

Objective of GUI - The GUI for operational cold flows and launch days should be minimalistic, make it easy to control solenoids, easy to update, increase safety, and eventually automate the nitrous oxide fill sequence.

Important Note - The GUI still can't read data from labjack. Our team hasn't figured out a nice way to read data from the labjack with Python OR Javascript. If anyone can get either to work, automated filling can actually make progress!!

Link to demo -

https://origamiaztec.github.io/BrowserBasedSRTGUI/babyHybridGUI-Demo/index.html

Link to github repo with code -

https://github.com/OrigamiAztec/BrowserBasedSRTGUI

Software PreRequisites:

Visual Studio Code for write code in a pretty IDE -

https://code.visualstudio.com/download

PuTTY for diagnosing issues, checking if ethernet or serial is reading correctly - https://www.putty.org/

Install node js to be able to communicate browser with serial and ethernet - https://nodejs.org/en/download/

Install "http" npm package - https://www.npmjs.com/package/http or npm i http
Install "express" npm package - https://www.npmjs.com/package/express or npm i express

Install "socket.io" npm package -https://www.npmjs.com/package/socket.io or npm isocket.io

Install "serialport" npm package - https://www.npmjs.com/package/serialport or npm i serialport

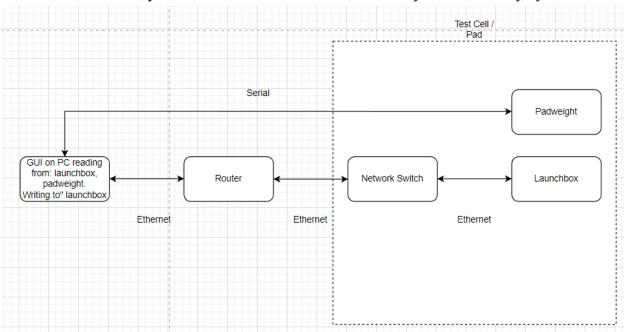
Relevant Youtube videos / tutorials for troubleshooting:

https://www.youtube.com/watch?v=uVnKfNmUuMo&list=FLKBMxJnMFzg-HHeEQJHzYBQ&index=1

https://www.youtube.com/watch?v=-L3oSsjFYh0&list=FLKBMxJnMFzg-HHeEQJHzYBQ&index=2

https://www.youtube.com/watch?v=gQYsUjT-IBo&t=1183s

Overall Hardware Layout - this is what's worked most recently to hot fire baby hybrid -



GUI File Organization:

Main index.html layout and IDs for button locations,
images, references to other pages,
references to font links, and
javascript functions when buttons
clicked

"css" folder - contains .css file that styles all items in html file. Makes everything look pretty.

server.js run this code and then open
"http://localhost:3000/" to run GUI,
this script collects data from serial or
ethernet port using "http" and "serial"
packages, then sends it as either
LaunchBox or PadWeight string data
to the browser using the javascript
"socket.io" package

"images" folder - contains images / logos used in the GUI.

"javascript" folder - contains .js file with functions for responding to button clicks and handing data from serial ./ ethernet. Says what happens when buttons are pressed and how data is handled.

Main Folder

"Public" folder

Javascript Data Processing Summary:

Receive data from launchbox IP address? emit that data packet to animationResponse.js with io.emit("data2", dataChunk.ToString());

Receive data from serial port COM port? emit that data packet to animationResponse.js with 'io.emit("padWeightSerialData", data)'

Receive "data2"? update html/css with document.querySelector('.last_received_text').textContent = "Last Received: ' + data.split("/")[1];

Receive "padWeightSerialData"? update HTML/CSS with runTankWeight = parseFloat(data.split(",")[0]); document.querySelector(".LC2Label2").textContent = runTankWeight + " lbf";

server.js

animationResponse.js

Code Breakdown:

Server.js