A project Report on

FIRE EXTINGUISHER ROBOT CAR WITH CALL ALERT

Submitted in partial fulfillment of the requirements for the award of the degree of

BACHELOR OF

TECHNOLOGY IN

ELECTRONICS & COMMUNICATION ENGINEERING

By

MANIKANTA KORIMILLI	20MH1A04F6
SANKHAPANI LOLABHATTU (TL)	20MH1A04F8
MOHANSAI MUDAPAKA	20MH1A04G4
JANAKI ORISSALA	20MH1A04G8
PASALA HARI SANDEEP	21MH5A0459

Under the Esteemed Guidance of

Mr. M. Kishore Kumar, M.Tech, (Ph.D).

Associate Professor

Department of Electronics & Communication Engineering

ADITYA COLLEGE OF ENGINEERING

(*Approved by AICTE, New Delhi & Permanently Affiliated to JNTUK, Kakinada, accredited by NBA & NAAC-A^{++})*

Recognized by UGC under sections 2(f) & 12(b) of UGC Act, 1956

Aditya Nagar, ADB road, Surampalem, E.G.Dt, A.P-533437

2020 - 2024

ADITYA COLLEGE OF ENGINEERING

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

BONAFIDE CERTIFICATE

This is to certify that the Project Report entitled

FIRE EXTINGUISHER ROBOT CAR WITH CALL ALERT

being submitted by

MANIKANTA KORIMILLI	**20MH1A04F6**
SANKHAPANI LOLABHATTU (TL)	**20MH1A04F8**
MOHANSAI MUDAPAKA	**20MH1A04G4**
JANAKI ORISSALA	**20MH1A04G8**
PASALA HARI SANDEEP	**21MH5A0459**

In partial fulfillment of the requirements for the award of the Degree of Bachelor of Technology in Electronics and Communication Engineering, at Aditya College of Engineering is a bonafide work carried out by them under my guidance and supervision. The result embodied in this report has not been submitted to any other University or Institution for the award of any degree or diploma.

Project Guide
Mr. M. Kishore Kumar, M.Tech, (Ph.D).
Associate Professor,
Department of ECE
Aditya College of Engineering

Head of the Department
Mrs. CH. JANAKI DEVI, M.Tech, (Ph.D).
Associate Professor & HOD
Department of ECE
Aditya college of engineering

EXTERNAL EXAMINER

DECLARATION

We hereby declare that the entire project work embodied in this dissertation entitled “Fire Extinguisher Robot Car with Call Alert” has been independently carried out by us. As per our knowledge, no part of this work has been submitted for any degree in any institution, university, or organization previously. We hereby boldly state that to the best of our knowledge, our work is free from plagiarism.

Yours sincerely,

MANIKANTA KORIMILLI	**20MH1A04F6**
SANKHAPANI LOLABHATTU (TL)	**20MH1A04F8**
MOHANSAI MUDAPAKA	**20MH1A04G4**
JANAKI ORISSALA	**20MH1A04G8**
PASALA HARI SANDEEP	**21MH5A0459**

ACKNOWLEDGEMENT

We express our sincere gratitude and heartful thanks to the under stated person for the successful completion of our final project on **"Fire Extinguisher Robot Car with Call Alert".**

First and foremost we wish to thank our beloved guide **Mr. M. Kishore Kumar, M.Tech, (Ph.D),** Assoc.Prof. for his kind guidance, valuable advices and utmost care at every stage of our final project.

We would like to thank **Mrs. CH. JANAKI DEVI, M.Tech, (Ph.D),** Assoc.Prof. *Head of the Department of E.C.E* who have provided vital information, which was necessary for success of the project.

We would like to thank Prof. **Dr. A. RAMESH, M.Tech, Ph.D** principal of **ADITYA COLLEGE OF ENGINEERING** for his kind guidance, commendable and noteworthy advices and even memorable encouragement to the same.

Yours sincerely,

MANIKANTA KORIMILLI	**20MH1A04F6**
SANKHAPANI LOLABHATTU (TL)	**20MH1A04F8**
MOHANSAI MUDAPAKA	**20MH1A04G4**
JANAKI ORISSALA	**20MH1A04G8**
PASALA HARI SANDEEP	**21MH5A0459**

ABSTRACT

The development of an Arduino-based Fire Extinguisher Robot with Call Alert is a remarkable innovation aimed at creating a self-directed robot capable of detecting and extinguishing fires in indoor environments. By incorporating a GSM module, the robot can send real-time Call alerts to designated phone numbers, ensuring prompt notification of fire incidents.

This project emphasizes the need for automation, immediate communication, and potential customization and expansion, thereby significantly contributing to the field of fire safety. It is equipped with an ultrasonic sensor to avoid collisions, in addition to sensors that detect fire, smoke, and combustible gas. This autonomous system is capable of automatically identifying fire locations and extinguishing fires using stored water, showcasing a promising advancement in fire safety technology.

KEYWORDS: **Fire Safety, Real-Time communication, Call alarm, Fire Extinguisher, Arduino, IoT, Smoke Sensor, GSM module.**

INDEX

CHAPTER 4: **SYSTEM REQUIREMENT SPECIFICATION**

CHAPTER-1 INTRODUCTION

1.1 Project Overview

Nowadays, machinery and robotic design become important in helping human. This Fire Protection Robot was design to help people in any destructive burnt situation where this robot can extinguish burnt area immediately using autonomous system. This autonomous system will be designed using programming in PIC18F4550 and others additional circuit.

In real life, destructive burnt area often happens without our realization. Therefore, this type of robot will require a high demands in the market because of its usefulness to the human as well as the environment transmit fire information to cell phone using GSM modern.

The objective of the project will be to design a CALL ALERT electronic Fire Protection Robot toolkit which can replace the traditional Fire Protection Robot. The toolkit send the fire and send CALL ALERT to owner of the house, The system is made efficient by SIMs so that the CALL ALERT can be received by number of devices boards in a locality using techniques of time division multiple access.

The GSM modem receives the CALL ALERT. The AT commands are serially transferred to the modem. In return the modem transmits the stored message through the wireless link. The microcontroller validates the then perform specific task on the device.

1.2 Project Objectives

The objectives for this project are:

- To study a robot which can search, detect and extinguish burnt area immediately and develop a program using PIC18F4550 to control the movement of the robot. Besides, lean how to connect microcontroller and GSM modem.
- To design the robot that includes the flame sensor to detect the fire and than send notification by Short Message Service (CALL ALERT).
- To analyze how the robot performance to detect the angle of burnt area in front of the robot and detecting buritt area in 0m~2m in radius.

1.3 Project Scope

The project scopes for this project are:

- The robot detecting burnt area in Om-2m in radius.
- Robot detect fire event, and use extinguish to fight the fire source and the modem connected to the programmable device.
- The robot can turn 360 and than robot can extinguish fire at angle 30° from the fire extinguisher nozzle.

1.4 Problem Statement

The security of home, laboratory, office, factory and building is important to human life. We develop security system that contains a fire protection robot using sensor. The security system can detect abnormal and dangerous situation and notify us. First, we design a fire protection robot with extinguisher for the intelligent building. Besides, Human had difficulties to detect the small burnt cause by electrical appliances. The late time user takes to extinguish the fire. User may take a late time to extinguish fire like finding the water source to extinguish fire when want to extinguish the fire. The fire difficulties to detect the small burnt area and location that is hard to be reach by the user. Sometimes tough fire extinguished for example spaces are hard to see. Besides is cost the loss suffered in the event of fire slow to act.

PUNCA KEBAKARAN BANGUNAN NEGERI PULAU PINANG

BULAN JANUARI - OGOS TAHUN 2006

KOD	PUNCA	JUMLAH PANGGILAN	KERUGIAN (RM)	DISELAMATKAN (RM)
PK 1	LETRIK	76	95,584,070.00	180,271,030.00
PK 2	PUNTUNG ROKOK	7	204,000.00	1,234,000.00
PK 3	PERCIKAN API	6	2,542,000.00	51,421,000.00
PK 4	MERCUN / BUNGA API	2	5,000.00	20,000.00
PK 5	UBAT NYAMUK / COLOK / LILIN	21	1,317,700.00	7,387,957.70
PK 6	DAPUR GAS / MINYAK	19	944,460.00	73,061,450.00
PK 7	REAKSI SPONTAN	3	225,000.00	5,105,000.00
PK 8	SENGAJA DIBAKAR - NIAT BAIK	3	200.00	0.00
PK 8	SENGAJA DIBAKAR - NIAT JAHAT	17	900,500.00	2,920,500.00
PK 9	TIDAK DIKETAHUI	7	3,900,000.00	2,798,400.00
PK 10	TINDAK BALAS KIMIA	1	8,000,000.00	10,100,000.00
PK 11	BUDAK BERMAIN MANCIS	2	7,500.00	217,500.00
PK 12	LAIN-LAIN PUNCA	34	3,961,500.00	6,916,200.00
	JUMLAH	**198**	**117,591,930.00**	**341,456,037.70**

Figure 1.1 : Destructive burnt source for Penang

From figure 1.1, the destructive burnt cause by electrical is the highest source. From this table, the designing of Autonomous Fire Protection Robot with Notification must be suitable with this type of destructive burnt.

1.5 Expected Results

The expected results for this project are:

i. Autonomous searching, detecting and extinguish burnt area.

ii. Extinguish fire on the wall (315°) and 45° upper side.

iii. The robot can turn 360°.

iv. Send notification by Short Message Service (CALL ALERT) using GSM modem

CHAPTER 2
LITERATURE REVIEW

2.1 Introduction

This chapter are discussing about a study on the previous project based on fire fighter robot project and thesis. The entire project had been studied and analyzed their principles, method and applications.

2.2 Robot

Robot is a machine that looks like a human being and performs various complex tasks. Now, let's have a good look at existing firefighting robots.

Virtual Reality Simulation of Fire Fighting Robot [9] (Indonesia) is a virtual adaptation of competition robot, that took part in Panitia Kontes Robot Cerdas Indonesia competition in 2006. This system was developed in MATLAB/Simulink with the help of «Virtual Reality Toolbox» plug-in. It is oriented for initial testing of controlling algorithms. Its important to notice, that even the robot itself doesn't have enough level of functionality, because of low-detailed formalization of environment.

The robot could operate only in corridor-room environment, without strange objects. Only one fire source is meant and there are auxiliary marks on floor, that mean for example room entrance.

Pokey the Fire-Fighting Robot [10] (USA) is the firefighting robot, that made its way out of competitions, and became more "serious" than other systems. In [10] there are detailed description of used equipment and basic algorithms of operating. Robots operating environment is a building, so the robot is equipped with necessary sensors, for example, with a line sensor, that could be unuseful in conditions of dense smoke. The main advantages of robot are:

- using of two types of fire sensors, working in different ways;
- using of complex firefighting tool;

The main disadvantages are:

- short distance of sensor's work, the fire could be recognized at the distance not more than 1.5m. at longer distances the sensors works bad, ad developers say
- low efficiency of onboard computer, able only to carry main tasks, without its extension and complexation;
- absence of optical means of environment perception.

The device is described as autonomous mean of firefighting in houses and any civil buildings. Fire Protection Robot [11] (USA) another competition project, developed for 15th Annual Trinity College Fire Fighting Robot Competitions Robot has more complex organization, than one, shown above and is oriented for solving larger variety of tasks. The main system's advantages are:

- more complex algorithms, used for fire detection.
- using of sound sensor for activating.

The main disadvantages are:

- low-efficiency computer;
- low-power chassis;
- absence of home-return algorithm;
- absence of mapping;

Firefighting Robot [12] is an American Trinity College development, that was only on early-prototype stage (in 2008). It was supposed to this robot to be an autonomous device, with 15 minutes limited working time, after which it will return to the supply station. This approach is one of the best variants for firefighting in houses and non-industrial buildings. The main disadvantages are:

- the little working time;
- low-stock of "water":

The planning low-cost is a system's main advantage. In special order it's necessary to notice firefighting robots, included in Russian Ministry of Emergency Situations. Among them are "ABR-ROBOT", "El-4", "EI-10". These models are far away from competition projects, they armed with a real armour and firefighting tools, but their main disadvantage consists in remote controlling. They aren't autonomous. The

DESIGNER	YEAR	DESCRIPTION
Archytas	(347 BC)	A bird-shaped model propelled by jet of what was probably steam, said to have actually flown some 200 yards. This machine, which its inventor called The Pigeon, may have been suspended on a wire or pivot for its flight.
Al-Jazari	(1206)	A boat with four automatic musicians that floated on a lake to entertain guests at royal parties.
Leonardo da Vinci	(1519)	Further analysis of the plans has led some to believe that the robot would have been able to sit up, wave its arms and move its head and jaw. It is not known whether he attempted to build the robot.
	(1782)	A mechanical duck that was able to eat and digest grain, as well as flap its wings. Vaucanson gained celebrity across Europe for his constructs.
Pierre Jaquet Droz	(1790)	Animated dolls, or automata, to help sell watches and mechanical birds. He and his son created three dolls, each with a unique function. One can write, another plays music, and the third draws pictures. Some consider these devices to be the oldest examples of the computer.
Hisashige Tanaka	(1881)	Array of inventions including automatic gas lamps, clocks, and extremely complex mechanical toys, some of which were capable of serving tea, firing arrows drawn from a quiver, or even painting a Japanese kanji character.
Karel Čapek	(1938)	The word robot was introduced by Czech writer Karel Čapek in his play R.U.R. (Rossum's Universal Robots) premiered in 1920.

George Westinghouse	(1914)	A humanoid robot known as Elektro, exhibited at the 1939 and 940 World's Fairs. Seven feet tall, weighing 265 pounds, humanoid in appearance, it could walk by voice command, talk (using a 78-rpm record player).
Nikola Tesla	(1943)	Tesla used radio waves to move the craft in a small pool of water in Madison Square Garden, New York City during the Electrical Exhibition in 1898.
William Grey Walter	(1977)	The first electronic autonomous robot was created by William Grey Walter at Bristol University, England in 948. It was named Elsie, or the Bristol Tortoise. This robot could sense light and contact with external objects, and use these stimuli to navigate.
Jacques de Vaucanson		

2.3 Previous Robot Overview

The robot below shows the characteristic of the previous robot that have been similar with this robot project and used in the literature reviews:

2.3.1 Rolly Firefighter Robot by William Dubel, Hector Gongora, Kevin Bechtold, and Daisy Diaz

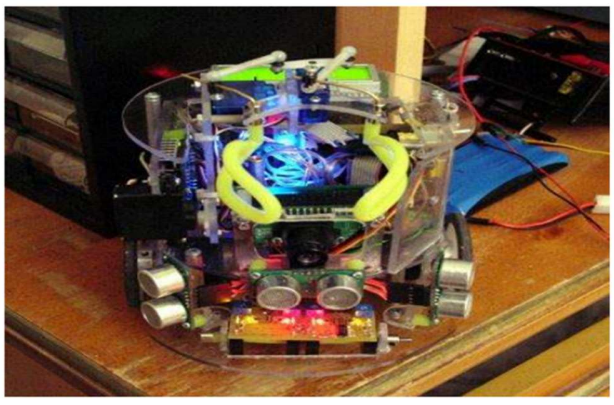


Figure 2.1: Rolly Firefighter Robot by William Dubel, Hector Gongora, Kevin Bechtold, and Daisy Diaz

This firefighting robot is designed to search for a fire in a small floor plan of a house, extinguish the fire (by placing a cup over the LEDs), and then return to the front of the house. The navigation of the robot throughout the house is achieved by data provided by a line tracker and ultrasound transducers. The deployment of the extinguishing device is implemented with a custom arm controlled by servos.

2.3.2 Fire Protection Robot by Viet Do, Ryan Norder, and Ryan Spraetz

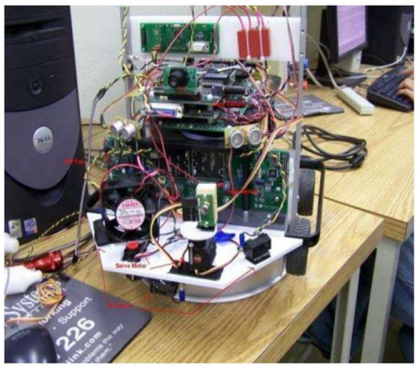


Figure 2.2: Fire Protection Robot by Viet Do, Ryan Norder, and Ryan Spraetz

This robot designed to enter a room and seek out a spot where there is extreme heat possibly due to a fire. Upon entering the room, the robot will once again use the color camera to pinpoint a spot where there is a large concentration of light. Once the robot has driven up to the light source, the heat sensor is activated to check and see if there is a large amount of heat being generated. If there is an excessive amount of heat generated, the fan is turned on and rotated quickly with a servo motor to put out the flame. If the flame is not put out the fan will turn on again and continue to blow on the flame. Once the flame is extinguished, the robot leaves the home.

2.3.3 Autonomous Mobile Robot: Recognize and Response to fire by Nik Md Hafizul hasmi B Md Suhaimi

Figure 2.3: Autonomous Mobile Robot: Recognize and Response to fire by Nik Md Hafizul hasmi B Md Suhaimi

This project will discuss about the development of a mobile robot which is can be train and control an autonomous robot that has a multifunction. The robot acquires basic navigation skills as well as the ability to detect a fire and to extinguish it. This robot is controlled by a microcontroller PIC16F84A and supported by RC circuits as driver for DC motors and other electronic components. This robot equipped with fire sensor that can be expand and attract so it can recognize and response to fire to operating water pump system. The battery monitoring circuit also equipped in this robot to make an easier to monitoring the overall robot battery power.

2.4 GSM Modem

GSM is a cellular network, which means that mobile phones connect to it by searching for cells in the immediate vicinity. Table 2.1 show the types of GSM Modem and used technology.

YEAR	STANDARD	MOBILE TELEPHONE SYSTEM	TECHNOLOGY	PRIMARY MARKETS

1981	NMT540	NORDIC MOBILE TELEPHONY	ANALOG UE	EUROPE,MIDDI E EAST
1985	TACS	TOTAL ACCESS COMMUNUNICATION SYSTEM	ANALOG UE	EUROPE AND CHINA
1986	NMT900	NORDIC MOBILE TELEPHONY	ANALOG ue	EUROPE, MIDDLE EAST
1991	GSM	GLOBAL SYSTEM FOR MOBILE COMMUNICATION	DIGITAL	WORLD-WIDE
1991	TDMA	TIME	DIVISION DIGITAL MULTIPLE ACCESS	AMERICA
1993	CDMA	CODE	DIVISION DIGITAL MULTIPLE ACCESS	NORTH AMERICA, KOREA
1992	GSM 1800	GLOBAL SYSTEM FOR MOBILE COMMUNICATION	DIGITAL	EUROPE
1994	PDC	PERSONAL DIGITAL CELLULAR	DIGITAL	JAPAN
1995	PCS 1900	PERSONAL COMPUTER SERVICES	DIGITAL	NORTH AMERICA

CHAPTER 3
METHODOLOGY

3.1 Introduction

There are many sections in the development of designing autonomous robot. These sections can be simplified using methodic approach.

3.1.1 Show the block diagram for this robot.

This block diagram shows how Autonomous fire fighter robot implements. The main function of this robot is to become an unmanned support vehicle, developed to detect and extinguish the fire. By using such robots, fire identification and rescue activities can be done with higher security without placing firefighters at high risk and in dangerous conditions. In other words, robots can reduce the need for firefighters to get into dangerous situations. Additionally, having a compact size and automatically controlled robot will also aid the extinguishing of fire in narrow spaces like tunnels or hazardous environments such as nuclear power plants.

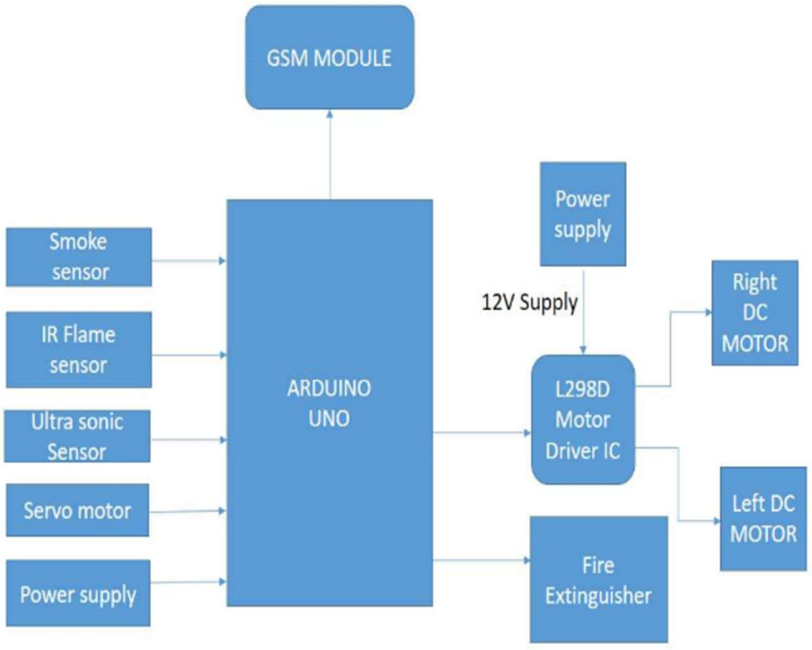


Figure 3.1: Block diagram of Autonomous Fire Protection Robot with Notification

3.1.2 Operation of Autonomous Fire Protection Robot with Notification

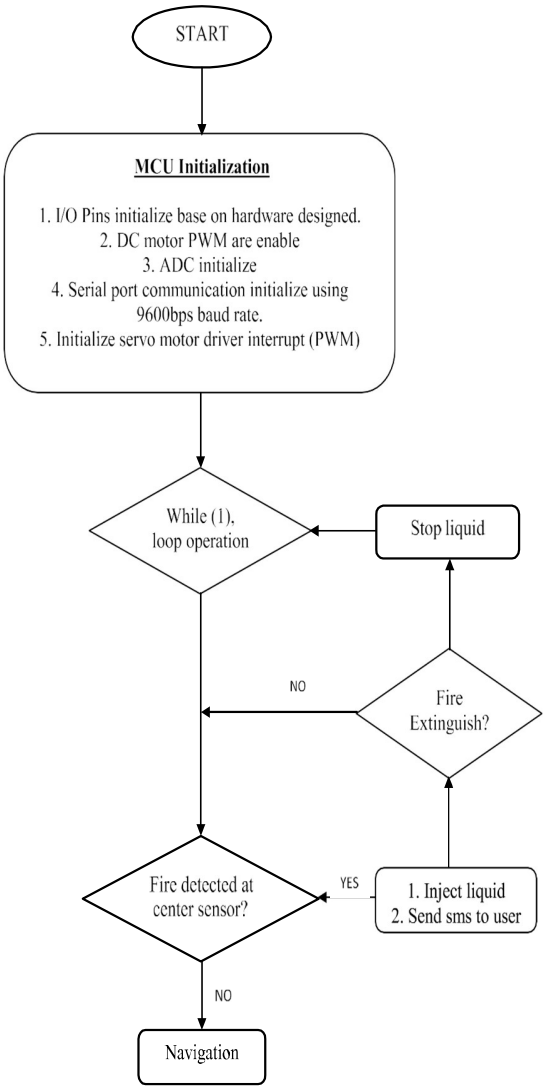


Figure 3.2: Show to how the operation of Autonomous Fire Protection Robot with Notification

The methodology flow chart first start the project and than analyze the project depend the objective and scope project. Describe the literature review of the project and define the implement of the methodology part. The system can be categorized into three parts at the methodology. First part mechanical part, second part is hardware part and last part the system is software part. Than combine the three of part and than testing for the complete flow system

at the figure 3.1(c).

The progress of designing an autonomous fire protection robot will be discussed in detail. Therefore, this chapter deals with the actual design and construction of the system. The system can be categorized into three parts as below:

1. Mechanical part,
2. Hardware part and simulations, and
3. Software part (microprogramming).

3.2 Phase 1- Mechanical Part Design

This sub-topic will discusses about the mechanical part design of the Autonomous Fire Fighter Robot. The body kit is used to protect the electronic circuit

This sub-topic will discusses about the mechanical part design of the Autonomous Fire Fighter Robot. The body kit is used to protect the electronic circuit from the any obstacles especially liquid where it may cause the electronic circuit malfunction. The designing of robot body kit were base on ideas below:

i. Base on the functions that the robot will perform.

ii. Determine where to place the internal components that will necessary to make the robot operational.

iii. Minimize the weight of load that the robot carrying to reduce the power needed by the robot.

iv. Minimize the gravity center for easy to spot the stability point while static or moving condition.

V. The microcontroller validates the then perform specific task on the device.

3.2.1 Robot Design Chassis

This robot contains two wheels at rear side and one free wheel at front side.

The free wheel used to stabilize the robot and use to rotate the robot 360°. Figure 3.2.2 shows the design of robot body kit using Solid Work 2007 SP2.0.

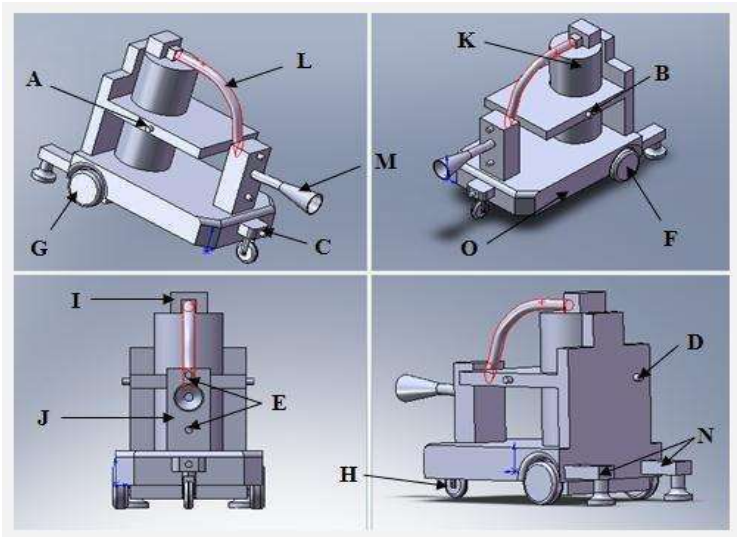


Figure 3.3: Design of robot body kit using Solid Work 2007 SP2.0

3.2.2 Body Kit Design Structure

The body kit is used to protect the electronic circuit from the any obstacles especially liquid where it may cause the electronic circuit malfunction for the robot.

The designing of the robot body kit was base on ideas below:

- Base on the functions that the robot will perform.
- Determine where to place the internal components that will necessary to make the robot operational.
- Minimize the weight of load that the robot carrying to reduce the power needed by

the robot.

- Minimize the gravity center for easy to spot the stability point while static or moving condition.

3.2.3 Steering Method

In order to tasking robot to finding and extinguish the destructive burnt area, the steering method is the important thing that must be emphasis. These methods help the Autonomous Fire Fighter Robot to achieved objectives to extinguished fire fully. Table 3.1 below shows the detected sensor and its future task in commanding the motor that rotate wheel.

Table 3.1: The detected sensor and future task

SENSOR	TASK
Left	Rotate right wheel till front sensor detected flame
Right	Rotate left wheel till front sensor detected flame

CHAPTER 4

SYSTEM REQUIREMENT SPECIFICATION

4.1 HARDWARE REQUIREMENTS

4.1.1 L293D Driver module

The Motor Driver is a module for motors that allows you to control the working speed and direction of two motors simultaneously. This Motor Driver is designed and developed based on L293D IC.L293D is a 16 Pin Motor Driver IC. This is designed to provide bidirectional drive currents at voltages from 5 V to 36 V.

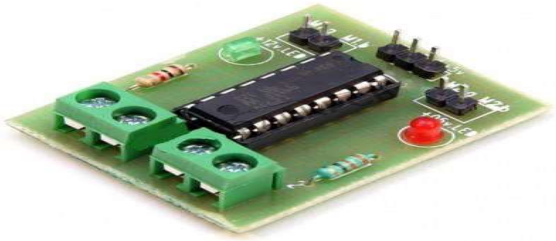


Figure 4.1: L293D motor driver module

L293D is a dual H-bridge motor driver integrated circuit (IC). Motor drivers act as current amplifiers since they take a low-current control signal and provide a higher-current signal. This higher current signal is used to drive the motors.L293D contains two inbuilt H-bridge driver circuits. In its common mode of operation, two DC motors can be driven simultaneously, both in forward and reverse direction. The motor operations of two motors can be controlled by input logic at pins 2 & 7 and 10 & 15. Input logic 00 or 11 will stop the corresponding motor. Logic 01 and 10 will rotate it in clockwise and anticlockwise directions, respectively.

Enable pins 1 and 9 (corresponding to the two motors) must be high for motors to start operating. When an enable input is high, the associated driver gets enabled. As a result, the outputs become active and work in phase with their inputs.

4.1.1.1 Features of L293D DRIVER MODULE

- can be used to run Two DC motors with the same IC.
- Speed and Direction control is possible
- Motor voltage Vcc2 (Vs): 4.5V to 36V
- Maximum Peak motor current: 1.2A
- Maximum Continuous Motor Current: 600mA
- Supply Voltage to Vcc1(vss): 4.5V to 7V
- Transition time: 300ns (at 5Vand 24V)
- Automatic Thermal shutdown is available
- Available in 16-pin DIP, TSSOP, SOIC packages

Applications

- Used to drive high current Motors using Digital Circuits
- Can be used to drive Stepper motors
- High current LED's can be driven
- Relay Driver module (Latching Relay is possible)

4.1.1.2 Pin description

The L293D driver module has 16pins. They are as follows:

ENABLE:

When enable is pulled low, the module is disabled which means the module will not turn on and it fails to drive motors. When enable is left open or connected to 3.3V, the module is enabled i.e the module remains on and driving of motors also takes place.

VCC:

Supply voltage 3.3v to 5v

GND:

Ground pin.

INPUT & OUTPUT:

These two pins acts as an input and output interface for communication.

Table pin description table

Pin No	Function	Name
1	Enable pin for Motor 1; active high	Enable 1,2
2	Input 1 for Motor 1	Input 1
3	Output 1 for Motor 1	Output 1
4	Ground (0V)	Ground
5	Ground (0V)	Ground
6	Output 2 for Motor 1	Output 2
7	Input 2 for Motor 1	Input 2
8	Supply voltage for Motors; 9-12V (up to 36V)	Vcc_2
9	Enable pin for Motor 2; active high	Enable 3,4
10	Input 1 for Motor 1	Input 3
11	Output 1 for Motor 1	Output 3
12	Ground (0V)	Ground
13	Ground (0V)	Ground
14	Output 2 for Motor 1	Output 4
15	Input2 for Motor 1	Input 4
16	Supply voltage; 5V (up to 36V)	Vcc_1

4.1.2 Flame Sensor Module

A flame sensor module that consists of a flame sensor (IR receiver), resistor, capacitor, potentiometer, and comparator LM393 in an integrated circuit. It can detect infrared light with a wavelength ranging from 700nm to 1000nm.The far-infrared flame probe converts the light detected in the form of infrared light into current changes. Sensitivity is adjusted through the onboard variable resistor with a detection angle of 60 degrees.

Working voltage is between 3.3v and 5.2v DC, with a digital output to indicate the presence of a signal. Sensing is conditioned by an LM393 comparator.

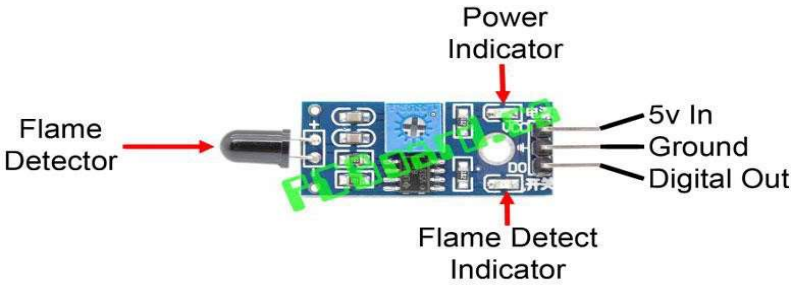


Figure 4.2: Flame Sensor Module

4.1.3 Photo diode

A photodiode is one type of light detector, used to convert the light into current or voltage based on the mode of operation of the device. It comprises of optical filters, built-in lenses and also surface areas. These diodes have a slow response time when the surface area of the photodiode increases. Photodiodes are alike to regular semiconductor diodes, but that they may be either visible to let light reach the delicate part of the device. Several diodes intended foruse exactly as a photodiode will also use a PIN junction somewhat than the usual PN junction.

Some photodiodes will look like a light emitting diode. They have two terminals coming from the end. The smaller end of the diode is the cathode terminal, while the longer end of the diode is the anode terminal. See the following schematic diagram for the anode and cathode side. Under forward bias condition, conventional current will flow from the anode to the cathode, following the arrow in the diode symbol. Photocurrent flows in the reverse direction.

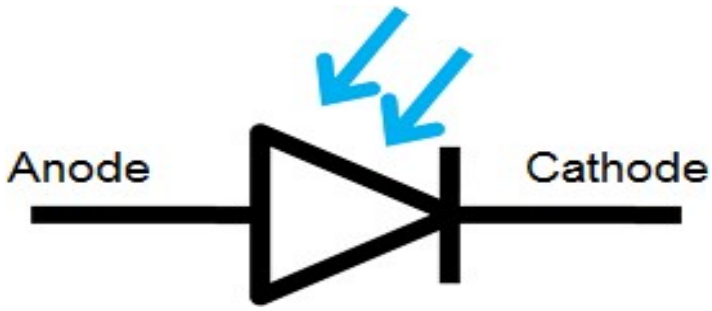


Figure 4.3: Photo diode symbol

4.1.3.1 Applications of Photodiode

- The applications of photodiodes involve in similar applications of photodetectors like charge-coupled devices, photoconductors, and photomultiplier tubes.
- These diodes are used in consumer electronics devices like smoke detectors, compact disc players, and televisions and remote controls in VCRs.
- In other consumer devices like clock radios, camera light meters, and street lights, photoconductors are more frequently used rather than photodiodes.

- Photodiodes are frequently used for exact measurement of the intensity of light in science & industry. Generally, they have an enhanced, more linear response than photoconductors.
- Photodiodes are also widely used in numerous medical applications like instruments to analyze samples, detectors for computed tomography and also used in blood gas monitors.

4.1.4 LM393 Comparator

For sensors, there are normally two ways as output, the analog value or digital value output.

- **Analog value**: Most sensors only provide the analog value, so it outputs a voltage value to indicate the sensing parameters. Arduino read this value on A0 to A9, and from 0 till 1023. In AVR, the analog voltage varies from 0V till 5V. We sign AO (Analog Output) as a pin name on many sensor boards
- **Digital Value**: Sometimes we only want the sensors only give feedback when the sensing value read a threshold that we want, so when it reached the feedback is 1, and 0 vice verse.

Here the LM393 IC do the voltage comparing here, a reference voltage (UR) is set by the adjustable potentiometer, when the analog output value over this value, the LM393 will output a digital value to indicate this sensor is triggered by reaching this setup threshold.

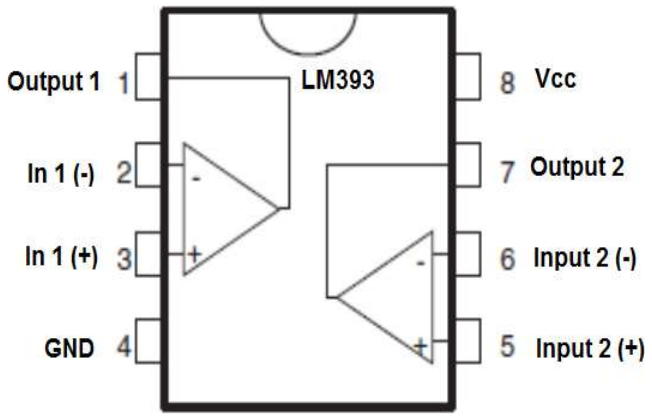


Figure 4.4: Circuit diagram of LM393 comparator

4.1.5 DC Motor:

Motors convert electrical energy into mechanical energy. A **DC motor** is an electric motor that runs on direct current (DC) electricity.

Figure 4.5: DC Motor

In any electric motor, operation is based on simple electromagnetism. A current-carrying conductor generates a magnetic field; when this is then placed in an external magnetic field, it will experience a force proportional to the current in the conductor, and to the strength of the external magnetic field. As you are well aware of from playing with magnets as a kid, opposite (North and South) polarities attract, while like polarities (North and North, South and South) repel. The internal configuration of a DC motor is designed to harness the magnetic interaction between a current-carrying conductor and an external magnetic field to generate rotational motion Direct current (DC) motors are widely used to generate motion in a variety of products. Permanent magnet DC (direct current) motors are enjoying increasing popularity in applications requiring compact size, high torque, high efficiency, and low power consumption.

In a brushed DC motor, the brushes make mechanical contact with a set of electrical contacts provided on a commutator secured to an armature, forming an electrical circuit between the DC electrical source and coil windings on the armature. As the armature rotates on an axis, the stationary brushes come into contact with different sections of the rotating commutator.

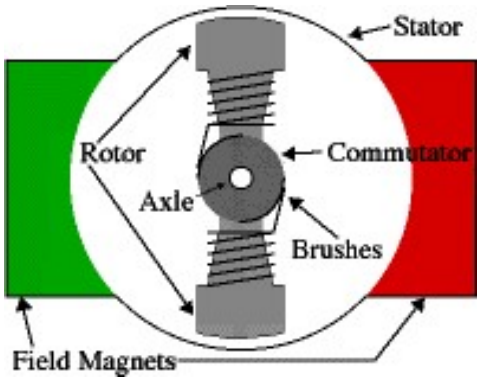


Figure 4.6: Internal architecture of dc motor

Permanent magnet DC motors utilize two or more brushes contacting a commutator which provides the direct current flow to the windings of the rotor, which in turn provide the desired magnetic repulsion/attraction with the permanent magnets located around the periphery of the motor.

The brushes are conventionally located in brush boxes and utilize a U-shaped spring which biases the brush into contact with the commutator. Permanent magnet brushless dc motors are widely used in a variety of applications due to their simplicity of design, high efficiency, and low noise. These motors operate by electronic commutation of stator windings rather than the conventional mechanical commutation accomplished by the pressing engagement of brushes against a rotating commutator.

A brushless DC motor basically consists of a shaft, a rotor assembly equipped with one or more permanent magnets arranged on the shaft, and a stator assembly which incorporates a stator component and phase windings. Rotating magnetic fields are formed by the currents applied to the coils.

The rotator is formed of at least one permanent magnet surrounded by the stator, wherein the rotator rotates within the stator. Two bearings are mounted at an axial distance to each other on the shaft to support the rotor assembly and stator assembly relative to each other. To achieve electronic commutation, brushless dc motor designs usually include an electronic controller for controlling the excitation of the stator windings.

4.1.6 Water Pump

The water pump is operated at 5v which can be interfaced with Arduino

Figure 4.7: 5v water pump

4.1.7 Servo Motor

A servo is a small DC motor with the following components added: some gear reduction, a position sensor on the motor shaft, and an electronic circuit that controls the motor's operation. In other words, a servo is to a DC motor what the Arduino is the ATmega microcontroller---components and housing that make the motor easy to use. This will become abundantly clear when we work with unadorned DC motors next week.

The gear reduction provided in a servo is large; the basic hobby servo has a 180:1 gear ratio. This means that the DC motor shaft must make 180 revolutions to produce 1 revolution of the servo shaft. This large gear ratio reduces the speed of the servo and proportionately increases its torque. What does this imply about small DC motors? Servo motors are typically used for angular positioning, such as in radio control airplanes. They have a movement range of 0 up to 180 degrees, but some extend up to 210 degrees. Typically, a potentiometer measures the position of the output shaft at all times so the controller can accurately place and maintain its position.

Figure 4.8: Servo motor

PPM uses 1 to 2ms out of a 20ms timeperiod to encode its information. The servo expects to see a pulse every 20milliseconds (.02 seconds). The length of the pulse will determine how far the motor turns. A 1.5 millisecond pulse will make the motor turn to the 90 degree position (often called the neutral position). If the pulse is shorter than 1.5 ms, then the motor will turn the shaft to closer to 0degrees. If the pulse is longer than 1.5ms, the shaft turns closer to 180 degrees. The amount of power applied to the motor is proportional to the distance it needs to travel. So, if the shaft needs to turn a large distance, the motor will run at full speed. If it needs to turn only a small amount, the motor will run at a slower speed.

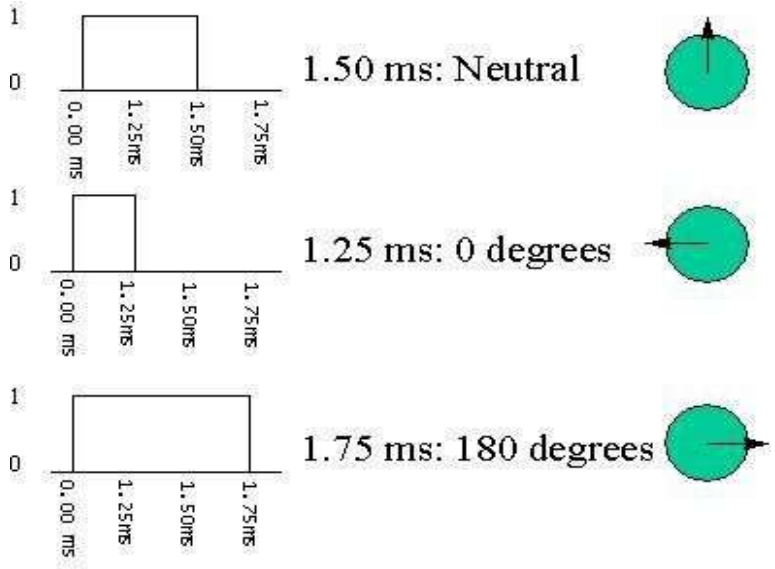


Figure 4.9: Show the Time periods

4.2 MICROCONTROLLER ATMEGA 328

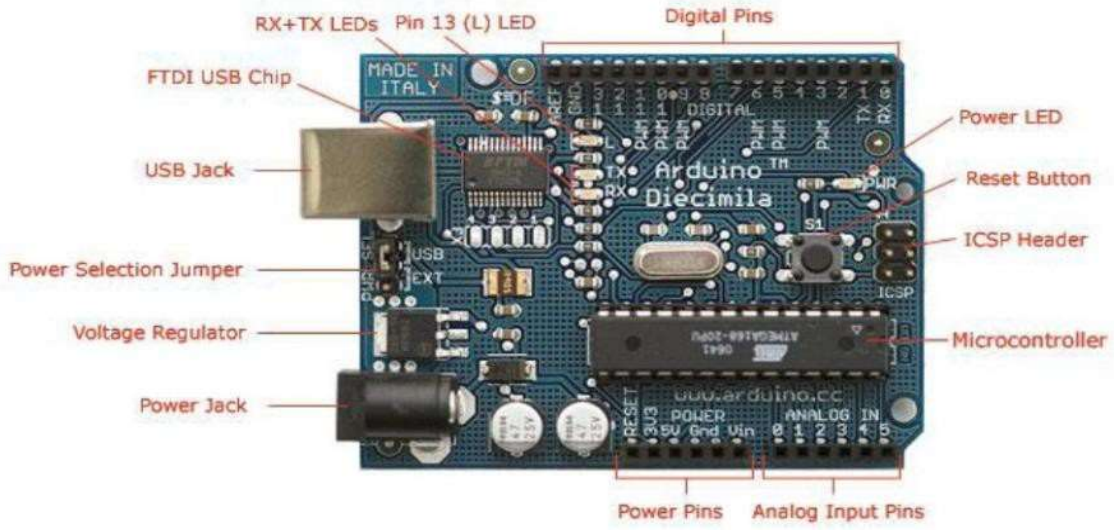


Figure 4.10: Arduino Mega board

The Atmel 8-bit AVR RISC-based microcontroller combines 32 KB ISP flash memory with read-while-write capabilities, 1 KB EEPROM, 2 KB SRAM, 23 general purpose I/O lines, 32 general purpose working registers, three flexible timer/counters with compare modes, internal and external interrupts, serial programmable USART, a byte- oriented 2-wire serial interface, SPI serial port, 6-channel 10-bit A/D converter (8-channels in TQFP and QFN/MLF packages), programmable watchdog timer with internal oscillator, and five software selectable power saving modes. The device operates between 1.8-5.5 volts. The device achieves throughputs approaching 1 MIPS.

4.2.1 Applications

Today the ATmega328 is commonly used in many projects and autonomous systems where a simple, low-powered, low-cost micro-controller is needed. Perhaps the most common implementation of this chip is on the popular Arduino development platform, namely the

Arduino Uno and Arduino Nano models.

4.2.2 Features

- 28-pin AVR Microcontroller
- Flash Program Memory: 32 kbytes
- EEPROM Data Memory: 1 kbytes
- SRAM Data Memory: 2 kbytes
- I/O Pins: 23
- Timers: Two 8-bit / One 16-bit
- A/D Converter: 10-bit Six Channel
- PWM: Six Channels
- RTC: Yes with Separate Oscillator
- MSSP: SPI and I²C Master and Slave Support
- USART: Yes
- External Oscillator: up to 20MHz

The Atmega328 is a very popular microcontroller chip produced by Atmel. It is an 8-bit microcontroller that has 32K of flash memory, 1K of EEPROM, and 2K of internal SRAM.

The Atmega328 is one of the microcontroller chips that are used with the popular Arduino Duemilanove boards. The Arduino Duemilanove board comes with either 1 of 2 microcontroller chips, the Atmega168 or the Atmega328. Of these 2, the Atmega328 is the upgraded, more advanced chip. Unlike the Atmega168 which has 16K of flash program memory and 512 bytes of internal SRAM, the Atmega328 has 32K of flash program memory and 2K of Internal SRAM.

The Atmega328 has 28 pins, It has 14 digital I/O pins, of which 6 can be used as PWM outputs and 6 analog input pins. These I/O pins account for 20 of the pins.

4.3 PIN DIAGRAM OF ATMEGA328

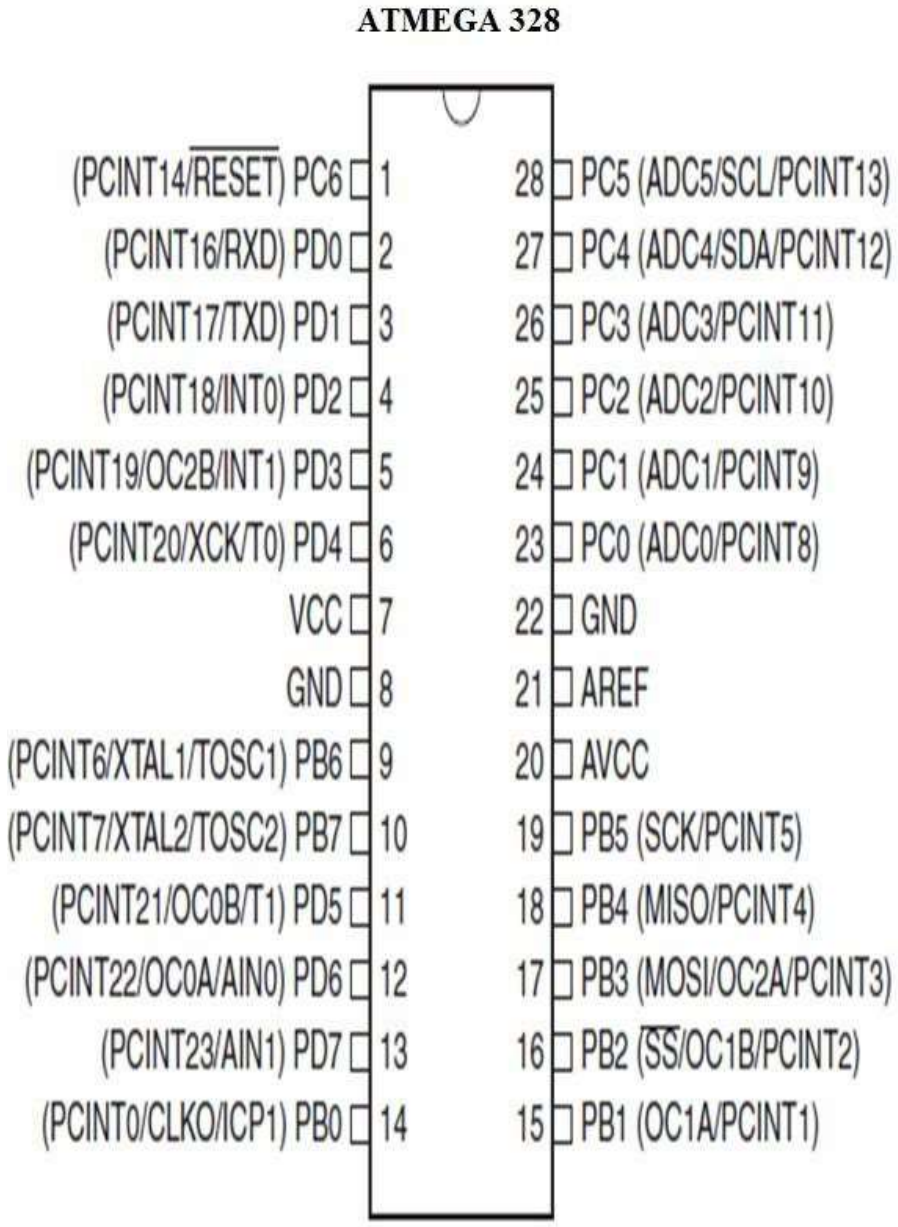


Figure 4.11: Pin diagram of Atmega328

Table 3.3: Description of each pins of ATmega328

Pin Number	Description	Function
1	PC6	Reset
2	PD0	Digital Pin (RX)
3	PD1	Digital Pin (TX)
4	PD2	Digital Pin
5	PD3	Digital Pin (PWM)
6	PD4	Digital Pin
7	Vcc	Positive Voltage (Power)
8	GND	Ground
9	XTAL 1	Crystal Oscillator
10	XTAL 2	Crystal Oscillator
11	PD5	Digital Pin (PWM)
12	PD6	Digital Pin (PWM)
13	PD7	Digital Pin
14	PB0	Digital Pin
15	PB1	Digital Pin (PWM)
16	PB2	Digital Pin (PWM)
17	PB3	Digital Pin (PWM)
18	PB4	Digital Pin
19	PB5	Digital Pin
20	AVcc	Positive voltage for ADC (power)
21	Aref	Reference Voltage
22	GND	Ground
23	PC0	Analog Input
24	PC1	Analog Input
25	PC2	Analog Input
26	PC3	Analog Input
27	PC4	Analog Input
28	PC5	Analog Input

As stated before, 20 of the pins function as I/O ports. This means they can function as an input to the circuit or as output. Whether they are input or output is set in the software. 14 of the pins are digital pins, of which 6 can function to give PWM output. 6 of the pins are for analog input/output. Two of the pins are for the crystal oscillator, this is to provide a clock pulse for the Atmega chip. A clock pulse is needed for synchronization so that communication can occur in synchrony between the Atmega chip and a device that it is connected.

The Atmega328 chip has an analog-to-digital converter (ADC) inside of it. This must be or else the Atmega328 wouldn't be capable of interpreting analog signals. Because there is an ADC, the chip can interpret analog input, which is why the chip has 6 pins for analog input. The ADC has 3 pins set aside for it to function- AVCC, AREF, and GND. AVCC is the power supply, positive voltage, that for the ADC. The ADC needs its own power supply in order to work. GND is the power supply ground. AREF is the reference voltage that the ADC uses to convert an analog signal to its corresponding digital value. Analog voltages higher than the reference voltage will be assigned to a digital value of 1, while analog voltages below the reference voltage will be assigned the digital value of 0. Since the ADC for the Atmega328 is a 10-bit ADC, meaning it produces a 10-bit digital value, it converts an analog signal to its digital value, with the AREF value being a reference for which digital values are high or low. Thus, a portrait of an analog signal is shown by this digital value; thus, it is its digital correspondent value. The last pin is the RESET pin. This allows a program to be rerun and start over. And this sums up the pin out of an Atmega328 chip.

4.4 ARCHITECTURE

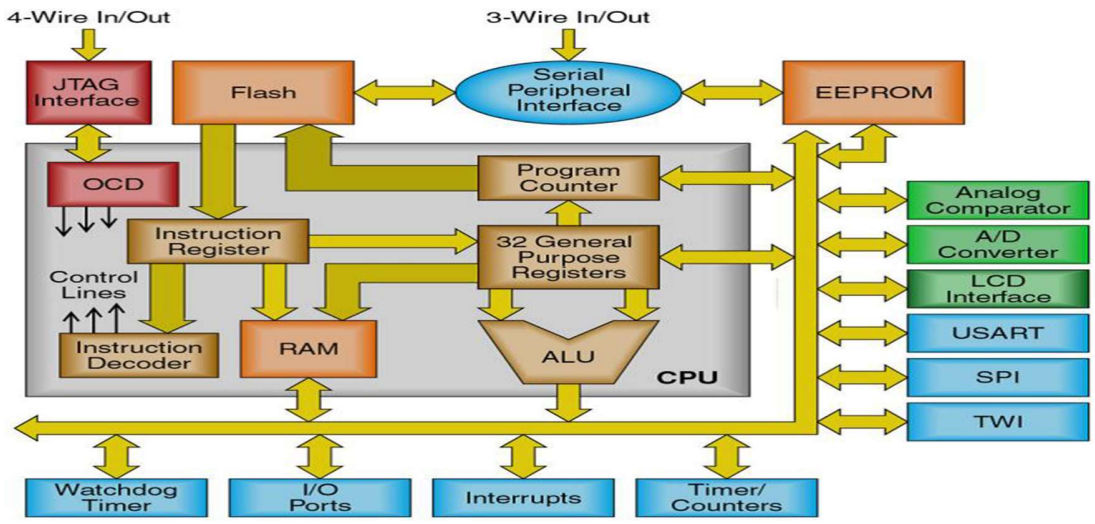


Figure 4.12: Architecture of AVR

4.5 FEATURES OF AVR.

• High-performance, Low-power AVR® 8-bit Microcontroller
• Advanced RISC Architecture
– 131 Powerful Instructions – Most Single-clock Cycle Execution
– 32 x 8 General Purpose Working Registers
– Fully Static Operation
– Up to 16 MIPS Throughput at 16 MHz
– On-chip 2-cycle Multiplier
• Nonvolatile Program and Data Memories
– 32K Bytes of In-System Self-Programmable Flash
• Endurance: 10,000 Write/Erase Cycles
– Optional Boot Code Section with Independent Lock Bits
• In-System Programming by On-chip Boot Program
• True Read-While-Write Operation
– 1024 Bytes EEPROM
• Endurance: 100,000 Write/Erase Cycles
– 2K Byte Internal SRAM
– Programming Lock for Software Security
• JTAG (IEEE std. 1149.1 Compliant) Interface
– Boundary-scan Capabilities According to the JTAG Standard
– Extensive On-chip Debug Support
– Programming of Flash, EEPROM, Fuses, and Lock Bits through the JTAG Interface
• Peripheral Features
– Two 8-bit Timer/Counters with Separate Pre scalers and Compare Modes
– One 16-bit Timer/Counter with Separate Pre scaler, Compare Mode, and Capture Mode
– Real Time Counter with Separate Oscillator
– Four PWM Channels
– 8-channel, 10-bit ADC
• 8 Single-ended Channels

• 2 Differential Channels with Programmable Gain at 1x, 10x, or 200x
– Byte-oriented Two-wire Serial Interface
– Programmable Serial USART
– Master/Slave SPI Serial Interface
– Programmable Watchdog Timer with Separate On-chip Oscillator
– On-chip Analog Comparator
• Special Microcontroller Features
– Power-on Reset and Programmable Brown-out Detection
– Internal Calibrated RC Oscillator
– External and Internal Interrupt Sources
– Six Sleep Modes: Idle, ADC Noise Reduction, Power-save, Power-down, Standby and Extended Standby
• I/O and Packages
– 32 Programmable I/O Lines
– 40-pin PDIP, 44-lead TQFP, and 44-pad QFN/MLF
• Operating Voltages
– 2.7 - 5.5V for ATmega32L
– 4.5 - 5.5V for ATmega32
• Speed Grades
– 0 - 8 MHz for ATmega32L
– 0 - 16 MHz for ATmega32
• Power Consumption at 1 MHz, 3V, 25°C for ATmega32L
– Active: 1.1 mA
– Idle Mode: 0.35 mA
– Power-down Mode: < 1 μA

4.6 DIFFERENT FLAVOURS OF ARDUINO WITH THEIR CONFIGURATION

Table 3.4: Different Flavors of ARDUINO with their Configuration

Arduino	Processor	Flash KiB	EEPROM KiB	SRAM KiB	Digital I/O pins	...with PWM	Analog input pins	USB Interface type	Dimensions inches	Dimensions mm
Diecimila	ATmega168	16	0.5	1	14	6	6	FTDI	2.7" x 2.1"	68.6mm x 53.3mm
Duemilanove	ATmega168/328P	16/32	0.5/1	1/2	14	6	6	FTDI	2.7" x 2.1"	68.6mm x 53.3mm
Uno	ATmega328P	32	1	2	14	6	6	ATmega8U2	2.7" x 2.1"	68.6mm x 53.3mm
Mega	ATmega1280	128	4	8	54	14	16	FTDI	4" x 2.1"	101.6mm x 53.3mm
Mega2560	ATmega2560	256	4	8	54	14	16	ATmega8U2	4" x 2.1"	101.6mm x 53.3mm
Fio	ATmega328P	32	1	2	14	6	8	None	1.6" x 1.1"	40.6mm x 27.9mm
Nano	ATmega168 or ATmega328	16/32	0.5/1	1/2	14	6	8	FTDI	1.70" x 0.73"	43mm x 18mm
LilyPad	ATmega168V or ATmega328V	16	0.5	1	14	6	6	None	2" ø	50mm ø

4.7 BASIC TERMINOLOGIES IN ARDUINO

4.7.1 Analog to Digital Converter (ADC)

- The process of Analog to digital conversion is shown in figure.
- The Arduino has 10 bits of Resolution when reading analog signals.
- 2^{10}=1024 increments.
- Influence also by how fast you sample.

4.7.2 Pulse Width Modulation (PWM)

- The Arduino has 8bit of resolution, when outputting a signal using PWM.
- The range of output voltage is from 0 to 5 Volts.
- 2^{8}=255 Increments
- Average of on/off(digital signals to make an average voltage),Duty cycle in 100% of 5Volts.

4.8 LANGUAGE REFERENCES:

The Microcontroller on the board is programmed using the Arduino programming language (based on wiring) and the arduino development environment (based on processing).

4.8.1 Arduino programming language (APL) (based on wiring)

The Arduino programming language is an implementation of wiring, a similar physical computing platform, which is based on the Processing multimedia programming environment.

4.8.2 Wiring

Wiring is an open-source programming framework for microcontrollers. Wiring allows writing cross-platform software to control devices attached to a wide range of

microcontroller boards to create all kinds of creative coding, interactive objects, spaces or physical experiences. The framework is thoughtfully created with designers and artists in mind to encourage a community where beginners through experts from around the world share ideas, knowledge and their collective experience. There are thousands of students, artists, designers, researchers, and hobbyists who use Wiring for learning, prototyping, and finished professional work production.

4.9 ARDUINO DEVELOPMENT ENVRONMENT

4.9.1 Processing

Processing is an open source programming language and environment for people who want to create images, animations, and interactions. Initially developed to serve as a software sketchbook and to teach fundamentals of computer programming within a visual context, Processing also has evolved into a tool for generating finished professional work. Today, there are tens of thousands of students, artists, designers, researchers, and hobbyists who use Processing for learning, prototyping, and production.

4.9.2 Embedded C

Embedded C is a set of language extensions for the C Programming language by the C Standards committee to address commonality issues that exist between C extensions for different embedded systems. Historically, embedded C programming requires nonstandard extensions to the C language in order to support exotic features such as fixed- point arithmetic, multiple distinct memory banks, and basic I/O operations.

4.9.3 Difference between C and Embedded C

Though C and embedded C appear different and are used in different contexts, they have more similarities than the differences. Most of the constructs are same; the difference lies in their applications.

C is used for desktop computers, while embedded C is for microcontroller based applications. C takes more resources of a desktop PC like memory, OS, etc. while programming on desktop systems what embedded C cannot. Embedded C has to use the limited resources (RAM, ROM, I/O's) on an embedded processor. Thus, program code must fit into the available program memory. If code exceeds the limit, the system is likely to crash.

Compilers for C (ANSI C) typically generate OS dependent executable files. Embedded C requires compilers to create files to be downloaded to the microcontrollers/microprocessors where it needs to run. Embedded compilers give access to all resources which is not provided in compilers for desktop computer applications.

Embedded systems often have the real-time constraints, which is usually not there with desktop computer applications.

Embedded systems often do not have a console, which is available in case of desktop applications.

The C programming language is perhaps the most popular programming language for programming embedded systems. C continues to be a very popular language for micro-controller developers/programmers due to the code efficiency and reduced overhead and development time. C offers low-level control and is considered more readable than assembly language which is a little difficult to understand. Assembly language requires more code writing, whereas C is easy to understand and requires less coding. Plus, using C increases portability, since C code can be compiled for different types of processors. We can program microcontrollers using Atmel Atmega328, AVR or PIC.

Here by developing the programs as per the electronic hardware using Atmel Atmega328 micro controller. For the operations like: blink led, increment decrement counters, token displays etc.

Most C programmers are spoiled because they program in environments where not only there is a standard library implementation, but there are frequently a number of other

libraries available for use. The cold fact is, that in embedded systems, there rarely are many of the libraries that programmers have grown used to, but occasionally an embedded system might not have a complete standard library, if there is a standard library at all. Few embedded systems have capability for dynamic linking, so if standard library functions are to be available at all, they often need to be directly linked into the executable. Oftentimes, because of space concerns, it is not possible to link in an entire library file, and programmers are often forced to "brew their own" standard c library implementations if they want to use them at all. While some libraries are bulky and not well suited for use on microcontrollers, many development systems still include the standard libraries which are the most common for C programmers.

C remains a very popular language for micro-controller developers due to the code efficiency and reduced overhead and development time. C offers low-level control and is considered more readable than assembly. Many free C compilers are available for a wide variety of development platforms. The compilers are part of an IDEs with ICD support, breakpoints, single-stepping and an assembly window. The performance of C compilers has improved considerably in recent years, and they are claimed to be more or less as good as assembly, depending on who you ask. Most tools now offer options for customizing the compiler optimization. Additionally, using C increases portability, since C code can be compiled for different types of processors.

4.9.4 Software

The software used by the arduino is Arduino IDE. The Arduino IDE is a cross-platform application written in Java, and is derived from the IDE for the Processing programming language and the Wiring project. It is designed to introduce programming to artists and other newcomers unfamiliar with software development. It includes a code editor with features such as syntax highlighting, brace matching, and automatic indentation, and is also capable of compiling and uploading programs to the board with a single click. There is typically no need to edit make files or run programs on a command- line interface. Although building on command-line is possible if required with some third-party tools such as Ino.

The Arduino IDE comes with a C/C++ library called "Wiring" (from the project of the same name), which makes many common input/output operations much easier. Arduino programs are written in C/C++, although users only need define two functions to make a runnable program:

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

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

A typical first program for a microcontroller simply blinks a LED on and off. In the Arduino environment, the user might write a program like this:

```
#define LED_PIN 13
void setup (){
pinMode(LED_PIN, OUTPUT);// enable pin 13 for digital output
}
void loop (){
digitalWrite(LED_PIN, HIGH);// turn on the LED
delay(1000);// wait one second (1000 milliseconds)
```

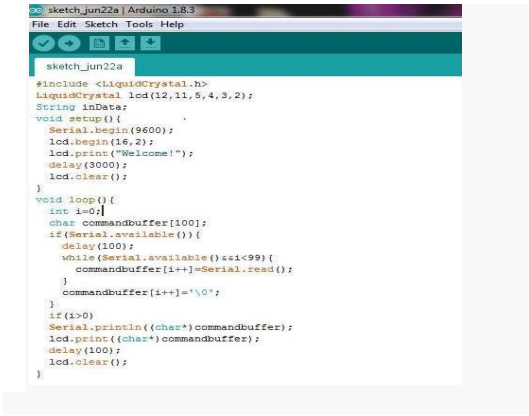


Figure 4.13: A Screenshot of Arduino IDE

For the above code to work correctly, the positive side of the LED must be connected to pin 13 and the negative side of the LED must be connected to ground. The above code would not be seen by a standard C++ compiler as a valid program, so when the user clicks the "Upload to I/O board" button in the IDE, a copy of the code is written to a temporary file with an extra include header at the top and a very simple main() function at the bottom, to make it a valid C++ program.

The Arduino IDE uses the GNU tool chain and AVR Libc to compile programs, and uses AVR dude to upload programs to the board.

For educational purposes there is third party graphical development environment called Mini blog available under a different open source license.

PROGRAM COMPILING

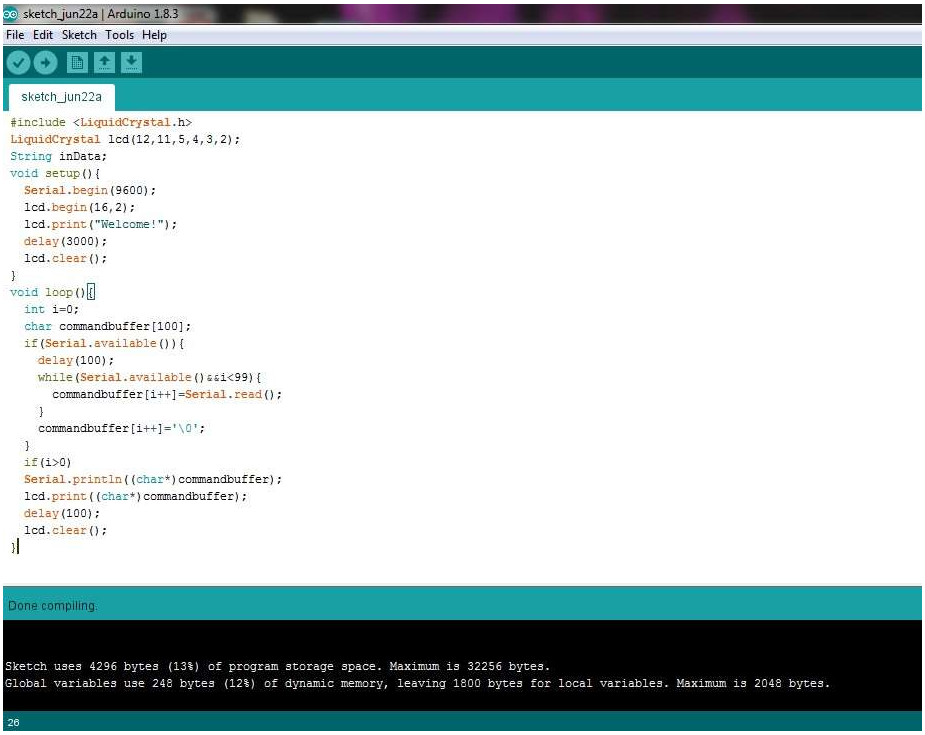


Figure 4.14: Program compiling using arduino IDE.

SELECTING BOARD

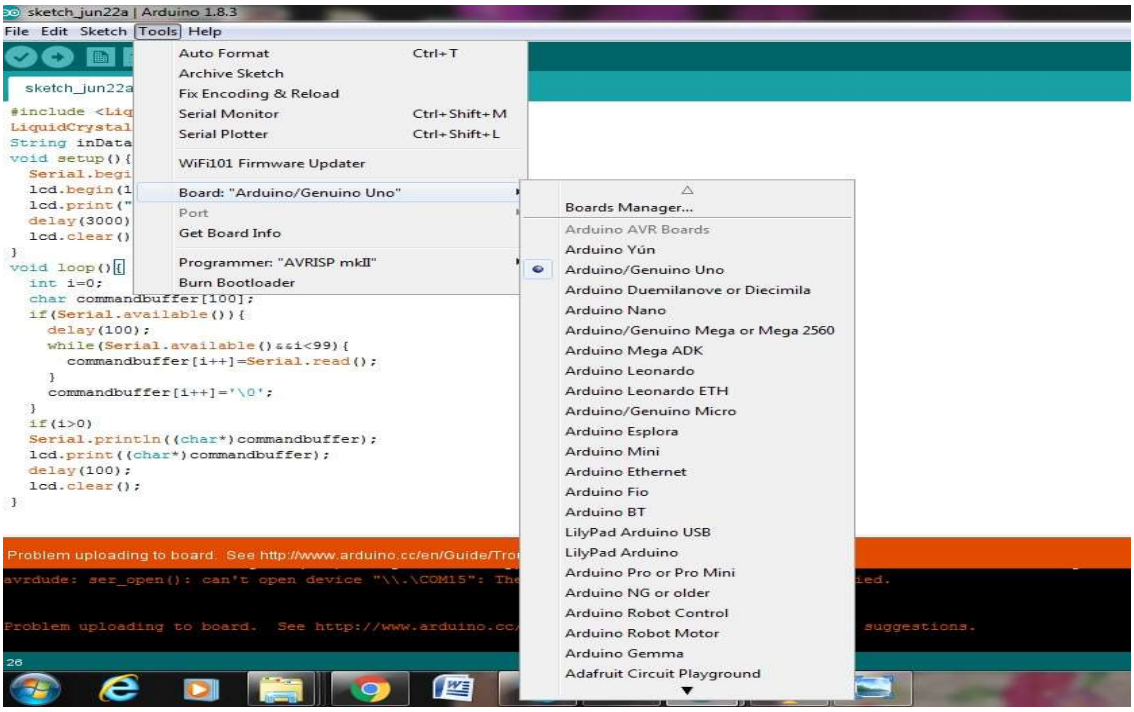


Figure 4.15: Selecting the board from Tools menu

SELECTING PORT

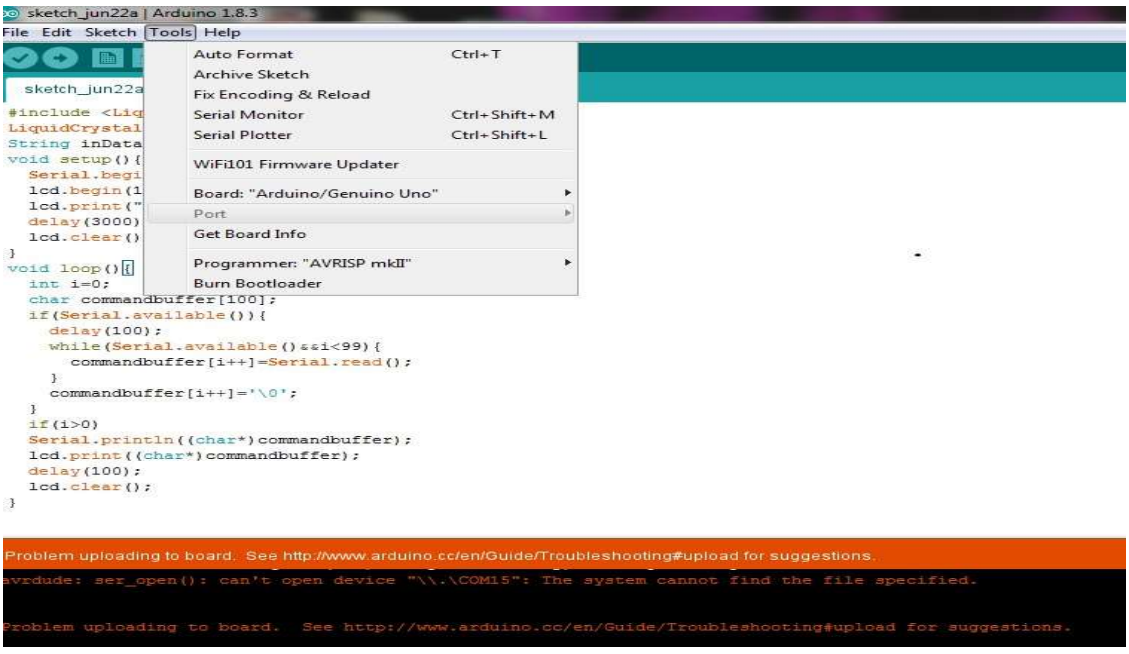


Figure 4.16: Selecting the port

UPLOADING PROGRAM

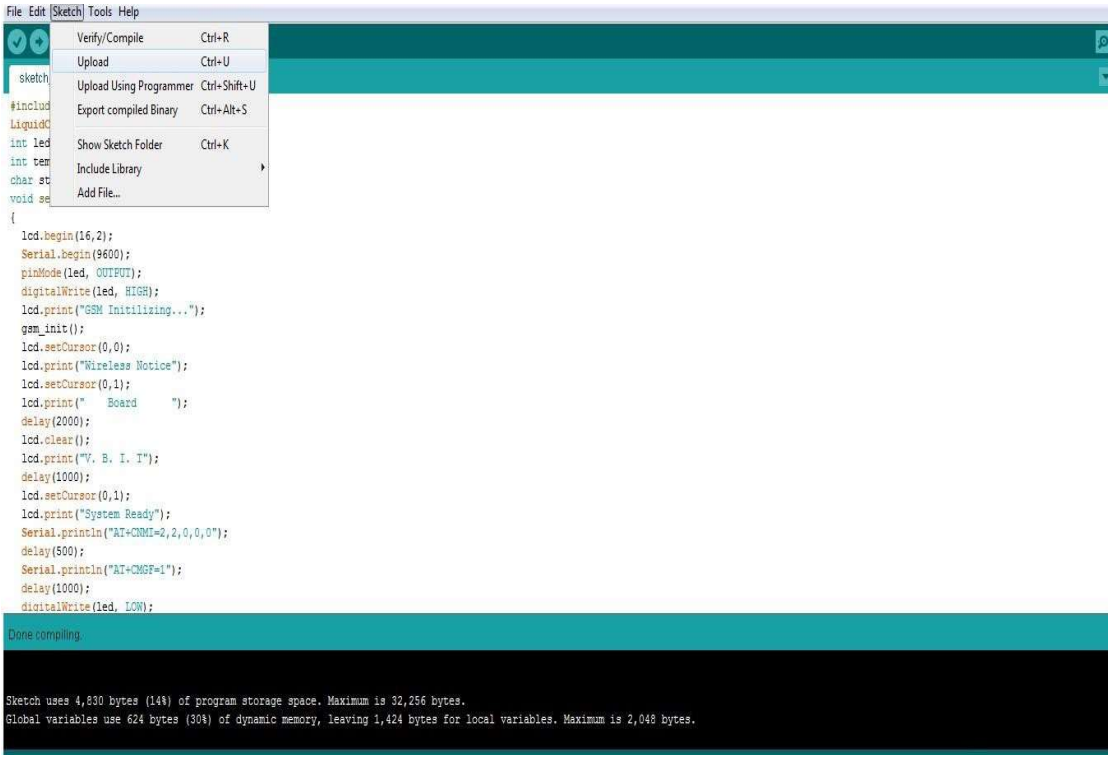


Figure 4.17: Uploading program to the arduino.

4.9.5 Language Reference

Arduino programs can be divided in three main parts: structure, values (variables and constants), and functions.

Available data types in ARDUINO IDE are

- void
- boolean
- char **(0 – 255)**
- byte - **8 bit data (0 – 255)**
- int - 16-bit data (32,767 - -32,768)

- long – **32 bit data (2,147,483,647 to -2,147,483,648)**
- float
- double
- string - char array
- String - object
- array

4.10 AT COMMANDS

At commands are used to control MODEMs. AT is the abbreviation for Attention. These commands come from Hayes commands that were used by the Hayes smart modems. The Hayes commands started with AT to indicate the attention from the MODEM. The dial up and wireless MODEMs (devices that involve machine to machine communication) need AT commands to interact with a computer. These include the Hayes command set as a subset, along with other extended AT commands**.**

AT commands with a mobile phone can be used to access following information and services:

1. Information and configuration pertaining to mobile device or Bluetooth module.
2. CALL ALERT services.
3. MMS services.
4. Fax services.
5. Data and Voice link over mobile network.

The Hayes subset commands are called the basic commands and the commands specific to a Bluetooth network are called extended AT commands.

4.11 TYPES OF AT COMMANDS:

There are four types of AT commands:

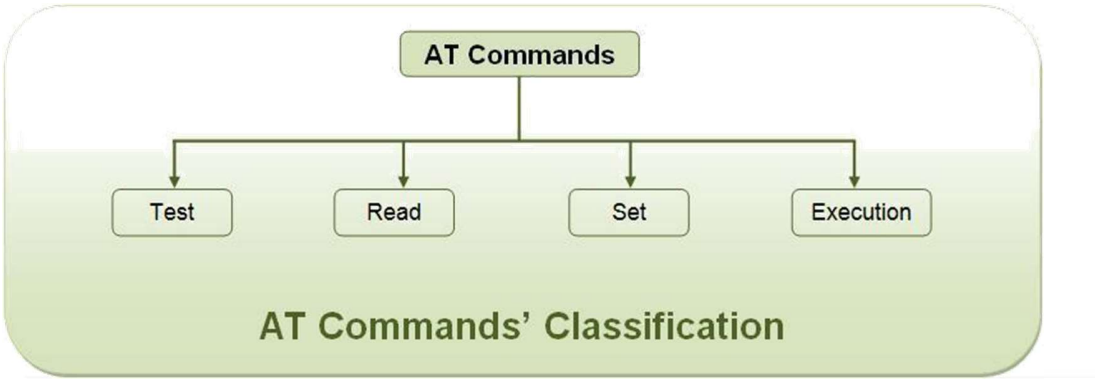


Figure 4.18: AT commands classification

4.11.1 Explanation of commonly used AT commands

1) AT - This command is used to check communication between the module and the computer.
For example, AT OK

The command returns a result code OK if the computer (serial port) and module are connected properly. If any of module or SIM is not working, it would return a result code ERROR.

2) +CMGF - This command are used to set the CALL ALERT mode. Either text or PDU mode can be selected by assigning 1 or 0 in the command.

SYNTAX: AT+CMGF=<mode>
0: for PDU mode
1: for text mode

The text mode of CALL ALERT is easier to operate but it allows limited features of CALL ALERT. The PDU (protocol data unit) allows more access to CALL ALERT services but the operator requires bit level knowledge of TPDUs. The headers and body of CALL ALERT are accessed in hex format in PDU mode so it allows availing more features.

For example,

AT+CMGF=1

OK

3) +CMGW - This command is used to store message in the SIM.

SYNTAX: AT+CMGW=" Phone number">Message to be stored Ctrl+z

As one types AT+CMGW and phone number, „>" sign appears on next line where one can type the message. Multiple line messages can be typed in this case. This is why the message is terminated by providing a „Ctrl+z" combination. As Ctrl+z is pressed, the following information response is displayed on the screen.+CMGW: Number on which message has been stored.

4) +CMGS - This command is used to send a CALL ALERT message to a phone number.

SYNTAX: AT+CMGS= serial number of message to be send.

As the command AT+CMGS and serial number of message are entered, CALL ALERT is sent to the particular SIM.

For example,

AT+CMGS=1

OK

5) ATD - This command is used to dial or call a number.

SYNTAX: ATD<Phone number> (Enter)

For example,

ATD123456789

6) ATA - This command is used to answer a call. An incoming call is indicated by a message „RING‟ which is repeated for every ring of the call. When the call ends „NO CARRIER‟ is displayed on the screen.

SYNTAX: ATA (Enter)

As ATA followed by enter key is pressed, incoming call is answered.

For example, RING

RING

ATA

7) ATH - This command is used to disconnect remote user link with the GSM module.

SYNTAX: ATH (Enter)

4.12 LIST OF AT COMMANDS

The AT commands for both, bluetooth module and the mobile phone, are listed below. Some of these commands may not be supported by all the bluetooth modules available. Also there might be some commands which won't be supported by some mobile handsets.

TABLE 3.5: AT Commands

Command	Description
AT	Checking communication between the module and computer.

CALL CONTROL

Command	Description
ATA	Answer command
ATD	Dial command
ATH	Hang up call
ATL	Monitor speaker mode
ATM	Monitor speaker mode
ATO	Go on-line
ATP	Set pulse dial as default
ATT	Set tone dial as default
AT+CSTA	Select type of address
AT+CRC	Cellular result codes

DATA CARD CONTROL

Command	Description
ATI	Identification
ATS	Select an S-register
ATZ	Recall stored profile
AT&F	Restore factory settings
AT&V	View active configuration
AT&W	Store parameters in given profile
AT&Y	Select set as power up option
AT +CLCK	Facility lock command

AT +COLP	Connected line identification presentation
AT +GCAP	Request complete capabilities list
AT+GMI	Request manufacturer identification
AT+GMM	Request model identification
AT+GMR	Request revision identification
AT+GSN	Request product serial number identification (IMEI)

PHONE CONTROL

Command	Description
AT+CBC	Battery Charge
AT+CGMI	Request manufacturer identification
AT+CGMM	Request model identification
AT+CGMR	Request revision identification
AT+CGSN	Request product serial number identification
AT+CMEE	Report mobile equipment error
AT+CPAS	Phone activity status
AT+CPBF	Find phone book entries
AT+CPBR	Read phone book entries
AT+CPBS	Select phone book memory storage
AT+CPBW	Write phone book entry

COMPUTER DATA INTERFACE

Command	Description
ATE	Command Echo
ATQ	Result code suppression
ATV	Define response format
ATX	Response range selection
AT&C	Define DCD usage
AT&D	Define DTR usage
AT&K	Select flow control
AT&Q	Define communications mode option
AT&S	Define DSR option
AT+ICF	DTE-DCE character framing
AT+IFC	DTE-DCE Local flow control
AT+IPR	Fixed DTE rate

SERVICE

Command	Description
AT+CLIP	Calling line identification presentation
AT+CR	Service reporting control
ATV+DR	Data compression reporting

NETWORK COMMUNICATION PARAMETER

Command	Description
ATB	Communications standard option
AT+CBST	Select bearer service type
AT+CEER	Extended error report
AT+CRLP	Radio link protocol
AT+DS	Data compression

MISCEELANEOUS

Command	Description
AT/	Re-execute command line
AT?	Command help
AT*C	Start CALL ALERT interpreter
AT*T	Enter
AT*V	Activate V.25bis mode
AT*NOKIA TEST	Test command
AT+CESP	Enter CALL ALERT block mode protocol

CALL ALERT TEXT MODE

Command	Description
AT+CCALL ALERT	Select message service
AT+CPMS	Preferred message storage
AT+CMGF	Message format
AT+CSCA	Service center address

AT+CSMP	Set text mode parameters
AT+CSDH	Show text mode parameters
AT+CSCB	Select cell broadcast message types
AT+CSAS	Save settings
AT+CRES	Restore settings
AT+CNMI	New message indications to TE
AT+CMGL	List messages
AT+CMGR	Read message
AT+CMGS	Send message
AT+CMSS	Send message from storage
AT+CMGW	Write message to memory
AT+CMGD	Delete message

CALL ALERT PDU MODE

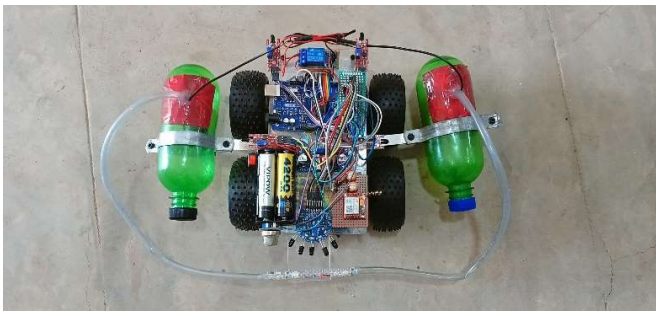
Command	Description
AT+CMGL	List messages
AT+CMGR	Read message
AT+CMGS	Send message
AT+CMGW	Write message to memory

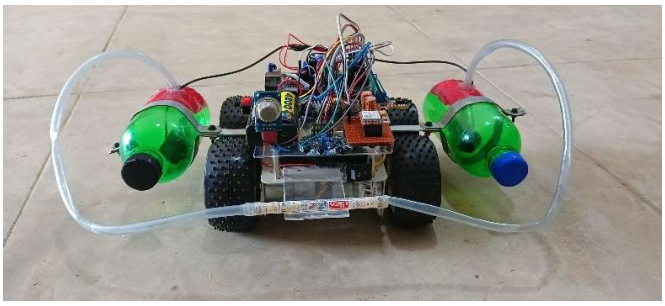
CHAPTER 5

RESULT

The Fire Extinguisher Robot Car with Call Alert is a highly efficient solution for fire emergencies. It combines fire detection, communication, and autonomous navigation to ensure quick response time and minimize damage. The robot is equipped with integrated sensors that detect fire and immediately send a call alert to designated contacts via a GSM or cellular module. It has motorized wheels and pathfinding algorithms that allow it to navigate towards the fire while detecting any obstacles for safe movement. Once it reaches the fire's location, the robot activates its water sprinkler system to extinguish the flames. This project showcases the effective integration of various technologies, resulting in a reliable and automated response to fire hazards. It enhances safety and reduces potential damage by providing a quick and efficient response to fire emergencies.

- Fire Detection: Utilizes sensors (e.g., flame sensors, temperature sensors, or smoke detectors) to accurately detect the presence of a fire. Ensures rapid response by immediately recognizing fire hazards.
- Call Alert System: Integrated communication module (e.g., GSM or cellular module) triggers a call alert to pre-set emergency contacts upon fire detection. Provides real-time notification, enhancing response times for human intervention if needed.





CHAPTER 6

6.1 Conclusion :

The prototype of the fire fighter robot was efficiently designed. This prototype has facilities to be integrated with many sensors making it move forward. The toolkit detects the infrared light emitted by the fire with photo diode and sends signal to controller. We intend to extend this work to provide a keypad programmed to allow manipulation of robot to move desired direction with help of motor driver module and extinguish the flames using water tank which is rotated at 180 degrees with help of servo in order for faster result. This future work will also explore to the use of a long distance sensor with suitable hardware to get more better and faster results addition to the characters.

62 Future scope :

The project has been motivated by the desire to design a system that can detect fires and take appropriate action, without any human intervention. The development of sensor networks and the maturity of robotics suggests that we can use mobile agents for tasks that involve perception of an external stimulus and reacting to the stimulus, even when the reaction involves a significant amount of mechanical actions. This provides us the opportunity to pass on to robots tasks that traditionally humans had to do but were inherently life- threatening. Fire-fighting is an obvious candidate for such automation. Given the number of lives lost regularly in fire- fighting, the system we envision is crying for adoption. Our experience suggests that designing a fire-fighting system with sensors and robots is within the reach of the current sensor network and mobile agent technologies. Furthermore, we believe that the techniques developed in this work will carry over to other areas involving sensing and reacting to stimulus, where we desire to replace the human with an automated mobile agent.

However, there has been research on many of these pieces in different contexts, e.g. coordination among mobile agents, techniques for detecting and avoiding obsta cles, on-thefly communication between humans and mobile agents, etc. It will be both interesting and challenging to put all this together into a practical, autonomous fire-fighting service.

REFERENCES

1. P. D. Minns, C Programming For the PC the MAC and the Arduino Microcontroller System. Author House, 2013
2. M. Banzi, Getting started with arduino. " O'Reilly Media, Inc.", 2009
3. A. M. Gibb, New media art, design, and the Arduino microcontroller: A malleable tool. PhD thesis, Pratt Institute, 2010
4. M. Margolis, Arduino cookbook. " O'Reilly Media, Inc.", 2011
5. D. Mellis, M. Banzi, D. Cuartielles, and T. Igoe, "Arduino: An open electronic prototyping platform, " in Proc. CHI, vol. 2007, 2007
6. C. K. Joo, Y. C. Kim, M. H. Choi, and Y. J. Ryoo, Self localization for intelligent mobile robot using multiple infrared range scanning system, In Control, Automation and Systems ICCAS'07, Seoul, Korea, 2007, 606-609.
7. http://electronicsforu.com/electronics-projects/hardware-diy/arduino-ir-firefighter robot
8. http://maker.robotistan.com/arduino-dersleri
9. J. Xu, W. Coombe, N. Boyson, A. Ohira, X. Gu, 143.472 Industrial Systems Design and Integration Fire Fighting Robot, 2006

APPENDIX

```
int fsen1=30;
int fsen2=28;
int fsen3=26;
int fsen4=24;
int fsen5=22;
int rsen1=32;
int rsen2=34;
int lsen1=36;
int lsen2=38;
int data[9]={0,0,0,0,0,0,0,0,0};
int IN1 = 2;
int IN2 = 3;
int IN3 = 4;
int IN4 = 5;
int IN5 = 8;
int IN6 = 9;
int IN7 = 10;
int IN8 = 11;
int mode=0;

void setup() {
  Serial.begin(9600);
  pinMode(fsen1,INPUT);
  pinMode(fsen2,INPUT);
  pinMode(fsen3,INPUT);
  pinMode(fsen4,INPUT);
  pinMode(fsen5,INPUT);
```

```
  pinMode(rsen1,INPUT);
  pinMode(rsen2,INPUT);
  pinMode(lsen1,INPUT);
  pinMode(lsen2,INPUT);
  // put your setup code here, to run once:

  pinMode(IN1, OUTPUT);
  pinMode(IN2, OUTPUT);
  pinMode(IN3, OUTPUT);
  pinMode(IN4, OUTPUT);
  pinMode(IN5, OUTPUT);
  pinMode(IN6, OUTPUT);
  pinMode(IN7, OUTPUT);
  pinMode(IN8, OUTPUT);
  digitalWrite(IN1, LOW);
  digitalWrite(IN2, LOW);
  digitalWrite(IN3, LOW);
  digitalWrite(IN4, LOW);
  digitalWrite(IN5, LOW);
  digitalWrite(IN6, LOW);
  digitalWrite(IN7, LOW);
  digitalWrite(IN8, LOW);
stop();

}

void loop() {

  data[0]=digitalRead(fsen1);
  data[1]=digitalRead(fsen2);
  data[2]=digitalRead(fsen3);
```

```
  data[3]=digitalRead(fsen4);
  data[4]=digitalRead(fsen5);
  data[5]=digitalRead(rsen1);
  data[6]=digitalRead(rsen2);
  data[7]=digitalRead(lsen1);
  data[8]=digitalRead(lsen2);

  for(int i=0;i<9;i++)
  {
    Serial.print(data[i]);
    Serial.print(",");

  }
  Serial.println(".");
  delay(50);

if(data[2]==1)
{
  forward();
}
else if (data[0]==1||data[1]==1||data[5]==1||data[6]==1)
{
  right();
  delay(300);
}

else if (data[3]==1||data[4]==1||data[7]==1||data[8]==1)
{
  left();
  delay(300);
}
```

```
else
{
  stop();
}

 if ( data[1]==1 && data[2]==1 && data[3]==1)
{
  stop();

  Serial.print ("pump on");
}



}

void left()
  {
  digitalWrite(IN1, HIGH);
  digitalWrite(IN2, LOW);

  digitalWrite(IN3, HIGH);
  digitalWrite(IN4, LOW);

  digitalWrite(IN5, HIGH);
  digitalWrite(IN6, LOW);

  digitalWrite(IN7, HIGH);
  digitalWrite(IN8, LOW);

}
```

```
  void right()
  {
  digitalWrite(IN1, LOW);
  digitalWrite(IN2, HIGH);

  digitalWrite(IN3, LOW);
  digitalWrite(IN4, HIGH);

  digitalWrite(IN5, LOW);
  digitalWrite(IN6, HIGH);

  digitalWrite(IN7, LOW);
  digitalWrite(IN8, HIGH);

  }
void forward()
{
  digitalWrite(IN1, LOW);
  digitalWrite(IN2, HIGH);

  digitalWrite(IN3, LOW);
  digitalWrite(IN4, HIGH);

  digitalWrite(IN5, HIGH);
  digitalWrite(IN6, LOW);

  digitalWrite(IN7, HIGH);
  digitalWrite(IN8, LOW);

}
void backward()
```

```
{
  digitalWrite(IN1, HIGH);
  digitalWrite(IN2, LOW);

  digitalWrite(IN3, HIGH);
  digitalWrite(IN4, LOW);

  digitalWrite(IN5, LOW);
  digitalWrite(IN6, HIGH);

  digitalWrite(IN7, LOW);
  digitalWrite(IN8, HIGH);

}
void stop()
{
  digitalWrite(IN1, LOW);
  digitalWrite(IN2, LOW);

  digitalWrite(IN3, LOW);
  digitalWrite(IN4, LOW);

  digitalWrite(IN5, LOW);
  digitalWrite(IN6, LOW);

  digitalWrite(IN7, LOW);
  digitalWrite(IN8, LOW);

}
```