Using Ground Station Software

Connect to NUC with ethernet and set static IP address on user computer to 192.168.1.151. In Ubuntu 18.04 this is done by adding the following lines of code to the end of the /etc/network/interfaces file, replacing ‘enp2s0’ with the name of the ethernet port:

auto enp2s0
iface enp2s0 inet static
address 192.168.1.151
netmask 24

Then, log into the NUC using the following command and password:

ssh -Y argus@192.168.1.150
Password: skobuffs

Run the following command to start while in directory with argus software:

cd Documents/ARGUS/
sudo ./argus.py

To quit the program, click the red “QUIT” button.

To use GPS location click on the “GPS” Radio button. If the “ARGUS.qth” radio button stays checked, look at the terminal to see the error:

“No GPS Connected to ‘GPS path’. Are you root?”

This means that either the GPS is not plugged in or the software is not being run as a root user.

“No GPS data received from ‘GPS path’. Check if red light is blinking.”

This means that the GPS is connected, but the software is not receiving GPS data. If the red light is blinking on the GPS, this means that it has not found a lock yet and you have to wait for it to find a lock and the red light to become solid. If the red light is solid, try clicking the GPS radio button again.

To begin the connection with the motor, click “Connect Motor”. If the button does not switch to red displaying “Disconnect Motor”, check the terminal to see the error:

“No motor controller connected to ‘Motor Path’. Are you root?”

This means that the software cannot connect to the motor controller. This can either mean that it is not being run as a root user, or that the motor controller is not connected. If it is being run as root AND the motor controller is plugged in to the computer and turned on, try clicking the “Connect Motor” button again.

Once the motor is connected, it will set the displayed angles to the current angles the motor controller believes that the motor is pointing at.

Initially, the software will be in manual control mode. This means that you can enter an input command for the ground station to move to. This will be used for calibration.

To calibrate, the Sun must be in view above the horizon. Use the arrows on the front of the motor controller to point at the Sun, pointing the crosshairs of the calibration scope directly at the center of the Sun. Recalculate the Sun azimuth and elevation angles on the ground station GUI by clicking ?????. Then, switch the motor controller into pointing angle mode by pushing the “Recalculate Sun Position” button, and using the arrow keys set the current pointing angles to the known location of the Sun, displayed on the GUI. Switch the motor controller back into control mode by pushing ?????. Finally, click the “Retrieve Motor Angles” button on the GUI to set the GUI angles to the actual pointing angles of the motor. Now the ground station is calibrated and ready to track a satellite.

To track a satellite, the ground station must first be put into program tracking mode by clicking the “Program Track” radio button. This makes it so you can’t move it manually with the azimuth and elevation inputs. Then, if SWAS is not the desired satellite to track, enter the new TLE location in the “TLE File:” entry. Finally, once everything is set up, click “Start Tracking” to begin sending commands to the motor controller of the current satellite position. The upcoming pass frame can be used to see the next three upcoming passes for any given satellite to know when a good pass to track occurs. If the TLE file is changed, click “Recalculate Passes” to update the upcoming passes for the new satellite.

If something happens and the motor is on a path where something bad could happen, such as bad cable wrap or an uncalibrated motor moving to an undesired position, the red “Stop Motor Motion” button can be used to stop the motor in its tracks. It is a good idea to click the “Retrieve Motor Angles” button after to ensure that the motor controller and software are on the same page.

To run the signal processing software, type ./ARGUS_Signal_Reception.py while in the ARGUS folder. This will write the incoming data to a file and display in a GUI. Be careful to not start it until the satellite is above the horizon or else the data file will have a ton of meaningless data at the beginning.

Setting up a NUC with Ubuntu 18.04

<u>To ensure software is updated and to install python and git:</u>

```
sudo apt update && sudo apt upgrade
sudo apt-get install python3 git
```

<u>Setting up the tracking GUI:</u>

```
sudo apt-get install python3-pip
pip3 install numpy
pip3 install pillow
```

pip3 install matplotlib==2.2.3
pip3 install ephem
pip3 install pyserial
sudo apt-get install python3-tk
Cd ~/Documents/
git clone https://github.com/addodge/ARGUS
cd ARGUS/
Directory MUST include:
argus.py
argusUtils.py
ARGUS.qth
ARGUS_logo.gif
ARGUS_logo.png
setup.py
SWAS.tle
sudo chmod 755 argus.py
sudo python3 setup.py install

Setting up the Signal Processing:

sudo apt install gnuradio cmake
Libiio:
git clone https://github.com/analogdevicesinc/libiio.git
cd libiio/
cmake -DCMAKE_INSTALL_PREFIX=/usr .
make
sudo make install
cd ..
Libad9361-iio:
git clone https://github.com/analogdevicesinc/libad9361-iio.git
cd libad9361-iio/
cmake -DCMAKE_INSTALL_PREFIX=/usr .
make
sudo make install
cd ..
Gr-iio:
git clone https://github.com/analogdevicesinc/gr-iio.git
cd gr-iio/
cmake -DCMAKE_INSTALL_PREFIX=/usr .
make
sudo make install
cd ..
sudo ldconfig

Setting up static IP address over ethernet port:

Static IP must set on both user computer and ground station computer to be able to ssh over an ethernet cable.

On the user computer using Ubuntu 18.04 - add the following lines of code to the end of the /etc/network/interfaces file, replacing ‘enp2s0’ with the name of the ethernet port:

```
auto enp2s0
iface enp2s0 inet static
address 192.168.1.151
netmask 24
```

Setting a static IP address on the user computer using other operating systems was not tested. You must set a static IP address on the ethernet port to 192.168.1.151 with a netmask of 24.

On the ground station computer (Intel NUC), we also set a static IP address. Add the following lines of code to the /etc/network/interfaces file, replacing ‘eno1’ with the name of the ethernet port. A monitor, keyboard, and mouse is required for this part but not for anything else once it is possible to secure shell login to the computer.

```
auto eno1
iface eno1 inet static
address 192.168.1.150
netmask 24
```

Now, you can ssh into the ground station computer using the following command. Our user on the ground station is ‘argus’ and password is ‘skobuffs’, but on a different computer it can be replaced with any username. The -Y option allows X11 forwarding so that graphical user interfaces can be forwarded over the connection and displayed on the user computer.

```
ssh -Y argus@192.168.1.150
Password: skobuffs
```

I set an alias on the user computer to make this easier and not have to remember the specific IP address with the following command in my ~/.bashrc file:

```
alias sshargus="ssh -Y argus@192.168.1.150"
```

Upgrading PLUTO SDR

To get a shell for Pluto SDR:

```
ssh root@192.168.2.1
Password: analog
```

To fool SDR into thinking the range is 70MHz-6GHz:

```
fw_setenv attr_name compatible
fw_setenv attr_val ad9364
reboot
```

Close shell, ssh again (key changes, remove from known hosts):

```
ssh-keygen -f "/home/addodge/.ssh/known_hosts" -R "192.168.2.1"
```

ssh root@192.168.2.1
Password: analog

To make pluto use both processors:

fw_setenv maxcpus
pluto_reboot reset