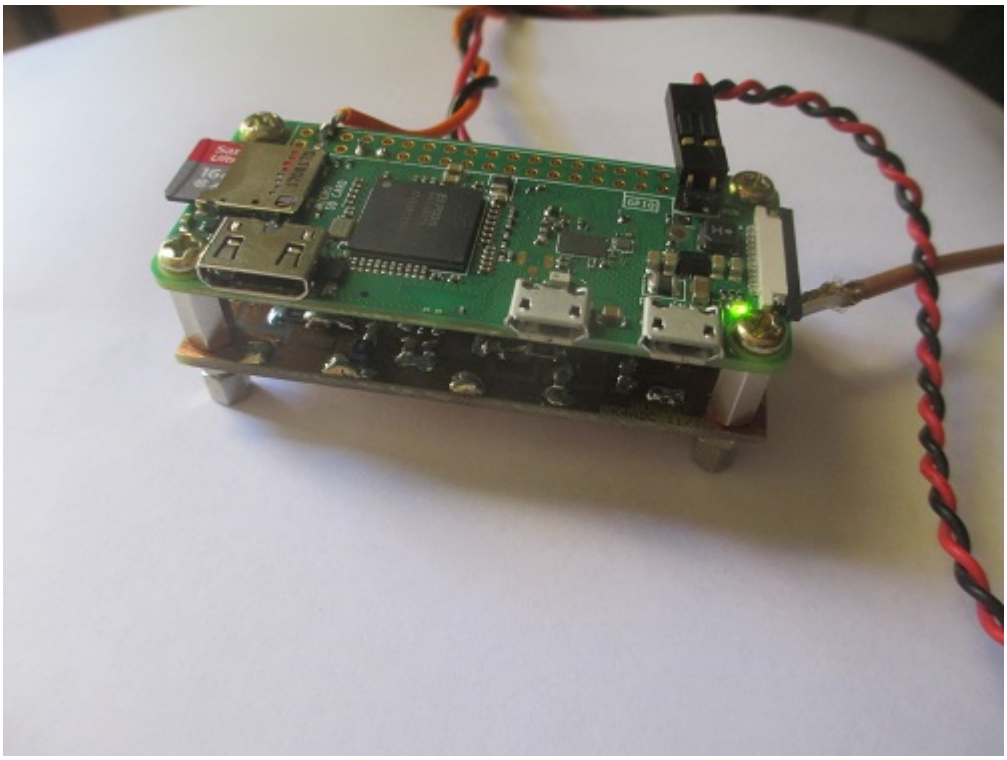
My Slice of 'Pi'

27 December 2018



My initial 'Pi' Sandwich.. Pi Zero W + PA PCB underneath..
Read/black lead to 2 pin header is to a reset switch..

WSPR is ‘all the rage’ at the moment on 50MHz so I thought I would contribute to add to the very few WSPR stations actively and routinely transmitting from the QG62/Brisbane area.

A while ago I chanced across an article about using a Raspberry Pi as WSPR beacon transmitter so that seemed the low-cost approach to achieving that. I did a lot of ‘Google-ing’ and found a lot of articles about the concept then ordered a Raspberry Pi Zero W (with onboard wireless) for $22 posted. I bought a couple of class 10 16GB microSD cards too ($8 each), downloaded the RaspBian code on my PC.

Note: I have provided titles for you to search for as the URLs may change.

Another article “HEADLESS PI ZERO W WIFI SETUP (WINDOWS)” provided the best methodology I found for setting up the RaspBian Stretch software for the microSD card to boot the Zero. I have to admit that I used Rufus-3.3 to put the ISO image on the card having used Rufus previously for other software projects using USB sticks and SD cards. There are two important steps after the ISO is transferred: the file wpa_supplicant.conf which unlocks the WLAN access and the file SSH which opens up the system for PuTTY access.

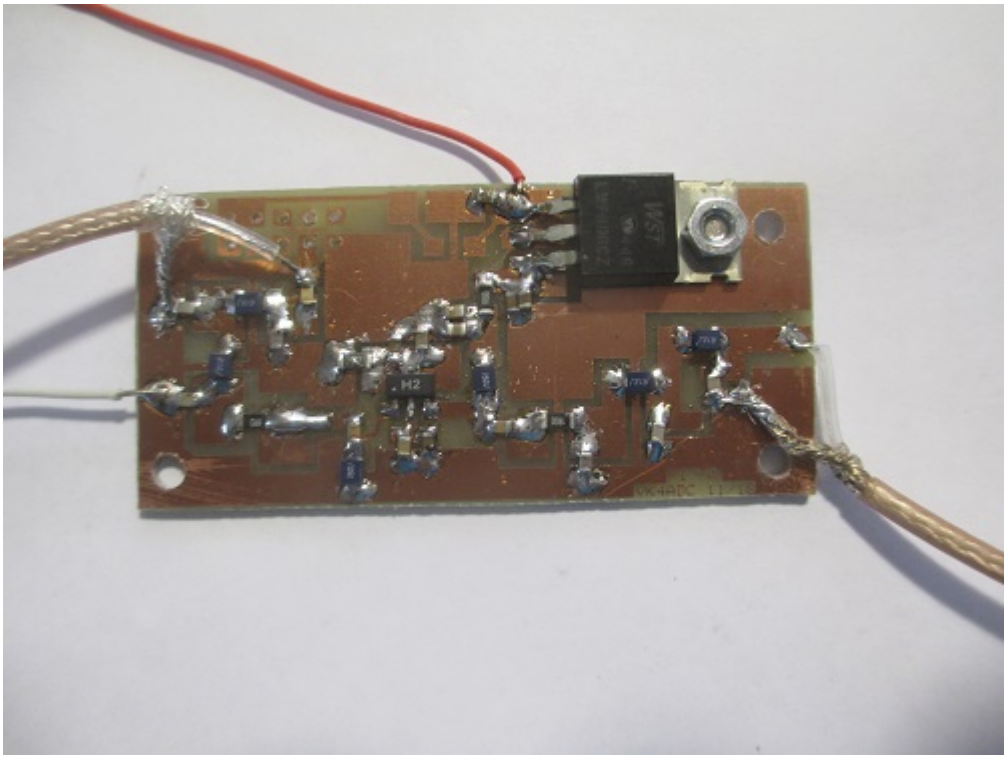
Fortunately I use Debian on my local webserver so I wasn’t entirely lost with the Linux OS requirements. If you follow the instructions exactly – and I mean so exactly that there aren’t even any extra spaces – then it all usually works. I found copy-and-paste is the best method: copy from the article and paste into the Putty SSH application once you are up and running the RaspBian OS.

Ditto installing the WSPR code as described in “GitHub - JamesP6000/WsprryPi: Raspberry Pi WSPR transmitter using NTP based frequency calibration “. Copy and paste as much as you can from the instructions sequence in that article..

That gets you to the command line ready to get the Zero ready to go. The command from the $ prompt is something like ‘sudo wspr –f VK4ADC QG62lg 10 20M’ should get something happening. Just using the Pi and a short length of wire on GPIO4 (pin 7, with pin 9 = gnd), I was able to hear the WSPR signal being generated on 20M - the tone frequency varying very slightly as the data encoding is applied. At that point, you are generating WSPR but the story doesn’t end there. A brief explanation is probably best at this point: ‘wspr –f VK4ADC QG62lg 10 20M’ where the VK4ADC is the callsign to use, the QG62lg is the 6 character grid square, 10 is the power in dBm – 10mW for a basic Pi output, 20M is the band to be used. The command line will produce a single WSPR transmit cycle on about 14095 KHz. Hitting the Up arrow will let it re-prompt and then the Enter key repeats the WSPR sequence.

You MUST apply at least low pass filtering (LPF) – preferably band pass (BPF) filtering in lieu – between the Zero and an antenna. I found that the articles by YU1LM were helpful ('HF/50MHz Receiving and Transmitting Band Pass Filters'), particularly Part/Ver 3, as they are designed around standard values of inductors and capacitors. Mine didn't come out with the pass frequencies exactly as described but that is a side issue.

I wasn't satisfied with the idea of having the +10dBm (10 mW) output level so searched through my SMD parts to find a SHF-0289 1 watt SOT-89 microwave FET amplifier device (more details later). The PA was built up to be the same dimensions as the Pi Zero so that they could be 'stacked' and was a double sided PCB material with the second size unetched. The configuration was as a self-biased amplifier (ie using a source resistor) and powered from a 7806 +6V 1A positive regulator on the PCB. The input to the gate was via a LPF with 3dB cutoff of 74MHz, the output from the drain similarly via another LPF with the same characteristics. With the Zero driving it at 50MHz and through the LPF on both the input and output, I was able to see +25.5dBm from the output – about 250mW – a far more useful power level. At that point I was able to generate a useful WSPR signal so I connected it to my 6M J-pole and left it running for a while. No spots on WSPRNet resulted. (It turned out that there was no sporadic E and no locals on 6M at the time !) I plonked it back onto 20M and still using the 6M J-pole, was spotted immediately by VK3WHO so I knew it was working. Later that day after putting it back on 6, the 6M WSPR spots started appearing. The only drawback on using the Zero for WSPR was the presence of wide-band noise even on HF while transmitting but I hope that (eventually) it will be solved by more filtering.



PA PCB during testing.. about 250mW out when driven with the Pi Zero W.
PA FET is labelled with M2, all SMD parts except for the 3-terminal +6V regulator.

During my reading about WSPR beacon activities, I noted that there was a need to make the frequency shift around a bit so as not to obscure weaker signals and while the WsprryPi software has this as a command line option, I chose to approach it slightly differently because I opted to use the -f (free run) mode since my Zero seemed to take an inordinately long time to frequency stabilise. My command line was altered to 'sudo wspr –f –r VK4ADC QG62lg 25 50294500 50294520 50294480' – i.e. a shift of 20Hz either side.

That was all fine as far as I was concerned and then I started getting emails: did I know that I was transmitting EVERY 2 minute period ? Well hello, yes. It seems that that is not *proper* to do that and that a 1-in-5 is more acceptable a transmit duty cycle. Just remember that this is not a transmit/receive operation – purely WSPR transmit – so what am I supposed to do ? This led to a revamp of the command line to 'sudo wspr –f –r VK4ADC QG62lg 25 50294500 0 0 0 0' – a 1-in-5 timing revision – and no enforced frequency shifting. One thing I might mention is that at 50.293, the Raspberry Pi Zero has a downward drift of 3-4 Hz over the 100-odd second transmit period. I expect that by (eventually) fitting a small square heatsink over the CPU and extending that over to the X1 crystal oscillator that this drift might be reduced once temperature stability is reached.

Something to remember at this stage is that after DC power is applied, the Zero boots you to the login prompt and you have to login, then do the ‘sudo..’ bit each time. The article “The VK4PK Whispering Pi” article in Part 8 holds the answer there. Making the Zero boot into a script that runs the WSPR command line is the outcome. It is a bit tricky but it does work, remembering that you need to insert your callsign and grid square data in the relevant places. I also commented out all of the unwanted lines by inserting a # at the beginning of each of those lines. One step that is missing from that article is the command “ sudo chmod 755 /home/pi/WsprryPi/daemon.sh” and if that isn’t done after the daemon.sh file is created then the service will respond with a 203/Exec error after the reboot and when checked with “sudo systemctl status qrpi.service”.

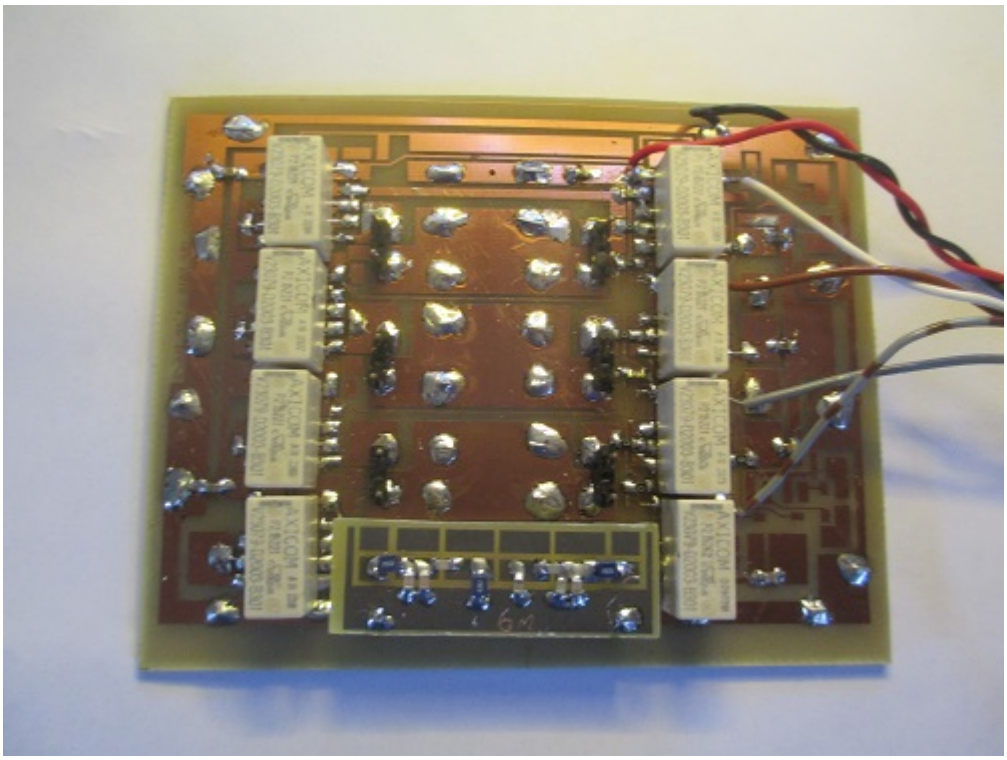
There are several useful commands to control the running transmitter, "wspr" via the qrpi service tag. On some occasions you may want to turn the wspr off or run the process manually from the terminal login. They are as follows:

"systemd" commands

FUNCTION	COMMAND
Reload all the systemd configurations	sudo systemctl daemon-reload
Stops the service immediately	sudo systemctl stop qrpi.service
Restarts a service	sudo systemctl restart qrpi.service
Shows the status the service (including whether it is running or not)	sudo systemctl status qrpi.service
Enables the service to be started on bootup	sudo systemctl enable qrpi.service
Disables the service to not start during bootup	sudo systemctl disable qrpi.service

Now comes the realisation: I am virtually wasting 8 out of every 10 minutes while the Zero is not producing WSPR output. What can I do about that ? Of course the obvious answer is to make it generate on three or four other bands. That needs different BPFs in line as the bands are changed – easily enough on the WSPR command line – but that is only the generated frequencies and not the filter switching.

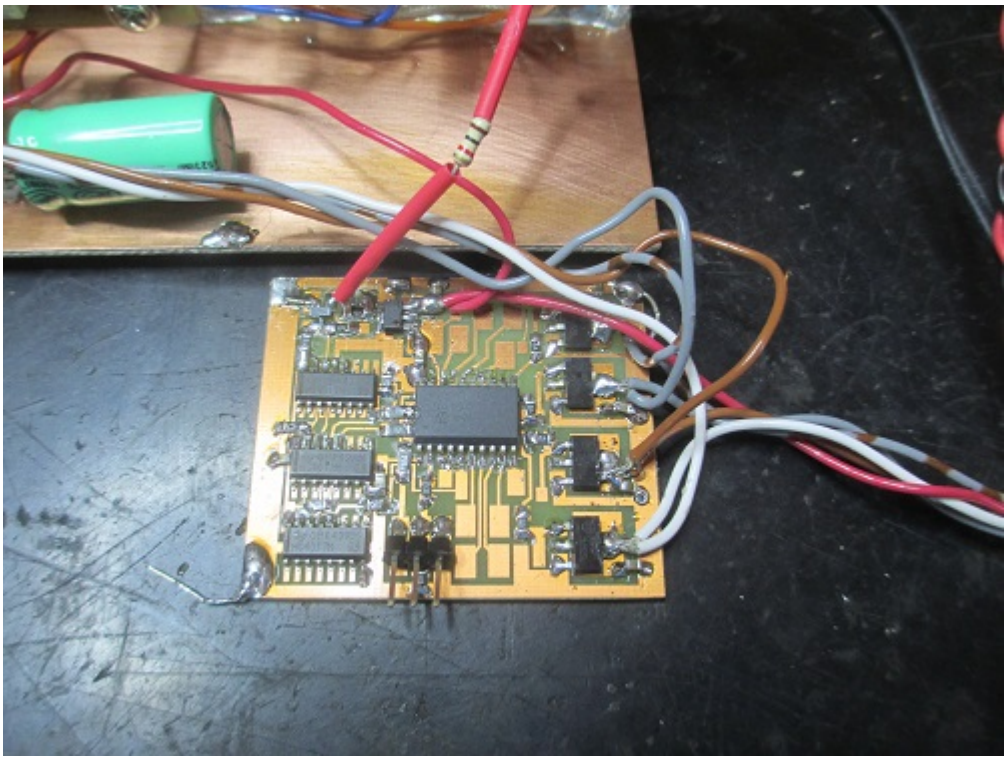
An article by KD0VJS titled ‘My Setup Using a Raspberry Pi, RF Amp and Relay Switch’ on WSPRNet provided an idea as to how to utilise extra GPIO pins to operate a relay module to achieve multi-band functionality so I set about laying out another (95 x 70mm) PCB that would support multiple BPF designs and creating filters for the selected bands : 6M, 30M, 20M and 17M (although other bands could be substituted at a later stage). Two SMD relays per filter would switch input and output and thus provide isolation filter-to-filter. The current WSPR command line coding would not work this way so some extra work would be required to achieve that functionality.



The filter switching PCB with the 50MHz BPF installed. Three other filters on filter sub-assembly boards fit on pairs of pins at each end. The board layout caters for two different common types of SMD DPDT relays and, while not fitted, has transistor driver options available so that it could be run directly from the GPIO pins on the Zero per the KD0VJS article.

Then the brain struck again… I had previously worked on a little project using a PICAXE plus external 74HC4017 dividers that would work up to 50-60MHz. If I sampled the RF from the Zero, I could make the appropriate filter switch in within a matter of 100-200 milliseconds by 'counting' the frequency and switching an output pin - while the total WSPR transmit cycle is a continuous RF carrier for around 96-110 seconds. To be candid, that approach was better for me: I could maintain my current software configuration for the WSPR command line and the process of band switching would be automatic as soon as RF was detected.

I set about laying out (yet another) PCB based (40x50mm in size this time) on a 20X2 PICAXE plus 3 of the HC4017 devices (giving an external divide by 1000) and providing for 4 FET-driven output pins that would correspond to the 4 filter relay pairs to be switched. The code was actually quite simple – using the divide by 1000 output from the 3rd 4017 and counting pulses for 100mS gave a nice easy output value, around 25 (pulses/counts) per MHz of input frequency with the PICAXE running at 16MHz. This value was then tested in the software against a set of pre-determined values range (with a SELECT CASE statement) and the relevant Port B output 'switch' output pin turned on & turning all others off. This switching transistor (FET actually) then pulls the relevant band's filter relays into circuit. There are 4 more unused port B pins on the 20X2 that could be fed to extra FETs to extend the capability to 8 filter relay sets but that would also mean an extra duplicate 'filter band-switching PCB' would need to be added. Not difficult other than the space required – and no PCB changes required – but not warranted.



This is the PICAXE 20X2 relay controller PCB: RF in and a switching function out.
The 'mid-air' 1K resistor reduces the loading that the input stage of the PICAXE board places on the GPIO4 pin on the Zero PCB.
Layout : 3 x 74HC4017s (cascaded) at LHS, PICAXE 20X2 controller at centre, 4 x NTF3055 FETs at RHS.
Programming header connector at bottom centre.

Voila, automatic band switching in the presence of RF from the Pi Zero/PA. The beauty of it is that by narrowing each set of band values in the PICAXE code, it is easy to make the filters ‘drop out’ out-of-band – and when it does that (ie all relays drop out) then the output is automatically terminated on a 47 ohm resistor on the filter switching PCB.

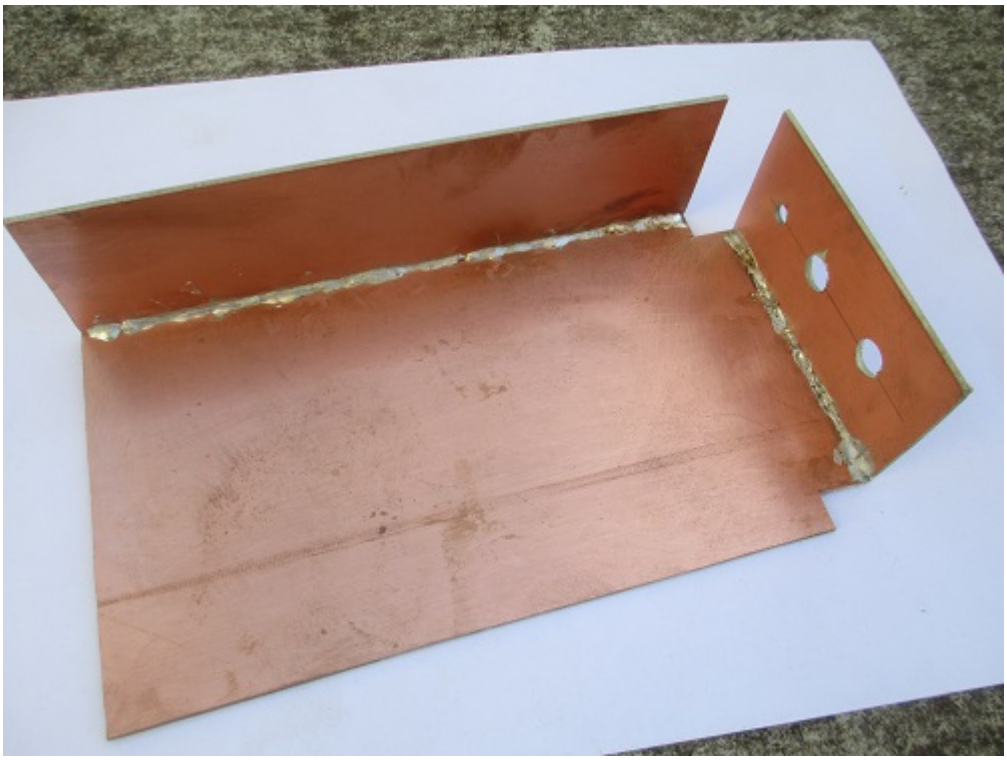
The WSPR outcome would typically be a 1-in-4 duty cycle PER BAND with each and every two minute WSPR period used for transmit but I was OK with that – the WSPR police be damned.

In reality, the initial band switching was changed to be 6M, 30M, 20M, 6M, 17M, 15M then repeat ad infinitum. That was possible because the 18MHz BPF passed the 21MHz RF with only about 1dB extra insertion loss. The WSPR beacon was placed on-air in multiband mode on 26 Dec 2019 and subsequently spots were noted on 10, 14, 18 and 50MHz within a few minutes of being connected to external antennas.

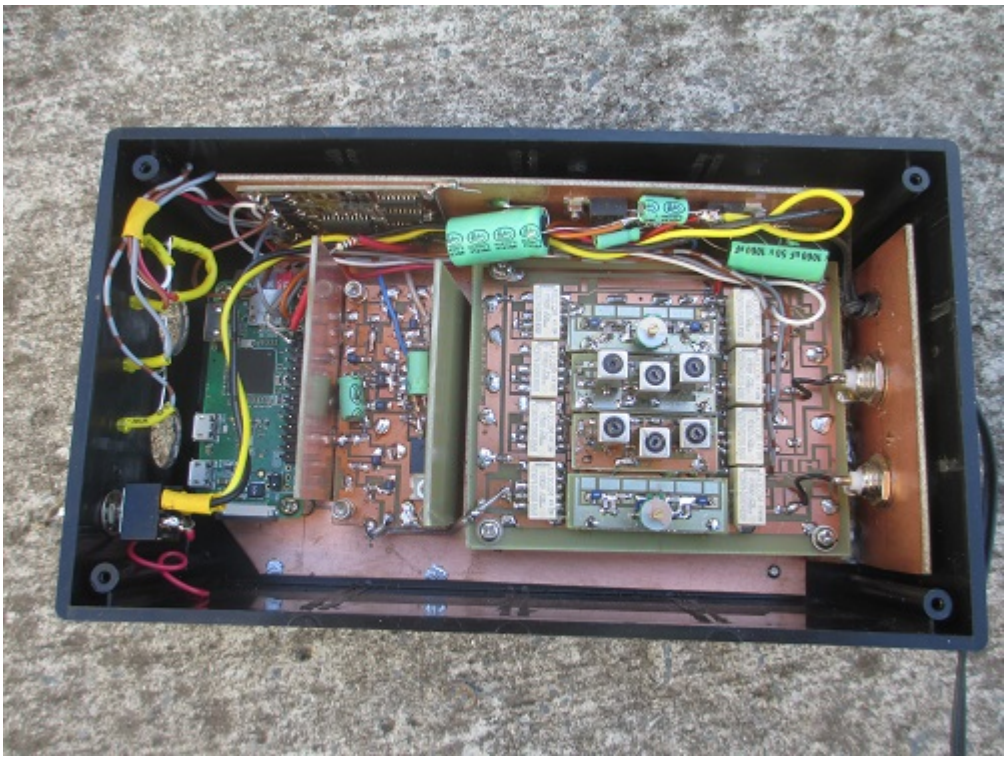
Of course the cost of my WSPR project had blown out. Instead of my original $40 investment, and not including the parts from my expansive ‘bits box’, there I was buying in more SMD relays (about $2.80 each) plus more photo-sensitive PCB materials. The 20X2 chip in SMD form is around $15 in itself.

The whole project now cost more like $100-$120 in bits and a far cry from the original view : a low power WSPR 6M-only beacon that I could run 24x7 off my solar-charged battery bank. The battery bank power is still a ‘go’ although the power consumption is a bit higher with the PA and relay currents taken into account.

The final assembly of my Pi beacon:

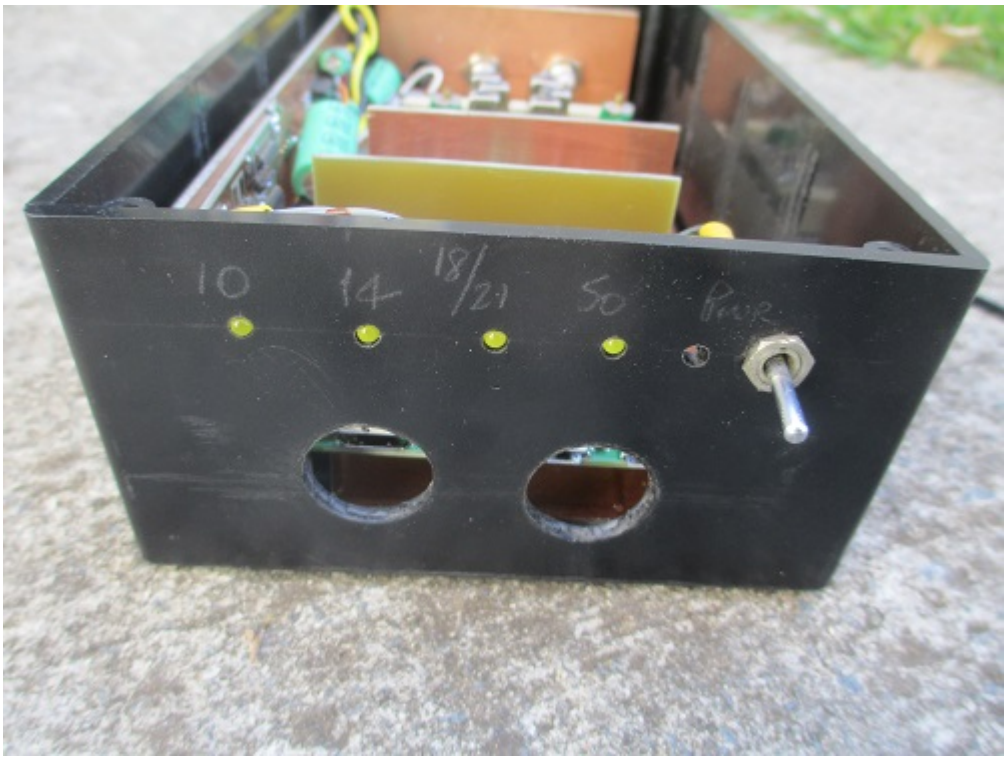


The PCB baseboard beforehand...

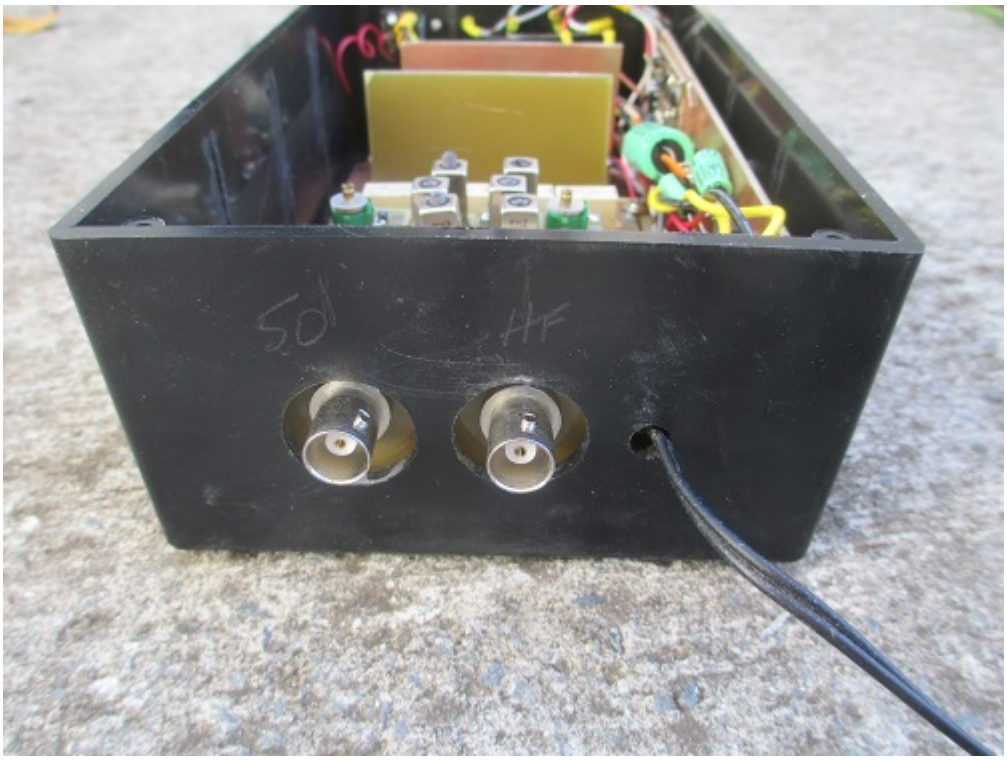


And after the subassemblies have been fitted..
Pi Zero W at LHS,
PICAXE relay controller PCB vertically above it,
PA PCB to Zeros' right,
filter switching board with more BPFs in place towards RHS,
HF BNC output at top RHS, 6M BNC output at lower RHS.

2 x 3-terminal 1A (+8V and +5V) regulators plus electrolytic capacitors (green) on vertical of PCB baseboard.



'Front" View,
large hole at left is for HDMI-to-miniHDMI adapter to fit through,
hole at right is for microUSB 'OTG' adapter to fit through.



'Rear' View,
6M BNC at left, HF BNC at right,
figure-8 power lead through hole.

Then there is still the probable matter of making up a separate multi-band antenna for HF so that it can stay connected all of the time plus another 6M omni-directional antenna, maybe a turnstile /'crossed dipoles' or a halo/squalo (square-halo) so that it is horizontally polarised.

I guess the good news for me is that my basic 200mW of 6M WSPR with the J-pole has been copied in all Australian states, New Zealand, Fiji, New Caledonia, and possibly elsewhere that I haven't noticed too, so it is serving as a useful propagation tool. Coupled with my WSPRView for Windows software, it is easy to see when and where propagation is available to my grid square.

Just a few reports as to how it was working:

2018-12-18 05:00,VK4ADC,50.294489,-25,VK5PJ,PF95mk,1525,

2018-12-18 05:10,VK4ADC,50.294492,-9,VK3XL,QF21pw,1352,

2018-12-18 05:10,VK4ADC,50.294540,-21,FK1TS,RG37fr,1488,

2018-12-18 05:20,VK4ADC,50.294491,-18,VK7BO,QE38mo,1612,

2018-12-18 05:30,VK4ADC,50.294467,-22,VK5AKK,PF94ix,1575,

2018-12-18 05:30,VK4ADC,50.294478,-10,VK5LA,QF05gq,1390,

Then after going ‘multi-band’ on Boxing Day :

2018-12-26 01:26,VK4ADC,50.294600,-20,FK1TS,RG37fr,1488,

2018-12-26 02:08,VK4ADC,14.097170,-16,VK3WHO,QF22rl,1295,

2018-12-26 02:10,VK4ADC,18.106186,-25,VK3KHZ/4,QF22pe,1330,

2018-12-26 02:16,VK4ADC,14.097169,-15,VK3WHO,QF22rl,1295,

2018-12-26 02:24,VK4ADC,14.097169,-16,VK3WHO,QF22rl,1295,

2018-12-26 02:28,VK4ADC,18.106185,-22,VK3KHZ/4,QF22pe,1330,

2018-12-26 02:36,VK4ADC,14.097168,-13,VK3WHO,QF22rl,1295,

2018-12-26 02:48,VK4ADC,14.097169,-13,VK3WHO,QF22rl,1295,

2018-12-26 02:52,VK4ADC,18.106186,-25,VK3KHZ/4,QF22pe,1330,

2018-12-26 03:00,VK4ADC,14.097169,-13,VK3WHO,QF22rl,1295,

2018-12-26 03:04,VK4ADC,18.106184,-27,VK3KHZ/4,QF22pe,1330,

2018-12-26 03:12,VK4ADC,14.097172,-12,VK3WHO,QF22rl,1295,

2018-12-26 03:18,VK4ADC,21.096196,-27,VK3KHZ/4,QF22pe,1330,

2018-12-26 03:24,VK4ADC,14.097167,-7,VK3WHO,QF22rl,1295,

2018-12-26 03:24,VK4ADC,14.097182,-19,VK2LX,QF44rs,897,

2018-12-26 03:30,VK4ADC,21.096198,-29,VK3KHZ/4,QF22pe,1330,

2018-12-26 03:34,VK4ADC,10.140248,-21,VK4XJB,QG52xh,99,

2018-12-26 03:36,VK4ADC,14.097166,-8,VK3WHO,QF22rl,1295,

2018-12-26 03:40,VK4ADC,18.106182,-24,VK3KHZ/4,QF22pe,1330,

2018-12-26 03:46,VK4ADC,10.140247,-20,VK4XJB,QG52xh,99,

2018-12-26 03:48,VK4ADC,14.097165,-26,VK3WHO,QF22rl,1295,

2018-12-26 03:54,VK4ADC,18.106181,-24,VK3KHZ/4,QF22pe,1330,

2018-12-26 03:58,VK4ADC,10.140249,-20,VK4XJB,QG52xh,99,

2018-12-26 04:00,VK4ADC,14.097167,-13,VK3WHO,QF22rl,1295,

There aren’t any 6M spots showing in the above apart from FK1TS at the beginning of the list but the 6M antenna was definitely connected! {There has been a lack of Sporadic E propagation over the last week or so and it seems the few local 6M operators weren’t running any 6M WSPR gear at the time either.}

The antenna used on HF at this time was a vertical whip based on a modified trap section from an old triband yagi but re-tuned for 10 and 18MHz. Operation at 14, 21 and 24MHz will be very inefficient as there are (currently) no resonant sections for those bands. The 6M J-Pole is the antenna currently used on 6M.

A couple of spots that did surprise me with my mere 200-odd mW of RF power were..

2018-12-26 08:34,VK4ADC,10.140251,-27,KPH,CM88mc,11390,

2018-12-26 08:46,VK4ADC,10.140251,-29,KPH,CM88mc,11390,

Grid CM88cc turns out to be near San Francisco and KPH would appear to be a former maritime coast station callsign and is typically activated for special events. https://en.wikipedia.org/wiki/KPH_(radio_station) (https://en.wikipedia.org/wiki/KPH_(radio_station))

NOTES:

The WSPR command line at the bottom of the daemon.sh file is gradually being adjusted to give final frequencies under my control, eg 50294280 10140150 14097000 50294270 18106000 14097010, so that the final frequencies are near – but not on top of - the centre frequencies. I did this after fitting a heatsink to the top of the Pi's CPU & over the 25MHz clock crystal oscillator and noting that the drift was now usually zero (0Hz), even on 6M, after the unit had temperature stabilised. I use the –f (free run) option as I can't always be sure that the Pi is accessing the WLAN 24/7.

I have currently removed the 21MHz entry until such time as I can create an alternative antenna that 'works' at this frequency then it may be re-instated..

The WSPR project has been housed in a plastic "jiffy box" (as shown above) with large-ish holes in each end for the various connectors (keyboard & HDMI one end, 6M and HF BNC connectors the other), a figure-8 for DC power.. and that is all to connect in or out. There are 4 LEDs on one end, along with a DC power switch, and the LEDs simply show which BPF is currently selected (10 / 14 / 18 / 50 MHz). Maybe eventually the pencilled labelling will be replaced by something more permanent..

It should continue to run for some time and be a useful propagation tool.

Addendum :

The original command line in daemon.sh was "sudo /home/pi/WsprryPi/wspr -r -f $CALLSIGN $QTH 23 .. and then the list of frequencies" and that is fine if you are runnin g the same power level on all of those frequencies.

My plans for just running 200mW is in a state of flux as I found a HF power amp in the junkbox - and then also a low-band VHF power amp that I had already retuned to 50MHz.

IF I get those going then the power level on each band will change so I needed an easy way to change the power indication in the WSPR setup.

The "a" part of the loop below is a bogey as "a" is never decremented so the loop runs forever.

The -x 1 part of the line says repeat once then exit where originally I used -r : repeat forever

The following lines replace the single WSPR command line (as detailed a few lines above) ...

```
a=10
until [ $a -lt 10 ]
do
sudo /home/pi/WsprryPi/wspr -x 1 -f $CALLSIGN $QTH 23 50294280  { eg 200mW}
sudo /home/pi/WsprryPi/wspr -x 1 -f $CALLSIGN $QTH 20 10140150  { eg 100mW }
sudo /home/pi/WsprryPi/wspr -x 1 -f $CALLSIGN $QTH 33 14097030  {  eg 2 watts }
sudo /home/pi/WsprryPi/wspr -x 1 -f $CALLSIGN $QTH 37 18106010  { eg 5 watts }
sudo /home/pi/WsprryPi/wspr -x 1 -f $CALLSIGN $QTH 40 21095990  { eg 10 watts }
done
```

Edit the daemon.sh file (eg with nano } then

sudo systemctl daemon-reload

sudo systemctl restart qrpi.service

sudo systemctl status qrpi.service

Obviously the text within the { } is not entered, they are just to indicate the different power options for each band.

Update : Thursday 17th January 2019 0200Z :-

The RF output powers from my Pi-WSPR beacon were raised from the project's original 200mW to 5 watts on 50.293 and 2 watts (+/-) on the 10, 14, 18 and 21MHz frequencies.

The power amplifiers introduced were..
(1) a modified & retuned (to 50MHz) 25 watt PA section from a low-band VHF FM828E, power reduced and variable, set to 5 watts out.. It can produce 25 watts out (tested and measured) so is really idling along at 5 watts out;
and
(2) a new homebrew 2 watt HF amp based on a 2N4427 driving into a 2N5589 (both NPN RF transistors) and the output derived after a 30MHz LPF section. The power efficiency fo this amp is relatively poor because inter-stage impedance match was eliminated to allow the amp to operate over the 10 to 21MHz range without requiring re-tuning adjustments. It draws 1 amp at 13.5V (ie 13.5W) to create the 2 watts of RF out.

The WSPR command lines in daemon.sh were edited to +37dBm for 6m and +33dBm for the HF band entries to relate to the new power levels, daemon-reloaded then qrpi.service restarted.

Within a hour of having the WSPR beacon plus its 2 new power amps back on air, the 6m signal was reported for the first time at over 200KM in the absence of sporadic E or any ducting condition, using the halo antenna so no antenna gain involved. First indications on HF were signal reports improved by about 10dB by some spotters, the same as the ratio of old to new power levels, (no so much with others) with new / different spotter callsigns appearing for the first time. It will be interesting to see what signal levels are reported late into the afternoon / evening when propagation on 10 and 14MHz both pick up.

Friday 18th January :

A sloping (60-70 degrees) wire antenna fed via a 9:1 UN-UN (VK6YSF design) was installed and seems to produce better SWR across the 4 bands. The wire is about 9-10 metres long, the top end (via an egg insulator) is up near the top of an ironbark gum tree on a halyard - about 10-12 metres above ground. The bottom end (via an egg insulator) terminates on the end of the shack building with the "ground" on the UN-UN going via a coax braid to a small bolt on the metal cladding. I may eventually put an earth stake into the ground if this arrangement stays.

Certainly reports have been coming in from worldwide due to the higher power level plus improved antenna arrangement. There seems to be periods of 'drop-outs' in th spots which are possibly the result of the software clock losing NTP sync as the WLAN signal is down at -84dBm (or worse) at the best of times. The drop-outs seem that the transmissions are not in the 2 minute WSPR window and therefore not decoded, with RF still going out. The clock re-syncs and then the reports come back ! To be resolved..