



President's Corner

2021 ARRL and TAPR Digital Communications Conference

If all goes well COVID 19-wise, we will hold the 2021 ARRL and TAPR Digital Communications Conference (DCC) in-person in Charlotte, North Carolina on September 17-19, 2021. However, if COVID 19 is still a problem and in-person conferences are out of the question, we will hold a virtual DCC as we did in 2020.



This is a developing story, so keep up to date by visiting the 2021 DCC webpage at

<https://tapr.org/digital-communications-conference-dcc/>

Normally at this time of year, we would be preparing for Hamvention, but the folks running the Big Show cancelled this year's edition because of the darn virus. Hopefully, there will be a Hamvention in 2022 and TAPR will be there.

73,

Steve Bible, N7HPR

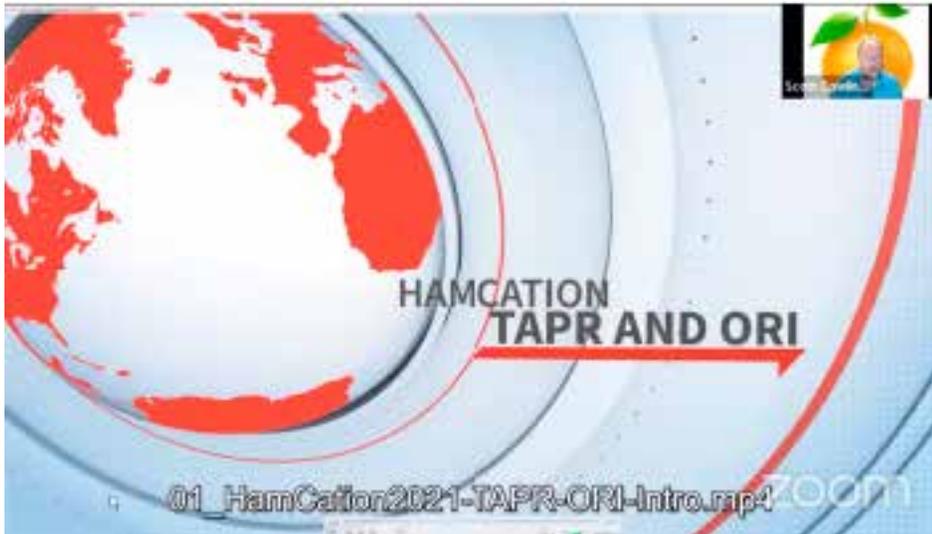
TAPR President

###

2021 ARRL & TAPR DCC	01
TAPR at Hamcation	02
TAPR Wear Available	02
Processing I/Q Data from a FlexRadio into Gnuradio	03
Donate to TAPR	14
What's in Store for You at TAPR	15
More Interested in Building Sailboats Lately...	18
Serbian Signals	19
Being a BBS/NODE SysOp is More than Writing Bulletins	20
Write Here!	23
On the Net	23
The Fine Print	24
Our Membership App	25

TAPR at Hamcation

Florida's annual Hamcation was a virtual event this year, but TAPR was present with a technical forum led by Michelle Thompson, W5NYV, and Scotty Cowling, WA2DFI. You can view the forum, titled "TAPR and ORI" (Open Research Institute) on YouTube at <https://tinyurl.com/yy3jdy3f>



###

TAPR Wear Available



Personalized Land's End clothing with the TAPR logo and your name and call sign are now available from the TAPR Store at <http://business.landsend.com/store/tapr/>

Select from the Men's or Women's catalog. (To make shopping easier, there are "TAPR Recommended Shirts" in the Men's catalog including two styles of polo shirts, each available with or without pockets.)

The logo is available in three colors -- red, blue and white. The name/call sign monogram thread will match the logo color. (We recommend that you use the white logo with dark colored shirts.)

Prices are very reasonable, for example, after adding the logo and monogram, a mesh pocket shirt is \$39.85 plus shipping and sales tax where applicable. Processing time is 5-7 days.

###

Processing I/Q Data from a FlexRadio into Gnuradio

An intermediate to advanced level project

By William D. Engelke, AB4EJ, University of Alabama

If you want to learn about DSP (digital signal processing - in particular, working with the data coming from a Software Defined Radio, or SDR), the first thing you need to understand is the concept of I/Q (“analytic”) data. There are many good tutorials about this topic (for example, see:

<http://whiteboard.ping.se/SDR/IQ>), so we will not repeat the explanations here, but assume that you know enough to be interested in collecting I/Q data from your FlexRadio so you can experiment with doing some DSP on your own. This article focuses on radios like the FlexRadio 6400, 6600, 6700 (plus their M versions with front panel) that can be used with the SmartSDR software package. Several other SDRs can be made to output I/Q data in a similar format.

What you need:

- A FlexRadio® such as the 6400, 6400M, 6600, 6600M, 6700
- A computer running MS Windows®, with SmartSDR® installed and running
- A computer running Linux (e.g., Raspberry Pi 4 or Odroid N2) (I use Ubuntu)
- A network interconnection (such as an Ethernet switch)

connecting the radio and both computers

Why two computers? The FlexRadio will output I/Q data out of the box when paired with the SmartSDR software, which is MS-Windows based. It is fairly straightforward to get the Raspberry Pi to talk to the Flex system, i.e., radio and SmartSDR. (If you are a very accomplished programmer, you might be able to get the Raspberry Pi to work directly with the radio, i.e., without SmartSDR; but this is a significant project and beyond the scope of this article).

Let’s say you have a Raspberry Pi 4, Odroid N2, or similar small computer running Ubuntu, or similar O/S, and would like to get some I/Q data from your FlexRadio to store, process, or pull into DSP using Gnuradio. Here is one way to do it...

STEP 1. Set up SmartSDR (it should be running on the Windows- based computer and connected to the radio). Using the menu Settings -> Radio Setup... Set up a Static IP Address and make note of it; in this case I have set up 192.168.1.66 as the FlexRadio’s local IP address (see Figure 1).

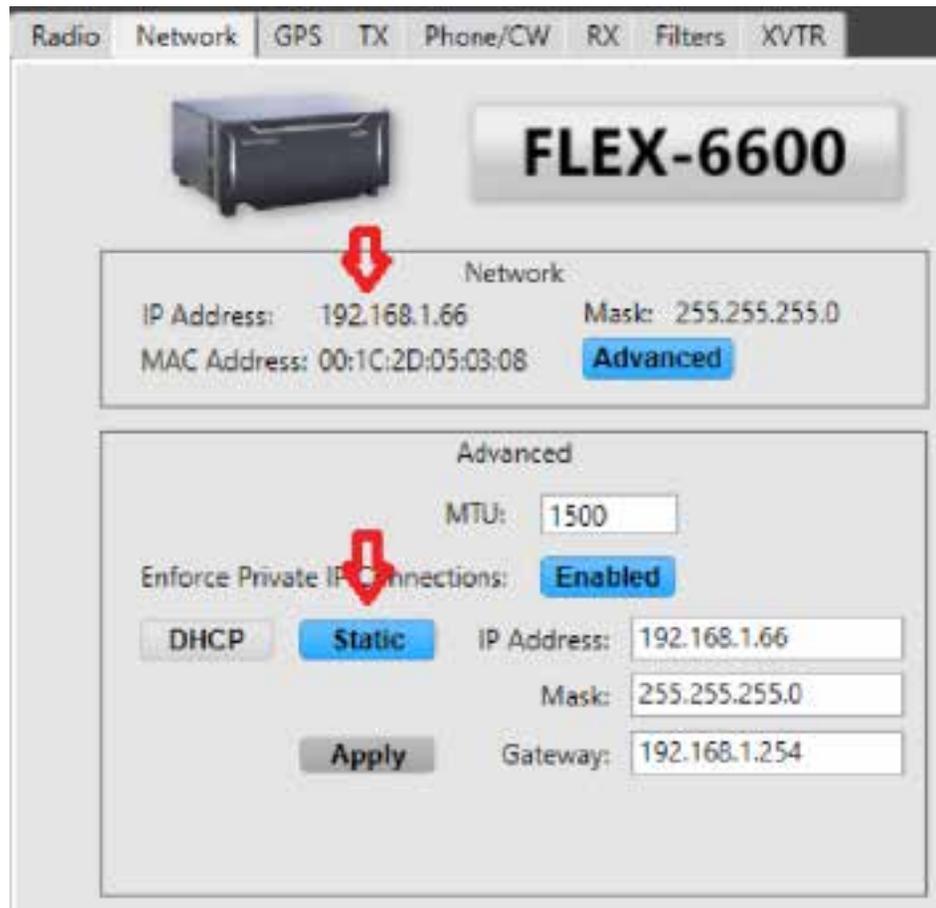


Figure 1. Setting up the FlexRadio to use a static IP address.

STEP 2. Set up SmartSDR so that the FlexRadio is receiving on Slice A, and activate DAXIQ Channel 1 (see Figure 2).

Notes about this:

- This method does not involve turning on IQ Streams in the DAX Control Panel. That feature of DAX does not provide any easy way to direct the data to a given destination.



Figure 2. Designating the DAXIQ channel.

- The DAXIQ channel setting for a given slice is independent of the DAX channel setting for that slice, as we see in the figure above.
- TIP: Having said all that about the independence of these settings, if you find that a channel will not send IQ data when commanded, sometimes the solution is to turn the IQ Stream for that channel on and off using the DAX Control Panel, which seems to kick something loose in the software.

STEP 3. You need a program that can command the radio to start sending I/Q data from a running slice.

Download the source code from:

https://github.com/AB4EJ-1/TangerineSDR-pi/blob/master/flexComm/flexIQdata_one.c

and save it in a directory on your Raspberry Pi.

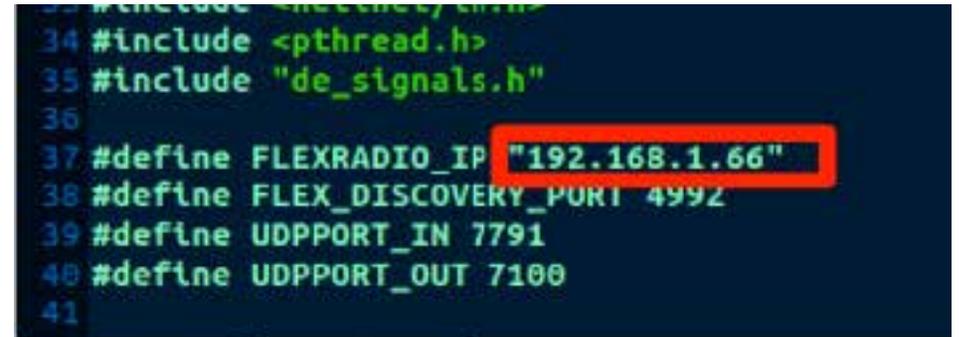
Edit the source code of the c program *flexIQdata_one.c* to put the IP address (from STEP 1) of your Flex into the define line for *FLEXRADIO_IP* (see Figure 3).

Now the c program knows how to connect to the Flex radio. Note that the Flex discovery port is fixed at 4992.

Compile the program using:

```
gcc flexIQdata_one.c -o flexIQdata1 -pthread
```

(Note that you need to have the pthread c library installed on



```

34 #include <pthread.h>
35 #include "de_signals.h"
36
37 #define FLEXRADIO_IP "192.168.1.66"
38 #define FLEX_DISCOVERY_PORT 4992
39 #define UDPPORT_IN 7791
40 #define UDPPORT_OUT 7100
41

```

Figure 3. Entering IP address of FlexRadio into the *flexIQdata1* program.

the Raspberry Pi for this compile; many Linux distributions include this pre-installed).

Now you have an executable (*flexIQdata1*) that will activate DAX IQ channel 1, producing a UDP stream of IQ data, which it will send to the Raspberry Pi at UDP port 7790. (Port 7790 is rather arbitrary, but lives within the normal Linux restrictions on what ports can be used for what). Note that this program lets the FlexRadio IQ output run at its default sample rate of 48 ksps.

STEP 4. Testing

To see this work, start a terminal on your Raspberry Pi and use the *tcpdump* utility to monitor port 7790:

```
sudo tcpdump -i eth0 udp port 7790
```

Ensure the Flex is running with a slice active and receiving. Now, on a separate terminal window on the Linux system, navigate to the directory where you have compiled the flexIQdata1 executable, and run it:

```
./flexIQdata1
```

If flexIQdata1 is talking to the FlexRadio like it should, you will see a series of messages appear in the terminal window you are using; these are status messages from the Flex, for example, see Figure 4.

```
S5FB9017F|radio slices=3 panadapters=3 lineout_gain=6 lineout_mute=0 headphone_
ain=32 headphone_mute=0 remote_on_enabled=0 pll_done=0 freq_error_ppb=-4 cal_fr
q=10.000000 tnf_enabled=0 snap_tune_enabled=1 nickname=Bill callsign=AB4EJ bina
ral_rx=0 full_duplex_enabled=1 band_persistence_enabled=1 rtty_mark_default=212
enforce_private_ip_connections=1 backlight=50 mute_local_audl
-----
|radio filter_sharpness VOICE level=2 auto_level=1
S5FB9017F|radio filter_sharpness CW level=2 auto_level=1
S5FB9017F|radio filter_sharpness DIGITAL level=2 auto_level=1
S5FB9017F|radio static_net_params ip=192.168.1.66 gateway=192.168.1.254 netmask
255.255.255.0
S5FB9017F|radio oscillator state=external setting=auto locked=1 ext_present=1 g
sdo_present=0 tcxo_present=1
S5FB9017F|interlock acc_txreq_enable=0 rca_txreq_enable=0 acc_tx_ena
-----
lock tx_client_handle=0x00000000 state=RECEIVE reason= source= tx_allowed=0 ampl
fier=
-----
```

Figure 4. Initial output of flexIQdata1, showing FlexRadio response.

When the messages stop, key in 1 followed by Enter. This tells the flexIQdata1 program to command the FlexRadio to activate DAXIQ channel 1 and send data to the IP address of the Raspberry, at the UDP port 7790. Additional messages should come out similar to the following (see Figure 5).

```
borfido@borfido: ~/projects/rangermesdk-pi/flexComm
File Edit View Search Terminal Help
1
user entered 49
sent c0|client udpport 7790
-----
-----
sent c1|stream create daxiq=1 port=7790
-----
|stream 0x20000000 type=dax_iq daxiq_channel=1 pan=0x40000000 daxiq_rate=48000
lient_handle=0xA2FCA298 active=1 ip=192.168.1.207
SA2FCA298|radio daxiq_capacity=16 daxiq_available=14
R1|0|20000000
-----
1 - start Flex data
2 - stop Flex data
user entered 10
1 - start Flex data
2 - stop Flex data
```

Note the FlexRadio stream ID of 2000 0000 here.

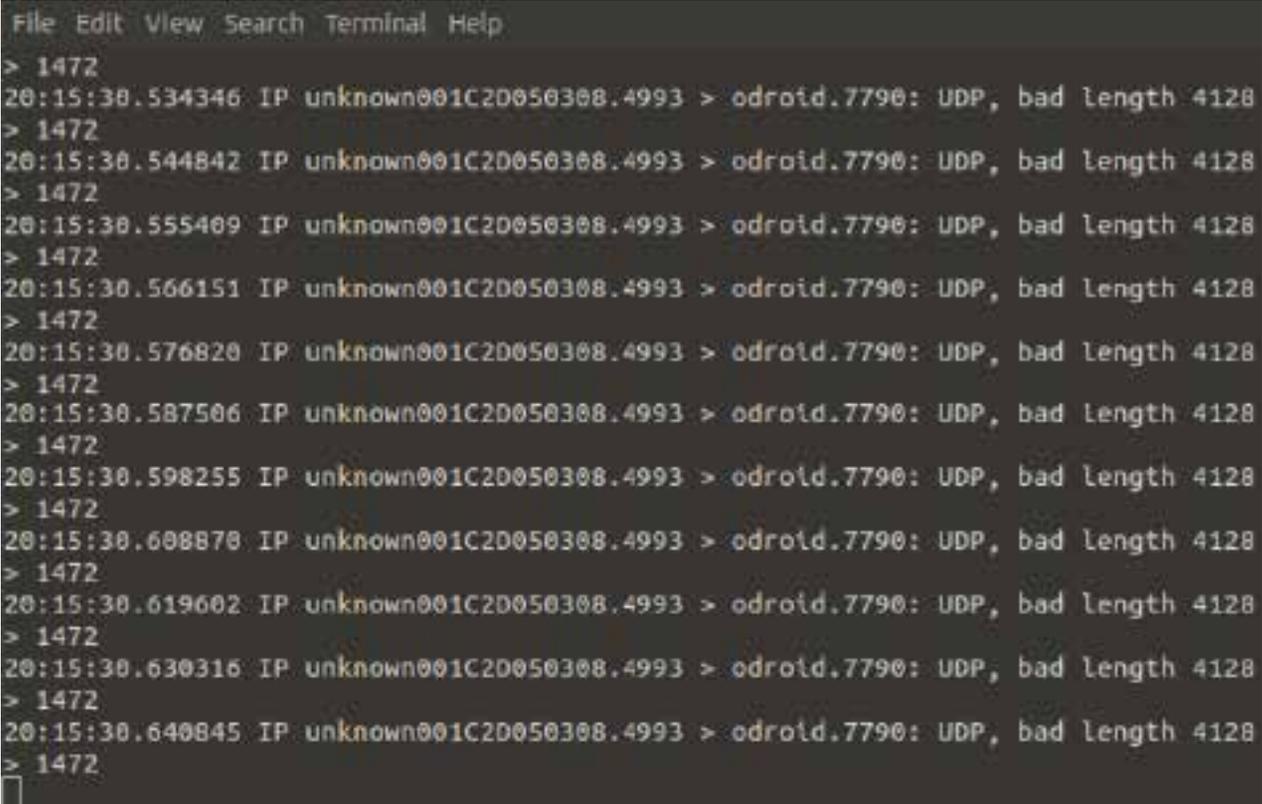
Figure 5. flexIQdata1 waiting for additional user commands (1 or 2)

This output isn't the most user-friendly, and demonstrates the highly asynchronous nature of this system; a running thread waits for the Flex Radio to send data and displays it, without cleanly synchronizing with user input (this is version zero, as you can tell). You should see the FlexRadio stream ID of 0x20000000 here.

If everything is working, you should see a continuous series of messages appear on the terminal that is running tcpdump (see Figure 6).

What this is telling you is:

- The FlexRadio is outputting data from its UDP port 4993, routing the output to the Raspberry Pi (or Odroid in this case) at UDP port 7790.
- The “bad length” of 4128 doesn't indicate an error; it simply means that the packet size is greater than the standard IP frame length limit of 1472; but these packets (built from multiple frames) can theoretically be up to 64 k bytes in length (the normal I/Q data packet length from the FlexRadio is 4128).

A terminal window with a dark background and light text. The window title is "Terminal". The output shows a series of lines from a tcpdump command. Each line starts with a timestamp (e.g., 20:15:30.534346), followed by "IP unknown001C2D050308.4993 > odroid.7790: UDP, bad length 4128". The lines are separated by "> 1472" on a new line, indicating the standard IP frame length limit. The pattern repeats for approximately 12 lines, showing a continuous stream of data packets that exceed the standard frame size.

```
File Edit View Search Terminal Help
> 1472
20:15:30.534346 IP unknown001C2D050308.4993 > odroid.7790: UDP, bad length 4128
> 1472
20:15:30.544842 IP unknown001C2D050308.4993 > odroid.7790: UDP, bad length 4128
> 1472
20:15:30.555409 IP unknown001C2D050308.4993 > odroid.7790: UDP, bad length 4128
> 1472
20:15:30.566151 IP unknown001C2D050308.4993 > odroid.7790: UDP, bad length 4128
> 1472
20:15:30.576820 IP unknown001C2D050308.4993 > odroid.7790: UDP, bad length 4128
> 1472
20:15:30.587506 IP unknown001C2D050308.4993 > odroid.7790: UDP, bad length 4128
> 1472
20:15:30.598255 IP unknown001C2D050308.4993 > odroid.7790: UDP, bad length 4128
> 1472
20:15:30.608870 IP unknown001C2D050308.4993 > odroid.7790: UDP, bad length 4128
> 1472
20:15:30.619602 IP unknown001C2D050308.4993 > odroid.7790: UDP, bad length 4128
> 1472
20:15:30.630316 IP unknown001C2D050308.4993 > odroid.7790: UDP, bad length 4128
> 1472
20:15:30.640845 IP unknown001C2D050308.4993 > odroid.7790: UDP, bad length 4128
> 1472
```

Figure 6. tcpdump output showing FlexRadio port 4993 sending data to port 7790.

You can command the FlexRadio to stop sending data by entering a 2 into flexIQdata1. TIP: you can stop the flexIQdata1 program with CTRL-C, but the ports will remain committed. Use killall -9 flexIQdata1 to completely clean up after the program, otherwise you may get a “port in use” error the next time you try to run it.

Working with Data Content

Getting the Flex to send I/Q data, while it can be interesting and challenging in itself, is only half the battle. Next, you have to be able to decode the packets. The FlexRadio packet layout adheres closely to the VITA-49 standard. There are many online articles about this standard, but most of them talk in general terms about the standard without detailing what the data actually looks like. This reference goes into the details:

<https://www.redrapids.com/images/whitepapers/TWP-000-001-R00.pdf>

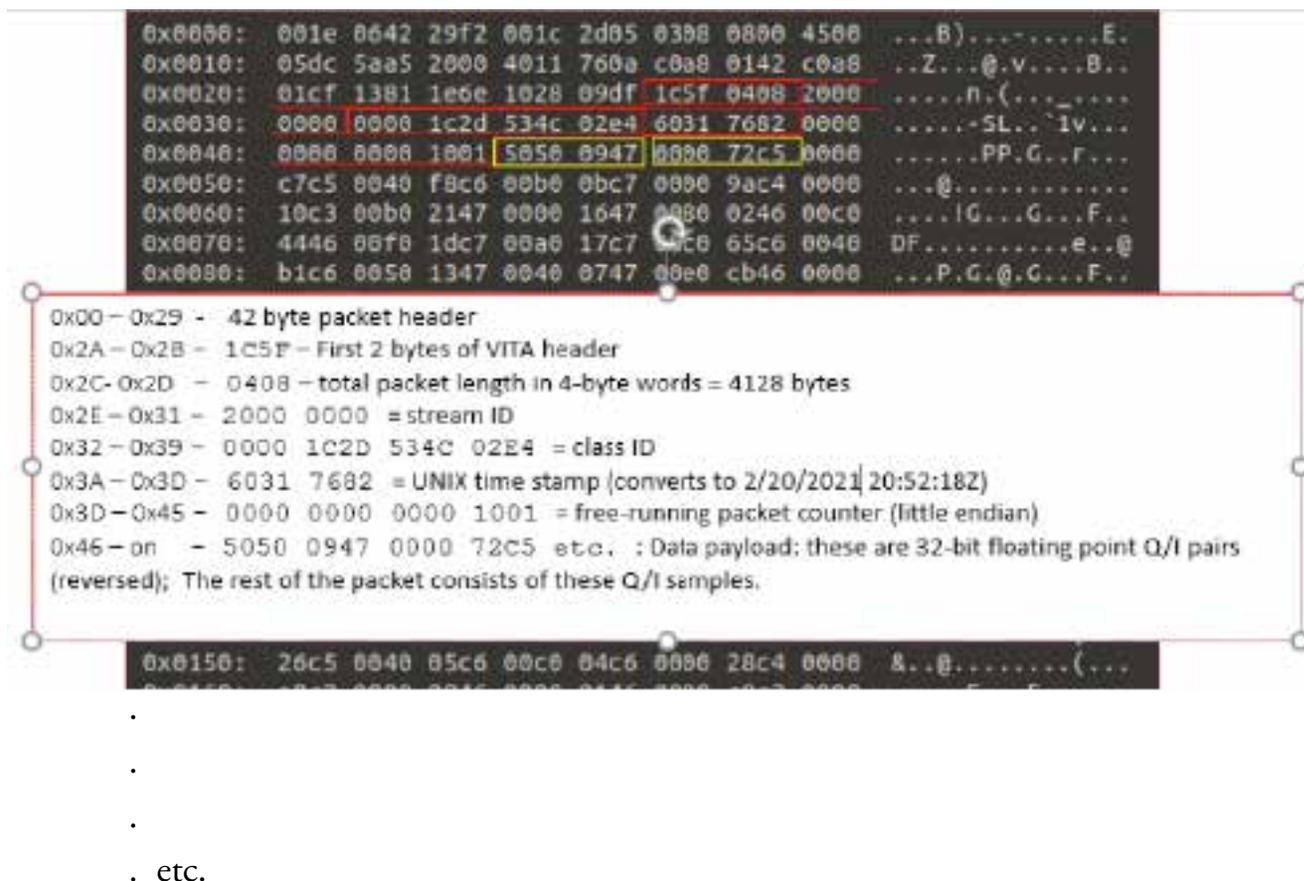


Figure 7. Decoding packets from the FlexRadio

Let's take a look at a data packet from the Flex (see Figure 7). The first 42 bytes are the standard IP packet header. Next come:

- a 2-byte VITA header (here 0x1C5F)- this is defined in the VITA-49 standard
- a 2-byte packet length (in 4-byte words), so 0x0408 = 4128 in decimal; you will see this is the packet length in tcpdump.
- a 4-byte stream ID. This matches the Flex Radio's stream ID# (0x2000 0000)
- 8 bytes of class ID. (If you know how to decode this, please let me know.)
- a 4-byte UNIX time stamp (i.e., number of seconds since January 1, 1970).
- a free-running packet counter, in little endian format (VITA-49 also allows this to be fractional seconds).

Then comes the data payload, which is pairs of 32-byte floating point numbers, in Q/I order. (If you treat this as I/Q order, you will see that frequency relationships are reversed. (This Q/I order is apparently to maintain compatibility with the TAPR HPSDR, which used this convention). Each packet contains 512 Q/I samples (each sample being two 32-bit floating point numbers).

Here, from the flexIQdata1 program, is the structure of this data in c:

```
struct flexDataSample
{
    float I_val_int;
    float Q_val_int;
};

typedef struct flexDataBuf
{
    char VITA_hdr1[2]; // rightmost 4 bits is a
    packet counter
    int16_t VITA_packet_size;
    char stream_ID[4];
    char class_ID[8];
    uint32_t time_stamp;
    uint64_t fractional_seconds;
    struct flexDataSample flexDatSample[512];
} FLEXBUF;
```

Using the Data

Now we have the FlexRadio outputting VITA-49 data packets on UDP port 7790. What can we do with this? We can route it to Gnuradio and do digital signal processing! For simplicity, we will display signals on a waterfall. This is redundant with SmartSDR, of course, but this lets us see that our output is right, and if you want to do other serious DSP you can do this in Gnuradio by adding in and connecting up the blocks. Where to start?

First, you need to have a recent version of Gnuradio installed, which includes the latest UDP Source block. You can find installation scripts for Gnuradio for the Raspberry Pi and the Odroid at N4XWE's github page here:

<https://github.com/n4xwe> (Note: installing gnuradio may take quite a bit of patience.)

Next, we have to set up an intermediate program to strip off the VITA-49 headers and re-send the packets to gnuradio (as of this writing, the Gnuradio UDP source block does not yet directly support the VITA-49 header). Download the following program from github:

<https://github.com/AB4EJ-1/TangerineSDR-pi/blob/master/iqproject/rgnu.c>

Edit the file, putting the IP address of your Linux machine (where you have Gnuradio installed) into the definition of DESTINATION .

Compile this using:

```
gcc rgnu.c -o rgnu
```

Now, on your Linux box, you need to have two terminals running. On the first one, start flexIQdata1 and enter a 1 to start the data feed; on the second terminal, start rgnu using command `./rgnu` . This will start rgnu running - it receives VITA-49 packets on UDP port 7790, removes the VITA header and resends the packets to the designated IP address on UDP port 8790. You can use tcpdump to look at the data coming from the FlexRadio, but remember that tcpdump does not show packets when a program is routing data to another port on the same IP address, so you may not be able to observe the packets going to port 8790.

Here is a gnuradio flowgraph which shows hooking up the UDP source block to receive and use the packets coming from the rgnu program. (Find this at: https://github.com/AB4EJ-1/TangerineSDR-pi/blob/master/iqproject/UDP_waterfall.grc)

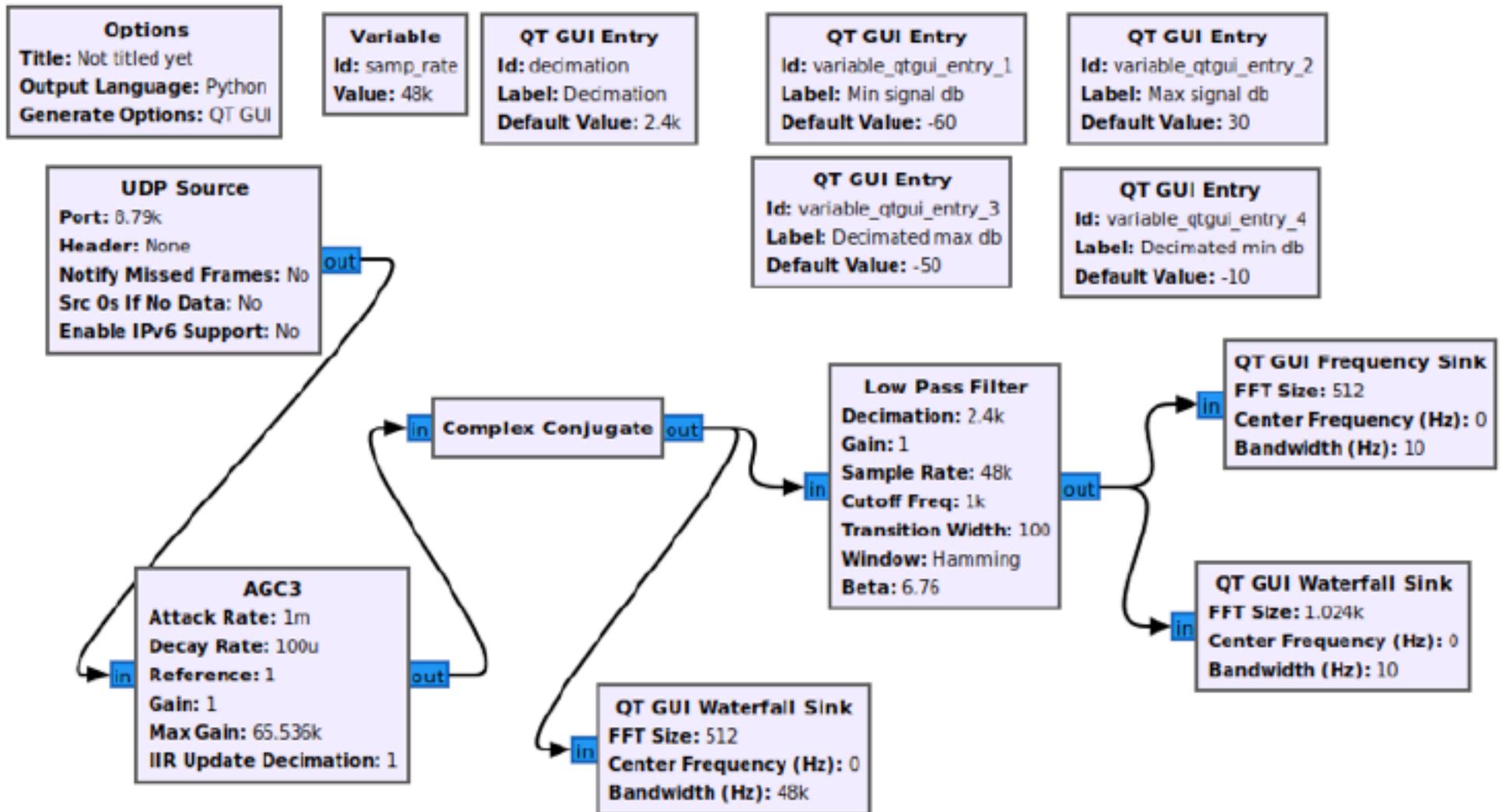


Figure 8. Gnuradio flowgraph for inbound and processing packets from rgnu

About the blocks, from left to right:

- UDP Source – receives the data packets sent by rgnu and routes them into Gnuradio.
- AGC3 – adjusts gain so that you can monitor a signal source that varies in strength due to diurnal variations, such as WWV at 5 MHz.
- Complex Conjugate: reverses the Q/I data pairs to I/Q for correct frequency relationship
- Low Pass filter – decimates the 48,000 samples/sec down to 20 samples/sec, removing any noise-creating higher frequencies. This allows us to observe tiny frequency shifts.
- The two QT GUI Waterfall Sink blocks provide a waterfall output. Here, we are showing how movement of the ionosphere causes a Doppler shift in the frequency of WWV as the sun goes down. (This assumes your FlexRadio uses a GPS Disciplined Oscillator as its 10 MHz master clock). This technique is so sensitive that you can even observe WWV Doppler shifts caused by solar flares, transient ionospheric disturbances, and the polar vortex. Figure 9 shows an example waterfall produced by the QT GUI Waterfall Sink on the lower right of the flowgraph. This particular one is from a Grape monitoring 5 MHz; I chose it here due to its dramatic illustration of sunrise effect on WWV frequency. You can produce similar waterfalls using

the FlexRadio, but note that the 10 MHz WWV signal can be masked by bleed-over from the 10 MHz clock in the FlexRadio, so try 5 MHz.

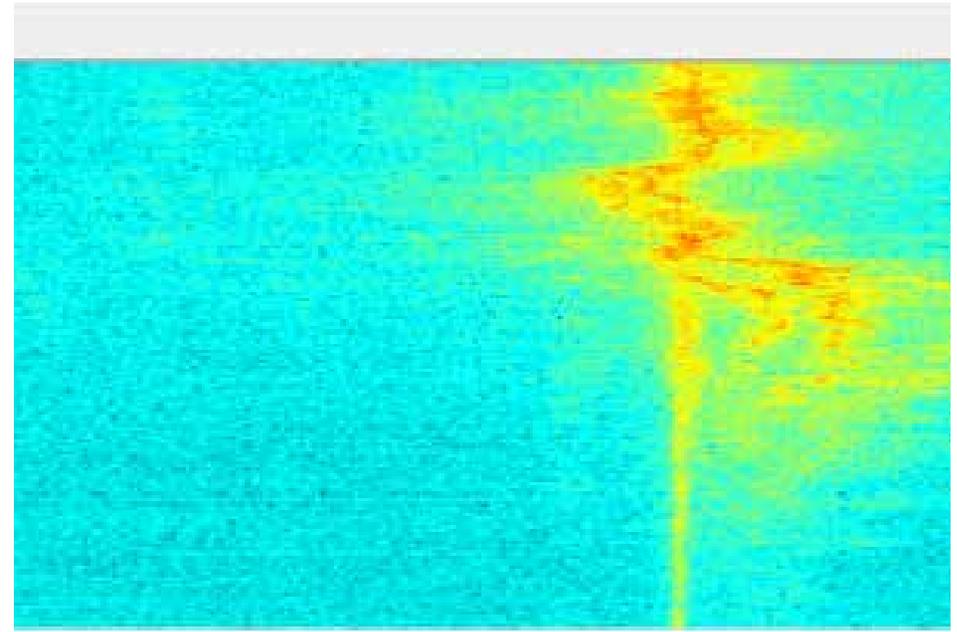


Figure 9. Observing 5 MHz WWV Doppler shift at sunrise (Feb. 26, 2021).

This final step, where we zoom into the 1 to 3 Hz region around the WWV carrier is something that the Flex does not directly support in SmartSDR, as it does not let you zoom in that tightly, but we can see that the Doppler shift is definitely there!

You can listen to the signal by adding an Audio Sink and plugging headphones to the Raspberry Pi or Odroid. You can also save the signal to files and later work with the files.

This type of frequency analysis and observation of Doppler shifts in WWV is fundamental to the HamSCI Tangerine SDR project, for which TAPR is building the hardware. (A low cost version of that system, called the “Grape” is being developed by Case Western Reserve University.) If you are interested in doing your own Space Weather observations, keep an eye out for further developments on these projects, and plan to join the team for the HamSCI virtual workshop March 19-20, 2021. See <https://www.hamsci.org/hamsci2021>

About the Author

William D. Engelke (AB4EJ) is a Research Engineer with the University of Alabama (Tuscaloosa, AL), and is heading up the software development for the TAPR TangerineSDR-based Personal Space Weather Station. Engelke was first licensed in 1965 and holds a BS in Electrical Engineering from Virginia Tech, and a MSISE from the University of Alabama in Huntsville. See qrz.com for more details.

###

Donate to TAPR

TAPR is now participating in the AmazonSmile program!

When you shop using the AmazonSmile program, Amazon makes a donation to TAPR equal to 0.5% of the price of your eligible AmazonSmile purchases.

AmazonSmile is the same Amazon you know. Same products, same prices, same service.

Bookmark the TAPR AmazonSmile Program link:

<https://smile.amazon.com/ch/86-0455870>

That link takes you to a special login portal where you enter your normal Amazon credentials and get redirected at the same Amazon home page except there will now be a notice that you are supporting TAPR.

Other ways to donate to TAPR, email contact@tapr.org:

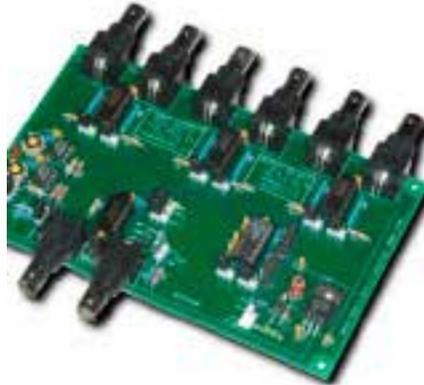
###

What's in Store for You at TAPR

Here is a summary of what you will currently find for sale at TAPR's online store (<https://tapr.org/shop/>). Don't forget, TAPR members receive a discount on many of the items on sale.



TAPR Membership – \$30



TADD-3 Pulse-Per-Second Distribution Amplifier – \$89



TADD-1 RF Distribution Amplifier – \$130



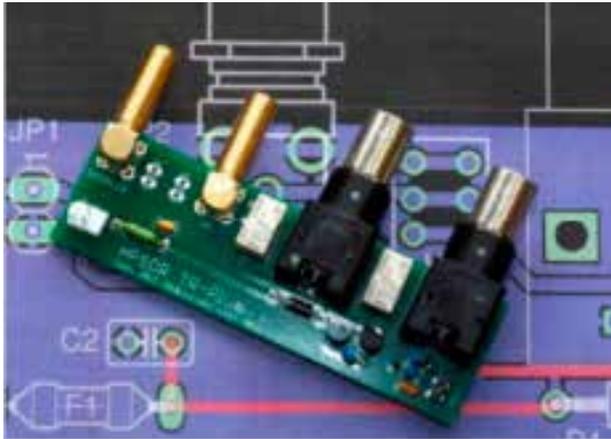
TAPR TICC – \$229



TADD-2 Mini Pulse-Per-Second Divider – \$62



TADD-2 Pulse-Per-Second Divider – \$89



TR-Plus Transmit/Receive Switch for Pure Signal – \$54



FatPPS Pulse-Stretcher – \$54



HPSDR Boards (Sale!!!) – \$33 to \$253



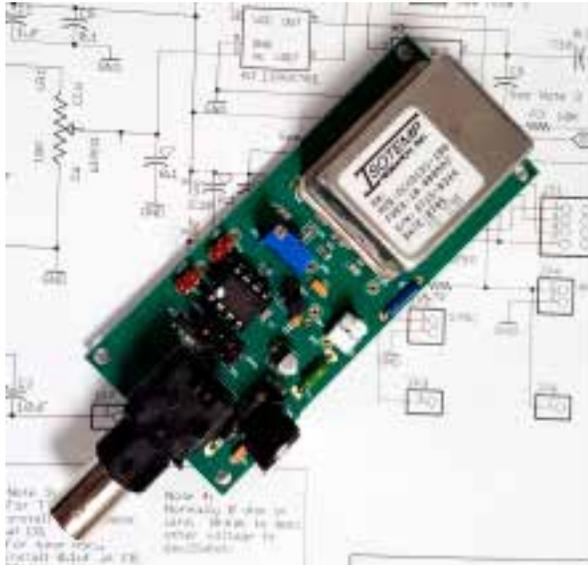
TAPR Personalized Badge – \$19



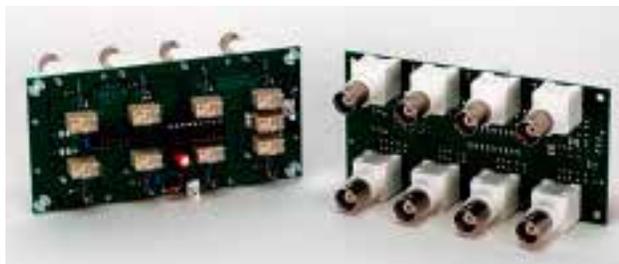
TNS-BUF Isolation Amplifier – \$122



WSPR – \$32



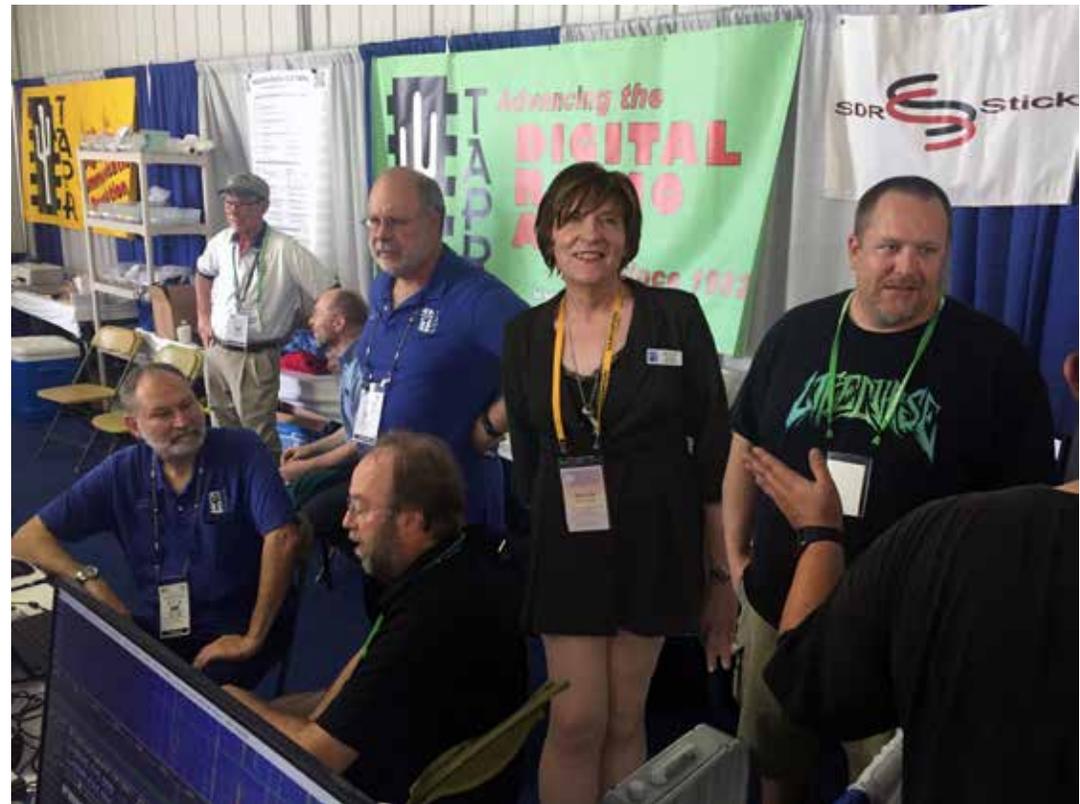
Pulse Puppy –\$79 to \$109



TASS RF Switch System – \$24 to \$129

###

TAPR Booth at Hamvention 2019



See You in 2022!

More Interested in Building Sailboats Lately (but make sure to check out WinRPR and VARA software modems)

By Maiko Langelaar, VE4KLM

It's hard to believe it's been one year since we have not been able to practice the hand to hand aspect of martial arts. It was nice enough to be able to do weapons last summer, but due to a serious illness in our household, that too came to an end last fall. In our household, we have to be especially careful what we bring home, one vascular disease is enough, we don't need COVID showing up, as it has for so many people, what a terrible year.

This past December, I started the design and construction of a model of a variant of the Nordic folk boat - from scratch. The idea is to build a full size sail boat before I kick the bucket. That's all I have been wanting to do lately. It will take years, but that's okay.

I worry about xNOS turning into just another internet app. It's not the first time, seems to surface every so often, and I find it discouraging at times. There hasn't been a local packet radio community around here for 20 years, and after having lost most of my equipment to aging and damage years ago, I find it hard to justify buying new equipment. You have to understand that the kicks you get out of doing packet radio with yourself run out after some time. The problem is that I enjoy the software



development aspect, but there is a danger of it becoming just another app, and one can easily lose perspective on the radio side of things.

So from time to time, I feel the need to get the radio back into amateur radio with regard to xNOS, forcing me to turn to HF once again to help get back on track. A few months ago, two amateur radio colleagues pushed me to develop drivers for a couple of HF software modems I had no idea existed.

Serbian Signals

By Misko Skoric, YT7MPB

Ham radio club “Novi Sad” (YU7BPQ) in Novi Sad, Serbia, loc. JN95wf, is active twice a week (Tuesdays and Thursdays afternoon local time) in VHF packet, HF packet (both ‘classic’ 300bd and robust), and HF pactor mode. Op. Misko (YT7MPB) brings in his own modems and sound-card interfaces to exchange mails by using club’s radios & antennas. Even though it is not 24/7 operation, the club welcomes new packet and pactor contacts on 20m and 40m.

Ham radio union SRV (YU7APV) in Novi Sad, Serbia, loc. JN95wf, has an email server (packet radio) installed. It works in test mode until VHF radio and antenna become available. The new BBS (“new” after some 25+ years of inactivity) has telnet forwarding with several station around the world, and is planned to serve local ham community in Novi Sad, the capital of Vojvodina province in northern Serbia.

###

SCS WinRPR Robust Packet Radio software modem
EA5HVK VARA HF and FM software modems

These are very cool modes. Don’t get me wrong, I love hardware modems, and as much as I love watching the PTC-IIusb light up like a Christmas tree, these software modems are quite amazing, so give them a try. A quick online search will take you there. There is a decent community online to support both RPR and VARA, each with their own dedicated mailing lists. We should really acknowledge the Virginia Packet Network (vapn.org), for without their help and infrastructure, I would not be doing RPR today.

I just wish solar conditions were not so dismal right now ...

###

Being a BBS/NODE SysOp is More than Writing Bulletins

By Brian Rogers, NIURO

I've been a BBS/Node System Operator since the early 90s on a variety of operating systems from DOS to OS/2 to Linux, using a multitude of software for different hardware devices and various mail systems communicating via various transports. It seems as hams however we've hit a crossroads where the skills are being lost and the hand-off of knowledge is not being well received. Worse yet, the energy used into the excuses would be better spent into mastering the skill of one area than to try to bite off more than one can chew and really run a muck of things – on a production network no less.

As a Linux partner developer many come to me for assistance with devices such as their Rpi Linux or their various distros of desktop Linux. This is all well and good however STEP 1 when wanting to be a node/bbs SysOp is to be a system administrator first and foremost. If you're not, you're only going to be asking for serious trouble in the near future. This includes but doesn't limit you to learning:

- basic shell commands
- routing (including policy routing ipv4 AND one should learn IPv6)
- firewalling

- installing/upgrading apps
- upgrading your OS

If you can't accomplish these 5 things, you honestly should not be running a bbs/node until you do and learning is a major part of why we enjoy this hobby. This leaves you vulnerable to be in the hands of someone else and even then it's YOUR license not their's that's at risk. If your outsourced admin lets a message go through with an illegal cuss word, it's YOU could be taking a permanent vacation so why put yourself at risk because you put the cart well before the horse? You studied too hard to pass your license exams to allow this to occur I would hope. Research and learn at least the first 4 of the 5 listed skills. Focus on those first until you feel comfortable in strolling around inside your own drive.

Once you are, investigate how to proceed with a software or system you may be interested in. Ask some questions of those who you know run the system(s) you may be interested in running yourself – get pros and cons and if they tell you there are no cons then they're not being honest. Not every system can do everything you may wish it to do. Decide what services you wish to offer those in your area and plan accordingly. Once you scope out what you wish to do and perhaps the software(s)

you would like to use, search the web for support lists, copies of documents and manual (man) pages to see if you might ahead of time find answers to some questions you may have. If not join a list and post questions. Most lists archive their mails and your question may help someone else in the future.

When you end up getting a working system that's productive "on the air", the first thing you'll want to do is grab a full backup! Nothing can be more frustrating while you're learning than to make a critical mistake and not know how to fix it. With a backup you can restore an existing working system in a matter of minutes. I suggest the use of an external drive such as a Passport USB drive and CloneZilla which is freeware. CloneZilla will grab the master boot sector and the entire drive for you. Also if you have a main drive failure you'll be happy to know that you have your data safely stored where you can restore it after getting a new drive to restore it on. I would suggest rerunning your backup at least monthly. For a packet station mail that may be lost would be kept fairly small.

Please... PLEASE... learn firewalling for your OS! Don't rely on just a simple script to handle things for you because most canned firewalls know nothing of the protocols most hams use. Learn how to write your own if possible. ALL traffic that flows through your system is your responsibility... no one else's. Firewalls come in many forms such as an external hardware device from

a company such as Barracuda Networks, to something so basic as a Message Filter (m_filter) system. Whether someone uses your system to launch attacks against others or floods you with unwanted content that you may possibly in turn spew out the amateur radio packet network.

Watch your message filter plug-in config as well. Insure it's grabbing what you need it to hold for your review, kill things you want it to kill, and let things pass that you wish it to pass. If it's not then please adjust the config file for it.

You also want to insure you don't archive bulletins on the live global network either. If you desire to archive mail, do so OFFLINE on a web or FTP server. Doing so puts you at risk for making your partners look bad in regards to duping old mails. The average age of public bulletin mail is 14 days with 30 days acceptable. Anything more than 60 days is overkill and most likely any mails that need to be reread will be searched via the web on a SIG, or reasked on the bulletin network. Remember this is part of being a system operator AND practicing good system administration skills.

Set up a lab environment for your "experimenting". Don't do this on a public working network. It shows your forwarding partners you have little regard for them, and it breaks the flow of mail for your end users. How long would you continue to use

gmail or hotmail IF google and microsoft “experimented” with alpha and beta configs on the global internet? If you found you were losing mail OR your important emails were not getting out, you’d be finding a new provider pretty darn quick. Lately I hear of many guys wanting to use the production packet network as a platform to “experiment” on. Experimenting is the basis of our hobby however when you put mail and possible ecomm mail at risk, then you’re already causing more harm than good. Here’s where Rpi units would really shine! Raspberry Pi units are test cards, so use them offline to make a test environment with. They’re inexpensive, and you can use a cross-over cat5 cable to link them with if you need to.

Keep your forwarding partners down to as much of a minimum as possible. This doesn’t mean that you’re shunning anyone, it means you’re being responsible and not trying to receive the same bulletin a dozen or so times because all your partners will be trying to send it to you too. This sucks up bandwidth for no real reason while at the same time you’re honoring the flow of the design of heirarchical addressing. If we all respect the original design our founding fathers gave us, you could forward with a small handfull and your mail can still go world wide.

Find a good elmer! One that’s been doing this stuff for 15-30 years. They are out there. If you can find one that writes or contributes source code to the system you’re investigating and

they have the time to assist you that’s even better and please do your best to appreciate them. The more elmers are lost, the worse the system becomes so best of luck, happy studying, and enjoy your new BBS/Node!

73

Note: The author is a 25+ year node/bbs sysop, Linux partner developer, and contributes code to the JNOS2 and LinFBB projects along with maintaining his own projects.



Write Here!

Your *PSR* editor is working on the next issue of *PSR* and hopes to find a few good writers, particularly ham radio operators working on the digital side of our hobby, who would like to write about their activities and have them published here in *PSR*.



You don't have to be Hiram Percy Maxim to contribute to *PSR* and you don't have to use *Microsoft Word* to compose your thoughts.

Your *PSR* editor can handle just about any text and graphic format, so don't be afraid to submit whatever you have to wallow@tapr.org --- she can handle it!

The deadline for the next issue of *PSR* is May 1, so write early and write often.

###

On the Net

By Mark Thompson, WB9QZB

Facebook

As you may know, TAPR has a Facebook page, www.facebook.com/TAPRDigitalHam.

However, I also created a TAPR Facebook Group, www.facebook.com/groups/TAPRDigital/.

If you have a Facebook account, "Like" the TAPR Facebook page and join the TAPR Facebook Group.

If you join the group click on the Events link and indicate you're Going to the events.



On Twitter, Too

Access the TAPR Twitter account at www.twitter.com/taprdigital.



Also on YouTube

TAPR now has its own channel on YouTube: the TAPR Digital Videos Channel: www.youtube.com/user/TAPRDigitalVideo.



At this time, there are a slew of videos on our channel including many from the TAPR-ARRL Digital Communications Conference (DCC) that you may view at no cost, so have at it!

###

PSR

#148 Winter 2020, ISSN: 1052-3626

Published by TAPR

Phone +1 972 413 8277

E-mail: contact@tapr.org

URL www.tapr.org

Facebook www.facebook.com/TAPRDigitalHam

Twitter www.twitter.com/taprdigital

TAPR Office Hours: Monday to Friday, 9 AM to 5 PM Eastern Time

Submission Guidelines

TAPR is always interested in receiving information and articles for publication. If you have an idea for an article you would like to see, or you or someone you know is doing something that would interest TAPR, please contact the editor (wallou@tapr.org) so that your work can be shared with the Amateur Radio community. If you feel uncomfortable or otherwise unable to write an article yourself, please contact the editor for assistance. Preferred format for articles is plain ASCII text (OpenOffice or *Microsoft Word* is acceptable). Preferred graphic formats are PS/EPS/TIFF (diagrams, black and white photographs), or TIFF/JPEG/GIF (color photographs). Please submit graphics at a minimum of 300 DPI.

Production / Distribution

PSR is exported as Adobe Acrobat and distributed electronically at www.tapr.org

PSR Editor: Stana Horzepa, WA1LOU

E-mail wallou@tapr.org

TAPR Officers

President: Steve Bible, N7HPR, n7hpr@tapr.org

Vice President: Scotty Cowling, WA2DFI, wa2dfi@tapr.org

Secretary: Stana Horzepa, WA1LOU, wallou@tapr.org

Treasurer: Tom Holmes, N8ZM, n8zm@tapr.org

TAPR Board of Directors

Board Member, Call Sign, Term Expires, e-mail address

John Ackermann, N8UR, 2022, n8ur@tapr.org

Steve Bible, N7HPR, 2023, n7hpr@tapr.org

George Byrkit, K9TRV, 2021, k9trv@tapr.org

Scotty Cowling, WA2DFI, 2021, wa2dfi@tapr.org

Stana Horzepa, WA1LOU, 2023, wallou@tapr.org

John Koster, W9DDD, 2021, w9ddd@tapr.org

Dave Larsen, KV0S, 2022, kv0s.dave@gmail.com

Bruce Raymond, ND8I, 2022, bruce@raymondtech.net

Darryl Smith, VK2TDS, 2023, vk2tds@tapr.org

TAPR is a not-for-profit scientific research and development corporation [Section 501(c)(3) of the US tax code]. Contributions are deductible to the extent allowed by US tax laws. TAPR is chartered in the State of Arizona for the purpose of designing and developing new systems for digital radio communication in the Amateur Radio Service, and for disseminating information required, during, and obtained from such research.

PSR Advertising Rates

Full Page Ad for 1 issue: \$100, 4 issues: \$350

Half Page Ad for 1 issue: \$75, 4 issues: \$250

Quarter Page Ad for 1 issue: \$50, 4 issues: \$175



Membership Application

TAPR

1 Glen Ave., Wolcott, CT 06716-1442

Phone +1 972 413 8277, Monday-Friday, 9AM-5PM Eastern Time

E-mail contact@tapr.org URL www.tapr.org

Join or renew online at <https://tapr.org/logout/>

Benefits of a TAPR Membership:

- Subscription to the quarterly PSR
- 10% off most TAPR kits and publications
- Access to the TAPR digital library
- Latest information on TAPR R&D projects
- Co-sponsor of the annual TAPR-ARRL Digital Communications Conference (DCC)

Name _____ Call Sign _____

Address _____

City _____ State/Province _____ Postal Code _____

Country _____ Daytime Phone No. _____

E-mail Address _____

New Renewal \$30 X _____ number of years = \$ _____ total

Payment Method: Check Credit Card Money Order Cash Don't send cash through the mail !

STOP! Provide the following information only if paying by mail with a credit card:

AMEX Discover Mastercard VISA

Credit Card No. _____ Expiration Date _____ Security Code _____

Card Holder's Name _____

TAPR is a community that provides leadership and resources to radio amateurs for the purpose of advancing the radio art.