A

MAJOR PROJECT REPORT ON

VOICE CONTROLLED PICK AND PLACE ROBOT

Submitted in partial fulfilment of the requirement for the award of degree of

BACHELOR OF TECHNOLOGY

IN

ELECTRONICS AND COMMUNICATION ENGINEERING

SUBMITTED BY

TASMMISETTI. HARI CHANDRA PRASAD	**218R1A04C0**
TANNEERU MOUNIKA	**218R1A04C1**
VAJEERA PRANATHI	**218R1A04C2**
VEERABOINA TEJASHWINI	**218R1A04C3**

Under the Esteemed Guidance of

Dr. M. AMRU
professor

DEPARTMENT OF ELECTRONICS & COMMUNICATION ENGINEERING

CMR ENGINEERING COLLEGE

UGC AUTONOMOUS

(Approved by AICTE, Affiliated to JNTU Hyderabad, Accredited by NBA)

Kandlakoya(V), Medchal(M), Telangana – 501401

(2024-2025)

CMR ENGINEERING COLLEGE

UGC AUTONOMOUS

(Approved by AICTE, Affiliated to JNTU Hyderabad, Accredited by NBA)

Kandlakoya(V), Medchal Road, Hyderabad - 501 401

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

CERTIFICATE

This is to certify that the major-project work entitled **"VOICE CONTROLLED PICK AND PLACE ROBOT"** is being submitted by **T. HARI CHANDRA PRASAD** bearing Roll No **218R1A04C0, T. MOUNIKA** bearing Roll No **218R1A04C1, V. PRANATHI** bearing Roll No **218R1A04C2, V. TEJASHWINI** bearing Roll No **218R1A04C3** in B.Tech IV-II semester, Electronics and Communication Engineering is a record Bonafide work carried out during the academic year 2024-25. The results embodied in this report have not been submitted to any other University for the award of any degree.

INTERNAL GUIDE	HEAD OF THE DEPARTMENT
Dr. M. AMRU	**Dr. SUMAN MISHRA**

EXTERNAL EXAMINER

ACKNOLEDGEMENT

We sincerely thank the management of our college **CMR Engineering College** for providing required facilities during our project work. We derive great pleasure in expressing our sincere gratitude to our Principal **Dr. A. S. Reddy** for his timely suggestions, which helped us to complete the project work successfully. It is the very auspicious moment we would like to express our gratitude to **Dr. SUMAN MISHRA,** Head of the Department, ECE for his consistent encouragement during the progress of this project.

We take it as a privilege to thank our project coordinator **Dr. T. SATYANARAYANA**, Associate Professor, Department of ECE for the ideas that led to complete the project work and we also thank him for his continuous guidance, support and unfailing patience, throughout the course of this work. We sincerely thank our project internal guide **Dr. M. AMRU,** Professor of ECE for guidance and encouragement in carrying out this project work.

DECLARATION

We hereby declare that the major project entitled "**VOICCE CONTROLLED PICK AND PLACE ROBOT**" is the work done by us in campus at **CMR ENGINEERING COLLEGE,** Kandlakoya during the academic year 2024-2025 and is submitted as major project in partial fulfilment of the requirements for the award of degree of **BACHELOR OF TECHNOLOGY** in **ELECTRONICS AND COMMUNICATION ENGINEERING COLLEGE FROM JAWAHARLAL NEHRU TCHNOLOGICAL UNIVERSITY, HYDERABAD**

T. HARI CHANDRA PRASAD	**(218R1A04C0)**
T. MOUNIKA	**(218R1A04C1)**
V. PRANATHI	**(218R1A04C2)**
V. TEJASHWINI	**(218R1A04C3)**

ABSTRACT

This project presents a **Voice-Controlled Pick and Place Robot** that uses voice commands to perform object manipulation tasks with high precision and efficiency. The system integrates an **Arduino microcontroller** as the central processing unit, interfacing with various modules such as a **Bluetooth module** for wireless voice communication, an **ESP32 Camera** for visual monitoring, and **L298 motor drivers** to control the movement of the robotic arm and wheels. The robot receives voice commands via a mobile application connected through Bluetooth, which are then processed by the Arduino to perform actions like picking up and placing objects. The L298 motor drivers control the motors (M1 and M2) for both the arm and the base of the robot, ensuring smooth and accurate movement. The power supply provides the necessary voltage for all the components to operate efficiently. The ESP32 Cam provides real-time visual feedback, enabling remote monitoring of the robot's performance. This system is ideal for applications in automation, warehousing, and industries where precision, safety, and efficiency are critical. The integration of voice control enhances ease of use and accessibility, making the system adaptable for various industrial and domestic applications.

Keywords: Voice-Controlled Robot, Pick and Place Robot, Arduino Microcontroller, Bluetooth Module, ESP32 Camera, L298 Motor Driver, Wireless Communication.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER-1
INTRODUCTION

Pick and place robotic arms are extensively utilized in various industries, particularly in mechanical environments and manufacturing sectors, where they assist in picking and placing components at designated locations with high precision. These systems play a crucial role in automating repetitive tasks, ensuring operational efficiency, and minimizing human intervention in hazardous or labor-intensive processes. In this work, a mobile robot has been specifically developed to assist individuals with physical disabilities, such as wheelchair-bound persons, in performing routine tasks. The robot is equipped with the capability to pick and place objects at desired locations through intuitive voice commands. This innovation aims to enhance the independence of disabled individuals by enabling them to perform daily activities without requiring external assistance.

The mobile robot can move freely across different areas, allowing it to approach small cupboards or shelves where objects are stored. Using voice commands, such as “left,” “right,” “straight,” and other directional inputs, the user can guide the robot to navigate smoothly in their environment. Once the robot reaches the designated location, it can accurately pick the object and transport it to the desired destination. The automated navigation system ensures that the robot can efficiently follow instructions and respond appropriately to user commands. The system integrates a voice recognition module that processes human voice inputs and translates them into corresponding actions. The use of voice commands enhances ease of operation and makes the system highly user-friendly, especially for individuals with mobility challenges. By responding effectively to basic directional commands, the robot ensures seamless operation in a home or limited space environment. This innovation significantly improves the quality of life for individuals with limited mobility by giving them the autonomy to complete everyday tasks such as retrieving objects from shelves or placing items at designated spots. By reducing dependence on caregivers, the system promotes independence and empowers individuals to perform routine activities effortlessly.

The voice-controlled pick and place robot is a promising solution for enhancing the mobility and independence of disabled individuals. Its ability to perform tasks based on simple voice commands makes it highly practical and beneficial in both domestic and industrial environments. The system demonstrates a remarkable potential to improve the daily lives of physically challenged users by offering greater control and convenience.

1.1EMBEDDED SYSTEM

An Embedded System is a combination of computer hardware and software, and perhaps additional mechanical or other parts, designed to perform a specific function. A good example is the microwave oven. Almost every household has one, and tens of millions of them are used every day, but very few people realize that a processor and software are involved in the preparation of their lunch or dinner.

This is in direct contrast to the personal computer in the family room. It too is comprised of computer hardware and software and mechanical components (disk drives, for example). However, a personal computer is not designed to perform a specific function rather; it is able to do many different things. Many people use the term general purpose computer to make this distinction clear. As shipped, a general-purpose computer is a blank slate; the manufacturer does not know what the customer will do wish it. One customer may use it for a network file server another may use it exclusively for playing games, and a third may use it to write the next great American novel.

Frequently, an embedded system is a component within some larger system. For example, modern cars and trucks contain many embedded systems. One embedded system controls the anti-lock brakes, other monitors and controls the vehicle's emissions, and a third displays information on the dashboard. In some cases, these embedded systems are connected by some sort of a communication network, but that is certainly not a requirement.

At the possible risk of confusing you, it is important to point out that a general-purpose computer is itself made up of numerous embedded systems. For example, my computer the existence of the processor and software could be completely unnoticed by the user of the device. Such is the case for a microwave oven, VCR, or alarm clock. In some cases, it would even be possible to build an equivalent device that does not contain the processor and software. This could be done by replacing the combination with a custom integrated circuit that performs the same functions in hardware. However, a lot of flexibility is lost when a design is hard-cooled in this way. It is much easier, and cheaper, to change a few lines of software than to redesign a piece of custom hardware.

consists of a keyboard, mouse, video card, modem, hard drive, floppy drive, and sound card-each of Which is an embedded system? Each of these devices contains a processor and software and is designed to perform a specific function. For example, the modem is designed to send and receive digital data over analog telephone line. That's it and all of the other devices can be summarized in a single sentence as well.

If an embedded system is designed well, the existence of the processor and software could be completely unnoticed by the user of the device. Such is the case for a microwave oven, VCR, or alarm clock. In some cases, it would even be possible to build an equivalent device that does not contain the processor and software. This could be done by replacing the combination with a custom integrated circuit that performs the same functions in hardware. However, a lot of flexibility is lost when a design is hard-cooled in this way. It is much easier, and cheaper, to change a few lines of software than to redesign a piece of custom hardware.

1.2 HISTORY AND FUTURE

Given the definition of embedded systems earlier is this chapter; the first such systems could not possibly have appeared before 1971. That was the year Intel introduced the world's first microprocessor. This chip, the 4004, was designed for use in a line of business calculators produced by the Japanese Company Busycon. In 1969, Busycon asked Intel to design a set of custom integrated circuits-one for each of their new calculator models. The 4004 was Intel's response rather than design custom hardware for each calculator, Intel proposed a general-purpose circuit that could be used throughout the entire line of calculators. Intel's idea was that the software would give each calculator its unique set of features.

The microcontroller was an overnight success, and its use increased steadily over the next decade. Early embedded applications included unmanned space probes, computerized traffic lights, and aircraft flight control systems. In the 1980s, embedded systems quietly rode the waves of the microcomputer age and brought microprocessors into every part of our kitchens (bread machines, food processors, and microwave ovens), living rooms (televisions, stereos, and remote controls), and workplaces (fax machines, pagers, laser printers, cash registers, and credit card readers).

It seems inevitable that the number of embedded systems will continue to increase rapidly. Already there are promising new embedded devices that have enormous market potential; light switches and thermostats that can be central computer, intelligent air-bag systems that don't inflate when children or small adults are present, pal-sized electronic organizers.

Personal digital assistants (PDAs), digital cameras, and dashboard navigation systems. Clearly, individuals who possess the skills and desire to design the next generation of embedded systems will be in demand for quite some time.

1.3 REAL TIME SYSTEMS

One subclass of embedded is worthy of an introduction at this point. As commonly defined, a real-time system is a computer system that has timing constraints. In other words, a real- time system is partly specified in terms of its ability to make certain calculations or decisions in a timely manner. These important calculations are said to have deadlines for completion. And, for all practical purposes, a missed deadline is just as bad as a wrong answer.

The issue of what if a deadline is missed is a crucial one. For example, if the real-time system is part of an airplane's flight control system, it is possible for the lives of the passengers and crew to be endangered by a single missed deadline. However, if instead the system is involved in satellite communication, the damage could be limited to a single corrupt data packet. The more severe the consequences, the more likely it will be said that the deadline is "hard" and thus, the system is a hard real-time system. Real-time systems at the other end of this discussion are said to have "soft" deadlines.

All of the topics and examples presented in this book are applicable to the designers of real-time system who is more delight in his work. He must guarantee reliable operation of the software and hardware under all the possible conditions and to the degree that human lives depend upon three system's proper execution, engineering calculations and descriptive paperwork.

1. **Application Areas :** Nearly 99 per cent of the processors manufactured end up in embedded systems. The embedded system market is one of the highest growth areas as these systems are used in very market segment- consumer electronics, office automation, industrial automation, biomedical engineering, wireless communication, Data communication, telecommunications, transportation, military and so on.
2. **Office automation:** The office automation products using em embedded systems are copying machine, fax machine, key telephone, modem, printer, scanner etc.
3. **Medical electronics:** Almost every medical equipment in the hospital is an embedded system. These equipment include diagnostic aids such as ECG, EEG, blood pressure measuring devices, X-ray scanners; equipment used in blood analysis, radiation,

colonoscopy, endoscopy etc. Developments in medical electronics have paved way for more accurate diagnosis of diseases.

4. **Computer networking:** Computer networking products such as bridges, routers, Integrated Services Digital Networks (ISDN), Asynchronous Transfer Mode (ATM) and frame relay switches are embedded systems which implement the necessary data communication protocols. For example, a router interconnects two networks. The two networks may be running different protocol stacks. The router's function is to obtain the data packets from incoming pores, analyse the packets and send them towards the destination after doing necessary protocol conversion. Most networking equipment, other than the end systems (desktop computers) we use to access the networks, are embedded systems.

5. **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 cameras are embedded systems. The network equipment includes multiplexers, multiple access systems, Packet Assemblers Dissemblers (PADs), sate11ite modems etc. IP phone, IP gateway, IP gatekeeper etc. are the latest embedded systems that provide very low-cost voice communication over the Internet.
6. **Wireless technologies:** Advances in mobile communications are paving way for many interesting applications using embedded systems. The mobile phone is one of the marvels of the last decade of the 20'h century. 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 centres are also powerful embedded systems.
7. **Insemination:** Testing and measurement are the fundamental requirements in all scientific and engineering activities. The measuring equipment we use in laboratories to measure parameters such as weight, temperature, pressure, humidity, voltage, current etc. are all embedded systems. Test equipment such as oscilloscope, spectrum analyser, logic analyser, protocol analyser, radio communication test set etc. are embedded systems built around powerful processors. Thank to miniaturization, the test and measuring equipment are now becoming portable facilitating easy testing and measurement in the field by field-personnel.

8. **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 nowadays. Security devices at homes, offices, airports etc. for authentication and verification are embedded systems. Embedded systems find applications in. Every industrial segment- consumer electronics, data communication, telecommunication, defence, security etc.

1.4 OVERVIEW OF PROJECT

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. The software residing on the memory chip is also called the 'firmware'. The embedded system architecture can be represented as a layered architecture.

The operating system runs above the hardware, and the application software runs above the operating system. The same architecture is applicable to any computer including a desktop computer. However, there are significant differences. It is not compulsory to have an operating system in every embedded system.

The goal of this project is to develop an advanced vehicle security system that integrates real-time monitoring, tracking, and protection features. This system aims to enhance vehicle safety by providing real-time data to vehicle owners, ensuring theft deterrence, and enabling immediate response in case of a security breach.

For small appliances such as remote control units, air conditioners, toys etc., there is no need for an operating system and you can write only the software specific to that application. For applications involving complex processing, it is advisable to have an operating system. In such a case, you need to integrate the application software with the operating system and then transfer the entire software on to the memory chip. Once the software is transferred to the memory chip, the software will continue to run fora 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. As shown in Fig. the building blocks are:

- Central Processing Unit (CPU)
- Memory (Read-only Memory and Random Access Memory)

- Input Devices
- Output devices
- Communication interfaces
- Application-specific circuitry Central Processing Unit (CPU):

The Central Processing Unit (processor, in short) can be any of the following: microcontroller, microprocessor or Digital Signal Processor (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. D5P is used mainly for applications in which signal processing is involved such as audio and video processing.

Memory:

The memory is categorized as Random Access 11emory (RAM) and Read Only Memory (ROM). The contents of the RAM will be erased if power is switched off to the chip, whereas 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 processor reads the ROM; the program is program is executed.

Input devices:

Unlike the desktops, 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 focuser interaction; they take inputs from sensors or transducers 1'fnd produce electrical signals that are in turn fed to other systems.

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.

Communication interfaces:

The embedded systems may need to, interact with other embedded systems at they may have to transmit data to a desktop. To facilitate this, the embedded systems are provided with one or a few communication interfaces such as RS232, RS422, RS485, Universal Serial Bus (USB), IEEE 1394, Ethernet etc.

Application-specific circuitry:

Sensors, transducers, special processing and control circuitry may be required fat an embedded system, depending on its application. This circuitry interacts with the processor to carry out the necessary work. The entire hardware has to be given power supply either through the 230 volts main supply or through a battery. 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 focuser interaction; they take inputs from sensors or transducers 1'fnd produce electrical signals that are in turn fed to other systems. This circuitry interacts with the processor to carry out the necessary work. The entire hardware has to be given power supply either through the 230 volts main supply or through a battery. 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 focuser interaction; they take inputs from sensors or transducers 1'fnd produce electrical signals that are in turn fed to other systems. The hardware has to design in such a way that the power consumption is minimized.

CHAPTER-2
LITERATURE SURVEY

This proposed system was finalized after checking out various references given and the implementations with methodologies was also discussed. Past ten years of reference papers was used for it.

The detailed report on these references are given below, Sotiris Stavridis, Pietro Falco, Zoe Doulgeri, “Pick-and-place in dynamic environments with a mobile dual-arm robot equipped with distributed distance sensors”, Mobile bimanual manipulation in an exceedingly dynamic and unsure environment requires the continual and fast adjustment of the robot motion for the satisfaction of the constraints imposed by the task, the robot itself, and also the environment. We formulate the pick-and-place task as a sequence of mobile manipulation tasks with a mix of relative, global and native targets.

2.Hae-Chang Kim, In-Hwan Yoon, Jae-Bok Song, “Target Position Estimation for Pick-and-Place Tasks using a Mobile Manipulator, A mobile manipulator uses marker detection and hand-eye calibration to catch up on the performance limitation of position estimation during pick-and-place tasks. However, whether these methods are applied or not, a mistake occurs proper to the limitation of camera calibration, so it's tough to locate the object within the correct position.

3.Muhammad Affan, Syed Umaid Ahmed, Riaz Uddin, “Pick-and-Place Task using Wheeled Mobile Manipulator”, A Control Design Perspective, This paper is aimed at students and roboticists to produce the concise theoretical and applied knowledge necessary for the control design of mobile manipulators. For this purpose, topics like kinematics, motion planning, and control theory are explored. Moreover, this information is integrated from the attitude of the Mecanum wheeled 5-R mobile manipulator for the pick-and-place task of the cube.

4. Fengyi Wang, J. Rogelio Guadarrama Olvera, “Optimal Order Pick-and-Place of Objects in Cluttered Scene by a Mobile Manipulator", In this paper, we present a quick method for autonomously plane manipulation tasks for mobile manipulators. An optimal order is defined by the planner to perform and the objects are taken from pick and place operations from a cluttered scene to specific deposit areas considering both, manipulator and mobile base motion. The grasping feasibility of the objects was examined by the first method with an inverse reachability map.

5. K.N.V. Sriram, Suja Palaniswamy, “Mobile Robot Assistance for Disabled and Senior

Citizens Using Hand Gestures", Numbness and movability impairments impact the autonomy of elder people while performing their independent tasks. Gesture role acts as a bridge between humans and machines. This work focuses on Human-Robot Interaction (HRI), designed for the assistance of wheelchair-bound people with the assistance of mobile robots.

.

2.1 EXISTING SYSTEM

In traditional pick and place systems, industrial robotic arms are programmed to perform repetitive tasks, typically in manufacturing, mechanical sites, and assembly lines. These systems follow pre-defined trajectories and execute commands based on pre-set parameters. However, knowing the strengths and weaknesses of current approaches can set expectations appropriately, identify risk areas to tool adopters, and suggest ways in which tool builders can meet industrial needs For individuals with disabilities, basic assistive technologies such as manual wheelchairs, mechanical grabbers, or semi-automated devices help them perform daily tasks. However, these systems often require physical effort, manual control, or assistance from caregivers, which can be cumbersome and inefficient.

Disadvantages:

- **Lack of Flexibility:** Industrial robots follow pre-defined trajectories, making them unsuitable for dynamic or home environments.
- **Manual Operation:** Most assistive devices for disabled individuals require manual intervention, which may not be feasible for users with limited mobility.
- **Limited Autonomy:** Traditional systems lack the ability to operate autonomously based on user inputs, reducing the independence of the user.
- **Complex Interfaces:** Some existing systems use complex interfaces that are not user-friendly for disabled individuals, especially those with cognitive or speech impairments.
- **High Cost and Maintenance:** Industrial-grade pick and place robots are expensive and require regular maintenance, making them impractical for domestic use.

2.2 PROPOSED SYSTEMS

The proposed system introduces a **voice-controlled pick and place mobile robot** designed specifically to assist wheelchair-bound and physically challenged individuals. This robot is fully automated and capable of responding to **voice commands** to perform object manipulation tasks, such as picking up objects and placing them at desired locations.

The system integrates:

- **Arduino Microcontroller:** Processes the input and controls the motors.
- **Bluetooth Module:** Facilitates wireless voice communication between the user and the robot.
- **ESP32 Cam Module:** Provides real-time monitoring and feedback.
- **L298 Motor Driver:** Controls the movement of the robotic arm and wheels.
- **Advantages**:
- **Hands-Free Operation:** The robot responds to voice commands, eliminating the need for manual effort, making it highly accessible for individuals with physical limitations.
- **Increased Autonomy:** Users can independently perform tasks such as picking and placing objects, thereby reducing dependence on caregivers.
- **User-Friendly Interface:** Voice command functionality simplifies the interaction between the user and the robot, ensuring ease of use.
- **Real-Time Monitoring:** The ESP32 Cam provides real-time visual feedback, enabling users or caregivers to monitor the robot's actions remotely.
- **Enhanced Mobility and Precision:** The L298 motor driver ensures smooth and accurate control of the robot's wheels and arm, allowing it to navigate tight spaces and complete tasks effectively.
- **Cost-Effective Solution:** Compared to industrial-grade systems, the proposed system is a low-cost, efficient, and practical solution for domestic use.
- **Customizable and Scalable:** The system can be adapted to different environments and upgraded to include additional functionalities based on the user's needs.

2.3 EMBEDDED INTRODUCTION

Many embedded systems have substantially different design constraints than desktop computing applications. No single characterization applies to the diverse spectrum of embedded systems. However, some combination of cost pressure, long life-cycle, real-time requirements, reliability requirements, and design culture dysfunction can make it difficult to be successful applying traditional computer design methodologies and tools to embedded applications. There is currently little tool support for expanding embedded computer design to the scope of holistic embedded system design. However, knowing the strengths and weaknesses of current approaches can set expectations appropriately, identify risk areas to tool adopters, and suggest ways in which tool builders can meet industrial needs.

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) codesign 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.

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, implementation synthesis and performance evaluation of the prototype.

2..3.1 SPECIFICATION

During this part of the design process, the informal requirements of the analysis are transformed to formal specification using SDL. All the major toy makers across the world have been coming out with advanced interactive toys that can become our friends for life 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.

2.3.2 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.

2.3.3 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.

2.3.4 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.

2.3.5 APPLICATIONS

Embedded systems are finding their way into robotic toys and electronic pets, intelligent cars and remote controllable home appliances. All the major toy makers across the world have been coming out with advanced interactive toys that can become our friends for life. ‘Furby’ and ‘AIBO’ are good examples at this kind. Furbies have a distinct life cycle just like human beings, starting from being a baby and growing to an adult one. In AIBO first two letters stands for Artificial Intelligence. Next two letters represents robot.

The AIBO is robotic dog. Embedded systems in cars also known as Telematic Systems are used to provide navigational security communication & entertainment services using GPS, satellite. IBM is developing an air conditioner that we can control over the net. Embedded systems cover such a broad range of products that generalization is difficult. Here are some broad categories. Patient monitoring systems track vital signs and alert healthcare professionals to critical changes, ensuring timely interventions. Diagnostic equipment, such as MRI machines and blood glucose monitors, utilizes embedded systems for data processing and visualization.

- Aerospace and defence electronics: Fire control, radar, robotics/sensors, sonar.
- Automotive: Autobody electronics, auto power train, auto safety, car information systems.

- Broadcast & entertainment: Analog and digital sound products, camaras, DVDs, Set top boxes, virtual reality systems, graphic products.
- Consumer/internet appliances: Business handheld computers, business network computers/terminals, electronic books, internet smart handheld devices, PDAs.
- Data communications: Analog modems, ATM switches, cable modems, XDSL modems, Ethernet switches, concentrators.
- Digital imaging: Copiers, digital still cameras, Fax machines, printers, scanners.
- Industrial measurement and control: Hydro electric utility research & management traffic management systems, train marine vessel management systems.
- Medical electronics: Diagnostic devices, real time medical imaging systems, surgical devices, critical care systems.
- Server I/O: Embedded servers, enterprise PC servers, PCI LAN/NIC controllers, RAID devices, SCSI devices.
- Telecommunications: ATM communication products, base stations, networking switches, SONET/SDH cross connect, multiplexer.
- Mobile data infrastructures: Mobile data terminals, pagers, VSATs, Wireless LANs, Wireless phones.
- Embedded systems in many cases must be optimized for life-cycle and business driven factors rather than for maximum computing throughput.
- Home appliances are going the embedded way. LG electronics digital DIOS refrigerator can be used for surfing the net, checking e-mail, making video phone calls and watching TV.

2.4 WHY EMBEDDED?

An embedded system is a specialize computer system —a combination of a computer processor, computer memory and input/output peripheral devices—that has a dedicated function within a larger mechanical or electronic system. It is embedded as part of a complete device often including electrical or electronic hardware and mechanical parts. Because an embedded system typically controls physical operations of the machine that it is embedded within, it often has real-time computing constraints. Embedded systems control many devices in common use. In 2009, it was estimated that ninety-eight percent of all microprocessors manufactured were used in embedded systems.

Modern embedded systems are often based on microcontrollers (i.e. microprocessors with integrated memory and peripheral interfaces), but ordinary microprocessors (using external chips for memory and peripheral interface circuits) are also common, especially in more complex systems. In either case, the processor(s) used may be types ranging from general purpose to those specialized in a certain class of computations, or even custom designed for the application at hand. A common standard class of dedicated processors is the digital signal processor (DSP).

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

Embedded systems range in size from portable personal devices such as digital watches and MP3-players to bigger machines like home appliances, industrial assembly lines, robots, transport vehicles, traffic light controllers, and medical imaging systems. Often they constitute subsystems of other machines like avionics in aircraft and astrionics in spacecrafts. Large installations like factories, pipelines and electrical grids rely on multiple embedded systems networ-ked together. Generalized through software customization, embedded systems such as programmable logic controllers frequently comprise their functional units.

The embedded system life cycle involves several key stages, starting with conceptualization, where requirements are defined, followed by system design to determine hardware and software components. Hardware design focuses on creating the physical components, while software development involves programming the system's firmware. Next, integration and testing ensure hardware and software function together, followed by prototyping and validation to test the system under real-world conditions. After refining the design, the system enters manufacturing for mass production and deployment for installation and field testing. Ongoing maintenance and support ensure the system remains functional, and eventually, the system reaches end of life (EOL) when it is retired and replaced.

During hardware design, physical components such as microcontrollers, sensors, and circuits are selected and developed. Software development involves coding the firmware and application software, which is then integrated and tested to ensure all components function as intended. After integration, prototyping and validation ensure the system performs under real-world conditions, leading to production and manufacturing, where the system is mass-produced. Once deployed, the system enters a phase of maintenance and support, which involves bug fixes, updates, and system monitoring to ensure continued functionality.

Embedded systems range from those low in complexity, with a single microcontroller chip, to very high with multiple units, peripherals and networks, which may reside in equipment racks or across large geographical areas connected via long-distance communications lines.

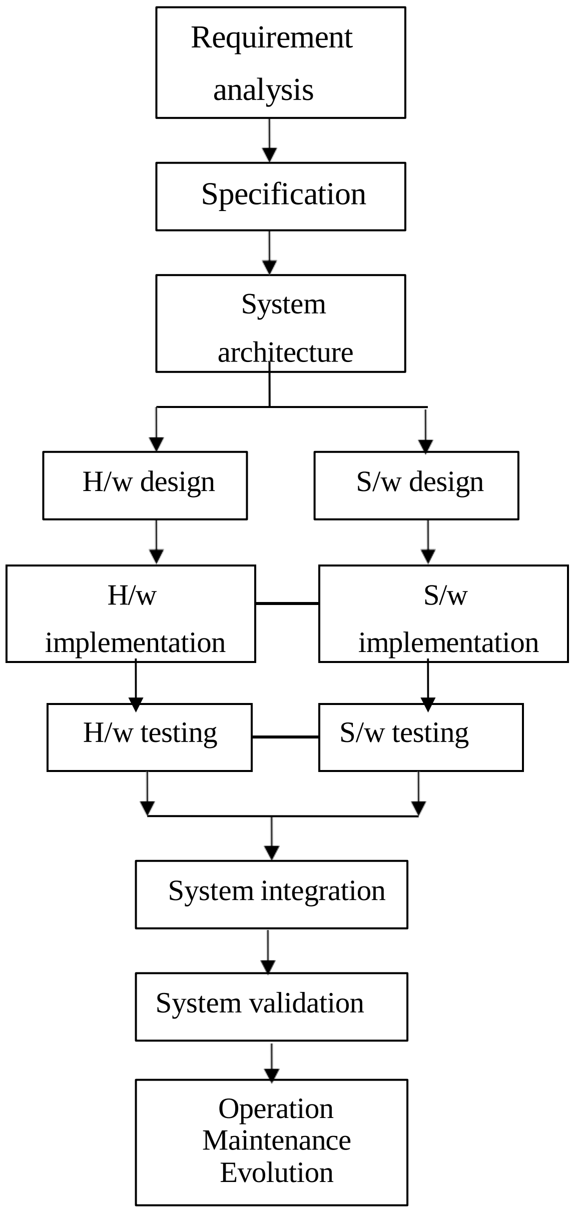


Fig 2.1: Embedded Development Life Cycle

2.5 DESIGN APPROACHES

The design approach focuses on integrating these components efficiently to ensure real-time tracking, enhanced security features, and remote communication. Below is a detailed explanation of the design approach for each of the components and how they integrate in the system.

2.6 EMBEDDED SYSTEM WITH ARDUINO

The Arduino microcontroller serves as the central control unit of the entire vehicle protection and tracking system. It is chosen for its ease of use, availability, and versatility in handling multiple I/O operations. The design approach involves programming the Arduino to interact with the GPS module, GPS modem, night vision camera, and other sensors. The system's architecture is designed to enable continuous data collection, processing, and communication in real-time.

Data Processing: The Arduino is responsible for reading the GPS coordinates from the GPS module and processing other sensor data (e.g., temperature, alcohol, impact sensors) in real-time. The microcontroller processes all incoming data and triggers specific actions based on pre-defined conditions. For example, if the temperature sensor detects overheating, it sends an alert; if the vehicle moves outside a geofenced area, an alarm is triggered.

Control and Communication: The Arduino also controls the communication between the GPS modem and external devices (such as mobile phones or cloud servers). It ensures that location data and alerts are transmitted via GSM/GPRS, and it can send commands (e.g., remote vehicle immobilization) based on user inputs. Additionally, the night vision camera can be controlled by the Arduino, triggering it to take images or video when specific conditions are met, such as movement or impact detection. The design approach involves programming the Arduino to interact with the GPS module, GPS modem, night vision camera, and other sensors. The system's architecture is designed to enable continuous data collection, processing, and communication in real-time

CHAPTER-3
HARDWARE REQUIREMETS

3.1 HARDWARE

Microcontroller:

Introduction:

Microcontroller as the name suggest, a small controller. They are like single chip computers that are often embedded into other systems to function as processing/controlling unit. For example, the control you are using probably has microcontrollers inside that do decoding and other controlling functions. They are also used in automobiles, washing machines, microwaves ovens, toys…. etc, where automation is needed.

3.1.1 Arduino Uno Microcontroller:

The Arduino Uno is a microcontroller board based on the ATmega328 (datasheet). It has 14 digital input/output pins (of which 6 can be used as PWM outputs), 6 analog inputs, a 16 MHz crystal oscillator, a USB connection, a power jack, an ICSP header, and a reset button. It contains everything needed to support the microcontroller; simply connect it to a computer with a USB cable or power it with a AC-to-DC adapter or battery to get started.

The Uno differs from all preceding boards in that it does not use the FTDI USB-to-serial driver chip. Instead, it features the Atmega8U2 programmed as a USB-to-serial converter. "Uno" means "One" in Italian and is named to mark the upcoming release of Arduino 1.0. The Uno and version 1.0 will be the reference versions of Arduino, moving forward. The Uno is the latest in a series of USB Arduino boards, and the reference model for the Arduino platform; for a comparison with previous versions, see the index of Arduino boards.

The Arduino Uno 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 of 6 to 20 volts. If supplied with less than 7V, however, the 5Vpin may supply less than five volts and the board may be unstable. If using more than 12V, the voltage regulator may overheat and damage the board. The recommended range is 7 to 12 volts
The power pins are as follows:

- **VIN.** The input voltage to the Arduino board when it's using an external power source (as opposed to5 volts from the USB connection or other regulated powe).
- **5V.** The regulated power supply used to power the microcontroller and other components on the board. This can come either from VIN via an on-board regulator, or be supplied by USB or another regulated 5V supply.
- **3.3V.**A 3.3 volt supply generated by the on-board regulator. Maximum current draw is 50 mA.
- **GND.** Ground pins.

Memory:

The Atmega328 has 32 KB of flash memory for storing code (of which 0,5 KB is used for the bootloader); It has also 2 KB of SRAM and 1 KB of EEPROM (which can be read and written with the EEPROM library).

Input and Output:

Each of the 14 digital pins on the Uno can be used as an input or output, using pin Mode(), digital Write(), and digital Read() functions. They operate at 5 volts. Each pin can provide or receive a maximum of 40 mA and has an internal pull-up resistor (disconnected by default) of 20-50 kOhms. In addition, some pins have specialized functions:

- **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, arising or falling edge, or a change in value. See the attach Interrupt() function for details.
- **PWM: 3, 5, 6, 9, 10, and 11.** Provide 8-bit PWM output with the analog Write() function.
- **SPI: 10 (SS), 11 (MOSI), 12 (MISO), 13 (SCK).** These pins support SPI communication, which although provided by the underlying hardware, is not currently included in the Arduino language.
- **LED: 13.** There is a built-in LED connected to digital pin 13. When the pin is HIGH value, the LED is on, when the pin is LOW, it's off.

The Uno has 6 analog inputs, 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 analog Reference() function. Additionally, some pins have specialized functionality:

- **I2C: 4 (SDA) and 5 (SCL).** Support I2C (TWI) communication using the Wire library.

There are a couple of other pins on the board:

- **AREF.** Reference voltage for the analog inputs. Used with analog Reference().
- **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:

The Arduino Uno has a number of facilities for communicating with a computer, another Arduino, or other microcontrollers. The ATmega328 provides UART TTL (5V) serial communication, which is available on digital pins 0 (RX) and 1 (TX). An ATmega8U2 on the board channels this serial communication over USB and appears as a virtual com port to software on the computer. The '8U2 firmware uses the standard USBCOM drivers, and no external driver is needed. However, on Windows, an *.inf file is required. The Arduino software includes a serial monitor which allows simple textual data to be sent to and from the Arduino 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 for serial communication on any of the Uno's digital pins. The ATmega328 also support I2C (TWI) and SPI communication. The Arduino software includes a Wire library to simplify use of the I2C bus.

3.1.2 ARDUINO UNO BOARD:

The Arduino Uno is a microcontroller board based on the ATmega328. It has 14 digital input/output pins (of which 6 can be used as PWM outputs), 6 analog inputs, a 16 MHz ceramic resonator, a USB connection, a power jack, an ICSP header, and a reset button.

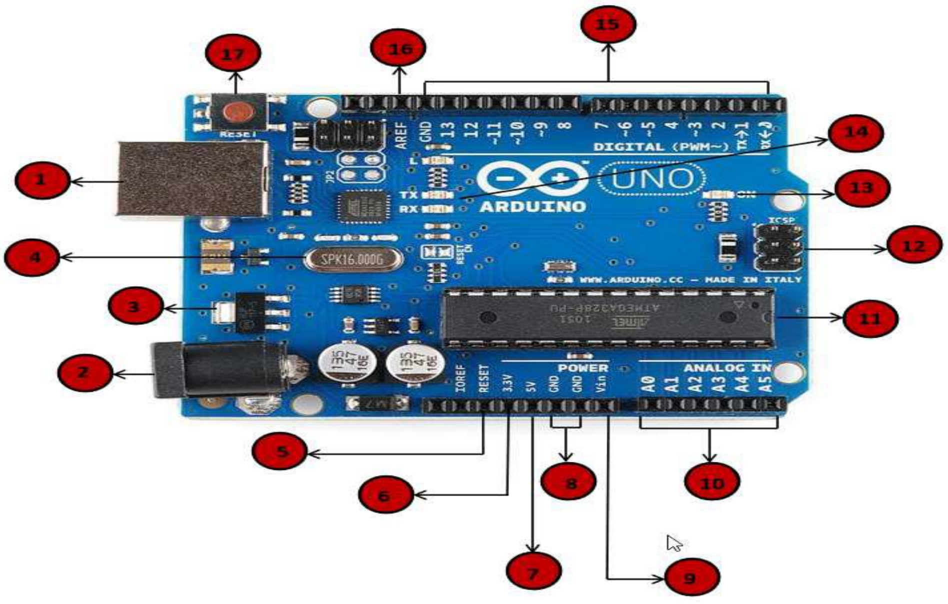
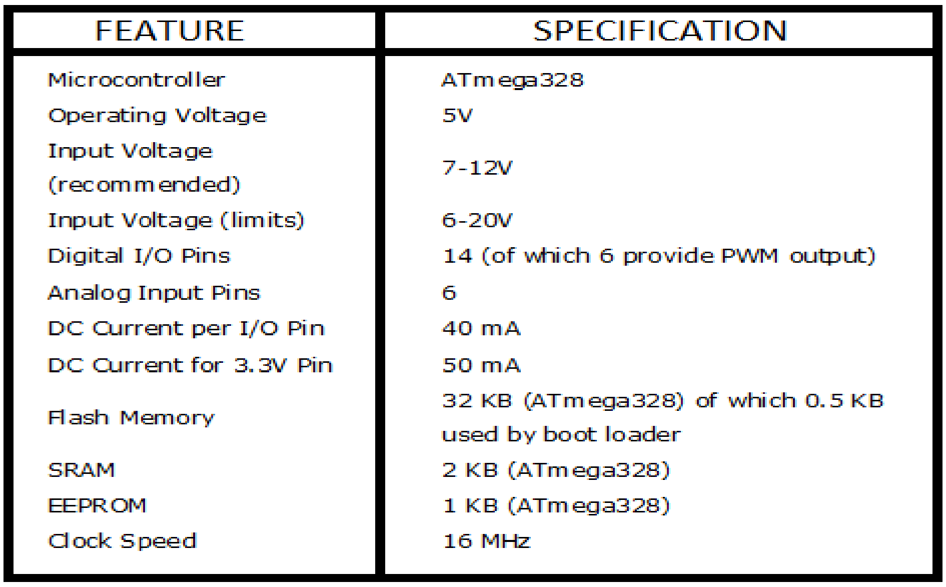


Figure 3.1: Arduino uno board

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 converters.

3.1.2.1 Technical Specifications:



FEATURE	SPECIFICATION
Microcontroller	ATmega328
Operating Voltage	5V
Input Voltage (recommended)	7-12V
Input Voltage (limits)	6-20V
Digital I/O Pins	14 (of which 6 provide PWM output)
Analog Input Pins	6
DC Current per I/O Pin	40 mA
DC Current for 3.3V Pin	50 mA
Flash Memory	32 KB (ATmega328) of which 0.5 KB used by boot loader
SRAM	2 KB (ATmega328)
EEPROM	1 KB (ATmega328)
Clock Speed	16 MHz

Table 3.2: Arduino uno specifications

The Arduino Uno 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 of 6 to 20 volts. If supplied with less than 7V, however, the 5V pin may supply less than five volts and the board may be unstable. If using more than 12V, the voltage regulator may overheat and damage the board. The recommended range is 7 to 12 volts.

1.USB Interface:

Arduino board can be powered by using the USB cable from your computer. All you need to do is connect the USB cable to the USB connection

2.External power supply:

Arduino boards can be powered directly from the AC mains power supply by connecting it to the power supply (Barrel Jack)

3.Voltage Regulator:

The function of the voltage regulator is to control the voltage given to the Arduino board and stabilize the DC voltages used by the processor and other elements.

4.Crystal Oscillator:

The crystal oscillator helps Arduino in dealing with time issues. How does Arduino calculate time? The answer is, by using the crystal oscillator. The number printed on top of the Arduino crystal is 16.000H9H. It tells us that the frequency is 16,000,000 Hertz or 16 MHz.

5,17. Arduino Reset:

It can reset your Arduino board, i.e., start your program from the beginning. It can reset the UNO board in two ways. First, by using the reset button (17) on the board. Second, you can connect an external reset button to the Arduino pin labelled RESET (5).

6-9. Pins (3.3, 5, GND, Vin):

- 3.3V (6): Supply 3.3 output volt
- 5V (7): Supply 5 output volt
- Most of the components used with Arduino board works fine with 3.3 volt

  and 5 volt.
- GND (8)(Ground): There are several GND pins on the Arduino, any of which can be used to ground your circuit.
- Vin (9): This pin also can be used to power the Arduino board from an

  external power source, like AC mains power supply.

10.Analog pins:

The Arduino UNO board has five analog input pins A0 through A5. These pins can read the signal from an analog sensor like the humidity sensor or temperature sensor and convert it into a digital value that can be read by the microprocessor.

11.Main microcontroller:

Each Arduino board has its own microcontroller (11). You can assume it as the brain of your board. The main IC (integrated circuit) on the Arduino is slightly different from board to board. The microcontrollers are usually of the ATMEL Company. You must know what IC your board has before loading up a new program from the Arduino IDE. This information is available on the top of the IC. For more details about the IC construction and functions, you can refer to the data sheet.

The Atmega8U2 programmed as a USB-to-serial converter. "Uno" means "One" in Italian and is named to mark the upcoming release of Arduino 1.0. The Uno and version 1.0 will be the reference versions of Arduino, moving forward. The Uno is the latest in a series of USB Arduino boards, and the reference model for the Arduino platform; for a comparison with previous versions.

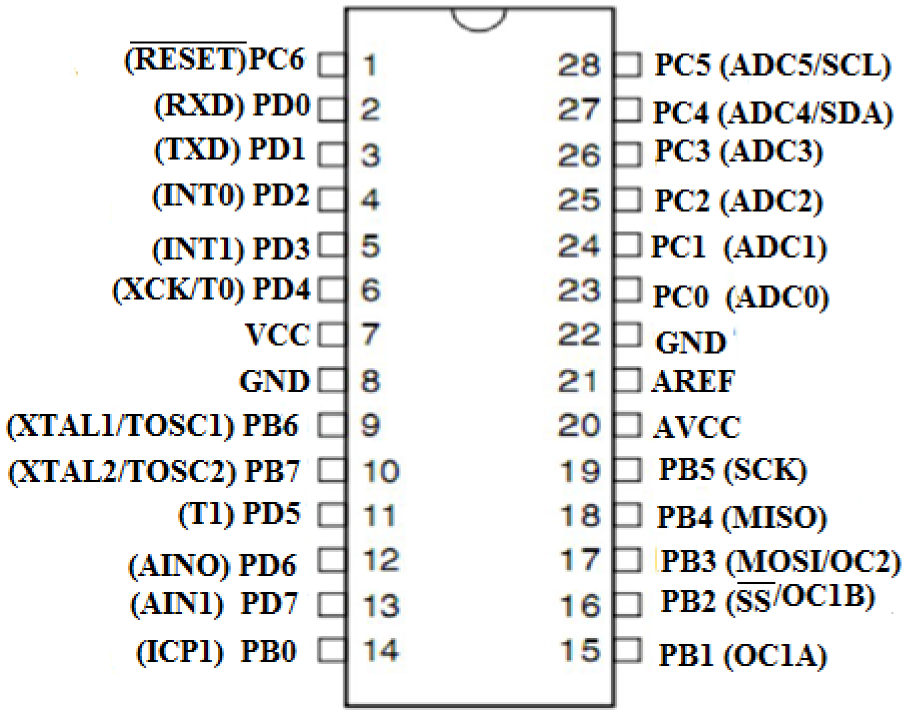


Figure 3.3: Pin diagram

3.1.2.2 Pin Description:

VCC: Digital supply voltage.

GND: Ground.

Port B (PB [7:0]) XTAL1/XTAL2/TOSC1/TOSC2:

Port B is an 8-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). The Port B output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, Port B pins that are externally pulled low will source current if the pull-up resistors are activated. The Port B pins are tri-stated when a reset condition becomes active, even if the clock is not running.

Depending on the clock selection fuse settings, PB6 can be used as input to the inverting Oscillator amplifier and input to the internal clock operating circuit.

Depending on the clock selection fuse settings, PB7 can be used as output from the inverting Oscillator amplifier.

If the Internal Calibrated RC Oscillator is used as chip clock source, PB [7:6] is used as TOSC [2:1] input for the Asynchronous Timer/Counter2 if the AS2 bit in ASSR is set.

Port C (PC [5:0]):

Port C is a 7-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). The PC[5:0] output buffers have symmetrical drive characteristics with both high sink and source.

Port C pins that are externally pulled low will source current if the pull-up resistors are activated. The Port C pins are tri-stated when a reset condition becomes active, even if the clock is not running.

PC6/RESET:

If the RSTDISBL Fuse is programmed, PC6 is used as an I/O pin. Note that the electrical characteristics of PC6 differ from those of the other pins of Port C.

If the RSTDISBL Fuse is unprogrammed, PC6 is used as a Reset input. A low level on this pin for longer than the minimum pulse length will generate a Reset, even if the clock is not running. Shorter pulses are not guaranteed to generate a Reset.

Port D (PD[7:0]):

Port D is an 8-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). The Port D output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, Port D pins that are externally pulled low will source current if the pull-up resistors are activated. The Port D pins are tri-stated when a reset condition becomes active, even if the clock is not running.

AVCC: AVCC is the supply voltage pin for the A/D Converter, PC[3:0], and PE[3:2]. It should be externally connected to VCC, even if the ADC is not used. If the ADC is used, it should be connected to VCC through a low-pass filter. Note that PC[6:4] use digital supply voltage, VCC.

AREF: AREF is the analog reference pin for the A/D Converter.

ADC [7:6] (TQFP and VFQFN Package Only): In the TQFP and VFQFN package, ADC[7:6] serve as analog inputs to the A/D converter. These pins are powered from the analog supply and serve as 10-bit ADC channels.

12. ICSP pin: Mostly, ICSP (12) is an AVR, a tiny programming header for the Arduino consisting of MOSI, MISO, SCK, RESET, VCC, and GND. It is often referred to as an SPI (Serial Peripheral Interface), which could be considered as an "expansion" of the output. Actually, you are slaving the output device to the master of the SPI bus.

13. Power LED indicator: This LED should light up when you plug your Arduino into a power source to indicate that your board is powered up correctly. If this light does not turn on, then there is something wrong with the connection.

14. TX and RX LEDs: On your board, you will find two labels: TX (transmit) and RX (receive). They appear in two places on the Arduino UNO board. First, at the digital pins 0 and 1, to indicate the pins responsible for serial communication. Second, the TX and RX led (13). The TX led flashes with different speed while sending the serial data. The speed of flashing depends on the baud rate used by the board. RX flashes during the receiving process.

15. Digital I / O: The Arduino UNO board has 14 digital I/O pins (15) (of which 6 provide PWM

(Pulse Width Modulation) output. These pins can be configured to work as input digital pins to read logic values (0 or 1) or as digital output pins to drive different modules like LEDs, relays, etc. The pins labeled "~" can be used to generate PWM.

16. AREF: AREF stands for Analog Reference. It is sometimes, used to set an external reference voltage (between 0 and 5 Volts) as the upper limit for the analog input pinsworking.

3.2 POWER SUPPLY

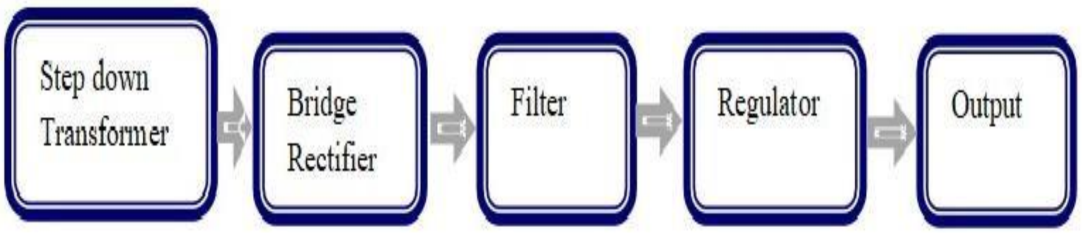


Fig 3.4: Block diagram for power supply

The input to the circuit is applied from the regulated power supply. The a.c. input i.e., 230V from the mains supply is step down by the transformer to 12V and is fed to a rectifier. The output obtained from the rectifier is a pulsating d.c voltage.

In order to get a pure d.c voltage, the output voltage from the rectifier is fed to a filter to remove any a.c components present even after rectification. Now, this voltage is given to a voltage regulator to obtain a pure constant dc voltage. The design of the power supply such as power consumption, voltage regulation, and energy efficiency, as many embedded systems. operate in environments where power resources are limited.

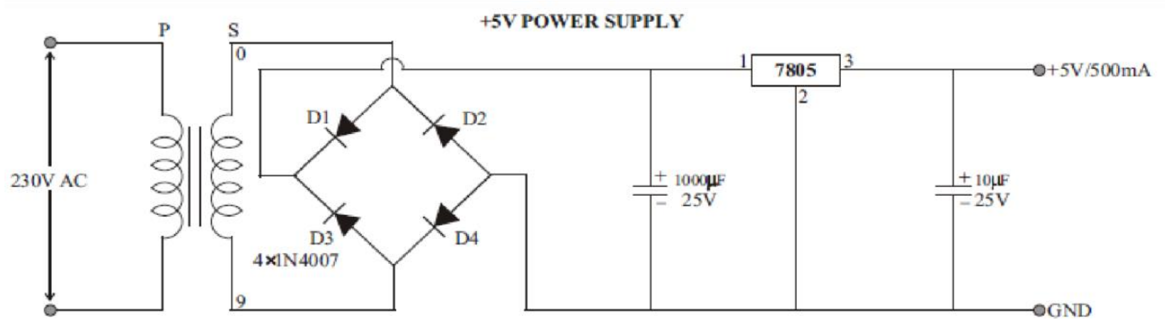


Fig 3.5: Circuit diagram of power supply

3.2.1 STEP DOWN TRANSFORMER

Usually, DC voltages are required to operate various electronic equipment and these voltages are 5V, 9V or 12V. But these voltages cannot be obtained directly. Thus the a.c input available at the mains supply i.e., 230V is to be brought down to the required voltage level. This is done by a transformer. Thus, a step down transformer is employed to decrease the voltage to a required level.

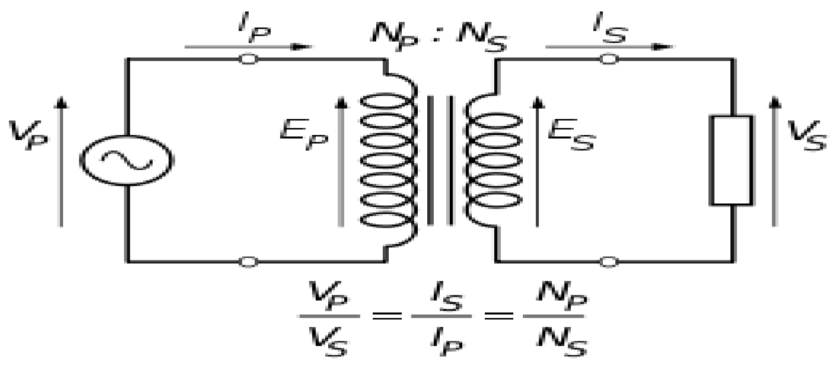


Fig 3.6: Step-down transformer

3.2.2 RECTIFIER:

Input **Output**

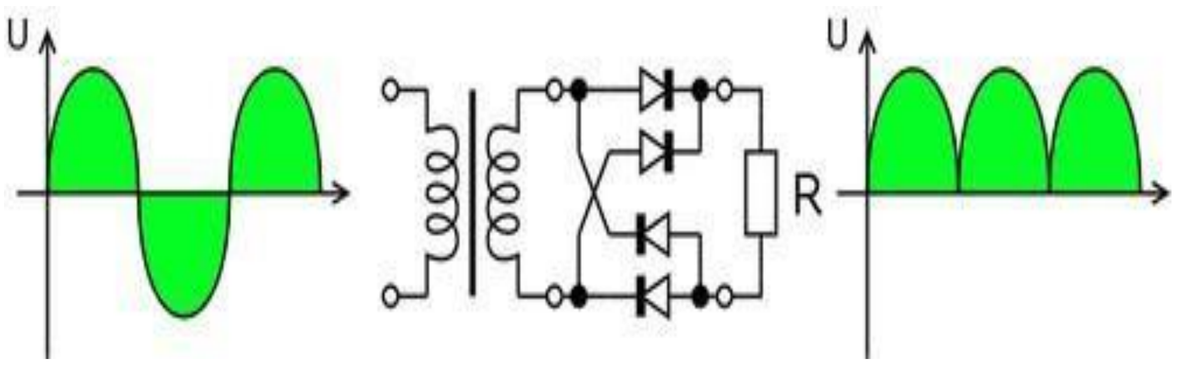


Fig 3.7: Bridge rectifier

The output from the transformer is fed to the rectifier. It converts A.C. into pulsating D.C. The rectifier may be a half wave or a full wave rectifier. In this project, a bridge rectifier is used because of its merits like good stability and full wave rectification.

3.2.3 FILTER

Capacitive filter is used in this project. It removes the ripples from the output of rectifier and smoothens the D.C. Output received from this filter is constant until the mains voltage and load is maintained constant. However, if either of the two is varied, D.C. voltage received at this point changes. Therefore a regulator is applied at the output stage.

3.2.4 VOLTAGE REGULATOR

As the name itself implies, it regulates the input applied to it. A voltage regulator is an electrical regulator designed to automatically maintain a constant voltage level. In this project, power supply of 5V and 12V are required. In order to obtain these voltage levels, 7805 and 7812 voltage regulators are to be used. The first number 78 represents positive supply and the numbers 05, 12 represent the required output voltage levels.

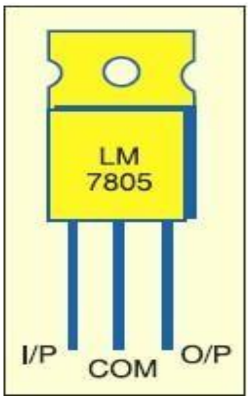


Fig 3.8: Voltage Regulator

Features:

• Output Current up to 1A.

• Output Voltages of 5, 6, 8, 9, 10, 12, 15, 18, 24V.

• Thermal Overload Protection.

• Short Circuit Protection.

• Output Transistor Safe Operating Area Protection.

3.3 L298N Module

The L298N Motor Driver module consists of an L298 Motor Driver IC, 78M05 Voltage Regulator, resistors, capacitor, Power LED, 5V jumper in an integrated circuit.

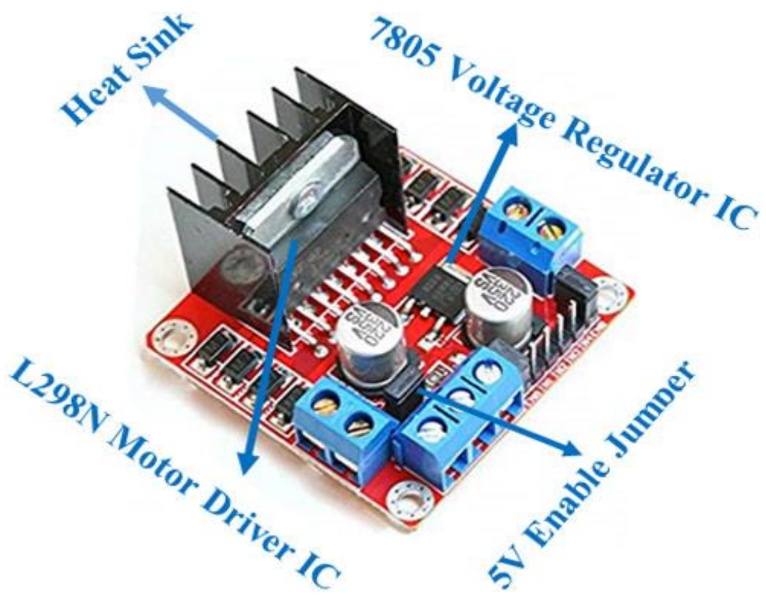


Fig 3.9: L298N Motor Driver Module

78M05 Voltage regulator will be enabled only when the jumper is placed. When the power supply is less than or equal to 12V, then the internal circuitry will be powered by the voltage regulator and the 5V pin can be used as an output pin to power the microcontroller. The jumper should not be placed when the power supply is greater than 12V and separate 5V should be given through 5V terminal to power the internal circuitry.

ENA & ENB pins are speed control pins for Motor A and Motor B while IN1& IN2 and IN3 & IN4 are direction control pins for Motor A and Motor B.

Internal circuit diagram of L298N Motor Driver module is given below:

Table 3.3: L298N Module Pinout Configuration

Pin Name	**Description**
IN1 & IN2	Motor A input pins. Used to control the spinning direction of Motor A
IN3 & IN4	Motor B input pins. Used to control the spinning direction of Motor B
ENA	Enables PWM signal for Motor A

ENB	Enables PWM signal for Motor B
OUT1 & OUT2	Output pins of Motor A
OUT3 & OUT4	Output pins of Motor B
12V	12V input from DC power Source
5V	Supplies power for the switching logic circuitry inside L298N IC
GND	Ground pin

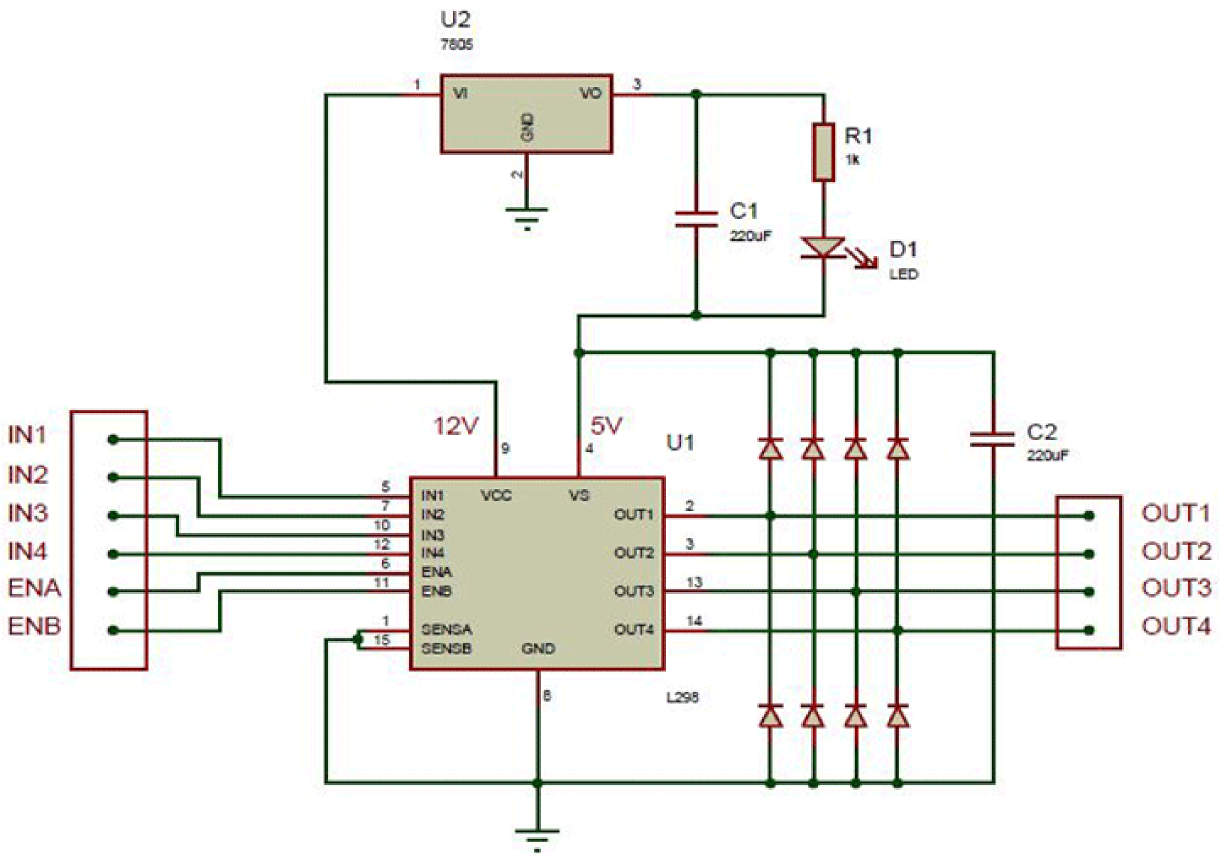


Fig 3.10: L298N Module Pinout

Applications

- Drive DC motors.
- Drive stepping motors
- In Robotics

Features & Specifications

- Driver Model: L298N 2A
- Driver Chip: Double H Bridge L298N
- Motor Supply Voltage (Maximum): 46V
- Motor Supply Current (Maximum): 2A
- Logic Voltage: 5V
- Driver Voltage: 5-35V
- Driver Current:2A
- Logical Current:0-36mA
- Maximum Power (W): 25W
- Current Sense for each motor
- Heatsink for better performance
- Power-On LED indicator

3.4 DC MOTOR

A 12-volt DC motor operating at 100 RPM (Revolutions Per Minute) is commonly used in applications requiring moderate speed and high torque. These motors convert electrical energy into mechanical energy through the interaction of magnetic fields within the motor's structure. The 12-volt power supply is typical for small industrial devices, robotics, automotive applications, and hobby projects.

Key Features:

- Voltage Rating: 12 V DC
- Speed: 100 RPM (can vary depending on load and motor design)
- Torque: High torque output suitable for heavy-load applications
- Current Draw: Varies with load; typically between 1-5 A
- Motor Type: Brushed or brushless DC motor (commonly brushed in this speed range)
- Efficiency: Moderate, with efficiency increasing under optimal load conditions

Working Principle:

A DC motor works on the principle of Lorentz Force, where a current-carrying conductor in a magnetic field experiences a force. This force causes the rotor (armature) to spin, converting electrical energy into mechanical motion. In a brushed DC motor, commutators and brushes.

Design and Construction:

Stator: Provides a magnetic field, either via permanent magnets or electromagnetic windings.
Rotor (Armature): A coil of wire that rotates within the magnetic field
Commutator & Brushes: Ensure the direction of current remains consistent to produce rotation.
Bearings: Reduce friction and support the rotating shaft.

Applications:

- Robotics: For precise control of wheel rotation and mechanical arms.
- Automotive: Power windows, seat adjustments, and other small mechanical systems.
- Industrial Equipment: Conveyor belts, mixers, and small machinery.
- Hobby Projects: DIY models, remote-controlled cars, and educational kits.

Advantages:

- Simple design and easy to control.
- High torque at low speeds.
- Cost-effective and widely available.

Disadvantages:

- Brushed motors require maintenance due to brush wear.
- Lower efficiency compared to brushless motors.
- Generates electrical noise (EMI) due to brush contact.
- Performance Characteristics:
- Speed-Torque Curve: The speed decreases as load increases, but torque increases.
- Load Response: Can maintain consistent torque under varying loads.
- Thermal Considerations: High current can cause overheating if not properly managed.
- A 12-volt DC motor with 100 RPM is a versatile and reliable component for various applications requiring moderate speed and good torque. Its simplicity, ease of control, and cost-effectiveness make it ideal for both industrial and hobbyist use.

3.5 NodeMCU 0.9

Pin mapping

Pin numbers written on the board itself do not correspond to ESP8266 GPIO pin numbers. Constants are defined to make using this board easier:

static	const	uint8_t	D0	=	16;
static	const	uint8_t	D1	=	5;
static	const	uint8_t	D2	=	4;
static	const	uint8_t	D3	=	0;
static	const	uint8_t	D4	=	2;
static	const	uint8_t	D5	=	14;
static	const	uint8_t	D6	=	12;
static	const	uint8_t	D7	=	13;
static	const	uint8_t	D8	=	15;
static	const	uint8_t	D9	=	3;
static	const	uint8_t	D10	=	1;

If you want to use NodeMCU pin 5, use D5 for pin number, and it will be translated to 'real' GPIO pin 14.

Node MCU 1.0

This module is sold under many names for around $6.50 on AliExpress and it's one of the cheapest, fully integrated ESP8266 solutions.

It's an open hardware design with an ESP-12E core and 4 MB of SPI flash.

According to the manufacturer, "with a micro USB cable, you can connect NodeMCU devkit to your laptop and flash itwithout any trouble". This is more or less true: the board comes with a CP2102 onboard USB to serial adapter which just works, well, the majority of the time. Sometimes flashing fails and you have to reset the board by holding down FLASH + RST, then releasing FLASH, then releasing RST. This forces the CP2102 device to power cycle and to be re-numbered by Linux.

The board also features a NCP1117 voltage regulator, a blue LED on GPIO16 and a 220k/100k Ohm voltage divideron the ADC input pin.

Olimex MOD-WIFI-ESP8266-DEV

This board comes with 2 MB of SPI flash and optional accessories (e.g. evaluation board ESP8266-EVB or BAT-BOXfor batteries).

The basic module has three solder jumpers that allow you to switch the operating mode between SDIO, UART and FLASH.

The board is shipped for FLASH operation mode, with jumpersTD0JP=0, IO0JP=1,IO2JP=1. According to the manufacturer, “with a micro USB cable, you can connect NodeMCU devkit to your laptop and flash itwithout any trouble”. This is more or less true: the board comes with a CP2102 onboard USB to serial adapter whichjust works, well, the majority of the time. Sometimes flashing fails and you have to reset the board by holding down FLASH + RST, then releasing FLASH, then releasing RST. This forces the CP2102 device to power cycle and to be re-numbered by Linux. It’s an open hardware design with an ESP-12E core and 4 MB of SPI flash.

According to the manufacturer, “with a micro USB cable, you can connect NodeMCU devkit to your laptop and flash itwithout any trouble”. This is more or less true: the board comes with a CP2102 onboard USB to serial adapter whichjust works, well, the majority of the time. Sometimes flashing fails and you have to reset the board by holding down FLASH + RST, then releasing FLASH, then releasing RST. This forces the CP2102 device to power cycle and to be re-numbered by Linux.

CHAPTER-4
SOFTWARE REQUIREMENTS

4.1 ARDUINO SOFTWARE

The Arduino is a family of microcontroller boards to simplify electronic design, prototyping and experimenting for artists, hackers, hobbyists, but also many professionals. People use it as brains for their robots, to build new digital music instruments, or to build a system that lets your house plants tweet you when they're dry. Arduinos (we use the standard Arduino Uno) are built around an ATmega microcontroller — essentially a complete computer with CPU, RAM, Flash memory, and input/output.

What you will need:

- A computer (Windows, Mac, or Linux)
- An Arduino-compatible microcontroller (anything from this guide should work)
- A USB A-to-B cable, or another appropriate way to connect your Arduinocompatible microcontroller to your computer.

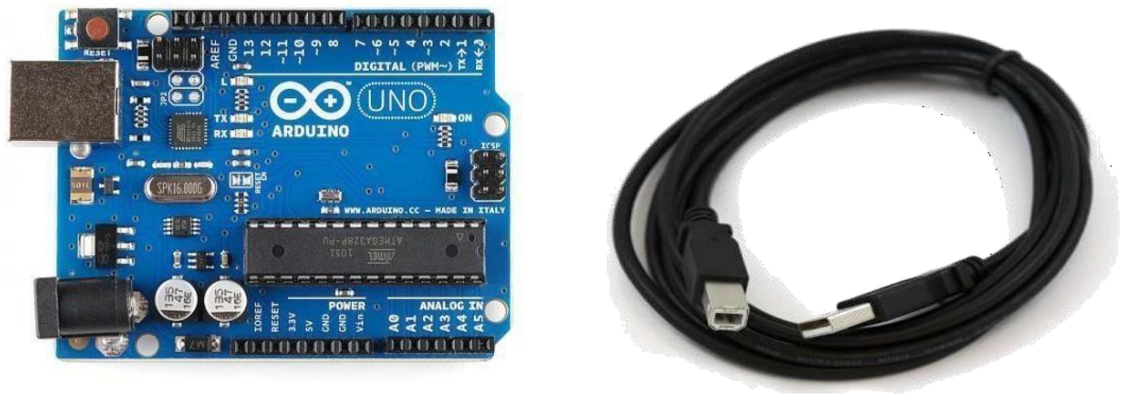


Fig 4.1: Arduino UNO

- An Arduino Uno
- Windows 7, Vista, and XP
- Installing the Drivers for the Arduino Uno (from Arduino.cc)
- Plug in your board and wait for Windows to begin it's driver installation process After a few moments, the process will fail, despite its best efforts
- Click on the Start Menu, and open up the Control Panel
- Look under Ports (COM & LPT). You should see an open port named "Arduino UNO (COMxx)".

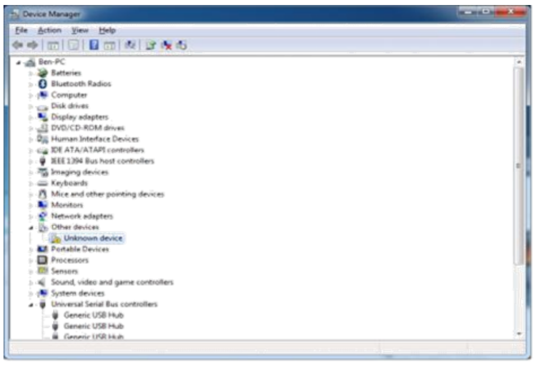
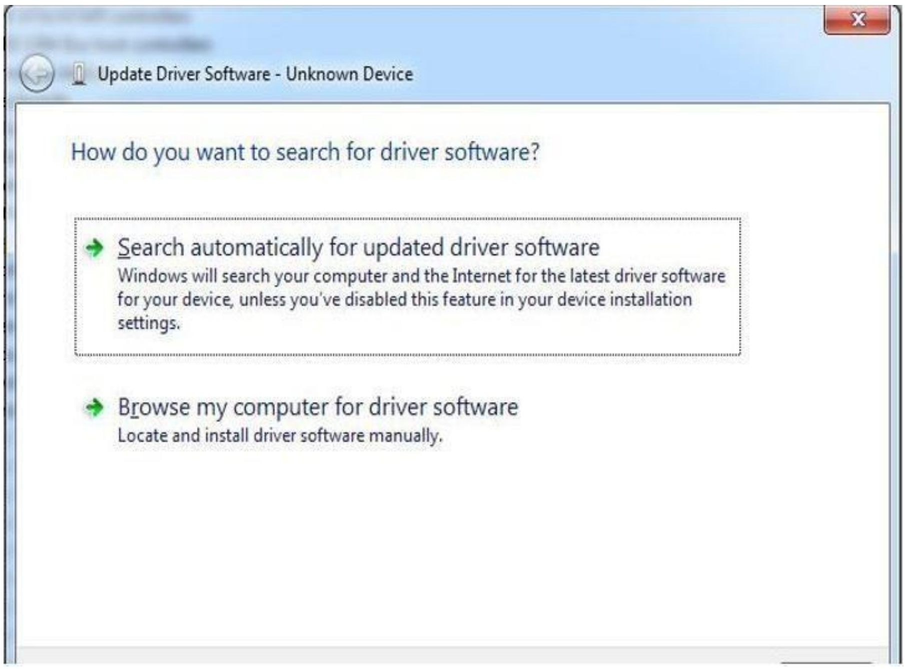


Fig 4.2: Device Manager

- If there is no COM & LPT section, look under 'Other Devices' for 'Unknown Device'.

  Right click on the "Arduino UNO (COMxx)" or "Unknown Device" port and choose the "Update Driver Software" opti Next, choose the "Browse my computer for Driver software" option.



Update Driver Software - Unknown Device

How do you want to search for driver software?

Search automatically for updated driver software
Windows will search your computer and the Internet for the latest driver software for your device, unless you've disabled this feature in your device installation settings.

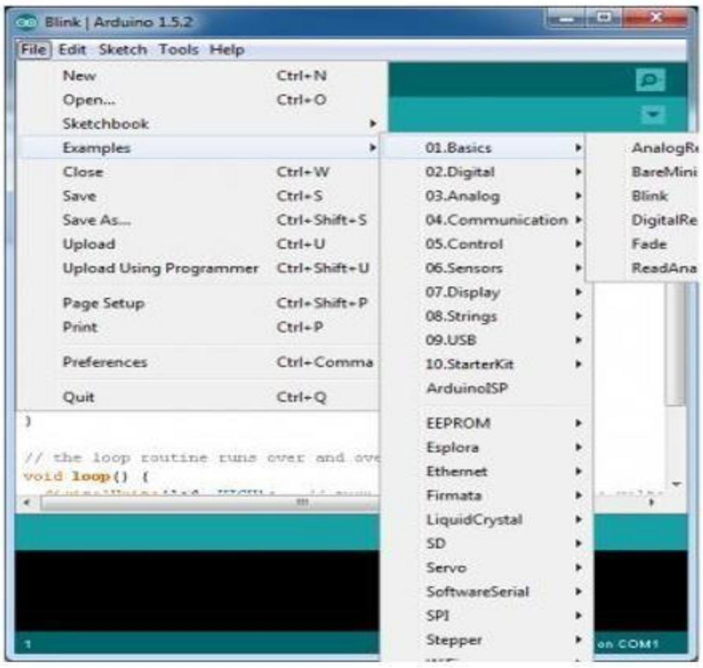
Browse my computer for driver software
Locate and install driver software manually.

Fig 4.3: Update Driver Software

- Finally, navigate to and select the Uno's driver file, named "ArduinoUNO.inf", located in the "Drivers" folder of the Arduino Software download (not the "FTDI USB Drivers" sub-directory).
- If you cannot see the .inf file, it is probably just hidden. You can select the 'drivers' folder with the 'search sub-folders' option selected instead. ○
  Windows will finish up the driver installation.

After following the appropriate steps for your software install, we are now ready to test your first program with your Arduino board!

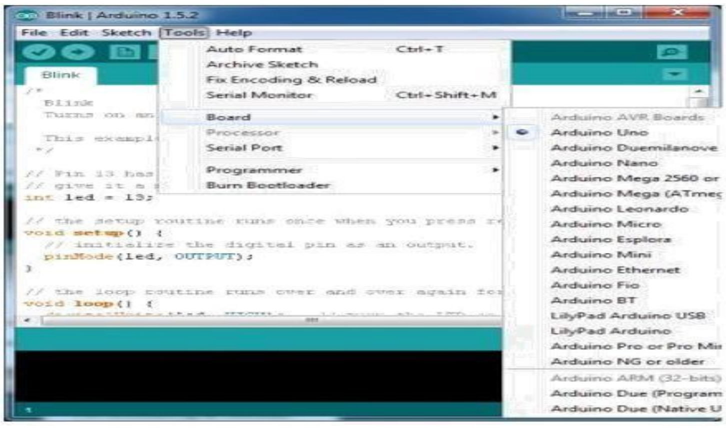
- Launch the Arduino application
- If you disconnected your board, plug it back in
- Open the Blink example sketch by going to: File > Examples > 1.Basics > Blink
- After a second, you should see some LEDs flashing on your Arduino, followed by the message 'Done Uploading' in the status bar of the Blink sketch.



```
}

// the loop routine runs over and ove
void loop() {
```

Fig 4.4: Arduino File basics

- If everything worked, the onboard LED on your Arduino should now be blinking! You just programmed your first Arduino!
- Select the type of Arduino board you're using: Tools > Board > your board type

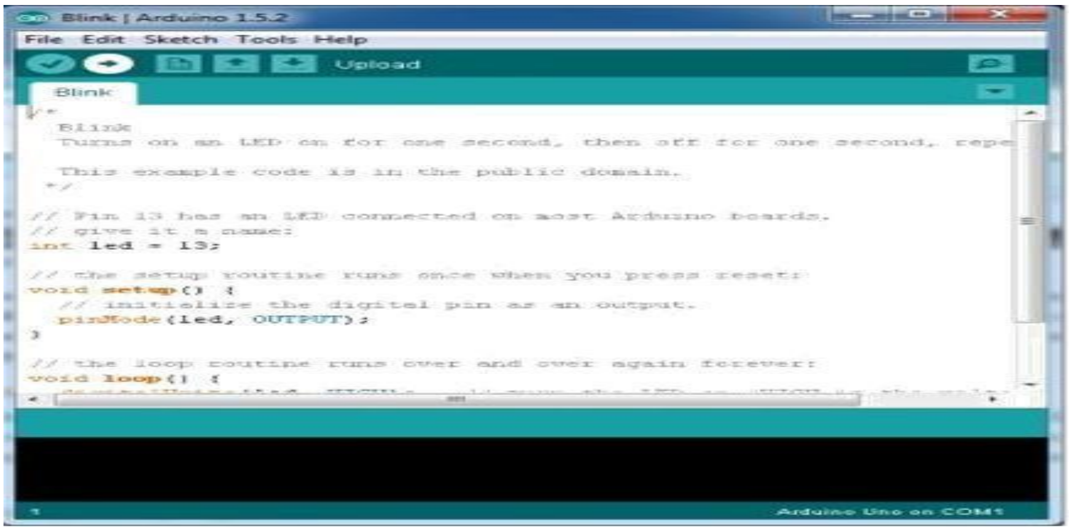


```
/*
  Blink
  Turns on an
  This exampl
 */

// Pin 13 has
// give it a
int led = 13;

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

// the loop routine runs over and over again fo
void loop() {
```

Fig 4.5: Arduino Tools Board

- Select the serial/COM port that your Arduino is attached to: Tools > Port > COMxx.

- If you’re not sure which serial device is your Arduino, take a look at the available ports, then unplug your Arduino and look again.
- The one that disappeared is your Arduino. With your Arduino board connected, and the Blink sketch open, press the ‘Upload’ button.



```
/*
  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.
// 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);
}

// the loop routine runs over and over again forever:
void loop() {
```

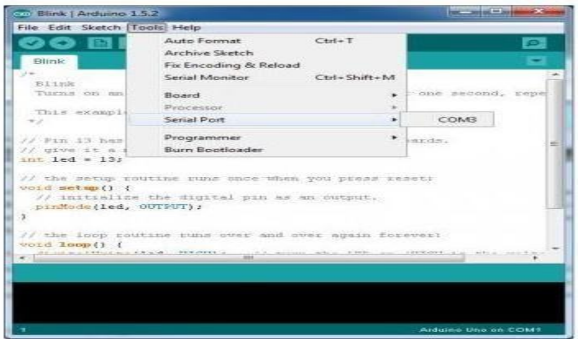


Fig 4.6: Blink Arduino Tools

CHAPTER -5
WORKING MODEL AND ITS COMPONENTS

5.1 BLOCK DIAGRAM

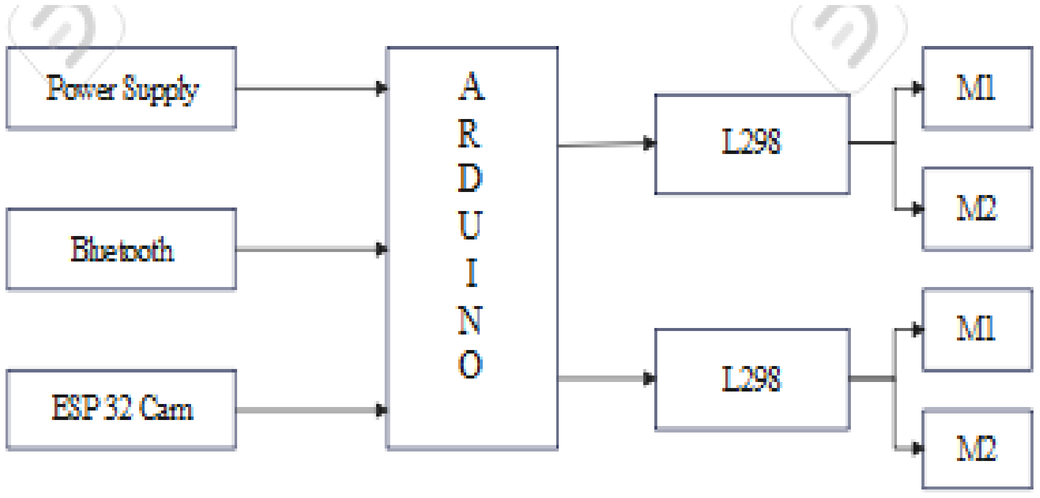
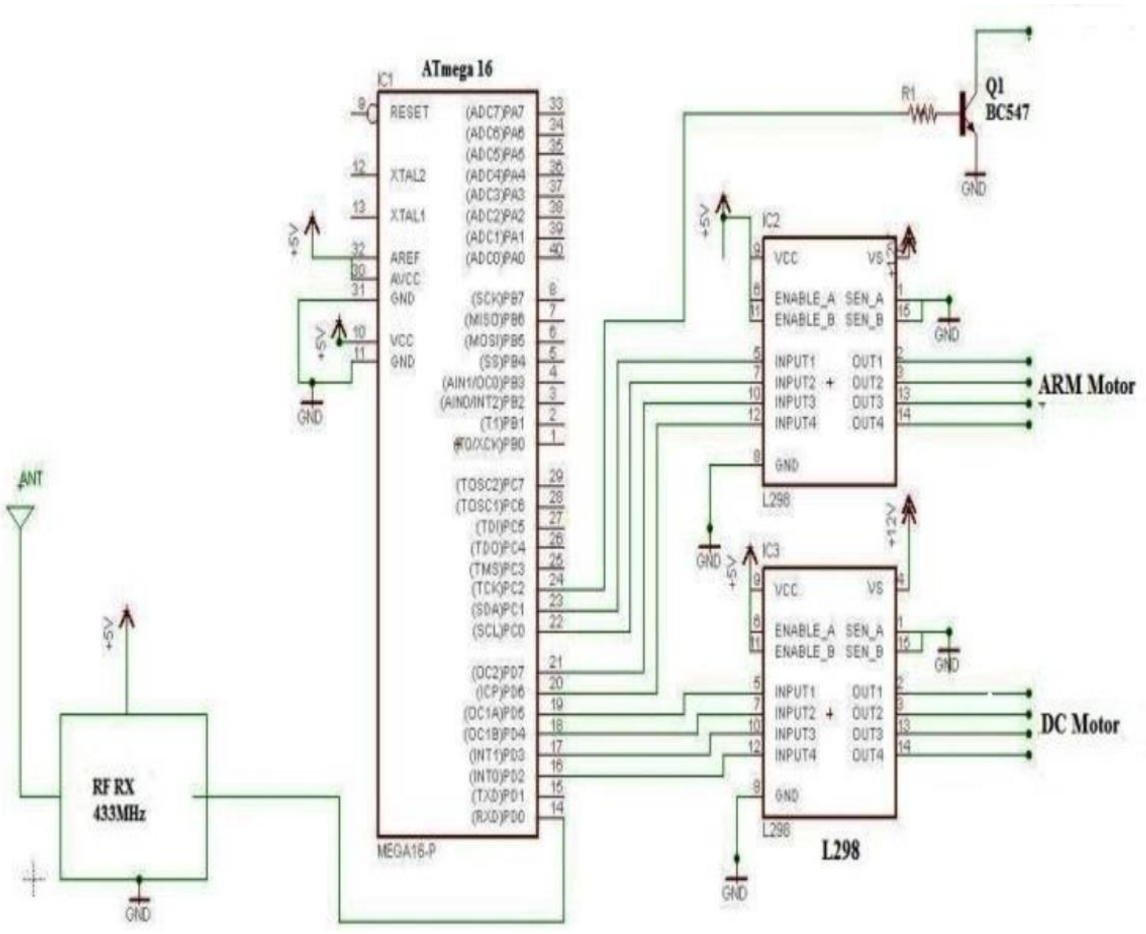


Fig 5.1: Block Diagram

5.2 WORKING



5.2.1 INTRODUCTION TO ARDUINO:

The Arduino is a family of microcontroller boards to simplify electronic design, prototyping and experimenting for artists, hackers, hobbyists, but also many professionals. People use it as brains for their robots, to build new digital music instruments, or to build a system that lets your house plants tweet you when they're dry. Arduinos (we use the standard Arduino Uno) are built around an ATmega microcontroller — essentially a complete computer with CPU, RAM, Flash memory, and input/output pins, all on a single chip. Unlike, say, a Raspberry Pi, it's designed to attach all kinds of sensors, LEDs, small motors and speakers, servos, etc. directly to these pins, which can read in or output digital or analog voltages between 0 and 5 volts.

The Arduino connects to your computer via USB, where you program it in a simple language (C/C++, similar to Java) from inside the free Arduino IDE by uploading your compiled code to the board. Once programmed, the Arduino can run with the USB link back to your computer, or stand-alone without it — no keyboard or screen needed, just power. Arduino boards are microcontroller-based platforms that enable users to create various electronic projects.

The Arduino is a family of microcontroller boards to simplify electronic design, prototyping and experimenting for artists, hackers, hobbyists, but also many professionals. People use it as brains for their robots, to build new digital music instruments, or to build a system that lets your house plants tweet you when they're dry. The Arduino is a family of microcontroller boards to simplify electronic design, prototyping and experimenting for artists, hackers, hobbyists, but also many professionals. People use it as brains for their robots, to build new digital music instruments, or to build a system that lets your house plants tweet you when they're dry. Arduinos (we use the standard Arduino Uno) are built around an ATmega microcontroller — essentially a complete computer with CPU, RAM, Flash memory, and input/output pins, all on a single chip. Unlike, say, a Raspberry Pi, it's designed to attach all kinds of sensors, LEDs, small motors and speakers, servos, etc. directly to these pins, which can read in or output digital or analog voltages between 0 and 5 volts.

The Arduino is a family of microcontroller boards to simplify electronic design, prototyping and experimenting for artists, hackers, hobbyists, but also many professionals. People use it as brains for their robots, to build new digital music instruments, or to build a system that lets your house plant.

The most common board is the Arduino Uno, which features an a Tmega328P microcontroller, but there are several other models tailored for specific applications, such as the Arduino Mega, Nano, and Leonardo.

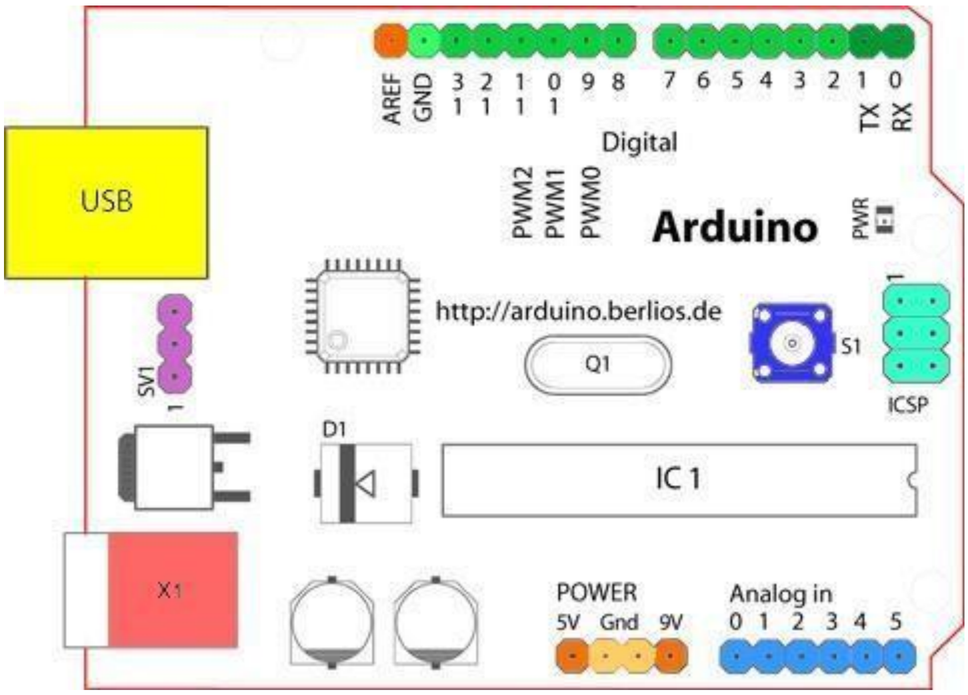


Fig 5.2: Structure of Arduino Board

Looking at the board from the top down, this is an outline of what you will see (parts of the board you might interact with in the course of normal use are highlighted)

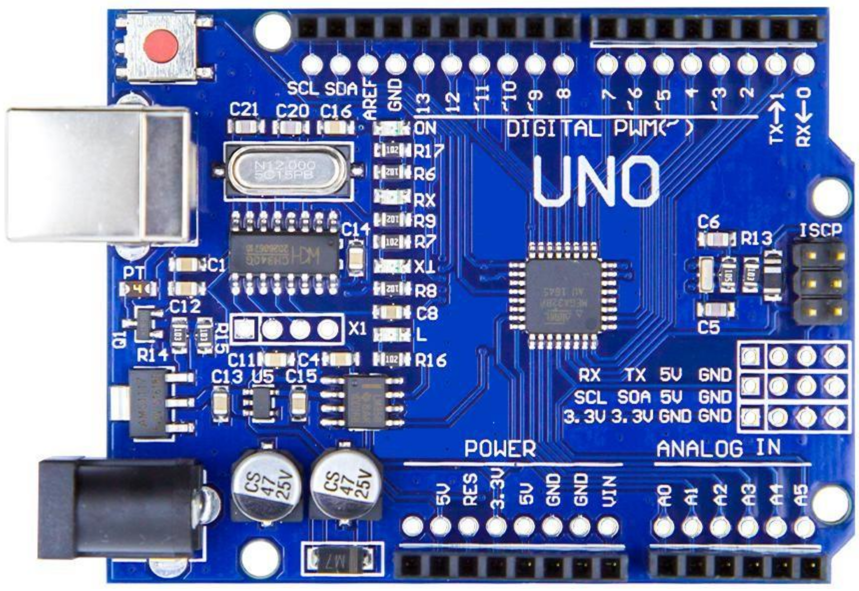


Fig5.3: Arduino Board

Starting clockwise from the top center:

- Analog Reference pin (orange)
- Digital Ground (light green)
- Digital Pins 2-13 (green)

- Digital Pins 0-1/Serial In/Out - TX/RX (dark green) - These pins cannot be used for digital i/o (Digital Read and Digital Write) if you are also using serial communication (e.g. Serial.begin).
- Reset Button - S1 (dark blue)
- In-circuit Serial Programmer (blue-green)
- Analog In Pins 0-5 (light blue)
- Power and Ground Pins (power: orange, grounds: light orange)
- External Power Supply In (9-12VDC) - X1 (pink)
- Toggles External Power and USB Power (place jumper on two pins closest to desired supply) - SV1 (purple)
- USB (used for uploading sketches to the board and for serial communication between the board and the computer; can be used to power the board) (yellow)

DIGITAL PINS

In addition to the specific functions listed below, the digital pins on an Arduino board can be used for general purpose input and output via the pin mode(), Digital read(), and Digital write() commands. Each pin has an internal pull-up resistor which can be turned on and off using digital Write() (w/ a value of HIGH or LOW, respectively) when the pin is configured as an input. The maximum current per pin is 40mA.

- **Serial: 0 (RX) and 1 (TX).** Used to receive (RX) and transmit (TX) TTL serial data. On the Arduino Diecimila, these pins are connected to the corresponding pins of the FTDI USB-to-TTL Serial chip. On the Arduino BT, they are connected to the corresponding pins of the WT11 Bluetooth module. On the Arduino Mini and LilyPad Arduino, they are intended for use with an external TTL serial module (e.g. the Mini-USB Adapter).

- **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 attach Interrupt(), function for details.

- **PWM: 3, 5, 6, 9, 10, and 11** Provide 8-bit PWM output with the analog write() function. On boards with an ATmega8, PWM output is available only on pins 9, 10, and 11.

- **BT Reset: 7.** (Arduino BT-only) Connected to the reset line of the Bluetooth module.

- **SPI: 10 (SS), 11 (MOSI), 12 (MISO), 13 (SCK).** These pins support SPI communication, which, although provided by the underlying hardware, is not currently included in the Arduino language.
- **LED: 13.** On the Decimole and Lilypad, there is a built-in LED connected to digital pin 13. When the pin is HIGH value, the LED is on, when the pin is LOW, it's off.

ANALOG PINS

In addition to the specific functions listed below, the analog input pins support 10-bit analog-to-digital conversion (ADC) using the analog Read() function. Most of the analog inputs can also be used as digital pins: analog input 0 as digital pin 14 through analog input 5 as digital pin 19. Analog inputs 6 and 7 (present on the Mini and BT) cannot be used as digital pins.

- **I^{2}C: 4 (SDA) and 5 (SCL).** Support I^{2}C (TWI) communication using the Wire library (documentation on the Wiring website).

POWER PINS

- **VIN** (sometimes labelled "9V"): The input voltage to the Arduino 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. Also note that the Lily Pad has no VIN pin and accepts only a regulated input.
- **5V:** The regulated power supply used to power the microcontroller and other components on the board. This can come either from VIN via an on-board regulator, or be supplied by USB or another regulated 5V supply.
- **3V3** (Decimole-only) : A 3.3 volt supply generated by the on-board FTDI chip.
- **GND:** Ground pins.

OTHER PINS

- **AREF:** Reference voltage for the analog inputs. Used with analog Reference().
- **Reset:** (decimole-only) Bring this line LOW to reset the microcontroller. Typically used to add a reset button to shields which block the one on the board.

ATMEGA328

The ATmega328 is an 8-bit microcontroller based on the AVR architecture. It is popular for its balance of performance, power consumption, and ease of use, making it a favourite among hobbyists and professionals for various electronics projects.

The ATmega328 can be programmed using the Arduino IDE, which simplifies the process with a user-friendly interface and a set of libraries. Users typically write in a simplified version of C/C++. The IDE also provides built-in functions that allow for easy interaction with the microcontroller's hardware features.

PIN DIAGRAM

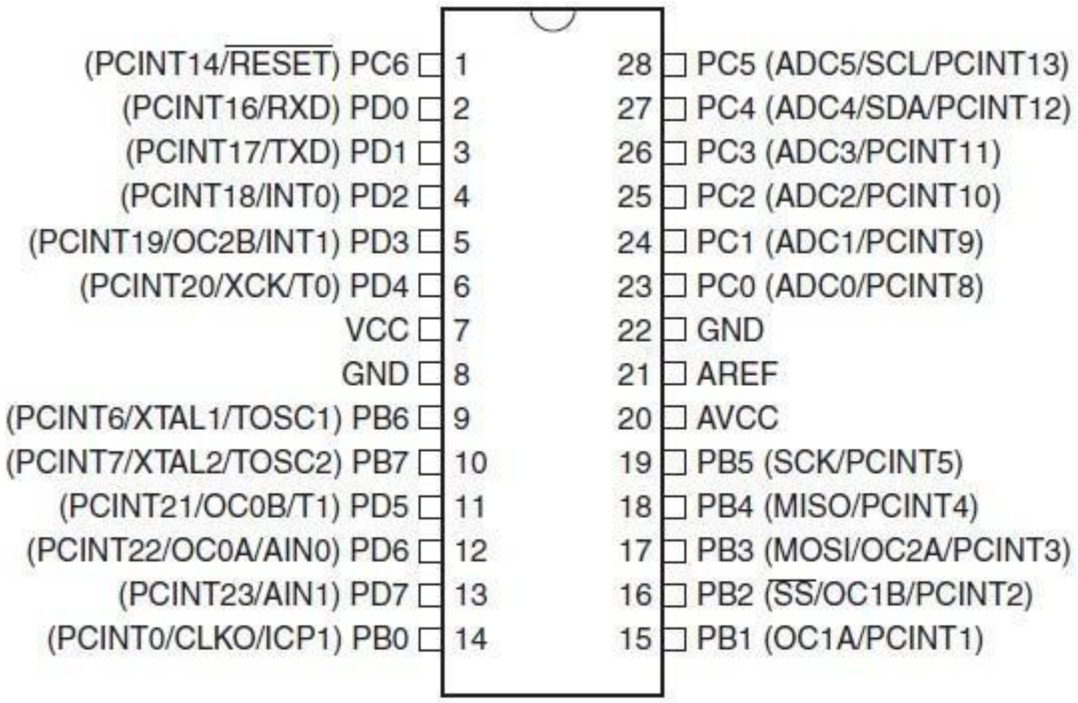


Fig 5.4: Pin Configuration of Atmega328

Pin Description VCC:

Digital supply voltage. GND:

Ground.

Port A (PA7-PA0):

Port A serves as the analog inputs to the A/D Converter. Port A also serves as an 8-bit bidirectional I/O port, if the A/D Converter is not used. Port pins can provide internal pull- up resistors (selected for each bit). The Port A output buffers have symmetrical drive characteristics with both high sink and source capability. When pins PA0 to PA7 are used as inputs and are externally pulled low, they will source current if the internal pull-up resistors are activated. The Port A pins are tri-stated when a reset condition becomes active, even if the clock is not running.

Port B (PB7-PB0):

Port B is an 8-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). The Port B output buffers have symmetrical drive characteristics with both high sink and source.

Port B is an 8-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). The Port B output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, Port B pins that are externally pulled low will source current if the pull-up resistors are activated. The Port B pins are tri-stated when a reset condition becomes active, even if the clock is not running. Port B also serves the functions of various special features of the ATmega32.

Port C (PC7-PC0):
Port C is an 8-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). The Port C output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, Port C pins that are externally pulled low will source current if the pull-up resistors are activated. The Port C pins are tri-stated when a reset condition becomes active, even if the clock is not running. If the JTAG interface is enabled, the pull- up resistors on pins PC5(TDI), PC3(TMS) and PC2(TCK) will be activated even if a reset occurs. The TD0 pin is tri-stated unless TAP states that shift out data are entered.

Port D (PD7-PD0):
Port D is an 8-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). The Port D output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, Port D pins that are externally pulled low will source current if the pull-up resistors are activated. The Port D pins are tri-stated when a reset condition becomes active, even if the clock is not running. Port D also serves the functions of various special features of the ATmega32.

Reset (Reset Input):
A low level on this pin for longer than the minimum pulse length will generate a reset, even if the clock is not running. Shorter pulses are not guaranteed to generate a reset.

XTAL1:
Input to the inverting Oscillator amplifier and input to the internal clock operating circuit.
XTAL2:
Output from the inverting Oscillator amplifier. AVCC:
AVCC is the supply voltage pin for Port A and the A/D Converter. It should be externally connected to VCC, even if the ADC is not used. If the ADC is used, it should be connected to VCC through a low-pass filter.

AREF: AREF is the analog reference pin for the A/D Converter.

FEATURES

- 1.8-5.5V operating range
- Up to 20MHz
- Part: ATMEGA328P-AU
- 32kB Flash program memory
- 1kB EEPROM
- 2kB Internal SRAM
- 2 8-bit Timer/Counters
- 16-bit Timer/Counter
- RTC with separate oscillator
- 6 PWM Channels
- 8 Channel 10-bit ADC
- Serial USART
- Master/Slave SPI interface
- 2-wire (I2C) interface
- Watchdog timer
- Analog comparator
- 23 IO lines
- Data retention: 20 years at 85C/ 100 years at 25C
- Digital I/O Pins are 14 (out of which 6 provide PWM output) o Analog Input Pins are 6.
- DC Current per I/O is 40 mA
- DC Current for 3.3V Pin is 50mA

AVR CPU CORE

The AVR core combines a rich instruction set with 32 general purpose working registers. All the 32 registers are directly connected to the Arithmetic Logic Unit (ALU), allowing two independent registers to be accessed in one single instruction executed in one clock cycle. The resulting architecture is more code efficient while achieving throughputs up to ten times faster than conventional CISC microcontrollers.

In order to maximize performance and parallelism, the AVR uses a Harvard architecture with separate memories and buses for program and data. Instructions in the program memory are executed with a single level pipelining. While one instruction is being executed, the next instruction is pre-fetched from the program memory. This concept enables instructions to be executed in every clock cycle. The program memory is In-System Reprogrammable Flash memory. The fast-access Register File contains 32 x 8-bit general purpose working registers with a single clock cycle access time. This allows single-cycle Arithmetic Logic Unit (ALU) operation. In a typical ALU operation, two operands are output from the Register File, the operation is executed, and the result is stored back in the Register File– in one clock cycle.

The main function of the CPU core is to ensure correct program execution. The AVR CPU is capable to access memories, perform calculations, control peripherals, and handle interrupts.

OVERVIEW

This section discusses the AVR core architecture in general. The main function of the CPU core is to ensure correct program execution. The CPU must therefore be able to access memories, perform calculations, control peripherals, and handle interrupts.

Six of the 32 registers can be used as three 16-bit indirect address register pointers for Data Space addressing – enabling efficient address calculations. One of these address pointers can also be used as an address pointer for look up tables in Flash program memory. These added function registers are the 16-bit X-, Y-, and Z-register, described later in this section.

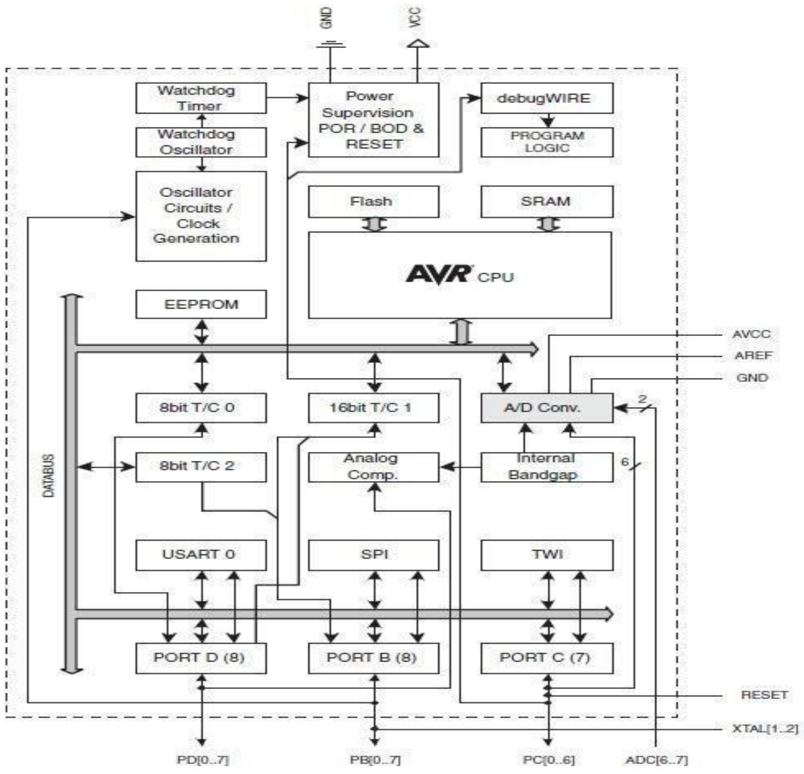


Fig 5.5: Block Diagram

The ALU supports arithmetic and logic operations between registers or between a constant and a register. Single register operations can also be executed in the ALU. After an arithmetic operation, the Status Register is updated to reflect information about the result of the operation. An advanced version of a microcomputer that is integrated into a tiny chip is known as the AVR microcontroller. This microcontroller includes a processor, programmable I/O peripherals & memory. The memory spaces in the AVR architecture are all linear and regular memory maps. A flexible interrupt module has its control registers in the I/O space with an additional Global Interrupt Enable bit in the Status Register. All interrupts have a separate Interrupt Vector in the Interrupt Vector table. The interrupts have priority in accordance with their Interrupt Vector position. The lower the Interrupt Vector address, the higher the priority.

Program flow is provided by conditional and unconditional jump and call instructions, able to directly address the whole address space. Most AVR instructions have a single 16-bit word format. Every program memory address contains a 16- or 32-bit instruction.

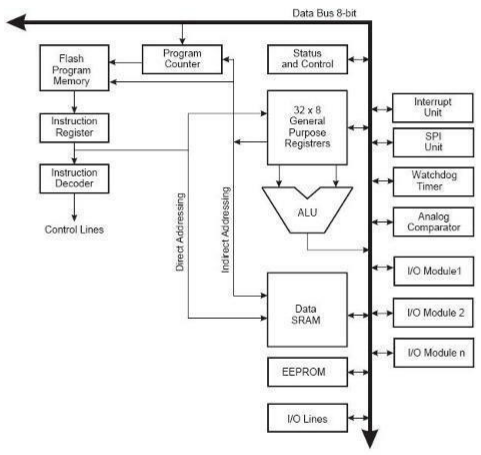


Fig 5.6: AVR core architecture

Program Flash memory space is divided in two sections, the Boot Program section and the Application Program section. Both sections have dedicated Lock bits for write and read/write protection. The SPM instruction that writes into the Application Flash memory section must reside in the Boot Program section. During interrupts and subroutine calls, the return address Program Counter (PC) is stored on the Stack. The Stack is effectively allocated in the general data SRAM, and consequently the Stack size is only limited by the total SRAM size and the usage of the SRAM. All user programs must initialize the SP in the Reset routine (before subroutines or interrupts are executed). The Stack Pointer (SP) is read/write accessible in the I/O space. The data SRAM can easily be accessed through the five different addressing modes supported in the AVR architecture.

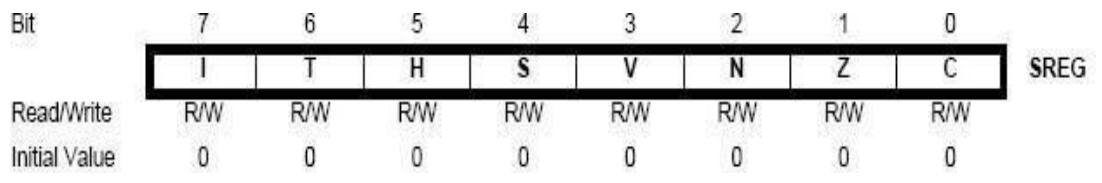
ALU – ARITHMETIC LOGIC UNIT

The high-performance AVR ALU operates in direct connection with all the 32 general purpose working registers. Within a single clock cycle, arithmetic operations between general purpose registers or between a register and an immediate are executed. The ALU operations are divided into three main categories – arithmetic, logical, and bit functions. Some implementations of the architecture also provide a powerful multiplier supporting both signed/unsigned multiplication and fractional format. See the "Instruction Set" section for a detailed description.

STATUS REGISTER

The Status Register contains information about the result of the most recently executed arithmetic instruction. This information can be used for altering program flow in order to perform conditional operations. Note that the Status Register is updated after all ALU operations, as specified in the Instruction Set Reference. This will in many cases remove the need for using the dedicated compare instructions, resulting in faster and more compact code. The Status Register is not automatically stored when entering an interrupt routine and restored when returning from an interrupt. This must be handled by software.

The AVR Status Register – SREG is defined as:



Bit	7	6	5	4	3	2	1	0	
	I	T	H	S	V	N	Z	C	SREG
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

Fig 5.7: AVR status register

Bit 7 – I: Global Interrupt Enable

The Global Interrupt Enable bit must be set for the interrupts to be enabled. The individual interrupt enable control The Global Interrupt Enable bit must be set for the interrupts to be enabled. The individual interrupt enable control is then performed in separate control registers. If the Global Interrupt Enable Register is cleared, none of the interrupts are enabled independent of the individual interrupt enable settings. The I-bit is cleared by hardware after an interrupt has occurred, and is set by the RETI instruction to enable subsequent interrupts. The I-bit can also be set and cleared by the application with the SEI and CLI instructions, as described in the instruction set reference.

Bit 6 – T: Bit Copy Storage
A bit from a register in the Register File can be copied into T by the BST instruction, and a bit in T can be copied into a bit in a register in the Register File by the BLD instruction.

Bit 5 – H: Half Carry Flag

The Half Carry Flag H indicates a Half Carry in some arithmetic operations The Half Carry Flag H indicates a Half Carry in some arithmetic operations. Half Carry Is useful in BCD arithmetic. See the "Instruction Set Description" for detailed information.

Bit 4 – S: Sign Bit

The S-bit is always an exclusive or between the Negative Flag N and the Two's Complement Overflow Flag V. See the "Instruction Set Description" for detailed information.

Bit 3 – V: Two's Complement Overflow Flag

The Two's Complement Overflow Flag V supports two's complement arithmetic.

Bit 2 – N: Negative Flag

The Negative Flag N indicates a negative result in an arithmetic or logic operation.

Bit 1 – Z: Zero Flag

The Zero Flag Z indicates a zero result in an arithmetic or logic operation.

Bit 0 – C: Carry Flag

The Carry Flag C indicates a carry in an arithmetic or logic operation.

STACK POINTER

The Stack is mainly used for storing temporary data, for storing local variables and for storing return addresses after interrupts and subroutine calls. Note that the Stack is implemented as growing from higher to lower memory locations. The Stack Pointer Register always points to the top of the Stack. The Stack Pointer points to the data SRAM Stack area where the Subroutine and Interrupt Stacks are located. A Stack PUSH command will decrease the Stack Pointer.

The Stack in the data SRAM must be defined by the program before any subroutine calls are executed or interrupts are enabled. Initial Stack Pointer value equals the last address of the internal SRAM and the Stack Pointer must be set to point above start of the SRAM.

The AVR ATmega128A Stack Pointer is implemented as two 8-bit registers in the I/O space. The number of bits actually used is implementation dependent.

Note that the data space in some implementations of the AVR architecture is so small that only SPL is needed. In this case, the SPH Register will not be present.SPH and SPL - Stack Pointer High and Low Register.

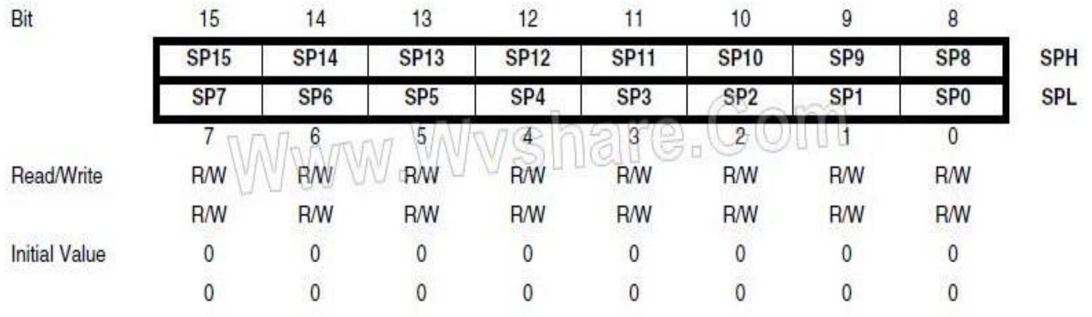
Table 5.2: Stack Pointer instructions

Instruction	Stack pointer	Description
PUSH	Decremented by 1	Data is pushed onto the stack
CALL , ICALL RCALL	Decremented by 2	Return address is pushed onto the stack with a subroutine call or interrupt
POP	Incremented by 1	Data is popped from the stack
RET RETI	Incremented by 2	Return address is popped from the stack with return from subroutine or return from interrupt

INTERRUPT RESPONSE TIME

The interrupt execution response for all the enabled AVR interrupts is four clock cycles minimum. After four clock cycles the program vector address for the actual interrupt handling routine is executed.

During this four clock cycle period, the Program Counter is pushed onto the Stack. The vector is normally a jump to the interrupt routine, and this jump takes three clock cycles. If an interrupt occurs during execution of a multi-cycle instruction, this instruction is completed before the interrupt is served. If an interrupt occurs when the MCU is in sleep mode, the interrupt execution response time is increased by four clock cycles. This increase comes in addition to the start-up time from the selected sleep mode.



Bit	15	14	13	12	11	10	9	8	
	SP15	SP14	SP13	SP12	SP11	SP10	SP9	SP8	SPH
	SP7	SP6	SP5	SP4	SP3	SP2	SP1	SP0	SPL
	7	6	5	4	3	2	1	0	
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	
	0	0	0	0	0	0	0	0	

Fig 5.8: SPH and SPL - Stack Pointer High and Low Register

AVR Memories

This section describes the different memories in the ATmega328. The AVR architecture has two main memory spaces, the Data Memory and the Program Memory space. In addition, theATmega328 features an EEPROM Memory for data storage. All three memory spaces are linear and regular.

In-System Reprogrammable Flash Program Memory:

The ATmega328 contains 4/8/16/32Kbytes On-chip In-System Reprogrammable Flash memory for program storage. Since all AVR instructions are 16 or 32 bits wide, the Flash is organized as 2/4/8/16K x 16. For software security, the Flash Program memory space is divided into two sections, Boot Loader Section and Application Program Section. The Flash memory has an endurance of at least 10,000 write/erase cycles. The ATmega328 Program Counter (PC) is 11/12/13/14 bits wide, thus addressing the 2/4/8/16K program memory locations.

SRAM Data Memory:

ATmega328 is a complex microcontroller with more peripheral units than can be supported within the 64 locations reserved in the Opcode for the IN and OUT instructions. For the Extended I/O space from 0x60 - 0xFF in SRAM, only the ST/STS/STD and LD/LDS/LDD instructions can be used.

The lower 768/1280/1280/2303 data memory locations address both the Register File, the I/O memory, Extended I/O memory, and the internal data SRAM. The first 32 locations address the Register File, the next 64 location the standard I/O memory, then 160 locations of Extended I/O memory, and the next 512/1024/1024/2048 locations address the internal data SRAM. The five different addressing modes for the data memory cover: Direct, Indirect with Displacement, Indirect, Indirect with Pre-decrement, and Indirect with Post-increment. In The Register File, Registers R26 to R31 Feature the indirect addressing pointer registers. The direct addressing reaches the entire data space. The Indirect with Displacement mode reaches 63 address locations from the base address given by the Y- or Z register.

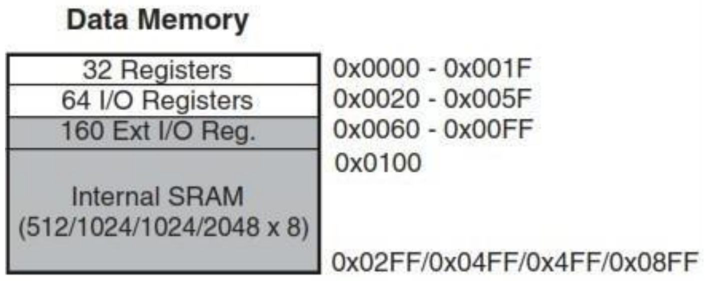


Fig 5.9: Data Memory Map

When using register indirect addressing modes with automatic pre-decrement and postincrement, the address registers X, Y, and Z are decremented or incremented. The 32 general purpose working registers, 64 I/O Registers, 160 Extended I/O Registers, and the 512/1024/1024/2048 bytes of internal data SRAM in the ATmega328 are all accessible through all these addressing modes.

INTERRUPTS

This section describes the specifics of the interrupt handling as performed in the Atmega328. In Atmega328Each Interrupt Vector occupies two instruction words and the Reset Vector is affected by the BOOTRST fuse, and the Interrupt Vector start address is affected by the IVSEL bit in MCUCR.

When the IVSEL bit in MCUCR is set, Interrupt Vectors will be moved to the start of the Boot Flash Section. The address of each Interrupt Vector will then be the address in this table added to the start address of the Boot Flash Section.Table below shows reset and Interrupt Vectors placement for the various combinations of BOOTRST and IVSEL settings. 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 versionR2) programmed as a USB- to-serial converter.

Table 5.3: Reset and Interrupt Vectors in ATMEGA 328 and ATMEGA 328P

Vector No.	**Program Address**	**Source**	**Interrupt Definition**
1	0x0000	RESET	External Pin, Power-on Reset, Brown-out Reset and Watchdog System Reset
2	0x0002	INT0	External Interrupt Request 0
3	0x0004	INT1	External Interrupt Request 0
4	0x0006	PCINTO	Pin Change Interrupt Request 0
5	0x0008	PCINT1	Pin Change Interrupt Request 1
6	0x000A	PCINT2	Pin Change Interrupt Request 2
7	0x000C	WDT	Watchdog Time-out Interrupt
8	0x000E	TIMER2 COMPA	Timer/Counter2 Compare Match A
9	0x0010	TIMER2 COMPB	Timer/Counter2 Compare Match B
10	0x0012	TIMER2 OVF	Timer/Counter 2 Overflow

11	0x0014	TIMER1 CAPT	Timer/Counter 2 Capture Event
12	0x0016	TIMER1 COMPA	Timer/Counter1 Compare Match A
13	0x0018	TIMER1 COMPB	Timer/Counter1 Compare Match B
14	0x001A	TIMER 1 OVF	Timer/Counter1 Overflow
15	0x001C	TIMER0 COMPA	Timer/Counter0 Compare Match A
16	0x001E	TIMER0 COMPB	Timer/Counter0 Compare Match B
17	0x0020	TIME0 OVF	Timer/Counter0 Overflow
18	0x0022	SPI, STC	SPI Serial Transfer Complete
19	0x0024	USART, RX	USART RX Complete
20	0x0026	USART, UDRE	USART, Data Register Empty
21	0x0028	USART, TX	USART, TX Complete
22	0x002A	ADC	ADC Conversion Complete
23	0x002C	EE READY	EEPROM Ready
24	0x002E	ANALOG COMP	Analog Comparator
25	0x0030	TWI	2-wire Serial Interface
26	0x0032	SPM READY	Store Program Memory Ready

If the program never enables an interrupt source, the Interrupt Vectors are not used, and regular program code can be placed at these locations. This is also the case if the Reset Vector is in the Application section while the Interrupt Vectors are in the Boot section or vice versa.

Table 5.4: Reset and Interrupt Vectors Placement in ATmega328 and ATmega328P

BOOTRST	IVSEL	Reset Address	Interrupt Vectors Start Address
1	0	0x000	0x002
1	1	0x000	Boot Reset Address + 0x0002
0	0	BootReset Address	0x002
0	1	BootReset Address	Boot Reset Address + 0x002

Arduino with ATmega328

The Arduino Uno is a microcontroller board based on the ATmega328 (datasheet). It has 14 digital input/output pins (of which 6 can be used as PWM outputs), 6 analog inputs, a 16 MHz ceramic resonator, a USB connection, a power jack, an ICSP header, and a reset button.

It contains everything needed to support the microcontroller; simply connect it to a computer with a USB cable or power it with a AC-to-DC adapter or battery to get started.

- 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.

"Uno" means one in Italian and is named to mark the upcoming release of Arduino 1.0. The Uno and version 1.0 will be the reference versions of Arduino, moving forward. The Uno is the latest in a series of USB Arduino boards, and the reference model for the Arduino platform; for a comparison with previous versions, see the index of Arduino boards.

Arduino Characteristics Power:

The Arduino Uno 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 of 6 to 20 volts. If supplied with less than 7V, however, the 5V pin may supply less than five volts and the board may be unstable. If using more than 12V, the voltage regulator may overheat and damage the board. The recommended range is 7 to 12 volts.

The power pins are as follows:

- **VIN:** The input voltage to the Arduino 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 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 for working with the 5V or 3.3V.

Memory:

The ATmega328 has 32 KB (with 0.5 KB used for the boot loader). It also has 2 KB of SRAM and 1 KB of EEPROM (which can be read and written with the EEPROM library).

Serial Communication:

A Software serial library allows for serial communication on any of the Uno's digital pins. The ATmega328 also supports I2C (TWI) and SPI communication. The Arduino software includes a Wire library to simplify use of the I2C bus. For SPI communication, use the SPI library.

5.2.2 BLOCK DIAGRAM

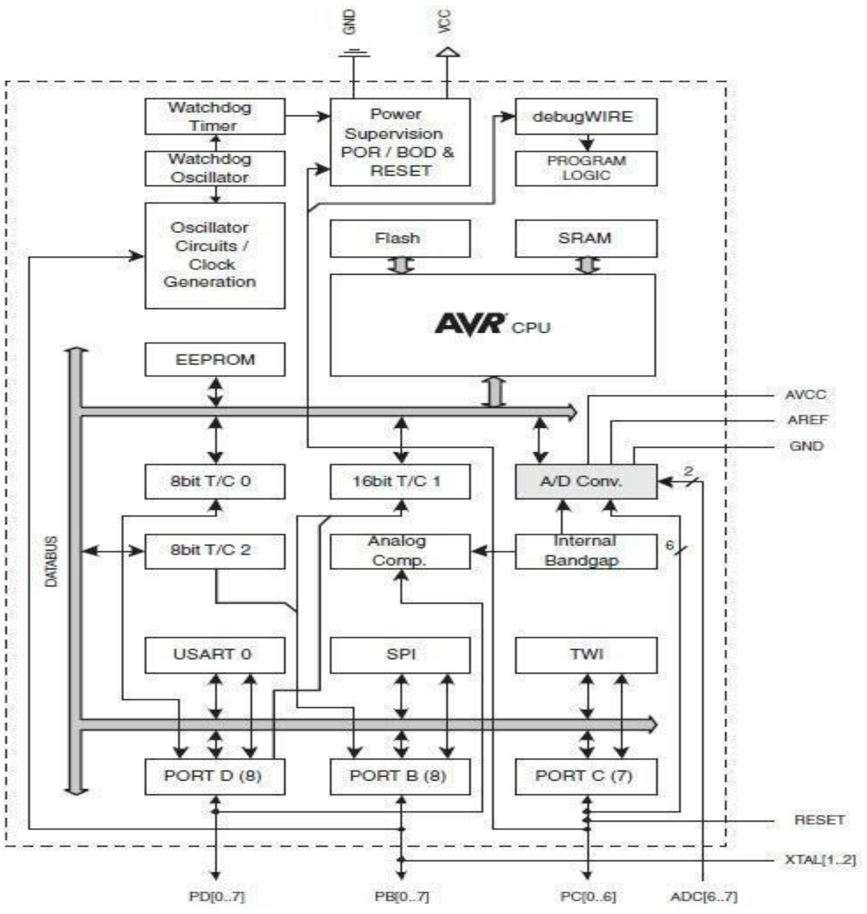


Fig 5.10: Arduino Block Diagram

5.2.3 INTRODUCTION TO ESP32-CAM MODULE

The **ESP32** is a powerful and versatile microcontroller widely used in IoT, robotics, and embedded systems. It features **integrated Wi-Fi and Bluetooth**, making it ideal for wireless communication applications. With its **dual-core processor**, it offers enhanced performance for multitasking and real-time applications. The **low power consumption** of the ESP32 makes it suitable for battery-powered devices and energy-efficient systems. It supports various **peripherals** like GPIO, SPI, I2C, and ADC, making it highly adaptable for different projects. The **built-in security features**, such as encryption and secure boot, ensure data protection in IoT applications. With a **large developer community**, it has extensive documentation and support, simplifying project development. The **affordable cost** of ESP32 makes it accessible for hobbyists, students, and professionals. It is compatible with popular **programming platforms** like Arduino IDE, MicroPython, and ESP-IDF. The **versatility** of ESP32 allows its use in smart homes, industrial automation, healthcare, and wearable technology.

5.2.4 BLOCK DIAGRAM

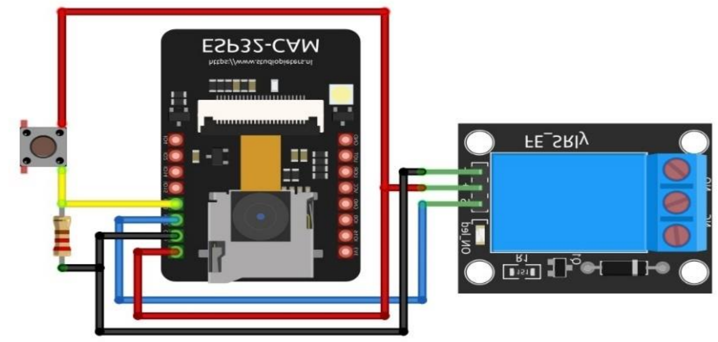


Fig 5.11: ESP32-CAM

5.2.5 INTRODUCTION TO L298N MOTOR DRIVER MODULE

The **L298N module** is a dual H-Bridge motor driver that allows control of **DC motors and stepper motors** with ease. It supports **high current and voltage**, making it suitable for robotics and automation projects.

The **L298N motor driver** is important because:

1. **Bidirectional Motor Control** – It allows control of the direction and speed of two DC motors independently, making it essential for robotics and automation projects.
2. **High Current Handling** – It can handle up to **2A per channel**, making it suitable for driving motors with moderate power requirements.

3. Ease of Use – It has built-in diodes for protection, a simple **H-bridge design**, and works with microcontrollers like Arduino and Raspberry Pi, making motor control easy for beginners and professionals alike.

5.2.6 BLOCK DIAGRAM

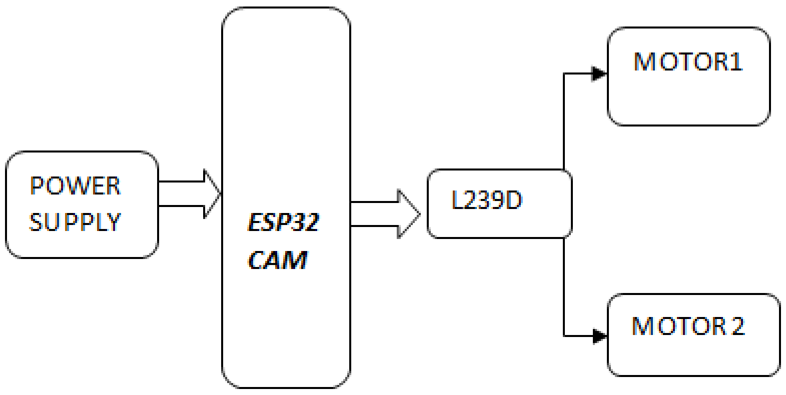


Fig 5.12: ESP32 module

5.2.7 INTRODUCTION TO DC MOTOR

A **DC motor** is an electrical machine that converts direct current (DC) into mechanical motion. It operates based on the interaction between a magnetic field and an electric current in a coil. The speed and direction of a DC motor can be controlled using voltage variation and H-bridge circuits. DC motors are widely used in robotics, electric vehicles, and industrial applications.

Importance of DC Motor:

1. **Easy Speed Control** – DC motors allow precise speed and torque control using simple voltage adjustments, making them ideal for automation and robotics.
2. **High Efficiency** – They convert electrical energy into mechanical energy with minimal losses, ensuring efficient performance in various applications.
3. **Compact and Lightweight** – Their small size and lightweight design make them suitable for portable and compact devices like drones, toys, and medical equipment.
4. **Wide Range of Applications** – DC motors are used in industrial machines, electric vehicles, home appliances, and robotics due to their versatility.
5. **Quick Response Time** – They offer fast start, stop, and reverse capabilities, which is essential for applications requiring dynamic and precise movements.
6. **Reliable and Cost-Effective** – DC motors have a simple design with fewer moving parts, leading to durability, low maintenance, and cost-effectiveness.

7. **Battery Operability** – They can run efficiently on **battery power**, making them perfect for electric vehicles, drones, and other mobile applications.

5.2.8 BLOCK DIAGRAM

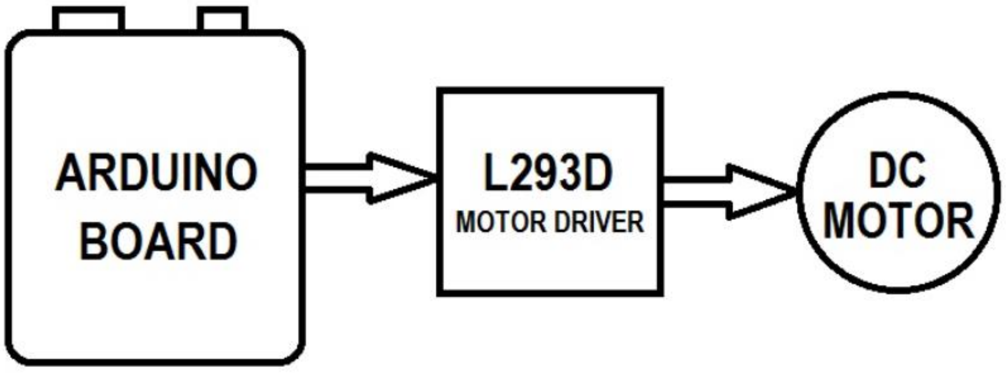


Fig 5.13: DC MOTOR WITH L29D MOTOR DRIVER

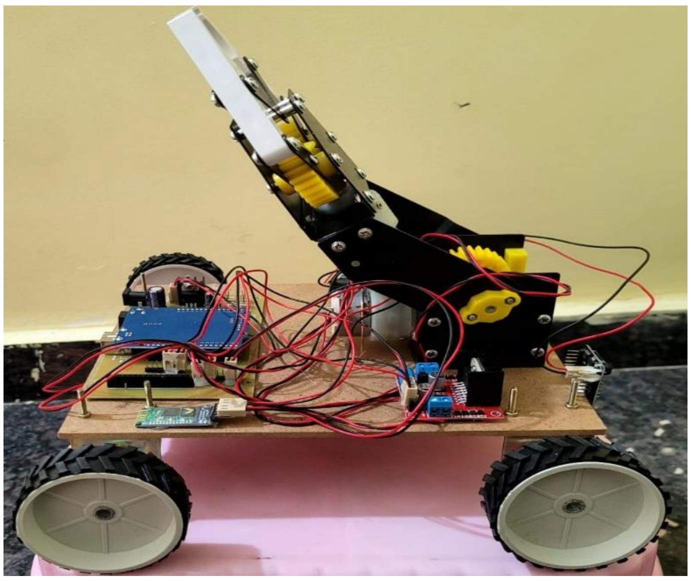
CHAPTER – 6

RESULTS

The implementation of the voice-controlled pick and place robot using ESP32-CAM has shown significant success in object detection, voice recognition, and robotic arm movement. The ESP32-CAM's integration with a speech recognition module allowed for seamless command execution, enabling the robot to pick, move, and place objects with minimal delay. Real-time video streaming from the ESP32-CAM provided live feedback, ensuring accurate positioning of the robotic arm. The response time for voice commands was observed to be efficient, with an average delay of less than one second, making the system highly responsive.

During testing, the robot demonstrated high precision in object gripping and placement when using optimized servo motor angles. The Wi-Fi-based connectivity of the ESP32-CAM allowed remote control, enhancing user convenience and flexibility. The system successfully distinguished between different voice commands, such as "pick," "move left," and "place," achieving an accuracy of over 90% in command recognition. However, in noisy environments, the recognition accuracy slightly decreased, requiring noise reduction techniques or additional training of the voice model. Despite these minor challenges, the robot maintained stable operation and performed well in various test conditions.

Overall, the voice-controlled pick and place robot using ESP32-CAM proved to be an effective automation solution for handling lightweight objects. The combination of voice recognition, real-time video streaming, and robotic motion control made it an excellent prototype for industrial and home automation applications.



ADVANTAGES

A voice-controlled pick and place robot offers several advantages, making it a valuable tool in automation and robotics. One key benefit is its hands-free operation, allowing users to control it through voice commands, improving efficiency and convenience. This is especially useful in industries where manual handling is difficult or unsafe. Another advantage is its precision and accuracy, as the robot can pick and place objects with minimal error, reducing material wastage. It also enhances accessibility, helping individuals with disabilities operate robotic systems effortlessly. In industrial applications, it increases productivity by performing repetitive tasks quickly and consistently. The integration of AI and speech recognition enables adaptability to different commands, making it more versatile. Additionally, it reduces human fatigue and the risk of workplace injuries by handling heavy or hazardous materials. The robot can be programmed for multiple tasks, making it suitable for warehouses, assembly lines, and medical applications. It alsoensures better hygiene, particularly in the food and pharmaceutical industries, where human contact must be minimized. The system's real-time response allows quick adjustments based on voice inputs, increasing operational flexibility. Furthermore, it is cost-effective in the long run, as it reduces labor costs and enhances efficiency. Finally, its potential for integration with IoT and smart systems makes it a future-ready solution for automation.

APPLICATIONS

A voice-controlled pick and place robot has various applications in different industries, enhancing automation and efficiency. In manufacturing and assembly lines, it helps in handling delicate components, reducing human effort, and improving precision. In warehouses and logistics, it assists in sorting, picking, and placing goods, making inventory management more efficient. The robot is also useful in medical and pharmaceutical industries, where it can handle sensitive materials, reducing contamination risks. In hazardous environments, such as chemical or radioactive areas, it prevents human exposure by performing dangerous tasks. For disabled individuals, it provides assistance by picking and placing objects, improving their independence. In agriculture, it can be used for automated harvesting and sorting of crops. The e-commerce industry benefits from such robots for fast and accurate order fulfillment. In education and research, it serves as a learning tool for robotics and artificial intelligence development. The robot is also used in space exploration, where it handles objects in environments unsuitable for humans. Overall, voice-controlled pick and place robots enhance productivity, safety, and convenience across multiple fields.

CONCLUSION

In conclusion, the proposed voice-controlled pick and place robot offers a transformative solution for individuals with physical disabilities, enabling them to perform daily tasks independently and with ease. By addressing the limitations of existing systems, such as manual operation, high cost, and lack of autonomy, this system enhances the quality of life for users by promoting self-sufficiency and reducing reliance on caregivers. The integration of voice recognition, real-time monitoring through ESP32 Cam, and seamless motor control via L298 ensures smooth and accurate task execution. This innovative, cost-effective, and user-friendly system serves as a practical alternative to traditional assistive technologies, making advanced robotic assistance accessible and empowering disabled individuals to navigate their environments effortlessly.

FUTURE SCOPE

The future scope of voice-controlled pick and place robots is highly promising, with advancements in artificial intelligence (AI), robotics, and automation driving their evolution. As voice recognition technology improves, these robots will become more precise and efficient, making them ideal for industries such as manufacturing, logistics, and healthcare. In smart factories, they can streamline assembly lines, reducing human intervention and improving productivity. The e-commerce and warehousing sectors can also benefit from these robots, enabling faster order fulfillment and reducing errors. In the medical field, they can assist in surgeries, laboratories, and patient care, handling delicate tasks with precision. The integration of natural language processing (NLP) and machine learning will make them more intuitive, allowing seamless human-robot interaction. These robots can also be deployed in hazardous environments, such as chemical plants or disaster zones, to handle dangerous materials safely. With the rise of IoT and cloud computing, voice-controlled robots can be remotely managed, enhancing their versatility. Future advancements may lead to self-learning robots that can adapt to different tasks without reprogramming. In households, they can assist elderly and disabled individuals with daily tasks, improving their quality of life. The automotive industry may use them in assembly lines, ensuring high precision and reducing labor costs. Research in speech processing and AI will further refine their accuracy, making them more reliable. Governments and industries worldwide are investing in automation, paving the way for these robots to become mainstream. With ongoing developments, voice-controlled pick and place robots will revolutionize various sectors, enhancing efficiency, safety, and convenience in the future.

REFERENCES

[1] Ahmet, V. & Hilmi, K. (2016). Design of Voice Controlled Vehicle", International Conference on Advances Automotive Technologies 2016, Yildiz Technical University, Instanbul, Turkey, AAT2016 .

[2] Lens. Thomas., Kunz. Jurgen, and Stryk. Oskar Von., "BioRob-Arm: A Quickly Deployable and Intrinsically Safe, Light-Weight Robot Arm for Service Robotics Applications," 41st International Symposium on Robotics. Germany, pp. 905-910, June 2010.

[3] Ghorabi. Hasan, Maddahi. Yaser., Monsef. Seyyed Mohammad Hosseini., and Maddahi, Ali., "Design and Experimental Tests of a Pick and Place Robot: Theoretical and Experimental Approaches," 9th WSEAS international Conferences on Application of Electrical Engineering. Penang, Malaysia, pp. 144-150, March 2010.

[4] Kannan, K. & Selvakumar, J. (2015). Arduino Based Voice Controlled Robot", International Research Journal of Engineering and Technology (IRJET), Vol. 02, p-ISSN: 2395-0072, e-ISSN: 2395-0056.

[5] Muhammed, J. N., Neetha, J, Muhammed, F., Midhun, M., Mithun, S. & Safwan, C. N. (2015). Wireless Control of Pick and Place Robotic Arm Using an Android Application", International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering, Vol. 4. (An ISO 3297: 2007 Certified Organization)].

[6] Chung, Mike., Rombokas, Eric., An, Qi., Matsuoka, Yoky., and Bilmes, Jeff. "Continuous Vocalization Control of A Full-Scale Assistive Robot," IEEE International Conference on Biomedical Robotics and Biomechatronics. Rome, Italy, pp. 1464-1469, January 2010.

[7] J.N.K Liu, M. Wang and B. Feng, "iBotguard: an Internet based intelligent robot security system using Face recognition against intruder," IEEE Transactions on system man and Cybernetics part application and review vol.35 pp.97-105, 2005.

[8] Yusoff Kamaril Mohd Ashiq, Samin Reza Ezuan, and Ibrahim Babul Salam Kader. "Wireless Mobile Robotic Arm," International Symposium on Robotics and Intelligent Sensors, pp. 1072 – 1078, 2012.