CHAPTER 1

INTRODUCTION

1.1 INTRODUCTION

In this chapter, we will present our motivation that leads us to make our project "SMART ENGINE MANAGEMENT SYSTEM". There are many causes of road accidents like the weather conditions, running red lights, teenage drivers, night driving, design defects, unsafe lane change, wrong way driving etc. But on compiling the conclusions based on our survey, we found that there are three main causes of road accidents which are*reckless driving, drunken driving* and *drivers' distraction*. Their surveys on them are given as followed.

1.1.1 RECKLESS DRIVING

More than 90% of road deaths in 2016 were attributed to rash and negligent driving with the latest national crime research bureau statistics revealing 1.5 lakh deaths in 1.35 lakh road accidents due to delinquent driving. The crime data released by the bureau covers all crimes booked under [Indian Panel Code]. The NCRB is yet to release its data on all accidents and suicides that were reported during 2016, which includes those where FIRs are filed.

While the NCRB covers all IPC data, as of now only the road transport ministry has records of all road accidents in 2016. As per this report, 1.51 lakh people died in road accidents and a driver's fault was responsible for 80.3% of these fatalities. Some road safety experts have raised questions on such details and argued that there might be lack of proper investigation in many road accident deaths.

Table 1.1 Reckless driving statistics

City	Accidents		Death		Severity	
	2016	2015	2016	2015	2016	2015
Delhi	↓ 7,375	8,085	↓ 1,591	1,622	↑ 21.6	20.1
Chennai	↑ 7,486	7,328	↑ 1,183	886	↑ 15.8	12.1
Jaipur	↑ 3,004	1,894	↑ 890	476	↑ 29.6	25.1
Bangalore	↑ 5,323	4,834	↑ 835	713	↑ 15.7	14.7
Kanpur	↓ 1,451	1,496	↑ 684	665	↑ 47.1	44.5
Mumbai	↑ 24,639	23,468	↓ 562	611	↓ 2.3	2.6
50 million-plus cities	↓ 89,835	1,11,024	↑ 17,797	16,513	↑ 19.8	14.9

[Accidents, deaths and severity on roads, 2015-16, city-wise; Dipak Dash, Big cities see rise in road accident toll, August 27, 2017: The Times of India]

It has sometimes been argued that road accident deaths are entered under negligent and rash driving in FIRs as piecing together accidents is not always easy. “There is no clause in IPC to put a road accident caused due to pothole or bad engineering. In most of the cases, police reach the spot after accidents have happened and hence they mainly depend on eyewitness accounts. Usually, these eyewitnesses say the vehicle was moving fast,” said a senior police officer. This may lead to an inflation of accidents were negligent driving is the cause, but this remains the chief cause of deaths on the roads.

Since victims of road accidents are mainly compensated from the accumulated third-party insurance premium collected from vehicles, most of the cases are registered under rash and negligent driving head, he added.

“Across the world countries that have brought down road deaths have first done work to get the right data, find the exact reasons. Unless we do this ground work, we won’t be able to treat the disease,” said road safety expert RohitBaluja.

1.1.2 DRUNKEN DRIVING

According to WHO reports, drinking and driving is one of the main causes of road crashes. While in high income countries, about 20% of fatally injured drivers have excess alcohol in their blood, in some low and middle-income countries the figure may be up to 69%.

Prohibition seems to have helped Bihar rein in deaths caused by drunk driving, with the number of such fatalities dropping 60% between 2015 and 2016. The liquor ban came into force in the state in 2016.

The number of fatalities caused by drunk driving almost doubled in Uttar Pradesh in the same span -from 1,404 in 2015 to 2,716 in 2016 -according to the government's latest report on road accidents.

Bihar is one of three states that saw fewer deaths, injuries and accidents caused by drunk driving in 2016 than the year before, the others being Jharkhand and Haryana. However, West Bengal, Punjab, Pondicherry and Odisha are among the states and Union territories where the number of deaths in drunk-driving-related accidents has gone up. Delhi registered a five-fold increase in such fatalities -while six persons died in drunk-driving-related cases in 2015, 30 were killed last year.

After TOI first reported on April 21 the lower drunk driving-related road fatalities in Bihar, the state government had cited the drop as one of the achievements of its complete ban on liquor.

At 541, Bihar reported the maximum decline in drunk-driving-related road deaths, in absolute figures, between 2015 and 2016, among the eight states and Union Territories where the number of deaths fell.By percentage, the decline was sharpest in Haryana.

Table 1.2 Drunken driving statistics

DRUNK DRIVING DEATHS			
State	(2015)	(2016)	Change
	Deaths		
Bihar	867	326	-541
Jharkhand	906	408	-498
Uttar Pradesh	1,404	2,716	1,312
Rajasthan	344	372	-28
Haryana	478	165	-313
Madhya Pradesh	705	547	-158
All India	**6,755**	**6,131**	**-624**

[Drunk driving deaths, 2015-16, state-wise; Dipak Dash, Drink driving deaths down 60% in Bihar, Sep 9,2017: The Times of India]

Though 624 fewer road deaths across the country in 2016 on account of drunk driving is taken as a positive indicator, law enforcement authorities and traffic safety experts admit that official data shows a much lower number than is the reality.

1.1.3 DRIVERS' DISTRACTION

Emerging as one of the top causes of death among the most productive age groups, road crashes have developed into a major public health crisis across the world. According to the World Health Organization (WHO), road crashes kill 1.2 million people and permanently disable another 50 million every year.

Over the last decade, road crash has become the tenth leading cause of death in the world, and is predicted to rise to the fifth position by 2030. India is the number one contributor to global road crash mortality and morbidity figures. Every hour, 16 lives are lost to road crashes in India. In the last decade alone, India lost 1.3 million people to road crashes and another 5.3 million were disabled for life.

The WHO categorizes driver distraction as an important risk factor for road crash injuries. The United States Department of Transportation terms distracted driving as one of the most dangerous driver behaviors and an epidemic which has increased with the proliferation of mobile phones.

Distracted driving is defined as any activity that diverts a person's attention from their primary task of driving. These types of activities include the usage of a mobile phone, eating and drinking, conversation with co-passengers, self-grooming, reading or watching videos, adjusting the radio or music player and even using a

GPS system for navigating locations. Amongst these, mobile phone usage is said to be the most distracting factor.

In the last decade alone, India lost 1.3 million people to road crashes and another 5.3 million have been seriously injured. India has the highest number of road crash fatalities, with a crash occurring every minute and one death every four minutes. While it has just 1% of the world's vehicles, India accounts for over 10% of global road crash fatalities. According to the 'Road Accidents in India', 2015 report of Ministry of Road Transport and Highways (MoRTH), 146,133 people were killed in road crashes in 2015 alone including 12,589 children. This number is not only the highest that India has ever recorded in history, but it represents a 53.9% increase over the last decade, and nearly a ten-fold increase since 1970.

Table 1.3 Distraction driving statistics

CITY	TOTAL NO. OF ROAD CRASHES	POPULATION	ROAD CRASHES/10,000 PEOPLE	TOTAL NO. OF ROAD CRASH DEATHS
Bengaluru	5,001	8,520,435	5.86	890
Chennai	7,328	8,653,521	8.47	886
Delhi	7,148	16,349,831	4.37	1,316
Jaipur	3,151	3,046,163	10.36	939
Kanpur	1,051	2,765,348	3.80	889
Kolkata	4,981	14,112,536	3.53	421
Mangaluru**	-	619,664	-	-
Mumbai	2,551	18,414,288	1.38	611

['Accidental Deaths & Suicides in India' 2015, National Crime Records Bureau]

Not only does the loss or impairment of a breadwinner of a family in a road crash inflict emotional trauma on lakhs of families, it imposes a severe financial burden by pushing entire households into poverty. In a 2014 report, the erstwhile Planning Commission of India had estimated that the annual cost of road crashes in India is 3% of its GDP. With India's GDP in 2015-16 being INR 136 lakh crore, these figures translate into an annual monetary loss of INR 4.07 lakh crore. Ironically, it is over five times the budget of the Ministry of Road Transport and Highways, the nodal agency for ensuring road safety in India.

Table 1.4 Phone usage statistics

TABLE 4: TYPE OF MOBILE PHONE USED BY TYPOLOGY

	Feature phone	Smart phone without internet	Smart phone with internet	
	23%	10%	67%	male
	19%	8%	73%	female
	29%	12%	59%	male
	11%	11%	78%	female
	63%	11%	26%	male
	0%	0%	0%	female
	76%	8%	16%	male
	0%	0%	0%	female
Total	53%	10%	37%	male
	15%	10%	75%	female

[Courtesy: Vodafone]

According to the Telecom Regulatory Authority of India (TRAI), India currently has 1 billion mobile phone subscribers.5 Due to the high penetration of mobile phones, their use on the road has also increased as proven by this study with 1 out of 2 respondents of this survey having used a phone while driving. Unfortunately, in India, there is no data that is being currently captured at the crash site by authorities to record the connection between mobile phone usage and crashes.

The Government of India in a 2015 report revealed that "2,270" people were killed in "8,359" crashes due to 'driver's inattentiveness'.6 But, there is no sub-set of this data that points out as to the exact number out of these 8,359 crashes which were caused as a result of mobile phone usage. On the other hand, understanding the growing threat that mobile phone usage during driving possesses, countries like USA have been capturing data at the crash site. For example, in 2014, there were 2,955 fatal crashes that occurred on U.S. roadways that involved distraction (10% of all fatal crashes), out of which there were 385 fatal crashes reported to have involved the use of cell phones as distractions (13% of all fatal distraction-affected crashes).

1.2 CONCLUSION

The recent survey report shows the decrease in number of road accidents in the first two major causes. This has been pulled through social awareness, prohibitions and other safety techniques. But the remaining one i.e. distraction is increasing day by day due to advance in mobile industries, fast food chain etc. These industries can’t be stopped because it affects the rise of the humans' living standards. But still, we can’t ignore the lives of billions of people and hence we want to make this project for the sole purpose of reducing the number of road accidentsoccurring due to the distraction of driver.

CHAPTER 2

LITERATURE SURVEY

Before we present our project idea of "SMART ENGINE MANAGEMENT SYSTEM", we would like to introduce some research papers that we collected throughout days and nights so that we can solidify the foundation of making this project. Our collected research papers have shown us some different ideas that we can take during the making of our project. This chapter will also show inferences we drawn out from these research paper so that we can make of selecting these papers.

2.1 RESEARCH PAPERS

[1] Dong Hui,et.al in,"The ECU control of diesel engine based on CAN",

The authors mentioned that the dynamic, comfort and emission performances of vehicle directly dependent on the control of an engine. Engine's electronic control unit (ECU) receives the signals from the sensors, and gets the current engine status. Then depending on the signals, the ECU calculates fuel injection rate, injection timing and quantity. Accordingly the actuators which are driven by ECU carry out the better fuel control. Whether the vehicle could work efficiently and reliably almost dependents on whether the engine ECU is normal or not. One way to test engine ECU is using load box which can diagnose and calibrate the faults. This paper chooses CAN bus to realize the communication between ECU and load box because of its widely using and well function. This paper formulates the CAN bus communication protocol which meets the needs of load box test and designs the CAN communication module and signal input/output module. MPC5633 made by Freescale is used as the microcontroller unit (MCU) in ECU. Through the experiments showed in paper proves the testing programs of ECU control can realize the function requirements.

[2] Mark Gallagher,et.al in,"The xk8 engine management system and electronic engine control module",

The authors mentioned that the increasing demand for feature enhancements on passenger vehicles combined with more stringent emissions and legislative requirements world-wide has lead to a high level of complexity within engine control modules, associated emission control hardware and equally important software. This paper briefly explains how the functions of the AJV8 engine control work and how the diagnostics form an integrated part of the system design.

[3] N.Dheerthi,et.al in,"Real Time Data Monitoring System with Intelligent Vehicle Tracking using ARM7",

The authors made this project is to design and develop an affordable user friendly low cost vehicle black box i.e. real time data monitoring with speed control operation of the vehicle in restricted zone. The system does major operation which is data monitoring that records various parameters like, state of driver(both drowsiness of driver, alcohol consumption of driver), seat belt state, crash state, engine temperature and speed control of the vehicle. System design uses ARM7 as embedded microcontroller. It also uses GPS and GSM modules to identify the position of vehicle and to transfer the real time data to central station. The main advantage of this system is low power consumption since ARM7 is used as an embedded microcontroller and also cost to implement overall system is low.

[4] E.Vargil Vijay,et. al in, ,"Electronic Control Unit for an Adaptive Cruise Control System & Engine Management System in a Vehicle using Electronic Fuel Injection",

This paper is about simulation design of an Electronic Control Unit (ECU) for an adaptive cruise control (ACC) and for an engine management system (EMS). The simulation model allows processing of various sensors data like engine speed, engine temperature, and distance from the front vehicle. By collecting data from various sensors, ECU controls the vehicle speed by using electronic fuel injector actuator. ECU model simulates the failures to design a fault-tolerant system and controls the engine speed with the aid of an open loop control technique using EMS. ECU provides safety to the user by avoiding the collision by using IR Sensor and deployment of the air bags using MEMS accelerometer in the case of emergency using Adaptive Cruise Control (ACC). ECU mainly concentrates on functionality of fuel metering. This Simulation model could be used as a tool for swift development and test models of ECU in order to control the engine in laboratory for fuel economy and engine performance and safety improvement purposes for vehicles. The simulation has been realized using Simulink and State flow, which are components of Mathwork's MATLAB software.

[5] Masato Yamauchi,et. al in,"Toward formal verification of ECU for gasoline direct injection engines",

Currently, electronic control units are employed for almost all automobiles. An engine control unit (ECU), which is also called a power train control module, is an electronic control unit for an engine. A fault in computer software for an ECU may cause a hazardous event or a fatal accident. So, in this study, the authors employ a formal method to design a computer program of ECU. Specifically, the authors employ a formal method called I/O- automation, and model an ECU of a gasoline direct injection engine with a specification language based on I/O-automation theory. They also introduced an I/O-automation that is with regard to the correctness of the timing of engine's ignition, and then discussed how to prove the correctness of the design of ECU.

[6] AamirSarwarJahan,et.al in,"GPS enabled speed control embedded system speed limiting device with display and engine control interface",

This paper aims to outline the design of an embedded system that will automatically control the speed of a motor vehicle based on its location determined by a GPS device. The embedded system will make use of an AVR ATMega128 microcontroller connected to an EM-406A GPS receiver. The large amount of location input data justifies the use of an ATMega128 microcontroller. The output of the ATMega128 will be an LCD module which will display information of the current and the set-point speed of the vehicle at the current position. A discrete indicator LED will flash at a pre-determined frequency when the speed of the vehicle has exceeded the recommended speed limit. Finally, the system will have outputs that will communicate with the Engine Control Unit (ECU) of the vehicle. The ECU is simulated as an external device with two inputs that will acknowledge pulse-trains of particular frequencies to limit the speed of a vehicle. The speed control system was programmed using mixed language C and Assembly with the latter in use for some pre-written subroutines to drive the LCD module. To check current speed that will be compared against the recommended speed for the vehicle's location. The memory locations in the ATMega128 can be used to store set-point speed values against a particular set of location co-ordinates.

[7] Stephan Mühlbacher-Karrer,et.al in,"A Driver State Detection System—Combining a Capacitive Hand Detection Sensor With Physiological Sensors",

With respect to automotive safety, the driver plays a crucial role. Stress level, tiredness, and distraction of the driver are therefore of high interest. In this paper, a driver state detection system based on cellular neural networks (CNNs) to monitor the driver's stress level is presented. They propose to include a capacitive-based wireless hand detection (position and touch) sensor for a steering wheel utilizing ink-jet printed sensor mats as an input sensor in order to improve the performance. A driving simulator platform providing a realistic virtual traffic environment is utilized to conduct a study with 22 participants for the evaluation of the proposed system. Each participant is driving in two different scenarios, each representing one of the two no-stress/stress driver states. A "threefold" cross validation is applied to evaluate the concept. The subject dependence is considered carefully by separating the training and testing data. Furthermore, the CNN approach is benchmarked against other state-of-the-art machine learning techniques. The results show a significant improvement combining sensor inputs from different driver inherent domains, giving a total related detection accuracy of 92%.

[8] TullioCuatto,et.al in,"A case study in embedded system design: an engine control unit DAC",

A number of techniques and software tools for embedded system design have been recently proposed. However, according to the authors, the current practice in the designer community is heavily based on manual techniques and on past experience rather than on a rigorous approach to design. To advance the state of the art, it is important to address a number of relevant design problems and solve them to demonstrate the power of the new approaches. They chose an industrial example in automotive electronics to validate our design methodology an existing commercially available Engine Control Unit. They discuss in detail the specification, the implementation philosophy, and the architectural trade-off analysis. They also analyse the results obtained with this approach and compare them with the existing design underlining the advantages offered by a systematic approach to embedded system design in terms of performance and design time.

[9] SwapnilAwateb,et.al in,“Development of custom-made engine control unit for a research engine”,

In this paper, hardware development, software coding using embedded C programming language for an Arduino ATMEGA microcontroller. Calibration of electronic control unit (ECU) was tested for a research engine to control the fuel injection flow rate with respect to the suction top dead centre (TDC) of the engine. The system could control the fuel flow rate with a variable reluctance sensor, an Arduino microcontroller and a solenoid operated injector in a closed loop domain with a varying pulse width modulator controlled exclusively by the engine operator. The injection flow rates were measured and calibrated with the calculated fuel flow rates for different equivalence ratios of the engine. The results showed a very close match between the measured fuel flow rates after calibration and the calculated fuel flow rates at 1500 RPM. The gasoline mass flow rate error was reduced from 40% to 3.25% by compensating the ON/OFF time of the pulse width.

[10] MSc. AlexandreGiordaniAndreoli,et.al in,“Development of an electronic management system for a rotary combustion engine”,

Nowadays, the electricity generation is a commonly discussed subject. A large number of studies about converting mechanical energy into electrical energy are done. According to the authors however, there is also the need for studying new internal combustion engines so the chemical energy can be more efficiently converted into mechanical energy or movement. This paper describes the development of an electronic management system for an embedded control and management of a new concept spark ignition rotary engine prototype for a promising use in electrical energy generation. The embedded software creating was explained since its project through its implementation. Several aspects of implementation were discussed, from the hardware, like choosing the microcontroller between PIC18F4550 to Arduino Mega 2560.

[11] Felix Sygulla,et.al in ,“A flexible and low-cost tactile sensor for robotic applications”

For humans, the sense of touch is essential for interactions with the environment. With robots slowly starting to emerge as a human-centric technology, tactile information becomes increasingly important. Tactile sensors enable robots to gain information about contacts with the environment, which is required for safe interaction with humans or tactile exploration. In this paper, they present a novel design for a tactile sensor that can be built with low-cost, widely available materials, and low effort. The sensor is flexible, may be cut to arbitrary shapes and may have a customized spatial resolution. Both pressure distribution and absolute pressure on the sensor is detected. An experimental evaluation of their design shows low detection thresholds as well as high sensor accuracy.

[12] J. Mamala, et. al in,"Hardware-in-the-Loop Type Simulator of Spark Ignition Engine Control Unit",

The paper presents simulator of injection-ignition system of internal combustion engine with spark ignition developed at the Technical University of Opole. This system is based on Bosch series engine ECU with software 7.5. It allows analysing of motor parameters using both on-board diagnostic system and data network CAN BUS under different operating conditions of the system. For this purpose, the simulator is equipped with a number of additional devices enable to generate repetitive input signals for the ECU, allowing opening the actual working conditions. Crucial for the simulator was to generate the signals informing ECU about the instantaneous position of the crankshaft and camshaft. For this purpose, the module arduino was used. This type of solution allows, recreating momentary engine operating conditions. This article also presents the concept of development with new features and sentences feasible for this type of device and simulation mode.

[13]Seong Won Park,et. al in,"A Flexible Capacitive Pressure Sensor for Wearable Respiration Monitoring System",

This paper presents the design, fabrication, and characterization of a wearable capacitive pressure sensor for respiration-monitoring systems. For the dielectric layer of the proposed capacitive sensor, Porous Ecoflex with a porosity of ~36% was prepared from a manually made sugar cube via a simple melting process. A Polydimethylsiloxane (PDMS) based Silver nanowire (AgNWs) and Carbon fibers (CFs) thin films were used for the sensor electrodes. The fabricated flexible pressure sensor exhibited a high sensitivity of 0.161 kPa-1 for low pressure regime (< 10 kPa), a wide working pressure range of < 200 kPa, and a high durability over 6,000 cycles. Since the proposed sensor is flexible and resizable, it can be integrated into clothes and easily placed at any location of the human body. Finally, the practicality of the sensor was successfully demonstrated by integrating the sensor into a waist belt to monitor the real-time respiration signal of the human being. The finding is highly useful to monitor respiration signal for the detection of diseases such as sleep apnea, asthma and others.

[14]B. Ben Slimen, et. al in, "A Hierarchical Control Scheme Based on Prediction and Preview: an Application to the Cruise Control Problem",

This paper deals with a hierarchical architecture of the Engine Control Unit (ECU) made-up of two levels: the higher one, called powertrain level, computes the current and the future torque demands; the lower level, called engine level, uses these information to provide an appropriate motor control. In continuous-time framework, a

H2-control design with preview is proposed for the lower level and a H2-control with input time-delay is considered in the higher one. The relevance of this ECU configuration is illustrated, by simulation, in case of a cruise control problem. Handling this problem, simplified models of the torque producer and the driveline are used.

[15]C. Kandler, et. al in, "A novel mid-range approach for idle speed control of a hybrid electric powertrain"

In this paper the problem of idle speed control for the powertrain of a hybrid electric vehicle is investigated. On the one hand, the proposed control structure ensures control of the powertrain's idle speed, by using the electric motor as a secondary actuator besides the combustion engine. On the other hand, it enables the use of the electric motor as a generator by appropriate load level shifting. The coordination of the two actuators is provided by a novel control structure, which is based on the idea of mid-ranging control. Based on a nonlinear system description, a suitable design model is derived, capturing the fundamental properties of the powertrain. The nonlinear dynamic behavior of the intake manifold pressure is handled via an inversion-based linearization. The controller design itself is based upon a frequency shaped design, which allows for explicit consideration of the powertrain's oscillatory behavior, which is due to a dual mass flywheel. Time delays, which occur in both the plant input and output channels, are encountered by extending the man filter with a predictor. Finally, measurements from a passenger car are presented to validate the performance of the new controller structure.

2.2 INFERENCES DRAWN OUT OF PAPERS

The following inferences were drawn out from the above mentioned research papers are given as below:

- CAN bus to realize the communication between ECU and load box because of its widely using and well function. CAN bus communication protocol which meets the needs of load box test, and designs the CAN communication module and signal input /output module
- GPS and GSM modules can be used to identify the position of vehicle and to transfer the real time data to central station. The main advantage of this system is low power consumption and also cost to implement overall system is low.
- An ECU of a gasoline direct injection engine with a specification language based on I/O-automaton theory was modelled to prove the correctness of the design of ECU
- A Capacitive-based wireless hand detection (position and touch) sensor for a steering wheel utilizing ink-jet printed sensor mats as an input sensor. They can be used on the steering wheel for taking the input from the driver.

- A novel design for a tactile sensor can be built with low-cost, widely available materials, flexible and low efforts were presented. They also can be used on the steering wheel for taking the input from the driver.
- Bosch series engine ECU with software 7.5 allows analysing of motor parameters using both on-board diagnostic system, and data network CAN BUS under different operating conditions of the system.
- There is a wearable capacitive pressure sensor for respiration-monitoring systems for the detection of diseases such as sleep apnea, asthma and others, which can be used as an add-on in our project.
- Engine Control Unit (ECU) made-up of two levels: the higher one, called powertrain level and the lower level called the engine level.

CHAPTER 3

PROBLEM DEFINITION

3.1 INTRODUCTION

Out of the estimated 1.4 million serious road accidents/collisions occurring annually in India, hardly 0.4 million are recorded. Further, only a minimal percentage of these collisions are scientifically investigated, in the absence of which, the real causes and consequences are never known. Therefore remedial measures as well as punishment for the violators are also arbitrary.

On account of various political and socio-economic conditions, generally, the larger vehicles are often labelled the culprit in cases of vehicle-to-vehicle crashes. Road safety can only be improved when we understand its causes and consequences of road accidents/collisions so as to work out remedial measures.

World Health Organization (WHO) in its first ever Global Status Report on Road Safety pointed to speeding, drunk driving, seat belts and driver carelessness as the main contributing factors. Every hour, 40 people under the age of 25 die in road accidents around the globe. According to the WHO, this is the second most important cause of death for 5 to 29 year olds.

In India alone, the death toll rose to 14 per hour at the ending of the last decade as opposed to 13 the previous year. The total number of deaths every year due to road accidents has now passed the 135,000 mark, according to the latest report of National Crime Records Bureau or NCRB.

3.2 AIM

In order to avoid the accidents caused due to carelessness of the driver, various technologies and methods have been developed. This project is one of such contributions.

The aim of this project to implement a ‘Smart engine management system’ that automatically detects the status of driver’s hands on the steering wheel and takes action accordingly.

3.3 BASIC PRINCIPLE

In this project, the speed of the engine will be controlled in accordance with the status of driver's hands (i.e. whether they are on the steering wheel or not). We will be dealing with the sensors on the steering wheel, that’ll input the status of the driver’s hands’ placement on the steering wheel, a microcontroller for receiving the inputs and processing the data accordingly and sending the output to the ECU for further processing and the ECU that will carry out the required task i.e. controlling the speed of the engine.

Here are some particular things that we intend to do to make a car safe from road accidents:

- The parameters written within an ECU will be changed by an external microcontroller.
- The new changed parameters will be corresponded to the changed speed of the engine that was done in accordance with the placement of driver's hands on the steering wheel.
- The parameters will be kept on changing as long as the driver's hands' status changes i.e. if the driver removes his/her hands, the car will be slowed down and vice versa.

3.3 FLOWDIAGRAM OF THE PROJECT

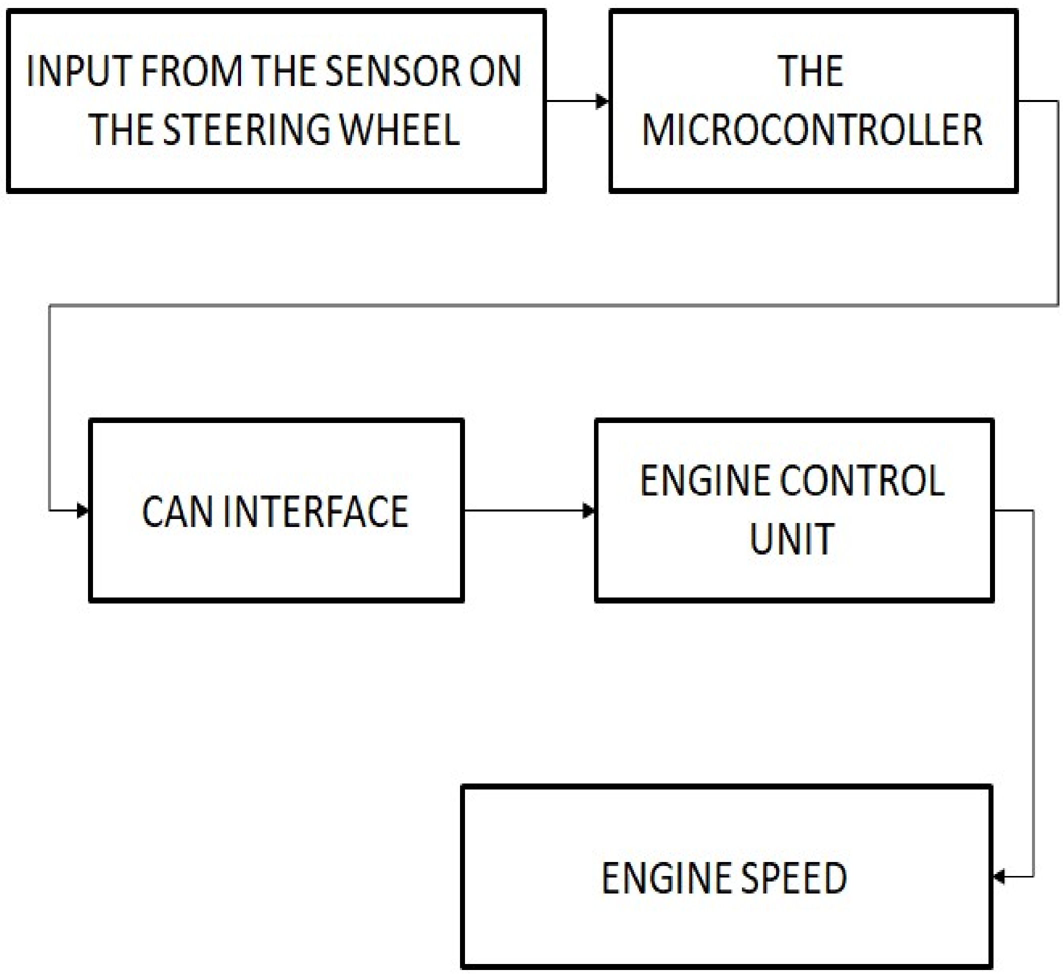

Figure 3.1: General block diagram of the project

- The above figure shows the connection between the inputs (i.e. the steering wheel) and the output (controlled engine speed).
- The input from the sensors that are mounted on the steering wheel is sent to the microcontroller.
- The microcontroller takes a decision based on the algorithm monitoring the sensors.

- The decision taken results in the generation of a map that is sent to the engine control unit (ECU), which will update its parameters' values.
- The communication is realized using CAN interface, which is an abbreviation of controller area network. The ECU having received the required data will in turn control the speed of the engine.

3.3.1 Input from the Steering Wheel

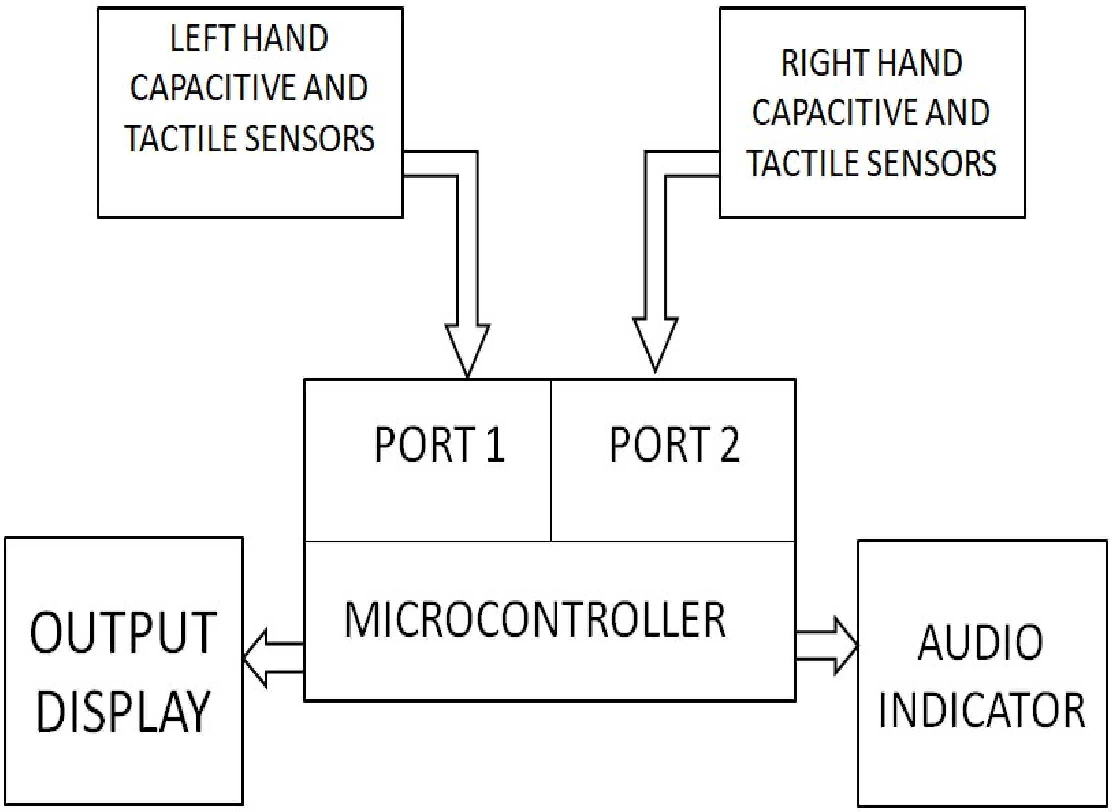

Figure 3.2: Block diagram showing the connection between the sensors and the microcontrollers

- The sensors are mounted on the steering wheel, which are a combination of capacitive and tactile sensors. They are divided into the sensors dealing with the left hand and the sensors dealing with the right hand.
- Both the sensors are connected to the microcontroller using two ports of the microcontroller. The changing physical parameters are reflected through the sensors to the microcontroller.
- An additional display displaying the output and an audio indicator (in the form of a beeper) are also connected to the microcontroller for indicative purposes.
- The microcontroller sends the signal to the indicators (audio and visual) based on the inputs from the sensors. The output to the ECU is also sent by the microcontroller, as shown in the figure 3.1.

3.3.2 Microcontroller and ECU Interface

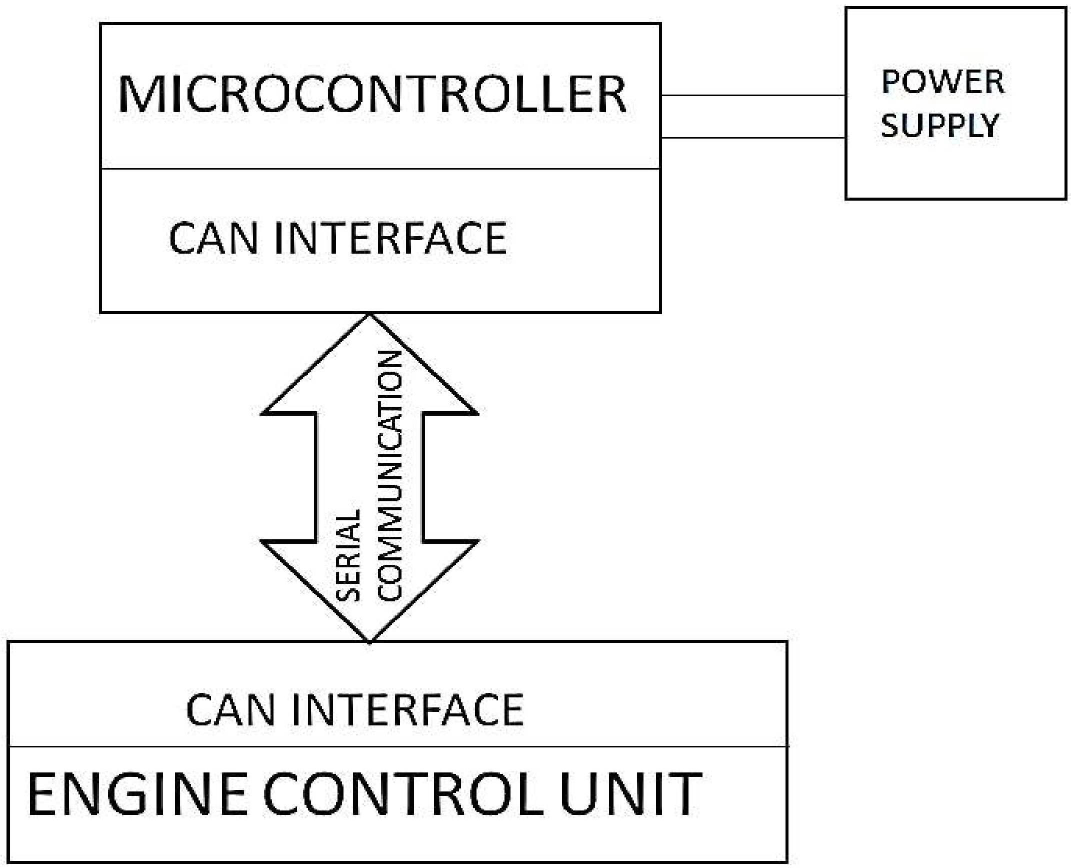

Figure 3.3: Interfacing between the microcontroller and the engine control unit

- The figure shows the interfacing between the microcontroller and the engine control unit, that is realized using CAN standards.
- The CAN interfacing involves the communication between two different devices using serial communication i.e. communication bit by bit. Another reason behind the use of CAN communication is because it is the only one supported by the ECUs, nowadays.
- Power supply is given to the microcontroller, which is a battery. For future purposes, we intend to use a controlled voltage as the input power form the vehicle's alternator is fluctuating.
- The power required by the ECU is delivered to it using a different circuitry (inbuilt in vehicle's electronics).

3.3.3 ECU with Sub-Controllers

- The ECU is connected to various sub-microcontrollers, which in turn control the various sub-operations.

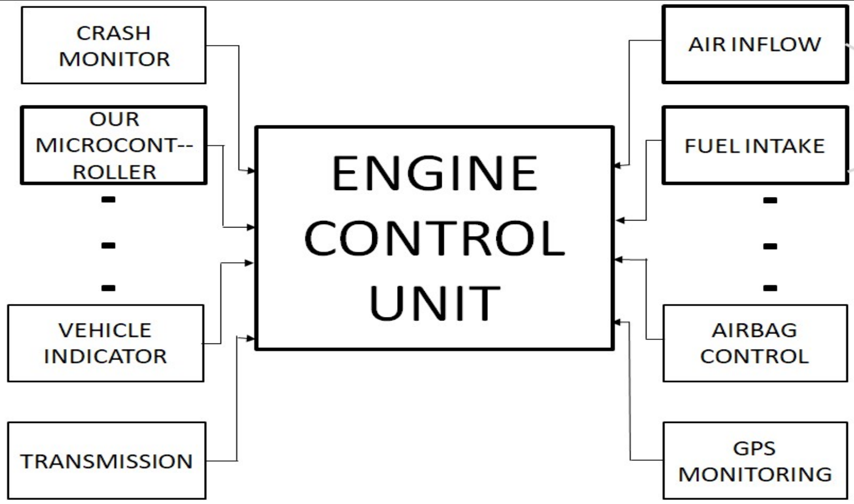

Figure 3.4: ECU connected to various sub-controllers

- The microcontroller (our) will send the signal to the ECU, which also takes inputs from air inflow and fuel intake microcontrollers. The ECU will also send the signal to these both controllers to change the parameters that in turn control the sped of the engine accordingly.

3.3.4 Speed Monitoring By ECU

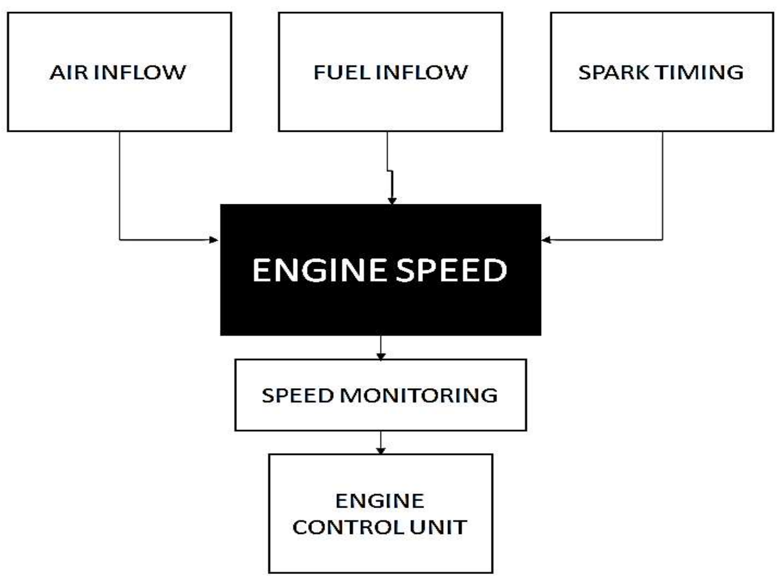

Figure 3.4: Engine speed feedback to the ECU

- The block diagram shows that fuel intake, air inflow and the spark timing control the speed of the engine and these three parameters can be varied to achieve variable engine speeds.
- The engine speed is monitored using a speed gauge mounted on the engine itself. The gauge is a gear with one tooth missing. The gauge rotates at the engine speed and the speed of the engine is monitored by the number of times the gap occurs in a given frame of time.

- The monitored speed is sent to the ECU, which makes the decisions accordingly.

3.3.5 Project approach

Since with all the explanation we have given above, we were not able to work on ECU due to following reasons

2. **Cost of ECU**: the cost of ECU for normal cars is around 50,000 to 1.5 lakh rupees. For expensive car, it is around 7 to 10 lakh rupees.
3. **Communication port:** since for the purpose of safety, the companies provide no communication port so that it could not be tampered. But there are some companies that provide the manual tuned ECU but the cost factor make the hurdle for us.
4. **Security**: there is some tight security provided to the ECU, there is the chip called the 'immobilizer'. It halts the connection between the ECU of one car to another car. Even if we do, the ECU of one car will burn the ECU circuit.

With all these due reasons we are making our own ECU to execute our project smart engine management system.

3.4 HARDWARE USED

3.4.1 Arduino UNO

Arduino Uno is a microcontroller board based on the ATmega328P (datasheet in appendix section). It has 14 digital input/output pins (of which 6 can be used as PWM outputs), 6 analog inputs, a 16 MHz quartz crystal, a USB connection, a power jack, an ICSP header and a reset button. It contains everything needed to support the microcontroller; simply connect it to a computer with a USB cable or power it with a AC-to-DC adapter or battery to get started.

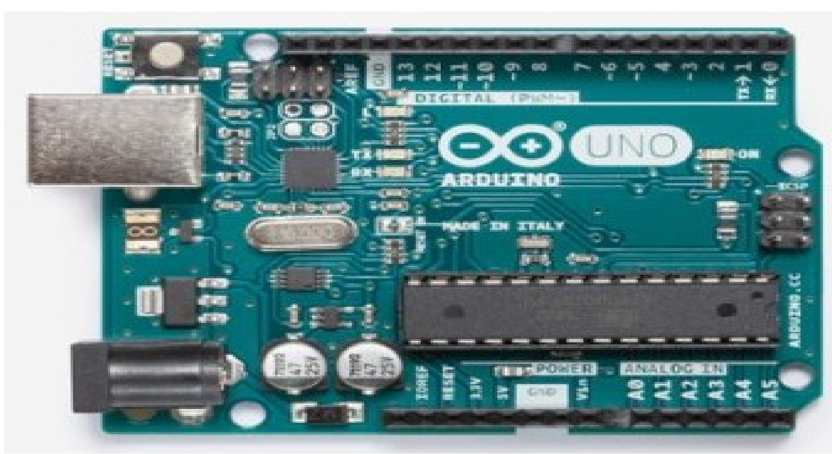

Fig 3.5 Arduino UNO

The Uno differs from all preceding boards is that it does not use the FTDI USB to serial driver chip. Instead, it features the Atmega16U2 (Atmega8U2 up to version R2) programmed as a USB-to-serial converter. Revision 2 of the Uno board has a resistor pulling the 8U2 HWB line to ground, making it easier to put into DFU mode. Revision 3 of the board has the following new feature: 1.0 pinout added SDA and SCL pins that are near to the AREF pin and two other new pins placed near to the RESET pin, the IOREF allow the shield to adapt the voltage provided from the board. In future, shields will be compatible both with the board that uses the AVR, which operate with 5V and with the Arduino Uno, which operate with 3.3V. The second one is a not connected pin that is reserved for future purposes.

- Stronger RESET circuit.
- Atmega 16U2 replace the 8U2.

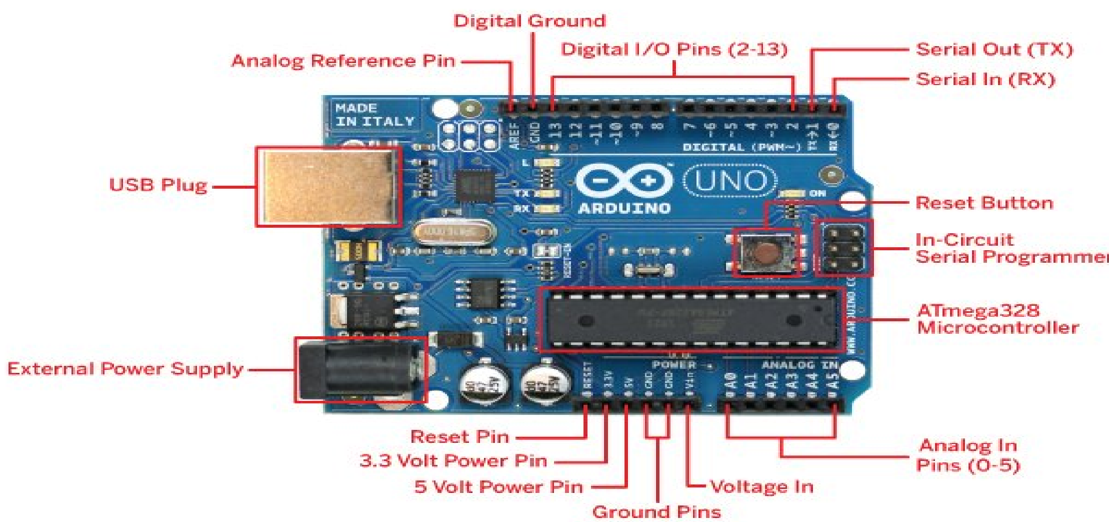

Fig. 3.6 Pin out of arduino

"Uno" means one in Italian and is named to mark the upcoming release of Arduino 1.0. The Uno and version 1.0 will be the reference versions of Arduino, moving forward. The Uno is the latest in a series of USB Arduino boards, and the reference model for the Arduino platform; for a comparison with previous versions.

Table 3.1 Specification of Arduino

Microcontroller	ATmega 328
Operating voltage	5v
Input Voltage (recommended)	7-12 V
Input Voltage (limits)	6-20 V
Digital I/O Pins	14 (6 of PWM)

Analog Input Pins	6
DC current per I/O Pin	40 mA
DC current for 3.3V Pin	50 mA
Flash Memory	32 KB of which 0.5 KB used by bootloader
SRAM	2 KB
EEPROM	1KB
Clock Speed	16 Mhz

Power

The Arduino Uno can be powered via the USB connection or with an external power supply. The power source is selected automatically. External (non-USB) power can come either from an AC-to-DC adapter (wall-wart) or battery. The adapter can be connected by plugging a 2.1mm center positive plug into the board's power jack. Leads from a battery can be inserted in the Gnd and Vin pin holders of the POWER connector.

The board can operate on an external supply of 6 to 20 volts. But if supplied more than 12V, the voltage regulator may overheat and damage the board. The recommended range is 7 to 12 volts.

The power pins are as follows:

- **Vin:** The input voltage to the Arduino board when it's using an external power source (as opposed to 5 volts from the USB connection or other regulated power source). You can supply voltage through this pin, or, if supplying voltage via the power jack, access it through this pin.
- **5V:** This pin outputs a regulated 5V from the regulator on the board. The board can be supplied with power either from the DC power jack (7 - 12V), the USB connector (5V), or the Vin pin of the board (7-12V). Supplying voltage via the 5V or 3.3V pins bypasses the regulator, and can damage your board.
- **3.3V:** A 3.3 volt supply generated by the on-board regulator. Maximum current draw is 50 mA.
- **GND:** Ground pins.

Memory

The ATmega328 has 32 KB (with 0.5 KB used for the boot loader). It also has 2 KB of SRAM and 1 KB of EEPROM (which can be read and written with the EEPROM library).

Input and output

Each of the 14 digital pins on the Uno can be used as an input or output, using pinMode(), digitalWrite(), and digitalRead() functions. They operate at 5 volts. Each pin can provide or receive a maximum of 40 mA and has an internal pull-up resistor (disconnected by default) of 20-50 KOhms. In addition, some pins have specialized functions:

- **Serial: 0 (RX) and 1 (TX).** Used to receive (RX) and transmit (TX) TTL serial data. These pins are connected to the corresponding pins of the ATmega8U2 USB-to-TTL serial chip.
- **External Interrupts: 2 and 3.** These pins can be configured to trigger an interrupt on a low value, a rising or falling edge, or a change in value.
- **PWM: 3, 5, 6, 9, 10, and 11.** Provide 8-bit PWM output with the analogWrite() function.
- **SPI: 10 (SS), 11 (MOSI), 12 (MISO), 13 (SCK).** These pins support SPI communication using the SPI library.
- **LED: 13.** There is a built in LED connected to digital pin 13. When the pin is HIGH value, the LED is on, when the pin is LOW, the LED is off.

The Uno has 6 analog inputs, labeled A0 to A5, each of which provide 10 bits of resolution (i.e. 1024 different values). By default they measure from ground to 5 volts, though is it possible to change the upper end of their range using the AREF pin and the analogReference() function. Additionally, some pins have specialized functionality:

- **TWI: A4 or SDA pin and A5 or SCL pin.** Support TWI communication using the wire library.
- **AREF.** Reference voltage for the analog inputs. Used with analogReference().
- **Reset.** Bring this line LOW to reset the microcontroller. Typically used to add a reset button to shields which block the Uno on the board.

Communication

The Arduino Uno has a number of facilities for communicating with a computer, another arduino, or other microcontrollers. The ATmega328 provides UART TTL (5V) serial communication, which is available on digital pins 0 (RX) and 1 (TX). An ATmega16U2 on the board channels this serial communication over USB and appears as a virtual COM port to software on the computer. The 16U2 firmware uses the standard USB COM drivers, and no external driver is needed. However, on Windows, an .inf file is required. The arduino software includes a serial monitor which allows simple textual data to be sent from the arduino board. The RX and TX LEDs on the board will flash when data is being transmitted via the USB-to-serial chip and USB connection to the computer (but not for serial communication on pins 0 and 1). A software serial library allows for serial communication on any of the Uno's digital pins.

3.4.2 Tactile sensor

A tactile sensor is a device that measures information arising from physical interaction with its environment. Tactile sensors are generally modeled after the biological sense of cutaneous touch which is capable of detecting stimuli resulting from mechanical stimulation, temperature, and pain (although pain sensing is not common in artificial tactile sensors).

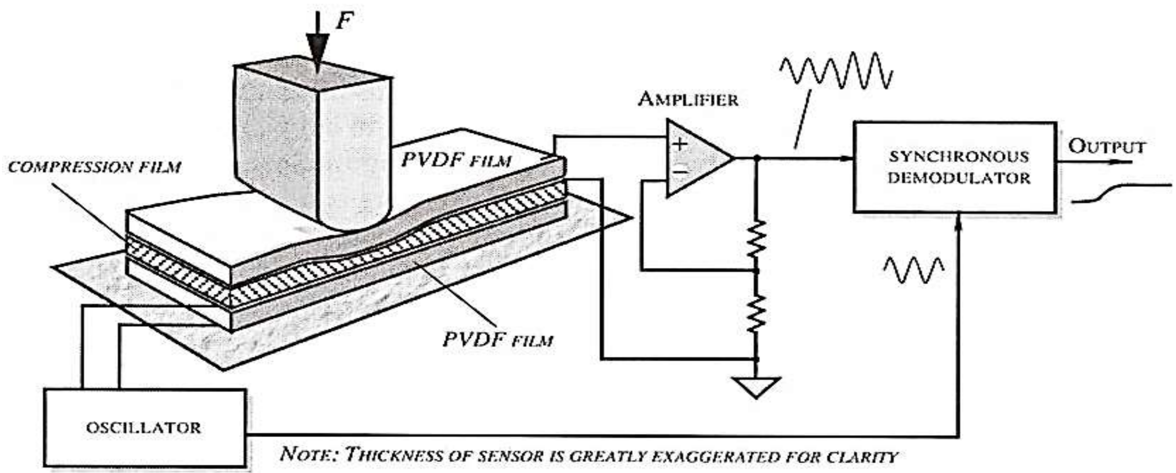

Fig 3.7 Tactile Sensor

Tactile sensors are used in robotics, computer hardware and security systems. A common application of tactile sensors is in touch screen devices on mobile phones and computing.

Tactile sensors may be of different types including piezoresistive, piezoelectric, capacitive and elastoresistive sensors.

3.4.3 Liquid Crystal Display (LCD)

liquid-crystal display (LCD) is a flat-panel display or other electronically modulated optical device that uses the light-modulating properties of liquid crystals. Liquid crystals do not emit light directly, instead using a backlight or reflector to produce images in color or monochrome. LCDs are available to display arbitrary images (as in a general-purpose computer display).

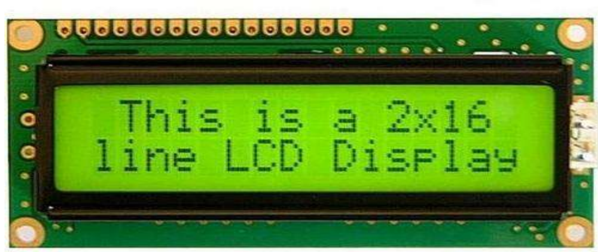

Fig. 3.8 16x2 LCD display

It can fixed images with low information content, which can be displayed or hidden, such as preset words, digits, and seven-segment displays, as in a digital clock. They use the same basic technology, except that arbitrary images are made up of a large number of small pixels, while other displays have larger elements.

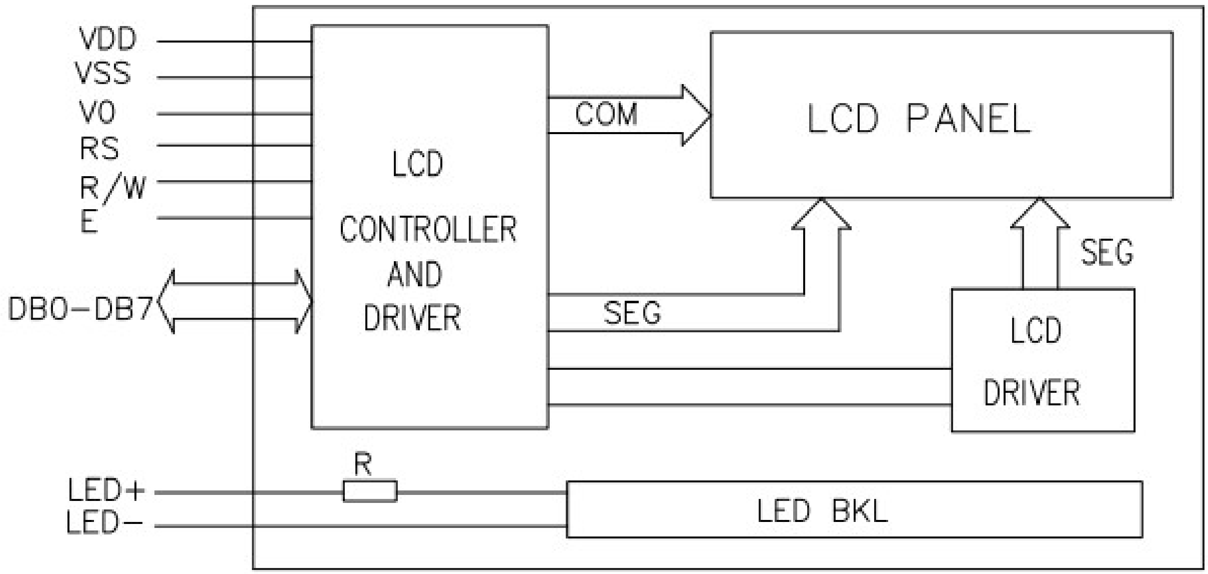

Fig. 3.9 16x2 LCD block diagram

Features:

- 5x8 dots with cursor
- 16 characters with *2 lines display
- 4-bit or 8-bit MPU interfaces
- Built in controller
- Display mode & Backlight variations

Table 3.2 LCD pin description

Pin no.	Symbol	External connection	Function
1	Vss	Power supply	Signal ground for LCM
2	V_{DD}		Power supply for logic for LCM
3	V_0		Contrast adjust
4	RS	MPU	Register select signal
5	R/W	MPU	Read/write select signal
6	E	MPU	Operation (data read/write) enable signal
7~10	DB0~DB3	MPU	Four low order bi-directional three-state data bus lines. Used for data transfer between the MPU and the LCM. These four are not used during 4-bit operation.
11~14	DB4~DB7	MPU	Four high order bi-directional three-state data bus lines. Used for data transfer between the MPU

VSS, VDD and VEE

Pin 1 (VSS) is a ground pin and it is certainly needed that this pin should be grounded for LCD to work properly. VEE and VDD are given +5 volts normally. However VEE may have a potentiometer voltage divider network to get the contrast adjusted. But VDD is always at +5V.

RS, R/W and E

These three pins are numbered 4, 5 and 6 as shown above. RS is used to make the selection between data and command register. For RS=0, command registers selected and for RS=1 data register is selected.

R/W gives you the choice between writing and reading. If set (R/W=1) reading is enabled. R/W=0 when writing.

Enable pins is used by the LCD to latch information presented to its data pins. When data is supplied to data pins, a high to low pulse must be applied to this pin in-order for the LCD to latch in the data present at the data pins. It may be noted here that the pulse must be of minimum 450ns wide.

D0-D7

The 8-bit data pins, D0-D7, are used to send information to the LCD or read the contents of LCD's internal register.

3.4.4 Piezoelectric buzzer

Piezoelectric buzzers, or piezo buzzers, as they are sometimes called, were invented by Japanese manufacturers and fitted into a wide array of products during the 1970s to 1980s. This advancement mainly came about because of cooperative efforts by Japanese manufacturing companies. In 1951, they established the Barium Titanate Application Research Committee, which allowed the companies to be "competitively cooperative" and bring about several piezoelectric innovations and inventions.

Fig 3.10 Piezoelectric buzzer

Features:

- Wire lead with feedback
- 12v DC rating
- Low profile
- 4.5 kHz rated frequency

A piezoelectric element may be driven by an oscillating electronic circuit or other audio signal source, driven with a piezoelectric audio amplifier. Sounds commonly used to indicate that a button has been pressed are a click, a ring or a beep. A piezoelectric buzzer/beeper also depends on acoustic cavity resonance or Helmholtz resonance to produce an audible beep

3.4.5 Fuel valve (Solenoid valve)

A solenoid valve is an electromechanically operated valve. The valve is controlled by an electric current through a solenoid: in the case of a two-port valve the flow is switched on or off; in the case of a three-port valve, the outflow is switched between the two outlet ports. Multiple solenoid valves can be placed together on a manifold.

Fig 3.11 Fuel valve (Solenoid valve)

Solenoid valves are the most frequently used control elements in fluidics. Their tasks are to shut off, release, dose, distribute or mix fluids. They are found in many application areas. Solenoids offer fast and safe switching, high reliability, long service life, good medium compatibility of the materials used, low control power and compact design.

3.4.6 DC motor

A DC motor is any of a class of rotary electrical machines that converts direct current electrical energy into mechanical energy. The most common types rely on the forces produced by magnetic fields. Nearly all types of

DC motors have some internal mechanism, either electromechanical or electronic, to periodically change the direction of current flow in part of the motor.

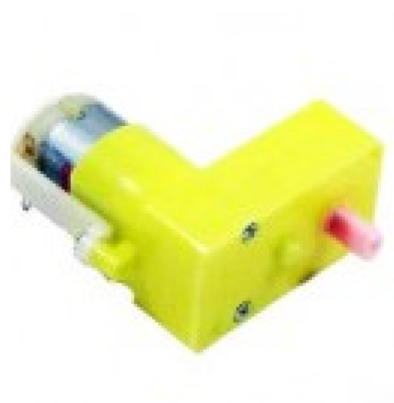

Fig. 3.12 DC motor

In this project we are using this to create a door mechanism for controlling the air flow of the engine.

3.4.7 Alcohol gas sensor (MQ-3)

The MQ3 gas sensor is alcohol sensor which is used to detect the alcohol concentration on your breath. This sensor provides an analog resistive output based on alcohol concentration. When the alcohol gas exist, the sensor's conductivity gets higher along with the gas concentration rising.

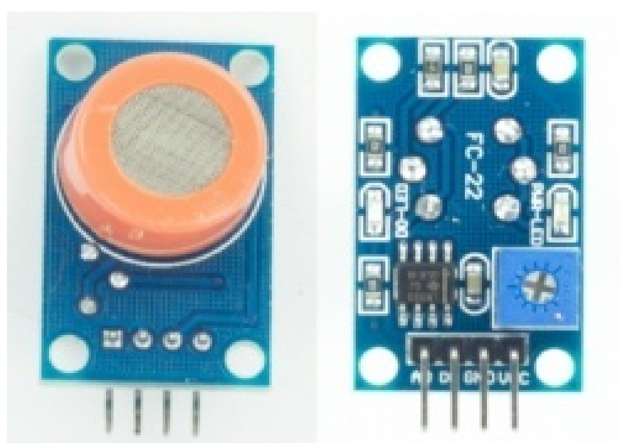

Fig.3.13 Alcohol gas sensor (MQ-3)

Features:

- High sensitivity to alcohol and small sensitivity to Benzine.
- Fast response and high sensitivity.
- Stable and long life.
- Simple drive circuit.

•

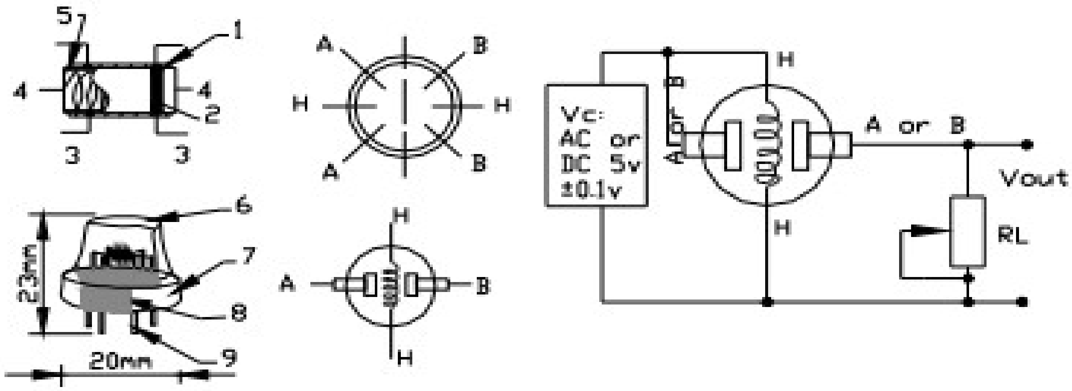

Fig. 3.14 design of alcohol sensor

Specifications:

Table 3.3 standard work condition

Symbol	Parameter name	Technical condition	Remarks
Vc	Circuit voltage	5V±0.1	AC OR DC
V_H	Heating voltage	5V±0.1	ACOR DC
R_L	Load resistance	200KΩ	
R_H	Heater resistance	33Ω±5%	Room Tem
P_H	Heating consumption	less than 750mw	

Table 3.4 environment condition

Symbol	Parameter name	Technical condition	Remarks
Tao	Using Tem	-10℃-50℃	
Tas	Storage Tem	-20℃-70℃	
R_H	Related humidity	less than 95%Rh	
O_2	Oxygen concentration	21%(standard condition)Oxygen concentration can affect sensitivity	minimum value is over 2%

Table 3.5 sensitivity characteristic

Symbol	Parameter name	Technical parameter	Remarks
Rs	Sensing Resistance	1MΩ - 8 MΩ (0.4mg/L alcohol)	Detecting concentration scope: 0.05mg/L—10mg/L Alcohol
α (0.4/1 mg/L)	Concentration slope rate	≤0.6	
Standard detecting condition	Temp: 20℃±2℃ Vc:5V±0.1 Humidity: 65%±5% Vh: 5V±0.1		
Preheat time	Over 24 hour		

Sensitive material of MQ-3 gas sensor is SnO2, which with lower conductivity in clean air. When the target alcohol gas exist, the sensor's conductivity is higher along with the gas concentration rising. MQ-3 gas sensor has high sensitivity to Alcohol, and has good resistance to disturb of gasoline, smoke and vapor.

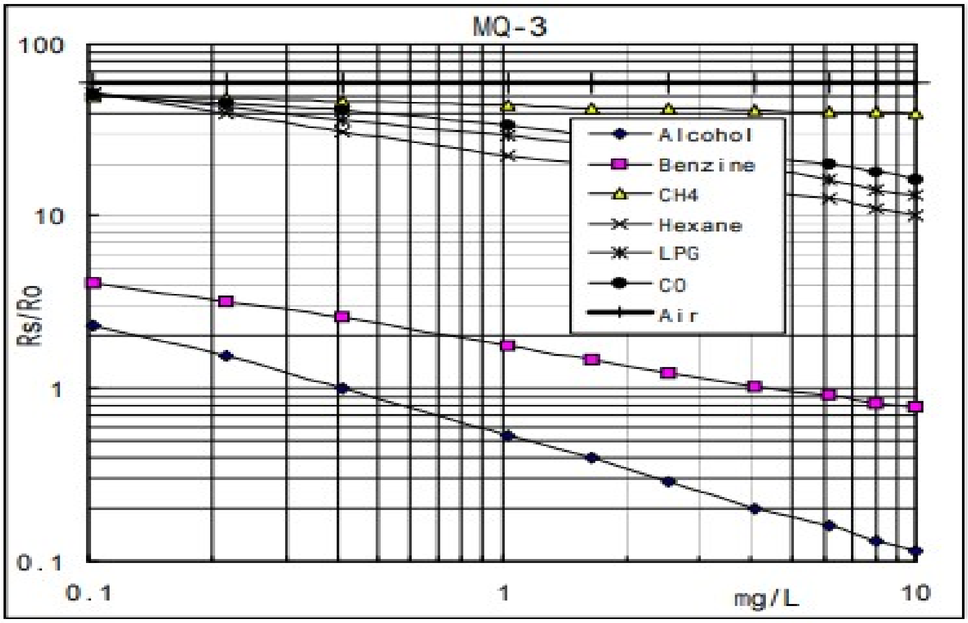

Fig. 3.15 sensitivity characteristics of MQ-3

The sensor could be used to detect alcohol with different concentration; it is with low cost and suitable for different application. It is widely used in domestic alcohol gas alarm, industrial alcohol gas alarm and portable alcohol detector

3.4.8 Relay

A **relay** is an electrically operated switch. Many relays use an electromagnet to mechanically operate a switch, but other operating principles are also used, such as solid-state relays. Relays are used where it is necessary to control a circuit by a separate low-power signal, or where several circuits must be controlled by one signal. The first relays were used in long distance telegraph circuits as amplifiers: they repeated the signal coming in from one circuit and re-transmitted it on another circuit. Relays were used extensively in telephone exchanges and early computers to perform logical operations.

Fig.3.16 Relay

A type of relay that can handle the high power required to directly control an electric motor or other loads is called a contactor. Solid-state relays control power circuits with no moving parts, instead using a semiconductor device to perform switching. Relays with calibrated operating characteristics and sometimes multiple operating coils are used to protect electrical circuits from overload or faults; in modern electric power systems these functions are performed by digital instruments still called "protective relays".

Magnetic latching relays require one pulse of coil power to move their contacts in one direction, and another, redirected pulse to move them back. Repeated pulses from the same input have no effect. Magnetic latching relays are useful in applications where interrupted power should not be able to transition the contacts.

Magnetic latching relays can have either single or dual coils. On a single coil device, the relay will operate in one direction when power is applied with one polarity, and will reset when the polarity is reversed. On a dual coil device, when polarized voltage is applied to the reset coil the contacts will transition. AC controlled magnetic latch relays have single coils that employ steering diodes to differentiate between operate and reset commands

3.4.9 Battery and variable voltage AC to DC adaptor

Battery

An electric **battery** is a device consisting of one or more electrochemical cells with external connections provided to power electrical devices such as flashlights, smart phones, and electric cars. When a battery is supplying electric power, its positive terminal is the cathode and its negative terminal is the anode.[2] The terminal marked negative is the source of electrons that when connected to an external circuit will flow and deliver energy to an external device.

Fig. 3.17 9v Battery

When a battery is connected to an external circuit, electrolytes are able to move as ions within, allowing the chemical reactions to be completed at the separate terminals and so deliver energy to the external circuit. It is the movement of those ions within the battery which allows current to flow out of the battery to perform work.[3]Historically the term "battery" specifically referred to a device composed of multiple cells, however the usage has evolved additionally to include devices composed of a single cell

Variable voltage AC to DC adaptor

DC power supplies use AC mains electricity as an energy source. Such power supplies will employ a transformer to convert the input voltage to a higher or lower AC voltage. A rectifier is used to convert the transformer output voltage to a varying DC voltage, which in turn is passed through an electronic filter to convert it to an unregulated DC voltage.

Fig. 3.18 Variable voltage AC to DC adaptor

The filter removes most, but not all of the AC voltage variations; the remaining AC voltage is known as *ripple*. The electric load's tolerance of ripple dictates the minimum amount of filtering that must be provided by a power supply. In some applications, high ripple is tolerated and therefore no filtering is required. For example, in some battery charging applications it is possible to implement a mains-powered DC power supply with nothing more than a transformer and a single rectifier diode, with a resistor in series with the output to limit charging current.

3.4.10 Engine

An **engine** or **motor** is a machine designed to convert one form of energy into mechanical energy. Heat engines burn a fuel to create heat which is then used to do work. Electric motors convert electrical energy into mechanical motion; pneumatic motors use compressed air; and clockwork motors in wind-up toys use elastic energy. In biological systems, molecular motors, like myosin in muscles, use chemical energy to create forces and eventually motion.

Fig. 3.19 generator engine

An **engine-generator** or **portable generator** is the combination of an electrical generator and an engine (prime mover) mounted together to form a single piece of equipment. This combination is also called an *engine-generator set* or a *gen-set*. In many contexts, the engine is taken for granted and the combined unit is simply called a *generator*

In this project we are using a engine-generator of 600kW of power, 400-500V of voltage and 50Hz of frequency.

3.5 SOFTWARE USED

3.5.1 Arduino IDE

The Arduino integrated development environment (IDE) is a cross-platform application (for Windows, macOS, Linux) that is written in the programming language Java. It originated from the IDE for the languages *Processing* and *Wiring*. It includes a code editor with features such as text cutting and pasting, searching and replacing text, automatic indenting, brace matching, and syntax highlighting, and provides simple *one-click* mechanisms to compile and upload programs to an Arduino board. It also contains a message area, a text console, a toolbar with buttons for common functions and a hierarchy of operation menus. The source code for the IDE is released under the GNU General Public License, version 2

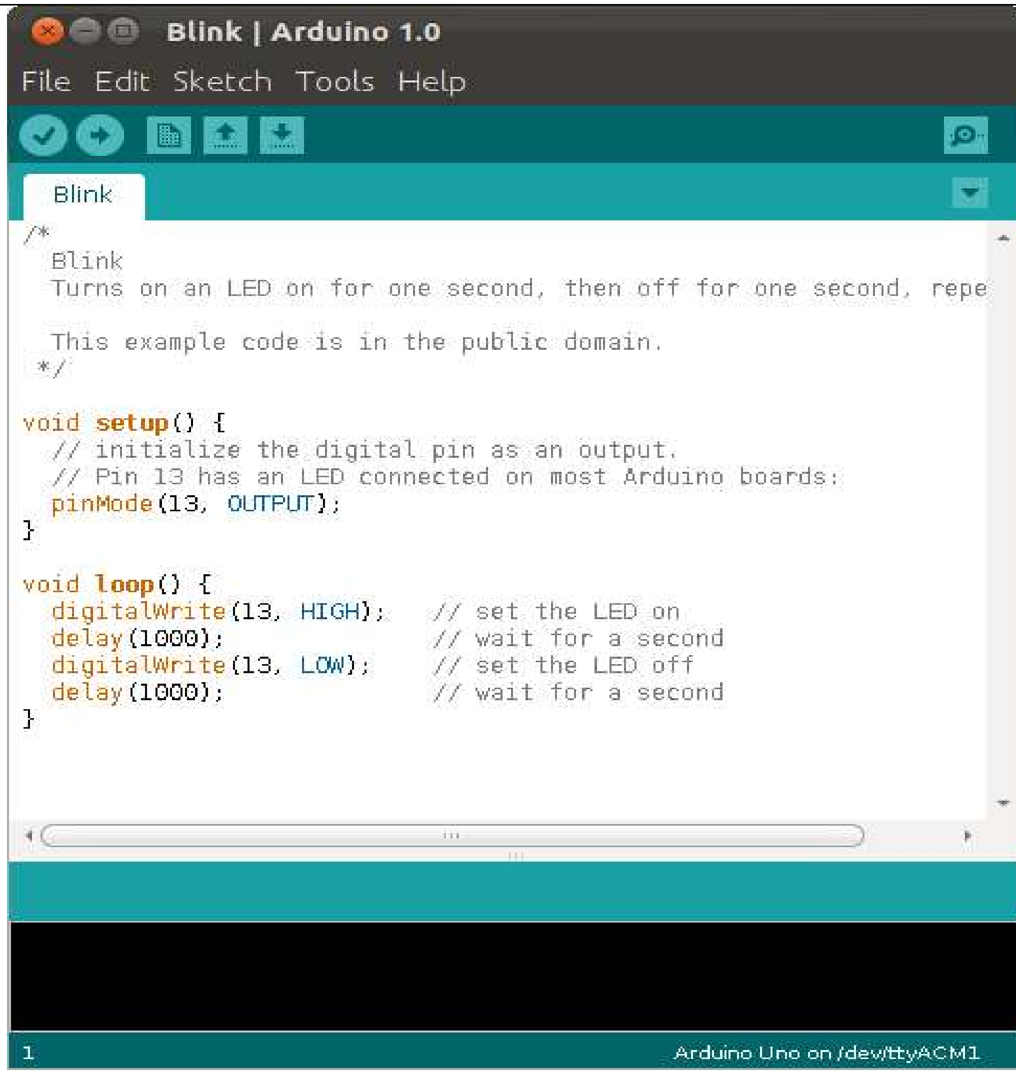

Fig.3.20 Arduino IDE

The Arduino IDE supports the languages C and C++ using special rules of code structuring. The Arduino IDE supplies a software library from the Wiring project, which provides many common input and output procedures. User-written code only requires two basic functions, for starting the sketch and the main program loop, that are compiled and linked with a program stub *main()* into an executable cyclic executive program with the GNU toolchain, also included with the IDE distribution. The Arduino IDE employs the program *avrdude* to convert the executable code into a text file in hexadecimal encoding that is loaded into the Arduino board by a loader program in the board's firmware.

3.5.2 PROTEUS

The **Proteus Design Suite** is a proprietary software tool suite used primarily for electronic design automation. The software is used mainly by electronic design engineers and technicians to create schematics and electronic prints for manufacturing printed circuit boards.

It was developed in Yorkshire, England by Labcenter Electronics Ltd and is available in English, French, Spanish and Chinese languages

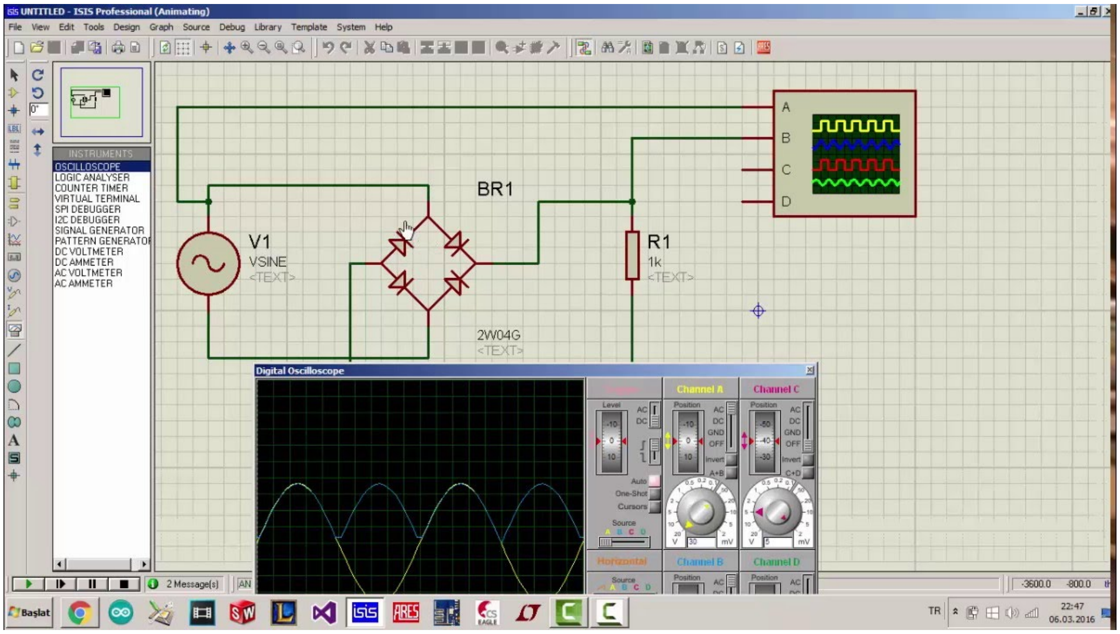

Fig. 3.21 proteus simulator

The Proteus Design Suite is a Windows application for schematic capture, simulation, and PCB layout design. It can be purchased in many configurations, depending on the size of designs being produced and the requirements for microcontroller simulation. All PCB Design products include an autorouter and basic mixed mode SPICE simulation capabilities.

CHAPTER 4

DESIGN METHODOLOGY

4.1 CICUIT DIAGRAM WITH SIGNAL FLOW

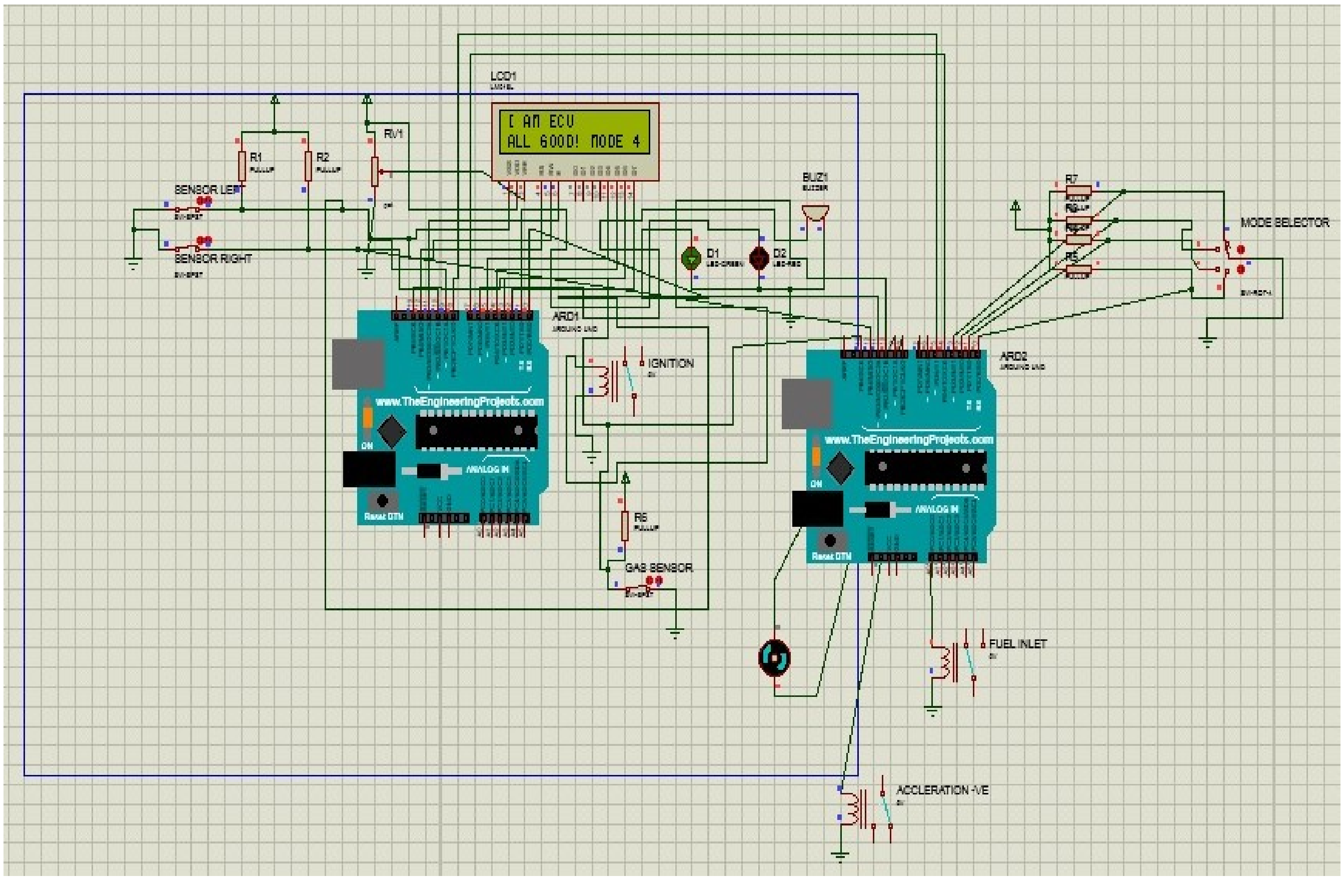

Fig. 4.1 Circuit diagram of project

As given in Fig. 4.1, the circuit diagram comprises of tactile sensor, buzzer, LCD, LED, arduino UNO, relays, alcohol sensor (MQ-3), fuel valve (solenoid valve) and engine. The explanation in detail is given as below:

- First the alcohol sensor will check the driver's state i.e. whether the person is drunk or not. If the driver is drunk, our system won't allow a driver to handle the steering wheel and if the driver is not drunk it will allow the driver to allow the steering wheel.
- Then the mode of the driving will be selected as cruze mode (plain road), hill mode (slant road) and the off road (unbalanced road) with the switch. These modes will determine response time of the contact between the hand and the steering wheel as given in Fig.4.2.

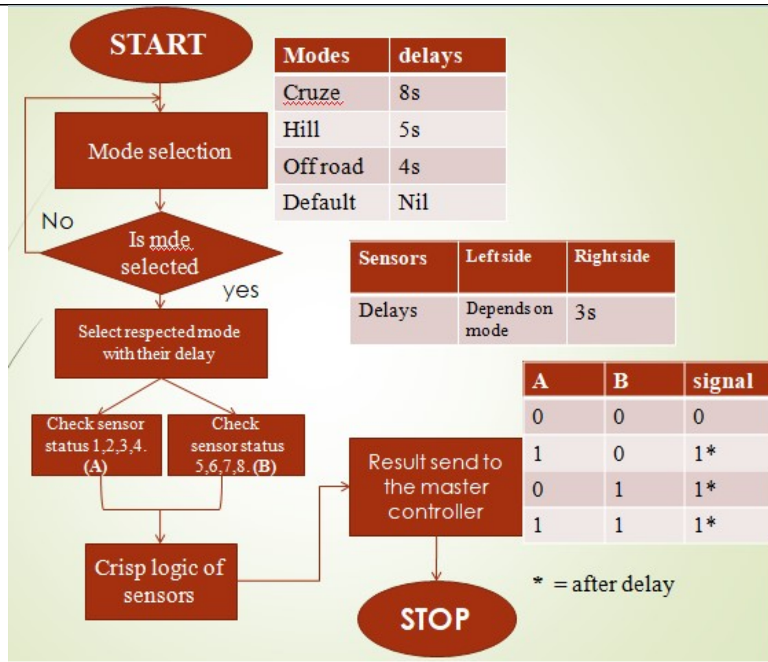

Fig 4.2 Flow chart project working

➢ When the mode is selected, the response will be determined for the sensors and with this information the controller will send the signals to the parameter of engine (air flow, fuel valve and choke) to decrease the engine speed or to phase out the engine speed.

4.2 SOFTWARE USE

4.2.1 DETAIL

What is arduino technology?

Arduino is an open-source electronics platform based on easy-to-use hardware and software.Arduino boards are able to read inputs - light on a sensor, a finger on a button, or a Twitter message - and turn it into an output - activating a motor, turning on an LED, publishing something online. You can tell your board what to do by sending a set of instructions to the microcontroller on the board. To do so you use the Arduino programming language (based on Wiring), and the Arduino Software (IDE), based on Processing.

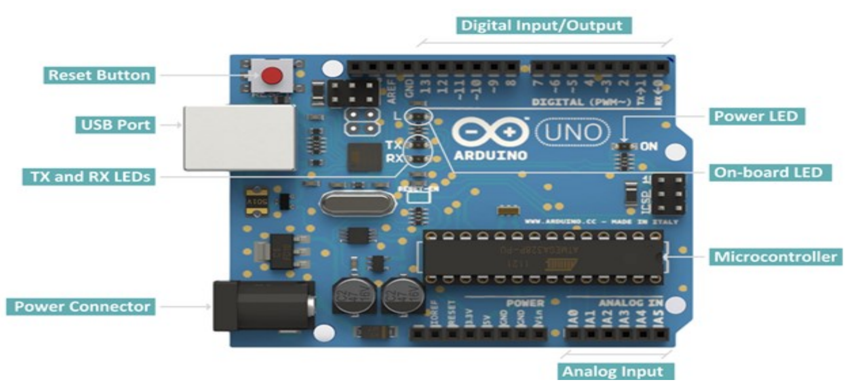

Fig. 4.3 Arduino UNO board

Over the years Arduino has been the brain of thousands of projects, from everyday objects to complex scientific instruments. A worldwide community of makers - students, hobbyists, artists, programmers, and professionals - has gathered around this open-source platform, their contributions have added up to an incredible amount of accessible knowledge that can be of great help to novices and experts alike.

Arduino was born at the Ivrea Interaction Design Institute as an easy tool for fast prototyping, aimed at students without a background in electronics and programming. As soon as it reached a wider community, the Arduino board started changing to adapt to new needs and challenges, differentiating its offer from simple 8-bit boards to products for IoT applications, wearable, 3D printing, and embedded environments. All Arduino boards are completely open-source, empowering users to build them independently and eventually adapt them to their particular needs. The software, too, is open-source, and it is growing through the contributions of users worldwide.

4.2.2 Arduino 1 (Master controller)

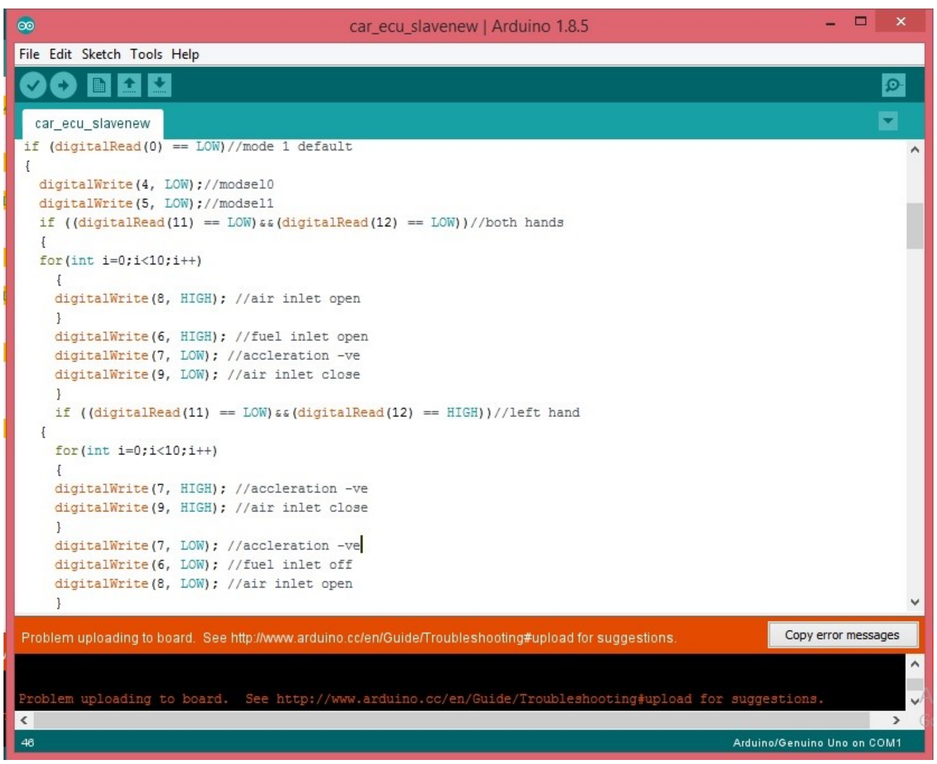

Fig. 4.4 Master controller program

```
#include <LiquidCrystal.h>
const int rs = 12, en = 11, d4 = 5, d5 = 4, d6 = 3, d7 = 2;
LiquidCrystal lcd(rs, en, d4, d5, d6, d7);
void setup() {
  // setup code here, to run once:
```

```
 lcd.begin(10, 2);
 lcd.print("I AM ECU      ");
 pinMode(0, OUTPUT);            //green led
 pinMode(1, OUTPUT);            //red led
 pinMode(6, OUTPUT);            //buzzer
 pinMode(7, INPUT);             //mode selector 0
 pinMode(8, INPUT);             //mode selector 1
 pinMode(9, INPUT);             //left sensor
 pinMode(10, INPUT);            //right sensor
 pinMode(13, INPUT);            //alcohol detector
}

void loop() {
  // main code here, to run repeatedly:
 if (digitalRead(13) == LOW)    //no alcohol
 {
 if ((digitalRead(9) == LOW)&&(digitalRead(10) == LOW))         // both hands
  {
   digitalWrite(1, LOW);           //red
   digitalWrite(0, HIGH);          //green
   lcd.setCursor(0, 1);
   lcd.print("ALL GOOD!    ");
  if ((digitalRead(7) == LOW)&&(digitalRead(8) == LOW))
  {
   lcd.setCursor(10, 2);
   lcd.print("MODE 1        ");
   }
   else if ((digitalRead(7) == LOW)&&(digitalRead(8) == HIGH))
  {
   lcd.setCursor(10, 2);
   lcd.print("MODE 2        ");
   }
   else if ((digitalRead(7) == HIGH)&&(digitalRead(8) == LOW))
```

```
 {
 lcd.setCursor(10, 2);
 lcd.print("MODE 3        ");
 }
 else if ((digitalRead(7) == HIGH)&&(digitalRead(8) == HIGH))
 {
 lcd.setCursor(10, 2);
 lcd.print("MODE 4        ");
 }
}
else if ((digitalRead(9) == LOW)&&(digitalRead(10) == HIGH))        //left only
{
 digitalWrite(1, HIGH);        //red
 digitalWrite(0, LOW);         //green
 lcd.setCursor(0, 1);
 lcd.print("HOLD RIGHT    ");
 if ((digitalRead(7) == LOW)&&(digitalRead(8) == LOW))
 {
 lcd.setCursor(10, 2);
 lcd.print("MODE 1        ");
 }
 else if ((digitalRead(7) == LOW)&&(digitalRead(8) == HIGH))
 {
 lcd.setCursor(10, 2);
 lcd.print("MODE 2        ");
 }
 else if ((digitalRead(7) == HIGH)&&(digitalRead(8) == LOW))
 {
 lcd.setCursor(10, 2);
 lcd.print("MODE 3        ");
 }
 else if ((digitalRead(7) == HIGH)&&(digitalRead(8) == HIGH))
 {
```

```
 lcd.setCursor(10, 2);
 lcd.print("MODE 4        ");
 }
}
else if ((digitalRead(9) == HIGH)&&(digitalRead(10) == LOW))             //right only
{
 digitalWrite(1, HIGH);                  //red
 digitalWrite(0, LOW);                   //green
 lcd.setCursor(0, 1);
 lcd.print("HOLD LEFT    ");
 if ((digitalRead(7) == LOW)&&(digitalRead(8) == LOW))
 {
 lcd.setCursor(10, 2);
 lcd.print("MODE 1        ");
 }
 else if ((digitalRead(7) == LOW)&&(digitalRead(8) == HIGH))
 {
 lcd.setCursor(10, 2);
 lcd.print("MODE 2        ");
 }
 else if ((digitalRead(7) == HIGH)&&(digitalRead(8) == LOW))
 {
 lcd.setCursor(10, 2);
 lcd.print("MODE 3        ");
 }
 else if ((digitalRead(7) == HIGH)&&(digitalRead(8) == HIGH))
 {
 lcd.setCursor(10, 2);
 lcd.print("MODE 4        ");
 }
}
else                                                      //no hand
{
```

```
    digitalWrite(1, HIGH);                  //red
    digitalWrite(0, LOW);                   //green
    digitalWrite(6, HIGH);
    lcd.setCursor(0, 1);
    lcd.print("HOLD IT      ");
    if ((digitalRead(7) == LOW)&&(digitalRead(8) == LOW))
    {
    lcd.setCursor(10, 2);
    lcd.print("MODE 1         ");
    }
    else if ((digitalRead(7) == LOW)&&(digitalRead(8) == HIGH))
    {
    lcd.setCursor(10, 2);
    lcd.print("MODE 2         ");
    }
    else if ((digitalRead(7) == HIGH)&&(digitalRead(8) == LOW))
    {
    lcd.setCursor(10, 2);
    lcd.print("MODE 3         ");
    }
    else if ((digitalRead(7) == HIGH)&&(digitalRead(8) == HIGH))
    {
    lcd.setCursor(10, 2);
    lcd.print("MODE 4         ");
    }
   }
  }
  else if (digitalRead(13) == HIGH)                  //alcohol detected
  {
   lcd.setCursor(0, 1);
   lcd.print("YOU ARE DRUNK");
   digitalWrite(1, HIGH);                            //red
   digitalWrite(0, LOW);                             //green
```

```
lcd.begin(10, 2);
lcd.print("I AM ECU       ");
}
}
```

4.2.3 Arduino 2 (Slave controller)

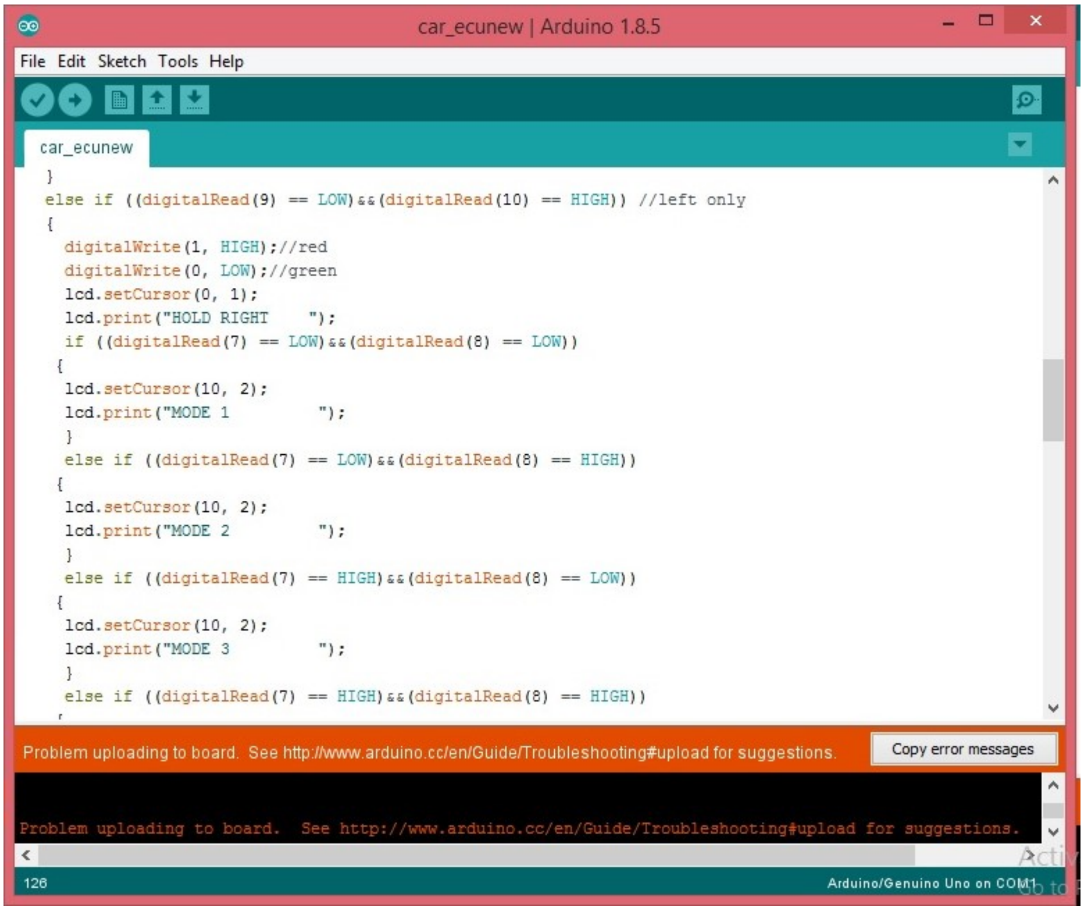

Fig. 4.4 Slave controller program

```
void setup() {
  // setup code here, to run once:
pinMode(0, INPUT);                  //mode 1 default
pinMode(1, INPUT);                  //mode 2 cruse
pinMode(2, INPUT);                  //mode 3 hill
pinMode(3, INPUT);                  //mode 4 offroad
pinMode(6, OUTPUT);                 //fuel inlet
pinMode(7, OUTPUT);                 //accleration -ve
pinMode(8, OUTPUT);                 //air inlet open(motor cw)
pinMode(9, OUTPUT);                 //air inlet close(motor ccw)
```

```
pinMode(10, OUTPUT);                    //ignition
pinMode(4, OUTPUT);                     //mode selector 0
pinMode(5, OUTPUT);                     //mode selector 1
pinMode(11, INPUT);                     //left sensor
pinMode(12, INPUT);                     //right sensor
pinMode(13, INPUT);                     //alcohol detector

}

void loop() {
                                        // main code here, to run repeatedly:
 if (digitalRead(13) == LOW)//no alcohol
 {
  digitalWrite(10, HIGH);               //ignition off
if (digitalRead(0) == LOW)              //mode 1 default
{
  digitalWrite(4, LOW);                 //modsel0
  digitalWrite(5, LOW);                 //modsel1
  if ((digitalRead(11) == LOW)&&(digitalRead(12) == LOW))       //both hands
  {
  for(int i=0;i<10;i++)
   {
   digitalWrite(8, HIGH);               //air inlet open
   }
   digitalWrite(6, HIGH);               //fuel inlet open
   digitalWrite(7, LOW);                //accleration -ve
   digitalWrite(9, LOW);                //air inlet close
   }
   if ((digitalRead(11) == LOW)&&(digitalRead(12) == HIGH))           //left hand
  {
   for(int i=0;i<10;i++)
   {
   digitalWrite(7, HIGH);               //accleration -ve
```

```
    digitalWrite(9, HIGH);                      //air inlet close
    }
    digitalWrite(7, LOW);                       //accleration -ve
    digitalWrite(6, LOW);                        //fuel inlet off
    digitalWrite(8, LOW);                        //air inlet open
    }
    if ((digitalRead(11) == HIGH)&&(digitalRead(12) == LOW))            //right hand
   {
    for(int i=0;i<10;i++)
    {
    digitalWrite(7, HIGH);                      //accleration -ve
    digitalWrite(9, HIGH);                      //air inlet close
    }
    digitalWrite(7, LOW);                       //accleration -ve
    digitalWrite(9, LOW);                       //air inlet close
    digitalWrite(6, LOW);                       //fuel inlet off
    digitalWrite(8, LOW);                       //air inlet open
    }
    if ((digitalRead(11) == HIGH)&&(digitalRead(12) == HIGH))           //no hand
   {
    for(int i=0;i<10;i++)
    {
    digitalWrite(7, HIGH);                      //accleration -ve
    digitalWrite(9, HIGH);                       //air inlet close
    }
    digitalWrite(7, LOW);                       //accleration -ve
    digitalWrite(9, LOW);                       //air inlet close
    digitalWrite(6, LOW);                       //fuel inlet off
    digitalWrite(8, LOW);                        //air inlet open
    }
  }
  else if (digitalRead(1) == LOW)               //mode 2 cruze
  {
```

```
digitalWrite(4, LOW);                    //modsel0
digitalWrite(5, HIGH);                   //modsel1
if ((digitalRead(11) == LOW)&&(digitalRead(12) == LOW))          //both hands
{delay(30);
 for(int i=0;i<10;i++)
 {
 digitalWrite(8, HIGH);                  //air inlet open
 }
 digitalWrite(6, HIGH);                  //fuel inlet open
 digitalWrite(7, LOW);                    //accleration -ve
 digitalWrite(9, LOW);                   //air inlet close
 }
 if ((digitalRead(11) == LOW)&&(digitalRead(12) == HIGH))        //left hand
{delay(80);
 for(int i=0;i<10;i++)
 {
 digitalWrite(7, HIGH);                  //accleration -ve
 digitalWrite(9, HIGH);                  //air inlet close
 }
 digitalWrite(7, LOW);                   //accleration -ve
 digitalWrite(6, LOW);                   //fuel inlet off
 digitalWrite(8, LOW);                   //air inlet open
 }
 if ((digitalRead(11) == HIGH)&&(digitalRead(12) == LOW))        //right hand
{delay(30);
 for(int i=0;i<10;i++)
 {
 digitalWrite(7, HIGH);                  //accleration -ve
 digitalWrite(9, HIGH);                  //air inlet close
 }
 digitalWrite(7, LOW);                    //accleration -ve
 digitalWrite(9, LOW);                   //air inlet close
 digitalWrite(6, LOW);                   //fuel inlet off
```

```
  digitalWrite(8, LOW);                    //air inlet open
  }
  if ((digitalRead(11) == HIGH)&&(digitalRead(12) == HIGH))          //no hand
 {
  for(int i=0;i<10;i++)
  {
  digitalWrite(7, HIGH);                   //accleration -ve
  digitalWrite(9, HIGH);                   //air inlet close
  }
  digitalWrite(7, LOW);                    //accleration -ve
  digitalWrite(9, LOW);                    //air inlet close
  digitalWrite(6, LOW);                    //fuel inlet off
  digitalWrite(8, LOW);                    //air inlet open
  }
}
else if (digitalRead(2) == LOW)            //mode 3 hill
{
 digitalWrite(4, HIGH);                    //modsel0
 digitalWrite(5, LOW);                     //modsel1
 if ((digitalRead(11) == LOW)&&(digitalRead(12) == LOW))          //both hands
 {delay(30);
  for(int i=0;i<10;i++)
  {
  digitalWrite(8, HIGH);                   //air inlet open
  }
  digitalWrite(6, HIGH);                   //fuel inlet open
  digitalWrite(7, LOW);                    //accleration -ve
  digitalWrite(9, LOW);                    //air inlet close
  }
  if ((digitalRead(11) == LOW)&&(digitalRead(12) == HIGH))          //left hand
 {delay(50);
  for(int i=0;i<10;i++)
  {
```

```
  digitalWrite(7, HIGH);                    //accleration -ve
  digitalWrite(9, HIGH);                    //air inlet close
  }
  digitalWrite(7, LOW);                     //accleration -ve
  digitalWrite(6, LOW);                     //fuel inlet off
  digitalWrite(8, LOW);                     //air inlet open
  }
  if ((digitalRead(11) == HIGH)&&(digitalRead(12) == LOW))          //right hand
 {delay(20);
  for(int i=0;i<10;i++)
  {
  digitalWrite(7, HIGH);                    //accleration -ve
  digitalWrite(9, HIGH);                    //air inlet close
  }
  digitalWrite(7, LOW);                     //accleration -ve
  digitalWrite(9, LOW);                     //air inlet close
  digitalWrite(6, LOW);                     //fuel inlet off
  digitalWrite(8, LOW);                     //air inlet open
  }
  if ((digitalRead(11) == HIGH)&&(digitalRead(12) == HIGH))         //no hand
 {
  for(int i=0;i<10;i++)
  {
  digitalWrite(7, HIGH);                    //accleration -ve
  digitalWrite(9, HIGH);                    //air inlet close
  }
  digitalWrite(7, LOW);                     //accleration -ve
  digitalWrite(9, LOW);                     //air inlet close
  digitalWrite(6, LOW);                     //fuel inlet off
  digitalWrite(8, LOW);                     //air inlet open
  }
}
else
```

```
{
 digitalWrite(4, HIGH);                    //modsel0
 digitalWrite(5, HIGH);                    //modsel1
 if ((digitalRead(11) == LOW)&&(digitalRead(12) == LOW))          //both hands
 {delay(30);
  for(int i=0;i<10;i++)
  {
  digitalWrite(8, HIGH);                   //air inlet open
  }
  digitalWrite(6, HIGH);                   //fuel inlet open
  digitalWrite(7, LOW);                    //accleration -ve
  digitalWrite(9, LOW);                    //air inlet close
  }
  if ((digitalRead(11) == LOW)&&(digitalRead(12) == HIGH))        //left hand
 {delay(40);
  for(int i=0;i<10;i++)
  {
  digitalWrite(7, HIGH);                   //accleration -ve
  digitalWrite(9, HIGH);                   //air inlet close
  }
  digitalWrite(7, LOW);                    //accleration -ve
  digitalWrite(6, LOW);                    //fuel inlet off
  digitalWrite(8, LOW);                    //air inlet open
  }
  if ((digitalRead(11) == HIGH)&&(digitalRead(12) == LOW))        //right hand
 {delay(20);
  for(int i=0;i<10;i++)
  {
  digitalWrite(7, HIGH);                   //accleration -ve
  digitalWrite(9, HIGH);                   //air inlet close
  }
  digitalWrite(7, LOW);                    //accleration -ve
  digitalWrite(9, LOW);                    //air inlet close
```

```
    digitalWrite(6, LOW);                   //fuel inlet off
    digitalWrite(8, LOW);                   //air inlet open
    }
    if ((digitalRead(11) == HIGH)&&(digitalRead(12) == HIGH))         //no hand
   {
    for(int i=0;i<10;i++)
    {
    digitalWrite(7, HIGH);                   //accleration -ve
    digitalWrite(9, HIGH);                  //air inlet close
    }
    digitalWrite(7, LOW);                   //accleration -ve
    digitalWrite(9, LOW);                    //air inlet close
    digitalWrite(6, LOW);                    //fuel inlet off
    digitalWrite(8, LOW);                   //air inlet open
    }
  }
  }
  else
  {
   digitalWrite(10, LOW);                   //ignition off
   }
  }
```

4.3 RESULT AND DISCUSSION

4.3.1 Result

Following are the result of the project we have been working on:

- The project started with the idea of reducing the number of road accidents. It aimed at reducing the rod accidents causing due to careless driving.
- The project started with the study of various components.
- The stoppage of the engine prevents the mismanagement of the steering wheel thus preventing the accidents.

- The stoppage of the engine when the alcohol is detected by the alcohol sensor also prevents the accidents due to drinking.
- We are supposed to make a real time project.

4.3.2 Discussions

Following are the discussions of the project have been working on:

- Due to shortfall of time, we have made the prototype only.
- This is not 100% efficient system because suddenly stoppage of the engine can cause accident also

CHAPTER 5

APPLICATIONS, ADVANTAGES AND LIMITATIONS

5.1 APPLICATIONS

1. **Alcohol detection system:** For the driver's safety our system also has the alcohol sensor (MQ-3) to detect state of dive whether he/she is alcoholic or not. Thus it will avoid the drunken people
2. **Mobile usage:** With this system, the driver won't be allowed to pick the phone while driving because if the driver picks the phone, he/she will leave the steering and the car will stop.
3. **Four wheeler vehicle safety:** Our system is designed on the attentiveness of the driver; if the driver is not attentive the car will stop and saves both car and driver.
4. **Motor vehicle safety: S**ome little changes in our system and we can also apply it for 2 wheelers vehicle.
5. **Information:** With the ability of monitoring driver's status the companies can gather information to develop some strong system.

5.2 ADVANTAGES

1. Controlling Engine: As the ECU controls every aspect of the engine, we don't have to add the extra hardware to implement the engine control (except the PIC microcontroller which is used to control the sensory part and giving signal commands to the ECU).

2. Safety: Our project is to ensure the safety of the driver from the accident caused by the distractions, leading to removal of hands from the steering wheel.

3. Tampering: Our project is quite difficult only because the parameters in the ECU arequite vast and complex and modifying one parameter could lead to the change in the efficiency of the system which is not desired. Thus, a normal person cannot do any kind of tampering with the system.

4. Features Addition: We are making this project to stop the accidents causes by distraction of driver with the method of CAN interfacing. But, if we wish to attach (somewhere in the future) some device regarding the accident prevention, it will be quite easy with just having some minor changes in microcontroller programming.

5. Cost Effective: The total installation (except ECU, because every modern vehicle has it) of our system will be very cheap (maybe around Rs. 1000/- to 2000/-) so it is very cost effective project for the noble purpose.

5.3 LIMITATIONS

1. Physiological Nature: As for distraction purpose we assume that the driver should be attentive by using two hands. But this system cannot determine whether the driver is fit for driving or not which is also the cause of distraction.

2. Alcohol Consumption: As we intend to design our project to prevent the accident caused by distraction so, this cannot detect if the driver is drunk or not.

3. Driving experience: This system will make the driver safe form accident but it may cause its driving experience not smooth enough. Although, we can overcome this problem by taking feedback from drivers, making the project work in real time by adapting to the changing environment.

4. Acceleration: When the system is activated, it will interrupt the acceleration process causing the car's acceleration to go down.

CHAPTER 6

FUTURE SCOPE AND CONCLUSION

6.1 FUTURE SCOPE

In future implementation of the project will handle other vehicle controlling parameters like engine temperature, alcohol consumption, drowsiness of driver, seat belt state, position of vehicle, speed of the vehicle, etc.

In Future more real datum or more parameters of vehicle like millage, tier pressure, fuel level in vehicle, parking information and expected distance travel that to be monitored and controlled by using different controllers and comparing those results with the designed system and efficiency of system is identified.

Those real time parameters are necessary for the crash analysis for the insurance agencies, crime cases etc. to investigate on accident cases further.

6.2 CONCLUSION

This project is one of the important sensor based project. The main unit of this project is the alcohol sensor and touch sensor. Microcontroller is the heart of this project. It is the CPU of the complete project. If the person inside the car has consumed the alcohol then the alcohol detection is done by the project. If the person mishandles the steering wheel then this is also detected by this project. Both sensors are interfaced by microcontroller. In other words the output of both of these sensors is given to the microcontroller. Microcontroller consists of an inbuilt ADC. The output of the alcohol sensor is analog in nature .It converts the output to digital as the microcontroller is a digital device, it requires digital input. Microcontroller gives high pulse to the buzzer circuit and the buzzer is turned on. Due to this, the ignition of the car is deactivated; this project is used as an accident avoider system as this system is used to avoid the road accidents which are increasing ay by day due t carelessness of driver and also because of alcohol consumption