Raspberry Pi Packet TNC,
APRS Digipeater, IGate

Version 1.6 – November 2019 – New build procedure with "cmake." Decluttering.

August 2017 – Additional comments for Raspbian Stretch.

Version 1.4 – April 2017 – Updates for Raspbian Jessie vs. Wheezy.

Observations about USB audio adapter microphone input.

Version 1.3 – February 2016

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 great 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. Dire Wolf has been tested with models 1, 2, 3, 4, and Zero. Other single board computers, such as the BeagleBone, Cubie Board, and Orange Pi Nano, are also known to work but you might have to make a few minor adjustments to the procedures here. Dire Wolf has also been tested with operating systems such as https://archlinuxarm.org/platforms/armv8/broadcom/raspberry-pi-3   but the procedures are a little different.   New users probably want to stick with the Raspberry Pi and Raspberry Pi OS (formerly called Raspbian) for the least amount of aggravation.

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

If using the UDRC interface, from NW Digital Radio, follow instructions here instead for initial setup.

https://nw-digital-radio.groups.io/g/udrc/wiki/UDRC%E2%84%A2-and-Direwolf-Packet-Modem

After that section 5 and later might be helpful.

My homebrew interface.

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. A model 1 is more than adequate.

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.

Interface built by WB0NIU.
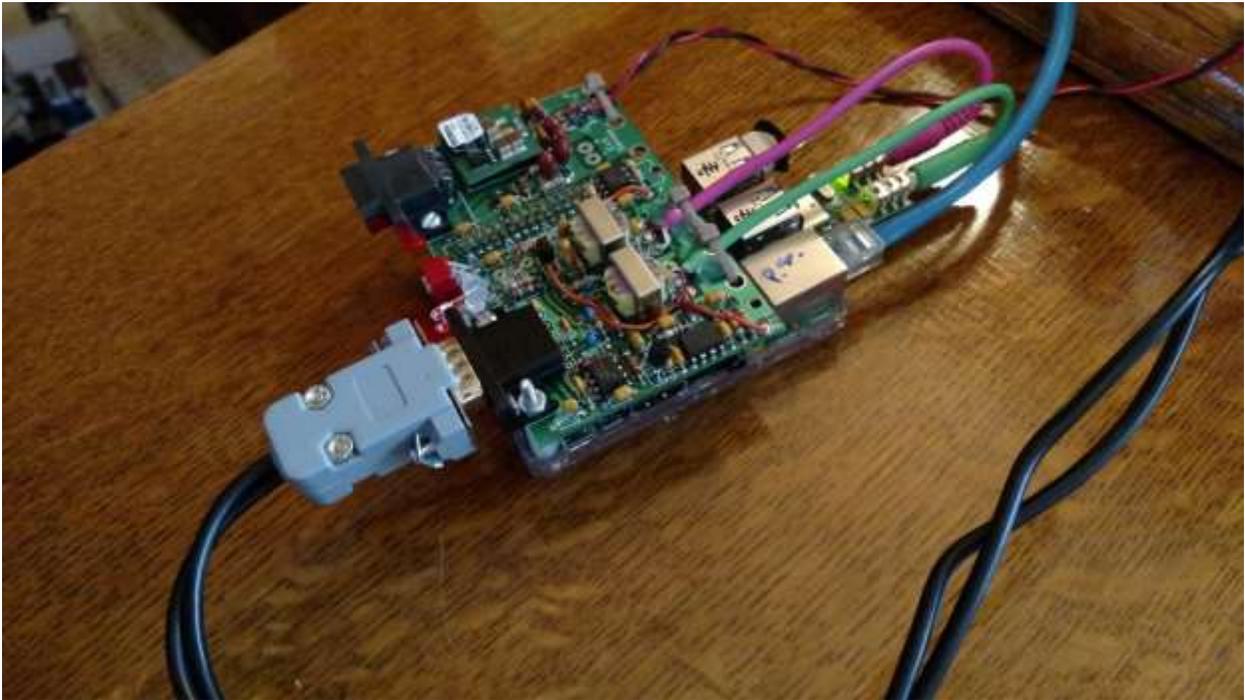
Fits on top of the Raspberry Pi.

Includes:

- 5 volt switching regulator.
- RS-232 port with voltage level converter.
- Analog to Digital converter to measure voltage.
- Temperature sensor.
- Audio transformers and PTT opto-isolator to break up ground loop between computer and radio.
- Standard connector used on most TNCs so standard radio-specific cables can be used.

1 Install Operating System

You will need an SD memory card which acts like a disk drive. A 16 GB SD memory card is fine. It will be about half full after we are done with the installation. Be sure to get a Class 10 or faster.

Raspberry Pi OS (formerly known as Raspbian) is a version of Debian Linux specially customized for the Raspberry Pi. Down load it from here and follow the installation instructions.

https://www.raspberrypi.com/software/

You can use either:

- 32 bit version or
- 64 bit version on hardware that supports it.

You can use either:

- Full version with desktop -- recommended, instructions below assume this choice.
- Lite version with no desktop -- you will need to improvise for some steps.

1.1   New System Setup & Configuration

In my case, the RPi has a monitor, keyboard, and mouse only during initial setup.   After that it is run "headless" (no monitor) in the "shack" and accessed from a PC running the VNC client software from http://www.realvnc.com

Remote access is disabled initially for security reasons. You can enable it with the configuration tool found under the Raspberry pi menu.
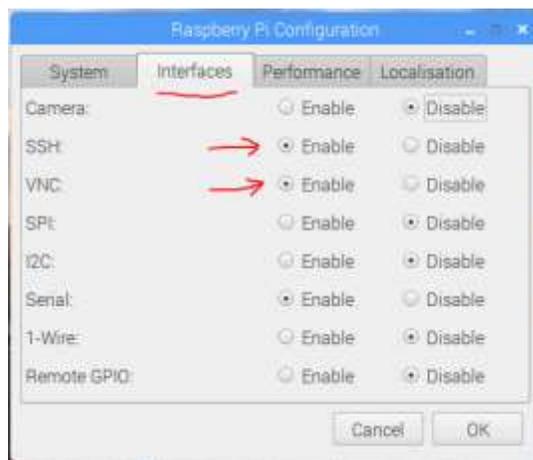
1.2   Make a backup of your SD card (optional)

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

1.3   Security Considerations

I suggest that you get everything working first with the default "pi" user account because there are fewer things that can go wrong.   Come back to this section later if you want to create a different user account.

Raspbian has a predefined user "pi" with password "raspberry" and various privileges. It can run "sudo" (do command as super user) without a password.   This is very convenient for you but also convenient for uninvited guests who might have bad intentions.

Simply changing the default password to something more obscure would go a long way towards protecting your system from intruders.

If you create another user account, and disable the predefined "pi," you will want to mimic most of the properties of the original "pi" user.   "pi" is a member of several groups which enable various privileges. In the Bullseye (Debian 11) version we find:

- adm - administration
- dialout – full access to serial ports
- cdrom – access to CD drive
- sudo – can execute any command with sudo
- audio – access to sound hardware
- video – access to video devices
- plugdev – access to removable devices
- games – access to game software
- users – standard users group
- netdev – can manage network interfaces
- input – input devices such as keyboard and mouse
- render
- netdev
- lpadmin
- gpio – General Purpose I/O pins
- i2c – I^2C
- spi - SPI

A new user account could be added like this:

```
sudo adduser john

sudo usermod –G adm,dialout,sudo,audio,video,plugdev,users,gpio,i2c john
```

I've heard of some people running direwolf as root user. Ordinary user accounts are fine as long as they have the permissions mentioned above. **Using root access more than necessary increases the chances of damage from a misbehaving application or accidentally typing the wrong thing.**

1.4   For Advanced Users

To maximize chances of success, you probably want to keep things as simple and standard as possible the first time.

After you gain more experience with Direwolf and intend to run Direwolf on a 24/7/365 basis, there are several other things that should be considered. This includes minimizing writes to the SD card, additional security through firewalls, remote management, patching, etc. All of these items are beyond the scope of this document. You can find many useful best practices listed here:

http://www.trinityos.com/HAM/CentosDigitalModes/RPi/rpi2-setup.html

2   Build & Install Dire Wolf

A fresh operating system install should include basic software development tools such as the gcc compiler and git. A few more packages will need to be installed.

2.1   Install cmake   (new for version 1.6)

```
sudo apt-get install cmake
```

2.2   Install library development packages

Install the "libasound2-dev" & "libudev-dev" packages with the following commands:

```
sudo apt-get install libasound2-dev

sudo apt-get install libudev-dev
```

Failure to install these packages might result in build errors like:

```
Could NOT find ALSA (missing: ALSA_LIBRARY ALSA_INCLUDE_DIR)

Could NOT find UDEV (missing: UDEV_LIBRARY UDEV_INCLUDE_DIR)
```

2.3   Download Dire Wolf source code

Clone the git repository and checkout the desired version. Perform these steps logged in as the predefined "pi" user or some other account you created.   It is recommended that you do **NOT** use the root account because it is too easy to accidentally trash your system. Steps requiring special privilege are prefixed by "sudo."

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

At this point you should have the most recent stable version which is probably what you want in most cases.

In some cases, you might want the latest (sometimes 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, type:

```
git checkout dev
```

2.4   Compile and Install (new procedure for version 1.6)

Compile and install the application, assuming you are in the "direwolf" subdirectory.

```
mkdir build && cd build
cmake -DUNITTEST=1 ..
make -j4
make test
sudo make install
```

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

/usr/local/bin/direwolf	The main 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
    text2tt
    ll2utm
    utm2ll
    log2gpx
    gen_packets
    aclients
    appserver
    atest
    ttcalc
    kissutil
    cm108
    dwespeak.sh
	Utilities related to APRStt gateway, UTM coordinates, log file to GPX conversion, test frame generation, TNC comparison, and a Touch Tone to Speech sample application.

/usr/local/bin/telem-balloon.pl
    telem-bits.pl
    telem-data91.pl
    telem-data.pl
    telem-eqns.pl
    telem-parm.pl
    telem-unit.pl
    telem-volts.py
/usr/local/share/doc/direwolf/
    APRS-Telemetry-Toolkit.pdf
/usr/local/share/doc/direwolf/conf/
    telem-m0xer-3.txt
    telem-balloon.conf
    telem-volts.conf
	APRS Telemetry Toolkit.

/usr/local/share/applications/direwolf.desktop /usr/local/share/pixmaps/direwolf_icon.png	Application definition with icon location, command to execute, etc. You might want to copy this to ~/Desktop.

	Icon for the desktop.
/usr/local/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/local/share/direwolf/symbolsX.txt symbols-new.txt	Descriptions and codes for APRS symbols.

/usr/local/share/doc/direwolf/
    2400-4800-PSK-for-APRS-Packet-Radio.pdf
    A-Better-APRS-Packet-Demodulator-Part-1-
        1200-baud.pdf
    A-Better-APRS-Packet-Demodulator-Part-2-
        9600-baud.pdf
    A-Closer-Look-at-the-WA8LMF-TNC-Test-
        CD.pdf
    APRS-Telemetry-Toolkit.pdf
    APRStt-Implementation-Notes.pdf
    APRStt-interface-for-SARTrack.pdf
    APRStt-Listening-Example.pdf
    AX25_plus_FEC_equals_FX25.pdf
    Bluetooth-KISS-TNC.pdf
    CHANGES.md
    Going-beyond-9600-baud.pdf
    Raspberry-Pi-APRS.pdf
    Raspberry-Pi-APRS-Tracker.pdf
    Raspberry-Pi-SDR-IGate.pdf
    README.md
    Successful-APRS-IGate-Operation.pdf
    User-Guide.pdf
    WA8LMF-TNC-Test-CD-Results.pdf
    Why-is-9600-only-twice-as-fast-as-1200.pdf
	Various documentation, mostly in PDF form.
	
	README.md is an overview.

/usr/local/share/doc/direwolf/conf/
    direwolf.conf
    sdr.conf
    telem-m0xer-3.txt
    telem-balloon.conf
    telem-volts.conf
/usr/local/share/doc/direwolf/scripts/
    dw-start.sh
	Sample configuration files and other examples.

/usr/local/share/man/man1/*	"man" pages with concise on-line help.
/etc/udev/rules.d/99-direwolf-cmedia.rules	Set group and mode of HID devices corresponding to C-Media USB Audio adapters. This will allow us to use the CM108/CM119 GPIO pins for PTT.

2.5   Get initial configuration file.

If you are upgrading from an earlier version, you will probably have a customized version of "direwolf.conf" in your home directory. You will want to make a backup copy in case it gets overwritten.

Put a copy of the "direwolf.conf" configuration file, the "dw-start.sh" script, and a few other sample files in your home directory. In theory, it should not overwrite an existing file.   (But you already made a backup copy, just to be safe. Right?)

```
make install-conf
```

Put an application startup icon on your desktop:

```
cp /usr/local/share/applications/direwolf.desktop  ~/Desktop
```

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

2.6   First Test

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

```
cd ~
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.

> Note to packagers (or those who install from a package rather than building from source):
>
> The build process attempts to use the NEON vector processing instructions, if possible, because it has a big impact on performance.
>
> If you compile on an RPi model 2 (or higher) then try to run the result on an RPI model 0 or zero, you will get an illegal instruction error.

If you want to run direwolf on a model Zero (not the Zero 2) you must compile it there. Don't compile it on a higher model and copy the result over.

3   Interface for radio

To receive we need to get audio, from the radio, into the computer.

To transmit, we need to get audio from the computer to the radio. We also need a push to talk (PTT) control.

The discussion here is abbreviated and concentrating on the Raspberry Pi. For a more complete and general discussion, see the separate file *Radio-Interface-Guide.pdf*.

3.1   Audio input / output

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: 7/7
  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
card 0: ALSA [bcm2835 ALSA], device 1: bcm2835 IEC958/HDMI
[bcm2835 IEC958/HDMI]
  Subdevices: 1/1
  Subdevice #0: subdevice #0
card 0: ALSA [bcm2835 ALSA], device 2: bcm2835 IEC958/HDMI1
[bcm2835 IEC958/HDMI1]
  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. There are many possibilities.

3.1.1   Signalink USB

The **Tigertronics SignaLink USB** is a pricey solution but easy because it contains all of the interface circuitry. It includes a VOX circuit to key the transmitter when transmit audio is present. Be sure to turn the "DLY" delay control completely counter-clockwise to minimize the time the transmitter is kept on after the audio has ended.
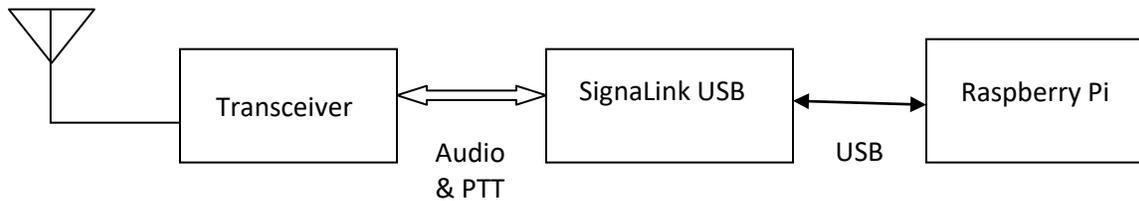
Don't try using the SignaLink USB for 9600 baud! It doesn't have enough bandwidth. See 9600 section of the **User Guide** for more details.

3.1.2 USB audio adapters

USB audio adapters are the cheapest and most popular.

Recommended:

> I've had good luck with this one http://www.amazon.com/gp/product/B001MSS6CS. It has been frequently recommended in the forums and I've never heard of anyone having trouble with it. You don't need to add drivers and it all just works.
>
> I popped one open and it had a C-Media CM119 inside. Of course, if you buy one today there is no guarantee that it will be the same.)

Here is a suggested circuit:
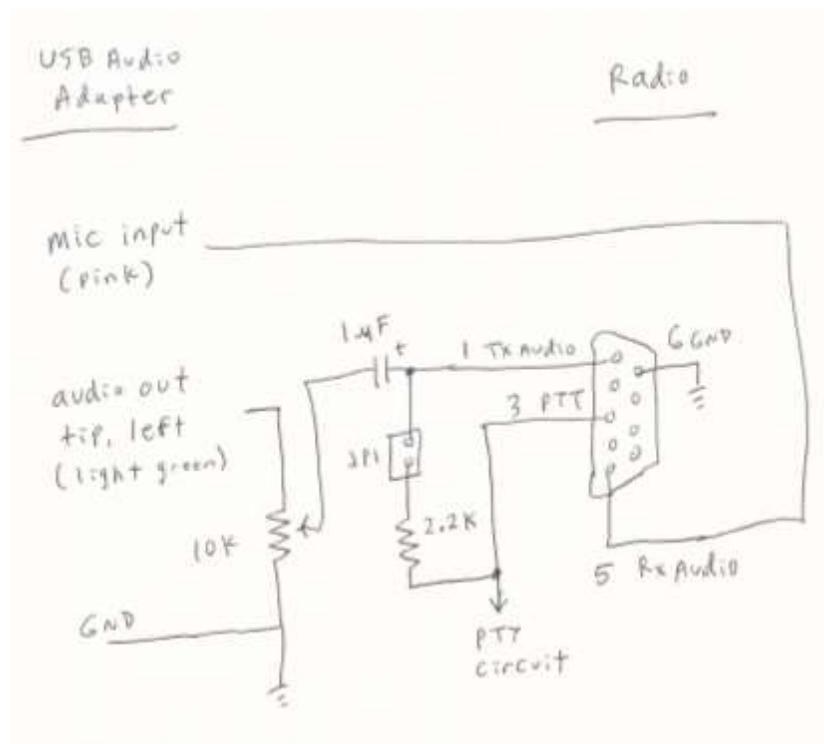
Some older hand held radios use a single pin for microphone audio and PTT. In this case, the jumper is installed so the Mic input is pulled to ground with a 2.2k resistor.

I like to use a 9 pin female D connector so it is compatible with most other traditional TNCs (e.g. KPC-3 Plus) and trackers.   Having one common connector makes it easy to mix-n-match different combinations of TNCs and Radios.   Preassembled cables are available for different types of radios from vendors such as Byonics.

- o   Pin 1   Transmit audio to radio Microphone input.
- o   Pin 3   Ground for Push to Talk (PTT).
- o   Pin 5   Receive audio from radio speaker, etc.
- o   Pin 6   Ground.
- o   Pin 7   Sometimes used for external power in.

3.1.2.1   Observations about the microphone input

Here is a sample diagram for what you might find inside of one of these USB audio adapters: http://www.qsl.net/om3cph/sb/CM108_DataSheet_v1.6.pdf See second to last page.

It shows the microphone jack tip capacitively coupled to the CM108 chip with 1 µF. The ring has a microphone bias voltage.

Here is another that is just the opposite: http://www.hardwaresecrets.com/datasheets/CM109.pdf
The tip is a DC bias voltage and the audio input is on the ring.

This http://www.repeater-builder.com/voip/pdf/cm119-datasheet.pdf has both on the ring and the tip is not connected.

I guess you just need to try both and see what works.   If you are not getting any audio input, or it is extremely weak – only from crosstalk – that might be the problem.   The printed circuit board, mentioned a couple pages later, has a jumper so you can easily try either one.

Both the Syba product mentioned above, and a different one from Adafruit, have the tip and ring connected together. They have an open circuit voltage of 4.51 volts which drops down to about half that when connected to ground through a 1.5 k resistor.

☞   **The lesson, here, is that you should use a 3.5 mm stereo (TRS) plug, not**

a mono (TR) plug for the microphone input. If you use a mono cable, the longer sleeve on the plug might short the microphone input to ground.

Finally, you might want to stick a 1 µF capacitor between the receive audio and the microphone input, due to the DC bias, but I never found it to be necessary.

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.

3.1.4   USB audio adapters with built in PTT control

The C-Media CM108 and CM119 chips are very popular for USB to Audio adapters. They have GPIO pins that we can use for the PTT signal.   This is a very tidy solution because everything goes through a single USB cable.   Dire Wolf version 1.5 added support which makes these very easy to use.

At this time, I'm aware of four commercial products using this technique:

- **DINAH**   https://hamprojects.info/dinah/
- **DMK URI**   http://www.dmkeng.com/URI_Order_Page.htm
- **RB-USB RIM**   http://www.repeater-builder.com/products/usb-rim-lite.html
- **RA-35**   http://www.masterscommunications.com/products/radio-adapter/ra35.html
- **DRA-30**   http://www.masterscommunications.com/products/radio-adapter/dra/dra30.html

There are several similar homebrew projects:

http://www.qsl.net/kb9mwr/projects/voip/usbfob-119.pdf
http://rtpdir.weebly.com/uploads/1/6/8/7/1687703/usbfob.pdf

http://www.repeater-builder.com/projects/fob/USB-Fob-Construction.pdf
https://irongarment.wordpress.com/2011/03/29/cm108-compatible-chips-with-gpio

3.1.5   DRAWS & UDRC™ interface from NW Digital Radio

DRAWS, and the earlier UDRC are high performance soundcards that fit on top of the Raspberry Pi. They can use higher audio sampling rates making them ideal for 9600 baud and above. The 6 pin mini-DIN radio connector matches the "data" connectors on many transceivers, making hook up very easy. For details, see http://nwdigitalradio.com/draws/ and   https://nw-digital-radio.groups.io/g/udrc/wiki/UDRC%E2%84%A2-and-Direwolf-Packet-Modem#Basic-Configuration

3.1.6   Fe-Pi Audio

This is another audio board with an I^{2}C / SPI interface so it doesn't tie up a USB port. This would be especially beneficial with the Pi zero which has only a single USB port that you might want to use for something else. https://fe-pi.com/ Be sure to follow the special setup instructions carefully.

3.2 Determine "card" number for audio device

After adding a suitable 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]


    …
    (We saw this earlier)
    …
```

New ☞ `card 1: Device [USB PnP Sound Device], device 0: USB Audio [USB Audio]]`
```
    Subdevices: 1/1
    Subdevice #0: subdevice #0

pi@raspberrypi:~$ arecord –l

**** List of CAPTURE Hardware Devices ****
```
New ☞ `card 1: Device [USB PnP Sound Device], device 0: USB Audio [USB Audio]`
```
    Subdevices: 1/1
    Subdevice #0: subdevice #0
```

The important thing to notice is the "card" number. It's a holdover from when audio devices were cards plugged into a computer motherboard.

3.3 Transmitter PTT

If you want to transmit, some method is needed to activate the transmitter push-to-talk (PTT) function.

Possible alternatives include:

3.3.1 RTS control line of RPi serial port

Possible, but I can't imagine why anyone would want to do this when there are better ways.

3.3.2 RTS control line of USB to RS232 adapter

Again, possible, but I can't imagine why anyone would want to do this when there are better ways.

3.3.3 VOX operation where transmitter is activated by transmit audio signal

The SignaLink USB uses this technique. It is important to set the delay to the minimum value so the transmitter will be turned off quickly after transmit audio stops.

There are several homebrew circuits using the same idea including
https://sites.google.com/site/kh6tyinterface/

The VOX built into transceivers are designed for voice and keep the transmitter on much too long after the audio has stopped. It's not a good idea to send a dead carrier because other TNCs will think the channel is clear and start transmitting. The **User Guide** has a detailed explanation.

Repeat: Using VOX built into transceivers is generally a bad idea.

3.3.4 Use one of the RPi General Purpose I/O (GPIO) pins

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? I would try to avoid those with special functions such as UART, SPI, PWM, or I^{2}C. These are my suggestions for the best choices. Note that the physical pin number on the connector, and the GPIO number (used by the software), are not the same.

For very early RPi model 1 (before the "plus" versions):

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

The later models (A+, B+, 2, 3, 4) have a larger connector with additional GPIO pins. Any of the pins listed above or these new ones 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.
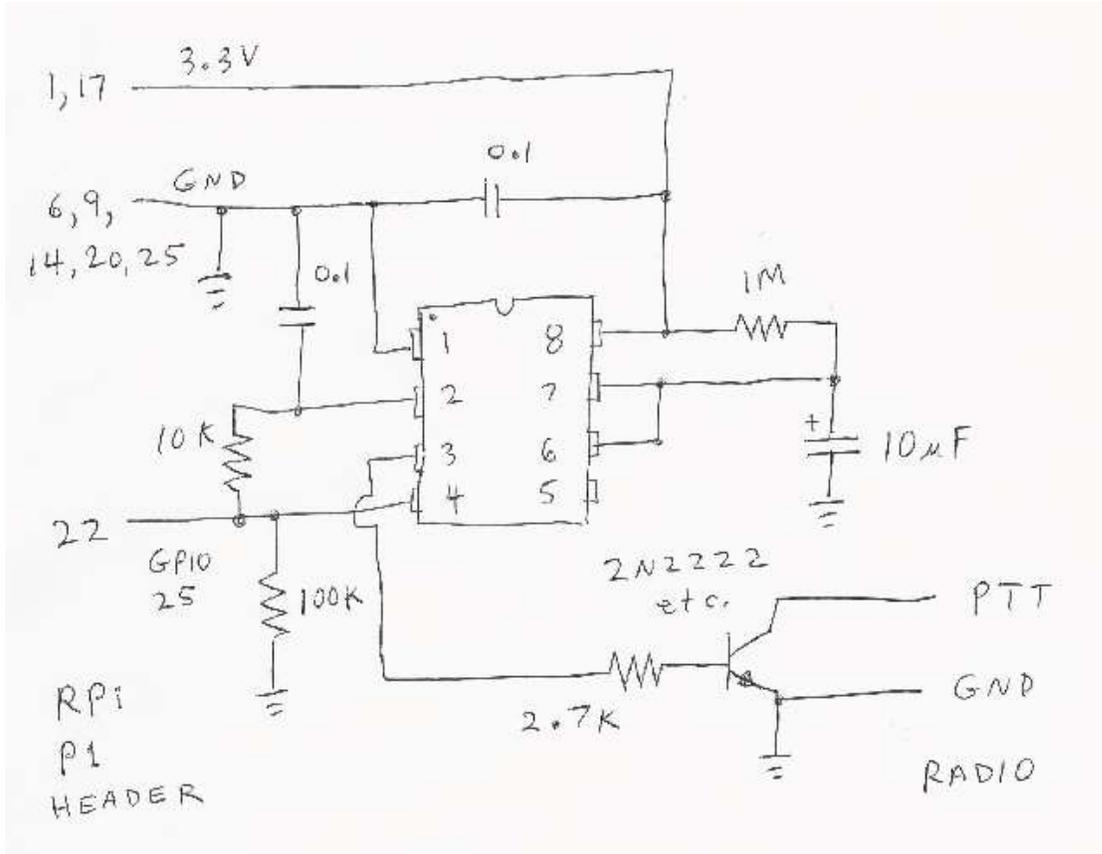
You could get by with only a resistor and transistor but a software failure could cause the transmitter to be stuck on, jamming the radio channel, annoying other people, and possibly damaging the transmitter from overheating.

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 0.01 µf capacitor from pin 4 to ground. I suspect this was caused by RF, from the transmitter, getting into the digital circuitry.

3.3.5   CM108/CM119 USB Audio Adapter GPIO pins

Many of the popular USB Audio adapters have unused general purpose input output (GPIO) pins inside. Some products use one of these to activate the push to talk (PTT) control.

- **DINAH**   https://hamprojects.info/dinah/
- **DMK URI**   http://www.dmkeng.com/URI_Order_Page.htm
- **RB-USB RIM**   http://www.repeater-builder.com/products/usb-rim-lite.html
- **RA-35**   http://www.masterscommunications.com/products/radio-adapter/ra35.html

The audio part and the GPIO part show up as different Linux devices.   The relationship between them is not obvious. The included "**cm108**" application will demystify the situation.   In your configuration file, use:

> PTT CM108

Dire Wolf will figure out which HID (human interface device) corresponds to the audio device. For more information, see *User Guide* section called "PTT with C-Media CM108/CM119 GPIO."

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.

Note that it has jumpers available so that receive audio can be sent to either the tip or the ring of the microphone input of the USB Audio adapter.

3.5 Other Alternatives

The KF5INZ "Easy Digi" interface 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

In previous operating system versions (Wheezy, Jessie);

> We previously determined the "card" number for the desired audio interface. Typically this will be 1 unless you did something to change the ordering.

> 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.

For Raspbian Stretch:

> A USB audio adapter now shows up as card 0. You might be able to take the default device and not have to worry about the ADEVICE line in the configuration file.

For Raspbian Buster:

> A USB audio adapter is back to card 1 again.

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.

When using PTT from the USB Audio adapter (e.g. **DINAH, DMK URI**, **RB-USB RIM**) simply use this:

```
PTT CM108
```

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.
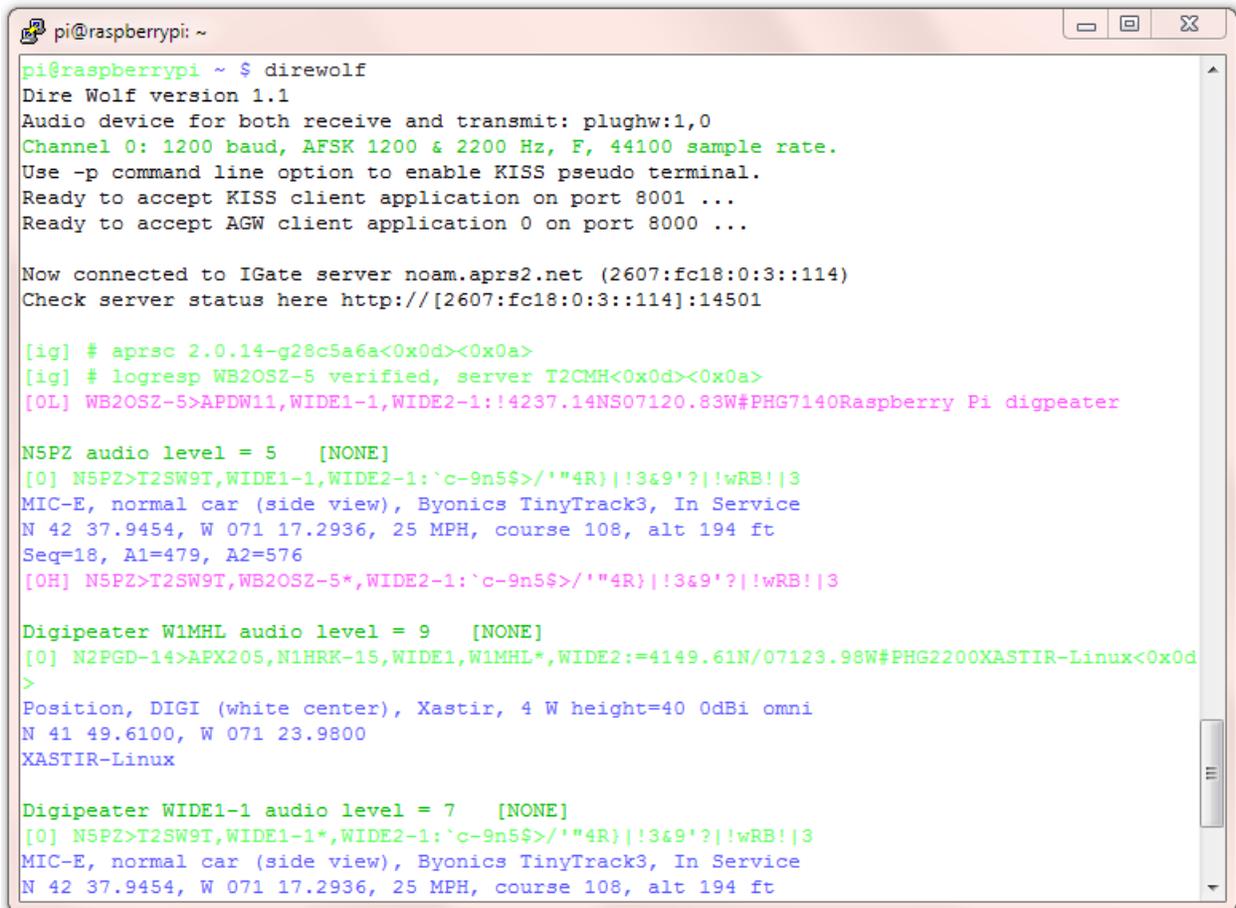
```
DCD GPIO 24
```

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. Why do people insist on doing that?

Typical results will look something like this:

```
pi@raspberrypi: ~

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.

```
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, Xastir, APRS-TW, YAAC, SARTrack, Outpost PM, 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.
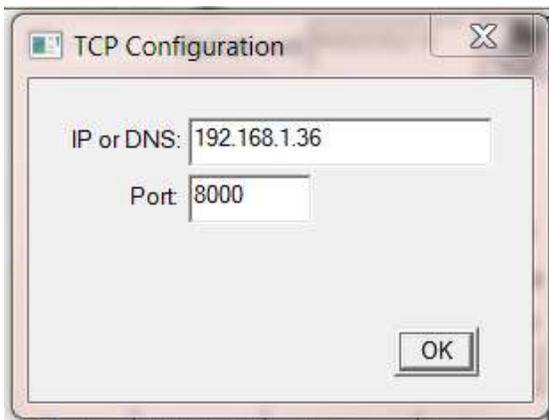
From the Configure menu, pick Ports → New Port…

Choose type of AGW and give it a meaningful name.
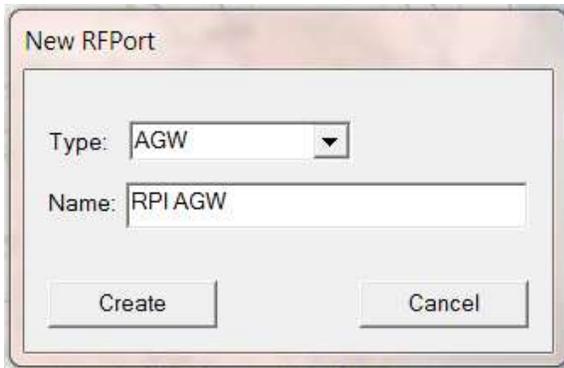
Click on Create and pick port type of TCP/IP.

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

6   Other Common Configuration Changes

6.1   Automatic Startup

You will probably want your TNC / application server / digipeater to start up automatically after a reboot. This often causes confusion as there are many ways to do this.   You will find other approaches in the forums but here is one solution which should be usable for many use cases.

The important thing to remember is that direwolf writes a lot of information to "stdout."   This information is valuable, for troubleshooting, and really needs to go somewhere. If you simply try starting direwolf from /etc/rc.local you will probably be disappointed.   It is better to run direwolf as an ordinary user, rather than root, so there is less chance of damaging your system if something goes wrong.

If you followed the installation steps above, you should have a file named dw-start.sh in your home directory. If it is not there, copy it from /usr/local/share/doc/direwolf/scripts/dw-start.sh

This script will test whether direwolf is already running. If not, it will be started.

My suggestion is to run this script from cron so if direwolf stops running for any reason, it will be automatically restarted.   Use the "crontab –e" command and add a line like this, substituting you own user name if it is not pi:

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

The line above will run the /home/pi/dw-start.sh script once per minute. Dire Wolf will be started automatically if not running already. If a previous instance of Dire wolf crashes, or is terminated for any other reason, it will be restarted within a minute. A log of restarts can be found in /var/tmp/dw-start.log.

 dw-start.sh will try to determine if you have a graphical desktop and select either GUI or CLI mode. You can override this by looking for "RUNMODE=AUTO" near the top of the dw-start.sh file and modifying as described in the script's comments.

If you are running a "lite" version of Linux without the graphical desktop, popping up a GUI window is not an option and will give Xsession errors. In this case, the dw-start.sh script will use "screen" to run direwolf in a detached session.

If the "screen" utility is not already installed, add it with "sudo apt-get install screen" on Debian-based distributions.

Later you can use

```
    screen –list
```

to get a list of sessions and attach to an existing session with

```
    screen –D  –r direwolf
```

You can again "detach" from the direwolf screen session with

```
        control-a
        then "d"
```

at any time and direwolf will continue to run.

Alternatively, <u>you could use systemd to start it up</u>.

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-
```

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 the digipeater name. "WB2OSZ-5*" in my case. Yours will be different. The "*" indicates that it has been used up and won't be digipeated again.

The *User Guide* describes what the configuration means. *APRS-Digipeaters.pdf* goes into much more detail about how APRS digipeaters work.

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.

Refer to the *User Guide* for more details on beaconing options available.

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
```

For more details about setting up an IGate station, see the related document: **Successful-APRS-IGate-Operation.pdf**

7   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:
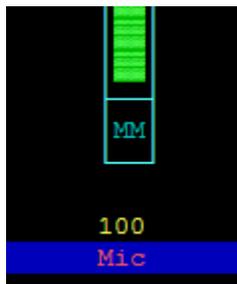
This means the input is muted. Press the "M" key to unmute. You might not be able to get the audio level input low enough if Auto Gain Control is enabled on the soundcard input. If you see something like this:

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."

If you don't get any audio out, check to make sure that it is not muted (showing MM).

More tips for troubleshooting USB audio adapters can be found here:

https://learn.adafruit.com/usb-audio-cards-with-a-raspberry-pi/instructions

7.1   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 …

These troubleshooting messages are enabled with the "-a" command line option.