Raspberry Pi Packet TNC,
APRS Digipeater, IGate

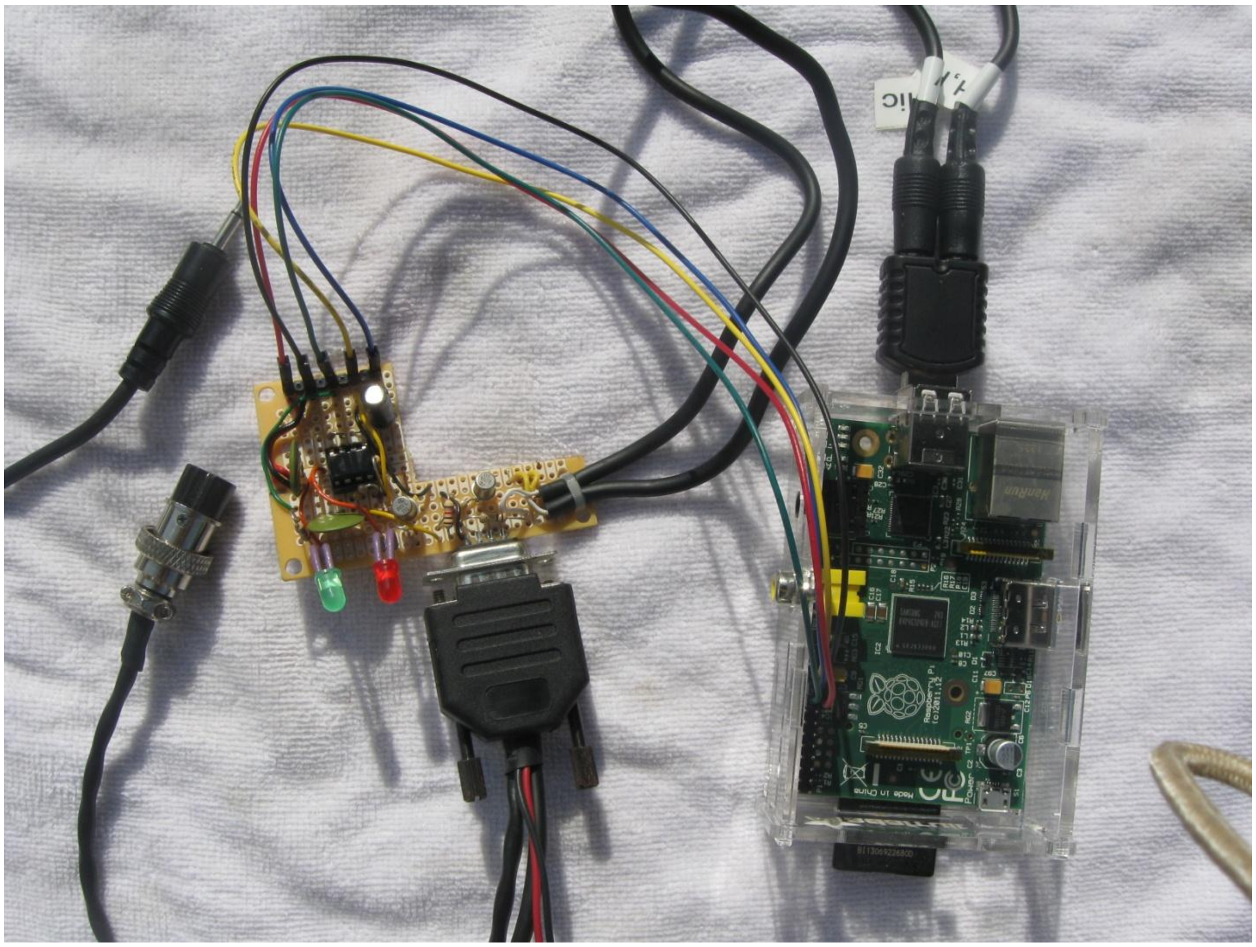
Version 1.3 – development snapshot I -- December 2015

In the early days of Amateur Packet Radio, it was necessary to use a “Terminal Node Controller” (TNC) with specialized hardware. Those days are gone. You can now get better results at lower cost by connecting your radio to the “soundcard” interface of a computer and running free software.

The Raspberry Pi (RPi) is a good platform for running a software TNC, APRS digipeater, and IGate. Why use a larger computer and waste hundreds of watts of power? All you need to add is a USB Audio Adapter ($8 or less) and a simple PTT circuit to activate the transmitter.

This document is a Quick Start guide for running Dire Wolf on the Raspberry Pi and describes special considerations where it may differ from other Linux systems.

After completing the steps here, refer to the User Guide for more details on the Linux version.



Left: Connections to speaker and Microphone jack of transceiver.

Middle: Interface circuit with a timer to limit transmission time.

It uses the standard 9 pin connector found on most TNCs and trackers so the same radio-specific cables can be used.

The two LEDs are for Data Carrier Detect (DCD) and the Push to Talk (PTT) signals.

The shape of the board was not intentional. It was just a scrap piece of perfboard left over from another project.

Right: Raspberry Pi.

At the top is a USB audio adapter. The current software version can handle 3 audio interfaces at the same time.

Lower left are GPIO connections for PTT and the DCD LED.

1 Install Appropriate Operating System 5
1.1 If operating system is already installed. 5
1.2 New System Setup & Configuration 6
1.3 Optionally enable IPv6. 6
1.4 Make a backup of your SD card (optional) 7
2 Install Dire Wolf 8
2.1 Install sound library 8
2.2 Download Dire Wolf source code 8
2.3 Compile and Install 9
2.4 Get initial configuration file. 11
2.5 First Test 11
3 Interface for radio 12
3.1 Audio input / output 12
3.1.1 Signalink USB 13
3.1.2 Cheap USB audio adapters 13
3.1.3 Avoid USB hub for audio adapter 14
3.2 Determine “card” number for audio device 14
3.3 Transmitter PTT 15
3.4 Printed Circuit Board 17
3.5 Other Alternatives 18
4 Configure for Radio Interface 19
4.1 Audio Interface Device 19
4.2 PTT Method 19
4.3 Optional DCD Indicator 19
5 Run Dire Wolf application 20
5.1 Use with client applications 22
6 Other Common Configuration Changes 24
6.1 Automatic Startup 24
6.2 Digipeater Operation 24
6.3 Enable Beaconing 25
6.4 Internet Gateway (IGate) 25
7 Receive Decoding Performance and CPU Requirements 27
8 Troubleshooting 28
8.1 Before that time… 29

8.2 USB Hubs 29
8.3 Use with AX25 kissattach 30

1 Install Appropriate Operating System

Select a version of Linux that has hardware floating point support. Dire Wolf makes extensive use of floating point calculations. Trying to use the slower software floating point will probably result in disappointment.

The Raspbian “Wheezy” and “Jessie” distributions from http://www.raspberrypi.org/downloads are known to work with the instructions here. I haven’t tried the others and don’t know how they might differ.

The Raspian operating system distribution comes with the **gcc** compiler and most required libraries pre-installed. If you use a different operating system version, you might need to install a suitable compiler and/or additional libraries.

1.1 If operating system is already installed.

If you already have a different operating system version, verify that gcc is configured to generate hardware floating point code. Enter the “gcc –v” command and observe the result. Make sure that “--with-fpu=vfp --with-float=hard” appears in the configuration.

```
pi@raspberrypi:~$ gcc -v

Using built-in specs.
COLLECT_GCC=gcc
COLLECT_LTO_WRAPPER=/usr/lib/gcc/arm-linux-gnueabihf/4.6/lto-wrapper
Target: arm-linux-gnueabihf
Configured with: ../src/configure -v --with-pkgversion='Debian 4.6.3-12+rpi1' -
-with-bugurl=file:///usr/share/doc/gcc-4.6/README.Bugs --enable-
languages=c,c++,fortran,objc,obj-c++ --prefix=/usr --program-suffix=-4.6 --
enable-shared --enable-linker-build-id --with-system-zlib --libexecdir=/usr/lib
--without-included-gettext --enable-threads=posix --with-gxx-include-
dir=/usr/include/c++/4.6 --libdir=/usr/lib --enable-nls --with-sysroot=/ --
enable-clocale=gnu --enable-libstdcxx-debug --enable-libstdcxx-time=yes --
enable-gnu-unique-object --enable-plugin --enable-objc-gc --disable-sjlj-
exceptions --with-arch=armv6 --with-fpu=vfp --with-float=hard --enable-
checking=release --build=arm-linux-gnueabihf --host=arm-linux-gnueabihf --
target=arm-linux-gnueabihf
Thread model: posix
gcc version 4.6.3 (Debian 4.6.3-12+rpi1)
```

1.2 New System Setup & Configuration

If you are starting with a brand new RPi, the lessons here: http://learn.adafruit.com/ are easy to follow and will get you running quickly.

- https://learn.adafruit.com/adafruit-raspberry-pi-lesson-1-preparing-and-sd-card-for-your-raspberry-pi
- https://learn.adafruit.com/adafruits-raspberry-pi-lesson-2-first-time-configuration
- https://learn.adafruit.com/adafruits-raspberry-pi-lesson-3-network-setup
- https://learn.adafruit.com/adafruits-raspberry-pi-lesson-4-gpio-setup
- https://learn.adafruit.com/adafruits-raspberry-pi-lesson-6-using-ssh
- https://learn.adafruit.com/adafruit-raspberry-pi-lesson-7-remote-control-with-vnc

The last two are optional, depending on your situation. In my case, I have a headless (no monitor, keyboard) Raspberry Pi in the radio "shack" which is accessed from other computers various places over the network. The VNC software allows me to have a virtual desktop displayed on other computers. If you are using a monitor / keyboard / mouse connected to the RPi, you won't need to configure login by ssh and VNC.

Get the latest software and firmware updates.

```
sudo apt-get update

sudo apt-get dist-upgrade

sudo rpi-update

sudo reboot
```

The audio system on the Raspberry Pi has a history of many problems. Pulseaudio was never right and it was even worse the last time I struggled with it. I believe it is no longer included in the current version of Raspbian. (See http://elinux.org/R-Pi_Troubleshooting#Removal_of_installed_pulseaudio)

Remove pulseaudio if it is installed. Note: this applies only to the Raspberry Pi, and probably other similar ARM-based systems. Pulseaudio is fine on desktop/laptop computers with x86 processors.

```
sudo apt-get remove --purge pulseaudio

sudo apt-get autoremove

rm -rf /home/pi/.pulse
```

At this point reboot to make sure you are running the latest versions.

```
sudo reboot
```

1.3 Optionally enable IPv6.

You can enable IPv6 immediately by typing: "sudo modprobe ipv6".

To start it up automatically after each reboot, edit /etc/modules and add a new line at the end containing “ipv6” (without the quotes).

1.4 Make a backup of your SD card (optional)

After going through all of these steps, you might want to make a backup so you can get back to this point quickly if the memory card gets trashed. Here’s how:
https://www.raspberrypi.org/forums/viewtopic.php?p=239331

2 Install Dire Wolf

2.1 Install sound library.

First, install the "libasound2-dev" package with the following command:

```
sudo apt-get install libasound2-dev
```

Failure to install libasound2-dev step will result in a compile error resembling "audio.c:…: fatal error: alsa/asoundlib.h: No such file or directory"

2.2 Download Dire Wolf source code

Follow these steps to clone the git repository and checkout the desired version.

```
cd ~
git clone https://www.github.com/wb2osz/direwolf
cd direwolf
git tag
```

You should see a list of releases something like this:

```
1.0
1.1
1.2
1.3-dev-F
1.3-dev-I
```

Most people will want the latest stable version, in this case 1.2. In that case, type:

```
git checkout 1.2
```

In some cases, you might want an unstable development version to test a bug fix or get a preview of a new (possibly incomplete) feature that will be in the next release. In that case, you would specify a development snapshot tag such as:

```
git checkout 1.3-dev-I
```

The very adventurous might want the absolute latest development version which might not be stable. In that case, type:

```
git checkout dev
```

2.3 Optional support for hamlib

Skip this step if you don't want to use "hamlib."

"hamlib" can be included to provide support for more types of PTT control.

Build from source because a Raspbian package doesn't seem to be available. Here is my "cheat sheet" version boiled down from http://hamlib.sourceforge.net/manuals/1.2.15/_rdmedevel.html

```
sudo apt-get install automake libtool texinfo
cd ~
git clone git://hamlib.git.sourceforge.net/gitroot/hamlib/hamlib
cd hamlib
sh autogen.sh
make
make check
sudo make install
cd ~
```

You should now have many new files including:

- `/usr/local/include/hamlib/rig.h`
- `/usr/local/lib/libhamlib.so`

When building direwolf, the compiler and linker know enough to search /usr/local/include/… and /usr/local/lib/… but when it comes time to run direwolf, you might see a message like this:

```
direwolf: error while loading shared libraries: libhamlib.so.2: cannot
open shared object file: No such file or directory
```

Edit your ~/.bashrc file and add this after the initial comment lines, and before the part that tests for running interactively.

```
export LD_LIBRARY_PATH=/usr/local/lib
```

Type this so it will take effect now, instead of waiting for next login:

```
source ~/.bashrc
```

Edit `direwolf/Makefile.linux` and look for this section:

```
# Uncomment following lines to enable hamlib support.
#CFLAGS += -DUSE_HAMLIB
#LDFLAGS += -lhamlib
```

Remove the # from the beginning of the last two lines.

2.4 Optional support for gpsd

This is covered in the separate document, ***Raspberry-Pi-APRS-Tracker.pdf.***

2.5 Compile and Install

Compile an install the application.

```
cd ~/direwolf
make
sudo make install
make install-rpi
```

You should now have files, and more, in these locations, under /usr/local, owned by root.

/usr/local/bin/direwolf	The application.
/usr/local/bin/decode_aprs	Utility to interpret "raw" data you might find on http://aprs.fi or http://findu.com
/usr/local/bin/tt2text /usr/local/bin/text2tt /usr/local/bin/ll2utm /usr/local/bin/utm2ll /usr/local/bin/log2gpx /usr/local/bin/gen_packets	Utilities related to APRStt gateway, UTM coordinates, log file to GPX conversion, and test packet generation.
/usr/share/applications/direwolf.desktop	Application definition with icon, command to execute, etc.
/usr/share/direwolf/tocalls.txt	Mapping from destination address to system type. Search order for tocalls.txt is first the current working directory and then /usr/share/direwolf.
/usr/share/direwolf/symbolsX.txt /usr/share/direwolf/symbols-new.txt	Descriptions and codes for APRS symbols.
/usr/share/direwolf/dw-icon.png	Icon for the desktop.
/usr/local/share/doc/direwolf/* /usr/local/man/man1/*	Various documentation.
/usr/local/share/doc/direwolf/examples/*	Sample configuration and other examples.

You should also have these files, under /home/pi.

/home/pi/Desktop/direwolf.desktop	Symbolic link to /usr/share/applications/direwolf.desktop. This causes an icon to be displayed on the desktop.
/home/pi/dw-start.sh	Script to start Dire Wolf if it is not running already.

2.6 Get initial configuration file.

If this is the first time, you are installing Dire Wolf perform this step:

```
make install-conf
```

When upgrading from an earlier version, you will probably want to skip this step because it will wipe out your earlier configuration file.

This step should have copied the initial configuration file to the home directory, /home/pi.

/home/pi/direwolf.conf	Configuration file. Search order is current working directory then the user's home directory.

2.7 First Test

Go to your home directory and try to run "direwolf."

```
cd /home/pi
direwolf
```

You should see something like this, because we have not yet configured it for using an audio device.

```
Dire Wolf version …
Audio device for both receive and transmit: default
Could not open audio device default for input
No such file or directory
Pointless to continue without audio device.
```

We will perform the necessary configuration in a later step.

3 Interface for radio

APRS, or other packet radio, operation requires connections between your transceiver and computer.

(a) Received audio from receiver.

This can be very simple. All you need is a cable from the speaker of your radio to the computer's microphone or line in. Some people like to use audio isolation transformers but I never found this to be necessary.

(b) Transmit audio to transmitter.

Audio output from the computer goes to the microphone input of your transceiver. A direct connection might be acceptable in some cases but typically you would want a trim pot to decrease the signal level and a capacitor (perhaps around 0.1 µF) to block DC.

(c) Push to Talk (PTT) signal to activate transmitter.

Traditionally, this has often been with one of the control lines of an RS-232 serial port. It would be possible to use a USB-to-serial adapter but there is a better way. The Raspberry Pi has a bunch of general purpose input output (GPIO) pins.

If you use a VOX circuit, to turn on transmitter automatically when audio is present, no PTT connection would be necessary. You can also ignore the warning message about PTT not being configured.

3.1 Audio input / output

The Raspberry Pi has built-in audio output but no audio input.

You can get a list of audio **output** devices with the "aplay -l" (that's lower case L option) command.

```
pi@raspberrypi:~$ aplay –l

**** List of PLAYBACK Hardware Devices ****
card 0: ALSA [bcm2835 ALSA], device 0: bcm2835 ALSA [bcm2835 ALSA]
  Subdevices: 8/8
  Subdevice #0: subdevice #0
  Subdevice #1: subdevice #1
  Subdevice #2: subdevice #2
  Subdevice #3: subdevice #3
  Subdevice #4: subdevice #4
  Subdevice #5: subdevice #5
  Subdevice #6: subdevice #6
  Subdevice #7: subdevice #7
card 0: ALSA [bcm2835 ALSA], device 1: bcm2835 ALSA [bcm2835 IEC958/HDMI]
  Subdevices: 1/1
  Subdevice #0: subdevice #0
```

You can get a list of audio **input** devices with the “arecord -l” (again lower case L option) command:

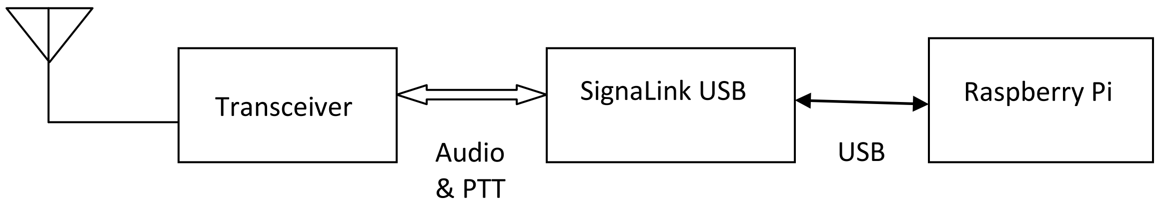
```
pi@raspberrypi:~$ arecord -l

**** List of CAPTURE Hardware Devices ****
```

There aren’t any! It will be necessary to add some sort of sound input device.

3.1.1 Signalink USB

The **Tigertronics SignaLink USB** is a pricey solution but easy because it contains all of the interface circuitry. Be sure to turn the “DLY” delay control completely counter-clockwise.



Don’t try using the SignaLink USB for 9600 baud! It doesn’t have enough bandwidth. See 9600 section of the User Guide for more details.

3.1.2 Cheap USB audio adapters

A cheap USB audio adapter can also be used but the drivers and firmware can be little problematic.

Recommended:

I’m currently using this Syba adapter http://www.amazon.com/gp/product/B001MSS6CS The sticker on the package indicated it has a CMedia HS100 chip (see http://www.cmedia.com.tw/ApplicationIndex/C1Serno-1/C2Serno-4.html) but “lsusb” reports:

```
Bus 001 Device 004: ID 0d8c:0139 C-Media Electronics, Inc.
```

The C-Media product id 0139 indicates it’s the CM108**AH** chip. The HS100 has a product id of 013C.

When I bought another one more recently, “lsusb” reports:

```
Bus 001 Device 006: ID 0d8c:0008 C-Media Electronics, Inc.
```

So it appears to have a different type of chip inside. This also works fine.

Avoid:

This one looks interesting because it has a short cable which provides more flexibility in the physical arrangement of the various components, specifically using multiple devices at the same time.. http://www.adafruit.com/product/1475

"lsusb" reports this:

```
Bus 001 Device 005: ID 0d8c:000c C-Media Electronics, Inc. Audio Adapter
```

This appears to be the CM108 or CM109 chip. According to the CM108 datasheet, the product id can vary from 0008 to 000f and is programmable by the MSEL and MODE pins. The CM109 datasheet says the same thing so I don't know how you can tell them apart.

It works fine on Windows 7 but I have not yet been able to get audio input working properly on the RPi even with the latest (December 2014) firmware. I'd like to hear from anyone that is successful. (Update, September 2015. This still doesn't work properly with the RPi. All audio samples are 0.)

There is a report of this one spewing out a lot of RFI at 144 MHz so you might want to avoid it even after a properly working driver is available.

3.1.3 Avoid USB hub for audio adapter

Avoid putting a USB hub between the Raspberry Pi and the USB Audio adapter. I've received reports that exactly the same configuration works fine with the direct connection and a significant number of audio samples are lost when going through a hub.

3.2 Determine "card" number for audio device

After plugging in a suitable USB audio interface, you should see something like this, in response to the "aplay –l" and "arecord –l" commands:

```
pi@raspberrypi:~$ aplay -l

**** List of PLAYBACK Hardware Devices ****
card 0: ALSA [bcm2835 ALSA], device 0: bcm2835 ALSA [bcm2835 ALSA]
  Subdevices: 8/8
  Subdevice #0: subdevice #0
  Subdevice #1: subdevice #1
  Subdevice #2: subdevice #2
  Subdevice #3: subdevice #3
  Subdevice #4: subdevice #4
  Subdevice #5: subdevice #5
  Subdevice #6: subdevice #6
  Subdevice #7: subdevice #7
card 0: ALSA [bcm2835 ALSA], device 1: bcm2835 ALSA [bcm2835 IEC958/HDMI]
  Subdevices: 1/1
  Subdevice #0: subdevice #0

card 1: CODEC [USB Audio CODEC], device 0: USB Audio [USB Audio]
  Subdevices: 1/1
  Subdevice #0: subdevice #0

pi@raspberrypi:~$ arecord -l

**** List of CAPTURE Hardware Devices ****

card 1: CODEC [USB Audio CODEC], device 0: USB Audio [USB Audio]
  Subdevices: 1/1
  Subdevice #0: subdevice #0
```

The built-in (output only) interface will normally be card 0 unless you did something to change the order. The additional USB audio adapter will typically be card 1, as seen in the example above. If you see a different number, you will have to modify a later step when we put the card number in the configuration file.

3.3 Transmitter PTT

If you want to transmit, some method is needed to activate the transmitter push-to-talk (PTT) function. Traditionally, this has been done with the RTS or DTR signal from a serial port.

Possible alternatives include:

- Reconfigure the built in serial port to have the RTS control.

  The Raspberry Pi has a built in serial port which is normally configured as a console. It is possible to reconfigure this to be available for application use. In the default configuration, it does not have the usual RTS/CTS flow control signals. As described here, http://elinux.org/RPi_Serial_Connection#Handshaking_lines , it is possible to reconfigure some of the GPIO lines for the RTS/CTS signals. I haven't tried this.

- Use a USB to RS-232 converter cable to supply a full RS-232 port. You will need to put something like this in the configuration file.

  ```
  PTT  /dev/ttyUSB0  RTS
  ```

- VOX operation where transmitter is activated by transmit audio signal. The SignaLink USB uses this technique. (Homebrew circuit example: https://sites.google.com/site/kh6tyinterface/)

I do not recommend using the VOX built into transceivers because they are designed for voice and keep the transmitter on much too long after the audio has stopped. The User Guide has a detailed explanation.

- Use one of the General Purpose I/O (GPIO) pins. I'm currently using this along with the cheap USB audio adapter.

  CAUTION! The general purpose input output (GPIO) pins are connected directly to the CPU chip. There is no buffering or other protection. The interface uses 3.3 volts and will not tolerate 5 volt signals. Static discharge, from careless handing, could destroy your Raspberry Pi.

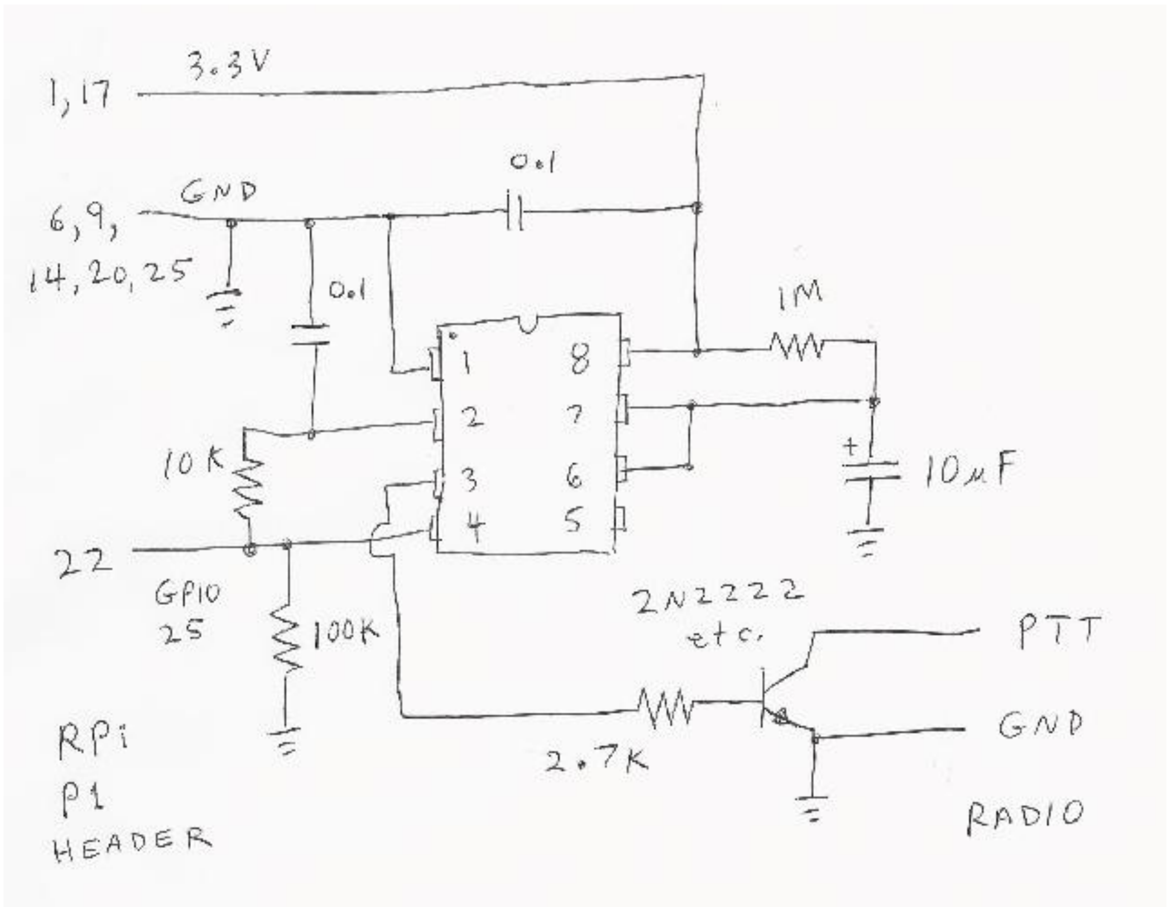
There are many GPIO pins. How would you choose an appropriate one? These are my suggestions for the best choices. The others have special functions such as UART, SPI, PWM, or I^{2}C. A few others changed positions between board revisions 1 & 2 which could cause confusion.

- P1-11 GPIO 17
- P1-15 GPIO 22
- P1-16 GPIO 23
- P1-18 GPIO 24
- P1-22 GPIO 25

The A+, B+, and model 2 have a larger connector with additional GPIO pins. Any of them would be suitable:

- P1-29 GPIO 5
- P1-31 GPIO 6
- P1-32 GPIO 12
- P1-33 GPIO 13
- P1-35 GPIO 19
- P1-36 GPIO 16
- P1-37 GPIO 26
- P1-38 GPIO 20
- P1-40 GPIO 21

Here is a suggested circuit using a **CMOS** 555 timer (LMC555, TLC555, ICM7555, etc.) to limit transmissions to about 10 seconds. **Don't try using the original 555** because it needs a minimum of 4.5 volts and we have only 3.3 here. The time can be increased by making the 10 µF capacitor larger. It's roughly 1 second for each µF.



You could get by with only a resistor and transistor but a software failure could cause the transmitter to be stuck on, jamming the radio channel, annoying other people, and possibly damaging the transmitter from overheating.

Finally, put a command like this in the configuration file with the actual I/O signal being used.

```
PTT  GPIO  25
```

One person reported that the timeout did not work properly and the problem was solved by adding a 0.0.1 µf capacitor from pin 4 to ground. https://groups.yahoo.com/neo/groups/Raspberry_Pi_4-Ham_RADIO/conversations/messages/4737

3.4 Printed Circuit Board

Here is a design for a printed circuit board https://github.com/jaymzx/RPi-WD-Packet-Interface

The README.md file contains a link to where you can order them.

3.5 Other Alternatives

The KF5INZ “Easy Digi” interface http://www.aracc.org/EASYDIGI!.pdf keeps popping up in discussions of interfaces. It uses audio transformers and an opto-isolator for PTT to break up ground loops. This could help avoid issues with AC hum and RFI.

Audio transformers are fine for 1200 baud but should be avoided if you want to use 9600 which requires more audio bandwidth. See 9600 section of the User Guide for more details.

4 Configure for Radio Interface

We need to make a couple simple changes to the configuration file. Change to your home directory.

```
        cd /home/pi
```

You should find a file called “direwolf.conf.” Modify it with your favorite text editor, as explained in the following steps.

4.1 Audio Interface Device

We previously determined the “card” number for the desired audio interface. Typically this will be 1.

Look in the “direwolf.conf” file for a line like this:

```
        # ADEVICE  plughw:1,0
```

Remove the “#” from the beginning of the line. If the card number, found in the earlier step, is not 1, use that number instead.

4.2 PTT Method

Look in “direwolf.conf” for a line like this:

```
        # PTT GPIO 25
```

Assuming you are using GPIO 25 for the PTT control, simply remove the “#” from the beginning of the line. If using some other PTT method, refer to the User Guide for a discussion of other possibilities.

4.3 Optional DCD Indicator

If you would like a data carrier detect (DCD) LED to light up when it looks like a signal is present, add a line like this:

```
        DCD GPIO -24
```

This means drive GPIO 24 to ground for signal present. Connect an LED like this:

Pin 18 (GPIO 24) ---- (cathode) LED (anode) ---- 270 ohm resistor ---- 3.3 volts

If using the printed circuit board mentioned earlier, use the opposite polarity because we want a positive signal to turn on the LED.

```
        DCD GPIO 24
```

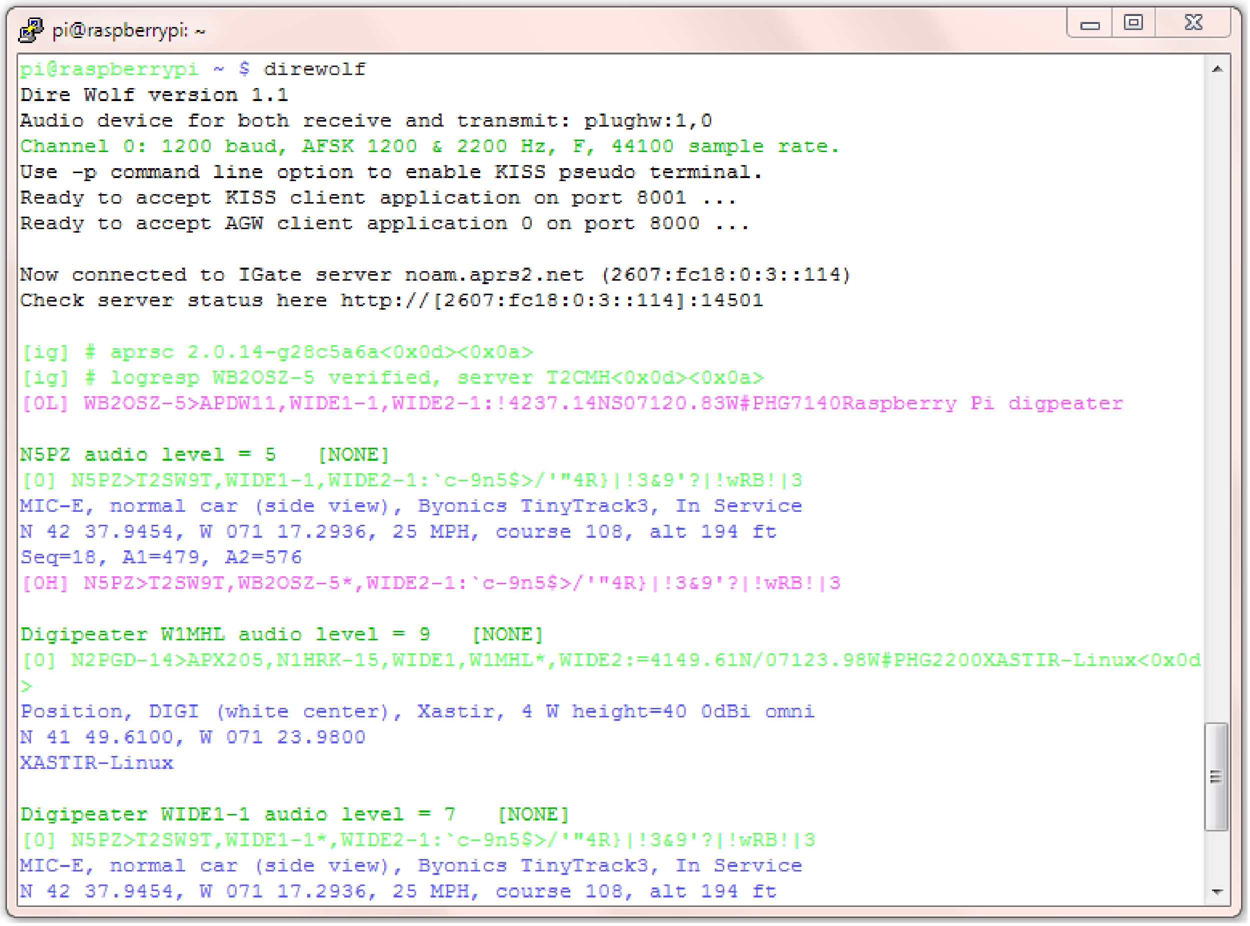
5 Run Dire Wolf application

Start up the application with the desktop icon or from the command line:

```
direwolf
```

Just run it as an ordinary user. There is no need to run it as root.

Typical results will look something like this:



```
pi@raspberrypi ~ $ direwolf
Dire Wolf version 1.1
Audio device for both receive and transmit: plughw:1,0
Channel 0: 1200 baud, AFSK 1200 & 2200 Hz, F, 44100 sample rate.
Use -p command line option to enable KISS pseudo terminal.
Ready to accept KISS client application on port 8001 ...
Ready to accept AGW client application 0 on port 8000 ...

Now connected to IGate server noam.aprs2.net (2607:fc18:0:3::114)
Check server status here http://[2607:fc18:0:3::114]:14501

[ig] # aprsc 2.0.14-g28c5a6a<0x0d><0x0a>
[ig] # logresp WB2OSZ-5 verified, server T2CMH<0x0d><0x0a>
[0L] WB2OSZ-5>APDW11,WIDE1-1,WIDE2-1:!4237.14NS07120.83W#PHG7140Raspberry Pi digpeater

N5PZ audio level = 5    [NONE]
[0] N5PZ>T2SW9T,WIDE1-1,WIDE2-1:`c-9n5$>/'"4R}|!3&9'?|!wRB!|3
MIC-E, normal car (side view), Byonics TinyTrack3, In Service
N 42 37.9454, W 071 17.2936, 25 MPH, course 108, alt 194 ft
Seq=18, A1=479, A2=576
[0H] N5PZ>T2SW9T,WB2OSZ-5*,WIDE2-1:`c-9n5$>/'"4R}|!3&9'?|!wRB!|3

Digipeater W1MHL audio level = 9    [NONE]
[0] N2PGD-14>APX205,N1HRK-15,WIDE1,W1MHL*,WIDE2:=4149.61N/07123.98W#PHG2200XASTIR-Linux<0x0d
>
Position, DIGI (white center), Xastir, 4 W height=40 0dBi omni
N 41 49.6100, W 071 23.9800
XASTIR-Linux

Digipeater WIDE1-1 audio level = 7    [NONE]
[0] N5PZ>T2SW9T,WIDE1-1*,WIDE2-1:`c-9n5$>/'"4R}|!3&9'?|!wRB!|3
MIC-E, normal car (side view), Byonics TinyTrack3, In Service
N 42 37.9454, W 071 17.2936, 25 MPH, course 108, alt 194 ft
```

Let's examine each section.

On the first line, we have the application version.
Next we have confirmation of the audio device being used.

The modem is configured for standard 1200 baud operation.

```
Dire Wolf version 1.2
Audio device for both receive and transmit: plughw:1,0
Channel 0: 1200 baud, AFSK 1200 & 2200 Hz, F, 44100 sample rate.
```

This is a summary of the interfaces available for client applications.

```
Use -p command line option to enable KISS pseudo terminal.
Ready to accept KISS client application on port 8001 ...
Ready to accept AGW client application 0 on port 8000 ...
```

This shows that the station is operating as an Internet Gateway and has successfully connected to a server. A URL is provided to check on the server status. "[ig]" at the beginning of the line indicates a response from the IGate server.

```
Now connected to IGate server noam.aprs2.net (2607:fc18:0:3::114)
Check server status here http://[2607:fc18:0:3::114]:14501

[ig] # aprsc 2.0.14-g28c5a6a<0x0d><0x0a>
[ig] # logresp WB2OSZ-5 verified, server T2CMH<0x0d><0x0a>
```

This is a beacon, identifying the station. "[0L]" means it is transmitting on radio channel 0 at low priority. Notice that transmitted lines are displayed in magenta.

```
[0L] WB2OSZ-5>APDW11,WIDE1-1,WIDE2-1:!4237.14NS07120.83W#
                                    PHG7140Raspberry Pi digpeater
```

Here we are receiving a nearby station directly. "[0]" means it was heard on radio channel 0. Received signals are in green.

```
N5PZ audio level = 5   [NONE]
[0] N5PZ>T2SW9T,WIDE1-1,WIDE2-1:`c-9n5$>/'"4R}|!3&9'?|!wRB!|3
```

What to all those strange characters mean? The next few lines contain the decoded information in human readable form. In this example, the last line is telemetry data.

```
MIC-E, normal car (side view), Byonics TinyTrack3, In Service
N 42 37.9454, W 071 17.2936, 25 MPH, course 108, alt 194 ft
Seq=18, A1=479, A2=576
```

This packet is a candidate for digipeating so we retransmit it. "[0H]" means it is being transmitted on channel 0. Digipeated packets go out at a higher priority. Notice how "WIDE1-1" was replaced by "WB2OSZ-5" so the actual path taken can be determined.*

```
[0H] N5PZ>T2SW9T,WB2OSZ-5*,WIDE2-1:`c-9n5$>/'"4R}|!3&9'?|!wRB!|3
```

Now we hear another station. But this time it is not direct. We are actually hearing a digipeater. It is not a candidate for digipeating because the via path has been all used up. In this case, the last line is the comment.

```
Digipeater W1MHL audio level = 9   [NONE]
[0] N2PGD-14>APX205,N1HRK-15,WIDE1,W1MHL*,WIDE2:=4149.61N/07123.98W#
                                             PHG2200XASTIR-Linux<0x0d>
Position, DIGI (white center), Xastir, 4 W height=40 0dBi omni
N 41 49.6100, W 071 23.9800
XASTIR-Linux
```

Now we hear the first station again, but from a digipeater, not directly. The digipeater did not insert its own call so we don't know who retransmitted the packet.

```
Digipeater WIDE1-1 audio level = 7   [NONE]
[0] N5PZ>T2SW9T,WIDE1-1*,WIDE2-1:`c-9n5$>/'"4R}|!3&9'?|!wRB!|3
MIC-E, normal car (side view), Byonics TinyTrack3, In Service
N 42 37.9454, W 071 17.2936, 25 MPH, course 108, alt 194 ft
Seq=18, A1=479, A2=576
```

This is also a candidate for digipeating because "WIDE2-1" has not been used up yet. However, we transmitted this same packet within the past 30 seconds so the duplicate is dropped.

```
Digipeater: Drop redundant packet.
```

That's all you will see. No flashy graphics. No maps. No user interaction. But loads of valuable information for monitoring activity and troubleshooting problems.

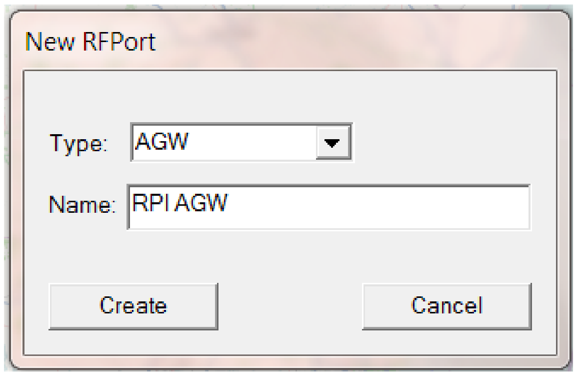
Dire Wolf is just a replacement for a TNC. To do more interesting things, you will need an application such as APRSISCE/32, UI-View32, Xastir, APRS-TW, YAAC, SARTrack, RMS Express, etc.

5.1 Use with client applications

Client applications can run on different computers and communicate with Dire Wolf over your local network. For example, you might have a Raspberry Pi in your "shack," in a cold damp basement, connected to your radio equipment. You might want to use a client application (such as APRSIS32, YAAC, or Xastir) on a laptop in a more comfortable location, perhaps next to the pool or fireplace. Here is an example of how you could configure APRSISCE/32 in this situation.

From the Configure menu, pick Ports → New Port…

Choose type of AGW and give it a meaningful name.

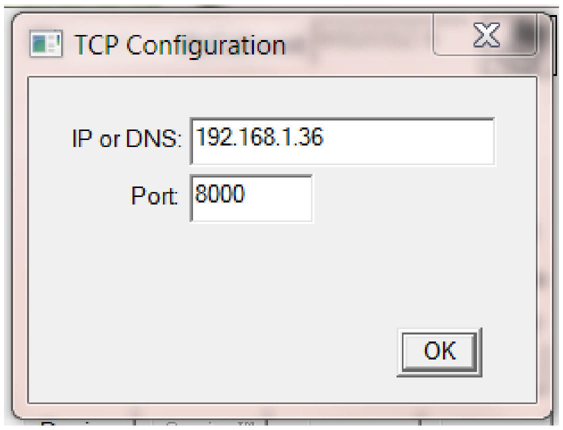

New RFPort

Type: AGW

Name: RPI AGW

Click on Create and pick port type of TCP/IP.

Enter the IP address of the Raspberry Pi, and the default port of 8000.


TCP Configuration

IP or DNS: 192.168.1.36

Port: 8000

6 Other Common Configuration Changes

6.1 Automatic Startup

You probably want your TNC / application server / digipeater to start up automatically after a power interruption.

If you followed the installation steps above, you should have a file named dw-start.sh in your home directory. Run the "crontab –e" command and add a line like this:

```
* * * * *  /home/pi/dw-start.sh  >/dev/null  2>&1
```

This script will run once per minute. Dire Wolf is started automatically if not running already. If it crashes, or is terminated for any other reason, it will be restarted. A log of restarts can be found in `/tmp/dw-start.log`.

6.2 Digipeater Operation

Edit the **/home/pi/direwolf.conf** file and look for a section like this:

```
# Station identifier for this channel.
# Multiple channels can have the same or different names.
#
# Naturally it must be up to letters and digits with an optional ssid.
# The APRS specification requires that it be upper case.
#
#
# Example (don't use this unless you are me):  MYCALL   WB2OSZ-5
#

MYCALL NOCALL
```

Change "NOCALL" to your ham radio call and optional SSID.

Next, look for a section like this:

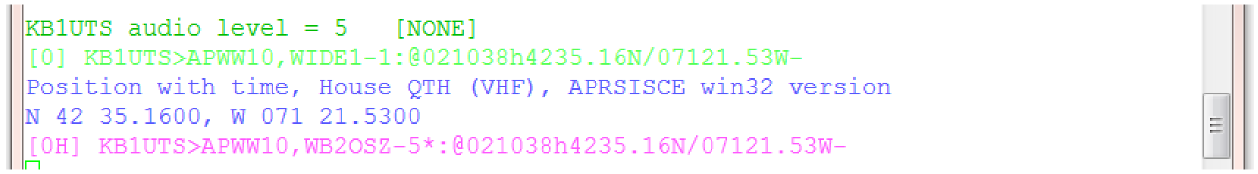
```
#-------------------------------------------------------
# ----------  Example 1:  Typical digipeater  ----------
#-------------------------------------------------------

#
# For most common situations, use something like this by removing
# the "#" from the beginning of the line.
# To disable digipeating, put # at the beginning of the line.
#

#
#DIGIPEAT 0 0 ^WIDE[3-7]-[1-7]$ ^WIDE[12]-[12]$
```

Remove the "#" character from the beginning of the last line shown above. It is necessary to stop and restart the application to notice configuration file changes.

Here is an example of what you should see:



```
KB1UTS audio level = 5    [NONE]
[0] KB1UTS>APWW10,WIDE1-1:@021038h4235.16N/07121.53W-
Position with time, House QTH (VHF), APRSISCE win32 version
N 42 35.1600, W 071 21.5300
[0H] KB1UTS>APWW10,WB2OSZ-5*:@021038h4235.16N/07121.53W-
```

Dark green: Information about the station we heard. Either the originating station or a digipeater.

Green: Raw received data. Notice that the digipeater field contains "WIDE1-1."

Blue: An explanation for troubleshooting. The destination (APWW10) is used to determine the type of system generating the signal.

Magenta: This is the re-transmitted packet. Notice that the digipeater field now contains "WB2OSZ-5*." The "*" indicates that it has been used up and won't be digipeated again.

6.3 Enable Beaconing

Be sure that MYCALL has been set as in previous step.

Look for a section like this in direwolf.conf file.

```
# PBEACON delay=0:15  every=30 overlay=S symbol="digi" lat=42^37.14N lon=071^20.83W
       power=50 height=20 gain=3 comment="Chelmsford MA" via=WIDE1-1,WIDE2-1
# PBEACON delay=10:15 every=30 overlay=S symbol="digi" lat=42^37.14N lon=071^20.83W
       power=50 height=20 gain=3 comment="Chelmsford MA"
# PBEACON delay=20:15 every=30 overlay=S symbol="digi" lat=42^37.14N lon=071^20.83W
       power=50 height=20 gain=3 comment="Chelmsford MA"

#
# Modify this for your particular situation before removing
# the # comment character from the beginning of the lines above.
#
```

Remove the "#" comment character from the beginning of the "PBEACON" lines. Make necessary adjustments to latitude, longitude, comment, etc. These options, and more, are described in the User Guide.

Note that each position beacon command must be on a single line. Multiple lines are shown above due to page width limitations.

6.4 Internet Gateway (IGate)

Dire Wolf can serve as a gateway between the radio network and servers on the Internet. This allows information to be retrieved from locations such as http://aprs.fi or http://findu.com. Information can optionally be relayed from the servers, through your station, and on to the radio.

First you need to specify the name of a Tier 2 server. The current preferred way is to use one of these regional rotate addresses:

• noam.aprs2.net	- for North America
• soam.aprs2.net	- for South America
• euro.aprs2.net	- for Europe and Africa
• asia.aprs2.net	- for Asia
• aunz.aprs2.net	- for Oceania

Each name has multiple addresses to achieve load balancing and resiliency. Visit http://aprs2.net/ for the most recent information. You also need to specify your login name and passcode. For example:

```
IGSERVER  noam.aprs2.net
IGLOGIN  WB2OSZ-5  123456
```

Contact the author if you can't figure out how to generate a passcode for your ham radio call.

If you want to transmit information from the servers, you need to specify the radio channel and the via path for the packet header. Example:

```
IGTXVIA  0  WIDE1-1,WIDE2-1
```

You might want to apply a filter for what packets will be obtained from the server. Example:

```
IGFILTER  m/50
```

Read more about filters available here:

http://www.aprs-is.net/javaprsfilter.aspx

Finally, we don't want to flood the radio channel. The IGate function will limit the number of packets transmitted during 1 minute and 5 minute intervals. If a limit would be exceeded, the packet is dropped and warning is displayed in red.

```
IGTXLIMIT  6  10
```

7 Receive Decoding Performance and CPU Requirements

As Dire Wolf evolved over the years, a lot of effort has gone into continual improvement of the decoder performance. How do we measure this in an objective repeatable fashion? The de facto standard is the TNC Test CD from WA8LMF. It contains recordings of almost 26 minutes with more than 1000 packets.

Rather than running it real-time, the CD was "ripped" to files so we can process it quicker and easier. The included "atest" application has the same decoder that is in Dire Wolf, it's just repackaged to read from an audio file instead of the soundcard.

In version 1.2 we tried something new to compensate for the imbalance between the two audio tones. The separate document, "**A Better APRS Packet Demodulator, Part 1, 1200 Baud**" explains this in detail. The higher performance demodulators unfortunately take more CPU time to squeeze a few more frames out of the noise. For a desktop/laptop computer, this is only a very small fraction of the available processing power.

For the "E+" demodulator, Dire Wolf is taking about 2/3 of the CPU capacity, of a model B, just to receive. Additional bursts of processing are required to transmit and communicate with attached applications. The operating system needs to talk to the soundcard and other devices. It can't keep up.

We add a new option, to divide the audio sample rate by the specified integer. The amount of CPU power required drops dramatically without too much impact on the demodulator performance. On the command line it is "-D n" and in the confugration file, it is "/n" on the MODEM command.

Decoder	atest options	Packets decoded from Track 2	Seconds to process	Comments
E	-P E	988	894	Default rate.
E	-P E -D 2	985	348	Divide audio sample rate by 2.
E	-P E -D 3	977	237	Divide audio sample rate by 3.
E+	-P E+	1008	981	Default rate.
E+	-P E+ -D 2	1005	401	Divide audio sample rate by 2.
E+	-P E+ -D 3	1006	275	Divide audio sample rate by 3. ** New default for version 1.2. **
E+	-P E+ -D 4	1001	231	Divide audio sample rate by 4.

You can change it, individually per channel in the configuration file. On a Raspberry Pi, the audio sample rate is now divided by 3 by default. The following are equivalent in version 1.2.

```
MODEM  1200
MODEM  1200  E+
MODEM  1200  E+  /3
```

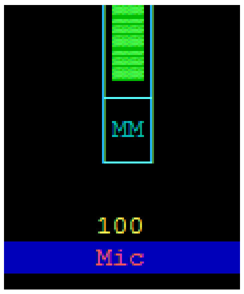
8 Troubleshooting

First check the audio gain on your input device. I've always found it necessary set it at the maximum or fairly high. In one case, with a new USB audio adapter, the initial microphone input gain was set to zero.

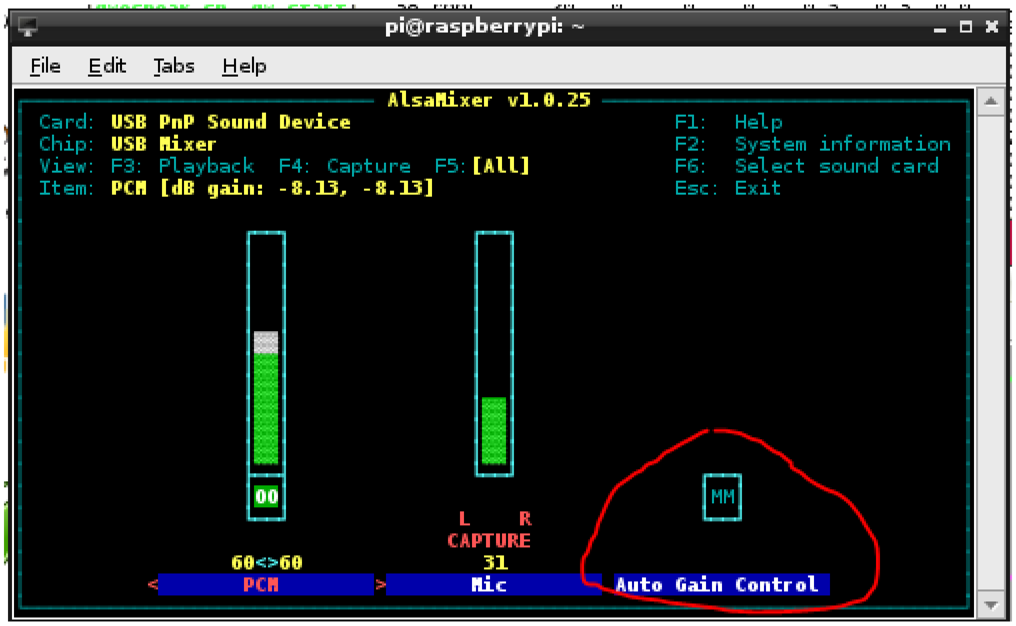
Different systems may have different applications for configuring the sound system. If using "alsamixer" follow this procedure:

- Press F6 to select the "sound card." Use up and down arrows and press Enter.
- Press F4 to set the Capture level. Use up arrow to set level to the max or fairly high.

Even if you crank up the gain to maximum, the input might be muted. Notice the "MM" here:



This means the input is muted. Press the "M" key to unmute. You might not be able to get the audio level input low enough if Auto Gain Control is enabled on the soundcard input. If you see something like this:



```
pi@raspberrypi: ~
File  Edit  Tabs  Help
                         AlsaMixer v1.0.25
Card:  USB PnP Sound Device                         F1:  Help
Chip:  USB Mixer                                    F2:  System information
View:  F3: Playback   F4: Capture   F5: [All]       F6:  Select sound card
Item:  PCM [dB gain: -8.13, -8.13]                  Esc: Exit

             00                                 MM
                          L    R
                          CAPTURE
           60<>60           31
         <   PCM     >      Mic       Auto Gain Control
```

Be sure the Auto Gain Control shows "MM" which means disabled. If it shows "00," select it with the right/left arrow keys and press the "M" key so it displays "MM."

Did you run apt-get and rpi-update to get the latest software and firmware? After many months of aggravation, it finally worked right after an update mid October 2013.

8.1 Before that time…

The SignaLink USB worked OK but I couldn't get a cheap USB audio adapter to operate properly.

The same USB audio adapter worked fine on Microsoft Windows. It worked fine with Ubuntu Linux on a regular PC. It was a miserable failure on the Raspberry Pi.

Every 100 seconds, there was a debugging message similar to these in green:

```
ADEVICE0: Sample rate approx. 39.2 k, 0 errors, audio level CH0 …
ADEVICE0: Sample rate approx. 39.8 k, 0 errors, audio level CH0 …
ADEVICE0: Sample rate approx. 39.5 k, 0 errors, audio level CH0 …
```

It reveals that the audio input system was providing less than 4000000 samples during a period of 100 seconds. We expect this to be close to 4410000 for the 44.1 kHz sample rate. Instead we find it is about 10% lower than expected. With the sample rate that far off, or 10% of the audio samples getting lost somewhere, nothing gets decoded. If no signals are decoded, look for these messages for a clue.

The USB audio device was fine on Windows and Ubunu Linux on a regular PC, so it was probably a software driver issue. After the most recent software/firmware update, it's now fine. The messages every 100 seconds show the expected number of audio samples for that time period.

If you can't copy any signals, make sure you have software/firmware no earlier than mid October 2013, crank up the audio input level, and then look for the debug messages every 100 seconds.

8.2 USB Hubs

Avoid putting a USB hub between the Raspberry Pi and the USB Audio adapter. I've received reports that exactly the same configuration works fine with the direct connection and a significant number of audio samples are lost when going through a hub. With direct connection this was observed:

```
ADEVICE0: Sample rate approx. 44.1 k, 0 errors, audio level CH0 …
```

With the hub in the middle, numbers like this were seen:

```
ADEVICE0: Sample rate approx. 42.7 k, 0 errors, audio level CH0 …
ADEVICE0: Sample rate approx. 42.8 k, 0 errors, audio level CH0 …
```

Some complain about these annoying messages but they provide valuable debugging information.

8.3 Use with AX25 kissattach

Sometimes kissattach has an issue with the Dire Wolf pseudo terminal. This shows up most often on Raspbian but sometimes occurs with other versions of Linux.

```
kissattach: Error setting line discipline: TIOCSETD: Device or resource busy
Are you sure you have enabled MKISS support in the kernel
or, if you made it a module, that the module is loaded?
```

Consult the ***Linux AX25*** section of the ***User Guide*** for more details and work-arounds.