By

Jacob Drewniak

Michael Jang

Final Report for ECE 445, Senior Design, [Fall 2018]

TA: Evan Widloski

12, December 2018

Project No. 10

**AUTOMOTIVE DATA LOGGER**

**Abstract**

This report goes into depth about the design, development, manufacturing, and testing of an automotive data logger for the Illini Motorsports FSAE team. The circuit board will store incoming data (RPM, throttle position, brake pressure, etc.) from the car’s numerous sensors and store them into memory. Team members will then be able to easily access this data through several different media in a simple, organized fashion. Ultimately, this will enable proper engineering decision-making in order to facilitate the design, validation, and manufacturing of a Formula-style race car.

**Contents**

[1. Introduction 4](#_Toc532419049)

[1.1 High-Level Requirements 4](#_Toc532419050)

[2 Design 5](#_Toc532419051)

[2.1 External System 5](#_Toc532419052)

[2.1.1 13.5V Battery 5](#_Toc532419053)

[2.1.2 Sensors 5](#_Toc532419054)

[2.2 Power Supply 6](#_Toc532419055)

[2.2.1 Super Capacitor Stack 6](#_Toc532419056)

[2.2.2 Super Capacitor Controller 6](#_Toc532419057)

[2.2.3 5V Regulator 7](#_Toc532419058)

[2.2.4 3.3V Regulator 7](#_Toc532419059)

[2.3 Control Unit 7](#_Toc532419060)

[2.3.1 Microcontroller 7](#_Toc532419061)

[2.3.2 CAN Transceiver 8](#_Toc532419062)

[2.3.3 SD Card 8](#_Toc532419063)

[2.3.4 Temperature Sensor 8](#_Toc532419064)

[2.3.5 Real-Time Clock 8](#_Toc532419065)

[2.4 User Interface 8](#_Toc532419066)

[2.4.1 USB-UART 9](#_Toc532419067)

[2.4.2 Bluetooth 9](#_Toc532419068)

[2.4.3 Antenna and Impedance Matching 9](#_Toc532419069)

[2.5 Software 9](#_Toc532419070)

[3. Design Verification 12](#_Toc532419071)

[3.1 SD Card Interface 12](#_Toc532419072)

[3.2 USB Interface 12](#_Toc532419073)

[3.3 CAN Interface 13](#_Toc532419074)

[3.4 Supercapacitor Power 13](#_Toc532419075)

[3.5 Peripheral Circuitry 13](#_Toc532419076)

[4. Costs 14](#_Toc532419077)

[4.1 Parts 14](#_Toc532419078)

[4.2 Labor 15](#_Toc532419079)

[5. Conclusion 16](#_Toc532419080)

[5.1 Accomplishments 16](#_Toc532419081)

[5.2 Shortcomings 16](#_Toc532419082)

[5.3 Ethical considerations 16](#_Toc532419083)

[5.4 Future work 16](#_Toc532419084)

[References 17](#_Toc532419085)

[Appendix A Requirement and Verification Table 19](#_Toc532419086)

# 1. Introduction

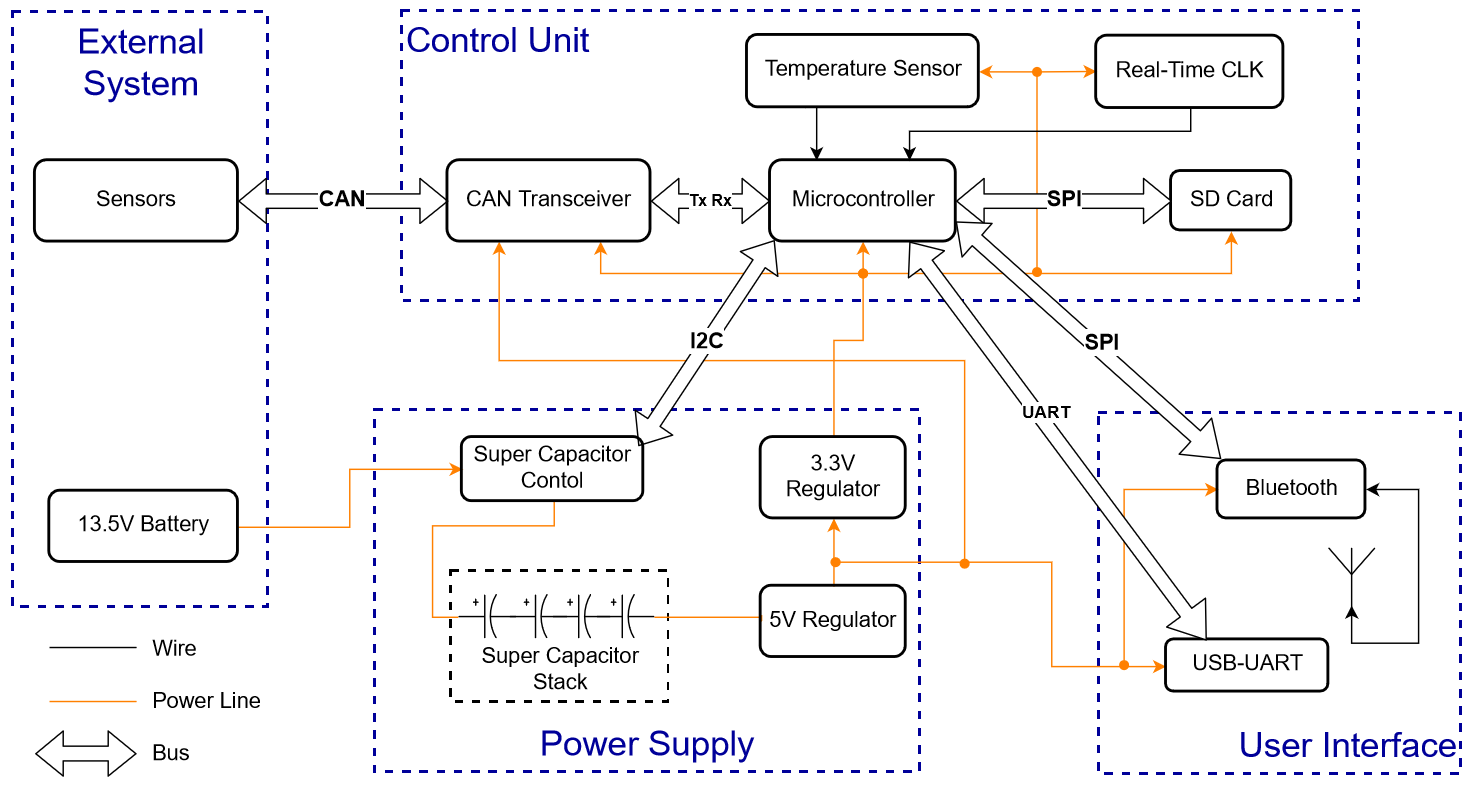
The primary objective of the electronics subsystem of an FSAE team is to collect useful data in support of the design and validation process. This data can range from ignition timing to suspension strain values and is absolutely essential for a finely-tuned race car. The team’s current data logger, along with many commercial data acquisition systems, is made by Vector Informatik but is not ideal for several reasons. The software required to download the data from commercial loggers is difficult to use and quite slow. The files the Vector logger produces have a maximum size of 5.25 MB and are not parsed correctly for simple data extraction. Post processing is necessary to produce usable data. Finally, the team has faced issues with its reliability during testing, often losing or data for seemingly no reason. Replacing this data logger with a different commercially available option is not within the team’s budget, so the best solution is to build our own with a simpler user-interface, better extraction methods, and faster speeds.

The goal of this project is to design a data logger that can save all incoming sensor data at high frequency and make it quickly and easily available to the user through a simple, streamlined interface. This can be done through either USB or Bluetooth connectivity depending on the testing scenario. The team already employs a custom-built sensor processor that converts analog and digital sensor data to the CAN (Controller Area Network) communication protocol, so the data logger must simply log incoming CAN messages from the sensor processor [1]. This simplifies the overall architecture to merely storing and transmitting the data to team members at the necessary times with several protection measures built in to prevent data corruption or losses.

## 1.1 High-Level Requirements

* The system must be able to store up to 4GB of data into non-volatile memory.
* The data stored in memory must be able to be transferred the user through USB and Bluetooth.
* In case the system loses battery power, it must be able to power itself long enough to properly power down and ensure that no data is lost or corrupted in the process.

# 2 Design



**Figure 1 High-Level Block Diagram**

This design has three main components: the power supply, the control unit, and the data transfer mechanism. The power supply module will drive the entire circuit with the proper voltages (3.3V and 5V) required by the chosen chips. There will also be a feature in which we can temporarily power the entire system in case of an electrical power failure. The control unit is responsible for all the data transfer and storage on the board, with the PIC32MZ microcontroller as the brains of the operation. Finally, the user interface will allow the team to be able to read logged data through both USB and Bluetooth connectivity.

## 2.1 External System

The external system is comprised of the components on the car that interact with the board. This block consists of two main aspects of the vehicle: the battery and the sensors that provide the data to be stored.

### 2.1.1 13.5V Battery

The car and all of its electrical components, including the designed data logger, are powered by a lithium-ion battery that is nominally at 12V. Normal operating conditions usually range from 13V-14V. The battery can be charged using a lithium-ion battery charger when the car is off or by the car’s alternator while the engine is running.

### 2.1.2 Sensors

The car has a large number of sensors that are monitored by a separate sensor processing module. That module relays all that data to the logger over CAN to be stored for later use. Sensors give the team important data points such as lateral acceleration, battery voltage, large current draws (water pump, radiator fan, etc.), steering angle, wheel speed, etc. All these data points are essential to log in order to calibrate/tune the car, validate mechanical and electrical designs, and make improvements to future cars.

## 2.2 Power Supply

The power supply subsystem will provide the necessary voltages throughout the board using two regulators with enough current capacity to properly power all circuitry. The main power supply will be provided by the car battery. There will also be backup power circuitry to provide temporary power if battery power is severed during data storage.

### 2.2.1 Super Capacitor Stack

A stack of multiple super capacitors will be used as a power backup for the board if the battery supply is cut off because it is the most space efficient methodology as opposed to electrolytic capacitors. This is necessary so that the system can properly power down without losing any essential data that is in the process of being logged or corrupting the file system that will affect future data storage. The pack voltage must be close to the supply battery voltage of 12V-14V to power the board for about 2 seconds to ensure that no data is being transmitted or stored when the backup loses charge. Because single supercapacitors can rarely exceed between 2.7V and 3V because of dielectric breakdown at higher voltages, a series stack of four is necessary. The buck converter that supplies power to the rest of the board has a minimum input voltage of 6.5V. The absolute maximum current consumption of the board is 500 mA, so the capacitance of the stack can be calculated as followed by Eq. 1.

(Eq. 1)

Because the stack has capacitors in series, individual capacitance can be calculated in Eq. 2.

(Eq. 2)

Accounting for a safety factor of about 10, each capacitor should have a capacitance of around 8F. The chosen supercapacitors have a peak stack voltage of 10.8V and are 10F each and are capable of power the board for over a minute before losing power.

### 2.2.2 Super Capacitor Controller

This IC is connected to all four super capacitors and will function similarly to a battery management system. The monitor functionality includes measuring state of charge, voltage level, and current I/O. The controller aspects of this IC include cell balancing, pack protection, and microcontroller interfacing [2]. The LTC3350 supercapacitor backup controller was chosen to perform this functionality. It controls a stack of up to 4 supercapacitors. Controller functionality includes performing constant current/constant voltage charging of the supercapacitor stack, monitoring stack currents and voltages with a 14-bit delta-sigma ADC (analog to digital converter), performing internal cell balancing of up to 10mA with 10mV resolution, programmable overage and underage protection, and input power failure detection. Programmable protection and charging parameters as well as ADC measurements are communicated with the microcontroller over an I2C bus [3].

### 2.2.3 5V Regulator

A buck converter is used to step down the 13.5V battery supply to a 5V voltage rail that powers several IC components on the board. A buck converter was chosen because of the wide input voltage range and higher efficiency at large voltage drops from supply to output [4]. This prevents large amounts of power losses and heat dissipation into the board that could cause improper functionality. This 5V regulator will directly power the 3.3V regulator, CAN Transceiver, and USB-UART modules. The regulator will typically output about 300mA during normal operation. Accounting for worst case current draw and a safety factor, the 5V regulator must output a maximum of 500mA.

### 2.2.4 3.3V Regulator

As mentioned above, this 3.3V supply originates from the 5V regulator which then powers most of components throughout the system. Unlike the 5V regulator, this regulator is a low-dropout linear regulator. Because the drop from 5V to 3.3V is fairly small, the efficiency of the linear regulator is reasonable and, therefore, it doesn’t dissipate much power. Although the efficiency is at a reasonable level, power dissipation from a linear regulator should never be ignored. The power dissipation calculation can be given in Eq. 3.

(Eq. 3)

Even though power dissipation must be considered, a linear regulator is still beneficial due to its simplicity, size, and cleaner outputs when compared to a switching regulator [6].  As given in the TPS73633 datasheet [7], the output noise equation (Eq. 4) is given to ensure output voltage tolerances are met.

(Eq. 4)

Since the 3.3V regulator powers about two thirds of the components, the necessary max output current can be determined to be 350mA derived from the 500mA requirement of the 5V regulator. From the tolerances on both 3.3V and 5V regulators, the minimum dropout voltage for the linear regulator can be found in Eq. 5.

(Eq. 5)

## 2.3 Control Unit

The control unit for this system manages the incoming CAN messages and stores the data into memory through the use of a microcontroller and CAN transceiver. When storing data into memory, a time stamp from a real-time clock is stored with the data for later correlation with different test events. The microcontroller then transfers the data through Bluetooth and USB for user communication.

### 2.3.1 Microcontroller

The PIC32 microcontroller (PIC32MZ2048EFM100) is capable of receiving communications from various peripheral modules using several different communications protocols. It receives messages from the CAN transceiver and stores this into an SD card through SPI communication [8]. The microcontroller sends the data located in the SD card to the Bluetooth IC through a separate SPI bus and the USB-UART IC through UART. The microcontroller is also responsible for communicating with the super capacitor controller to set protection limits and to monitor and make decisions if power loss or overvoltage occurs using I2C [3].

### 2.3.2 CAN Transceiver

The CAN transceiver is a necessary component because the PIC32 is not compatible with CAN’s 5V differential signals. The MCP2562 will read all the high-powered CAN messages coming from throughout the car via the CAN bus harness and produce low-powered 3.3V Tx/Rx signals that are readable by the microcontroller [9]. Since the CAN transceiver is directly connected to an outside wire harness, the IC must have ESD protection. The main ESD concern stems from inductive spikes from long wires. ESD is a very difficult parameter to predict, but an ESD rating of +/- 2kV is a safe number to expect from the wire harness.

### 2.3.3 SD Card

Some form of non-volatile memory is necessary to hold all the data that the microcontroller and CAN transceiver process. For the system, an SD card was chosen due to its storage capabilities, size/weight, straightforward SPI interface, and speed capabilities. The SD card is capable of being read or written to at speeds of at least 12.5MB/s, which is significantly faster than any of the communication ICs are capable of. The car runs a 1 Mbps CAN bus, a 12 Mbps USB-UART chip, and a 2Mbps Bluetooth chip, so any SD card is more than fast enough. Assuming 8-hour testing sessions (3600 seconds) without pulling data, the SD card must have minimum storage of 3.6 GB.

### 2.3.4 Temperature Sensor

The only sensor included in this circuit is the temperature sensor. This will communicate with the microcontroller for circuit diagnostic purposes. The logger will likely be placed in a high-temperature environment near the fuel tank of the car, and this sensor will relay the temperature of the board to ensure that all chips are still within their temperature operating range. Since the board will be placed near high temperature environments, a max temperature rating of 80 ̊C must be met. Also, driving conditions can get as low as -10 ̊C, so that will be the lower bound for the temperature rating. The chosen chip, the MAX6610, can read temperatures between -40 ̊C and 125 ̊C. It produces an analog output that is read by the microcontroller’s internal analog-to-digital converter module.

### 2.3.5 Real-Time Clock

The real-time clock (RTC) lets the microcontroller know what the date and the time is so that the logged data can be paired with a date and time. This will make it convenient for the team because the time and date can be a way to indicate which tests were being run. The IC must keep track of the time even when the board is off so that he time doesn’t get interrupted. Because of this, the chosen IC (MCP79522) will be powered by a 3V coin cell battery. The particular battery is the CR1220 which has a capacity of ≈37mAh [17]. With the IC having a max current consumption of 1μA [11], the time that the IC can be powered is given in Eq. 6.

(Eq. 6)

## 2.4 User Interface

The user interface module makes it possible for any user to read the data that has been stored in the SD card. There are two methods to communicate the data: USB and Bluetooth. The Bluetooth IC is useful for smaller data sets between test runs. The USB interface is used for transferring large sets of data from longer testing sessions because it has a significantly faster data rate than the Bluetooth IC.

### 2.4.1 USB-UART

The FT230X IC communicates with the microcontroller through UART to transfer data stored in the SD card to the user. The IC will then convert these messages to the standard USB protocol so that the data can easily be transferred to a laptop with a simple cable at speeds of up to 3 MBps.

### 2.4.2 Bluetooth

The Bluetooth IC receives 2 Mbps SPI signals containing relevant data from the microcontroller (originating from the SD card) and converts them to the necessary signals for the antenna to be able to transmit the data wirelessly to an external device. The Nordic Semiconductor NRF51822 was chosen as the Bluetooth IC for this system because it meets all the requirements and has a plethora of additional capabilities and GPIO pins that can be used if necessary. It also provides a recommended impedance matching network to simplify the design process. Some additional capabilities include temperature sensor, 10-bit ADC, and ARM processor. It even has a low-powered off mode that can be used to reduce power consumption while data is not being transmitted [13]. Nordic also provides a software development tools for its Bluetooth devices that facilitates implementation of the BLE stack firmware. [14]

### 2.4.3 Antenna and Impedance Matching

The antenna transmits logged data from the SD card to the user through Bluetooth. It must transmit in the 2.4 GHz to 2.484 GHz band with a signal strength capable of being recognized at a range of 10m. An inverted-F quarter wave trace antenna will be used to minimize power loss and simplify impedance matching. [15] It will be connected to the main PCB by a coaxial cable. The impedance matching scheme must match the optimal input impedance of the Bluetooth IC with that of the antenna.

The antenna was originally planned to be part of this project, but due to the difficulties involved in designing an effective PCB trace antenna, the antenna portion of the design was dropped. An external antenna was purchased instead.

## 2.5 Software

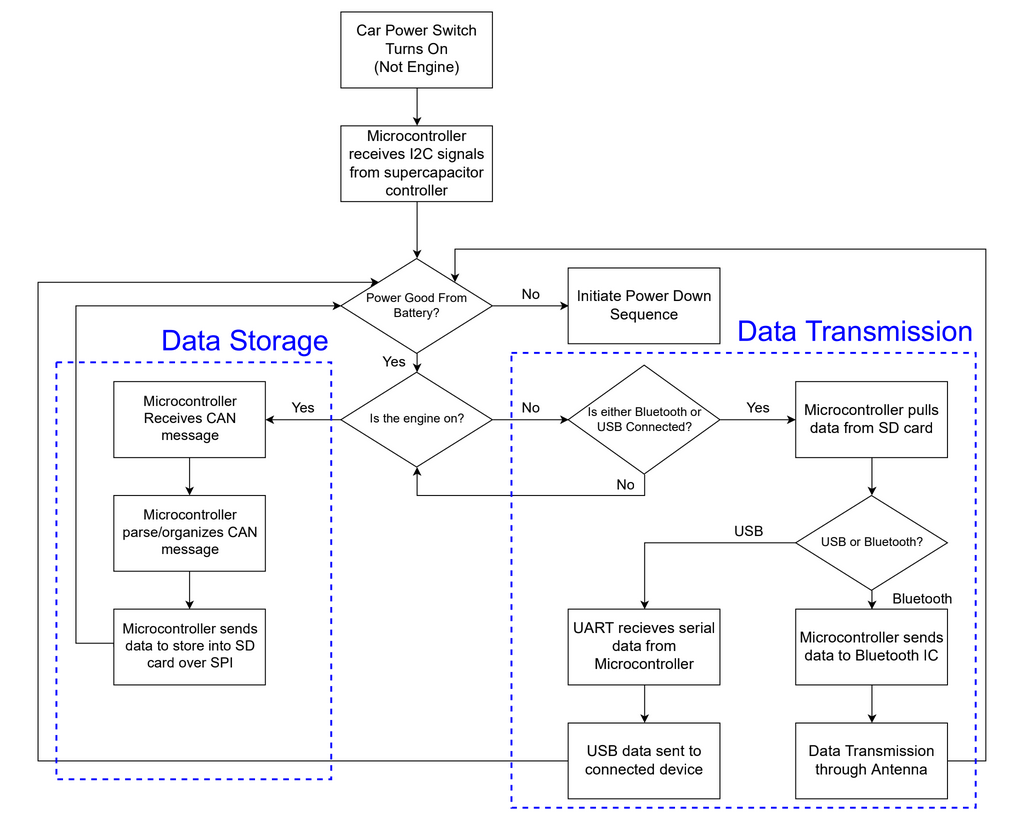
Because there is a lot of data processing in this design, it is inherent that there is a large software portion to the system. All of the software and brains of the system lies within the microcontroller. The developed code will be stored into the microcontrollers 2MB of flash memory [8]. From a high level, the microcontroller has to parse through all the CAN messages whenever there is a CAN interrupt. An interrupt happens whenever a new CAN message is detected. The microcontroller will then write the content of CAN message to the SD card through SPI with the addition of the time stamp acquired from the RTC. Whenever a read is required either through USB or Bluetooth, the microcontroller will send the SD card a read command. The USB or Bluetooth module will then transmit CAN data to the user.

From there, software will be developed on the laptop to easily parse the raw data and format it in a way that is easily readable. Because the software is not uploaded to the microcontroller, this portion is not considered to be part of the logger design, but it is still important to understand. The team has a CAN specification which will act as a template for the desired data format. The CAN specification has an allocated a list of IDs. Within each ID, there are multiple data points because each data point is typically two bytes. Each byte in the ID is also assigned to a specific data point that is relevant to the respective ID. Table 1 shows an example of the structure of one of the team’s CAN Spec IDs (Motec is the team’s engine control unit):

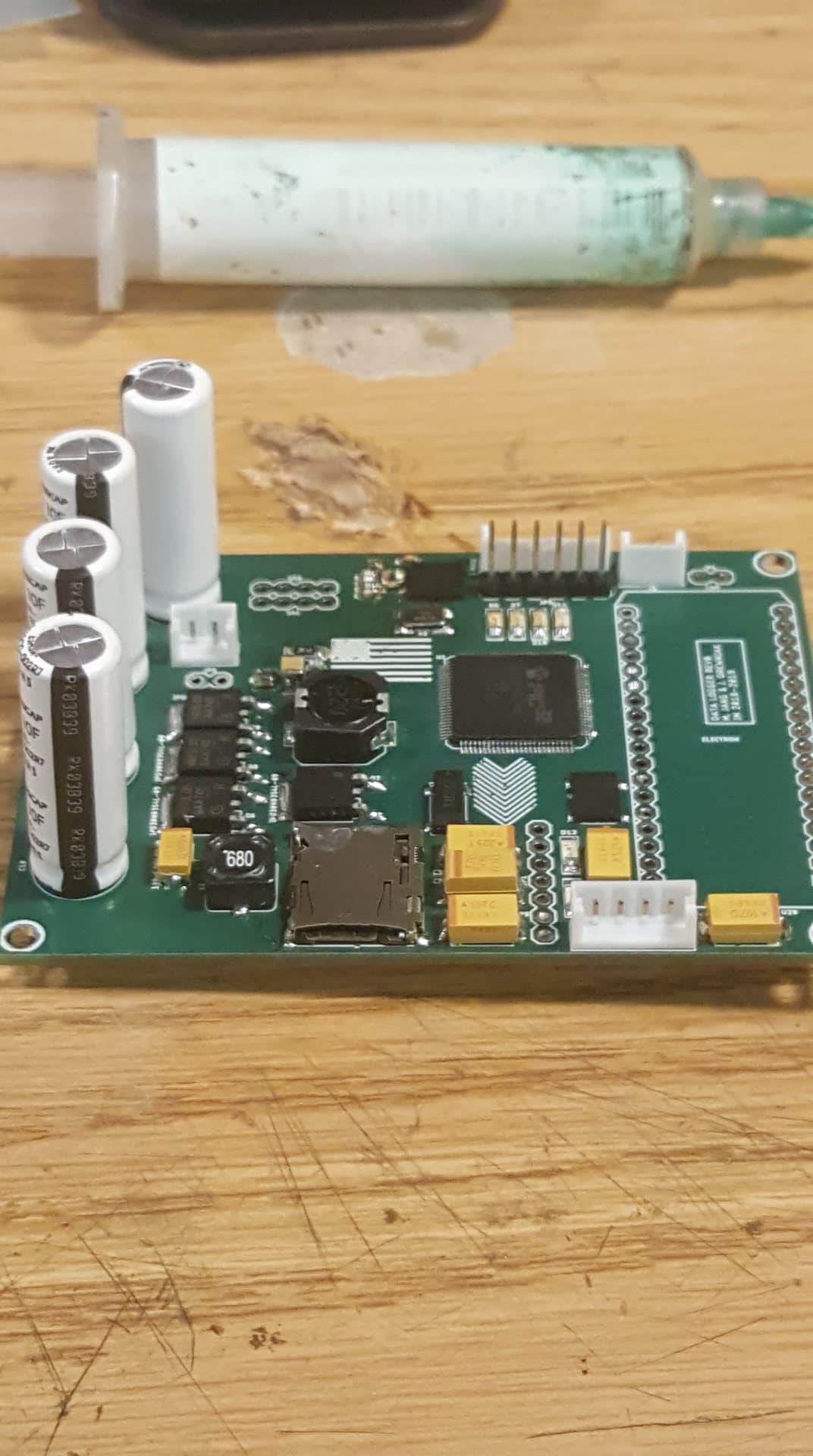
|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Origin** | **ID** | **Byte0** | **Byte1** | **Byte2** | **Byte3** | **Byte4** | **Byte5** | **Byte6** | **Byte7** |
| Motec | 0x100 | RPM | | Throttle Position | | Lambda | | Battery Voltage | |

**Table 1 Example of Illini Motorsports CAN Spec ID**

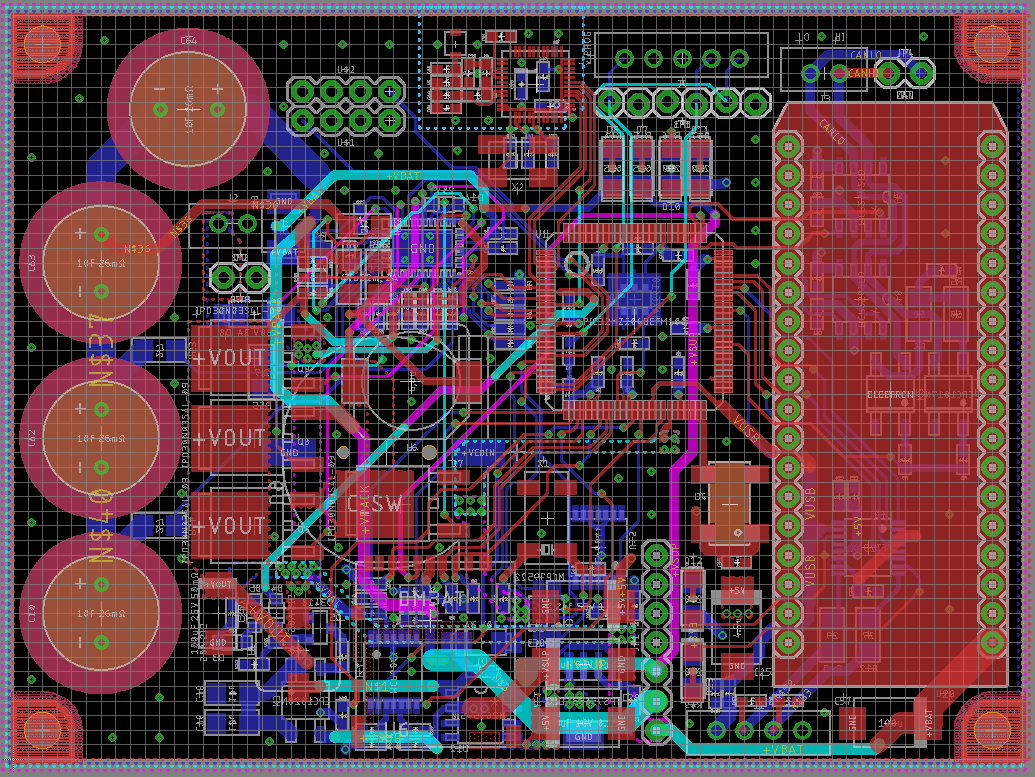
Figure 2 displays the desired software architecture. It details anticipated conditions seen by the logger under normal operating conditions and determines the correct course of action. It elucidates when data storage, Bluetooth, or USB transmission are used, as well as the correct power down conditions of the board.



**Figure 2 Software Flow Chart**



**Figure 3 Data Logger PCB**



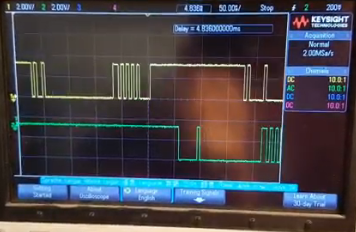
**Figure 4 Data Logger PCB Layout**

# 3. Design Verification

## 3.1 SD Card Interface

To verify the SPI communication interface with the SD card, the SPI data lines were probed using an oscilloscope. SD cards have very specific protocols for initialization and data transfer commands, so monitoring them enabled verification of correct command sequences. As seen in Figure 4, the yellow lines are the commands being sent by the microcontroller, and the green lines denote the response from the SD card.

**Figure 5 SD Card SPI Communication**



Once correct SPI communication was verified, a spoofed CAN message was sent to the SD card. Comparing the scope trace to the raw data that was stored on the SD card ascertained that the write functionality was working properly. Although the read functionality was written in software, it was never tested.

## 3.2 USB Interface

The USB interface was tested in two phases. First, the FT230X chip has an output that powers an external LED when transmissions are being sent or received. Once basic transmission functionality was achieved by monitoring the LED, virtual COM port software was used to read the raw data being sent from the board, as seen in Figure 6.



**Figure 6 USB Communication with Laptop**

An engine temperature message was sent to verify that everything was working properly, and the results were quite successful.

## 3.3 CAN Interface

The CAN interface is something that the FSAE team has implemented in the past, so thorough testing was not implemented. The board was plugged into the car with a breakpoint in the CAN interrupt code, and gear position was sent out over CAN and set to a variable in the code. The value verified during the debugging process. Correct functionality was achieved.

## 3.4 Supercapacitor Power

Testing the supercapacitor circuitry was extremely simple. The controller came with default settings that fulfilled the requirements, so no software needed to be implemented. LEDs are connected to the power input and voltage regulators on the board for debugging purposes, so merely disconnecting the power supply and timing how long the LEDs stayed on was enough to validate the design. They stayed lit for well over a minute. An oscilloscope was also connected to the supercapacitor outputs to verify the negative linear discharge that was expected.

Unfortunately, as mentioned in more detail in Section 5, an accident occurred before any documentation was taken, so there is no proof that it was working at one point. This also prevented the I2C bit-bang code that was written from being fully tested.

## 3.5 Peripheral Circuitry

Because so much time was spent on the above software, communication with the temperature sensor and the real-time clock was never achieved.

# 4. Costs

## 4.1 Parts

|  |  |  |  |
| --- | --- | --- | --- |
| **Part** | **Manufacturer** | **Unit Cost** | **Quantity** |
| PIC32MZ2048EFM100 | Microchip | $12.05 | 1 |
| MCP2562FD | Microchip | $1.04 | 1 |
| LTC3350 | Linear | $11.18 | 1 |
| 10F 26mOhm EDLC Capacitor | NessCap | $3.72 | 4 |
| SD Card Socket | Hirose Electric | $1.88 | 1 |
| ASVMB-24.000MHZ-XY-T | Abracon | $2.19 | 1 |
| Electron Device | Particle.io | $69.00 | [1](http://particle.io/) |
| MCP2562FD | Microchip | $1.04 | 1 |
| MCP79522 | Microchip | $1.28 | 1 |
| WRL-00691 | SparkFun | $19.95 | 1 |
| MCP6044 | Microchip | $1.41 | 2 |
| USB to UART | FTDI | $2.38 | 1 |
| ABS25-32.768KHZ-T | Abracon | $0.35 | 1 |
| MAX6610 | Maxim | $2.24 | 1 |
| LM2675-5.0 | TI | $4.37 | 1 |
| TPS73633DCQR | TI | $2.33 | 1 |
| 1Mb EEPROM | Microchip | $2.94 | 1 |
| 18 Position Female Headers | Sullins Connector | $1.43 | 2 |
| 8 Position Female Headers | Sullins Connector | $0.89 | 1 |
| 8 Position Right-Angle Headers | Sullins Connector | $0.38 | 1 |
| B5B-EH-A | JST | $0.24 | 1 |
| B4B-EH-A | JST | $0.20 | 1 |
| B2B-EH-A | JST | $0.14 | 2 |
| 6-pin Header | Molex | $2.17 | 1 |
| IPD30N03S4L-09 | Infineon | $0.65 | 4 |
| AO3418 | Alpha & Omega | $0.44 | 1 |
| CPC1017N | IXYS | $1.08 | 2 |
| CPC1020N | IXYS | $3.20 | 1 |
| 3.3V Zener Diode | NXP Semiconductors | $0.36 | 1 |
| 30V Shottky Diode | Diodes Incorperated | $0.51 | 1 |
| 20V 1A Diode | Vishay | $0.38 | 1 |
| 16V TVS Diode | ON Semiconductor | $0.47 | 1 |
| 75V Signal Diode | Fairchild | $0.13 | 3 |
| 30V Shottky Diode | Diodes Incorperated | $0.51 | 1 |
| LTST-C150GKT | Lite-On | $0.30 | 6 |
| 22u 3A Inductor | Wurth Electronics | $2.76 | 1 |
| 68uH Inductor | Wurth | $2.52 | 1 |
| All Resistors | Yageo | $8.52 | 1 |
| All Ceramic Capacitors | Kemet | $18.92 | 1 |
| All Tantalum Capacitors | AVX | $16.26 | 1 |
| **Total** | **$220.74** | | |

**Table 2 Parts BOM**

## 4.2 Labor

Assuming an estimated starting salary of ≈$75,000, that translates to $36.00/hour given the Eq. 7.

(Eq. 7)

Taking into account the work put into the project this semester, approximately an average 15 hours were spent each week for 16 weeks. The labor cost for the semester is calculated in Eq. 8.

(Eq. 8)

There is much more work needed to be done, roughly 75% of the project is done with most of the remaining work including software development. The total labor cost of the entire project is estimated in Eq. 9.

(Eq. 9)

Adding the labor cost and the total parts cost from Table 2, the total cost yields:

# 5. Conclusion

## 5.1 Accomplishments

In the end, the CAN transceiver is able to input external CAN messages and convert/relay the message to the microcontroller. The microcontroller can also read and write this data to the SD card. The microcontroller is capable of successfully transferring CAN data over UART to the USB-UART conversion chip, which ultimately sends the data to the user’s laptop over USB. The supercapacitor circuitry was also successfully implemented with the default controller settings, but communication was never implemented due to some errata in the microcontroller.

## 5.2 Shortcomings

All requirements regarding the Bluetooth and antenna module have not been met because headers were not added for programming the Bluetooth IC. It was thought that the IC could be programmed by the PIC32 microcontroller over SPI. In actuality, the IC needs to be programmed externally from through separate software. Unfortunately, no solution could be achieved because the IC was a QFN package that lacked exposed pads to solder wires to.

The supercapacitors no longer power the board for over 2 seconds because of an accident in which the board was shorted. The short originated when an alligator clip powering the board slipped and contacted several components in the controller circuitry. That IC and its supporting circuitry was determined to be no longer functional. In an attempt to salvage the supercapacitor functionality, a protection MOSFET was disconnected and the drain was shorted with the source, ultimately shorting the capacitors to the power supply and the rest of the board’s circuitry. The capacitors promptly sunk about 3A of current into the rest of the board which caused many other components of the system to malfunction, making the situation significantly worse.

## 5.3 Ethical considerations

The Lithium-Ion car battery is extremely dangerous if the terminals are shorted together. Extreme caution is needed when handling the battery near metal tables or anything capable of being a conductor. Lithium-Ion batteries can also explode if overcharged or if they are exposed to temperatures outside the recommended operating conditions. The team ensures that the battery is well ventilated and never excessively charged under IEEE Code of Ethics #1 (safe, healthy, and environmentally safe) [12].

The purpose of this project is to design a faster, lighter, and safer race car by improving the usage of sensor data. This data can be used to validate mathematical and computational models or provide relevant numbers where models do not exist. This is in direct compliance with IEEE Code of Ethics #5 (to improve understanding through technology) [12]. To reduce the chances of violating IEEE Code of Ethics #3 (to be and realistic with data interpretation), the team always tries to validate all data through both basic hand calculations and complex computational models [12]. If the numbers vastly disagree, we expend every effort to discover why.

## 5.4 Future work

The next steps in the design process are to redesign the PCB slightly to incorporate the programming headers for the Bluetooth IC and develop the external software to parse and organize the raw CAN data. Code also needs to be written to incorporate the temperature sensor and real-time clock into the design. The enclosure for the board will need to be designed and 3D printed in the near future as well.

# References

[1] Kvaser, ‘CAN Protocol Tutorial’, 2018. [Online]. Available: <https://www.kvaser.com/can-protocol-tutorial/>

[2] R. Roderick, ‘A Look Inside Battery-Management Systems’, 2015. [Online]. Available: <https://www.electronicdesign.com/power/look-inside-battery-management-systems>

[3] Linear Technology, ‘High Current Supercapacitor Backup Controller and System Monitor’, 2014. [Online]. Available: [www.analog.com/media/en/technical-documentation/data-sheets/3350fc.pdf](http://www.analog.com/media/en/technical-documentation/data-sheets/3350fc.pdf)

[4] Renesas, ‘Switching Regulators’, 2018, [Online]. Available: <https://www.renesas.com/us/en/products/power-management/switching-regulators.html>

[5] Texas Instruments, ‘LM2675 SIMPLE SWITCHER® Power Converter High Efficiency 1-A Step-Down Voltage Regulator’, 2016. [Online]. Available: <http://www.ti.com/general/docs/lit/getliterature.tsp?genericPartNumber=lm2675&fileType=pdf>

[6] D. Knight, ‘Introduction to Linear Voltage Regulators’, 2016. [Online]. Available: <https://www.digikey.com/en/maker/blogs/introduction-to-linear-voltage-regulators>

[7] Texas Instruments, ‘CAP-FREE NMOS 400 mA LOW-DROPOUT REGULATORS WITH REVERSE CURRENT PROTECTION’, 2014. [Online]. Available: <http://www.ti.com/general/docs/lit/getliterature.tsp?genericPartNumber=tps73633-ep&fileType=pdf>

[8] Microchip Technology, ‘PIC32MZ Embedded Connectivity with Floating Point Unit (EF) Family’, 2018. [Online]. Available: [www.microchip.com/mymicrochip/filehandler.aspx?ddocname=en574992](http://www.microchip.com/mymicrochip/filehandler.aspx?ddocname=en574992)

[9] Microchip Technology, ‘High-Speed CAN Transceiver’, 2014. [Online]. Available: [www.microchip.com/mymicrochip/filehandler.aspx?ddocname=en561044](http://www.microchip.com/mymicrochip/filehandler.aspx?ddocname=en561044)

[10] Maxim Integrated, ‘Precision, Low-Power, 6-Pin SOT23 Temperature Sensors and Voltage References’, 2003. [Online]. Available: <https://datasheets.maximintegrated.com/en/ds/MAX6610-MAX6611.pdf>

[11] Microchip Technology, ‘Battery-Backed SPI Real-Time Clock/Calendar’, 2018, [Online]. Available: [www.microchip.com/mymicrochip/filehandler.aspx?ddocname=en557883](http://www.microchip.com/mymicrochip/filehandler.aspx?ddocname=en557883)

[12] Future Technology Devices International, ‘FT230X  (USB to BASIC UART IC)’, 2016. [Online]. Available: [www.ftdichip.com/Support/Documents/DataSheets/ICs/DS\_FT230X.pdf](http://www.ftdichip.com/Support/Documents/DataSheets/ICs/DS_FT230X.pdf)

[13] Nordic Semiconductor, ‘nRF51822: Multiprotocol Bluetooth® low energy/2.4 GHz RF System on Chip’, 2018, [Online]. Available: <http://www.nordicsemi.com/eng/content/download/13358/214991/file/nRF51822_PS%20v3.1.pdf>

[14] Nordic Semiconductor, ‘nRF51 Series: SoftDevices’, 2018, [Online]. Available:<http://infocenter.nordicsemi.com/index.jsp?topic=%2Fcom.nordic.infocenter.s140.sds%2Fdita%2Fsoftdevices%2Fs130%2Fble_protocol_stack%2Fble_protocol_stack.html>

[15] Antenna Theory, ‘Inverted-F Antenna (IFA)’, 2018,[Online]. Available: <http://www.antenna-theory.com/antennas/aperture/ifa.php>

[16] IEEE, ‘IEEE Code of Ethics’, 2018. [Online]. Available: <https://www.ieee.org/about/corporate/governance/p7-8.html>

[17] Power Glory Battery Tech Co., ‘Specifications For Lithium Battery Model: CR1220’, 2011. [Online]. Available: <https://www.google.com/url?sa=t&rct=j&q=&esrc=s&source=web&cd=2&ved=2ahUKEwiEg_T7pJnfAhXC44MKHZkAAVMQFjABegQIAxAC&url=http%3A%2F%2Fwww.farnell.com%2Fdatasheets%2F1496882.pdf&usg=AOvVaw3_jDqpVJESAh4Ju76CVYHb>

# Appendix A Requirement and Verification Table

See section 5.2 for unmet requirements

|  |  |  |
| --- | --- | --- |
|  | |  |
| Requirement | Verification | Verification status  (Y or N) |
| 1. Supply a DC 5V and 3.3V +/- 5% to the board | 1. Oscilloscope to measure output voltages | Y |
| 1. Pack voltage of 10.8V-12V in the supercapacitors to sufficiently power the board if power is lost | 1. Oscilloscope to measure voltage across capacitors | Y |
| 1. Supercapacitor powers the 5V Regulator for at least 2 seconds at a max current consumption | 1. On board LEDs indicate 5V and 3.3V status. Disconnect power from board and see how long it takes for LEDs to turn off | N |

**Table 3 Power Supply Requirements and Verifications**

|  |  |  |
| --- | --- | --- |
|  | |  |
| Requirement | Verification | Verification status  (Y, N, or Partial) |
| 1. Must input CAN protocol at a minimum rate of 1 Mbps | 1. Send known CAN signals to the logger. Read directly from the SD card and compare data | Y |
| 1. Convert and store all CAN messages into an SD card (3.6 GB minimum) through SPI. | 1. See #1 | Y |
| 1. The SD card must send the stored data out through USB and Bluetooth for the data to be externally read and processed. | 1. See Table 5 Verification #1 | P |

**Table 4 Control Unit Requirements and Verifications**

|  |  |  |
| --- | --- | --- |
|  | |  |
| Requirement | Verification | Verification status  (Y, N, or Partial) |
| 1. USB-UART and Bluetooth ICs must properly convert UART/SPI signals to the USB and Bluetooth protocol (respectively) while maintaining the integrity of the data. | 1. Send known CAN signals to the logger. Read data from USB and Bluetooth and compare the data. | P |
| 1. The Bluetooth IC must also relay the data to the antenna for wireless communication | 1. Transmit Bluetooth data to laptop | N |
| 1. Antenna must have data transfer range of about 10m | 1. Transmit Bluetooth data 10m away from the board | N |

**Table 5 User Interface Requirements and Verifications**