LPRDS-CMS-2011

Lafayette Photovoltaic Research and Development System

Cell Management System

ECE492 – Spring 2011
CONTENTS

Executive Summary ................................................................................................................................................................ 4
Introduction – Project Overview....................................................................................................................................... 5
LPRDS-CMS-2011 – Technical Overview .......................................................................................................................8
  Hardware ................................................................................................................................................................ ...............8
  Sensors ................................................................................................................................................................ ...............9
  Thermal Analysis ......................................................................................................................................................... 10
  Partial Bypass ................................................................................................................................................................ 17
  Redundant Temperature Safety System ............................................................................................................. 22
  Firmware ................................................................................................................................................................ ..............25
  Cell Balancing Algorithm .......................................................................................................................................... 26
  State of Charge Algorithm ........................................................................................................................................ 31
  I2C Communication ..................................................................................................................................................... 33
  Justification For Balancing ............................................................................................................................................ 36
LPRDS-CMS-2011 – General Project Requirements ................................................................................................ 39
  Budget ................................................................................................................................................................ ...................40
  Power, Voltage, Current Analysis ............................................................................................................................... 43
  GPR002 & GPR004 – Environmental & Hazardous Materials ........................................................................ 44
  GPR003 – EMI/EMC Analysis ...................................................................................................................................... 50
  GPR006 – Reliability .................................................................................................................................................... 53
  GPR007 – Maintainability ............................................................................................................................................. 60
  GPR008 – Manufacturability ........................................................................................................................................ 62
  Maintenance Manual ........................................................................................................................................................ 65
  User’s Manual................................................................................................................................................................ 100
Appendices ................................................................................................................................................................ ............ 116
  Matlab and Simulink Simulation Files ................................................................................................................... 116
    graphs.m ................................................................................................................................................................ ....... 116
    balancing_alg.m .......................................................................................................................................................... 118
    balancing_sim.mdl .................................................................................................................................................... 120
  MATLAB and Simulink Testing Files ...................................................................................................................... 121
    testSetup.mdl .............................................................................................................................................................. 121
    process_test.m ........................................................................................................................................................... 121
  Hyperterminal Script File ........................................................................................................................................... 122
    testscript.txt ................................................................................................................................................................ 122
EXECUTIVE SUMMARY

This report documents technical sections describing the design decisions that were made throughout the LPRDS-CMS development. Not all original system requirements were met. The overall scope of the project has been scaled down from its original form. However, we have successfully developed a cell management system that can be utilized in a variety of applications. The technical overview of the project describes the design decisions made by the team, as well as the justification and other aspects of the design changes made to the LPRDS-CMS. The technical overview includes the complete design of the cell management system. All subcircuits as well as all components and design choices are covered. Many of the subsystems design by previous years’ teams remain intact but are not used explicitly in our system.

The report then goes into a general project requirements section. This section of the final report consists of the documentation of some of the general project requirements associated with the LPRDS system. Included in this section are the budget, power budget, and the “ilities” reports. The ilities reports included within this section are environmental, EMI, hazmat, safety, reliability, maintainability, manufacturability, sustainability, and ethics.

The next section of the report is a maintenance manual that explains the unique technical principles and details of system operation. The maintenance manual also includes information on any advanced maintenance or calibration techniques that could be applied by an expert user. A set of schematics, pin outs, interface control documents, communication protocols, cables assignments, PCB board layouts, and the semantics of all interfaces are documented. The report concludes with Appendices that document work that was referenced throughout the report. The appendices include Matlab files, firmware, and spice simulations that were used in the design of our system.
The Lafayette Photovoltaic Research and Development System is a multi-year, multi-team Senior Electrical and Computer Engineering capstone design project. The scope of the current year’s project is an energy storage system to work with LPRDS, which is an energy management system developed by the 2009 and 2010 Senior Design Project classes. The LPRDS system obtains power from solar panels and forwards the power to a 120V AC load or an ESS. The 2011 Senior Design Project completes the ESS, incorporating a per-cell battery management system sensitive to concerns of power conservation, smart-grid technology, and rechargeable LiFePO4 cells.

Understanding the justification for designing a cell management system is an imperative to consumers as well as members of industry. As the world struggles to develop a sustainable energy future, it has become the imperative of society’s engineers to make the consumption of exhaustible resources more efficient. Global energy consumption is projected to increase by about 45% within the next 25 years (figure 2). Marked technological advances have directed industry into the “green” era, developing solar panels, wind turbines, smart grids, hybrid cards and energy-efficient battery storage systems. Increased awareness of the benefits won through power efficiency has caused universities and business to develop efficient, eco-friendly energy distribution and storage systems.

Because some renewable energy sources do not provide a constant supply, the ability to store the energy created is an important feature of an energy management system. The varying generation of power from solar panels or windmills can be handled one of two ways. Either the excess power can be sent back onto the power grid, and the producer compensated accordingly, or the producer can have his or her own energy storage system.
to store the excess power generated. Smart grid technology controls the regulation of energy through the system, determining the best times to charge and discharge the battery system or to use grid power. Therefore, the use of rechargeable battery systems, in either an energy storage system (ESS) or uninterrupted power supply (UPS) applications, is an integral part of any energy management or storage system.

There exist more choices for cell chemistry than ever before. Newer chemistries such as lithium-iron-phosphate (LiFePO₄) have significant advantages. LiFePO₄ is a heavy-metal free, environmentally friendly chemistry with a low self-discharge rate and high fidelity over cell lifetime. Managing the charge of individual cells ensures better state of health of a battery pack. Accurately balancing the cells within a pack ensures overcharge or undercharge does not occur, thus increasing the lifetime of each cell. However, due to certain characteristics of the charge and discharge curves of the LiFePO₄ cell, complex charging, discharging and state of charge algorithms are required for balancing.

The goal of this year’s project is to design a per-cell management system for a LiFePO₄ battery stack. In order to effectively design a balancing algorithm, certain failure conditions must be considered. Overcharging or over-discharging of the cells, over-current and over-temperature, and other issues are addressed by the cell balancing algorithm, current sensor and a redundant temperature safety system. Some additional requirements for this project include the ability to visually demonstrate the functionality of the system, which is accomplished with both a set of LEDs placed on the board, and an I²C interface which can be used to report the voltage, current, temperature and state of charge of each of the cells.

Specific temperature requirements must be met by the design, which is why it is important to manage the heat dissipated by the various components, resistive bypass, and the cells themselves. Thermal analysis helps determine the proper placement of the temperature sensors for the redundant temperature safety system.
The final goal of this design was to produce a one board per pack printed circuit board (OBPP PCB) which is mounted on top of a pack of four LiFePO₄ cells, offset by four aluminum and four copper standoffs. Because the cell anode and cathode terminals of the cells are made of copper and aluminum respectively, the standoffs are matched accordingly to prevent corrosion on the terminals. The goal in making the modular pack board is to allow for expandability and full system integration while still providing the ability for stand-alone functionality of charging and discharging a single, four-cell pack.

*Figure 5 – OBPP PCB, 4 LiFePO₄ cells and 8 standoffs make up the LPRDS Cell Management System*
HARDWARE

The hardware design of the OBPP PCB is split into five sections: microcontroller, bypass hardware, sensor hardware, isolation and a redundant temperature safety system. The microcontroller used in this design is a PIC 18F2423, which has A-to-D converters, I²C interface capabilities and programmable EEPROM memory. Sensor hardware is needed to be able to support the cell balancing algorithm, including a current sensor, temperature sensors and differential amplifiers to measure the voltage across the terminals of a particular cell. Op-amp buffers included in the design allow the PIC’s low-impedance A-to-D inputs to accurately digitize sensors signals. Another key component of the OBPP PCB design is the opto-isolation which helps to solve both the “floating ground problem” and isolate the data signals. This isolation better implements the partial resistive bypass solution and interconnects between OBPP boards. Since the majority of the design of the system concerns the bypass solution, temperature safety system, and thermal analysis, those topics will be handled in more detail. The physical OBPP board contains four layers, which contain components and heat sinks, 5V power supply and battery-to-battery connections, ground and additional components, respectively. The figure below shows the layout design of the OBPP PCB.

Figure 6 - Layout of OBPP PCB
The OBPP PCB contains several sensors in order to monitor the functionality of the pack. The sensors included on the board include temperature sensors, a current sensor and a several differential voltage amplifiers. The differential voltage amplifiers are what allow the PIC to measure and monitor the voltages of each of the cells to aid in the cell balancing algorithm. The current sensor is an integral part of the cell balancing algorithm and the state of charge algorithm. Based upon the direction of the current through the current sensor, the cell balancing algorithm decides whether to enter the charging or discharging state. In the state of charge algorithm, the voltage output from the current sensor is added or subtracted to or from a register in the PIC’s memory based upon the direction of the current. This coulomb counting is used to indicate the state of charge of the pack. The current sensor is rated to 30A for accurate readings, but is rated to not fail up to 150A of current. The sensors were carefully picked to be able to work with the ADCs which are available on the PIC.

One area of concern for the design of the sensor arrays, especially the temperature sensors, was where to place them so as to get the best result of the average temperature of the pack. In order to determine the best position, a thermal analysis was conducted to understand the temperature concerns that this project would face.
The thermal analysis was carried out using the multi-purpose software tool, COMSOL Multiphysics, modeling one four-cell pack with a simplified version of the OBPP. The materials in the simulation are aluminum (battery cells, battery terminals, heat sink and tray), copper (battery terminals, conductive slabs and power resistor), FR4 (circuit board) and acrylic plastic (bottom plastic frames). The main purpose of this analysis is to provide an intuitive visual thermodynamic analysis of the proposed system design, determine the best location to place the temperature sensor on the circuit board and determine the necessary heat sink for the circuit board.

There are several scenarios which are worthy of discussion: stationary analysis and time-dependent analysis, one battery cell overheating, all four battery cells overheating and the bypass resistor overheating. The last scenario should be considered with three conditions including no heat sink, thermally contacting heat sink and a non-thermally contacting heat sink.

The thermal analysis for a bypass resistor without any heat sink yielded an incredible temperature rise of about 3900 Kelvin. Clearly, the system would never be able to get this hot, since the solder holding the component on the board would melt prior to a temperature rise of this magnitude. To ensure these simulations were correct, this scenario was tested, and indeed, the bypass resistor did get hot enough to melt the solder holding the part to the power supply. Upon completion of the thermal analysis, it is apparent that a heat sink is absolutely necessary to avoid overheating the hardware (figure 6). In addition, the placement of the redundant temperature sensor would ideally be located in the central part of the circuit board.
The thermal analysis was carried out using the multi-purpose software tool COMSOL Multiphysics. This tool allowed us to model one 4-cell pack with a simplified version of the OBPP. The materials that were used were aluminum (battery cells, battery terminals, heat sink and tray), copper (battery terminals, conductive slabs and power resistor), FR4 (circuit board) and acrylic plastic (bottom plastic frames). The main purpose of this analysis is to:

- Provide an intuitive visual thermodynamic analysis of the proposed system design.
- Figure out the best location to place the temperature sensor on the circuit board is another goal to be accomplished through this analysis.
- Figure out if a heat sink is needed on the circuit board.

There were three scenarios that were simulated with both a stationary analysis and a time-dependent analysis as well. The three (3) scenarios that were done in simulation are the following:

1. One battery cell is overheating.
2. All four battery cells overheating
3. Bypass resistor overheating with no heat sink
4. Bypass resistor overheating with heat sink with resistance and heat sink not being thermally in contact
5. Bypass resistor overheating with heat sink with resistance and heat sink thermally in contact
6. The final design of the board and overall 4-pack system was modeled and simulated.

The COMSOL Multiphysics tool was used throughout the project for different purposes. At an earlier stage of the project COMSOL was used to get a sense of the thermodynamics of our model. Later on when the board was fabricated and the design was finalized COMSOL was used to model the final design as well as to confirm the Environmental requirements.

- Conclusion: After carrying out the thermal analysis we concluded that a heat sink is absolutely necessary for ensuring the integrity of the hardware. Also, the placement of the redundant temperature would ideally be located in the central part of the circuit board. This is dependent on whether the layout of the OBPP board allows it.
Stationary Analysis with one cell overheating

Heat Source is the Battery Cell
Stationary Analysis with four cells overheating

Heat Sources are the four Battery Cells
Bypass Resistor Overheating with no Heat Sink

Heat Source is Bypass Resistor
Bypass Resistor Overheating (With Convective Cooling in Heat sink)

Bypass Resistor Overheating (Heat Sink and Bypass resistor thermally touching)
Final Design Simulation (2 Bypass resistors functioning at 30°C ambient)
The hardware design implements resistive bypass in support of the cell balancing algorithm. The major reasons for the team’s choice in favor of resistive bypass over active balancing, are the following. Firstly, that the space constraints placed upon the project do not accommodate the number of components needed to satisfactorily complete an active balancing scheme. Secondly, that the time and resources required to develop, simulate and test an active balancing algorithm exceed the time frame of the project.

As indicated by its name, partial resistive bypass bypasses only a portion of the current going into the battery cell. The electrical configuration of the cell, power resistor and transistor is displayed in the figure to the right. The power resistor and transistor are placed between the terminals of the battery cell instead of between the positive terminal and pack ground. Thus, the available voltage seen by each power resistor is the range only one cell, 2.8V – 3.8V.

Therefore, the maximum current allowed to flow through the power resistor is equal to the voltage of the cell divided by the power resistor value, 1.5Ω. The resulting curve is a linear relationship between current and voltage, resulting in current values of 1.87A – 2.53A, corresponding to the 2.8V – 3.8V range.

Isolation is used in this design to bypass each individual cell without concern about the possible “floating ground problem.” Since the cells are connected in series, the total pack voltage is approximately 12V. When attempting to measure the voltage of any cell above the one closest to pack ground, the “negative” terminal for each cell is not absolute ground, but a relative ground to that cell. These relative grounds are at...
9V, 6V, 3V and 0V for cells one through four of the pack. The difficulty is that voltages seen at the nodes are greater than the voltage the microcontroller can provide to turn on the bypass transistor. For this reason, the isolators are used by each bypass circuit to create local grounds to turn on a transistor, regardless of its placement in the battery stack. An alternative technique is capacitive isolation which requires a pulse width modulated signal to turn on the transistor.
ABSTRACT

This memo covers the design choices involved in choosing a cell bypass method and the reasoning behind choosing a passive balancing solution.

TECHNICAL FINDINGS

There are two major categories of cell balancing: active and passive. Active balancing utilizes some method of shuttling charge from a higher charge cell to lower charge cells. The most common method uses inductive shuttling, using an inductor as the intermediate stage between cells, but there are also approaches using capacitors or other batteries. This method has the advantage of retaining much more power within the system, as opposed to converting it to heat in a dissipation path. Conversely, it has scalability issues, as it necessitates connections to and from each cell to each other cell through the charge shuttle. Once the number of cells grows beyond a certain point, the sheer number of interconnects are a significant design obstacle. Also, in a high power scenario, adding an inductive load to the system brings up several safety issues.

The other cell balancing method is passive balancing. This technique utilizes resistive bypass paths to divert current away from cells that are fully charged. Any excess power is burned off through a power resistor, allowing the other cells in a series stack to continue charging while preventing overcharging in those that are at capacity. Obviously, the first issue with passive balancing is the largely wasteful bypass path, where power is not only wasted, but can easily generate a large amount of heat. That said, it is much easier to design and create a balancing algorithm for this method. It can also scale much more easily as more cells are added to the system.

RECOMMENDATIONS AND DECISIONS

Currently, passive cell balancing is the most reasonable design approach. Based on the limited timeframe for design and prototyping, active balancing has too many unknowns with concern to actual operation and even theoretical simulations. Passive balancing, while less efficient, is much easier and safer to implement without exhaustive simulation and testing.
ABSTRACT

This memo outlines the design process for choosing a bypass resistor, including resistance value, power rating, case and mount style, and heatsink.

TECHNICAL FINDINGS

All initial calculations were made with a sample resistor and heatsink that were reasonable choices for the analysis. Because heat dissipation is easily our most limiting metric, we begin by determining the upper limit of both heat change and power dissipated, as well as the important physical parameters of the components. From the MATLAB script:

```matlab
%Temperature Design Curves for Bypass Resistor
%Initial assumptions using an LTO50 resistor and an HS-193 heatsink, TO-220

delT = 40 - 20; % in Celcius, 40 being the max. temp and 20 being ambient

This allows a calculation of a static power, which leads to a calculation of current based on a constant power dissipation and varying resistance, given by:

```matlab
Pstill = delT/(RThr + RThss); % Power dissipation, no airflow
Pflow = delT/(RThr + RThhsf); % Power dissipation, airflow

R = 1:.01:50; % Sample range of electrical resistance values
```

Finally, we can plot these curves to understand the relationship between current and resistance while power dissipated is constant. As a useful comparison, Ohm’s Law is included, calculated as I=V/R, using a constant 3 Volts over the range of R.

```matlab
% Plots no flow current in red, flow current in blue, 3V/R in green.
semilogx(R,Istill, 'r', R,Iflow, 'bl', R, 3./R, 'g');
```
RECOMMENDATIONS AND DECISIONS

After considering the tradeoffs associated with different resistor values, a consensus was made for the 2-4 Ω range, with the optimal being 2.2 Ω. This value leaves some room for thermal control through PWM control, as well as space for a better heatsink to be used. Also, this means even in the worst case scenario, the system should still not exceed 50 °C. Alternative heatsinks could be used to expand this range even further.
The redundant temperature safety system (RTSS) is vitally important for the safe operation of the fully-integrated system. The RTSS has independent functionality to shut down the entire system when temperature exceeds a $T_{max}$ threshold. The value of this threshold is determined by a resistor placed on the board, appropriately sized by the following equation:

$$R_{set} = \left( \frac{39M}{T_{max} + 281.6} \right) - 90.3K$$

The safety loop which connects the OBPPs is held nominally at a constant high until one of the thermostatic switches used in the design of the RTSS pulls it low, thus interrupting the line. This then trips the overall safety loop located on the master device. The design allows RTSS outputs from multiple packs to be “wire-ored”, forming a daisy-chain which terminates at a master device capable of providing the appropriate hardware to trip the safety line.
ABSTRACT

The purpose of this memo is to present the process and ideas behind the redundant temperature safety system. Due to a change in our requirements, our team did not construct an ESS controller board meaning the current design was not implemented by us, but should be used in future designs. A voltage regulator, temperature sensing chip, capacitor, and an optocoupler, will be added to each OBPP board. The connection of the OBPP boards runs through a comparator and triggers a switch of the safety loop on the Master Slave Board. This system is designed to shut down the system when there are temperatures that exceed 65°C (45°C above the ambient temperature) on any of the OBPP boards. This is 5°C higher than the requirement in the Statement of Work stating a maximum temperature of 60°C above ambient (60°C). There is a 2°C setpoint accuracy for the temperature sensing chip and our chips and circuits on the OBPP all have the ability to run at 60°C. We have raised the temperature for the safety trigger activation to 65°C in order to provide a buffer of 2 - 3°C and limit any false positives.

TECHNICAL FINDINGS

The value of the $R_{SET}$ resistor determines the temperature at which the AD22105 chip becomes active. From the data sheet, this resistance can be determined by the equation

$$R_{SET} = \frac{39 M\Omega \cdot ^\circ C}{T_{SET} \cdot ^\circ C + 281.6^\circ C} = 90.3 \, k\Omega.$$ 

The $R_{SET}$ value which allows an activation temperature of 65°C is 22.1 kΩ.

RECOMMENDATIONS AND DECISIONS

The 12 Volt DC supply on the OBPP will run through an LM2936 Voltage Regulator providing the AD22105 Resistor Programmable Thermostatic Switch with a source voltage of 5 Volts. As shown in the attached diagram, a capacitor should run in parallel to the AD22105 in order to lower the noise on the signal. The $R_{SET}$ value should be 22.1 kΩ to allow an activation temperature of 65°C. The OUT pin is connected to the $R_{PULL-UP}$ pin, passed through a 6N135 Optocoupler and tied to the output from the temperature sensing system on each of the OBPP boards. If the signal from any of the OBPP boards is below a certain threshold voltage (meaning an activated temperature sensor), then the connection between all the OBPP boards will be driven low. This wire passes through another 6N136 Optocoupler and a comparator and in the case of temperature sensor activation, closes a switch to trigger the safety loop. This Redundant Temperature Safety System is designed for multiple packs and a connection to the ESS controller board. This system is not designed for a stand-alone battery pack and must be connected to the ESS controller board.

ATTACHED DOCUMENTS

The document “Redundant Temperature Safety System Schematic.pdf” can be opened with Adobe Acrobat Pro and shows the general outline of the system, including connections between OBPP boards and the connection to the Safety Relay.
The hardware design of the system is certainly important, but the majority of the functionality of the OBPP is controlled in software. The firmware design of the OBPP needs to control the cell balancing algorithm, the state of charge algorithm, and the I²C communication between the master device and OBPP. The hardware supplies the foundation upon which the software can function and complete the remaining requirements of the project.
The cell balancing algorithm incorporates the partial resistive bypass hardware mentioned earlier, while constantly checking the voltage, current and temperature sensor signals to ensure that the pack is operating within the specified parameters. In the algorithm are the decisions made for each individual cell, which are then applied to every cell in the pack. The design reflects the decision for the cells not to balance in one cycle, but rather for their states of charge to converge to within about 5% of each other over a number of charge and discharge cycles. The deviation from highest state of charge to lowest state of charge within the pack determines how many cycles are needed for balancing.

Initially, the firmware checks the current sensor to see which direction the current is flowing. Based upon that value, the firmware will then transition either into the “charging” or “discharging” state. Simply indicating a change in direction may not be enough in this case to indicate a change of state for the system. The value used to determine the charging or discharging state must be large enough so that trickle or parasitic currents do not send a faulty signal to the firmware to indicate a change in current direction by accident. Within the charging and discharging states, the firmware saves the active state in order to be able to detect a change of charging state later.

If the current sensor detects sufficient current in the positive direction, the firmware enters the “charging” state. In this state, the firmware checks voltage and temperature sensor values to ensure that they are do not exceed the variables $v_{\text{max}}$ or $T_{\text{max}}$ respectively. If exceeded, firmware transitions to the “off” state and awaits a reset. If not, the firmware checks three difference values by subtracting from the active cell voltage the voltages of the other three cells in the pack. If any of these difference values are greater than the variable $\Delta v$, then that cell is put into bypass for a time, $t_{\text{bypass}}$. Once bypassing, the firmware again checks voltage and temperature sensor values, and if they exceed $v_{\text{max}}$ or $T_{\text{max}}$, transitions to the “off” state. If not exceeded, the firmware checks to see if the time, $t_{\text{bypass}}$, has expired, and if so, transitions to the “charging” state, unless the current direction has changed while the program was in bypass, in which case, the firmware resets.

If the current sensor detects a large enough current in the negative direction, the firmware will enter the “discharging” state. This state is significantly less complex because bypassing is not implemented while discharging the cell. The firmware then checks if the temperature is greater than $T_{\text{max}}$ or the voltage is lower than $v_{\text{min}}$, and if so, goes to the “off” state. If the thresholds are not passed, the firmware loops back to the “discharging” state while checking to ensure that the current direction indicates discharging. If the current

<table>
<thead>
<tr>
<th>Variable</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>$v_{\text{max}}$</td>
<td>3.8V</td>
</tr>
<tr>
<td>$v_{\text{min}}$</td>
<td>2.8V</td>
</tr>
<tr>
<td>$\Delta v$</td>
<td>50mV</td>
</tr>
<tr>
<td>$T_{\text{max}}$</td>
<td>100°C</td>
</tr>
<tr>
<td>$t_{\text{bypass}}$</td>
<td>20 min</td>
</tr>
</tbody>
</table>

variable names and suggested values
direction has changed, then the firmware resets. When the firmware enters the “off” state, the firmware immediately turns off all bypassing and waits for a change in charging state or a reset signal.

The team ran several simulations to test the cell balancing algorithm. They demonstrate the advantages that balancing via partial resistive bypass can provide. The first simulation balanced four cells with initial states of charge of 40%, 25%, 12% and 10% respectively. During simulation, the difference between the highest and lowest cells decreases to about 5% after only four charge and discharge cycles by implementing the balancing algorithm on the charge cycle only.

In figure 7, there are three places where the 40% initial cell is put into bypass as the voltage across the terminals of the battery drops significantly enough for a visible change. Additionally, in the first and third areas where the cells go into bypass, near the end of the bypass time, the cell voltages begin to converge.

In order to prove the robustness of the cell balancing algorithm, the team ran special case simulations to show that the algorithm can handle extreme cases where one cell is much higher than a cluster of three at a lower voltage (figure 8). Figure 8 displays the state of charge and cell voltage graphs of a pack whose cells' initial states of charge are 90%, 25%,
12% and 10%. This figure also shows that the deviation between high and low cell does increase the number of cycles which the algorithm takes to balance the cells of the pack. For example, approximately 16 cycles are required to balance the cells in this special scenario, whereas the first simulation proved to balance the cells in only four cycles.

(a) Voltage and (b) SOC of pack with initial SOCs of cells from each other
Almost all scenarios converge eventually. However, in extreme cases (e.g. a grouping of 99%, 3%, 2% and 1% states of charge) the algorithm may converge slowly as to be not worth the time to balance this type of pack. The only scenario which the cell balancing algorithm does not balance is when the high cell is 100% and the low cell is 0%. This occurs because the check for a full or empty state passes and the firmware transitions to the “off” state and remains in an infinite loop.
Charging? [T/F]

Charging

High Temp or High V? [T/F]

Current cell V > any other by > 50mV? [T/F]

bypass

High Temp or High V? [T/F]

Time limit reached? [T/F]

Dis-charging?

Discharging

High Temp or Low V? [T/F]

Was charging, now discharging or was discharging, now charging? [T/F]

OFF

reset

Cell Balancing Algorithm Flow Diagram
STATE OF CHARGE ALGORITHM

One of the requirements of the system is a “fuel gauge” indicating the state of charge of a battery at any time. Some methods of obtaining this calculation include voltage verses charge characteristics, electrochemical impedance spectroscopy, and coulomb counting. Unfortunately, cell voltage characteristics alone are not useful for determining state of charge because cell voltage is nearly constant over much of the charge range. Electrochemical impedance spectroscopy is difficult to implement as a state of charge algorithm because it requires characterizing the impedance response of a cell to both changes in temperature, current and voltage across the cell. Each of these cause changes in the impedance response of the cell which leads to extensive look-up tables or complex algorithms to compensate for the various changes inherent to this type of state of charge calculation. On the other hand, coulomb counting is a viable option based upon the active hardware and microcontroller implemented on the OBPP. A register in the IC’s memory is used to integrate the value of current flowing in or out of the cell. The integration time step is determined by a timer interrupt routine.

In order to initialize the state of charge in the “fuel gauge”, the firmware sets the register in the PIC memory to 300, indicating a 50% SOC. Once this reference has been set, the firmware can then begin to charge or discharge, adding or subtracting the value of current seen by the current sensor to or from the register in the PIC’s memory. For every amp-hour that is discharged from the cell, the cell’s state of charge in memory will reflect the change. Any time past the initial charge cycle, the firmware flow can transition between “charge” and “discharge” states, depending upon the current direction. In the “charge” cycle, similar to the “discharge” cycle, for every amp-hour that is charged to the cell, the cell’s state of charge in memory will reflect this positive addition. At the end of the “charge” cycle, the firmware sets the pack SOC to 99% and at the end of the “discharge” cycle, the firmware sets the pack SOC to 1%. For monitoring the operation of the OBPP, the cells’ states of charge are available to user I2C interface commands.
Current Integration:

Set SOC to 50%:
memory[9] = 18000

Set SOC to 1%:
memory[9] = 500

Set SOC to 99%:
memory[9] = 35500

Current Integration:

Sets the State of Charge (SOC) Algorithm

Technical Description of the State of Charge Algorithm

Simplified Description of the State of Charge Algorithm
The communication protocol used in this design is I²C, which obtains information from or sends commands to the OBPP PBC microprocessor. One of the requirements for the project is to obtain voltage, current and temperature information for each individual battery cell and state of charge of the whole pack. This functionality is built into the system so that a master device, whether it is a PCB board or a PC, can request any of the information gathered concerning each cell. Some other I²C possible functionality for the master device is to bypass individual cells by command, rather than automatically.

<table>
<thead>
<tr>
<th>Byte</th>
<th>Write</th>
<th>Read</th>
<th>Poss. Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>BD ADDR</td>
<td>0bXXXXXXX0</td>
<td>0bXXXXXXX1</td>
<td>0h50—0hF7</td>
</tr>
<tr>
<td>MEM ADDR</td>
<td>0hXX</td>
<td>0hXX</td>
<td>0h00—0h1B</td>
</tr>
<tr>
<td>DATA 1</td>
<td>0h0X</td>
<td>0hF0</td>
<td>0h00—0hFF</td>
</tr>
<tr>
<td>DATA2</td>
<td>0hXX</td>
<td>0h00</td>
<td>0h00—0hFF</td>
</tr>
</tbody>
</table>
Limiting this analysis to the ATMega16/32 (tentative microcontroller for the OBPP) focuses the research onto three distinct protocols: USART (Universal Synchronous and Asynchronous serial Receiver and Transmitter), SPI (Serial Peripheral Interface), and Two-Wire Serial Interface (TWI or I2C). Each of these is an established protocol used in many microcontroller environments, but I2C (and to a lesser extent, SPI) warrants the most research and development.

The criterion for evaluating a communication protocol has several aspects. Firstly, the number of connections that must be made for successful communication is important considering the limitations imposed by our mechanical design. Ease of use and ease of electrical isolation are important factors as well. Finally, a discussion of actual protocol (whether it is bidirectional, addressed, framed, synchronous or asynchronous) is an important aspect.

First, we examine the SPI protocol. This is a three or four wire interface (clock, data, and slave select) that has a master/slave configuration, where the master will use the slave select line to indicate which slave it is directing transmission at, and the slaves only respond to queries, rather than full duplex (unless the four wire configuration is used). Even with the additional wire, the slave does not initiate communication. Also, because addressing is done with a slave select line, our master controller would either need 16 independent slave select lines or use an addressing scheme in a higher software layer.

The USART interface is more flexible, with a minimum of 3 connections for full duplex synchronous transmission. Still, all of the data is on serial busses, so true full duplex communication between the master and slaves is unfeasible.

There is some nomenclature confusion with the final protocol discussed here. According to the ATMega16 documentation, it is called Two-wire Serial Interface. This protocol is much more commonly referred to as I2C, or Inter-Integrated Circuit protocol. The name change might be an attempt to avoid a copyrighted name, but whatever the reason, the underlying idea is the same. This method of communication requires two or three busses (clock, data, ground) which are pulled up to power supply voltage by resistors. Open drain/collector pins can selectively pull the line down to ground to create a signal. The actual data line is completely bidirectional, which poses unique problems when trying to electrically isolate the system. This single, bidirectional data link also forces a half duplex mode, as well as tightly controlling communication through master polling and slaves responding, while being quiet for the rest of the time. There are some design alternatives to incorporate some semblance of interrupts, but most involve adding more lines to the system.

The real deciding factor is an ancillary feature related to the TWI and the microcontroller low power consumption “sleep” mode: the most consistent method of interrupting sleep mode or low power mode is
with an interrupt generated by finding an address match on the TWI data line. Considering our stringent power consumption requirements, this is an invaluable feature.

RECOMMENDATIONS AND DECISIONS

Even considering the extra effort required to finalize a communication method on top of the existing protocol, the supplementary functions of using TWI are more than worth it when observed from a board function level. Research into isolating circuits built specifically for I2C should simplify the isolation challenge. Therefore, the focus of any additional innovation will lie in using this scheme for our master/slave communications.
JUSTIFICATION FOR BALANCING

The justification for balancing cells is simple: the more useful range that can be obtained from a pack of cells, the better performance the pack can offer to the user. If, for example, one cell is at 80% and one cell is at 20%, during unbalanced charge and discharge cycles, the deviation between the two cells will remain constant. Because of this, only a small amount of the capacity of the cells will be used. This is why it is necessary to balance the cells. If the cells are balanced, and the deviation between cells is decreased, a greater amount of cell capacity will be able to be used.

From looking at the state of charge or voltage graphs, it may be difficult to see the effects of balancing or to see noticeable convergence. This is why it is necessary to use other methods other than looking at the SOC and voltage curves for a charging or discharging cell. The method used is to look at the standard deviation of the cells while converging.

Figure 1 – Unbalanced charge/discharge characteristic of 4 LiFePO₄ cells

Figure 2 – (a) voltage and (2) SOC characteristics of cells beginning with 90%, 25%, 12% and 10% state of charges, showing convergence of the cells
In order to do the standard deviation calculation, a MATLAB program was used to parse the SOC data from the simulation done to obtain figure 2. The data obtained was in units of %SOC, which needed to be converted to SOC, amps, watts and finally joules to be able to obtain a good metric which could be compared to the amount of work done by the bypass resistors. As is demonstrated by the graph in figure 3, the period of time when the bypasses are activated is the same point in time when the standard deviation of the pack decreases. The two graphs are overlaid in figure 4 to show that the slopes of both the imbalance of the cells and the work done by the bypass resistors decrease in magnitude as time progresses.
In order to understand the efficiency of the system, the group performed an analysis of the system in terms of the imbalance of the cells (the standard deviation in Joules) and the work done by the bypass resistors to reduce the imbalance. The result was the following merit of efficiency in Joules/Joule:

\[
\eta(J/J) = \frac{\max(\sigma) - \min(\sigma)}{\max(J) - \min(J)} = \frac{3.3483 \times 10^6 - 1.4917 \times 10^5}{1.3847 \times 10^5 - 0} = 23.1 \text{ J/J}
\]

For the simulation, the efficiency of the system was 23.1 J/J. In the actual verification of the system in ATR T001, the efficiency of the system was about 8.5 J/J. The reason for this difference is due to the fact that the amount of imbalance in the test was less than that in the simulation.

Interestingly, the longer the pack balances, the less effective it is. This makes sense, since having cells with SOCs at 100% and 0% would take an incredibly long time to balance and would also not be an efficient use of the design.
LPRDS-CMS-2011 – GENERAL PROJECT REQUIREMENTS

As indicated by the statement of work for the LPRDS-CMS-2011 project, there are specific general project requirements which must be met by the design. Analyses were conducted of these areas of concern and reported to show that the system is in compliance with the stated requirements. In this section are analyses of the budget, power, voltage and current usage, environmental and hazardous materials, EMI/EMC concerns, reliability, maintainability and manufacturability,
### BUDGET

<table>
<thead>
<tr>
<th>Qty</th>
<th>Vendor Part</th>
<th>Description</th>
<th>Unit Price</th>
<th>Total Price</th>
</tr>
</thead>
<tbody>
<tr>
<td>12</td>
<td>FJB102TMTC-ND</td>
<td>Power BJT</td>
<td>$1.06</td>
<td>$12.72</td>
</tr>
<tr>
<td>12</td>
<td>PWR163S-25-1R50F-ND</td>
<td>Power Resistor</td>
<td>$2.64</td>
<td>$31.68</td>
</tr>
<tr>
<td>3</td>
<td>DUM2250ARWZ-ND</td>
<td>I2C isolator</td>
<td>$6.00</td>
<td>$18.00</td>
</tr>
<tr>
<td>3</td>
<td>ACS7714LLCTR-20A-T</td>
<td>Current Sensor</td>
<td>$4.73</td>
<td>$14.19</td>
</tr>
<tr>
<td>6</td>
<td>296-26730-1-ND</td>
<td>Dual Rail-to-Rail Op Amp</td>
<td>$1.93</td>
<td>$11.58</td>
</tr>
<tr>
<td>12</td>
<td>160-1786-1-ND</td>
<td>Single Opto-Isolator</td>
<td>$1.44</td>
<td>$17.28</td>
</tr>
<tr>
<td>9</td>
<td>LM2936DT-5.0-ND</td>
<td>Voltage Regulator</td>
<td>$2.39</td>
<td>$21.51</td>
</tr>
<tr>
<td>3</td>
<td>390-1086-5-ND</td>
<td>Dual Opto-Isolator</td>
<td>$3.80</td>
<td>$11.40</td>
</tr>
<tr>
<td>3</td>
<td>3522-2K-ND</td>
<td>Blade Fuse Clip</td>
<td>$1.02</td>
<td>$3.06</td>
</tr>
<tr>
<td>25</td>
<td>F1018-ND</td>
<td>Blade Fuse</td>
<td>$0.77</td>
<td>$19.25</td>
</tr>
</tbody>
</table>

Grand Total: $160.67

<table>
<thead>
<tr>
<th>Qty</th>
<th>Vendor Part</th>
<th>Description</th>
<th>Unit Price</th>
<th>Total Price</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>I2CKIT1</td>
<td>I2C starter kit</td>
<td>$89.95</td>
<td>$89.95</td>
</tr>
</tbody>
</table>

Grand Total: $89.95

<table>
<thead>
<tr>
<th>Qty</th>
<th>Vendor Part</th>
<th>Description</th>
<th>Unit Price</th>
<th>Total Price</th>
</tr>
</thead>
<tbody>
<tr>
<td>50</td>
<td>541-100KCCT-ND</td>
<td>1/8W 0805 100k Resistor, Cut Tape</td>
<td>$0.05</td>
<td>$2.41</td>
</tr>
<tr>
<td>1000</td>
<td>541-10KCCT-ND</td>
<td>1/8W 0805 10k Resistor, Cut Tape</td>
<td>$0.02</td>
<td>$18.85</td>
</tr>
<tr>
<td>50</td>
<td>541-22.1KCCT-ND</td>
<td>1/8W 0805 22.1k Resistor, Cut Tape</td>
<td>$0.05</td>
<td>$2.41</td>
</tr>
<tr>
<td>50</td>
<td>541-150CCT-ND</td>
<td>1/8W 0805 150 Resistor, Cut Tape</td>
<td>$0.05</td>
<td>$2.41</td>
</tr>
<tr>
<td>50</td>
<td>541-510CCT-ND</td>
<td>1/8W 0805 510 Resistor, Cut Tape</td>
<td>$0.05</td>
<td>$2.41</td>
</tr>
<tr>
<td>100</td>
<td>587-1268-1-ND</td>
<td>1 nF 0805 Capacitor, Cut Tape</td>
<td>$0.12</td>
<td>$12.11</td>
</tr>
<tr>
<td>100</td>
<td>587-1300-1-ND</td>
<td>10 uF 0805 Capacitor, Cut Tape</td>
<td>$0.14</td>
<td>$14.38</td>
</tr>
<tr>
<td>3</td>
<td>AD22105ARZ-ND</td>
<td>RTSS Temp Sensor</td>
<td>$2.99</td>
<td>$8.97</td>
</tr>
<tr>
<td>12</td>
<td>MCP9700T-E/TTCT-ND</td>
<td>Temperature Sensor</td>
<td>$0.25</td>
<td>$3.00</td>
</tr>
<tr>
<td>10</td>
<td>516-1309-ND</td>
<td>Through-Hole LED, Yellow</td>
<td>$0.42</td>
<td>$4.20</td>
</tr>
<tr>
<td>10</td>
<td>516-1792-1-ND</td>
<td>Through-Hole LED, Green</td>
<td>$0.46</td>
<td>$4.62</td>
</tr>
<tr>
<td>20</td>
<td>516-1791-1-ND</td>
<td>Through-Hole LED, Red</td>
<td>$0.46</td>
<td>$9.24</td>
</tr>
</tbody>
</table>

Grand Total: $85.01
<table>
<thead>
<tr>
<th>Qty</th>
<th>Vendor Part</th>
<th>Description</th>
<th>Unit Price</th>
<th>Total Price</th>
</tr>
</thead>
<tbody>
<tr>
<td>20</td>
<td>PM103MBALST</td>
<td>1x3 PIN Straight-PCB Male Connector</td>
<td>$3.08</td>
<td>$61.60</td>
</tr>
<tr>
<td>20</td>
<td>PM103FOOLOO</td>
<td>1x3 PIN Straight Female Connector</td>
<td>$1.87</td>
<td>$37.40</td>
</tr>
</tbody>
</table>

Grand Total: $99.00

<table>
<thead>
<tr>
<th>Qty</th>
<th>Vendor Part</th>
<th>Description</th>
<th>Unit Price</th>
<th>Total Price</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>PIC18F2423-I/SO-ND</td>
<td>PIC 18F2423 SOIC Microcontr.</td>
<td>$5.52</td>
<td>$55.20</td>
</tr>
<tr>
<td>50</td>
<td>541-51.0KCCT-ND</td>
<td>1/8W 0805 51k Resistor, Cut Tape</td>
<td>$0.05</td>
<td>$2.41</td>
</tr>
<tr>
<td>50</td>
<td>541-33.0KCCT-ND</td>
<td>1/8W 0805 33k Resistor, Cut Tape</td>
<td>$0.05</td>
<td>$2.41</td>
</tr>
<tr>
<td>50</td>
<td>541-470CCT-ND</td>
<td>1/8W 0805 470 Resistor, Cut Tape</td>
<td>$0.05</td>
<td>$2.41</td>
</tr>
<tr>
<td>9</td>
<td>296-26730-1-ND</td>
<td>Dual Rail-to-Rail Op Amp</td>
<td>$1.93</td>
<td>$17.37</td>
</tr>
<tr>
<td>20</td>
<td>425-2077-1-ND</td>
<td>PC123 photocoupler</td>
<td>$0.40</td>
<td>$8.00</td>
</tr>
</tbody>
</table>

Grand Total: $87.80

<table>
<thead>
<tr>
<th>Qty</th>
<th>Dimension</th>
<th>Description</th>
<th>Unit Price</th>
<th>Total Price</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1/2&quot; x 6' Rod Hex</td>
<td>Aluminum</td>
<td>$25.42</td>
<td>$25.42</td>
</tr>
<tr>
<td>1</td>
<td>1/2&quot; x 3' Rod Hex</td>
<td>Copper</td>
<td>$74.29</td>
<td>$74.29</td>
</tr>
<tr>
<td>1</td>
<td>sheet, 2&quot; x 10'</td>
<td>Stainless Steel</td>
<td>$25.54</td>
<td>$25.54</td>
</tr>
<tr>
<td>1</td>
<td>Special sizing</td>
<td>Tap Set</td>
<td>$73.74</td>
<td>$73.74</td>
</tr>
</tbody>
</table>

Grand Total: $198.99

<table>
<thead>
<tr>
<th>Qty</th>
<th>Vendor Part</th>
<th>Description</th>
<th>Unit Price</th>
<th>Total Price</th>
</tr>
</thead>
<tbody>
<tr>
<td>75</td>
<td>879-PM16S12S32</td>
<td>Crimp Connector</td>
<td>$0.67</td>
<td>$50.25</td>
</tr>
<tr>
<td>12</td>
<td>512-TIP102</td>
<td>TO-220 Power BJT</td>
<td>$0.69</td>
<td>$8.28</td>
</tr>
<tr>
<td>12</td>
<td>652-PWR220T-20-1R50F</td>
<td>TO-220 1.5 Ohm Power Resistor</td>
<td>$3.57</td>
<td>$42.84</td>
</tr>
<tr>
<td>10</td>
<td>810-FK26CGG2A562J</td>
<td>5.6 nF 100V Capacitor</td>
<td>$0.40</td>
<td>$4.00</td>
</tr>
<tr>
<td>25</td>
<td>579-MCP6024-I/P</td>
<td>MCP6024 Op Amp DIP</td>
<td>$1.59</td>
<td>$39.75</td>
</tr>
</tbody>
</table>

Grand Total: $145.12
<table>
<thead>
<tr>
<th>Qty</th>
<th>Vendor Part</th>
<th>Description</th>
<th>Unit Price</th>
<th>Total Price</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>620-1225-1-ND</td>
<td>Current Sensor - 30A</td>
<td>$4.73</td>
<td>$23.65</td>
</tr>
<tr>
<td>5</td>
<td>620-1229-1-ND</td>
<td>Current Sensor - 5A</td>
<td>$4.73</td>
<td>$23.65</td>
</tr>
<tr>
<td>3</td>
<td>AD22105ARZ-ND</td>
<td>RTSS Temp Sensor</td>
<td>$3.72</td>
<td>$11.16</td>
</tr>
<tr>
<td>100</td>
<td>BER220-ND</td>
<td>TO-220 Isolator</td>
<td>$0.13</td>
<td>$13.34</td>
</tr>
<tr>
<td>1000</td>
<td>478-3193-ND</td>
<td>CAP CER .10UF 50V 20%</td>
<td>$0.04</td>
<td>$44.80</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td><strong>Grand Total:</strong> $116.60</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Qty</th>
<th>Vendor Part</th>
<th>Description</th>
<th>Unit Price</th>
<th>Total Price</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>94735A210</td>
<td>Black Nylon 4-40 screw</td>
<td>$4.91</td>
<td>$4.91</td>
</tr>
<tr>
<td>1</td>
<td>93835A320</td>
<td>PTFE Shoulder Spacer w/ Flange</td>
<td>$9.71</td>
<td>$9.71</td>
</tr>
<tr>
<td>50</td>
<td>92510A690</td>
<td>Aluminum unthreaded spacer</td>
<td>$0.22</td>
<td>$11.00</td>
</tr>
<tr>
<td>1</td>
<td>91772A110</td>
<td>SS 4-40 screw</td>
<td>$4.16</td>
<td>$4.16</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td><strong>Grand Total:</strong> $29.78</td>
</tr>
</tbody>
</table>

- Power Supply: $160
- Boards-First spin: $200
ABSTRACT

The tables below show the current, voltage, and power dissipated in each of the major components on our board.

<table>
<thead>
<tr>
<th>COMPONENTS</th>
<th>Name Description</th>
<th>$I_{\text{dis}}$(A)</th>
<th>$V_{\text{dis}}$(V)</th>
<th>$P_{\text{dis}}$(W)</th>
<th>#</th>
<th>Total $P_{\text{dis}}$(W)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>ACS714LLCTR-20A-T Current Sensor</td>
<td>0.01</td>
<td>5</td>
<td>0.05</td>
<td>1</td>
<td>0.05</td>
</tr>
<tr>
<td>2</td>
<td>F1018-ND Blade Fuse</td>
<td>15</td>
<td>0.025</td>
<td>0.378</td>
<td>1</td>
<td>0.378</td>
</tr>
<tr>
<td>3</td>
<td>425-2077-1-ND PC123 photodetector</td>
<td>0.02</td>
<td>1.2</td>
<td>0.024</td>
<td>4</td>
<td>0.096</td>
</tr>
<tr>
<td>4</td>
<td>541-100KCCT-ND 1/8W 0805 100k Resistor Cut Tape</td>
<td>0.00005</td>
<td>5</td>
<td>0.0003</td>
<td>20</td>
<td>0.005</td>
</tr>
<tr>
<td>5</td>
<td>541-10KCCT-ND 1/8W 0805 10k Resistor Cut Tape</td>
<td>0.0005</td>
<td>5</td>
<td>0.0025</td>
<td>10</td>
<td>0.025</td>
</tr>
<tr>
<td>6</td>
<td>541-22.1KCCT-ND 1/8W 0805 22.1k Resistor, Cut Tape</td>
<td>0.00023</td>
<td>5</td>
<td>0.0011</td>
<td>10</td>
<td>0.01132217</td>
</tr>
<tr>
<td>7</td>
<td>541-75.0CCT-ND 1/8W 0805 75 Resistor Cut Tape</td>
<td>0.02667</td>
<td>2</td>
<td>0.0533</td>
<td>10</td>
<td>0.533333333</td>
</tr>
<tr>
<td>8</td>
<td>541-510CCT-ND 1/8W 0805 510 Resistor Cut Tape</td>
<td>0.0098</td>
<td>5</td>
<td>0.049</td>
<td>10</td>
<td>0.490196078</td>
</tr>
<tr>
<td>9</td>
<td>516-1309-ND Through-Hole LED, Yellow</td>
<td>0.0023</td>
<td>5</td>
<td>0.0115</td>
<td>1</td>
<td>0.0115</td>
</tr>
<tr>
<td>10</td>
<td>516-1792-1-ND Through-Hole LED, Green</td>
<td>0.0023</td>
<td>5</td>
<td>0.0115</td>
<td>1</td>
<td>0.0115</td>
</tr>
<tr>
<td>11</td>
<td>516-1791-1-ND Through-Hole LED, Red</td>
<td>0.0023</td>
<td>5</td>
<td>0.0115</td>
<td>5</td>
<td>0.0575</td>
</tr>
<tr>
<td>12</td>
<td>652-PWR220T-20-1R50F TO-220 1.5 Ohm Power Resistor</td>
<td>2.13333</td>
<td>3.2</td>
<td>6.8267</td>
<td>4</td>
<td>27.30666667</td>
</tr>
<tr>
<td>13</td>
<td>296-26730-1-ND Dual Rail-to-Rail Op Amp</td>
<td>0.0001</td>
<td>12</td>
<td>0.0012</td>
<td>5</td>
<td>0.006</td>
</tr>
<tr>
<td>14</td>
<td>LM2936DT-5.0-ND Voltage Regulator</td>
<td>0.0015</td>
<td>3.3</td>
<td>0.0045</td>
<td>1</td>
<td>0.00495</td>
</tr>
<tr>
<td>15</td>
<td>512-TIP102 TO-220 Power BJT</td>
<td>2.13333</td>
<td>0.6</td>
<td>1.28</td>
<td>4</td>
<td>5.12</td>
</tr>
</tbody>
</table>

$\text{Power}_{\text{Total}}$ = 34.1069583
The main environmental factor that was taken into consideration when designing the LPRDS Cell Management System is Relative Humidity (RH). The effects of humidity on electronic devices are the following:

- High levels of humidity may increase the conductivity of insulators leading to malfunction.
- Low levels of humidity may make materials brittle.
- Condensation can take place when an electronic device is moved from cold temperature to a lower temperature. **However, condensation is an environmental effect that is not part of the Environmental Requirement.**
- Also, low levels of humidity can favor the buildup of static electricity which can result in spontaneous shutdown.

The overall system meets the following specifications:

- Demonstrate reliable and normal functional operation in ambient lab temperatures of 15°C to 30°C, 10% to 80% RH, non-condensing. This requirement was proven with a two stage test. The two stages of verification were the following:
  - A COMSOL simulation with the updated model of our system was carried out. The scenario that was simulated was two bypass resistors being run to see how warm the overall system would get in steady-state. Figure 1 shows the result of this simulation.
  - As a complementary analysis, two bypass SMS power resistors were hooked up to a heat sink and their temperature was measured to see how warm the heat sink would get.
- The overall storage environment of 0°C to +60°C, 5% to 95% RH, non-condensing.
- Electronic components are rated for commercial temperature range of 0-70°C or better.

Every component that comprises the overall system meets the required specification about the percentage relative humidity it must handle.
COMSOL Thermal Analysis

The COMSOL simulation was done setting the ambient temperature at 30°C. This was done by setting the sides of the bottom aluminum plate as well as setting the external temperature at 30°C. The simulation that was run was the scenario were two bypass resistors connected to the same heat sink are being used and are inherently heating up. The results of the simulation are the following:

- The heat sink will reach a maximum temperature of 405.53 K.
- The components on the board where the board got the hottest are rated to withstand even higher temperature.
- Also, the redundant temperature system provides a safety mechanism in case of reaching such temperatures.

![FIGURE 1 - STATIONARY ANALYSIS FINAL OBPP BOARD DESIGN](image-url)
Heat Sink Characterization

With the purpose of complementing the COMSOL Thermal Analysis that was carried out, two bypass resistors were connected to a heat sink with the purpose of monitoring the temperature to verify the steady state behavior.

The results of this experiment did not agree with the COMSOL Thermal Analysis. The temperature of the heat sink leveled off at a value of 316K rather than 405.53K. The following are reasons that explain the discrepancies between the simulated results and the experimental data:

- The ambient temperature in the simulation was 300K, the ambient temperature (temperature of the surface) of the experiment was 292K.
- There was a cooling air current from the AC vent that was placed directly above the heat sink.
- The convective processes used in the COMSOL thermal simulations are not fully understood and leave room for error and uncertainty.
- The infrared thermometer is not 100% precise which brings even more uncertainty.
- The area below the heat sink was the part of the board with the higher temperature. The part of the board directly below the heat sink in the middle of the board where there are copper traces with solder covering had a steady-state temperature of 340K.
Hazardous Materials (GPR004)

Hazardous materials should be avoided according to the Hazmats (GPR004) Requirement. If use of a certain hazardous materials is essential and there is no non-hazardous alternative, the use of hazardous materials must comply with the attached Lafayette College Chemical Hygiene Plan.

The Lafayette College Chemical Hygiene Plan lists carcinogens, reproductive toxins, and corrosive substances as hazardous materials. A Material Safety Data Sheet (MSDS) will be provided with the safety plan for all parts requiring one. The 2011 LPRDS Cell Management System does not have any hazardous materials in its design and none of the parts require an MSDS. During the course of this project we did not explicitly purchase any chemicals listed in the Chemical Hygiene plan. Also, none of the parts we ordered had warnings or otherwise indicated that they contained these substances.

All the parts in the design are 2009/95/EC RoHS compliant. The cells that make up the system are LiFePO4 (no NiCd or Lead-acid batteries).
<table>
<thead>
<tr>
<th>Carcinogens</th>
<th>Reproductive Toxins</th>
<th>Teratogenic Substances</th>
</tr>
</thead>
<tbody>
<tr>
<td>Phosphorus hydride</td>
<td>Phosphine</td>
<td>Phosgene</td>
</tr>
<tr>
<td>Carbon disulfide</td>
<td>Cyanide</td>
<td>Cyanogen chloride</td>
</tr>
<tr>
<td>Acrylonitrile</td>
<td>Acrylamide</td>
<td>Acrylamide derivatives</td>
</tr>
<tr>
<td>1,2-Dimethylnitrosamine</td>
<td>1,2-Dimethylnitrosamine</td>
<td>1,2-Dimethylnitrosamine</td>
</tr>
<tr>
<td>Nitrosamines</td>
<td>Nitrosamines</td>
<td>Nitrosamines</td>
</tr>
<tr>
<td>1,4-Butynediole</td>
<td>1,4-Butynediole</td>
<td>1,4-Butynediole</td>
</tr>
<tr>
<td>N-Nitroso compounds</td>
<td>N-Nitroso compounds</td>
<td>N-Nitroso compounds</td>
</tr>
<tr>
<td>N,N-Dimethylamine</td>
<td>N,N-Dimethylamine</td>
<td>N,N-Dimethylamine</td>
</tr>
<tr>
<td>Dimethyl sulfide (DNSO)</td>
<td>Dimethyl sulfide (DNSO)</td>
<td>Dimethyl sulfide (DNSO)</td>
</tr>
<tr>
<td>Formaldehyde</td>
<td>Formaldehyde</td>
<td>Formaldehyde</td>
</tr>
<tr>
<td>Benzene</td>
<td>Benzene</td>
<td>Benzene</td>
</tr>
<tr>
<td>Acrylonitrile</td>
<td>Acrylonitrile</td>
<td>Acrylonitrile</td>
</tr>
<tr>
<td>Acrylamide</td>
<td>Acrylamide</td>
<td>Acrylamide</td>
</tr>
<tr>
<td>Acrylamide derivatives</td>
<td>Acrylamide derivatives</td>
<td>Acrylamide derivatives</td>
</tr>
<tr>
<td>Nitrosamines</td>
<td>Nitrosamines</td>
<td>Nitrosamines</td>
</tr>
<tr>
<td>1,4-Butynediole</td>
<td>1,4-Butynediole</td>
<td>1,4-Butynediole</td>
</tr>
<tr>
<td>N-Nitroso compounds</td>
<td>N-Nitroso compounds</td>
<td>N-Nitroso compounds</td>
</tr>
<tr>
<td>N,N-Dimethylamine</td>
<td>N,N-Dimethylamine</td>
<td>N,N-Dimethylamine</td>
</tr>
<tr>
<td>Dimethyl sulfide (DNSO)</td>
<td>Dimethyl sulfide (DNSO)</td>
<td>Dimethyl sulfide (DNSO)</td>
</tr>
<tr>
<td>Formaldehyde</td>
<td>Formaldehyde</td>
<td>Formaldehyde</td>
</tr>
<tr>
<td>Benzene</td>
<td>Benzene</td>
<td>Benzene</td>
</tr>
<tr>
<td>Acrylonitrile</td>
<td>Acrylonitrile</td>
<td>Acrylonitrile</td>
</tr>
<tr>
<td>Acrylamide</td>
<td>Acrylamide</td>
<td>Acrylamide</td>
</tr>
<tr>
<td>Acrylamide derivatives</td>
<td>Acrylamide derivatives</td>
<td>Acrylamide derivatives</td>
</tr>
<tr>
<td>Nitrosamines</td>
<td>Nitrosamines</td>
<td>Nitrosamines</td>
</tr>
<tr>
<td>1,4-Butynediole</td>
<td>1,4-Butynediole</td>
<td>1,4-Butynediole</td>
</tr>
<tr>
<td>N-Nitroso compounds</td>
<td>N-Nitroso compounds</td>
<td>N-Nitroso compounds</td>
</tr>
<tr>
<td>N,N-Dimethylamine</td>
<td>N,N-Dimethylamine</td>
<td>N,N-Dimethylamine</td>
</tr>
<tr>
<td>Dimethyl sulfide (DNSO)</td>
<td>Dimethyl sulfide (DNSO)</td>
<td>Dimethyl sulfide (DNSO)</td>
</tr>
<tr>
<td>Formaldehyde</td>
<td>Formaldehyde</td>
<td>Formaldehyde</td>
</tr>
<tr>
<td>Benzene</td>
<td>Benzene</td>
<td>Benzene</td>
</tr>
<tr>
<td>Acrylonitrile</td>
<td>Acrylonitrile</td>
<td>Acrylonitrile</td>
</tr>
<tr>
<td>Acrylamide</td>
<td>Acrylamide</td>
<td>Acrylamide</td>
</tr>
<tr>
<td>Acrylamide derivatives</td>
<td>Acrylamide derivatives</td>
<td>Acrylamide derivatives</td>
</tr>
<tr>
<td>Nitrosamines</td>
<td>Nitrosamines</td>
<td>Nitrosamines</td>
</tr>
<tr>
<td>1,4-Butynediole</td>
<td>1,4-Butynediole</td>
<td>1,4-Butynediole</td>
</tr>
<tr>
<td>N-Nitroso compounds</td>
<td>N-Nitroso compounds</td>
<td>N-Nitroso compounds</td>
</tr>
<tr>
<td>N,N-Dimethylamine</td>
<td>N,N-Dimethylamine</td>
<td>N,N-Dimethylamine</td>
</tr>
<tr>
<td>Dimethyl sulfide (DNSO)</td>
<td>Dimethyl sulfide (DNSO)</td>
<td>Dimethyl sulfide (DNSO)</td>
</tr>
<tr>
<td>Organic Compounds</td>
<td>Inorganic Compounds</td>
<td></td>
</tr>
<tr>
<td>-------------------</td>
<td>---------------------</td>
<td></td>
</tr>
<tr>
<td>4-nitrophenol</td>
<td>Selenium sulphide</td>
<td></td>
</tr>
<tr>
<td>2,4-dinitrophenol</td>
<td>Nickel compounds</td>
<td></td>
</tr>
<tr>
<td>2-aminophenol</td>
<td>Lead compounds</td>
<td></td>
</tr>
<tr>
<td>Hydrazine</td>
<td>Cadmium compounds</td>
<td></td>
</tr>
<tr>
<td>Phenylhydrazine</td>
<td>Beryllium compounds</td>
<td></td>
</tr>
<tr>
<td>Azomethine</td>
<td>Thorium oxide</td>
<td></td>
</tr>
</tbody>
</table>

**Vinylic compounds:**

- Xylene
- Vinyl chloride
- Organochlorine compounds
- 3-chloro-2-methylpropane
- Chloroform
- Carbon tetrachloride
- Nitric gas
- Dichloroethylene
- 1,2-dichloroethane
- Dichloroacetylene
- Ethylene
- Formic aldehyde
- Formaldehyde
ABSTRACT

There is a project requirement regarding electromagnetic radiation radiated or conducted from the LPRDS Cell Management System. The design should comply with the US CFR Title 47 Part 15 regulations for Class A digital equipment. Compliance with subpart B of these regulations is required if there is unintentional electromagnetic radiation. Compliance with subpart of these regulations C is required if the design contains intentional radiators.

TECHNICAL FINDINGS

The 2011 LPRDS Cell Management System complies with the US CFR Title 47 Part 15 regulations for Class A digital equipment. Since there is no intentional radiation coming from the management system, subpart B of the US CFR Title 47 Part 15 regulations for Class A digital equipment was examined and verified.

The designed system fulfills the conduction specification for Class A digital equipment. Also, the proper verification was done in order to obtain the equipment authorization that is required for unintentional radiators. Furthermore, requirement number 15.105 regarding the information that needs to be disclosed to the public was also fulfilled. The information that needs to be shared with the end users will be included in the user’s manual that the LPRDS Cell Management System design team will deliver at project completion.

RECOMMENDATIONS AND DECISIONS

After looking over the Electronic Code of Federal Regulations (e-CFR), the 2011 LPRDS Cell Management System’s compliance with these regulations has been proven. Through this analysis the EMI/EMC Requirement was met.
<table>
<thead>
<tr>
<th>Carcinogenic</th>
<th>Reproductive Toxins</th>
<th>Other Agents</th>
</tr>
</thead>
<tbody>
<tr>
<td>toluene</td>
<td>1,3-butadiene</td>
<td>nitrosamines</td>
</tr>
<tr>
<td>polychlorinated diphosphates</td>
<td>phenol</td>
<td>aromatic amines</td>
</tr>
<tr>
<td>nonylphenol</td>
<td>perfluorinated compounds</td>
<td>nitrosamines</td>
</tr>
<tr>
<td>N-nitroso compounds</td>
<td>dimethylheptanone</td>
<td>aromatic compounds</td>
</tr>
<tr>
<td>nitrobenzene</td>
<td>lead compounds</td>
<td>N-nitroso compounds</td>
</tr>
<tr>
<td>tetrachloroethylene</td>
<td>lead compounds</td>
<td>N-nitrosamines</td>
</tr>
<tr>
<td>trichloroethylene</td>
<td>lead compounds</td>
<td>N-nitroso compounds</td>
</tr>
<tr>
<td>perchloroethylene</td>
<td>lead compounds</td>
<td>N-nitroso compounds</td>
</tr>
<tr>
<td>formaldehyde</td>
<td>lead compounds</td>
<td>N-nitroso compounds</td>
</tr>
<tr>
<td>Ethylene</td>
<td>lead compounds</td>
<td>N-nitroso compounds</td>
</tr>
<tr>
<td>acrylonitrile</td>
<td>lead compounds</td>
<td>N-nitroso compounds</td>
</tr>
<tr>
<td>styrene</td>
<td>lead compounds</td>
<td>N-nitroso compounds</td>
</tr>
<tr>
<td>acrylamide</td>
<td>lead compounds</td>
<td>N-nitroso compounds</td>
</tr>
<tr>
<td>vincristine</td>
<td>lead compounds</td>
<td>N-nitroso compounds</td>
</tr>
<tr>
<td>cyclophosphamide</td>
<td>lead compounds</td>
<td>N-nitroso compounds</td>
</tr>
<tr>
<td>asparagine</td>
<td>lead compounds</td>
<td>N-nitroso compounds</td>
</tr>
<tr>
<td>doxorubicin</td>
<td>lead compounds</td>
<td>N-nitroso compounds</td>
</tr>
<tr>
<td>etoposide</td>
<td>lead compounds</td>
<td>N-nitroso compounds</td>
</tr>
<tr>
<td>daunorubicin</td>
<td>lead compounds</td>
<td>N-nitroso compounds</td>
</tr>
<tr>
<td>mitomycin</td>
<td>lead compounds</td>
<td>N-nitroso compounds</td>
</tr>
<tr>
<td>nitrogen mustard</td>
<td>lead compounds</td>
<td>N-nitroso compounds</td>
</tr>
<tr>
<td>methotrexate</td>
<td>lead compounds</td>
<td>N-nitroso compounds</td>
</tr>
<tr>
<td>melphalan</td>
<td>lead compounds</td>
<td>N-nitroso compounds</td>
</tr>
<tr>
<td>cytarabine</td>
<td>lead compounds</td>
<td>N-nitroso compounds</td>
</tr>
<tr>
<td>camptothecin</td>
<td>lead compounds</td>
<td>N-nitroso compounds</td>
</tr>
<tr>
<td>bleomycin</td>
<td>lead compounds</td>
<td>N-nitroso compounds</td>
</tr>
<tr>
<td>pentostatin</td>
<td>lead compounds</td>
<td>N-nitroso compounds</td>
</tr>
</tbody>
</table>
This is a plan to fulfill the reliability requirements for the subsystems (OBPP and ESS Controller board) as well as the overall system.

The following are the reliability requirements for the project:

- Reliability must be demonstrated in the ATP both by analysis and by Inspection.
- The Mean Time Between Failures (MTBF) of the system must be greater than then one thousand (1000) hours over the system lifetime.
  - Every part of the subsystem in the full BOM must be explicitly considered in the MTBF analysis.
  - The use of MIL-HDBK-217, Bellcore TR-332, or other equivalent techniques are encouraged for the analysis.
- Parts with power dissipation over 25 mW (milliwatts) shall be identified and reliability derating of these components based

**OBPP Board Simplified Circuit**

A simplified version of the OBPP circuit board was used to carry out the reliability analysis. The following is a table of every component included on this model.

<table>
<thead>
<tr>
<th>Component Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ATMega16A-PU</td>
<td>ATMega16 Controller</td>
</tr>
<tr>
<td>FJB102</td>
<td>Power BJT</td>
</tr>
<tr>
<td>PWR163</td>
<td>Power Resistor (SMS)</td>
</tr>
<tr>
<td>257 Blade Series Fuse</td>
<td>Fuse</td>
</tr>
<tr>
<td>TLC2254</td>
<td>Rail to Rail Op-Amp (4)</td>
</tr>
<tr>
<td>TC1023</td>
<td>Temp. Sensor</td>
</tr>
<tr>
<td>LM2936</td>
<td>Voltage Regulators</td>
</tr>
</tbody>
</table>
In finding the reliability of the OBPP circuit, the Military Handbook 217F (MIL-HDBK-217F) was used. The failure rates of each component used was found by taking the product of the base failure rate and all of the pi factors. The pi factors included depend on the type of the component. The following is the general equation for finding the failure rate:

\[ \lambda_p = \lambda_b \pi_T \pi_A \pi_R \pi_S \pi_C \pi_Q \pi_E \]

Where \( \lambda_p \) is the failure rate, \( \lambda_b \) is the base failure rate and the rest are the \( \pi \)-factors of the component.

The following tables describe in detail how the MTBF was calculated for every component in the simplified circuit model. The components' failure rates because everything is connected in series. The design revolves around the microcontroller but everything is still in series.

<table>
<thead>
<tr>
<th>Reliability of a 257 Series Blade Fuse</th>
</tr>
</thead>
<tbody>
<tr>
<td>[ \lambda_p = \lambda_b \pi_E ] Failures/10^6 Hours</td>
</tr>
<tr>
<td>Basic Failure Rate, ( \lambda_b )</td>
</tr>
<tr>
<td>Environment Factor, ( \pi_E )</td>
</tr>
<tr>
<td>Failure Rate, ( \lambda_p ) (Failures/10^6 Hours)</td>
</tr>
<tr>
<td>MTBF</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Reliability of the ATMega16 Microcontroller</th>
</tr>
</thead>
<tbody>
<tr>
<td>[ \lambda_p = (C_1 \pi_T + C_2 \pi_E) \pi_Q \pi_L ] Failures/10^6 Hours</td>
</tr>
<tr>
<td>Die Complexity Failure Rate, ( C_1 )</td>
</tr>
<tr>
<td>Package Failure Rate, ( C_2 )</td>
</tr>
<tr>
<td>Environment Factor, ( \pi_E )</td>
</tr>
<tr>
<td>Temperature Factor, ( \pi_T )</td>
</tr>
<tr>
<td>Learning Factor, ( \pi_L )</td>
</tr>
<tr>
<td>Quality Factor, ( \pi_F )</td>
</tr>
<tr>
<td>Failure Rate, ( \lambda_p ) (Failures/10^6 Hours)</td>
</tr>
<tr>
<td>MTBF</td>
</tr>
</tbody>
</table>
### Reliability of the Power Resistors (SMS)

\[
\lambda_p = \lambda_b \pi_T \pi_P \pi_S \pi_Q \pi_E \quad \text{Failures/10^6 Hours}
\]

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Basic Failure Rate, (\lambda_b)</td>
<td>0.0037</td>
</tr>
<tr>
<td>Temperature Factor, (\pi_T)</td>
<td>N/A</td>
</tr>
<tr>
<td>Environment Factor, (\pi_E)</td>
<td>Ground, Benign (Gb)</td>
</tr>
<tr>
<td>Power Factor, (\pi_P)</td>
<td>22.5 W</td>
</tr>
<tr>
<td>Power Stress Factor, (\pi_S)</td>
<td>Column 1</td>
</tr>
<tr>
<td>Quality Factor, (\pi_Q)</td>
<td>Non established Reliability</td>
</tr>
<tr>
<td><strong>Failure Rate, (\lambda_p)</strong></td>
<td><strong>0.05526</strong></td>
</tr>
<tr>
<td><strong>MTBF</strong></td>
<td><strong>18.0963</strong></td>
</tr>
</tbody>
</table>

### Reliability of the LM2936 Voltage Regulator

\[
\lambda_p = \lambda_b \pi_T \pi_S \pi_C \pi_Q \pi_E \quad \text{Failures/10^6 Hours}
\]

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Basic Failure Rate, (\lambda_b)</td>
<td>Regulator</td>
</tr>
<tr>
<td>Temperature Factor, (\pi_T)</td>
<td>(T_J=150°C)</td>
</tr>
<tr>
<td>Environment Factor, (\pi_E)</td>
<td>Ground, Benign (Gb)</td>
</tr>
<tr>
<td>Contact Construction Factor, (\pi_C)</td>
<td>Metallurgically Bonded</td>
</tr>
<tr>
<td>Stress Factor, (\pi_S)</td>
<td>Column 1</td>
</tr>
<tr>
<td>Quality Factor, (\pi_Q)</td>
<td>Plastic</td>
</tr>
<tr>
<td><strong>Failure Rate, (\lambda_p)</strong></td>
<td><strong>0.1072</strong></td>
</tr>
<tr>
<td><strong>MTBF</strong></td>
<td><strong>9.324</strong></td>
</tr>
</tbody>
</table>
### Reliability of the FBJ102 Power BJT

\[
\lambda_p = \lambda_b \pi_T \pi_A \pi_R \pi_S \pi_Q \pi_E \text{ Failures} / 10^6 \text{ Hours}
\]

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Basic Failure Rate, ( \lambda_b )</td>
<td>NPN</td>
</tr>
<tr>
<td>Temperature Factor, ( \pi_T )</td>
<td>( T_J = 150^\circ \text{C} )</td>
</tr>
<tr>
<td>Application Factor, ( \pi_A )</td>
<td>Switching</td>
</tr>
<tr>
<td>Power Rating Factor, ( \pi_R )</td>
<td>( P_r = 80.0 )</td>
</tr>
<tr>
<td>Voltage Stress Factor, ( \pi_S )</td>
<td>( V_s = 0.006 )</td>
</tr>
<tr>
<td>Quality Factor, ( \pi_Q )</td>
<td>Plastic</td>
</tr>
<tr>
<td>Environment Factor, ( \pi_E )</td>
<td>Ground, Benign (GB)</td>
</tr>
<tr>
<td>**Failure Rate, ( \lambda_p )</td>
<td>( \text{Failures/10}^6 \text{ Hours} )</td>
</tr>
<tr>
<td>(Failures/10^6 Hours)</td>
<td>( 0.007786 )</td>
</tr>
<tr>
<td><strong>MTBF</strong></td>
<td>( 127.226 )</td>
</tr>
</tbody>
</table>

### Reliability of the 6N135 Optoisolator

\[
\lambda_p = \lambda_b \pi_T \pi_Q \pi_E \text{ Failures} / 10^6 \text{ Hours}
\]

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Basic Failure Rate, ( \lambda_b )</td>
<td>Single Device</td>
</tr>
<tr>
<td>Temperature Factor, ( \pi_T )</td>
<td>( T_J = 100^\circ \text{C} )</td>
</tr>
<tr>
<td>Quality Factor, ( \pi_Q )</td>
<td>Plastic</td>
</tr>
<tr>
<td>Environment Factor, ( \pi_E )</td>
<td>Ground, Benign (GB)</td>
</tr>
<tr>
<td>**Failure Rate, ( \lambda_p )</td>
<td>( \text{Failures/10}^6 \text{ Hours} )</td>
</tr>
<tr>
<td>(Failures/10^6 Hours)</td>
<td>( 0.132 )</td>
</tr>
<tr>
<td><strong>MTBF</strong></td>
<td>( 7.5757 )</td>
</tr>
</tbody>
</table>
The final definition of a failure is the following and is depicted by Figure 2:

- The failure of **ALL** the 4 power (bypass) resistors, the 4 power BJT's **AND** the 4 Op-Amp IC's.

- The failure of **EITHER** the microprocessor, the voltage regulator, the opto-isolator (I2C interface) **OR** the 257 series fuse.

The definition of a failure helps improve the Mean Time Between Failures (MTBF) and inversely decrease the failure rate. This is because of the parallel subsystem composed by the 4 battery cells. This means that for there to be a failure originating from this subsystem all 4 battery cells with their corresponding bypass mechanisms must all simultaneously fail.

Using the depicted configuration, the calculated MTBF came out to be 9975 hrs, which is roughly one year.
FIGURE 2 - RELIABILITY SCHEMATIC
FIGURE 3 - SCHEMATIC OF THE SIMPLIFIED CIRCUIT USED FOR THE RELIABILITY ANALYSIS OF THE OBPP
ABSTRACT

Necessary components of a Maintainability Report include a Mean Time To Repair (MTTR) that is less than one week over the system lifetime (five years), maintainability requirements demonstrated in the ATP by analysis and inspection. Also, important elements within this report are a list of recommended spare parts as well as sample troubleshooting procedures with timing estimates.

REPORT

Based on our knowledge of the boards designed and the current system, it is important to maintain a stock of important spare parts which are most likely to fail or need replacement. In order to be able to quickly respond to the need of part replacement, it is recommended to store extra fuses, connectors, wires, and boards.

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
<th>Vendor 1</th>
<th>Vendor 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>OBPP BOARD</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1 F1018-MD</td>
<td>Fuse Blade</td>
<td>Digi-Key</td>
<td>Littelfuse Inc</td>
</tr>
<tr>
<td>2 3522-2K-ND</td>
<td>Holder Fuse Auto Blade</td>
<td>Digi-Key</td>
<td>Keystone Electronics</td>
</tr>
<tr>
<td>3 3M D2S10-6002-AR</td>
<td>Wire-board signal connector</td>
<td>Digi-Key</td>
<td>Newark</td>
</tr>
<tr>
<td>4 3M CHG-2010-</td>
<td>Wire-board signal connector</td>
<td>Digi-Key</td>
<td>Newark</td>
</tr>
<tr>
<td>J01010-KCP</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5 APP PM103FBALST</td>
<td>1X3 Female Straight PCB Solder (Power Connector)</td>
<td>CW Distribution</td>
<td>Allied</td>
</tr>
<tr>
<td>6 APP PM103MBALST</td>
<td>1X3 PIN Straight PCB Solder (Power Connector)</td>
<td>CW Distribution</td>
<td>Allied</td>
</tr>
<tr>
<td>7 APP PM103MOOLOO</td>
<td>1X3 PIN Cable Connector (Power Connector)</td>
<td>CW Distribution</td>
<td>Allied</td>
</tr>
<tr>
<td>8 OBPP</td>
<td>Board for OBPP</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10 Wires</td>
<td>As shown in ICD-CMS-Rev2</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>---</td>
<td>---</td>
<td>---</td>
<td>---</td>
</tr>
<tr>
<td></td>
<td>ACS714LLCTR-20A-T</td>
<td>Current Sensor</td>
<td>Digi-key</td>
</tr>
<tr>
<td>12</td>
<td>93835A320</td>
<td>Shoulder spacer w/ flange</td>
<td>McMaster-Carr</td>
</tr>
<tr>
<td>13</td>
<td>92510A690</td>
<td>Aluminum Unthreaded Spacer</td>
<td>McMaster-Carr</td>
</tr>
<tr>
<td>14</td>
<td>Rubber TO-220 back pads (heatsink isolation)</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Simple troubleshooting problems are included in the User’s Manual with included steps and procedures to fix the presented problem. A likely failure which a novice could diagnose and repair using User Manual procedures may include a blown fuse or the need for a system reset. The user would be able to follow the instructions in the User Manual instructions that would exhibit how to repair the failure by demonstrating the location of the reset button the board or the steps required to replace a fuse. The diagnosis and repair of unlikely failures such as the need to reprogram the microcontroller on the OBPP or ESS board will be possible by an expert by using the resources included in the Maintenance Manual.

The expected time for the repair of likely failures such as blown fuses, broken connectors or wires, or necessary board replacements is about 2 days. This 2 day time for repair is an estimated value taking into consideration the need to remove the broken part, determine the replacement part, and perform the replacement action. It should require less than 3 hours to remove a broken component from the system and a maximum of 30 minutes to determine the equivalent part to replace the broken part. The action of replacement should require, at most, 1 full work day of time meaning the overall MTTR would be between one and two days. This MTTR will be maintained for the lifetime of the system, five years.
ABSTRACT

A major goal of the Manufacturability Report is to demonstrate a production design which shows that the project could reasonably be manufactured in quantities greater than 1000 units per year. The components used must have a sustainable supply source over the five year system lifetime to determine a successful production design. A sustainable supply source is shown within this Manufacturability Report by demonstrating that each item in the Bill of Materials is available from at least two independent suppliers. In turn, the availability of the components necessary for this project produces a successful production design. Additionally, critical components will be identified and discussed in terms of system requirements.

REPORT

The following table shows the parts needed according to the Bill of Materials and provides two independent vendors supplying each part. These parts seem to be aligned with industry trends due to their various reasons. All components are RoHS compliant, can be found from multiple vendors (distribution reliability), have low power consumption, and are delivered in standard packaging. These are signals that show that the components we use will be available in the future as they all follow current trends within industry.

<table>
<thead>
<tr>
<th>#</th>
<th>Name</th>
<th>Description</th>
<th>Vendor 1</th>
<th>Vendor 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>PIC18F2423-I/ SO-ND</td>
<td>PIC 18F2423 SOIC Microcontr.</td>
<td>Digikey</td>
<td>Mouser</td>
</tr>
<tr>
<td>2</td>
<td>425-2077-1-ND</td>
<td>PC123 photocoupler</td>
<td>Digikey</td>
<td>Sharp Microelectronics</td>
</tr>
<tr>
<td>3</td>
<td>PM103FOOLOO</td>
<td>1x3 PIN Straight Female Connector</td>
<td>Digikey</td>
<td>Mouser</td>
</tr>
<tr>
<td>4</td>
<td>541-100KCCT-ND</td>
<td>1/8W 0805 100k Resistor, Cut Tape</td>
<td>Digikey</td>
<td>Vishay/Dale</td>
</tr>
<tr>
<td>5</td>
<td>541-10KCCT-ND</td>
<td>1/8W 0805 10k Resistor, Cut Tape</td>
<td>Digikey</td>
<td>Vishay/Dale</td>
</tr>
<tr>
<td>6</td>
<td>541-22.1KCCT-ND</td>
<td>1/8W 0805 22.1k Resistor, Cut Tape</td>
<td>Digikey</td>
<td>Vishay/Dale</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>---</td>
<td>---</td>
<td>---</td>
<td>---</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>541-75.0CCT-ND</td>
<td>1/8W 0805 75 Resistor, Cut Tape</td>
<td>Digikey</td>
<td>Vishay/Dale</td>
</tr>
<tr>
<td>8</td>
<td>541-510CCT-ND</td>
<td>1/8W 0805 510 Resistor, Cut Tape</td>
<td>Digikey</td>
<td>Vishay/Dale</td>
</tr>
<tr>
<td>9</td>
<td>587-1268-1-ND</td>
<td>1 nF 0805 Capacitor, Cut Tape</td>
<td>Digikey</td>
<td>Vishay/Dale</td>
</tr>
<tr>
<td>10</td>
<td>587-1300-1-ND</td>
<td>10 µF 0805 Capacitor, Cut Tape</td>
<td>Digikey</td>
<td>Vishay/Dale</td>
</tr>
<tr>
<td>11</td>
<td>AD22105ARZ-ND</td>
<td>RTSS Temp Sensor</td>
<td>Digikey</td>
<td>Analog Devices</td>
</tr>
<tr>
<td>12</td>
<td>MCP9700T-E/TTCT-ND</td>
<td>Temperature Sensor</td>
<td>Digikey</td>
<td>Mouser</td>
</tr>
<tr>
<td>13</td>
<td>516-1309-ND</td>
<td>Through-Hole LED, Yellow</td>
<td>Digikey</td>
<td>Newark</td>
</tr>
<tr>
<td>14</td>
<td>516-1792-1-ND</td>
<td>Through-Hole LED, Green</td>
<td>Digikey</td>
<td>Octopart</td>
</tr>
<tr>
<td>15</td>
<td>516-1791-1-ND</td>
<td>Through-Hole LED, Red</td>
<td>Digikey</td>
<td>Newark</td>
</tr>
<tr>
<td>16</td>
<td>512-TIP102</td>
<td>TO-220 Power BJT</td>
<td>Mouser</td>
<td>Fairchild Semiconductor</td>
</tr>
<tr>
<td>17</td>
<td>652-PWR220T-20-1R50F</td>
<td>TO-220 1.5 Ohm Power Resistor</td>
<td>Mouser</td>
<td>Newark</td>
</tr>
<tr>
<td>18</td>
<td>DUM2250ARWZ-ND</td>
<td>I2C isolator</td>
<td>Digikey</td>
<td>Analog Devices</td>
</tr>
<tr>
<td>19</td>
<td>ACS714LLCTR-20A-T</td>
<td>Current Sensor</td>
<td>Digikey</td>
<td>Newark</td>
</tr>
<tr>
<td>20</td>
<td>296-26730-1-ND</td>
<td>Dual Rail-to-Rail Op Amp</td>
<td>Digikey</td>
<td>Texas Instruments</td>
</tr>
<tr>
<td>21</td>
<td>LM2936DT-5.0-ND</td>
<td>Voltage Regulator</td>
<td>Digikey</td>
<td>National Semiconductor</td>
</tr>
<tr>
<td>22</td>
<td>390-1086-5-ND</td>
<td>Dual Opto-Isolator</td>
<td>Digikey</td>
<td>Newark</td>
</tr>
<tr>
<td>23</td>
<td>3522-2K-ND</td>
<td>Blade Fuse Clip</td>
<td>Digikey</td>
<td>Newark</td>
</tr>
<tr>
<td>24</td>
<td>F1018-ND</td>
<td>Blade Fuse</td>
<td>Digikey</td>
<td>Newark</td>
</tr>
<tr>
<td>25</td>
<td>-</td>
<td>6x1 100 mil jumper (ICSP)</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>26</td>
<td>-</td>
<td>2x1 100 mil jumper (reset)</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>27</td>
<td>-</td>
<td>5x2 100 mil jumper (data connection)</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>28</td>
<td>-</td>
<td>Aluminum Heatsinks</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>29</td>
<td>93835A320</td>
<td>Shoulder spacer w/ flange</td>
<td>McMaster-Carr</td>
<td>Assembly Alliance</td>
</tr>
<tr>
<td>30</td>
<td>92510A690</td>
<td>Aluminum Unthreaded Spacer</td>
<td>McMaster-Carr</td>
<td>Assembly Alliance</td>
</tr>
<tr>
<td>31</td>
<td>91772A110</td>
<td>SS Pan Head Phillips Screw 4-40 1/2&quot; length</td>
<td>McMaster-Carr</td>
<td>Assembly Alliance</td>
</tr>
<tr>
<td>32</td>
<td>-</td>
<td>Copper Battery Terminal Standoffs</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>33</td>
<td>-</td>
<td>Aluminum Battery Terminal Standoffs</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>
Additionally, the major critical components in our system include the resistor used to set the safety loop activation temperature, the threshold voltage for the cell balancing algorithm, the coulomb counting for the fuel gauge algorithm, and the resistors to manage the bypass loop. The 22.1kΩ resistor used to set the activation temperature of 65°C in the Redundant Temperature Safety System has a tolerance of 1%. This resistor tolerance provides a range of temperatures between 64.69°C – 66.06°C that would activate the AD22105 integrated circuit and trigger the safety loop. This range is acceptable as there is a 2°C setpoint accuracy allowing the minimum temperature for activation of the safety loop to be 62°C. The threshold voltage for balancing the battery cells is another system critical value. The voltage limit used in this algorithm is defined by firmware and coded onto a chip maintaining the system requirements. Furthermore, the limits used for coulomb counting are critical to the fuel gauge algorithm. On the other hand, the output of this fuel gauge algorithm is not critical to the system as this method is used to provide an overall picture of the entire battery charge level. The exact values are not important for the system function. Finally, using SPICE simulations from the file “bypassdc.cir” as the basis, the resistors used to manage the bypass loop have all been tested within 5% tolerance ranges and found to maintain all specification requirements.

CONCLUSION

The tables in the report section provide a list of all parts used in our system as well as two independent suppliers of these parts, demonstrating a sustainable supply source. Based on these sources, it is reasonable to project a successful production design and the ability to manufacture our system in quantities greater than 1000 units per year over the five year system lifetime. Finally, the tolerances of all critical components still provide results that are within system specification requirements.
MAINTENANCE MANUAL

The maintenance manual is a low level document that explains the unique technical principles and details of system operation. The maintenance manual also includes information on any advanced maintenance or calibration techniques that could be applied by an expert user. Typical faults and their fixes are documented so that a novice user can modestly maintain the system. A set of schematics, pin outs of all connectors, interface control documents, communication protocols, signal assignments of all cables, PCB board layouts, and the semantics of all interfaces are included within this manual.
LPRDS-CMS-2011
Lafayette Photovoltaic Research and Development System
Cell Management System
ECE492 – Spring 2011

Lafayette College Electrical and Computer Engineering Department
Maintenance Manual
# Table of Contents

I. Introduction .................................................................................................................. 3

II. Assembly of CMS ........................................................................................................ 3

III. Programming the PIC ................................................................................................ 5

IV. Fault Management ..................................................................................................... 10

V. Schematics .................................................................................................................... 12

VI. Pinouts of all connectors ............................................................................................ 20

VII. Semantics of interfaces ............................................................................................. 21

VIII. Battery Management ................................................................................................. 32
I. Introduction

Purpose

The purpose of the Maintenance Manual is to document the technical details of the LPRDS-CMS operation. The manual includes all source file locations, all top-level system schematics, and brief explanations of the major system components, including common issues, and troubleshooting tips.

LPRDS System Overview

The Lafayette Photovoltaic Research and Development System - Cell Management System (LPRDS-CMS) is an experimental system. It uses a photovoltaic array mounted on the roof of the Acopian Engineering Center (AEC) to provide DC power for storage in a battery array and conversion to AC power by an inverter. The system is housed in AEC room 410. A row of outlets in room AEC 400 provides power from the inverter using battery storage or a combination of PV power and batteries depending on the amount of energy required by the load. This standalone system can be monitored and controlled by using the I2C interface.

About the Cell Management System (CMS)

The Cell Management System (CMS) is comprised of the OBPP board and 4 LiFePO₄ battery cells. The main function of the OBPP is to balance the state of charge of the cells. The CMS is capable of standalone use, but is also designed to be scalable and to be easily integrated in the future into the current ESS. Under standalone conditions, packs of 4 cells each are able to be charged or discharged. During charge cycles, the OBPP attempts to balance the State of Charge (SOC) of the four cells in a pack in order to increase the lifetime and efficiency of the battery pack.
II. Assembly of CMS

These are the materials that comprise a CMS:
- 4 LiFePO₄ battery cells
- 1 OBPP board
- 8 spacers (4 copper and 4 aluminum)
- 8 screws
- 8 washers

These are the steps to follow when assembling a CMS.

**Step 1:** Screw in one (1) copper spacer and one (1) aluminum spacer onto the terminals of each of the four (4) battery cells.

Figure 1 – This figure shows a four-cell pack before (1) and after placing the standoffs
**Step 2:** Place the OBPP over the 8 standoffs placed on each terminal

![Image of OBPP being placed over standoffs]

**NOTE:** Make sure that the two (2) heat sinks with the eight (8) spacers are correctly attached to the OBPP.

**Step 3:** Place the eight (8) washers onto the OBPP.

![Image of OBPP with washers placed on top]

**NOTE:** The washers that were used are round, as opposed to the hexagonal washers depicted on the figure above.

**Step 4:** Place the eight (8) Phillips screws.

**Common Issues**

There are no known issues with the CMS assembly at this time.
III. Programming the PIC

About the PIC that was used in the design

The PIC that was used was the PIC18F2423 SOIC Microcontroller.

PIC Programming

If a Programmable Interface Controller (PIC) has an issue, and it is determined that it must be reprogrammed, the following instructions will reprogram the PIC.

Step 1: Connect the USB connector from the Microchip PICkit 2 shown on Figure 1 to the PC.

Step 2: Make sure that the arrow from the PICkit 2 showed on the left on Figure 1 matches up with the arrow on the OBPP shown on Figure 2. Once the PICkit 2 is correctly connected to the PC and to the OBPP carry out Step 3.
Step 3: Open MPLAB IDE v8.56.

Step 4: Go to Project->Open… The project that is needed will be in the following location:

S:\ECE_Scratch\ECE492-SP11\FINAL PROJECT DOCUMENTS\Firmware\OBPPFirmware.mcp

Or

LPRDS-CMS-2011 Website: http://sites.lafayette.edu/ece492-sp11/ → “Resources” → “OBPP Firmware”
**Step 5:** Go to Configure->Select Device and make sure the following Device (PIC18F2423) and Device Family (ALL) are chosen as shown on Figure 3.

*FIGURE 6 - SETTINGS FOR SELECTING THE PROPER DEVICE*
**Step 6:** Go to Configure->Configuration Bits. On the popup window check the *Configuration bits in code* option as shown on Figure 4.

![Configuration Bits](image)

**FIGURE 7 - CHECKING THE CONFIGURE BITS IN CODE OPTION.**

**Step 7:** Go to Programmer->Select Programmer-> 7 PICkit 2.
Step 8: Click on the Build All option as shown on the figure below.

Step 9: Click on the Program the target device option as shown on the figure below.
Step 10: Click on the Bring target MCLR to Vdd option as shown on the figure below.

Common Issues

- It is important to keep in mind that when the Microchip PICkit 2 is connected to the OBPP without the four battery cells being attached to the OBPP, the OBPP is powered through the PICkit 2.
IV. Fault Management

Purpose

The purpose of the Maintenance Manual is to document the technical details of the LPRDS-CMS operation. The manual includes all source file locations, all top-level system schematics, and brief explanations of the major system components, including common issues, and troubleshooting tips.

Fault Management Table

<table>
<thead>
<tr>
<th>Component</th>
<th>Method of Testing</th>
<th>Expected Result</th>
<th>Solution</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cell Voltages</td>
<td>Multimeter/Probes</td>
<td>At least 10 Amp Hours to charge</td>
<td>Replace Cell</td>
</tr>
<tr>
<td>Green LED</td>
<td>Visual</td>
<td>Green LED on when charged, blinking when charging</td>
<td>Replace Green LED</td>
</tr>
<tr>
<td>Red LED</td>
<td>Visual</td>
<td>Red LED on when discharged, blinking when discharging</td>
<td>Replace Red LED</td>
</tr>
<tr>
<td>Yellow LED</td>
<td>Visual</td>
<td>Yellow LED on when Slave mode, blinking when stand-alone</td>
<td>Replace Yellow LED</td>
</tr>
<tr>
<td>Bypass LED</td>
<td>Visual</td>
<td>Red Bypass LED on when bypassing the cell</td>
<td>Replace Red Bypass LED</td>
</tr>
<tr>
<td>$I^2$C</td>
<td>Check reading/writing</td>
<td>Able to read/write into registers</td>
<td>Reset PIC</td>
</tr>
<tr>
<td>Blown Fuse</td>
<td>Multimeter/Probes</td>
<td>Very low resistance across fuse</td>
<td>Replace Fuse</td>
</tr>
<tr>
<td>Current Sensor</td>
<td>Check cell voltages</td>
<td>Sum of cell voltages same as voltage at plug</td>
<td>Replace Current Sensor</td>
</tr>
<tr>
<td>All</td>
<td>Infrared Thermometer</td>
<td>Temperatures within specified safety range</td>
<td>Check other components</td>
</tr>
</tbody>
</table>

The voltages across each cell should never exceed 3.8 Volts and each cell should take at least 10 Amp Hours to charge. If the time to charge is too fast or the cell voltage is above 3.8 Volts, then the cell is most likely damaged and should be replaced. If an LED does not turn on or blink in response to a corresponding action, the LED may be burnt out and should be replaced. To replace an LED on the board, the LED should be unsoldered and a replacement LED should be soldered into the same position as the original LED. To test the operation of the $I^2$C, commands can be sent in order to read from registers and write into registers using Realterm. If the commands are not acknowledged correctly or don’t provide the proper response, the PIC should be reset (See Section 3 for PIC Reset Instructions). If there is a high resistance across the fuse, then the fuse has been blown and needs to be replaced. In order to replace a fuse, the burnt-out fuse must be thrown out according to the environmental procedures and replaced with a new fuse. If the sum of the cell voltages does not match the voltage at the Anderson connector plug, then there is a problem with the current sensor and the current sensor should be replaced.
V. Schematics

- Original schematics available on the site website under “Project Documents” → “Design Documents” → “Design Files and Schematics” in case changes need to be made.

OBPP Schematic

Figure from S:\ECE_Scratch\ECE492-SP11\Schematic PDF's\obpp.pdf

Or
LPRDS-CMS-2011 Website: [http://sites.lafayette.edu/ece492-sp11/](http://sites.lafayette.edu/ece492-sp11/) → “Design Documents” → “Design Files and Schematics”
Isolator Schematic

Figure from S:\ECE_Scratch\ECE492-SP11\Schematic PDF's\isolator.pdf

Or

Bypass Schematic

Figure from S:\ECE_Scratch\ECE492-SP11\Schematic PDF's\bypass.pdf

Or
Cell Stack Schematic

Figure from S:\ECE_Scratch\ECE492-SP11\Schematic PDF's\cell_stack.pdf
Or
Cell Sensor Schematic

Figure from S:\ECE_Scratch\ECE492-SP11\Schematic PDF's\cellsensor.pdf

Or

Microcontroller Schematic

Figure from S:\ECE_Scratch\ECE492-SP11\Schematic PDF's\microcontroller.pdf
Or
Unit Cell Schematic

Figure from S:\ECE_Scratch\ECE492-SP11\Schematic PDF’s\unitcell.pdf

Or

Cell Management System

Figure from S:\ECE_Scratch\ECE492-SP11\FINAL PROJECT DOCUMENTS\Schematics and Layout\Cell_Mgmt_System.pdf

Or


PCB Layout Design:
VI. Pinouts of all connectors

There are two (2) connectors on the OBPP, which are:
- Anderson connector
- Data connector

Pinout for Anderson Connector (3-pin connector)

<table>
<thead>
<tr>
<th>Pin #</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Power (+12V)</td>
</tr>
<tr>
<td>2</td>
<td>No Connection</td>
</tr>
<tr>
<td>3</td>
<td>Ground</td>
</tr>
</tbody>
</table>

Pinout for Data Connector (2x5 connector)
<table>
<thead>
<tr>
<th>Pin #</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>External Power (+5V)</td>
</tr>
<tr>
<td>2</td>
<td>External SDA (data for I²C)</td>
</tr>
<tr>
<td>3</td>
<td>External SCL (clock for I²C)</td>
</tr>
<tr>
<td>4</td>
<td>No Connection</td>
</tr>
<tr>
<td>5</td>
<td>No Connection</td>
</tr>
<tr>
<td>6</td>
<td>External RTSS (Redundant Temperature Sensing System)</td>
</tr>
<tr>
<td>7</td>
<td>External Done</td>
</tr>
<tr>
<td>8</td>
<td>Ground</td>
</tr>
<tr>
<td>9</td>
<td>No Connection</td>
</tr>
<tr>
<td>10</td>
<td>No Connection</td>
</tr>
</tbody>
</table>
VII. Semantics of interfaces (hardware & software)

I2C Commands

The following is the table of all of the instructions that were created using I2C (See I2C User Manual for more information):
### D. I2C Instruction Set

<table>
<thead>
<tr>
<th>Command #</th>
<th>Input Range</th>
<th>Value Range</th>
<th>Default Value</th>
<th>Description</th>
<th>Calculation</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x00</td>
<td>0x0000 - 0xFFFF</td>
<td>-25A - 25 A</td>
<td>R/O</td>
<td>Using a read command, this returns the <strong>current</strong> passing through the addressed board.</td>
<td>( I = \left( \frac{X}{0xFFFF} \right) \times 50 - 25 )</td>
</tr>
<tr>
<td>0x01</td>
<td>0x0000 - 0xFFFF</td>
<td>0 - 5 V</td>
<td>R/O</td>
<td>Using a read command, this returns <strong>Cell 1 Voltage</strong>.</td>
<td>( V1 = \left( \frac{X}{0xFFFF} \right) \times VDD )</td>
</tr>
<tr>
<td>0x02</td>
<td>0x0000 - 0xFFFF</td>
<td>0 - 5 V</td>
<td>R/O</td>
<td>Using a read command, this returns <strong>Cell 2 Voltage</strong>.</td>
<td>( V2 = \left( \frac{X}{0xFFFF} \right) \times VDD )</td>
</tr>
<tr>
<td>0x03</td>
<td>0x0000 - 0xFFFF</td>
<td>0 - 5 V</td>
<td>R/O</td>
<td>Using a read command, this returns <strong>Cell 3 Voltage</strong>.</td>
<td>( V3 = \left( \frac{X}{0xFFFF} \right) \times VDD )</td>
</tr>
<tr>
<td>0x04</td>
<td>0x0000 - 0xFFFF</td>
<td>0 - 5 V</td>
<td>R/O</td>
<td>Using a read command, this returns <strong>Cell 4</strong></td>
<td>Calculate voltage from returned hex value (X) as:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Voltage.</td>
<td>[ V4 = \left( \frac{X}{0xFFFF} \right) \times VDD ]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>---</td>
<td>---</td>
<td>---------</td>
<td>-------------------------------------------------</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0x05</td>
<td>0x0000 - 0xFFFF</td>
<td>R/O</td>
<td>Using a read command, this returns <strong>Cell 1 Temperature</strong>. Calculate temperature (°C) from returned hex value (X) as: [ T1 = \left( \frac{X}{0xFFFF} \times 5000 \right) - 1409.1 \div 29.6 ]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0x06</td>
<td>0x0000 - 0xFFFF</td>
<td>R/O</td>
<td>Using a read command, this returns <strong>Cell 2 Temperature</strong>. Calculate temperature (°C) from returned hex value (X) as: [ T2 = \left( \frac{X}{0xFFFF} \times 5000 \right) - 1409.1 \div 29.6 ]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0x07</td>
<td>0x0000 - 0xFFFF</td>
<td>R/O</td>
<td>Using a read command, this returns <strong>Cell 3 Temperature</strong>. Calculate temperature (°C) from returned hex value (X) as: [ T3 = \left( \frac{X}{0xFFFF} \times 5000 \right) - 1409.1 \div 29.6 ]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0x08</td>
<td>0x0000 - 0xFFFF</td>
<td>R/O</td>
<td>Using a read command, this returns <strong>Cell 4 Temperature</strong>. Calculate temperature (°C) from returned hex value (X) as: [ T4 = \left( \frac{X}{0xFFFF} \times 5000 \right) - 1409.1 \div 29.6 ]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Code</td>
<td>Description</td>
<td>Value</td>
<td>Read/Write</td>
<td>Action</td>
<td>Calculation</td>
</tr>
<tr>
<td>------</td>
<td>-------------</td>
<td>-------</td>
<td>------------</td>
<td>--------</td>
<td>-------------</td>
</tr>
<tr>
<td>0x09</td>
<td>0x01F4 – 0x34BC</td>
<td>~1% - ~99%</td>
<td>R/O</td>
<td>Using a read command, this returns the <strong>State of Charge</strong> of the pack of cells addressed.</td>
<td>Calculate the <strong>State of Charge (%)</strong> from the returned hex value (X) as: $SOC = \left(\frac{X \text{ to Decimal}}{6}\right)$</td>
</tr>
<tr>
<td>0x0A</td>
<td>0x0000 – 0xFFFF</td>
<td>0 – 65,536 min</td>
<td>R/O</td>
<td>Using a read command, this returns <strong>Total Run Time</strong> since last reset.</td>
<td>Calculate run time from returned hex value (X) as: $T = (X \text{ to Decimal})$</td>
</tr>
<tr>
<td>0x0B</td>
<td>0x0000 – 0x0FFF</td>
<td>0 – 2,048 min</td>
<td>0x0000</td>
<td>0 min / Off</td>
<td>Using a read command, this returns the <strong>State of Bypass 1</strong>. If Bypass 1 is on, Rightmost bit returned is 1, otherwise, it is 0. Using a write command, this can turn <strong>Bypass 1</strong> On for the specified amount of time, or Off.</td>
</tr>
</tbody>
</table>
| 0x0C | 0x0000 – 0x0FFF | 0 – 2,048 min | 0x0000 | Using a read command, this returns the **State of Bypass 2**
Using a write command, this can turn **Bypass 2** On for the specified amount of time, or Off |
| 0x0D | 0x0000 – 0x0FFF | 0 – 2,048 min | 0x0000 | Using a read command, this returns the **State of Bypass 3**
Using a write command, this can turn **Bypass 3** On for the specified amount of time, or Off |
| 0x0E | 0x0000 – 0x0FFF | 0 – 2,048 min | 0x0000 | Using a read command, this returns the **State of Bypass 4**
Using a write command, this can turn **Bypass 4** On for the specified amount of time, or Off |
<table>
<thead>
<tr>
<th>Register</th>
<th>Values</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x0F</td>
<td>0x0000 or 0x00FF</td>
<td>Off (0x0000) or On (0x00FF)</td>
</tr>
<tr>
<td></td>
<td>Off (0x0000)</td>
<td>Off (0x0000)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Using a read command, this returns the current State of the <strong>Done Signal</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Using a write command, this can set the <strong>Done Signal</strong> either high or low</td>
</tr>
<tr>
<td>0x10</td>
<td>0x01 – 0xFF</td>
<td>0x54</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Using a read command, this returns the <strong>Current Address</strong> of the board</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Using a write command, this can change the <strong>Current Address</strong> of the board</td>
</tr>
<tr>
<td>0x11</td>
<td>0x0000 – 0x0002</td>
<td>R/O</td>
</tr>
<tr>
<td></td>
<td>0x0000 – standby</td>
<td></td>
</tr>
<tr>
<td></td>
<td>0x0001 – charging</td>
<td></td>
</tr>
<tr>
<td></td>
<td>0x0002 – discharging</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Using a read command, this returns the current <strong>Charge State</strong> of the pack.</td>
</tr>
<tr>
<td>0x12</td>
<td>0x0000 or 0x00FF</td>
<td>0x0000</td>
</tr>
<tr>
<td></td>
<td>0x0000 – Automatic</td>
<td></td>
</tr>
<tr>
<td></td>
<td>0x00FF – Slave</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Using a read command, this returns the current <strong>Mode of the System</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Using a write</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-------</td>
<td>-------</td>
<td>-------</td>
</tr>
<tr>
<td>0x13</td>
<td>0x0000 – 0xFFFF</td>
<td>0 – 5V</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>3.8V (0xC29)</td>
</tr>
<tr>
<td>0x14</td>
<td>0x0000 – 0x0FFF</td>
<td>0 – 5V</td>
</tr>
<tr>
<td>------</td>
<td>-----------------</td>
<td>-------</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>0x15</th>
<th>0x0000 – 0x0FFF</th>
<th>0 – 255 deg C</th>
<th>0xB0A</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>Using a read command, this returns the <strong>Temperature Threshold</strong> used in the cell balancing algorithm</td>
</tr>
<tr>
<td>Address (0x16)</td>
<td>Address Range (0x0000 – 0x00FF)</td>
<td>Value Range</td>
<td>Value</td>
</tr>
<tr>
<td>---------------</td>
<td>---------------------------------</td>
<td>-------------</td>
<td>------</td>
</tr>
<tr>
<td>0x0014</td>
<td>0 – 255 minutes</td>
<td>0x0014</td>
<td></td>
</tr>
<tr>
<td>0x0029</td>
<td>0 – 5 V</td>
<td>0x0029</td>
<td>50mV</td>
</tr>
<tr>
<td>0x0014</td>
<td>0 – 65636 minutes</td>
<td>R/O</td>
<td></td>
</tr>
<tr>
<td>Address</td>
<td>Range</td>
<td>Value</td>
<td>Type</td>
</tr>
<tr>
<td>---------</td>
<td>-------</td>
<td>-------</td>
<td>------</td>
</tr>
<tr>
<td>0x19</td>
<td>0x0000 – 0xFFFF</td>
<td>0 – 65636 minutes</td>
<td>R/O</td>
</tr>
<tr>
<td>0x1A</td>
<td>0x0000 – 0xFFFF</td>
<td>0.0 – 255.255</td>
<td>R/O</td>
</tr>
<tr>
<td>0x1B</td>
<td>0x0000 – 0xFFFF</td>
<td>0 – 255</td>
<td>R/O</td>
</tr>
</tbody>
</table>

\[ T = (X \text{ to Decimal}) \]

\[ \text{Upper Byte -> Version Number} \]

\[ \text{Lower Byte -> Subversion Number} \]

\[ \text{Ex: 0204 -> Version 2.4} \]

\[ \text{SN = (X \text{ to Decimal})} \]
**The meaning of the LED’s on the OBPP**

The LED’s on the OBPP are visual signals which inform the users and the maintainers about the current status of the CMS. The following figure depicts what LED’s mean:

- Solid – Charged
- Blink – Charging
- Solid – Discharged
- Blink – Discharging
- Solid – Slave
- Blink – Stand-alone
- Solid – Bypassing

**Source File Location**

The source files for the OBPP firmware can be found in the following location on any of the AEC 4th floor computers:

S:\ECE_Scratch\ECE492-SP11\FINAL PROJECT DOCUMENTS\Firmware\......

Or

On the LPRDS-CMS-2011 website under “Resources” → “OBPP Firmware”
VIII. Battery Maintenance

About the Batteries:
The batteries which were selected are Lithium Iron Phosphate Batteries. Their nominal Voltage is 3.2V, and their capacity is 10A-h. These batteries do not need periodic maintenance. Their chemistry is not explosive and stable even under under/over charge situations. A typical service time for LiFePO4 is about 6-7 years or 3000 cycles, depending which comes first. While the cycles are not counted, we approximate that the batteries should go through 3000 cycles while in full use approximately every 3 years. For optimal use, batteries should be replaced every 3 years.

There are a few good rules of thumb when it comes to battery conditioning and maintenance. A few basic rules are:

- Cycle the battery a few times before using for the first time
- Store in a cool dry place
- If the memory effect is a problem, cycle the battery a few times every few weeks to undo the effect
- If the batteries need to be stored for a long time unused, then store them in a cool dry place and charge the batteries before their next use to account for their self discharge.

Picture of the LiFePO4 Battery:

Common Issues
There are no known issues with the LiFePO4 Batteries at this time.
The user manual is a high level document which describes the set up and operation of the LPRDS-CMS. System components, system installation, charging/discharging instructions, I²C operation, and simple troubleshooting are all included in the document. In addition to a troubleshooting matrix recommending common solutions, instructions on how to install new components or reset the board are included. The appendix includes a description of all major components: LED’s, pinouts, and connectors and their operation.
Lafayette Photovoltaic Research and Development System
Cell Management System
ECE492 – Spring 2011
I. Introduction
II. Getting Started
   A. System Components
   B. Extras
   C. Warnings
III. Installation
IV. Operating Instructions
   A. Charging
   B. Discharging
   C. I2C Operation
V. I2C Instruction Set
VI. Troubleshooting
   A. System Reset
   B. Replacing a Fuse
VII. Appendix
   A. Board Components
   B. Cell Maintenance
I. Introduction

Cell Management System (CMS)

The main function of the CMS is to balance and monitor the state of charge of a pack of four Lithium Iron Phosphate Cells. The CMS can be used in automatic mode, and can also be easily monitored and controlled via I²C communication. Each pack of cells contains a single Printed Circuit Board (PCB), which is referred to as the OBPP (One Board Per Pack). The OBPP acts as the brain of the CMS, as well as a visual indicator of the state of the CMS.

Under automatic conditions, a pack of 4 cells can be charged or discharged by the user and the CMS will automatically balance the cells’ state of charge over several cycles, thus improving the lifetime of the cells, and improving the pack performance. The user can visually monitor charge and discharge cycles by observing the LED indicators on the OBPP. For more information about these indicators, see Operating Instructions Section.

In addition to automatic mode, the user can also control and monitor the CMS from a PC terminal. A range of commands allows the user to set cell bypass conditions, change parameters, or read cell voltages, current, or temperature. For more about this command set, see the Operating Instructions Section.
II. Getting Started

A. System Components

1. Batteries - The Cell Management System includes 4 Li-Ion batteries. Specifically, the batteries are Lithium Iron Phosphate (LiFePO4) cells.

   Capacity: 3.2V, C = 10Ah
   Size: 82x27x108 mm (L x W x H).

2. One Board Per Pack (OBPP) - The OBPP is the board responsible for managing and monitoring each cell in a pack of four cells. This board must be mounted in the appropriate orientation onto a pack of four cells. See the Installation Section for instructions.

3. Screws and Spacers - Provided with the CMS are:

   Spacers: Four (4) Aluminum, four (4) copper
   Screws: 8 (5/8” 10-32)
   Lock Washers: 8 (10-32)

4. Cables - Provided with the CMS are:

   Power Cable: 1 to connect to another pack in series
   I²C Cable: 1 for connection to I²C bus

5. PC to I2C Converter: The converter provided with the CMS is necessary for the use of the I2C features. This device is used to convert either RS-232 or USB to I2C.

6. RealTerm: This is free software that is useful for sending basic hex/ascii/binary commands via a PCs USB or RS-232 COM ports. This software is used for sending basic I2C commands.
B. Extras You May Need

1. **Constant Current Power Supply**- For cell charging, it is recommended to use a constant current source. The current source should be limited to no more than 2C (20A). It is recommended to use a constant 10A source, which will charge an empty pack in about 1 hour.

3. **Relay**- It may be convenient to use a relay when charging a pack. A common emitter output on the OBPP provides a logic high output when a pack becomes fully charged, and a logic low when the pack is fully discharged. Connecting a power supply or load to the pack through a relay controlled by this output will prevent the pack from being over charged or over discharged. In the event that a relay is not available, the user can view and LED indicator on the OBPP that will signify when charging or discharging is complete.

4. **Safety Glasses**- Please be cautious when working with high voltage electronics. Particularly when connecting several packs in series, or installing the OBPP, it is important to wear safety glasses.

5. **PicKit Serial Programmer**: The PicKit is a device that is used for In Circuit Serial Programming (ICSP) of PIC u-controllers. This device may be needed if the CMS firmware requires recompiling or software updates.

C. Warnings

1. **DO NOT** short the terminals of any cells. This could result in serious injury or cause the permanent damage to the CMS and/or cells.

2. **DO NOT** touch heat sinks while board is in operation. They, or other components on the board may become extremely hot!
III. Installation

A. Installing the OBPP

1. Screw spacers onto the terminals of four cells. Match Copper spacers with Copper terminals, and Aluminum spacers with Aluminum terminals.

2. Place the OBPP on top of a pack of four cells, as shown below, lining the holes with the top of the spacers.

3. Once aligned, secure the OBPP with the provided screws, being careful not to short any of the cells with a screw or screwdriver.

IV. Operating Instructions
A. Charging

1. If it is the first time using the CMS, or after a system crash, reset the CMS by jumping the reset pins on the OBPP, as shown in figure 2.

![Figure 2 - Reset Pin](image)

2. If a relay is unavailable, skip to step 7.

3. If a relay is available, attach the positive terminal of the relay switch to the “Done” pin on the OBPP and the negative terminal of the relay switch to the “GND” pin on the OBPP. Use a 10KOhm Resistor to pull up the “Done” Signal to VDD.

4. Attach the positive terminal of the power supply to the positive terminal of the relay.

5. Attach the negative terminal of the power supply to the negative terminal of the pack.

6. Attach the negative terminal of the relay to the positive terminal of the pack. The above configuration will cause the relay to open when the cell becomes fully charged. This will prevent the cells from being over charged.

7. For a setup without a relay, connect the positive terminal of the power supply to the positive terminal of the pack, and the negative terminal of the power supply to the negative terminal of the pack as show in figure 3.
8. After checking to make sure the relay is closed (on), turn on the power supply so that it is current limited within a range of 5-20A. Once on, the pack will begin to charge. The green LED labeled “CHRG” should blink, as well as the yellow LED labeled “ON.” If this is not the case, perform a system reset, or refer to the maintenance manual for instructions to reprogram the CMS.

*Note: **DO NOT** turn a power supply on before attaching to a pack.
B. Discharging

1. If it is the first time using the CMS, or after a system crash, reset the CMS by jumping the reset pins on the OBPP, as shown in figure 2.

2. If a relay is unavailable, skip to step 7.

3. If a relay is available, attach the positive terminal of the relay switch to the “Done” pin on the OBPP and the negative terminal of the relay switch to the “GND” pin on the OBPP. Use a 10KOhm Resistor to pull up the “Done” Signal to V_{DD}.

4. Attach the positive terminal of the load* to the positive terminal of the pack.

5. Attach the negative terminal of the load to the positive terminal of the relay.

6. Attach the negative terminal of the relay to the negative terminal of the pack. The above configuration will cause the relay to open when the cell becomes fully discharged. This will prevent the cells from being over discharged.

7. For a setup without a relay, connect the positive terminal of the load to the positive terminal of the pack, and the negative terminal of the load to the negative terminal of the pack as show in figure 3.

8. Check to make sure the relay is closed (on), once on, the pack will begin to discharge.

*DO NOT use a load that requires more than 20A. Doing so could cause the fuse to give out, or in certain situations cause permanent damage to the CMS.
C. I2C Operation

1. If it is the first time using the CMS, or after a system crash, reset the CMS by jumping the reset pins on the OBPP, as shown below.

2. Open RealTerm I2C. If RealTerm I2C is not installed on your PC, go to:  
   http://realterm.sourceforge.net/index.html#downloads_Download  
   To download, as well as for RealTerm help.

3. Using either an RS-232 or USB cable, connect the PC to I2C converter to the PC running RealTerm I2C.

4. Connect the provided I2C communication cable between the PC to I2C converter and the OBPP. Be sure to orient the cables as shown below.

5. In RealTerm, under the 'Port' tab:
   a. Select ‘Baud’ of 57600
   b. Select ‘Port’ 3, the Port number for the USB port
   c. Set ‘Parity’ to ‘None’
   d. Set ‘Data Bits’ to ‘8’
   e. Set ‘Stop Bits’ to ‘1 bit’
   f. Set ‘Hardware Flow Control’ to ‘RTS/CTS’
   g. Set ‘Software Flow Control’ to ‘Xon Char: 17’ and ‘Xoff Char: 19’. Do not check ‘Receive’ or ‘Transmit’
   h. Click ‘Change’ Button to save settings

6. In RealTerm, under the ‘I2C’ tab:
   a. Set ‘Bus Num’ to ‘1’
   b. Set ‘Address’ to 0x54, which is the default address of an OBPP, or else set to the known address of the board.
   c. Set ‘SubAddr’ to 0
7. Sending Commands

Read Command Format:

<table>
<thead>
<tr>
<th>Board Address</th>
<th>Command Number</th>
<th>Data Byte High</th>
<th>Data Byte Low</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default 0x54</td>
<td>(0x00-0x1B)</td>
<td>0xF0</td>
<td>0x00</td>
</tr>
</tbody>
</table>

Write Command Format:

<table>
<thead>
<tr>
<th>Board Address</th>
<th>Command Number</th>
<th>Data Byte High</th>
<th>Data Byte Low</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default 0x54</td>
<td>(0x00-0x1B)</td>
<td>0x0X</td>
<td>0xXX</td>
</tr>
</tbody>
</table>

8. Pay careful attention to the boundary conditions for command sets. Failure to send correct commands could result in failure of the system.
E. Troubleshooting

1. Common Failures

<table>
<thead>
<tr>
<th>Component</th>
<th>Error</th>
<th>Possible Solution</th>
</tr>
</thead>
<tbody>
<tr>
<td>Batteries</td>
<td>Cell appears to have capacity of less than 10 A-h</td>
<td>Replace Cell</td>
</tr>
<tr>
<td>Any LED</td>
<td>LED never turns on</td>
<td>Replace Green LED</td>
</tr>
<tr>
<td>Any LED</td>
<td>LED acts in an unexpected manner</td>
<td>Reset OBPP</td>
</tr>
<tr>
<td>I²C</td>
<td>I²C Read Returns 0xFFFF or other unexpected value</td>
<td>Reset OBPP</td>
</tr>
<tr>
<td>I²C</td>
<td>I²C Write does not change parameters as expected</td>
<td>Reset OBPP</td>
</tr>
<tr>
<td>Current Sensor</td>
<td>I²C returns invalid data for Current reading</td>
<td>See Maintenance Manual for Instructions to replace Current Sensor</td>
</tr>
<tr>
<td>Temperature Sensor</td>
<td>I²C returns invalid data for Temperature reading</td>
<td>See Maintenance Manual for Instructions to replace Temperature Sensor</td>
</tr>
<tr>
<td>Fuse</td>
<td>CMS unable to be charged/ discharged - I²C not responding as expected</td>
<td>Replace Fuse Reset OBPP</td>
</tr>
</tbody>
</table>

2. Replacing a Fuse

If you find the pack is unable to take a charge or a discharge, or if certain components are not responding properly, the problem may be a blown fuse. A blown fuse will look like the one shown in the figure __. If your fuse looks similar to this, follow the instructions below to replace the fuse.

   a. Remove the broken fuse.
   b. Using the part described below, install the new Fuse. There is no directionality to the part, so orientation does not matter.

   | 25 Amp Automotive Blade Fuse | F1018-ND  | Blade Fuse | Digikey |

3. Resetting Board

If you find any issues with I²C communication, or if the CMS appears to be stuck in a state, a possible solution is to reset the firmware. This can be done by simply putting a jumper across the reset pins, as shown in figure 2.
VII. Appendix

A. Board Components

a. The meaning of the LED's on the OBPP

The LED's on the OBPP are visual signals which inform the users and the maintainers about the current status of the CMS. The following figure depicts what LED's mean:

- Solid – Charged
- Blink – Charging
- Solid – Discharged
- Blink – Discharging
- Solid – Slave
- Blink – Stand-alone
- Solid – Bypassing

b. Pinouts of all connectors
There are two (2) connectors on the OBPP, which are:
- Anderson connector
- Data connector
### Pinout for Anderson Connector (3-pin connector)

<table>
<thead>
<tr>
<th>Pin #</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Power (+12V)</td>
</tr>
<tr>
<td>2</td>
<td>No Connection</td>
</tr>
<tr>
<td>3</td>
<td>Ground</td>
</tr>
</tbody>
</table>

![Anderson Connector Diagram](image)

### Pinout for Data Connector (2x5 connector)

<table>
<thead>
<tr>
<th>Pin #</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Power (+5V)</td>
</tr>
<tr>
<td>2</td>
<td>External SDA (data for I²C)</td>
</tr>
<tr>
<td>3</td>
<td>External SCL (clock for I²C)</td>
</tr>
<tr>
<td>4</td>
<td>No Connection</td>
</tr>
<tr>
<td>5</td>
<td>No Connection</td>
</tr>
<tr>
<td>6</td>
<td>External RTSS (Redundant Temperature Sensing System)</td>
</tr>
<tr>
<td>7</td>
<td>External Done</td>
</tr>
<tr>
<td>8</td>
<td>Ground</td>
</tr>
<tr>
<td>9</td>
<td>No Connection</td>
</tr>
<tr>
<td>10</td>
<td>No Connection</td>
</tr>
</tbody>
</table>

![Data Connector Diagram](image)
MATLAB AND SIMULINK SIMULATION FILES

GRAPHS.M

% Before running this .m file, run balancing_sim.mdl in Simulink to obtain
% the appropriate simulation results.

% This file analyzes the data from the simulation of the balancing of
% four cells within a LPRDS-CMS-2011 pack.

% The following analysis contains data for SOC, Voltage, Average SOC,
% standard deviation and power dissipation.

% Compute the average SOC of the four cells
avg = (SOCcbalanced.signals.values(1:80001,1)+SOCcbalanced.signals.values(1:80001,2)+SOCcbalanced.signals.values(1:80001,3)+SOCcbalanced.signals.values(1:80001,4))/4;

% Compute the standard deviation of the four cells
stand = sqrt(((SOCcbalanced.signals.values(1:80001,1)-avg).^2+(SOCcbalanced.signals.values(1:80001,2)-avg).^2+(SOCcbalanced.signals.values(1:80001,3)-avg).^2+(SOCcbalanced.signals.values(1:80001,4)-avg).^2)/4);
stand2 = stand./100; % convert to SOC
stand3 = stand2.*12.8; % convert to Watts
stand4 = stand3.*80000; % convert to Joules

% Compute the overall power dissipation of the bypass resistors
% for the full simulation time.
power_dissipated = PowerDissTotal.signals.values(1:80001,1);
pdtotal(1) = power_dissipated(1);
for i=2:1:80001
    pdtotal(i) = pdtotal(i-1) + power_dissipated(i);
end

% Plot the Average SOC of the four cells
figure(1);
a = plot(avg);
set(a,'LineWidth',2);
set(gca,'FontSize',14);
axis([0 80000 0 100]);
set(gca,'XTickLabel',[0,2.8,5.6,8.3,11.1,13.9,16.7,19.4,22.2]);
title('Average SOC of Cells','FontSize',18);
xlabel('Time (hours)');
ylabel('SOC (%)');

% Plot the SOC of the four cells on the same graph
figure(2);
s = plot(SOCcbalanced.signals.values(1:80001,1:4));
set(s,'LineWidth',2);
set(gca,'FontSize',14);
axis([0 80000 0 100]);
set(gca,'XTickLabel',[0,2.8,5.6,8.3,11.1,13.9,16.7,19.4,22.2]);
title('SOC Curves of Charge/Discharge of Pack with Balancing','FontSize',18);
xlabel('Time (hours)');
ylabel('SOC (%)');

% Plot the Voltage of the four cells on the same graph
figure(3);
v = plot(balancedCharge_3.signals.values(1:80001,1:4));
set(v,'LineWidth',2);
set(gca,'FontSize',14);
axis([0 80000 2.7 3.9]);
title('Voltage Curves of Charge/Discharge of Pack with Blancing','FontSize',18);
xlabel('Time (hours)');
ylabel('Voltage (V)');

% Plot the total power dissipation of the four bypass resistors
figure(4)
p = plot(PowerDissTotal.signals.values(1:80001,1));
set(p,'LineWidth',2);
set(gca,'FontSize',14);
axis([0 80000 0 6]);
set(gca,'XTickLabel',[0,2.8,5.6,8.3,11.1,13.9,16.7,19.4,22.2]);
title('Power Dissipation of Bypass Resistors','FontSize',18);
xlabel('Time (hours)');
ylabel('Power (Watts)');

% Plot the standard deviation of the four cells
figure(5);
st = plot(stand);
set(st,'LineWidth',2,'Color','magenta');
set(gca,'FontSize',14);
axis([0 80000 0 35]);
set(gca,'XTickLabel',[0,2.8,5.6,8.3,11.1,13.9,16.7,19.4,22.2]);
title('Standard Deviation (%SOC) of Cells During Charge/Discharge','FontSize',18);
xlabel('Time (hours)');
ylabel('Standard Deviation in SOC(%)');

% Plot the standard deviation of the current of the pack
figure(6);
st2 = plot(stand2);
set(st2,'LineWidth',2,'Color','blue');
set(gca,'FontSize',14);
axis([0 80000 0 3.5]);
set(gca,'XTickLabel',[0,2.8,5.6,8.3,11.1,13.9,16.7,19.4,22.2]);
title('Standard Deviation (A) of Cells During Charge/Discharge','FontSize',18);
xlabel('Time (hours)');
ylabel('Standard Deviation in Amps');

% Plot the standard deviation of the power of the pack
figure(7);
st3 = plot(stand3);
set(st3,'LineWidth',2,'Color','red');
set(gca,'FontSize',14);
axis([0 80000 0 3.5]);
set(gca,'XTickLabel',[0,2.8,5.6,8.3,11.1,13.9,16.7,19.4,22.2]);
title('Standard Deviation (W) of Cells During Charge/Discharge','FontSize',18);
xlabel('Time (hours)');
ylabel('Standard Deviation in Watts');

% Plot the standard deviation of the energy of the pack
figure(8);
st4 = plot(stand4);
set(st4,'LineWidth',2,'Color','green');
set(gca,'FontSize',14);
axis([0 80000 0 45]);
set(gca,'XTickLabel',[0,2.8,5.6,8.3,11.1,13.9,16.7,19.4,22.2]);
title('Standard Deviation (J) of Cells During Charge/Discharge','FontSize',18);
xlabel('Time (hours)');
ylabel('Standard Deviation in Watts');
xlabel('Time (hours)');
ylabel('Standard Deviation in Joules');

% Plot the work done by the bypass resistors
figure(10);
pd = plot(pdtotal);
set(gca,'LineWidth',2,'Color','red');
set(gca,'FontSize',14);
axis([0 80000 0 140000]);
set(gca,'XTickLabel',[0,2.8,5.6,8.3,11.1,13.9,16.7,19.4,22.2],'
XAxisLocation','top','YAxisLocation','right');
grid on;
title('Work Done (J) by Bypass During Charge/Discharge','FontSize',18);
xlabel('Time (hours)');
ylabel('Work Done by Bypass (Joules)');

eff = (stand5(1)-stand5(80001))/(pdtotal(80001)-pdtotal(1))

---------------------------------------------------------------------------

BALANCING_ALG.M
---------------------------------------------------------------------------

function [y,o1,o2,o3,o4] = fcn(V1,V2,V3,V4, time)

%Simulate the cell Balancing Algorithm for the CMS

balance = false;

t1 = 0;
t2 = 0;
t3 = 0;
t4 = 0;

runTime = 3300;
iSOC1 = 10;
iSOC2 = 10;
iSOC3 = 20;
iSOC4 = 15;
o1 = 0;
o2 = 0;
o3 = 5;
o4 = 0;
const1 = 9;
const2 = const1;
const3 = const1;
const4 = const1;
dif12 = V1-V2;
dif13 = V1-V3;
dif14 = V1-V4;
dif21 = V2-V1;
dif23 = V2-V3;
dif24 = V2-V4;
dif31 = V3-V1;
dif32 = V3-V2;
dif34 = V3-V4;
dif41 = V4-V1;
dif42 = V4-V2;
dif43 = V4-V3;

if(balance)
  if (((dif12 > .04) || (dif13 > .04) || (dif14 > .04)) && V1<3.5)
    %pw1 = 10;
\begin{verbatim}
% o1 = 1;
t1 = time;
end
if(((dif21 > .04) || (dif23 > .04) || (dif24 >.04)) && V1<3.5)
  %pw2 = 10;
  % o2 = 1;
t2 = time;
end
if(((dif31 > .04) || (dif32 > .04) || (dif34 >.04)) && V1<3.5)
  %pw3 = 10;
  % o3 = 1;
t3 = time;
end
if(((dif41 > .04) || (dif42 > .04) || (dif43 >.04)) && V4<3.5)
  %pw4 = 10;
  % o4 = 1;
t4 = time;
end
if((time-t1 >20)
  %pw1 = 90;
  % o1 = 0;
end
if((time-t2 >20)
  % pw2 = 90;
  % o2 = 0;
end
if((time-t3 >20)
  %pw3 = 90;
  % o3 = 0;
end
if((time-t4 >20)
  %pw4 = 90;
  % o4 = 0;
end
end
if(V1>=3.8 || V2>=3.8 || V3>=3.8 || V4>=3.8)
y = 1;
else
  y = 0;
end
\end{verbatim}
BALANCING_SIM.MDL
% Matlab script to process data gathered by testSetup.mdl
% when running tests on a LPRDS-CMS-2011 CMS Pack

cell1 = CMSTest1.signals.values(1:21728,1); % replace number of elements in structure specific to test
cell2 = CMSTest1.signals.values(1:21728,2); % replace number of elements in structure specific to test
cell3 = CMSTest1.signals.values(1:21728,3); % replace number of elements in structure specific to test
cell4 = CMSTest1.signals.values(1:21728,4); % replace number of elements in structure specific to test
time = CMSTest1.time;

figure(1)
plot(time, cell1, time, cell2, time, cell3, time, cell4);
set(gca,'FontSize',14);
axis([0 80000 0 45]);
% add more ticks specific to length of test
set(gca,'XTick',[0,3600,7200,10800,14400,18000,21600,25200,28800,32400]);
% add more tick labels specific to length of test
set(gca,'XTickLabel',[0,1,2,3,4,5,6,7,8,9]);
HYPERTERMINAL SCRIPT FILE

TESTSCRIPT.TXT

S5400F000R02P
S5401F000R02P
S5402F000R02P
S5403F000R02P
S5404F000R02P
S5405F000R02P
S5406F000R02P
S5407F000R02P
S5408F000R02P
S5409F000R02P
S540BF000R02P
S540CF000R02P
S540DF000R02P
S540EF000R02P
S5411F000R02P
S540AF000R02P
S5419F000R02P
/*
 * LPRDS CMS Spring 2011
 * ECE 492
 * OBPP Firmware Version 1.3
 * Author: Justin Bunnell
 * Last Updated: 5/5/11
 */

#include <p18f2423.h>
#include <delays.h>
#include <timers.h>
#include <adc.h>
#include <sw_i2c.h>

#pragma config LVP = OFF      // Disables Low Voltage Programming

//Function Declarations
void getCurr(unsigned char x, unsigned char y);
void getVolts1(unsigned char x, unsigned char y);
void getVolts2(unsigned char x, unsigned char y);
void getVolts3(unsigned char x, unsigned char y);
void getVolts4(unsigned char x, unsigned char y);
void getTemp1(unsigned char x, unsigned char y);
void getTemp2(unsigned char x, unsigned char y);
void getTemp3(unsigned char x, unsigned char y);
void getTemp4(unsigned char x, unsigned char y);
void getSOC(unsigned char x, unsigned char y);
void getSeconds(unsigned char x, unsigned char y);
void setBypass1(unsigned char x, unsigned char y);
void setBypass2(unsigned char x, unsigned char y);
void setBypass3(unsigned char x, unsigned char y);
void setBypass4(unsigned char x, unsigned char y);
void setDone(unsigned char x, unsigned char y);
void i2cAddress(unsigned char x, unsigned char y);
void getState(unsigned char x, unsigned char y);
void getMode(unsigned char x, unsigned char y);
void upperVoltThresh(unsigned char x, unsigned char y);
void lowerVoltThresh(unsigned char x, unsigned char y);
void tempThresh(unsigned char x, unsigned char y);
void bypassDuration(unsigned char x, unsigned char y);
void bypassThresh(unsigned char x, unsigned char y);
void getTotalMinutes(unsigned char x, unsigned char y);
void getCycleMinutes(unsigned char x, unsigned char y);
void getVersion(unsigned char x, unsigned char y);
voidgetID(unsigned char x, unsigned char y);

//Array of Pointers to Functions
void (*p[]) (unsigned char x, unsigned char y) =
{
    getCurr,
    getVolts1,
    getVolts2,
    getVolts3,
    getVolts4,
    getTemp1,
    getTemp2,
    getTemp3,
    getTemp4,
    getSOC,
    getSeconds,
    setBypass1,
    setBypass2,
    setBypass3,
    setBypass4,
setDone,
getI2cAddress,
getState,
getMode,
upperVoltThresh,
lowerVoltThresh,
tempThresh,
bypassDuration,
bypassThresh,
bypassTime,
getTotalMinutes,
getCycleMinutes,
getVersion,
getID
};

#define BLINKY
#define USE_INTERRUPT
#define INIT_I2C
#define INIT_ADC
#define USE_TIMER
#define INIT_TIMER
#define SET_MEMORY
#define W_CS PORTA.2

// Memory Allocation
#define MEM_SIZE 28
signed int memory[MEM_SIZE];
unsigned char mem_index;

// ADC Definitions
#define NUM_ADC_CHANNELS 9
signed int adc_result[NUM_ADC_CHANNELS];
unsigned char adc_channel;
unsigned char adc_channel_select[] =
{
  ADC_CH0,
  ADC_CH1,
  ADC_CH2,
  ADC_CH3,
  ADC_CH4,
  ADC_CH5,
  ADC_CH10,
  ADC_CH11,
  ADC_CH12
};

// Charging Algorithm Definitions
signed int dif12, dif13, dif14;
signed int dif21, dif23, dif24;
signed int dif31, dif32, dif34;
signed int dif41, dif42, dif43;

// Bypass Time Definition
signed int b_1_time;
signed int b_2_time;
signed int b_3_time;
signed int b_4_time;

// Bypass Indicator Definition
unsigned char b1_true, b2_true, b3_true, b4_true;

// Bypass Enable Port Definition
#define B_1_EN LATAbits.LATA7
#define B_2_EN LATCbits.LATC2
#define B_3_EN LATBbits.LATB2
#define B_4_EN LATBbits.LATB5

// Done Port Definition
#define DONE LATCbits.LATC5
//Charge State Definition
unsigned char charge_state;

//Bypass Time Definition (I2C)
int bypassTime1, bypassTime2, bypassTime3, bypassTime4; //for bypass time set with i2c

//I2C transmission and receiving Buffer Allocation
unsigned char txByte[2];
unsigned char rxByte[4];
volatile int i = 0;
volatile int it = 0;
volatile unsigned char i2c_complete = 0;  //I2C command complete indicator
volatile unsigned char i2c_test;

//Store Timer
#ifdef USETIMER
union Timers tsamp;
#define MacroWriteTimer0(t)  
  tsamp.lt=t;    
  TMR0H=tsamp.bt[1];  
  TMR0L=tsamp.bt[0];
#endif

#define T_TICK (-29500/NUM_ADC_CHANNELS + 40) //timer resets every 1 second
unsigned char ticks; // real time clock, updated by ISR after all ADCs are updated. Set by T_TICK
signed int current_time; // updated by foreground loop

#define LED1 LATAbits.LATA6 //Yellow (on)
#define LED2 LATCbits.LATC0 //Green (Charging)
#define LED3 LATCbits.LATC1 // Red (Discharging)

//Setup ADC Interrupt
#ifdef USEINTERRUPT
#define MacroSetChanADC(channel) (ADCON0 = ((channel >>1) & 0b00111100) | (ADCON0 & 0b11000011))
#define MacroConvertADC() (ADCON0bits.GO = 1)
//Interrupt Subroutine
#pragma interrupt isr  // save=PROD
void isr(void){
  if(PIR1bits.SSPIF){ //If I2C interrupt flag is set
    if(SSPSTATbits.R == 1){ //If a read command
      i = 0;
      rxByte[3] = SSPBUF;
      if(rxByte[1] <28){
        txByte[0] = ((char)((0xFF00 & memory[rxByte[1]])>>8));  //set transmission buffer
        txByte[1] = ((char)((0x00FF & memory[rxByte[1]])));
    }
    if(SSPCON1bits.CKP == 0){ //if i2c clock is being held low
      SSPBUF = txByte[it]; //write to i2c buffer
      SSPCON1bits.CKP = 1; //release i2c clock line
      it++;
    }else{
      it = 0;
    }
  }else{ //a write command
    rxByte[i] = SSPBUF; //read i2c buffer
    i++;
  }
  //include this code to recover when incorrect command lengths are sent
}
//if not included, i2c may crash if incorrect command structure is used
//  if((SSPSTATbits.D == 1))
//     i++;
//  else if(SSPSTATbits.D == 0 && i==0)
//     i++;
//  else {
//      i -= 1;
//      //i2c_complete = 1;
//  }
if((i)==4){
    i = 0;
    i2c_complete = 1;  //signifies that full i2c command was received
}

PIR1bits.SSPIF = 0;  //clear i2c interrupt flag
}

if(INTCONbits.TMR0IF){  // Timer 0 interrupt
    ADCON1 = 0b00000000;  //turn on all a/d converters
    MacroWriteTimer0( T_TICK );  // reset set timer period
    adc_result[adc_channel] = (((unsigned int)ADRESH)<<8) | ((unsigned int)ADRESL); // get
    ADC result as 2's complement
    if(adc_channel == 0) ticks++;  // New tick begins immediately after channel 0 update
    if(adc_channel >= (NUM_ADC_CHANNELS -1))
        adc_channel = (adc_channel+1);
    else
        adc_channel = 0;
    MacroSetChanADC(adc_channel_select[adc_channel]);
    MacroConvertADC();  // Start NEXT conversion
    if(ticks == 60){
        ticks = 0;
        memory[24] = memory[24] + 0x0001;   //Total Run Time (Min)
    }
    memory[10] = (signed int)ticks;  //Seconds Counter
    INTCONbits.TMR0IF = 0;  // Clear TMR0 Interrupt flag
    ADCON1 = 0b00000000;  //turn off a/d converters for use as i/o ports
}

#pragma code highvector=0x08
void highvector(void)
{
    _asm goto isr _endasm
}
#pragma code
#endif // USEINTERRUPT

//Blink the status LEDs
#ifdef BLINKY
void ledblink1(unsigned char ontime){//On LED
    LED1=1;
    Delay10KTCYx(ontime);
    LED1=0;
    Delay10KTCYx(ontime);
}

void ledblink2(unsigned char ontime){//Charging LED
    LED2=1;
    Delay10KTCYx(ontime);
    LED2=0;
}
void ledblink3(unsigned char ontime);//Discharging LED
    LED3=1;
    Delay10KTCYx(ontime);
    LED3=0;
    Delay10KTCYx(ontime);
}

/*
 * Runs the main program loop. Also, initializes memory, and thus default settings.
 * Main will begin automatically on Reset or Reprogramming
 */

void main(void)
{
    Delay10KTCYx( 1 );       // Delay to allow Vpp to be applied for programming (Must be included)

    OSCCON = OSCCON | 0x40;  // Switch to fast internal 1 MHz RC Clock (TOSC=125 ns)

    // Set up Port A
    LATA = 0x00;  // Pre-clear
    TRISA = 0b00101111;

    // Set up port B
    LATB = 0x00;  // Pre-clear
    TRISB = 0b00011011;

    //Set up port C
    LATC = 0x00;
    TRISC = 0x18;

    //Set up I2C control registers

    #ifdef INITI2C
        PIR1bits.SSPIF = 0; //clear the 12c interrupt flag
        SSPSTAT = (0b100000000) | (SSPSTAT & 0b00111111);  //disable i2c interrupt
        PIE1bits.SSPIE = 1;  //enable i2c interrupt
        SSPCON1 = 0b00110110;
        SSPCON2 = 0b10000000;
        SSPADD = 0b01010100;  //Only the lower 7 bits are used for the address bit 7 is ignored
    #endif

    //Initialize ADC

    #ifdef INITADC
    //Set up A/D Conversion Manually
    //ADCON0 = 0b00100101;
    //ADCON1 = 0b00000101;
    //ADCON2 = 0b10001000;

    OpenADC(ADC_FOSC_2 & ADC_RIGHT_JUST & ADC_2_TAD,
            ADC_CH0 & ADC_INT_OFF & ADC_VREFPLUS_VDD & ADC_VREFMINUS_VSS,
            0b00000000);
    
    adc_channel = 0;
    SetChanADC(adc_channel_select[adc_channel]);
    ConvertADC();  // Start FIRST conversion
    #endif

    // Set up sampling rate timer

    #ifdef INITTIMER
    OpenTimer0(0);
    TIMER_INT_ON &
    T0_16BIT &
    
    

T0_SOURCE_INT & // Internal 8 MHz clock (divided by 8)
T0_PS_1_8 & // Timer ticks 8us
}
	WriteTimer0( T_TICK ); // Set AD Sampling Rate
#endif

// Set up interrupts
#ifdef USEINTERRUPT
PIR1bits.ADIF = 0;   // Clear ADC interrupt flag
INTCONbits.TMR0IF = 0; // Clear TMR0 interrupt flag
RCONbits.IPEN = 1;   // Set Priorities
INTCONbits.GIEL = 0; // disable low priority interrupts
INTCONbits.GIEH = 1; // enable high priority interrupts
IPR1bits.SSPIP = 1;  // MSSP high priority interrupts
IPR1bits.ADIP = 1;   // A/D Converter is high priority
INTCON2bits.TMR0IP = 1; // TMR0 is high priority
#endif  // USEINTERRUPT

//Initialize A/D converters to prevent reading incorrect values on startup
adc_result[0] = 0x0800;
adc_result[1] = 0x099A;
adc_result[2] = 0x099A;
adc_result[3] = 0x099A;
adc_result[4] = 0x099A;
adc_result[5] = 0x000F;
adc_result[6] = 0x000F;
adc_result[7] = 0x000F;
adc_result[8] = 0x000F;

//Initialize Memory Allocation
#ifdef SETUPMEMORY
//Pack Current (0x0000 - 0xFFFF)
memory[0] = 0x0800;
//Cell 1 Voltage (0x0000 - 0xFFFF / 0-5V)
memory[1] = 0x099A;
//Cell 2 Voltage (0x0000 - 0xFFFF / 0-5V)
memory[2] = 0x099A;
//Cell 3 Voltage (0x0000 - 0xFFFF / 0-5V)
memory[3] = 0x099A;
//Cell 4 Voltage (0x0000 - 0xFFFF /0-5V)
memory[4] = 0x099A;
//Cell 1 Temperature (0x0000 - 0xFFFF)
memory[5] = 0x000F;
//Cell 2 Temperature (0x0000 - 0xFFFF)
memory[6] = 0x000F;
//Cell 3 Temperature (0x0000 - 0xFFFF)
memory[7] = 0x000F;
//Cell 4 Temperature (0x0000 - 0xFFFF)
memory[8] = 0x000F;
//Current Sum (Integration)
memory[9] = 300;
//Total Run Time from last reset (minutes)
memory[10] = 0x0000; // may not need both...do the calc later...
//Bypass Switch 1 (Upper 7 bits for Duration/ LSB for on or off)
memory[11] = 0x0000; //Set to 0 min, bypass off
//Bypass Switch 2 (Upper 7 bits for Duration/ LSB for on or off)
memory[12] = 0x0000; //Set to 0 min, bypass off
//Bypass Switch 3 (Upper 7 bits for Duration/ LSB for on or off)
memory[13] = 0x0000; //Set to 0 min, bypass off
//Bypass Switch 4 (Upper 7 bits for Duration/ LSB for on or off)
memory[14] = 0x0000; //Set to 0 min, bypass off
//DONE Signal(on or off 0x0000 or 0x00FF)
memory[15] = 0x0000; //Done Signal Low (off)
//I2C Address
memory[16] = 0x0054; //0x0054 by default

//System State ???
memory[17] = 0x0000; //Reset State

//System Mode {automatic or not / 0x0000 or 0x00FF}
memory[18] = 0x0000; //automatic mode by default

//Upper Voltage Threshold (0-5V)
memory[19] = 0x0C29; //3.8V by default

//Lower Voltage Threshold (2-4V)
memory[20] = 0x0086; //2.8V by default

//Upper Temperature Threshold (0 - 255 deg C)
memory[21] = 0x0B0A; //60 deg C by default

//Bypass Duration (for Charging Alg) (0-255 min)
memory[22] = 20; //20 min by default

//Bypass Threshold (Charging Alg) (0-5V 0x0000-0x0FFF)
memory[23] = 41; //50 mV by default

//Total Run Time (0x0000 to 0xFFFF / 0 to 65536 min)
memory[24] = 0x0000; //pos logic by default

//Cycle Time (0x0000 to 0xFFFF / 0 to 65536 min)
memory[25] = 0x0000;

//Version ID (Version.Subversion)
memory[26] = 0x0010; //(1.0)

//Board # (0-255)
memory[27] = 0x0001; //(board # 1)
#endif

//Initial charge state
charge_state = 0;

//Initial Done Signal
DONE = 0;

//Initial Bypass Times
bypassTime1 = 0;
bypassTime2 = 0;
bypassTime3 = 0;
bypassTime4 = 0;

//Initial Bypass States
b1_true = 0;
b2_true = 0;
b3_true = 0;
b4_true = 0;

//Initial LED Status
LED1 = 0;
LED2=0;
LED3 = 0;

/*
   Main While Loop
   This section is responsible for the
   charging algorithm, fuel gauge algorithm,
   and response to i2c commands
*/
while(1){

    //Fuel Gauge Algorithm
    if(current_time != memory[25])
    {

    }
current_time = memory[25];
} else if(memo
else if(memory[0] < 0x07C0){//if pack is discharging (need to fix value)
    if(charge_state == 2){//if pack is discharging
        charge_state = 2;
        memory[25] = 0x0000;
        b_1_time = 0x7fff;
        b_2_time = 0x7fff;
        b_3_time = 0x7fff;
        b_4_time = 0x7fff;
        DONE = 1;
        LED2 = 0;
    }
    if(DONE == 1)
        ledblink3(15);
    else{//pack is neither charging or discharging
        if((charge_state == 1) && (DONE == 0)){
            charge_state = 0;
            memory[25] = 0x0000;
            b_1_time = 0x7fff;
            b_2_time = 0x7fff;
            b_3_time = 0x7fff;
            b_4_time = 0x7fff;
            B_1_EN = 0;
            memory[11] = 0x0000;
            B_2_EN = 0;
            memory[12] = 0x0000;
            B_3_EN = 0;
            memory[13] = 0x0000;
            B_4_EN = 0;
            memory[14] = 0x0000;
            LED2 = 0;
            LED3 = 0;
        }else if((charge_state == 1) && (DONE == 1)){
            charge_state = 0;
            memory[25] = 0x0000;
            b_1_time = 0x7fff;
            b_2_time = 0x7fff;
            b_3_time = 0x7fff;
            b_4_time = 0x7fff;
            B_1_EN = 0;
            memory[11] = 0x0000;
            B_2_EN = 0;
            memory[12] = 0x0000;
            B_3_EN = 0;
            memory[13] = 0x0000;
            B_4_EN = 0;
            memory[14] = 0x0000;
            LED2 = 1;
            LED3 = 0;
        }else if((charge_state == 2) && (DONE == 0)){
            charge_state = 0;
            memory[25] = 0x0000;
            b_1_time = 0x7fff;
            b_2_time = 0x7fff;
            b_3_time = 0x7fff;
            b_4_time = 0x7fff;
            B_1_EN = 0;
            memory[11] = 0x0000;
            B_2_EN = 0;
            memory[12] = 0x0000;
            B_3_EN = 0;
            memory[13] = 0x0000;
            B_4_EN = 0;
            memory[14] = 0x0000;
            LED2 = 0;
            LED3 = 0;
        }else if((charge_state == 2) && (DONE == 1)){
            charge_state = 0;
            memory[25] = 0x0000;
            b_1_time = 0x7fff;
            b_2_time = 0x7fff;
            b_3_time = 0x7fff;
            b_4_time = 0x7fff;
            B_1_EN = 0;
            memory[11] = 0x0000;
            B_2_EN = 0;
            memory[12] = 0x0000;
            B_3_EN = 0;
            memory[13] = 0x0000;
            B_4_EN = 0;
            memory[14] = 0x0000;
            LED2 = 0;
            LED3 = 0;
        }else if(charge_state == 0){
            charge_state = 0;
            memory[25] = 0x0000;
            b_1_time = 0x7fff;
            b_2_time = 0x7fff;
If memory[18] < 0x00FF { //if in automatic mode
    ledBlink1(15);
    if((charge_state == 1) && (DONE == 0)){//if pack is currently charging
        // ledBlink2(15);
        // Check for bypass conditions. If they exist, set bypass on
            B_1_EN = 1;
            memory[11] = 0x0001;
            b_1_time = memory[25];
            b1_true = 1;
        }
            B_2_EN = 1;
            memory[12] = 0x0001;
            b_2_time = memory[25];
            b2_true = 1;
        }
            B_3_EN = 1;
            memory[13] = 0x0001;
            b_3_time = memory[25];
            b3_true = 1;
        }
            B_4_EN = 1;
            memory[14] = 0x0001;
            b_4_time = memory[25];
            b4_true = 1;
        }
        // Check for end of bypass
        if(((memory[25] - b_1_time) >= memory[22]) && (b1_true == 1)){
            B_1_EN = 0;
            memory[11] = 0x0000;
            b_1_time = 0x7fff;
            b1_true = 0;
        }
        if(((memory[25] - b_2_time) >= memory[22]) && (b2_true == 1)){
            B_2_EN = 0;
            memory[12] = 0x0000;
            b_2_time = 0x7fff;
            b2_true = 0;
        }
        if(((memory[25] - b_3_time) >= memory[22]) && (b3_true == 1)){
            B_3_EN = 0;
            }///
memory[13] = 0x0000;
b_3_time = 0x7fff;
b3_true = 0;
}
if((memory[25] - b_4_time) >= memory[22]) && (b4_true == 1)){
    B_4_EN = 0;
    memory[14] = 0x0000;
b_4_time = 0x7fff;
b4_true = 0;
}
//check if any cell is fully charged
    //charge_state = 0;
    DONE = 1;
    LED2 = 1;
    //Bypass not enabled
    memory[11] = 0x0000;
    B_2_EN = 0;
    memory[12] = 0x0000;
    B_3_EN = 0;
    memory[13] = 0x0000;
    B_4_EN = 0;
    memory[14] = 0x0000;
b1_true = 0;
b2_true = 0;
b3_true = 0;
b4_true = 0;
    memory[9] = 594;
}
//check if any cell is overheating
    //charge_state = 0;
    DONE = 1;
    LED2 = 0;
    LED3 = 0;
}
else if(charge_state == 2) && (DONE == 1)){//If the pack is currently discharging
    // ledblink3(15);
    //Bypass not enabled
    memory[11] = 0x0000;
    B_2_EN = 0;
    memory[12] = 0x0000;
    B_3_EN = 0;
    memory[13] = 0x0000;
    B_4_EN = 0;
    memory[14] = 0x0000;

    //check if any cell is fully discharged
    //charge_state = 0;
    DONE = 0;
    LED3 = 1;
    memory[9] = 6;
}
//check if any cell is overheating
    //charge_state = 0;
    DONE = 0;
    LED2 = 0;
    LED3 = 0;
} else { // If the pack neither charging or discharging

    // Bypass not enabled -- take no action
    if (b1_true == 0)
        B_1_EN = 0;
    memory[11] = 0x0000;
    if (b2_true == 0)
        B_2_EN = 0;
    memory[12] = 0x0000;
    if (b3_true == 0)
        B_3_EN = 0;
    memory[13] = 0x0000;
    if (b4_true == 0)
        B_4_EN = 0;
    memory[14] = 0x0000;
}
} else { // if not in automatic mode

    LED1 = 1;

    if ((charge_state == 1) && (DONE == 0)) {
        // Check to see if any cell has become fully charged
            memory[19]) || (memory[4] >= memory[19]))
            DONET = 1;
        LED2 = 1;
        // memory[25] = 0x0000;
        b_1_time = 0x7fff;
        b_2_time = 0x7fff;
        b_3_time = 0x7fff;
        b_4_time = 0x7fff;
        memory[9] = 594;
    }

    // Check if any cell is overheating
            memory[21]) || (memory[8] > memory[21]))) {
        // &&(startingUp == 0))
        // charge_state = 0;
        DONE = 1;
        LED2 = 0;
        LED3 = 0;
    }
}
}

else if ((charge_state == 2) && (DONE == 1)) {

    // If any cell becomes fully discharged
            memory[20]) || (memory[4] <= memory[20]))
        // charge_state = 0;
        DONET = 1;
        LED3 = 1;
        // memory[25] = 0x0000;
        b_1_time = 0x7fff;
        b_2_time = 0x7fff;
        b_3_time = 0x7fff;
        b_4_time = 0x7fff;
        memory[9] = 6;
    }

    // Check if any cell is overheating
            memory[21]) || (memory[8] > memory[21]))) {
        // &&(startingUp == 0))
        // charge_state = 0;
        DONET = 0;
        LED2 = 0;
        LED3 = 0;
    }
}
//If bypass timer has expired, turn off bypass
if(((memory[25] - b_1_time) >= bypassTime1) && (b1_true == 1)){
  B_1_EN = 0;
  memory[11] = 0x0000;
  b_1_time = 0x7fff;
  bypassTime1 = 0;
  b1_true = 0;
  memory[11] = 0x0028;
}
if(((memory[25] - b_2_time) >= bypassTime2) && (b2_true == 1)){
  B_2_EN = 0;
  memory[12] = 0x0000;
  b_2_time = 0x7fff;
  bypassTime2 = 0;
  b2_true = 0;
  memory[12] = 0x0028;
}
if(((memory[25] - b_3_time) >= bypassTime3) && (b3_true == 1)){
  B_3_EN = 0;
  memory[13] = 0x0000;
  b_3_time = 0x7fff;
  bypassTime3 = 0;
  b3_true = 0;
  memory[13] = 0x0028;
}
if(((memory[25] - b_4_time) >= bypassTime4) && (b4_true == 1)){
  B_4_EN = 0;
  memory[14] = 0x0000;
  b_4_time = 0x7fff;
  bypassTime4 = 0;
  b4_true = 0;
  memory[14] = 0x0028;
}

// When I2C command is fully received, call the appropriate command
if(i2c_complete){
  if(rxByte[1] < 28){
    (*p[rxByte[1]])(rxByte[2],rxByte[3]);
    i2c_complete = 0;
  }
}

// Gets the current
void getCurr(unsigned char x, unsigned char y){
  // R/O
}

// Gets Voltage at Cell 1
void getVolts1(unsigned char x, unsigned char y){
  // R/O
}

// Gets Voltage at Cell 2
void getVolts2(unsigned char x, unsigned char y){
  // R/O
}

// Gets Voltage at Cell 3
void getVolts3(unsigned char x, unsigned char y){

// R/O

// gets Voltage at Cell 4
void getVolts4(unsigned char x, unsigned char y){
    // R/O
}

// gets Temperature at Cell 1
void getTemp1(unsigned char x, unsigned char y){
    // R/O
}

// gets Temperature at Cell 2
void getTemp2(unsigned char x, unsigned char y){
    // R/O
}

// gets Temperature at Cell 3
void getTemp3(unsigned char x, unsigned char y){
    // R/O
}

// gets Temperature at Cell 4
void getTemp4(unsigned char x, unsigned char y){
    // R/O
}

// returns sum of current integration
void getSOC(unsigned char x, unsigned char y){
    // R/O
}

// returns the seconds (0-59)
void getSeconds(unsigned char x, unsigned char y){
    // R/O
}

// sets bypass switch 1
void setBypass1(unsigned char x, unsigned char y){
    if((rxByte[2]) & 0x80) == 0x00){
        memory[11] = (rxByte[3] & 0x00FF) | (((unsigned int)rxByte[2])<<8) & 0xFF00);
        B_1_EN = 1;
        b_1_time = memory[25];
        bypassTime1 = (memory[11])>>1;
        b1_true = 1;
    }
}

// sets bypass switch 2
void setBypass2(unsigned char x, unsigned char y){
    if((rxByte[2]) & 0xF0) == 0x00){
        memory[12] = (rxByte[3] & 0x00FF) | (((unsigned int)rxByte[2])<<8) & 0xFF00);
        B_2_EN = 1;
        b_2_time = memory[25];
        bypassTime2 = (memory[12])>>1;
        b2_true = 1;
    }
}

// sets bypass switch 3
void setBypass3(unsigned char x, unsigned char y){
if(((rxByte[2]) & 0xF0) == 0x00) {
    memory[13] = (rxByte[3] & 0x00FF) | (((unsigned int)rxByte[2])<<8) & 0xFF00);
    B_3_EN = 1;
    b_3_time = memory[25];
    bypassTime3 = (memory[13])>>1;
    b3_true = 1;
}

// sets bypass switch 4
void setBypass4(unsigned char x, unsigned char y) {
    if(((rxByte[2]) & 0xF0) == 0x00) {
        memory[14] = (rxByte[3] & 0x00FF) | (((unsigned int)rxByte[2])<<8) & 0xFF00);
        B_4_EN = 1;
        b_4_time = memory[25];
        bypassTime4 = (memory[14])>>1;
        b4_true = 1;
    }
}

// sets DONE signal high or low
void setDone(unsigned char x, unsigned char y) {
    if(((rxByte[2]) & 0xF0) == 0x00)
        memory[15] = (rxByte[3] & 0x00FF) | (((unsigned int)rxByte[2])<<8) & 0xFF00);
}

// gets/sets board address
void i2cAddress(unsigned char x, unsigned char y) {
    if((x & 0xF0) == 0x00)
        memory[16] = (rxByte[3] & 0x00FF) | (((unsigned int)rxByte[2])<<8) & 0xFF00);
}

// gets the current charge state
void getState(unsigned char x, unsigned char y) {
    // R/O
}

// gets or sets System mode (auto or not)
void getMode(unsigned char x, unsigned char y) {
    if(((rxByte[2]) & 0xF0) == 0x00) {
        memory[18] = (rxByte[3] & 0x00FF) | (((unsigned int)rxByte[2])<<8) & 0xFF00);
        b1_true = 0;
        b2_true = 0;
        b3_true = 0;
        b4_true = 0;
        B_1_EN = 0;
        memory[11] = 0x0000;
        B_2_EN = 0;
        memory[12] = 0x0000;
        B_3_EN = 0;
        memory[13] = 0x0000;
        B_4_EN = 0;
        memory[14] = 0x0000;
    }
}

// gets or sets the upper voltage threshold for the charging algorithm
void upperVoltThresh(unsigned char x, unsigned char y) {
    if(((rxByte[2]) & 0xF0) == 0x00) {
        memory[19] = (rxByte[3] & 0x00FF) | (((unsigned int)rxByte[2])<<8) & 0xFF00);
        b1_true = 0;
        b2_true = 0;
        b3_true = 0;
        b4_true = 0;
        B_1_EN = 0;
        memory[11] = 0x0000;
        B_2_EN = 0;
        memory[12] = 0x0000;
        B_3_EN = 0;
        memory[13] = 0x0000;
//gets or sets the lower voltage threshold for the charging algorithm
void lowerVoltThresh(unsigned char x, unsigned char y) {
    if(((rxByte[2]) & 0xF0) == 0x00) {
        memory[20] = (rxByte[3] & 0x00FF) | (((unsigned int)rxByte[2])<<8) & 0xFF00;
        b1_true = 0;
        b2_true = 0;
        b3_true = 0;
        b4_true = 0;
        B_1_EN = 0;
        memory[11] = 0x0000;
        B_2_EN = 0;
        memory[12] = 0x0000;
        B_3_EN = 0;
        memory[13] = 0x0000;
        B_4_EN = 0;
        memory[14] = 0x0000;
        charge_state = 0;
    }
}

//gets or sets the temperature threshold for the charging algorithm
void tempThresh(unsigned char x, unsigned char y) {
    if(((rxByte[2]) & 0xF0) == 0x00) {
        memory[21] = (rxByte[3] & 0x00FF) | (((unsigned int)rxByte[2])<<8) & 0xFF00;
    }
}

//gets or sets the bypass duration for the charging algorithm
void bypassDuration(unsigned char x, unsigned char y) {
    if(((rxByte[2]) & 0xF0) == 0x00) {
        memory[22] = (rxByte[3] & 0x00FF) | (((unsigned int)rxByte[2])<<8) & 0xFF00;
    }
}

//gets or sets the voltage thresholds for bypass conditions
void bypassThresh(unsigned char x, unsigned char y) {
    if(((rxByte[2]) & 0xF0) == 0x00) {
        memory[23] = (rxByte[3] & 0x00FF) | (((unsigned int)rxByte[2])<<8) & 0xFF00;
    }
}

//gets the total minutes count since last reset
void getTotalMinutes(unsigned char x, unsigned char y) {
    //R/O
}

//gets the minutes count since last change in charge state
void getCycleMinutes(unsigned char x, unsigned char y) {
    //R/O
}

//gets the Version Number
void getVersion(unsigned char x, unsigned char y) {
    //R/O
}

//gets the board ID
void getID(unsigned char x, unsigned char y) {
    //R/O
}
1.) To Open the OBPP Firmware for modification, go to:

   http://sites.lafayette.edu/ece492-sp11/resources/obpp-firmware-2/

   Download "OBPP Firmware"

2.) If MPLAB is not installed on your PC, got to resources/software on the website and download MPLAB

3.) Open "OBPPFirmware.mcp" using MPLAB

4.) Connect the USB connector from the Microchip PICkit 2 to the PC.

5.) Make sure that the arrow from the PICkit 2 matches up with the arrow on the OBPP. Once the PICkit 2 is correctly connected to the PC.

6.) In MPLAB, Go to Configure->Select Device and select Device (PIC18F2423) and Device Family (ALL).

7.) Go to Configure->Configuration Bits. On the popup window check the Configuration bits in code option.

8.) Go to Programmer->Select Programmer-> 7 PICkit 2.

9.) Click on the Build All option.

10.) Click on the Program the target device option. Click on the Bring target MCLR to Vdd option. If program was successful, you are ready to begin modifying the code.

11.) Pay close attention to the comments in the code if you are making any changes. Also, make sure to save a new copy before making any changes!
<table>
<thead>
<tr>
<th>Qty</th>
<th>Component</th>
<th>Vendor Part</th>
<th>Description</th>
<th>Vendor</th>
<th>Price (total)</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>Power Darlington BJT</td>
<td>512-TIP102</td>
<td>TO-220 Power BJT</td>
<td>Mouser</td>
<td>$2.76</td>
</tr>
<tr>
<td>4</td>
<td>Power Resistor, 1.5 Ohm</td>
<td>652-PWR220T-20-1R50F</td>
<td>TO-220 1.5 Ohm Power Resistor</td>
<td>Mouser</td>
<td>$14.28</td>
</tr>
<tr>
<td>4</td>
<td>Red LED, bypass indicator</td>
<td>516-1791-1-ND</td>
<td>Through-Hole LED, Red</td>
<td>Digikey</td>
<td>$1.85</td>
</tr>
<tr>
<td>4</td>
<td>0805 resistor, 75 Ohm</td>
<td>541-75.0CCT-ND</td>
<td>1/8W 0805 75 Resistor, Cut Tape</td>
<td>Digikey</td>
<td>$0.19</td>
</tr>
<tr>
<td>4</td>
<td>Optocoupler, bypass enable signal</td>
<td>425-2077-1-ND</td>
<td>PC123 photocoupler</td>
<td>Digikey</td>
<td>$1.60</td>
</tr>
<tr>
<td>4</td>
<td>0805 resistor, 225 Ohm (510's in parallel)</td>
<td>541-510CCT-ND</td>
<td>1/8W 0805 510 Resistor, Cut Tape</td>
<td>Digikey</td>
<td>$0.19</td>
</tr>
</tbody>
</table>

### SCHEMATIC: CELL SENSOR

<table>
<thead>
<tr>
<th>Qty</th>
<th>Component</th>
<th>Vendor Part</th>
<th>Description</th>
<th>Vendor</th>
<th>Price (total)</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>2 circuit Op Amp</td>
<td>296-26730-1-ND</td>
<td>Dual Rail-to-Rail Op Amp</td>
<td>Digikey</td>
<td>$7.72</td>
</tr>
<tr>
<td>20</td>
<td>0805 resistor, 100 Kohm</td>
<td>541-100KCCT-ND</td>
<td>1/8W 0805 100k Resistor, Cut Tape</td>
<td>Digikey</td>
<td>$0.96</td>
</tr>
<tr>
<td>4</td>
<td>0805 resistor, 51 Kohm</td>
<td>541-51.0KCCT-ND</td>
<td>1/8W 0805 51k Resistor, Cut Tape</td>
<td>Digikey</td>
<td>$0.19</td>
</tr>
<tr>
<td>4</td>
<td>0805 capacitor, 0.1 uF</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>4</td>
<td>0805 capacitor, 10 uF</td>
<td>587-1300-1-ND</td>
<td>10 uF 0805 Capacitor, Cut Tape</td>
<td>Digikey</td>
<td>$0.14</td>
</tr>
<tr>
<td>4</td>
<td>Temperature Sensor</td>
<td>AD22105ARZ-ND</td>
<td>RTSS Temp Sensor</td>
<td>Digikey</td>
<td>$0.25</td>
</tr>
</tbody>
</table>

### SCHEMATIC: CELL STACK

<table>
<thead>
<tr>
<th>Qty</th>
<th>Component</th>
<th>Vendor Part</th>
<th>Description</th>
<th>Vendor</th>
<th>Price (total)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Current Sensor</td>
<td>ACS714LLCTR-30A-T</td>
<td>Current Sensor</td>
<td>Digikey</td>
<td>$4.73</td>
</tr>
<tr>
<td>1</td>
<td>2 circuit Op Amp</td>
<td>296-26730-1-ND</td>
<td>Dual Rail-to-Rail Op Amp</td>
<td>Digikey</td>
<td>$1.93</td>
</tr>
<tr>
<td>1</td>
<td>Voltage Regulator, 5V</td>
<td>LM2936DT-5.0-ND</td>
<td>Voltage Regulator</td>
<td>Digikey</td>
<td>$2.39</td>
</tr>
<tr>
<td>3</td>
<td>0805 capacitor, 0.1 uF</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>1</td>
<td>0805 capacitor, 1 nF</td>
<td>587-1268-1-ND</td>
<td>1 nF 0805 Capacitor, Cut Tape</td>
<td>Digikey</td>
<td>$0.12</td>
</tr>
<tr>
<td>1</td>
<td>electrolytic capacitor, 33 uF</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>
### SCHEMATIC: MICROCONTROLLER

<table>
<thead>
<tr>
<th></th>
<th>Description</th>
<th>Part Number</th>
<th>Component Details</th>
<th>Supplier</th>
<th>Price</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Microcontroller</td>
<td>PIC18F2423-I/SO-ND</td>
<td>PIC 18F2423 SOIC Microcontr.</td>
<td>Digikey</td>
<td>$5.52</td>
</tr>
<tr>
<td>2</td>
<td>Temperature Sensor, Switch</td>
<td>AD22105ARZ-ND</td>
<td>RTSS Temp Sensor</td>
<td>Digikey</td>
<td>$7.44</td>
</tr>
<tr>
<td>3</td>
<td>LED, status - 1 Red, 1 Yellow, 1 Grn</td>
<td>516-1309-ND</td>
<td>Through-Hole LED, Yellow</td>
<td>Digikey</td>
<td>$0.42</td>
</tr>
<tr>
<td></td>
<td></td>
<td>516-1792-1-ND</td>
<td>Through-Hole LED, Green</td>
<td>Digikey</td>
<td>$0.46</td>
</tr>
<tr>
<td></td>
<td></td>
<td>516-1791-1-ND</td>
<td>Through-Hole LED, Red</td>
<td>Digikey</td>
<td>$0.46</td>
</tr>
<tr>
<td>3</td>
<td>0805 resistor, 225 Ohm (510's in parallel)</td>
<td>541-510CCT-ND</td>
<td>1/8W 0805 510 Resistor, Cut Tape</td>
<td>Digikey</td>
<td>$0.14</td>
</tr>
<tr>
<td>1</td>
<td>0805 resistor, 470 Ohm</td>
<td>541-470CCT-ND</td>
<td>1/8W 0805 470 Resistor, Cut Tape</td>
<td>Digikey</td>
<td>$0.05</td>
</tr>
<tr>
<td>1</td>
<td>0805 resistor, 10 Kohm</td>
<td>541-10KCCT-ND</td>
<td>1/8W 0805 10k Resistor, Cut Tape</td>
<td>Digikey</td>
<td>$0.05</td>
</tr>
<tr>
<td>2</td>
<td>0805 resistor, 22.1 Kohm</td>
<td>541-22.1KCCT-ND</td>
<td>1/8W 0805 22.1k Resistor, Cut Tape</td>
<td>Digikey</td>
<td>$0.10</td>
</tr>
<tr>
<td>2</td>
<td>0805 capacitor, 0.1 uF</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>1</td>
<td>6x1 header (ICSP)</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>1</td>
<td>2x1 header (reset)</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>

### SCHEMATIC: ISOLATOR

<table>
<thead>
<tr>
<th></th>
<th>Description</th>
<th>Part Number</th>
<th>Component Details</th>
<th>Supplier</th>
<th>Price</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>I2C Isolator</td>
<td>DUM2250ARWZ-ND</td>
<td>I2C isolator</td>
<td>Digikey</td>
<td>$6.00</td>
</tr>
<tr>
<td>1</td>
<td>Two channel Isolator</td>
<td>390-1086-5-ND</td>
<td>Dual Opto-Isolator</td>
<td>Digikey</td>
<td>$3.80</td>
</tr>
<tr>
<td>3</td>
<td>0805 resistor, 10 Kohm</td>
<td>541-10KCCT-ND</td>
<td>1/8W 0805 10k Resistor, Cut Tape</td>
<td>Digikey</td>
<td>$0.14</td>
</tr>
<tr>
<td>1</td>
<td>0805 resistor, 510 Ohm</td>
<td>541-510CCT-ND</td>
<td>1/8W 0805 510 Resistor, Cut Tape</td>
<td>Digikey</td>
<td>$0.05</td>
</tr>
<tr>
<td>3</td>
<td>0805 capacitor, 0.1 uF</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>

### SCHEMATIC: OBPP

<table>
<thead>
<tr>
<th></th>
<th>Description</th>
<th>Part Number</th>
<th>Component Details</th>
<th>Supplier</th>
<th>Price</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Fuse Holder</td>
<td>3522-2K-ND</td>
<td>Blade Fuse Clip</td>
<td>Digikey</td>
<td>$1.02</td>
</tr>
<tr>
<td>1</td>
<td>25 Amp Automotive Blade Fuse</td>
<td>F1018-ND</td>
<td>Blade Fuse</td>
<td>Digikey</td>
<td>$0.77</td>
</tr>
<tr>
<td>1</td>
<td>5x2 header (data connection)</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>1</td>
<td>Anderson Connector</td>
<td>PM103MBALST</td>
<td>1x3 PIN Straight Male Connector</td>
<td>CW Dist.</td>
<td>$3.08</td>
</tr>
<tr>
<td>101</td>
<td>OBPP components</td>
<td>-</td>
<td>-</td>
<td>Advanced Circuits</td>
<td>$66.00</td>
</tr>
<tr>
<td>1</td>
<td>OBPP Printed Circuit Board</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>HEATSINK HARDWARE</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-------------------</td>
<td>---</td>
<td>---</td>
<td>---</td>
<td>---</td>
<td></td>
</tr>
<tr>
<td>2 Al heatsink</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>8 unthreaded Al spacer</td>
<td>92510A690</td>
<td>Aluminum Unthreaded Spacer</td>
<td>McMaster-Carr</td>
<td>$1.76</td>
<td></td>
</tr>
<tr>
<td>4 Flanged shoulder washer</td>
<td>93835A320</td>
<td>PTFE Shoulder spacer w/flange</td>
<td>McMaster-Carr</td>
<td>$3.88</td>
<td></td>
</tr>
<tr>
<td>4 TO-220 Isolation pad</td>
<td>BER220-ND</td>
<td>TO-220 Isolator</td>
<td>Digikey</td>
<td>$0.53</td>
<td></td>
</tr>
<tr>
<td>8 1/2&quot; 4-40 Philips head screw (heatsink conn)</td>
<td>91772A110</td>
<td>SS Pan Head Phillips Screw 4-40 1/2&quot; length</td>
<td>McMaster-Carr</td>
<td>$0.33</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>BATTERY CONNECTION HARDWARE</th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>4 LiFePO4 Battery Cells</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>4 Cu Standoffs</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>4 Al Standoffs</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>8 1/2&quot; 10-24 Philips head screw (battery conn)</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Total Items</th>
<th>Total Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>149</td>
<td>$141.33</td>
</tr>
</tbody>
</table>
OBPP ERRATA

SENIOR DESIGN II - SPRING 2011
LPRDS CMS - OBPP HARDWARE DESIGN ERRATA

All directions refer to the PADS Layout file, and are referenced with that point of view.
Nested Items have the same comment as the parent item.

1. Fuse clip should be moved down, to avoid interfering with the connector clip.

2. Connector decal should include a silkscreen of the physical outline of the component.

3. IL611A should be taken out and replaced with two PC123 optocouplers. The associated resistors R2 and R7 should be replaced with 255 Ohm resistors (510||510)

4. The decal for the PIC18F2423 (refdes=U24) should have longer pin pads, for convenience
   A) MCP9700 (refdes=U7,U13,U17,U21)
   B) AD22105 (refdes=U23,U25)
   C) ADUM2250 (refdes=U3)

5. Thermal reliefs should be added for LEDs and headers, as these are most likely to need replacing.

6. Refdes silkscreen for R24 is missing. (Bottom layer, between Batt4+ and Batt3-, next to R22).

7. Voltage regulator has the capacitors swapped; the 0.1 uF cap should be between V12 and GND, and the 10 uF cap between V5 and GND.

8. The 10 uF cap on the voltage regulator (refdes=C1, adjusted for Errata #7) was replaced with a 33 uF electrolytic capacitor in accordance with the ESR requirements for the regulator's stability.

9. The resistors in series with the bypass LEDs (refdes=R12,R21,R30,R39) were changed from 510 Ohms to 75 Ohms.

10. The resistors in the Bypass Enable data lines from the microcontroller to the optocouplers (refdes=R11,R20,R29,R37) were changed from 470 Ohms to 225 Ohms (two 510 Ohm resistors in parallel).
    A) Status LED series resistors (refdes=R3,R4,R5).
11. The resistors from IN- to GND in the Temperature Buffer (refdes=R18,R27,R38,R45) were changed from 33 KOhms to 51 KOhms.

12. The resistors for the Current Sensor Buffer (refdes=R9,R10) were removed, and the buffer was changed to be unity gain.

13. Move silkscreen reference designators for R27 and R20 so they are not cut off.
Aggregate Battery Stack
Aggregate Battery Stack (A6)

There are four wire assemblies in the Aggregate Battery Stack which connect the digital and power lines which interface the interconnect between OBPP boards. Due to the method of inter-board communication, the wire assemblies do not have only two connections, P1 and P2, but rather, have several connections which are in a daisy-chain configuration.

When referring to the signal ‘Power’, this document refers to the HV supply lines running into the aggregate battery stack. On the ICD layout of the top-level ESS, the polarities of the voltage supplies are indicated. When referring to the signal ‘I2C & Temp Safety Loop’, this document considers both the lines dedicated to the redundant temperature safety system (RTSS) and the I2C interface to be one wiring harness, with Red, Black and White, and Green and Blue wires used for the RTSS and I2C lines, respectively.

<table>
<thead>
<tr>
<th>W1 – power distribution wire assembly</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pin</td>
</tr>
<tr>
<td>Connects To</td>
</tr>
<tr>
<td>Gauge</td>
</tr>
<tr>
<td>Color</td>
</tr>
<tr>
<td>Signal</td>
</tr>
</tbody>
</table>

| Pin | P6 | P7 | P8 | P9 | P10 |
| Connects To | A5-J1 | A6-J1 | A7-J1 | A8-J1 | W2-P1 |
| Gauge | 12 AWG | 12 AWG | 12 AWG | 12 AWG | 12 AWG |
| Color | Red | Red | Red | Red | Black |
| Signal | Power | Power | Power | Power | Power |

<table>
<thead>
<tr>
<th>W2 - power distribution wire assembly</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pin</td>
</tr>
<tr>
<td>Connects To</td>
</tr>
<tr>
<td>Gauge</td>
</tr>
<tr>
<td>Color</td>
</tr>
<tr>
<td>Signal</td>
</tr>
</tbody>
</table>

| Pin | P6 | P7 | P8 | P9 | P10 |
| Connects To | A13-J1 | A14-J1 | A15-J1 | A16-J1 | ABS-J4 |
| Gauge | 12 AWG | 12 AWG | 12 AWG | 12 AWG | 12 AWG |
| Color | Red | Red | Red | Red | Black |
| Signal | Power | Power | Power | Power | Power |
### W3 – digital communication & temp sensor loop interconnect (boards A1-A16)

<table>
<thead>
<tr>
<th>Pin</th>
<th>P1</th>
<th>P2</th>
<th>P3</th>
<th>P4</th>
<th>P5</th>
</tr>
</thead>
<tbody>
<tr>
<td>Connects To</td>
<td>ABS-J3 &amp; J1</td>
<td>A1-J2</td>
<td>A2-J2</td>
<td>A3-J2</td>
<td>A4-J2</td>
</tr>
<tr>
<td>Gauge</td>
<td>28 AWG</td>
<td>28 AWG</td>
<td>28 AWG</td>
<td>28 AWG</td>
<td>28 AWG</td>
</tr>
<tr>
<td>Color</td>
<td>Grey - Ribbon</td>
<td>Grey - Ribbon</td>
<td>Grey - Ribbon</td>
<td>Grey - Ribbon</td>
<td>Grey - Ribbon</td>
</tr>
<tr>
<td>Signal</td>
<td>I²C &amp; Temp Safety Loop</td>
<td>I²C &amp; Temp Safety Loop</td>
<td>I²C &amp; Temp Safety Loop</td>
<td>I²C &amp; Temp Safety Loop</td>
<td>I²C &amp; Temp Safety Loop</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Pin</th>
<th>P6</th>
<th>P7</th>
<th>P8</th>
<th>P9</th>
</tr>
</thead>
<tbody>
<tr>
<td>Connects To</td>
<td>A5-J2</td>
<td>A6-J2</td>
<td>A7-J2</td>
<td>A8-J2</td>
</tr>
<tr>
<td>Gauge</td>
<td>28 AWG</td>
<td>28 AWG</td>
<td>28 AWG</td>
<td>28 AWG</td>
</tr>
<tr>
<td>Color</td>
<td>Grey - Ribbon</td>
<td>Grey - Ribbon</td>
<td>Grey - Ribbon</td>
<td>Grey - Ribbon</td>
</tr>
<tr>
<td>Signal</td>
<td>I²C &amp; Temp Safety Loop</td>
<td>I²C &amp; Temp Safety Loop</td>
<td>I²C &amp; Temp Safety Loop</td>
<td>I²C &amp; Temp Safety Loop</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Pin</th>
<th>P10</th>
<th>P11</th>
<th>P12</th>
<th>P13</th>
</tr>
</thead>
<tbody>
<tr>
<td>Connects To</td>
<td>A9-J2</td>
<td>A10-J2</td>
<td>A11-J2</td>
<td>A12-J2</td>
</tr>
<tr>
<td>Gauge</td>
<td>28 AWG</td>
<td>28 AWG</td>
<td>28 AWG</td>
<td>28 AWG</td>
</tr>
<tr>
<td>Color</td>
<td>Grey - Ribbon</td>
<td>Grey - Ribbon</td>
<td>Grey - Ribbon</td>
<td>Grey - Ribbon</td>
</tr>
<tr>
<td>Signal</td>
<td>I²C &amp; Temp Safety Loop</td>
<td>I²C &amp; Temp Safety Loop</td>
<td>I²C &amp; Temp Safety Loop</td>
<td>I²C &amp; Temp Safety Loop</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Pin</th>
<th>P14</th>
<th>P15</th>
<th>P16</th>
<th>P17</th>
</tr>
</thead>
<tbody>
<tr>
<td>Connects To</td>
<td>A13-J2</td>
<td>A14-J2</td>
<td>A15-J2</td>
<td>A16-J2</td>
</tr>
<tr>
<td>Gauge</td>
<td>28 AWG</td>
<td>28 AWG</td>
<td>28 AWG</td>
<td>28 AWG</td>
</tr>
<tr>
<td>Color</td>
<td>Grey - Ribbon</td>
<td>Grey - Ribbon</td>
<td>Grey - Ribbon</td>
<td>Grey - Ribbon</td>
</tr>
<tr>
<td>Signal</td>
<td>I²C &amp; Temp Safety Loop</td>
<td>I²C &amp; Temp Safety Loop</td>
<td>I²C &amp; Temp Safety Loop</td>
<td>I²C &amp; Temp Safety Loop</td>
</tr>
<tr>
<td>Item</td>
<td>Part #</td>
<td>Vendor</td>
<td>Mfg</td>
<td>Price</td>
</tr>
<tr>
<td>------</td>
<td>--------</td>
<td>--------</td>
<td>-----</td>
<td>-------</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**W3 - Data Line**

<table>
<thead>
<tr>
<th>Item</th>
<th>Part #</th>
<th>Vendor</th>
<th>Mfg</th>
<th>Price</th>
<th>Qty</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>2' FT. FLAT CABLE, JOCQNL, 28AWG</td>
<td>2PM-90110-0101</td>
<td>Molex</td>
<td>W4</td>
<td>$8.17</td>
<td>1</td>
<td>$8.17</td>
</tr>
<tr>
<td>WIRE BOARD, CONN. SOCKET, TOPS</td>
<td>98110-0101</td>
<td>Molex</td>
<td>W4</td>
<td>$14.82</td>
<td>1</td>
<td>$14.82</td>
</tr>
<tr>
<td>2' FT. FLAT CABLE, JOCQNL, 28AWG</td>
<td>2PM-90110-0101</td>
<td>Molex</td>
<td>W4</td>
<td>$8.17</td>
<td>1</td>
<td>$8.17</td>
</tr>
<tr>
<td>WIRE BOARD, CONN. SOCKET, TOPS</td>
<td>98110-0101</td>
<td>Molex</td>
<td>W4</td>
<td>$14.82</td>
<td>1</td>
<td>$14.82</td>
</tr>
</tbody>
</table>

**W2 - Power Distribution Line**

<table>
<thead>
<tr>
<th>Item</th>
<th>Part #</th>
<th>Vendor</th>
<th>Mfg</th>
<th>Price</th>
<th>Qty</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>1' FT. FLAT CABLE, JOCQNL, 28AWG</td>
<td>2PM-90110-0101</td>
<td>Molex</td>
<td>W4</td>
<td>$8.17</td>
<td>1</td>
<td>$8.17</td>
</tr>
<tr>
<td>WIRE BOARD, CONN. SOCKET, TOPS</td>
<td>98110-0101</td>
<td>Molex</td>
<td>W4</td>
<td>$14.82</td>
<td>1</td>
<td>$14.82</td>
</tr>
<tr>
<td>1' FT. FLAT CABLE, JOCQNL, 28AWG</td>
<td>2PM-90110-0101</td>
<td>Molex</td>
<td>W4</td>
<td>$8.17</td>
<td>1</td>
<td>$8.17</td>
</tr>
<tr>
<td>WIRE BOARD, CONN. SOCKET, TOPS</td>
<td>98110-0101</td>
<td>Molex</td>
<td>W4</td>
<td>$14.82</td>
<td>1</td>
<td>$14.82</td>
</tr>
</tbody>
</table>

**W1 - Power Distribution Line**

<table>
<thead>
<tr>
<th>Item</th>
<th>Part #</th>
<th>Vendor</th>
<th>Mfg</th>
<th>Price</th>
<th>Qty</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>2' FT. FLAT CABLE, JOCQNL, 28AWG</td>
<td>2PM-90110-0101</td>
<td>Molex</td>
<td>W4</td>
<td>$8.17</td>
<td>1</td>
<td>$8.17</td>
</tr>
<tr>
<td>WIRE BOARD, CONN. SOCKET, TOPS</td>
<td>98110-0101</td>
<td>Molex</td>
<td>W4</td>
<td>$14.82</td>
<td>1</td>
<td>$14.82</td>
</tr>
<tr>
<td>2' FT. FLAT CABLE, JOCQNL, 28AWG</td>
<td>2PM-90110-0101</td>
<td>Molex</td>
<td>W4</td>
<td>$8.17</td>
<td>1</td>
<td>$8.17</td>
</tr>
<tr>
<td>WIRE BOARD, CONN. SOCKET, TOPS</td>
<td>98110-0101</td>
<td>Molex</td>
<td>W4</td>
<td>$14.82</td>
<td>1</td>
<td>$14.82</td>
</tr>
</tbody>
</table>

**SUGGESTED FULL INTEGRATION CONNECTOR BOM**

<table>
<thead>
<tr>
<th>Item</th>
<th>Part #</th>
<th>Vendor</th>
<th>Mfg</th>
<th>Price</th>
<th>Qty</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**URD-5CS-2011** AP, the necessary connectors and making BOM, the system will not be built.
<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
<th>Price</th>
<th>Vendor</th>
<th>Total</th>
<th>Vendor #</th>
<th>Part Des</th>
<th>Mil</th>
<th>Part #</th>
</tr>
</thead>
<tbody>
<tr>
<td>3Pt.</td>
<td>24 AWG Wire, Black</td>
<td>$0.03</td>
<td>187415ND</td>
<td>$0.03</td>
<td>1</td>
<td>24 AWG Wire, Black</td>
<td>1</td>
<td>24 AWG Wire, Black</td>
</tr>
<tr>
<td>3Pt.</td>
<td>24 AWG Wire, Red</td>
<td>$0.03</td>
<td>187415ND</td>
<td>$0.03</td>
<td>1</td>
<td>24 AWG Wire, Red</td>
<td>1</td>
<td>24 AWG Wire, Red</td>
</tr>
<tr>
<td>3Pt.</td>
<td>Test Data Cable 1</td>
<td>$0.48</td>
<td>19068</td>
<td>$0.48</td>
<td>1</td>
<td>Test Data Cable 1</td>
<td>1</td>
<td>Test Data Cable 1</td>
</tr>
<tr>
<td>3Pt.</td>
<td>20AWG, Black, 20AWG, red</td>
<td>$0.35</td>
<td>19068</td>
<td>$0.35</td>
<td>1</td>
<td>20AWG, Black, 20AWG, red</td>
<td>1</td>
<td>20AWG, Black, 20AWG, red</td>
</tr>
<tr>
<td>3Pt.</td>
<td>Test Data Cable 2</td>
<td>$0.49</td>
<td>19068</td>
<td>$0.49</td>
<td>1</td>
<td>Test Data Cable 2</td>
<td>1</td>
<td>Test Data Cable 2</td>
</tr>
<tr>
<td>3Pt.</td>
<td>Test Data Cable 3</td>
<td>$0.49</td>
<td>19068</td>
<td>$0.49</td>
<td>1</td>
<td>Test Data Cable 3</td>
<td>1</td>
<td>Test Data Cable 3</td>
</tr>
</tbody>
</table>

**Note:** The table above lists items with their respective costs and descriptions.
<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
<th>Qty</th>
<th>Price</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>10G OCM TRANS</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>2</td>
<td>10G OCM TRANS</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>3</td>
<td>10G OCM TRANS</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>4</td>
<td>10G OCM TRANS</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>

**Total** | $16.51