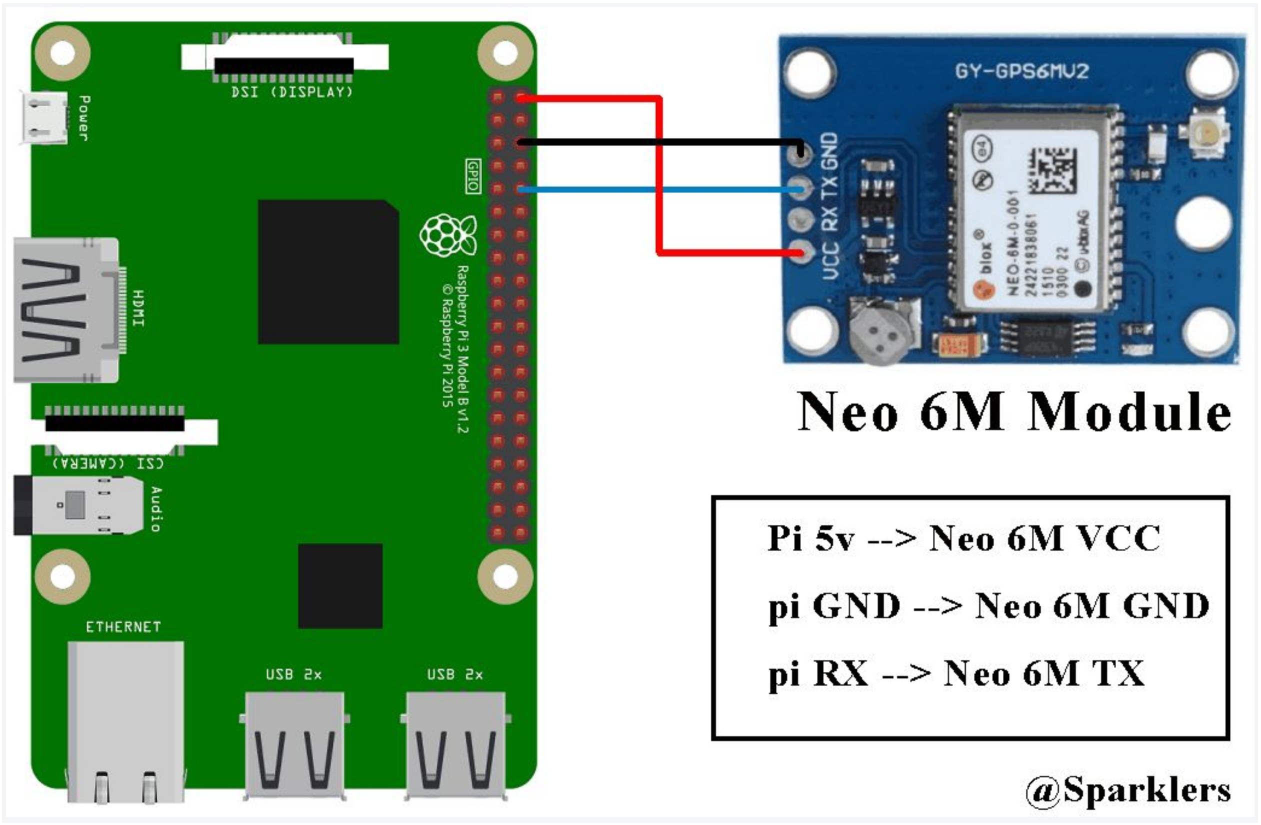
**NEO 6M GPS Sensor**

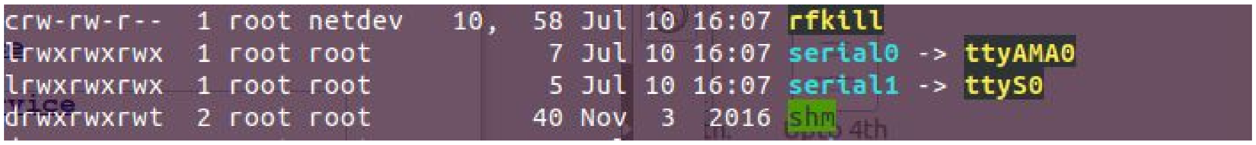
GPIO Pin Setup: Please note that this setup requires the Raspberry Pi to be running Raspbian, and was successfully tested on Raspbian Buster. Here are some instructions on setting up the NEO 6M GPS Sensor to work with the Raspberry Pi, as well as some example programs where the GPS logs data to a `.csv` file and then maps it.

1. Wire the sensor to the Pi as shown in the image below, which was taken from the Sparklers the Makers GitHub page referenced at the end of this tutorial



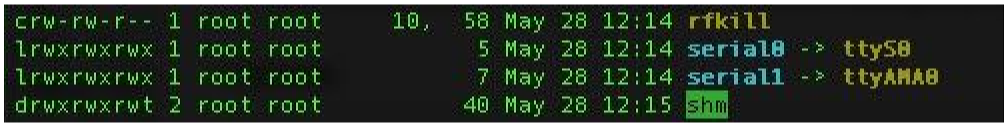
2. Edit the `/boot/config.txt` file to enable UART, the Serial Interface, and disable Bluetooth.
   a. `sudo nano /boot/config.txt`
   b. Add the following lines:
```
dtparam=spi=on
dtoverlay=pi3-disable-bt
core_freq=250
enable_uart=1
force_turbo=1
```
   c. Now save and exit by pressing `ctrl+x`, type `y`, and then press `enter`.
3. Next, make a copy of the `/boot/cmdline.txt` file.
   a. `sudo cp /boot/cmdline.txt /boot/cmdline_backup.txt`
   b. `sudo nano /boot/cmdline.txt`
4. Now edit it to disable the UART serial console.

    a. `dwc_otg.lpm_enable=0 console=tty1 root=/dev/mmcblk0p2 rootfstype=ext4 elevator=deadline fsck.repair=yes rootwait quiet splash plymouth.ignore-serial-consoles`
    b. Now save and exit by pressing `ctrl+x`, type `y`, and then press `enter`.
5. Now reboot the Pi
    a. `sudo reboot`
6. After rebooting the Pi, make sure you are near a window, outside, or are in a location where the sensor can get a GPS signal.
    a. There will be a blinking LED on the NEO Sensor when it receives a signal from GPS satellites. Depending on the specific chip, the light could flash green, blue, purple, or even red. The exact colour is not important, just as long as one flashes.
    b. If the light does not blink within five minutes, move closer to a window or to a place under the open sky. Keep in mind that the sensor could take as long as 15 minutes to acquire its first satellite fix.
7. Once the LED is blinking, run the following command:
    a. `sudo cat /dev/ttyAMA0`
    b. This should print out streams of data, and can be stopped with `ctrl+c`. If there are error messages, this is likely due to a weak signal or a lack of signal. Essentially, this step verifies that the sensor is working.
8. Next, we need to make sure the Pi knows where the sensor is wired to.
    a. Run the `ls -l /dev` command.
    b. If your output looks like the image below (from the Sparklers the Makers GitHub page), run the following commands:



```
crw-rw-r--  1 root netdev   10,  58 Jul 10 16:07 rfkill
lrwxrwxrwx  1 root root          7 Jul 10 16:07 serial0 -> ttyAMA0
lrwxrwxrwx  1 root root          5 Jul 10 16:07 serial1 -> ttyS0
drwxrwxrwt  2 root root         40 Nov  3  2016 shm
```

    c. `sudo systemctl stop serial-getty@ttyAMA0.service`
    d. `sudo systemctl disable serial-getty@ttyAMA0.service`
    e. However, if your output looks like the image below (from the Sparklers the Makers GitHub page), run the following commands:



```
crw-rw-r-- 1 root root     10,  58 May 28 12:14 rfkill
lrwxrwxrwx 1 root root          5 May 28 12:14 serial0 -> ttyS0
lrwxrwxrwx 1 root root          7 May 28 12:14 serial1 -> ttyAMA0
drwxrwxrwt 2 root root         40 May 28 12:15 shm
```

    f. `sudo systemctl stop serial-getty@ttyS0.service`
    g. `sudo systemctl disable serial-getty@ttyS0.service`
9. Next, install some of the libraries that you'll need to run the program we're about to get:
    a. `sudo pip3 install folium`
    b. `sudo pip install pynmea2`
    c. `sudo apt-get install python-pandas`
10. From there, download the code from GitHub.
    a. `cd`

   b. `mkdir git`
   c. `cd ~/git`
   d. `git clone http://github.com/amichael1227/gps_test.git`
11. To make sure the sensor works properly, move the code to its own directory.
   a. `cd ~/git/gps_test`
   b. `mv ~/git/gps_test/pi_tests ~/`
12. Before we use the GPS sensor at all, we need to update the GPS sensor so it knows what the date and time is
   a. `cd ~/pi_tests`
   b. `sudo python gps_update.py`
   c. This program will continuously spit out values, but you can end it as soon as it starts spitting out the proper time. This can be done by pressing `ctrl+c`.
13. Now, run the first program to make sure we are getting a good GPS reading.
   a. sudo python `initial_gps_test.py`
   b. Running this program should continuously print you `Latitude= xx.xxx and Longitude= xx.xxx`.
14. Next, let's log the GPS data in a `.csv` file. This program should both print it to gps.csv as well as print the `Latitude= xx.xxx and Longitude= xx.xxx` continuously on the screen.
   a. `sudo python initial_data_logger.py`
15. You can verify this by opening the `gps.csv` file with a spreadsheet editor, and you should see three columns of data that contain the latitude, longitude, and timestamp in that order. This data may be useful for you, but there need to be headers to map out the information. You can either add these yourself, or you can let the program do it for you.
   a. `sudo python gps_data_logger.py`
16. Now, when you open up the `gps.csv` file, you will find that the data columns are named `Latitude`, `Longitude`, and `Timestamp`. This means that the data can now be mapped out
   a. One way to do this is to use a website that takes an uploaded `.csv` file and plots it out. An example of such a site is:
      i. https://www.gpsvisualizer.com/map_input?form=data
   b. Another way is using the folium library. An example of this is built into the `gps_mapper.py` script.
      i. First, edit the gps_mapper.py script to reflect your location
         1. `nano gps_mapper.py`
         2. Replace the `[41.643442,-71.178253]` part of the `mapPATH` definition with the coordinates you want your map to center on.
         3. You can also edit the `zoom` of the map as well.
      ii. `sudo python3 gps_mapper.py`
17. Open the RobotPath.html file in your preferred web browser and view your map!

The information for this sensor tutorial was compiled from the following sources:
https://sparklers-the-makers.github.io/blog/robotics/use-neo-6m-module-with-raspberry-pi/
https://stackoverflow.com/questions/42879408/writing-variables-to-a-csv-row-python

https://ozzmaker.com/how-to-save-gps-data-to-a-file-using-python/
http://comet.lehman.cuny.edu/owen/teaching/datasci/foliumLab.html
https://projects.raspberrypi.org/en/projects/mapping-the-weather/
https://www.raspberrypi.org/forums/viewtopic.php?t=168440
https://github.com/FranzTscharf/Python-NEO-6M-GPS-Raspberry-Pi/blob/master/Neo6mGPS.py

Launcher File Setup: We can set the Pi up so that the `gps_data_logger.py` will run at boot, regardless of whether or not you are connected to your Pi. The file that does this, `launcher.sh`, is already written and included in the GitHub Repository that was cloned earlier on in this section. That being said, the Pi still needs to know to start the program at boot.

1. Edit the `launcher.sh` file and uncomment the execute command.
    a. `cd ~/pi_tests`
    b. `nano launcher.sh`
    c. Remove the `#` in front of the `#sudo python initial_data_logger.py`, so that it reads `sudo python initial_data_logger.py`.
    d. Now save and exit by pressing `ctrl+x`, type `y`, and then press `enter`.
2. Make the `launcher.sh` file executable
    a. `chmod 755 launcher.sh`
3. Test the file, it should run the Python script, you can exit out with `ctrl+c`.
    a. `sh launcher.sh`
4. Make a logs directory for error logs.
    a. `cd`
    b. `mkdir logs`
5. Have the `launcher.sh` file run at boot.
    a. `sudo crontab -e`
        i. This command opens the crontab and may prompt you to choose an editor, choose whichever one you are most comfortable with.
    b. Add this line at the bottom:
        i. `@reboot sh /home/pi/bbt/launcher.sh >/home/pi/logs/cronlog 2>&1`
    c. Now save and exit by pressing `ctrl+x`, type `y`, and then press `enter`.
6. Reboot.
    a. `sudo reboot`
7. If the code doesn't start running, you can check the error logs for troubleshooting.
    a. `cd ~/logs`
    b. `cat cronlog`

The information for this sensor tutorial was compiled from the following source: https://www.instructables.com/id/Raspberry-Pi-Launch-Python-script-on-startup/