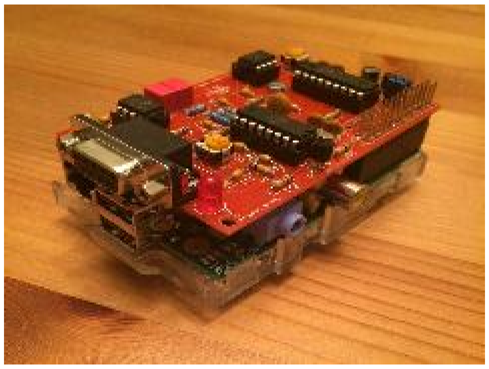
NB7O Raspberry Pi APRS iGate Project



Description

This project is to create an APRS iGate/fill digipeater for the rural area where we live in North East Oregon I am using a 2m FM handheld radio tuned to 144.390 MHz attached to a copper j-pole antenna. Previously I used a rollup j-pole antenna made from 300 Ohm TV twinlead housed in a pvc pipe. Both antennas work very well.

Local APRS users send their position report from their amateur radio station and my radio injects the received data signal into a Terminal Node Controller (TNC) that resides atop the Raspberry Pi. The TNC decodes the audio tones and creates data packets the Raspberry Pi can receive, process and pass along to the APRS Internet Service (APRS-IS) network.

I am documenting this setup and to help others who are new to using Raspberry Pi's in the amateur radio service so that they can easily create APRS iGates.

Major Components

1. **Raspberry Pi 2** or **Pi 3** with:
   a. Power source,
   b. 16Gb or larger microSD card,
   c. Bud Industries "BUD PI SANDWHICH PS-11591" Case
      i. Available from Mouser Electronics as part # 563-PS-11591 ($5.60 ea)
   d. Cat 5/6 network cable with RG-45 connectors or Wifi Adapter (Pi 2 only Pi 3 has wifi built in)

2. **TNC-Pi Ver 2**
   a. Available from Coastal Chipworks - http://tnc-x.com/TNCPi.htm
      i. TNC-Pi 2 (TNC-X for Raspberry Pi B+, Pi 2, and Pi3) Kit ($40)
      ii. TNC-Pi 2 (TNC-X for Raspberry Pi B+, Pi 2, and Pi3) wired and tested ($65)
      iii. Original TNC-Pi (TNC-X for Raspberry Pi) Kit ($40)
      iv. Original TNC-Pi (TNC-X for Raspberry Pi) wired and tested($65)

3. **Internet Access**
    a. I use a Cat 6 cable to an HSMM Mesh node to connect the Raspberry Pi to a mesh network at our residence. The mesh network has a permanent Internet gateway installed and provides access to the internet for sending APRS reports to APRS-IS.
    b. If you have wifi access to the location where the igate will be installed you can use an 802.11b/g/n wireless USB adapter. Or
    c. Connect the Raspberry Pi via a cat 5/6 network cable directly to a network switch or router that has Internet access.

4. A **2m transceiver** capable of operating on 144.390 and capable of connecting a cable to it from the TNC-Pi
    a. I am using an Icom W32a operating at 5 watts.

5. A **data cable** with appropriate connector(s) for the radio end and a db-9 connector for the TNC end.
    a. Cables may be purchased commercially from a variety of sources or homebrewed.

6. **2m vertical antenna**
    a. Preferably mounted at a good elevation to give your station the best opportunity possible to send and receive packet messages to APRS stations

Software

1. **Raspbian OS**
2. **APRX**
    a. Program that takes the serial data coming off the TNC and routes them to APRS-IS

Documentation

1. **APRX Manual**
    a. Available at http://ham.zmailer.org/oh2mqk/aprx/aprx-manual.pdf
2. **TNCPi Manual**
    a. Available at http://tnc-x.com/TNCPi.pdf

Installation, Configuration, and Setup OS on Raspberry PI 2 (PI 3 users skip ahead to similar section for that version)

The first step is to configure the TNC and its communication with the Raspberry Pi. Since you will connect the Raspberry Pi to the TNC via serial port, make sure it is available. (Raspberry Pi is configured by default to use the internal serial port as the console port.)

Many of the files you're editing will need to be edited through `sudo` because you will not have privileges to edit the file as the standard 'PI' user. I prefer to use the file editor `nano` that is included with a standard creation of Raspbian OS for the Pi and my directions will be written using that file editor. If you are familiar and comfortable with a different editor then you may want to use it instead. Editing commands will be presented like this: `sudo nano 'filename'`.

1. Getting started
    a. Download the latest Raspbian OS image and install it onto the microSD chip using Win32DiskManager.
        i. Insert the SD chip into the SD slot for the Pi
        ii. Connect the raspberry pi to your local network via wifi, mesh net, switch, etc
        iii. Apply power and boot the raspberry pi
        iv. Determine the IP address of the raspberry Pi
        v. SSH into the Pi using the IP address from iii on port 22 and putty
            1. User Id: pi
            2. User password: raspberry
    b. Update the operating system
        i. Run the following commands to ensure your OS is up to date (hit enter after each and wait until one runs to completion before starting another):
            1. **sudo apt-get update**
            2. **sudo apt-get upgrade**

2. After creating a new Raspbian OS image some OS configuration changes are required:
    a. At the OS prompt
        i. Type `sudo raspi-config`
            1. This will open the 'Raspberry Pi Software Configuration Tool
        ii. Use Option 1 to Expand Filesystem
            1. This will tell the Pi to use all of the micro SD space as one contiguous filespace/drivespace.
        iii. Use Option 2 to change the Pi password to one of your choosing (preferably something you will remember)
        iv. Use Option 6 (Interfacing Options – you will have to repeatedly select this menu option to continue with steps 1 through 5)
            1. Select 'P4'
                a. Disable SPI
            2. Select 'P5'
                a. Disable I2C
            3. Select 'P6

                a. Disable login over Serial
                b. Then enable the Hardware Serial port
            4. Select ‘P7’
                a. Disable the 1 Wire Interface
            5. Select ‘P8’
                a. Disable the GPIO Remote Access
        v. Exit the Configuration tool and reboot the Pi.
            1. Did you know you can force the Pi to reboot by typing:
                a. `Sudo reboot`
    b. Reconnect to the Pi with putty
            1. Don’t forget to use your new password

3. The next step is to disable the Pi from using the GPIO pins, specifically the serial port, for anything else before we can attach the TNCPi to them. To accomplish this you need to edit 2 files: `/boot/cmdline.txt` and `/etc/inittab`
    a. Type `sudo nano /boot/cmdline.txt`
    b. In `/boot/cmdline.txt`, locate and edit the line reads like this
        i. dwc_otg.lpm_enable=0 console=ttyAMA0,115200 kgdboc=ttyAMA0,115200 console=tty1 root=/dev/mmcblk0p2 rootfstype=ext4 elevator=deadline rootwait
    c. Remove the `console=...` and `kgdboc=...` parts so it looks like this:
        i. dwc_otg.lpm_enable=0 root=/dev/mmcblk0p2 rootfstype=ext4 elevator=deadline fsck.repair=yes rootwait
    d. Save the changes
    e. Type `sudo nano /etc/inittab`.
        i. If present comment out the line:
            1. T0:23:respawn:/sbin/getty .L ttyAMA0 115200 vt100
            2. To comment out the line, simply put a `#` in front of the line.
    f. Save the changes (if any)
    g. Reboot the Raspberry Pi for the changes to take effect.
    h. Reconnect to the Pi with putty

Installation, Configuration, and Setup OS on Raspberry PI 3 (PI 2 users skip ahead to Install APRX section)

The first step is to configure the TNC and its communication with the Raspberry Pi. Since you will connect the Raspberry Pi to the TNC via serial port, make sure it is available. (Raspberry Pi is configured by default to use the internal serial port as the console port.)

Many of the files you're editing will need to be edited through `sudo` because you will not have privileges to edit the file as the standard 'PI' user. I prefer to use the file editor `nano` that is included with a standard creation of Raspbian OS for the Pi and my directions will be written using that file editor. If you are familiar and comfortable with a different editor then you may want to use it instead. Editing commands will be presented like this: `sudo nano '`*`filename`*`'`.

2. The next step is to disable the Pi from using the GPIO pins, specifically the serial port, for anything else before we can attach the TNCPi to them. To accomplish this you need to edit 2 files: `/boot/cmdline.txt` and `/etc/inittab`
    a. Type `sudo nano /boot/cmdline.txt`
        i. Remove the following line in `/boot/cmdline.txt` if it exists:
            1. `console=ttyAMA0,115200 kgdboc=ttyAMA0, 115200`
        ii. Add the following lines to `/boot/config.txt`:
            1. `enable_uart=1`
            2. `dtoverlay=pi3-miniuart-bt`
            3. `core_freq=250`

3. The following lines need to be added to `/lib/systemd/system/hciattach.service` to configure the universal asynchronous receiver-transmitter (UART) to communicate the with TNC Pi. (You may have to create the file in case it doesn't exist in the latest version of the distribution. If the file does not exist it will be created when you execute the sudo nano command). Make certain that the line is typed exactly as printed below; remember Linus is case sensitive. Also make certain that the line is typed continuously on one line even though the line may appear to have a <enter/new line> in the text below due to the word wrap of the word processor.

4. Type `sudo nano /lib/systemd/system/hciattach.service`
    a. Add the following lines (exactly as written)
        i. `[Unit]`
        ii. `ConditionPathIsDirectory=/proc/device-tree/soc/gpio@7e200000/bt_pins`
        iii. `Before=bluetooth.service`
        iv. `After=dev-ttyS0.device`
        v. `[Service]`
        vi. `Type=forking`
        vii. `ExecStart=/usr/bin/hciattach /dev/ttyS0 bcm43xx 921600 noflow –`
        viii. `[Install]`
        ix. `WantedBy=multi-user.target`
    b. Save the changes to this file

5. Once you've made those changes, reboot the Raspberry Pi to ensure it will automatically pick them up during a restart

Install APRX

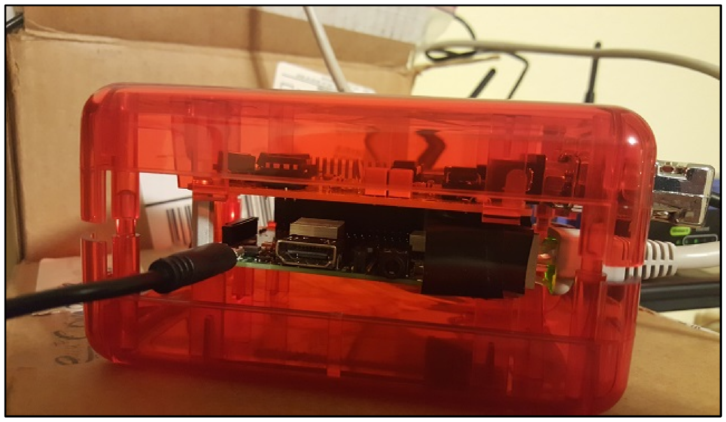
1. First need to download the software from the project page.
    a. Using an internet browser log onto http://ham.zmailer.org/oh2mqk/aprx/
        i. Look at the file listings for a file similar to aprx_2.08.593-1_armhf.deb
            1. This was the current APRX version at the time of this paper being prepared.
                a. (Be sure to download the most recent .deb version for the armhf architecture.)
    b. Easiest way to get it on the pi is to use `wget`.
        i. On the Pi type

```
wget http://ham.zmailer.org/oh2mqk/aprx/aprx_2.08.593-1_armhf.deb
```

        ii. The file will download into the directory that you are currently logged into.
    c. Once the file is downloaded, you need to use dpkg to install the .deb file.
        i. To install using the debian package manager, run the following command
            1. `sudo dpkg -i aprx_2.08.593-1_armhf.deb`
                a. (...or whatever your filename is.)
    d. During the debian package install you may notice a few uninstalled required packages. To download and install those missing files run the following:
        i. `sudo apt-get install -f`
2. To post APRS data that your station receives on the APRS-IS server you need to generate a passcode.
    a. Using an Internet Browser log onto http://n5dux.com/ham/aprs-passcode or any of the many APRS-IS password creation sites.
    b. Write down the passcode the webpage gives you. You will use the password in the next step.
        i. Note – all SSID's for your root callsign uses the same APRS-IS password.

3. Now with the APRX software installed and your APRS-IS key created you need to edit the `aprx.conf` file in `/etc.` directory.
    a. In the APRX manual the sections of the .conf file is documented.
    b. It may be helpful to review a "simplied" version without as much visual clutter.
    c. The APRX manual also has a few examples as well.
    d. Below is a copy of my `aprx.conf` file.
        i. In the sample below text in red should be kept exactly as it is. Text in blue should be edited to meet local information; e.g. callsign, lat/lon of the station, beacon text to be sent to nearby stations, etc.
        ii. The serial device is ttyAMA0(zero not Oh)

Rasp PI with TNCPI in Bud PS-11591 case



NB7O Rasp PI APRS station using APRX.

Sorry for the mess but this was the best place to set the components where I could get them in the picture without having to unplug.

aprx.conf for NB7O

```
mycall NB7O-1

myloc lat 4550.14N lon 11920.23W

<aprsis>
  login     NB7O-1     # login defaults to $mycall
  passcode XXXX
  # server    rotate.aprs2.net #switching server to experiment
  server rotate.aprs.net 14580
  heartbeat-timeout   0    # Disabler of heartbeat timeout
</aprsis>

<logging>
  pidfile /var/run/aprx.pid
  rflog /var/log/aprx/aprx-rf.log
  aprxlog /var/log/aprx/aprx.log
</logging>

<interface>
   serial-device /dev/ttyAMA0  19200 8n1    KISS
   callsign     $mycall  #callsign defaults to $mycall
   alias        RELAY,WIDE,TRACE
   tx-ok        true
   telem-to-is  false
</interface>

<beacon>
   beacon for NB7O-1 raw "!4550.14N/11920.23W`APRX-HSMM-Mesh - iGate on an
HSMM-Mesh node—APRS over 2.4Ghz!"
</beacon>

<digipeater>
   transmitter $mycall
   <source>
      source $mycall
     relay-type     digipeat

   </source>
</digipeater>
```

4. Once the `aprx.conf` file is configured you will need to restart the aprx service to apply your changes. Restart aprx through this command:
5. `sudo service aprx restart`

6. You should now be able to see your iGate showing up online at sites like http://aprs.fi/ and you can also see any traffic received by your igate by viewing the file `aprx-rf.log`.
    a. To view the data traffic and stations that your igate has received
        i. At the Pi OS prompt type `cat /var/log/aprx/aprx-rf.log`

Sample of NB7O aprx-rf.log

```
2016-11-06 18:26:06.972 NB7O-1    T NB7O-1>APRX28,:!4550.14N/11920.23W`APRX-HSMM-Mesh - Raspberry Pi iGate on an HSMM-Mesh node—
APRS over 2.4Ghz!
2016-11-06 18:34:12.986 NB7O-1    R N7NOA-9>TVPUQR,WIDE1-1,WIDE2-1:`/)Dqqdk/`"9:}HAPPY HOLIDAYS_%
2016-11-06 18:34:12.987 NB7O-1    T N7NOA-9>TVPUQR,NB7O-1*,WIDE1*,WIDE2-1:`/)Dqqdk/`"9:}HAPPY HOLIDAYS_%
2016-11-06 18:36:17.986 NB7O-1    R N7NOA-9>TVPSPY,WIDE1-1,WIDE2-1:`/)Jqqlk/`"8,}HAPPY HOLIDAYS_%
2016-11-06 18:36:17.987 NB7O-1    T N7NOA-9>TVPSPY,NB7O-1*,WIDE1*,WIDE2-1:`/)Jqqlk/`"8,}HAPPY HOLIDAYS_%
2016-11-06 18:41:43.064 NB7O-1    R KF7WQ-9>TUUPPT,WIDE1-1,WIDE2-1:`/._l^`u\`"5K}147.020MHz T103 +060 Semper Fi_"
2016-11-06 18:41:43.064 NB7O-1    T KF7WQ-9>TUUPPT,NB7O-1*,WIDE1*,WIDE2-1:`/._l^`u\`"5K}147.020MHz T103 +060 Semper Fi_"
2016-11-06 18:42:33.548 NB7O-1    T NB7O-1>APRX28,:!4550.14N/11920.23W`APRX-HSMM-Mesh - Raspberry Pi iGate on an HSMM-Mesh node—
APRS over 2.4Ghz!
2016-11-06 18:44:25.984 NB7O-1    R N7NOA-9>TUUWXQ,WIDE1-1,WIDE2-1:`/0Tr!nk/`"5f}HAPPY HOLIDAYS_%
2016-11-06 18:44:25.985 NB7O-1    T N7NOA-9>TUUWXQ,NB7O-1*,WIDE1*,WIDE2-1:`/0Tr!nk/`"5f}HAPPY HOLIDAYS_%
2016-11-06 18:50:26.987 NB7O-1    R N7NOA-9>TUURQU,WIDE1-1,WIDE2-1:`/0Er"0k/`"5a}HAPPY HOLIDAYS_%
2016-11-06 18:50:26.987 NB7O-1    T N7NOA-9>TUURQU,NB7O-1*,WIDE1*,WIDE2-1:`/0Er"0k/`"5a}HAPPY HOLIDAYS_%
2016-11-06 18:52:28.988 NB7O-1    R N7NOA-9>TUUPVT,WIDE1-1,WIDE2-1:`/2Ar"2k/`"5_}HAPPY HOLIDAYS_%
2016-11-06 18:52:28.988 NB7O-1    T N7NOA-9>TUUPVT,NB7O-1*,WIDE1*,WIDE2-1:`/2Ar"2k/`"5_}HAPPY HOLIDAYS_%
2016-11-06 18:53:18.987 NB7O-1    R N7NOA-9>TUTYYT,WIDE1-1,WIDE2-1:`/3r!vk/`"5a}HAPPY HOLIDAYS_%
2016-11-06 18:53:18.988 NB7O-1    T N7NOA-9>TUTYYT,NB7O-1*,WIDE1*,WIDE2-1:`/3r!vk/`"5a}HAPPY HOLIDAYS_%
2016-11-06 18:56:13.937 NB7O-1    R N7NOA-9>TUTWUY,WIDE1-1,WIDE2-1:`/2~q?k/`"5k}HAPPY HOLIDAYS_%
2016-11-06 18:56:13.938 NB7O-1    T N7NOA-9>TUTWUY,NB7O-1*,WIDE1*,WIDE2-1:`/2~q?k/`"5k}HAPPY HOLIDAYS_%
2016-11-06 18:59:58.548 NB7O-1    T NB7O-1>APRX28,:!4550.14N/11920.23W`APRX-HSMM-Mesh - Raspberry Pi iGate on an HSMM-Mesh node—
APRS over 2.4Ghz!
2016-11-06 19:04:34.939 NB7O-1    R N7NOA-9>TUTUUV,WIDE1-1,WIDE2-1:`/'"r5k/`"6J}HAPPY HOLIDAYS_%
2016-11-06 19:04:34.940 NB7O-1    T N7NOA-9>TUTUUV,NB7O-1*,WIDE1*,WIDE2-1:`/'"r5k/`"6J}HAPPY HOLIDAYS_%
2016-11-06 19:18:40.881 NB7O-1    T NB7O-1>APRX28,:!4550.14N/11920.23W`APRX-HSMM-Mesh - Raspberry Pi iGate on an HSMM-Mesh node—
APRS over 2.4Ghz!
2016-11-06 19:35:13.548 NB7O-1    T NB7O-1>APRX28,:!4550.14N/11920.23W`APRX-HSMM-Mesh - Raspberry Pi iGate on an HSMM-Mesh node—
APRS over 2.4Ghz!
2016-11-06 19:54:01.156 NB7O-1    T NB7O-1>APRX28,:!4550.14N/11920.23W`APRX-HSMM-Mesh - Raspberry Pi iGate on an HSMM-Mesh node—
APRS over 2.4Ghz!
2016-11-06 20:13:48.973 NB7O-1    T NB7O-1>APRX28,:!4550.14N/11920.23W`APRX-HSMM-Mesh - Raspberry Pi iGate on an HSMM-Mesh node—
APRS over 2.4Ghz!
2016-11-06 20:33:38.548 NB7O-1    T NB7O-1>APRX28,:!4550.14N/11920.23W`APRX-HSMM-Mesh - Raspberry Pi iGate on an HSMM-Mesh node—
APRS over 2.4Ghz!
2016-11-06 20:53:18.548 NB7O-1    T NB7O-1>APRX28,:!4550.14N/11920.23W`APRX-HSMM-Mesh - Raspberry Pi iGate on an HSMM-Mesh node—
APRS over 2.4Ghz!
2016-11-06 21:10:23.546 NB7O-1    T NB7O-1>APRX28,:!4550.14N/11920.23W`APRX-HSMM-Mesh - Raspberry Pi iGate on an HSMM-Mesh node—
APRS over 2.4Ghz!
2016-11-06 21:25:47.001 NB7O-1    R KF7WQ-9>TUUPPT,WIDE1-1,WIDE2-1:`/.dl+*u\`"5[}147.020MHz T103 +060 Semper Fi_"
2016-11-06 21:25:47.001 NB7O-1    T KF7WQ-9>TUUPPT,NB7O-1*,WIDE1*,WIDE2-1:`/.dl+*u\`"5[}147.020MHz T103 +060 Semper Fi_"
2016-11-06 21:26:48.925 NB7O-1    R KF7WQ-9>TUUPQQ,WIDE1-1,WIDE2-1:`/.^m4Nu\`"5I}147.020MHz T103 +060_"
2016-11-06 21:26:48.926 NB7O-1    T KF7WQ-9>TUUPQQ,NB7O-1*,WIDE1*,WIDE2-1:`/.^m4Nu\`"5I}147.020MHz T103 +060_"
2016-11-06 21:27:03.546 NB7O-1    T NB7O-1>APRX28,:!4550.14N/11920.23W`APRX-HSMM-Mesh - Raspberry Pi iGate on an HSMM-Mesh node—
APRS over 2.4Ghz!
2016-11-06 21:27:20.925 NB7O-1    R KF7WQ-9>TUUPSR,WIDE1-1,WIDE2-1:`/.]n\u\`"5A}147.020MHz T103 +060_"
2016-11-06 21:27:20.925 NB7O-1    T KF7WQ-9>TUUPSR,NB7O-1*,WIDE1*,WIDE2-1:`/.]n\u\`"5A}147.020MHz T103 +060_"
2016-11-06 21:28:53.935 NB7O-1    R KF7WQ-9>TUUPYW,WIDE1-1,WIDE2-1:`/.\m>Ru\`"5E}147.020MHz T103 +060_"
2016-11-06 21:28:53.935 NB7O-1    T KF7WQ-9>TUUPYW,NB7O-1*,WIDE1*,WIDE2-1:`/.\m>Ru\`"5E}147.020MHz T103 +060_"
2016-11-06 21:45:23.546 NB7O-1    T NB7O-1>APRX28,:!4550.14N/11920.23W`APRX-HSMM-Mesh - Raspberry Pi iGate on an HSMM-Mesh node—
APRS over 2.4Ghz!
```