View metadata, citation and similar papers at core.ac.uk
brought to you by CORE
provided by DigitalCommons@CalPoly

Nissan Leaf On-Board Diagnostic Bluetooth Utility

By

John Allande
Philip Tyler
Eric Woodruff

College of Engineering

Cal Polytechnic State University

San Luis Obispo

2013

Statement of Disclaimer

Since this project is a result of a class assignment, it has been graded and accepted as fulfillment of the course requirements. Acceptance does not imply technical accuracy or reliability. Any use of information in this report is done at the risk of the user. These risks may include catastrophic failure of the device or infringement of patent or copyright laws. California Polytechnic State University at San Luis Obispo and its staff cannot be held liable for any use or misuse of the project.

Executive Summary

The team's sponsor, John Dunning, owns an Electric Vehicle. The electric vehicle market is young, as is the research and testing on the cars themselves. Most EV owners know the most expensive component in their vehicles is the battery, and would like to know when the battery will fail. By experimenting with data from the OBD II port in Dr. Dunning's Nissan Leaf (2013 EV) the team will create a Android app to interface with a Bluetooth module to receive OBD data. The program will collect data focused on battery pack voltage, battery pack current, motor RPM, and state of charge to help John Dunning find trends in the battery over time. Hopefully, these trends can be used to estimate the life of his battery.

Table of Contents

List of Nomenclature

OBD - On-board diagnostics is an automotive term referring to a vehicle's self-diagnostic and reporting capability. OBD systems give the vehicle owner or a repair technician access to state of health information for various vehicle sub-systems.

Bluetooth - Bluetooth is a wireless technology standard for exchanging data over short distances (using short-wavelength radio transmissions in the ISM band from 2400–2480 MHz) from fixed and mobile devices, creating personal area networks (PANs) with high levels of security.

CAN Bus - CAN bus (for controller area network) is a vehicle bus standard designed to allow microcontrollers and devices to communicate with each other within a vehicle without a host computer.

Nissan Leaf -The Nissan Leaf (also formatted "LEAF" as a acronym for Leading, Environmentally friendly, Affordable, Family car) is a five-door hatchback electric car manufactured by Nissan and introduced in Japan and the United States in December 2010.

Electric Vehicle - An electric car is an automobile that is propelled by one electric motor or more, using electrical energy stored in batteries or another energy storage device. Electric motors give electric cars instant torque, creating strong and smooth acceleration.

Nexus 7 Tablet - The Nexus 7 is a tablet computer developed by Google in conjunction with Asus. It is the first tablet in the Google Nexus series of consumer devices using the Android operating system and built by an original equipment manufacturer partner.

Csv file format - A comma-separated values (CSV) file stores tabular data (numbers and text) in plain-text form. Plain text means that the file is a sequence of characters, with no data that has to be interpreted instead, as binary numbers.

Byte - The byte is a unit of digital information in computing and telecommunications that consists of eight bits.

Bit - A bit is the basic unit of information in computing and digital communications. A bit can have only one of two values, and may therefore be physically implemented with a two-state device. The most common representation of these values are 0 and 1. The term bit is a contraction of binary digit.

UART - The Universal Asynchronous Receiver/Transmitter takes bytes of data and transmits the individual bits in a sequential fashion. At the destination, a second UART re-assembles the bits into complete bytes.

Introduction

The team consists of three Cal Poly San Luis Obispo undergraduate engineering students. The team was advised by Art MacCarley, an electrical engineering professor with extensive knowledge of vehicle controls, software, and mechanical expertise. John Dunning, the team's sponsor, is currently a research scholar at Cal Poly San Luis Obispo and long time electric vehicle engineer and enthusiast. John Dunning will be allowing us to access his Nissan Leaf, the platform where the team will be developing and testing the product on. Ike Bayraktar, President and CEO of Electricore Inc. has provided us the funding for this project. This project will also involve the use of Cal Poly resources such as labs, equipment, research material, and faculty support.

What the team plans to do for the sponsor?

John Dunning is most interested in the vehicles battery characteristics. Data that the team can obtain from the OBD relating to battery charge, charging times, and charging cycles will be the team's main priority. Also, the team is going to setup the device in the vehicle that is less obtrusive to the driver.

Goals:

1. To analyze the vehicles battery related data that is accessible through the OBD port in the Nissan leaf
2. Log the data values pertinent to the client's needs, all without voiding the cars warranty
3. Use the logs of collected data values to create visual graphics of the cars battery characteristics that will help reveal trends within the battery
4. To create a GUI for the Nexus 7 tablet. The GUI will display the graphed data in a clean and organized manner so that the end user can utilize this data for any means that they desire

Project Management

Art MacCarley - Project advisor
Philip Tyler - Software and firmware development
Eric Woodruff - Hardware and firmware development
John Allande - Software and firmware development

Stakeholders:

Art MacCarley (advisor)
John S. Dunning (sponsor)
Ike Bayraktar (sponsor)
Potential Customers
Cal Poly (ENGR470/471)

Background

Project Background:

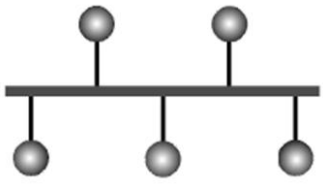
All cars and light trucks built and sold in the United States after January 1, 1996 were required to be equipped with OBD II ports. OBD stands for on board diagnostics and the "II" is the latest standard required by all car manufacturers to implement on their vehicles. The OBD-II standard specifies the type of diagnostic connector and its pin out, the electrical signaling protocols available, and the messaging format. It also provides a candidate list of vehicle parameters to monitor along with how to encode the data for each. Even though there is a unified connector and its respective pin outs, multiple signaling protocols are allowed and use through the port.

The Nissan Leaf uses the signaling protocol commonly referred to as the CAN bus. The CAN bus network in the LEAF connects all the major modules like the brakes, locks, lighting, engine and more and relays all the information between each other. Since almost every component of the car is connected through the CAN bus, all these components can be monitored and controlled by the OBD II port through different ECU's or electronic control units. As vehicles today become more and more automated with ever increasing amounts of sensors and electronic controls, the car of the future will be more similar to your computer than your sports car.

A device that has inspired possible design queues for the teams final device is Progressive's Snapshot device. The Progressive's Snapshot plugs into a car's diagnostic port and then begins to monitor the driver's driving habits. The device keeps track of how often the driver presses hard on their brakes, how many miles they drive, and how often one drives between midnight and 4:00 a.m. There is another similar device that has already been developed and on the public market that is called the Bluetooth OBDII Connector. This device comes with an app on the android market, provides driving trends, and useful information on the car to the owner.

What is a CAN Bus?

CAN or controller area network is a vehicle bus standard designed to allow microcontrollers and other devices to communicate with each other within a vehicle. Figure 3 describes the topology of the controller area network. Each of the dots represents a different microcontroller or system that uses the bus. Figure 4 is an oscilloscope capture of the data streaming across the differential signal CAN bus while Figure 5 shows the format for a single CAN message.



Bus network

Figure 3 - CAN Bus Network Topology

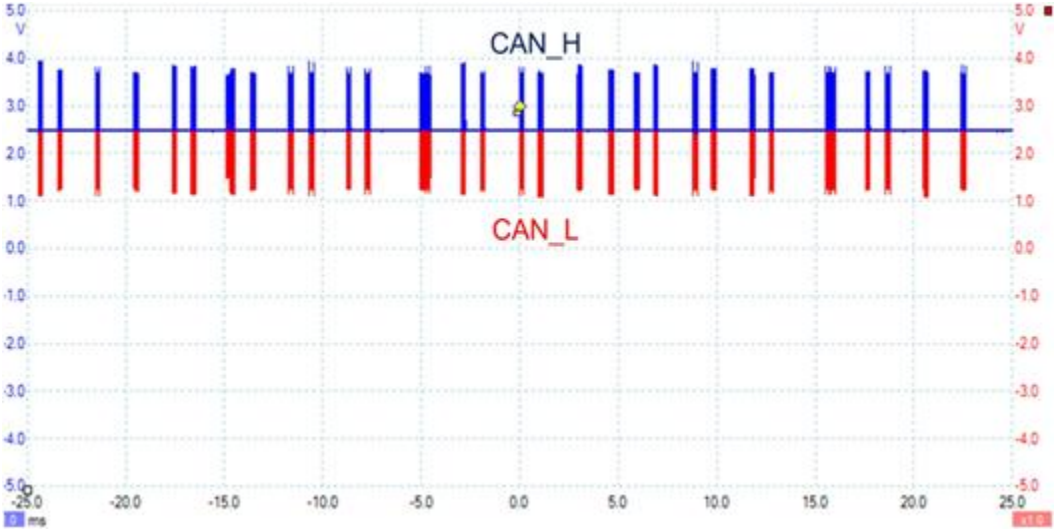


Figure 4 - Example of CAN bus activity

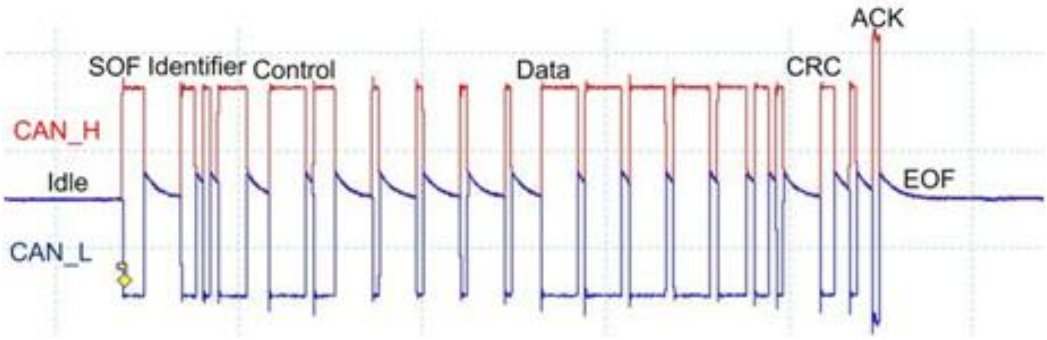


Figure 5 - CAN Message Structure

The identifier field contains the message id of that specific message. The data field can be 0-8 bytes (0-64 bits) per message depending on number in the control field (0-8 depending on number of bytes to send).

Nissan Leaf CAN Bus Network

The Nissan Leaf has 3 CAN buses.
Primary CAN Bus (pins 14 and 6)
Contains: throttle position, vehicle speed, brakes, blinkers, etc
EV CAN (pins 12 and 13)
Contains: battery voltage, battery current, motor RPM, state of charge, etc
AV CAN (pins 11 and 3)
Contains: information between the Navigation unit and buttons

Current products:

John Dunning currently has an OBD device, called the CAN-DO box developed by Gary Giddings; the CAN-DO box is specifically designed for the Nissan Leaf that displays data such as electric vehicle speed and battery charge. The team is using this device as the team's starting platform and will be further developing the work Gary has already accomplished. The teams final device will be the only OBD Bluetooth device that has been developed for the Nissan Leaf; most OBD devices are developed for combustion engine vehicles. Some examples of these devices are Craven Speed's "Bluetooth or Wi-Fi OBDII Connector" which can be viewed under the References section.

Design Development

The team originally planned to build a more compact, wireless device for John Dunning. However, one of the members of the team who had good mechanical engineering skills had to leave the team for an internship he received after the first quarter. Therefore, the team had to change the final design for the project concluding on only adding wireless connectivity and creating a GUI on the Android Nexus 7 tablet. Due to this, the team does not have any design discussions that helped create the final product.

Description of the Final Product

Engineering Specs

1. OBD interpreter communicates via Bluetooth with the Nexus 7 tablet
2. To not interfere with the knees and legs of the driver
3. Android app to manage and collect data.

Embedded System Software

1. Ease of use
2. Connect to the Nexus 7 tablet via Bluetooth
3. Consistent results

Android App

1. Connect to the Nexus 7 tablet via Bluetooth
2. Allow user to start and stop collecting data (GPS, Voltage, Current, State of Charge, RPM, Time Stamp)
3. Be able to visually display live the incoming data
4. Be able to email the logged data to a personal email account

Milestones

1. Connect and read data off the Nissan Leaf CAN bus **1/28/13 - 2/11/13**
    a. Spliced an OBD cable
    b. Using a digital multimeter using the ohm resistance setting, the team determined which pin connected to which wire
    c. Placed each wire into a breadboard in order to read from the two pins that connected to the CAN bus of the OBD-II port
    d. Connected an oscilloscope to the two CAN bus pins and successfully received data from the cars OBD-II port

2. Compare the data read with Garry Giddings recipe files **2/11/13 - 2/18/13**
    a. The data received with the device matches up with the software provided by Garry Giddings; his software can be downloaded at www.wwwsite.com/puzzles/cando

3. Search for unknown parameters (hacking/tracing) **2/18/13 - 3/1/13**
    a. After hours of research and exchanging emails with Gary Giddings, the team came to the conclusion that discovering new message-ids was much harder than originally anticipated
    b. The team found out that one would need proprietary information to know any more of the message-ids that are not already found by www.MyNissanLeaf.com

4. Build an app for the Nexus 7 tablet in order to display logged data **3/1/13 - 6/10/13**
    a. The team successfully built an Android app for the Nexus 7 that John Dunning has received

Gantt Chart

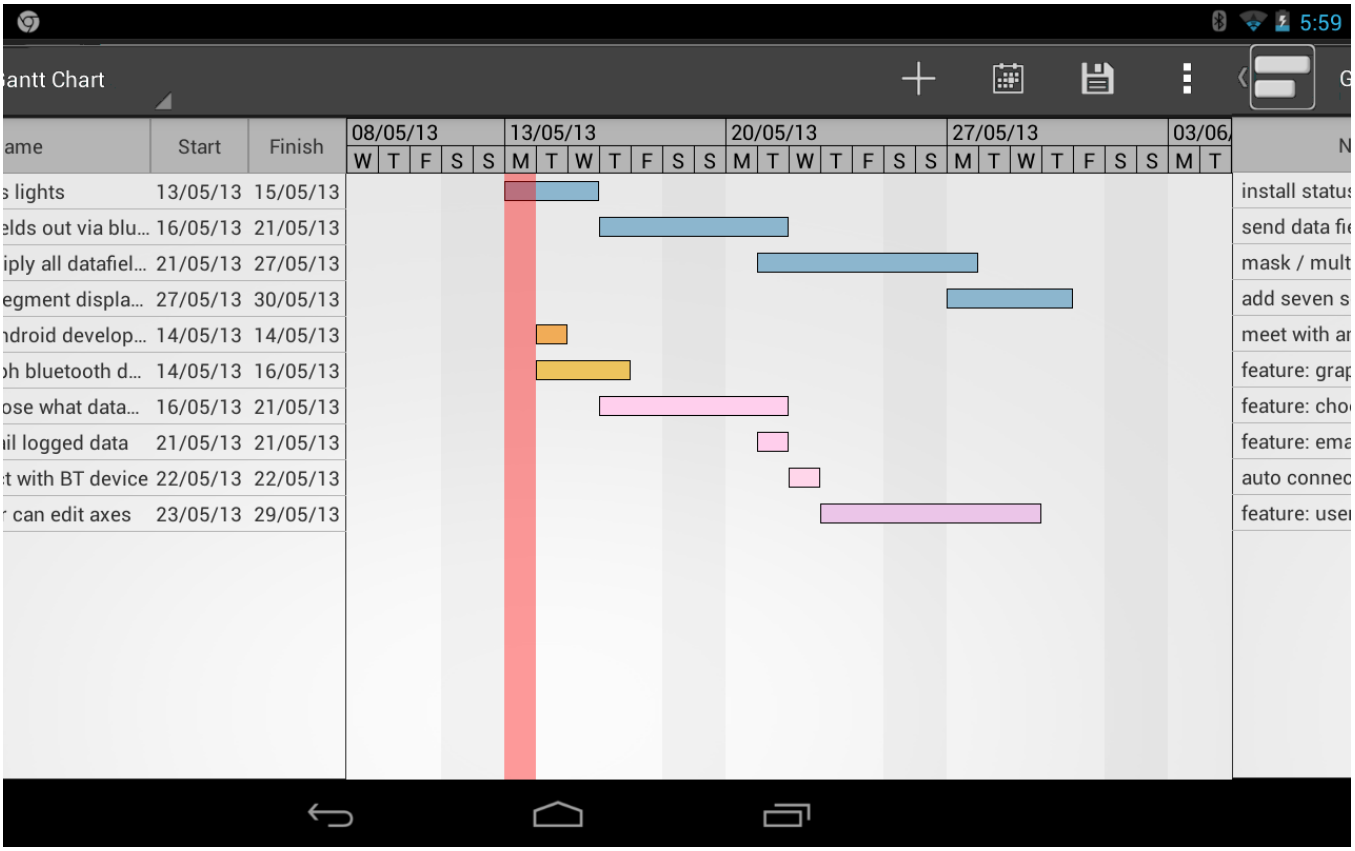


Figure 1 - The team's Gantt chart

Firmware (Loaded onto AVR-CAN board)

1. *Description*
    a. Initialize UART-0 at 115200 baud rate
    b. Wait until there is activity on the RxCan
    c. Wait until receive a successful message-id and its corresponding data
    d. If there is no error, send the message and data to the Bluetooth so that it can send it to the Nexus 7

Message-ids

Message Id	Data
5BC	State of Charge
1DA	Motor Amps and RPM
1DB	Battery Pack Voltage and Current

Bluetooth Message Structure

The device will transmit 11 bytes for a single data point. The first byte sent out is the sync byte. Once the Nexus 7 tablet reads the byte 0x53, the program will know the data is coming in the format below. Each data point has a specific message id as well as up to 8 bytes of data. For example, the battery voltage parameter has a message id of 0x01 and the value is represented in hex. The example below is sending the value 360 in the third row in the table below. 360 decimal = 0x168 hex

1	2	3	4	5	6	7	8	9	10	11
sync	id0	id1/dlc	data0	data1 LSB	data2	data3	data4	data5	data6	data7 MSB
0x53	0x01	0x00	0x68	0x01	0x00	0x00	0x00	0x00	0x00	0x00

Table 1 - Data Structure of Bluetooth Message with example of 360V message

Hardware Schematic

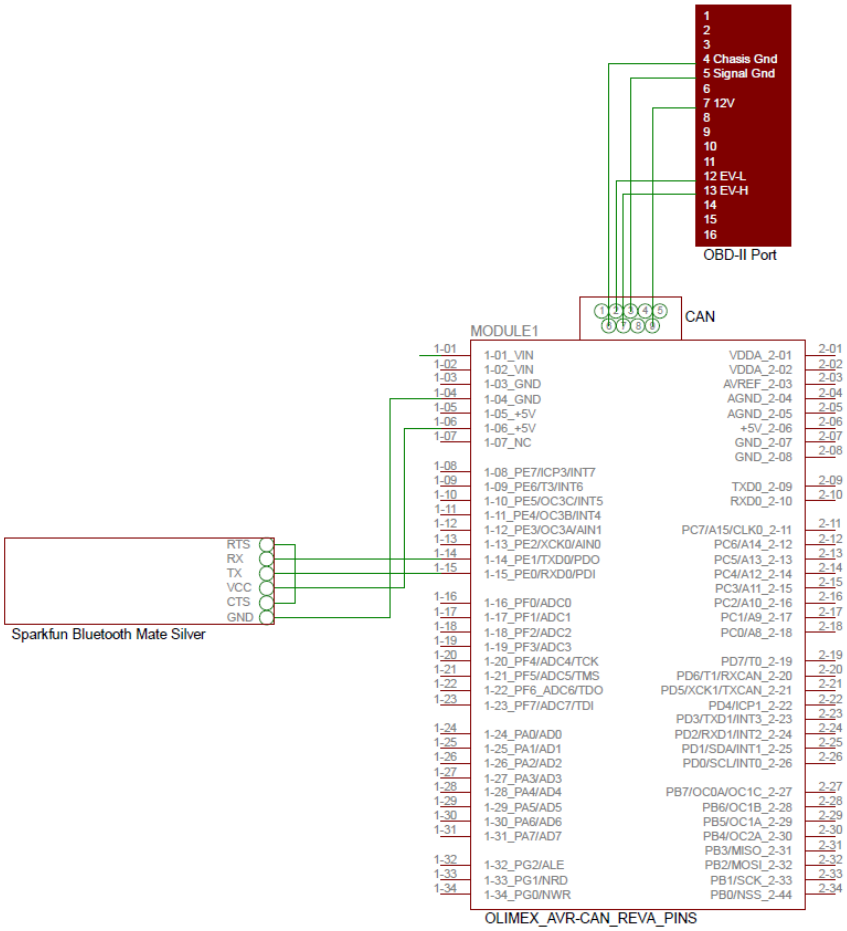


Figure 1 – Hardware schematic of the hardware within the device's plastic box

Software (Android Application)

1. *Description*
    a. Designed with Eclipse IDE running the Android SDK plug-in
    b. Images (background, buttons, etc) were designed specifically for a portrait-oriented Google Nexus 7 tablet. See figure below.
    c. Designed for Android Version 4.2
    d. Utilized the Activity class's lifecycle to connect and disconnect from Bluetooth.
2. *Architecture*
    a. On startup (Activity.onCreate()) the app checks the Nexus's Bluetooth is enabled, and is paired to another Bluetooth device named, "OBD Bluetooth."
        i. App terminates with an error dialog if either check fails.
    b. Then (Activity.onStart()) the app attempts five times to create a Bluetooth socket with the OBD Bluetooth device. If successful, the app sets it's layout to the figure below, otherwise it terminates with an error dialog.
    c. The "Collect EV Data" button, when pressed, calls a function that creates an AsyncTask object.
        i. The AsyncTask represents a background thread that constantly reads 11-byte messages sent by the OBD Bluetooth device, and writes the EV data values to corresponding files.
        ii. When the "Collect EV Data" button is pressed again the collection thread is canceled and the UI is refreshed.
        iii. All the created log files are then attached to an email, created by an Android Gmail intent.
        iv. The user can then send the log files to any email address, then return to the EV Data Utility App to collect more data.
    d. When the App is stopped or terminated, Activity.onStop() is called which closes the Bluetooth socket and frees allocated resources.

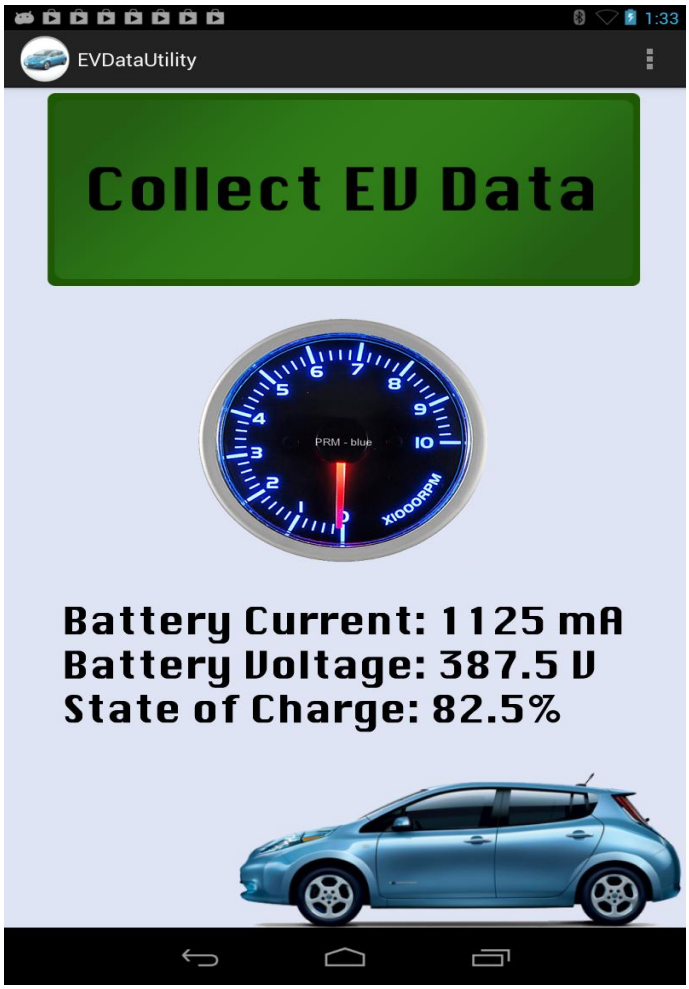


Figure 2 - EV Data Utility App after Connecting to OBD Bluetooth Board

User-Manual

1. Plug in the OBD-II cable into the OBD-II port of the Nissan Leaf. The OBD-II port is located underneath the steering wheel around the driver's knees.
2. Turn the ignition on so the device can turn on.
3. Open the EVDataUtility app on the Nexus 7 and click “Start Capture”. The red LED on the CAN-box should stop blinking and the green LED should light up once the devices have been connected.
4. When finished collecting data, click “Stop Capture”.

Component Selection and Cost Breakdown

1. RN-42 Bluetooth chip - $16
2. Olimex AVR-CAN board - $33
3. Plastic box - $5
4. Nexus 7 tablet - $199
5. 3mm Light Pipe (2) - $1 each
6. OBD-II cable - $5

7. Sparkfun OBD-II UART development board - $49.95
8. Dremel – personal property
9. Windows Computer – Personal property
10. Soldering tool – Provided by school
11. Wire stripper - personal property
12. Zip ties - $1
13. Android Development Kit - provided online by Google

Total Cost: $310

Product Realization/Design Verification (Testing)

Test 1- Use Sparkfun board to communicate with a car

Purpose - After reading online and searching for various OBD readers, the team purchased an OBD-II to UART development board. The team thought that this board was a good starting point to start communicating with the CAN Bus.

Setup - The team tested the device by plugging the development board into a 09' Honda Civic Si using an OBD-II reader program on a laptop.

Results - The program displayed a data in real time and worked as advertised.

Figure 3 - Testing for CAN signals with a Sparkfun board

Test 2 - Use oscilloscope to view CAN messages

Purpose - Before the team had the final device operational, the team wanted to view the data streaming from the Nissan Leaf's OBD port.

Setup - Using a modified OBD cable so that only data from the CAN bus was received, the team connected an Agilent MSO-X 3014A Oscilloscope to the OBD port using the OBD-II UART development board to view the CAN bus data in real time.

Results - This test was successful.

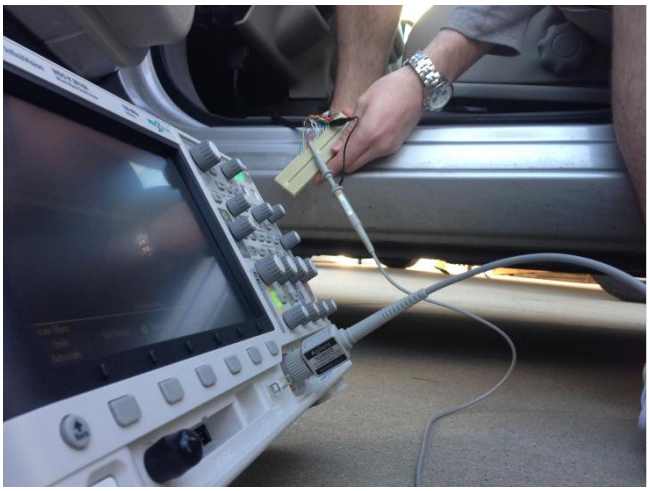


Figure 4 - Using the oscilloscope to view the EV CAN Bus

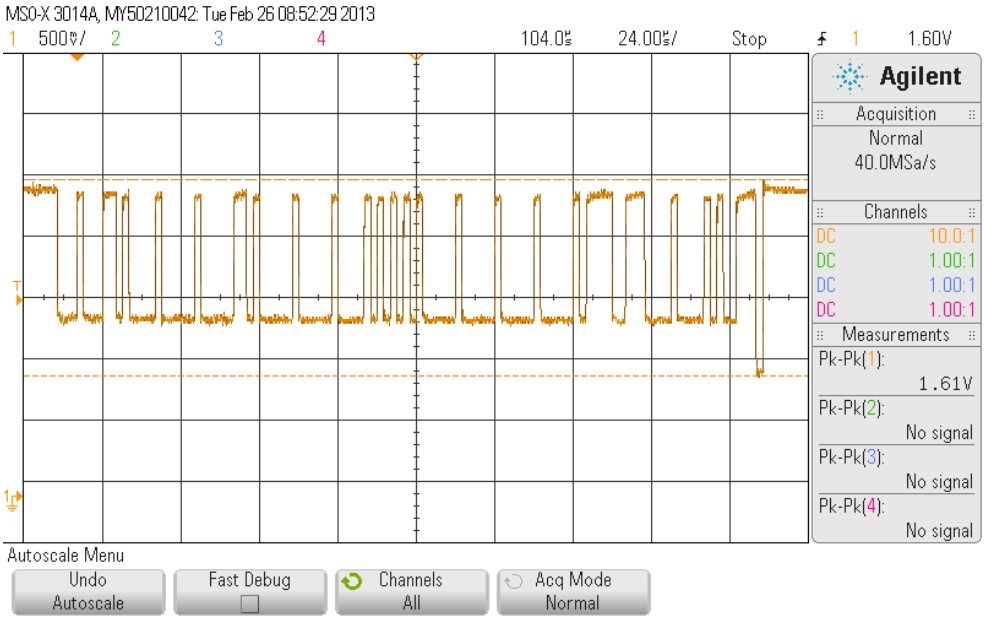


Figure 10 - Oscilloscope Capture of EV CAN Message on Nissan Leaf

Test 3 - Use the CAN-DO box to view CAN messages

Purpose - The team wanted to become more familiar with the CAN bus and CAN-Do box.

Setup - Purchased a CAN-DO box from Garry Giddings online and assembled the kit.

Results - The team used the CAN-Do program provided with the CAN-Do box to capture the data streaming across the Nissan Leaf bus. The team learned that data could not be read from the OBD-II port unless the ignition was on.

Test 4 - Record data on small drive using the CAN-Do box

Purpose - The team wanted to record live data on a short drive.

Setup - The team plugged the CAN-Do box into the OBD-II port of the Nissan Leaf and started recording data. After driving around Cal Poly for 10 minutes, all data was recorded with a laptop.

Results – The team was able to graph the vehicles battery voltage and current over the course of the 10 minute trip It is interesting to note how the spikes in voltage and current are inversely proportional to each other.

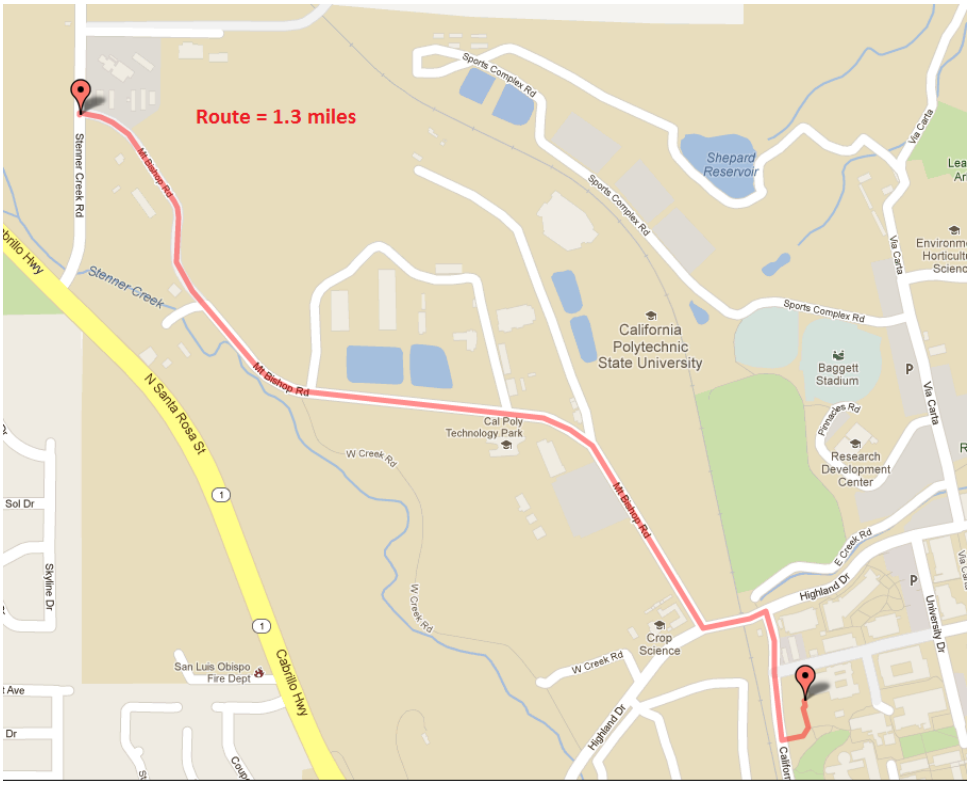


Figure 5 - Route of Test Drive

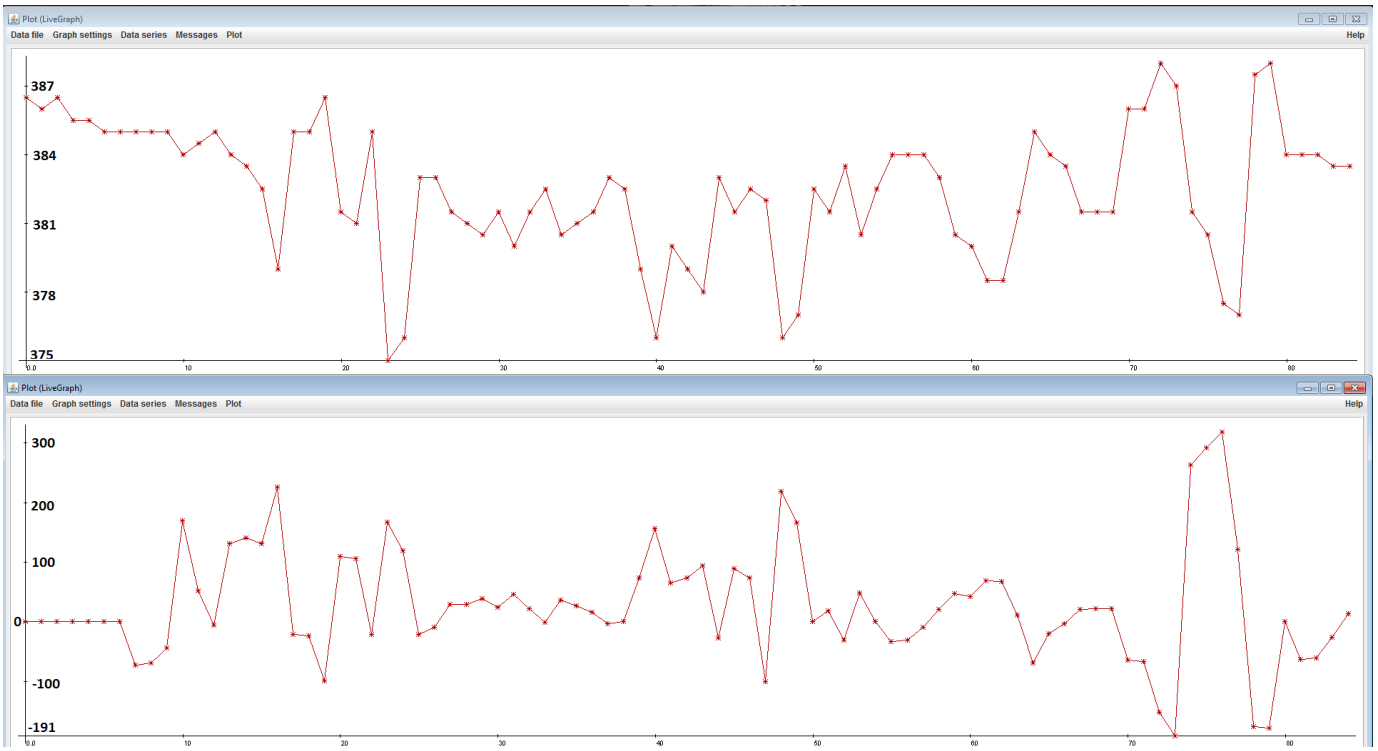


Figure 6 - Battery Voltage (top) and Battery current (bottom) during test drive.

Test 5- Understand the format of data being sent to Nexus 7

Purpose- The team wanted to verify that the modified CAN-Do box is able to receive data from the CAN bus.

Setup- After hooking up the device to the Nissan Leaf's OBD-II port, a program called Realterm was used to display the raw data.

Results- Data is not available on the bus unless the ignition is on. Once the ignition is on, data streams on the bus continuously.

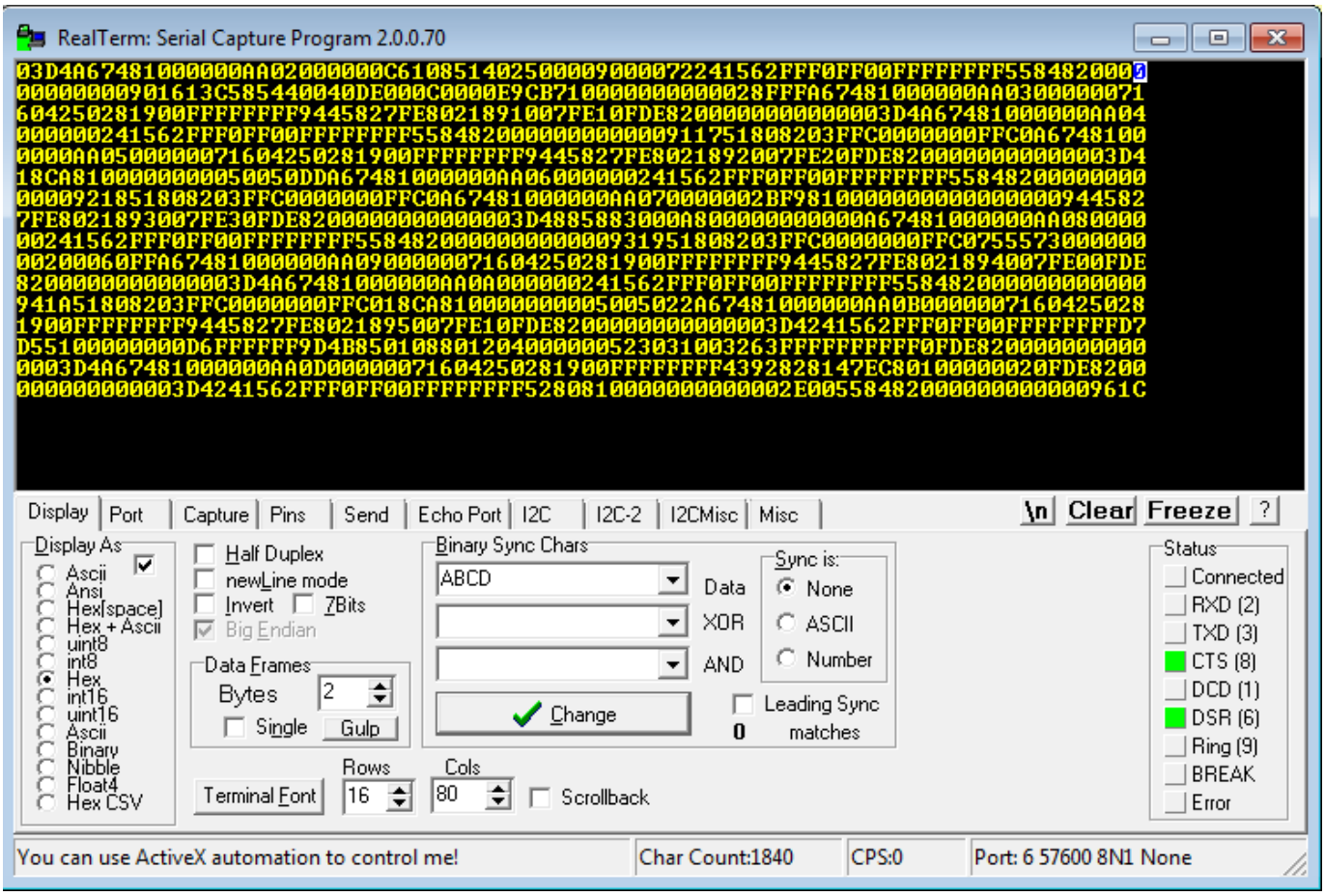


Figure 7 - Screen capture of Data received on EV Bus

Test 6 - Send known data from Bluetooth

Purpose- The team conducted this test in order to ensure proper connection and communication between the OBD Bluetooth device within the final product and the Nexus 7 Tablet.

Setup – The team programmed the OBD Bluetooth device to send values ranging from 360 to 380 to simulate battery voltage. For example, the battery voltage parameter has a message id of 0x01 and the value is represented in hex. The data that is being sent is 360 which can be viewed in the third row of the table below. 360 in decimal = 0x168 in hex.

Results- Received all the data successfully

1	2	3	4	5	6	7	8	9	10	11
sync	id0	id1/dlc	data0 LSB	data1	data2	data3	data4	data5	data6	data7 MSB
0x53	0x01	0x00	0x68	0x01	0x00	0x00	0x00	0x00	0x00	0x00

Table 2 - Data Structure of Bluetooth Message with example of value 360 for message-id 0x01

Firmware for this test

```
// send 360
uart_putchar(0x53); // send sync byte
uart_putchar(0x01); // send LL byte
uart_putchar(0x00); // send DH byte
uart_putchar(0x68); // data0
uart_putchar(0x01); // data1
uart_putchar(0x00); // data2
uart_putchar(0x00); // data3
uart_putchar(0x00); // data4
uart_putchar(0x00); // data5
uart_putchar(0x00); // data6
uart_putchar(0x00); // data7
DelayMsec(100);
```

Estimated Total Hours
200 hours - Software
200 hours - Hardware
100 hours - Integration and Testing

Actual Total Hours:
150 hours - Software
100 hours - Hardware
200 hours - Integration and Testing

The team overestimated how much time it would take to read data from the Nissan Leaf CAN Bus.

Conclusions & Recommendations

Conclusion:

After three quarters of planning, designing, and building the device, the device successfully streams data from the EV bus data off the Nissan Leaf to the Nexus 7. The Android app on the tablet successfully receives data for the four message-ids described previously. The app also emails the logged data to a personal email account, RPM values are displayed using a graphical tachometer (however the RPM values at this time are invalid), and the other values are digitally displayed. The team would recommend for another person or group to discover more message-ids, fix the RPM function of the app, improve the efficiency and stability of the firmware/software, add more status LEDs, and design a smaller PCB so that overall device is smaller, more compact, and more power efficient.

Future Development:

Android App

- More reliable Bluetooth Socket Creation
- Write files to external memory instead of Cache
- Select with program to send log data files to (Dropbox, Google Drive, etc)

OBD Bluetooth Box

- More status LEDs
- Internal Data masking and manipulation
- Recognize more EV message IDs

How to Install the Firmware on CAN-AVR Board

- Install the WinAVR
- NOTE: For the files to compile correctly, rename the file to spy.c
- Use Programmer's Notepad within WinAVR to edit and compile spy.c
- Power on the AVR-CAN Board using 12V power supply provided
- Use Atmel Studio 4 (not Studio 6) to connect to the AVR-JTAG-USB programmer and load spy.hex (compiled file from Programmer's Notepad)

Transition Plan

1. Read this entire report, maybe a couple of times, to get an understanding of the project and how it was designed and developed into the current product it is
2. Follow the installation steps to begin working on the project
3. Read carefully the OBDBluetoothv2.c file and all the comments to understand the functionality of the firmware
4. Read carefully the Android project files and all the comments to understand the functionality of the software for the Android app

Acknowledgements

Advisors:

- Dr. Art MacCarley: Main advisor/supervisor of project. Experienced with Automotive electronics and systems
- Dr. James Widmann: Interdisciplinary Capstone Professor/Advisor
- Dr. Lily Laiho: Interdisciplinary Capstone Professor/Advisor
- Dr. Richard Savage: Interdisciplinary Capstone Professor/Advisor

Sponsors:

- John Dunning: Main customer and project requirements correspondent, owner of Nissan Leaf the project is designed around
- Garry Gidding: Creator of the CAN-DO box, and avid Nissan Leaf OBD data hobbyist
- Ike Bayraktar: CEO of Electricore, generous financial donor of entire project
- Cal Poly SLO: Classroom, electrical, and other resources

Figure 8 - Senior Design Expo May 30th (Eric Woodruff, Philip Tyler, Dr. Art MacCarley, Dr. John Dunning)

References

Philip Tyler
John Allande
Kevin Shelton
Eric Woodruff
11-10-12

OBD/EV

John Dunning Requirements (Whats)		Weighting (Total 100)	BlueToothChip	Custom PCB	OBD Chip	Durable Plastic Casing	Java GUI	Test Driven Development	Mode Button Interrupt	RGB LED and PWM	Store Data on Connected Devices	Smallest Ics Possible	Solid CAD Design	CAD \Desing for Manufacturing	Gui Built with Qt	Reference Mint.com GUI	Be Able to Replace instruction Set	Constant Stream of CAN Bus				Benchmarks: Craven	Benchmarks: Mavizon	
			Engineering Requirements (HOWS)																					
Customer Requirements (Step #2)	Small	6		●		○	○		Δ			●	Δ	○	○	○						4	3	
	Portable	5	○	●	Δ	○	Δ		Δ						Δ	Δ						4	4	
	Easy to Use	6	○				●															2	5	
	Wireless	6	●					Δ			○											4	4	
	Beautiful GUI	4					●				○				●	●						0	5	
	Cross Platform	4	○				●	○	○						●	●	○					4	5	
	Aesthetically Pleasing	3		○		●	●			●			●		●	●						2	5	
	Durable	5		●		●	Δ								Δ	Δ						3	4	
	Consistant Performance	5	○	Δ	●		○		●						○	○	○	○				3	4	
	Unlimited Data Storage	3																○				0	0	
	Concatinate Results	4					●	○			○							●				0	0	
	Find Trends over time	4					●	○			○							●				0	4	
	Status LEDs	3		○					Δ	●												2	4	
	Mobile Phone App	2	○						○													4	5	
	Secure Physical Connection	4				●												Δ				3	2	
	Low Cost	4	Δ	●	○	●						Δ										5	4	
	Mass Manufacturable	2		●								●		●								5	4	
	Easy Connectivity	4	●			●			Δ													3	4	
	Show ALL data available	3					●								●	●		●				2	3	
	Updatable software	2			Δ		●	Δ							●	●	●					2	3	
	Accessory Integration	2	●				●								●	●						2	0	
	Sleep Mode	4		Δ					●	○												0	0	
	Real Time Data	6	○		●		●	○	○						●	●						4	4	
	Use on Different OBD-II proto	6			●			●						○				●				4	5	
	Controllable Sampling Rate	3			●		●		○						●	●		●				0	0	
	Safety	6		●	●																	3	3	
	Does not void warranty	6		●	●	○																4	4	

Figure 9 - Engineering Requirements for the team's project

OBD-II Cable

WIRE COLOR	OBDII PIN
Black	1
Brown	2
Red	3
Orange	4
Yellow	5
Green	6
Blue	7
Purple	8
Grey	9
White	10
White w/stripe	11
Brown w/stripe	12
Red w/stripe	13
Orange w/stripe	14
Yellow w/stripe	15
Green w/stripe	16

Figure 10 - Relationship of the wires in the OBD-II cable with their respective pins

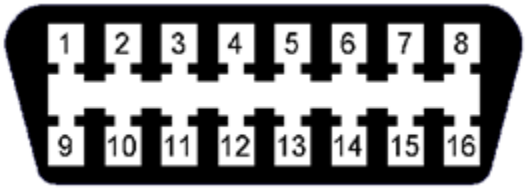


Figure 11 - OBD-II Connector

Figure 12 - OBD-II Port Location

Works Cited

SparkFun Electronics. SparkFun Electronics Inc., 0 June 2013. Web. 04 June 2013. <https://www.sparkfun.com/>.

"AVR-CAN." *Olimex*. Olimex, 2013. Web. 04 June 2013. <https://www.olimex.com/Products/AVR/Development/AVR-CAN/>.

Gidding, Gary. "LEAF CAN-Do Analysis Programby GaryG." *Gary G's CAN-Do Program Page*. VIPS, 14 Apr. 2013. Web. 04 June 2013. <http://www.wwwsite.com/puzzles/cando/>.

Gidding, Gary. "LEAF SOC-Meter Projectby GaryG." *Gary G's SOC-Meter Page*. VIPS, 12 Oct. 2012. Web. 04 June 2013. <http://www.wwwsite.com/puzzles/socmeter/>.

"My Nissan Leaf Forum." *My Nissan Leaf Forum*. MyElectricCarForum, 04 June 2013. Web. 04 June 2013. <http://www.mynissanleaf.com/index.php>.

"On-Board Diagnostics." *(OBD)*. SAE International, 2013. Web. 04 June 2013. <http://standards.sae.org/electrical-electronics-avionics/obd/standards/>.

"Bluetooth or WiFi OBDII Connector." *CravenSpeed Online*. Craven Speed, 2011. Web. 10 June 2013.

"Your Smart Driving Assistant." *Automatic* -. Automatic Labs, Inc., 2013. Web. 10 June 2013.

"SF BayLEAFs." *SF BayLEAFs*. SF Bayleafs, 2013. Web. 04 June 2013. <http://sfbayleafs.org/>.

Contact References

1. Philip Tyler (Philip.b.tyler@gmail.com)
2. Eric Woodruff (ewwoodruff@gmail.com)
3. John Allande (jjohnallande@gmail.com)
4. Art MacCarley (amaccarl@calpoly.edu)
5. John Dunning (gmwestern@aol.com)
6. Gary Giddings (ggother99@wwwsite.com)