Controlling USB & Serial Devices – Remotely

V1.0

Mark – WS7M

Overview

The ability to control things remotely has been a goal of mine since I first became an Amateur Radio Operator.

I remember the excitement of having my meager little station and working stations all over the US. But when our family would go on a trip, my station sat idle and it always seemed that this happened when the best contests were playing out.

I wished I could run my station remotely sometimes.

When I was dreaming of remote operations, we barely had 300 baud telephone models and they were expensive. The internet hadn't even been conceived.

In the 1980's I ran Packet BBS system both on 2m and on HF. Since everything was RS-232 and was fairly slow I achieved my first remote system. Using a 2400 baud modem I could call my home system from work and I could adjust parameters in my packet software.

This was a huge step forward but I still could not fiddle with my HF transceiver, a Kenwood, at that time nor could I adjust or tune my tube based linear amp.

But the biggest issue slower control faced was audio. If you wanted to hear the transceiver receive audio it was tricky.

Shortly after that solutions began to appear to run your station remote. But they typically were expensive. I think remote rig, just to get started today costs about $490.00. Add to that cables and other miscellaneous things you need and easily your remote cost can approach $1000.00.

As of roughly 2015 several true remote options started to become available. Icom radios can be run remote using their BA-1 server software and a client program. FlexRadios also can be run remote using their SmartSDR software.

These remote options both include audio. Not only can you control your radio completely, you can receive and transmit audio from your remote location provided your network is fast enough.

But both of these solutions to this day do nothing about the myriad other devices you need to control remotely. These solutions focus on the radio, not the accessories in your shop.

But has experimenters and hobbies there are ways we can do what we want to go if we are willing to put a little effort in.

This paper will discuss two solutions for controlling both USB and RS-232 devices remotely.

RS-232

RS-232 is a serial protocol where a “word”, typically eight bytes long, is broadcast over a single transmit wire by sending each byte at a time. The receiver reassembles the bytes back into the word and passes them to the host.

Early personal computers all included RS-232 ports but as USB has become the new standard it is difficult to find a PC delivered with serial ports. However it is easy to purchase serial port add on cards or cable for your computer.

The reason we need to do this is that while computers have moved away from RS-232 in favor of USB, there are many devices that still have an RS-232 port on them and we hams use quite a few of these device types.

Personally I prefer RS-232 over USB. USB is extremely fast, is plug-n-play but is limited to a cable length of about 15 feet. To exceed this length, you need to purchase special extenders. RS-232 is slower, requires manual setup but cable lengths if done right can be 2000 feet.

I am sure you are scratching your head and wondering why I still like RS-232? It is because it is also very easy to program for. Most compilers from C through the Borland Turbo compilers all included built in RS-232 support. Even Windows has built in support for RS-232. It is simple to open an RS-232 device and send data using just about any popular language.

USB on the other hand is harder. For a number of years after it became the new standard compilers did not natively offer USB device support. You had to purchase add-ons to get it to work.

Those days are past, but one fact still remains; USB while faster is still more difficult to use from a program development standpoint.

So to control an RS-232 device remotely we have many good options now. All involve purchasing a device called a “serial device server”. This a device that uses TCP/IP networking to allow data to be sent over TCP/IP and the device translates that data to RS-232 for your RS-232 device to use.

RS-232 Device Servers

If you simply search Amazon.com for “USR TCP Serial” you will be presented with pages of devices starting in price from $25.99 up through several hundred dollars.

Most of these devices will do exactly what we want but how easy this is depends upon what the device offers.

I call this “TCP Server Mode”. Different manufacturers may have different names but what it simply means is that the device, once setup, is a TCP server that you can connect to. Once connected any data sent to the device is sent out the RS-232 port on the device at the rate you have specified. Any data received through the RS-232 port is sent back over the TCP server connection for you to receive.

For devices that support this mode all that is required to send/receive data to your RS-232 device from around the world is an open port.

The open port with not without its security issues but that is well addressed on the internet and will not be discussed here. The one benefit is that while the port is open and hackers can send data to it, they really have to discover the RS-232 protocol your device uses to do any harm.

I have one of these serial servers and open ports on my network and I've seen hacker activity on those ports but none, so far, have succeeded in doing any harm.

One easy way around this is that some firewalls are starting to allow port forwarding only for known IP addresses which means a hacker can't even get to your device.

Another company that makes fantastic serial server devices is Moxa. www.moxa.com. These devices are amazing and come with great software support but they are substantially more expensive than the stuff seen on Amazon.

So the TCP Server Mode discussed here makes it easy to use either terminal program like Putty (https://www.chiark.greenend.org.uk/~sgtatham/putty/) to send RS-232 commands or if you are a reasonable programmer to have a remote program sending data over the internet to your RS-232 devices.

Some companies like Moxa provide software that will fool your remote computer into thinking it has a true COM port, like COM7. The software will translate RS-232 sent to COM7, send it over the network to your serial device server.

This software is sometimes worth the extra cost of a more expensive serial device server so that you can run software provided by your RS-232 device manufacturer to control your device. Most software that expects to work with RS-232 devices only knows how to look for COM ports on your computer.

Example:

I have a Palstar HF Auto tuner. It has an RS-232 port on the back. Palstar will not release the RS-232 protocol because the owner seems to think that is giving away his expensive technology. There is one piece of software that can talk to this tuner. It is free to download but when loaded on your PC it presents you a list of visible COM ports.

In my case I might want to control my HF Auto tuner via my laptop from a hotel. This program which I have no control over only knows how to talk to COM ports. I cannot give it a network address of my RS-232 device server.

But since I have a Moxa device server and their software supports the creation of virtual COM ports I can create a COM7 or whatever port I like and tell it that my serial server device is out on the internet at IP address a.b.c.d and port ## and it will fool the Palstar tuner program into thinking it is talking over a real RS-232 port.

So it depend upon your needs. If you think you have a need for something like this, then the extra money spent on a serial device server with COM port support like Moxa is worth it.

If you can write software then the lower priced devices and a little software coding time on your part will provide a cheaper solution.

I am a developer by trade and I ended up with the Moxa devices from a project that we did. So I use them just like the less expensive devices through software I write.

FRLogger, my logging program for Flex Radios (see www.ws7m.net) knows how to talk to serial device servers over TCP/IP. FRLogger can talk to Elecraft KPA amps and tuners and SPE amplifiers using RS-232 and serial device servers.

USB

Like computers, USB is slowly taking over devices as well. My SPE amp features both an RS-232 port and a USB port. SPE provides a cool little program that when talking to your amp over USB literally replicates the front display of the amp allowing you to fully control it.

When I first got my SPE amp I was thrilled that SPE provided this software free of charge. But all amplifiers are noisy and I really didn't want this thing 15 feet way. So my first remote was to by a simple 50 foot USB cable with built in extenders.

This allowed me to put my SPE amp in another room where the antenna cables where and where 220V was located.

There was just one problem. This worked great as long as I was at the computer that the USB connected to. I still could not control my map from my laptop on WiFi.

My prior employer used Indian based testers to test our software. Because our software was key locked using a USB device the testers could not run the software without access to a usb key.

Rather than deliver 50 usb keys to the Indian testing firm, our company went with a very expensive USB sharing system that allowed the Indian testers to have access to USB keys plugged into a special device in our Denmark office.

This seemed to be to be the answer to what I wanted to do which was to control my SPE amp, using their console software over WiFi and the internet. But the device they were using cost thousands of dollars and required specialized hardware to get it to work. It was a great system, very robust and reliable but hugely expensive.

It was by chance when I was researching this that I happened to type into Google "virtual USB" and the first link the popped up blew my mind.

www.virtualhere.com at an initial read seemed to do everything I wanted to do. I cringed when I hit the pricing menu item. But then fell back in awe when the price was $49. It seemed too good to be true.

So I purchased it. The first thing I tried was to put a USB server from VirtualHere on my ham shack main PC, a Windows server which had my USB cable running to my SPE amp. I then put the USB client software from VirtualHere on my laptop.

Over WiFi I gave it a try. I connected the VirtualHere client on my laptop to the VirtualHere server on my shack PC. I then ran the SPE software on laptop and instantly it found my amp and connedted!

So I could go anywhere in my WiFi network and completely control my amp. I was excited. This meant I could run Flex Radio SmartSDR and still have the ability to tweak settings on my amp from my deck table outside.

The VirtualHere site has great support and it was easy to find the port number that VirtualHere uses. So I opened that port in my firewall. The next week from work I ran my amp remotely from my work PC. It worked EXTREMELY well.

A about a month later I had to travel to Denver which was like 75 miles away but I was staying in a rather cheap hotel. The WiFi in this hotel was terrible. SmartSDR bare functioned but the USB control the amp was still rock solid. The only this was that due to the network the real time updates of power and SWR were sometimes slow to arrive.

So I was happy. This gave me virtually complete remote control of my entire station.

I was then sent to California for work. In the hotel run I made about 5 CW contacts using the amp then things fell apart. SmartSDR kept losing its connection then suddenly I could not connect to the USB server.

My wife called a few minutes later to tell me that my PC screen was showing the blue screen of death. I asked her to reboot it but it would not come back to life. So for three days in California I could do little to control my station. I had no idea what state the amp was in so I chose to simply not operate.

Upon my return I found my PC dead. My amp was ok, but it was online meaning if I'd tried to transmit it would have been at high power. This was unacceptable. If I was going to operate remote I would need something more reliable.

Enter the Raspberry Pi.

A Raspberry Pi, model 3 B+ costs about $50. Maybe less these days. It has 4 USB ports and when loaded with fresh software I've seen them run for a year straight. VirtualHere has Linux support. So I decided to create a "Control Pi".

I purchased a new Pi 3 B+, loaded fresh software on it (Raspian), and installed the VirtualHere optimized Raspberry Pi version. I gave my Raspberry Pi a static IP on my network so I could point my port forwarding at it.

I plugged in the SPE USB cable, now a much shorter version and I fired up my laptop to give it a try on my WiFi. It instantly connected and worked perfectly.

For the next year I used my station both on WiFi and over the internet and never suffered a single glitch as I had in California using my shack Windows PC. Performance was as good or better. Only when my location featured poor network did I ever see any lag in the USB console software talking to my SPE amp.

As I mentioned above the beauty of running the SPE console software is that it is like being right in front of the amp. Like many devices these days there are menus within menus and certain buttons get you to certain places. The SPE console software allows you to access all of that.

As this idea played itself out I conversed several times with the VirtualHere developer, a super nice guy. Every time I thought I had something I would like to ask him to add to VirtualHere I found he had already done it and using a command line flag would do what I wanted.

I asked him about what kinds of USB devices should work. His reply was all. Since VirtualHere implements true USB pass through it should not matter what the device is.

To test this theory I setup another Raspberry Pi and plugged USB from the PI to my shack PC. I then plugged in a second mouse into my laptop and with a little messing around I was able to make that second mouse be the mouse on my shack PC through VirtualHere over my WiFi network.

I had no need for this really but wanted to see how transparent VirtualHere was and in my opinion, I think it would work with virtually any USB device. Certainly, those on devices we Hams want to use should work just fine.

The cheap solution for controlling your USB device remotely involves the following:

- Raspberry Pi Model 3 B+ - Approximately $50 on Amazon. Some kits are more and come with everything you need to load the device.
- Your Raspberry Pi needs to have the latest Raspian installed and loaded. If you have not used a Raspberry Pi before there is lots of help on the internet on how to do things. But having a USB keyboard, Mouse, and HDMI monitor and cable for the Raspberry Pi will make things much simpler.
- Your Raspberry Pi should be given a static PI on your home network. You will want to open a port through your firewall to allow VirtualHere communications from the internet to get to your Pi. By default the port is 7575 but it can be easily changed to any port you want. Because the protocol going over that port is proprietary to VirtualHere, hacking is difficult.
- VirtualHere USB server – This free to install, but when you connect a client it will require an activation code. Read the full instructions on the VirtualHere website: www.virtualhere.com. About $50.
- You will need a VirtualHere client installed on the PC that you plan to use to connect to your Raspberry Pi over the network to control your devices.

The total cost for this setup is around $100 and about 3-4 hours of your personal time to set things up.

Use

The process for remote use is fairly simple:

1) On your PC you run the VirtualHere client software. Since you are over the internet you would manually add a VirtualHere server to the list and give it your public IP or if you have a domain name or dynamic DNS you can use those. You specify the port number which you opened in your firewall and configured in the server as the listening port.
2) On your PC you instruct the VirtualHere client to connect. It will let you know if connection is successful.
3) If connection is successful, then you can run whatever software you would normally run to connect to your remote USB devices. In my case this would be the SPE console software.

At this point you use your PC as if you were directly connected to your USB devices. There is rarely a need to manually terminate the VirtualHere connection but you can do that as well from the VirtualHere client.

It really is that simple.

The most difficult part of this process is the setup of the Raspberry Pi, installation of software on the PI, configuration of ports, opening your firewall etc. It can take some tweaking and testing to get everything to work.

But as I said above, the Raspberry Pi can be extremely stable. I checked my Pi before starting this paper and found it had been running straight for 376 days.

All of this can also work over a VPN. If you are a security nut and do not want to open ports then you can make all of this work over a VPN. If your VPN provides you remote access to your subnet then VirtualHere will automatically find the server and allow connection.

In my testing the VPN provides a minor slow down in data few but it was not noticeable controlling my SPE amp.

Added Benefits

The Raspberry Pi offers a substantial amount of IO that can be used to control relays or almost anything you desire. The Pi also has an RS-232 port on it but it is difficult to use unless you want to write a small python or C program on the PI to use it.

The IO over is very easy to use.

Searching Amazon for "Raspberry Pi Relay Board" yields pages and pages of results.

This is a 3 relay board: $25

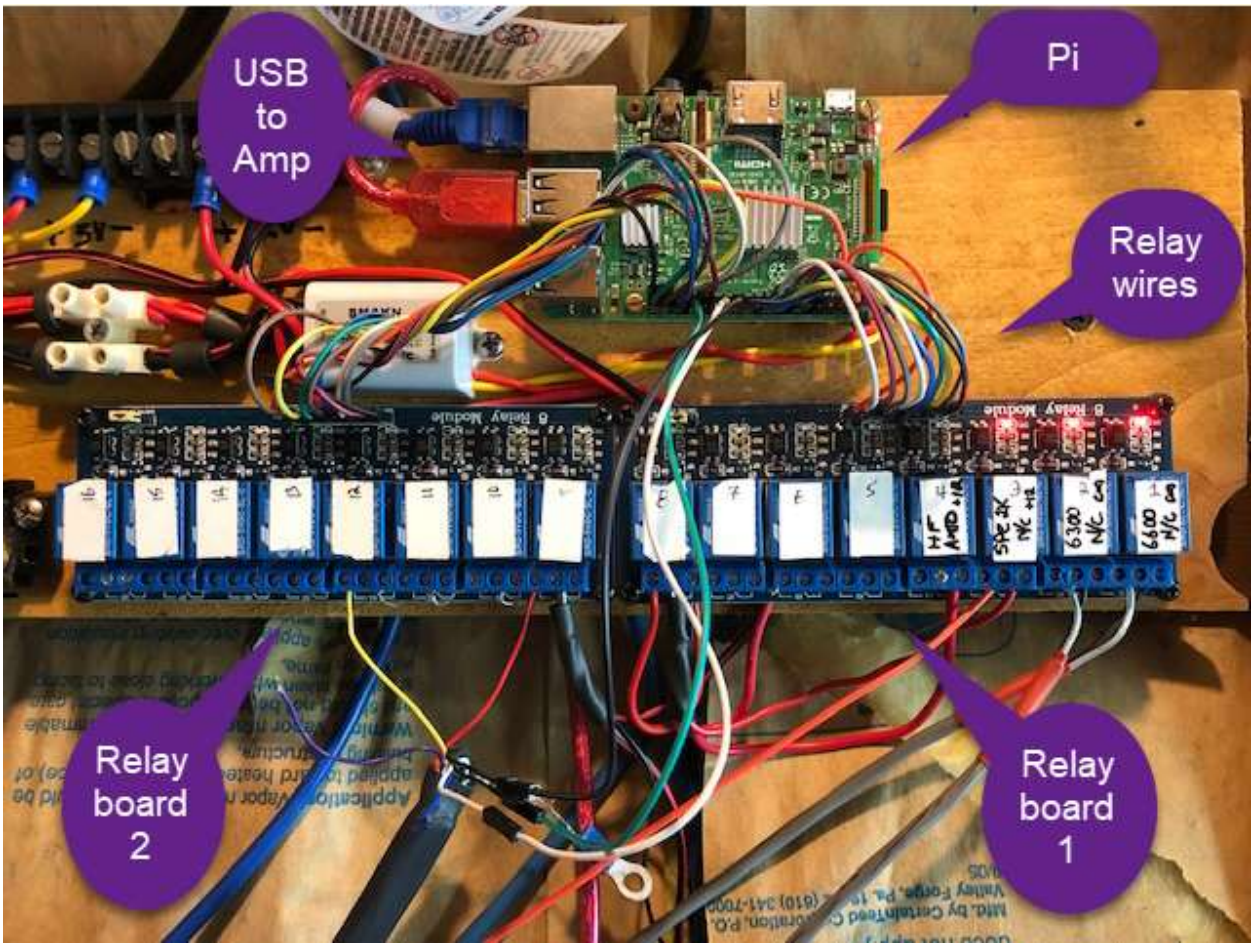
https://www.amazon.com/Raspberry-Pi-Expansion-Module-XYGStudy/dp/B01G05KLIE/ref=sxts_sxwds-bia?keywords=raspberry+Pi+relay+board&pd_rd_i=B01G05KLIE&pd_rd_r=232b4e9d-dbc4-4f46-ac89-89f27a955017&pd_rd_w=Ky20c&pd_rd_wg=Kx9jD&pf_rd_p=f0479f98-a32d-45cd-9c12-7aaced42b1ec&pf_rd_r=SKRBTR8CQJBP8PFM101Y&qid=1558963447&s=gateway

Here is an 8 relay board: $11

https://www.amazon.com/DZS-Elec-Channel-Indicator-Raspberry/dp/B078GTFVNS/ref=sr_1_2_sspa?keywords=raspberry+Pi+relay+board&qid=1558963504&s=gateway&sr=8-2-spons&psc=1

In my own setup I use two 8 relay boards to control various station functions:

1) Flex 6600 remote on/off
2) Flex 6300 remote on/off
3) SPE amp remote on/off
4) DC power (12V) to Palstar Tuner
5) DC power to Flex Radios
6) Palstar tuner bypass
7) Antenna switch



To control the IO on the Raspberry Pi I like to use a library called PIGPIO. (http://abyz.me.uk/rpi/pigpio/). This library allows control from C programs, Python, and even the command line on the Pi using their utility.

It is simple to install and use and all instructions are on their website.

Since the Raspberry Pi is running Linux installing a webserver and crafting some webpages to control your station is not terribly difficult.

My station control web is simple. It is password protected:

Once logged in:

This layout is intentional because it works very well on my phone.

The general idea of this web control is at the top I have “Station preparation” Enable/Disable. These are quick link to setup the station for use or to shut it down.

Enable turns on all external devices and gets things ready. Disable turns stuff off.

If I need to restart my Flex radio I have links for that.

I have links to power on/off the SPE amp

I have some audio controls. I tend to create RFI into our home intercom so I can turn that on/off.

Lastly there is something called main power cycle. I have encountered twice a situation where my radio was hung badly enough that the normal remote on/off did not reboot the radio. The power cycle is designed to simply pull the plug on the DC to all components for a period of 30 seconds then DC is reapplied. Twice this feature has allowed me to do a complete remote reset of everything.

Many of the controls on this webpage toggle the relays on the Raspberry Pi relay board.

But some of the control talk to Digital Logger Web Power Switch devices. The US uses 110V power so I use a number of these devices around my home: https://www.digital-loggers.com/lpc7.html Sending commands to these devices from the Raspberry pi is simple.

Digital Loggers makes a DC controller: https://dlidirect.com/products/dc-power-controller

And I am sure there are other devices that do similar things.

For me the goal was to bring it all together into my Raspberry Pi based webpage.

Making It Available

The key to a remote station is making it available over the internet. If you are like me, I have standard residential internet. No special static IP address or speed.

So to be able to find my stuff on the internet since my provider can change my IP address at will I use NOIP. https://www.noip.com/. I pay for a premium account because I want stability.

NOIP offers a Linux tool that I installed on my Control Pi that periodically notifies the NOIP server of my public IP address.

This way my chosen domain goes right to my station control. No need to remember addresses.

Your firewall, if you have one, and I hope you do, needs to have some ports open for most of this stuff to work OR you need to setup a VPN.

VPNs are great but they add an extra layer of complexity in getting remote stuff to work right. Plus for VPN you add a few extra steps to get that connected first.

Since Smart SDR began to offer Smart Link for remote connection I have not used my VPN except in rare occurrences. I get better performance using Smart Link and having open ports for my device controls.

A few security and General Cautions

The internet is a bad place. In many countries of this world there are people with nothing better to do than to try and cause issues and problems. Sometimes this is just because the can or they get a thrill doing so. In other cases the country government is helping them to do it.

For a number of years I ran a music composition site where people could sign up and share and collaborate on music they were writing. The site had the ability to upload music files for sharing and review.

Each day this site received some 25,000 penetration attempts. The site had a 2 factor sign up authentication. It did not implement CAPCHA at that time.

The signup process was to register, input an email and the site would email you a link to confirm your email. If the link was not clicked in 24 hours the site deleted all traces of you.

I received a notice once day from my hosting service that my email send limit was approaching. When I logged into the site I saw that there were some 50,000+ pending registrations all added that day.

So someone, somewhere wrote a script to run the registration the site over and over providing random names. To what purpose I have no real idea except to cause trouble. There was nothing they were gonna get from the site. No financial data was there. Only login names, emails and passwords all of which were heavily encrypted.

Even if they had managed to download the user database everything in it was 256 bit level encrypted. Each record was different. They would have spent a whole lot of time decrypting that only to find they would get nothing more than an email address, most of which were publicly available elsewhere.

But it just goes to show you that your site will be hacked.

My control Pi gets slammed probably 50 times a day. My web page is written so that if I see more then 250 connects from either the same or random IPs that the website goes deaf. It will only except a special rotating code that I put in to come back alive.

So to the remote hacker suddenly it just stops responding. It sends “accept” codes but then never sends any data back.

The beauty of this is that they think they have succeeded in killing the webserver and move on to something more interesting. I connect send a special rotating code and the website comes back alive and I can control my devices.

So it goes without saying that if you following any of the processes in this document on how to remote control your devices be aware you will get hacked. Not only am I not responsible for any damages but you will be assuming the full risk by putting your devices online.

That being said, it is pretty easy to make it hacker resistant and at worst case boring (like I do) for the hacker when suddenly nothing happens.

Just be aware of the threats.

Conclusion

In this document I’ve given you sources and ideas for how to achieve full station control remotely over the internet. I have not given you much more than the sources and ideas for things.

I am willing to provide more but it takes time. There is no turnkey solutions to these things and we must create them specific to our needs.

I feel that remoting your station is a learning experience. I could provide a canned solution but then I’d have 100 people asking me for a special modification to handle their special need.

So I’m taking the approach of not directly feeding you fish, but rather teaching you to fish so you can feed yourself.

I am available to answer questions. Send me an email. Describe your issue or problem and I’ll see what I can do to help.

Mark – WS7M

ws7m@arrl.net