INTELLIGENT BATTERY CONTROLLER

By

Jed Boyer (jedmb2)

Christopher Jones (chjones2)

Joseph Roche (jproche2)

Final Report for ECE 445, Senior Design, Fall 2020

TA: Dean Biskup

09 December 2020

Project No. 21

Abstract

The Intelligent Battery Controller is a microcontroller-based switching circuit for automotive and marine battery banks. Specifically, this design monitors voltage levels of two batteries and identifies whether the battery's voltage has dropped to a predetermined value. When this occurs, the resulting design allows for control signals to be sent out to relays that can connect or disconnect the batteries to preserve total battery storage. To allow for user interface, there are three switches that each have a corresponding function: system turn on/off, engine start, both batteries on, and Bluetooth on/off. From testing the device, these control signals work properly, but further testing will need to be done with a working relay system. There are plans to add Bluetooth or RF capabilities to allow for distance gauging from a cellphone or keyfob.

Contents

1. Introduction

In many recreational vehicles, such as boats and campers, dual battery systems are used to increase the runtime between charge sessions. In this time period, devices such as radios, depth finders, lights, and pumps are drains on batteries. If the operator of a vehicle is not careful, this can leave him or her stranded without a way to start an engine and replenish the batteries. The existing method of counteracting this issue is adding a battery selector switch. This solution is good because it gives the user the added ability to alter how the batteries are being charged and drained. By this method, manual switches can allow for distinct reserve batteries to be security against over discharge. Additionally, both batteries can be connected in parallel, for charging, and completely shut off, for storage. The problem with this setup is that it is manual, which introduces human error. It is this challenge that we have set out to address. By expanding on the existing principals of the manual switch, we have ambitions of making an intelligent battery controller. This controller will allow for automated processes of switching inputs to allow for longer runtimes and safe levels of reserves. Our improved device will actively keep track of the charge in the batteries and precisely switch batteries when it is deemed necessary. Then, it will completely stop the discharge when just enough energy remains to start an engine. Allowing human control is a must, so we will allow for the operator to reconnect the batteries in an “override” function that will allow the engine to be started. Additionally, we will provide for the health of the deep cycle batteries which prefer deep discharge cycles. Eliminating human error will mean less stress on the end user, and fewer instances where batteries have been discharged too far to start an engine. In these instances, the use of our device could mean the difference between a user being stranded in a difficult position, whether that be in the forest, desert, or on a large body of water, and making it to a safe and secure location.

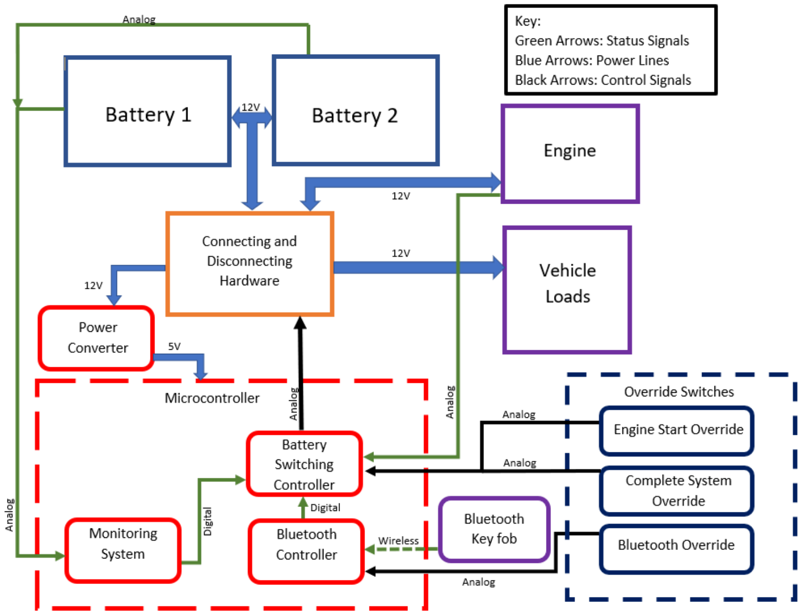
In the following pages, we will outline our design and implementation struggles as we made progress towards developing a proof-of-concept device. Several of the components intended for use in our initial design were deemed inappropriate, arrived nonfunctioning or did not operate as advertised. These challenges required us to adapt or find replacement parts. In the end, we were successful in demonstrating our basic principles and feel confident that with modifications to increase the scale size and meet regulatory specifications, our device could be an asset to operators of all skill levels in many industries.

2 Design

2.1 Block Diagram

Figure 1 shows the block diagram for the Intelligent Battery Controller.

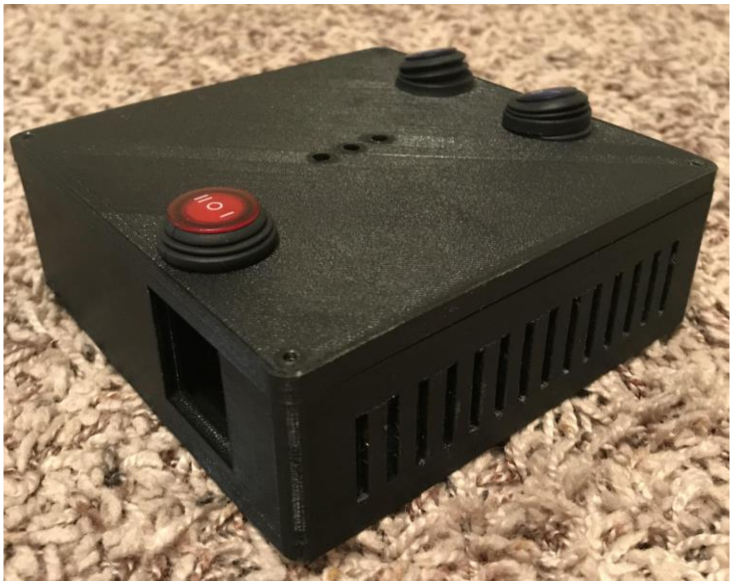
Figure 1 - Block Diagram



2.2 Physical Design

The case for the device's components was designed in AutoDesk Fusion360 and was 3D printed using PLA. It was constrained to be as close as we could possibly make it to the size of existing switches. The hardware that holds the top and bottom of the case together are #10 stainless steel screws with a length of 3/4". These screws are also used internally to hold clamps for the PCB, and in the holes of the various modules to attach them to the case. The screw holes are completely printed into the designed case to allow for easy line up of the parts. See Figure 2 directly below this section for reference. Further details on the structure are provided below in sections 2.2.1 and 2.2.2. Unfortunately, we were unable to put the device into the box due to timing constraints of the semester, but we did verify that everything would fit if we had been able to insert the parts and debug in the case.

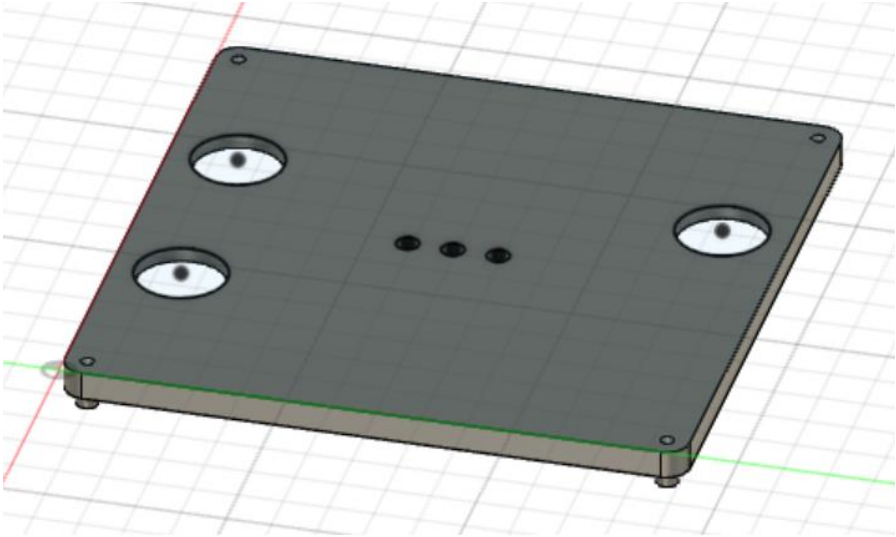
Figure 2 - Case Design



2.2.1 Lid (Top) Physical Design

For reference, the lid prior to printing can be seen below in Figure 3. Overall, it is a ¼” thick piece of plastic with holes to house 3 override switches and 3 status LEDs. The switches that were selected were panel mount, meaning they required a maximum of 1/8” of material to clamp to. For this reason, the area underneath the switches had to be recessed to hold them securely. In the center of the lid is where the LEDs are mounted in the through holes. In the design, these holes were 5mm in diameter to accept a 5mm LED. However, after printing we noticed that we could not fit the diodes though. After drilling to a slightly larger size, they fit perfectly. On the bottom of the lid are locating pins around the screw holes to make locating the top on the base easier.

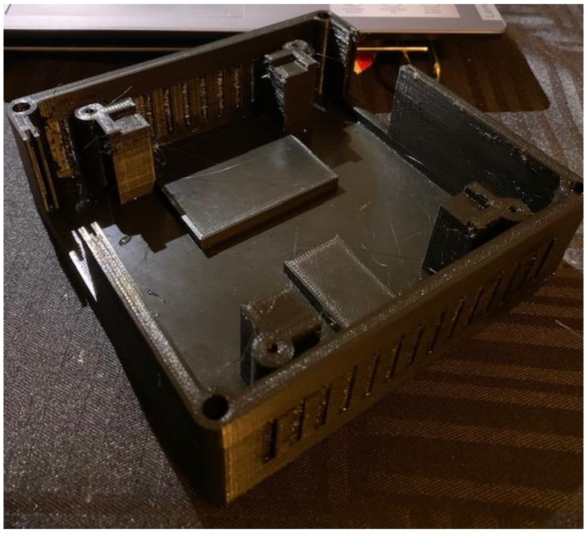
Figure 3 - Lid as Seen in AutoDesk Fusion360



2.2.2 Base (Bottom) Physical Design

For reference, the printed base can be seen below in Figure 4. In that image, it is obvious that its dimensions match those of the lid to allow for a nicely fitted package. Overall, the box is only 5.34" x 5.34". On what is considered the top and bottom of the box, there are ventilation holes that allow for airflow to get to the components. Within the perimeter, along the bottom, there are module pads where we would mount the relay board and Bluetooth module. The larger of the two pads is present for the relays and the smaller one has been sized to hold the Bluetooth module. The intention of the pads is to provide an area of increased thickness to screw these modules to. Additionally, there are four platforms that rise from the bottom to hold the PCB in a recessed tray. On the backs of these pillars, there are screw holes on top to allow for the PCB to be snugly clamped down with additional plastic fingers. The last features of the box are the holes in the walls that allow for wires to be run through. These feature slots that can hold an addition to the wall which would be inserted once the wires and innards have been placed. This design consideration was implemented because it allowed for us to insert and remove wires without disassembling any part of our circuit.

Figure 4 - Bottom of the Case



2.3 Components/Subsystems

2.3.1 PCB Design

The Printable Circuit Board (PCB) is the foundation for the connections of our electrical circuit. It houses holes and pads to mount the various components necessary for our design. The design of the PCB is shown in Figure 5, with numbers in the figure corresponding to different devices and circuit elements that operate the design listed in Table 1.

Figure 5 - PCB Schematic

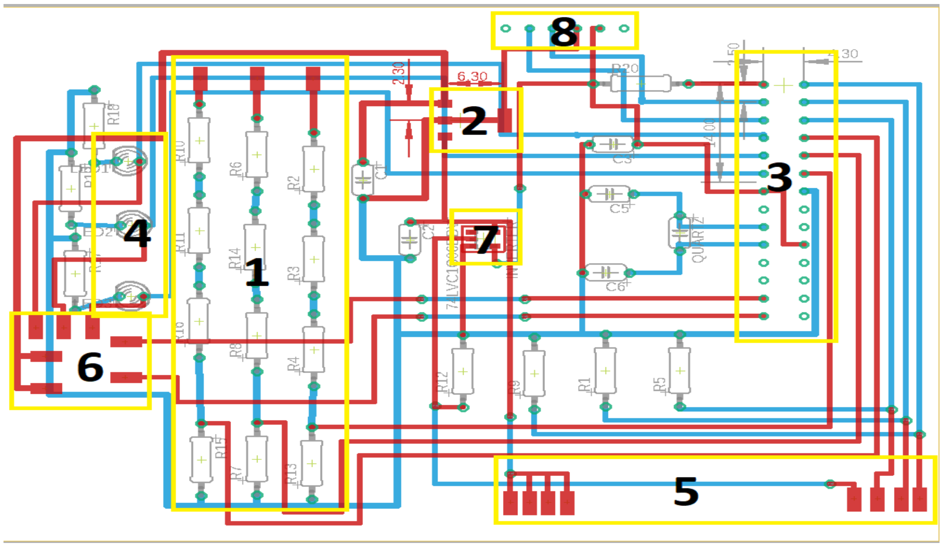


Table 1 - Circuit Elements

Number	Design Element
1	Voltage Divider
2	5V Voltage Regulator
3	ATmega328p-pu
4	Status LEDs
5	Switch Control Pads
6	Power, Relay Output Pads
7	Inverter
8	Bluetooth Module

2.3.1.1 Design Element

The voltage divider is a necessary element in our circuit due to the constraints of the ATMEGA328-PU microcontroller. The microcontroller can only accept a maximum voltage of 5.5V as an input, which our 12V battery exceeds. To overcome this, we used a series of four resistors, each with 100kOhms of resistance to divide the incoming battery voltage by four, bringing the maximum voltage to about 3V. There are three voltage divider circuits: one for Battery 1, one for Battery 2, and one for the Engine-Start signal. The voltage across the last resistor in the series for each circuit is connected to an analog input pin on the ATMEGA328-PU microcontroller.

2.3.1.2 5V Voltage Regulator

The voltage regulator we used in our design is the MCP1799 low-dropout voltage regulator with a fixed output of 5V. We incorporated the 4-pin design, footprint SOT-223. This component was used to supply 5V Vcc converted from 12V battery input to the switches and ATMEGA328-PU microcontroller. This was the replacement to the regulator we had first sourced in our design, the LM2940CT-5.0. The previous regulator we tried to use did not match the footprint created for it in the PCB design. In addition to this, its method of operation was not a fixed output of 5V, but rather a 5V difference to the input voltage (i.e., 12V input outputs 7V).

2.3.1.3 ATMEGA328-PU

The ATMEGA328-PU is the core component to our battery switching control design. Using the Arduino IDE created by Atmel, we were able to upload software onto the microcontroller that allows it to accept digital and analog inputs from sources such as the voltage divider or override switch controls. Using these input signals, the microcontroller can output status and control signals to illuminate our status LEDs and energize relays. The hole size and spacing of the PCB design did not match that of the microcontroller, leading to a workaround of jumping wires, and the use of a breadboard with the microcontroller mounted on it.

2.3.1.4 Status LEDs

The status LEDs are necessary to allow the user to identify which state of operation the device is currently in. The ATmega328-PU microcontroller sends digital outputs to the LEDs to show when the device is active, inactive, or when Battery 1 or Battery 2 is being connected via relay controls.

2.3.1.5 Switch Control Pads

The switch control pads are needed to facilitate the use of our override switches. The switch pads receive 5V from the voltage regulator in combination with a resistor to send signals of either digital high or digital low to analog input pins on the microcontroller.

2.3.1.6 Power, Relay Output Pads

This area of pads provides a few necessary functionalities for our device. The leftmost pads are where our 12V batteries can connect to the 5V regulator. The upper pads are direct signals from each of the status LEDs, the two related to Battery 1 and Battery 2 able to connect to relay controls. The rightmost pads are additional digital outputs that can be used for relay controls to avoid high current outputs on a single microcontroller pin.

2.3.1.7 Inverter

The intent of the inverter in our design was to provide a way to provide a buffer between the 5v output of the regulator and our microcontroller. We had later found the inverter we used to not facilitate the functionality we desired, and after difficulty with soldering such a small component numerous times, we decided to bypass this device by jumping the switch controlling its input pin to the Vcc rail of the PCB, providing power to the ATMEGA328.

2.3.1.8 Bluetooth Module

The PCB included mounting holes designed for the HC-05 Bluetooth module. This module was intended to be used to provide additional functionality to our design. The module was not compatible with the method we intended to use for our functionality, so it was not included.

2.3.2 Relay Module

The relay module we used is the SunFounder 2 Channel DC 5V Relay Module. This module has two relays with an energizing voltage of 5V, and current draw from 40 to 80mA. The relay was intended to be used in conjunction with the ATMEGA328-PU microcontroller to connect our batteries to our load. The functionality of the relay is explained via truth table in Table 2.

Table 2 - Relay Outputs

Input	Relay 1 Energized	Relay 2 Energized
IN1 Low, IN2 Low	No	No
IN1 Low, IN2 High	No	Yes
IN1 High, IN2 Low	Yes	No
In1 High, In2 High	Yes	Yes

This would be used in conjunction with our test bench to simulate engine starting and passive loads, but the relay module we received was defective. The continuity between contacts on the relay module would not change as one would expect, even when energized. This meant that the same two contacts on each relay were permanently connected, and as such, we could not use the relay as we had intended.

2.3.3 Override Switch Functions

To allow for user control over our device, we have three switches with override functions that change the behavior of the device according to the user's needs. This system consists of two single-pole, single throw switches, and one single-pole, double throw switch. The functionalities of the switch controls are as follows in Table 3. For the columns with colored boxes, a colored space corresponds to active, a blank space corresponds to inactive, and an X refers to that input not being influenced by the override switch.

Table 3 - Switch Functions

Switch	Device On	Battery 1 Connected	Battery 2 Connected	Bluetooth Disabled
Power Off				Y
Connect Both	🟩	🟥	🟨	N
Engine Start	🟩		🟨	N
Bluetooth	🟩	X	X	Y

2.4 System Theory of Operation

The theory of operation of our device is reliant on the functionality of the microcontroller. Overall, the microcontroller manages what the status of discharge should be by monitoring inputs from the user and batteries. When the engine is running, the microcontroller receives a signal on the proper set of resistors that tells it to connect both batteries because the device should be in a charge state. The only other way of connecting both batteries is to use the operator override switch corresponding to connect both. The main functionality of the device is in monitoring battery levels to see when a threshold is crossed. Once this set point is hit, the first battery disconnects and the second begins to draw. Once again, we continue to monitor until the second threshold is hit, when enough charge is stored to start the engine. At this point, the second battery is disconnected. From here, the only way out of this state is with the start override button, which connects the output for 20 seconds to allow for the engine to be turned on. At this point, the microcontroller switches battery titles to allow for shared discharge cycles, and the operation continues in the same orientation.

2.4.1 Checking States and Statuses

Checking states is done with analog reads from the voltage divider to the chip, which come in as digital values due to the microprocessor’s analog to digital converter. These digital values range from 0 to 1023, where 1023 represents five volts. The equation for creating a useful voltage out of this number is found here in Equation (1).

$$V = (4\,resistors * Digital\,Value * 5V)/1023.0 \tag{1}$$

This is the voltage that the microprocessor operates on when it makes its conditional decisions. Additionally, more analog inputs are connected to each of the override buttons. These buttons are already referenced to 5V since they are sent power by the regulator on the board, so no conversion was done on their values, and no voltage divider was necessary. Instead, all the microprocessor does it simply check its conditionals against the digital values provided. After all these decisions were made, the microprocessor simply turns outputs from the chip to the relays and output LEDs high or low according to the correct state.

3. Design Verification

The Intelligent Battery controller has been tested thoroughly. In Appendix A, these tests can be viewed. Most of these tests passed but showed that the system has some flaws that need to be fixed.

3.1 Microcontroller

All requirements for the Microcontroller passed. The ATMEGA can read battery voltage levels of both batteries, take inputs from switches, and determine the necessary output for the relays.

Table 4 shows the outputs for each relay and switch input.

Table 4 - Microcontroller Verification Results

Test #	Both Battery Switch	Engine Override Switch	Battery 1 above 9V?	Battery 2 above 11V?	Voltage for Relay 1	Voltage for Relay 2
Test 1	OFF	OFF	Y	Y	4.83V	0V
Test 2	ON	OFF	Y	Y	4.84V	4.83V
Test 3	ON	OFF	N	Y	4.83V	4.84V
Test 4	ON	OFF	Y	N	4.82V	4.86V
Test 5	ON	OFF	N	N	4.84V	4.85V
Test 6	OFF	OFF	N	Y	0V	4.83V
Test 7	OFF	OFF	Y	N	4.85V	0V
Test 8*	OFF	ON	N	N	0V	4.83V

***This test will not have an indefinite state. The voltage for Relay 2 goes to 0V after 20 seconds.**

Test one shows what a typical, no-switches-on output should be. The Both Battery Switch works as seen from tests two, three, four, and five. Test six shows that the microcontroller does switch which battery is driving the load when battery 1 reaches a voltage level below 9V. Test seven depicts the battery 1 taking on the load with battery 2 shut off due to lack of voltage. Finally, test eight verifies that the engine override switch works as intended.

Table 5 shows the Arduino IDE's values being interpreted and what the multimeter showed. These values were obtained using one of the 12V battery packs. The values ranged from 4-12V using a potentiometer.

This test also showed that the user interface switches worked. Requirements 7 and 8 are fulfilled.

Table 5 - Microcontroller Battery Sensing Results

Test #	Voltage Measured by Arduino IDE	Voltage Measured by Multimeter
1	11.98V	12.04V
2	11.00V	11.02V
3	9.95V	9.96V
4	8.02V	8.04V
5	6.98V	7.02V
6	5.99V	6.01V
7	5.03V	5.06V
8	4.02V	4.03V

The microcontroller's worst deviation from the multimeter measurement is 0.5%. This deviation is suitable for this design as we are only reading these voltages to determine when to switch the batteries off. The results do not need to be extremely accurate.

3.2 Voltage Regulator

The voltage regulator tested correctly. All the requirements were met.

Table 6 shows the results of testing. The voltage was varied using a potentiometer and one of the battery packs.

Table 6 - Varying Voltage on the Voltage Regulator Results

Test #	Voltage at Input Pin	Voltage at 5V Output Pin
1	12.38V	5.05V
2	12.02V	5.03V
3	11.05V	5.03V
4	10.07V	5.02V
5	7.05V	5.03V
6	6.53V	5.04V
7	6.08V	5.02V
8	5.57V	5.03V
9	5.32V	4.82V

The voltage regulator keeps its 5V output up until 5.5V. This meets the requirement for the design.

3.3 Power Switch

The power switch works as intended. The results are shown in table 7.

Table 7 - Power Switch Results

Power Switch State	Voltage on 5V Rail
ON	4.98V
OFF	0V

3.4 Relay Module

The Relay module purchased did not work. The relays themselves did not activate correctly when voltage was applied to the latch pin. Table 8 shows the results to reflect this failure.

Table 8 - Relay Module Results

Voltage on Latch Pin of Relay 1	Voltage on Latch Pin of Relay 2	Output Pin Voltage of Relay 1	Output Pin Voltage of Relay 2
4.83V	0V	0V	0V
0V	4.84V	0V	0V
0V	0V	0V	0V
4.80V	4.80V	0V	0V

This failure could be accounted for as either damaged in shipping or the producer's fault. More testing would need to be done with a new relay module. Preferably with a new manufacturer.

3.5 Bluetooth Module

The Bluetooth module was never implemented. This part of the design was an add-on to the design to increase some complexity. It can be introduced as a future implementation. As a result, the verifications for the Bluetooth module failed. No testing has been done.

4. Costs

4.1 Parts

Table 9 shows the parts that were used on this project. The bulk cost for this device has been excluded due to it not being fully tested nor operational. It is ill-advised to start purchasing in bulk until further testing on a fully functional device is done. It is also advised that a new relay module be sourced.

Table 9 - Parts Cost Breakdown

Product	Manufacturer	Retail Cost ($)	Amount	Cost ($)
14 AWG Wire	Fermerry	13.90	1	13.90
2 Channel DC 5V Relay Module	SunFounder	6.79	1	6.79
2Set 8 x AA Thicken Battery Holder	QTEAKTAK	6.99	1	6.99
24 AA Batteries	Duracell	16.21	1	16.21
Twidec/5Pcs 12V 20A Waterproof Lighted Round Rocker	Twidec	9.99	1	9.99
3 Pin Round Rocker Toggle ON/Off Waterproof	WINOMO	7.29	1	7.29
16 MHz Crystal Clock	TXC CORPORATION	0.25	10	2.50
HC-05 Wireless Bluetooth RF Transceiver	HiLetGo	7.99	1	7.99
ATMEGA328-PU	Microchip Technology	1.90	10	19.00
5 100W Watt Shell Power Aluminum Housed Case Wirewound Resistor 4R Ohm	Comidox	8.99	1	8.99
22pF Capacitor	Vishay Beyschlag/Draloric/BC Components	0.18	10	1.82
10K Ohm Resistor	Xicon	0.10	30	3.00
22 uF Capacitor	TDK Corporation	1.49	5	7.45
0.47 uF Capacitor	KEMET	0.20	10	2.00
5V Regulator	ON Semiconductor	0.13	10	1.30
Total				**115.22**

4.2 Labor

Reports from job recruitment sites list the average annual salary of an entry-level electrical engineer in the United states to be from $55,000 [1] to $75,000 [2]. Using these sources as a basis, we will define an entry-level electrical engineer salary to be $65,000, or $31.25/hour.

Our development costs for this design are estimated as follows: Assuming 7 hours of work weekly for 3 entry-level engineers over the course of 7 weeks at an hourly rate of $31.25/hour, we obtain the following labor cost equation Equation (2):

$$Cost_{labor} = 7_{hours} * 3_{workers} * 7_{weeks} * 31.25_{\frac{\$}{hour}} = \$4{,}593.75 \quad (2)$$

Following the cost of labor is Table 10 showing our resulting schedule. This schedule is what our team did each week broken down to individual tasks. Some of these tasks were moved around or changed. For example, anything related to Bluetooth was solely dedicated to debugging the main design.

Table 10 - Work schedule determined from the design document

Week of:	Chris (ONL) Task:	Jed Task:	Joey Task:
10/5	Go through Design Review feedback, modify designs for device and test bench	-Order parts that have been approved -Research Bluetooth interface	Go through Design Review feedback, modify designs for device and test bench
10/12	-Review PCB design -Order final parts -Develop final device Schematic	Create preliminary app needed for Bluetooth application	Design PCB and order
10/19	Design case for the device (Black Box), 3D print prototype and assemble	Build the device internals	Build the device internals
10/26	Write code to upload to board for device operation	Debug Bluetooth functions	Assemble Test Bench
11/2	-Based on feedback, make sure the device is meeting our specified stipulations. -Order replacement parts	Debug device, make alterations to code, physical design, etc.	Debug test bench, make alterations to physical design, etc.
11/9	-Analyze power data and feedback -Based on device behavior, make sure the device is meeting our specified stipulations.	Debug device, make alterations to code, physical design etc.	Debug device, make alterations to code, physical design etc.
11/16	Begin Final Paper	Demonstration Prep/ Work out final bugs in system	Demonstration Prep/ Work out final bugs in test bench

11/23	Final Paper	Create Final Presentation	Begin Final Paper
11/30	Final Paper	Final Presentation last minute details / Check	Final Paper
12/7	Final Paper	Final Paper	Final Paper

5. Conclusion

5.1 Accomplishments

Looking back on this project, we were able to successfully implement the core functionality of our small-scale device. This should simplify a transition to a larger revised device on the basis established though this semester's work. On a personal level, we were each able to fine tune our problem solving and team working skills in preparation for our future careers as engineers.

5.2 Uncertainties

Our uncertainties on this project were fairly limited to the parts that we were unable to verify. First and foremost was the intended application of Bluetooth functionality. Due to time constraints of the semester, we opted to focus on the main functionality of the device rather than the add on module. Additionally, due to the misrepresentation of the initial voltage regulator that was purchased, we were unable to verify that the current regulator was able to meet our requirements of producing 5V for all inputs from 6V-13V. In the future, we would test this functionality by bringing the device to the lab and using a power supply to easily alter voltages. In the same realm, we lacked the tools needed to verify that our power consumption requirement was met. Theoretically, we were able to meet this self-imposed limit, but without testing it would be hard to guarantee. Finally, we were unable to use our testbench and incorporate the relay module into the working device. This was out of our control as the relay module that we were sent was broken and possibly has other issues.

5.3 Ethical considerations

The device we created, processes we employed, and treatment of team members met all expectations addressed in the IEEE code of ethics [3]. In the order found in section 7.8 of the IEEE Policies documentation, our group and/or device:

1. is safe to use and prioritizes the welfare of the operator
2. educates society about the benefits of emerging technology by implementation of an intelligent system
3. avoided all conflicts of interest
4. avoided all bribery and unlawfulness
5. accepted recommendations from colleagues and educators and were honest about our work in all aspects
6. improved technical abilities and assisted others with them after we were qualified
7. treated everyone respectfully, avoiding any form of discrimination
8. avoided all forms of harassment
9. did not injure others or anything associated with them
10. supported each other in following ethical choices

In our project it was important that we avoided risks whenever possible. In our small scale with limited technical knowledge, we avoided the use of lead acid batteries to protect ourselves. Additionally, we ensured that the wires we used were of respectable size for the current we were carrying and avoided shorting leads at all costs. As the project continues, it is important that we keep in mind the specifications that will govern the full-scale device from various organizations. This is where design decisions become more important to mitigate the harm to operators and their respected property.

5.4 Future work

To create a functional product for the harsh environments the device would be used in, we would need to make changes that would allow for the device to be waterproof and shockproof. First and foremost, the electrical components would need to be coated to ensure that they are waterproof. Additionally, waterproof relays would need to be used and the case design altered to remove the ventilation that allows water to enter the device. The box itself would need to be isolated to prevent shock from boats rocking in the waves and campers being pulled down the road from harming the device. These considerations, plus many more would come to light as we begin to address United States Coast Guard, Underwriters Laboratory or International Standard Organization approval. These would be required for the device to be legally and/or safely used for our intended applications on boats, campers, trucks, ATVs, etc.

The final hurdle that we would need to cross would be increasing the scale of our device for larger applications. To do this, we would need larger cables and relays to handle the higher ampacities, and to ‘tune’ the device to the intended application by altering the set voltages that turn the batteries on and off.

References

[1] ZipRecruiter, “Entry Level Electrical Engineer Annual Salary,” *ZipRecruiter*, Sep 24, 2020. [Online]. Available: https://www.ziprecruiter.com/Salaries/Entry-Level-Electrical-Engineer-Salary [Accessed Sep 30, 2020]

[2] Glassdoor, “Salary: Entry Level Electrical Engineer Salaries,” Glassdoor, September 21, 2020. Available: https://www.glassdoor.com/Salaries/entry-level-electrical-engineer-salary-SRCH_KO0,31.htm [Accessed Sep 30, 2020]

[3] Institute of Electrical and Electronics Engineers, "7.8 IEEE Code of Ethics," *Institute of Electrical and Electronics Engineers.* 2020. [Online]. Available: http://www.ieee.org/about/corporate/goverancne/7-p8.thml [Accessed: Sep 15, 2020]

[4] DroneBot Workshop, “From Arduino Uno to ATmega328 – Shrinking your Arduino Projects,” 2018. [Online]. Available: https://dronebotworkshop.com/arduino-uno-atmega328/ [Accessed: Sep 28, 2020]

Appendix A Requirement and Verification Table

Table 11 shows the requirements, verification, and status of the verification for the Intelligent Battery Controller.

Table 11 - System Requirements and Verifications

Requirement	Verification	Verification status (Y or N)
1. Voltage Regulator Outputs 4.5-5.5V to the microcontroller at input voltages ranging from 6-13V.	1. Verify voltage output with a multimeter (FLUKE 115). a. Switch the dial to measure DC voltage on the multimeter. b. Connect the test probes to the multimeter such that the black test probe is connected to the COM port on the multimeter and the red test probe is connected to the V/Ω/Hz port. c. Place the COM/black test probe of the voltmeter at the ground of the battery terminal. d. Place the red probe to the 5V side of the voltage regulator. e. Verify on the screen of the multimeter that DC voltage is stepped down to 4.5-5.5V. f. Continue to check this screen as voltages vary from the battery. g. (optional) if in the lab, hook a DC voltage supply and vary the voltages ranging from 6-13V on the input pin of the voltage regulator.	Y
2. Microcontroller polls voltage levels from the batteries every 20-30 seconds.	2. This can be checked in the Arduino IDE in the control window on the breadboard before installing it to the PCB. a. Open the Arduino IDE software b. Make the necessary	Y

	connections to the Arduino for modifications as seen in Figure 6. c. Make sure the code is uploaded to the ATMEGA through the Arduino board. d. On the breadboard, place a wire connection to PIN 9 on the ATMEGA. e. Grab any battery or voltage source with a known voltage value. f. Touch the wire connected to PIN 9 on the ATMEGA to the positive terminal of the battery. g. Connect the negative terminal of the battery to the same GND as the ATMEGA or PIN 27. h. Look at the Arduino IDE control window and see if the voltages on the screen match the voltage source and are appearing in 20-30 second intervals.	
3. Microcontroller sends out digital signals 4.5-5.5V until a voltage criterion*, switch criterion, or Bluetooth criterion is met.	3. This can be checked in the Arduino IDE control window and with the ATMEGA on the breadboard. a. Open the Arduino IDE software. b. Make the necessary connections to the Arduino for modifications as seen in Figure 6. c. Make sure the code is uploaded to the ATMEGA through the Arduino board. d. On the Arduino, place wires on PINs 26 and 25 e. Get a multimeter (FLUKE 115) and turn the setting to read out voltage. Also, make sure the black and red probes are plugged into the correct ports on the multimeter. f. Place the red probe on the	Y, but Bluetooth was not implemented

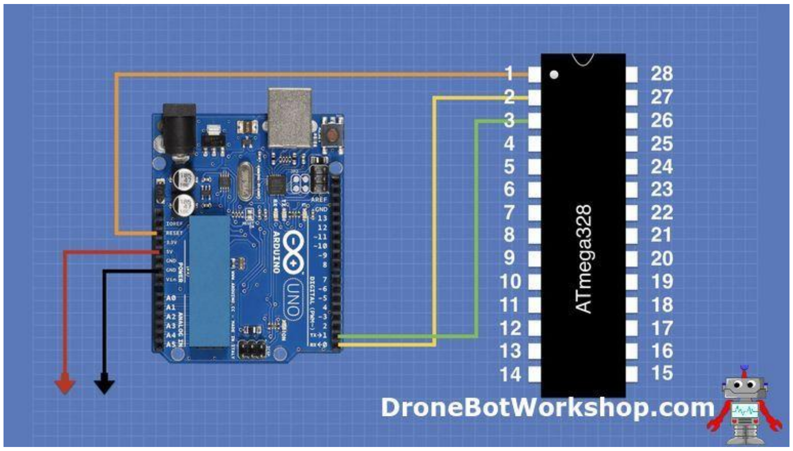
		PIN 26 wire and the black probe to GND. g. Make sure the override switch is turned off. h. The multimeter should read 4.5-5.5V. i. Turn the override switch off. j. The multimeter should now read 0V. k. Do steps 6-10 but place the red probe on PIN 25. l. Turn the override switch off. m. Place the red probe of the multimeter on PIN 26, again. n. Take the Bluetooth keyfob app on one's phone and walk 30-50 feet away from the Bluetooth module. o. Press the off button on the app. p. Go back to the multimeter. It should read 0V. q. Repeat steps 13-16 but place the red probe of the multimeter on PIN 25. r. Using a 1k potentiometer, connect 1 pin to a 12V battery. Connect another pin to GND. s. Connect PIN 9 of the ATMEGA to the third pin of the potentiometer. t. Place the multimeter's red probe on the same pin as the ATMEGA's PIN 9. u. Turn the potentiometer until the voltage reading is lower than 9V. v. Check PIN 26 of the ATMEGA by placing the red probe there. It should read 0V. w. Place the probe back on the potentiometer. x. Turn the potentiometer until the voltage is above 8V. y. Check PIN 26 of the	

	ATMEGA by placing the red probe there. It should now read 4.5-5.5V. z. Turn the potentiometer back down to below 9V. aa. Put the red probe on PIN 25. bb. Using a different potentiometer, repeat steps 19-25, but the voltage level is now 11V and the ATMEGA PIN to probe is PIN25.	
4. Bluetooth module must be able to receive signals from a mobile app to shut off batteries from 30-50ft away.	4. Testing this requires a multimeter. a. Attach probes to GND and to the 12V rail of the batteries. b. Walk 30-50 feet away. c. Press the “OFF” button. d. Go and check the multimeter. It should read 0V.	N
5. Relay module receives control signals from the ATMEGA and properly switches on and off the corresponding batteries.	5. Testing this requires a multimeter. a. Attach probes to the middle socket and either side socket of either relay. b. Get the system in a state where the batteries much switch (i.e. unplugging battery 1). c. The relays should now switch from their original position and voltage will appear on the multimeter’s screen. d. This voltage will either be the voltage of the battery it is connected to or zero.	N
6. When the Bluetooth override switch is activated, the system will ignore any Bluetooth inputs from the app.	6. Activate the Bluetooth override switch, then check the functionality of all control signals to verify that the Bluetooth signal does not impact operation. a. Run through the standard verification for the microprocessor with the switch off.	N
7. Both batteries will be connected upon activating the engine	7. Measure voltage and current on the load side after activating the switch	Y

override switch.	to confirm the control signal functionality. a. This can be done using a multimeter. b. After activation, place probes on each battery rail in series to check for a current. c. There should be a current coming out of both batteries.	
8. After a period 10-20s after activating the engine start override switch, if the engine is not started, both batteries will be disconnected.	8. Activate the switch, wait 10-20s, then measure voltage and current on the load side to verify the disconnection of the batteries. a. Using a multimeter, check the 12V rail. b. It should read 0V after disconnection. c. After 20-30 seconds with the engine started, verify that the batteries do not disconnect. d. Place probes on the 12V rail. e. It should not be 0V.	Y
9. The power switch works. This means, when the switch is flipped, no power to the system is given.	9. This can be verified using a multimeter. a. Switch the power on. b. Check to see if there is voltage on the 5V rail. There should be 5V. c. Switch the power off. d. Using the multimeter, check for voltage again. There should be 0V.	Y

***Voltage criteria is the determined voltage that is set within the code of the ATMEGA. This means, for our system, when 9V is 'seen' by the microcontroller the signal for that battery will turn off and disallow current to pass through the MOSFET that is controlling that battery. This happens when the other battery meets a voltage level of 10V as well.**

Figure 6 - Proper setup for programming the ATMEGA [4].


1
2
3
4
5
6
7
8
9
10
11
12
13
14
28
27
26
25
24
23
22
21
20
19
18
17
16
15
ATmega328
DroneBotWorkshop.com