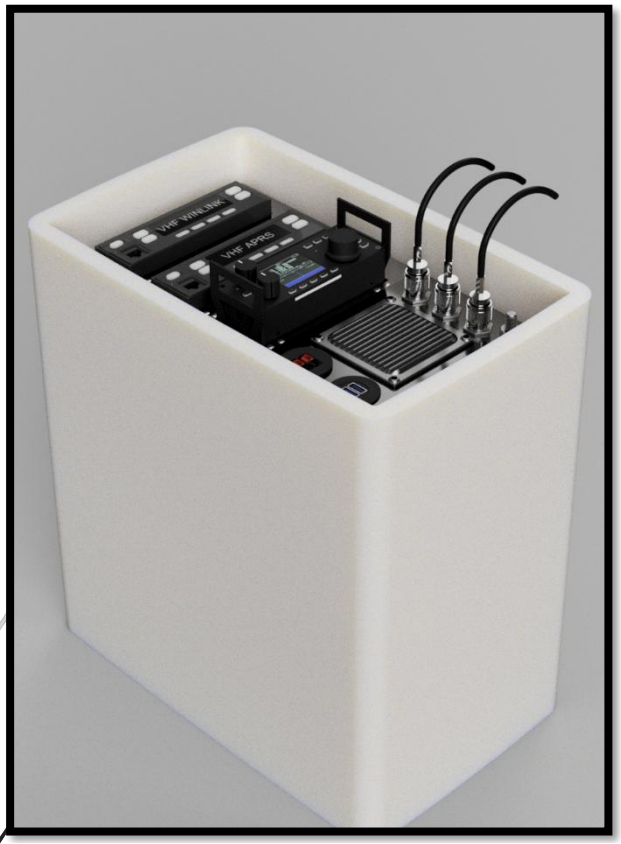
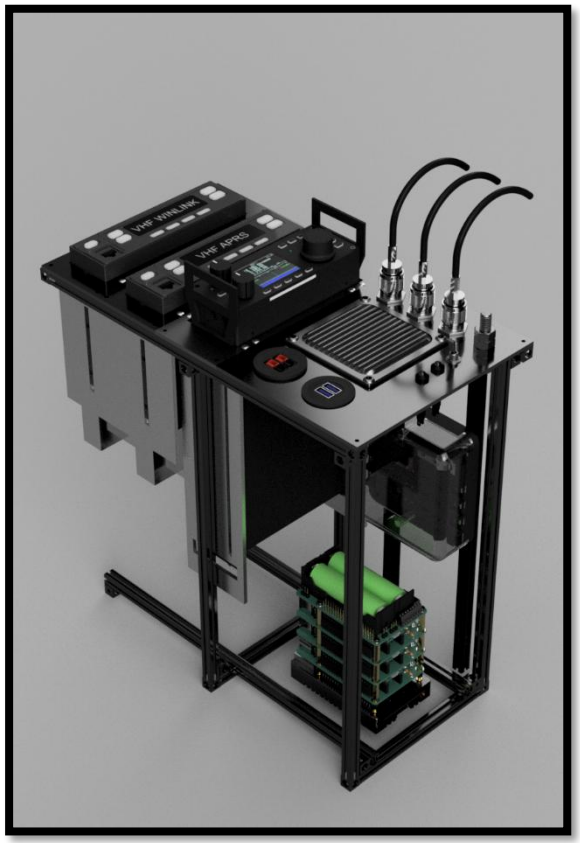
11/18/2023

Portable Modular Packet Go-Box for Amateur Radio

Three radios & A Raspberry Pi





Mitchell J. Skurnik
KN6FOA

Table of Contents

Introduction

Starting in the Fall of 2020, I thought up the idea of a modular system for creating a portable go-box. I wanted something where everything (minus power) was self-contained and except for attaching antennas, nothing needed to be setup. I wanted it to be as simple as opening the box, connecting the antennas, power cable, and then being able to use it immediately.

The following skills were necessary to complete this project:

- Familiarity with CAD
- Access to CAD software
- Familiarity with Linux
- Basic soldering

I do not own a 3D printer, if I had, this project would have been easier to build. I decided to build the entire thing out of aluminum, but you can easily convert all of this to work with a 3D printer instead. The cost may be cheaper to do so. Also, if you have access to a CNC router, you do not need to send this out for laser cutting.

My Requirements

I had simple requirements for the entire project:

1. Total cost of materials should be as inexpensive as possible. Amateur radio is already an expensive hobby and building a box that you may not use that often should as inexpensive as possible.
2. As few specialty tools as possible needed.
3. No exceptional skills required such as welding.
4. Except for the power source and antennas, everything should fit within the enclosure.
5. Enclosure should be waterproof and as light as possible.
6. Controllable via a phone.
7. Three radios including an HF radio station.

Design

Initial Design with Illustrator

My initial design drawn using Adobe Illustrator. I figured I could get away with a single laser cut aluminum part that the radios all attached to. After getting the plate fabricated it was apparent that all the radios would not fit within the ammo box that I sourced from Harbor Freight.

Using Fusion 360

After realizing that illustrator was just not going to cut it for making all the parts, I downloaded a one-year trial of Autodesk's Fusion 360. At this point it had been seventeen years since I last used anything in a 3D program or CAD, but the software was surprisingly intuitive. This enabled me to create everything in 3D and check that everything would fit before making any parts.

Figure 1 - Xiegu G90

Utilizing a nice set of calipers and metric rules, I was able to 3D model all the radios and parts that I needed to fit within the case. Using sites like McMaster-Carr I was able to import existing 3D models of parts I would purchase and for the rest I was able to just create everything by hand.

Figure 2 - Raspberry Pi 4B w/3x TNC hats

Taking a photo of the faces of the radios helped as I was able to import those into Fusion, scale them to the appropriate size, and trace everything while confirming the dimensions with my calipers.

While creating hyper accurate 3D models of your components are not necessary, they are useful and if I may say so, look beautiful. Realistically you can get away with a primitive design that has all the mounting points you need but something about modeling everything is just fun.

There are great tutorials on YouTube to help you learn how to utilize Fusion 360. Using variables and equations help when adjusting things that cannot change and things that can such as material thickness or placement of screws.

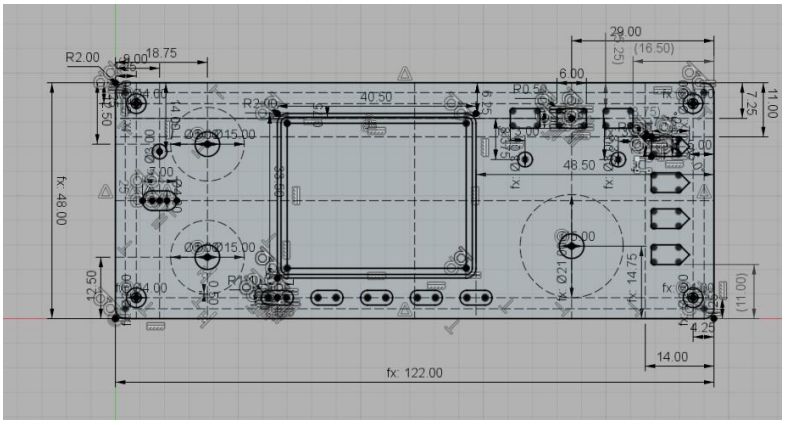


Figure 3 - Engineering drawing of Xiegu G90 front panel.

Ammo box design

I went through dozens of assorted designs. My original plans were to house everything inside of a steel ammo box from Harbor Freight. That did not work out as it did not have enough room for the radios, cables, raspberry pi, and everything else it needed.

After a month of searching, I located this amazing plastic ammo box (figure 4) on Amazon. This thing was beautiful but even after spending months of planning, when I got the parts I came across one critical flaw: the lid took up too much room and there was no room to move the radios down.

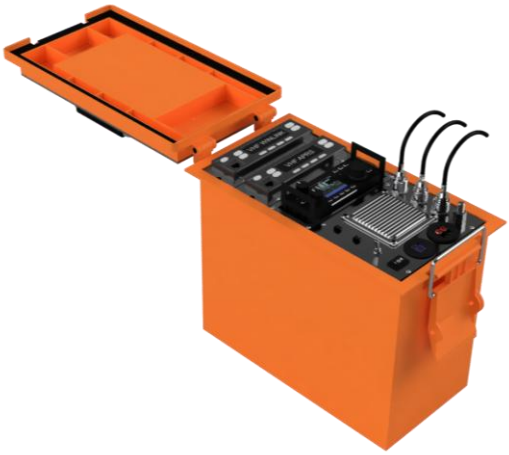


Figure 4 - Orange ammo box design.

I considered fabricating a new lid that would accommodate the extra height I needed but concluded that it would be too difficult with access to a 3D Printer. I will attempt creating a new lid of this box at a point in the future as this form factor is the most compact.

Across this entire project, the one component that has given me the most grief is the Xiegu G90 HF Radio. It has nothing to do with the quality of the radio (it is solid) but due to the length (or height in my case) of the radio in the enclosure. At 250mm end-to end and needing another 50mm or so for power and other cables to attach to the back, the radio is 75mm taller than the VHF/UHF compatriots used in this project.

Figure 5 - Internal view of previous concept for the go box.

Sourcing a Waterproof Container

Unbelievably, the sourcing of an adequate waterproof container for this project assumed the bulk of the project's time. My simple rules for the enclosure are as follows:

1. Cost must not exceed $100 USD (I know it seems silly to limit it on a project costing over $2,000 but you must save $$ wherever you can)
2. Needs to be waterproof or at least splash resistant
3. Requires zero or little modification as I wanted anyone to be able to assemble this with as few specialized tools as needed

Figure 6 - Monoprice Dry box Cooler

I could never seem to find an enclosure that would accommodate everything I needed at a price that was affordable. I would find ammo boxes that were tall enough to house the equipment but not wide enough or I would find ammo boxes that were wide enough but too short.

I tried finding tool chests, ammo boxes, tackle boxes, rack-mount enclosures/road cases, and so forth; they all ended up not working with the dimensions needed to house the equipment, were too expensive, or were too heavy. In the case of road cases, they were too heavy (30 lbs. empty for a wooden enclosure), too expensive (over $100), and too big.

After over a year of searching, I finally discovered dry boxes/coolers. The neat thing about dry boxes is that they have a weatherproof seal, are manufactured from plastic and foam which means they are light, and you can get an appropriately sized one for under $100. The one I purchased from Monoprice was on sale for 25% off and only cost me around $75 before tax. What I liked is that this was about half the weight of a road case and much more compact.

The only downsides to this container were that it was a weird shape and not easy to create a CAD drawing of, and that while the interior dimensions are 13.0" x 7.7" x 14.4" (330 x 449 x 365 mm), they are that at the largest point. The interior tapers down and the bottom is about 20mm smaller in length and width than the top. The top is also not square and has a little bit of an arch to it. This required me to build a frame for each of the module to attach to.

Mockup Using Cardstock

Before sending the drawings out to get laser cut at Send Cut Send, I printed all the drawings out onto card stock at 1:1 scale and cut out the voids. I then used these pieces to make sure that everything fit correctly. I am glad I did this as at least six of the holes for the radio parts were too small and would have required modification of the parts after laser cutting.

I should have followed my own advice, slowed down, and utilized this more as it would have saved me a bunch of time and money.

Failure and Redesign

As of writing this document, I have spent over 18 months working on this project. I had dozens of failed ideas and attempts during my design process. Even after carefully measuring everything, I ended up having parts fabricated that did not fit in their intended place. These were costly mistakes in the tune of hundreds of dollars. But with every failure I learned at least one important lesson. That is part of the learning experience with iterative design. Failure should always accompany a learning experience.

Fabrication

Laser Cut Parts

Utilizing a company called Send Cut Send, I uploaded all my drawings to the manufacturer and had my parts laser cut out of 0.125" 6061 Aluminum. They now also offer CNC milling/routing, tapping, bending services, and other services that

were not available when I started this project. Due to the cost of the bending services, I opted to bend/break the aluminum myself but if you have the money, I suggest having them do it as it will save time and you will not need to deal with annealing your parts before bending them.

The current final iteration of this project arrived at my house after a short lead time. I checked the parts for proper fitment, and I noticed three of the dimensions were not correct. This was a design defect and not a manufacturing defect. I should have used the card stock technique to mockup all the parts prior to getting everything laser cut. Instead of sending revised drawings back out for cutting, I decided to truck along and just modify the parts where I could. The revised CAD drawings included with this document reflect the modifications that I did not get manufactured.

Creating the Extruded Aluminum Frame

Since the project required a super structure to hold everything together, I went with 10mm x 10mm x 300mm extruded aluminum beams I found on Amazon. Again, I measured the dimensions of the enclosure and used Fusion 360 to layout the beams. From this I was able to create a simple cut list for the aluminum parts and figure out the quantity of parts I needed to order.

Material and Tool List

Tools

I used the following tools in making this project:

- Digital set of calipers
- Metric ruler
- Screw drivers
- Allen Keys
- Metal files
- Metal saw capable of cutting through aluminum
- Screw taps for taping threads into the cut aluminum pieces
- Plyers
- Magnetic metal break attachment for bench vise
- Propane/butane torch

Radios

For this project, I sourced three radios. The only constraint I had for picking the radios was that they were inexpensive, at least 20 W, and that they were compact enough to fit in a small enclosure.

Kenwood TM-V71A VHF/UHF radio

At less than $350 used, I purchased this VHF/UHF radio from eBay. Its size, 50W of power output, and built-in data connection on the back of the radio were the main selling points for me

Kenwood TK-762HG VHF radio

You can purchase these radios for $30-60 on eBay. It is a 45W 8-channel mobile radio and the ones you find on eBay have all pulled from fleets of garbage trucks and taxi cabs. They are durable and are reprogrammable with software (KPG-56D) with a cheap FTDI cable to change their default frequency range from 148-178 MHz to a usable range of 136-162MHz that falls well within the 2m amateur radio band.

A UHF version of the radio exists but I do not recommend it as the frequency range is only modifiable from 400-430MHz which does not give enough range within the 70cm band.

Xiegu G90 20W HF Radio

At between $400 and $450 new, this 20W HF SDR radio is an easy decision in terms of use for a portable go-box for packet radio. It has 20W of power capability can expand to 100W with an inexpensive and compact PAX100 RF Power Amplifier if you so desire. There is sufficient room left in the case for this addition, but I have chosen not to at this time. I may add it at a point in the future. Do note that you must disable the auto tuner feature on the G90 when using this amplifier or you will damage it. This means that you will need to use a resonant antenna for the bands you are utilizing.

Electronics

Raspberry Pi 4b 8GB

For this project, I chose the Raspberry Pi 4B as it has more than enough RAM and IO ports. I also used a large CNC milled aluminum heatsink/case to help dissipate heat as the Raspberry Pi 4B can get surprisingly hot under moderate CPU load.

USB GPS antenna

While not strictly necessary for this project, it is nice to be able to send position reports and to view a list of nearby Winlink RMS stations to which you can connect. Additionally, this allows you to automatically update the time.

UPS Battery Backup Hat

Again, I did not need this for this project, but it adds battery backup for uninterrupted continuous use of the Raspberry Pi. This is useful if you need to switch from shore power to battery or solar and do not want to have to safely power down the Raspberry Pi to do so. Additionally, it adds a real-time clock to the Raspberry Pi which means after power cycling the pi, it will retain the date and time and will not need a manual reset.

TNC-Pi9k6 Teensy based modem

I picked the TNC-Pi9k6 as it is a robust TNC that simply plugs into the GPIO headers of the Raspberry Pi. This means you only need to plug in either the serial cable or the mini-din cable coming from the radio into the hat. Fewer cables reduce complexities and saves on space.

Radio Antennas

For the VHF and UHF radios, I sourced a Nagoya UT-72 19-Inch Magnetic Mount Antenna. They are easy to use, and I have been using these on my car and on my apartment balcony for over two years with no problems to report.

For the HF radio, I was able to source a used Super Antenna MP1DXTR80 HF SuperWhip off Amazon for $267 which is around $100 off. I chose this antenna as it is easily field deployable and can be resonant on every amateur band from the 80m band all the way to 220 with great SWR. This is especially important for the Xiegu G90 if you opt to use the 100W power amplifier with it since you cannot use the built-in tuner.

Aluminum

Sourcing small enough extruded aluminum beams was a big challenge as most companies do not carry 10mm x 10mm beams or if they do, it costs too much money. I was finally able to find a company on Amazon called MakerBeam which sells a wide selection of hobby grade parts necessary for this project.

Parts List

The prices below are what I paid for at the time and your price may vary. These prices also reflect buying only the necessary number of parts and does not always include spare parts. I have included links where available.

QTY	Description	Supplier	Price	Sub Total
1	12V to 5V USB Type-C right angle buck converter 15W	Amazon	$ 14	$ 14
1	12-way fuse block	Amazon	$ 15	$ 15
1	80mm Computer Fan Filter Grills	Amazon	$ 13	$ 13
1	FTDI USB Programming Cable for Kenwood TM-V71	Amazon	$ 29	$ 29
1	Geekworm Raspberry Pi 4 Aluminum Case	Amazon	$ 27	$ 27
1	Geekworm Raspberry Pi UPS	Amazon	$ 46	$ 46
1	M2.5 Male Female Hex Brass Spacer Standoff set	Amazon	$ 13	$ 13
1	M3x8mm Machine screws hex socket (50 pcs)	Amazon	$ 8	$ 8
1	M8 50mm Screws	Amazon	$ 11	$ 11
2	MakerBeam 25 Pieces T-Slot Nuts w/Screws	Amazon	$ 18	$ 35
4	MakerBeam 300x10x10mm Beam Black anodized (4 pcs)	Amazon	$ 15	$ 58
2	MakerBeam Corner Cube Black 12 pcs	Amazon	$ 22	$ 44
1	MakerBeam M3 Bolts with Square Head, 6mm (250 pack)	Amazon	$ 16	$ 16
2	MakerBeam OpenBeam Right Angle Bracket (12pcs)	Amazon	$ 8	$ 16
3	Male PL-259 to Male PL-259 12" jumper cable	Amazon	$ 8	$ 24
2	Nagoya UT-72 19-Inch Magnetic Mount Antenna	Amazon	$ 35	$ 70
1	Raspberry Pi 4B 8GB Kit w/microSD Card	Amazon	$ 120	$ 120
1	SO-239 Adapter Bulkhead Panel Mount (4 pcs)	Amazon	$ 13	$ 13
1	Super Antenna MP1DXTR80 HF SuperWhip	Amazon	$ 267	$ 267
1	USB GPS Dongle	Amazon	$ 14	$ 14
1	USB Panel Mount adapter	Amazon	$ 19	$ 19
1	USB to fan PWM Cable	Amazon	$ 8	$ 8
1	FTDI USB Programming cable for TK-762HG	eBay	$ 30	$ 30
1	Kenwood TK-762HG VHF Mobile Radio	eBay	$ 60	$ 60
1	Kenwood TM-V71A VHF/UHF Radio	eBay	$ 330	$ 330
3	TNC-Pi9k6 - Raspberry Pi Hat w/Teensy 3.6	Etsy	$ 82	$ 246
1	TNC interface cable for TK-762HG	Ham Made Parts	$ 35	$ 35
1	TNC interface cable for TM-V71A	Ham Made Parts	$ 30	$ 30
1	TNC interface cable for Xiegu G90	Ham Made Parts	$ 50	$ 50
1	Aluminum Rivets 3/32" For 0.251"-0.375" Thickness	McMaster-Carr	$ 16	$ 16
1	Monoprice Emperor 20 Portable Cooler 5 Gal	Monoprice	$ 75	$ 75
1	Xiegu G90 20W HF Radio	Radioddity	$ 425	$ 425
1	Complete set of laser-cut parts	Send Cut Send	$ 37	$ 37
46			Total:	$ 2,249

Figure 7 - Parts List

Assembly

Extruded Aluminum Frame

The assembly of the extruded aluminum frame is not difficult, but it is a little bit time consuming. Take your time with cutting everything and measure twice before cutting. There are only three cuts needed for the aluminum extrusions and they are all for supporting the bottom of the frame. Use a metal file or sandpaper to deburr the edges.

Assemble all the parts according to the cad drawing with the appropriate corner cubes, t-nuts, and screws. Do not over tighten the screws as the aluminum is soft and can get damaged causing the t-nuts to not slide properly through the channels. Insert 8 t-nuts into each of the top rails to use later for securing the accessory and radio plates.

Next, attach the open beam right-angle corner brackets to add additional rigidity to the frame. Be sure to install the additional t-nuts for the Raspberry Pi assembly for later installation of the Raspberry Pi.

Radio Modules

Each radio module comprises three pieces of laser-cut aluminum and a radio. The two side parts need a 90-degree bend applied to them. Use your digital calipers to measure out the distance between the outer edge and the middle bend line in the engineering drawing. Use the calipers to score the aluminum and then use a sharpie or pen to darken the line. You then need to anneal the aluminum with a propane or butane torch, so the aluminum does not break/fracture/snap when bending. Be sure to do this in a well-ventilated area away from anything that can catch fire and wear appropriate safety gear. A good pair of vice locks with some padding on the ends is useful here to keep grip of the metal without marring the surface or burning yourself. I heated my parts up for about 30 seconds or so on each side and then placed them in my bench vise with the magnetic vise mounted metal break bender attachment jaws inserted in it. It is best to insert the part and bend it in a single motion. Do not attempt to bend the aluminum if it has had too much time to cool. If needed, you can re-heat the part, but I suggest doing it in a single go if possible. Your milage may vary. I easily destroyed over $100 worth of parts by not knowing that.

After completing all the bends, use a rivet gun to rivet the parts together. Be sure to use rivets that are long enough to reach through both aluminum parts such as the ones listed in the Parts section.

Once the three pieces are together, you can then move on to attaching the radio to the aluminum parts. Due note that certain radios require you to pass a cable to the front of the head unit to use the radio in packet mode. Run the cable through the hole on either side of the plate before placing the radio between the two side parts. Secure the radio with the appropriately sized screws and washers. Note that on most radios, the side screw holes offer no protection to the board, and you must not exceed a certain length, or you risk damaging internal components. Read your radio's documentation before screwing everything in and source appropriate length screws before proceeding.

To attach the assembled unit into the frame, place the t-nuts that you inserted onto the top two rails in the Aluminum Frame setup section approximately where the screw holes are in the plate. Secure the module with one of the bolts. If your bolts are too long, use a couple of nuts to make up the difference in the bolt.

Repeat this process for each radio module you wish to add.

Accessory Plate

Parts

The accessory plate comprises of the following parts

- (1) Laser cut aluminum plate
- (3) SO-239 bulkhead connectors
- (1) Grounding screw
- (1) LED power indicating light (optional)
- (1) Power switch (optional)
- (1) Anderson PowerPole bulkhead connector
- (1) 80mm PWM Fan
- (1) 80mm Fan Grille
- (1) USB Front panel connector

Assembly Steps

1. Attach the three antenna connections leaving equal amounts of space on each side of the plate. Use plyers or adjustable wrenches to tighten down the nuts on both sides so when you screw in an antenna on top, it will not loosen the assemble.
2. Attach the grounding screw placing two nuts and a washer on the bottom and two nuts and a washer on top.
3. Attach the fan and the fan grille with the provided screws.
4. Attach the USB front panel connector.
5. Crimp 6-8 inches of appropriate gauged wire to the Anderson PowerPole connector and attach ring terminals to the other end. You will connect this to the fuse block later.
6. Attach the Anderson PowerPole connector to the plate.
7. Optional: Attach the power switch to the UPS switch header.
8. Optional: Attach the LED power status light to the GPIO header.

Electrical Connections

All electrical connections will flow through the fuse distribution block that which is positively fuse.

Cable Assembly

1. For each radio, create a cable assemble with enough slack to reach all the radios and allow for zip tying of the cables to the frame. Also leave enough slack so you can remove the radio without needing to remove the entire frame from the dry box.
2. Crimp ring terminals on one end of the cable
3. Crimp or solder appropriate power connector on the other end for your radios
4. Crimp ring terminals to the USB-C buck converter.
5. Use heat shrink to finish off all connectors.

Block Assembly Steps

We will only be using the left side of the 12-way fuse block so you can use the right side for spare fuses. You may need to check with the user manual of your radios to find this out.

1. Attach the fuse block to the aluminum frame using 4 t-nuts and the appropriate screws.
2. Insert the appropriate rated fuse for the USB-C buck converter into the top left slot.
3. Insert fuses appropriate for your radios.
4. Connect the positive end of each cable assembly to the fused side of the block.
5. Connect the negative end of each cable assemble to the negative side of the block.
6. Connect each power cord to their appropriate radios.

TNC-Pi9k6 Firmware Update

The firmware on the individual TNC-Pi9k6 hats needs to be reconfigured to hard code the I2C port that will communicate with the Raspberry Pi. You cannot do this via the configuration utility. You must compile and upload the code using the Arduino software.

Installing and Configuring Arduino IDE

Follow the TNC-Pi9k6 documentation for setting up your Arduino IDE. Download the firmware source code from my GitHub instead of from where the documentation says to. I have included missing dependencies with the original source code. Feel free to use something like "Beyond Compare" to run a DIFF on the two sets of source code.

Modifying code

The *TeensyConfigPacket.h* file is the only file that we will be modifying.

First, find the line that contains "HOSTPORT Serial1" and comment it out by placing two forward slashes "//" at the start of the line. This will comment it out and disable the use of the data transmitting over the serial port on the GPIO headers. This change will accommodate using multiple TNC-Pi9k6 hats on the same Raspberry Pi as they will all conflict with each other.

Finally, locate the line that contains "I2CSLAVEADDR" and set it to one of the following values:

- 0x1E (This is the default)
- 0x08 (Use for radio 1)
- 0x09 (Use for radio 2)
- 0x12 (Use for radio 3)

This field defines the I2C port that the TNC will communicate with over the GPIO headers. You will need to pick a different port for each of the three TNCs that you are attaching to the Raspberry Pi. After changing the value, save the file, and deploy it to the Teensy board.

Raspberry Pi

The assembly of the Raspberry Pi is straight forward.

1. Place the Raspberry Pi in the aluminum case and following all the instructions that came with the case.
2. Use stand offs from the parts list to attach the three TNC-Pi9K6. Please read the TNC-Pi9k6 documentation and adjust any jumpers appropriate for your radio before installing.
    a. Between each layer of TNC, I used Kapton Tape to insulate the connections as the through hole components come remarkably close to shorting out against other components.
3. Install the UPS hat. Note, charge the 18650 cells before you install them.
4. Secure the Raspberry Pi down to the extruded aluminum frame right angle brackets.
5. Attach the serial or mini-din cables to the TNC-Pi9k6 hats. It does not matter which one you plug them into as we will be defining in software which TNC-Pi9k6 connects to what radio.
6. Optional: Attach an LED to the Accessory Plate as well as a power switch.

Figure 8 - Raspberry Pi w/3 TNC hats

Configuration

Initial Configuration

After installing Raspbian on to your microSD card, insert it into the Raspberry Pi and power it up. You will need a keyboard, mouse, and monitor to complete this section of the guide. It is easier to do this before you attach the Raspberry Pi to the extruded aluminum frame.

Follow all instruction on the Raspberry Pi including setting a strong password. Enable the VLC server and use VNC Viewer on your computer to finish the installation. Once you can connect via VNC, you can install the Pi onto the extruded aluminum frame.

Update your Pi to the latest version of the operating system before starting the next section of this document.

Build-a-Pi

After configuring your Raspberry Pi, we are going to install and use a series of scripts that KM4ACK has created called Build-a-Pi. This installs and configures 95% of all of software that you will need. This process takes about 2 hours to complete so be prepared to wait until it finishes.

If you wish to manually install everything onto your Pi by yourself, may the odds be ever in your favor.

Features to install

- AX25
  - This is a set of protocols designed for amateur radio use. This will connect via a KISS (keep it simple stupid) interface to PAT and other software.
- CHIRP
  - A utility to configure most amateur radios. Great to have but optional.
- CONKY
  - Desktop system monitor configuration customized for amateur radio use.
- Call Sign Lookup
  - An offline database of call signs to look up.
- GPREDICT
  - Realtime satellite tracking and orbit prediction. Useful if you use an Othernet Dreamcatcher or wish to bounce a signal off a satellite.
- GPS
  - Various GPS utilities including the ability to set the system's clock
- Grid Tracker
  - Grid location finder
- Hotspot & Hotspot Tools
  - Creates a hotspot on the Pi so you can connect to it in the field without additional hardware
- PAT
  - Cross-platform Winlink web-based client
- PAT Menu
  - A simple PAT configuration interface that includes the ability to backup configuration files.
- XASTIR
  - A nice APRS client capable of RX/TX of locations, messages, alerts, and weather information.

Customization

We need to assign which hats connect to which radio. Note that you will need to be familiar with Linux for the next few steps.

Configure AX25 Ports

First off, we need to configure the AX25 ports. To do this, we will need to confirm the i2c bus addresses that we configured with the TNC hats that we setup earlier.

Type "i2cdetect 1" in a terminal. If prompted, hit *Y*, and enter to continue.

```
     0  1  2  3  4  5  6  7  8  9  a  b  c  d  e  f
00:                         08 09 -- -- -- -- -- --
10: -- -- 12 -- -- -- -- -- -- -- -- -- -- -- -- --
20: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
30: -- -- -- -- -- -- 36 -- -- -- -- -- -- -- -- --
40: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
50: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
60: -- -- -- -- -- -- -- -- UU -- -- -- -- -- -- --
70: -- -- -- -- -- -- -- --
```

Figure 9 - i2cdetect output

If you followed my guide, we configured the i2c bus addresses to use ports 08, 09, and 12.

In the next step, we are going to launch a text editor with elevated privileges to modify the axports configuration file. If there is anything in there that does not start with a #, put a # at the start of each line. This is a comment and will disable that line.

```
sudo nano /etc/ax25/axports
```

Now that we are in nano editing the ax ports config file, we need to define three new ax25 ports. Use the following to configure ports 08, 09, and 12. Be sure to replace *CALLSIGN* with your callsign

```
i2c08   CALLSIGN-10       1200    128     2       WinLink 1
i2c09   CALLSIGN-11       1200    128     2       WinLink 2
i2c12   CALLSIGN-12       1200    128     2       WinLink 3
```

To save your changes, hit *ctrl + x*, type *Y* to confirm saving, and hit enter to save the file.

Cron Job at start up

Now we need to create a script that runs at startup which will connect the i2c bus ports to the KISS interfaces.

Create a startup script

Start by running the following in the terminal:

```
cd /bin
nano configureax25.sh
```

Next, paste the following into the nano window

```
#!/bin/bash

# Kill existing processes and connections
sudo killall kissattach
sudo killall i2ckiss
sudo ifconfig ax0 down
sudo ifconfig ax1 down
sudo ifconfig ax2 down

# Create I2C KISS Connection
sudo i2ckiss 1 8  i2c08 127.0.0.108 # i2c Port 08 (HEX 08)
sudo i2ckiss 1 9  i2c09 127.0.0.109 # i2c Port 09 (HEX 09)
sudo i2ckiss 1 18 i2c12 127.0.0.112 # i2c Port 12 (HEX 18)
```

Note that on line 13 we are passing a value of *18* instead of *12* as 18 is the Hex of 12. If you are not using the ports I am, be sure to use a decimal to hexadecimal calculator to figure out what value you need to use.

Again, to save your changes, hit ctrl + x, type *y* to confirm saving, and hit enter to save the file.

Configure startup

Now that we have a file named */bin/configureax25.sh* we need to change the permissions of this file to allow for executions. Execute the following to do that:

```
chmod +x configureax25.sh
```

We can evaluate that this is working by typing the following into the terminal:

```
./configureax25.sh
```

The output should look something like what is below. Do not worry if you see the ax# errors

```
ax0: ERROR while getting interface flags: No such device
ax1: ERROR while getting interface flags: No such device
ax2: ERROR while getting interface flags: No such device
i2ckiss 1.0.1.0
slave device is: /dev/pts/1
Resetting TNC...
AX.25 port wl2k bound to device ax0
i2ckiss 1.0.1.0
slave device is: /dev/pts/5
Resetting TNC...
AX.25 port i2c09 bound to device ax1
i2ckiss 1.0.1.0
slave device is: /dev/pts/6
Resetting TNC...
AX.25 port i2c12 bound to device ax2
```

Next, we need to define the boot-time job. To do that we use a utility called *crontab* which defines scheduled tasks

```
crontab -e
```

If prompted, pick option one which is nano, the same editor we have been using in the previous steps.

Use the arrows to go to the second to last line and hit *enter* to create a new line if needed. Add the following to create an entry that runs after Linux has rebooted:

```
@reboot /bin/configureax25.sh
```

We have now configured the script to run at startup.

To use these ax25 ports with PAT Winlink, use the following syntax to connect to a host via one of the radios replacing *i2c08* and *SomeCallSign* with the station you are trying to connect with.

```
ax25://i2c08/SomeCallSign-10
```

Diagnose Problems

To diagnose connection problems, you can execute the following in the terminal to view all ax25 traffic in your Pi

```
sudo axlisten -cartttt
```

Use PAT to connect to a station to view the traffic. To exit this utility, hit *ctrl* + *c* to quit.

You can also use another tab in the terminal to connect to a station and then view the output in the *axlisten* window. This is easier than switching between your browser and PAT.

```
pat connect ax25://i2c08/SomeCallSign-10
```

Using PAT

I will not go over how to use PAT beyond what I have specified in this documentation. I suggest viewing KM4ACK's videos on the subject on his YouTube channel for further information.

Testing

Automated Testing

To facilitate semi-automated testing, I created a bash script that will automatically try to connect to a list of nodes for the specified radios. Personally, I used this to drive around to various places around me, park, and then execute the script. I would then collect the results in an Excel spreadsheet for further analysis.

This simple script can be downloaded from my GitHub repository, and I have provided an example below. What this does is simplifies testing antenna configuration and makes it easier to repeat tedious test. If a GPS dongle is connected, GPS coordinates and altitude can be displayed as well if you set the "gps" value in the script to "1".

After downloading the script, the file's file mode will need to be set to execute. To do this, execute the following in the terminal.

```
1.  chmod +x dualRadioTest.sh
```

Usage

Use radios attached to i2c ports 8 and 12 and try to connect to stations WA1AAA-10 and KA1AAA-10.

```
1.  ./dualRadioTest.sh 8,12 WA1AAA-10,KA1AAA-10
```