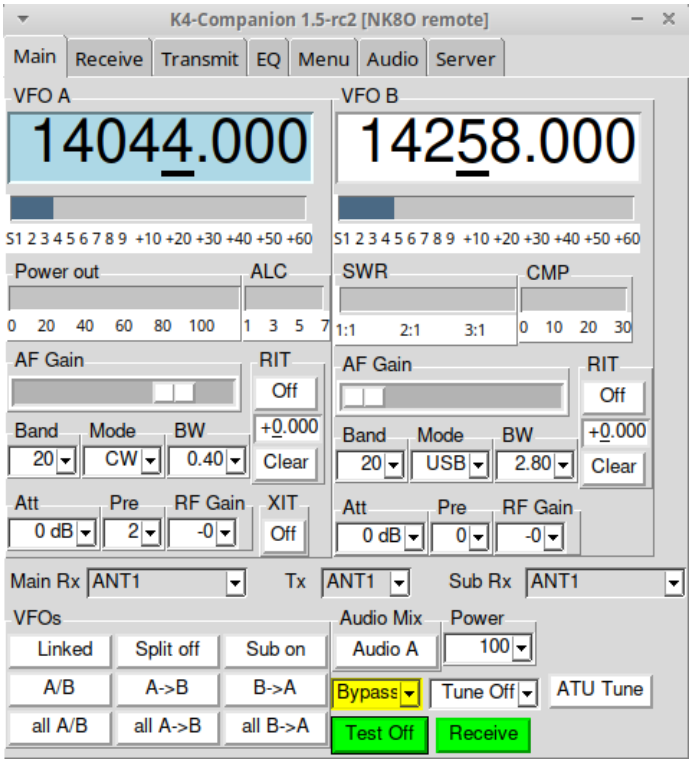
K4 Companion

An application for remote control of Elecraft the K4 series SDR transceiver



Developed by Dale Farnsworth, W7DA

Based on a simple utility originally devised by Charles Powell, NK8O

Revision date: June 30th, 2025
The current version of K4 Companion is v1.5

Table of Contents

Introduction

K4 Companion is an application written in python3 that can remotely control an Elecraft K4 transceiver via TCP/IP. It currently controls the main K4 features and is very usable as is, but new features are being added all the time. K4 Companion is very configurable.

K4 Companion began life as a simple macro-sending program called K4Macro-Python, created by Charles Powell, NK8O. It has now grown far beyond a simple macro-sending program into a full-fledged remote control program for the K4.

Please send problem reports either: by sending an email, by entering an issue on github, or by making a pull request. Problem reports and suggestions are greatly appreciated.

Configuration information is maintained in a separate YAML file named, by default, k4companion.yaml. Custom configurations can be loaded with the *–config* option between the python executable and the desired configuration file.

Dale Farnsworth, W7DA
dale@farnsworth.org

License

Windows Installer & Binaries

Two options are available those who do not wish to create a full installation of Python3 or to examine the open source code. A Windows installer and Windows binary executable files are now available. The automated installer that places all items in the proper directory, creates a Start Menu entry, and it also will create a desktop shortcut if that option is selected. Double click the installer, then select the desired options. The user manual is also provided, found in %PROFILE%/k4companion/. **You may need to disable Windows security briefly to download the installer.** A method to start your K4 remotely is also needed. Please refer to the K4 User Manual or the various forums that address this function.

NOTE: The installer automatically places opus.dll, a mandatory dependency, in the %PROFILE%/k4companion directory. K4 Companion looks for opus.dll in the same directory (folder) as k4companion.exe and k4companion.yaml . The Windows installer does this automatically. K4 Companion has been compiled under Windows 11, and tested on both Windows 11 and Windows 10.

If preferred, manual installation is possible. There are three files that are needed to run under the Windows operating system.

The three files, available on the github download page are:

- k4companion.exe
- k4companion.yaml
- opus.dll

The three k4companion files should be placed in a directory (folder) of your choice. These need to be placed in the same directory (folder). When starting the program, splash screen appears as the program opens (Windows version only).

The configuration file, k4companion.yaml, remains fully editable with your favorite text editor. You may break things if you don't know what you are doing. Re-download the YAML file and start over if something breaks. If you have made significant sequential changes to the YAML file, it is a good idea to save the working copies with some kind of identifier.

A note regarding Windows Defender: The Defender reaction will range from none at all, to identifying K4 Companion as a virus. If identified as a virus, Defender will quarantine the executable and ask if you want to delete it. Instructions regarding bypassing Windows Defender are found at https://answers.microsoft.com/en-us/windows/forum/all/how-to-allow-an-ap-through-windows-defender/caab364c-9e65-448e-8635-98721be847ba

Use of K4 Companion outside your local network requires forwarding of port 9205, a VPN, or creating a SOCKS 5 proxy. Unlike the K4 to K4 connections, K4 Companion uses port **9205.** The K4 to K4 connection is made via port 9204.

Skip to page 9 for information on running K4 Companion.

Installation on Linux

One of the beauties of open source software is that it is typically cross-platform. Using Python3, the application is available on any platform that has Python3 available. It was developed on Linux, but it has now been successfully ported to both Mac OS, and Windows.

Here are the items one needs to take full advantage of K4 Companion and all its features:

1. You must have a way to turn your radio on remotely. This is detailed in the K4 user manual (pages 17 & 18) and requires the momentary closure and grounding of pin 8 to pin 5 on the DB15 ACC connector. Unfortunately the Wake On LAN feature is not available for the K4, at least not at this time. Other details are beyond the scope of this document. Nonetheless, there are many operators who simply leave the radio on at all times.
2. A computer capable of running a full installation of Python is needed. This has been tested with Xubuntu, Debian 12, Chromebook, various iterations of the Dell XPS series, and several Lenovo laptops, running a variety of Linux flavors.
3. Install portaudio (*sudo apt install portaudio19-dev* on Debian based distributions)
4. Install Python3. This is pre-installedd on most Linux distributions. Please refer to readily available documentation for installation on other operating systems.
5. Install 'python-tk' *(sudo apt install python3-tk* on Debian based distributions)
6. Install 'pip3', required for the remaining dependencies *(sudo apt install python3-pip)*
7. Install 'pyyaml' using pip
8. Install 'opuslib' for audio support using pip
9. Install 'numpy', and 'pyaudio' with using (Some of these may be pre-installed) portaudio is required for pyaudio to install correctly
10. Install 'PySocks' with pip. This allows SOCKS 5 proxy connections
11. **For access outside your local network, open port 9205 with port forwarding on your router,** a VPN, or SOCKS 5 proxy. K4 to K4 connections are accomplished via port 9204.

Notes: the typical installation of Python dependencies is with the command *pip install xxxxx* however security consciousness has caused the need for override with a number of operating systems. If your OS complains about installing the extras, use the command *pip install –break-system-packages xxxxx* . It doesn't really break anything. It installs the package in the individual user's profile. I have done many such installs and it has never been a problem. Incidentally the *pip* command can be used to install '.whl' Python files. Chirp, the well known open source HT programming software is provided this way, and using the .whl file is an easy way to keep up with changes and updates. p*ipx* is an alternative to using *pip* and it usually does not require additional arguements.

Although running on Mac OS has been challenging, if you are experimenting the "official" version Python 3 is 3.9, not 3.13. Installing Python 3.13.5 may cause some difficulties with two versions running side by side unless care is taken to call the correct version. Older (Intel) versions may not be qualified for full update to the most current OS release, and necessary dependencies may no longer available. See page 7 for further information.

When installing the pip dependencies you **must** specify *pip3* or create an alias in .bash_profile (macOS) .bashrc (Linux) or .zshrc. **Background colors** do not work under Mac OS. Buttons that have background colors unfortunately do not change on Mac OS. The most recent versions of K4 Companion have been tested on Mac OS and they work quite well on some systems, but not so much on others. The YAML file can be edited to your liking, but it is easy to break things. If it stops working, use a saved copy of the file, or simply download a fresh YAML file and start over. The latest version is available **here**: https://github.com/DaleFarnsworth/K4-Companion/

Full Python Installation on Windows

Installation on Windows follows the same principles as Linux. It is quite similar, once Python3 is installed. **NOTE:** Windows binaries are available. See Page 3.

Here are the basic step:

1. As with Linux, you must have a way to turn your radio on remotely. This is detailed in the K4 user manual (pages 17 & 18) and requires the momentary closure and grounding of pin 8 to pin 5 on the DB15 ACC connector. Unfortunately the Wake On LAN feature is not available for the K4, at least not at this time. Again, there are many operators who simply leave the radio on at all times.
2. A computer capable of running a full installation of Python is needed. Windows 11 is the platform upon which testing has been done. With Windows 10, things may or may not work well.
3. Install Python3 https://www.python.org/downloads
4. Update pip *python -m pip install –upgrade pip*
5. Install numpy *pip install numpy*
6. Install opuslib *pip install opuslib*
7. Install pyaudio *pip install pyaudio*
8. Install socks support *pip install pysocks*
9. Install YAML support *pip install pyyaml*
10. From Contributions on the K4 Companion github site, download opus.dll and move it to the folder where K4 Companion is placed
11. **Forward port 9205 for access from outside your local network,** use a VPN, or configure a SOCKS 5 proxy.

Download k4companion and k4companion.yaml from the github site. Place them in a convenient folder. I suggest using PowerShell to use to open k4companion. It allows the use of the Ampersand, like Linux, to fork a process into the background and gives a little more flexibility. One caveat: if you close PowerShell or your cmd program on a running version of K4 Companion, it will exit the program. You will also need a copy of opus.dll in the same folder as the k4companion script. Opus.dll is on the github site.

The utility has been tested on Windows and works in the same way as it does under Linux. Known quirks include extensions are incorrectly assigned when files are downloaded. The main file may download with the ‘.txt’ extension from the website and it should be either ‘.py’, or left blank. Windows also wants to drop the ‘.yaml’ extension. It is best in both instances to specify the extension when the downloads are saved, and to double-check the file names after downloading. If you have missed any of the dependencies, Python is very good at telling you what is needed.

The latest version is available **here**: https://github.com/DaleFarnsworth/K4-Companion/

MacOS Python Installation

Please read and follow all steps carefully. These instructions are verified on several MacOS machines running MacOS 15.5. It should work on other versions of MacOS as well. The usual disclaimers apply. Install at your own risk. Management, writers, producers, programmers, and all others associates are not responsible if you break your computer. Having said that, there is little to be broken and there are no reports of any problems.

1. Install Xcode from the App Store. After downloading, open Xcode. There is another Download and Install button. Select MacOS, not iPad or other versions as they are not needed. It does not actually download the code again but it does install the necessary code. Accept the license for Xcode. After completely installing Xcode, open a Terminal window (Applications → Utilities → Terminal). Run: *sudo xcode -license* and agree to the license terms. It has been found that MacOS does not always require this step
2. Download and install the latest version of Python 3 from *https://www.python.org/dowloads* As of this writing, the latest is version python 3.13.5 (as of 2025-06-14). Use the “macOS 64-bit universal2 installer”
3. Optional step to use the Bash terminal shell. If you prefer to use zsh, the default Terminal shell, see below. Change the default Terminal shell to Bash. Open a terminal shell (found in Applications → Utilities). Run: *chsh -s /bin/bash* This step is due to Bash being commonly used in many Linux/Unix applications. Again, zsh shell users see comments near the end of this document
4. Create an alias to the current version of Python. The default version of Python3 installed on macOS is 3.9.6. Creating an alias points to the current version 3.13.5. In Terminal, run: *nano .bash_profile* Note the ‘dot’ at the beginning of the filename. Enter the following in .bash_profile: ‘*alias python=/usr/local/bin/python3*’. On a second line, enter: ‘*alias pip=/Library/Frameworks/Python.framework/Versions/3.13/bin/pip3*’. Note the actual commands do not use the single quotes ‘’. Issue the command *source .bash_profile* <enter> to activate the changes. It is recommended to completely close Terminal (from the menu bar) and re-open, especially of the new ‘alias’ entries are not recognized. This will re-read the configurations and system file locations. Zsh shell users see below for instructions.
5. Next, type: *pip install numpy* <enter>
6. Opus codec is next dependency. This is a download, build, install process. It is not at all complicated. Go to *https://opus-codec.org/downloads/* and download the file opus-1.5.2.tar.gz This will likely go to the Downloads folder on your Mac. In Terminal, type *tar zxvf opus-1.5.2.tar.gz* <enter>. This will unpack the Opus library build files into a directory opus-1.5.2. In Terminal, type the following commands: *cd opus-1.5.2* <enter> then *./configure* <enter> (and note the ‘dot’ before the ‘slash’). Next type *make* <enter> and finally *sudo make install* <enter>. You will be asked for your
7. password at this point, and the files will install into their proper places.
8. Next in Terminal, type *pip install opuslib* <enter>
9. Another build process comes follows, but it is done a little differently. In Terminal, type: git clone https://github.com/PortAudio/portaudio This sounds daunting, but it will do its own thing and copy the files you need. Type the following four commands in Terminal

as shown: *cd portaudio ; ./configure ; make ; sudo make install* <enter>. These commands can be executed sequentially, as shown, using the semicolon ';', or individually. If the process completes normally, you will be asked for your password to complete the installation. If any of these steps fails, back up and re-issue the command. If everything has been done in order and executed correctly, the library should install.

10. There is one more file that must be placed in the system. While still in the *portaudio* directory, type: *cd include ; sudo cp pa_mac_core.h /usr/local/include/* <enter>. This is a necessary file for the K4 Companion's libraries, but it is not installed by default
11. Next, install PyAudio, PySocks, and PyYAML: *pip install pyaudio* <enter>, PySocks, *pip install pysocks* <enter>, PyYaml *pip install pyyaml* <enter>
12. Install MacPorts https://guide.macports.org/chunked/installing.macports.html . Download theappropriate version and click on the package to install.You should once again completely exit Terminal, then restart to access '*port*'
13. Install pkgconfig with Terminal, using the command *sudo port install pkgconfig* <enter> Allow the install to add the needed dependencies. Install cmake: *sudo port install cmake* <enter> Allow all dependencies to be installed '*Y'*. Install cairo: *sudo port install cairo* <enter>. Allow all dependencies to be installed *'Y'.*
14. Install Python pkgconfig *pip install pkgconfig* <enter>
15. Install pyobjc *pip install pyobjc* <enter>

If all has gone well, download k4companion and k4companion.yaml from W7DA's repository on github at *https://github.com/DaleFarnsworth/K4-Companion* Warning: if downloading from the web pages, the file k4companion will download as k4companion.txt. This file should be renamed as k4companion with the file extension deleted. To run the application directly, go to Terminal. Navigate to wherever you have placed the K4 Companion files. Type: *chmod 755 k4companion* <enter>. It is not necessary to modify or change permissions on the YAML file. With this installation of K4 Companion, the program must be started from the command line. Again, navigate to the folder (directory) containing the two files, k4companion and k4companion.yaml To start the program, use the following command *./k4companion &* <enter>. The ampersand '*&*' forks the program into the background. Then the terminal window can be closed. You should be ready to put the radio's information in the Server tab at this time. Since MacOS has extra security, the first time opening the program you must "Allow" K4 Companion through macOS security. Because it couldn't connect immediately, it is necessary to press "Connect" after allowing the program. If you are familiar with shell scripts, there is a way to create a desktop or other shortcut that can be called from Finder. When the script is run for the first time, it will advise of any missing dependencies or skipped steps.

Note for zsh shell users:
To create an alias with zsh, open Terminal. You should be in your home directory (type cd <enter> if you have navigated elsewhere. Type: nano .zsh The alias will take the following forms: *alias python="/usr/local/bin/python3"* and *pip="/Library/Frameworks/Python.framework/Versions/3.13/bin/pip3"* With zsh the double quotes are needed. It is best to close Terminal and reopen so paths can be properly established each time changes are made and before MacPorts is installed.

Note to MacOS users: RUMlogNG is a free (and excellent!) logging utility for MacOS, and the developer, Thomas – DL2RUM, has implemented a few features that are not yet in K4 Companion. Most notably, the panadapter and mini-pan. These programs run perfectly side-by-side. Connect RUMlogNG via TCP/IP by using the radio's address and port 9200 with the radio selection as Elecraft K4 – ctrl. RUMlogNG is available free from the App Store. RUMlogNG configuration page.

Settings for running side-by-side with K4 Companion

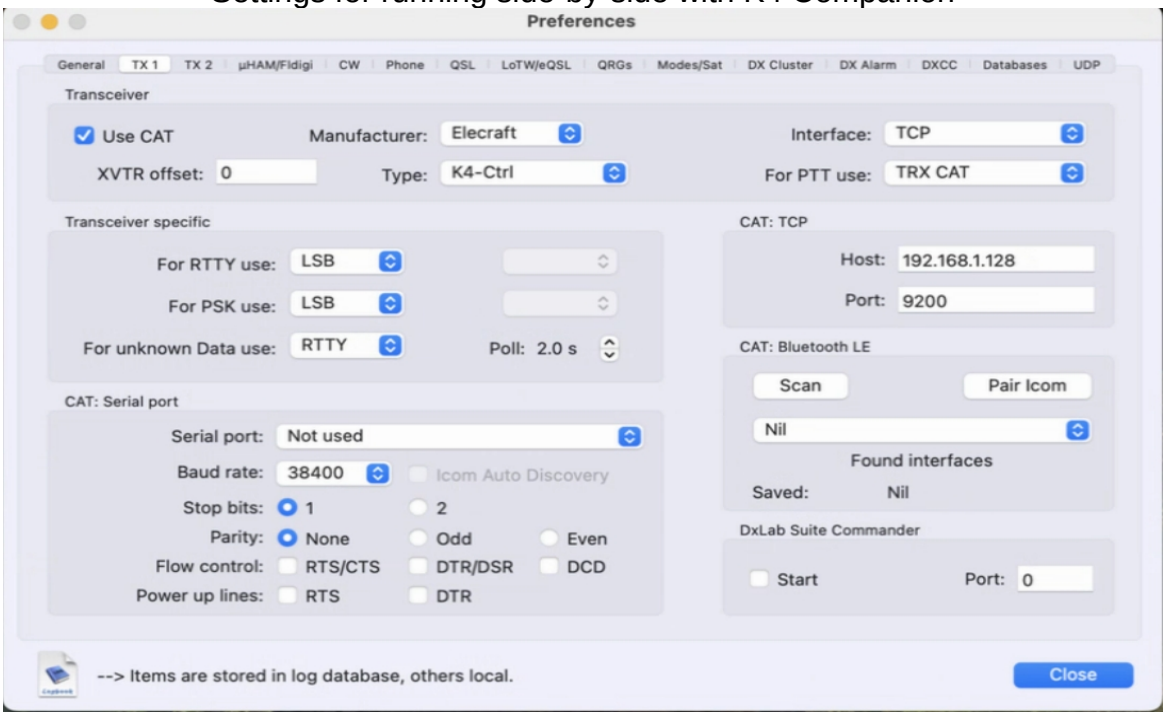


Image used with permission, Thomas Linder, DL2RUM

Work on a MacOS installer in the form of a Python 'wheel' is ongoing.

Running K4 Companion

There are several ways to run K4 Companion, depending on the operating system. To execute the script directly, it requires the file to marked as executable. Under Linux and Mac OS, this can be done immediately after downloading the script. In a terminal window and using the command line, issue the command *chmod 755 k4companion .* It is not necessary to make any modifications or change permissions of the YAML file. The YAML file should be in the same directory or folder as the *k4companion* executable. Otherwise, the command line becomes complicated. No modifications of the Windows executable files is needed.

Methods for starting K4 Companion:

- Assuming the file has been made executable, from a Linux terminal window issue the command *./k4companion & (enter)* from the directory or folder where the script, as well as the configuration file are located. This will work with the "stock" configuration YAML. To use a custom configuration, use *./k4companion --config my_k4companionfile.yaml & (enter)*
- The alternative method, which requires no modifications nor marking files as executable is as follows: *python k4companion & (enter)*. The same approach can be used here for a custom YAML file
- To start the Windows binary navigate to the file in the chosen folder, then either click on k4companion.exe or highlight and press 'e*nter*'
- If the Windows installer has been employed, K4 Companion can be run from the Start Menu or a Desktop icon if this option was selected at the time of installation
- In all cases, access from outside your local network requires forwarding of port 9205

Linux users: If you create a shortcut or desktop launcher K4 Companion, it must contain the full path to the executable in order to find the configuration file.

Windows users: Versions of PowerShell > 6.0 allow the use of the ampersand (*&)* to fork processes into the background. Windows requires calling *python k4companion & (enter)* and does not recognize '*python3'.* If the installer was employed, K4 Companion will be on the Start Menu, and if selected, there will also be a desktop shortcut. A taskbar shortcut is possible and the icon can be assigned to it as well.

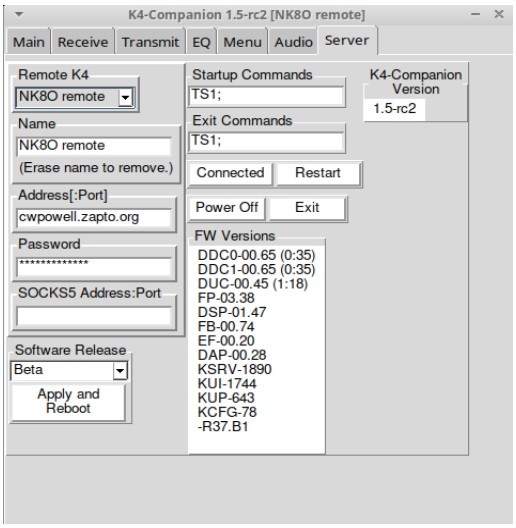
```
cwpowell@newXPS:~/bin$ ./k4companion &
```

```
cwpowell@newXPS:~/bin$ python3 k4companion &
```

Remote operation of K4 Companion can be achieved by one of several methods. Port forwarding through most home routers is possible. The default port is **9205.** An SSH 'tunnel' can be set, or a SOCK 5 proxy can be configured, or a VPN connection established. A service such as "Tailscale" may also be used, but typically it will require setting a SOCKS 5 proxy through a Linux (or mac) computer on your internal network. Tailscale overcomes some of the issues with services such as Starlink or cell network connections by eliminating the need for access to the router or a lack of port forwarding capabilities, and it is free for personal use.

Simultaneous connection K4 to K4 on port 9204 and K4 Companion to K4 on port 9205 are not possible.

Configuration: Server tab



Server configuration is done through the 'Server' tab. Configuration parameters **must** be entered but K4 Companion will save these items and open after going back to the Main tab. The required items are:

- Specifying the 'Name' for the profile. Each new profile defaults to 'K4', but this can be changed when the profile is created or specified. Up to 10 profiles are possible with K4 Companion. In the 'Remote K4' pulldown after a profile is created, select the name for the connection. If only a single connection is created, K4 Companion will connect automatically. Otherwise the program opens to this page so the profile can be selected
- The address of the K4 to be controlled. This can either be the serial number of the radio in the form K4-SNxxxxx.local, or it can be the actual IP address of the K4. There are reports that using the actual IP address makes the connection more responsive, and the serial number setting may not work under Windows. Your experience may be different. For remote operation this will be either your external IP address, or a name you have created in a service such as DynDNS or other such services. The default is **port 9205.** Specifying the connection port is not necessary unless you have re-directed to a different port through your router
- The password to access your K4. For all remote control this **must** be set on the radio. The default password will NOT work. In addition, the number of connections to your K4 must also be specified. This can be set to one, and up to four connections allowed. No connections will be permitted if the parameter is set to zero or left at default.
- SOCKS 5 proxy is fully supported, although this is optional. The SOCKS 5 proxy address is entered in the 'SOCKS 5 Address:Port'. The default port is 1080 and need not be specified unless a different port is used. When operating from a single remote address, it is possible to "open" your router to accept only that address, making a proxy connection unnecessary. Nonetheless, a proxy connection offers greater flexibility in connection options
- Once information is entered, K4 Companion will attempt to connect. If the information is entered correctly and the server is running, a connection will be established. **Hint:** enter password information first, then the server address

- The Startup and Exit commands put the radio in Test Mode when K4 Companion opens and return it to Test Mode for safety when the program exits. These can be changed at your own risk!

Firmware upgrades or downgrades, and beta releases can be installed remotely. The column on the right shows what is actually installed and not available releases. Please check on the Elecraft site for information on the various releases and feature, or other information.

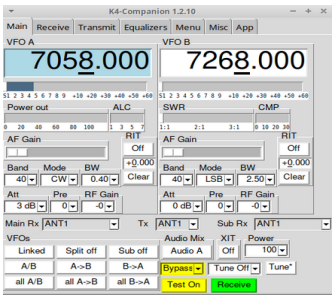
On the right hand side, the connection status is shown. '**Power Off**' turns off the K4. '**Exit**' will close K4 Companion without shutting down the radio. '**Restart**' can be used if configuration is changed or if there is a brief loss of connectivity, i.e., the Internet connection drops briefly or network congestion causes a problem. '**Connected'** reflects the connection status.

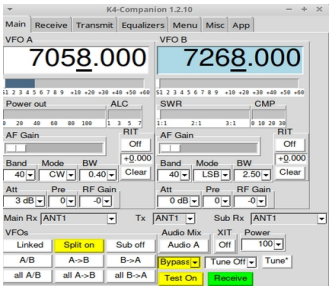
The version of K4 Companion is displayed at the upper right.

Operation and Controls

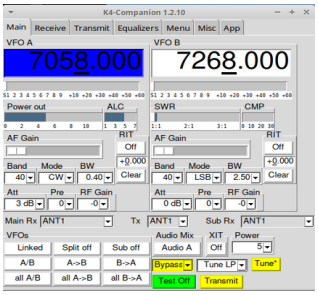
K4 Companion is logically organized with largely self-explanatory labels and functions. Each VFO can be linked, operated independently, swapped, placed into Split mode, Sub-receiver mode, or both simultaneously. A diversity receive selection is also available. Various other functions are controlled in the main tab, including preamps, attenuators, RF gain, ATU functions, audio controls, keyer speed, plus sidetone volume as well as power output, and the rig can be shut down from the K4 Companion.

The active transmit window is shown in pastel pink, so it is possible to see at a glance which VFO will transmit when the radio is keyed. This prevents confusion when using Split or Sub modes.





In the image on the left, VFO A is active for transmitting, and on the right, VFO B is ready for transmit. Note below that the K4 is actually transmitting at 5 watts, the ALC is normal at 5, and the SWR is slightly greater than 1:1. The details available at a glance are the frequency of each VFO, the S-meter reading for each VFO (if both are active), power out either 0-10 watts or 0-110 watts, SWR, and for phone operation, compression. When transmitting, the active transmit VFO show dark blue



The scroll rate for each VFO can be set independently from 1 Hz to 10 kHz. The rate is determined by clicking on a segment of the VFO, then scrolling with touchpad, mouse, or arrow keys.

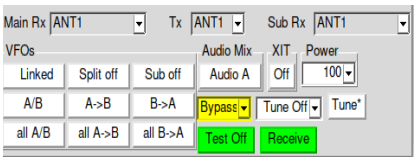
The VFO boxes also support **direct entry** Increment can be changed with the keyboard arrows ← → or mouse/touchpad. The active scroll rate increment is underlined. The Up ↕ Down arrows on the keyboard can be used if there is a single click in the VFO box. Scrolling always occurs at the underlined increment, and this can be changed as noted above.

Pull-down menus shown in the images above are available for each VFO to set the following parameters:

- Band selection, 160 through 6 meters
- Mode selection, including all available modes for the K4 (does not include digital sub-modes)
- Bandwidth selection from 50 Hz to 5 kHz (for AM reception)
- RF attenuation in 3 dB steps up to -21 dB
- Preamp selection − none, 1, 2, or 3 (depending on the band)
- RF gain, in -10 dB increments down to -60 dB. Note: Elecraft recommends use of Attenuation rather than RF gain. Attenuation reduces the signal to the ADC. This is different from the K3

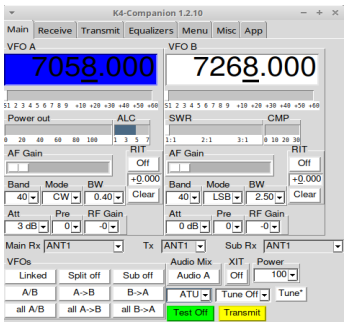
Each of these selections will “scroll” with the mouse or touchpad, in the same manner as the VFO controls.

The following controls are fairly self-explanatory, at least once you are familiar with K4 Companion.



When the application opens, it sets the K4 to Test Mode, as shown above. This button lights up yellow when engaged but turns green when off. Test Mode on startup prevents unintended transmission when setting up. **This can be disabled in the apps tab**, but it can be a useful feature. For safety, K4 Companion returns the radio to Test Mode on exit. Using Test Mode, it is also possible to check **ALC** and **compression** without actually transmitting.

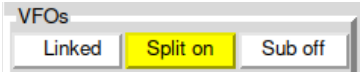
Tx/Rx button at the lower right of the app has two features. If selected, the button warns that the K4 is in transmit mode, and at it turns yellow. Note that pressing this button does not actually put out any power. It is the equivalent of using a foot switch or pressing the PTT on the mic without actually speaking. One could use it to suppress QSK in CW mode but there is very little utility in that. QSK parameters are adjustable and the results are automatic. The button also turns yellow when sending CW as the transmitter is keyed.



Audio output selection is labeled “Audio Mix”. Available selection are A, B, or A+B. If other selections are specified via macros, these will be reflected in the pull-down, but they are not directly selectable from K4 Companion. XIT is selectable with the next button in this row and controlled in the RIT digit box. The box and the button turn blue for XIT and green for RIT. Both RIT and XIT frequency delta can be selected directly or by scrolling with a mouse, touchpad, or keyboard arrows. Transmit power can be selected either with pull-down presets or by direct entry.

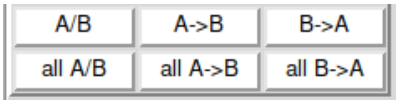
In the next row of buttons the options are:

- Linked VFOs or Unlinked. Elecraft calls this “Band Independence”
- Split on, Split off. This button lights up yellow as a secondary warning for the change of transmit VFO, and VFO B turns light blue to warn that it is now active for transmit



- Sub activates the K4 sub-receiver. This can be used either with Split mode to move the transmit VFO to B, or without. The main and sub-receivers have separate volume slider controls, located below the VFOs and meters.

The middle row of the VFO section is self-explanatory.

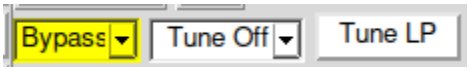


- Swap VFOs A/B (may be activated repeatedly maintain different modes, for example)
- VFO A to B
- VFO B to A

The finals row “All” swaps or copies all parameters, equivalent to a “double tap” of the buttons on the K4.

Antenna selection is now available, since version 1.2.7. These combinations mirror the selections available on the K4.

ATU controls function via pull-down:



- Auto = ATU active, or Bypass
- Tune = normal ATU function = single tap on the K4
- TUNE = extended tune = double tap on the K4.

The “Tune” pull-down selects two states:

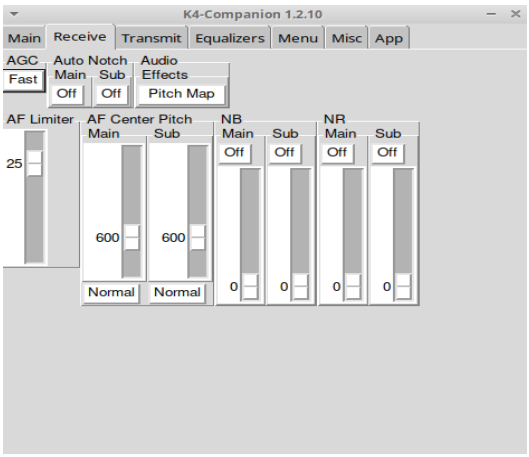
- Tune produces a carrier at the full (selected) power carrier
- Tune LP produces a low power carrier at 5 watts

The remaining button shows the “Tune” state that is active.

To EXIT K4 Companion, use the window ‘X’ at the top of the frame. Location will depend on your operating system. For Windows and Linux this is typically found at the upper right. On MacOS, it is usually on the left.

This completes the tour of the main tab of K4 Companion.

Receive Tab

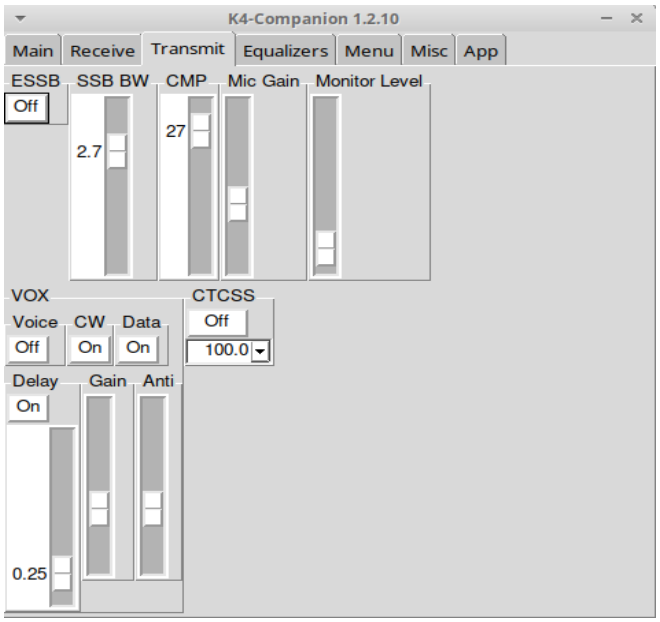


Controls for AGC, Audio Peaking filter, Auto Notch (SSB) and Audio Effects are in the top row. The function of each is explained in the K4 User Manual. Options for Audio effects include

- Simulated Stereo – introduces a slight delay in L − R audio to reduce listening fatigue
- Pitch map − moves audio from left to right as the deviation from the center frequency decreases or increases
- Off − removes all tuning and listening effects

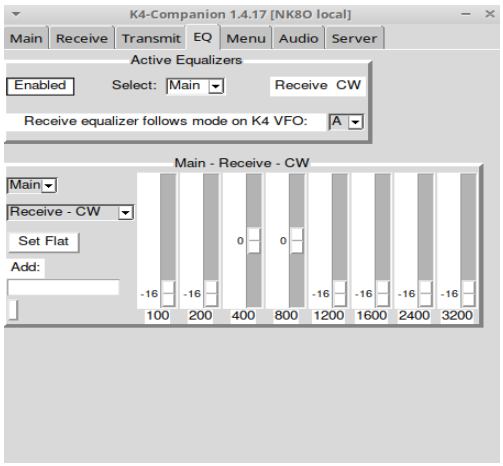
Note that the AF limiter only functions if the AGC is “Off”. This prevents extremely loud signals from blasting through at full-volume if the AGC is absent. AF Center Pitch functions by mode, and returns to the last user settings when changing modes. Noise Blanker, Noise Reduction, and RIT are explained in the K4 user manual.

Transmit Tab



- Transmit controls include ESSB (wide-band, high fidelity SSB). When selected, the SSB bandwidth varies between 3 kHz and 4.5 kHz
- SSB bandwidth for standard or CESSB (Controlled Envelope SSB) ranges from 2.4 to 2.8 kHz
- CMP, when set > 0, specifies the level of transmit audio compression when > 0, automatically enables CESSB
- Mic Gain − set to optimize SSB drive
- Monitor level − useful primarily if using K4 Companion for ancillary control of the radio while operating in the shack. No audio is returned when operating remotely. This control is independent of Sidetone level on the Main tab.
- VOX on/off with sliders to select parameters
- CTCSS – sets access PL™ tones for FM repeater operation

EQ Tab

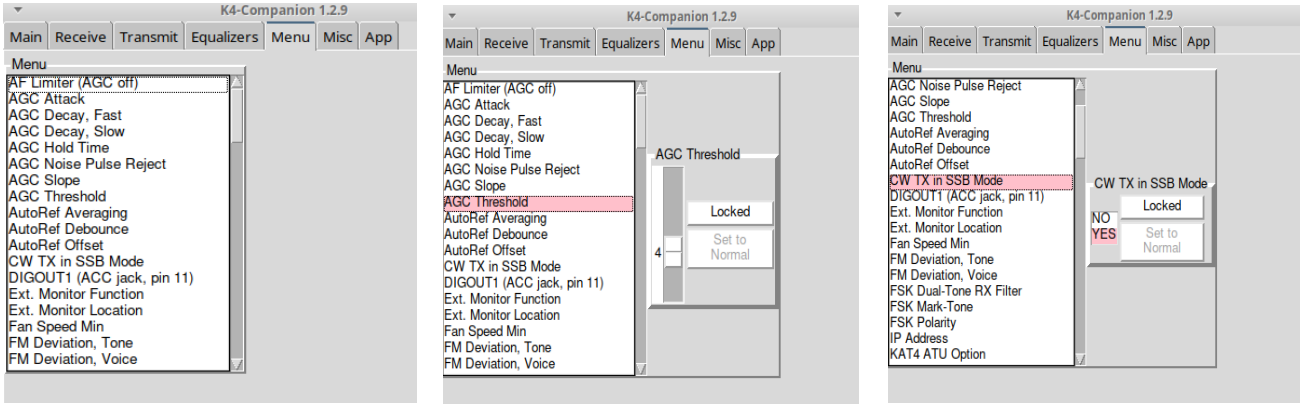


Transmit and Receive equalization can enabled or disabled, and be set by mode. These are personal preferences. When enabled **the settings on the K4 are altered.** The K4 settings do not revert to previous settings if the Equalizers on K4 Companion are disabled.

Please refer to K4 forums or the K4 User Manual for recommendations.

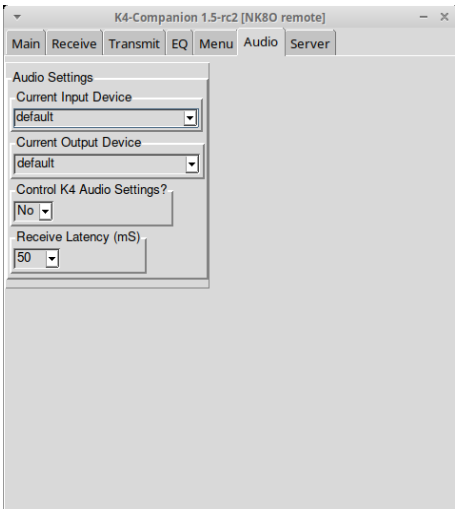
Menu Tab

The Menu tab reproduces **all** of the menu functions of the K4 menu functions. These functions are covered in the user manual. The configuration menu is a long, scroll-able list. When an item is selected, it appears in pink. Then a secondary widget opens that offers the menu choices. Some of these are simple YES/NO selections, others present a list, and some have a slider to select a value for the menu item.



The images show a few of the selections possible through the menu system. Please be sure to read the K4 user manual carefully. Changing menu settings without due consideration can have unexpected results.

Audio Tab



The Audio tab currently contains four settings:

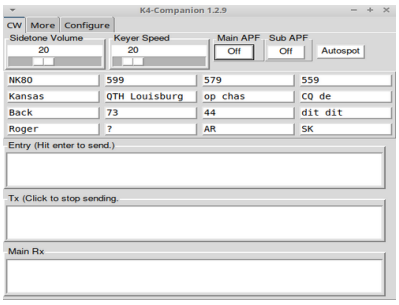
- Current Input Device
- Current Output Device
- Control K4 Audio Settings?
- Receive Latency

Typically the Input/Output settings will be correct when K4 Companion opens and they need not be changed. These settings vary by operating system. Audio connections are generally set to ‘default’ in a Linux installation but the choices may vary by OS ‘flavor’, and sound systems installed. In Windows there are a variety of settings, but often the choices mention ‘Speaker’ and ‘Microphone.’ Mac users should consistently see consistent settings that mention ‘Speaker’ and ‘Microphone’ since there is less variability in the hardware.

Control K4 Audio Settings? If you are using K4 Companion to control the radio while sitting in the shack, this can be set to Yes. Otherwise for remote operations, set it to No. See Notes on Operation below. Once this tab is configured, the settings are saved automatically. No changes are needed unless something in your operational configuration changes. The equalizer settings do change on the radio, regardless of the settings selection in the Audio tab. Frequent remote users may want to set the radio’s audio to zero, and ‘Control K4 Audio Settings?’ to ‘No’. In this way, the radio will be silent during remote operations.

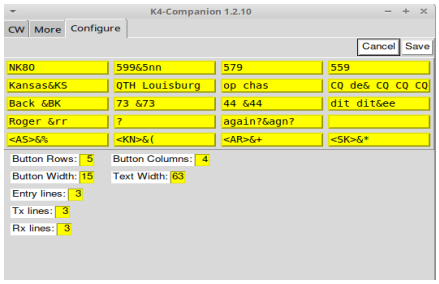
The Receive Latency setting is useful for remote connections where there is considerable jitter on the network path. The lowest setting is usually acceptable in the home network setting. A very high setting could be needed for Internet connections with limited upload speeds and higher variability. Note that increasing this setting results in a slight delay in the audio that is returned to allow sufficient time for audio packets to re-assemble in the proper order. The delay becomes more prominent as the value is increased.

CW Window



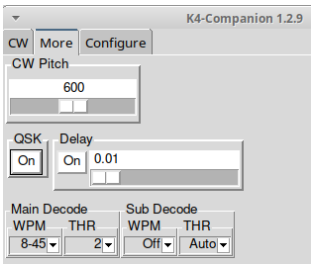
K4 Companion has a CW functionality via keyboard. There are a number user configurable macros that are easily set. These can be used sequentially in any combination. The typing buffer holds any text that is set until the user hits ‘enter’. The text will be sent and shown below in the Tx box. A cursor follows the text that is being sent. The sidetone will also reproduce each character audibly, unless the Sidetone level is set to zero. Tabs now divide some of the functions to prevent clutter of the interface. Audio peaking filter selection and Autospot are buttons in the CW tab. No spot tone is available remotely.

Several combination items require special characters to send correctly, but they have designations common to Elecraft. For example, ‘AR’ is sent with ‘+’ and ‘SK’ is sent with ‘*’. Other prosigns include ‘(‘ for **KN**, ‘=’ for **BT**, ‘%’ for **AS**, ‘_’ for **AA** (used in sending message traffic) and finally ‘!’ for **VE**. To label these items, use the “&” as the divider to differentiate the label from what is actually sent. The label can be anything you like that tells you what is being sent. In the instance above, ‘Kansas’ sends ‘KS’. The special characters ‘AR’ and ‘SK’ are show as below. Symbols that will not transmit on the K4 are excluded and a brief warning appears if any is entered in the text box.



Editing the buttons is accomplished with the ‘Configure’ tab. The ampersand (&) is used between the label and what is actually sent. To finish editing, click ‘Save’. Changes can be made at any time. In the default configuration there are 20 macro buttons. The number of rows, columns, and other configuration can be edited to taste in the ‘Configure’ mode, but there is no harm in leaving the default configuration. One might use one row for CWT, another for hunting parks, and another for chasing DX. There is no distinction between upper case and lower case letters when sent. Brackets <> can be used in button descriptors but not to transmit . These characters are disable in on the transmit side of the buttons

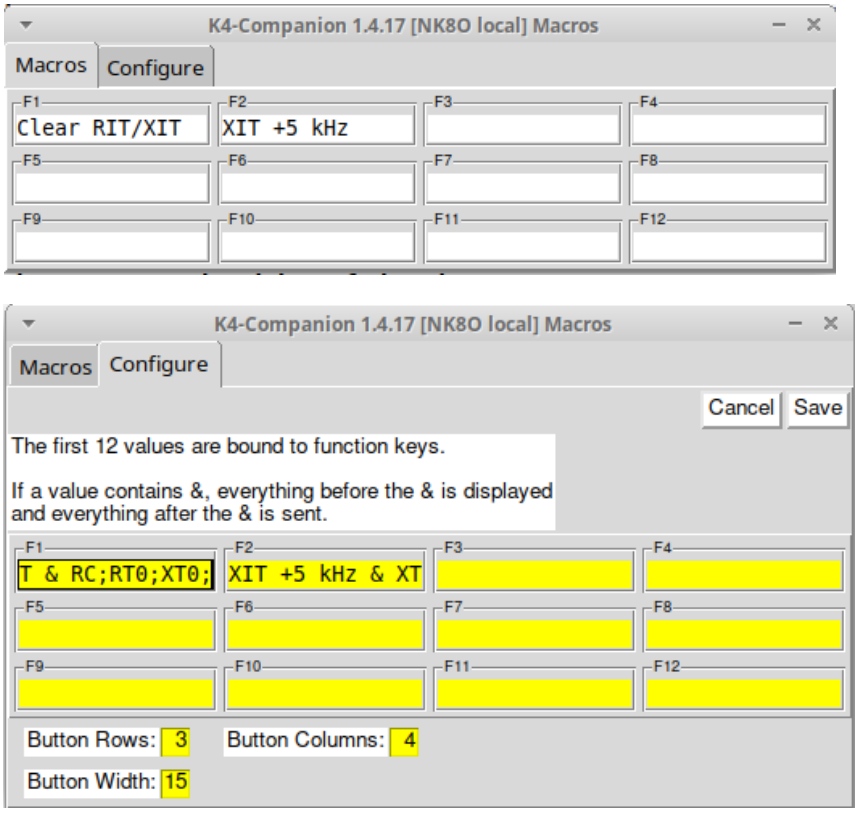
Some of the less obvious Elecraft CW designation are installed by default on the primary setup. These can be moved, changed, deleted, or edited to suit your preferences.



Finally, sending can be quickly canceled by a single click in the TX box. On editing if the user has a change of heart, the ‘Cancel’ button will undo any changes that were not saved.

Parameters such as CW offset/pitch are set in the “More” tab. QSK settings and CW decoder are also controlled here. The third line of the CW window does not appear unless decoding is enabled.

Macros Window



Macros add great flexibility and control to the K4. The Macros window adds the capability of adding creating from one (1) to two-hundred (200) macro buttons. The first 12 are bound to the function keys. These can be activated if the window focus is either in the Main window or the Macros window.

The macros window can be opened and closed in the Transmit tab of the main window or closed by clicking the X in the corner of the window. The ampersand (&) is used to delineate the button description from the macro string, in the same way it is in the CW window. Please refer to the Elecraft K4 Programmer's Reference or various K4 user forums for more information regarding macro strings.

Debugging

No guarantee is made that this software will run on a particular machine or distributions. However, debugging information may be useful to the developer. Information for submissions is listed on github.com by opening an problem on the “Issues” tab of the K4 Companion site. To initiate debugging, use the ‘-d’ option: *./k4companion -d -d .* This will show a running list of actions by the script onscreen. To save the debugging information to a file, use *./k4companion -d >> mydebugfile 2>&1& .* The debug file can be named anything you like and the file will be created if it doesn’t exist already. To append an existing log file, use *./k4companion -d >> mydebugfile 2>&1&.* The file can then be posted or e-mailed to Dale, W7DA. Information from dmesg may be helpful as well: *sudo dmesg ,* although more likely if there is a system failure somewhere and not necessarily with K4 Companion.

Unfortunately the debugging option does not work with the compiled Windows EXE file. Report the problem in as much detail as possible to the developer, and copy any error messages that do appear. The scripting has been thoroughly tested prior to compiling the Windows binary, so it is likely that the problems will be found in any case.

Notes on Operation

Audio functions on Linux can sometimes be challenging. A working knowledge of ALSA, alsamixer, and alsactl is useful. Pavucontrol can assist with selecting the correct audio input and output. Be sure to check the Configuration tab in pavucontrol, and if audio is not working, try another combinations. These may include “Stereo Duplex,” “Play HiFi quality Music,” and “Pro Audio,” among others.

Regarding Windows: I have noted it is prone to random audio failures, including disabling the audio for unknown reasons. You might have to dig into the subsystem a bit. I recommend using the Sound App, not the control panel. Use *mmsys.cpl* to access the Sound App. Control panel is often not very enlightening and the Sound App gives more information. As always, YMMV − your mileage may vary!

An experiment proved that K4 Companion will run on WSL – Windows Subsystem for Linux, but it is not easy to set up, and there are quite a few stumbling blocks. Choppy audio is probably the greatest impediment, probably due to adding extra layers of complexity to the audio subsystems although this may be ameliorated with the recently added audio setting for ‘Buffered Output Packets’. It does run under Parallels in macOS, as well as other emulators.

Enjoy!

Charles
NK8O
VE3ISD
5H3DX
nk8o@arrl.net

This manual was produced using FOSS, Free Open Source Software

Recent Revision History

Revisions since the last production release.

For the full revision history, please refer to the changelog found on the repository.

```
Version v1.5

Update changelog for v1.5

Maintain CW and Macros positions on tab change

Remove obsolete settings file, if it exists
   The .../settings-0.ini file is no longer used and should be removed.
   This commit removes it.

Remove the server proxy when it is set to blank

Fix bug in creation of new servers
   Previously, we would cache the number of servers, resulting in
   incorrect restart behavior when going from one server to multiple
   servers. This fix is to not cache the number of servers.

   Also, change the default name of a new server from "K4" to "K4 #1".

Discard queued audio on transition to transmit

Set audio output queue size by specifying latency

Add per-band ATU on/off setting to the To-Do list

Maintain cursor position on VFO adjutments

Fix initial setting of whether macros are enabled

Workaround for sub-recevier on setting
   THe K4 returns SB3; when the sub-receiver is on, rather than the
   documented SB1;. We'll now accept SB3; and treat it as SB1;.

Change formatting of printed warning messages

Make buffered audio packet count configurable
   Adds a dropdown in the Audio tab to set the number of buffered audio
   output packets.

Increase output audio queue size
   Allow up to 4 audio packets to be queued rather than 1.
   This is in hopes of mitigating audio artifacts caused by network
   jitter.

Fix VFO field character validation code
   Previously, the insertion cursor was set to the far left or far right of
   the field on every character. Now it only happens when an invalid
   character is entered.
```

```
Limit binding to 12 function keys

Append CW and Macros to their window titles

Allow backspace on Tx text field if possible
   It is only possible if there are greater than 10 unsent characters in
   the Tx text field.

Move XIT toggle button into the VFO A group

Modify initial macro values
   Include "Clear RIT/XIT" and "XIT +5 kHz" buttons.

   Also don't show a popup message for blank macros.

Add Macros window

Show error popup on missing config file

Look for k4companion.yaml in the script dir
   If k4companion.yaml is not found in the current directory,
   look for it in the directory where k4companon itself is located.

Refactor TextButtons out of the CW class
   THis is in preparation for using them in the Macros class.

internal only: Introduce TR pseudo cmd
   Combine the TX; and RX; commands into a single command,
   TRT; and TRR;, respectively. This simplifies some internal logic.

Enable space bar to toggle TX/RX
   When the main window has focus, the space bar will not toggle
   transmit/receive.

Clean up tab selection logic

Keep the CW Tx queue as full as possible
   To receive accurate feedback from the K4 on how many characters are in
   its CW Tx queue, we are limited to a maximum of 9 characters in its
   queue.

   We'll try to keep up to 9 characters in the K4's CW tx queue, even if
   that means that we'll normally be sending 1 character at a time to the
   K4.

Add CW-R and DATA-R modes

tools/update: remove 'Proposed' from commit title
```