

To build a frequency agile remote base, there are not a lot of VHF/UHF radios that support CIV/CAT control. However the Kenwood TM-V71A has this. And there is hamlib support.

Hamlib (Ham Radio Control Library) is a software development project that provides a standardized Application Programming Interface (API) to control amateur radio equipment, such as transceivers.

You'll want to use that radio in a single band mode (band A only) else channel control is convoluted. Don't worry you can mix VHF and UHF channels within that single band.

Set the Packet radio data band to A, (menu item 517), and the data speed to 1200 (menu item 518), and the PC port speed to 38400 (menu item 519), and lastly set the Squelch output setting to SQL so the exported signaling follows the tone decoder. (menu item 520)

I used the allstarlink version 3 appliance installed on a Raspberry Pi 4 (4 GB), but it should work with less resources (down to a Pi3 with 1 GB ram).

<https://allstarlink.github.io/install/pi-appliance/pi-detailed/>

Do not install hamlib support from the repository with apt-get. Download and install from the source code. The repo version didn't work for me and was hard to properly uninstall. I used Hamlib 4.5.5.

<https://sourceforge.net/projects/hamlib/files/hamlib/4.5.5/>

```
./configure  
make  
make install
```

The supported radio list is here, and it also gives you the model ID so that you can address it. For the KenwoodTM-V71A that is 2035

<https://github.com/Hamlib/Hamlib/wiki/Supported-Radios>

The sound interface I use is a DMK UR1x, but there are other options from repeater builder as well as other sources and you also have the ability to modify a USM sound dongle.

<http://www.dmkeng.com/images/UR1x%20Data%20Sheet.pdf>

This is the audio interface between the computer and radio that also handles the PTT and COR support.

Follow the steps for a standard allstarlink install and setup. From the simpleusb-tune-menu, you'll set your audio levels. If you don't have a deviation meter or other high end test equipment compare the audio levels from your handheld when sending a DTMF digit. First set the transmit audio from the node. Monitor this with an oscilloscope or volt meter and match it to what you measure with your handheld from a test receiver. Use the parrot mode to match the receiver input level to the node. There are other good tutorials to set your levels.

Some relevant other settings since we are deal with flat audio and external tone decoding:

```
carrierfrom=usb  
ctcssfrom=no  
plfilter = no
```

Channel control is done in rtp.conf,

9701= cmd,/usr/local/bin/rigctl -m 2035 -r /dev/ttyUSB0 -s 38400 E 1 ; this tunes to channel 1

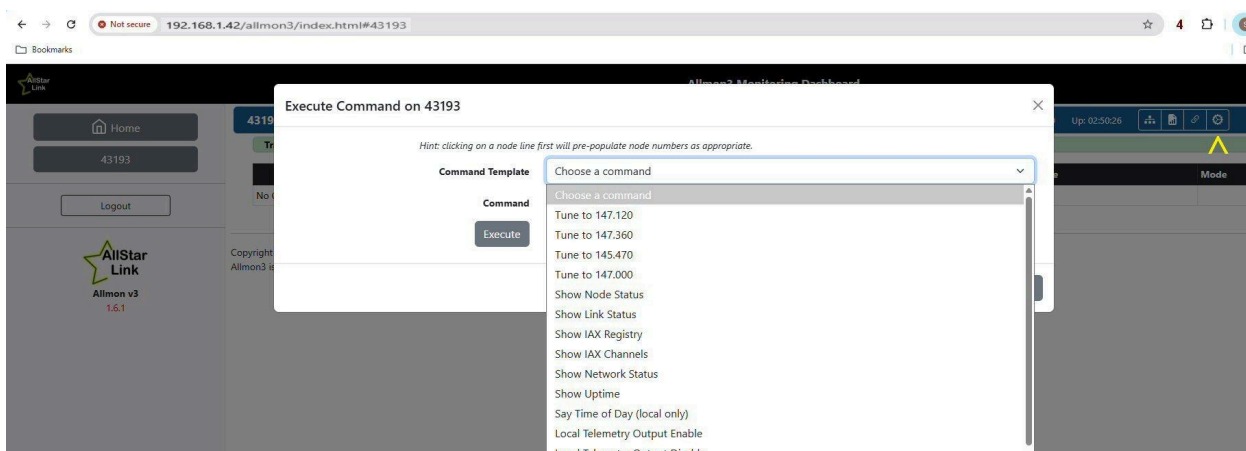
9702= cmd,/usr/local/bin/rigctl -m 2035 -r /dev/ttyUSB0 -s 38400 E 2; this tunes to channel 2

Define and add a line like this for each programmed channel in the radio. (These are the DTMF user macros *9701, etc)

You can also use a similar syntax as shown above to automatically channel steer the radio based on time of day for nets or other periods of radio traffic. Set that in the standard linux crontab, using the rigctl system calls.

Allstar3 ships with allmon3. This is a modern, web-based monitoring and management dashboard designed for AllStarLink. This can be opened to remote use via port forwarding rules on your network. This will allow remote app/phone users to be able to control the channel the link radio is on. I use the DVSwitch App (on Android) to handle the audio streaming to talk in and out over the link radio. Many people with IOS use the RepeaterPhone app.

From there we can edit the allmon3 ini file so that there are web buttons for tuning the radio to a specific channel.



Those changes will be made in /etc/allmon3/web.ini

Under the [syscmds] section we will use the rpt fun command to mimic an over-the-air DTMF keypress sequence which we defined earlier in rtp.conf

rpt fun 43193 *9701 = Tune to 147.120

rpt fun 43193 *9702 = Tune to 147.360

It's also possible to add echolink support to an Allstar node, but you'll be restricted to the crappy GSM audio that echolink uses. Don't be a loser, set up an allstar account even if you don't deploy a node like this. Then you can talk in with the G.711 codec.

If you start down this path to build a frequency agile node and find other things I should have noted in this document, please let me know.

Steve, KB9MWR