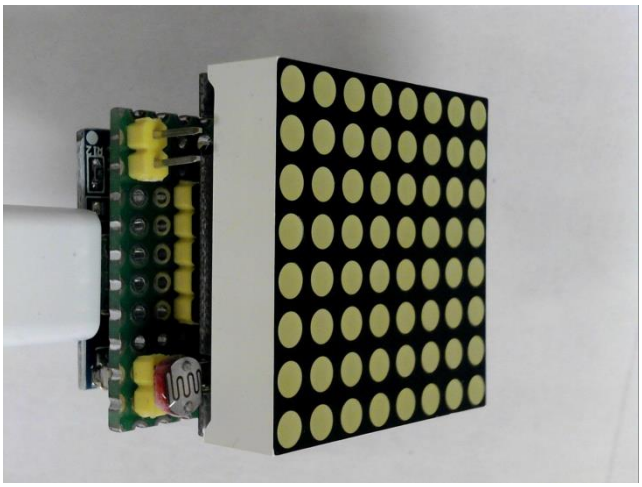
ARDUINO ESP8266 BASED MICRO NTP CLOCK

Author: 6V6GT

Version v1.01 07-May-2019

1 INTRODUCTION

This is an easy to assemble NTP clock project using a small number of components. It is extremely small measuring only 32x32x25 mm and the display device, an 8x8 LED dot matrix module displays the hours and minutes in a controlled sequence.



It uses a Wemos D1 Mini pattern ESP8266 micro controller and a Robotdyn 8x8 led module with a Max 7221 driver chip.

The initial configuration, that is joining the device to a wireless LAN, setting the time zone etc. is done through a web browser. Once configured, it should need no further attention and will handle daylight saving time changes automatically.

2 CONTENTS

2.1 ABBREVIATIONS

AP Mode	Access point mode. In this mode, the ESP8266 creates an access point which wireless devices can connect to. It is used for an initial configuration or where no other WLAN is available.
DST	Daylight saving time.
Eeprom	Electrically erasable read only memory
ESP8266	A microcomputer compatible with the Arduino development environment.
Flash Memory	Rewriteable permanent storage
IC	Integrated Circuit
IDE	Interactive Development Environment.
.ino File	The main file of an Arduino program.
IP Address	Network address of a connected device in example format 192.168.1.99
NTP	Network time protocol
PSK	Pre-shared key - a WLAN equivalent of a password.
RTC	Real time clock - a crystal controlled time keeping module.
SMD	Surface mounted soldered on components (no connecting wires)
SSID	WLAN network name
STA mode	An ESP8266 where the device is used as a wireless client of an existing network
UTC	A standard time: Coordinated Universal Time similar to GMT
WLAN	Wireless Local Area Network

3 SOLUTION DESCRIPTION

3.1 OVERVIEW

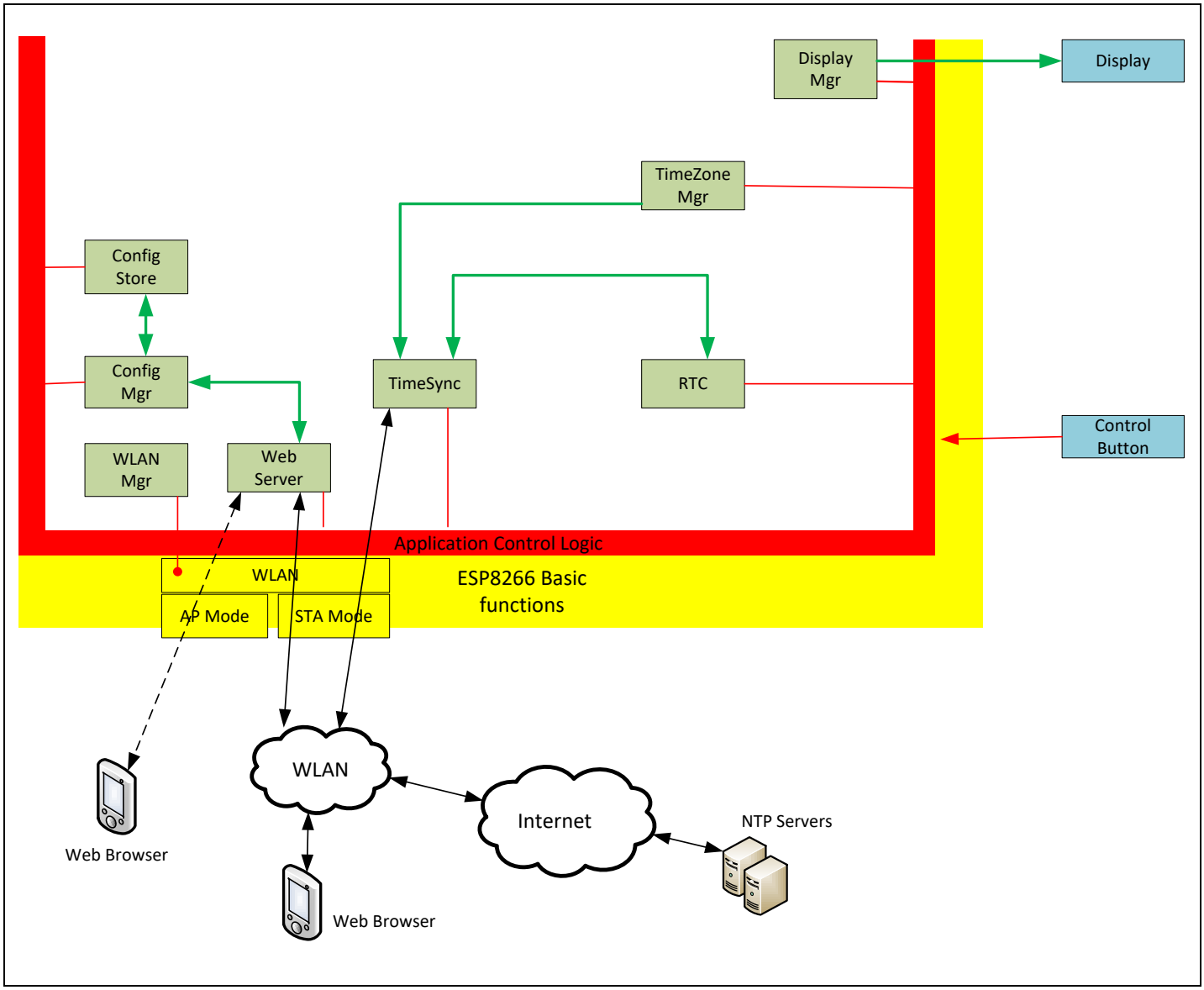
The solution is based on using an ESP8266 microcontroller to handle the internet connection needed to contact the time servers in order to obtain regularly a current time stamp. The ESP8266 does this when it acts as client of the user's WLAN (so called STA mode). The time stamp thus obtained is a highly accurate and high resolution epoch time based on seconds and fractions of a second since 1st Jan 1900 (UTC). The system time of the ESP8266 has a granularity of 1 second. A special feature of the TimeSync module in this solution is that it waits until the next second rollover before applying any time updates to maintain a high degree of accuracy. Of course, the current application does not require such a fraction of a second accuracy, but the design is based on Speaking Clock design [Ref 1] which could benefit from this. Another feature, also important for the satisfactory operation of the ESP8266, is that the call to the NTP server is non-blocking pending the arrival of the NTP reply. A time zone library is used to convert the UTC time obtained from the NTP server into the user's local time zone also handling daylight saving time changes.

The configuration of the device is through a web browser and the user can optionally tailor the systems 'factory' configuration to his local own environment (although this is recommended only for testing). During the initial configuration, when the user's WLAN credentials are not yet known to the device, it is forced into access point (AP) mode and the user contacts it through its own network to enter the required configuration parameters. The device can also be forced into AP mode on request.

The display module is based on a MAX7221 (or MAX7219 which is very similar) and a small LED dot matrix display. The display is 5volt. The design takes the ultra-cautious approach of using voltage dividers on all signal pins between the ESP8266 and the MAX72xx pins. This works and protects the ESP8266 against any 5 volt pullup tendency within the MAX72xx, however, the user should verify the design on a bread board before soldering. Display brightness is controlled by an LDR and gives 3 different brightness levels and a hysteresis control to prevent flicker when the ambient light level approaches a level boundary. The display can be logically rotated (90,180,270 degrees) in software if required.

During time display, the cycle is: hour display (600mS) ; colon (':') display (300mS) ; minutes display (600mS) followed by a blank display of 600mS then the cycle repeats itself. If the device displays its IP address, for example when pushing the control button during normal operation, this scrolls across the display at a rate of 1 second per digit and, at the end, is replaced with a time display.

3.2 FUNCTION BLOCK DIAGRAM



3.3 MAIN SOFTWARE MODULE DESCRIPTION

There is a near 1:1 relationship between the software modules identified in the diagram above and the C++ compilation units in the program.

For further details, see the source code and comments.

3.3.1 TIME HANDLING

The internal time of all devices is maintained in UTC. At the point where is it is used (display) it is converted using offset for the selected time zone. Similarly for manual time entry, if used. The time is entered by the user in local time and converted to UTC. Sub second accuracy is achieved for both NTP and RTC time sources. The RTC is optional and ignored if not present.

3.3.1.1 TimeSync

This obtains a time from NTP (where available) and maintains the System time and the RTC. The system time and RTC have a granularity of 1 second. NTP delivers a time including fractions of a second so time updates are scheduled for the next full second. Priority of the time sources is enforced here. Only if NTP is, or becomes, unavailable, typically at system start, does the RTC get used as a time source.

3.3.1.2 TimeZoneManager

This is based on a design by Jack Christensen with a small modification to permit dynamic updating of the time zone being the timezone instead of being defined only at system start.

3.3.1.3 RTC

This is optional and not shown in the construction. This is also a modification of a standard library design intended for a predecessor of the DS3231 chip with corrections to make it compatible with the ESP8266. A future activity is to select a completely standard library (there are many to evaluate) and integrate it here. Special is the configuration feature to generate a 1Hz squarewave where the falling edge indicates a seconds rollover.

3.3.2 CONFIGURATION MANAGEMENT

When the ESP8266 is first loaded with the Speaking Clock software, it has a 'factory' or default configuration. The configuration is managed through a web server which the user contacts through a web browser. A WLAN manager can create a wireless access point (AP Mode) or, if credentials are available, join an existing WLAN.

The configuration is stored in flash memory (emulated Eeprom) so it is preserved with the device is powered off, and a copy is maintained in RAM from the time the device is switched on.

3.3.2.1 WLAN Manager

This selects which wireless mode is active for the device. Normal is STAmode where the device has the access data to join an existing WLAN. An non-configured device uses AP mode where the device creates its own WLAN.

The control button can also be used to force a device into AP mode if required.

3.3.2.2 Web Server

This is configured to deliver a static web page to the browser containing all the logical pages for the configuration settings. When the browser loads this page, a return to the web server is initiated to collect all the dynamic data in the form of a JSON string. If the use makes changes in the browser and hits the "update" button, these changes are returned to the web server and loaded back into flash memory.

3.3.2.3 Config Manager

This is used to load the flash memory configuration into the Config Store in RAM and, in the case of changes, copies the RAM version back into flash memory so that at the next startup of the device, the newest changes will still be available. If the device has never been configured it will not have a special marker (referred to as a

magic string in the program code) in the flash memory so the factory configuration is installed. Further, if the format of the flash memory has been changed, for example to add new configuration parameters, then the magic string must also be changed to force the flash memory data back to a factory condition.

3.3.3 DISPLAY MANAGER

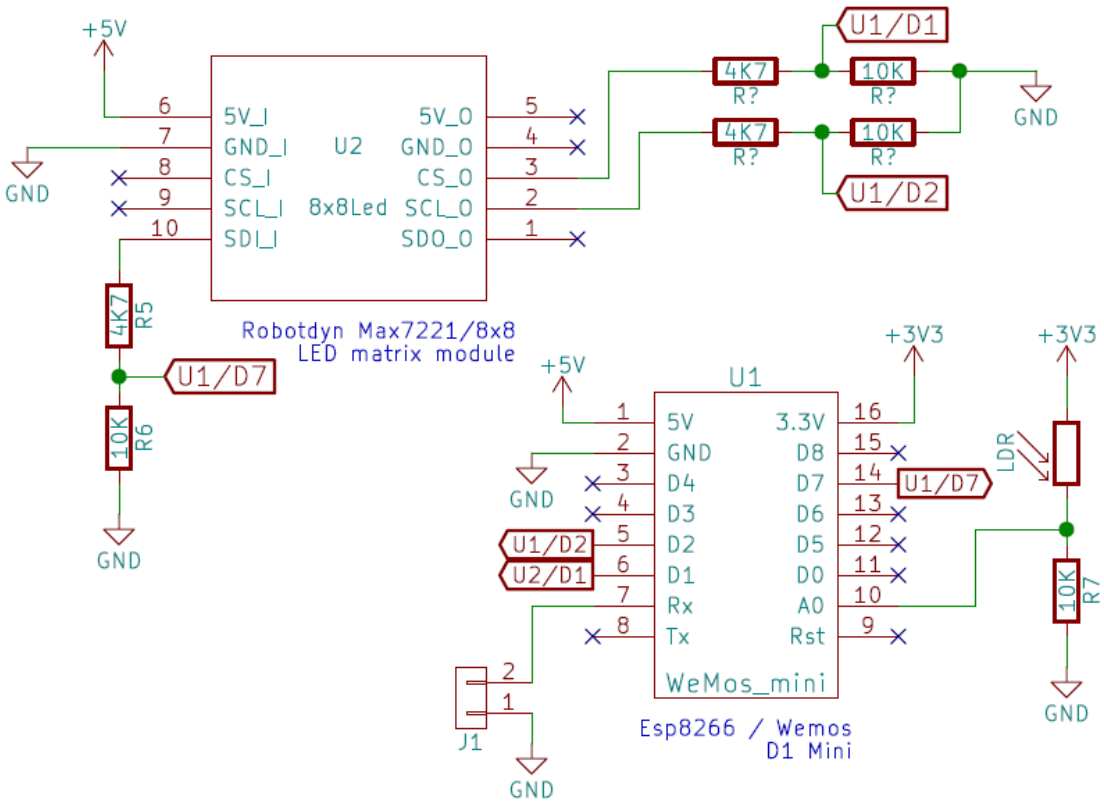
This is relatively simple because the chosen LCD screen (with I2C backback) is very easy to drive. This contains the definition of the special symbols (see user manual), the formatting of the time and date in the display and the logic for the timed sequences of multiple pages e.g. configuration information.

3.3.4 APPLICATION CONTROL LOGIC

The .ino file has a setup section and a loop section. The setup section starts, in a controlled order, the setup routines for all the software modules. Similarly, the loop section, on each iteration, calls the loop section in each of the software modules. The handling of the control button is also done here.

3.4 SCHEMATIC DIAGRAM

Note 1: The jumper J1 on the schematic is actually used as a push button (see 5.3)


+5V
GND
6
7
8
9
10
5V_I
GND_I
CS_I
SCL_I
SDI_I
U2
8x8Led
5V_O
GND_O
CS_O
SCL_O
SDO_O
5
4
3
2
1
4K7
R5
U1/D7
10K
R6
GND
Robotdyn Max7221/8x8
LED matrix module
4K7
R?
4K7
R?
10K
R?
10K
R?
U1/D1
U1/D2
GND
+5V
+3V3
+3V3
U1
1
2
3
4
5
6
7
8
5V
GND
D4
D3
D2
D1
Rx
Tx
3.3V
D8
D7
D6
D5
D0
A0
Rst
16
15
14
13
12
11
10
9
U1/D2
U2/D1
U1/D7
LDR
10K
R7
GND
GND
J1
2
1
GND
WeMos_mini
Esp8266 / Wemos
D1 Mini

3.5 MAIN COMPONENTS/ MODULES USED

Wemos Mini-D1 ESP8266 WiFi Module

32 bit microcontroller with in-built Wifi and 4MB flash memory and CH340G USB interface for programming and file upload.

It is essentially a 3.3 volt device and the pins are not specified for 5 volt tolerance. It can supply 5volt devices through the pin 5V which is connected to the USB power pin.

Note: This device has resistor network on pin A0 so it is accepts 3.3 volts. If you use another ESP8266 module, you should check this otherwise the maximum voltage could be limited to 1 volt.

Display

Robotdyn Matrix LED 8x8 module. White color. 32x32mm. Driver MAX7221. (5 Volt)

RTC DS3231

This is optional and the program is structured to handle one. No instructions are given here, however, to integrate one.
This is a precision clock module. In this application, it is configured additionally to deliver a pulse on pin SQW at each seconds rollover for sub second accuracy.
These are 3 or 5 volts compatible.
Note: these are intended to be used with rechargeable cells (LIR2032) for backup. If you wish to use cheaper CR2032 non-rechargeable cells, you **must** disable the charging circuit say by removing the resistor directly above the SCL label by the 4-hole group on the board.

4 CONSTRUCTION

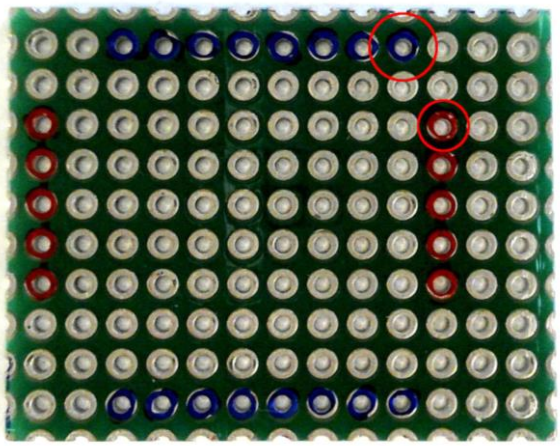
No printed circuit board design has been produced for this project. The layout here is a suggestion for a compact design.

You have 2 main options for the construction. One is to make the design as compact as possible which means soldering device pins directly to the board. The other is to solder female header pins to the board and make the ESP8266 and the LED matrix devices pluggable. The first option has been used in the illustrations, but please note that the order of construction with this option is critical as is testing at each stage because the direct soldering prevents access to the covered components making later corrections very difficult. This also means you have to add spacers to the header pins to get enough clearance for all the underlying components. If in doubt, use the female headers on the board and plug the devices into these.

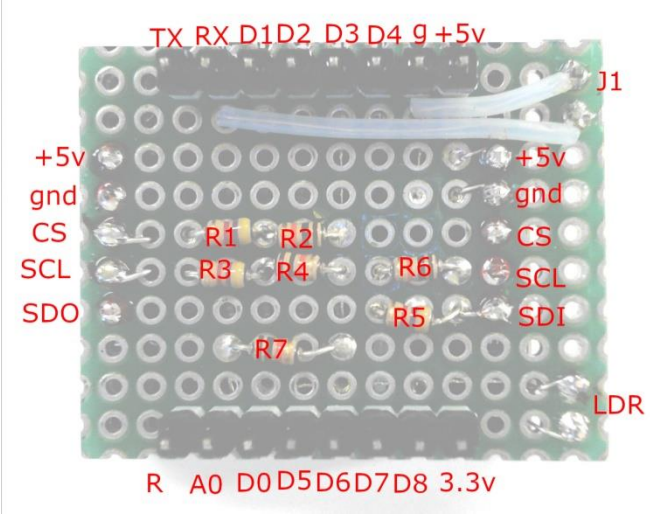
You are advised to test the design on a bread board before building the project, but note, you cannot solder header pins onto the ESP8266 module at this stage in the construction unless you choose to use to solder female header pins to the board.

The component inventory is visible from the schematic. Note that the resistors are miniature 1/8 watt types.

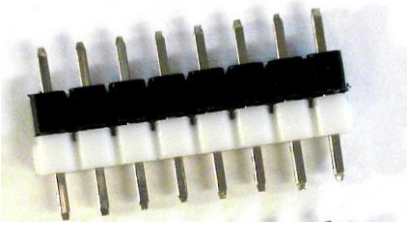
Prototype board fragment “ESP” side. The larger red circle indicates +5v of the ESP 8266 board. The smaller red circle indicates the +5v (in) of the MAX module.



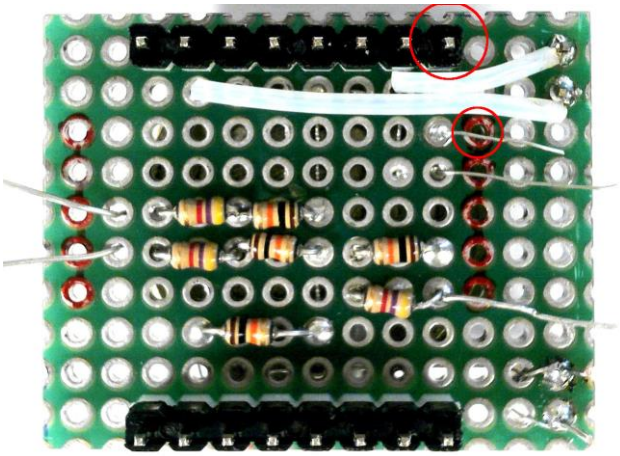
The “ESP” side again, with component labels. The horizontal rows are for the ESP8266. The vertical rows are for the MAX module.



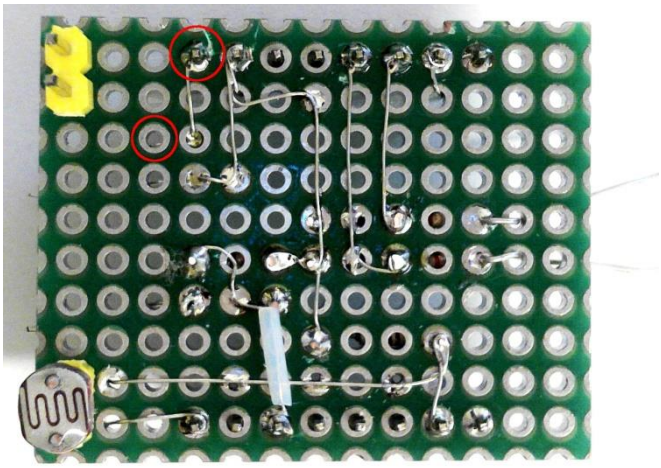
Header pins with spacers taken from another row of header pins to ensure adequate spacing for components between the boards. The header pins for the ESP8266 and the Max module are treated in the same way.



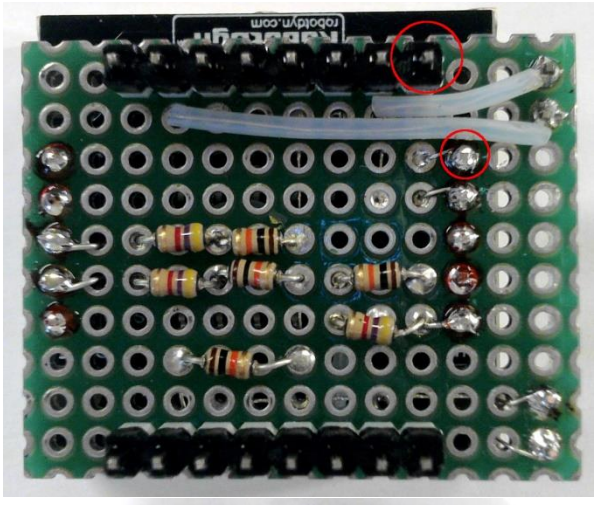
The “ESP” side again, now with the small components mounted and the header pins for the ESP8266. Before soldering these header pins to the board, temporarily mount the ESP8266 to ensure the correct alignment of the pins. Again, the larger red circle indicates +5v of the ESP8266 board. The smaller red circle indicates the +5v (in) of the MAX module.
The LDR connection is in the bottom right hand corner and the 2 pin connector in the top right corner.
Note the 5 flying wires prepared for the connection to the MAX module.



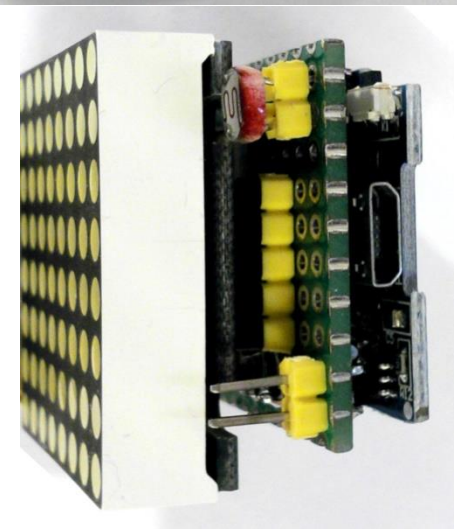
The “MAX” side of the board flipped along a vertical axis showing the connections. Refer to the schematic if a connection is not clear from this picture.



The MAX module has now been soldered in and 5 connections made to it. Ensure the correct orientation!
You can just see a part of the back of it on the top edge of the image.



The next step is to solder the ESP board to the header pins prepared for it. Again, ensure the correct orientation !
Before doing so, it is recommended to do a final test of the assembly since you will not easily be able to make any corrections after this.



5 USER MANUAL

5.1 INITIAL SET UP

This describes the basic set up of an unconfigured or 'factory condition' device.

- Compile and load the sketch to the ESP8266 then open the serial console at 115200 baud.
- The device will be in forced into AP mode, so connect your smartphone or WLAN capable PC to the SSID in the display (by default SpeakCL)
- Open a web browser and enter the IP Address seen on the display (by default 192.168.4.1)
- Follow the steps in "Web Configuration" below to enter the credentials of your WLAN (SSIP and PSK) and timezone information.
- Power the device off, then on again and it should, after a few seconds, fetch the correct time from the configured NTP server.
- When the device is joined to your WLAN, you can now contact it using the IP address which can be displayed on the LCD screen by pressing the control button for more than 8 seconds.

5.2 WEB CONFIGURATION

Web configuration mode is entered by typing the device's IP address into a web browser in the same WLAN.

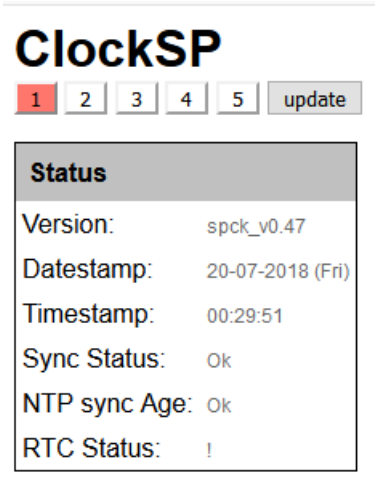
If the device has not joined an existing WLAN, i.e. the device is in AP Mode, then it will be first necessary to connect the smartphone or WLAN capable PC to the network created by the device. (see Initial Setup above)

You can navigate between pages by selecting the page number from the top bar and make changes. Press 'Update' when you have completed your changes.

Configuration page 1

This shows the software version, date and time stamps at the last browser refresh. The sync status is OK if the device clock has got a valid time stamp from one of its sources.
NTP Sync Age is OK if the last NTP Sync was less than X minutes ago, otherwise a '!' is shown. The same with RTC status. Since NTP has a higher priority than the RTC, a '!' is usually shown for the RTC if the NTP is 'Ok'.

The device name in bold at the top, in this case ClockSP can be changed by the user. It is also the name of the SSID when the device is in AP mode.



Configuration page 2

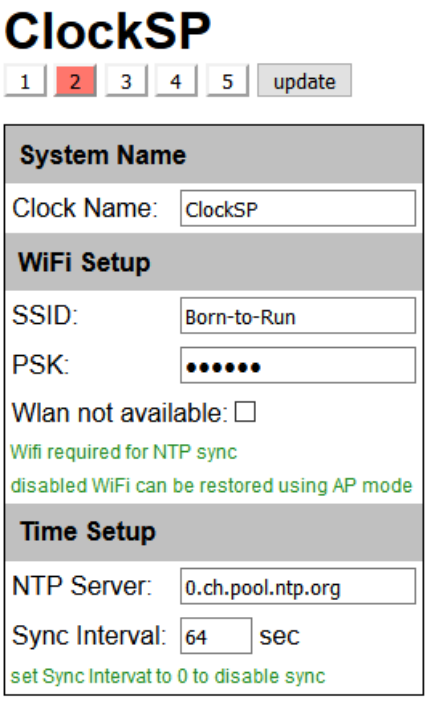
The clock name can be changed. It cannot be blank.

An SSID and PSK (password). These must be present even if there is no local WLAN. Just use the word 'dummy'

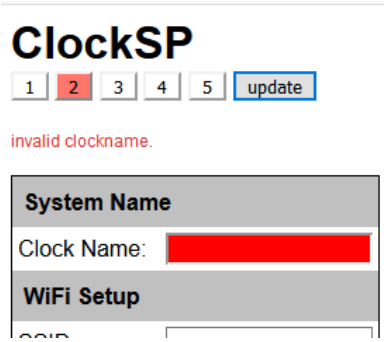
Once a PSK has been registered in the system, there is no need to reenter it each time you make a change on the page.

If no WLAN is available, click the box. Since NTP will not then work, it will then be necessary to set the time manually on page 3.

The default time server and sync interval are shown here. If you wish to disable NTP sync, set the time interval to 0.



Errors appear like this on update, in this case the clock name cannot be blank.

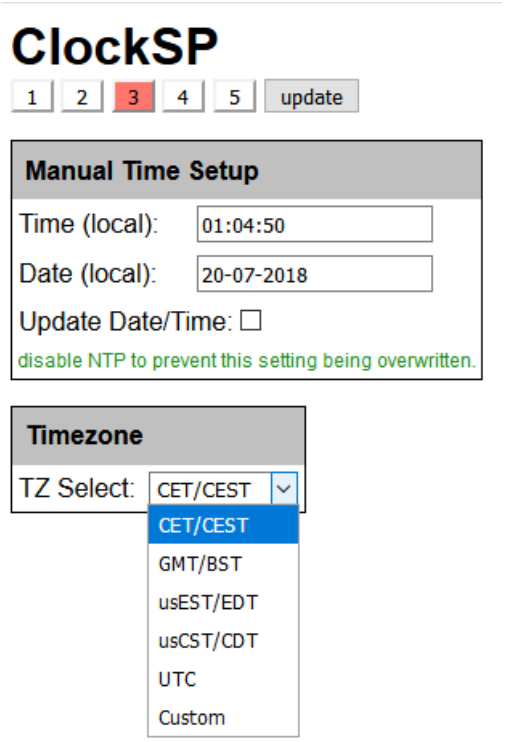


Configuration page 3

If you need to set the time manually, do it according to the example format here then click the checkbox and then update.

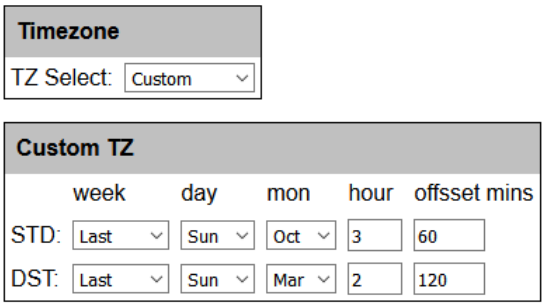
If NTP is left active (page 2) then any time you set manually will be overridden at the next NTP synchronization.

5 predefined timezones are configured. If yours is not in the list, select custom then enter the rules for your timezone as illustrated below.



On selection of Custom timezone, you see a panel similar to this.

In this example, standard time (STD) begins at 03:00 am on the last Sunday in October and the offset from UTC is +60 minutes.
Daylight Saving Time (DST) begins at 02:00 am on the last Sunday in March and the offset from UTC is +120 minutes.



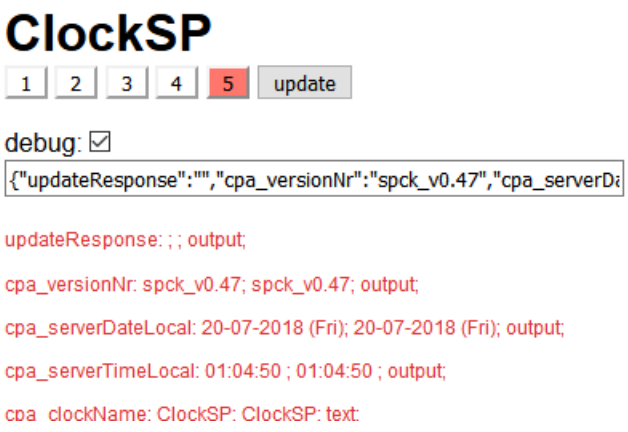
Configuration page 4

Blank. Any scroll option is not relevant for the

Configuration page 5

This is intended for advanced level troubleshooting.

Selecting debug shows the data transmitted between the browser and the web server



5.3 Control Button

The control button has a number of functions depending on the state of the system when it is pressed and the length of time it is pressed for. It is active on release.

In this design, the button has been replaced by a 2 pin jumper marked J1 on the schematic. Shorting the jumper has the effect of pushing the button.

5.3.1 Display IP address

If the clock is running in a normal state, that is configured and displaying the time, then pressing the button momentarily will force the clock to display the IP address it has been assigned by scrolling it across the display.

5.3.2 Force device into AP Mode

If the button is held in during system start for at least 8 seconds (but less than 20 seconds!), the device is put into AP Mode, that is it creates its own wireless lan and the SSID and IP address are shown on the LCD screen. The device remains in the mode until it is disconnected from the power source.

This is normally used for the initial configuration of the device through a smart phone or WLAN capable PC.

5.3.3 Force Factory Reset

If the button is held in during system start for at least 30 seconds, the device will go into a factory reset mode and all user configuration is erased. Any WLAN credentials which were entered into the device, or any other nonstandard configuration has to be re-entered in AP mode. This may be used in advanced trouble shooting, or for certain data privacy reasons.

6 TROUBLE SHOOTING TIPS.

1. If your hardware choice, particularly the ESP8266 Module or the display is different to that specified, suspect that first.
2. If you have made any changes to the software, always revert to the last known good version. Note: The design is particularly sensitive to any blocking code because the internal audio buffers have to be replenished every few milliseconds.
3. Look at the output on the serial console (115200 Baud) for indications of an error state. Add additional error messages as required within the source code and recompile.
4. You can comment out selected function calls from the loop of the .ino file to disable selected modules in an attempt to isolate the problem.

7 SOFTWARE DISTRIBUTION

Location: https://forum.arduino.cc/index.php?topic=614763.0

Tested with the following, but others could work:

- Arduino IDE v 1.8.2
- Arduino ESP8266 Core Software 2.4.1
- Wemos D1 Mini (4MB flash)

8 APPENDIX

A document collection, some directly referred to above or used in the solution and some generally relevant further reading.

[Ref 1] Arduino ESP8266 Speaking Clock

https://forum.arduino.cc/index.php?topic=559652.0

The NTP time synchronization and configuration modules used here were derived from the above speaking clock project