“Vehicle Speed Monitoring System Using Microcontroller”.

PROJECT REPORT

OF MAJOR PROJECT

POSTGRADUATION DIPLOMA IN EMBEDDED SYSTEM AND DESIGN

SUBMITTED BY,

Miss.Naik Namrata.
Miss.Ashwini Raichur.
Miss.Poonam Mahale.
Miss.Rutuja Patil.

September 2022

SUNBEAM INFOTECH HINJEWADI PUNE.

Certificate

This is to certify that the project report entitled "**Vehicle Speed Monitoring System using STM32"**is a bonafide work carried out by,

Miss.Naik Namrata.
Miss.Ashwini Raichur.
Miss.Poonam Mahale.
Miss.Rutuja Patil

Under our supervision, during the year 2022-2023 and submitted to the course coordinator of DESD course, Sunbeam Infotech Hinjewadi Pune in partial fulfillment of the requirements for the award of the Post- Graduation Diploma in Embedded System and Design.

Lab Mentor	**Course Coordinator**
(Mr. Ajay Kumar)	**(Mr. Devendra Dhande)**

UNDERTAKING

We hereby declare that the details furnished above are true and correct to the best of our knowledge and belief and we undertake to inform authorities about any changes therein, immediately. In case any of the above information is found to be false or untrue or misleading or misrepresenting, I am aware that I may be held liable for it.

Sr. No.	Name of Student	Roll No.	Sign
1.	Naik Namrata.	69862	
2.	Ashwini Raichur.	70229	
3.	Poonam Mahale.	70403	
4.	Rutuja Patil.	70467	

Place: Pune.

Date:

INDEX

CHAPTER 1

INTRODUCTION

CHAPTER 01
INTRODUCTION

1.1 Problem Description :

We observe the drivers fatigue driving and vehicle theft activity which causes social real time problem like accidents and many more hazards conditions. We daily see or read such type of activities which are raising the question of our safety and security in both public and private sectors. The graph below shows the road accidents statistics in India from 2016-2020. According to the graph approximately 37 deaths occurs per 100 crashes. So, there is a need of real time monitoring of vehicle which also stores and updates its database in certain situations.

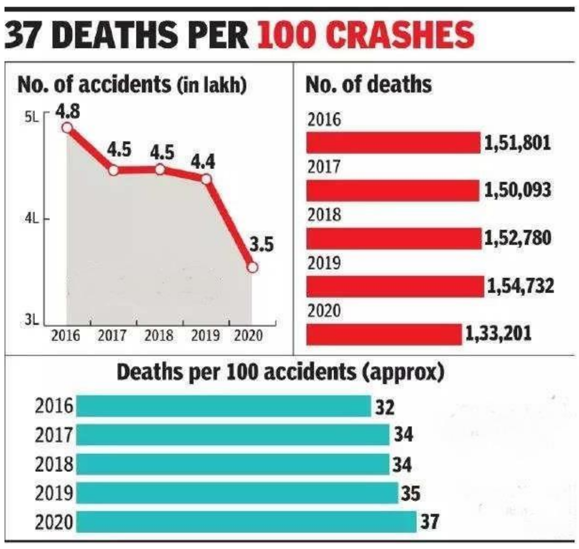


Fig. 1.1 road accidents statistics

1.2 System Objective

- To detect the Vehicle using IR Sensors
- To Calculate the Speed of Detected Vehicle
- To check the Speed Limit of Vehicle
- To alert the Authorities if speed limit is raised above 40 km/hr.

1.3 Overview of Project

This proposed system provides a fully monitoring of the vehicle which is helpful in school zones and hospital areas, it provides accurate arrival time of the vehicle at particular location or stop. In order to reduce man power and saving of money, here the system provides easy Monitoring solution using Embedded STM32 Board and NodeMCU. The proposed system gets speed information of the vehicle using IR proximity sensors is in normal range or exceeds the normal range using NodeMCU. If the speed exceeds the limit the buzzer sets ON to indicate the speed crosses the normal limit.

1.4 Literature Survey

A system for vehicle over speed detection with SMS alert is presented. It consists of a controller designed using Arduino Mega to monitor the location and speed of the vehicle obtained using a GPRS+GPS Quadband Module (SIM908), GSM antenna, GPS antenna and SIM card. If the vehicle presents in any of the defined regions, then the controller compares the speed of the vehicle with maximum allowable speed in that area. If over speeding is detected, a buzzer sound is generated from an active buzzer used in this system to alert the driver regarding exceeding the over speed.

Attention (AT) commands are used to access the SIM908 functionalities. GSM Antenna is required for sending and receiving messages. It is also needed for receiving calls. Max3232 module is used as an interface between Arduino and SIM908 module. The GPS coordinates and speeds of the vehicle which are continuously calculated are stored in a memory card along with the time[1].

A system which can be installed into a vehicle, possibly during manufacturing, to detect if a vehicle is breaching the speed limit and if so, the driver is notified of the fine via an SMS instantly and a copy of the ticket is printed on the administrator and can be mailed to the driver. After switching the device on, the microcontroller initializes all the hardware devices connected to it with the default values and then normal operation continues. After initialization, the timer and the ADC are started. At each ADC interrupt the measured accelerometer value on each of its axis is added to the average value. This average value is calculated at every 1 second, due to the timer interrupt. The average values for each of the axes are used as acceleration values, measured prior to the interrupts, to update the speed of the vehicle regularly. Once this value is updated, the output port for the 8- bit speed value is updated accordingly. The GSM module is used to send periodic updates of the signal strength, speed and location values to the server [2].

The data sent from the GSM module are extracted from the url-encoded parameters. The speed limit at the specified GPS coordinates is acquired from speed-limit API. If a violation occurred the fine amount is calculated based on the speed and the speed limit of the location. Once all the data is available, the database connection is made and inserted into the table of ‘fines’. Once the database is updated, the SMS notification is sent and the PDF is printed with the details of the speeding event. The proposed equipment/device will compare the present position and speed of the vehicle with applicable traffic rules and on occurrence of any violation, will caution the driver in his/her choice of language and will also send the violation information to the supervising authorities.

The availability and precision of worldwide Global Positioning System(GPS) gives more effectiveness and safety for vehicles utilizing highways, roads and mass transit system. The GPS allows the automatic localization of vehicles and navigation systems in today’s widely used vehicles around the world. By combining GPS positioning technology with systems that can display geographic information or systems that can automatically transmit data to screens or computers, a new dimension is achieved in surface transport[3].

A study to design and develop a low-cost system that can accurately detect speed by using an array of active infrared sensors enabled by specialized algorithm and submit violation related data to the data-center. The system consists of (1) a series of an Active Infrared Sensors (from Sensor 1 to Sensor N) to detect vehicles, (2) a Microcontroller to read the sensor’s outputs, enforce the backtracking mechanism, calculate speed, and submit the speed to another microcontroller that works as an access point and camera controller to take a photo of over-speeding vehicle the picture with other related information such as, detected speed, location and date, will be sent to the data center for further processes[5].

An automatic wireless system for monitoring vehicle speed on the road, identify a speeding vehicle and imposing penalty for the speeding offenders. A prototype system has been developed in a laboratory environment to generate random speed data using a mechanical wheel, measure the speed data with a Shimmer wireless sensor and transfer the data wirelessly to a client computer for further analysis. Software has been developed using a Java based socket programming technique to monitor the vehicle speed in a server computer and to send the data associated with a speeding vehicle to a remotely placed client computer. The graphical user interface (GUI) can visually display the speed of a vehicle at any particular time. The functionality of the software has been tested by simulating different traffic scenarios with low and high-speed limits (40 and 60 km/hr respectively). To do that a high or low speed limit can be set in the GUI. The mechanical wheel is run at different speeds and the GUI continuously displays the speed. If the vehicle speed is higher than the set speed limit for the road, the system automatically detects it and generates a report with the time of speeding, vehicle number, vehicle speed etc. to be saved in the client computer in order to take further necessary actions for the speeding offender [6].

CHAPTER 02

COMPONENTS USED

2.1 Hardware Components:

2.1.1 STM32F407G-DISC1 Board:

The STM32F4DISCOVERY Discovery kit allows users to easily develop applications with the STM32F407VG high performance microcontroller with the ARM® Cortex®-M4 – 32 bit core. Based on STM32F407VG, it includes an ST-LINK/V2 or ST-LINK/V2-A embedded debug tool, two ST-MEMS digital accelerometers, a digital microphone, one audio DAC with integrated class D speaker driver, LEDs, push buttons and a USB OTG micro-AB connector. To expand the functionality of the STM32F4DISCOVERY Discovery kit with the Ethernet connectivity, LCD display and more. The STM32F4DISCOVERY Discovery kit comes with the STM32 comprehensive free software libraries and examples available with the STM32Cube package, as well as a direct access to the ARM® embed Enabled™ on-line resources.

In this Project, STM32F407G-DISC1 will act as an master Device. ARM processor will receive the output of the IR Proximity Sensors which would be time in seconds and then using this output speed would be calculated and then forwarded using Serial communication to ESP8266 Module.

STM32F407VG-Disc1 Features:

- Up to 17 timers: up to twelve 16-bit and two 32-bit timers up to 168 MHz, each with up to 4 IC/OC/PWM or pulse counter and quadrature encoder input.
- Memories
    - Up to 1 Mbyte of Flash memory
    - Up to 192+4 Kbytes of SRAM including 64-Kbyte of CCM (core coupled memory) data RAM
    - 512 bytes of OTP memory
    - Flexible static memory controller supporting Compact Flash, SRAM, PSRAM,NOR and NAND memories

- LCD parallel interface, 8080/6800 modes
- Clock, reset and supply management
    - 1.8 V to 3.6 V application supply and I/Os
    - POR, PDR, PVD and BOR
    - 4-to-26 MHz crystal oscillator
    - Internal 16 MHz factory-trimmed RC (1% accuracy)
    - 32 kHz oscillator for RTC with calibration
    - Internal 32 kHz RC with calibration
- Up to 15 communication interfaces
    - Up to 3 × I^{2}C interfaces (SMBus/PMBus)
    - Up to 4 USARTs/2 UARTs (10.5 Mbit/s, ISO 7816 interface, LIN, IrDA, modem control)
    - Up to 3 SPIs (42 Mbits/s), 2 with muxed full-duplex I^{2}S to achieve audio class accuracy via internal audio PLL or external clock
    - 2 × CAN interfaces (2.0B Active)
      SDIO interface

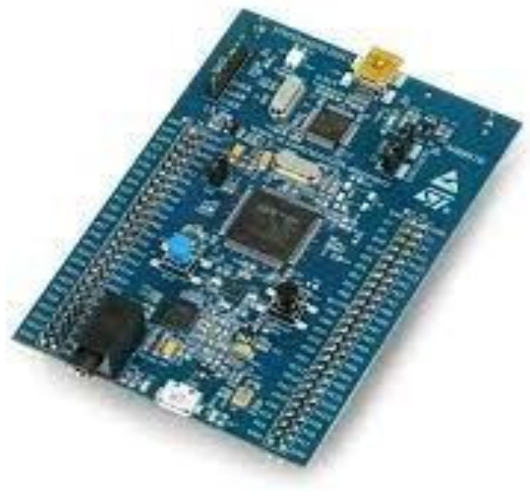


Fig. 2.1.1 STM32F407VG-DISC 1

2.1.2 Node MCU-WiFi Module ESP8266:

The ESP8266 Wi-Fi Module is a self contained SOC with integrated TCP/IP protocol stack that can give any microcontroller access to your Wi-Fi network. The ESP8266 is capable of either hosting an application or offloading all Wi-Fi networking functions from another application processor. Each ESP8266 module comes pre-programmed with an AT command set firmware .The ESP8266 module is an extremely cost effective board with a huge, and ever growing, community.

This module has a powerful enough on-board processing and storage capability that allows it to be integrated with the sensors and other application specific devices through its GPIOs with minimal development up-front and minimal loading during runtime. Its high degree of on-chip integration allows for minimal external circuitry, including the front-end module, is designed to occupy minimal PCB area. The ESP8266 supports APSD for VoIP applications and Bluetooth co-existance interfaces, it contains a self-calibrated RF allowing it to work under all operating conditions, and requires no external RF parts. In this Project, We are going to use ESP8266 module as a slave device which will get the calculated speed of Vehicle and pass it to LCD as well as there will be an entry of the speed into Database.

Features of ESP8266 Module:

- Processor: L106 32-bit RISC microprocessor core based on the Tensilica Xtensa Diamond Standard 106Micro running at 80 MHz[5]
- Memory:
  1) 32 KB instruction RAM
  2) 32 KB instruction cache RAM
  3) 80 KB user-data RAM
  4) 16 KB ETS system-data RAM
- External QSPI flash: up to 16 MB is supported (512 KB to 4 MB typically included)
- IEEE 802.11 b/g/n Wi-Fi
- Integrated TR switch, balun, LNA, power amplifier and matching network WEP or WPA/WPA2 authentication, or open networks
- 17 GPIO pins[6]
- Serial Peripheral Interface Bus (SPI)
- I²C (software implementation)[7]

I²S interfaces with DMA (sharing pins with GPIO)\

- UART on dedicated pins, plus a transmit-only UART can be enabled on GPIO2
- 10-bit ADC (successive approximation ADC)

Fig. 2.1.2 ESP8266 – Node MCU

2.1.3 IR Proximity Sensors:

An IR proximity sensor is a sensor able to detect the presence of nearby object without physical Contact. a proximity sensor often emits an electromagnetic field or a beam of electromagnetic radiation (infrared, For instance) and looks changes in the field or return signal. the object being sensed is often referred as proximity sensor target. Proximity sensors can have high reliability and long functional life because of absence of the mechanical parts and lack of physical contact between the sensor and the sensed object.

These Sensors are also used in machine vibration monitoring to measure the variation in distance between a shaft and its support bearing. In this Project, We are going to use IR Proximity Sensors for sensing the vehicle at the start and end of the Lane.

Fig. 2.1.3 IR Proximity Sensors

2.1.4 LCD Display:

The term LCD stands for liquid crystal display. It is one kind of electronic display module used in an extensive range of applications like various circuits & devices like mobile phones, calculators, computers, TV sets, etc. These displays are mainly preferred for multi-segment light-emitting diodes and seven segments. The main benefits of using this module are inexpensive; simply programmable, animations, and there are no limitations for displaying custom characters, special and even animations, etc. In this project, we are going to Use LCD display for displaying the values of vehicle speed.

Features of LCD 16x2:

The features of mainly include the following: -

- The operating voltage of this LCD is 4.7V-5. this LCD 3V
- It includes two rows where each row can produce 16-characters.
- The utilization of current is 1mA with no backlight
- Every character can be built with a 5×8 pixel box
- The alphanumeric LCDs alphabets & numbers
- Is display can work on two modes like 4-bit & 8-bit
- These are obtainable in Blue & Green Backlight

It displays a few custom generated characters

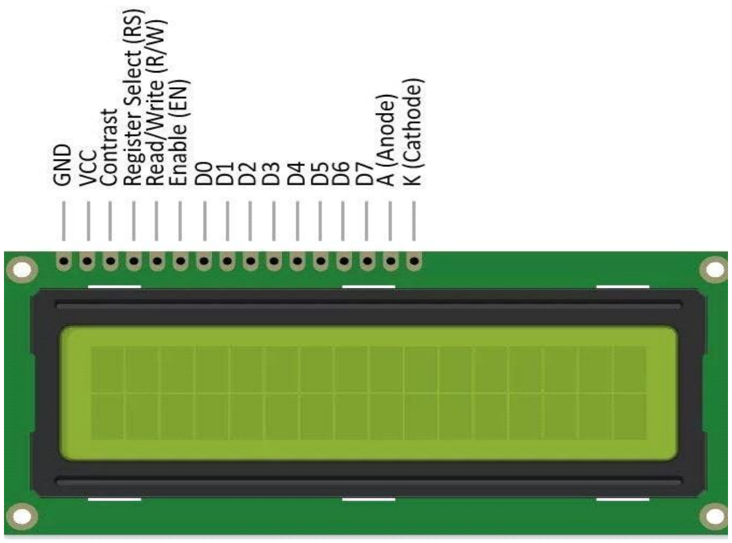


Fig. 2.1.4 LCD Display

2.1.5 I2C Module for LCD display:

This is a RoHS compliant I2C Serial LCD Daughter board that can be connected to a standard 16×2 or 20×4 Character Display Module that supports 4-bit mode. All Character Modules sold on our site support 4-bit mode, and nearly all commercially available 16×2 and 20×4 line character modules support it too.

This board has a PCF8574 I2C chip that converts I2C serial data to parallel data for the LCD display.. The I2C address is 0x3F by default, but this can be changed via 3 solder jumpers provided on the board. This allows up to 3 LCD displays to be controlled via a single I2C bus (giving each one it’s own address)

Features of I2C Module:

- 5V power supply.
- Serial I2C control of LCD display using PCF8574.
- Backlight can be enabled or disabled via a jumper on the board.
- Contrast control via a potentiometer.
- Can have 8 modules on a single I2C bus (change address via solder jumpers)address, allowing.
- Size : 41.6 x 19.2 mm.

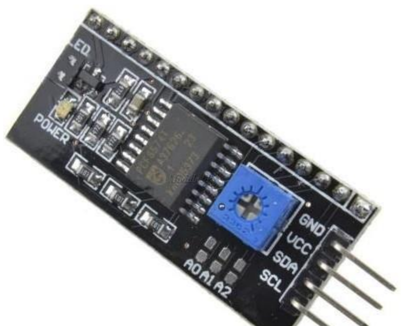


Fig. 2.1.51 I2C Module for LCD Display

2.1.6 Buzzer:

A buzzer or beeper is an audio signaling device, which may be mechanical, electromechanical, or piezoelectric (piezo for short). Typical uses of buzzers and beepers include alarm devices, timers, and confirmation of user input such as a mouse click or keystroke.

Fig. 2.1.6 Buzzer

2.2 SOFTWARE USED:

2.2.1 STM32Cube IDE:

STM32Cube IDE is an all-in-one multi-OS development tool, which is part of the STM32Cube software ecosystem. STM32Cube IDE is an advanced C/C++ development platform with peripheral configuration, code generation, code compilation, and debug features for STM32 microcontrollers and microprocessors. It is based on the Eclipse®/CDT™ framework and GCC toolchain for the development, and GDB for the debugging. It allows the integration of the hundreds of existing plugins that complete the features of the Eclipse® IDE.

STM32Cube IDE integrates STM32 configuration and project creation functionalities from STM32CubeMX to offer all-in-one tool experience and save installation and development time. After the selection of an empty STM32 MCU or MPU, or preconfigured microcontroller or microprocessor from the selection of a board or the selection of an example, the project is created and initialization code generated. At any time during the development, the user can return to the initialization and configuration of the peripherals or middleware and regenerate the initialization code with no impact on the user code.

STM32Cube IDE includes build and stack analysers that provide the user with useful information about project status and memory requirements. STM32Cube IDE also includes standard and advanced debugging features including views of CPU core registers, memories, and peripheral registers, as well as live variable watch, Serial Wire Viewer interface, or fault analyser.

All features of STM32CubeIDE:

- Integration of services from STM32CubeMX:STM32 microcontroller, microprocessor, development platform and example project selection Pinout, clock, peripheral, and middleware configuration Project creation and generation of the initialization code Software and middleware completed with enhanced STM32Cube Expansion Packages
- Based on Eclipse®/CDT™, with support for Eclipse® add-ons, GNU C/C++ for Arm® toolchain and GDB debugger
- STM32MP1 Series: Support for Open STLinux projects: Linux Support for Linux
- Additional advanced debug features including: CPU core, peripheral register, and memory views Live variable watch view System analysis and real-time tracing (SWV)CPU fault analysis tool RTOS-aware debug support including Azure
- Support for ST-LINK (STMicroelectronics) and J-Link (SEGGER) debug probes
- Import project from Atollic® TrueSTUDIO® and AC6 System Workbench for STM32 (SW4STM32)
- Multi-OS support: Windows®, Linux®, and macOS®, 64-bit versions only

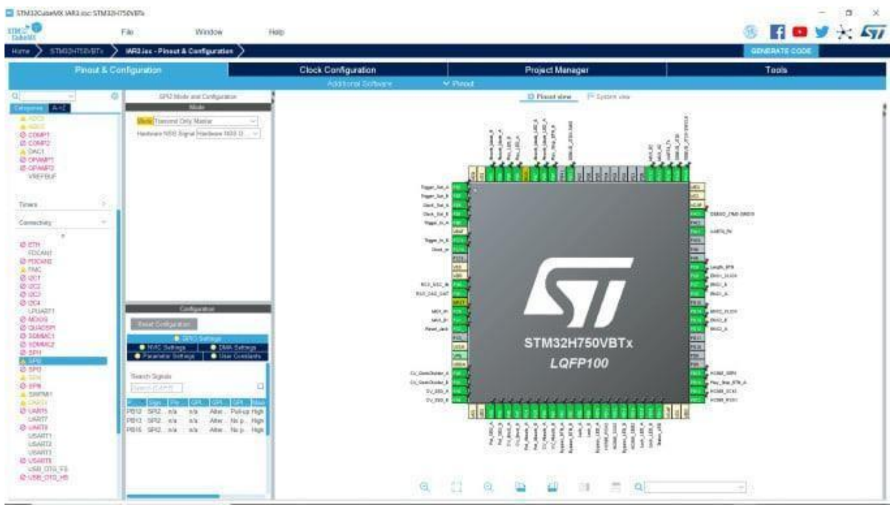


Fig. 2.2.1 STM32CubeIDE

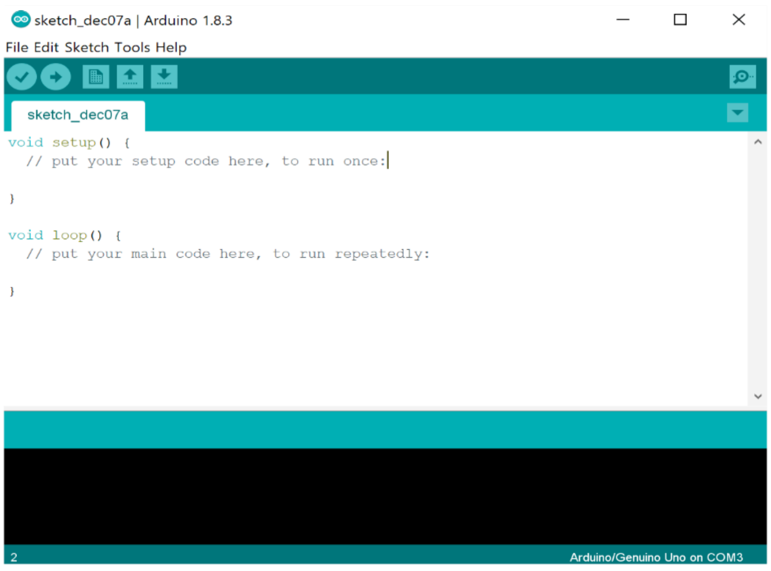
2.2.2 Arduino IDE:

The open-source Arduino Software (IDE) makes it easy to write code and upload it to the board. This software can be used with any Arduino board. The program for Arduino is developed in any language with a compiler that generates compatible code for the processers in the microcontroller. The development has to happen externally to the Arduino board using an integrated development environment (IDE) exclusive to Arduino and it runs on cross-platform OS viz., Windows, Linux, and Macs. It mainly supports Java language and C and C++ with some special conditions attached to it.

The Arduino is more than just hardware, however, it's also software. There's this thing called the Arduino IDE (Integrated Development Environment). This is a software application that you download onto your computer, and then you use it to program the Arduino boards.

It's completely free software, and it's pretty easy to use. It looks a lot like a text editor. The Arduino IDE is where you write your code that actually gets loaded onto the Arduino board itself.

The Arduino code that you write is called a sketch. The Arduino code itself is basically a derivative of the C and C++ programming languages, but with some Arduino-specific functions and structure and also need to install hardware specific libraries. These libraries help to run code on hardware, so if you program an Arduino, you're basically programming in C and C ++ programming languages.


sketch_dec07a | Arduino 1.8.3
File Edit Sketch Tools Help
sketch_dec07a
2
Arduino/Genuino Uno on COM3

```
void setup() {
  // put your setup code here, to run once:

}

void loop() {
  // put your main code here, to run repeatedly:

}
```

Fig. 2.2.2 Arduino ID

CHAPTER 03

WORKING OF PROJECT

3.1 Block Diagram :

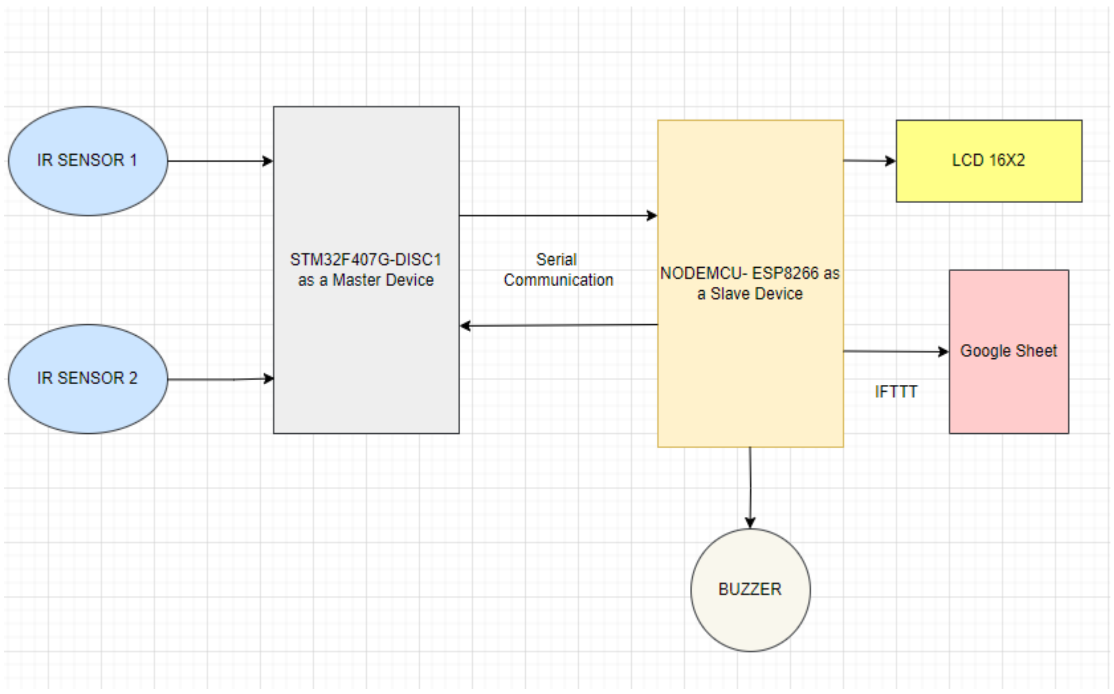


Fig 3.1.Block Diagram

3.1.1 Hardware Connections :

i. Output pin of IR sensor 1 is connected to the PA0 pin and Output pin of IR sensor 2 is connected to PA1 pin of STM32 Board.

ii. PA2 pin of STM32 Board(Master device) is connected to Tx pin and PA3 pin of STM32 Board is connected to Rx pin of ESP8266 Module(slave device).

iii. SCL pin of I2C module is connected to D1 pin SDA pin I2C module is connected to D2 pin of ESP8266 Module.

iv. Positive terminal of Buzzer is connected to D3 pin of ESP8266 Module.

3.1.2 Circuit Diagram:

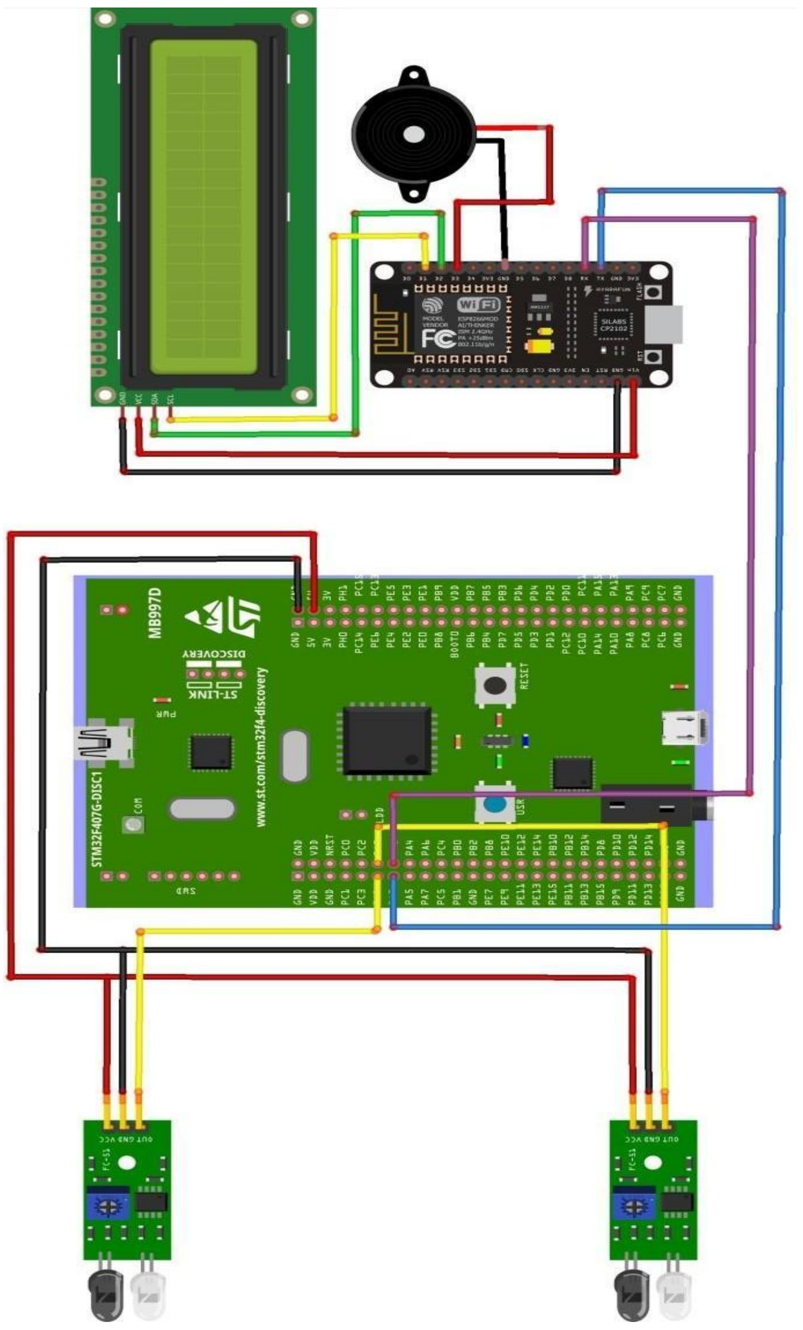


Fig 3.1.2.circuit diagram

3.2 Flowchart:

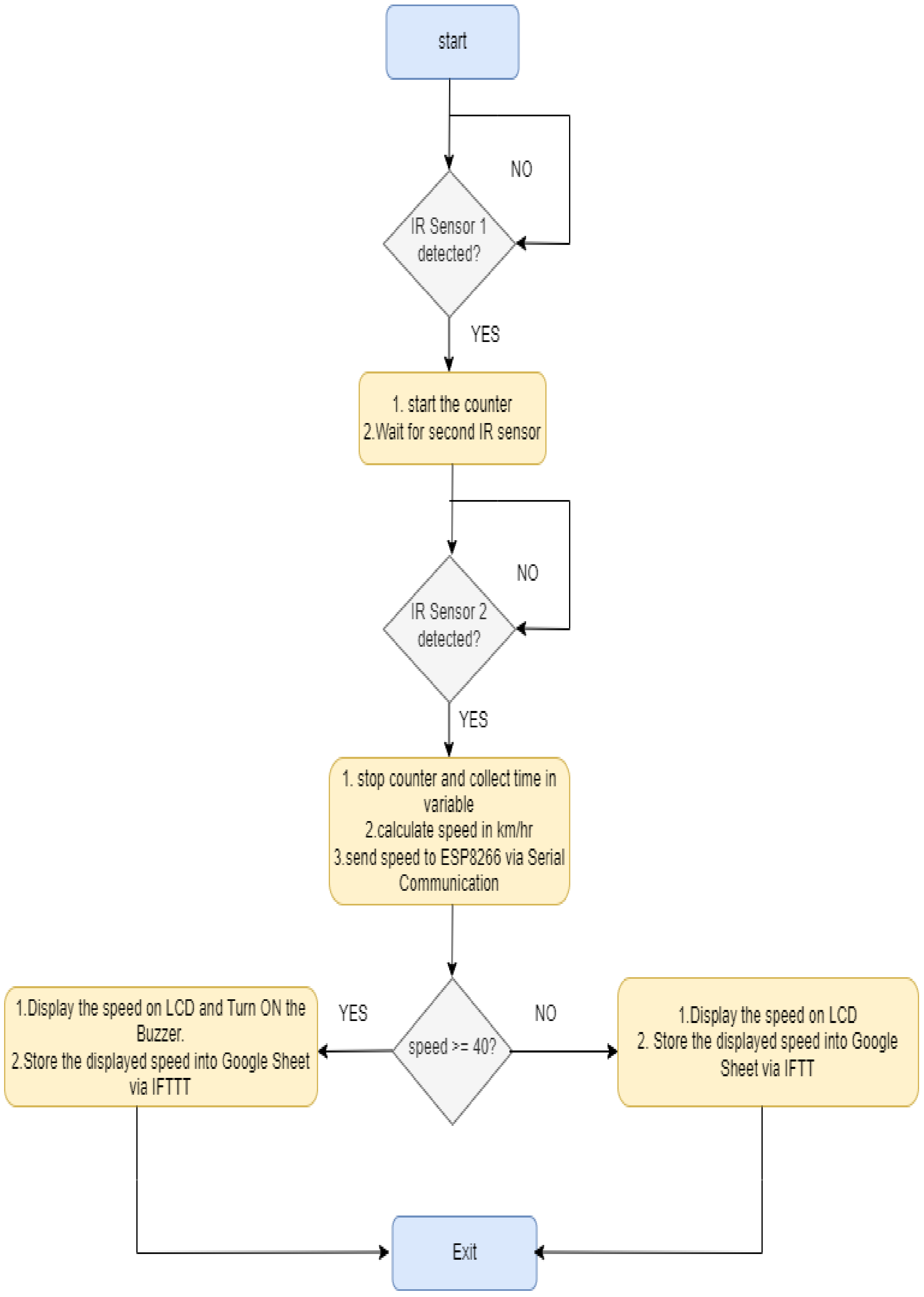

start
NO
IR Sensor 1 detected?
YES
1. start the counter
2.Wait for second IR sensor
NO
IR Sensor 2 detected?
YES
1. stop counter and collect time in variable
2.calculate speed in km/hr
3.send speed to ESP8266 via Serial Communication
YES
speed >= 40?
NO
1.Display the speed on LCD and Turn ON the Buzzer.
2.Store the displayed speed into Google Sheet via IFTTT
1.Display the speed on LCD
2. Store the displayed speed into Google Sheet via IFTT
Exit

Fig 3.2.flowchart of project

3.3 Working :

The heart of this system is STM32f407G-DISC1 which is used for decision making. IR Proximity Sensors which are used for vehicle detection are interfaced to STM32 which works as a Master. The state of IR Proximity sensors is as follows :

IR Sensor = 0 ; indicates that the vehicle is detected
IR Sensor = 1 ; indicates that the vehicle is not detected.
IR sensor 1 and IR sensor 2 are placed at a fixed distance 50 m.

STM32F407G-DISC1 is responsible for starting the timer . IR Sensor 1 detects the vehicle it will turn on the timer once a vehicle is detected by the IR Sensor 2 the Master will stop the timer and collect the timer value in seconds for the vehicle moving in forward direction and vice-versa for the vehicle moving in reverse direction. It then calculates the speed of vehicle by the given formula;
Speed = Distance/ Time;
For conversion of speed into km/hr formula used is as follows:
speed = speed * 18/5;
Further this converted speed is forwarded to NodeMCU using Serial Communication.

The NodeMCU is used to collect the converted speed from Master using serial communication. NodeMCU checks the speed limit if the speed limits the threshold value(considering threshold value as 40km/hr) then it sets the Buzzer on and forwards the speed value to LCD and stores the value into google sheet. In case of speed value less than threshold value it only forwards the speed value to LCD.

3.3.1 UART Protocol :

UART or Universal Asynchronous Receiver Transmitter is a serial communication device that performs parallel – to – serial data conversion at the transmitter side and serial – to – parallel data conversion at the receiver side.

HOW UART WORKS ?

In UART Serial Communication, the data is transmitted asynchronously i.e. there is no clock or other timing signal involved between the sender and receiver. Instead of clock signal, UART uses some special bits called Start and Stop bits.
These bits are added to the actual data packet at the beginning and end respectively. These additional bits allows the receiving UART to identify the actual data.

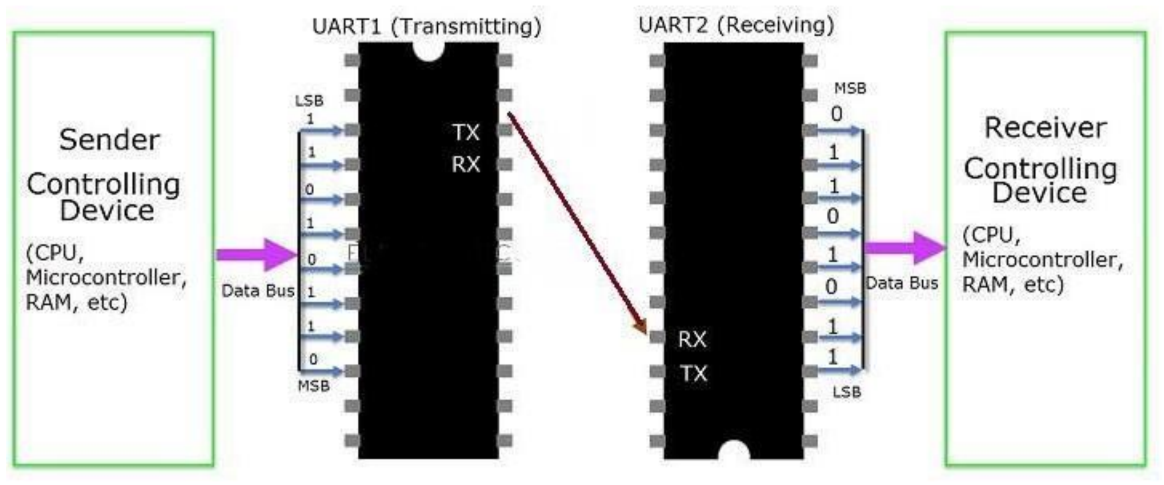


Fig 3.3.1.UART working

The image above shows a typical UART connection. The transmitting UART receives data from the controlling device through the data bus. The data received by the transmitting UART from the data bus is parallel data.

To this data, the UART adds Start, Parity and Stop bits in order to convert it into a data packet. The data packet is then converted from parallel to serial with the help of shift register and is transmitted bit – by – bit from the TX pin.

The receiving UART receives this serial data at the RX pin and detects the actual data by identifying the start and stop bits. Parity bit is used to check the integrity of the data.

Up on separating the start, parity and stop bits from the data packet, the data is converted to parallel data with the help of shift register. This parallel data is sent to the controller at the receiving end through a data bus

Structure of Data Packet or Frame :

The data in UART serial communication is organised in to blocks called Packets or Frames. The structure of a typical UART Data Packet or the standard framing of the data is shown in the following image.

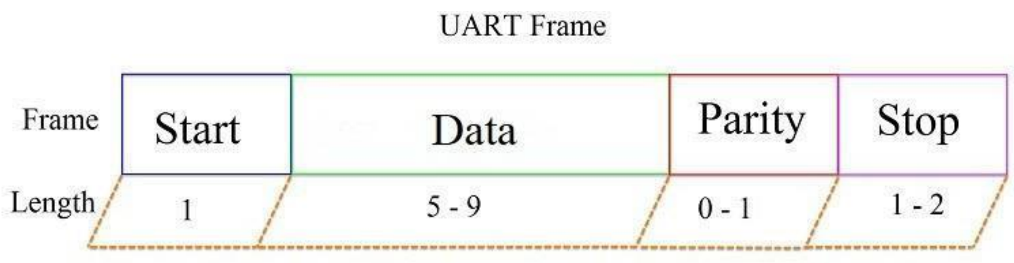


Fig 3.3.2.UART frame

Start Bit: Start bit is a synchronisation bit that is added before the actual data. Start bit marks the beginning of the data packet. Usually, an idle data line i.e. when the data transmission line is not transmitting any data, it is held at a high voltage level (1).

In order to start the data transfer, the transmitting UART pulls the data line from high voltage level to low voltage level (from 1 to 0). The receiving UART detects this change from high to low on the data line and begins reading the actual data. Usually, there is only one start bit.

Stop Bit: The Stop Bit, as the name suggests, marks the end of the data packet. It is usually two bits long but often only on bit is used. In order to end the transmission, the UART maintains the data line at high voltage (1).

Parity Bit: Parity allows the receiver to check whether the received data is correct or not. Parity is a low – level error checking system and comes in two varieties: Even Parity and Odd Parity. Parity bit is optional and it is actually not that widely used.

Data Bits: Data bits are the actual data being transmitted from sender to receiver. The length of the data frame can be anywhere between 5 and 9 (9 bits if parity is not used and only 8 bits if parity is used). Usually, the LSB is the first bit of data to be transmitted (unless otherwise specified).

Advantages of UART

- Requires only two wires for full duplex data transmission (apart from the power lines).
- No need for clock or any other timing signal.
- Parity bit ensures basic error checking is integrated in to the data packet frame.

Disadvantages of UART

- Size of the data in the frame is limited.
- Speed for data transfer is less compared to parallel communication.
- Transmitter and receiver must agree to the rules of transmission and appropriate baud rate must be selected.

CHAPTER 04

SOFTWARE IMPLEMENTATION

4.1 STM32 Code Logic:

step 1 : Initializing time variable to zero and declaring the count, new Time variable.

step 2 : keeping distance constant with value 50 in meters.

step 3 : declaring speed and old speed variable with float data type. using character str as a buffer of size 12 bytes.

step 4 : creating a function for calculation of speed with Time parameter of type uint8_t and return type as float.

step 5 : for vehicle moving in forward direction

a) checking if PA0==0 and PA1==1 then

b) start the counter i.e increment the timer variable followed by delay of 100msec till PA1 ==0

c) Calling speed Cal function, getting the string length in count variable and then transmitting the calculated speed in form of string to the NodeMCU using UART_Transmit function.

d)set the speed and time variable value to zero.

step 6 : for vehicle moving in reverse direction

a) checking if PA0==1 and PA1==0 then

b) start the counter i.e increment the timer variable followed by delay of 100msec till PA0 ==0

c) Calling speedCal function, getting the string length in count variable and then transmitting the calculated speed in form of string to the NodeMCU using UART_Transmit function.

d)Set the speed and time variable value to zero.

step 7 : defining speed calculation function.

```
float speedCal(uint8_t Time)
{
        newTime = Time * 100.0;                 // convert counter into milisec
        newTime = newTime/1000.0;               //convert milisec into sec
        speedOld = (distance/newTime);          //calculate speed
        speedOld = speedOld * 3.6;       //(18/5) convert speed m/s into km/hr
        return speedOld;                        //return calculated speed
}
```

4.2 NodeMCU Code Logic :

step 1 : declaring a character buff of size 8 bytes to collect speed from stm32, declaring volatile byte index to the buffer and speed variable of type float.

step 2 : defining wifi credentials i.e ssid and password required to send data to google sheet.

step 3 : defining the key and event name of IFTTT

step 4 : void setup()

{

a) start serial communication using serial.begin with baud rate 9600

b) calling Wifi.disconnect() function to disconnect the previously connected wifi if any.

c) calling wifi.begin() this would connect to wifi.

d) while the wifi.status() is not connected it would wait for 300miliseconds and then again check for wifi connected status.

e) once the wifi is connected Print the IP address of node MCU using Wifi.localIP() on the serial monitor.

f) turn on the LCD , clear the LCD screen and set the Buzzer connected to D3 pin to OUTPUT mode.

}

```
step 5:
void loop()
{
a) check if the byte is received from stm32 using Serial.available() >0
condition. if condition is true then read that byte using Serial.read();
b) if index is less than size of buffer then save the data in next index in the
array buff.
c) for normal speed condition check for end of word by comparing it with "\n" if
condition is true then convert the speed which was in string form into float
using atof() function.
d) if speed <=40 then print speed value and pass string named "normal speed"
to LCD. Call google_sheet(); reset index value to Zero.
e) if speed >40 then print speed value and pass string named "High speed"
to LCD. Set the buzzer pin high for 1000ms and then turn off the buzzer.
Call google_sheet(); reset index value to Zero.
}
```

```
step 6 :
google_sheet()
{
a) In IFTTT we are creating query. Query is created using wifi name,
wifi password, key and event name.
b) Once the query is triggered, data is send to the google sheet.
}
```

CHAPTER 05

APPLICATIONS & ADVANTAGES

5.1 APPLICATIONS :

This Vehicle speed monitoring system can be implemented at

- Highways (National Highways, State level Highways).
- Inside university campus areas or inside any company’s premises.
- School zones.
- Near Hospital areas.
- Heavy traffic areas in cities

5.2 ADVANTAGES :

- Used to measure the speed of the vehicle in order to avoid accidents.
- Consumes less power.
- Simple circuit connections.
- Stores the speed data globally.
- Reliability.

CHAPTER 06
RESULTS

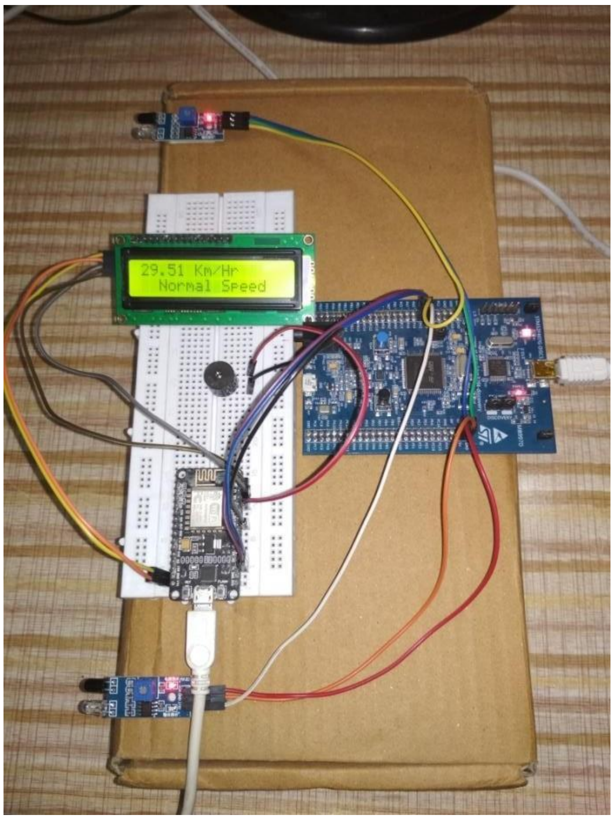


Fig 6.1. Image shows the normal speed of the vehicle

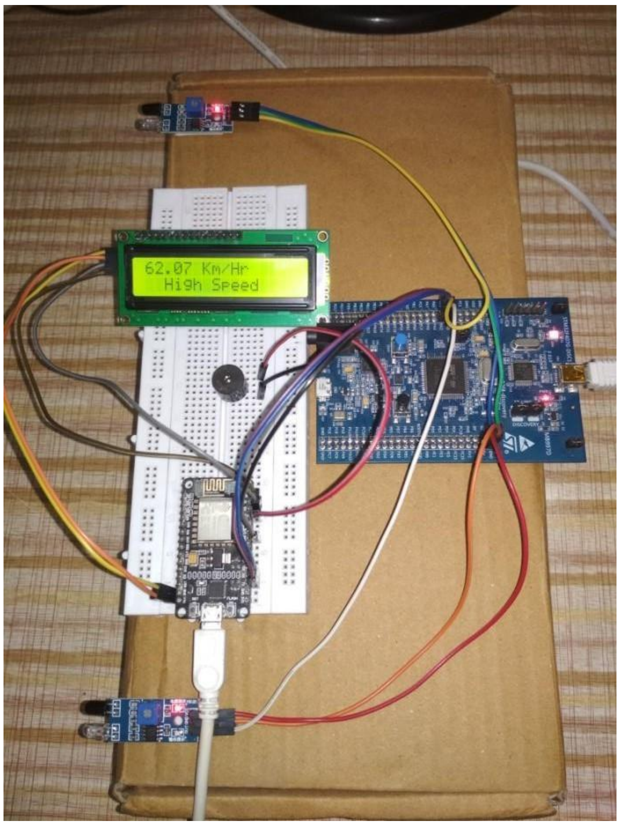


Fig 6.2. Image shows the high speed of the vehicle which turns on the buzzer

CHAPTER 07
CONCLUSION & FUTURE SCOPE

7.1 Conclusion :

The main objective of this project, which is design and construction of a vehicle speed monitoring system was achieved. From the results obtained, the system can be seen to effectively detect and display the speed of a moving vehicle. This will go a long way in reducing accidents on roads, curb the loss of properties attributed to accidents and in earnest save lives. This system is applicable in car speed detection in low traffic areas viz; Hospitals, School Campuses, rural areas and so on.

7.2 Future Scope :

- Can be used to track the location of vehicle by using GPS Module.
- Can maintain records of insurance validity of vehicle, owner details, etc.
- Can be used to send the fine receipt to the owner who crosses the speed limit by identifying the vehicle using cameras.

REFERENCES

1. Ravi Kishore Kodali and Sairam, M Department of Electronics and Communication Engineering National Institute of Technology, Warangal WARANGAL, INDIA. **“Over Speed Monitoring System”**.
2. P.R. Kambadkone, G.P. Hancke and T.D. Ramotsoela Department of Electrical, Electronic and Computer Engineering University of Pretoria Tshwane**. “Real Time Speed Detection and Ticketing System”**.
3. Zakaria Khan Department of Aeronautics & Astronautics Institute of Space Technology Islamabad, Pakistan Zakisss@hotmail.com. Ayesha Department of Aeronautics & Astronautics Institute Technology Islamabad, Pakistan Ayeshashafiqueaz@gmail.com. **“Wireless Speed Monitoring System using GNSS Technology”.**
4. Iszaidy1A.Alias2,R.Ngadiran3,R.B.Ahmad4,M.I.Jais5,D.Shuhaiza6 1,2,3,4,5,6affiliations Embedded, Network and Advance Computing (ENAC) School of Computer and Communication Engineering University Malaysia Perlis Pauh Putra, Perlis, Malaysia**. “Video Size Comparison for Embedded Vehicle Speed Detection & Travel Time Estimation System by Using Raspberry Pi”**.
5. Butare Jimmy Damascene and Ryosuke Okuda Department of Information Systems Graduate School of Information Technology, Kobe Institute of Computing, Kobe, Japan**. “Low-cost Speed Limit Monitoring System for Developing Countries Using a Series of Active Infrared Sensors ”**.
6. M. Ahsan*, J. McManis** and M. S. J. Hashmi* *School of Mechanical and Manufacturing Engineering, Dublin City University, Dublin, Ireland **School of Electronic Engineering, Dublin City University, Dublin, Ireland. **“Prototype System Development for Wireless Vehicle Speed Monitoring”**.