IOT BASED DIGITAL WASTE MANAGEMENT ROBOT FOR GARBAGE CLEANING

A PROJECT REPORT

Submitted by

JAGADEESHAN.D	**211419105048**
DHARANI DHARAN.A	**211419105029**
GOKUL.G	**211419105036**
ALLEN FERNANDEZ.T	**211419105005**

in partial fulfilment for the award of the degree

of

BACHELOR OF ENGINEERING

in

ELECTRICAL AND ELECTRONICS ENGINEERING

PANIMALAR ENGINEERING COLLEGE

(An Autonomous Institution, Affiliated to Anna University, Chennai)

APRIL 2023

PANIMALAR ENGINEERING COLLEGE

(An Autonomous Institution, Affiliated to Anna University, Chennai)

BONAFIDE CERTIFICATE

Certified that this project report **"IOT BASED DIGITAL WASTE MANAGEMENT ROBOT FOR GARBAGE CLEANING "** is the bonafide work of **" JAGADEESHAN.D (211419105048), DHARANI DHARAN.A (211419105029), GOKUL.G(211419105036), ALLEN FERNANDEZ.T (211419105005) "** who carried out the project work under my supervision.

SIGNATURE

Dr. S.SELVI, M.E, Ph.D.

HEAD OF THE DEPARTMENT

PROFESSOR,

Department of Electrical and Electronics Engineering,
Panimalar Engineering College,
Chennai-600 123.

SIGNATURE

Mr. P.SANTHOSH, M.Tech.

ASSISTANT

PROFESSOR,

Department of Electrical and Electronics Engineering,
Panimalar Engineering College,
Chennai-600 123.

Submitted for End Semester Project Viva Voce held on.....at Panimalar Engineering College, Chennai**.**

INTERNAL EXAMINER **EXTERNAL EXAMINER**

ACKNOWLEDGEMENT

Our sincere thanks to our Honourable Founder and Chairman, **Dr.JEPPIAAR,M.A.,B.L.,Ph.D.,** for his sincere endeavour in educating us in his premier institution.

We would like to express our deep gratitude to our beloved **Secretary and Correspondent, Dr.P.CHINNADURAI, M.A.,M.Phil.,Ph.D** for his enthusiastic motivation which inspired us a lot in completing this project and our sincere thanks to our Directors **Mrs.C.VIJAYA RAJESWARI, Dr.C.SAKTHI KUMAR, M.E.,Ph.D** and **Dr.SARANYASREE SAKTHIKUMAR,B.E,M.B.A,Ph.D** for providing us with the necessary facilities for the completion of this project.

We would like to express thanks to our Principal, **Dr. K. MANI M.E., Ph.D.,** for having extended his guidance and cooperation.

We would also like to thank our **Head of the Department**, **Dr.S. SELVI, M.E., Ph.D., Professor and Head, Department of Electrical and Electronics Enginering** for her encouragement.

Personally, we thank our Guide **Mr. SANTHOSH, M.Tech.in Department of Electrical and Electronics engineering** for the persistent motivation and support for this project, who at all times was the mentor of germination of the project from a small idea.

We express our sincere thanks to the project coordinators **Dr.S.DEEPA & Dr.N.MANOJ KUMAR,M.E.,Ph.D., in Department of Electrical and Electronics Engineering** for the Valuable suggestions from time to time at every stage of our project.

Finally, we would like to take this opportunity to thank our family members, faculty and non-teaching staff members of our department, friends, well-wishers who have helped us for the successful completion of our project.

ABSTRACT

As the second most populous country in the world India face a major problem in waste management. As of now there are traditional waste management systems like periodic and routine clearing by the various civic bodies like the municipal corporation. But even though these routine maintenances is carried out we often come across overflowing garbage bins from which the garbage spills on to the streets. Now, with the upcoming large number of smart cities, large numbers of responsibilities are also required to be fulfilled. The prime need of a smart lifestyle begins with cleanliness and cleanliness begins with dustbin. A society will get its waste dispatched properly only if the dustbins are placed well and collected well.

The main problem in the current waste management system in most of the Indian cities is the unhealthy status of dustbins. To design and implement for autonomous movement along with trash or waste collection by means of suction inside a small room or home with obstacle avoidance. In this project we have tried to upgrade the level of the urban waste management system.

CONTENTS

LIST OF FIGURES

LIST OFABBREVIATION

RSSI	Received Signal Strength Indicator
LCD	Liquid Crystal Display
WSN	Wireless Sensor Network
RFID	Radio Frequency Identification
ESP32	Espressif System's 32-bit Microcontroller
WiFi	Wireless Fidelity
RF	Radio Frequency
SOC	System-on-Chip
TSMC	Taiwan Semiconductor Manufacturing Company
SRAM	Static Random Access Memory
LED	Light Emitting Diode
PWM	Pulse Width Modulation
GPIO	General Purpose Input/Output
SPI	Serial Peripheral Interface
SDIO	Secure Digital Input/Output
MMC	MultiMediaCard
SD	Secure Digital
UART	Universal Asynchronous Receiver/Transmitter
DAC	Digital-to-Analog Converter
LAN	Local Area Network

MAC	Media Access Control
ROM	Read-Only Memory
SHA	Secure Hash Algorithm
AES	Advanced Encryption Standard
AIDC	Automatic Identification and Data Capture
TTL	Transistor-Transistor Logic
DLL	Dynamic Link Library
IC	Integrated Circuit
AVR	Advanced Virtual RISC (a microcontroller family)
EEPROM	Electrically Erasable Programmable Read-Only Memory
FTDI	Future Technology Devices International
VIN	Voltage Input

CHAPTER 1

INTRODUCTION

Now-a-days people are living in a busy life. So their working time will be more and irregular. In this situation people always find a way to save their time for relaxation. So cleaning a room or particular area will be considering as a boring and tedious job. Garbage is the major problem not only in cities but also in rural areas of India. It is a major source of pollution. Indian cities alone generate more than 100 million tons of solid waste a year. In 2000, India's Supreme Court directed all Indian cities to implement a comprehensive waste-management programme that would include household collection of segregated waste, recycling and composting. These directions have simply been ignored. No major city runs a comprehensive program m of the kind envisioned by the Supreme Court. It is not wrong to say that India is on verge of garbage crisis even though 9000crore rupees are allotted for the Swachh Bharath Abhiyan. There are already different type of garbage collection robots [1-3] like Robo-dumpster which mainly aims at collecting garbage from full cans and dispose it designated area and Dust cart which is designed to navigate through urban areas avoiding static and dynamic obstacle and waste door to door.

CHAPTER 2

LITERATURE SURVEY

AUTOMATED INDOOR WASTE MANAGEMENT SYSTEM EMPLOYING WAVEFRONT ALGORITHM AND RECEIVED SIGNAL STRENGTH INDICATOR VALUES-BASED MOBILE ROBOT.

To live is to help each other, to serve humanity, as a whole. But, in retrospection, the way waste management is being done clearly defies this. Waste management, both indoor and outdoor, is almost done manually. This is not only unhygienic, but also requires significant amount of valuable human resource to get it done. The unresolved problem of waste management is more of a hindrance in indoor environments like shopping malls, hospitals, schools, offices and homes; since, at least outdoor waste management is automated to an extent. Therefore a proposal to fully automate indoor waste management, by making the existing disposal outlets more intelligent and using a movable waste collecting robot, is discussed in this paper. The filling of the dustbin is monitored by ultrasonic sensors and if it is filled to the brim, the Arduino Nano controller transmits the data to the robot with the aid of wireless Zigbee 802.15.4 protocol. The robot is designed in such a way that it effectively tracks the location of the filled dustbin and collects the waste in its storage part. The RSSI (Received Signal Strength Indicator) value from the message received is used to identify which dustbin is full and its location based on Wave Front Algorithm.

AUTHORS: Poorani Ravindhiran, 2 Pradeep Gopal

SMART DUSTBIN: AN EFFICIENT GARBAGE MANAGEMENT APPROACH FOR A HEALTHY SOCIETY

As the population is increasing day by day, the environment should be clean and hygienic. In most of the cities the overflowed garbage bins are creating an unhygienic environment. This will further lead to a rise of different types of diseases. This will degrade the standard of living. To overcome these situations an efficient smart garbage management method has to be developed. In recent decades there is a quick development in the rate of urbanization and therefore there is a need of economical urban improvement designs. Presently utilizing new age innovation and vital approach, the idea of brilliant urban areas is coming up all around the globe. A brilliant city is inadequate without an efficient waste administration framework. This paper depicts the utilization of our model of "Smart Dustbin" in dealing with the waste accumulation management. The dustbin itself works as a robot, when it is full on command from authorized person it goes to prelearned path (for first time user has to guide towards garbage dumping area) and empties itself. An authorized person gives command from Webpage where dustbin status is updated regularly.

AUTHORS: Jayshree Ghorpade-Aher, Anagha Wadkar

SMART DUAL DUSTBIN MODEL FOR WASTE MANAGEMENT IN SMART CITIES

As urbanization is spreading rapidly, there is an increase in production of waste. Waste management is a crucial issue to be considered at public places where waste is overflowed from the bins and may cause different diseases. The present work focuses to develop a model of smart dustbin which can be effectively used at public places in smart cities. The model has two dustbins (named as Dustbin A and Dustbin B) which will be kept at public places mostly. Dustbin A can be used but Dustbin B cannot be used until Dustbin A is full. Dustbin B can only be used once Dustbin A is full and then Dustbin A will not open until the waste is cleared in the Dustbin A. Whenever any dustbin is filled up, a message is sent to the concerned authority. This will avoid overflow of waste in the bin. Dustbins have automatically close and open feature depending on the presence of an obstacle. In our system, the garbage level in the dustbins is detected with the help of Ultrasonic sensor and presence of the obstacle is detected by IR Sensors and communication to the authorized control room by GSM system. includes potential expansion to other regions, integration with emerging technologies, enhancements based on user feedback, collaboration with stakeholders, integration with existing waste management systems, customization for different sectors, and data-driven insights and analytics.

AUTHORS: G Sai Rohit1,Student Member IEEE,M Bharat Chandra

IOT BASED SMART GARBAGE ALERT SYSTEM USING ARDUINO UNO

Waste management is one of the primary problem that the world faces irrespective of the case of developed or developing country. The key issue in the waste management is that the garbage bin at public places gets overflowed well in advance before the commencement of the next cleaning process. It in turn leads to various hazards such as bad odor & ugliness to that place which may be the root cause for spread of various diseases.To avoid all such hazardous scenario and maintain public cleanliness and health this work is mounted on a smart garbage system. The main theme of the work is to develop a smart intelligent garbage alert system for a proper garbage management. This paper proposes a smart alert system for garbage clearance by giving an alert signal to the municipal web server for instant cleaning of dustbin with proper verification based on level of garbage filling. This process is aided by the ultrasonic sensor which is interfaced with Arduino UNO to check the level of garbage filled in the dustbin and sends the alert to the municipal web server once if garbage is filled. After cleaning the dustbin, the driver confirms the task of emptying the garbage with the aid of RFID Tag. RFID is a computing technology that is used for verification process and in addition, it also enhances the smart garbage alert system by providing automatic identification of garbage filled in the dustbin and sends the status of clean-up to the server affirming that the work is done. The whole process is upheld by an embedded moduleintegrated with RF ID and IOT Facilitation.

The real time status of how waste collection is being done could be monitored and followed up by the municipality authority with the aid of this system. In addition to this the necessary remedial / alternate measures could be adapted. An Android application is developed and linked to a web server to intimatethe alerts from the microcontroller to the urban office and to perform the remote monitoring of the cleaning process, done by the workers, thereby reducing the manual process of monitoring and verification. The notifications are sent to the Android application using Wi-Fimodule.

Authors: Dr.N.Sathish Kumar, B.Vijayalakshmi.

CHAPTER 3

SYSTEM DESIGN

3.1.EXISTING SYSTEM:

The existing system includes traditional garbage bins placed at public places, such as parks, streets, and other communal areas, where people dispose of their waste.

The system incorporates ultrasonic sensors that are installed inside the garbage bins to measure the level of garbage in real-time. These sensors use ultrasonic waves to detect the distance between the sensor and the garbage, providing accurate information about the garbage level. The ultrasonic sensors are interfaced with Arduino UNO, which is a microcontroller board that processes the sensor data and sends it to the municipal web server for further action.The system includes a central web server maintained by the municipal authority, which receives the garbage level data from the Arduino UNO and sends instant alerts to the designated personnel for prompt action. The system uses RFID tags that are attached to the garbage bins for verification purposes. The driver of the garbage collection vehicle confirms the task of emptying the garbage bins by scanning the RFID tags, which updates the server with the status of the cleanup. The system includes an Android application that is linked to the municipal web server, providing real-time alerts, notifications, and remote monitoring of the garbage collection process. The application allows municipal authorities to monitor the garbage level, cleanup status, and receive alerts for prompt action. The system uses Wi-Fi modules to facilitate communication between the garbage bins, sensors, Arduino UNO, web server, and the Android application, enabling seamless data transmission and real-time monitoring.

3.2 PROPOSED SYSTEM:

Machine vision system, robot hardware design and fuzzy controller design for autonomous multi-agent mobile robot platform. In this study we present the machine vision system and the hardware design of our mobile robot with the design of the fuzzy controller. This paper describes the implementation of robot's position and tracking with a use of a vision system. The ground truth data is obtained by the overhead camera over the setup to track the current position of the robot. The process in the camera includes robot detection, image to coordinate transformation, and background subtraction. Several positions were chosen to test the vision system and tracking of the mobile robot using Roborealm. The hardware design of the mobile robot is presented using Sketch Up Pro. The overall dimension was limited to 7.5 cm x 7.5 cm x 7.5 cm in compliance with the standard of the Federation of International Robot-Soccer Association for Micro-soccer Robot World Tournament.

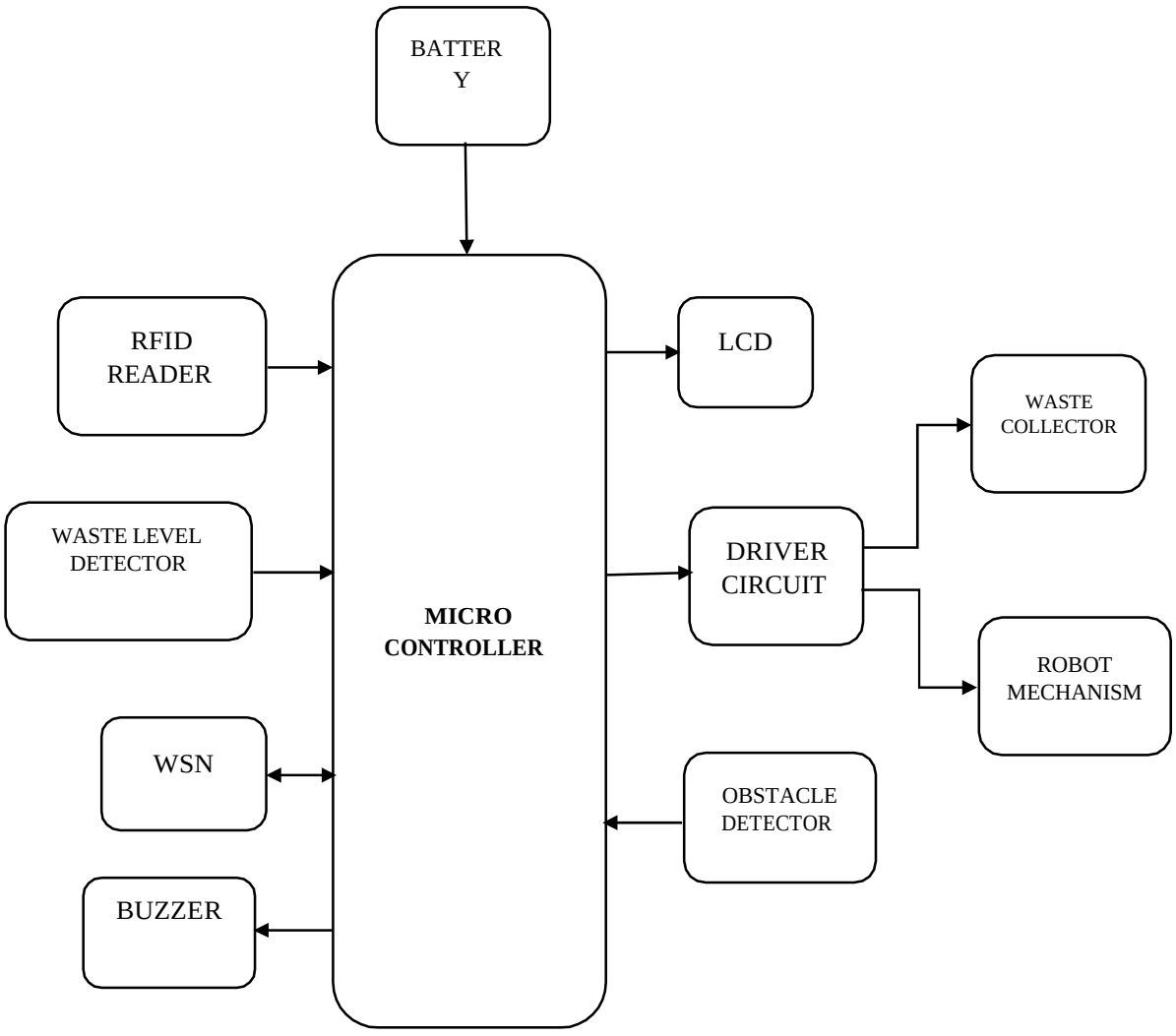
3.3 SYSTEM WORKING:

The smart garbage alert system, which is designed to work with the help of a robot, is a highly efficient and effective solution for waste management in public spaces. The robot uses ultrasonic sensors to measure the level of garbage in the bin and sends an automatic alert to the municipal web server when the bin reaches a certain threshold. The RFID technology is used for verification purposes, ensuring that the garbage is collected and the bin is emptied. The system can be customized to cater to specific market segments and can be scaled up or down depending on the needs of the organization. The remote monitoring capabilities offered through the android application make it easier to manage the cleaning process and track progress. Overall, the robot working model provides a

sustainable and efficient solution for managing waste in public spaces, promotingpublic health and hygiene.

3.4 BLOCK DIAGRAM:

3.4.1 PUBLIC GARBAGE COLLECTOR:



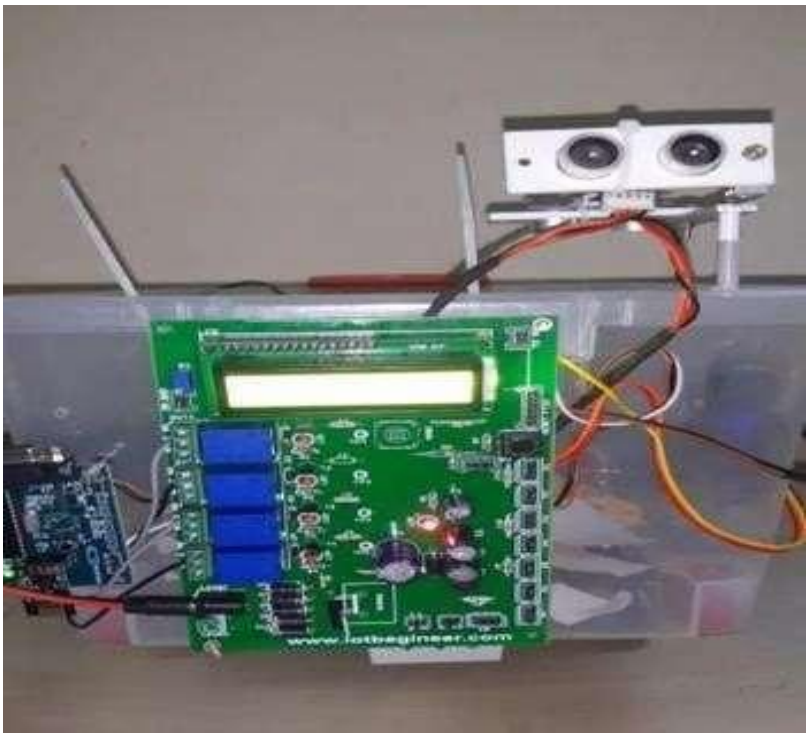
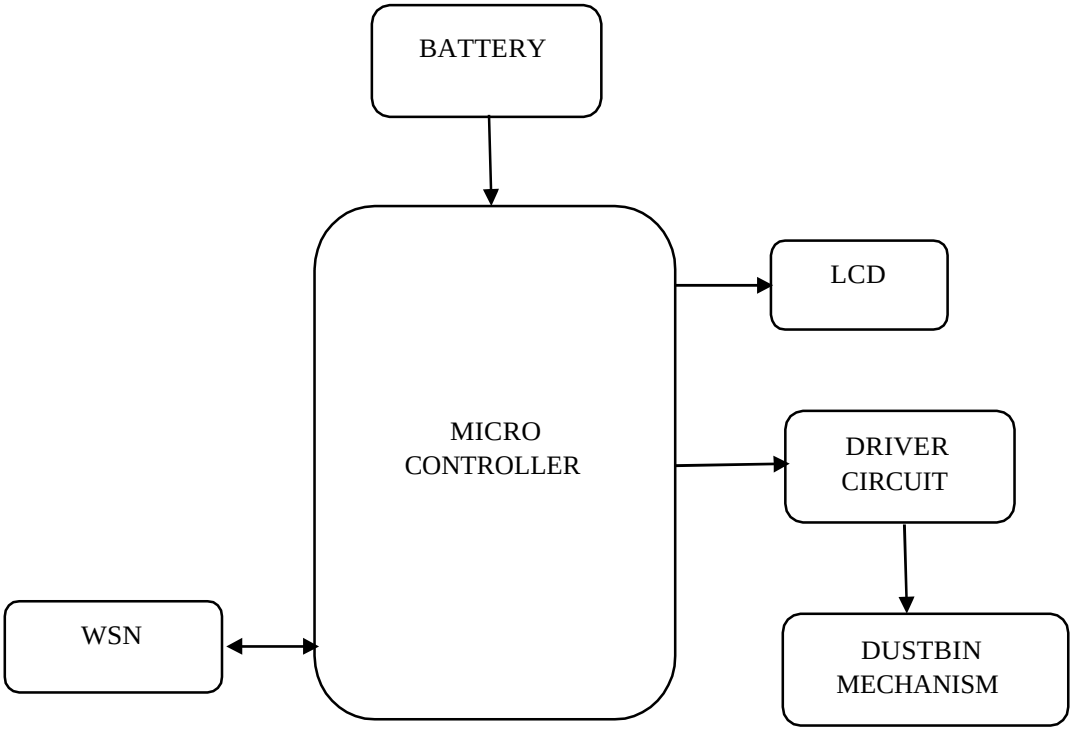


Figure.3.1 Public trash collector.

3.4.2 HOME GARBAGE COLLECTOR:



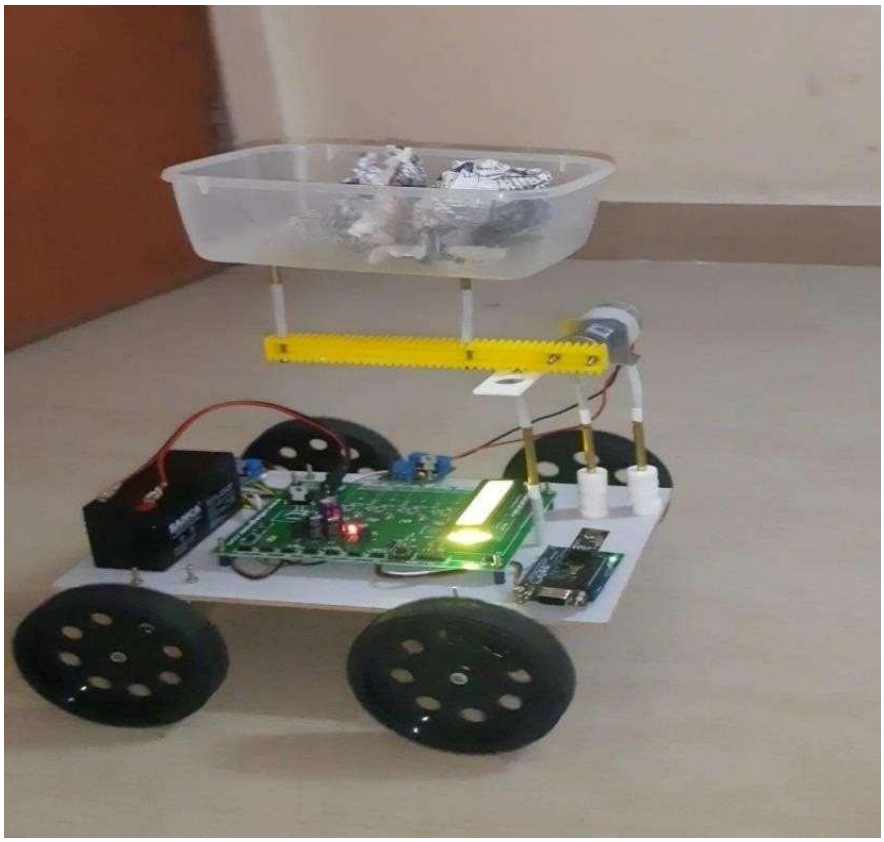


Figure.3.2 Home Trash collector.

CHAPTER 4

HARDWARE IMPLIMENTATION

4.1 HARDWARE REQUIREMENTS:

- MICROCONTROLLER
- BATTERY
- ULTRASONIC SENSORS
- RFID READER
- WSN MODULE
- DRIVER CIRCUIT
- DC MOTORS
- BUZZER
- LIMIT SWITCHES
- LCD

4.2 HARDWARE DESCRIPTION:

4.2.1 Microcontroller:

ESP32

Arduino is a great platform for beginners into the World of Microcontrollers and Embedded Systems. With a lot of cheap sensors and modules, you can make several projects either as a hobby or even commercial.As technology advanced, new project ideas and implementations came into play and one particular concept is the Internet of Things or IoT. It is a connected platform, where several “things” or devices are connected over internet for exchange of information. In DIY

community, the IOT projects are mainly focused on Home Automation and Smart Home applications but commercial and industrial IoT projects have far complex implementations like Machine Learning, Artificial Intelligence, Wireless Sensor Networks etc. The important thing in this brief intro is whether it is a small DIY project by a hobbyist or a complex industrial project, any IoT project must have connectivity to Internet. This is where the likes of ESP8266 and ESP32 come into picture. If you want to add Wi-Fi connectivity to your projects, then ESP8266 is a great option. But if you want build a complete system with Wi-Fi connectivity, Bluetooth connectivity, high resolution ADCs, DAC, Serial Connectivity and many other features, then ESP32 is the ultimate choice.

What is ESP32?

ESP32 is a low-cost System on Chip (SoC) Microcontroller from Espressif Systems, the developers of the famous ESP8266 SoC. It is a successor to ESP8266 SoC and comes in both single-core and dual-core variations of the Tensilica's 32-bit Xtensa LX6 Microprocessor with integrated Wi-Fi and Bluetooth. The good thing about ESP32, like ESP8266 is its integrated RF components like Power Amplifier, Low-Noise Receive Amplifier, Antenna Switch, Filters and RF Balun. This makes designing hardware around ESP32 very easy as you require very few external components.

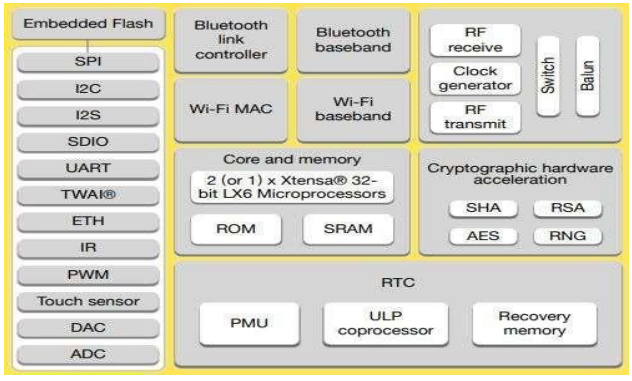


Figure.4.1 ESP-32 Module

Another important thing to know about ESP32 is that it is manufactured using TSMC's ultra-low-power 40 nm technology. So, designing battery operated applications like wearables, audio equipment, baby monitors, smart watches, etc., using ESP32 should be very easy.

4.2.2 General Description:

ESP32 is a series of low-cost, low-power system on a chip microcontroller with integrated Wi-Fi and dual-mode Bluetooth. The ESP32 series employs either a Tensilica Xtensa LX6 microprocessor in both dual-core and single-core variations, Xtensa LX7 dual-core microprocessor or a single-core RISC-V microprocessor and includes built-in antenna switches, RF balun, power amplifier, low-noise receive amplifier, filters, and power-management modules. ESP32 is created and developed by Espressif Systems, a Shanghai-based Chinese company, and is manufactured by TSMC using their 40 nm process.

Figure.4.2 ESP32

4.2.3 Specifications of ESP32:

ESP32 has a lot more features than ESP8266 and it is difficult to include all the specifications in this Getting Started with ESP32 guide. So, I made a list of some of the important specifications of ESP32 here. But for complete set of specifications, I strongly suggest you to refer to the Datasheet.

- Single or Dual-Core 32-bit LX6 Microprocessor with clock frequency up to 240 MHz.
- 520 KB of SRAM, 448 KB of ROM and 16 KB of RTC SRAM.
- Supports 802.11 b/g/n Wi-Fi connectivity with speeds up to 150 Mbps.
- Support for both Classic Bluetooth v4.2 and BLE specifications.
- 34 Programmable GPIOs.
- Up to 18 channels of 12-bit SAR ADC and 2 channels of 8-bit DAC
- Serial Connectivity include 4 x SPI, 2 x I^2C, 2 x I^2S, 3 x UART.
- Ethernet MAC for physical LAN Communication (requires external PHY).
- 1 Host controller for SD/SDIO/MMC and 1 Slave controller for SDIO/SPI.
- Motor PWM and up to 16-channels of LED PWM.
- Secure Boot and Flash Encryption.
- Cryptographic Hardware Acceleration for AES, Hash (SHA-2), RSA, ECC and RNG.

4.3 BATTERY:

In electricity, a battery is a device consisting of one or more electrochemical cells that convert stored chemical energy into electrical energy. Since the invention of the first battery (or "voltaic pile") in 1800 by Alessandro Volta and especially since the technically improved Daniell cell in

1836, batteries have become a common power source for many household and industrial applications. According to a 2005 estimate, the worldwide battery industry generates US$48 billion in sales each year, with 6% annual growth. There are two types of batteries: primary batteries (disposable batteries), which are designed to be used once and discarded , and secondary batteries (rechargeable batteries), which are designed to be recharged and used multiple times. Batteries come in many sizes from miniature cells used to power hearing aids and wristwatches to battery banks the size of rooms that provide standby power for telephone exchanges and computer data centers.

A battery is a device that converts chemical energy directly to electrical energy. It consists of a number of voltaic cells; each voltaic cell consists of two half-cells connected in series by a conductive electrolyte containing anions and cations. One half-cell includes electrolyte and the electrode to which anions (negatively charged ions) migrate, i.e., the anode or negative electrode; the other half-cell includes electrolyte and the electrode to which cations (positively charged ions) migrate, i.e., the cathode or positive electrode. In the redox reaction that powers the battery, cations are reduced (electrons are added) at the cathode, while anions are oxidized (electrons are removed) at the anode. The electrodes do not touch each other but are electrically connected by the electrolyte. Some cells use two half-cells with different electrolytes. A separator between half-cells allows ions to flow, but prevents mixing of the electrolytes.

Batteries are classified into two broad categories, each type with advantages and disadvantages.

- **Primary batteries** irreversibly (within limits of practicality) transform chemical energy to electrical energy. When the initial supply of reactants

is exhausted, energy cannot be readily restored to the battery by electrical means.

- **Secondary batteries** can be recharged; that is, they can have their chemical reactions reversed by supplying electrical energy to the cell, restoring theiroriginal composition.

Some types of primary batteries used, for example, for telegraph circuits, were restored to operation by replacing the components of the battery consumed by the chemical reaction. Secondary batteries are not indefinitely rechargeable due to dissipation of the active materials, loss of electrolyte and internal corrosion.

4.3.1 Primary batteries:

Main article: *Primary cell*

Primary batteries can produce current immediately on assembly. Disposable batteries are intended to be used once and discarded. These are most commonly used in portable devices that have low current drain, are used only intermittently, or are used well away from an alternative power source, such as in alarm and communication circuits where other electric power is only intermittently available. Disposable primary cells cannot be reliably recharged, since the chemical reactions are not easily reversible and active materials may not return to their original forms. Battery manufacturers recommend against attempting recharging primary cells. Common types of disposable batteries include zinc–carbon batteries and alkaline batteries. In general, these have higher energy densities than rechargeable batteries, but disposable batteries do not fare well under high-drain applications with loads under 75 ohms (75 Ω).

4.3.2 Secondary batteries:

Main article: *Rechargeable battery*

Secondary batteries must be charged before use; they are usually assembled with active materials in the discharged state. Rechargeable batteries or *secondary cells* can be recharged by applying electric current, which reverses the chemical reactions that occur during its use. Devices to supply the appropriate current are called chargers or rechargers.

The oldest form of rechargeable battery is the lead–acid battery. This battery is notable in that it contains a liquid in an unsealed container, requiring that the battery be kept upright and the area be well ventilated to ensure safe dispersal of the hydrogen gas produced by these batteries during overcharging. The lead–acid battery is also very heavy for the amount of electrical energy it can supply. Despite this, its low manufacturing cost and its high surge current levels make its use common where a large capacity (over approximately 10 Ah) is required or where the weight and ease of handling are not concerns.

4.4 Ultrasonic Sensor:

4.4.1 General Description:

Ultrasonic sensor emits ultrasonic pulses, and by measuring the time of ultrasonic pulse reaches the object and back to the transducer. The sonic waves emitted by the transducer are reflected by an object and received back in the transducer. After having emitted the sound waves, the ultrasonic sensor will switch to receive mode. The time elapsed between emitting and receiving is proportional to the distance of the object from the sensor.

4.4.2 Product Description:

Ultrasonic transmitter emitted an ultrasonic wave in one direction and started timing when it launched. Ultrasonic spread in the air and would return immediately when it encountered obstacles on the way. At last, the ultrasonic receiver would stop timing when it receives the reflected wave. The distance of sensor from the target object is calculated. It offers excellent non-contact range detection with high accuracy and stable readings in an easy-to-use package. Its operation is not affected by sunlight or black material. The supply voltage to the sensor is 5VDC. The sensor has two pins namely trig and echo which is connected to the controller to give digital input.

Figure.4.3 Ultrasonic sensor

4.4.3 Features:

- Working Voltage: 5VDC
- Quiescent Current: <2mA
- Working Current: 15mA
- Detecting Range: 2cm - 4.5m
- Trigger Input Pulse width: 10uS

4.4.4 Applications:

- Robot navigation
- Obstacle avoidance
- Engineering measurement tools
- Industrial control system

4.5 RFID Reader:

4.5.1 General Description:

A Radio Frequency Identification Reader (RFID reader) is a device used to gather information from an RFID tag, which is used to track individual objects. Radio Frequency waves are used to transfer data from the tag to a reader. The RFID tag it must be within the range of an RFID reader, in order to be read. RFID technology allows several items to be quickly scanned and enables fast identification of a particular product, even when it is surrounded by several other items.

4.5.2 Product Description:

Radio frequency identification (RFID) is one method for Automatic Identification and Data Capture (AIDC). RFID tags are used in many industries. An RFID system consists of three components: an antenna and transceiver and a transponder. The antenna uses radio frequency waves to transmit a signal that activates the transponder. When activated, the tag transmits data back to the antenna. An RFID reader's function is to interrogate RFID tags. The means of interrogation is wireless and because the distance is relatively short; line of sight between the reader and tags is not necessary. A reader contains an RF module, which acts as both a transmitter and receiver of radio frequency signals.

The transmitter consists of an oscillator to create the carrier frequency; a modulator to impinge data commands upon this carrier signal and an amplifier to boost the signal enough to awaken the tag. The receiver has a demodulator to extract the returned data and also contains an amplifier to strengthen the signal for processing. A microprocessor forms the control unit, which employs an operating system and memory to filter and store the data. The data is now ready to be sent to the network.

Figure.4.4 RFID reader

4.5.2 Features :

- Supply voltage: 12v DC
- Output: UART and TTL
- In-built buzzer indicator
- Signal LED is placed

4.5.3 Applications:

- Passports
- Toll booth passes
- Hospitals

4.6 ZIGBEE:

4.6.1 General Description:

ZigBee is an IEEE 802.15.4-based specification for a suite of high-level communication protocols used for wireless networking. It is a wireless technology developed as an open global standard to address the unique needs of low-cost, low-power wireless M2M networks. ZigBee (CC2500) is a low cost true single chip 2.4 GHz transceiver designed for very low power wireless applications. The RF transceiver is integrated with a highly configurable baseband modem.

4.6.2 Product Description:

ZigBee devices are required to conform to the IEEE 802.15.4-2003 Low Rate Wireless Personal Area Network (LR-WPAN) standard. The standard specifies the lower protocol layers are the physical layer (PHY), and the Media Access Control portion of the data link layer (DLL). The technology defined by the ZigBee specification is intended to be simpler and less expensive than other wireless personal area networks (WPANs), such as Bluetooth or Wi-Fi. Its low power consumption limits transmission distances to 10–100 meters line-of sight, depending on power output and environmental characteristics. ZigBee devices can transmit data over long distances by passing data through a mesh network of intermediate devices to reach more distant ones. ZigBee is typically used in low data rate applications that require long battery life and secure networking. ZigBee has a defined rate of 250 kbit/s, best suited for intermittent data transmissions from a sensor or input device.

Figure.4.5 Zigbee

4.6.3 Features:

- Supply voltage: 5v DC
- Detection range: (10-30) m
- RS232 Output
- TTL UART also provided
- Frequency: 2.4GHz
- Tx and Rx Status LEDs
- Low power

4.6.4 Applications:

- Lighting controls
- Switching
- Wireless keyboard and mouse
- Consumer electronics

4.7 L293D DC MOTOR DRIVER MODULE:

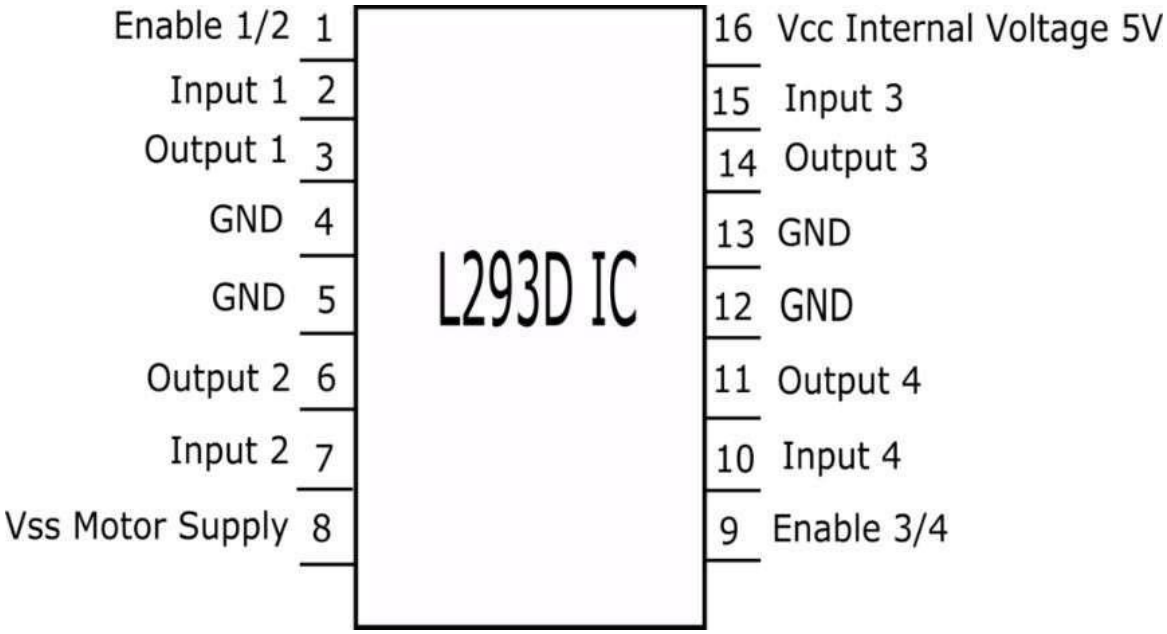
The project designed around L293D IC. The L293D device is quadruple high-current half-H driver. The 293D is designed to provide bidirectional drive current up to 600mA a voltage from 5V to 36V. L293D Adapter Board can be used as dual DC motor driver or bipolar stepper motor driver. Useful in robotics application, bidirectional DC motor controller and stepper motor driver. Separate logic supply to reduce dissipation. L293D includes the output clamping diodes for protections.

4.7.1 Specifications:

- Motor/Logic supply 5 to 36 V
- Logic controls input 7 VDC max
- Inhibit facility/enable
- High Noise immunity
- Over temperature protection
- Capable of delivering output current up to 600 mA per channel
- The control/interface lines are accessible with Berg connector
- Header connector for motor and supply connection
- PCB dimensions 36 mm x 24 mm

L293D IC Pin Out:

The L293D is a 16 pin IC, with eight pins, on each side, to controlling of two DC motor simultaneously. There are 4 INPUT pins, 4 OUTPUT pins and 2 ENABLE pin for each motor.



Input 1	**Input 2**	**Result**
0	0	Stop
0	1	Anti Clockwise
1	0	Clockwise
1	1	Stop

Working Mechanism:

Rotation of motor depends on Enable Pins. When Enable 1/2 is HIGH, motor connected to left part of IC will rotate according to following manner:

4.8 DC MOTOR:

4.8.1 General Description:

The relationship between torque vs speed and current is linear as shown left; as the load on a motor increase, Speed will decrease. The graph pictured here represents the characteristics of a typical motor. As long as the motor is used in the area of high efficiency (as represented by the shaded area) long life and good performance can be expected. However, using the motor outside this range will result in high temperature rises and deterioration of motor parts. A motor's basic rating point is slightly lower than its maximum efficiency point. Load torque can be determined by measuring the current drawn when the motor is attached to a machine whose actual load value is known.

4.8.2 Product Description:

Geared dc motors can be defined as an extension of dc motors. A geared DC Motor has a gear assembly attached to the motor. The speed of motor is counted in terms of rotations of the shaft per minute and is termed as RPM. The gear assembly helps in increasing the torque and reducing the speed. Using the correct combination of gears in a gear motor, its speed can be reduced to any desirable figure. This concept where gears reduce the speed of the vehicle but increase its torque is known as gear reduction. A DC motor can be used at a voltage lower than the rated voltage. But, below 1000 rpm, the speed becomes unstable, and the motor will not run smoothly.

Figure.4.6 DC motor

4.8.3 Features:

- Supply voltage: 12VDC
- Power ratings: 0.36watts
- Speed: 10rpm
- Long Lifetime, Low Noise, Smooth Motion
- Equipped with high efficiency

4.8.4 Applications:

- Coin Changing equipment
- Peristaltic Pumps
- Damper Actuators
- Fan Oscillators.
- Photo copier
- Ticket printer

4.9 BUZZER:

4.9.1 General Description:

A buzzer or beeper is an audio signaling device, which may be mechanical, electromechanical, or piezoelectric. Typical uses of buzzers and beepers include alarm devices, timers and confirmation of user input such as a mouse click or key stroke. Buzzer is an integrated structure of electronic transducers, DC power supply, widely used in computers, printers, copiers, alarms, electronic toys, automotive electronic equipment, telephones, timers and other electronic products for sound devices. Active buzzer 5V Rated power can be directly connected to a continuous sound, this section dedicated sensor expansion module

and the board in combination, can complete a simple circuit design, to "plug and play."

4.9.2 Product Description:

A buzzer or beeper is an audio signaling device, which may be mechanical, electromechanical, or piezoelectric. Typical uses of buzzers and beepers include alarm devices, timers, and confirmation of user input such as a mouse click or keystroke. It generates consistent single tone sound just by applying D.C voltage. Using a suitably designed resonant system, this type can be used where large sound volumes are needed. At Future Electronics we stock many of the most common types categorized by Type, Sound Level, Frequency, Rated Voltage, Dimension and Packaging Type.

Figure.4.7 Buzzer

4.9.3 Features:

- Input supply: 5 VDC
- Current consumption: 9.0 mA max.
- Oscillating frequency: 3.0 ±0.5 KHz
- Sound Pressure Level: 85dB min.

4.9.4 Applications:

- Confirmation of user input (ex: mouse click or keystroke)
- Electronic metronomes
- Sporting events
- Judging Panels
- Annunciator panels

4.10 16×2 LCD:

4.10.1 General Description:

LCD stands for liquid crystal display. They come in many sizes 8x1 , 8x2 , 10x2 , 16x1 , 16x2 , 16x4 , 20x2 , 20x4 ,24x2 , 30x2 , 32x2 , 40x2 etc . Many multinational companies like Philips Hitachi Panasonic make their own special kind of LCD'S to be used in their products. All the LCD'S performs the same functions (display characters numbers special characters ASCII characters etc). Their programming is also same and they all have same 14 pins (0-13) or 16 pins (0 to 15). Alphanumeric displays are used in a wide range of applications, including palmtop computers, word processors, photocopiers, point of sale terminals, medical instruments, cellular phones, etc.

4.10.2 Product Description:

This is an LCD Display designed for E-blocks. It is a 16 character, 2-line alphanumeric LCD display connected to a single 9-way D-type connector. This allows the device to be connected to most E-Block I/O ports. The LCD display requires data in a serial format, which is detailed in the user guide below. The display also requires a 5V power supply. Please take care not to exceed 5V, as this will cause damage to the device. The 5V is best generated from the E-blocks Multi programmer or a 5V fixed regulated power supply. The 16 x 2 intelligent alphanumeric dot matrix displays is capable of displaying 224 different characters and symbols. A full list of the characters and symbols is printed on pages 7/8 (note these symbols can vary between brand of LCD used). This booklet provides all the technical specifications for connecting the unit, which requires a single power supply (+5V).

Figure.4.8 16X2 LCD

4.10.3 Features:

- Input voltage: 5v
- E-blocks compatible
- Low cost
- Compatible with most I/O ports in the E-Block range
- Ease to develop programming code using Flow code icons

CHAPTER 5

SOFTWARE IMPLIMENTATION

5.1 SOFTWARE DESCRIPTION:

The software component of the system plays a critical role in collecting and processing data from the sensors and RFID technology to enable real-time monitoring and alerting. The software is designed to run on an embedded module integrated with RFID and IoT facilitation, which is interfaced with an Arduino UNO board. The software includes programming code written in a suitable programming language, such as C++, to control the operation of the sensors and RFID technology, and to communicate with the municipal web server. The ultrasonic sensor measures the level of garbage in the bin and sends the data to the software, which processes the information and determines if the garbage has reached the threshold for alerting the municipal web server. The RFID technology is used for verification purposes, with the software validating the RFID tag to confirm that the garbage has been collected and the bin emptied. Additionally, the software includes an Android application that is linked to the web server and receives notifications from the embedded module. The application allows for remote monitoring of the cleaning process, reducing the need for manual monitoring and verification. The software component of our project is crucial in enabling real-time monitoring, alerting, and remote management of the smart garbage alert system, ensuring efficient and effective waste management in public spaces.

5.1.1 Arduino Software (IDE):

Get the latest version from the download page. You can choose between the Installer (.exe) and the Zip packages. We suggest you use the first one that installs directly everything you need to use the Arduino Software (IDE), including the drivers. With the Zip package you need to install the drivers manually. The Zip file is also useful if you want to create a portable installation.

When the download finishes, proceed with the installation and please allow the driver installation process when you get a warning from the operating system.

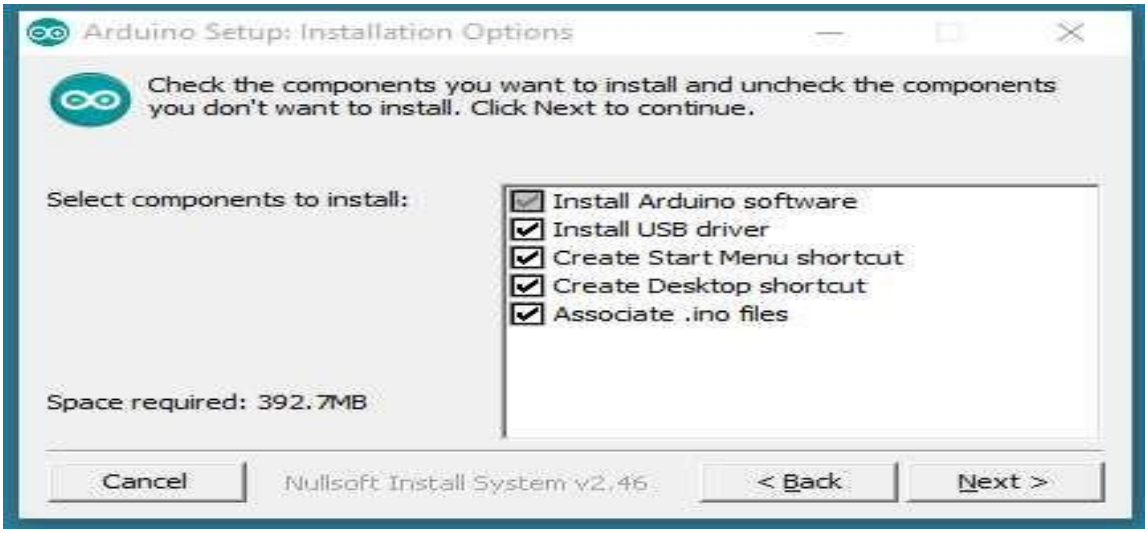


Figure.5.1 Install Arduino software

Choose the components to install Choose the installation directory (we suggest to keep the default tone) The process will extract and install all the required files to execute properly the Arduino Software (IDE)

5.1.2 Arduino Boot loader Issue:

The current boot loader burned onto the Arduino UNO is not compatible with ROBOTC. In its current form, you will be able to download the ROBOTC Firmware to the Arduino UNO, but you will not able to download any user

programs. The reason for this is because there is a bug in the Arduino UNO firmware that does not allow flash write commands to start at anywhere but the beginning of flash memory (0x000000). See the bottom of this page for more technical details. Because ROBOTC is not able to burn a new bootloader as of today, you will need to use the Arduino's Open-Source language with a modified bootloader file to re-burn your bootloader on your Arduino UNO boards. The enhanced bootloader is backwards compatible with the original one. That means you'll still be able to program it through the Arduino programming environment as before, in addition to ROBOTC for Arduino.

5.2 Hardware Needed:

To burn a new version of the Arduino boot loader to your UNO, you'll need an AVR ISP Compatible downloader.

5.2.1 Using an AVR ISP (In System Programmer)

- Your Arduino UNO (to program)
- An AVR Programmer such as the AVR Pocket Programmer
- An AVR Programming Cable (the pocket programmer comes with one)

If you have extra Arduino boards, but no ISP programmer, SparkFun.com has a cool tutorial on how to flash a bootloader using an Arduino as an ISP.

5.2.3 Using another Arduino as an ISP

- Your Arduino UNO (to program)
- A Working Arduino (doesn't matter what kind)
- Some Male-to-Male Jumper Cables

For instructions on this method, take a look at the SparkFun.com website: http://www.sparkfun.com/tutorials/247

5.3 SOFTWARE NEEDED:

ROBOTIC is not currently able to burn a bootloader onto an Arduino board, so you'll need to download a copy of the latest version of the Arduino Open-Source programming language.

- Arduino Official Programming Language - Download Page

In addition, you'll need the ROBOTIC modified bootloader. You can download that here:

- ROBOTIC Modified UNO Bootloader - Modified Bootloader

5.3.1 Bootload Download Instructions:

- Download the Arduino Open-Source Software and a copy of the Modified Bootloader File
- Copy the Modified Bootloader File into the /Arduino-1.0/hardware/Arduino/bootloaders/stk500v2/ and overwrite the existing bootloader.

- Power up your Arduino UNO (either via USB or external power)
- Plug in your AVR ISP Programmer to your computer (make sure you have any required drivers installed)
- Connect your AVR ISP Programmer into your Arduino UNO Board via the ISP Header (the 2x3 header pins right above the Arduino Logo)
- Launch the Arduino Open-Source Software

- Change your settings in the Arduino Software to look for an Arduino UNO

- Change your settings in the Arduino Software to select your ISP Programmer Type (Check your programmer's documentation for the exact model)

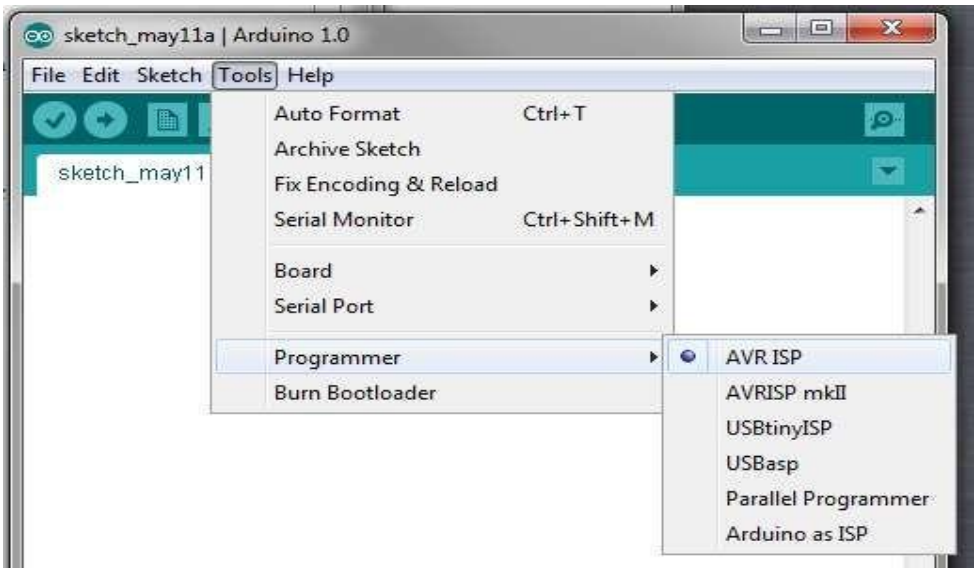


Figure.5.2 Arduino IDE Programmer

- Select the "Burn Bootloader" option under the "Tools" menu. The modified bootloader will now be sent to your Arduino. This typically take a minute or so.

- You should be all set to download ROBOTC firmware and start using your Arduino UNO with ROBOTC.

5.4 TECHNICAL DETAILS:

The Arduino Boot loader sets the "erase Address" to zero every time the boot loader is called. ROBOTC called the "Load Address" command to set the address in which we want to write/verify when downloading program. When writing a page of memory to the Arduino, the Arduino boot loader will erase the existing page and write a whole new page. In the scenario of downloading firmware, everything is great because the Erase Address and the Loaded Address both start at zero.

In the scenario of writing a user program, we start writing at memory location 0x7000, but the Boot loader erases information starting at location zero because the "Load Address" command doesn't update where to erase. Our modification is to set both the Load Address and the Erase Address so the activity of writing a user program doesn't cause the firmware to be accidentally erased.

5.4.1 Summary:

Microcontroller	Arduino UNO
Operating Voltage	5V Input Voltage (recommended)
Input Voltage (limits)	6-20V
Digital I/O Pins	54 (of which 14 provide PWM output)
Analog Input Pins	16
DC Current per I/O Pin	40mA
DC Current for3.3VPin	50mA
Flash Memory	256 KB of which 8 KB used by bootloader
SRAM	8KB
EEPROM	4KB
Lock Speed	16MHz

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 V_{in} 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.

They differ from all preceding boards in that it does not use the FTDI USB-to-serial driver chip. Instead, it features the programmed as a USB-to-serial converter.

5.4.1 The power pins are as follows:

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

The ATMEGA has 256 KB of flash memory for storing code (of which 8 KBis used for the bootloader), 8 KB of SRAM and 4 KB of EEPROM (which can be read and written with the EEPROM library).

Each of the 54 digital pins on the Mega 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 and has an internal

pull-up resistor (disconnected by default) of 20-50k Ohms. In addition, some pins have specialized functions:

- **Serial: 0 (RX) and 1 (TX); Serial 1: 19 (RX) and 18 (TX); Serial 2: 17 (RX) and 16 (TX); Serial 3: 15 (RX) and 14 (TX).** Used to receive (RX) and transmit (TX) TTL serial data. Pins 0 and 1 are also connected to the corresponding pins of the ATMEGA USB-to-TTL Serial chip.
- **External Interrupts: 2 (interrupt 0), 3 (interrupt 1), 18 (interrupt 5), 19 (interrupt 4), 20 (interrupt 3), and 21 (interrupt 2).** These pins can be configured to trigger an interrupt on a low value, a rising or falling edge, or a changing value. See the attach Interrupt () function for details.
- **PWM: 0to13.** Provide 8-bit PWM output with the analogWrite() function.
- **SPI: 50 (MISO), 51 (MOSI), 52 (SCK), 53 (SS).** These pins support SPI communication, which, although provided by the underlying hardware, is not currently included in the Arduino language. The SPI pins are also broken out on the ICSP header, which is physically compatible with the Duemilanove and Diecimila.
- **LED: 13.** There is a built-in LED connected to digital pin 13. When the pin is HIGH value, the LED is on, when the pin is LOW, it's off.
- **I^2C: 20 (SDA) and 21 (SCL).** Support I^2C (TWI) communication using the Wire library (documentation on the Wiring website). Note that these pins are not in the same location as the I^2C pins on the Duemilanove.

The Arduino UNO has 16 analog inputs, each of which provide 10 bits of resolution (i.e., 1024 different values). By default, they measure from ground to

5 volts, though is it possible to change the upper end of their range using the AREF pin and analog Reference () function.

There are a couple of other pins on the board:

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

5.4.2 Communication:

The Arduino UNO has a number of facilities for communicating with a computer, another Arduino, or other microcontrollers. The Arduino UNO provides four hardware UARTs for TTL (5V) serial communication.

An ATMEGA on the board channels one of these over USB and provides a virtual comport to software on the computer (Windows machines will need a .inf file, but OSX and Linux machines will recognize the board as a COM port automatically. The Arduino software includes a serial monitor which allows simple textual data to be sent to and from the board. The RX and TX LEDs on the board will flash when data is being transmitted via the ATmega8U2 chip and USB connection to the computer (but not for serial communication on pins 0 and1).

A Software Serial library allows for serial communication on any of the digital pins.

The Arduino UNO also supports I2C (TWI) and SPI communication. The Arduino software includes a Wire library to simplify use of the I2C bus; see the documentation on the Wiring website for details. To use the SPI communication, please see the Arduino UNO datasheet.

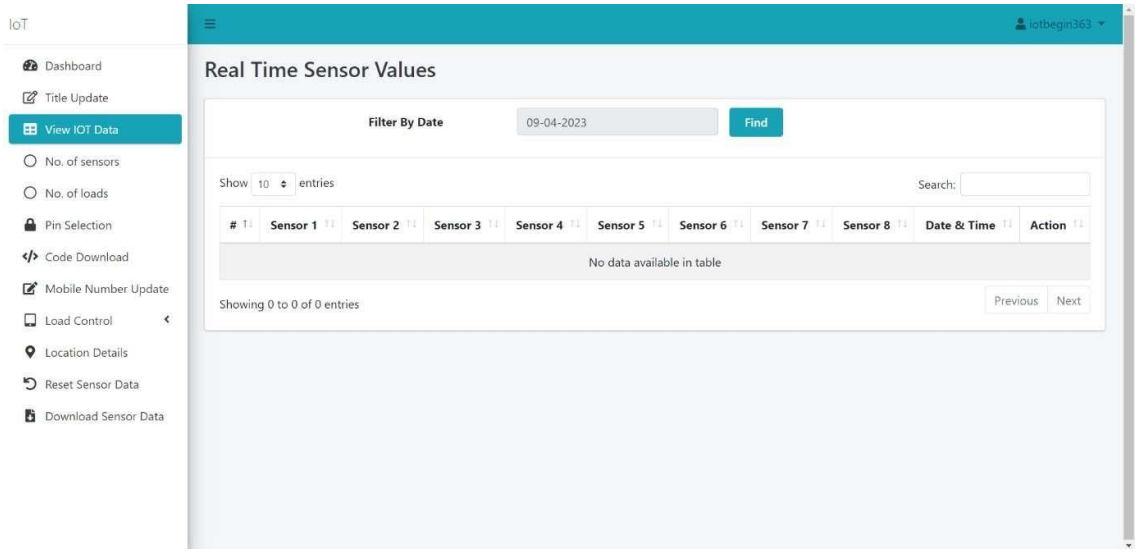


Figure.5.3 Real time sensor value

5.4.3 Programming:

The Arduino UNO can be programmed with the Arduino software (download). For details, see the reference and tutorials.

The Arduino UNO on the Arduino UNO comes preburned with a bootloader that allows you to upload new code to it without the use of an external hardware programmer. It communicates using the original STK500 protocol (reference, C header files).

You can also bypass the bootloader and program the microcontroller through the ICSP (In-Circuit Serial Programming) header; see these instructions for details.

5.5 AUTOMATIC (SOFTWARE) RESET:

Rather than requiring a physical press of the reset button before an upload, the Arduino UNO is designed in a way that allows it to be reset by software running on a connected computer. One of the hardware flow control lines (DTR) of the ATmega8U2 is connected to the reset line of the Arduino UNO via a 100 nanofarad capacitor. When this line is asserted (taken low), the reset line drops long enough to reset the chip. The Arduino software uses this capability to allow you to upload code by simply pressing the upload button in the Arduino environment. This means that the bootloader can have a shorter timeout, as the lowering of DTR can be well-coordinated with the start of the upload.

This setup has other implications. When the Arduino UNO is connected toeither a computer running Mac OS X or Linux, it resets each time a connection ismade to it from software (via USB). For the following half-second or so, the bootloader is running on the UNO. While it is programmed to ignore malformed data (i.e., anything besides an upload of new code), it will intercept the first few bytes of data sent to the board after a connection is opened. If a sketch running on the board receives one-time configuration or other data when it first starts, make sure that the software with which it communicates waits a second after opening the connection and before sending this data.

The Mega contains a trace that can be cut to disable the auto-reset. The pads on either side of the trace can be soldered together to re-enable it. It's labeled "RESET-EN". You may also be able to disable the auto-reset by connecting 110 ohm resistor from 5V to the reset line; see this forum thread for details.

5.5.1 Physical Characteristics and Shield Compatibility:

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

The maximum length and width of the UNO PCB are 4 and 2.1 inches respectively, with the USB connector and power jack extending beyond the former dimension. Three screw holes allow the board to be attached to a surface or case. Note that the distance between digital pins 7 and 8 is 160 mil (0.16"), not an even multiple of the 100-mil spacing of the other pins.

The UNO is designed to be compatible with most shields designed for the Diecimila or Duemilanove. Digital pins 0 to 13 (and the adjacent AREF and GND pins), analog inputs 0 to 5, the power header, and ICSP header are all in equivalent locations. Further the main UART (serial port) is located on the same pins (0 and 1), as are external interrupts 0 and 1 (pins 2 and 3 respectively). SPI is available through the ICSP header on both the Mega and Duemilanove / Diecimila.

Please note that I^2C is not located on the same pins on the Mega (20and21) as the Duemilanove / Diecimila (analog inputs 4 and 5).

5.5.2 How to use Arduino:

Arduino can sense the environment by receiving input from a variety of sensors and can affect its surroundings by controlling lights, motors, and other actuators. The microcontroller on the board is programmed using the Arduino programming language (based on Wiring) and the Arduino development environment (based on Processing). Arduino projects can be stand-alone or they

can communicate with software on running on a computer (e.g. Flash, Processing, MaxMSP).

Arduino is a cross-platform program. You'll have to follow differentinstructions for your personal OS. Check on the Arduino site for the latest instructions. *http://arduino.cc/en/Guide/HomePage*

Once you have downloaded/unzipped the Arduino IDE, you can plug the Arduinoto your PC via USB cable.

5.6 EMBEDDED C:

5.6.1 About Embedded C:

High-level language programming has long been in use for embedded- systems development. However, assembly programming still prevails, particularly for digital-signal processor (DSP) based systems. DSPs are often programmed in assembly language by programmers who know the processor architecture inside out. The key motivation for this practice is performance, despite the disadvantages of assembly programming when compared to high- level language programming.

If the video decoding takes 80 percent of the CPU-cycle budget instead of 90 percent, for instance, there are twice as many cycles available for audio processing. This coupling of performance to end-user features is characteristic of many of the real-time applications in which DSP processors are applied. DSPs have a highly specialized architecture to achieve the performance requirements for signal processing applications within the limits of cost and power consumption set for consumer applications. Unlike a conventional Load-Store (RISC) architecture, DSPs have a data path with memory-access units that directly feed into the arithmetic units. Address registers are taken out of the

general-purpose register file and placed next to the memory units in a separate register file.

A further specialization of the data path is the coupling of multiplication and addition to form a single cycle Multiply-accumulate unit (MAC). It is combined with special-purpose accumulator registers, which are separate from the general-purpose registers. Data memory is segmented and placed close to the MAC to achieve the high bandwidths required to keep up with the streamlined data path. Limits are often placed on the extent of memory-addressing operations. The localization of resources in the data path saves many data movements that typically take place in a Load-Store architecture.

The most important, common arithmetic extension to DSP architectures is the handling of saturated fixed-point operations by the arithmetic unit. Fixed-point arithmetic can be implemented with little additional cost over integer arithmetic. Automatic saturation (or clipping) significantly reduces the number of control-flow instructions needed for checking overflow explicitly in the program. Changes in technological and economic requirements make it more expensive to continue programming DSPs in assembly. Staying with the mobile phone as an example, the signal-processing algorithms required become increasingly complex. Features such as stronger error correction and encryption must be added. Communication protocols become more sophisticated and require much more code to implement. In certain markets, multiple protocol stacks are implemented to be compatible with multiple service providers. In addition, backward compatibility with older protocols is needed to stay synchronized with provider networks that are in a slow process of upgrading.

Today, most embedded processors are offered with C compilers. Despite this, programming DSPs is still done in assembly for the signal processing parts or, at best, by using assembly-written libraries supplied by manufacturers.

The key reason for this is that although the architecture is well matched to the requirements of the signal-processing application, there is no way to express the algorithms efficiently and in a natural way in Standard C. Saturated arithmetic. For example, is required in many algorithms and is supplied as a primitive in many DSPs. However, there is no such primitive in Standard C. To express saturated arithmetic in C requires comparisons, conditional statements, and correcting assignments. Instead of using a primitive, the operation is spread over a number of statements that are difficult to recognize as a single primitive by a compiler.

5.6.2 Description:

Embedded C is designed to bridge the performance mismatch between Standard C and the embedded hardware and application architecture. It extends the C language with the primitives that are needed by signal-processing applications and that are commonly provided by DSP processors. The design of the support for fixed-point data types and named address spaces in Embedded C is based on DSP-C. DSP-C [1] is an industry-designed extension of C with which experience was gained since 1998 by various DSP manufacturers in their compilers. For the development of DSP-C by ACE (the company three of us work for), cooperation was sought with embedded-application designers and DSP manufacturers.

The Embedded C specification extends the C language to support freestanding embedded processors in exploiting the multiple address space functionality, user-defined named address spaces, and direct access to processor and I/O registers. These features are common for the small, embedded processors used in most consumer products. The features introduced by Embedded-C are fixed-point and saturated arithmetic, segmented memory spaces, and hardware I/O addressing. The description we present here addresses the extensions from a language-design perspective, as opposed to the programmer or processor architecture perspective.

5.6.3 Multiple Address Spaces:

Embedded C supports the multiple address spaces found in most embedded systems. It provides a formal mechanism for C applications to directly access (or map onto) those individual processor instructions that are designed for optimal memory access. Named address spaces use a single, simple approach to grouping memory locations into functional groups to support MAC buffers in DSP applications, physical separate memory spaces, direct access to processor registers, and user-defined address spaces.

The Embedded C extension supports defining both the natural multiple address space built into a processor's architecture and the application-specific address space that can help define the solution to a problem. Embedded C uses address space qualifiers to identify specific memory spaces in variable declarations. There are no predefined keywords for this, as the actual memory segmentation is left to the implementation. As an example, assume that **X** and **Y** are memory qualifiers. The definition:

```
X int a[25] ;
```

Means that **a** is an array of 25 integers, which is located in the **X** memory.
Similarly (but less common):

```
X int * Y p ;
```

Means that the pointer **p** is stored in the **Y** memory. This pointer points to integer data that is located in the **X** memory. If no memory qualifiers are used, the data is stored into unqualified memory. For proper integration with the C language, a memory structure is specified, where the unqualified memory encompasses all other memories. All unqualified pointers are pointers into this unqualified memory. The unqualified memory abstraction is needed to keep the compatibility

of the **void *** type, the **NULL** pointer, and to avoid duplication of all library code that accesses memory through pointers that are passed as parameters.

5.6.4 Named Registers:

Embedded C allows direct access to processor registers that are not addressable in any of the machine's address spaces. The processor registers are defined by the compiler-specific, named-register, storage class for each supported processor. The processor registers are declared and used like conventional C variables (in many cases volatile variables). Developers using Embedded C can now develop their applications, including direct access to the condition code register and other processor-specific status flags, in a high-level language, instead of inline assembly code.

Named address spaces and full processor access reduces application dependency on assembly code and shifts the responsibility for computing data types, array and structure offsets, and all those things that C compilers routinely and easily do from developers to compilers.

5.6.5 I/O Hardware Addressing:

The motivation to include primitives for I/O hardware addressing in Embedded C is to improve the portability of device-driver code. In principle, a hardware device driver should only be concerned with the device itself. The driver operates on the device through device registers, which are device specific. However, the method to access these registers can be very different on different systems, even though it is the same device that is connected. The I/O hardware access primitives aim to create a layer that abstracts the system-specific access method from the device that is accessed. The ultimate goal is to allow source-code portability of device drivers between different systems. In the design of the I/O hardware-addressing interface, three requirements needed to be fulfilled:

1. The device-drive source code must be portable.
2. The interface must not prevent implementations from producing machine code that is as efficient as other methods.
3. The design should permit encapsulation of the system-dependent access method.

The design is based on a small collection of functions that are specified in the <iohw.h> include file. These interfaces are divided into two groups; one group provides access to the device, and the second group maintains the access method abstraction itself.

To access the device, the following functions are defined by Embedded C:

```
unsigned int iord( ioreg_designator );
void iowr( ioreg_designator, unsigned int value );
void ioor( ioreg_designator, unsigned int value );
void ioand( ioreg_designator, unsigned int value );
void ioxor( ioreg_designator, unsigned int value );
```

These interfaces provide read/write access to device registers, as well as typical methods for setting/resetting individual bits. Variants of these functions are defined (with **buf** appended to the names) to access arrays of registers. Variants are also defined (with l appended) to operate with **long** values.

All of these interfaces take an I/O register designator **ioreg_designator** as one of the arguments. These register designators are an abstraction of the real registers provided by the system implementation and hide the access method from the driver source code. Three functions are defined for managing the I/O register designators. Although these are abstract entities for the device driver, the driver does have the obligation to initialize and release the access methods. These functions do not access or initialize the device itself because that is the task of the

driver. They allow, for example, the operating system to provide a memory mapping of the device in the user address space.

```
void iogroup_acquire( iogrp_designator );
void iogroup_release( iogrp_designator );
void iogroup_map( iogrp_designator, iogrp_designator );
```

The **iogrp_designator** specifies a logical group of I/O register designators; typically this will be all the registers of one device. Like the I/O register designator, the I/O group designator is an identifier or macro that is provided by the system implementation.The map variant allows cloning of an access method when one device driver is to be used to access multiple identical devices.

5.6.6 Embedded C Portability:

By design, a number of properties in Embedded C are left implementation defined. This implies that the portability of Embedded C programs is not always guaranteed. Embedded C provides access to the performance features of DSPs. As not all processors are equal, not all Embedded C implementations can be equal For example, suppose an application requires 24-bit fixed-point arithmetic and an Embedded C implementation provides only 16 bits because that is the native size of the processor. When the algorithm is expressed in Embedded C, it will not produce outputs of the right precision. In such a case, there is a mismatch between the requirements of the application and the capabilities of the processor. Under no circumstances, including the use of assembly, will the algorithm run efficiently on such a processor. Embedded C cannot overcome such discrepancies. Yet, Embedded C provides a great improvement in the portability and software engineering of embedded applications. Despite many differences between performance-specific processors, there is a remarkable similarity in the special-

purpose features that they provide to speed up applications. Writing C code with the low-level processor-specific support may at first appear to have many of the portability problems usually associated with assembly code. In the limited experience with porting applications that use Embedded C extensions, an automotive engine controller application (about 8000 lines of source) was ported from the eTPU, a 24-bit special-purpose processor, to a general-purpose 8-bit Freescale 68S08 with about a screen full of definitions put into a single header file. The porting process was much easier than expected. For example, variables that had been implemented on the processor registers were ported to unqualified memory in the general-purpose microprocessor by changing the definitions in the header definition and without any actual code modifications. The exercise was to identify the porting issues and it is clear that the performance of the special-purpose processor is significantly higher than the general-purpose target.

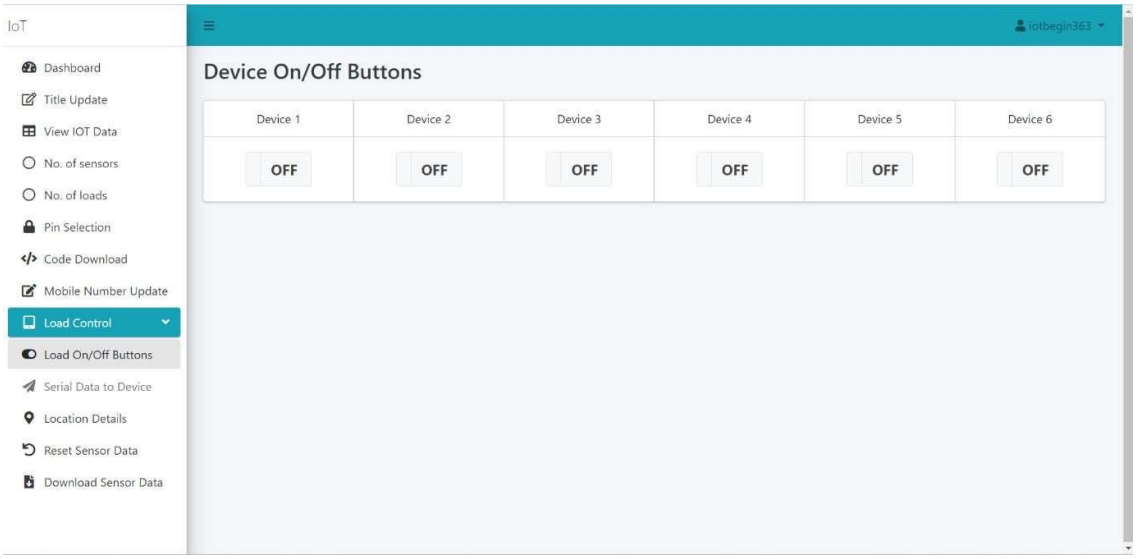


Figure.5.4 Device control (ON/OFF)

APPLICATIONS

Public Spaces: The system can be deployed in public spaces such as parks, streets, and public plazas to manage waste effectively and maintain cleanliness, reducing the risk of diseases and improving public health.

Urban Areas: The system can be implemented in urban areas with high population density to optimize waste collection routes and schedules, reducing operational costs and improving efficiency.

Smart Cities: The system aligns with the concept of smart cities, where IoT-enabled technologies are utilized to enhance the quality of life for citizens by providing efficient and sustainable solutions for waste management.

Tourist Attractions: The system can be installed at popular tourist attractions to manage waste generated by visitors, ensuring cleanliness and maintaining a positive visitor experience.

Residential Areas: The system can be implemented in residential areas to optimize waste collection schedules and routes, reducing unnecessary pickups and improving operational efficiency for waste management companies.

Industrial Areas: The system can be utilized in industrial areas to monitor waste generated by factories and warehouses, ensuring proper disposal and compliance with environmental regulations.

Educational Institutions: The system can be deployed in schools, colleges, and universities to manage waste generated by students and staff, promoting environmental awareness and sustainability practices.

FUTURE SCOPE

Expansion to other regions/countries: If our project is successful in addressing the garbage management issue in the current target region or country, there may be opportunities to expand the project to other regions or countries facing similar challenges. This could involve adapting the software to local waste management practices, regulations, and cultural norms, and scaling up the solution to a larger population.

Integration with emerging technologies: With the rapid advancement of technologies such as artificial intelligence, Internet of Things (IoT), and robotics, there may be opportunities to integrate our software with these emerging technologies. For example, we could explore the use of machine learning algorithms to optimize garbage collection routes, use IoT sensors to monitor waste levels in real-time, or leverage robotics for automated garbage collection and sorting.

Enhancements based on user feedback: As our project gets implemented and used by the end-users, we may receive feedback from them regarding its effectiveness, usability, and potential areas for improvement. This feedback can be valuable in identifying areas where our software can be enhanced, such as user interface improvements, feature additions, or performance optimizations. Incorporating user feedback into future iterations of the software can help us continuously improve and meet the evolving needs of our users.

Collaboration with stakeholders: Collaboration with stakeholders such as local government bodies, waste management agencies, environmental organizations, and other relevant entities can provide opportunities for joint efforts to address the garbage management issue. This could involve partnerships for data sharing, joint initiatives for awareness campaigns, or collaboration for policy advocacy.

Collaborative efforts can help amplify the impact of our project and create a more comprehensive and sustainable solution.

Integration with existing waste management systems: Our software can be integrated with existing waste management systems to improve their efficiency and effectiveness. For example, integrating our software with waste collection trucks or waste processing plants can enable real-time monitoring and optimization of waste collection routes, resource allocation, and waste processing operations. This can result in cost savings, improved operational efficiency, and better environmental outcomes.

Customization for different sectors: Our project can also explore the customization of the software for different sectors, such as residential, commercial, industrial, or institutional sectors. Each sector may have unique waste management requirements and challenges, and tailoring our software to their specific needs can result in a more targeted and effective solution.

Data-driven insights and analytics: As our software collects and processes data related to waste generation, collection, and disposal, there may be opportunities to derive insights and analytics from the data. This can help identify patterns, trends, and areas for improvement in waste management practices, and inform decision-making by relevant stakeholders.

LIMITATIONS

Initial Cost: The implementation of the smart garbage alert system may require a significant upfront cost for hardware, software, and infrastructure setup, which could be a limitation for smaller organizations or municipalities with limited budgets.

Maintenance and Upgrades: The system may require ongoing maintenance and upgrades to ensure its optimal performance, which could add to the operational costs over time.

Reliance on Technology: The system relies on technology components such as ultrasonic sensors, RFID tags, and IoT connectivity, which could be susceptible to technical failures, environmental factors, or cyber-attacks, impacting the system's reliability and effectiveness.

Adoption and User Acceptance: The successful implementation of the system relies on the willingness of the end-users, such as municipal workers, to adopt and utilize the technology effectively. User training and change management may be required to ensure smooth adoption and acceptance of the system.

Connectivity: The system relies on a stable internet connection for real-time data transmission and communication with the web server, which could be a limitation in areas with poor internet connectivity or during network outages.

Environmental Factors: The performance of the ultrasonic sensors used for garbage level detection could be affected by environmental factors such as extreme weather conditions, humidity, or dust, which may impact the accuracy of the system.

Scalability: The system may face challenges in scaling up to accommodate a larger number of garbage bins or expanding to cover a wider geographical area, requiring additional resources and infrastructure.

Privacy and Security: The system may collect and transmit data related to garbage levels and cleaning activities, which raises concerns about privacy and security of the data. Robust data protection measures and privacy policies need to be in place to mitigate these risks.

CONCLUSION

Our main aim is to design and implement for autonomous movement along with trash or waste collection by means of suction inside a small room or home with obstacle avoidance. Future work includes complex environment and obstacles in mobile robot navigation.

In conclusion, the introduction of garbage collection robots in India presents a promising solution to the issue of garbage pollution in both urban and rural areas. These robots have the potential to save time, effort, and resources in waste management, making it more convenient for people with busy lifestyles. With proper implementation, these technologies can contribute to the goals of the Swachh Bharath Abhiyan and help address the garbage crisis in India.

However, it is important to recognize that the success of these robots will depend on various factors such as adequate infrastructure, training, maintenance, and public awareness. The government, municipal bodies, and relevant stakeholders need to invest in the adoption and integration of these technologies, along with proper planning and execution.

Public participation and awareness are crucial in ensuring responsible waste management practices, including segregating waste at the source. Education programs and campaigns should be carried out to promote a culture of responsible waste disposal among the general population.

Overall, garbage collection robots offer a promising solution to the challenge of waste management in India, but their success will depend on comprehensive planning, effective implementation, and sustained efforts towards creating a cleaner and healthier environment for all. By leveraging technology and collective efforts, we can address the garbage crisis and contribute to a more sustainable future.

REFERENCE

[1] Krishna Nirde, Prashant S. Mulay, Uttam M. Chaskar , “Iot Based Solid Waste Management System For Smart City ”. ICICCS 2017.

[2] Balamurugan S, Abhishek Ajith, Snehal Ratnakaran, S. Balaji, R. Marimuthu, “Design of Smart Waste Management System”. IEEE 2017. Dr. N. Sathish Kumar, B.Vijayalaksmi, R. Jenifer Prarthana, A. Shankar, “IOT based Smart Garbage alert system using Arduino UNO ”. IEEE 2016.

[3] Vincenzo Catania, Daniela Ventura, “An Approch[sic] for Monitoring and Smart Planning of Urban Solid Waste Management Using Smart-M3 Platform ”. 15th Conference of Fruct Association.

[4] Sateesh Reddy Avutu, Dinesh Bhatia, B. Venkateshwara Reddy, “Design of Touch Screen Based Robot with Obstacle Detection Module for Autonomous Path Navigation”. 2017 IEEE Region 10 Conference.

[5] J.Baca et al., “Modular Robot Locomotion based on a Distributed Fuzzy Controller: The Combination of ModRED’s Basic Module Motions”, 2013 IEEE/RSJ International Conference on Intelligent Robot and Systems (IROS) November 3-7, 2013. Tokyo, Japan

[6] P.Moubarak, P.Ben-tzvi, “Modular and Reconfigurable Mobile Robotics”, Robotics and Autonomous System 60(2012) 1648-1663

[7] Wei, H. Youdong, C. Tan, J. “Sambot: A Self- Assembly Modular Robot System”, IEEE/ASME Transactions on Mechatronics, vol. 16, No. 4, August 2011

[8] K. Passino, S. Yurkovich, “Fuzzy Control”, Addison Wesley Longman Inc, 1998

[9] Jiang H., Peng Q., Lee Ha., C Teoh El., Sng HL, “Color Vision and Robot/Ball Identification for a Large Field Soccer Robot System”, 2nd International Conference on Autonomous Robots and Agents, December 13-15 2004, Palmerson North, New Zealand

[10] S.Huang, S.WU, “Vision-based Robotic Motion Control for Non-Autonomous Environment”, Proceedings of the European Control Conference 2007, Kos, Greece, July 2-5, 2007

[11] S. V. Kumar, T. S. Kumaran, A. K. Kumar and M. Mathapati, "Smart garbage monitoring and clearance system using internet of things," 2017 IEEE International Conference on Smart Technologies and Management for Computing, Communication, Controls, Energy and Materials (ICSTM), Chennai, 2017, pp. 184-189.

[12] S. Thakker and R. Narayanamoorthi, "Smart and wireless waste management," 2015 International Conference on Innovations in Information, Embedded and Communication Systems (ICIIECS), Coimbatore, 2015, pp. 1-4.

[13] M. Arebey, M. A. Hannan, H. Basri and H. Abdullah, "Solid waste monitoring and management using RFID, GIS and GSM," 2009 IEEE Student Conference on Research and Development (SCORED), UPM Serdang, 2009, pp. 37-40.

[14] M. Hannan, M. Arebey, R. A. Begum, and H. Basri, "Radio Frequency Identification (RFID) and communication technologies for solid waste bin and truck monitoring Fig.6: Text message received by the authorized mobile Fig.5: Text message received by the authorized mobile 4 system", Waste Management, Vol. 31, pp. 2406-2413, 2011.

[15] J. Joshi et al., "Cloud computing based smart garbage monitoring system," 2016 3rd International Conference on Electronic Design (ICED), Phuket, 2016, pp. 70-75

APPENDIX

Code for the prototype:

WASTE MANAGEMENT ROBOT:

```
#include <HTTPClient.h>
#include <WiFi.h>
#include <ArduinoJson.h>
#include<LiquidCrystal.h>
const int rs = 5, en = 18, d4 = 19, d5 = 21, d6 = 22, d7 = 23;
LiquidCrystal lcd(rs, en, d4, d5, d6, d7);
int W,S;
float V;
String N;
char a[14];
int count;
int r1      = 2;
int r2      = 4;
int r3      = 13;
int r4      = 12;
int BUZZER = 15;
int TRIGpin1 = 32;
int ECHOpin1 = 33;
int TRIGpin2 = 25;
int ECHOpin2 = 26;
long duration1;
int distance1;
```

```
long duration2;
int distance2;
char received;
String sensor1_status;
String sensor2_status;
String sensor3_status;
String sensor4_status;
String sensor5_status;
String sensor6_status;
String sensor7_status;
String sensor8_status;
String sms_status;

void setup()
{
  Serial.begin(9600);
  Serial2.begin(9600);
  lcd.begin(16,2);
  pinMode(TRIGpin1,OUTPUT);
  pinMode(ECHOpin1, INPUT);
  pinMode(TRIGpin2,OUTPUT);
  pinMode(ECHOpin2, INPUT);
  pinMode(BUZZER ,OUTPUT);
  pinMode(r1, OUTPUT);
  pinMode(r2, OUTPUT);
  pinMode(r3, OUTPUT);
  pinMode(r4, OUTPUT);
```

```
 digitalWrite(BUZZER,LOW);
 digitalWrite(r1,LOW);
 digitalWrite(r2,LOW);
 digitalWrite(r3,LOW);
 digitalWrite(r4,LOW);              //Serial connection
// Serial.println("iotbegin363");
 WiFi.begin("iotbegin363", "iotbegin363");    //WiFi connection
 while (WiFi.status() != WL_CONNECTED)
 { //Wait for the WiFI connection completion
 // Serial.println("Waiting for Wi-Fi connection");
  lcd.setCursor(0, 0);
  lcd.print("connecting to ");
  lcd.setCursor(0, 1);
  lcd.print("iotbegin363 ");
 }
 lcd.clear();
 lcd.print("Wi-Fi connected ");
 delay(1000);
 lcd.setCursor(0, 0);
 lcd.print(" SMART DUSTBIN ");
 lcd.setCursor(0, 1);
 lcd.print("     ROBOT      ");
 delay(3000);
 lcd.clear();
}
void loop()
{
```

```
if(Serial2.available()>0)
 {
  count=0;
  while(Serial2.available() && count<14)
  {
    a[count]=Serial2.read();
    count++;
    delay(5);
  }
 }
 //5200140DB8F3
 if(a[7]=='D')
 {
   digitalWrite(r1,HIGH);
    digitalWrite(r2,LOW);
    digitalWrite(r3,HIGH);
    digitalWrite(r4,LOW);
    delay(5000);
    digitalWrite(r1,LOW);
    digitalWrite(r2,LOW);
    digitalWrite(r3,LOW);
    digitalWrite(r4,LOW);
    digitalWrite(BUZZER,HIGH);
    delay(3000);
    digitalWrite(BUZZER,LOW);
    Serial.print('b');
    a[7]='*';
```

```
 }
  digitalWrite(TRIGpin1, LOW);
  delayMicroseconds(4);
  digitalWrite(TRIGpin1, HIGH);
  delayMicroseconds(15);
  digitalWrite(TRIGpin1, LOW);
  digitalWrite(TRIGpin2, LOW);
  duration1 = pulseIn(ECHOpin1, HIGH);
  digitalWrite(TRIGpin2, LOW);
  delayMicroseconds(4);
  digitalWrite(TRIGpin2, HIGH);
  delayMicroseconds(15);
  digitalWrite(TRIGpin2, LOW);
  duration2 = pulseIn(ECHOpin2, HIGH);
  distance1 = duration1*(0.034/2);
  distance2 = duration2*(0.034/2);
  lcd.setCursor(0,0);
  lcd.print("Obj.Dis=");
  lcd.print(distance1);
  lcd.print("         ");
  lcd.setCursor(0,1);
lcd.print("DUSTBIN=");
lcd.print(distance2);
lcd.print("         ");
if(distance2<=10)
{
   sensor3_status="DUSTBIN FILLED ";
```

```
   sms_status="1";
}
else
{
  sensor3_status="                ";
   sms_status="0";
}
  sensor1_status="Obj.Dis="+String(distance1);
  sensor2_status="DUSTBIN="+String(distance2);
  iot();
}
void iot()
{
  if (WiFi.status() == WL_CONNECTED)
  {
    DynamicJsonDocument jsonBuffer(JSON_OBJECT_SIZE(3) + 300);
    JsonObject root = jsonBuffer.to<JsonObject>();
    root["sensor1"] = sensor1_status;
    root["sensor2"] = sensor2_status;
    root["sensor3"] = sensor3_status;
    root["sensor4"] = sensor4_status;
    root["sensor5"] = sensor5_status;
    root["sensor4"] = sensor6_status;
    root["sensor5"] = sensor7_status;
    root["sensor4"] = sensor8_status;
    root["sms"]     = sms_status;
    String json;
```

```
serializeJson(jsonBuffer, json);
if (sensor1_status != "null")
{
  HTTPClient http;    //Declare object of class HTTPClient
  http.addHeader("username", "iotbegin363"); //Specify content-type header
  http.addHeader("Content-Type", "application/json");
  int httpCode = http.POST(json);   //Send the request
  String payload = http.getString();   //Get the response payload
  http.end(); //Close connection
}
HTTPClient http;    //Declare object of class HTTPClient
http.addHeader("username", "iotbegin363"); //Specify content-type header
int httpCode = http.GET();
String payload = http.getString();                 //Get the response payload
//Serial.println(httpCode);  //Print HTTP return code
//Serial.println(payload);    //Print request response payload
http.end(); //Close connection
StaticJsonDocument<300> parseload;
deserializeJson(parseload, payload);
//Serial.println();
JsonObject load_data = parseload["data"];
String device1_status = load_data["device1"];
String device2_status = load_data["device2"];
String device3_status = load_data["device3"];
String device4_status = load_data["device4"];
String device5_status = load_data["device5"];
String device6_status = load_data["device6"];
```

```
   if((device1_status == "1")&&(device2_status == "0")&&(device3_status ==
"0")&&(device4_status == "0"))
   {
   digitalWrite(r1,HIGH);
   digitalWrite(r2,LOW);
   digitalWrite(r3,HIGH);
   digitalWrite(r4,LOW);
   }
   else if((device1_status == "0")&&(device2_status == "1")&&(device3_status
== "0")&&(device4_status == "0"))
   {
    digitalWrite(r1,LOW);
    digitalWrite(r2,HIGH);
    digitalWrite(r3,LOW);
    digitalWrite(r4,HIGH);
   }
   else if((device1_status == "0")&&(device2_status ==
"0")&&(device3_status == "1")&&(device4_status == "0"))
   {
    digitalWrite(r1,LOW);
    digitalWrite(r2,HIGH);
    digitalWrite(r3,HIGH);
    digitalWrite(r4,LOW);
   }
   else if((device1_status == "0")&&(device2_status == "0")&&(device3_status
== "0")&&(device4_status == "1"))
   {
    digitalWrite(r1,HIGH);
```

```
    digitalWrite(r2,LOW);
    digitalWrite(r3,LOW);
    digitalWrite(r4,HIGH);
   }
   else
   {
    digitalWrite(r1,LOW);
    digitalWrite(r2,LOW);
    digitalWrite(r3,LOW);
    digitalWrite(r4,LOW);
   }
   sms_status="0";
  }
  else
  {
   //Serial.println("Error in WiFi connection");
  }
}
```

CODE FOR HOME DUSTBIN:

```
#include<LiquidCrystal.h>
const int rs = 5, en = 18, d4 = 19, d5 = 21, d6 = 22, d7 = 23;
LiquidCrystal lcd(5, 18, 19, 21, 22, 23);
int R1=33;
int R2=32;
int R3=12;
int R4=13;
```

```
int R5=25;
int R6=26;
char rec;
void setup()
{
  lcd.begin(16, 2);
  Serial.begin(9600);
  lcd.setCursor(0,0);
  lcd.print("HOME DUSTBIN");
  lcd.setCursor(0,1);
  lcd.print("   ROBOT    ");
  pinMode(R1, OUTPUT);
  pinMode(R2, OUTPUT);
  pinMode(R3, OUTPUT);
  pinMode(R4, OUTPUT);
  pinMode(R5, OUTPUT);
  pinMode(R6, OUTPUT);
  digitalWrite(R1,LOW);
  digitalWrite(R2,LOW);
  digitalWrite(R3,LOW);
  digitalWrite(R4,LOW);
  digitalWrite(R5,LOW);
  digitalWrite(R6,LOW);
}
void loop()
```

```
{
 if(Serial.available()>0)
 {
 rec=Serial.read();
 Serial.println(rec);
 }
 lcd.setCursor(15,1);
lcd.print(rec);
lcd.print("         ");
if(rec == 'b')
{
 digitalWrite(R1,HIGH);
 digitalWrite(R2,LOW);
 digitalWrite(R3,HIGH);
 digitalWrite(R4,LOW);
 digitalWrite(R5,LOW);
 digitalWrite(R6,LOW);
 delay(3000);
 digitalWrite(R1,LOW);
 digitalWrite(R2,LOW);
 digitalWrite(R3,LOW);
 digitalWrite(R4,LOW);
 delay(2000);
 digitalWrite(R5,HIGH);
 digitalWrite(R6,LOW);
```

```
  delay(1800);
  digitalWrite(R5,LOW);
  digitalWrite(R6,LOW);
  delay(10000);
  digitalWrite(R6,HIGH);
  digitalWrite(R5,LOW);
  delay(1800);
  digitalWrite(R5,LOW);
  digitalWrite(R6,LOW);
  digitalWrite(R1,LOW);
  digitalWrite(R2,HIGH);
  digitalWrite(R3,LOW);
  digitalWrite(R4,HIGH);
  delay(3000);
  digitalWrite(R1,LOW);
  digitalWrite(R2,LOW);
  digitalWrite(R3,LOW);
  digitalWrite(R4,LOW);
  rec=' ';
 }
}
```