Build

U3 needs a steady hand and good eyes. If U3 is a bridge too far there's the option to exclude U3 and related parts and install C8 & RV1. U3 is the best option. Refer to the schematic for more info.

1. Install U3, the SN74LVC1G125
2. Install the 5V regulator and associated parts
3. Test 5V and 12V are where you'd expect
4. Install all the remaining parts

Winding T1

Wind 10 turns bifilar onto a FT37-43 toroid.

A - A is one winding (wire)

B – B is the other winding (wire)



Building a LPF

The design uses the QRP Labs LPF, see https://qrp-labs.com/lpfkit.html for detail.

ESP8266

The ESP8266 variant used is the ESP-12F on a NodeMCU development board.

https://www.aliexpress.com/item/1005007470886370.html

Si5351

The Si5351 development board is everywhere.

https://www.aliexpress.com/item/4000040035169.html

SN74LVC1G125

https://www.aliexpress.com/item/1005009595297159.html

Toroids

https://intelec.nz/

Configure

There are three parts to configuration:

1. Program the ESP8266
2. Establish a Wi-Fi connection
3. Calibrate the Si5351 frequency to 10MHz (or as close as you can)

Program the ESP8266

There are two options here:

1. Use the Arduino IDE
2. Use https://esptool.spacehuhn.com/

Both methods require the ESP8266 to be connected to a computer via USB. Do not connect 12V.

Use the Arduino IDE

Open and compile the .ino.

The board manager settings are noted in the comment at the top of the .ino.

The required libraries are noted next to the #include statements in the .ino.

If you get really really stuck, email marcbranch22@gmail.com.

Use https://esptool.spacehuhn.com/

This is a web tool, all you need is the .bin.

Follow your nose with the web tool, it's pretty intuitive.

If you get really really stuck, email marcbranch22@gmail.com.

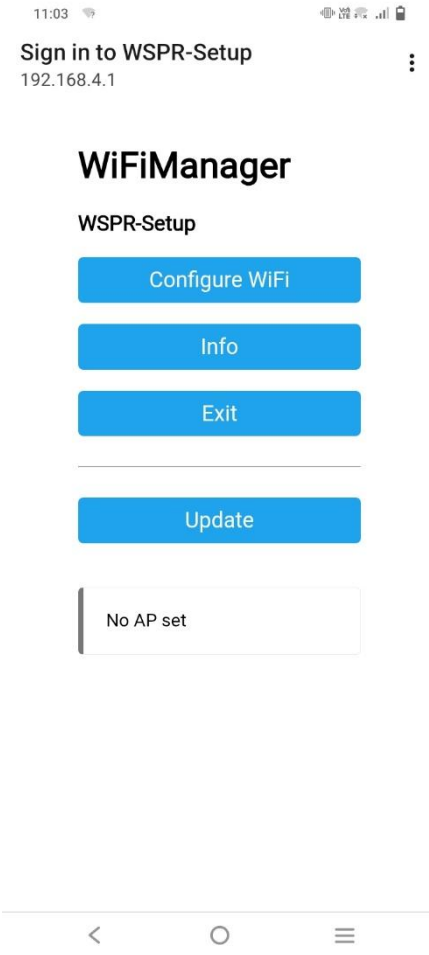
Wi-Fi connection

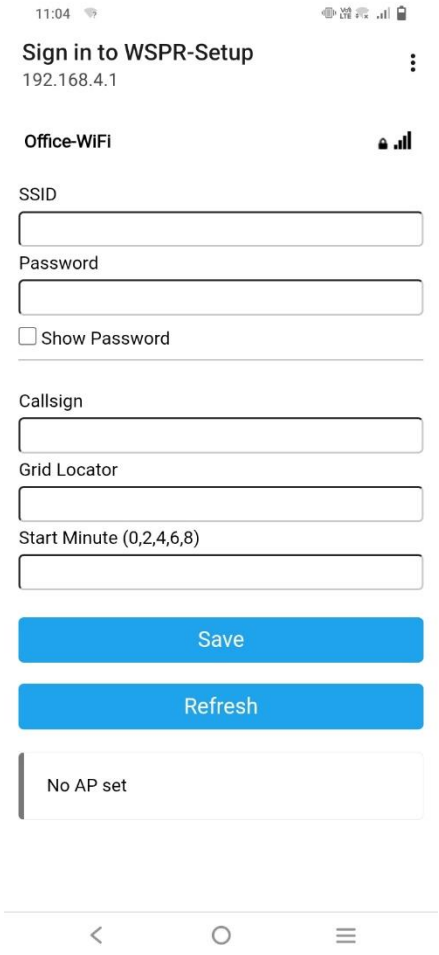
The Wi-Fi needs to be setup first. The software will not run until the ESP8266 has connected to your Wi-Fi. This is a one time job, the ESP8266 saves the connection info after successfully connecting.

While the ESP8266 is establishing a connection a blue LED will blink on the ESP8266 board, once connected the blue LED goes out.

First, connect power. This can be via the 12V connector or via the USB on the ESP8266, but don't have both power sources connected … 5V from the U3 regulator will feed 5V back to whatever the USB is connected to (computer/laptop?) and offers the possibility of damaging the USB device. Unlikely but not impossible.

To connect the ESP8266 to Wi-Fi you need to configure via a device that is already connected to the Wi-Fi, your phone is a good option. On your device (eg phone) go to settings etc, to the place you add a Wi-Fi connection. You will see a new device called "WSPR-Setup", click/tap on it. That will open WiFiManager, click/tap Configure WiFi, that will open a page where you enter the Wi-Fi and WSPR info.





SSID and Password are what they are on your Wi-Fi.

Callsign and Grid are your details. WSPR uses a 4 char Grid (not 6 char).

Start Minute

WSPR messages can be transmitted every two minutes, on any even minute, at second zero. It was decided that transmitting every two minutes is a bit silly so the software only transmits every ten minutes.

The Start Minute box sets the even minute, eg 4 means transmit on minutes 4, 14, 24, etc.

Save

When all entered, click/tap Save, the ESP8266 will connect to your Wi-Fi.

The LED will stop blinking and likely turn off. Perfect, you're connected.

If, by chance, the LED goes from blinking to on, have a look at the time. You've connected to Wi-Fi on the even minute that matches your transmit start minute. Have a chuckle, no problem, the LED will turn off after 110 seconds.

Calibrate the Si5351

Set the band jumpers to Calibrate.

Connect the ESP8266 to a computer via USB. Do not connect 12V.

Connect a frequency counter to TP1. If you don't have a frequency counter, clip a length of wire to TP1 and monitor on an accurate receiver, SDR, etc.

Calibrate via the Arduino IDE Serial Monitor

Click on tools->serial monitor. Select 115200 baud.

Press (click) the tiny reset button on the edge of the ESP8266 module.

Follow onscreen instructions.

Remove the USB cable (and counter).

Calibrate via https://terminal.spacehuhn.com/

Settings: 115200 baud, line ending \n, all three boxes ticked. Note the \n is not the default and needs changing and saving.

Connect and follow your nose.

Sometimes (randomly) you need to reset the ESP8266.

Remove the USB cable (and counter).

Use

The blue LED has three states:

- Blinking - establishing a Wi-Fi connection
- On – transmitting
- Off – idle, waiting for the next transmit cycle

The jumpers, both band and power, are read by the ESP8266 on startup only. If you change the jumpers without a reset the transmitter will carry on based on the jumper settings at startup.

To change band or power, set the jumpers, then reset the ESP8266. A reset can be done by removing and then reconnecting the 12V … or just click the reset button on the edge of the ESP8266 board. The blue LED will blink indicating the reset.

Idle current

The transmitter draws approx. 130mA in idle mode, 60mA of that is the ESP8266 Wi-Fi and 20mA through the resistor divider.

T1 and PA limitation

The Si5351 and SN74LVC1G125 work happily 2200m thru 2m. The PA … not so much.

The PA as specified has been tested (and works) 80-10m. I suspect it will work 160m and with a bit of luck 6m.

For lower bands my guess is more turns on a (say) FT50-43 will likely work. For the higher bands, perhaps replace T1 with an air wound inductor. These bands are for the intrepid experimenter.

C8 and RV1

If the builder uses this option, I suggest max power of 100mW, set by RV1 (the power jumpers have no impact). Put a 50 ohm load on the output and measure approx. 6.3Vpp with an oscilloscope.

The RV1 setting will be different on each band due to the Si5351 amplitude being frequency dependant.