WSPR Without Tears

Introduction

WSPR Without Tears is my attempt to put together a WSPR transmitter kit that avoids (hopefully) most of the problems associated with building a fairly complicated digital mode kit. Check out my website (http://WsprWithoutTears.com) for more information.

WSPR stands for Weak Signal Propagation Reporter Network. It's a digital mode used by hams. The process is similar to beacons. WSPR is a wonderful communication mode created by Joe Taylor (K1JT). WSPR Without Tears is my attempt at simplifying the process of getting an actual transmitter up and running so you can enjoy working with WSPR and dealing with the headaches associated with building a working system.

What makes it really interesting is that WSPR is optimized for operating at very low power. For example, it's normal to transmit with 200 mW from the US and be received in Australia. There are hams all over the world with WSPR receivers. They record what they receive in a central database that can be queried over the Internet. So you put your WSPR beacon on the air and monitor its performance from the Internet. Monitoring WSPR is sort of addictive. It's fascinating to see what effect time of day, seasons, and weather have on propagation.

There is a wide array of choices for setting up a WSPR beacon. The most obvious and possibly the simplest is to use your regular transceiver and antenna. This requires a computer and sound card connection. Most of us have at least one computer in the shack and many of us are already using digital modes, so the equipment is already in place to run WSPR. All you have to do is download the software (free) and you're off to the races. This approach works well and the price is right if you're already are set up for digital modes. Even if you're not set up for digital modes, the interface between your transceiver and computer can be had for less than $100.

Using your base station for WSPR is an excellent approach. The downside is that you are tying up your station and antenna while WSPR is on, so leaving WSPR on for an extended length of time gets in the way of radio. The output power of many transceivers can't be adjusted below 5 watts. An output power of 5 watts will certainly work, but it's interesting to see how little power you can get away with and that's a lot less than 5 watts.

The alternative to using your regular station for WSPR is to have a completely separate WSPR station. That's a lot more realistic when you're dealing with output powers in the range of milliwatts. When I started down this path I looked at a different kit. The kit was high quality and well done. However, there are some operational complaints I had with it that I wanted to fix.

Table of Contents

First, everything was set using menus which are set using push buttons. That's a lot of button pushes to program in modes, callsign, date, time, ... Also, WSPR is picky about setting the time and transmit frequency. The time has to be accurate within a couple of seconds and the frequency definitely needs to be accurate. The entire WSPR window is only 200 Hz wide. It's easy to see how being off frequency (being off ±100 Hz in 10 MHz is not a lot) could put you completely outside of the window. Using the kit involved split second button pressing to set the time (which had to be reset whenever the power was turned off) and frequency calibration. Both could be accomplished using a GPS module, but this added cost and complexity to the project.

The second issue is a nit pic. A WSPR cycle is just under two minutes long. It's considered good form to not transmit continuously, so a typical ensemble is one cycle on and four cycles off. There was no easy way to know when I was transmitting.

Approach

I found an alternate approach using a Raspberry Pi (rPi) computer to generate WSPR transmissions. rPi is about the size of a deck of playing cards and costs $35. It has an amazing amount of computing power for something so small and so cheap. It runs the Linux operating system.

The rPi can be configured to generate the WSPR signal without any external components! That's just amazing to me. Somebody figured out how to modulate the clock/oscillator and generate a WSPR signal. The rPi will also set the time by calling an NTP server and use the time information to adjust the frequency, eliminating the need for any calibration or timing setup.

There are two problems with using the rPi for WSPR. First, the output signal is a square wave with harmonics from here to next Tuesday. A low pass filter is called for in order to keep other hams and the FCC happy. Second, the output power is around 10 mW. I know I talked about the thrill of using low power to make contacts, but it might be nice to start with some more power.

My WSPR rPi is a circuit board (pcb) that plugs into the rPi 40-pin header. It boosts the signal level to about 200 mW and contains a seven pole low pass filter to keep out-of-band transmissions down by 45 dB. It also contains an LED the lights when transmitting and a crude output power indicator. (I found it annoying to not know how well my system was working without connecting a 'scope or power meter and wanted something simple as a sanity check.)

The rPi is a computer, and most computers need to be shutdown properly rather than just have the power turned off. I added a push button to the circuit board to initiate a shutdown sequence in the rPi. I wrote a program to run in the background that monitors the pushbutton and shuts the rPi down when it's pressed.

Raspberry Pi

The rPi is a computer and is subject to all of the headaches associated with computers. I wanted to make life easier for anybody trying to duplicate my WSPR experience by limiting the amount of computer exposure you're subjected to. Loading the software into the rPi is not difficult, but there are quite a few areas that can go

wrong and it could be frustrating if you've never done it before. It only takes one problem to kill the whole process. There are a lot of steps involved in getting to a working WSPR beacon.

I created a version of the software that runs automatically when the rPi is turned on. There are several parameters that you (the user) have to set, such as your callsign and grid square. I programmed a web server in the rPi (I told you it's a computer :-) that lets you fill this stuff in from a web page. In other words, you bring up Internet Explorer or Chrome or Firefox on your PC and type in the rPi's address (I'll get to that) and the web page comes up and you enter the data. This means no programming of the rPi at all. Bring up the web page, type in your callsign, grid square, and WiFi parameters and you're up and running.

The operating system (Linux) and all of the software are stored on a micro SD card. The SD card is plugged into the rPi. Set it and forget it.

Antenna

I'm using a half wave end fed dipole antenna. Any antenna will work. End fed dipoles are nice in that there is no heavy feed connection pulling them down in the middle, so they are a little easier to string up. Their downside is that voltages are very high at the antenna feedpoint. Fortunately, at the power levels we're talking about the voltage is only about 40 volts. (The voltages would be in the range of 500-600 volts at 100 watts.)

Performance

I draped my antenna around my shack. The highest point is the hinge on the closet door frame. The rest of the wire is laying on the floor. I've managed to snag Europe a couple of times with 200 mW. I regularly hit all of the USA and parts of Canada and Mexico from my south Alabama QTH and I'm even in a hole.

One of my friends connected another prototype to a 30m dipole antenna at 40 ft and reached Australia and New Zealand from South Alabama with 200 mW. Of course, you mileage may vary.

Quickstart – Amplifier/Filter Board

Got to the website (WsprWithoutTears.com) for more information.

Parts List

Check the parts received against the parts list below. Unless you're reasonably comfortable reading color bands (chart below) on resistors, you might want to verify the resistances using a meter (digital volt meter – DVM).

A note about the color bands: The last band is a tolerance indicator (gold, silver, nothing). The tolerance (how close the actual resistor value is to what it's supposed to be) isn't a factor in the WSPR board, so if the tolerance band is not the same as listed, don't worry about it.

It's also easy to get the resistor backwards and read the values in reverse. Needless to say, you'll end up with the wrong resistance value.

What you'll need to complete the assembly

1. *Raspberry Pi 2.0 Model B or Raspberry Pi 3.0*
2. *2 amp (minimum) power supply w/micro USB connector.*
3. *Ethernet cable*.
4. *WiFi USB dongle*. You only need this if you're using an rPi 2.0 and you want to use WiFi. WiFi is built into the rPi 3.0.
5. *Soldering Iron and solder.* A small tip and adjustable temperature are good.
6. *DVM*.

I'm assuming that you know how to solder. If not, there are plenty of tutorials on the web. Probably the biggest thing to keep in mind is that dirt and oil do not solder very well, so it's important to clean the stuff you want to solder before you solder it. Isopropyl alcohol works well for this. ***Wear safety glasses when soldering***!

I can tell you to read all of the directions first, but if you're like me you'll just smirk and start building the kit anyway. That's okay. If you run into trouble read a little further to see if I've addressed the issue. The only thing you ***really*** want to look out for is putting the 40-pin connector on the correct ***side*** *of the circuit board* – it goes on the ***other*** side from the components. **Don't mess this one up!**

Part	Value	Description	
C1	100nF	104	
C2	100nF	104	
C3	47uF		
C4	270pF	271	
C5	560pF	561	
C6	560pF	561	
C7	270pF	271	
C8	100nF	104	
D1	1N4148	diode	
J1	40-pin	connector	
J2	BNC	RF connector	
L1	FT37-43	toroid	
L2	1uH	brn-blk-gold-sil	
L3	1.2uH*	brn-blk-gold-sil	(larger dumbell shape; **incorrectly marked**)
L4	1uH	brn-blk-gold-sil	
LED1	LED		
Q2	BS170	MOSFET	
R1	20K	pot	
R2	47	yel-pur-blk-gld	
R3	20K	red-blk-ora-gld	
R4	1	brn-blk-gold-gld	
R5	15K	brn-grn-ora-gld	
R6	270	red-pur-brn-gld	
R7	2K	red-blk-red-gld	
S1	Switch		
wire		4 in piece of hookup wire for winding L1	
pcb		printed circuit board	
SD Card		programmed with Raspberry Pi Linux/WSPR image	

* Component L3 is mismarked given basic color codes.
You can tell the difference between L2 and L4 and L3, because L3 has a more distinct dumbell appearance (see the assembly figures).

Figure 1: Amplifier/Filter Board (AFB) Bill of Materials

Let's Do It!

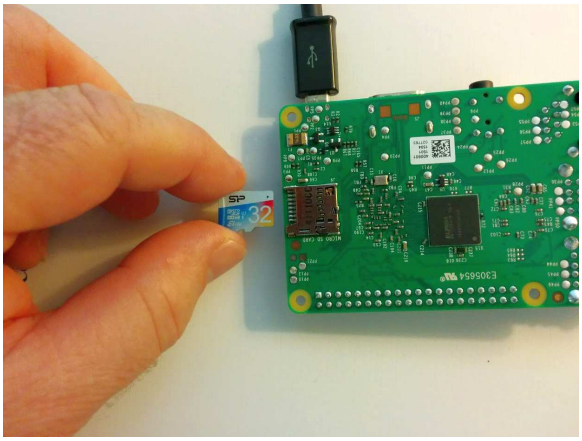

Figure 2: Inserting SD Card into rPi

Figure 3: SD Card Inserted into rPi

First, insert the SD card into the rPi (Figure 2: Inserting SD Card into rPi and Figure 3: SD Card Inserted into rPi). The SD card sorta slides in up side down. On rPi 2.0 it ratchets in (pops in then out a little). On rPi 3.0 there's a friction fit.

Figure 4: Resistor Color Code is the magic decoder ring for figuring out component values.

Color	Digit	Multiplier	Tolerance (%)
Black	0	10^0 (1)	
Brown	1	10^1	1
Red	2	10^2	2
Orange	3	10^3	
Yellow	4	10^4	
Green	5	10^5	0.5
Blue	6	10^6	0.25
Violet	7	10^7	0.1
Grey	8	10^8	
White	9	10^9	
Gold		10^{-1}	5
Silver		10^{-2}	10
(none)			20

Figure 4: Resistor Color Code

Here's how it works: *red-blk-ora-gld* is read as

red = 2, black = 0, orange = 3, gold = 5%, which translates into -

2 (red) 0 (black) 000 (orange) = 20,000 ohms = 20K ohms with a 5% tolerance

(third band is how many zeros to add on to the end), or 20K ohms.

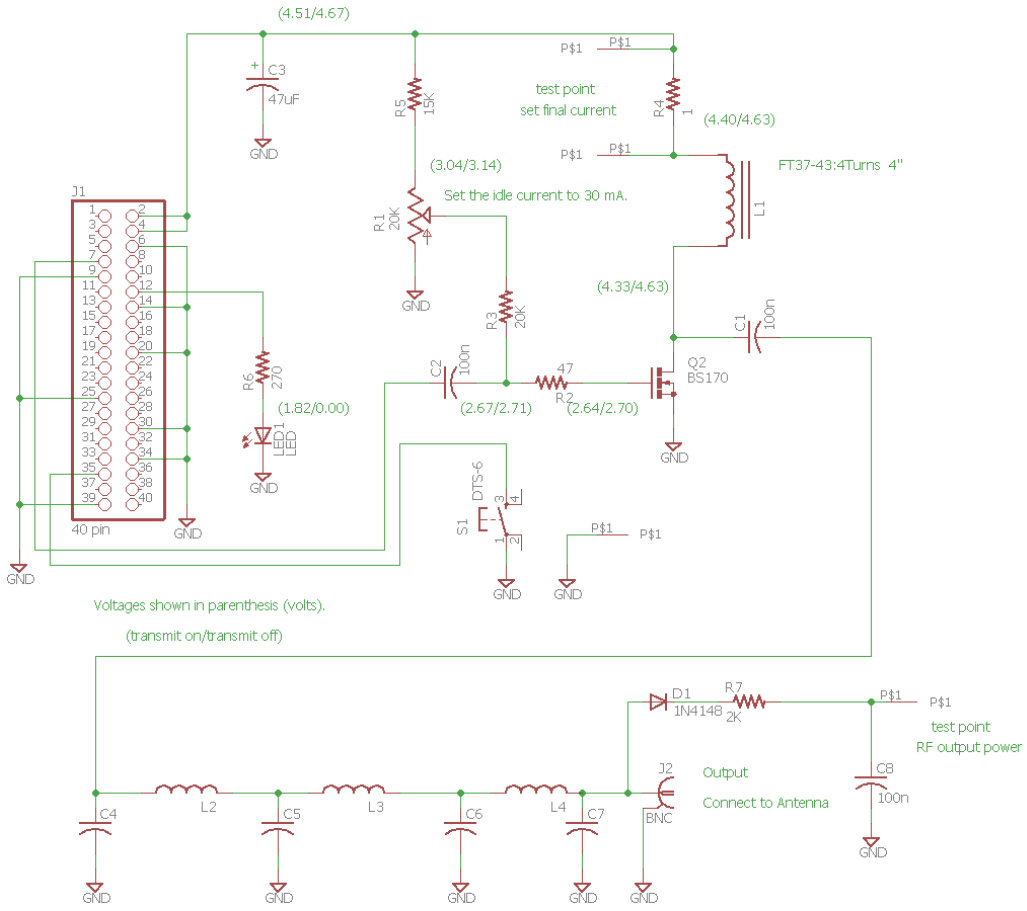

Figure 5: Amplifier/Filter Board Schematic

I figure the first order of business is to show you the schematic and some pictures of the journey so you can see what everything is supposed to look like along the way. First thing on the list is the bare Amplifier/Filter Board (AFB) circuit board; the foundation on which to build the kit.

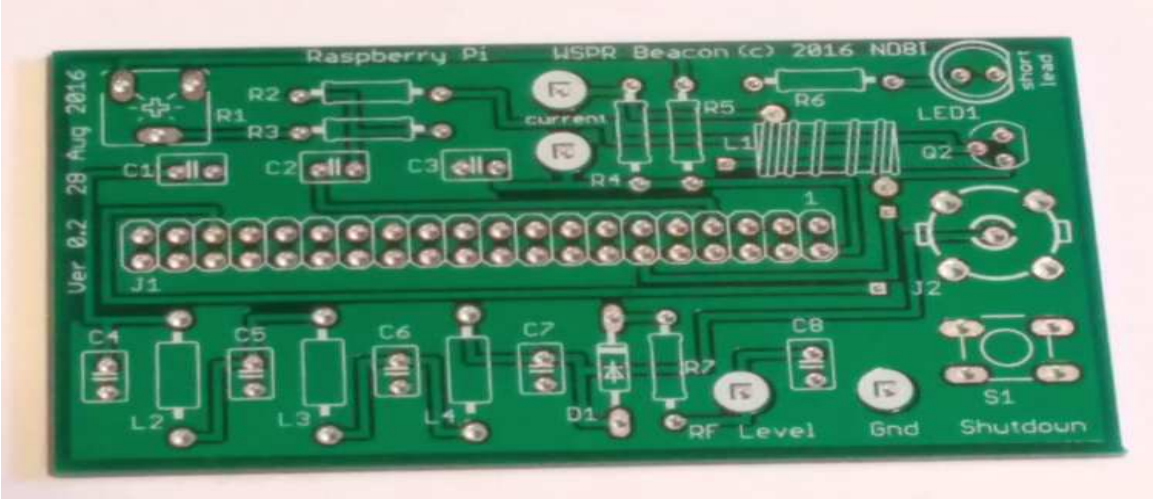

Figure 6: Unpopulated Circuit Board (Top View)

Here is what the AFB should look like when it's populated –

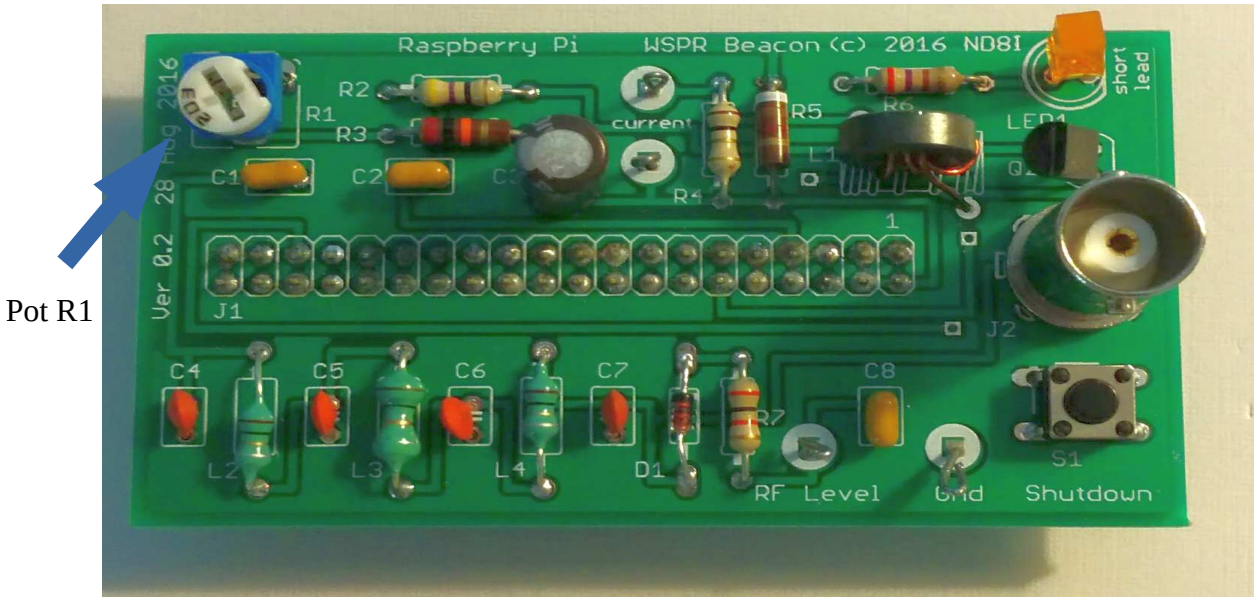

Figure 7: Populated Circuit Board

Here's a side view of the AFB when it's populated –

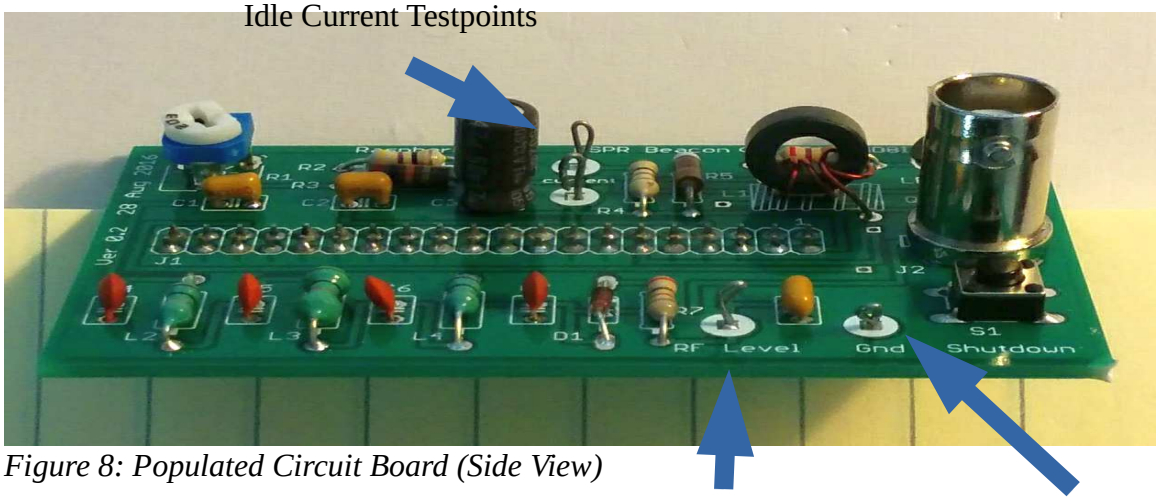

Figure 8: Populated Circuit Board (Side View)

And finally the underside. Note the connector on the ***underside*** of the board. I emphasize this multiple times because this is the one mistake you don't want to make – putting the connector on the wrong side of the board. It's pretty much impossible to unsolder all 40 pins and put the connector on the correct side of the board.

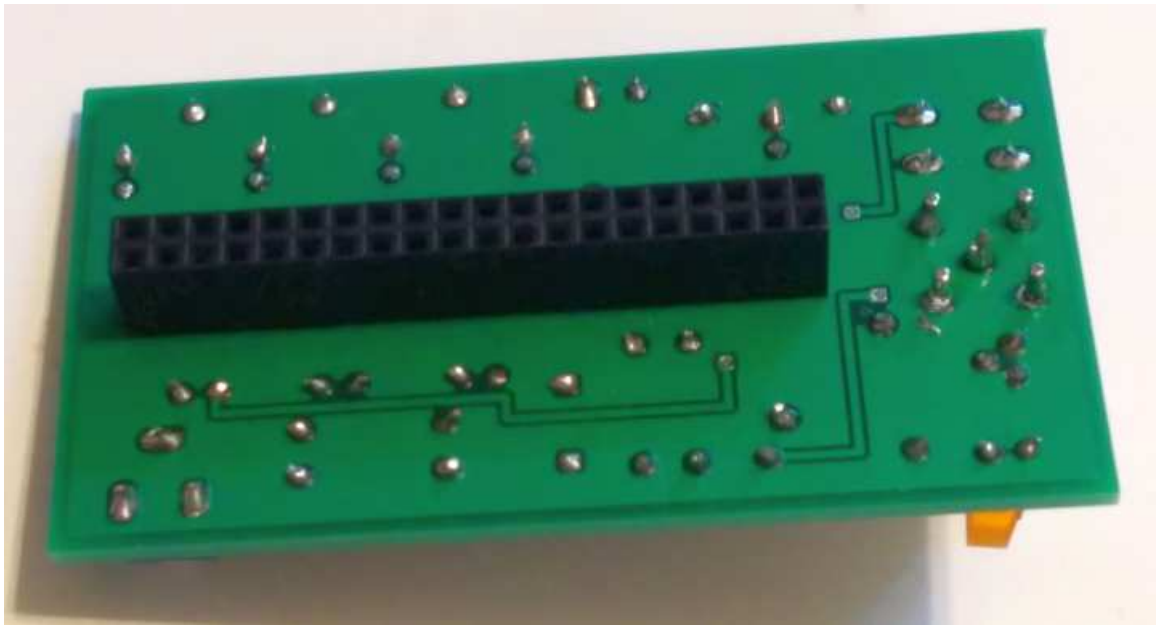

Figure 9: Circuit Board (Bottom View)

The finished assembly is shown in Figure 10: WSPR Beacon. This is the AFB plugged into an rPi 2.0.

The micro USB power connector is coming out of the lower left. RF output comes through a BNC connector shown in the upper left. (A short jumper cable is shown rather than the BNC-BNC adapter in this picture.) A WiFi dongle is plugged into one of the USB ports. The rPi LEDs are on the left hand side of the rPi board below the AFB board. They are not visible in the picture.

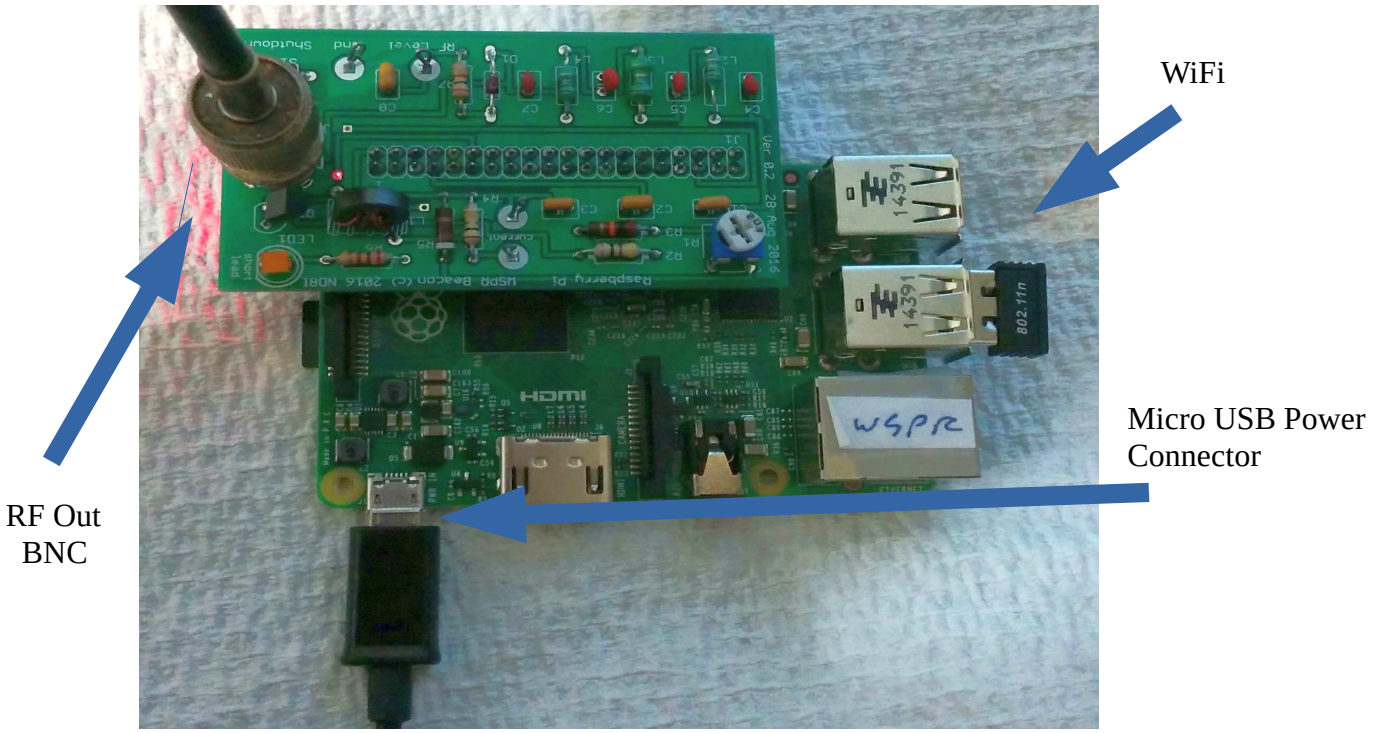

Figure 10: WSPR Beacon

The side view of the completed assembly (Figure 11: WSPR Beacon (Side View)) shows how the AFB is plugged into the rPi. It's definitely possible to connect them incorrectly. Notice from this view that the rPi USB connectors are on the right hand side and the AFB BNC connector is on the left. That's how it should be. Also, make sure that the rPi pins and the AFB plug are lined up. It is possible to offset one from the other.

Assemble the AFB

1. Install all resistors (except for R1) one at a time. Save four of the clipped leads to use later for P1, P3, P5 and P6 (test points). Measure resistor values with a DVM if you're even slightly unsure.

2. Install all capacitors and inductors (except for the toroid, L1) one at a time.

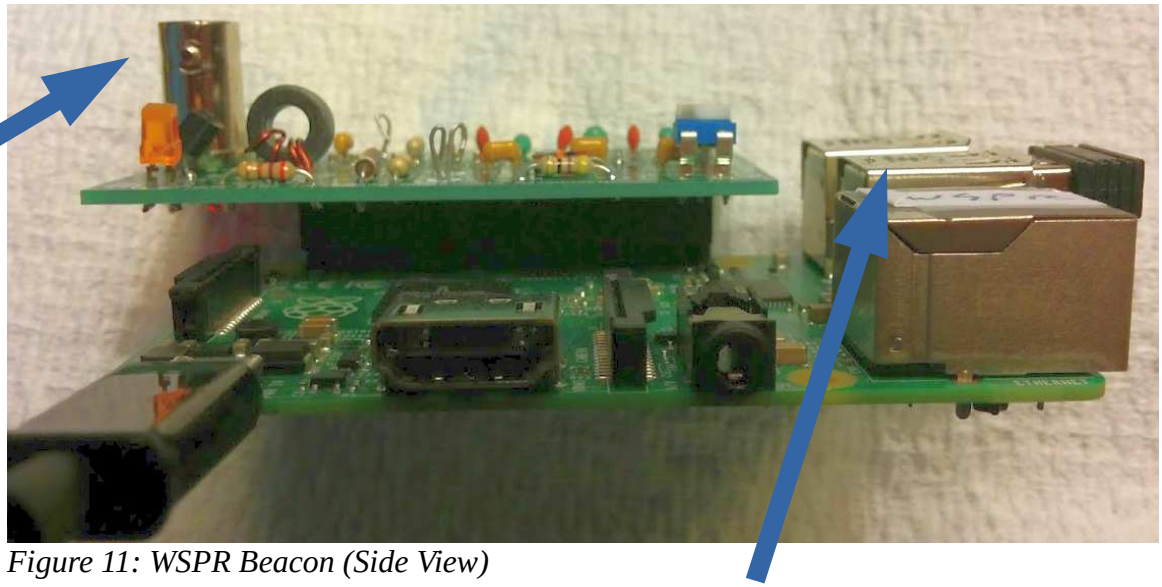

Figure 11: WSPR Beacon (Side View)

3. Install D1, LED1, C3, and Q2. The orientation is important for all four parts.
    - LED1 has one lead that is shorter than the other. The short lead should be placed in the hole towards the endge of the circuit board (pcb) labeled *short lead*.
    - D1 has a black band around one end. The end with the band should be oriented with the silk screen white band on the pcb.
    - C3 positive (longer lead) should face towards the BNC connector. The negative lead (marked -) should face towards C1 and C2.
    - The flat side of Q2 should be aligned with the flat side of the silk screen on the pcb. The middle lead of Q2 will have to be bent outward to match the hole.

4. Wind four turns of hookup wire around the toroid. Each time the wire passes through the middle of the toroid counts as one turn. You have to wind the wire in the correct direction so that it lines up with the holes in the pcb. Install, solder and clip leads for the toroid (L1).

5. Install S1 and R1. Push these parts all the way through the holes. They should sorta snap in once you've got them installed (they should be seated). Solder them and don't bother clipping the leads.

6. Install J1 (40-pin header). This piece goes on the **BOTTOM** of the pcb. I want to emphasize this because it's very painful to undo if you get it wrong. All of the rest of the parts are installed on the top side of the pcb. **J1 is installed on the bottom** so that the pcb can attach to the matching pins on the Raspberry Pi (rPi).

    The best way to install a header like this is to have it in position and solder **one** corner pin. Then pick the pcb up with one hand and put slight pressure on the 40-pin connector, pressing it into

the pcb. Use your other hand to hold the soldering iron to remelt solder on the pin just soldered. This takes all of the play/slop out of the connection.

Repeat the operation with the pin on the opposite corner. The connector is now firmly attached to the pcb. Solder all of the remaining pins to complete connector installation.

7. Install J2 (BNC). Place the BNC connector in the pcb (**top** side) and solder the middle pin. Now push on the BNC connector with one hand and melt the middle pin solder with your soldering iron just like you did with the 40-pin header in step 6. The BNC connector is now seated. Go ahead and solder the remaining four pins. **CAUTION:** The pins are connected to the barrel of the BNC connector. Soldering the pins will heat up the BNC connector barrel. Don't touch the barrel!

8. Solder four short pieces of wire (what's left from the resistor leads works just fine here) into the holes marked *RF Level, Gnd* and C*urrent*. There are two holes by *Current*. Clip the wires to about 3/4” and make a small loop in the ends (see Figure 7: Populated Circuit Board and Figure 8: Populated Circuit Board (Side View)).

9. Check your work. Actually, take a break at this point. Go get a cup of coffee to clear your head. Now come back and walk through the parts list and verify that every part on the AFB is correct.

10. Hold the assembled pcb (AFB) next to the connector on the rPi to verify that everything looks about right. Plug the AFB into the rPi noting that the BNC connector should adjacent to the SD card. Make sure that the header pins and the connector on the AFB are actually lined up – I haven't done it, but I suspect that you (well, not you, but somebody else) could misalign the two and still assemble the parts.

11. If you have a heat sink on your rPi CPU, remove the AFB from the rPi and place a piece of electrical tape on the underside of the AFB so that the heat sink doesn't touch anything on the underside of the AFB and short something out.

Setup and Adjustment

What You Need

I'm assuming that you have on hand -

1. Assembled Amplifier/Filter Board (AFB)
2. Raspberry Pi version 2.0 or later
3. 2+ amp power supply for the rPi
4. DVM
5. Assembled antenna kit
6. Small flat blade screw driver
7. Ethernet cable to connect rPi to your router

8. SSID and passkey for your router (go to the Troubleshooting/Getting SSID and Passkey section if you don't have these)
9. PC connected to your network
10. Optional: USB WiFi adapter for rPi 2.0 (rPi 3.0 has WiFi built in)

Initial Setup

The AFB should be connected to the rPi and power is ***off*** at this point.

1. Plug the SD card into the rPi and connect the power supply. The SD card sorta slides in up side down. On rPi 2.0 it ratchets in (pops in then out a little). On rPi 3.0 there's a friction fit.
2. Connect the rPi to your router with the Ethernet cable. This part is a little tricky. The rPi has to be connected to a DHCP server. That's normally your router. *It won't work if you plug the rPi into your laptop.*
3. Connect the antenna to the AFB BNC connector using the BNC-BNC adapter.
4. Turn the pot (R1) on the AFB ***counter clockwise*** until it stops turning with the small screw driver.

Smoke Test & Set Idle Current

1. Turn on the power for the rPi (connect the power supply lead to the rPi, then connect the power supply to the wall outlet). The red and green LEDs will flash for a while. No flashing at all indicates a problem. Wait until they're done flashing (20-30 sec) then go to the next step.
2. Make sure that nothing is hot or smoking. Smoke coming off anything is always a bad sign …
3. Using the DVM, measure the power supply voltage between the lower test point by *Current* and *Gnd*. This should be between 4.5 – 5.0 volts (see Figure 8: Populated Circuit Board (Side View).
4. Using the DVM, measure the voltage across the terminals labeled *current* on the AFB (near the toroid). Turn pot *R1* on the AFB *clockwise* until the voltage across the terminals is roughly 0.030 volts. It's not really critical, just get it in the general range (say 0.025-0.035 volts) (see Figure 7: Populated Circuit Board and Figure 8: Populated Circuit Board (Side View).

   Note: This is the idle current adjustment for the MOSFET. You're measuring voltage across a 1 ohm resistor, so you're setting the idle current to about 30 mA. Make sure that the LED is *off* when you're making this measurement. If it's on, wait until it goes off to make the measurement (less than 2 mins).

   Bonus: You do know that everything electronic works from magic smoke? The way you know this is true is because if you ever let the magic smoke out, the electronics never work again.

Locate the rPi IP Address

1. Working from your PC, download *WSPR Executables* (zip file) from the web site (WsprWithoutTears.com), unzip the zip file, and extract the program WSPR_Locate.exe to a suitable work area.
2. Execute WSPR_Locate.exe. The program will take about five minutes to execute and will (hopefully) give you the IP address of the rPi.

   Some assumptions have been made: 1) your network is a private Class C network (IP addresses begin with 192.168..), and 2) you have DHCP enabled on your router (router hands out IP addresses). If you have no idea what I'm talking about then don't worry about it – these are typical settings for a home network.

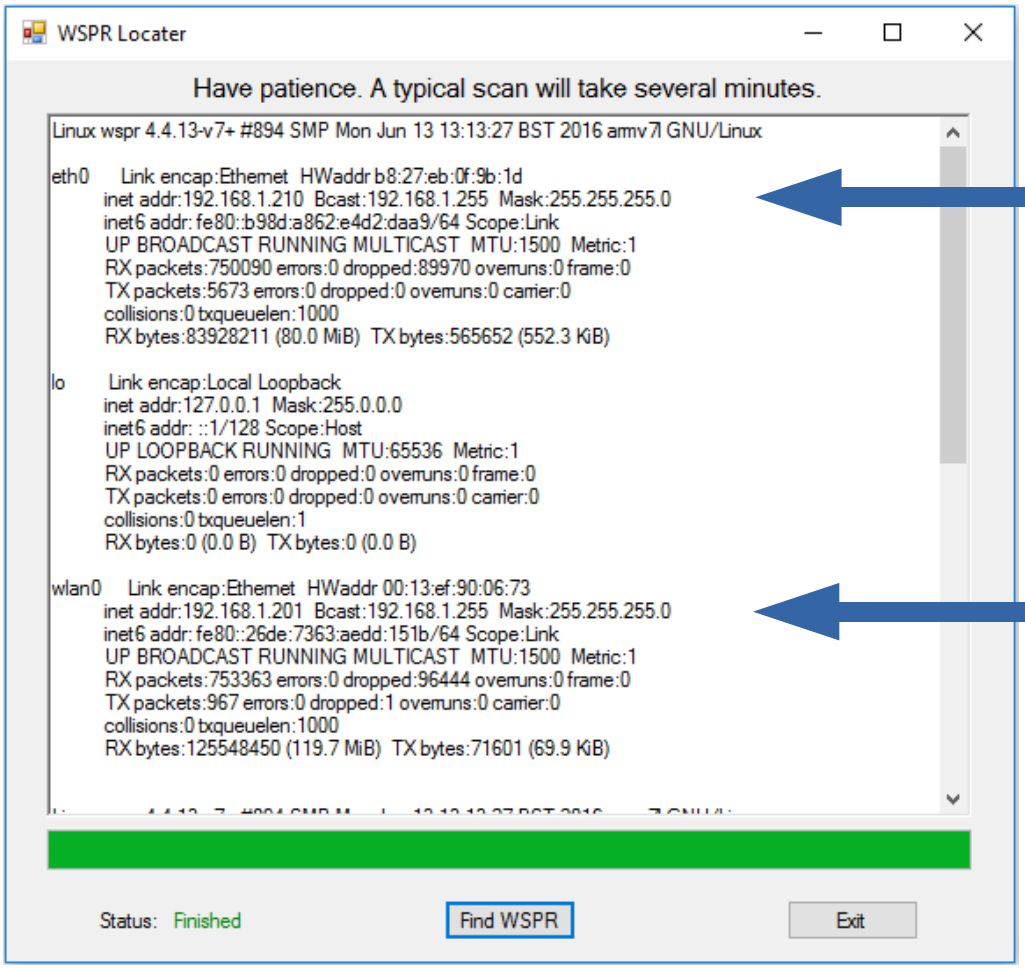

Figure 12: WSPR Locater

With any luck, the display will look something like Figure 12: WSPR Locater. Look for the line starting with *eth0*. The line below it shows *inet addr: 192.168.1.210*. Your setup should be similar

Setting WSPR Configuration

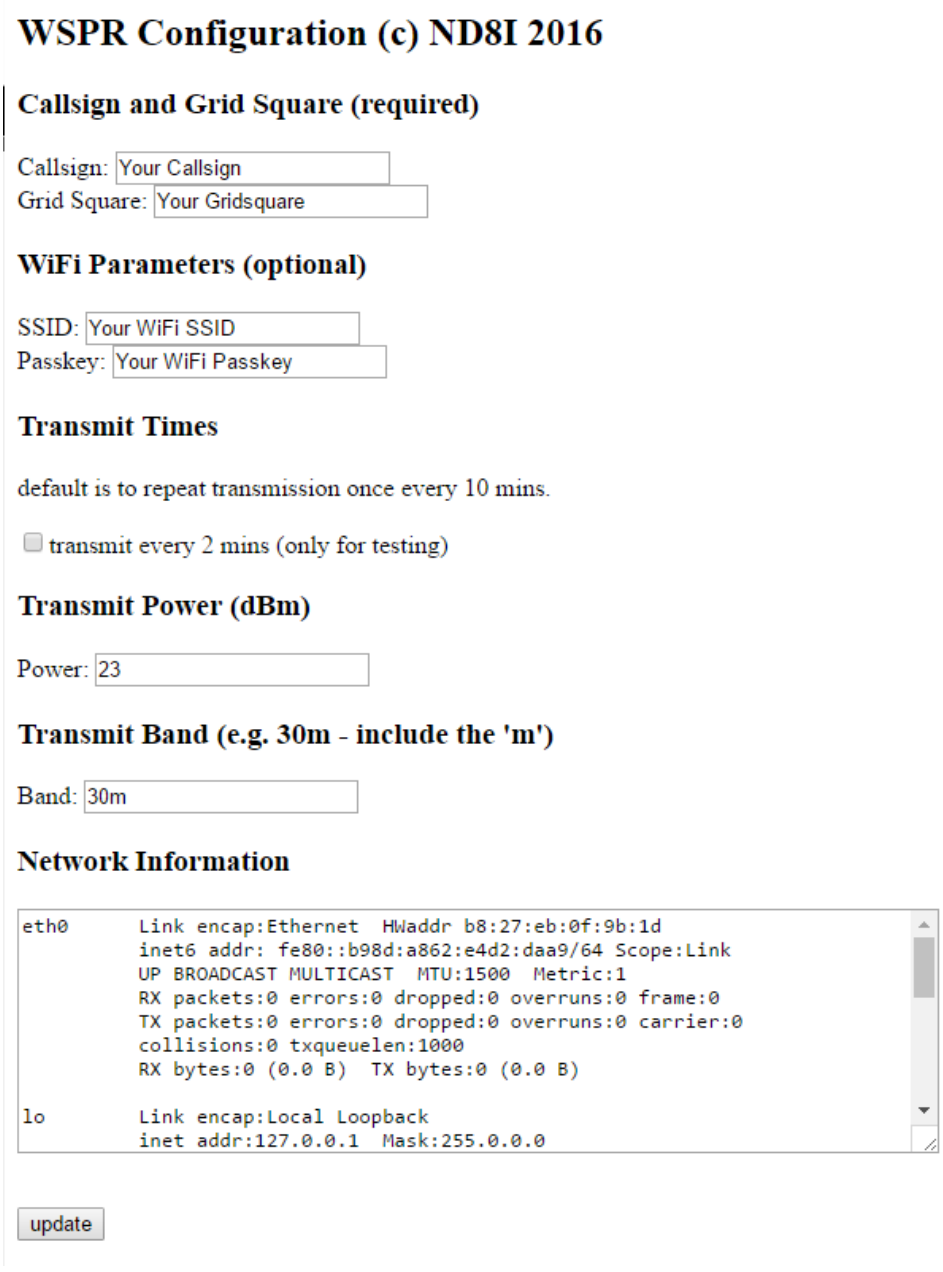

Figure 13: WSPR Configuration Screen

except for the last two numbers (it will begin with 192.168, but the last two numbers will probably be different). Record the full number (*192.168.aaa.bbb* – aaa and bbb are the numbers for your rPi) for the next step. That's the IP address of your rPi.If the program fails to find the rPi, try it again. If that fails, go to the Troubleshooting section in the full documentation.

Note: The first time you run WSPR_Locater *inet addr* will be displayed for *eth0*, the Ethernet connection. After setting the WiFi portion of the WSPR configuration (see Figure 13: WSPR Configuration Screen), WSPR_Locater will display *inet addr* for *eth0* if the Ethernet cable is still connected and *inet addr* for *wlan0*, the WiFi connection.

1. Open a web browser (Internet Explorer, Firefox, Chrome, …) and type in the IP address you just recorded (192.168.0.67 for example). This will bring up the configuration screen shown in Figure 13: WSPR Configuration Screen -
2. Enter your *callsign* and *gridsquare parameters*.
3. Enter the *SSID* and *passkey* (password) for your WiFi network. These are what you use to enable your cell phone and things like Roku to use your home network. If you don't know what they are you'll have to go digging through your router to figure them out (see the *Troubleshooting* section).

   If you elect to not use WiFi then you don't need to enter SSID and Passkey. You will have to keep the rPi connected to the network by Ethernet cable because the rPi uses the Internet for time synchronization.
4. Check/uncheck the *transmit every 2 mins* checkbox. (I recommend leaving it unchecked.)

   A WSPR message takes just under two minutes to send. The fastest you can send WSPR messages is every two minutes. It's considered bad form to flood the WSPR servers with messages. The default is to send a WSPR message every ten minutes. If you check the *transmit every 2 mins* checkbox then messages will be sent every two minutes. This setting is useful for testing and short term experiments.
5. The *Transmit Power* and *Transmit Band* should be left as *23 dBm* and *30m* for the current configuration.
6. When all done click on the *update* button. This will store the data on the rPi.

Restart rPi

1. Press the pushbutton on the AFB. Hold for two seconds and release. This commands the rPi to enter shutdown mode (you're always supposed to shutdown a computer properly or the file system can be damaged – pulling the plug is bad).
2. Unplug the power supply, wait 30 sec, then plug the power supply back in and wait until the LEDs on the rPi stop flashing.

Measure RF Output

1. LED1 on the AFB turns on when the AFB is transmitting. Wait until LED1 is on and measure the voltage across the test points labeled *RF Level* and *Gnd*. This should be in the range of 3-5 volts.

2. Finally, string your antenna up around the room, over doors and lamps to get some of it off of the floor. Later, you can find ways to arrange it so that it radiates better – like you would with any antenna. This is good enough for now.

IP Addresses

Ignore this section if you're not using WiFi.

I have made the assumption that you are using WiFi to connect to the rPi. This is my preference, but it is by no means necessary. What may be confusing is that initially you have to use an Ethernet (wired) connection to talk to the rPi. This is because the WiFi login information has not yet been programmed into the rPi.

After you have set up WiFi (see Figure 13: WSPR Configuration Screen) you will have two valid IP addresses assigned to the rPi. Either one will work. You probably wouldn't have gone to the trouble of setting up WiFi if you were just going to use Ethernet. So now is the time to remove the Ethernet cable. Your rPi will now have one IP address. There is no advantage to having two IP addresses, so you should remove the Ethernet cable if you're planning on using WiFi.

Quickstart – Antenna

Got to the website (WsprWithoutTears.com) for more information.

Parts List

Check the parts received against the parts listed below.

Part	Value
wire	42-0” ft 22 ga wire
coax	5 ft RG-174
toroid	FT37-43
P1	BNC connector
J1	BNC-BNC adapter
xxx	14” 30 Ga magnet wire
xxx	3” 24 Ga magnet wire
xxx	cable tie

Figure 14: Antenna Parts List

Note: The magnet wire gauges are unimportant – it's just what I happened to have around when I made my prototype.

What you'll need to complete the assembly

1. *Antenna kit*
2. Wire Stripper
3. *Soldering Iron and solder.* A small tip and adjustable temperature are good.
4. Small screw driver or pin
5. Matches or cigarette lighter
6. sandpaper

Stripping The Coax

I opted to use RG-174 coax for the antenna because it's, well light and relatively cheap. At least cheaper than the alternatives. That being said, RG-174 can be hard to prepare unless you know the

secret. The secret is to pull the braid apart. Do this a little at a time and work your way along the braid, say 1/4” at a time.

Use regular wire strippers or a knife to remove the outer plastic sheathing for about 1-1/2” as shown in Figure 15: Unpacking Braid. Then push the braid back to spread it apart (also shown in Figure 16: Loosening Braid).

Use a small screwdriver (e.g. jeweler's screwdriver) or pin or any similar object and poke into the braid, just enough to get under it (Figure 16: Loosening Braid).

Figure 15: Unpacking Braid

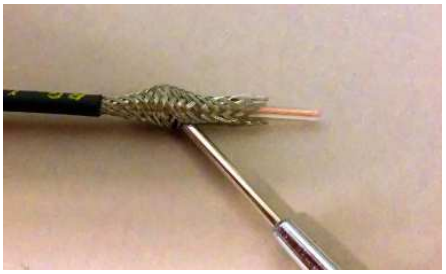

Figure 16: Loosening Braid

Pull a strand of the braid out (Figure 17: Unpacking Braid 2 and Figure 18: Unpacking Braid 3).

Figure 17: Unpacking Braid 2

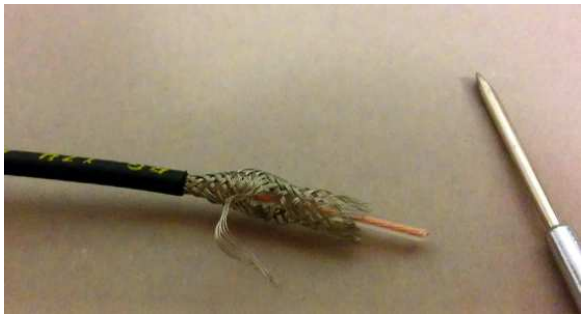

Figure 18: Unpacking Braid 3

Repeat until the braid is unwoven (Figure 19: Preparing Coax 1). Twist the braid and clip the end so that it's square (cut the stragglers off) and strip 3/8” insulation from the center conductor (Figure 20: Preparing Coax 2).

Figure 19: Preparing Coax 1

Figure 20: Preparing Coax 2

Winding The Toroid

Yeah, I know. Winding toroids is tedious. We're starting with two pieces of magnet wire and a toroid (Figure 21: Winding Toroid 1). Take the longer piece of magnet wire and pass it through the inside of the toroid (Figure 22: Winding Toroid 2). Leave about 3/4” sticking out past the toroid (Figure 23: Winding Toroid 3). I'm right-handed, so I hold the toroid with my left thumb and forefinger and feed the wire with my right hand.

Fold the wire over the top of the toroid and fish the end back through the middle of the toroid. Pull the wire snug (you don't have to kill it, just take the slack out). Repeat for 24 turns total. A turn is officially when the wire passes through the middle of the toroid.

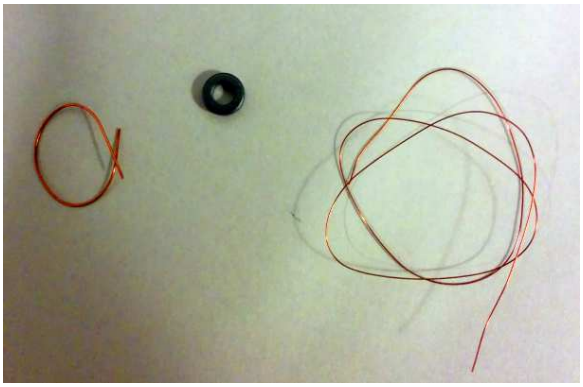

Figure 21: Winding Toroid 1

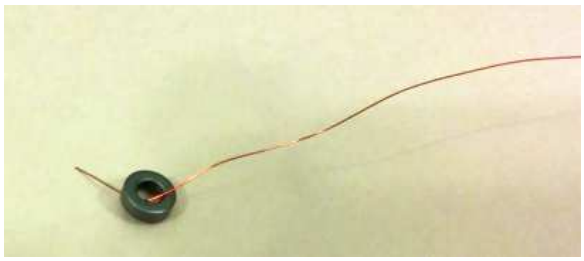

Figure 22: Winding Toroid 2

Now repeat the process with the other (thicker, shorter) piece of magnet wire and wind 3 turns. For convenience, I wind starting about a third of turn clockwise from where the first winding started (Figure 24: Winding Toroid 4). You'll have to scrunch up the turns periodically in order to fit them all on.

Trim the leads to about 3/4” (Illustration 12).

Remove the Enamel from the Magnet Wire

This part is also tedious. Supposedly, you can just burn the enamel off. I've never had any luck doing that. What I actually do is cook the end of the wire with a cigarette lighter or match for a few seconds. This makes the enamel easier to remove with sandpaper. I use a small piece of sandpaper folded in the middle and pull the end of the wire through the sandpaper. I pinch the sandpaper a little to help with enamel removal. A caution: don't pinch the sandpaper very hard with the thin magnet wire or the wire will break. The finished piece should look like Figure 25: Winding Toroid 5.

As a sanity check, verify that both windings have continuity using your DVM (this will tell you if you've done a good enough job removing the enamel).

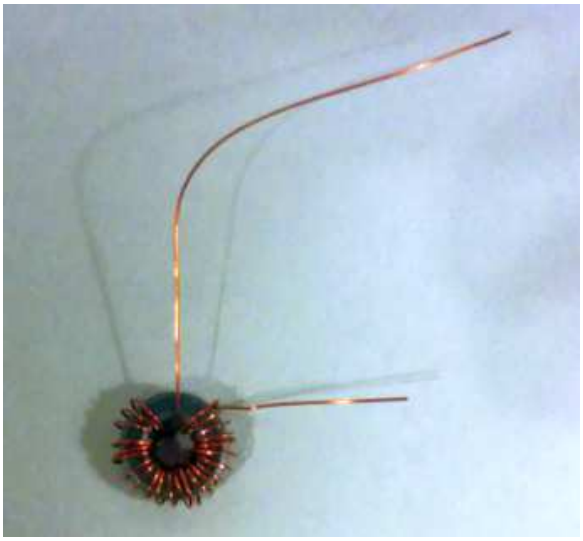

Figure 23: Winding Toroid 3

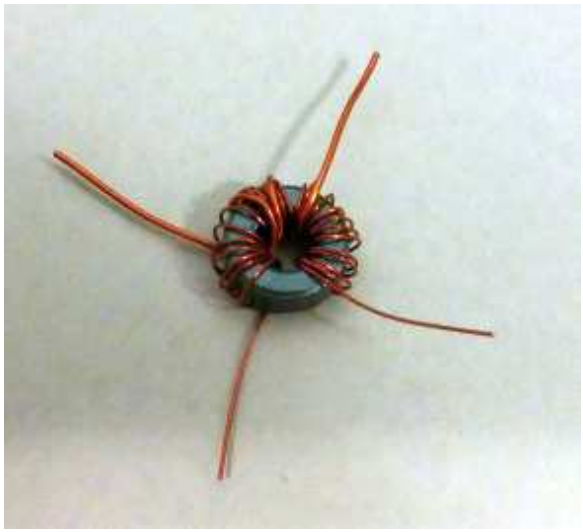

Figure 24: Winding Toroid 4

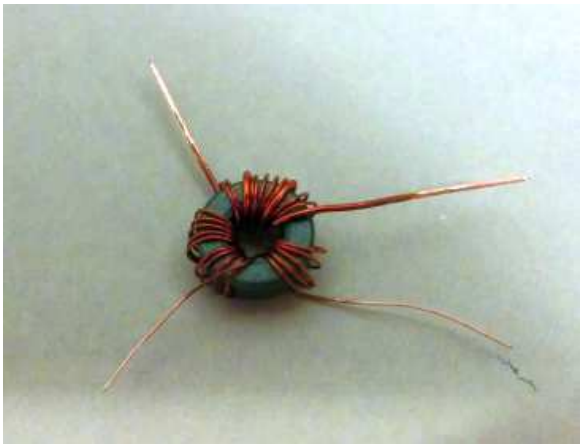

Figure 25: Winding Toroid 5

Assembling Everything

The schematic for the antenna is shown in Figure 26: End Fed Dipole Schematic.

1. Twist together and solder one end of the thin magnet wire, one end of the thick magnet wire, and the braid from the coax (Figure 27: Soldering Toroid).
2. Connect and solder the center wire of the coax to the other end of the thick wire.
3. Connect and solder the the antenna (42 ft long hookup wire) to the other end of the thin wire.

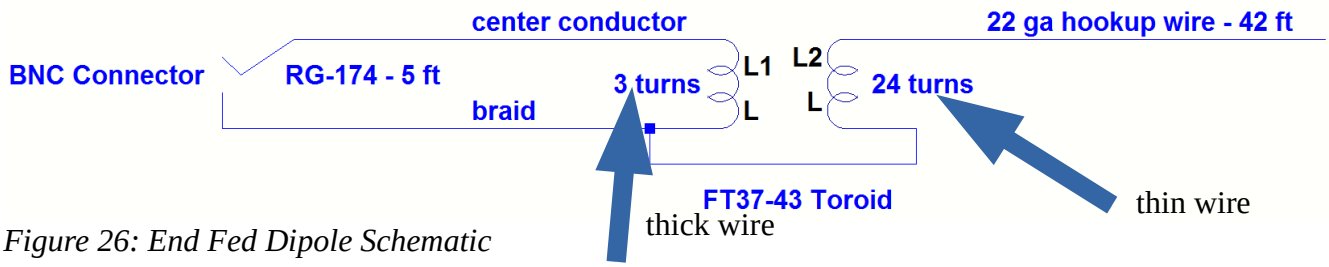

Figure 26: End Fed Dipole Schematic

Your antenna should look like Figure 27: Soldering Toroid.

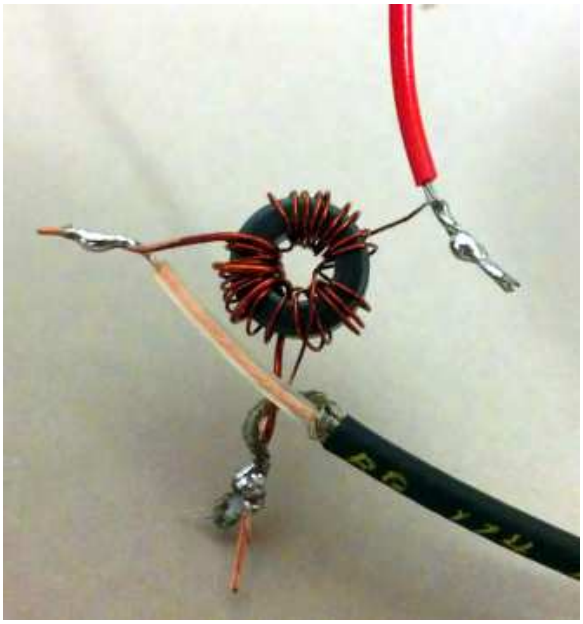

Figure 27: Soldering Toroid

Figure 28: Strain Relief

Wrap the cable tie around the coax and hookup wire as shown in Figure 28: Strain Relief to provide strain relief.

Solder the other end of the coax to the BNC connector as shown in Figure 29: Antenna BNC 1 and Figure 30: Antenna BNC 2. It helps to have helping hands (soldering tool with alligator clips) or someone else with an extra set of hands/vice to hold the BNC connector, but it's not necessary. I find that it's easier to tin the coax center lead and fill the BNC center connector with solder (melt solder in it) before trying to solder the coax to the BNC center connector (Figure 29: Antenna BNC 1).

Figure 29: Antenna BNC 1

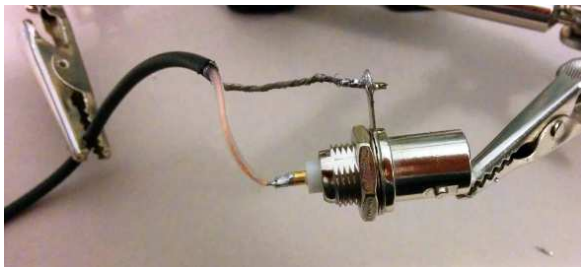

Figure 30: Antenna BNC 2

Warning:

The antenna has been designed to be used with a low-power WSPR transmitter. The toroid is the weak link in the chain. It limits how much power the antenna can safely handle. The magnetic flux in the toroid reaches its maximum allowable value with 0.25 watts of input power. Don’t try running 25 watts (or even 1 watt) into it or you’ll let the smoke out.

Troubleshooting

I Built It, But It Doesn't Work

Well, I'd hoped that you wouldn't ever have to get to this page, but …

I've tried very hard to make sure that everything is correct and that you're working with good parts. However, sometimes stuff happens ;-) The good news is it's all fixable (except for soldering that 40 pin connector on the wrong side of the circuit board). Follow the steps in the order given. I'm assuming that something isn't working right at this point, so let's find out what it is.

1. Check that the components are assembled on the pcb correctly and that the pcb is plugged into the rPi correctly.
2. Check that you've got the rPi Ethernet cable plugged into the rPi and that the other end of the Ethernet cable is going to your router (not your laptop).
3. Check that the red LED on the rPi turns on when power is applied.
4. Verify that the software is working by bringing up the web page (type the IP address into the address line of your browser on your PC/Mac/Linux box: 192.168.aaa.bbb). If the web page displays then the software is probably working okay. The rPi should have power for this test.

   You may have to run the WSPR_Locater program to find the IP address of the rPi and then enter the IP address into the address bar of the browser.

   If the web page doesn't display or WSPR_Locater can't find the rPi then press and hold the shutdown button for 2 sec and unplug the power supply. Wait for 30 sec and plug in the power supply. Wait for the LEDs to stop blinking and try to access the web page again. If this fails to work there may be a problem with the software load on the SD card.
5. Press the pushbutton on the AFB and hold for two seconds. Wait until the LEDs on the rPi stop flashing and unplug the power to the rPi.
6. Unplug the power supply and remove the AFB from the rPi. Look at the board. Is anything odd looking or out of place? Look at both sides of the board.
7. Check component values. Check the values of all of the resistors and capacitors.

   Resistors generally have four color bands. The first two are the resistor value. The third is the multiplier. The fourth is the tolerance. So you read, say, white-brown-red-gold as:

   white = 9, brown = 1, red = 2, gold = 5

This is interpreted as 91 followed by 2 zeros, or 9100 ohms. The gold means the tolerance is 5%. It's really easy to look at a resistor backwards and see gold as yellow and white as silver. In that case you would read this as

yellow = 4, red = 2, brown = 1, silver = 10

which would be interpreted as 42 followed by 1 zero, or 410 ohms and have a 10% tolerance. As you might have guessed, interpreting a 9100 ohm resistor as a 410 ohm resistor might cause a problem.

Also, as a matter of convention, we tend to avoid writing out lots of zeros. The convention for that if you're unfamiliar is a suffix. The useful suffixes are M (mega/million), K (kilo/thousand), m (milli/one one thousandth), u (micro/one one millionth), n (nano/one one billionth), p (pico/one one trillionth). Think of the suffix as a multiplier.

For example, using our 9100 ohm resistor, we would list that as 9.1K. 9.1 thousand is the same as 9100.

Second example, 0.000,000,000,680 would be written as 680p (you can see the utility in using the multiplier notation).

Make sure that you haven't read any resistor values backwards.

Capacitor values are shown as three digits. The first two digits are the value and the third digit is the multiplier (number of zeros to add). Unfortunately, you have to sorta know what multiplier to use. Capacitors are typically listed in either uF (micro farads) or pF (pico farads). If the value is given as a three digit number then it's in pF (pico farads). If it's given as a small number (typically less than 1.0), then it's in uF (micro farads).

So a capacitor shown as 681 would have a value of 680 pF (68 plus one zero) and a capacitor with a value of 0.01 would be 0.01 uF.

There are three inductors and unfortunately, they look just like resistors. The color coding is even the same as used for resistors. The units are micro henries (uH). The other unfortunate part is that the 1.2 uH inductor color coding appears to be mismarked. It's marked as a 1.0 uH inductor. It does in fact measure 1.2 uH. So the two inductors that look alike are 1.0 uH. The inductor that looks more like a dumbbell is actually 1.2 uH.

8. Verify that the components that are supposed to be aligned in a certain way are actually aligned that way. This includes the LED, diode D1, and the MOSFET Q2.

9. Okay, it's time to reassemble the circuit board to the rPi header, get our your voltmeter and turn on the power. The voltmeter ground lead should be connected to the ground test (*Gnd*) point next to the push button.

I've listed voltages in the schematic (below) as *transmit/idle*, so for example, the voltage on the power supply rail (top wire in the schematic) is shown as (4.51/4.67). This means that I measured 4.51 volts between the power supply rail and ground (*Gnd*) while the unit was transmitting, and 4.69 volts when is was not transmitting (idle).

Keep in mind that I'm using a cheap DVM, you're probably using a cheap DVM, and component tolerances could be all over the place. This means that if your readings are within, say 10%-20% of mine that they're probably okay.

(See the Amplifier/Filter Board Schematic, Figure 35: WSPRnet Database).

I'd suggest printing out this page and measuring and recording the voltages listed. Actually, it's probably simpler to just measure the voltages (and write them down) on both sides of R5, both sides of L1, and both sides of R2.

The higher voltage across R5 is the power supply rail. It should be between 4.5-5.0 volts. The voltages on both sides of L1 should be about the same when not transmitting and one significantly lower than the other when transmitting (LED on). The voltages on both sides of R2 should be about the same whether transmitting or not. The voltages at R2 will vary depending on how pot R1 is set. The important part is that they're above 0.0 volts and have pretty much the same voltage on either side of R2.

If you find voltages that differ significantly from what's listed on the schematic, start by verifying component values and looking for bad solder joints (resolder the connections of all of the components in line with the bad voltage). If that fails, let me know because we're probably looking at a bad component, which I will be happy to replace.

Getting SSID and Passkey

Start by looking on the router for login information. Also, try the Internet for a router manual (I know, I hate to read directions, too).

Next, you're looking for your router's web page. Open a browser and type the IP address *192.168.0.1* into the address textbox at the top (just as if you were typing www.google.com). If nothing happens try *192.168.1.1* and *192.168.2.1*.

Log into the router using the username and password printed on the side of the router or in the manual.

Look through the various wireless setting for *SSID* and passkey (or something similar). I'm sorry, but I can't offer too much more because each router will be different.

Appendix 1 – Antenna Response

I measured the antenna using a miniVNATiny from miniRadioSolutions.com and got the following responses (Illustration 32).

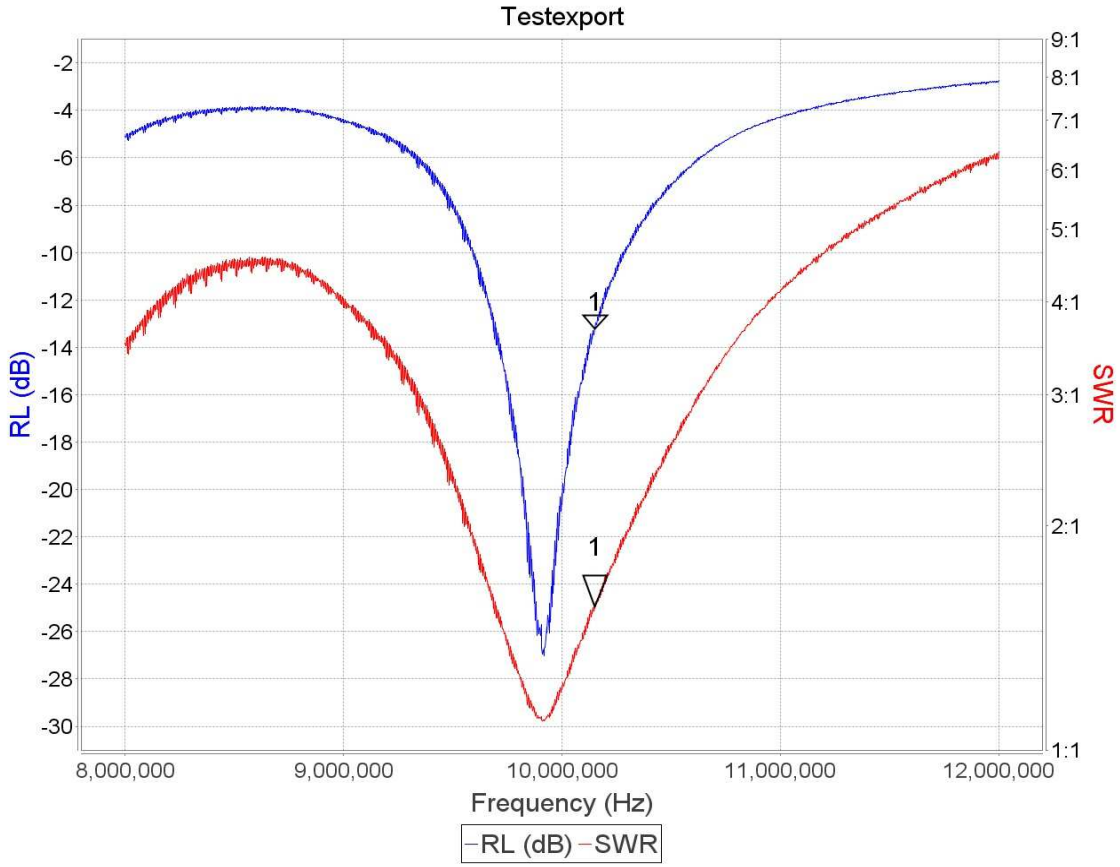

Figure 31: Antenna Response - Raised 6'

According to the antenna formula for a dipole the optimal length for the antenna is 46 ft 2 in (468 ft/10.14 MHz = 46 ft 2 in). I chose 41 ft because that length seems to match the suboptimal elevation I'm expecting the antenna to actually be used at. My antenna was draped across several door hinges for the response plot. I'm expecting that the antenna should work adequately at low elevations. Of course, it's still not as good as an dipole at a reasonable elevation.

Appendix 2 – Programming an SD Card

Okay, something happened, and despite your best intentions somehow the SD card got messed up and you're stuck trying to reprogram it. It happens. Linux (what's running on the rPi) is a great operating system, but it is finicky about the file system (on the SD card).

Setup

1. SD Card; at least 4GB, 8GB is better. Class 10.
2. SD Card reader/writer. I've got a Targus Micro SD/T-F. It doesn't really matter. It connects your SD card to a USB port.
3. Win32DiskImager – It's a freebee. Download it from: *sourceforge.net/projects/win32diskimager/*.
4. Install Win32DiskImager.
5. Disk Image – Download it from my website – *WsprWithoutTears.com*

   Click on *WSPR Baseline link*. The screen will display a gibberish line (link to the file on DropBox). Copy this to the clipboard (highlight it with your mouse and type <CTRL>-C while it's highlighted). (<CTRL>-C means hold the control key down while pressing the C key.)

   Click in address bar of your browser (where you'd type *www.google.com*), type <CTRL>-A followed by <CTRL>-V. Press *Enter.*
6. Unzip the file after it's finished downloading (it will take a while – this is a good sized file).

Program SD Card

1. Plug the SD Card reader/writer into a USB port.
2. Plug the SD Card into the reader/writer.
3. Start Win32DiskImager.

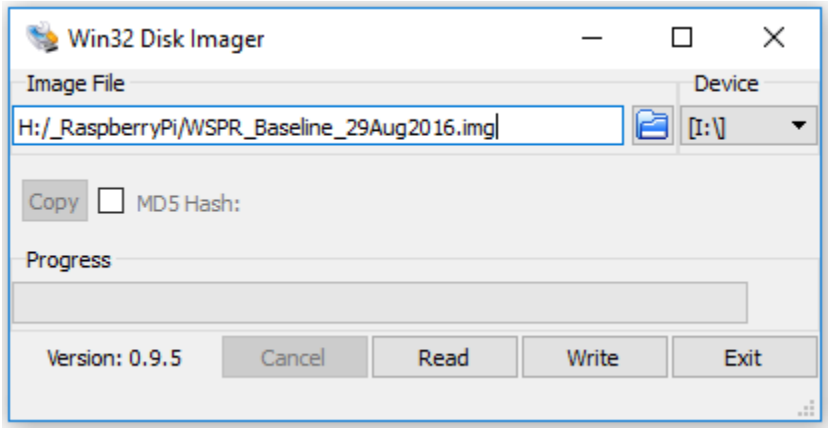

Figure 32: Win32DiskImager

4. Click on the folder icon in the upper right and find the disk image you downloaded and unzipped.
5. Click *Write* and go get a cup of coffee. Note: Win32DiskImager shows a small message box when it's done writing the SD card. Unfortunately, the message box likes to hide behind anything else you have up on the screen so you can't see it. Be forewarned that you might have to go looking for it.
6. Click *Exit*.
7. Remove the SD card and install it into the rPi.

If you haven't annoyed the gods of the Internet too much today you should be in luck and everything will work!

Appendix 3 – Communicating With the rPi over SSH

This sorta seems like the antithesis of what I've been trying to accomplish. I wanted to hide the hard details so that you could concentrate on getting a WSPR signal on the air. Okay, you're not afraid of Linux and you want to talk to your rPi over SSH. Actually, I sympathize – that's my favorite mode for talking to the rPi.

1. Download the program KiTTY.exe (no, not a cat) as part of the *WSPR Executables* from WsprWithoutTears.com or from http://www.9bis.net/kitty/?page=Download.
2. Get the IP address of the rPi from WSPR_Locater.
3. Run the KiTTY and enter the IP address for the rPi.
4. Username: ***pi*** Password: ***wspr***

Appendix 4 – Postprocessing Data

The first thing you're going to want to do once your WSPR kit is up and running is to see how your signal is getting out. Go to the website *www.wsprnet.org* (enter *www.wsprnet.org* in a browser) and click on *Map* in the upper right hand corner.

WSPRnet Map

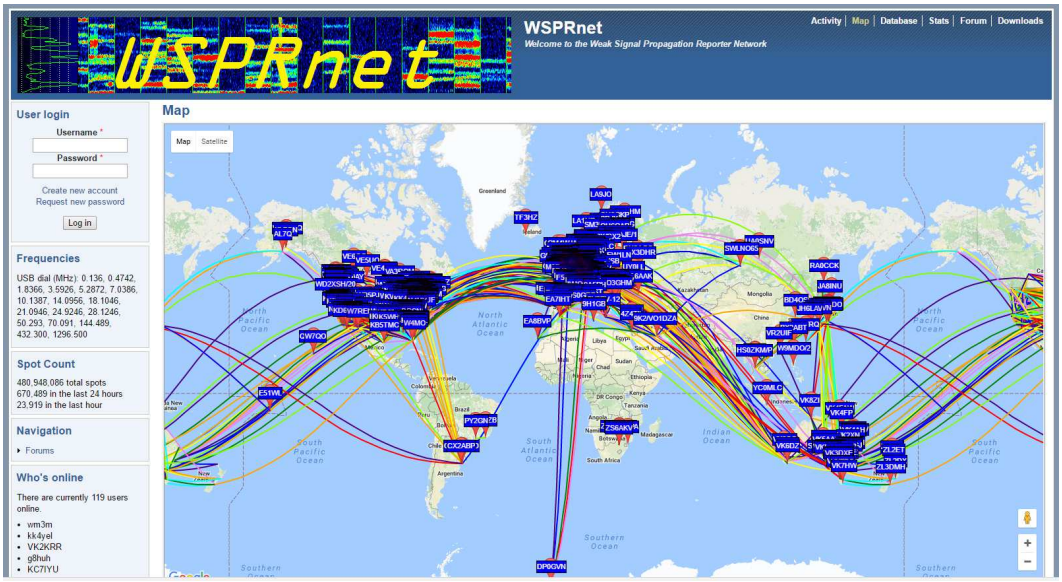

Figure 33: WSPRnet Display

Scroll down to the bottom of the graph and change Band to 30m, insert your *callsign*, change the *Time Period* to 3 hours, click on the *Day/Night* overlay checkbox and then click on the *Update* button. You can pan and zoom to see how far your signal has gotten.

Here's a typical view.

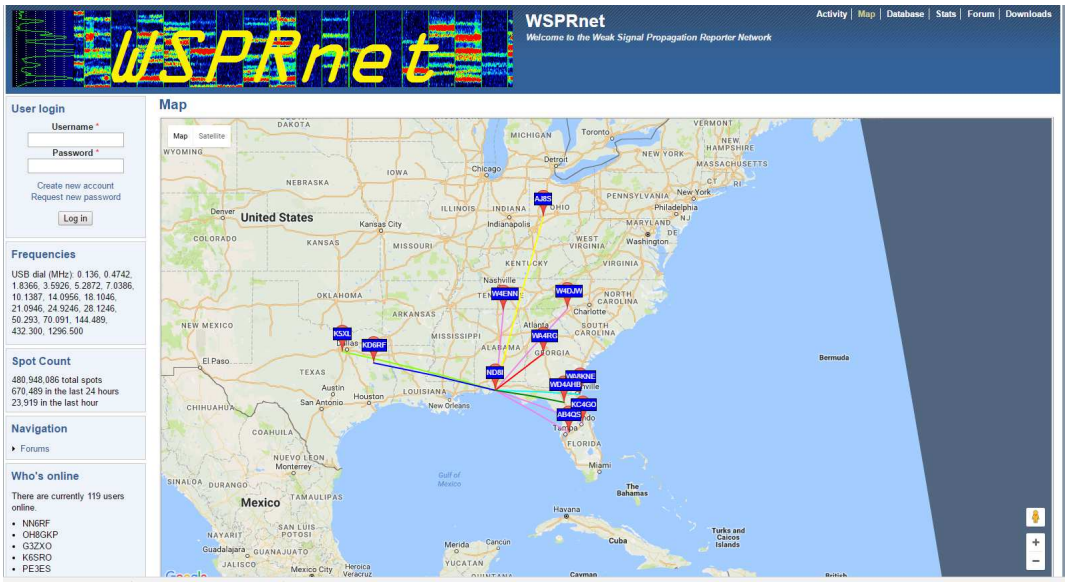

Figure 34: WSPRnet Display - Zoom

Now click on the *Database* button (next to *Map*) in the upper right hand corner. Click on *Specify query parameters* just below Database in the upper left and change *Band* to 30m and add your *callsign*. Click the *Update* button.

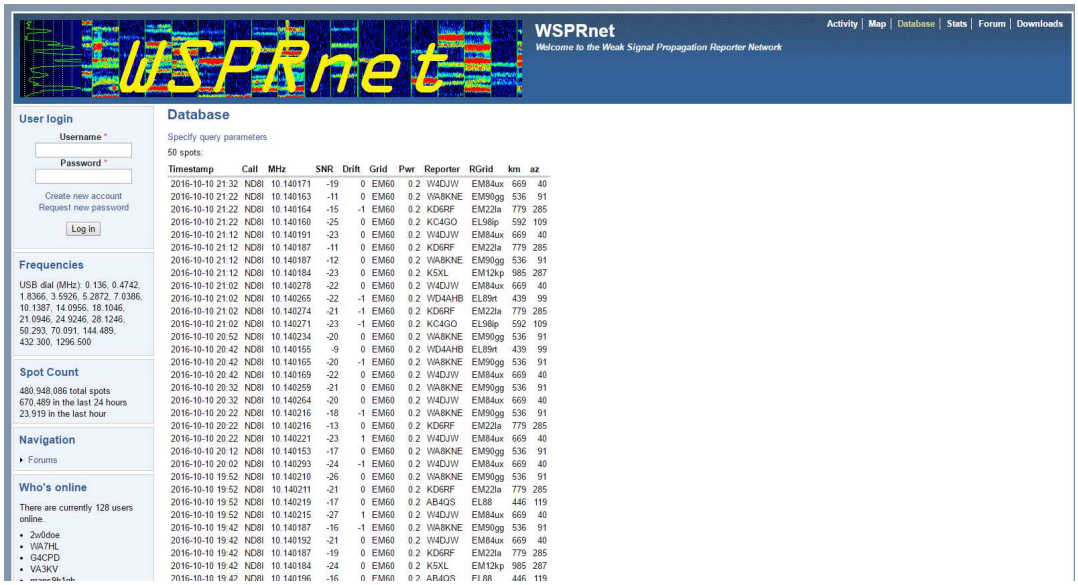

Figure 35: WSPRnet Database

This gives the particulars for the contacts – their gridsquares and the distance traveled. (Multiply km by 0.62 to get miles.)

Downloading Data

I'm fascinated to see how far my signal goes using just a piece of wire strung around the room (that's all I've done with my antenna). It's also interesting to see the effect of daytime vs nighttime on propagation.

After you've run your beacon for a while you might be tempted to play with the data some more. The WSPRnet people have stored information on all of the transmissions in files that you can download and process. This is under the *Downloads* tab. The files are stored by *year-month*. They are available in gzip or zip format.

Pick a month that you'd like to investigate and click on the word *zip*. I'm amused by what can only be described as wishful thinking on the part of the people who put together the web page. The writeup at the top says “Compressed file sizes range from 1-20MB.” The compressed files will be much closer to 300 MB and uncompressed go to about 1.4 GB.

Most of the entries were not information I cared about – they list *all* of the contacts in a particular month. I only want to see my contacts.

WSPR Data Filter

I wrote a program to extract only my contacts. The program is called *WSPR_Data_Filter.exe* and can be downloaded from the web site. The program removes everything but your contacts from the downloaded data.

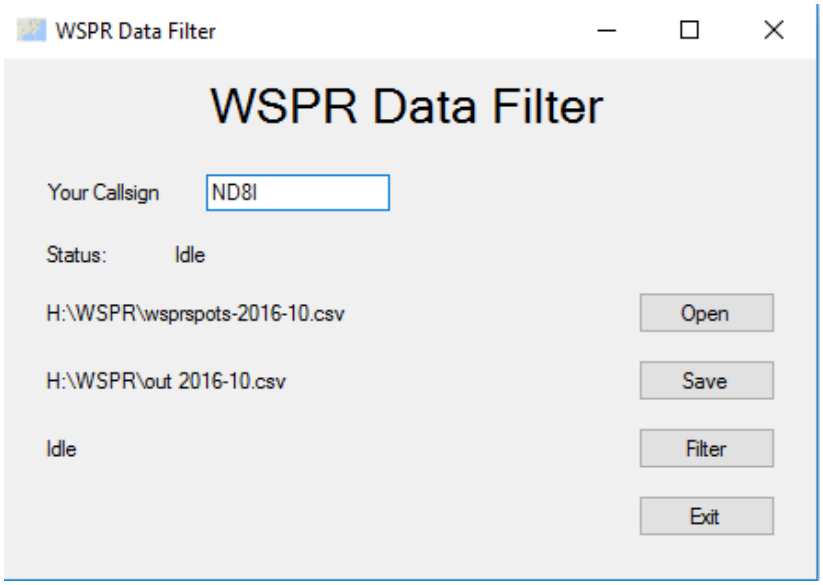

Figure 36: WSPR Data Filter

Here is how you use it -

1. Download a month's worth of data from WSPRnet.org.
2. Unzip the file (these are big files – it will take some time).

3. Download Run WSPR_Data_Filter (download from http://Wspr.WithoutTearscom/, click on *WSPR Executables* and extract the program from the .Zip file).
4. Run WSPR_Data_Filter.
5. Enter your *callsign*
6. Click the *Open* button, select the file you unpacked, and click *Open*.
7. Click the *Save* button, enter a *filename* to save the output as, and click *Save*.
8. Click the *Filter* button.
9. Click the *Exit* button when processing is done.

The output can be read into any spreadsheet program (e.g. Excel) directly for processing and graphing. I have written some post processing software using Octave to give a broader picture of your data (see below).