RASPBERRY PI BPQ NODE & DIGIPEATER

Author:	Gordon L. Gibby MD KX4Z
Date:	September 2016
Version	1.1 October 6 2016 -- added reboot cron task
	1.2 October 12 2016 -- added information on setting volume levels
	1.4 October 24 2016 -- added updates for things not working well under most recent distributions.

Purpose

These instructions will help you create a raspberry-pi based linux BPQ (linbpq) "node".   This node can function similarly to a "KA node" and automaticaly as a very capable digipeater. A digipeater is a store-and-forward AX.25 packet repeater that observes received packets, and when it find ones asking for digipeating by any of its callsigns/names, it repeats the packet "as is" except for adding an (*) asterisk after its own callsign in the header, to demonstrate that the repeated packet comes not from the originating station, but from this digipeater.

A digipeater or node constructed using these instructions can be extremely inexpensive.   It can utilize a $25 Baofeng UV5R transceiver, or a higher powered transceiver.

Digipeaters have advantages and disadvantages.   They are very simple! However, because they do not check packets, detect missing, corrupted or damaged packets and immedially get a resend, they can forward packets that are out of sequence, in a train that has a missing packet, corrupted or otherwise damaged.   That means the final target station has to do the job of rejecting, and asking for a resend....and that message too...has to go back down through the digipeater chain!   So they aren't the best for throughput on a lossy pathway.   However, their ease of use makes them ideal candidates for people who are just learning the ins and out of packet radio (like me).

This system has far greater possibilities, which aren't fully discussed in this paper, including:

- acting as a "node" with greater powers than a digipeater
- emulating an RMS_PACKET server
- automatically connecting to different radios to establish a robust packet chain of transmission
- accepting TELNET connections

If we all were familiar with all these systems, installing this entire system and getting it to work would take less than an hour.   However, just like me, you are probably LEARNING about everything as you go along, so don't get discouraged!!   This may take a bit to complete.   Work at it in sections.   If you have a TNC-X or a Signalink that you can borrow, that will be an easy first starting point before building the $10 TNC if that is your final goal!

Purchase List

Raspberry Pi Computer & Related

Suggested examples of purchaseable items are just suggestions. _

Raspberry PI Version 3 (which means it is a "B") $36
https://www.amazon.com/Raspberry-Pi-RASP-PI-3-Model-Motherboard/dp/B01CD5VC92/ref=sr_1_3?s=pc&ie=UTF8&qid=1473881712&sr=1-3&keywords=Raspberry+Pi+3

Clear protective plastic case with heatsinks: $6
https://www.amazon.com/Raspberry-Model-Protective-Heatsinks-Clear/dp/B01CDVSBPO/ref=sr_1_1?ie=UTF8&qid=1473881862&sr=8-1&keywords=raspberry+pi+3+cases

micro SDHC camera memory (and some Windows-based computer will allow you to do the initial formatting and loading of that card)
example: $9 https://www.walmart.com/ip/Kingston-Flash-memory-card-microSDHC-to-SD-adapter-included-16-GB-UHS-Class-1-Class10-microSDHC-UHS-I/47394503 Class 10 means fastest.

Large cellphone "backup" battery that advertizes that it can be charged and used at the same time, and has the micro-usb connector to provide power to the Raspberry PI. This allows you to create the effect of a UPS-- uninterruptible power supply--for your Raspberry PI. (Some cheaper models cannot be both charged and used at the same time) You choose the size, the suggested one is powerful; I've much significantly smaller ones than this, but remember it has to be able to charge at same time as used:
$20
https://www.amazon.com/gp/product/B00FBD3MVA/ref=oh_aui_detailpage_o00_s00?ie=UTF8&psc=1

High capacity USB cell phone charger with the micro-B connector used for Android phones. Recommend >= 2 Amps example: $10
https://www.amazon.com/Intocircuit%C2%AE-Universal-Including-Intocircuit-Smartphone/dp/B00LAR3BLW/ref=sr_1_1?ie=UTF8&qid=1473882091&sr=8-1&keywords=cell+phone+ac+charger+2A

USB mouse (for setup, not required for usage) You can use any available one, or pick one up from Walmart

USB keyboard (for setup,not required for usage)

If you don't have an HTML monitor (many TV's will work) I have had good success with a HTML--> VGA converter I bought for $25 from Walmart.

Terminal Node Controller (TNC)
You can use many different TNCs. For all choices, you will need an appropriate cable to connect to your ham radio, and back to the TNC of your choice.

Suggested TNC's: (From MOST expensive to cheapest)

- **TNC-X** -- connects easily through the USB port. Configuration information will be provided below. **(Kit $88, built $123)** http://www.tnc-x.com/ This option does not need the software DIREWOLF -- after getting your Raspberry Pi going, you can skip all the DIREWOLF stuff and go way down to the section on installing LINBPQ.
- **Signalink sound-card.** Approximately $100. Just like the homebrew soundcard TNC below, this option uses the DireWolf software to make the Signalink into a TNC. I suggest looking at www.cheaperham.com for the lowest prices on Signalinks.
- **TNC-PI 2** -- not discussed in this paper, but it should work. $65 http://www.tnc-x.com/TNCPi.htm if you can get the configuration right.
- **Homebrew "$10 sound-card based TNC"** -- by far the cheapest, but you have to solder it together from a commercial sound dongle. Like the Signalink, this option uses the software TNC-emulator DireWolf.. All the instructions to build your TNC are found here: http://www.qsl.net/kx4z/InexpensiveTNC.pdf

Continuing on, to build your Raspberry Node that will work with the TNC you chose:

Format your SDHC card, using a Windows based computer.

Note: In all the following, when you need to type something into a terminal window from your Raspberry Pi, I use `Courier TypeFont` **for exactly what you need to type.**

1. Purchase an SDHC card that is between 8 and 32 Gigabytes (8 is plenty).
2. Using an appropriate adapter if necessary, install the new micro SDHC card into your computer and CAREFULLY NOTE WHAT DRIVE IT BECOMES. (You don't want to accidentally erase your C: drive on your computer, right?) Use the SD Association's Formatting tool. Download from sdcard.org. Set "FORMAT SIZE ADJUSTMENT" option

ON in “options” menu to make sure the entire card id formatted. I have used this on Windows computers. Windows Link: https://www.sdcard.org/downloads/formatter_4/eula_windows/index.html

3. There is a separate link for apple computers on the page.

PreLoad your SDHC card with NOOBS

1. Stands for “New Out Of the Box Software.” Allows you to get your raspberrry operating system of choice.
2. On your windows computer with the SDHC card still installed, navigate to: www.raspberrypi.org, click on DOWNLOADS button. Download NOOBS; I suggest using “offline and network install” as a ZIP file. Version as of 2016-05-27 is 1.9.2, 1.02 gigabytes. *That might take a while, depending on your network connection speed.* I don't operate an Apple computer but it probably works similarly for this and the next step.
3. Unpack the ZIP file onto the SD card that you're going to use. (On the computer I use, when you click the file from within the Windows Explorer it offers you the option to Extract All and to specify where you want it to go.) Once again, be CAREFUL exactly where you extract this to -- the extraction is large, 91 files comprising about a gigabyte.

Bring Up Your Rapberry PI Operating System

1. Plug the SDHC card into your Raspberry. Connect up a mouse and keyboard to the Raspberry, using the USB slots farthest from the Ethernet connector (nearest the edge of the card) and an HDMI monitor. Get ready, but don't yet connect up the micro-usb power to the card. Follow the directions on this video to get your Raspberry started. https://www.youtube.com/watch?v=y4GOG4P-4tY Although you may be offered a chance to connect to your WIFI system, it isn't necessary because you downloaded the NOOBS version that included “offline” installation of RASPBIAN.

2. When offered, click the box by the RASPBIAN operating system and then click INSTALL. This is going to take a while, as it is going to write 3209 Mbytes worth....on my system at 5 Mb/second, so about 10 minutes worth.

   Optional: Additional video with more ideas for you: https://www.youtube.com/watch?v=-6OGuhLtKbU

3. *Note: On my system, installation automaticaly set “boot directly to desktop” and “desktop log in as user “pi”, so that the system would nicely turn itself on. If you are offered options related to this, be sure to set them accordingly.*

4. Once Raspbian comes up, you can go into the
   MENU | Preferences | Raspberry Pi Configuration
   and set the nation, timezone, and country for WIFI -- after which my wifi began to work

properly. You'll probably need to put in your wifi system's password etc. You need to be on the Internet for the next portion.

5. Under the tab "Interfaces" I enabled the **I2C** and **Remote GPIO**

Update and Add Sound Library to Your Raspberry Pi

Bring up a terminal window (the icon on the menu bar that looks like a black chalkboard). Your prompt starts out with a $ because you aren't superuser. If you type the command "`whoami`" it will tell you that you are user "pi"

Typing the command "`date`" will give you the date and time. "`info date`" will give you the syntax if you need to fix anything. I fixed the time with "`date hhmm`" After that my updates went better.

UNIX HINT: to kill any terminal screen (or the thing runningin it) type CTRL-C

Updating your onboard software to the latest fixes:

```
sudo apt-get update   (I got some errors on this one)
sudo apt-get dist-upgrade
sudo rpi-update
```

Install libasound2-dev package

```
sudo apt-get install libasound2-dev
```

Install a graphical user interface program scheduler

```
sudo apt-get install gnome-schedule
```

Download DireWolf C-Source Code and Make Your Own Executable

Using the built-in browser on your raspberry (looks like a globe) download the latest unix DireWolf sourcecode. This can be done from

Page: https://github.com/wb2osz/direwolf/releases

Click on and download **direwolf-1.3.tar.gz** (or whatever the latest appears to be and use that new name in the all the following if it has been updated beyond version 1.3)

My downloads showed up in the directory

```
/home/pi/Downloads
```

I then made a directory for them
```
mkdir  /home/pi/direwolf
```
Copy what you downloadedto that directory:
```
cp /home/pi/Downloads/* /home/pi/direwolf
```
Move yourself to that directory
```
cd /home/pi/direwolf
```
And extract all the files from the “tarball”
```
tar -zxvf direwolf-1.3.tar.gz
```

Note for your education, those options include:

-z work on gzip compression automatically
-x extract archives
-v verbose output so you can tell what is going on
-f read the following file as input

The extraction processwill cause a new subdiretory **direwolf-1.3 to be** created and populated with all the files. Move there
```
cd /home/pi/direwolf-1.3
```
.

Compile the entire source code and make an executable on your own raspberry pi (this is amazing): [replace the version with whatever version you obtain]

`make –f Makefile.linux`   *This will take a while and compile a BUNCH of c code.*

`sudo make install`   (is what the program recommend and what I did; as root)

`make install-conf`   Gives you an initial configuration file

`make install-rpi`   Gives you a desktop icon

```
cd /home/pi
```
and you can find the conf file there
```
ls
```

Dire Wolf is now installed and you should see an icon on your desktop for it.

Configure Your New DireWolf Sofware TNC

If you haven't done so already plug in your USB-based TNC (which might be an altered sound dongle) into the slot that is high and nearest to the Ethernet 10BaseT socket.

(for this to work right, you have to do a reboot of your raspberry pi)

Next find your sound card's device address with this command:
```
aplay -l
```

If everything is working you'll see the default (raspberry hardware) ALSA audio device at card 0 and your USB TNC (whichever one you chose) at card 1: (amongst the gibberish)

card 0: ALSA..........device 0
card 0: ALSA...,,,,,,,device 0
card 1: [something related to USB or CODEC] device 0: [something related to usb]

Remember that you're on card 1.

Now hunt for your mic input on your TNC

```
arecord -l
```

The Raspberry doesn't come with an input so you should see only

card 1:   [something related to USB or CODEC for a Signalink]: device 0 {more metion of USB]

Armed with this information we can now use any Text Editor to edit the **direwolf.conf** configuration file.   A Text Editor doesn't put in extraneous formatting stuff the way a wordprocessor does....   There are several possibilities on your Raspberry Pi:

Menu | Accessories | Text Editor	PREFERRED OPTION
nano	A small text editor
vi	A really complicated text editor

Unless you are a Unix Guru I suggest you use the Text Editor because it looks like something you're likely familiar with and it works.   Just for reference sake, I include the following brief information on the ubiquitous vi editor (don't use this unless you're forced to):

Brief intro to vi commands

vi filename   opens filename for text editing

vi has two modes, command and text.

[ESC]   puts you back into command mode -- do this whenever you are confused.   arrow keys work to move you around in command mode.   Dont ever go into text mode until you are sure you are exactly where you want to be

i   puts you into text mode and will begin inserting IMMEDIATELY

a   puts you into text mode will begin appending IMMEDIATELLY

DO NOT USE THE ARROW KEYS in text mode.

As soon as you are finished editing, hit [ESC] and get back into commmand mode, otherwise you risk adding the darndest text in the wierdest locations you ever imagined....which can have disastrous results.

:q   quit

:w	write out to the filename
:q!	quit without writing
/text	find text in the file

Are you sufficiently convinced this is not for beginners?

To be the safest, ***use the built-in text editor*** -- go to **Menu | Accessories | Text Editor** and use it to find your direwolf.conf file and edit as follows, then SAVE when you're done.

Open **/home/pi/direwolf.conf. In the Text Editor you'll have to click on a few directory-structures, but with a bit of trying, you'll find it.**

1. In the FIRST AUDIO DEVICE PROPERTIES you need to have an ADEVICE statement that fits with whatever soundcard system you're using. There are several options, pick and try until you get one that works with your setup.

`ADEVICE plughw: 1, 0` Used to work with both Signalink and USB soundcard dongle, but seems to fail now, possibly due to updates to firmware in my Raspberry
`ADEVICE plughw: CARD-Set,Dev=0` Works with USB Adafruit audio adapter
`ADEVICE plughw:CARD=CODED,DEV=0` Works with Signalink

2. In the section CHANNEL 0 PROPERTIES edit the MYCALL to replace NOCALL with your callsign -- and probably with a -SSID, usually 7, such as

```
MYCALL  K1ABC-7
```

3. Leave the MODEM 1200 line uncommented.

4. Go down and verify that you have
```
AGWPORT 8000
KISSPORT 8001
```

5. You can create a beacon to allow you to test that this is working and direwolf can transmit:

```
PBEACON delay=1 every=2 overlay=S symbol=”digi”
lat=00^00.00N long=000^00.00W   power=5 height=5 gain=0
comment=”[your call] Test Direwolf”
```

Don't forget to turn this off later on after you have it all working

6. Now save the file with the original file name (direwolf.conf)

Testing your homebrew or sound-card based TNC

Plug in your radio to your TNC and your TNC into the USB slot. Tune to some frequency where you can receive packet signals, click on Direwolf icon to get it going and see if it can hear packet signals. Adjust the volume so that it works, and so that the volume number is around 50.

Adjust the transmitter potentiometer on your TNC so that your signal is just below maximum amplitude (an attempt to set the deviation within proper limits)

Once you have this working, you probably want to go back into the direwolf.conf and comment out (#) the PBEACON line because you're going to have your NODE do the beacons, not your interface software from now on.

Setting up the LinBPQ Node

Instructions direct from the author of the BPQ software:

https://dl.dropboxusercontent.com/u/31910649/LinBPQ_RMSGateway.html

Other instruction sources that were very helpful to me:

http://www.wcares.org/special-interests-3/aprs/aprs-raspberry-pi-virtual-tnc/
http://www.wcares.org/special-interests-3/aprs/aprs-raspberry-pi-virtual-tnc/
https://philcrump.co.uk/Raspberry_Pi_APRS_Digipeater

http://www.seapac.org/documents/pdf/2015-n7qnm-Implememting%20an%20APRS%20Digipeater%20on%20a%20Raspberry%20Pi.pdf

An already-built pi version of LinBPQ (bpq software for linux) is available at:
https://dl.dropbox.com/u/31910649/pilinbpq

Create a subdirectory to hold the software and configuration files:

```
mkdir /home/pi/linbpq
```

```
    cd /home/pi/linbpq
    wget https://dl.dropbox.com/u/31910649/pilinbpq
```

Rename the program to linbpq, then make it executable

```
        mv pilinbpq linbpq
        chmod +x linbpq
```

You also need some web pages for the management interface. Create directory HTML (capitals) under your linbpq directory, and download and unzip https://dl.dropbox.com/u/31910649/HTMLPages.zip into it.

```
        mkdir /home/pi/linbpq/HTML
        cd /home/pi/linbpq/HTML
        wget
        https://dl.dropbox.com/u/31910649/HTMLPages.zip
        unzip HTMLPages.zip
        cd ..
```

Configuration of your linBPQ Node

The executable does not come with the required bpq32.cfg configuration file. This file is very important! The best way I can provide it to you is to have a sample file on this website, or email to you.

Link to sample file:
as simple html text: http://qsl.net/kx4z/bpq32.html
as a PDF: http://qsl.net/kx4z/bpq32.pdf

Using your browser on your Raspberry Pi, load the sample file into a window. The select the entire text and paste it into the Text Editor. Then save it in the same directory as your linbpq (/home/pi/linbpq) Edit the radio port lines as appropriate depending on whether you are using a $10 TNC, a Signalink, or a TNC-X (the latter of which is the only onethat doesn't need DIREwOLF, and would connect to USB0)

NO MATTER HOW YOU RETRIEVE THE TEXT OF THE FILE, WHENYOU SAVE IT IN YOUR /home/pi/linbpq DIRECTORY t he file **has** to be named exactly **bpq32.cfg** The case of the letters is important!

RADIO FREQUENCY INTERFERENCE

It is crucial that you deal with radio frequency interference to avoid possible damage to your setup and the need to start over with your flash drive.... There is so much RFI near even a 5 watt walkie talkie that I have seen analog voltmerters "peg" on the 10 volt DC scale connected to NOTHING. In some ways, a handitalkie is the WORST offender for RFI. Why? Where is the bottom half of that rubber-duckie dipole? It is the radio itself -- including any wire connected to it.....which would include the audio wires going to your TNC or soundcard!!! Since you aren't HOLDING the walkie talkie when it is used on a raspberry pi, the full RF current of the "other half" of the dipole can head towards your precious electronics!

Here's how to protect your systems:

> Put ferrite clamp-on beads on the cables from the radio to the sound card, and if 1possible, between the sound card and the Raspberry (that lead is short).
> Install two or three 2" loops in the cable from the radio. Use electrical tape or a zip tie to hold them in place.
> Notice that the TNC-X is in a shielded box. So is the signalink. Your raspberry is NOT, and if you used the $10 TNC, it isn't either! Let's fix that. Put insulating baggies or other material (such as an envelope) around the $10 TNC and put the Raspberry PI in a plastic case (if you haven't already). Now put the entire setup inside aluminum foil (you can even just wap the system or put some on the outside of a gallon baggie that makes it easy to move them in and out. Use some aluminum foil to make a "shield braid" for the cable heading to the radio, and connect it to the aluminum that shielding the Raspberry and the $10 TNC.
> Get everything running before you turn on the radio. Then remove the USB cables for the MOUSE and KEYBOARD (they aren't needed once the application is started), get everything shielded, and THEN turn on the radio.
> If you have to back out of the software or fix something, turn the radio OFF first, and then plug the mouse and keyboard back in and go about your repair work..

Running your Node

Finally! We get to run the thing. If you are using a sound-card type TNC, first double click on the DireWolf icon and it should start and look for a connection.

Then startup a terminal window and get into your linbpq directory:

```
cd /home/pi/linbpq
```

Start the program

```
./linbpq/linbpq
```

(note the period in front)

In a later version of this document I'll include information how to get the file to automatically start -- and stay running!

Install TELNET application on your Raspberry

Bring up a terminal window.

```
sudo apt-get install telnet
```

Now once your linbpq is running, you can connect to it using telnet, in addition to a packet over-the-air connection.   (If you know your IP number [`ifconfig` will tell you that] , you can even connect from TELNET on another computer.   Telnet no longer automatially comes on Windows machines, but you can re-enable it.)

On your own raspberry Pi, it is easy:

```
telnet localhost 8010
```

(The 3rd port in the supplied config file accepts telnet connections.)

In order for this to work, you will need to have included a user name and password for yourself in the TELNET port section of bpq32.cfg

Monitor your system with a Web Browser

Using your raspberry's browser, navigate to http://localhost:8080, or using another computer's browser to http://[your ip number]:8080 and guess what, you have web access to control your Node software!

Making your software start up automatically and re-start if it crashes

We take advantage of the linux **cron** daemon to get this to happen for us.

First we position two *scripts* within /home/pi.   One script checks on direwolf, and if not running, starts it.   The other does the same for linbpq.   You can capture the verbiage for these files on the web, paste into the Text Editor and save them, or you can type them in yourself:

filename: /home/pi/direwolfscript   Web: http://qsl.net/kx4z/direwolfscript.html

```
# find direwolf process identifier
```

```
/usr/bin/pgrep direwolf
# if not running...
if [ $? -ne 0 ];
then /home/pi/direwolf-1.3/direwolf
fi
```

filename: /home/pi/linbpqscript Web: http://qsl.net/kx4z/linbpqscript

```
# find linbpq process identifier
/usr/bin/pgrep linbpq
# if not running...
if [ $? -ne 0 ];
then /home/pi/linbpq/linbpq
fi
```

Next we have to get these scripts called every 2 minutes from cron. This can be done by manually filling out the crontab file by issuing the command

```
crontab -e
```

which will first ask you for your choice of editor (select nano) and then allow you to edit the file, write it out, and close.

Or, since you installed the graphical Task Scheduler, you can go to the Menu

Menu | System Tools | Scheduled Tasks

and enter in two new tasks:

== ======= FIRST TASK ==============

Command	**`/bin/bash /home/pi/direwolfscript >/dev/null`**
Default behaviour	
Advanced:	(click)
Minute:	*/2
Hour	*
Day	*
Month	*
Weekday	*

(Apply)
========== = ====================

== ======= SECOND TASK =============

Command	**`/bin/bash  /home/pi/linbpqscript >/dev/null`**
Default behaviour	
Advanced:	(click)
Minute:	*/2
Hour	*
Day	*
Month	*
Weekday	*

(Apply)

========== = ===================

<u>Making your Raspberry not be permanently crippled by RFI</u>

There is the possibility that you may have some RFI to your USB-port connection to a Signalink, or other interface TNC. In my experience, this will shut down the port, and Direwolf will be unable to function, hence your system quits performing. (**`plughw`** fails) In my limited experience, one system in a low-RF environment appears to have zero problem with this, while another, in a higher-RF environment has locked up the USB port occasionally.

The problem is that unless you provide for this, it *remains* down.

There may be another solution, but rebooting the raspberry pi does work. Hence you may wish to add a cron job to reboot your raspberry pi at intervals. Rebooting is very quick, and the unit takes only a minute to do so. In my case, I selected every six hours. Unfortunately, user “pi” can't do this -- the cron job will fail because the operating system will ask for a password authentication. The way around this is to have root set up a cron job to reboot.

I don't know exactly how to do this in the graphical system, but it is reasonably easy to do in a terminal window. So start one up, then

```
    sudo su
```

to grab root authority, then

```
    crontab -e
```

to open an editor to edit the crontab file for root (which is different from the one you effectively created above for user pi, to start Direwolf and linbpq.

Then use editor nano (or vim if you prefer) to put in the following line:

```
    0 */6 * * * /sbin/shutdown -r now >/dev/null
```

the second entry is */6 which means midnight, 0600, 1200, 1800 (hours divisble by 6); there are five total entries (minute, hour, day, month, day-of-week) and spaces between them.

Another housekeeping item you might want to take care of within your root crontab table is to delete various `chatconfig.conf.bakX` and other .bak files that accumulate.... you can create a script that looks something like this:

```
rm /home/pi/chatconfig.conf.bak*  >/dev/null
sleep 4
```

and then call that script just before the shutdown/reboot command, with another line in the root crontab table (see below)   The purpose of the sleep commands is to let the system finish writing out the changes to non-volatile memory

```
0 */6 * * * /bin/bash /home/pi/cleanupscript >/dev/null
2 */6 * * * /sbin/shutdown -r now >/dev/null
```

Setting the Audio Gain (Volume) Levels

You'll need to set the mic gain and audio output levels if you're using a soundcard-based technology. If the audio output isn't at 100%, Signalink and similar devices (such as the $10 TNC) may not key the transmitter properly.

```
alsamixer
```

will bring up a “somewhat graphical” volume control. Use Left/Right cursor keys to reach your sound-card device. Use the up/down arrow keys to adjust volume. You probably want the audio output at maximum and the mic gain at 50%.

NOTE: for some reason, my most recent distributions of Raspian updates end up with `alsamixer` not providing mic / playback volume controls for USB-soundcard based devices.... Suggest you set whatever volume is provided to the max to be sure that PTT circuitry works well. I will be investigating fixes to this issue. Other possiblities are direct adjustment of receive and tx volumes with `aplay` and `arecord` commands....

You're Finished!

That's It.   Your system should now work.   As you read and learn more, you can edit your LINBPQ node's configuration file (keep a backup!) and add more features and learn how to get even more out of it.   It has amazing powers to connect digital streams from one radio to another, to WINLINK, from one frequency to another, out many different protocols, including PACTOR, WINMOR, PACKET, etc.   We owe a lot of thanks to John Wiseman G8BPQ for writing this for us.