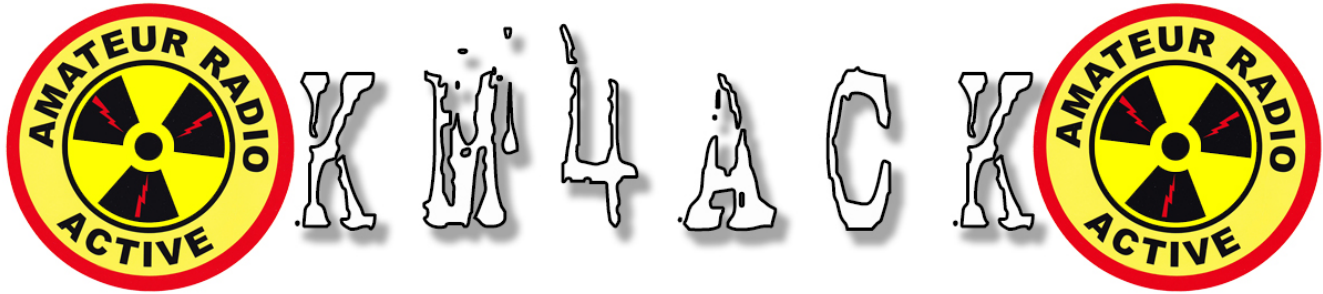


## Pi 4 Build Notes



[YouTube](#) | [Webstore](#) | [Patreon](#) | [Instagram](#) | [Twitter](#) | [Tip Jar](#) | [Linux 101](#)

## PI 4 Build Notes Revised 20200225

Revisions listed at bottom of page.

### | **Foreword**

Occasionally, the page will render in pure markdown, especially on slow internet connections. If that happens, try refreshing the page.

If you run into an issue, jump over to the [forum](#) and we will try to help you out.

This tutorial was written using a [Raspberry Pi 4](#) running Raspbian Buster (Desktop Only). It should work fine on a Pi 3 running Buster.

If you are brand new to Linux and the Raspberry Pi, I offer a [Linux 101 for Ham Radio](#) class that will cover the basics of the terminal window and take the mystery out of the command line interface. The video runs around 45 minutes.

Keep in mind that this is my **personal** setup. You may choose to do things differently or have different needs that I do. Use what you want and discard the rest. Keep in mind though that some sections are dependent upon other sections. If you are brand new to Linux, you should follow each section in order until you get a basic understanding. Some things that I download such as the autohotspot script are available on my [github](#). Other useful scripts not covered in this document are there as well. Be advised that you should [backup your pi](#) between each step just to be safe. This allows you to only go back one step should something go wrong instead of having to start completely over.

What isn't included in this document is detailed instructions for configuring each application for your radio/sound card setup. This tutorial is meant to get the base built and applications installed. The only HF rig I own is the 857D so it is near impossible for me to answer questions concerning configuration for radio X or sound card Y if I don't own those. The best place to search for answers is in the forums run by the author(s) of the particular application. Links to many of these can be found in the reference section at the bottom of this page.

Notes in this document are in regular text. All code to be entered in the terminal window will be in code blocks. (Grey boxes). You may have to scroll right in the code block to see all of the code. Installation steps are in

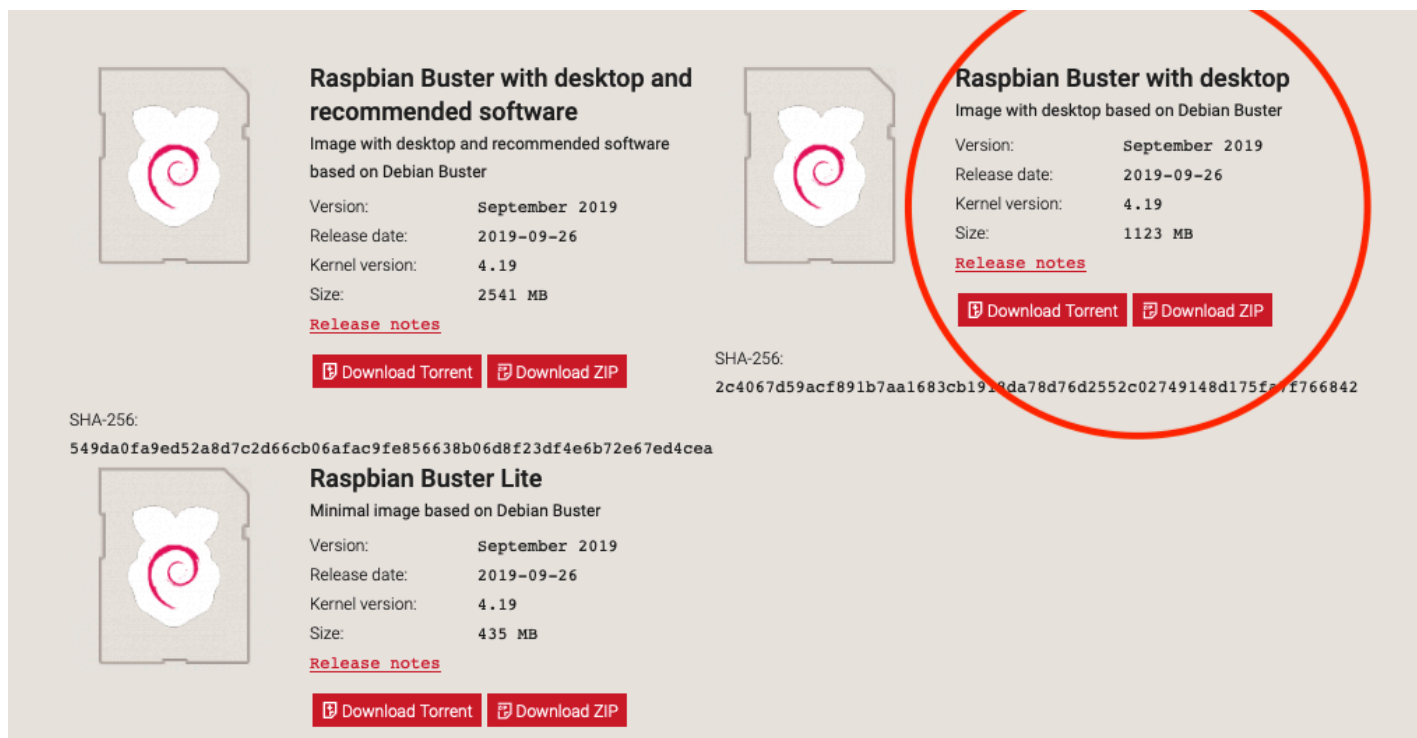
chronological order for the pi 4 setup. Notes/thoughts will be at the beginning of each section.

A lot of time has been invested in this page. If you find it useful please consider dropping a buck or two in the [tip jar](#) or support the channel through [Patreon](#)

# | Initial Setup

## [Reference Video](#)

FIG 1



**DO NOT USE NOOBS** for creating the card. Learn to flash the SD card

without taking shortcuts! There are plenty of [videos on YouTube](#) where you can learn. This will keep your card size to a minimum and we know exactly what it going on the card. Also, typically it is a waste of resources to install the full blown version of Buster. The "desktop only" version (see fig 1) is the one I use. It will keep extra junk off of our system and still give us a familiar desktop environment to work with. Let's keep our system as lean & mean as possible! I recommend this [USB>SD Card adapter](#). It will come in handy not only for burning your pi image to the SD card initially but it is also the one I use when it is time to backup the pi. I have [2 SD cards](#) for every pi. One that is in the pi currently and a second that contains my latest backup image.

The SSH and VNC stuff listed below can be skipped for now if you choose to connect a monitor, keyboard, and mouse to your pi. Since I always run [headless](#) and access my pi from my Mac when in the shack and with a tablet in the field, these steps are critical to my setup. If you plan to run your pi headless, I recommend completing the SSH & VNC steps now.

**NOTE: You will need to plug your pi into your router via a cat5 cable and know the pi's IP address.** Finding your pi's IP address on your network is beyond the scope of this document. Google is your friend :)

- Flash SD card [Raspbian Buster Desktop Only 2019-09-26](#) See Fig 1
- [Enable SSH](#)
- Insert SD card & boot the Pi
- SSH into the pi. See [SSH video](#)
  - Default user=pi password=raspberry
  - Turn on VNC
  - Set screen resolution

- Reboot
- Use [VNC viewer](#) on your PC to connect to pi
- Complete setup wizard
  - Set Local Info (Language, Keyboard, Time Zone)
  - Change Password
  - Connect to Wifi
  - Update Software
  
- Change Hostname (This step can be skipped)

After the setup wizard is complete, open a terminal window and verify everything is up to date with

```
sudo apt-get update  
sudo apt-get upgrade
```

## | ***Install Hotspot***

Reference:[Easy as Pi Hotspot Video](#)

This step allows us to run headless in the field when working portable. I also use this in my Jeep for my mobile pi. The pi will create it's own hotspot that we can connect to with a wireless device such as a phone or tablet. It can also be set to connect to a known wifi network when the network becomes

visible.

```
cd ~/Downloads
wget https://raw.githubusercontent.com/km4ack/pi-scripts/master/autohotspotN-setup
sudo chmod +x autohotspotN-setup
sudo ./autohotspotN-setup
```

Follow the onscreen instructions when prompted during the install.

```
reboot
```

## *Test Section for Hotspot*

Let's test the new hotspot. First run

```
sudo nano /etc/wpa_supplicant/wpa_supplicant.conf
```

Look for the line that begins with "ssid=" What follows the '=' sign is the name of your wifi in your home/shack. Add a '1' to the end of the line before the closing quotation mark. So if you line looks like ssid="MY-WIFI" change it

to be `ssid="MY-WIFI1"` If you see your current SSID in multiple places inside the file, be sure to mod each with '1'. To exit the nano editor, press 'ctrl+x' on the keyboard, then 'y', then 'enter'. At this point your pi will no longer see your home/shack wifi and should create a hotspot when we run

```
sudo /usr/bin/autohotspotN
```

Grab your cell phone and take a look at the available wifi connections. You should see one called "RPiHotspot". Connect to it using the password that you setup just a few minutes ago when installing the script. Once you know you can connect, we can proceed. Run

```
sudo nano /etc/wpa_supplicant/wpa_supplicant.conf
```

Remove the '1' we put at the end of the line that begins `ssid=`. Exit nano editor, press 'ctrl+x' on the keyboard, then 'y', then 'enter'. Now run

```
sudo /usr/bin/autohotspotN
```

You should see that your pi has once again connected to the wifi in your home/shack.

## Hotspot automation

Let's automate the process so that the pi will check every 5 minutes to see if a known SSID is in range. If so, connect to it. If not, create the hotspot. Note: the hotspot script will be run at boot even if you don't follow the commands below. Following the commands below, however, will allow it to change modes automatically without a reboot.

```
crontab -e
```

If this is the first time you are using cron, you will need to pick your editor. I use nano and this tutorial will reference the nano editor anytime we need to make a mod to a text or config file. Scroll to the bottom of the crontab by using the down arrow on your keyboard. Then paste the following lines. The first is just a comment as indicated by the "#" at the beginning of the line. The second line is the command.

```
#run auto hotspot script every 5 minutes
*/5 * * * * sudo /usr/bin/autohotspotN > /dev/null 2>&
1
```

To exit the nano editor, press 'ctrl+x' on the keyboard, then 'y', then 'enter'



One last thing to do. I like to modify the hotspot name that appears on your wireless device. Follow these instructions to customize yours. Run

```
sudo nano /etc/hostapd/hostapd.conf
```

Look for the line `SSID=RPiHotspot` and change the hotspot name to be what you want. Be sure to leave the `SSID=` at the beginning of the line. Everything beyond the `'=` is the hotspot name. Don't use spaces in the name. The line should look like this `SSID=NEW-HOTSPOT-NAME`

Save and close nano as before.

Refer back to the test section above and double check that it works as expected and then [backup your pi](#)

## | ***GPS Install***

### [Manual Install Video](#)

If you only plan to work with your pi while it is connected to the internet, the GPS install isn't necessary as the pi will get the correct time from the internet.

**CAUTION: If using a Pi 4, only plug the GPS into one of the USB 2.0 ports. Do NOT use the USB 3.0 (Blue) ports.** Otherwise the time may be

off by a couple of seconds and applications such as JS8Call and FT8 will not work correctly.

This will install and configure the [GPS Dongle](#) to enable you to get the current time when off-grid. The raspberry pi doesn't have a real time clock built in. This means that without a GPS or add on real time clock, the raspberry pi will NOT keep accurate time when it is powered down. The script below was written to accommodate this [GPS Dongle](#). It may work with other USB GPS units but hasn't been tested. Having accurate time is critical to applications like JS8Call & FT8. Go ahead and plug in your GPS Dongle now.

**It may take 10 or 15 minutes or longer for the GPS to lock onto satellites the first time it is connected to the pi.**

If using the GPS dongle listed above, you will know that it has a lock when the green light starts flashing. After 10-15 minutes if you don't have the green flashing light on the GPS dongle, try moving the unit close to a window.

```
cd ~/Downloads
wget https://raw.githubusercontent.com/km4ack/pi-scripts/master/gpsinstall
sudo chmod +x gpsinstall
sudo ./gpsinstall
```

Be sure to reboot at the end of the gpsinstall script as directed.

## **|** *GPS Test*

To verify that everything is working correctly, run the following

```
systemctl is-active gpsd
systemctl is-active chronyd
```

Both should return "active". To see your time sources, run

```
chronyc sources -v
```

Finally, to see a graphical representation of the GPS date, run

```
cgps
```

Press ctrl+c when done to exit

[Backup your pi](#)

## **|** ***FLRIG***

I use FLRIG to control the Yaesu 857D. FLRIG will also interface with other apps including JS8Call, FT8, and FLDIGI and allow us to control the rig from inside each of the apps. As of the time of writing the repositories have almost the very latest version of the app. This saves time as we can download from the repositories and not have to compile from source. Here is a [list](#) of supported transceivers.

```
sudo apt-get install -y flrig
```

Fire up FLRIG after install and go ahead and configure it for your rig. You will need a cable such as the [CT-62 Cable](#) connected between the 857D and the pi. This same cable can be used with the 857D when programming it with CHIRP.

[backup your pi](#)

## | **bashrc**

*NOTE: A couple of other sections in this document are dependent on making this mod to the bashrc file. If you skip this section, other sections will NOT work correctly.*

A little explanation is in order here. Ever wonder why you can type some commands from any directory in the terminal window and they work (ie.

sudo)? Yet for other commands you need to be in the specific directory for the command to work. Well, this has to do with your path in Linux. If you want to see what is included in your current path run

```
echo $PATH
```

and it will return all of the directories in your current path. If the command you are entering resides in any one of those directories, it can be run from anywhere on the system.

I modify the path to include a directory that I keep all of the scripts I have written or downloaded. This way I can run those scripts from anywhere on the system. First, let's create the directory for our scripts.

```
mkdir ~/bin
```

Now let's modify the bashrc file to include our new directory.

```
nano ~/.bashrc
```

Paste the following line into the file at the very bottom.

```
#Add path where my scripts reside
export PATH=$PATH:$HOME/bin
```

To exit the nano editor, press 'ctrl+x' on the keyboard, then 'y', then 'enter'

You will need to close all terminal windows and reopen for this to take effect.  
Now if we run

```
echo $PATH
```

you should see a new directory in your path. "/home/LINUX-USER-NAME/bin"  
On my system I am currently still using the default pi username. If you are still using pi, you will see "/home/pi/bin" . Now any file that we place inside the new directory (~/.bin) can be run from anywhere on the system.

## | ***Pat Winlink***

[Installing Pat Winlink Part One](#) | [Installing Pat Winlink Part Two](#)

[Pat Winlink](#) is a cross platform application that allows us to utilize the Winlink system on our pi. Since this is considered by many as a mission critical application, let's get it going on our pi. As of the time of this writing

the current version of Pat is 0.8.0.

```
cd ~/Downloads
wget https://github.com/la5nta/pat/releases/download/v
0.8.0/pat_0.8.0_linux_armhf.deb
sudo dpkg -i pat_0.8.0_linux_armhf.deb
```

At this point you will need to configure Pat Winlink. I have a [video](#) that will walk you through the process but you can open the config file with

```
pat configure
```

### **|** *SYSTEMD (autostart Pat at boot)*

Now let's setup Pat to start at boot. Note that if you have changed your user name to something other than pi, you will need to change the commands below to match your user name.

```
sudo systemctl start pat@pi
sudo systemctl enable pat@pi
reboot
```

After reboot, let's verify that pat is running

```
pidof pat
```

You should get a number returned. This is the process id number of pat and indicates that pat is running. Now open the browser on your pi. If you followed my configuration directions for Pat, your port number should be 5000. If so, you will navigate to 127.0.0.1:5000 If you used another port number in the pat configuration, use that port number instead of 5000. The Pat mailbox screen should now be open. Click action>connect In the new window, choose telnet for the alias in the top left dropdown box and click connect. Watch the black box at the bottom of the screen and verify that the connection is successful. *Pro tip: Bookmark this page so it is easy to access in the future.*

Now is a great time to [backup your pi](#)

## | **ARDOP & ARDOP-GUI**

The ARDOP modem is needed in order for Pat Winlink to communicate over HF. This section will walk you through the install.

### | **ARDOP**

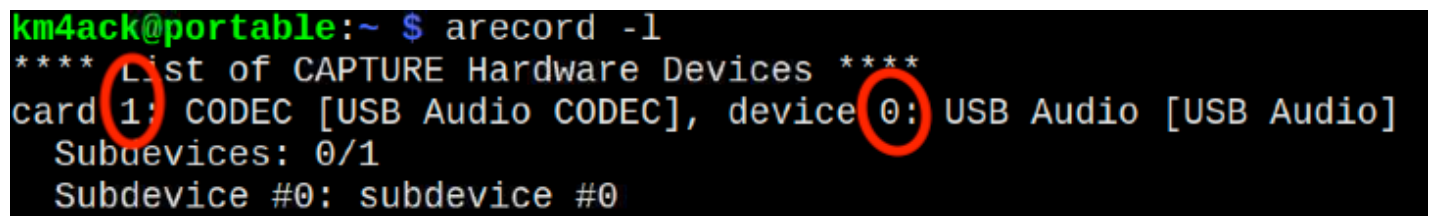


```
mkdir ~/ardop
cd ~/ardop
wget https://www.cantab.net/users/john.wiseman/Downloads/Beta/piardopc
sudo chmod +x piardopc
```

That's it for getting ARDOP installed. So how do we use it? First we need to find our sound card information. Make sure your USB sound card is plugged into the pi and run

```
arecord -l
```

FIG 2



```
km4ack@portable:~ $ arecord -l
**** List of CAPTURE Hardware Devices ****
card 1: CODEC [USB Audio CODEC], device 0: USB Audio [USB Audio]
  Subdevices: 0/1
  Subdevice #0: subdevice #0
```

Note the card number and device number for the USB Audio CODEC. In FIG 2 this info is 1 and 0. If your card/device numbers are different than what is shown in FIG 2, then substitute your numbers in the commands below after plughw: You need to change it in both instances. To start the ARDOP modem

```
cd ~/ardop
./piardopc 8515 plughw:1,0 plughw:1,0
```

Leave the terminal window open and open the browser. Navigate to the Pat Mailbox the same way we did in the Pat section. Use the menu to start you ARDOP connection.

Once you are finished with your Winlink session, close the browser. In the terminal window where piardopc is running, press ctrl+c to stop the ARDOP modem.

## *ARDOP-GUI*

### [ARDOP-GUI demo](#)

The ARDOP-GUI will give you a graphical representation of your ARDOP connections. You should still be in the ~/ardop directory. If so the first command in this group isn't needed but won't hurt anything if you run it.

```
cd ~/ardop
wget https://www.cantab.net/users/john.wiseman/Downloads/Beta/piARDOP_GUI
sudo chmod +x piARDOP_GUI
ln -s ~/ardop/piARDOP_GUI ~/Desktop/ARDOP-GUI
```

You should now have an icon on your desktop that you can double click to start ARDOP-GUI. Choose 'execute' when presented with the option. The first time starting the ARDOP-GUI, you will need to enter 'local' for the host and '8515' for the port. Now go ahead and close the app. Configuration is complete. Whenever you start a Pat Winlink ARDOP connection, you can double click on the ARDOP-GUI to see a graphical representation of the connection during the Winlink session.

## | ***Find ARDOP***

### [How to Use FindARDOP](#)

Find ARDOP is a script that I have written that will download the latest list of ARDOP stations from <https://winlink.org> Using this script and a cron job, this will update the list automatically every night. It will also give you a way to interact with the information in a useful way. The install assumes that you have followed the bashrc section above. Install with the following commands

```
cd ~/bin
wget https://raw.githubusercontent.com/km4ack/pi-scripts/master/ardop/findardop
sudo chmod +x findardop
cd ~/ardop
wget https://github.com/km4ack/pi-scripts/raw/master/ardop/grid-map.pdf
sudo apt-get install -y evince
```

```
ln -s ~/bin/findardop ~/Desktop/findardop
```

To start FindARDOP, double click the icon on the desktop. When prompted, choose "Execute in Terminal" Note: The first time the script is run it will take a few minutes to start as it needs to download the list for the first time. After the first start, it will bring you into the menu almost immediately.

The script is best viewed when the terminal is run full screen. Also, if your font is too large, it may wrap each station info into two lines making it much harder to read the results. Now let's automate the download to happen each night at 0200.

```
crontab -e
```

At the bottom of the file enter the following two lines. The first is a comment and the second is the actual command. I like to leave comments for each command in cron so that I can remember later what each command is doing.

```
#Download ardop list from winlink.org each night at 02  
00  
00 02 * * * $HOME/ardop/getardoplist
```

To exit the nano editor, press 'ctrl+x' on the keyboard, then 'y', then 'enter'

## | **Hamlib (Rig Control)**

### [Rig Control Video](#)

Although most often I use FLRIG to control the 857D, sometime we want direct rig control without using FLRIG. I use this for many custom scripts that I have written. Hamlib gives me a way to interact with the radio from a bash script. As of the time this document was written, 3.3 is the latest version.

```
cd ~/Downloads
wget https://sourceforge.net/projects/hamlib/files/hamlib/3.3/hamlib-3.3.tar.gz
tar -xzf hamlib-3.3.tar.gz
rm hamlib-3.3.tar.gz
cd hamlib-3.3
./configure
make
sudo make install
sudo ldconfig
```

## | **Packet Tools (Required for 2M packet**

## | *with Pat Winlink)*

This section will install all of the tools needed for 2M packet work with Pat Winlink. These same tools may be used for other tasks well such as a Winlink Gateway, APRS igate, or APRS digipeater.

## | *Direwolf*

Direwolf is a software TNC. It can be used stand alone for many applications or in conjunction with other applications like Pat Winlink for email or YAAC for APRS.

```
cd ~
git clone https://www.github.com/wb2osz/direwolf
cd direwolf
sudo apt-get install -y libasound2-dev
make
sudo make install
make install-conf
```

Let's get a very basic configuration done for direwolf. This assumes that you are using a USB sound card. (Signalink or similar) First, you will need your sound card number and device number. See FIG 2

```
arecord -l  
cd ~  
nano direwolf.conf
```

Scroll through the file and find

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

First remove the "#" symbol at the beginning of the line. Next, if your sound card number and device number is different than 1,0; you will need to change this line to match.

Continue scrolling down the file until you locate

```
MYCALL=NOCALL
```

and change NOCALL to your call sign. Finally, a few lines below your call sign, you will find

```
MODEM 1200
```

```
#MODEM 300  
#MODEM 9600
```

Be certain that there is no "#" in front of MODEM 1200. Press 'ctrl+x', then 'y', then 'enter' to save and exit. This completes the simplest of configuration and should be good enough to get you started.

## AX25

AX.25 (Amateur X.25) is a data link layer protocol originally derived from layer 2 of the X.25 protocol suite and designed for use by amateur radio operators

```
sudo apt-get install -y ax25-tools  
sudo nano /etc/ax25/axports
```

Add the following two lines to the bottom of the file for use with Pat Winlink during 2M packet connections. **BE SURE NOT TO LEAVE BLANK LINES IN THIS FILE** Replace NOCALL with your call sign.

```
wl2k NOCALL 1200 255 7 Winlink
```



Exit nano by pressing "ctrl+x", then "y", and then "enter"

```
sudo apt-get install -y ax25-apps
reboot
```

## *Pat Winlink Test*

To start the tools for 2M packet connections, open a terminal window and enter

```
direwolf -p
```

Now press ctrl+t on the keyboard to open a new tab in the terminal window. In the new terminal window enter

```
sudo kissattach /dev/pts/1 wl2k
sudo kissparms -c 1 -p wl2k
```

Now open the browser, navigate to the Pat mailbox and proceed with your packet connection.

# THIS COMPLETES THE BASE SETUP WITH WINLINK.

Each of the following may be installed at your descretion

## | *JS8Call*

### [Install JS8Call](#)

```
cd ~/Downloads
wget http://files.js8call.com/2.1.1/js8call_2.1.1_armhf.deb
sudo dpkg -i js8call_2.1.1_armhf.deb
```

You may see dependency errors after completing the steps above. Fix the errors with

```
sudo apt-get --fix-broken install
sudo dpkg -i js8call_2.1.1_armhf.deb
```

# ***MOIAX Applications***

These two apps were written by Mark, MOIAX. Both are fantastic add ons for JS8Call. One will help us interface with the GPS and the other will simplify sending different messages with JS8Call. If you use JS8Call, these two apps are MUST HAVES! NOTE: The GPS Install section must be complete before the GPS application will work.

## ***MOIAX GPS Tool***

### [JS8Call GPS Utility](#)

```
pip3 install gps
pip3 install maidenhead
cd ~/bin
wget https://raw.githubusercontent.com/m0iax/js8calltools/master/js8callgpsUI.py
wget https://raw.githubusercontent.com/m0iax/js8calltools/master/gps_listener.py
sudo chmod +x js8callgpsUI.py
ln -s ~/bin/js8callgpsUI.py ~/Desktop/JS8-GPS-Tool
```

## ***MOIAX JS8Call Messenger***

### [JS8Call Messenger](#)

```
cd ~/bin
pip3 install psutil
wget https://raw.githubusercontent.com/m0iax/js8call_aprsmessaging_interface/master/aprs_msgJS8Call.py
sudo chmod +x aprs_msgJS8Call.py
ln ~/bin/aprs_msgJS8Call.py ~/Desktop/JS8Call-Messenger
```

To start either app, double click on the icon on the desktop and choose "Execute" when given the option.

## | **WSJTX**

### [FT8 Install Video](#)

This is by far one of the most often requested applications for the raspberry pi. While I don't use FT8, I do use WSPR for checking band conditions, testing antennas, and doing antenna comparisons. As of the time of writing, the current version is 2.1

```
cd ~/Downloads
wget --no-check-certificate https://physics.princeton.edu/pulsar/k1jtx/wsجتx_2.1.2_armhf.deb
```

```
sudo dpkg -i wsjtx_2.1.2_armhf.deb
```

You may see dependency errors after completing the steps above. Fix the errors with

```
sudo apt-get --fix-broken install  
sudo dpkg -i wsjtx_2.1.2_armhf.deb
```

Look for WSJTX in the main menu to start the application.

## | *Chirp*

### [Install Chirp](#)

Chirp is an application that allows you to program many radios. To see which radios are supported see the list on [this page](#)

```
sudo apt-get -y install python-gtk2 python-serial pyth  
on-libxml2  
mkdir ~/chirp  
cd ~/chirp  
wget https://trac.chirp.danplanet.com/chirp_daily/LATE
```

```
ST/chirp-daily-20200521.tar.gz
tar -xzf chirp-daily-20200521.tar.gz
cd chirp-daily-20200521
sudo python setup.py install
```

## | **Gpredict**

Gpredict is a real time satellite tracking and orbit prediction program for the Linux desktop. It uses the SGP4/SDP4 propagation algorithms together with NORAD two-line element sets (TLE).

```
cd ~/Downloads
wget https://launchpad.net/~gpredict-team/+archive/ubuntu/ppa/+files/gpredict_2.2.1-ubuntu16.04~ppa2_armhf.deb
sudo dpkg -i gpredict_2.2.1-ubuntu16.04~ppa2_armhf.deb
```

As before, this may fail because of dependencies. Fix with

```
sudo apt --fix-broken install
sudo dpkg -i gpredict_2.2.1-ubuntu16.04~ppa2_armhf.deb
```

Look for the program in the main menu.

## | **FLDIGI/FLARQ**

As fo the time of this writing, the near latest version of FLDIGI is in the repositories. This allows us to save time by not compiling from source. This will also install FLARQ.

```
sudo apt-get install -y fldigi
```

## | **YAAC**

YAAC (Yet Another APRS Client) can be used to interface with direwolf when creating an igate or digipeater.

```
cd ~/Downloads
wget https://www.ka2ddo.org/ka2ddo/YAAC.zip
sudo apt-get install openjdk-8-jre librx-tx-java
cd ~
mkdir YAAC
cd YAAC
unzip $HOME/Downloads/YAAC.zip
```

To start the program

```
java -jar YAAC.jar
```

## | ***Xastir***

Xastir is an alternative to YAAC. Both perform the same basic functions but YAAC seems to be updated more often.

```
sudo apt-get install xastir
```

## | ***Conky***

[Conky Video](#)

## | ***Pat Menu***

[Pat Menu Install-Configure](#)



# | **HotSpot Tools**

[HotSpot Tools](#)

# | **More Applications**

[DL1GKK Raspberry Pi Site](#)

# | **Reference**

- [JS8Call](#)
- [JS8Call Forum](#)
- [JS8Call Documentation](#)
- [Raspberry Pi 4 Ham Radio Forum](#)
- [Pat Winlink](#)
- [Pat Winlink Forum](#)
- [Direwolf Documentation](#)
- [Direwolf Forum](#)
- [WSJTX](#)
- [John Wiseman's Download Page](#)
- [Hamlib](#)
- [YAAC](#)
- [Linux Ham](#)
- [Conky Documentation](#)

# | **Revisions**

20200530 add hotspot tools update pat menu to v2 update chirp links add forum link for help

20200225 update wsjtx links update chirp links

20200210 update chirp links

20200117 update chirp links, add conky video/doc links, add pat menu video link

20191227 update forum link for direwolf in reference section Update JS8Call links to 2.1.1

20191211 update js8call links to v2.0

20191210 add missing command under Winlink test section

20191209 update chirp links

20191123 update chirp links

20191115 Add Part 3 to page

20191112 Add Part 2 to page

20191104 Add video links for Chirp, JS8Call

20191024 Update WJSTX download link

20191022 Add reference section

20191002 Orignal Document Created

This page is written with [SimpleNote](#) in markdown. It has been quite some time since I have written in the markdown language. Writing this document in markdown has refreshed my memory and reminded me of how simple and useful markdown can be.

As an Amazon Associate I earn from qualifying purchases

Published with Simplenote