KA9Q-radio

Created: Tuesday, April 18, 2023, 21:21
Last Modified: Sunday, May 28, 2023, 19:08

KA9Q-radio

---

preparing the software

- git clone
- create and activate a conda environment for this (e.g. ka9q)
- follow instructions in INSTALL.txt -- install libraries, select Makefile, make and make install
- follow instructions in docs/FFT3W.md

Signal Path

- **frontend** --> multicasts IF signals from SDR hardware as RTP/UDP stream
- **radiod** --> (takes IF stream from a frontend, demodulates one or more channels in the IF stream, and outputs PCM as multicast RTP/UDP streams)
- **data module** (*monitor* listens to PCM output, *opusd* transcodes PCM, *pcmrecord* records stream to disk, *packetd* demodulates, *wspr-decode* decodes WSPR signals, *aprsd* processes APRS packets, aprsfeed) -->
- **control** (manage demodulator instances -- mainly HF because typically define channelized U/VHF in config files) -->
- **monitor** (listen to multicast audio output)

For example: funcubed (funcubed.conf)--> radiod (radiod@FCD) --> opusd --> monitor

Funcube Dongle Pro Plus example

- ka9q-radio source includes a udev rule for fcdpro+ which creates a symlink called FCDPP in /dev. Move this to /etc/udev/rules.
- edit /etc/radio/funcubed.conf to run **funcubed** frontend:

```
sudo
```

- setup frontend module as a service:

  ```
  sudo systemctl enable funcubed@2m
  sudo systemctl start funcubed@2m
  sudo systemctl stop funcubed@2m
  sudo systemctl restart funcubed@2m
  sudo systemctl status funcubed@2m
  ```

- config an instance of radiod.
  Its name, e.g., /etc/radio/radiod@FCD.conf, includes the name "FCD" of an instance you defined in the radio frontend config.

AirspyHF+ example

RX888 Mk2 example:

OS needs to recognize USB device.

frontend --> multicasts IF signals from SDR hardware as RTP/UDP stream

edit rx888d.conf:

- **[Label]** -- used to enable and start the device service, e.g., sudo systemctl enable rx888d@Label. Phil most commonly fills this with the type of antenna connected to the radio, like [G5RV] or [2m]
- **iface** -- must correspond to correct ethernet device, e.g., iface = eno1
- **status** -- must correspond to what you define as **input** in radiod@hf.conf, e.g., status = rx888-status.local

You then start a service for any [Label] in rx888d.conf. For instance, for [g5rv]:

```
sudo systemctl enable rx888d@g5rv
sudo systemctl start rx888d@g5rv
sudo systemctl stop rx888d@g5rv
sudo systemctl restart rx888d@g5rv
sudo systemctl status rx888d@g5rv
```

radiod --> (takes IF stream from a frontend, demodulates one or more channels in the IF stream, and outputs PCM as multicast RTP/UDP streams)

edit radiod@hf.conf:

- **input** -- corresponds to **status** in rx888d.conf, e.g., input = rx888-status.local
- **[Label]** -- corresponds to the streams of interest one defines, e.g., [WSPR], [WWV], [W1AW-cw], etc.
- **data** -- in each [Label] defines a pcm stream for a downstream service, like monitor, to ask for and manage, e.g., data = wwv-pcm.local or data = wspr-pcm.local
- **mode** -- of demodulation, e.g. mode = am, mode = lsb, mode = iq, mode = pm (phase-modulated), etc.
- **freq** -- list of space-delimited frequencies.

data module (*monitor* listens to PCM output, *opusd* transcodes PCM, *pcmrecord* records a pcm stream to

disk, *packetd* demodulates, *wspr-decode* decodes WSPR signals, *aprsd* processes APRS packets, aprsfeed) -->

still gotta figure out other demodulation/decoding

control (manage demodulator instances -- mainly HF because typically define channelized U/VHF in config files) -->

still gotta figure out control

monitor (listen to multicast audio output)

invoke with one of the data streams defined in radiod@hf.conf, e.g., monitor wwv-pcm.local, monitor wspr-pcm.local, etc.

antenna
airspyhf+ discovery
usb
ubuntu 22.04