EEL 4924 Electrical Engineering Design
(Senior Design)

Preliminary Design Report

A Wireless ECU Monitoring System

Team WEMS

27 January 2009

Abstract *The Wireless ECU Monitoring System (WEMS) will provide a wireless interface to a car's electronic control unit (ECU) to monitor vital engine parameters and performance statistics. WEMS will automatically log ECU parameters for later retrieval on a home PC through a wireless connection. WEMS will log ECU parameter values such as speed, and derive values such as distance traveled, average acceleration and deceleration in order to provide the user with a clear view of driving habits and automobile performance. The WEMS system will use the logged parameters to automatically notify the user when and why maintenance is required.*

Team Members:

Name: Kirchgessner, Robert
Email: rkirchge@gmail.com
Phone: 407-267-8043

Name: Houghton, Nathaniel
Email: natex@ufl.edu
Phone: 352-216-3394

Table of Contents

Introduction....4
Project Features and Objectives....5
  Hardware Features 5
    Logging and tracking of ECU parameters....5
    Logging and tracking of derived parameters....5
    Real Time Monitoring of ECU parameters over wireless interface....6
    Easy ECU interfacing via. ODB-II connector....6
    Retrieval of Error codes....6
    Clearing of Error Codes....6
  Software Features (PC)....6
    Automatic User Notification....7
    Error Code Clearing....7
    Real-time polling of ECU Parameters....7
    Downloading / Clearing of Logged Parameters....8
Concept/Technology....8
Challenges....10
  Powering the WEMS Device....11
  Interfacing with the ECU 11
Cost Objectives....11
Division of Labor and Timeline....12

Index of Figures

Drawing 1: Automatic User Notification Concept....7
Drawing 2: WEMS Unit Interfaced with Car ECU....9
Drawing 3: WEMS Unit Attached to PC....10
Drawing 4: WEMS Gantt Chart....15

Index of Tables

Table 1: Tentative Component List....12

Introduction

Modern automobiles use a computer known as an electronic control unit (ECU) to monitor and control engine parameters in order to optimize engine function and notify the user (via the check engine light) when a possible malfunction has occurred. Although automobile maintenance facilities have devices to interface with the ECU, consumers rarely have access to or have any idea that the ECU exists. The Wireless ECU Monitoring System will provide the user with access to ECU parameters and engine status via a personal home computer. This would allow a user to be notified as a car's performance degrades, so the owner can take action before a major problem occurs. A WEMS module would allow a user to monitor how their car is performing, and keep track of when it is time to bring their car in for maintenance.

Products exist that interface with the ECU through the OBD-II interface. These products, however, are most often connected to a car only during maintenance and tune-ups. The ECU does not generally keep track of distance traveled, and does not provide access to a car's odometer reading. Although the distance traveled could be derived from an automobiles speed over time, an automobile is generally not driven while the ECU device is attached. By having the WEMS continuously interfaced with the ECU, we will be able to monitor parameters over time and derive values such as distance traveled, average acceleration and average deceleration and store them for later retrieval. This will provide the user with a better view of how their car is performing over time, and even give the user access to data about driving habits.

One product that offers a wireless Bluetooth connection exists, and is called the “ElmScan 5 Bluetooth” (available here: http://www.scantool.net/elmscan-5-bluetooth.html). One major difference between this device and the WEMS project is that the WEMS will be able to log data for retrieval at a later time. The ElmScan 5 Bluetooth adapter simply allows a real time display of statistics from the ECU.

Project Features and Objectives

The WEMS system will be composed of three major components: a hardware monitoring system that will be interfaced with an automobiles ECU, a USB device that will be connected to a home PC, and a software suite that will automatically download data from the WEMS system when the device is in range and provide the user with important information.

Hardware Features

The hardware components of the WEMS system will have the following features:

- Logging and tracking of ECU parameters.
- Logging and tracking of derived parameters (such as distance traveled.)
- Real Time Monitoring of ECU parameters over wireless interface (XBee.)
- Easy ECU interfacing via. ODB-II connector.
    - Support of ISO 9141-2 and ISO 15765 CAN ODB-II standards.
- Retrieval of Error codes.
- Clearing of Error Codes (turning off “Check Engine” light.)

Logging and tracking of ECU parameters.

The WEMS hardware platform will automatically begin logging ECU parameters when an automobiles ECU is detected. The WEMS system will store the most recent value of each ECU parameter in non-volatile memory, as well as record variations in each parameter over time.

Logging and tracking of derived parameters.

Since each parameter is being tracked over time, it is possible to calculate time-derived parameters such as distance traveled (from the speed of the automobile over time.) These parameters are not typically provided by the ECU.

Real Time Monitoring of ECU parameters over wireless interface.

Since each parameter is being logged automatically by the ECU, access to all the recorded parameter values will be provided at any time over the wireless XBee interface. A user will be able to utilize the USB WEMS wireless interface device at any time to download parameter values.

Easy ECU interfacing via. ODB-II connector.

The WEMS unit will connect easily to modern automobile's ODB-II connector, and operate without user intervention once connected. The WEMS device will support two major ODB-II standards: ISO 9141-2 and ISO 15765 CAN. These standards were chosen because they are the most widely used standards today.

Retrieval of Error codes.

The WEMS unit will automatically detect error codes when the ECU reports them, and display on the debug LCD the reason for the error.

Clearing of Error Codes

The WEMS unit will provide a push-button to automatically reset any ECU error codes once they occur. This will provide the consumer with a method of checking if the problem has been resolved if maintenance had to be performed “in the field.”

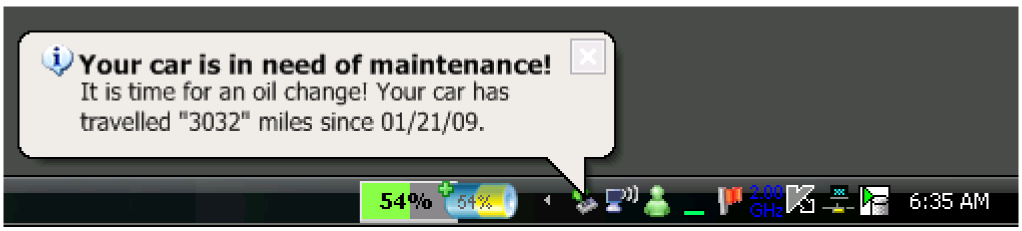
Software Features (PC)

The computer software component of the WEMS system will have several major features, including:

- Automatic User Notification (notification of error codes and / or events.)
- Clearing of Error Codes.
- Real time polling of ECU statistics.
- Downloading / Clearing of logged statistics.

Automatic User Notification

When a vehicle with WEMS comes into range of a PC with a WEMS interface, error codes and certain logged statistics (as determined by configuration settings) will be automatically downloaded through the vehicle's WEMS module. At this point, if any of the logged statistics (such as maximum vehicle speed, coolant temperature) are above or below a configured threshold, or if there are any ECU error codes, the user will be notified through a graphical user interface (a dialog box, message box, or similar notification). Logged statistics will also be stored locally on the PC to allow user examination. This feature is "Automatic User Notification". This feature can be turned off if the user would rather manually retrieve data from his vehicle (see Downloading / Clearing Logged Statistics). Drawing 1 shows the concept of an automatic user notification.



Drawing 1: Automatic User Notification Concept

Error Code Clearing

After error codes have been received and the vehicle has been serviced appropriately, the "Check Engine" light may not automatically turn off. In this case, the PC software can be used to signal the WEMS module to clear the vehicle's ECU error codes and turn off that annoying and troubling "Check Engine" light. If the vehicle does not have its key in the ignition and the ECU is off, it will not be possible to communicate with the vehicle's ECU and clear the error codes. In this case, the WEMS module will be instructed to clear the error codes the next time the vehicle is started.

Real-time polling of ECU Parameters

Although one of the main advantages of using WEMS is the ability to have data logged throughout vehicle operation, real-time polling of ECU statistics will also be possible. Our software will at least provide a real-time tabular view of the vehicle's ECU statistics. If time permits, a graphical view may also be developed.

Downloading / Clearing of Logged Parameters

The statistics logged during vehicle operation can be downloaded through the PC software any time a vehicle with a WEMS is in wireless range. After the statistics have been downloaded the software will also provide a way for the user to erase the statistics from the WEMS module. The amount of flash memory available on the WEMS module will not be expansive so periodic clearing of logged statistics will be necessary.

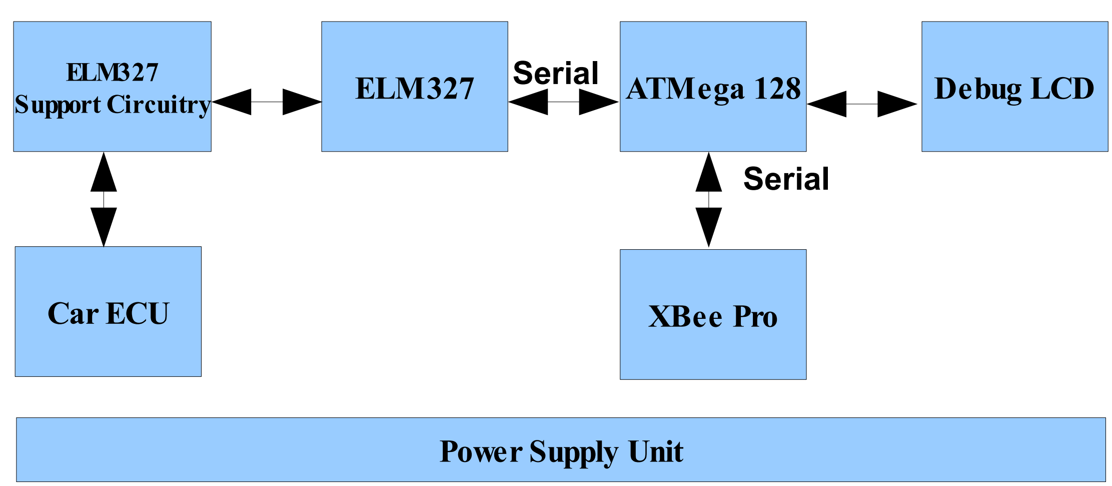
Concept/Technology

The components selected for our project were chosen because we believe they will allow us to implement the features described in this document in the amount of time we have available. The ATmega128 chip was chosen over other microprocessors for several reasons. One of the most important of these is our familiarity with these chips. Also, the ATmega128 has dual UARTs, which is important for linking between the ELM327 and XBee. The ability to program in the C language is also an advantage for Team WEMS.

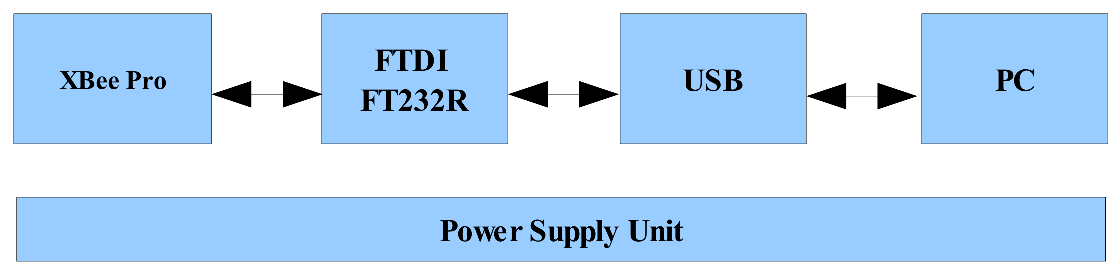
For linking to the car's ODB-II interface, we had several options to choose from. One possibility is to construct the entire interface circuit from discrete components. This was quickly eliminated as a possibility, especially as we researched and learned that would not be implementing one interface, but rather six [2]. Also, since our device will be interfacing to expensive equipment (our cars!) we decided it would be better to go with a tested solution that would allow us to focus on the features of WEMS that we find interesting and useful. As a result, we have picked the ELM327 chip, which supports all six ODB-II protocols. It requires some discrete support circuitry, but not nearly as much as would be required to implement its features on our own. The ELM327 provides a serial link for communication with a computer or microprocessor.

For our wireless interface there were several solutions we have looked at. These include the XBee, XBee Pro, and the Nordic nRF24 series. The XBee and XBee Pro were the two devices most recommended in Junior and Senior Design for their ease of use and, in the case of the XBee Pro, long communication range. The XBee Pro has a line of sight range of up to one mile, according to its documentation. Since the user of WEMS will most likely have their car located some distance away from their home PC, and the signal may have to travel through several walls, the XBee Pro appears to be the best choice.

For our UART to USB interface we have selected the FTDI FT232R. This chip was selected because of its ease of use, our familiarity with it, and the excellent driver support. There are also several simple breakout boards available in lab that will allow us to do some preliminary testing. If we had chosen to go with a chip such as the Silicon Labs CP2102 we would have to first develop and debug a breakout board. Also, after reading some comparisons of the CP2102 vs the FT232R, several people have reported that the Linux support for the CP2102 is somewhat lacking. Although our initial software will most likely not be for Linux, having that path open for future development is a valuable reason to select the FTDI FT232R.



Drawing 2: *WEMS Unit Interfaced with Car ECU*



Drawing 3: *WEMS Unit Attached to PC*

Materials and Resources:

- Atmel ATMega128 microprocessor.
- OBD-II cables and interface chip (ELM327, http://www.elmelectronics.com/connect.html#ELM327).
- Wireless transmitter / receiver (XBee Pro)
- FTDI FT232R (http://www.ftdichip.com/Products/FT232R.htm)
- Standby Power IC.
- Miscellaneous components (headers, resistors, capacitors, diodes, voltage regulators, etc).
- Computer with tools for developing for ATMega microprocessor.
- Oscilloscope/LSA for debugging.
- Bench top power-supply for testing. Vehicle ECU compatible with OBD-II interface (possibly an ECU from a car in the junkyard that could be powered up in the lab).

Challenges

In development of the WEMS system, it is expected that we will experience two major technical challenges: First, since the device will be running from a 12V car battery, there will be challenges designing the power circuitry in order to prevent damage or data loss due to current and voltage spikes as well as maintain Wi-Fi communication while the ignition is off. Secondly, since so many ODB-II standards exist, creating an interface that will operate properly when connected to either of the aforementioned supported standards when the WEMS system is connected. The software (for both the microprocessor and PC) will also be challenging due to the number of features we wish to include.

Powering the WEMS Device

The primary source of power to a car's electrical system is a 12V DC battery that is used to provide power to an engine's starter, air conditioning, stereo, computer systems, window defoggers, headlights, brake lights etc. When the current demand from components is too heavy, or the battery voltage begins to drop, the alternator (which acts as a generator) switches on to assist with current demand and recharge the 12V battery [1]. Due to all the high-power systems connected to a car's battery, there is a great deal of noise and power fluctuations in a cars 12V supply. Since the WEMS device will be driven directly by the 12V power lines connected via the ODB-II connector, it will be necessary to have the appropriate power circuitry to protect our device from current and voltage spikes spikes as well as current and voltage drops during normal car operation. The WEMS system will also need to operate in a low-power mode while the car is not on to minimize current draw while not in use to avoid draining the battery.

Interfacing with the ECU

Interfacing to the ECU via the ODB-II interface will be difficult due to the large number of different standards utilized in the automobile industry. It will be necessary for our device to have the proper supporting circuitry for each standard, and then our device will have to automatically select which standard to use once connected to the ECU.

Cost Objectives

A Tentative Component List can be seen in Table 1. These components have a total cost of around $160. This cost will provide a user of the WEMS with the WEMS module for installation into their vehicle, the wireless communication interface for their PC, and software. For the number of features provided we believe $150 is an excellent price.

The most expensive components in our design are the XBee Pro wireless modules, followed by the ELM327 ODB-II interface chip.

ELM327	$26
J1962M Cable	$16
2 x XBee Pro	$32 x 2
ATmega128	$15.00
1 x UART to USB (FTDI)	$5
1 x LCD	$8
Miscellaneous components	Estimated $10-$15
Estimated total cost	$144-149

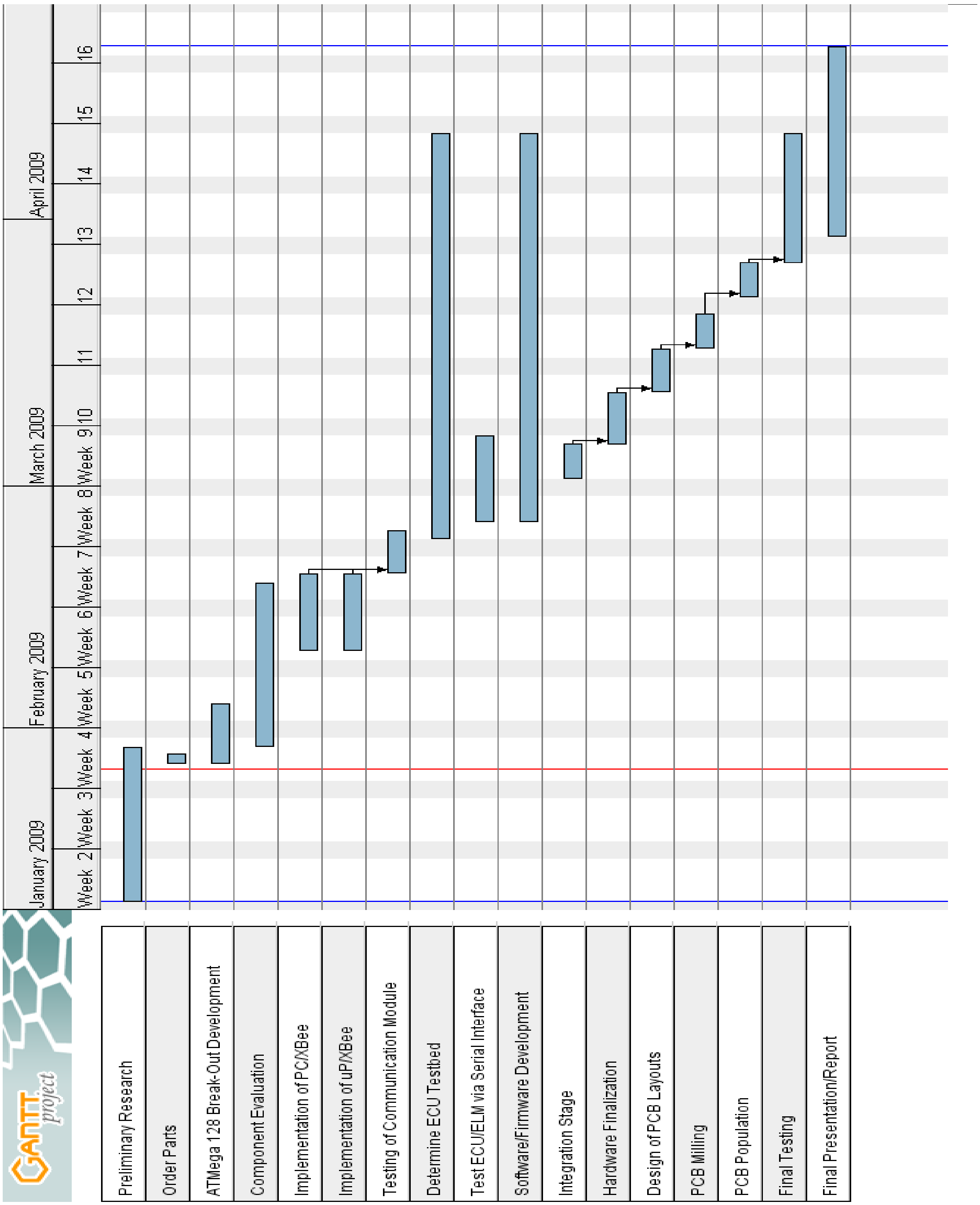
Table 1: Tentative Component List

To compare the cost of WEMS to other ODB-II devices, the ElmScan 5 Bluetooth product costs $200. Another product offered by the same company (that provides no wireless connection) costs $140.

Division of Labor and Timeline

Task	Assigned	Description
Preliminary Research	Kirchgessner, Robert Houghton, Nathaniel	Finding information on components, required circuitry, circuit layout etc.
Order Parts	Kirchgessner, Robert Houghton, Nathaniel	Order the components needed to begin testing and development.
ATmega128 Break-out PCB	Kirchgessner, Robert Houghton, Nathaniel	Design/Mill breakout board for ATmega128.
Component Evaluation	Kirchgessner, Robert Houghton, Nathaniel	Testing individual components. Learning how to interface with each component and ensure it will work as planned.
Implementation of PC<->XBee Interface	Houghton, Nathaniel	This interface includes the USB to serial to XBee Pro.
Implementation of µP<->XBee Interface	Kirchgessner, Robert	This interface utilizes the XBee Pro to a Microprocessor's Serial Communications Port.
Testing of Communication	Kirchgessner, Robert Houghton, Nathaniel	Ensure that the communications interface PC<-> µP works as intended.

Determine ECU Testbed	Kirchgessner, Robert Houghton, Nathaniel	Find an ECU source (junkyard ECU, personal vehicle ECU)
Testing ECU<->ELM<->Serial interface	Kirchgessner, Robert Houghton, Nathaniel	Make sure that the ELM chip and supporting circuitry works properly.
Integration Stage	Kirchgessner, Robert Houghton, Nathaniel	This stage involves connecting all components of the WEMS system together (on a bread board).
Software/Firmware Development	Kirchgessner, Robert Houghton, Nathaniel	Develop PC software and µP firmware.
Hardware Finalization (Testing/Debugging)	Kirchgessner, Robert Houghton, Nathaniel	Once all the components are integrated on a bread board, it will be necessary to ensure that the device functions properly.
Design Final PCB Layout	Kirchgessner, Robert	Design the final PCB(s) (WEMS and USB interface.)
PCB Milling	Kirchgessner, Robert Houghton, Nathaniel	Fabricate and solder components to the PCB(s).
PCB Population	Kirchgessner, Robert Houghton, Nathaniel	Populate the PCB with components.
Final Testing	Kirchgessner, Robert Houghton, Nathaniel	Testing the device in a car to ensure that the device functions properly. During this stage we will also find out the device limitations, such as wireless range.
Final Presentation/Report	Kirchgessner, Robert Houghton, Nathaniel	The final presentation where we will demonstrate our device (possibly including a video of the device in use.)



Drawing 4: WEMS Gantt Chart

References

[1] "Understanding Your Import Car's Electrical System,"Autohaus Arizona, Inc. 25 January 2009. <http://www.autohausaz.com/html/auto_electrical_systems.html.>
[2] "On-board diagnostics", Wikipedia. 26 January 2009. <http://en.wikipedia.org/wiki/On-Board_Diagnostics>.