A

Project Report

on

CAR BLACK BOX USING PIC MICROCONTROLLER

Submitted to

JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY ANANTAPUR, ANANTAPURAMU

in partial fulfillment of the requirements for the award of the Degree of

BACHELOR OF TECHNOLOGY

in

ELECTRONICS AND COMMUNICATION ENGINEERING

Submitted by

D. AMRUTHA	**(199E1A04K8)**
J. AMRUTHA	**(199E1A04L7)**
K. AZARUDDIN	**(199E1A04L8)**
R. GOURI SREE	**(199E1A04O5)**
K. VENKAT HARSHITH RAYAL	**(199E1A04M9)**

Under the Guidance of
S. SALMA, M.tech (Ph.D)
Assistant Professor

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING
SRI VENKATESWARA ENGINEERING COLLEGE
(Approved by AICTE, New Delhi & Affiliated to JNTUA, Ananthapuramu)
Karakambadi Road, TIRUPATI – 517507

2019-2023

SRI VENKATESWARA ENGINEERING COLLEGE

(Approved by AICTE, New Delhi & Affiliated to JNTUA, Ananthapuramu)
Karakambadi Road, TIRUPATI – 517507

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

CERTIFICATE

This is to certify that the project report entitled **"CAR BLACK BOX USING PIC MICROCONTROLLER"** *a bonafide record of the project work done and submitted by*

D. AMRUTHA	**(199E1A04K8)**
J. AMRUTHA	**(199E1A04L7)**
K. AZARUDDIN	**(199E1A04L8)**
R. GOURI SREE	**(199E1A04O5)**
K. VENKAT HARSHITH RAYAL	**(199E1A04M9)**

for the partial fulfillment of the requirements for the award of B. Tech Degree in **ELECTRONICS ANDCOMMUNICATION ENGINEERING**, *JNT University Anantapur, Ananthapuramu.*

GUIDE **Head of the Department**

External Viva-Voce Exam Held on _____

INTERNAL EXAMINER **EXTERNAL EXAMINER**

ACKNOWLEDGEMENT

We are thankful to our guide **S.Salma** for her valuable guidance and encouragement. Her helping attitude and suggestions have helped us in the successful completion of the project.

We would like to express our gratefulness and sincere thanks to **Dr.D.Srinivasulu Reddy,** Head of the Department of ELECTRONICS AND COMMUNICATION ENGINEERING, for his kind help and encouragement during the course of our study and in the successful completion of the project work.

We have great pleasure in expressing our hearty thanks to **Dr.C.Chandra Sekhar** our beloved Principal for his constant encouragement and support in all aspect to complete this project.

Successful completion of any project cannot be done without proper support and encouragement. We sincerely thanks to the **Management** for providing all the necessary facilities during the course of study.

We would like to thank our parents and friends, who have the greatest contributions in all our achievements, for the great care and blessings in making us successful in all our endeavors.

D. AMRUTHA (199E1A04K8)
J. AMRUTHA (199E1A04L7)
K. AZARUDDIN (199E1A04L8)
R. GOURI SREE (199E1A04O5)
K. VENKAT HARSHITH RAYAL (199E1A04M9)

TABLE OF CONTENTS

ABSTRACT

Millions of people die due to the accidents. The car black box is used to analyse the cause of accidents like an airplane black box. This paper proposes a model of a car black box system which can be installed in the cars. The aim of this paper is to achieve accident analysis the working process of vehicles. The black box system also uses GPRS/GSM sensor to collect the data. The car black box system mainly helps the insurance companies to do car crash investigations and to record the road status to prevent or decrease death rates. This paper proposes a technique to monitor the vehicle performance and the behaviour of the driver using sensors with the use of IoT technology.

In this system we continuously monitor the vehicle performance using sensors and the behaviour of driver with the use of IOT Technology. The Vehicle black box receives the information from various sensors like the MEMS sensor. If the accident occurs, by using GPRS/GSM the information is sent to the cloud.

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

S.NO.	NAME	ABBREVIATION
1	PIC	Peripheral Interface Controller
2	I2C	Inter-Integrated Circuit
3	LCD	Liquid Crystal Display
4	LED	Light Emitting Diode
5	GSM	Global System for Mobile communication
6	GPRS	General Packet Radio Service
7	SIM	Subscriber Identity Module
8	IDE	Integrated Development Environment
9	MEMS	Micro electromechanical system
10	MPLAB	Microchip PIC(Peripheral Interface Controller) Development studio

CHAPTER-1

INTRODUCTION

According to the world health organization, more than million people die each year because of the accidents. To prevent this, the car black box system is introduced. Like black box in flight, the car black box technology can play a vital role in vehicle crash investigations. Hence it is significant to have recorders which will track all the activity in vehicles during and after accident or crash. This car black box system is mainly classified into two sections. First section detects and collects the information from the vehicle, and it is implemented using various type of sensors. Second section presents the data to the user in simplified way, and it is implemented by using the PIC microcontroller which is programmed to record the data. If any vehicle crashes, the data will send to the cloud. The investigators can use this recorded data obtained from the car black box to identify the actual reason of the accident This accident black box has accident tracking system. The board cannot be damage while the car is fully damage because is located to bottom of the car.

1.1 Objectives:

- A car black box is to accurately log data related to the vehicle's operation. The PIC microcontroller can be used to collect and store this data in real-time.
- The black box can monitor various vehicle parameters, such as engine RPM, fuel consumption, and temperature. The PIC microcontroller can analyse this data to help improve vehicle performance and identify potential issues.

1.2 Embedded system implementation:

An embedded system is a microprocessor-based computer hardware system with software that is designed to perform a dedicated function, either as an independent system or as a part of a large system. At the core is an integrated circuit designed to carry out computation for real-time operations. Complexities range from a single microcontroller to a suite of processors with connected peripherals and networks; from no user interface to complex graphical user interfaces. The complexity of an embedded system varies significantly depending on the task for which it is designed. Embedded system applications range from digital watches and microwaves to hybrid vehicles and avionics. As much as 98 percent of all microprocessors manufactured are used in embedded systems. Embedded systems are managed by microcontrollers or digital signal processors (DSP), application-specific

integrated circuits (ASIC), field-programmable gate arrays (FPGA), GPU technology, and gate arrays. These processing systems are integrated with components dedicated to handling electric and/or mechanical interfacing. Embedded systems programming instructions, referred to as firmware, are stored in read-only memory or flash memory chips, running with limited computer hardware resources. Embedded systems connect with the outside world through peripherals, linking input and output devices.

The internet of things, or IoT, is a system of interrelated computing devices, mechanical and digital machines, objects, animals or people that are provided with unique identifiers (UIDs) and the ability to transfer data over a network without requiring human-to-human or human-to-computer interaction. A thing in the internet of things can be a person with a heart monitor implant, a farm animal with a biochip transponder, an automobile that has built-in sensors to alert the driver when tire pressure is low or any other natural or man-made object that can be assigned an Internet Protocol (IP) address and is able to transfer data over a network

1.3 Embedded System:

Embedded system includes mainly two sections, they are

1.Hardware

2.Software

1.3.1 Embedded System Hardware:

As with any electronic system, an embedded system requires a hardware platform on which it performs the operation. Embedded system hardware is built with a microprocessor or microcontroller. The embedded system hardware has elements like input output (I/O) interfaces, user interface, memory, and the display.

Usually, an embedded system consists of:

- Power Supply
- Processor
- Memory
- Timers
- Serial communication ports
- Output/Output circuits
- System application specific circuits

Embedded systems use different processors for its desired operation. Some of the

processors used are.

1.Microprocessor

2.Microcontroller

3.Digital signal processor

Microprocessor vs Microcontroller

Microprocessor:

- **CPU** on a chip.
- We can attach required amount of ROM, RAM and I/O ports.
- Expensive due to external peripherals.
- Large in size.
- General-purpose

Microcontroller:

- **Computer** on a chip.
- Fixed of on chip ROM, RAM, I/o ports
- Low cost.
- Compact in size.
- Specific purpose.

1.3.2 Embedded system Software:

The embedded system software is written to perform a specific function. It is typically written in a high-level format and then compiled down to provide code that can be lodged within a non-volatile memory within the hardware. An embedded system software is designed to keep in view of the three limits:

• Availability of system memory

• Availability of processor's speed

• When the system runs continuously, there is a need to limit power dissipation forevents like stop, run, and wake up.

1.4 Applications of Embedded System:

Embedded systems have different applications. A few select applications ofembedded systems are smart cards, telecommunications, satellites, missiles, digitalconsumer electronics, computer networking, etc.

Embedded Systems in Automobiles

- Motor Control System
- Engine or Body Safety
- Robotics in Assembly Line
- Mobile and E-Com Access

Embedded systems in Telecommunications

- Mobile computing
- Networking
- Wireless Communications

Embedded Systems in Smart Cards

- Banking
- Telephone
- Security Systems

1.5 Implementation Flow:

Stage 1:

Considering the problems of existing methods and giving solution to that problem by considering the basic requirements for our proposed system

Stage 2:

Considering the hardware requirement for the proposed system for this we need to select the below components:

1. Microcontroller
2. Inputs for the proposed system (ex: sensors, drivers etc...,)
3. Outputs

Stage 3:

After considering hardware requirements, now we need to check out the software requirements. Based on the microcontroller we select there exists different software for coding, compiling, debugging. we need to write source code for that proposed system based on our requirements and compile, debug the code in that software.

After completing all the requirements of software and hardware we need to bring both together to work our system. For this we need to burn our source code into microcontroller, after burning our source code to microcontroller then connect all input and output modules as per our requirement.

CHAPTER-2
LITERATURE REVIEW

- **Design and Fabrication of Black Box for Automobiles Ananthakrishnan V.K (2015)**

The research paper has been developed in order to record informational data, such as engine, vehicle speed and its temperature, etc. to revolutionize the field of motor vehicle accident investigation. It can be also used for vehicle mapping and accident alert with the help of GPS and GSM technology. This paper is designed with the use of embedded systems. Embedded systems are playing important role in our lives every day, even though they might not necessarily be visible. An embedded system can be defined as a control system or computer system designed to perform a specific task and also be defined as a single purpose computer. Some of the embedded systems we use every day are menu control system on television, the timer in a microwave oven and so on with some amount of intelligence built in. An embedded system contains at least one microprocessor which performs the logic operations for the system. Many embedded systems use one or more microcontrollers, in the type of microprocessor that emphasizes self-sufficiency and cost-effectiveness, instead of a general-purpose microprocessor.

- **Black Box with Intelligent Vehicle System Prafulla U (2017)**

The main purpose of this paper is to develop a prototype model of the Vehicle Black Box System VBBS that can be installed into any vehicle all over the world. This prototype can be designed with minimum number of circuits. The VBBS can contribute to constructing safer vehicles, improving the treatment of crash victims, helping insurance companies with their vehicle crash investigations, and enhancing road status in order to decrease the death rate.

- **Implementation of Car Black-Box using ARM Vidya S (2018)**

This paper is proposed to develop the module which provides the solution to the existing accidental issues. As here the title indicates the paper is about advanced technologies in car for making it more interactive for the collision avoidance and for the purpose of analysing the accident detail to solve accidental cases. In this design ARM7(LPC2129) is used as an embedded controller. As soon as the accident occurs the message will be sent to the nearest emergency medical service or to the respective family member. Various sensors are used to

record different parameters that determines the vehicle status at real time. These measured values will be stored in memory(black box) such as vehicles engine temperature, vehicle distance from obstacle, vehicle speed, carbon dioxide content in smoke, alcohol content, fuel level in tank. The main components of the system consists of the real time sensors like gas sensor, float sensor, accident detection sensor, alcohol sensor, ultrasonic sensor and real time clock buzzer, GSM and module, DC motor.

CHAPTER-3

EXISTING METHOD

Road accidents are a human tragedy. They involve high human suffering and monetary costs in terms of untimely deaths, injuries, and loss of potential income. There are so many new techniques such as Antilock Braking System (ABS), Adaptive Cruise Control (ACC), and Anti-Collision System (ACS) to avoid accidents and in spite of all this, such large number of accidents takes place. Hence this project presents a system which gives an idea about what can be done to analysis the working process of vehicles and other facilities after accident as soon as possible.

3.1 Disadvantages:

- Someone has to witness the incident.
- Moreover, there are delays and inaccuracies due to the expression problem of the witness.

CHAPTER 4
PROPOSED METHOD

Here we are using MEMS sensor to detect the accident. Potentiometer to represent the speed of the vehicle. Whenever the accident occurred then GSM will send the data to the thingspeak application. The PIC16F877A ia as a main controller for connecting all the sensors. LM35 sensor is used to monitor the engine heat. Whenever the engine has high heat or accident occurred then GSM will send data to the cloud.

4.1 BLOCK DIAGRAM FOR PROPOSED METHOD:

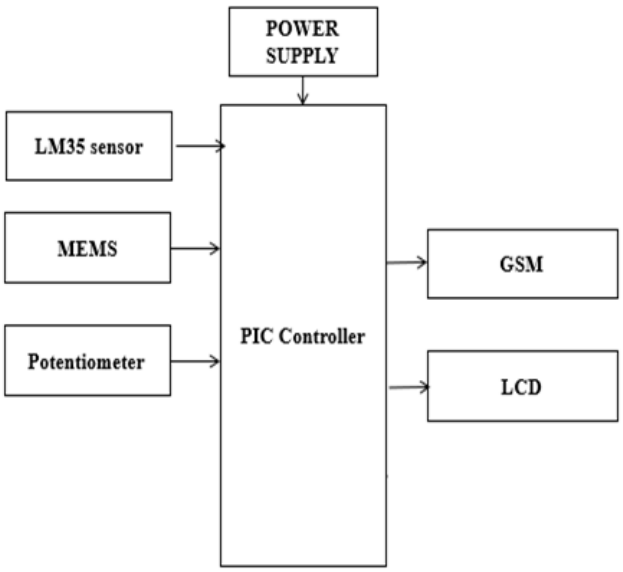


Fig.4.1: Block diagram of proposed system

4.2 Working:

This project is developed to collect the data of the speed, acceleration, and temperature. It can be used for accident reconstruction, driver behaviour analysis, and vehicle diagnostics. It is a device that records data about a vehicle's operation and transmits it to a remote server for analysis. The device is designed to be sensitive to the vehicle's movements, and it can be configured to provide alerts when it detects unusual events. As we know current accident ratio, there are mostly these three reasons are behind the accident. Considering the practical aspects after the accident, we need to prepare all insurance, policy claims. There are several clauses in the polices, to safety each and every clause we need proper documentation, this accident block box helps us.

The MEMS sensor is used to detect the vehicle's acceleration, braking, and other movements. The push button is used for gears and previous recorded data. The potentiometer is used speed. The potentiometer is connected to one of the analog input pins of the microcontroller, and the microcontroller reads the value of the potentiometer to adjust the sensitivity of the sensor be data recorded by the microcontroller is stored in a memory module. The push button is used to record the gear shift information of the vehicle. This information is important in analysing the driving behaviour of the driver. Another push button is used to retrieve the previously recorded data from the EEPROM. This information is important in understanding the vehicle's history and identifying any potential problems.. The GSM module is used to transmit the recorded data to a remote server. The module is connected to the microcontroller through a serial interface. The LED is used to indicate the status of the device. For example, the LED can be used to indicate whether the device is recording data or not. The board cannot be damage while the car is fully damage because is located to bottom of the car.

When an accident occurs, the black box system detects the impact and stores the data in the EEPROM. The data is also sent to the ThingSpeak application through the GSM module. This data can be used to analyse the accident and identify the cause.

4.3 Hardware Architecture:

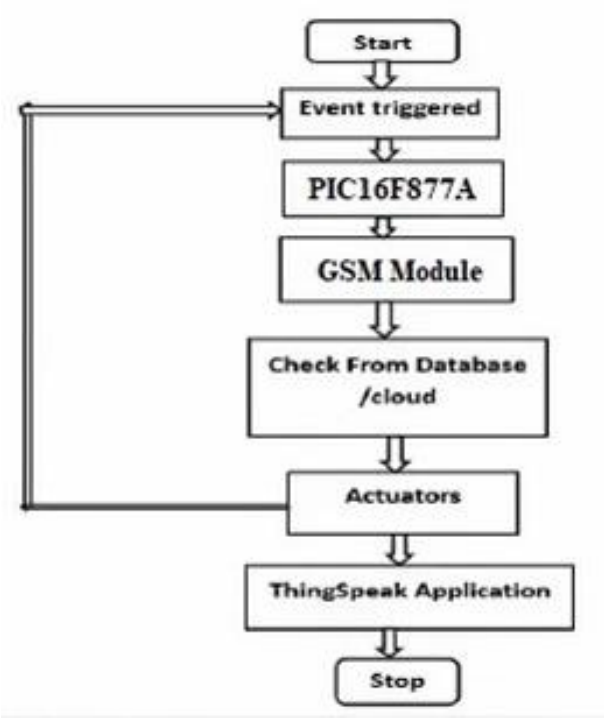


Fig.4.2: Flow chart of hardware architecture

4.3.1 Input design:

The input design involves setting up the connections between the components and the microcontroller using appropriate pins and interfaces. The microcontroller's software program is also designed to read the data from the sensors, store it in the EEPROM, and transmit it to the ThingSpeak application through the GSM module when an accident occurs. The gear shift and previous data record buttons are also programmed to record and retrieve data from the EEPROM, respectively. The PIC microcontroller is the central processing unit that controls and monitors all the input components. The push button is used to recording the data. The potentiometer is used to adjust the sensitivity of the MEMS sensor. This sensor measures the vehicle's acceleration, deceleration, and turning. The LM35 sensor measures the temperature inside the car. he MEMS sensor measures the vehicle's movement in three dimensions (x, y, and z-axis). The LCD display is used to show the recorded data. The EEPROM is a non-volatile memory that is used to store the recorded data Gear Shift Button is used to record the gear shift information of the vehicle Previous Data Record Button is used to retrieve the previously recorded data from the EEPROM.

4.3.2 Output Design:

The output design involves setting up the LCD display to show the recorded data in an easily understandable format. The recorded data can be displayed as numerical values, graphs, or animations depending on the user's preference. The GSM module is also set up to transmit the recorded data to the ThingSpeak application, where it can be analysed and processed. The user can also receive alerts on their mobile device when an accident occurs. The design should be user-friendly and easy to understand to enable the user to make informed decisions about their driving habits. The LCD display is used to show the recorded data in real-time. The data can be shown in different formats, such as numerical values, graphs, or animations. The GSM module is used to transmit the recorded data to the ThingSpeak application. The data can be transmitted as a text message or as a data packet. The GSM module also sends alerts to the user when an accident occurs. The EEPROM is used to store the recorded data. The gear shift button is used to record the gear shift information of the vehicle. The gear shift data can be displayed on the LCD or transmitted to the ThingSpeak application. The previous data record button is used to retrieve the previously recorded data from the EEPROM. The data can be displayed on the LCD or transmitted to the ThingSpeak application.

CHAPTER-5

HARDWARE & SOFTWARE REQUIRMENTS

HARDWARE REQUIREMENTS:

5.1 PIC Microcontroller

5.1.1 PIC Introduction

Peripheral Interface Controllers (PIC) is one of the advanced microcontrollers developed by microchip technologies. These microcontrollers are widely used in modern electronics applications. A PIC controller integrates all types of advanced interfacing ports and memory modules. These controllers are more advanced than normal microcontrollers like 8051. The first PIC chip was announced in 1975 (PIC1650). As with a normal microcontroller, the PIC chip also combines a microprocessor unit called CPU and is integrated with various types of memory modules (RAM, ROM, EEPROM, etc), I/O ports, timers/counters, communication ports, etc.

Microcontroller **PIC16F877A** is one of the PIC micro-Family microcontrollers which is popular at this moment, start from beginner until all professionals. Because very easy to use **PIC16F877A** and use FLASH memory technology so that can be write-erase until thousand times. The superiority this Risc Microcontroller compared to with another microcontroller 8-bit especially at a speed of and his code compression. The 16F877A is a capable microcontroller that can do many tasks because it has a large enough programming memory (large in terms of sensor and control projects) of 8k words and 368 Bytes of RAM. This is enough to do many different projects.

Note: There is a more modern part (the 16F887) that has nearly the same functionality as the 16F877A but also includes an internal clock – like the 16F88 and the 18F4550. In addition, the 16F887 also has low power operation using nano watt^{TM} technology.

Interface	16F877A	16F887	Description
RA4/T0CKI	Open drain	Normal CMOS	The pin is physically different so the input characteristics are changed.
Cost	Expensive	Cheaper	Modern devices are usually cheaper.
ADC	Yes	More useful	More controls although different registers are used.
Nano Watt™	No	Yes	16F88x - ultra low power operation (battery operation).
Internal Clock	No	Yes	8Mhz to 31kHz 1% accuracy.
External Gate	No	Yes	External Timer1 gate input (start Timer1 counter).
Volt reference	No	Yes	Internal 0.6V voltage reference.
RS485, LIN 2.0	No	Yes	Enhanced USART supports RS485 and LIN 2.0 operation.
Parallel Slave port	Yes	No	Acts as an 8 -bit processor interface i.e. another 8 bit processor can read and write to this interface controlling the 16F877A as a slave processor.

Table-5.1: difference between 16F877A & 16F887

The four features that might make you use a 16F887 instead of a 16F877(A) are

- External gate.
- Volt Reference.
- Nano Watt™.
- Internal Clock.

All the above depend on your specific application requirements.

PIC16F877A has 40 pins by 33 paths of I/O. The 40 pins make it easier to use the peripherals as the functions are spread out over the pins. This makes it easier to decide what external devices to attach without worrying too much if there are enough pins to do the job.

One of the main advantages is that each pin is only shared between two or three functions so it's easier to decide what the pin function (other devices have up to 5 functions for a pin).

5.1.2 Pinout

The pinout of the 16F877A is:

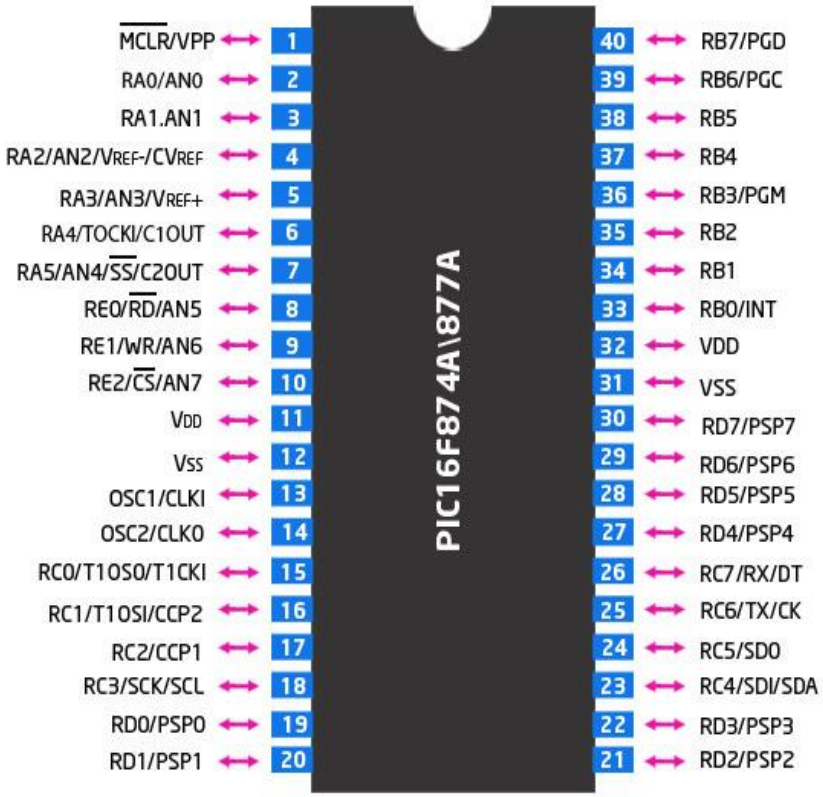


Fig.5.1: PIC16F87A pinout

5,1.3 Features of PIC16F877A (PIC16F877A Introduction)

The PIC16F877A CMOS FLASH-based 8-bit microcontroller is upward compatible with the PIC16C5x, PIC12Cxxx, and PIC16C7x devices. It features 200 ns instruction execution, 256 bytes of EEPROM data memory, self-programming, an ICD, 2 Comparators, 8 channels of 10-bit Analog-to-Digital (A/D) converter, 2 capture/compare/PWM functions, an asynchronous serial port that can be configured as either 3-wire SPI or 2-wire I2C bus, a USART, and a Parallel Slave Port.

High-Performance RISC CPU

- Lead-free; RoHS-compliant
- Operating speed: 20 MHz, 200 ns instruction cycle

- Operating voltage: 4.0-5.5V
- Industrial temperature range (-40° to +85°C)
- 15 Interrupt Sources
- 35 single-word instructions
- All single-cycle instructions except for program branches (two-cycle)

Special Microcontroller Features

- Flash Memory: 14.3 Kbytes (8192 words)
- Data SRAM: 368 bytes
- Data EEPROM: 256 bytes
- Self-reprogrammable under software control
- In-Circuit Serial Programming via two pins (5V)
- Watchdog Timer with on-chip RC oscillator
- Programmable code protection
- Power-saving Sleep mode
- Selectable oscillator options
- In-Circuit Debug via two pins

Peripheral Features

- 33 I/O pins; 5 I/O ports
- Timer0: 8-bit timer/counter with 8-bit Prescaler
- Timer1: 16-bit timer/counter with Prescaler
- Can be incremented during Sleep via external crystal/clock
- Timer2: 8-bit timer/counter with 8-bit period register, Prescaler, and postscaler
- Two Capture, Compare, PWM modules
- 16-bit Capture input; max resolution 12.5 ns
- 16-bit Compare; max resolution 200 ns
- 10-bit PWM
- Synchronous Serial Port with two modes:
- SPI Master
- I2C Master and Slave
- USART/SCI with 9-bit address detection
- Parallel Slave Port (PSP)
- 8 bits wide with external RD, WR, and CS controls

- Brown-out detection circuitry for Brown-Out Reset

Analog Features

- 10-bit, 8-channel A/D Converter
- Brown-Out Reset
- Analog Comparator module
- analog comparators
- Programmable on-chip voltage reference module
- Programmable input multiplexing from device inputs and internal VREF
- Comparator outputs are externally accessible.

Fig.5.2: PIC16F877A

5.2 MEMS Sensor:

The term MEMS stands for micro-electro-mechanical systems. These are a set of devices, and the characterization of these devices can be done by their tiny size & the designing mode. The designing of these sensors can be done with the 1- 100-micrometer components. These devices can differ from small structures to very difficult electromechanical systems with numerous moving elements beneath the control of incorporated micro-electronics. Usually, these sensors include mechanical micro-actuators, micro-structures, micro-electronics, and micro-sensors in one package. This article discusses what is a MEMS sensor, working principle, advantages and it's applications.

MEMS are low-cost, and high accuracy inertial sensors and these are used to serve an extensive range of industrial applications. This sensor uses a chip-based technology namely micro-electro-mechanical-system. These sensors are used to detect as well as measure the external stimulus like pressure, after that it responds to the pressure which is measured pressure

with the help of some mechanical actions. The best examples of this mainly include revolving of a motor for compensating the pressure change.

The MEMS accelerometers can be divided into two important micro system architectures: piezo resistive and capacitive. Even though both of these two types of accelerometers possess internal proof masses which are excited by acceleration, the differences of these two architectures lie in the transduction mechanism which is used to the movement correlation of the internal proof mass to accelerate.

By sensing the mounting angle, the sensor can assist in compensating for the devices mounting angle, and therefore makes it possible to use ACCELEROMETER FACTSHEET MEMS 3-AXIS ACCELEROMETER normal SMD technology in high density boards, and also to realise the precise detection of the inclination angle. An interface IC within the sensor package also has temperature sensing and self-diagnosis functions.

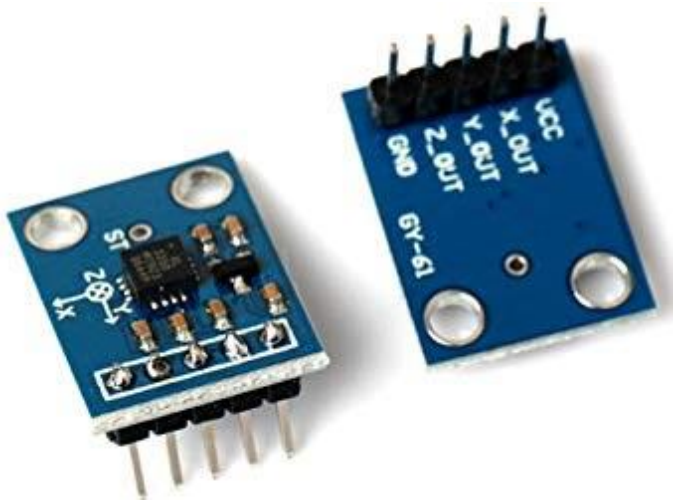


Fig.5.3: MEMS Sensor

5.3 LCD:

LCD (Liquid Crystal Display) is the innovation utilized in scratch pad shows and other littler PCs. Like innovation for light-producing diode (LED) and gas-plasma, LCDs permit presentations to be a lot more slender than innovation for cathode beam tube (CRT). LCDs expend considerably less power than LED shows and gas shows since they work as opposed to emanating it on the guideline of blocking light.

A LCD is either made with a uninvolved lattice or a showcase network for dynamic framework show. Likewise alluded to as a meagre film transistor (TFT) show is the dynamic framework LCD. The uninvolved LCD lattice has a matrix of conductors at every crossing point of the network with pixels. Two conductors on the lattice send a current to control the light for any pixel. A functioning framework has a transistor situated at every pixel crossing point, requiring less current to control the luminance of a pixel.

5.3.1 Data/Signals/Execution of LCD

Now that was all about the signals and the hardware. Let us come to data, signals and execution. Two types of signals are accepted by LCD, one is data and one is control. The LCD module recognizes these signals from the RS pin status. By pulling the R / W pin high, data can now also be read from the LCD display. Once the E pin has been pulsed, the LCD display reads and executes data at the falling edge of the pulse, the same for the transmission case. It takes 39-43µS for the LCD display to place a character or execute a command. It takes 1.53ms to 1.64ms except for clearing display and searching for cursor to the home position.

Fig.5.4: LCD

5.3.2 Pin Diagram:

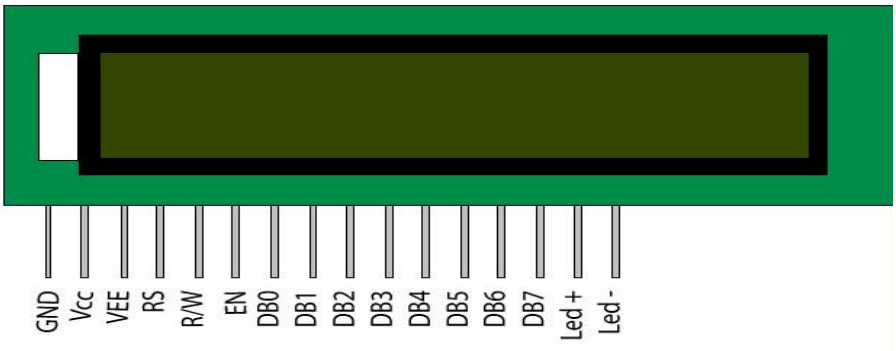


Fig.5.5: LCD pin diagram

Pin No	Function	Name
1	Ground (0V)	Ground
2	Supply voltage; 5V (4.7V – 5.3V)	Vcc
3	Contrast adjustment; through a variable resistor	V_{EE}
4	Selects command register when low; and data register when high	Register Select
5	Low to write to the register; High to read from the register	Read/write
6	Sends data to data pins when a high to low pulse is given	Enable
7	8-bit data pins	DB0
8		DB1
9		DB2
10		DB3
11		DB4
12		DB5
13		DB6
14		DB7
15	Backlight V_{CC} (5V)	Led+
16	Backlight Ground (0V)	Led-

Table-5.2: pin description

RS (Register select)

A 16X2 LCD has two order and information registers. The determination of the register is utilized to change starting with one register then onto the next. RS=0 for the register of directions, while RS=1 for the register of information.

Command Register

The guidelines given to the LCD are put away by the direction register. An order is a direction given to LCD to play out a predefined assignment, for example, instating it, clearing its screen, setting the situation of the cursor, controlling showcase, and so on. Order preparing happens in the direction register.

Data Register:

The information register will store the information that will be shown on the LCD. The information is the character's ASCII incentive to show on the LCD. It goes to the information register and is prepared there when we send information to the LCD. While choosing RS=1, the information register.

5.3.3 Command codes for LCD:

Sr. No.	Hex Code	Command to LCD instruction Register
1	01	Clear display screen
2	02	Return home
3	04	Decrement cursor (shift cursor to left)
4	06	Increment cursor (shift cursor to right)
5	05	Shift display right
6	07	Shift display left
7	08	Display off, cursor off
8	0A	Display off, cursor on
9	0C	Display on, cursor off
10	0E	Display on, cursor blinking
11	0F	Display on, cursor blinking
12	10	Shift cursor position to left
13	14	Shift cursor position to right
14	18	Shift the entire display to the left
15	1C	Shift the entire display to the right
16	80	Force cursor to beginning (1st line)
17	C0	Force cursor to beginning (2nd line)
18	38	2 lines and 5×7 matrix

Table-5.3: Command codes for LCD

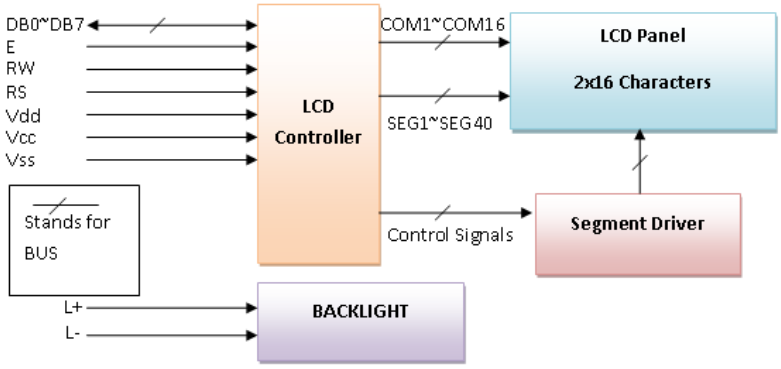


Fig.5.6: Block diagram of LCD

5.4 Potentiometer:

A potentiometer (also known as a pot or potmeter) is defined as a 3 terminal variable resistor in which the resistance is manually varied to control the flow of electric current. A potentiometer acts as an adjustable voltage divider.

Fig.5.7: potentiometer

A potentiometer is a passive electronic component. Potentiometers work by varying the position of a sliding contact across a uniform resistance. In a potentiometer, the entire input voltage is applied across the whole length of the resistor, and the output voltage is the voltage drop between the fixed and sliding contact as shown below.

This is a very basic instrument used for comparing the emf of two cells and for calibrating ammeter, voltmeter, and watt-meter. The basic **working principle of a potentiometer** is quite simple. Suppose we have connected two batteries in parallel through a galvanometer. The

negative battery terminals are connected together, and positive battery terminals are also connected together through a galvanometer as shown in the figure below.

5.5 GSM/GPRS Module:

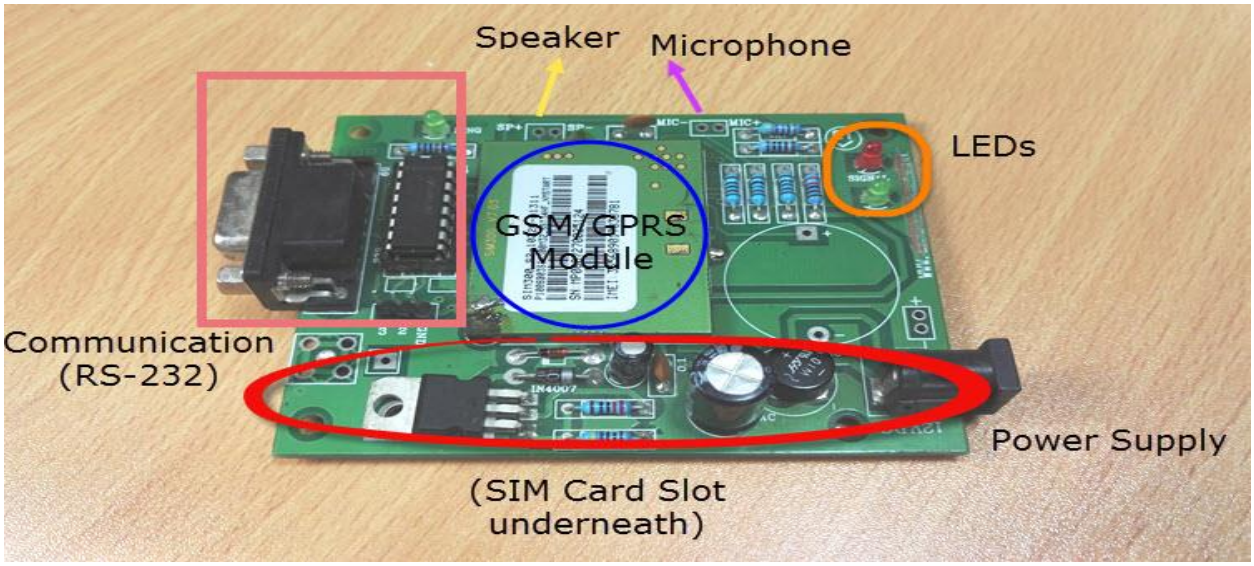


Fig.5.8: GSM/GPRS Module

GPRS Modules are one of the commonly used communication modules in embedded systems. A GPRS Module is used to enable communication between a microcontroller (or a microprocessor) and the GPRS Network. Here, GSM stands for Global System for Mobile Communication and GPRS stands for General Packet Radio Service.

A GPRS MODEM comprises of a GPRS Module along with some other components like communication interface (like Serial Communication – RS-232), power supply and some indicators. With the help of this communication interface, we can connect the GSM GPRS Module on the GPRS MODEM with an external computer (or a microcontroller).

5.5.1 Features and Specifications of SIM800A Module:

- SIM800A Quad Band GSM Module
- Bands: GSM 850MHz, EGSM 900MHz, DCS 1800MHz, PCS 1900MHz
- Coding schemes: CS-1, CS-2, CS-3, CS-4 Tx power: Class 4 (2W), Class 1(1W)
- GPRS class 2/10.
- Control via AT commands (3GPP TS 27.007, 27.005 and SIMCOM enhancedAT (command set).
- Voltage Supply Required- 9VDC to 12VDC with at least 2A Peak Current

Capability

- High-Quality Product (Not hobby grade).
- 5V interface for direct communication with MCU kit.
- TTL Rx and TTL Tx and DB9 Connector Based RS232 Outputs
- Configurable baud rate.
- Built-in SIM Card holder.
- Built-in Network Status LED.
- Inbuilt Powerful TCP/IP protocol stack for internet data transfer over GPRS.
- Low power.
- Operating temperature: -40C to +85C
- External Finger type antenna
- Weight - 40gm

5.5.2 Applications:

- Remote Data Monitor and Control.
- Water, gas and oil flow metering.
- AMR (automatic meter reading).
- Power station monitoring and control.
- Remote POS (point of sale) terminals.
- Traffic signals monitor and control.
- Fleet management.
- Power distribution network supervision.
- Central heating system supervision.
- Weather station data transmission.
- Hydro-logic data acquisition.
- Vending machine.
- Traffic info guidance.
- Parking meter and Taxi Monitor.
- Telecom equipment supervision (Mobile base station, microwave or opticalrelay station).

5.6 Switch:

In electrical engineering, a switch is an electrical component that can disconnect or connect the conducting path in an electrical circuit, interrupting the electric current or diverting it from one conductor to another.

Switches are made in many different configurations; they may have multiple sets of contacts controlled by the same knob or actuator, and the contacts may operate simultaneously, sequentially, or alternately. A switch may be operated manually, for example, a light switch or a keyboard button, or may function as a sensing element to sense the position of a machine part, liquid level, pressure, or temperature, such asa thermostat. Many specialized forms exist, such as the toggle switch, rotaryswitch, mercury switch, push-button switch, reversing switch, relay, and circuit breaker.

Fig.5.9: Switch

5.7 LM35 sensor:

The LM35 is a temperature sensor that can measure the temperature in Celsius scale. It is a popular analog sensor used in many electronic projects, such as temperature control systems, temperature logging devices, and weather stations.

The LM35 sensor outputs an analog voltage proportional to the temperature it senses. For example, at 0 degrees Celsius, the output voltage of the LM35 is 0 volts, and at 100 degrees Celsius, the output voltage is 1 volt. This linear relationship between temperature and voltage output makes it easy to interface the LM35 with analog circuits, microcontrollers, and other digital devices.

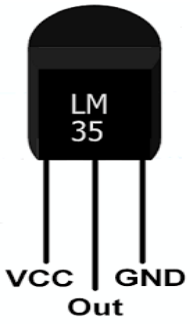


Fig.5.10: LM35 sensor

5.8 Power Supply:

A power supply is a component that provides at least one electrical charge with power. It typically converts one type of electrical power to another, but it can also convert a different Energy form in electrical energy, such as solar, mechanical, or chemical.

A power supply provides electrical power to components. Usually the term refers to devices built into the powered component. Computer power supplies, for example, convert AC current to DC current and are generally located along with at least one fan at the back of the computer case.

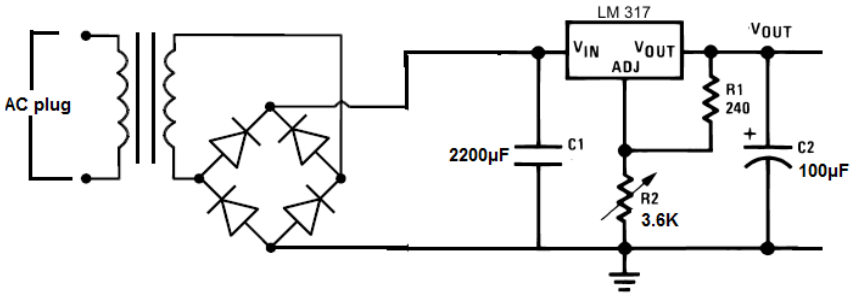
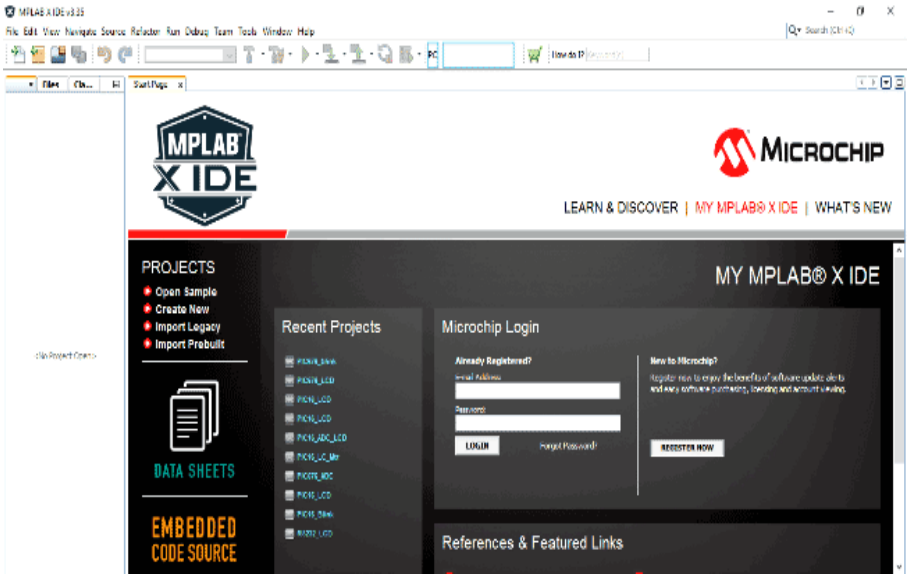


Fig.5.11: power supply

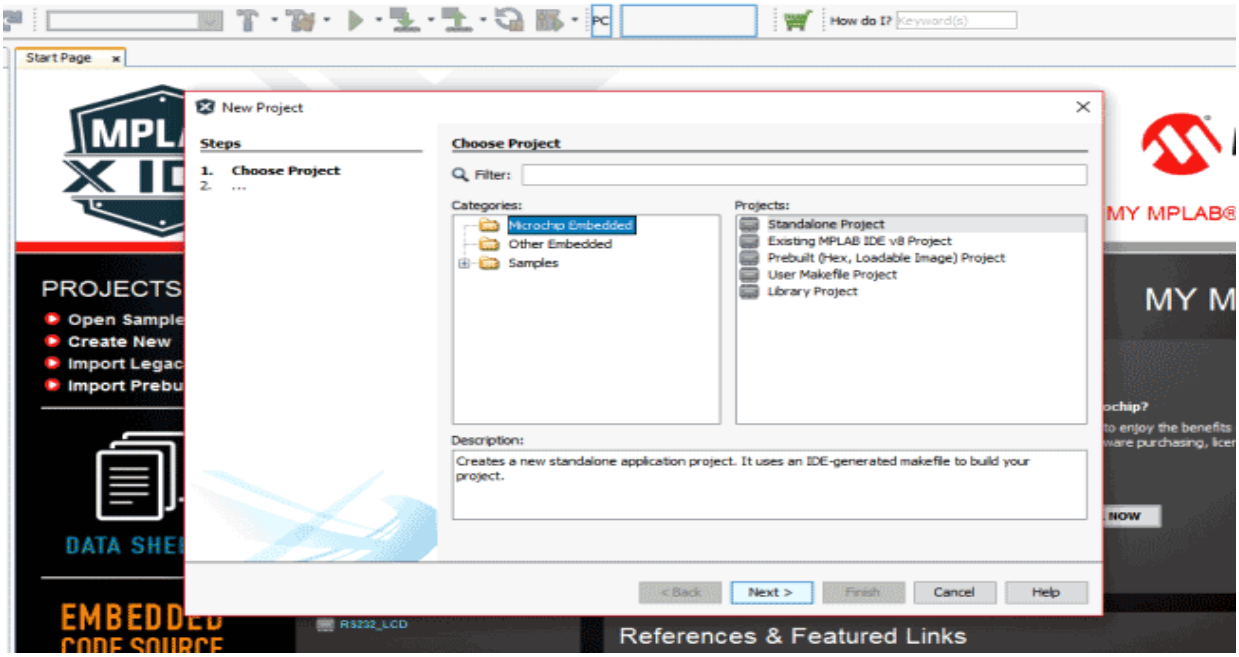
SOFTWARE REQUIRMENTS:

5.9 MPLAB:

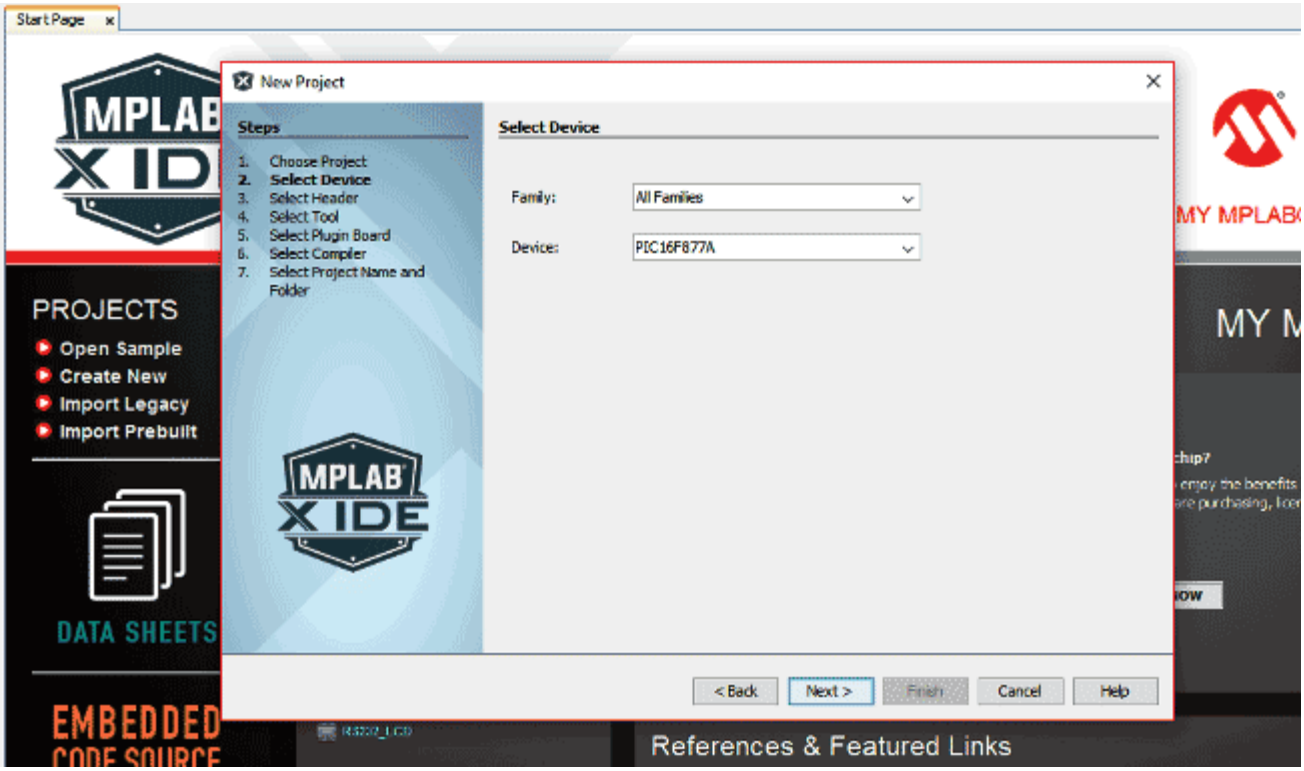
Step 1: Launch the MPLAB-X IDE that we installed in the previous class, once loaded it should look something like this.



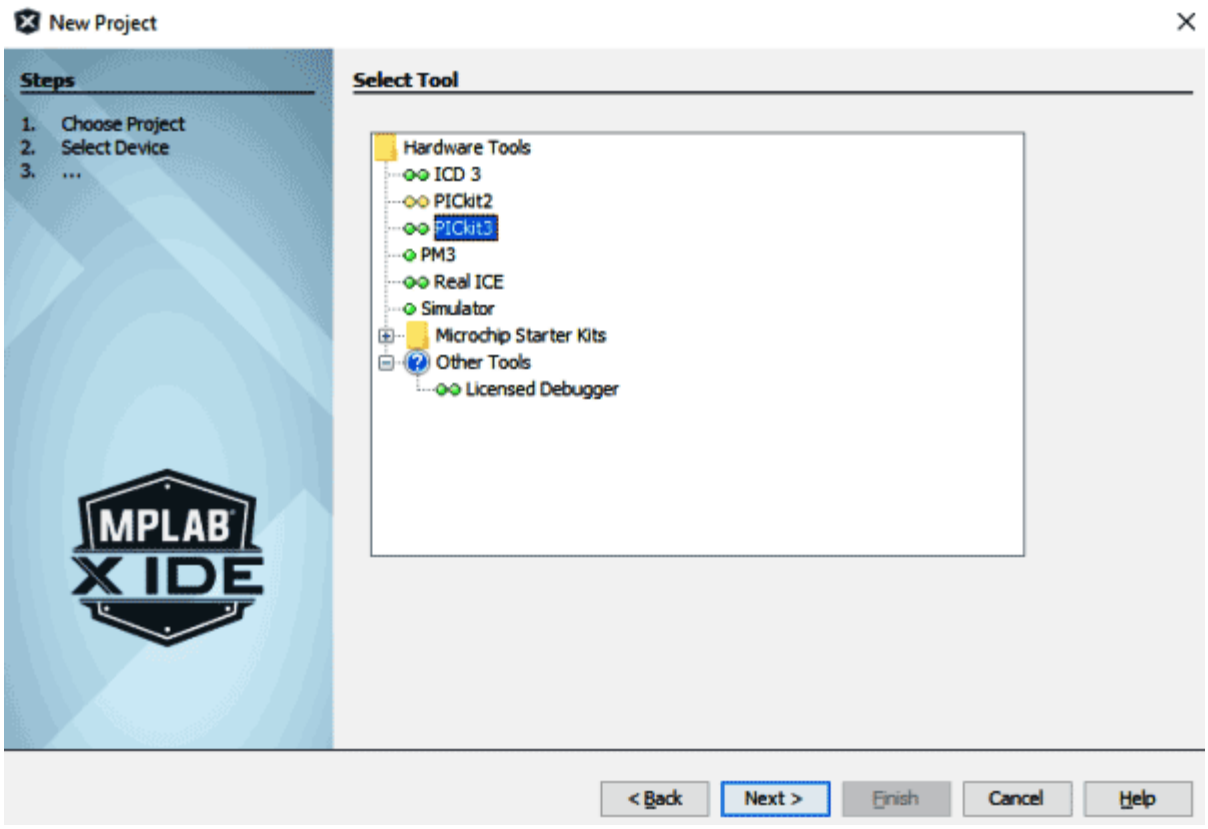
Step 2: Click on Files -> New Project, or use the hotkey Ctrl+Shift+N. You will get the following POP-UP, from which you have to select *Standalone Project* and click Next.



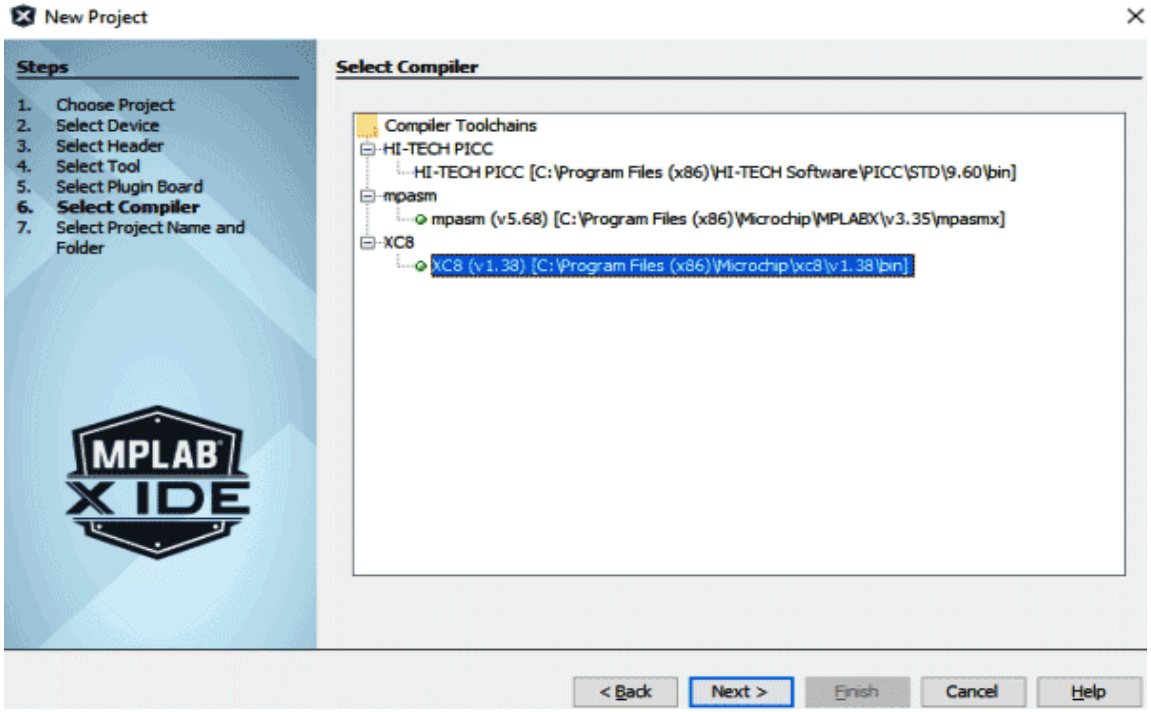
Step 3: Now we have to select our Device for the project. So type as PIC16F877A over the *Select Device* dropdown section. Once done it should be like this and then Click on Next.



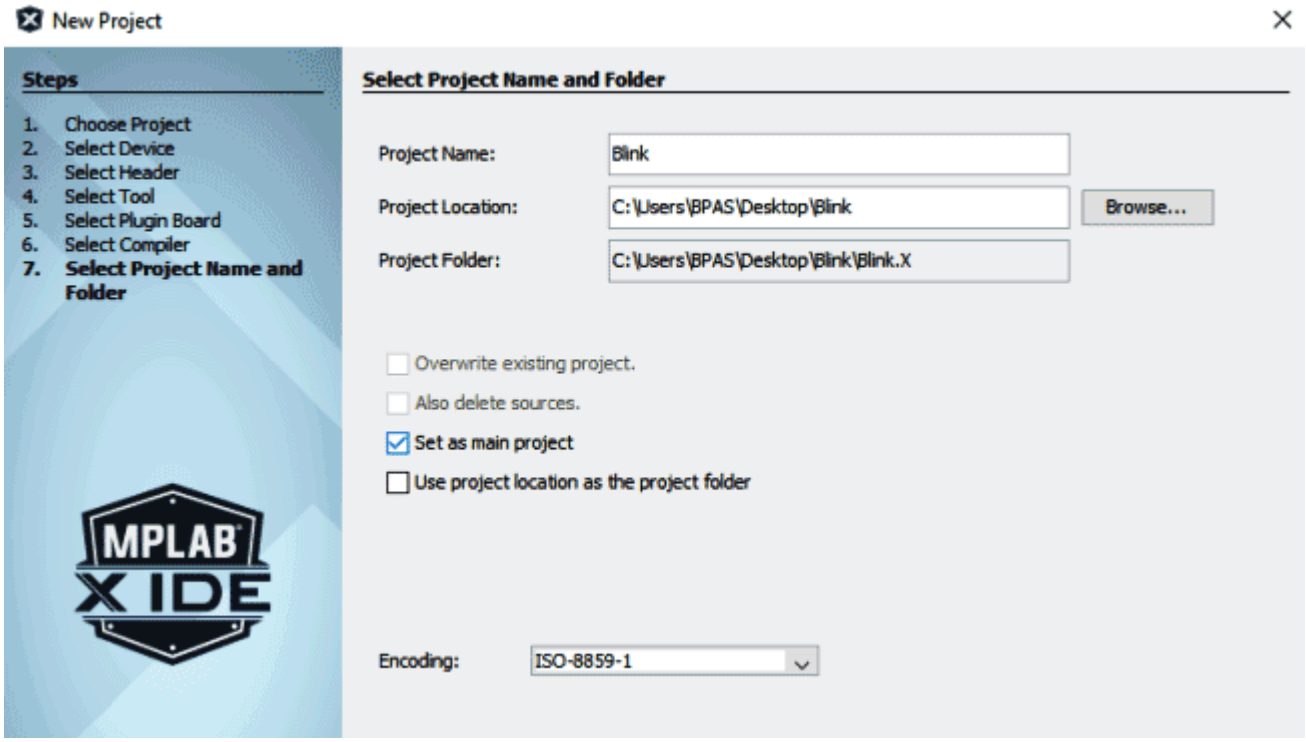
Step 4: The next page will allow us to select the tool for our project. This would be PicKit 3 for our project. Select PicKit 3 and click on next



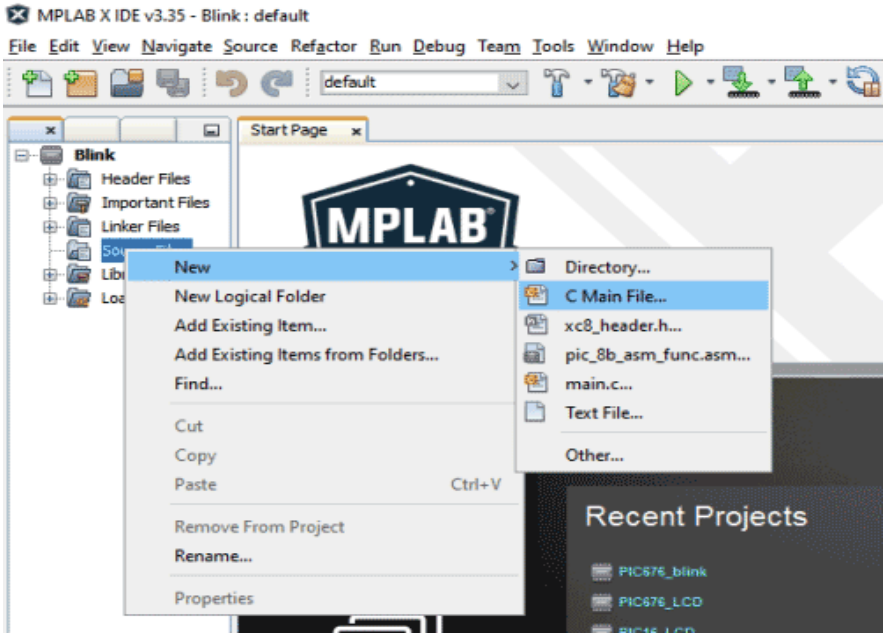
Step 5: Next page will ask to select the compiler, select the XC8 Compiler and click next.



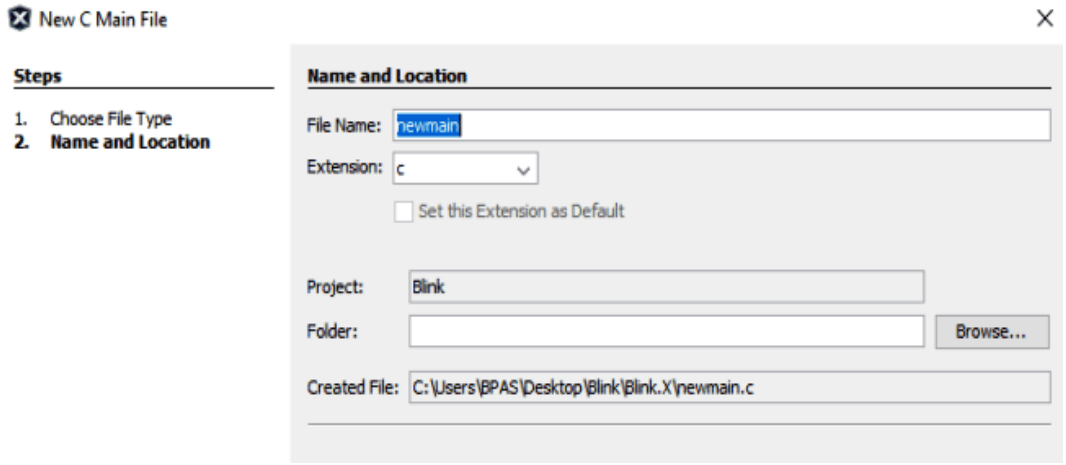
Step 6: In this page we have to name our project and select the location where the project has to be saved. I have named this Project as **Blink** and saved it on my desktop. You can name and save it in your preferable way. Our project will be saved as a folder with the Extension **.X**, which can be directly launched by MAPLB-X. Click Finish once done.



Step 7: That's it!!! Our project has been created. The left most window will show the project name (Here Blink), click on it so that we can view all the directories inside it.



Step 8: The following dialog box will appear in which the name of the C-file has to be mentioned. I have named in Blink again, but the choice is left to you. Name it in the File name column and click on finish.



Step 9: Once the C main File is created, the IDE will open it for us with some default codes in it, as shown below.

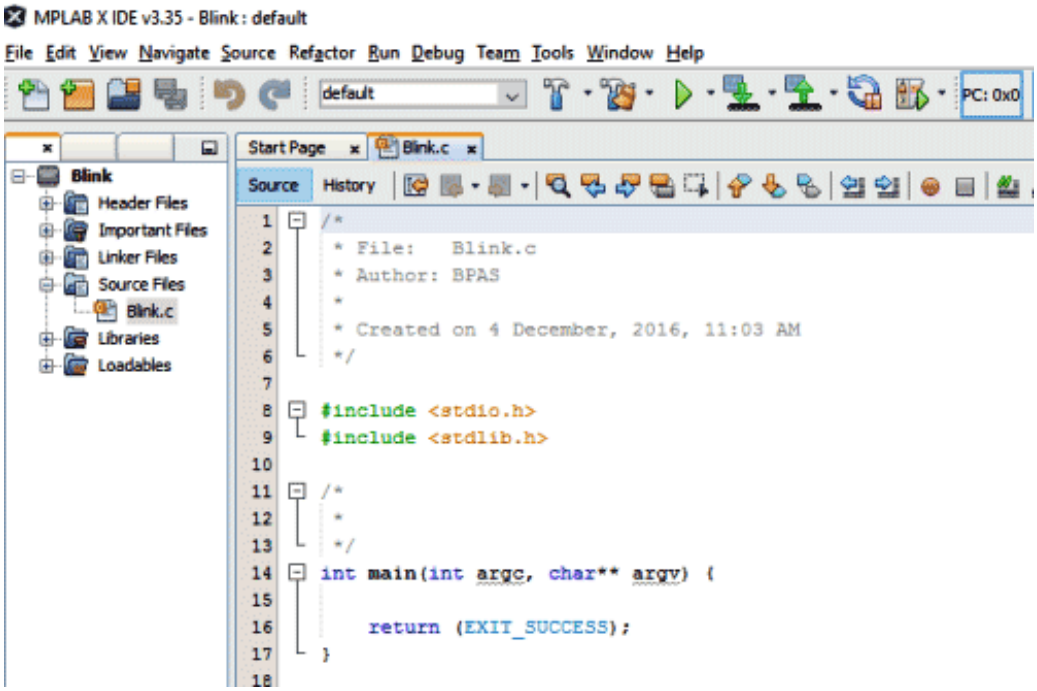


Fig.5.12: MPLAB IDE

Step 10: That's it now we can start programming our code in the C-main File. The default code will not be used in our tutorials. So let's delete them completely.

CHAPTER-6

ADVANTAGES AND APPLICATIONS

6.1 ADVANTAGES:

- Integrated system.
- Highly secured system.
- Simple and not harmed system.

6.2 APPLICATIONS:

- Insurance claim.
- Accident investigation.
- Event Record.

CHAPTER-7

RESULTS & DISCUSSION

The below figure shows the hardware setup of car black box using PIC microcontroller. It consists of potentiometer, switch, MEMS sensor and temperature sensor acts as input to microcontroller. The GSM module, LCD are the output. The whole operations are controlled using PIC16F877A microcontroller.

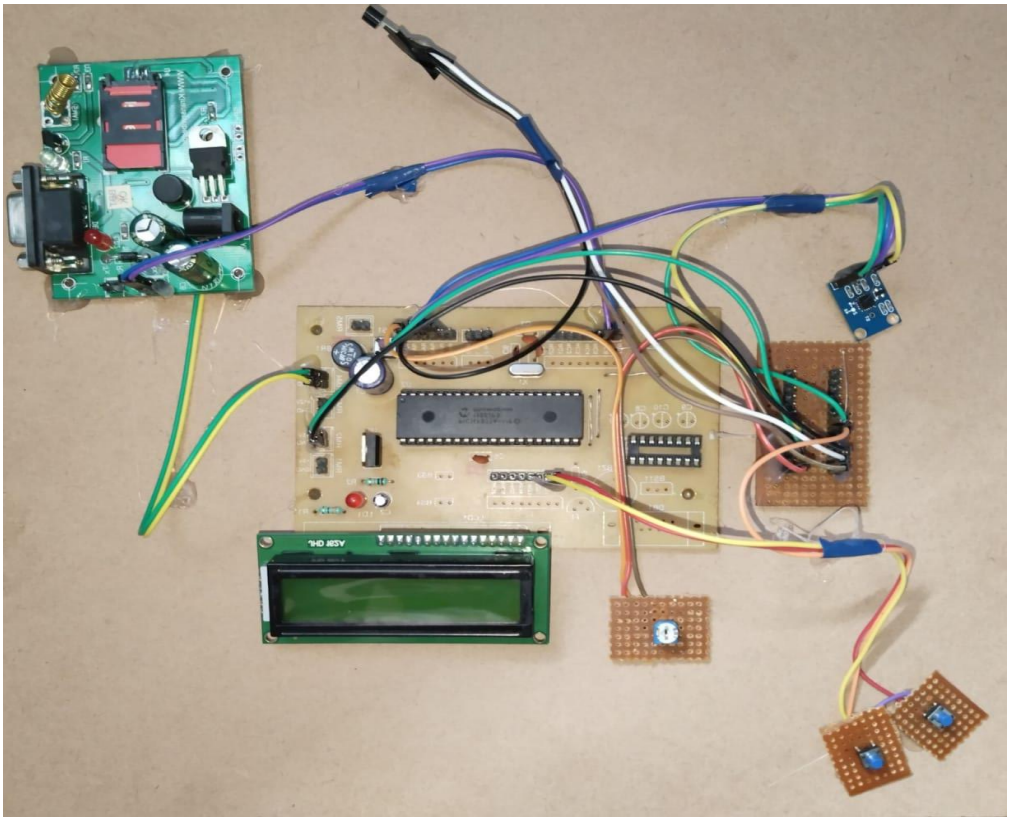


Fig.7.1: Working Hardware model

The testing results of a car black box using PIC microcontroller, push button, potentiometer, LM35 sensor, MEMS sensor, LCD, and GSM involve verifying the accuracy and functionality of the system. It is essential to test each component of the system thoroughly to ensure that the system performs optimally in real-world scenarios. The LM35 sensor is tested by exposing it to different temperatures and verifying that the recorded values on the LCD display and ThingSpeak application are accurate. The MEMS sensor is tested by recording the vehicle's acceleration, deceleration, and sudden stops. The recorded data should be displayed accurately on the LCD and transmitted to the ThingSpeak application. The push buttons are tested by pressing them and verifying that the gear shift information is recorded correctly and can be retrieved using the previous data record button. The GSM module is tested by sending

data to the cloud and displayed correctly on the LCD display and the user's mobile device. The EEPROM is tested by recording data and verifying that the data can be retrieved using the previous data record button. The retrieved data should be displayed accurately on the LCD and transmitted to the ThingSpeak application. The expected results of the testing scenarios are that the recorded data is accurate and can be transmitted to the ThingSpeak application and displayed correctly on the LCD. The system should also be able to send alerts to the user's mobile device when an accident occurs.

Fig.7.2: Output Value

The collection of data store in EEPROME. In different situations we get different data. Now see the test cases of this project. As we are operating in the PC, we keep PC screenshots and image of these test cases.

TEST CASE 1:

Fig.7.3: LCD Recorded

TEST CASE 2:

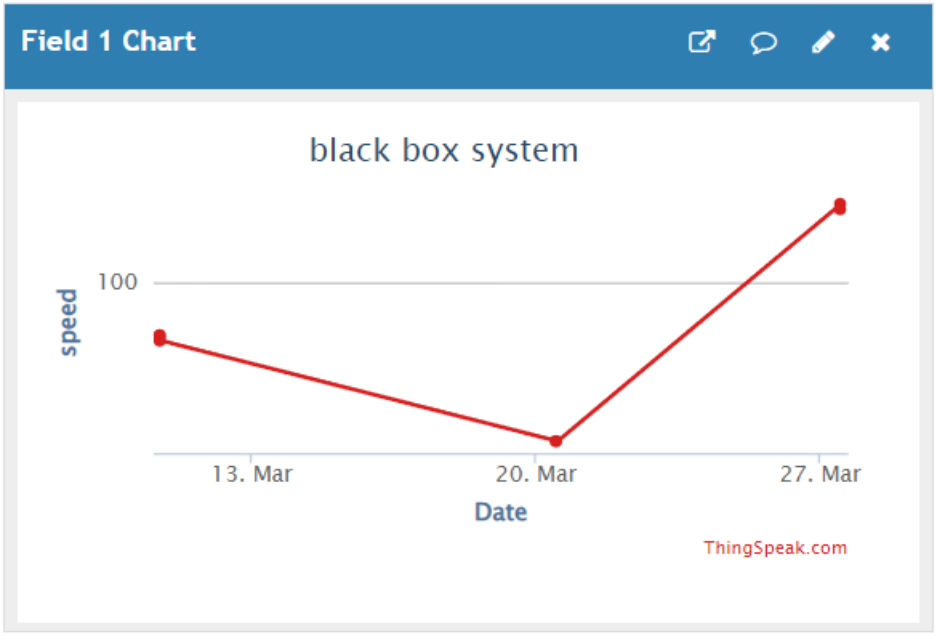


Fig.7.4: Speed Readings

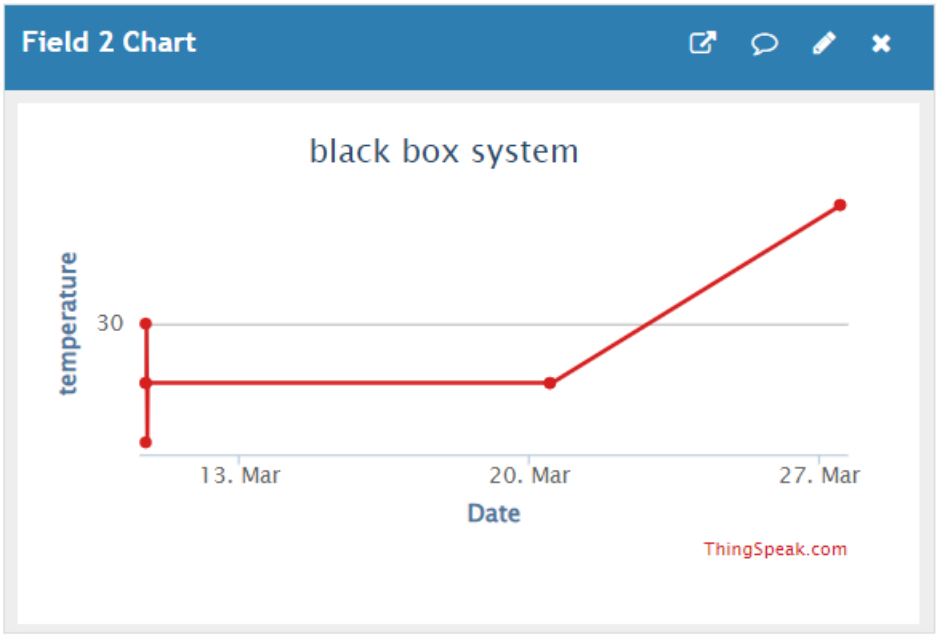


Fig.7.5: Temperature Readings

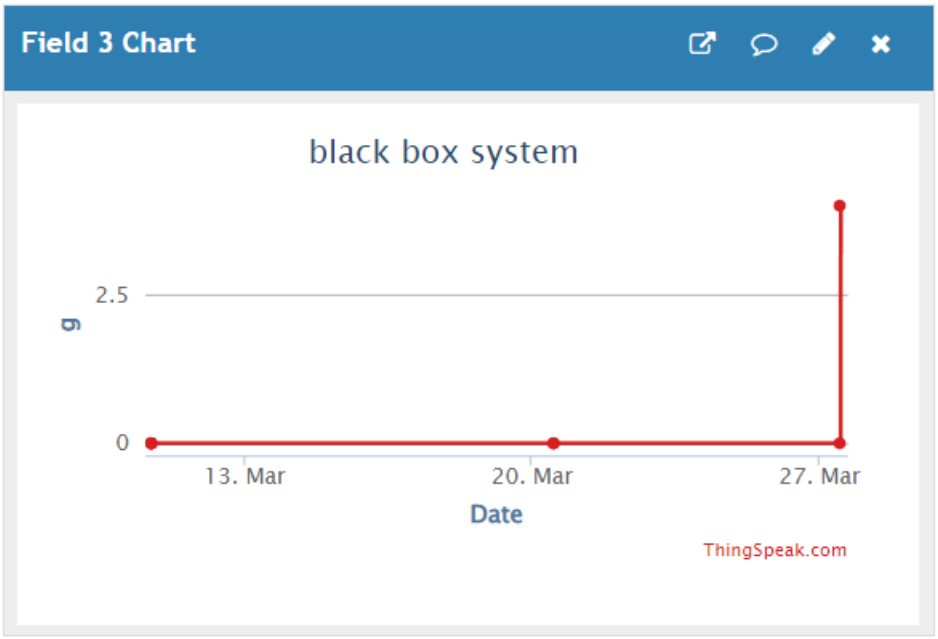


Fig.7.6: Gear Readings

CHAPTER-8

CONCLUSION

The proposed system will provide important information at the time of any accident. When any type of accident will occur due to any reason, the car black box system provides necessary data to generate the report of accident and its cause. This paper offers a user-friendly program to analyse the data of the accident. This car black box system can be implemented in any vehicle. As soon as the driver runs or start the vehicles the system will start collecting data from all the sensors along with date and time. The data stored in the memory can be retrieved after the accident using Thingspeak cloud.

CHAPTER-9
FUTURE SCOPE

In future we can interface the system with vehicle airbag system. This will optimize the proposed technology to the maximum extent and deliver the best accident detection system. The proposed system can be improvised by adding other components which will be able to collect several information such as recording the voice, recording the video, to detect the part of the vehicle where it is hit.

CHAPTER-10

REFERENCES

[1] Amin, Md Syedul, Jubayer Jalil, and Mamun Bin Ibne Reaz. "Accident detection and reporting system using GPS, GPRS and GSM technology." 2012 International Conference on Informatics, Electronics & Vision (ICIEV). IEEE, 2012.

[2] Desai, Vikas, Swati P. Nawale, and Sachin R. Kokane. "Design and Implementation of GSM and GPS Based Vehicle Accident Detection System." IJIT 1.03 (2013): 1-4.

[3] Vaishnavi, M., et al. "Intelligent alcohol detection system for car." International Journal of Scientific & Engineering Research 5.11 (2014): 2229-5518..

[4] Kim, Jungyun, et al. "Vibration sensor characteristics of piezoelectric electro-active paper." Journal of Intelligent Material Systems and Structures 21.11 (2010): 1123-1130.

[5] Katkar, Suhas, Mahesh Manik Kumbhar, and Priti Navanath Kadam. "Accident Prevention System Using Eye Blink Sensor." International Research Journal of Engineering and Technology (IRJET) 3.05 (2016): 1588-1590.

[6] Muruganandham, PR Mukes, and R. Mukesh. "Real time web based vehicle tracking using GPS." World Academy of Science, Engineering and Technology 61.1 (2010): 91-9.

Appendix-A

Project Source Code:

```
#include<pic.h>
#include<stdio.h>
#define lcd_data PORTB
#define lcd_rs RB0
#define lcd_en RB1
#define buzzer RD4
#define motor RD2
#define pulse  RD0
#define  ed  RD1
unsigned int g=0;
float p,tmpr,f,result;
 unsigned int t1,t2,t3;
unsigned int test=0,value;
unsigned int speed=0;
void delay(unsigned int x)
{
  unsigned int k;
    unsigned int j;
      for(k=0;k<x;k++)
        for(j=0;j<165;j++);
}
void lcdcmd(unsigned char value)                    // LCD COMMAND
```

```
 {
        lcd_data=value&(0xf0); //send msb 4 bits
    lcd_rs=0;       //select command register
    lcd_en=1;      //enable the lcd to execute command
        delay(3);
        lcd_en=0;
    lcd_data=((value<<4)&(0xf0));          //send lsb 4 bits
    lcd_rs=0;      //select command register
    lcd_en=1;     //enable the lcd to execute command
        delay(3);
        lcd_en=0;
 }
void lcd_init(void)
{
 lcdcmd(0x02);
 lcdcmd(0x02);
 lcdcmd(0x28);  //intialise the lcd in 4 bit mode*/
 lcdcmd(0x28);  //intialise the lcd in 4 bit mode*/
 lcdcmd(0x0e);  //cursor blinking
 lcdcmd(0x06);  //move the cursor to right side
 lcdcmd(0x01);  //clear the lcd
}
void lcddata(unsigned char value)
 {
    lcd_data=value&(0xf0); //send msb 4 bits
    lcd_rs=1;       //select data register
```

```
   // while(1);
     lcd_en=1;      //enable the lcd to execute data
              delay(3);
              lcd_en=0;
     lcd_data=((value<<4)&(0xf0));          //send lsb 4 bits
     lcd_rs=1;       //select data register
     lcd_en=1;      //enable the lcd to execute data
              delay(3);
              lcd_en=0;
    delay(3);
  }
void ADC_Init()
{
  ADCON0 = 0x41; //ADC Module Turned ON and Clock is selected
  ADCON1 = 0xC0; //All pins as Analog Input
               //With reference voltages VDD and VSS
}
unsigned int ADC_Read(unsigned char channel)
{
  if(channel > 7) //If Invalid channel selected
    return 0;     //Return 0
  ADCON0 &= 0xC5; //Clearing the Channel Selection Bits
  ADCON0 |= channel<<3; //Setting the required Bits
  delay(10);
  //__delay_ms(2); //Acquisition time to charge hold capacitor
  ADCON0 = (ADCON0 | 0x04);
```

```
  //GO_nDONE = 1; //Initializes A/D Conversion
  //while(GO_nDONE); //Wait for A/D Conversion to complete
while((ADCON0&0X04)==0X04);
return ((ADRESH<<8)+ADRESL); //Returns Result
}

void msgdisplay(const unsigned char b[]) // send string to lcd
  {
unsigned char s,count=0;
for(s=0;b[s]!='\0';s++)
 {
  count++;
  if(s==16)
   lcdcmd(0xc0);
   if(s==32)
   {
   lcdcmd(1);
   count=0;
   }
  lcddata(b[s]);
 }
}
void convert(unsigned int c)
{
   unsigned int a1,b1,a,b,d,e;
   a1=c/1000;
```

```
  a1=a1|0x30;
  lcddata(a1);
  b1=c%1000;
  a=b1/100;
  a=a|0x30;
  lcddata(a);
  b=c%100;
  d=b/10;
  d=d|0x30;
  lcddata(d);
  e=b%10;
  e=e|0x30;
  lcddata(e);
}
void serinit()
{
 TRISC7 = 1;
 TRISC6 = 0;
//SPBRG = 12;//19200
 SPBRG = 25;//9600
// SPBRG = 51;//4800
// SPBRG = 103;//2400
// SPBRG = 207;//1200
 BRGH = 1;
 TXSTA=0x24;
 RCSTA=0x90;
```

```
 TXEN=1;
 SPEN=1;
}
void sertxc(unsigned char tx)
{
 TXREG = tx;
 while(!TRMT);
}
void sertxs(const unsigned char *txs)
{
 for(;*txs != '\0';txs++)
 {
  TXREG = *txs;
  while(!TRMT);
 }
}
unsigned char receive()
{
 unsigned char rce;
   CREN = 1;
   OERR=0;
    while(RCIF==0);//{if(OERR == 1)OERR=0;}
                        rce=RCREG;
   CREN=0;
         RCIF=0;
 return rce;
```

```
}
void okcheck()
{
 unsigned char rce;
 do{
    rce = receive();
   }while(rce != 'K');
}



#define send sertxs         //define your serial string sending function
void converts(unsigned int c)
{
   unsigned int a1,b1,a,b,d,e;
   a1=c/1000;
   a1=a1|0x30;
   sertxc(a1);
   b1=c%1000;
   a=b1/100;
   a=a|0x30;
   sertxc(a);
   b=c%100;
   d=b/10;
   d=d|0x30;
   sertxc(d);
   e=b%10;
```

```
   e=e|0x30;
   sertxc(e);
}
 void GPRS_send1();
void main()
{
unsigned int  x,y;
 unsigned char rr,count;
unsigned char mydata;
   unsigned int t=0;
unsigned int soil=0;
motor=0;
pulse=1;
 TRISB = 0x00;      //OUTPUT
 //TRISD = 0xff;      //INPUT
 TRISD = 0x0F;
TRISC = 0xFF;
unsigned int a1,a2,a3,a4;
  TRISA=0x2F;//make portA as input.
  PORTA=0X00;
  TRISB=0x00;//Output
 ADCON1=0X8E;
 ADCON0=0X01;
 ADC_Init();
   serinit();
```

```
                    //INTEGRATED MULTI FUNCTIONAL ENVIRONMENTAL SENSORS
USING GSM TECHNOLOGY
   lcd_init();
   lcdcmd(1);
   msgdisplay("WELCOME");
delay(400);
 lcdcmd(1);
 msgdisplay("WIRELESS ");
lcdcmd(0xc0);
 msgdisplay("BLACK BOX");
delay(400);
    lcdcmd(1);
   lcdcmd(1);
  while(1)
  {
lcdcmd(1);
if(pulse==0)
{
g=g+1;
if(g>4)
{
g=0;
}
}
lcdcmd(0x80);
  msgdisplay("S:");
```

```
  lcdcmd(0x82);
speed = ADC_Read(0);
speed=speed/7;
//speed= map(speed,1023,0,100,0);
convert(speed);
delay(100);
lcdcmd(0xc0);
  msgdisplay("X");
  lcdcmd(0xC1);
x = ADC_Read(1);
convert(x);
lcdcmd(0xc5);
    msgdisplay("Y");
  lcdcmd(0xC6);
y = ADC_Read(2);
convert(y);
delay(200);
      delay(500);
lcdcmd(0x86);
  msgdisplay("T:");
a1 = ADC_Read(3);
f=a1;           //unsigned to float conversion
p=(f*5)/1023;    //equivalent voltage calculation from ADC output
tmpr=p*100;
lcdcmd(0x88);
convert(tmpr);
```

```
    lcdcmd(0xcb);
convert(g);
delay(500);
eeprom_write(0,speed);
eeprom_write(1,tmpr);
eeprom_write(2,g);
if(ed==0)
{
t1=eeprom_read(0);
t2=eeprom_read(1);
t3=eeprom_read(2);
lcdcmd(0x01);
lcdcmd(0x80);
  msgdisplay("s");
 convert(t1);
lcdcmd(0xc0);
 msgdisplay("t");
 convert(t2);
lcdcmd(0xc8);
  msgdisplay("g");
  convert(g);
delay(1000);
}
if(x<200 || y<200 || x>380 || y>380)
{
            lcdcmd(0x01);
```

```
            msgdisplay("ACCIDENT");
delay(300);
GPRS_send1();
}
  }
  }
 void GPRS_send1()
            {
            lcdcmd(0x01);
            msgdisplay("UPLOADING DATA..");
delay(300);
  delay(500);
  sertxs("AT\r\n");
  delay(500);
 sertxs("AT+CPIN?\r\n");//check for sim
  delay(500);
  sertxs("AT+CREG?\r\n"); // checking sim registeration
  delay(500);
  sertxs("AT+CGATT?\r\n");//checking if MS is connected to GPRS
  delay(500);
 sertxs("AT+CIPSHUT\r\n");
  delay(500);
 sertxs("AT+CIPSTATUS\r\n"); // current connection status
  delay(500);
 sertxs("AT+CIPMUX=0\r\n");// start multiconnection
  delay(1000);
```

```
            sertxs("AT+CSTT=\"Idea Internet\"");// APN of the sim
            sertxs("\"\r\n");
  delay(1000);
sertxs("AT+CIICR\r\n");// start wireless connection with GPRS
  delay(1000);
 sertxs("AT+CIFSR\r\n");//get local IP address
  delay(1000);
  sertxs("AT+CIPSPRT=0\r\n");
  delay(2000);
  sertxs("AT+CIPSTART=\"TCP\",\"api.thingspeak.com\",\"80\"\r\n");// start TCP connection
AT+HTTPPARA=\"URL\",\"api.thingspeak.com/update\"
  delay(2000);
sertxs("AT+CIPSEND\r\n");// send data through TCP/UDP connection
  delay(3000);
             sertxs("GET
https://api.thingspeak.com/update?api_key=1CK1TXJN2BZ4OSE7&field1=");//1CK1TXJN2BZ4OS
E7
            converts(speed);
             sertxs("&field2=");
converts(tmpr);
 sertxs("&field3=");
converts(g);
  sertxs("\"\r\n");
  delay(2000);
  //send_1(26);
            sertxc(0X1A);
```

```
  delay(1000);
   sertxs("\"\r\n");
  sertxs("AT+CIPSHUT\r\n");
  delay(1000);
            lcdcmd(0x01);
  //lcd.setCursor(0, 1);
                        msgdisplay("UPLOADED");
   delay(1000);
            lcdcmd(0x01);
}
```