Documentation for RJOVER

Tyler Kovach

May 2, 2021

Contents

About

This is a short document that should contain all of the necessary information needed to boot and install an OS and prepare a Raspberry Pi 4B for use with the RJOVER project. This document will never be complete as likely there will be bugs along the way and new advancements in the field that render this document completely useless. However, as of the date on the front cover this guide should be up to date. As always, use this document at your own risk.

As the RJOVE project was originally created to be an inclusive learning environment for new students we strive to maintain this ease of use and maximize the learning potential from this new antenna receiver. We the RJOVER team want to strive to continue in the path of predecessors and make this document as inclusive as possible to new users. We will try our best to explain what everything is, but if we miss anything do not be afraid to Google! Sometimes the best way to learn is to do research and try new ideas out. If there are any questions feel free to contact the RJOVE team, or us the RJOVER team.

What is New

As may have come across in the About section, the RJOVER team is not the original creators of this project. This project was originally created by NASA to begin research of

Jupiter atmospheric signals sent here to the Earth. Their original devices were created in a different age, where all radio signals were received and analyzed using physical circuits and sound cards. We now live in a truly digital age where information is more easily accessible and have powerful computers lying around nearly everywhere that we can take advantage of to do data analysis.

What is amazing is that engineers have found ways to fully simulate the processes necessary to filter and process signals without the need for physical circuitry anymore. Our project is thus minimizing the cost for you to help with this research and modernize it with more modern devices. The only 2 devices we recommend you to use with this project are an RTL-SDR and a Raspberry Pi 4B.

The RTL-SDR found here Amazon link to RTL-SDR is what we used in our project, what this guide is made for, and what we recommend you use. SDR stands for a software-defined-radio and its acts just like what its name sounds. This is the device that will be replacing the standard RJOVE receiver. We will eventually get to a point of explaining how it works, but for right now it is just a device.

Secondly, a Raspberry Pi is not what it sounds like. In fact, these are small computers that run on ARM Processors. For our project this will act as our hub computer and control the RTL-SDR as well as act as a storage medium for the data the project collects.

Bill of Materials

This section will describe all of the necessary materials needed to setup and properly use the RJOVE project, as well as contain links to each of the necessary elements.

- 1 RTL-SDR Amazon link to RTL-SDR
- 1 Raspberry Pi 4B with atleast 4Gb of RAM Amazon link to RaspPi 4B
- 1 Micro SD card ?????insert size???? Amazon link to SD card
- 1 power adapter Amazon link to Pi Power Supply
- Atleast 1 Micro-HDMI to HDMI, or some other preferred video connection Amazon link to HDMI
- 1 monitor
- 1 keyboard
- 1 computer you are familiar using
- 1 Micro SD card Writer
- An Internet connection

What is an OS

An OS, also known as an Operating System, is a type of program that facilitates your interaction with a computer. Many new users come across one or two of the major OS types. A brief list of a few popular ones include Windows, MacOS, IOS, and Android. What all of these have in common are that they perform a gateway function to allow users to easily control computers without having to know specific machine code instructions. They also form the foundation on which computer programs run from. The first two are similar since their modern forms are desktop x86 64bit architectures. The second two are made for ARM processors in mobile phones which are more power efficient. Take a look at the Raspberry Pi. Does it look like a modern laptop or PC? No, not really, it is more similar to a modern smartphone than a desktop. More specifically it uses an aarch64 architecture. This means non of the above operating systems will work "out of the box" and we will have to use alternate forms of operating systems on this.

The OS of our choice is a version of Ubuntu Linux called Lubuntu. Linux is an OS not dissimilar to Windows and MacOS. The largest difference is that it is known for being entirely open source and free. Recently it has been gaining in popularity so it has been less hard to get programs to be coded for Linux users who aren't programmers. Typically this is when a lot of Linux jargon is thrown at the reader and they get all confused on what it all means. We will not take this approach since a simple google search of how to install Linux, or what is Linux, will answer those questions beyond unreasonable doubt. Instead we will only introduce material only deemed important and leave the rest as an interesting research project for interested readers.

The biggest difference between Linux and Windows for instance is that Windows is a GUI designed OS whereas Linux is a terminal based OS. Recently there have been more versions of Linux that have GUIs, but most users tend to use the terminal for most of the tasks since it just makes using your computer more efficient. If we wanted to be more formal, these different versions of Linux I have been mentioning are known as distributions. There are many, many different distros of Linux available to download and install. I give you permission to leave this tutorial and go google some Linux distributions. Two of the most common introductory Linux distros are Ubuntu and Mint. Another more complicated distro is known as Arch. For the purpose of this tutorial we will disregard Linux Mint since as of writing this, they do not support the aarch64 architecture for the Raspberry Pi 4B.

Ubuntu is such a large distribution that it has divided up into a couple of flavors that allow for different functionality while utilizing the same core package manager and default applications. These flavors change up the environment that you are computing in. Some flavors act similar to Windows in the form of having LXQt and Cinnamon desktop environments. A great example of this is Lubuntu. Other more notable flavors are Kubuntu and Ubuntu Gnome. For a more inclusive list visit this link. Here we will be using Lubuntu for the actual operation of the Raspberry Pi since it is known for being one of the lightest resource consuming flavor of Ubuntu. This is perfect for the Raspberry Pi since it has limited system resources and has to make the most out of the resources it has. We want to minimize the amount of RAM it takes to run the desktop environment so this RAM can be instead

put to use actually decoding signals.

Along the way to Lubuntu we will spend a couple of minutes in the Ubuntu Gnome basic environment. It may look a bit strange as it is a completely new form of a desktop created only for Linux. This creates both benefits and disadvantages. On a modern computer it is really responsive and a neat environment to work in. However, the cost is this environment will on average consume 4GB of RAM all of the time. With a Raspberry Pi with only 4GB of RAM, this would mean nothing can get accomplished on this computer as long as the desktop is being run. This is why we abandon this environment in favor of Lubuntu in case you were wondering.

People write books on the topic of Linux Distros so I hope this was enough information to atleast describe why we are using the OS we are and leave enough breadcrumbs for interested readers to do some more research and learn more about Linux.

Installation of Boot Media

The first step to begin the install is to navigate your way to this website to begin the install process of the Ubuntu OS onto the SD card.

Plug in the SD card reader and put the SD card in the corresponding slot on the reader. Wait until your computer recognizes that an SD card has been inserted to be used. Typically the computer chimes and a popup appears saying that the SD card is ready to be used. Open up the Raspberry Pi Imager app and click on the "Choose OS" button on the bottom left. A menu should appear asking for which OS you would like to install. We will be installing Ubuntu desktop for this tutorial. In the menu select Other general-purpose OS- Ubuntu- and finally Ubuntu Desktop 64-bit. Right now as of writing this tutorial the desktop we will be using is Ubuntu 20.10 but in the future, this will likely change.

Next, select the SD card from the middle column and select the SD card. The next step will delete all the data on the SD card so realize that this is a not undo-able endeavor. The SD card Will be reformatted to be used as the boot drive for the Pi. Follow the rest of the steps as directed by the program after clicking write in the right column.

This should take some time, 30 minutes on Windows. Afterward, plug it into the Raspberry Pi. The port you are looking for is on the opposite side of the Pi from all of the USB plugs. If you are looking down at the PCB of the Pi, insert the SD card upside down, so you should be looking at the golden contacts pads when you slide it into the SD reader.

Switching to Lubuntu

The next step on our journey will be booting the Pi onto the SD Boot card you made in the previous step. This is important since if you improperly insert commands you may end up breaking the Pi, deleting important data from the SD card, forcing you to reinstall a clean boot image of the OS once more.

Make sure you have a linux compatible keyboard plugged into a USB port and have a monitor connected into one of the micro HDMI ports before you press the power button. On boot Linux will search for these devices and use them to let you into environment. You will likely see a large rainbow square which is the default booting image of a Raspberry Pi. Next you might see some scrolling text. The screen should eventually change and a desktop environment should pop up. This is the OS that you installed onto the micro SD card.

Follow the onscreen instructions to setup the OS. It will likely ask you to select a keyboard layout. English is the traditional QWERTY layout. Next it will likely ask you to insert a timezone so it can figure out what times it should display on the clock. Finally, it will ask you to plug in your name, the computer's name, your username, and a password. Remember this password since it will be necessary to use it to install any applications onto the OS.

Once you have gotten into the desktop you may notice some lag, this is due to almost all of the system resources are being used to run the OS. We will try to exit this as quickly as possible. First we need to setup an internet connection. Hit the Super key (On most English keyboards this is a windows icon on the bottom left side) to pull up the main menu for Ubuntu. Type in "settings" into the search bar at the top of the screen and press enter on the first result. This is the Ubuntu GUI for the settings. I wouldn't go about customizing this OS from here since we will be ditching it very soon. However we need to input an internet connection. If you have Ethernet you should plug in your cord now into the Pi and you should select the Network section in the drop-down menu. In a few seconds you should see an internet connection appear under the wired category to confirm that you indeed connected the Ethernet. If you have WiFi we will navigate to the Wi-Fi section in settings. Note the Raspberry Pi 4B for this tutorial does not support 5GHz Wi-Fi so you will only be able to find the 2.4GHz bands. Select the network and enter the password. If everything entered was correct, in a few moments you should have an internet connection.

Next, press the Super key again and type in "terminal" into the search bar that appears at the top of the screen. When the terminal pops up you will execute your first terminal command.

```
sudo apt install lubuntu-desktop
```

You should need to enter your password and the terminal should take control. A pink menu should appear 5 minutes in asking for a default system setting. Navigate using arrow keys to highlight sddm3 and hit enter. This will make it so the next time we boot into Ubuntu we will be booting into Lubuntu instead.

Now that you will be waiting for a bit we can go over an explanation for what you just did. The first term in the code above is "sudo". This stands for superuser do. Unlike in Windows where you have to open applications with a right click and a selection of "Run as Admin", Linux allows you to do the exact same thing with just one word in the command. This is extremely convenient! A super user in Linux can be thought of as roughly the same as an Admin in Windows. However, when you run sudo you can access and change nearly everything on your computer since all of the system files are restricted to root/super user access. If you do not type "sudo" into the terminal it will say "Access denied" or something

similiar due to you just being a regular user. Perhaps try it out after the terminal finishes running.

The second word typed into the terminal was "apt". This stands for Advanced Packaging Tool. Whenever you use the command apt you are interacting with the package manager for the Ubuntu distro installed onto your system. As you might expect, the wording for this command changes depending on what distro you have installed on your system. For the entirety of this tutorial we will be staying on an Ubuntu distro so this command will not change.

The third word typed into the terminal was "install". This works as you would expect. Putting this all together, this command is changing you into the root superuser which allows you to activate the Advanced Packaging Tool to install "lubuntu-desktop". If that made sense, you're well on your way to understanding how to interface with the Linux desktop. By now the terminal should be finished so we will continue with the tutorial.

Next we will be restarting the Raspberry Pi to boot into Lubuntu. We will use the terminal again.

```
sudo shutdown
```

You will have to wait a few seconds as the Pi closes the few applications and goes to a black screen with a few words left on the screen. From here unplug the USB-C and the screen should blank out. If you plug the USB-C back in, the Pi should restart and you should see the rainbow square again.

Installing Necessary Applications

We are finally on the final and longest step of the install. Here we will install all of the applications needed to make the Pi work with the RJOVE antenna.

When the screen changes you should see an icon with a circle and a bird inside. This is Lubuntu's icon. The program may look very different from the Ubuntu Gnome we were just in, but it uses all the same applications just with different textures and simpler graphics. To log in there should be a box with your username that you should select. You will need to input the password created earlier in the tutorial to log in to Lubuntu.

Lubuntu should carry over the Wi-Fi settings from Ubuntu and/or the Ethernet connection to the Pi. Hit the Super key and the menu for Lubuntu should appear in the bottom left corner. Type terminal and select either of the 2 options. The terminal should pop up on the screen. Next you will need to enter 2 commands into the terminal.

```
sudo apt update
```

This command may take a while to execute as the Pi has a lot of PPA's. PPA stands for Personal Package Archive and acts as a hub for developers to create their own repos to aid in software distribution. As the Pi has a different architecture from regular computers,

it has a lot of special PPA's that it must access. We are refreshing the original list to allow the Pi to get the newest code.

```
sudo apt upgrade
```

This command updates all of the software installed by APT and the PPAs. This will likely take a very long time as the Pi has a lot of updates to go through. I recommend you run this series of two commands every once and a while after this tutorial is over. It will ensure that your Pi has updated code and security for the Linux system. Typically the install time is not this long since you will typically only update 1-5 different applications.

To keep the system lean we will remove the old desktop environment to maximize space on the SD card.

```
sudo apt purge ubuntu-desktop
sudo apt autoremove
```

Next, we will setup GnuRadio.

```
sudo apt install gnuradio gnuradio-dev gnuradio-doc
```

This command will install GnuRadio onto your Pi. After it finishes you should be able to hit the super key, type in GnuRadio and the GnuRadio companion app should become visible to select. This concludes the setup of the software necessary to do the signal analysis on the Pi. Next we will tackle the issue of making the Pi correctly recognize the RTL-SDR and connect to it properly.

Connecting the RTL-SDR

Next, we will make the Pi compatible with the RTL-SDR. There are going to be a lot of steps that need to be followed in the terminal to properly setup the RTL-SDR. Interestingly SDRs were originally TV tuners made for computers, but some engineers thought it would be interesting if theri capabilities were expanded. Unfortunately, by default Linux expects this to still act as a TV tuner. These drivers are what we will be changing. Open up the terminal again.

```
sudo apt install git cmake build-essential libusb-1.0-0-dev
sudo apt install gedit gr-osmosdr
```

This install is collecting the tools needed to retrieve git files, allow us to compile them in the form of cmake, allow us to build the files using build-essential, and finally a library needed for access to USB devices. In case this tutorial is deprecated and newer versions of the build or libusb packages are available I recommend also running the generic update to make sure everything is relevant.

```
sudo apt update
sudo apt upgrade
```

Now we are ready to collect and build the needed RTL2832U Osmocom drivers.

```
git clone git://git.osmocom.org/rtl-sdr.git
cd rtl-sdr/
mkdir build
cd build
cmake ../ -DINSTALL_UDEV_RULES=ON
make
sudo make install
sudo ldconfig
sudo cp ../rtl-sdr.rules /etc/udev/rules.d/
```

Next we have to blacklist the driver that makes Linux think that the RTL-SDR is a TV dongle. Linux will then next default to the drivers we just created for the RTL-SDR.

```
cd /etc/modprobe.d/
sudo gedit blacklist-rtl.conf
```

A file editor should pop up. You will need to enter the following line into the first line for the newly created file.

```
blacklist dvb_usb_rtl28xxu
```

Click the Save button in the top right of the editor and close the editor if it is not closed automatically. Now we will restart the Pi to see if the changed have taken effect.

```
sudo shutdown
```

Unplug the power cord, wait a few seconds, and then plug it back in. Also, plug in the RTL-SDR into one of the blue USB 3.0 ports. We will next test that the SDR is working. Open up the terminal again.

```
rtl_test -s 2400000
```

You should see that an RTL device is found in the terminal and it should find the Rafeal Micro R280T tuner. If you get dropped samples, lower the -s number. Messages like PPL not locked, E4000 tuner not found, it using R820T instead of R820T2 show that the driver and the dongle are working.

This concludes our setup of the RTL-SDR.

Coding in GnuRadio

Hopefully with your collection of materials you also have access to our RJOVER repo on Github here: Link to our Github Repo. There are standard ways to pull the repository for use on your computer. I will not walk through the steps as they are well documented here.

Assuming you have been able to properly pull the repo, the version that is currently being used is TestV5.grc and the subsequent FinalV.py. To get the code runnable on your

computer, open up GnuRadio and select FinalV.grc. It should open a block signal diagram. To compile this and make sure you have the proper Python runnable script, click on the icon that looks like a cube with a downward pointing arrow underneath it. Let it run for about 20 seconds and then close the program. This action has generated a python script for use on the computer.

If you poked around in the specification for this diagram you may have noticed that it writes to an unknown filename in the File Sink block. This is correct since we have written a bash script that will take this generated python file and turn it into a usable loop-able version that appends the date of collection to the beginning of the file.

To modify the FinalV.py file to be usable on your machine we will have to run a series of commands to allow the bash script to be executable and make its changes to the files we are telling it to. First navigate to the directory where the .sh files are held. You can do this by using cd followed by the folder name and repeat until there. Then type in the following commands.

```
sudo chmod 775 binaryjob.sh
sudo chmod 775 editor.sh
```

These two commands will make these scripts executable. To actually use editor.sh we will run the command

```
sudo ./editor.sh
```

This will pop up a GUI asking for the freshly created python script you have just generated from GnuRadio. Select that script. In our case, click on FinalV.py. The script should then modify the code and it should take less than a second.

Next we want to run the code. Depending for how long you want to run the code, there is a value inside binaryjob.sh that describes the run time. It is on line 22. It is currently set to 60s. We recommend keeping it here for initial testing, but for longer runs you can change this at your discretion. Perhaps set it to 12h for 12 hours worth of collection before resetting.

To begin the data collection run the following command

```
sudo ./binaryjob.sh
```

This command should trigger another GUI to pop up. The first prompt wants you to select the modified .py file that just came from the editor.sh script. Next the GUI will ask for you to select a location and filename for the stored data. Do not insert a date beforehand since they will automatically have a data placed before the name unless you deselect that option later. This filename created should end with .dat to endure data compatibility. Next the GUI should pop up a text box. If you enter 0 no dates will be automatically prefixed in front of the names you chose. If you plan to record multiple data sets using the same name, all following data sets will over write the original data set which is not ideal. Enter any other number that is not 0 to append the dates to the beginning of the file. Finally, the GUI will ask how many reps will be required. One rep means after the first collection for the set time

you selected beforehand, the program will automatically shutdown its collection. Finishing this the program should spin into action and you should see a window popup.

Congratulations! You have successfully ran our program and have reached the end of the tutorial. Everything else is left for you to figure out and play with to improve the data collection more than we already have. As always, if there are any issues with this documentation, feel free to reach out to any of us and we will try and help to resolve your issues.

Helpful References

There are a lot of resources on understanding Linux that are available on Google. I will only be listing relevant resources I used to create this document, but feel free to search for others.

RTL-SDR Quick Start Guide

Ubuntu Guide for Install