WIFI Wireless Rotator Control -- Satellite or HF/VHF/UHF

by Gordon Gibby KX4Z

Running multi-conductor rotator cabling a long way out from your rotator controller to your tower or antenna can get expensive fast! Especially if you are running DUAL rotators for a satellite system. Often you need reasonably sized conductors to run the electric motor of the rotator(s). I have been using 5-conductor #18 AWG cabling https://www.amazon.com/dp/B0CL6NMC6Z and that is running $0.67 per foot. That is about the same price as bulk RG8X (https://www.dxengineering.com/parts/dxe-8x) and about half the price of LMR-400 equivalent coaxial cable: https://www.dxengineering.com/parts/dxe-400max So it isn't cheap!

One of the solutions to this problem is to use WIFI to remotely control the rotators. There are semi off-the-shelf controllers made to do this (if you provide weatherproofing). Examples include:

- AF6SA wifi remote control: http://www.af6sa.com/projects/WiFi_Rotor.html $119. http://www.af6sa.com/projects/Kits.html youtube review: https://www.youtube.com/watch?v=hHy40_-eohU (This same product can be found -- for a higher price! -- from eBay vendors.)

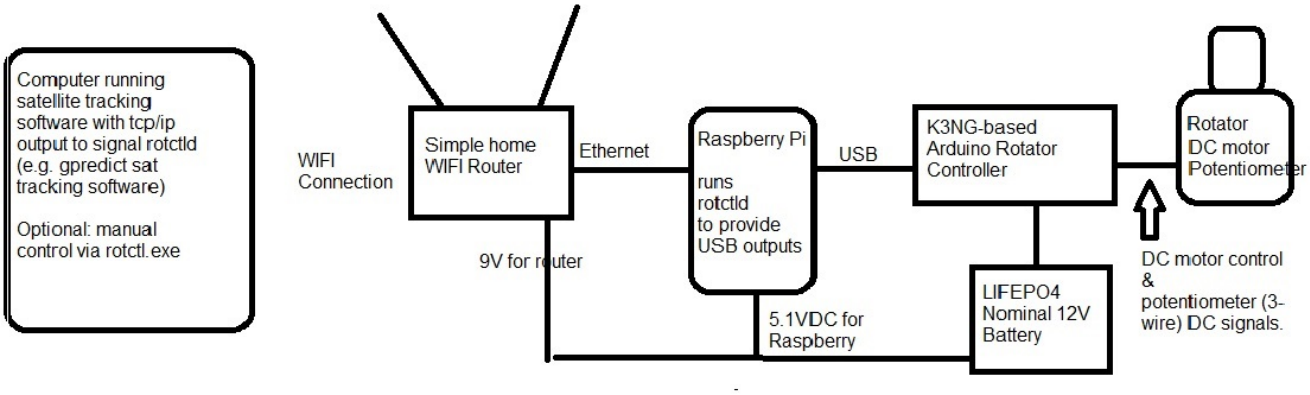
But I already have created the printed circuit board for a version of the K3NG rotator controller (freely available Gerber files that you can have fabricated at your favorite PCB shop: https://www.nf4rc.club/how-to-docs/gerber-files-for-k3ng-rotator-printed-circuit-board/ and also a pcbway shared project:

https://www.pcbway.com/project/shareproject/K3NG_Rotator_Dual_Axis_Controller_Board_fd759cc2.html). I already have that Arduino-based rotator controller nicely controlling a homebrew dual-axis satellite system, with a USB input, I wondered if I couldn't just use a Raspberry Pi to make it wireless?

The answer is YES!

The idea is to have the Raspberry Pi connect to a home-type WIFI network, using WIFI router and then use any desired automated or keyboard control to send commands to Raspberry Pi. This is easy, by simply having the Raspberry pi run the tiny little hamlib program ***rotctld*** (part of the hamlib package). And I already had an older Raspberry Pi 3B running a "shelter server" as part of a much older project, complete with a cheap home router already. (See previous articles on the Raspberry Pi older shelter server: https://www.nf4rc.club/how-to-docs/county-ares-docs/shelter-wifi/ and https://www.nf4rc.club/how-to-docs/county-ares-docs/shelter-wifi-2/) The zero-cost ***rotctld*** (hamlib) system is well-known and zero-cost ***gpredict*** directly talks to it. There are probably many other systems that do, also!

The overview of the system then looks like this:



Controlling the Rotator

Using this system, only coaxial cables need to be run the distance to the antenna rotator -- the rotator can be controlled wirelessly over WIFI. A large number of software packages are available that take advantage of rotctld to allow control of antenna rotation systems:

No.	Manual or Automated Control	Software and availability	Use
1	Automated	gpredict https://sourceforge.net/projects/gpredict/ Available for both Linux and Windows operating systems	Free software that downloads satellite Keplerian elements and automatically keeps antenna pointed at desired satellite
2	Both	calrotator -- https://www.pianetaradio.it/blog/catrotator/ Extensive documentation available	Free software available for both Linux and Windows that allows manual or automated control (interfaces with WSJT-X and with a satellite control system)
3	?Both	Web Radio Control https://doc.webradiocontrol.tech/	Commercial product with free trial that allows remote control of ham station over Internet, including rotators
4	Both	N1MM rotator control (portion of free N1MM logging system) https://n1mmwp.hamdocs.com/setup/interfacing/#n1mm-rotator-control	Integrated solution with graphical interface.
5	Manual	rotctl (Part of hamlib package) https://github.com/Hamlib/Hamlib Available for a dizzying array of platforms	Manual simple control of rotator can be used in practice or for simple testing.

Another one of the hamlib programs, **rotctl** (available for both Linux and Windows) can manually control the antenna at the same time if desired!

Hamlib's ***rotctl*** is excellent for initial testing of your system. I haven't studied all of its available keyboard commands, but a few simple ones include:

Command Example	Result
p	Will interrogate the rotator for the current positioning and print that out on the screen.
P 100 50	Will cause the controller to go to azimuth=100 degrees, elevation = 50 degrees
S	Will immediately stop the rotator if needed

The typical K3NG software sets the baud rate of the Arduino to 9600 using the #define for **CONTROL_PORT_BAUD_RATE**, which is defined in **rotator_settings.h** 9600 works fine, but you can change if if you wish.

Here are the steps that I used to program my Raspberry Pi to make this work:

No.	Details	Comments
1	You need to know the IP number that your raspberry pi will be using on your chosen WIFI router. This is often set with a web-based control for your particular router. Review the documentation for your router to find how to set this to make it permanent.	My Raspberry was set to be on (private net) address 10.10.10.10
2	The default port number in hamlib for rotator control is 4533.	You can change this if you wish, but it works fine and gpredict defaults to this.
3	My raspberry pi had a firewall known as **`ufw`**, known in the trade as "uncomplicated firewall" You can easily find lots of documentation, such as: https://help.ubuntu.com/community/UFW By executing the following commands, I instructed the firewall to allow communications to the software over port 4533 sudo ufw allow 4533/tcp sudo ufw allow 4533/udp	Probably only one of these is actually required. You can find much more information on how I set up the older raspberry pi to provide a web server for shelter residents in these two articles: https://www.nf4rc.club/how-to-docs/county-ares-docs/shelter-wifi/ https://www.nf4rc.club/how-to-docs/county-ares-docs/shelter-

		wifi-2/
4	I downloaded **hamlib** and installed it on the raspberry pi. You can find it here: https://github.com/Hamlib/Hamlib/releases/download/4.6.2/hamlib-4.6.2.tar.gz as part of this page: https://github.com/Hamlib/Hamlib/releases/tag/4.6.2 and then you'll need to google a bit to learn how to "extract" it and install it.	
5	Finding the USB Port: run the command ls /def/tty* and study the listed ports before you connect your raspberry to the K3NG rotator controller. Repeat that command after you have added the K3NG rotator controller with the appropriate USB cable. There should be a new entry -- and that will be the port to which you will direct rotctld to send its commands. In my case, it was `/dev/ttyUSB0` ("zero" not "Oh")	
6	**Making rotctld autostart on pi reboot:** Each successive update of raspberry pi software has different means of automatically starting up software at reboot. More recent versions tend to use **systemd**, but my older system works tolerably well by just adding a line to **`/etc/rc.local`**, which gets executed each time the raspberry reboots. Using `su vi /etc/rc.local` (because I like vi -- you might prefer a different editor, of course!), I added the line (all on one line) `rotctld.exe -m 603 -r /etc/ttyUSB0` systemd is a more advanced method of autostart. More information on autostart setup using systemd can be found in multiple locations: https://documentation.suse.com/smart/systems-management/html/systemd-basics/index.html	The simple solution probably won't restart the program if it quits, but rotctld seems incredibly robust and automatically handle connections and disconnections from multiple different human and computer client connections over port 4533 If you don't specify an address for rotctld to listen to, it listens to ALL the IP numbers of the raspberry and the default port is 4533. If you need special adjustments, consider studying: https://www.mankier.com/1/rotctld

	https://people.redhat.com/pladd/systemd_NYRHUG_2016-03.pdf	

You can test the workings of the wifi by using a windows computer onto which hamlib has been installed, and which is on the same network as your raspberry pi. Bring up a terminal windows using the Windows key + r and typing in "cmd" to get a terminal window.

```
rotctl -m 2 -r 10.10.10.10:4533  -vvvv
```

(replace the IP # with the proper one for your setup)
The -vvvv allows you to see a bunch of debugging information which may help if it doesn't work immediately.