A Major Project Report

On

IOT-Controlled Landmine Detection Robot with Live Video

Submitted

In partial fulfillment for the requirement for the award of the degree of

BACHELOR OF TECHNOLOGY

in

COMPUTER SCIENCE AND ENGINEERING

By

Mamidi Yashwanth Reddy **21641A05A3**

Under the Guidance of

Dr. K. Rekha Devi

Assistant Professor Dept, of CSE

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

VAAGDEVI COLLEGE OF ENGINEERING

(UGC Autonomous, Accredited by NAAC with "A")

Bollikunta, Khila Warangal (Mandal), Warangal Urban – 506005(T.S)

(2021-2025)

VAAGDEVI COLLEGE OF ENGINEERING

(Autonomous)

(Accredited by NBA, Accredited by NAAC with "A")

Bollikunta, Khila Warangal (Mandal), Warangal Urban – 506005(T.S)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

(Accredited by NBA)

CERTIFICATE

This is to certify that the project entitled "**IOT-Controlled Landmine Detection Robot with Live Video**" is submitted by **Mamidi Yashwanth Reddy** bearing Hall Ticket Nos. **21641A05A3** in partial fulfillment of the requirements for the award of the Degree in Bachelor of Technology in Computer Science and Engineering during the academic year 2024-2025.

GUIDE

Dr. K. Rekha Devi

Assistant Professor Dept, of CSE

HEAD OF THE DEPARTMENT

Dr. N. Satyavathi

Associate Professor Dept, of CSE

EXTERNAL EXAMINER

ACKNOWLEDGEMENT

I would like to take this opportunity to express our sincere gratitude and deep respect to **Dr. K. Prakash**, **Principal** of **Vaagdevi College of Engineering**, for providing us with all the necessary assistance and for his support and inspiration, which enabled us to carry out this project work at the institute.

Our heartfelt thanks extend to **Dr. N. Satyavathi, Associate Professor, Head of the Department,** Computer Science and Engineering,Vaagdevi College of Engineering, for equipping us with the required infrastructure and granting us the freedom to successfully conduct this project work.

I are immensely grateful to our guide**, Dr.K.Rekha Devi, Assistant Professor**, Department of Computer Science and Engineering, for her constant support and invaluable guidance throughout the duration of this project work. Their insights and encouragement have been instrumental in our progress.

I would also like to thank the Project Coordinators, **Dr.K.Rekha Devi, Assistant Professor** and **Mr. Ch. Aravind Kumar, Assistant Professor**, for their thoughtful suggestions, motivation, and encouragement, which contributed significantly to the completion of our project.

Finally, I extend our sincere thanks to all the teaching and non-teaching staff members of the Department of Computer Science and Engineering, whose direct and indirect support was vital in ensuring the successful completion of this work.

Mamidi Yashwanth Reddy **21641A05A3**

DECLARATION

I hereby certify that the work which is being presented in the project report entitled **IOT-Controlled Landmine Detection Robot with Live Video** in thepartial fulfillment of the requirements for the award of the degree of **BATCHLOR OF TECHNOLOGY** and submitted in the **DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING, VAAGDEVI COLLEGE OF ENGINEERING**, is an authentic record of my own work carried out during the time period from 2021 to 2025 under the Guidance of **Dr.K.Rekha Devi, Assistant Professor**. The matter presented in this project report has not been submitted by me for the award of any other degree of this or any other institute.

Signature of the students with date

Mamidi Yashwanth Reddy 21641A05A3

ABSTRACT

Landmines pose a significant threat in post-conflict areas, endangering civilian lives and hindering development. This project presents a cost-effective, remote-controlled landmine detection robot designed using Arduino technology. The robot is equipped with a metal detector sensor capable of identifying metallic landmines buried beneath the ground. Controlled via wireless communication (e.g., Bluetooth or RF module), the robot allows the operator to navigate hazardous zones from a safe distance. The system integrates an Arduino microcontroller to process sensor data and control the robot's movement. Upon detecting a potential landmine, the robot provides real-time alerts through visual or audio indicators. This solution aims to enhance safety in mine-affected regions while minimizing risk to human deminers. The simplicity and affordability of the design make it suitable for deployment in low-resource settings. Landmines continue to be a persistent threat in many post-war regions around the world, causing injury and death to civilians long after conflicts have ended. The manual detection and demining process is highly dangerous, time-consuming, and resource-intensive. To address this critical issue, this project proposes the design and implementation of a remote-controlled landmine detection robot using Arduino as the central control unit. The system aims to enhance demining safety and efficiency through automation and remote operation.The robot is equipped with a metal detector sensor that can identify buried metallic objects, typically associated with landmines. The detection mechanism is mounted on a movable robotic platform, which is driven by DC motors controlled via an Arduino Uno microcontroller. The robot can be navigated wirelessly using Bluetooth, RF, or Wi-Fi modules, allowing the operator to control the robot from a safe distance and avoid direct contact with potentially dangerous areas.Upon detecting metal beneath the surface, the sensor sends signals to the Arduino, which then triggers visual (LED indicators) and/or auditory (buzzer) alarms to notify the user of a potential landmine. The robot's movements—forward, backward, left, and right—are controlled remotely through a mobile application or joystick interface, offering precise maneuverability in challenging terrains.

INDEX PAGE

TABLE INDEX

LIST OF FIGURES

CHAPTER-1

INTRODUCTION

1.1 Introduction

Landmines remain one of the most dangerous remnants of war, posing serious risks to civilians and military personnel in post-conflict zones. Traditional landmine detection methods are not only time-consuming but also highly hazardous for human operators. With advancements in robotics and embedded systems, there is an increasing focus on developing intelligent, automated solutions to reduce human involvement in dangerous operations.This project presents a remote-controlled landmine detection robot integrated with a robotic arm, designed using Arduino as the central microcontroller. The robot is equipped with a metal detector sensor capable of identifying buried metallic landmines and a robotic arm that can be used for marking, manipulating, or cautiously handling suspicious objects. The system is wirelessly controlled using Bluetooth, RF, or Wi-Fi modules, allowing the operator to remain at a safe distance while navigating and managing operations in hazardous areas.The robotic platform offers high mobility, obstacle navigation, and real-time detection, making it suitable for mine-affected terrains. The integration of the robotic arm adds significant functionality by enabling remote interaction with detected objects, which enhances both safety and versatility. Powered by a rechargeable battery and built on a durable chassis, the robot provides a reliable and low-cost solution for mine detection tasks.By combining sensor technology, wireless control, and robotic manipulation, this project aims to contribute a safer, efficient, and practical alternative to traditional mine detection methods.

Landmines, remnants of past wars and conflicts, continue to pose a serious humanitarian and security challenge in many parts of the world. Every year, thousands of people are injured or killed by accidental landmine detonations, many of whom are civilians, including children. The conventional methods used for landmine detection—such as manual probing, sniffer dogs, and trained personnel—are not only slow and costly but also extremely dangerous. As a result, there is an urgent need for the development of safer, more efficient, and cost-effective technological solutions to address this global issue.

In response to this challenge, this project focuses on the development of a remote-controlled landmine detection robot integrated with a robotic arm, built using the Arduino microcontroller platform. Arduino is chosen for its open-source flexibility, ease of programming, and extensive support for interfacing with various sensors and actuators. The core objective of this robot is to detect buried metallic landmines using a metal detector sensor and to allow remote handling or marking of suspected objects using a robotic arm—all while keeping the operator at a safe distance from the hazardous area.

The robotic platform consists of a durable, all-terrain chassis equipped with DC geared motors for mobility, which are controlled via a motor driver module interfaced with the Arduino. Wireless communication—achieved through Bluetooth, RF, or Wi-Fi modules—enables real-time remote control of the robot's movements and robotic arm actions. The onboard metal detector continuously scans the ground as the robot moves, and when it detects a metallic object (potentially a landmine), it sends a signal to the Arduino. This, in turn, triggers an alert system using LEDs or buzzers to notify the operator of a possible threat.

The unique addition of a robotic arm enhances the robot's functionality beyond basic detection. The arm can be used to mark the location of a suspected mine, move light obstacles, or interact with unknown objects from a safe distance. This helps avoid the need for human intervention in close proximity to danger zones and provides added flexibility in field operations. The arm can be controlled through the same wireless interface, with servo motors ensuring precise movements.This project demonstrates a multidisciplinary approach involving robotics, electronics, wireless communication, and embedded programming. The system is designed to be low-cost, scalable, and easily replicable, making it suitable for deployment in regions with limited resources. Moreover, the modularity of the design allows future upgrades, such as the integration of GPS modules for location tracking, cameras for live video feed, or machine learning algorithms for object classification.

In conclusion, this remote-controlled landmine detection robot with a robotic arm represents a significant step forward in developing safer and more effective tools for landmine detection. It not only minimizes human exposure to life-threatening risks but also paves the way for smarter, technology-driven demining solutions.

CHAPTER-2

LITERATURE REVIEW

Acute Manandhar, Peter A. Torrione, Leslie M. Collins and Kenneth D. Morton, "Multiple-Instance Hidden Markov Model for GPR-Based Landmine Detection," IEEE transactions on Geosciences and remote sensing, vol. 53, no. 4, pp. 1737-1745, April 2015. 2)Jaradat, M.A, "Autonomous navigation robot for landmine detection applications," IEEE transactions on Mechatronics and its Applications, vol.16, no. 3, pp. 1-5, April 2012.3)"Design and Implementation of Landmine Robot" Wade Ghribi, Ahmed Said Badawy, Mohammed Rahmathullah, Suresh Babu Changalasetty. IJEIT Volume 2, Issue 11, May 2013. 4)Ghribi, W., Badawy, A.S., Rahmathullah, M. and Changalasetty, S.B." Design and implementation of landmine robot". Int. J. Engg. Innov. Technol. 2(11): 250-256, March 2013. 5)P. Gonzalez de Santos, E. Garcia, J. Estremera and M.A. "Using walking robots for landmine detection and location". Armada Industrial Automation Institute-CSIC . Camp Real, Km. 0,200- La Poveda 28500 Arganda del Rey, Madrid, Spain, January 2014. 6)Ilaria Bottigliero. 120 Million Landmines Deployed Worldwide: Fact or Fiction. Pen and Sword Books Ltd, Barnsley, South Yorkshire, UK, June 2014. 7)MacDonald J., Lockwood J.R., Mc Fee J.E., Altshuler T., Broach J. T., L. Carin,Harmon R.S., Rappaport C., Scott W.R., and Weaver R. Alternatives for landmine detection. Technical report, RAND (http://www.rand.org/ publications/ MR/ MR 1608/ MR 1608. appg.pdf), February2013. 8)Jaradat M A, Bani Salim M N and Awad F H, "Autonomous Navigation Robot for Landmine Detection Applications", 8th International Symposium on Mechatronics and its Applications (ISMA), April 2012. L. Robledo, M. Carrasco and D. Mery," A survey of land mine detection technology" International Journal of Remote Sensing Vol. 30, No. 9, 10 May 2009, 2399–2410 [2] Jebasingh Kirubakaran.S.J, Anish kumar jha, Dheeraj kumar, Sadambi Poorna chandram Prakash, "Mine Detecting Robot with Multi Sensors Controlled Using HC-12 Module" International Journal of Engineering & Technology [3] Bharath J, "Automatic Land Mine Detection Robot Using Microcontroller", International Journal of Advance Engineering and Research Development Volume 4, Issue 3, March-2017 [4] Zhenjun He, Jiang Zhang, Peng Xu, Jiaheng Qin and Yunkai Zhu, "Mine Detecting Robot Based on Wireless Communication with Multi-sensor". [5] Jaradat M A, Bani Salim M N and Awad F H (2012), "Autonomous Navigation Robot for Landmine Detection Applications". [6] Kuo-Lan Su, Hsu-Shan Su,

Sheng-Wen Shiao and JrHung Guo (2011), “Motion Planning for a Landmine Detection Robot”, Artificial Life and Robotics. [7] Kaur Gurpreet, “Multi algorithm-based Landmine Detection using Ground Penetration Radar”, IEEE, 2016. [8] Kishan Malaviya, et.al, “Autonomous Landmine Detecting and Mapping Robot”, IJIRCCE, Vol. 3, Issue 2, 2015.

Related works

1. Arduino-Based Landmine Detection Robot (Singh et al., 2019)

This project involved a simple robot built using an Arduino UNO, metal detector coil, and RF transmitter-receiver module. The robot was capable of detecting metallic objects and signaling the operator wirelessly. Although low-cost and easy to implement, the system was limited by its range and the inability to distinguish between real landmines and metal debris.

2. GPR and Metal Detector Hybrid Robot (Li et al., 2021)

Li and colleagues developed a hybrid system using both Ground Penetrating Radar (GPR) and metal detection to reduce false positives. The robot used a rugged chassis for off-road mobility and featured real-time data transmission. The system achieved better detection rates than metal-only systems but was costly and had power management issues.

3. Semi-Autonomous Mine Detection Robot Using Raspberry Pi (Rahman et al., 2016)

This prototype integrated a Raspberry Pi, wireless camera, and ultrasonic sensors for obstacle detection and navigation. The metal detection module was controlled using the Pi GPIO pins, and the video feed was transmitted to a remote laptop. The semi-autonomous capability allowed limited decision-making, enhancing efficiency. However, environmental interference affected metal detection accuracy.

4. Landmine Detection Using IOT and Android App (Sharma et al., 2015)

In this project, a metal detector robot was connected via Bluetooth to an Android app. The app could control the robot and display alert messages when a mine was detected. The approach demonstrated ease of use and mobility but was constrained by Bluetooth’s short-range communication.

5. Military Surveillance Robot with Mine Detection (Patil & Jadhav, 2020)

This system combined surveillance features such as a live camera feed with landmine detection using a metal detector. Controlled using RF modules, it was designed for remote operation in military zones. The project highlighted the integration of multiple functionalities but suffered from limited terrain adaptability.

6. AI-Powered Landmine Classification (Mendes et al., 2020)

Mendes et al. implemented a neural network to analyze sensor signals and reduce false positives in landmine detection. Their system was trained on sample datasets of metal objects buried at different depths. Though highly promising, the system required significant computational power and sensor calibration, making it less feasible for basic field robots.

Summary of Gaps and Opportunities

While various approaches exist for building landmine detection robots, many of them face similar challenges:

- High false positive rates due to undifferentiated metal detection.
- Poor performance in rough terrain.
- Limited communication range and real-time feedback.
- Lack of integration between sensors, navigation, and data processing.

Your project aims to address these gaps by developing a remote-controlled robot that integrates metal detection, wireless video streaming, GPS tracking, and terrain navigation—offering a more balanced and practical solution for real-world demining scenarios.

CHAPTER-3

EXISTING SYSTEM

The existing systems for remote-controlled landmine detection robots primarily consist of basic robotic platforms integrated with metal detectors and controlled via RF or Bluetooth modules. These systems are typically used in academic or experimental settings and are designed to detect metallic objects buried underground. While they offer a safer alternative to manual demining, they often suffer from high false-positive rates, limited communication range, poor terrain adaptability, and lack advanced features like GPS tracking or real-time video feedback. Additionally, many of these systems cannot distinguish between actual landmines and harmless metal debris, limiting their effectiveness in real-world demining scenarios.

CHAPTER 4

PROPOSED SYSTEM

The robot consists of a rugged four-wheel drive or tracked chassis capable of traversing uneven and dangerous terrains. A metal detector coil is mounted at the front and close to the ground to scan for buried metallic objects. A robotic arm, controlled via servos, is attached to the body and can be used to place markers, lift small debris, or assist in examining suspected landmine locations. The robot is operated remotely using RF or Wi-Fi communication via a custom controller or smartphone interface.

An Arduino Uno or ESP32 microcontroller serves as the brain of the system, handling motor control, sensor data processing, and arm movement. Ultrasonic sensors are used for obstacle detection, while a wireless camera provides a live video feed to assist the operator in navigating and operating the robot and arm. A buzzer and LED system provides immediate alerts when a metallic object is detected.

Key Components

- Arduino Uno / ESP32 – Controls the robot, sensors, and robotic arm.
- Metal Detector – Detects underground metallic landmines.
- Robotic Arm – 4-DOF or 6-DOF arm for manipulation and marking.
- Wireless Camera – Provides real-time video feedback to the user.
- RF/Wi-Fi Module – Enables remote control of the robot and arm.
- Ultrasonic Sensors – Assist in obstacle detection and path planning.
- Motor Driver (L298N) – Controls DC motors and servo motors.
- Power Supply – Rechargeable battery for mobility and operation.

System Functionality

- The user remotely navigates the robot through a wireless interface.
- The metal detector continuously scans for buried metallic objects.

- On detection, the system triggers alerts (buzzer/LED) and transmits location/video to the operator.
- The robotic arm is used to place a marker or move debris near the detection point.
- Optional GPS can be added to log detection locations for mapping.

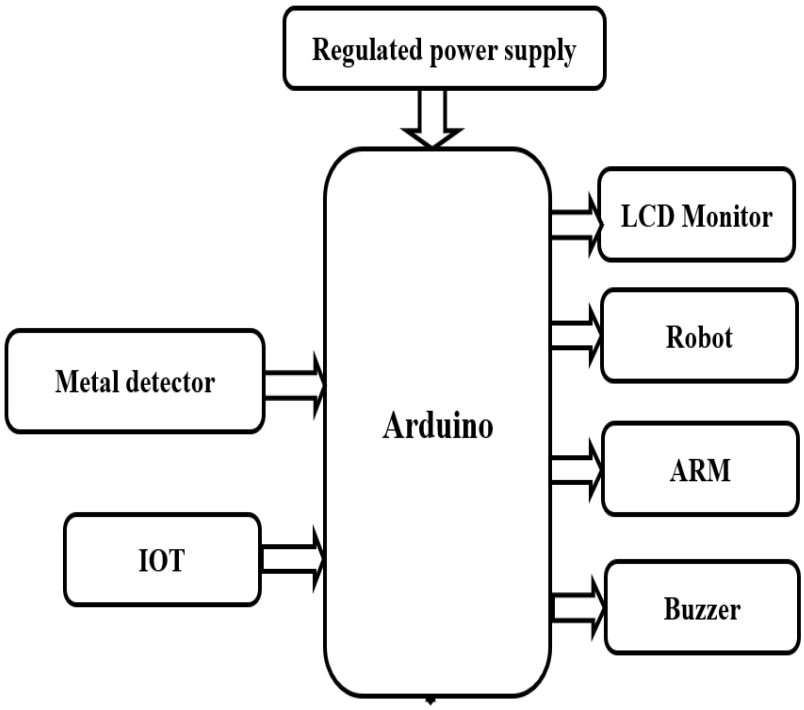
Advantages

- Enables safe mine detection without risking human life.
- Robotic arm increases flexibility for marking and manipulation tasks.
- Real-time video and sensor feedback enhance operator control.
- Suitable for deployment in diverse terrains and dangerous zones.

In addition to enhancing safety during demining operations, the inclusion of a robotic arm significantly expands the robot's functionality beyond simple detection. The arm can be used to place warning flags or markers at the location of a detected mine, helping demining teams avoid dangerous areas and plan extraction routes more effectively. In some cases, it may also be used to gently remove surface debris such as leaves, rocks, or loose soil that may be covering the mine, allowing for better visibility without direct human contact. The arm's movement is precisely controlled via servos and can be manipulated remotely through a dedicated controller or interface, ensuring accurate and delicate operation even from a safe distance.

To further improve the robot's effectiveness, it can be equipped with a GPS module to record and map mine locations. This geolocation data can be transmitted in real time or stored for later analysis, creating a digital record of suspected minefields. The robot's design also allows for future upgrades, such as integrating AI for object recognition, adding temperature or humidity sensors to assess terrain conditions, or using GPRS modules for long-range communication. Overall, the proposed system offers a modular, scalable, and user-friendly solution to one of the world's most dangerous humanitarian challenges—landmine detection and clearance.

BLOCK DIAGRAM



DESCRIPTION:

1. Overview of the System

The block diagram represents a **remote-controlled landmine detection robotic system** designed for safe and efficient mine detection operations. At the heart of the system lies the **Arduino microcontroller**, which acts as the central unit responsible for processing data, managing sensor inputs, controlling outputs, and maintaining overall coordination of the robot's functionality. The system integrates key components including a **metal detector**, **robotic arm**, **buzzer**, **IoT module**, **LCD monitor**, and **robotic drive unit**, all powered by a **regulated power supply**.

The purpose of this robot is to detect buried metallic landmines, alert the operator, and optionally mark or manipulate the area using a robotic arm. Through remote communication,

possibly via IoT or Bluetooth, the robot can be operated from a safe distance, reducing human risk in hazardous areas.

2. Regulated Power Supply

The entire system is powered by a **regulated power supply**, which ensures that all components receive a stable and appropriate voltage level. The Arduino board, sensors, motors, and display modules typically operate on 5V or 12V DC. This regulated power source protects the electronic components from voltage fluctuations and overloads that could otherwise damage the circuitry or lead to malfunction.

Power is distributed to all peripherals from this supply unit, making it a critical part of the robot's infrastructure. The power supply can be implemented using a battery pack (e.g., 12V rechargeable battery) with voltage regulators like the 7805 IC to provide 5V output for Arduino and sensors.

3. Arduino (Microcontroller Unit)

The **Arduino** acts as the **brain of the system**, controlling every input and output. It receives data from the **metal detector** and **IoT module**, processes this data, and responds by sending signals to other components such as the **buzzer**, **LCD monitor**, **robot motors**, and **robotic arm**.

It reads analog or digital signals from the metal detector to determine if a metallic object (possibly a landmine) is present. It then commands the buzzer to alert, displays information on the LCD, and can be programmed to stop or slow down the robot. Additionally, it controls the robotic arm's movement based on manual commands or pre-programmed instructions.

4. Metal Detector

The **metal detector** is a crucial part of the robot, responsible for identifying the presence of metallic materials buried underground. It generally works based on the principle of electromagnetic induction: when the coil in the detector passes over a metallic object, it experiences a change in inductance, producing a signal.

This signal is fed to the Arduino, which interprets the reading and triggers an alert if a potential landmine is detected. The detector is typically mounted close to the ground on the front of the robot, scanning the soil as the robot moves.

5. IoT Module

The **IoT (Internet of Things) module** allows the robot to communicate with a **remote monitoring system** via the internet. Using Wi-Fi-enabled modules like the **ESP8266 or ESP32**, the robot can transmit data such as detection events, robot status, location (if GPS is included), and even stream images or video if a camera is attached.

This enables **real-time remote monitoring and control**, enhancing operational safety and effectiveness. Operators can control the robot and receive detection alerts through a web or mobile app interface.

6. LCD Monitor

The **LCD monitor** is used for displaying critical real-time data such as detection status, sensor readings, system status, and control feedback. For example, when a landmine is detected, the display can show messages like "Metal Detected," signal strength, or even GPS coordinates if applicable.

This helps field personnel quickly interpret the situation without needing external monitoring systems. A **16x2 or 20x4 LCD** is typically used due to its ease of integration with Arduino and good visibility in daylight.

7. Robot (Drive Unit)

This block represents the **movement mechanism** of the robot, which includes the **DC motors, motor drivers (like L298N or L293D), wheels or tracks**, and chassis. The robot's movement (forward, backward, left, right, stop) is controlled through the Arduino based on user input from the remote controller or IoT interface.The mobility system is designed to be robust enough to traverse rough or uneven terrain where landmines may be buried. Motor drivers are essential to interface the Arduino with the high-current motors used in the drive system.

8. Robotic Arm

The **robotic arm** is a major enhancement in this design. It allows the robot to **interact physically** with the environment — for example, placing a marker at the location of a detected mine or clearing surface debris like leaves or soil.

The arm is typically driven by **servo motors** and can be 4-DOF (Degrees of Freedom) or more, depending on the application. The Arduino sends pulse-width modulation (PWM) signals to each servo to control joint angles and movement. The arm is controlled remotely and requires precise command handling.

9. Buzzer

The **buzzer** acts as an audio alert system that is triggered when the metal detector senses a potential landmine. It provides **instant audible feedback** to nearby operators or rescue workers. In some designs, a visual LED indicator is also included to complement the buzzer.

This simple yet effective alarm system ensures that even if the operator misses an LCD message or camera feed, they are still alerted to a possible danger through sound.

CHAPTER-5

UML DIAGRAMS

1. ACTIVITY DIAGRAM

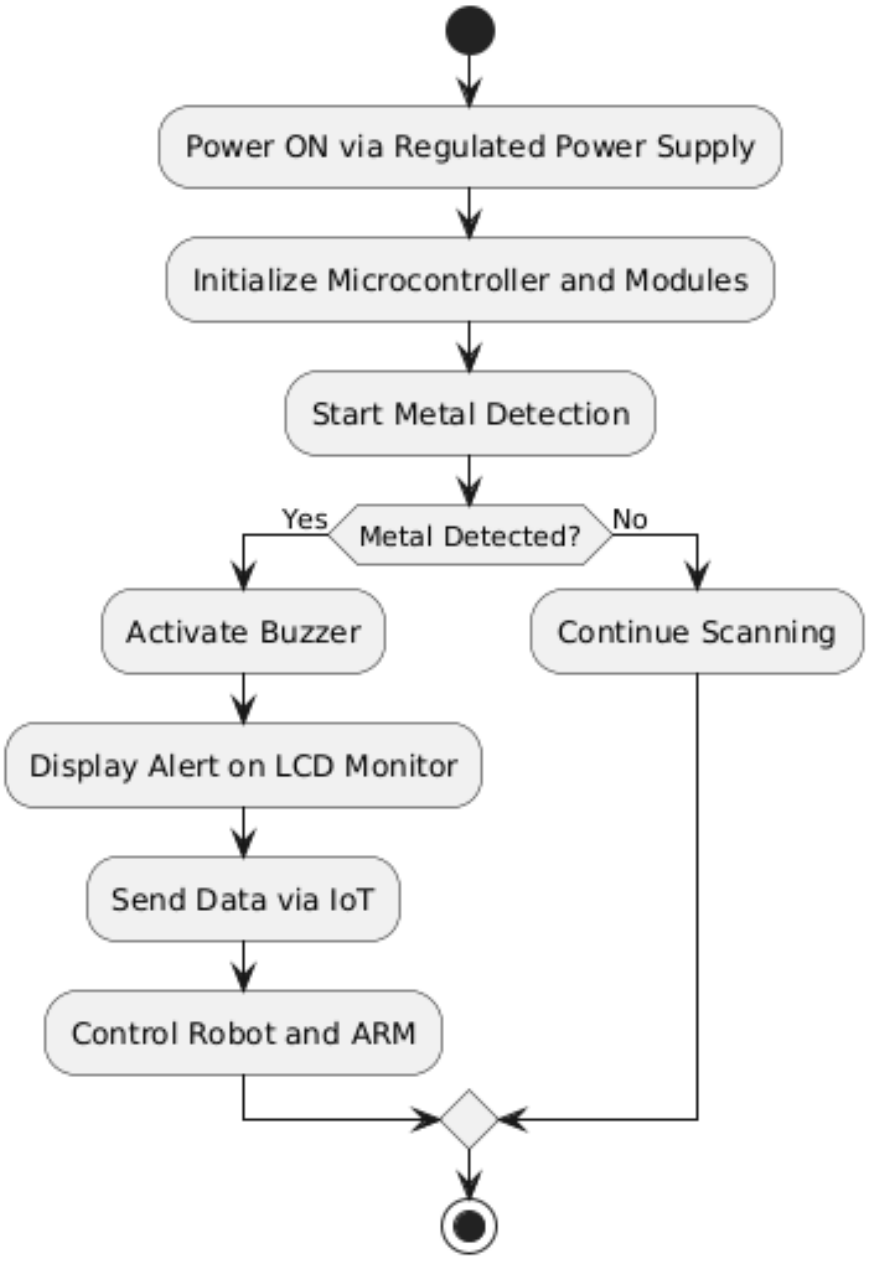


Fig 5.1 :Activity Diagram

2. CLASS DIAGRAM

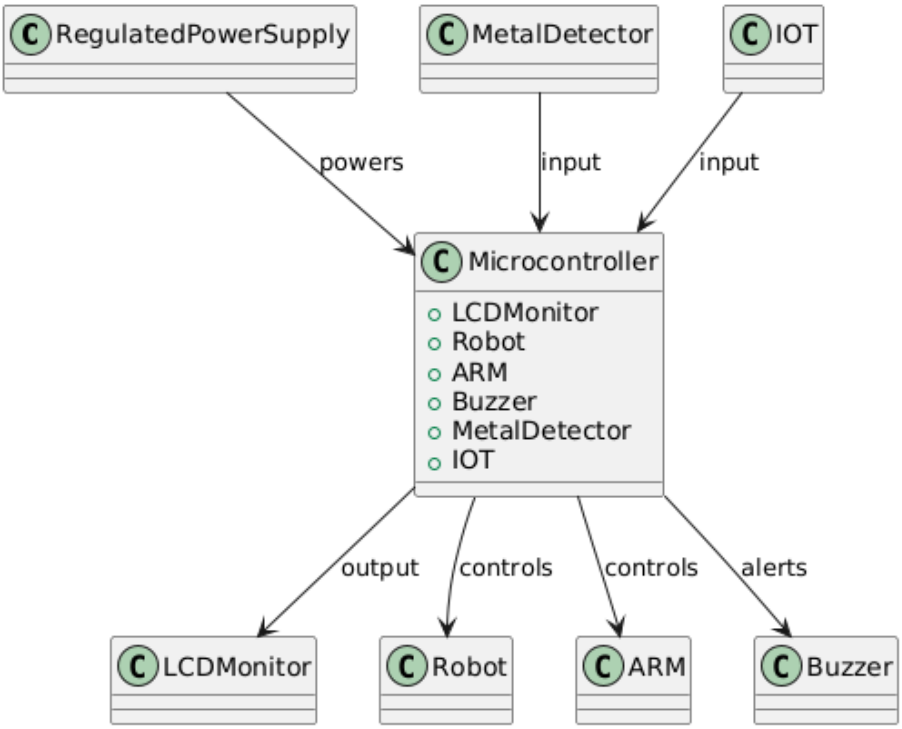


Fig 5.2 : Class Diagram

3. Component Diagram

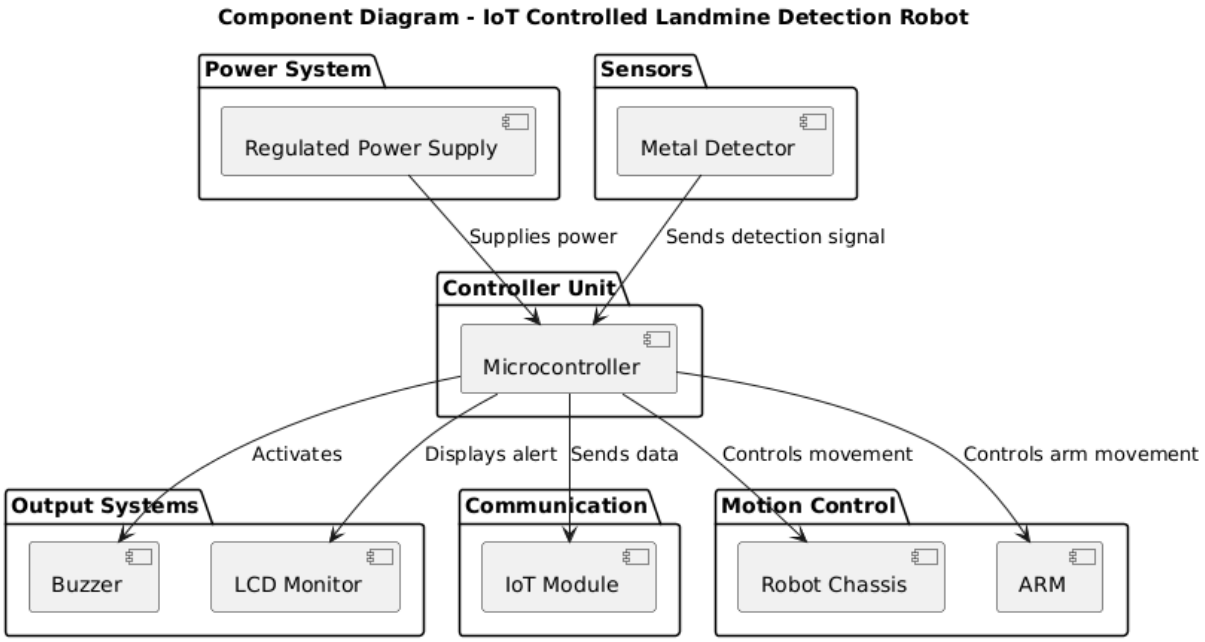


Fig 5.3 : Component Diagram

4. Entity-Relationship Diagram

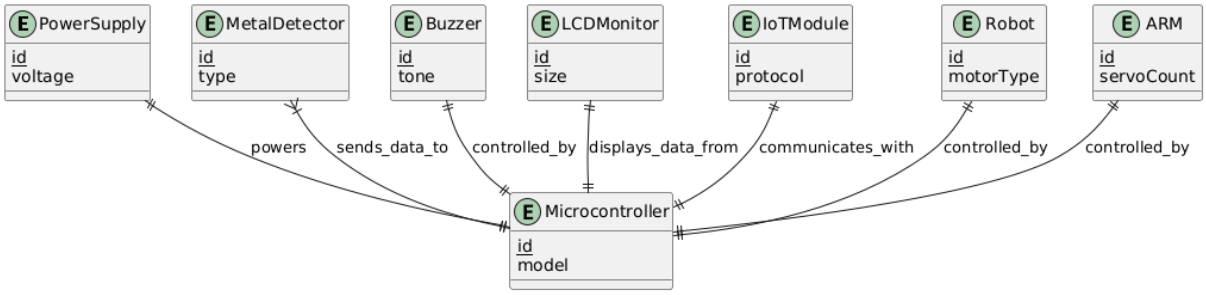


Fig 5.4 : Entity-Relationship Diagram

5. Use Case Diagram

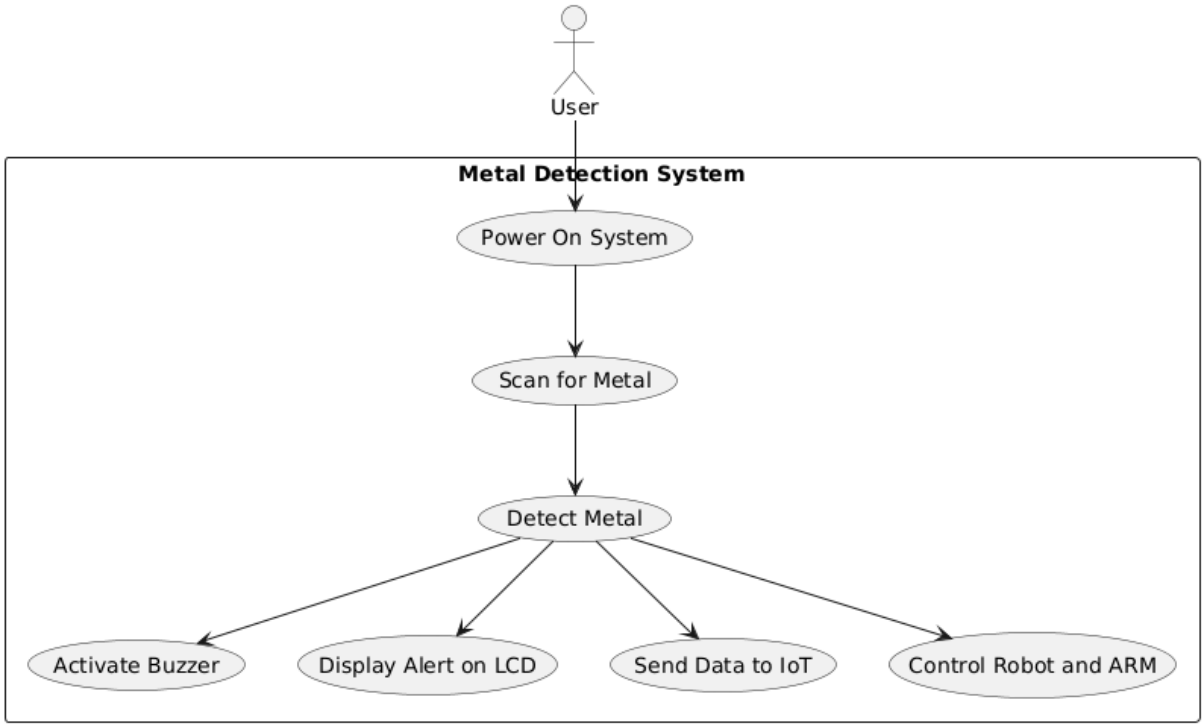


Fig 5.5 : Use Case Diagram

6. Sequence Diagram

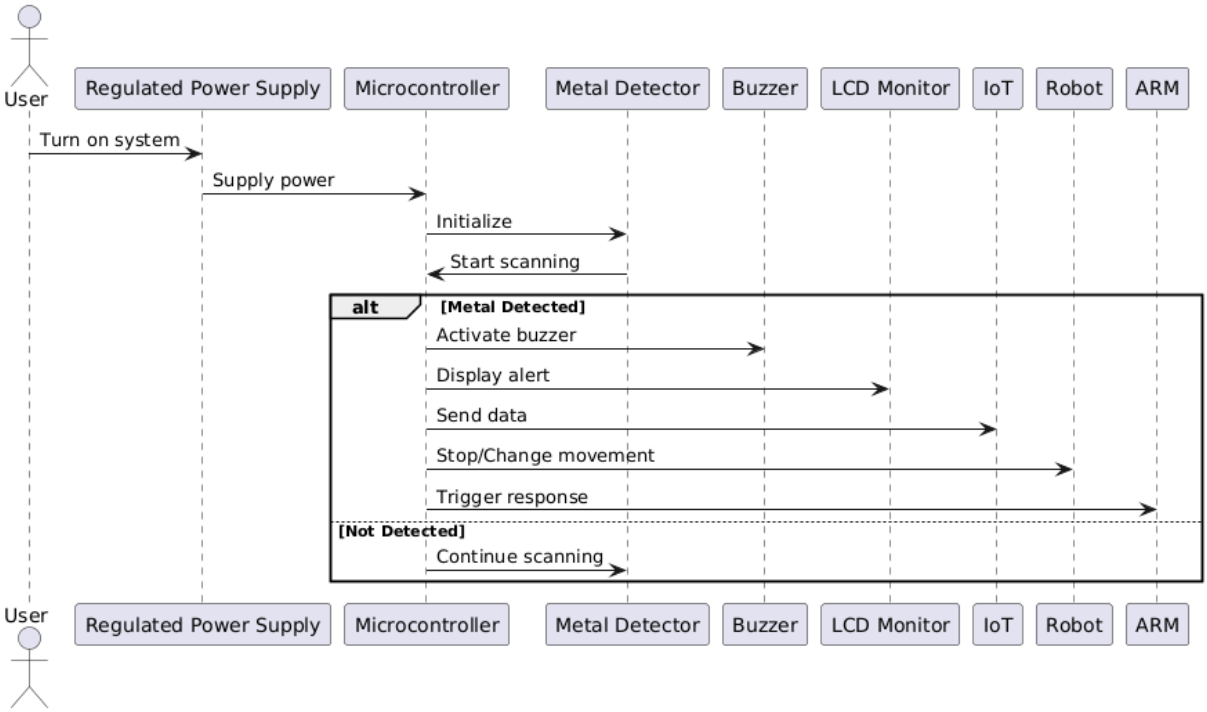


Fig 5.6 : Sequence Diagram

CHAPTER-6

EMBEDDED SYSTEMS

6.1 Embedded Systems:

An embedded system is a computer system designed to perform one or a few dedicated functions often with real-time computing constraints. It is embedded as part of a complete device often including hardware and mechanical parts. By contrast, a general-purpose computer, such as a personal computer (PC), is designed to be flexible and to meet a wide range of end-user needs. Embedded systems control many devices in common use today.

Embedded systems are controlled by one or more main processing cores that are typically either microcontrollers or digital signal processors (DSP). The key characteristic, however, is being dedicated to handle a particular task, which may require very powerful processors. For example, air traffic control systems may usefully be viewed as embedded, even though they involve mainframe computers and dedicated regional and national networks between airports and radar sites. (Each radar probably includes one or more embedded systems of its own.)

Since the embedded system is dedicated to specific tasks, design engineers can optimize it to reduce the size and cost of the product and increase the reliability and performance. Some embedded systems are mass-produced, benefiting from economies of scale.

Physically embedded systems range from portable devices such as digital watches and MP3 players, to large stationary installations like traffic lights, factory controllers, or the systems controlling nuclear power plants. Complexity varies from low, with a single microcontroller chip, to very high with multiple units, peripherals and networks mounted inside a large chassis or enclosure.

In general, "embedded system" is not a strictly definable term, as most systems have some element of extensibility or programmability. For example, handheld computers share some elements with embedded systems such as the operating systems and microprocessors which power them, but they allow different applications to be loaded and peripherals to be connected. Moreover, even systems which don't expose programmability as a primary feature

generally need to support software updates. On a continuum from "general purpose" to "embedded", large application systems will have subcomponents at most points even if the system as a whole is "designed to perform one or a few dedicated functions", and is thus appropriate to call "embedded". A modern example of embedded system is shown in fig: 2.1.

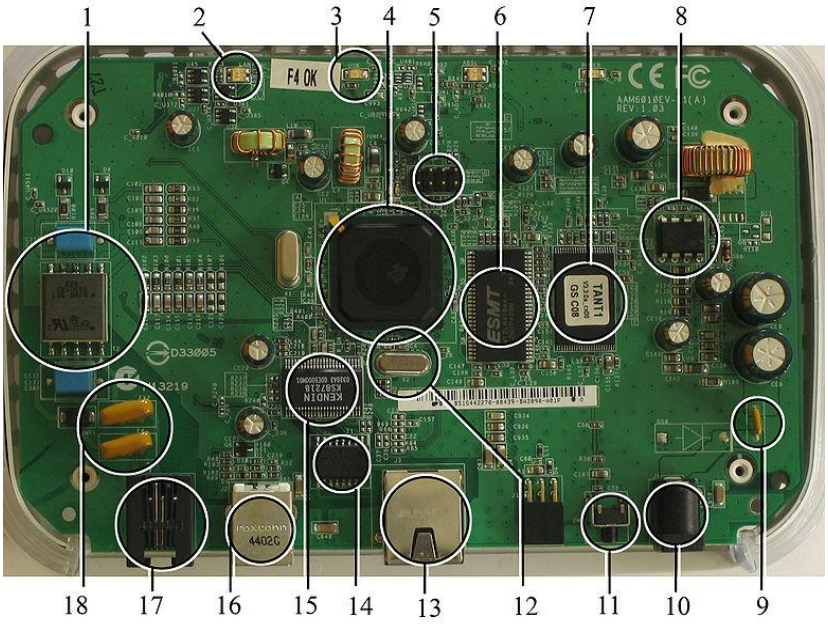


Fig 6.1: A modern example of embedded system

Labeled parts include microprocessor (4), RAM (6), flash memory (7).Embedded systems programming is not like normal PC programming. In many ways, programming for an embedded system is like programming PC 15 years ago. The hardware for the system is usually chosen to make the device as cheap as possible. Spending an extra dollar a unit in order to make things easier to program can cost millions. Hiring a programmer for an extra month is cheap in comparison. This means the programmer must make do with slow processors and low memory, while at the same time battling a need for efficiency not seen in most PC applications. Below is a list of issues specific to the embedded field.

6.1.1 History:

In the earliest years of computers in the 1930–40s, computers were sometimes dedicated to a single task, but were far too large and expensive for most kinds of tasks performed by embedded computers of today. Over time however, the concept of programmable controllers evolved from traditional electromechanical sequencers, via solid state devices, to the use of computer technology.

One of the first recognizably modern embedded systems was the Apollo Guidance Computer, developed by Charles Stark Draper at the MIT Instrumentation Laboratory. At the project's inception, the Apollo guidance computer was considered the riskiest item in the Apollo project as it employed the then newly developed monolithic integrated circuits to reduce the size and weight. An early mass-produced embedded system was the Autonetics D-17 guidance computer for the Minuteman missile, released in 1961. It was built from transistor logic and had ahard disk for main memory. When the Minuteman II went into production in 1966, the D-17 was replaced with a new computer that was the first high-volume use of integrated circuits.

6.1.2 Tools:

Embedded development makes up a small fraction of total programming. There's also a large number of embedded architectures, unlike the PC world where 1 instruction set rules, and the Unix world where there's only 3 or 4 major ones. This means that the tools are more expensive. It also means that they're lowering featured, and less developed. On a major embedded project, at some point you will almost always find a compiler bug of some sort.

Debugging tools are another issue. Since you can't always run general programs on your embedded processor, you can't always run a debugger on it. This makes fixing your program difficult. Special hardware such as JTAG ports can overcome this issue in part. However, if you stop on a breakpoint when your system is controlling real world hardware (such as a motor), permanent equipment damage can occur. As a result, people doing embedded programming quickly become masters at using serial IO channels and error message style debugging.

6.1.3 Resources:

To save costs, embedded systems frequently have the cheapest processors that can do the job. This means your programs need to be written as efficiently as possible. When dealing with large data sets, issues like memory cache misses that never matter in PC programming can hurt you. Luckily, this won't happen too often- use reasonably efficient algorithms to start, and optimize only when necessary. Of course, normal profilers won't work well, due to the same reason debuggers don't work well.

Memory is also an issue. For the same cost savings reasons, embedded systems usually have the least memory they can get away with. That means their algorithms must be memory efficient (unlike in PC programs, you will frequently sacrifice processor time for

memory, rather than the reverse). It also means you can't afford to leak memory. Embedded applications generally use deterministic memory techniques and avoid the default "new" and "malloc" functions, so that leaks can be found and eliminated more easily.Other resources programmers expect may not even exist. For example, most embedded processors do not have hardware FPUs (Floating-Point Processing Unit). These resources either need to be emulated in software, or avoided altogether.

6.1.4 Real Time Issues:

Embedded systems frequently control hardware, and must be able to respond to them in real time. Failure to do so could cause inaccuracy in measurements, or even damage hardware such as motors. This is made even more difficult by the lack of resources available. Almost all embedded systems need to be able to prioritize some tasks over others, and to be able to put off/skip low priority tasks such as UI in favor of high priority tasks like hardware control.

6.2 Need ForEmbedded Systems:

The uses of embedded systems are virtually limitless, because every day new products are introduced to the market that utilizes embedded computers in novel ways. In recent years, hardware such as microprocessors, microcontrollers, and FPGA chips have become much cheaper. So when implementing a new form of control, it's wiser to just buy the generic chip and write your own custom software for it. Producing a custom-made chip to handle a particular task or set of tasks costs far more time and money. Many embedded computers even come with extensive libraries, so that "writing your own software" becomes a very trivial task indeed.From an implementation viewpoint, there is a major difference between a computer and an embedded system. Embedded systems are often required to provide Real-Time response. The main elements that make embedded systems unique are its reliability and ease in debugging.

6.2.1 Debugging:

Embedded debugging may be performed at different levels, depending on the facilities available. From simplest to most sophisticate they can be roughly grouped into the following areas:

- Interactive resident debugging, using the simple shell provided by the embedded operating system (e.g. Forth and Basic)
- External debugging using logging or serial port output to trace operation using either a monitor in flash or using a debug server like the Remedy Debugger which even works for heterogeneous multicore systems.
- An in-circuit debugger (ICD), a hardware device that connects to the microprocessor via a JTAG or Nexus interface. This allows the operation of the microprocessor to be controlled externally, but is typically restricted to specific debugging capabilities in the processor.
- An in-circuit emulator replaces the microprocessor with a simulated equivalent, providing full control over all aspects of the microprocessor.
- A complete emulator provides a simulation of all aspects of the hardware, allowing all of it to be controlled and modified and allowing debugging on a normal PC.
- Unless restricted to external debugging, the programmer can typically load and run software through the tools, view the code running in the processor, and start or stop its operation. The view of the code may be as assembly code or source-code.

Because an embedded system is often composed of a wide variety of elements, the debugging strategy may vary. For instance, debugging a software(and microprocessor) centric embedded system is different from debugging an embedded system where most of the processing is performed by peripherals (DSP, FPGA, co-processor). An increasing number of embedded systems today use more than one single processor core. A common problem with multi-core development is the proper synchronization of software execution. In such a case, the embedded system design may wish to check the data traffic on the busses between the processor cores, which requires very low-level debugging, at signal/bus level, with a logic analyzer, for instance.

6.2.2 Reliability:

Embedded systems often reside in machines that are expected to run continuously for years without errors and in some cases recover by themselves if an error occurs. Therefore the software is usually developed and tested more carefully than that for personal computers, and unreliable mechanical moving parts such as disk drives, switches or buttons are avoided.

Specific reliability issues may include:

- The system cannot safely be shut down for repair, or it is too inaccessible to repair. Examples include space systems, undersea cables, navigational beacons, bore-hole systems, and automobiles.
- The system must be kept running for safety reasons. "Limp modes" are less tolerable. Often backups are selected by an operator. Examples include aircraft navigation, reactor control systems, safety-critical chemical factory controls, train signals, engines on single-engine aircraft.
- The system will lose large amounts of money when shut down: Telephone switches, factory controls, bridge and elevator controls, funds transfer and market making, automated sales and service.

A variety of techniques are used, sometimes in combination, to recover from errors—both software bugs such as memory leaks, and also soft errors in the hardware:

- Watchdog timer that resets the computer unless the software periodically notifies the watchdog
- Subsystems with redundant spares that can be switched over to
- software "limp modes" that provide partial function
- Designing with a Trusted Computing Base (TCB) architecture[6] ensures a highly secure & reliable system environment
- An Embedded Hypervisor is able to provide secure encapsulation for any subsystem component, so that a compromised software component cannot interfere with other subsystems, or privileged-level system software. This encapsulation keeps faults from propagating from one subsystem to another, improving reliability. This may also allow a subsystem to be automatically shut down and restarted on fault detection.
- Immunity Aware Programming

6.3 Explanation of Embedded Systems:

6.3.1 Software Architecture:

There are several different types of software architecture in common use.

- Simple Control Loop:

In this design, the software simply has a loop. The loop calls subroutines, each of which manages a part of the hardware or software.

- Interrupt Controlled System:

Some embedded systems are predominantly interrupt controlled. This means that tasks performed by the system are triggered by different kinds of events. An interrupt could be generated for example by a timer in a predefined frequency, or by a serial port controller receiving a byte. These kinds of systems are used if event handlers need low latency and the event handlers are short and simple.

Usually these kinds of systems run a simple task in a main loop also, but this task is not very sensitive to unexpected delays. Sometimes the interrupt handler will add longer tasks to a queue structure. Later, after the interrupt handler has finished, these tasks are executed by the main loop. This method brings the system close to a multitasking kernel with discrete processes.

- Cooperative Multitasking:

A non-preemptive multitasking system is very similar to the simple control loop scheme, except that the loop is hidden in an API. The programmer defines a series of tasks, and each task gets its own environment to “run” in. When a task is idle, it calls an idle routine, usually called “pause”, “wait”, “yield”, “nop” (stands for no operation), etc.The advantages and disadvantages are very similar to the control loop, except that adding new software is easier, by simply writing a new task, or adding to the queue-interpreter.

- Primitive Multitasking:

In this type of system, a low-level piece of code switches between tasks or threads based on a timer (connected to an interrupt). This is the level at which the system is generally considered to have an "operating system" kernel. Depending on how much functionality is required, it introduces more or less of the complexities of managing multiple tasks running conceptually in parallel.

As any code can potentially damage the data of another task (except in larger systems using an MMU) programs must be carefully designed and tested, and access to shared data must be controlled by some synchronization strategy, such as message queues, semaphores or a non-blocking synchronization scheme.

Because of these complexities, it is common for organizations to buy a real-time operating system, allowing the application programmers to concentrate on device functionality rather than operating system services, at least for large systems; smaller systems often cannot afford the overhead associated with a generic real time system, due to limitations regarding memory size, performance, and/or battery life.

- Microkernels And Exokernels:

A microkernel is a logical step up from a real-time OS. The usual arrangement is that the operating system kernel allocates memory and switches the CPU to different threads of execution. User mode processes implement major functions such as file systems, network interfaces, etc.

In general, microkernels succeed when the task switching and intertask communication is fast, and fail when they are slow. Exokernels communicate efficiently by normal subroutine calls. The hardware and all the software in the system are available to, and extensible by application programmers. Based on performance, functionality, requirement the embedded systems are divided into three categories:

6.3.2 Stand Alone Embedded System:

These systems takes the input in the form of electrical signals from transducers or commands from human beings such as pressing of a button etc.., process them and produces desired output. This entire process of taking input, processing it and giving output is done in standalone mode. Such embedded systems comes under stand alone embedded systems

Eg: microwave oven, air conditioner etc..

6.3.3 Real-time embedded systems:

Embedded systems which are used to perform a specific task or operation in a specific time period those systems are called as real-time embedded systems. There are two types of real-time embedded systems.

- Hard Real-time embedded systems:

These embedded systems follow an absolute dead line time period i.e.., if the tasking is not done in a particular time period then there is a cause of damage to the entire equipment.

Eg: consider a system in which we have to open a valve within 30 milliseconds. If this valve is not opened in 30 ms this may cause damage to the entire equipment. So in such cases we use embedded systems for doing automatic operations.

- Soft Real Time embedded systems:

These embedded systems follow a relative dead line time period i.e.., if the task is not done in a particular time that will not cause damage to the equipment.

Eg: Consider a TV remote control system, if the remote control takes a few milliseconds delay it will not cause damage either to the TV or to the remote control. These systems which will not cause damage when they are not operated at considerable time period those systems comes under soft real-time embedded systems.

6.3.4 Network communication embedded systems:

A wide range network interfacing communication is provided by using embedded systems.

- Consider a web camera that is connected to the computer with internet can be used to spread communication like sending pictures, images, videos etc.., to another computer with internet connection throughout anywhere in the world.

- Consider a web camera that is connected at the door lock.

Whenever a person comes near the door, it captures the image of a person and sends to the desktop of your computer which is connected to internet. This gives an alerting message with image on to the desktop of your computer, and then you can open the door lock just by clicking the mouse. Fig: 3.2 show the network communications in embedded systems.

Fig 6.2: Network communication embedded systems

6.3.5 Different types of processing units:

The central processing unit (c.p.u) can be any one of the following microprocessor, microcontroller, digital signal processing.

- Among these Microcontroller is of low cost processor and one of the main advantage of microcontrollers is, the components such as memory, serial communication interfaces, analog to digital converters etc.., all these are built on a single chip. The numbers of external components that are connected to it are very less according to the application.

- Microprocessors are more powerful than microcontrollers. They are used in major applications with a number of tasking requirements. But the microprocessor requires many external components like memory, serial communication, hard disk, input output ports etc.., so the power consumption is also very high when compared to microcontrollers.

- Digital signal processing is used mainly for the applications that particularly involved with processing of signals

6.4 APPLICATIONS OF EMBEDDED SYSTEMS:

- Avionics, such as inertial guidance systems, flight control hardware/software and other integrated systems in aircraft and missiles
- Cellular telephones and telephone switches
- Engine controllers and antilock brake controllers for automobiles
- Home automation products, such as thermostats, air conditioners, sprinklers, and security monitoring systems
- Handheld calculators and computers
- Household appliances, including microwave ovens, washing machines, television sets etc
- Medical equipment&Personal digital assistant
- Computer peripherals such as routers and printers

CHAPTER-7

HARDWARE DESCRIPTION

7.1 Micro controller: 6.1 Micro controller:

Fig: 7.1 : Microcontrollers

7.1.1 Introduction to Microcontrollers:

Circumstances that we find ourselves in today in the field of microcontrollers had their beginnings in the development of technology of integrated circuits. This development has made it possible to store hundreds of thousands of transistors into one chip. That was a prerequisite for production of microprocessors, and the first computers were made by adding external peripherals such as memory, input-output lines, timers and other. Further increasing of the volume of the package resulted in creation of integrated circuits. These integrated circuits contained both processor and peripherals. That is how the first chip containing a microcomputer, or what would later be known as a microcontroller came about.

Microprocessors and microcontrollers are widely used in embedded systems products. Microcontroller is a programmable device. A microcontroller has a CPU in addition to a fixed amount of RAM, ROM, I/O ports and a timer embedded all on a single chip. The fixed amount of on-chip ROM, RAM and number of I/O ports in microcontrollers makes them ideal for many applications in which cost and space are critical.

The AVR is a modified Harvard architecture 8-bit RISC single chip microcontroller which was developed by Atmel in 1996. The AVR was one of the first

microcontroller families to use on-chip flash memory for program storage, as opposed to One-Time Programmable ROM, EPROM, or EEPROM used by other microcontrollers at the time.

Crystal Oscillator:

XTAL1 and XTAL2 are input and output, respectively, of an inverting amplifier which can be configured for use as an On-chip Oscillator, Either a quartz

Crystal or a ceramic resonator may be used. The CKOPT Fuse selects between two different Oscillator amplifier modes. When CKOPT is programmed, the Oscillator output will oscillate a full rail-to-rail swing on the output. This mode is suitable when operating in a very noisy environment or when the output from XTAL2 drives a second clock buffer. This mode has a wide frequency range. When CKOPT is unprogrammed, the Oscillator has a smaller output swing. This reduces power consumption considerably.

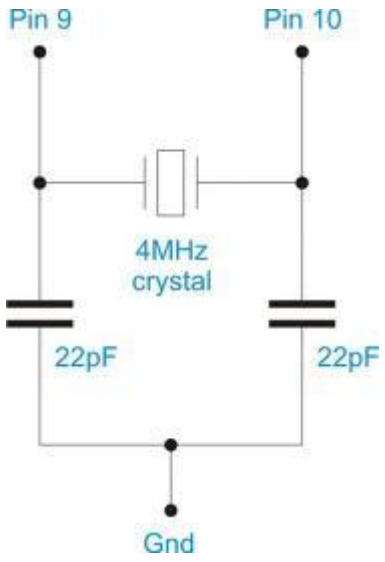


Fig 7.2 : Crystal Oscillator

This mode has a limited frequency range and it cannot be used to drive other clock buffers. For resonators, the maximum frequency is 8 MHz with CKOPT unprogrammed and 16 MHz with CKOPT programmed. C1 and C2 should always be equal for both crystals and resonators. The optimal value of the capacitors depends on the crystal or resonator in use, the amount of stray capacitance, and the electromagnetic noise of the environment. For ceramic resonators, the capacitor values given by the manufacturer should be used. The Oscillator can operate in three different modes, each optimized for a specific frequency range. The operating mode is selected by the fuses CKSEL3..1

7.1.2 Architecture:

Memory: It has **8 Kb** of Flash program memory (10,000 Write/Erase cycles durability), **512 Bytes** of EEPROM (100,000 Write/Erase Cycles). **1Kbyte** Internal SRAM

I/O Ports: 23 I/ line can be obtained from three ports; namely Port B, Port C and Port D.

Interrupts: Two External Interrupt source, located at port D. 19 different interrupt vectors supporting 19 events generated by internal peripherals.

Timer/Counter: Three Internal Timers are available, two 8 bit, one 16 bit, offering various operating modes and supporting internal or external clocking.

SPI (Serial Peripheral interface): ATmega8 holds three communication devices integrated. One of them is Serial Peripheral Interface. Four pins are assigned to Atmega8 to implement this scheme of communication.

USART: One of the most powerful communication solutions is USART and ATmega8 supports both synchronous and asynchronous data transfer schemes. It has three pins assigned for that. In many projects, this module is extensively used for PC-Micro controller communication.

TWI (Two Wire Interface): Another communication device that is present in ATmega8 is Two Wire Interface. It allows designers to set up a commutation between two devices using just two wires along with a common ground connection, As the TWI output is made by means of open collector outputs, thus external pull up resistors are required to make the circuit.

Analog Comparator: A comparator module is integrated in the IC that provides comparison facility between two voltages connected to the two inputs of the Analog comparator via External pins attached to the micro controller.

Analog to Digital Converter: Inbuilt analog to digital converter can convert an analog input signal into digital data of **10bit** resolution. For most of the low end application, this much resolution is enough.

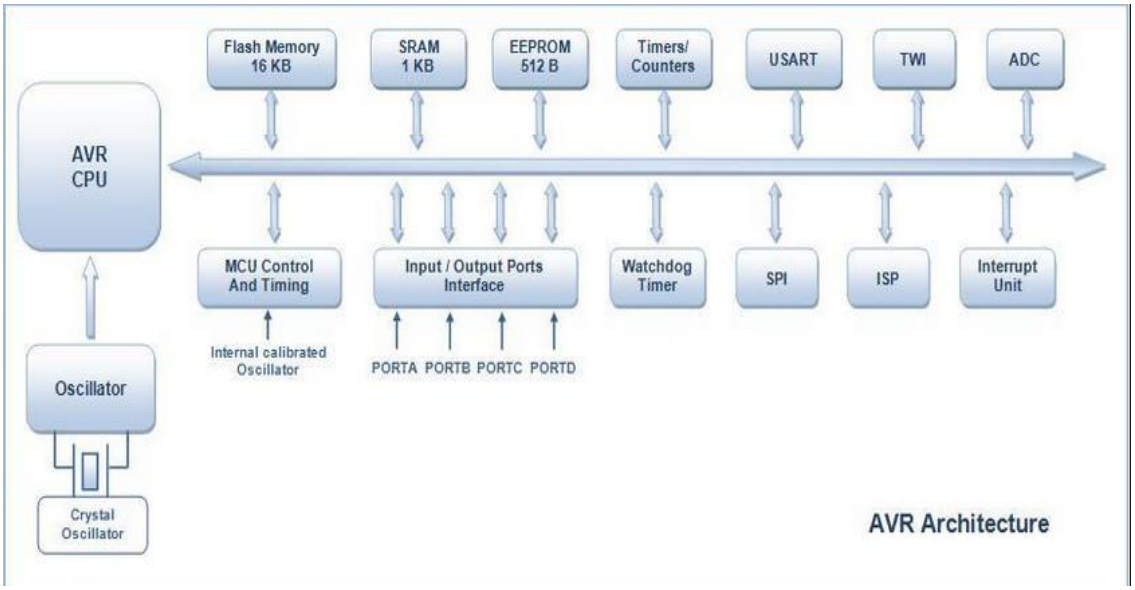


Fig 7.3 : AVR Architecture

Microcontroller: Microcontroller can be termed as a single on chip computer which includes number of peripherals like RAM, EEPROM, Timers etc., required to perform some predefined task.

The computer on one hand is designed to perform all the general purpose tasks on a single machine like you can use a computer to run a software to perform calculations or you can use a computer to store some multimedia file or to access internet through the browser, whereas the microcontrollers are meant to perform only the specific tasks, for e.g., switching the AC off automatically when room temperature drops to a certain defined limit and again turning it ON when temperature rises above the defined limit.

There are number of popular families of microcontrollers which are used in different applications as per their capability and feasibility to perform the desired task, most common of these are Arduino, **AVR** and PIC microcontrollers. In this article we will introduce you with **AVR** family of microcontrollers.

AVR was developed in the year 1996 by Atmel Corporation. The architecture of **AVR** was developed by Alf-Egil Bogen and Vegard Wollan. AVR derives its name from its developers and stands for **Alf-Egil Bogen Vegard Wollan RISC microcontroller**, also known as **A**dvanced **V**irtual **R**ISC. The AT90S8515 was the first microcontroller which was based on **AVR architecture** however the first microcontroller to hit the commercial market was AT90S1200 in the year 1997.

AVR microcontrollers are available in three categories:

1. **TinyAVR** – Less memory, small size, suitable only for simpler applications

2. **MegaAVR** – These are the most popular ones having good amount of memory (upto 256 KB), higher number of inbuilt peripherals and suitable for moderate to complex applications.

3. **XmegaAVR** – Used commercially for complex applications, which require large program memory and high speed.

ARCHITECTURE

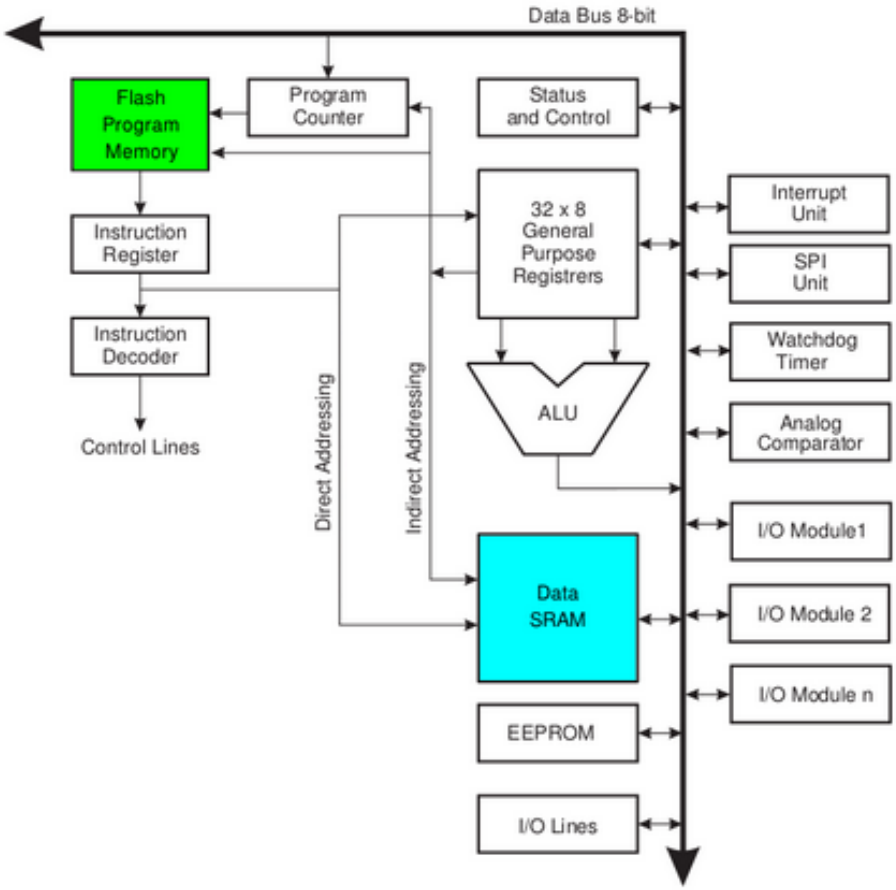


Fig 7.4 : Architecture

Device architecture

Flash, EEPROM, and SRAM are all integrated onto a single chip, removing the need for external memory in most applications. Some devices have a parallel external bus option to allow adding additional data memory or memory-mapped devices. Almost all devices (except the smallest TinyAVR chips) have serial interfaces, which can be used to connect larger serial EEPROMs or flash chips.

Program memory

Program instructions are stored in non-volatile flash memory. Although the MCUs are 8-bit, each instruction takes one or two 16-bit words.

The size of the program memory is usually indicated in the naming of the device itself (e.g., the ATmega64x line has 64 kB of flash while the ATmega32x line has 32 kB).

There is no provision for off-chip program memory; all code executed by the AVR core must reside in the on-chip flash. However, this limitation does not apply to the AT94 FPSLIC AVR/FPGA chips.

Internal data memory

The data address space consists of the register file, I/O registers, and SRAM.

Internal registers

The AVRs have 32 single-byte registers and are classified as 8-bit RISC devices.

In most variants of the AVR architecture, the working registers are mapped in as the first 32 memory addresses (0000_{16}–$001F_{16}$) followed by the 64 I/O registers (0020_{16}–$005F_{16}$).

Actual SRAM starts after these register sections (address 0060_{16}). (Note that the I/O register space may be larger on some more extensive devices, in which case the memory mapped I/O registers will occupy a portion of the SRAM address space.)

Even though there are separate addressing schemes and optimized opcodes for register file and I/O register access, all can still be addressed and manipulated as if they were in SRAM.

In the XMEGA variant, the working register file is not mapped into the data address space; as such, it is not possible to treat any of the XMEGA's working registers as though they were SRAM. Instead, the I/O registers are mapped into the data address space starting at the very beginning of the address space. Additionally, the amount of data address space dedicated to I/O registers has grown substantially to 4096 bytes (0000_{16}–$0FFF_{16}$). As with previous generations, however, the fast I/O manipulation instructions can only reach the first 64 I/O register locations (the first 32 locations for bitwise instructions). Following the I/O registers, the XMEGA series sets aside a 4096 byte range of the data address space which can be used optionally for mapping the internal EEPROM to the data address space (1000_{16}–$1FFF_{16}$). The actual SRAM is located after these ranges, starting at 2000_{16}.

EEPROM

Almost all AVR microcontrollers have internal EEPROM for semi-permanent data storage. Like flash memory, EEPROM can maintain its contents when electrical power is removed.

In most variants of the AVR architecture, this internal EEPROM memory is not mapped into the MCU's addressable memory space. It can only be accessed the same way an external peripheral device is, using special pointer registers and read/write instructions which makes EEPROM access much slower than other internal RAM.

However, some devices in the SecureAVR (AT90SC) family use a special EEPROM mapping to the data or program memory depending on the configuration. The XMEGA family also allows the EEPROM to be mapped into the data address space.

Since the number of writes to EEPROM is not unlimited — Atmel specifies 100,000 write cycles in their datasheets — a well designed EEPROM write routine should compare the contents of an EEPROM address with desired contents and only perform an actual write if the contents need to be changed.

Note that erase and write can be performed separately in many cases, byte-by-byte, which may also help prolong life when bits only need to be set to all 1s (erase) or selectively cleared to 0s (write).

Program execution

Atmel's AVRs have a two stage, single level pipeline design. This means the next machine instruction is fetched as the current one is executing. Most instructions take just one or two clock cycles, making AVRs relatively fast among eight-bit microcontrollers.

The AVR processors were designed with the efficient execution of compiled C code in mind and have several built-in pointers for the task.

MCU speed

The AVR line can normally support clock speeds from 0 to 20 MHz, with some devices reaching 32 MHz. Lower powered operation usually requires a reduced clock speed. All recent (Tiny, Mega, and Xmega, but not 90S) AVRs feature an on-chip oscillator, removing the need for external clocks or resonator circuitry. Some AVRs also have a system clock prescaler that can divide down the system clock by up to 1024. This prescaler can be reconfigured by software during run-time, allowing the clock speed to be optimized.

Since all operations (excluding literals) on registers R0 - R31 are single cycle, the AVR can achieve up to 1 MIPS per MHz, i.e. an 8 MHz processor can achieve up to 8 MIPS. Loads and stores to/from memory take two cycles, branching takes two cycles. Branches in the latest "3-byte PC" parts such as ATmega2560 are one cycle slower than on previous devices

Features:

• High-performance, Low-power Atmel®AVR® 8-bit Microcontroller

• Advanced RISC Architecture

– 130 Powerful Instructions – Most Single-clock Cycle Execution

– 32 × 8 General Purpose Working Registers

– Fully Static Operation

– Up to 16MIPS Throughput at 16MHz

– On-chip 2-cycle Multiplier

• High Endurance Non-volatile Memory segments

– 8Kbytes of In-System Self-programmable Flash program memory

– 512Bytes EEPROM

– 1Kbyte Internal SRAM

– Write/Erase Cycles: 10,000 Flash/100,000 EEPROM

– Data retention: 20 years at 85°C/100 years at 25°C(1)

– Optional Boot Code Section with Independent Lock Bits

In-System Programming by On-chip Boot Program

True Read-While-Write Operation

– Programming Lock for Software Security

• Peripheral Features

– Two 8-bit Timer/Counters with Separate Prescaler, one Compare Mode

– One 16-bit Timer/Counter with Separate Prescaler, Compare Mode, and Capture

Mode

– Real Time Counter with Separate Oscillator

– Three PWM Channels

– 8-channel ADC in TQFP and QFN/MLF package

Eight Channels 10-bit Accuracy

– 6-channel ADC in PDIP package

Six Channels 10-bit Accuracy

– 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

– Five Sleep Modes: Idle, ADC Noise Reduction, Power-save, Power-down, and Standby

• I/O and Packages

– 23 Programmable I/O Lines

– 28-lead PDIP, 32-lead TQFP, and 32-pad QFN/MLF

• Operating Voltages

– 2.7V - 5.5V (ATmega8L)

– 4.5V - 5.5V (ATmega8)

• Speed Grades

– 0 - 8MHz (ATmega8L)

– 0 - 16MHz (ATmega8)

• Power Consumption at 4Mhz, 3V, 25oC

– Active: 3.6mA

– Idle Mode: 1.0mA

– Power-down Mode: 0.5µA

Brown-out Detector:

If the Brown-out Detector is not needed in the application, this module should be turned off. If the Brown-out Detector is enabled by the BODEN Fuse, it will be enabled in all sleep modes, and hence, always consume power. In the deeper sleep modes, this will contribute significantly to the total current consumption. Refer to “Brown-out Detection” on page 38 for details on how to configure the Brown-out Detector.

Internal Voltage Reference the Internal Voltage Reference will be enabled when needed by the Brown-out Detector, the Analog Comparator or the ADC. If these modules are disabled as described in the sections above, the internal voltage reference will be disabled and it will not be consuming power. When turned on again, the user must allow the reference to start up before the output is used. If the reference is kept on in sleep mode, the output can be used immediately. Refer to “Internal Voltage Reference” on page 40 for details on the start-up time. Watchdog Timer If the Watchdog Timer is not needed in the application, this module should be turned off.

If the Watchdog Timer is enabled, it will be enabled in all sleep modes, and hence, always consume power. In the deeper sleep modes, this will contribute significantly to the total current consumption. Refer to “Watchdog Timer” on page 41 for details on how to configure the Watchdog Timer. Port Pins When entering a sleep mode, all port pins should be configured to use minimum power.

The most important thing is then to ensure that no pins drive resistive loads. In sleep modes where the both the I/O clock (clkI/O) and the ADC clock (clkADC) are stopped, the input buffers of the device will be disabled. This ensures that no power is consumed by the input logic when not needed. In some cases, the input logic is needed for detecting wake-up conditions, and it will then be enabled. Refer to the section “Digital Input Enable and Sleep Modes” on page 53 for details on which pins are enabled. If the input buffer is enabled and the

input signal is left floating or have an analog signal level close to VCC/2, the input buffer will use excessive power.

Power-on Reset:

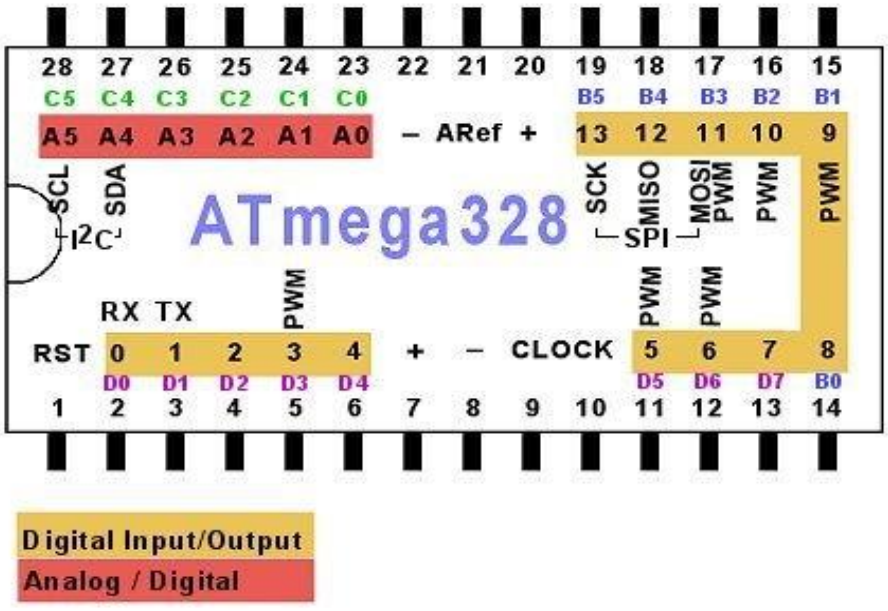
A Power-on Reset (POR) pulse is generated by an On-chip detection circuit. The detection level is defined in Table 15. The POR is activated whenever VCC is below the detection level. The POR circuit can be used to trigger the Start-up Reset, as well as to detect a failure in supply voltage.

A Power-on Reset (POR) circuit ensures that the device is reset from Power-on. Reaching the Power-on Reset threshold voltage invokes the delay counter, which determines how long the device is kept in RESET after VCC rise. The RESET signal is activated again, without any delay, when VCC decreases below the detection level.

External Reset:

An External Reset is generated by a low level on the RESET pin. Reset pulses longer than the minimum pulse width (see Table 15) will generate a reset, even if the clock is not running. Shorter pulses are not guaranteed to generate a reset. When the applied signal reaches the Reset Threshold Voltage – VRST on its positive edge, the delay counter starts the MCU after the time-out period tTOUT has expired.

7.1.3 Pin diagram:



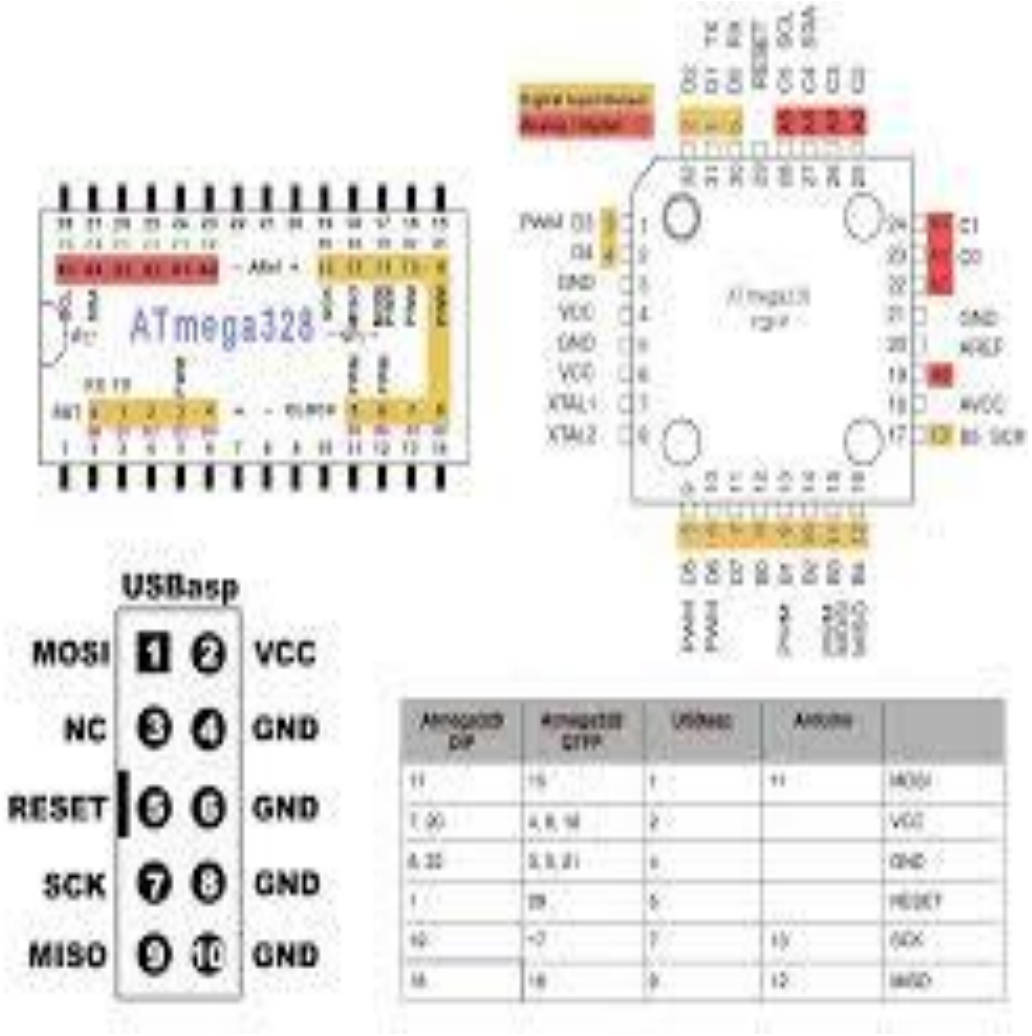


Fig 7.5 : PIN DIAGRAM OF ATMEGA328

VCC

Digital supply voltage magnitude of the voltage range between 4.5 to 5.5 V for the ATmega8 and 2.7 to 5.5 V for ATmega8L

GND

Ground Zero reference digital voltage supply.

PORTB (PB7.. PB0)

PORTB is a port I / O two-way (bidirectional) 8-bit with internal pull-up resistor can be selected. This port output buffers have symmetrical characteristics when used as a source or sink. When used as an input, the pull-pin low externally will emit a current if the pull-up resistor is activated it. PORTB pins will be in the condition of the tri-state when RESET is active, although the clock is not running.

PORTC (PC5.. PC0)

PORTC is a port I / O two-way (bidirectional) 7-bit with internal pull-up resistor can be selected. This port output buffers have symmetrical characteristics when used as a source or sink. When used as an input, the pull-pin low externally will emit a current if the pull-up resistor is activated it. PORTC pins will be in the condition of the tri-state when RESET is active, although the clock is not running.

PC6/RESET

If RSTDISBL Fuse programmed, PC6 then serves as a pin I / O but with different characteristics. PC0 to PC5 If Fuse RSTDISBL not programmed, then serves as input Reset PC6. LOW signal on this pin with a minimum width of 1.5 microseconds will bring the microcontroller into reset condition, although the clock is not running.

PORTD (PD7.. PD0)

PORTD is a port I / O two-way (bidirectional) 8-bit with internal pull-up resistor can be selected. This port output buffers have symmetrical characteristics when used as a source or sink. When used as an input, the pull-pin low externally will emit a current if the pull-up resistor is activated it. PORTD pins will be in the condition of the tri-state when RESET is active, although the clock is not running.

RESET

Reset input pin. LOW signal on this pin with a minimum width of 1.5 microseconds will bring the microcontroller into reset condition, although the clock is not running. Signal with a width of less than 1.5 microseconds does not guarantee a Reset condition.

AVCC

AVCC is the supply voltage pin for the ADC, PC3 .. PC0, and ADC7..ADC6. This pin should be connected to VCC, even if the ADC is not used. If the ADC is used, AVCC should be connected to VCC through a low-pass filter to reduce noise.

Aref

Analog Reference pin for the ADC.

ADC7 .. ADC6

ADC analog input there is only on ATmega8 with TQFP and QFP packages / MLF.

PORTS

Term "port" refers to a group of pins on a microcontroller which can be accessed simultaneously, or on which we can set the desired combination of zeros and ones, or read from

them an existing status. Physically, port is a register inside a microcontroller which is connected by wires to the pins of a microcontroller. Ports represent physical connection of Central Processing Unit with an outside world. Microcontroller uses them

The Atmega8 has 23 I/O ports which are organized into 3 groups:

- Port B (PB0 to PB7)
- Port C (PC0 to PC6)
- Port D (PD0 to PD7)

We will use mainly 3 registers known as **DDRX, PORTX** & **PINX**. We have total four PORTs on my ATmega16. They are **PORTA, PORTB, PORTC** and **PORTD**. They are multifunctional pins. Each of the pins in each port (total 32) can be treated as input or output pin.

Applications

AVR microcontroller perfectly fits many uses, from automotive industries and controlling home appliances to industrial instruments, remote sensors, electrical door locks and safety devices. It is also ideal for smart cards as well as for battery supplied devices because of its low consumption.

EEPROM memory makes it easier to apply microcontrollers to devices where permanent storage of various parameters is needed (codes for transmitters, motor speed, receiver frequencies, etc.). Low cost, low consumption, easy handling and flexibility make ATmega8 applicable even in areas where microcontrollers had not previously been considered (example: timer functions, interface replacement in larger systems, coprocessor applications, etc.).

In System Programmability of this chip (along with using only two pins in data transfer) makes possible the flexibility of a product, after assembling and testing have been completed. This capability can be used to create assembly-line production, to store calibration data available only after final testing, or it can be used to improve programs on finished products.

Keypad

The keypad used in this project is 4x4 keypad which have eight I/O pins. It is divided as rows and columns the first four pins (pink, light blue, black, orange) of the keypad are the columns

while the following four pins (red, yellow, dark blue, purple) are the rows. Circuit of the keypad

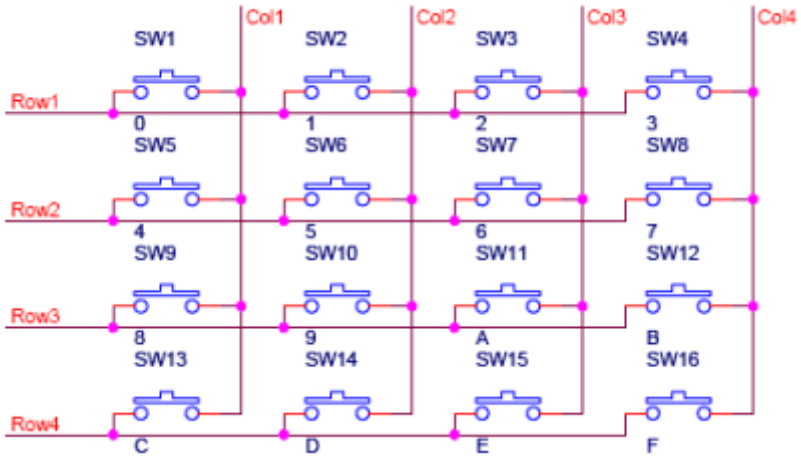


Fig 7.6 : Circuit of the keypad

7.2 REGULATED POWER SUPPLY:

Power supply is a supply of electrical power. A device or system that supplies electrical or other types of energy to an output load or group of loads is called a power supply unit or PSU. The term is most commonly applied to electrical energy supplies, less often to mechanical ones, and rarely to others.A power supply may include a power distribution system as well as primary or secondary sources of energy such asConversion of one form of electrical power to another desired form and voltage, typically involving converting AC line voltage to a well-regulated lower-voltage DC for electronic devices. Low voltage, low power DC power supply units are commonly integrated with the devices they supply, such as computers and household electronics.

- Batteries.
- Chemical fuel cells and other forms of energy storage systems.
- Solar power.
- Generators or alternators.

7.2.1 POWER SUPPLY:

All digital circuits require regulated power supply. In this article we are going to learn how to get a regulated positive supply from the mains supply.

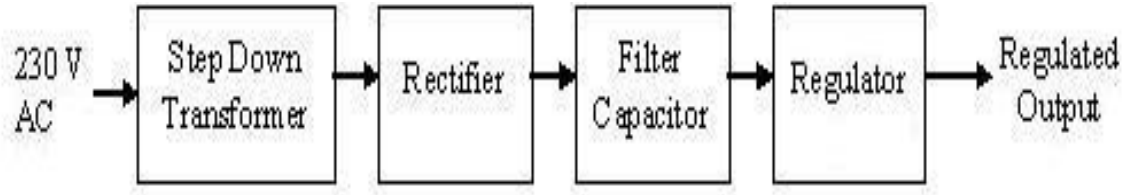


Fig 7.7 : Block diagram of power supply

the basic block diagram of a fixed regulated power supply. The basic circuit diagram of a regulated power supply (DC O/P) with led connected as load

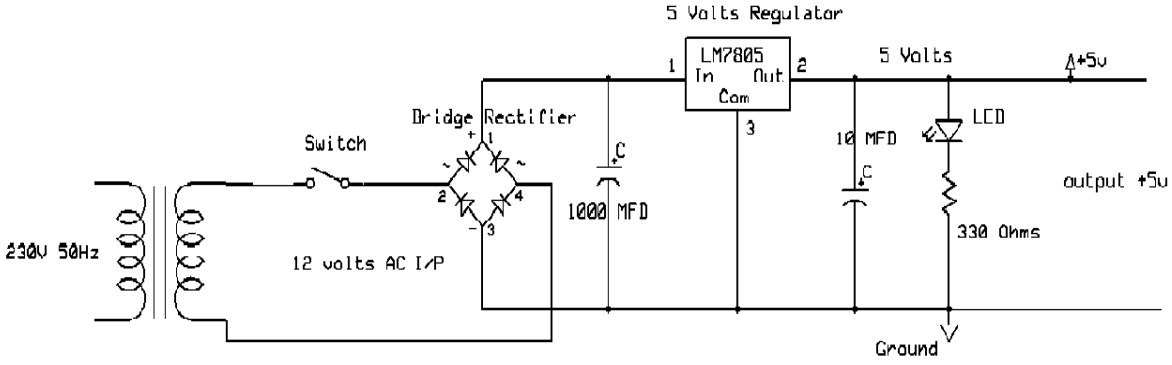


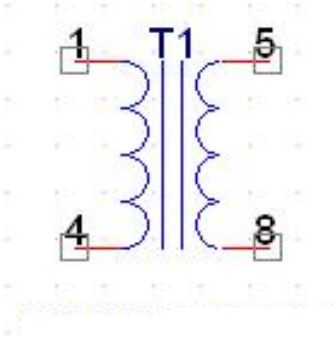
Fig 7.8 : Circuit diagram of Regulated Power Supply with Led connection

The components mainly used in above figure are

- 230V AC MAINS
- TRANSFORMER
- BRIDGE RECTIFIER(DIODES)
- CAPACITOR
- VOLTAGE REGULATOR(IC 7805)
- RESISTOR
- LED(LIGHT EMITTING DIODE)

The detailed explanation of each and every component mentioned above is as follows:

TRANSFORMER



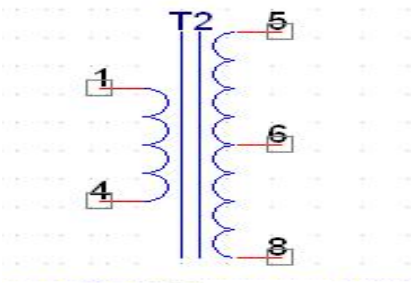


Fig :7.9 Transformer **Centre-Tap Transformer**

A transformer consists of two coils also called as “WINDINGS” namely PRIMARY & SECONDARY. They are linked together through inductively coupled electrical conductors also called as CORE. A changing current in the primary causes a change in the Magnetic Field in the core & this in turn induces an alternating voltage in the secondary coil. If load is applied to the secondary then an alternating current will flow through the load. If we consider an ideal condition then all the energy from the primary circuit will be transferred to the secondary circuit through the magnetic field. So

$$P_{primary} = P_{secondary}$$

$$I_p V_p = I_s V_s$$

$$\frac{V_s}{V_p} = \frac{N_s}{N_p}$$

Bridge Rectifier

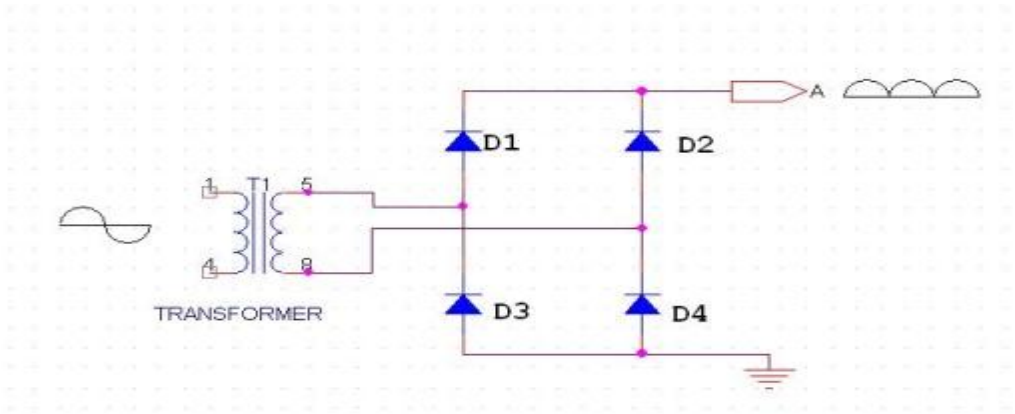


Fig 7.10 Half wave Bridge rectifier

As the name suggests it converts the full wave i.e. both the positive & the negative half cycle into DC thus it is much more efficient than Half Wave Rectifier & that too without using a center tapped transformer thus much more cost effective than Full Wave Rectifier. It consists of four diodes namely D1, D2, D3 and D4. During the positive half cycle diodes D1 & D4 conduct whereas in the negative half cycle diodes D2 & D3 conduct thus the diodes keep switching the transformer connections so we get positive half cycles in the output.

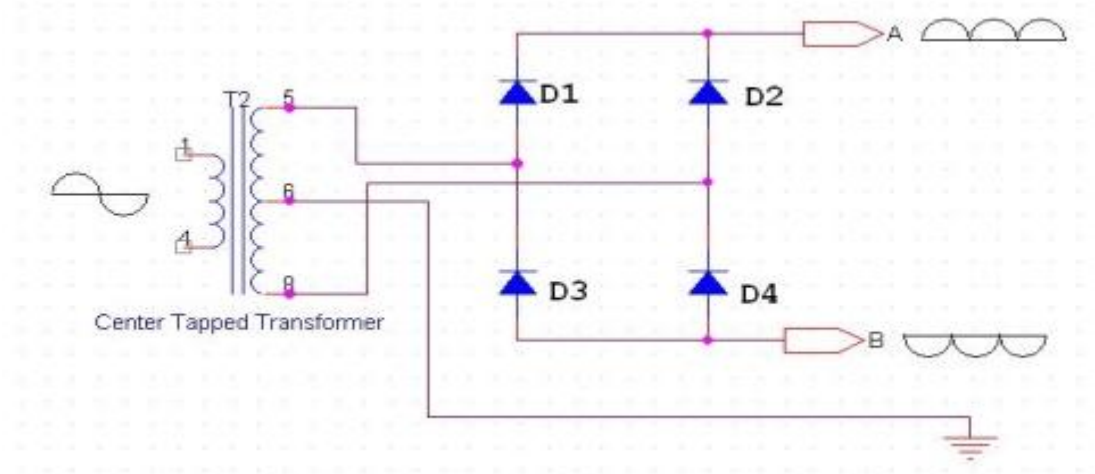


Fig 7.11 : Full wave Bridge rectifier

If we use a center tapped transformer for a bridge rectifier we can get both positive & negative half cycles which can thus be used for generating fixed positive & fixed negative voltages.

FILTER CAPACITOR

Even though half wave & full wave rectifier give DC output, none of them provides a constant output voltage. For this we require to smoothen the waveform received from the rectifier. This can be done by using a capacitor at the output of the rectifier this capacitor is also called as “FILTER CAPACITOR” or “SMOOTHING CAPACITOR” or “RESERVOIR CAPACITOR”. Even after using this capacitor a small amount of ripple will remain.We place the Filter Capacitor at the output of the rectifier the capacitor will charge to the peak voltage during each half cycle then will discharge its stored energy slowly through the load while the rectified voltage drops to zero, thus trying to keep the voltage as constant as possible.

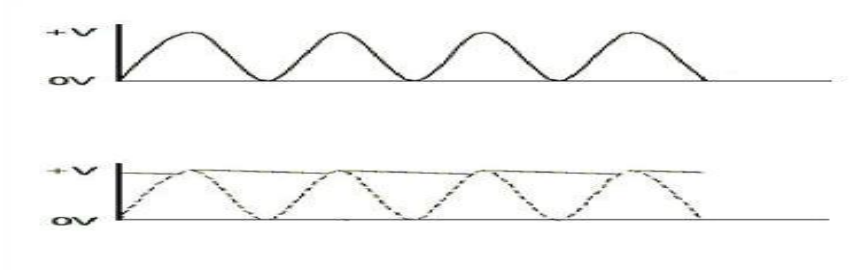


Fig 7.12 : Input and Output waveforms

If we go on increasing the value of the filter capacitor then the Ripple will decrease. But Then the costing will increase. The value of the Filter capacitor depends on the current consumed By the circuit, the frequency of the waveform & the accepted ripple.

$$C = \frac{V_r F}{I}$$

where,

Vr= accepted ripple voltage. (Should not be more than 10% of the voltage)

I= current consumed by the circuit in Amperes.

F= frequency of the waveform.

A half wave rectifier has only one peak in one cycle so F=25 Hz

whereas a full wave rectifier has Two peaks in one cycle so F=100 Hz.

7.3 VOLTAGE REGULATOR

A voltage regulator (also called a 'regulator') with only three terminals appears to be a simple device, but it is in fact a very complex integrated circuit. It converts a varying input voltage into a constant 'regulated' output voltage. Voltage Regulators are available in a variety of outputs like 5V, 6V, 9V, 12V and 15V. The LM78XX series of voltage regulators are designed for positive input. For applications requiring negative input, the LM79XX series is used. Using a pair of 'voltage-divider', Resistors can increase the output voltage of a regulator circuit.

It is not possible to obtain a voltage lower than the stated rating. You cannot use a 12V regulator to make a 5V power supply. Voltage regulators are very robust. These can withstand over-current draw due to short circuits and also over-heating. In both cases, the regulator will cut off before any damage occurs. The only way to destroy a regulator is to apply reverse voltage to its input. Reverse polarity destroys the regulator almost instantly. Fig: 3.3.11 shows voltage regulator. The L78 series of three terminal positive regulators is available in TO-220, TO-220FP, DPAK, DPAK packages and several fixed output voltages, making it useful in a wide of applications.

These regulators can provide local on-card regulation eliminating the distribution problems associated with single plant regulation. Each type embeds internal current limiting, thermal shutdown and safe area protection, making is essential indestructible. If adequate heat sinking is provided, they can deliver over 1A output current. Although designed primarily as fixed voltage regulators, these devices can be used with external components to obtain adjustable voltage and currents

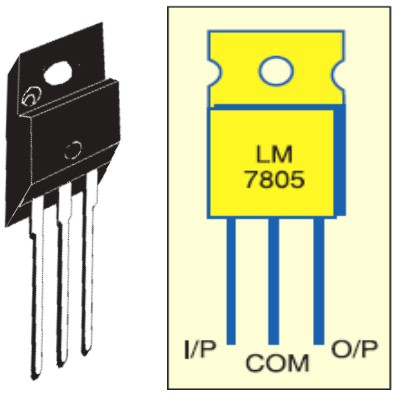


Fig 7.13 : TO-220

FEATURES OF LM7805

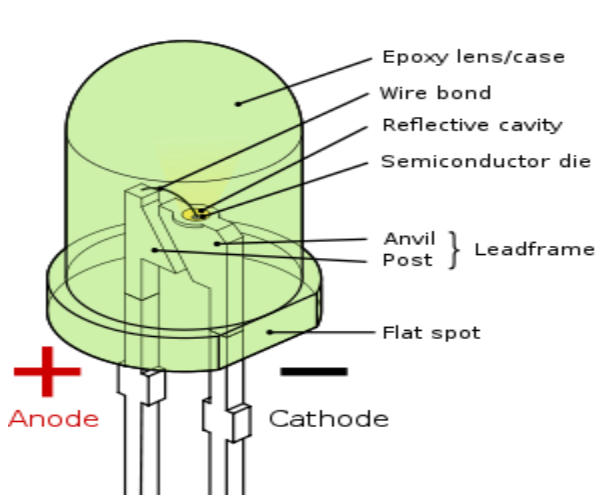
- Output current up to 1.5A
- Output voltages of 5,6,8,8.5,9,12,15,18,24V
- Thermal overload protection
- Short circuit protection output transition SOA protection
- 2/ output voltage tolerance
- Guaranteed in extended temperature range

7.4 LED

A light emitting diode (LED) is a semiconductor light source. LEDs are used as indicator lamps in many devices, and are increasingly used for lighting. Introduced as a practical electronic component in 1962, early LEDs emitted low-intensity red light, but modern versions are available across the visible, ultraviolet and infrared wavelengths, with very high brightness.

The LED is based on the semiconductor diode. When a diode is forward biased, electrons are able to recombine with holes within the device, releasing energy in the form of photons. This effect is called electroluminescence and the color of the light

(corresponding to the energy of the photon) is determined by the energy gap of the semiconductor. An LED is usually small in area (less than 1 mm^2), and integrated optical components are used to shape its radiation pattern and assist in reflection. LEDs present many advantages over incandescent light sources including lower energy consumption, longer lifetime, improved robustness, smaller size, faster switching, and greater durability and reliability. However, they are relatively expensive and require more precise current and heat management than traditional light sources. Current LED products for general lighting are more expensive to buy than fluorescent lamp sources of comparable output.



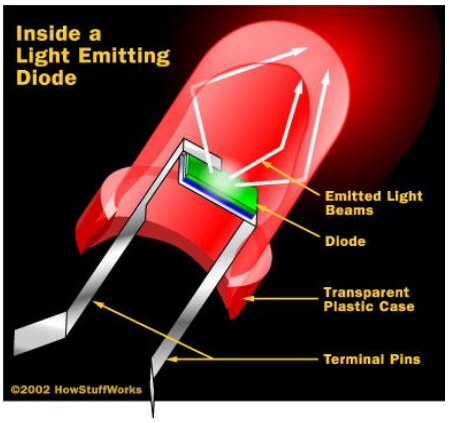


Fig 7.14 : Working of LED

7.5 D.C. Motor ROBOT

A dc motor uses electrical energy to produce mechanical energy, very typically through the interaction of magnetic fields and current-carrying conductors. The reverse process, producing electrical energy from mechanical energy, is accomplished by an alternator, generator or dynamo. Many types of electric motors can be run as generators, and vice versa. The input of a DC motor is current/voltage and its output is torque (speed).

Fig 7.15 : DC Motor

The DC motor has two basic parts: the rotating part that is called the armature and the stationary part that includes coils of wire called the field coils. The stationary part is also called the stator. Figure shows a picture of a typical DC motor, Figure shows a picture of a DC armature, and Fig shows a picture of a typical stator. From the picture you can see the armature is made of coils of wire wrapped around the core, and the core has an extended shaft that rotates on bearings. You should also notice that the ends of each coil of wire on the armature are terminated at one end of the armature. The termination points are called the commutator, and this is where the brushes make electrical contact to bring electrical current from the stationary part to the rotating part of the machine.

Operation:

The DC motor you will find in modem industrial applications operates very similarly to the simple DC motor described earlier in this chapter. Figure 12-9 shows an electrical diagram of a simple DC motor. Notice that the DC voltage is applied directly to the field winding and the brushes. The armature and the field are both shown as a coil of wire. In later diagrams, a field resistor will be added in series with the field to control the motor speed.

When voltage is applied to the motor, current begins to flow through the field coil from the negative terminal to the positive terminal. This sets up a strong magnetic field in the field winding. Current also begins to flow through the brushes into a commutator segment and then through an armature coil. The current continues to flow through the coil back to the brush that is attached to other end of the coil and returns to the DC power source. The current flowing in the armature coil sets up a strong magnetic field in the armature.

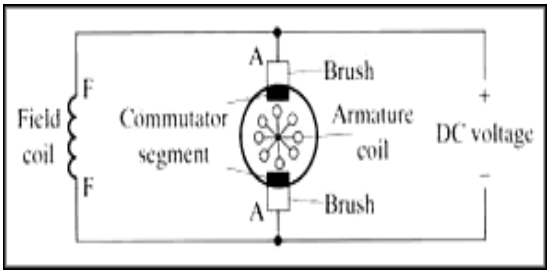


Fig 7.16 : Simple electrical diagram of DC motor

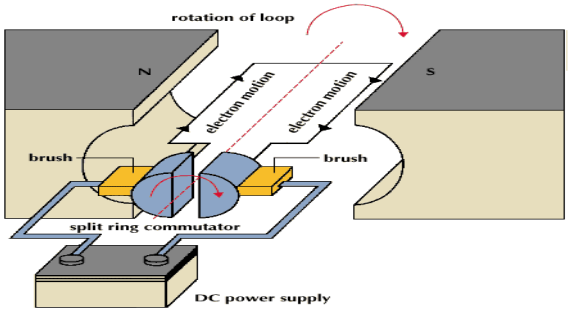


Fig 7.17 : Operation of a DC Motor

The magnetic field in the armature and field coil causes the armature to begin to rotate. This occurs by the unlike magnetic poles attracting each other and the like magnetic poles repelling each other. As the armature begins to rotate, the commutator segments will also begin to move under the brushes. As an individual commutator segment moves under the brush connected to positive voltage, it will become positive, and when it moves under a brush connected to negative voltage it will become negative. In this way, the commutator segments continually change polarity from positive to negative. Since the commutator segments are connected to the ends of the wires that make up the field winding in the armature, it causes the magnetic field in the armature to change polarity continually from North Pole to South Pole. The commutator segments and brushes are aligned in such a way that the switch in polarity of the armature coincides with the location of the armature's magnetic field and the field winding's magnetic field. The switching action is timed so that the armature will not lock up magnetically with the field. Instead the magnetic fields tend to build on each other and provide additional torque to keep the motor shaft rotating.

When the voltage is de-energized to the motor, the magnetic fields in the armature and the field winding will quickly diminish and the armature shaft's speed will begin to drop to zero. If voltage is applied to the motor again, the magnetic fields will strengthen and the armature will begin to rotate again.

Types of DC motors:

1. DC Shunt Motor,

2. DC Series Motor,

3. DC Long Shunt Motor (Compound)

4. DC Short Shunt Motor (Compound)

The rotational energy that you get from any motor is usually the battle between two magnetic fields chasing each other. The DC motor has magnetic poles and an armature, to which DC electricity is fed, The Magnetic Poles are electromagnets, and when they are energized, they produce a strong magnetic field around them, and the armature which is given power with a commutator, constantly repels the poles, and therefore rotates.

1. The DC Shunt Motor:

In a 2 pole DC Motor, the armature will have two separate sets of windings, connected to a commutator at the end of the shaft that are in constant touch with carbon brushes. The brushes are static, and the commutator rotate and as the portions of the commutator touching the respective positive or negative polarity brush will energize the respective part of the armature with the respective polarity. It is usually arranged in such a way that the armature and the poles are always repelling.

The general idea of a DC Motor is, the stronger the Field Current, the stronger the magnetic field, and faster the rotation of the armature. When the armature revolves between the poles, the magnetic field of the poles induce power in the armature conductors, and some electricity is generated in the armature, which is called back emf, and it acts as a resistance for the armature. Generally an armature has resistance of less than 1 Ohm, and powering it with heavy voltages of Direct Current could result in immediate short circuits. This back emf helps us there.

When an armature is loaded on a DC Shunt Motor, the speed naturally reduces, and therefore the back emf reduces, which allows more armatures current to flow. This results in more armature field, and therefore it results in torque.

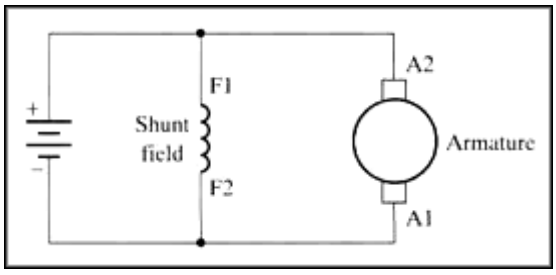


Fig 7.18 : Diagram of DC shunt motor

When a DC Shunt Motor is overloaded, if the armature becomes too slow, the reduction of the back emf could cause the motor to burn due to heavy current flow thru the armature.

The poles and armature are excited separately, and parallel, therefore it is called a Shunt Motor.

2. The DC Series Motor:

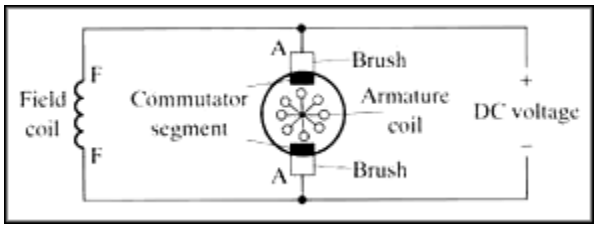


Fig 7.19 : Diagram of DC series motor

A DC Series Motor has its field coil in series with the armature. Therefore any amount of power drawn by the armature will be passed thru the field. As a result you cannot start a Series DC Motor without any load attached to it. It will either run uncontrollably in full speed, or it will stop.

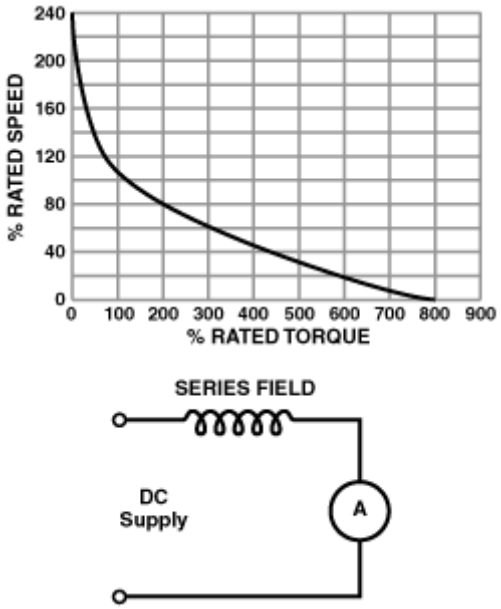


Fig 7.20: Diagram of DC series motor graph representation

When the load is increased then its efficiency increases with respect to the load applied. So these are on Electric Trains and elevators.

3. DC Compound Motor:

A compound of Series and Shunt excitation for the fields is done in a Compound DC Motor. This gives the best of both series and shunt motors. Better torque as in a series motor, while the possibility to start the motor with no load.

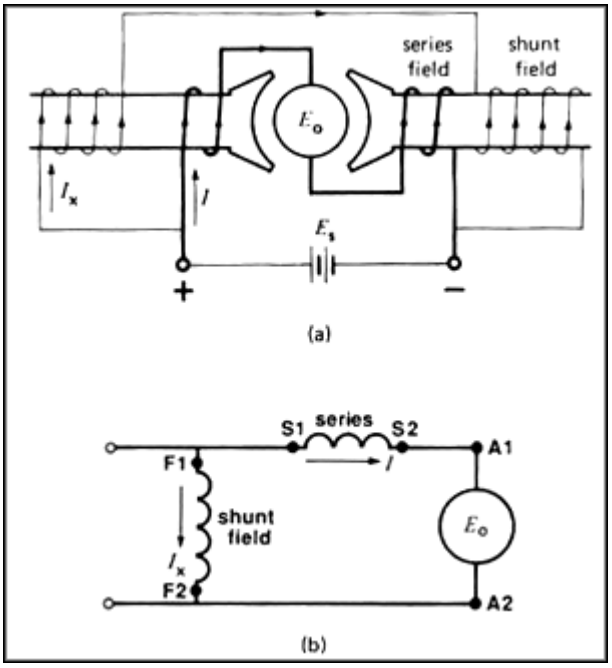


Fig 7.21 : Diagram of DC compound motor

Above is the diagram of a long shunt motor, while in a short shunt, the shunt coil will be connected after the serial coil.

A Compound motor can be run as a shunt motor without connecting the serial coil at all but not vice versa.

7.6 METAL DETECTOR

A **metal detector** is an electronic instrument which detects the presence of metal nearby. Metal detectors are useful for finding metal inclusions hidden within objects, or metal objects buried underground. They often consist of a handheld unit with a sensor probe which can be swept over the ground or other objects. If the sensor comes near a piece of metal this is indicated by a changing tone in earphones, or a needle moving on an indicator. Usually the device gives some indication of distance; the closer the metal is, the higher the tone in the earphone or the higher the needle goes. Another common type are stationary "walk through" metal detectors used

for security screening at access points in prisons, courthouses, and airports to detect concealed metal weapons on a person's body.

The simplest form of a metal detector consists of an oscillator producing an alternating current that passes through a coil producing an alternating magnetic field. If a piece of electrically conductive metal is close to the coil, eddy currents will be induced in the metal, and this produces a magnetic field of its own. If another coil is used to measure the magnetic field (acting as a magnetometer), the change in the magnetic field due to the metallic object can be detected.

The first industrial metal detectors were developed in the 1960s and were used extensively for mineral prospecting and other industrial applications. Uses include detecting land mines, the detection of weapons such as knives and guns (especially in airport security), geophysical prospecting, archaeology and treasure hunting. Metal detectors are also used to detect foreign bodies in food, and in the construction industry to detect steel reinforcing bars in concrete and pipes and wires buried in walls and floors.

Fig 7.22 : Metal Detector

7.7 LCD DISPLAY

LCD Background:

One of the most common devices attached to a micro controller is an LCD display. Some of the most common LCD's connected to the many microcontrollers are 16x2 and 20x2

displays. This means 16 characters per line by 2 lines and 20 characters per line by 2 lines, respectively.

Basic 16x 2 Characters LCD

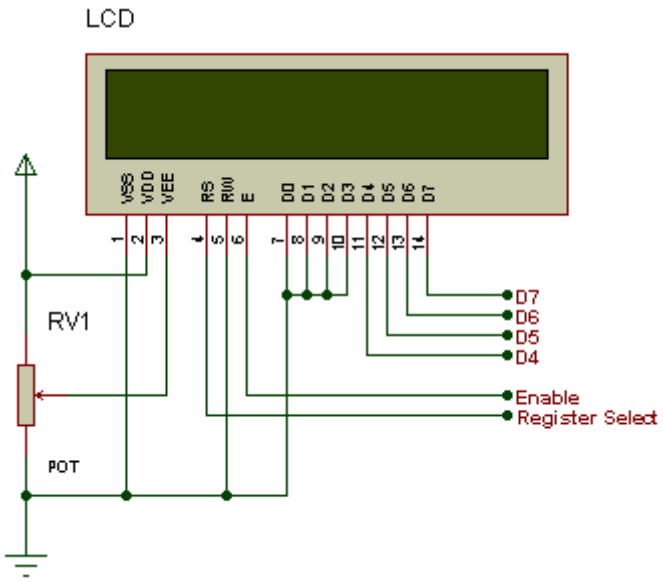


Fig 7.23 : LCD Pin diagram

Pin description:

Pin No.	**Name**	**Description**
Pin no. 1	**VSS**	Power supply (GND)
Pin no. 2	**VCC**	Power supply (+5V)
Pin no. 3	**VEE**	Contrast adjust
Pin no. 4	**RS**	0 = Instruction input 1 = Data input
Pin no. 5	**R/W**	0 = Write to LCD module 1 = Read from LCD module
Pin no. 6	**EN**	Enable signal
Pin no. 7	**D0**	Data bus line 0 (LSB)

Pin no. 8	**D1**	Data bus line 1
Pin no. 9	**D2**	Data bus line 2
Pin no. 10	**D3**	Data bus line 3
Pin no. 11	**D4**	Data bus line 4
Pin no. 12	**D5**	Data bus line 5
Pin no. 13	**D6**	Data bus line 6
Pin no. 14	**D7**	Data bus line 7 (MSB)

Table 7.1: Character LCD pins with Microcontroller

The LCD requires 3 control lines as well as either 4 or 8 I/O lines for the data bus. The user may select whether the LCD is to operate with a 4-bit data bus or an 8-bit data bus. If a 4-bit data bus is used the LCD will require a total of 7 data lines (3 control lines plus the 4 lines for the data bus). If an 8-bit data bus is used the LCD will require a total of 11 data lines (3 control lines plus the 8 lines for the data bus).

The three control lines are referred to as **EN**, **RS**, and **RW**.

The **EN** line is called "Enable." This control line is used to tell the LCD that we are sending it data. To send data to the LCD, our program should make sure this line is low (0) and then set the other two control lines and/or put data on the data bus. When the other lines are completely ready, bring **EN** high (1) and wait for the minimum amount of time required by the LCD datasheet (this varies from LCD to LCD), and end by bringing it low (0) again.

The **RS** line is the "Register Select" line. When RS is low (0), the data is to be treated as a command or special instruction (such as clear screen, position cursor, etc.). When RS is high (1), the data being sent is text data which should be displayed on the screen. For example, to display the letter "T" on the screen we would set RS high.

The **RW** line is the "Read/Write" control line. When RW is low (0), the information on the data bus is being written to the LCD. When RW is high (1), the program is effectively querying (or reading) the LCD. Only one instruction ("Get LCD status") is a read command. All others are write commands--so RW will almost always be low.

Finally, the data bus consists of 4 or 8 lines (depending on the mode of operation selected by the user). In the case of an 8-bit data bus, the lines are referred to as DB0, DB1, DB2, DB3, DB4, DB5, DB6, and DB7.

Circuit Description:

Above is the quite simple schematic. The LCD panel's Enable and Register Select is connected to the Control Port. The Control Port is an open collector / open drain output. While most Parallel Ports have internal pull-up resistors, there is a few which don't. Therefore by incorporating the two 10K external pull up resistors, the circuit is more portable for a wider range of computers, some of which may have no internal pull up resistors.

We make no effort to place the Data bus into reverse direction. Therefore we hard wire the R/W line of the LCD panel, into write mode. This will cause no bus conflicts on the data lines. As a result we cannot read back the LCD's internal Busy Flag which tells us if the LCD has accepted and finished processing the last instruction. This problem is overcome by inserting known delays into our program.

The 10k Potentiometer controls the contrast of the LCD panel. Nothing fancy here. As with all the examples, I've left the power supply out. We can use a bench power supply set to 5v or use an onboard +5 regulator. Remember a few de-coupling capacitors, especially if we have trouble with the circuit working properly.

SETB RW

Handling the EN control line:

As we mentioned above, the EN line is used to tell the LCD that we are ready for it to execute an instruction that we've prepared on the data bus and on the other control lines. Note that the EN line must be raised/ lowered before/after each instruction sent to the LCD regardless of whether that instruction is read or write text or instruction. In short, we must always manipulate EN when communicating with the LCD. EN is the LCD's way of knowing that we are talking to it. If we don't raise/lower EN, the LCD doesn't know we're talking to it on the other lines.

Thus, before we interact in any way with the LCD we will always bring the **EN** line low with the following instruction:

CLR EN

And once we've finished setting up our instruction with the other control lines and data bus lines, we'll always bring this line high:

SETB EN

The line must be left high for the amount of time required by the LCD as specified in its datasheet. This is normally on the order of about 250 nanoseconds, but checks the datasheet. In the case of a typical microcontroller running at 12 MHz, an instruction requires 1.08 microseconds to execute so the EN line can be brought low the very next instruction. However, faster microcontrollers (such as the DS89C420 which executes an instruction in 90 nanoseconds given an 11.0592 MHz crystal) will require a number of NOPs to create a delay while EN is held high. The number of NOPs that must be inserted depends on the microcontroller we are using and the crystal we have selected.

The instruction is executed by the LCD at the moment the EN line is brought low with a final CLR EN instruction.

Checking the busy status of the LCD:

As previously mentioned, it takes a certain amount of time for each instruction to be executed by the LCD. The delay varies depending on the frequency of the crystal attached to the oscillator input of the LCD as well as the instruction which is being executed.

While it is possible to write code that waits for a specific amount of time to allow the LCD to execute instructions, this method of "waiting" is not very flexible. If the crystal frequency is changed, the software will need to be modified. A more robust method of programming is to use the "Get LCD Status" command to determine whether the LCD is still busy executing the last instruction received.

The "Get LCD Status" command will return to us two tidbits of information; the information that is useful to us right now is found in DB7. In summary, when we issue the "Get LCD Status" command the LCD will immediately raise DB7 if it's still busy executing a command or lower DB7 to indicate that the LCD is no longer occupied. Thus our program can query the LCD until DB7 goes low, indicating the LCD is no longer busy. At that point we are free to continue and send the next command.

Applications:

- Medical equipment
- Electronic test equipment
- Industrial machinery Interface
- Serial terminal
- Advertising system
- EPOS
- Restaurant ordering systems
- Gaming box
- Security systems
- R&D Test units
- Climatizing units
- PLC Interface
- Simulators
- Environmental monitoring
- Lab development
- Student projects
- Home automation
- PC external display
- HMI operator interface.

7.8 Buzzer

Fig 7.24 : Buzzer

Basically, the sound source of a piezoelectric sound component is a piezoelectric diaphragm. A piezoelectric diaphragm consists of a piezoelectric ceramic plate which has electrodes on both sides and a metal plate (brass or stainless steel, etc.). A piezoelectric ceramic plate is attached to a metal plate with adhesives. Applying D.C. voltage between electrodes of a piezoelectric diaphragm causes mechanical distortion due to the piezoelectric effect. For a misshaped piezoelectric element, the distortion of the piezoelectric element expands in a radial direction. And the piezoelectric diaphragm bends toward the direction. The metal plate bonded to the piezoelectric element does not expand. Conversely, when the piezoelectric element shrinks, the piezoelectric diaphragm bends in the direction Thus, when AC voltage is applied across electrodes, the bending is repeated, producing sound waves in the air.

To interface a buzzer the standard transistor interfacing circuit is used. Note that if a different power supply is used for the buzzer, the 0V rails of each power supply must be connected to provide a common reference.

If a battery is used as the power supply, it is worth remembering that piezo sounders draw much less current than buzzers. Buzzers also just have one ‘tone’, whereas a piezo sounder is able to create sounds of many different tones.

To switch on buzzer -high 1

To switch off buzzer -low 1

Notice (Handling) In Using Self Drive Method

1) When the piezoelectric buzzer is set to produce intermittent sounds, sound may be heard continuously even when the self drive circuit is turned ON / OFF at the "X" point shown in Fig. 9. This is because of the failure of turning off the feedback voltage.

2) Build a circuit of the piezoelectric sounder exactly as per the recommended circuit shown in the catalog of the transistor and circuit constants are designed to ensure stable oscillation of the piezoelectric sounder.

3) Design switching which ensures direct power switching.

4) The self drive circuit is already contained in the piezoelectric buzzer. So there is no need to prepare another circuit to drive the piezoelectric buzzer.

5) Rated voltage (3.0 to 20Vdc) must be maintained. Products which can operate with voltage higher than 20Vdc are also available.

6) Do not place resistors in series with the power source, as this may cause abnormal oscillation. If a resistor is essential to adjust sound pressure, place a capacitor (about 1μF) in parallel with the piezo buzzer.

7) Do not close the sound emitting hole on the front side of casing.

8) Carefully install the piezo buzzer so that no obstacle is placed within 15mm from the sound release hole on the front side of the casing.

CHAPTER-8

SIMULATION TOOLS

This project is implemented using following software's:

- Express PCB – for designing circuit
- Arduino IDE compiler - for compilation part
- Proteus 7 (Embedded C) – for simulation part

8.1 Express PCB:

Breadboards are great for prototyping equipment as it allows great flexibility to modify a design when needed; however the final product of a project, ideally should have a neat PCB, few cables, and survive a shake test. Not only is a proper PCB neater but it is also more durable as there are no cables which can yank loose.

Express PCB is a software tool to design PCBs specifically for manufacture by the company Express PCB (no other PCB maker accepts Express PCB files). It is very easy to use, but it does have several limitations.

It can be likened to more of a toy then a professional CAD program.

It has a poor part library (which we can work around)

It cannot import or export files in different formats

It cannot be used to make prepare boards for DIY production

Express PCB has been used to design many PCBs (some layered and with surface-mount parts. Print out PCB patterns and use the toner transfer method with an Etch Resistant Pen to make boards. However, Express PCB does not have a nice print layout. Here is the procedure to design in Express PCB and clean up the patterns so they print nicely.

8.1.1 Preparing Express PCB for First Use:

Express PCB comes with a less then exciting list of parts. So before any project is started head over to Audio logical and grab the additional parts by morsel, ppl, and tangent,

and extract them into your Express PCB directory. At this point start the program and get ready to setup the workspace to suit your style.

Click View -> Options. In this menu, setup the units for “mm” or “in” depending on how you think, and click “see through the top copper layer” at the bottom. The standard color scheme of red and green is generally used but it is not as pleasing as red and blue.

8.1.2 The Interface:

When a project is first started you will be greeted with a yellow outline. This yellow outline is the dimension of the PCB. Typically after positioning of parts and traces, move them to their final position and then crop the PCB to the correct size. However, in designing a board with a certain size constraint, crop the PCB to the correct size before starting.

Fig: 4.1 show the toolbar in which the each button has the following functions:

Fig 8.1 : Tool bar necessary for the interface

- The select tool: It is fairly obvious what this does. It allows you to move and manipulate parts. When this tool is selected the top toolbar will show buttons to move traces to the top / bottom copper layer, and rotate buttons.
- The zoom to selection tool: does just that.
- The place pad: button allows you to place small soldier pads which are useful for board connections or if a part is not in the part library but the part dimensions are available. When this tool is selected the top toolbar will give you a large selection of round holes, square holes and surface mount pads.
- The place component: tool allows you to select a component from the top toolbar and then by clicking in the workspace places that component in the orientation chosen using the buttons next to the component list. The components can always be rotated afterwards with the select tool if the orientation is wrong.
- The place trace: tool allows you to place a solid trace on the board of varying thicknesses. The top toolbar allows you to select the top or bottom layer to place the trace on.

- The Insert Corner in trace: button does exactly what it says. When this tool is selected, clicking on a trace will insert a corner which can be moved to route around components and other traces.
- The remove a trace button is not very important since the delete key will achieve the same result.

8.1.3 Design Considerations:

Before starting a project there are several ways to design a PCB and one must be chosen to suit the project's needs.

Single sided, or double sided?

When making a PCB you have the option of making a single sided board, or a double sided board. Single sided boards are cheaper to produce and easier to etch, but much harder to design for large projects. If a lot of parts are being used in a small space it may be difficult to make a single sided board without jumpering over traces with a cable. While there's technically nothing wrong with this, it should be avoided if the signal travelling over the traces is sensitive (e.g. audio signals).

A double sided board is more expensive to produce professionally, more difficult to etch on a DIY board, but makes the layout of components a lot smaller and easier. It should be noted that if a trace is running on the top layer, check with the components to make sure you can get to its pins with a soldering iron. Large capacitors, relays, and similar parts which don't have axial leads can NOT have traces on top unless boards are plated professionally.

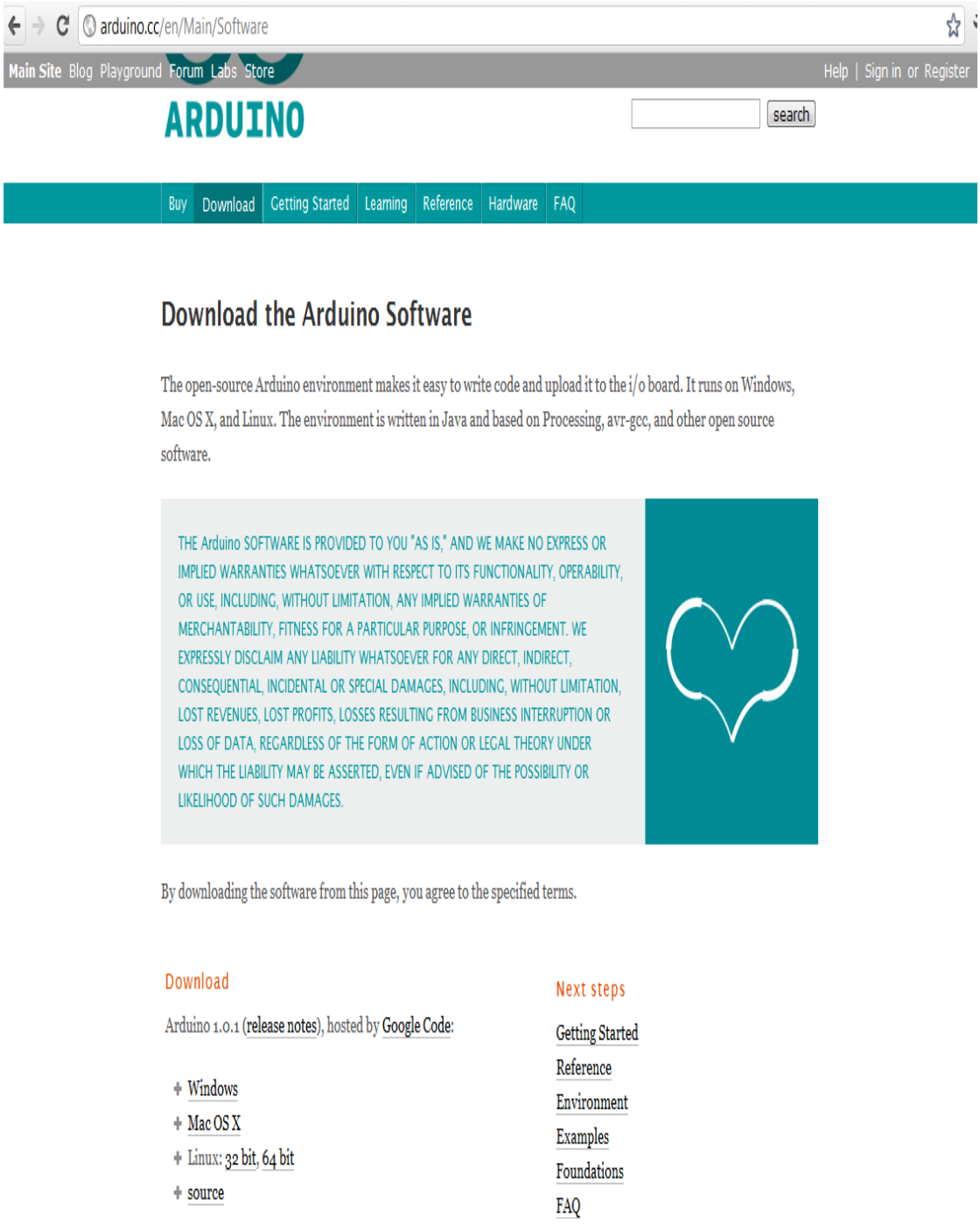
Ground-plane or other special purposes for one side

When using a double sided board you must consider which traces should be on what side of the board. Generally, put power traces on the top of the board, jumping only to the bottom if a part cannot be soldiered onto the top plane (like a relay), and vice- versa.

Some projects like power supplies or amps can benefit from having a solid plane to use for ground. In power supplies this can reduce noise, and in amps it minimizes the distance between parts and their ground connections, and keeps the ground signal as simple as possible. However, care must be taken with stubborn chips such as the TPA6120 amplifier

from TI. The TPA6120 datasheet specifies not to run a ground plane under the pins or signal traces of this chip as the capacitance generated could effect performance negatively.

Arduino compiling



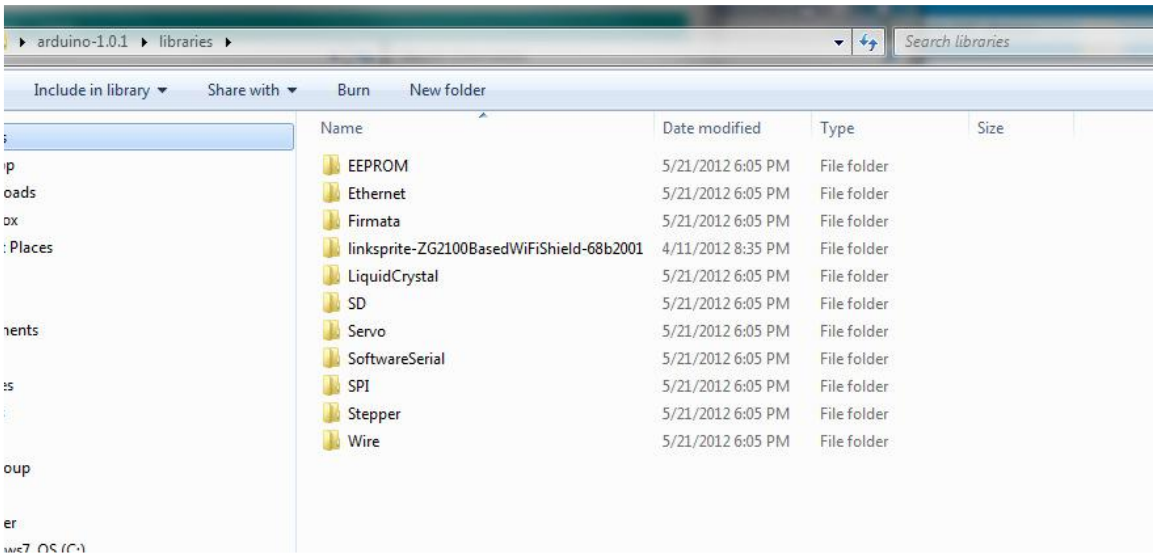
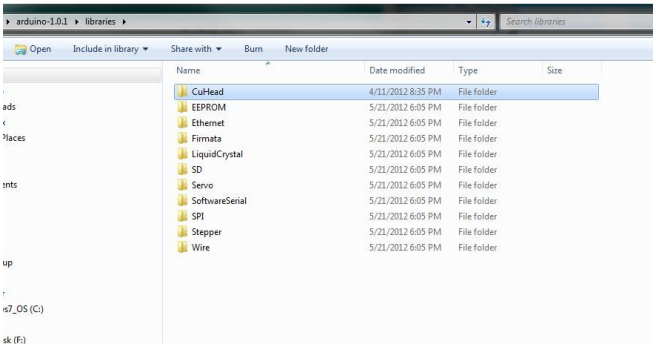
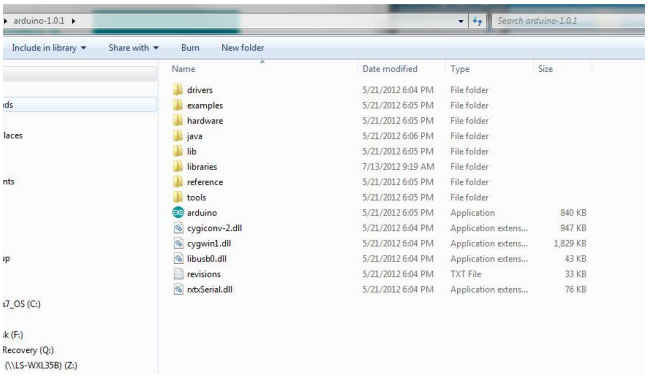


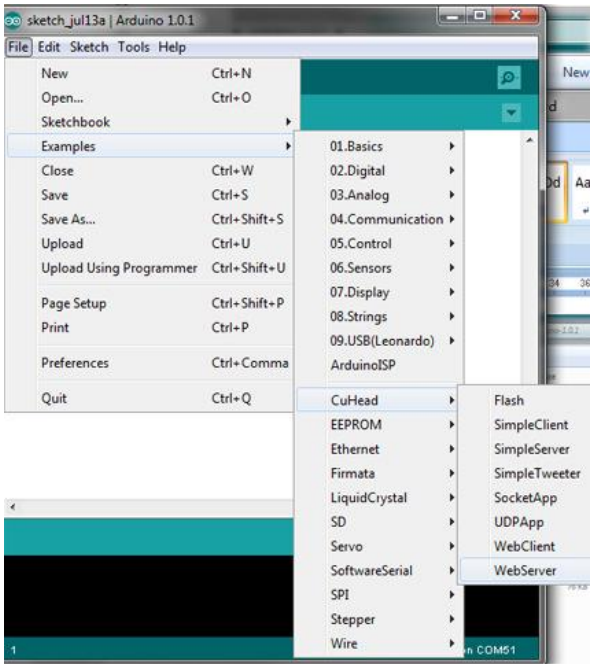
Fig 8.2 : In next step download library



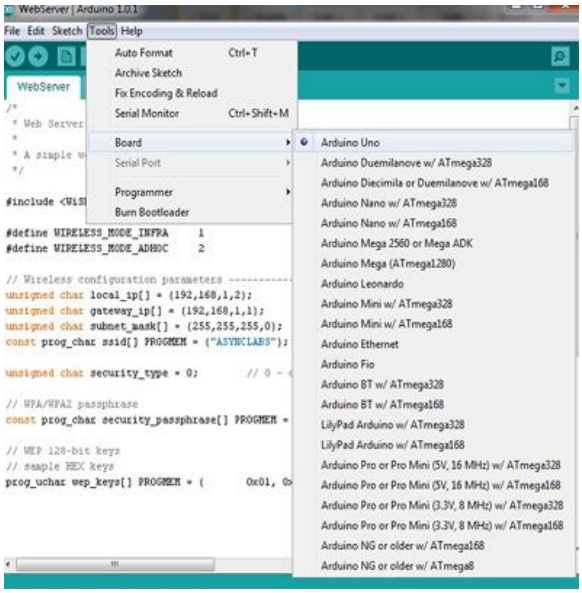
As Arduino doesn't recognize the directory name, please rename it



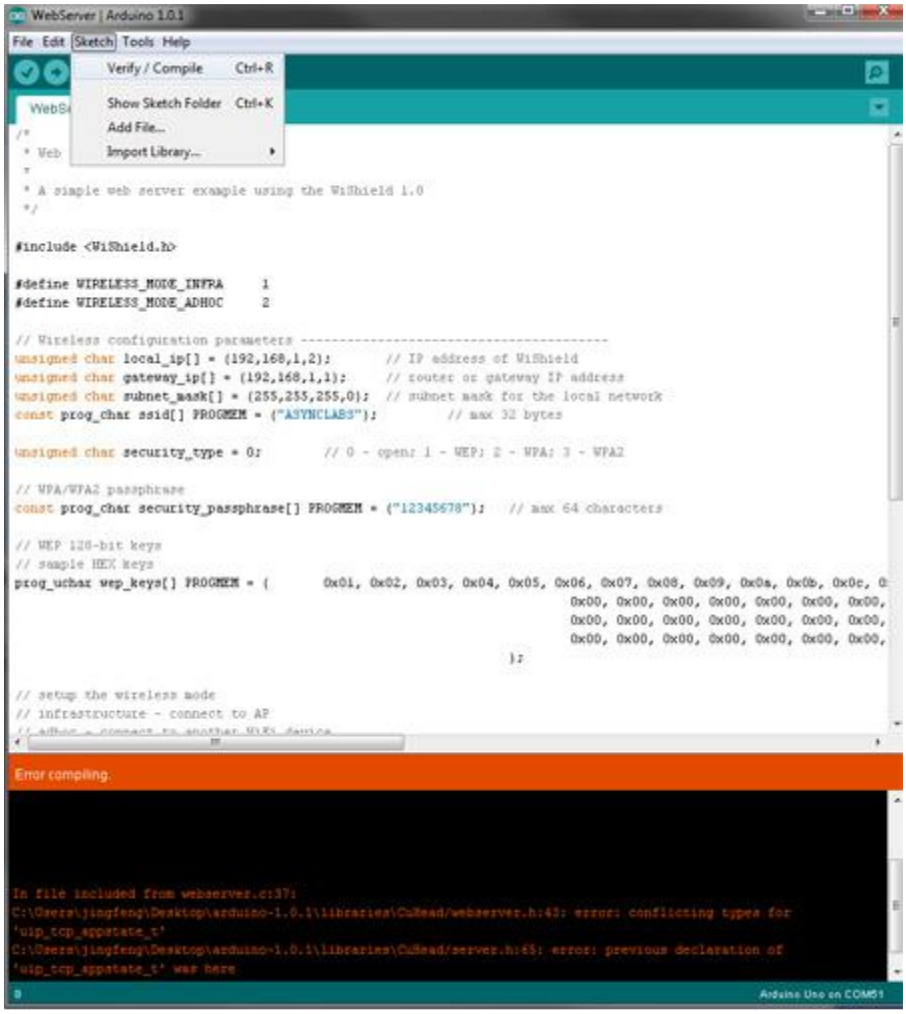
Launch Arduino by double click "arduino" below



One example



Select the target board as “Arduino Uno”:



Click Sketch-> Verify/Compile:

CHAPTER 9

SOURCE CODE

```
#include <LiquidCrystal.h>
#include <stdio.h>
#include "esp_camera.h"
#include <WiFi.h>
#include "esp_timer.h"
#include "img_converters.h"
#include "Arduino.h"
#include "fb_gfx.h"
#include "soc/soc.h"             // disable brownout problems
#include "soc/rtc_cntl_reg.h"    // disable brownout problems
#include "esp_http_server.h"
// Replace with your network credentials
const char* ssid = "iotserver";
const char* password = "iotserver123";
LiquidCrystal lcd(13, 12, 14, 27, 26, 25);
#include <Wire.h>
#include "dht.h"
int tempc=0,humc=0;
char gchr='x';
char modes='x';
int sti=0;
```

```
String inputString = "";         // a string to hold incoming data
boolean stringComplete = false;  // whether the string is complete
int m1a       = 21;
int m1b       = 19;
int m2a       = 18;
int m2b       = 5;
int m3a       = 16;
int m3b       = 17;
int m4a       = 4;
int m4b       = 2;
int buzzer    = 23;
int metal     = 22;
void okcheck0()
{
  unsigned char rcr;
  do{
      rcr = Serial.read();
    }while(rcr != 'K');
}
void setup()
{
   Serial.begin(9600);//serialEvent();
   pinMode(m1a, OUTPUT);pinMode(m1b, OUTPUT);
   pinMode(m2a, OUTPUT);pinMode(m2b, OUTPUT);
   pinMode(m3a, OUTPUT);pinMode(m3b, OUTPUT);
```

```
 pinMode(m4a, OUTPUT);pinMode(m4b, OUTPUT);
 pinMode(buzzer, OUTPUT);
 pinMode(metal, INPUT);
 digitalWrite(m1a, LOW);digitalWrite(m1b, LOW);
 digitalWrite(m2a, LOW);digitalWrite(m2b, LOW);
 digitalWrite(m3a, LOW);digitalWrite(m3b, LOW);
 digitalWrite(m4a, LOW);digitalWrite(m4b, LOW);
 digitalWrite(buzzer, HIGH);
lcd.begin(16, 2);
lcd.print("  IOT Landmine");
lcd.setCursor(0,1);
lcd.print("     Robot      ");
  delay(15000);
lcd.clear();
lcd.setCursor(0, 0);
lcd.print("Metal:");//6,0
lcd.setCursor(0, 1);
lcd.print("Fire:"); //5,1
while(Serial.available() > 0)
    {
     char t = Serial.read();
    }
Serial.flush();
delay(2000);
// serialEvent();
```

```
}
void loop()
{
   Serial.write('*');
   if(digitalRead(metal) == HIGH)
     {
      lcd.setCursor(6,0);lcd.print("Det");
       Serial.print("Metal_Detected");
       digitalWrite(buzzer, LOW);
       goto mn1;
     }
   if(digitalRead(metal) == LOW)
     {
      lcd.setCursor(6,0);lcd.print("    ");
            Serial.print("Metal_Not_Detected");
      digitalWrite(buzzer, HIGH);
       goto mn1;
     }
 mn1: Serial.write('#');
    for(int pti=0;pti<400;pti++)
       {
        while(Serial.available())
            {
             char inChar = (char)Serial.read();
             if(inChar == '*')
```

```
      {sti=1;
      }
     if(sti == 1)
      {
       inputString += inChar;
      }
     if(inChar == '#')
      {sti=0;
       stringComplete = true;
      }
    }
if(stringComplete)
  {
    if(inputString[1] == 'f')
     {
      digitalWrite(m1a,HIGH);digitalWrite(m1b,LOW);
      digitalWrite(m2a,HIGH);digitalWrite(m2b,LOW);
     }
    if(inputString[1] == 'b')
     {
      digitalWrite(m1a,LOW);digitalWrite(m1b,HIGH);
      digitalWrite(m2a,LOW);digitalWrite(m2b,HIGH);
     }
    if(inputString[1] == 'l')
     {
```

```
    digitalWrite(m1a,LOW);digitalWrite(m1b,HIGH);
    digitalWrite(m2a,HIGH);digitalWrite(m2b,LOW);
  }
if(inputString[1] == 'r')
  {
    digitalWrite(m1a,HIGH);digitalWrite(m1b,LOW);
    digitalWrite(m2a,LOW);digitalWrite(m2b,HIGH);
  }
if(inputString[1] == 's')
  {
    digitalWrite(m1a,LOW);digitalWrite(m1b,LOW);
    digitalWrite(m2a,LOW);digitalWrite(m2b,LOW);
  }
if(inputString[1] == '1')
  {
    digitalWrite(m3a,HIGH);digitalWrite(m3b,LOW);
  }
if(inputString[1] == '2')
  {
    digitalWrite(m3a,LOW);digitalWrite(m3b,HIGH);
  }
if(inputString[1] == '3')
  {
    digitalWrite(m3a,LOW);digitalWrite(m3b,LOW);
  }
```

```
        if(inputString[1] == '4')
          {
            digitalWrite(m4a,HIGH);digitalWrite(m4b,LOW);
          }
        if(inputString[1] == '5')
          {
            digitalWrite(m4a,LOW);digitalWrite(m4b,HIGH);
          }
        if(inputString[1] == '6')
          {
            digitalWrite(m4a,LOW);digitalWrite(m4b,LOW);
          }
        inputString = "";
        stringComplete = false;
      }
    // sensor_fun();
     delay(10);
    }
}
```

CHAPTER-10

RESULTS

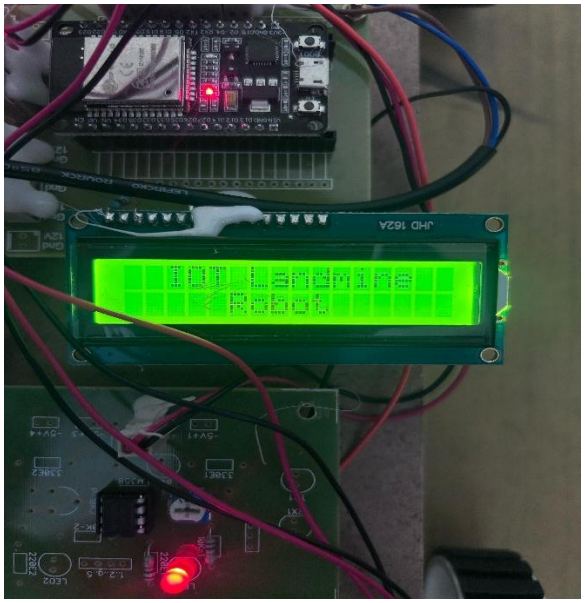


Fig 10.1 : LED screen

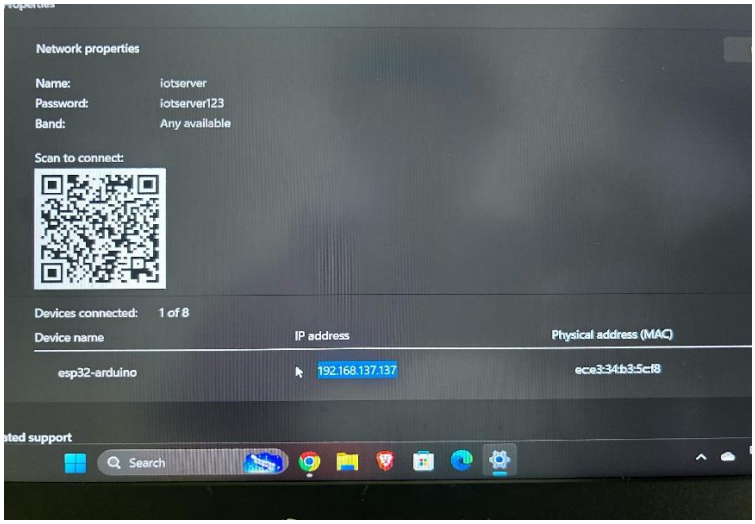


Fig 10.2 : setting up mobile hotspot in laptop

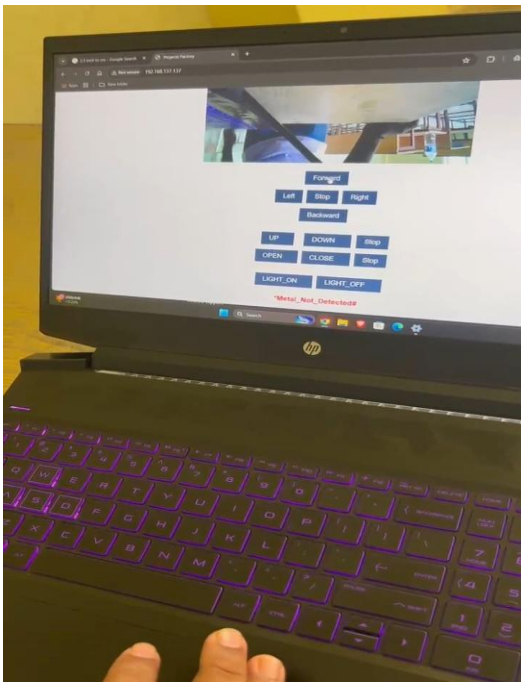


Fig 10.3 : User Interface

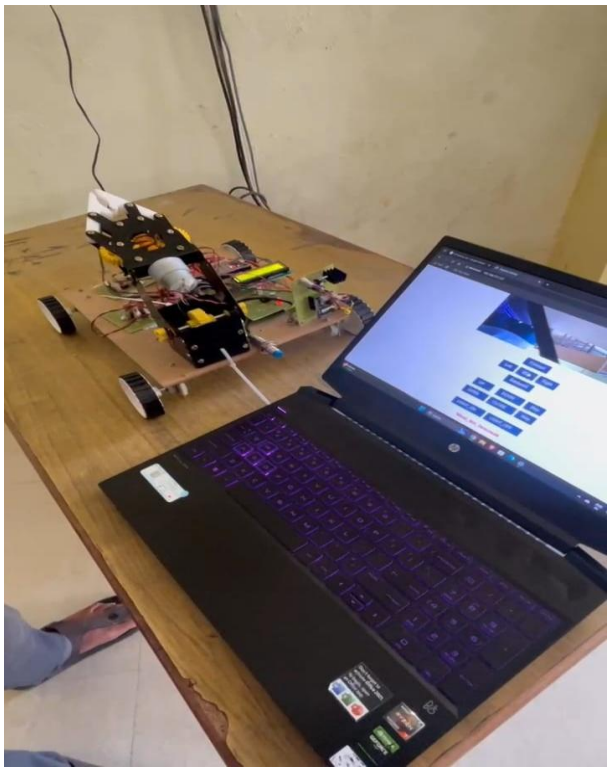


Fig 10.4 : Robot and user Interface

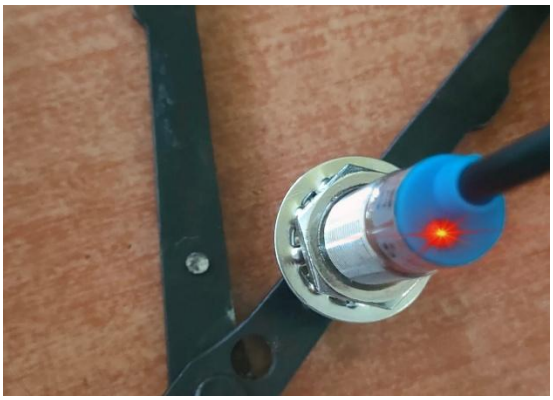


Fig 10.5 : detecting metal through metal detector

Fig 10.6 : Displaying detected on Led screen

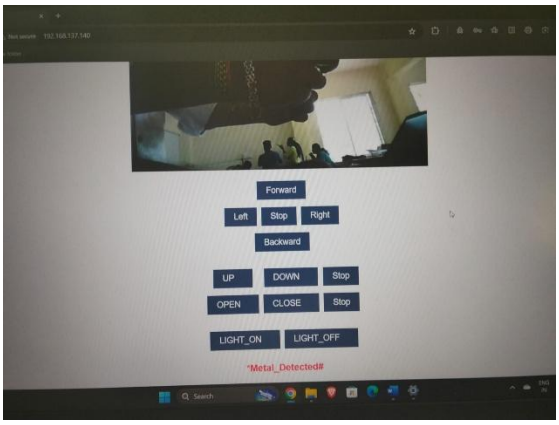


Fig 10.7 : Showing detected in Laptop

CHAPTER 11

CONCLUSION

The development of the remote-controlled landmine detection robot with an integrated robotic arm presents a significant advancement in ensuring human safety during landmine clearance operations. By combining metal detection, robotic mobility, remote communication, and manipulation capabilities, this system offers a reliable and efficient solution for identifying and marking potentially hazardous areas without exposing personnel to danger.

The use of an Arduino microcontroller enables seamless coordination between various modules such as the metal detector, robotic arm, buzzer alert system, and IoT-based communication. This enhances real-time control, feedback, and monitoring, making the system both user-friendly and technically robust. The addition of the robotic arm further improves functionality by enabling the robot to mark or interact with detected sites, thus increasing precision and usability in field operations.

Overall, this project demonstrates how embedded systems, robotics, and wireless technology can be effectively integrated to address real-world problems. With further enhancements like GPS tracking, autonomous navigation, and advanced sensor integration, this system has the potential to evolve into a fully autonomous mine detection and mapping solution that can be deployed in real demining missions across the globe.

Future scope

While the current prototype effectively demonstrates basic landmine detection and remote navigation, there is significant potential for enhancement and expansion. Future developments may include:

1. **GPS and Mapping Capabilities:**
    - Incorporating GPS can help in tracking the robot's exact location and mapping detected landmine positions for later demining operations.
2. **Autonomous Navigation:**

- By integrating ultrasonic or LiDAR sensors along with AI algorithms, the robot could be upgraded to autonomously navigate and detect landmines without manual control.

3. **Machine Learning for Object Classification:**
    - Implementing machine learning models could help distinguish between actual landmines and harmless metallic debris, reducing false positives.
4. **Advanced Communication Systems:**
    - Replacing short-range modules (like Bluetooth) with long-range alternatives (such as LoRa or GSM) would enable operation over wider areas.
5. **Multi-Sensor Detection:**
    - Incorporating additional sensors like Ground Penetrating Radar (GPR) or Infrared (IR) could improve detection accuracy and allow for non-metallic landmine detection.
6. **Improved Terrain Handling:**
    - Designing a more rugged, all-terrain chassis would allow the robot to operate in harsher environments such as muddy fields, rocky paths, or dense vegetation.
7. **Swarm Robotics:**
    - Developing a network of such robots working in coordination could drastically reduce the time required to survey and clear large areas.

REFERENCE:

Acute Manandhar, Peter A. Torrione, Leslie M. Collins and Kenneth D. Morton, "Multiple-Instance Hidden Markov Model for GPR-Based Landmine Detection," IEEE transactions on Geosciences and remote sensing, vol. 53, no. 4, pp. 1737-1745, April 2015. 2)Jaradat, M.A, "Autonomous navigation robot for landmine detection applications," IEEE transactions on Mechatronics and its Applications, vol.16, no. 3, pp. 1-5, April 2012.3)"Design and Implementation of Landmine Robot" Wade Ghribi, Ahmed Said Badawy, Mohammed Rahmathullah, Suresh Babu Changalasetty. IJEIT Volume 2, Issue 11, May 2013. 4)Ghribi, W., Badawy, A.S., Rahmathullah, M. and Changalasetty, S.B." Design and implementation of landmine robot". Int. J. Engg. Innov. Technol. 2(11): 250-256, March 2013. 5)P. Gonzalez de Santos, E. Garcia, J. Estremera and M.A. "Using walking robots for landmine detection and location". Armada Industrial Automation Institute-CSIC . Camp Real, Km. 0,200- La Poveda 28500 Arganda del Rey, Madrid, Spain, January 2014. 6)Ilaria Bottigliero. 120 Million Landmines Deployed Worldwide: Fact or Fiction. Pen and Sword Books Ltd, Barnsley, South Yorkshire, UK, June 2014. 7)MacDonald J., Lockwood J.R., Mc Fee J.E., Altshuler T., Broach J. T., L. Carin,Harmon R.S., Rappaport C., Scott W.R., and Weaver R. Alternatives for landmine detection. Technical report, RAND (http://www.rand.org/ publications/ MR/ MR 1608/ MR 1608. appg.pdf), February2013. 8)Jaradat M A, Bani Salim M N and Awad F H, "Autonomous Navigation Robot for Landmine Detection Applications", 8th International Symposium on Mechatronics and its Applications (ISMA), April 2012. L. Robledo, M. Carrasco and D. Mery," A survey of land mine detection technology" International Journal of Remote Sensing Vol. 30, No. 9, 10 May 2009, 2399–2410 [2] Jebasingh Kirubakaran.S.J, Anish kumar jha, Dheeraj kumar, Sadambi Poorna chandram Prakash, "Mine Detecting Robot with Multi Sensors Controlled Using HC-12 Module" International Journal of Engineering & Technology [3] Bharath J, "Automatic Land Mine Detection Robot Using Microcontroller", International Journal of Advance Engineering and Research Development Volume 4, Issue 3, March-2017 [4] Zhenjun He, Jiang Zhang, Peng Xu, Jiaheng Qin and Yunkai Zhu, "Mine Detecting Robot Based on Wireless Communication with Multi-sensor". [5] Jaradat M A, Bani Salim M N and Awad F H (2012), "Autonomous Navigation Robot for Landmine Detection Applications". [6] Kuo-Lan Su, Hsu-Shan Su, Sheng-Wen Shiao and JrHung Guo (2011), "Motion Planning for a Landmine Detection Robot", Artificial Life and Robotics. [7] Kaur Gurpreet, "Multi algorithm-based

Landmine Detection using Ground Penetration Radar", IEEE, 2016. [8] Kishan Malaviya, et.al, "Autonomous Landmine Detecting and Mapping Robot", IJIRCCE, Vol. 3, Issue 2, 2015.