To: Robert Severinghaus
From: Jace Jenkins, Elray “Mana” Santiago, Jonathan Ciulla, Luis Camargo
Date: 26 March 2021
Subject: Testing Results Report

Testing a product is one of the core steps to the engineering design process. It is integral to a team’s success to have sufficient testing to give consumers the satisfaction of knowing their product works and is ready to last. Below is the overview of our team's testing process.

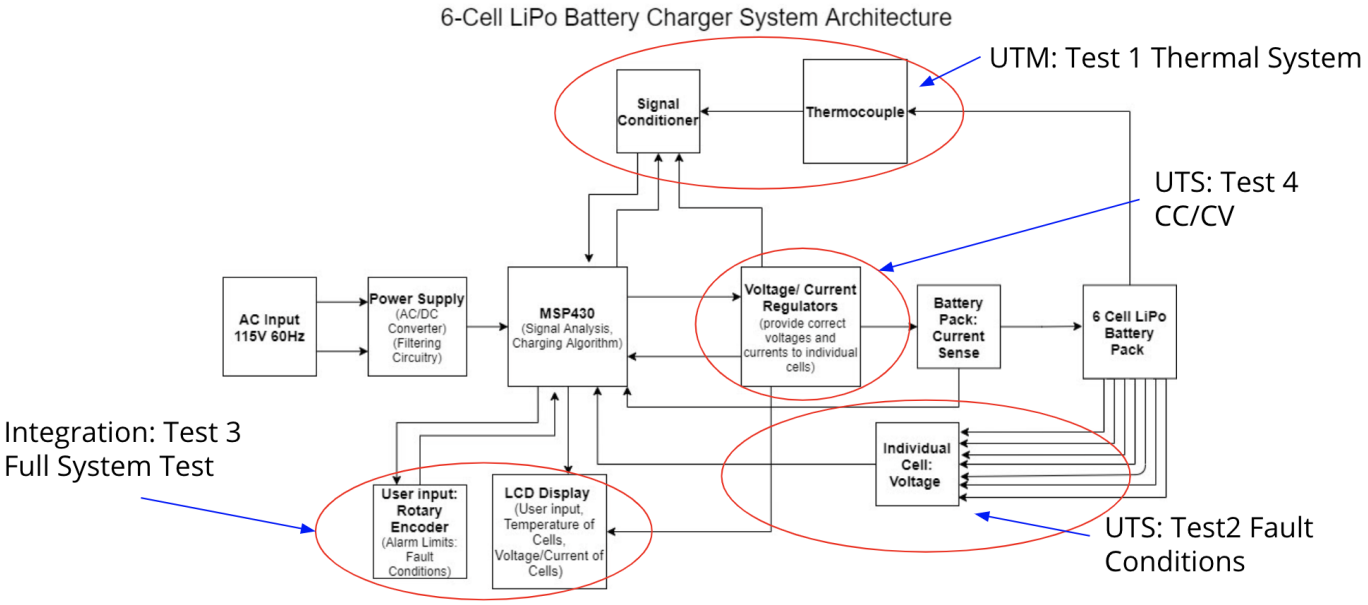
For our team’s senior capstone project, we have been tasked with designing and constructing a battery charger that is capable of charging a lithium polymer battery with six cells. Our requirements state that the battery charger must be powered by a standard 115 volts (V) AC outlet and that we must use an MSP430 microcontroller for controlling our charging algorithm. Now, that our project is constructed, the moment of truth is here as we finish the final stages of testing. To compare our project to the requirements defined by our client, we established four distinct tests that have been executed and documented for transparency. We performed three unit tests, one matrix and two step-by-step, where each test mainly focused on a single system of our project. We also implemented an integration test that tied in all the different subsystems as well as one integration test. Our first, step-by-step test took us around three hours to complete from start to finish which was relatively quicker than the rest of our tests. Our matrix test and second step-by-step test both took about 5-6 hours to complete, due to a prolonged setup time and the time needed in-between steps. Our integration test, which we spent the most time on, took us around 10 hours to finish. Although there were a few steps in our tests that did not produce the expected results, the overall outcome of each of the four tests was considered successful by our team.

2a. Introduction

Our client, John Lehman, is the vice president of product development at Dataforth Corporation and he has been providing valuable insight for us along the way. Being part of a senior capstone is a valuable opportunity for John and his company to showcase Dataforth's products and recruit potential candidates for employment. The problem to solve that John has equipped us with is to charge a six-cell lithium-polymer (LiPo) battery using an MSP430 microcontroller, which is no simple task. To safely and efficiently charge a LiPo battery, a lot of factors must be taken into consideration because of the hazards involved with the chemistry of such a battery. We made sure to do a lot of research beforehand to ensure that our design would be effective and that there would be no risk of harm for the user. The most fundamental aspect of our design is the power supply because, without it, our product would be useless. Our power supply plugs into a standard AC outlet of 115V and can supply 36V and 10 amperes (A) to our main power and ground lines. The power supply's output goes into five separate voltage regulators which are all used to step down that voltage even further, for the rest of our circuitry. The first two regulators are outputting a little over 5V which will be providing power for our MSP430 microcontroller, Liquid Crystal Display (LCD), and fan. The MSP430's 5V output will be utilized to power our thermocouple conditioning module which is used to read from the thermocouple sensors. The schematic showing our thermocouple connections to the MSP430 pinouts is shown in Appendix C. The MSP430's 3.3V output is used for supplying power to our rotary encoder which will be used for scrolling through the menu and selecting options by the user. The third voltage regulator is outputting 25.2V to our battery pack which is only happening during the constant voltage phase of our charging algorithm. The last two regulators are tied together in a parallel connection and they are outputting 2.35A to our battery pack during the constant current phase of our charging algorithm. Two solid-state relay switches are employed for switching between the constant voltage and constant current phases during charging, which is controlled by the MSP430 microcontroller. The schematic of our constant voltage and constant current regulators along with their respective relays are found in Appendix A. The individual cells of our battery pack are connected to a voltage divider that is being read by the microcontroller for determining what phase of charging should be in execution at the moment. The schematic showing the voltage dividers and individual cell pinouts is shown in Appendix B. The charging algorithm will switch from the constant current phase to the constant voltage phase once the cell voltages reach 4V and the charging will be complete once the current drops down to a charging rate of 0.1 C. For protecting the individual battery cells from overcharging we are utilizing integrated circuits specifically designed for cell protection. As the battery pack is being

charged, the MSP430 will be receiving the cell voltages and battery pack temperature as inputs and outputting them on our LCD along with a timer for the user's sake. In our program, we have several fault flags that can be triggered by unwanted readings which will cause the charging of the battery to stop, so that the risk of harming the user will be lowered.

2.b System Architecture:



2c. Requirements, status, type of test:

Test 1: UTM					Test 2: UUT			Test 3: Integration			Test 4: UUT		
Test	Ambient Temp on Thermocouple A	Thermocouple B (Test)	Average	Expexted Result	Step	Action	Expected Result	Step	Action	Expected Result	Step	Action	Expected Result
1	22.0°C	31.0°C	28.5°C	26.5°C	1	No battery connected/Faulty battery	Fault display telling user to connect the battery	1	User selects time limit and temp. limit using rotary encoder	User's input is accepted and begins detecting for a battery connection	1	Connect CC and CV rails to the MSP430 and quickly test the solid state relays	Turn on/off the output of the voltage regulators
2	22.0°C	36.0°C	30.2°C	29.0°C	2	Connect 1S battery to the circuit	Correct number of cells with the pack voltage	2	Unit autodetect Cell Count	LCD display shows correct cell count (1,2,3,4,5, or 6S)	2	Turn on CC regulators and turn off CV regulator	Measure around 2.3 A coming from the output
3	22.0°C	40.0°C	32.8°C	31.0°C	3	Connect 2S battery with 1 faulty cell	Fault display telling user "Faulty Battery" disconnect the battery	3	after 5 minutes, disconnect battery and test the voltage of all cells	All voltages nearly equal (+/- 0.4V) : 3.8 V,	3	Use stopwatch to keep track of how long the CC regulator stays on while checking the temp of the heat sinks	CC regulators will hit the thermal cut off around 2 mins and stop supplying current
4	20.0°C	47.0°C	35.0°C	33.5°C	4	Connect 2S battery with good cells	Correct number of cells displayed on LCD along with the pack voltage	4	Reconnect the battery and charge for 10 minutes	All voltages nearly equal (+/- 0.4V) : 3.9 V	4	Set user limits and connect battery	Battery temp and voltage displayed on LCD before turning on the switch connecting the CC regulator
5	18.0°C	50.0°C	35.3°C	34.0°C	5	Connect 3S with 1 faulty cell	Fault display telling user "Faulty Battery" disconnect the battery	5	Verify when one cell reaches 4.1V the system enables constant voltage and the current	One cell voltage is 4.1V, Measure the current across the sense resistor and verify with a multimeter			
6	18.0°C	57.0°C	38.1°C	37.5°C	6	Connect 3S battery with good cells	Correct number of cells displayed on LCD along with the pack	6	Wait till battery is fully charged	Constant voltage does turn off prints battery fully charged to the LCD.			
7	19.0°C	63.0°C	42.3°C	41.0°C	7	Connect 4S with 1 faulty cell	Fault display telling user "Faulty Battery" disconnect the battery	7	Remeasure cell voltages	All voltages nearly equal (+/- 0.4V) : 4.15 V			
8	19.0°C	71.0°C	45.3°C	45.0°C	8	Connect 4S battery with good cells	Correct number of cells displayed on LCD along with the pack voltage						
9	20.0°C	82.0°C	52.9°C	51.0°C	9	Connect 5S with 1 faulty cell	Fault display telling user "Faulty Battery" disconnect the battery						
					10	Connect 5S battery with good cells	Correct number of cells displayed on LCD along with the pack						
					11	Connect 6S with 1 faulty cell	Fault display telling user "Faulty Battery" disconnect the battery						
					12	Connect 6S battery with good cells	Correct number of cells displayed on LCD along with the pack						

2d. Most important requirements

1.5.1.1 Fault conditions: Battery over or undercharging ±100mV, battery reached one hundred percent charge capacity, faulty battery connection, wrong user input for battery type, battery over or under current ±50mA, faulty cells(disconnected cells, chemically unstable cells, abnormal temperature variance of cells)

This requirement is of the utmost importance to our client due to our insurance of the user's safety while the battery is in the charging phase. Our team can not allow the user to experience any type of harm due to the cells of the battery being over or undercharged. If any over or undercharging were to occur, our team has implemented redundant safety measures in the form of software and hardware that would cut off all current to the battery pack. If our team did not meet this requirement, our final product would be deemed unsafe for the user and therefore unacceptable for the client's demonstration at the automotive expo in Novi, Michigan. The accuracy of the measurements for both the voltage and current need to be within the acceptable range to minimize the potential for a runaway effect occurring where the battery would become chemically unstable and hazardous to the user.

1.5 User settable alarms limits via rotary encoder with a button for fault conditions detection for defective cells within the battery pack

The user-settable alarms stem directly off of our client's project proposal. The importance of this requirement is greater when compared directly against our team's additional requirements. This fact incurred our team's attention as our project would not suffice the client's expectations upon product delivery. The safety of the system was at risk when our team allowed the user to alter the fault conditions for when the system would cut off the charging cycle. To mitigate any additional risks by allowing the user autonomy over the system, our team reframed from full-autonomy and only allowed the user to set a timer limit, as well as the temperature limit within a range deemed safe by the team. If we did not meet this requirement the alarms and fault conditions our system would have the capability to allow for the user to interface with the system.

1.3 Display thermals of individual cells and the overall average of the six cells to LCD.

While this requirement was not specified by our client, the importance of this requirement remains equal to our other ones. The display of the individual cell voltages, while the battery is being charged, provides the user with an insight into how the charge being supplied to the cells is being divided amongst them. Also, this requirement is an implicit safety feature since it provides the user the ability to turn off the charging in the case that a cell voltage is reading higher than the system allows for. If our team did not meet the requirement, our project would not be a failure because our client did not specify this requirement, but rather a self-imposed requirement.

2e. Types of tests

For our unit matrix test, we wanted to focus on a single subsystem that is crucial to our project and we concluded that our thermocouple module was the best fit. For the matrix test, the only differences between the inputs can be in numerical value. This was perfect for testing the

thermocouple because the sensors can only take an input of temperature and the only way we could change the inputs to the sensor would be by changing the temperature. To perform this test, we decided on using a cup of water that would be changing in temperature to compare our thermocouple to another temperature sensor that was reading off of a multimeter. An image located in Appendix G shows how we used our hardware to execute this test. The temperature reading from the multimeter was considered to be the actual temperature and then we would measure the water's temperature using one of our thermocouple sensors. Since we were only using one thermocouple to measure from the water while the other thermocouple sensor was kept at room temperature, we expected the temperature reading displayed on the LCD to be the room temperature, plus half the difference between the water's temperature and the room's temperature. This is because our LCD is programmed to be displaying the average of the two thermocouple sensor readings, as shown in Appendix E. We chose to do this test second because it required using little amounts of hardware or software and we expected it to take longer than only one other test.

For our first step-by-step test, we decided on testing our fault conditions related to the battery due to safety being one of our biggest concerns with this project. To perform this test, we started with having the battery disconnected to see if the LCD would notify the user that the battery must be connected and from there we began connecting individual cells of the battery pack. We connected a single cell to ensure the cell's voltage would be displayed and then we purposely connected an additional faulty cell to check that the display would notify the user of a faulty cell. As shown in Appendix H, we then repeated this process counting up to six cells and we decided to perform this test first because it was the quickest and simplest of our four tests. With our required matrix and step-by-step tests out of the way, we had one more test to complete before we finished with our integration test, which would involve the most hardware and software.

For our third unit test, we decided on doing another step-by-step test that would incorporate crucial hardware that was not included in our previous unit tests. Testing our constant voltage and current phases of our charging algorithm seemed like a perfect subsystem to dive into before we started with our integration test. To perform this test, we started with turning on the constant current and constant voltage regulators separately and testing their outputs with no load attached. After we confirmed that the outputs were correct, we attached the LiPo battery and tested supplying our constant voltage and constant current rails to the actual battery to see if it would charge or damage the battery. For safety, our battery was kept inside a LiPo safe bag during this test as shown in Appendix I. We executed this test third because it was the most complex unit test, but it is much more simple compared to our integration test.

For our final test, we needed to tie in all the subsystems of our project into one integration test. To perform this test, we needed to use all the hardware and software that our final product would be implemented to ensure that there are no more changes needed in our design. This test involves putting all the previous tests together and adding all the microcontroller control that was left out in the previous experiments. This test is necessary to confirm that our microcontroller will sense the change in cell voltage and switch the charge that is being supplied to the battery from constant current to constant voltage at the correct time and that it will stop all charging once the cells reach their limit of 4.2V. The written code for our charging algorithm can be found in Appendix F. Without proper control of our charging algorithm, permanent damage could be done to the battery pack and the user could be potentially harmed.

2f. Major Test

CC/CV Test (Unit Step-By-Step)

The first major test completed was a unit step-by-step test that tested the constant current and the constant voltage systems. The constant current and constant voltage are required for various phases of the charging process. The system initiates with the constant current phase till the battery reaches 70% fully charged when then it disables the constant current and enables the constant voltage for the remainder of the charging cycle. The constant voltage continues till the current supplied to the battery is at a charge rate of 0.1 (which is equivalent to 0.1 multiplied 1.250A = 125mA). Once the current drops below this value, the charger will then print the charge completed to the screen until the battery is disconnected.

Our first step in the test was to ensure the solid-state relays did enable and disable the constant current and constant voltage system (CC/CV). This step was a success considering that when the enable signal was sent to the solid-state relay for either the constant current or constant voltage would be enabled. Next, our team tested the turning off of the constant voltage and then enabling the constant current. This test was a success and the current output from the constant current was measured at 2.354A when the expected result was about 2.3A. The following test was to test the endurance of the constant current system in terms of heat dissipation. Our team measured the amount of time before the constant current would reach its thermal cutoff. Our test was a failure because the constant current system turned off at forty seconds when the expected result was one minute. The fourth step was to connect the CC/CV system to the MSP430 and set the user-defined fault conditions. The result from this test was such that the user successfully set the fault conditions and the cell count and cell voltages along with the pack temperature to the screen.

The fifth step tested the charging functionality of the constant current to the battery pack. The system passed the test, and the current measured to the input of the battery was 2.3A which was the expected value of the current. The sixth step was to allow the constant current system to be enabled for ten minutes and verify the thermal cutoff was not reached and the heatsinks attached worked as intended. The seventh step was to the constant voltage while the battery pack was connected. Our team conducted a CV test for a two-cell battery pack with cell one reading 3.88V and cell two 3.87V, total pack was measured at 7.75V before charging. The charging voltage was 9.2V and the results from this test were that cell one was measured at 3.89V and cell two was measured at 3.89V with the total pack voltage at 7.79V. The average increase for this test was that the battery cells would increase 10mV every ten minutes as seen in appendix L.

The next test was to allow the constant current charge of the two-cell battery for ten minutes. Before charging the battery pack was measured at 7.78V, after the charge cycle for 10 minutes the battery pack reached a total pack voltage of 7.89V. The average charge during the

CC charging cycle was 25mV per 10 minutes as seen in Appendix L. We concluded our CC/CV test by disabling both the CC and CV systems, which the system passed.

Average Pack Temperature Display Test (Unit Matrix)

Testing the accuracy of our thermocouples is important because if a LiPo battery exceeds its temperature limit the battery could be permanently damaged or catch on fire. For our final product to be safe for the user, our temperature readings must be accurate within a certain range, which is being put to the test in this unit matrix test. For a test to be considered a matrix test, the inputs of the test should all be collected the same way and the inputs should only change in value. This type of test works perfectly for confirming the accuracy of our thermocouple sensors because we will only be changing the temperature that is read by the thermocouple and confirming the reading on the LCD. To perform the test, we started by taking the ambient temperature of the room which will be the temperature that one of the two thermocouple sensors is kept. Then we heated a cup of water in the microwave and measured the temperature of the water using a thermometer. We waited until the temperature of the water-cooled down to our desired temperature and then inserted one of our thermocouple sensors into the water and recorded the temperature that was displayed on the LCD. For example, if one thermocouple was at room temperature and the other was at 3 degrees Celsius (℃) above room temperature, we expected the LCD to show a temperature of 1.5 ℃ above room temperature due to our calculated temperature being the average of the two thermocouples. We set a range of +/-1.5 ℃ for comparing our actual results with our expected results to determine whether that trial failed or passed. After each trial, we switched thermocouples, let them cool back down to room temperature, and recorded the room temperature again in case it changed from the last trial. We switched the thermocouples every time to ensure that we were not only changing the The input of one and to confirm both thermocouples are accurate. We repeated these steps many times over and tested our readings at four temperatures above 50 ℃ and four temperatures below 50 ℃.

Overall, we were satisfied with our results of this test and we consider our thermocouples to be accurate enough for our application. For our first three trials, the room temperature was at 22 ℃ and we heated our cup of water to 31 ℃, 36 ℃, and 40 ℃. Our expected results for the first trial were 26.5 ℃ which comes from the average room temperature of 22 ℃ and the temperature of the heated water which was 31℃. Our actual results came out to be 28.5℃ which was past our range of +/-1.5 ℃, so we considered that trial to be a failure. For the second trial, we were within 1.2 of our expected result so we considered that trial to be a success. For the third trial, we read 1.8 ℃ above our expected temperature so that was also considered a failure.

The next five trials were all within 1.5 ℃ and they were all counted as successes but our final trial was over the expected temperature by 1.9 ℃ and was counted as a failure. As you can see from our results, our thermocouples are not accurate, but they are accurate enough that we are not worried about using them as they currently are in our design. The biggest failure we had was two ℃ above the expected temperature which is only 0.5 ℃ away from success and our most accurate trial resulted in only a difference of 0.3 ℃. Our thermocouple system was highly accurate with a correlation coefficient of 0.9975,

Full System Integration (Integration Test)

The final test was the full system integration. This was chosen to be executed last due to its dependencies on the success of all subsystem testing. The integration test consisted of the LCD, fan, thermocouple, rotary encoder, and cell voltage reading subsystems. This test was Whitebox considering our team required the knowledge of which systems were to be implemented. The first step in our test was to allow the user to determine whether or not they wanted to select a customizable fault condition or to use the default settings. Step one passed as it works for all user input cases. Step two, was to ensure the cell autodetection was accurate and displayed the correct number of cells with their respective voltages. Step three in our test was to allow the battery to commence the charging phase for five minutes in constant current mode. The cell voltages average prior to charging was: cell1 = 3.95 V, cell2 = 3.83 V, cell3 = 3.83 V, cell4 = 3.85 V, cell5 = 3.92 V, cell6 = 4.00 V Total pack 23.23 V cell AVG 3.850V and post charging. The cell voltage average was 3.905 with cell1 = 3.99 V, cell2 = 3.85 V, cell3 = 3.85 V, cell4 = 3.88 V, cell5 = 3.98 V, cell6 = 4.03 V Total pack 23.43 V after a total recorded charge time of five minutes and nine seconds. The fourth step in the integration test was to reconnect the battery pack to charge for ten minutes and recheck the cell voltages. The next step in the testing was to reconnect the battery and remeasure the individual cell voltages after ten minutes of charging. The cell voltages after ten minutes and eight-second was: cell1 = 4.02 V, cell2 = 3.88 V, cell3 = 3.88 V, cell4 = 3.91 V, cell5 = 4.01 V, cell6 = 4.12 V, total pack 23.66V, and the cell average was 3.943V. The sixth test was to verify that after any cell reached a voltage of 4.0V the system would trigger that the battery is fully charged. The seventh test was to then wait till the battery was fully charged after the current reading was below 0.1 C-rating (C). The last step in the test was to remeasure the cell voltages.

2g. Analysis of results:

During our testing phase of the design process, our team encountered multiple unforeseen issues. The issues were a direct result of a lack of integration before the testing phase. Our team had conducted various subsystem tests upon their respective completion. One of the most prominent issues that arose was the user interface system and cell autodetect systems did not properly allow for smooth interaction when using the rotary encoder. Also, the cell autodetect function did not properly account for when no cells were detected and if a battery was connected after the system was turned on. Our team was able to correct the issues with live patches.

In terms of hardware, the constant current constant voltage unit step-by-step test shed light on the issues with our lack of preparation and system compatibility when our team tied both the outputs from the constant current constant voltage together. The issue we faced was that the constant current solid-state relay would experience excessive heat when the constant voltage was enabled. Our team is unsure of the exact cause of this issue because there could be a multitude of reasons such as the output lines of the CC and CV being tied to one another allowing for current to flow backward through the CC solid-state relay while the CV relay was enabled. Another possibility could have been that there was no load present on the output of the CC and CV while we had been testing. This could have resulted in the current flowing backward through the disabled solid state. Our team is confident the issue stems from the lack of a load on the CC and CV output. In the case when a load was present there CC and CV test resulted in exceedingly well results. The battery would charge at 50mV per ten minutes while the CV was enabled. As for when the CC was enabled the battery charged at 110mV per ten minutes which surpassed our hypothesized results. The most important system requirements of safety were not met entirely due to the CC solid-state relay increasing in temperature when it was disabled.

The following major test completed was our unit matrix test which evaluated the accuracy of the thermocouple and signals conditioning system. Overall, the test was a success considering the system proved to be accurate with a tolerance level of plus or minus 1.5°C. The test had three failures where the system measured the temperature either +2.0°C, +1.9°C, and +1.8°C too high. As for the remaining six tests, our system performed as expected with the average displayed temperature measured +1.03°C than the actual temperature. Major requirements of displaying accurate average battery pack temperature to the LCD were met. Through further analysis, our team corrected the correctional value for the code that calculates the individual temperature from each of the two thermocouples.

Our system integration test utilized several protoboards to accumulate every component within our design. After the successful completion of our CC/CV tests, we then upgraded our battery size to a six-cell LiPo battery versus the two cells used previously. The conclusions of

our results, represented in Appendix N, went as follows; Upon turning the system on, the LCD asked for the user fault conditions to be enabled and a list of options followed exactly as intended depending on the option that was chosen. From here the system did not have a battery connected so our fault conditions were triggered prompting the user to attach a battery. With the battery attached the charging process was able to begin as expected. The base voltages of the cells were recorded: Cell1 = 3.95 V, Cell2 = 3.83 V, Cell3 = 3.83 V, Cell4 = 3.85 V, Cell5 = 3.92 V, Cell6 = 4.00 V, Total pack = 23.23 V, and the CellAVG = 3.850V. After five minutes of testing the results showed a sizable amount of charge in each cell which showed that our system was able to effectively charge a LiPo battery as well as handle the loads of the system. The results are as follows: total time 5:09:50, Cell1 = 3.99 V, Cell2 = 3.85 V, Cell3 = 3.85 V, Cell4 = 3.88 V, Cell5 = 3.98 V, Cell6 = 4.03 V, Total pack = 23.43 V, and the CellAVG = 3.905V. Again the pack was charged for an additional 10 minutes and then the voltages were recorded with the overall CellAVG increasing to a total of 3.93V. According to Appendix N, our results yield the expected outcome in addition to the success of having our system switch correctly between CC/CV modes and shutting the system off when the cell average hit a certain voltage threshold. Therefore, the integration test can be concluded as a success.

2h. Lessons learned

As expected, we ran into many problems during our testing period, but fortunately, we were able to learn from our mistakes. During our matrix test, we had a lot of trouble heating our thermocouple inputs to an exact, desired temperature. Initially, we had specific intervals above the room temperature that we wanted to test the readings at, but it proved too difficult to control the temperature and we had to settle with heating water in the microwave. After starting our test we realized the thermocouple was not as sensitive as previously thought, which made our results less accurate. If we would have spent more time finding a suitable way to isolate the thermocouple better for a more accurate reading, our test results would have improved. Also, we ended up testing our thermocouple at temperatures that are much higher than our battery pack will ever reach because it required less effort and time. It would have made more sense to test at temperatures that will coincide with our application but we needed to move on to the rest of our tests. By confirming that our thermocouples were indeed accurate and finding out that more isolation is needed on the sensors, our final product will be even more precise.

During our first step-by-step test referenced in Appendix M, we had to dive back into our code before being able to gather all our results. When we first attempted the test we quickly realized that we had overlapping fault conditions. After multiple phases of trial and error, we had fixed our code and made sure that only the correct faults were triggered at the appropriate times. At first, our code was displaying a warning reading “Connect Battery” whenever a faulty cell was connected and this was because we had reused the same fault flag multiple times in our code. Once we had initialized more fault flags that were unique to certain faults, our code worked properly and all test steps succeeded. Also, a few of the user input requirements that were being checked in this test did have to be revised. Our user must set limits before the LCD will display any of the readings, so this requirement had to be checked for us to read from the LCD. Originally, we were going to allow our user to input presets for the current and voltage, but we had trouble using potentiometers to change the outputs of our regulators. Now the user is only setting limits on the temperature and time, which will shut off charging once either limit is reached. From executing this test, we learned about our oversights regarding the code and now our code is much more efficient and effective. Currently, our LCD is switching back and forth from displaying the temperature and the cell voltages which could be fixed by using a larger LCD. If we had more time, we could improve this test by using a larger LCD that could display all the data on the same screen while also immediately displaying any faults that occur.

During the testing of our constant voltage and current regulators, we ran into problems with overcurrent. Unfortunately, we were delayed multiple times because of damage done to the relays while testing the outputs of our regulators. We wanted to test our regulators with a

resistive load before we attached our battery pack, for safety reasons and this resulted in ruining our relay switches. Once we ordered new relays, we went back over our design and confirmed that the battery was needed on the load to prevent issues. After the shipping of our new relays was delayed, we could finally get back to our test and once we double-checked all the connections to the battery pack, we plugged in our two-cell LiPo battery. During this test, we were able to confirm that both our constant voltage and constant current regulators were working as expected. We were able to turn on and off both the regulators by changing the inputs to the relay switches and we measured the battery before and after charging to ensure that we were indeed increasing the battery's charge capacity. From this test, we learned that having the battery pack on our load was necessary to prevent damage to our circuit and that the orientation of the battery's balance port was also very important.

After our testing phase was completed, we then moved onto integrating a printed circuit board (PCB) that was created from our schematics. For this, we utilized the software EasyEDA which helped to translate our schematics into a fully integrated PCB. Through multiple stress tests, we have concluded that our PCB will not be fit for final product implementation. This is due to tracing and routing errors that were overlooked in the PCB design process. From this process, we have learned the importance of grounding issues that may occur given multiple sources. In conclusion, although this may not be a specific requirement for our client, we will continue to read over every trace to pinpoint exact discrepancies within our overall PCB design.

Appendix A

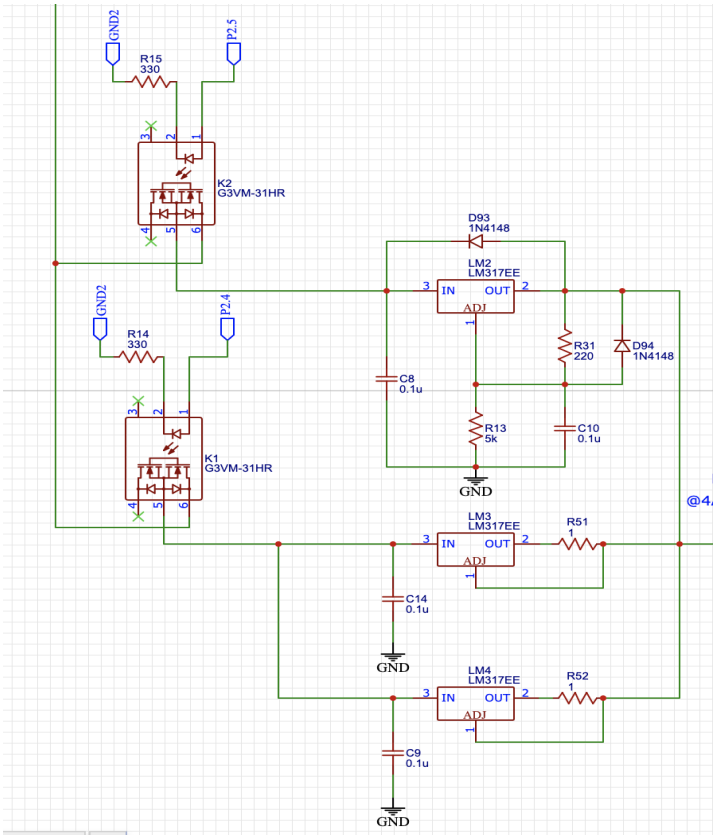


Figure 1. Constant Current and Constant Voltage Schematic

Figure 1. Dis[plays our CC and CV system where the leftmost green wire feeding into the G3VM chips on pin six is the supply coming from our main power supply input of 30V. The G3VM chips are enabled or disabled by the control signal coming from the MSP430 which is tied to pin one on both chips. The constant voltage is the top LM317 labeled as LM2 which outputs a constant voltage of 25.2V with a varying current determined by the charge progress of the battery. The constant current subsystem consists of two LM317 voltage regulators which are acting as current regulators in parallel to combine their current outputs. Both the CC and CV outputs are tied together as a unified output to provide power to the battery.

Appendix B

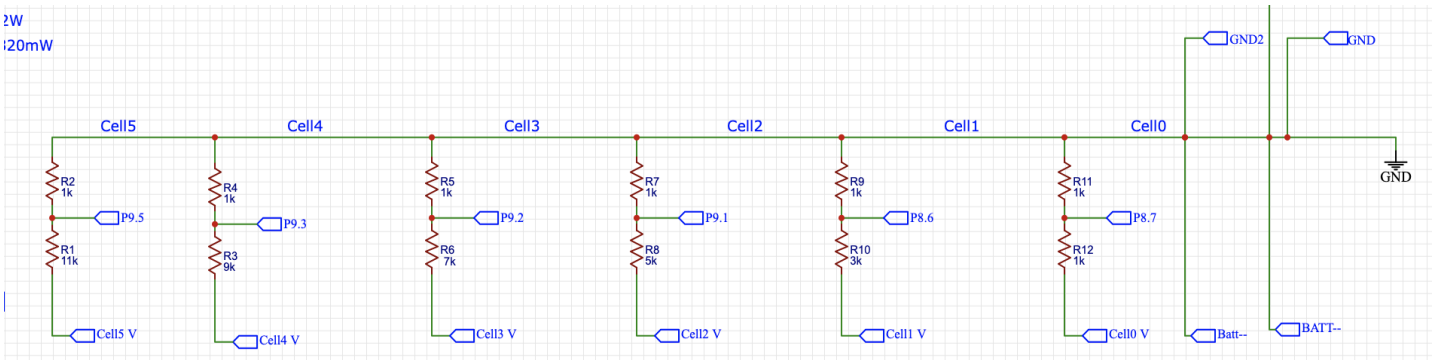


Figure 2. Individual Cell Voltage Measurement System

The figure above demonstrates our team's design for the cell voltage readings provided by the MSP430. We used a series of voltage dividers to allow for higher voltage readings than the stock allowable voltage reading which was 3.3V. The pack voltage of our six-cell battery was measured at 22.2V. The MSP430 read the analog input from the voltage dividers and used a correction algorithm to recalculate the true cell voltage of each cell.

Appendix C

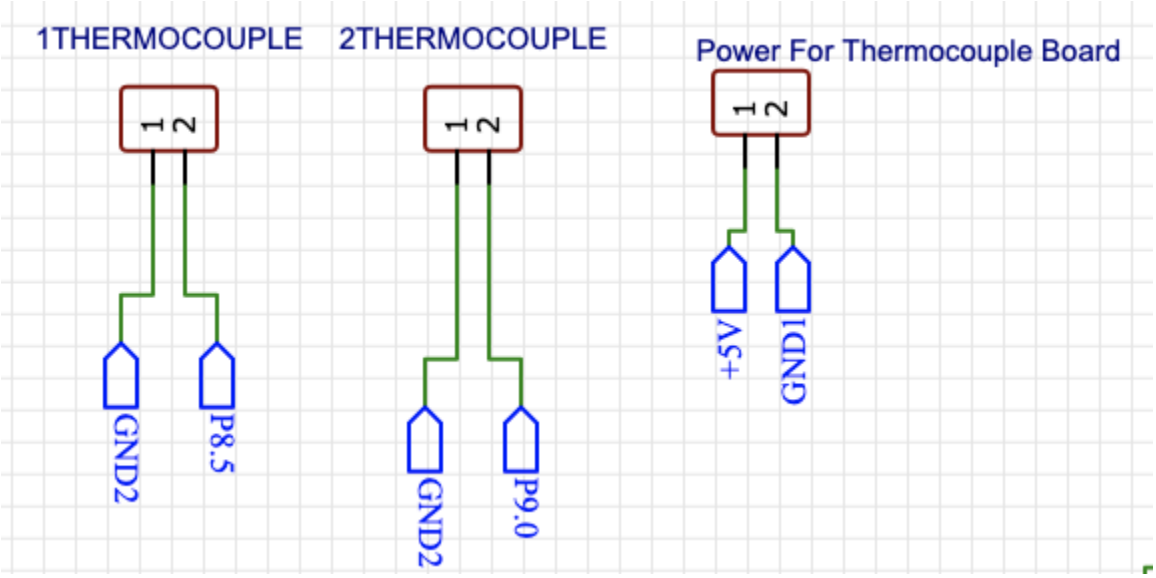


Figure 3. Thermocouple Temperature System

The figure above shows the connections of the thermocouple subsystem. The first thermocouple feeds into port 8.5 and thermocouple 2 feeds into port 9.0 on the MSP430. The power for the thermocouple signal conditioner board is provided by the MSP430 5V rail. All grounds reference the MSP430 ground.

Appendix D

```
781 void TrueCellVoltage(int ADC12MEM1, int ADC12MEM2, int ADC12MEM3, int ADC12MEM4,
782                        int ADC12MEM5, int ADC12MEM6)
783 {
784     int c = 2;
785     A = 1240.69; // reciprocal of 12-Bit ADC step size (.000806)^-1 = 1240.69
786     Cell0TV = (c * ADC12MEM1) / A; //Generalized formula
787
788     Cell1TV = (c * ADC12MEM2) / A;
789
790     Cell2TV = (c * ADC12MEM3) / A;
791
792     Cell3TV = (c * ADC12MEM4) / A;
793
794     Cell4TV = (c * ADC12MEM5) / A;
795
796     Cell5TV = (c * ADC12MEM6) / A;
797
798     // fault condition for if any cell below 3.0V
799     if ((ADC12MEM1 < 0x745) || (ADC12MEM2 < 0x745) || (ADC12MEM3 < 0x745)
800             || (ADC12MEM4 < 0x745) || (ADC12MEM5 < 0x745)
801             || (ADC12MEM6 < 0x745))
802     {
803         faultFlag2 = 1;
804     }
805     else
806     {
807         faultFlag2 = 0;
808     }
809
810     // Overvoltage Protection  4.25V == 0xA4C
811     if ((ADC12MEM1 > 0xA4C) || (ADC12MEM2 > 0xA4C) || (ADC12MEM3 > 0xA4C)
812             || (ADC12MEM4 > 0xA4C) || (ADC12MEM5 > 0xA4C)
813             || (ADC12MEM6 > 0xA4C))
814     {
815         faultFlag3 = 1;
816     }
817     else
818         faultFlag3 = 0;
819 }
```

Figure 4. True Cell Voltage Algorithm

The true cell voltage algorithm calculates the analog inputs from the voltage divider off of each cell considering the MSP430 is not capable of reading voltages greater than 3.3V and our max cell voltage is 4.2V. With our known ratio of dividing the input analog signal by 2 for each cell given the voltage divider resistors, we first multiply each analog input voltage by 2 then divide that value by the reciprocal of the 12-bit analog to digital converter (ADC) step size. The reciprocal of the ADC step size was calculated by (.000806mV)^-1 which = 12040.69.

Appendix E

```
770 void TrueTemp(int a, int b)
771 {
772     float temp = 29.045; //Divided the thermocouple ADC decimal value (639) by a
773                          //known degree Celcius (22) ... 639/22 = 29.045
774     RealTemp1 = a / temp; //Divide each thermocouple ADC value by known step size
775     RealTemp2 = b / temp; //Divide each thermocouple ADC value by known step size
776
777     AvgTemp = (RealTemp1 + RealTemp2) / (2); //Average of the two thermocouples
778
779 }
780
```

Figure 5. Average Temperature Algorithm

The average temperature is calculated by converting the hexadecimal value in the respective ADC memory register to decimal and dividing that value by the known ambient temperature. With the known thermocouple step size, we were able to pass the raw ADC memory register value to be divided by the known step size. Once both of the thermocouple temperatures were calculated, the average of the two values was then calculated.

Appendix F

```
679                     // If any of the cells reach 2.0V == 0x9B2 (raw input 4.0V), Start Constant Voltage
680                                   //3.90V                     //4.0V
681                     if((AvgPackCharge > 0x973) && (AvgPackCharge < 0x9B2))
682                     {
683                         P2OUT &= ~BIT4; //disable Constant Curr
684                         // enable the constant voltage MOSFET P2.5
685                         P2OUT |= BIT5;
686                         P9OUT |= BIT7;// gree led flash when on
687                         P1OUT &= ~BIT0;
688                     }
689                     if(AvgPackCharge < 0x973) //3.90v
690                     {
691                        P2OUT &= ~BIT5; // disable the constant voltage MOSFET P2.5
692                        P2OUT |= BIT4; //enable Constant Curr
693                        P1OUT |= BIT0;
694                        P9OUT &= ~BIT7;// red led flash when on
695
696                     }
697                     // Else Enter Constant Current
698                     if (AvgPackCharge > 0x9B2) //4.0
699                     {
700
701                         P2OUT &= ~ BIT4 | BIT5; //disable Constant Current/Voltage MOSFETs P2.4 P2.5
702                         BattFullCharge = 1;
703
704                     }
```

Figure 6. cc/cv code

The figure listed above demonstrates the charging algorithm which takes the analog input from the cell voltages and determines whether or not to enable or disable the CC and CV circuits based upon the cell voltage. If the pack cell voltage average is below 4.0V, the CC is disabled and the CV would then be enabled till the battery current is equal to or less than 130mA. The CV will be enabled while the pack average is between 3.90V and 4.0V. If the battery pack voltage reaches the full charge threshold of over 4.0V, the program will then consider the battery fully charged and disable both the CC and CV circuits and print to the screen that the battery has reached a full state of charge.

Appendix G

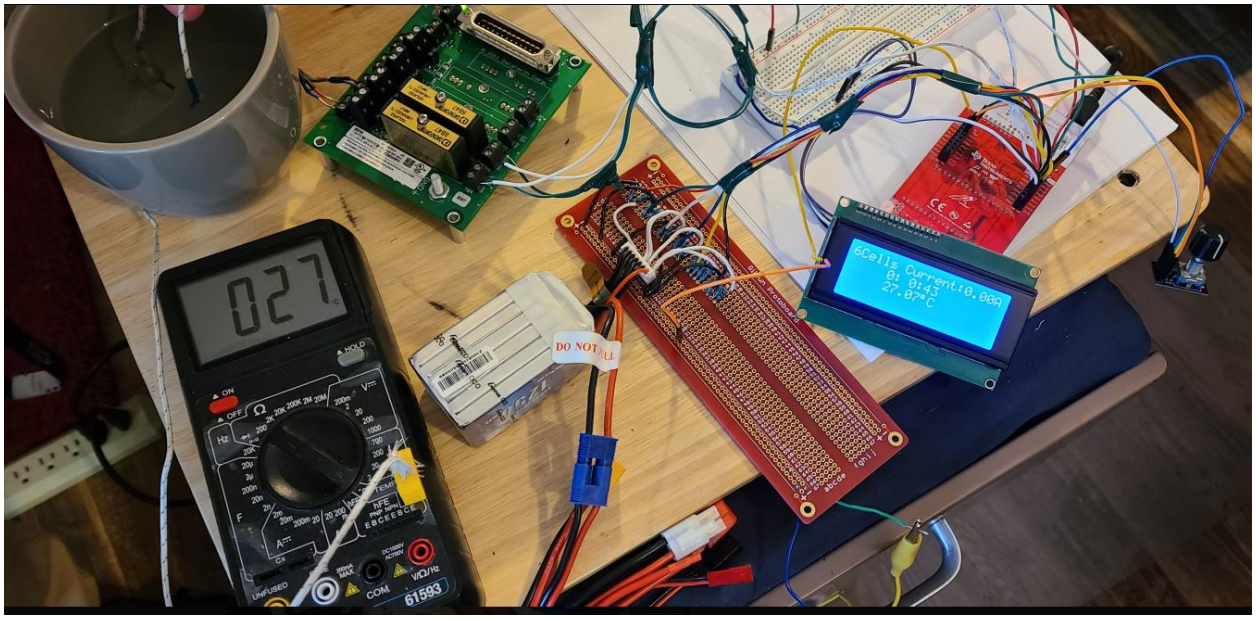


Figure 7. Hardware setup for thermocouple test

Shown in this image are the thermocouple module, LiPo battery, rotary encoder, MSP430 microcontroller, voltage divider, multimeter temperature sensor, and a cup of water. As you can see, the multimeter temperature sensor along with the thermocouple probe is both in the cup of water and both of their temperatures are being displayed on the LCD and the multimeter. Our LiPo battery is connected to the voltage divider which allows the MSP430 to read the cell voltages and the MSP430 is connected to the thermocouple, LCD, and rotary encoder. LC

Appendix H

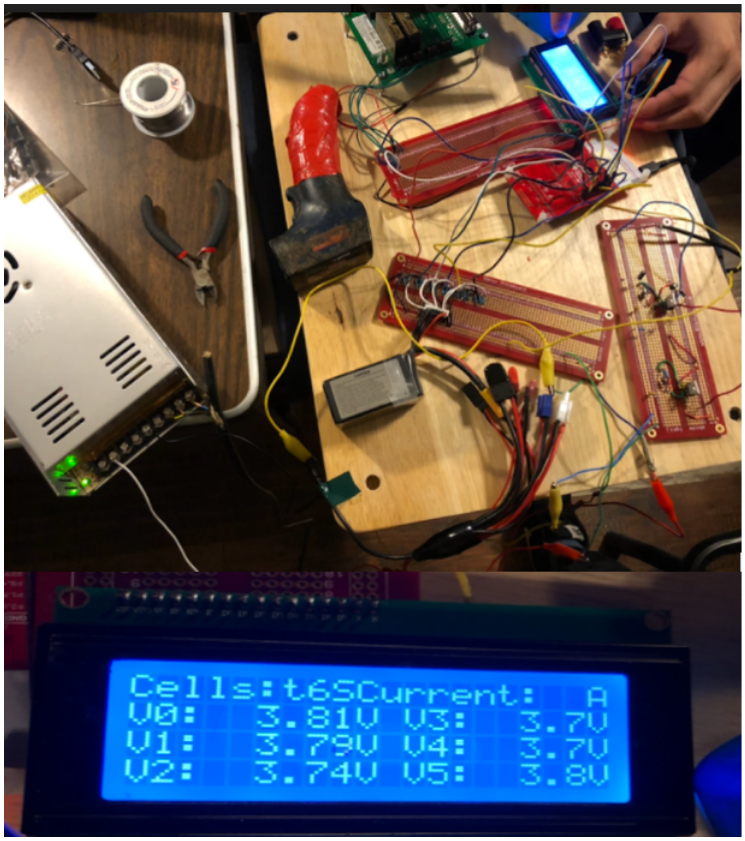


Figure 8. Hardware setup for fault detection test

This image shows our power supply on the left side with our LiPo battery, voltage divider, regulators, MSP430, and LCD on the right side. The regulators on the right side of the board are stepping down the voltage from the supply and powering the MSP430 microcontroller. The MSP430 is reading the cell voltages from the voltage divider and powering the LCD which is outputting the cell voltages for the user. On the bottom, there is a zoomed-in image of what the LCD is showing the user.

Appendix I

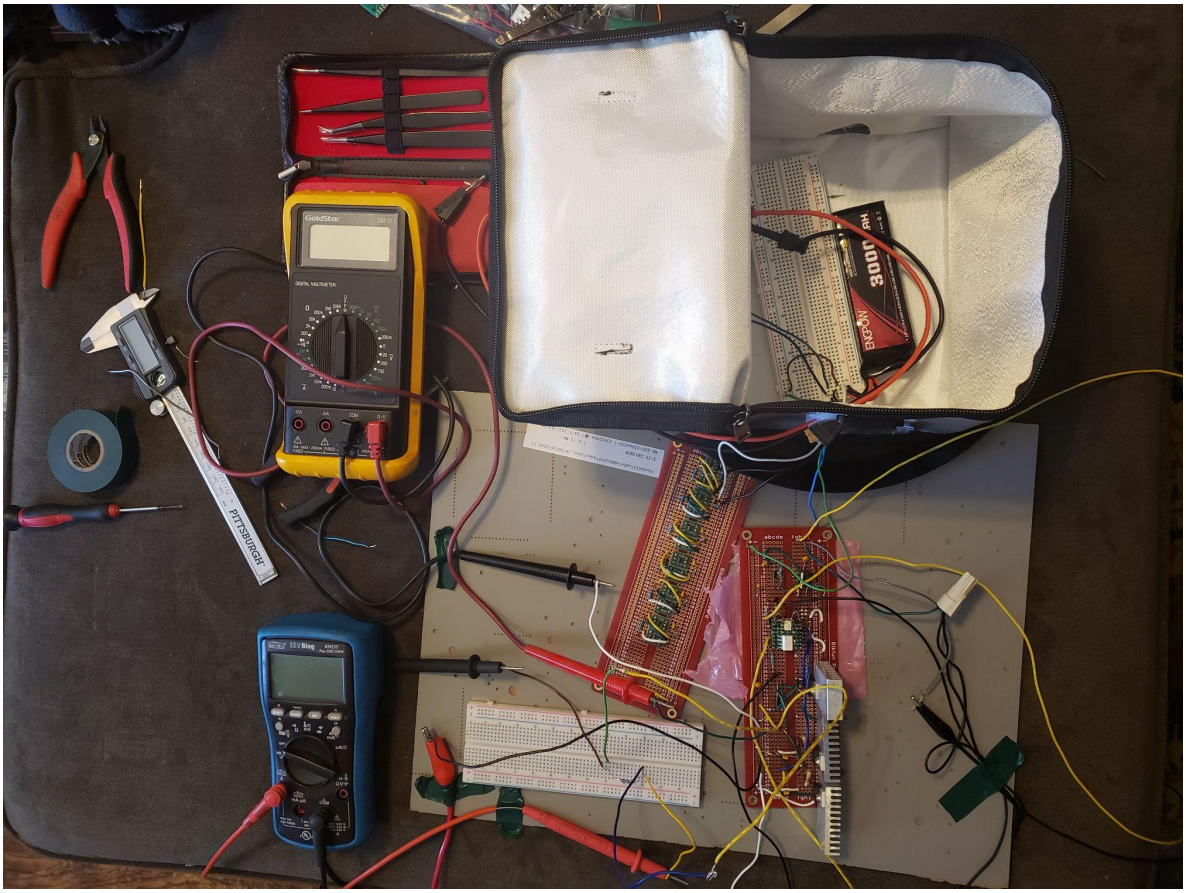


Figure 9. CC & CV Hardware Setup

The CC and CV testing was conducted using the solid-state relays which control the state of either system. The two multimeters were used to measure the voltage of the battery pack, displayed on the top yellow multimeter, and the bottom multimeter was used to measure the input voltage from the power supply. Also, our team had our cell protection circuit connected to the battery pack to sense if any over or under voltage or current was present. The cell protection circuit is seen as the left red protoboard in the figure. If any fault were to occur, the cell protection would divert the excess current of voltage directly to the ground.

Appendix J

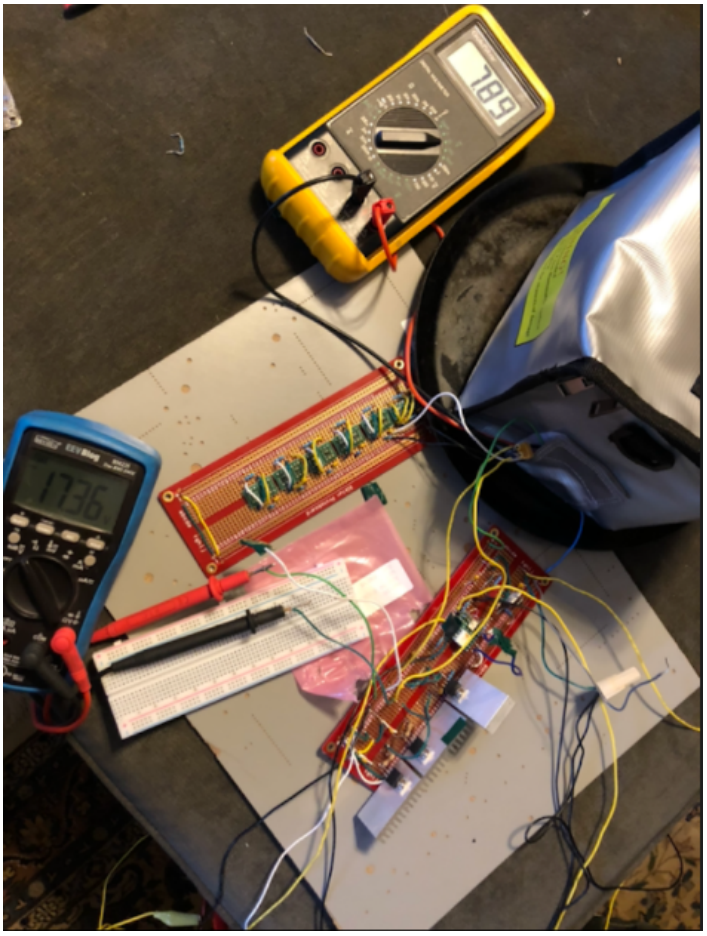


Figure 10. CC Test for 10 minutes

This image shows the hardware setup for how we did the stress testing of our constant current phase. The board on the right contains the regulators and solid-state relays that turn them on and off. The board on the left contains the cell protection ICs that are connected to the individual cells of the LiPo battery that is located inside the LiPo safe bag in the top right corner of the image. The blue multimeter is reading the voltage output of our constant current regulators to notify us if the regulators reach the thermal cut off while the yellow multimeter is reading the pack voltage of our 2 cell LiPo battery.

Appendix K

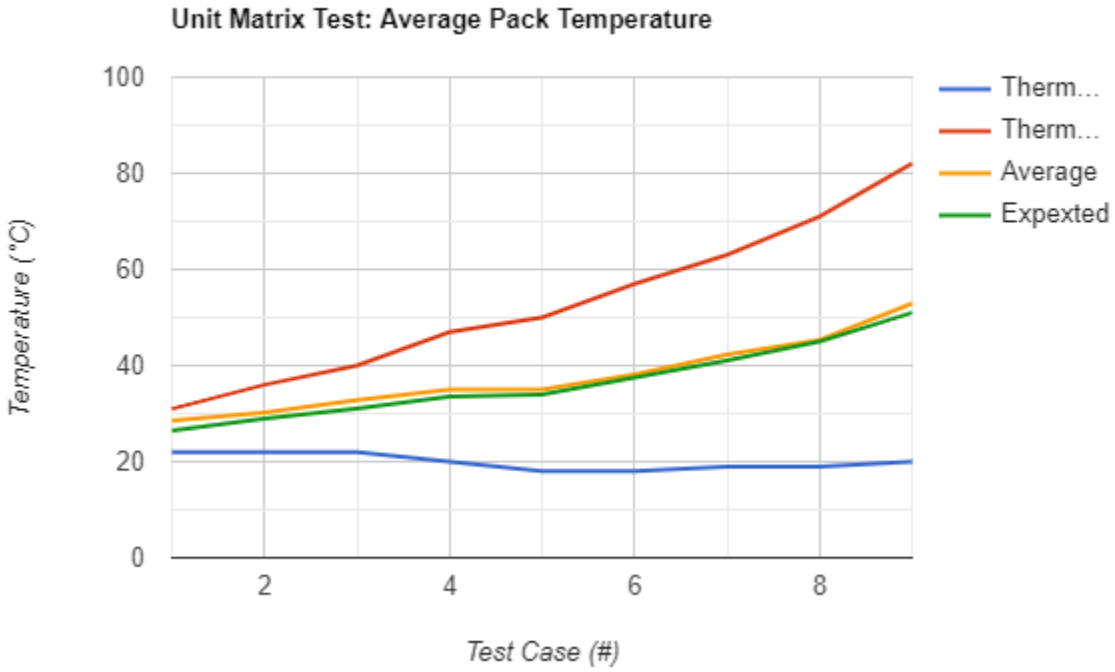


Figure 11. Graph of matrix test results

In this image, we have four plotted lines that represent the thermocouple readings of our matrix test. The blue line represents thermocouple A that we controlled and kept at room temperature while the red line represents thermocouple B that we had inside the heated cup of water. The yellow line represents the average of thermocouples A and B, which is the temperature that is displayed on our LCD screen. The green line represents the temperature that we expected to receive on our output for each trial. As you can see, our yellow line is always very similar to the green line, indicating that our thermocouples are indeed accurate.

Appendix L

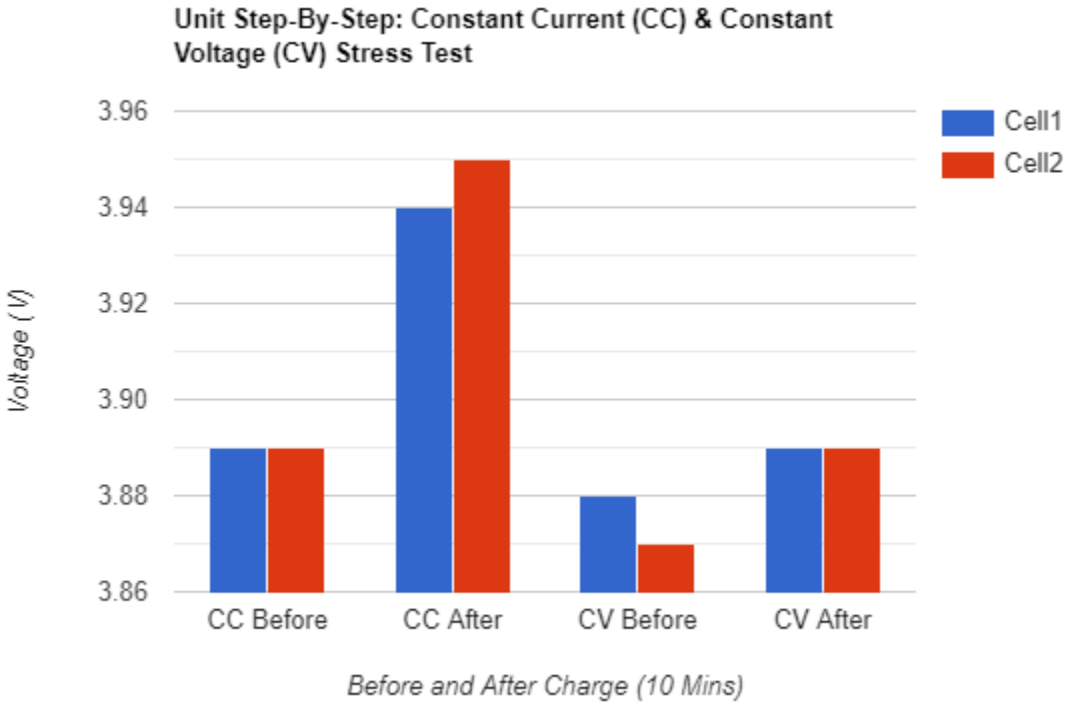


Figure 12. Graph of CC/CV test results

This graph shows the voltages of cells 1 and 2 during our test both before and after charging. Both cells were equal before we began charging with constant current and after charging for 10 minutes, cell 1 increased by 0.05V and cell 2 increased by 0.06 V. The cells were at 3.88 and 3.87 V before charging the battery with constant voltage and after the cells were both at 3.89V, indicating an increase of 0.01 and 0.02V. Both cells did increase in voltage during 10 minutes of charging and as you can see, our constant current phase charges the battery quicker than the constant voltage phase, which we expected.

Appendix M

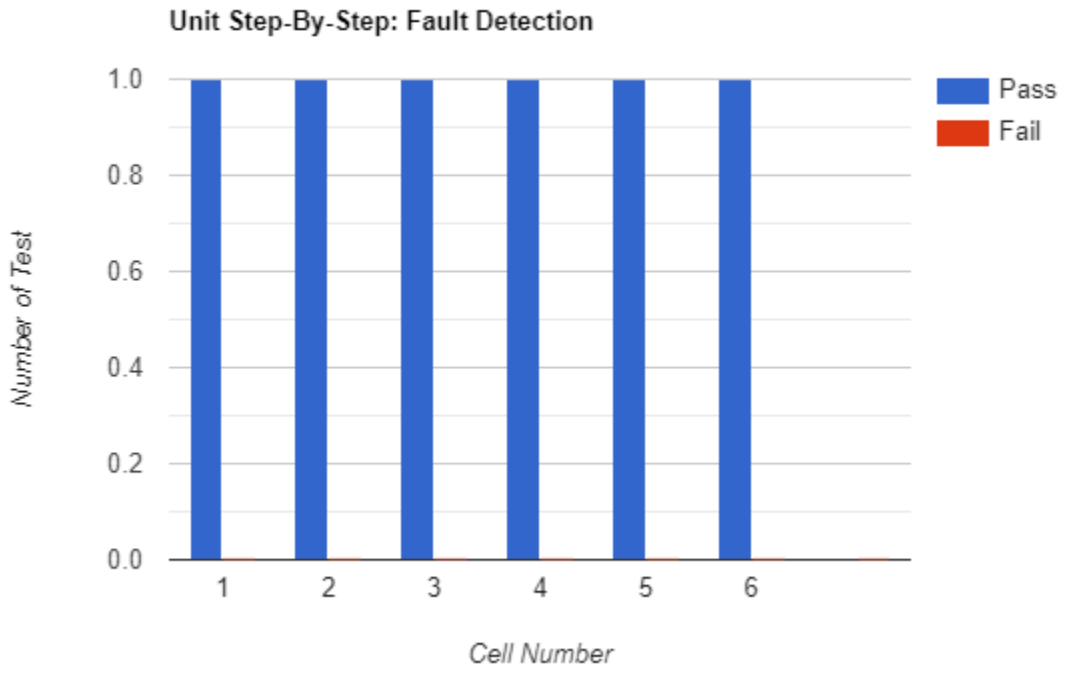


Figure 13. Graph of fault detection test results

This test was conducted on a pass or fail basis. To set up the test, each cell of a six-cell LiPo battery would be connected one-by-one and then as more cells are added a faulty cell, given by a bench power supply, would then be added to set off the fault conditions. The above diagram shows the relationship of each cell to the number of tests. The results show that each test worked properly and correctly displayed the information on the LCD.

Appendix N

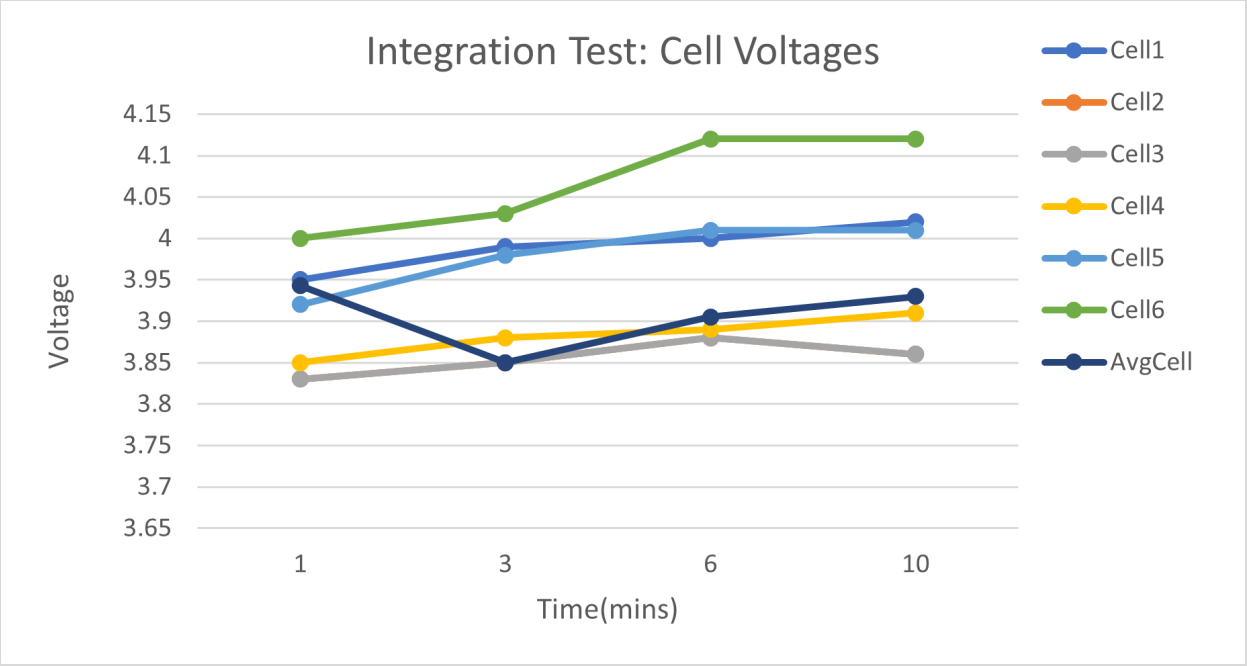


Figure 14: Graph of Integration test

This test shows the values of each of the cells and the average cell voltage through 10 minutes of testing with measurements taken every 3-4 minutes. All of our features are running at this point including the fan, rotary encoder, and LCD. Those cell values are the ones that are displayed on the LCD screen as they charge with our whole system integrated. The user at this point navigated through the menu to begin the charging process.