CHAPTER - 1
INTRODUCTION

1.1 INTRODUCTION TO EMBEDDED SYSTEMS

Each day, our lives become more dependent on 'embedded systems', digital information technology that is embedded in our environment. More than 98% of processors applied today are in embedded systems, and are no longer visible to the customer as 'computers' in the ordinary sense. An Embedded System is a special-purpose system in which the computer is completely encapsulated by or dedicated to the device or system it controls. Unlike a general-purpose computer, such as a personal computer, an embedded system performs one or a few predefined tasks, usually with very specific requirements. Since the system is dedicated to specific tasks, design engineers can optimize it, reducing the size and cost of the product. Embedded systems are often mass- produced, benefiting from economies of scale.

The increasing use of PC hardware is one of the most important developments in high-end embedded systems in recent years. Hardware costs of high-end systems have dropped dramatically as a result of this trend, making some projects feasible which previously would not have been done because of the high cost of non-PC-based embedded hardware. But software choices for the embedded PC platform are not nearly as attractive as the hardware.

Typically, an embedded system is housed on a single microprocessor board with the programs stored in ROM. Virtually all appliances that have a digital interface -- watches, microwaves, VCRs, cars -- utilize embedded systems. Some embedded systems include an operating system, but many are so specialized that the entire logic can be implemented as a single program. 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. The applications software on such processors is sometimes referred to as firmware.

1.2 DEFINITION OF EMBEDDED SYSTEMS

Embedded system is defined as, for a particular/specific application implementing the software code to interact directly with that particular hardware that we built. Software is used for providing features and flexibility.

Hardware = {Processors, ASICs, Memory...} is used for Performance (& sometimes security).There are many definitions of embedded systems but all of these can be combined into a single concept. An embedded system is a special purpose computer system that is used for a particular task.

1.3 EXISTING SYSTEM

Ias checn strong waste receptacle observing framework trash container set the public spot then Camera set for trash canister area. The camera caught a picture for trash container. Radio Frequency Identification (RFID), GPS and GIS send pictures for work stations. The RFID peruser and camera are mounted in the truck, when truck comes nearer to the receptacle RFID peruser imparts RFID tag. and sends all data. The System utilizes controlling Hut. This Controlling Hut is SMS Technology. The GPS and GPRS planning worker to dissect information of different areas. The control station accumulated all the data and put it away in the framework data set. The receptacle status and waste truck wked.

1.4 DRAWBACKS OF EXISTING SYSTEM

1. In this system proposed there was a just dustbin without any indicator to indicate the level of waste in the dustbin.
2. There is no continuous monitoring of the dustbin

1.5 PROPOSED SYSTEM

There is a need to effectively manage all of the rubbish bins that are located across the city to lower the costs of clearing the garbage bins and the personnel that is currently involved in this operation. This can be accomplished by combining cloud computing, IoT, and machine learning to provide truck drivers with a roadmap that displays the most effective way to take and collect garbage bins fast to save time and fuel. This project is helpful in the Government of India's major initiative “SWACHH BHARAT ABHIYAN”. To achieve smart city development and implementation, various technologies are to be used and this project will be helpful for waste management. The relevant data is given to the concerned authorities to collect the solid waste accumulated in garbage bins by continuously monitoring the level of solid waste existing in the garbage bins. This procedure is less expensive and requires fewer people. Because there is less travel to rubbish bins, the amount of money spent on fuel is reduced. This contributes to the cleanliness and orderliness of our surroundings.

1.6 ADVANTAGES OF PROPOSED SYSTEM

1. It makes the work easier for the municipality that they will get notification in the Blynk IOT application if the dustbin get totally filled.
2. Fast response
3. Person nearby is detected.

CHAPTER-2
PROJECT DESCRIPTION

2.1 HISTORY OF EMBEDDED SYSTEMS

The first recognizably modern embedded systems was the Apollo Guidance Computer, developed by “Charles Stark Draper” at the MIT Instrumentation Laboratory. At the project inception, the Apollo guidance computer was considered the riskiest item in the Apollo project as it employed the then newly developed monolithic integrated circuits to reduce the size and weight. An early mass produced embedded system was the Autonetics D-17 guidance computer for the Minuteman missile, released in 1961.It was built from transistor logic and had a hard disk for main memory .When the Minuteman II went into production in 1966, the D-17 was replaced with a new computer that was the first high volume use of integrated circuits. This program alone reduced prices on quad and gate ICs from $1000/ each to $3/ each permitting their use in commercial products.

Since these early applications in the 1960s, embedded systems have come down in price and there has been a dramatic rise in processing power and functionality. The first microprocessor for example, the Intel 4004, was designed for calculators and other small systems but still required many external memory and support chips. In 1978 National Engineering Manufacturers Association released a “standard” for programmable microcontrollers, including almost any computer based controllers, such as single board computers, numerical and event based controllers. Embedded Systems are designed to some specific task, rather than be a general-purpose computer for multitasks. 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 requirement, allowing the system hardware to be simplified to reduce cost.

2.2 FEATURES OF EMBEDDED SYSTEMS

The versatility of the embedded computer system lends itself to utility in all kinds of enterprises, from the simplification of deliverable products to a reduction in costs in their development and manufacture. Complex systems with rich functionality employ specialOperating systems that take into account major characteristics of embedded systems. Embedded operating systems have minimized footprint and may follow real-time operating system specifics.The special computers system is usually less powerful than general-purpose systems, although some expectations do exist where embedded systems are very powerful and complicated. Usually a low power consumption CPU with a limited amount of memory is used in embedded systems. Many embedded systems use very small operating systems; most of these provide very limited operating system capabilities.Since the embedded system is dedicated to specific tasks, design engineers can optimize it, reducing the size and cost of the product, or increasing the reliability and performance. Some embedded systems are mass-produced, benefiting from economies of scale. Some embedded systems have to operate in extreme environment conditions such as very high temperature & humidity. For high volume systems such as portable music players or mobile phones, minimizing cost is usually the primary design consideration. Engineers typically select hardware that is just “good enough” to implement the necessary functions. For low volume or prototype embedded systems, general purpose computers may be adapted by limiting the programs or by replacing the operating system with a real-time operating system.

2.3 CHARACTERISTICS OF EMBEDDED SYSTEM

Embedded computing systems generally exhibit rich functionality; complex functionality is usually the reason for introducing CPUs into the design. However, they also exhibit many non-functional requirements that make the task especially challenging:

- Real-time deadlines that will cause system failure if not met
- Multi-rate operation
- In many cases, low power consumption
- Low manufacturing cost, which often means limited code size.

Workstation programmers often concentrate on functionality. They may consider the performance characteristics of a few computational kernels of their software, but rarely analyze the total application. They almost never consider power consumption and manufacturing cost. The need to juggle all these requirements makes embedded system programming very challenging and is the reason why embedded system designers need to understand computer architecture.

2.4 OVERVIEW OF EMBEDDED SYSTEMS

Every embedded system consists of custom-built hardware built around a Central Processing Unit (CPU). This hardware also contains memory chips onto which the software is loaded. The software residing on the memory chip is also called the ‘firmware’. The embedded system architecture can be represented as a layered architecture as shown in Fig.

The same architecture is applicable to any computer including a desktop computer. However, there are significant differences. It is not compulsory to have an operating system in every embedded system. For small appliances such as remote control units, air conditioners, toys etc., there is no need for an operating system and you can write only the software specific to that application.

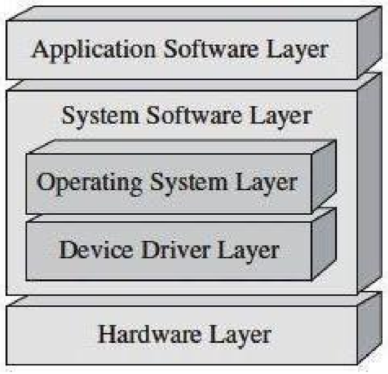


Figure 2.4 LAYERED ARCHITECTURE EMBEDDED SYSTEMS

For applications involving complex processing, it is advisable to have an operating system. In such a case, you need to integrate the application software with the operating system and then transfer the entire software on to the memory chip. Once the software is transferred to the memory chip, the software will continue to run *for* a long time. You don't need to reload new software.

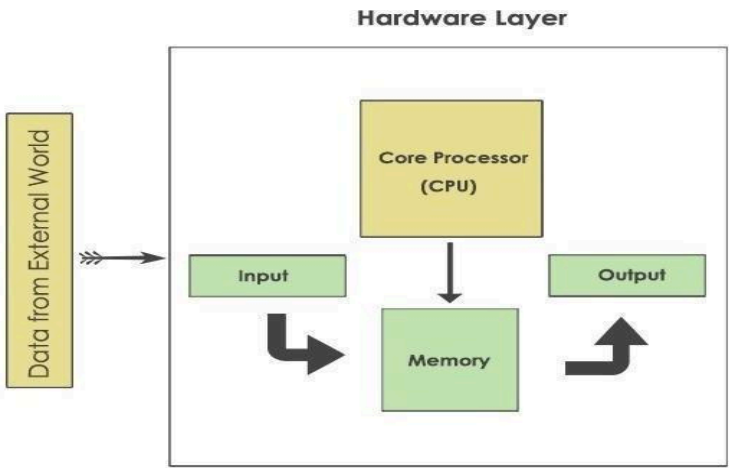


FIGURE 2.4A BLOCK DIAGRAM OF EMBEDDED SYSTEMS

Now, let us see the details of the various building blocks of the hardware of an embedded system. As shown in Fig. the building blocks are;

- Central Processing Unit (CPU)
- Memory (Read-only Memory and Random Access Memory)
- Input Devices
- Output devices
- Communication interface
- Application-specific circuitry

2.4.1 CENTRAL PROCESSING UNIT (CPU)

The Central Processing Unit (processor, in short) can be any of the following: microcontroller,microprocessor or Digital Signal Processor (DSP). A micro-controller is a low-cost processor. Its main attraction is that on the chip itself, there will be many other components such as memory, serial communication interface, analog-to-digital converter etc. So, for small applications, a microcontroller is the best choice as the number of external components required will be very less. On the other hand, microprocessors are more powerful, but you need to use many external components with them. D5P is used mainly for applications in which signal processing is involved such as audio and video processing.

2.4.2 MEMORY

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

2.4.3 INPUT DEVICES

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

2.4.4 OUTPUT DEVICES

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

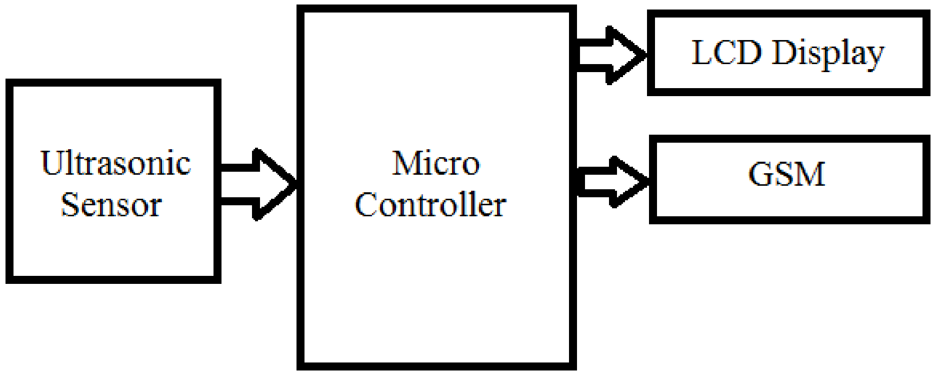
2.4.5 COMMUNICATION INTERFACE

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

2.4.6 APPLICATION OF SPECIFIC CIRCUITRY

Sensors, transducers, special processing and control circuitry may be required fat an embedded system, depending on its application. This circuitry interacts with the processor to carry out the necessary work. The entire hardware has to be given power supply either through the 230 volts main supply or through a battery. The hardware has to be designed in such a way that the power consumption is minimized.

BLOCK DIAGRAM



This IoT based Garbage Monitoring system is a very innovative system which will help to keep the cities clean. This system monitors the garbage bins and informs about the level of garbage collected in the garbage bins via an Indication. For this the system uses ultrasonic sensors placed over the bins to detect the garbage level and compare it with the garbage bins depth. The system makes use of an Arduino uno board, LCD screen, IoT modem for sending data. The system is powered by a 12V transformer. The LCD screen is used to display the status of the level of garbage collected in the bins. Whereas IoT is built to show the status to the user, monitoring it with Indication.The Indication consists of text related to all garbage bins. The LCD screen shows the status of the garbage level. The system puts on LCD screen continuous monitoring of garbage with an arduino board. Thus this system helps to keep the city clean by informing about the garbage levels of the bins by providing Indication to the respective person.

2.4.7 ADVANTAGES

system uses sensors for identification of personnel and measuring garbage level. Continuous data regarding garbage levels are available for personnel through the display of the garbage bin. This smart garbage bin provides an automated lid to personnel.

2.5 HARDWARE CONSTRAINTS

2.5.1 POWER SUPPLY

2.5.2 REGULATED POWER SUPPLY

The DC power supply is practically converted to each and every stage in an electronic system. Thus a common requirement for all these phases will be the DC power supply. All low power systems can be run with a battery. But, for a long time operating devices, batteries could prove to be costly and complicated. The best method used is in the form of an unregulated power supply –a combination of a transformer, rectifier and a filter. The diagram is shown below.

All devices will have a certain power supply limit and the electronic circuits inside these devices must be able to supply a constant DC voltage within this limit. This DC supply is regulated and limited in terms of voltage and current. But the supply provided from mains may be fluctuating and could easily break down the electronic equipment, if not properly limited. This work of converting an unregulated alternating

current (AC) or voltage to a limited Direct current (DC) or voltage to make the output constant regardless of the fluctuations in input, is done by a regulated power supply circuit.

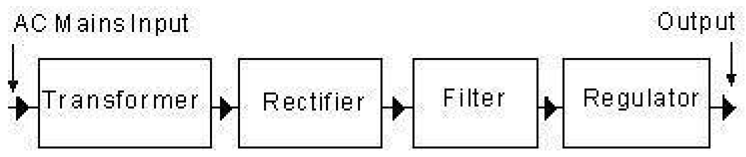


FIGURE 2.5.2 BLOCK DIAGRAM OF REGULATED POWER SUPPLY

All the active and passive electronic devices will have a certain DC operating point (Q-point or Quiescent point), and this point must be achieved by the source of DC power. A step down transformer is used to reduce the voltage level to the devices needs. In India, a 1 Ø supply is available at 230 volts. The output of the transformer is a pulsating sinusoidal AC voltage, which is converted to pulsating DC with the help of a rectifier.

This output is given to a filter circuit which reduces the AC ripples, and passes the DC components. But there are certain disadvantages in using an unregulated power supply.

Regulated power supply is an electronic circuit that is designed to provide a constant dc voltage of predetermined value across load terminals irrespective of ac mains fluctuations or load variations.

A regulated power supply essentially consists of an ordinary power supply and a voltage regulating device, as illustrated in the figure. The output from an ordinary power supply is fed to the voltage regulating device that provides the final output. The output voltage remains constant irrespective of variations in the ac input voltage or variations in output (or load) current.

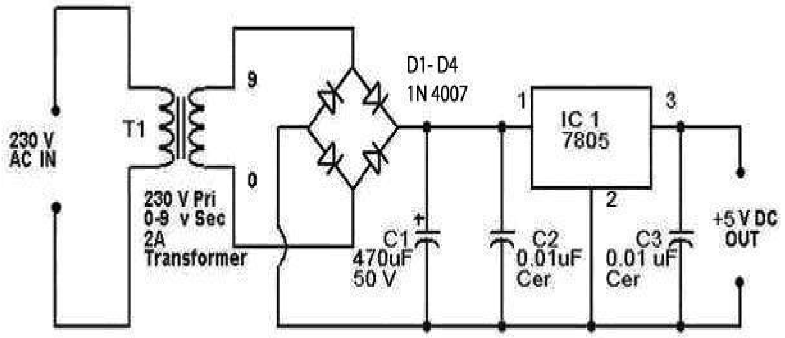


FIGURE 2.5.2A CIRCUIT DIAGRAM OF REGULATED POWER SUPPLY

2.5.3 Relay

A relay is an electrically operated switch. It consists of a set of input terminals for a single or multiple control signals, and a set of operating contact terminals. The switch may have any number of contacts in multiple contact forms, such as make contacts, break contacts, or combinations thereof.

Relays are used where it is necessary to control a circuit by an independent low-power signal, or where several circuits must be controlled by one signal. Relays were first used in long- distance telegraph circuits as signal repeaters: they refresh the signal coming in from one circuit by transmitting it on another circuit. Relays were used extensively in telephone exchanges and early computers to perform logical operations.

The traditional form of a relay uses an electromagnet to close or open the contacts, but relays using other operating principles have also been invented, such as in solid-state relays which use semiconductor properties for control without relying on moving parts. Relays with calibrated operating characteristics and sometimes multiple operating coils are used to protect electrical circuits from overload or faults; in modern electric power systems these functions are performed by digital instruments still called protective relays.

Latching relays require only a single pulse of control power to operate the switch persistently. Another pulse applied to a second set of control terminals, or a pulse with opposite polarity, resets the switch, while repeated pulses of the same kind have no effects. Magnetic latching relays are useful in applications when interrupted power should not affect the circuits that the relay is

2.6 ULTRASONIC SENSOR

An ultrasonic sensor is an electronic device that measures the distance of a target object by emitting ultrasonic sound waves and converts the reflected sound into an electrical signal.

In order to calculate the distance between the sensor measures the time it takes between the emission of the sound by transmitter to its contact with the receiver.The formula for this calculation is

$$D = \frac{1}{2} T*C$$

Where D is distance , T is the time and C is the speed of sound - 343 m/s.

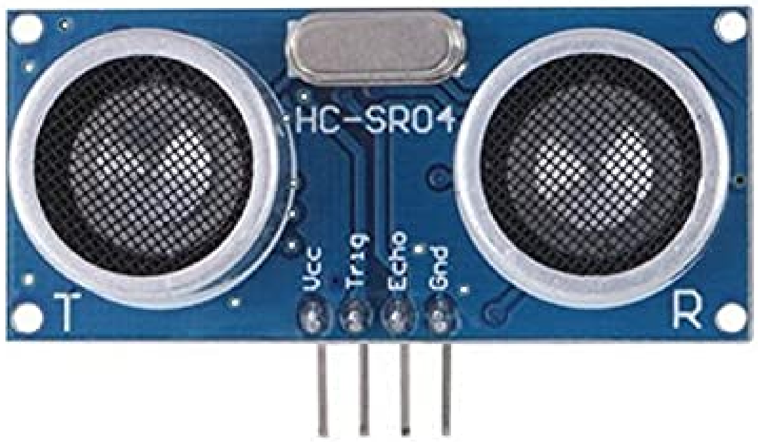


FIGURE 2.6 ULTRASONIC SENSOR

2.7 NODEMCU

The Internet of Things (IoT) has been a trending field in the world of technology. It has changed the way we work. Physical objects and the digital world are connected now more than ever. Keeping this in mind, Espressif Systems (A Shanghai-based Semiconductor Company) has released an adorable, bite-sized WiFi enabled microcontroller – ESP826**6**, at an unbelievable price! For less than $3, it can monitor and control things from anywhere in the world – perfect for just about any IoT project.

ESP-12E Module

The development board equips the ESP-12E module containing ESP8266 chip having Tensilica Xtensa® 32-bit LX106 RISC microprocessor which operates at 80 to 160 MHz adjustable clock frequency and supports RTOS.

ESP-12E Chip

- Tensilica Xtensa 32-bit LX106
- 80 to 160 MHz Clock Freq.
- 4MB external flash
- 802.11b/g/n Wi-fi transceiver

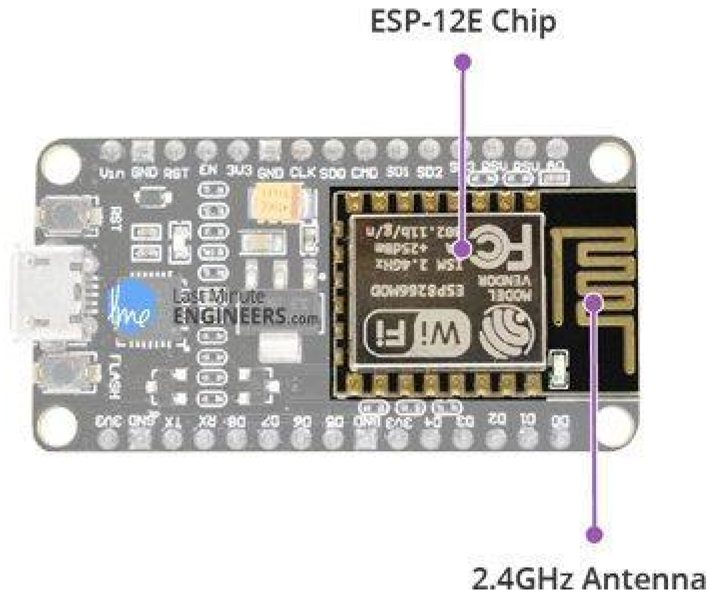


FIGURE 2.7A ESP-12E CHIP

There's also **128 KB RAM and 4MB of Flash memory** (for program and data storage) just enough to cope with the large strings that make up web pages, JSON/XML data, and everything we throw at IoT devices nowadays.

The ESP8266 Integrates **802.11b/g/n HT40 Wi-Fi transceiver**, so it can not only connect to a WiFi network and interact with the Internet, but it can also set up a network of its own, allowing other devices to connect directly to it. This makes the ESP8266 NodeMCU even more versatile.

Power Requirement

As the operating voltage range of ESP8266 is **3V to 3.6V**, the board comes with a LDO voltage regulator to keep the voltage steady at 3.3V. It can reliably supply up to 600mA, which should be more than enough when ESP8266 pulls as much as **80mA during RF transmissions**. The output of the regulator is also broken out to one of the sides of the board and labeled as 3V3. This pin can be used to supply power to external components.

Power Requirement

- Operating Voltage : 2.5 V to 3.6V
- On-board 3.3V 600mA regulator
- 80mA operating current
- 20 µA during Sleep mode

Power to the ESP8266 NodeMCU is supplied via the **on-board MicroB USB connector**. Alternatively, if you have a regulated 5V voltage source, the **VIN pin** can be used to directly supply the ESP8266 and its peripherals.

Warning:

The ESP8266 requires a 3.3V power supply and 3.3V logic levels for communication. The GPIO pins are not 5V-tolerant! If you want to interface the board with 5V (or higher) components, you'll need to do some level shifting.

The ESP8266 NodeMCU has total **17 GPIO pins** broken out to the pin headers on both sides of the development board. These pins can be assigned to all sorts of peripheral duties, including:

- **ADC channel** – A 10-bit ADC channel.
- **UART interface** – UART interface is used to load code serially.
- **PWM outputs** – PWM pins for dimming LEDs or controlling motors.
- **SPI, I2C & I2S interface** – SPI and I2C interface to hook up all sorts of sensors and peripherals.
- **I2S interface** – I2S interface if you want

Multiplexed I/Os

- 1ADC channels
- 2 UART interfaces
- 4 PWM outputs
- SPI , I2C & I2S interface

\

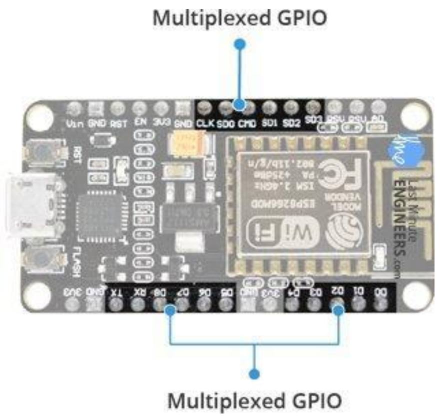


FIGURE 2.7B MULTIPLEXED GPIO OF NODEMCU

Thanks to the ESP8266's **pin multiplexing feature** (Multiple peripherals multiplexed on a single GPIO pin). Meaning a single GPIO pin can act as PWM/UART/SPI.

On-board Switches & LED Indicator

The ESP8266 NodeMCU features two buttons. One marked as **RST** located on the top left corner is the Reset button, used of course to reset the ESP8266 chip. The other **FLASH** button on the bottom left corner is the download button used while upgrading firmware.

Switches & Indicators

- RST - Reset the ESP8266 chip
- FLASH - Download new programs
- Blue LED - User programmable

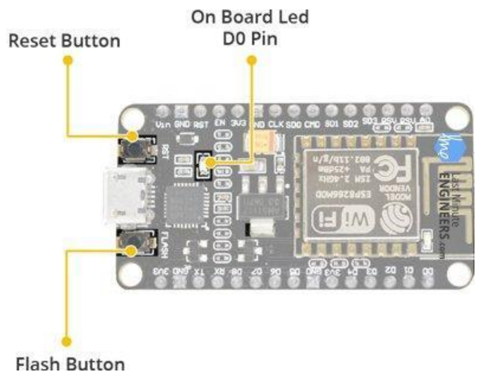


FIGURE 2.7C SWITCHES AND INDICATORS IN NODEMCU

The board also has a **LED indicator** which is user programmable and is connected to the D0 pin of the board.

Serial Communication

The board includes CP2102 USB-to-UART Bridge Controller from Silicon Labs, which converts USB signal to serial and allows your computer to program and communicate with the ESP8266 chip.

Serial Communication

- **CP21202 USB - to- UART converter**
- **4.5 Mbps communication speed**
- **Flow control support**

The ESP8266 NodeMCU has a total of 30 pins that interface it to the outside world. The connections are as follows:

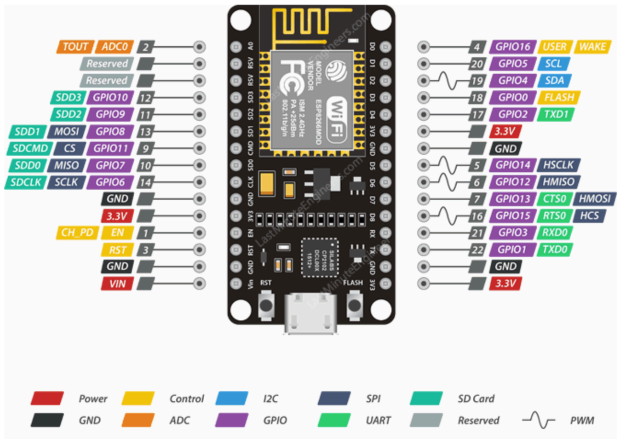


FIGURE 2.7D PIN SPECIFICATIONS OF NODEMCU

POWER PIN There are four power pins viz. one VIN pin & three 3.3V pins. The VIN pin can be used to directly supply the ESP8266 and its peripherals, if you have a regulated 5V voltage source. The 3.3V pins are the output of an on-board voltage regulator. These pins can be used to supply

I2C Pins are used to hook up all sorts of I2C sensors and peripherals in your project. Both I2C Master and I2C Slave are supported. I2C interface functionality can be realized programmatically, and the clock frequency is 100 kHz at a maximum. It should be noted that I2C clock frequency should be higher than the slowest clock frequency of the slave device.

GPIO Pins ESP8266 NodeMCU has 17 GPIO pins which can be assigned to various functions such as I2C, I2S, UART, PWM, IR Remote Control, LED Light and Button programmatically. Each digital enabled GPIO can be configured to internal pull-up or pull-down, or set to high impedance. When configured as an input, it can also be set to edge-trigger or level-trigger to generate CPU interrupts.

ADC Channel The NodeMCU is embedded with a 10-bit precision SAR ADC. The two functions can be implemented using ADC viz. Testing power supply voltage of VDD3P3 pin and testing input voltage of TOUT pin. However, they cannot be implemented at the same time.

UART Pins ESP8266 NodeMCU has 2 UART interfaces, i.e. UART0 and UART1, which provide asynchronous communication (RS232 and RS485), and can communicate at up to 4.5 Mbps. UART0 (TXD0, RXD0, RST0 & CTS0 pins) can be used for communication. It supports fluid control. However, UART1 (TXD1 pin) features only data transmit signal so, it is usually used for printing log.

SPI Pins ESP8266 features two SPIs (SPI and HSPI) in slave and master modes. These SPIs also support the following general-purpose SPI features:

- 4 timing modes of the SPI format transfer
- Up to 80 MHz and the divided clocks of 80 MHz
- Up to 64-Byte FIFO

SDIO Pins ESP8266 features Secure Digital Input/Output Interface (SDIO) which is used to directly interface SD cards. 4-bit 25 MHz SDIO v1.1 and 4-bit 50 MHz SDIO v2.0 are supported.

PWM Pins The board has 4 channels of Pulse Width Modulation (PWM). The PWM output can be implemented programmatically and used for driving digital motors and LEDs. PWM frequency range is adjustable from 1000 µs to 10000 µs, i.e., between 100 Hz and 1 kHz.

Control Pins are used to control ESP8266. These pins include Chip Enable pin (EN), Reset pin (RST) and WAKE pin.

- EN pin – The ESP8266 chip is enabled when EN pin is pulled HIGH. When pulled LOW the chip works at minimum power.
- RST pin – RST pin is used to reset the ESP8266 chip.
- WAKE pin – Wake pin is used to wake the chip from deep-sleep.

ESP8266 Development Platforms

Now, let’s move on to the interesting stuff!

There are a variety of development platforms that can be equipped to program the ESP8266. You can go with Espruino – JavaScript SDK and firmware closely emulating Node.js, or use Mongoose OS – An operating system for IoT devices (recommended platform by Espressif Systems and Google Cloud IoT) or use a software development kit (SDK) provided by Espressif or one of the platforms listed on WiKiPedia.

Fortunately, the amazing ESP8266 community took the IDE selection a step further by creating an Arduino add-on. If you're just getting started programming the ESP8266, this is the environment we recommend beginning with, and the one we'll document in this tutorial.

This ESP8266 add-on for Arduino is based on the amazing work by Ivan Grokhotkov and the rest of the ESP8266 community. Check out the ESP8266 Arduino GitHub repository for more information.

Installing the ESP8266 Core on Windows OS

Let's proceed with installing the ESP8266 Arduino core.

The first thing is having the latest Arduino IDE (Arduino 1.6.4 or higher) installed on your PC. If don't have it, we recommend upgrading now.

To begin, we'll need to update the board manager with a custom URL. Open up Arduino IDE and go to File > Preferences. Then, copy below URL into the Additional Board Manager URLs text box situated on the bottom of the window:

http://arduino.esp8266.com/stable/package_esp8266com_index.json

Hit OK. Then navigate to the Board Manager by going to Tools > Boards > Boards Manager. There should be a couple new entries in addition to the standard Arduino boards. Filter your search by typing esp8266. Click on that entry and select Install.

Once installation is completed dump the code mentioned below.Once the code is uploaded, LED will start blinking. You may need to tap the RST button to get your ESP8266 to begin running the sketch.

2.8 LIQUID CRYSTAL DISPLAY

LCD stands for Liquid Crystal Display. LCD is finding widespread use replacing LEDs (seven segment LEDs or other multi segment LEDs) because of the following reasons:

- The declining prices of LCDs.

- The ability to display numbers, characters and graphics. This is in contrast to LEDs, which are limited to numbers and a few characters.

- Incorporation of a refreshing controller into the LCD, thereby relieving the CPU of the task of refreshing the LCD. In contrast, the LED must be refreshed by the CPU to keep displaying the data.

- Ease of programming for characters and graphics.

These components are “specialized” for being used with the microcontrollers, which means that they cannot be activated by standard IC circuits. They are used for writing different messages on a miniature LCD.

A model described here is for its low price and great possibilities most frequently used in practice. It is based on the HD44780 microcontroller (Hitachi) and can display messages in two lines with 16 characters each. It displays all the alphabets, Greek letters, punctuation marks, mathematical symbols etc. In addition, it is possible to display symbols that the user makes up on its own. Automatic messages on display (shift left and right), appearance of the pointer, backlight etc. are considered as useful characteristics.

2.8.1 LCD CONNECTIONS

Depending on how many lines are used for connection to the microcontroller, there are 8-bit and 4-bit LCD modes. The appropriate mode is determined at the beginning of the process in a phase called “initialization”. In the first case, the data are transferred through outputs D0-D7 as it has been already explained. In case of 4-bit LED mode, for the sake of saving valuable I/O pins of the microcontroller, there are only 4 higher bits (D4-D7) used for communication, while other may be left unconnected.

Consequently, each data is sent to LCD in two steps: four higher bits are sent first (that normally would be sent through lines D4-D7), four lower bits are sent afterwards. With the help of initialization, LCD will correctly connect and interpret each data received. Besides, with regards to the fact that data is rarely read from LCD (data mainly are transferred from microcontroller to LCD) one more I/O pin may be saved by simply connecting the R/W pin to the Ground. Such saving has its price. EvenEven though message displaying will be normally performed, it will not be possible to read from a busy flag since it is not possible to read from display.

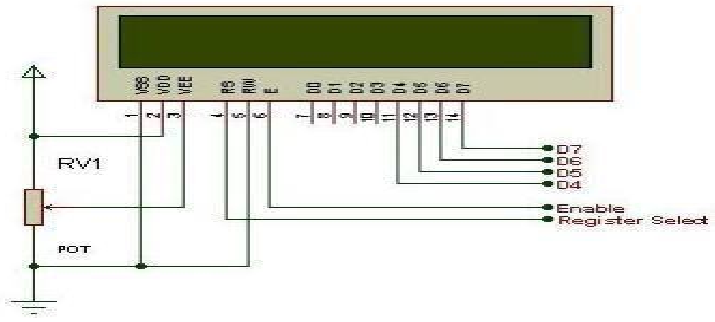


FIGURE 2.8.1 LCD CONNECTIONS

2.8.2 LCD PIN CONFIGURATION

Pin No:	**Pin Name:**	**Description**
1	Vss (Ground)	Ground pin connected to system ground

2	Vdd (+5 Volt)	Powers the LCD with +5V (4.7V – 5.3V)
3	VE (Contrast V)	Decides the contrast level of the display. Grounded to get maximum contrast.
4	Register Select	Connected to Microcontroller to shift between command/data register
5	Read/Write	Used to read or write data. Normally grounded to write data to LCD
6	Enable	Connected to Microcontroller Pin and toggled between 1 and 0 for data acknowledgement

7	Data Pin 0	Data pins 0 to 7 form a 8-bit data line. They can be connected to Microcontroller to send 8-bit data. These LCD’s can also operate on 4-bit mode in such case Data pin 4,5,6 and 7 will be left free.
8	Data Pin 1	
9	Data Pin 2	
10	Data Pin 3	
11	Data Pin 4	
12	Data Pin 5	

13	Data Pin 6	
14	Data Pin 7	
15	LED Positive	Backlight LED pin positive terminal
16	LED Negative	Backlight LED pin negative terminal

TABLE NO. 2.8.2 PIN CONFIGURATION OF LCD

2.8.3 LCD INITIALISATION

Once the power supply is turned on, the LCD is automatically cleared. This process lasts for approximately 15mS. After that, the display is ready to operate. The mode of operating is set by default. This means that:

- Display is cleared
- Mode

- DL = 1 Communication through 8-bit interface N = 0 Messages are displayed in one line
- F = 0 Character font 5 x 8 dots

- Display/Cursor on/off D = 0 Display off
- U = 0 Cursor off

- B = 0 Cursor blink off

- Character entry

ID = 1 Addresses on display are automatically incremented by 1 S = 0 Display shift off

Automatic reset is mainly performed without any problems. If for any reason power supply voltage does not reach full value in the course of 10mS, display will start perform completely unpredictably if voltage supply unit cannot meet this condition or if it is needed to provide completely safe operating, the process of initialization by which a new reset enabling display to operate normally must be applied.Algorithm according to the initialization is being performed depends on whether connection to the microcontroller is through 4- or 8-bit interface. All left over to be done after that is to give basic commands and of course- to display messages.

2.8.4 FEATURES OF 16×2 LCD

- Operating Voltage is 4.7V to 5.3V
- Current consumption is 1mA without backlight
- Alphanumeric LCD display module, meaning can display alphabets and numbers
- Consists of two rows and each row can print 16 characters.
- Each character is build by a 5×8 pixel box
- Can work on both 8-bit and 4-bit mode
- It can also display any custom generated characters

2.9 BUZZER

A buzzer or beeper is an audio signaling device, which may be mechanical, electromechanical, or piezoelectric (piezo for short). Typical uses of buzzers and beepers include **alarm devices, timers, and confirmation of user input such as a mouse click or keystroke**.

FIGURE 2.9 BUZZER

CHAPTER - 03

SOFTWARE SPECIFICATIONS

3.1 INTRODUCTION TO ARDUINO IDE

Arduino is a prototype platform (open-source) based on 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.

3.1.1 KEY FEATURES OF ARDUINO IDE

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, connecting 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 how to set up the Arduino IDE. Once we learn this, we will be ready to upload our program onthe Arduino board.

3.2 ARDUINO DATA TYPES

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

The following table provides all the data types that you will use During Arduino programming.

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.

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 which the 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.

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

Float

Data type for a 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.

3.3 STEPS TO UPLOAD THE PROGRAM IN ARDUINO BOARD

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

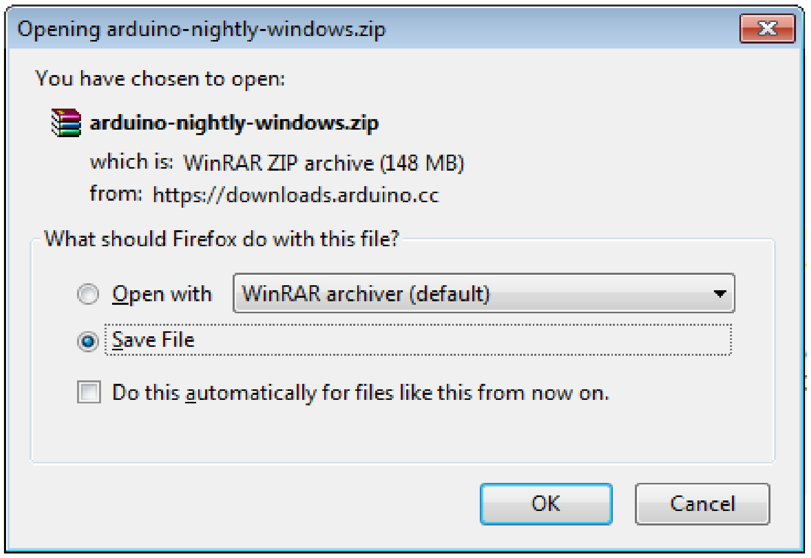
Step 1: First you must have your Arduino board (you can choose your favorite board) anda USB cable.

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 you would connect to a USB printer as shown in the following image.

FIGURE 3.3 USB CABLE

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.

FIGURE 3.3A DOWNLOADING OF ARDUINO SOFTWARE

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.

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

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

Here, we have selected the 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.

Step 8: Upload the program to your board.

CHAPTER - 4

IMPLEMENTATION

4.1 SCHEMATIC DIAGRAM

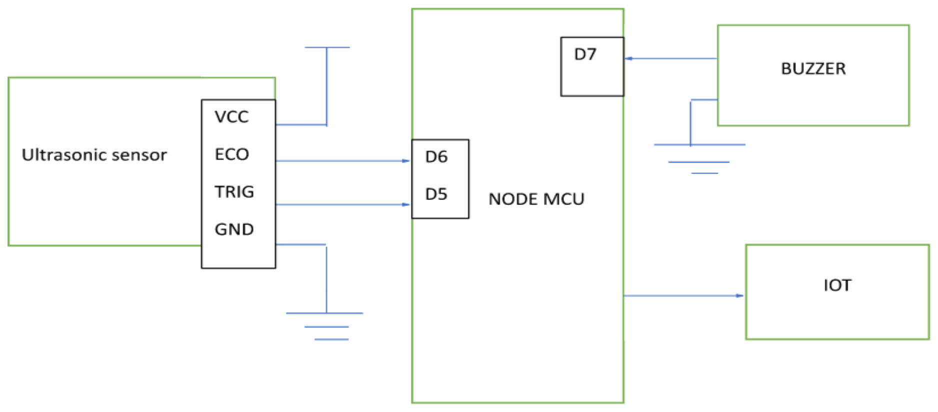


FIGURE 4.1 : SCHEMATIC DIAGRAM OF GARBAGE MONITORING AND AUTO ALERTING SYSTEM TO MUNICIPALITY

4.2 WORKING

Whenever power is supplied to the ultrasonic sensor it will detect the level of garbage by emitting ultrasonic sound waves that are converted into the electrical signals .In this project the code will run in the backend in arduino IDE software .

CASE I :

If the garbage level exceeds above 50cm it indicates LOW on LCD.

In this case there will be no message sended to the local municipalities.

CASE II :

If the garbage level is between 40 cm to 20 cm it indicates MEDIUM on LCD .

In this case there will be no message sended to the local municipalities.

CASE III :

If the garbage level is below 20 cm it indicates LOW on LCD .

In this case a message sended to the local municipalities through blynking IOT app that "The garbage which is in this XXXXX area is full" the buzzer which is attached to the bin will produce the sound which is in the frequency range of 2 to 6 KHz .So that the person who tries to throw the garbage in to that bin he will detect that tha bin is full.

4.3 CODING IMPLEMENTATION

```
#include <LiquidCrystal_I2C.h>
LiquidCrystal_I2C lcd(0x27,16,2); //D3-SCL , D4-SDA
#include<SoftwareSerial.h>
SoftwareSerial gsm(D5,D6);
const int trig=D3;
const int echo=D4;
long duration,distance;
String data=" ";
const int buzzer=D0;
bool a=true;
void setup(){
 Serial.begin(9600);
 gsm.begin(9600);
pinMode(trig,OUTPUT);
  pinMode(echo,INPUT);
 pinMode(buzzer,OUTPUT);
  digitalWrite(buzzer,LOW);
  lcd.init(); lcd.backlight();
  delay(100); lcd.print("*  READY  *");
  delay(1500);
  lcd.clear();
  lcd.print("    GARBAGE    ");
  lcd.setCursor(0,1);
  lcd.print(" MONITOR SYSTEM ");
  delay(2000);
  lcd.clear();}
 void loop()
{ultrasonic();
  lcd.clear();
 lcd.print("LEVEL : ");lcd.print(data);
  lcd.setCursor(0,1);
  lcd.print("distance : ");lcd.print(distance);
  delay(300);
 if(distance <= 20) {
    data="FULL";
    digitalWrite(buzzer,HIGH);
   lcd.clear();
    lcd.print("LEVEL : ");lcd.print(data);
    lcd.setCursor(0,1);
    lcd.print("  SENDING  SMS  ");
    delay(300);
    if(a==true)
    {
      sms();
      a=false;
    }
    lcd.clear(); lcd.print("LEVEL : ");lcd.print(data);
    lcd.setCursor(0,1);
    lcd.print("   SMS   SENT   ");
    delay(1000);
  } if(distance <= 50  && distance > 20)
  {
    a=true;
    data="MEDIUM";
  }
  if(distance >= 50)
  {
    a=true;data="NORMAL";
    digitalWrite(buzzer,LOW);}
 Serial.println(distance);}
void ultrasonic(){
digitalWrite(trig,LOW);
  delayMicroseconds(2);
  digitalWrite(trig,HIGH);
  delayMicroseconds(10);
  digitalWrite(trig,LOW);
  duration=pulseIn(echo,HIGH);
  distance=duration*0.034/2;}
void sms()
{
    String data="THE LEVEL OF GARBAGE IS HIGH : "+String(distance)+" CM \n";+"PLEASE CLEAN IT.";delay(500);
    delay(2000);
  Serial.println((char)26);gsm.println((char)26);
  delay(2000);
  Serial.println();gsm.println();
  delay(2000);{
Serial.println("AT+CMGS=\"+916281653902\"\r");gsm.println("AT+CMGS=\"+916281653902\"\r");    // enter your mobile number instead of xxxxxxxxxxx
  delay(2000);
  Serial.println(data);gsm.println(data);
  delay(2000);
  Serial.println((char)26);gsm.println((char)26);
  delay(2000);
  Serial.println();gsm.println();
  delay(2000);
}
```

CHAPTER - 5

OUTPUTS

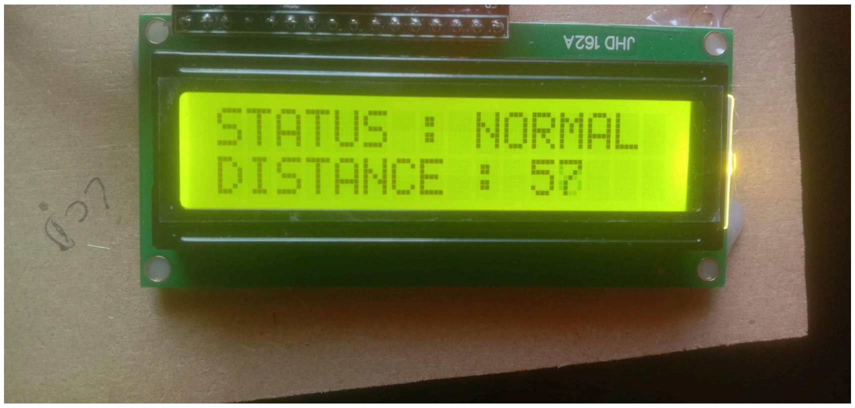


FIGURE 5.1 INDICATING NORMAL ON LCD

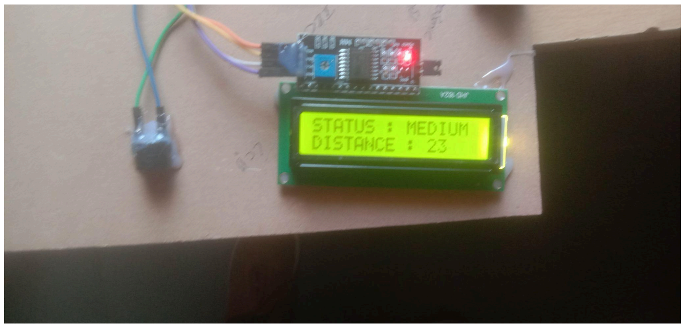


FIGURE 5.2 INDICATING MEDIUM ON LCD

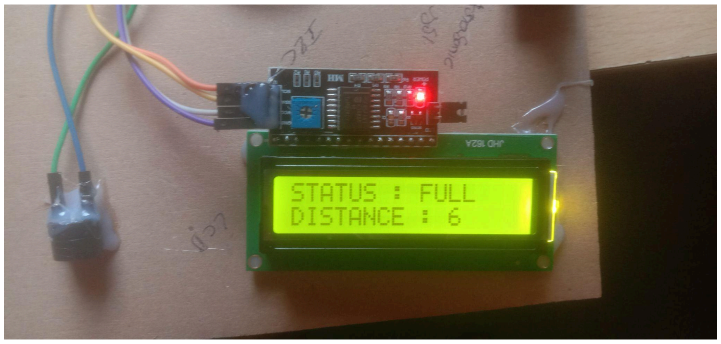


FIGURE 2.3 INDICTING FULL ON LCD

CHAPTER - 6
CONCLUSION & FUTURE SCOPE

6.1 CONCLUSION

This system was proposed because we are living in an age where tasks and systems are fusing together with the power of IOT to have a more efficient system of working and to execute jobs quickly! With all the power at our fingertips this is what we have come up with.The Internet of Things (IoT) shall be able to incorporate transparently and seamlessly a large number of different systems, while providing data for millions of people to use and capitalize. Building a general architecture for the IoT is hence a very complex task, mainly because of the extremely large variety of devices, link layer technologies, and services that may be involved in such a system. One of the main concerns with our environment has been solid waste management which impacts the health and environment of our society. The detection, monitoring and management of wastes is one of the primary problems of the present era. The traditional way of manually monitoring the wastes in waste bins is a cumbersome process and utilizes more human effort, time and cost which can easily be avoided with our present technologies. This is our solution, a method in which waste management is automated. This is our IoT Garbage Monitoring system, an innovative way that will help to keep the cities clean and healthy.

6.2 FUTURE SCOPE

In future we can implement this project as the “Garbage separator using sensors” which separates the garbage into wet and the dry categories.

CHAPTER - 7

REFERENCES

1) Mahar, A., Malik, R.N., Qadir, A., Ahmed, T., Khan, Z., and Khan, M.A., (2007), "Review and analysis of current solid waste management situation in urban areas of Pakistan", In # &'$&# (pp. 34-41).

2) Rajput "Scenario of Solid Waste Management," IEEE Journal on Emerging and Selected Topics in Circuits and Systems, vol. 3, no. 1, pp. 45–54, 2009.

3) Yadav I.C and Devi N.L (2009). Studies on Municipal Solid Waste Management in Mysore City- A case study. Report and Opinion, 1(3), 15-21.

4) Shivayogimath, "M2M-based metropolitan platform for IMS-enabled road traffic management in IoT," IEEE Communications Magazine, vol. 49, no.11, pp. 50-57, 2007.

5) Agarwal, A., Singhmar, A., Kulshrestha, M., Mittal, A.K., 2005. Municipal solid waste recycling and associated markets in Delhi, India. Journal of Resources, Conservation and Recycling 44(1), 73-90. Ashan, N., 1999

6) Sharholy, M., Ahmad, K., Mahmood, G., Trivedi, R.C., 2008. Municipal Solid waste management in Indian cities. A review, Journey of Waste Management 28,459-467.

7) Upadhyay, V. P., M. R. Prasad, A. Srivastav & K. Singh. 2005. Eco tools for urban waste management in India. Journal of Human Ecology 18: 253-269.

8) Zhu Ming Hau, Fan Xiu Min, Alberto Rovetta, He Qi Chang, Federico Vicentini, Liu Bing Kai, Alessandro Giusti, & Liu Yi, 2009, 'Municipal solid waste management in Pudong new area, China', Waste Management, vol. 29, pp. 1227- 1233.

9) Moqsud M. Azizul and Shigenori Hayashi(2006), "An Evaluation of Solid Waste Management Practice In Japan" Daffodil International University Journal of Science and Technology, Vol. 1(1), pp 39-44.