Software Documentation

General Information

GitHub: github.com/rohanmen/SeniorDesign

File System:

```
startup.sh
API/Docs
    /Documentation.pages
    /GPIO_Pi2.png
    /Node/command_template.json
          /database.json
          /server.js
          /node_modules
    /Python/commands.py
            /controller.py
            /server.py
            /data.json
```

API/startup.sh
-starts two scripts Node/server.js and Python/server.js
-start both those scripts in the API directory, else the script won't be able to find the .json files.

API/Docs
-directory containing all documentation for the project

API/Docs/documentation.pages
-current document
-contains software documentation

AP/Docs/GPIO_Pi2.png
-image documenting the pin layout for raspberry pi 2

API/Node/command_template.json
-template for a command received from the API call

API/Node/database.json
-contains all commands queued up but not yet requested by the python server
-commands are loaded from file when server.js starts
-commands are written to this file when server.js is interupted/shuts down to save the unused instructions

API/Node/server.js
-main node file containing the backend code
-RESTful API
-runs on port 8080 (can be changed)
-main call: ip:8080api//pull_wait_push/:psu_id?/:seconds

-example: 192.168.1.1:8080/api/pull_wait_push/1/10
-can add more API calls, just follow the example in the code

API/Node/node_modules
-libraries for the node language
-contains the express framework used to create APIs

API/Python/commands.py
-contains all the functions needed to interphase with the hardware
-uses the GPIO library for the raspberry pi

API/Python/controller.py
-GUI program that allows the user to control the system using a keyboard

API/Python/server.py
-script that pulls the queued commands from the API and converts them into instructions
-uses commands.py to issue all of the commands

API/Python/data.json
-contains the location of each psu based on its ID
-the ID of the psu correspond to its location in the json array
(this is just the current implementation and can easily be changed if needed)

Pin Layout for Raspberry Pi 2

/*INSERT IMAGE HERE*/

Pin Assignments

Linear Actuator
Pin1: 35
Pin2: 37

Track Actuator 1 (used for horizontal movement)
Pin1: 36
Pin2: 38

Track Actuator 2 (vertical movement left)
Pin1: 3
Pin2: 5

Track Actuator 3 (vertical movement right)
Pin1: 11
Pin2: 13

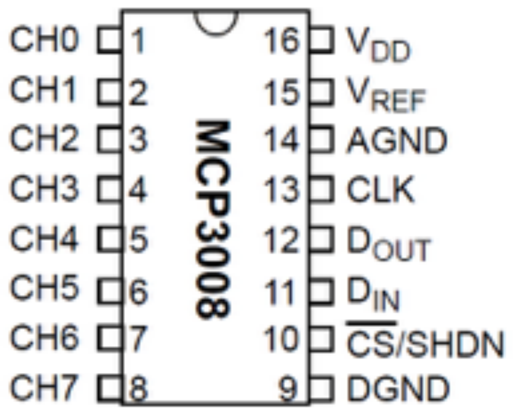
ADC Pins

SPICLK: 23
SPIMISO: 21
SPIMOSI: 19
SPICS: 22

ADC Channels
Linear Actuator Feedback (built in potentiometer): 0
Track Actuator Feedback (string potentiometer): 1
Vertical Feedback: 2 **currently empty**
Current Feedback: 3

Limit Switches Pins for Vertical Movement
Level 0: 29
Level 1: 31

Wiring for ADC



MCP3008 VDD -> 3.3V (red)
MCP3008 VREF -> 3.3V (red)
MCP3008 AGND -> GND (black)
MCP3008 CLK -> #23 (orange)
MCP3008 DOUT -> #21 (yellow)
MCP3008 DIN -> #29 (blue)
MCP3008 CS -> #22 (violet)
MCP3008 DGND -> GND (black)