ISSN: 0974-2115
www.jchps.com Journal of Chemical and Pharmaceutical Sciences

A Web based standalone vehicle monitoring system

Vignesh RS*, Varun Kumar M
VIT University, Vellore, Tamil Nadu, India - 632014
***Corresponding author: E-Mail: vignashrs@gmail.com**

ABSTRACT

The main aim of the project is to develop a standalone system which monitors the vehicle and data services. This system has two modules, interfacing with the OBD connector to the car and collects the continuous real time data of the vehicle. The data which is the form of hexadecimal is converted to the SI units. This continuous retrieved data is saved into Excel file. This vast data is analyzed and shorted into understandable form in a lookup table in the text file. These two files are attached to email id of the end user of vehicle. This updating of data to mail is done in automatic way in regular intervals in automatic manner. The other module is the analyzed data from OBD is display in the form of various sensor values as the real time performance graph with rich GUI. This display has touch so we can slide right or left and get the other sensor values to display.

KEY WORDS: standalone, OBD Connector, GUI.

1. INTRODUCTION

The standalone system which we opted, as two tasks to be done, first task is to updates the vehicles different sensor data values to email. The mail attachment has two file one is Excel file this makes the user to access the data remotely. Other file is Text file which gives the information to user how vehicle gone into different ridding conditions. Based on the previous ride data he can adjust the next ride such a way that vehicle can be used in efficient manner. This file will be having a lookup table shows each sensor maximum, minimum and average values. This Text file gives the specific ride details in an understanding manner. Second task is to display the real time performance graph for each subsystems of vehicle on the 3.2 inch touch screen display which is attached to the Raspberry Pi. This standalone system is interfaced to car using OBD connector ELM327.

2. EXPERIMENTAL

Hardware: Raspberry Pi as SoC BCM2836 which is having ARM Cortex-A7 quad-core processor is enhanced with image, graphics processing, peripherals and interface options such as EtherCAT. SoC works at 900MHz. It has memory 1GB LPDDR2 and onboard flash 16GB.

Inch Touch Screen Display: The 3.2 inch TFT LCD module is an uncommon outline for Raspberry Pi for convenient application. It highlights a 3.2" showcase with 320x240 16bit shading pixels and resistive touch screen.

ELM327 Connector: The ELM327 is intended to go about as a scaffold between these On-Board Diagnostics (OBD) port and a standard RS232 serial interface. We have USB port on other side instead of standard RS232.

3G Dongle: A dongle is a little bit of equipment that interfaces with another gadget to give it extra usefulness. Which gives the internet access.

Software: As the hardware fully runs on Linux operating system we need to select the better programming platform which compatible to the hardware without any troubleshoot. We chose the java as the programming language which is compatible to the hardware.

For Mailing Module:

Java Mail: The Java Mail API is a standard extension for reading, composing, and sending electronic messages. We use this package to create MUA type programs. Here user interacts with MUA type programs to read and write mails.

Apache POI: For creating Excel This Apache POI is an open source library developed and distributed by Apache Software Foundation. It helps the developer to create, modify and display MS office files using java program.

JSSC: The JSSC stands as Java Simple Serial Connector, this library will be working with serial port from Java. This library was simple and reliable, with JSSC you can get the port names, read and write data, control lines RTS and DTR, receive Event etc. JSSC designed to operate 24/7 multi-threaded systems

For GUI Module:

AWT: The Abstract Window toolkit supports Graphical User Interface (GUI).This model is robust in event handling. On the top of the AWT architecture swing classes can be used as builder.

Design of the System: The system design is mainly depends upon the OBDII data which is serial data retrieved by connecting ELM327 connector. The connections are made as shown below diagram. The 3G dongle is inserted in the USB port for configuring internet connection. A 3.2 inch resistive touch display is interfaced to the GPIO pins of the Raspberry Pi.

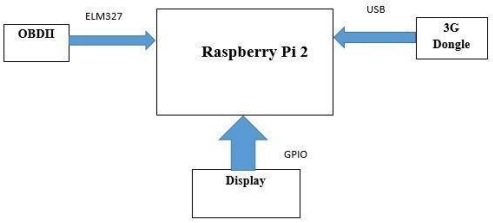

Fig.1. System Design

Implementation of system:

Serial Port Initialization: Java Simple Serial Connector is the library used in the java, which will allow us to access the serial data. After including the jar files we used these objects: Serial Port, Serial Port Event, Serial Port Listener, and Serial Port Exception.

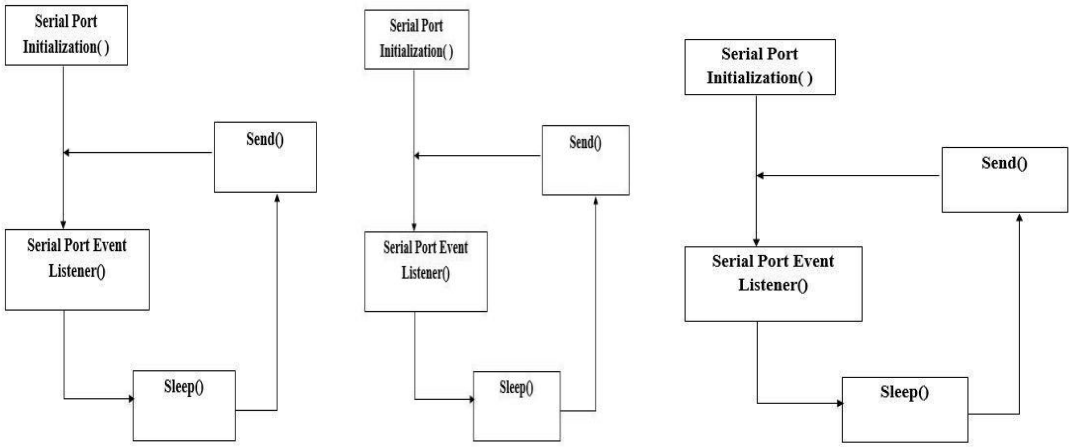

Fig.2. Serial Port Initialization

GUI Module: The GUI module is always Invoked and GUI gets updated according serial data. This invoking will get always a new score which will be update in Paint Component. The different methods which are implemented in for GUI development are; Get Score, Paint Components, Create GUI and Show, Init GUI.

Get Score: This method is used to get the data of different sensor values by giving them different PID values as command values. We fix the max, min values for the graph on X-axis and Y-axis based on the standard values given by the SAE.

Paint Components: All the painting code is placed in PaintComponent method. Whenever it is time to paint the graph this method will be invoked. This method will begin the higher priority in the class hierarchy. The subsystem gets painted whenever this method gets executed. The only method that will override for all the practical purpose is PaintComponent. This is the component of the Swing.

Create GUI and Show: In this method GUI is created and showed. A frame is created in this function, we optimized the frame of the GUI according to display of the Raspberry Pi. We preferred the frame with 320*240. Whenever event of mouse occurs, the related method inside the listener object invoked and required event gets executed.

Init GUI: This method is used to initializing GUI by appending the score value to for every data point and get the content into the main panel. The appending is done by using add to the list.

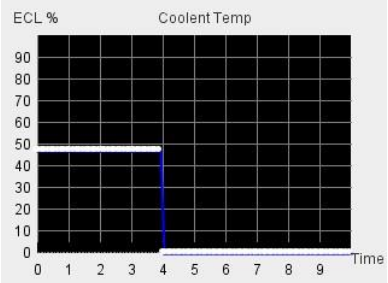

Fig.3. GUI Frame

Mailing Module: The mailing module will send the two attached files to the user mail id. The mailing module will completed when the two required data files are created. So for creating the excel file and text file. Workbook implementation is done; Workbook Creating, Write Workbook, Close Workbook, Shorting Algorithm, Internet configure, Mailing Implementation.

Workbook Creating: There are two spread sheet formats HSSF and XSSF. We use XSSF which is known as XML Spread Sheet File. Once the workbook is created we create a sheet. We create a header to excel sheet, the header will on the top row.

Write Workbook: We write the each sensor value into the respective column of the sensor in each row for every 3 to 5 seconds interval based on the sleep value. The values are saved in an order by creating each cell value and setting the cell value. The values are going to be entered for a period of time, that value we fix as half an hour this value can be changed as our wish.

Close Workbook: Once the writing is done it should be closed, while closing is going to happen at every half an hour of ride. This file is going to save in the specified folder path, that path is going to be access whenever it is required.

Shoring Algorithm: This was implemented for the shorting vast data into small and easily understanding way. For this we taken each sensor value into consideration we made shorting for each sensor. We took parameters like maximum, minimum and average

Internet Configure: This configuration is done directly in the Linux, using shell script. The automatically connecting to 3g network to raspberry pi without any human interaction. We used SAKIS 3G program and configured to pi to get connections.

Mailing implementation: The mailing method is implemented by SMTP protocol. The host we use is "smtp.gmail.com", because we used Gmail as the mailing option. The port we used is 587. Then we get into the session object for mail password authentication.

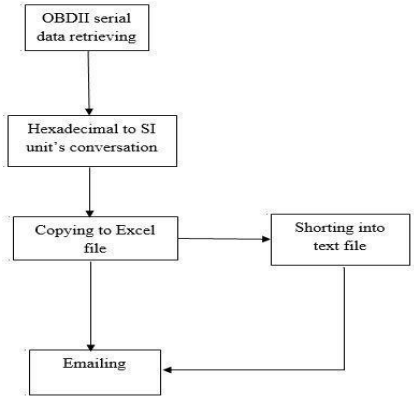

Fig.4. Mailing Module

3. WORKING/ RESULTS

The monitoring module will show different sensor performance graph in the real time. The sensors values which we have taken into consideration to display are Engine Load, ECT, Throttle Position, vehicle speed, fuel pressure, RPM. It is a touch screen display we can select whichever the sensor value we want to display this is optional to the user. The display will show the sensor value for ten seconds of time span. That means the sensor value will be holding on the graph for ten seconds of time. In results figures 6 &7 shows the files which have been attached to mail.

	A	B	C	D	E	F	G	H	I	J	K
1	Time	Engine Lo	Coolent T	Fuel Rail F	IMAP	RPM	Speed	Air Tempe	MAF	Throttle Position	
2	0	32.54902	45.09804	54	99	901	0	54	6	11.2625	
3	0	32.15686	45.09804	54	99	905	0	54	6	11.3125	
4	0	32.15686	45.09804	54	99	901	0	54	6	11.2625	
5	0	32.54902	45.09804	54	99	897	0	54	6	11.2125	
6	0	32.54902	45.09804	54	99	900	0	55	6	11.25	
7	0	32.54902	45.09804	54	99	902	0	55	6	11.275	
8	0	32.15686	45.09804	54	99	900	0	55	6	11.25	
9	0	32.15686	45.09804	54	98	1742	0	58	27	21.775	
10	0	21.56863	45.09804	54	160	4220	0	64	36	52.75	
11	0	0	45.09804	54	111	942	0	58	7	11.775	
12	0	31.76471	45.09804	54	100	897	0	54	6	11.2125	
13	0	32.94118	45.09804	54	99	904	0	54	6	11.3	
14	0	32.54902	45.09804	54	99	899	0	55	6	11.2375	
15	0	32.54902	45.09804	54	99	903	0	55	6	11.2875	
16	0	32.54902	45.09804	54	99	903	0	55	6	11.2875	
17	0	32.54902	45.4902	54	99	903	0	55	6	11.2875	
18	0	32.54902	45.4902	54	99	902	0	56	6	11.275	
19	0	32.15686	45.4902	54	99	903	0	56	7	11.2875	
20	0	32.15686	45.4902	54	99	890	0	56	15	11.125	
21	0	26.66667	45.4902	54	101	1236	0	59	12	15.45	
22	0	32.54902	45.4902	54	101	1261	2	58	11	15.7625	
23	0	31.37255	45.4902	54	101	972	3	57	6	12.15	
24	0	33.72549	45.4902	54	99	906	1	56	6	11.325	
25	0	34.11765	45.4902	54	99	906	0	56	7	11.325	
26	0	24.70588	45.4902	54	100	1222	0	59	9	15.275	

Fig 6.Vehicle's real time engine performance value in Excel file

Sensor	Min. (%)	Max. (%)	Avg. (%)
Engine Load	0.0	35.0	29.0
Engine Coolant	45.0 Å^{0}C	45.0 Å^{0}C	45.0 Å^{0}C
Fuel Pressure	54.0 kpa	54.0 kpa	54.0 kpa
Intake MAP	98.0 kpa	160.0 kpa	102.0 kpa
RPM	0.0 RPM	4220.0 RPM	1024.0 RPM
Speed	0.0 Km/h	3.0 Km/h	1.0 Km/h
Intake Air Temp.	54.0 Å^{0}C	64.0 Å^{0}C	56.0 Å^{0}C
MAF Rate	6.0 gm/sec	36.0 gm/sec	9.0 gm/sec

Fig.7.Vehicle's core engine configuration

4. CONCLUSION

Using a classified system supported to the engine rate, ECT, Throttle position and we calculated engine load could be a teach instance of your time with a model for rating the driving force was developed. The user can read those values in real time. These files area unit hooked up to email id. This transferring of data to email is done in automatic way with regular intervals in automatic manner and consequently deriving riding higher his /her ratings given by the system. The settings were tested out extensively go into the real time world, supported the ratings given by the system. The driving force will be judged or hierarchal.

REFERENCES

Ami Wiesel, Yonina C Eldar, and Shlomo Shamai, Zero-Forcing Precoding and Generalized Inverses, IEEE Transactions on signal processing, 56 (9), 2008.

Apostoloff N & Zeilinksy A, Vision in and out of vehicles: integrated driver and road scene monitoring, International Journal of Robotics Research, 23, 2004, 513-528.

Broadhurst A, Basker S, Kanade T, Monte Carlo road safety reasoning, IEEE Proceeding of Intelligent Vehicle Symposium, 2005, 319-324.

Jungyong Park, Byungju Lee, and Byonghyo Shim, A MMSE Vector Precoding with Block Diagonalization for Multiuser MIMO Downlink, IEEE Transactions on communications, 60 (2), 2012.