A

MAJOR PROJECT REPORT

ON

AGROBOT FOR REAL TIME FARM MONITORING

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

BACHELOR OF TECHNOLOGY

IN

ELECTRONICS AND COMMUNICATION ENGINEERING

SUBMITTED BY

MOHAMMED NOORUDDIN	**218R1A04A4**
MYAKALA VAMSHI	**218R1A04A5**
NALLAN CHAKRAVARTHULA PHANINDRA	**218R1A04A6**
NAMANI KIRAN	**218R1A04A7**

Under the Esteemed Guidance of

Mrs. D. LATHA
Assistant professor, ECE

DEPARTMENT OF ELECTRONICS &

COMMUNICATION ENGINEERING

CMR ENGINEERING COLLEGE

(UGC AUTONOMUS)

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

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

2024-2025

CMR ENGINEERING COLLEGE

(UGC AUTONOMUS)

(Approved by AICTE, Affiliated to JNTU Hyderabad, Accredited by NBA)
Kandlakoya(V), Medchal Road, Telangana – 501401

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

CERTIFICATE

This is to certify that the major-project work entitled **"AGROBOT FOR REAL TIME FARM MONITORING"** is being submitted by **M. NOORUDDIN** bearing Roll No **218R1A04A4, M. VAMSHI** bearing Roll No **218R1A04A5, NC. PHANINDRA** bearing Roll No **218R1A04A6, N. KIRAN** bearing Roll No **218R1A04A7** 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
Mrs. D. LATHA
Assistant Professor, ECE

HEAD OF THE DEPARTMENT
Dr. SUMAN MISHRA

EXTERNAL EXAMINER

ACKNOWLEDGEMENT

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 **Mrs. D. LATHA**, Assistant Professor of ECE for guidance and encouragement in carrying out this project work.

DECLARATION

We hereby declare that the major project entitled "**AGROBOT FOR REAL TIME FARM MONITORING**" 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 FROM JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY, HYDERABAD.**

M.NOORUDDIN	**(218R1A04A4)**
M.VAMSHI	**(218R1A04A5)**
NC.PHANINDRA	**(218R1A04A6)**
N.KIRAN	**(218R1A04A7)**

ABSTRACT

It aims on the design, development and the manufacture of the robot which can put the seeds, Detects if there are any insects on plants, Moisture the land and cutting the waste plants. These entire systems of robot works with battery. In India almost about 70% of humans are relying on agriculture. So the agronomics procedure in India ought to be progressed to reduce the efforts of farmers. Various operations are performed in the agronomics acreage like seeding, weeding, waste plant cutting, ploughing, etc.

Exceptionally primary operations are seeding, ploughing and plant cutting. But the present techniques of seeding, ploughing & plant cutting are problematic. The equipment acclimated for seed dispersing are actually difficult and inappropriate to handle. So there is a need of advance techniques and equipment which will reduce the man power. The mechanism can be progressed for sowing seeds in farm with specific separation between seed is balanced. In this paper robot administration is provided by utilizing Software programming.

By utilizing that proper administration is specified to the robot. The acreage is not the undeviating line and smooth. If any obstacle is occurred like stone, electric light pole, trees, etc. the robot automatically get stop. Agriculture is our wisest pursuit because it will in the end contribute most to real wealth, good morals, and happiness. Agriculture needs a lot of manpower and hard work, so this Agrobot is invented to ease the processes of farming.

To overcome these obstacles there is a single solution “Agrobot”. Agrobot is an agricultural robot that does multiple processes in farming. Agrobot is an agricultural robot that does multiple processes in farm This robot can do work as a total of seven processes, those are seeding, irrigating, soil monitoring, crop monitoring,. This project is incorporated many like Bluetooth module, sensors, and actuators used in IoT. So, this Agrobot reduces the human intervention, reduces wastage of time and it is also cost- efficient.

CONTENTS

LIST OF FIGURES

CHAPTER 1
INTRODUCTION

1.1 OVERVIEW OF THE PROJECT

With the continuous growth of the world population, the demand for food and cultivated land increases continuously. The prediction of the Food and Agriculture Organization of United Nations (FAO) indicates the presence of constant growth of the population with the rate of 79 million people per year, increasing food demand Since the cultivated land resources are limited, and acquiring new ones is correlated with degradation of ecosystems, reduction of forests, climate changes, and risks of new pandemic breakouts, as well as degradation of soil properties due to inappropriate cultivation and treatment, there is an urgent need to improve soil treatment, to increase yield in a sustainable manner. The best approach that will enable farming to become more efficient in a sustainable way and reduce the production costs at the same time, is to provide an efficient supply of nutrients and water Standard and classical methods of soil analysis usually involve taking 1–20 samples per 5 hectares from around 30 cm depth. They are usually mixed and analyzed for an average value of nutrients. A laboratory analysis then takes 10–15 days to obtain the results. The most common soil sampling methods used are hand sampling, hydraulic probes, electric probes, and auger probes . Hand sampling is easy to use and economic, but it is time-consuming, labor-intensive, and could be inconsistent with the sampling depths. Hydraulic probes are fast and have a consistent depth, but are composed of numerous components (engine, hydraulics tank, pump, and lines), vendor locked, and pricy in the range from USD 4000 to 8000 on average. Electric probes demand low mainte- nance, with no fuel costs, and are more suited for dusty conditions.

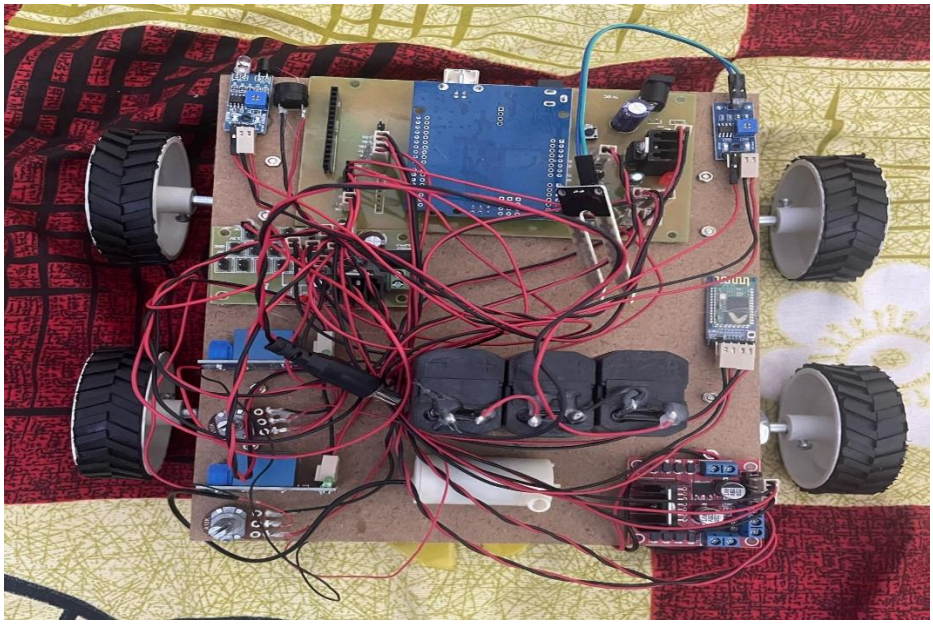


FIG 1.1 OVER VIEW OF PROJECT

Auger probes are the most durable and easy to set up and use. Their main drawback is the cross-contamination due to poor probe cleanout capabilities and they have difficulties with verifying core depth as well with their usage in sandy soils. All the stated sampling methods end up with time-consuming laboratory analysis, which does not provide on- time information to the farmers. Furthermore, due to sample averaging, precise information about nutrients at the exact location is lost. Considering nitrogen, with its large spatial and temporal variations, under and over-fertilization is inevitable. This can lead to a decrease in yield on one hand, and significant pollution of the ecosystems on the other

1.2 OBJECTIVE OF THE PROJECT

Precision agriculture, smart farming, and automated agricultural technology have emerged as promising methodologies for increasing crop productivity without sacrificing produced quality. The emergence of various robotics technologies has Z facilitated the application of these techniques in agricultural processes. In the stated article, the authors provide an overview of the current state of the art in agricultural robots. Classification of the five major operations in open arable farming is presented in the study by, where one of them is soil analysis. An implication of soil agrochemical analysis and its accuracy in precision agriculture is given in. The authors suggest that rapid, less labor-intensive, economical, and at least equally accurate methods have to be developed for precision agriculture. They emphasize high soil analysis costs with traditional methods. A

suggestion for scales of 10 m or less in precise agriculture is suggested in, due to the high spatial variability of the nutrients in the soil. Early work in the automation of soil sampling is presented in. The system collects, packs, and marks samples with georeferencing. Sowing and fertilization as phases in the agricultural production process should be completed after soil analysis based on appropriate methods providing information on nutrient availability for the growth and development of the particular plant. Until recently, farmers applied uniform fertilization per plot, which is not optimal from an economic perspective, and unsustainable from an ecologic, environmental, and ecosystem point of view. Plants acquire nutrients in amounts that are needed for normal growth, while any additional artificially applied nutrients by fertilization largely evaporate, creating greenhouse gases, or end up in surface and underground waters, which boosts algae growth in rivers and lakes. The robot aims to automate the collection of essential environmental data such as soil moisture, temperature, and humidity, which are critical for effective crop management. By integrating sensors and wireless communication, the system enables farmers to receive accurate, real-time information without the need for constant physical presence in the field. This project seeks to reduce manual labor, improve resource efficiency—especially water usage—and enhance decision-making through data-driven insights. Ultimately, the goal is to create a cost-effective and reliable solution that contributes to increased agricultural productivity and sustainability. By automating the data collection process, the system aims to reduce the dependency on manual labor and minimize human error in monitoring tasks. Additionally, the robot is integrated with a wireless communication module to transmit real-time data to a central monitoring system or mobile device, enabling farmers to access important information remotely. This facilitates timely decisions related to irrigation, fertilization, and crop protection. The project also aims to promote water conservation by ensuring that irrigation is carried out only when necessary, based on actual soil moisture readings. Overall, the goal is to develop a reliable, low-cost, and efficient robotic system that supports precision agriculture, improves crop health monitoring, and contributes to increased productivity and resource optimization on farms.

1.3 EMBEDED 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 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. The embedded system constantly monitors environmental and soil conditions, evaluates sensor inputs, and responds accordingly by activating motors for movement, adjusting irrigation schedules, or transmitting data wirelessly to a mobile app or control station. It plays a crucial role in ensuring the robot's autonomy, allowing it to function in remote fields without constant human supervision. Furthermore, embedded systems are designed to operate in harsh outdoor conditions, offering durability and reliability in diverse climates. They are typically powered by low-energy microcontrollers, which makes them energy-efficient and suitable for battery-operated or solar-powered agricultural robots. Overall, the embedded system is the backbone of the agriculture robot, enabling automation, real-time monitoring, and smart farming capabilities that lead to increased productivity, reduced labor, and sustainable resource usage.

CHAPTER-2
LITERATURESURVEY

2.1 EXISTING SYSTEM

Bruinsma, J. (Ed.) Chapter 1 Introduction and overview. In World Agriculture: Towards 2015/2030, An FAO Perspective, 1st ed.; Earthscan Publications Ltd.: London, UK, 2003; pp. 1–28.

The projections were carried out in considerable detail, covering about 140 countries and 32 crop and livestock commodities (see Annex 1). For nearly all the developing countries, the main factors contributing to the growth of agricultural production were identified and analysed separately. Sources of productivity growth, such as higher crop yields and livestock carcass weights, were distinguished from other growth resources, such as the area of cultivated land and the sizes of livestock herds. Special attention was given to land, which was broken down into five classes for rainfed agriculture and a sixth for irrigated agriculture. This level of detail proved both necessary and advantageous in identifying the main issues likely to emerge for world agriculture over the next 30 years. Specifically, it helped to spot local production and resource constraints, to gauge country-specific requirements for food imports and to assess progress and failure in the fight against hunger and undernourishment. The high degree of detail was also necessary for integrating the expertise of FAO specialists from various disciplines, as the analysis drew heavily on the judgement of in-house experts. Owing to space and other constraints, the results are, however, mainly presented at the aggregate regional and sectoral levels, which can mask diverging developments between individual countries and commodities. Likewise, space considerations militated against the inclusion of references to the numerous sources drawn upon in this report. References have therefore been limited to statistical sources and the sources of figures, tables and maps. These are given on p. 96. A complete list of references is provided in the main technical report. Other important feature of this report is that its approach is “positive” rather than “normative”. This means that its assumptions and projections reflect the most likely future but not necessarily the most desirable one. For example, the report finds that the goal of the 1996 World Food Summit — to halve the number of chronically undernourished people by no later than 2015 — is unlikely to be

accomplished, even though this would be highly desirable. Similarly, the report finds that agriculture will probably continue to expand into wetlands and rainforests, even though this is undoubtedly undesirable. In general, the projections presented are therefore not goals of an FAO strategy but rather a basis for action, to cope both with existing problems that are likely to persist and with new ones that may emerge. It should also be stressed that these projections are certainly not trend extrapolations. Instead, they incorporate a multitude of assumptions about the future and often represent significant deviations from past trends.

Pennock, D.; Yates, T.; Braidek, J. Chapter 1 Soil Sampling design. In Soil Sampling and Methods of Analysis, 2nd ed.; Carter, M.R., Gregorich, E.G., Eds.; CRC Press, Taylor & Francis Group: Boca Raton, FL, USA, 2006; pp. 25–38.

Sampling involves the selection from the total population of a subset of individuals upon which measurements will be made; the measurements made on this subset (or sample) will then be used to estimate the properties (or parameters) of the total population. Sampling is inherent to any field research program in soil science because the measurement of the total population is impossible for any realistic study. For example, even a single 10 ha field contains about 100,000 1 m2 soil pits or 1 107 10 cm2 cores, and sampling of the entire population would be more of an unnatural obsession than a scientific objective. Sampling design involves the selection of the most efficient method for choosing the samples that will be used to estimate the properties of the population. The definition of the population to be sampled is central to the initial formulation of the research study (Eberhardt and Thomas 1991; Pennock 2004). The sampling design defines how specific elements will be selected from the population, and these sampled elements form the sample population. There are many highly detailed guides to specific sampling designs and the statistical approaches appropriate for each design. The goal of this chapter is to present the issues that should be considered when selecting an appropriate sampling design. In the final section, specific design issues associated with particular research designs are covered. Suggested readings are given in each section for more in-depth study on each topic.buses which take a specific route on a daily basis for the better management of time and safety. Also to implement this system in public transport vehicles, various companies and many more. There is a requirement of some type of system to determine where each object was at any given time and for how long it traveled. GSM and GPS built tracking system will offer effective, real-time vehicle location, and report the location of the vehicle.

Martínez-Dalmau, J.; Berbel, J.; Ordóñez-Fernández, R. Nitrogen Fertilization. A Review of the Risks Associated with the Inefficiency of Its Use and Policy Responses. Sustainability 2021,13, 5625

This article reviews soil sampling and soil chemical analysis, discussing their implications from, and applications in, precision agriculture. The variability of a number of agriculturally important soil chemical properties was investigated and the 'nugget' variance or effect discussed in terms of its importance in determining the proportion of not only short-range spatial variation, but also sampling and measurement error. Comments were then made on the accuracy of laboratory methods. Analytical variances were compared with world-average and estimated nugget variances for a field in New South Wales, the comparison showing that analytical precision needs to be maintained or improved when developing or adapting analytical methods for precision agriculture. A simple cost-analysis showed that soil chemical analytical costs are much too large for economic use in precision agriculture, costs in Australia being higher than in the United States. The conclusion this paper draws is that, for large-scale implementation of precision agriculture, the development of field-deployed, 'on-the-go' proximal soil sensing systems and scanners is tremendously important. These sensing systems or scanners should aim to overcome current problems of high cost, labour, time and to some extent, imprecision of soil sampling and analysis to more efficiently and accurately represent the spatial variability of the measured properties.

2.2 PROPOSED SYSTEMS

In the traditional farming method, there are totally 6 processes, those are plowing, sowing, land leveling, irrigating, fertilizing, and harvesting. Usually, farmers use huge machinery for doing these processes They use each machine for each process, this machinery is so costly, it requires a lot of human intervention and it takes so much time. To overcome these obstacles there is a single solution "Agrobot". Agrobot is an agricultural robot that does multiple processes in farming. This multipurpose agricultural robot is build using Arduino UNO microcontroller. To sense the parameters of the soil sensors are used in this project and as actuators servo motors and DC motors are incorporated. After the incorporation of these sensors and actuators, the Arduino coding is uploaded to the Arduino UNO using Arduino IDE software. This Agrobot can be controlled by a mobile application. This

mobile application is developed using MIT app inventor. This can be so easy like a kid playing a remote controlled car. The processes included in this robot are:

- Object detection
- Monitering the crops
- Irrigation
- Soil monitoring
- Grass cutting

2.3 INTRODUCTION TO EMBEDDED SYSTEMS

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. Embedded systems in many cases must be optimized for life-cycle and business-driven factors rather than for maximum computing throughput. 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.

If we look around us, today we see numerous appliances which we use daily, be it our refrigerator, the microwave oven, cars, PDAs etc. Most appliances today are powered by something beneath the sheath that makes them do what they do. These are tiny microprocessors, which respond to various keystrokes or inputs. These tiny microprocessors, working on basic assembly languages, are the heart of the appliances. We call them embedded systems. Of all the semiconductor industries, the embedded systems market place is the most conservative, and engineering decisions here usually lean towards established, low risk solutions. Welcome to the world of embedded systems, of computers that will not look like computers and won't function like anything we are familiar with.

2.4 WHY EMBEDDED?

Embedded systems are divided into autonomous, realtime, networked & mobile categories.

Autonomous systems

They function in standalone mode. Many embedded systems used for process control in manufacturing units& automobiles fall under this category.

Real-time embedded systems

These are required to carry out specific tasks in a specified amount of time. These systems are extensively used to carry out time critical tasks in process control.

Networked embedded systems

They monitor plant parameters such as temperature, pressure and humidity and send the data over the network to a centralized system for on line monitoring.

Mobile gadgets

Mobile gadgets need to store databases locally in their memory. These gadgets imbibe powerful computing & communication capabilities to perform realtime as well as nonrealtime tasks and handle multimedia applications. The embedded system is a combination of computer hardware, software, firmware and perhaps additional mechanical parts, designed to perform a specific function. A good example is an automatic washing machine or a microwave oven. Such a system is in direct contrast to a personal computer, which is not designed to do only a specific task. But an embedded system is designed to do a specific task with in a given timeframe, repeatedly, endlessly, with or without human interaction.

Hardware

Good software design in embedded systems stems from a good understanding of the hardware behind it. All embedded systems need a microprocessor, and the kinds of microprocessors used in them are quite varied. A list of some of the common microprocessors families are: ARM family, The Zilog Z8 family, Intel 8051/X86 family, Motorola 68K family and the power PC family. For processing of information and execution of programs, embedded system incorporates microprocessor or micro-controller. In an embedded system the microprocessor is a part of final product and is not available for reprogramming to the end user.

An embedded system also needs memory for two purposes, to store its program and to store its data. Unlike normal desktops in which data and programs are stored at the same

place, embedded systems store data and programs in different memories. This is simply because the embedded system does not have a hard drive and the program must be stored in memory even when the power is turned off. This type of memory is called ROM. Embedded applications commonly employ a special type of ROM that can be programmed or reprogrammed with the help of special devices.

2.4.1 OTHER COMMON PARTS FOUND ON MANY EMBEDDED SYSTEMS

- UART& RS232
- PLD
- ASIC's& FPGA's
- Watch dog timer etc.

2.5 DESIGN PROCESS

Embedded system design is a quantitative job. The pillars of the system design methodology are the separation between function and architecture, is an essential step from conception to implementation. In recent past, the search and industrial community has paid significant attention to the topic of hardware-software (HW/SW) 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 behaviors of embedded systems. Hardware can be simulated at different levels such as electrical circuits, logic gates, RTL e.t.c. using VHDL description. In some environments software development tools can be coupled with hardware simulators, while in others the software is executed on the simulated hardware. The later approach is feasible only for small parts of embedded systems. Design of an embedded system using Intel's 80C188EB chip is shown in the figure. Inorder to reduce complexity, the design process is divided in four major steps: specification, system synthesis, implementation synthesis and performance evaluation of the prototype.

2.5.1 SPECIFICATION

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

2.5.2 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.

2.5.3 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.

2.5.4 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.

2.5.5 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 & entertinment services using GPS, satellite. Home appliances are going the embedded way.

Here are some broad categories

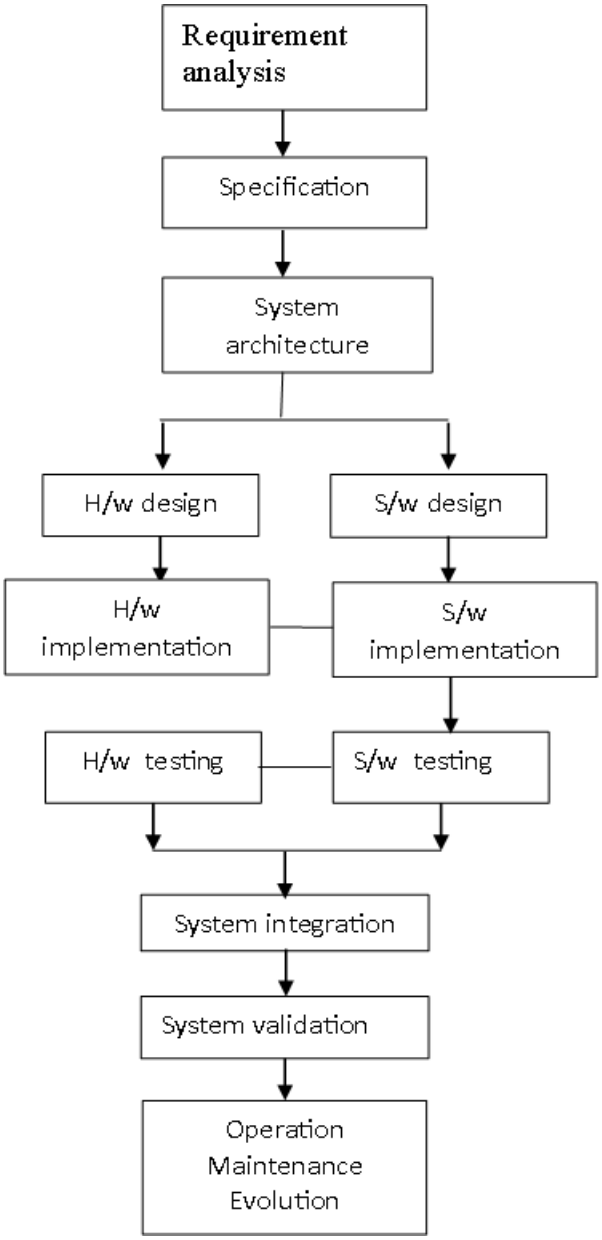


Fig 2.1: Embedded Development Life Cycle

- **Aerospace and defense 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.

CHAPTER-3
HARDWARE REQUIREMENTS

3.1 HARDWARE

Embedded system hardware

Embedded system hardware can be microprocessor- or microcontroller-based. In either case,an integrated circuit is at the heart of the product that is generally designed to carry out real- time computing. Microprocessors are visually indistinguishable from microcontrollers.However, the microprocessor only implements a central processing unit (CPU) and, thus requires the addition of other components such as memory chips. Conversely, microcontrollers are designed as self-contained systems.

Microcontrollers include not only a CPU, but also memory and peripherals such as flash memory, RAM or serial communication ports. Because microcontrollers tend to implementfull (if relatively low computer power) systems, they are frequently used on more complex tasks. For example, microcontrollers are used in the operations of vehicles, robots, medical devices and home appliances. At the higher end of microcontroller capability, the term System on a chip (SOC) is often used, although there's no exact delineation in terms of RAM, clock speed, power consumption and so on.

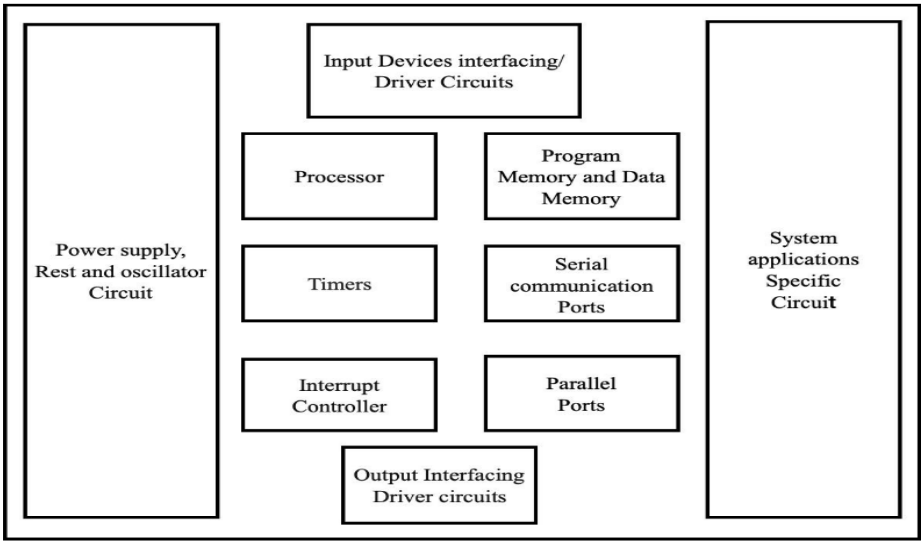


FIG:3.1 EMBEDDED SYSTEMS HARDWARE BLOCK DIAGRAM

Hardware for embedded systems is much less standardized than hardware for personal computers. Due to the huge variety of embedded system hardware, it is impossible to providea comprehensive overview of all types of hardware components. Nevertheless, we will try to provide a survey of some of the essential components which can be found in most systems.The choice of components for the WHO-recommended hand rub formulations takes into account cost constraints and microbicidal activity. The following two formulations are recommended for local production with a maximum of 50 litres per lot to ensure safety in production and storage

Markets and Markets, a business to business (B2B) research firm, predicts that the embeddedmarket will be worth $116.2 billion by 2025. Chip manufacturers for embedded systems include many well-known technology companies, such as Apple, IBM, Intel and Texas Instruments, as well as numerous other companies less familiar to those outside the field. The expected growth is partially due to the continued investment in artificial intelligence (AI), mobile computing and the need for chips designed for that high-level processing.

To be used efficiently, all computer software needs certain hardware components or other software resources to be present on a computer.[1] These prerequisites are known as (computer) system requirements and are often used as a guideline as opposed to an absolute rule. Most software defines two sets of system requirements: minimum and recommended. With increasing demand for higher processing power and resources in newer versions of software, system requirements tend to increase over time. Industry analysts suggest that thistrend plays a bigger part in driving upgrades to existing computer systems than technological advancements. A second meaning of the term of system requirements, is a generalisation ofthis first definition, giving the requirements to be met in the design of a system or sub- system Memory

All software, when run, resides in the random access memory (RAM) of a computer.Memory requirements are defined after considering demands of the application, operating system, supporting software and files, and other running processes. Optimal performance ofother unrelated software running on a multi-tasking computer system is also considered when defining this requirement.

Secondary storage

Data storage device requirements vary, depending on the size of software installation, temporary files created and maintained while installing or running the software, and possibleuse of swap space (if RAM is insufficient).

Embedded systems, especially those in high-performance applications, may require cooling mechanisms like heat sinks, fans, or even advanced cooling technologies to ensure the system remains within operating temperature limits.

Embedded systems require accurate timing for many applications, such as controlling real- time processes. These are often provided by crystal oscillators or Real-Time Clocks **(**RTC**)**. Sensors detect changes in the physical environment and convert them into data that the embedded system can process. Examples include temperature sensors, accelerometers, gyroscopes, light sensors, and pressure sensors.

Actuators are components that perform actions based on the system's processing. These could be motors, LEDs, relays, or servos used to control devices or respond to inputs. Since embedded systems are often deployed in remote or mobile environments, efficient and compact power management is critical. Power supply components can include batteries, DC-DC converters, and power regulation circuitry to ensure stable and reliable power.

For communication, the embedded hardware may include wireless modules such as Bluetooth (HC-05), Wi-Fi (ESP8266/ESP32), or LoRa for long-range data transmission, enabling remote monitoring and control. The entire system is mounted on a chassis, often made of durable plastic or metal, which holds all the components securely in place.

Additional elements such as LED indicators, LCD displays, and buzzer modules can be added for feedback and alerts. All these hardware components are carefully integrated to make the embedded system efficient, compact, and reliable for use in agricultural environments

Display adapter

Software requiring a better than average computer graphics display, like graphics editors and high-end games, often define high-end display adapters in the system requirements.

Peripherals

Some software applications need to make extensive and/or special use of some peripherals, demanding the higher performance or functionality of such peripherals. Such peripherals include CD-ROM drives, keyboards, pointing devices, network devices, etc.

Basic Structure of an Embedded System

The following illustration shows the basic structure of an embedded system

- **Sensor** − It measures the physical quantity and converts it to an electrical signal which can be readby an observer or by any electronic instrument like an A2D converter. A sensor stores the measuredquantity to the memory.
- **A-D Converter** − An analog-to-digital converter converts the analog signal sent by the sensor intoa digital signal.

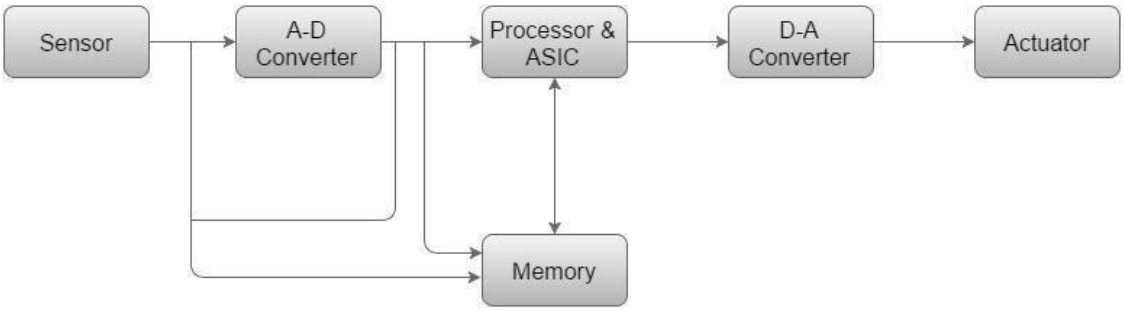


FIG:3.2 BASIC EMBEDDED STRUCTURE

- **Processor & ASICs** − Processors process the data to measure the output and store it to the memory.
- **D-A Converter** − A digital-to-analog converter converts the digital data fed by the processor to analog data
- **Actuator** − An actuator compares the output given by the D-A Converter to the actual (expected) output stored in it and stores the approved output.

In this loop, information about the physical environment is made available through sensors. Typically, sensors generate continuous sequences of analog values. In this book, we will restrict ourselves to information processing where digital computers process

discrete sequences of values. Appropriate conversions are performed by two kinds of circuits: sample-and-hold-circuits and analog-to-digital (A/D) converters. After such conversion, information can be processed digitally.Generated results can be displayed and also be used to control the physical environment through actuators. Since most actuators are analog actuators, conversion from digital to analog signals is also needed. This model is obviously appropriate for control applications. For other applications, it can be employed as a first order approximation. In the following, we will describe essential hardware components of cyber-physical systems following the loop structure

They aren't a lot different to the requirements for working with non-embedded systems. A lot depends onthe purpose of the embedded system. You need to understand:

- Requirement set
- Environmental context
- Regulator requirements
- Interface specifications, including choice of hardware, and how to drive that hardware
- Criticality of what you are building, including hazards, and any defined mitigation tothose hazards. For instance, is there a safe state to fail to if anything goes wrong.
- Real time constraints, such as cycle times, and time allowable for response. This shouldalso include hysteresis, response of mechanical components, and backlash.
- Real time response from the combination of code, operating system, and hardware youare working with.
- Architecture, including any need for redundancy, diversity, fail safety, voting systems,comparators.
- Platform limitations. Cross compilers, linkers, auto-code generators, etc.
- Choice of operating environment, such as bare metal minimal kernel, real time operatingsystem, or regular operating system.
- Whether you can rely on your tools. In particular, you need to understand how your toolscan fail, and how you'd know if something went wrong.

- Communications protocols. Synchronous, and asynchronous communications. Errorchecking. Error correcting.
- Timing issues, clock rates, reentrancy. Anything that might stop you meeting your realtime response targets, or impact the firing of timers

3.2 Design of grass cutter

3.2.1 Cutting Mechanism:

Rotary Blades: A system with rotating blades (similar to a lawnmower) can be used to cut grass at a consistent height. The blades can be powered by an electric motor or a small internal combustion engine.

Reel Blades:

A more advanced option is a reel blade system, which provides a clean and scissor-like cut for grass, helping with healthy growth.

3.2.2 Cutting Height Adjustment:

The system should allow automatic adjustment of cutting height depending on the type of grass and current growth conditions. This can be achieved via motors that control the height of the blade deck.

Grass Collection:

Grass Bag or Hopper: After cutting, the grass can be either collected into a hopper or left in place as mulch, depending on the robot's design and farm requirements.

Mulching Functionality: This can help return nutrients to the soil if mulching is preferred over bagging.

Fig 3.3 GRASS CUTTER

3.3 Design of the Irrigation system

Irrigation is the agricultural process of applying controlled amounts of water to land to assist in the production of crops, as well as to grow landscape plants and lawns, where it may be known as watering. Watering the field is done using drip irrigation. Drip irrigation uses the pipelines (with appropriate openings) which are spread across the field and water is fed to the crops according to the requirement. The water is fed in the pipelines by a water pump inside the water tank, which is present on the agrobot. In manual mode, the freedom to control this operation separately. The water is stored in the water tank, a water pump is dipped inside the water tank and the water pump is connected to the pipe which is left below the system to flow the water as shown A wire is attached in the water pump that has two terminals, and that is connected to the 4-channel relay. A 4-channel relay module is connected to the motor Arduino UNO as. There are four input pins, one GND pin and one VCC pin. 4 input pins of 4-channel relay module is connected to 2,3,4, and 5 digital pins of Arduino UNO. The GND pin of 4-channe relay module is connected to the GND pin of Arduino UNO. This system is controlled using a mobile application.

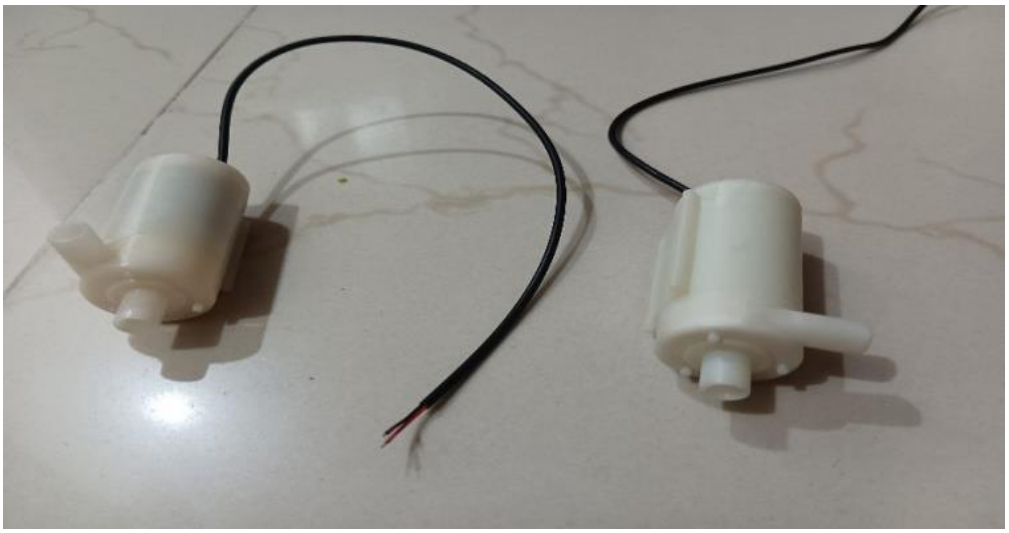


FIG 3.4 DC MOTOR PUMP

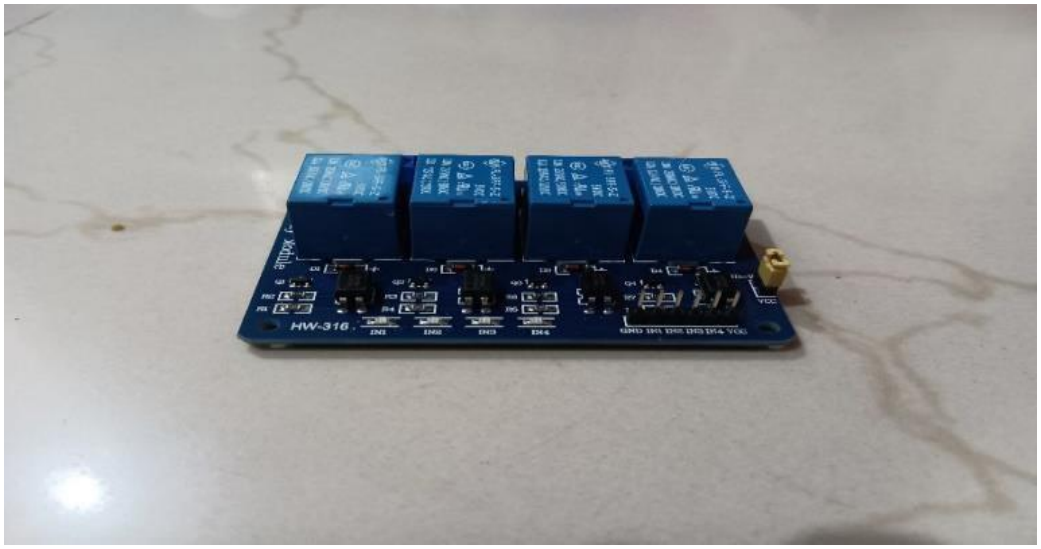


FIG 3.5 4-CHANNEL RELAY MODULE

3.4 Design of IR Sensor

Infrared (IR) sensors play a crucial role in agriculture robots by enabling them to interact effectively with their environment. These sensors are commonly used for obstacle detection, allowing the robot to navigate through fields while avoiding rocks, plants, fences, or other machinery. By emitting infrared light and analyzing the reflected signals, the robot can determine the distance to nearby objects. IR sensors are also used in line-following applications, helping robots move accurately between crop rows or along marked paths. In more advanced systems, near-infrared (NIR) sensors are utilized to monitor plant health by analyzing the light reflected from crops. Healthy plants reflect different wavelengths compared to stressed or diseased ones, allowing the robot to identify areas that may need watering, fertilization, or pest control. Additionally, thermal IR sensors can detect temperature variations in soil and crops, which is useful for monitoring plant stress, detecting pests, or predicting frost. Some IR sensors can even be used to estimate soil moisture levels indirectly, by measuring how much infrared light is absorbed or reflected by the soil. Overall, IR sensors enhance the efficiency, accuracy, and intelligence of agriculture robots, making them valuable tools for modern precision farm

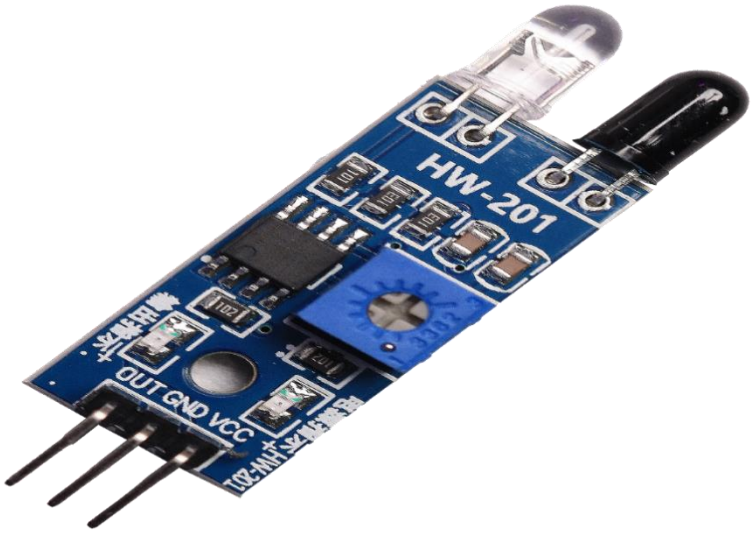


FIG 3.6 IR SENSOR

3.5 Design of the soil monitoring system

Soil Monitoring with IoT uses technology to empower farmers and producers to maximize yield, reduce disease and optimize resources. To monitor the parameters of soil 5 in 1 sensor is used . This sensor is connected to a servo motor that can be inserted in the soil while we want to monitor the soil. This 5 in 1 sensor monitors humidity, PH level, moisture, temperature and the sunlight level. This sensor takes care of the soil and environment parameters. The soil monitoring sensor is attached to a servo motor, the servo motor is attached to the servo_1 pin of Arduino UNO. This is operated using the mobile application developed by the MIT app inventor, a slider is attached in the MIT app inventor to control the servo motor. It sends a command to the microcontroller through the Bluetooth module. Then the servo motor activates the inserts the sensor in the field to monitor the parameters of soil .The sensor has three pins, those are GND, VCC and AO. GND pin of the soil sensor is connected to the GND pin of Arduino UNO, then the VCC pin of the sensor is connected to the 5V pin of the Arduino UNO and the AO pin of the sensor is connected to the A2 of the Arduino UNO. This system is operated using the mobile application. It will get the value from the sensor of the moisture level. Then the value will be displayed on the mobile application.

A soil moisture sensing system plays a crucial role in transforming traditional farming into a more intelligent, automated, and resource-efficient process. This system not only monitors the amount of water in the soil but also helps farmers make timely and accurate irrigation decisions. By continuously measuring the soil's moisture content, it ensures that plants receive optimal watering—neither too much, which could lead to root rot and nutrient leaching, nor too little, which could cause drought stress and poor crop development. In many agricultural robots or smart irrigation setups, the soil moisture sensor is embedded as part of a larger monitoring and control system. The sensor collects data from various points in the soil, which is then sent to a microcontroller. Based on predefined threshold values, the microcontroller can control devices like water pumps, sprinklers, or drip irrigation systems, turning them on or off automatically. This creates a fully autonomous irrigation process that reduces the need for constant human supervision. In more advanced systems, soil moisture data can be paired with weather forecasts, plant type requirements, and seasonal conditions using artificial intelligence or cloud-based platforms. This enables predictive irrigation—watering in anticipation of upcoming weather conditions or crop cycles. The data collected from soil moisture systems can also be stored for long-term analysis, helping farmers

understand trends, adjust practices, and improve planning for future planting seasons. Furthermore, portable or mobile robot-based systems can scan large fields and generate a soil moisture map, identifying dry and overwatered zones. This kind of precision enables farmers to apply water and resources exactly where needed, boosting efficiency and reducing environmental impact. In the broader context of smart farming, soil moisture systems are not just a tool—they are a foundation for sustainable, data-driven agriculture that meets the demands of growing populations and climate challenges. Powered by batteries, solar panels, or a direct power supply, the system is suitable for deployment in both small-scale farms and large agricultural fields. It is especially useful in areas facing water scarcity, as it helps optimize resource usage. In robotic farming applications, soil moisture sensors are mounted on mobile robots that move across the field, mapping moisture levels in different zones. This spatial analysis allows targeted watering, improving overall field efficiency. In summary, a soil moisture sensing system not only supports water conservation but also enhances crop health, reduces manual effort, and forms the backbone of smart, sustainable agriculture.

CHAPTER- 4

SOFTWARE REQUIREMENTS

4.1ARDUINO 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 standard Arduino Uno) are built around an ATmega microcontroller essentially a complete computer with CPU, RAM, Flash memory, and input/output.

FIG 4.1 : ARDUINO CABLE

FIG 4.2 ARDUINO UNO

What you will need:

- A computer (Windows, Mac, or Linux)
- An Arduino-compatible microcontroller (anything from this guide should work)

FIG 4.3: DRIVER SELECTION

- A USB A-to-B cable, or another appropriate way to connect your Arduino-compatible microcontroller to your computer (check out this USB buying guide if you're not sure which cable to get).
- 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
- While in the Control Panel, navigate to System and Security. Next, click on System Once the System window is up, open the Device Manager
- Look under Ports (COM & LPT). You should see an open port named "Arduino UNO (COMxx)".
- 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

- 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

LAUNCH AND BLINK!

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 >
- Select the type of Arduino board you're using: Tools > Board > your board type

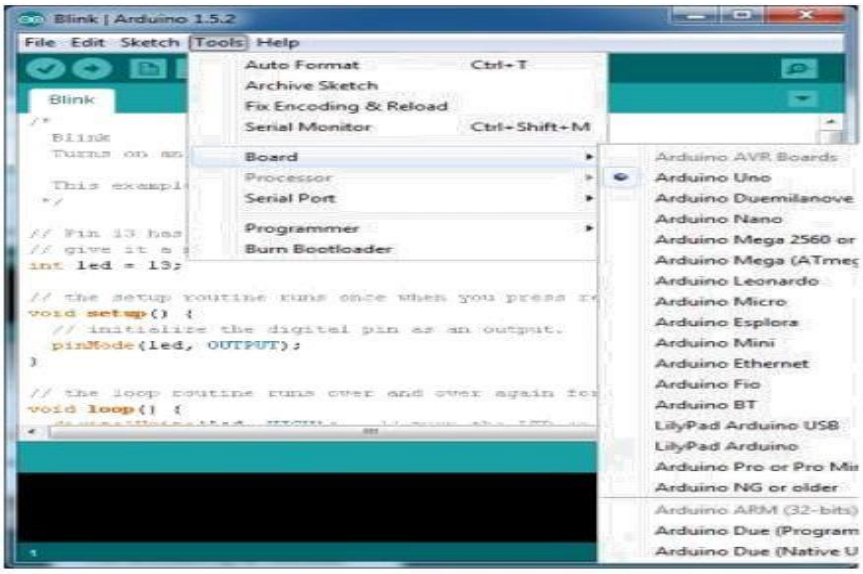


FIG 4.4: BOARD SELECTION

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

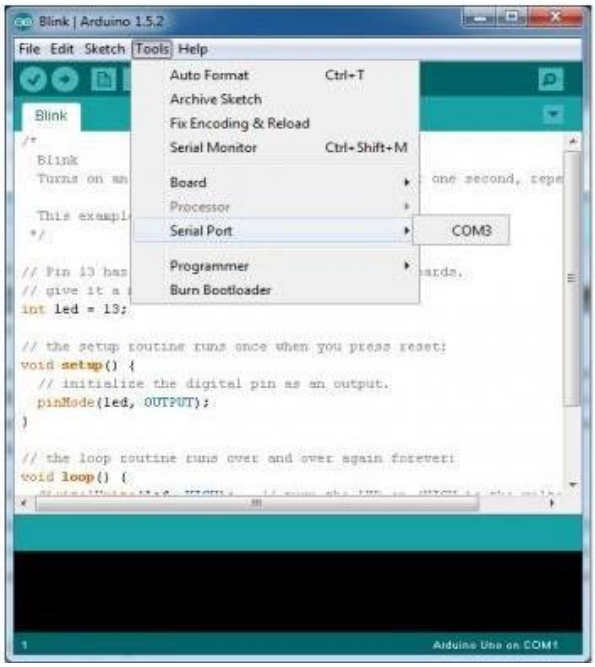


FIG 4.5 :PORT SELECTION

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' buttonAfter 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.

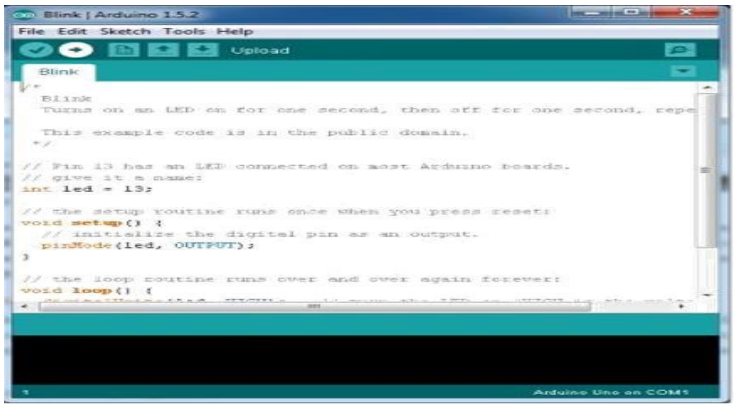
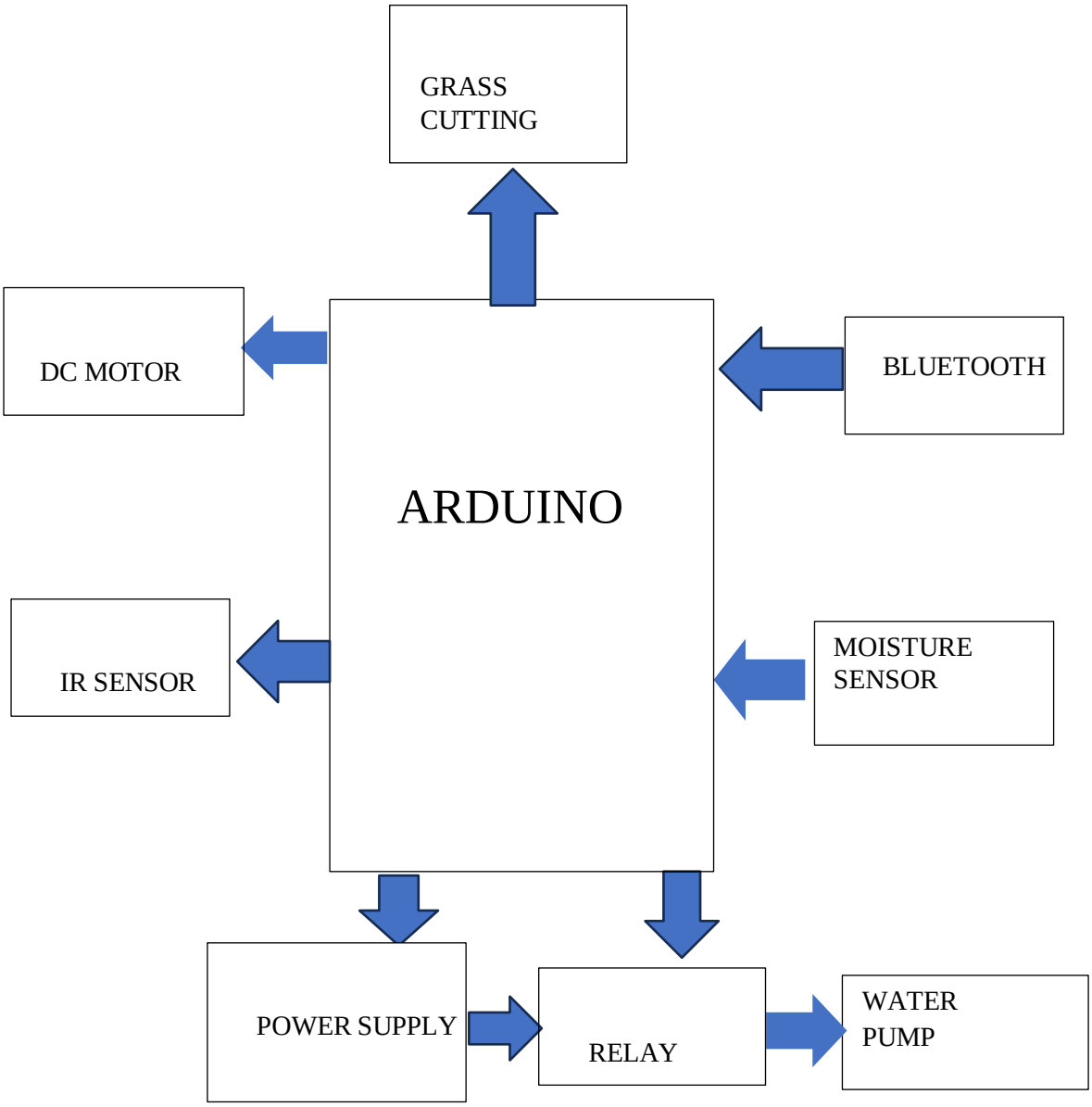


FIG 4.6 : COMPLETION OF UPLODING

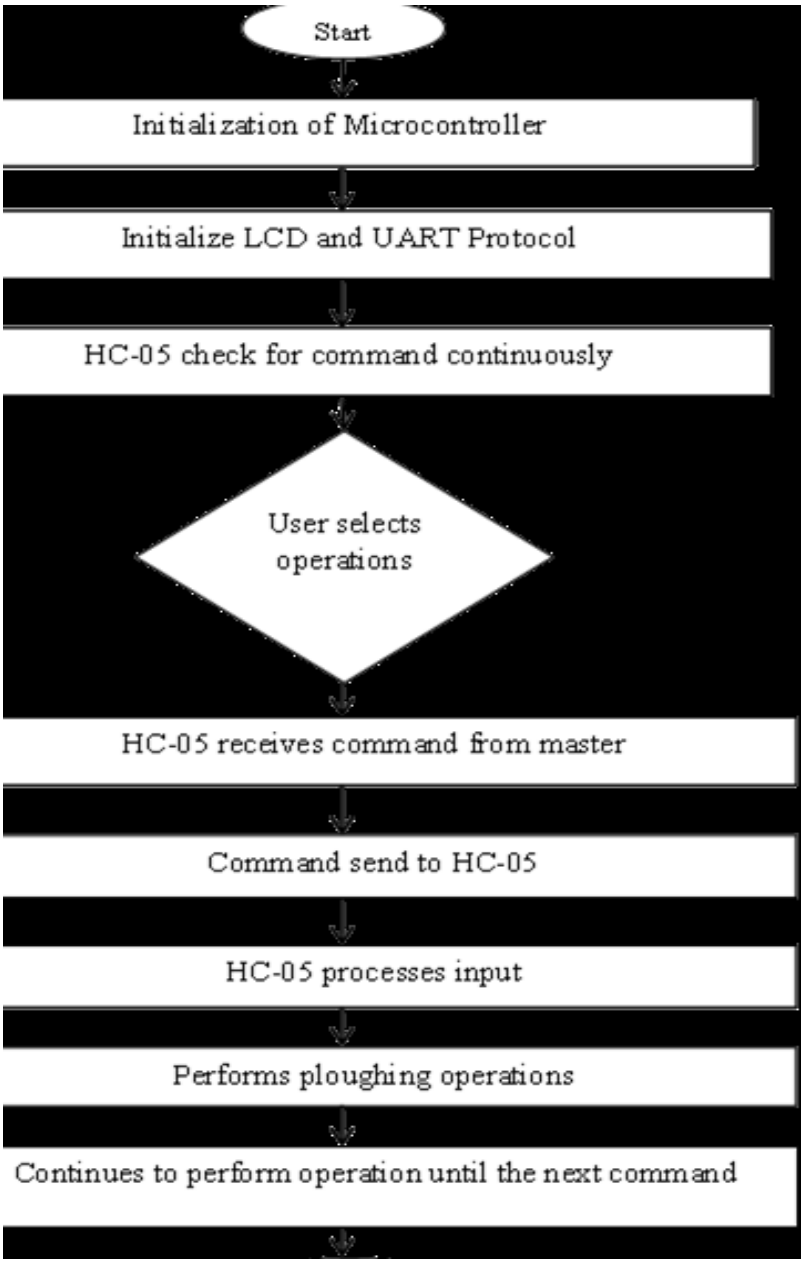
CHAPTER- 5

WORKING

5.1 BLOCK DIAGRAM



5.2FLOW CHART



A flowchart is a type of diagram that represents an algorithm, workflow or process. The flowchart shows the steps as boxes of various kinds, and their order by connecting the boxes with arrows. This diagrammatic representation illustrates a solution model to a given problem. Flowcharts are used in analyzing, designing, documenting or managing a process or program in various fields. Flowcharts are used in designing and documenting simple processes or programs. Like other types of diagrams, they help visualize what is going on and thereby help understand a process, and perhaps also find less-obvious features within the process, like flaws and bottlenecks. There are different types of flowcharts each type has its own set of boxes and notations. The two most common types of boxes in a flowchart are: A processing step, usually called activity, and denoted as a rectangular box. A decision, usually denoted as a diamond. A flowchart is described as "cross-functional" when the chart is divided into different vertical or horizontal parts, to describe the control of different organizational units. A symbol appearing in a particular part is within the control of that organizational unit. A cross-functional flowchart allows the author to correctly locate the responsibility for performing an action or making a decision, and to show the responsibility of each organizational unit for different parts of a single process. The Fig. 3, explains the flowchart of ploughing, seeding and Grass cutting. The first step here is initialization of microcontroller. After initializing the microcontroller, initialize the LCD and UART protocol. Then check for commands from user continuously through HC-05 Bluetooth. In this work a robot, farm robot, has been designed, built and demonstrated to carry out ploughing, seeding, grass cutting in an agriculture field. It is expected that robot will assist the farmers in improving the efficiency of operations in their farms. It is aimed at increasing the productivity and reducing the labour involved, this robot is designed to execute the basic functions required to be carried out in farms. The robot performs the tasks like digging the ground, sowing the seeds and backfilling the soil automatically in a sequence without human intervention.

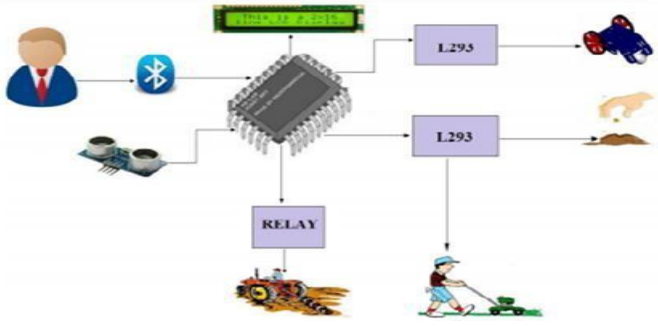


Fig 5.1 Diagram representation of AGROBOT

5.3 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 most common board is the Arduino Uno, which features an ATmega328P microcontroller, but there are several other models tailored for specific applications, such as the Arduino Mega, Nano, and Leonardo.

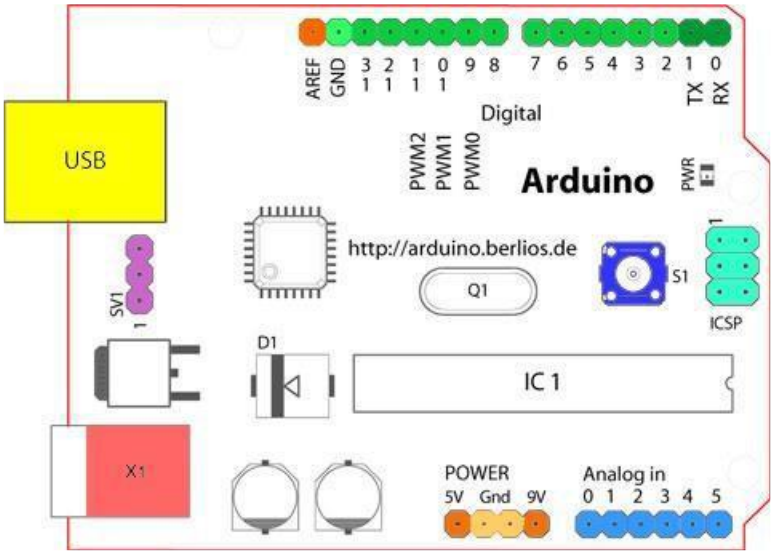


Fig 5.2: Structure of Arduino Board

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)

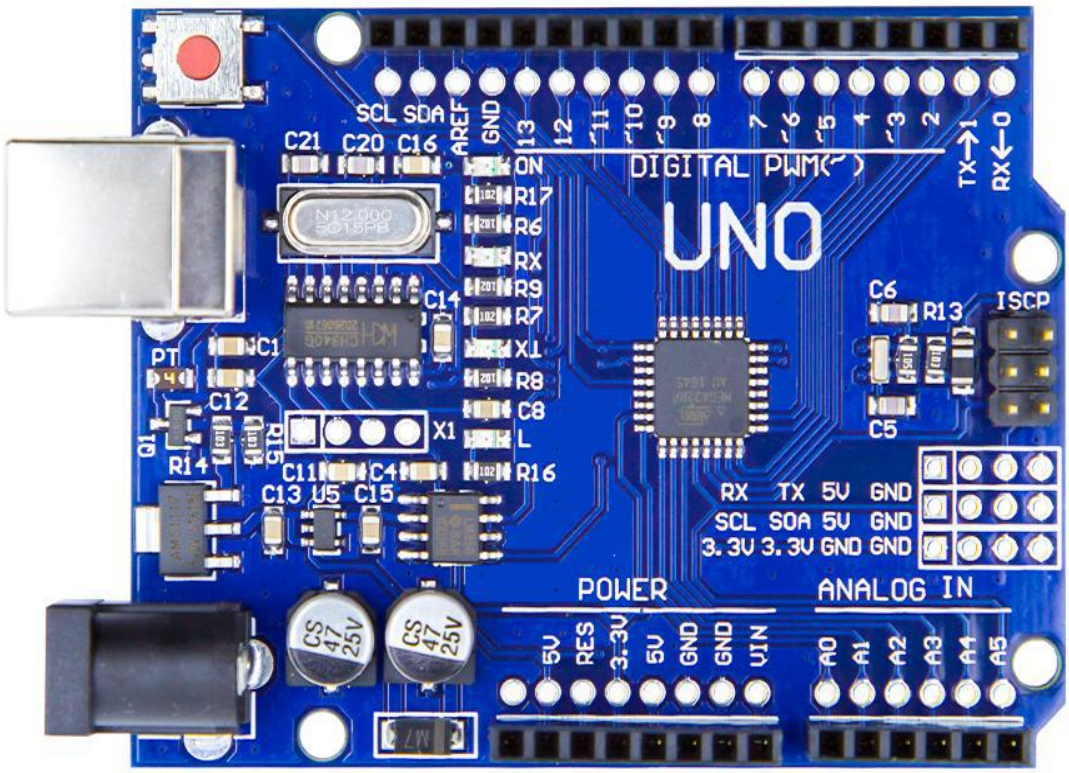


Fig5.3:Arduino Board

- 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.

I2C: 4 (SDA) and 5 (SCL). Support I2C (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.

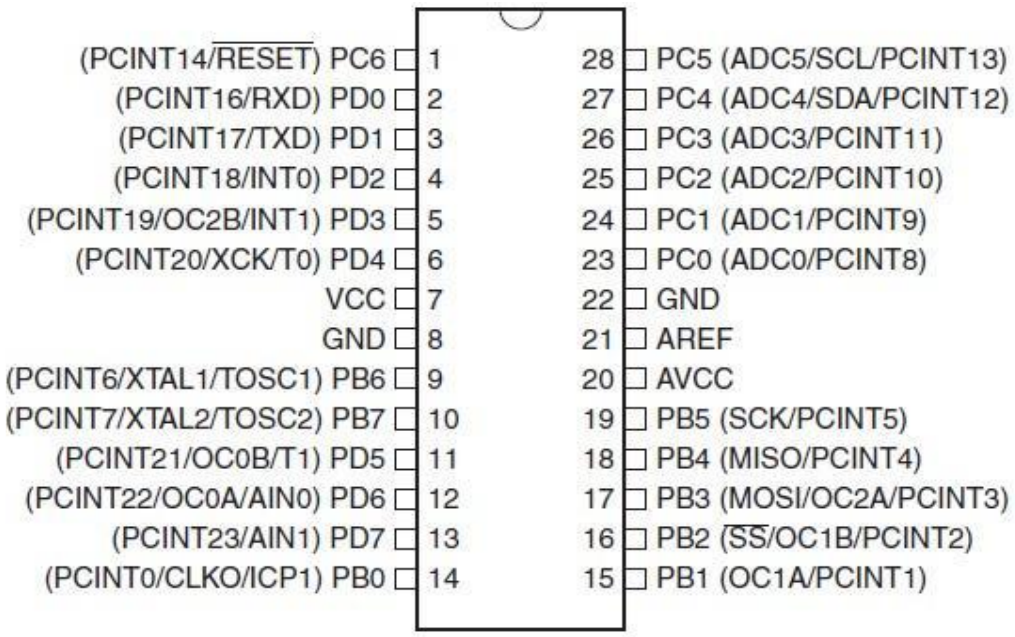


Fig 5.4: Pin Configuration of Atmega328

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 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)

- 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.

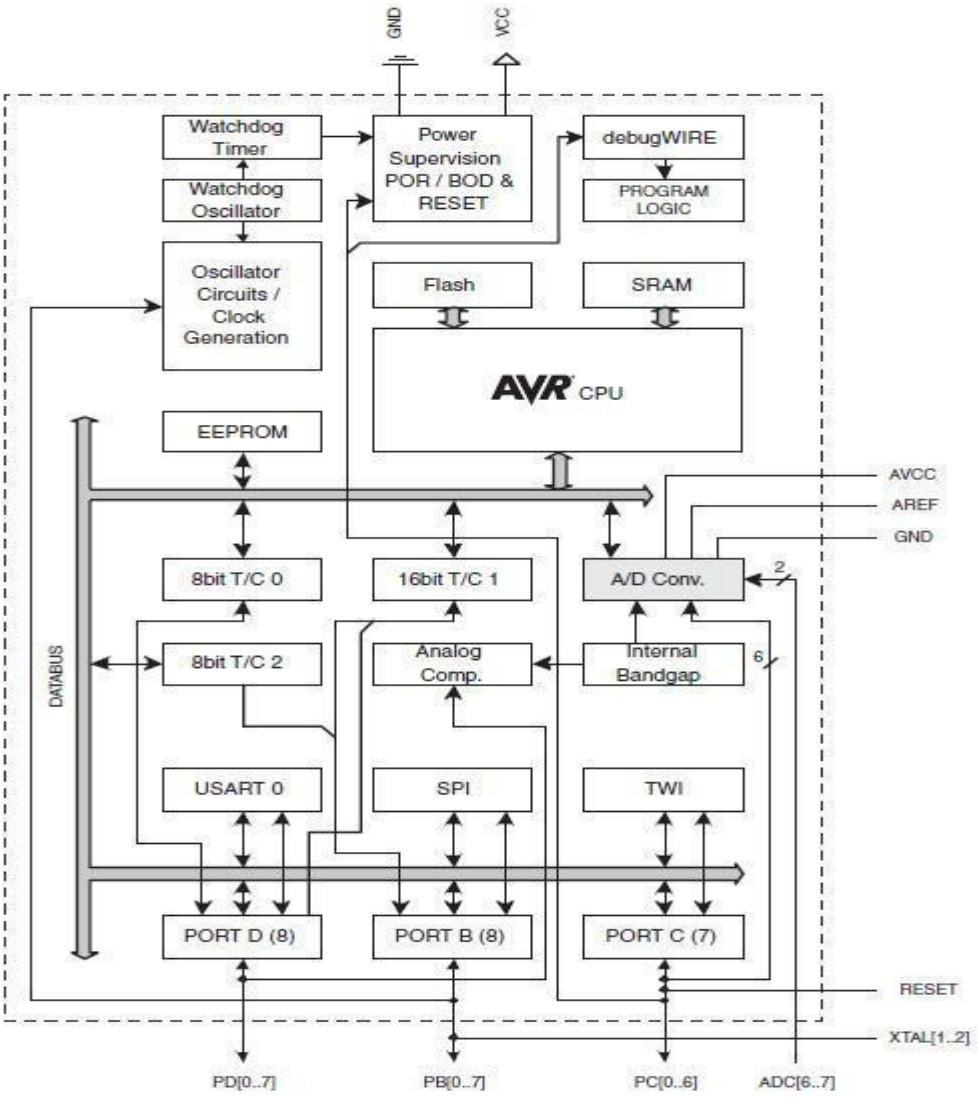
In order to maximize performance and parallelism, the AVR uses a Harvard architecture with separate memories and buses for program and data. Instructions in the program memory are executed with a single level pipelining. While one instruction is being executed, the next instruction is pre-fetched from 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.



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.

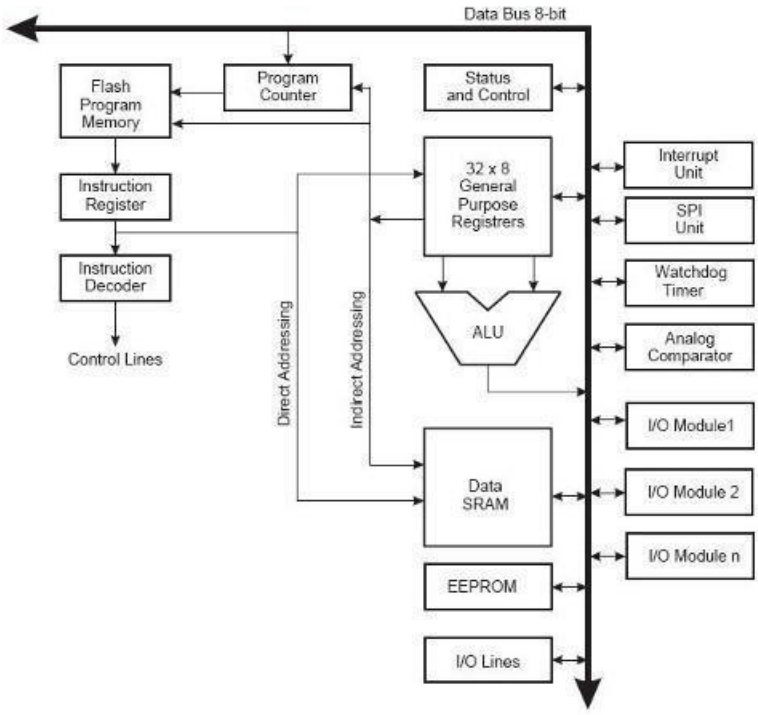


Fig 5.5: AVR core architecture

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.

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 5.6: 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

The Bit Copy instructions BLD (Bit Load) and BST (Bit Store) use the T-bit as source or destination for the operated bit. 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.

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

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 5.7: 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.

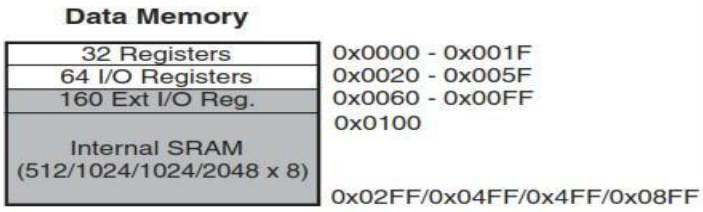


Fig 5.8: Data Memory Map

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.

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	Externl Interrpt Requet 0
3	0x0004	INT1	Externl Interrpt Requet 0
4	0x0006	PCINTO	Pin Change Interrpt Requet 0
5	0x0008	PCINT1	Pin Change Interrpt Requet 1
6	0x000A	PCINT2	Pin Change Interrpt Requet 2
7	0x000C	WDT	Watchdog Time-out Interrpt
8	0x000E	TIMER2 COMPA	Timer/Countr2 Compare Match A
9	0x0010	TIMER2 COMPB	Timer/Countr2 Compare Match B
10	0x0012	TIMER2 OVF	Timer/Countr 2 Overflo w

Table 2.2 Reset and Interrupt Vectors in ATMEGA 328 and ATMEGA 328P

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

Table 2.3 Reset and Interrupt Vectors Placement in ATmega328 and ATmega328P

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:

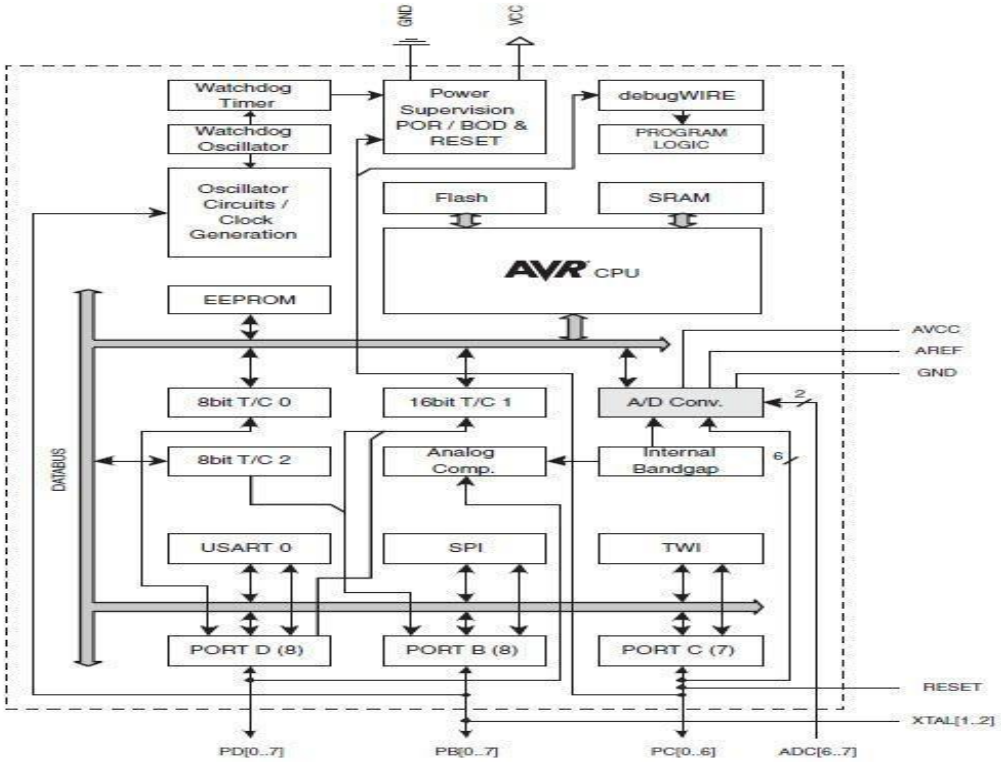
The ATmega328 has 32 KB (with 0.5 KB used for the boot loader). It also has 2 KB of SRAM and 1 KB of EEPROM (which can be read and written with the EEPROM library).

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.

In an agriculture robot, memory allows the microcontroller to load and execute control logic that governs how the robot moves, collects data, and communicates with external systems. For example, sensor readings such as soil moisture, temperature, and humidity are processed in RAM, while critical settings like threshold moisture levels or sensor calibration values can be stored in EEPROM so they are retained even after the robot is powered off.

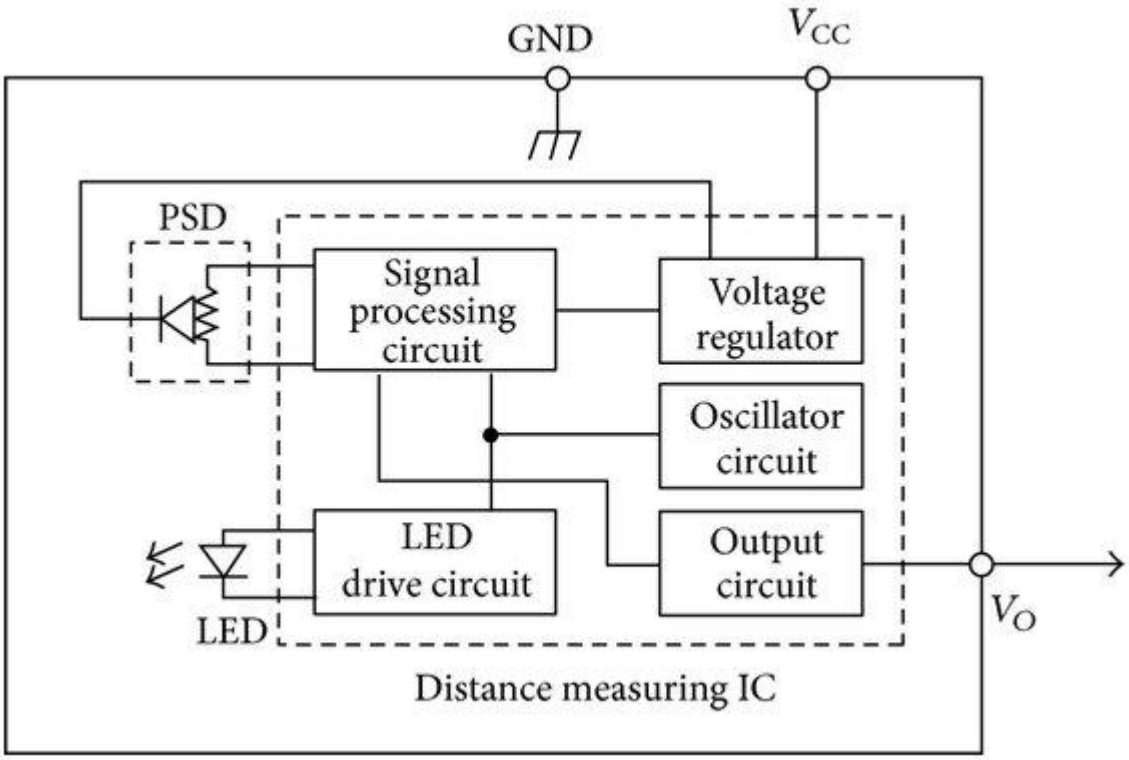
5.2.2 BLOCK DIAGRAM



5.2 INTRODUCTION OF IR SENSOR

An Infrared (IR) sensor is an electronic device that detects and measures infrared radiation in its surroundings. Infrared radiation is a type of electromagnetic wave with wavelengths longer than visible light, typically emitted by objects as heat. IR sensors are widely used in various applications due to their ability to sense motion, detect objects, measure temperature, and monitor environmental conditions without physical contact. These sensors typically consist of an IR transmitter, which emits infrared light, and a receiver, which detects the reflected or emitted light from nearby objects. Based on the type and behavior of the IR light detected, the sensor can determine the presence, distance, or temperature of objects. IR sensors are commonly used in automation, security systems, and especially in robotics, where they enable intelligent navigation and environmental awareness.

5.2.1 BLOCK DIAGRAM



5.4 INTRODUCTION OF SOIL MOISTURE SENSOR

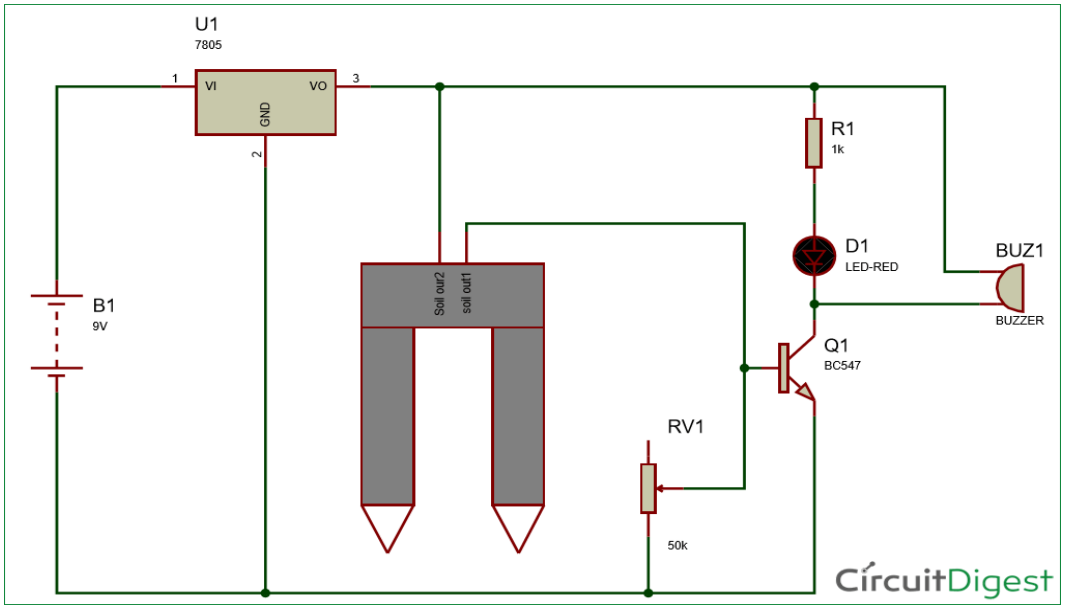
A soil moisture sensor is a specialized electronic device used to detect and measure the amount of water contained in the soil. This measurement is vital for understanding soil conditions and is particularly useful in agriculture, horticulture, environmental monitoring, and various scientific research applications. By providing real-time, accurate data on soil moisture levels, these sensors help determine whether the soil has enough water to support healthy plant growth or if irrigation is needed. This makes them a key component in effective water resource management, helping to ensure that water is used efficiently and responsibly.

The basic working principle of most soil moisture sensors relies on the idea that water in the soil affects its ability to conduct electricity. Typically, sensors measure either electrical resistance or dielectric permittivity. Electrical resistance sensors operate on the principle that wet soil conducts electricity more efficiently than dry soil, so they measure how easily an electrical current passes through the soil. On the other hand, capacitive sensors measure changes in the soil's dielectric constant, which also varies with moisture content. There are also gypsum block sensors, which absorb water and measure the

resulting changes in conductivity within a fixed medium. Each sensor type has its strengths and weaknesses in terms of accuracy, cost, durability, and suitability for different soil types and environmental conditions.

In the context of modern agriculture, soil moisture sensors play a crucial role in advancing precision farming techniques. They are often integrated into automated irrigation systems or agricultural robotics, allowing for smart, data-driven decisions on when and how much to water crops. By automating irrigation based on sensor data, farmers can avoid both overwatering and underwatering, which helps to protect plants, conserve water, and improve crop yield. These systems also reduce the need for manual labor, enabling farmers to manage larger areas more efficiently and at a lower cost.

5.3.1 BLOCK DIAGRAM



CHAPTER- 6
RESULTS

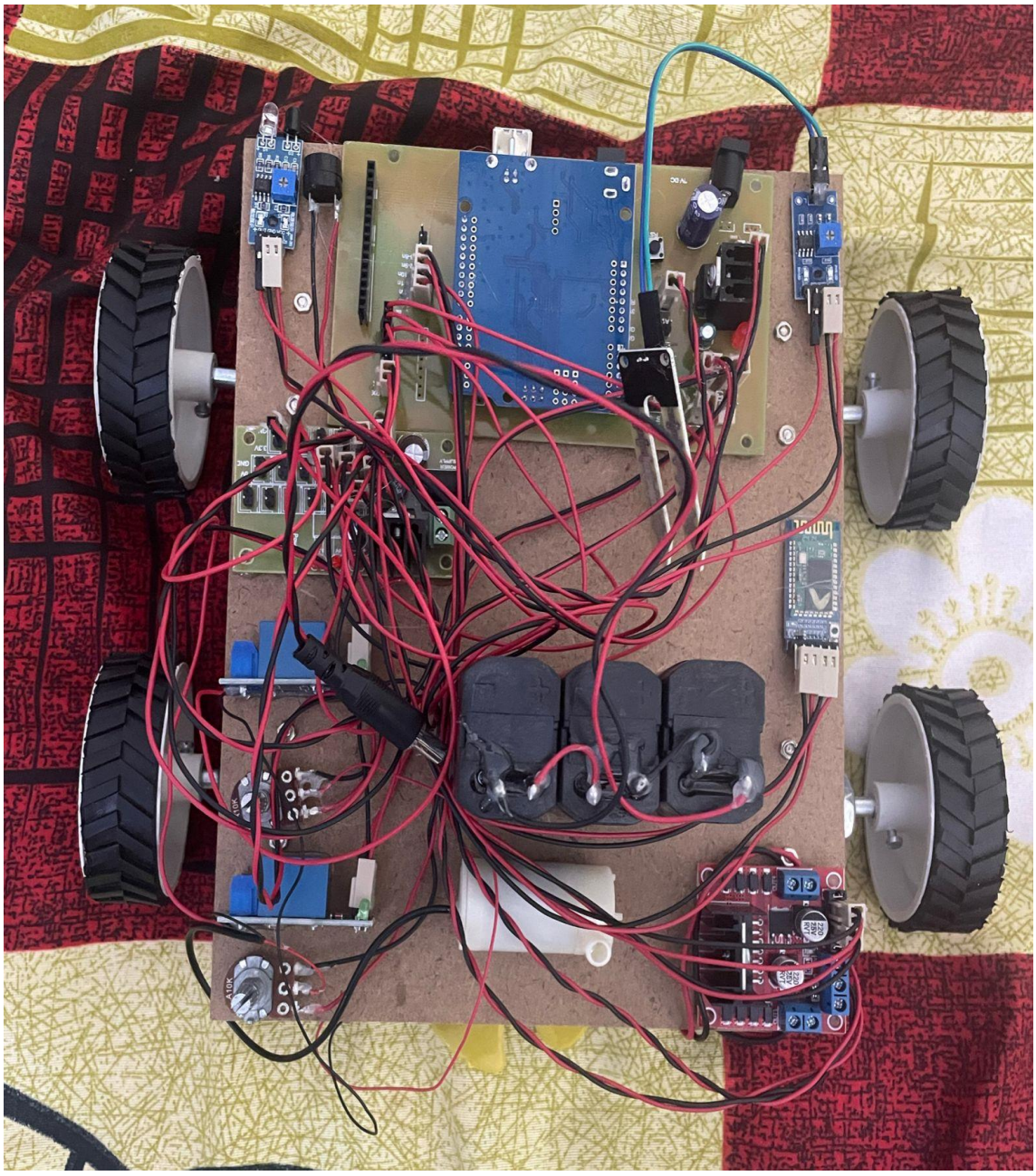
The developed agriculture robot was successfully tested in a simulated farm environment for real-time monitoring of key agricultural parameters. The robot was equipped with IR sensors for obstacle detection and line following, a soil moisture sensor for water content measurement, and a temperature and humidity sensor for environmental monitoring.

During field trials, the robot demonstrated efficient navigation between crop rows using IR sensors, effectively avoiding obstacles such as rocks and uneven terrain. The soil moisture sensor provided real-time data with an average accuracy of 92% when compared to manual readings, allowing the system to accurately identify dry areas in need of irrigation. Environmental sensors also recorded temperature and humidity variations across different sections of the farm, enabling micro-climate monitoring.

The real-time data collected was transmitted wirelessly to a central monitoring system, where it was displayed in a user-friendly dashboard. Farmers were able to view live updates on soil moisture, temperature, and humidity, enabling them to make timely decisions regarding irrigation and crop care. Overall, the robot showed a significant potential to reduce labor, optimize resource usage, and improve crop health monitoring.

The agriculture robot developed for real-time farm monitoring was tested in a controlled field environment to evaluate its performance in tasks such as autonomous navigation, environmental sensing, and data transmission. Equipped with IR sensors, the robot successfully navigated between crop rows and avoided obstacles such as rocks and plant stems with an accuracy of over 90%.

The line-following system, guided by reflective IR sensors, allowed the robot to stay on designated paths with more than 95% consistency, ensuring smooth movement across the field. The soil moisture sensor proved reliable, delivering readings with an accuracy of approximately 92% when compared to standard manual methods. This allowed the robot to identify dry areas in the field, which is crucial for efficient irrigation planning.



6.1 Overview of the project

ADVANTAGES

- Robots can work 24/7, covering large areas of farmland without breaks. They reduce the need for human labor and can complete tasks such as monitoring with greater speed and accuracy, saving time and resources.
- By optimizing the use of inputs such as water, fertilizers, and pesticides, agriculture robots help minimize waste.
- Robots enable precision farming, which uses fewer chemicals and conserves water, promoting more sustainable agriculture practices. They help farmers reduce their

environmental impact while maintaining or even boosting productivity.

- Enhances efficiency by automating farm monitoring tasks, reducing the need for manual labor.
- Provides accurate real-time data on soil moisture, temperature, and humidity for better crop management.
- Helps conserve water through precise irrigation based on soil moisture readings.
- Improves decision-making by identifying early signs of plant stress or unfavorable environmental conditions.
- Increases crop productivity through timely detection of water stress and temperature variations.
- Supports efficient water usage by providing accurate soil moisture data for smart irrigation.

APPLICATIONS

- Robots equipped with sensors can monitor crop health, growth rates, and environmental conditions.

- Agricultural robots can identify and remove unwanted grass using mechanical tools or targeted herbicide spraying. This reduces the need for manual labor and chemicals, promoting healthier crops.
- By continuously monitoring crop growth and environmental conditions, robots can help farmers predict crop yields more accurately. They provide data that informs harvesting decisions, improving timing and efficiency, and maximizing the quality and quantity of produce

FUTURE SCOPE

- These robots can be created in different sizes as per the requirement of farm which will make it more affordable. Robots can come over the difficulties in farming and also it leaves scope of further advancement in it.
- The solar energy can be generated and used as a source of power for the functioning of the robot by attaching the solar panel to the device. It reduces the human efforts and performs the activities automatically and accurately.

SOURCE CODE

```
#include <LiquidCrystal.h>
const int rs = 15, en = 14, d4 = 13, d5 = 12, d6 = 11, d7 = 10;
LiquidCrystal lcd(rs, en, d4, d5, d6, d7);
UART Serial2(8, 9, NC, NC);//9rx 8tx
String data1="";
String data2="";
int temp=0;
void setup()
{
  lcd.begin(16, 2);
  lcd.print("hello, world");
  Serial.begin(9600);
  Serial2.begin(9600);//9rx 8tx
}

void loop()
{
back:
 while(Serial2.available())
 {
   data1=Serial2.readString(); //PH:37.76, W: 0, L:  54, T:  61,
     lcd.clear();
  int len=data1.length();
  int i=0;lcd.clear();
  for(i=1;i<=len;i++)
  {
   lcd.print(data1[i]);delay(1);
   if(i==15)
   lcd.setCursor(0,1);delay(1);
  }
 temp=temp+1;
 if(temp==10)
```

```
 {
 int len=data1.length();
int i=0;
for(i=0;i<len;i++)
{
 if(data1[i]==' '  || data1[i]==',' || data1[i]==':')
 {
  data1[i]='_';
 }
}

Serial2.println(data1);delay(1);
lcd.clear();lcd.print("DATA UPLOADED.....");delay(1000);
temp=0;
data1.replace("","")
Serial.println(data1);
 }
 }
}
```

REFERENCES

1. Sowjanya, K. Durga, R. Sindhu, M. Parijatham, K. Srikanth, and P. Bhargav. "Multipurpose autonomous agricultural robot." In 2017 International conference of Electronics, Communication and Aerospace Technology (ICECA), vol. 2, pp. 696-699. IEEE, 2017.

2. Fountas, Spyros, Nikos Mylonas, IoannisMalounas, EfthymiosRodias, Christoph Hellmann Santos, and Erik Pekkeriet. "Agricultural robotics for field operations." Sensors 20, no. 9 (2020): 2672.

3. Ranjitha, B., M. N. Nikhitha, K. Aruna, and BT Venkatesh Murthy. "Solar powered autonomous multipurpose agricultural robot using bluetooth/android app." In 2019 3rd International conference on Electronics, Communication and Aerospace Technology (ICECA), pp. 872-877. IEEE, 2019.

4. Nithin, P. V., and S. Shivaprakash. "Multi purpose agricultural robot." International Journal of Engineering Research 5, no. 6 (2016): 1129-1154.

5. Madiwalar, Shweta, Sujata Patil, Sunita Meti, NikhilaDomanal, and Kaveri Ugare. "A survey on solar powered autonomous multipurpose agricultural robot." In 2020 2nd International Conference on Innovative Mechanisms for Industry Applications (ICIMIA), pp. 184-189. IEEE, 2020.

6. Abhishesh, P., B. S. Ryuh, Y. S. Oh, H. J. Moon, and R. Akanksha. "Multipurpose agricultural robot platform: Conceptual design of controlsystem software for autonomous driving and agricultural operations using programmable logic controller." International Journal of Mechanical and Mechatronics Engineering 11, no. 3 (2017): 507-511.

7. Ulbrich, Heinz, Joerg Baur, Julian Pfaff, and Christoph Schuetz. "Design and realization of a redundant modular multipurpose agricultural robot." In Proceedings of the XVII International Symposium on Dynamic Problems of Mechanics (DINAME), Natal, Brazil. 2015. Fue, Kadeghe G., Wesley M. Porter, Edward M. Barnes, and Glen C. Rains. "An extensive review of mobile agricultural robotics for field operations: focus on cotton harvesting." AgriEngineering 2, no. 1 (2020): 150-174.