

Development of Accident Avoidance System for Vehicles

**This Report Presented in partial fulfillment of the requirements for the
Award of Degree of
Bachelor of Science in Electrical and Electronic Engineering**

Submitted By

Shekh Md Rokonzaman ID:123-33-1189

Ripon Chandra Barman ID:143-33-2290

Supervised By

Mr. Md. Dara Abdus Satter

Assistant Professor

Department of EEE

Daffodil International University



DEPARTMENT OF ELECTRICAL AND ELECTRONIC ENGINEERING

FACULTY OF ENGINEERING

DAFFODIL INTERNATIONAL UNIVERSITY

DHAKA, BANGLADESH

December 2018

TO
OUR BELOVED PARENTS
&
HONOURABLE SUPERVISER
Mr. Md. Dara Abdus Satter

APPROVAL

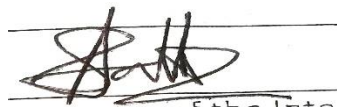
In this Project entitled “**Development of Accident avoidance System for vehicles**” is done by Shajedul Islam, ID:122-33-1080, Shekh Md Rokonuzzaman, ID: 123-33-1189, and Ripon Chandra Barman, ID: 143-33-2290 in EEE program of Daffodil International University, under my supervision. This work has been carried out by them in the laboratories of the Department of Electrical and Electronic Engineering was presented to the audience of the Exam Committee on January 2018 and has been accepted as satisfactory.

Signature of the candidates

Shekh Md Rokonuzzaman
ID #: 123-33-1189

Ripon Chandra Barman
ID #: 143-33-2290

Countersigned



Mr. Md. Dara Abdus Satter

Assistant Professor

Department of Electrical and Electronic Engineering

Faculty of Engineering

Daffodil International University

Certification

This is to certify that this project and thesis entitled “**Development of Accident avoidance System for vehicles**” is done by the following students this work has been carried out by them in the laboratories of the Department of Electrical and Electronic Engineering under the Faculty of Engineering of Daffodil International University in partial fulfillment of the requirements for the degree of Bachelor of Science in Electrical and Electronic Engineering. The presentation of the work was held on December, 2018.

CONTENTS

List of Figures		vi
List of Tables		vii
Acknowledgment		viii
Abstract		ix
Chapter 1: INTRODUCTION		1-3
1.1	Introduction	01
1.2	Aim of the project	01
1.3	Scopes	02
1.4	Methodology	02
1.5	Organization of the Report	02
1.6	Project Outline	03
Chapter 2: SYSTEM REVIEWS		04-06
2.1	Introduction	04
2.2	General Block Diagram	04
2.2.1	Block Diagram Description	05
2.3	Circuit Diagram	05
2.3.1	Working Process of our Circuit	05
2.4	List of Component used in circuit	06
2.5	Conclusion	06
Chapter 3: COMPONENT DESCRIPTION		07-24
3.1	Introduction	07
3.2	Description of Ultrasonic Sensor	07
3.2.1	Working Principle of Ultrasonic Sensor	08
3.2.2	Pin Description of Ultrasonic Sensor	08
3.2.3	Applications	09
3.3	Description of 433MHz RF Module	09

3.3.1	RF Transmitter	09
3.3.2	RF Receiver	10
3.3.3	Pin Definition	11
3.4	Description of l293D	12
3.5	Basic description of controller unit	14
3.5.1	Technical specification of Arduino UNO	14
3.5.2	Description of microcontroller	15
3.5.3	Block Diagram of ATmega328p	16
3.5.4	Pin configuration on ATmega328p	17
3.5.5	Pin description	17
3.6	Basic Arduino Software	19
3.6.1	Features of Arduino software	20
3.7	Wires	22
3.8	Cost Analysis	23
3.8.1	Cost Sheet	24
3.9	Conclusion	25
Chapter 4: SOFTWARE ANALYSIS		26-29
4.1	Introduction	26
4.2	Description of our Software	26
4.3	Flow Chart Diagram	28
4.3.1	Program Explanation	29
Chapter 5: HARDWARE IMLEMENTATION		30-34
5.1	Introduction	30
5.2	Interfacing of Ultrasonic Sensor	30
5.3	Interfacing with RF Module	31
5.4	Interfacing with L293D	32
5.5	Battery Interfacing	33
5.6	System Operation	34
5.7	Conclusion	34
Chapter 6: RESULTS AND DISCUSSION		35-37
6.1	Introduction	35
6.2	Experimental Setup	35
6.3	Result	36

6.4	Advantages	36
6.5	Disadvantages	37
6.6	Conclusion	37
Chapter 7:	CONCLUSION	38-50
7.1	Conclusion	38
7.2	Application	38
7.3	Limitations of the works	39
7.4	Future Work	39
	References	40
	Appendix A	41

LIST OF FIGURES

Figure #	Figure Caption	Page #
2.1	General Block Diagram	04
2.2	Circuit Diagram	05
3.1	Ultrasonic Sensor	07
3.1.1	Working Principle of Ultrasonic Sensor	08
3.1.2	Pins of Ultrasonic Sensor	09
3.2	RF Transmitter	10
3.2.1	RF Receiver	11
3.2.2	TX and RX module	12
3.3	L293D	13
3.3(a)	Arduino Nano	14
3.3(b)	Arduino Nano	15
3.3.1	Block Diagram of ATmega328p	16
3.3.2	Pin Configurations of ATmega328p	17
3.5	Arduino Software	19
3.6(a)	FtoM Jumper Wire	23
3.6(b)	MtoF Jumper wire	23
3.6.1	220v carried wire	23
4.1	Software platform	27
4.2	Compiling Window	27
4.3	Flow chart of our system	28
5.1	Interfacing of Ultrasonic Sensor.	30
5.2	Interfacing with RF Transmitter	31
5.2.1	Interfacing with RF Receiver	32
5.3	Interfacing with L293D	32
5.4	Interfacing with Battery	33
6.1	Setup of our system	35

LIST OF TABLES

Table #	Table Caption	Page #
2.5	List of Components used in Circuit	6
3.2.2	Pin Description of Pulse Sensor	10
3.4.1	Technical Specification Arduino Nano	14
3.10	Cost Sheet	25
5.2	Interfacing of Ultrasonic Sensor	31
7.2	Applications	38

ACKNOWLEDGEMENT

First of all, I give thanks to Allah or God. Then I would like to take this opportunity to express my appreciation and gratitude to my project supervisor **Mr. Md. Dara Abdus Satter**, of **Department of EEE** for being dedicated in supporting, motivating and guiding me through this project. This project can't be done without his useful advice and helps. Also thank you very much for giving us opportunity to choose this project.

Apart from that, I would like to thank my entire friends for sharing knowledge; information and helping us in making this project a success. Also, thanks for lending me some tools and equipment's.

To my beloved family, I want to give them my deepest love and gratitude for being very supportive and also for their inspiration and encouragement during my studies in this University.

ABSTRACT

This project, we discuss a new technique on that vehicle technology, we explained how to keep 10-meter distance between one vehicle and another vehicle, or another object on this vehicle doesn't crash or cause any traffic problem in vehicle raining time. The aim of the project is avoiding to accidents mainly due to not knowing the following distance is 10m between one vehicle to another vehicle. In this project we proposed system comprises an idea of having safety while reversing a vehicle, detects any object within the following distance, and displays the distance between one vehicle and another vehicle to the driver using data. We have used ultrasonic sensors to detect any vehicle on both front and back side of our vehicle. This system is also used in large crane which is mainly operated in harbor area. If the car reaches 10-meter, car was stop and other side under 10m then stop on car. We used 4 ultrasonic sensors when any sensor found object then decide abide this side. On this distance is also indicated to the vehicle driver. In this project system, the safety is maintained on crowded areas and in vehicle reversing process.

CHAPTER 1

INTRODUCTION

1.1 Introduction

The industry strategy for automotive safety systems has been evolving over the last 20 years. Initially, individual passive devices and features such as seatbelts, airbags, knee bolsters, crush zones, etc. was developed for saving lives and minimizing injuries when an accident occurs. Later, preventive measure such as improving visibility, headlights, windshield wipers, tire traction, etc. were deployed to reduce the probability of getting into an accident. Now we are at the stage of actively avoiding accidents as well as providing maximum protection to the vehicle occupants and even pedestrians. The systems that are under intense development include collision avoidance systems. In this project we concentrate on advanced ideas such as pre-crash sensing, an ultrasonic sensor is used to sense the object in front of the vehicle and gives the signal to the microcontroller unit. Based on the signal received from the ultrasonic sensor, the microcontroller unit sends a signal to the braking unit for applying the brake automatically.

1.2 Aim of the Project

In order to accomplish the overall of this project, the following are aim of the project:

1. To design obstacle sensor system.
2. Develop an algorithm that combines sensor strengths and limits sensor shortcomings
3. Ensure the system responds in real time.
4. Identify objects that are potential threats

1.3 Scopes

The scope of the project entails designing, programming and implementing a vehicle collision avoidance system that will be able to stop the vehicle before it hits an obstacle. A program is to be written to make sure the vehicle responds in real time. The project will be implemented using a toy car.

1.4 Methodology

This research will be dedicated to attempt alternative solution for this known problem by developing low cost domestic anti-collision warning system model that would be mounted on the existing car models and alert the driver in danger zone.

Therefore, rather than putting aside inbuilt active safety system development to the car manufacturers, the user shall find ways to solve the problem by developing domestic active safety system model that would be developed later to be fitted to road vehicle despite their model and year of make. This initiated the writer of this manuscript to contribute his share by constructing a model that will enable to conduct further research for finding alternative solution to minimize this life-threatening menace of vehicle accident in Kenya.

1.5 Organization of the Report

The industry strategy for automotive safety systems has been evolving over the last 20 years. Initially, individual passive devices and features such as seatbelts, airbags, knee bolsters, crush zones, etc. was developed for saving lives and minimizing injuries when an accident occurs. Later, preventive measure such as improving visibility, headlights, windshield wipers, tire traction, etc. were deployed to reduce the probability of getting into an accident. Now we are at the stage of actively avoiding accidents as well as providing maximum protection to the vehicle occupants and even pedestrians. The systems that are under intense development include collision avoidance systems.

1.6 Project Outline

This Project is organized as follows:

Chapter 2: System Reviews

In this project we are going to make an Accident avoidance System for vehicles using Arduino that will detect the object using the sonar Sensor and will show the readings.

Chapter 3: Component Description

System hardware design composed of Arduino Nano, Ultrasonic Sensor, I293D, 433 MHz, Some Wires, Arduino compiler, Android apps.

Chapter 4: Software Analysis

In this chapter we discuss about software use and implementation on microcontroller in this project we use Arduino UNO and software Arduino IDE.

Chapter 5: Hardware Implementation

This is one of the most important chapters of this report. In this chapter we will show our completed project's outlook that means Connection lawyer & operation representation of our project

Chapter 6: Results and Discussion

It finds wide application due to its features and low power. In this chapter we will discuss about tests & results in our "Accident avoidance System for vehicles" project.

CHAPTER 2

SYSTEM REVIEWS

2.1 Introduction

In this project we are going to make an Accident avoidance System for vehicles using Arduino that will detect the object using the sonar Sensor and will show the readings in distance cm on the LCD connected to it, so that object distance can be monitored from anywhere. In this project one by one discuss about this system and also discuss about risk on this project. On this project first work ultrasonic sensor work and read data and sensor data send microcontroller and microcontroller discuss stop or go. When sensor found data upper 10mm then microcontroller command motor driver motor driver control car motor, when motor driver found command go or stop then motor driver command go or stop.

2.2 General Block Diagram

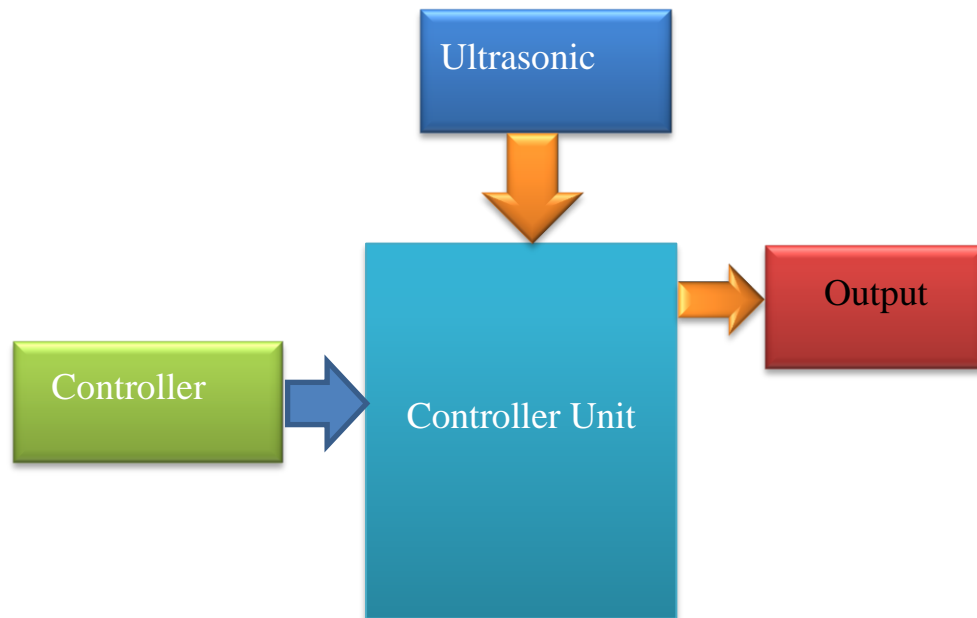


Fig. 2.1: General Block Diagram

2.2.1 Block Diagram Description:

The block diagram of Accident avoidance System for vehicles as shown in fig.2.1, using the Arduino Arduino Nano, we can operate the ultrasonic when the dedicated command is available, then ultrasonic sensor sends a digital data output to control unit. At this time the control units (Arduino Nano) will send commands to result data on output data.

2.3 Circuit Diagram

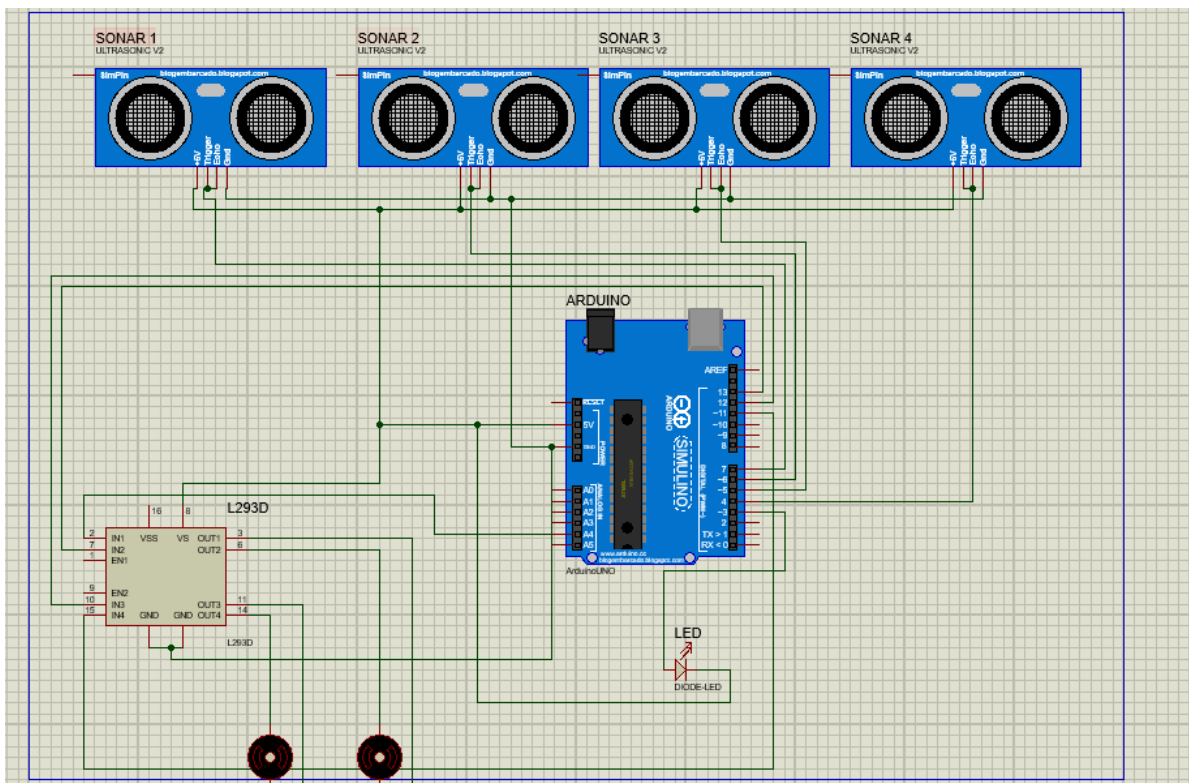


Fig. 2.2: Circuit Diagram

In this circuit diagram we show that Arduino Uno to connected ultrasonic sensor and connected to motor driver IC.

2.3.1 Working Process of our Circuit:

The ultrasonic Sensor Control car comes with a free application called “Accident avoidance System for vehicles”. This application controls the various appliances connected to our Arduino Uno, ultrasonic Sensor and L293D. When the ultrasonic Sensor found some data than this data sent to our Arduino. The Arduino finds out which signal was sent and compares it to the predefined signals assigned for each appliance. When it identifies that signal, the Arduino activates the relay hooked up to its digital pin by passing 5V through it. Thus, the data send Arduino using by Digital Data. When ultrasonic Sensor send data and same time Arduino send this data on send ON or OFF. Finally, we saw the ultrasonic Sensor data in this our result on this project.

2.4 List of Components used in Circuit:

No	Component Name	Quantity	Used
1.	Ultrasonic Sensor	04	To direct Data Send
2.	Microcontroller: - ATmega328P (Arduino Nano)	02	To Control the System.
3.	LCD Display	01	Showing Output.
4.	Chassis 2 Wheel	01	Car Body
5.	l293D	01	Motor control
6.	FS1000A 433MHz RF Module	01	communication
7.	Arduino compiler		To compile code.
8.	Battery	02	Power Supply
9.	Wires	10	To connection.
10.	Switch	01	ON/OFF
11.	Capacitor	03	

2.5 Conclusion:

The Accident avoidance System for vehicles which used in many applications because of its portable it's uses everywhere.

CHAPTER 3

COMPONENT DESCRIPTION

3.1 Introduction

System hardware design composed of Arduino Nano, Ultrasonic Sensor, 1293D, 433 MHz, Some Wires, Arduino compiler, Android apps. In this chapter we will discuss about component description, features, working procedure and cost analysis of our all component.

3.2 Description of Ultrasonic Sensor

Pulse Sensor Amped is a plug-and-play heart-rate sensor for Arduino and Arduino compatibles. It can be used by students, artists, athletes, makers, and game & mobile developers who want to easily incorporate live heart-rate data into their projects. Pulse Sensor adds amplification and noise cancellation circuitry to the hardware. It's noticeably faster and easier to get reliable pulse readings. Pulse Sensor Amped works with either a 3V or 5V Arduino.



Fig. 3.1: Ultrasonic Sensor

3.2.1 Working Principle of Ultrasonic Sensor:

In this Ultrasonic Sensor working on used for measurement of physical quantities on ultrasonic waves on this remote. Ultrasonic wave doesn't have capability on human ear. In this ultrasonic sensor module distance 2cm to 400 cm distance capability and also, it's easily connector to microcontroller. In this Ultrasonic Sensor power consumption is low so it is suitable for any robotic and automatic control unite. In this Ultrasonic Sensor work frequency rate 40khz. If the user is interested in calculating the distance between the object and the transmitter, the sensor module is configured so to do it. The result of the calculation is a pulse whose width is related to the measured distance

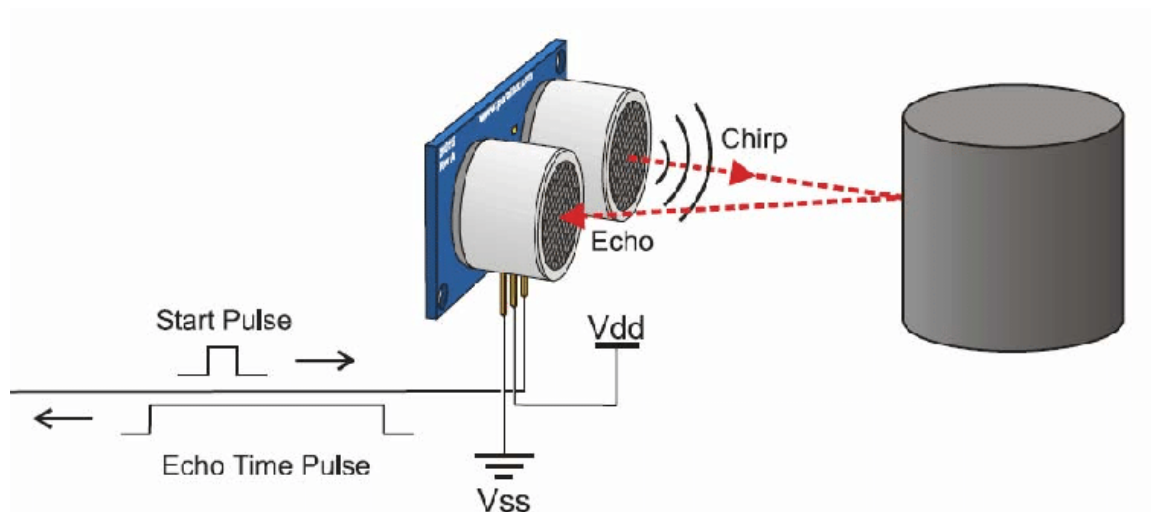


Fig. 3.1.1: Working Principle of Ultrasonic Sensor

3.2.2 Pin Description of Ultrasonic Sensor

In this Ultrasonic Sensor pin describe on figure 3.1.2. Four pins have on this module and on in this pin shows on finger. Ultrasonic Sensor work analog data rate and send Trig pin and Echo pins, +5v pin use power port and also GND pin.

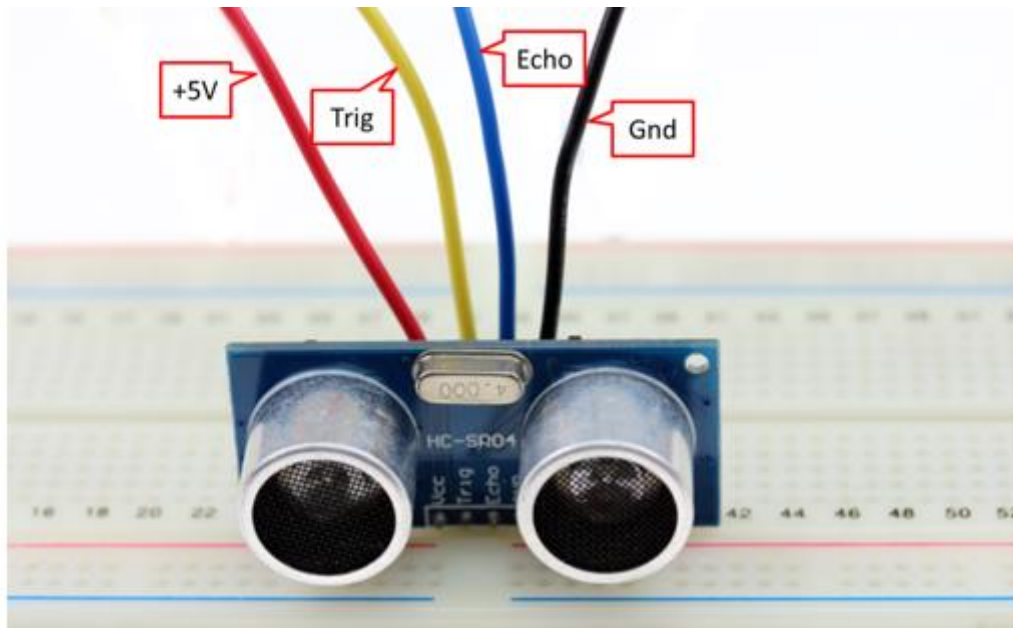


Fig 3.1.2: Pins of Ultrasonic Sensor

1. GND is the common pin on this project
2. Echo is the 2nd pin in this module echo pulse signal from the module to send microcontroller.
3. Trig in the 3rd pin on this module and its work pulse with on the microsecond on the range 40KHz frequency ultrasonic waves on this air.
4. +5V in the 5v supply pin

3.2.3 Applications:

- Wearable Devices
- Object found Devices
- Hole counter Devices

3.3 Description of 433MHz RF Module

3.3.1 RF Transmitter:

The RF module are very small and wide operating range voltage (3V-12V). In this low-cost RF transmitter module transmitted can be up to 100meters on this module antenna

designed and working is environment and also supply voltage will seriously impact effective distance. In this module good for distance of short, it also saves battery life and power device development. In these wireless transmitters work in 315 MHz frequency works. On this module friendly to breadboard circuit and it also work with micromolar and very simple to connected.

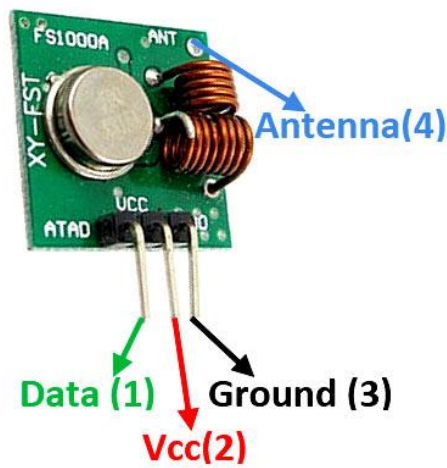


Fig 3.2: RF Transmitter

Specifications:

- 315MHz Frequency works
- 3v~12v working voltage in this module
- 19mm x 19mm x 5mm
- ASK Modulate Mode

3.3.2 RF Receiver:

These RF receiver modules are very small in dimension. The low-cost RF Receiver can be used to receive RF signal from transmitter at the specific frequency which determined by the product specifications. Super regeneration design ensures sensitive to weak signal.

The receiver has 4 pins, but we actually use 3 of them: GND (Ground), VCC (5V) and one DATA pin. Same as RF transmitter, these RF receivers are breadboard friendly too. Both RF transmitter and receiver must work in pair in order to communicate with each other.

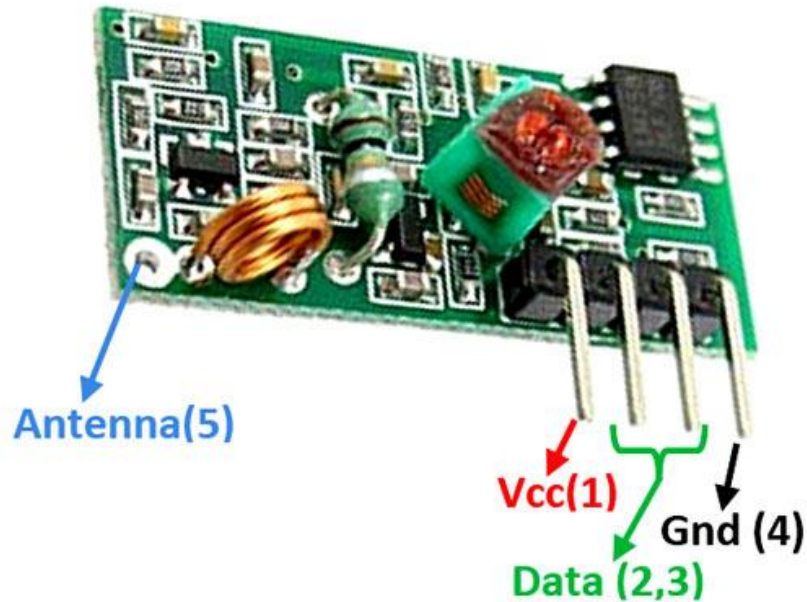


Figure 3.2.1: RF Receiver

Specifications:

- 315MHz Frequency worked
- 5.0v-0.5v working voltage
- 33mm x 13mm x 5mm Dimension
- ASK Modulate Mode

3.3.3 Pin Definition:

RF Transmitter Pinout

Pin	Description
1	GND
2	VCC (3.5-12v)
3	TX Data



Fig 3.2.2: TX and RX module

RF Receiver Pinout

PIN	Description
1	GND
2	RX DATA
3	RX DATA
4	VCC (5v)

3.4 Description of l293D

In this project we used L293D motor driver, in this motor driver control two motor on this project. This IC 8pins and price is low of other motor driver.

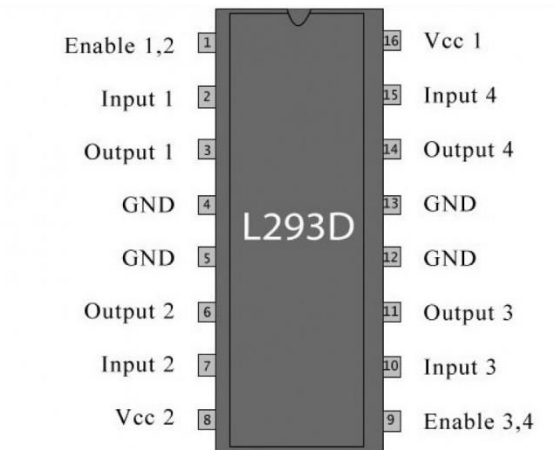


Fig 3.3: L293D

In this fig we show that l293D motor driver we discuss about all pi out

1. Enable or 5v
2. Input for microcontroller for motor 1
3. Output motor 1 pin +
4. GND
5. GND
6. Output motor 1pin –
7. Input for microcontroller for motor 1
8. VCC 2 (12 we used)
9. Enable pin or 5v
10. Input for microcontroller for motor 2
11. Output motor 2 pin –
12. GND
13. GND
14. Output Motor 2 pin +
15. Input for microcontroller for motor 2
16. VCC 1 (5V)

3.5 Basic Description of controller unit:

In this microcontroller unit we using ATmega 328 microcontroller. We also use Arduino IDE 1.8.8 software it can easily use on AVR microcontroller. We used Arduino UNO in this Arduino UNO used AVR ATmega 328 microcontroller. Arduino IDE is open source platform so its easily used to software and Hardware. Anyone uses it because its designers, hobbyists, artists and anyone interest in interacting endearment.

In this microcontroller programmed on this board using Arduino IDE programming language need C. At this Arduino UNO ATmega328 microcontroller-based board it also 14 pin digital output and input and also pin 6 PWM can be used, analog input 6 pin ceramic resonator 16Mhz, 2.0 USB connector, power port, rest button and ICSP header.

Microcontroller support USB to power or AC-to-DC adapter or battery to get.

3.5.1 Technical Specification Arduino Nano:

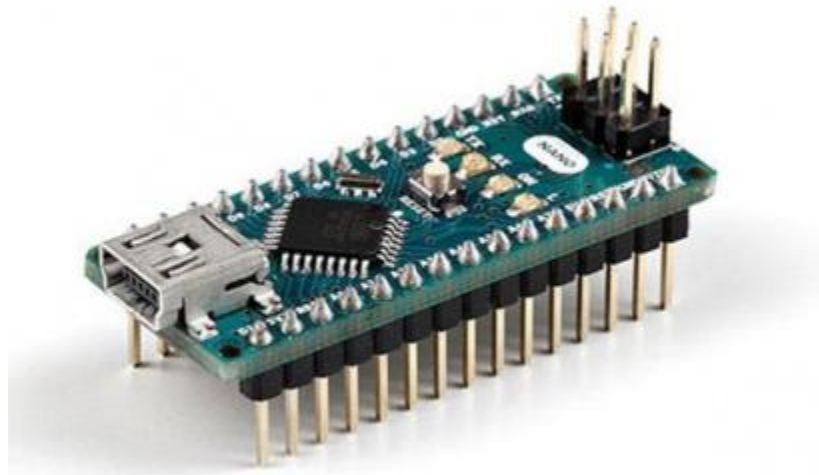


Fig. 3.3(a): Arduino Nano.

Microcontroller	ATmega328
Operating Voltage	5v
Supply Voltage (recommended)	5-9V

Maximum supply voltage	18V
Digital I/O Pins	14 (PWM output on 6 pin)
Analog Input Pins	8
DC Current for 3.3V Pin	50 mA
DC Current per I/O Pin	40 mA
Flash Memory	32 kb
SRAM	2 KB (ATmega328)

3.5.2 Description of Microcontroller: ATmega328:

In this Arduino UNO ATmega328 based board microcontroller. In this microcontroller digital input & output pin total 14 and total analog pin 8, it's also 16Mz crystal, one micro USB port, an ICSO header and reset baton.

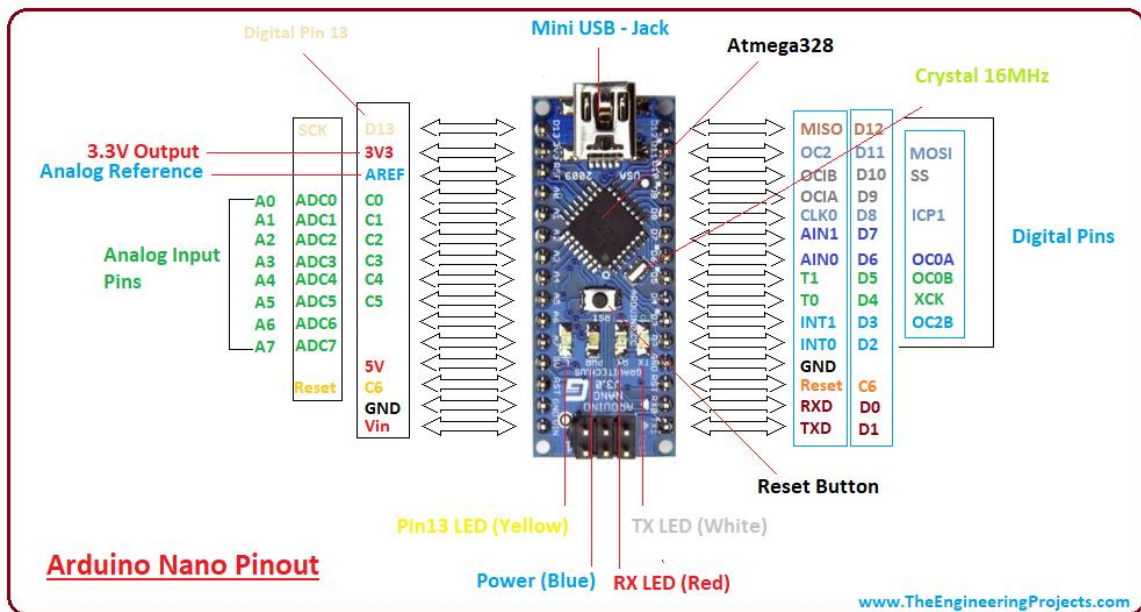


Fig. 3.3(b): Arduino Nano.

In this microcontroller need to program upload USB cable and also power input in this USB port. Arduino UNO work using drive chop FTDI USB-to-serial. In this Arduino UNO operated ATmega328 programmed it's USB-to-serial device. Nano is the one of the popular boards and it also latest USB Arduino, it also latest model Arduino device. Arduino UNO need low power to control and its size small so many types work we used it. Arduino UNO number of communications with PC, Anther Arduino or other microcontrollers.

ATmega328 work UART TTL serial communication, in this Arduino UNO digital pin 0(RX) and digital pin 1(TX). And also, FTDI FT232RL serial communication channels on this board over USB and FTDI drivers.

3.5.3 Block Diagram of Microcontroller – (ATmega328):

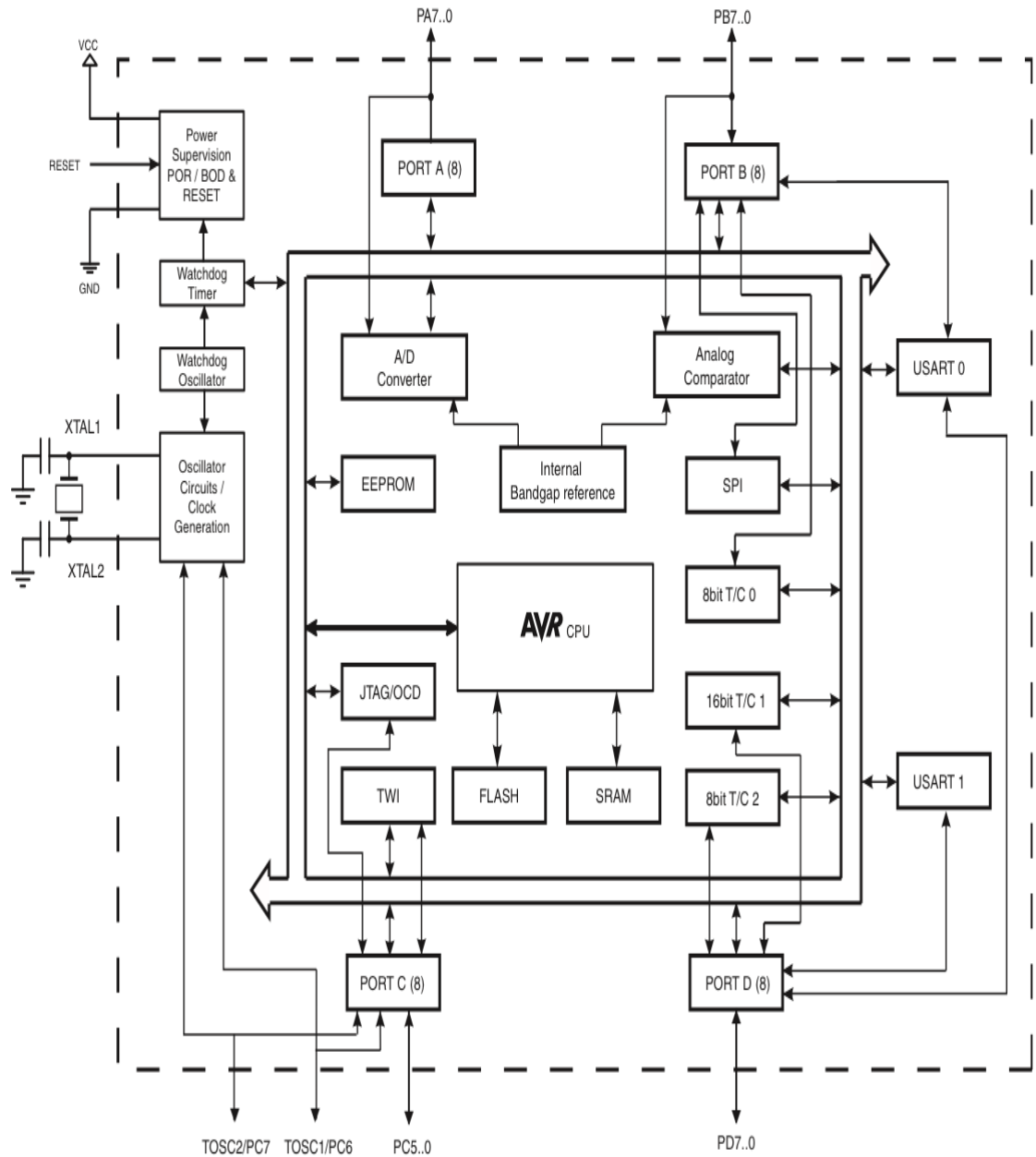


Fig. 3.3.1: Block Diagram of Microcontroller – (Atmega328)

3.5.4 Pin Configurations of Microcontroller – (ATmega328):

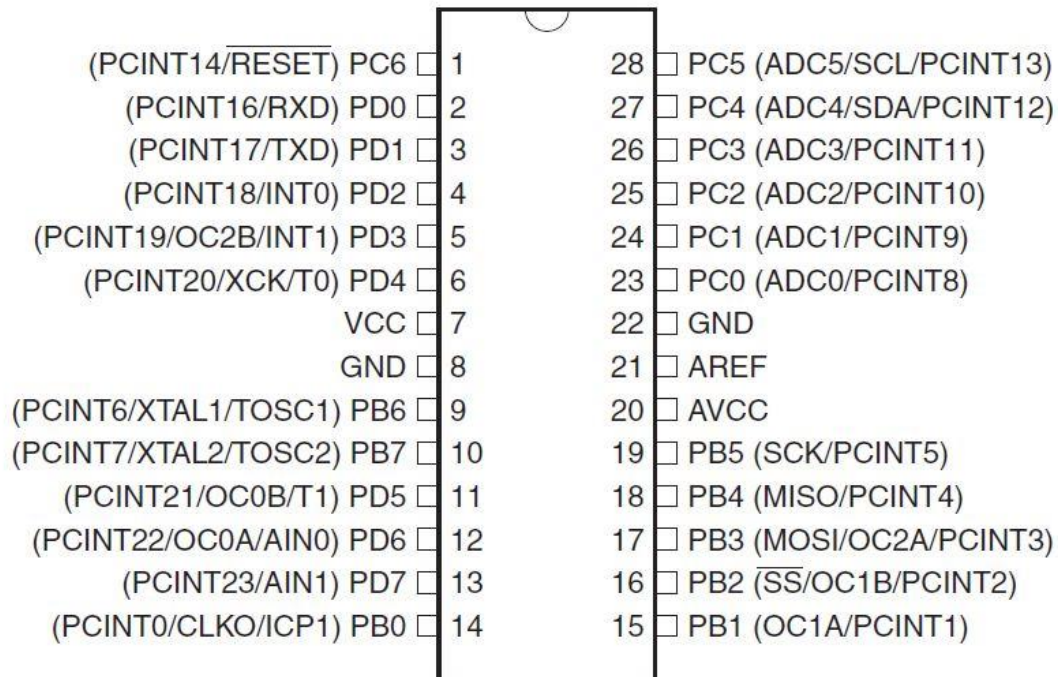


Fig. 3.3.2: Pin Configurations Of Microcontroller –(Atmega328)

3.5.5 Pin Descriptions:

The below gives a description for each of the pins, along with their function: -

The pour pins are as follows:

- VIN: The input voltage to the Arduino board once it's using an external pour supply (as opposed to 5 volts from the USB connection or different regulated pour source). we are able to offer voltage through this pin, or, if supply voltage via the pour jack, access it through this pin.
- 5V: The regulated power offer used to power the microcontroller and different elements on the board. this will come back either from VIN via an on-board regulator, or be supplied by USB or another regulated 5V offer.
- 3V: A 3.3-volt supply generated by the on-board regulator. maximum current draw is 50 mA.
- GND: Ground pins.

Input & Output of Arduino Nano:

On this microcontroller digital pin 14 can be used output or input, using mode DigitalWrite, and DigitalRead functions. In this board operate in 5V. Every pin received 40mA and 20-50KOhms current. We also discuss some pin functions.

- **Serial:** These pins are connected to the pins of the ATmega328 USB-to-TTL Serial chip. 0 is (RX) and 1 is (TX). Used to receive (RX) and transmit (TX) TTL serial data.
- **External Interrupts:** In this Arduino UNO 2 and 3 pins can be configured to trigger an interrupt on a low value, a rising or falling edge value.
- **PWM:** In this board PWN pines 3,5,6,9,10,11 in this 8 pin output analogWrite() function.
- **SPI:** On this ss pin 10, MOSI pine 11, SCK pin 13 in this pis support SPI communication by using the Library SPI.
- **LED:** In this board pin 13 built in LDE connected. When in this pine high value found LED on and low value is OFF.
- **I 2C: 20 (SDA) and 21 (SCL).** The ATmega328 also supports SPI communication and I2C. The Arduino IDE includes many libraries to simplify use of the I2C bus.

There are a couple of other pins on the board:

- **Reset :** On this pont LOW to reset the microcontroller. It used to add a reset button to shields which block the one on the board.
- **AREF :** Analog inputs are reference voltage. It used analog Reference voltage.

Memory: The ATmega328 has 32 KB of flash memory for storing code (of that 0.5 KB is used for the bootloader), a pair of K of SRAM and 1 KB of EEPROM (which is read and written with the EEPROM library)

VIN: Vin pin using the input external voltage source in the pin we use external power supply. It also offer through the pin voltage or supply voltage via the ability point.

5V. In this pin output 5v in this board. In this power input also, DC 5-12V on micro USB connection 5v and also VIN pin input 5V or 3.5v.

3.6 Basic Arduino Software:



Fig. 3.5: Arduino Software.

Arduino software IDE as open source platform it's free for all, it's very popular for all young students. In this software language is C++ or C its also work many of libraries and its work is very easy for all Arduino use AVR C programming language based on this Similarly, we can add AVR-C code directly into our Arduino programs if we want to.

The Arduino Integrated Development Environment - or Arduino Software (IDE) - contains a text editor for writing code, a message area, a text console, a toolbar with buttons for common functions and a series of menus. It connects to the Arduino and Genuine hardware to upload programs and communicate with them. Programs written using Arduino Software

(IDE) are called sketches. These sketches are written in the text editor and are saved with the file extension. ino. The editor has features for cutting/pasting and for searching/replacing text. The message area gives feedback while saving and exporting and also displays errors. The console displays text output by the Arduino Software (IDE), including complete error messages and other information. The bottom right hand corner of the window displays the configured board and serial port. The toolbar buttons allow we to verify and upload programs, create, open, and save sketches, and open the serial monitor.

3.6.1 Features of Arduino Software:

- File
- Edit
- Sketch
- Tools
- Help

Sketchbook

The Arduino software (IDE) uses the concept of a book a standard place to store our programs (or sketches). The sketches in our book may be opened from the File > book menu or from the Open button on the toolbar. the first time we have a tendency to run the Arduino software package, it'll mechanically produce a directory for our book. we will read or amendment the situation of the book location from with the Preferences dialog.

Beginning with version one.0, files are saved with a .ino file extension. Previous versions use the. pde extension. we have a tendency to should still open. pde named files in version one.0 and later, the software package can mechanically rename the extension ino.

Tabs, Multiple Files, and Compilation

Allows us to manage sketches with more than one file (each of which appears in its own tab). These can be normal Arduino code files (no visible extension), C files (.c extension), C++ files (.cpp), or header files (.h).

Uploading

Before uploading our sketch, we want to pick the right things from the Tools > Board and Tools > Port menus. The boards square measure delineate below. On the Mac, the port is maybe one thing like /dev/tty.usbmodem241 (for associate degree UNO or Mega2560 or Leonardo) or /dev/tty.usbserial-1B1 (for a Duemilanove or earlier USB board), or On Windows, it's most likely COM1 or COM2 (for a serial board) or COM4, COM5, COM7, or higher (for a USB board) - to seek out out, we glance for USB serial device within the ports section of the Windows Device Manager. On Linux, it ought to be /dev/ttyACMx , /dev/ttyUSBx or similar. Once we've elect the right port and board, press the transfer button within the toolbar or choose the transfer item from the Sketch menu. Current Arduino boards can reset mechanically and start the transfer. With older boards (pre-Diecimila) that lack auto-reset, we'll ought to press the push on the board simply before beginning the transfer. On most boards, we'll see the RX and Lone-Star State LEDs blink because the sketch is uploaded. The Arduino software package (IDE) can show a message once the transfer is complete, or show a slip-up. When we transfer a sketch, we're victimisation the Arduino bootloader, atiny low. It permits we have a tendency to to transfer code while not victimisation any extra hardware. The bootloader is active for a number of seconds once the board resets; then it starts whichever sketch was last uploaded to the microcontroller. The bootloader can blink the on-board (pin 13) LED once it starts (i.e. once the board resets).

Third-Party Hardware

Support for third-party hardware can be added to the hardware directory of our sketch block directory. Platforms put in there might embrace board definitions (which seem in the board menu), core libraries, bootloaders, and technologist definitions. To install, produce the hardware directory, then unzip the third-party platform into its own sub-directory. (Don't

use "Arduino" as the sub-directory name or we'll override the built-in Arduino platform.) To uninstall, merely delete its directory.

Libraries

Libraries supply further practicality to be employed in sketches, e.g. operating with hardware or manipulating information. To use a library in an exceedingly } very sketch, opt for it from the Sketch > Import Library menu. this will insert one or plenty of #include statements at the highest of the sketch and compile the library with our sketch. as a results of libraries unit uploaded to the board with our sketch, they increase the quantity of space it takes up. If a sketch not wants a library, simply delete its #include statements from the best of our code. Some libraries square measure clathrate with the Arduino code. Others square measure usually downloaded from a variety of sources or through the Library Manager. starting with version two.12of the IDE, we tend to do can import a library from a zipper file and use it in Associate in Nursing open sketch.

Serial Monitor

Displays serial data being sent from the Arduino or Genuine board (USB or serial board). To send data to the board, enter text and click on the "send" button or press enter. Choose the baud rate from the drop-down that matches the rate passed to Serial.begin in our sketch. Note that on Windows, Mac or Linux, the Arduino or Genuine board will reset (rerun our sketch execution to the beginning) when we connect with the serial monitor. We can also talk to the board from Processing, Flash, MaxMSP.

Preferences

Some preferences are set within the preferences dialog (found beneath the Arduino menu on the Mack, or File on Windows and Linux). the remainder is found within the preferences file, whose location is shown within the preference dialog.

3.7 Wires:



Fig. 3.6(a): Female to Male Jumper Wire. Fig. 3.6(b): Male to Male jumper

In this project we need couples of Female to Male and Male to Male jumper wire. That is for connecting Bluetooth module and relay module to the Arduino UNO board.



Fig. 3.6.1: 220v carried wire.

And we need some 220v carried wire to connect load in the relay module.

3.8 Cost Analysis

In this section we will show cost of our project that means cost sheet representation of our project.

3.8.1 Cost Sheet:

No	Component Name	Quantity	Purchase Price (TK)
1.	Ultrasonic Module	4	400.00
2.	Arduino Nano	1	400.00
3.	ATmega 328p	1	150.00
4.	433MHz RF Module	1	200.00
5.	Capacitor	5	40.00
6.	Battery	2	500.00
7.	Arduino Software	1	Free
8.	IC connector	2	100.00
9.	LED	5	15.00
10.	Resistor	20	20.00
11.	Veroboard	2	40.00
12.	L293D	1	150.00
13.	Power Adapter 12v	1	150.00
14.	Wires	40	120.00
15.	Chassis 2 Wheel	1	700.00
16.	Push Button	5	25.00
	Total Cost		=3,010.00

Comparison: -

Our all components are available in market. We get all components are very reasonable price. So that we make this project more cost efficient.

3.9 Conclusion

Five main Component & some tools are used in this system to makes it. This Project is used to Accident avoidance System for vehicles Our all component is very simple & available in our country market.

CHAPTER 4

SOFTWARE ANALYSIS

4.1 Introduction

In this chapter we discuss about software use and implementation on microcontroller in this project we use Arduino UNO and software Arduino IDE and programming language C and we also use Arduino program in this chapter document and development program we show that.

4.2 Description of our Software

The ASCII text file Arduino setting help to done to jot down code, in this code upload I/O board into IDE. In this software runs on Windows, Mac OS X, and Linux. The setting is written in Java and supported process, avr-gcc, and another open supply software. The screen shot of Arduino 1.8.7 is shown below...

It is also capable of assembling and uploading programs to the board with one click. there's usually no ought to edit build files or run programs on a command-line interface. though building on command-line is feasible if needed with some third-party tools like particle. The Arduino IDE comes with a C/C++ library known as "Wiring" (from the project of constant name), that makes several common input/output operations a lot of easier. Arduino programs are written in C/C++, though users solely want define two functions to form a runnable program:

setup () – a function run once at the start of a program that can initialize settings

loop() – a function called repeatedly until the board put off

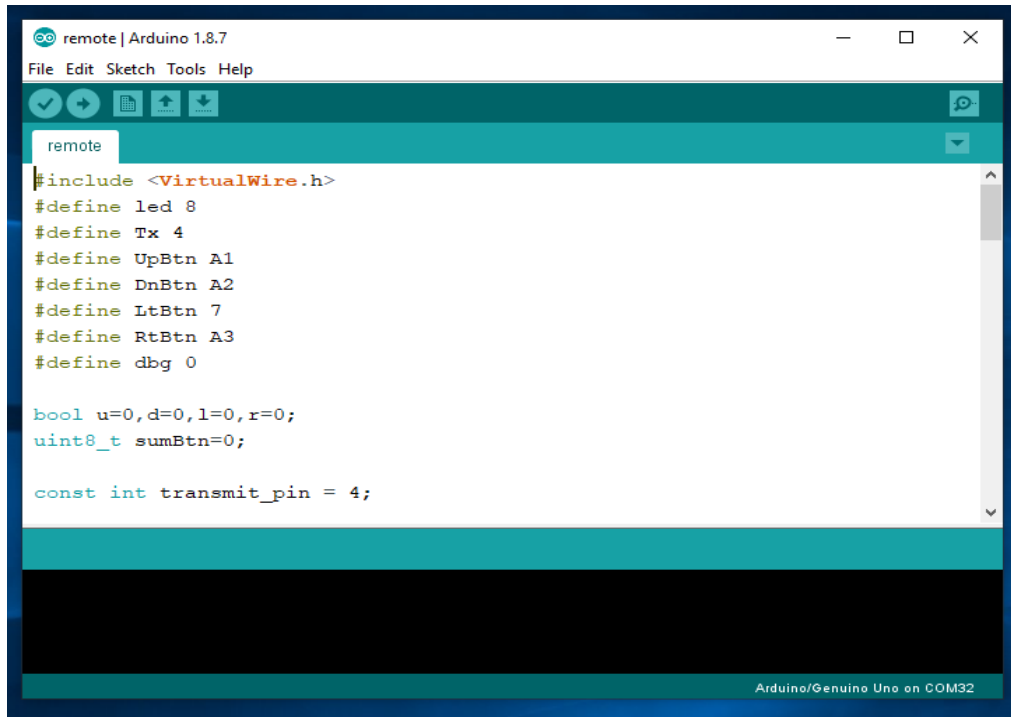


Fig. 4.1: Software Platform

The compiled window of our code is shown below:

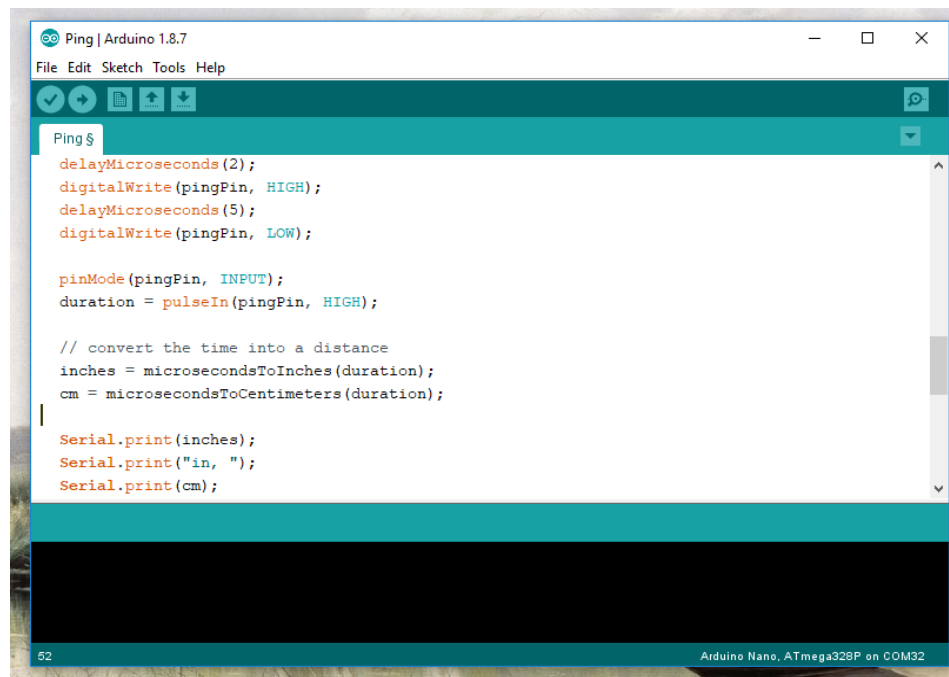


Fig. 4.2: Compiling window

4.3 Flow Chart Diagram:

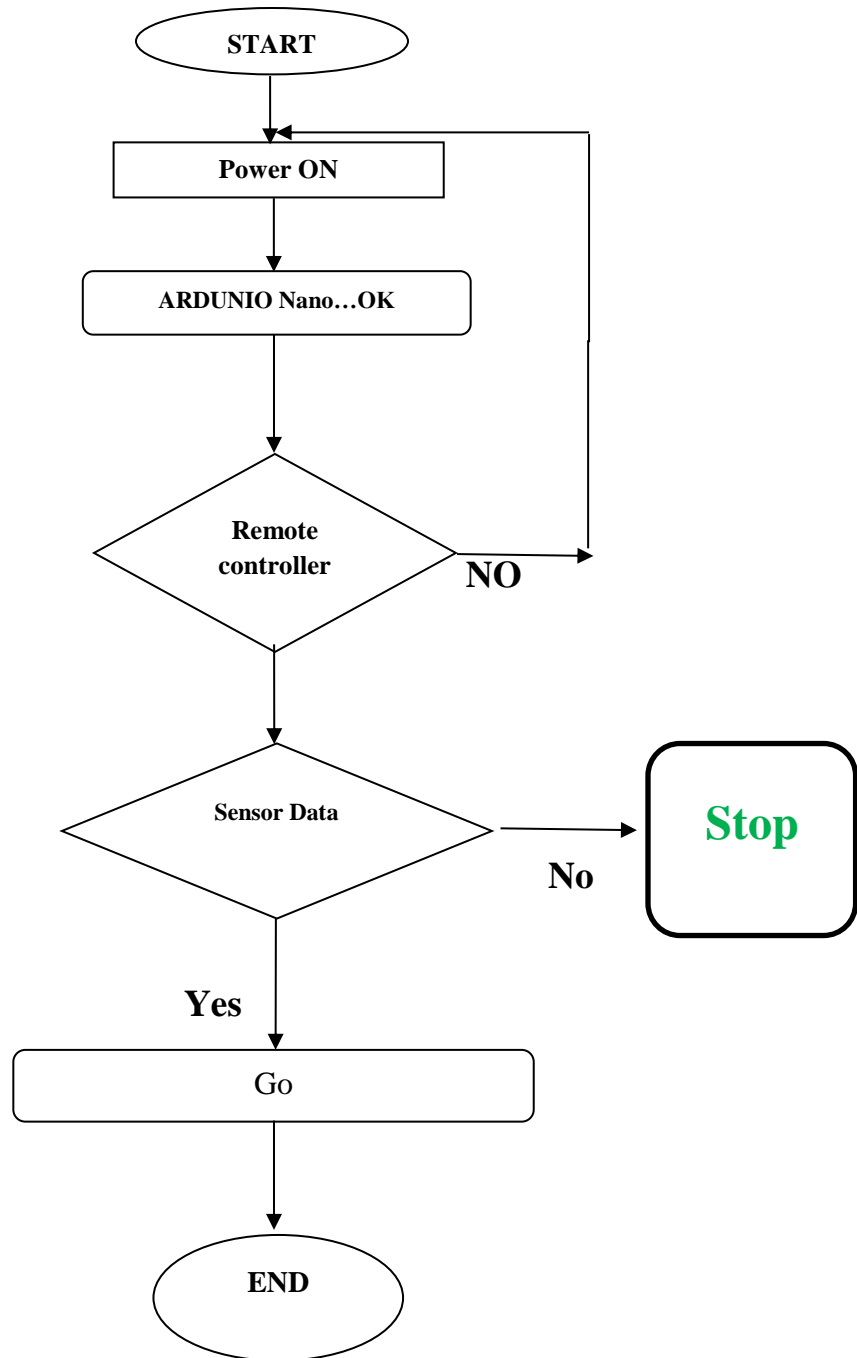


Fig. 4.3: Flow chart of our system

4.3.1 Program Explanation

Computing: a group of coded directions that a computer can understand to resolve a retardant or manufacture a desired result. 2 basic sorts of computer programs are (1) associate OS, that gives the foremost basic directions a computer uses in its operations, and (2) a computer program, that runs on the OS and can a specific job like data processing.

Programs unit written either in one altogether high-level programming languages (such as BASIC, C, Java) that are easier but execute relatively slowly, or in one altogether low-level languages (assembly language or machine language) that are really sophisticated but execute in no time.

- Software Serial Communication is used with PWM pin of Arduino.

<SoftwareSerial.h> - Header library is made us of

- Pin 2,3,4 – Configured as Input
- Pin TX,RX – Configured as Output

The system software developed in Embedded C, C++ language which has the ability of receiving the data from sensor and transmitting the data, and controls all the appliances that connected. Because of its low power consumption, easy usage, reliability it is used in other fields. Software analysis is a very important part of our system. A Software analysis makes sure good design. A proper Software analysis and its burn into Arduino UNO the project to a smooth end.

CHAPTER 5

HARDWARE IMLEMENTATION

5.1 Introduction

This is one of the most important chapters of this report. In this chapter we will show our completed project's outlook that means Connection lawyer & operation representation of our project. We will discuss about component's interfacing with Arduino UNO of our system.

5.2 Interfacing of Ultrasonic Sensor

HC-SR04 is connected with Arduino, by attaching its 3pin on our PCB board Data pin A4, A5 with respectively and Ultrasonic sensor operates on 5 volts.

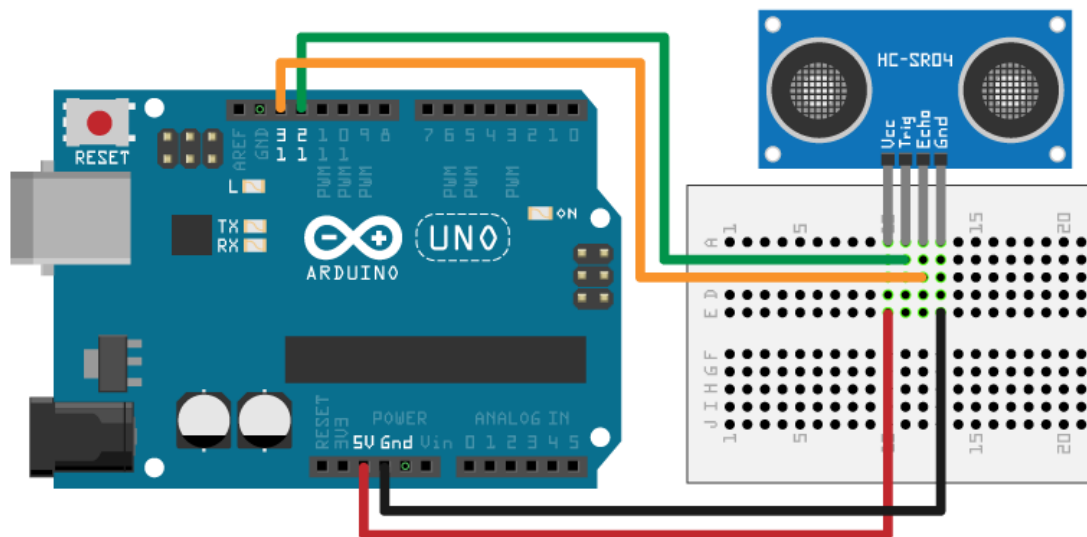


Fig. 5.1: Interfacing of Ultrasonic Sensor.

Ultrasonic HC-SR04	Arduino UNO
VIN	5V
GND	GND
Tring	A4
Echo	A4

5.3 Interfacing with RF Module

RF Transmitter

The RF Transmitter module uses SPI (Serial Peripheral Interface) communication protocol, the name of each connection pin is marked on the back of the module as show below.



Fig 5.2: Interfacing with RF Transmitter

RF Receiver

The RF Receiver module uses SPI (Serial Peripheral Interface) communication protocol, the name of each connection pin is marked on the back of the module as show below.

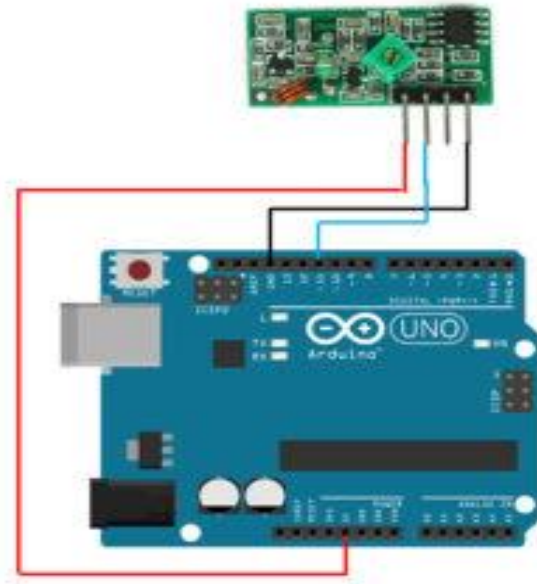


Fig 5.2.1: Interfacing with RF Receiver

5.4 Interfacing with L293D

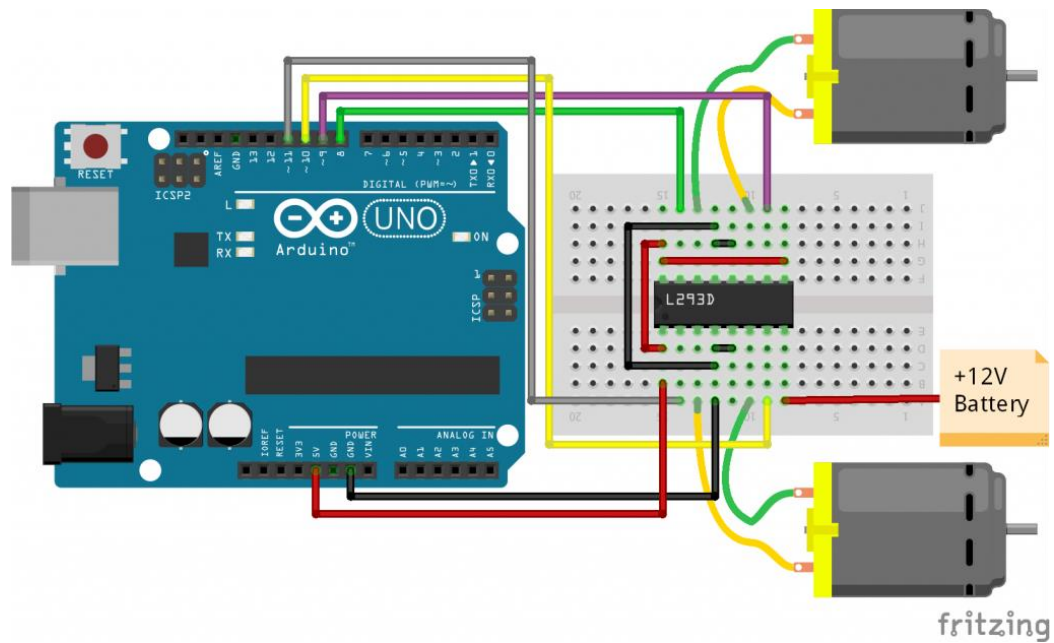


Fig 5.3: Interfacing with L293D

In this module connected with Arduino show this fig and we show that connection.

5.5 Battery Interfacing

The below figure shows that how to connect a Battery with this project.

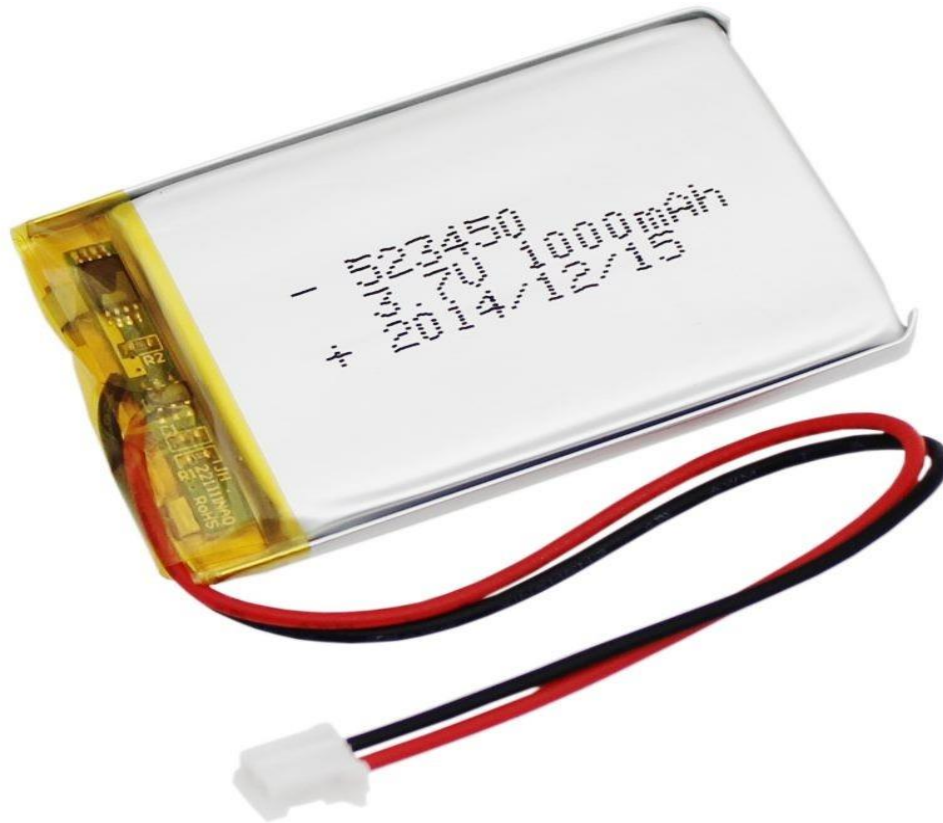


Fig 5.4: Interfacing with Battery.

Battery have two pins. 1 pin is +V and 2 pins is -V of battery charging. In this project we are using a battery. That two pins are the connected battery charger IC of our project.

5.6 System Operation

In this project base on ATmega328 microcontroller. The ultrasonic Sensor is act as a switch. The project is also just like the auto system. The inputs given to the microchip are the outputs of ultrasonic. The outputs of the controller are given to the ultrasonic sensor. The Output of ultrasonic is given to the 4 pins of ATmega328. the load device is amplified and digitized by analog to digital convertor and is given to the port pins of controller. once the microcontroller receive output from the digital data on or off, it offers command to operate the ultrasonic sensor of ATmega328p severally. For the protection circuitry fan is given controller.

5.8 Conclusion

To design, interfacing with Arduino UNO of the circuit is the basic purpose of our project. Our Project work is already completed. The constructed circuit is very nicely working.

CHAPTER 6

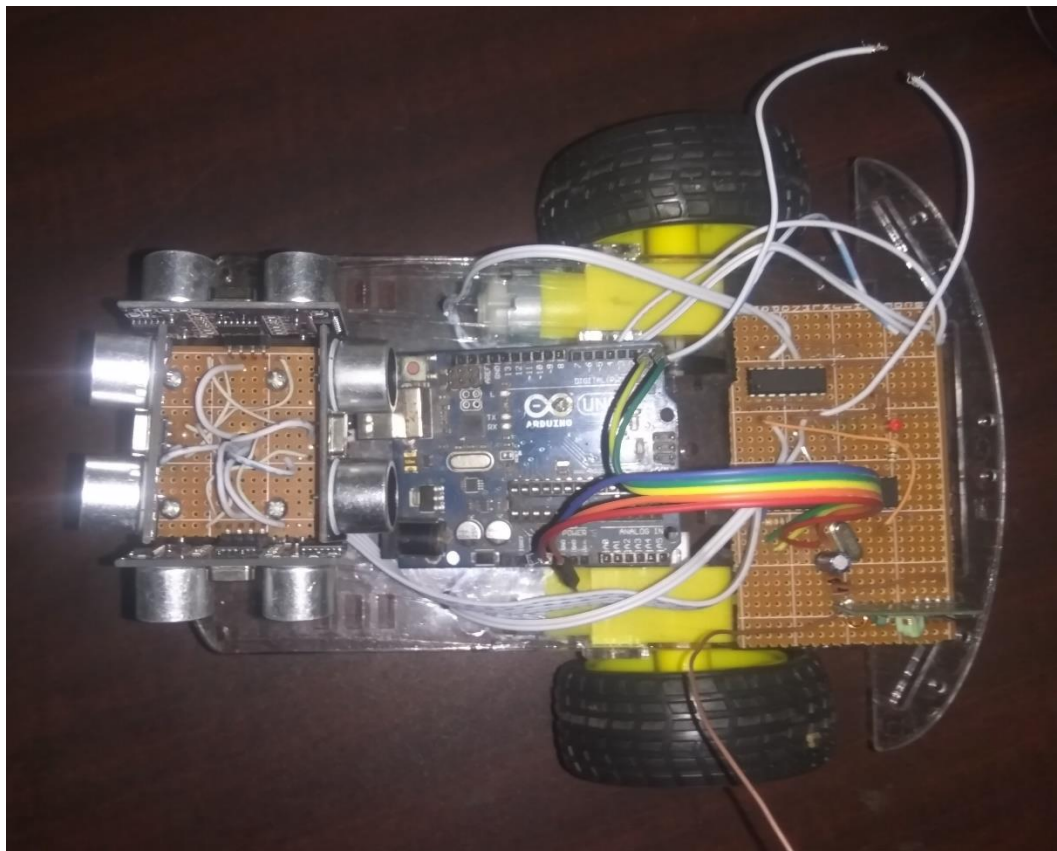
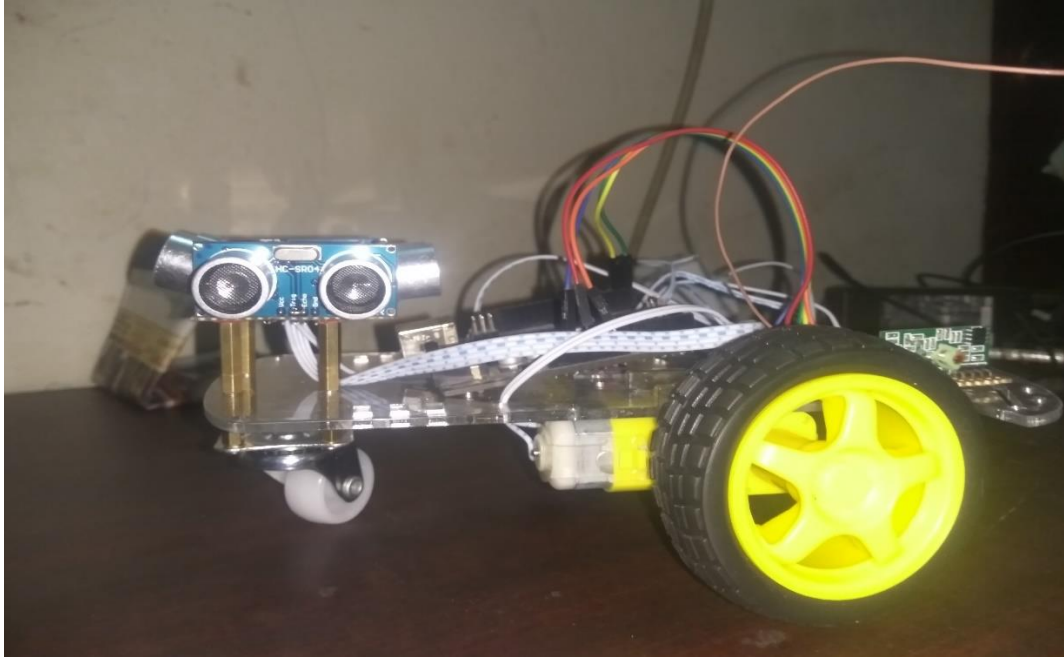
RESULTS AND DISCUSSIONS

6.1 Introduction

The system proposed consists of five major modules named, Ultrasonic Sensor, ATmega328P, 1293D, 433MHz RF Module & Battery. It uses ATmega328, in this microcontroller is 8 bits. In this microcontroller application and features in wide and it also power is low. In this chapter we will discuss about tests & results in our “Accident avoidance System for vehicles” project.

6.2 Experimental Setup

ATmega328 is the base of the system. The inputs given to the Pulse Sensor are the outputs is digital data ON or OFF on Ultrasonic result. The output of the controller is given to the Ultrasonic sensor. The Output of this data is given to the D3, D4, D5, D6, D7 pin of ATmega328. The weight sensor is amplified and digitized by analog to digital converter and is given to the port pins of controller.



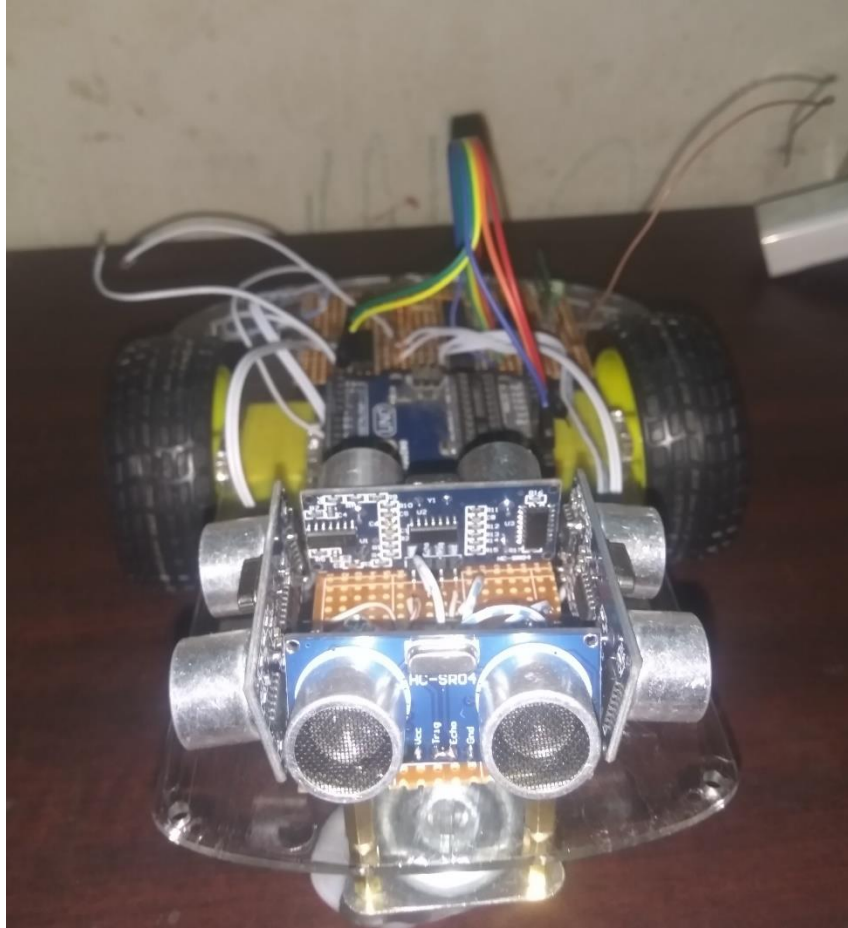


Fig 6.1: Setup of our system

The Ultrasonic sensor data receiver pins are connected to the A4 & A5 of ATmega328p respectively. For the protection circuitry exhaust fan is provided with controller.

6.3 Result

In this project will done when works complete. When we upload this code to Arduino use Arduino IDE and power on this system. In this project we use 4 ultrasonic sensor and one motor driver IC and one microcontroller ATmega 328p and use RF module. When ultrasonic sensor send data to microcontroller and microcontroller received data and

microcontroller send command data to Motor driver IC. Motor driver IC control car go or stop. When finally, we test it work perfectly and we also test another place and other object when any object found car distance 10mm then stop car.

6.4 Advantages

- Portable
- Low Power Consumption
- Easy to Use

6.5 Disadvantages

- Not Precise
- Requires A Light Source

6.6 Conclusion

We tested our system & it's worked properly. As beginning of the code section, we use unit testing. When we made our modules, we test it by unit testing. After complete development process of our full system, we test it by integration testing system. In every test section we found some problems and we solve those problems as soon as possible. After completing all process including testing, we are assuring that it's ready for home and commercial use.

CHAPTER 7

CONCLUSION

7.1 Conclusion

A rear end anti-collision warning system is designed and mounted on a very simple and easily understandable model constructed to demonstrate the system and it was found functional. The sensor was able to read distances that are at shorter range accurately. The system was not real time because there were incidences where the microcontroller didn't receive any feedback. This is due to noise from the environment.

A distance sensor that detects objects at long distances is required to apply on a real vehicle. Therefore, if the right materials are collected, it is possible to enhance its features so that it can be used in vehicles. This model is also a good tool to use for demonstration for anti-collision warning system research.

7.2 Applications

- In this project we used our any kind of vehicles
- In this project help to our driving time some time we out on sense that time is work
- Many other applications in this project

7.3 Limitations of the Work

- In-accurate method of is risk some moment of car going and stop time
- Logic used in very simple. Therefore, results may vary as for a sophisticated instrument for the same purpose.

7.4 Future work

So far, we've been able to build a automaton early sort vehicle to avoid accident for front side and back facet. Our main plan is to possess protection from all facet. If there comes an object from left facet, the automotive can mechanically move to the correct facet and also the object comes from the correct facet, the item can mechanically move to the left facet. For additional implementation in future to use in real automotive and to notice long distance for the big cars, we are going to use measuring system. There ought to be a switch to ON and OFF this software system for parking system. Since East Pakistan is associate degree over dense country most of the accident occurs here as a result of the drivers cannot management their speed throughout holdup and cause tiny accidents and even results in the large accidents. we tend to area unit even coming up with this to apply on a scheme car which is able to not be ecofriendly however additionally are a wise car for this generation.

REFERENCES:

- [1] <https://www.arduino.cc/>, retrieved on November 2018.
- [2] INVESTIGATION OF LASER AND ULTRASONIC RANGING SENSORS FOR MEASUREMENTS OF CITRUS CANOPY VOLUME S. D. Tumbo, M. Salyani, J. D. Whitney, T. A. Wheaton, W. M. Miller
- [3] Experimental Characterization of Polaroid Ultrasonic Sensors in Single and Phased Array Configuration Alex Cao* and Johann Borenstein** The University of Michigan, Department of Mechanical Engineering
- [4] Cognitive efficiency in robot control by Emotiv EPOC Chowdhury, P. ; Sch. of Eng. & Comput. Sci., Dhaka, Bangladesh ; Kibria Shakim, S.S. ; Karim, M.R. ; Rhaman, M.K.
- [5] Agricultural Machinery Safety Alert System Using Ultrasonic Sensors L.Guo, Q. Zhang, S. Ha T. Yagi, Y. Kuno, Y. Uchikawa, "Prediction of eye movements from EEG," Proceedings of the 6th International Conference on Neural Information Processing (ICONIP'99), Perth Austria, 16-20, pp. 1127-1131, Nov 1999.
- [6] Global Integration of Ultrasonic Sensors Information in Mobile Robot Localization L. Moreno, J. M. Armingol, A. de la Escalera and M. A. Salichs Universidad Carlos III de Madrid, Division of Systems Engineering and Automation.
- [7] Analysis of Vehicle Detection with WSN-Based Ultrasonic Sensors Youngtae Jo and Inbum Jun * Department of Computer Information and Communication Engineering, Kangwon National University, Chuncheon, Gangwondo 200-701, Korea, sensors ISSN 1424-8220
- [8] <http://www.engineersgarage.com/electronic-components/16x2-lcd-moduledatasheet> [last visited 1-04-2015]
- [9] <http://www.ti.com/lit/ds/symlink/1293.pdf>36, Issue 2, Part 1, pages 2352- 2359, Mar2009. [last visited on 4-04-2015]
- [10] <http://www.instructables.com/id/Easy-ultrasonic-4-pin-sensor-monitoringhc-sr04/?ALLSTEPS> [last visited on 5-04-2015]
- [11] <https://www.irjet.net/archives/V2/i7/IRJET-V2I7119.pdf>
- [12] <https://www.e-tinkers.com/2017/11/how-to-use-lcd-5110-pcd-8544-with-arduino/>
- [13] Submitted to Symbiosis International University on 2018-06-05
- [14] <https://www.arduino.cc/en/guide/environment>
- [15] Dietmayer K, Sparbert J, Streller D. Model based object classification and object tracking in traffic scenes from range images. IEEE Intelligent Vehicle Symp; Tokyo. 2001 Jan.

- [16] Furstenberg, KCh, Dietmayer KDJ, Eisenlauer S. Multilayer laserscanner for robust object tracking and classification in urban traffic scenes. 9th World Congress on Intelligent Transport Systems; Chicago. 2002 Oct.
- [17] Kato T, Ninomiya Y, Masaki I. An obstacle detection method by fusion of radar and motion stereo. IEEE Trans Intell Transp Syst. 2002 Sep; 3(3):182–8.
- [18] Aufreere R, Gowdy J, Mertz C, Thorpe C, Wang C, Yata T. Perception for collision avoidance and autonomous driving. Mechatronics. 2003; 13:1149–61.
- [19] Verma R, Del Vecchio. Semiautonomous vehicle safety. A hybrid control approach. IEEE Robot Autom Mag. 2011 Oct; 18(3):44–54.
- [20] Trapagier PG, Nagel J, Kinney PM, Koutsougeras C, Dooner M. KAT-5: Robust systems for autonomous vehicle navigation in challenging and unknown terrain. 2006 Aug; 23(8):509–26.

APPENDIX A

Important Initialization & Setup:-

- Software Serial Communication is used with PWM pin of Arduino.

<SoftwareSerial.h> - Header library is made us of!

- Pin 2,3,4 – Configured as Input
- Pin TX,RX – Configured as Output

The Code is given below:

1. Remote Program

```
#include <VirtualWire.h>
```

```
#define led 8
```

```
#define Tx 4
```

```
#define UpBtn A1
```

```
#define DnBtn A2
```

```
#define LtBtn 7
```

```
#define RtBtn A3
```

```
#define dbg 0
```

```
bool u=0,d=0,l=0,r=0;
```

```
uint8_t sumBtn=0;
```

```
const int transmit_pin = 4;
```

```

void setup()
{
    // Initialise the IO and ISR

    vw_set_tx_pin(Tx);

    vw_setup(2000);    // Bits per sec

    pinMode(led, OUTPUT);

    pinMode(UpBtn, OUTPUT);

    pinMode(DnBtn, OUTPUT);

    pinMode(LtBtn, OUTPUT);

    pinMode(RtBtn, OUTPUT);

    if(dbg) Serial.begin(9600);;
}

byte count = 1;

void loop()
{
    char msg[1] = {'h'};

    ReadBtn();

    if(sumBtn >0 ){

        digitalWrite(led, HIGH);

        msg[0] = sumBtn;

        vw_send((uint8_t *)msg, 1);

        vw_wait_tx(); // Wait until the whole message is gone

```



```

}
else {
    digitalWrite(led, LOW);
    msg[0] = 5;
    vw_send((uint8_t *)msg, 1);
    vw_wait_tx(); // Wait until the whole message is gone
    delay(22);
}

//count = count + 1;
if(dbg) debug();
//delay(300);
}

void ReadBtn(){
    u = digitalRead(UpBtn);
    d = digitalRead(DnBtn);
    l = digitalRead(LtBtn);
    r = digitalRead(RtBtn);
    sumBtn = 0;
    sumBtn |= u;
    sumBtn |= d<<1;
    sumBtn |= l<<2;
    sumBtn |= r<<3;
}

```

```
void debug(){  
    Serial.print(" U:");Serial.print(u);  
    Serial.print(" D:");Serial.print(d);  
    Serial.print(" L:");Serial.print(l);  
    Serial.print(" R:");Serial.print(r);  
    Serial.print(" SUM:");Serial.print(sumBtn,BIN);  
    Serial.println("");  
  
}
```

2. Main Device Program

```
#include <NewPing.h>  
#include <VirtualWire.h>  
  
#define LED 9  
  
#define Rx 10  
  
#define TxE 55  
  
  
#define SnF A5  
  
#define SnB A3  
  
#define SnL A4  
  
#define SnR A2  
  
  
#define LMA 7
```

```

#define LMB 13

#define RMA 12

#define RMB 11

#define Frange 10

#define Brange 10

#define Lrange 10

#define Rrange 10

#define dbg 0

#define SONAR_NUM 4

#define MAX_DISTANCE 100

uint8_t sf,sb,sl,sr,oldGot;

uint8_t buf[1];

uint8_t buflen = 1;

    NewPing sonar[SONAR_NUM] = { // Sensor object array.

        NewPing(SnF, SnF, MAX_DISTANCE), // Each sensor's trigger pin, echo pin, and max
distance to ping.

        NewPing(SnB, SnB, MAX_DISTANCE),

        NewPing(SnL, SnL, MAX_DISTANCE),

        NewPing(SnR, SnR, MAX_DISTANCE)

    };

void setup()

{ if(dbg) Serial.begin(9600);

```

```

delay(1000);

vw_set_rx_pin(Rx);

vw_set_ptt_pin(TxE);

vw_setup(2000); // Bits per sec

vw_rx_start(); // Start the receiver PLL running

pinMode(LED, OUTPUT);

pinMode(LMA, OUTPUT);

pinMode(LMB, OUTPUT);

pinMode(RMA, OUTPUT);

pinMode(RMB, OUTPUT);

}

void loop()

{

// while(9) {

// digitalWrite(LMA,1);

// digitalWrite(LMB,0);

// digitalWrite(RMA,0);

// digitalWrite(RMB,1);

// }

if (vw_get_message(buf, &buflen)) // Non-blocking

{

```

```

digitalWrite(LED, LOW);delay(5);
if(dbg){
    Serial.print("Got: ");
    Serial.print(buf[0], HEX);
    Serial.print(' ');
    Serial.println();
}
digitalWrite(LED, HIGH); delay(5);
if(buf[0] != oldGot){
    switch(buf[0]){
        case 1:
            Forward();digitalWrite(LED, HIGH);
            break;
        case 2:
            Backward();digitalWrite(LED, HIGH);Serial.println("Backward");delay(20);
            break;
        case 4:
            Left();digitalWrite(LED, HIGH);Serial.println("Left"); delay(20);
            break;
        case 8:
            Right();digitalWrite(LED, HIGH);
            break;

        case 5:
            OFF();digitalWrite(LED, LOW);

```

```

        break;

    }
}
else autoOFF();
oldGot = buf[0];
}
if(dbg) {ReadSoanr();debug();}
}

void ReadSoanr(){
    delay(30);sf = sonar[0].ping_cm();
    delay(30);sb = sonar[1].ping_cm();
    delay(30);sl = sonar[2].ping_cm();
    delay(30);sr = sonar[3].ping_cm();
}

void Forward(){
    delay(30);sf = sonar[0].ping_cm();
    if(sf == 0 || sf>Frange){
        if(dbg)Serial.println("Forward");
        digitalWrite(LMA,1);
        digitalWrite(LMB,0);
        digitalWrite(RMA,1);
        digitalWrite(RMB,0);
    }
}

```

```

    else {OFF(); if(dbg)Serial.println("OFF from Forward");}
}

void Backward(){
    delay(30);sb = sonar[1].ping_cm();
    if(sb==0 || sb>Brange){
        digitalWrite(LMA,0);
        digitalWrite(LMB,1);
        digitalWrite(RMA,0);
        digitalWrite(RMB,1);
    }
    else {OFF(); if(dbg)Serial.println("OFF from Backward");}
}

void Left(){
    delay(30);sl = sonar[2].ping_cm();
    if(sl==0 || sl>Lrange){
        digitalWrite(LMA,0);
        digitalWrite(LMB,1);
        digitalWrite(RMA,1);
        digitalWrite(RMB,0);
    }
    else {OFF(); if(dbg)Serial.println("OFF from Left");}
}

void Right(){
    delay(30);sr = sonar[3].ping_cm();
    if(sr==0 || sr>Rrange){

```

```

    digitalWrite(LMA,1);
    digitalWrite(LMB,0);
    digitalWrite(RMA,0);
    digitalWrite(RMB,1);
}
else {OFF(); if(dbg)Serial.println("OFF from Right");}
}

void OFF(){
    digitalWrite(LMA,0);
    digitalWrite(LMB,0);
    digitalWrite(RMA,0);
    digitalWrite(RMB,0);
}

void autoOFF(){
    if(buf[0] == 1){delay(30);sf = sonar[0].ping_cm();}
    else if(buf[0] == 2){delay(30);sb = sonar[1].ping_cm();}
    else if(buf[0] == 4){delay(30);sl = sonar[2].ping_cm();}
    else if(buf[0] == 8){delay(30);sr = sonar[3].ping_cm();}

    if(oldGot== 1){if(sf > 0 && sf<Frangle){OFF();}}
    else if(oldGot== 2){if(sb > 0 && sb<Brangle){OFF();}}
    else if(oldGot== 4){if(sl > 0 && sl<Lrange){OFF();}}
    else if(oldGot== 8){if(sr > 0 && sr<Rrange){OFF();}}
}

void debug(){

```



```
Serial.print(" Front:");Serial.print(sf);  
Serial.print(" Back:");Serial.print(sb);  
Serial.print(" Left:");Serial.print(sl);  
Serial.print(" Right:");Serial.print(sr);  
Serial.println();  
delay(100);  
  
}
```