



VERBIS - Desktop 8x8 RGB LED Matrix Word Clock

Motto: In Verbis Virtus...(There is power in words)

 Difficulté **Moyen**

 Durée **6 heure(s)**

 Catégories **Décoration, Électronique, Mobilier, Maison**

 Coût **25 USD (\$)**

Sommaire

[Introduction](#)

[Video d'introduction](#)

[Étape 1 - Making the enclosure](#)

[Étape 2 - Making the electronics](#)

[Étape 3 - The Software. Programming the ESP-01 Board](#)

[Étape 4 - Configuring and Using the Clock](#)

[Étape 5 - Final Thoughts](#)

[Commentaires](#)

Introduction

There are many other Word Clock projects on the Internet so it is rather difficult to choose a project as a source of inspiration because each of them offered me useful ideas. But if I must choose one, it's going to be the one that started all, the beautiful project created by Doug Jackson, presented on Wikifab.

It's hard to bring something new with any future design, but hopefully my project will grab your attention with the fresh and cool ideas inside.

What are the pros of my project?

- very affordable, cheap electronics and other parts;
- opensource web server for easy management;
- accurate time from ntp server;
- it is possible to have multiple languages;
- minute by minute time indication (not by 5 minutes like standard word clocks), dynamic time display and amazing flexibility for future ideas...

The idea behind this Word Clock is to have all the words necessary for telling the time and to place these words in an 8x8 letters layout as in a word search puzzle. In this way, with a small number of leds we can tell the time minute by minute. Because the words are placed randomly you can tell the time by reading the flashing words each after another (the words are already displayed on the clock face). I started with using only one color, red, also green 8x8 matrix led arrays.

Sadly, the electronics were a little bit complicated when using the MAX7219 led driver and the ESP12 module of ESP8266 microcontroller, the wiring was tedious and prone to errors. But with the WS2812 RGB Led Matrix things became much easier (and cheaper ...). The schematics are simple, there is no need for the esp12 board, an ESP-01 module can be used. Also the flashing words are gone, I used color change to accentuate the words for telling the time (using many colors was a big deal).



Matériaux

Enclosure:

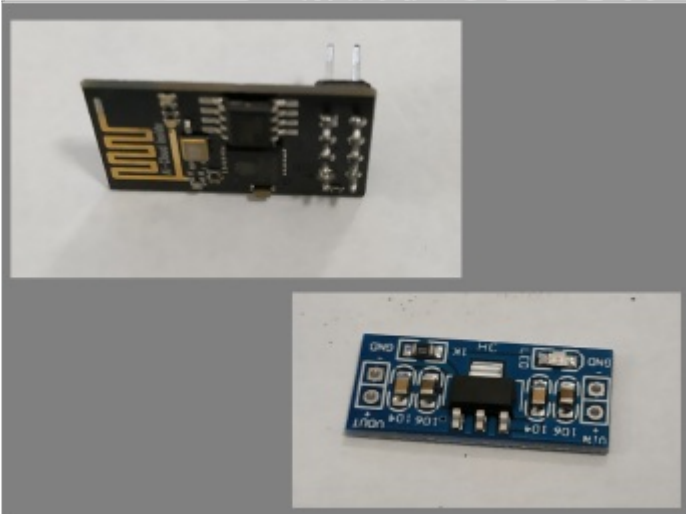
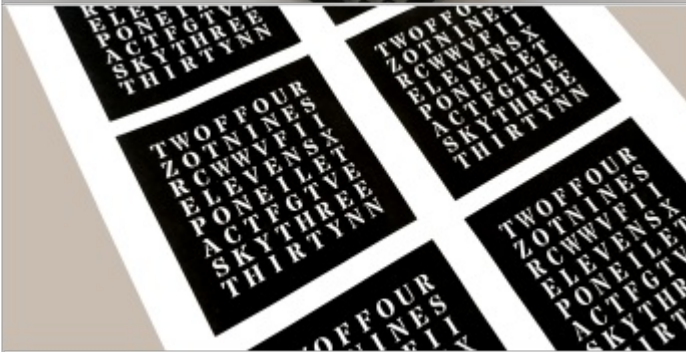
1. wooden photo frame (80x80mm interior) thin frame (1cm);
2. 3mm plexiglass support;
3. 80x80mm 3mm grey smoked transparent display plexiglass sheet;
4. 3D printed 80x80mm 8x8 plastic grid;
5. letter layout 80x80mm Printed Paper Sheet;
6. 3d printed plastic box for electronics;
7. 2mm diameter 10mm length screws.

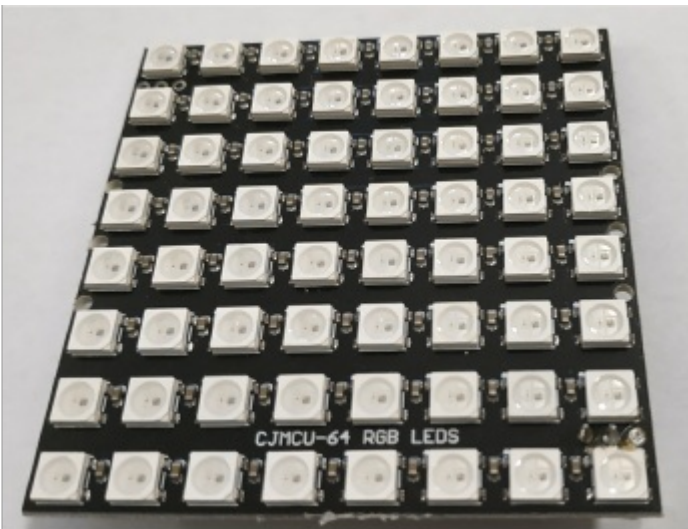
Electronics

1. ESP-01 Board;
2. 5v to 3.3v stabilizer module;
3. 80x80mm RGB 8x8 LedMatrix;
4. 3pin 2.54mm Pitch XH Pin Header+Connector;
5. Female DC Conector cable 5.5mmx2.1mm(2.5mm);
6. 5v/1.5A Power supply with 5.5mmx2.1mm(2.5mm) Male connector.

Outils

- drilling machine
- drill bits
- screwdriver
- plastic clamps
- electric soldering iron (station)
- hot glue gun





- 📄 En letter layout.svg
 - 📄 Ro letter layout.svg
 - 📄 Hu letter layout.svg
 - 📄 De letter layout.svg
 - 📄 Empty.svg
-

Étape 1 - Making the enclosure

Wooden photo frame

You can make your own frame, there are a dozen of articles about this, even here on Instructables. But a simpler solution would be finding a framing company where you can order a personalized frame with your required dimensions and you can choose from many frame types. This is exactly what I did. I ordered my frames with a specific dimension: the framed photo, in my case object (display) is 80x80mm. I also asked for an accurate dimension, I didn't want the frame to be too big for the 3D printed plastic grid.

Plexiglass support

The plexiglass support can be also be made DIY but for an amateur it is not very easy to cut and blend plexiglass. So I ordered several supports from an advertising company that makes all kinds of plexiglass objects. The dimensions I used are: width - 120mm, first part length - 180mm, second part length - 50mm, 15° bending angle.

Display plexiglass sheet

The 3mm grey smoked plexiglass sheet can be cut from a bigger sheet, obtaining the required 80x80mm dimension.

Plastic grid

The STL file for 3D printing can be downloaded from Tinkercad

Display Printed Paper Sheet

The SVG file for the Printed Paper Sheet is attached, and it can be edited with Inkscape. You can make your own display layout based on this SVG file, I used Word Search Construction Kit software to generate a words layout for the time display. You can print the file repeatedly on the same sheet of paper to achieve a good, opaque, black background. I got very good results with a cheap inkjet printer and standard white copier paper. I cut off the layout with a pair of scissors.

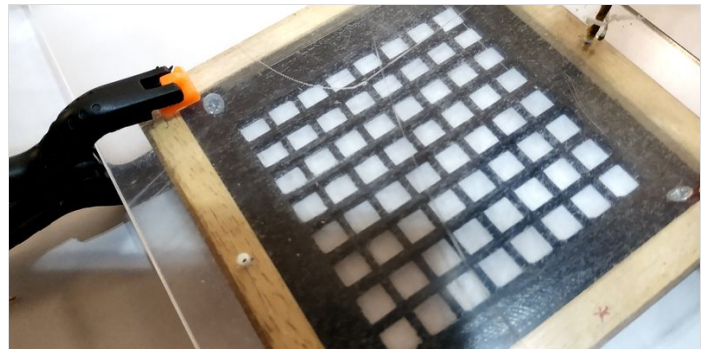
Plastic box for electronics

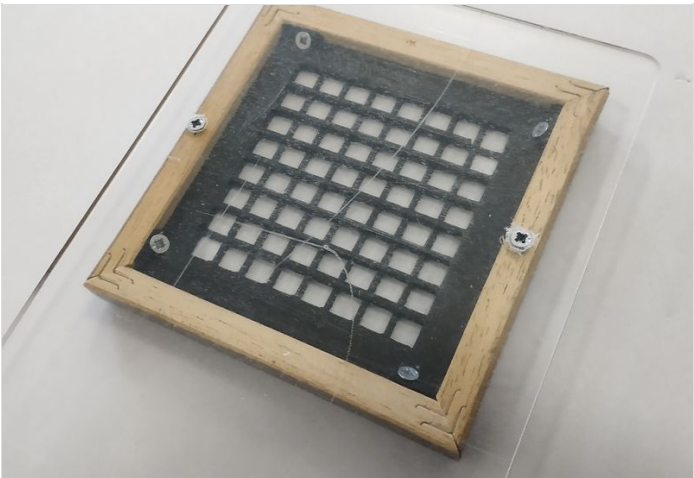
The files that you can 3d print are also on Tinkercad. I used some already purchased jewelry boxes, I only designed a new box base because the boxes were too tall. The files on Tinkercad are based on this type of boxes.

Detailed instructions (follow the images above)

- choose (and mark) a side of the frame to be the top of the clock, clean the smoked plexiglass sheet, put it in the frame;
- place the printed paper sheet and the 3D printed grid;
- drill with 2 mm diameter bit through the plastic grid to make room for the screws in the frame;
- screw the plastic grid;
- mark on the frame the place for holes and lock the frame to the plexiglass support;
- drill the holes with a 2mm diameter bit (enlarge the holes in the support with a 3mm diameter bit, make the coining with a 10mm diameter bit) and screw it all together.

The last image shows an almost finished enclosure.





Étape 2 - Making the electronics

ESP-01

Cheap and versatile microcontroller module with WiFi capability, if you don't know about it read this good how-to [A Beginner's Guide to the ESP8266](#).

3.3v stabilizer module

The ESP-01 requires to be powered with 3.3v, I used here a 4pin module.

Led Matrix 64led RGB Matrix with WS8212 IC

You can read more in [Getting Started With NeoPixel / WS2812 RGB LED by Open Green Energy](#).

3 Pin Header and connector

I used this connector because it permits easy assembly-disassembly of the enclosure.

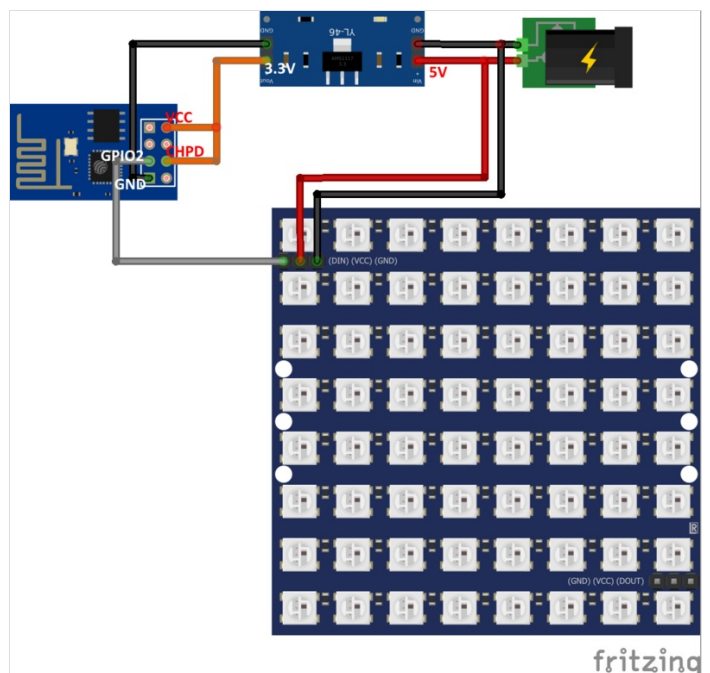
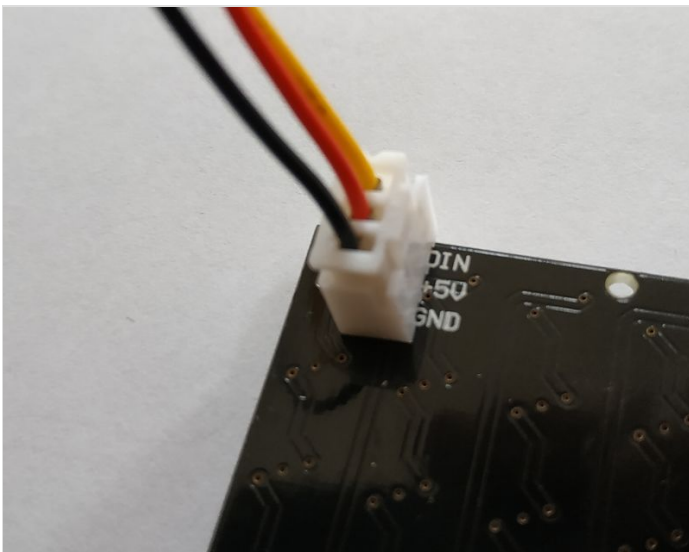
DC connector and DC power supply

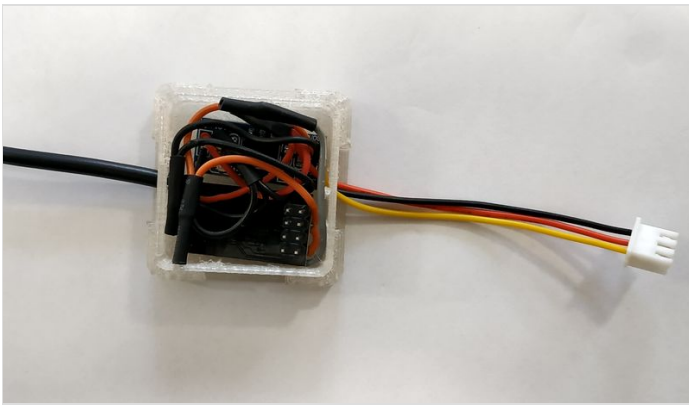
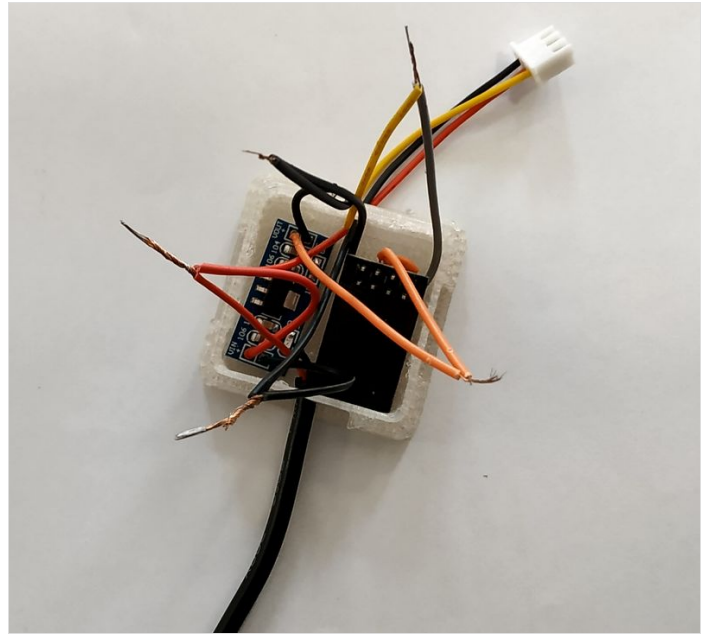
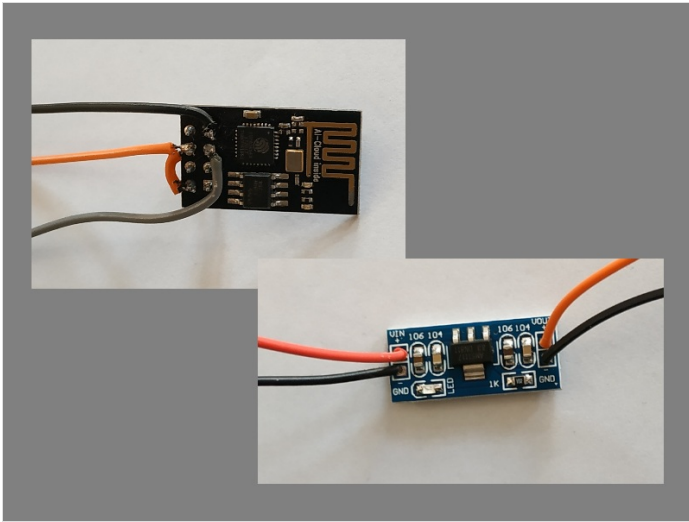
The power supply is 5v and 1.5A maximum, it is rather enough because not all leds are lightened up at full brightness and full white. Also I opted for a separate DC connector because it is simple to replace a defective power supply.

Schematics - Very simple, made with Fritzing, see the image above.

Detailed instructions

- solder the 3pin connector to the LED Matrix;
- (optional) change the order of the red and black wire in the header (it is good to respect some rules for colors used in wirings, so a red color wire for 5v and a black color wire for ground);
- drill a hole in the plexiglass support where the 3pin connector is located and enlarge the hole (with a Dremel for example) to accommodate the 3pin connector;
- make the modules soldering like in the image;
- place the modules, the power and the XH wire ends in the plastic box;
- twist and solder together the wire ends; isolate and strengthen the twisted wire ends with 2mm interior heat shrink tubing;
- fix the Led Matrix to the 3D printed plastic grid (with a hot melt glue gun);
- assemble all :)





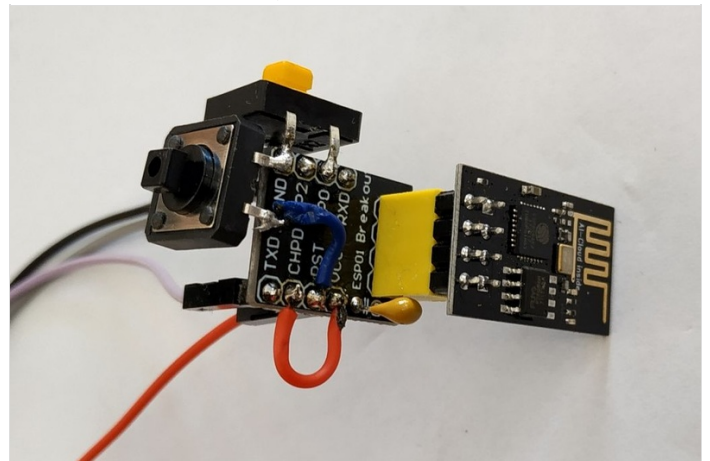
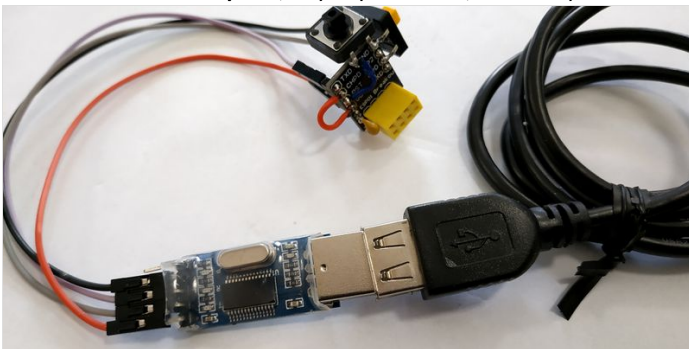
Étape 3 - The Software. Programming the ESP-01 Board

There are several ways for programming the ESP-01 board, my way for flashing the ESP-01 is using an adapter with a PL2303 chip (USB to Serial converter). Also to make the programming easier, I used a breadboard adapter like in the image and I connected it to the USB to Serial converter. You can see the wiring between these two modules: 3.3v-RED, Ground-BLACK, RX from the converter to TX on adapter-PURPLE, TX from the converter to RX on adapter-GREY. On the adapter I soldered a switch (PROG) between GPIO0 and GND pins and a switch (RESET) between RST and GND pins.

Step by step instructions:

- download and install the Arduino IDE;
- download the source for the Verbis word clock;
- install in Arduino IDE the ESP support libraries - more here;
- insert the ESP-01 Board in the breadboard adapter and connect the USB to Serial converter to an USB port of the computer;
- open in Files-Examples-ESP8266 the CheckFlashConfig sketch and then make the configuration options in TOOLS like in the image (you will have another COM port eventually);
- open a serial monitor (TOOLS - SERIAL MONITOR), make the configs (Both NL&CR, 115200 baud) and push the RESET switch on the breadboard adapter;
- you will see the 'ready' word on the last line, it means the ESP Board is working and you can find on your phone a new WiFi Access Point;
- to put the ESP board in programming mode, gently push the PROG switch and keep pushing, then push and release the RESET switch, THEN(!) release the PROGRAM switch, in the SERIAL MONITOR you must see a garbled line after the ready line;
- click the UPLOAD button in the Arduino IDE, wait for the sketch to be compiled and uploaded, then verify the SERIAL MONITOR again, if you have a line with 'Flash Chip configuration ok' then the programming options in the Arduino IDE are good. If no, make the necessary changes;
- unzip the sources and open, with the Arduino IDE, the VerbisMain.ino file, put the ESP board in programming mode and upload the program, if everything is good you will see in the SERIAL MONITOR the word clock's initial configuration and a WiFi Access Point on your phone.

The programming of the ESP-01 board can be made before you solder any wire to the module (see step 2 - The Electronics) but can also be made when the soldering is already made (everything assembled in the electronics box). In this case **remove the RED wire between the converter and the adapter** (very important !!!) because you will use 3.3v from the stabilizer module, not from the USB to Serial Converter.



```
VerbisMain | Arduino 1.6.9
File Edit Sketch Tools Help
Auto Format Ctrl+T
Archive Sketch
Fix Encoding & Reload
Serial Monitor Ctrl+Shift+M
Serial Plotter Ctrl+Shift+L

Board: "Generic ESP8266 Module"
Flash Mode: "DIO"
Flash Frequency: "40MHz"
CPU Frequency: "80 Mhz"
Flash Size: "512K (64K SPIFFS)"
Debug port: "Disabled"
Debug Level: "None"
Reset Method: "ck"
Upload Speed: "115200"
Port: "COM3"
Get Board Info
Programmer: "ArduinoISP"
Burn Bootloader

#ifdef DE_clock
// Include localized hour generation functions
#include "DE/HourStrings_DE.h"
#include "DE/TimeGen_DE.h"
// Include the localized HTML "Pages"
#include "DE/Page_Admin_DE.h"
#include "DE/Page_WTFSetsings_DE.h"
#include "DE/Page_Information_DE.h"
#include "DE/Page_NetworkConfiguration_DE.h"
#include "DE/Page_SetTime_DE.h"
#include "DE/Page_DisplaySettings_DE.h"
#endif
#endif EN_clock
```

```
VerbisMain | Arduino 1.6.9
File Edit Sketch Tools Help
VerbisMain | VerbisMain | Page_Style.css.h | global.h | Page_Style.css.h | global.h | Page_Style.css.h | global.h
1 #define DE_clock //clock language
2 #define PAGEID_VERBIS_WTFS_CONFIG 0
3 #include "PageID.h"
4
5 // ESP8266 template with phone config web page
6 // based on BVE_MiniConfig_V1 from Andreas Spies
7 // and ESPBASE from Pedro Albuquerque https://github.com/Pedroalbuquerque/ESPBASE
8 //
9
10 #include <ESP8266WiFi.h>
11 #include <ESP8266WebServer.h>
12 #include <WiFi.h>
13 #include <Timer.h>
14 #include <EEPROM.h>
15 #include "global.h"
16 #include "DE.h"
17 // Include STYLE and Script "Pages"
18 #include "Page_Admin_DE.h"
19 #include "Page_Information_DE.h"
20 #include "Page_NetworkConfiguration_DE.h"
21 #include "Page_SetTime_DE.h"
22 #include "Page_DisplaySettings_DE.h"
23 #endif
24
25 #ifdef DE_clock
26 // Include localized hour generation functions
27 #include "DE/HourStrings_DE.h"
28 #include "DE/TimeGen_DE.h"
29 // Include the localized HTML "Pages"
30 #include "DE/Page_Admin_DE.h"
31 #include "DE/Page_WTFSetsings_DE.h"
32 #include "DE/Page_Information_DE.h"
33 #include "DE/Page_NetworkConfiguration_DE.h"
34 #include "DE/Page_SetTime_DE.h"
35 #include "DE/Page_DisplaySettings_DE.h"
36 #endif
37 #endif
```



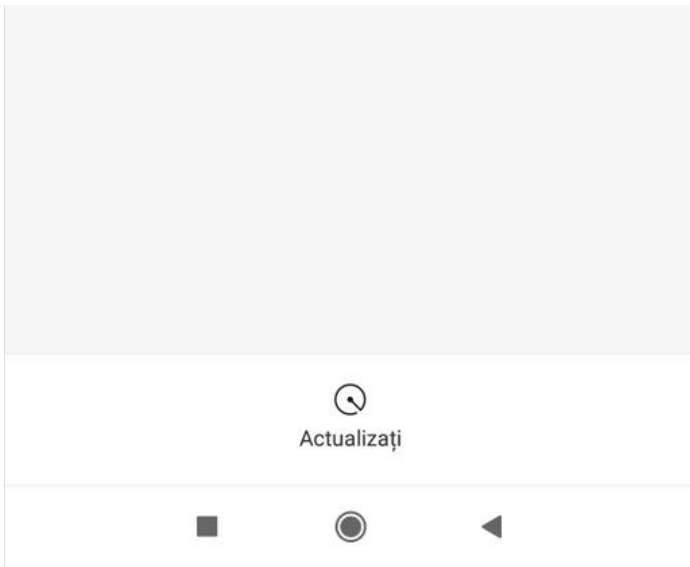
```
VerbisMain | Arduino 1.6.9
File Edit Sketch Tools Help

VerbisMain | LedMatrix_functions.h | NTP.h | Page_Script.js.h | Page_Style.css.h | global.h

1
2 #define EM_clock //clock
3 #define FASTLED_INTERRUPT
4 #include "FastLED.h"
5
6 // ESP8266 template with
7 // based on BVE_WebConfig
8 // and ESP8266 from Pedro
9 //
10 #include <ESP8266WiFi.h>
11 #include <ESP8266WebServer.h>
12 #include <Arduino.h>
13 #include <Ticker.h>
14 #include <ESP8266.h>
15 #include "global.h"
16 #include "NTP.h"
17 // Include STYLE and Script
18 #include "Page_Script.js.h"
19 #include "Page_Style.css.h"
20 #include "LedMatrix_functions.h"
21
22 #ifdef DE_clock
23 // Include localized hour
24 #include "DE/SourStrings_DE.h"
25 #include "DE/TimeGen_DE.h"
26 // Include the localized
27 #include "DE/Page_Admin_DE.h"
28 #include "DE/Page_NTP_DE.h"
29 #include "DE/Page_Information_DE.h"
30 #include "DE/Page_Network_DE.h"
31 #include "DE/Page_SetTime_DE.h"
32 #include "DE/Page_DisplaySettings_DE.h"
33 #endif
34
35 #ifdef EM_clock
36 // Include localized hour generation functions
37 #include "EN/SourStrings_EN.h"
38 #include "EN/TimeGen_EN.h"
39 // Include the localized HTML "Pages"
```

```
COM3
Reading Configuration
Configuration NOT FOUND!!!!
Setting AP mode default parameters
Wifi ip:192.168.4.1
HTTP server started
Printing Config
BMCPI:
DayLight:1
HourFormat:0
TimeDisplay:1
NTP update every 3 sec
Timezone 20
IP:192.168.1.100
Mask:255.255.255.0
Gateway:192.168.1.254
SSID:VERBIS-664c2a
PWD:
ntp ServerName:0.ro.pool.ntp.org
Device Name:VERBIS
Device Name:#0000FF
Device Name:#FFFFFF
Fastled Setup done
Autoscroll
Both N. & CR
115200 baud
```





Étape 4 - Configuring and Using the Clock

After correctly flashing the ESP-01 module, start the clock. On the display you will see an animation like this.

You will find a new WiFi Access Point on your phone called VERBIS and a serial (a bunch of numbers). You can connect to this Access Point, with no password, open a browser (Chrome is preferred) and go to <http://192.168.4.1> page. (When the ESP-01 is in Access Point mode it will always have the IP 192.168.4.1 This IP is hardwired in the microcontroller's firmware).

You will see now the word clock's main menu. Go to the 'NETWORK CONFIGURATION' submenu and wait for the search for the surrounding routers to finish. Select your router's name from the list (in my case the 'mi' router), the name will be copied to the 'SSID' textbox, enter the password and push the 'SAVE' button. Wait a little bit and try to reconnect to the clock on the same IP (192.168.4.1). Select 'NETWORK INFORMATION' where you can find what IP the clock has gotten. Restart the clock (unplug and plug the power adapter) and you will see this pattern on the display connect to your router normally and use the IP found from 'NETWORK INFORMATION' submenu.

The clock is trying to connect to the router and the ntp server. If there is connection with the ntp server, the word clock's internal time will synchronize with the NTP server's time and the corresponding words will appear on the display. If there is no connection, the pattern will continue to show. The clock will try to reconnect to the NTP server each minute until a successful connection.

If you have no router you can access the 'MANUAL TIME SETTING' submenu and enter the time in the corresponding text boxes. Also if the clock is connected to a router but it doesn't have a connection to the ntp server, you can use this submenu to set the time until a time synchronization. In this case, when the connection to the ntp server will occur, the time will synchronize automatically in the background. The 'NTP SETTING' submenu options are pretty explanatory, I don't have too much to say about them...

If you access the 'DISPLAY SETTINGS' submenu you can change the two colors between which the transition occurs when the words are displayed. Also, you can change if the words 'IT IS' are displayed or not at the beginning of the time telling.

13:46 ●

0,3KB/s 📶 📶 🔋

📄 192.168.4.1

3



VERBIS Administration

NETWORK CONFIGURATION

NETWORK INFORMATION

NTP SETTINGS

SET EXACT TIME

DISPLAY SETTINGS



13:48 0,0KB/s

192.168.4.1/config.html

< Network Configuration

Connect to Router with these settings:

SSID:

Password:

DHCP:

IP: . . .

Netmask: . . .

Gateway: . . .

SAVE

Connection State:
CONNECTED

Networks:

Found 4 Networks

Name	Quality	Enc
mi	100%	*
vsm61	92%	*
DIGI-02347276	24%	*
VASY-WiFi	20%	*

REFRESH

13:48 0,0KB/s

192.168.4.1/info.html

Network Information

SSID : mi
IP : 192.168.43.51
Netmask : 255.255.255.0
Gateway : 192.168.43.1
Mac : 5C:CF:7F:66:4C:2A

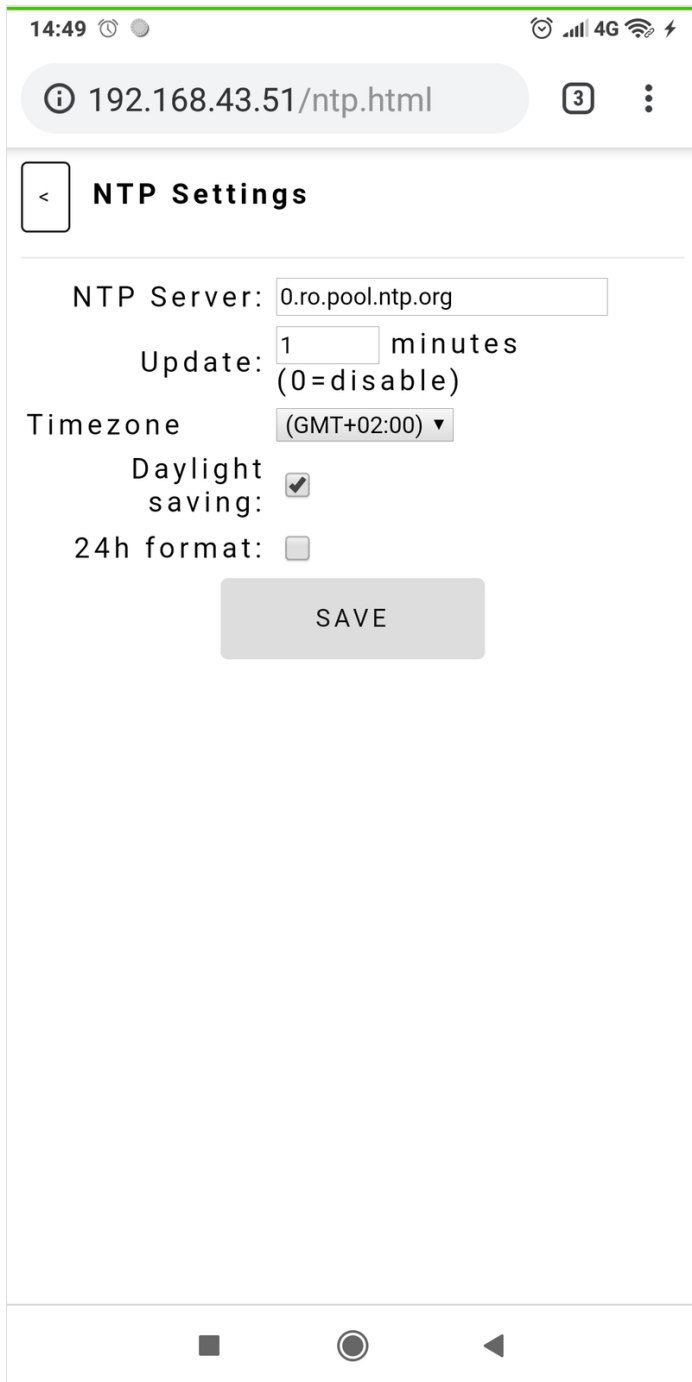
9:58 0,1KB/s

192.168.4.1/time.html

Manual time setting

Hour:
Minute:
Second:

SAVE



Étape 5 - Final Thoughts

What's next?

An alternate enclosure can be made using a wide wooden photo frame, you can see a rendering and an actual image above. The 15° tilt is made with a small plexiglass support.

The electronics can be drastically simplified by using a WS2812 RGB LED Controller Module (in the image with the wide photo frame the clock is made with such a module). The box used in this case is also on Thinkercad. Of course, you will need an ESP-01 module for this approach.

If you have access to a laser cutter you can make your paper sheet by laser cutting a black sheet, use stencil fonts in this case.

You can make a VERBIS word clock also in other languages. I have made the letter layout and the programming for german, hungarian and romanian. Change in the first line of the source code the language (DE_clock, RO_clock or HU_clock). The printable layouts are attached too. But why stop here? Print a paper sheet (empty sheet in files) and you can make a binary clock, you can make a Fibonacci clock, but also a digital clock (with the hours and minutes colors changing alternatively similar to VERBIS word clock).

Of course you need to modify the ESP-01 firmware but I think it is not so difficult if you already have a template you can rely on.

But many many other projects are possible with this cheap and easy to build design.

You can find more photos and info at [projects instructable](#).

