IoTECH
Internet of Things Extensible Car Hub

Nigel S. Paine, CSE; Raghid Bahnam, EE; Nicholas A. Korniyenko, EE; Christopher R. Ingerson, CSE

Abstract — "IoTECH", also known as the Internet of Things Extensible Car Hub, is an IoT device that interfaces with the OBD-II diagnostics port and a variety of external sensors. In an era of WiFi-enabled devices, we seek to implement a device that is unique to automotive systems. This system will allow the user to create an application that will interface with an array of sensors and communicate serially or wirelessly via Bluetooth, WiFi, and 3G data. User data is transmitted via 3G to a web server where notifications are then broadcast. A sample application being implemented with IoTECH is a "TempAlert" system which notifies a user if there is a person in the car and a certain temperature threshold is reached.

I. INTRODUCTION

We see more IoT devices integrated into homes across the country, but there aren't such devices out there for automotive applications. Nest thermostat and Hue Philips Light Bulbs are examples of IoT devices. Both are app controllable where prefered lighting and temperature can be controlled via app. Nest can be programmed to set the temperature when the user comes home where Hue can change each light bulb to a prefered color and brightness [21][22]. Existing automotive IoT devices are limited to the space underneath the steering column where the devices plug into the OBD-II port [13][14][15]. These devices may have a variety of sensors but are limited to being under the steering column. For example, a device is able to give GPS coordinates but not give a picture of a breakin, or detect motion of an endangered child due to temperature. What we plan to create is a "smart hub" which will be directly connected to a vehicle's OBD-II port but is also extended with sensors to produce many more different applications that can't be done under the steering column. This includes infrared sensing and picture taking done by this extension. This IoT device would be both extensible and modular, capable of integrating with most sensors via a hardwired or wireless connection. One of the main goals of this device is to receive live-updated information directly to the user's phone, whether it is getting temperature alerts or notifying the user if the car has been hit. We hope to extend the usability of the OBD-II port beyond its current capabilities using our technology. Our device could save lives of children left by their parents in hot cars, catch thieves red handed, report that your car got hit, and so on forth. It will make this country safer. The applications can be limitless. We want this device like home IoT devices to be extended not only to a few applications and sensors but to maybe hundreds to suit each person's needs. Table 1 shows the hardware specifications for our Hub and Extension.

Requirement	Specification
Small/Portable	Hub size of a phone (2.5"x2.5"x2.5") w/ extension (5"x2.5"x3")
Lightweight	Be able to drive car without device falling down and damaging diagnostics port (hub <100g & extension <200g)
Extensible	Supports Serial, WiFi, Bluetooth, and 3G communications
No lag communication	Notification Delay < 1 min
Long-lasting car battery life	Lasts 160-200hrs (~1 week)
Long-lasting extension battery life	Lasts > 12hrs.

Table 1. System Specifications

II. DESIGN

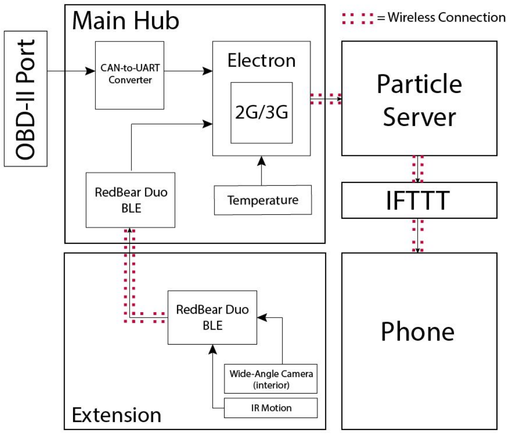


Fig. 1. IoTECH Signals Block Diagram

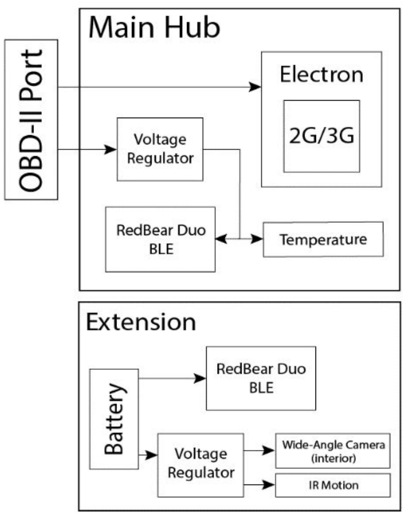


Fig. 2. IoTECH Power Block Diagram

A. IoTECH Hub

The most important part of the design in Figure 1 is the main hub. For our hub, we are using the Particle Electron [24] as the main device. The main goal of our hub is to request data from the OBD-II port and extension and use the data from the OBD-II port, extension, and its own external sensors in order to run a set of designed applications.

The OBD-II port provides crucial diagnostic information about the current status of the car, and also provides constant power for the Electron. It supplies 12 volts, which can be fed directly into the V_{in} pin of the Electron. The Electron has a built-in voltage regulator that can take an input of up to 18 volts and converts it to the 5 volts necessary to run the Electron. More information about the power supplied by the OBD-II port will be provided in later sections.

Attached to the Electron, we will have a Bluetooth module [25] that is able to accept communications from the IoTECH Extension; the extension will be explained in the next section. The data received by the module will be transmitted to the Electron via the TX/RX pins, where the data can be stored in variables and used in applications.

Currently, we only have one sensor on the main hub, measuring the temperature and humidity of the ambient environment inside the car. The purpose of this is to make sure that any young children or animals left in the car, whether intentionally or accidentally are not in any danger as temperatures rise in the summer. All of the variables are processed in the firmware of the Electron, and once an application evaluates to true, an event is published to the cloud via 3G. Communication with the server and the integration of If This Then That [26] will be discussed later.

B. IoTECH Extension

The next biggest component in the IoTECH device, and more of a proof-of-concept than anything is the IoTECH “Extension”. The extension’s role is to “extend” the functionality of the main smart hub and allow users to interface with external sensors via Bluetooth (LE) and WiFi (802.11 b/g/n) connectivity [4]. This device is powered by a 3.7V Lithium Polymer (LiPo) battery [17] and holds any sensors that require better location placement in the car (i.e. not under the dashboard). The LiPo battery is boosted to 5V with a step-up voltage regulator [18] to support powering any sensors relying on a 5V input.

The main processing unit on the extension is the RedBear Duo [4] with Bluetooth/WiFi capabilities and firmware supporting the Redbear API in addition to many of the libraries from the Particle and Arduino APIs [8]. The main programming development environment being used to upload code to the board is the Arduino IDE [19].

The first main sensor used on the extension is the wide-angle camera. The camera we chose is the LinkSprite JPEG Serial UART Infrared camera. This camera can have a variable resolution but has a default resolution of 160x120 with night vision enabled by infrared LEDs and up to 120 degree field of view [9]. Serial communication is enabled by connecting the camera to the RedBear Duo’s serial pins (pin D16 and D17 or the RX/TX pins as seen in Figure 3 with the Redbear pinout map). The given Arduino code is not readily

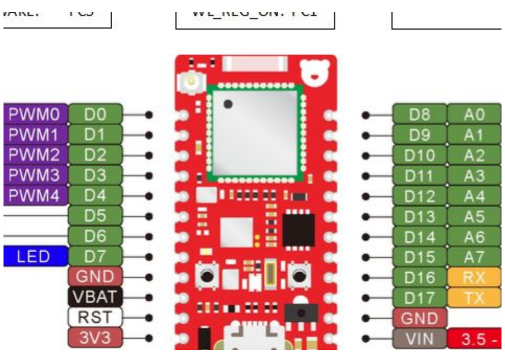


Fig. 3. RedBear Duo Pinout

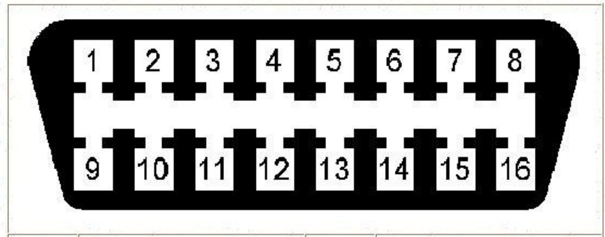
compatible with the Redbear Duo due to hardware limitations so changes in the code have to be implemented to make sure that when serial data is sent over to the Redbear device, the data buffer does not overflow, in addition to the baud rates being set accordingly to avoid data errors. Camera picture data is sent as a stream of hex values formatted in JPEG format.

The second sensor being implemented on the extension is the HC-SR501 passive infrared (PIR) motion sensor. This sensor will be able to communicate in parallel with the camera so that when motion is triggered, the camera turns on and snaps a picture of the car's interior. The delay on the motion sensor can be set directly on the device (ranging from 0.3 seconds to 5 minutes) or via software. The motion sensor also has an angle of view of up to 110 degrees and up to 7 meters away [10].

With the motion sensor and camera, the "TempAlert" system can be implemented more robustly, allowing the user to only get notifications when there is motion detected in the car in addition to receiving a low-resolution picture (to avoid high 3G data consumption) identifying who or what is in the car. TempAlert is a proof-of-concept application developed by our team incorporating IoTECH allowing the user to receive notifications if the internal temperature exceeds a set threshold while also detecting motion and capturing an image to ID false positives.

All hardware components are being tested via the Arduino serial monitor [20] to validate that the correct data is being sent/received. The serial monitor echoes data via the USB-connected Redbear Duo. Data tests will also be run in parallel with the smart hub on the Particle Electron microcontroller to validate that data is being sent via Bluetooth and WiFi in a secure and consistent manner.

C. OBD-II Communication



PIN	DESCRIPTION	PIN	DESCRIPTION
1	Vendor Option	9	Vendor Option
2	J1850 Bus +	10	j1850 BUS
3	Vendor Option	11	Vendor Option
4	Chassis Ground	12	Vendor Option
5	Signal Ground	13	Vendor Option
6	CAN (J-2234) High	14	CAN (J-2234) Low
7	ISO 9141-2 K-Line	15	ISO 9141-2 Low
8	Vendor Option	16	Battery Power

Fig. 4. OBD-II Pinout [6]

Reading OBD-II was done in few stages. OBDII uses three protocols to communicates with the outside word. Since 2008, one protocol was mandated in every small size car in the USA. It is known as the CAN protocol [23]. With speed up to 1 mbps, it is the fasted protocol that OBDII uses. IoTech works only with cars that uses CAN protocol. Before being able to read any data, the 12V of the car's battery needed to be regulated to 5V and 3.3V to provide the required voltage that the system needs to operate. This was done by using L7805C that regulated the voltage down to 5V. That 5 V was then further regulated down to 3.3V using LE33.

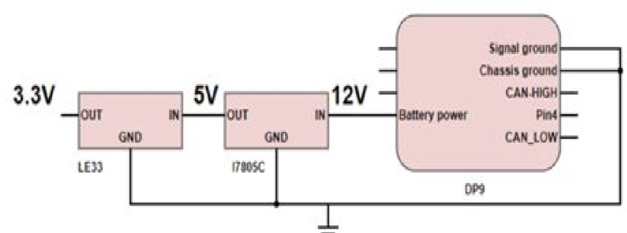


Fig. 5. Voltage regulator for the car battery

The system responsible of reading the data consists of three subsystems. They are: a differential subsystem, an interrupter subsystem, and finally the host. CAN bus is a two lines communication. It consists of CAN-H and CAN-L. The first subsystem is a differential chip known as MCP2551. It takes the difference between CAN-H and CAN-L and produce one single. That single is then sent to the second subsystem, The Interrupter. The interrupter is STN1110. It is the world's smallest, lowest cost multiprotocol OBD-II to UART interpreter IC. It has the capability to interpret any protocol to universal asynchronous receiver-transmitter (UART). This is illustrated in fig 6 below. STN1110 provides an easy means of accessing vehicle data, including diagnostic trouble codes, MIL status, VIN, Inspection and Maintenance (I/M) information, In-use Performance Tracking (IPT), and hundreds of real-time parameters. for the ELM327 command set, while outperforming the original ELM327 IC in every category: stability, performance, and features. Finally, the interrupter is connected to the host through RX, TX ports.

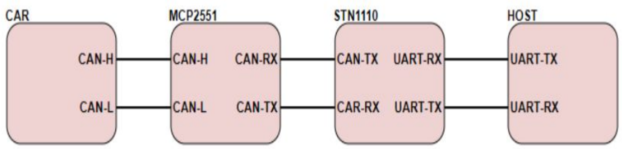


Fig. 6. OBD II system

The system is a message-based protocol. The user request specific information for the interrupter by sending a request of HEX values. The Host will then receive a message that has the requested information. OBD-II provides many modes. For now, we are only interested in live data. That is mode 01. Every request starts with 01 to indicate the request for live data. That 01 is then followed by the parameter value. After requesting a specific parameter. OBD-II will respond with another HEX value. That value start with 41 to indicate that it is live. The follow is values that can be translated to real life data using provided equations.

To have a proof of concept, different data was obtained from the car. Speed, RPM, Ambient temperature, and coolant temperature were some of the parameters obtained from the car. Table 2 shows data obtained by requesting specific parameters. The society of Automotive Engineers (SAE) provides equations to translate the data obtained to real live results.

PID	Tx	Rx	equation	result
Ambient Temperature	0146	41 46 3C	A - 40 °C	68 °F
Coolant Temperature	0105	41 05 7E	Value - 40	186.8 °F
RPM	010C	41 0C D1 CC	Value / 4	13427 RPM
Speed	010D	41 0D DE	-	66 km/h

Table 2. Data Requested and Received from OBD-II

D. Server Communication

Stable connection to a server is a very crucial aspect of the functionality of this project. The hub communicates with the Particle Cloud server using UDP (User Datagram Protocol), which saves a lot of data, as well as power. By not constantly requiring handshakes in order to keep the connection between the hub and cloud active, users are able to conserve their data and make sure that they have available data for when they need it. Acknowledgements are manually required when meaningful data is pushed, however, to ensure that the triggers are received.

When an event is published, a trigger is executed by IFTTT. If This Then That is a free service that can be integrated with Particle systems. In the case of our "TempAlert" application, the service will execute when the necessary prerequisites are met; the temperature is too hot, and movement is detected in the car. When our trigger is executed, users will receive a text message notifying them that their pet or child is in danger.

E. Power Distribution Systems

Our power distribution system is distributed into two different systems as specified earlier; the Hub and the Extension. These two systems can be seen in the block diagram in Figure 2. Each of these systems run off a different power supply and are vital in running our system.

The Hub is where a lot of our main electronics are located that have sensors on them as well as the microprocessor that sends data via 3G. Its power source is the pin 16 located on the OBD-II port, this pin is what powers the OBD-II devices used to get codes needed for diagnostics. Pin 4 is the chassis ground and pin 5 is the signal ground. This gets its power from the battery of the car which gets recharged when the car is on using an alternator. The voltage on this pin can change and be different at different times. This matters on how the alternator charges the battery and how much the battery can hold charge for [6].

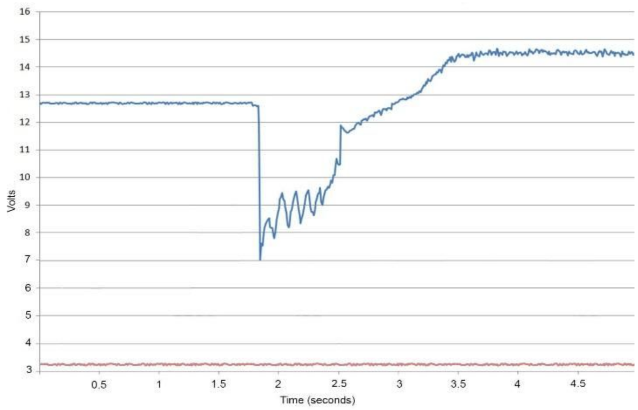


Fig. 7. Experimental Data for Car Battery Voltage Over Time

Figure 7 shows this very well; the pin starts a little less than 13 volts, as the car gets ignited it drops down to about 7 volts but then increases to 14.7 volts. This of course may be different for different vehicles. Another factor to consider is the temperature. As the temperature becomes smaller the battery's voltage falls 0.01 volts every 10 degrees Fahrenheit [5].

The Hub consists of three switching voltage regulators. One located on the hub that is used on the electron microcontroller that has a 3v3 pin that can output 800mA. The model name for it is called the TPS62291 which outputs 1 Amp, can take in 2.3-6 volts, output 3.3 volts, and is 96% efficient [1]. It is used as a pull up for the temperature sensor. The two other voltage regulators that are being used are a 5V and another 3.3V regulator. The 5V is used for some of the OBD-2 circuitry, and the 3.3 V is used for the Bluetooth/Wi-Fi Redbear duo, microcontroller, temperature sensor, as well as some of the OBD-II circuitry. The D24v22F3 [3] is the 3.3 V output voltage regulator that can output 2.6 Amps and take in from 4V to 36V. It has an efficiency of 85-95% [3]. The 5V regulator has yet to be bought and is currently being replaced by 5V linear regulator. The reason why we use another 3.3 voltage regulator is because the Redbear duo may need about 1 amp of current which the electron voltage regulator will not be able to give [4]. Another reason is because fairly economical to buy the 2.6 A regulator. One can add as up to 2.6 A of sensors to it if it is found necessary which we may do for other applications.

The Electron microcontroller can take a voltage of anywhere from 3.9 to 17 volts through its power management chip BQ24195 [7]. In the design for the IoTECH we use the Vin pin as a way to power our microcontroller. It also states in its specifications that a 470 uF capacitor is required in the Vin pin just in case if the electron needs to pull more power when sending 3g data so a 50V capacitor is used [2]. It is directly connected to pin 16 from the OBD-II port since it can handle 11-15 volts. There also is a 470uF 50V capacitor placed at the power source as a surge protection capacitor, just as a precaution of a 40 Volts surge that can happen on the vehicle's battery although this is unlikely. There is also a 50 volt capacitor located on in Vin pin of the 3.3V buck converter which will protect from any surges as well.

The OBD-II circuitry in total takes up to 876 mW of power and only requires 73 mA of current to use it, this is measured value we found. It uses a 3.3V and 5V power supply and as stated earlier will be supplied by the 2 switching voltage regulators that are going to be on the Hub.

Table 3 shows the Hub Temperature Alert Systems power consumption. This is used to power the application for this system. It consists of the Maximum power consumption, typical power consumption, and sleep power consumption. It then breaks down the sections into our microcontroller the Electron, sensors, voltage regulators and the total. It also specifies how long the car battery would be able to pull our system. The max power consumption is 18 to 22 hours of use (30.3W); this is if everything is going to be on all the time at absolute maximum clock frequencies, and so on. The typical power consumption hours is 163-200 hours (3.3W), this is what we can expect for typical use of the system. The sleep power consumption mode would cause the battery in the car to last 14516-18000 hours (37mW). All these numbers are found from datasheets of all the sensors and computations. When using actual measured numbers for our applications we found that The Hub would last us about 236 hours of battery life typically and would take 2280 mW-2850mW while Idling and taking temperature measurements.

Hub Temperature Alert System			
Max	**Max Power Consumption**	**Typical Power Consumption**	**Sleep Power Consumption**
Electron	27000 mW	2750 mW	30mW
Sensors	3040.95mW	500mW	4.7mW
Voltage Reg.	264 mW	55.5 mW	2.31mW
Total	30.3W	3.3 W	37.01 mW
Current at 12-15 volts	2.02Amps-2.525 Amps	227mA- 275mA	2.5 mA-3.1mA
Hours Battery would Last	18 Hours - 22 Hours	163 Hours-200 Hours	14516 Hours-18000 Hours

Table 3. Hub Temperature Alert System Power Consumption [2][3][4][12]

Our extension consists of the Red Bear Duo as well as one Boost Converter, a Camera and IR Motion Sensor. It is powered by a 4400mAh 3.7 volt li-po battery. It consists of 2 electrolytic capacitors. The 16V 100uF capacitor protects from surges to the Red Bear Duo microprocessor. While the 25V 47uF capacitor is used which is close to the recommended amount for the 5V boost converter. This is recommended because of LC voltage spikes protection [11].

The RedBear Duo works on a voltage between 3.5-8V therefore it is connected directly to the battery [8], while the camera and IR Motion Sensors work on 5 volts. This is why we use the Pololu 5V Step-Up Voltage Regulator U3V12F5 which outputs up to 1.4 Amps. Although our sensors only use up to 1.25 Watts, which is 337 mA, this is for the purpose of adding more sensors to the extension for other applications [11].

Looking at Table 4 we have the extension temperature alert systems power consumption. It consists of the boost converter, sensors, RedBear Duo and the total [8][9][10][11]. It then goes down and breaks up the battery life based on 2000 mAH and 4400mAH batteries. At first our extension was going to use a 2000mAH battery but then we decided to use a larger battery because it added hours to the battery usage. We found that if everything would have been working at a maximum with high clock frequencies then the battery would last 3.4 hours (4.8W). For typical use 14 hours (1.1W) and in sleep mode 35.2 hours (0.5W). We also decided to measure different instances of using sensors and the Redbear Duo and found that the battery could last anywhere from 8.6 to 30 hours mattering on usage. These were experimental measurements with different instances of one sensor on and another one off; The total power consumption being from 1887 mW-540 mW. This can be seen in Figure 8.

Extension Temperature Alert System			
	Max	**Typical**	**Sleep**
Boost Converter	155 mW	61 mW	59 mW
Sensors	1.25 W	0.55 W	0.4 W
Red Duo	3.4 W	0.56 W	5.2 mW
Total	4.8 W	1.1 W	0.5 W
Total Current (At 3.7V)	1.3 A	0.3 A	0.1 A
Hours Battery Would Last (2000mAH)	1.5 Hours	6.3 Hours	16 Hours
Hours Battery Would Last (4400mAH)	3.4 Hours	14 Hours	35.2 Hours

Table 4. Extension Temperature Alert System Power Consumption [8][9][10][11]

Measured Extension Power Instances								
Camera	ON	ON	ON	ON	ON	ON	ON	ON
IR Light	ON	ON	ON	ON	OFF	OFF	OFF	OFF
Low Power Camera Mode	OFF	ON	OFF	ON	OFF	ON	OFF	ON
IR Sensor	ON	OFF	ON	OFF	ON	OFF	ON	OFF
Bluetooth Transm.	ON	ON	OFF	OFF	ON	ON	OFF	OFF
Power (mW)	1831-1887	1653	1460-1520	1284	910-976	773	540-606	403
Battery Life (Hours)	8.6-8.9	9.84	10.7-11.1	12.6	16.6-17.8	21	26-30	40

Fig. 8. Extension Measured Power Instances

III. PROJECT MANAGEMENT

Deliverable	**Delivered?**
Power electronics circuit built	Yes
Ability to read OBD-II car data	Yes
Ability to read other sensor data	Yes
Breadboard prototype circuits	Yes
Display data from sensors	Yes
Extra: Wireless communication between hub and server	Yes
Extra: Implement IFTTT triggers	Yes
Extra: Ability to capture images using serial IR camera	Yes

Table. 5. MDR Deliverables

Referring to Table 5, our team was able to successfully complete all deliverables in addition to delivering some extra functionality for MDR. All circuits were breadboarded and all implemented sensors were successfully working. We were also able to successfully read OBD-II CAN bus data and send/receive commands. In addition, our group was able to implement wireless functionality with the IFTTT server and add in a working camera module.

Our two main software developers are Chris Ingerson (CSE) and Nigel Paine (CSE), working on implementing all of backend code necessary to communicate between devices (sensors, hub/extension, server, etc.). Their expertise will also revolve around data processing and interpreting sensor data in a way that is useful for the user. Our two main hardware engineers are Nicholas Korniyenko (EE) and Raghid Bahnam (EE). Their expertise involves circuitry and power management and are the main points of contact for any hardware-related issues.

Breaking down the roles further, Raghid's main area of expertise is OBD-II communication, whereas Nick's area of expertise is the power distribution. Chris will be working on the IoTECH hub while Nigel will be working on the IoTECH extension. Each of the team members work together to fill in where necessary and the workload should be pretty evenly distributed throughout with weekly meetings with our advisor and at least two meetings a week with the team.

As we have reached the halfway point in the senior design project, we have much left to do including designing our product with printed custom circuit boards, designing an enclosure, implementing power saving features in the sensor code, enabling wireless connectivity via Bluetooth/WiFi for hub/extension communication, and then putting it all together. Refer to Figure 9 for a Gantt chart demonstrating our outlook for CDR and our game plan for the next phase in the project and being one step closer to completion. Depending on progress by CDR we plan to also implement a mobile app and potentially implement additional applications.

CDR Timeline

	12/7	12/14	12/21	12/28	1/4	1/11	1/18	1/25	2/1	2/8	2/15	2/22	3/1
Website (Chris)	x												
Draft Report (Everyone)	x	x											
Humidity Sensor (Chris)			x	x	x	x	x						
WiFi Connectivity (Nigel & Nick)			x	x	x								
Bluetooth Connectivity (Nigel & Nick)						x	x						
MDR Report (Everyone)							x	x					
Merge OBD & Hub Breadboards (Raghid & Nick)								x					
Combine Sensor Code (Chris & Nigel)								x					
Work on Add'l Sensors (Raghid & Nick)									x	x	x		
Work on Add'l Software (Chris & Nigel)									x	x	x		
Design PCB (Nigel & Raghid)											x	x	
Schedule CDR (Nigel)													x
			WINTER BREAK										

Fig. 9. Executive plan for CDR

IV. CONCLUSION

In a conclusion, we had a successful MDR. We delivered twice as much as promised and we are on our way to finish our project on time. Surely, there will be problems that will arise, but with the progress we made this semester, we are confident that we will have adequate time and resources to provide a working design for FPR.

ACKNOWLEDGEMENTS

We would like to acknowledge our advisor Professor Jay Taneja for helping us work through planning our project, from inception to final product. Also a thank you to our evaluators Professor Baird Soules and Professor Michael Zink for helping us improve our device and offering advice on presentation and implementation.

REFERENCES

[1] Texas Instruments, “TPS6229x 1-A Step Down Converter in 2 x 2 DRV Package ,” TPS62290 datasheet, June 2007 [Revised – January 2016].

[2] “Particle,” Particle Datasheets Documentation | Electron datasheet.[Online].Available: https://docs.particle.io/datasheets/electron-(cellular)/electron-datasheet/#schematic. [Accessed: 20-Dec-2017].

[3] “Pololu 3.3V, 2.6A Step-Down Voltage Regulator D24V22F3,” *Pololu Robotics & Electronics*. [Online].Available: https://www.pololu.com/product/2857. [Accessed: 20-Dec-2017].

[4] Redbear, “Redbear/Duo,” *GitHub*, 10-Aug-2016. [Online]. Available: https://github.com/redbear/Duo/tree/master/hardware/datasheets. [Accessed: 20-Dec-2017].

[5] AA1Car, “Diagnosing A Car Battery That Runs Down,” *http://www.aa1car.com/*. [Online]. Available: http://www.aa1car.com/library/battery_runs_down.htm.[Accessed: 20-Dec-2017].

[6] “OBD II power when key not in ignition,” *Stack Exchange*. [Online]. Available: https://mechanics.stackexchange.com/questions/23047/obd-ii-power-when-key-not-in-ignition. [Accessed: 20-Dec-2017].

[7] “BQ24195 (ACTIVE) I2C Controlled 2.5A/4.5A Single Cell Charger with 5.1V 1.3A/2.1A Synchronous Boost Operation | TI.com,” *Texas Instruments*. [Online]. Available: http://www.ti.com/product/BQ24195. [Accessed: 20-Dec-2017].

[8] Redbear, “redbear/Duo,” GitHub, 11-Sep-2017. [Online]. Available: https://github.com/redbear/Duo. [Accessed: 20-Dec-2017].

[9] “LinkSprite JPEG Color Camera Serial UART Interface With Infrared,” LinkSprite, Nov-2010. [Online]. Available: http://www.linksprite.com/upload/file/1291522825.pdf. [Accessed: 20-Dec-2017].

[10] “HC-SR501 PIR MOTION DETECTOR,” Particle. [Online]. Available: https://docs.particle.io/assets/datasheets/electronsensorkit/HC-SR501.pdf. [Accessed: 20-Dec-2017].

[11] “Pololu 5V Step-Up Voltage Regulator U3V12F5,” *Pololu Robotics & Electronics*. [Online]. Available: https://www.pololu.com/product/2115. [Accessed: 20-Dec-2017].

[12]“Waterproof DS18B20 Digital temperature sensor extras,” *Adafruit*. [Online]. Available: https://www.adafruit.com/product/381. [Accessed: 20-Dec-2017].

[13]“ T-Mobile SyncUP DRIVE™ ,” *T-Mobile*. [Online]. Available: https://explore.t-mobile.com/t-mobile-sync-up-drive.[Accessed: 20-Dec-2017].

[14]“Snapshot From Progressive | Big Discounts For Good Drivers,” *Progressive*.[Online].Available: https://www.progressive.com/auto/discounts/snapshot/.[Accessed: 20-Dec-2017].

[15] https://www.automatic.com/pro/

[16] OBD Solutions “Multiprotocol OBD to UART Interpreter Datasheet’’ STN1110 datasheet Oct. 2010 [Revised July. 2012].

[17] Industries, A. (2018). Lithium Ion Polymer Battery - 3.7v 2500mAh. [online] Adafruit.com. Available at: https://www.adafruit.com/product/328 [Accessed 6 Feb. 2018].

[18] Pololu.com. (2018). Pololu 5V Step-Up Voltage Regulator U3V12F5. [online] Available at: https://www.pololu.com/product/2115 [Accessed 6 Feb. 2018].

[19] Arduino.cc. (2018). Arduino Reference. [online] Available at: https://www.arduino.cc/reference/en/ [Accessed 6 Feb. 2018].

[20] Arduino.cc. (2018). Arduino Reference - Serial. [online] Available at: https://www.arduino.cc/reference/en/language/functions/communication/serial/ [Accessed 6 Feb. 2018].

[21] “Hue White and color ambiance Single bulb BR30,” *Philips*. [Online]. Available: https://www2.meethue.com/en-us/p/hue-white-and-color-ambiance-single-bulb-br30/46677468941. [Accessed: 12-Feb-2018].

[22] “Your life with Nest,” *Nest*. [Online]. Available: https://nest.com/thermostats/nest-learning-thermostat/overview/. [Accessed: 12-Feb-2018].

[23]EngineersGarage, “CAN Protocol - Understanding the Controller Area Network Protocol,” *EngineersGarage*, 21-Mar-2017. [Online]. Available: https://www.engineersgarage.com/article/what-is-controller-area-network. [Accessed: 12-Feb-2018].

[24] Particle (2018) Particle Guides. [online] Available at: https://docs.particle.io/guide/getting-started/intro/core/. [Accessed 17 October 2017].

[25] HC Serial Bluetooth Products. Available at: https://cdn.makezine.com/uploads/2014/03/hc_hc-05-user-instructions-bluetooth.pdf. [Accessed 16 January 2018].

[26] If This Then That (2018). IFTTT. Available at: https://ifttt.com/. [Accessed 12 October 2017].