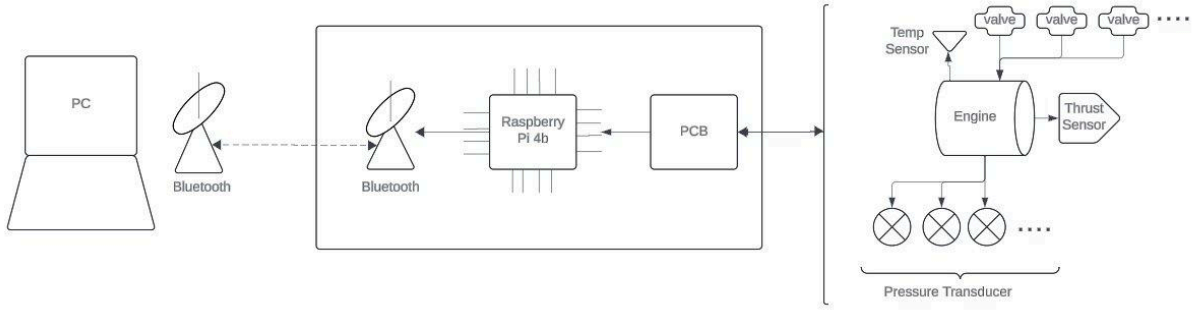
High-Level Technical Overview

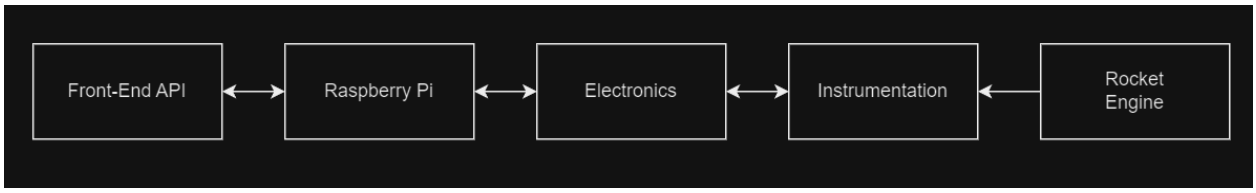
High-Level Electrical System Design

- Highlights the flow of data in the system



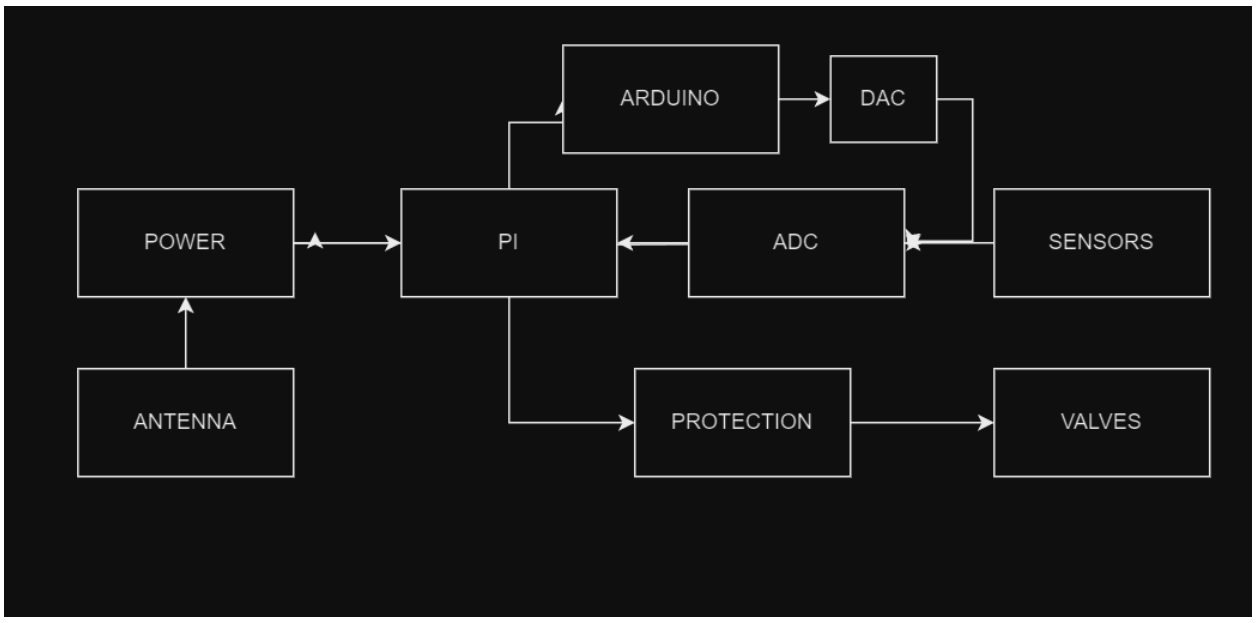
High-Level Block Diagram of Electrical System

- Adds more official terminology



Functional Diagram of Hardware Box

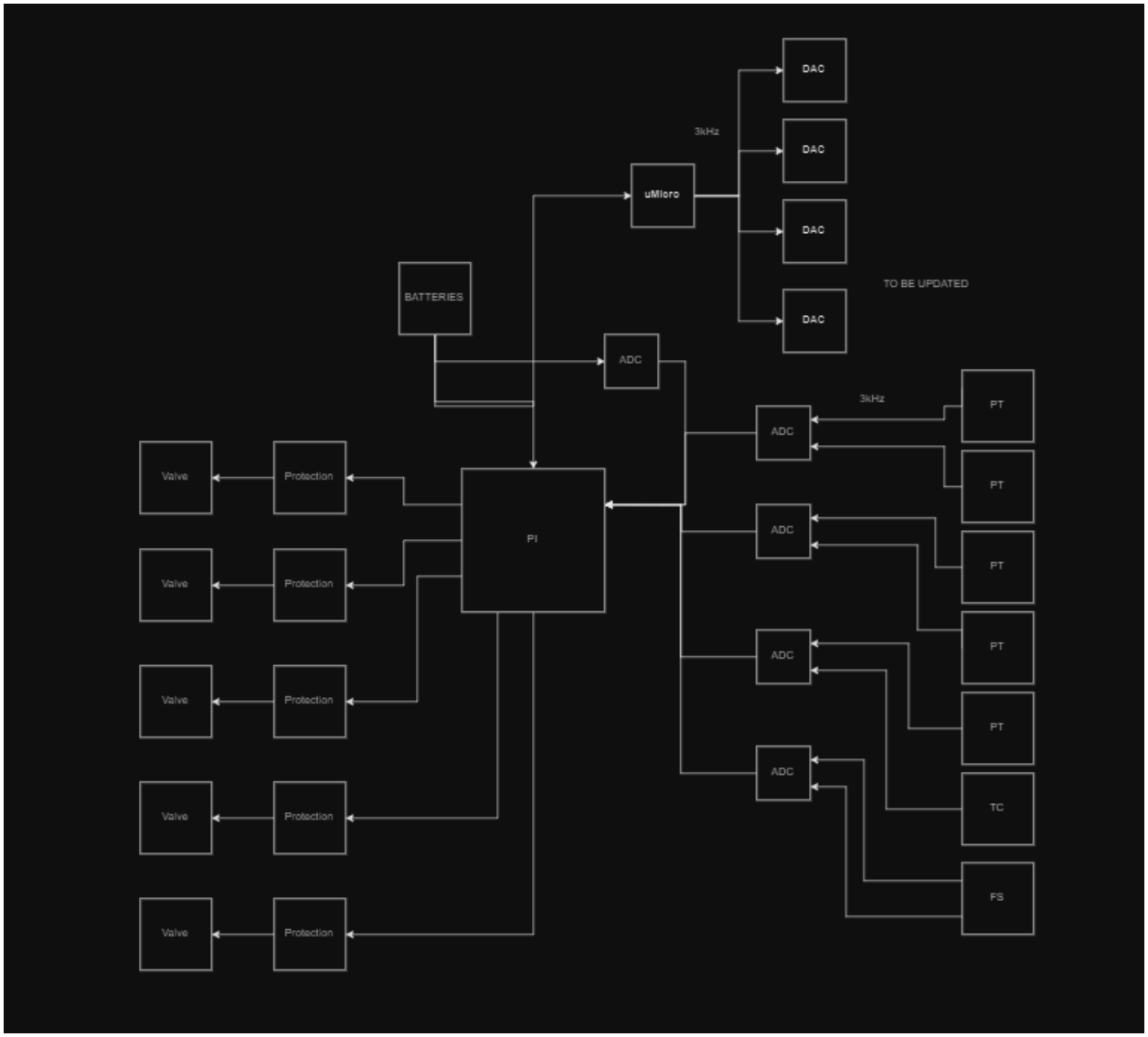
- Highlights the major hardware systems interacting with each other
- Missing are the LED's and Sensor Filters



Box Hardware Details

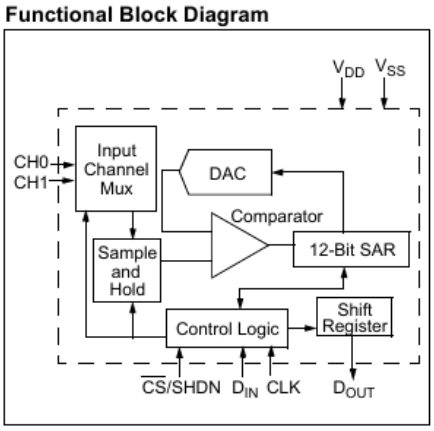
Functional Diagram

Mid-Level Function Diagram

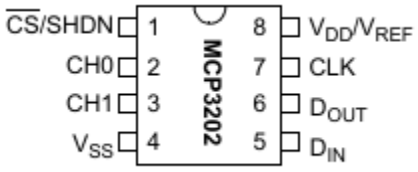
- I do not know the testing and sensor switching for the system correct
- Missing antenna and and testing/sensor switching will be updated
- 

ADC 1 and 2 for Pressure Transducers

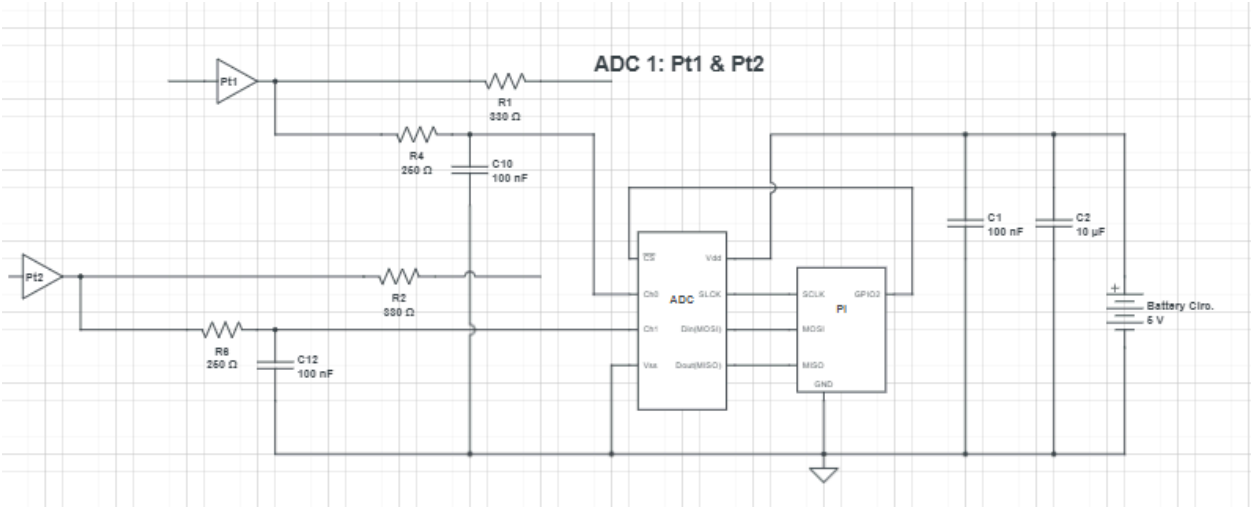
- MCP3202
- SPI
- 12-bit resolution
- SAR
- 5V
- <= 100ksps
- will use PI 5v rail as reference voltage



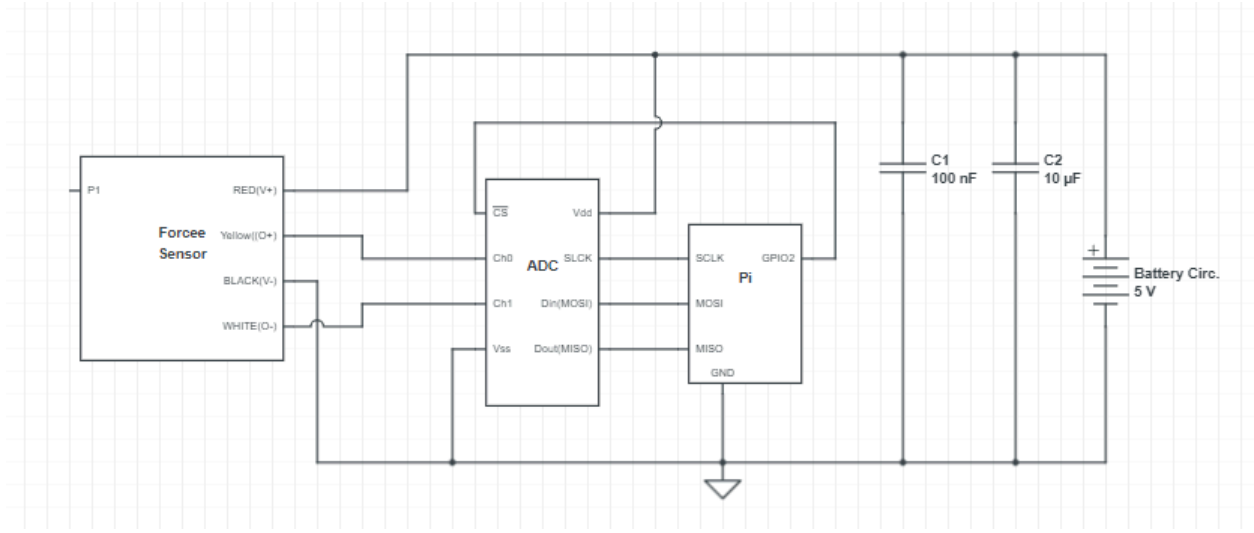
-



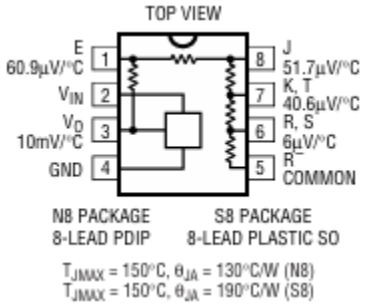
-

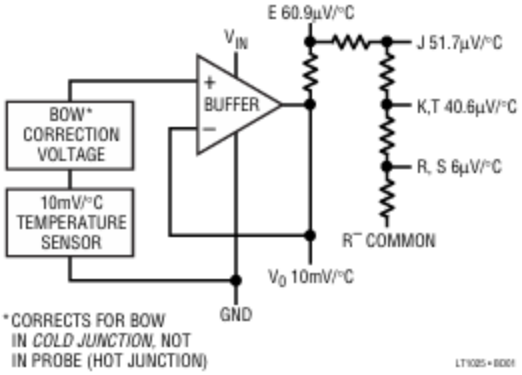


Force Sensor ADC

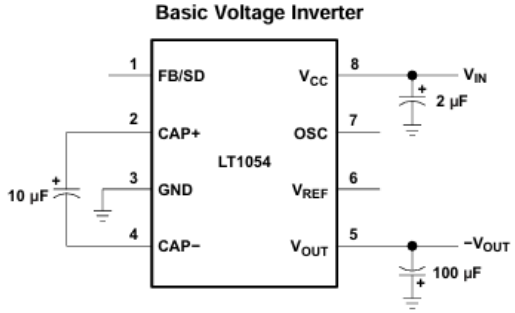


ThermoCouple

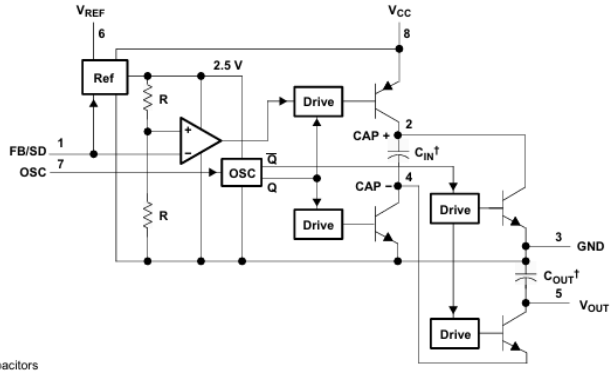
- Cold Junction Compensator (LT11025)
    - Thermocouples work by using 0C as a reference if the system isnt at 0C we need a way to artificially show that 0V is OC and thats a CJC
- 0.5C accuracy
- 10mV/C output
- CJC
    - 



- Inverter

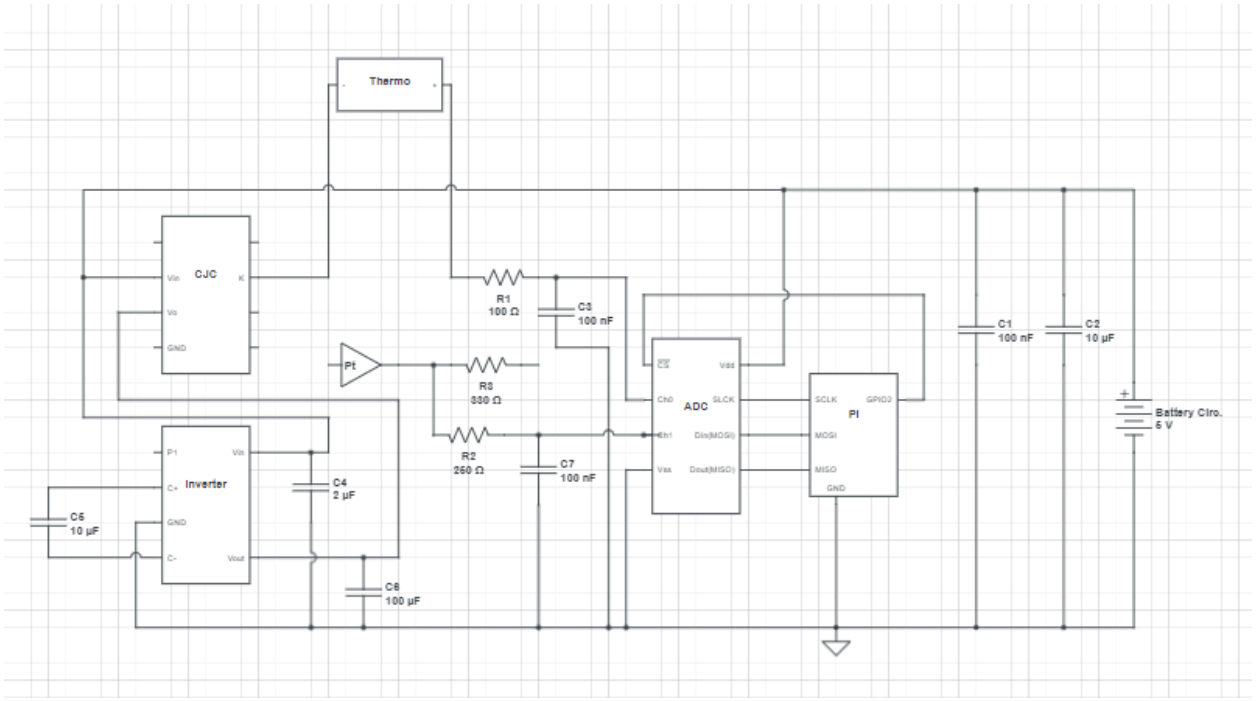


7.2 Functional Block Diagram



† External capacitors

Pin numbers shown are for the P package.

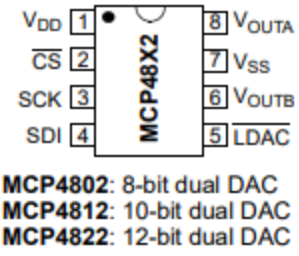


Power System

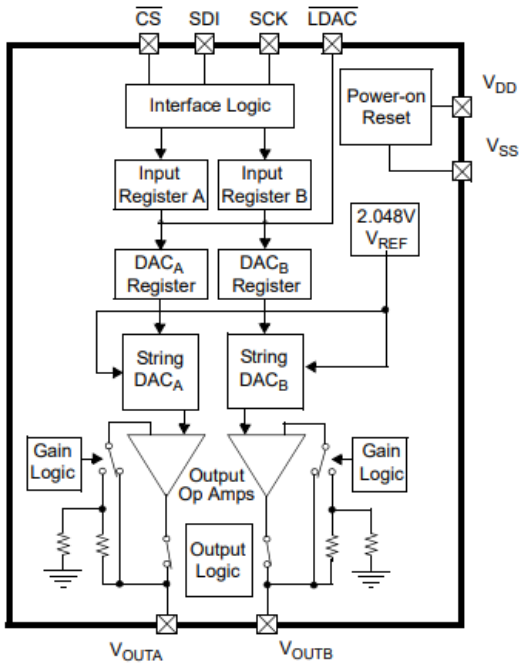
- **TO BE ADDED**

DAC For Testing

- MCP4802
- 12-bit resolution
- 3-wire SPI
- Internal Voltage Reference
- Arduino uMicro

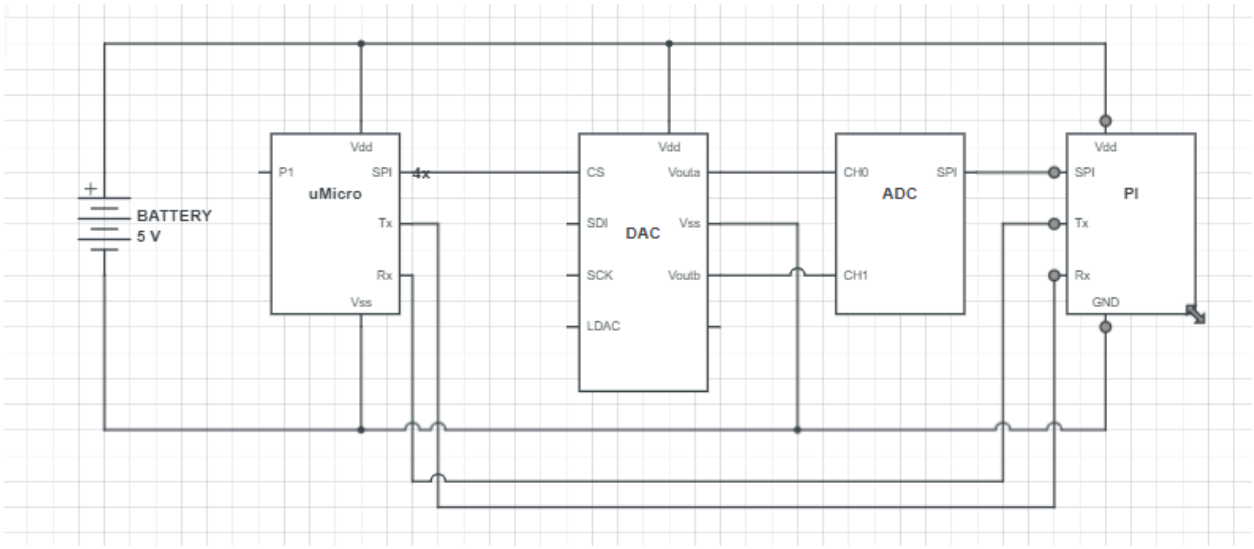
- 

MCP4802: 8-bit dual DAC
MCP4812: 10-bit dual DAC
MCP4822: 12-bit dual DAC

- 

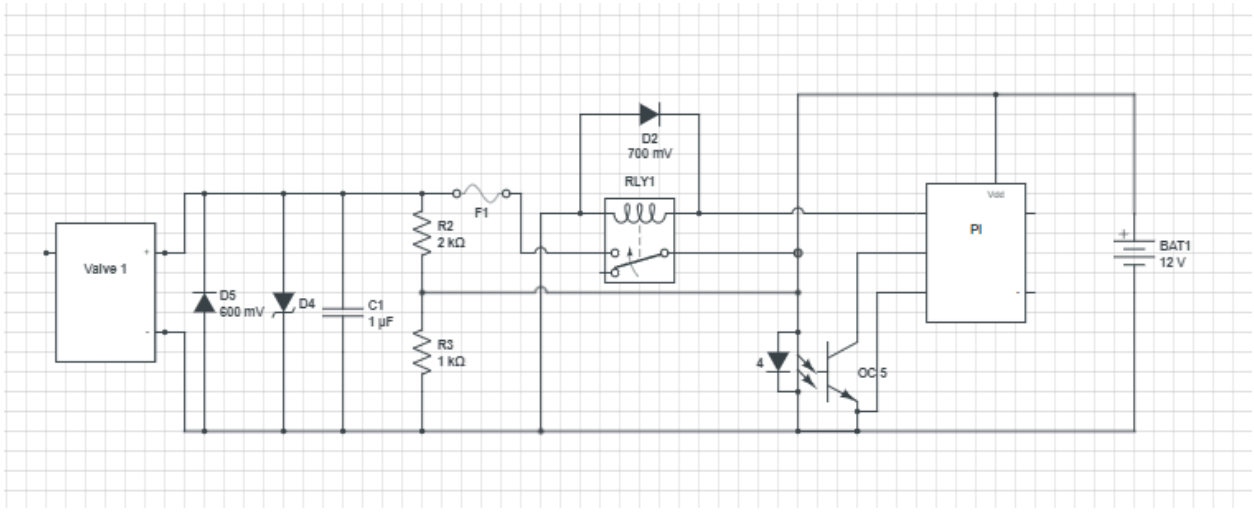
- Details
  - Using a microcontroller to control the SPI lines of the Arduino to control the DAC hardware
  - I want a async way to send data
    - variable pwm to analog converter was not viable
    - Arduino or esp32 does not have enough memory
    - Same SPI network would not have for mux for tx and rx data

    Landed on ucontroller controlling for asynch spi lines and pre flashing with test data



Valve Circuits

- **Look into replacing Relays with MOSFETs**
- **12V**

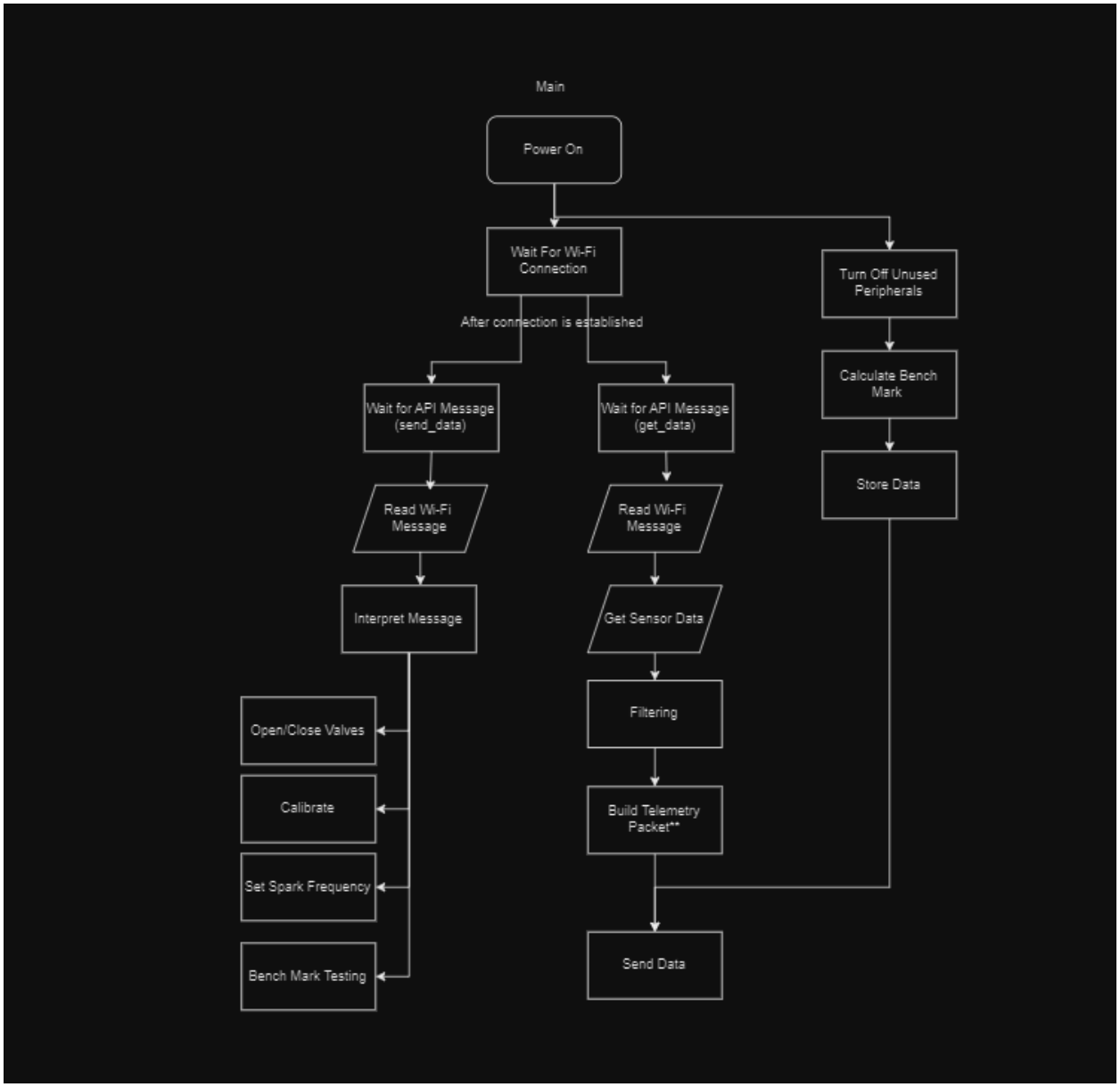


- **24V**
  - **Look into boost converter for 24V valve**

Instrumentation Hardware Details

- **TOBE ADDED**

Software Details

- **High-Level Idea of the Software System**
- 

- OOP Structure
    - Pi Side
        - Client
        - Telemetry/Wifi
        - GPIO
        - PWM

        - ADC
        - DAC
        - Coil
        - Linux
    - Py Side
        - Metrics
        - Telemetry
        - Wi-Fi Host
        - System Health

ADC: MCP3202

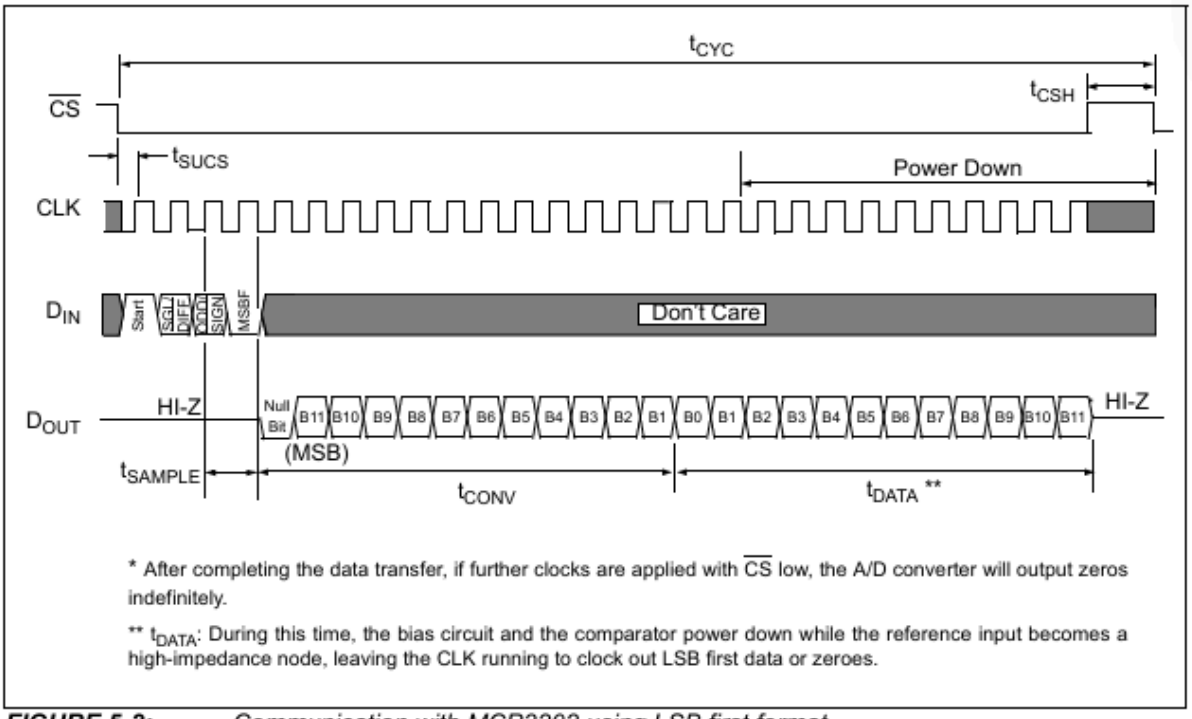


FIGURE 5-2: *Communication with MCP3202 using LSB first format.*

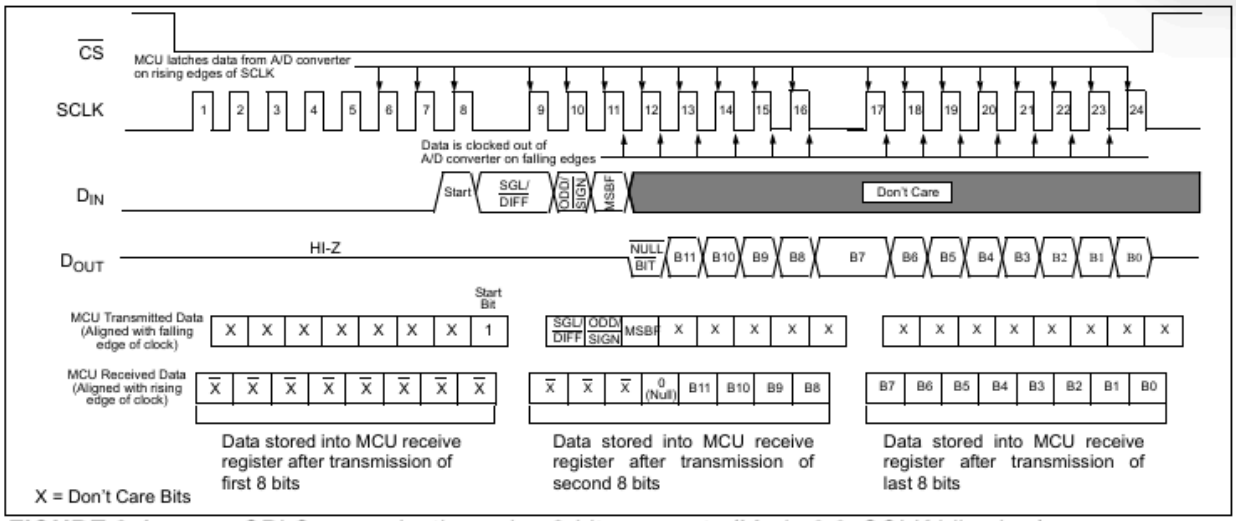


FIGURE 6-1: *SPI Communication using 8-bit segments (Mode 0,0: SCLK idles low).*

TABLE 5-1: CONFIGURATION BITS FOR THE MCP3202

	Config Bits		Channel Selection		GND
	SGL/ DIFF	ODD/ SIGN	0	1	
Single-Ended Mode	1	0	+	—	-
	1	1	—	+	-
Pseudo-Differential Mode	0	0	IN+	IN-	
	0	1	IN-	IN+	

DAC: MCP4802

Where:

bit 15 **A/B:** DAC_A or DAC_B Selection bit
- 1 = Write to DAC_B
- 0 = Write to DAC_A

bit 14 — Don't Care

bit 13 **GA:** Output Gain Selection bit
- 1 = 1x ($V_{OUT} = V_{REF} * D/4096$)
- 0 = 2x ($V_{OUT} = 2 * V_{REF} * D/4096$), where internal V_{REF} = 2.048V.

bit 12 **SHDN:** Output Shutdown Control bit
- 1 = Active mode operation. V_{OUT} is available.
- 0 = Shutdown the selected DAC channel. Analog output is not available at the channel that was shut down. V_{OUT} pin is connected to 500 kΩ (typical).

bit 11-0 **D11:D0:** DAC Input Data bits. Bit x is ignored.

REGISTER 5-1: WRITE COMMAND REGISTER FOR MCP4822 (12-BIT DAC)

W-x	W-x	W-x	W-0	W-x	W-x	W-x	W-x	W-x	W-x	W-x	W-x	W-x	W-x	W-x	W-x
$\overline{A}$/B	—	$\overline{GA}$	$\overline{SHDN}$	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
bit 15															bit 0

Bench Mark Testing

- This is the main framework I want to use to test the overall performance of the system for setting standards and general and data-driven ways to push improvements.
- I want to make a standardized way to test the system as a whole
- Things to be measured and compared
    - Power draw
        - Improve power efficiency
    - Wi-fi speed
        - faster ,stable,reliable wifi connection
    - tx/rx speed
        - Might be synonymous with wifi
    - Sensor accuracy
        - Improve signal of crucial sensor data especially with emi
    - Abort Logic
        - Proper safety measures are working properly
    - Memory and CPU Usage
        - Smaller mem and cpu footprint
    - Temperature
        - Temperature varying performance and limits
    - Security
        - how to measure security performance
    - Filtering
        - How well filtering algos work

Sketch of what the system will look like

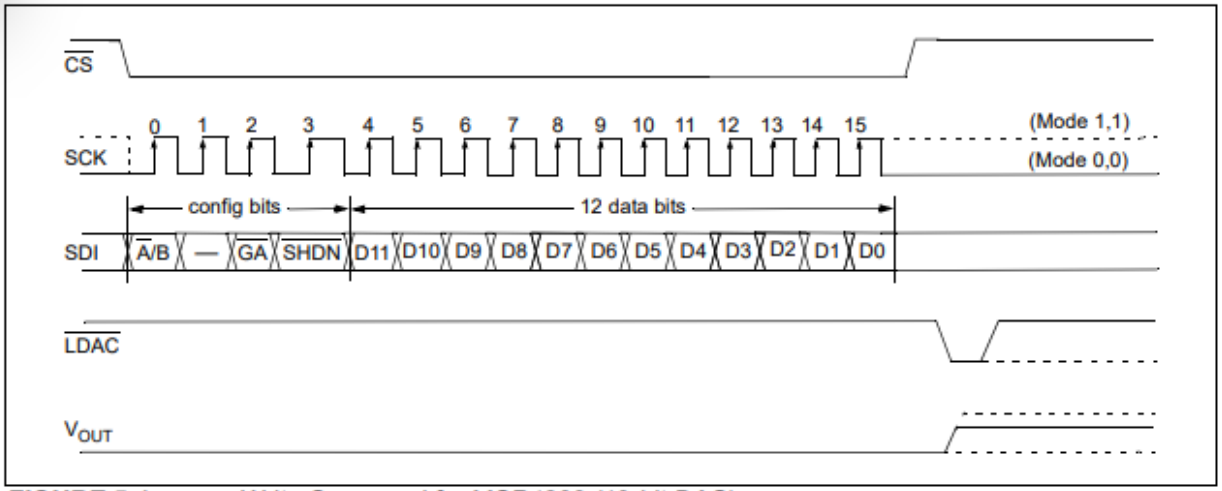


FIGURE 5-1: *Write Command for MCP4822 (12-bit DAC).*