



WSPR on Raspberry Pi

Session 12 – WSPR on a Raspberry Pi

This session explores how to install and run WSPR on a Raspberry Pi. We will look at TAPR's easy solution and how to roll your own including a low pass filter.

Table of Contents

WHAT IS WSPR?	1
WHAT IS WSPRNET?.....	2
HOW WOULD WE USE IT?.....	2
BILL OF MATERIALS	2
DO-IT-YOURSELF	3
INSTALL WSPRRYPI.....	3
RUN WSPRRYPI	4
TEST WSPRRYPI	5
ADD LOW PASS FILTER	6
TAPR WSPR	6
RESULT OF THE WSPR TEST ON MY 40M OCF DIPOLE	7
WHAT'S NEXT	12
APPENDIX	12

WHAT IS WSPR?

WSPR (pronounced "whisper") is an acronym for Weak Signal Propagation Reporter. It is a protocol, implemented in a computer program, used for weak-signal radio communication between amateur radio operators. The protocol was designed, and a program written initially, by Joe Taylor, K1JT, and is available as part of the WSJT-X software package. The software code is now open source and is developed by a small team. The program is designed for sending and receiving low-power transmissions to test propagation paths on the MF and HF bands.

We will examine using a different open source program written for the Raspberry Pi.

WSPR implements a protocol designed for probing potential propagation paths with low-power transmissions. Transmissions carry a station's callsign, Maidenhead grid locator, and transmitter power in dBm. The program can decode signals with a signal-to-noise ratio as low as -28 dB in a



WSPR on Raspberry Pi

2500 Hz bandwidth. Stations with internet access can automatically upload their reception reports to a central database called WSPRnet, which includes a mapping facility.

An accurate clock is essential both for transmission and decoding of received signals. The Pi uses NTP for its clock.

WHAT IS WSPRNET?

WSPRnet is a database containing reception records from WSPR receivers. We need both a transmitter and receiver stations to use WSPR. The receiver is included as part of WSJT-X software, available at <https://wsjt.sourceforge.io/wsjitx.html>

WSJT-X software includes several modes including WSPR and FT8 and FT4, and many of us use FT8/FT4 as part of our station operations. You can also run in WSPR receiving mode and hear WSPR transmitters and report your receptions to the WSPRnet.

WSPRnet.org also includes a mapping facility to display spots in a variety of ways. There are also another 6 or so websites that provide alternate mapping and data.

HOW WOULD WE USE IT?

We would implement a WSPR transmitter on one or more HF frequencies. Typically, you would operate your transmitter for 24 hours or more so you can see where your signal is received. You might also want to operate during the daytime and then again at night to see the difference.

By examining the map of where your transmitter is received, you can get a feeling for where your station works for reception, both local and DX. This is a good way to test where the propagation of your antenna works well. You can try your transmitter on a variety of antennas to see how propagation varies by frequency and distance.

BILL OF MATERIALS

Workshop documents are available at <https://github.com/hwdornbush> For this session, select Repositories > 12-WSPR.

For this and all workshops, any revised documents will be available here.

- Raspberry Pi 3 or 4
- Power supply, 5v, 2.5-5 amps



WSPR on Raspberry Pi

- MicroSD card, at least 16 GB. Choose one size and type of card and stick to it, such as Sandisk Ultra 32GB, which is what I use.
- Optional: case, fan, heatsinks
- One or both of:
 - HDMI Monitor, keyboard, mouse with MicroHDMI to HDMI adapter cable (Pi 4) or HDMI to HDMI cable (Pi 3)
 - Computer to use as an SSH or VNC client.
- Computer to load the system image on a MicroSD card
- USB to MicroSD card reader
- AN HF antenna for the band(s) you want to test with WSPR
- Cable to connect the antenna to the Pi
- For the TAPR easy solution, a TAPR WSPR Kit or WWOT (WSPR Without Tears) assembled module. (<https://tapr.org/product/wspr/>)
- For the do it yourself solution,
 - A low pass filter kit, such as <https://shop.qrp-labs.com/LPF>
 - Pushbutton switch
 - LED and 220 ohm resistor
- For testing the solutions, we will use a NanoVNA and TinySA.

DO-IT-YOURSELF

WsprryPi is a software program that runs on a Raspberry Pi and generates the WSPR transmissions. It can be implemented on any Pi. The signal that it generates is a square wave, so there are lots of harmonics, which we know is a bad thing.

We will install WsprryPi on a Pi and examine its output using a TinySA spectrum Analyzer. We will then install a low pass filter from QRP Labs and examine the improved output.

We will use a NanoVNA to analyze the effectiveness of the low pass filter.

INSTALL WSPRRYPI

The software and documentation is at <https://wsprry-pi.readthedocs.io/en/latest/>

You will find documentation about WSPR, Wsprry-Pi, and more here.

The software is installed using

```
$ curl -L installwspr.aa0nt.net | sudo bash
```

This will copy the software needed and install it on your Pi OS. The Pi OS can be any version so for this workshop, I will use Raspberry Pi OS Legacy (32 bit) Lite. This version does not include the desktop which we don't need as installation is via an embedded web page.



WSPR on Raspberry Pi

If you have problems running the above command, as I did, you can re-run it as needed until you get a successful installation.

RUN WSPRRYPI

With this software, you can run WSPR using either the command line or a web interface. The web interface saves your parameters and sends them to the WsprriPi program when it next runs.

The antenna connects to GPIO4 which is physical pin 7, and ground is connected to physical pin 9.

If you connect a pushbutton switch between physical pins 35 and 39, then when you push the button, the Pi will shut down.

If you connect an LED and serial 330 Ohm resistor between physical pins 12 and 14, then the LED will light when the Pi transmits. It's nice to know when it is transmitting.

The pushbutton switch and Led are included on the TAPR board.

To use the web interface, navigate to <http://{IP Address or hostname.local}/wspr/>


In my case, I navigated to <http://pi3wspr.local/wspr/>



I also could have navigated to <http://192.168.8.39/wspr/>

You will see a window like:



WSPR on Raspberry Pi

 Wsprry Pi

 GitHub  Documentation

For server: PI3WSPR 

Wsprry Pi Configuration

Control
Enable Transmission: ☒ Enable LED: ☒

Operator Information
Call Sign: 
Valid. Grid Square: 
Valid.

Station Information
Transmit Power: 
Valid. Frequency: 
Ok. Random Offset: ☐

Advanced Information
Self Calibration: ☒ PPM Offset:

Transmit Power
This sets power on the Pi GPIO only; any amplification must be taken into account.


16mA
10.6dBm

Enter the fields for call sign, grid square, transmit power, and frequency.

Transmit power is power in dBm. Typically, for a Pi only, this is 10. For the TAPR hat, the power is 20.

Frequency can be in MHz or a band, like 20m. Notice the 0 following 20m in this example. This specifies to transmit on 20m, and then on 0m, or no transmission. It is suggested that you not transmit continuously, but perhaps every 10 minutes. For this, you would enter "20m 0 0 0 0"

To save your entries, click Save. The entries will be effective when the current transmission window ends, as the WSPR cycle is just short of 2 minutes long.

You can shutdown your Pi by using the  in the upper right of the window.

For more information, see

https://wsprry-pi.readthedocs.io/en/latest/Web_UI_Operations/index.html

TEST WSPRRYPI

We can test by using the web interface to run the program. If we monitor the transmission with a TinySA Spectrum Analyzer, we can see what harmonics are present and at what relative



WSPR on Raspberry Pi

strength with this basic implementation that generates the radio transmission as a square wave.

I will need a way to connect the transmitter to an antenna. For this first test, I will use a small vertical antenna. I used the BNC connector that I wired up for our Arduino tests and adapter to connect to my antenna.

TinySA provides a program that will display and control the TinySA from a computer, called “tinySA-App.exe”

ADD LOW PASS FILTER

I built a low pass filter for 20M using a QRP Labs kit. If I add this to my circuit, I can test again with the low pass filter and see how the harmonics change with the low pass filter.

I can also measure the low pass filter using a NanoVNA to see the slope of the filter response.

TAPR WSPR

TAPR.org provides complete kits and assembled solutions for WSPR transmitters using a Raspberry Pi. TAPR offers two versions of a WSPR “hat” to use with Raspberry Pi single-board computers:

- 30m and 40m fully assembled and tested
- 160m, 80m, 20m, 15m/17m, 10m/12m partial kit – user has to solder low pass filter components (supplied)

I have TAPR hats for 40m, 30m, and 20m. At the time I purchased them, all were available fully assembled and tested, while the 20m version is now available as a kit. TAPR is migrating to kit versions for all frequencies.

The TAPR hat includes a handy switch to shut down the Pi before powering off. As stated above, you can instead wire a switch to the pins shown. The hat also includes a built-in LED.

Installation is easier than with the do-it-yourself. Just plug the hat onto the Pi GPIO pins and use the same program as above.

We can test the effectiveness of the TAPR low pass filter using the TinySA.

Alternately, download the image from TAPR and install on a microSD card, and boot in your Pi. Follow the instructions at

https://files.tapr.org/product_docs/WSPR/TAPR-VersatileWSPR%20-%20v1.3.pdf

to configure and run the WSPR transmitter.



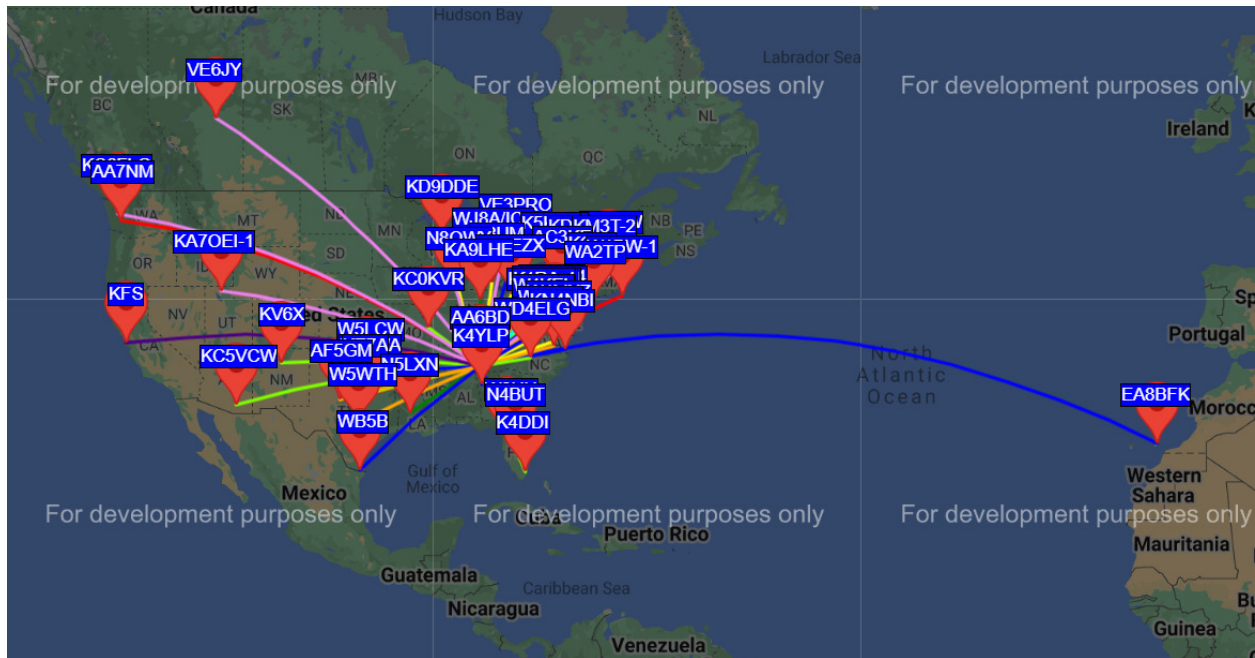
WSPR on Raspberry Pi

RESULT OF THE WSPR TEST ON MY 40M OCF DIPOLE

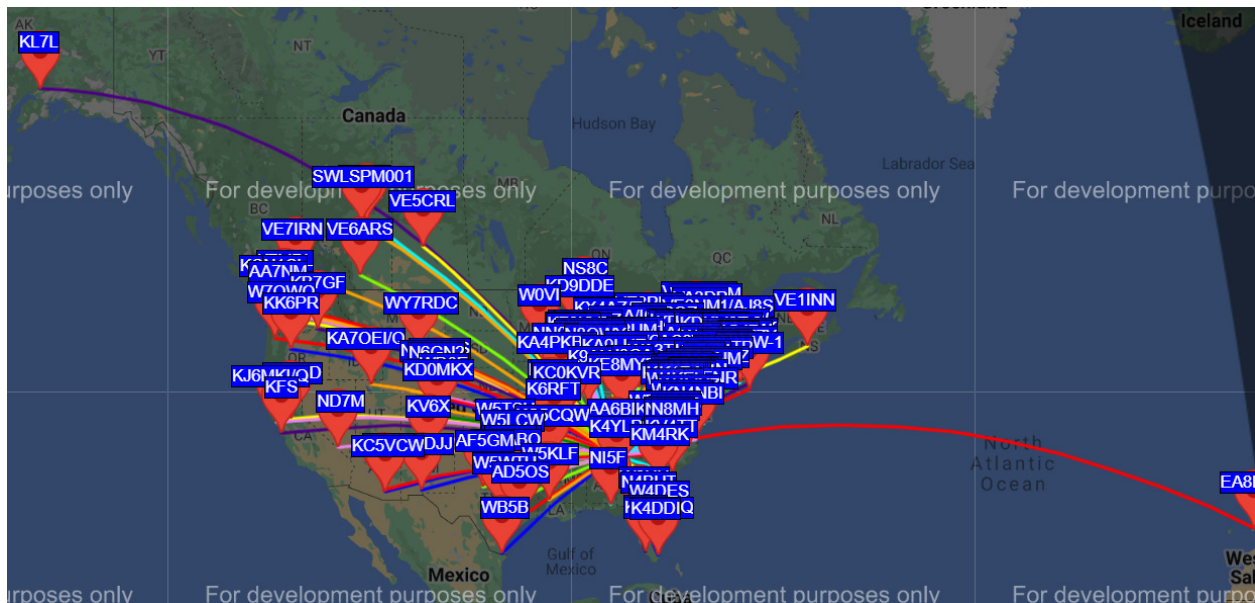
I ran the WSPR transmitter on 20M and viewed the receptions by WSPR receivers. A caveat is that you will only see receptions where there are receivers, so areas with no spots could be because your antenna didn't reach there or because there were no receivers in the area.

I ran for daytime and then nighttime to see what difference there was on 20M.

After running for just three cycles, I saw many spots.



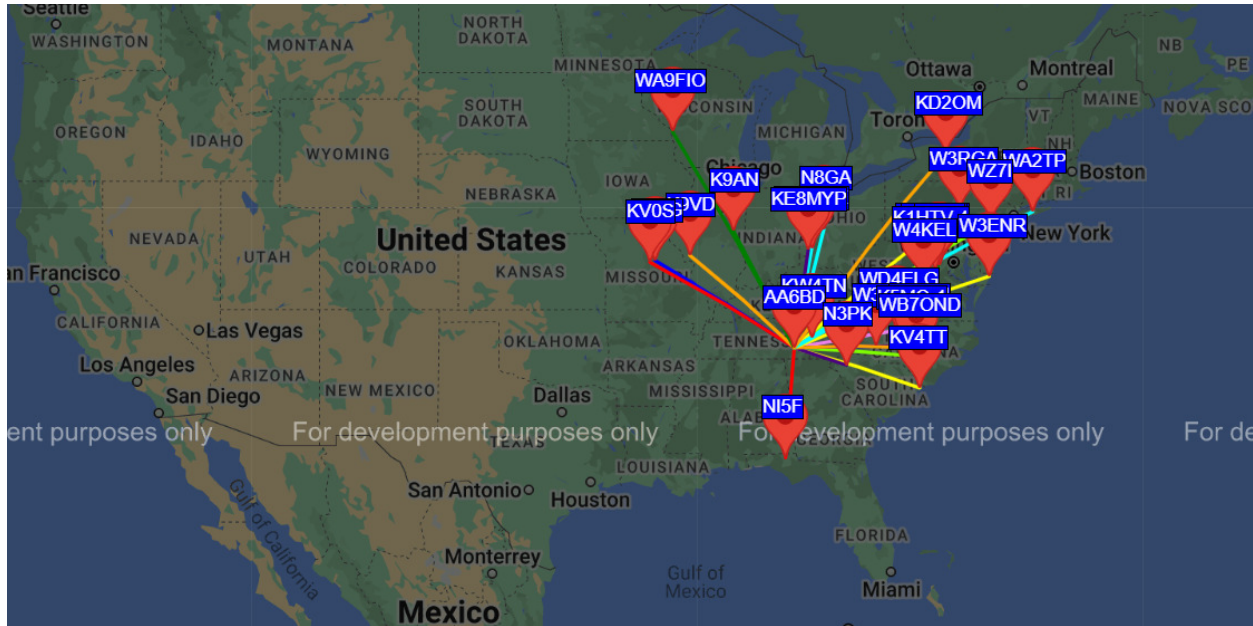
After 4 hours, the spots were





WSPR on Raspberry Pi

I then switched to 40M in the afternoon. 40M propagation is much shorter distance this afternoon.



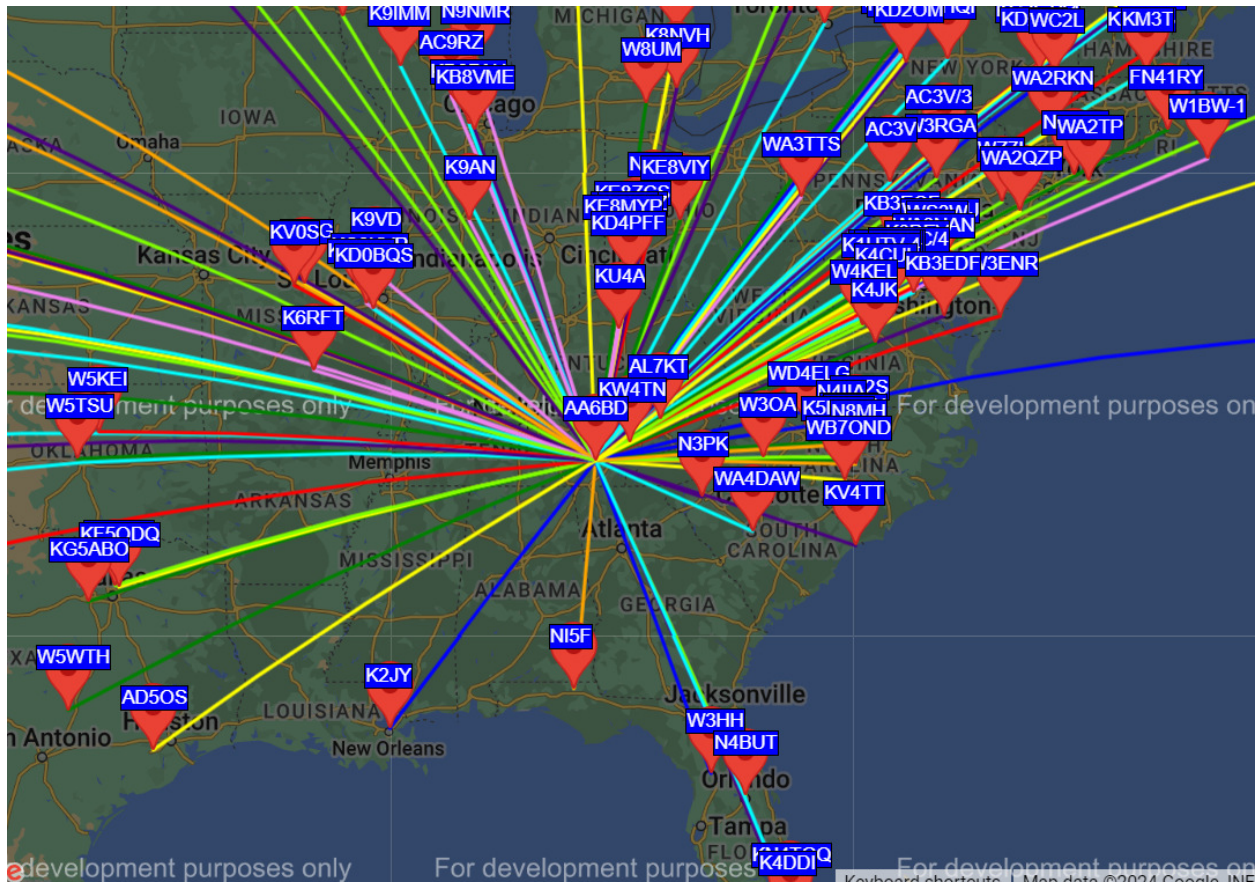
After dark, 40M became a lot more active:



Including more spots near home which were not there with 20M:



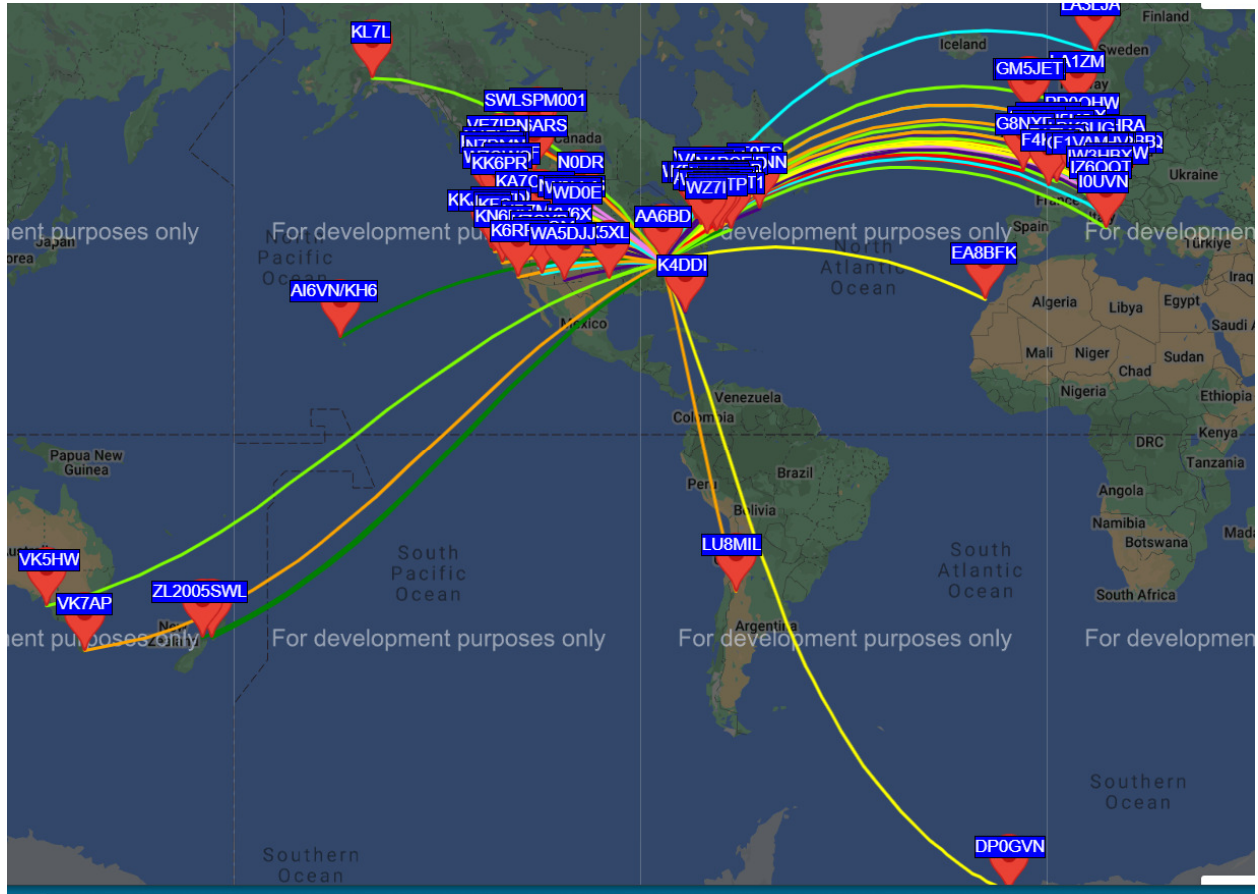
WSPR on Raspberry Pi



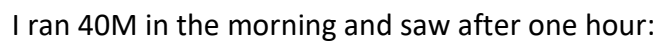
I then ran 20M overnight and wow, every continent, including Australia, reported my station:



WSPR on Raspberry Pi



But notice that in the US, there were no close spots:





WSPR on Raspberry Pi

WHAT'S NEXT

I will consider, as should you, running a WSPR Receiver to see what I/you can receive at the same location and antenna.

We will continue examining the use of the NanoVNA for amateur radio purposes including antenna analysis. We will examine the difference between antenna resonance and low SWR.

APPENDIX

The original software is found at <https://github.com/JamesP6000/WsprryPi>

- Download and compile code:

```
sudo apt-get install git
```

```
git clone https://github.com/JamesP6000/WsprryPi.git
```

```
cd WsprryPi
```

```
make
```

- Install to /usr/local/bin:

```
sudo make install
```

The command to run on 20M is

```
$ sudo wspr -r AA6BD EM75 33 20m
```

where 33 is my TxPower in dBm.

-r is to repeat