Automated Battery Characterization

Team: sddec20-25
Advisor: Nathan Neihart
Client: Solar Car
Website: http://sddec20-25.sd.ece.iastate.edu/

Joe DeFrancisco, Ben Kenkel, Bryan Kalkhoff,
Connor Luedtke, Ryan Willman, Kyle Czubak

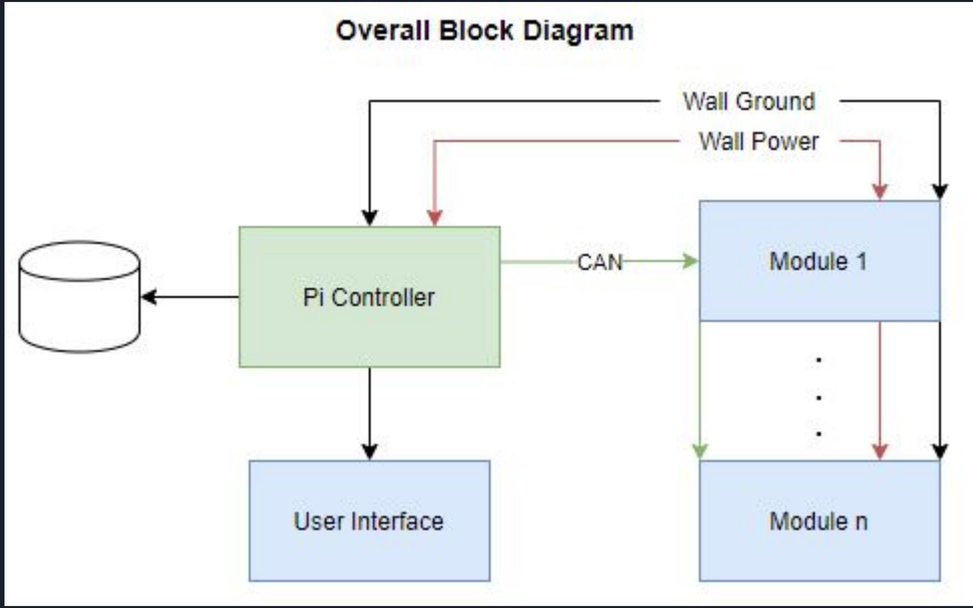
Project Plan

Problem Statement

- Best performance with matched batteries
- PrISUm Solar Car currently does not have an efficient way of characterizing batteries on a battery pack scale.
    - Currently the procedure is to measure the open circuit voltage of each battery and group them based on that measurement.
- Process is time consuming on large scale (1000 or more batteries)
- We will design and build an automated lithium battery characterizer

Conceptual Sketch

- A Raspberry Pi is communicating with 8 battery modules and a web application
  - A single screen is attached to see battery test progress
- 3 Profiles
  - Charge, Discharge, and Characterization
- CAN communication



Requirements

- Functional
  - Characterize 8 lithium-ion batteries safely
  - Store the battery information and a way to analyze data in the future
  - Safely operate unsupervised
  - Be able to connect multiple characterizers
  - Compute the capacity and internal resistance of each of the batteries
- Non-Functional
  - Batteries should not operate outside of temperature specifications as listed in JEITA guidelines
  - Must finish battery characterization in under 24 hours
  - Require minimum user interaction
- Test Program
  - Rated Capacity
  - Internal Resistance

Technical Considerations

- Measurement Accuracy
  - To have valid results, the voltage and current measurements will need to be accurate.
  - Main driver of the requirement is the internal resistance measurement, requiring μV level measurement accuracy.
- Unsupervised Operation
- Thermal Management
  - Full charge and discharge cycles will dissipate a lot of heat. This will need to be modeled so that proper heat sinking and cooling measures can be taken during design process.
- Bus Load
  - Uses multiple buses, I^2C and CAN, and needs to balance the load of both buses

Market Survey

- The current system that solar car owns can only test 1 batteries at a time
- Few off the shelf solutions (one such device CBA HR from West Mountain Radio)
  - This one cost ~$160 for the device itself
  - Does a lot of what our device will do
  - Our device has better voltage accuracy and integrates with Solar Car’s systems better
- Our System
  - Up to 8 batteries
  - Designed with the use of multiple modules in mind
  - Custom database to export data to
- PCB Cost:
  - Physical Board: $50
  - Components for Board: $273
- Other Required Components: $ 50
- **Total: ~$375**

Potential Risks and Mitigation

- Lithium ion batteries are very sensitive to temperature and can be prone to thermal events (ie exploding).
- To limit potential thermal events both hardware and software measures are being used
- During testing we had appropriate equipment including fire extinguishers and buckets of sand available in case of any problems.
- Examples of protections:
  - Shutdown if battery temperature exceeds 60 degrees celsius.
  - Short circuit monitoring and fusing
  - Battery holders are keyed, so they cannot be put in backwards

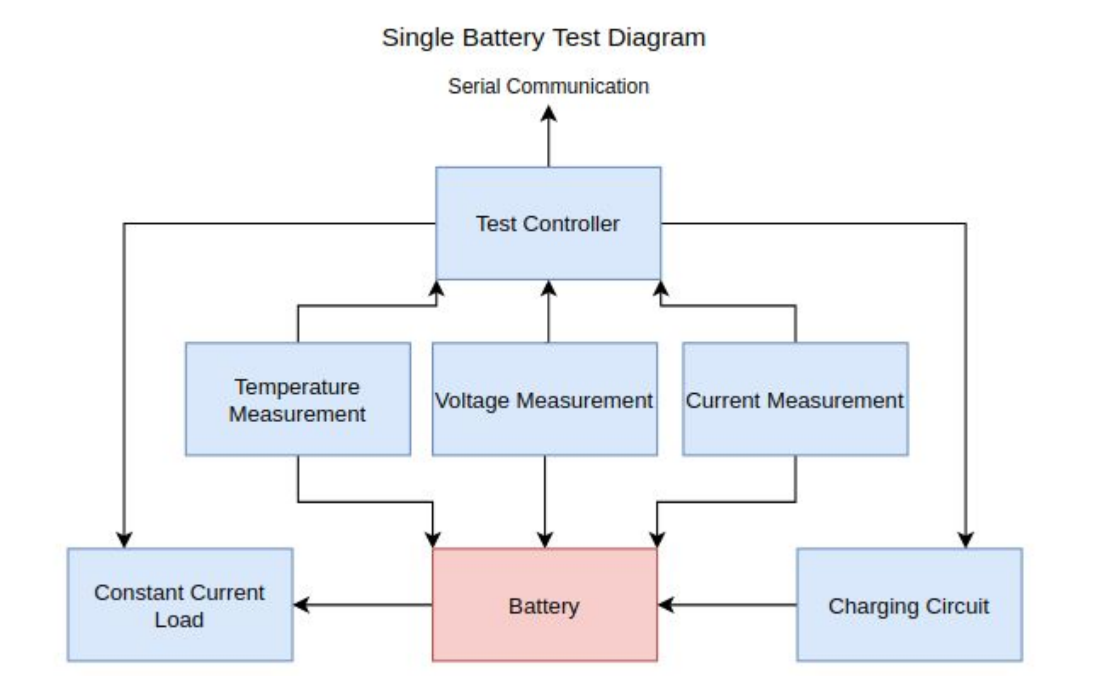
Engineering Standards and Design Practices

- Agile Development
- IEEE 1679.1-2017: IEEE Guide for the Characterization and Evaluation of Lithium-Based Batteries in Stationary Applications
- JEITA Battery Guidelines for Charging and Discharging Lithium Batteries

System Design

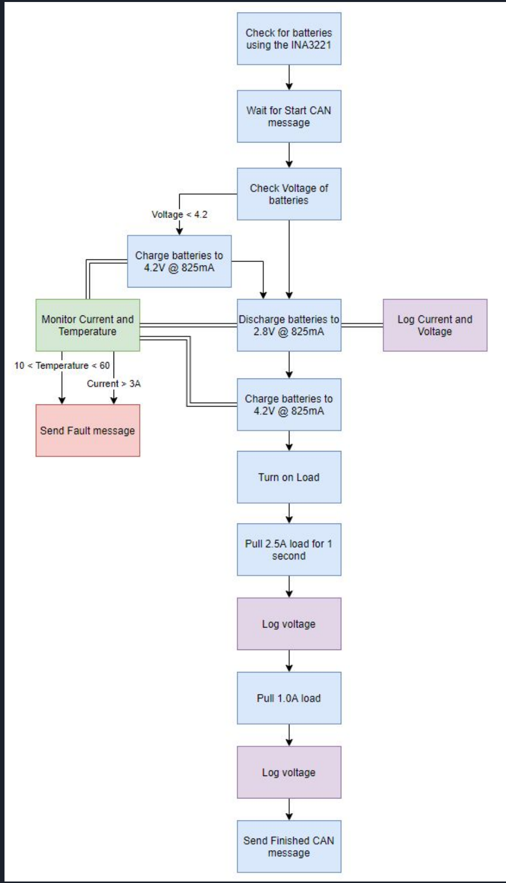
Hardware - Summary

- Used Altium for schematics and PCB layout
- Project was broken down into several groups
  - Power, microcontroller, charge, and discharge
- Pi Communication using CAN
  - CAN is fast enough and have experience
- I2C for on board IC communication



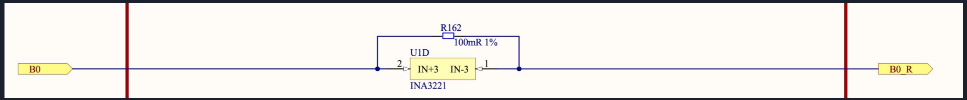
Embedded Flow

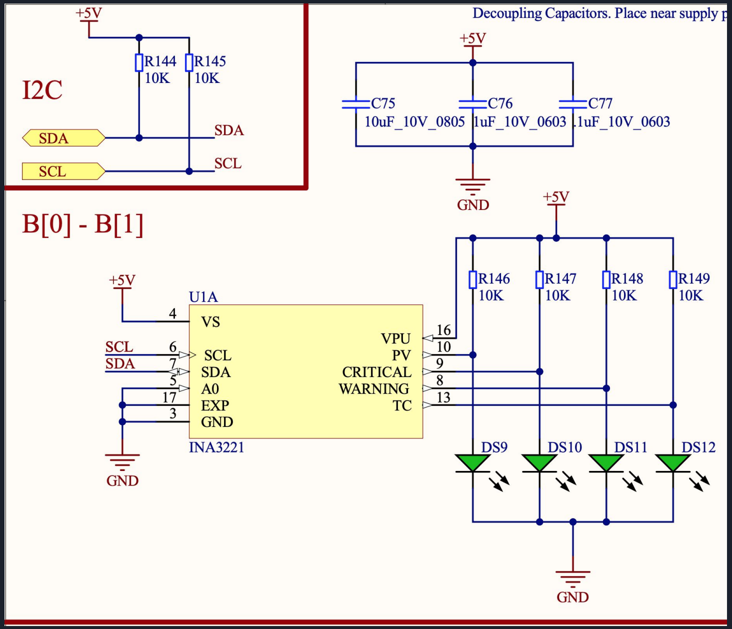
- Using measurement IC to see if a voltage exists
- Always monitor the current and temperature of the batteries
  - Need to monitor voltage during discharge
  - Charger IC monitors voltage during charging
- Discharge rate is same as DAC output
  - Ex. 1V = 1A
- Turn on Load
  - Flip relay to discharge
  - Set DAC output



Current & Voltage Measurement

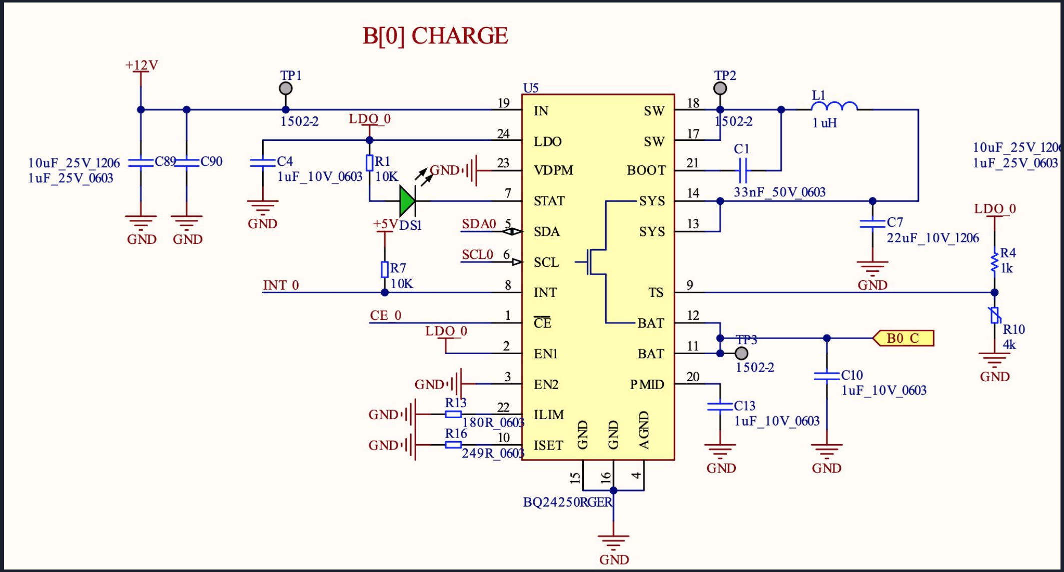
- INA3221
    - 100mOhm sense resistor
- 13-bit internal ADC
- I2C compatible data bus





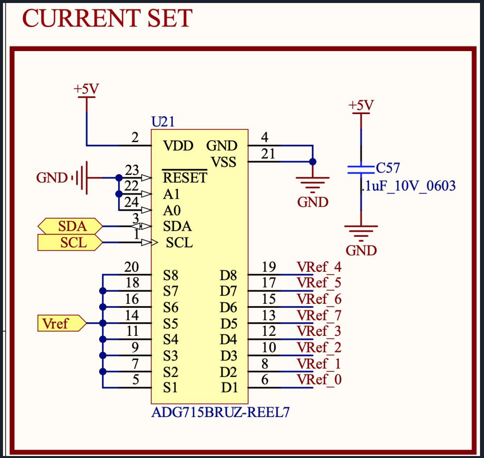
Charge Circuit

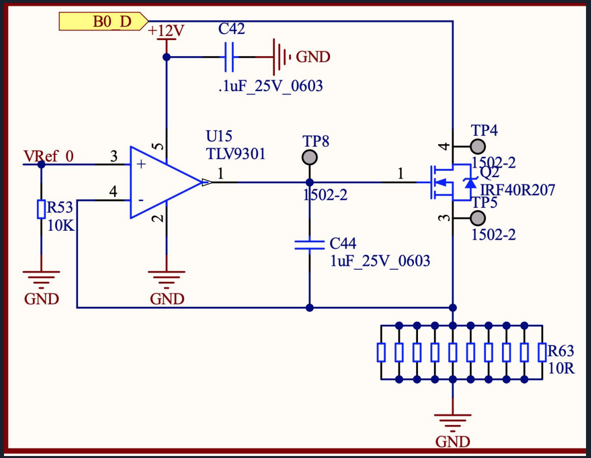
- Chose TI BQ2425x
  - Constant charge IC
- Self regulating
  - Will know when the batteries are charged
- Communicates to microcontroller via I^2C
- Thermal shutdown
- Programmable charge voltage and current
  - Will help with characterizing batteries



Constant Current Load

- An active circuit is required to discharge batteries at constant current across entire voltage range.
- Vref is generated the DAC on the microcontroller





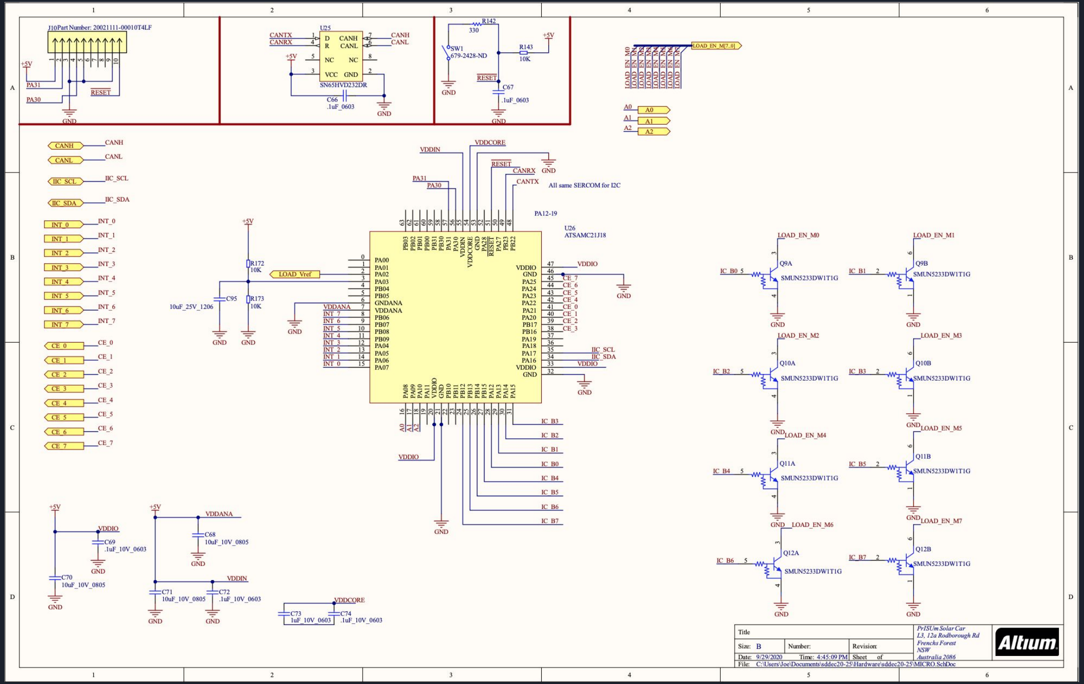
Requirements:

- Programmable current from 0-2.5A
- Capable of dissipating 4.3W continuous(worst case).
- Stable with minimal overshoot.

EEVBlog Video:
https://www.youtube.com/watch?v=8xX2SVcItOA&t=111s

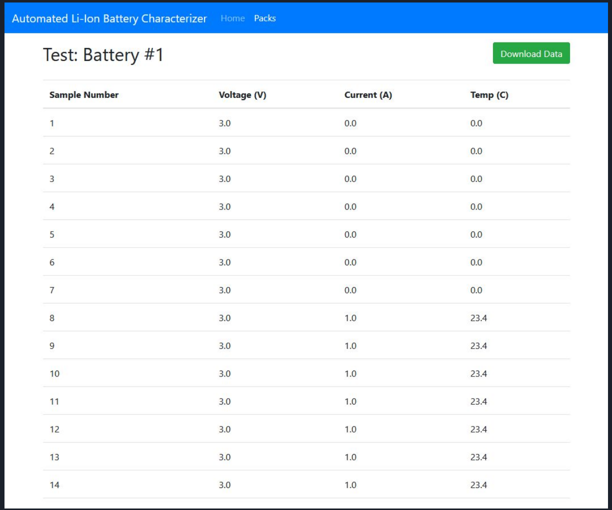
Microcontroller

- Using ATSAMC21J20A-AUT microcontroller
- Pre-Biased NPN for driving relay coils
- CAN Transceiver for PI communication

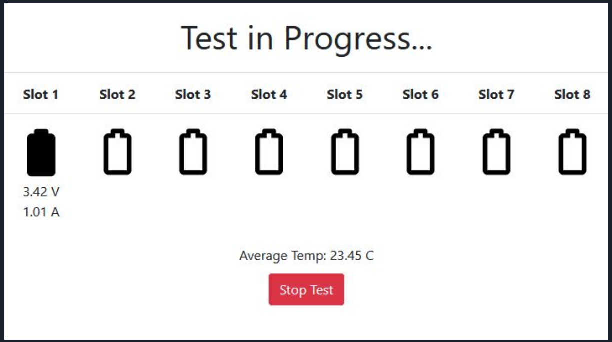


Raspberry Pi

- Raspberry Pi
  - Web backend
  - Database
- Interact with user and hardware
  - Start battery testing
    - Receiving testing data
  - Look up batteries
  - Create packs

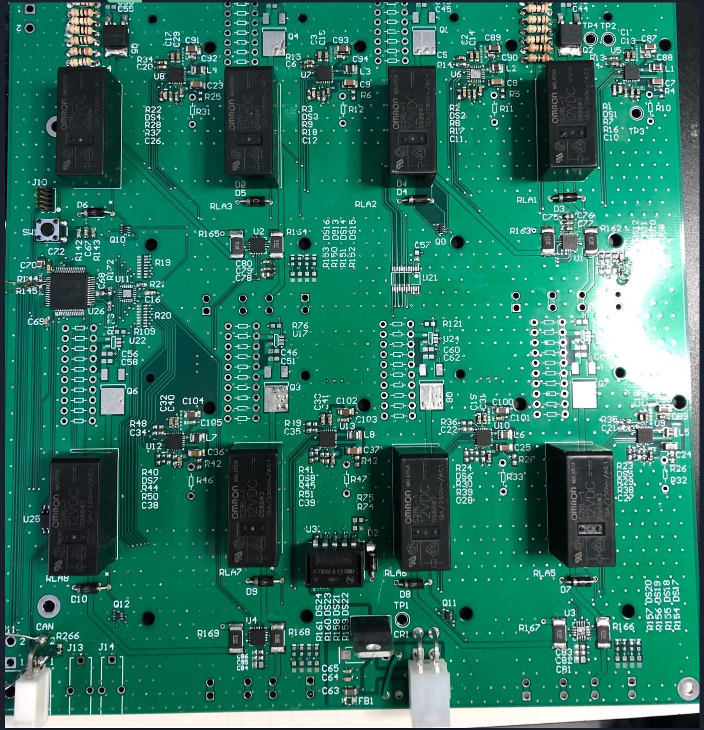


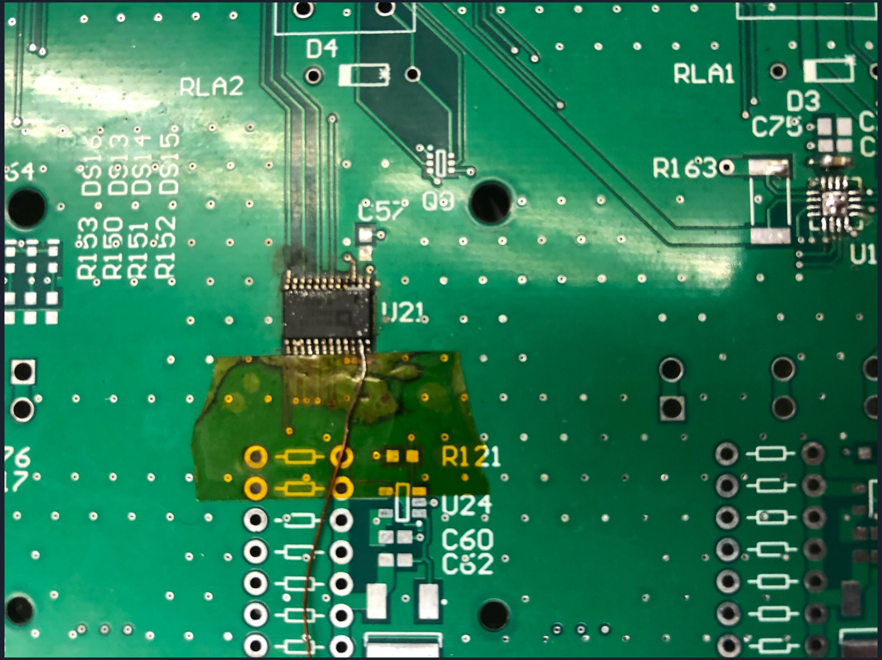
Sample Number	Voltage (V)	Current (A)	Temp (C)
1	3.0	0.0	0.0
2	3.0	0.0	0.0
3	3.0	0.0	0.0
4	3.0	0.0	0.0
5	3.0	0.0	0.0
6	3.0	0.0	0.0
7	3.0	0.0	0.0
8	3.0	1.0	23.4
9	3.0	1.0	23.4
10	3.0	1.0	23.4
11	3.0	1.0	23.4
12	3.0	1.0	23.4
13	3.0	1.0	23.4
14	3.0	1.0	23.4



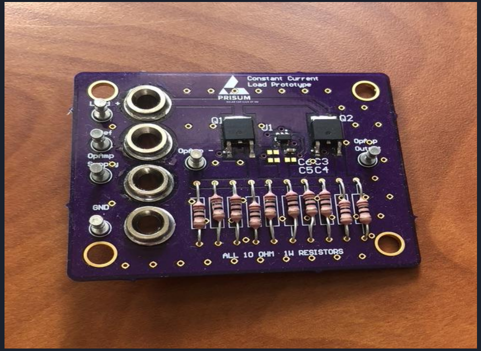
Testing Process and Results

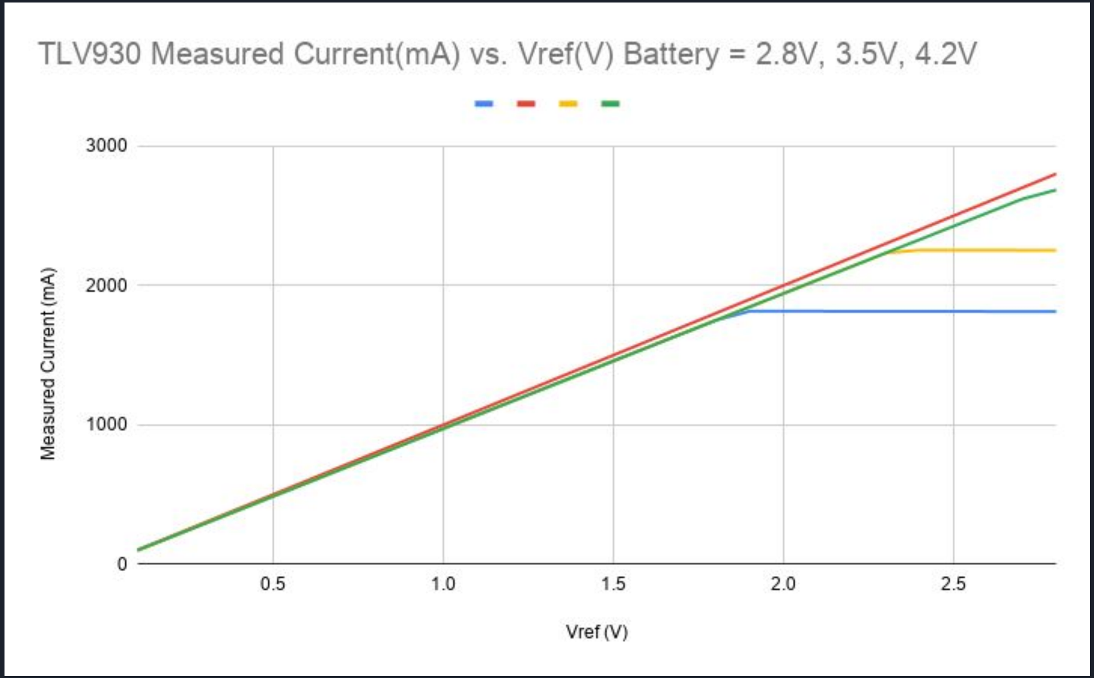
Pictures of boards here





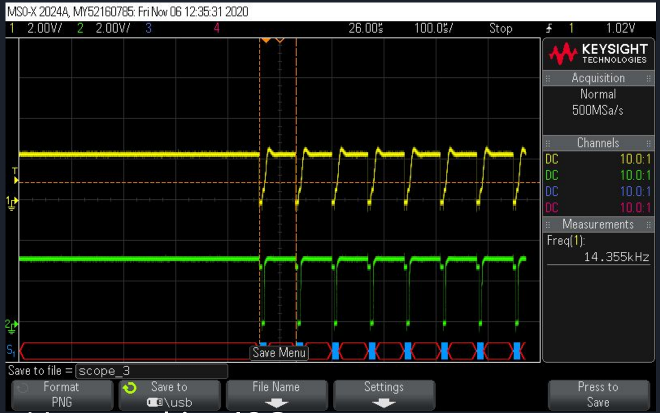
Load Circuit Test



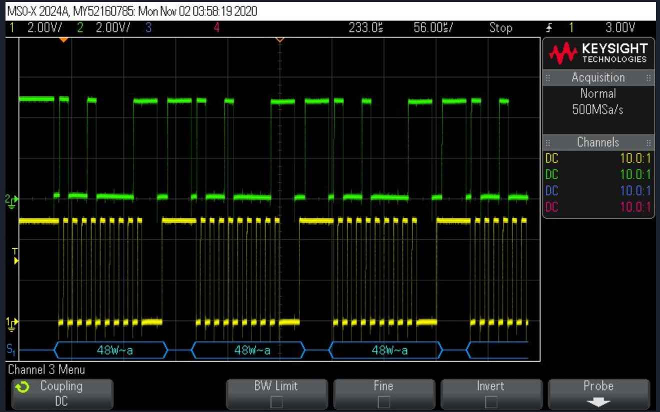


Communication Test

- CAN worked with little to no errors
  - Confirmed using external device to print values on bus and oscilloscope
- The largest source of bug came from the I2C bus.
  - We were able to communicate with the ADG715BRUZ-REEL7 (analog switch) consistently.
  - Broke when we added another part to the bus (one error seen on the right)
- Tried several ways to isolate I2C bug
  - Verified voltages, resistance, and pin layout



Non-working I2C



Working I2C Bus (minus acknowledge)

Conclusion

Project Milestones & Status - Hardware

Milestones

- Completed all circuits required for functionality:
  - Charging
  - Load
  - Measurement
  - Microcontroller
- PCB Layout
  - PCB Rev 1 Soldered
- Tests
  - Communication
  - Load Circuit

Status

- Need to design Rev 2 of PCB
  - Fix analog switch reset so it's not tied low
  - Add more LEDs and Test points
- Potentially look for a new charger that can have up to 8, I2C addresses
  - Simplify I2C bus

Project Milestones & Status - Software

Milestones

- Read and Write using I2C and CAN
- Working DAC output
- Will send data messages when asked
- Running website with connected database.
  - Can store from web app
  - Start test and watch status
  - Can look at past data

Status:

- Microcontroller communicating with rest of ICs
- Download battery data for pack
- Characterize battery pack

Future of the Project

- Project will be given to PrISUm Solar Car to finish if desired
  - We would give the following list of steps we would follow to continue work

- Reorder new parts to retest PCB
  - A current theory is that multiple trips through a reflow oven has damaged some of the ICs
  - Assemble a fresh board, a short developed on our first board that we can’t find.
- Order Rev 2 that fixes several small bugs that made it difficult to test/operate the board
  - I.e adding test points and LED
  - Fixing the RESET pin as we had to cut traces and mod wire the pin to Vdd to operate the current board

Questions?