CalQRP Club Project

NTP Dual Clock

Read Me First v1.0

By Yin Shih, N9YS

This month's CalQRP Club Project is a Dual NTP (Network Time Protocol) Clock by W8BH. This is a relatively easy build, with most of the complexity in the firmware. The instructions are in 2 parts. The file NTP_clock_build.pdf discusses the hardware build and the file NTP_DualClock.pdf mainly discusses the firmware build.

The instructions as written are based on the use of an ESP32-WROOM-32 MCU, however the MCU provided with the kit is the ESP8266 Wemos D1 Mini. This is a more compact, cheaper MCU, that still does the job. The instructions for the firmware build based on an ESP8266 Wemos D1 Mini are discussed in Appendix A, on pp.12 of the NTP_DualClock.pdf file.

1) Complete the hardware build per instructions.

2) Configure the Arduino IDE

    a. First copy the .ino file into a directory of the same name
    b. Open the .ino file which should launch the Arduino IDE
    c. Add the ESP8266 board library to the IDE as described in step A1 of Appendix A. Select the Wemos D1 Mini as the target board. Set the baud rate (921600). Set the COM port (you will have to figure out which COM port is assigned to the target).

3) Edit the .ino file to set definitions for your site and location

    This involves edits near the head of the .ino file, that have the following lines:

```
#define TITLE              "NTP TIME"
#define WIFI_SSID          "yourSSID"
#define WIFI_PWD           "yourPASSWORD"
#define NTP_SERVER         "pool.ntp.org"

#define TZ_RULE            "EST5EDT,M3.2.0/2:00:00,M11.1.0/2:00:00"
```

yourSSID should be edited to your wifi SSID, and yourPASSWORD should be edited to your WiFi password. Note that the new contents should be enclosed by double quotes as shown above.

The leading portion of TZ_RULE should also be edited, based on your local time zone as follows:

Eastern:	EST5EDT
Central:	CST6CDT
Mountain:	MST7MDT
Pacific:	PST8PDT

if you are not in one of the 4 continental US time zones, other timezone strings can be found here:

https://github.com/nayarsystems/posix_tz_db/blob/master/zones.csv

4) Attempt to Compile/Verify from the Arduino IDE

   This will succeed or, more likely, generate an error that a library wasn't found. Libraries such as TFT_eSPI and ezTime may be added to the Arduino IDE by invoking Sketch/Include Library/Manage Libraries. Once in Manage Libraries, search for the library name and add it to the project.

5) Once all libraries are correct and the .ino file can be built. Go to the directory: Users\yourWindowsUsername\Documents\Arduino\libraries\TFT_eSPI\ and find the file User_Setup.h. Review the definitions in User_Setup.h and ensure that the definitions conform to the instructions in step A3 of Appendix A.

   By conform is meant that #defines may or may not have a "//" preceding the #define. "//" forces the compiler to treat the line as a comment and not a part of the code. Removing the "//" means that the line is recognized and will affect the code compilation. Step A3 identifies the #defines that are meant to be recognized, with all other lines treated as comments (by having a preceding "//").

6) At this point, an Upload can be performed to load the ESP8266 with the final firmware build.

P.S. If someone is interested in a minor coding project, it would be a useful addition to modify the code to get solar weather info, which is updated regularly at

https://services.swpc.noaa.gov/text/wwv.txt, and adding it to the display. This would involve getting the wwv.txt file over the internet, parsing it for the relevant data, and then finding or making room on the display to add the solar weather info.