Software dependencies:

-Must have smbus python package installed

-must have ROS2 Humble installed

Hardware dependencies:

-Needs to be running on raspberry pi

-GPIO 2 (SDA) and GPIO 3 (SCL) must be connected to an I2C device (preferably the PWM driver on the modularis main board) before running or else the pwm_control_node will return an error

How to run the two necessary nodes:

-There are two nodes: **duty_cycle** and **pwm_control_node**. Duty_cycle simply takes in a duty cycle keyboard input ranging from 0.0 to 1.0. pwm_control_node subscribes to the duty_cycle topic to then send that command to the PWM driver chip to determine the pulse output from the PWM driver. For example, a duty cycle of 0.6 will correspond to an output square wave with a 60% positive width.

-The launch file launches both nodes and opens the duty_cycle node in another terminal so that the user can input the duty_cycle. This does not seem to work well when ssh into raspberry pi from another device, so for now it is best to run to just run both nodes in two different terminals rather than using the launch file.

Steps:

-Open two terminals

-Ensure device you are ssh-ing from is connected to eduroam wifi (or whichever wifi raspberry pi is currently connected to, it should be default eduroam currently)

-Type “ssh aprilab@10.138.160.34” into both terminals. It should then prompt you for the password, which should be “apr1lab”

-In both terminals, cd ros2_ws then type “source install/setup.bash” and “source /opt/ros/humble/setup.bash”

-In one terminal, type “ros2 run pwmdriver duty_cycle” to run the duty_cycle node. Once it starts running, you can enter the desired duty_cycle * and press enter.

-In the second terminal, type “ros2 run pwmdriver pwm_control_node” to start the pwm_control_node node. This node will publish the I2C commands to the PWM driver chip.

-Press ctrl + c in both terminals to stop the nodes

*When first starting the duty_cycle node, if the main board is connected to the thrusters at that moment then wait until five beeps are heard from the ESC. The first three indicate each phase of the thruster is recognized, with the last two indicating that the initialization pulse is sensed. The pwm_control_node first sends an initialization pulse of 1500us (0.293 duty_cycle) which is required to initialize the ESC. After the beeps are heard to indicate successful initialization, send duty cycle values above 0.3 for higher thruster speeds. Duty cycle value of 0.293 should result in thrusters stopping, but this should be confirmed first

How to connect thrusters to main board and raspberry pi:

-Raspberry pi should connect to header pins on main board using ribbon connector

-ESC signal and ground (white and black wires) should connect to thruster pins on the main board

*Currently the pwm_control_node is configured to only have the PWM driver chip send PWM signals for thruster 0 and thruster 1 but that can be changed in the code by changing line 22 and 27 in pwmdriver/pwmdriver/pwm_control_node.py