OBD Bluetooth Interface and Android App

Project Proposal Document

Patrick Landis (pal25)
Schuyler Thompson (sdt16) (Current Team Lead)
Advisor: Swarup Bhunia, PhD

January 28, 2013

Executive Summary

A user with an Android based phone will be able to read and interpret diagnostic and operational data from their 1996 model year or newer car. On some models, they will be able to control some aspects of the car, such as starting the engine or rolling down the windows. This will be accomplished by having their Android device pair with a OBD (On-board Diagnostic) interface that is connected to the car over bluetooth, using a secure channel.

1 Technical

1.1 Problem Description

Every car in the United States since 1996 is required, by law, to contain on-board diagnostics. This standard, codified in SAE J1939 as on-board diagnostics II (OBD-II). The OBD-II specification officially requires a few different things. The OBD-II specification requires that all cars make available data link connector and also place restrictions on where they can be placed. The specification defines a set of standard diagnostic trouble codes, the freeze frame (which is sort of like a black box), and requirements for MIL lighting (which deals with the check engine light). In addition it also defines specific terminology for emissions controls components, defines diagnostic trouble codes. A device is then plugged into the data link port to read messages over various protocols implemented by the car manufacturer.

Currently, technicians must manually plug in a device into this port in order to read OBD-II messages. Furthermore the current solutions only have basic functionality and are primarily for technicians. Our project aims to add wireless functionality for OBD-II via Bluetooth to be interfaced with Android compatible devices. Furthermore this project aims to make communication cryptographically secure. By allowing for cryptographically secure wireless OBD-II messages to be transferred to fairly ubiquitous Android-enabled devices this project is geared to enthusiasts and curious electronic hackers.

1.2 Project Specifications

This project has quite a few objectives because there are quite a few parts to it. This section will refer to Appendix I, a diagram which shows the information exchange, to clarify things.

1.2.1 OBD-II Interface Component

- This component requires a circuit which will have the functionality to interface directly to a car's OBD-II port.
- This component is required to plug into the data link connector specified in the OBD-II specification SAE J1962
- This component will need to be able to act as a transceiver with the OBD-II port.
- The component will also be responsible for encoding and decoding messages for each of the protocols allowed in the OBD-II standard which are:
    1. SAE J/1850 (PWM/VPW)
    2. ISO 15765-2 / SAE J2284 (CAN)
    3. SAE J2411 (Single Wire CAN)
    4. ISO 9141-2/14230-4 (Keyword Protocol)
    5. Ford DCL (Mostly Depreciated)

- This component will be required to communicate with the central microcontroller in via UART.
- This component will be powered from the OBD-II port.
- This component will be responsible for buffering messages until they can be sent to the MCU.

1.2.2 Bluetooth Interface Component

- This component will be required to adhere to the Bluetooth 2.1+ specification.
- This component will be required to communicate with the central microcontroller in via UART.
- This board will be responsible for communicating via Bluetooth to Android enabled devices.
- This component will be powered from the OBD-II port.
- This component will be responsible for setting up Bluetooth communications with Android enabled devices.
- This component will be responsible for negotiating the shared key for cryptography purposes.

1.2.3 Central Microcontroller

- This MCU must be able to communicate with the OBD-II interface component via UART.
- This MCU must be able to communicate with the Bluetooth interface component via UART.
- This component will be powered from the OBD-II port.
- MCU will be used to control message flow of the system.
- The MCU with be required to store a list of recent pairing keys to allow Android devices to auto-connect.

1.2.4 Bluetooth Functionality Android-side

- There will be writing an Android app to receive data from the OBD reader.
- It will use the Android SDK to implement the bluetooth functionality.
- The app will be able to display all of the basic messages from the diagnostic part of OBD.

- This includes the fault codes, monitor readiness, and whether the malfunction indicator light (MIL) is on or off.
- The OBD fault codes follow a pattern of N0000, a letter followed by 4 numbers. The app will contain the database of all of the standard codes and their meanings for easy lookup.
- The monitor readiness is indicating whether the all of the monitors used by the OBD system are ready to make a reading. They are reset when the OBD system is reset, and if the monitors are set to not ready, the car will fail any emissions check performed by a DMV.
- The MIL is the light commonly known as the check engine light.
- The app will use the Android Holo theme, and will follow the generally accepted design principals of Android UI design.
- The app will pair with one dongle at a time.
- The app will work with Android 4.1.

1.2.5 Cryptographically Securing Messages

This section doesn't describe any specific component of the system. This section describes functional requirements for the overall system to cryptographically secure wireless messages. The basis of this approach is to use Bluetooth's Secure Simple Pairing for security. This pairing protocol is mandatory for all BT 2.1+ devices.

- Will use "Just Works" form of pairing.
- The pairing key will be stored on the non-Android side of communication in the MCU.
- Both devices must acknowledge the pairing request.

1.3 Project Work Strategy

All of the above components need to be designed and built.
Technical tasks that need to be done:

- Design OBD interface board
- Design layout of Android app
- Design bluetooth board and interface
- Pair with bluetooth module, send and receive data
- Interpret data, display basic info about received data
- Complete user interface: the app will work according to the spec

- Implement OBD board according to schematic
- Implement bluetooth board according to schematic
- Interface the boards, program the microcontrollers

1.4 Verification and/or Testing

1.4.1 OBD-II Interface Component

Because OBD-II simulators are expensive a live car will need to be used for the functional testing of signals pertaining to the OBD-II port. Below is a list of tests that will need to be completed to verify that the OBD-II interface board is properly working.

- Any regulator circuits will be bench-tested. The OBD-II port supplies power from a car's battery and is usually fuse-protected with the cigarette lighter (usually 10-15A fuse). The regulators will be tested to make sure the regulators are properly supplying power at 5V and/or 3.3V from 12V bench supply.
- To ensure that signals are being properly parsed, any messages received will be measured against a known functional "scanner tool" used by technicians.
- This interface component will be tested on different vehicle brands to test the different communication protocols.

1.4.2 Bluetooth Interface Component

- The bluetooth module is controlled though the microcontroller, so the testing will be done with that.
- If there is an electrical problem, the specification sheet contains all of the information needed to troubleshoot.

1.4.3 Central Microcontroller

- The code will be unit tested for all functionality.
- There will be a test program for sending basic bluetooth commands. This will allow easy determination if a problem is bluetooth related or higher level than that.

1.4.4 Bluetooth Functionality Android-side

- The code will be unit tested before it goes on the device to verify basic functionality.
- The Android emulator included in the Android SDK will be used to test the user interface as the software is being developed.
- An Android VM (separate from the emulator) will be used to test the bluetooth functionality. The VM is harder to work with, but the emulator does not support testing bluetooth.

- There will be a separate test within the app to test the bluetooth application level protocol. This will be useful for debugging, in order to quickly verify if issues are protocol related or another issue.

1.4.5 Cryptographically Securing Messages

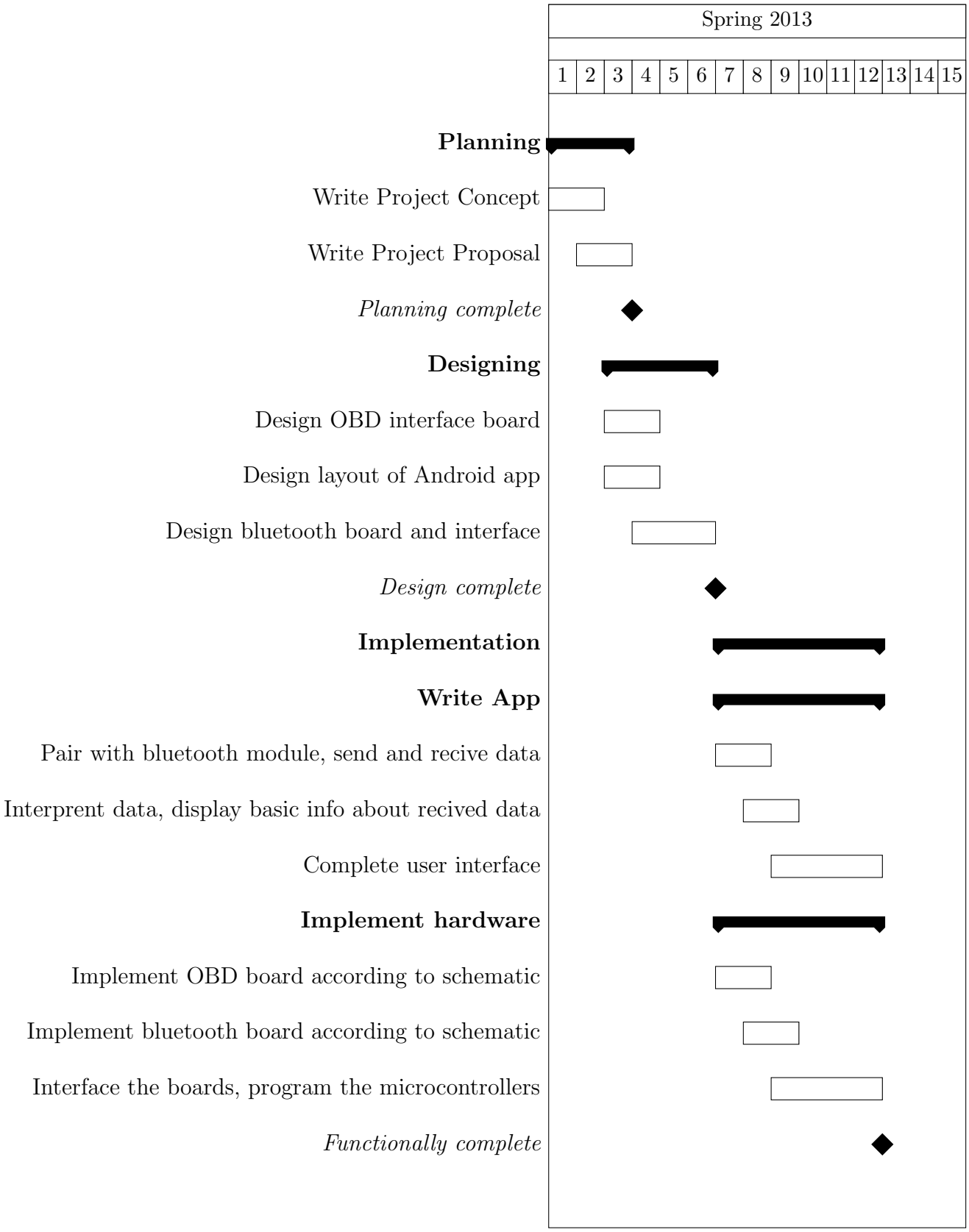
- This is a feature of the bluetooth standard, so if the devices pair, the Android phone will indicate if the communication is secure.
- A second phone will be used to verify that another phone cannot be paired without the permission of the permission of the board bluetooth module.

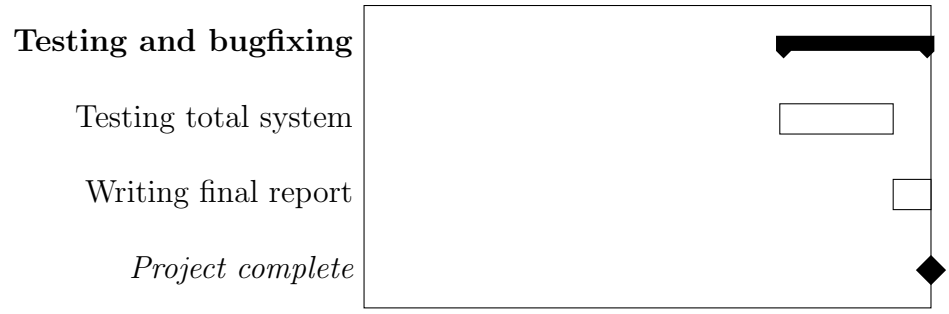
2 Management

2.1 Team Complement

- Patrick Landis (pal25): Hardware development/verification and microcontroller programming
- Schuyler Thompson (sdt16): Software development/verification and microcontroller programming

2.2 Project Management Plan





2.3 Budget

Item	Description	Price ($ USD)
STN1110	OBD-II Interface core	0.00 (Free samples)
Microchip RN-41	Bluetooth stack implementation on IC	19.90
Atmel ATMEGA328	Not sure exactly which AVR core	0.00 (Free samples)
Scanner Tool	To test/validate OBD messages	15.00
PCB Fabrication	If needed, paid by circuits lab	0.00
Misc. Components	If needed, paid by circuits lab	0.00
Total		19.90

3 Appendicies

3.1 Appendix I

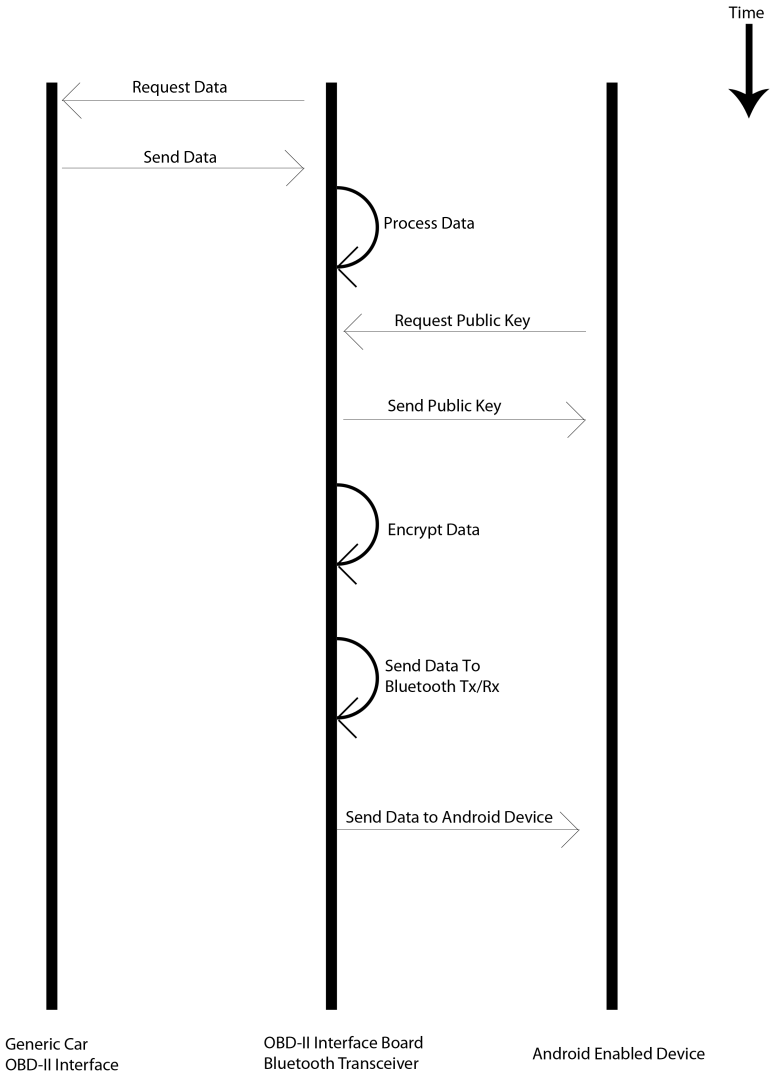


Figure 1: Information Flow Diagram