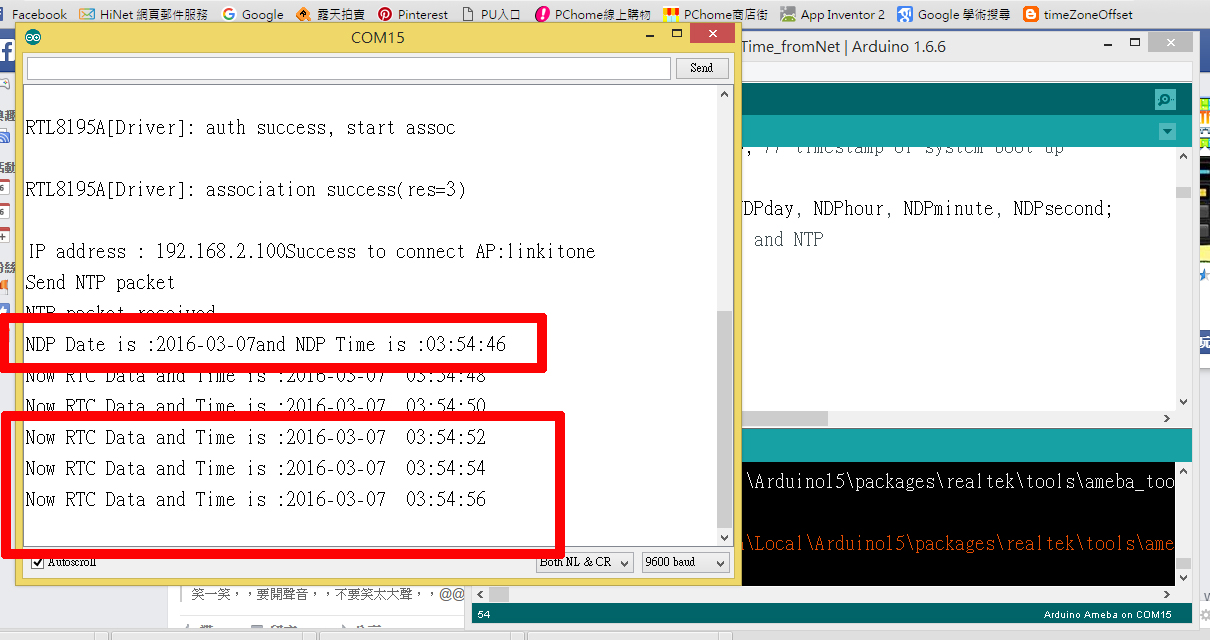
**開啟程式**SetTime\_fromNet

**程式位址：https://github.com/brucetsao/BruceCourses/blob/master/105ANQU\_IOT/Code/SetTime\_fromNet/SetTime\_fromNet.ino**





|  |
| --- |
| #include <Wire.h>  #include "RTClib.h"  RTC\_DS1307 RTC;  // above is used for RTC  #include "PMType.h"  #include <WiFi.h>  #include <PubSubClient.h>  #include <WiFiUdp.h>  // Aboev is used for WIFI  uint8\_t MacData[6];  char ssid[] = "TSAO\_1F"; // your network SSID (name)  char pass[] = "TSAO1234"; // your network password  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];  IPAddress Meip , Megateway , Mesubnet ;  String MacAddress ;  int status = WL\_IDLE\_STATUS;  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  unsigned long epoch ;  int NDPyear, NDPmonth, NDPday, NDPhour, NDPminute, NDPsecond;  // this is used for WIFI and NTP  void setup() {  Serial.begin(9600);  initRTC() ;  // init RTC Modules  MacAddress = GetWifiMac() ;  ShowMac() ;  initializeWiFi();  ShowNTPDateTime() ;  SetRTCTime(NDPyear, NDPmonth, NDPday, NDPhour, NDPminute, NDPsecond);  delay(1500);  }  void loop() { // run over and over  delay(1000); // delay 1 minute for next measurement  Serial.print("Now RTC Data and Time is :") ;  Serial.print(ShowDateTime()) ;  Serial.print("\n") ;  delay(1000) ;  }  void ShowNTPDateTime()  {  retrieveNtpTime() ;  getCurrentTime(epoch + timeZoneOffset, &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");  }  void ShowMac()  {  Serial.print("MAC:");  Serial.print(MacAddress);  Serial.print("\n");  }  void ShowInternetStatus()  {  if (WiFi.status())  {  Meip = WiFi.localIP();  Serial.print("Get IP is:");  Serial.print(Meip);  Serial.print("\n");  }  else  {  Serial.print("DisConnected:");  Serial.print("\n");  }  }  void initRTC()  {  Wire.begin();  RTC.begin();  if (! RTC.isrunning()) {  Serial.println("RTC is NOT running!");  }  }  String ShowDateTime()  {  return StrDate() + " " + StrTime() ;  }  String StrDate() {  String ttt ;  //nowT = now;  DateTime now = RTC.now();  ttt = print4digits(now.year()) + "-" + print2digits(now.month()) + "-" + 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 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 ;  }  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) {  String ttt ;  if (number >= 0 && number < 16)  {  ttt = String("0") + String(number, HEX);  }  else  {  ttt = String(number, HEX);  }  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 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 ;  epoch = secsSince1900 - seventyYears ;  epochSystem = epoch - millis() / 1000;  }  }  void SetRTCTime( int yr, int mon, int dd, int hr, int mins, int secs) {  RTC.adjust(DateTime(yr, mon, dd, hr, mins, secs));  }  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)) {  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 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);  }  Serial.print("Success to connect AP:") ;  Serial.print(ssid) ;  Serial.print("\n") ;  // 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);  Serial.print("\n");  // 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);  } |

****

SetTime\_fromNet **程式重點解說**

* #include <WiFi.h> 使用網路必要函數
* uint8\_t MacData[6]; 儲存 MAC資料
* initRTC() ; 初始化時鐘模組
* ShowNTPDateTime() ; 取得網路時間並顯示
* retrieveNtpTime() ;取得網路時間
* getCurrentTime(epoch+timeZoneOffset, &NDPyear, &NDPmonth, &NDPday, &NDPhour, &NDPminute, &NDPsecond); 將網路時間存入變數
* SetRTCTime(NDPyear, NDPmonth, NDPday, NDPhour, NDPminute, NDPsecond); 將取得網路時間調整時鐘模組