

CI-V Router Customization

James Michener K9JM

I have put together a package for a “Basic” CI-V router, as a general solution to a common problem in the ICOM community. I am also making only the CI-V router shield available in the hopes others will wish to customize their own solution to meet their own stations need. In my own shack, the “basic” router is a subset of what I use on a daily basis. What I used is customized for my own purposes. This write up is to hopefully encourage others to “trick out” their own solution.

I have been active radio amateur now for well over 40 years, with radios from every decade. I have two large 6 volt golf cart batteries that provide power to a two meter radio and a Elecraft K2/100. I have a nominal collection of antennas in my small antenna farm and an Array Solutions “Six Pack” antenna switch. I have many items that operate from 12 volts, including 12 volt relays that turn on and off AC power to sections of the ham shack. I also have the desire to reduce the number of control boxes in shack, with flexibility of control is highly desired.

The first feature I wanted was the ability to take the frequency data from the ICOM CI-V bus and select the proper antenna. This is a simple feature. I built a small board that has a CMOS decoder IC and six relay drivers that I placed inside of the six pack control box. The decoder is powered off six pack rotary switch. The rotary switch, switches 12 volts to the relays, There are many unused positions and one is used to enable the decoder. This makes one position to be “automatic” while being able over ride the automatic by spinning the dial. As my antenna farm changes, the selected antenna is programmed through a table within the router. The software is part of the router library and permits selection of a different antenna for each kilohertz of the electromagnetic spectrum. I have it set up so that it selects the station dummy load when tuned outside of the amateur bands. I have a hard time remembering where the WARC bands are located.

The second feature I wanted was the ability to turn on and off 12 volts to radios, accessories and relays. The unit contains ten **large** power P channel FETs driving power through a PTC resettable fuse.

The third feature was control. I had thought of using serial control, but with the Arduino it is easy to incorporate Ethernet. So, added was the Arduino Ethernet Shield.

<http://arduino.cc/en/Main/ArduinoEthernetShield>

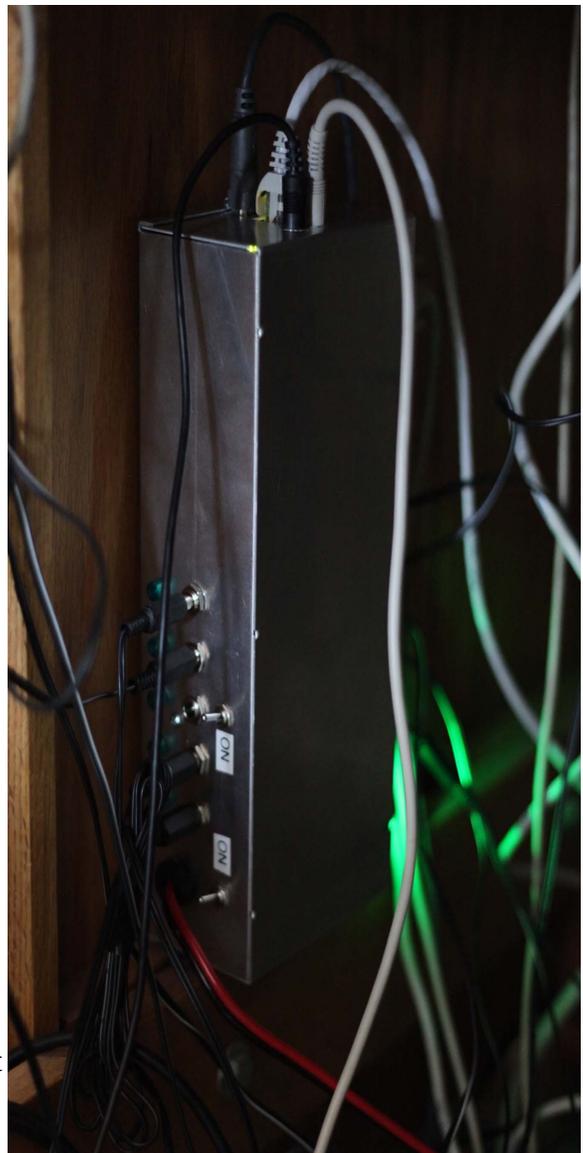


Illustration 1: CI-V Router Tricked out

This device has a hardware Ethernet stack and Arduino supplies a library so that it is easy to build an HTTP server. I built a simple HTTP server that offers up multiple web pages to provide control, but also simple HTTP “commands” so I can quickly control or acquire status. Everything that the CI-V router touches can be controlled over the network, and over the internet.

Since the box has a high current connection to the battery, the Arduino's 10 bit A/D converter can measure the voltage of the battery. Since the Arduino can turn on and off the 12 volt battery charger, it can turn off the charger once the battery is charged, and if I forget, it can turn it on when the battery voltage goes too low. Any ICOM command can be either read or set via the internet. One simple feature is the ability for anyone anywhere find out what frequency I am listening to, and how long it has been since I was on the radio.

I also built a simple Arduino based control panel. It has a simple menu structure, an optical encoder, switches and an OLED display with an Ethernet connection. This box has become the central controller for the station, sending out “quick” HTML commands to other devices over Ethernet. This device can be viewed as a hardware browser, the optical encoder and push button switches provide a more “old school” radio interface to everything. At the same time, I still have the ability to control everything using a computer through a browser, and I can customize the controlling web pages. Another feature is that this box also acts as an NTP client, so that when not being used as a control panel, it displays the date and time (in UTC), keeping itself in sync with atomic clocks over the internet. It has become the station clock.



Illustration 2: Sample web page



Illustration 3: Control Panel

Many have asked, why a control panel when one can use a browser? These question comes largely from the 'younger' generation than I. With the control panel I can program a list of commands to perform, a “macro”, where you can come into the shack and with a press of a button, everything in the shack jumps to a pre-configured state, without having to fire up a computer, or browser.

The same Arduino that does routing can also do filter decoding for SO2R.

In my shack, I only have a PW-1 and an 756P2, the third port of the router is not used. One of the three ports of the router can be jumpered so

that an external interface can be connected. In my case the third port is hooked up to an RS-232 interface. During development, this RS-232 interface connects to my computer providing status information. It also queries either the Electraft K2/100 or my Kenwood TS-940 for frequency information, so as to select the proper antenna off the second radio port of the Six Pack antenna switch. The proper frequency information can be directed over the network to a home brew antenna tuner, but that is another story.

The problem with most “off the shelf” solutions is that they do *exactly* what you want. I have rolled many of my own microprocessor projects in the past, but the Arduino boards provides a cost effective hardware solution as well as 20,000+ users and developers, so one doesn't have to reinvent the wheel.

For those thinking of tricking out their own CI-V router, or other Arduino project. Here are a few comments from my experience.

The integrated development environment that comes with the Arduino is functional but leaves much to be desired. If you already use Microsoft Visual Studio, you know that it is a very powerful editor for programmers, Visual Micro has a “plug-in” for Visual Studio for the Arduinos. Google Visual Micro for more information. Highly recommended. For the Eclipse lovers, yes there is an Arduinio plug in for Eclipse CDT. It is a bit cumbersome, but very good.

Also, when designing interfaces for the Arduino, remember that when you download code to the processor that it goes into a reset mode. Under these conditions, all the I/O pins goes into a high-Z state. Remember to put pull up or pull down resistors on the I/O lines to make things go the right way during resets. If you want the line to stay put, the best you can do is to implement a very high Z interface, such as CMOS gate input or a 2N7000 FET. Placing a .01 cap to ground will hold the state of the line during a software download into the Arduino.

If you have an idea of a way to trick out an Arduino, do searches to see if someone hasn't already built hardware to aid in the project. The amount of stuff available out there is amazing. Everything from stepper motor drivers, RF IDs, IR remote controls, WiFi, 2 Gigbyte flash data storage, every kind of sensor, POE (Power over Ethernet) the list goes on and on. There are also many small modules that provide for RS232 interfaces, switching regulators, LED arrays... etc. Here is a place with many of these odd things

http://www.mdply.com/index.php?main_page=product_info&cPath=9_41&products_id=295

I encourage anyone to consider tricking out their own CI-V router. This is an exciting and educational side of amateur radio.