A Winlink HF Gateway Using an Inexpensive HF SDR Transceiver
Craig J. Coley, WB5KUO

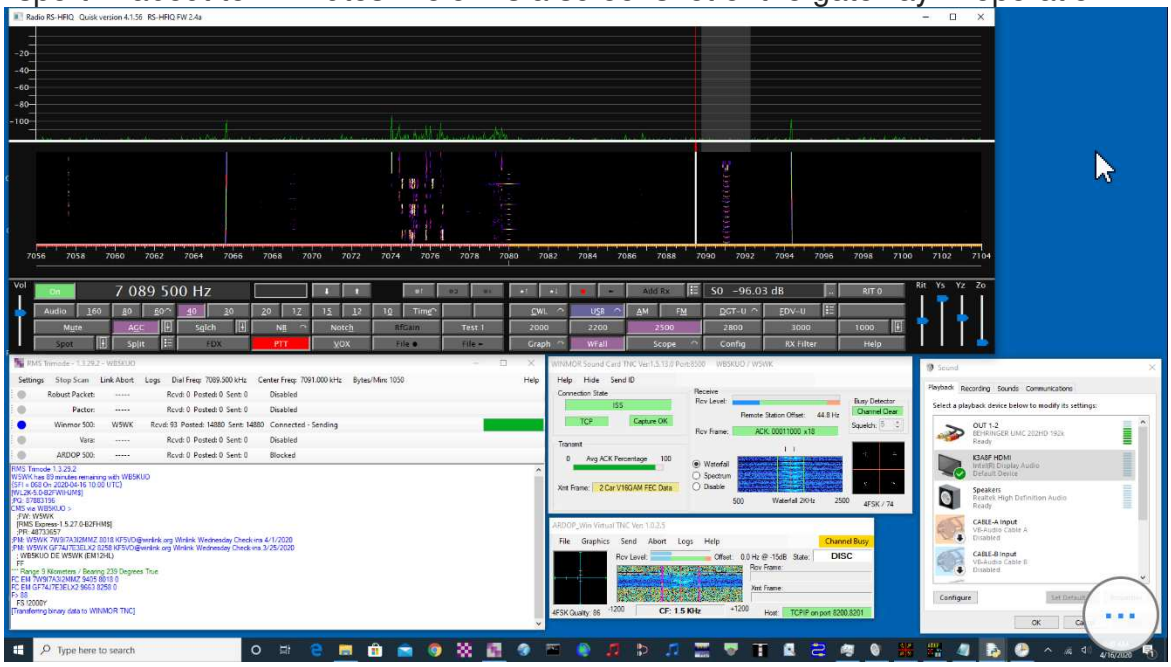
Worldwide Winlink HF Gateways enable remote and maritime mobile stations access to internet connected email using amateur radio frequencies. In times of emergency, the highly redundant and geographically diverse nature of Winlink gateways ensures transfer of critical information to relief agencies hundreds or even thousands of miles distant.

The typical Winlink HF Gateway involves an amateur radio HF transceiver, digital rig control interface, sound card, Windows based computer, and specialized Winlink control and modem software. Constructing a dedicated HF Gateway using this approach can easily cost several thousand dollars. By using the available computer resources for more than just rig control and Winlink interface, it is technically possible to reduce the installation size and cost by using an inexpensive SDR HF transceiver.

Before building a Winlink HF Gateway, all Winlink Sysops must be approved by the Winlink System Administrator before being allowed to connect to the Winlink network. This means you MUST have a Winlink account, be an active user of radio email, and be willing to setup dedicated hardware with backup power for the task:

https://winlink.org/content/join_gateway_sysop_team_sysop_guidelines

The WB5KUO Winlink HF Gateway runs Winmor500 on 7089 and 7091 and Winmor1600/VARA on 7102. The dedicated gateway hardware, including the computer and monitor, operate on a 12 VDC Lithium battery with power distribution individually fused using a Powerpole distribution panel. If needed, the entire unit can be readied for transport in about ten minutes. Below is a screenshot of the gateway in operation:



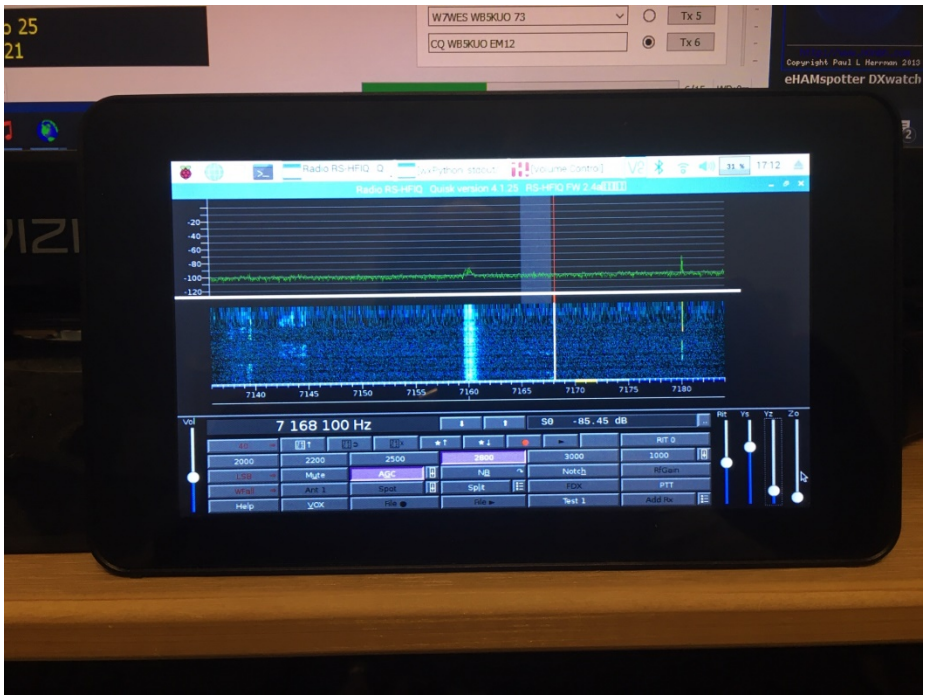
SDR Transceivers in a Winlink Environment

A Software Defined Radio (SDR) uses computer resources to create virtual operator controls, programmable filtering, multimode demodulation, AGC, and audio processing normally performed by transceiver hardware. By performing these functions in software, the transceiver hardware can be greatly simplified to reduce complexity and cost. An example of a low cost SDR HF transceiver is the RS-HFIQ transceiver available for $239.00 from Hobby PCB https://www.hobbypcb.com/. This transceiver operates 80m-10m at 5W and uses an open-source hardware and software design.

The computer processing of in-phase and quadrature (IQ) signals to and from the SDR takes time and this contributes to a common complaint with SDRs, latency. Latency is the delay between when a signal is received at the antenna and when it appears as audio to the operator. In some SDR programs like HDSDR, this can take 500-700 milliseconds, far beyond the 250 millisecond maximum latency allowed for Winlink. In addition to the SDR software itself, other contributors to latency are the soundcard driver, sound card buffer size, and Virtual Audio Cables (VAC). Each of these sources of latency must be addressed to meet the Winlink latency requirements.

Quisk SDR Software

The lowest latency SDR program tested is Quisk, originally written in 2008 by James Ahlstrom, N2ADR. N2ADR wrote Quisk in Python to support a direct sampling SDR of his own design for QSK CW so its latency is VERY low. Since it is written in Python, it will work on Windows or Linux, including the Raspberry Pi. A Raspberry Pi 3B+ running Quisk on a touchscreen for an SSB voice transceiver is shown below:



Quisk is a powerful and generic SDR program not tied to any specific hardware but as such, it comes with a steep learning curve and configuring it from scratch can be difficult. Discussions regarding Quisk configuration in this paper will be limited to that which is necessary to operate the RS-HFIQ transceiver in a Windows environment.

It was discovered during testing that Quisk has a memory leak that slowly increases transmit latency over time. This issue was addressed by creating a task in Z-CRON to restart Quisk at the top of every hour. Since the Quisk restart is transparent to RMS Trimode and takes only about 5 seconds, testing showed only a slight increase in link time with no dropped connections with either Winmor500 or VARA. The setup of the Quisk_Restart task in Z-CRON will be described later in the text.

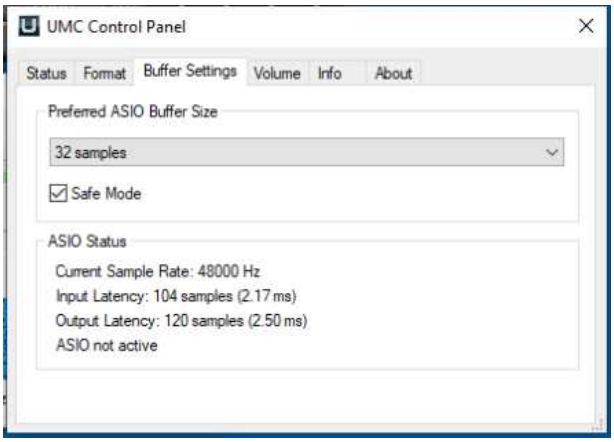
A Word about Computer Hardware

Running SDR software along with the Winmor, ARDOP and VARA modems is processor intensive. While I have configured a Winlink SDR HF Gateway using an Intel Atom 1.3 GHz Mini-PC with only 4 GB RAM, the processor and memory resources ran well over 50% and the latency was not consistently below 250 milliseconds. For this reason, it is recommended to use at least an Intel Core i3 running at 1.7 GHz, 8 GB of RAM, 64 GB SSD, and Windows 10x64. The computer used for this installation was a 12 VDC QOTOM Mini PC - i3 4010U, purchased used on eBay for $115.00, including Windows 10x64. Windows was fully patched before installing the rest of the software:

1. Install the USB Soundcard Driver

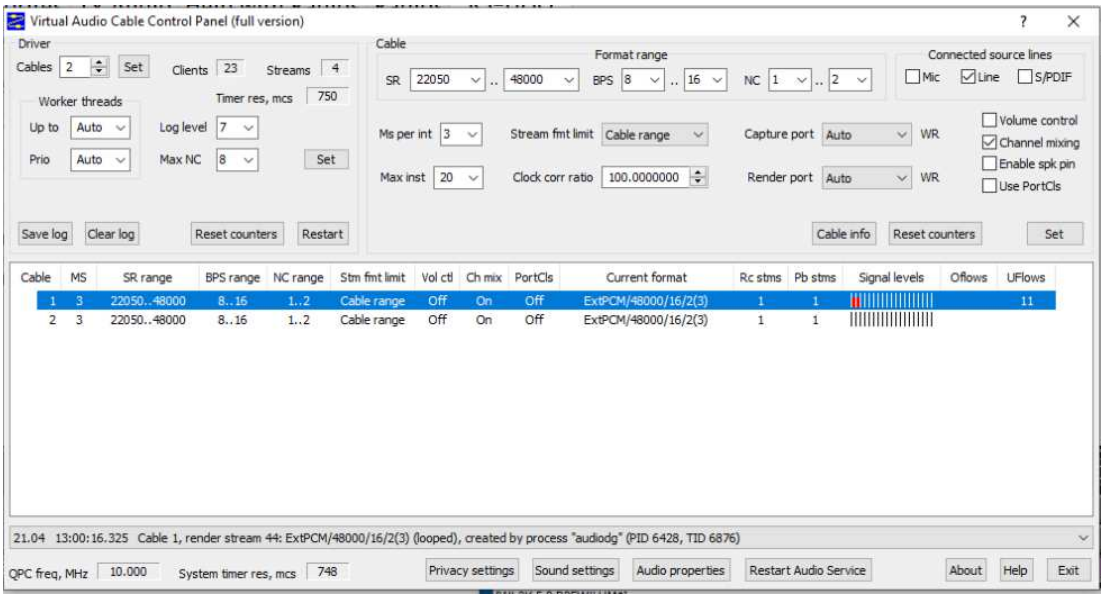
A major contributor to latency can be the soundcard if it is not handled efficiently in the Windows environment. Another problem is not having sufficient resolution to handle strong signals within the receiver passband. It is for this reason that the 24 bit Behringer UMC202HD was chosen. The UMC202HD can operate up to 192K samples per second and has a driver that allows the buffer size to be specified. When installing the driver, set the Buffer Settings to 32 samples and Safe Mode. Note the calculated latency of 2.17 and 2.50 milliseconds:

For best result, it is recommended that the PAD be engaged for both channels, GAIN 1 and GAIN 2 be set to ¼ scale, and OUTPUT set to ¾ scale.



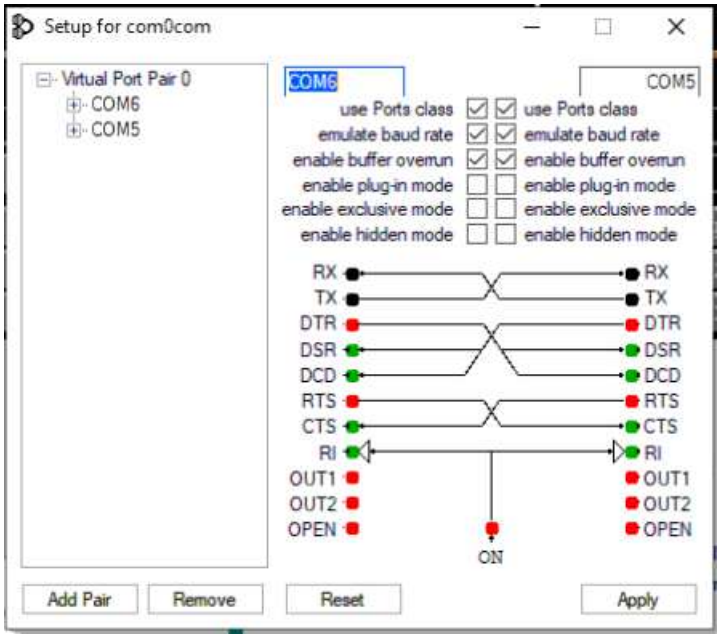
2. Install the Virtual Audio Cable (VAC)

The VAC software can be downloaded from https://vac.muzychenko.net/en/. This software costs $30 for personal use but has the lowest latency of any virtual audio cable evaluated. Once installed, enter the configuration menu and create a second cable. When configured, there should be VAC Line 1 and Line 2. Line 1 will be used to transfer the receive audio from Quisk to the Winlink modems and Line 2 will be used to transfer the microphone audio from the Winlink modems to Quisk:



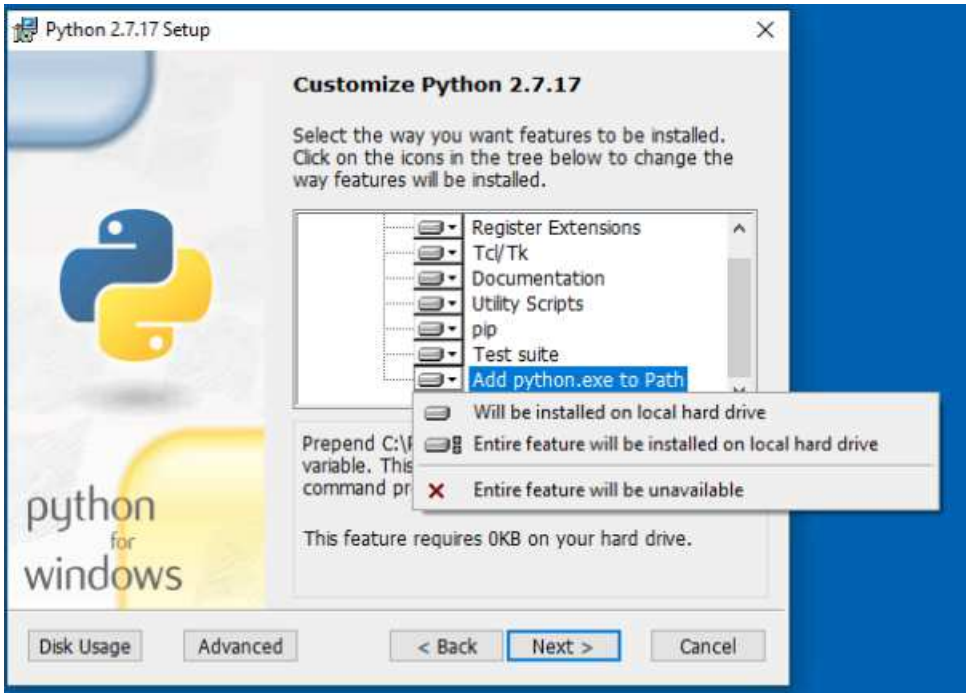
3. Install the Virtual COM Port for Rig Control

Since Quisk will essentially be replacing the controls on an external rig, a virtual COM port pair is required so that Winlink can control the rig as if it was an external radio. Com0Com is an excellent program to do this and can be downloaded from https://sourceforge.net/projects/com0com/ . In this installation, COM5 and COM 6 were chosen using the configuration as shown below. Quisk will eventually be receiving rig control on COM 6 while RMS Trimode will be sending rig control on COM 5:



4. Install Python 2.7

Before Quisk can be installed, the Python 2.7 interpreter must be installed. This is a free download from http://www.Python.org. Python 2.7 is not the latest version but is required to operate with the RS-HFIQ rig control program that was written several years ago. Make sure you select the option to add Python to the system path. Quisk is compatible with both the 32 bit and 64 bit versions of Python so install the version that is compatible with your computer.



5. Install Quisk

Once Python is installed, uses the following instructions from N2ADR's website to install Quisk and all of its required components.

http://james.ahlstrom.name/quisk/docs.html#Installation

Windows Initial Installation

If you have Python, you can continue to use the version you have if it is recent. If you don't have Python installed, visit http://www.python.org and download the Windows installer for the most recent Python3. Use 64-bit for a 64-bit computer. To get there from python.org, click on "Downloads", then "Windows", then "Latest Python 3 Release". Under "Files" click on "Windows installer". The x86 is the 32-bit version and x86-64 is the 64-bit version. When installing, write down your install directory so you know where Python is located. There is an option to "Add python.exe to Path". It is best to select this option so that you can start python by just typing "python" instead of the whole path from your install directory. But don't do this if you have multiple versions of Python.

Next open a Command Prompt. This is located under "Windows System" on Windows 10; or use the search bar. Enter "python --version" and then "pip --version" to make sure that these programs are installed, and what version you have. If "python" is not found, it might not be on your Path. Try your_install_directory\python --version. For pip, it is your_install_directory\Scripts\pip --version. If you can find Python but not pip, you can run pip with "python -m pip".

First upgrade some Python modules to the newest version, then install Quisk. Enter these commands. The upgrade of pip gave error messages on my machine, but it worked anyway.

pip install --upgrade pip

pip install --upgrade setuptools

pip install --upgrade wxPython

pip install --upgrade pyserial

pip install --upgrade quisk

You should then be able to start Quisk with the command "quisk" or "your_install_directory\Scripts\quisk". To create a Quisk shortcut on your desktop, right-click an empty space and select "New" and "Shortcut". Use "your_install_directory\Scripts\quisk.exe" as the command and "Quisk" as the name.

6. Install the RS-HFIQ Python Script for Rig Control

Because Quisk is generic, it expects a Python rig control program for radios it is expected to control directly. Stefan Klein, DL1NKS, has provided this and it can be downloaded at https://github.com/dl1ksv/rshfiq. To operate in a Windows environment, two changes to “hardware_usbserial.py” need to be made using IDLE to edit the file.

The first change is to identify the COM port of the RS-HFIQ transceiver in line 23; in this installation it is COM7.

```
class Hardware(BaseHardware):
    def __init__(self, app, conf):
        BaseHardware.__init__(self, app, conf)

        # Default port settings for RS-HFIQ

        serialport.port = "COM7"
        serialport.baudrate = 57600
        serialport.bytesize = serial.EIGHTBITS #number of bits per bytes
        serialport.parity = serial.PARITY_NONE #set parity check: no parity
        serialport.stopbits = serial.STOPBITS_ONE #number of stop bits
        serialport.timeout = 1            #non-block read
        serialport.rtscts = False
```

The second change is the addition of three lines of Python code after line 88 to improve the stability of frequency changes made to the RS-HFIQ during startup and Winlink frequency scans:

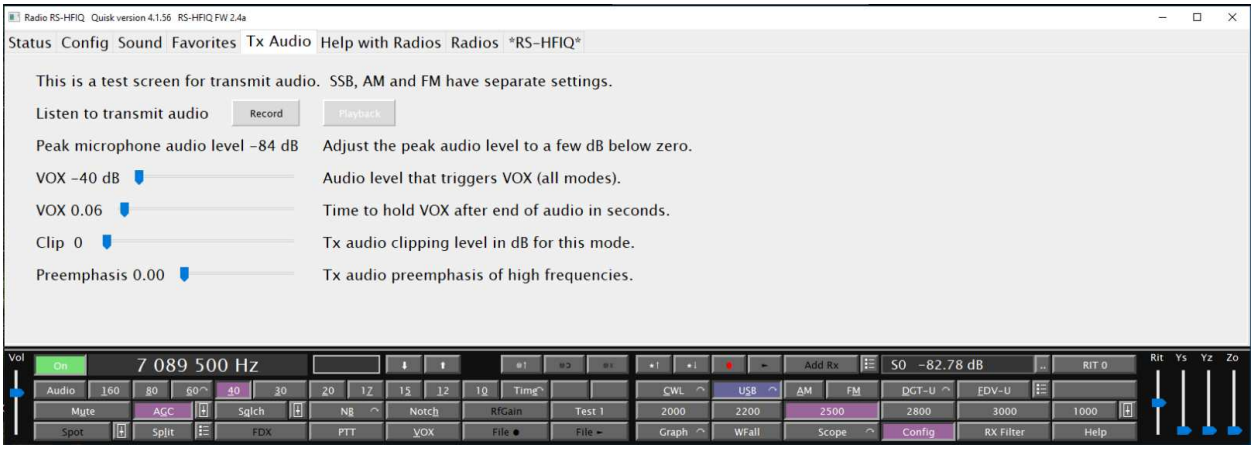
```
    def ChangeFrequency(self, tune, vfo, source='', band='', event=None):
        if self.vfo <> vfo :
            if serialport.isOpen() :
                self.vfo =vfo
                vfo_string = "*F" + str(self.vfo) + "\r"
                if DEBUG == 1:
                    print("Tuning to: ", vfo_string)
                print("Tuning to: ", self.vfo)
                serialport.write(vfo_string)
                time.sleep(1)
                serialport.write(vfo_string)
                serialport.write("*OF3\r")


        return tune, self.vfo
```

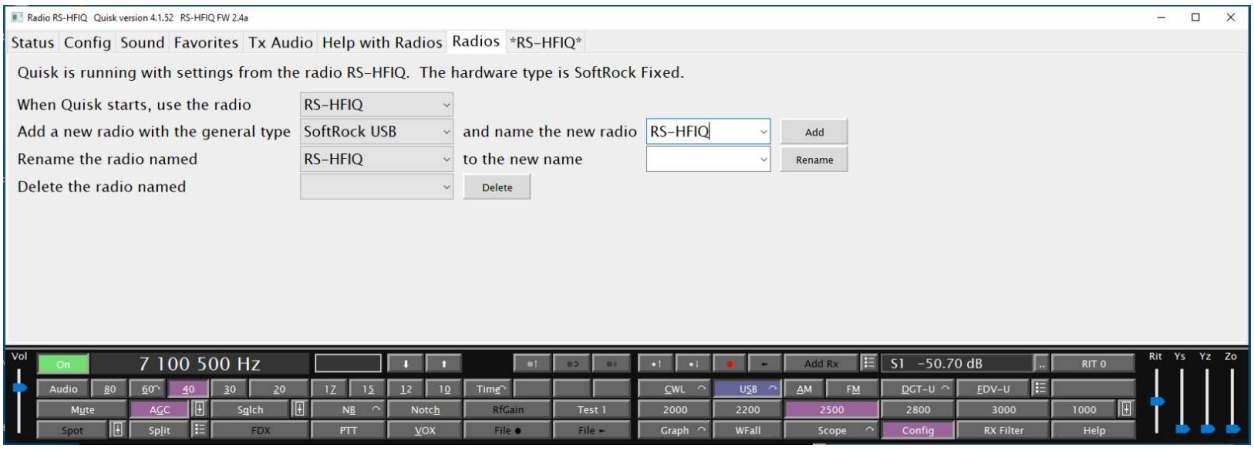
Save or copy the file to C:\Python27\Lib\site-packages\Quisk

7. Configure Quisk

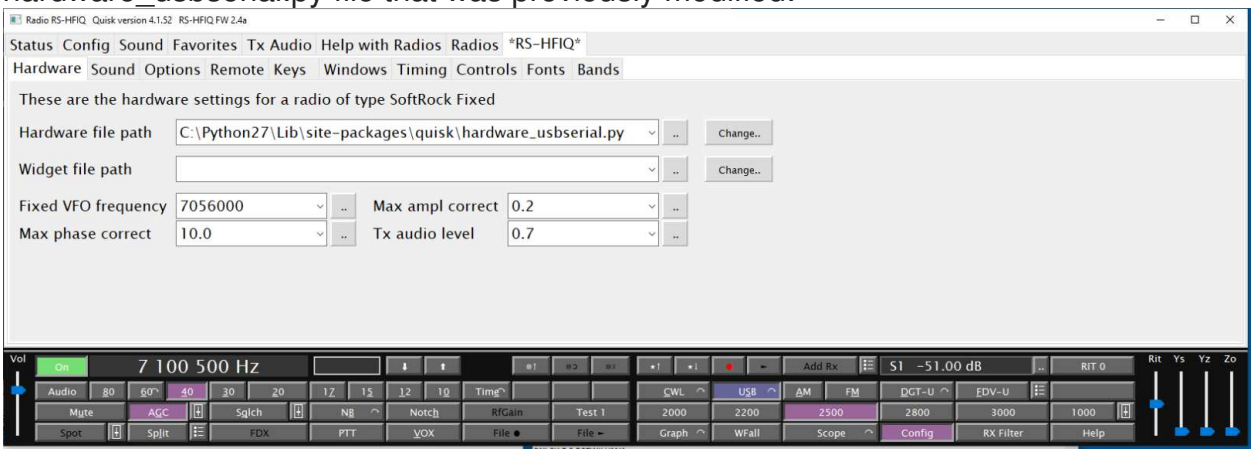
Once Quisk is running, the first thing is to enter is the Config menu and the TX Audio tab. Set VOX to -40, VOX Delay to .06, Clip to 0, and Preemphasis to 0.00.



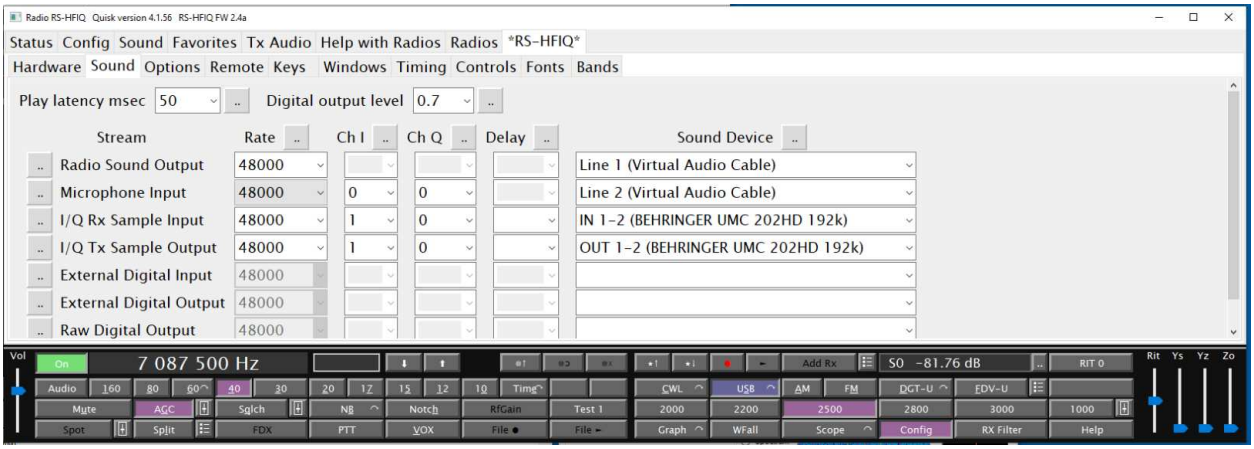
Select the Radios menu and add a new radio called “RS-HFIQ”:



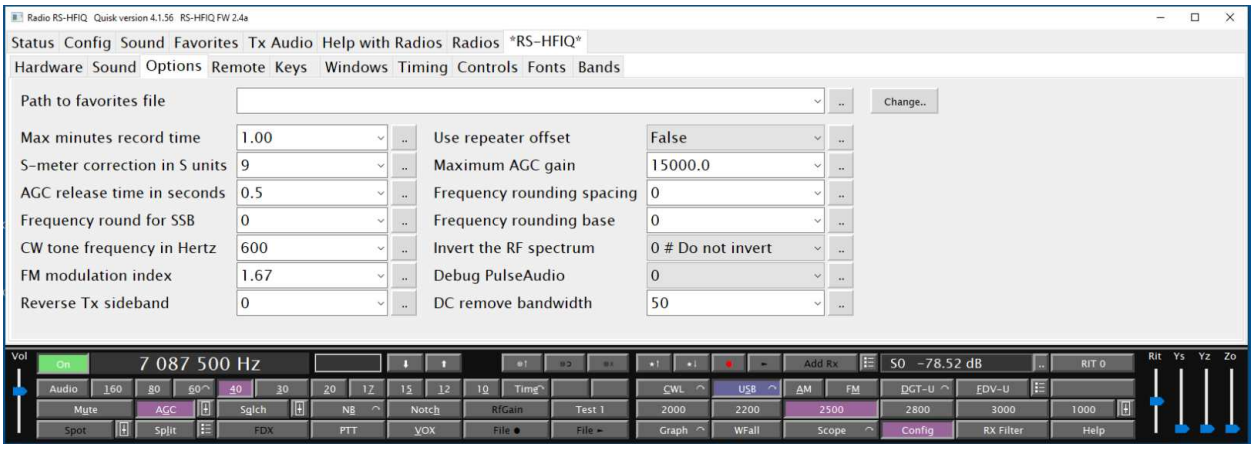
Select the RS-HFIQ Radio and then the Hardware tab under it. In this menu, specify the hardware_usbserial.py file that was previously modified:



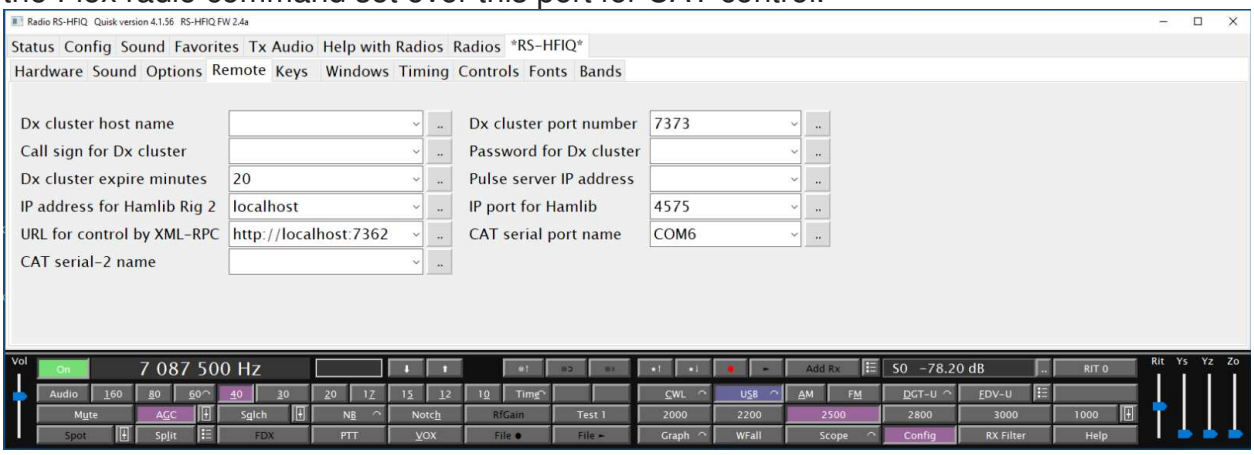
Select the sound tab. Set Play Latency, sample rate, VAC cables, and sound card as shown below. If the receive or transmit sideband is incorrect, this can be changed by the order of the CH I and CH Q settings from I=1,Q=0 to I=0,Q=1.



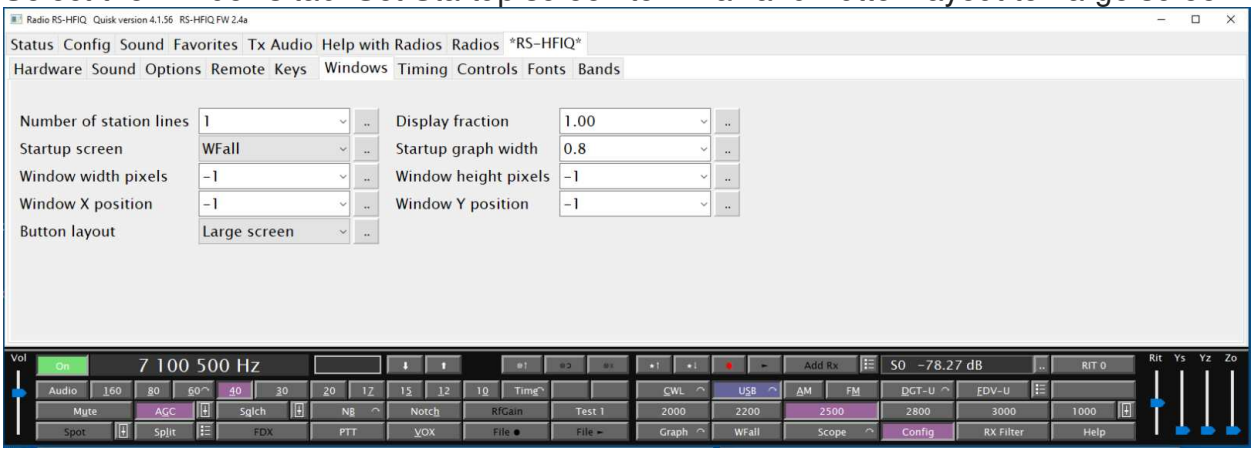
Select the Options tab. Set S-meter correction to 9dB, AGC release time to 0.5 seconds, Maximum AGC Gain to 15000, Invert the RF spectrum to Do not invert, and DC remove bandwidth to 50 Hz.



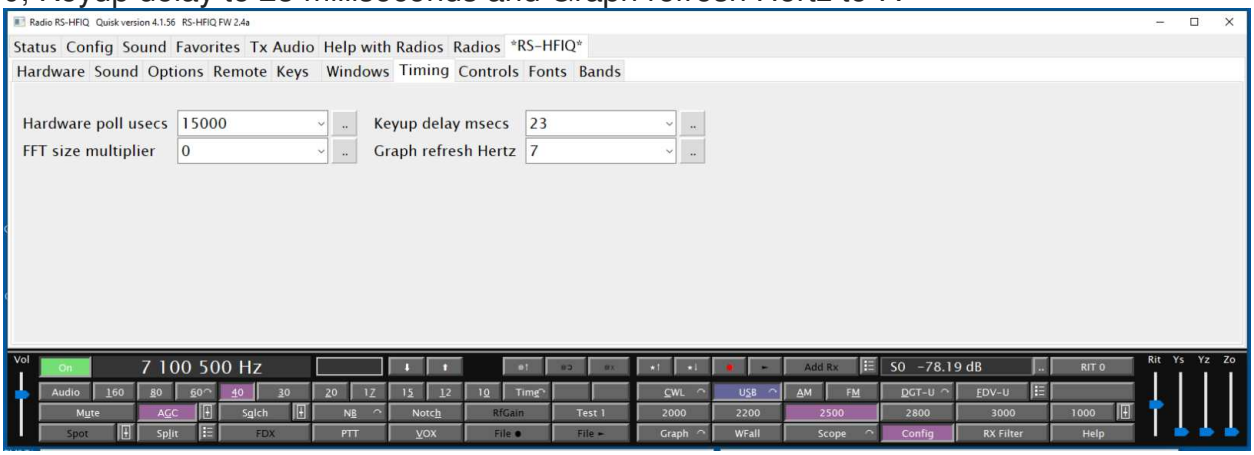
Select the Remote tab. Set the CAT serial port name to COM6 that was one of the serial ports from the pair previously created using Com0Com. Quisk mimics a subset of the Flex radio command set over this port for CAT control.



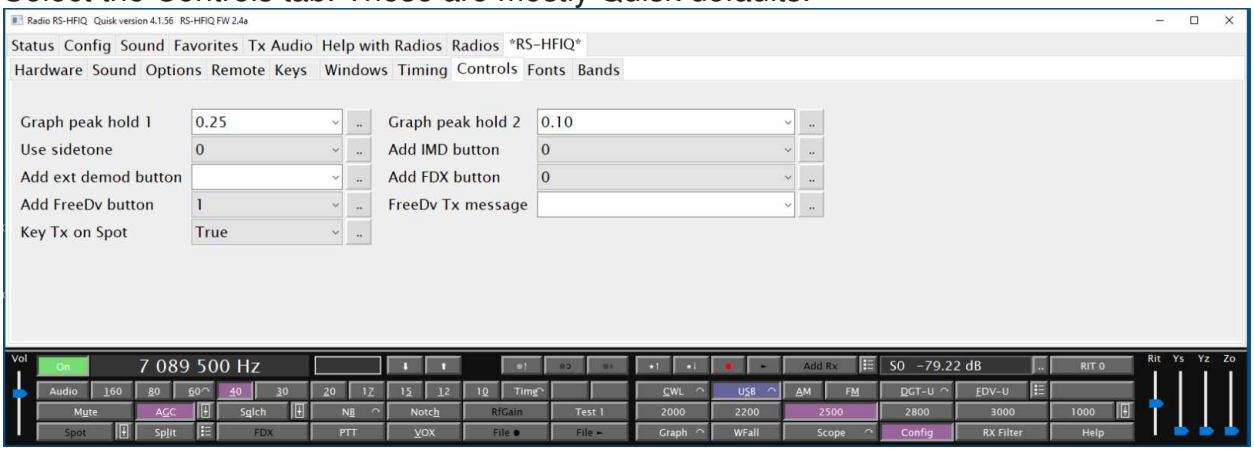
Select the Windows tab. Set Startup screen to WFall and Button layout to Large screen.



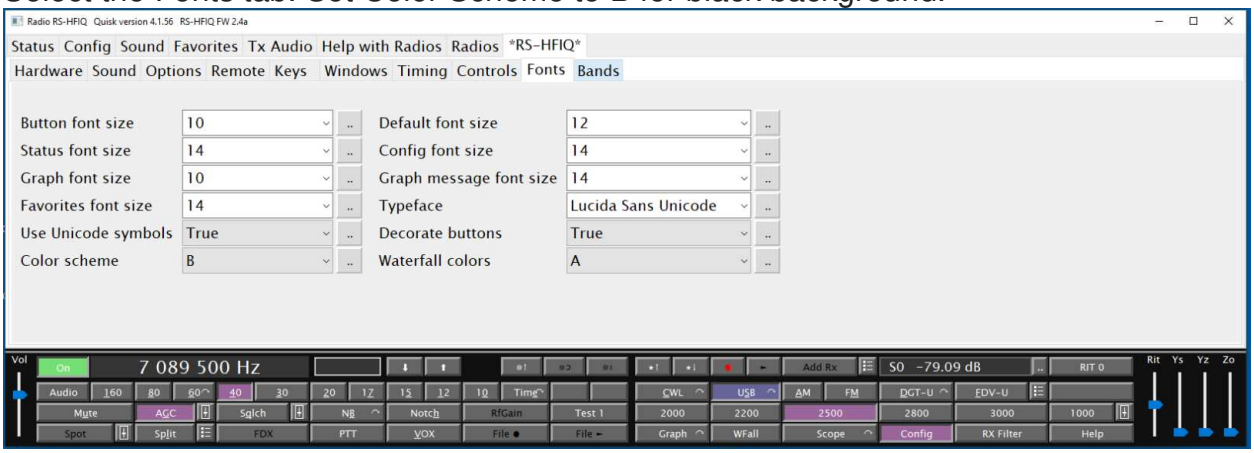
Select the Timing tab. Set hardware poll to 15000 microseconds, FFT size multiplier to 0, Keyup delay to 23 milliseconds and Graph refresh Hertz to 7.



Select the Controls tab. These are mostly Quisk defaults.



Select the Fonts tab. Set Color Scheme to B for black background.

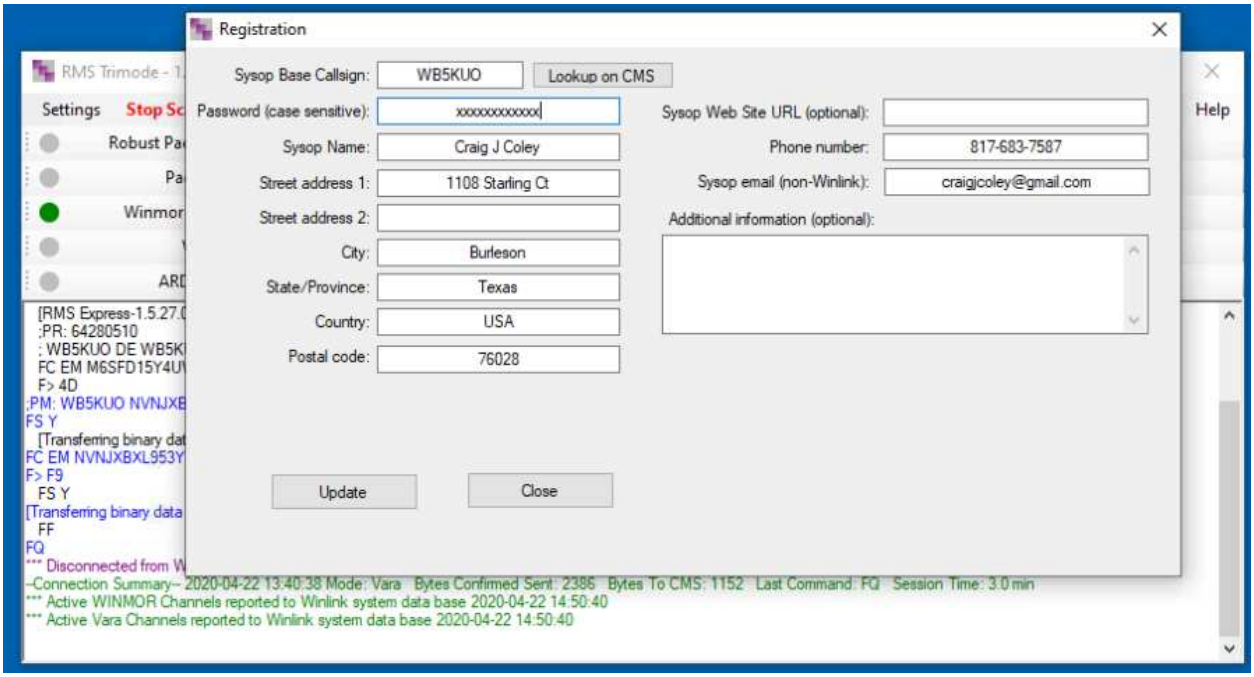


Close Quisk to save the changes and Restart Quisk to apply the changes. These settings are now the default any time Quisk is started.

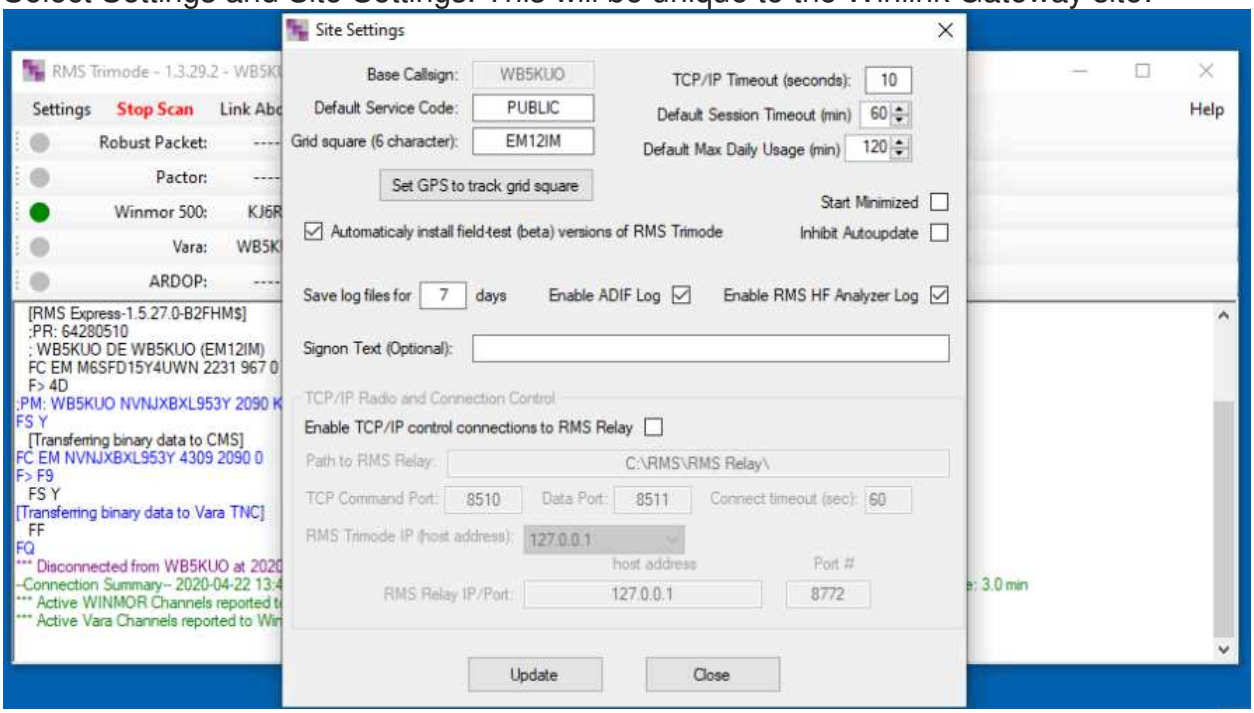
8. Install and Configure the Winlink Sysop Software

Winlink Sysop software can be downloaded at: https://downloads.winlink.org/
The VARA HF Modem can be downloaded at: https://rosmodem.wordpress.com/

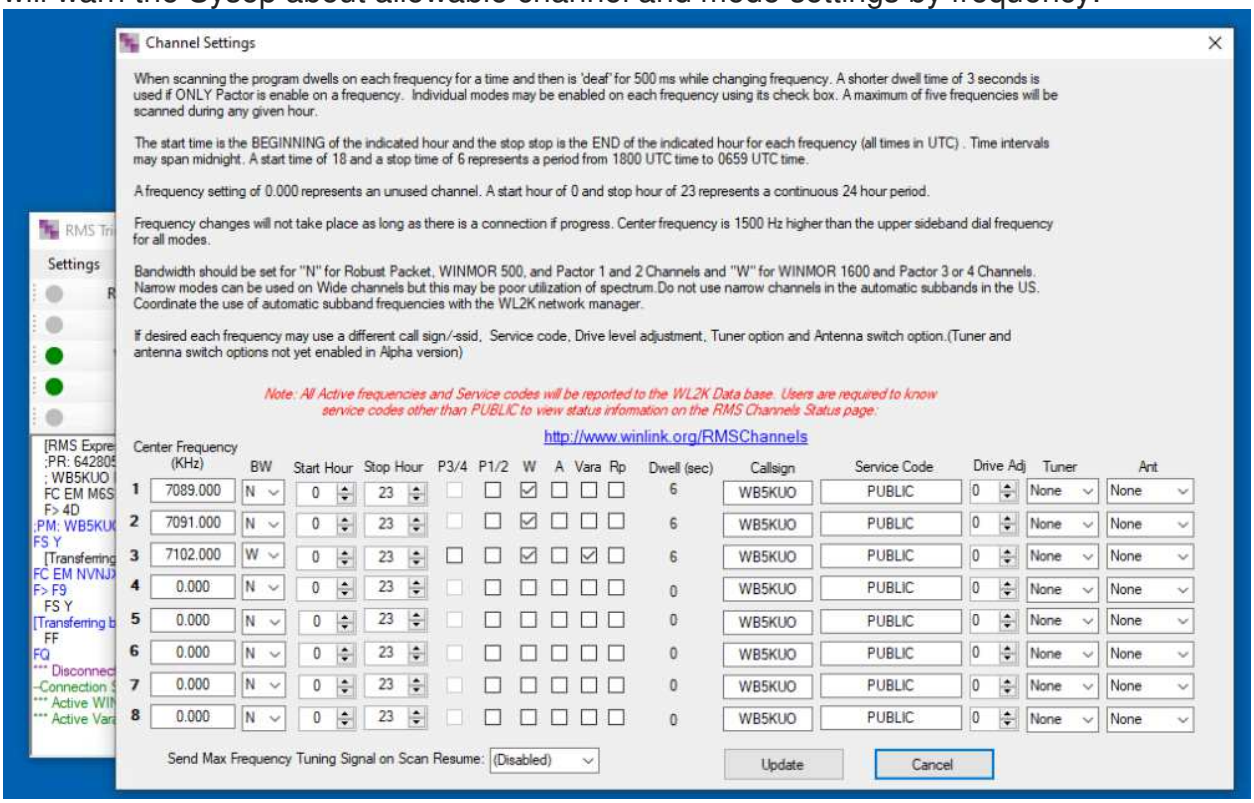
Start RMS Trimode and select Settings and Registration. This will be unique to the Winlink user:



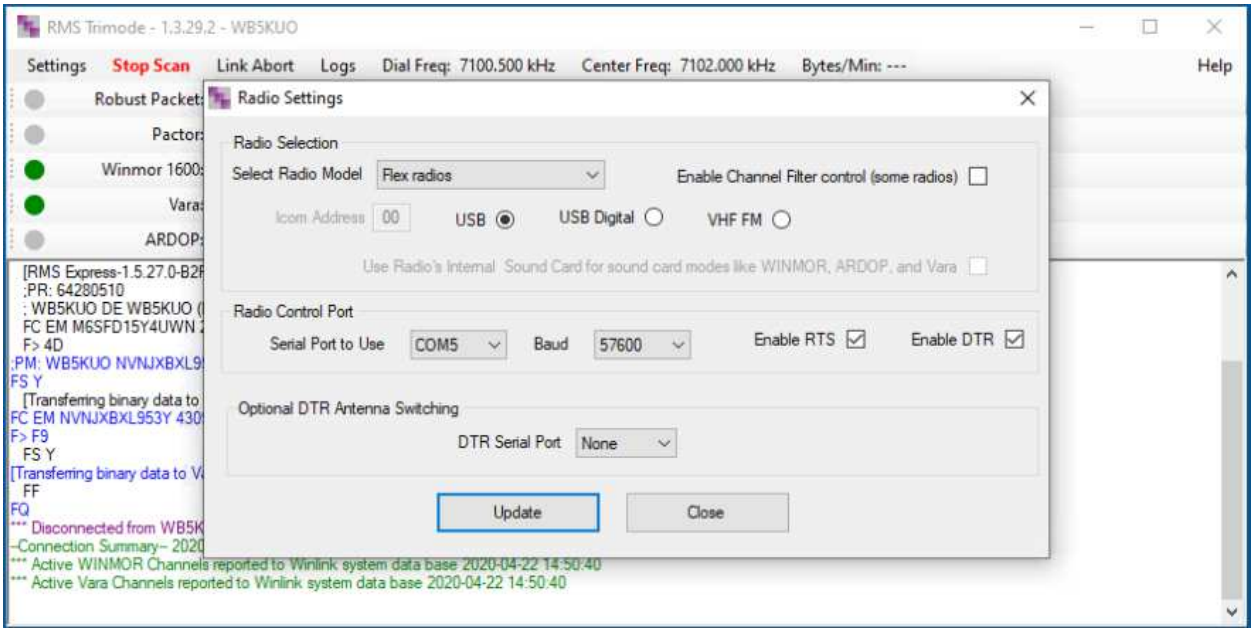
Select Settings and Site Settings. This will be unique to the Winlink Gateway site:



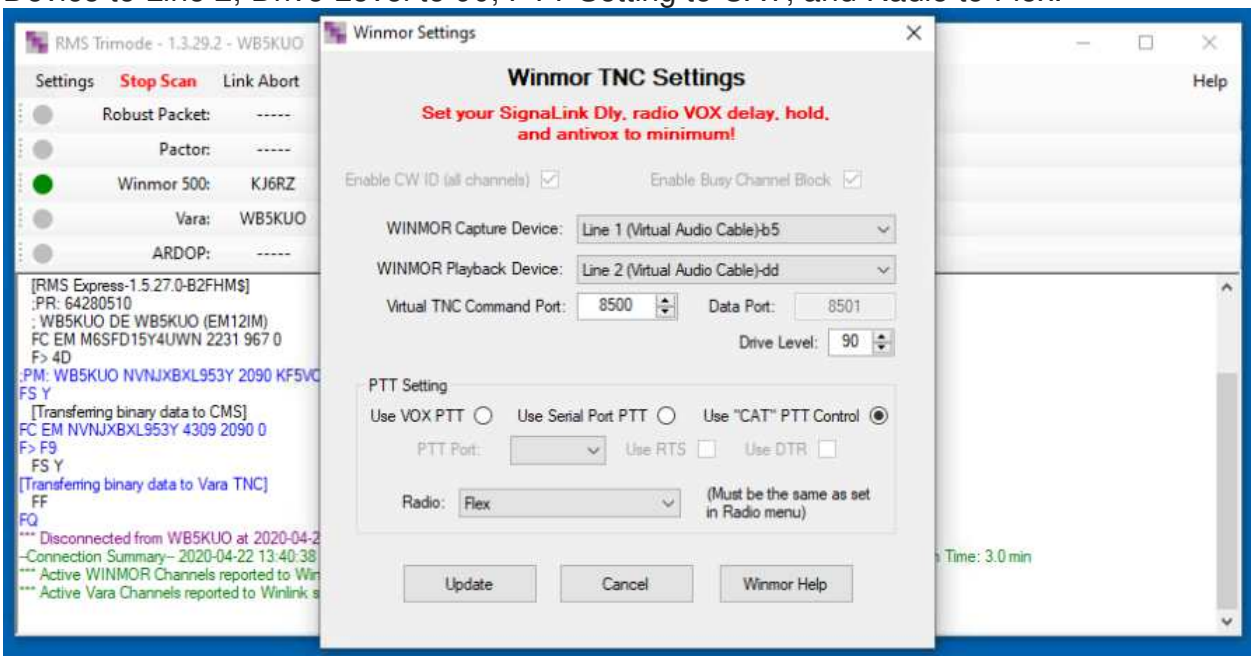
Select Settings and Channel Settings. Operating channels and supported modes are up to the Sysop but should be selected to avoid interference with existing stations. Trimode will warn the Sysop about allowable channel and mode settings by frequency:



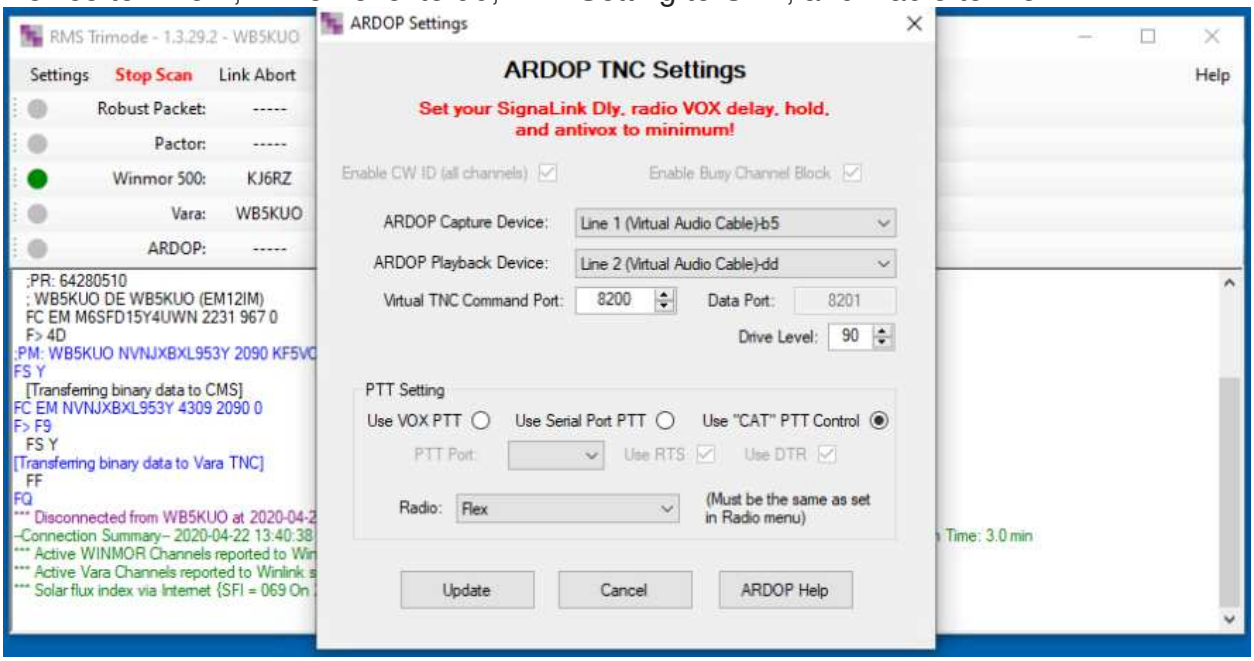
Select Settings and Radio Settings. Set Radio Model to Flex Radios, mode to USB, serial port to COM 5, Baud to 57600, check RTS and DTR, and DTR Serial Port to None.



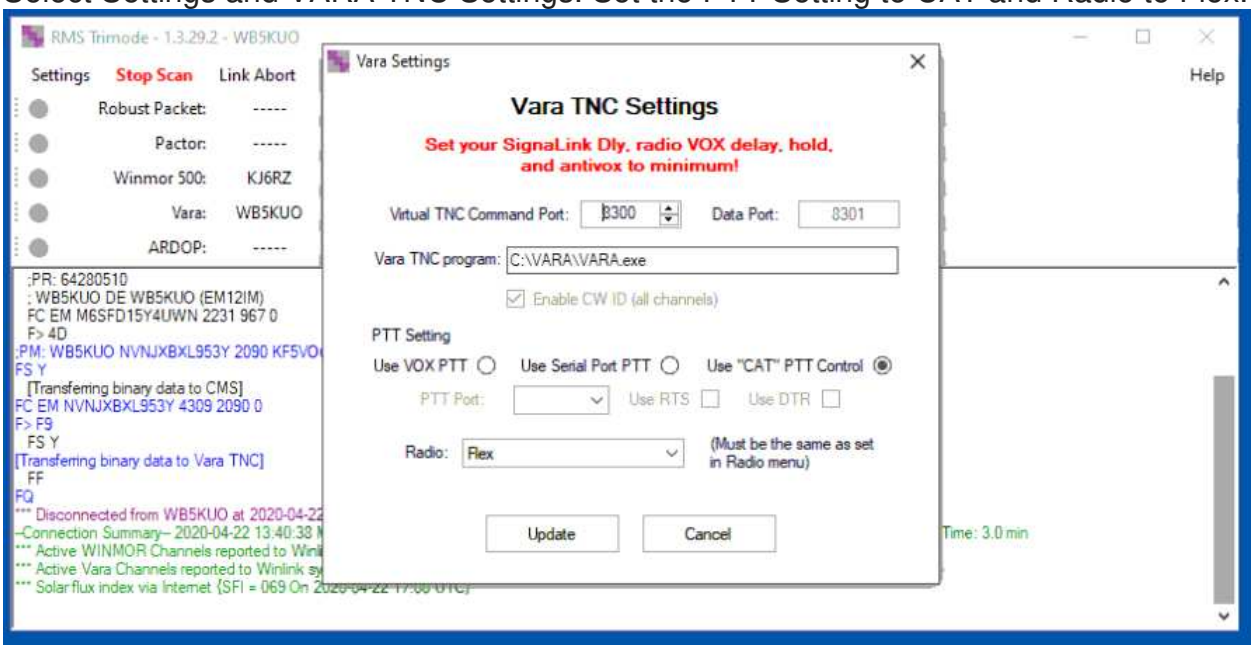
Select Settings and Winmor TNC Settings. Set the Capture Device to Line 1, Playback Device to Line 2, Drive Level to 90, PTT Setting to CAT, and Radio to Flex.



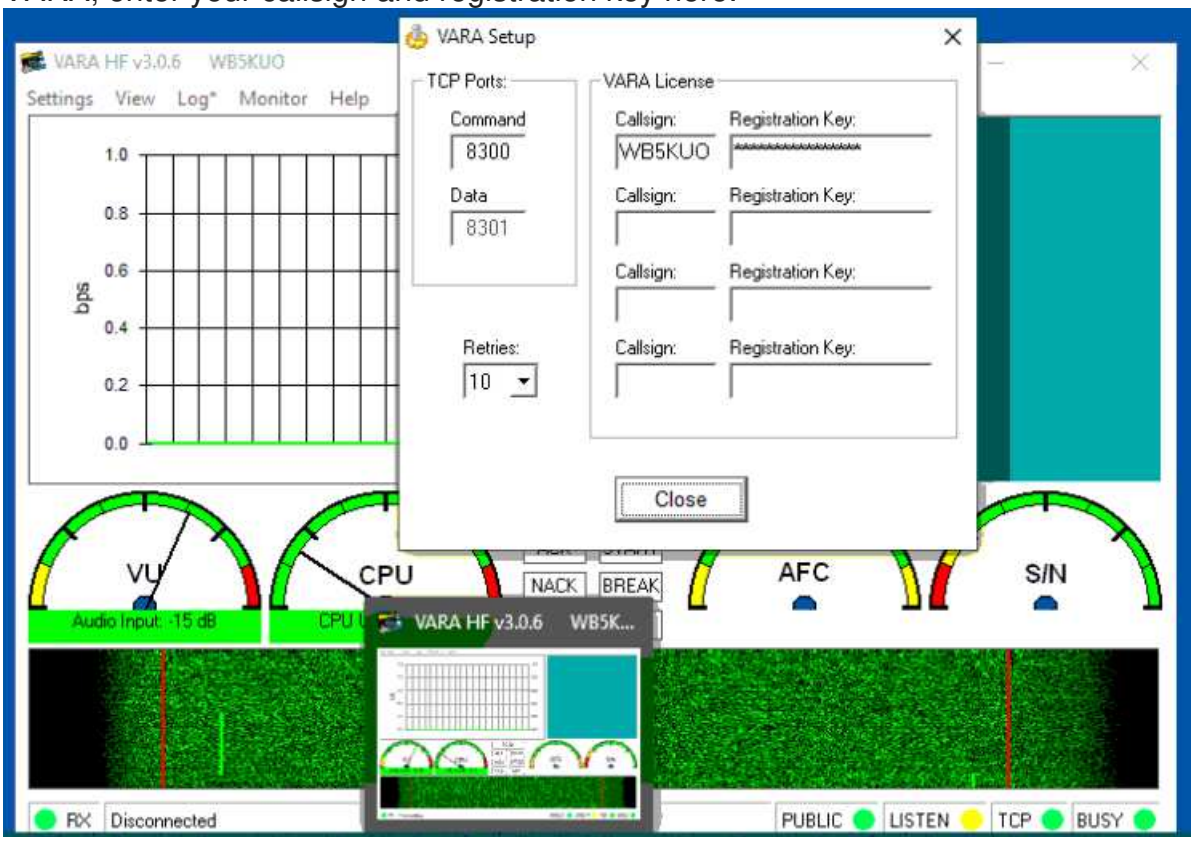
Select Settings and ARDOP TNC Settings. Set the Capture Device to Line 1, Playback Device to Line 2, Drive Level to 90, PTT Setting to CAT, and Radio to Flex.



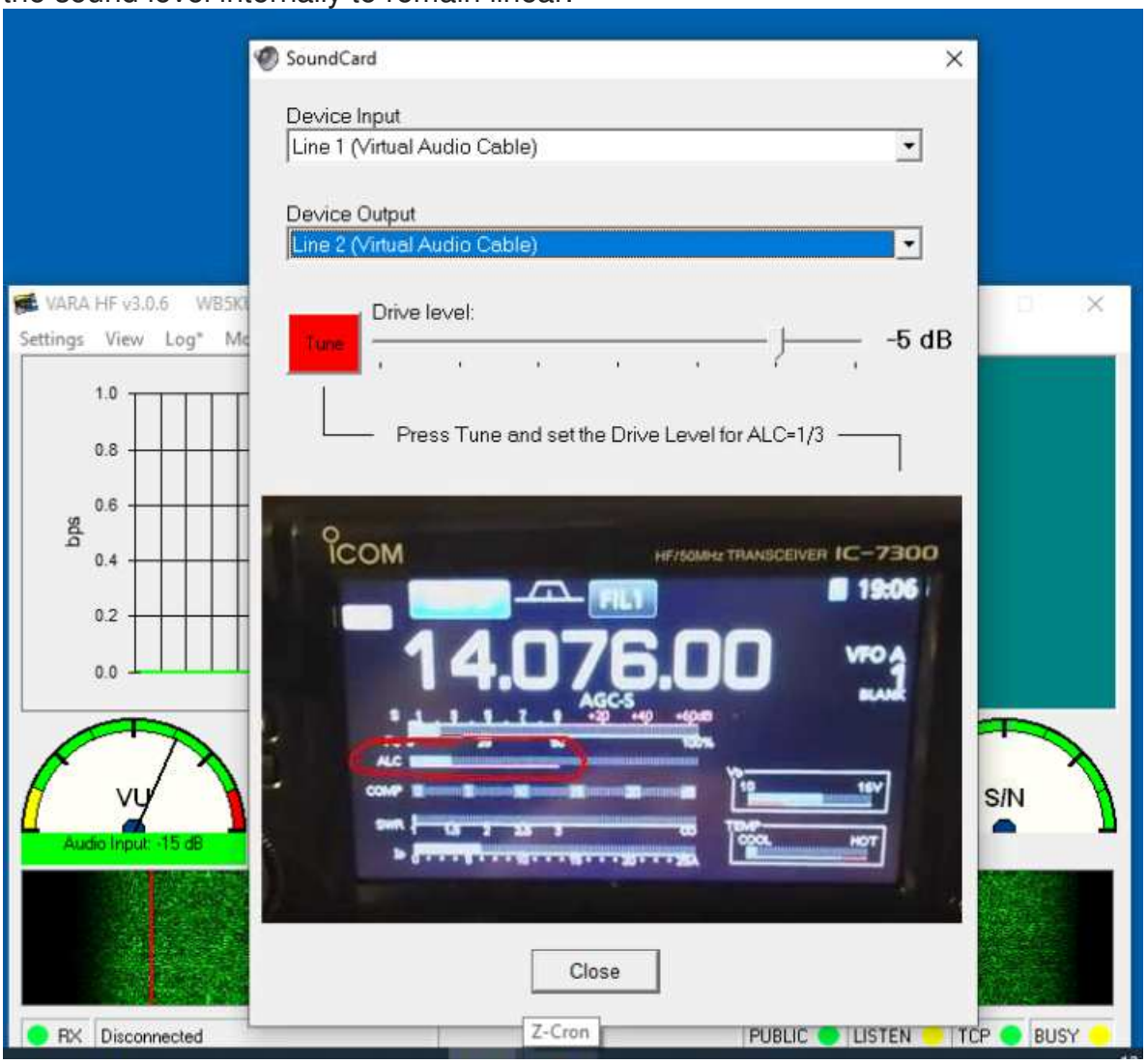
Select Settings and VARA TNC Settings. Set the PTT Setting to CAT and Radio to Flex.



Open the VARA TNC and select Settings and VARA Setup. If you have registered VARA, enter your callsign and registration key here.

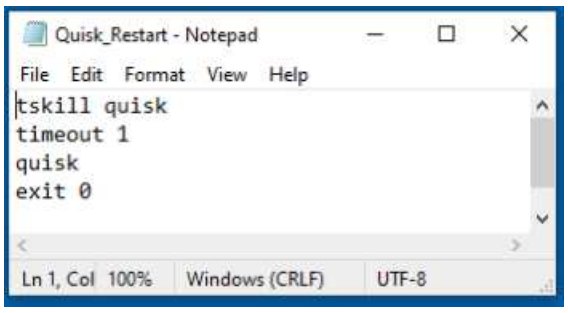


Open the VARA TNC and select Settings and Soundcard. Set the Device Input to Line 1, Device Output to Line 2 and the Drive Level to -5 dB. Quisk will automatically adjust the sound level internally to remain linear.



9. <u>Create the Quisk Restart Batch File</u>

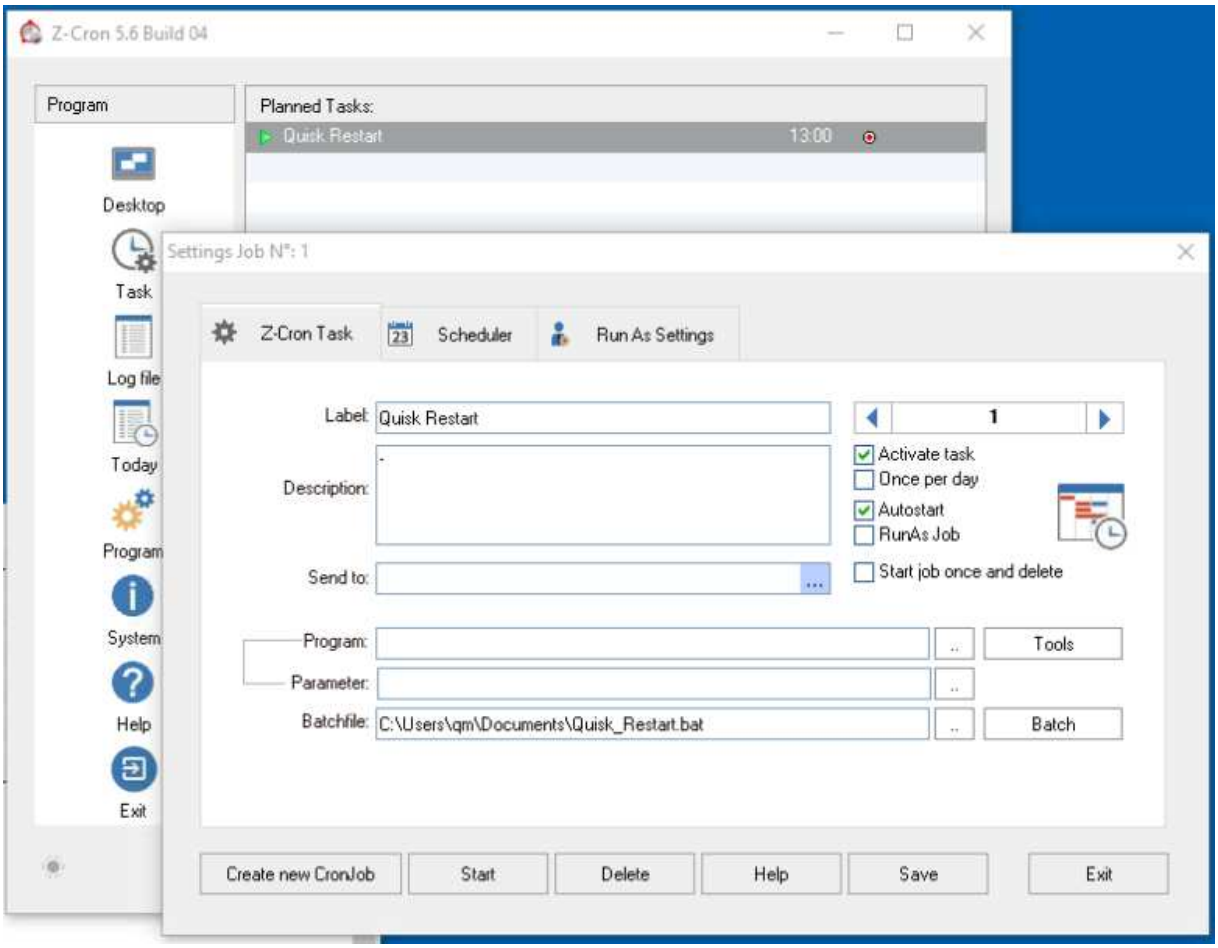
Open notepad and create the following file. When finished, save as Quisk_Restart.bat in the Documents folder. This will give it the user's full privileges.



10. Install Setup Z-CRON

Z-CRON is an easy to use freeware Task Scheduler that can be downloaded at https://www.z-cron.com/download.html

Create a Task Called “Quisk Restart”:



Set the Scheduler to run every hour of every day: