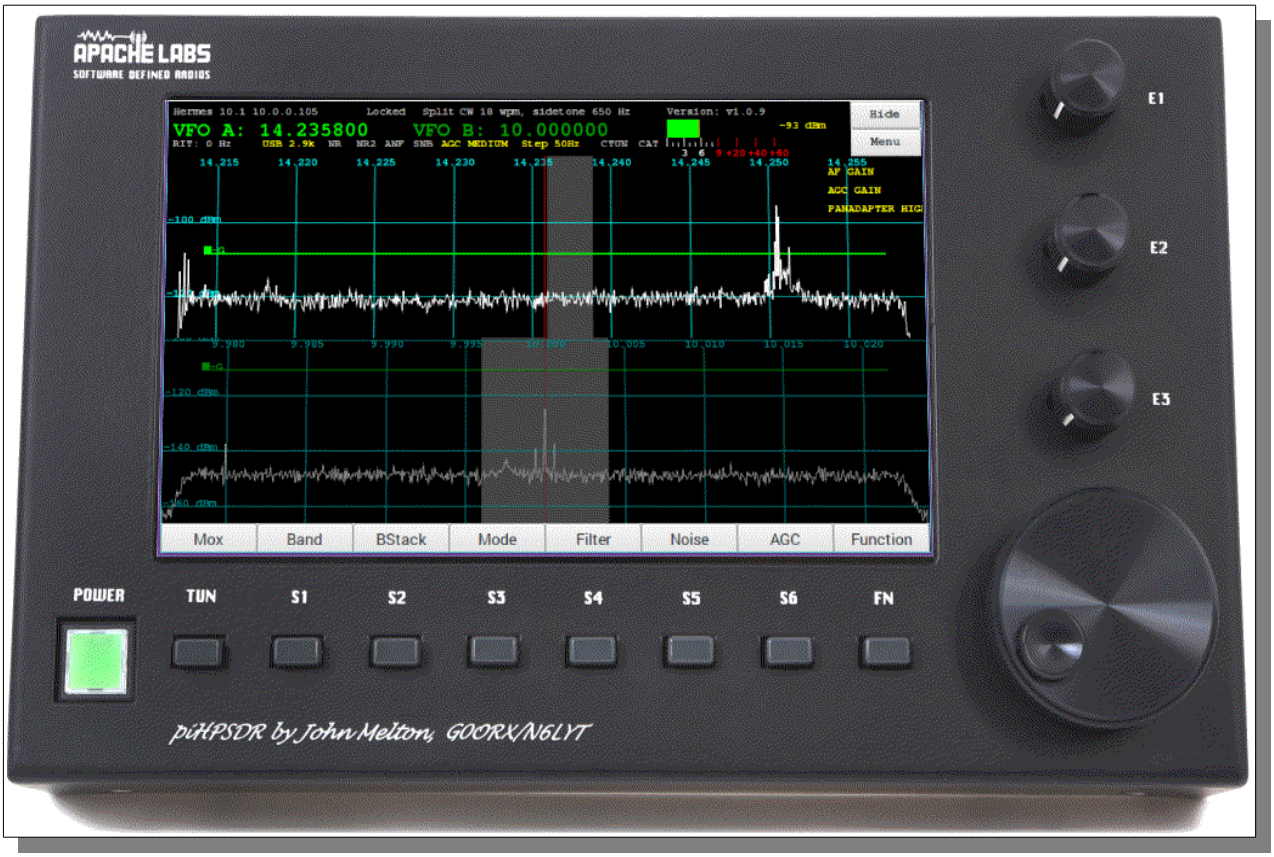
Apache Labs LLC



pihpsdr showing two receivers (20M USB + WWV) + waterfalls + on screen sliders + menu toolbar

TWO RECEIVER Version

piHPSDR Controller

(v1.1.0-beta)

Users Guide

by

John Melton GØORX/N6LYT

This document contains references to the Apache Labs Transceiver products
http://www.apache-labs.com

In cooperation with VK6PH, NRØV, W5WC, K5SO, KA6S
and the OpenHPSDR Hardware and Software Projects
http://openhpsdr.org

Contents - piHPSDR Controller Users Guide

1. Introduction

Introduction by John Melton GØORX/N6LYT

Thu Aug 5, 2016 12:52 am (PDT) .

Posted by: "John Melton" g0orx

I think it is time to answer some of the questions about the controller.

Some background ...

I presented a paper at DCC last year on how to build a CAT controller using an Arduino and some buttons and encoders, even one with an LCD screen showing the mode, frequency and an S meter. The idea of this was to try to get people building them.

When I returned I started looking at what a Raspberry Pi (model 2 at the time) could do as a standalone controller. I had already ported WDSP to Linux so had a good starting point. Very soon after I received
an email from LA2NI, Kjell Karlsen asking if my Android code would run on the Raspberry Pi as he wanted to try to build a standalone system with a touch screen. I told him about my early development work on the Raspberry Pi and that started a project that is still on going today.

The idea was not to build an all singing/dancing copy of PowerSDR but to build something that could be used with a small portable system. Kjell has since gone on to build a very nice system using a Hermes and Apollo all built into a small case with the Raspberry Pi (now a model 3).

Abhi (Apache Labs) got interested in the project and wanted to build a controller initially without the built in Hermes/Apollo. Hopefully we will see a complete portable system sometime from Apache Labs.

So, that is what we have today. The software is still in development and new features are being added as well as bugs being fixed. It is designed to use with the radio, not as a remote console. The RPi does not have any audio input but does have audio output. It is currently possible to output the audio locally on the RPi but to connect a microphone you will need to plug a USB audio card into the device. I will be adding support for connecting a microphone to the this as well.

The RPi will work over WiFi using the built in controller. The signal quality will determine what sample rate it will run at. The RPi does not have GigE so cannot connect directly to a radio running the new Ethernet protocol, but will work through a switch or over WiFi. However, it has been tested using a USB to GigE dongle and that does work. Just remember this will not run at the higher sample rates.

I have looked at several other more powerful single board computers, but always I have run into problems with their implementation of the GPIO, which is used to interface the buttons and encoders.

We are using the Raspberry Pi official 7 inch touch screen. This has a resolution of 800x480 so we are struggling with screen real estate.

There is work in progress to also use the controller remotely across the Internet. This will require a server running at the radio that is connected to by a thin client running on the controller. This will stream the audio and spectrum information with all the DSP work being done on the server. It is not based on my ghpsdr3 project but of course that was a good starting point for the design.

Finally, all the software is open source. It will run on a Raspberry Pi without the buttons/encoders. It will also run on other Linux systems and I have even built it and had it running on Mac OS X. I have not
tried to build it on Windows as I do not have a Windows system.

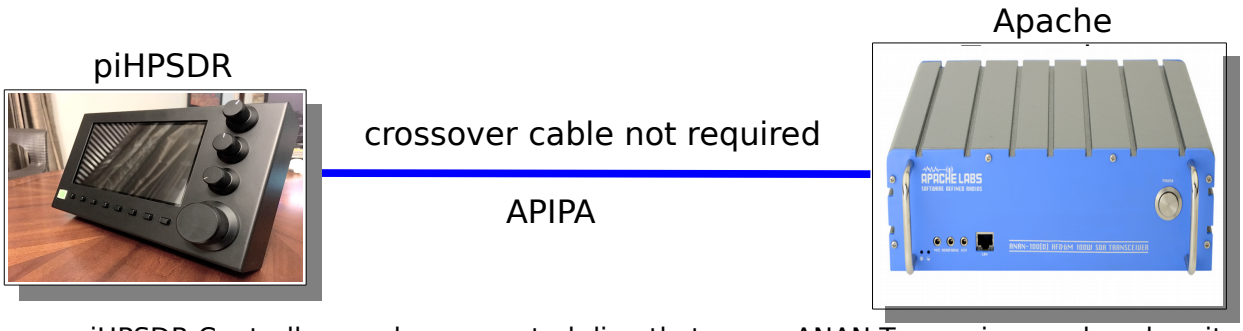
-- John g0orx/n6lyt

Note: This Note: this document assumes familiarity with the Apache Transceiver and standard software such as PowerSDR or Thetis. The Apache Transceiver Users Guides are located here: https://apache-labs.com/al-downloads/1001/ANAN-USER-GUIDES.html

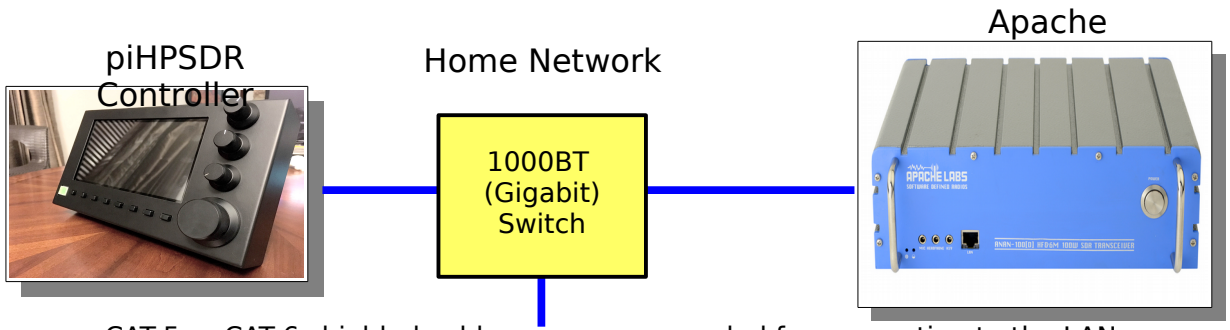
2. Controller connection requirements

Network LAN Connection

Auto-negotiation of a the RaspberryPi 100BT Ethernet connection allows operation of the Controller at the 48000, 96000, and 192Ksps sample rates.



The new piHPSDR-Controller can be connected directly to your ANAN Transceiver and each unit will make use of the APIPA assignment for an IP address.



CAT-5 or CAT-6 shielded cables are recommended for connection to the LAN

Controller Hardware requirements

- 13.8vdc 2.5A (minimum 12vdc 2A) Power connection
- 100BT LAN connection to your Apache Transceiver

Note: when running the dual receiver version of pihpsdr, you may need to place passive heatsinks on the CPU and LAN chip on the RaspberryPi-3b. Be mindful of the temperature alarm shown on the right side of the display screen.

Heatsinks like these:

- https://www.modmypi.com/blog/how-to-install-heat-sinks-on-the-raspberry-pi
- https://www.element14.com/community/community/raspberry-pi/raspberrypi_projects/blog/2016/03/03/raspberry-pi-3-cooling-heat-sink-ideas
- https://www.amazon.com/Addicore-Raspberry-Heatsink-Aluminum-Sinks/dp/B00HPQGTI4/ref=pd_lpo_147_bs_t_2?_encoding=UTF8&psc=1&refRID=6D4ND9MW55Y93DD05V26

Controller software requirements:

- Pihpsdr Controller Operating System and application program are factory installed on an SDHC card.

- If you have purchased the piHPSDR-Controller KIT, you will need to create and install an SDHC card with the RaspberryPi operating system and pihpsdr application.

- Instructions for Installation of Rpi software are detailed in the Install pdf document that is available at:
  https://github.com/g0orx/pihpsdr/raw/master/release/documentation/pihpsdr-install.pdf

- If you wish to get the latest version of pihpsdr source code, executable binaries, and documentation the instructions are available at:
  https://github.com/g0orx/pihpsdr/raw/master/release/documentation/pihpsdr-build.pdf

Note: it is a good idea to make a backup of your Operating System and piHPSDR software. Use the RaspberryPi Menu → Accessories → SD Card Copier utility. Use a new SDHC card of the same capacity as the original.

This is easily done using a USB – SDHC card reader/writer dongle plugged into one of the RPi USB ports on the left side of the piHPSDR Controller.

A pdf document reader is included in the standard RaspberryPi operating system distribution.

3. Front panel controls

Switches **Encoders**

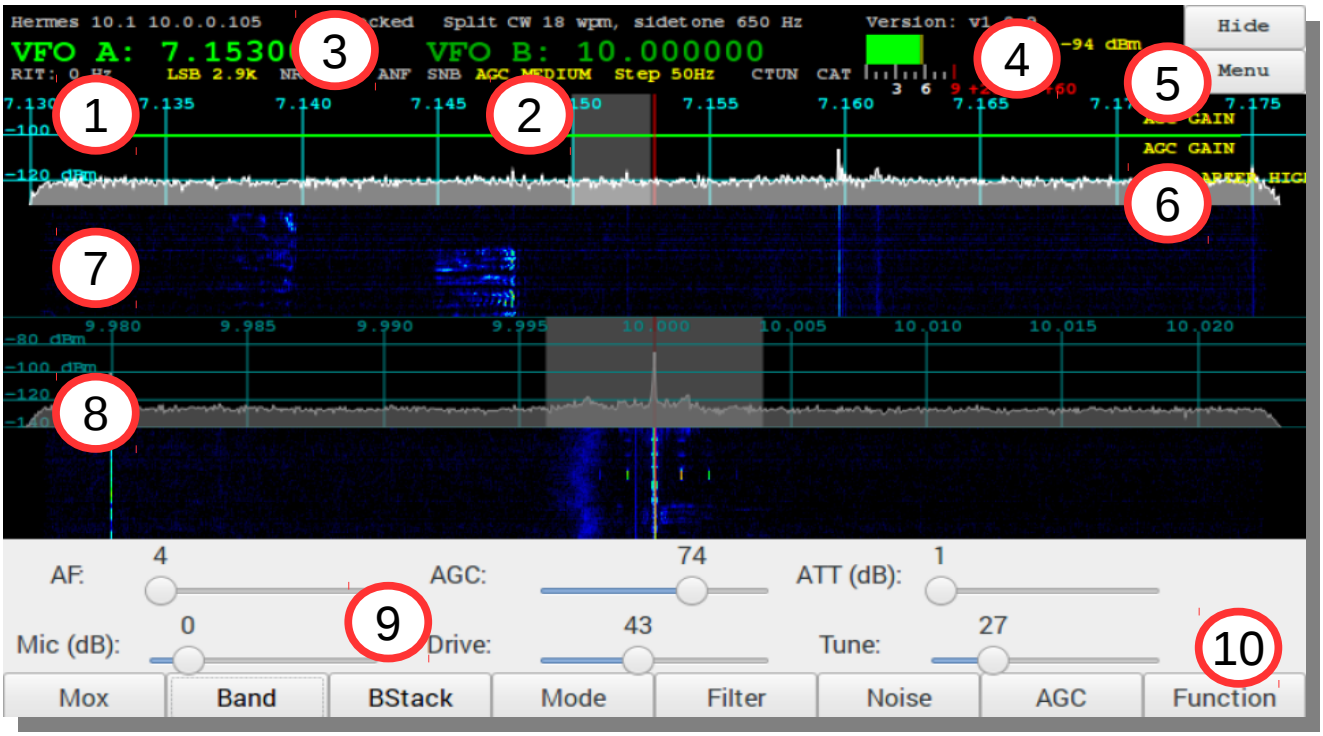
Default assignments of switches and rotary encoders

- **Power** ON/OFF – switches external 12vdc 2A power to the Controller
- **TUN** – MOX/TUNE
- **S1** – Band
- **S2** – Band Stack
- **S3** – Mode
- **S4** – Filter
- **S5** – Noise Blanker
- **S6** – AGC
- **FN** – Function switch to toggle 8 on screen touch buttons such as TUNE or MOX
- **E1** – push encoder knob for selection from Encoder Assignment menu of 11 functions
- **E2** – push encoder knob for selection from Encoder Assignment menu of 11 functions
- **E3** – push encoder knob for selection from Encoder Assignment menu of 11 functions
- **E4** – VFO – main tuning knob

Note: Appendix page 56 has a table showing the Switch, Encoder, and Touch Screen activation details.

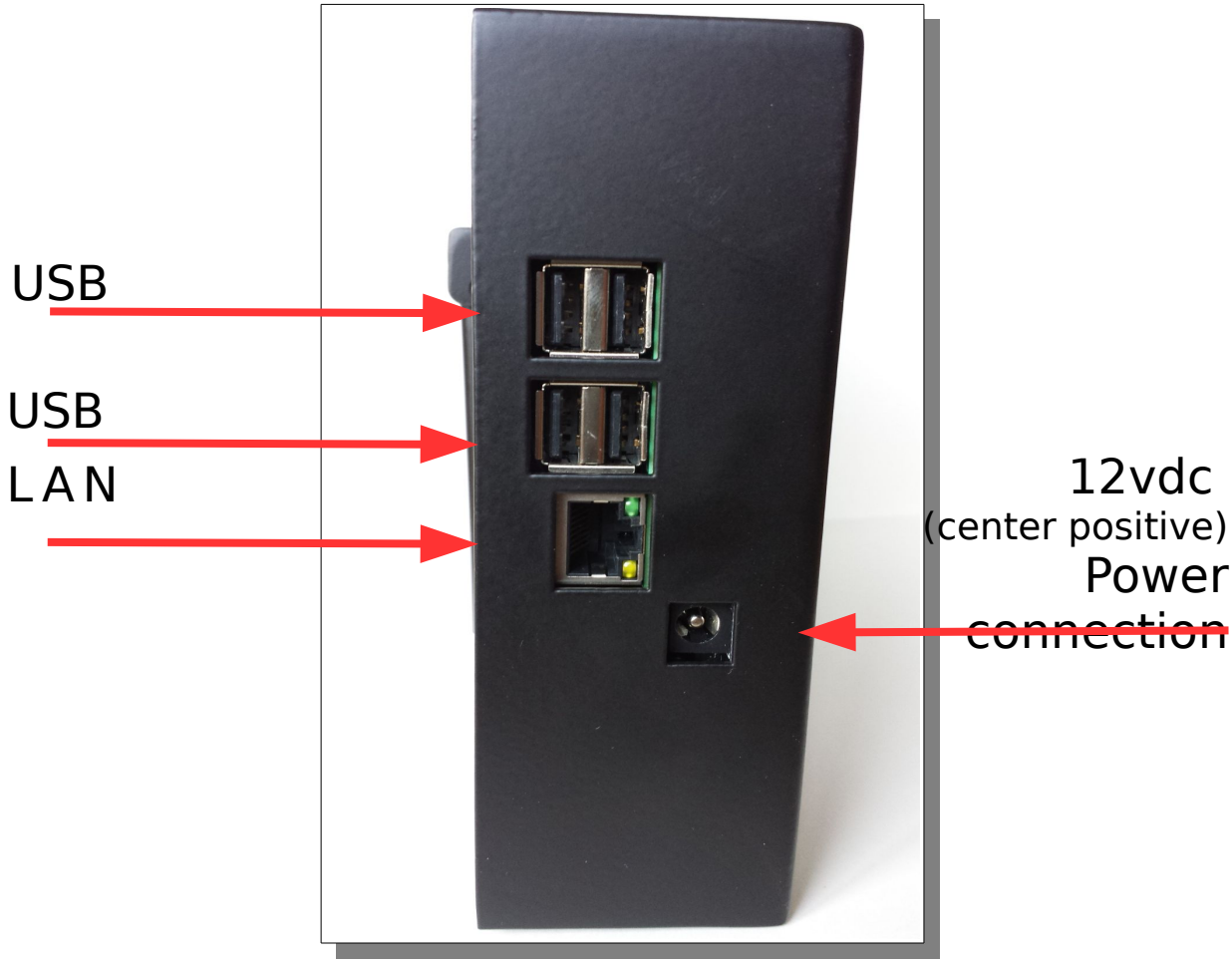
Note: if you are not using a touch sensitive display panel, you may wish to operate pihpsdr from your favorite Linux system. You can use either the RealVNC connection or connect a mouse/keyboard for easy access.

4. Front Panel Display



1. VFO-A Receiver Rx0
2. VFO-B Receiver Rx1
3. Status Lines above and below VFO frequency
4. Meter displays S-Meter and ALC displays
5. Main MENU on screen control
6. Panadapter display Rx0
7. Waterfall Rx0
8. Panadapter display Rx1
9. 6 Slider controls
10. Function Menu activation

5. Side panel connections



6. Quick Start Instructions

please refer to front and back panel illustrations

Hardware Setup

- Carefully unpack the piHPSDR Controller.
- Connect a **CAT 5/6 Ethernet cable** between the rear panel LAN jack with proper access to your Apache Transceiver, Gigabit switch, or suitable router. Please see Network LAN connection page 9.
- Connect the supplied piHPSDR Controller power cable to a **fused 12vdc** typical (13.8vdc) Amateur Radio station power supply.
- **Apache Transceiver connections to MIC, KEY, Headphones**

Software Setup

The piHPSDR Controller software is pre-formatted on the Operating System SDHC card inserted in the RaspberryPi.

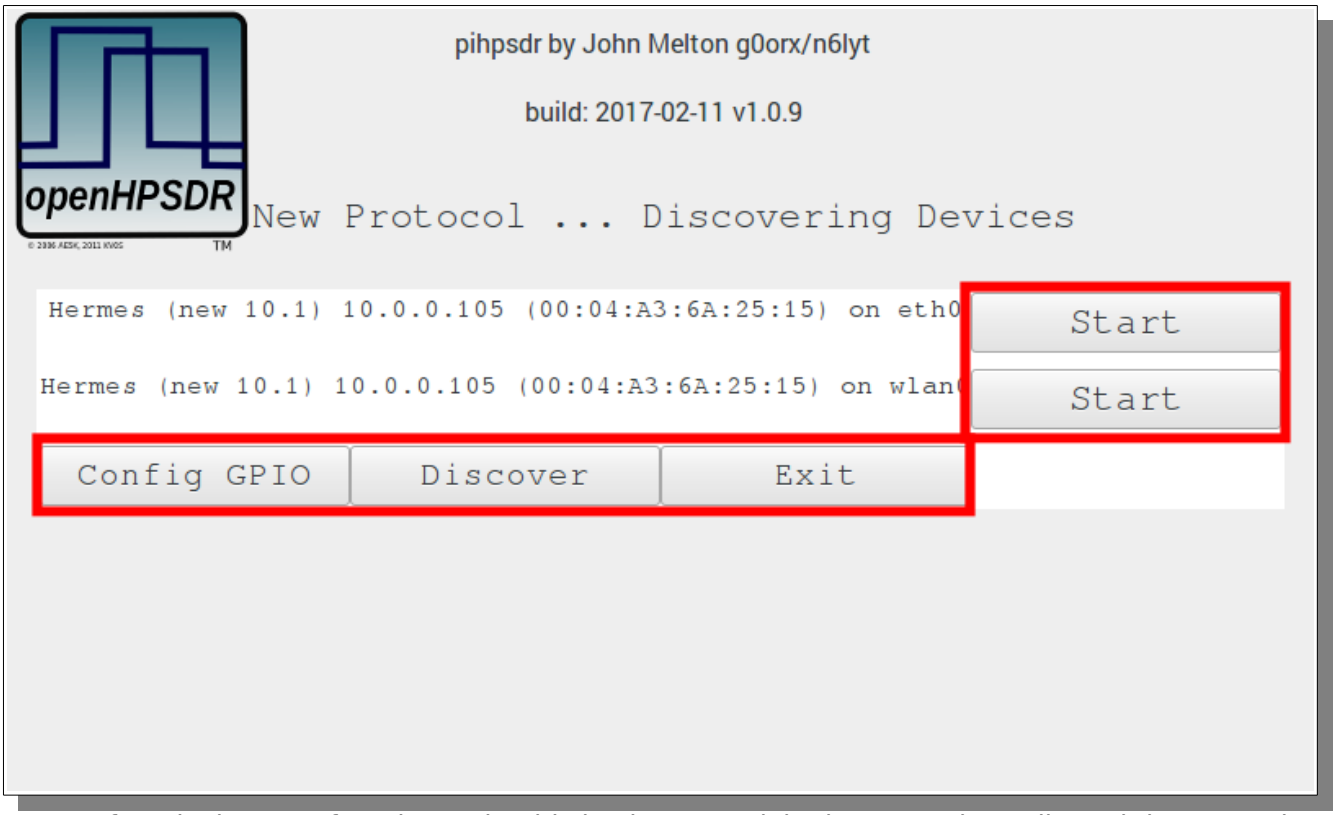
- Alternative software configurations are not provided by Apache Labs.
- A person familiar with the Raspbian operating system may wish to modify or update the contents of the SDHC card using appropriate Linux tools. Please refer to page 10.

Note: pihpsdr is OpenSource. If you would like to compile the program on a Linux system, the source, binaries, and documentation are located at
https://github.com/g0orx/pihpsdr/tree/master/release/documentation

7. Discovery Menu

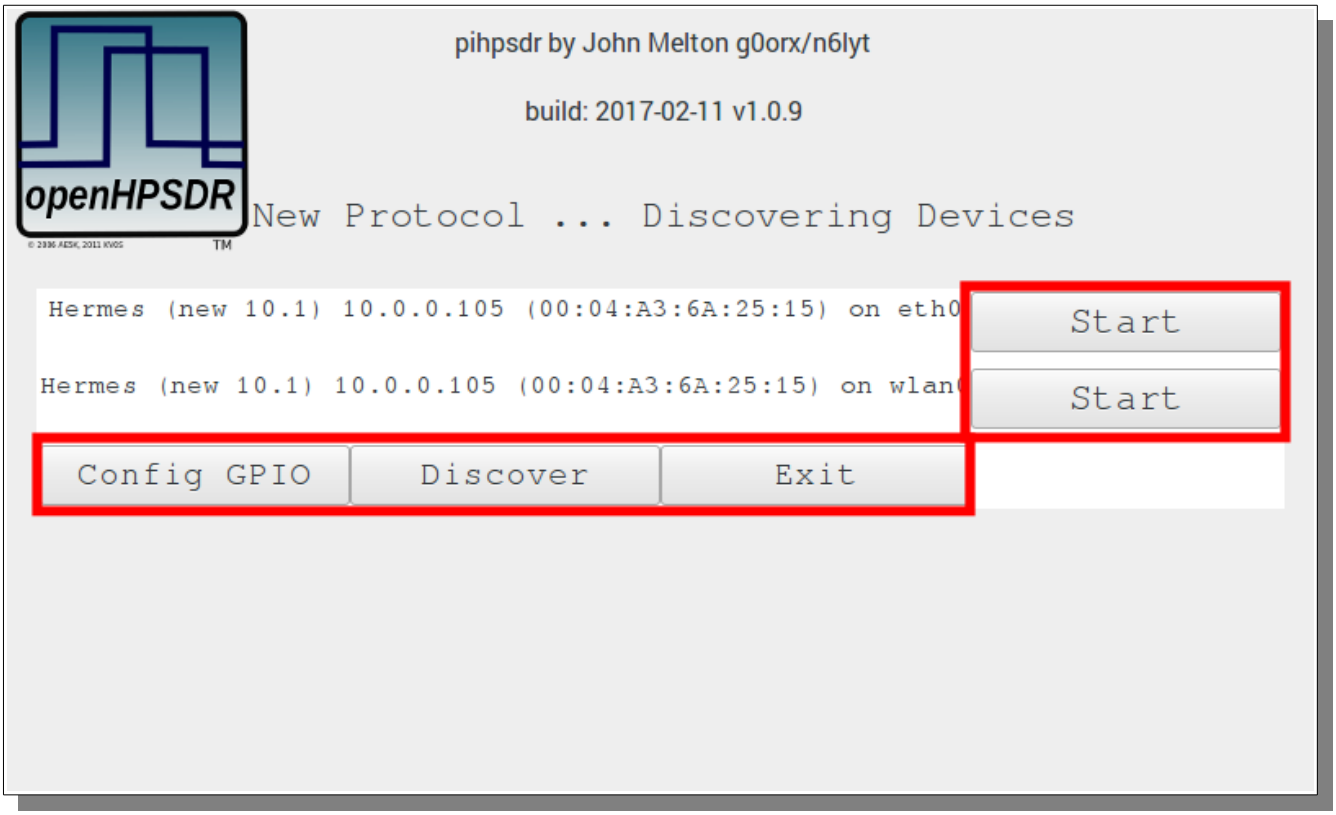
Discover Ethernet Connections to RPi

- When first started, piHPSDR will try to discover all the HPSDR compatible radios on the network. It will look for devices running both the original (old) and the new Ethernet (new) protocol.
- **If one or more Transceiver interfaces are found they will be identified by the device type,** the software version, the IP address and the MAC address of the device.



- If no devices are found you should check connectivity between the radio and the network that piHPSDR is connected to. Tapping on OK will try discovery again. Tapping on Cancel will exit piHPSDR back to the Raspberry Pi desktop.
- The Discover protocol will allow a device to see and respond even if they are not on the same subnet. If this is the case, the Start button will be disabled and the text replaced with **Subnet** The most common cause of this problem is usually that the radio has not been able to get a DHCP address and has defaulted to the Self Assigned IP Address, or the device has a static IP address that is not on the same subnet as the piHPSDR.

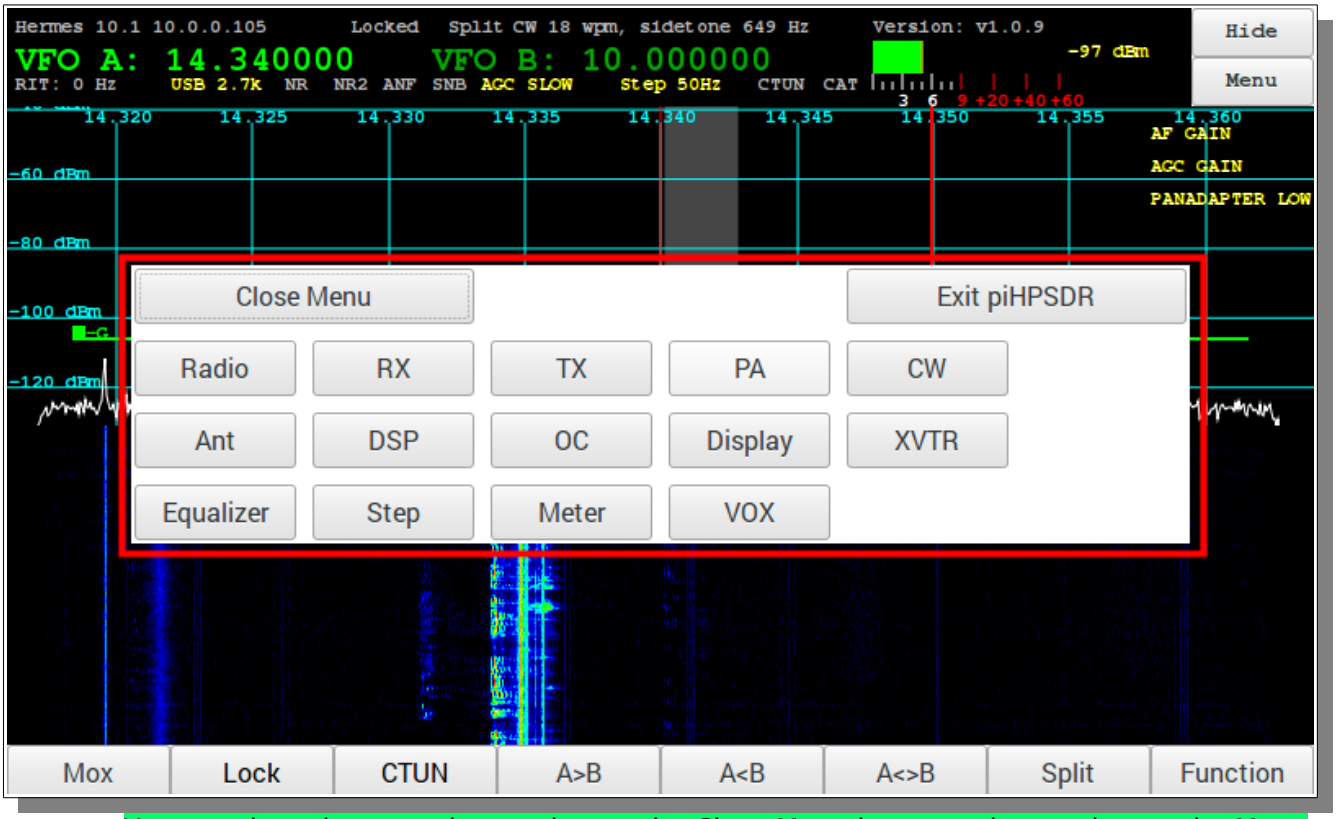
Pihpsdr program START button



- **Tapping on the Start button will start the radio – on the selected Ethernet interface**.
- In this example you see **both** the Rpi **Wireless** and **Ethernet** connections are active and were discovered properly.
- Depending on your particular WLAN and LAN setup, either will work well with pihpsdr 100Mbps network requirement. **Both the Rpi and your Apache Transceiver must be on the same subnet.**
- Configuration of the **GPIO** default connections are shown on page **57**

8. Main Menu

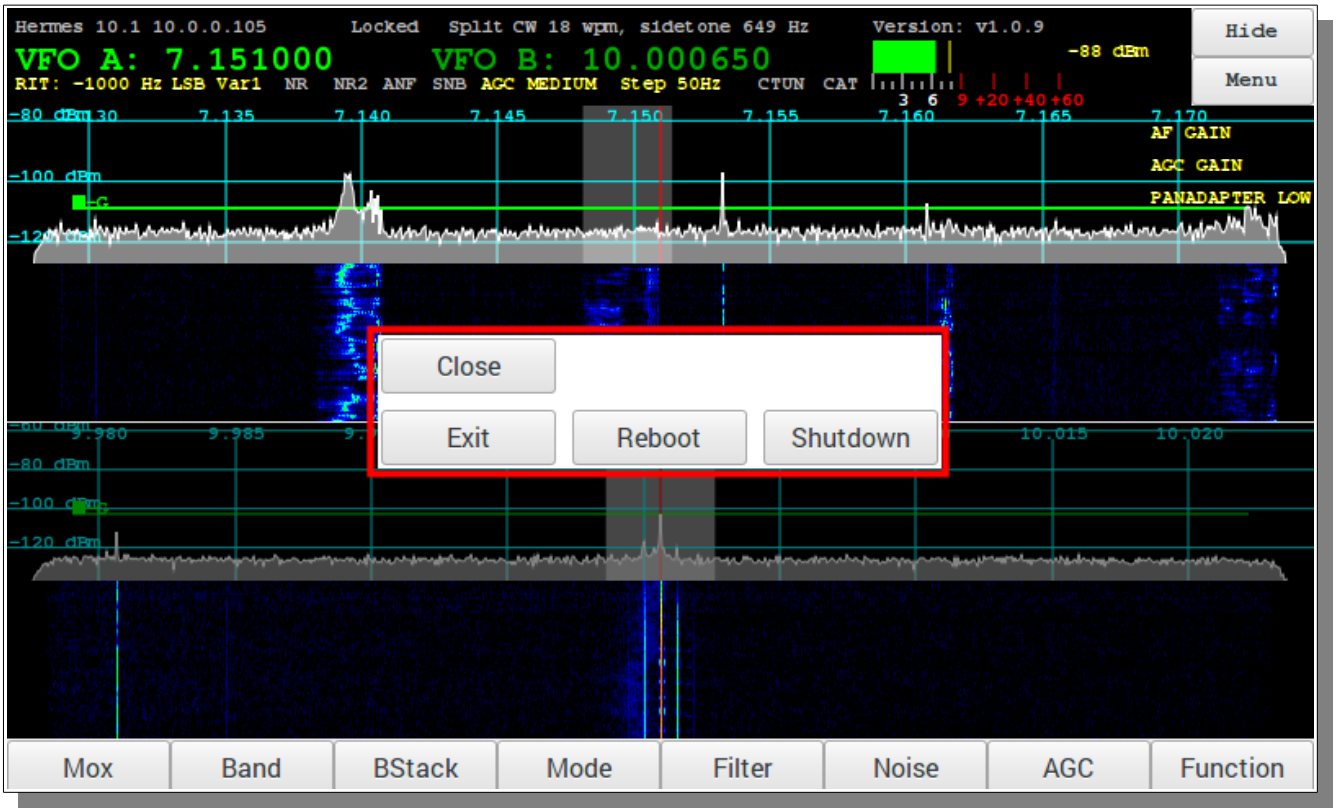
Tapping the **Menu button** brings up the menu dialog.



You can close the menu by tapping on the Close Menu button or by tapping on the Menu button again.

- Radio – see page 16
- RX – see page 17
- TX – see page 19
- PA – see page 21
- CW – see page 22
- Ant – see page 23
- DSP – see page 24
- OC – see page 25
- Display – see page 26
- XVTR – see page 28
- Equalizer – see page 29
- Step – see page 30
- Meter – see page 31
- VOX – see page 32

Menu → Exit piHPSDR

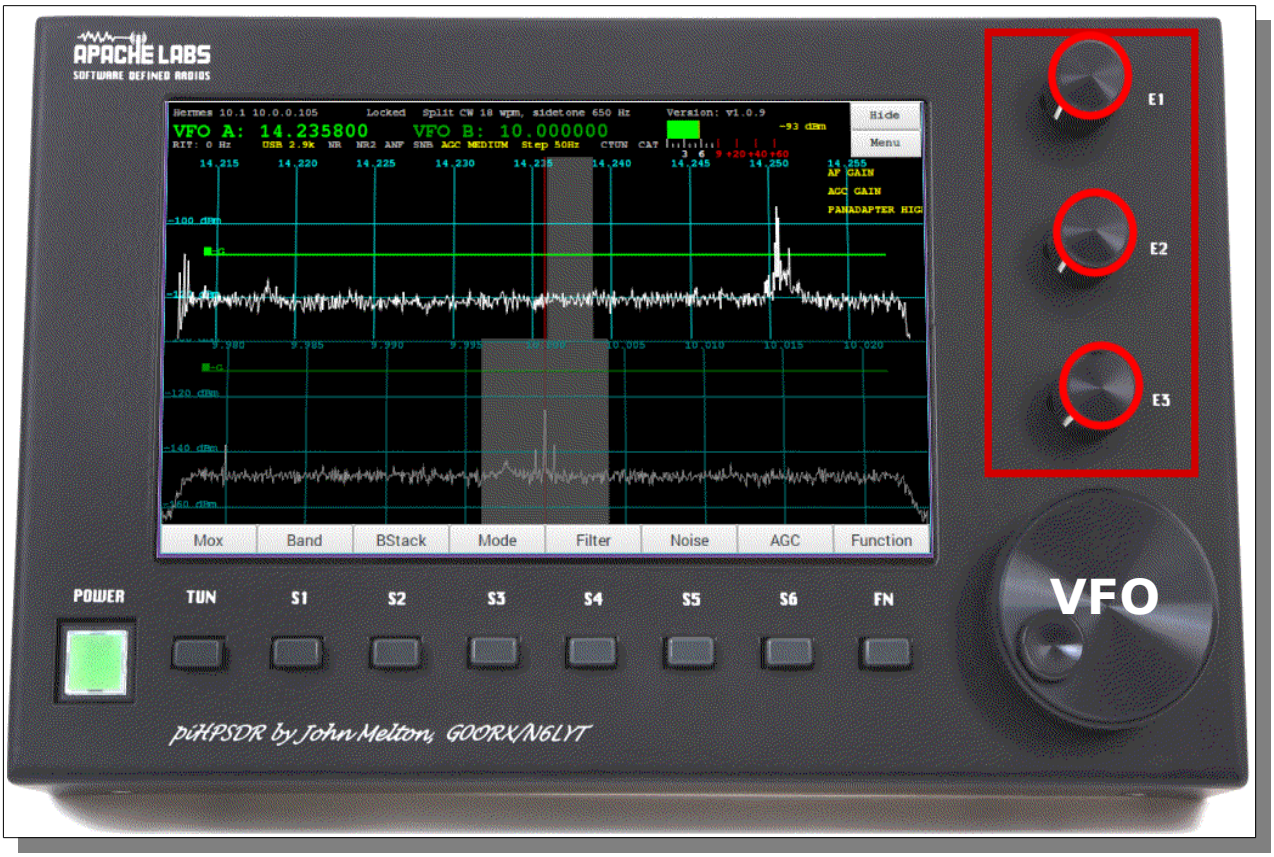


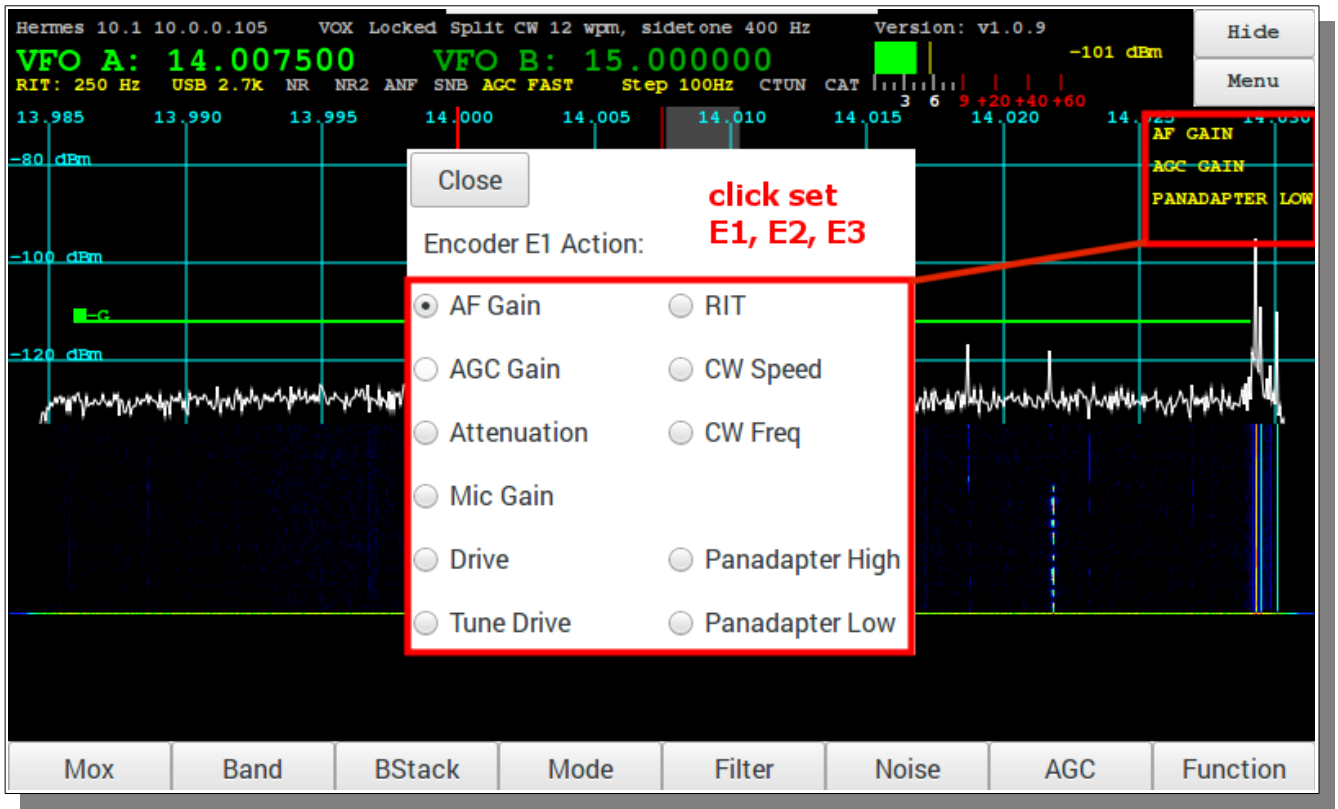
The Exit piHPSDR button will bring up a sub-dialog allowing you to end the piHPSDR application in a number of ways:

- **Close –** close this dialog box
- **Exit** – exit piHPSDR back to the Raspberry Pi Desktop
- **Reboot** – reboot the Raspberry Pi
- **Shutdown** – Shutdown the Raspberry Pi - (note the Controller power button remains lighted)

You can also tap the Menu button again to close the Exit Menu and bring up the Menu selection.

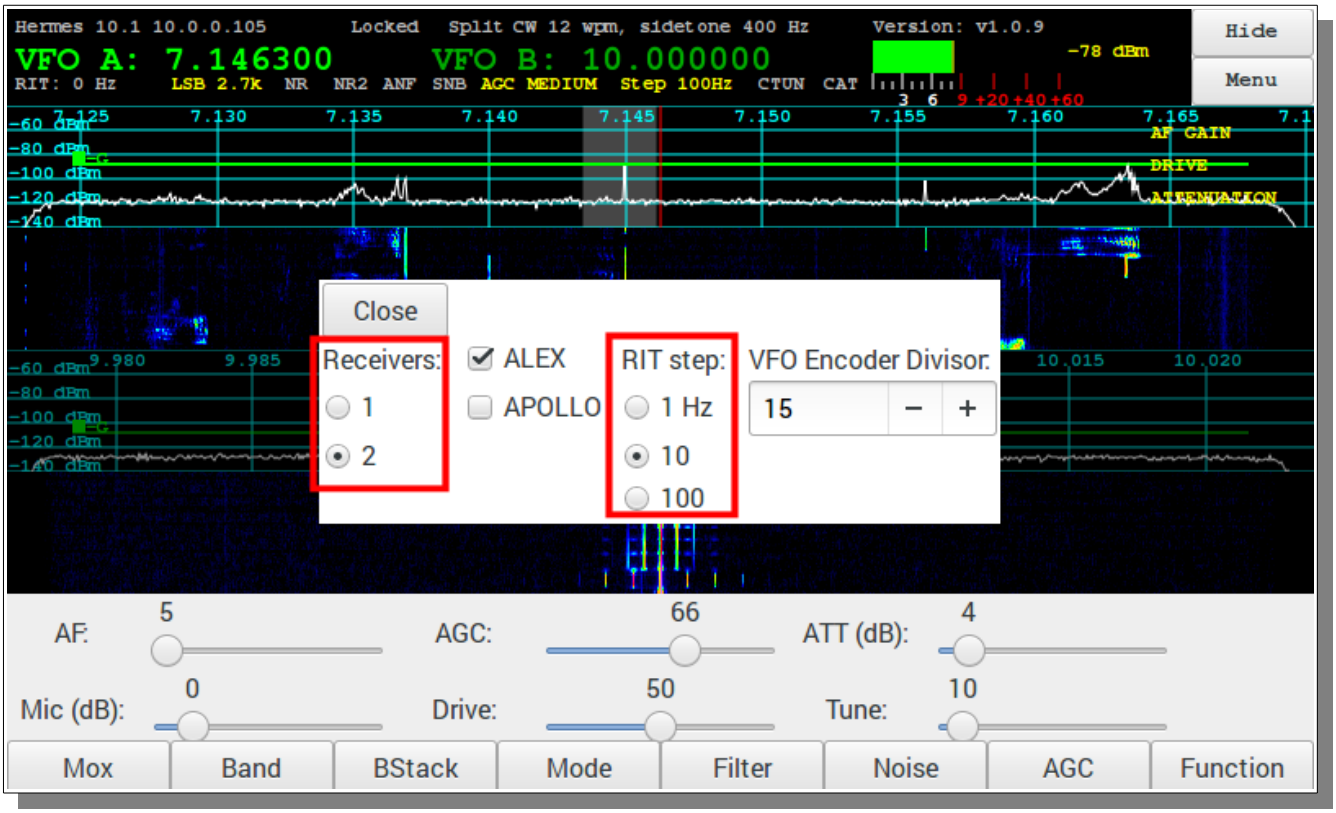
E1, E2, E3 Click-Set Assignments





Click-Set Assignments

Menu → Radio



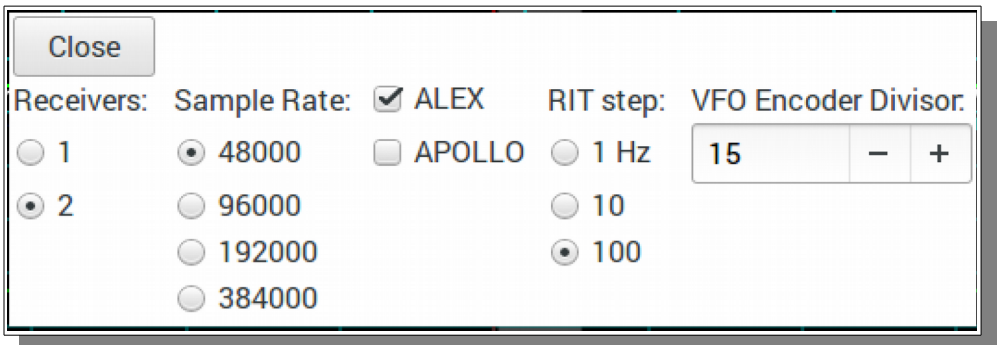
Illustrates Menu → **Radio using the New Protocol**

- **Receivers -** changing pihpsdr to 1 or 2 receivers.
- **RIT Step size** – This value is the increment or step size for the RIT control and mouse wheel tuning.
- **VFO Encoder Divisor** is set at the factory to 15

  set to 15, 1 revolution of the (600ppr) encoder = 4kHz/revolution

  setting the Encoder Divisor to 30 slows the VFO Encoder tuning to 2kHz/revolution.

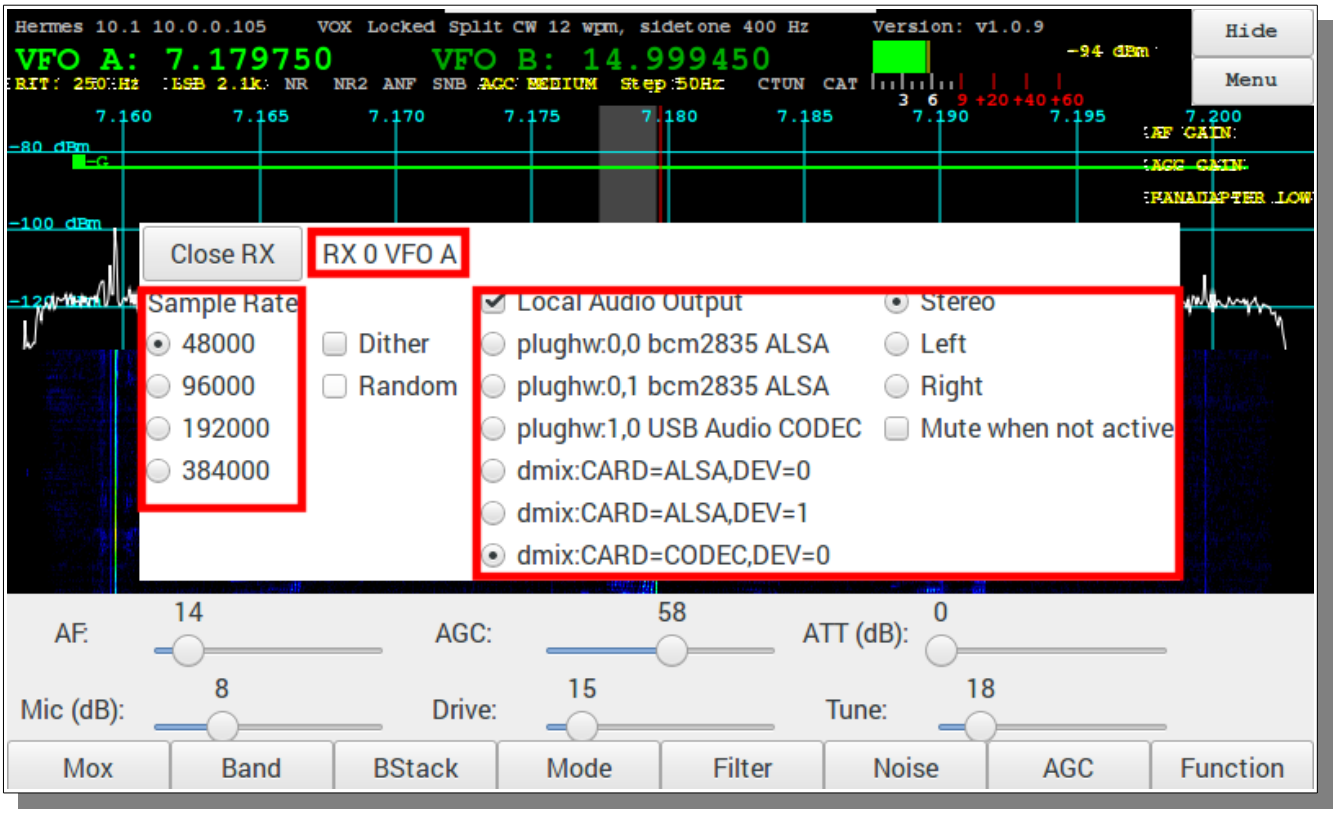
  setting the Encoder Divisor to 7 increased the VFO Encoder tuning to 9kHz/revolution.



This illustration shows the Menu → Radio when using the **original protocol (USB)**

Menu → Rx
(individual receiver settings)

When you touch or select either PANADAPTER you change to that VFO and the RX menu changes from Rx0 to Rx1.



- **Sample Rate** - The Sample Rate selection selects the width of the Panadapter displayed on the piHPSDR Controller screen. This is the rate which **pihpsdr** uses to decode a portion of the 60Mhz spectrum data from the Apache Transceiver.

  Note: Original Protocol only provides for an identical sample rate for both receivers, shown in the Menu → Radio

  "New Ethernet Protocol" in Beta test Receivers can have individual sample rates, shown in the Menu → Rx

  If your Apache Transceiver has been updated to the 2017 firmware, you can change the sample rate of each receiver independently. This is a feature of the "New Ethernet Protocol" firmware. Independent Receiver sample rates have been implemented in the February 2017 pihpsdr software release.

  Note: 768k and 1.536K are not available on Rpi.

- **Dither** - Dither is a built-in electronic feature of the Linear Technologies LTC-2208 Analog to Digital Converter chip inside each Apache Software Defined Radio Transceiver.
- **Random** - Random is a built-in electronic feature of the Linear Technologies LTC-2208 Analog to Digital Converter chip inside each Apache Software Defined Radio Transceiver.
- **Local Audio Output** – This column identifies all the audio output connections that pihpsdr recognizes when it starts.

  By default the audio output from the receiver is sent back to the radio for output to the audio connections on the radio.

You can also output the audio to either the audio output connector on the Raspberry Pi or a USB connected audio device.

To enable output to a selected device check the **Local Audio Output** check box. The output will be directed to the selected device.

plughw:0,0 bcm2835 ALSA is the Raspberry Pi output to the audio connector.

plughw:0,1 bcm2835 ALSA is the Raspberry Pi output to the HDMI interface.

plughw:1.0 USB Audio CODEC is a USB connected audio device.

Only one output stream can be directed to the devices listed above, but the ALSA sound system includes a mixer that lets multiple streams output to a single device.

dmix:CARD=ALSA,DEV=0 is a mixer for the **plughw:0,0 bcm2835 ALSA** output

dmix:CARD=ALSA,DEV=1 is a mixer for the **plughw:0,1 bcm2835 ALSA** output

dmix:CARD=CODEC,DEV=0 is a mixer for the **plughw:1.0 USB Audio CODEC** output

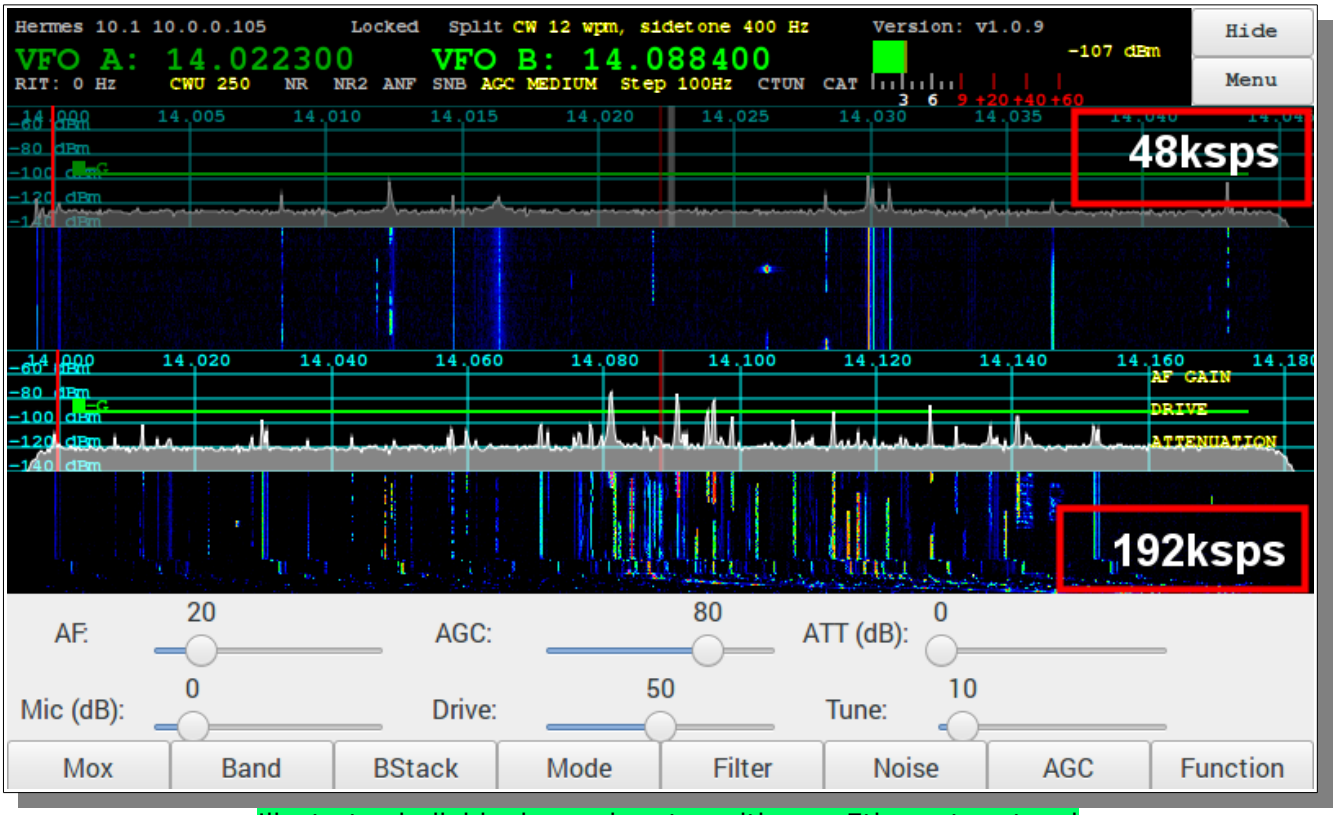
- **Stereo, Left, Right, Mute when not active**

If you select the same **dmix** device for both receivers, the output from both will be mixed and sent to that device. You can select that the receivers output is either Stereo, Left or Right to select the channel(s) that the audio is sent on. This would allow one receiver inn the Left headphone/speaker and the other receiver in the Right headphone/speaker.

The **Mute when not active** when selected will simulate the output to the radio where only the active receiver is output to the device.

Note that if one receiver's audio is set to output to the plughw device the other receiver cannot be connected to the dmix device for that audio output.

Menu → Rx (continued)

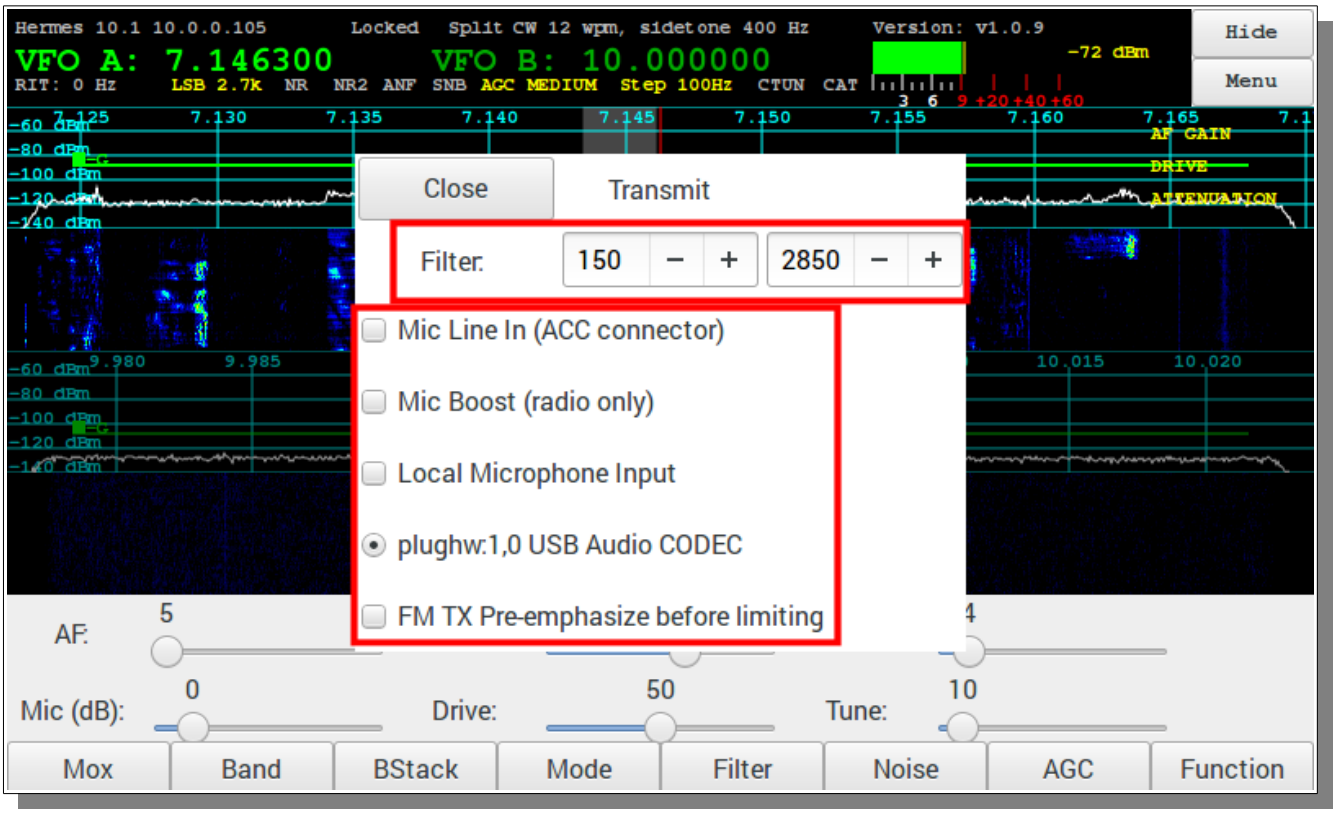


Illustrates individual sample rates with new Ethernet protocol

Note: Original Protocol only provides for an identical sample rate for both receivers, shown in the Menu → Radio

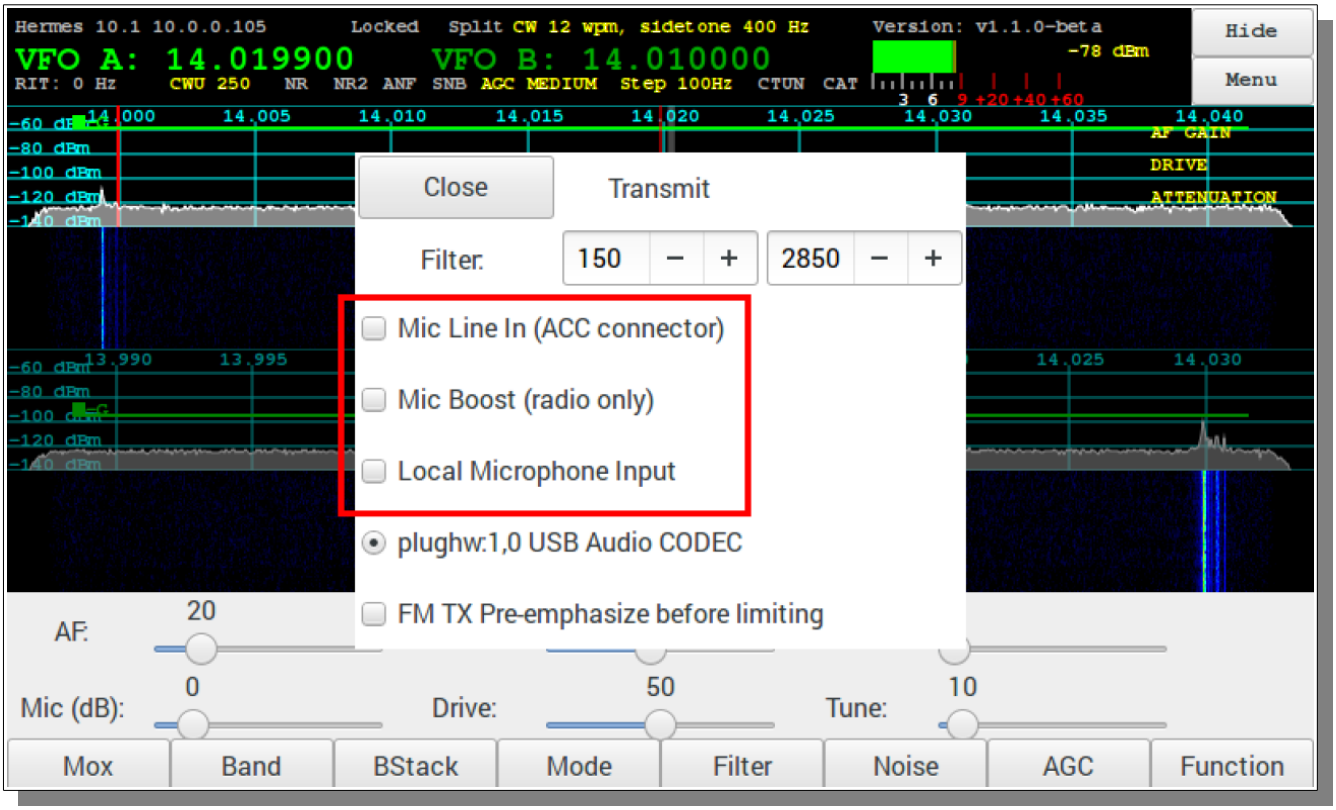
“New Ethernet Protocol” in Beta test Receivers can have individual sample rates, shown in the Menu → Rx

Menu → TX with USB Input Source



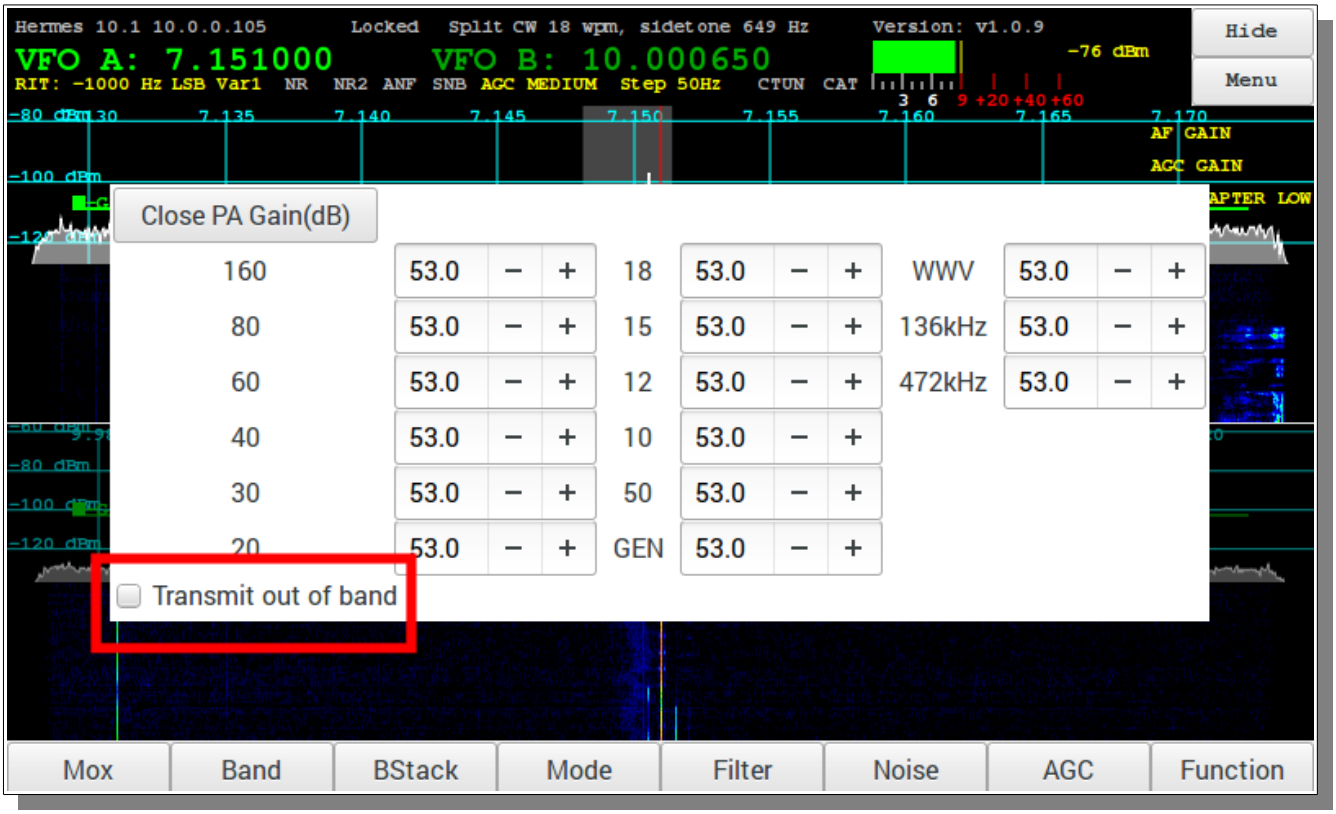
- **Local Microphone Input** – Local Audio refers to the soundcard MIC connection (shown here USB)
- Filter indicates the upper and lower frequencies for the Transmit bandwidth.
- Audio CODEC shown here is an external USB Audio **INPUT** device connected to the Rpi.
- FM TX Pre-Emphasis may be required for normal voice quality communications by the FM Server you are connecting to.

Menu → TX Transceiver MIC input



- With Local Microphone Input unchecked, the TX menu changes to above illustration where you can select only Apache Transceiver input sources.

Menu → PA Gain by Band

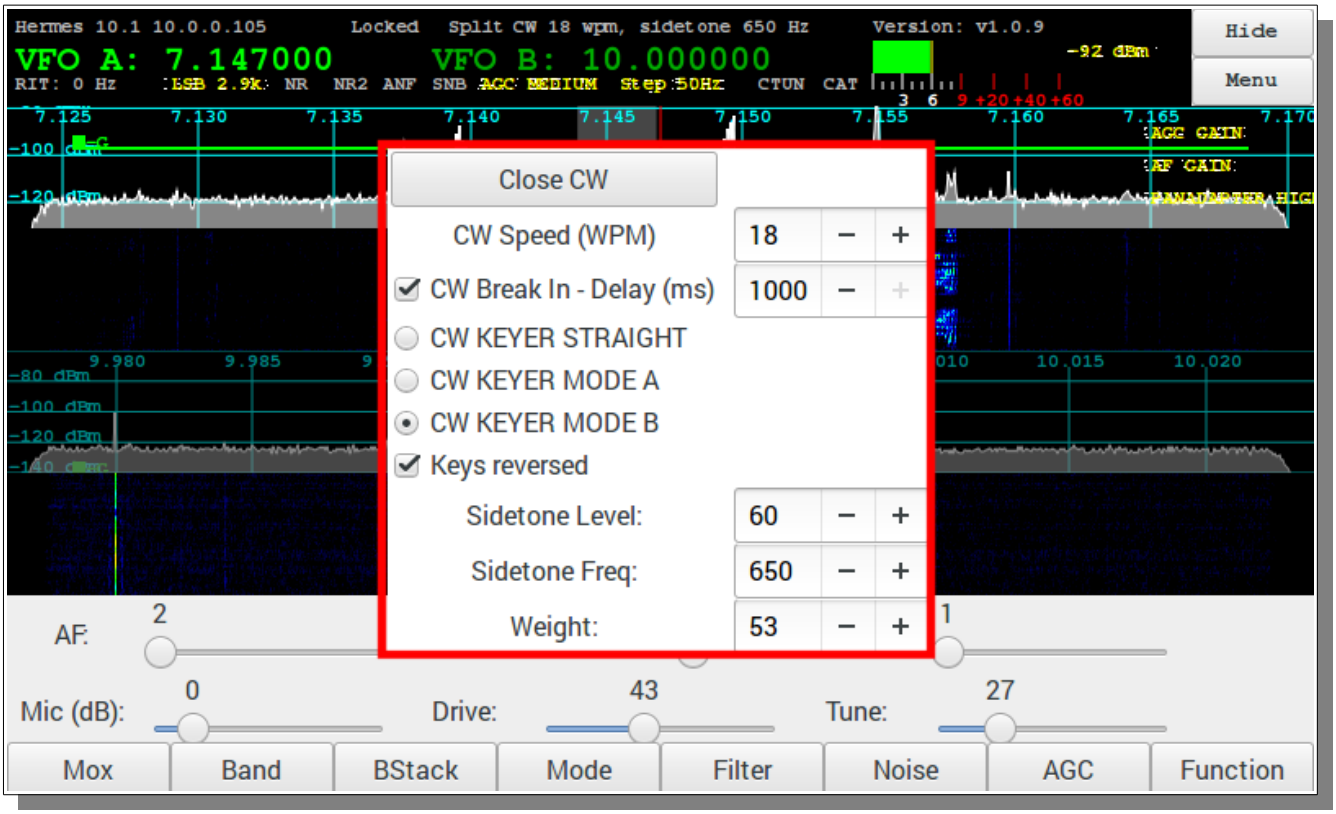


- Sets the PA gain for each band by adjusting the drive level. This adjustment allows you to set the Transceiver for maximum rated Transmit output.
- the setting are **increased** the drive level is **decreased**. This adjustment is set for maximum rated output protect the PA Final Amplifier.

Caution: The User should follow the instructions for PA Gain settings in PowerSDR/Thetis. These instructions define setting the TUNE POWER to the Maximum Rated Output while adjusting the PA Gain settings. Operationally the user can use the pihpsdr Drive slider to adjust for a lower output power – for example when driving a Linear Amplifier.

- Note**: Transmit out of band** – enables transmit outside of ham bands. Default disabled.

Menu → CW

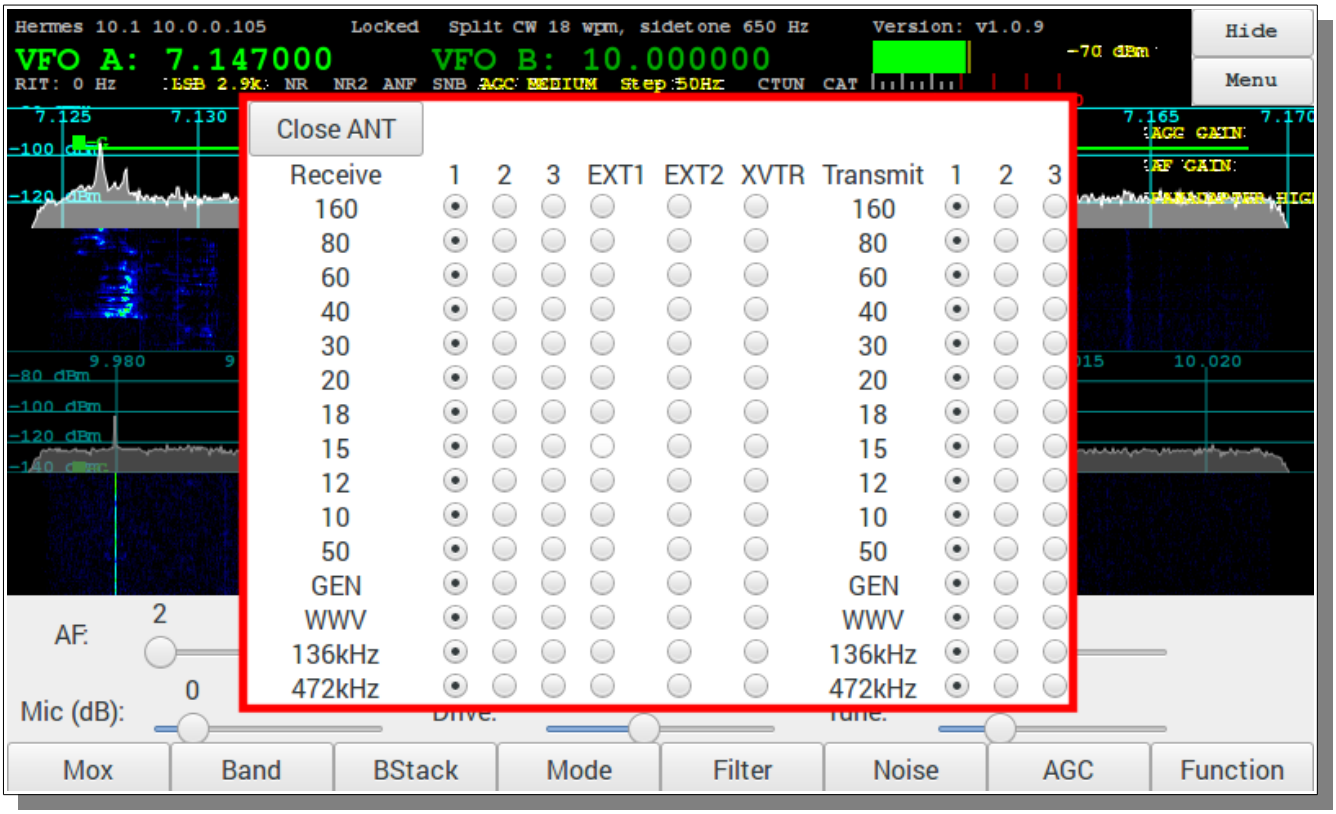


Controls the firmware CW Keyer.

- **CW Speed** (WPM) – sets speed of dot/dash generator when in Mode A or Mode B.
- **CW Break In** – when enabled sets the delay time in milliseconds to switching to receive.
- **CW KEYER STRAIGHT** – selects the key connected will be a straight key.
- **CW KEYER MODE A** – selects a paddle key running in Mode A
- **CW KEYER MODE B** – selects a paddle key running in Mode B
- **Keys reversed** – when enabled the dot/dash paddles are reversed.
- **Sidetone Leve**l – sets the audio level of the sidetone at the Headphone or Line Out of the Apache Transceiver.
- **Sidetone Freq** – set the frequency of the sidetone at the Headphone or Line Out of the Apache Transceiver.
- **Weight** – sets the dot/dash weighting.

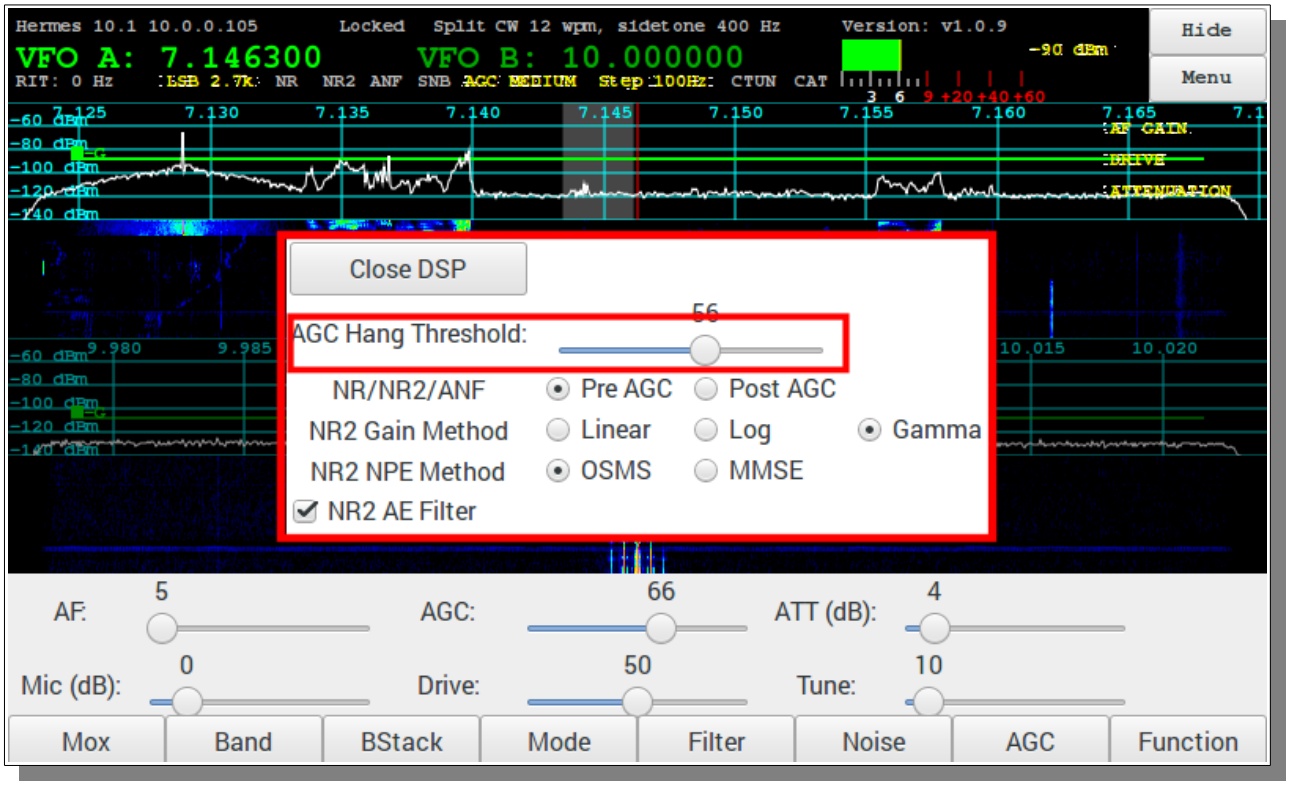
Menu → ANT

The Ant menu selects which antenna is used for receive and transmit on each band.



- The Antenna Menu will differ depending on the specific Apache Transceiver Model. This illustration shows the user selection of up to 3 Receive and Transmit Antennas. The EXT-1 and EXT-2 selections are available on the ANAN-100/100B/100D/200D.

Menu → DSP



The DSP has options for the DSP functions.

- **AGC Hang Threshold** – sets the Hang Threshold for the AGC.

- **NR/NR2/ANF** – selects where in the DSP processing the noise reduction functions are performed/ The default id Pre AGC processing.

  Pre AGC – perform noise reduction pre AGC

  Post AGC – perform noise reduction post AGC

- **NR2 Gain Method** – selects the method used for the gain processing. The default is Gamma.

  Linear - Gaussian speech distribution, linear amplitude scale

  Log - Gaussian speech distribution, log amplitude scale

  Gamma - Gamma speech distribution

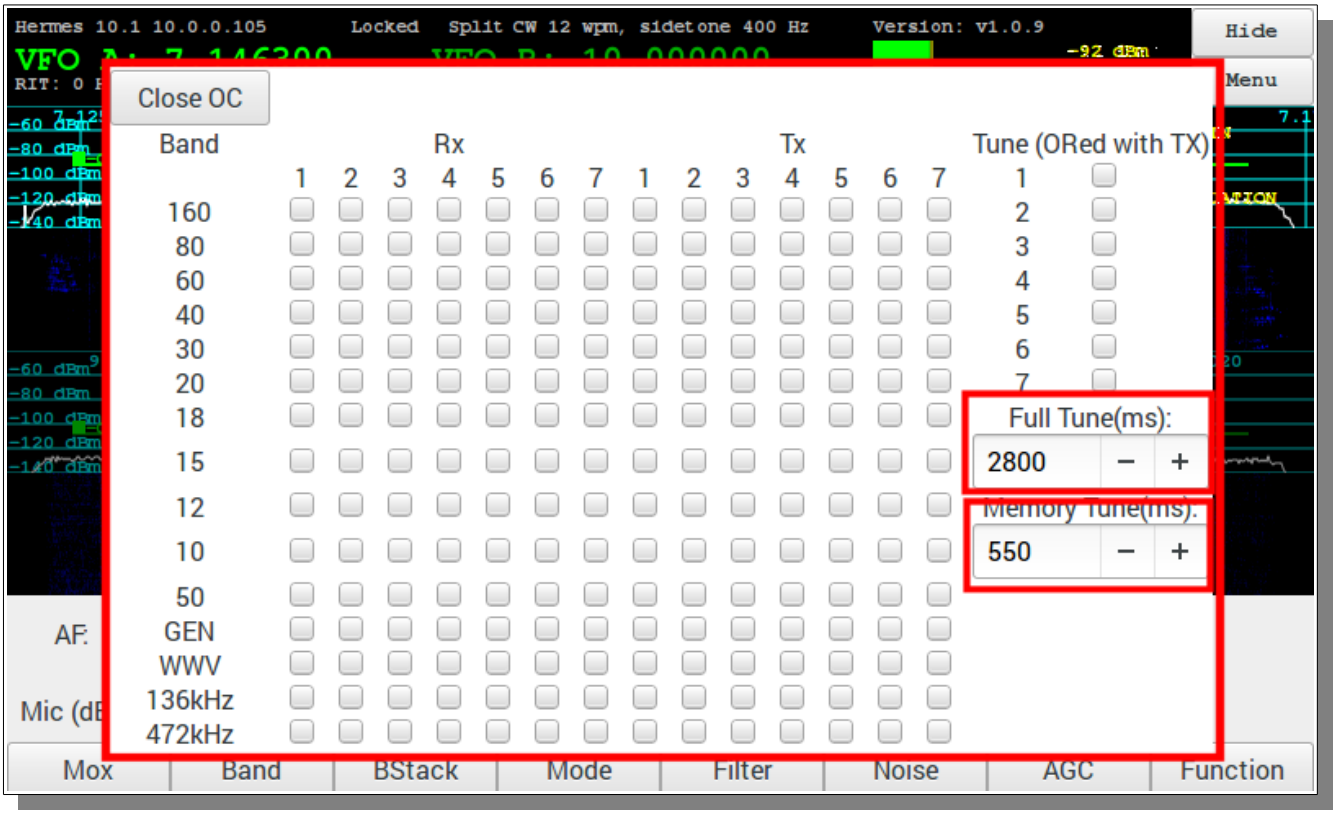
- **NR2 NPE Method** – selects the Noise-Power-Estimation method. The default is OSMS.

  OSMS - Optimal Smoothing Minimum Statistics

  MMSE - Minimum Mean -Square Error

- **NR2 AE Filter** - Enable Artifact elimination. Default enabled.

Menu → OC Open Collector Aux I/O connectors



The user can configure the Open Collector outputs for each band for both Transmit and Receive. These can be used to control an external device such as bandpass filters or external Linear Amplifier band selection. The default is none are enabled.

Refer to the Apache Labs Users Guides for details and limitations of using Open Collector FET outputs.

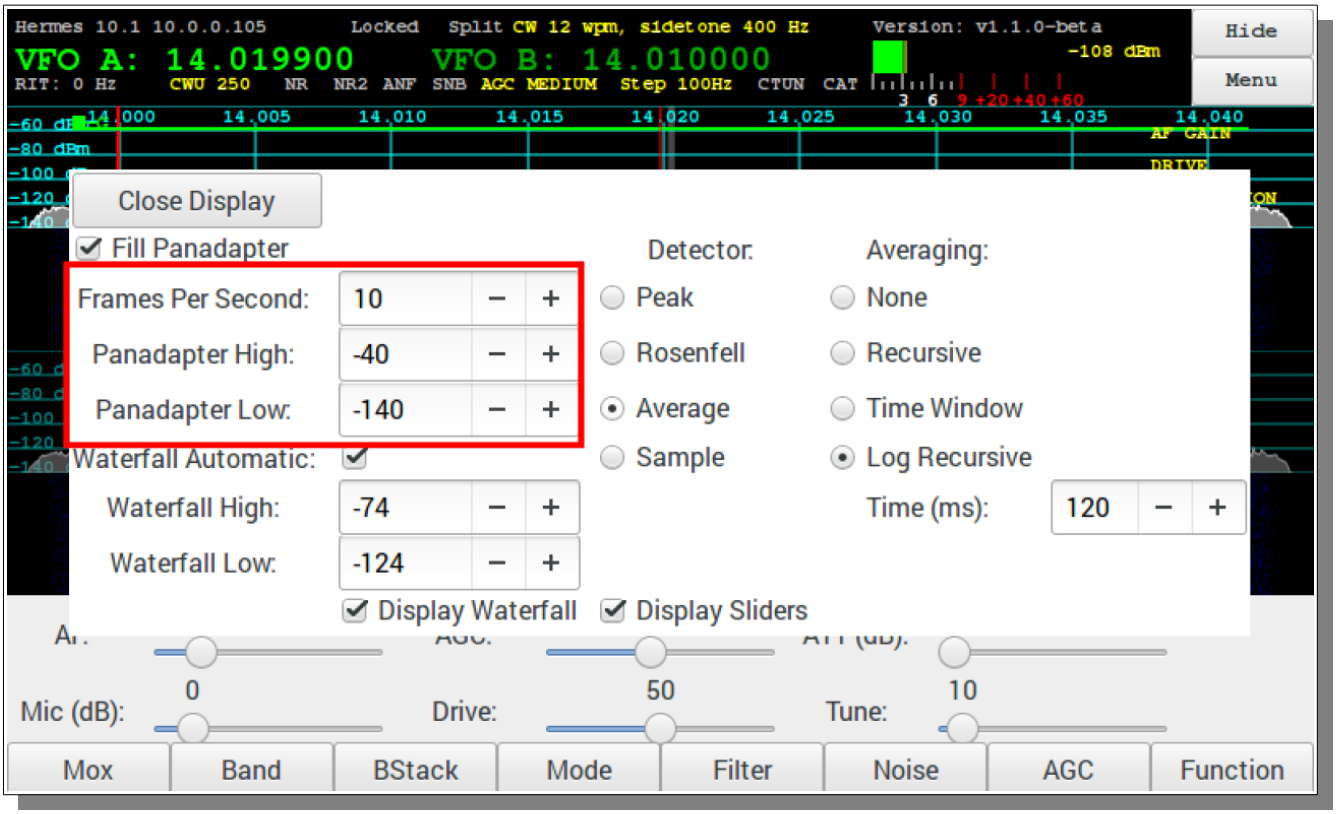
When an external ATU is used the Tune option can be used to signal to the ATU to start its tune function. The default is “none are enabled”.

- **Tune** – configure an OC to be turned on when Tx is enabled.

A future option is planned to enable the following OC controls.

- **Full Tune milliseconds** – specifies the time the OC is enabled when the Tuning.
- **Memory Tune milliseconds** – keydown time for TUNE when using an external Automatic Antenna Tuner

Menu → Display



- **Fill Panadapter** – when enabled the panadapter graph will be filled. When not enabled it will be drawn as a line. Default enabled.

- **Frames Per Second** – Update rate of Panadapter and Waterfall

- **Panadapter High** - Maximum signal level displayed in Panadapter

  Note: now included as a sub-menu selection for E1, E2, E3 Encoder function assignment

- **Panadapter Low** – Minimum signal level displayed in Panadapter

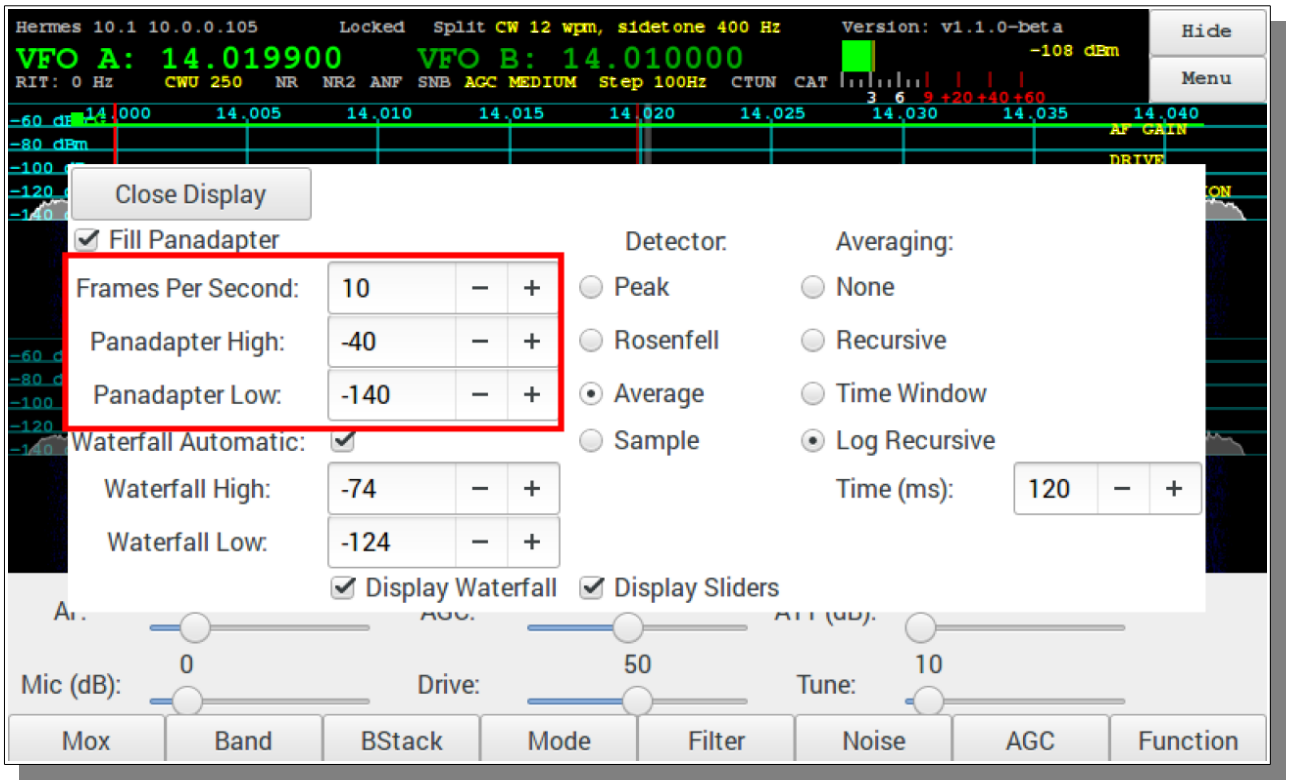
  Note: now included as a sub-menu selection for E1, E2, E3 Encoder function assignment

- **Waterfall Automatic** – When enabled the Waterfall High and Waterfall Low are adjusted automatically.

- **Waterfall High** – Manual control to set the maximum signal used in the waterfall.

- **Waterfall Low –** Manual control to set set the minimum signal used in the waterfall.

Menu → Display (continued)



- **Detector** – Selects Peak, Rosenfell, Average or Sample for the Panadapter display.
- **Averaging** – Selects the method for averaging the Panadapter display.
- **Display Panadapter** – when selected the Panadapter is displayed on the main screen.
- **Display Waterfall** – when selected the Waterfall display is displayed on the main screen.
- **Display Sliders** – when selected the slider controls are displayed on the main screen.

Menu → XVTR

Configure up to 8 transverters.

- **Title** – the name as it appears in the Band, Ant and OC menus.
- **Min Feq** – The minimum frequency in Hz.
- **Max Feq** – The maximum frequency in Hz.
- **LO Feq** – The Local Oscillator frequency in Hz.
- **Disable PA** – When checked, the Transceiver power amplifier will be disabled on transmit.

Note that the frequency the radio is tuned to is the selected frequency minus the LO frequency. In the example above the 144MHz to 146MHz Transverter frequency will be tuned to 28MHz to 30MHz on the radio.

When one or more Transverters are configured they will appear in the band selection dialog and also in the ANT selection menu and the PA Gain menu.

Menu → Equalizers

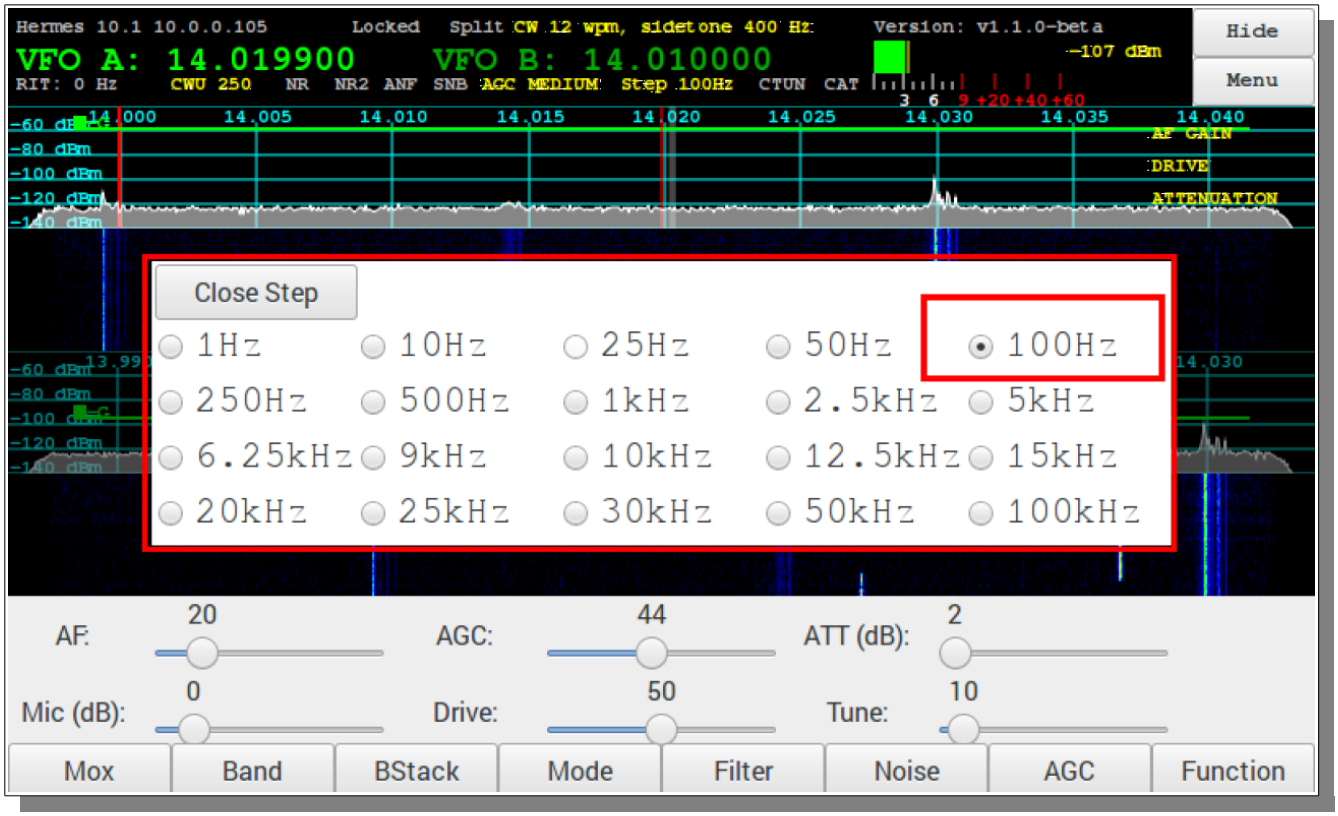
A 3 band graphic **equalizer** is implemented for **both Transmit and Receive**:

- **Preamp**
- **Low** – 0-400 Hz slider dB
- **Mid** – 400-1500 Hz slider dB
- **High** – 1500-6000 Hz slider dB

The radio buttons **TX Equalizer** and **RX Equalizer** select which values are displayed.

The equalizers can be enabled by the checkbox **Enable TX Equalizer** or **Enable Rx Equalizer**.

Menu → Step

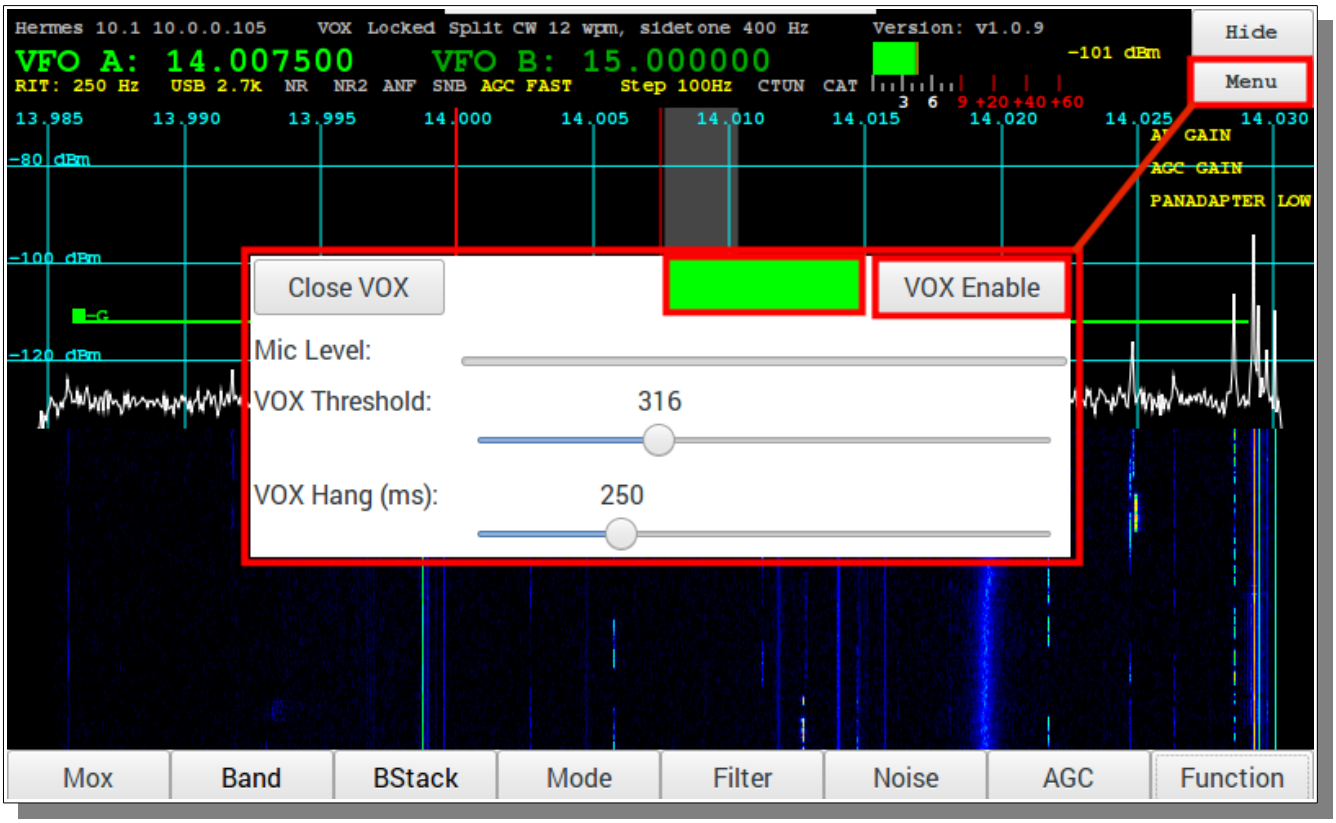


- **Step** sets the increment for **VFO Tuning rate** via Touch or VFO (E4) Encoder, and Mouse Wheel.

Menu → Meter

- S-Meter Peak and Average
- ALC Peak Average or GAIN
- Meter values appear in the upper right hand corner of the display.
- You should adjust your Microphone (dB) slider so that ALC does not exceed zero on voice peaks.

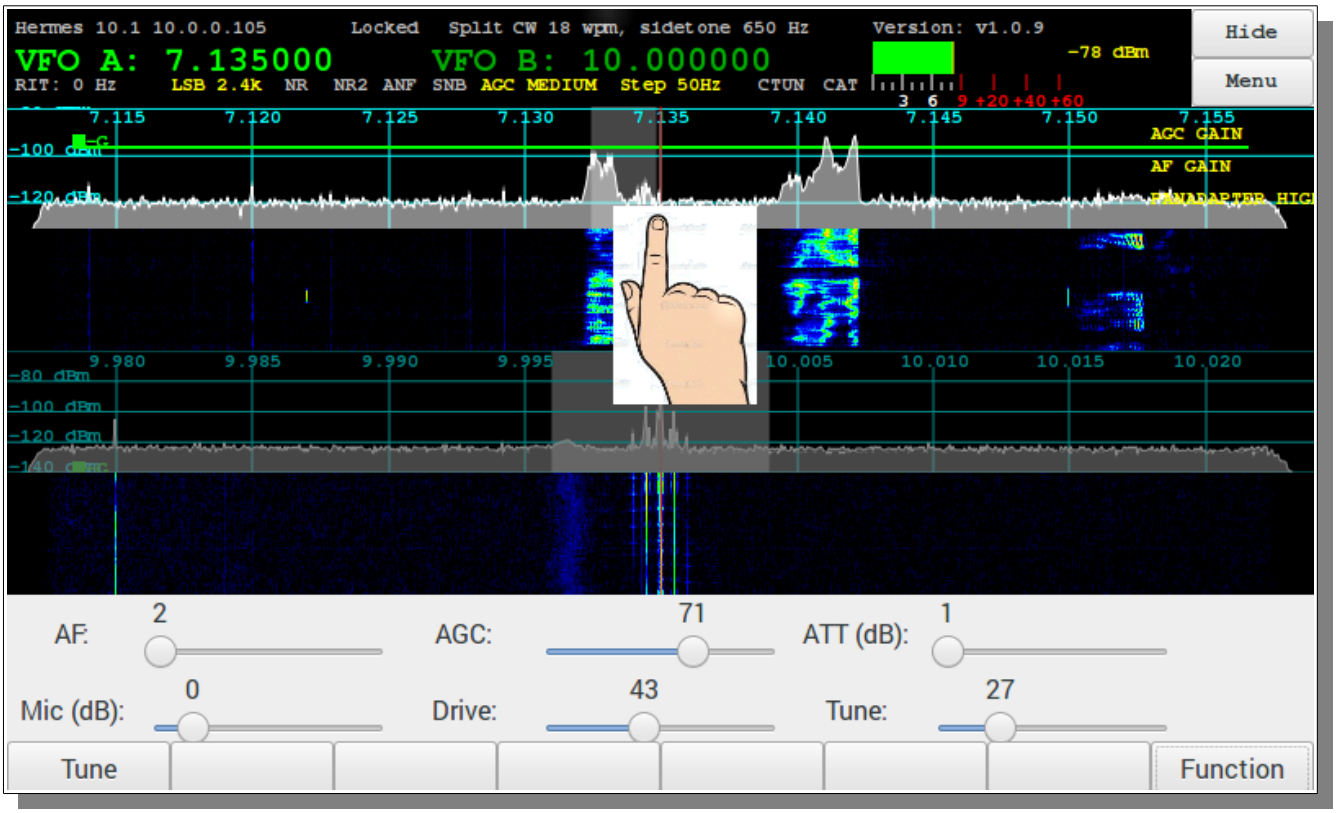
Menu → VOX



- Highlighted box indicates **Microphone Level** as you speak to adjust Threshold and Hang
- **Threshold** for VOX activation
- **Hang** for how long to hold VOX between words while speaking

9. On-Screen Controls and toolbar Buttons

Touch selection of **VFO-A** (Rx0)



Touch selection of **VFO-B** (Rx1)

-Cla 

ess

-Cla 

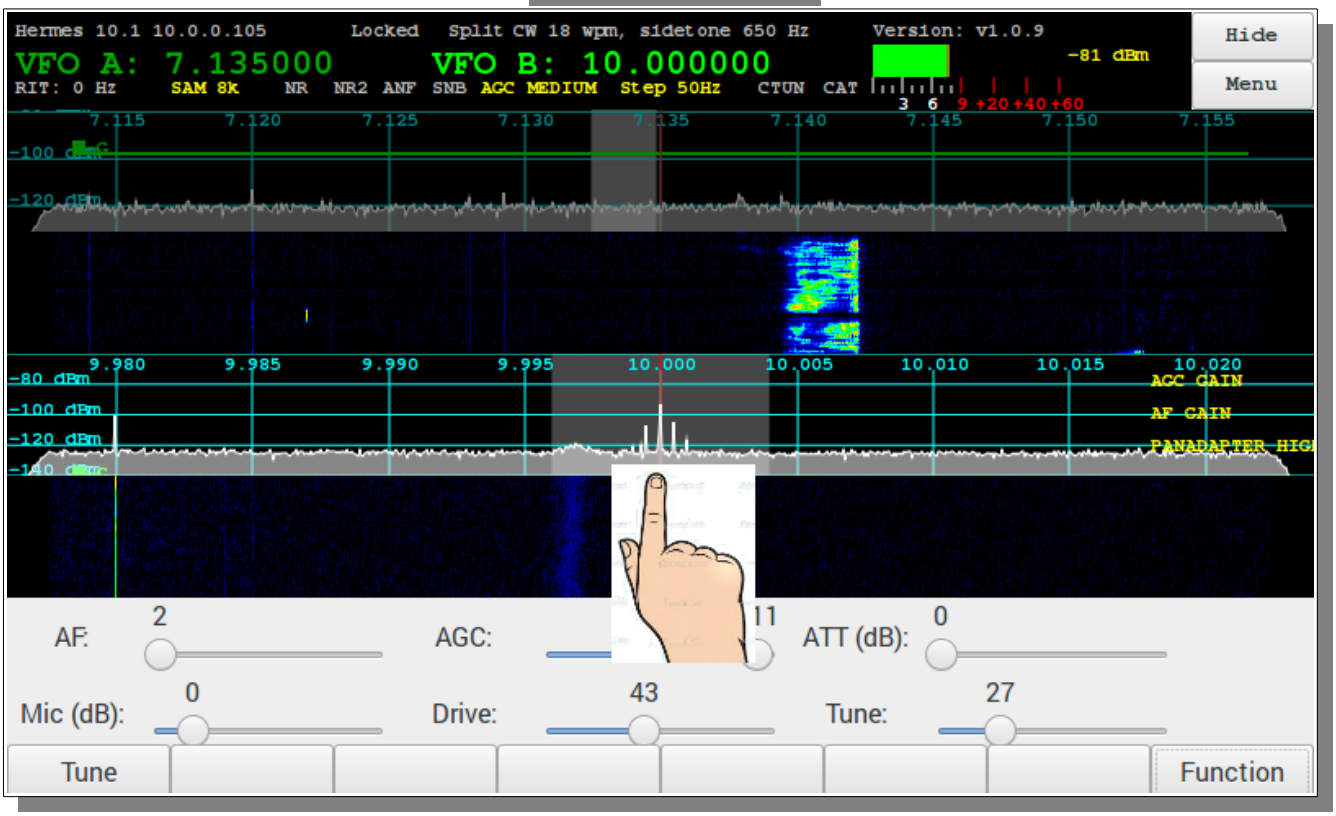
ess

-Cla 

ess

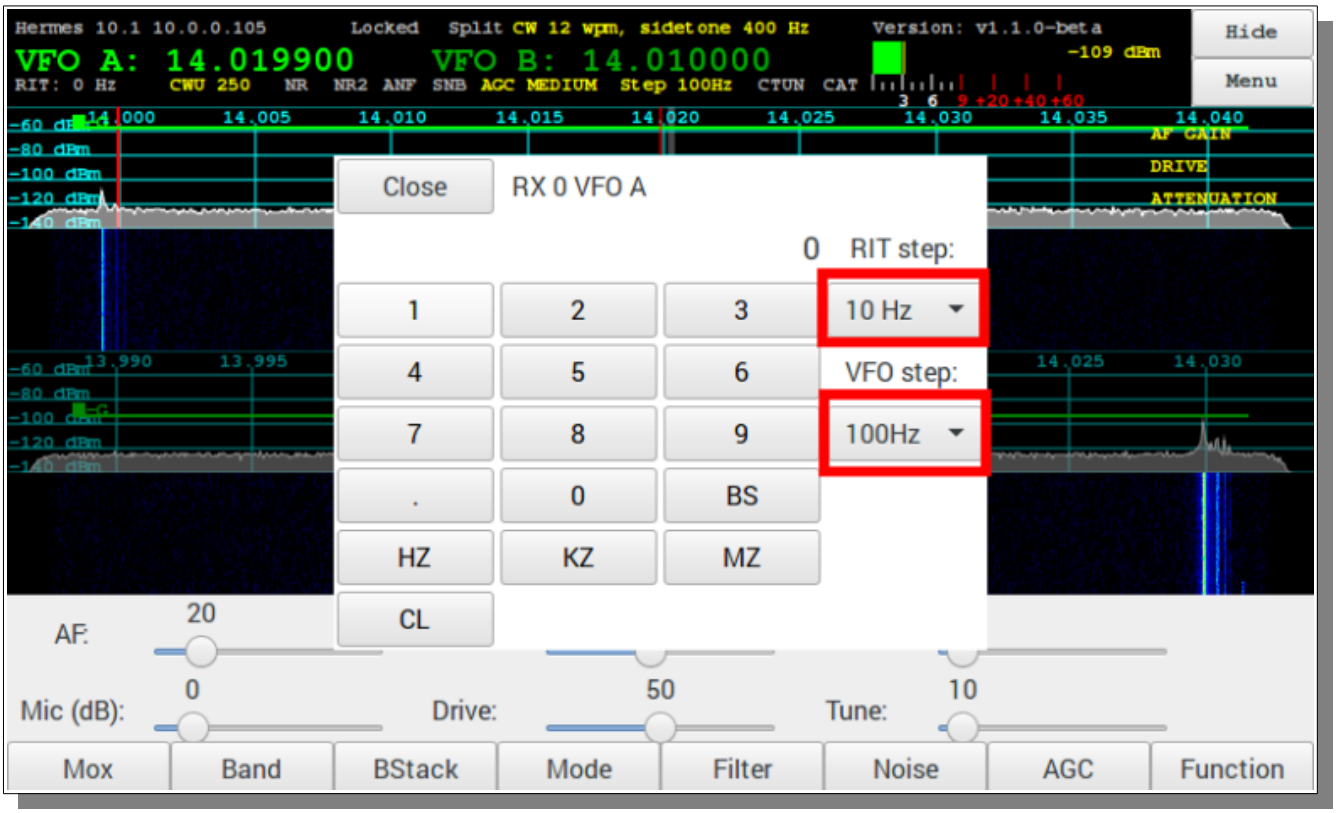
-Cla 

ess



Note: dragging the panadapter image with finger or mouse changes VFO frequency

Touch Direct Frequency Entry



- Touch or Mouse on VFO-A or VFO-B digits to bring up Direct Frequency Entry menu (example touch 1-4-3-2-0-KZ or 14.320 mHz = 14.320Mhz 20M)
  Note: there is a separate direct Freq entry for each VFO

- **RIT Step** Receiver Incremental Tuning

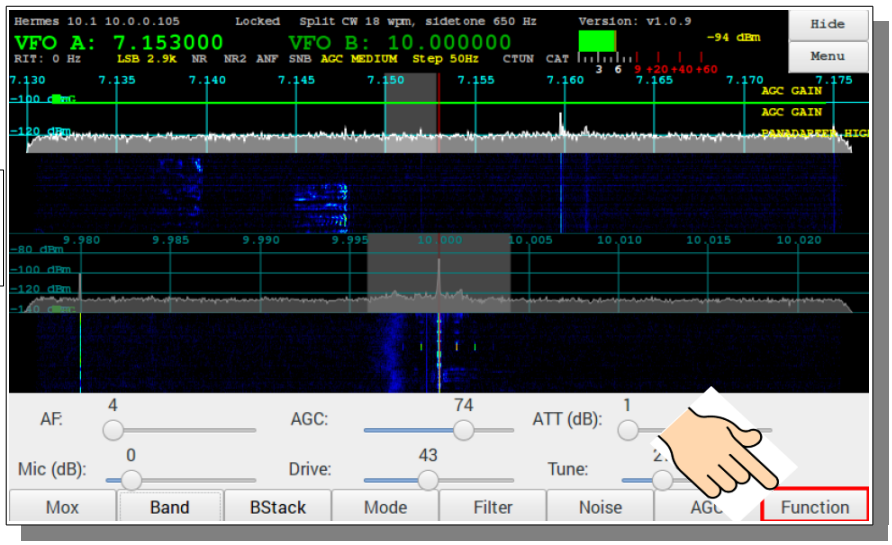
- **VFO Step** Frequency change per increment of VFO Encoder

Toolbar – Function button

There are eight (8) switches on the pihpsdr Controller. Switches S1 through S6 change function as you push the FN key

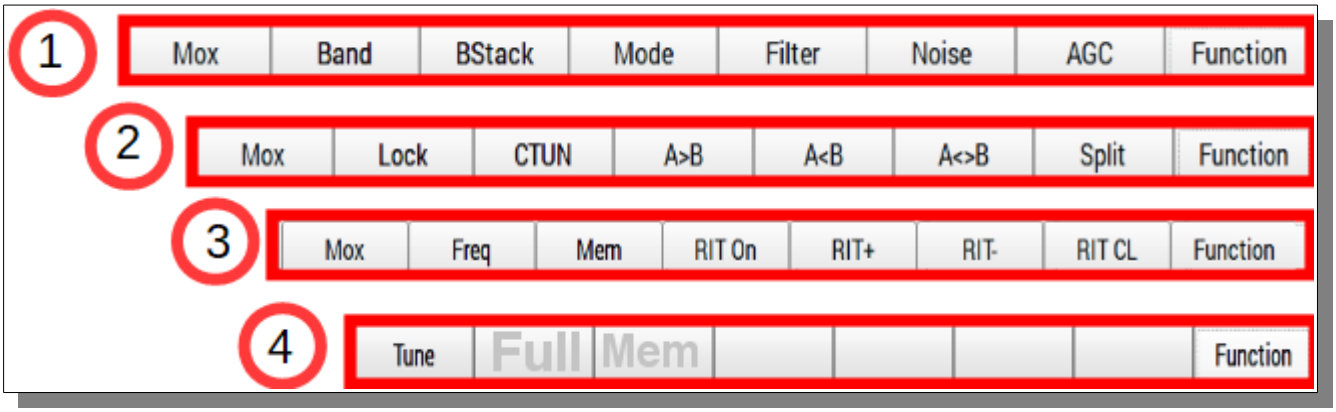
Physical Function "FN" Button

On-Screen Function Button

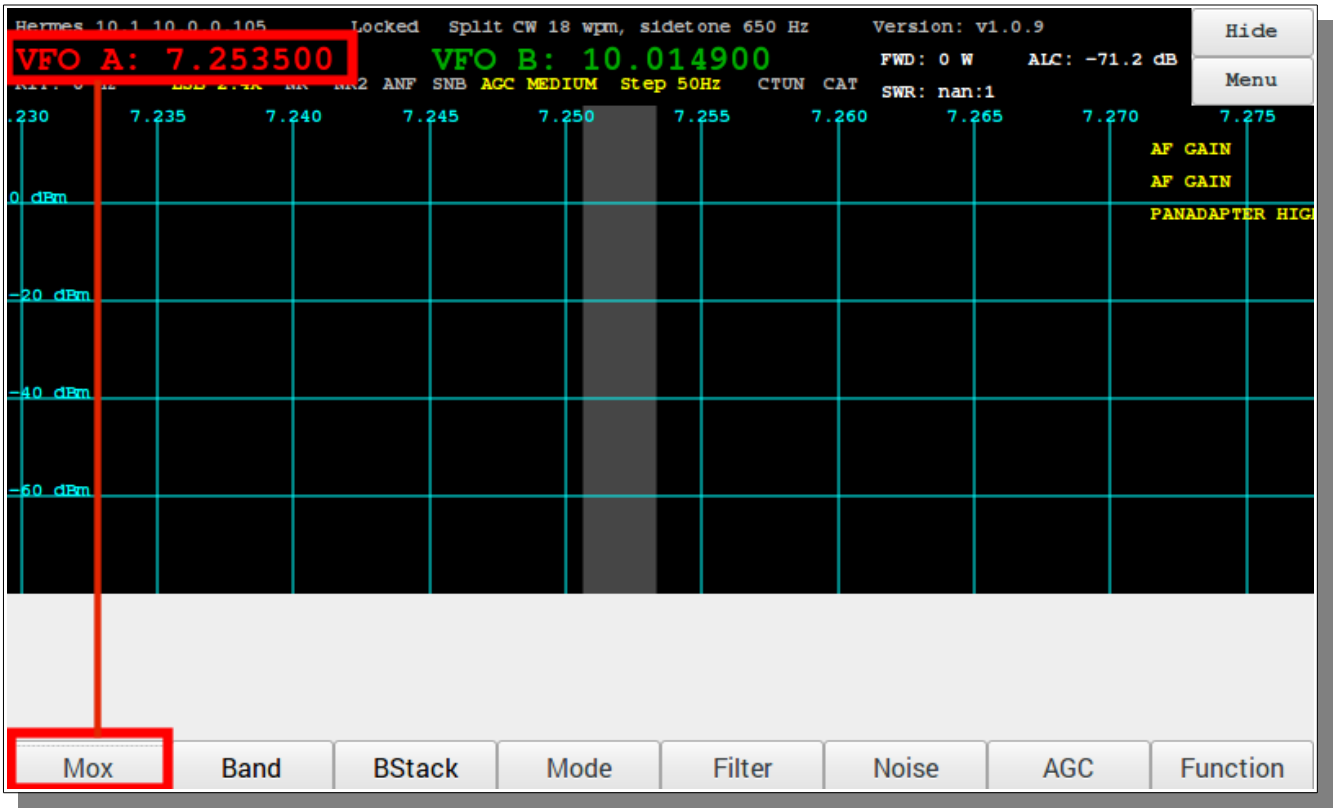


The function touch screen button or physical button enable the 4 functions buttons and encoders.

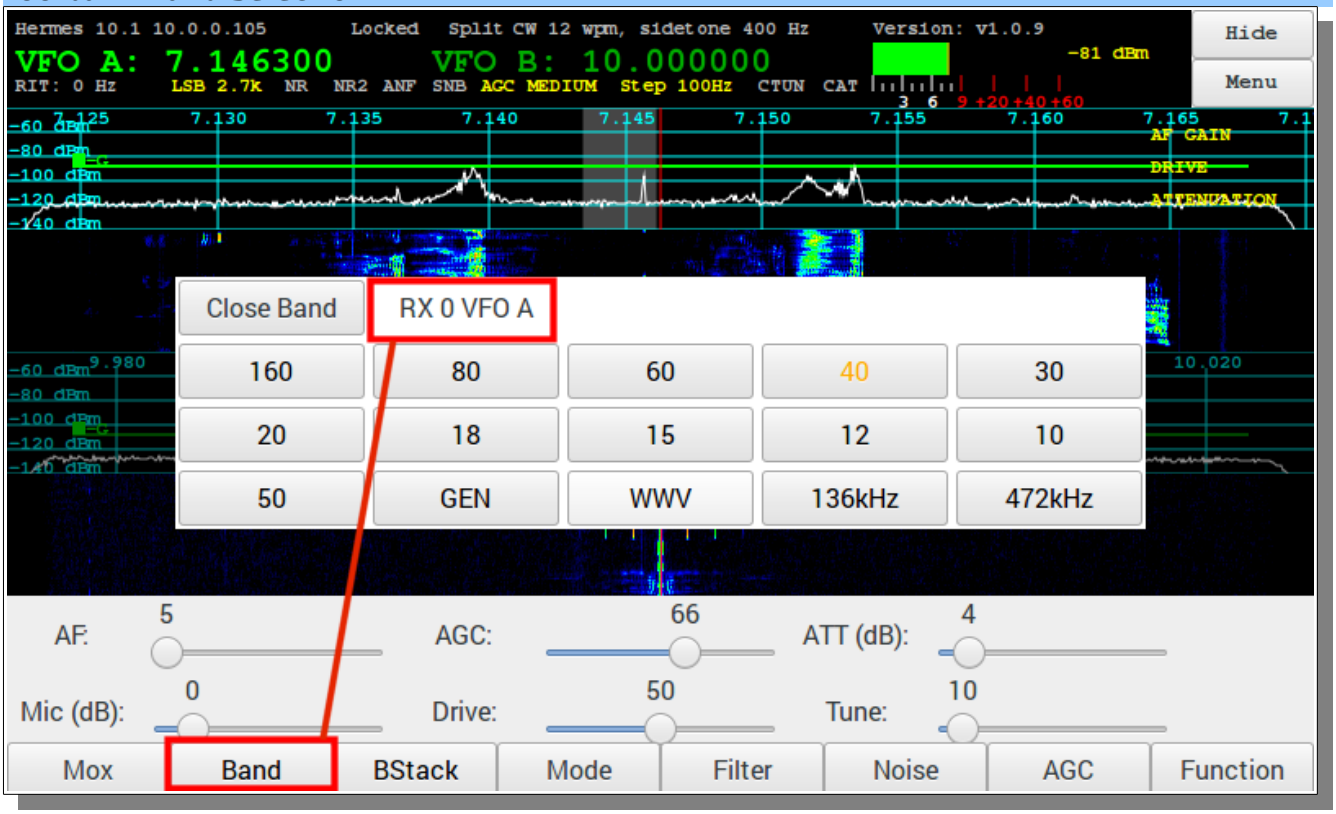
Four Toolbar menus – select with Function button or front panel switch



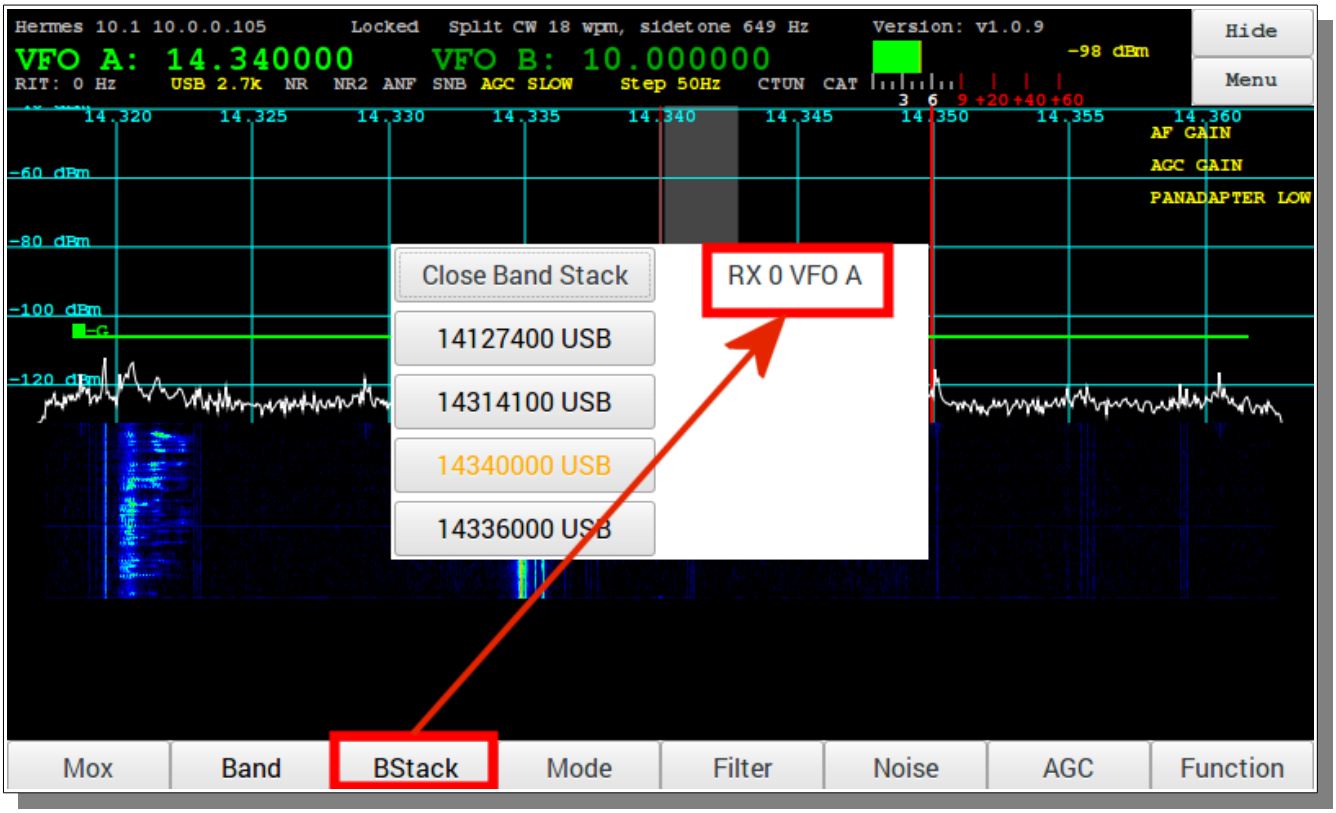
Toolbar – MOX

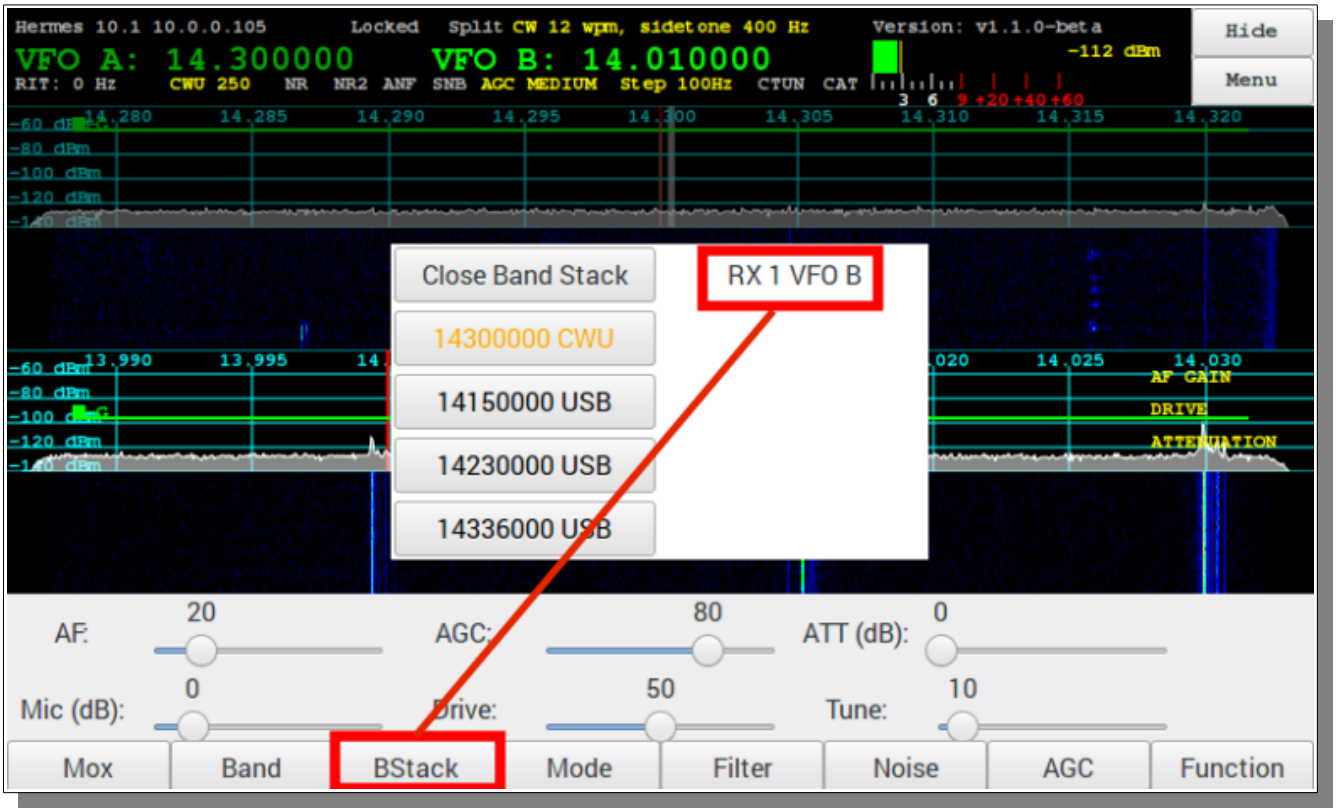


Toolbar – Band selection

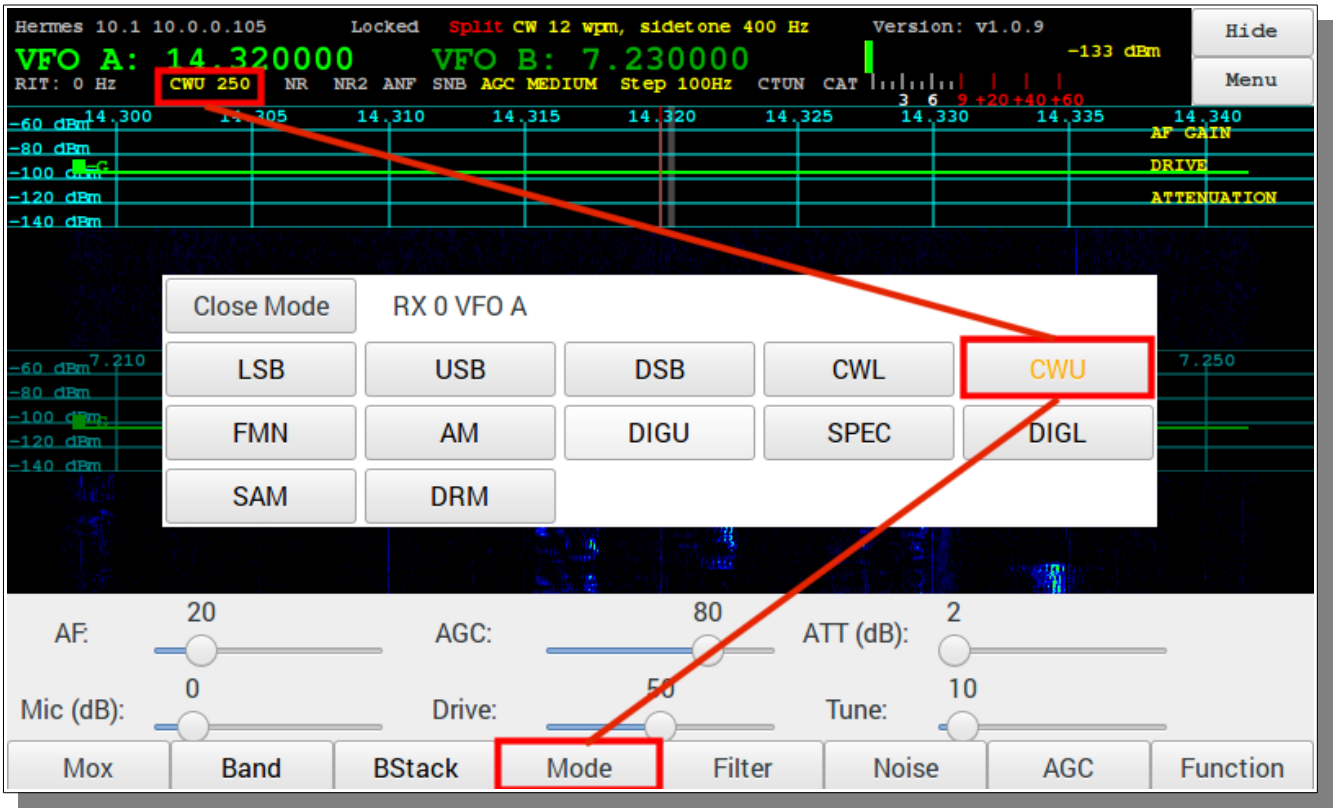


Toolbar – Bandstack for each VFO selected

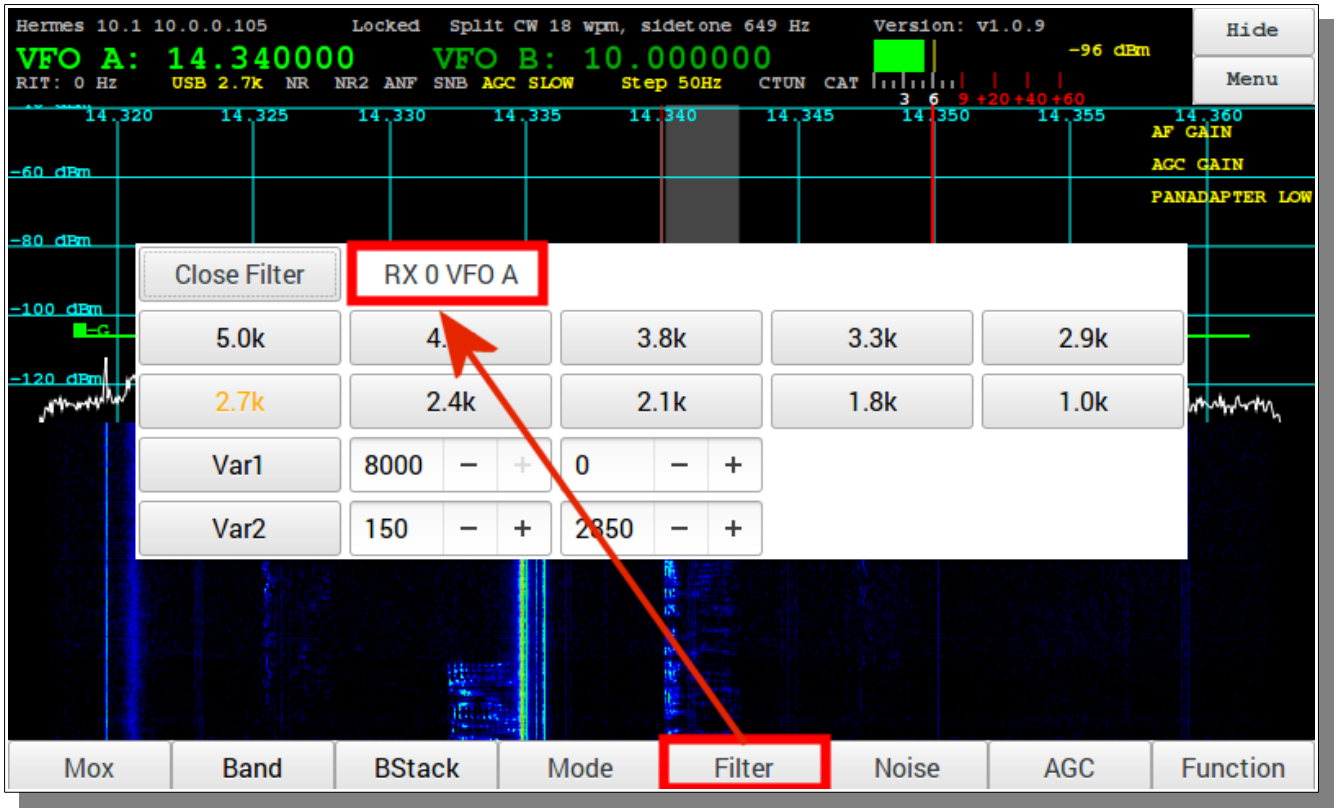




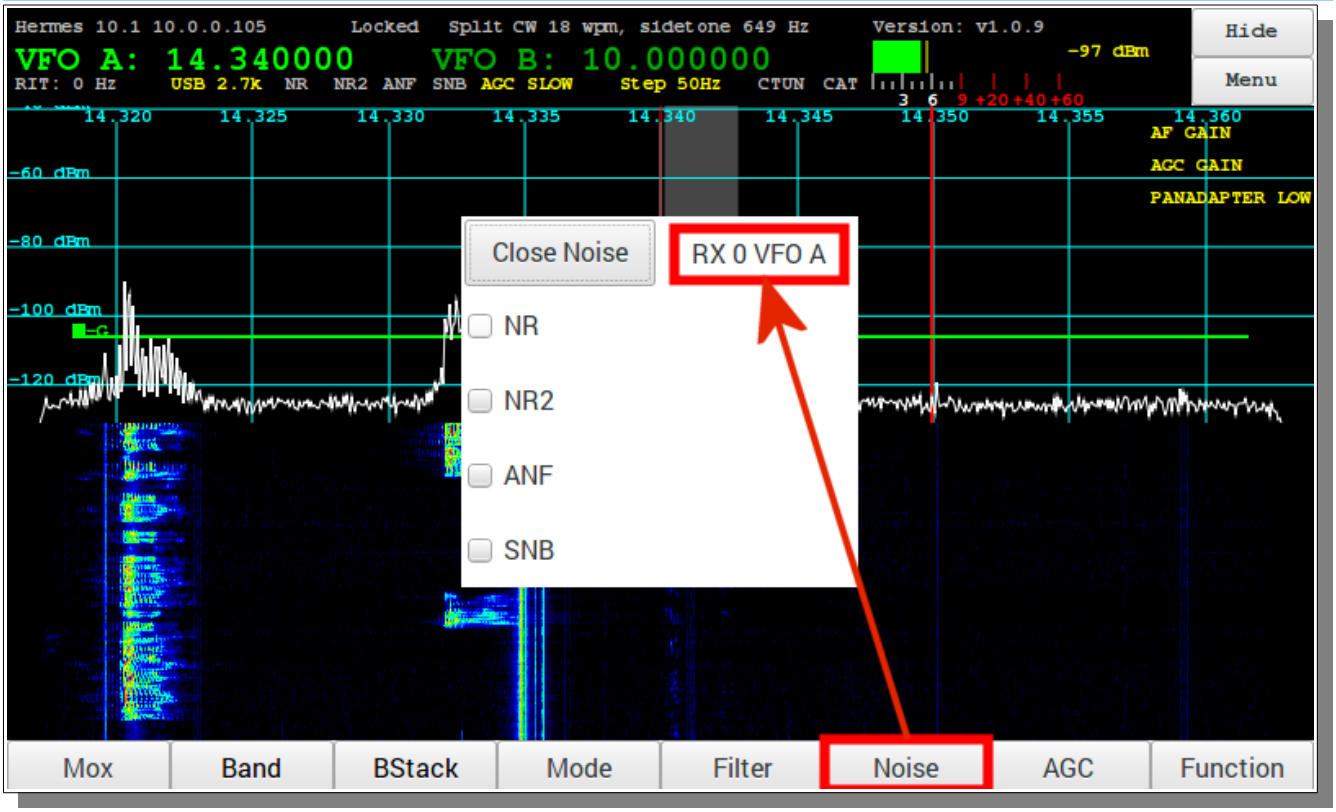
Toolbar - Mode



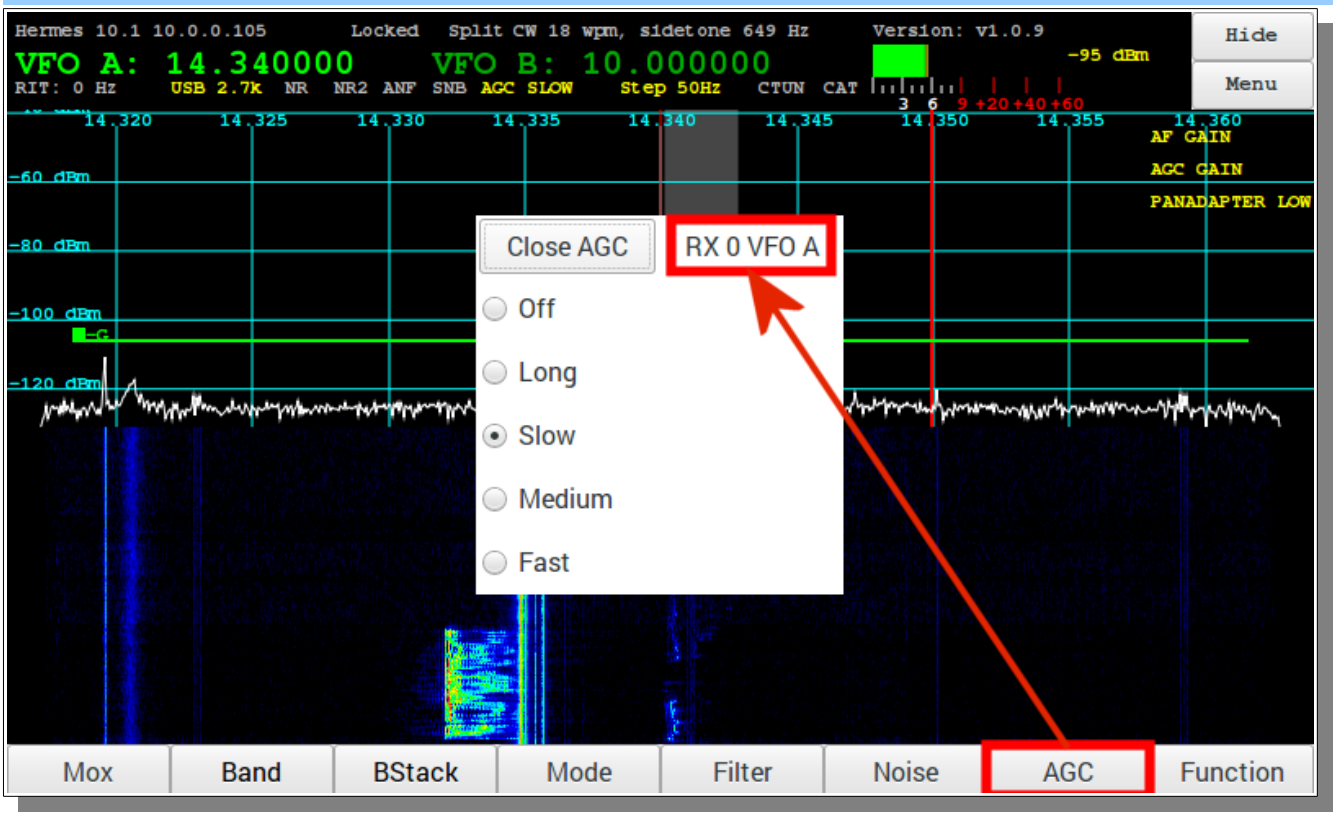
Toolbar - Filter



Toolbar – NOISE

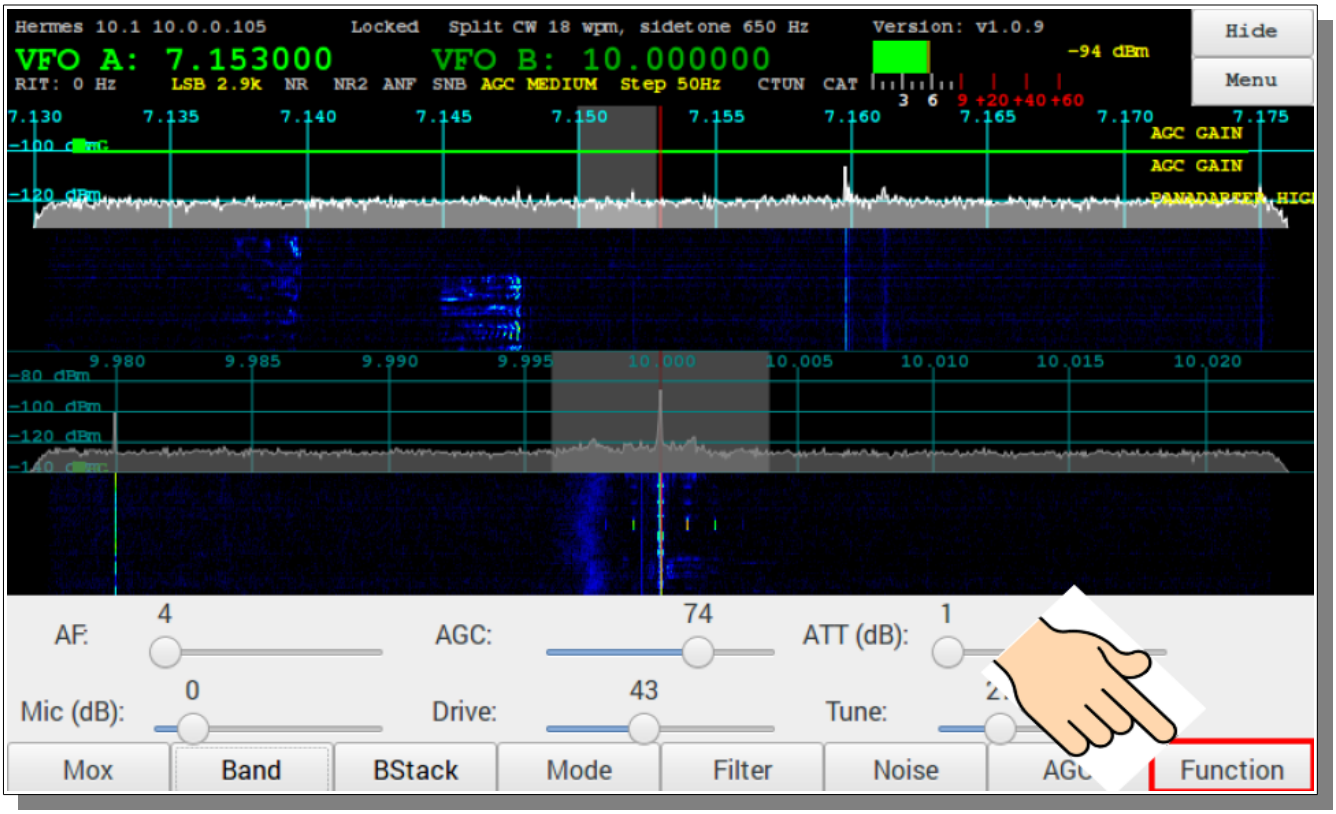


Toolbar – AGC



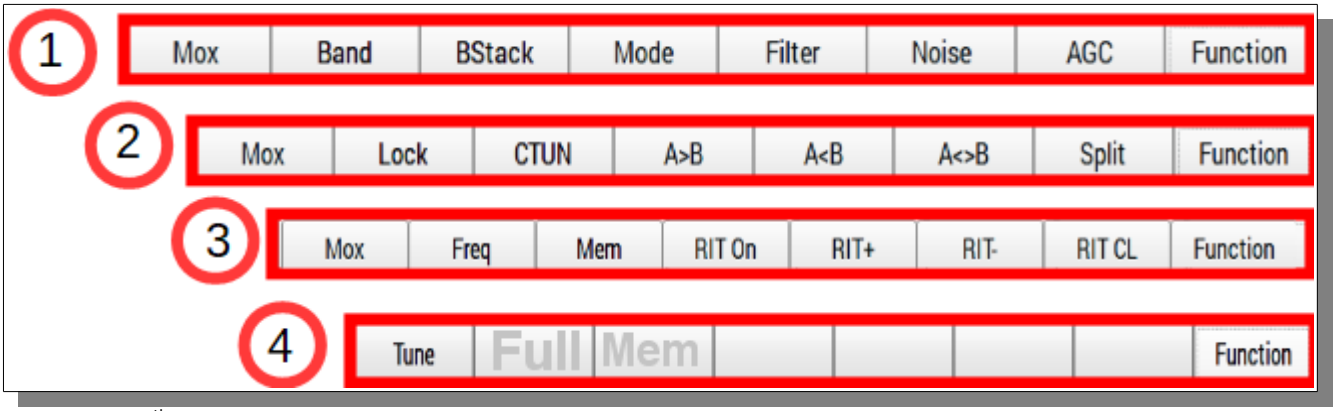
AGC default value is Medium

Toolbar – Function button



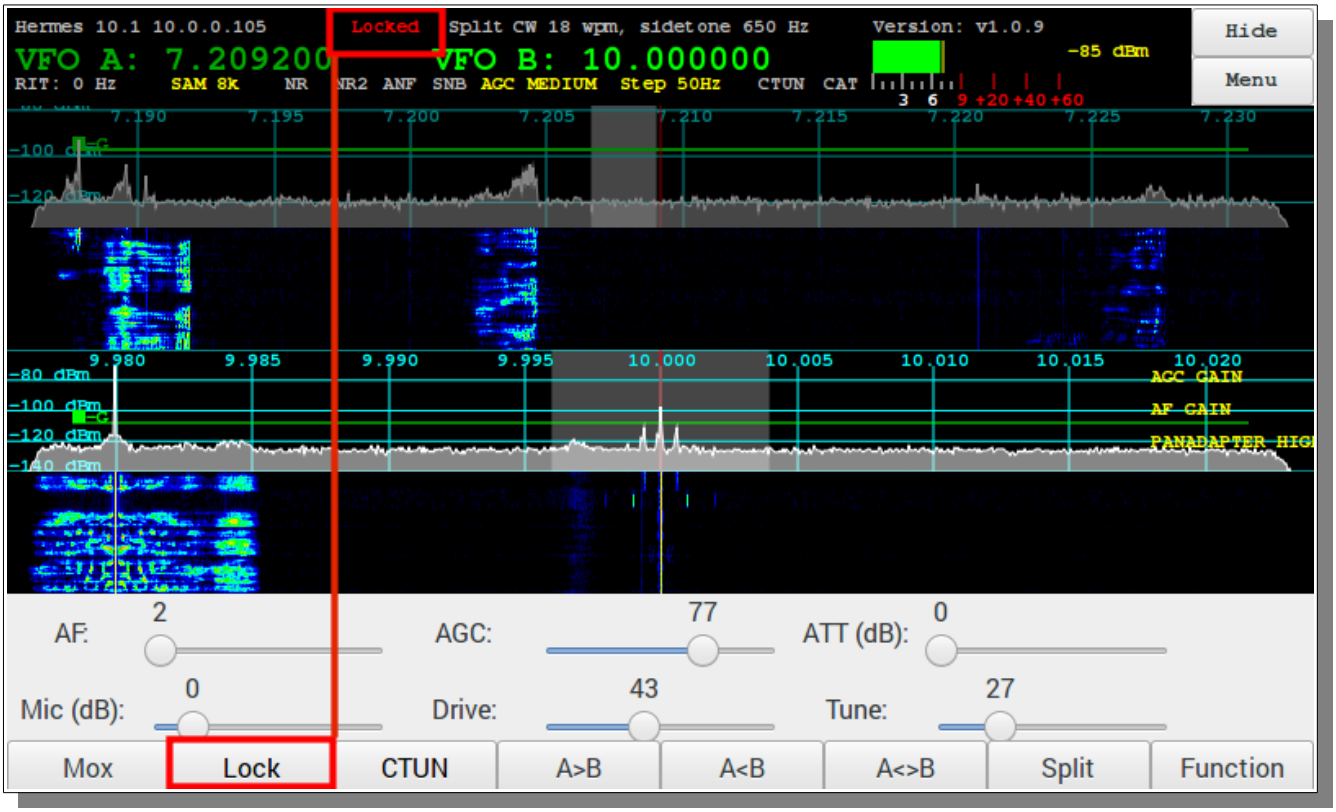
- The function touch screen button or physical button enable the optional functions of some of the buttons and encoders.

Four Function Toolbar menus – select with Function button or front panel switch



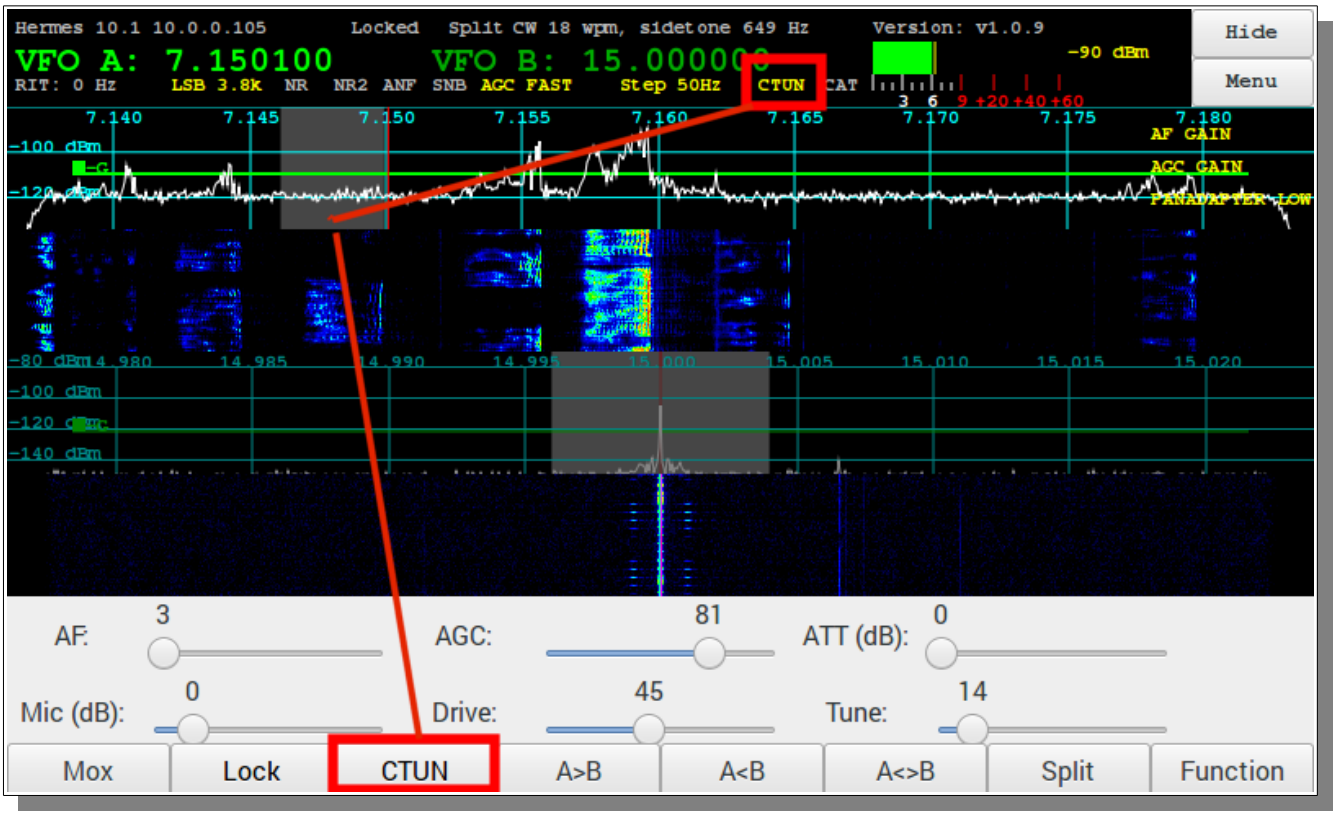
Note: the 4th Function Menu will include the Full Time and Mem Time from the Open Collector Menu in the future.

Toolbar VFO Lock



- The VFO can be locked by pressing the button on the AF Gain encoder or by tapping on the left side of the VFO display. To unlock press the AF Gain encoder button again or tap on the left side of the VFO display.
- When the VFO is locked, the red **Locked** text will be displayed near the VFO.

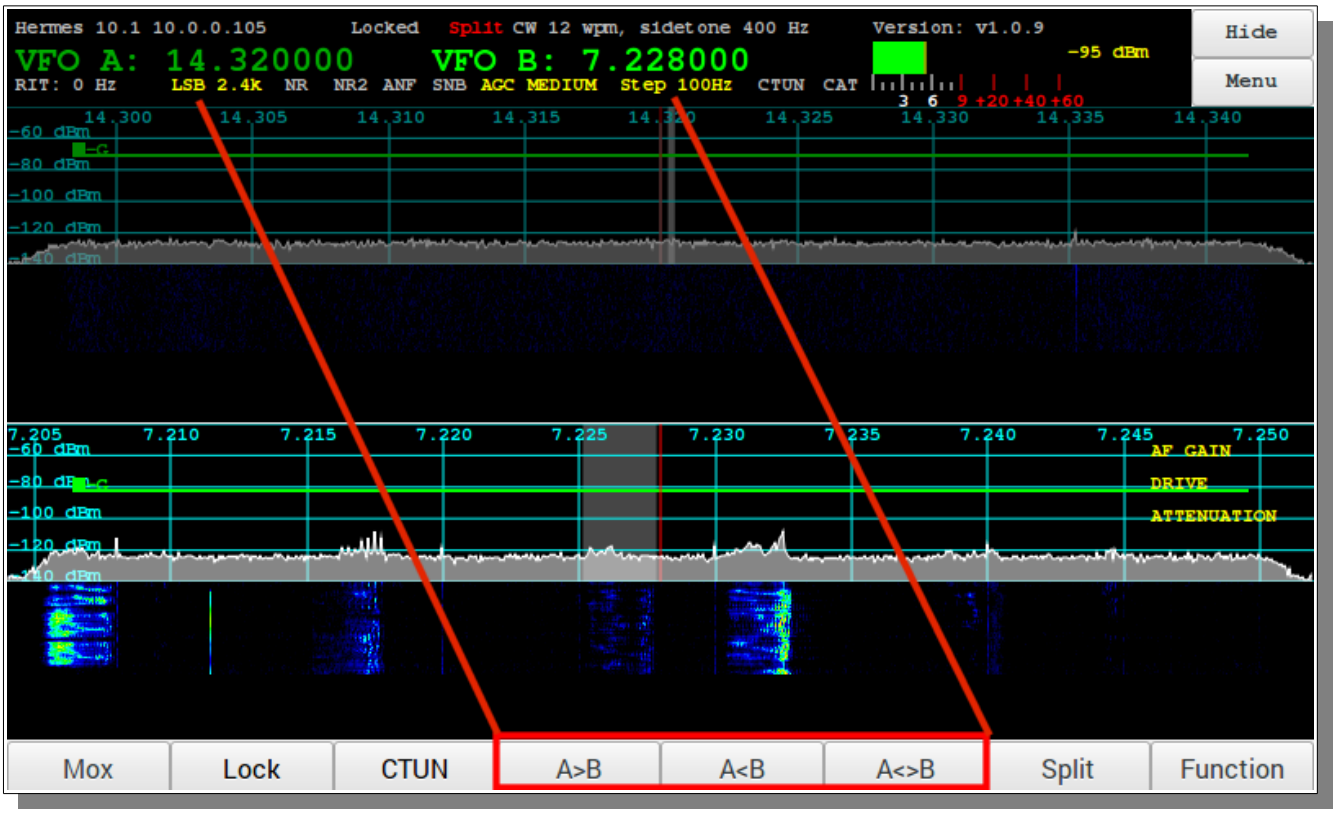
Toolbar CTUN – Click Tuning



Note: how filter is Tuned to a new frequency and CTUNE is announced in the status bar in Yellow

- Tapping on the CTUN button will enable or disable the click tuning function. When the function is enabled the CTUN button text will be shown in yellow.
- When CTUN is enabled, tuning is restricted to the passband currently displayed. The tuned frequency and filter moves withing the current passband display without moving the panadapter or waterfall left or right.

Toolbar A>B A<B A<>B



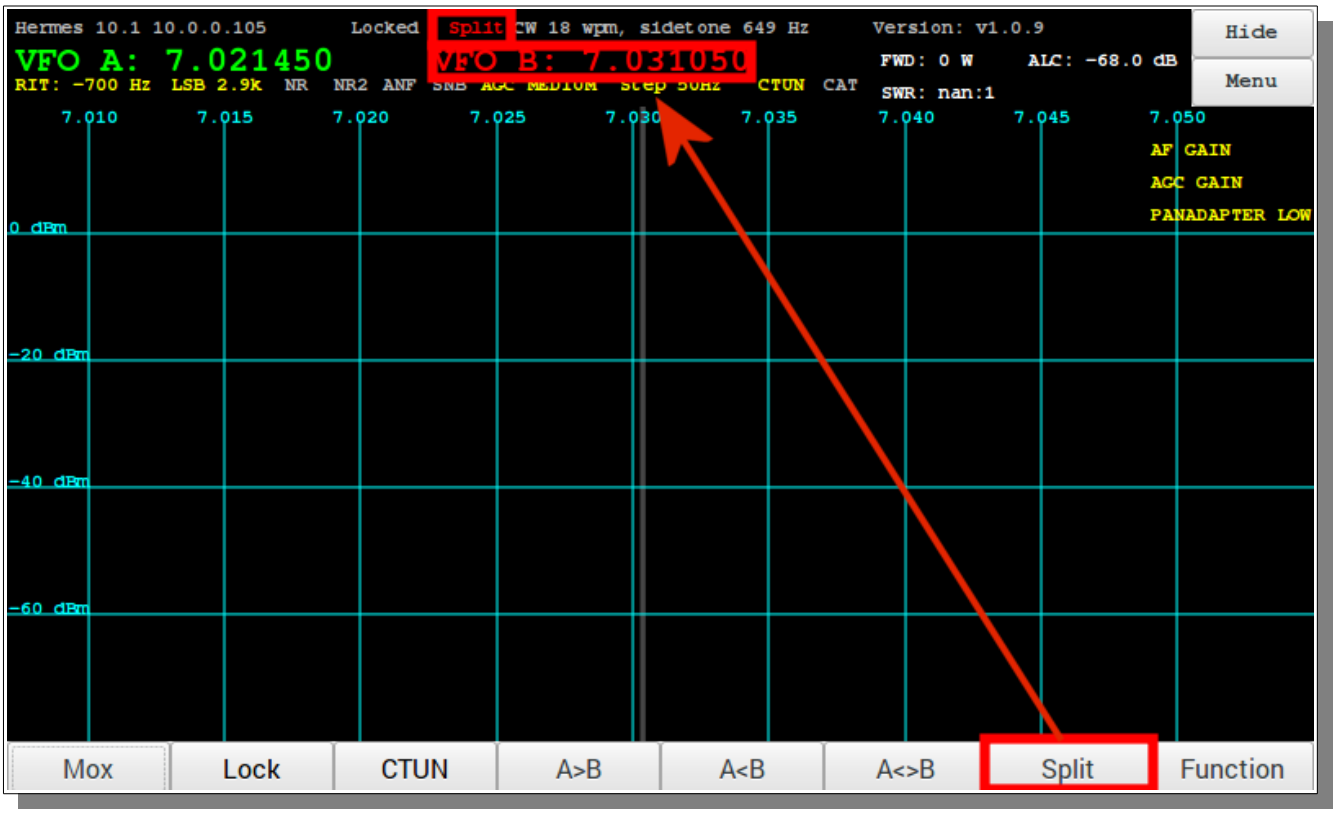
The toolbar allows you to copy the VFO frequency:

VFO-A into VFO-B (A>B)

VFO-B into VFO-A (A<B)

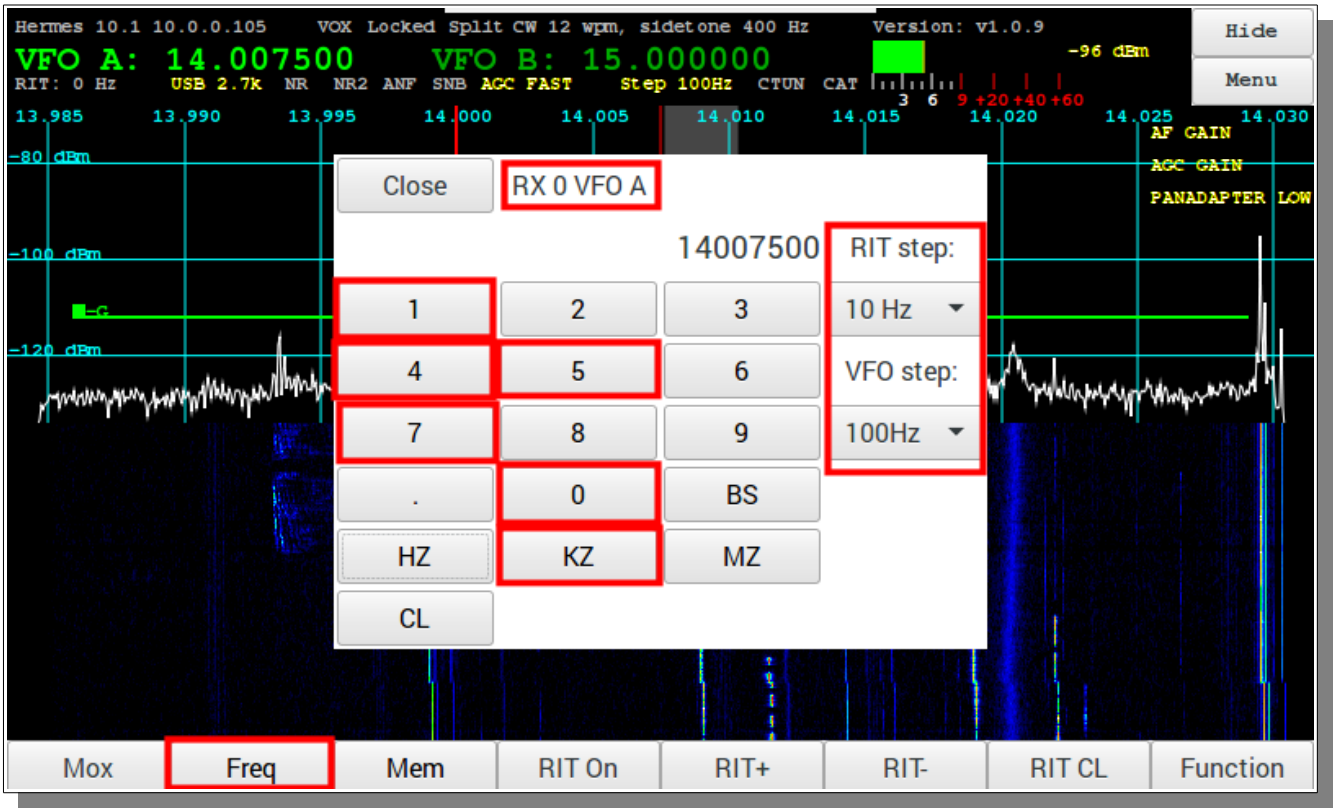
VFO-A swap with VFO-B (A<>B)

Toolbar SPLIT



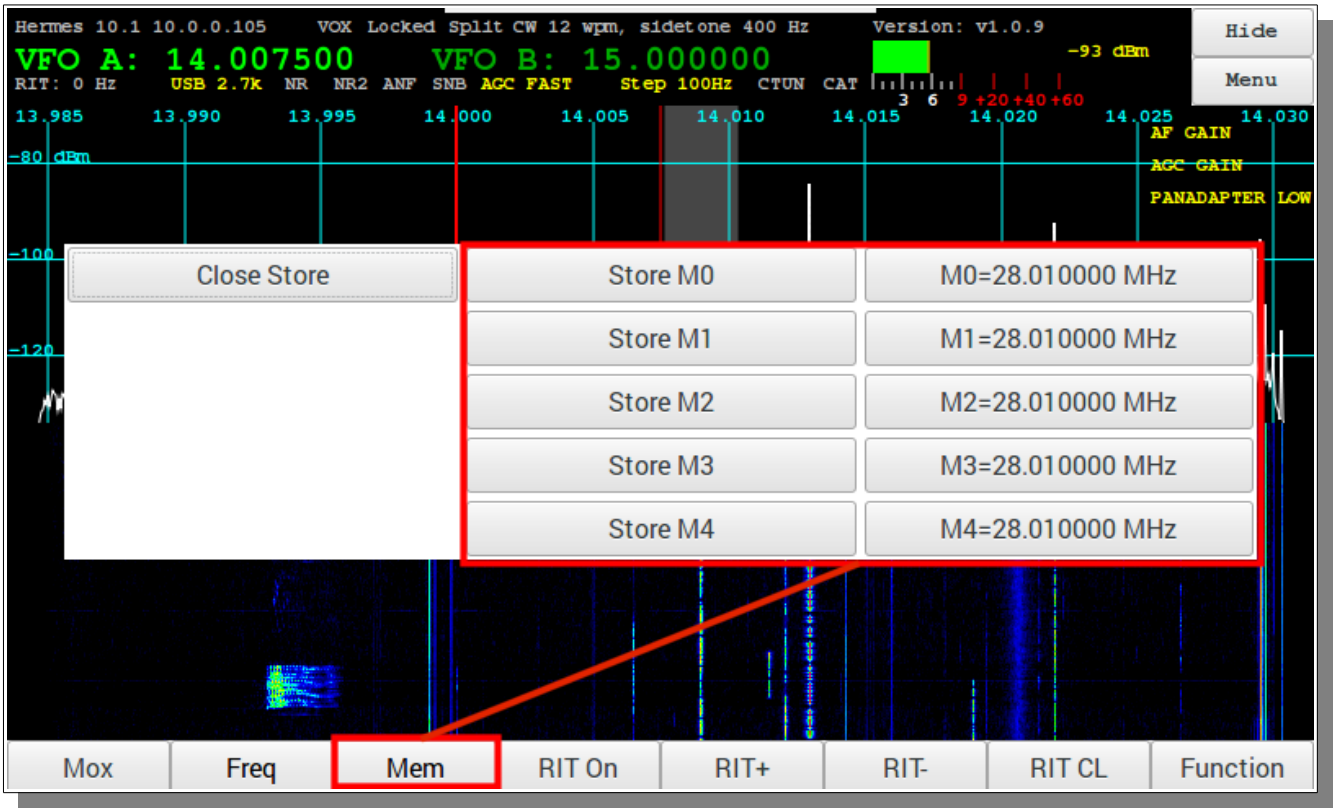
- The Split function allows you to select VFO-A for Receive and VFO-B for Transmit. The illustration shows transmission on VFO-B. This is a common practice when working Contests, DX, or Crossband.

Toolbar FREQ



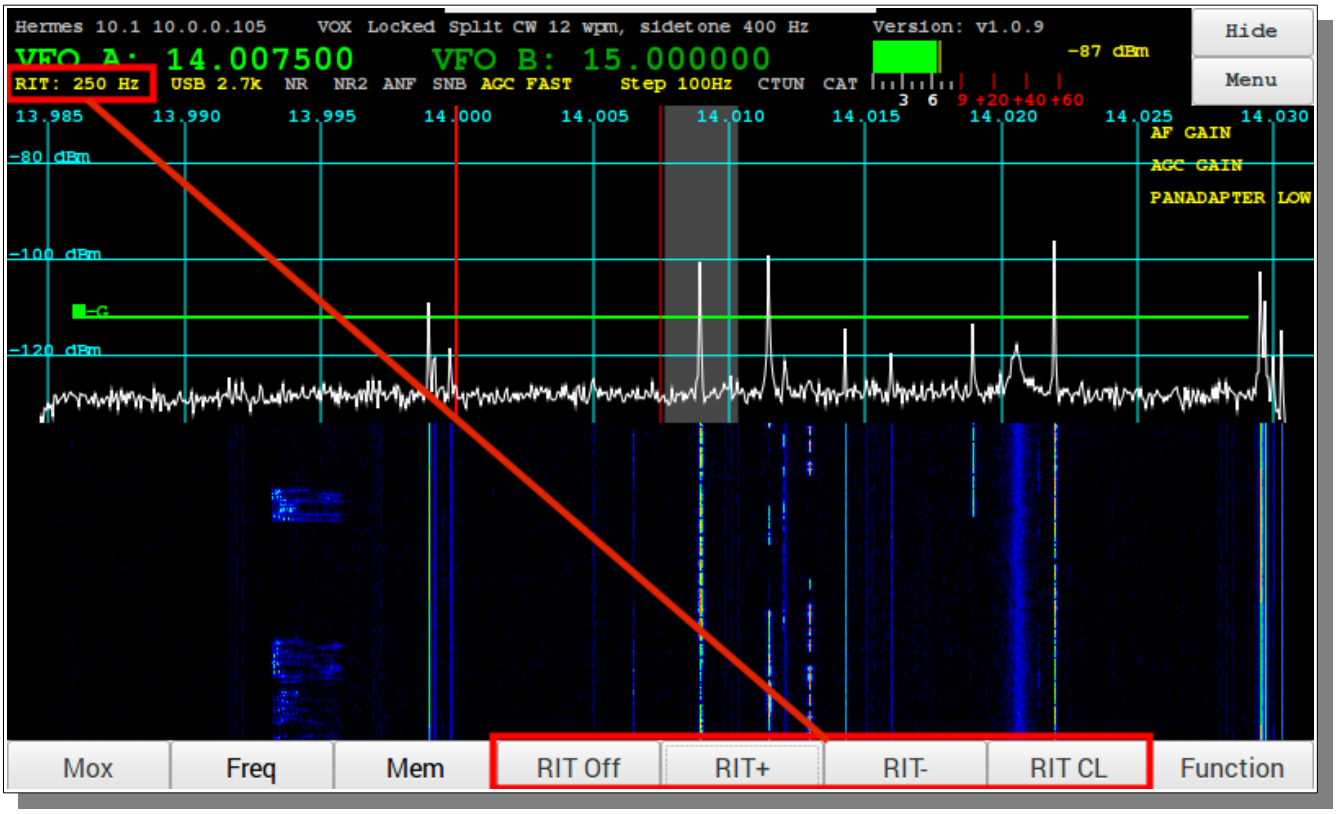
- **Direct frequency entry** using the touchpad is easily done by first selecting the VFO you wish to change, and then punching in the frequency in kHz or mHz. This illustration shows selection of 7.228 LSB on 40M..
- **RIT step** the increment in Hz for the RIT+ and RIT- toolbar entries (shown on page 48).
- **VFO step** the increment in Hz for the VFO Knob (E4) or Mouse Wheel.

Toolbar MEM



- Convenient storage and recall of five favorite frequencies

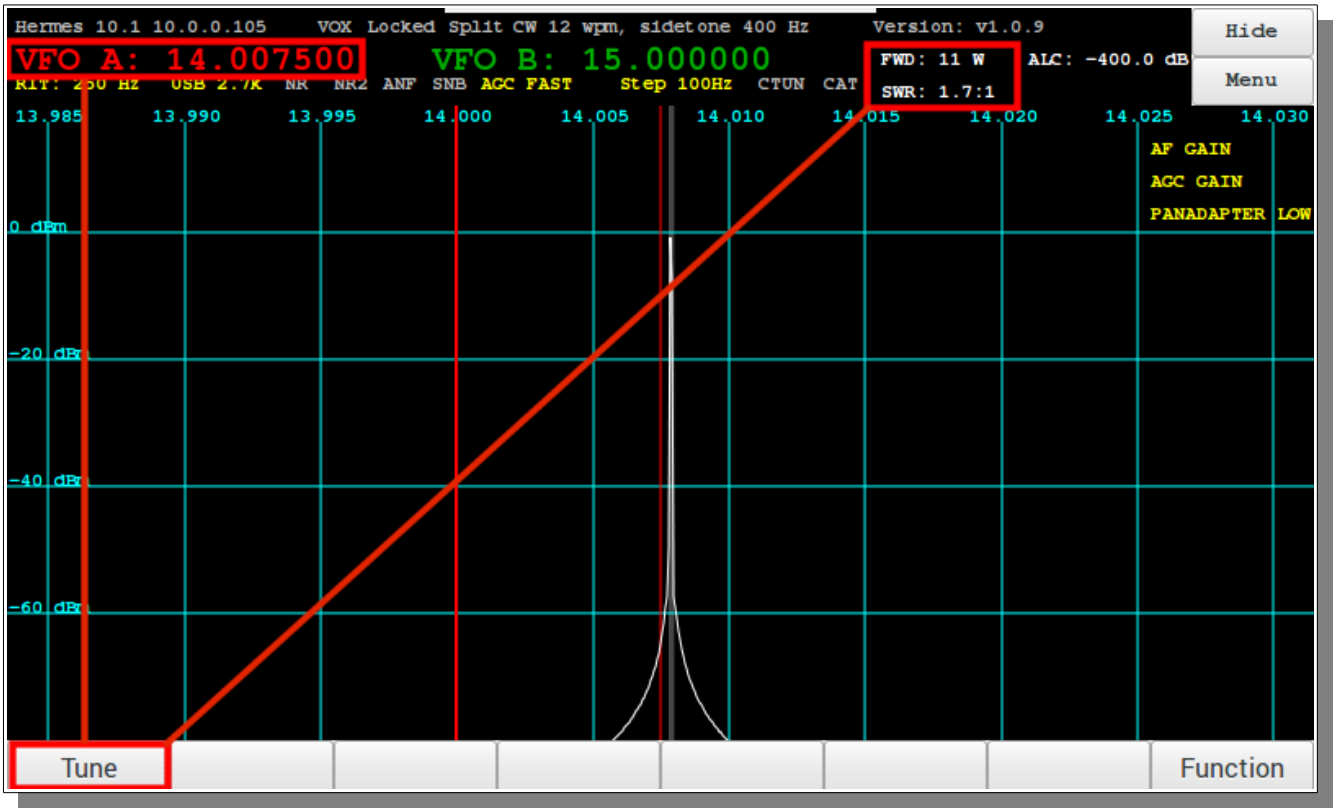
Toolbar RIT
RIT – Receiver Incremental Tuning



- Tapping the RIT OFF/ON enables or disables Receiver Incremental Tuning
- Tapping on the RIT+ (plus) or RIT- (minus) button will
- Tapping the RIT OFF or ON toggles the RIT function
- Tapping RIT CL = RIT CLEAR

Note: RIT offset shown in YELLOW TEXT below VFO-A

Toolbar → TUNE



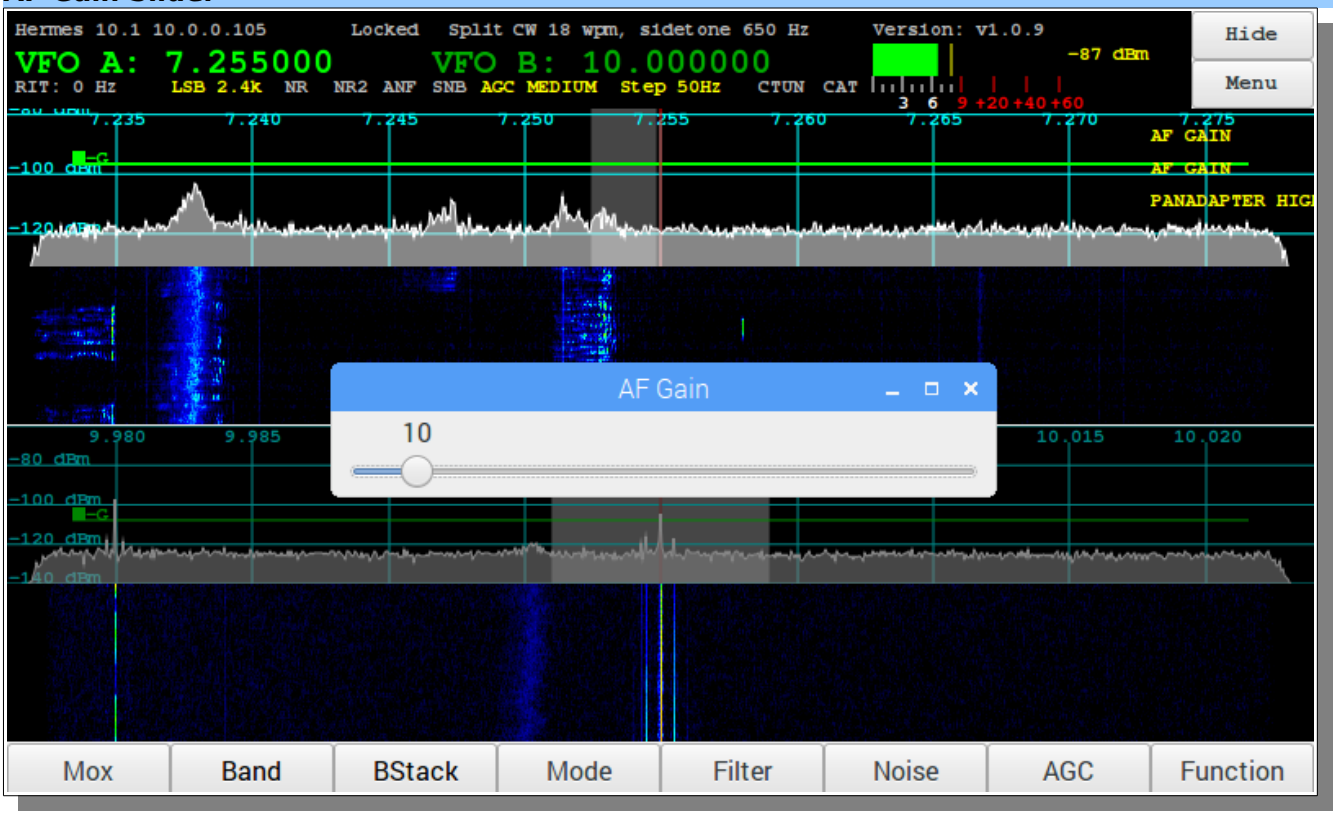
- The TUNE Toolbar gives quick access to keying the rig in CW and Generating TUNE watts.

Note: the FWD power and SWR measurements are indicated in the upper right
Shown here as 11W Forward Power and 1.7:1 SWR

This page reserved for future additions of

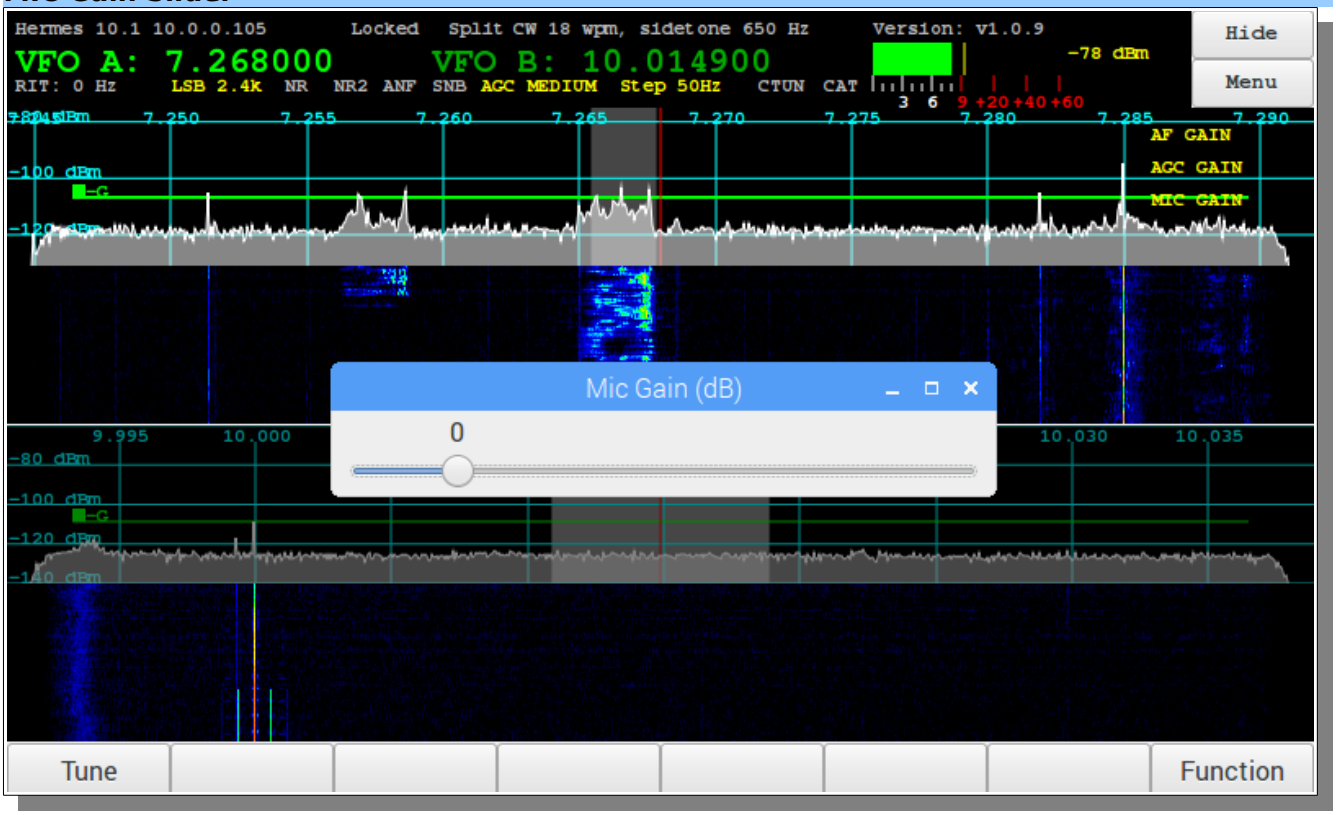
Open Collector - Full Tune and Memory Tune times (ms)

AF Gain Slider



AF Gain Slider is displayed using Function key to select one of the two uses for Encoder E1

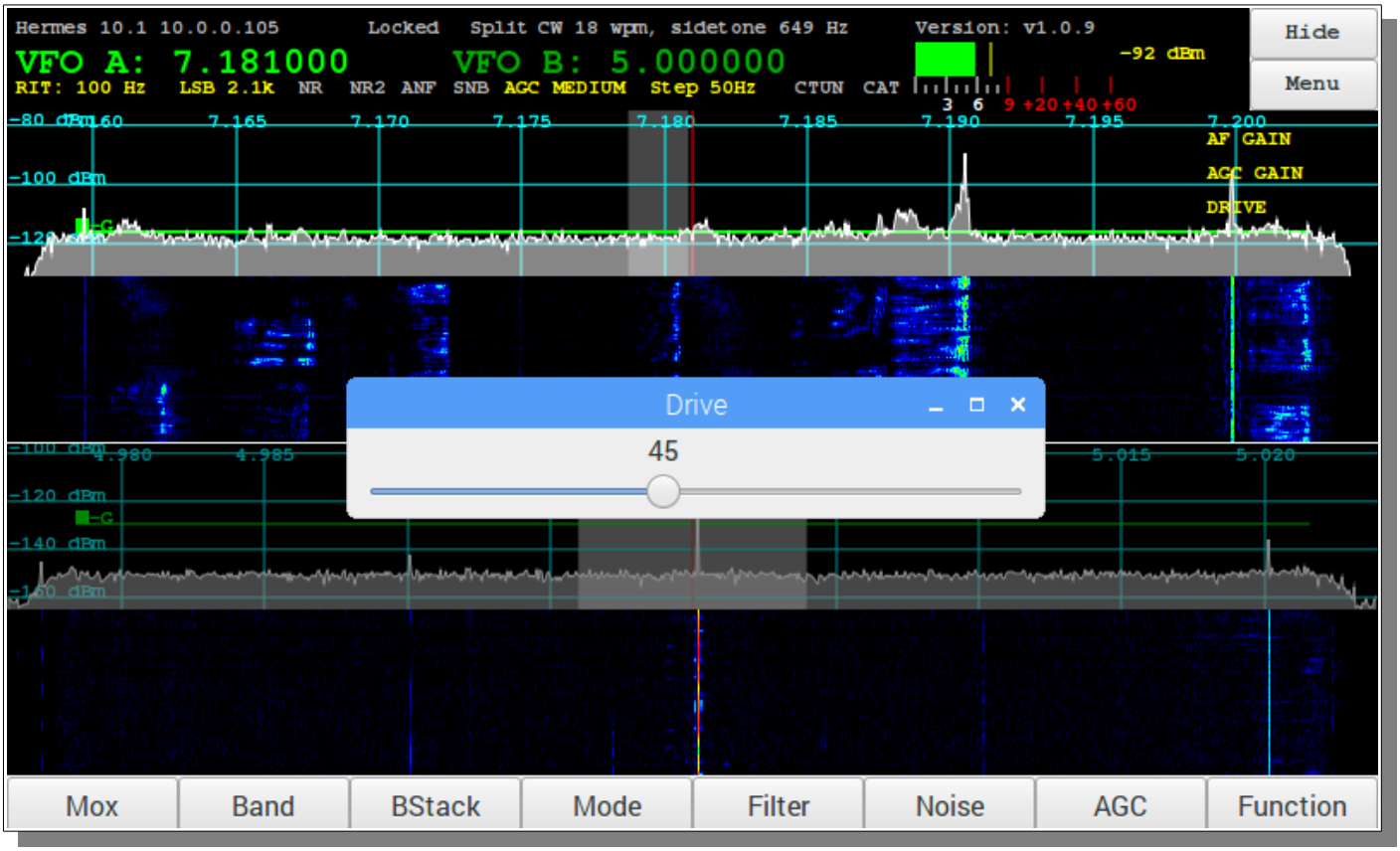
MIC Gain Slider



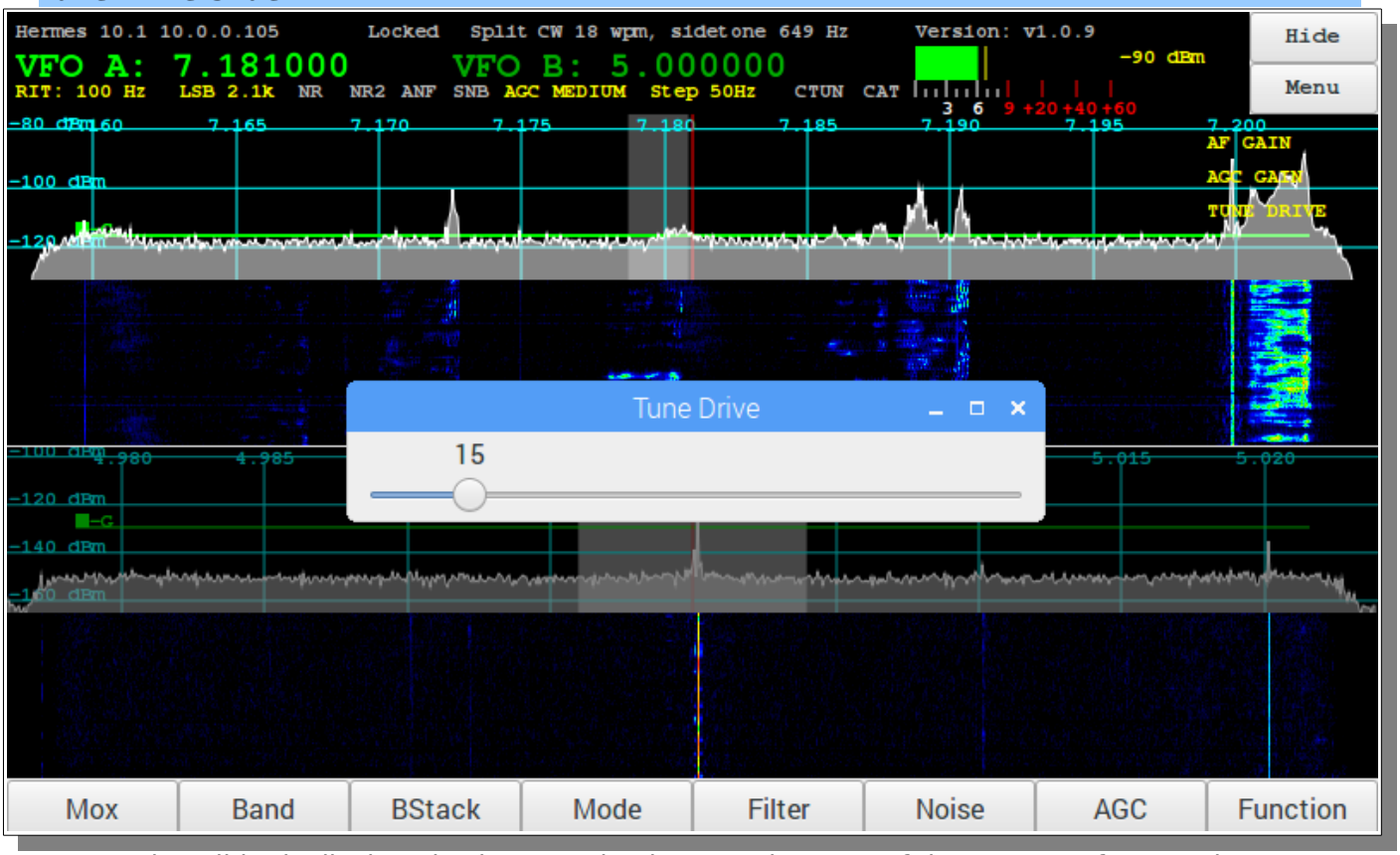
MIC Gain Slider is displayed using Function key to select one of the two uses for Encoder E1

PA Drive Slider

PA Drive Slider is displayed using Function key to select one of the two uses for Encoder E1

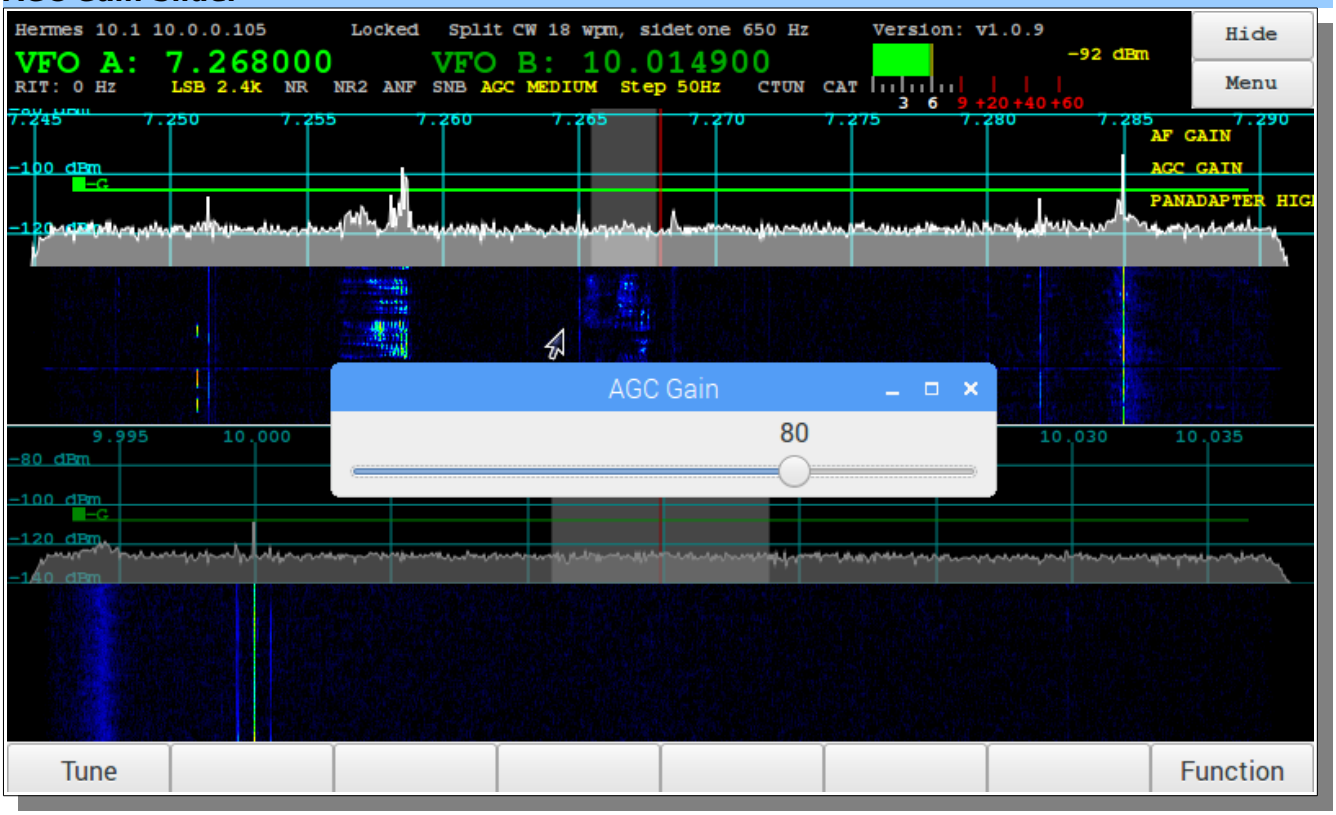


Tune Drive Slider



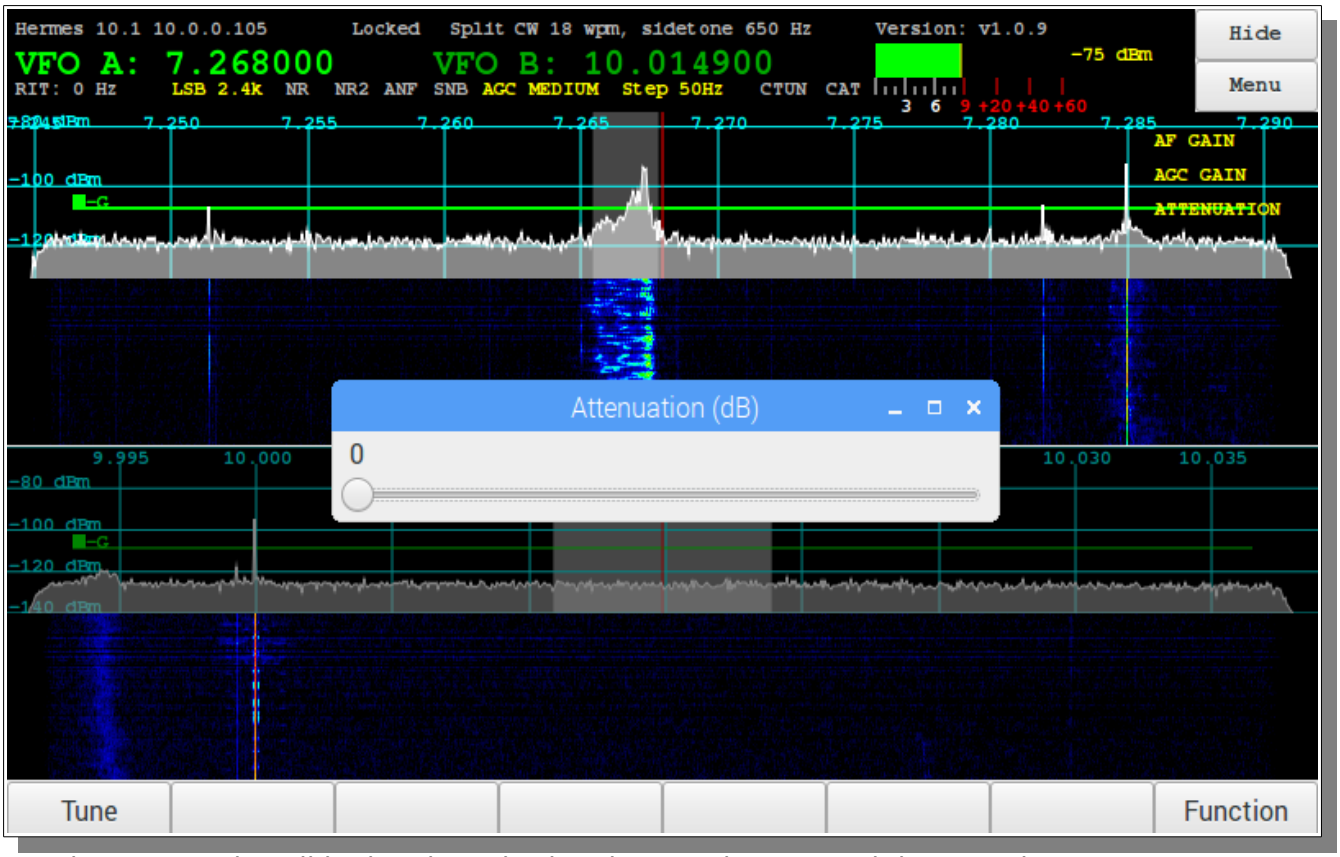
TUNE Drive Slider is displayed using Function key to select one of the two uses for Encoder E2

AGC Gain Slider



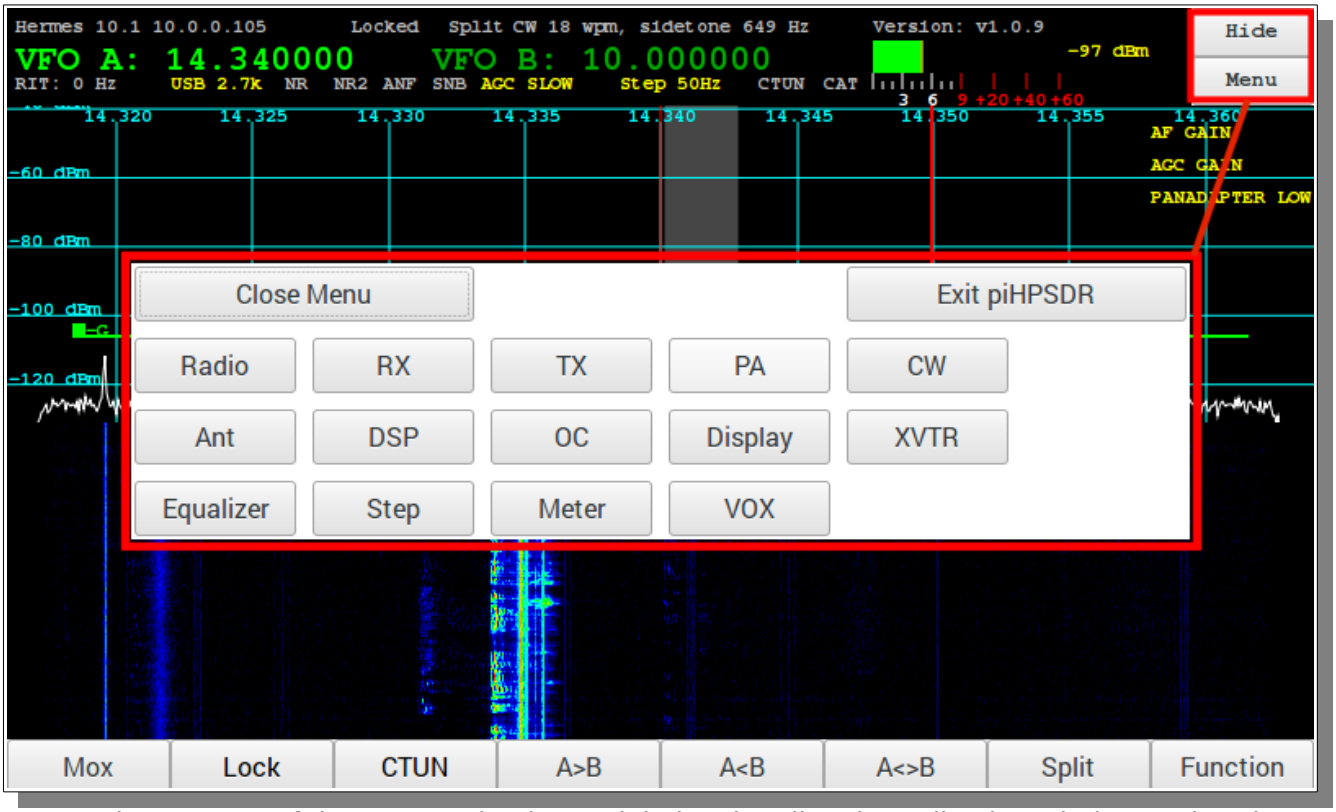
AGC Gain Slider is selected using the Function Key and then rotating E3

Receive Attenuation Slider



Receive Attenuation Slider is selected using the Function Key and then rotating E3

Main screen HIDE



- The purpose of the HIDE option is to minimize the pihpsdr application window and to then provide easy access to the RaspberryPi operating system and secondary applications such as Fldigi.

Menu or Toolbar → Meter

- Tapping on the Meter will display a list of options for updating the meter.

10. Tuning

VFO Encoder

The VFO encoder knob is used to tune the radio. By turning the encoder clockwise and anticlockwise the frequency will increment or decrement by the amount of the step value.

The General Menu has a field to set the resolution of the encoder.

Touch Screen

Touching and dragging on the panadapter or waterfall will move the frequency up or down. Note that it will move in step increments.

Tapping a frequency on the panadapter or waterfall will move to that frequency.

Mouse

Left down and holding then dragging while on the panadapter or waterfall will move the frequency up or down.

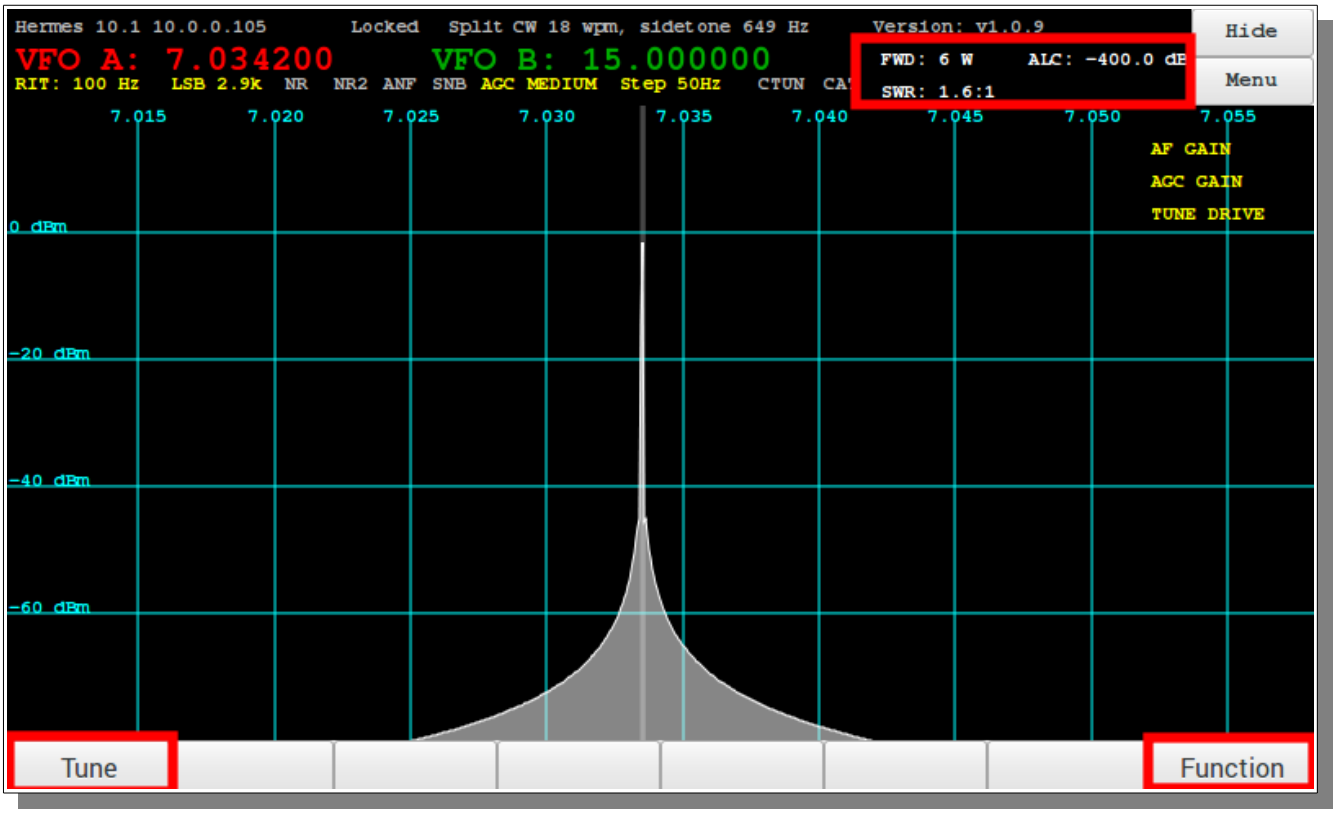
Left clicking will move to the selected frequency.

Moving the scroll wheel will increment or decrement the frequency by the step value.

11. TUNE/SWR/FWD Power

TUNE/SWR/FWD power

When the Function button is selected, it toggles the TUNE/MOX button on the far left. FWD power and SWR can then be read in the top of the pihpsdr window.



12. Appendix

Encoders and Switches

Factory Switch/Encoder/Touch activation

Switch or Encoder	Function	Touch Screen	GPIO A	GPIO B
Power ON/OFF	Controller power on/off	Controller power on/off	-----	-----
TUN – TUNE button	generates a carrier with Tune power selection slider	generates a carrier with Tune power selection slider	27	-----
MOX – MOX button	Space bar toggle for PTT or MOX button	Space bar toggle for PTT or MOX button	27	-----
S1 – Band	10 HF Bands + General Coverage + WWV + LF + XVRT	10 HF Bands + General Coverage + WWV + LF + XVRT	13	-----
S2 – BandStack	Four last used frequencies	Four last used frequencies	12	-----
S3 – Mode	LSB USB DSB CWL CWU FMN AM DRM FreeDV PSK	LSB USB DSB CWL CWU FMN AM DRM FreeDV PSK	6	-----
S4 – Filter	10 IF Filter widths	10 IF Filter widths	5	-----
S5 – Noise	5 Noise Reduction modes	5 Noise Reduction modes	24	-----
S6 – AGC	5 AGC Decay settings	5 AGC Decay settings	23	-----

Switch or Encoder	Function	Touch Screen	GPIO A	GPIO B
FN – Function	Function switch to toggle TUNE or MOX button displayed on the bottom left of the screen	Function switch to toggle TUNE or MOX and the action of E1, E2, E3 knobs	22	-----
E1 -- assignable	assignable	-------	20	26
E2 – assignable	assignable	-------	16	19
E3 – assignable	assignable	-------	4	21
E4 – VFO Main tuning knob	-------	-------	17	18

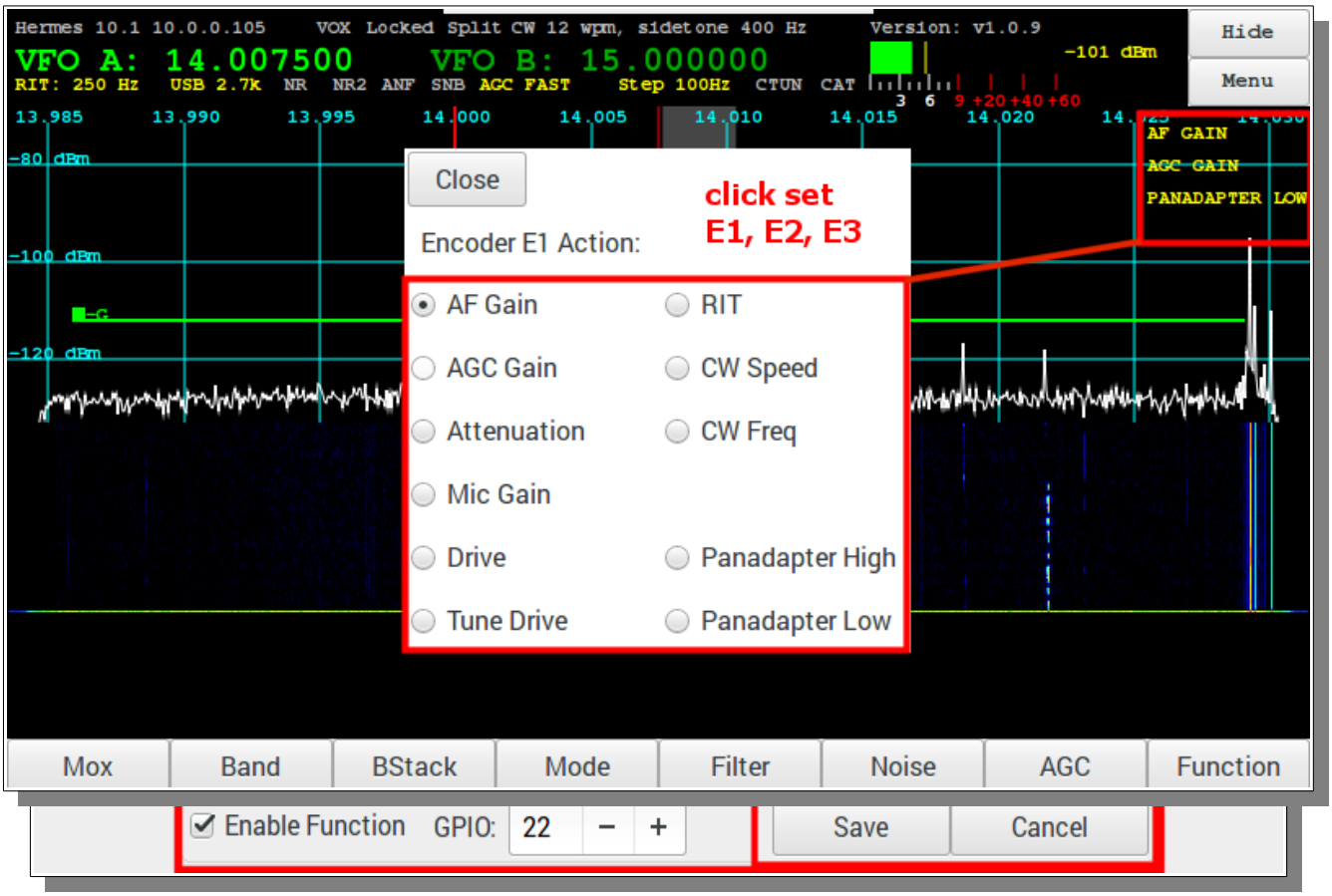
Encoders and Switches (continued)

Switch or Encoder	Function	Touch Screen	GPIO A	GPIO B
Menu	-------	Main piHPSDR Menu	13	-----
Menu → Band		Select band	-----	-----
Menu → Band Stack	3 to 5 level quick freq change	3 to 5 level quick freq change	12	-----
Menu → Mode	Mode	Mode	6	-----
Menu → Filter	IF Filter width	IF Filter Width	5	-----
Menu → Noise	Noise Blanker mode	Noise Blanker Mode	24	-----
Menu → AGC	Automatic Gain	Automatic Gain	4	21
Locked	Right click on VFO Frequency to toggle Frequency Lock	Touch VFO Frequency to toggle Frequency Lock	-----	-----
Meter	Right click on S-Meter for S-	Touch S-Meter for S-Meter	-----	-----

Switch or Encoder	Function	Touch Screen	GPIO A	GPIO B
	Meter and ALC peak/average	and ALC peak/average		

E1 E2 E3 click set 11 assignments

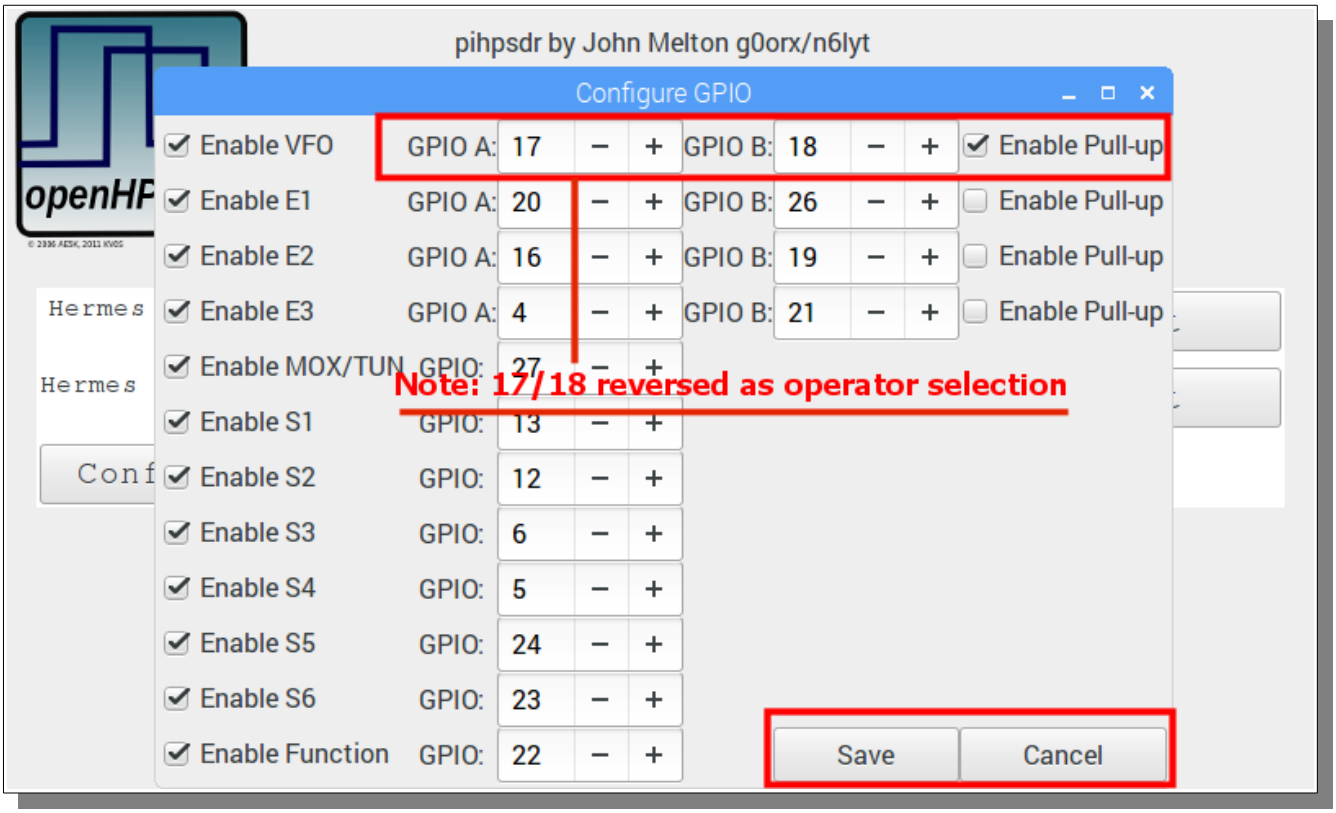
E1, E2, E3 click set



\GPIO Default assignments

GPIO pin Assignments (RaspberryPi)

Note: You may wish to **reverse the direction of Tuning on the VFO** Encoder. The GIPIO table shown at the START of pihpsdr allows convenient text entry. For example, the VFO direction of TUNING can easily be changed by swapping GPIO-A from 17 to 18, and GPIO-B from 18 to 17.



13. Reference materials

G0ORX Friedrichshaven pdf and video

Friedrichshafen 2016 publication

https://www.dropbox.com/sh/fva5d5mi93c93tq/AAD6dU-eBMR0cVK-E95hJC5la?dl=0

Jacinto Rebelo CU2ED for his homebrew of a pihpsdr Controller

Kjell Karlasen LA2NI for his complete RPi-e System

Scott WU2O homebrew Controller http://wu2o.dyndns.org/wu2o_pi_4.html

Bill Diaz KC9XG homebrew Controller and contributor to this manual

F'Hafen video on YouTube:
https://www.youtube.com/watch?v=U7QfP28YjCw

Outstanding Video from F'Hafen 2016 with KV0S Editing http://openhpsdr.org/videos.php

14. Steve Wilson KA6S special addition for Fldigi

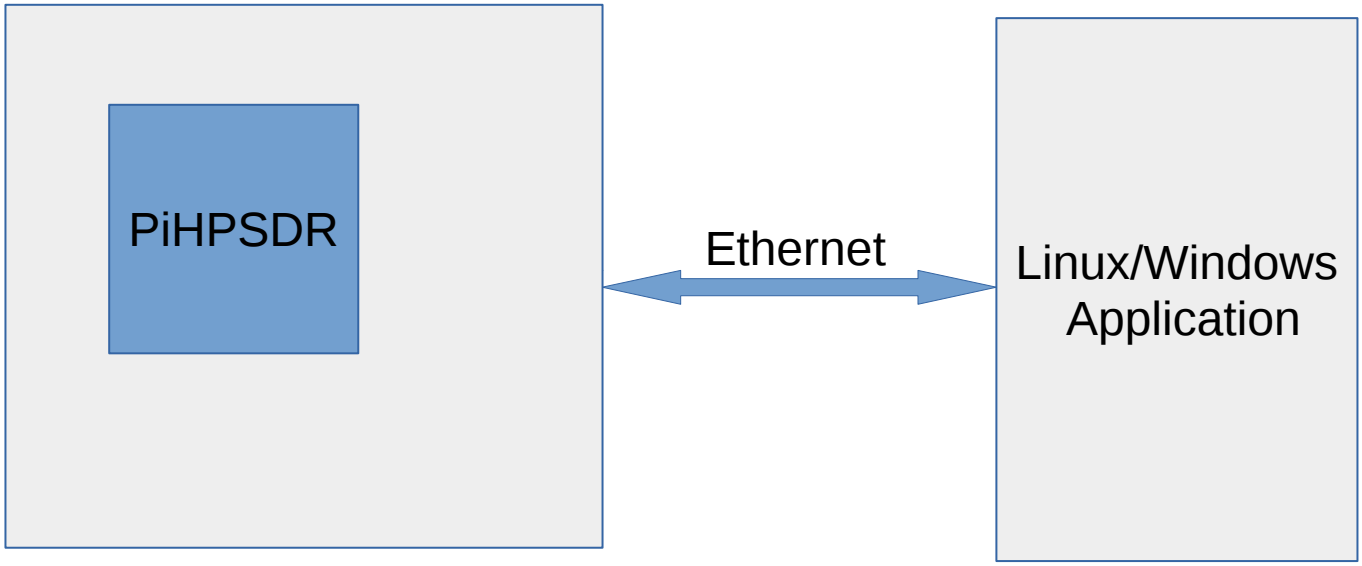
Using PiHPSDR with other applications:

PiHPSDR emulates several commands available in the CAT control found in the TS-2000. However, it receives commands over TCP/IP instead of through an RS-232 connection. For some applications this is trivial – for others some internet plumbing is required.

The Basics

PiHPSDR listens on Ports 19090-19092 for TCP/IP connections. It interprets TS-2000 protocol commands and will respond appropriately. It is NOT a full implementation. Some things don't make sense, i.e. they are features not shared between the TS-2000 and PiHPSDR. Others are still to be implemented.

Note: The latest version of Hamlib has a radio definition for the PiHPSDR, thus any application that uses Hamlib will be able to talk to PiHPSDR in its native manner, i.e. directly through TCP/IP without using the serial port. (Thanks to Jae, K5JAE for the hamlib port!) The serial port methodology is presented here for older iterations of Hamlib. To get access to to native support you will need to compile Hamlib from source and install it. That is beyond the scope of this manual.



Use the Raspberry PI
TCP/IP Address
And set for Port 19090

Mapping a Serial Port to TCP/IP port in Linux

Perhaps the easiest way to do this is via the utility “socat” available in most Linux distributions. This utility is sort of a swiss army knife for interconnectivity, and one its abilities is to map Linux Serial ports to TCP/IP ports.

To obtain “socat” and install it on an Ubuntu system type:

sudo apt-get install socat

To use it – first start PiPHSDR. You need to know the TCP/IP address of your Raspberry Pi. This will be a 32 bit number formated as XX.XX.XX.XX. I'll use 192.168.1.73 in my examples – since that happens to be what I have my Raspberry Pi set too.

To run it – type:
socat pty,link=/tmp/vtty,raw tcp:192.168.1.73:19090&
This will do two things. It creates a fake serial port called /tmp/vtty that can be opened by most applications and acts just like a serial port.

The next step is to point your application at /tmp/vtty where you would normally choose a serial port and set it to talk to a TS-2000. It really is that simple!

This allows applications that are built to use Hamlib to talk to PiHPSDR.

Example: Set up GRIG to talk to PiHPSDR

grig –-model=214 –-rig-file=/tmp/vtty *

/tmp/vtty is a temporary file – it disappears as soon as socat ends.

Grig is built around the Hamlib radio access library. The TS-2000 is model 214 in Hamlib. Any application that uses Hamlib should be able to communicate with PiHPSDR by using /dev/vtty and choosing model 214 as the radio.

Something else to realize is that only one application can talk to PIHPSDR at a time on a given port. PiHPSDR can use any of 19090-19092 as the CAT port, and supports 3 simultaneous connections.

If you have the latest version of Hamlib that supports PiHPSDR natively – the command to start grig would be:

grig –model=240 –rig-file=192.168.1.73:19090 &

Example: Connect FLDIGI to PiHPSDR

FLDIGI can be connected to PiHPSDR using the native PiHPSDR interface built into the latest version of Hamlib. However, it polls PiHPSDR way to fast.

Have no fear – FLRIG is here. FLRIG is a companion application to FLDIGI that can act as a server

for FLDIGI.

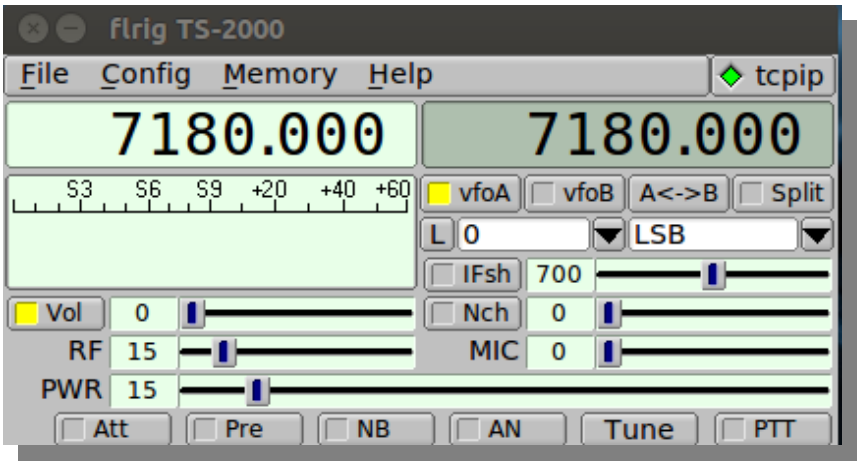
You can install FLRIG on with:
sudo apt-get install flrig

You can install FLDIGI with:
sudo apt-get install fldigi

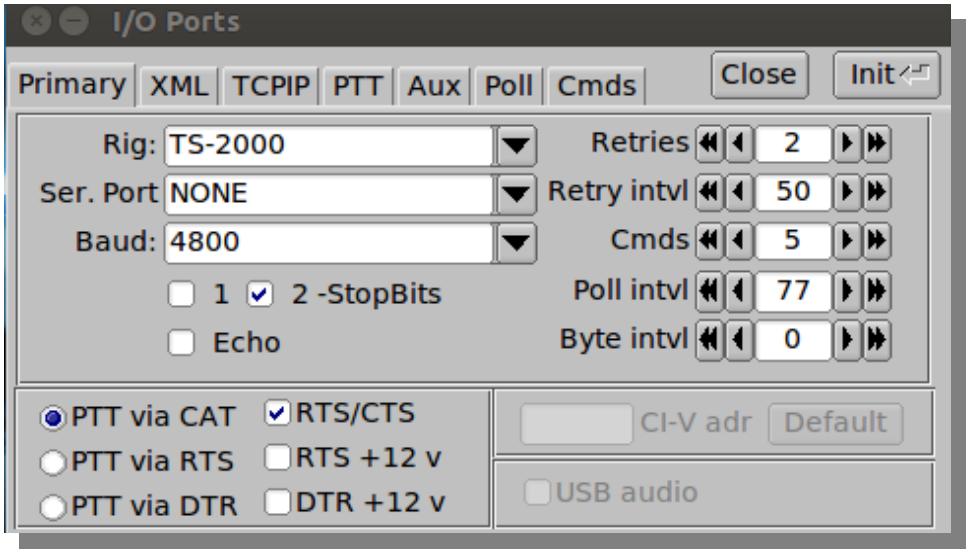
Start FLRIG first with

flrig

No need to start it with sudo since it talks directly to PiHPSDR.



To set up FLRIG – Hit the Config button on the top of the application and Choose Config/Setup/Transceiver – the Window below will pop up.



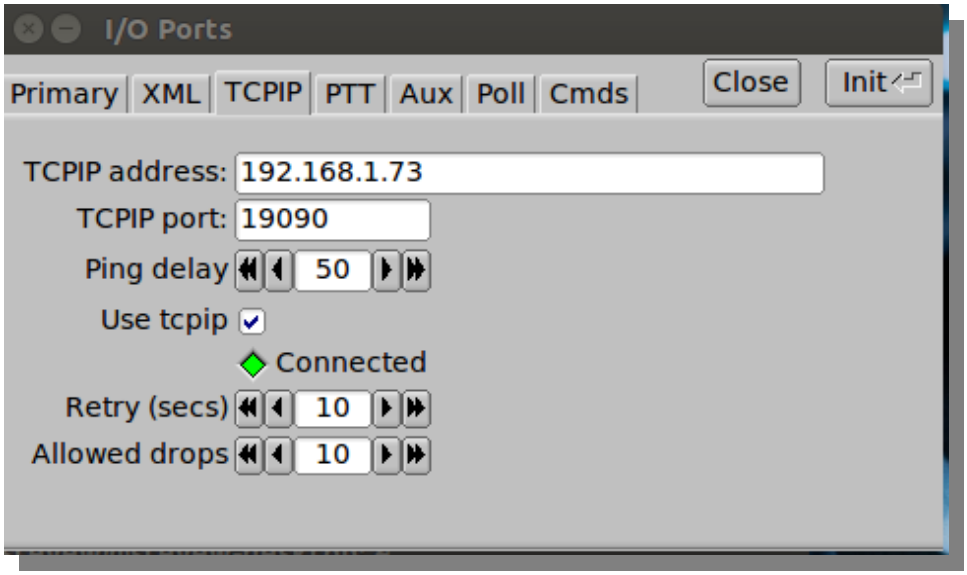
Use the arrow to the right of the Rig: box and slide it to TS-2000.

Also – check the PTT via CAT button in the lower left corner.

Hit Close

Now Choose Config/Setup/TCPIP

The window below will pop up.



Select the TCPIP tab. Enter the TCPIP address of your Raspberry Pi – in our example you see 192.168.1.73. Type in the TCPIP box 19090 and that should do it. Hit the “Use tcpip” button. If the connected light isn't green – hit the Init button in the upper right portion of the window.

Hit Close and you should have FLRIG up and controlling your PiHPSDR.

The Frequency should work by using the scroll wheel over the digit you want to change.

The volume, RF, PWR, and MIC sliders should all work.

The PTT should cause the radio to transmit.

The S Meter should register.

Now get FLDIGI running.

Type:

fldigi

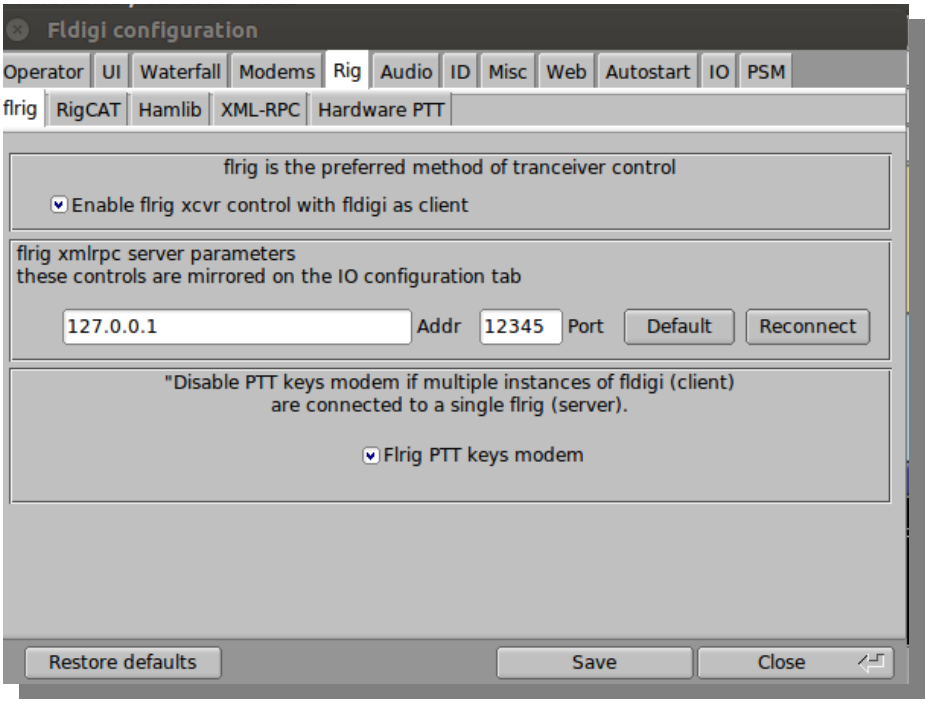
Choose Configure/Rig Control

Hit the flrig tab and the window below will pop up.

Hit the “Enable flrig xcvr control with fldigi as client” button.

Note that the TCP/IP address is 127.0.0.1 – which is the same as the machine you are running flrig and fldigi on. The TCP/IP port 12345 should already be configured.

Also hit the “Flrig PTT keys modem”



That is it.

You need to use actual audio cables from the Radio to your Linux computer for the audio as of now – but this should be all that is necessary for FLDIGI to talk to your radio!

Setting up rigctld

The Hamlib system must be installed to use some packages – specifically to run the rigctld daemon.

On Ubuntu:

sudo apt-get install libhamlib-utils
sudo apt-get install libhamlib2

Note: These packages don't yet have PiHPSDR supported natively. It is necessary to compile the latest version from source and that is beyond the scope of this manual.

Multi-Client Support

PiHPSDR can communicate with up to 3 independent clients simultaneously. An example of this might be running FLRIG/FLDIGI with the logging program CQLOG.

First – make sure you have CQLOG installed.

On Ubuntu:

sudo apt-get install CQLOG

We need do three things to get cqrlog running.
1) Create a virtual terminal – see the instructions above about running cqrlog. There is one change here. We're going to aim the virtual serial port /tmp/vtty at TCP/IP port 19091 instead of 19090.
2) Start up the hamlib rigctld – this is a “daemon” in Unix parlance that is responsible for interfacing between applications and the virtual terminal using the internal hamlib protocol. Note that SOME applications have the hamlib software built in, while others use the daemon – cqrlog uses the daemon. The rigctld must be started using “sudo”
3) Start up cqrlog. Specifically do NOT use “sudo” for this command.

In a single xterm – you can start up both socat AND the rigctld daemon. Note again the use of “sudo” prior to the command AND note that the first program can be put into the background safely. Rigctld doesn't work if you do that..

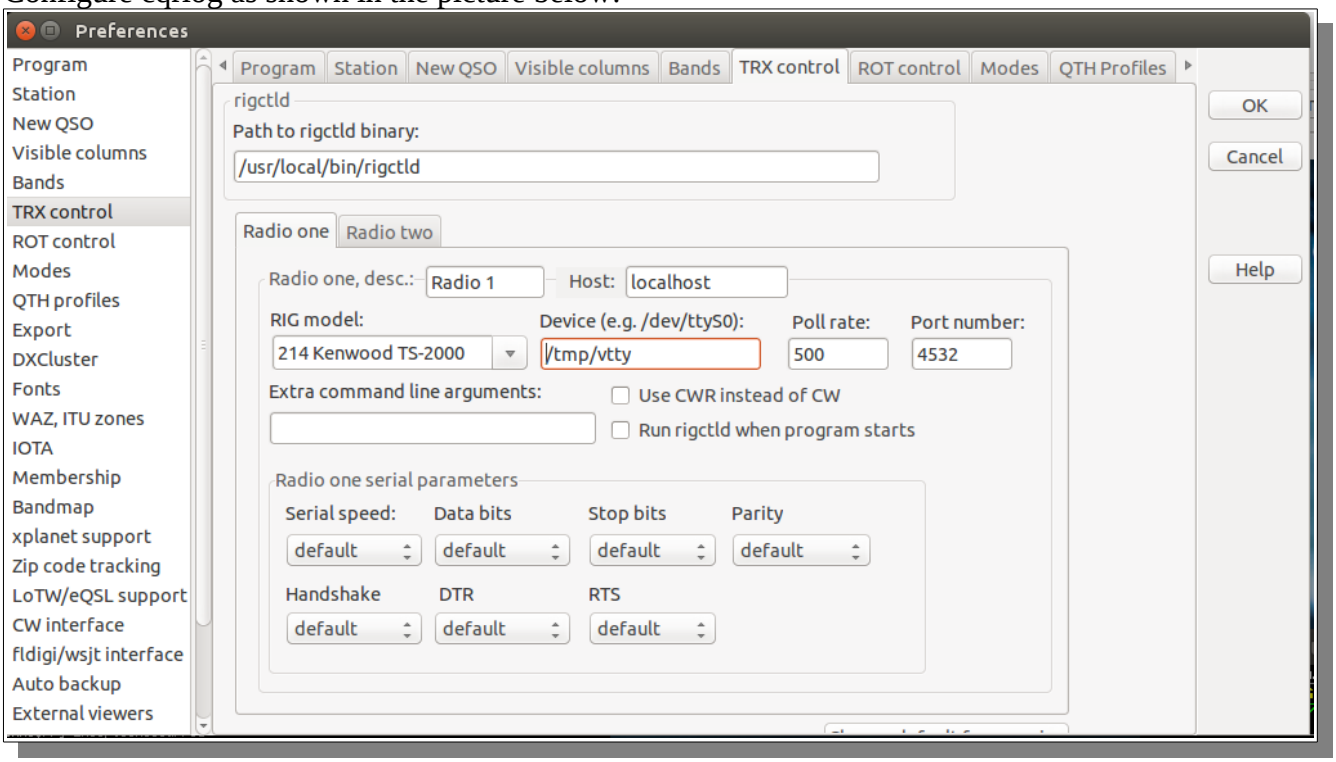
```
tevew@stevew-desktop:~$
tevew@stevew-desktop:~$
tevew@stevew-desktop:~$
tevew@stevew-desktop:~$
tevew@stevew-desktop:~$ sudo  socat  pty,link=/dev/vtty,raw  tcp:192.168.1.73:19091 &
[1] 19226
tevew@stevew-desktop:~$ sudo /usr/bin/rigctld -m 214 -r /dev/vtty
```

In another xterm – start up cqrlog

Enter: cqrlog &

This will start up cqrlog talking the PiHPSDR radio using port 19091. Note that this can be safely put into the background using the ampersand operator.

Configure cqrlog as shown in the picture below:



Note that we are choosing to NOT launch rigctld when the program starts – but rather starting it by hand.

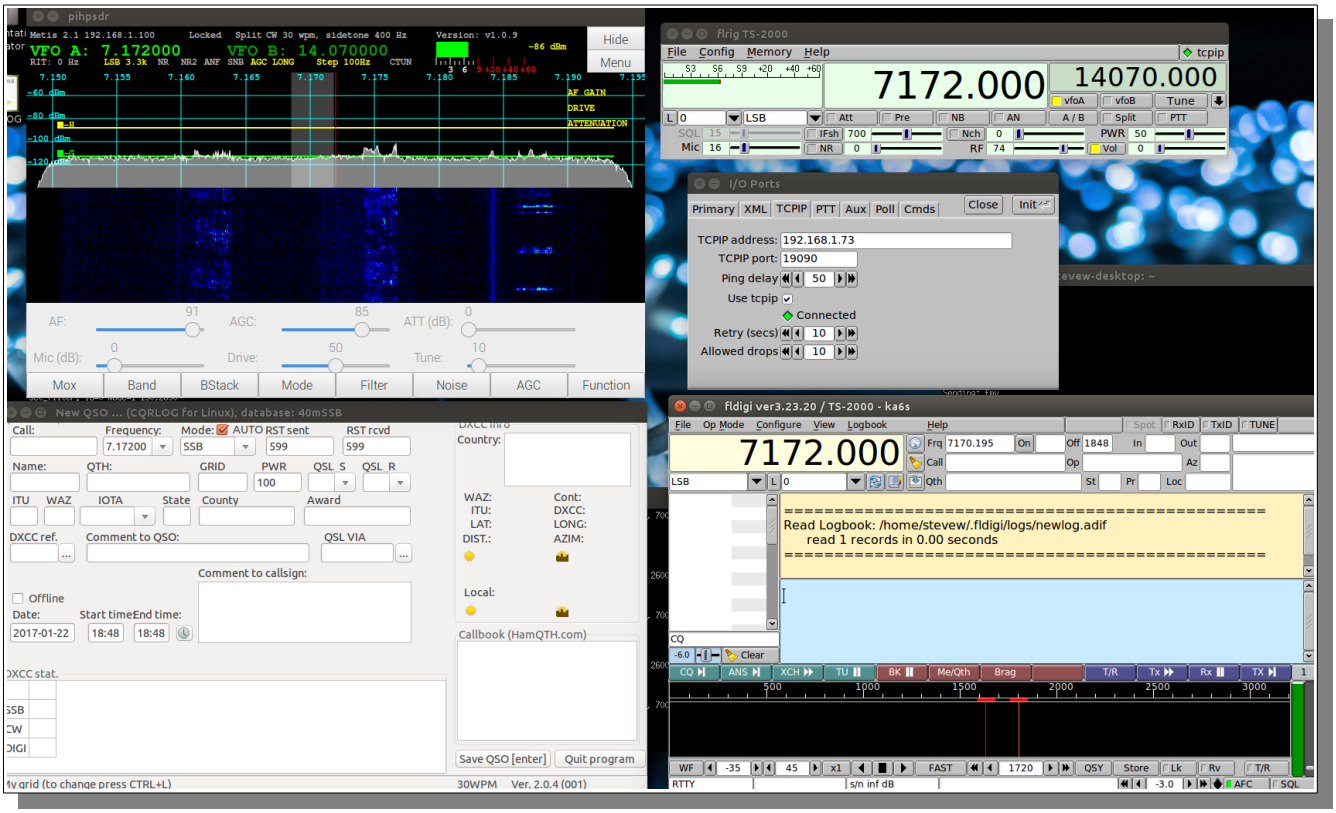
Next – start up flrig as you have previously using port 19090. It should be configured correctly from the first time you used it. Since cqrlog spouts messages – you may need to use another xterm…

% flrig &

And this you can start up FLDIGI using the same xterm..

%fldigi

The picture below shows all applications running – using the two TCP/IP ports.



Running applications with the latest version of Hamlib

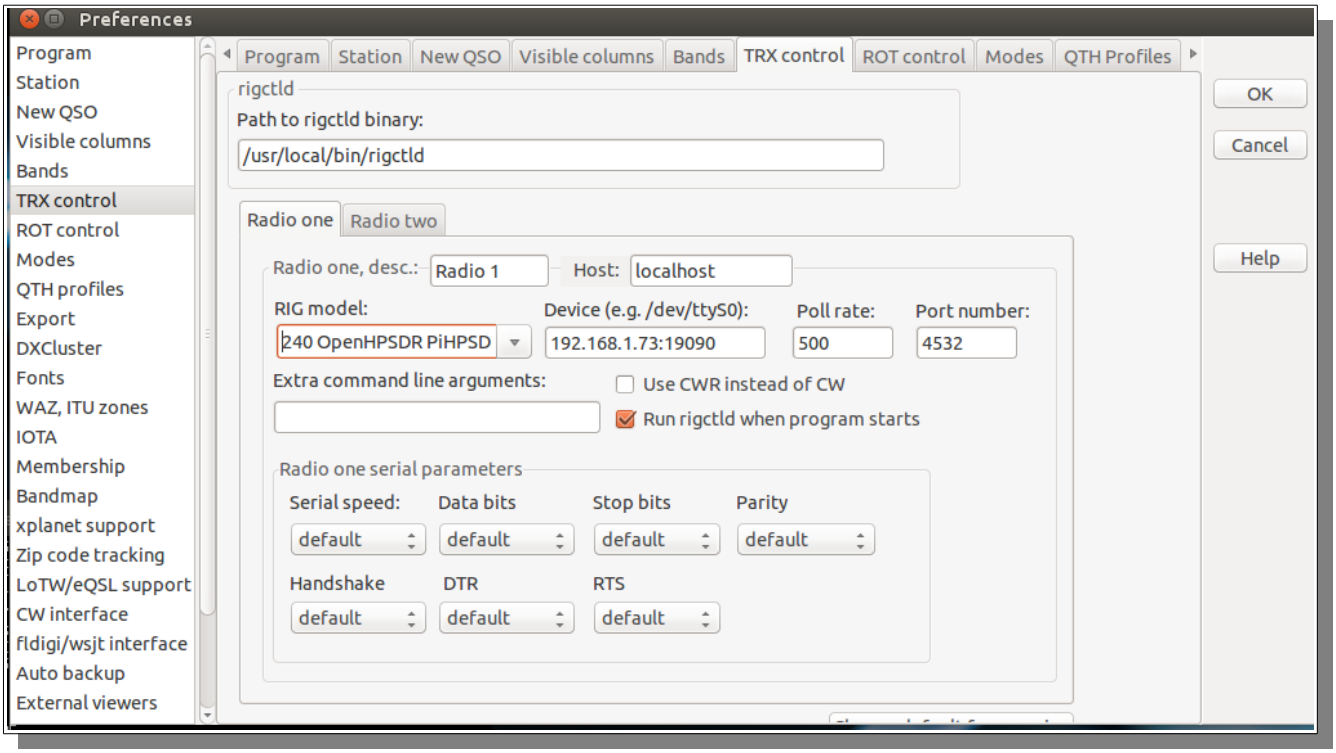
The biggest advantage of running applications through latest version of Hamlib is that the whole virtual terminal concept is removed! Applications can talk directly to PiHPSDR without any middleware. I'll use CQRLOG to illustrate the setup.

The first trick is to make rigctld runnable without being root. THIS IS A SECURITY HOLE – Proceed at your own risk! The following command allows rigctld to be run without being root…

sudo chown pi /usr/local/bin/rigctld

This makes the user “pi” owner of the rigctld daemon.

Next – start up cqrlog and set up its preferences as shown in the next picture.



Things to note:
Rig Model is OpenHPSDR PiHPSDR – radio model 240.
Device: Use the TCP/IP/Port number – so 192.168.1.73:19090
The “Run rigctld when program starts” is checked – since we don't have to root to run rigctld.

It is also possible to start cqrlog on OTHER than the Raspberry Pi. The settings would be the SAME since the device entry is what actually aims the communications at the PiHPSDR TCP/IP port. Rigctld can be started on another machine and it will be able to communicate via TCP/IP directly to the PiHPSDR application.

KA6S Summary

PiHPSDR provides 3 separate TCP/IP channels on ports 19090, 19091, and 19092 which can communicate with other applications using an emulation of the TS-2000 command set.

Some applications require a virtual serial port – use the socat command run to create the virtual serial port.

The very latest version of Hamlib supports PiHPSDR natively. Radio model is 240, and uses TCPIP Address/Port Number instead of the serial port.

Other applications require the hamlib rigctld to be present – start /usr/bin/rigctld with sudo.

To run FLDIGI – use FLRIG as the server for FLDIGI since FLDIGI has issues talking to the virtual serial port.

15. Supported CAT Commands

AG Command Sets/Reads AF Slider

	1	2	3	4	5	6	7	8	9	10
Set	A	G	P1	P2	P2	P2	;			
Read	A	G	;							
Response	A	G	P1	P2	P2	P2	;			

Notes:
P1 Ignored
P2 000-255 linearly mapped to 0-100 range

FA Command Sets VFO A Frequency

	1	2	3	4	5	6	7	8	9	10
Set	F	A	P1	P1	P1	P1	P1	P1	P1	P1
	P1	P1	P1	;						
Read	F	A	;							
Response	F	A	P1	P1	P1	P1	P1	P1	P1	P1
	P1	P1	P1	;						

Frequency in Hertz (11- digits)
Note that blank digits should be 0
Always reads VFO_A

FB Command Sets VFO B Frequency

	1	2	3	4	5	6	7	8	9	10
Set	F	B	P1	P1	P1	P1	P1	P1	P1	P1
	P1	P1	P1	;						
Read	F	B	;							
Response	F	B	P1	P1	P1	P1	P1	P1	P1	P1
	P1	P1	P1	;						

Frequency in Hertz (11- digits)
Note that blank digits should be 0
Reads VFO_A when one receiver operation is selected.

FR Command Sets/Reads which VFO is active receiver

	1	2	3	4	5	6	7	8	9	10
Set	F	R	P1	;						
Read	F	R	;							
Response	F	R	P1	;						

P1=0 – VFO A 1=VFO B
Only in 2 receiver mode will cause frequency swapping to between VFO A and VFO B and if Transmit != Receive VFO – Split will set.

FT Command Sets/Reads which VFO is active transmitter

	1	2	3	4	5	6	7	8	9	10
Set	F	T	P1	;						
Read	F	T	;							
Response	F	T	P1	;						

P1=0 – VFO A 1=VFO B
Sets Split if Active Transmitter not the same as active receiver

FW Command Sets/Reads receive filter width (Only for CW mode)

	1	2	3	4	5	6	7	8	9	10
Set	F	W	P1	P1	P1	P1	;			
Read	F	W	;							
Response	F	W	P1	P1	P1	P1	;			

P1 – (0000-9999 in Hz)
Frequency are mapped from TS2000 to PiHPSDR
CW 25/50→50 100→100 250→300 400->400 500→500 600→600 750→1000 800→1000

GT Command Set/Read AGC constant values

	1	2	3	4	5	6	7	8	9	10
Set										
Read	G	T	;							
Response	G	T	P1	P1	P1	;				

TS-2000 legal values 000-020
PiHPSDR = 000 = Off, 005=Fast, 010=Medium
010=Medium, 015=Slow, 020=Long
Note: Hamlib will send values as N*84 - PiHPSDR can detect commands using this scale and SET its internal state accordingly – but can only read back legal values according to the 000-020 mapping.

ID Command Read the transceiver ID number

	1	2	3	4	5	6	7	8	9	10
Set										
Read	I	D	;							
Response	I	D	P1	P1	P1	;				

P1 = 019: TS-2000

IF Command Sets VFO A Frequency

	1	2	3	4	5	6	7	8	9	10
Set										
Read	I	F	;							
Response	I	F	P1	P1	P1	P1	P1	P1	P1	P1
	P1	P1	P1	P2	P2	P2	P2	P3	P3	P3
	P3	P3	P3	P4	P5	P6	P7	P7	P8	P9
	P10	P11	P12	P13	P14	P14	P15	;		

P1 = Frequency in Hz (11 digits)
P2 = Step in Hertz (5 digits)
P3 = Active Receiver RIT in Hertz (5 digits)
P4 = 0: Rit Off 1: Rit On
P5 = 0: Rit Off 1: Rit On
P8 = 0: Mox=0 1: Mox=1
P9 = Radio Mode (See MD command)
P12 = 0: Split off 1: Split on
All other parameters are 0.

KS Command Sets and reads keyer speed

	1	2	3	4	5	6	7	8	9	10
Set	K	S	P1	P1	P1	;				
Read	K	S	;							
Response	K	S	P1	P1	P1	;				

P1 – 010 (min) – 060 (max) in WPM

LK Command Sets/reads the lock function

	1	2	3	4	5	6	7	8	9	10
Set	L	K	P1	P2	;					
Read	L	K	;							
Response	L	K	P1	P2	;					

Notes:
P1 = 0: Unlock 1: Lock
P2 ignored

MD Command Sets/Reads radio Mode

	1	2	3	4	5	6	7	8	9	10	
Set	M	D	P1	;							P1 = 1: LSB 2: USB 3: CWU 4: FMN 5: AM
Read	M	D	;								6: DIGL 7: CWL 9: DIGU
Response	M	D	P1	;							

MG Command Sets/Reads Mic Gain Slider

	1	2	3	4	5	6	7	8	9	10	
Set	M	G	P1	P1	P1	;					P1 – 000 (min) – 100 (max)
Read	M	G	;								
Response	M	G	P1	P1	P1	;					

NB Command Sets/Reads the Noise Blanker function status

	1	2	3	4	5	6	7	8	9	10	
Set	N	B	P1	;							P1 = 0: Off 1: On
Read	N	B	;								
Response	N	B	P1	;							

NR Command Sets/Reads the Noise Blanker function status

	1	2	3	4	5	6	7	8	9	10	
Set	N	R	P1	;							P1 = 0: Off 1: NR On 2: NR2 On
Read	N	R	;								
Response	N	R	P1	;							

NT Command Sets ANF bit (Autonotch in TS2000)

	1	2	3	4	5	6	7	8	9	10	
Set	N	T	P1	;							P1= 0: ANF Off 1: ANF On
Read	N	T	;								
Response	N	T	P1	;							

PC Command Sets/Reads Drive Slider

	1	2	3	4	5	6	7	8	9	10	
Set	P	C	P1	P1	P1	;					P1 – 005 (min) – 100 (max)
Read	P	C	;								
Response	P	C	P1	P1	P1	;					

RD Command Move RIT off frequency Down

	1	2	3	4	5	6	7	8	9	10	
Set	R	D	P1	P1	P1	P1	P1	;			P1,P2 ignored
Read	R	D	;								decrements rit_frequency by rit_increment
Response	R	D	P2	;							

RG Command Sets/Reads AGC slider

	1	2	3	4	5	6	7	8	9	10
Set	R	G	P1	P1	P1	;				
Read	R	G	;							
Response	R	G	P1	P1	P1	;				

P1 = 000 (min) to 255 (max) linearly scaled to -20 to 140 range of slider

RT Command Sets/Read the RIT function status

	1	2	3	4	5	6	7	8	9	10
Set	R	T	P1	;						
Read	R	T	;							
Response	R	T	P1	;						

P1 – 0: Off 1: On
Returns state of active receiver RIT

RU Command Move RIT off frequency Up

	1	2	3	4	5	6	7	8	9	10
Set	R	U	P1	P1	P1	P1	P1	;		
Read	R	U	;							
Response	R	U	P2	;						

P1,P2 ignored
increments rit_frequency by rit_increment of active receiver

RX Command Set Mox to 0 (turn off transmitter)

	1	2	3	4	5	6	7	8	9	10
Set	R	X	;							
Read										
Response										

SD Command Set /Read CW Breakin Delay

	1	2	3	4	5	6	7	8	9	10
Set	S	D	P1	P1	P1	P1	P1	;		
Read	S	D	;							
Response	P1	P1	P1	P1	P1	;				

P1 = 0000-1000 ms Breakin delay
0000= Set Full Breakin
If PiHPSDR has values above 1000 internally a value of 1000 is reported

SM Command Reads the S Meter

	1	2	3	4	5	6	7	8	9	10
Set										
Read	S	M	P1	;						
Response	S	M	P1	P2	P2	P2	P2	;		

P1 – 0: Main, 1: Sub
P2 = 0000 (min) to 0030 (max) main xcvr
0000 (min) to 0015 (max) sub xcvr

ST Command Sets/reads the frequency step

	1	2	3	4	5	6	7	8	9	10
Set	S	T	P1	P1	;					
Read	S	T	;							
Response	S	T	P1	P1	;					

Note that internal value is categorized into These slots for the read operation, Set uses the assigned value.
P1 – SSB 00-1 KHz 01: 2.5KHz 02 5KHz 03: 10KHz
AM/FM: 00: 5KHz 01: 6.25KHz 02: 10KHz 03: 12.5KHz 04: 15KHz 05: 20KHz 06: 25KHz 07: 30KHz 08: 50KHz 09: 100KHz

TX Command Set the transmitter MOX

	1	2	3	4	5	6	7	8	9	10
Set	T	X	;							
Read										
Response										

VD Command Set/Read Vox Delay

	1	2	3	4	5	6	7	8	9	10
Set	V	D	P1	P1	P1	P1	;			
Read	V	D	;							
Response	V	D	P1	P1	P1	P1	;			

P1 – 0000 (min) to 1000 (max)
Values are limited to these boundaries

VG Command Set/Read Vox Threshold (Vox GAIN on TS2000)

	1	2	3	4	5	6	7	8	9	10
Set	V	D	P1	P1	P1	;				
Read	V	D	;							
Response	V	D	P1	P1	P1	;				

Set Vox Threhold
P1 = (ooo) min to (oo9) max

16. Apache Support

Apache Labs International Support

Technical support for ANAN-10 from the factory is available via the Apache Labs Yahoo Group
http://groups.yahoo.com/group/apache-labs/

or directly via email **<support@apache-labs.com>**

Apache Yahoo Support Group

http://groups.yahoo.com/group/apache-labs/

OpenHPSDR Group

Instructions relating to joining the OpenHPSDR Group reflector are here:

http://lists.openhpsdr.org/listinfo.cgi/hpsdr-openhpsdr.org

The OpenHPSDR archives may also be searched here:

http://lists.openhpsdr.org/mmsearch.cgi/hpsdr-openhpsdr.org

The latest version of the OpenHPSDR User Manual can be obtained from

http://openhpsdr.org/documents.php Author Phil Harman VK6PH

17. Apache Service and Repair

1023 Tower B4, Spaze I-Tech Park

Sector - 49, Sohna Road

Gurgaon - 122001

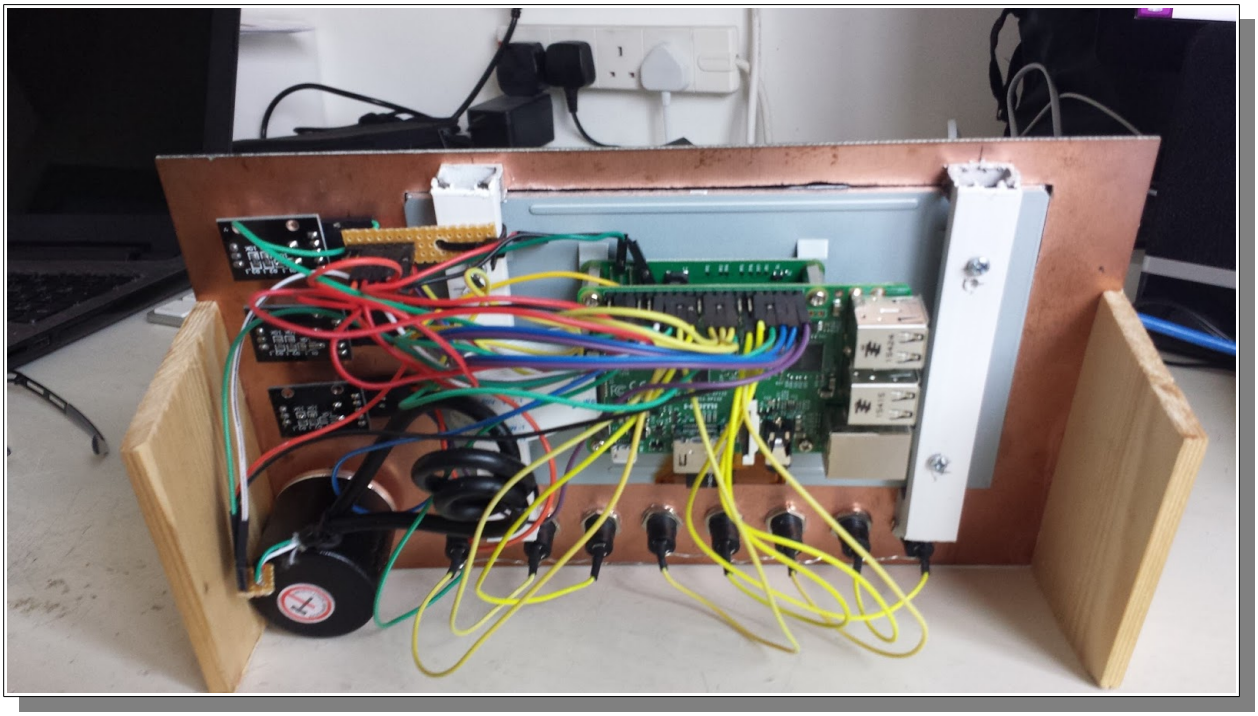
Haryana, India

Tel: 91-0124-4245173/4/5 (10AM - 6PM IST]

Email: support@apache-labs.com

Website: **http://www.apache-labs.com**

Early prototypes by John Melton G0ORX (circa: 2015)



Note: pihpsdr is OpenSource. If you would like to run the program on a Linux system, the source, binaries, and documentation are located here: **http://g0orx.blogspot.com/**

NOTES

Alphabetical Index