Synchronize to a Time Service

Introduction

- Network Time Protocol (NTP) is an internet protocol for synchronizing computer clocks to a time reference on the network.
- Most operating systems use NTP to keep computers on time. Time is important for computers when communicating with each other, handling files, software that time tags information, and for security.
- A time source is an NTP server which is a computer that has a time reference such as a GPS receiver or some device that keeps precise time.
- The protocol measures the time difference between the computer and the reference source and adjusts the computer’s time as needed. The measurement is done periodically such as once a minute or once every several minutes.
- You can go to www.ntp.org for details.

NTP Client

- In the Arduino IDE, open Manage Libraries in the Tools menu.
- After the list of libraries are loaded, enter NTP to search for the **NTPClient** library. It is written by Fabrice Weinberg.
- Install the library.

NTP Test Program

- Enter the program to the right.
- The setup is only shown here. The loop is on the next page.
- You will need internet access to use the NTP server.
- NTP uses UDP protocol so WiFiUdp.h is included.
- A UDP object is created along with an instant of the NTP client.
- In setup, the connection to a router with internet access is made and the NTP client is started.

```
#include <NTPClient.h>
#include <ESP8266WiFi.h>
#include <WiFiUdp.h>

WiFiUDP ntpudp;
NTPClient timeClient(ntpudp);

void setup() {
  Serial.begin(115200);
  WiFi.begin("ssid","password");
  while(WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
  timeClient.begin();
}
```

NTP Test Program

- The loop call timeClient.update() to get an update to time.
- The getFormmattedTime() function converts the time to human readable hours, minutes, and seconds.
- getEpochTime() gets the time in seconds. This is the number of seconds since January 1, 1970.
- Try out the program.

```
void loop() {
  timeClient.update();
  String tim = timeClient.getFormattedTime();
  uint32_t ttag = timeClient.getEpochTime();
  tim = tim + " " + String(ttag);
  Serial.println(tim);
  delay(1000);
}
```

More Time Formats

- Just getting the time of day doesn't provide enough information. The epoch can be used to convert to time with the date included.
- At the top of the program, add the **time.h** include file. It will provide functions for converting the epoch time to a better human readable time with the complete date.

```
#include <NTPClient.h>
#include <ESP8266WiFi.h>
#include <WiFiUdp.h>
#include <time.h>

WiFiUDP ntpudp;
NTPClient timeClient(ntpudp);

void setup() {
  Serial.begin(115200);
  WiFi.begin("ssid","password");
  while(WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
  timeClient.begin();
}
```

More Time Formats

- In the loop, add the highlighted lines.
- **struct tm** is a time structure that include seconds and microseconds.
- Notice that the ttag variable is now a **time_t** type variable. This is needed for the **localtime()** function to operate properly.
- **localtime()** converts the epoch time to the time structure.
- **strftime()** converts the time structure to human readable text.

```
void loop() {
  struct tm ts;
  char buf[80];
  timeClient.update();
  String tim = timeClient.getFormattedTime();
  time_t ttag = timeClient.getEpochTime();
  tim = tim + " " + String(ttag);
  Serial.println(tim);
  ts = *localtime(&ttag);
  strftime(buf,80,"%a %Y-%m-%d %H:%M:%S %Z",
  &ts));
  Serial.println(buf);
  delay(1000);
}
```

More info on strftime()

https://www.ibm.com/docs/en/zos/2.3.0?topic=functions-strftime-convert-formatted-time

End

- This code can be integrated into other programs that require time tagging data with absolute time. This would allow correlating data between different systems in different locations.