A Project Report on

“IMPLEMENTATION OF A SYSTEM TO CONTROL THE POSITION OF AN ANTENNA USING IOT”

Submitted in Partial Fulfillment of the Requirement for the Award of the Degree of

BACHELOR OF TECHNOLOGY

IN

ELECTRONICS AND COMMUNICATION ENGINEERING

Submitted By

S. DINESH	P.SUNEETHA
(19HM1A0439)	(19HM1A0435)
S. CHINNA DASTAGIRI	K. PRUDHVI
(19HM1A0440)	(19HM1A0423)

Under the Esteemed Guidance of

Mr. K. MAHAMMAD HANEEF M.TECH., (Ph.D.).,

Assistant Professor,

Dept. of Electronics & Communication Engineering

DEPARTMENT OF ELECTRONICS & COMMUNICATIONS ENGINEERING

ANNAMACHARYA INSTITUTE OF TECHNOLOGY AND SCIENCES

Affiliated to J.N.T.U.A., Anantapuramu, Approved by A.I.C.T.E., New Delhi

Utukur(P), C.K. Dinne (V&M), Kadapa-516003.

(2019-2023)

ANNAMACHARYA INSTITUTE OF TECHNOLOGY AND SCIENCES

Affiliated to J.N.T.U.A., Anantapuramu, Approved by A.I.C.T.E., New Delhi

Utukur (P), C.K. Dinne, (V&M), Kadapa-516003.

DEPARTMENT OF ELECTRONICS & COMMUNICATION ENGINEERING

CERTIFICATE

This is to certify that the project is titled "**IMPLEMENTATION OF A SYSTEM TO CONTROL THE POSITION OF AN ANTENNA USING IOT**" is bonafide work done by

S. DINESH	(19HM1A0439)
P. SUNEETHA	(19HM1A0435)
S. CHINNA DASTAGIRI	(19HM1A0440)
K. PRUDHVI	(19HM1A0423)

In the Partial fulfilment of the requirement for the award of the degree of **BACHELOR OF TECHNOLOGY** in **ELECTRONICS & COMMUNICATIONS ENGINEERING** in **ANNAMACHARYA INSTITUTE OF TECHNOLOGY AND SCIENCES, KADAPA** during the academic year 2019-2023.The results of this work have not been submitted to any other university or institutions for the award of any degree.

Project Guide

Mr. K. MAHAMMAD HANEEF M.Tech., (Ph.D.).,

Assistant Professor,

Department of ECE,

A.I.T.S., Kadapa.

Head of the Department

Dr. P.V.S. MURALI KRISHNA M.E., Ph.D.,

Professor, Dept. of ECE,

A.I.T.S., Kadapa.

Viva-voce exam held on dated:______________________.

Signature of the External Examiner.

ACKNOWLEDGEMENT

An endeavor of a long period can be successful only with the advice of many well-wishers. We take this opportunity to express our deep gratitude and appreciation to all those who encouraged us for successful completion of this project.

We are very much indebted to our guide **Mr**. **K. MAHAMMAD HANEEF**, **M. Tech**., **(Ph.D).,** Assistant Professor, Department of E.C.E, Annamacharya Institute of Technology and Sciences, Kadapa for his valuable guidance and suggestions till the end of project.

We are extending our gratefulness to our Project **Coordinators Ms. J. BEENA SINDHURI**, **M. Tech.,** and **Mr. S. JAFFAR ALI, M. Tech.,** Assistant Professors of ECE for their feedback and support throughout the project.

Our heartful thanks to **Dr. P.V.S. MURALI KRISHNA, M.E., Ph.D.,** Professor& Head, Department of Electronics & Communication Engineering, AITS, Kadapa.

We wish to express our sincere gratitude to **Dr. A. SUDHAKARA REDDY, Principal**, AITS, Kadapa, for providing the encouragement for doing this project.

We are very much thankful to **Dr. C. GANGI REDDY**, Honorable Secretary of Annamacharya Educational Trust, for providing the labs and other facilities for carrying out this project work.

Finally, we would like to express our sincere thanks to faculty members of E.C.E department, lab technicians and friends and family, who helped us directly or indirectly for the successful completion of this project successfully.

PROJECT ASSOCIATES

S. DINESH	**(19HM1A0439)**
P. SUNEETHA	**(19HM1A0435)**
S. CHINNA DASTAGIRI	**(19HM1A0440)**
K. PRUDHVI	**(19HM1A0423)**

DECLARATION

We here by declare that project report entitled **“IMPLEMENTATION OF A SYSTEM TO CONTROL THE POSITION OF AN ANTENNA USING IOT”** being submitted by us for the award of degree of Bachelor of Technology in Electronics & Communication Engineering, to Jawaharlal Nehru Technological University Anantapuramu, and is a bonafide record of work done in Annamacharya Institute of Technology & Sciences, Kadapa and has not been submitted to any other courses or university for the award of any degree.

<u>PROJECT ASSOCIATES</u>

S. DINESH	**(19HM1A0439)**
P. SUNNETHA	**(19HM1A0435)**
S. CHINNA DASTAGIRI	**(19HM1A0440)**
K. PRUDHVI	**(19HM1A0423)**

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABULAR COLUMNS

ABSTRACT

Over the past few years, IOT has become one of the most important technologies of 21st century. Every single Internet of Things (IOT) device needs an antenna. Proper positioning of antennas is necessary for wireless communication.

The idea is to develop a system which will control the movement of the antenna in all directions. In order to overcome the difficulty of adjusting manually, this proposed system helps in adjusting the position of the antenna. Remote operation is achieved by any smart-phone with android OS. The different directions of the Antenna are attained by using two Dc motors, one moves in vertical and other in horizontal direction. The motor actions are controlled by the microcontroller. The Mobile application acts as a remote and is used to send the signal to the microcontroller through the internet. The project helps to control the antenna's position by the user's command.

Keywords: *Arduino ATMEGA328p, ESP8266 Wi-Fi module, DC motors, L293D Motor Driver and Mobile Application (Mobile Telnet).*

CHAPTER 1

1. INTRODUCTION

1.1 Introduction Embedded Systems: -

An embedded system is a system which is going to do a predefined specified task is the embedded system and is even defined as combination of both software and hardware. A general-purpose definition of embedded systems is that they are devices used to control, monitor or assist the operation of equipment, machinery or plant. "Embedded" reflects the fact that they are an integral part of the system. At the other extreme a general-purpose computer may be used to control the operation of a large complex processing plant, and its presence will be obvious.

All embedded systems are including computers or microprocessors. Some of these computers are however very simple systems as compared with a personal computer.

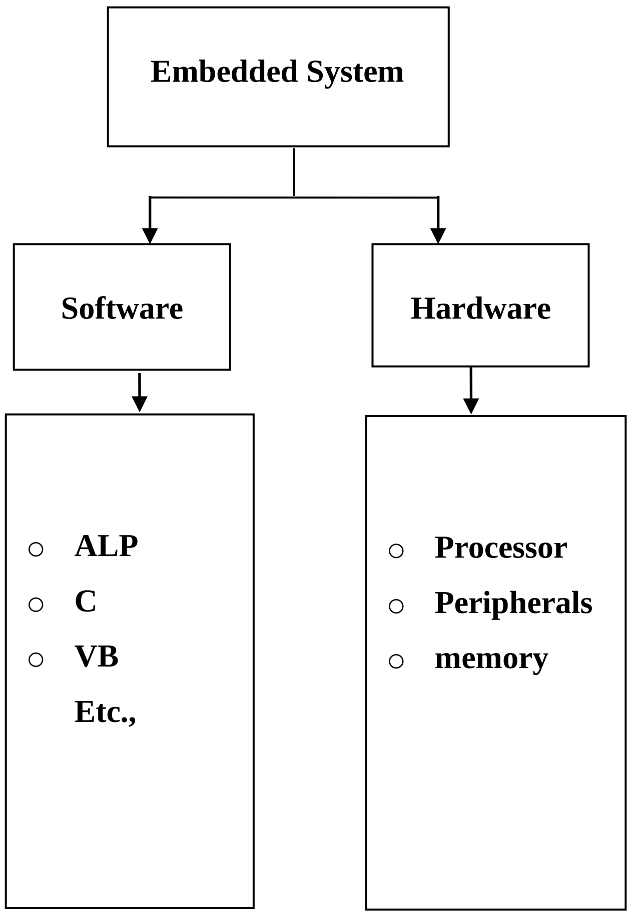


Fig 1.1: Block diagram of Embedded System

The very simplest embedded systems are capable of performing only a single function or set of functions to meet a single predetermined purpose. In more complex systems an application program that enables the embedded system to be used for a particular purpose in a specific application determines the functioning of the embedded system.

The ability to have programs means that the same embedded system can be used for a variety of different purposes. In some cases, a microprocessor may be designed in such a way that application software for a particular purpose can be added to the basic software in a second process, after which it is not possible to make further changes. The applications software on such processors is sometimes referred to as firmware.

The simplest devices consist of a single microprocessor (often called a "chip”), which may itself be packaged with other chips in a hybrid system or Application Specific Integrated Circuit (ASIC). Its input comes from a detector or sensor and its output goes to a switch or activator which (for example) may start or stop the operation of a machine or, by operating a valve, may control the flow of fuel to an engine. As the embedded system is the combination of both software and hardware. Software deals with the languages like ALP, C, and VB etc., and Hardware deals with Processors, Peripherals, and Memory.

Memory: Memory in an embedded system is a physical storage embedded device used to store two data types. The embedded system retrieves data from memory, processes it, and outputs data saved in memory. The data held might be intermediate data generated during the execution. The instructions or opcode executing the processor's function refer to the program information. When the program runs, the CPU retrieves and executes the instruction from memory**.**

Peripherals: Peripheral Devices in Embedded Systems Embedded systems communicate with the outside world via their peripheral.

Processor: Processors are the major part in embedded systems that take response from sensors in digital form and processing of this response to produce output in real-time processing environment is performed using processors. For an embedded system developer, it is essential to have the knowledge of both microprocessors and micro controllers. It is an IC which is used to perform some tasks.

1.2 Types of Embedded Systems

Embedded systems can be classified into different types based on performance, functional requirements and performance of the microcontroller.

Types of Embedded systems are given by

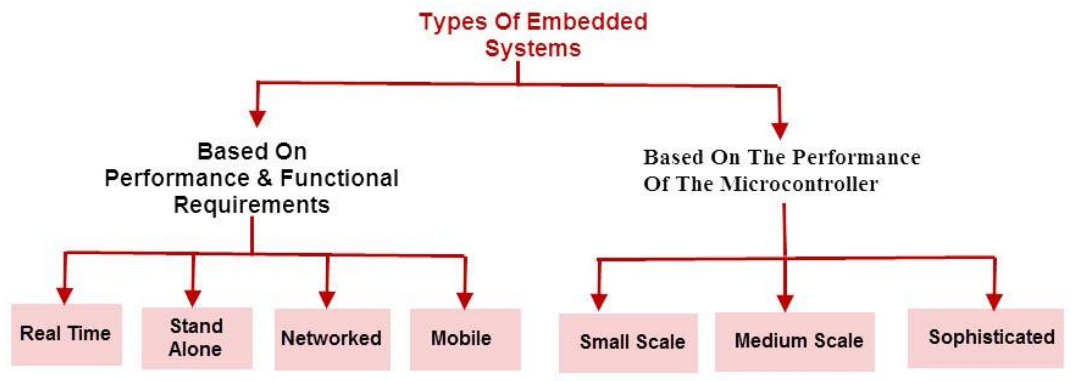

Types Of Embedded Systems
Based On Performance & Functional Requirements
Based On The Performance Of The Microcontroller
Real Time
Stand Alone
Networked
Mobile
Small Scale
Medium Scale
Sophisticated

Figure 1.2: Types of Embedded System

Embedded systems are classified into four categories based on their performance and functional requirements:

- Stand-alone embedded systems
- Real time embedded systems
- Networked embedded systems
- Mobile embedded systems

Embedded Systems are classified into three types based on the performance of the microcontroller such as

- Small scale embedded systems
- Medium scale embedded systems
- Sophisticated embedded systems

1.2.1 Stand Alone Embedded Systems:

Standalone embedded systems do not require a host system like a computer, it works by itself. It takes the input from the input ports either analog or digital and processes, calculates and converts the data and gives the resulting data through the connected device-Which either controls or drives and displays the connected devices. Examples for the stand-alone embedded systems are mp3 players, digital cameras, video game consoles, microwave ovens and temperature measurement systems.

1.2.2 Real Time Embedded Systems:

A real time embedded system is defined as; a system which gives required output in a particular time. These types of embedded systems follow the time deadlines for completion of a task. Real time embedded systems are classified into two types such as soft and hard real time systems.

1.2.3 Networked embedded systems:

These types of embedded systems are related to a network to access the resources. The connected network can be LAN, WAN or the internet. The connection can be any wired or wireless. This type of embedded system is the fastest growing area in embedded system applications.

The embedded web server is a type of system wherein all embedded devices are connected to a web server and accessed and controlled by a web browser. Example for the LAN networked embedded system is a home security system wherein all sensors are connected and run on the protocol TCP/IP.1.2.4 Mobile Embedded Systems:

Mobile embedded systems are used in portable embedded devices like cell phones, mobiles, digital cameras, mp3 players and personal digital assistants, etc. The basic limitation of these devices is the other resources and limitation of memory.

1.2.4 Small Scale Embedded Systems:

These types of embedded systems are designed with a single 8 or 16-bit microcontrollers that may even be activated by a battery.

For developing embedded software for small scale embedded systems, the main programming tools are an editor, assembler, cross assembler and integrated development environment (IDE).

1.2.5 Medium Scale Embedded Systems:

These types of embedded systems design with a single or 16- or 32-bit microcontroller, RISCs or DSPs. These types of embedded systems have both hardware and software complexities.

For developing embedded software for medium scale embedded systems, the main programming tools are C, C++, and JAVA, Visual C++, and RTOS, debugger, source code engineering tool, simulator and IDE.

1.2.6 Sophisticated Embedded Systems:

These types of embedded systems have enormous hardware and software complexities that may need ASIPs, IPs, PLAs, scalable or configurable processors.

They are used for cutting-edge applications that need hardware and software Co-design and components which have to assemble in the final system.

1.3 Overview of Embedded system Architecture:

Every embedded system consists of custom-built hardware built around a Central Processing Unit (CPU). This hardware also contains memory chips onto which the software is loaded. Once the software is transferred to the memory chip, the software will continue to run for a long time you don't need to reload new software. Now, let us see the details of the various building blocks of the hardware of an embedded system.

Block Diagram of embedded system is given by

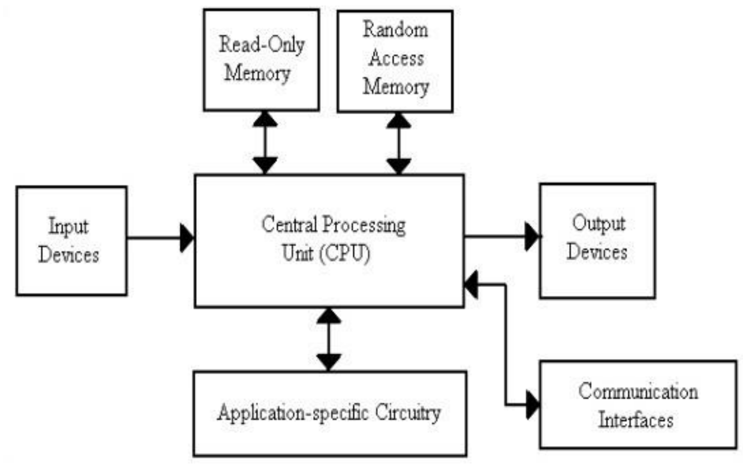


Figure 1.3: Embedded System Architecture

1.3.1 Central Processing Unit:

The Central Processing Unit (processor, in short) can be any of the following: microcontroller, microprocessor or Digital Signal Processing (DSP). A micro-controller is a low-cost processor. Its main attraction is that on the chip itself, there will be many other components such as memory, serial communication interface, analog-to-digital converter etc. So, for small applications, a micro-controller is the best choice as the number of external components required will be very less. On the other hand, microprocessors are more powerful, but you need to use many external components with them. DSP is used mainly for applications in which signaling is involved such as audio and video processing.

1.3.2 Memory:

The memory is categorized as Random Access Memory (RAM) and Read Only Memory (ROM). The content of the RAM will be erased if power is switched off to the chip, where ROM retains the contents even if the power is switched off. So, the firmware is stored in the ROM. When power is switched on, the processer reads the ROM. The program is executed.

1.3.3 Input Devices:

Unlike the desktop, the input devices to an embedded system have very limited capability. There will be no keyboard or a mouse, and hence interacting with the embedded system is no easy task. Many embedded systems will have a small keypad-you press one key to give a specific command. A keypad may be used to input only the digits. Many embedded systems used in process control do not have any input device for user interaction. They take inputs from sensors or transducers produce electrical signals that are in turn fed to other systems.

1.3.4 Output Devices:

The output devices of the embedded systems also have very limited capability. Some embedded systems will have a few Light Emitting Diodes (LEDs) to indicate the health status of the system modules, or for visual indication of alarms. A small Liquid Crystal Display (LCD) may also be used to display some important parameters.

1.4 Memory Architecture:

There two different type’s memory architectures there are:

1. Harvard Architecture
2. Von-Neumann Architecture

1.4.1 Harvard Architecture

Computers have separate memory areas for program instructions and data. There are two or more internal data buses, which allow simultaneous access to both instructions and data. The CPU fetches program instructions on the program memory bus.

The Harvard architecture is a computer architecture with physically separate storage and signal pathways for instructions and data. The term originated from the Harvard Mark I relay-based computer, which stored instructions on punched tape (24 bits wide) and data in electro-mechanical counters. These early machines had limited data storage, entirely contained within the central processing unit, and provided no access to the instruction storage as data. Programs needed to be loaded by an operator, the processor could not boot itself.

Modern uses of the Harvard architecture:

The principal advantage of the pure Harvard architecture - simultaneous access to more than one memory system - has been reduced by modified Harvard processors using modern CPU cache systems. Relatively pure Harvard architecture machines are used mostly in applications where tradeoffs, such as the cost and power savings from omitting caches, outweigh the programming penalties from having distinct code and data address spaces.

Digital signal processors (DSPs) generally execute small, highly-optimized audio or video processing algorithms. They avoid caches because their behavior must be extremely reproducible. The difficulties of coping with multiple address spaces are of secondary concern to speed of execution. As a result, some DSPs have multiple data memories in distinct address spaces to facilitate SIMD and VLIW processing. Texas Instruments TMS320 C55x processors, as one example, have multiple parallel data busses (two write, three read) and one instruction bus.

Block Diagram of Harvard Architecture is given by

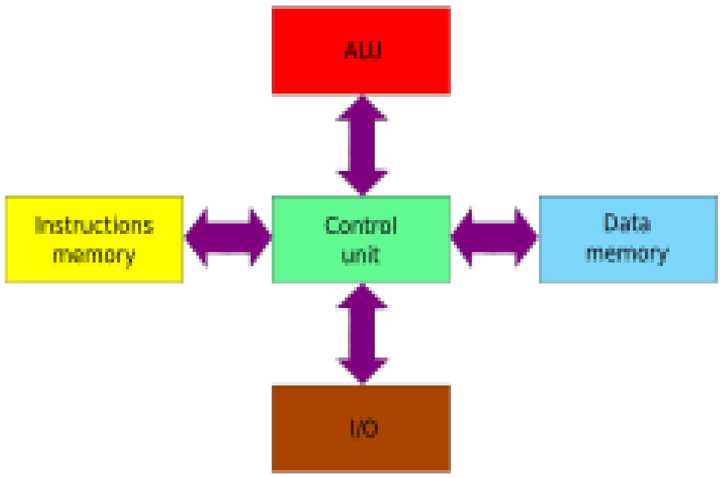


Figure 1.4: Harvard Architecture

Microcontrollers are characterized by having small amounts of program (flash memory) and data (SRAM) memory, with no cache, and take advantage of the Harvard architecture to speed processing by concurrent instruction and data access. The separate storage means the program and data memories can have different bit depths, for example using 16-bit wide instructions and 8-bit wide data. They also mean that instruction pre-fetch can be performed in parallel with other activities.

Examples include, the AVR by Atmel Corp, the PIC by Microchip Technology, Inc. and the ARM Cortex-M3 processor (not all ARM chips have Harvard architecture).

Even in these cases, it is common to have special instructions to access program memory as data for read-only tables, or for reprogramming.

1.4.2 Von-Neumann Architecture:

A computer has a single, common memory space in which both program instructions and data are stored. There is a single internal data bus that fetches both instructions and data. They cannot be performed at the same time

The Von Neumann Architecture is a design model for a stored-program digital computer that uses a central processing unit (CPU) and a single separate storage structure ("memory") to hold both instructions and data. It is named after the mathematician and early scientist John. Such computers implement a universal Turing machine and have a sequential architecture.

A stored-program digital is one that keeps its programmed instructions, as well as its data, in read-write, random-access memory (RAM). Stored-program computers were advancement over the program-controlled computers of the 1940s, such as the Colossus and the ENIAC, which were programmed by setting switches and inserting patch leads to route data and to control signals between various functional units.

In the vast majority of modern computers, the same memory is used for both data and program instructions. The mechanisms for transferring the data and instructions between the CPU and memory are, however, considerably more complex than the original von Neumann architecture.

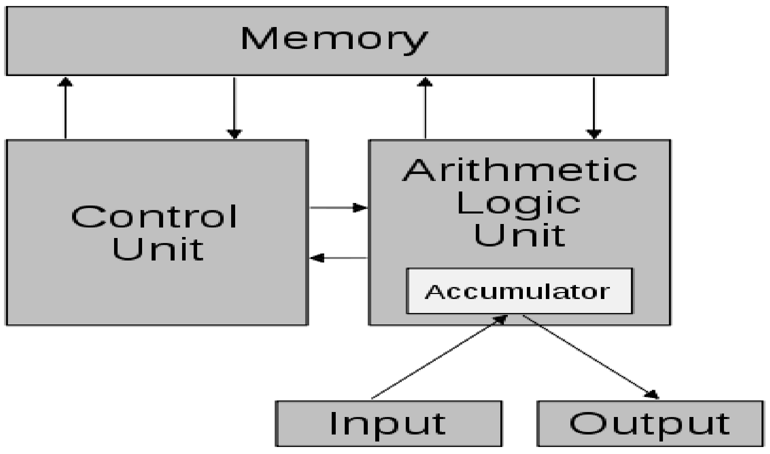


Figure 1.5: Schematic of the Von-Neumann Architecture

Basic Difference between Harvard and Von-Neumann Architecture:

- The primary difference between Harvard architecture and the Von Neumann architecture is in the Von Neumann architecture data and programs are stored in the same memory and managed by the same information handling system.
- Whereas the Harvard architecture stores data and programs in separate memory devices and they are handled by different subsystems.
- In a computer using the Von-Neumann architecture without cache; the central processing unit (CPU) can either be reading and instruction or writing/reading data to/from the memory. Both of these operations cannot occur simultaneously as the data and instructions use the same system bus.

- In a computer using the Harvard architecture the CPU can both read an instruction and access data memory at the same time without cache. This means that a computer with Harvard architecture can potentially be faster for a given circuit complexity because data access and instruction fetches do not contend for use of a single memory pathway.
- Today, the vast majority of computers are designed and built using the Von Neumann architecture template primarily because of the dynamic capabilities and efficiencies gained in designing, implementing, operating one memory system as opposed to two. Von Neumann architecture may be somewhat slower than the contrasting Harvard Architecture for certain specific tasks, but it is much more flexible and allows for many concepts unavailable to Harvard architecture such as self-programming, word processing and so on.
- Harvard architectures are typically only used in either specialized systems or for very specific uses. It is used in specialized digital signal processing (DSP), typically for video and audio processing products. It is also used in many small microcontrollers used in electronics applications such as Advanced RISK Machine (ARM) based products for many vendors.

1.5 Features of Embedded systems: -

- Embedded systems do a very specific task, they cannot be programmed to do different things.
- Embedded systems have very limited resources, particularly the memory. Generally, they do not have secondary storage devices such as the CDROM or the floppy disk.
- Embedded systems have to work against some deadlines. A specific job has to be completed within a specific time. In some embedded systems, called real-time systems, the deadlines are stringent. Missing a deadline may cause a catastrophe- loss of life or damage to property.
- Embedded systems are constrained for power. As many embedded systems
- Operate through a battery, the power consumption has to be very low.
- Some embedded systems have to operate in extreme environmental conditions such as very high temperatures and humidity.

1.6 Design process: -

Embedded system design is a quantitative job. The pillars of the system design methodology are the separation between function and architecture is an essential step from conception to implementation. In recent past, the search and industrial community has paid significant attention to the topic of hardware-software (HW/SW) code sign and has tackled the problem of coordinating the design of the parts to be implemented as software and the parts to be implemented as hardware avoiding the HW/SW integration problem marred the electronics system industry so long. In any large-scale embedded systems design methodology, concurrency must be considered as a first-class citizen at all levels of abstraction and in both hardware and software. Formal models & transformations in system design are used so that verification and synthesis can be applied to advantage in the design methodology. Simulation tools are used for exploring the design space for validating the functional and timing behaviours of embedded systems. Hardware can be simulated at different levels such as electrical circuits, logic gates, RTL, etc., using VHDL description. In some environments software development tools can be coupled with hardware simulators, while in others 05 the software is executed on the simulated hardware. The later approach is feasible only for small parts of embedded systems. Design of an embedded system using Intel's 80C188EB chip is shown in the figure. In order to reduce complexity, the design process is divided in four major steps: specification, system synthesis, and implementation synthesis and performance evaluation of the prototype.

1.6.1 Specification: -

During this part of the design process, the informal requirements of the analysis are transformed to formal specification using SDL.

1.6.2 System-Synthesis: -

For performing an automatic HW/SW partitioning, the system synthesis step translates the SDL specification to an internal system model switch contains problem graph& architecture graph. After system synthesis, the resulting system model is translated back to SDL.

1.6.3 Implementation-Synthesis: -

SDL specification is then translated into conventional implementation languages such as VHDL for hardware modules and C for software parts of the system.

1.6.4 Prototyping: -

On a prototyping platform, the implementation of the system under development is executed with the software parts running on multiprocessor unit and the hardware part running on a FPGA board known as phoenix, prototype hardware for Embedded Network Interconnect Accelerators

1.7 Applications of embedded systems: -

- Consumer Applications
- Computer Networking
- Telecommunications
- Wireless technologies
- Security

- Medical applications
- Aerospace and Defence

1.7.1 Consumer Applications:

At home we use a number of embedded systems which include digital camera, digital diary, DVD player, electronic toys, microwave oven, remote controls for TV and air-conditioner, VCO player, video game consoles, video records etc. Today’s high-tech car has about 20 embedded systems for transmission control, engine spark control, air-conditioning, navigation etc. Even wristwatches are now becoming embedded systems. The palmtops are powerful embedded systems using which we can carry out many general-purpose tasks such as playing games and word processing.

1.7.2 Computer Networking

Computer Networking products such as Bridges, Routers, Integrated Services Digital Networks (ISDN), Asynchronous Transfer Mode (ATM), X.25 and frame relay switches are embedded system which implemented the necessary data communication protocols. For example, a router interconnects two networks. The networks may be running different protocol stacks.

The router’s function is to obtain the data packets from incoming pores, analyze the packets and send them towards the destination after doing necessary protocol conversion. Most networking equipment’s, other than the end systems (desktop computers) we use to access the networks, are embedded systems.

1.7.3 Telecommunications:

In the field of telecommunications, the embedded systems can be categorized as subscriber terminals and network equipment. The subscriber terminals such as key telephones, ISDN phones, terminal adapters, web camera are embedded systems. The network equipment includes multiplexers, multiple access systems, Packet Assembler Dissembler (PADs), satellite modems etc.… IP phone, IP gateway, IP gatekeeper etc.… are the latest embedded systems that are provide very low-cost voice communication over the Internet.

1.7.4 Wireless Technologies:

Advances in mobile communication are paving way for many interesting applications using embedded systems. The mobile phone is one of the marvels of the last decade of the 20thcentury. It is a very powerful embedded system that provides voice communication while we are on the move. The Personal Digital Assistants and the palmtops can now be used to access multimedia services over the internet.

Mobile communication infrastructure such as base station controllers, mobile switching centers are also powerful embedded systems.

1.7.5 Security:

Security of persons and information has always been a major issue. We need to protect our homes and offices, and also the information we transmit and store. Developing embedded systems for security applications is one of the most lucrative businesses now-a-days. Security devices at homes, offices, airports etc.… for authentication and verification are embedded systems. Encryption devices are nearly 99 percent of the processors that are manufactured end up in embedded systems. Embedded systems find applications in every industrial segment – consumer electronics, transportation, avionics, biomedical engineering, manufacturing process control and industrial automation, data communication, telecommunication, defense, security etc

1.8 Introduction to Antenna Positioning System

An antenna positioning system is a technology used to adjust the position of an antenna to optimize the quality of wireless communication. The antenna is a critical component of wireless communication systems, as it transmits and receives signals over the airwaves. The positioning system ensures that the antenna is oriented correctly and pointed towards the source or destination of the signal to achieve optimal communication performance. Antenna positioning systems use a combination of sensors, algorithms, and motors to adjust the position of the antenna. The sensors detect the orientation of the antenna, and the algorithms calculate the optimal position based on the signal strength and direction. The motors then move the antenna to the calculated position. Antenna positioning systems are widely used in various applications such as satellite communication, wireless networks, and broadcasting. For example, in satellite communication, antenna positioning systems are used to track and maintain a connection with a moving satellite, while in broadcasting, they are used to adjust the direction of the antenna to reach a wider audience. Overall, antenna positioning systems play a crucial role in optimizing the performance of wireless communication systems and ensuring reliable connectivity.

An antenna positioning system is a mechanism or a system that is used to adjust the position of an antenna to optimize its performance in terms of signal reception or transmission. The system is typically used to ensure that the antenna is pointed in the right direction, which is critical for achieving optimal signal strength and minimizing interference. Antenna positioning systems are used in a variety of applications, such as satellite communication, terrestrial television, and wireless communication. In satellite communication, for instance, an antenna positioning system is used to track the position of a satellite and maintain a constant connection with it. In wireless communication, an antenna positioning system is used to ensure that the antenna is pointed in the direction of the intended receiver or transmitter, thereby enhancing signal strength and minimizing interference. Antenna positioning systems can be manual or automatic. In a manual system, the antenna is adjusted manually by an operator using mechanical controls. In an automatic system, the antenna position is adjusted automatically using sensors and motors controlled by a computer. The use of automatic antenna positioning systems can reduce the need for human intervention and improve efficiency and accuracy. Overall, antenna positioning systems play a critical role in optimizing the performance of antennas in various applications, ensuring that they are pointed in the right direction and maximizing their efficiency in terms of signal reception or transmission.

1.9 Introduction to IoT:

IoT stands for Internet of Things. It refers to a network of physical devices, vehicles, appliances, and other items that are embedded with sensors, software, and network connectivity, allowing them to collect and exchange data over the internet. The basic idea of IoT is to connect devices to the internet and enable them to communicate with each other and with other systems to improve efficiency, productivity, and convenience.

The sensors embedded in these devices can collect data on a wide range of parameters such as temperature, humidity, location, and movement, which can be used to make informed decisions and automate tasks. The IoT has numerous applications in various fields, including home automation, healthcare, transportation, agriculture, and manufacturing. For example, in home automation, smart devices such as thermostats, lights, and security systems can be controlled remotely through a mobile app, while in healthcare, wearables such as smart watches can monitor vital signs and alert doctors in case of emergencies. The growth of IoT is driven by advancements in technology such as wireless connectivity, cloud computing, and big data analytics. As more and more devices become connected to the internet, the potential for IoT to transform various industries and improve our lives is immense.

Characteristics of IoT:

The characteristics of IoT (Internet of Things) can be summarized as follows:

1. Connectivity: IoT devices are connected to the internet and can communicate with other devices and systems. They can also collect and exchange data over the internet.

2. Interoperability: IoT devices are designed to work with each other, regardless of the manufacturer or operating system. This allows for seamless integration of devices and systems, enabling them to work together to achieve common goals.

3. Scalability: IoT systems can be easily scaled up or down depending on the number of devices and the amount of data being generated. This allows for flexible deployment of IoT systems to meet changing needs. 4. Data-driven: IoT systems rely on data generated by sensors and devices to provide insights and inform decision-making. The data can be used for various purposes such as predicting maintenance needs, optimizing performance, and improving efficiency.

5. Real-time: IoT systems can provide real-time data and insights, enabling timely action and decision-making. This is particularly important in applications such as healthcare, where real-time monitoring of patient health can help prevent emergencies and save lives.

6. Security: IoT systems require robust security measures to protect against data breaches, cyber-attacks, and other threats. Security is an essential aspect of IoT to ensure the privacy and safety of users and their data.

Applications of IoT:

IoT (Internet of Things) has a wide range of applications across various industries. Here are some examples of applications of IoT:

1. Smart Homes: IoT-enabled devices such as smart thermostats, lighting, and security systems can be controlled remotely via smartphones, offering convenience and energy savings.

2. Healthcare: IoT devices such as wearables and medical sensors can collect and transmit data on patients' health, enabling remote monitoring and proactive medical interventions.

3. Manufacturing: IoT systems can optimize production and supply chain management by monitoring equipment performance, tracking inventory, and improving logistics.

4. Transportation: IoT-enabled systems can improve safety and efficiency in transportation by tracking vehicle performance, optimizing routes, and providing real-time traffic information.

5. Agriculture: IoT devices can monitor soil conditions, crop growth, and weather patterns, enabling farmers to make data-driven decisions to improve crop yields and reduce waste.

6. Energy Management: IoT systems can optimize energy consumption in buildings and industrial facilities by monitoring usage and adjusting settings for maximum efficiency.

7. Retail: IoT-enabled systems can provide personalized shopping experiences by tracking customer behavior and preferences, optimizing inventory management, and improving supply chain efficiency.

8. Wearables: wearable technology is the hall mark of IoT applications and one of the earliest industries to IoT.

Advantages and Disadvantages:

Advantages of IoT:

1. Increased Efficiency: IoT systems can improve efficiency by automating processes, reducing the need for manual intervention, and optimizing resource utilization.

2. Improved Decision Making: IoT systems generate large amounts of data that can be analyzed to provide valuable insights for informed decision-making.

3. Enhanced Safety and Security: IoT systems can enhance safety and security by monitoring critical infrastructure and providing real-time alerts in case of any potential threats.

4. Cost Savings: IoT systems can reduce costs by optimizing resource usage, reducing energy consumption, and improving maintenance efficiency.

5. Improved Customer Experience: IoT systems can provide personalized and customized experiences to customers by tracking their behavior and preferences.

Disadvantages of IoT:

1. Security Concerns: IoT systems are vulnerable to security breaches, hacking, and unauthorized access, leading to loss of data, privacy concerns, and potential safety risks.

2. Complexity: IoT systems can be complex and require specialized skills for installation, maintenance, and management.

3. Interoperability Challenges: IoT systems may face interoperability challenges due to a lack of standardization and compatibility issues between different devices and systems.

CHAPTER 2

2. LITERATURE SURVEY

LITERATURE SURVEY

Antenna positioning systems are critical components of wireless communication networks, enabling efficient and reliable signal transmission. With the increasing adoption of the Internet of Things (IoT), there has been a growing demand for IoT based antenna positioning systems to enhance communication capabilities. In this literature survey, we will review some of the key research works that have been published on IoT-based antenna positioning systems.

1. "Wireless Sensor Network for Real-Time Antenna Positioning" by H. Amrouche et al in the year 2015.

This paper proposes a wireless sensor network-based antenna positioning system that can accurately determine the position of an antenna in real-time. The proposed system utilizes a network of sensors and actuators to control the position of the antenna, and it has been tested in both indoor and outdoor environments.

2. "Internet of Things-Based Antenna Positioning System for Smart Agriculture" by M. Hasan in the year 2019.

This paper presents an IoT based antenna positioning system for smart agriculture applications. The proposed system uses wireless sensor nodes to measure the soil moisture level and temperature and adjust the antenna position accordingly. The system has been tested in a farm setting, and the results show that it can significantly improve the performance of wireless communication networks in agriculture.

3. "Design and Development of an IoT-based Antenna Positioning System for Wireless Sensor Networks" by S. S. Hema Latha and P. S. Rajan in the year 2021.

This paper discusses the design and development of an IoT-based antenna positioning system for wireless sensor networks. The proposed system includes a microcontroller, a WIFI module, and a servo motor to control the antenna position. The system can be controlled using a web-based interface, making it accessible from anywhere. The authors demonstrate the effectiveness of the proposed system in a real-world environment.

4. "IoT-Based Antenna Positioning System for Enhanced Wireless Communication" by V. S. Suresh and S. M. Mahajan in the year 2019.

This paper presents an IoT-based antenna positioning system for enhanced wireless communication. The system includes a Raspberry Pi board, a Wi-Fi module, a stepper motor, and an ultrasonic sensor. The authors demonstrate the effectiveness of the proposed system in improving the signal strength and quality in a wireless communication scenario.

5. "An IoT-Based Smart Antenna Positioning System for Precision Agriculture" by A. K. Singh and S. Kumar in the year 2020.

This paper discusses the design and development of an IoT-based smart antenna positioning system for precision agriculture. The proposed system includes a microcontroller, a GPS module, a Wi-Fi module, and a servo motor to control the antenna position. The system can be controlled using a mobile application, making it easy for farmers to use. The authors demonstrate the effectiveness of the proposed system in a precision agriculture scenario.

6. "IoT-Based Antenna Positioning System for Improved Wireless Coverage in Smart Buildings" by N. R. Garg and A. K. Singh in the year 2020.

This paper presents an IoT-based antenna positioning system for improved wireless coverage in smart buildings. The proposed system includes a Raspberry Pi board, a Wi-Fi module, a stepper motor, and an ultrasonic sensor. The authors demonstrate the effectiveness of the proposed system in improving the signal strength and quality in a smart building environment.

7. "IoT-Based Antenna Positioning System for Indoor Localization" by M. A. Rahman and M. R. Amin in the year 2021.

This paper discusses the design and development of an IoT-based antenna positioning system for indoor localization. The proposed system includes a microcontroller, a Wi-Fi module, a servo motor, and an ultrasonic sensor. The system can be controlled using a mobile application, making it easy for users to use. The authors demonstrate the effectiveness of the proposed system in an indoor localization scenario.

8. "Design and Implementation of an IoT-Based Antenna Positioning System for Wireless Sensor Networks" by M. Khaleel Ur Rahman and Md. Humayun Kabir in the year 2021.

This paper proposes an IoT-based antenna positioning system for wireless sensor networks (WSNs). The system uses a microcontroller and sensors to measure the orientation of the antenna and communicate the information over the internet to a central server.

9. "An IoT-Based Antenna Positioning System for Smart Agriculture" by Xiaoxia Ji, Yan Wang, and Hong sheng Xi in the year 2018.

This paper presents an IoT-based antenna positioning system for smart agriculture applications. The system uses GPS and environmental sensors to measure the location and condition of crops and adjusts the position of the antenna accordingly.

10. "IoT-Based Smart Antenna Positioning System for Mobile Robots" by Hadi Abdullah and

Mohammad Al-Khader in the year 2020.

This paper proposes an IoT-based smart antenna positioning system for mobile robots. The system uses ultrasonic sensors to measure the distance between the robot and the antenna and adjusts the antenna position accordingly.

11. "An IoT-Based Antenna Positioning System for Smart Cities" by Jinsheng Shi, Lin Huang, and Zhi gang Wen in the year 2021.

This paper proposes an IoT-based antenna positioning system for smart city applications. The system uses GPS and environmental sensors to measure the location and condition of the city infrastructure and adjusts the position of the antenna accordingly.

12. "IoT-Based Antenna Positioning System for Wireless Communication Networks" by Mohamed Fathi and Ahmed Elchibey in the year 2019.

This paper presents an IoT-based antenna positioning system for wireless communication networks. The system uses a microcontroller and sensors to measure the orientation of the antenna and communicate the information over the internet to a central server.

A literature survey has showed that S. Godse and other authors introduces the system [1] PIC microcontroller is used as a microcontroller in this system IJARSCT ISSN (Online) 2581-9429 International Journal of Advanced Research in Science, Communication and Technology (IJARSCT) Volume 6, Issue 1, June 2021 Copyright to IJARSCT DOI: 10.48175/IJARSCT-1456 675 www.ijarsct.co.in Impact Factor: 4.819 and it can be controlled by wireless device i.e., Bluetooth. The microcontroller receive data from remote control and send it to the motor through an interface. Chatterjee S., "Industrial Electronics and Control", New Delhi: Tata Mc Graw-Hill Publishing Company Limited. Using the remote control improves the advanced technology. And using the microcontroller develops the motor to maintain the desired position. Although this is the first approaching step to the control system, automation system and robotics systems, these can greatly serve to the industrial control. Richard, V., Editor, "Motor Control Electronics Handbook, Mc Graw-Hill, Boston. Nowadays, the most popular motor is a servo motor that is used to control and drive for heavy load application. On the other hand, the servo motor cost is extensively high for this application.

CHAPTER 3

3. EXISTING SYSTEM

Description:

IoT or the internet of things is characterized as a forthcoming innovation that empowers us to create worldwide networked machines and also the devices that can be helped for exchanging of communication. As we all know that the real-time application has been increasing day by day, the smart connection also had increased. Rapid population growth led to the increase in global life expectancy and the advance of technology, paving the pathway for the creation of age-friendly environments.

As we all know wireless communication systems operate on antennas for effective reception of signals. It is quite necessary to properly position the antennas in the direction of the transmitter for effective wireless communication. So here we propose an IoT-based antenna positioning system that allows for the remotely positioning of antennas based on IoT. In this system, the sensors and motors will be mounted on the antenna to detect its direction and its direction will be changed by motors using IoT. When the direction of a transmitting station changes over time, the antenna direction must also be changed accordingly. This system will help in monitoring antenna direction and transmitting new coordinates to position the antenna. This system appropriately positions the antenna accordingly. So basically, using this system we can wirelessly position antennas in the desired directions using IoT.

Block diagram of Existing system:

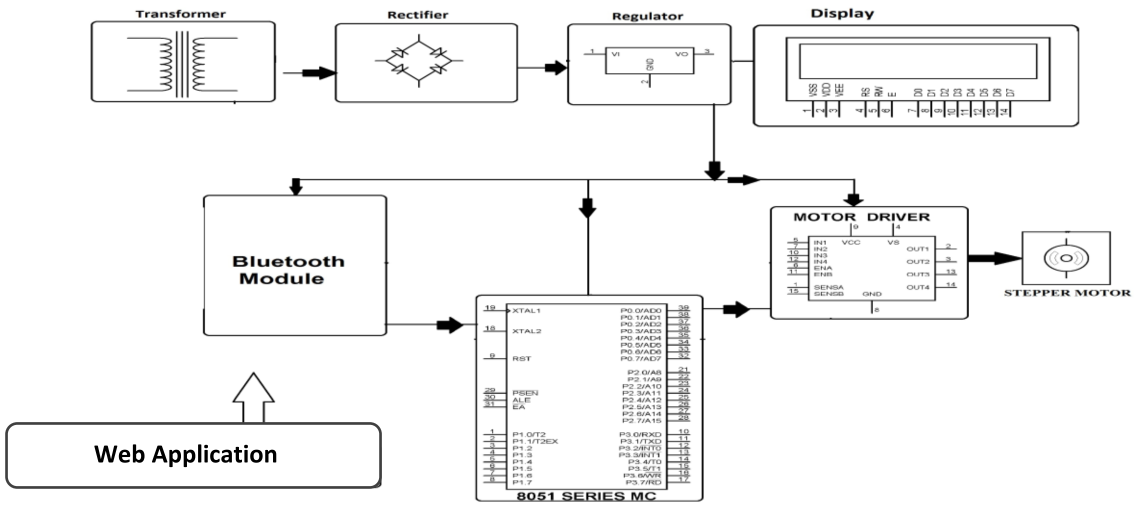


Fig. 3.1: Block Diagram of Existing System

IoT or the internet of things is characterized as a forthcoming innovation that empowers us to create worldwide networked machines and also the devices that can be helped for exchanging of communication. As we all know that the real-time application has been increasing day by day, the smart connection also had increased. Rapid population growth led to the increase in global life expectancy and the advance of technology, paving the pathway for the creation of age-friendly environments. It is quite necessary to properly position the antennas in the direction of the transmitter for effective wireless communication. So here we propose an IoT-based antenna positioning system that allows for the remotely positioning of antennas based on IoT. In this system, the sensors and motors will be mounted on the antenna to detect its direction and its direction will be changed by motors using IoT. When the direction of a transmitting station changes over time, the antenna direction must also be changed accordingly. This system will help in monitoring antenna direction and transmitting new coordinates to position the antenna. This system appropriately positions the antenna accordingly. So basically, using this system we can wirelessly position antennas in the desired directions using IoT.

Disadvantages of Existing System:

- Dependence on Internet Connectivity
- Limited Range
- High Cost.

CHAPTER 4

4. PROPOSED SYSTEM

All wireless communication systems work on antennas for reception of signals. Proper positioning of antennas is necessary according to satellites/transmitters to achieve effective wireless communication. So here we propose an IOT based antenna positioning system that allows for remotely positioning of antennas based over IOT. Here we use sensor-based system with motor on each antenna using antenna to check its facing direction that is transmitted over IOT. If the direction of a satellite or transmitting station changes over time, the antenna direction must also be changed accordingly. The receiving antennas may be placed far apart from each other across the globe. So, our system allows for antenna positioning over very long distances. The antenna positions are visible over internet to controlling operator on the IOT GUI. We here use IOT Gecko to develop the antenna monitoring system. Our system allows for monitoring antenna direction as well as transmitting new coordinates to position the antenna and motor appropriately positions the antenna accordingly.

Block Diagram of Proposed system

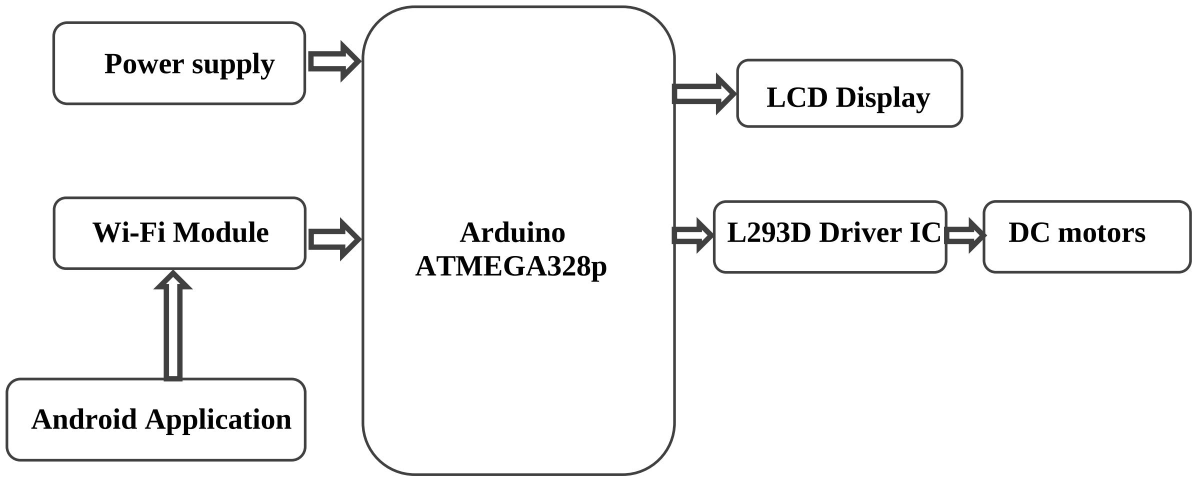


Fig.4.1: Block Diagram of Proposed System

Working Principle:

The working principle of an IoT-based antenna positioning system involves the use of various sensors, actuators, and IoT devices to control the position and orientation of an antenna. Here is a simplified overview of how such a system might work:

1. Data Collection: The antenna positioning system collects data.
2. IoT Devices and Connectivity: The sensor data is transmitted to an IoT device, such as a

microcontroller or Raspberry Pi, via a wireless communication protocol such as Bluetooth, Wi-Fi, or Zigbee.

3. Data Processing and Analysis: The IoT device processes and analyzes the sensor data to determine the optimal position and orientation of the antenna. This involves using algorithms to calculate the direction of the signal source and the best angle for the antenna to receive or transmit the signal.

4. Actuator Control: Once the optimal position and orientation of the antenna have been determined, the IoT device sends commands to the actuators to adjust the antenna's position and orientation accordingly.

5. Feedback and Control Loop: The positioning system continuously monitors the signal quality and adjusts the antenna's position and orientation to maintain optimal performance. This is done using a feedback and control loop, where the system constantly adjusts the antenna's position based on the feedback it receives from the sensors.

Advantages of Proposed System:

- Reduces the time.
- Cost effective
- Range of Detection is high.

<u>4.1 POWER SUPPLY</u>

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.

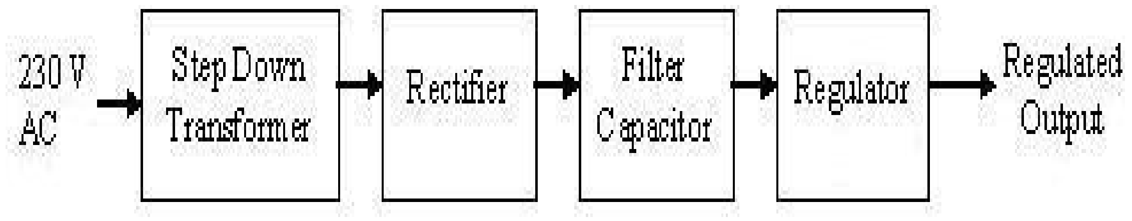


Fig. 4.2: shows the basic block diagram of a fixed regulated power supply.

4.1.1 Transformer

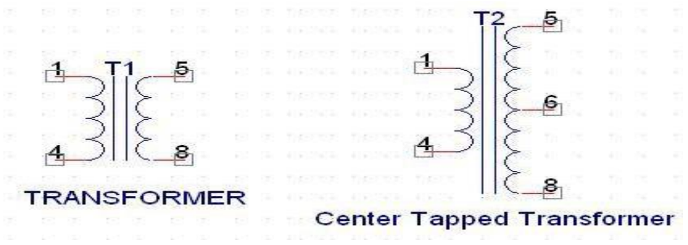


Fig.4.3: Transformer

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.

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

So, $I_p V_p = I_s V_s$ & $\frac{V_s}{V_p} = \frac{N_s}{N_p}$

The secondary voltage of the transformer depends on the number of turns in the Primary as well as in the secondary.

4.1.2 Rectifier

A rectifier is a device that converts an AC signal into DC signal. For rectification purpose we use a diode, a diode is a device that allows current to pass only in one direction i.e., when the anode of the diode is positive with respect to the cathode also called as forward biased condition & blocks current in the reversed biased condition.

Rectifier can be classified as follows:

1) Half Wave rectifier.

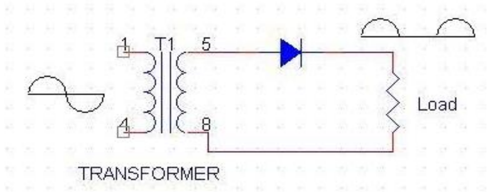


Fig. 4.4: Half Wave Rectifier

This is the simplest type of rectifier as you can see in the diagram a half wave rectifier consists of only one diode. When an AC signal is applied to it during the positive half cycle the diode is forward biased & current flows through it. But during the negative half cycle diode is reverse biased & no current flows through it. Since only one half of the input reaches the output, it is very inefficient to be used in power supplies.

2) Full wave rectifier.

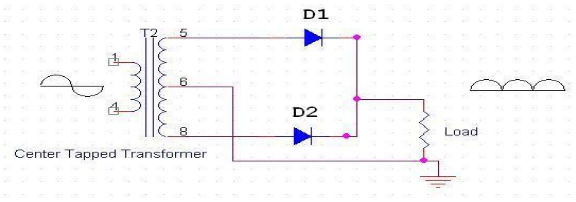


Fig. 4.5: Full Wave Rectifier

Half wave rectifier is quite simple but it is very inefficient, for greater efficiency we would like to use both the half cycles of the AC signal. This can be achieved by using a center tapped transformer i.e., we would have to double the size of secondary winding & provide connection to the center. So, during the positive half cycle diode D1 conducts & D2 is in reverse biased condition. During the negative half cycle diode D2 conducts & D1 is reverse biased. Thus, we get both the half cycles across the load. One of the disadvantages of Full Wave Rectifier design is the necessity of using a center tapped transformer, thus increasing the size & cost of the circuit. This can be avoided by using the Full Wave Bridge Rectifier.

3) Bridge Rectifier

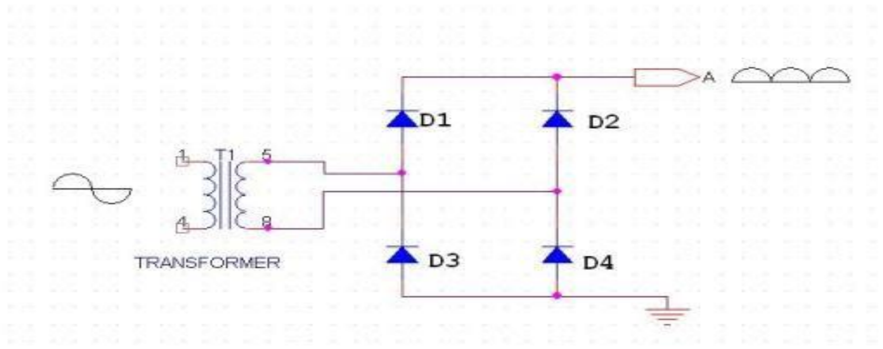
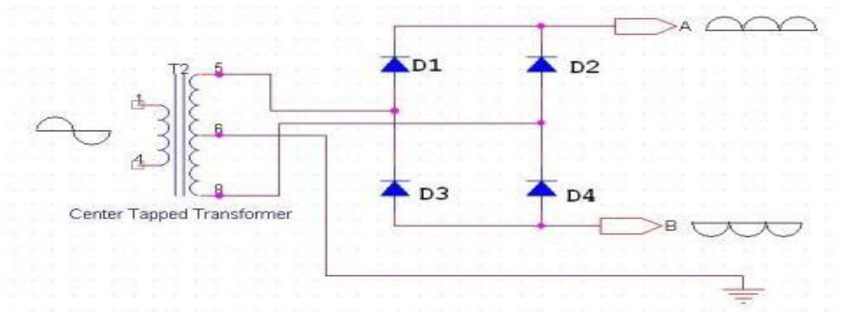


Fig.4.6: Bridge Rectifier

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. Full Bridge Wave Rectifier 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.



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.

4.1.3 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.

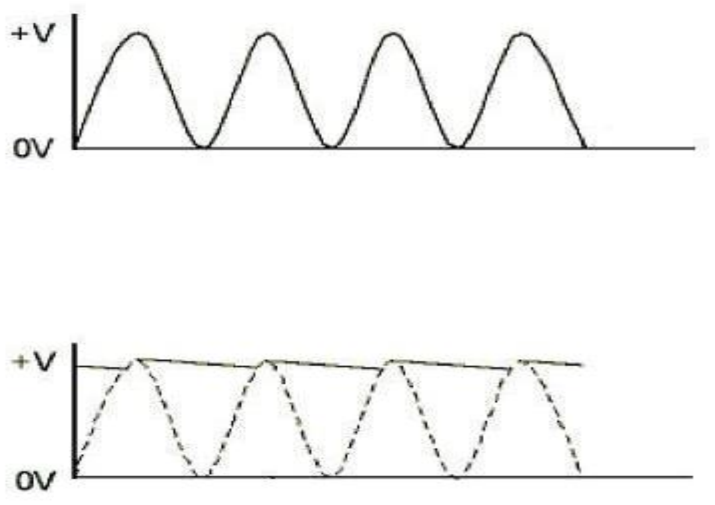


Fig. 4.7: Filter Capacitor output Waveforms

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=25hz

Whereas a full wave rectifier has Two peaks in one cycle so F=100hz

4.1.4 Voltage Regulators

A Voltage regulator is a device which converts varying input voltage into a constant regulated output voltage. Voltage regulator can be of two types:

Linear Voltage Regulator Also called as Resistive Voltage regulator because they dissipate the excessive voltage resistively as heat.

Switching Regulators, they regulate the output voltage by switching the Current ON/OFF very rapidly. Since their output is either ON or OFF it dissipates very low power thus achieving higher efficiency as compared to linear voltage regulators. But they are more complex & generate high noise due to their switching action. For low level of output power switching regulators tend to be costly but for higher output wattage they are much cheaper than linear regulators.

The most commonly available Linear Positive Voltage Regulators are the 78XX series where the XX indicates the output voltage. And 79XX series is for Negative Voltage Regulators.

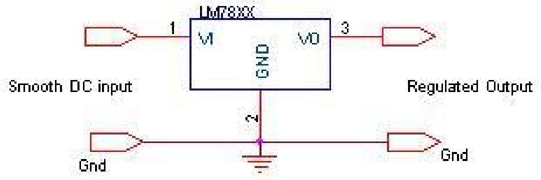


Fig. 4.8: Voltage Regulator

After filtering the rectifier output the signal is given to a voltage regulator. The maximum input voltage that can be applied at the input is 35V.Normally there is a 2-3 Volts drop across the regulator so the input voltage should be at least 2-3 Volts higher than the output voltage. If the input voltage gets below the Vmin of the regulator due to the ripple voltage or due to any other reason the voltage regulator will not be able to produce the correct regulated voltage.

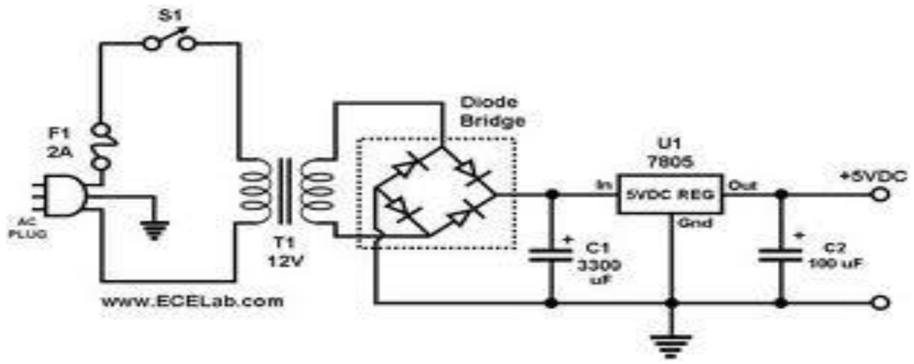


Fig.4.9: Circuit Diagram of power supply

4.2 IC 7805

7805 is an integrated three-terminal positive fixed linear voltage regulator. It supports an input voltage of 10 volts to 35 volts and output voltage of 5 volts. It has a current rating of 1 amp although lower current models are available. Its output voltage is fixed at 5.0V. The 7805 also has a built-in current limiter as a safety feature. 7805 is manufactured by many companies, including National Semiconductors and Fairchild Semiconductors. The 7805 will automatically reduce output current if it gets too hot. The last two digits represent the voltage; for instance, the 7812 is a 12-volt regulator. The 78xx series of regulators is designed to work in complement with the 79xx series of negative voltage regulators in systems that provide both positive and negative regulated voltages, since the 78xx series can't regulate negative voltages in such a system. The 7805 & 78 is one of the most common and well-known of the 78xx series regulators, as it's small component count and medium-power regulated 5V make it useful for powering TTL devices.

Specifications of IC7805 is shown in below table:

SPECIFICATIONS	**IC 7805**
V_{out}	5V
V_{ein} - V_{out} Difference	5V - 20V
Operation Ambient Temp	0 - 125°C
Output I_{max}	1A

Table 4.1: Specifications of IC7805

4.3 L293D Driver IC

Introduction

The L293 and L293D are quadruple high-current half-H drivers. The L293 is designed to provide bidirectional drive currents of up to 1 A at voltages from 4.5 V to 36 V. The L293D is designed to provide bidirectional drive currents of up to 600-mA at voltages from 4.5 V to 36 V. Both devices are designed to drive inductive loads such as relays, solenoids, dc and bipolar stepping motors, as well as other high-current/high-voltage loads in positive-supply applications. All inputs are TTL compatible. Each output is a complete totem-pole drive circuit, with a Darlington transistor sink and a pseudo- Darlington source. Drivers are enabled in pairs, with drivers 1 and 2 enabled by 1,2EN and drivers 3 and 4 enabled by 3,4EN. When an enable input is high, the associated drivers are enabled, and their outputs are active and in phase with their inputs. When the enable input is low, those drivers are disabled, and their outputs are off and in the high-impedance state. With the proper data inputs, each pair of drivers forms a full-H (or bridge) reversible drive suitable for solenoid or motor applications.

Features:

- Featuring Unit rode L293 and L293D
- Products Now from Texas Instruments

- Wide Supply-Voltage Range: 4.5 V to 36 V
- Separate Input-Logic Supply
- Internal ESD Protection
- Thermal Shutdown
- High-Noise-Immunity Inputs Functionally Similar to SGS L293 and SGS L293D
- Output Current 1 A Per Channel (600 mA for L293D)
- Peak Output Current 2 A Per Channel (1.2 A for L293D)
- Output Clamp Diodes for Inductive Transient Suppression (L293D)

Pin diagram:

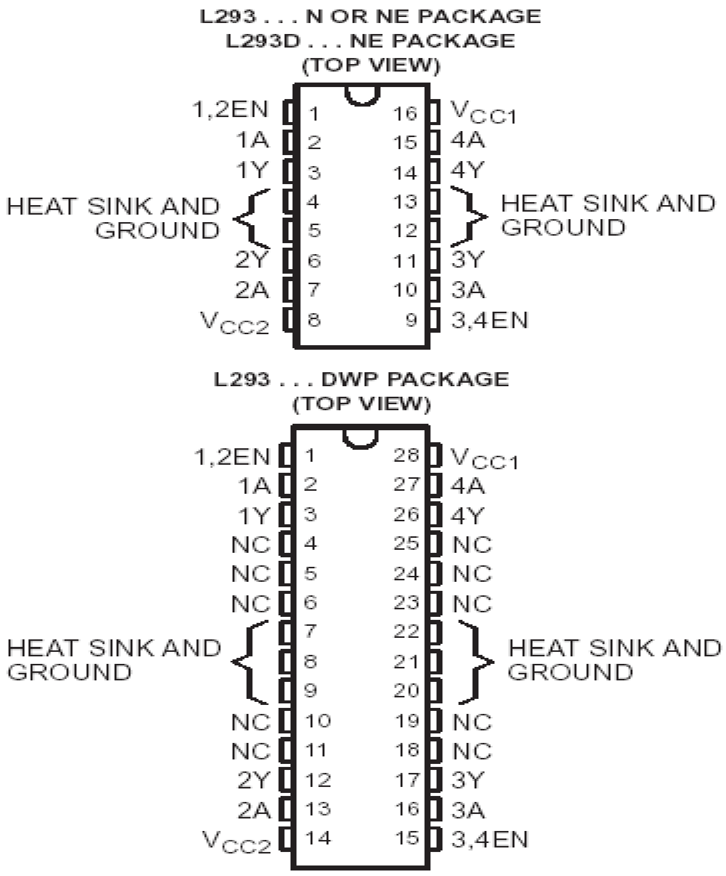


Fig. 4.10: Pin Diagram of L293D Driver

Description:

On the L293, external high-speed output clamp diodes should be used for inductive transient suppression. A VCC1 terminal, separate from VCC2, is provided for the logic inputs to minimize device power dissipation. The L293and L293D are characterized for operation from 0 C to 70 C.

Block diagram:

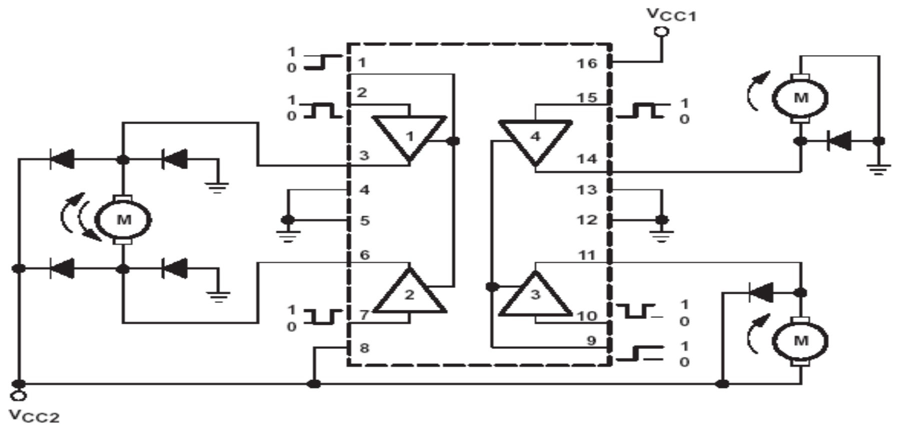
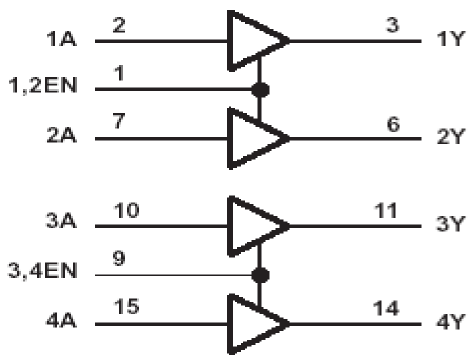


Fig.4.11: Block Diagram of L293D

Logic diagram:



FUNCTION TABLE
(each driver)

INPUTS†		OUTPUT
A	EN	Y
H	H	H
L	H	L
X	L	Z

H = high level, L = low level, X = irrelevant,
Z = high impedance (off)
† In the thermal shutdown mode, the output is in the high-impedance state, regardless of the input levels.

Fig.4.12: Logic and Function table of L293D

Applications:

- Audio
- Automotive
- Broadband
- Digital control
- Military
- Optical networking
- Security

4.4. DC MOTOR

Introduction

A DC motor is designed to run on DC electric power. Two examples of pure DC designs are Michael Faraday's homopolar motor (which is uncommon), and the ball bearing motor, which is (so far) a novelty. By far the most common DC motor types are the brushed and brushless types, which use internal and external commutation respectively to create an oscillating AC current from the DC source -- so they are not purely DC machines in a strict sense.

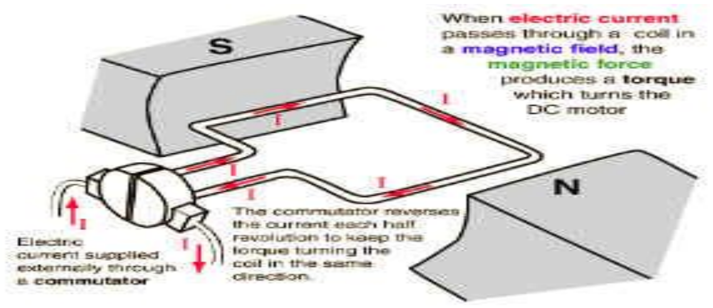

S
N
When electric current passes through a coil in a magnetic field, the magnetic force produces a torque which turns the DC motor
The commutator reverses the current each half revolution to keep the torque turning the coil in the same direction.
Electric current supplied externally through a commutator
I

Fig.4.13: DC Motor

<u>**Types of DC motors:**</u>

1. Brushed DC Motors
2. Brushless DC motors
3. Coreless DC motors

1. Brushed DC motors:

The classic DC motor design generates an oscillating current in a wound rotor with a split ring commutator, and either a wound or permanent magnet stator. A rotor consists of a coil wound around a rotor which is then powered by any type of battery.

Many of the limitations of the classic commutator DC motor are due to the need for brushes to press against the commutator. This creates friction. At higher speeds, brushes have increasing difficulty in maintaining contact. Brushes may bounce off the irregularities in the commutator surface, creating sparks. This limits the maximum speed of the machine. The current density per unit area of the brushes limits the output of the motor. The imperfect electric contact also causes electrical noise. Brushes eventually wear out and require replacement, and the commutator itself is subject to wear and maintenance. The commutator assembly on a large machine is a costly element, requiring precision assembly of many parts. there are three types of dc motor 1. dc series motor 2. dc shunt motor 3. dc compound motor - these are also two type a. cumulative compound b. deferral compound.

2. Brushless DC motors:

Some of the problems of the brushed DC motor are eliminated in the brushless design. In this motor, the mechanical "rotating switch" or commutator/brush gear assembly is replaced by an external electronic switch synchronized to the rotor's position. Brushless motors are typically 85-90% efficient, whereas DC motors with brush gear are typically 75-80% efficient.

Midway between ordinary DC motors and stepper motors lies the realm of the brushless DC motor. Built in a fashion very similar to stepper motors, these often use a permanent magnet external rotor, three phases of driving coils, one or more Hall effect sensors to sense the position of the rotor, and the associated drive electronics. The coils are activated, one phase after the other, by the drive electronics as cued by the signals from the Hall effect sensors. In effect, they act as three-phase synchronous motors containing their own variable-frequency drive electronics. A specialized class of brushless DC motor controllers utilize EMF feedback through the main phase connections instead of Hall effect sensors to determine position and velocity. These motors are used extensively in electric radio-controlled vehicles. When configured with the magnets on the outside, these are referred to by modelists as outrunner motors. Brushless DC motors are commonly used where precise speed control is necessary, as in computer disk drives or in video cassette recorders, the spindles within CD, CD-ROM (etc.) drives, and mechanisms within office products such as fans, laser printers and photocopiers.

They have several advantages over conventional motors:

- Compared to AC fans using shaded-pole motors, they are very efficient, running much cooler than the equivalent AC motors. This cool operation leads to much-improved life of the fan's bearings.
- Without a commutator to wear out, the life of a DC brushless motor can be significantly longer compared to a DC motor using brushes and a commutator. Commutation also tends to cause a great deal of electrical and RF noise; without a commutator or brushes, a brushless motor may be used in electrically sensitive devices like audio equipment or computers.
- The same Hall effect sensors that provide the commutation can also provide a convenient tachometer signal for closed-loop control (servo-controlled) applications. In fans, the tachometer signal can be used to derive a "fan OK" signal.
- The motor can be easily synchronized to an internal or external clock, leading to precise speed control.
- Brushless motors have no chance of sparking, unlike brushed motors, making them better suited to environments with volatile chemicals and fuels. Also, sparking generates ozone which can accumulate in poorly ventilated buildings risking harm to occupants' health.
- Brushless motors are usually used in small equipment such as computers and are generally used to get rid of unwanted heat.

- They are also very quiet motors which is an advantage if being used in equipment that is affected by vibrations.

Modern DC brushless motors range in power from a fraction of a watt to many kilowatts. Larger brushless motors up to about 100 kW rating are used in electric vehic. They also find significant use in high-performance electric model aircraft.

3. Coreless DC motors:

Nothing in the design of any of the motors described above requires that the iron (steel) portions of the rotor actually rotate; torque is exerted only on the windings of the electromagnets. Taking advantage of this fact is the coreless DC motor, a specialized form of a brush or brushless DC motor. Optimized for rapid acceleration, these motors have a rotor that is constructed without any iron core. The rotor can take the form of a winding-filled cylinder inside the stator magnets, a basket surrounding the stator magnets, or a flat *pancake* (possibly formed on a printed wiring board) running between upper and lower stator magnets. The windings are typically stabilized by being impregnated with Electrical epoxy potting systems. Filled epoxies that have moderate mixed viscosity and a long gel time. These systems are highlighted by low shrinkage and low exotherm. Typically, UL 1446 recognized as a potting compound for use up to 180C (Class H) UL File No. E 210549.

Because the rotor is much lighter in weight (mass) than a conventional rotor formed from copper windings on steel laminations, the rotor can accelerate much more rapidly, often achieving a mechanical time constant under 1 m/s. This is especially true if the windings use aluminum rather than the heavier copper. But because there is no metal mass in the rotor to act as a heat sink, even small coreless motors must often be cooled by forced air.

4.5 ESP8266 WIFI MODULE:

4.5.1 Introduction

The **ESP8266** is a low-cost Wi-Fi microchip, with a full TCP/IP stack and microcontroller capability, produced by Expressive Systems in Shanghai, China. The chip first came to the attention of Western makers in August 2014 with the **ESP-01** module, made by a third-party manufacturer Ai-Thinker. This small module allows microcontrollers to connect to a Wi-Fi network and make simple TCP/IP connections using Hayes-style commands. However, at first, there was almost no English-language documentation on the chip and the commands it accepted. The very low price and the fact that there were very few external components on the module, which suggested that it could eventually be very inexpensive in volume, attracted many hackers to explore the module, the chip, and the software on it, as well as to translate the Chinese documentation. The **ESP8285** is an ESP8266 with 1 MiB of built-in flash, allowing the building of single-chip devices capable of connecting to Wi-Fi These microcontroller chips have been succeeded by the ESP32 family of devices, including the pin-compatible ESP32-C3.

4.5.2 Features:

- 802.11 b/g/n
- Wi-Fi Direct (P2P), soft-AP
- Integrated TCP/IP protocol stack
- Integrated TR switch, balun, LNA, power amplifier and matching network
- Integrated PLLs, regulators, DCXO and power management units
- +19.5dBm output power in 802.11b mode
- Power down leakage current of <10uA
- 1MB Flash Memory Integrated low power 32-bit CPU could be used as application processor
- SDIO 1.1 / 2.0, SPI, UART
- STBC, 1×1 MIMO, 2×1 MIMO
- A-MPDU & A-MSDU aggregation & 0.4ms guard interval
- Wake up and transmit packets in < 2ms
- Standby power consumption of < 1.0mW (DTIM3)

4.5.3 Description:

The ESP8266 WIFI Module is a self-contained SOC with integrated TCP/IP protocol stack that can give any microcontroller access to your WIFI network. The ESP8266 is capable of either hosting an application or offloading all Wi-Fi networking functions from another application processor. Each ESP8266 module comes pre-programmed with an AT command set firmware, meaning, you can simply hook this up to your Arduino device and get about as much WIFI-ability as a WIFI Shield offers (and that’s just out of the box)! The ESP8266 module is an extremely cost-effective board with a huge, and ever growing, community.

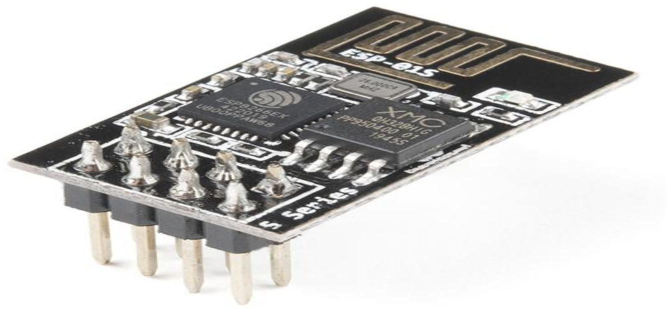


Fig.4.14: ESP8266 WIFI Module

Pin configuration of ESP8266 WIFI is shown in below figure:

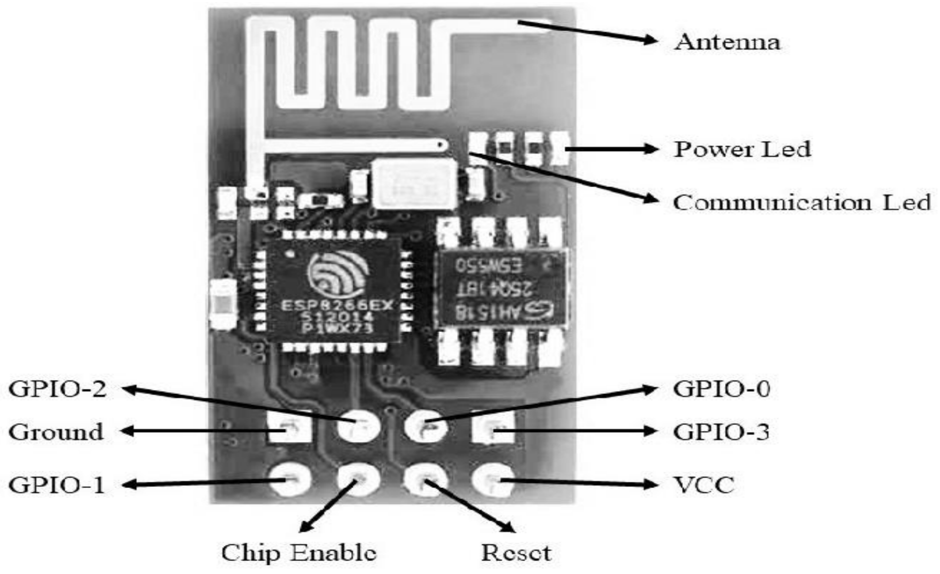


Fig.4.15: Pin Configuration of ESP8266 WIFI module

4.5.4 Specification of ESP 8266:

- Wi-Fi Direct (P2P), soft-AP
- Integrated TCP/IP protocol stack
- Integrated TR switch, balun, LNA, power amplifier and matching network
- Integrated PLLs, regulators, DCXO and power management units
- 19.5dBm output power in 802.11b mode
- Power down leakage current of <10uA
- 1MB Flash Memory
- Integrated low power 32-bit CPU could be used as application processor
- Standby power consumption of < 1.0mW (DTIM3)

This module has a powerful enough on-board processing and storage capability that allows it to be integrated with the sensors and other application specific devices through its GPIOs with minimal development up-front and minimal loading during runtime. Its high degree of on-chip integration allows for minimal external circuitry, including the front-end module, is designed to occupy minimal PCB area.

The ESP8266 supports APSD for VoIP applications and Bluetooth co-existence interfaces, it contains a self-calibrated RF allowing it to work under all operating conditions, and requires no external RF parts. There is an almost limitless fountain of information available for the ESP8266, all of which has been provided by amazing community support. In the *Documents* section below you will find many resources to aid you in using the ESP8266, even instructions on how to transforming this module into an IoT (Internet of Things) solution.

4.6 LCD Display

Introduction

To display interactive messages, we are using LCD Module. We examine an intelligent LCD display of two lines, 16 characters per line that is interfaced to the controllers. The protocol (handshaking) for the display is as shown. Whereas D0 to D7th bit is the Data lines, RS, RW and EN pins are the control pins and remaining pins are +5V, -5V and GND to provide supply. Where RS is the Register Select, RW is the Read Write and EN is the Enable pin. The display contains two internal byte-wide registers, one for commands (RS=0) and the second for characters to be displayed (RS=1). It also contains a user-programmed RAM area (the character RAM) that can be programmed to generate any desired character that can be formed using a dot matrix. To distinguish between these two data areas, the hex command byte 80 will be used to signify that the display RAM address 00h will be chosen. Port1 is used to furnish the command or data type, and ports 3.2 to 3.4 furnish register select and read/write levels. The display takes varying amounts of time to accomplish the functions as listed. LCD bit 7 is monitored for logic high (busy) to ensure the display is overwritten. Liquid Crystal Display also called as LCD is very helpful in providing user interface as well as for debugging purpose. The most common type of LCD controller is HITACHI 44780 which provides a simple interface between the controller & an LCD. These LCDs are very simple to interface with the controller as well as are cost effective.

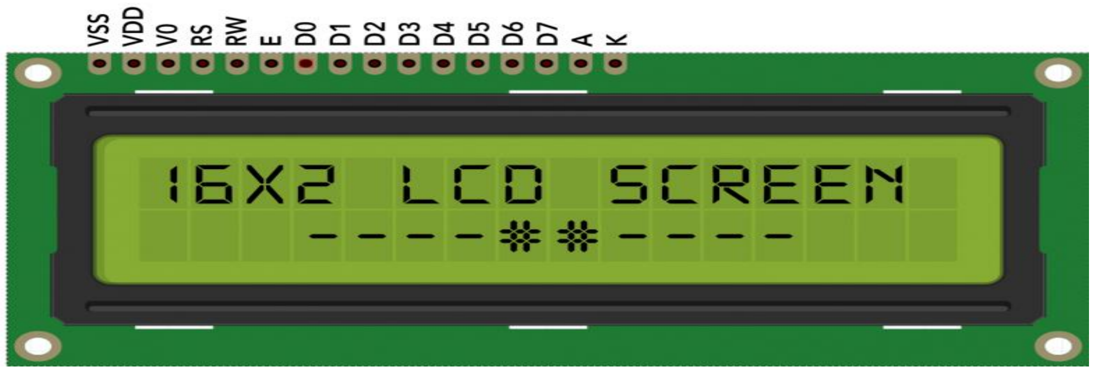
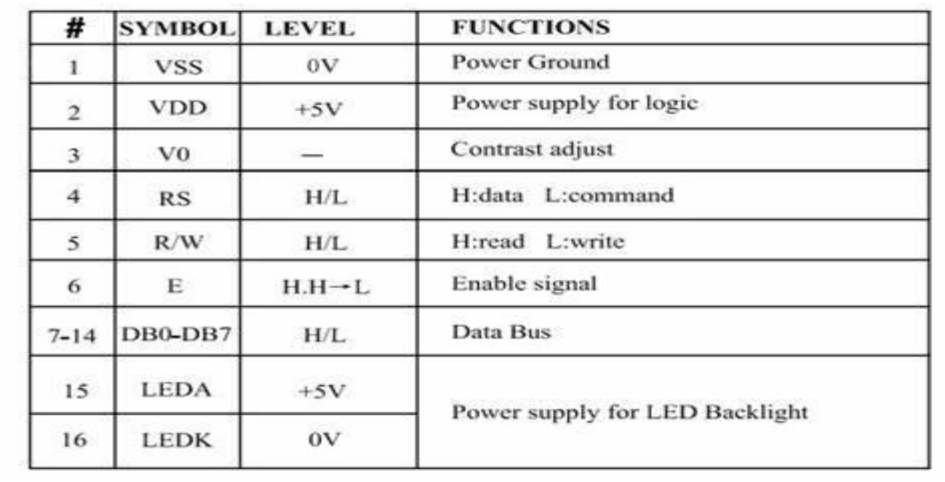


Fig. 4.16: 16 X 2 Line Alphanumeric LCD Display

The most used ALPHANUMERIC displays are 1x16 (Single Line & 16 characters), 2x16 (Double Line & 16 character per line) & 4x20 (four lines & Twenty characters per line). The LCD requires 3 control lines (RS, R/W & EN) & 8 (or 4) data lines. The number on data lines depends on the mode of operation. If operated in 8-bit more than 8 data lines + 3 control lines i.e., total 11 lines are required. And if operated in 4-bit more than 4 data lines + 3 control lines i.e., 7 lines are required. How do we decide which mode to use? It's simple if you have sufficient data lines you can go for 8-bit mode & if there is a time constrain i.e., display should be faster than we have to use 8- bit mode because basically 4-bit mode takes twice as more time as compared to 8-bit mode.



#	SYMBOL	LEVEL	FUNCTIONS
1	VSS	0V	Power Ground
2	VDD	+5V	Power supply for logic
3	V0	—	Contrast adjust
4	RS	H/L	H:data L:command
5	R/W	H/L	H:read L:write
6	E	H.H→L	Enable signal
7-14	DB0-DB7	H/L	Data Bus
15	LEDA	+5V	Power supply for LED Backlight
16	LEDK	0V	

Table 4.2: Symbols and Functions of LCD pins

<u>Operation of LCD</u>:

When RS is low (0), the data is to be treated as a command. When RS is high (1), the data being sent is considered as text data which should be displayed on the screen. When R/W is low (0), the information on the data bus is being written to the LCD. When RW is high (1), the program is effectively reading from the LCD.

Most of the times there is no need to read from the LCD so this line can directly be connected to ground thus saving one controller line. The ENABLE pin is used to latch the data present on the data pins. A HIGH - LOW signal is required to latch the data.

The LCD interprets and executes our command at the instant the EN line is brought low. If you never bring EN low, your instruction will never be executed.

4.7 ARDUINO ATMEGA328P MICROCONTROLLER

<u>Introduction to ARDUINO</u>: Arduino is a computer hardware and software company, project, and user community that designs and manufactures microcontroller kits for building digital devices and interactive objects that can sense and control objects in the physical world. The project's products are distributed as open-source hardware and software, which are licensed under the GNU Lesser General Public License (LGPL) or the GNU General Public License (GPL),[1] permitting the manufacture of Arduino boards and software distribution by anyone. Arduino boards are available commercially in preassembled form, or as do-it-yourself kits.

Arduino board designs use a variety of microprocessors and controllers. The boards are equipped with sets of digital and analog input/output (I/O) pins that may be interfaced to various expansion boards (*shields*) and other circuits. The boards feature serial communications interfaces, including Universal Serial Bus (USB) on some models, which are also used for loading programs from personal computers. The microcontrollers are typically programmed using a dialect of features from the programming languages C and C++. In addition to using traditional compiler toolchains, the Arduino project provides an integrated development environment (IDE) based on the Processing language project. The Arduino project started in 2005 as a program for students at the Interaction Design Institute Ivrea in Ivrea, Italy,[2] aiming to provide a low-cost and easy way for novices and professionals to create devices that interact with their environment using sensors and actuators. Common examples of such devices intended for beginner hobbyists include simple robots, thermostats, and detectors. The name *Arduino* comes from a bar in Ivrea, Italy, where some of the founders of the project used to meet. The bar was named after Arduino of Ivrea, who was the margrave of the March of Ivrea and King of Italy from 1002 to 1014.

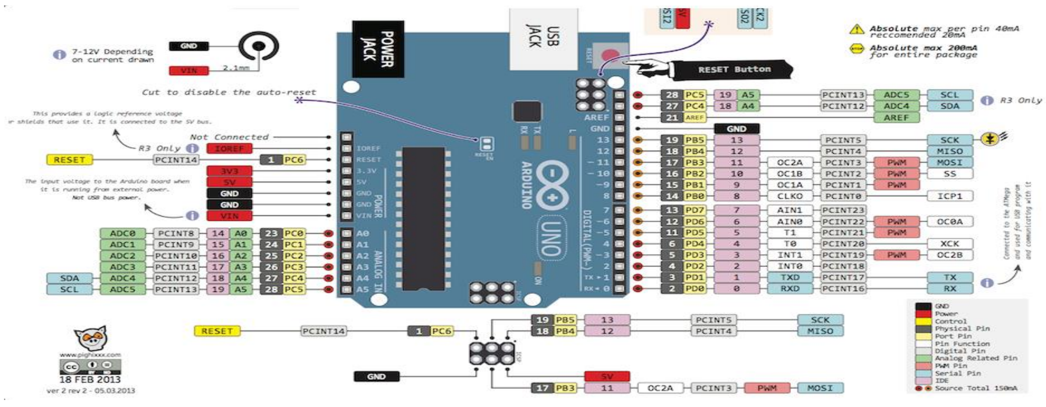


Fig. 4.17: Arduino Uno Board.

History

The origin of the Arduino project started at the Interaction Design Institute Ivrea (IDII) in Ivrea, Italy.[2] At that time, the students used a BASIC Stamp microcontroller at a cost of $100, a considerable expense for many students. In 2004, Colombian student Hernando Barragán created the development platform *Wiring* as a Master's thesis project at IDII, under the supervision of Massimo Banzi and Casey Reas, who are known for work on the Processing language. The project goal was to create simple, low-cost tools for creating digital projects by non-engineers. The Wiring platform consisted of a printed circuit board (PCB) with an ATmega168 microcontroller, an IDE based on Processing and library functions to easily program the microcontroller.[4]

In 2005, Massimo Banzi, with David Mellis, another IDII student, and David Cuartielles, added support for the cheaper ATmega8 microcontroller to Wiring. But instead of continuing the work on Wiring, they copied the Wiring source code and renamed it as a separate project, called Arduino.[4]

The initial Arduino core team consisted of Massimo Banzi, David Cuartielles, Tom Igoe, Gianluca Martino, and David Mellis,[2] but Barragán was not invited to participate.[4] Following the completion of the Wiring platform, lighter and less-expensive versions were distributed in the open-

source community.[5] Adafruit Industries, a New York City supplier of Arduino boards, parts, and assemblies, estimated in mid-2011 that over 300,000 official Arduinos had been commercially produced,[6] and in 2013 that 700,000 official boards were in users' hands

Hardware

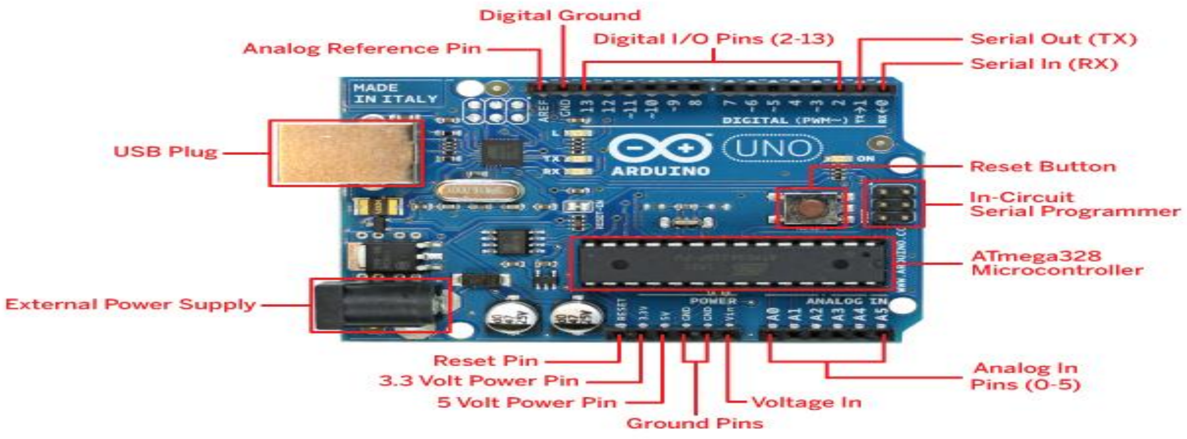


Fig.4.18: pin configuration of Arduino board.

Arduino is open-source hardware. The hardware reference designs are distributed under a Creative Commons Attribution Share-Alike 2.5 license and are available on the Arduino website. Layout and production files for some versions of the hardware are also available. The source code for the IDE is released under the GNU General Public License, version 2.[8] Nevertheless, an official Bill of Materials of Arduino boards has never been released by Arduino staff. Although the hardware and software designs are freely available under copy left licenses, the developers have requested that the name *Arduino* be exclusive to the official product and not be used for derived works without permission. The official policy document on use of the Arduino name emphasizes that the project is open to incorporating work by others into the official product.[9] Several Arduino-compatible products commercially released have avoided the project name by using various names ending in *-duino*.[10]

An Arduino board consists of an Atmel 8-, 16- or 32-bit AVR microcontroller (ATmega8, ATmega168, ATmega328, ATmega1280, ATmega2560), but other makers' microcontrollers have been used since 2015. The boards use single-row pins or female headers that facilitate connections for programming and incorporation into other circuits. These may connect with add-on modules termed *shields*. Multiple, and possibly stacked shields may be individually addressable via an I²C serial bus. Most boards include a 5 V linear regulator and a 16 MHz crystal oscillator or ceramic resonator. Some designs, such as the LilyPad, run at 8 MHz and dispense with the onboard voltage regulator due to specific form-factor restrictions.

Arduino microcontrollers are pre-programmed with a boot loader that simplifies uploading of programs to the on-chip flash memory. The default bootloader of the Aduino UNO is the optiboot bootloader.[12] Boards are loaded with program code via a serial connection to another computer. Some serial Arduino boards contain a level shifter circuit to convert between RS-232 logic levels and transistor–transistor logic (TTL) level signals. Current Arduino boards are programmed via Universal Serial Bus (USB), implemented using USB-to-serial adapter chips such as the FTDI FT232. Some boards, such as later-model Uno boards, substitute the FTDI chip with a separate AVR chip containing USB-to-serial firmware, which is reprogrammable via its own ICSP header. Other variants, such as the Arduino Mini and the unofficial Boarduino, use a detachable USB-to-serial adapter board or cable, Bluetooth or other methods, when used with traditional microcontroller tools instead of the Arduino IDE, standard AVR in-system programming (ISP) programming is used.

The Arduino board exposes most of the microcontroller's I/O pins for use by other circuits. The *Diecimila*,[a] *Duemilanove*,[b] and current *Uno*[c] provide 14 digital I/O pins, six of which can produce pulse-width modulated signals, and six analog inputs, which can also be used as six digital I/O pins. These pins are on the top of the board, via female 0.1-inch (2.54 mm) headers. Several plug-in application shields are also commercially available.

The Arduino Nano, and Arduino-compatible Bare Bones Board[13] and Boarduino[14] boards may provide male header pins on the underside of the board that can plug into solder less bread boards. Many Arduino-compatible and Arduino-derived boards exist. Some are functionally equivalent to an Arduino and can be used interchangeably. Many enhance the basic Arduino by adding output drivers, often for use in school-level education, to simplify making buggies and small robots. Others are electrically equivalent but change the form factor, sometimes retaining compatibility with shields, sometimes not. Some variants use different processors, of varying compatibility.

Software development

A program for Arduino may be written in any programming language for a compiler that produces binary machine code for the target processor. Atmel provides a development environment for their microcontrollers, AVR Studio and the newer Atmel Studio.

The Arduino project provides the Arduino integrated development environment (IDE), which is a cross-platform application written in the programming language Java. It originated from the IDE for the languages *Processing* and *Wiring*. It includes a code editor with features such as text cutting and pasting, searching and replacing text, automatic indenting, brace matching, and syntax highlighting, and provides simple *one-click* mechanisms to compile and upload programs to an Arduino board. It also contains a message area, a text console, a toolbar with buttons for common functions and a hierarchy of operation menus.

A program written with the IDE for Arduino is called a *sketch*.[40] Sketches are saved on the development computer as text files with the file extension. ino. Arduino Software (IDE) pre-1.0 saved sketches with the extension. pde. The Arduino IDE supports the languages C and C++ using special rules of code structuring.

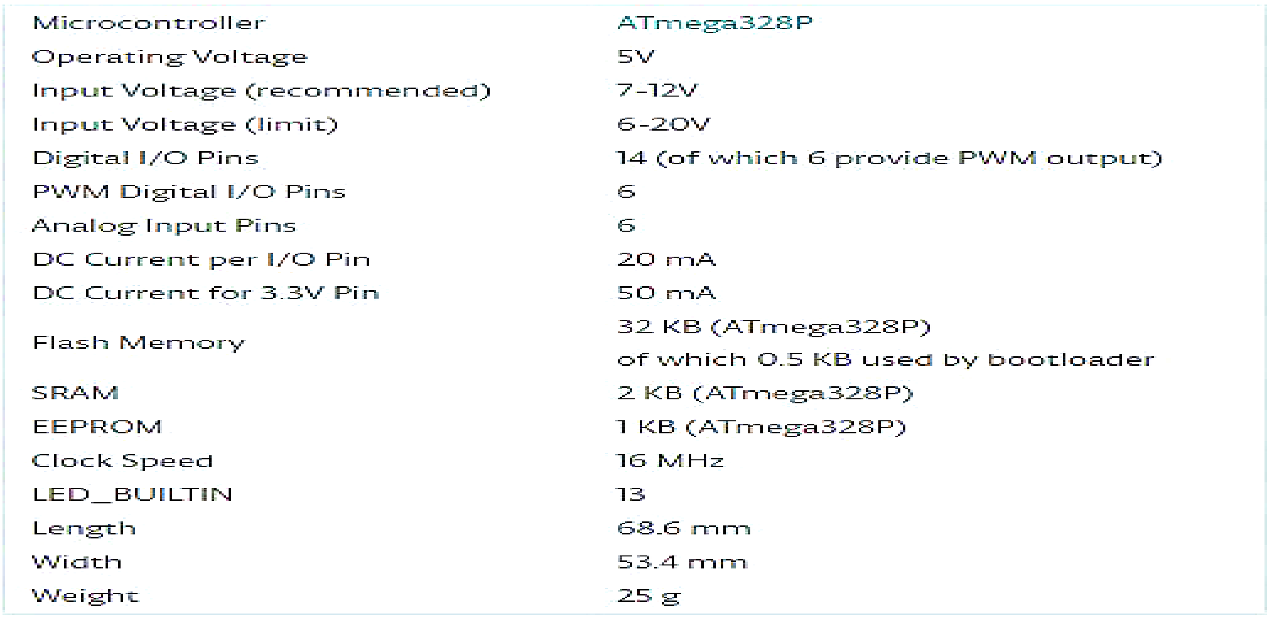
The Arduino IDE supplies a software library from the Wiring project, which provides many common input and output procedures. User-written code only requires two basic functions, for starting the sketch and the main program loop, that are compiled and linked with a program stub *main()* into an executable cyclic executive program with the GNU toolchain, also included with the IDE distribution. The Arduino IDE employs the program *avrdude* to convert the executable code into a text file in hexadecimal encoding that is loaded into the Arduino board by a loader program in the board's firmware.

Applications

- Xoscillo, an open-source oscilloscope
- Arduinome, a MIDI controller device that mimics the Monome
- OBDuino, a trip computer that uses the on-board diagnostics interface found in most modern cars
- Ardupilot, drone software and hardware

- Gameduino, an Arduino shield to create retro 2D video games
- Arduino Phone, a do-it-yourself cellphone
- Water quality testing platform
- Automatic titration system based on Arduino and stepper motor
- Low-cost data glove for virtual reality applications
- Impedance sensor system to detect bovine milk adulteration
- Homemade CNC using Arduino and DC motors with close loop control by Homofaciens
- DC motor control using Arduino and H-Bridge

Technical specsifications:



Microcontroller	ATmega328P
Operating Voltage	5V
Input Voltage (recommended)	7-12V
Input Voltage (limit)	6-20V
Digital I/O Pins	14 (of which 6 provide PWM output)
PWM Digital I/O Pins	6
Analog Input Pins	6
DC Current per I/O Pin	20 mA
DC Current for 3.3V Pin	50 mA
Flash Memory	32 KB (ATmega328P) of which 0.5 KB used by bootloader
SRAM	2 KB (ATmega328P)
EEPROM	1 KB (ATmega328P)
Clock Speed	16 MHz
LED_BUILTIN	13
Length	68.6 mm
Width	53.4 mm
Weight	25 g

Table 4.3: Technical specifications of Atmega328p

Programming

The Arduino/Genuino Uno can be programmed with the (Arduino Software (IDE)). Select "Arduino/Genuino Uno from the Tools > Board menu (according to the microcontroller on your board). For details, see the reference and tutorials. The ATmega328 on the Arduino/Genuino Uno comes preprogrammed with a bootloader that allows you to upload new code to it without the use of an external hardware programmer. It communicates using the original STK500 protocol (reference, C header files).You can also bypass the bootloader and program the microcontroller through the ICSP (In-Circuit Serial Programming) header using Arduino ISP or similar; see these instructions fordetails. The ATmega16U2 (or 8U2 in the rev1 and rev2 boards) firmware source code is available in the Arduino repository. The ATmega16U2/8U2 is loaded with a DFU bootloader, which can be activated by:

- On Rev1 boards: connecting the solder jumper on the back of the board (near the map of Italy) and then rese ing the 8U2.
- On Rev2 or later boards: there is a resistor that pulling the 8U2/16U2 HWB line to ground, making it easier to put into DFU mode.

- You can then use Atmel's FLIP software (Windows) or the DFU programmer (Mac OS X and Linux) to load a new firmware. Or you can use the ISP header with an external programmer (overwriting the DFU bootloader). See this user-contributed tutorial for more information

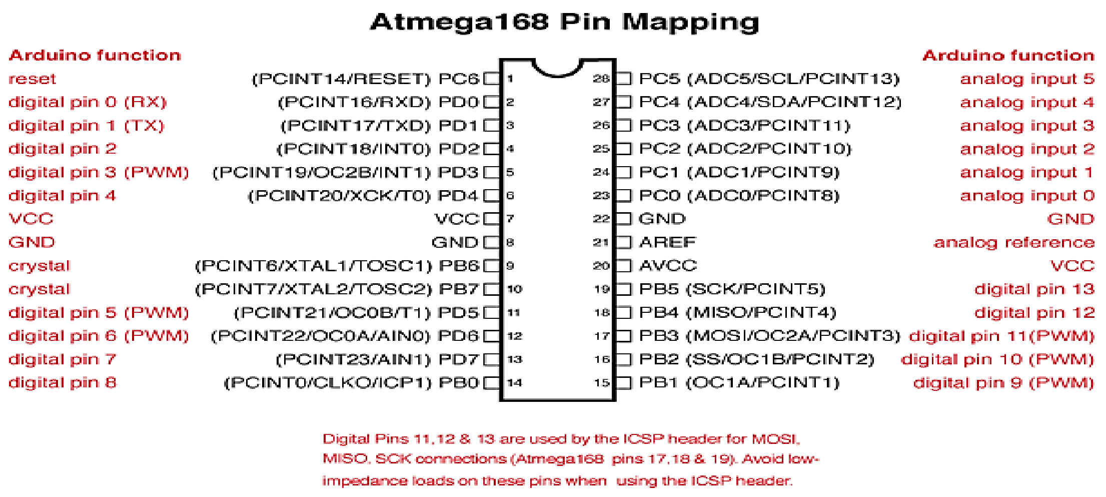


Fig.4.19: Pin Configuration of Arduino Atmega328p/168 Microcontroller

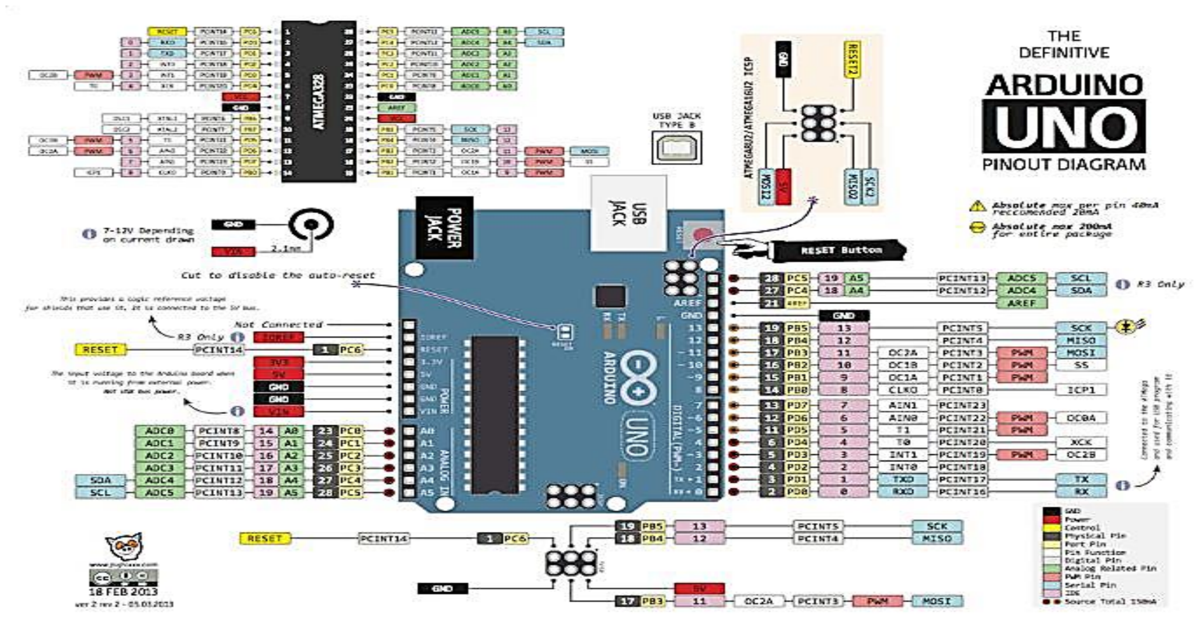


Fig.4.20: Pin Diagram of Arduino Uno Board.

Warnings

The Arduino/Genuino Uno has a resettable polyfused that protects your computer's USB ports from shorts and overcurrent. Although most computers provide their own internal protection, the fuse provides an extra layer of protection. If more than 500 mA is applied to the USB port, the fuse will automatically break the connection until the short or overload is removed.

Differences with other boards

The Uno differs from all preceding boards in that it does not use the FTDI USB-to-serial driver chip. Instead, it features the Atmega16U2 (Atmega8U2 up to version R2) programmed as a USB-to-serial converter.

Power

The Arduino/Genuino Uno board can be powered via the USB connection or with an external power supply. The power source is selected automatically. External (non-USB) power can come either from an AC-to-DC adapter (wall-wart) or battery. The adapter can be connected by plugging a 2.1mm center-positive plug into the board's power jack. Leads from a battery can be inserted in the GND and Vin pin headers of the POWER connector.

The board can operate on an external supply from 6 to 20 volts. If supplied with less than 7V, however, the 5V pin may supply less than five volts and the board may become unstable. If using more than 12V, the voltage regulator may overheat and damage the board. The power pins are as follows:

- Vin. The input voltage to the Arduino/Genuino board when it's using an external power source (as opposed to 5 volts from the USB connection or other regulated power source). You can supply voltage through this pin, or, if supplying voltage via the power jack, access it through this pin.
- 5V. This pin outputs a regulated 5V from the regulator on the board. The board can be supplied with power either from the DC power jack (7 - 12V), the USB connector (5V), or the VIN pin of the board (7-12V). Supplying voltage via the 5V or 3.3V pins bypasses the regulator, and can damage your board. We don't advise it.
- 3V3. A 3.3-volt supply generated by the on-board regulator. Maximum current draw is 50 mA.
- GND. Ground pins.
- IOREF. This pin on the Arduino/Genuino board provides the voltage reference with which the microcontroller operates. A properly configured shield can read the IOREF pin voltage and

select the appropriate power source or enable voltage translators on the outputs to work with the 5V or 3.3V.

Memory: The ATmega328 has 32 KB (with 0.5 KB occupied by the bootloader). It also has 2 KB of SRAM and 1 KB of EEPROM (which can be read and written with the EEPROM library).

Input and Output: See the mapping between Arduino pins and ATmega328P ports. The mapping for the Atmega8, 168, and 328 is identical. Each of the 14 digital pins on the Uno can be used as an input or output, using pinMode(), digitalWrite(), and digitalRead() functions. They operate at 5 volts. Each pin can provide or receive 20 mA as recommended operating condition and has an internal pull-up resistor (disconnected by default) of 20-50k ohm. A maximum of 40mA is the value that must not be exceeded on any I/O pin to avoid permanent damage to the microcontroller.

- Serial: 0 (RX) and 1 (TX). Used to receive (RX) and transmit (TX) TTL serial data. These pins are connected to the corresponding pins of the ATmega8U2 USB-to-TTL Serial chip.
- External Interrupts: 2 and 3. These pins can be configured to trigger an interrupt on a low value, a rising or falling edge, or a change in value. See the attachInterrupt() function for details.
- PWM: 3, 5, 6, 9, 10, and 11. Provide 8-bit PWM output with the analogWrite() function.
- SPI: 10 (SS), 11 (MOSI), 12 (MISO), 13 (SCK). These pins support SPI communication using the SPI library.
- LED: 13. There is a built-in LED driven by digital pin 13. When the pin is HIGH value, the LED is on, when the pin is LOW, it's off.
- TWI: A4 or SDA pin and A5 or SCL pin. Support TWI communication using the Wire library.
- The Uno has 6 analog inputs, labeled A0 through A5, each of which provide 10 bits of resolution (i.e., 1024 different values). By default they measure from ground to 5 volts, though is it possible to change the upper end of their range using the AREF pin and the analogReference() function. There are a couple of other pins on the board:
- AREF. Reference voltage for the analog inputs. Used with analogReference().
- Reset. Bring this line LOW to reset the microcontroller. Typically used to add a reset button to shields which block the one on the board.

Communication

Arduino/Genuino Uno has a number of facilities for communicating with a computer, another Arduino/Genuino board, or other microcontrollers. The ATmega328 provides UART TTL (5V) serial communication, which is available on digital pins 0 (RX) and 1 (TX). An ATmega16U2 on the board channels this serial communication over USB and appears as a virtual com port to software on the computer. The 16U2 firmware uses the standard USB COM drivers, and no external driver is needed. However, on Windows, a .inf file is required. The Arduino Software (IDE) includes a serial monitor which allows simple textual data to be sent to and from the board. The RX and TX LEDs on the board will flash when data is being transmitted via the USB-to-serial chip and USB connection to the computer (but not for serial communication on pins 0 and 1). A Software Serial library allows serial communication on any of the Uno's digital pins. The ATmega328 also supports I2C (TWI) and SPI communication. The Arduino Software (IDE) includes a Wire library to simplify use of the I2C bus; see the documentation for details. For SPI communication, use the SPI library.

Revisions: Revision 3 of the board has the following new features:1.0 pin out: added SDA and SCL pins that are near to the AREF pin and two other new pins placed near to the RESET pin, the IOREF that allow the shields to adapt to the voltage provided from the board. In future, shields will be compatible with both the board that uses the AVR, which operates with 5V and with the Arduino Due that operates with 3.3V. The second one is a not connected pin, that is reserved for future purposes.

- Stronger RESET circuit.
- ATMEGA 16U2 replace the 8U2.

CHAPTER 5

5. SOFTWARE DESCRIPTION

5.1 ARDUINO IDE:

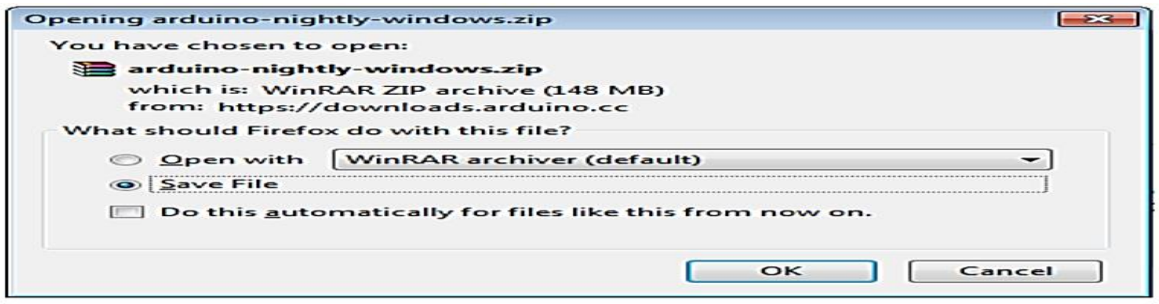
5.1.1 Software introduction:

Arduino IDE (Integrated Development Environment) is an open-source software platform used to write and upload code to Arduino boards. It provides a user-friendly interface for writing and editing code, as well as for uploading it to the Arduino board.

Arduino IDE is available for Windows, Mac, and Linux operating systems and can be downloaded from the Arduino website for free. Arduino IDE supports a programming language based on Wiring, a simplified version of C/C++. The IDE includes a text editor, a compiler, a debugger, and a serial monitor, which allows you to interact with the Arduino board and view its output. It also provides a library manager, which allows you to easily add and manage external libraries. With Arduino IDE, you can write code to control various components and sensors, such as LEDs, motors, temperature sensors, and more. The IDE provides a wide range of examples and tutorials to help you get started with programming and using Arduino boards. Additionally, the community of Arduino users is active and helpful, making it easy to find support and guidance when needed.

5.1.2 Steps for Running the Arduino IDE:

Arduino IDE Software. You can get different versions of Arduino IDE from the Download page on the Arduino Official website. You must select your software, which is compatible with your operating system (Windows, IOS, or Linux). After your file download is complete, unzip the file.


Opening arduino-nightly-windows.zip

You have chosen to open:

arduino-nightly-windows.zip

which is: WinRAR ZIP archive (148 MB)

from: https://downloads.arduino.cc

What should Firefox do with this file?

Open with: WinRAR archiver (default)

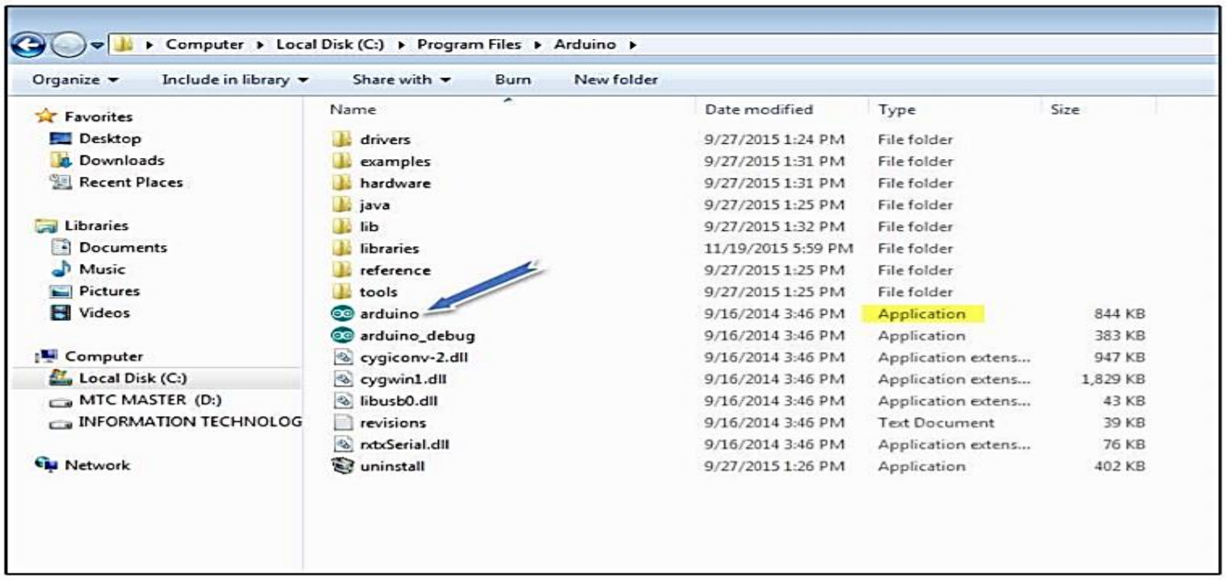
[✓ Save File]

Do this automatically for files like this from now on.

Fig. 5.1: Opening arduino-nightly-windows.zip

Launch Arduino IDE. After your Arduino IDE software is downloaded, you need to unzip the folder. Inside the folder, you can find the application icon with an infinity label (application.exe). Double-click the icon to start the IDE.

Launching of Arduino IDE is shown in below figure:



Name	Date modified	Type	Size
drivers	9/27/2015 1:24 PM	File folder	
examples	9/27/2015 1:31 PM	File folder	
hardware	9/27/2015 1:31 PM	File folder	
java	9/27/2015 1:25 PM	File folder	
lib	9/27/2015 1:32 PM	File folder	
libraries	11/19/2015 5:59 PM	File folder	
reference	9/27/2015 1:25 PM	File folder	
tools	9/27/2015 1:25 PM	File folder	
arduino	9/16/2014 3:46 PM	Application	844 KB
arduino_debug	9/16/2014 3:46 PM	Application	383 KB
cygiconv-2.dll	9/16/2014 3:46 PM	Application extens...	947 KB
cygwin1.dll	9/16/2014 3:46 PM	Application extens...	1,829 KB
libusb0.dll	9/16/2014 3:46 PM	Application extens...	43 KB
revisions	9/16/2014 3:46 PM	Text Document	39 KB
rxtxSerial.dll	9/16/2014 3:46 PM	Application extens...	76 KB
uninstall	9/27/2015 1:26 PM	Application	402 KB

Fig:5.2: Launch Arduino IDE

Open your first project. Once the software starts, you have two options:

Create a new project.

Open an existing project example.

To create a new project, select File --> New

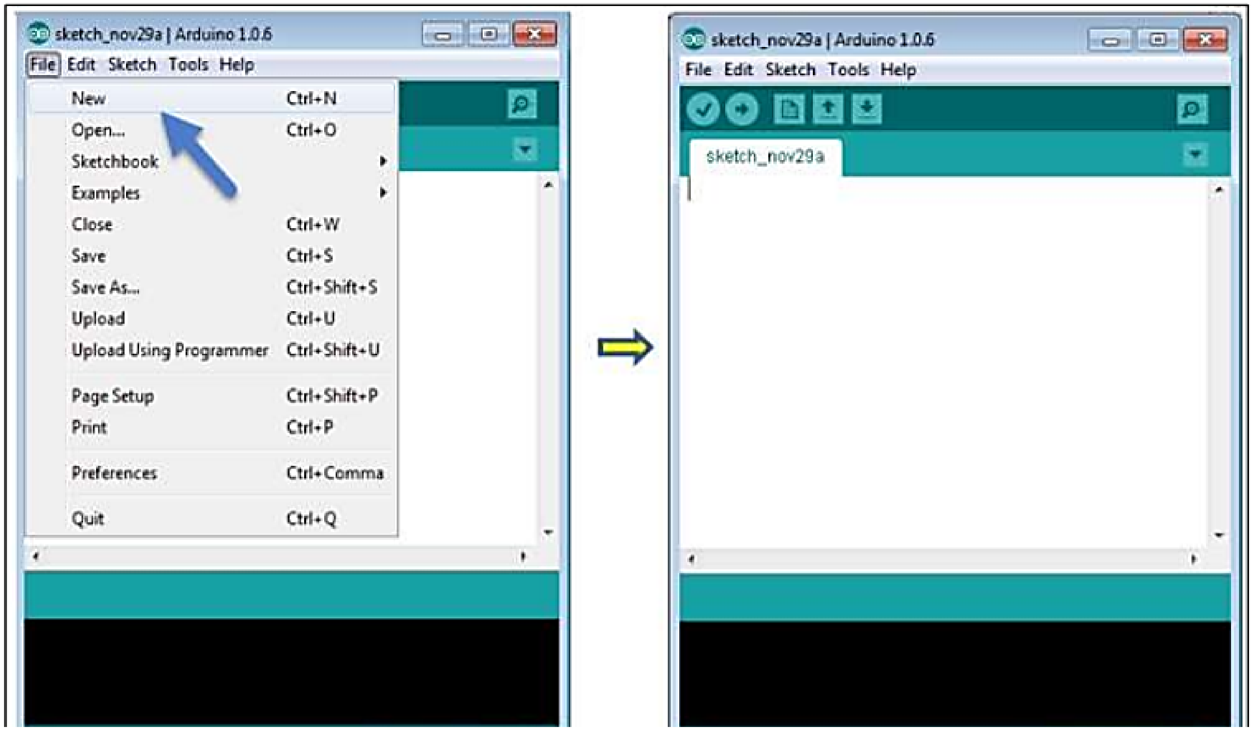
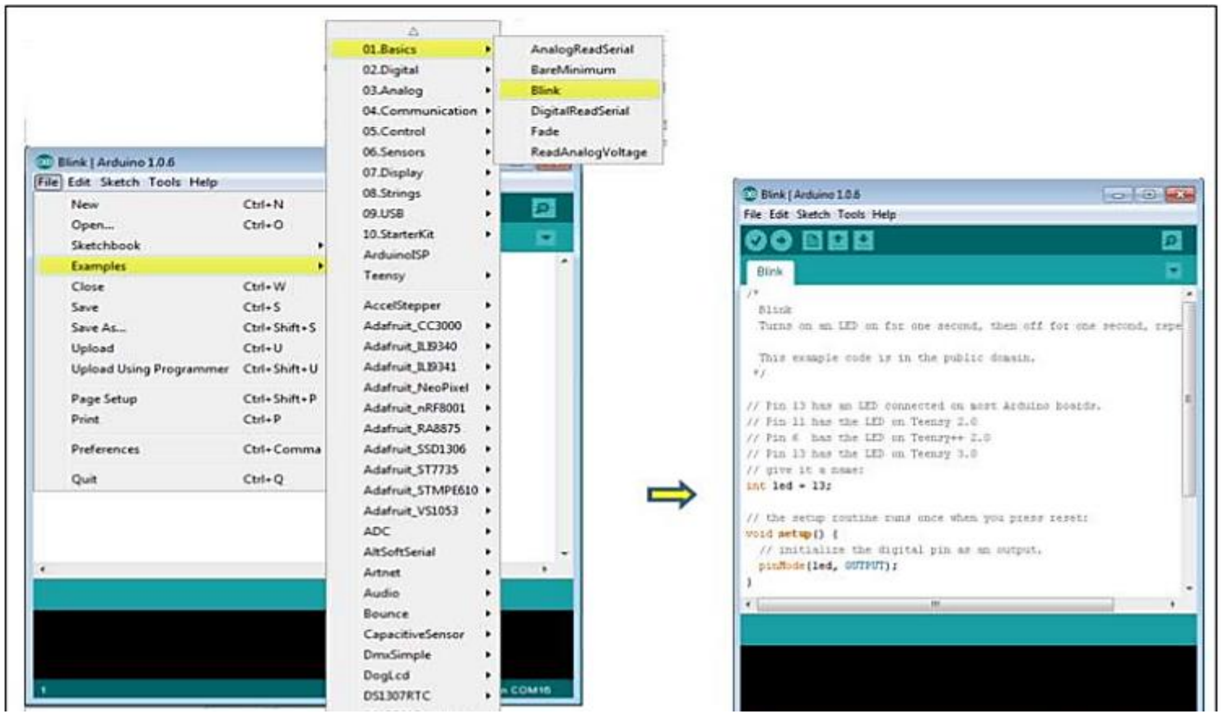


Fig:5.3: Create a new project

The interface of an Arduino IDE when we open existing project is shown in below figure:



```
/*
  Blink
  Turns on an LED on for one second, then off for one second, repe

  This example code is in the public domain.
 */

// Pin 13 has an LED connected on most Arduino boards.
// Pin 11 has the LED on Teensy 2.0
// Pin 6  has the LED on Teensy++ 2.0
// Pin 13 has the LED on Teensy 3.0
// give it a name:
int led = 13;

// the setup routine runs once when you press reset:
void setup() {
  // initialize the digital pin as an output.
  pinMode(led, OUTPUT);
}
```

Fig: 5.4: Open an existing project example

Here, we are selecting just one of the examples with the name Blink. It turns the LED on and off with some time delay. You can select any other example from the list Select your serial port. Select the serial device of the Arduino board. Go to Tools -> Serial Port menu. This is likely to be COM3 or higher (COM1 and COM2 are usually reserved for hardware serial ports). To find out, you can disconnect your Arduino board and re-open the menu, the entry that disappears should be of the Arduino board. Reconnect the board and select that serial port.

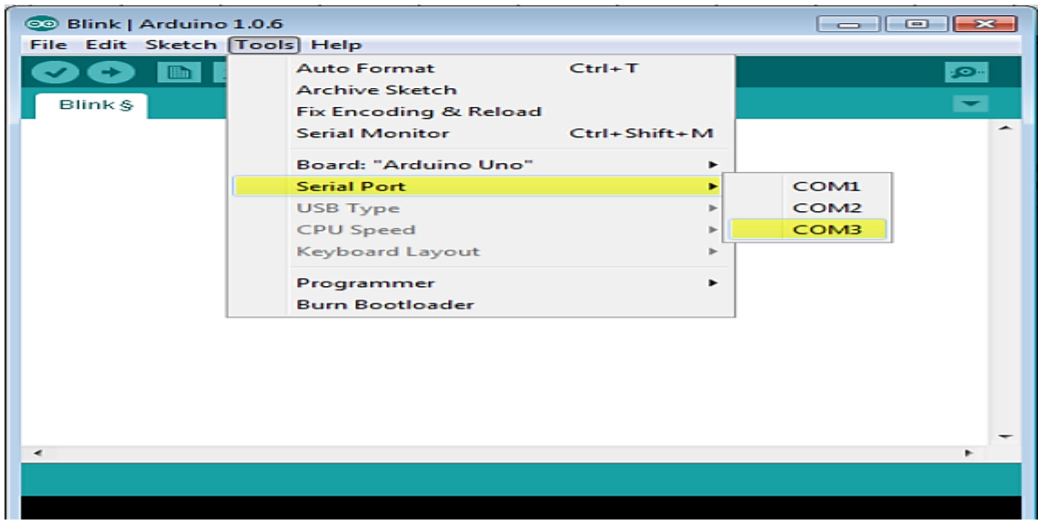


Fig: 5.5: Select your serial port

Before explaining how we can upload our program to the board, we must demonstrate the function of each symbol appearing in the Arduino IDE toolbar.

A- Used to check if there is any compilation error.

B- Used to upload a program to the Arduino board.

C- Shortcut used to create a new sketch.

D- Used to directly open one of the example sketches.

E- Used to save your sketch.

F- Serial monitor used to receive serial data from the board and send the serial data to the board.

Now, simply click the "Upload" button in the environment.

Wait a few seconds; you will see the RX and TX LEDs on the board, flashing. If the upload is successful, the message "Done uploading" will appear in the status bar.

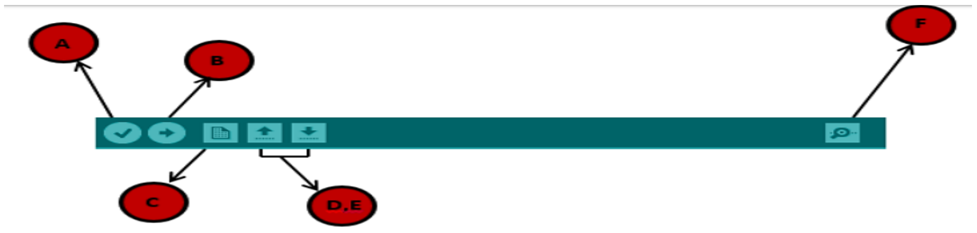
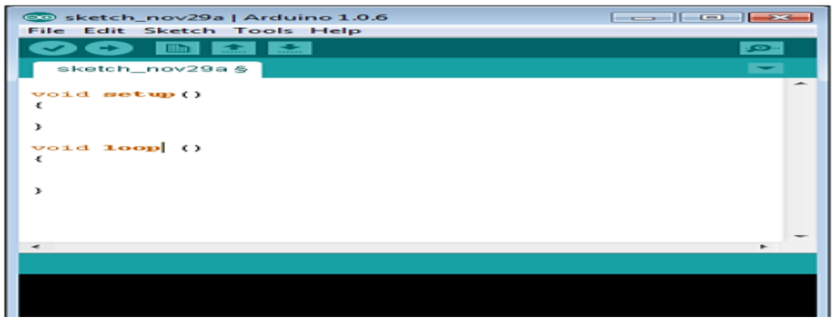


Fig 5.6: function of each symbol appearing in the Arduino IDE toolbar

In this chapter, we will study in depth, the Arduino program structure and we will learn more new terminologies used in the Arduino world. The Arduino software is open-source. The source code for the Java environment is released under the GPL and the C/C++ microcontroller libraries are under the LGPL. Sketch: The first new terminology is the Arduino program called “sketch”. Structure Arduino programs can be divided in three main parts: Structure, Values (variables and constants), and Functions. In this tutorial, we will learn about the Arduino software program, step by step, and how we can write the program without any syntax or compilation error. Let us start with the Structure. Software structure consist of two main functions:

Setup () function

Loop () function



```
void setup ()
{
}
void loop ()
{

}
```

Fig: 5.7: Bare minimum code

Data types in C refers to an extensive system used for declaring variables or functions of different types. The type of a variable determines how much space it occupies in the storage and how the bit pattern stored is interpreted. The following table provides all the data types that you will use during Arduino programming.

5.2 Embedded C language:

Embedded C is a variant of the C programming language that is used to program embedded systems, which are computer systems that are designed to perform specific functions and are typically built into other devices. Embedded C has many of the same features as standard C, but also includes additional features that are specifically designed for use in embedded systems, such as support for low-level hardware access and efficient use of memory and processing resources. In embedded systems, it's common to work with microcontrollers, which are small computer chips that contain a processor, memory, and input/output peripherals, all on a single chip. Embedded C is used to write code that controls the behavior of these microcontrollers and interfaces with external devices.

Some common features of Embedded C include:

- Support for bit manipulation operations and direct access to hardware registers.
- Ability to work with low-level interrupts, which are used to handle asynchronous events and input/output operations.
- Support for inline assembly language, which allows programmers to write code that directly accesses the processor and other hardware resources.
- Optimizations for memory and processing efficiency, which are essential in embedded systems where resources are often limited.

Embedded C is used in a wide range of applications, including automotive systems, medical devices, industrial automation, and consumer electronics. It's a powerful and versatile language that allows programmers to develop highly efficient and reliable embedded systems.

5.2.1. Embedded C language Code:

```
#include <LiquidCrystal.h>
//LiquidCrystal lcd (13, 12, 11, 10, 9, 8);
LiquidCrystal lcd (8, 9, 10, 11, 12, 13);
unsigned char rcv, count, gchr, gchr1, robos = 's';
int sti = 0;
String inputString = "";        // a string to hold incoming data
```

```
boolean stringComplete = false;  // whether the string is complete
int m11 = 7;
int m12 = 6;
int m21 = 5;
int m22 = 4;
void okcheck()
{
  unsigned char rcr;
  do {
    rcr = Serial.read();
  } while (rcr != 'K');
}
void setup() {
  lcd.begin(16, 2);
  Serial.begin(115200);
  serialEvent();
  pinMode(m11, OUTPUT);
  pinMode(m12, OUTPUT);
  pinMode(m21, OUTPUT);
  pinMode(m22, OUTPUT);
  digitalWrite(m11, HIGH);
  digitalWrite(m12, HIGH);
  digitalWrite(m21, HIGH);
  digitalWrite(m22, HIGH);
  lcd.setCursor(0, 0);
  lcd.print("WELCOME TO THE");
  lcd.setCursor(4, 1);
  lcd.print("PROJECT");
  delay(1000);
  lcd.clear();
  lcd.setCursor(0, 0);
  lcd.print("DISH ANTENA POS");
  lcd.setCursor(0, 1);
  lcd.print("CTRL SYS ON IOT");
  delay(3000);
  lcd.clear();
  lcd.print("Wifi init");
  Serial.write("AT\r\n");          delay(500); okcheck();
  Serial.write("ATE0\r\n");         okcheck();
  Serial.write("AT+CWMODE=3\r\n"); delay(500);
  Serial.write("AT+CIPMUX=1\r\n"); delay(500);        okcheck();
  Serial.write("AT+CIPSERVER=1,23\r\n");     okcheck();
  lcd.clear();
  lcd.print("Waiting For");
  lcd.setCursor(0, 1);
  lcd.print("Connection");
  do {
    rcv = Serial.read();
  } while (rcv != 'C');
  lcd.clear();
  lcd.print("Connected");
  delay(2000);
  lcd.clear();
}
void loop() {
  lcd.setCursor(0, 0);
  lcd.print("ready to recive cmd"); //4,0
  lcd.setCursor(0, 0);
  lcd.print("                    "); //4,0
  delay(1000);
  if (stringComplete)
  {
    if (gchr == '1')
    {
      gchr = 'x';
      robos = 'm';
```

```
   lcd.setCursor(0, 0);
   lcd.print("CMD RCVD : 1");
   lcd.setCursor(0, 1);
   lcd.print("+10°E ROTATED");
   digitalWrite(m11, HIGH);
   digitalWrite(m12, LOW);  delay(2000);
   digitalWrite(m11, HIGH);
   digitalWrite(m12, HIGH);
   Serial.write("AT+CIPSEND=0,14\r\n"); delay(500);
   Serial.write("+10°E ROTATED\r\n"); delay(500);    lcd.clear();
  }
  if (gchr == '2')
  {
   gchr = 'x';
   robos = 'm';
   lcd.setCursor(0, 0);
   lcd.print("CMD RCVD : 2");
   lcd.setCursor(0, 1);
   lcd.print("+10°W ROTATED");
   digitalWrite(m11, LOW);
   digitalWrite(m12, HIGH);  delay(2000);
   digitalWrite(m11, HIGH);
   digitalWrite(m12, HIGH);
   Serial.write("AT+CIPSEND=0,14\r\n"); delay(500);
   Serial.write("+10°W ROTATED\r\n"); delay(500);    lcd.clear();
  }
  if (gchr == '3')
  {
   gchr = 'x';
   robos = 'm';
   lcd.setCursor(0, 0);
   lcd.print("CMD RCVD : 3");
   lcd.setCursor(0, 1);
```

```
      lcd.print("+10°N ROTATED");
      digitalWrite(m21, LOW);
      digitalWrite(m22, HIGH);  delay(2000);
      digitalWrite(m21, HIGH);
      digitalWrite(m22, HIGH);
      Serial.write("AT+CIPSEND=0,14\r\n"); delay(500);
      Serial.write("+10°N ROTATED\r\n"); delay(500);    lcd.clear();
    }
    if (gchr == '4')
    {
      gchr = 'x';
      robos = 'm';
      lcd.setCursor(0, 0);
      lcd.print("CMD RCVD : 4");
      lcd.setCursor(0, 1);
      lcd.print("+10°S ROTATED");
      digitalWrite(m21, HIGH);
      digitalWrite(m22, LOW);  delay(2000);
      digitalWrite(m21, HIGH);
      digitalWrite(m22, HIGH);
      Serial.write("AT+CIPSEND=0,14\r\n"); delay(500);
      Serial.write("+10°S ROTATED\r\n"); delay(500);    lcd.clear();
    }
    inputString = "";
    stringComplete = false;
  }
}
void serialEvent()
{
  while (Serial.available())
  {
    char inChar = (char)Serial.read();
    if (inChar == '*')
    { sti = 1;
      inputString += inChar;
    }
    if (sti == 1)
    {
      inputString += inChar;
    }
    if (inChar == '#')
    { sti = 0;
      gchr = inputString[2];
      stringComplete = true;
    }
  }
}
```

<u>**5.3 MOBILE TELNET**</u>**:**

<u>**Introduction:**</u>

Mobile Telnet is a mobile application that allows users to access remote computers and devices through Telnet protocol. Telnet is a network protocol used to establish a remote connection to a device, such as a server, router, or switch, and access its command-line interface (CLI).

Mobile Telnet applications are available for both iOS and Android devices and can be downloaded from their respective app stores. Once installed, users can connect to remote devices using their IP address or domain name and login credentials.

Mobile Telnet applications provide a terminal emulator that displays the CLI of the remote device, allowing users to issue commands and receive output as if they were directly connected to the device. This can be useful for system administrators, network engineers, or developers who need to access and manage remote devices while on-the-go.

Some Mobile Telnet applications also support SSH (Secure Shell) protocol, which provides a secure encrypted connection between the client and the remote device, ensuring that sensitive information is not intercepted or compromised during the remote session.

Telnet is a network protocol used to establish a connection to a remote server or device over the internet. While telnet was originally designed for use on desktop and server computers, there are mobile versions available for smartphones and tablets.

To use telnet on a mobile device, you'll need to download a telnet client app from the app store. There are many different telnet client apps available for both iOS and Android, so you'll want to choose one that meets your needs and is compatible with your device. Once you've downloaded and installed the telnet client app, you can use it to connect to a remote server or device by entering the server's IP address or hostname, as well as the port number, username, and password (if required). From there, you can use telnet commands to interact with the remote device or server, just as you would on a desktop computer. It's worth noting that telnet is an unencrypted protocol, which means that any data transmitted over the connection is sent in plain text and could potentially be intercepted by a third party. As a result, telnet is not considered a secure method for remote access and is often replaced by more secure protocols such as SSH.

CHAPTER 6

6. EXPERIMENTAL RESULTS

IoT-based antenna positioning systems allow for remote control of antennas in real-time. These systems can automatically adjust antenna direction, tilt, and rotation to ensure optimal signal strength and coverage. IoT-based antenna positioning systems can be used in a variety of applications, including smart cities, agriculture, and transportation.

Some potential benefits of IoT-based antenna positioning systems include:

1. Improved signal range: By continuously adjusting antenna direction and tilt, these systems can ensure a stronger and more reliable signal.
2. Reduced maintenance costs: Since these systems can be remotely controlled and automatically adjusted, they can reduce the need for manual maintenance and repairs.
3. Increased efficiency: By optimizing antenna positioning, IoT-based systems can increase the efficiency of wireless communication systems and reduce energy consumption.

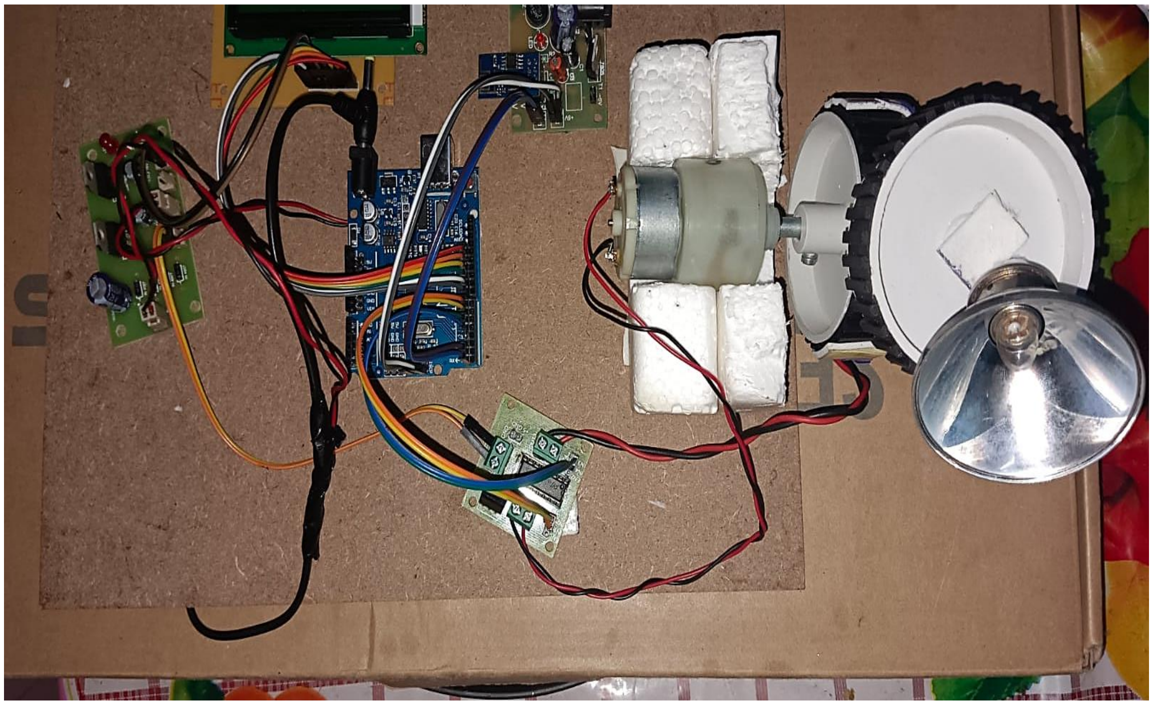
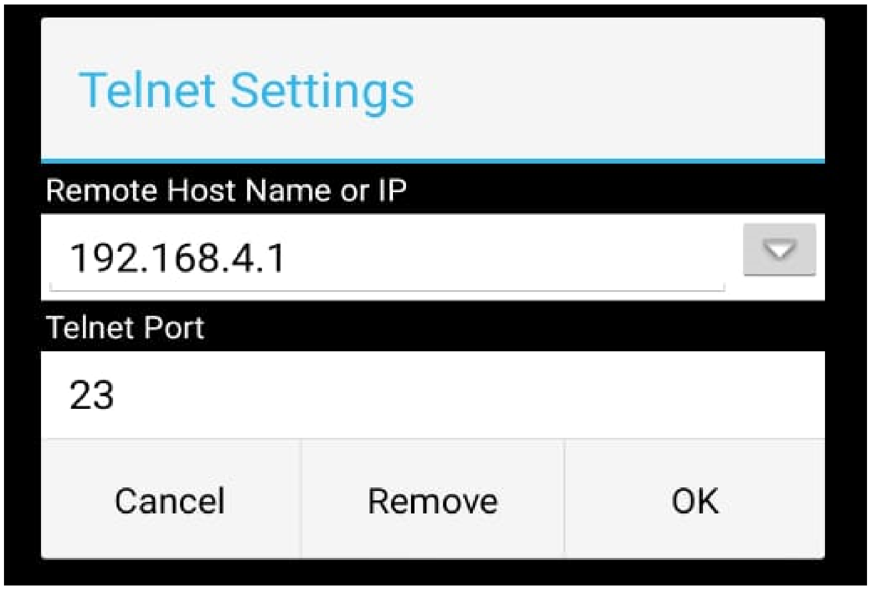


Fig. 6.1: Proposed system Kit

IoT-based antenna positioning systems have the potential to improve wireless communication systems and offer a range of benefits for different applications. However, the effectiveness of these systems may vary depending on factors such as the quality of the underlying wireless network and the accuracy of the positioning algorithm

Interface of Mobile telnet is shown below



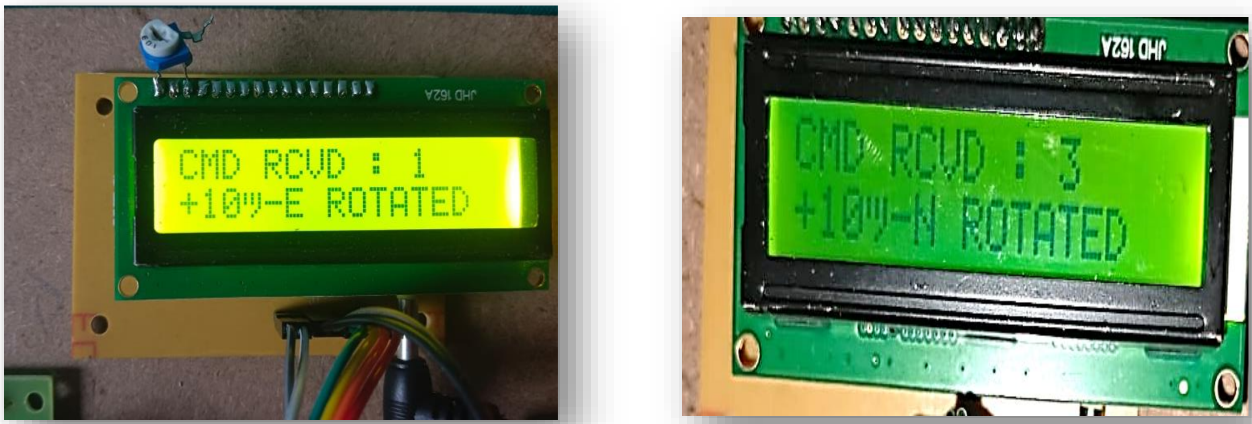
Telnet Settings

Remote Host Name or IP: 192.168.4.1

Telnet Port: 23

Fig. 6.2: Interface of Mobile Telnet

the output of our project is displayed on LCD in figure



```
CMD RCVD : 1
+10°-E ROTATED
```

```
CMD RCVD : 3
+10°-N ROTATED
```

Fig. 6.3: Output displaying on LCD display

CHAPTER 7

7. CONCLUSION & FUTURE SCOPE

CONCLUSION

As the existing system contains the Bluetooth module that can facilitate to change the position of antenna only if the user is available within the 10m of range hardly. If the user is out of this range the positioning is not possible. The proposed system contains a Wi-Fi module that allows user to change the position of antenna if required, within 45 meters of range which is more flexible.

FUTURE SCOPE

A system that includes an antenna with an ability of changing its position automatically, whenever the signal strength is poor may also be seen in future, that will make the consumers effortless to get the good signal.

CHAPTER 8

8. REFERENCES

REFERENCES

[1]. Amritha A. S, Divya sree M V, Jesna Prem, Kavya sree S.M. and Keerthana Vasu (2015). Microcontroller Based Wireless 3D Position Control for Antenna. International Journal of Science and Research (IJSR) , 6(4) : 924 – 927

[2]. Akor, P.A. Enokela, J. A. Agbo, D. O. An Implementation of A Gsm Controlled Security Robotic Vehicle, American Journal of Engineering Research (AJER), 6(12):130 - 135

[3]. BekirSaitCiftler & Abdullah Kadri,(2017) "IoT Localization for Bistatic Passive UHF RFID Systems With 3-D Radiation Pattern" ,published in IEEE,Vol no4,pp 905-913,August 2017

[4]. Daqiang Zhang, Laurence T. Yang, Hong yu Huang,(2011) "Searching in Internet of Things: Vision and Challenges", Ninth IEEE International Symposium on Parallel and Distributed Processing with Applications,. .

[5]. http://www.instructables.com

[6]. http://en.wikipedia.org

[7]. Louis Coetzee, Johan Eksteen ,(2011) "The Internet of Things – Promise for the Future? An Introduction ", IST-Africa 2011 Conference Proceedings Paul Cunningham and Miriam Cunningham (Eds) IIMC International Information Management Corporation, ISBN:9781905824-24-3.

[8]. Maske, S.A. Shelake, A. V. Shinde, A. S. and Mugade, N.K. (2017). Dish Positioning by Using IR Remote. International Research Journal of Engineering and Technology (IRJET), 4(6) : 266 – 269

[9]. Pooja, R. Shradha, S. Komal, S. Priyanka, T. Akshata,R. Implementing an IOT based Antenna Positioning System, International Journal for Research in Applied Science & Engineering Technology (IJRASET), 6(4): 2898 -2900

[10]. Prajwal, B. Pranjal, G. and Preeti, P. (2015). Remote Alignment of Dish Positioning by Android Application. International Journal of Engineering Research & Management Technology, 2(2): 267 - 269

[11]. Rahul, M. Shubham, K. Sunil, S. and Tina, R. (2017). Home automation using Wi-Fi. International Journal of Research In Science & Engineering: 181 - 185

[12]. Rahane, S. D. Mhaske, S. A. Shin gate, S.R. (2018). Design of Advanced Antenna

Positioning System International Journal of Research in Advent Technology, 6(3):251 – 253

[13]. Surya, D.C. Pankaj, R. Arvind, K. and Irshad, A. (2014). microcontroller based wireless automatic antenna positioning system International Journal of Electronics, Electrical and Computational System, 3(6):12 - 26

[14]. Yinghui Huang, Guanyu Li,” Descriptive Models for Internet of Things”, International Conference on Intelligent Control and Information Processing, August2010 - Dalian, China