**GEOT Software Module  
Hardware**

The hardware consists of two Raspberry Pi (RPi) computers, each connected to a breadboard containing various electronic components. These boards are mostly independent, however there are three wires connecting the two boards.

The programs running on each board are mostly independent, and may be stopped or started at any time, regardless of what the other board is doing. There are 2 caveats to this: First, flap position, which is read by both modules, requires power from Module 2. Power is enabled when Module2IO.py is run, and remains on as long as Module 2 is powered.

The second caveat is the airspeed warning, which is produced by Module 1 and used by Module 2. This, and other hardware inputs, can be overridden by setting the variable overrideDefaults to 0 in Module#IO.py

**Software**

Four .py files have been provided for each module (with # replaced by the respective module number, 1 or 2). These files are located in the Module# folder on the desktop:

**Module#.py**  - This is the file to be written by the GEOT students. The existing code should not be modified, to ensure compatibility with the other files. The line #YOUR CODE HERE should be replaced with the code developed by the students. Run this file to run the program. **Module#IO.py** – This file sets up the connections between the RPi and the electrical hardware. It is called by the existing code in Module#.py. It does not need to be modified by the students, however it can be used to provide Module#.py with specific input values for testing/debugging, via the section labelled ‘#Default values’. These values will be used if overrideDefaults is set to 0. **Module#Example.py** – This is a completed example of the code the GEOT students will produce. The students should first attempt to write their own code without looking at this example. It is by no means the only possible implementation, nor the best, but can be referred to if stuck. **Module#Unittests.py**  - This file provides various inputs to the Module#.py file, and checks the output against expected values. This is not an exhaustive test, but should be able to determine if the produced software performs as intended.

The following lines may be entered into a terminal to run the tests:

cd /home/rpi/desktop/Module# (Replace rpi with rpi2 for Module2)

python –m coverage run –m unittest Module#Unittests

This will output the number of failed tests, or an ok message if all tests pass.

python –m coverage html

This produces an html file detailing which lines of code have been accessed during testing.

**Inputs**All inputs to the system are provided via physical components, consisting of a pressure sensor, and 5 potentiometers.

For all potentiometers, rotation clockwise increases the value, counterclockwise decreases.

A small flathead screwdriver should be used to manipulate the potentiometers, via the green slot.

The range of these values has been pre-set, and may exceed the typical range expected for the input.

|  |  |  |  |
| --- | --- | --- | --- |
| **Input** | **Variable** | **Board** | **Physical Component** |
| Pitot Pressure | pitot | 1 | Syringe, connected to pressure sensor on Module 1 board. Press to increase, pull to decrease. Will leak pressure over time. Do not exceed 25 PSI. |
| Static Pressure | stat | 1 | Potentiometer on Module 1 board, furthest from T adapter. Brown wire. |
| Air Density | rho | 1 | Potentiometer on Module 1 board, closest to T adapter. Orange wire. |
| Aircraft Gross Weight | weight | 2 | Potentiometer on Module 2 board, closest to T adapter. Orange wire. |
| Vertical Accelertation | vac | 2 | Potentiometer on Module 2 board, centre. Brown wire. |
| Flap Position | flap | 2 | Potentiometer on Module 2 board, furthest from T adapter. Blue wire. |

