TELEMETRY SYSTEM ARCHITECTURE FOR A SOLAR CAR

Eric Walter, Nicholas Glover, Jesse Cureton and Kurt Kosbar
Telemetry Learning Center
Department of Electrical and Computer Engineering
Missouri University of Science and Technology
Rolla, MO, USA

ABSTRACT

This paper discusses the telemetry system used to monitor the performance of a solar-powered vehicle during testing and competitions. Car-side systems collect and transmit data onboard over an ISO 11898 / CAN bus. A bridge then converts this data into TCP/IP packets, which are transmitted via Ethernet to a Wi-Fi access point. The data is distributed through an IEEE 802.11N 5GHz mesh network to provide real time data to remote computers running telemetry software. This software displays and logs data from the car, allowing team members to monitor the vehicle.

Keywords: Telemetry System Design, CAN Bus, Wi-Fi, COTS, Solar Vehicles

1. INTRODUCTION

A practical means of transportation that utilizes renewable energy is a key part of our ability to decrease our environmental footprint. The sport of solar car racing has been pushing the development of solar technology and its applications since the 1980's. As with any vehicle, there are specific challenges with regard to monitoring the vehicle's condition. The Missouri S&T Solar Car team has implemented a custom telemetry system in the team's vehicle. The primary objective of this system is to enable real time feedback from the car's electrical and mechanical systems during operation to allow real-time strategy adjustments based on weather conditions, driver status, and vehicle status.

This system can be viewed primarily as three separate systems: an on-vehicle layer that consists of a controller area network bus that allows systems on the car to communicate, a network layer that consists of an IEEE 802.11N wireless mesh network that allows data transmission off the vehicle, and a software layer which implements the backend to view transmitted data. This system is entirely modular, allowing the team to update any individual system isolated from the rest.

The primary design goal with the system was to implement it using industry standard communication protocols. This allows team members to gather important experience in the tools they will use in the field, as well allow team members with this experience to immediately begin working on the system with a minimal learning curve.

2. VEHICLE DETAILS

An overview of the vehicle's electrical system can be seen in Figure 1. The heart of the vehicle is a 20kg lithium polymer battery pack consisting of 25 series packs of 5 parallel Dow Kokam 8Ah

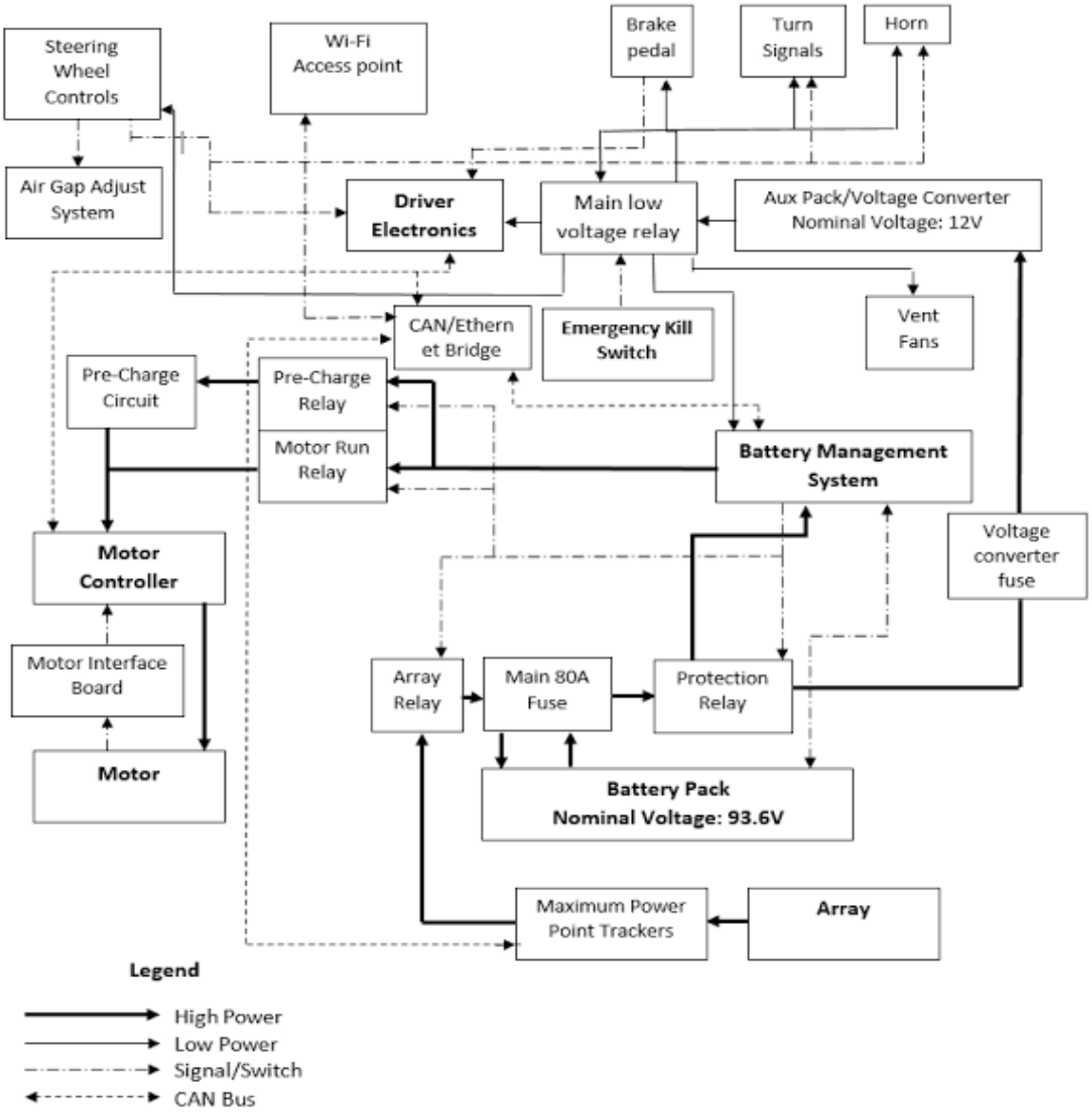

Figure 1 - Electrical Subsystem Diagram

cells. The nominal voltage of the pack is 93.6V, which with the individual pack capacity of 40Ah gives a nominal whole-pack capacity of 3.7kWh. Monitoring this pack is a commercial battery management system. The system monitors all of the cells in the pack, implementing cell balancing, measurement of cell voltages, temperatures, pack current, and pack isolation in the event of a fault.

The upper portion of the car consists of a 6 m^2 array of monocrystalline silicon-dioxide solar cells, with a nominal power output of 1200W. This array is regulated by maximum power point trackers (MPPTs) that regulate the voltage of the array in real time to provide efficiency gains in the charging circuit.

Driving the car is a 7.5kW brushless permanent magnet motor, powered by a high efficiency (>99.2%) three-phase power inverter and motor controller.

3. PHYSICAL LAYER IMPLEMENTATION

Figure 2 details the physical layer of the vehicle. The system can be broke down into two main sections: the physical layer on the vehicle (the Vehicle Layer), and the networking layer that controls data transmission to the rest of the team (the Network Layer).

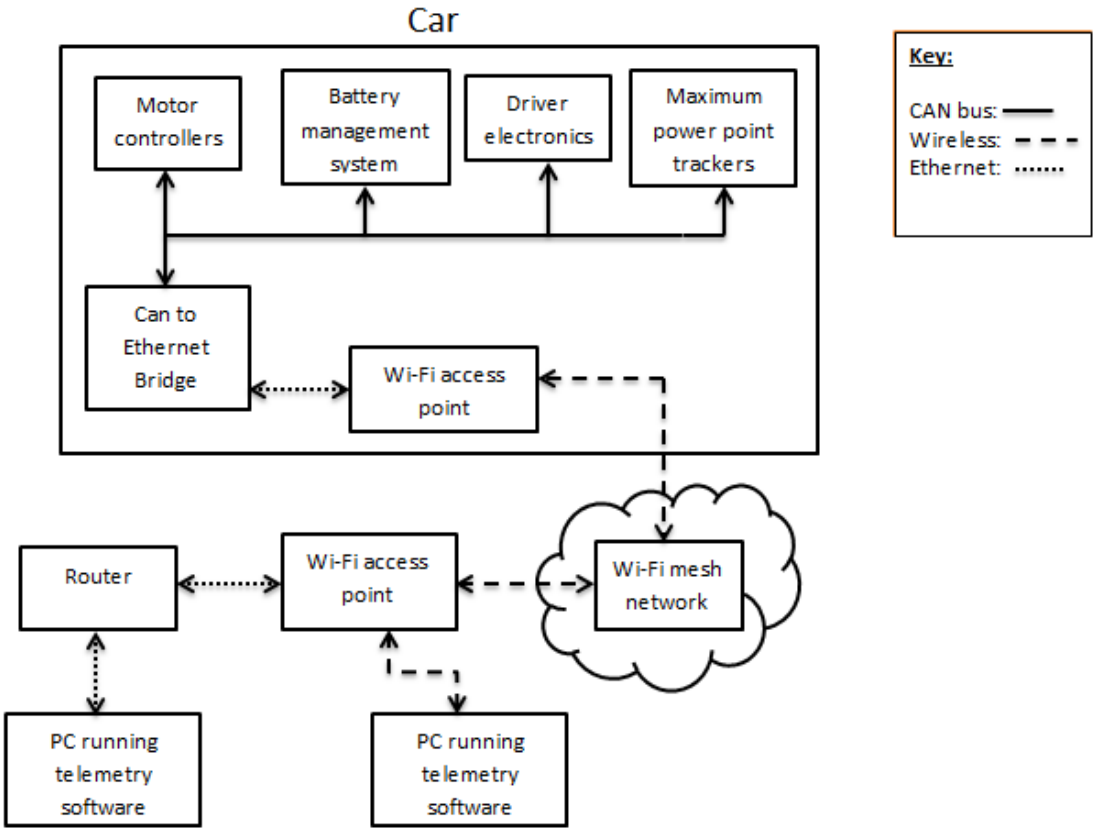

Figure 2 - Physical Layer Implementation

A. THE VEHICLE LAYER

The Vehicle Layer consists of a Controller Area Network (CAN) bus which implements CAN 2.0B [1]. This layer consists of all of the CAN devices on the vehicle: the motor controller, battery management system, driver electronics input processing hardware, and MPPTs. The final device on this bus is a CAN to Ethernet bridge. It converts each CAN message packet on the network into a User Datagram Protocol (UDP) [2] broadcast packet that is sent out over Ethernet. This UDP packet is the link between the physical Vehicle and Network layers and the software layer detailed in Section 4 – Software Layer Implementation.

Devices on the CAN network serve a variety of functions. These can be broken down as follows in Table 1.

Table 1 - CAN Device Functionality

Device	*CAN Function*
Motor Controller	Receives and parses CAN messages transmitted by the driver electronics system to control the motor state. Also transmits detailed telemetry data about the status of the motor.
Battery Management System	Implements the protection functionality detailed in Section 2 – Vehicle Details. Transmits detailed telemetry data regarding the status of both the entire battery pack and of individual cells in the pack.
Driver Electronics	Processes information from the vehicle's driver compartment and the controls therein. Uses this data to transmit CAN messages to the motor controller
Maximum Power Point Trackers	Transmits telemetry data about the state of the array

B. THE NETWORK LAYER

At the core of the network layer is an IEEE 802.11N [3] wireless network with full TCP/IP capabilities. The network runs on a 40MHz block of spectrum in the 5GHz band, and transmits with 1W of transmit power at each node. The number of nodes in the network varies depending on the situation, but is typically either three nodes when traveling in a convoy, or up to six when on a track.

When configured for traveling in a convoy, there is one node on the solar vehicle, along with one node each for a lead and chase vehicle that always travels with the solar vehicle. This ensures that anywhere in the convoy anyone running the software can view data coming from the vehicle. Inside of the chase vehicle there is a computer that is running the telemetry software layer full time

logging data from the vehicle and reading data off to the team. A typical track configuration would consist of multiple access points positioned around the track to ensure consistent connection with the car.

At any one of the wireless access point nodes, a wired Ethernet connection can break out to allow more computers to connect. At one wireless node, a wired connection does break out and connect to a router. This is the DHCP server for the network, ensuring that all TCP/IP devices receive an IP address and can function properly.

4. SOFTWARE LAYER IMPLEMENTATION

As mentioned in Section 3.1, the link between the software layer and the physical layer is UDP broadcasting. UDP broadcasts relieves strain on the network because it doesn't necessitate individual TCP connections between nodes. Instead, the CAN to Ethernet bridge on the Vehicle Layer broadcasts individual UDP packets, which can then be received by any node on the network which is "listening" for those packets.

A. SOFTWARE ARCHITECTURE

Due to the diversity of team development environments, wide platform needs, and familiarity with the language, the telemetry software is written in Java. This allows the software to run on the Linux based embedded system in the driver display, the Windows based PCs in the convoy, and the Mac computers used by many team developers.

Java is well-suited to this use. It is perhaps the easiest solution for cross-platform UI development [4], it has wide library support that allows the team potential to upgrade the software with additional features easily, and is a common, widely known language the team had experience with in the development cycle.

The most basic part of the software is its ability to parse the CAN messages that are being broadcast over UDP from the CAN-Ethernet bridge on the vehicle. The company which produces this bridge has made public a specification on this format [5], which is what was used in development of the team's software.

B. SOFTWARE DATA

The telemetry system displays every piece of data that is transmitted on the CAN bus on the vehicle. A full display can be seen in Figure 3. This includes:

- Battery management system flags
- Battery management faults & warnings
- Main pack voltage
- Individual minimum and maximum cell voltages
- Main pack net current
- Remaining pack capacity
- Average, minimum, and maximum cell temperatures
- Motor controller errors
- Distance traveled
- Speed
- Motor voltage/current
- Motor temperature
- Coulomb counting
- Individual sub-array currents and voltages

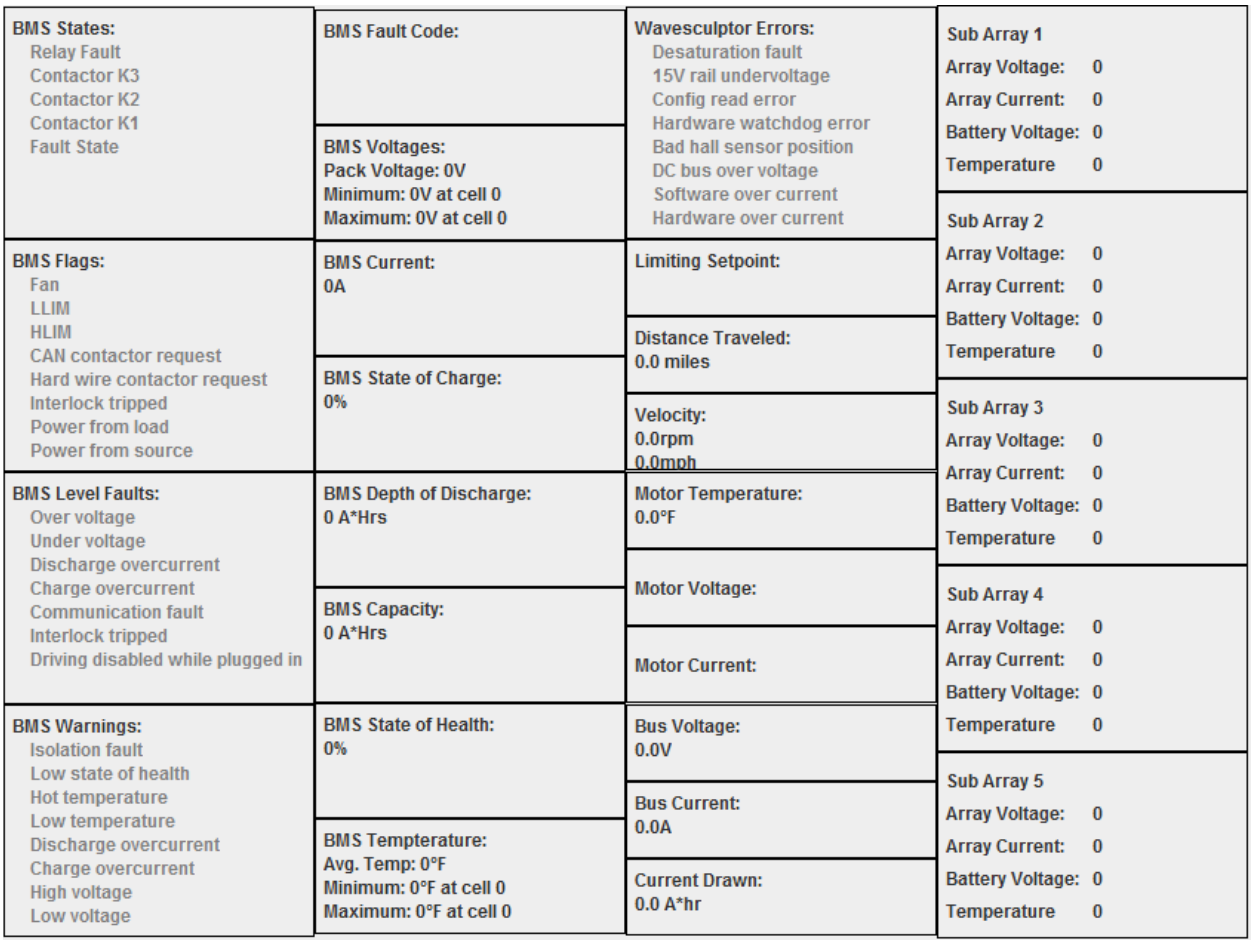

Figure 3 - Telemetry Display

C. SOFTWARE LOGGING

While real-time analysis of data is critical for the team to be able to function during a race, some of the largest insights come from more in depth analysis of data after the drive is finished. To that end, the telemetry software is capable of logging data that is received at set intervals. The team logs data any time the car is driving at 200ms intervals. The UI updates as packets are received, and continuous logging would provide the most accurate logs, however the team found the best balance between log size, enough resolution in the data, and programming convenience led to a rate of around 5Hz. Each entry printed in the log contains a timestamp, along with all of the data displayed in the UI at that time.

5. DATA ANALYSIS

As mentioned in Section 4.3, post-drive analysis provides perhaps the most valuable insights gathered in the team's telemetry system. The team can glean information much more in-depth than what can be seen in real-time; information like how different drivers handled a track can be compared, places where the vehicle struggled to get up a hill can be seen, and many more observations can be made. Figure 4 shows a graph of a drive around a parking lot at Missouri S&T. Between 5-7 minutes in the graph above, the speed can be seen varying almost like sinusoid. From this it can be inferred that this was during a drive on a looped circuit where the driver had to slow

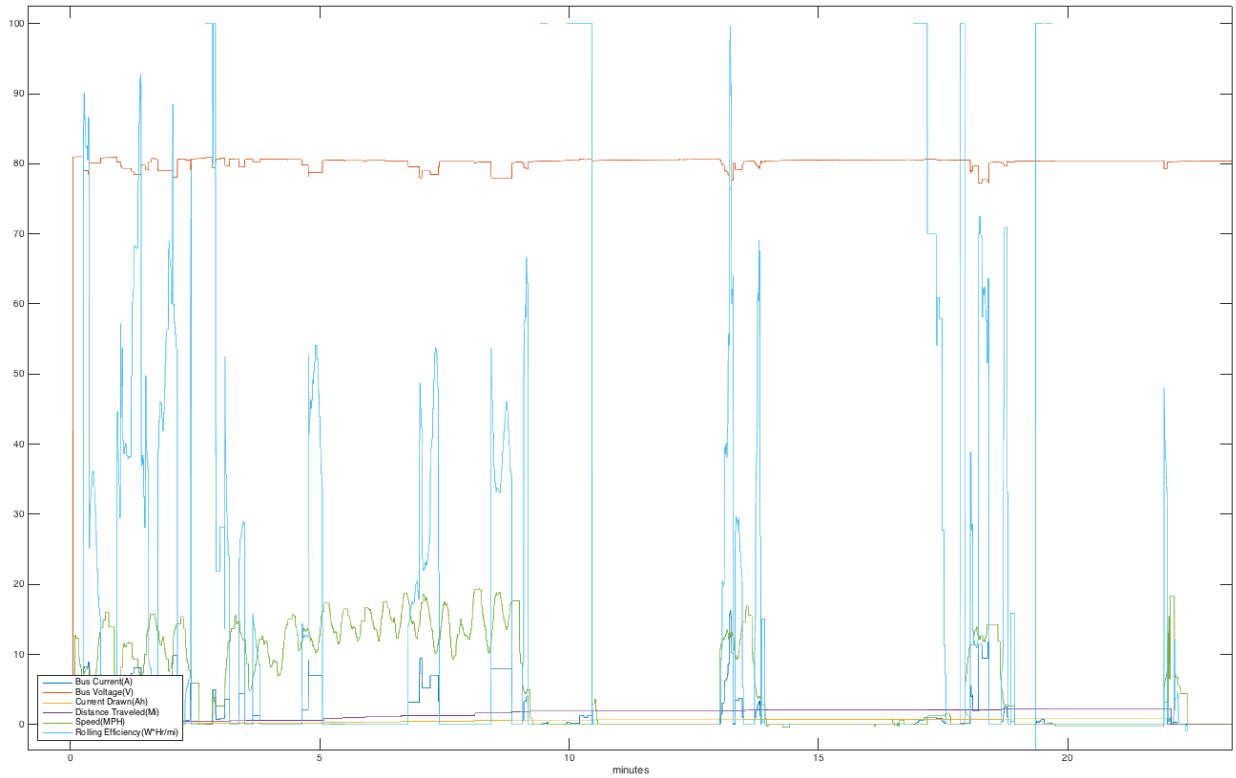

Figure 4 - Analysis Graph

significantly for the turns in the loop. This was in fact the case, where the circuit was a tight oval. While this is a simple example, this type of logging analysis can lead to much more in-depth analysis of more complicated courses and routes on longer drives. It can also be used to compare different drivers on one track, something that is immensely useful when the team finds itself with multiple drivers at every race.

6. CONCLUSION

The Missouri S&T Solar Car team has implemented a modular system to gather, transmit, and display data from its solar vehicle. These three layers allow a system that can be implemented on a stationary track over a very large area, or implemented on a moving vehicle convoy at high speeds.

The addition of telemetry software to the team's arsenal has been huge. It has turned from a team that was unable to analyze any statistics and merely guessing at efficiency and race strategy to a team that knows thoroughly how its car runs under many conditions. The team knows what drivers have what strengths, and can plan accordingly on how to place someone in the driver's seat. The benefits gained from this project have been more than worth the amount of time and team resources put into it.

As helpful as it has been, the team is still in the process of improving it. There are plans to add more real-time analysis to help bridge the gap between the insights of post-drive analysis and real-time data, along with plans to revamp the user interface of the software, and some changes at the physical layer to strengthen connections during certain scenarios.

7. ACKNOWLEDGEMENTS

Credit for the development and implementation of the electrical system is given to the Electrical Division of the Missouri S&T Solar Car Design Team. A special thanks is given to those who have supported them during the development of this system.

To Dr. Chris Ramsay, whose support of the team has helped it grow immensely. The team could not exist as it does today without his support.

To Mr. John Tyler, who has been with the team nearly as long as it has existed. His technical expertise in all things related to solar racing is second-to-none. He has been not only support, but a technical resource in every way possible.

8. REFERENCES

[1] International Organization for Standardization, "ISO 11898-1:2003," 2003. [Online]. Available: http://www.iso.org/iso/catalogue_detail.htm?csnumber=33422. [Accessed May 2015].

[2] IETF, "User Datagram Protocol," 28 August 1980. [Online]. Available: https://www.ietf.org/rfc/rfc768.txt. [Accessed May 2015].

[3] IEEE Standards Association, "802.11n-2009," 2009. [Online]. Available: https://standards.ieee.org/findstds/standard/802.11n-2009.html. [Accessed May 2015].

[4] Oracle, "About the JFC and Swing," [Online]. Available: http://docs.oracle.com/javase/tutorial/uiswing/start/about.html. [Accessed May 2015].

[5] Tritium Pty. Ltd., "tritium.com.au," 28 August 2011. [Online]. Available: http://tritium.com.au/wp-content/uploads/2012/07/TRI82.007v4_Ethernet_Interface.pdf. [Accessed May 2015].