**DATE:** *October 23, 2013*

**TO:** *Vijay Garg*

*Xiao Pu*

*Laura Coco*

**FROM:** *Anusha Ayub*

*Nafiz Haider*

*Muhammad Danyal Hussain*

*Munsefar Khaleque*

*Pulok Khan*

*Mohammad Arsalan Nasir*

**SUBJECT:** *Requirement Specifications Report for Texting While Driving Project.*

**INTRODUCTION**

This report will provide an overview of the Problem Statement and then expand on the required specifications. Through research and additional brainstorming of the problem, we have provided greater detail of the operational specifications of our Prevent Texting While Driving system. The overall system will consist of a hardware sensing mechanism and a software application on the driver’s phone. The sensing mechanism will consist of an OBD-II connector attached to the car port and our sensing module. This module will include a CAN-BUS Shield, a Bluetooth Low Energy Shield, and an Arduino Leonardo microcontroller and will take power and data from the car. This data will be used specifically to determine if the car is in a “drive” state. Then, the module will send a signal through Bluetooth to the driver’s phone. This will be interpreted by the software app and will disable the texting application on the driver’s phone based on the car’s state. This system is specifically targeted for teenage drivers and will ideally have a parent or other third party member install the phone application and module. Currently, the project is not yet in implementation as we have worked to find a feasible solution. This report appropriately outlines our specifications and problem solutions, excluding specific detail of how we will create and program the hardware and software system. The contents of this report will include a design project background and a design requirements section.

**DESIGN PROJECT BACKGROUND**

The goal for our design project is to create a system that prevents smartphone users from texting while driving. By properly defining the problem, we distilled our engineering challenge into two parts - recognition and transmission. Along with solving these problems, our solution should fulfill fundamental requirements outlined by the the essential customer needs. These needs will also constitute metric points that can help our team gauge where our solution is in the design pipeline.

**Problem Definition**

The objective for our senior design project is to detect users that have the potential to text while driving and restrict them from doing so. Many of the challenges we face lie in the front and back end of the system, which are detection and prevention, respectively. A challenge we face is to identify robust indicators that reliably characterize a user with a smartphone in the driver’s seat. These indicators should be well-defined as to prevent false positive identification when analyzing the indicators. Another potential obstacle is isolating only the driver’s phone and leaving the use of other phones uninterrupted. The challenge lies in identifying a method that will differentiate the driver’s phone from other phones in the car. In order for our system to properly make decisions, it must verify the presence of the phone and access it to block the texting app from being used. The communication between our controller and the driver’s phone needs to be properly interfaced to establish a functional connection that can be used to block the texting app. We also need to make sure that the user is not able to bypass our system and text while driving. This requires our system to be secure enough to prevent the user from finding any loopholes that can circumvent our system.

**Customer Needs Analysis**

Our design needs to detect texting and prohibit the user from further use of the phone. To consider our solution effective and functional, the design must fulfill multiple customer needs. One of the customer constraints is size. The hardware module needs to be small enough to fit comfortably within a vehicle without causing inherent hazards. A reasonable size for our module would have a length of 4 inches, a width of 4 inches and a height of 2 inches so that it could fit under the dashboard and above the gas and brake pedals. Another requirement is for the module to be nonintrusive. The addition of our solution to the vehicle should not prevent the driver from operating the vehicle as intended (such as blocking the view of the road, preventing use of pedals, etc.).  Both the hardware and software aspects should have low power consumption in order to avoid draining the batteries of the car and phone, respectively. For our design to be considered “low power”, its voltage rating needs to fall within the range of 7 to 12 volts and have a maximum current draw of <100 mA. Any driver with a smartphone should be able to use our design. This requires our software to be cross platform and our hardware to interface with any vehicle. The budget laid out by Texas Instruments for us to design the solution is $1000 and our costs should not exceed that. The system should be functional, meaning that it should prevent the user from texting. Lastly, the application must be easy to install, use and maintain.

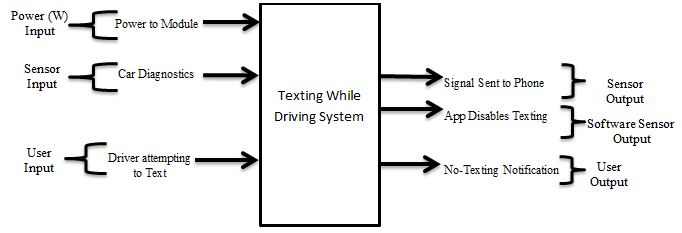
**Design Functionality**   
Using hardware and software components, our design needs to detect the presence of the driver’s phone, establish a connection with it, recognize when it's being used to text, and block the texting application from being used. The hardware needs to detect if the driver is texting. Our hardware implementation consists of an Arduino Leonardo, Bluetooth Low-Energy (BLE) Shield, and CAN-Bus Shield connected to the vehicles On-Board Diagnostics (OBD-II) Port using a DB9 to OBD-II cable. These components are interconnected through I/O pins by placing the CAN-Bus Shield onto the BLE Shield, which is then connected onto the Arduino. The dimensions of the CAN-Bus Shield are 2.67 inches in length by 2.08 inches in width by 0.5 inches in height [1]. The Arduino microcontroller has a length of 2.70 inches, a width of 2.10 inches, and a height of 0.6 inches for its dimensions [2]. The BLE Shield has dimensions of 2.6 inches for the length, 2.00 inches for the width, and 0.6 inches for the height [3]. Thus the overall dimensions when the Arduino and both the Shields are connected will be a length of 2.70 inches, a width of 2.10, and a height of 1.7 inches.  The ODB-II port is under the vehicle’s dashboard and above the gas and brake pedals and does not hinder the driver’s ability to safely operate the vehicle. The hardware module transfers the vehicle’s diagnostic data such as whether the car is on, vehicle speed, rpm, etc. from the vehicle’s OBD-II port through the DB9 to OBD-II cable to the CAN-Bus Shield. The CAN-Bus shield interprets the diagnostic data from the vehicle and relays it to the Arduino microcontroller which then can process the data to determine whether the vehicle is being driven or not.  By capturing real-time information from the vehicle, isolating the relevant data and processing it in a way that is useable by the software, we can transmit the data through Bluetooth to the driver’s smartphone using the BLE Shield. Using the BLE shield allows for a more low power solution compared to regular Bluetooth because BLE technology only transmits information when it is needed rather than transmitting continuously. The BLE Shield transmits the data signaling whether or not the car is being driven through Bluetooth to the driver’s smartphone. An application on the smartphone then takes the signal as an input to determine whether or not to block the smartphone’s texting capabilities. Our product will not prevent drivers from using their smartphones to access essential or emergency applications such as maps and phone calls.

**DESIGN REQUIREMENTS**

This section will detail the specific criteria that our design will meet in order to successfully prevent the driver from texting. The subsections specifically detail the inputs and outputs that the system will deal with, the user interface that the driver (a typical teenager) and the installer/customer (such as a parent) will interact with, key environmental specifications that the system must operate under, and key specifications and test criteria that the design must meet.

**Inputs and Outputs**

The following figure is the input and output diagram for our system. The sensor system input and outputs consists of car diagnostic information sent into the driving system and a Bluetooth signal sent to the phone, leading to disabling of the texting application if driving diagnostics is determined. The user input and output consists of the driver attempting to open his or her texting application but failing, as a no-texting while driving notification appears. In addition, power from the vehicle is inputted to the module to turn on our system. Following this figure, there will be two subsections that specifically detail the inputs and outputs.

**Figure 1 Input Output Block Diagram- Prevent Texting While Driving**

The black box within Figure 1 can be broken down to better describe our system. There is an OBD-II cable connection from the car to the hardware module, which supplies power and car diagnostics data from the car to the module [4]. The sensor contains a BLE Shield that sends a signal, which is processed by the Arduino microcontroller, to the driver’s phone [3, 5]. Then, the app we create and install within the phone interprets the data and disables texting on the driver’s phone if he or she is determined to be driving. If the driver attempts to text, a notification will appear on his/her phone stating that texting has been disabled.

***Input Specifications***

Power Input:

1. Power to module- This input is in Watts. Power provided to the Can-Bus (and rest of module) from the car through the OBD-II cable. The Arduino Leonardo board requires 6-20 V range of input voltage and each pin can receive a maximum of 40 mA [5]. This will directly relate to the power the module could operate under.

Sensor Input:

1. Car Diagnostics- This is an input of data. Car Diagnostics (such as speed) passed to the Arduino for data processing from the car through the OBD-II cable. The CAN nodes of the CAN-Shield transfer data at a baud rate of 1 megabit/second [6]. This defines how fast diagnostics data will be sent to the arduino board for processing.

User Input:

1. Driver Attempting to Text- This is a user-provided input. The driver tries to send a text while driving (the texting should be turned off).

***Output Specifications***

Sensor Input:

1. Signal sent to phone- This is a digital output signal. The BLE Shield has Bluetooth and will propagate Bluetooth signal to all phones within the car, and this signal is to be interpreted by the (driver’s) phone with the built-in app. The interpretation will determine whether the user should be allowed to text or not. Uses same 2.4 GHz frequency band as regular Bluetooth, can operate up to 150 meters (but more typically around 15 meters or 50 feet ), and can transmit a data rate of 1 Mbps [3]. Consumes low power and is ideal for our system due to this.

Software Input:

1. App disables texting- This is a software-provided output. The application processes the Bluetooth signal sent by the user and disables the texting function of the driver’s phone if the driver is in a driving state (as determined by the Arduino.) This should keep the texting application disabled until the car is out of a driving state.

User Input:

1. No-Texting Notification- This is a user Output. The driver will be unable to text while the car is in a state of driving (5 mph or greater speed) and will receive a notification message on his/her phone when attempting to access the texting application.

**User Interface**

The hardware component of our prototype requires a one-time installation which would be done by parents or some sort of administrator. The design would need to be connected to the OBDII port located generally under the dashboard of the car. In addition, parents would have to download the required software app onto their teenager’s smartphone. Once they’ve downloaded the app from the app store and opened it, they would immediately be asked to set a password. This password would be used later to prevent teenagers from disabling the app. After the installation process and software download, there will be no further need for administrators to interact with the prototype.

Teenagers would have minimal interfacing with the prototype due to our goal of preventing any sort of interaction with their phones while driving. If some teenager is hoping to rid of their texting habit on their own merit, then they would take the steps that were mentioned earlier on their own. If the driver were to attempt to text, the texting app would issue a warning and phone would then block the app.  In the case that a teenager would attempt to uninstall the app, they would be required to input a password which the parent or other such administrator would only have access to, therefore preventing the teenager from halting the app.  Once the driver has reached their destination, our prototype would read that the car is not in motion and unblock all texting apps. There is no need for either parents or teenagers to access the prototype’s app other than the first time it’s downloaded.

**Key Environmental Specifications**

The system must work within the environment and operating ratings of a car. The sensor module will be powered on by the car itself, and thus the car malfunctioning is out of the scope of our specifications. The hardware prototype needs to fit below the dashboard, specifically having less than the width and length of the dashboard and the height nothing more than 4 inches so as to not interfere with leg room for the driver.

All of the hardware components support a temperature range of -40 to 85°C which means our design solution must operate between those temperatures [1, 2, 3]. Unlike the temperature range, the voltage range that the design solution can operate at varies by the individual devices. The Arduino microcontroller and the CAN-bus shield function between 2.7 and 5.5V and the BLE shield functions between 1.9 and 3.6 V which is all outlined in Table 1 [1, 2, 3]. Our prototype will only work between 2.7 and 3.6V which will be taken care of by the OBDII cable. The car’s battery will provide more power than needed, but the cable will supply the appropriate amount of power to the system which will prevent it from burning out. Refer to the table below for specific environmental requirements of the individual hardware components.

**Table 1. Component-Based Environmental Specifications**

|  |  |  |
| --- | --- | --- |
| Hardware Component | Operational Temperature Range | Operational Voltage Range |
| CAN-bus Shield | -40ºC - 85ºC | 2.7V – 5.5V |
| Arduino Leonardo Microcontroller | -40ºC - 85ºC | 2.7V – 5.5V |
| BLE Shield | -40ºC - 85ºC | 1.9V – 3.6V |

PULOK’s PART GOES HERE

**CONCLUSION**

This report described the design background and requirements of our text prevention project. Its purpose was to provide greater detail than the problem statement, as we have chosen an implementation method as well as specific hardware components for our project. Our system will consist of a hardware sensing mechanism which will use an Arduino microprocessor connected to the driver’s car to determine the state of the car. When a driving state is determined, the board will use BLE to communicate with the driver’s phone via an application that we have created to disable texting. This information allows us to begin building our system, and leads to the next step of more detailed design and testing. We look forward to continuing our research and building our prototype.

**REFERENCES**

[1] Microchip Technology Inc. (2007, Nov.). “MCP2515 CAN Controller” [Online].

Available: <https://www.sparkfun.com/datasheets/DevTools/Arduino/MCP2515.pdf>

[Accessed Oct. 21, 2013].

[2] Atmel Corporation. (2010, Nov.). “ATmega16U4 Microcontroller” [Online].

Available: <http://www.atmel.com/Images/7766s.pdf> [Accessed Oct. 21, 2013].

[3] Nordic Semiconductor. “nRF8001 Bluetooth Low Energy Solution” [Online]. Available: <http://www.nordicsemi.com/eng/Products/Bluetooth-R-low-energy/nRF8001#Download> [Accessed Oct. 20, 2013].

[4] Huang, Stanley. *Connect Arduino to your car through OBD-II* [Online].

Available: <http://arduinodev.com/connect-arduino-to-a-car-through-obd-ii-port/>

[5] Arduino. (2013) *Arduino Leonardo* [Online].

Available: <http://arduino.cc/en/Main/arduinoBoardLeonardo>

[6] Axiomatic Global Electronic Solutions. (2006, July 6). *Q&A – What is CAN?*[Online].

Available: <http://www.axiomatic.com/whatiscan.pdf>

[7] Halleck, Thomas. (2013, June 29). *What is BLE? How Bluetooth Low Energy In Android 4.3 Could Revolutionize The Smartwatch* [Online] Available: <http://www.ibtimes.com/what-ble-how-bluetooth-low-energy-android-43-could-revolutionize-smartwatch-1328743>

[8] *IEEE Standard for Telecommunications and Information Exchange Between Systems- LAN/MAN*, IEEE Standard 802, 2002.

[9] *IEEE Standard Microcomputer System Bus*, IEEE Standard 796, 1983.

**APPENDIX A: APPLICABLE STANDARDS**

1. LAN/MAN Standard: “The lower transport layers [(Logical Link Control and Adaptation Protocol (L2CAP), Link Manager Protocol (LMP), baseband and radio] of the Bluetooth™ wireless technology are defined. Bluetooth is an industry specification for short-range radio frequency (RF)-based connectivity for portable personal devices. The IEEE 802.15.1 Task Group has reviewed and provided a standard adaptation of the Bluetooth specification (version 1.1) medium access control (MAC) (L2CAP, LMP, and baseband) and physical layer (PHY) (radio). Also specified is a clause on service access points (SAPs), which includes a logical link control (LLC)-MAC interface for the ISO/IEC 8802-2 LLC. A normative annex is included that provides a Protocol Implementation Conformance Statement (PICS) proforma, and a informative high-level behavioral ITU-T Z.100 specification and description language (SDL) model for an integrated Bluetooth MAC sublayer are also specified [8].”
2. Microcomputer System Bus Standard: “A general-purpose microcomputer system bus is defined, and the device-independent electrical and functional interface requirements that a module shall meet in order to interconnect and communicate unambiguously by way of the system are specified. Signal definitions and timing and electrical specifications are covered in detail for users who evaluate or design products that will be compatible with the IEEE Std 796 system bus structure. Only with the interface characteristics of microcomputer devices are covered; design specifications, performance requirements, and safety requirements of modules are omitted. The use of the standard will enable independently manufactured devices to be connected into a single functional system, permit products with a wide range of capabilities to be introduced to the system simultaneously, and result in a system with a minimum of restrictions on the performance characteristics of devices connected to the system. You will receive an email from Customer Service with the URL needed to access this publication online [9].”