11 Naughty Signals; or, the Abuse of a Raspberry Pi

by Russell Handorf

There are a lot of different projects that have rejuvenated interest in HAM Radio, more notably Software Defined Radio (SDR). The more prominent projects and products are the USRP by Ettus Research, BladeRF by Nuand, and the HackRF by Mike Ossmann (in the order from the most expensive to least expensive). These radios vary in capability and have their own distinct utility, depending on what radio communication you'd like to study; however, if all you are specifically interested in is receiving a simplistic signal, then the Realtek SDR is typically the best and cheapest choice. This article will show you how to combine a Realtek SDR and a Raspberry Pi into a poor man's software defined radio tool for exploring how to receive and transmit in related radio systems.

11.1 Bandpass Filter

It is very important to have and to use a bandpass filter when using the Raspberry Pi as an FM transmitter, because PiFM is essentially a square wave generator. This means that you'll have a lot of harmonics as depicted in Figure 21. While the direct operational frequency range of PiFM is approximately 1 MHz to 250 MHz, the harmonics are still strong enough to reach frequencies below 1 MHz and as high as 500 MHz.

Because of these square wave characteristics, a mechanical SAW filter would be ideal to be able to control the frequencies you wish to transmit. However, there filters can set you back more than the Raspberry Pi, and may be hard to come by, unless there's a neighborly Ham Radio Outlet near you. So you may have to make your own band-pass filter.

To make your own high band and/or low band pass filters, you can assemble them based on the schematic in Figure 19.[54] Parts for the various amateur bands are listed in Figure 20.

11.2 Raspberry Pi FM Transmitter

For over a year now, it has been documented how to turn the Raspberry Pi into an FM transmitter by using the PiFM software.[55] Richard Hirst first demonstrated this technique in some C and Python code that generated spread-spectrum clock signals to output FM on GPIO pin #4. Oliver Mattos and Oskar Weigl have since enhanced PiFM to add more capabilities.

Be aware, however, that this technique has another problem beyond bleeding RF and having to use filters. Namely, the transmitter doesn't shut down gracefully after you quit PiFM. Therefore, you'll need a script to silence the transmission. We'll call it `pi-shutdown.sh` in the various examples that follow.

```
#/bin/bash
#pi-shutdown.sh
touch /tmp/empty && /home/pi/pifm /tmp/empty
```

11.3 AFSK

Audio Frequency Shift Keying (AFSK) is simply a method to modulate digital data as an analogue tone; you'll certainly recognize this as the tones your modem made. AFSK characteristically represents 1 as a "mark" and 0 as a "space". While not fast, AFSK does work very well in many applications where data is communicated over a consistent radio frequency. Because of these attributes, AFSK is frequently used for radio communications in industrial applications, embedded systems, and more. Using a program called `minimodem`, you'll be easily able to receive and transmit AFSK with a Realtek SDR and a Raspberry Pi. Marc1 from `kprod.eu` demonstrated some very simple techniques for doing so, which a few other neighbors have been tweaked and updated in the examples to follow.

To receive 1200 baud AFSK transmissions, a one-line script is all that's needed:

```
rtl_fm -f 146.0M -M wbfm -s 200000          \
       -r 48000 -o 6                        \
| sox -traw -r48k -es -b16 -c1 -V1 -        \
      -twav -                               \
| minimodem --rx -8 1200
```

What's happening here is that the program `rtl_fm` is tuned to 146.0 MHz, sampling at 200,000

[54] `http://www.kitsandparts.com/univlpfilter.php`
[55] `https://github.com/rm-hull/pifm`

samples per second and converting the output at a sample rate of 48000 Hz. The output from this is sent to `sox`, which is converting the audio received to the WAV file format. The output from `sox` is then sent to `minimodem`, which is decoding the WAV stream at 1200 baud, 8 bit ASCII.

Transmitting an AFSK signal is just as easy:

```
echo "knock knock... : `date +%c`"        \
| minimodem --tx -f -8 1200              \
             -f /home/pi/sentence.wav
/home/pi/pifm /home/pi/sentence.wav      \
               146.0 48000
/home/pi/pi-shutdown.sh
```

11.4 Other Transmission Examples

Because of the scriptability and simplicity of PiFM, other forms of transmissions become easily achievable too.

Morse Code (CW)

Either done by playing a pre-made audio file with dits and dahs, or by using the `cwwav` program written by Thomas Horsten to output directly to PiFM.[56]

```
echo hello world                         \
| cwwav -f 700 -w 20                     \
         -o /home/pi/morse.wav
/home/pi/pifm /home/pi/morse.wav         \
               146.0 48000
/home/pi/pi-shutdown.sh
```

Numbers Station

A numbers station is typically a government-owned transmitter that sends encoded messages to spies, operators, or employees of that said government anywhere in the world, where the messages are typically one way and seemingly random. The script below mimics the Cuban numbers station identified as HM01.[57] What is interesting about it is that the data it sends is encoded with a common HAM Radio protocol called RDFT. Transmitting RDFT on a Raspberry Pi can be difficult, therefore using a simple FM transmission of THOR8 or QPSK256 should be adequate; using FLDIGI should be of great help to create these messages.

A script can easily speak a series of words into the air by piping them into the `text2wave` utility:

```
system("echo $text | text2wave -F 22050 - "
       "| /home/pi/pifm - 144 22050");
```

DVBT with Metadata

One common practice for those who work with the RTL dongle is to remove to remove the DVB-T digital television kernel module. To receive this challenge, however, you will need to re-enable that module. To transmit it, you'll need hardware from Hides,[58] which can be had for a very low cost. The script below works with the UT-100C.

[56] `https://github.com/Kerrick/cwwav`
[57] `http://www.qsl.net/py4zbz/eni.htm`
[58] `http://www.hides.com.tw/product_cg74469_eng.html`

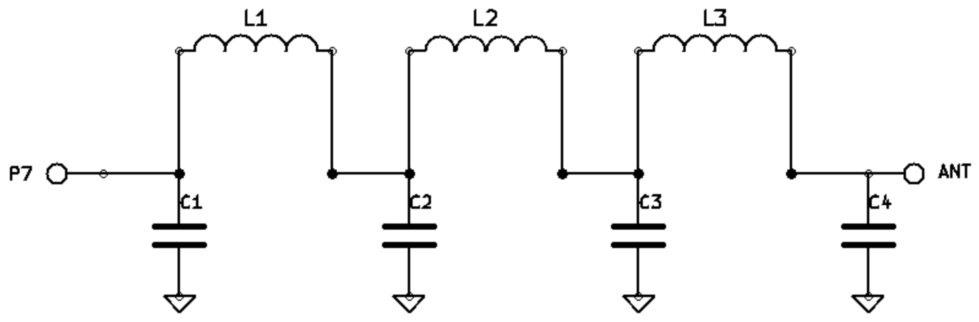


Figure 19: Bandpass Filter for Reducing PiFM Harmonics

```
modprobe usb-it950x
mkfifo ~/desktop
avconv -f x11grab -s 1024x768               \
  -framerate 30 -i :0.0                     \
  -vcodec libx264 -s 720x576                \
  -f mpegts                                 \
  -mpegts_original_network_id 1             \
  -mpegts_transport_stream_id 1             \
  -mpegts_service_id 1                      \
  -metadata                                 \
     service_provider="FCC CALL SIGN"       \
  -metadata                                 \
     service_name="Dialin for Dollars!"     \
  -muxrate 3732k -y ~/desktop &
tsrfsend ~/desktop 0 730000 6000 4          \
                   1/2 1/4 8 0 0 &
```

SSTV

Gerrit Polder developed a simple means of converting an image into a SSTV signal and then sending it out via the PiFM utility. Using his program, *PiSSTV*, command line transmissions of SSTV broadcasts with the Raspberry Pi are easy to achieve without the need for a graphical environment.

11.5 Howdy to the caring Neighbors

Thanks to the PiFM program, there are many portable options allowing HAM operators, experimenters, and miscreants to explore and butcher the radio waves on the cheap. The main goal of this article is to document the work of many friendly folks in this arena, gathering in one place the information currently scattered across the bits and bobs of the Internet. Owing to the brilliant hacks of these neighbors, it should become apparent why any radio nut should consider having a Raspberry Pi armed with a filter and some code. While out of scope for the article, it should also become clear how you too can make a very inexpensive and portable HAM station for a large variety of digital and analog modes.

I'd like to extend a warm, hearty, and, eventually, beer-supplemented thank-you to Dragorn, Zero_Chaos, Rick Mellendick, DaKahuna, Justin Simon, Tara Miller, Mike Ossmann, Rob Ghilduta, and Travis Goodspeed for their direct support.

Band λ Meters	C1, C4	C2, C3	L1, L3	L2
160	820	2200	$4.44\mu H, 20T, 16''$	$5.61\mu H, 23T, 18''$
80	470	1200	$2.43\mu H, 21T, 16''$	$3.01\mu H, 24T, 18''$
40	270	680	$1.38\mu H, 18T, 14''$	$1.70\mu H, 20T, 15''$
30	270	560	$1.09\mu H, 16T, 12''$	$1.26\mu H, 17T, 13''$
20	180	390	$0.77\mu H, 13T, 11''$	$0.90\mu H, 14T, 11''$
17	100	270	$0.55\mu H, 11T, 9''$	$0.68\mu H, 12T, 10''$
15	82	220	$0.44\mu H, 11T, 9''$	$0.56\mu H, 12T, 10''$
12	100	220	$0.44\mu H, 11T, 9''$	$0.52\mu H, 12T, 10''$
10	56	150	$0.30\mu H, 9T, 8''$	$0.38\mu H, 10T, 9''$

Figure 20: Filter Bill of Materials

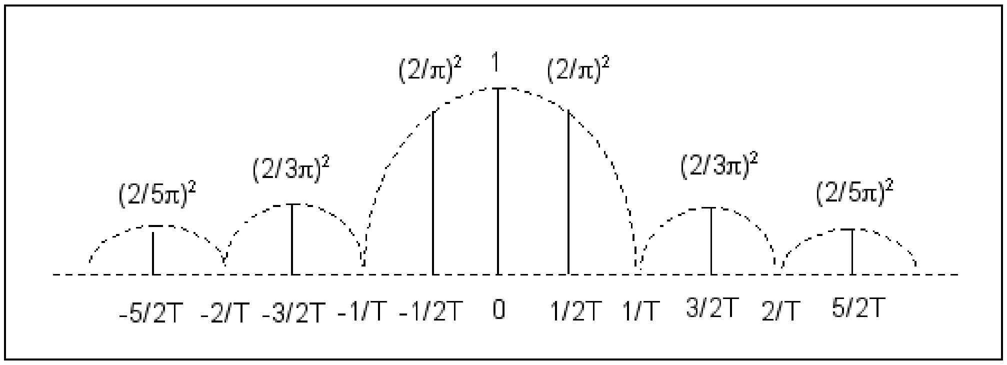
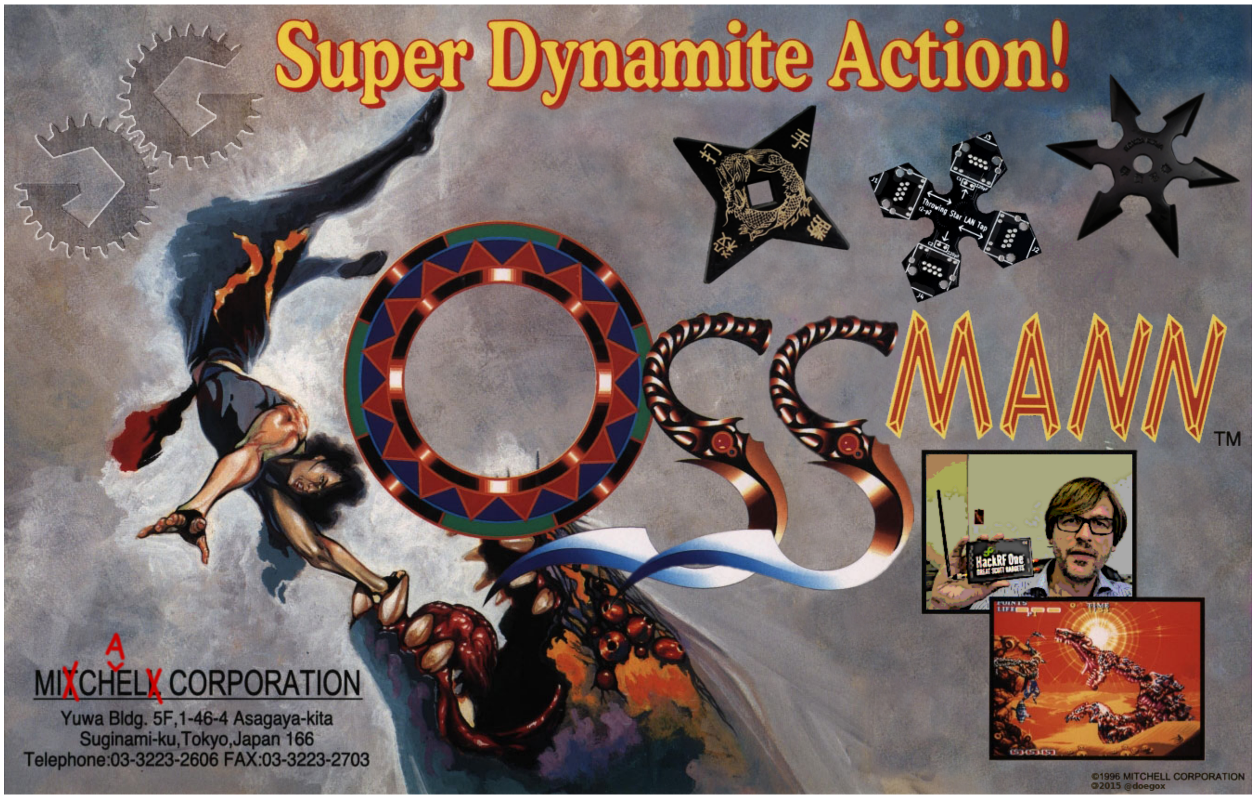


Figure 21: PiFM Harmonic Emissions


Super Dynamite Action!
OSSMANN™
MI~~T~~CHEL~~L~~ CORPORATION
A
Yuwa Bldg. 5F,1-46-4 Asagaya-kita
Suginami-ku,Tokyo,Japan 166
Telephone:03-3223-2606 FAX:03-3223-2703
©1996 MITCHELL CORPORATION
©2015 @doegox