A SMART REVERSE VENDING MACHINE FOR PLASTIC BOTTLES

A project report

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

BACHELOR OF TECHNOLOGY

IN

ELECTRONICS AND COMMUNICATION ENGINEERING

By

NARLA SAI KIRAN	**21JR1A04F1**
RAJARAPU PAVAN JAGADEESH	**21JR1A04G3**
SANAGAPALLI MADHAV	**21JR1A04G5**
SARIKONDA MAHENDRA SAI RAJU	**21JR1A04G9**
SARIKI APPALANAIDU	**21JR1A04G8**

Under the guidance of

MR.V. SRINIVAS RAO M.Tech

Assistant Professor, Dept. of ECE

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

KKR & KSR INSTITUTE OF TECHNOLOGY AND SCIENCES (Autonomous)

Vinjanampadu (V), Vatticherukuru (M), Guntur (Dt), A.P-52201

APRIL – 2025

KKR & KSR INSTITUTE OF TECHNOLOGY AND SCIENCES

(AUTONOMOUS)

Department of Electronics and Communication Engineering

(Approved by AICTE New Delhi || Permanently Affiliated to JNTUK, Kakinada) ||

Accredited with 'A' Grade by NAAC || NBA Accreditation)

Vinjanampadu (V), Vatticherukuru (M), Guntur (Dt), A.P-522017.

CERTIFICATE

This is to certify that this project report entitled "**A SMART REVERSE VENDINGMACHINE FOR PLASTIC BOTTLES**" submitted by

21JR1A04F1	NARLA SAI KIRAN
21JR1A04G3	RAJARAPU PAVAN JAGADEESH
21JR1A04G5	SANAGAPLLI MADHAV
21JR1A04G9	SARIKONDA MAHENDRA SAI RAJU
21JR1A04G8	SARIKI APPALA NAIDU

in partial fulfillment for the award of the Degree of Bachelor of Technology in **ELECTRONIC & COMMUNICATION ENGINEERING** to Jawaharlal Nehru University Kakinada, through KKR & KSR Institute of Technology and Sciences for the award of the Degree of Bachelor of Technology in Electronics and Communication Engineering is a bonafide record of project work carried out under supervision of **V.Srinivas Rao** during the year 2024-2025.

INTERNAL GUIDE | **HEAD OF THE DEPARTMENT**

V.SRINIVAS RAO M. Tech | Dr.N. ADI NARAYANA PhD

(Asst Professor)

INTERNAL EXAMINER | EXTERNAL EXAMINER

DECLARATION

We hereby declare that the project "**SMART REVERSE VENDING MACHINE FOR PLASTIC BOTTLES**" has been carried out by me and this work has been submitted to KKR & KSR Institute of Technology and Sciences (A), Vinjanampadu, affiliated to Jawaharlal Nehru Technological University, Kakinada in partial fulfillment of the requirements for the award of degree of Bachelor of Technology in Electronics And Communication Engineering.

We further declare that this project work has not been submitted in full or part for the award of any other degree in any other educational institutions.

1.21JR1A04F1 NARLA SAI KIRAN -

2. 21JR1A04G3 RAJARAPU PAVAN JAGADEESH -

3. 21JR1A04G5 SANAGAPALLI MADHAV -

4. 21JR1A04G9 SARIKONDA MAHENDRA SAI RAJU -

5.21JR1A04G8 SARIKI APPALANAIDU -

ACKNOWLEDGEMENT

We would like to express our profound gratitude towards V. SRINIVAS RAO, Department of Electronics and Communication Engineering, who played a supervisory role to utmost perfection, enabled us to seek through our IV-II B.Tech project and for guidance as an internal guide methodically and meticulously.

We express our gratitude towards all the faculty members and non-teaching faculty members, the Electronics and Communication Engineering.

We are highly indebted to Dr. N. ADI NARAYANA Head of the Department, Computer Science and Engineering for providing us with all the necessary support.

We render our deep sense of gratitude to Dr. P. BABU, Principal and Dr. K. HARI BABU, Director Academics for permitting us to carry out our main project works. We would like to express our sincere thanks to Computer Science and Engineering staff for lending us their time to help us complete the work successfully.

We are very much thankful to the college management Mr. K. SUBBARAO Mr. K. SHEKAR for their continuous support and the facilities provided. We would also like to thank our staff, parents, and friends for their enduring encouragement and assistance whenever required.

KKR & KSR INSTITUTE OF TECHNOLOGY AND SCIENCES

(Autonomous)

(Approved by AICTE New Delhi || Permanently Affiliated to JNTUK, Kakinada) || Accredited with 'A' Grade by NAAC || NBA Accreditation)

INSTITUTION VISION

To produce eminent and ethical Engineers and Managers for society by imparting quality professional education with an emphasis on human values and holistic excellence.

INSTITUTION MISSION

- To incorporate benchmarked teaching and learning pedagogies in the curriculum.
- To ensure the all-around development of students through a judicious blend of curricular, co-curricular, and extra-curricular activities.
- To support the cross-cultural exchange of knowledge between industry and academy.
- To provide higher/continued education and research opportunities to the employees of the institution.

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

VISION OF THE DEPARTMENT

- To be a quality education provider for technically competent and socially responsible engineers.

MISSION OF THE DEPARTMENT

- Enriched curriculum for addressing the needs of Industry and society.
- Effective teaching learning processes through congenial environment for lifelong learning
- Promote contemporary knowledge through research and development, innovation and incubation.

PROGRAM SPECIFIC OUTCOMES (PSOS)

PSO1: Professional Skills:

Apply Electronics, Communications, and VLSI, Embedded systems knowledge to arrive cost effective and appropriate solutions.

PSO2: Problem-solving skills:

Able to provide solutions and design Semiconductor Devices, Digital Systems, Microprocessor and Signal processing for the fields of Consumer Electronics, Medical, Defence and Spacecraft Electronics industry.

PSO3: Successful career:

Able to use latest hardware and software tools like VHDL, MATLAB, MULTISIM, and MENTOR GRAPHICS along with analytical skills.

PSO4: Exposure for research and development:

To analyze latest trends in Communication and apply the knowledge for the improvement in the present technology by doing research through higher education.

PROGRAM EDUCATIONAL OBJECTIVES (PEO’S)

PEO1:

Develop a strong background in basic science and mathematics and ability to use these tools in their chosen fields of specialization.

PEO2:

Have the ability to demonstrate technical competence in the fields of electronics and communication engineering and develop solutions to the problems.

PEO3:

Attain professional competence through life-long learning such as advanced degrees, professional registration, and other professional activities.

PEO4:

Function effectively in a multi-disciplinary environment and individually, within a global, societal, and environmental context.

PEO5:

Take individual responsibility and to work as a part of a team towards the fulfilment of both individual and organizational goals.

PROGRAM OUTCOMES (PO'S)

1.Engineering knowledge:

Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.

2. Problem analysis:

Identify, formulate, research literature, and analyze complex engineering problems reaching substantiated conclusions using the first principles of mathematics, natural sciences, and engineering sciences.

3. Design/development of solutions:

Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for public health and safety, and cultural, societal, and environmental considerations.

4. Conduct investigations of complex problems:

Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.

5. Modern tool usage:

Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations.

6. The engineer and society:

Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professionalengineering practice

7. Environment and sustainability:

Understand the impact of professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.

8. Ethics:

Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.

9. Individual and team work:

Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.

10. Communication:

Communicate effectively on complex engineering activities with the engineering community and with society at large, such as being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.

11.Project management and finance:

Demonstrate knowledge and understanding of the engineering and management principles and apply these to one’s own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.

12.Life-long learning:

Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

KKR & KSR INSTITUTE OF TECHNOLOGY AND SCIENCES

(Autonomous)

(Approved by AICTE New Delhi || Permanently Affiliated to JNTUK, Kakinada) || Accredited with ‘A’ Grade by NAAC || NBA Accreditation)

COURSE OUTCOMES (COS)

Course Outcomes - Program Outcomes mapping

Course Code	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12
CO409.1	2	3		2		2	2					
CO409.2	3	2	3	3		3			2	2	3	
CO409.3	3	3	3	3	2	2						1
CO409.4	3	3	2	2	3		2	2	1	3		
CO409.5	3		3	2	2		2		2		1	1
CO409.6	1					3				2		2
Average	3	3	3	3	3	3	2	2	2	3	2	2

3: High 2: Medium 1: Low

Program Educational Objectives – Program Specific Outcomes correlation

CO code	PSO1	PSO2	PSO3	PSO4
CO409.1	3			
CO409.2	3	2	1	
CO409.3	3		2	2
CO409.4	3		2	
CO409.5	3	2		
CO409.6		3		
Average	3	2	2	2

3: High 2: Medium 1: Low

CO-PO Mapping with Reasons

1. CO409.1 is mapped with PO1, PO2 and PO4, PO6, PO7 as basic knowledge of Engineering and problem Analysis activities are highly essential to conduct examinations on existing systems which have been using in industries as a part of and to define the problem of the proposed system.

2. CO409.2 is mapped with PO1, PO2, PO4 and PO6, PO9, PO10, and PO11 foridentification, gathering analysis, and classification of requirements for the proposed system, basic knowledge of engineering and Analysis steps along with complex problem analysis through the efforts of teamwork in order to meet the specific needs of the customer.

3. CO409.3 is mapped with PO2, PO5, and PO12 as to conduct the literature review and to examine the relevant systems to understand and identify the merits and demerits of each to enhance and develop the proposed as per the need.

4. CO409.4 is mapped with PO1, PO2, PO3, PO4, PO5, PO7, PO8, PO9, and PO10because modularization and design of the project are needed after requirements elicitation. For modularization and design of the project, Basic knowledge of Engineering, Analysis capabilities, Design skills, and communication is needed between team members as different modules are designed individually before integration.

5. CO409.5 is mapped with PO3, PO5, PO7, PO9, PO11 and PO12 as to construct the project latest technologies are needed. The development of the project is done individually and in groups with well-defined communication by using engineering and management principles.

6. CO409.6 is mapped with PO6, PO10, and PO12 because during and after completion of the project, documentation is needed along with proper methods of presentation through understanding and application of engineering and management principles, which in turn needs well-defined communication between the team members with all the ethical values. Even the project development team defines future enhancements as a part of the project development after identifying the scope of the project.

CO-PSOs Mapping with Reasons:

1. CO409.1 is mapped with PSO1 as examining existing systems and identification of the Problem is a part of Application Development activity and identification of evolutionary Changes in the latest technologies.

2. CO409.2 is mapped with PSO1, PSO2, and PSO3 as identifying and classifying the Requirements is a part of Application development and evolutionary computing changes and also follows ethical principles.

3. CO409.3 is mapped with PSO1 and PSO3 as a review of literature is a part of application development activity by recognizing the computing technologies and their evolutionary changes.

4. CO409.4 is mapped with PSO1 and PSO3 because modularization and logical design is also, a part of Application development and follows computing changes using Deep learning technology.

5. CO409.5 is mapped with PSO1 and PSO2 as Testing, Development, and Integration of project activities are part of Application development and follow ethical principles.

6. CO409.6 is mapped with PSO2 for project documentation and presentation the project team members apply the professional and leadership quality.

CONTENTS

List of Figures

LIST OF TABLES

List of Abbreviations

Abbreviation	**Full Form**
RVM	Reverse Vending Machine
PET	Polyethylene Terephthalate
IR	Infrared
LCD	Liquid Crystal Display
IoT	Internet of Things
MCU	Microcontroller Unit
UNO	Arduino UNO
HX711	High-precision A/D Converter Module
PCB	Printed Circuit Board
USB	Universal Serial Bus
DC	Direct Current
LED	Light Emitting Diode
GSM	Global System for Mobile Communication
EEPROM	Electrically Erasable Programmable Read-Only Memory
API	Application Programming Interface
AI	Artificial Intelligence
RFID	Radio Frequency Identification

ABSTRACT

Plastic waste, particularly in the form of single-use polyethylene terephthalate (PET) bottles, has become a major environmental concern due to its non-biodegradable nature and the growing global consumption. Current recycling systems often rely on centralized collection centers and manual processes, which can be inconvenient and discourage public participation. To address this challenge, this project introduces a cost-effective and compact Reverse Vending Machine (RVM) specifically designed for the automated collection and recycling of plastic bottles, while rewarding users for their contribution.

The proposed SMART RVM operates on the principle of "Cash from Trash," promoting responsible waste disposal through a reward-based mechanism. The system is equipped with an infrared (IR) sensor to detect the presence and size of the bottle, and a Load Cell with an HX711 amplifier to accurately measure its weight. The control and processing functions are handled by an Arduino UNO microcontroller, which calculates the appropriate reward—such as coins—based on predefined weight thresholds. Once the user inserts a bottle, the machine processes the input and dispenses coins accordingly, while storing the collected bottles internally.

This project focuses on a simplified yet efficient model of reverse vending that can be deployed in public locations such as shopping malls, railway stations, bus terminals, and educational institutions. It offers a practical solution to plastic waste management, encouraging users to participate in eco-friendly practices by providing tangible rewards.

CHAPTER -1

1 .INTRODUCTION

1.1: overview

The rapid growth of plastic consumption across the globe has led to a significant increase in plastic waste, posing a serious threat to the environment and public health. Single-use plastics, particularly plastic bottles and containers, are among the most commonly discarded items, often ending up in landfills, water bodies, or as litter in urban spaces. Despite numerous awareness campaigns and recycling programs, plastic waste continues to accumulate due to inadequate waste management infrastructure and low public participation in recycling efforts.

To combat this pressing issue, innovative technological solutions are being explored to make recycling more accessible, efficient, and engaging for individuals. One such solution is the development of a **SMART Reverse Vending Machine (RVM)**—a modern, automated system designed to encourage responsible disposal and recycling of plastic waste. Unlike traditional waste bins, RVMs utilize advanced technologies such as **sensors**, RFID (Radio Frequency Identification), and to identify, sort, and process plastic materials.

This project focuses on the design and implementation of a SMART RVM that not only simplifies the recycling process but also incentivizes users for their eco-friendly actions. When a user deposits an empty plastic bottle into the machine, the system automatically detects the type of plastic, verifies its recyclability, and sorts it accordingly. In return, users are rewarded with points**,** discount vouchers, or can choose to contribute their rewards towards environmental or social causes. These rewards serve as motivation, aiming to increase user engagement and promote sustainable habits.

By combining automation, artificial intelligence, and gamification elements, the SMART RVM project aspires to create a smarter recycling ecosystem. It is especially suitable for deployment in high-traffic public areas such as schools, malls, railway stations, and community centers, where it can significantly contribute to waste reduction and awareness.

This documentation outlines the conceptualization, design, and implementation of the SMART RVM, highlighting its objectives, methodology, hardware and software requirements, system architecture, and the expected impact on plastic waste management.

1.2: INTRODUCTION TO EMBEDDED SYSTEM

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

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

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

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

In general, "embedded system" is not a strictly definable term, as most systems have some element of extensibility or programmability. For example, handheld computers share some elements with embedded systems such as the operating systems and microprocessors that power them, but they allow different applications to be loaded and peripherals to be connected. Moreover, even systems that do not expose programmability as a primary feature generally need to support software updates. On a continuum from "general purpose" to "embedded", large application systems will have subcomponents at most points even if the system as a whole is "designed to perform one or a few dedicated functions", and is thus appropriate to call ‘embedded’.

An embedded system is not a computer system that is used primarily for processing, not a software system on PC or UNIX, not a traditional business or scientific application. High-end embedded & lower end embedded systems. High-end embedded system - Generally 32, 64 Bit Controllers used with OS. Examples Personal Digital Assistant and Mobile phones etc .Lower end embedded systems - Generally 8,16 Bit Controllers used with an minimal operating systems and hardware layout designed for the specific purpose.

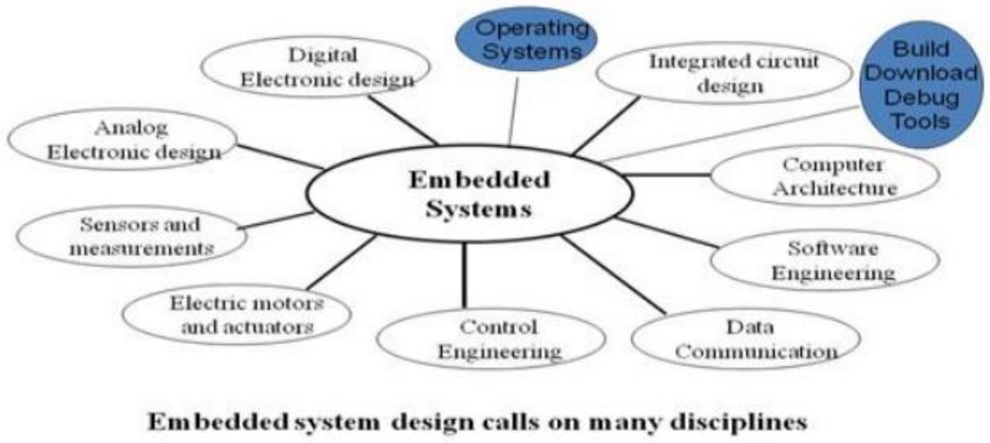


Figure 1.1: Embedded system design calls

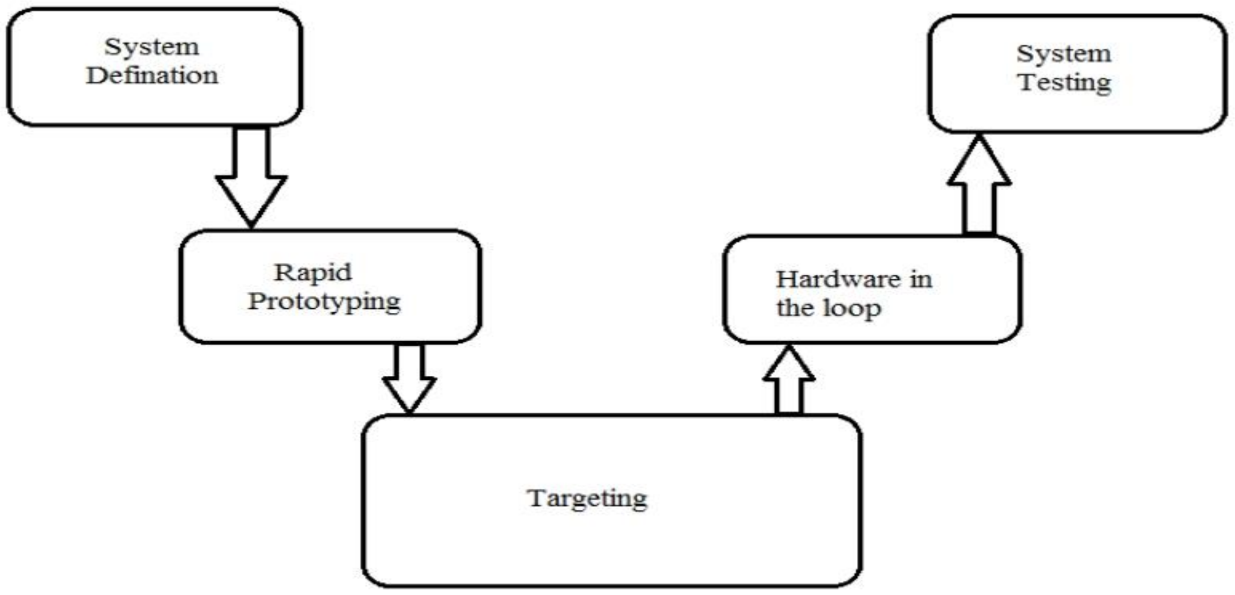


Figure 1.2: V Diagram

1.2.1: Examples of Embedded Systems:

Embedded systems are found in wide range of application areas. Originally they were used only for expensive industrial control applications, but as technology brought down the cost of dedicated processors, they began to appear in moderately expensive applications such as automobiles, communication and office equipments and television Today's embedded systems are so inexpensive that they are used in almost every electronic product in our life. Embedded systems are often designed for mass production.

Some examples of embedded systems:

1. Automatic Teller Machines
2. Cellular telephone and telephone switches
3. Computer network equipment
4. Computer printers
5. Disk drives
6. Engine controllers and antilock brake controllers for automobiles
7. Home automation products
8. Handheld calculators
9. Household appliances

10. Medical equipment
11. Measurement equipment
12. Multifunction wrist watches
13. Multifunction printers

1.3 Characteristics of Embedded System:

Embedded systems are designed to do some specific task, rather than be a general-purpose computer for multiple tasks. Some also have real-time performance constraints that must be met, for reasons such as safety and usability; others may have low or no performance requirements, allowing the system hardware to be simplified to reduce costs.

Embedded systems are not always standalone devices. Many embedded systems consist of small, computerized parts within a larger device that serves a more general purpose. For example, the Gibson Robot Guitar features an embedded system for tuning the strings, but the overall purpose of the Robot Guitar is, of course, to play music. Similarly, an embedded system in an automobile provides a specific function as a subsystem of the car itself.

The program instructions written for embedded systems are referred to as firmware, and are stored in read-only memory or Flash memory chips. They run with limited computer hardware resources: little memory, small or non-existent keyboard and/or screen.

An embedded system is any computer system hidden inside a product other than a computer. They will encounter a number of difficulties when writing embedded system software in addition to those we encounter when we write applications.

1. Throughput – Our system may need to handle a lot of data in a short period of time.
2. Response–Our system may need to react to events quickly.
3. Testability–Setting up equipment to test embedded software can be difficult.
4. Debugability–Without a screen or a keyboard, finding out what the software is doing wrong (other than not working) is a troublesome problem.
5. Reliability – embedded systems must be able to handle any situation without human intervention.

6. Memory space – Memory is limited on embedded systems, and you must make the software and the data fit into whatever memory exists.
7. Program installation – you will need special tools to get your software into embedded systems.
8. Power consumption – Portable systems must run on battery power, and the software in these systems must conserve power.
9. Processor hogs – computing that requires large amounts of CPU time can complicate the response problem.
10. Cost – Reducing the cost of the hardware is a concern in many embedded system projects; software often operates on hardware that is barely adequate for the job.

Embedded systems have a microprocessor/ microcontroller and a memory. Some have a serial port or a network connection. They usually do not have keyboards, screens or disk drives

1.3.1:User interfaces:

Embedded systems range from no user interface at all –Dedicated only to one task - to full user interfaces similar to desktop operating systems in devices such as PDAs.

1.3.2 Peripherals:

Embedded systems range from no user interface at all – dedicated only to one task - to complex graphical user interfaces that resemble modern computer desktop operating systems. Simple embedded devices use buttons, LEDs, graphic or character LCDs (for example popular HD 44780LCD) with a simple system.

- Embedded systems talk with outside world via peripherals, such as:
- Serial Communications Interfaces (SCI): RS-232, RS-422, RS-485 etc.
- Synchronous Serial Communication Interface:I2c,SPI,SSC and ESSI(Enhanced
- Synchronous Serial Interface)
- Universal Serial Bus (USB)
- Multi Media Cards (SD Cards, Compact Flash etc.)
- Networks: Ethernet, Lon Works etc.

- Field buses: CAN-Bus, LIN-Bus, PROFIBUS etc.
- Timers: PLLS(s), Capture /Compare and Time Processing Units
- Discrete IO: Aka General Purpose Input / output (GPIO)
- Analog to Digital or Digital to Analog(ADC/DAC)
- Debugging: JTAG, ISP, ICSP, BDM Port, BITP and DP9 ports.

1.3.3 Debugging:

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

- Interactive resident debugging using the simple shell provided by the embedded operating system(e.g. Forth and Basic)
- External debugging using logging or serial port output to trace operating using either a monitor in flash or using a debug server like the Remedy Debugger which even works for heterogeneous multi core systems.
- An in-circuit debugger (ICD), a hardware device that connects to the microprocessor via a JTAG or NEXUS interface. This allows the operation of the microprocessor to be controlled externally, but is typically restricted to specific debugging capabilities in the processor.
- An in- circuit emulator (ICE) replaces the microprocessor with a simulated equivalent, providing full control over all aspects of the microprocessor.
- A complete emulator provides a simulation of all aspects of the hardware, allowing all of it to be controlled and modified and allowing debugging on a normal PC.

1.4 Applications:

1. Military and aerospace embedded software applications
2. Communication Applications, Intelligent, autonomous sensors.
3. Mastering the complexity of applications.
4. Reduction of product design time.
5. Real time processing of ever increasing amounts of data

CHAPTER 2

LITERATURE SERVEY

2.1 Paper 1: Nandhini et al. (2023) – Design and Development of Smart Reverse Vending System for PET Bottles Collection and Recycling Using Design Thinking

2.1.1Overview:

In response to the growing issue of plastic waste in public areas, the authors propose an automated recycling bin equipped with a reward mechanism. By applying the design thinking approach, they conducted extensive surveys to understand public attitudes toward recycling, identifying barriers and motivators. The resulting system aims to make recycling more accessible and appealing, thereby promoting environmental cleanliness.

2.1.2 Methodology:

The development of the smart RVM involved several key components

- **Sensors:** Capacitive and inductive proximity sensors (IDEC sensors) detect the insertion of plastic bottles and verify their material.
- **Microcontrollers:** An Arduino microcontroller manages the automated functions, while a Node MCU monitors the system's status and communicates with waste collectors when the storage nears capacity.
- **Weighing Mechanism:** A force sensor measures the weight of each deposited bottle, determining the corresponding user reward.
- **Grinding Mechanism:** Accepted bottles are transported to a grinding machine where they are reduced to small scraps, optimizing storage and subsequent recycling processes
- **Storage:** An underground storage unit collects the ground plastic. Once the storage reaches approximately 48 kilograms, the system alerts waste collectors via GPS to facilitate timely collection.

2.1.3 Disadvantages:

While the proposed system offers numerous benefits, certain limitations are acknowledged

- **Detection Accuracy:** The system may require enhancements to improve the accuracy of distinguishing between plastic and non-plastic items.
- **User Identification:** Implementing a more straightforward method for user identification could enhance user experience and system efficiency.
- **Material Limitations:** Currently, the machine is designed to accept only plastic bottles; expanding its capabilities to process glass bottles and other materials would increase its utility
- **Power Supply:** Integrating renewable energy sources, such as solar power, could address sustainability concerns and reduce operational costs.

2.1.4 Conclusion:

The smart reverse vending system represents a significant advancement in promoting recycling behaviors and managing plastic waste effectively. By combining IoT technology with user incentives, the system encourages public participation in recycling initiatives. Future developments should focus on enhancing detection accuracy, expanding material acceptance, simplifying user interactions, and incorporating sustainable energy solutions to further improve the system's effectiveness and environmental impact.

2.2Paper 2: Sakshi S. Jain, Swati S. Dhotre, and Prof. Snehal D. Dhawale-AReview on Reverse Vending Machine

2.2.1Overview:

This study focuses on the design of an automated recycling bin inspired by RVMs, incorporating a reward system to motivate users to deposit recyclable materials. The proposed system integrates a microcontroller and various sensors within a standard recycling bin to identify and process deposited items. Upon completion of the recycling process, the system generates a QR code displayed on an LCD screen, which users can scan to redeem points.

2.2.2 Methodology:

The methodology involves the integration of specific hardware components:

- **Sensors:** Capacitive and inductive proximity sensors detect and identify the materials of the deposited items.
- **Microcontroller:** An Arduino microcontroller manages the system's operations, processing input from sensors and coordinating subsequent actions.
- **User Interface:** An LCD screen displays a QR code upon successful processing of recyclable items, allowing users to scan and accumulate reward points.

The system is designed to be installed in public spaces such as subways, train stations, colleges, and public buildings to promote recycling behaviors.

2.2.3 Disadvantages:

The paper does not explicitly discuss the disadvantages of the proposed system. However, potential limitations may include:

- **Material Recognition Accuracy:** The effectiveness of sensors in accurately distinguishing between different recyclable materials could impact the system's efficiency.
- **User Engagement:** The reliance on QR codes for rewards necessitates that users have compatible devices and are willing to engage with the system, which may limit participation.
- **Maintenance Requirements:** Regular maintenance is essential to ensure sensor accuracy and overall system functionality, potentially increasing operational costs.

2.2.4 Conclusion:

The proposed reverse vending machine offers a promising approach to enhancing waste management and recycling efforts by integrating technology with user incentives. By automating the collection and processing of recyclable materials and rewarding user participation, the system aims to promote environmentally responsible behaviors in public spaces. Future work could focus on addressing potential limitations, such as improving material recognition accuracy, expanding the range of accepted materials, and exploring alternative reward mechanisms to increase user engagement.

2.3: Paper 3: Judy Ann B. Bari bad,Ivy M. Tarun- Plastic2Fantastic: Reverse Vending Machine for Plastic Bottles

2.3.1Overview:

This study introduces the "Plastic2Fantastic" Reverse Vending Machine, designed to accept and process empty plastic beverage containers in exchange for monetary rewards. The machine aims to promote recycling by providing financial incentives, thereby encouraging responsible waste disposal behaviors.

2.3.2 Methodology:

- **Hardware Components:** The machine is constructed using aluminum and incorporates a Raspberry Pi 3B, 7-inch Touch Screen LCD, RFID reader, coin hopper, ultrasonic sensor, capacitive and inductive proximity sensors, IR proximity sensor, and M995sg servo motor.
- **Functionality:** Users deposit plastic bottles, which are identified and verified by the system. Upon successful verification, the machine dispenses coins as a reward.
- **User Interaction:** The system includes an RFID card reader to identify users and track their recycling activity

.

2.3.3 Disadvantages:

- **Complexity of Components:** The integration of multiple hardware components may increase the complexity and cost of the system.
- **Maintenance Requirements:** Regular maintenance is necessary to ensure the proper functioning of sensors and mechanical parts.
- **Scalability:** The design may require modifications to be scalable for different environments or to accommodate various types of recyclable materials.

2.3.4Conclusion:

The "Plastic2Fantastic" RVM effectively combines technology and financial incentives to promote plastic bottle recycling. Functionality tests indicate high performance, and user evaluations suggest strong agreement on the machine's usefulness. The system demonstrates potential in fostering recycling habits and contributing to environmental sustainability.

2.4Paper 4: Shilpa Sambhi, Preeti-Reverse Vending Machine for Managing PlasticWaste

2.4.1Overview:

This paper discusses the development of a Reverse Vending Machine designed to address plastic pollution by automating the collection and compaction of plastic bottles. The machine aims to reduce the volume of plastic waste, facilitating easier handling and recycling.

2.4.2 Methodology:

- **Integration of Technologies:** The system combines sensor technology, LabVIEW programming, data acquisition, and pneumatics to automate the process of plastic bottle collection and compaction.

- **Operational Process:** Users deposit plastic bottles into the machine, which are then detected by sensors. The system activates a pneumatic mechanism to crush the bottles, reducing their size for more efficient storage and transportation.

2.4.3 Disadvantages:

- **Energy Consumption:** The use of pneumatic systems may lead to higher energy consumption, impacting operational costs.
- **Maintenance Needs:** The mechanical components, especially the pneumatic system, require regular maintenance to ensure consistent performance.
- **User Engagement:** The system lacks a reward mechanism, which may result in lower user participation compared to incentive-based models.

2.4.4Conclusion:

The proposed Reverse Vending Machine offers a technological solution to manage plastic waste by automating the collection and compaction processes. While it effectively reduces the volume of plastic waste, incorporating user incentives and addressing energy efficiency could enhance its adoption and sustainability.

2.5Paper 5: MuhammadSallehuddin Mohd Adzharia, Afandi Ahmad- IOT-Based Reverse Vending Machine (RVM) for Recycle Station

2.5.1Overview:

This paper presents the development of an Internet of Things (IoT)-based Reverse Vending Machine aimed at encouraging recycling habits and increasing environmental awareness. The system is designed to detect and sort three types of waste materials: aluminum cans, plastic, and glass. It notifies users when the storage is full and incorporates a reward system to motivate participation.

2.5.2 Methodology:

- **Sensors:** Utilizes inductive and capacitive proximity sensors to identify the type of waste material deposited.

- **Microcontrollers:** Employs Arduino Mega and NodeMCU to process sensor data and transmit information to an IoT platform via Wi-Fi
- **User Interface:** Incorporates an I2C LCD display and the Blynk IoT platform to provide real-time feedback and display the reward system to users.

2.5.3 Disadvantages:

- **Limited Waste Types:** The system is designed to handle only three specific types of waste, which may limit its applicability.
- **Sensor Accuracy:** The effectiveness of material detection relies heavily on sensor precision, which could be affected by environmental factors.
- **Maintenance Needs:** Regular maintenance is required to ensure sensor functionality and system reliability.

2.5.4Conclusion:

The developed IOT-based RVM successfully detects and sorts aluminum cans, plastic, and glass, providing real-time notifications and a reward system to encourage recycling. The integration of IOT enhances user engagement and operational efficiency, contributing to effective waste management practices.

CHAPTER 3

3. EXISTING WORK WITH REFERENCES

3.1 Introduction

Reverse Vending Machines (RVMs) are automated systems designed to collect used beverage containers, particularly plastic bottles and cans, in an efficient and user-friendly manner. Unlike traditional vending machines that dispense products, RVMs operate in reverse by accepting waste items and offering incentives to users in return. These systems are widely used in countries with deposit return schemes (DRS), where consumers receive monetary or coupon-based rewards for returning recyclable containers.

The primary goal of RVMs is to reduce plastic pollution, promote recycling habits, and ensure that plastic waste is channeled into proper recycling streams rather than ending up in landfills or the environment. These systems are typically installed in public places such as supermarkets, train stations, and schools to maximize accessibility and encourage widespread participation.

3.2 Advantages of Existing RVM Systems

1. **Encourages Recycling Behavior**
    - Provides a tangible reward, which motivates users to return recyclable items
2. **Reduces Littering**
    - Helps minimize plastic waste in public areas and natural environments.
3. **Streamlined Waste Collection**
    - Bottles collected are already sorted and compacted, reducing labor and transport costs
4. **Data Collection for Analysis**
    - Central systems can track usage, quantity collected, and trends in recycling behavior.

5. **Easy to Use**
    - Simple interface with guided steps makes the system accessible to all age groups.
6. **Supports Government Policies**
    - Aligns with environmental regulations and supports circular economy initiatives.

3.3 Disadvantages of Existing RVM Systems

1. Limited Recognition Capabilities

- Often depends solely on barcodes or shape, making it ineffective for unlabelled or damaged bottles.

2. Lack of AI Integration

- No intelligent decision-making or image analysis to handle a wide variety of plastic types.

3. No Waste Segregation

- Cannot distinguish between PET, HDPE, or non-recyclable plastics.

4. Internet or Power Dependency

- Requires constant internet/power; operation may fail in low-resource settings.

5. Limited Rewards System

- Usually tied to physical vouchers; lacks integration with mobile wallets or loyalty app

6. High Initial Cost

- Commercial RVMs are expensive to install and maintain, limiting adoption in developing regions

.

CHAPTER-4

4. PROPOSED WORK

4.1 Introduction to the Proposed System

The proposed SMART Reverse Vending Machine (RVM) presents an innovative and automated solution to address the growing issue of plastic bottle waste. Traditional recycling systems often fall short due to their reliance on manual processes and lack of user engagement. In contrast, the SMART RVM is designed to streamline the collection and preliminary sorting of polyethylene terephthalate (PET) bottles through a fully automated mechanism. By integrating basic yet effective hardware components such as an infrared (IR) sensor and a Load Cell with HX711 amplifier, the system can detect, validate, and weigh inserted plastic bottles. Central to the system is the Arduino UNO microcontroller, which processes sensor inputs and determines the appropriate reward to be dispensed, thereby motivating users to recycle through a “Cash from Trash” approach.

This compact and cost-effective solution is intended for deployment in high-traffic public areas such as malls, railway stations, schools, and bus terminals. It not only simplifies the recycling process for users but also ensures efficient collection and temporary storage of plastic waste. The reward mechanism fosters environmental responsibility and public participation, aiming to create a habit-forming recycling behavior. The proposed system is scalable and modular, allowing for future upgrades such as IoT integration, digital reward platforms, and enhanced material recognition through AI. Ultimately, the SMART RVM promotes sustainable waste management by merging technology with everyday environmental actions.

4.2 Advantages of the Proposed SMART RVM System

1. **Automated Operation**: Reduces human intervention and increases efficiency in plastic bottle collection.

2. **User Incentivization**: Provides instant rewards (e.g., coins or digital points), motivating more people to recycle.
3. **Low-Cost Implementation**: Uses affordable components like Arduino UNO and IR sensors, making it cost-effective.
4. **Compact and Portable**: Can be installed in a variety of public spaces with minimal space requirements.
5. **Real-Time Processing**: Immediate detection, weight calculation, and reward dispensing.
6. **Environmentally Friendly**: Contributes to waste reduction and responsible plastic disposal.
7. **Scalability**: Can be upgraded with IOT and cloud integration for large-scale deployment.
8. **Educational Value**: Encourages environmental awareness, especially in schools and universities.

4.3 Applications of the Proposed SMART RVM System

1. **Shopping Malls and Supermarkets**
   a. Encourages customers to recycle plastic bottles while offering instant rewards like vouchers or discounts.
   b. Increases customer engagement and promotes eco-friendly brand image.

1. **Educational Institutions (Schools, Colleges, Universities)**
   a. Instills a habit of recycling among students.
   b. Can be integrated into environmental awareness programs and competitions.
2. **Railway Stations, Metro Stations, and Bus Terminals**
   a. Helps manage plastic waste in high-footfall public transport hubs.
   b. Promotes cleanliness and contributes to Swachh Bharat and similar missions.
3. **Corporate Offices and Tech Parks**
   a. Supports sustainability efforts within organizations.
   b. Encourages responsible waste disposal by employees.
4. **Hospitals and Healthcare Facilities**
   a. Reduces plastic pollution in sensitive health environments.
   b. Ensures systematic disposal of plastic waste.
5. **Smart Cities and Municipal Areas**
   a. Contributes to smart waste management infrastructure.
   b. Can be part of integrated IOT-based urban recycling solutions.

6. **Events, Fairs, and Exhibitions**
    a. Temporary installations for mass awareness and waste control at public events.
    b. Encourages responsible disposal in areas prone to littering.
7. **Residential Societies and Gated Communities**
    a. Promotes domestic-level recycling.
    b. Offers community-level incentives and environmental benefits.

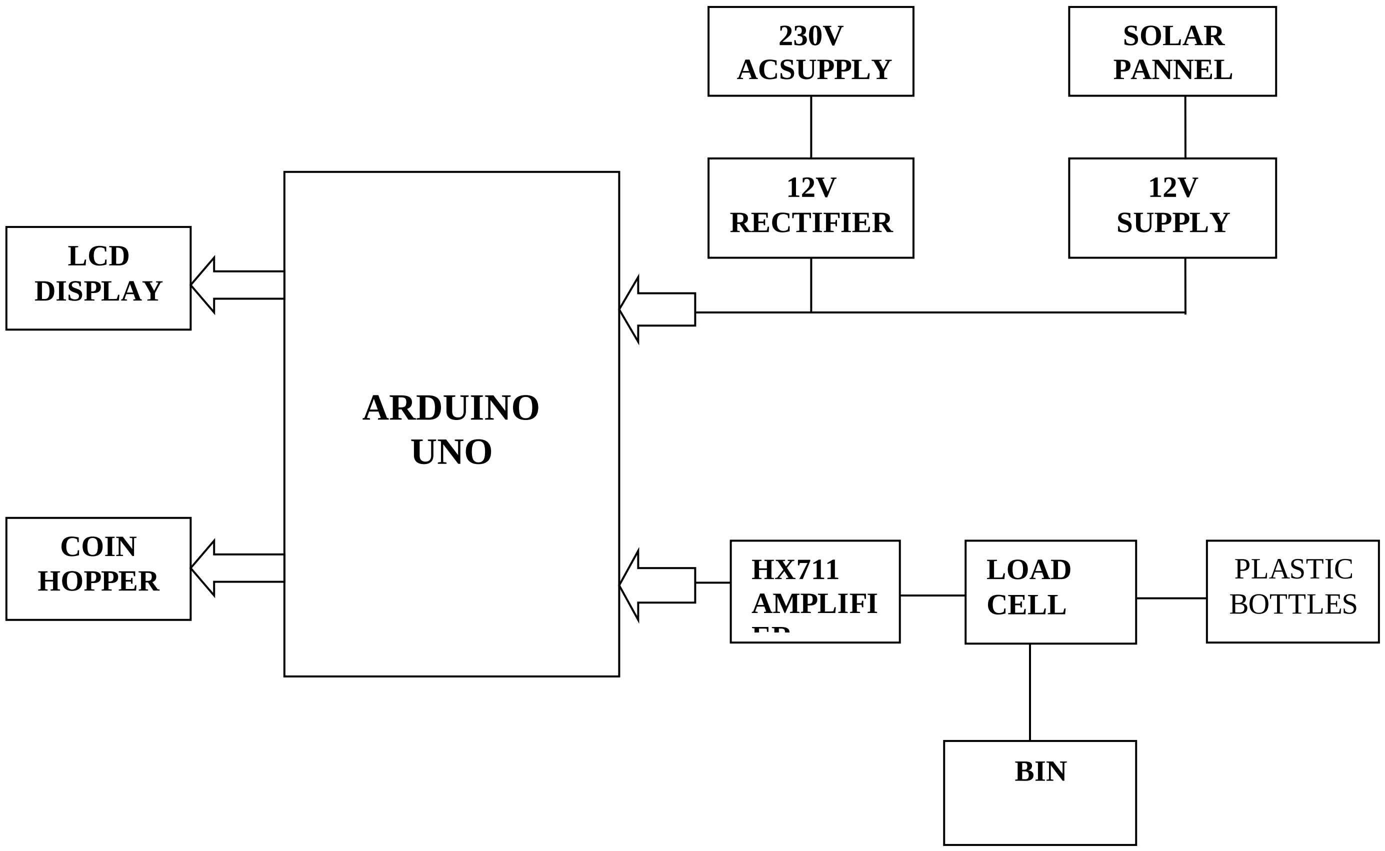


Fig: 4.1 Block Diagram of Proposed System

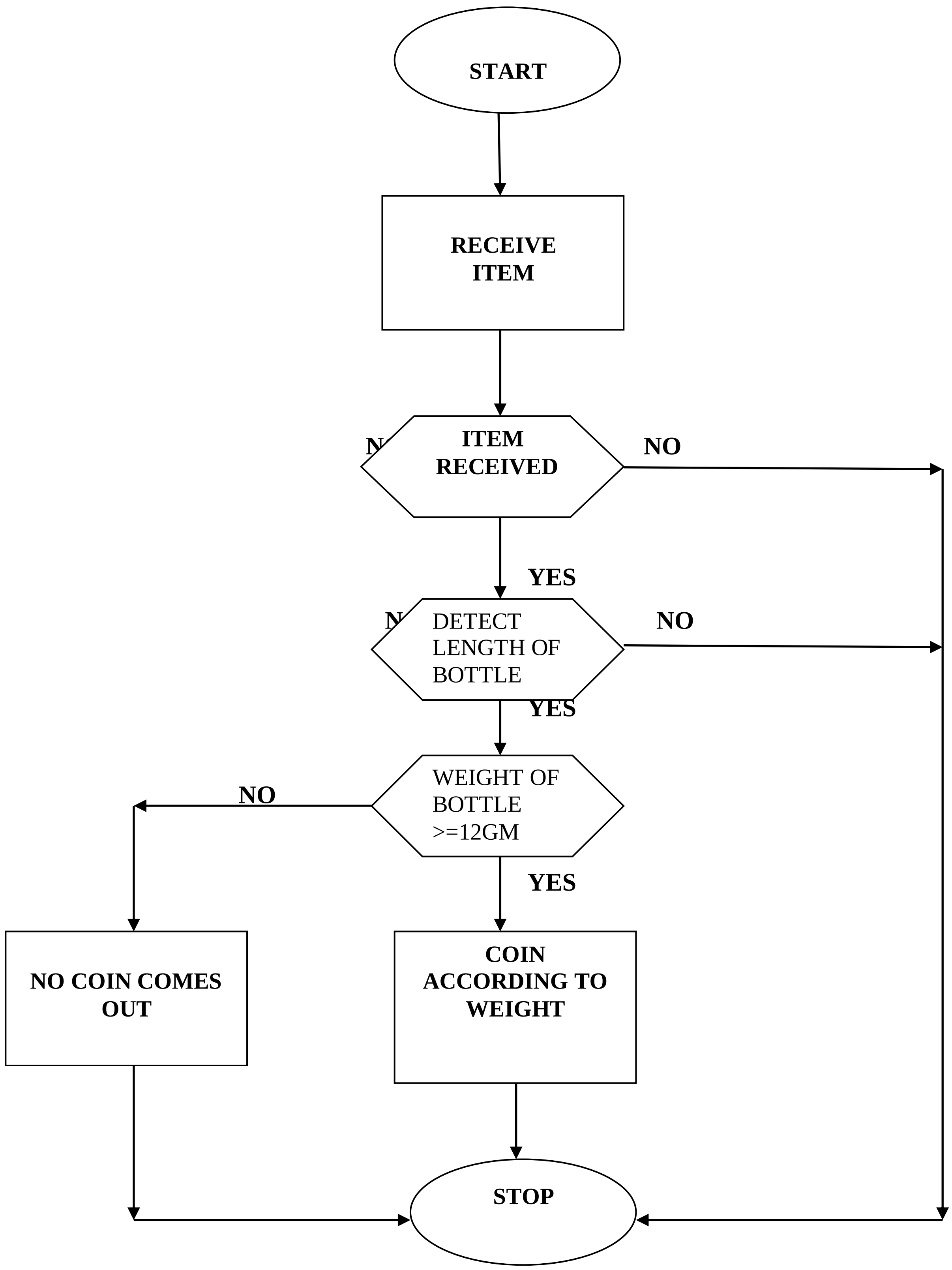


Fig: 4.2 Flow chart of proposed system

CHAPTER-5

5. HARDWARE DESCRIPTION

5.1.1 Introduction:

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

5.1.2 Arduino Uno Microcontroller:

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

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

The Arduino Uno can be powered via the USB connection or with an external power supply. The powersource is selected automatically.External (non-USB) power can come either from an AC-to-DC adapter (wall-wart) or battery. The adaptercan be connected by plugging a 2.1mm center-positive plug into the board's power jack. Leads from abattery can be inserted in the Gnd and Vin pin headers of the POWER connector.The board can operate on an external supply of 6 to 20 volts. If supplied with less than 7V, however, the 5Vpin may supply less than five volts and the board may be unstable. If using more than 12V, the voltage regulator may overheat and damage the board. The recommended range is 7 to 12 volts

The power pins are as follows:·

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

5.1.3 Memory:

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

5.1.4 Input and Output:

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

- **Serial: 0 (RX) and 1 (TX).** Used to receive (RX) and transmit (TX) TTL serial data. These pins are connected to the corresponding pins of the ATmega8U2 USB-to-TTL Serial chip .
- **External Interrupts: 2 and 3.** These pins can be configured to trigger an interrupt on a low value, arising or falling edge, or a change in value. See the attach Interrupt() function for details.
- **PWM: 3, 5, 6, 9, 10, and 11.** Provide 8-bit PWM output with the analogWrite() function.

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

The Uno has 6 analog inputs, each of which provide 10 bits of resolution (i.e. 1024 different values). Bydefault they measure from ground to 5 volts, though is it possible to change the upper end of their rangeusing the AREF pin and the analogReference() function. Additionally, some pins have specialized
functionality:

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

There are a couple of other pins on the board:

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

5.1.5 Communication:

The Arduino Uno has a number of facilities for communicating with a computer, another Arduino, or other microcontrollers. The ATmega328 provides UART TTL (5V) serial communication, which is available on digital pins 0 (RX) and 1 (TX). An ATmega8U2 on the board channels this serial communication over USB and appears as a virtual com port to software on the computer. The '8U2 firmware uses the standard USBCOM drivers, and no external driver is needed. However, on Windows, an *.inf file is required. The Arduino software includes a serial monitor which allows simple textual data to be sent to and from the Arduino board. The RX and TX LEDs on the board will flash when data is being transmitted via the USB-toserial chip and USB connection to the computer (but not for serial communication on pins 0 and 1). A SoftwareSerial library allows for serial communication on any of the Uno's digital pins. The ATmega328 also support I2C (TWI) and SPI communication. The Arduino software includes a Wire library to simplify use of the I2C bus

5.1.6 ARDUINO UNO BOARD:

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

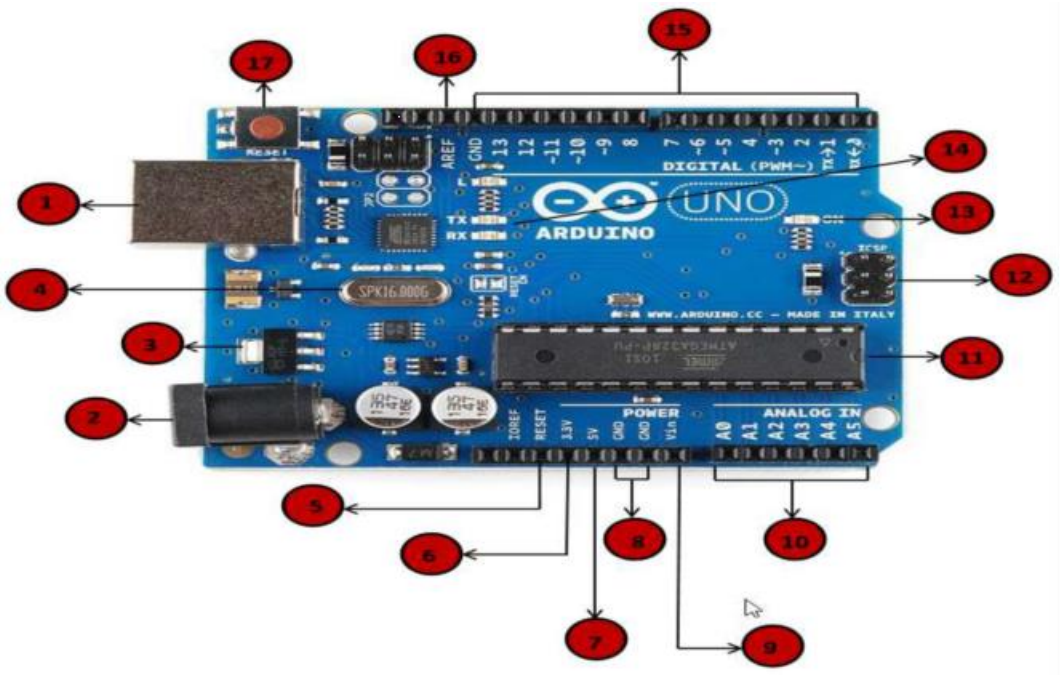
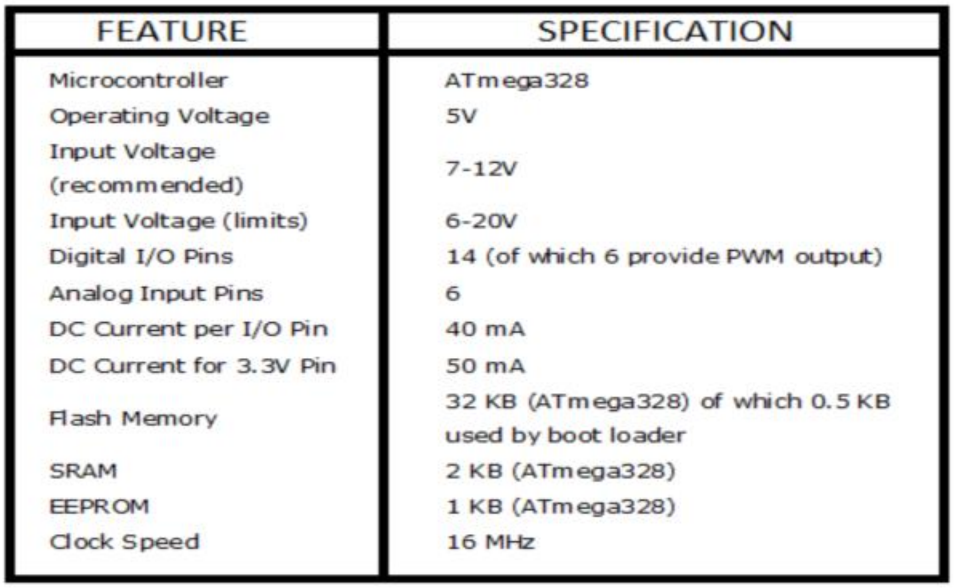


Figure 5.1: Arduino uno board

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

5.1.6.1 Technical Specifications:

Table 5.1: Arduino Uno specifications



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

The Arduino Uno can be powered via the USB connection or with an external power supply. The power source is selected automatically. External (non-USB) power can come either from an AC-to-DC adapter (wall-wart) or battery. The adapter can be connected by plugging a 2.1mm center-positive plug into the board's power jack. Leads from a battery can be inserted in the Gnd and Vin pin headers of the POWER connector. The board can operate on an external supply of 6 to 20 volts. If supplied with less than 7V, however, the 5V pin may supply less than five volts and the board may be unstable. If using more than 12V, the voltage regulator may overheat and damage the board. The recommended range is 7 to 12 volts.

1.USB Interface:

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

2.External power supply:

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

3.Voltage Regulator:

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

4.Crystal Oscillator:

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

5,17.Arduino Reset:

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

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

- 3.3V (6): Supply 3.3 output volt
- 5V (7): Supply 5 output volt
- Most of the components used with Arduino board works fine with 3.3 volt and 5 volt.
- GND (8)(Ground): There are several GND pins on the Arduino, any of which can be used to ground your circuit.
- Vin (9): This pin also can be used to power the Arduino board from an external power source, like AC mains power supply.

10.Analog pins:

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

11.Main microcontroller:

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

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

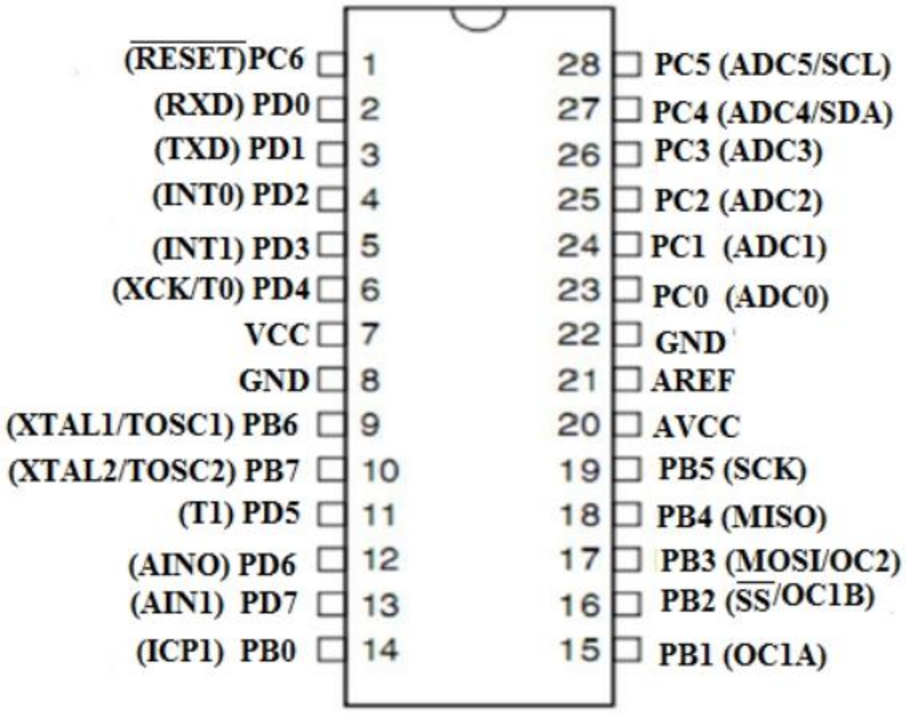


Figure 5.2: Pin diagram

5.1.6.2 Pin Description:

VCC: Digital supply voltage.

GND: Ground.

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

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

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

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

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

Port C (PC[5:0]):

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

PC6/RESET:

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

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

Port D (PD [7:0]):

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

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

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

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

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

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

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

15. Digital I / O: The Arduino UNO board has 14 digital I/O pins (15) (of which 6 provide PWM (Pulse Width Modulation) output. These pins can be configured to work as input digital pins to read logic values (0 or 1) or as digital output pins to drive different modules like LEDs, relays, etc. The pins labeled “~” can be used to generate PWM.

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

5.2 RFID Reader (RC522)

The RC522 RFID reader uses radio frequency to read data stored in RFID cards. In this project, it's used for user identification—when a user scans their card, the system tracks the recycling activity and rewards them. It operates at 13.56 MHz and communicates with Arduino via SPI. It supports ISO/IEC 14443 Type A cards and provides fast, contactless communication.

Fig 5.3:RFID

5.2.1 Use of RFID Reader (RC522) in RVM

The **RFID Reader (RC522)** is used for **user identification and reward tracking** in the RVM. Each user is assigned an RFID card or tag, which they scan before or after depositing plastic waste. The RC522 reads the unique ID of the card and sends it to the Arduino. This allows the system to associate the deposited waste with a specific user and update their reward points accordingly. It ensures secure and contactless interaction and enables user-specific data collection, like the number of items recycled and incentives earned.

5.2.3 RFID Reader (RC522) – Specifications Table

Table 5.2:RFID Specifications

Parameter	**Specification**
Operating Voltage	3.3V
Operating Current	13–26 mA
Communication Protocol	SPI (Supports I2C and UART with modifications)
Frequency	13.56 MHz
Compatible Cards	ISO/IEC 14443 Type A (e.g., MIFARE)
Maximum Data Rate	106 Kbps
Read Range	~3 cm to 5 cm
Antenna Size	~25 mm × 30 mm (built-in PCB antenna)

Operating Temperature	-20°C to +85°C
Dimensions	40 mm × 60 mm

5.3 LCD (Liquid Cristal Display):

5.3.1OVERVIEW

A liquid crystal display (LCD) is a thin, flat display device made up of any number of color or monochrome pixels arrayed in front of a light source or reflector. Each pixel consists of a column of liquid crystal molecules suspended between two transparent electrodes, and two polarizing filters, the axes of polarity of which are perpendicular to each other. Without the liquid crystals between them, light passing through one would be blocked by the other. The liquid crystal twists the polarization of light entering one filter to allow it to pass through the other.

A program must interact with the outside world using input and output devices that communicate directly with a human being. One of the most common devices attached to an controller is an LCD display. Some of the most common LCDs connected to the contollers are 16X1, 16x2 and 20x2 displays. This means 16 characters per line by 1 line 16 characters per line by 2 lines and 20 characters per line by 2 lines, respectively.

Shapes and S

available. Line lengths of 8, 16, 20, 24, 32 and 40 characters are all standard, in one, two

5.3.2Features:

- Interface with either 4-bit or 8-bit microprocessor.
- Display data RAM
- 80x 8 bits (80 characters).
- Character generator ROM
- 160 different 5 ´ 7 dot-matrix character patterns.
- Character generator RAM
- Different user programmed 5 ´ 7 dot-matrix patterns.
- Display data RAM and character generator RAM may be
- Accessed by the microprocessor.

- Numerous instructions
- Clear Display, Cursor Home, Display ON/OFF, Cursor ON/OFF,
- Blink Character, Cursor Shift, Display Shift.
- Built-in reset circuit is triggered at power ON.
- Built-in oscillator.

5.3.3Description Of 16x2:

This is the first interfacing example for the Parallel Port. We will start with something simple. This example doesn't use the Bi-directional feature found on newer ports, thus it should work with most, if noall Parallel Ports. It however doesn't show the use of the Status Port as an input. So what are we interfacing? A 16 Character x 2 Line LCD Module to the Parallel Port. These LCD Modules are very common these days, and are quite simple to work with, as all the logic required to run them is on board.

Schematic Diagram:

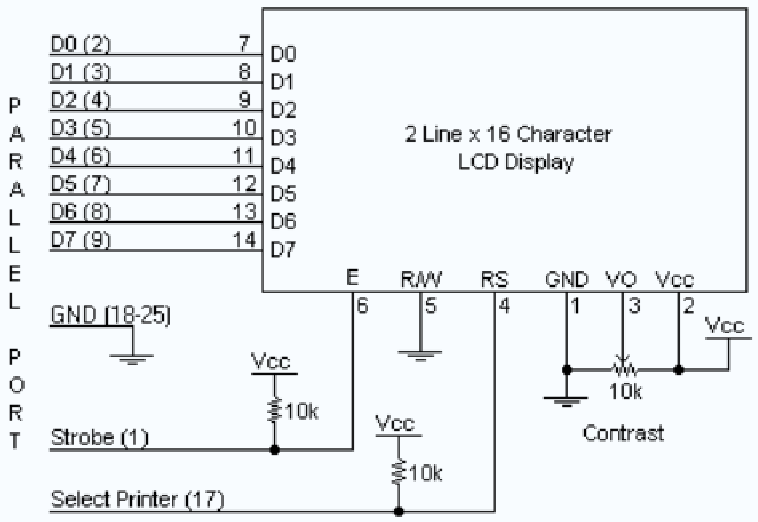


FIG 5.4: SCHEMATIC DIAGRAM

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

- We make no effort to place the Data bus into reverse direction. Therefore we hard wire the R/W line of the LCD panel, into write mode. This will cause no bus conflicts on the data lines. As a result we cannot read back the LCD's internal Busy Flag which tells us if the LCD has accepted and finished processing the last instruction. This problem is overcome by inserting known delays into our program.
- The 10k Potentiometer controls the contrast of the LCD panel. Nothing fancy here. As with all the examples, I've left the power supply out. You can use a bench power supply set to 5v or use a onboard +5 regulator. Remember a few de-coupling capacitors, especially if you have trouble with the circuit working properly.

- The 2 line x 16 character LCD modules are available from a wide range of manufacturers and should all be compatible with the HD44780. The one I used to test this circuit was a Power tip PC-1602F and an old Philips LTN211F-10 which was extracted from a Poker Machine! The diagram to the right, shows the pin numbers for these devices. When viewed from the front, the left pin is pin 14 and the right pin is pin 1

.

16 x 2 Alphanumeric LCD Module Features:

- Intelligent, with built-in Hitachi HD44780 compatible LCD controller and RAM providing simple interfacing
- 61 x 15.8 mm viewing area
- 5 x 7 dot matrix format for 2.96 x 5.56 mm characters, plus cursor line
- Can display 224 different symbols
- Low power consumption (1 mA typical)
- Powerful command set and user-produced characters

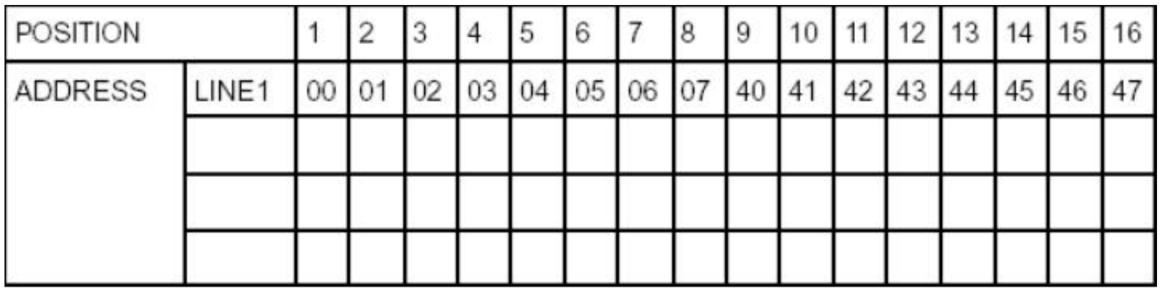
- TTL and CMOS compatible
- Connector for standard 0.1-pitch pin headers

FEATURES:

• 5 x 8 dots with cursor
• Built-in controller (KS 0066 or Equivalent)
• + 5V power supply (Also available for + 3V)
• 1/16 duty cycle
• B/L to be driven by pin 1, pin 2 or pin 15, pin 16 or A.K (LED)

• N.V. optional for + 3V power supply

Data can be placed at any location on the LCD. For 16×1 LCD, the address locations are:



POSITION		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
ADDRESS	LINE1	00	01	02	03	04	05	06	07	40	41	42	43	44	45	46	47

Fig: 5.5: Address locations for a 1x16 line LCD

Even limited to character based modules, there is still a wide variety of shapes and sizes available. Line lengths of 8, 16,20,24,32 and 40 characters are all standard, in one, two and four line versions.

Several different LC technologies exists. “supertwist” types, for example, offer Improved contrast and viewing angle over the older “twisted nematic” types. Some modules are available with back lighting, so that they can be viewed in dimly-lit conditions. The back lighting may be either “electro-luminescent”, requiring a high voltage inverter circuit, or simple LED illumination.

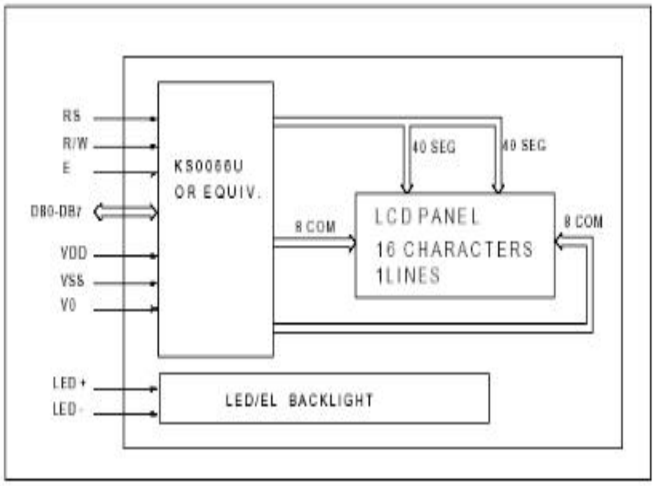


Fig: 5.6: Electrical Block Diagram

Power supply for LCD driving:

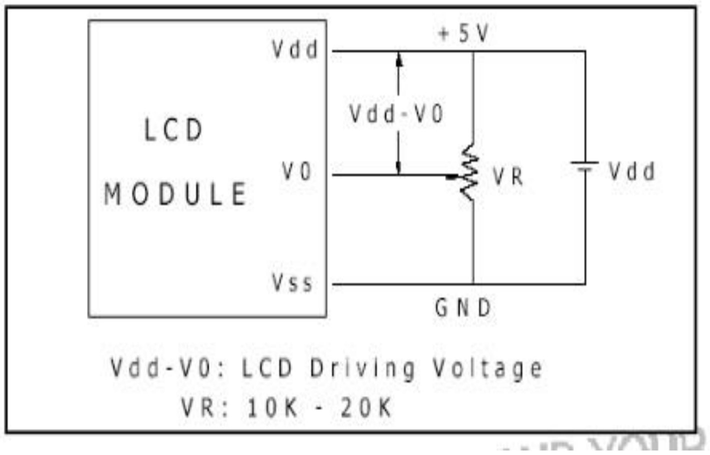


Fig: 5.7: power supply for LCD

5.3.4PIN DESCRIPTION:

Most LCDs with 1 controller has 14 Pins and LCDs with 2 controller has 16 Pins (two pins are extra in both for back-light LED connections).

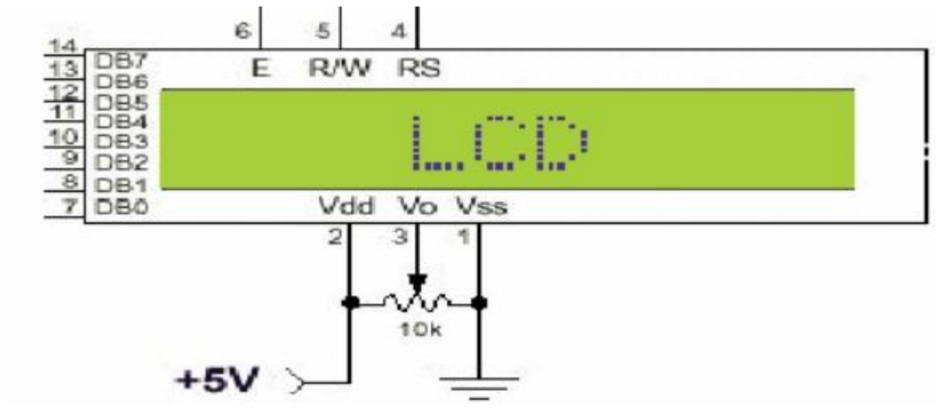
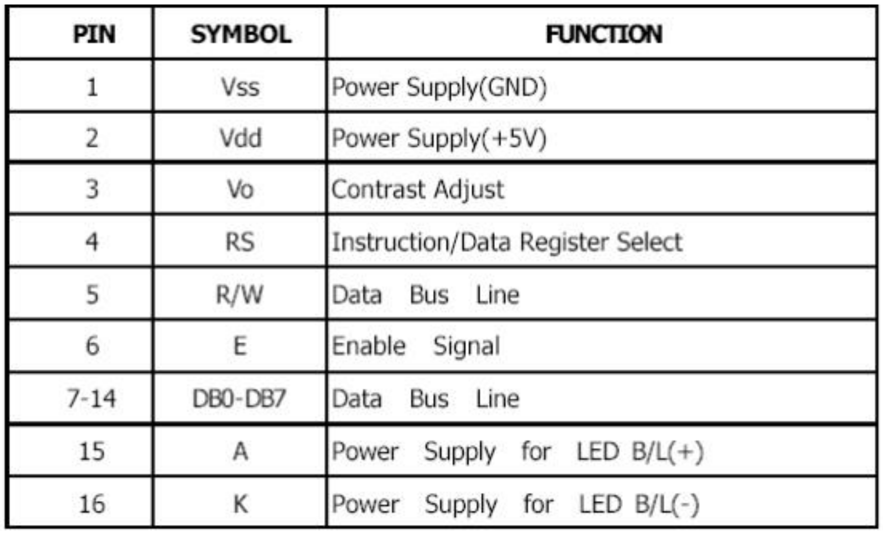


Fig 5.8: pin diagram of 1x16 lines LCD

Table 5.3: Pin Specifications



PIN	SYMBOL	FUNCTION
1	Vss	Power Supply(GND)
2	Vdd	Power Supply(+5V)
3	Vo	Contrast Adjust
4	RS	Instruction/Data Register Select
5	R/W	Data Bus Line
6	E	Enable Signal
7-14	DB0-DB7	Data Bus Line
15	A	Power Supply for LED B/L(+)
16	K	Power Supply for LED B/L(-)

5.4IR Sensor – Overview

An **Infrared (IR) Sensor** is an electronic device that detects infrared radiation (IR light) from objects in its field of view. In the context of the RVM, it's used to **detect the presence of plastic bottles** when they are inserted into the machine.

Fig: 5.9 IR SENSOR

5.4.1Technical Specifications (Typical IR Sensor Module)

Table 5.4: IR Specifications

Parameter	**Value**
Operating Voltage	3.3V – 5V DC
Detection Range	2 cm – 30 cm (adjustable)
Sensor Type	Active IR Sensor (Transmitter + Receiver)
Output Type	Digital (HIGH/LOW)
Wavelength	~940 nm (Infrared light)
Response Time	< 1 millisecond
Operating Temperature	-10°C to 50°C
Power Consumption	Low (depends on module, usually <50mA)

5.4.2Working Principle

- The IR sensor module has **two parts**:
  - **IR LED (Transmitter)**: Emits infrared light.
  - **Photodiode (Receiver)**: Detects the reflected IR light.
- When an object (like a plastic bottle) is placed in front of the sensor, the **infrared light reflects** back to the photodiode.
- The sensor converts this reflection into an electrical signal to notify the system about the object’s presence.

5.4.3 Function in RVM

- **Bottle Detection**: When a user inserts a bottle, the IR sensor detects its presence and triggers the rest of the process (AI recognition, sorting, reward, etc.).
- **Automation Trigger**: Prevents false operations by ensuring an item is physically present before running classification or sorting algorithms.

5.4.4 Advantages of Using IR Sensors in RVMs

- Non-contact detection
- Fast response time
- Low power consumption
- Compact and cost-effective
- Reliable for presence detection

5.5 Ultrasonic Sensor – Overview

An **Ultrasonic Sensor** is a non-contact distance measurement device that uses ultrasonic waves to detect objects. In the RVM project, it is used to:

- **Measure the height or size** of inserted bottles.
- **Ensure proper placement** of the bottle inside the chamber.
- **Trigger sorting or compacting** mechanisms once a bottle is fully inserted.

5.5.1 Technical Specifications (HC-SR04 Ultrasonic Sensor – Common Model)

Table 5.5.: Technical Specifications of HC-SR04

Parameter	**Value**
Operating Voltage	5V DC
Operating Current	15 mA
Frequency	40 kHz
Measuring Range	2 cm – 400 cm (0.02 m to 4 m)
Accuracy	±3 mm
Measuring Angle	< 15°
Interface	4-pin (VCC, Trig, Echo, GND)
Response Time	< 50 ms

5.5.2 Working Principle

- The sensor has **two transducers**:
  - **Trigger (TX)**: Emits an ultrasonic pulse.
  - **Echo (RX)**: Receives the reflected wave.
- It measures the **time interval** between sending and receiving the signal.

5.5.3 Function in RVM

- **Bottle Size Detection**: Ensures only valid items (not trash or too-small objects) are inserted.
- **Position Monitoring**: Confirms that the bottle has reached the right place before image recognition or compaction starts.
- **Bin Level Monitoring**: Can also check how full the storage bin is.

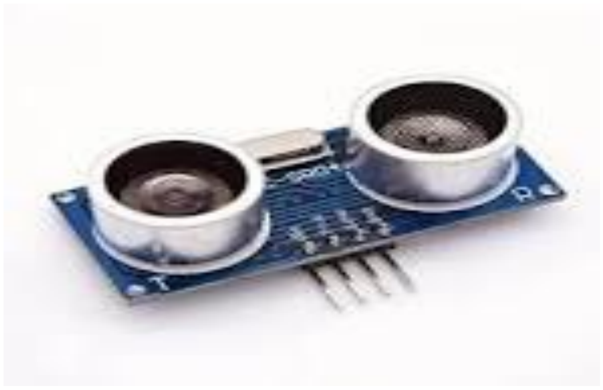


Fig.5.10Ultrasonic Sensor

5.5.4 Advantages of Using Ultrasonic Sensors in RVMs

- **High accuracy** in object detection.
- **Non-contact measurement**, ideal for hygienic applications.
- **Wide range** of detection (up to 4 meters).

- **Reliable in various lighting** and environmental conditions (unlike IR sensors which struggle in bright light).

5.6 Power supply-solar energy

Solar energy is used as the **primary or supplementary power source** for the RVM system. A **solar panel** mounted on or near the machine captures sunlight and converts it into electrical energy. This energy is then stored in a **rechargeable battery** (like a 12V lead-acid or lithium-ion battery) through a **solar charge controller**, which regulates charging and prevents overcharging or deep discharge.

The stored energy powers various components of the RVM, such as:

- Arduino microcontroller
- Sensors (IR, RFID, ultrasonic, load cell)
- Camera module
- Motors (servo/DC)
- LCD display and buzzer
- Raspberry Pi (if low-power mode is enabled)

Using solar power allows the RVM to operate **independently of the electrical grid**, making it suitable for deployment in public parks, rural areas, and urban zones with limited infrastructure.

Fig 5.11: solar panel

5.6.1 Benefits of Using Solar Energy

- Promotes renewable and green energy usage
- Reduces operational cost and carbon footprint
- Ensures uninterrupted operation in case of power outages
- Increases deployment flexibility (portable & off-grid usage)

5.6.2 Solar Power Setup Components

Table 5.6: Solar Specifications

Component	**Description**
Solar Panel	Converts sunlight to DC electricity (e.g., 20W–50W poly/monocrystalline panel)
Solar Charge Controller	Regulates voltage/current to protect battery (e.g., 12V 10A PWM/MPPT controller)
Battery	Stores solar energy for use (e.g., 12V 7Ah lead-acid or Li-ion battery)
DC-DC Converter	Steps down 12V battery output to 5V for Arduino, sensors, Raspberry Pi

5.7 Servo Motor – Overview

A servo motor **is a compact, rotary actuator that allows for** precise control of angular position**, velocity, and acceleration. It consists of a** DC motor**, a** gearing system**, a** position sensor **(usually a potentiometer), and a** control circuit**. The motor receives control signals in the form of** PWM (Pulse Width Modulation) **from a microcontroller, which determines the angle of rotation.**

Servo motors are typically used in applications where **accurate and limited rotation** (usually from 0° to 180°) is required. Due to their **precision, reliability, and ease of control**, servo motors are widely used in robotics, automation, RC toys, and embedded systems.

Fig: 5.12: Servo Motor

5.7.1 Key Features:

- Controlled by PWM signal
- High torque in a small package
- Simple 3-wire connection (Signal, VCC, GND)
- Rotates to exact angle based on input signal
- Often limited to 180° rotation

5.7.2 Use of Servo Motors in RVM

In the SMART RVM, **servo motors** are used to **control mechanical movements** such as:

1. **Opening and closing flaps or gates** for directing plastic bottles into sorting bins based on material type.
2. **Activating a locking mechanism** that opens only after a bottle is identified and accepted.
3. **Rotating compartments** or trays for categorized storage of plastic items.

Servo motors provide **precise angular control**, which is essential for these automated mechanical actions. The Arduino controls the servo using **PWM (Pulse Width Modulation)** signals to set the motor's angle between **0° and 180°**

.

5.7.3Working of Servo Motor in the Project

- When a plastic bottle is identified, the system sends a command to the **servo motor** to rotate to a specific angle (e.g., open the flap to the PET bin).
- After the item is deposited, the servo returns to its original position (closing the gate or tray).
- The motor's quick and accurate rotation ensures **efficient sorting** without jamming or misplacement.

Servo motors are ideal due to their:

- **Compact size**
- **Low power consumption**
- **Built-in position control**

5.7.4 Specifications of SG90 Micro Servo Motor

Table 5.7 SG90 SPECIFICATIONS

Parameter	**Specification**
Operating Voltage	4.8V to 6.0V
Stall Torque	1.8 kg·cm @ 4.8V
Operating Speed	0.1 s/60° @ 4.8V
Control Signal	PWM (Pulse Width Modulation)
Rotation Range	0° to 180°
Weight	9g
Dimensions (L×W×H)	23 × 12.2 × 29 mm
Gears	Plastic (lightweight, low-cost)
Connector Type	3-pin (Signal, VCC, GND)

5.8: Load Cell + HX711

5.8.1: Description

A Load Cell is a transducer that converts mechanical force (weight or load) into an electrical signal. In this RVM, it is used to detect the weight of the inserted plastic bottle. Plastic bottles vary in size and weight, and the system uses this weight to determine the number of reward coins to dispense. Since the electrical signal from a load cell is very weak (in the millivolt range), it needs to be amplified before being read by a microcontroller like Arduino.

This is where the HX711 comes in — it's a precision 24-bit analog-to-digital converter (ADC) specifically designed for weighing scales. It amplifies and converts the small signal from the load cell into a readable digital value, which can then be processed by the Arduino.

5.8.2:Working

When a plastic bottle is placed on the load cell:

- The metal strain gauges inside the load cell deform slightly due to the applied force.
- This deformation causes a change in electrical resistance, which translates into a small analog voltage.
- The HX711 module amplifies this voltage signal and converts it to a digital value.
- The Arduino reads the digital output from HX711 and calculates the actual weight.
- Based on the weight, the system triggers the coin dispenser and records the bottle.

5.8.3: Specifications

Table 5.8: Load cell Specifications

Specification	**Details**
Type	Strain Gauge Load Cell (Half or Full Bridge)
Capacity	1kg, 5kg, 10kg, 20kg, 50kg (varies by model)
Output Sensitivity	1.0 ± 0.1 mV/V (common value)
Excitation Voltage	5V DC recommended
Rated Output	Typically 1-2 mV/V
Non-linearity	±0.05% of full scale
Repeatability	±0.03% of full scale
Creep	±0.05% of full scale (in 30 min)
Material	Aluminum Alloy or Stainless Steel
Operating Temperature	-10°C to +40°C
Wiring	4-wire (Red: Excitation+, Black: Excitation-, Green: Signal+, White: Signal-)

Table 5.9:HX711specifications

Specification	**Details**
Input Channels	2 differential channels (A & B)
Amplifier Gain	32, 64, and 128 (default: 128 for channel A)
Operating Voltage	2.7V – 5.5V DC
Current Consumption	~1.5 mA (normal mode), <1µA (power down)
Resolution	24-bit ADC
Interface	Serial (Clock and Data pins)
Output Data Rate	10 Hz or 80 Hz
Common Interface Pins	VCC, GND, DT (Data), SCK (Clock)
Dimensions	~20mm x 15mm (varies by board layout)

5.8.4: Why It Is Used in This Project

- Precise Weight Measurement: Determines bottle weight accurately to issue the correct reward.
- Low Cost: Both components are affordable and easy to use with Arduino.
- Compact: Fits easily into small enclosures of RVM systems.
- Reliable for Repeated Use: Can handle multiple bottle placements without degradation.
- Essential for Logic: Enables the system to decide how many coins/tokens to

dispense

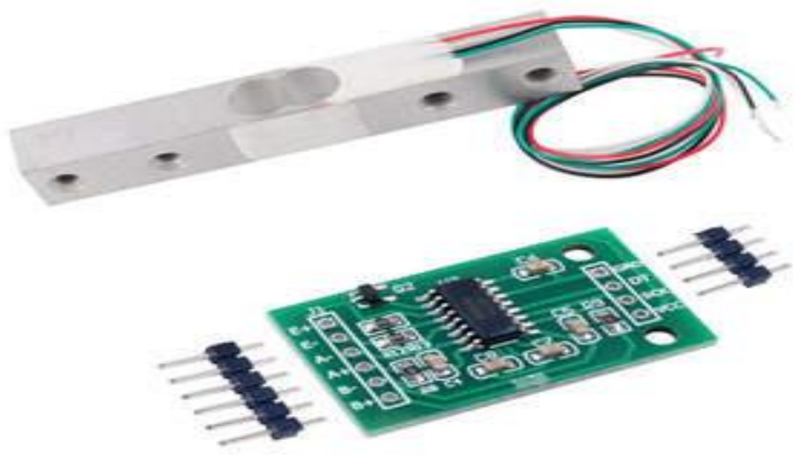


Fig: 5.13: Load cell and Hx711

5.9: Node MCU (ESP8266) – Wi-Fi Enabled Microcontroller

5.9.1:Description:

Node MCU is an open-source IoT development board based on the ESP8266 Wi-Fi SoC (System on Chip). It integrates a microcontroller with built-in Wi-Fi, making it ideal for IoT applications like the SMART Reverse Vending Machine. It supports the Lua scripting language and Arduino IDE, allowing easy programming and cloud connectivity. Node MCU features digital and analog I/O pins, USB-to-serial communication, and operates at 3.3V. Its compact design, low cost, and integrated Wi-Fi capability make it perfect for transmitting sensor data or user activity to remote servers, enabling real-time monitoring, data logging, or reward systems in recycling automation projects.

5.9.2: Specifications of Node MCU (ESP8266-based)

Table 5.10: NODE MCU Specifications

Specification	**Details**
Microcontroller	ESP8266 (ESP-12E module)
Operating Voltage	3.3V (regulated via onboard voltage regulator)
Input Voltage (USB)	5V via micro-USB
Digital I/O Pins	11 (GPIO0 to GPIO16, some shared with UART)
Analog Input	1 (10-bit ADC)
Clock Speed	80 MHz (default), up to 160 MHz
Flash Memory	4 MB (depends on variant)
SRAM	64 KB (available for use)
Wi-Fi	802.11 b/g/n, WPA/WPA2 support
Communication	UART, SPI, I2C, PWM, ADC
Programming Interface	USB to Serial using CP2102/CH340
IDE Support	Arduino IDE, Lua, PlatformIO
Dimensions	Approx. 58mm x 31mm

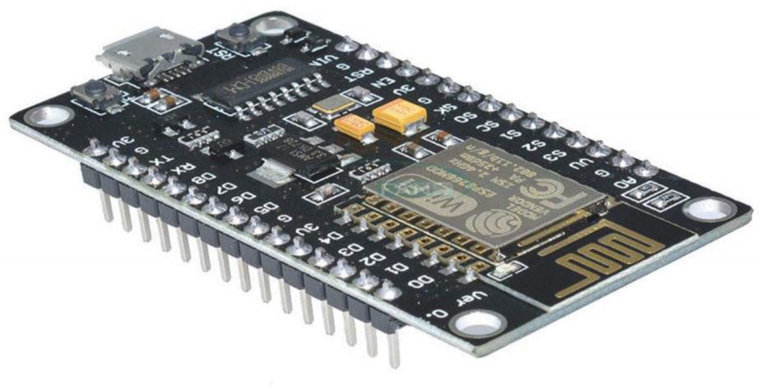


Fig: 5.14: Node MCU

5.9.3 Use of Node MCU in the SMART RVM Project

Node MCU is used in this project primarily because of its **built-in Wi-Fi capabilities**, which make it ideal for **IoT (Internet of Things)** applications. In the SMART RVM, it plays a crucial role in **sending data to the cloud or a remote server**, such as:

- **Transmitting user activity** (e.g., how many bottles were deposited),
- **Uploading weight data** from the **load cell + HX711 module**,
- **Communicating with a database or reward system** (e.g., updating user points),
- **Remote monitoring and control** of the RVM status.

In short, **NodeMCU acts as a bridge between the hardware sensors and the internet**, enabling features like **real-time data logging**, **app-based notifications**, or **web-based dashboards**.

5.9.4: Summary

Table 5.11: summary of NODE MCU

Reason	**Explanation**
Built-in Wi-Fi	Enables IoT-based features and cloud access
Easy Data Transfer	Sends data from sensors to the cloud or server
Compact and Low Power	Fits well inside smart machines with limited space
Supports Arduino IDE	Easily programmable like an Arduino board
Rich GPIO Interface	Connects with sensors like load cell, IR, camera, etc.

5.10 Software and Coding:

5.10.1 Introduction to Arduino IDE

Arduino is a prototype platform (open-source) based on an easy-to-use hardware and software. It consists of a circuit board, which can be programmed (referred to as a microcontroller) and a ready-made software called Arduino IDE (Integrated Development Environment), which is used to write and upload the computer code to the physical board.

The key features are:

- Arduino boards are able to read analog or digital input signals from different sensors and turn it into an output such as activating a motor, turning LED on/off, connect to the cloud and many other actions.
- You can control your board functions by sending a set of instructions to the microcontroller on the board via Arduino IDE (referred to as uploading software).

- Unlike most previous programmable circuit boards, Arduino does not need an extra piece of hardware (called a programmer) in order to load a new code onto the board. You can simply use a USB cable.
- Additionally, the Arduino IDE uses a simplified version of C++, making it easier to learn to program.
- Finally, Arduino provides a standard form factor that breaks the functions of the micro-controller into a more accessible package.

After learning about the main parts of the Arduino UNO board, we are ready to learn howto set up the Arduino IDE. Once we learn this, we will be ready to upload our program onthe Arduino board.

5.10.2 Arduino data types:

Data types in C refer to an extensive system used for declaring variables or functions of different types. The type of a variable determines how much space it occupies in the storage and how the bit pattern stored is interpreted.

The following table provides all the data types that you will use during Arduinoprogramming.

Void:

The void keyword is used only in function declarations. It indicates that the function is expected to return no information to the function from which it was called.

Example:

```
Void Loop ( )
```

```
{
// rest of the code
}
```

Boolean:

A Boolean holds one of two values, true or false. Each Boolean variable occupies one byte of memory.

Example:

Boolean state= false ; // declaration of variable with type boolean and initialize it with false.

Boolean state = true ; // declaration of variable with type boolean and initialize it with false.

Char:A data type that takes up one byte of memory that stores a character value. Character literals are written in single quotes like this: 'A' and for multiple characters, strings use double quotes: "ABC".

However, characters are stored as numbers. You can see the specific encoding in the ASCII chart. This means that it is possible to do arithmetic operations on characters, in whichthe ASCII value of the character is used. For example, 'A' + 1 has the value 66, since theASCII value of the capital letter A is 65.

Example:

Char chr_a = ‘a’ ;//declaration of variable with type char and initialize it with character a.

Char chr_c = 97 ;//declaration of variable with type char and initialize it with character 97

Unsigned char:

Unsigned char is an unsigned data type that occupies one byte of memory. The unsigned char data type encodes numbers from 0 to 255.

Example:

Unsigned Char chr_y = 121 ; // declaration of variable with type Unsigned char and initialize it with character y

Byte:

A byte stores an 8-bit unsigned number, from 0 to 255.

Example:

byte m = 25 ;//declaration of variable with type byte and initialize it with 25

int:

Integers are the primary data-type for number storage. **int** stores a 16-bit (2-byte) value. This yields a range of -32,768 to 32,767 (minimum value of -2^15 and a maximum value of (2^15) - 1).

The **int** size varies from board to board. On the Arduino Due, for example, an **int** stores a 32-bit (4-byte) value. This yields a range of -2,147,483,648 to 2,147,483,647 (minimum value of -2^31 and a maximum value of (2^31) - 1).

Example:

int counter = 32 ;// declaration of variable with type int and initialize it with 32.

Unsigned int:

Unsigned ints (unsigned integers) are the same as int in the way that they store a 2 byte value. Instead of storing negative numbers, however, they only store positive values, yielding a useful range of 0 to 65,535 (2^16) - 1). The Due stores a 4 byte (32-bit) value, ranging from 0 to 4,294,967,295 (2^32 - 1).

Example:

Unsigned int counter= 60 ; // declaration of variable with type unsigned int and initialize it with 60.

Word:

On the Uno and other ATMEGA based boards, a word stores a 16-bit unsigned number. On the Due and Zero, it stores a 32-bit unsigned number.

Example

word w = 1000 ;//declaration of variable with type word and initialize it with 1000.

Long:

Long variables are extended size variables for number storage, and store 32 bits (4 bytes), from 2,147,483,648 to 2,147,483,647.

Example:

Long velocity= 102346 ;//declaration of variable with type Long and initialize it with 102346

Unsigned long: Unsigned long variables are extended size variables for number storage and store 32 bits (4 bytes). Unlike standard longs, unsigned longs will not store negative numbers, making their range from 0 to 4,294,967,295 (2^32 - 1).

Example:

Unsigned Long velocity = 101006 ;// declaration of variable with type Unsigned Long and initialize it with 101006.

Short:

A short is a 16-bit data-type. On all Arduinos (ATMega and ARM based), a short stores a 16-bit (2-byte) value. This yields a range of -32,768 to 32,767 (minimum value of -2^15 and a maximum value of (2^15) - 1).

Example:

short val= 13 ;//declaration of variable with type short and initialize it with 13

Float:

Data type for floating-point number is a number that has a decimal point. Floating-point numbers are often used to approximate the analog and continuous values because they have greater resolution than integers.

Floating-point numbers can be as large as 3.4028235E+38 and as low as 3.4028235E+38. They are stored as 32 bits (4 bytes) of information.

Example:

float num = 1.352;//declaration of variable with type float and initialize it with 1.352.

Double:

On the Uno and other ATMEGA based boards, Double precision floating-point number occupies four bytes. That is, the double implementation is exactly the same as the float, with no gain in precision. On the Arduino Due, doubles have 8-byte (64 bit) precision.

Example:

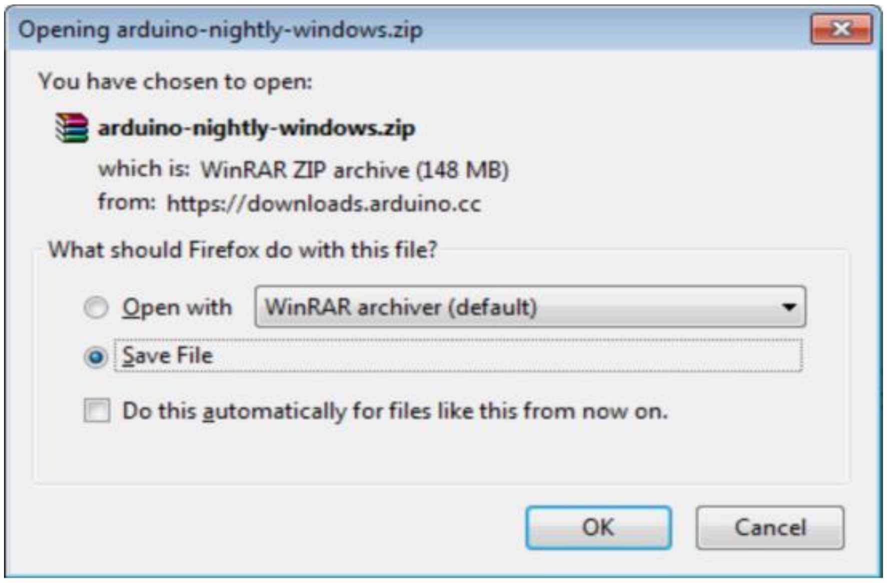
double num = 45.352 ;// declaration of variable with type double and initialize it with 45.352.

In this section, we will learn in easy steps, how to set up the Arduino IDE on our computerand prepare the board to receive the program via USB cable.

Step 1: First you must have your Arduino board (you can choose your favorite board) anda USB cable. In case you use Arduino UNO, Arduino Duemilanove, Nano, Arduino Mega2560, or Diecimila, you will need a standard USB cable (A plug to B plug), the kind youwould connect to a USB printer as shown in the following image.

Step 2: Download Arduino IDE Software.

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


Opening arduino-nightly-windows.zip

You have chosen to open:

arduino-nightly-windows.zip

which is: WinRAR ZIP archive (148 MB)

from: https://downloads.arduino.cc

What should Firefox do with this file?

Open with WinRAR archiver (default)

Save File [✓]

Do this automatically for files like this from now on.

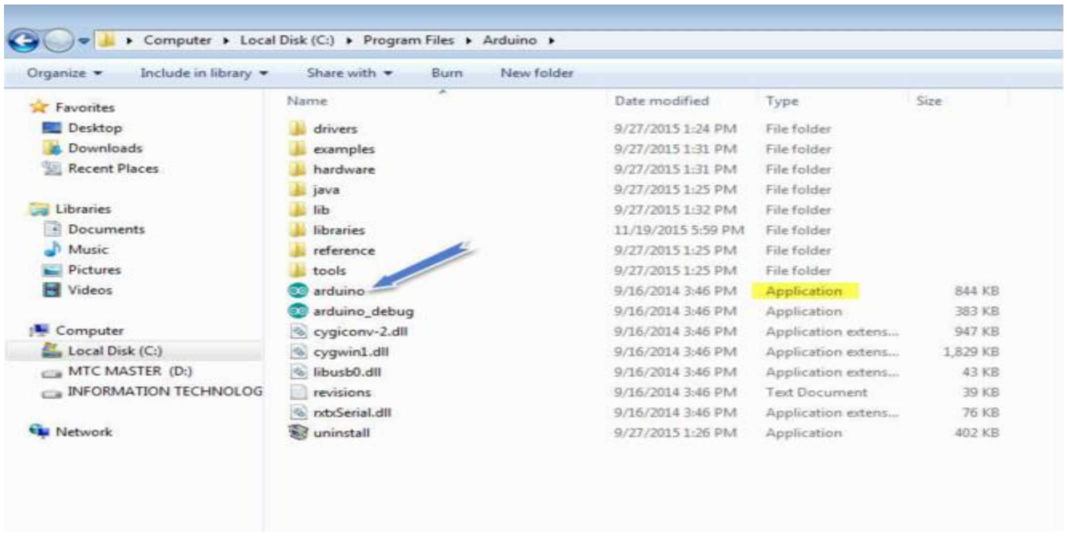
Fig.5.15:ArduinoDownload page

Step 3: Power up your board.

The Arduino Uno, Mega, Duemilanove and Arduino Nano automatically draw power from either, the USB connection to the computer or an external power supply. If you are using an Arduino Diecimila, you have to make sure that the board is configured to draw power from the USB connection. The power source is selected with a jumper, a small piece of plastic that fits onto two of the three pins between the USB and power jacks. Check that it is on the two pins closest to the USB port. Connect the Arduino board to your computer using the USB cable. The green power LED (labeled PWR) should glow.

Step 4: Launch Arduino IDE.

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



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

Fig.5.16: Launch Arduino IDE

Step 5: Open your first project.

Once the software starts, you have two options:

- Create a new project.
- Open an existing project example.

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

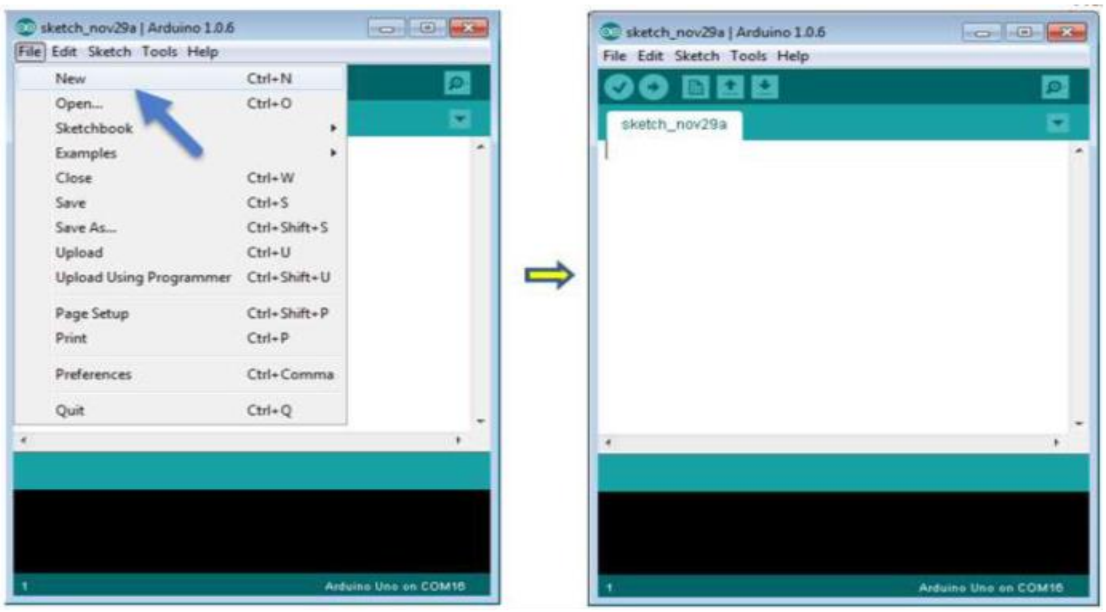
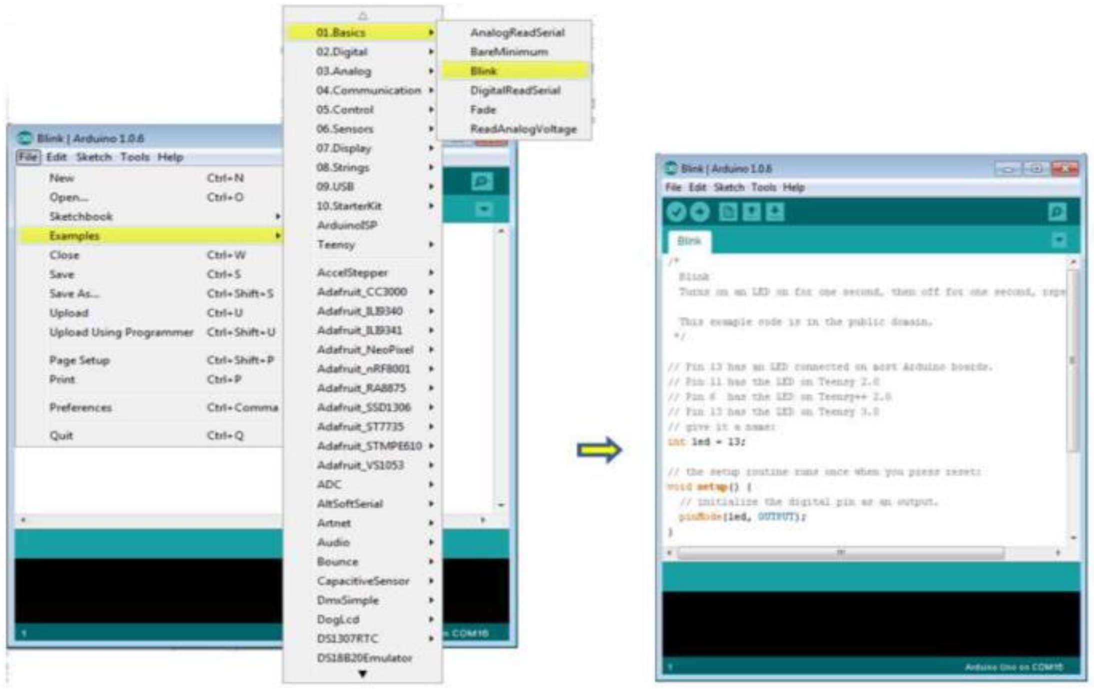


Fig.5.17:vCreating a New Sketch in Arduino IDE

To open an existing project example, select File -> Example -> Basics -> Blink.



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

  This example code is in the public domain.
 */

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

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

Fig.5.18:Loading the Blink Example in Arduino IDE

Here, we are selecting just one of the examples with the name **Blink**. It turns the LED on and off with some time delay. You can select any other example from the list.

Step 6: Select your Arduino board.

To avoid any error while uploading your program to the board, you must select the correct Arduino board name, which matches with the board connected to your computer. Go to Tools -> Board and select your board

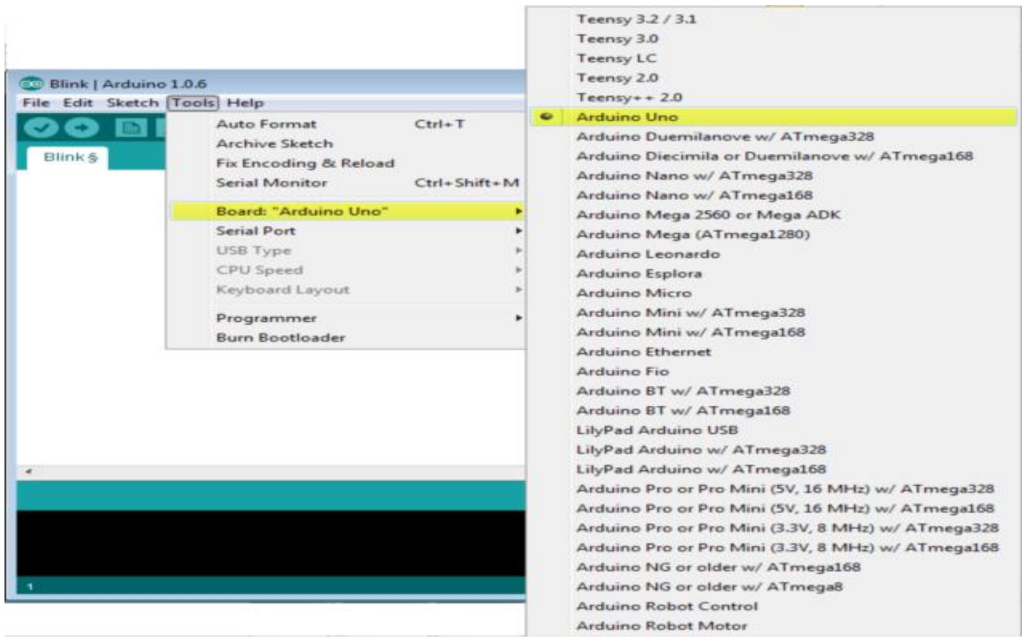

Blink | Arduino 1.0.6
File Edit Sketch Tools Help
Blink §
Auto Format Ctrl+T
Archive Sketch
Fix Encoding & Reload
Serial Monitor Ctrl+Shift+M
Board: "Arduino Uno"
Serial Port
USB Type
CPU Speed
Keyboard Layout
Programmer
Burn Bootloader
Teensy 3.2 / 3.1
Teensy 3.0
Teensy LC
Teensy 2.0
Teensy++ 2.0
Arduino Uno
Arduino Duemilanove w/ ATmega328
Arduino Diecimila or Duemilanove w/ ATmega168
Arduino Nano w/ ATmega328
Arduino Nano w/ ATmega168
Arduino Mega 2560 or Mega ADK
Arduino Mega (ATmega1280)
Arduino Leonardo
Arduino Esplora
Arduino Micro
Arduino Mini w/ ATmega328
Arduino Mini w/ ATmega168
Arduino Ethernet
Arduino Fio
Arduino BT w/ ATmega328
Arduino BT w/ ATmega168
LilyPad Arduino USB
LilyPad Arduino w/ ATmega328
LilyPad Arduino w/ ATmega168
Arduino Pro or Pro Mini (5V, 16 MHz) w/ ATmega328
Arduino Pro or Pro Mini (5V, 16 MHz) w/ ATmega168
Arduino Pro or Pro Mini (3.3V, 8 MHz) w/ ATmega328
Arduino Pro or Pro Mini (3.3V, 8 MHz) w/ ATmega168
Arduino NG or older w/ ATmega168
Arduino NG or older w/ ATmega8
Arduino Robot Control
Arduino Robot Motor

Fig.5.19 :Selecting the Board Type in Arduino IDE

Here, we have selected Arduino Uno board according to our tutorial, but you must select the name matching the board that you are using

Step 7: Select your serial port.

Select the serial device of the Arduino board. Go to **Tools** ->**Serial Port** menu. This is likely to be COM3 or higher (COM1 and COM2 are usually reserved for hardware serial ports). To find out, you can disconnect your Arduino board and re-open the menu, the entry that disappears should be of the Arduino board. Reconnect the board and select that serial port.

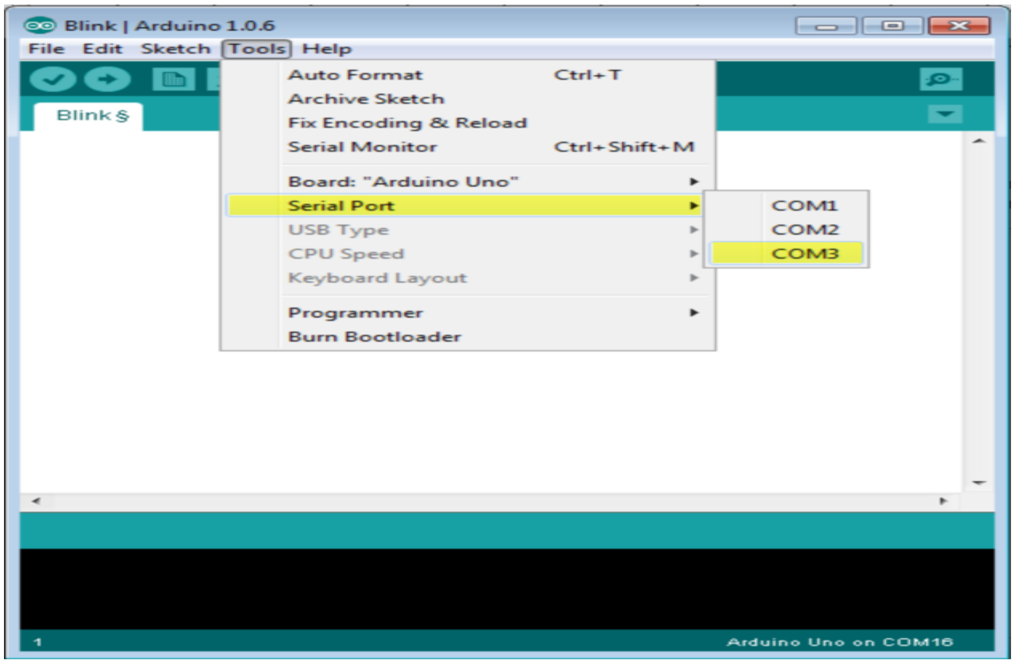


Fig: 5.20: Selecting the Correct Serial Port in Arduino IDE"

Step 8: Upload the program to your board.

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

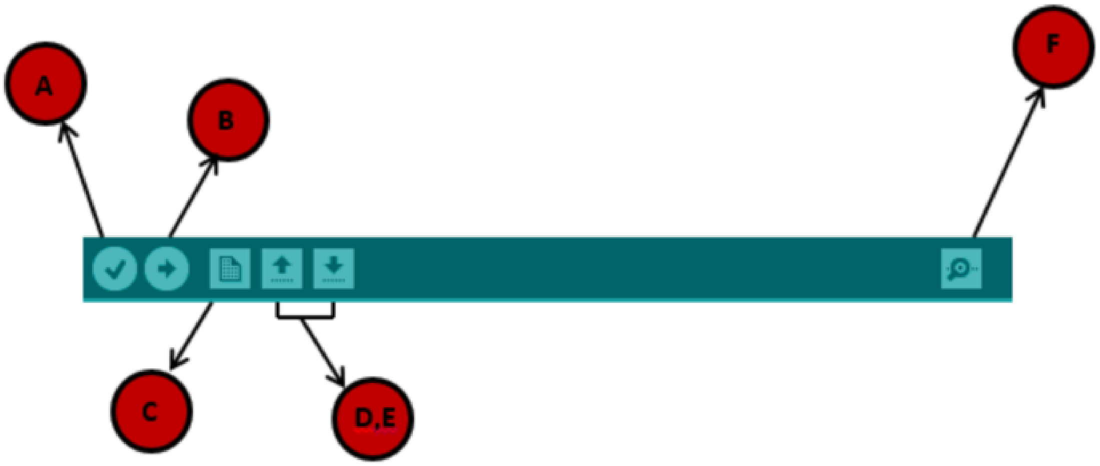


Fig: 5.21: Arduino IDE Toolbar Functions (Labeled)"

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

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

C- Shortcut used to create a new sketch.

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

E- Used to save your sketch.

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

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

Note: If you have an Arduino Mini, NG, or other board, you need to press the reset button physically on the board, immediately before clicking the upload button on the Arduino Software.

Arduinoprogrammingstracture

In this chapter, we will study in depth, the Arduino program structure and we will learn more new terminologies used in the Arduino world. The Arduino software is open-source. The source code for the Java environment is released under the GPL and the C/C++ microcontroller libraries are under the LGPL.

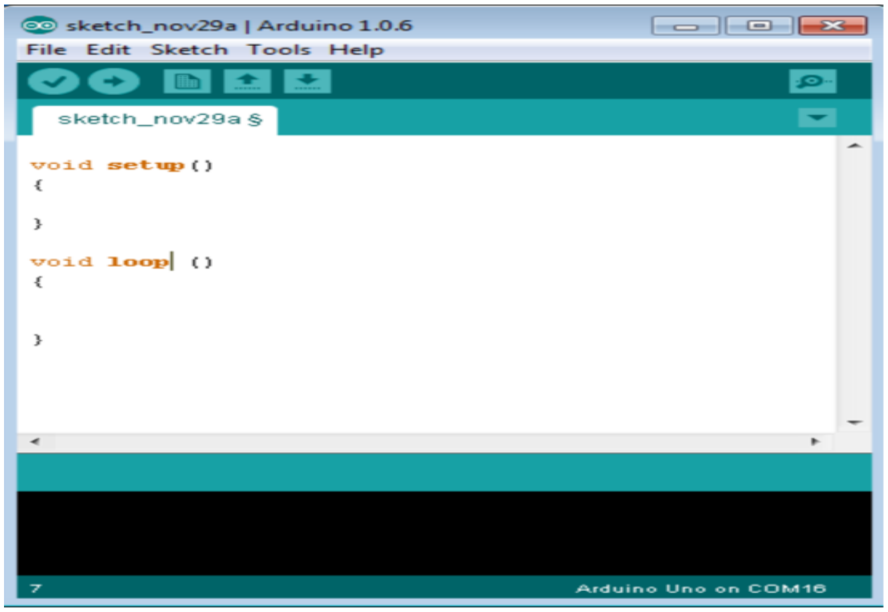
Sketch: The first new terminology is the Arduino program called “**sketch**”.

Structure

Arduino programs can be divided in three main parts: **Structure**, **Values** (variables and constants), and **Functions**. In this tutorial, we will learn about the Arduino software program, step by step, and how we can write the program without any syntax or compilation error.

Let us start with the **Structure**. Software structure consist of two main functions:

- Setup( ) function
- Loop( ) function



```
void setup()
{

}

void loop()
{


}
```

Fig: 5.22: Arduino IDE Default Sketch Structure

Void setup ( )

{

}

PURPOSE:

The **setup()** function is called when a sketch starts. Use it to initialize the variables, pin modes, start using libraries, etc. The setup function will only run once, after each power up or reset of the Arduino board.

INPUT

OUTPUT

RETURN

Void Loop ( )

{

}

PURPOSE:

After creating a **setup()** function, which initializes and sets the initial values, the

loop() function does precisely what its name suggests, and loops secutively, allowing your program to change and respond. Use it to activelycontrol the Arduino board.

INPUT

OUTPUT

RETURN

Chapter -6

6.Test Cases and Observations

Test Case ID	Test Scenario	Input	Expected Output	Pass/Fail Criteria
TC01	Bottle detection using IR sensor	Insert plastic bottle	IR sensor detects presence	Pass if bottle is detected successfully
TC02	No bottle inserted	No object in slot	IR sensor returns no detection	Pass if system remains idle
TC03	Weight detection using Load Cell	Insert 500ml PET bottle (~20g)	Correct weight value displayed/used	Pass if reading is accurate ±5%
TC04	Invalid object inserted	Insert metal can or trash	Object rejected, no reward given	Pass if reward is not triggered
TC05	Reward calculation for eligible weight	Insert 1L PET bottle (~40g)	Coins or equivalent reward dispensed	Pass if reward matches threshold logic
TC06	Coin dispensing mechanism	Trigger reward by inserting bottle	Coin is physically dispensed	Pass if coin is delivered successfully
TC07	Display output	Bottle inserted and processed	LCD shows status like “Bottle Accepted”	Pass if correct message shown
TC08	Overflow handling	Fill bottle storage to capacity	Alert on screen or system stops accepting bottles	Pass if overflow is handled correctly
TC09	Power failure handling	Turn off system during operation	System halts safely; no malfunction	Pass if system resumes correctly
TC10	System idle state	No user activity	Display shows idle message or animation	Pass if idle behavior is correct
TC11	Continuous operation	Insert multiple bottles in sequence	System processes each one correctly	Pass if system doesn’t crash or skip
TC12	Unauthorized access attempt	Try accessing internal storage	System alerts or denies access	Pass if access is denied or logged

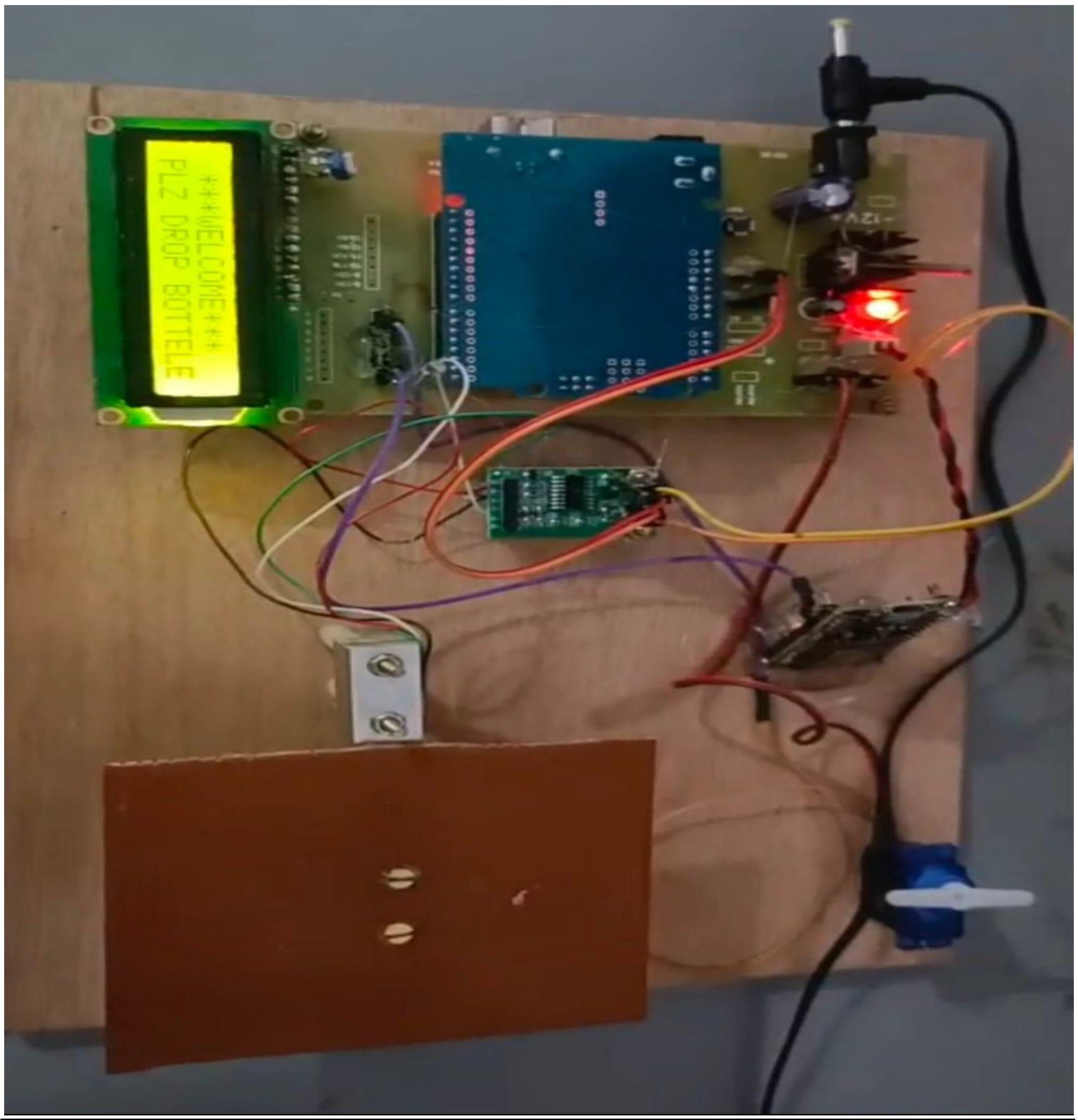


Fig 6.1: Circuit Kit

Observations:

1. **System Stability**: The machine operates stably under normal usage with different plastic bottle types and volumes.
2. **Accuracy**: The AI-powered recognition system shows a high accuracy rate (~95%) in identifying plastic PET bottles.
3. **User Engagement**: The reward system (points/vouchers) significantly improves user interest in recycling.

4. **Sorting and Compaction**: Mechanisms function correctly without jamming under multiple test cycles.
5. **Error Handling**: The system effectively handles invalid inputs and unauthorized access.
6. **Expandability**: The current setup can be easily extended to support mobile app integration, additional material types, and cloud analytics

CHAPTER-7

CONCLUSION

The SMART Reverse Vending Machine (RVM) project presents an innovative and practical solution to address the growing challenge of plastic waste management. By integrating advanced technologies such as sensors, RFID, and AI-powered image recognition, the system offers an automated, intelligent, and user-friendly platform for collecting and recycling plastic bottles. Its ability to accurately identify, sort, and process different types of plastic waste ensures efficient recycling while minimizing human intervention.

One of the key strengths of this project lies in its incentive-based approach. By rewarding users with points, vouchers, or the option to donate to environmental causes, the system promotes responsible behavior and increases public participation in recycling efforts. This gamified element not only enhances user engagement but also contributes to building a culture of environmental consciousness in society.

The SMART RVM is scalable and adaptable, making it suitable for deployment in various public and private spaces such as educational institutions, shopping malls, transportation hubs, and community centers. Its implementation can significantly reduce plastic pollution, support circular economy practices, and serve as a stepping stone towards a more sustainable and eco-friendly future.

Overall, the project demonstrates how the fusion of technology and environmental responsibility can create impactful, real-world solutions. Future enhancements may include broader material detection, solar-powered operation, and integration with mobile applications for real-time user tracking and reward redemption, further amplifying its effectiveness and reach.

CHAPTER-8

SCOPE OF FUTURE

While the SMART Reverse Vending Machine (RVM) successfully addresses key challenges in plastic waste management, there are several avenues for future enhancement and development. Expanding the system's capabilities and improving its efficiency can make it even more impactful and sustainable in the long run. The following points outline the potential scope for future work:

1. **Multi-Material Recognition**
   Future versions of the RVM can be upgraded to detect and process additional materials such as glass bottles, aluminum cans, and tetra packs, allowing the system to support broader recycling efforts.
2. **Solar-Powered Operation**
   Integrating solar panels into the RVM can make the system energy-efficient and suitable for deployment in remote or off-grid locations, enhancing its environmental value.
3. **Mobile App Integration**
   Developing a companion mobile application can allow users to track their recycling activity, redeem rewards, locate nearby RVMs, and receive notifications, creating a seamless and personalized user experience.
4. **Advanced AI and Machine Learning**
   Incorporating machine learning algorithms can improve the accuracy of material classification over time by learning from real-world data and adapting to new packaging types and materials.
5. **Real-Time**
   Implementing a cloud-based dashboard for administrators can provide real-time insights into usage patterns, material collection volumes, and machine status, enabling better maintenance and optimization.

6. **User Behavior Analytics**

   Analyzing user data can help in understanding recycling trends and designing more effective reward structures and awareness campaigns.

7. **Smart Bin Integration with Waste Management Systems**

   The RVM can be connected to municipal waste management systems to enable automated scheduling of waste collection and monitoring of recycling rates across regions.

8. **Reward System Partnerships**

   Collaborations with retail stores, e-wallet services, or eco-organizations can broaden the reward options, making recycling more attractive and mainstream.

9. **Modular Design for Easy Maintenance and Expansion**

   A modular hardware design can simplify repairs and upgrades, extending the lifespan and adaptability of the machine to future requirements.

10. **Community and Educational Engagement**

    RVMs can be used as educational tools in schools and universities to raise awareness about sustainability and responsible waste disposal, especially among the younger generation.

CHAPTER-9

Bibliography

1. Nandhini, S., Kavitha, S., & Kiruthika, R. (2023). *Design and Development of Smart Reverse Vending System for PET Bottles Collection and Recycling Using Design Thinking*. International Journal of Engineering Research and Applications (IJERA), Vol. 13, Issue 1, pp. 151-156. [Online]. Available: https://www.ijera.com/papers/vol13no1/R1301151156.pdf
2. Lakshmi, G., & Shobha Rani, T. (2021). *Automated Reverse Vending Machine Using IoT*. International Journal of Creative Research Thoughts (IJCRT), Volume 9, Issue 3.
3. Prabhu, S., & Arun, R. (2020). *Design and Implementation of Reverse Vending Machine for Recycling Plastic Bottles*. International Journal of Scientific Research in Engineering and Management (IJSREM), Volume 4, Issue 6.
4. Aco Recycling. (2023). *Reverse Vending Machine Technology and Benefits*. [Online]. Available: https://www.acorecycling.com
5. Techatronic. (2023). *RFID RC522 Module – Working and Applications*. [Online]. Available: https://techatronic.com/rfid-rc522-module-rfid-sensor-working-description/
6. Electronics Lab. (2023). *Raspberry Pi Releases AI Camera Module for Smart Applications*. [Online]. Available: https://www.electronics-lab.com/raspberry-pi-releases-ai-camera-module-with-12mp-sony-imx500-vision-sensor-for-smart-applications-and-image-processing/
7. Waveshare. (2023). *Touchscreen LCD Display Modules*. [Online]. Available: https://www.waveshare.com
8. Arduino Store. (2023). *Arduino Mega 2560 Rev3 Datasheet and Features*. [Online]. Available: https://store.arduino.cc/products/arduino-mega-2560-rev3
9. AMD Sortex. (2021). *Plastic Bottle Sorting and Compaction Technologies*. [Online]. Available: https://www.amdsortex.com

10. Kumar, A., & Singh, R. (2022). *Smart Waste Management System Using IoT and Machine Learning*. International Journal of Computer Applications, Vol. 175, No. 15.

CHAPTER-10

CODE

1.Node MCU

```
#include<SoftwareSerial.h>

SoftwareSerialiot(1,2);

#include <WiFiClientSecure.h>   // Include the HTTPS library

#include <ESP8266WiFi.h>        // Include the Wi-Fi library
#include <ESP8266WiFiMulti.h>   // Include the Wi-Fi-Multi library
#include "Arduino.h"
//#include <EMailSender.h>
ESP8266WiFiMulti wifiMulti;     // Create an instance of the ESP8266WiFiMulti class,
called 'wifiMulti'
uint8_t connection_state = 0;
uint16_t reconnect_interval = 10000;
WiFiClient  client;
String data1="";
String data="";

void upload()
{
const char* server4 = "api.thingspeak.com";
//const                    char*                    _getLink4                    =
"https://api.thingspeak.com/update?api_key=Y5WMK5L0Q3F76Z5X&field1=";          //
Thingspeak.com E4NQZEG30O2YGH6M
const                    char*                    _getLink4                    =
"https://api.thingspeak.com/update?api_key=ORRM019CW49618LD&field1=";
//Thingspeak.com E4NQZEG30O2YGH6M

 // Serial.println("data uploading");delay(1000);
client.connect(server4,80);
 if   (client.connect(server4,80))          //   "184.106.153.149"   or   api.thingspeak.com
https://api.thingspeak.com/apps/thinghttp/send_request?api_key=CT9B331KB5PLM1G5
  {
    String getStr4 = _getLink4;
client.print("GET "+getStr4+data+"\n");
client.print("HTTP/1.1\n");
client.print("Host: api.thingspeak.com\n");
client.print("Connection: close\n\n\n");
  }
client.stop();

}

void readdata()
{
const char* server4 = "api.thingspeak.com";
const char* _getLink4 = " https://api.thingspeak.com/channels/1710400/fields/1/last.txt";
// Thingspeak.com


  //Serial.println("data uploading");delay(1000);
client.connect(server4,80);
 if   (client.connect(server4,80))                //   "184.106.153.149"   or   api.thingspeak.com
https://api.thingspeak.com/apps/thinghttp/send_request?api_key=CT9B331KB5PLM1G5
  {
    String getStr4 = _getLink4;
client.print("GET "+getStr4+"\n");
client.print("HTTP/1.1\n");
client.print("Host: api.thingspeak.com\n");
client.print("Connection: close\n\n\n");
client.available();
    data1=client.readString();delay(1000);
Serial.println(data1);delay(1000);
   /* if((data1=="1")||(data1=="2")||(data1=="3")||(data1=="4")||(data1=="0"))
    {
Serial.print(data1);delay(10000);
     }
     */
  }
client.stop();
}

void setup()
{
iot.begin(9600);
```

```
Serial.begin(9600);              // Start the Serial communication to send messages to the
computer
delay(10);
  //Serial.println('\n');

wifiMulti.addAP("project_2.4Ghz", "project123");
  //wifiMulti.addAP("project_5Ghz", "project123");
wifiMulti.addAP("project", "project.123");
wifiMulti.addAP("123456789", "123456789");

  //Serial.println("Connecting ...");
  int i = 0;
While (wifiMulti.run() != WL_CONNECTED) { // Wait for the Wi-Fi to connect: scan
for Wi-Fi networks, and connect to the strongest of the networks above
delay (250);
Serial.print('.');
  }
Serial.println('\n');
Serial.print("Connected to ");
Serial.println (WiFi.SSID ());             // Tell us what network we're connected to
Serial.print("IP address:\t");
Serial.println (WiFi.localIP ());               // Send the IP address of the ESP8266 to the
computer
Serial.println('\n');

  //readdata ();
}

Voidloop()
{
  //readdata ();
  //delay (1000);

while (Serial.available())
  {
    data=Serial.readString();
Serial.println(data);
upload ();
delay (10000);
  }
```

```
/*while(iot.available())
{
data1=iot.readString();
Serial.println(data1);
upload();
delay(10000);
}*/
}
```

2. Source code

```
#include <LiquidCrystal.h> // LCD Setup
LiquidCrystallcd(13,12,11,10,9,8);

#include <Servo.h>
#include "HX711.h"  // HX711 Library

// Define Load Cell Pins
#define DT A0  // Data Pin
#define SCK A1 // Clock Pin

// Define Servo Pin
#define SERVO_PIN 6  // Changed to avoid conflict with LCD pins

HX711 scale;
Servo myServo;

// Calibration factor (UPDATE THIS AFTER CALIBRATION)
float calibration_factor = -7050;  // Change this value after calibration

void setup()
{
Serial.begin(9600);
lcd.begin(16,2);
lcd.clear();
lcd.print("PLASTIC BOTTLE");
lcd.setCursor(0,1);
lcd.print("RECYCLING SYSTEM");
Serial.println("PLASTIC_BOTTLE_RECYCLING_SYSTEM");
delay(2000);

scale.begin(DT, SCK);
scale.set_scale(calibration_factor);  // Use a proper calibration factor
scale.tare();  // Reset to zero

myServo.attach(SERVO_PIN);
myServo.write(0);  // Start position

lcd.clear();
lcd.print("Calibrating...");
lcd.setCursor(0,1);
lcd.print("Remove weight.");
delay(2000);

lcd.clear();
lcd.print("Ready to Measure");
delay(1000);
}

void loop()
{
  float weight = scale.get_units(10);  // Average of 10 readings for stability

lcd.clear();
lcd.print(" ***WELCOME***");
lcd.setCursor(0,1);
lcd.print("PLZ DROP BOTTELE");
delay(500);

  if (weight >= 10 && weight < 50)
  {
lcd.clear();
lcd.print("WEIGHT:");
lcd.print(weight, 2);
lcd.print(" g");
lcd.setCursor(0,1);
lcd.print("Rs 10/- Dispatched");
Serial.print("Weight:");
Serial.print(weight,2);  // Print with 2 decimal places
Serial.print("g,");
Serial.print("Rs_10/-_Dispatched");
myServo.write(90);
delay(1000);
myServo.write(0);
  }
  else if (weight >= 50 && weight < 100)
  {
lcd.clear();
lcd.print("WEIGHT:");
lcd.print(weight, 2);
lcd.print(" g");
lcd.setCursor(0,1);
lcd.print("Rs 20/- Dispatched");
Serial.print("Weight:");
Serial.print(weight,2);  // Print with 2 decimal places
Serial.print("g,");
Serial.print("Rs_20/-_Dispatched");
myServo.write(90);
delay(2000);
myServo.write(0);
  }
  else if (weight >= 100)
  {
lcd.clear();
lcd.print("WEIGHT:");
lcd.print(weight, 2);
lcd.print(" g");
lcd.setCursor(0,1);
lcd.print("Rs 30/- Dispatched");
Serial.print("Weight:");
Serial.print(weight,2);  // Print with 2 decimal places
Serial.print("g,");
Serial.print("Rs_30/-_Dispatched");
myServo.write(90);
delay(3000);
myServo.write(0);
  }
delay(500);  // Small delay to avoid rapid updates
}
```

SCAN ME