CORNERSTONE
MINNESOTA STATE UNIVERSITY MANKATO

Journal of Undergraduate Research at Minnesota State University, Mankato

Volume 2 | Article 1

2002

Formula-SAE Wireless Data Logger

Eric Holland
Minnesota State University, Mankato

Formula –SAE Wireless Data Logger April 27, 2002
Eric Holland Revision 2.00

Table of Contents

DOI: 10.56816/2378-6949.1179

I. Introduction

A. Abstract

The Formula-SAE (Society of Automotive Engineers) race car requires an onboard data logger to obtain and record the performance information received from the Fuel Injection Controller. This data is needed for analyzing the engine performance of the race car. It is advantageous to have this unit transmit the data via a wireless link to a handheld monitoring tool. Having user interface, like a LCD screen and a keypad, would make the monitoring tool easy to use. Also making the monitoring tool communicate with a PC would offer the beneficial features of saving and printing data. By having the PC connectivity, performance table updates in the Fuel Injection Controller could also be achieved. This will allow technicians to make changes to the engine performance more efficiently during the testing phases of the Formula-SAE race car.

B. Scope

Currently the Automotive Engineering students working on the Formula-SAE car do not have access to a wireless data logger. In the past a MOTECH programmable Fuel Injection Controller was used. This unit would be installed on the car and then a basic performance table would be uploaded to it. The car would then be test driven around and returned back to the pit crew; where a PC would be connected to the MOTECH unit via a serial RS-232 cable. The sensor data would be downloaded to the PC and analyzed. Then a new performance table would be created based on the sensor data and uploaded to the MOTECH unit. This is a very time consuming and tedious job; of upload data, stop car, load sensor data to the computer, analyze data, create new tables, upload new table to the MOTECH unit, and retest engine performance.

With the Wireless Data Logger proposed in this document, the technicians would be analyzing the sensor data, creating new performance tables, and uploads the new tables to the Fuel Injection Controller all while the car is being test driven. No stopping of the car or cables will be required while setting up the Fuel Injection Controller. This will greatly enhance the efficiency of setup time on the race car before each race.

Cliff Braunesreither, a Computer Engineering Technology student, has been working on designing a new Fuel Injection Controller for the Formula-SAE design team. I have been working with Cliff in developing an interface between his Fuel Injection Controller and the Wireless Data Logger. The Data Logger will retrieve, store, and transmit sensor data from the Fuel Injection Controller to a Monitoring Tool via a wireless connection. This Monitoring Tool will have a 20x4 character LCD screen and several buttons used to navigate through several menus and options. The monitoring tool will be handheld; battery powered, and will be able to communicate through a serial port to a PC.

II. Design Description

A. Philosophy and Tradeoffs

The main philosophy behind the design on the Wireless Data Logger was, ease of manufacturing. The design of the two pieces, Data Logger and Monitoring Tool, are so similar that one schematic is used to represent both of them. One PCB layout will be done for the two pieces. The components that are not shared by both pieces will only be populated on the corresponding board. These parts are noted on the schematic. Designing this way makes manufacturing the Wireless Data Logger easier because, only one schematic is needed, and only one PCB layout is needed for both the Data Logger and Monitoring Tool. Designing this way also makes the product less expensive.

The core of the two pieces, Data Logger and Monitoring Tool, will be Axiom's CME11E9-EVBU. This is a single board computer utilizing Motorola's 68HC11E9 8-Bit Microprocessor. The single board computer has the following features: 68HC11E9 processor, 8K EEPROM, 32K SRAM, RS-232 Hardware, LCD Screen Hardware, and Power supply voltage regulator. By using an existing single board computer verses designing one, allows more time to be spent on software development and keeps the complexity of the project to a reasonable level.

The Data Logger and Monitoring Tool will each have a CME11E9-EVBU in it. Different software will be written for each of these pieces, giving each piece its own functionality. A PCB board with all the peripheral hardware will be included in each piece. This board will sit on top of the EVBU board and plug into the header strips provided on Axiom's EVBU board. All the software for the EVBU boards will be written in Assembly language. This was chosen over C, because assembly is more compact and takes up less space in memory. This is important because of the limited application code memory, 8K EEPROM, the EVBU provides.

The peripheral hardware board was designed with two things in mind: Flexibility, and having a single board manufactured. The design of the two pieces, Data Logger and Monitoring Tool, are so similar that one board is used to represent both of them. Many zero ohm resistors were placed on the schematic and PCB to allow changes to be made to data lines and power lines easily. Test points are placed at key locations across the board. This will allow ease in attaching fly wires for testing and the rerouting of signals if necessary.

B. Block Diagrams

These are the functional block diagrams made for the Wireless Data Logger.

Figure #1: Concept Flow Chart

Figure #2: System Breakdown

Figure #3: RF Receiver Modules

Figure #4: RF Transmitter Modules

Figure #5: Keypad Circuitry

Figure #6: Automotive Buffer Circuitry

Figure #7: Power Supply and Protection Circuitry

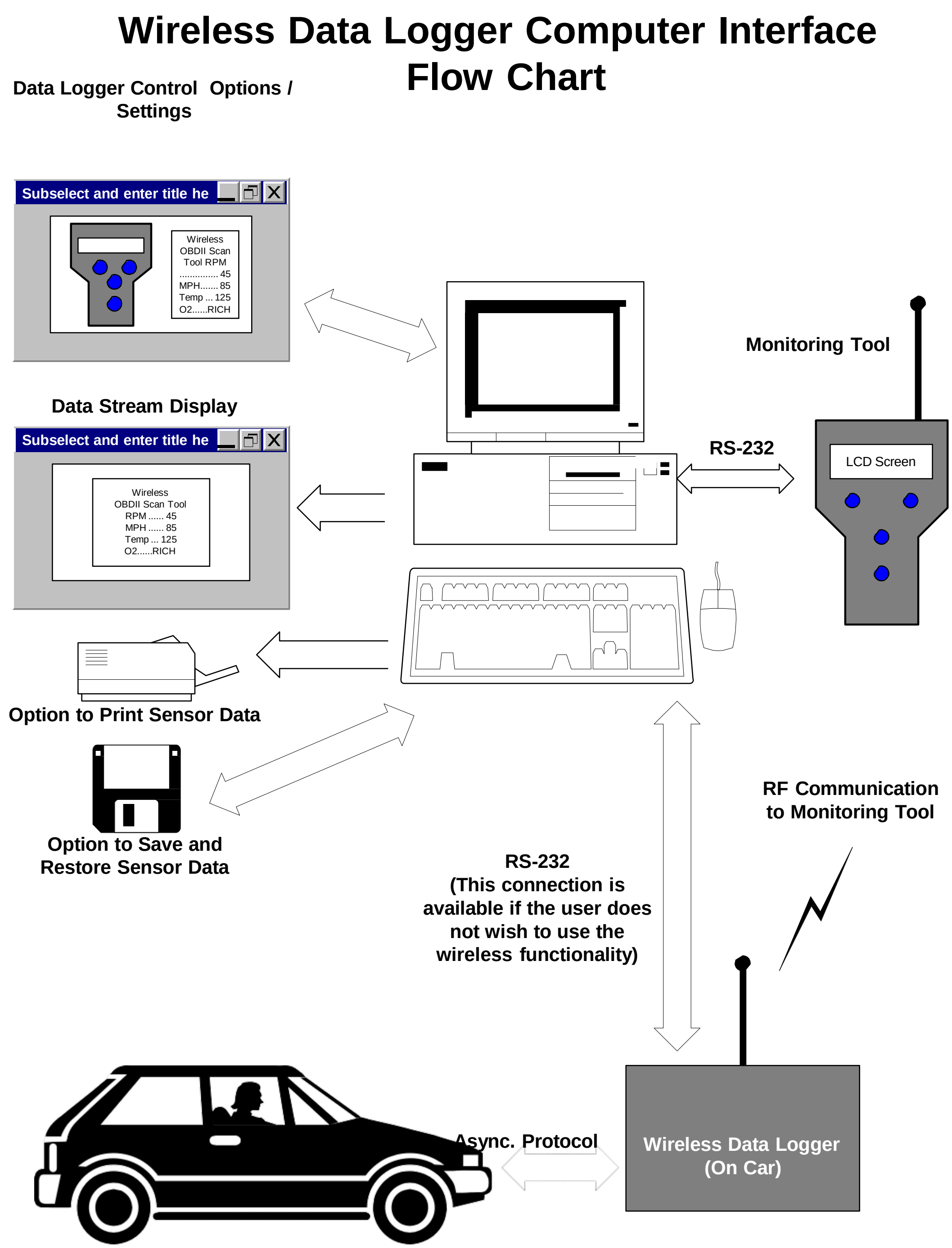


Figure #1 - Concept Flow Chart

This chart describes the flow of information from the car to the Data Logger, then to the Monitoring Tool. This chart also shows the Computer RS-232 access points.

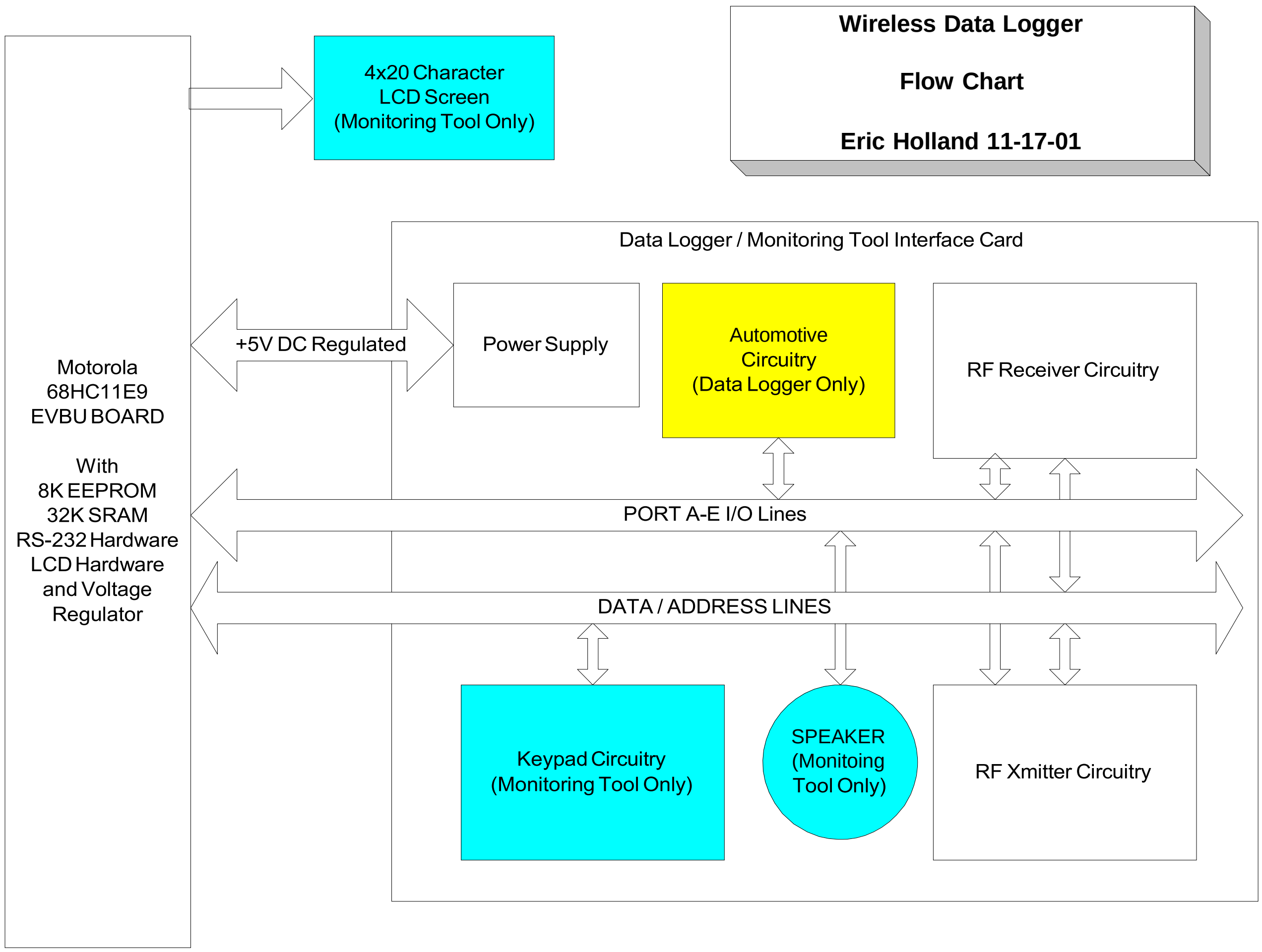


Figure #2 - System Breakdown

This chart shows the flow of information and power from the EVBU to the Interface card.

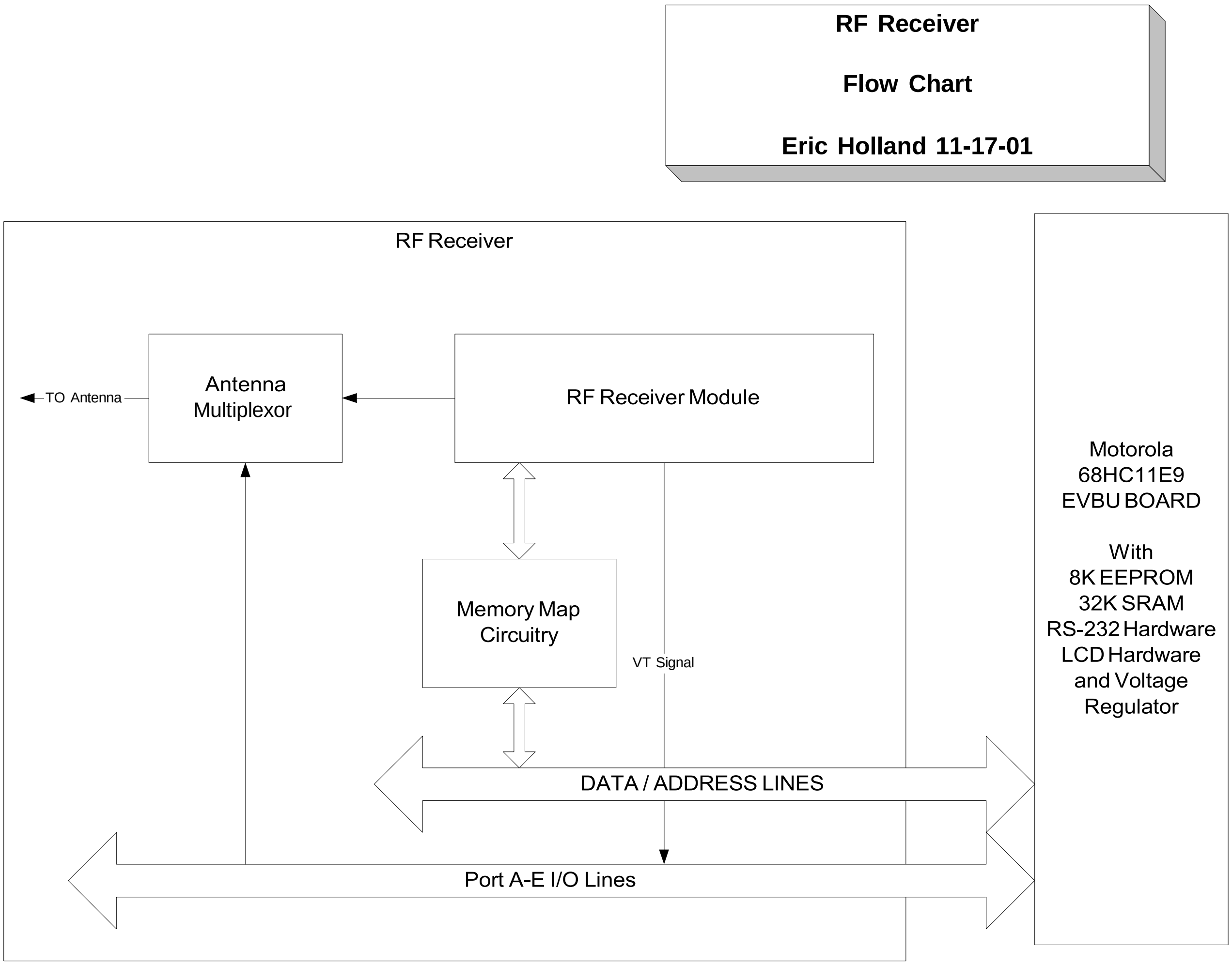


Figure #3 - RF Receiver

This block is on both the Data Logger and the Monitoring Tool. This block receives the RF signal from the antenna and translates the serial information in to a parallel format that can be sent to the microprocessor via the data bus.

DOI: 10.56816/2378-6949.1179

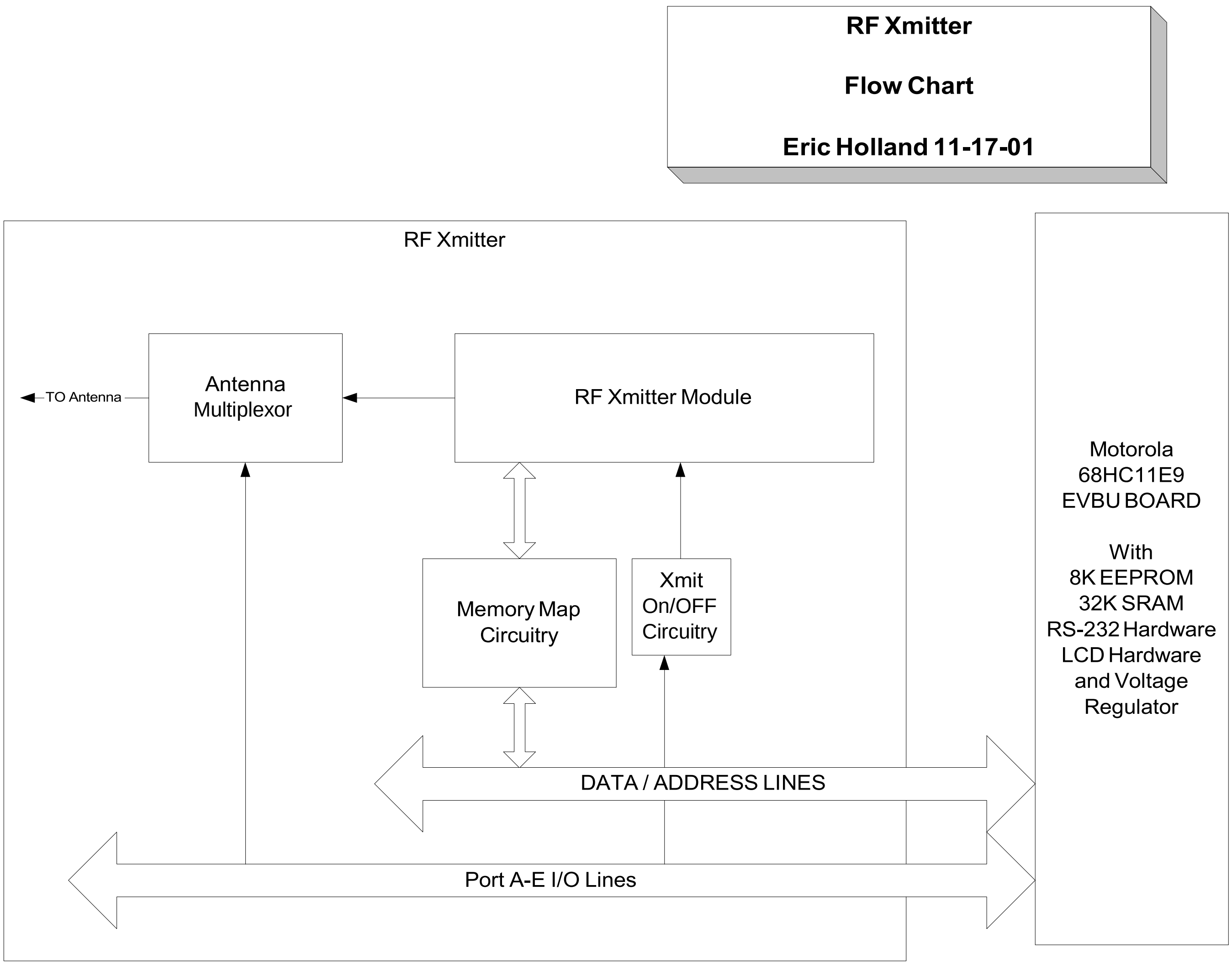


Figure #4 - RF Transmitter

This block is on both the Data Logger and the Monitoring Tool. This block takes the parallel data from the data bus and translates it into a RF signal and then transmits that signal over the antenna.

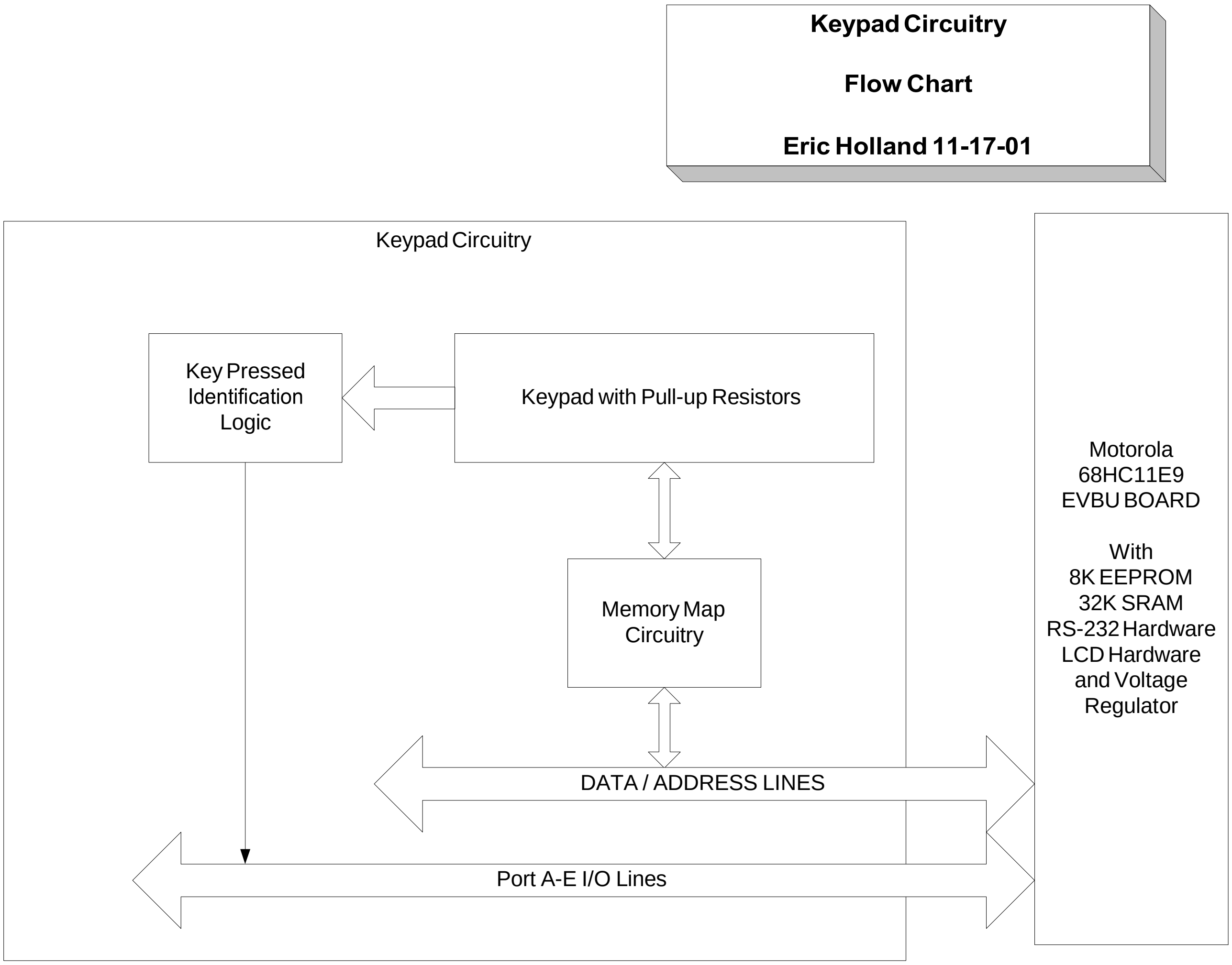


Figure #5 - Keypad Circuitry

This block is only on the Monitoring Tool. This block receives the input from the keypad and translates that into a parallel data format that can be sent to the microprocessor via the data bus.

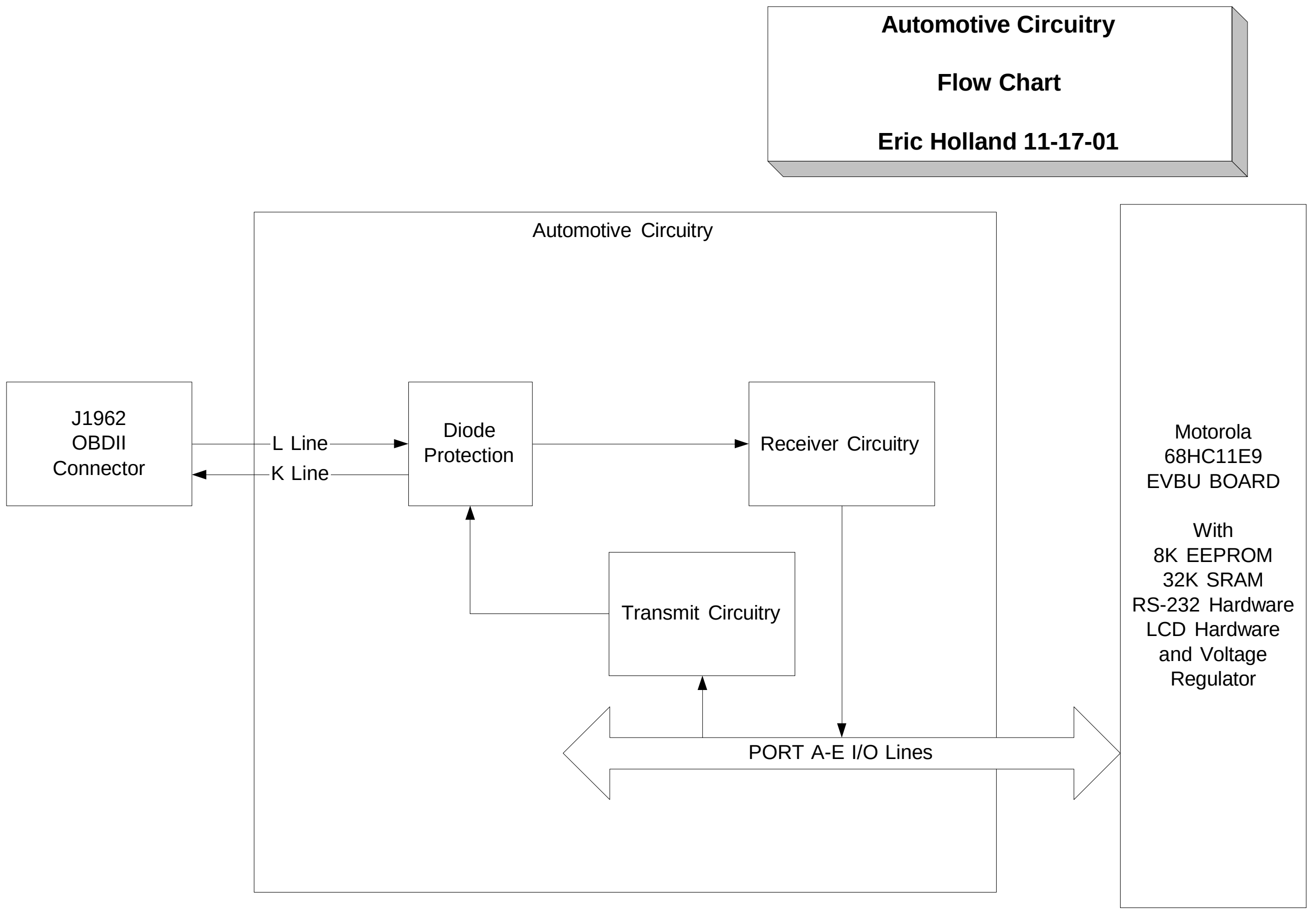


Figure #6 - Automotive Circuitry

This block is only on the Data Logger. This block receives and transmits data from the microprocessor to the Fuel Injection Controller.

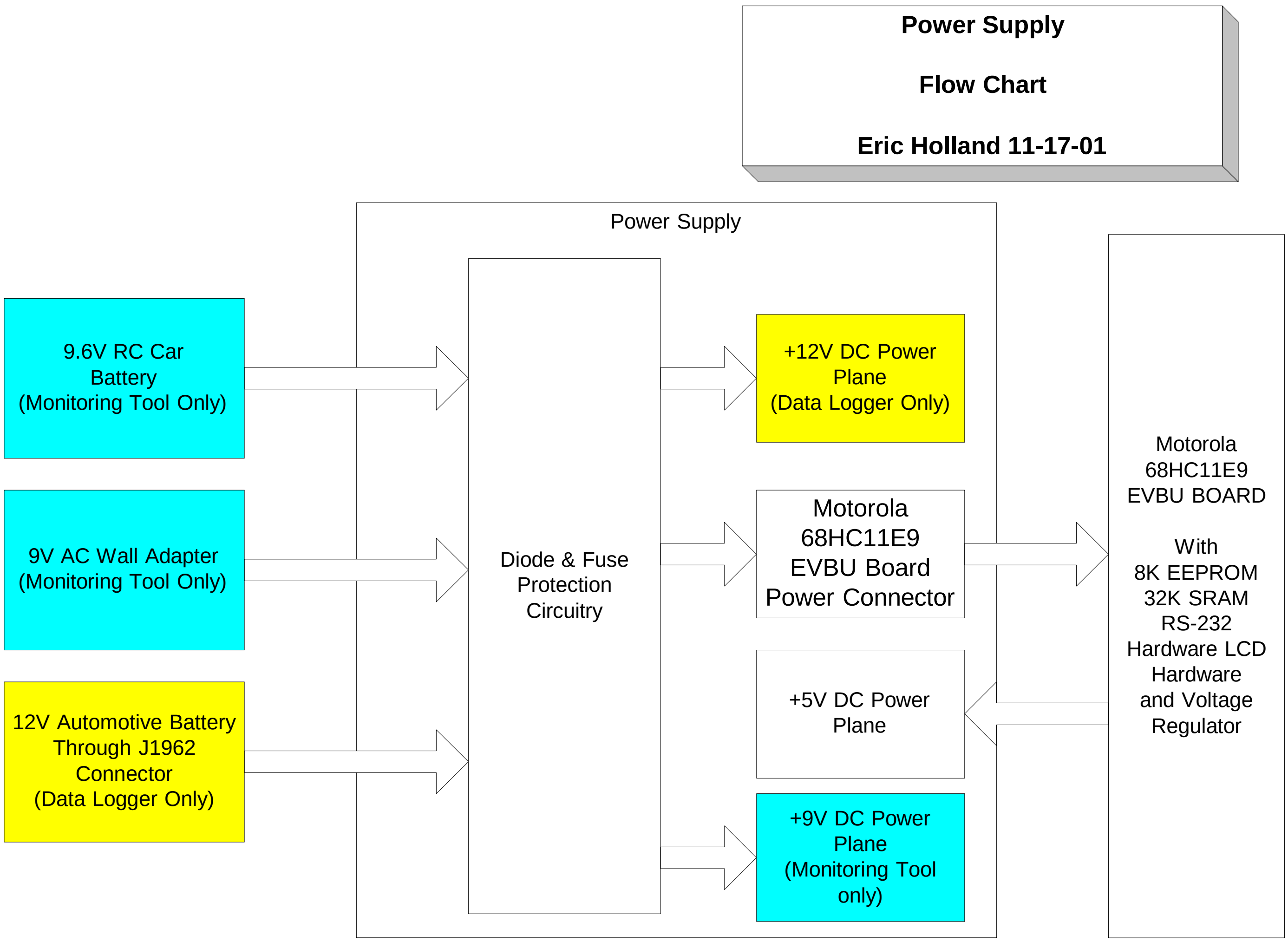


Figure #7 - Power Supply

This block is on both the Data Logger and the Monitoring Tool. This block receives power from either 9.6V Battery (on Monitoring Tool), 9V Wall Adapter (on Monitoring Tool), or 12V Automotive Battery (on Data Logger) The power is then regulated and connected to the appropriate power plane.

DOI: 10.56816/2378-6949.1179

C. Overview of Implementation

This section describes the Functional Block diagrams and describes the circuitry used to implement these blocks.

RF Receiver

This block is on both the Data Logger and the Monitoring Tool. This block receives the RF signal from the antenna and translates the serial information in to a parallel format that can be sent to the microprocessor via the data bus.

Antenna Multiplexor

This block allows both the transmitter and receiver to use the same antenna, by time multiplexing. The circuit used to implement this block contains 2 MOSFETS and two inverters. One inverter (U27C) is used to buffer the RADIOCON signal from the microprocessor. The other (U27B) is to invert the RADIOCON signal. When the RADIOCON signal is logic low the Transmitter will be connected to the antenna via the MOSFET (Q2). When the RADIOCON signal is logic high the Receiver will be connected to the antenna via the MOSFET (Q3). The key assumption with this design is that the MOSFETS have a large enough bandwidth to pass the 900MHz RF signal from the antenna to the RF modules.

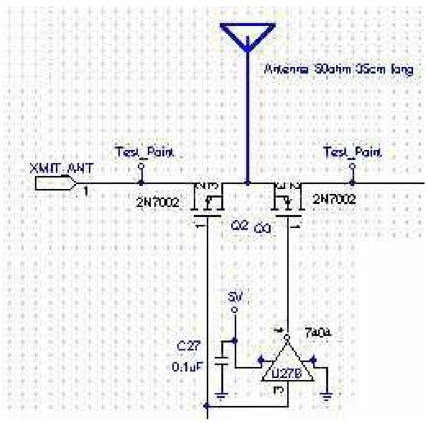


Figure #7 Antenna Multiplexor

Memory Map Circuitry

This block allows the Receiver to output its parallel data on the data bus, so the microprocessor can read in the data and do the appropriate function. The circuit used to implement this function contains an octal buffer (U8) and an octal D-Latch (U9). The octal buffer is used to enable the output of the Receiver IC Chip (U10) to be put on the data bus.

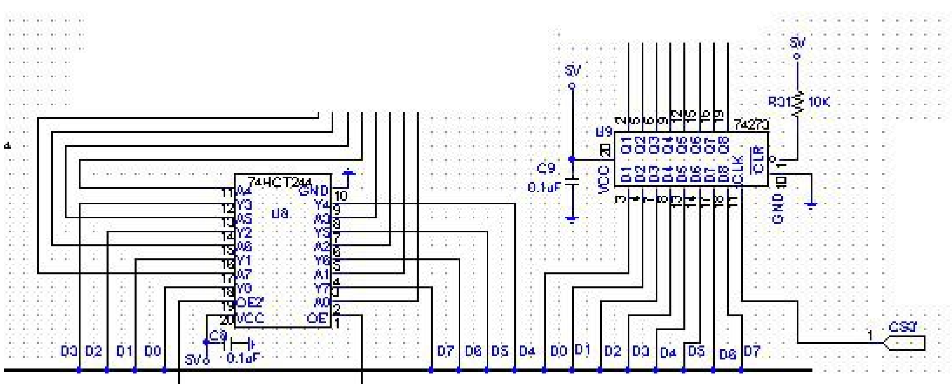


Figure #8 RF Receiver Memory Map Circuit

The CS1 line controls this function; when information is read from address $B590 the CS1 line enables octal buffer. The octal D-Latch is used to latch in data from the data bus into the Channel Select pins of the Receiver IC Chip (U10). The CS3 line controls this function; when information is stored to address $B5B0 the CS3 line enables octal D-Latch. This functionality allows the programmer to easily select up to 256 different RF Channels to use while communicating. The key assumptions with this design are that octal buffer has tri-state outputs and the D-Latch has high impedance inputs.

RF Receiver Module

This block takes the RF signal from the antenna and translates it into a parallel format. The circuit used to implement this block contains a RF receiver and Holtek's HT-648L (U10). The receivers that can be used are either Reynold's Electronics RWS-434 or LINX's RXM-900-HP. The RWS-434 is a 433MHz receiver that has a range of 400feet. The RXM-900-HP is a 900MHz receiver that has a range of 1000feet. The main difference in these modules is cost. The design allows for either of these receivers to be used. The HT-648L is an IC chip that takes the serial data out of the RF module and changes it to a parallel output. When data is valid on the output of the HT-648L the RFVT signal goes high; this signal is tied to an interrupt on the microprocessor.

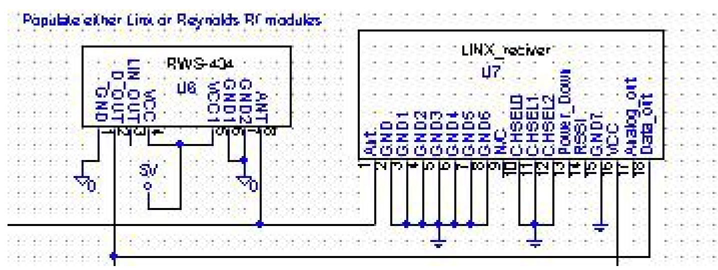


Figure #9 RF Receiver Modules

RF Transmitter

This block is on both the Data Logger and the Monitoring Tool. This block takes the parallel data from the data bus and translates it into a RF signal and then transmits that signal over the antenna.

Antenna Multiplexor

See RF Receiver.

Memory Map Circuitry

This block allows the Transmitter to input its parallel data from the data bus. The circuit used to implement this function contains two octal D-Latches (U4 & U5). The octal latch (U4) is used to enable the input of the Transmitter IC Chip (U3) to receive the data from data bus. The CS0 line controls this function; when information is stored to address $B580 the CS0 line enables D-Latch. The octal D-Latch (U5) is used to latch in data from the data bus into the Channel Select pins of the Transmitter IC Chip (U3). The CS4 line controls this function; when information is stored to address $B5C0 the CS4 line enables octal D-Latch. This functionality allows the programmer to easily select up to 256 different RF Channels to use while communicating. The key assumption with this design is that the D-Latches have high impedance inputs.

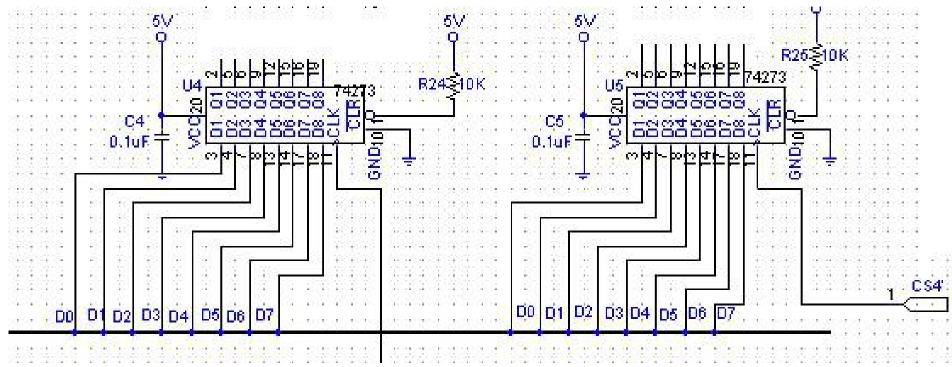


Figure #10 RF TX Memory Map Circuit

RF Transmitter Module

This block takes the parallel data and translates it into a RF signal. The circuit used to implement this block contains a RF transmitter and Holtek's HT-640 (U3). The transmitters that can be used are either Reynold's Electronics TWS-434 or LINX's TXM-900-HP. The TWS-434 is a 433MHz transmitter that has a range of 400feet. The TXM-900-HP is a 900MHz transmitter that has a range of 1000feet. The main difference in these modules is cost. The design allows for either of these transmitters to be used. The HT-640 is an IC chip that takes the parallel data from the data bus and converts it into a serial data stream, which is feed into the RF transmitter module.

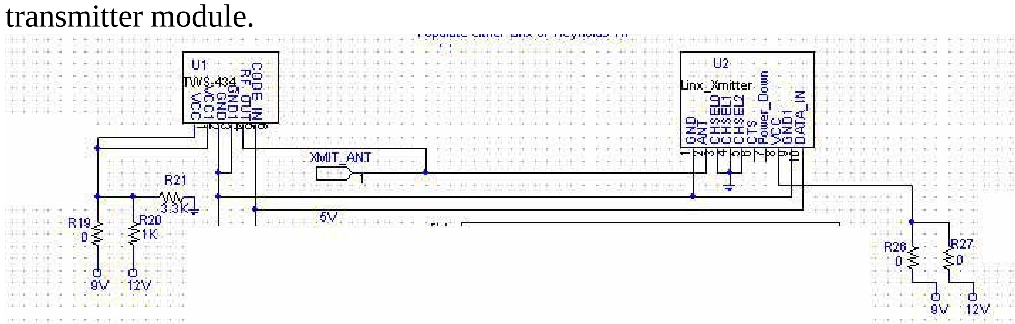
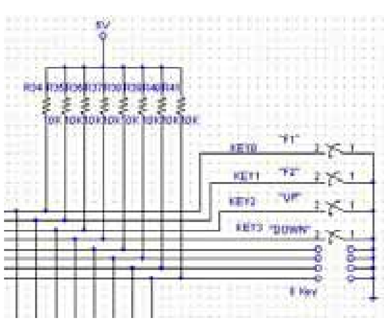


Figure #11 RF TX Modules

Keypad Circuitry

This block is only on the Monitoring Tool. This block receives the input from the keypad and translates that into a parallel data format that can be sent to the microprocessor via the data bus.



Keypad with Pull-up Resistors

Up to 8 normally open momentary push button keys can be used with this design. One side of the switches are connected to ground, the other side is connected to an octal buffer (U11) with 10KΩ pull-up resistors. The key assumption with this block is the 10KΩ resistors will allow enough current needed to operate the octal buffer inputs.

Figure #12 Keypad

Memory Map Circuitry

This block allows the keypad to output its parallel data on the data bus, so the microprocessor can read in the data and do the appropriate function. The circuit used to implement this function contains an octal buffer (U11). The octal buffer is used to enable the output of keypad to be put on the data bus. The CS2 line controls this function; when information is read from address $B5A0 the CS2 line enables octal buffer. The key assumptions with this design are that octal buffer has tri-state outputs.

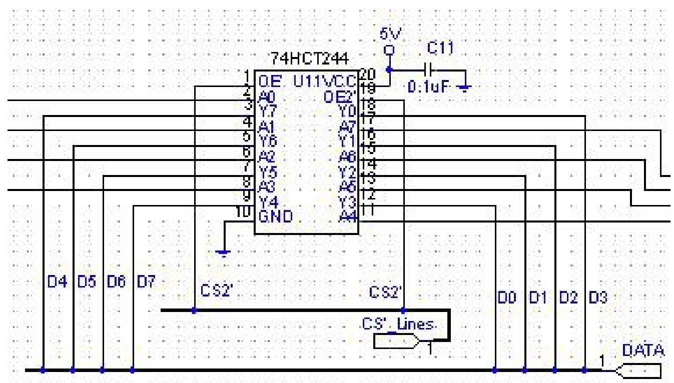


Figure #13 Keypad Memory Map Circuit

Key Pressed Identification Logic

This block gives a signal to the microprocessor when a key is pressed. Seven AND gates are used to create the KEYVT signal. This signal goes logic low when a key is pressed and is connected to an interrupt pin on the microprocessor.

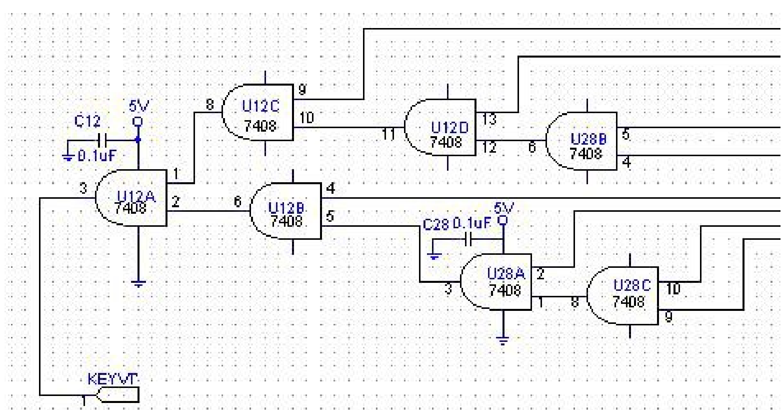


Figure #14 Key Pressed Logic

Automotive Circuitry

This block is only on the Data Logger. This block receives and transmits data from the microprocessor to the Fuel Injection Controller.

Diode Protection

This block is used to protect the Data Logger from voltage spikes on the K and L lines from –50 to +50V. This is implemented by two clamping diodes on each line. The diodes clamp the voltage on the lines to remain between 0 to 12V. The key assumption with this design is the resistors before the clamping diodes can withstand a voltage of 62V and the diodes can withstand a reverse voltage of 62V.

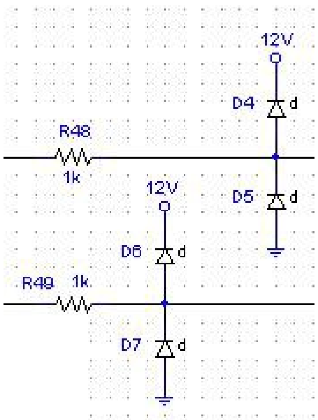
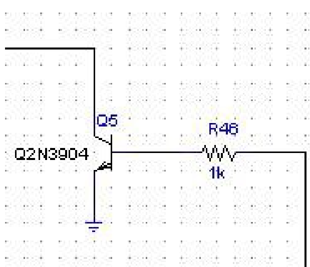


Figure #15 Diode Protection



Transmit Circuitry

This block is used to translate the data sent from the microprocessor in to a format that can be transmitted to the Fuel Injection Controller. This is implemented by the use of a BJT with a 10KΩ pull-up on it. When the TX line goes logic low the K line goes to 12V. When the TX line is logic high the K line goes to 0V.

Figure #16 Automotive TX

Receiver Circuitry

This block is used to translate the data sent from the Fuel Injection Controller to a format the microprocessor can read. This is implemented by using an analog comparator (U13). When the L line is 12V the RX line is 5V. When the L line is 0V the RX line is 0V. The RX line is connected to an interrupt pin on the microprocessor. This is do so when the Fuel Injection Controller sends data, an interrupt subroutine will be called and the data will be clocked into the microprocessor.

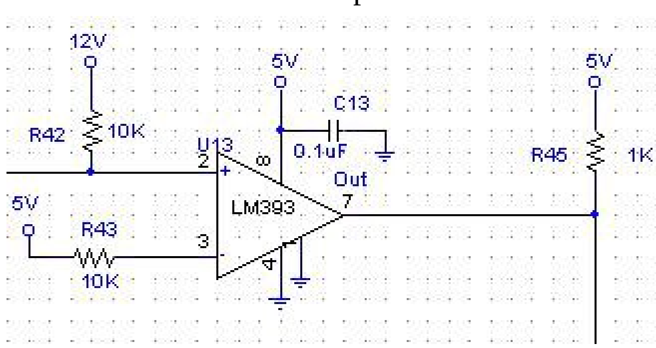


Figure #17 Automotive RX

Power Supply

This block is on both the Data Logger and the Monitoring Tool. This block receives power from either 9.6V Battery (on Monitoring Tool), 9V Wall Adapter (on Monitoring Tool), or 12V Automotive Battery (on Data Logger) The power is then regulated and connected to the appropriate power plane.

Diode & Fuse Protection Circuitry

This block protects the Data Logger and Monitoring Tool from a reverse voltage of up to 50V and current surges greater than 1.5 Amps. This is implemented by using a rectifier diode and a 1.5 amp fuse. The key assumption with this design is that the diode will have a reverse voltage of greater then 50V.

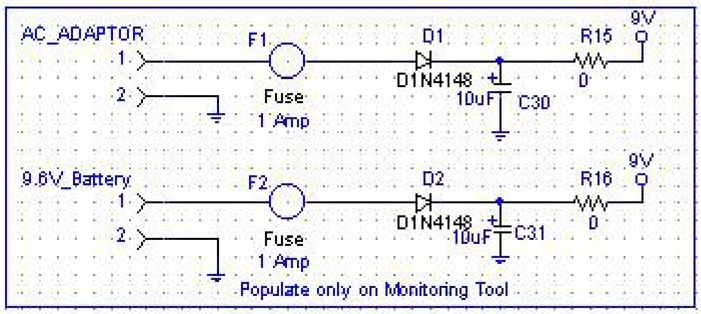


Figure #18 Diode & Fuse Protection

III. Discussion & Analysis

A. Limitations / Weaknesses

Limitations of this project include:

1. The Maximum Range of RF transmit and receive is 250 feet.
2. The amount of EEPROM for application code is only 8K so assembly code must be used instead of C.
3. The amount of SRAM is limited to 32K.
4. Only an 8 key keypad is supported in this design.

B. Further Work Required

The work still required to finish the project include:

1. Write the PC Software needed to communicate with either the Monitoring Tool or the Data Logger.
2. Mount and Test unit on the race car with the Fuel Injection Controller

IV. Simulation

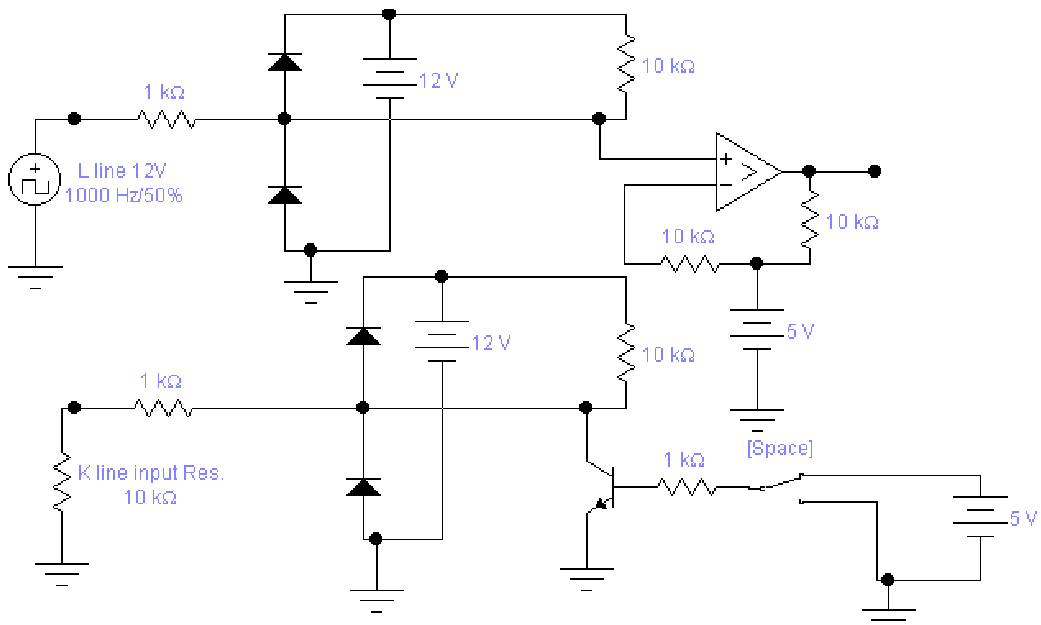
A. Approach

The Automotive Interface Circuitry was simulated. This was chosen because it is a critical path and has the most variation of output based on resistor tolerances. Considerations for simulation program include Pspice, Microsim version 8, and Electronics Workbench version 5.12. Electronics Workbench was chosen over Pspice due to my previous experiences with the program, and the use of models could be employed for various parts.

My best-case scenario would be if all the resistors were there rated value. My Nominal + - 10% simulations correspond to the values of the resistors. My worst case scenario has some resistors at +10% of their rated value and other resistors at –10% of their rated value.

B. Models

The following is the circuit simulated in Electronics WorkBench 5.12.

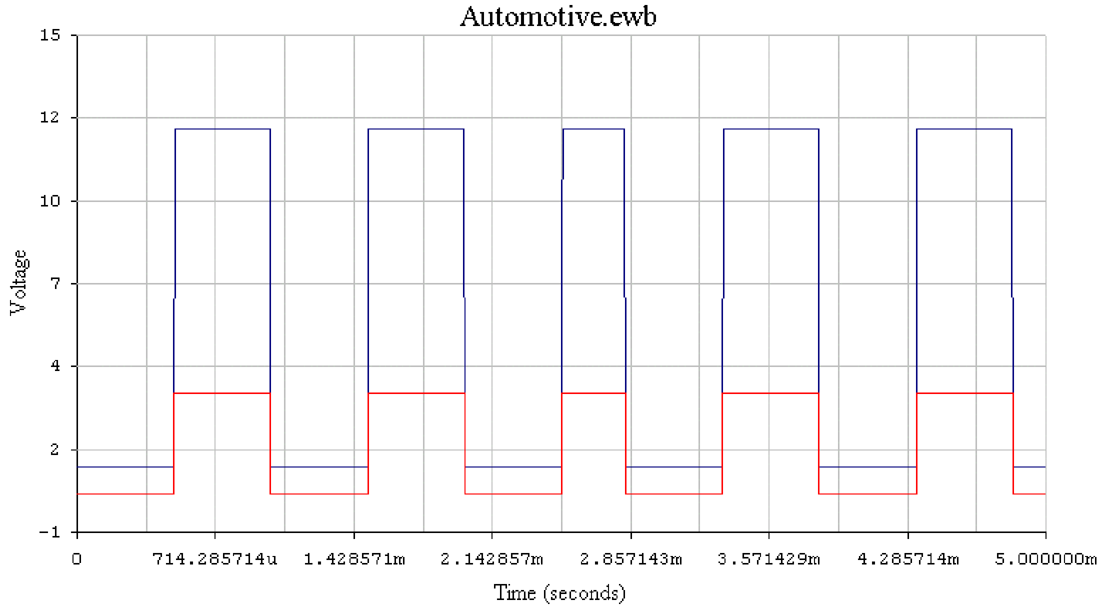


The square wave generator simulates data being sent to the Data Logger from the Fuel injection Controller. A 10KΩ resistor simulates the input impedance of the K-line on the Fuel Injection Controller. Batteries are used to simulate the regulated voltage provided on the Data Logger. The switch represents the output of the microprocessor.

C. Simulation Summary

Best-Case Simulation

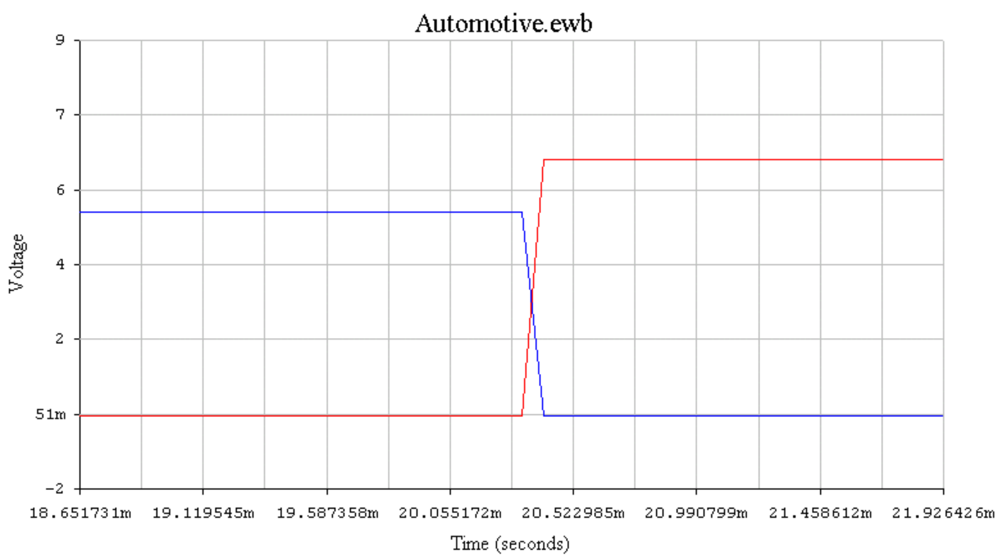
The following is the waveform of the receiver part of the automotive circuitry.



Trace #1 (Blue) Input waveform from Fuel Injection Controller.
Trace #2 (Red) Output waveform to go to microprocessor

The input to the comparator ranges from 11.75V for a logic high to 0.75V for a logic low. The microprocessor input is ranging from 3.75V for a logic high to 0V for a logic low. This is within the specs needed for the 68HC11, which needs the input to be greater than 3.5V for a logic high and below 0.3V for a logic low.

The following is the transmit waveform of the transmit part of the automotive circuitry.

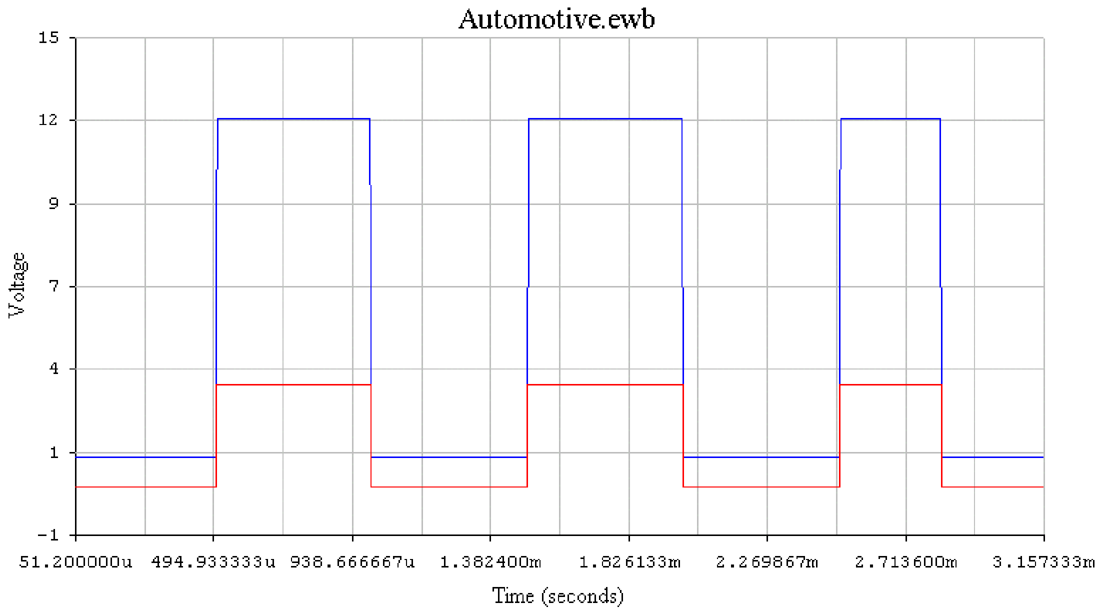


Trace #1 (Blue) Input waveform from microprocessor.
Trace #2 (Red) Output waveform to go to Fuel injection Controller on K line

The input to the transistor ranges from 5V for a logic high to 0.05V for a logic low. The transistor output is ranging from 6.5V for a logic high to 0.05V for a logic low. This is within the specs needed for the Fuel Injection Controller, which needs the input to be greater than 6.0V for a logic high and below 4.5V for a logic low.

Nominal +10% Simulations

The following is the waveform of the receiver part of the automotive circuitry.

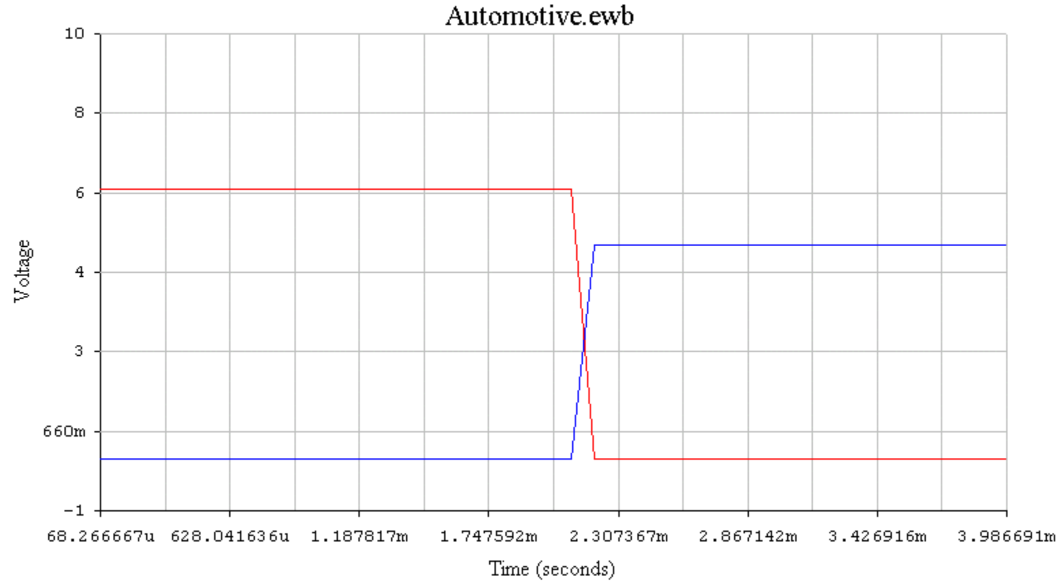


Trace #1 (Blue) Input waveform from Fuel Injection Controller.
Trace #2 (Red) Output waveform to go to microprocessor

The input to the comparator ranges from 12.0V for a logic high to 0.75V for a logic low. The microprocessor input is ranging from 3.75V for a logic high to 0V for a logic low. This is within the specs needed for the 68HC11, which needs the input to be greater than 3.5V for a logic high and below 0.3V for a logic low.

DOI: 10.56816/2378-6949.1179

The following is the transmit waveform of the transmit part of the automotive circuitry.

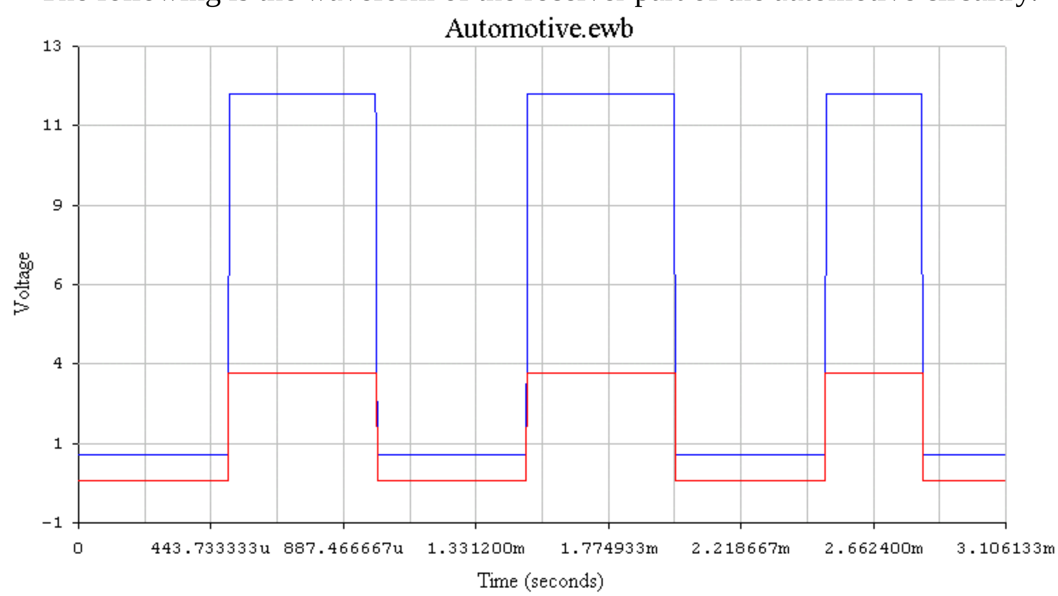


Trace #1 (Blue) Input waveform from microprocessor.
Trace #2 (Red) Output waveform to go to Fuel injection Controller on K line

The input to the transistor ranges from 5V for a logic high to 0.005V for a logic low. The transistor output is ranging from 6.2V for a logic high to 0.005V for a logic low. This is within the specs needed for the Fuel Injection Controller, which needs the input to be greater than 6.0V for a logic high and below 4.5V for a logic low.

Nominal –10% Simulations

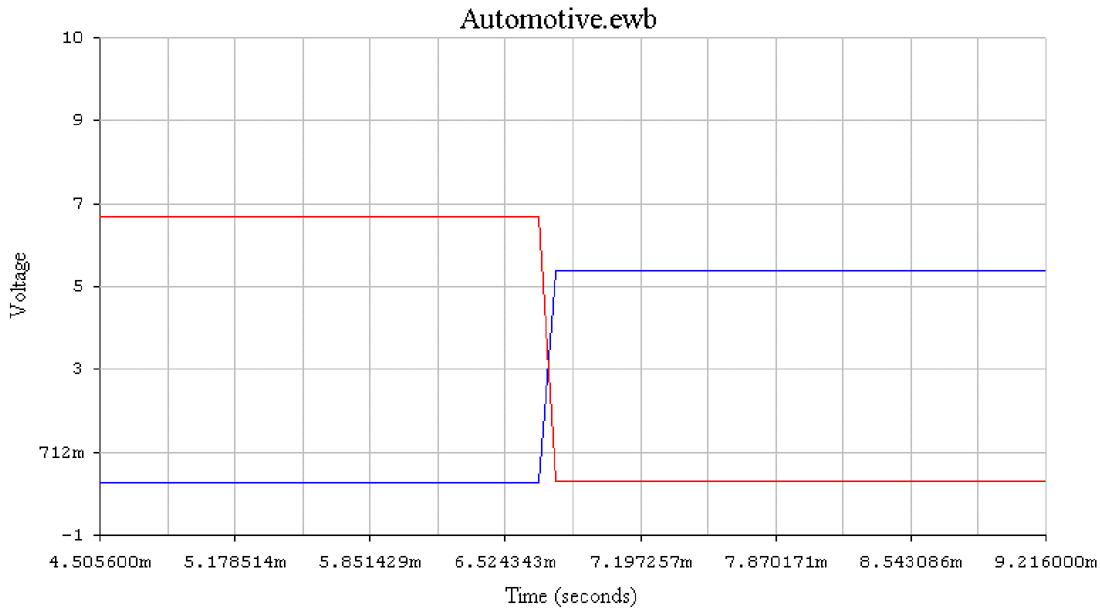
The following is the waveform of the receiver part of the automotive circuitry.



Trace #1 (Blue) Input waveform from Fuel Injection Controller.
Trace #2 (Red) Output waveform to go to microprocessor

The input to the comparator ranges from 12.0V for a logic high to 0.75V for a logic low. The microprocessor input is ranging from 3.75V for a logic high to 0V for a logic low. This is within the specs needed for the 68HC11, which needs the input to be greater than 3.5V for a logic high and below 0.3V for a logic low.

The following is the transmit waveform of the transmit part of the automotive circuitry.



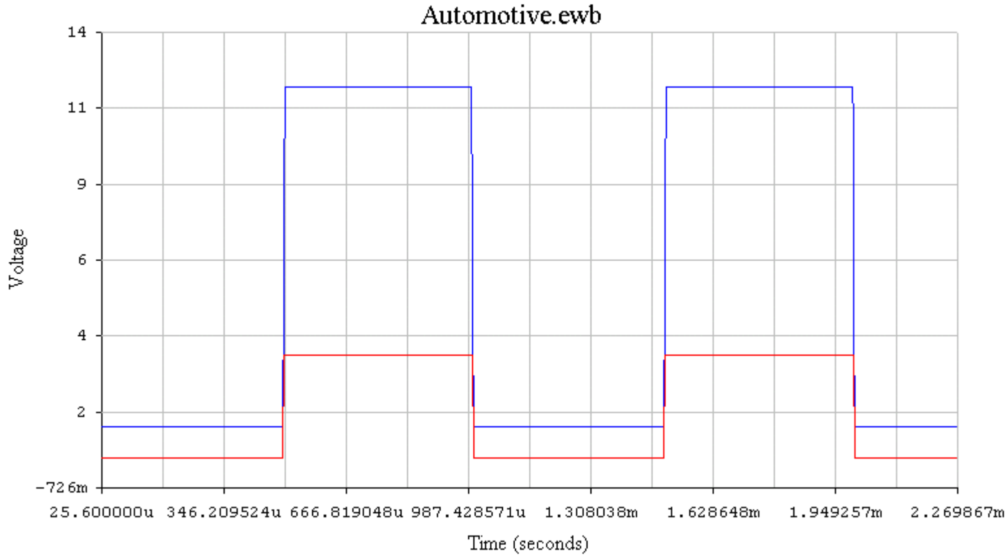
Trace #1 (Blue) Input waveform from microprocessor.

Trace #2 (Red) Output waveform to go to Fuel injection Controller on K line

The input to the transistor ranges from 5.3V for a logic high to 0.5V for a logic low. The transistor output is ranging from 6.85V for a logic high to 0.5V for a logic low. This is within the specs needed for the Fuel Injection Controller, which needs the input to be greater than 6.0V for a logic high and below 4.5V for a logic low.

Worst-Case Simulations

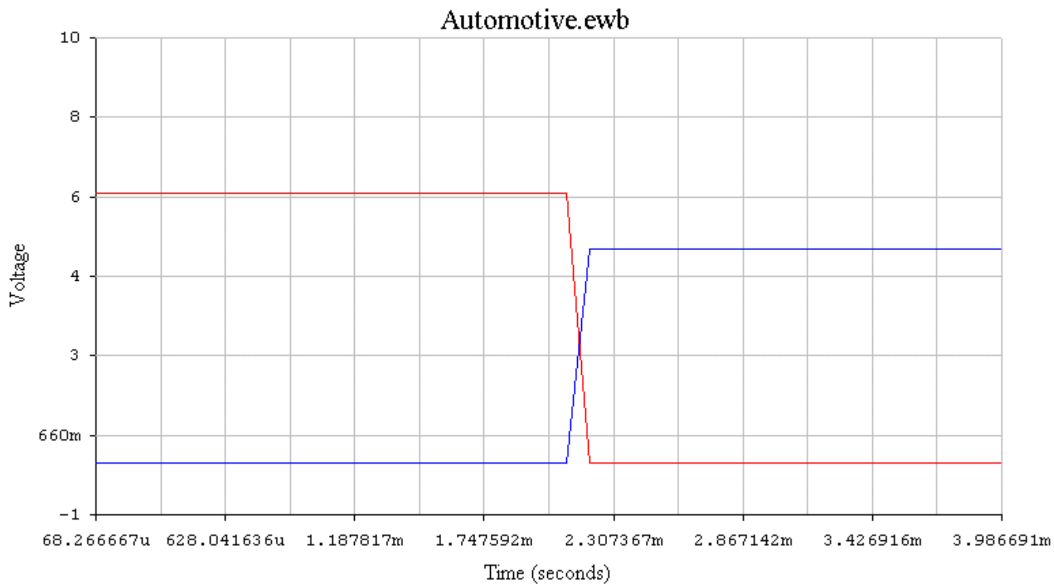
The following is the waveform of the receiver part of the automotive circuitry.



Trace #1 (Blue) Input waveform from Fuel Injection Controller.
Trace #2 (Red) Output waveform to go to microprocessor

The input to the comparator ranges from 12.0V for a logic high to 1.75V for a logic low. The microprocessor input is ranging from 3.75V for a logic high to 0V for a logic low. This is within the specs needed for the 68HC11, which needs the input to be greater than 3.5V for a logic high and below 0.3V for a logic low.

The following is the transmit waveform of the transmit part of the automotive circuitry.



Trace #1 (Blue) Input waveform from microprocessor.
Trace #2 (Red) Output waveform to go to Fuel injection Controller on K line

The input to the transistor ranges from 5.0V for a logic high to 0.05V for a logic low. The transistor output is ranging from 6.1V for a logic high to 0.05V for a logic low. This is within the specs needed for the Fuel Injection Controller, which needs the input to be greater than 6.0V for a logic high and below 4.5V for a logic low.

DOI: 10.56816/2378-6949.1179

V. Test Report

A. Summary of Design Changes

The following changes were made to the design in order to test the circuitry.

Peizo Buzzer (Speaker)

Pin 1 of the buzzer was connected to 9V and pin 2 was connected to R17. When Q4 is turned on, 9V is approximately put across the buzzer causing it to make noise. The volume of the buzzer was too loud, so the trace connecting pin 1 of the buzzer to 9V was cut and a fly wire was added connecting it to 5V.

Removal of U27A, U27D, U27E

It was found by testing that the inverters placed on the Chip Select lines of the Octal D-Latches were not necessary. So pins 1,2,8,9,10,and 11 were lifted on U27, disconnecting the inverters. And fly wires were added to connect pads 1&2, 8&9, 10&11 correspondingly.

Removal of Q4 and R22

It was found by testing that Q4 would not be needed to turn on and off the RF Transmitting Module. This is because the RF module uses On-Off Carrier to transmit. So as long as the Data-In pin is at logic low the transmitter is off. So Q4 and R22 were removed and a jumper was placed connecting the collector and emitter pad of Q4 together.

Pin out of U13

The pin out of U13 was originally laid out on the Printed Circuit Board to be:

Pin 1	*Output*
Pin 2	*Positive In*
Pin 3	*Negative In*
Pin 4	*GND*
Pin 5	*NC*
Pin 6	*NC*
Pin 7	*NC*
Pin 8	*VCC*

Further research and testing revealed that the actual pin out is the following:

Pin 1	*GND*
Pin 2	*Positive In*
Pin 3	*Negative In*
Pin 4	*GND*
Pin 5	*NC*
Pin 6	*NC*
Pin 7	*Output*
Pin 8	*VCC*

So the trace connecting pin1 of U13 to the out put was cut and a jumper was placed to connect pin 1 to GND. Then a fly wire was placed connecting pin 7 of U13 to the pad of R45.

Addition of R46 and D8

In order to prevent damage to the circuitry of the data logger when connected to a battery greater than 12V, a 10Ω 10Watt resistor was placed in series with the power input and a 12V 500mA zener diode was place after R46. This will clamp the voltage at 12Volts even if the battery voltage exceeds 12V.

Changing R45's Value

The value of R45 was changed to 1KΩ instead of its original 10KΩ. This was done because on the EVBU board a 10KΩ pull down is used on the same line. So a 10KΩ pull up will cut the voltage on that line in half. By changing R45 to 1KΩ the voltage stays within the logic levels of the microprocessor.

Changing C32's Value

The value of C32 was changed from 10uF to 220uF to better filter off any voltage fluctuations of the battery.

Trace Cut

Upon inspection of the Printed Circuit Board (PCB), it was noticed that line D3 was connected to the pad of pin 10 on U8. The pad of pin 10 on U8 was cut shorter to prevent the connection to line D3.

B. Form

1.1 The wireless data logger consists of two pieces; one is the data logger, which is placed in the car next to the fuel injection controller. The other is a handheld monitoring tool.

This can be seen in Picture #1 in the appendix.

1.2 Data Logger Case

1.2.1 The data logger will be housed in a RF shielded metal case.

The container will be visually inspected for compliance.

The data logger is housed in a cast aluminum case as seen in Picture #2 in the appendix, which meets the specification.

1.2.2 The case will be a box no larger than 1' x 1' x 4''.

The container will be measured using a ruler and the dimensions recorded to within +/-0.125".

The dimensions of the case are 7.5'' x 7.5'' x 2.5'', which meets the specification.

1.3 Monitoring Tool Case

1.3.1 The monitoring tool will be housed in a plastic container.

The container will be visually inspected for compliance.

The monitoring tool case is made of both Black ABS plastic and Aluminum sheet metal as seen in Picture #3 in the appendix, which meets the specification.

1.3.2 The tool will be handheld and no larger than 1' x 6'' x 3''.

The container will be measured using a ruler and the dimensions recorded to within +/-0.125".

The dimensions of the case are 8.5'' x 4.75'' x 2.75'', which meets the specification.

2.1 The monitoring tool:

2.2 The front face of the monitoring tool has the following:

2.2.1 A 20x4 character LCD screen will be in the top half of the 1' x 6'' side (See picture for location).

The front face will be visually inspected for compliance

The monitoring tool has a LCD screen in the top half of the 8.5" x 4.75" side as seen in Picture #3 in the appendix.

2.2.2 A 4-button keypad will be below the LCD screen (See picture for location)

The keypad will be visually inspected for compliance

The monitoring tool has a 4 button keypad under the LCD screen on the 8.5" x 4.75" side as seen in Picture #3 in the appendix.

2.2.2.1 The functions of the buttons will be

2.2.2.1.1 Up

2.2.2.1.2 Down

2.2.2.1.3 F1

2.2.2.1.4 F2

2.3 The top 6’’ x 3’’ side of the monitoring tool has the following:

The topside will be visually inspected for compliance of these features.

The top 4.75” x 2.75” side of the monitoring tool has the following: push button switch, 6.5” antenna, and an external AC adapter plug.

2.3.1 A push button power switch (See picture for location).

2.3.2 A black 6.5’’ long RF antenna (See picture for location).

2.3.3 An external AC adapter connector (See picture for location).

2.4 The bottom 6’’ x 3’’ side of the monitoring tool has the following:

2.4.1 A female DB-9 RS-232 serial connector (See picture for location).

The monitoring tool has the DB-9 female connector placed on the left 8.5” x 4.75” side. This is different from the specification and was changed to meet easier design implementation.

3.1 The data logger has the following features:

The data logger case will be visually inspected for compliance of these features.

The data logger case has the following a 6.5” long antenna and a water proof female GM connector. The DB-9 connector was integrated into the GM connector to decrease the amount of connectors needed, also this makes the data logger case rain proof.

3.2 A SAE J1962 communications jack (See picture for location).

3.3 A female DB-9 RS-232 serial connector (See picture for location).

3.4 A black 6.5’’ long RF antenna (See picture for location).

DOI: 10.56816/2378-6949.1179

C. Fit

1.1 The data logger operates on the Formula-SAE race car.

1.2 The unit will be able to handle the vibrations of the car.

The data logger will be mounted on the Formula-SAE car. The car will then be driven over terrain similar to the terrain of the racetrack the car will be raced on.

Due to the Formula-SAE not being complete by the scheduled testing time of the data logger. The data logger was shaken up by hand for 5 min. then turned on and passed inspection.

1.3 The unit will operate over the temperature range 0 to 55˚C in a dry environment.

The data logger will be placed in an environmental chamber at DATARADIO. The chamber's temperature will be varied from 0 - 55°C. The data logger will be tested by connecting a ECU emulator and using the monitoring tool to see if the correct data is being sent via the RF link.

	Testing Temperature	***Passed Y/N***
Run #1	***0 degrees C***	***Y***
Run #2	***55 degrees C***	***Y***

Table #1 Temperature Testing of the data logger

2.1 The monitoring tool can be power by an internal battery pack or an external AC Adapter power supply.

2.2 The tool will operate for at least 2 hours on 9.6Volt 1.1Ah battery pack

The monitoring tool will have a completely charged battery place in it. The tool will be left on until the unit stops operating. Time for the battery to discharge will be recorded.

Operation of Monitoring Tool	***Total Current Draw***
Stationary when Running code	***57.8mA***
When a button is pushed	***65.0mA***
When speaker is on	***89.3mA***
When Transmitting Wirelessly	***63.0mA***

Table #2 Total Current draw from the Monitoring Tool

The monitoring tool operates a minimum of 10 hours on a 9.6V 1.1Ah battery pack.

2.3 The AC adapter is connected to the plug on the top of the Monitoring tool (See picture for location).

An AC adapter will be plugged in the monitoring tool. The monitoring tool will then be turned on and all operations will be verified.

The monitoring tool operates fully with the use of the AC adapter.

3.1 The circuitry on both the data logger and monitoring tool will:

3.2 Follow FCC regulations on the wireless transmissions.

A spectrum Analyzer will be used to measure the RF output power of the Data Logger and the Monitoring Tool.

The output of the 433MHz transmitter is 8mW to a 50Ω antenna.

3.2 Follow RS-232 communication protocol when interfacing with a PC.

The TX232 and RX232 lines will be used to send software into the monitoring tool. The software will be then executed to see if the RS-232 lines operated correctly.

The TX232 and RX232 lines were used to load software into the monitoring tool and the software executed correctly.

4.1 The data logger will be powered by the car's 12Volt CBR600 motorcycle battery.

4.2 The data logger receives power from the connection with the Fuel injection controller.

The Fuel Injection Controller will be plugged into the data logger. The data logger will then be turned on and all operations will be verified.

Due to the Fuel Injection Controller not being completed by the scheduled testing date the data logger was just connected to the CBR600 battery and operated correctly.

4.3 The data logger will draw no more than 2 Amps from the battery.

A multimeter will be put in line with the incoming power from the Fuel Injection Controller and the input current to the data logger will be measured.

The data logger drew 52.3mA when it was not transmitting and 57.7mA when transmitting.

DOI: 10.56816/2378-6949.1179

D. Function

1.1 The data logger will communicate with the fuel injection controller, monitoring tool, and a PC.

1.2 The data logger is activated and deactivated by the Fuel injection controller.

1.2.1 When the key is in the ignition the Fuel injection controller turns on and then the data logger is activated.

The data logger will be connected the FIC and the key will be inserted into the ignition. A multimeter will be used to verify if power is applied to the Data logger.

Due to the Fuel Injection Controller not being completed by the scheduled testing date the data logger was just connected to the CBR600 battery and operated correctly.

1.2.2 When the key is removed from the ignition the Fuel injection controller turns off and then the data logger is deactivated.

The data logger will be connected the FIC and the key will be removed from the ignition. A multimeter will be used to verify if power is no longer applied to the Data logger.

Due to the Fuel Injection Controller not being completed by the scheduled testing date the data logger was just disconnected from the CBR600 battery and the unit stopped operating.

1.3 The data logger will request sensor data from the fuel injection controller via a serial link 4 times a second following a protocol specified in the Design Documentation.

1.3.1 Sensor Data includes: Oxygen content of fuel, Throttle position, Coolant Temp, Voltage of Battery, Manifold pressure, and engine speed.

A FIC emulator will be connected to the data logger. A scope with be connected to the TX and RX lines and the rated of transmission will be verified.

A 68HC11 EVBU board was programmed to operate like the FIC and transmitted data to the data logger. The data logger was then connected to a computer and the memory was checked to verify the correct data was transmitted.

1.4 The logger will receive the sensor data from the controller following a protocol specified in the Design Documentation.

A scope will be connected to the TX and RX lines and the transmission protocol will be verified.

A 68HC11 EVBU board was programmed to operate like the FIC and transmitted data to the data logger. The data logger was then connected to a computer and the memory was checked to verify the correct data was transmitted.

1.5 The logger will then store the data in volatile memory.

1.5.1 Up to 32K bytes of data can be stored.

The size of the SRAM used will be verified by the manufacturer's data sheet.

The SRAM used is the Samsung KGT0808C10-0L70 which is a 32KSRAM.

1.5.2 When the memory is full the newest data will be stored over the oldest data. In a stack configuration.

The data logger will be connected to the FIC emulator and data will be transmitted and stored into the 32K of SRAM on the Data Logger. When over 32K of data has been sent to the data logger, a PC will be connected and the information in the SRAM will be downloaded to the PC. The data will then be analyzed to verify the oldest data has been overwritten.

A 68HC11 EVBU board was programmed to operate like the FIC and transmitted data to the data logger. The data logger was then connected to a computer and the memory was checked to verify the correct data was transmitted.

1.6 The logger will sense the RF signal strength from the monitoring tool.

1.6.1 The signal strength is measured to determine if the Monitoring tool is within range (200 ft.) of the Data logger.

The range of the RF communications will be tested by moving the monitoring tool away from the data logger and observing the data received by the monitoring tool on the LCD screen. The distance between the two will be measured with a tape measure.

The monitoring tool received the correct information from the data logger when a distance of up to 250 feet separated them.

1.7 When the monitoring tool is within range (200 ft) the logger transmits the stored sensor data to the monitoring tool via a wireless link (433MHz AM modulated).

1.7.1 When the signal strength falls below acceptable levels, the logger stops transmitting to the monitoring tool.

The range of the RF communications will be tested by moving the monitoring tool away from the data logger and observing the data received by the monitoring tool on the LCD screen. The distance between the two will be measured with a tape measure.

The monitoring tool received the correct information from the data logger when a distance of up to 250 feet separated them.

1.7.1.1 The acceptable level will be determined by future research.

1.8 When the logger is connected to a PC via an RS-232 serial link.

1.8.1 The PC will request the sensor data via the serial link using RS-232 protocol.

1.8.1.1 This is done by the user selecting an option in the PC software's menu.

This will be tested by connecting a scope to the serial port on the PC and observing the data stream.

The PC software is not scheduled to be completed until Eric Holland's Graduate year Sept. 2002- May 2003 and thus can not be tested.

1.8.2 Upon receiving the request the Data logger will transmit all of the stored data to the PC.

This will be tested by connecting a scope to the serial port on the PC and observing the data stream.

The PC software is not scheduled to be completed until Eric Holland's Graduate year Sept. 2002- May 2003 and thus can not be tested.

1.8.3 The PC will then interpret the data and display in on the screen in graphical or table format. The user then has the option to print or save the data to the PC.

This will be tested by either saving the data to a disk or printing the data from the PC software.

The PC software is not scheduled to be completed until Eric Holland's Graduate year Sept. 2002- May 2003 and thus can not be tested.

2.1 The monitoring tool will communicate with the data logger and a PC.

2.2 The monitoring tool can be turned on / off via a power switch on the top of the tool (See picture for location).

The AC adapter will be inserted into the monitoring tool and the power switch will be pressed. A multimeter will be used to verify if power is applied to the Data logger.

The monitoring tool operates fully with the use of the AC adapter.

2.3 The monitoring tool will transmit an AM modulated RF signal at 433MHz to the data logger.

2.3.1 When the data logger receives the signal it measures the signal strength.

2.3.1.1 The signal strength is measured to determine if the Monitoring tool is within range (200 ft.) of the Data logger.

The range of the RF communications will be tested by moving the monitoring tool away from the data logger and observing the data received by the monitoring tool on the LCD screen. The distance between the two will be measured with a tape measure.

The monitoring tool received the correct information from the data logger when a distance of up to 250 feet separated them.

2.4 When the monitoring tool is within range (200 ft), the data logger will send the stored sensor data to the monitoring tool via a serial wireless link.

The range of the RF communications will be tested by moving the monitoring tool away from the data logger and observing the data received by the monitoring tool on the LCD screen. The distance between the two will be measured with a tape measure.

The monitoring tool received the correct information from the data logger when a distance of up to 250 feet separated them.

2.5 The monitoring tool will store the sensor data in volatile memory.

2.5.1 Up to 32K bytes of data can be stored.

The size of the SRAM used will be verified by the manufacturer's data sheet.

The SRAM used is the Samsung KGT0808C10-0L70 which is a 32KSRAM.

2.5.2 When the memory is full the newest data will be stored over the oldest data. In a stack configuration.

The monitoring tool will be connected to the data logger via the RF link and data will be transmitted and stored into the 32K of SRAM on the Monitoring tool. When over 32K of data has been sent to the data logger, a PC will be connected and the information in the SRAM will be downloaded to the PC. The data will then be analyzed to verify the oldest data has been overwritten.
A 68HC11 EVBU board was programmed to operate like the FIC and transmitted data to the data logger. The data logger was then connected to a computer and the memory was checked to verify the correct data was transmitted.

2.6 The monitoring tool will display the selected sensor data on the LCD screen.

2.6.1 The user chooses with sensor data to display by pressing the UP / Down buttons to select different menu options.

This will be tested by loading the application code into the monitoring tool and pressing the buttons to see if the data is displayed.
Software was loaded into the monitoring tool and the LCD screen and keypad was checked for proper operation.

Picture #4 LCD screen operation

2.6.2 When the monitoring tool is connected to a PC via an RS-232 serial link.

2.6.2.1 The PC will request the sensor data via the serial link using RS-232 protocol.

This will be tested by connecting a scope to the serial port on the PC and observing the data stream.
The TX232 and RX232 lines were used to load software into the monitoring tool and the software executed correctly.

2.5.2.1.1 This is done by the user selecting an option in the PC software's menu.

2.6.2.2 Upon receiving the request the Monitoring Tool will transmit all of the stored data to the PC.

This will be tested by connecting a scope to the serial port on the PC and observing the data stream.
The PC software is not scheduled to be completed until Eric Holland's Graduate year Sept. 2002- May 2003 and thus can not be tested.

2.6.2.3 The PC will then interpret the data and display in on the screen in graphical or table format. The user then has the option to print or save the data to the PC

This will be tested by either saving the data to a disk or *printing the data from the PC software.*

The PC software is not scheduled to be completed until Eric Holland's Graduate year Sept. 2002- May 2003 and thus can not be tested.

3.1 The PC software will have the following features for the user to select via a pull down (Windows style) menu.

3.2 Retrieve sensor data from the monitoring tool.

3.2.1 The user connects the monitoring tool to the PC serial port. Then the user starts program and chooses "Receive Data" from the pull down menu. The PC will then transmit a code to the monitoring tool via the serial port. Then the monitoring tool will send the stored data to the PC. Then PC will then interpret the data and display it on the screen.

This will be tested by either saving the data to a disk or *printing the data from the PC software.*

The PC software is not scheduled to be completed until Eric Holland's Graduate year Sept. 2002- May 2003 and thus can not be tested.

3.3 Save sensor data to a text file.

3.3.1 The user chooses "Save" from the pull down menu. The program then prompts the user to enter a file name and directory. The program then saves the data in a text file.

This will be tested by saving the data to a disk from *the PC software.*

The PC software is not scheduled to be completed until Eric Holland's Graduate year Sept. 2002- May 2003 and thus can not be tested.

3.4 Restore and display past saved sensor data files.

3.4.1 The user chooses "Restore" from the pull down menu. The program then prompts the user to enter a file name and directory. The program then opens the data and displays it on the screen.

This will be tested loading the data from a disk to *the PC software.*

The PC software is not scheduled to be completed until Eric Holland's Graduate year Sept. 2002- May 2003 and thus can not be tested.

3.5 Print sensor data.

3.5.1 The user chooses "Print" from the pull down menu. The program then prints the data to a printer connected to the parallel port.

This will be tested by printing the data from the PC software to a printer connected to the parallel port.

The PC software is not scheduled to be completed until Eric Holland's Graduate year Sept. 2002- May 2003 and thus can not be tested.

3.6 Update the fuel injection controller's engine performance tables.

This will be tested by visually inspecting the user interface of the program, and observing the car's performance with the monitoring tool.
The PC software is not scheduled to be completed until Eric Holland's Graduate year Sept. 2002- May 2003 and thus can not be tested.

3.5.1.1 The user selects an option in the PC software's menu to "Change ECU tables"

3.5.1.2 Then the user enters the new performance values into a table.

3.5.1.3 Then user then selects an option to "Transmit Table to ECU"

3.5.1.4 The PC then transmits the new table information to the monitoring tool via the RS-232 serial link.

3.5.1.5 The monitoring tool then transmits the new table information to the data logger via the RF wireless link.

3.5.1.6 The Data logger then transmits the new engine tables to the Fuel injection controller via the ISO 9141-2 serial link.

DOI: 10.56816/2378-6949.1179

E. Circuitry

Overview: The data logger and monitoring tool will each have an 8-bit microprocessor controlling the different pieces of the hardware. On the monitoring tool the microprocessor will control the RS-232 circuitry, RF modules, Keypad circuitry, and LCD memory mapping circuitry. On the Data Logger the microprocessor will control the RS-232 circuitry, RF modules, and automotive buffering circuitry.

Microprocessor and supporting circuitry: The microprocessor will be loaded with the application code. The system will be reset several times. During this time all I/O channels will be monitored and compared to expected waveforms.

The address and data lines were monitored with a logic analyzer while software was being run on the microprocessor. The I/O lines were verified to be working correctly.

RS-232 Circuitry: This circuitry will be tested by connecting it to a PC and using AXIDE2 send data back and forth.

The TX232 and RX232 lines were used to load software into the monitoring tool and the software executed correctly.

RF Modules: A serial data stream will be sent to the Transmit module by a 68HC11 EVBU board. The receiver module will be connected to a scope. The waveform will be compared to the expected waveform.

A scope was connected to the RF data in line and the waveform was verified.

Keypad Circuitry: The circuitry will be powered up and a multimeter will be used to check logic levels when the different buttons are pressed.

The keypad voltages were checked and verified.

	t Pressed Voltage	Pressed Voltage
	5.2V	0.0V
	5.2V	0.0V
	5.2V	0.0V
	5.2V	0.0V

Table #3 Keypad Voltage Chart

LCD Circuitry: The circuitry will be connected to the 68HC11 EVBU and code will be written to display "Automotive Interface Tooling" in the screen. If this works the LCD circuitry is correct.

Software was loaded into the monitoring tool and the LCD screen was functioning correctly. See Picture #4.

Automotive Buffers: A power supply will be connected to the input lines of these buffers. The voltage of the power supply will be varied from 0-20V to ensure the circuitry is properly protected.

The voltage on the input lines was increased to 20volts to insure the protection clamping diodes worked correctly. The Automotive buffer met specifications.

F. Equipment Required

The Equipment required is:

1. Multimeter
2. Oscilloscope
3. Logic Analyzer
4. 68HC11 EVBU
5. Fuel Injection Controller
6. FIC Emulator
7. Formula-SAE Race car
8. DATARADIO's Environmental Chamber
9. PC
10. Ruler
11. Tape Measure
12. Battery Charger

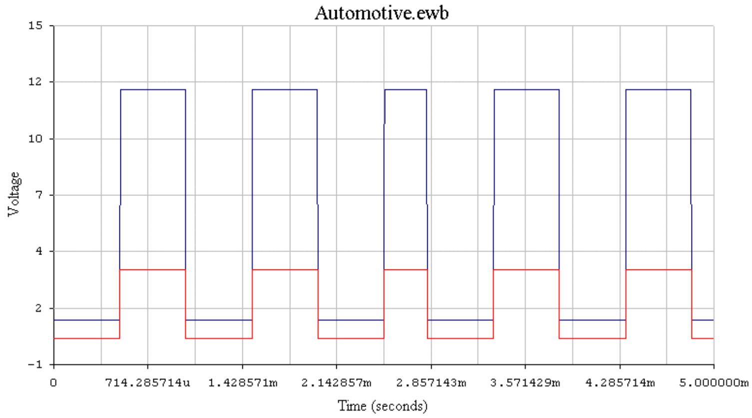
All items except DATARADIO's Environmental Chamber can be found in the senior design lab.

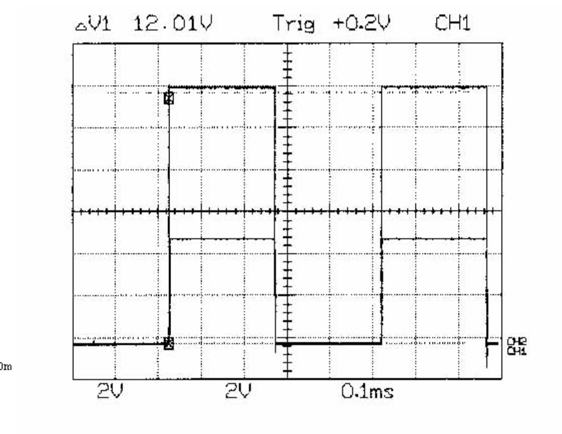
G. Expected Fault Coverage

Form: The identified tests cover all the items under this are of the specification. I therefore expect 100% coverage of form.

Fit: The identities test cover, to some degree, all items under this heading. I will not be able to directly test the specified vibration test, however it is expected that the planned test run on the car will provide a reasonable approximation. I would therefore expect a degree of cover age of approximately 75%.

Function: The identified tests will address all major functions, however cannot check them under all conditions. Some areas of deficiency include – checking range of the RF modules, PC software functionality, Fuel injection table update, and RS-232 waveform comparison. . I would therefore expect a degree of cover age of approximately 70%.





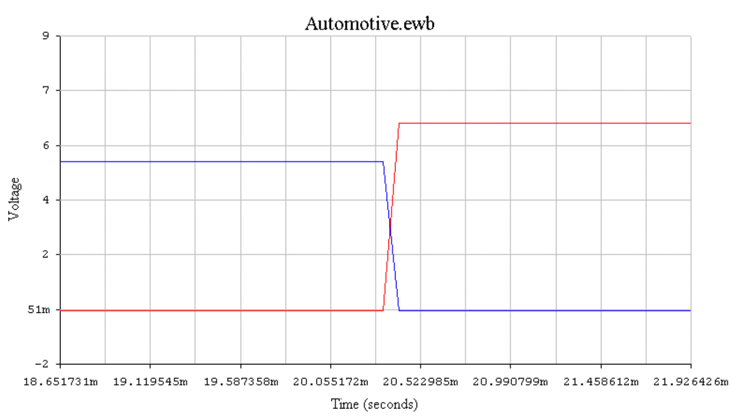
Simulation
Trace #1 (Blue) Input waveform from Fuel Injection Controller.

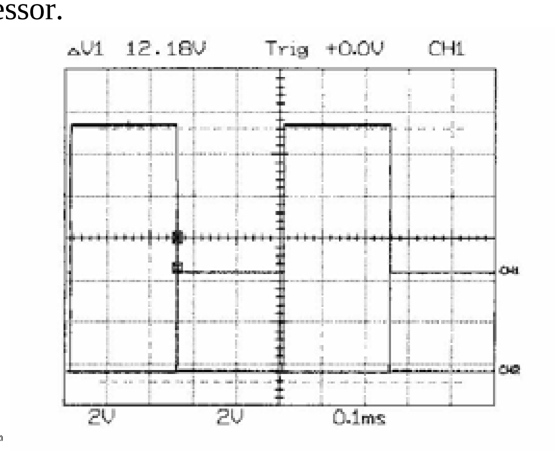
Trace #2 (Red) Output waveform to go to microprocessor

Actual Measurements

CH1 is the input from function generator (12Vpp). CH2 is the output of the comparator (5Vpp).

As can be seen from the actual measurement graph the Automotive Interface Receive Circuitry operates as simulated. When a 12Vpp pulse enters the comparator circuit it creates a 5Vpp pulse that goes to the microprocessor.





Simulation
Trace #1 (Blue) Input waveform from microprocessor.
Trace #2 (Red) Output waveform to go to Fuel injection Controller on K line

Actual Measurements
CH1 is the input from the function generator (5Vpp). CH2 is the output to the fuel injection controller.

As can be seen from the actual measurement graph the Automotive Interface Transmit Circuitry operates better than simulated. When a 5Vpp pulse enters the circuit from the microprocessor it creates a 0Vpp pulse that goes to the Fuel Injection Controller. When a 0Vpp pulse enters the circuit from the microprocessor it creates a 12Vpp pulse that goes to the Fuel Injection Controller. On the simulation the output voltage peak voltage is only 6.5 volts not 12volts. This was caused by using a 10KΩ input impedance for the Fuel Injection Controller. The actual measurements were taken using a 1MegΩ input impedance of the Scope.

I. Problems and Actions

The monitoring tool has a DB-9 female connector placed on the left 8.5” x 4.75” side. This is different from the specification that states that the DB-9 connector should be placed on the bottom of the monitoring tool. This was done in order to meet the size constraints of the box chosen for the monitoring tool.

The data logger case has a water proof female GM connector in place of the SAE J1962 connector which is in the specifications. This was changed in order to make the data logger case as water tight as possible. The DB-9 connector stated in the specifications was integrated into the GM connector to decrease the amount of connectors needed, also this makes the data logger case more rain proof.

All tests involving the Fuel Injection Controller were substituted with a 68HC11 EVBU board to emulate the FIC. This was done because at the scheduled testing time the FIC was not complete.

The vibration test involving the Formula-SAE race car was substituted with a manual shaking of the data logger. This was done because at the scheduled testing time the race car was not complete.

The PC software is not completed, due to time constraints. So the PC software could not be tested. The PC Software is scheduled to be completed during Eric Holland’s Graduate year. (Sept. 2002- May 2003)

VI. Project Summary

A. Overall Design Success

I am very pleased with the versatility and flexibility I put into this design. I made as many features as I could software determinable. That way if I need to make a change, the change can be made in software and not require and modifications to the circuit. The use of many test points and zero ohm resistors make hardware modifications extremely easy. This board can also be used for many different RF projects involving the 68HC11-evaluation board.

With the current Wireless Data Logger design a PC is still needed to change the performance tables and then the PC sends the new tables to the Monitoring Tool; which uploads the information to the Fuel Injection Controller via the RF link. In future development of this tool the programming of the performance tables could be done on the Monitoring Tool thus eliminating the need for a PC.

Also in the future development of this product a graphical LCD screen could be used to display the data instead of just using a 20x4 Character screen. This would allow the technicians to view sensor data over time in the form of a plot; also a graphical display offers more flexibility in the format of the information shown on the screen.

B. Degree of Simulation

In the simulation of the automotive interface circuitry, good results were found. In a worse case scenario the inputs and outputs of the circuit are within the specified ranges. These simulations were done with + - 10% resistor when in actuality + - 1% SMT resistors will be used, thus bringing that actual results closer to the simulated best-case scenario. Even though the results of the simulations were good, the coverage of these simulations was not a very large percentage of the project. This was just one of many critical paths. This was about 25% coverage of the project.

C. Conclusion and Recommendations

The design of the Wireless Data Logger is easily marketable. With the use of less expensive RF modules and buying parts in larger quantities the price could be made very competitive. The problem with this design is that it is only compatible with Cliff's Fuel Injection Controller. This unit was not made universal due to lack of resources.

Here is my list of future improvements:

1. Design in the ISO 9141-2 protocol, so the Data logger can be connected to any automotive (1996 or newer) and operate well.
2. Make use of Compact Flash cards for application code and Data storage. This would make upgrading the unit's memory very easy.
3. Design in a Larger Backlit LCD screen.
4. Make the entire Data Logger on a single board.
5. Make the entire Monitoring Tool on a single board.
6. Design in the capability of having Monitoring Tool program the Fuel Injection Controller with out the use of a PC.

VII. Attachments

Section 1: References

Section 2: Original Product Specifications

Section 3: Test Plan

Section 4: PSPICE Simulation File

Section 5: Software Code

Section 6: Design Computations

Section 7: Bill of Materials

Section 8: Schematics & Printed Circuit Board Layout

Section 9: Engineering Change Orders

Section 10: Product Drawings

Section 11: Product Pictures

Bibliography

1. "HC11 M68HC11 E Series Technical Data Book," Motorola Inc. 1995

2. "HC11 M68HC11 E Series Reference Manual," Motorola Inc. 1991

3. "CME11E9-EVBU Development Board Manual," Axiom Manufacturing 1999

4. "MC68HC11: An Introduction Software and Hardware Interfacing," Han-Way Huang 2001

5. "ISO 9141-2 Road Vehicles Diagnostic Systems," International Standard Organization 1994

6. "Interface Circuits for TIA/EIA-232-F," Texas Instruments Design Noted November 1998

7. "SAE J1979 Diagnostic Test Modes," Society of Automotive Engineers January 1997

8. "SAE J1962 Vehicle and Test Equipment Connector Identification." Society of Automotive Engineers June 1994

9. "SAE J1978 OBDII Scan Tool Specifications," Society of Automotive Engineers June 1994

10. "How Fuel Injection Systems Work," http://www.howstuffworks.com September 2001

11. "How Car Computers Work," http://www.hotstuffworks.com November 2001

Experts Consulted

1. Phillip McGee, Electrical Engineer, SPX, concerning OBDII communications protocols.

2. Dr. Han-way Huang, ECET Professor, MSU, concerning C++ programming for the 68HC11.

3. Kurt Raichle, Design Engineer, SPX, concerning ISO 9141-2 communications.

4. Mark Christensen, Design Engineer, DataRadio, concerning choices for an RF module.

5. Dr. Hudson, ECET Professor, MSU, concerning my design review.

6. Remis Norvilis, Engineering Student, concerning memory mapping the keypad during my design review.

7. Ted Yoder, Engineering Student, concerning my design review.

8. Cliff Braunesreither, CET Student, concerning the Fuel Injection Controller Interface. And sensor data parameters.

9. Dr. Bruce Jones, AET Professor, concerning proof of concept.

Referenced Specifications

1. CME11E9-EVBU Schematic from Axiom Manufacturing

Section 2: Specifications

Form

1.1 The wireless data logger consists of two pieces; one is the data logger, which is placed in the car next to the fuel injection controller. The other is a handheld monitoring tool.

- 1.2 Data Logger Case
  - 1.2.1 The data logger will be housed in a RF shielded metal case.
  - 1.2.2 The case will be a box no larger than 1' x 1' x 4''.
- 1.3 Monitoring Tool Case
  - 1.3.1 The monitoring tool will be housed in a plastic container.
  - 1.3.2 The tool will be handheld and no larger than 1' x 6'' x 3''.

2.1 The monitoring tool:

- 2.2 The front face of the monitoring tool has the following:
  - 2.2.1 A 20x4 character LCD screen will be in the top half of the 1' x 6'' side (See picture for location).
  - 2.2.2 A 4-button keypad will be below the LCD screen (See picture for location)
    - 2.2.2.1 The functions of the buttons will be
      - 2.2.2.1.1 Up
      - 2.2.2.1.2 Down
      - 2.2.2.1.3 F1
      - 2.2.2.1.4 F2
- 2.3 The top 6'' x 3'' side of the monitoring tool has the following:
  - 2.4.2 A push button power switch (See picture for location).
  - 2.4.3 A black 6.5'' long RF antenna (See picture for location).
  - 2.4.4 An external AC adapter connector (See picture for location).
- 2.5 The bottom 6'' x 3'' side of the monitoring tool has the following:
  - 2.5.1 A female DB-9 RS-232 serial connector (See picture for location).

3.1 The data logger has the following features:

- 3.2 A SAE J1962 communications jack (See picture for location).
- 3.3 A female DB-9 RS-232 serial connector (See picture for location).
- 3.4 A black 6.5'' long RF antenna (See picture for location).

Fit

1.1 The data logger operates on the Formula-SAE race car.

1.2 The unit will be able to handle the vibrations of the car.

1.3 The unit will operate over the temperature range 0 to 55˚C in a dry environment.

2.1 The monitoring tool can be power by an internal battery pack or an external AC Adapter power supply.

2.2 The tool will operate for at least 2 hours on 9.6Volt 1.1Ah battery pack

2.3 The AC adapter is connected to the plug on the top of the Monitoring tool (See picture for location).

3.1 The circuitry on both the data logger and monitoring tool will:

3.2 Follow FCC regulations on the wireless transmissions.

3.3 Follow RS-232 communication protocol when interfacing with a PC.

4.1 The data logger will be powered by the car’s 12Volt CBR600 motorcycle battery.

4.2 The data logger receives power from the connection with the Fuel injection controller.

4.3 The data logger will draw no more than 2 Amps from the battery.

Function

1.1 The data logger will communicate with the fuel injection controller, monitoring tool, and a PC.

1.2 The data logger is activated and deactivated by the Fuel injection controller.

1.2.1 When the key is in the ignition the Fuel injection controller turns on and then the data logger is activated.

1.2.2 When the key is removed from the ignition the Fuel injection controller turns off and then the data logger is deactivated.

1.3 The data logger will request sensor data from the fuel injection controller via a serial link 4 times a second following a protocol specified in the Design Documentation.

1.3.1 Sensor Data includes: Oxygen content of fuel, Throttle position, Coolant Temp, Voltage of Battery, Manifold pressure, and engine speed.

1.4 The logger will receive the sensor data from the controller following a protocol specified in the Design Documentation.

1.5 The logger will then store the data in volatile memory.

1.5.1 Up to 32K bytes of data can be stored.

1.5.2 When the memory is full the newest data will be stored over the oldest data. In a stack configuration.

1.6 The logger will sense the RF signal strength from the monitoring tool.

1.6.1 The signal strength is measured to determine if the Monitoring tool is within range (200 ft.) of the Data logger.

1.7 When the monitoring tool is within range (200 ft) the logger transmits the stored sensor data to the monitoring tool via a wireless link (433MHz AM modulated). The wireless transmission will follow the RS-232 protocol.

1.7.1 When the signal strength falls below acceptable levels, the logger stops transmitting to the monitoring tool.

1.7.1.1 The acceptable level will be determined by future research.

1.8 When the logger is connected to a PC via an RS-232 serial link.

1.8.1 The PC will request the sensor data via the serial link using RS-232 protocol.

1.8.1.1 This is done by the user selecting an option in the PC software’s menu.

1.8.2 Upon receiving the request the Data logger will transmit all of the stored data to the PC.

1.8.3 The PC will then interpret the data and display in on the screen in graphical or table format. The user then has the option to print or save the data to the PC.

2.1 The monitoring tool will communicate with the data logger and a PC.

2.2 The monitoring tool can be turned on / off via a power switch on the top of the tool (See picture for location).

DOI: 10.56816/2378-6949.1179

2.3 The monitoring tool will transmit an AM modulated RF signal at 433MHz to the data logger.

2.3.1 When the data logger receives the signal it measures the signal strength.

2.3.1.1 The signal strength is measured to determine if the Monitoring tool is within range (200 ft.) of the Data logger.

2.4 When the monitoring tool is within range (200 ft), the data logger will send the stored sensor data to the monitoring tool via a serial wireless link. The wireless transmission will follow the RS-232 protocol.

2.5 The monitoring tool will store the sensor data in volatile memory.

2.5.1 Up to 32K bytes of data can be stored.

2.5.2 When the memory is full the newest data will be stored over the oldest data. In a stack configuration.

2.6 The monitoring tool will display the selected sensor data on the LCD screen.

2.6.1 The user chooses with sensor data to display by pressing the UP / Down buttons to select different menu options.

2.6.2 When the monitoring tool is connected to a PC via an RS-232 serial link.

2.6.2.1 The PC will request the sensor data via the serial link using RS-232 protocol.

2.5.2.1.1 This is done by the user selecting an option in the PC software's menu.

2.6.2.2 Upon receiving the request the Monitoring Tool will transmit all of the stored data to the PC.

2.6.2.3 The PC will then interpret the data and display in on the screen in graphical or table format. The user then has the option to print or save the data to the PC

3.1 The PC software will have the following features for the user to select via a pull down (Windows style) menu.

3.2 Retrieve sensor data from the monitoring tool.

3.2.1 The user connects the monitoring tool to the PC serial port. Then the user starts program and chooses "Receive Data" from the pull down menu. The PC will then transmit a code to the monitoring tool via the serial port. Then the monitoring tool will send the stored data to the PC. Then PC will then interpret the data and display it on the screen.

3.3 Save sensor data to a text file.

3.3.1 The user chooses "Save" from the pull down menu. The program then prompts the user to enter a file name and directory. The program then saves the data in a text file.

3.4 Restore and display past saved sensor data files.

3.4.1 The user chooses "Restore" from the pull down menu. The program then prompts the user to enter a file name and directory. The program then opens the data and displays it on the screen.

3.5 Print sensor data.

3.5.1 The user chooses "Print" from the pull down menu. The program then prints the data to a printer connected to the parallel port.

3.6 Update the fuel injection controller's engine performance tables.

3.5.1.1 The user selects an option in the PC software's menu to "Change ECU tables"

3.5.1.2 Then the user enters the new performance values into a table.

3.5.1.3 Then user then selects an option to "Transmit Table to ECU"

3.5.1.4 The PC then transmits the new table information to the monitoring tool via the RS-232 serial link.

3.5.1.5 The monitoring tool then transmits the new table information to the data logger via the RF wireless link.

3.5.1.6 The Data logger then transmits the new engine tables to the Fuel injection controller via the ISO 9141-2 serial link.

Section 3: Test Plan

Form

1.1 The wireless data logger consists of two pieces; one is the data logger, which is placed in the car next to the fuel injection controller. The other is a handheld monitoring tool.

1.2 Data Logger Case

1.2.1 The data logger will be housed in a RF shielded metal case.

The container will be visually inspected for compliance.

1.2.2 The case will be a box no larger than 1’ x 1’ x 4’’.

The container will be measured using a ruler and the dimensions recorded to within +/-0.125”.

1.3 Monitoring Tool Case

1.3.1 The monitoring tool will be housed in a plastic container.

The container will be visually inspected for compliance.

1.3.2 The tool will be handheld and no larger than 1’ x 6’’ x 3’’.

The container will be measured using a ruler and the dimensions recorded to within +/-0.125”.

2.1 The monitoring tool:

2.2 The front face of the monitoring tool has the following:

2.2.1 A 20x4 character LCD screen will be in the top half of the 1’ x 6’’ side (See picture for location).

The front face will be visually inspected for compliance

2.2.2 A 4-button keypad will be below the LCD screen (See picture for location)

The keypad will be visually inspected for compliance

2.2.2.1 The functions of the buttons will be

2.2.2.1.1 Up

2.2.2.1.2 Down

2.2.2.1.3 F1

2.2.2.1.4 F2

2.6 The top 6’’ x 3’’ side of the monitoring tool has the following:

The topside will be visually inspected for compliance of these features.

2.6.1 A push button power switch (See picture for location).

2.6.2 A black 6’’ long RF antenna (See picture for location).

2.6.3 An external AC adapter connector (See picture for location).

2.7 The bottom 6’’ x 3’’ side of the monitoring tool has the following:

2.7.1 A female DB-9 RS-232 serial connector (See picture for location).

3.1 The data logger has the following features:

The data logger case will be visually inspected for compliance of these features.

3.2 A SAE J1962 communications jack (See picture for location).

3.3 A female DB-9 RS-232 serial connector (See picture for location).

3.4 A black 6'' long RF antenna (See picture for location).

Fit

1.1 The data logger operates on the Formula-SAE race car.

1.2 The unit will be able to handle the vibrations of the car.

The data logger will be mounted on the Formula-SAE car. The car will then be driven over terrain similar to the terrain of the racetrack the car will be raced on.

1.3 The unit will operate over the temperature range 0 to 55˚C in a dry environment.

The data logger will be placed in an environmental chamber at DATARADIO. The chamber's temperature will be varied from 0 - 55°C. The data logger will be tested by connecting a ECU emulator and using the monitoring tool to see if the correct data is being sent via the RF link.

2.1 The monitoring tool can be power by an internal battery pack or an external AC Adapter power supply.

2.2 The tool will operate for at least 2 hours on 9.6Volt 1.1Ah battery pack

The monitoring tool will have a completely charged battery place in it. The tool will be left on until the unit stops operating. Time for the battery to discharge will be recorded.

2.3 The AC adapter is connected to the plug on the top of the Monitoring tool (See picture for location).

An AC adapter will be plugged in the monitoring tool. The monitoring tool will then be turned on and all operations will be verified.

3.1 The circuitry on both the data logger and monitoring tool will:

3.2 Follow FCC regulations on the wireless transmissions.

A spectrum Analyzer will be used to measure the RF output power of the Data Logger and the Monitoring Tool. These measurements will be compared to the proper FCC documents.

3.3 Follow RS-232 communication protocol when interfacing with a PC.

The TX232 and RX232 lines will be monitored with a scope and compared to the IEEE standard for RS-232 communications.

4.1 The data logger will be powered by the car's 12Volt CBR600 motorcycle battery.

4.2 The data logger receives power from the connection with the Fuel injection controller.

The Fuel Injection Controller will be plugged into the data logger. The data logger will then be turned on and all operations will be verified.

4.3 The data logger will draw no more than 2 Amps from the battery.

A multimeter will be put in line with the incoming power from the Fuel Injection Controller and the input current to the data logger will be measured.

Function

1.1 The data logger will communicate with the fuel injection controller, monitoring tool, and a PC.

1.2 The data logger is activated and deactivated by the Fuel injection controller.

1.2.1 When the key is in the ignition the Fuel injection controller turns on and then the data logger is activated.

The data logger will be connected the FIC and the key will be inserted into the ignition. A multimeter will be used to verify if power is applied to the Data logger.

1.2.2 When the key is removed from the ignition the Fuel injection controller turns off and then the data logger is deactivated.

The data logger will be connected the FIC and the key will be removed from the ignition. A multimeter will be used to verify if power is no longer applied to the Data logger.

1.3 The data logger will request sensor data from the fuel injection controller via a serial link 4 times a second following the ISO 9141-2 protocol.

A FIC emulator will be connected to the data logger. A scope with be connected to the TX and RX lines and the rated of transmission will be verified.

1.3.1 Sensor Data includes: Oxygen content of fuel, Throttle position, Coolant Temp, Voltage of Battery, Manifold pressure, and engine speed.

1.4 The logger will receive the sensor data from the controller following ISO 9141-2 protocol.

A scope will be connected to the TX and RX lines and the transmission protocol will be verified.

1.5 The logger will then store the data in volatile memory.

1.5.1 Up to 32K bytes of data can be stored.

The size of the SRAM used will be verified by the manufacturer's data sheet.

1.5.2 When the memory is full the newest data will be stored over the oldest data. In a stack configuration.

The data logger will be connected to the FIC emulator and data will be transmitted and stored into the 32K of SRAM on the Data Logger. When over 32K of data has been sent to the data logger, a PC will be connected and the information in the SRAM will be downloaded to the PC. The data will then be analyzed to verify the oldest data has been overwritten.

1.6 The logger will sense the RF signal strength from the monitoring tool.

1.6.1 The signal strength is measured to determine if the Monitoring tool is within range (400 ft.) of the Data logger.

The range of the RF communications will be tested by moving the monitoring tool away from the data logger and observing the data received by the monitoring tool on the LCD screen. The distance between the two will be measured with a tape measure.

1.7 When the monitoring tool is within range (400 ft) the logger transmits the stored sensor data to the monitoring tool via a wireless link (433MHz AM modulated).

1.7.1 When the signal strength falls below acceptable levels, the logger stops transmitting to the monitoring tool.

The range of the RF communications will be tested by moving the monitoring tool away from the data logger and observing the data received by the monitoring tool on the LCD screen. The distance between the two will be measured with a tape measure.

1.7.1.1 The acceptable level will be determined by future research.

1.8 When the logger is connected to a PC via an RS-232 serial link.

1.8.1 The PC will request the sensor data via the serial link using RS-232 protocol.

1.8.1.1 This is done by the user selecting an option in the PC software's menu.

This will be tested by connecting a scope to the serial port on the PC and observing the data stream.

1.8.2 Upon receiving the request the Data logger will transmit all of the stored data to the PC.

This will be tested by connecting a scope to the serial port on the PC and observing the data stream.

1.8.3 The PC will then interpret the data and display in on the screen in graphical or table format. The user then has the option to print or save the data to the PC.

This will be tested by either saving the data to a disk or printing the data from the PC software.

2.1 The monitoring tool will communicate with the data logger and a PC.

2.2 The monitoring tool can be turned on / off via a power switch on the top of the tool (See picture for location).

The AC adapter will be inserted into the monitoring tool and the power switch will be pressed. A multimeter will be used to verify if power is applied to the Data logger.

2.3 The monitoring tool will transmit an AM modulated RF signal at 433MHz to the data logger.

2.3.1 When the data logger receives the signal it measures the signal strength.

2.3.1.1 The signal strength is measured to determine if the Monitoring tool is within range (400 ft.) of the Data logger.

The range of the RF communications will be tested by moving the monitoring tool away from the data logger and observing the data received by the monitoring tool on the LCD screen. The distance between the two will be measured with a tape measure.

2.4 When the monitoring tool is within range (400 ft), the data logger will send the stored sensor data to the monitoring tool via a serial wireless link.

The range of the RF communications will be tested by moving the monitoring tool away from the data logger and observing the data received by the monitoring tool on the LCD screen. The distance between the two will be measured with a tape measure.

2.5 The monitoring tool will store the sensor data in volatile memory.

2.5.1 Up to 32K bytes of data can be stored.

The size of the SRAM used will be verified by the manufacturer's data sheet.

2.5.2 When the memory is full the newest data will be stored over the oldest data. In a stack configuration.

The monitoring tool will be connected to the data logger via the RF link and data will be transmitted and stored into the 32K of SRAM on the Monitoring tool. When over 32K of data has been sent to the data logger, a PC will be connected and the information in the SRAM will be downloaded to the PC. The data will then be analyzed to verify the oldest data has been overwritten.

2.6 The monitoring tool will display the selected sensor data on the LCD screen.

2.6.1 The user chooses with sensor data to display by pressing the UP / Down buttons to select different menu options.

This will be tested by loading the application code into the monitoring tool and pressing the buttons to see if the data is displayed.

2.6.2 When the monitoring tool is connected to a PC via an RS-232 serial link.

2.6.2.1 The PC will request the sensor data via the serial link using RS-232 protocol.

This will be tested by connecting a scope to the serial port on the PC and observing the data stream.

2.5.2.1.1 This is done by the user selecting an option in the PC software's menu.

2.6.2.2 Upon receiving the request the Monitoring Tool will transmit all of the stored data to the PC.

This will be tested by connecting a scope to the serial port on the PC and observing the data stream.

2.6.2.3 The PC will then interpret the data and display in on the screen in graphical or table format. The user then has the option to print or save the data to the PC

This will be tested by either saving the data to a disk or *printing the data from the PC software.*

3.1 The PC software will have the following features for the user to select via a pull down (Windows style) menu.

3.2 Retrieve sensor data from the monitoring tool.

3.2.1 The user connects the monitoring tool to the PC serial port. Then the user starts program and chooses "Receive Data" from the pull down menu. The PC will then transmit a code to the monitoring tool via the serial port. Then the monitoring tool will send the stored data to the PC. Then PC will then interpret the data and display it on the screen.

This will be tested by either saving the data to a disk or *printing the data from the PC software.*

3.3 Save sensor data to a text file.

3.3.1 The user chooses "Save" from the pull down menu. The program then prompts the user to enter a file name and directory. The program then saves the data in a text file.

This will be tested by saving the data to a disk from *the PC software.*

3.4 Restore and display past saved sensor data files.

3.4.1 The user chooses "Restore" from the pull down menu. The program then prompts the user to enter a file name and directory. The program then opens the data and displays it on the screen.

This will be tested loading the data from a disk to *the PC software.*

3.5 Print sensor data.

3.5.1 The user chooses "Print" from the pull down menu. The program then prints the data to a printer connected to the parallel port.

This will be tested by printing the data from the PC software to a printer connected to the parallel port.

3.6 Update the fuel injection controller's engine performance tables.

This will be tested by visually inspecting the user interface of the program, and observing the car's performance with the monitoring tool.

3.5.1.1 The user selects an option in the PC software's menu to "Change ECU tables"

3.5.1.2 Then the user enters the new performance values into a table.

3.5.1.3 Then user then selects an option to "Transmit Table to ECU"

3.5.1.4 The PC then transmits the new table information to the monitoring tool via the RS-232 serial link.

3.5.1.5 The monitoring tool then transmits the new table information to the data logger via the RF wireless link.

3.5.1.6 The Data logger then transmits the new engine tables to the Fuel injection controller via the ISO 9141-2 serial link.

Circuitry

Overview: The data logger and monitoring tool will each have an 8-bit microprocessor controlling the different pieces of the hardware. On the monitoring tool the microprocessor will control the RS-232 circuitry, RF modules, Keypad circuitry, and LCD memory mapping circuitry. On the Data Logger the microprocessor will control the RS-232 circuitry, RF modules, and automotive buffering circuitry.

Microprocessor and supporting circuitry: The microprocessor will be loaded with the application code. The system will be reset several times. During this time all I/O channels will be monitored and compared to expected waveforms.

RS-232 Circuitry: This circuitry will be tested by connecting it to a PC and using Hyper Terminal send data back and forth. During this time a scope will be monitoring the serial port the data stream. This data will be compared to the expected waveforms.

RF Modules: A serial data stream will be sent to the Transmit module by a 68HC11 EVBU board. The receiver module will be connected to a scope. The waveform will be compared to the expected waveform.

Keypad Circuitry: The circuitry will be powered up and a multimeter will be used to check logic levels when the different buttons are pressed.

LCD Circuitry: The circuitry will be connected to the 68HC11 EVBU and code will be written to display "Eric Is Cool" in the screen. If this works the LCD circuitry is correct.

Automotive Buffers: A power supply will be connected to the input lines of these buffers. The voltage of the power supply will be varied from 0-50V to ensure the circuitry is properly protected.

Equipment Required

The Equipment required is:

13. Multimeter
14. Oscilloscope
15. Logic Analyzer
16. 68HC11 EVBU
17. Fuel Injection Controller
18. FIC Emulator
19. Formula-SAE Race car
20. DATARADIO's Environmental Chamber
21. PC
22. Ruler
23. Tape Measure
24. Battery Charger

All items except DATARADIO's Environmental Chamber can be found in the senior design lab.

Expected Fault Coverage

Form: The identified tests cover all the items under this are of the specification. I therefore expect 100% coverage of form.

Fit: The identities test cover, to some degree, all items under this heading. I will not be able to directly test the specified vibration test, however it is expected that the planned test run on the car will provide a reasonable approximation. I would therefore expect a degree of cover age of approximately 75%.

Function: The identified tests will address all major functions, however cannot check them under all conditions. Some areas of deficiency include – checking range of the RF modules, PC software functionality, Fuel injection table update, and RS-232 waveform comparison. . I would therefore expect a degree of cover age of approximately 70%.

DOI: 10.56816/2378-6949.1179

Section 4: PSPICE Simulation File

```
********* e:\simulations\automotive.ewb **********
*  Interactive Image Technologies                 *
*                                                 *
*  This File was created by:                      *
*    Electronics Workbench to SPICE netlist       *
*    conversion DLL                               *
*                                                 *
*  Sat Apr 27 19:11:58 2002                       *
**************************************************

* Battery(s)
*
V2 5 0 DC 12
*
V3 6 0 DC 12
*
V4 8 0 DC 5
*
V5 10 0 DC 5

* Resistor(s)
*
R5 4 5 10K
*
R7 11 10 10K
*
R4 3 6 10K
*
R2 2 3 1K
*
R3 1 4 1K
* K line input Res.
R1 0 1 10K
*
R6 7 9 1K
*
R8 12 10 10K

* Diode(s)
*
D1 0 3 D_ideal
*
D2 3 6 D_ideal
*
D3 0 4 D_ideal
*
D4 4 5 D_ideal

* NPN Transistor(s)
*
Q1 4 7 0 Qnideal

* Connector(s)
* node = 9, label =
* node = 8, label =
* node = 0, label =
* node = 0, label =
* node = 12, label =
* node = 10, label =
* node = 8, label =
* node = 9, label =
* node = 6, label =
```

```
* node = 7, label =
* node = 0, label =
* node = 2, label =
* node = 1, label =
* node = 1, label =

* Clock(s)
* L line 12V
V1 2 0 PULSE(0 12 0 1n 1n 500u 1m)

* Comparator(s)
*
XCOMP_VR1 3 11 12 comp_ideal

* Misc
.MODEL D_ideal D(Is=10f Rs=0 Cjo=0 Vj=1 Tt=0 M=500m BV=1e+30 N=1 EG=1.11
+XTI=3 KF=0 AF=1 FC=500m IBV=1m TNOM=27)

.MODEL Qnideal NPN(Is=1e-16 BF=100 BR=1 Rb=0 Re=0 Rc=0 Cjs=0 Cje=0 Cjc=0
+Vje=750m Vjc=750m Tf=0 Tr=0 mje=330m mjc=330m VA=1e+30 ISE=0 IKF=1e+30
+Ne=1.5 NF=1 NR=1 VAR=1e+30 IKR=1e+30 ISC=0 NC=2 IRB=1e+30 RBM=0 XTF=0
+VTF=1e+30 ITF=0 PTF=0 XCJC=1 VJS=750m MJS=0 XTB=0 EG=1.11 XTI=3 KF=0
AF=1
+FC=500m TNOM=27)

.SUBCKT comp_ideal 1 2 3
    R0 1 0 1e9
    R1 2 0 1e9
    VposPwr 4 0 3.5
    VnegPwr 5 0 230m
    A0 %vd(1 2) %g(4) %g(5) %gd(6 0) comp_ideal_curlimit
    R2 6 0 1e9
    A1 %vd(6 0) %vd(3 0) comp_ideal_slew
    R3 3 0 1e9
.ENDS
.MODEL comp_ideal_curlimit ilimit(in_offset=700m gain=200K
+   r_out_source=275.49 r_out_sink=454.878
+   i_limit_source=6m i_limit_sink=5m
+   v_pwr_range=1.0u i_source_range=1.0n i_sink_range=1.0n
+   r_out_domain=1.0n)
.MODEL comp_ideal_slew slew(rise_slope=19.7703MEG fall_slope=17.2796MEG)

.OPTIONS ITL4=25
.END
```

Section 5: Software Code

```
*****************************************************************
* Wireless Data Logger                                          *
* Copyright (c) 2002 by: Eric Holland                           *
* ALL RIGHTS RESERVED                                           *
*****************************************************************
*****************************************************************
* Date: 04/18/2002  Coded by: Eric Holland                      *
* Filename: Monitor.asm                                         *
*                                                               *
*****************************************************************
*          Description                                          *
*This progam is the Monitoring Tool Application Code for the Wireless
Data Logger. *
*                                                               *
*****************************************************************
LCDBAS  EQU     $B5F0
LCDDAT  EQU     $B5F1
PORTE   EQU     $0A
DDRD    EQU     $09
PORTD   EQU     $08
PORTA   EQU     $00
TIC3    EQU     $14
TCTL2   EQU     $21
TCTL1   EQU     $20
TFLG1   EQU     $23
TMSK1   EQU     $22
TCNT    EQU     $0E
TOC1    EQU     $16
TOC4    EQU     $1C
KEYPAD  EQU     $B5A0
* RF Addresses *******
TX_addr EQU     $B5C0
RX_addr EQU     $B5B0
TX_data EQU     $B580
RX_data EQU     $B590
* SCI stuff *********
BAUD     EQU  $2B  sci baud reg
SCCR1    EQU  $2C  sci control1 reg
SCCR2    EQU  $2D  sci control2 reg
SCSR     EQU  $2E  sci status reg
SCDR     EQU  $2F  sci data reg
TDRE     EQU  $80
RDRF     EQU  $20

Scroll   EQU    5       ;This is the variable that selects the number of
options in a menu

* For RAM *************
*       ORG     $E2
*       JMP     KEYPAD_INT      ;IC3 interrupt Jump Vector
*       ORG     $E5
*       JMP     RFVT_INT        ;IC2 interrupt Jump Vector
*       ORG     $00C4
*       JMP     RS232_INT       ;SCI interrupt Jump Vector
* For EEPROM *********
        ORG     $FFD6
        FDB     RS232_INT
        ORG     $FFEA
        FDB     KEYPAD_INT
        ORG     $FFEC
        FDB     RFVT_INT
```

```
        ORG     $1040           ; Starting place for RAM
Line    RMB     1               ; Variable to let me know what line in a
menu is selected
Line1   RMB     2
Line2   RMB     2
Line3   RMB     2
Line4   RMB     2
Screen  RMB     1               ; Variable to let me know what screen is
on the LCD
Key     RMB     1               ; Lets me know what key was pressed
* Sensor Data *********
CTS     RMB     1
RPM     RMB     1
AIR     RMB     1
MAP     RMB     1
O2      RMB     1
TPS     RMB     1
BATT    RMB     1
TEMP1   RMB     1
OrderRFRX RMB   1
DATRFRX  RMB    1
Range   RMB     1               ; Is the RF connection Good variable
TEMP4   RMB     1
TEMP    RMB     3               ; Used as a buffer for displaying data
ZERO    RMB     1               ; Needed to know when to stop printing
data to screen

* Screen Shots ********
        ORG     $E000 SCREEN1
FCB             $20,$FF,$FF,$FF
        FCC     "UTOMOTIVE        "
        FCB     $20,$FF,$20,$FF,$20,$FF,$FF,$FF
        FCC     "NTERFACE     "
        FCB     $20,$FF,$FF,$FF,$20,$20,$FF,$20,$20,$FF,$FF,$FF
        FCC     "OOLING   "
        FCB     $20,$FF,$20,$FF,$20,$FF,$FF,$FF,$20,$20,$FF,$20,$20
        FCC     "Rev 1  "
        FCB     0
SCREEN2 FCB     $20,$FF,$FF,$FF,$FF
        FCC     " WIRELESS "
        FCB     $FF,$FF,$FF, $FF,$20,$20
        FCC     "    DATA   LOGGER     "
        FCC     "                      "
        FCC     " Menu         Setup "
        FCB     0
SCREEN3 FCB     $FF,$20
        FCC     "DIAGNOSTICS MENU"
        FCB     $20,$FF
        FCC     " >"
        FCB     0
        FCC     "  1. Sensor Data     "
        FCB     0
        FCC     "  2. Car Dash Data   "
        FCB     0
        FCC     "  3. Program FIC     "
        FCB     0
        FCC     "  4.                 "
        FCB     0
        FCC     "  5.                 "
        FCB     0
        FCC     "                     "
        FCB     0
        FCC     "Enter         Exit "
        FCB     0
```

```
SCREEN4 FCB     $FF,$FF,$FF,$FF
        FCC     " SENSOR DATA "
        FCB     $FF,$FF,$FF,$20 FCC
        " O2=   % CTS=  C       "
        FCC     "AIR=   C MAP=   kPaG "
        FCC     "               Exit "
        FCB     0
SCREEN5 FCB     $FF,$FF,$FF
        FCC     " CAR DASH DATA "
        FCB     $FF,$FF,$20
        FCC     "ENGINE SPEED=   KRPM "
        FCC     "TPS=   %  BATT= . V "
        FCC     "               Exit "
        FCB     0
SCREEN6 FCB     $20,$FF,$FF,$FF
        FCC     " SETUP MENU "
        FCB     $FF,$FF,$FF
        FCC     "  >"
        FCB     0
        FCC     "  1. RF TX Channel   "
        FCB     0
        FCC     "  2. RF RX Channel   "
        FCB     0
        FCC     "  3. Send RS232      "
        FCB     0
        FCC     "  4. Help            "
        FCB     0
        FCC     "  5. Read Me         "
        FCB     0
        FCC     "                     "
        FCB     0
        FCC     "Enter         Exit "
        FCB     0
SCREEN7 FCB     $FF,$FF,$FF
        FCC     " RF TX CHANNEL "
        FCB     $FF,$FF
        FCC     " >"
        FCB     0
        FCC     "  1.  Default TX     "
        FCB     0
        FCC     "  2.  Default RX     "
        FCB     0
        FCC     "  3.  Unused         "
        FCB     0
        FCC     "  4.  Unused         "
        FCB     0
        FCC     "  5.  Unused         "
        FCB     0
        FCC     "                     "
        FCB     0
        FCC     "Select        Exit "
        FCB     0
SCREEN8 FCB     $FF,$FF,$FF
        FCC     " RF RX CHANNEL "
        FCB     $FF,$FF
        FCC     " >"
        FCB     0
        FCC     "  1.  Default TX     "
        FCB     0
        FCC     "  2.  Default RX     "
        FCB     0
        FCC     "  3.  Unused         "
        FCB     0
        FCC     "  4.  Unused         "
```

```
        FCB     0
        FCC     "  5.  Unused         "
        FCB     0
        FCC     "                     "
        FCB     0
        FCC     "Select        Exit "
        FCB     0
SCREEN9 FCB     $FF,$FF,$FF
        FCC     " RS232 SCREEN "
        FCB     $FF,$FF,$FF,$20
        FCC     "This will send the   "
        FCC     "data to the Computer "
        FCC     "Enable      Disable "
        FCB     0
SCREEN10        FCB     $FF,$FF,$FF,$FF,$FF,$FF,$FF
                FCC     " HELP "
                FCB     $FF,$FF,$FF,$FF,$FF,$FF,$FF,$20
                FCC     "Any Questions email  "
                FCC     "ericjohnholland@hotm "
                FCC     "ail.com        Exit "
                FCB     0
SCREEN11        FCB     $FF,$FF,$FF,$FF,$FF,$FF
                FCC     " READ ME "
                FCB     $FF,$FF,$FF,$FF,$FF,$20
                FCC     "Designed by:         "
                FCC     "Eric Holland May2002 "
                FCC     "Version 1.0    Exit "
                FCB     0
SCREEN12        FCB     $FF,$FF,$FF,$FF
                FCC     " PROGRAM FIC "
                FCB     $FF,$FF,$FF,$20 FCC
                "       NOT A CURRENT         "
                FCC     "       OPTION        "
                FCC     "               Exit "
                FCB     0
SCREEN13        FCC     " *** RFlink Lost *** "
                FCC    "     RF Connection     "
                FCC     "     To Car Lost!     "
                FCC     "                     "
                FCB     0
*****************************************************
MAIN
*************************************************************
*       ORG     $2400           ;For RAM
        ORG     $FFFE           ;set the reset vector
        FDB     $E600

        ORG     $E600           ;For EEPROM


        LDX     #$1000          ;sets Bias piot
        LDS     #$7FFF          ;Sets Stack Pionter

        JSR     Setup           ;Set up interrupts
        JSR     STARTUP         ;Run start up screens
        CLI                     ;Enables Interrupt

* 200ms Delay Loop for screen Refreashes *****************
Loop            PSHX
                LDX     #40000                  ;200ms LOOP
INNER4          NOP                             ;2 cycles
                NOP                             ;2 cycles
                DEX                             ;3 cycles
                BNE     INNER4                  ;3 cycles
```

```
                PULX

        INC     Range
        LDAA    Range
*       CMPA    #250            ;These commands are not needed
*       BLE     Next14
*       LDAA    #25
*       STAA    Range
Next14  CMPA    #25             ;If Range = 25 display Screen 13
        BLE     Next13
        LDY     #SCREEN13
        JSR     LCD OUT
*               LDAA    #$68
*               ADDA    #$80            ;set the posistion
*               STAA    LCDBAS
*               BRSET   0,X $80 *               ;delay
        JMP     Next12


Next13  JSR     ACTION          ;Updates any Key pressed data


* Displays Current Screen ***********
        LDAA    Screen
        CMPA    #2
                BNE     Next1
                LDY     #SCREEN2
                JSR     LCD OUT
Next1   CMPA    #3
                BNE     Next2
                JSR     LCDREF
                JSR     LCD OUT
Next2   CMPA    #4
                BNE     Next3
                LDY     #SCREEN4
                JSR     LCD OUT
Next3   CMPA    #5
                BNE     Next4
                LDY     #SCREEN5
                JSR     LCD OUT
Next4   CMPA    #12
                BNE     Next5
                LDY     #SCREEN12
                JSR     LCD OUT
Next5   CMPA    #6
                BNE     Next6
                JSR     LCDREF
                JSR     LCD OUT
Next6   CMPA    #7
                BNE     Next7
                JSR     LCDREF
                JSR     LCD OUT
Next7   CMPA    #8
                BNE     Next8
                JSR     LCDREF
                JSR     LCD OUT
Next8   CMPA    #9
                BNE     Next9
                LDY     #SCREEN9
                JSR     LCD OUT
Next9   CMPA    #10
                BNE     Next10
                LDY     #SCREEN10
                JSR     LCD_OUT
```

```
Next10  CMPA    #11
                BNE     Next11
                LDY     #SCREEN11
                JSR     LCD_OUT


Next11



        JSR     DATAupdate              ;Updates New data to the Screen
and in RS-232 buffers


Next12  JMP     Loop            ;Endless Loop


        SWI
        END
************************************************************************
*****************************************************
Setup
**************************************************************
        BSET    TCTL2,X %00000010       ;value to set falling edge
triggered
        BCLR    TFLG1,X %11111110       ;Clear IC3 flag
        BSET    TMSK1,X %00000001       ;Enables IC3 interrupt

        BSET    TCTL2,X %00100000       ;value to set Falling edge
triggered
        BCLR    TFLG1,X %11111011       ;Clear IC1 flag
        BSET    TMSK1,X %00000100       ;Enables IC1 interrupt

        BSET    TCTL2,X %00000100       ;value to set Rising edge
triggered
        BCLR    TFLG1,X %11111101       ;Clear IC2 flag
        BSET    TMSK1,X %00000010       ;Enables IC2 interrupt
* RS-232 Setup ************
         LDX  #$1000
         LDAA #$30
         STAA BAUD,X
         LDAA #$00
         STAA SCCR1,X
         LDAA #%00101100
         STAA SCCR2,X

                LDAA    #$3E            ;value to set D5-D2 for output
                STAA    DDRD,X          ;Sets PORTD for output
                LDAA    #%00000110
                STAA    PORTD,X         ;Startup values
                LDAA    #%00100101
                STAA    PORTA,X         ;Startup values
                LDAA    #4
                STAA    Screen
                LDAA    #1
                STAA    Line
                LDAA    #%111111111     ;TX_addr start up value
                STAA    TX_addr
                LDAA    #%111111111     ;RX_addr start up value
                STAA    RX_addr
                CLR     Key CTS
                CLR     RPM
                CLR     AIR
                CLR
```

```
CLR     MAP
CLR     O2
CLR     TPS
CLR     BATT
CLR     TEMP
CLR     TEMP+1
CLR     TEMP+2
LDAA    #223
STAA    O2
LDAA    #155
STAA    CTS
LDAA    #203
STAA    AIR
LDAA    #097
STAA    MAP
LDAA    #032
STAA    RPM
LDAA    #243
STAA    TPS
LDAA    #012
STAA    BATT
CLR     OrderRFRX
CLR     DATRFRX
CLR     Range
CLR     ZERO LDAA
        #$31
        STAA    O2_1
        LDAA    #$32
        STAA    O2_2
        LDAA    #$33
        STAA    O2_3
        LDAA    #$31
        STAA    CTS_1
        LDAA    #$32
        STAA    CTS_2
        LDAA    #$33
        STAA    CTS_3
        LDAA    #$31
        STAA    RPM_1
        LDAA    #$32
        STAA    RPM_2
        LDAA    #$33
        STAA    RPM_3
        LDAA    #$31
        STAA    AIR_1
        LDAA    #$32
        STAA    AIR_2
        LDAA    #$33
        STAA    AIR_3
        LDAA    #$31
        STAA    MAP_1
        LDAA    #$32
        STAA    MAP_2
        LDAA    #$33
        STAA    MAP_3
        LDAA    #$31
        STAA    TPS_1
        LDAA    #$32
        STAA    TPS_2
        LDAA    #$33
        STAA    TPS_3
        LDAA    #$31
        STAA    BATT_1
        LDAA    #$32
```

```
                        STAA    BATT_2
                        LDAA    #$33
                        STAA    BATT_3


        RTS
*************************************************************************
****************************************************
LCDREF
*************************************************************************
********************************
        LDY     Line1           ;LOAD value OF Line1 IN Y
        JSR     LCD OUT
        LDY     Line2           ;LOAD value OF Line2 IN Y
        JSR     LCD OUT
        LDY     Line3           ;LOAD value OF Line3 IN Y
        JSR     LCD OUT
        LDY     Line4           ;LOAD value OF Line 4 IN Y
        JSR     LCD_OUT
        RTS
*************************************************************************
*****************************************************
STARTUP
***************************************************************
        JSR     LCDSET          ;setups LCD Screen Parameters

        JSR     DATAupdate      ;Update RS-232 Buffers

        LDAA    #5
        STAA    Screen

        JSR     DATAupdate

        LDAA    #2
        STAA    Screen


        LDY     #SCREEN1        ;LOAD ADDRESS OF SCREEN1 IN Y
        JSR     LCD_OUT
                PSHX LDAB
                #50                     ;5 SECOND LOOP
OUTER           LDX     #20000          ;100ms LOOP
INNER           NOP
                NOP DEX
                BNE
                INNER DECB
                BNE
                OUTER PULX

        RTS
*************************************************************************
*****************************************************
LCD_OUT
***********************************************************
PUTSTRG LDAA    0,Y             ;OUTPUTS STRG ADDRESS IN Y
        BEQ     DONE            ;TO LCD SCREEN
        JSR     LCDWRAP
        INY
        BRA     PUTSTRG
DONE    RTS

*************************************************************************
*****************************************************
```

```
LCDSET
*CONFIGURES LCD SCREEN
*************************************************************
        PSHX
        LDX   #LCDBAS
        LDAA  #$3C              ; set 20x4 Display
        STAA  LCDBAS
        BRSET 0,X $80 *         ;delay
        LDAA  #$01              ; Clear & Home
        STAA  LCDBAS
        BRSET 0,X $80 *         ;delay
        LDAA  #$0C              ; Display on
        STAA  LCDBAS
        BRSET 0,X $80 *         ;delay
        LDAA  #$06              ; Cursor shift on
        STAA  LCDBAS
        BRSET 0,X $80 *         ;delay
        LDAA  #$14              ; Shift right
        STAA  LCDBAS
        LDAA  #$02              ; Cursor to Home
        BRSET 0,X $80 *         ;delay
        STAA  LCDBAS
        BRSET 0,X $80 *         ;delay
        PULX
        RTS
*************************************************************************
******************************************************
LCDWRAP
*************************************************************
        STAA    LCDDAT          ; display character
LCDLP   LDAA    LCDBAS          ; read next character position
        BMI     LCDLP           ; test if busy and wait if true
        CMPA    #$13            ; test for line 1 wrap
        BEQ     LCD1            ; if match, correct line wrap
        CMPA    #$53            ; test for line 2 wrap
        BEQ     LCD2            ; if match, correct line wrap
        CMPA    #$27            ; test for line 2 wrap
        BEQ     LCD3            ; if match, correct line wrap
        RTS
* correct line 1 wrap from line 3 to line 2
LCD1    LDAA    #$40            ; load line 2 start position
        ORAA    #$80            ; set command bit
        STAA    LCDBAS          ; write to display
        RTS
* correct line 2 wrap from line 4 to line 3
LCD2    LDAA    #$14            ; load line 3 start position
        ORAA    #$80            ; set command bit
        STAA    LCDBAS          ; write to display
        RTS
* correct line 3 wrap from line 2 to line 4
LCD3    LDAA    #$54            ; load line 4 start position
        ORAA    #$80            ; set command bit
        STAA    LCDBAS          ; write to display
        RTS
*************************************************************************
******************************************************
KEYPADSUB
**************************************************************
        LDAA    KEYPAD                  ;Load value of keypad from Data
Bus
        BCLR    PORTA,X %00100000       ;Turn on Speaker
                PSHX
                LDX     #30000          ;150ms LOOP
INNER_1         NOP
```

```
                NOP
                DEX
                BNE     INNER_1
                PULX
        BSET    PORTA,X %00100000       ;Turn off speaker

        LSRA                    ;shifts F1 bit into CCR carry bit
        BCC     F1
        LSRA                    ;Shifts F2 bit into CCR carry bit
        BCC     F2
        LSRA                    ;shifts UP bit into CCR carry bit
        BCC     UP
        LSRA                    ;Shifts DOWN bit into CCR carry bit
        BCC     DOWN
F1      LDAA    #1
        STAA    Key
        BRA     Done_1
F2      LDAA    #2
        STAA    Key
        BRA     Done_1
UP      LDAA    #3
        STAA    Key
        BRA     Done_1
DOWN    LDAA    #4
        STAA    Key
        BRA     Done_1

Done_1  RTS
*************************************************************************
*****************************************************
KEYPAD_INT
*************************************************************
        PSHA
        PSHB PSHX
        LDX
        #$1000
        JSR     KEYPADSUB
        BCLR    TFLG1,X %11111110       ;Clear IC3 flag
        PULX
        PULB
        PULA
        RTI
*************************************************************************
*****************************************************
ACTION
*************************************************************
        LDAA    Key
        CMPA    #1
        BNE     F2 1
        JMP     F1 2
F2 1    CMPA    #2
        BNE     UP 1
        JMP     F2 2
UP 1    CMPA    #3
        BNE     DOWN 1
        JMP     UP 2
DOWN 1  CMPA    #4
        BNE     BACK 1
        JMP     DOWN 2
BACK_1  JMP     BACK

****************************
UP_2            LDAA    Screen
                CMPA    #3              ;If screen 3 branch Sc_up
```

```
                BEQ     Sc_up
                CMPA    #6
                BEQ     Sc_up
                CMPA    #7
                BEQ     Sc_up
                CMPA    #8
                BEQ     Sc_up
                JMP     BACK
Sc_up           LDAA    Line
                CMPA    #1              ;If Line Not equal to 1 branch
S_up
                BNE     S_up
                JMP     BACK
S_up            LDD     Line2
                SUBD    #$16
                STD     Line2
                LDD     Line3
                SUBD    #$16
                STD     Line3
                DEC     Line            ;Line=Line-1
        JMP     BACK
*****************************
DOWN 2  LDAA    Screen
        CMPA    #3              ;If screen 3 branch Sc_down
        BEQ     Sc down
        CMPA    #6
        BEQ     Sc down
        CMPA    #7
        BEQ     Sc down
        CMPA    #8
        BEQ     Sc down
        JMP     BACK

Sc_down LDAA    Line
        CMPA    #Scroll-1       ;If Line Not equal to Scroll branch
S_down
        BLE     S_down
        JMP     BACK
S_down  LDD     Line2
        ADDD    #$16
        STD     Line2
        LDD     Line3
        ADDD    #$16
        STD     Line3
        INC     Line            ;Line =Line +1
        JMP     BACK
*****************************
F1_2    LDAA    Screen
        CMPA    #2              ;If screen 2 branch
        BEQ     S_2
        CMPA    #3              ;If screen 2 branch
        BEQ     S_3
        CMPA    #6
        BEQ     S_6
        JMP     BACK

        JMP     BACK

S_2     LDAA    #3
        STAA    Screen          ;Screen = 3
        LDD     #SCREEN3
        STD     Line1
        LDD     #SCREEN3+$17
        STD     Line2
```

```
        LDD     #SCREEN3+$2D
        STD     Line3
        LDD     #SCREEN3+$9B
        STD     Line4
        LDAA    #1
        STAA    Line
        JMP     BACK

S_3     LDAA    Line            ;Diag Menu Screen
        CMPA    #1
        BEQ     Sensor
        CMPA    #2
        BEQ     CarData
        CMPA    #3
        BEQ     ProgFIC
        JMP     BACK

Sensor  LDAA    #4
        STAA    Screen
        JMP     BACK

CarData LDAA    #5
        STAA    Screen
        JMP     BACK

ProgFIC LDAA    #12
        STAA    Screen
        JMP     BACK

S_6     LDAA    Line            ;Setup Menu Screen
        CMPA    #1
        BEQ     RFTX
        CMPA    #2
        BEQ     RFRX
        CMPA    #3
        BEQ     RS232
        CMPA    #4
        BEQ     Help
        CMPA    #5
        BEQ     Readme
        JMP     BACK

RS232   LDAA    #9
        STAA    Screen
        JMP     BACK

Help    LDAA    #10
        STAA    Screen
        JMP     BACK

Readme  LDAA    #11
        STAA    Screen
        JMP     BACK

RFTX    LDAA    #7
        STAA    Screen          ;Screen = 7
        LDD     #SCREEN7
        STD     Line1
        LDD     #SCREEN7+$17
        STD     Line2
        LDD     #SCREEN7+$2D
        STD     Line3
        LDD     #SCREEN7+$9B
        STD     Line4
```

```
        LDAA    #1
        STAA    Line
        JMP     BACK

RFRX    LDAA    #8
        STAA    Screen          ;Screen = 8
        LDD     #SCREEN8
        STD     Line1
        LDD     #SCREEN8+$17
        STD     Line2
        LDD     #SCREEN8+$2D
        STD     Line3
        LDD     #SCREEN8+$9B
        STD     Line4
        LDAA    #1
        STAA    Line
        JMP     BACK


*************************************************************************
F2 2    LDAA    Screen
        CMPA    #3
        BNE     Next 2
        LDAA    #2
        STAA    Screen
        JMP     BACK
Next 2  CMPA    #4
        BNE     Next 3
        JMP     S 2
        JMP     BACK
Next 3  CMPA    #5
        BNE     Next 4
        JMP     S 2
        JMP     BACK
Next 4  CMPA    #12
        BNE     Next 5
        JMP     S 2
        JMP     BACK
Next 5  CMPA    #2
        BNE     Next 6
scr6            LDAA    #6
                STAA    Screen          ;Screen = 6
                LDD     #SCREEN6
                STD     Line1
                LDD     #SCREEN6+$17
                STD     Line2
                LDD     #SCREEN6+$2D
                STD     Line3
                LDD     #SCREEN6+$9B
                STD     Line4
                LDAA    #1
                STAA    Line
        JMP     BACK
Next 6  CMPA    #6
        BNE     Next 7
        LDAA    #2
        STAA    Screen
        JMP     BACK
Next 7  CMPA    #7
        BNE     Next 8
        JMP     scr6
        JMP     BACK
Next 8  CMPA    #8
        BNE     Next_9
```

```
        JMP     scr6
        JMP     BACK
Next_9  CMPA    #9
        BNE     Next_10
        JMP     scr6
        JMP     BACK
Next_10 CMPA    #10
        BNE     Next_11
        JMP     scr6
        JMP     BACK
Next_11 CMPA    #11
        BNE     Next_12
        JMP     scr6
        JMP     BACK
Next_12




BACK    CLR     Key

        RTS

**************************************************************************
*****************************************************
DATAupdate
*************************************************************
        LDAA    Screen
        CMPA    #4
        BEQ     SENSOR1
        CMPA    #5
        BEQ     CAR_1
        JMP     BACK1
CAR_1   JMP     CAR1
SENSOR1 PSHX
        PSHA PSHB
        PSHY
        LDX
        #LCDBAS
*O2 Sensor****************************************
        LDAA    #$44
        ADDA    #$80            ;set the posistion
        STAA    LCDBAS
        BRSET   0,X $80 *               ;delay
                PSHX
                        LDAA    O2      ;SCALING CALCULATIONS
                        LDAB    #100
                        MUL LDX
                        #255 IDIV
                        XGDX
                        STAB
                        TEMP1

                CLRA            ;Binary to BCD
                LDAB    TEMP1
                LDX     #10
                IDIV
```

```
                STAB    TEMP+2
                STAB    O2_3
                XGDX LDX
                #10 IDIV
                STAB
                TEMP+1
                STAB    O2_2
                XGDX LDX
                #10 IDIV
                STAB
                TEMP
                STAB    O2_1
                PULX
        LDAA    TEMP
        ADDA    #$30
        STAA    TEMP
        LDAA    TEMP+1
        ADDA    #$30
        STAA    TEMP+1
        LDAA    TEMP+2
        ADDA    #$30
        STAA    TEMP+2
        LDY     #TEMP
        JSR     LCD_OUT
*CTS Sensor****************************************
        LDAA    #$4D
        ADDA    #$80            ;set the posistion
        STAA    LCDBAS
        BRSET   0,X $80 *               ;delay
                PSHX
                        LDAA    CTS     ;SCALING CALCULATIONS
                        LDAB    #100
                        MUL
                        LDX     #119
                        IDIV
                        XGDX
                        STAB    TEMP1

                CLRA            ;       Binary to BCD
                LDAB    TEMP1
                LDX     #10
                IDIV
                STAB    TEMP+2
                STAB    CTS_3
                XGDX
                LDX     #10
                IDIV
                STAB    TEMP+1
                STAB    CTS_2
                XGDX
                LDX     #10
                IDIV
                STAB    TEMP
                STAB    CTS_1
                PULX
        LDAA    TEMP
        ADDA    #$30
        STAA    TEMP
        LDAA    TEMP+1
        ADDA    #$30
        STAA    TEMP+1
        LDAA    TEMP+2
        ADDA    #$30
        STAA    TEMP+2
```

```
        LDY     #TEMP
        JSR     LCD_OUT
*AIR Sensor*****************************************
        LDAA    #$18
        ADDA    #$80            ;set the posistion
        STAA    LCDBAS
        BRSET   0,X $80 *               ;delay
                PSHX
                        LDAA    AIR     ;SCALING CALCULATIONS
                        LDAB    #100
                        MUL LDX
                        #119 IDIV
                        XGDX
                        STAB
                        TEMP1

                CLRA

                LDAB    TEMP1
                LDX     #10
                IDIV STAB
                TEMP+2
                STAB    AIR_3
                XGDX LDX
                #10 IDIV
                STAB
                TEMP+1
                STAB    AIR_2
                XGDX LDX
                #10 IDIV
                STAB
                TEMP
                STAB    AIR_1
                PULX
        LDAA    TEMP
        ADDA    #$30
        STAA    TEMP
        LDAA    TEMP+1
        ADDA    #$30
        STAA    TEMP+1
        LDAA    TEMP+2
        ADDA    #$30
        STAA    TEMP+2
        LDY     #TEMP
        JSR     LCD_OUT
*MAP Sensor*****************************************
        LDAA    #$21
        ADDA    #$80            ;set the posistion
        STAA    LCDBAS
        BRSET   0,X $80 *               ;delay
                PSHX
                CLRA LDAB

                MAP
                LDX     #10
                IDIV STAB
                TEMP+2
                STAB    MAP_3
                XGDX LDX
                #10 IDIV
                STAB
                TEMP+1
                STAB    MAP_2
                XGDX LDX
                #10
```

```
        IDIV STAB

        TEMP
        STAB    MAP_1
        PULX
LDAA    TEMP
ADDA    #$30
STAA    TEMP
LDAA    TEMP+1
ADDA    #$30
STAA    TEMP+1
LDAA    TEMP+2
ADDA    #$30
STAA    TEMP+2
LDY     #TEMP
JSR     LCD_OUT

LDAA    #$68
ADDA    #$80            ;set the posistion
STAA    LCDBAS
BRSET   0,X $80 *               ;delay

                LDAA    O2_1
                ADDA    #$30
                STAA    O2_1
                LDAA    O2_2
                ADDA    #$30
                STAA    O2_2
                LDAA    O2_3
                ADDA    #$30
                STAA    O2_3
                LDAA    CTS_1
                ADDA    #$30
                STAA    CTS_1
                LDAA    CTS_2
                ADDA    #$30
                STAA    CTS_2
                LDAA    CTS_3
                ADDA    #$30
                STAA    CTS_3
                LDAA    AIR_1
                ADDA    #$30
                STAA    AIR_1
                LDAA    AIR_2
                ADDA    #$30
                STAA    AIR_2
                LDAA    AIR_3
                ADDA    #$30
                STAA    AIR_3
                LDAA    MAP_1
                ADDA    #$30
                STAA    MAP_1
                LDAA    MAP_2
                ADDA    #$30
                STAA    MAP_2
                LDAA    MAP_3
                ADDA    #$30
                STAA    MAP_3

PULY PULB
PULA
PULX
JMP
BACK1
```

```
CAR1    PSHX
        PSHA
        PSHB
        PSHY
        LDX   #LCDBAS
*RPM Sensor*****************************************
        LDAA    #$4D
        ADDA    #$80            ;set the posistion
        STAA    LCDBAS
        BRSET   0,X $80 *               ;delay
                PSHX
                        LDAA    RPM     ;SCALING CALCULATIONS
                        LDAB    #10
                        MUL LDX
                        #159 IDIV
                        XGDX
                        STAB
                        TEMP1

                 CLRA

                LDAB    TEMP1
                LDX     #10
                IDIV STAB
                TEMP+2
                STAB    RPM_3
                XGDX LDX
                #10 IDIV
                STAB
                TEMP+1
                STAB    RPM_2
                XGDX LDX
                #10 IDIV
                STAB
                TEMP
                STAB    RPM_1
                PULX
        LDAA    TEMP
        ADDA    #$30
        STAA    TEMP
        LDAA    TEMP+1
        ADDA    #$30
        STAA    TEMP+1
        LDAA    TEMP+2
        ADDA    #$30
        STAA    TEMP+2

        LDY     #TEMP
        JSR     LCD_OUT
*TPS Sensor*****************************************
        LDAA    #$18
        ADDA    #$80            ;set the posistion
        STAA    LCDBAS
        BRSET   0,X $80 *               ;delay
                PSHX
                        LDAA    TPS     ;SCALING CALCULATIONS
                        LDAB    #100
                        MUL LDX
                        #255 IDIV
                        XGDX
                        STAB
                        TEMP1

                 CLRA

                LDAB    TEMP1
```

```
                LDX     #10
                IDIV STAB
                TEMP+2
                STAB    TPS_3
                XGDX LDX
                #10 IDIV
                STAB
                TEMP+1
                STAB    TPS_2
                XGDX LDX
                #10 IDIV
                STAB
                TEMP
                STAB    TPS_1
                PULX
        LDAA    TEMP
        ADDA    #$30
        STAA    TEMP
        LDAA    TEMP+1
        ADDA    #$30
        STAA    TEMP+1
        LDAA    TEMP+2
        ADDA    #$30
        STAA    TEMP+2
        LDY     #TEMP
        JSR     LCD_OUT
*BATT Sensor****************************************
        LDAA    #$23
        ADDA    #$80            ;set the posistion
        STAA    LCDBAS
        BRSET   0,X $80 *               ;delay
                PSHX
                        LDAA    CTS     ;SCALING CALCULATIONS
                        LDAB    #100
                        MUL LDX
                        #100 IDIV
                        XGDX
                        STAB
                        TEMP1
                CLRA

                LDAB    BATT
                LDX     #10
                IDIV STAB
                TEMP+2
                STAB    BATT_3
                XGDX LDX
                #10 IDIV
                STAB
                TEMP+1
                STAB    BATT_2
                XGDX LDX
                #10 IDIV
                STAB
                TEMP
                STAB    BATT_1
                PULX
        LDAA    TEMP
        ADDA    #$30
        STAA    TEMP
        LDAA    TEMP+1
        ADDA    #$30
        STAA    TEMP+1
        LDAA    TEMP+2
        ADDA    #$30
```

```
        STAA    TEMP+2
        LDY     #TEMP
        JSR     LCD_OUT

        LDAA    #$68
        ADDA    #$80            ;set the posistion
        STAA    LCDBAS
        BRSET   0,X $80 *               ;delay
        PULY
        PULB
        PULA
        PULX

BACK2                   LDAA    RPM_1
                        ADDA    #$30
                        STAA    RPM_1
                        LDAA    RPM_2
                        ADDA    #$30
                        STAA    RPM_2
                        LDAA    RPM_3
                        ADDA    #$30
                        STAA    RPM_3
                        LDAA    TPS_1
                        ADDA    #$30
                        STAA    TPS_1
                        LDAA    TPS_2
                        ADDA    #$30
                        STAA    TPS_2
                        LDAA    TPS_3
                        ADDA    #$30
                        STAA    TPS_3
                        LDAA    BATT_1
                        ADDA    #$30
                        STAA    BATT_1
                        LDAA    BATT_2
                        ADDA    #$30
                        STAA    BATT_2
                        LDAA    BATT_3
                        ADDA    #$30
                        STAA    BATT_3




BACK1   RTS
************************************************************************
*****************************************************
RFVT_INT
************************************************************
* This interuppt allows the user to press a key in Hyper terminal and
output the data to the PC's screen
* Hyper terminal setup need to be 9600 baud, hardware handshaking, and
ASCII characters
        PSHX PSHA
        PSHB
        LDX
        #$1000
        LDAA    RX_data                 ;Load in Data
        BCLR    TFLG1,X %11111101       ;Clear IC2 flag
        STAA  DATRFRX
        CMPA  #1
```

```
        BNE   Next1_1
                LDAA  #7
                STAA  OrderRFRX
                CLR     Range
                BRA   Done

Next1_1 LDAA  OrderRFRX
        CMPA    #7 BEQ
                H2_1
        CMPA    #6 BEQ
                H3_1
        CMPA    #5 BEQ
                H4_1
        CMPA    #4 BEQ
                H5_1
        CMPA    #3 BEQ
                H6_1
        CMPA    #2 BEQ
                H7_1
        CMPA    #1 BEQ
                H8_1
        JMP     I1_1

H2_1    LDAB    DATRFRX                 ;DATA to send
        STAB    CTS
        DEC     OrderRFRX
                BRA     I1_1
H3_1    LDAB    DATRFRX                 ;DATA to send
        STAB    RPM
        DEC     OrderRFRX               ;DATA to send
                BRA     I1_1
H4_1    LDAB    DATRFRX                 ;DATA to send
        STAB    AIR
        DEC     OrderRFRX                       ;DATA to send
                BRA     I1_1
H5_1    LDAB    DATRFRX                 ;DATA to send
        STAB    MAP
        DEC     OrderRFRX                       ;DATA to send
                BRA     I1_1
H6_1    LDAB    DATRFRX                 ;DATA to send
        STAB    O2
        DEC     OrderRFRX                       ;DATA to send
                BRA     I1_1
H7_1    LDAB    DATRFRX                 ;DATA to send
        STAB    TPS
        DEC     OrderRFRX                       ;DATA to send
                BRA     I1_1
H8_1    LDAB    DATRFRX                 ;DATA to send
        STAB    BATT
        DEC     OrderRFRX                       ;DATA to send
                BRA     I1_1

I1_1
Done    PULB
        PULA
        PULX
        RTI
************************************************************************
***************************************************
RS232_INT
************************************************************
        PSHX
        PSHA
        PSHY
```

```
        LDX     #$1000
        LDAA    SCSR,X  ;Need to read this in order to clear flags
        LDAA    SCDR,X  ;Data in

        LDAA    Screen
        STAA    TEMP4

        LDAA    #4
        STAA    Screen
        JSR     DATAupdate

        LDAA    #5
        STAA    Screen
        JSR     DATAupdate

        LDAA    TEMP4
        STAA    Screen

        LDY     #STRING1
loop11  LDAA    0,Y
        CMPA    #$04
        BEQ     DONE11 BRCLR
        SCSR,X TDRE *
        STAA    SCDR,X INY
        BRA     loop11

DONE11  PULY
        PULA
        PULX

        RTI
**************************************************************************
****************************************************
** ACSII Output to Hyper terminal **************************************
        ORG     $5000
STRING1 FCB     $0A,$0D,$0A,$0D,$0A,$0D
        FCB     $20,$FF,$FF,$FF,$FF
        FCC     " WIRELESS "
        FCB     $FF,$FF,$FF, $FF,$20,$20,$0A,$0D
        FCC     "    DATA  LOGGER     "
        FCB     $0A,$0D
        FCC     "                     "
        FCB     $0A,$0D,$0A,$0D
        FCB     $FF,$20
        FCB     $FF,$FF
        FCC     " SENSOR DATA "
        FCB     $FF,$FF,$FF,$20,$0A,$0D
        FCC     " O2="
O2 1    RMB     1
O2 2    RMB     1
O2 3    RMB     1
        FCC     "% CTS="
CTS 1   RMB     1
CTS 2   RMB     1
CTS 3   RMB     1
        FCC     "C    "
        FCB     $0A,$0D
        FCC     "AIR="
AIR 1   RMB     1
AIR 2   RMB     1
AIR 3   RMB     1
        FCC     "C MAP="
MAP_1   RMB     1
```

```
MAP 2   RMB     1
MAP 3   RMB     1
        FCC     "kPaG "
        FCB     $0A,$0D,$0A,$0D
        FCB     $FF,$FF,$FF
        FCC     " CAR DASH DATA "
        FCB     $FF,$FF,$20,$0A,$0D
        FCC     "ENGINE SPEED="
RPM 1   RMB     1
RPM 2   RMB     1
RPM 3   RMB     1
        FCC     "KRPM "
        FCB     $0A,$0D
        FCC     "TPS="
TPS 1   RMB     1
TPS 2   RMB     1
TPS 3   RMB     1
        FCC     "%  BATT="
BATT 1  RMB     1
BATT 2  RMB     1
BATT 3  RMB     1
        FCC     "V "
        FCB     $0A,$0D,$0
```

```
****************************************************************
* Wirless Data Logger                                          *
* Copyright (c) 2002 by: Eric Holland                          *
* ALL RIGHTS RESERVED                                          *
****************************************************************
****************************************************************
* Date: 04/18/2002  Coded by: Eric Holland                     *
* Filename: logger.asm                                         *
*                                                              *
****************************************************************
*          Description                                         *
*This progam is the Data Logger Application Code for the Wireless Data
Logger.    *
*                                                              *
****************************************************************
DDRD    EQU     $09
PORTD   EQU     $08
PORTA   EQU     $00
LCDBAS  EQU     $B5F0           ;LCD port address
LCDDAT  EQU     $B5F1
TCTL2   EQU     $21
TCTL1   EQU     $20
TFLG1   EQU     $23
TMSK1   EQU     $22
*SCI Stuff ****************
BAUD    EQU     $2B  sci baud reg
SCCR1   EQU     $2C  sci control1 reg
SCCR2   EQU     $2D  sci control2 reg
SCSR    EQU     $2E  sci status reg
SCDR    EQU     $2F  sci data reg
TDRE    EQU     $80
RDRF    EQU     $20
* RF Addresses *************
TX_addr EQU     $B5C0
RX_addr EQU     $B5B0
TX_data EQU     $B580
RX data EQU     $B590           ;RF Recieved Data

* START Header *************
Header  EQU     %000000001

        ORG     $1040           ;Starting place for RAM
DATA    RMB     1               ;RF Recieved Data
        FCB     0
DAT232  RMB     1
* Sensor Data **************
CTS     RMB     1
RPM     RMB     1
AIR     RMB     1
MAP     RMB     1
O2      RMB     1
TPS     RMB     1
BATT    RMB     1
Order   RMB     1               ;What is to be transmited
Order232 RMB    1

* For SRAM *****************
*       ORG     $E5
*       JMP     RFVT INT        ;IC2 interrupt Jump Vector
*       ORG     $00C4
*       JMP     RS232_INT       ;SCI interrupt Jump Vector
```

```
* For EEPROM ***************
        ORG     $FFD6
        FDB     RS232_INT

*************************************************************************
*****************************************************
MAIN
*************************************************************
        ORG  $FFFE        set the reset vector
        FDB  $E000
        ORG  $E000

 *      ORG     $2400           ;For SRAM
        LDX     #$1000          ;sets Bias piot
        LDS     #$7FFF          ;Sets Stack Pionter
 *      JSR     LCDSET          ;Setup LCD
        JSR     Setup           ;Sets up inital values


Start   LDAA    Order           ;What is to be Xmitted next
        CMPA    #8
                BEQ     H1
        CMPA    #7
                BEQ     H2
        CMPA    #6
                BEQ     H3
        CMPA    #5
                BEQ     H4
        CMPA    #4
                BEQ     H5
        CMPA    #3
                BEQ     H6
        CMPA    #2
                BEQ     H7
        CMPA    #1
                BEQ     H8


H1      LDAB    #Header                 ;DATA to send
                BRA     I1
H2      LDAB    CTS                     ;DATA to send
                BRA     I1
H3      LDAB    RPM                     ;DATA to send
                BRA     I1
H4      LDAB    AIR                     ;DATA to send
                BRA     I1
H5      LDAB    MAP                     ;DATA to send
                BRA     I1
H6      LDAB    O2                      ;DATA to send
                BRA     I1
H7      LDAB    TPS                     ;DATA to send
                BRA     I1
H8      LDAB    BATT                    ;DATA to send
                BRA     I1


I1      STAB    TX data
        BCLR    PORTD,X %00000100       ;Set RADIOCON LOW
        BSET    PORTD,X %00100000       ;SET RADIOTX_EN High

                PSHX
                LDX     #20                     ;100us LOOP
INNER3          NOP
                NOP
```

```
                DEX
                BNE     INNER3
                PULX

        BCLR    PORTD,X %00100000       ;SET RADIOTX_EN LOW


                PSHX
                LDX     #40000                  ;200ms LOOP
INNER4          NOP                             ;2 cycles
                NOP                             ;2 cycles
                DEX                             ;3 cycles
                BNE     INNER4                  ;3 cycles
                PULX

        BSET    PORTD,X %00000100       ;Set RADIOCON High
        DEC     Order
        BNE     Start
        LDAA    #8
        STAA    Order
        BRA     Start

        SWI
        END
************************************************************************
*****************************************************
Setup
*************************************************************
*       BSET    TCTL2,X %00000100       ;value to set Rising edge
triggered
*       BCLR    TFLG1,X %11111101       ;Clear IC2 flag
*       BSET    TMSK1,X %00000010       ;Enables IC2 interrupt


        LDAA    #$3E            ;value to set D5-D2 for output
        STAA    DDRD,X          ;Sets PORTD for output
        LDAA    #%00000110
        STAA    PORTD,X         ;Startup values
        LDAA    #%00100101
        STAA    PORTA,X         ;Startup values
        LDAA    #%111111111     ;TX_addr start up value
        STAA    TX_addr
        LDAA    #%111111111     ;RX_addr start up value
        STAA    RX_addr

        LDAA    #8
                STAA    Order
        LDAA    #2
                STAA    CTS
        LDAA    #3
                STAA    RPM
        LDAA    #4
                STAA    AIR
        LDAA    #5
                STAA    MAP
        LDAA    #6
                STAA    O2
        LDAA    #7
                STAA    TPS
        LDAA    #8
                STAA    BATT
        LDAA    #0
                STAA    Order232
* SCI Setup ***********************
```


DOI: 10.56816/2378-6949.1179

```
         LDAA #$30
         STAA BAUD,X
         LDAA #$00
         STAA SCCR1,X
         LDAA #%00101100
         STAA SCCR2,X


        CLI                              ;Enables Interrupt

        RTS
************************************************************************
*****************************************************

************************************************************************
*****************************************************
LCD_OUT
***********************************************************
PUTSTRG LDAA    0,Y             ;OUTPUTS STRG ADDRESS IN Y
        BEQ     DONE            ;TO LCD SCREEN
        JSR     LCDWRAP
        INY
        BRA     PUTSTRG
DONE    RTS

************************************************************************
*****************************************************
LCDSET
*CONFIGURES LCD SCREEN
************************************************************
        PSHX
        LDX   #LCDBAS
        LDAA  #$3C              ; set 20x4 Display
        STAA  LCDBAS
        BRSET 0,X $80 *         ;delay
        LDAA  #$01              ; Clear & Home
        STAA  LCDBAS
        BRSET 0,X $80 *         ;delay
        LDAA  #$0F              ; Display on
        STAA  LCDBAS
        BRSET 0,X $80 *         ;delay
        LDAA  #$06              ; Cursor shift on
        STAA  LCDBAS
        LDAA  #$14              ; Shift right
        STAA  LCDBAS
        LDAA  #$02              ; Cursor to Home
        STAA  LCDBAS
        BRSET 0,X $80 *         ;delay
        PULX
        RTS
************************************************************************
*****************************************************
LCDWRAP
************************************************************
        STAA    LCDDAT          ; display character
LCDLP   LDAA    LCDBAS          ; read next character position
        BMI     LCDLP           ; test if busy and wait if true
        CMPA    #$13            ; test for line 1 wrap
        BEQ     LCD1            ; if match, correct line wrap
        CMPA    #$53            ; test for line 2 wrap
        BEQ     LCD2            ; if match, correct line wrap
        CMPA    #$27            ; test for line 2 wrap
        BEQ     LCD3            ; if match, correct line wrap
        RTS
```

```
* correct line 1 wrap from line 3 to line 2
LCD1    LDAA    #$40            ; load line 2 start position
        ORAA    #$80            ; set command bit
        STAA    LCDBAS          ; write to display
        RTS
* correct line 2 wrap from line 4 to line 3
LCD2    LDAA    #$14            ; load line 3 start position
        ORAA    #$80            ; set command bit
        STAA    LCDBAS          ; write to display
        RTS
* correct line 3 wrap from line 2 to line 4
LCD3    LDAA    #$54            ; load line 4 start position
        ORAA    #$80            ; set command bit
        STAA    LCDBAS          ; write to display
        RTS
************************************************************************
*****************************************************
RFVT_INT
************************************************************
        SEI
        PSHX
        LDX     #$1000                  ;sets Bias piot
        LDAA    RX_data                 ;Load in Data
        ADDA    #$30                    ;Convert number to ASCII
        STAA    DATA
        LDY     #DATA
        JSR     LCD_OUT                 ;Displays number on screen
        BCLR    TFLG1,X %11111101       ;Clear IC2 flag
        PULX
        CLI
        RTI
************************************************************************
****************************************************
RS232_INT
*************************************************************
        PSHX
        PSHA PSHB
        LDX
        #$1000

        LDAA  SCSR,X    ;Need to read this in order to clear flags
        LDAA  SCDR,X    ;Data in
        STAA  DAT232
        CMPA  #1
        BNE   Next1
                LDAA  #7
                STAA  Order232
                BRA   Done

Next1   LDAA  Order232
        CMPA    #7
                BEQ     H2 1
        CMPA    #6
                BEQ     H3 1
        CMPA    #5
                BEQ     H4 1
        CMPA    #4
                BEQ     H5 1
        CMPA    #3
                BEQ     H6 1
        CMPA    #2
                BEQ     H7 1
        CMPA    #1
                BEQ     H8_1
```

```
        JMP     I1_1

H2 1    LDAB    DAT232                  ;DATA to send
        STAB    CTS
        DEC     Order232
                BRA     I1_1
H3 1    LDAB    DAT232                  ;DATA to send
        STAB    RPM
        DEC     Order232                ;DATA to send
                BRA     I1_1
H4 1    LDAB    DAT232                  ;DATA to send
        STAB    AIR
        DEC     Order232                        ;DATA to send
                BRA     I1_1
H5_1    LDAB    DAT232                  ;DATA to send
        STAB    MAP
        DEC     Order232;DATA to send BRA
                I1_1
H6_1    LDAB    DAT232                  ;DATA to send
        STAB    O2
        DEC     Order232;DATA to send BRA
                I1_1
H7_1    LDAB    DAT232                  ;DATA to send
        STAB    TPS
        DEC     Order232;DATA to send BRA
                I1_1
H8_1    LDAB    DAT232                  ;DATA to send
        STAB    BATT
        DEC     Order232;DATA to send BRA
                I1_1

I1_1
Done    PULB
        PULA
        PULX
        RTI
```

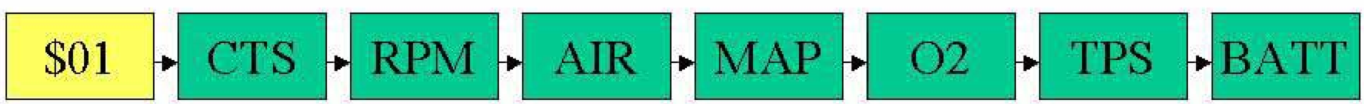
Section 6: Design Computations

The following is the protocol specified for communications between the Data Logger and the Fuel Injection Controller. This is also the flow of information across the Wireless link between the Data Logger and Monitoring Tool.

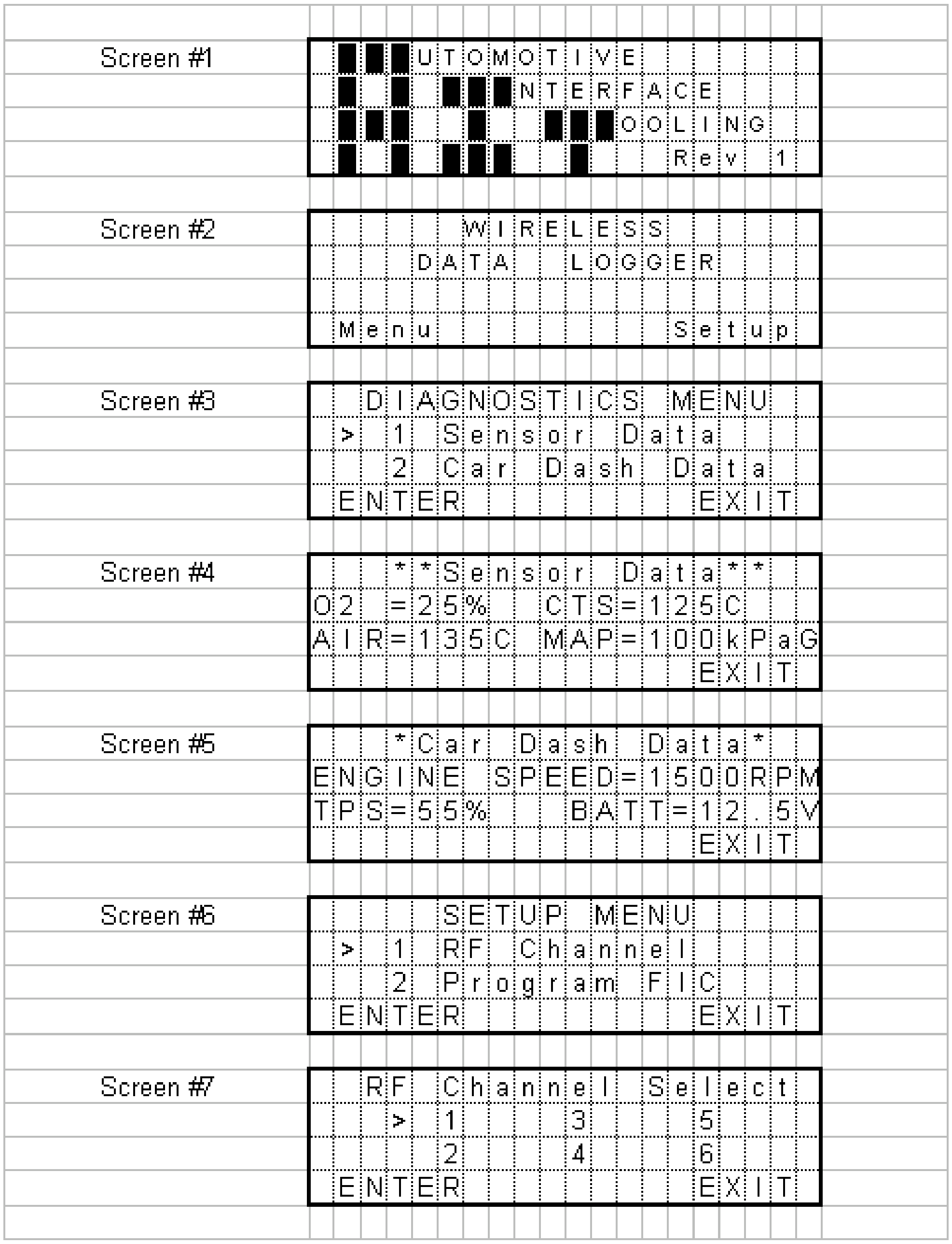
Fuel Injection Controller to Data Logger Flow Chart

Four times a Sec the Fuel Injection controller will send the sensor data to the Data Logger via the RS-232 serial link. The baud rate will be 9600. The serial information will have the following format. No Parity, 8 Data Bits, 1 Stop Bit, and no Handshaking.

The Fuel injection controller will send the data in packets. The first packet will be the unique header $01, so the Data Logger can distinguish the start of the sensor data.



The following are the screen shots of the Monitoring Tool.



The following memory map is for a 68HC11E9 as shipped in this development board. Other 68HC11 devices in the A and E series may also be used with this board. These optional devices differ in the amount of internal RAM, ROM and EEPROM available and the factory default value of the CONFIG register. Consult the technical reference for the specific device you are using for additional information.

FFFF	FFFE	RESET Vector Address
	E000	Memory Socket U7 (8K device) if ROMON disabled or
	D000	68HC711E9 Internal PROM (12K) in U1 if ROMON enabled
CFFF	B800	Program or Data Memory EEPROM or RAM in U6 (not installed)
B7FF	B600	HC11 Internal EEPROM in U1 Program or Data
B5FF	B580	Peripheral Area CS0 - CS7 CS7 = B5F2-B5FF; LCD = B5F0-B5F1; CS6 = B5E0-B5EF CS5 = B5D0-B5DF; CS4 = B5C0-B5CF; CS3 = B5B0-B5BF CS2 = B5A0-B5AF; CS1 = B590-B59F; CS0 = B580-B58F
B57F	8000	Program or Data Memory EEPROM or RAM in U6 (not installed)
7FFF	1040	Data Memory RAM in U5
103F	1000	68HC11 Internal Registers See 68HC11 Technical Data Manual
0FFF	0200	Data Memory RAM in U5
01FF	0000	68HC11 Internal RAM in U1 - (42-FF reserved by Buffalo Monitor)

Section 7: Bill of Materials

The following pages show the interface board BOM, the Data Logger upper level BOM, and the Monitoring Tool upper level BOM.

Wireless Data Logger

11/25/01

Total Cost of Interface Board = $144.94

Ref Des	Part #	Description	Qty	Price per unit	Total
U1	TWS-434	Transmitter Module	0	$12.95	$0.00
U2	LINX Xmitter	Transmitter Module	1	$31.29	$31.29
U3	HT-640	Holtek Encoder	1	$2.95	$2.95
U4,U5,U9	74HCT273	Octal D Latch	3	$1.25	$3.75
U6	RWS-434	Reciever Module	0	$12.95	$0.00
U7	Linx Reciever	Reciever Module	1	$46.90	$46.90
U8, U11	74HCT244	Octal Buffer 3-state	2	$1.25	$2.50
U10	HT-648L	Holtek Decoder	1	$2.95	$2.95
U12, U28	74HCT08	Quad AND GATE	2	$1.25	$2.50
U13	LM311	Single Analog Comparator	1	$0.50	$0.50
U27	74HCT04	6 Inverters	1	$1.25	$1.25
J1	IDC60pin	60pin Male Header	1	$0.75	$0.75
J2	IDC40	40pin Male Header	1	$0.75	$0.75
J3	J1962	OBDII Connector	1	$6.00	$6.00
C1, C2, C3, C4, C5, C6, C7, C8, C9, C10, C11, C12, C13, C27, C28		0.1uF Ceramic Capacitor	15	$0.05	$0.75
C30, C31, C32		10uF 25V Tantulumn Capacitor	3	$0.50	$1.50
D1, D2, D3, D4, D5, D6, D7		ON-Semi MBRA140T3	7	$0.50	$3.50

R1, R2, R3, R4, R5, R6, R7, R8, R9, R10, R11, R12, R13, R14, R15, R16, R19, R26, R27, R32, R33, R47		0 Ohm Resistors	22	$0.05	$1.10
R17		100 Ohm Reisitor	1	$0.05	$0.05
R18, R20, R22, R28, R29, R46		1K Ohm Resistor	6	$0.05	$0.30
R21		3.3K Ohm Resistor	1	$0.05	$0.05
R23, R30		390K 0.1% Resistor	2	$0.05	$0.10
R24, R25, R31, R34, R35, R36, R37, R38, R39, R40, R41, R42, R43, R44, R45		10K Ohm Resistor	15	$0.05	$0.75
F1, F2, F3		1.5 Amp SMT Fuse	3	$1.00	$3.00
Q1, Q4, Q5	2N3904	NPN-BJT	3	$0.25	$0.75
Q2, Q3	2N7002LT1	N-MOSFET	2	$0.50	$1.00
PCB Board		PCB Board	1	$30.00	$30.00

Wireless Data Logger

11/25/01

Part #	Description	Qty	Price per unit	Total
	Interface Board	1	$144.93	$144.93
	50 ohm Antenna	1	$8.00	$8.00
	DB-9 Male Plug	1	$1.00	$1.00
	Metal Case	1	$10.00	$10.00
				$0.00
CME11E9-EVBU	Axiom Evaluation Board	1	$99.00	$99.00

Wireless Data Logger

11/25/01

Part #	Description	Qty	Price per unit	Total
	Interface Board	1	$144.93	$144.93
	50 ohm Antenna	1	$8.00	$8.00
	DB-9 Male Plug	1	$1.00	$1.00
	Plastic Case	1	$6.00	$6.00
CME11E9-EVBU	Axiom Evaluation Board	1	$99.00	$99.00
HC-LCD	20x4 LCD Screen from Axiom	1	$35.00	$35.00
	Keypad Decal	1	$5.00	$5.00
	Power Button	1	$1.00	$1.00
	AC Adapter	1	$10.00	$10.00

Section 8: Schematic & Printed Circuit Board Layout

The following Pages show the Wireless Data Logger interface board and Axiom's CME11E9-EVBU.

Page 1: Connection to EVBU Board, Speaker, and Power supply circuitry

Page 2: RF Receiver Modules

Page 3: RF Transmitter Modules

Page 4: Keypad Circuitry

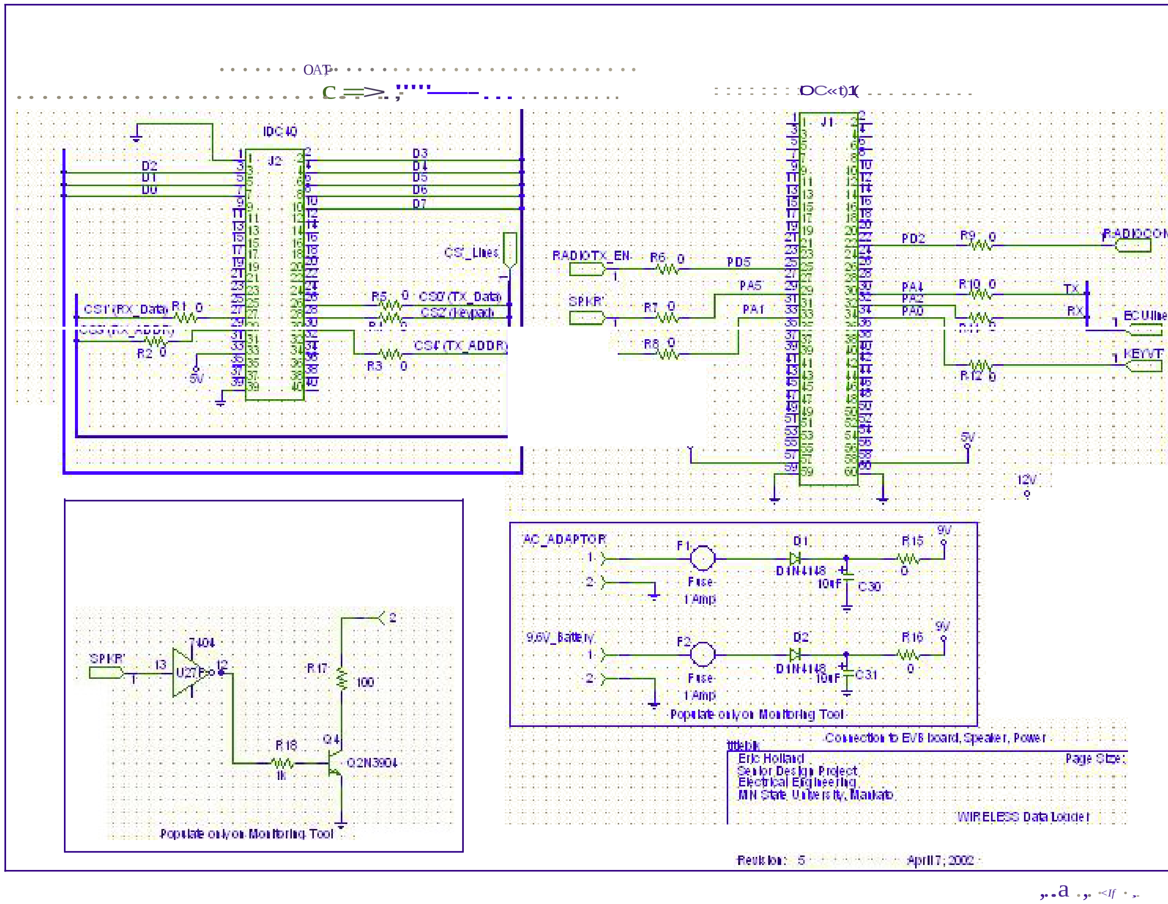
Page 5: Automotive Buffer Circuitry

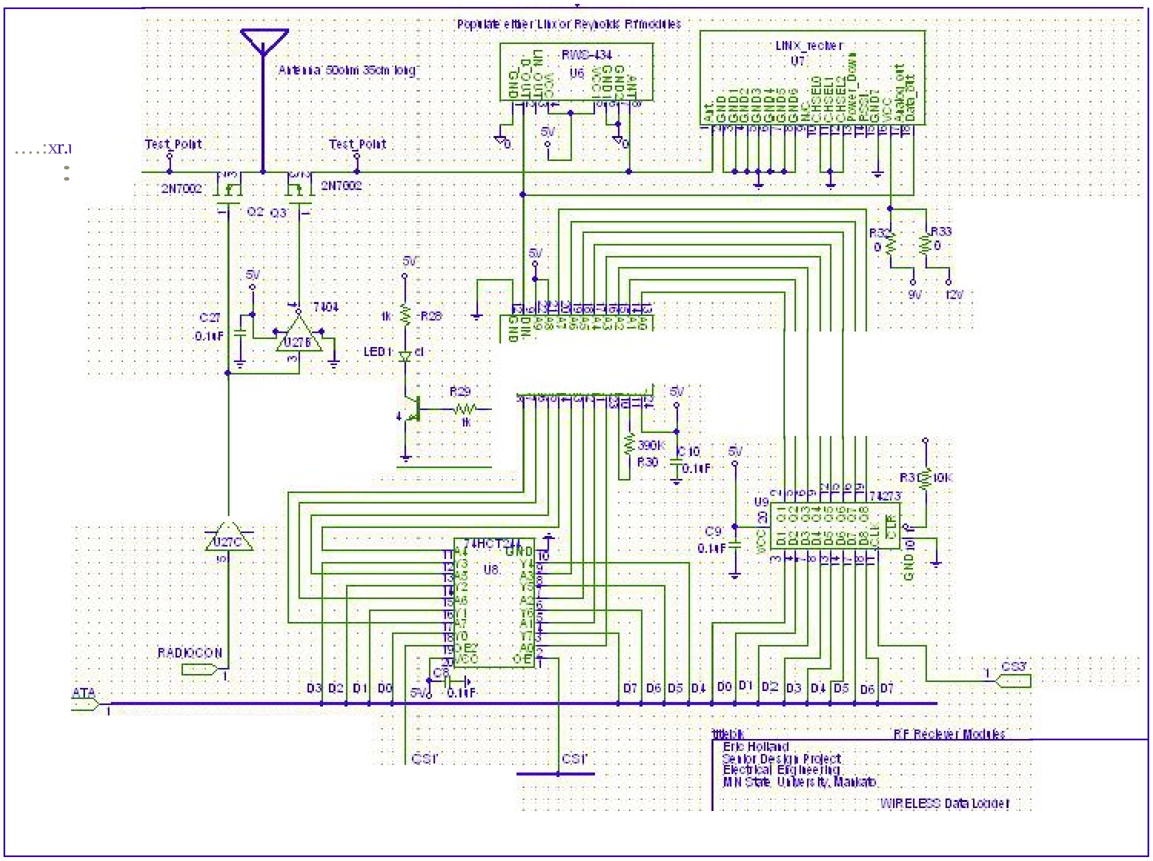
Page 6: Axiom Manufacturing's CME11E9-EVBU

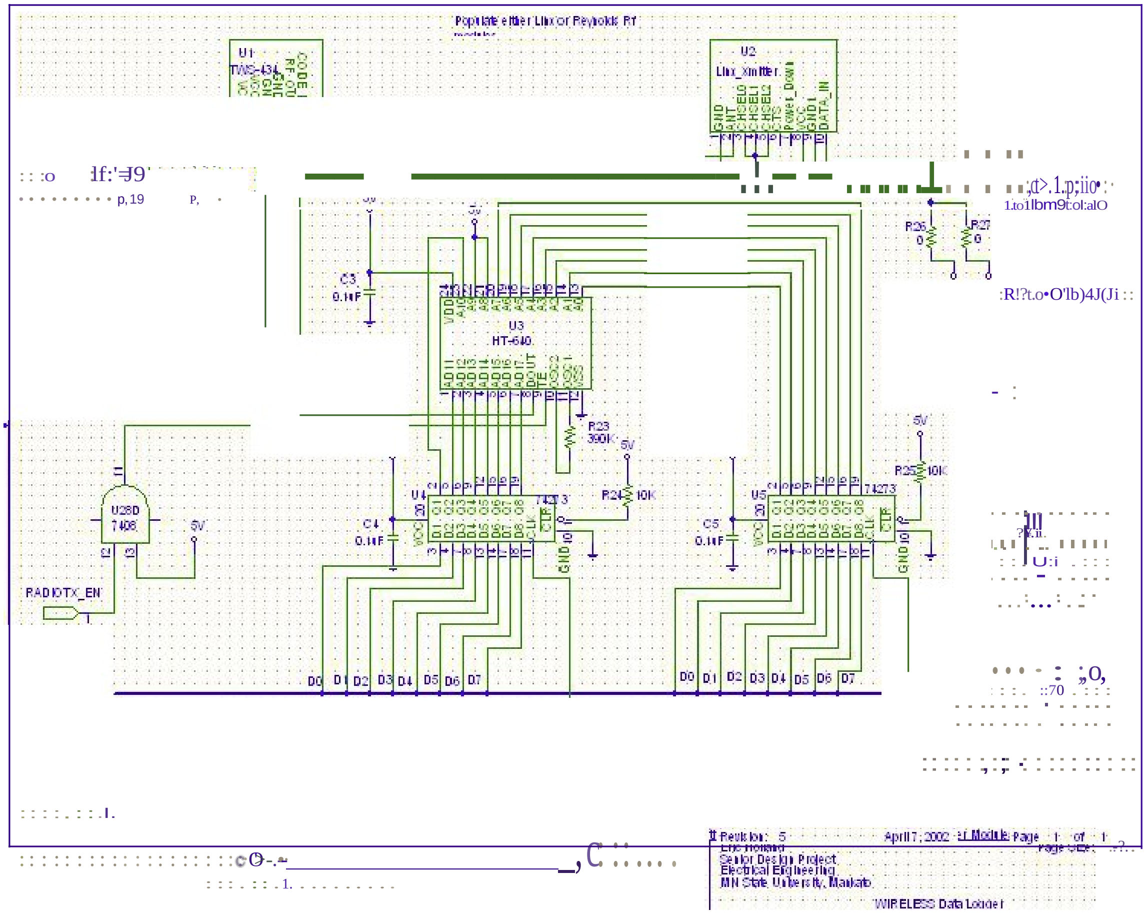
Page 7: Silk Screen and Sodermask

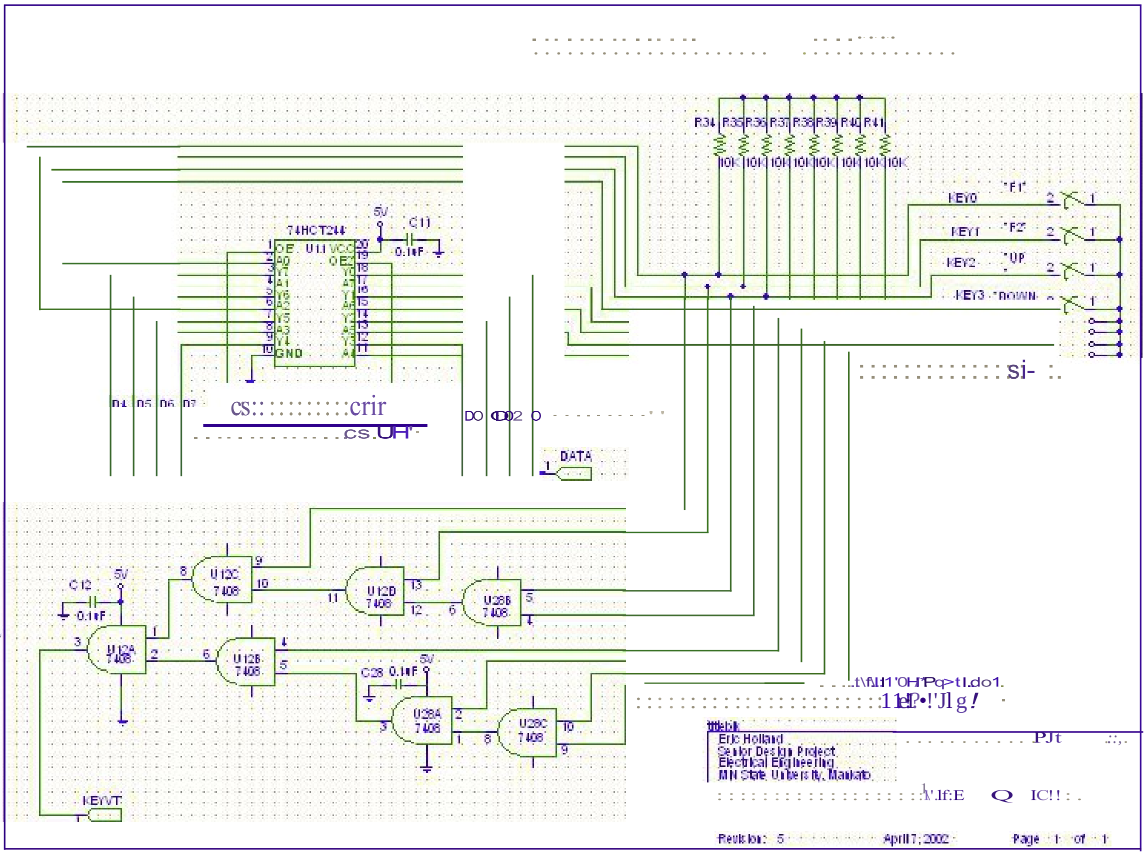
Page 8: Upper Trace Layer

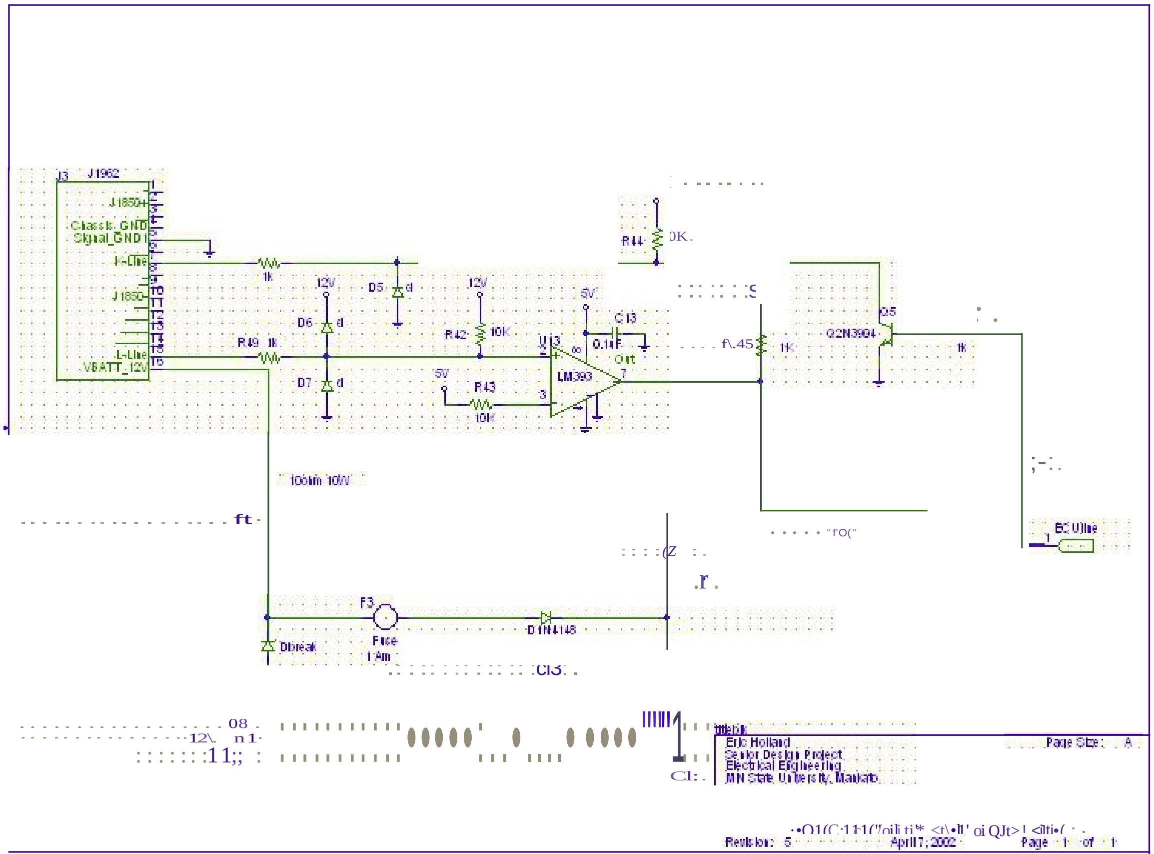
Page 9: Lower Trace Layer

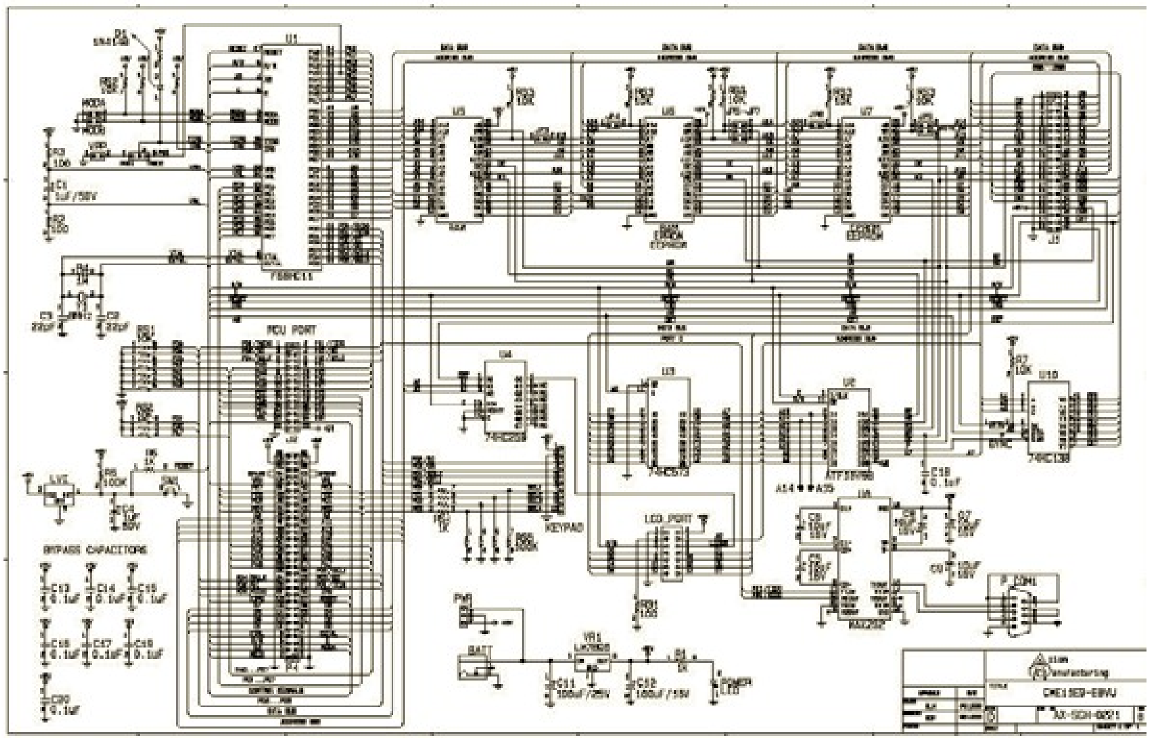

IDC40
J2
J1
CS_Lines
RADIO/TX_EN
SPKR
RADIOCON
ECUline
KEYVT
AC_ADAPTOR
9.6V_Battery
Fuse
1 Amp
D1N4148
Populate only on Monitoring Tool
Q2N3904
Connection to EVB board, Speaker, Power
Eric Holland
Senior Design Project
Electrical Engineering
MN State University, Mankato
WIRELESS Data Logger
Revision: 5
April 7, 2002

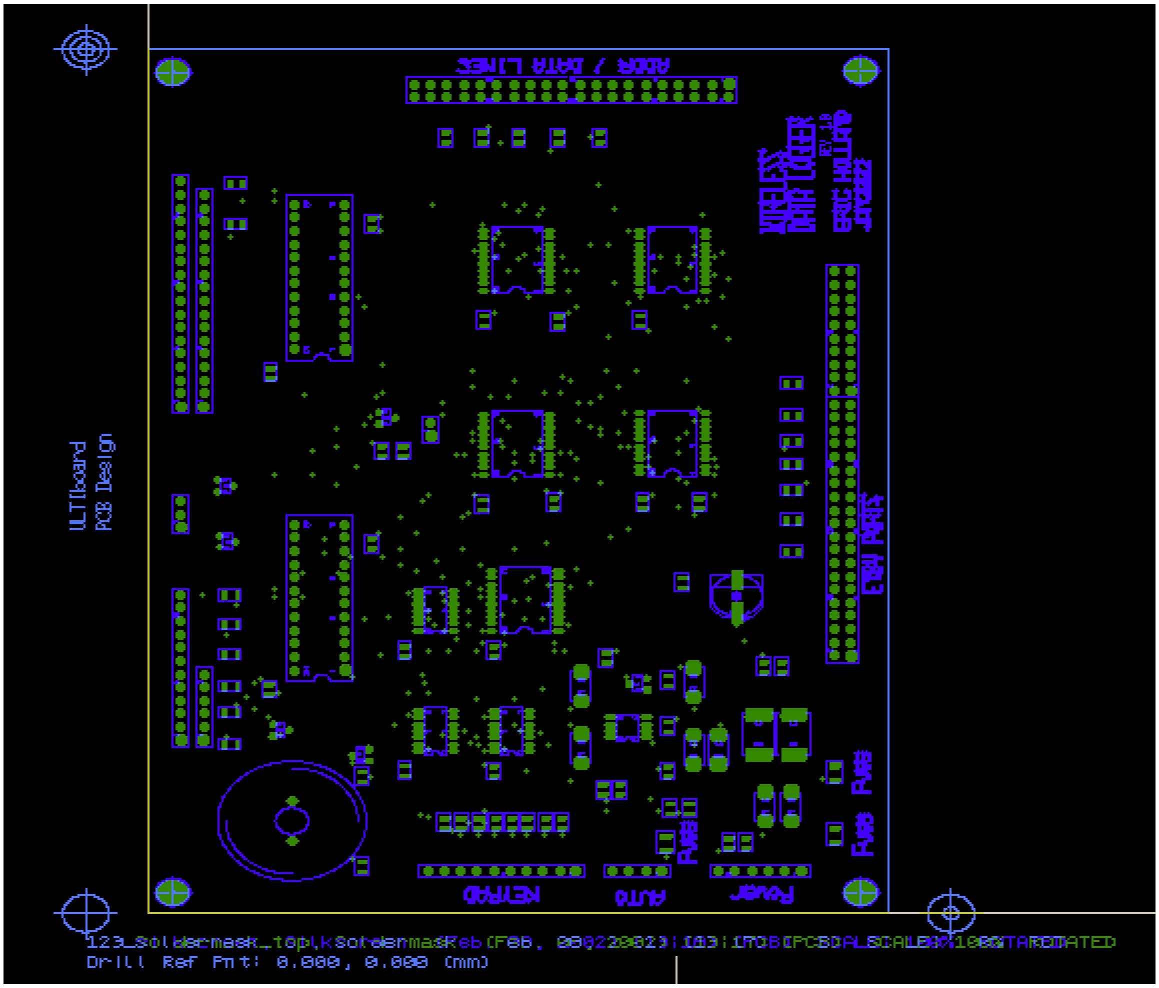

Populate either Linx or Reynolds Rf modules
Antenna 50ohm 35cm long
Test_Point
Test_Point
2N7002
2N7002
Q2
Q3
RWS-434
U6
LINX_recier
U7
R32
R33
9V
12V
5V
C27
0.1uF
7404
U27B
1k
R28
LED1
R29
1k
390K
R30
C10
0.1uF
C9
0.1uF
74273
U9
R31
10K
U27C
74HCT244
U8
RADIOCON
C8
0.1uF
ATA
CS3
CS1
Eric Holland
Senior Design Project
Electrical Engineering
MN State University, Mankato
RF Reciever Modules
WIRELESS Data Logger

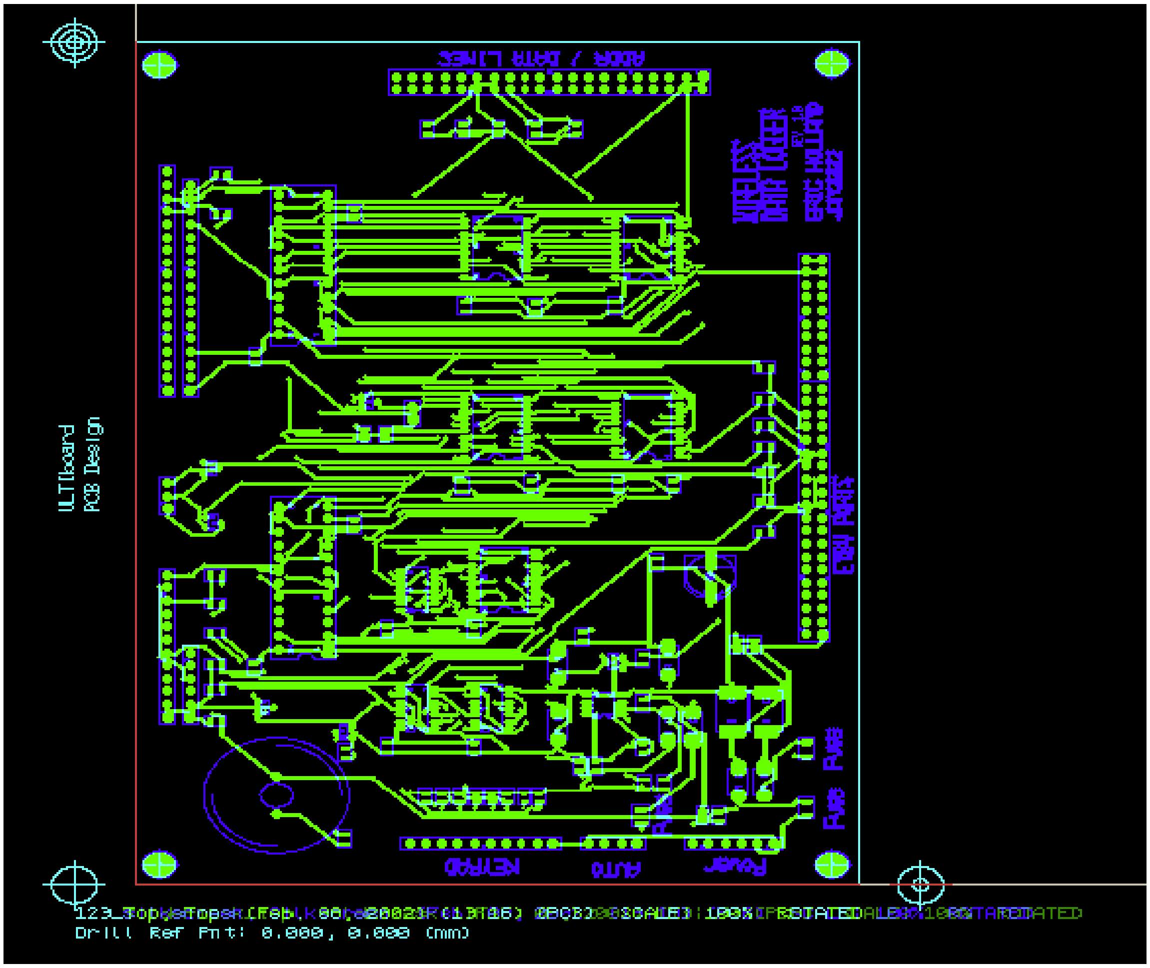

Populate either Linx or Reynolds Rf
U1
TWS-434
U2
Linx_xmitter
U3
HT-640
C3
0.1uF
R23
390K
R24
10K
U4
74273
U5
74273
C4
0.1uF
C5
0.1uF
R25
10K
R26
0
R27
0
U28D
7408
RADIOTX_EN
D0 D1 D2 D3 D4 D5 D6 D7
Revision: 5
April 7, 2002
Eric Holland
Senior Design Project
Electrical Engineering
MN State University, Mankato
WIRELESS Data Logger

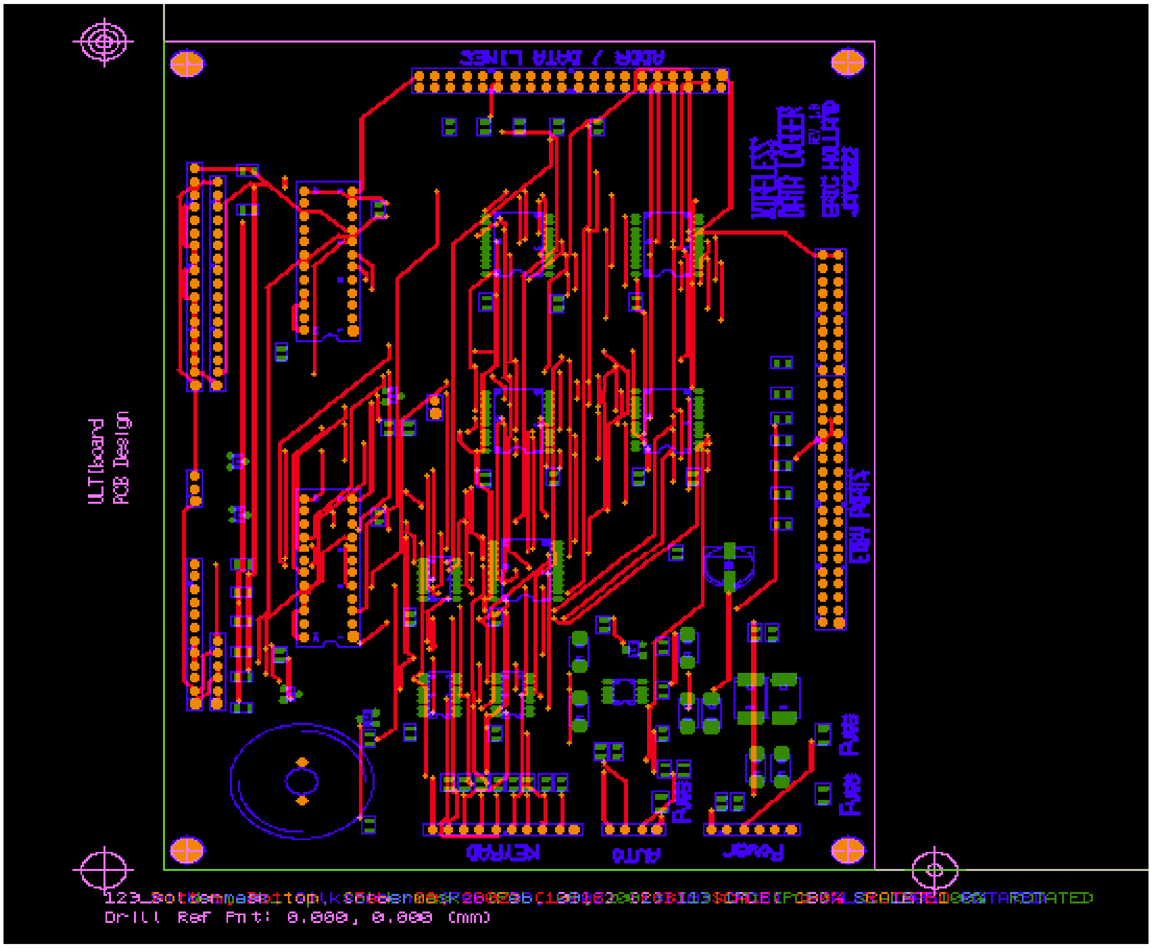

74HCT244
U11
5V
C11
0.1uF
R34 R35 R36 R37 R38 R39 R40 R41
10K
KEY0
KEY1
KEY2
KEY3
DATA
C12
U12A
U12B
U12C
U12D
U28A
U28B
U28C
7408
C28
KEYVT
Eric Holland
Senior Design Project
Electrical Engineering
MN State University, Mankato
April 7, 2002


J3 J1962
J1850+
Chassis_GND
Signal_GND
K-Line
J1850-
L-Line
VBATT_12V
R49 1k
1k
12V
D5
D6
D7
R42 10K
R43
10K
5V
U13
LM393
C13
0.1uF
Out
R44
Q5
Q2N3904
1k
ECU line
100hm 10W
F3
Fuse
1Am
D1N4148
Dbreak
Eric Holland
Senior Design Project
Electrical Engineering
MN State University, Mankato
Page Size: A
Revision: 5
April 7, 2002
Page 1 of 1


U1
MCU PORT
LCD PORT
KEYPAD
BYPASS CAPACITORS
74HC573
74HC138
A14
A15
BATT
PWR
POWER LED




ULTIboard
PCB Design
Drill Ref Pnt: 0.000, 0.000 (mm)


ULTIboard
PCB Design
Drill Ref Pnt: 0.000, 0.000 (mm)

Section 9: Engineering Change Orders

Engineering Change Order #1

Project: Wireless Data Logger

Project Leader: Eric Holland

Date Effective: **December 9, 2001**

Reason For Change:
The ISO 9141-2 protocol is very inefficient and has too much overhead information transmitted. So a new protocol was thought up and will be used. The Specifications for this protocol will be defined in the Design Report.

Old Specification Reads:

1.2 The data logger will request sensor data from the fuel injection controller via a serial link 4 times a second following the ISO 9141-2 protocol.

1.3 The logger will receive the sensor data from the controller following ISO 9141-2 protocol.

New Specification Reads:

1.2 The data logger will request sensor data from the fuel injection controller via a serial link 4 times a second following a protocol specified in the Design Documentation.

1.3 The logger will receive the sensor data from the controller following a protocol specified in the Design Documentation.

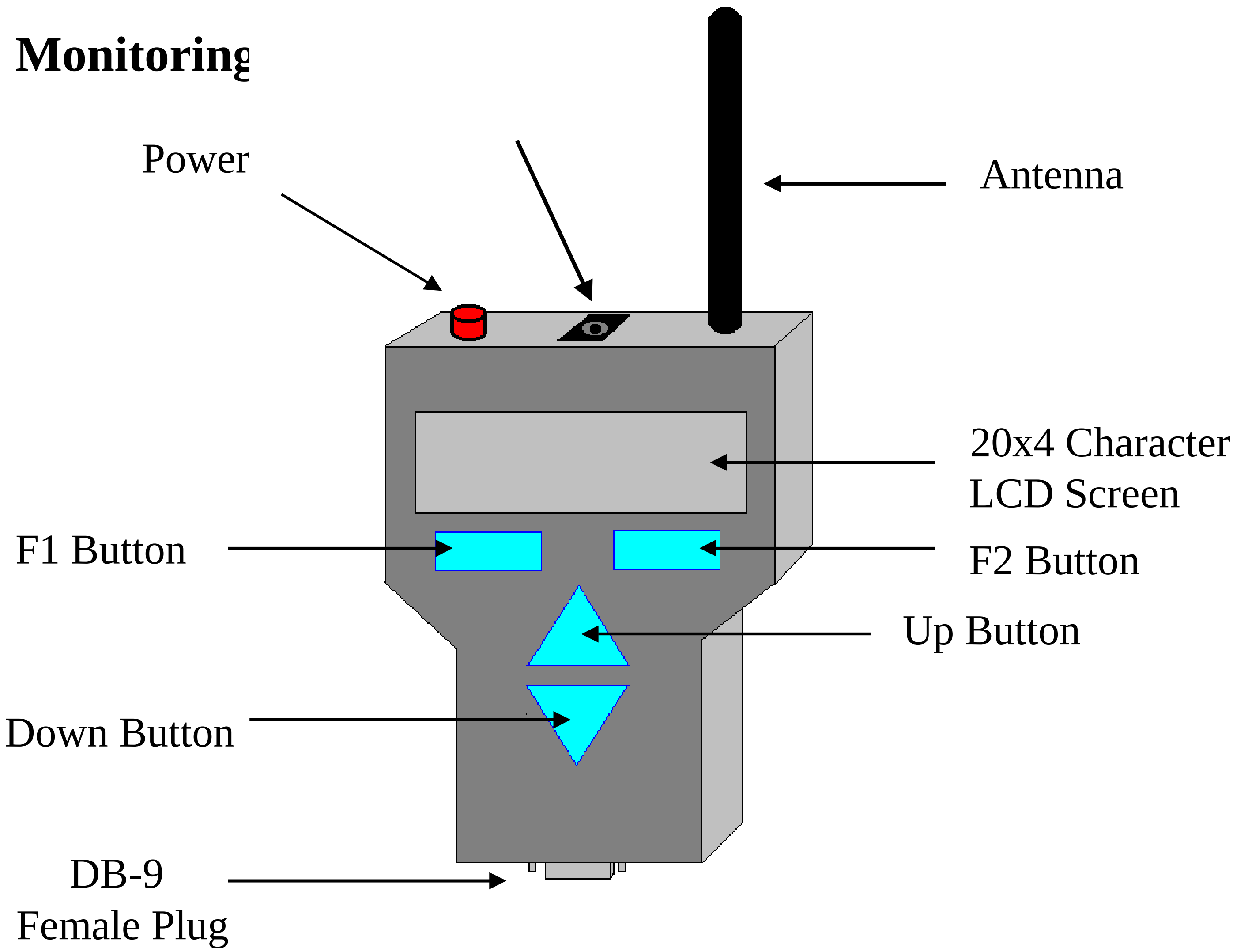
New Revision Number: Rev 1.2

Approvals: **Instructor**____________________________________

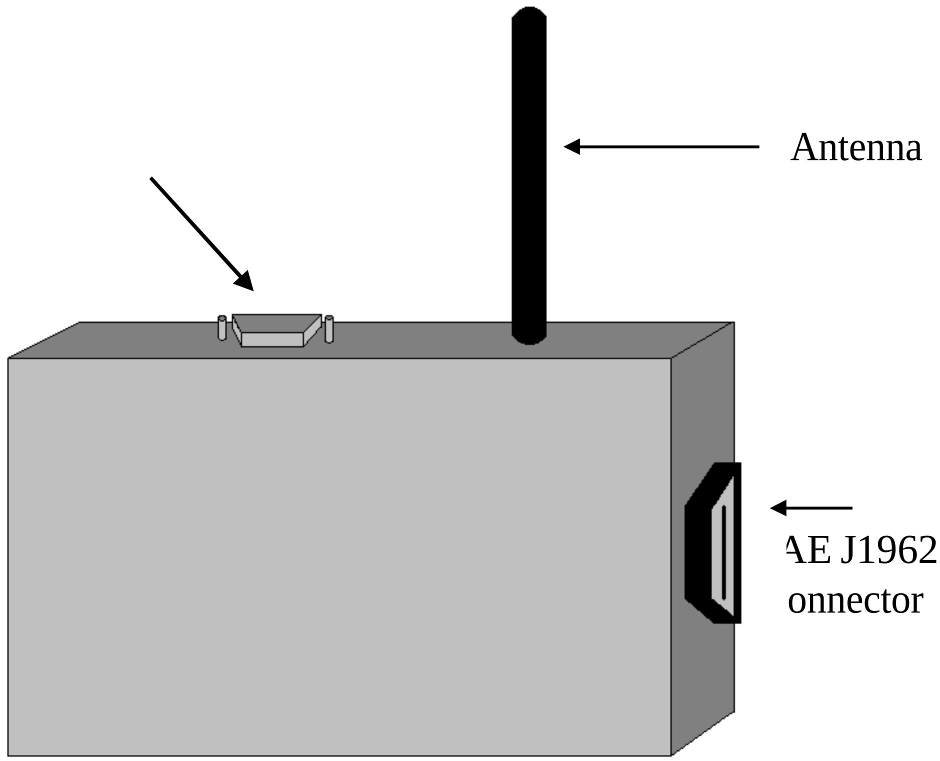
Project Leader________________________________

https://cornerstone.lib.mnsu.edu/jur/vol2/iss1/1
DOI: 10.56816/2378-6949.1179

Section 10: Product Drawings



Data L



Section 11: Product Pictures

I. Picture #1 Monitoring Tool and Data Logger

II. Picture #2 Data Logger

III. Picture #3 Monitoring Tool

Picture #5 Monitoring Tool (Inside)