Requirements

- 50GB+ of disk space for dual booting or virtualizing (VM) Ubuntu 22.04
    - 8GB flash drive if dual booting for installation media
- Internet access

Foreword

This guide will assume you are dual booting or installing Ubuntu as your primary operating system. Care must be taken not to overwrite your existing operating system and/or files. If you are unfamiliar with dual booting and the risks, do not continue. WSL2 is a known alternative to dual booting or a separate virtual machine; however, it has been found to be buggy and a less wholesome experience.

General Recommendations

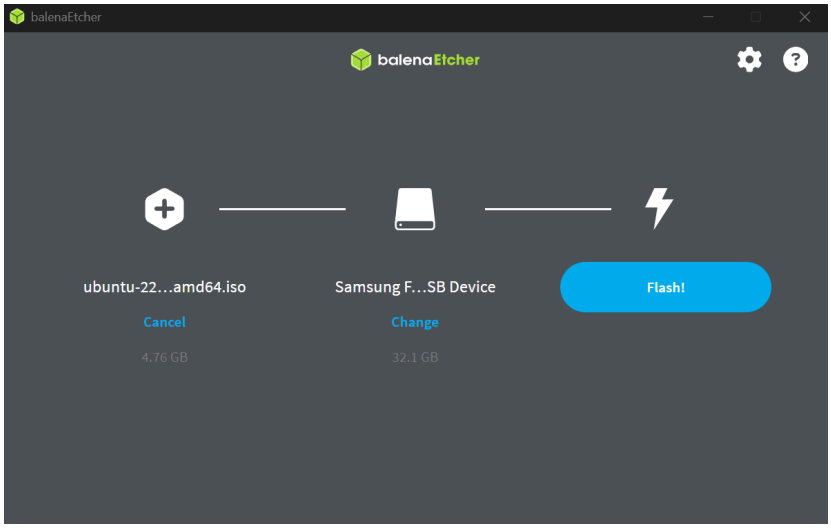
- In Ubuntu, make sure you “eject” any USB devices as Ubuntu might use buffered writing and cause issues if not safely removed
- Do not use spaces in filenames or directories
- Use ethernet

Preparation

- Download the “ubuntu-22.04.5-desktop-amd64.iso” file from https://releases.ubuntu.com/jammy/
    - Ubuntu 22.04 is verified as working. Later versions such as 24.04 are known to cause issues. Only stray away from 22.04 if there is a specific reason
    - Minor distro changes such as Kubuntu might work. Kubuntu 22.04.5 is verified working.
- Download BalenaEtcher or your favorite software to create installation media

Installation Media Creation and Ubuntu Installation

- Follow the instructions with BalenaEtcher to create installation media using your previous “ubuntu-22.04.5-desktop-amd64.iso” file. This will create a USB drive that is bootable that will be used to install Ubuntu 22.04

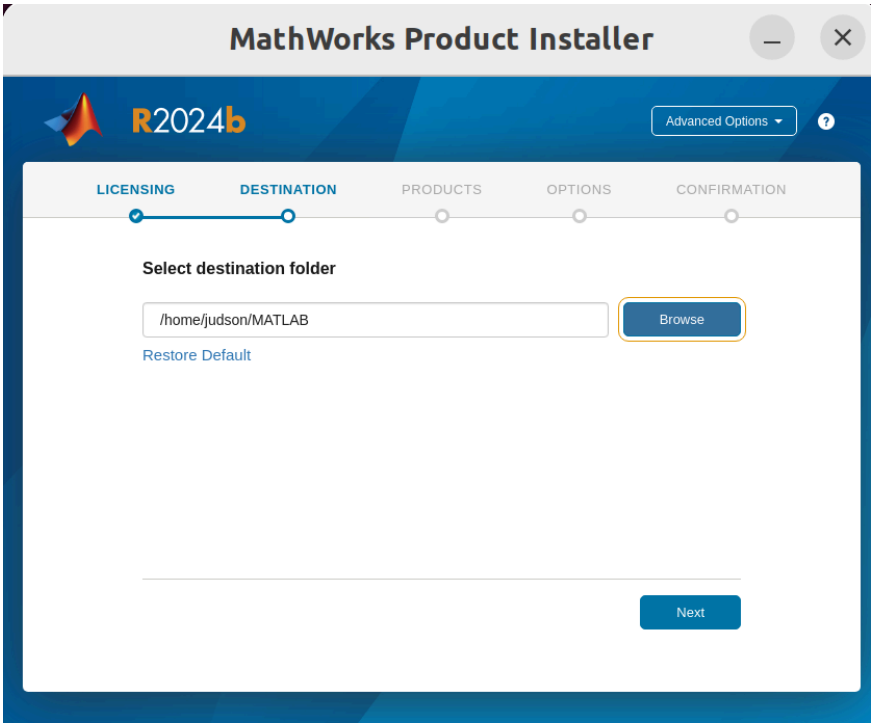
- 

- After the flashing process has ended, remove your USB drive
- Power off the system that is desired to have Ubuntu installed (target computer)
- Plug in the USB drive into your target computer
- Boot the target computer while repeatedly pressing the “DEL”, “F2”, “F10”, or other key to enter the BIOS/UEFI.
- From the BIOS, boot to the USB drive
- Follow the instructions to install Ubuntu.
    - DO NOT select “minimal installation” as this is known to cause issues. Make sure that “Normal installation” is selected
    - It is recommended to enable “Download updates while installing Ubuntu”
    - It is recommended to enable “Install third-party software for graphics and Wi-Fi hardware and additional media formats”
- When the installation of Ubuntu is complete, restart the target computer and remove the USB drive
- Boot into Ubuntu
- Ubuntu Pro is not required or recommended
- It is recommended to “update” applications if they appear. However, DO NOT “upgrade” anything
- It is recommended to install Visual Studio Code from the “Ubuntu Software” store

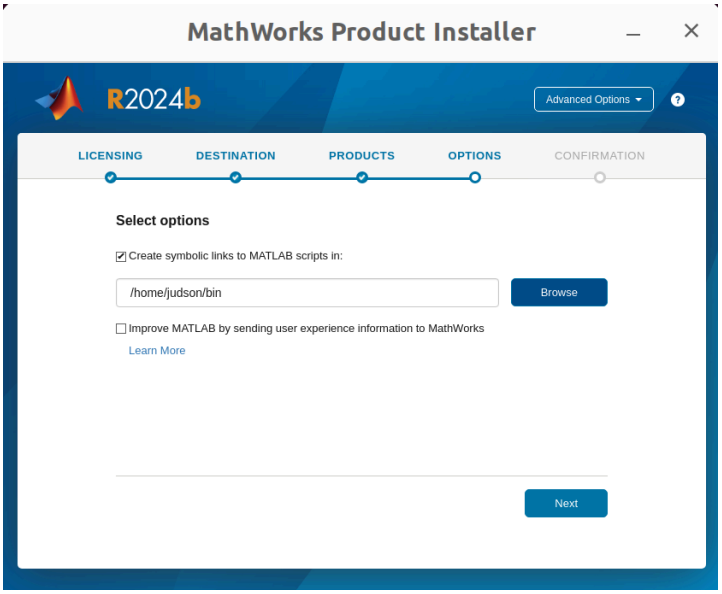
MATLAB/Simulink Installation

- Download MATLAB 2024b from https://www.mathworks.com/downloads/
    - Ensure that the download is for Linux
- After the download is complete, navigate to the download directory and launch terminal
- Unzip the MATLAB download with “unzip matlab_R2024b_Linux.zip”
    - Make sure not to use other unzippers as they are known to cause issues
- In the same directory, run “./install”
    - DO NOT use “sudo”, even when the next menu denies you access due to your installation path

    - If the installation menu becomes “unclickable”, relaunch the installer and make sure to not click away
        - Change the destination folder to one that your user has access to



- Ensure the following “products” are selected for install
    - MATLAB
    - Simulink
    - Embedded Coder
    - MATLAB Coder
    - Simulink Coder
    - UAV Toolbox
    - Vehicle Dynamics Blockset
- Check the “Create symbolic links to MATLAB scripts in “/home/user/bin”. You might need to create the “bin” directory

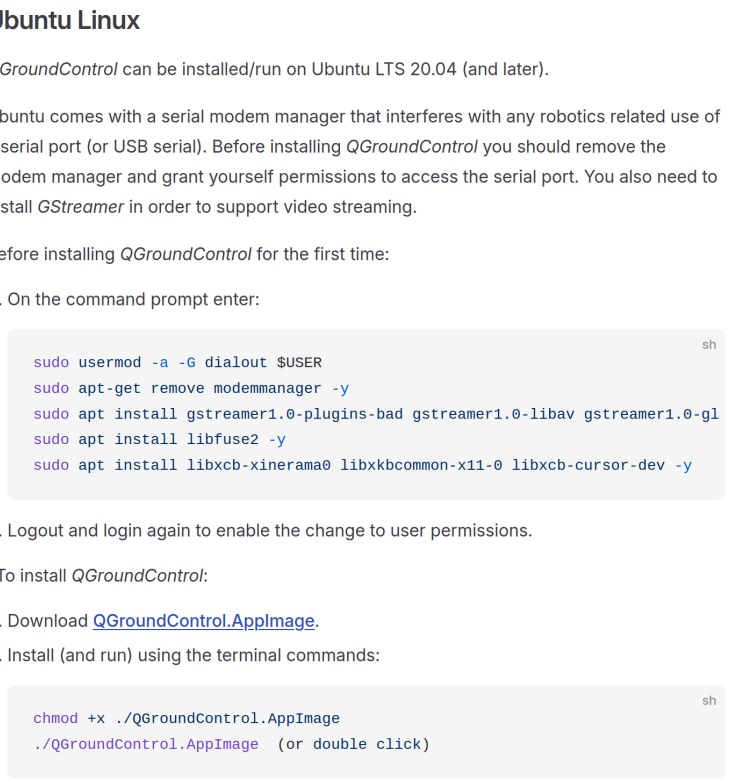


- Begin the installation
- You may ignore the messages about setting up compilers for the Coder products
- Continue to the next section “PX4 Repository Installation and Setup”

PX4 Repository Installation and Setup

- I recommend making a “Pixhawk” directory in your user’s home directory. Inside of it I create a “MATLAB” directory used for MATLAB and Simulink scripts. I also will install PX4 into “Pixhawk/PX4-Autopilot” and QGroundControl into “Pixhawk/QGC”
- Install git with “sudo apt install git”
- Navigate to where you would like to download the PX4 repository in the terminal
- Execute “git clone https://github.com/PX4/PX4-Autopilot.git” to pull the repository
- “cd PX4-Autopilot/”
- “git checkout v1.14.3 -f” to switch to the known working branch
- “git submodule update --init --recursive” to ensure all submodules are correct
    - Note the double “-”. There are 2 hyphens, not 1 dash
- “bash Tools/setup/ubuntu.sh” to install related tools for the building of PX4 and other tools such as Gazebo sim
    - Enter your sudo password as needed
- Download QGroundControl from https://docs.qgroundcontrol.com/master/en/qgc-user-guide/getting_started/download_and_install.html
- I recommend moving QGroundControl into “Pixhawk/QGC”
- Follow the steps to install QGroundControl
    - sudo usermod -a -G dialout $USER
    - sudo apt-get remove modemmanager -y
    - sudo apt install gstreamer1.0-plugins-bad gstreamer1.0-libav gstreamer1.0-gl -y

- sudo apt install libfuse2 -y
- sudo apt install libxcb-xinerama0 libxkbcommon-x11-0 libxcb-cursor-dev -y
- chmod +x ./QGroundControl.AppImage
- ./QGroundControl.AppImage
- Close QGroundControl after it opens



Ubuntu Linux

QGroundControl can be installed/run on Ubuntu LTS 20.04 (and later).

Ubuntu comes with a serial modem manager that interferes with any robotics related use of a serial port (or USB serial). Before installing *QGroundControl* you should remove the modem manager and grant yourself permissions to access the serial port. You also need to install *GStreamer* in order to support video streaming.

Before installing *QGroundControl* for the first time:

1. On the command prompt enter:

```sh
sudo usermod -a -G dialout $USER
sudo apt-get remove modemmanager -y
sudo apt install gstreamer1.0-plugins-bad gstreamer1.0-libav gstreamer1.0-gl
sudo apt install libfuse2 -y
sudo apt install libxcb-xinerama0 libxkbcommon-x11-0 libxcb-cursor-dev -y
```

2. Logout and login again to enable the change to user permissions.

To install *QGroundControl*:

1. Download QGroundControl.AppImage.
2. Install (and run) using the terminal commands:

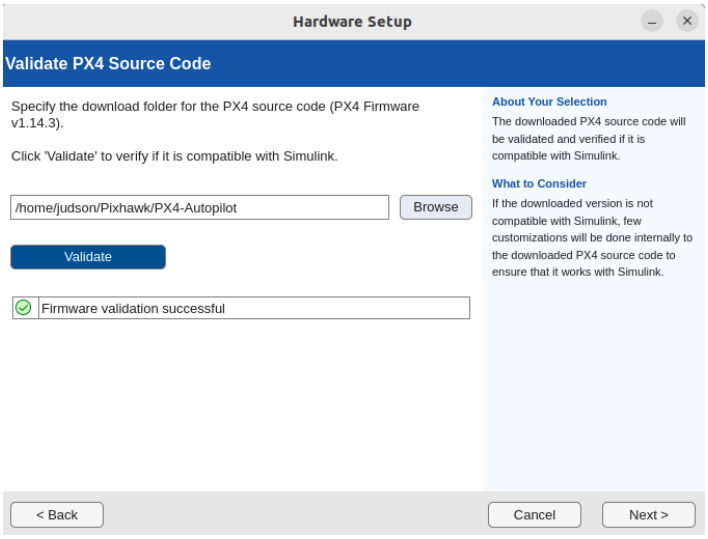
```sh
chmod +x ./QGroundControl.AppImage
./QGroundControl.AppImage  (or double click)
```

- Reboot your computer

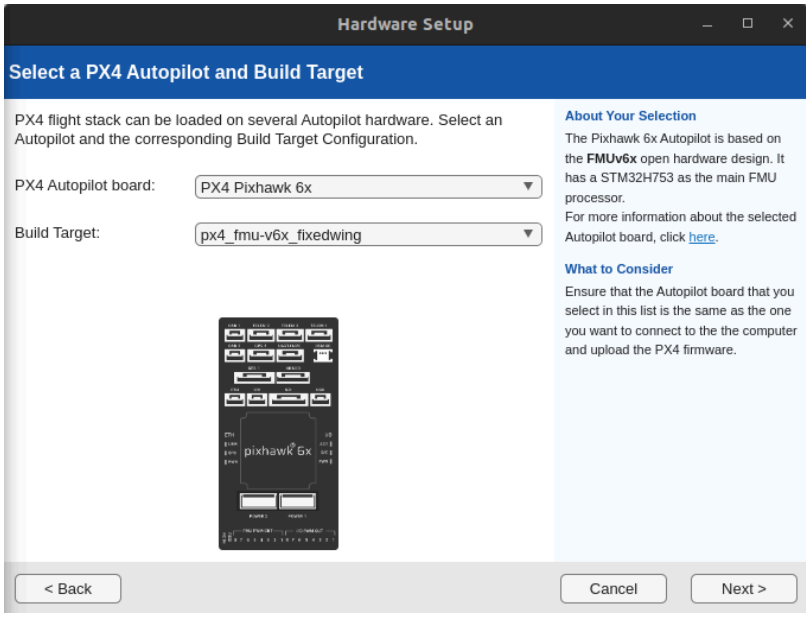
MATLAB/Simulink Further Setup

- Launch MATLAB by opening a terminal in “Pixhawk/MATLAB” or your directory you would like scripts to be generated and typing “matlab”
  - If the system does not recognize the command “matlab”, this means the symbolic links were not set up correctly. Try restarting first and then Google
  - If an error that “cranberra-gtk-module” could not be loaded, perform the following steps
    - “sudo apt-get install libcranberra-gtk-module”
    - “export GTK_PATH=/usr/lib/x86_64-linux-gnu/gtk-2.0”
    - Close MATLAB and reopen with the “matlab” command
- Click “Add-Ons” in the top ribbon
- Search for “PX4”
- Install “UAB Toolbox Support Package for PX4 Autopilots”
- When you get the “Installation Complete” screen, click “Setup Now”
- A new menu will appear under the title “Hardware Setup”

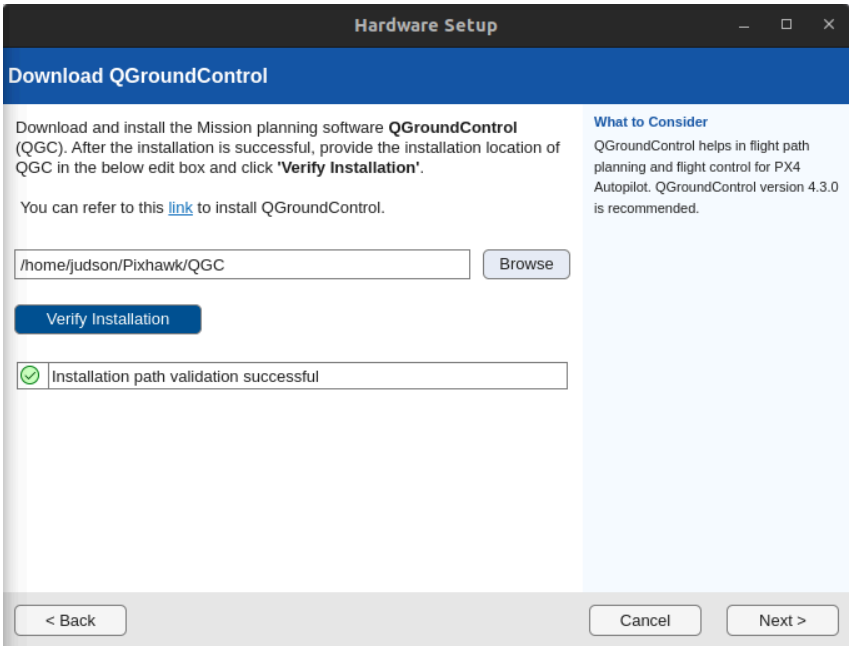
- Direct MATLAB to the directory in which you cloned the PX4 repository



  - Uncheck “Disable multi-copter and fixed-wing controller modules in PX4 Firmware”
  - Change the Board Configuration
    - If MATLAB does not allow you to actually select from the dropdown, proceed by closing matlab and relaunching via terminal with this command: “export ENABLE_QWEBWINDOW=true; matlab”
    - No permanent fix has been found yet



  - Keep the default startup script
  - Point MATLAB to where you installed QGroundControl



- Build firmware
    - You can choose to delete the past PX4 build folder if you do not care for previous builds. I typically choose to enable this feature so that I delete past builds
    - This will take some time. Monitor the MATLAB Command Window. Occasionally, there will be a message in the Hardware Setup window that shows a “Firmware build successful” even though it was not.
    - For the message: “AttributeError: module 'em' has no attribute 'RAW_OPT'”, run the following commands
        - pip uninstall em
        - pip uninstall empy
        - pip install empy==3.3.4
        - Retry the build process
- After the build is completed, connect the Pixhawk to the computer over USB
- Proceed to the next screen and click “Upload Firmware” to the Pixhawk
- After uploading the firmware, click “Get Accelerometer Data” to read accelerometer data and verify working
- Now that MATLAB has been verified to work with PX4 and the tools, one can setup Simulink projects

Adding MODBUS TCP Client Features to Simulink and PX4

- Navigate to your PX4-Autopilot directory
- Grab the custom GSE “msg/” directory from https://github.com/tamusrt/GSE-Miscellaneous/tree/main/Pixhawk_and_GSE/PX4-Autopilot

    - Note that this is not the complete PX4 repository. It is simply a “msg/” directory that contains the custom .msg files and the CMakeLists.txt
- Go to msg/
- Copy the files “PlcCoilsRead.msg”, “PlcCoilsWrite.msg”, “PlcDiscreteInputs.msg”, “PlcHoldingRegistersRead.msg”, “PlcHoldingRegistersWrite.msg”, “PlcInputRegisters.msg” into the msg/ directory of your PX4-Autopilot
- Open msg/CMakeLists.txt and add the following entries into the “set(msg_files…” set
    - PlcCoilsRead.msg
    - PlcCoilsWrite.msg
    - PlcDiscreteInputs.msg
    - PlcHoldingRegistersRead.msg
    - PlcHoldingRegistersWrite.msg
    - PlcInputRegisters.msg

```
        VehicleStatus.msg
        VehicleThrustSetpoint.msg
        VehicleTorqueSetpoint.msg
        VehicleTrajectoryBezier.msg
        VehicleTrajectoryWaypoint.msg
        VtolVehicleStatus.msg
        Wind.msg
        YawEstimatorStatus.msg
        SimulinkCustomMessage.msg
        PlcDiscreteInputs.msg
        PlcInputRegisters.msg
        PlcCoilsRead.msg
        PlcCoilsWrite.msg
        PlcHoldingRegistersRead.msg
        PlcHoldingRegistersWrite.msg
)
list(SORT msg_files)
```

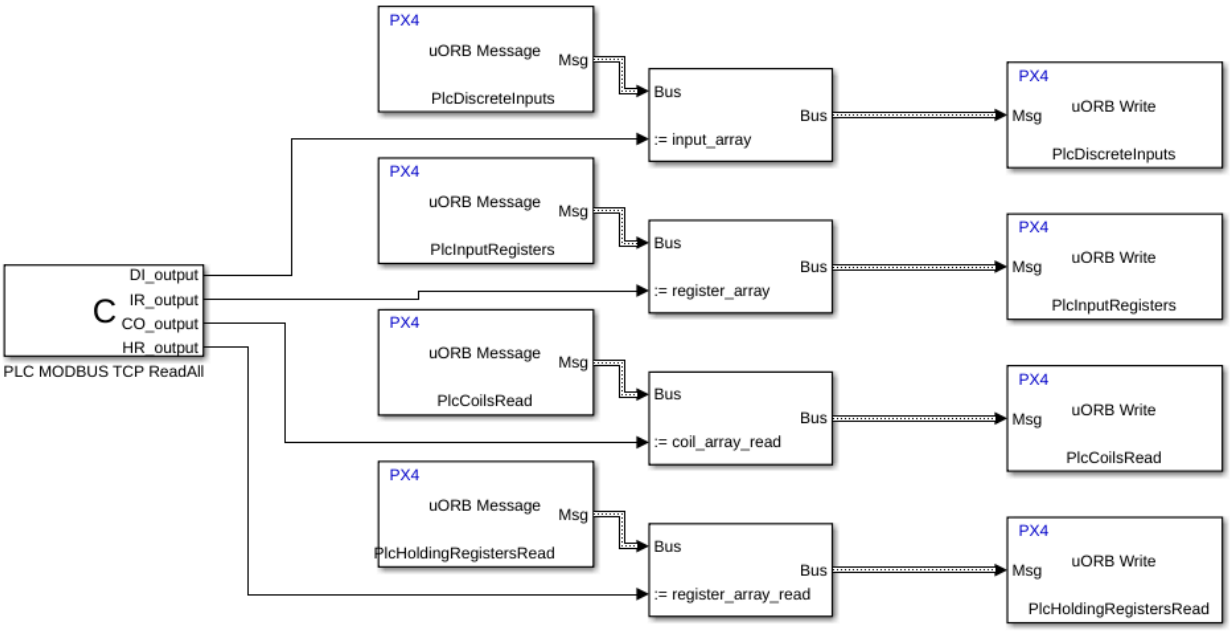
- In MATLAB, go back to Add-Ons and Manage Add-Ons to go back through the Hardware Setup process for PX4 Toolkit
    - This is necessary to rebuild PX4 with the new uORB message definitions to show up in Simulink
    - Follow the same steps as before
- After the rebuild is complete, you may go into Simulink.

MODBUS_TCP_Client_Template

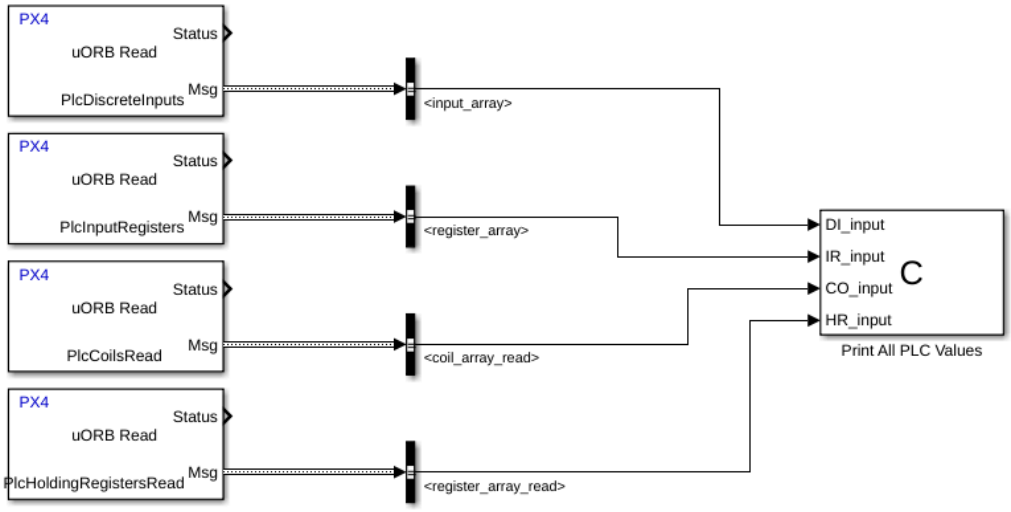
WORK IN PROGRESS

- Grab the GSE files from https://github.com/tamusrt/GSE-Miscellaneous/tree/main/Pixhawk_and_GSE/MATLAB
- Open up the “MODBUS_TCP_Client_Template/” directory. In there, there are the following files
    - modbus_tcp_client.h
    - modbus_tcp_client.cpp
    - MODBUS_TCP_Client_Template.slx
        - This is the Simulink model. Open up this file

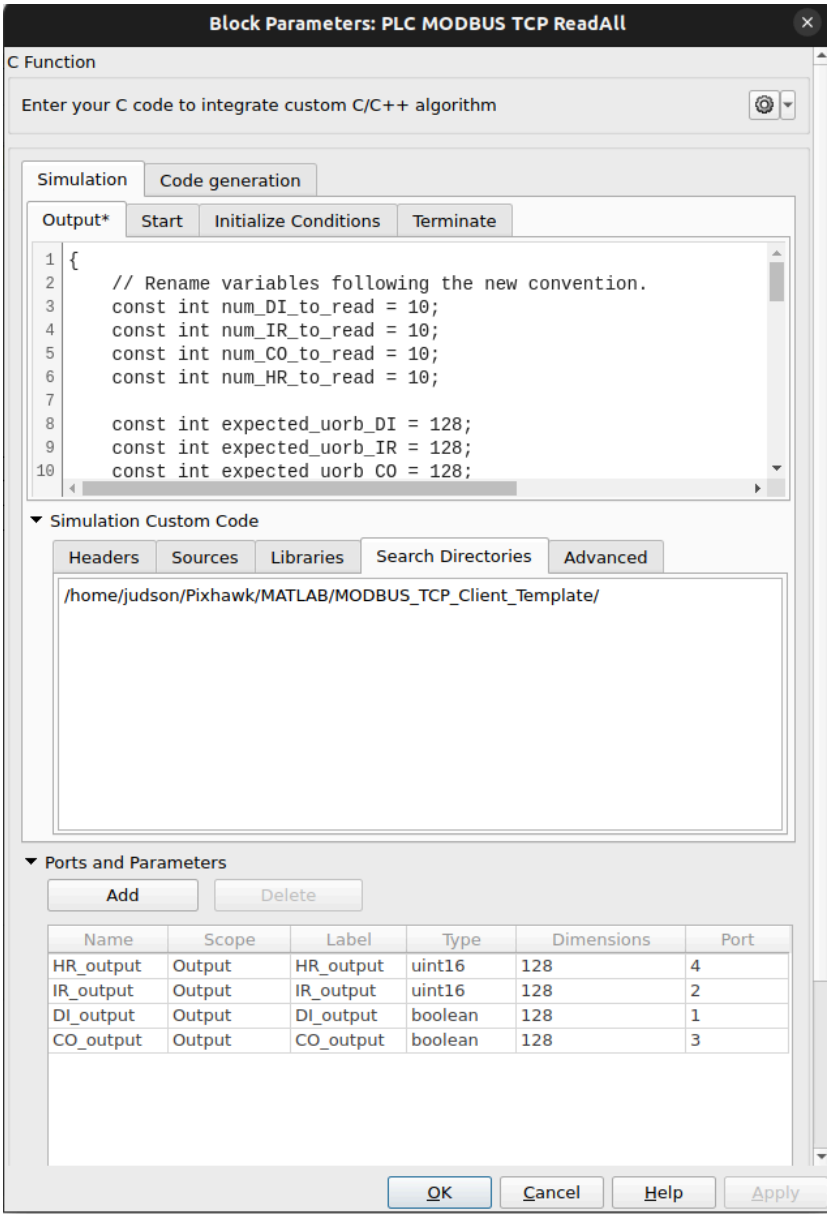
- In the template model, there will already be blocks setup. There are 2 sections
    - Upper Section: MODBUS reading / uORB writing section



    - Lower Section: Demonstration of reading the uORB topics that are connected to the MODBUS TCP Client



- In the upper section, double click on the “PLC MODBUS TCP ReadAll” block to open up its menu
- Navigate to “Search Directories” tab under “Simulation Custom Code”
- Change the directory shown to the directory that holds the “modbus_tcp_client.h” and “modbus_tcp_client.cpp” files.



- Build the project and look for errors