PROJECT REPORT

ON

IoT Based Drainage Monitoring System

**(In partial fulfillment of the requirements for the award of the degree of Bachelor of Technology in Electronics & Communication Engineering
Maulana Abul Kalam Azad University of Technology)**

Submitted by

Sl no.	Name	Roll No.
1	Neelesh Biswas	11900315027
2	Durba Sarkar	11900315016
3	Agrapriya Das	11900315004

Under the guidance of

Mr. Debajyoti Misra
Assistant Professor, ECE Dept., SIT

DEPARTMENT OF ELECTRONICS & COMMUNICATION ENGINEERING
SILIGURI INSTITUTE OF TECHNOLOGY
PO: SUKNA, SILIGURI, PIN: 734 009, WEST BENGAL
2018-2019

SILIGURI INSTITUTE OF TECHNOLOGY

PO: SUKNA, SILIGURI, PIN: 734 009, WEST BENGAL

2018-2019

DEPARTMENT OF ELECTRONICS & COMMUNICATION ENGINEERING

CERTIFICATE

*Certified that the project work entitled **'IoT Based Drainage Monitoring System'** is a bonafied work carried out by*

Neelesh Biswas	**ROLL NO. 11900315027**
Durba Sarkar	**ROLL NO. 11900315016**
Agrapriya Das	**ROLL NO. 11900315004**

*In partial fulfillment for the award for degree of **BACHELOR OF TECHNOLOGY** in **ELECTRONICS & COMMUNICATION ENGINEERING of** the **MAULANA ABUL KALAM AZAD UNIVERSITY OF TECHNOLOGY, KOLKATA** during the year 2018-2019. It is certified that all corrections/suggestions indicated for Internal Assessment has been incorporated in the report deposited in the Department. The project report has been approved as it satisfies the academic requirements in respect of Project Work prescribed for Bachelor of Engineering Degree*

.

Mr. Debajyoti Misra
(Asst. Professor, Dept of ECE, SIT)

Mr. Manas Kumar Parai
(In-Charge, Dept of ECE, SIT)

Acknowledgement

We have taken efforts in this project. However, it would not have been possible without the kind support and help of many individuals and organizations. We would like to extend our sincere thanks to all of them.

We are highly indebted to, Mr. Debajyoti Misra, Asst. Prof., ECE Dept for his guidance and constant supervision as well as for providing necessary information regarding the project & also for his support in completing the project.

We would also like to express our gratitude towards Mr. Manas Kr. Parai, In-Charge, ECE Dept. and Prof. Subhamay Sarkar, Project Co-ordinator, ECE Dept. for their kind co-operation and encouragement which help us in completion of this project.

We would like to express our special gratitude and thanks to industry persons for giving us such attention and time.

Our thanks and appreciations also go to my friends, seniors and people who have willingly helped us out with their abilities.

Finally, we would also like to thank our parents who constantly encouraged and supported us.

Neelesh Biswas (Roll No. 11900315027)

Durba Sarkar (Roll No. 11900315016)

Agrapriya Das (Roll No. 11900315004)

ECE Dept., SIT

Abstract

For making a smart city one needs to consider many parameters such as smart water, smart electricity, smart transportation etc. There will be a need of smart underground infrastructure which includes underground water pipelines, communication cables, gas pipelines, electric flow, etc. As most of the cities in India have adopted underground drainage system, it is very important that this system should work in a proper manner to keep the city clean, safe and healthy. If they fail to maintain the drainage system the pure water may get contaminated with drainage water and can spread infectious diseases. So different kind of work has been done to detect, maintain and manage these underground systems. This project represents the implementation and design functions for monitoring and managing underground drainage system with different approaches using Internet of Things.

Table of Contents

List of Figures

Chapter 1: Introduction

1.1 Background:

A smart city refers to a city, which focuses on capitalizing on new technologies and insights to transform the traditional style of systems, operations, real estate, and services. ICT plays an important role in bringing together systems and services, improving the quality of life of the citizens, whilst reducing the environmental footprint, in support of a low carbon economy.

The concept of smart city covers almost every aspect of society and people's livelihood, e.g., monitoring of public space, and management of underground pipelines and street illumination in respect of municipal facilities; construction, security, energy management and internal communication in respect of buildings; public transport service such as signal management, road traffic and parking monitoring; home automation and remote management; high-speed network and cloud storage; and electronic public and business services.

Sewage or household wastewater is a type of waste water which is produced by a group of people. It is specified by volume, physical condition, and the chemical and toxic constituents present. It consists of mostly grey water from sinks, tubs, showers and black water which is used to flush toilets, combined with the human waste that it flushes soaps and detergents. Sewage usually moves from the plumbing area of a building into a sewer, which carries it into an onsite sewage facility of which there are many kinds depending on the sewer design (sanitary sewer or combined sewer). The reality is that most wastewater produced globally remains untreated causing widespread water pollution, especially in low-income countries. To overcome this, the idea of "Smart Environment" came out. Smart Environment involves the implementation of green urban planning, through the use of internet-based technologies, to fully understand and analyze the distribution of public spaces, green belts, etc. It also throws light upon energy conservation, emission reduction, greening of channels, and revitalization of water bodies.

Following the idea of Smart Environment, we came out with a drainage monitoring system, which is actually an IoT module that works on Internet-based Technology. This system can be highly beneficial to the people living in a local area as well as people living in cities, towns or villages. First, it captures images and sends those images to the user through E-mail. After Sending the E-mail, it alerts nearby people by generating sound.

First, we make our Module robust both physically and algorithmically such that it is guaranteed to finish the designated jobs efficiently. Secondly, we would like to make our module easy to build and economic to manufacture so that, it can alleviate the cost of human and material resources to the largest extent. Thirdly, we aim to design our module in the way that it is very simple to operate. A good controlling system and methodology can, not only shorten the amount of time required to finish the job, but also allow a bigger user audience to use our product in real-life scenarios without much extra-training.

Our IoT Module is designed with Flow Sensor, Piezoelectric Buzzer, Raspberry Pi, which is a board works both as a microcontroller and a computer and Pi Camera (which is an add-on component as well as compatible to the Raspberry Pi) to detect images whenever sewage water is clogged in the system. Totally the system aims to aware people whenever the water in the drainage system gets full, so that further necessary steps people can take. It is fast, cost efficient and easy to use.

1.2 Nature and Purpose of Study:

The call for Drainage Monitoring System has been in demand for a very long time; there has been a lot of development in the field of Drainage Monitoring System. Yet there is a report on Water Pollution happening in millions of areas and localities. In this project the emphasis is on developing a Drainage Monitoring System that will be low cost in terms of mass production and also compatible with the existing technologies. To succeed in this attempt, the project is designed using Raspberry Pi which has an open source IoT platform.. For communication purposes wi-fi is used, which is designed in such a way that any computer or smartphone which has a wi-fi hotspot can receive the data transmitted from the robot using a third-party application. A piezoelectric buzzer is an electronic component used is also efficient in terms of value and performance; the features of the hardware and software will be discussed in detail in the later chapters.

1.3 Significance of this project:

The central idea of this project is to build a Drainage Monitoring System which is economically, technically and operationally feasible. The system requires minimal maintenance and as the system is autonomous it does not require any operator. It can also send data to any computer and smartphone because wi-fi is used for communication.

1.4 Objectives:

This project is focused by three main areas: Embedded Systems, Internet on Things & Image Processing. In this project our main focus is to maintain the drainage system so that any kind of clog or water blockage related problems can be prevented. Doing so, there can be some relaxation of manpower. Within an Instance, we can realize the clog in the system. We can also monitor the water level by seeing the images captured by the Pi Camera.

1.6 Method:

After we finalized the project title, we did the necessary study related to this project work for Electronic Circuit/System analysis and Electronic Circuit/System design and Module implementation and system integration. We made sketch of the IoT Drainage Monitoring System. Then we made the requirements list. After procuring all required materials we completed the architectural work for the system. After completing the body and connecting all required component we check and run the coding to give functionality to system.

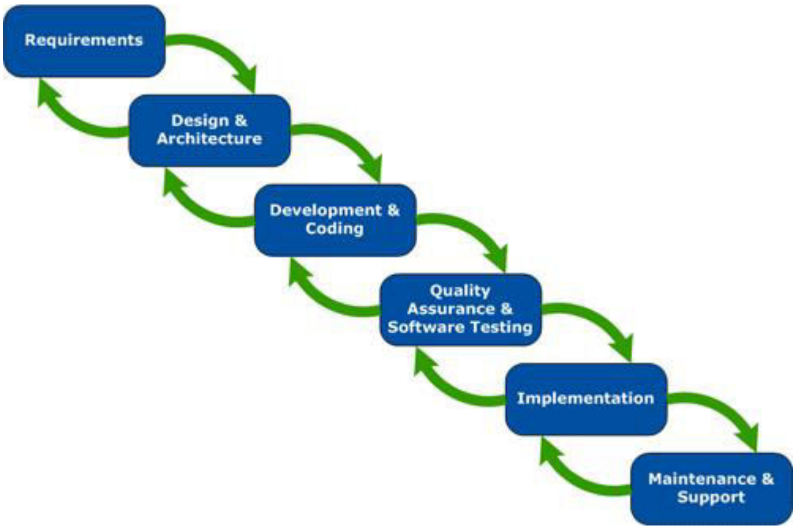


Fig 1.1: Step-by-step methods we followed in this System.

1.7 Overview of the technical area:

The system is designed with water flow sensor, an IoT Module embedded on development board, A buzzer, An LED, camera to detect clog in the sewage. The system is connected with a computer/smartphone viaWIFI to provide input for controlling the movement and to get visual feedback from search and rescue robot.

1.8 Overview of the report:

The outline of the thesis is as follows:

Chapter 1 discusses the goal and objectives of the project, the methods of how the project had been carried out, and overview of the project.

Chapter 2 discusses review of literature

Chapter 3 discusses system overview contains requirements and system/circuit specification.

Chapter 4 discusses Theoretical Foundations: The Engineering Mode, Details of theoretical model/understandings/components.

Chapter 5 discusses Operation containing Implementation of Hardware development platform, Code/software/theoretical model and design of the robot.

Chapter 6 discusses Result and discussion containing output, verification, validation and evaluation of the project.

Chapter 7 discusses Conclusion.

Chapter 2: Review of Literature:

2.1 Motivation:

- A global approximate by UNDP and UN-Habitat is that 90% of all wastewater generated is released into the environment untreated. In many developing countries the volume of domestic and industrial wastewater is frequently discharged lacking any treatment or seldom after primary treatment.
- Hawassa town is the capital of the Southern Nations, Nationalities and People Region (SNNPR) Hawassa City Administration and Sidama zone. It is located at a distance of 273 km South of Addis Ababa. The geographic coordinates of the town are approximately 7º 03' latitude North and 38º 29' longitudes East. Sewage water from the student's hostel buildings in the main campus is first diverted to septic tanks to remove the solids and then diverted to stabilization pond and a series of oxidation and polishing ponds. There is a possibility for utilizing the treated sewage water for irrigation purpose in the nearby hill side lands. Based on physical observation, soils on adjacent hillside land.

2.2 What is Internet of Things?

The future Internet will comprise not only millions of computing machines and software services but also billions of personal and professional devices, diminutive sensors and actuators, robots, and so on, and trillions of sentient, smart, and digitized objects. It is an overwhelmingly accepted fact that the fast-emerging and evolving Internet of Things (IoT) idea is definitely a strategic and highly impactful one to be decisively realized and passionately sustained with the smart adoption of the state-of-the-art information communication technology (ICT) infrastructures, a bevy of cutting-edge technologies, composite and cognitive processes, versatile and integrated platforms, scores of enabling tools, pioneering patterns, and futuristic architectures. Industry professionals and academicians are constantly looking out for appropriate use and business and technical cases in order to confidently and cogently proclaim the transformational power of the IoT concept to the larger audience of worldwide executives, end users, entrepreneurs, evangelists, and engineers.

The Internet of Things (IoT) is a technology which deals with connecting devices, machines and tools to the internet by means of wireless technologies. As of now there are over 9 billion 'Things' connected to the Internet. These 'Things' connected to the Internet are projected to cross 20 billion near future. Unification of technologies such as cloud computing, big-data, low-power embedded systems, machine learning are done by means of IoT. In the 2000s, we are heading into a new era of ubiquity, where the 'users' of the Internet will be counted in billions and where humans may become the minority as generators and receivers of traffic. Instead, most of the traffic will flow between devices and all kinds of 'things', thereby creating a much wider and more complex Internet of Things. The Internet of Things has many characteristics. Some of them are:

1) Efficient, scalable and associated architecture.
2) Unambiguous naming and addressing.
3) Abundance of sleeping nodes, mobile and non-IP devices.
4) Intermittent connectivity.

2.3 IoT Market Share:

In industries, IoT has its applications in various fields. But here are some of the areas where IoT has its Market Share:

1) Business/Manufacturing: Real-time analytics of supply chains and equipment, robotic machinery.
2) Healthcare: Portable health monitoring, electronic recordkeeping, pharmaceutical safeguards.
3) Retail: Inventory tracking, smartphone purchasing, anonymous analytics of consumer choices.
4) Security: Biometric and facial recognition locks, remote sensors.

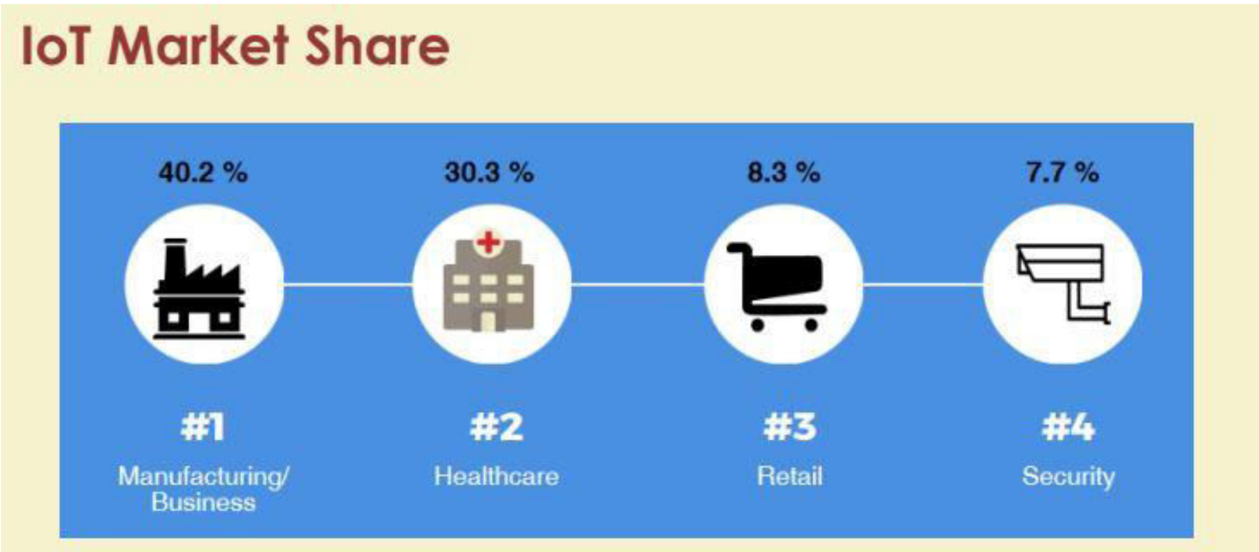


Fig: 2.1 Market Share of IoT in various areas.

2.4 Evolution of Connected Devices & Future of IoT:

1) ATMs: These ubiquitous money dispensers went online for the first time way back in 1974.
2) Web: World Wide Web made its debut in 1991 to revolutionize computing and communications.

3) Smart Meters: The first power meters to communicate remotely with the grid were installed in the early 2000s.
4) Digital Locks: Smartphones can be used to lock and unlock doors remotely, and business owners can change key codes rapidly to grant or restrict access to employees and guests.
5) Smart Healthcare: Devices connect to hospitals, doctors and relatives to alert them of medical emergencies and take preventive measures.
6) Smart Vehicles: Vehicles self-diagnose themselves and alert owners about system failures.
7) Smart Cities: City-wide infrastructure communicating amongst themselves for unified and synchronized operations and information dissemination.
8) Smart Dust: Computers smaller than a grain of sand can be sprayed or injected almost anywhere to measure chemicals in the soil or to diagnose problems in the human body.

The figure of online capable devices increased 31% from 2016 to 8.4 billion in 2017. Experts estimate that the IoT will consist of about 30 billion objects by 2020. It is also estimated that the global market value of IoT will reach $7.1 trillion by 2020.

Chapter 3: System Overview

3.1 Requirements:

A) <u>Hardware requirements</u>:

- Raspberry Pi 3 B+
- 16 GB Micro SD Card
- Water Flow Sensor/Hall Effect Sensor.
- Breadboard.
- Pi Camera.
- 2 LEDs.
- Power Supply.
- Piezoelectric Buzzer.
- Jumper Wires (for connection).
- Monitor.
- HDMI (female) to VGA (male) Adapter.
- Computer Peripherals (USB Keyboard & USB Mouse)

B) <u>Software requirements</u>:

i) For installing OS in Raspberry Pi:
   - 7zip.
   - Etcher.
   - SD Card Formatter.

ii) Softwares required in Raspberry Pi:
   - Python 3 IDLE.
   - ‘mailutils’ & ‘ssmtp’ for sending mails.
   - ‘picamera’ for capturing images.

3.2 Circuit & System Specification:

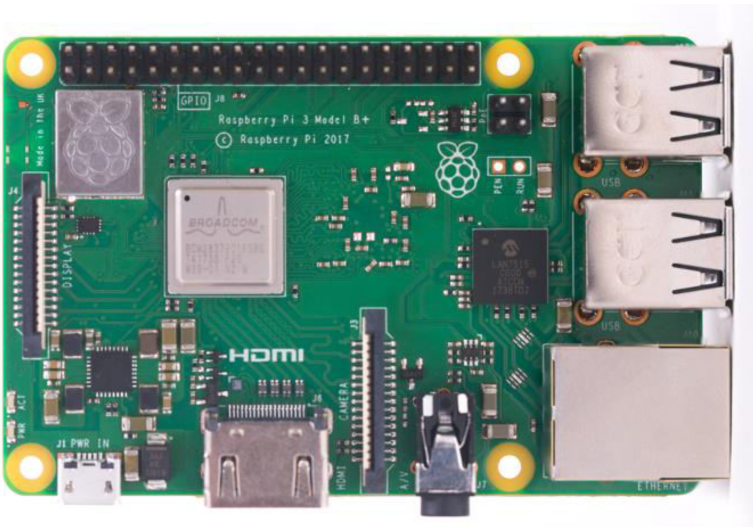
Raspberry Pi is a series of single-board computers developed in the UK. Raspberry Pi is highly preferred for programming (especially Python) and electronics projects. But for high processing power than other IoT modules, we use Raspberry Pi 3 B as the main component. It is a computer having an HDMI port, 4 USB Ports, an Ethernet Port, a 3.5mm audio input jack and a micro SD card slot as the SD card being the system storage of the Raspberry Pi 3. And on the other hand, it works like a microcontroller by having 40 GPIO pins for implementing our projects. Unlike Windows, we use Raspbian, a Debian-Linux based OS provided by Raspberry Pi Foundation of UK. Other OSes include Ubuntu Mate, Windows IoT Core, RISC OS, etc.

Software Specification:

- Windows 7 or higher (for installing & flashing Raspbian).

Hardware Specification:

- Developer: Raspberry Pi Foundation.
- Type: Single-board computer.
- Operating System: Raspbian OS.
- CPU: Quad cortex A53@1.2GHz.
- GPU: 400 MHz video core IV
- Ethernet: 10/100.
- Wireless: 802.11/Bluetooth 4.0.
- Video output: HDMI/Composite.



<u>Fig 3.1</u>: Raspberry Pi 3 B as seen in Top View.

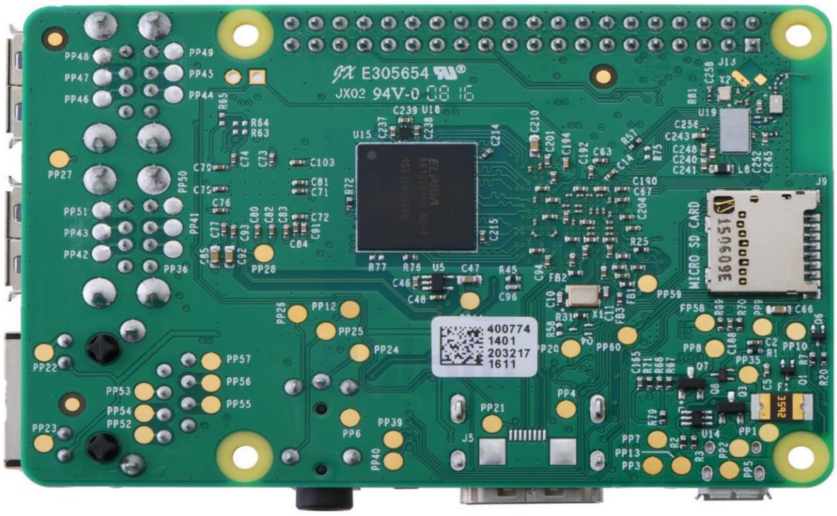


Fig 3.2: Raspberry Pi 3 B as seen in Bottom View.

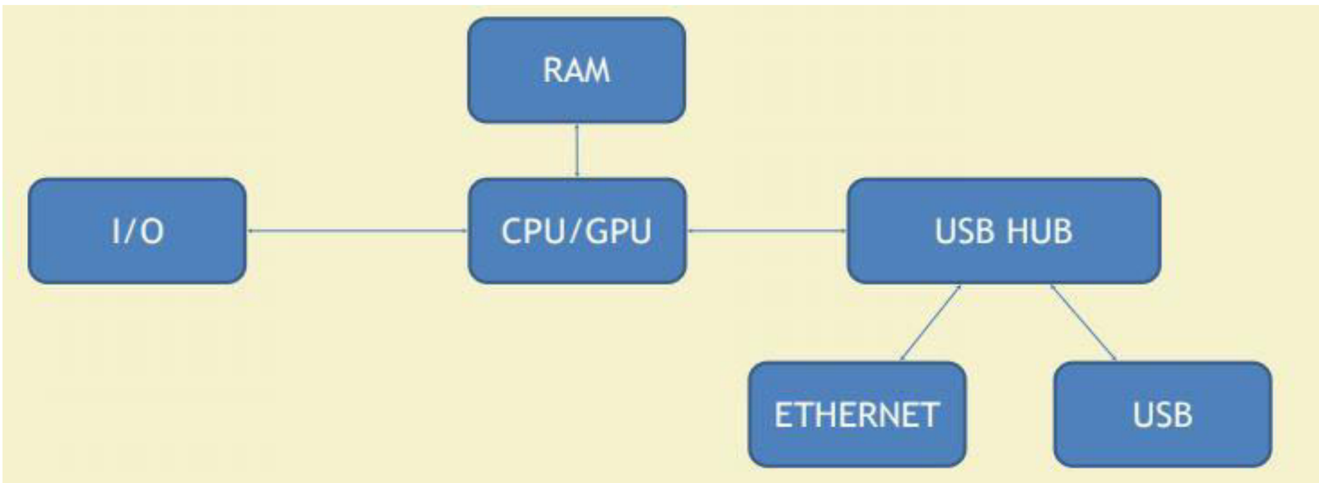


Fig 3.3: Basic Architecture of Raspberry Pi 3B+.

Chapter 4: Theoretical Foundation: The Engineering Mode

4.1 Introduction:

A theory is a contemplative and rational type of abstract or generalizing thinking, or the results of such thinking. Theories guide the enterprise of finding facts rather than of reaching goals, and are neutral concerning alternatives among values. A theory can be a body of knowledge, which may or may not be associated with particular explanatory models. To theorize is to develop this body of knowledge. A scientific theory is an explanation of an aspect of the natural world that can be repeatedly tested and verified in accordance with the scientific method, using accepted protocols of observation, measurement, and evaluation of results. Where possible, theories are tested under controlled conditions in an experiment. Engineering is the application of knowledge in the form of science, mathematics, and empirical evidence, to the innovation, design, construction, operation and maintenance of structures, machines, materials, software, devices, systems, processes, and organizations. For successfully complete an engineering project we must have strong theoretical knowledge in the required technologies and technical processes.

4.2 Working of IoT:

As we discussed in Chapter 2, The Internet of Things (IoT) is the network of physical objects or “things” embedded with electronics, software, sensors, and network connectivity, which enables these objects to collect and exchange data. The IoT allows objects to be sensed or controlled remotely across existing network infrastructure, creating opportunities for more direct integration of the physical world into computer-based systems, and resulting in improved efficiency, accuracy and economic benefit in addition to reduced human intervention. When IoT is augmented with sensors and actuators, the technology becomes an instance of the more general class of cyber-physical systems, which also encompasses technologies such as smart grids, virtual power plants, smart homes, intelligent transportation and smart cities.

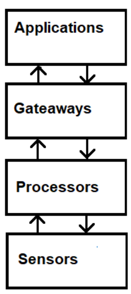


Fig 4.1: Block diagram of IoT.

The term “IoT” cannot be clarified without its working principle. IoT works with the help of four basic components: Application, Gateaways, Processors and Sensors:

1) Application: The applications for internet connected devices are extensive. Multiple categorizations have been suggested, most of which agree on a separation between consumer, enterprise (business), and infrastructure applications. The ability to network embedded devices with limited CPU, memory and power resources means that IoT finds applications in nearly every field. IoT devices are a part of the larger concept of home automation. Large smart home systems utilize a main hub or controller to provide users with a central control for all of their devices. These devices can include lighting, heating and air conditioning, media and security systems.

2) Getaways: An Internet of Things (IoT) Gateway provides the means to bridge the gap between devices in the field (factory floor, home, etc.), the Cloud, where data is collected, stored and manipulated by enterprise applications, and the user equipment (smart phones, tablets etc.). The IoT Gateway, provides a communication link between the field and the Cloud and can also offer local processing and storage capabilities to provide offline services and if required real time control over the devices in the field. The protocols that support the information exchange between interoperability domains can be classified to message-centric (AMQP, MQTT, JMS, REST) and data-centric (DDS, CoAP, XMPP). To use the full potential of IoT the interconnected devices communicate using lightweight protocols that don’t require extensive CPU resources. C, Java, Python and some scripting languages are the preferable choices used by IoT applications. To handle any needed protocol conversion, database storage or decision making (e.g. collision handling), IoT nodes use separate IoT gateways in order to supplement the low-intelligence within the IoT node.

3) Processor: A Processor performs basic operations in the data provided by the storage or memory. In IoT, the work of a processor is to process the code written in C or Java or in any programming language to a machine language which is uploaded to microcontroller boards like Arduino, ESP8266, etc.

4) Sensors: IoT loses its distinction without sensors. In the broadest definition, a sensor is a device, module, or subsystem whose purpose is to detect events or changes in its environment and send the information to other electronics, frequently a computer processor. Sensors are used in everyday objects such as touch-sensitive elevator buttons (tactile sensor) and lamps which dim or brighten by touching the base, besides innumerable applications of which most people are never aware. Moreover, analog sensors such as potentiometers and force-sensing resistors are still widely used. Applications include

manufacturing and machinery, airplanes and aerospace, cars, medicine, robotics and many other aspects of our day-to-day life.

a) OSI Layer:

The Open Systems Interconnection model (OSI model) is a conceptual model that characterizes and standardizes the communication functions of a telecommunication or computing system without regard to its underlying internal structure and technology. Its goal is the interoperability of diverse communication systems with standard protocols. The model partitions a communication system into abstraction layers. The original version of the model defined five layers:

1) Physical Layer: It defines the relationship between a device and a physical transmission medium (for example, an electrical cable, an optical fiber cable, or a radio frequency link). This includes the layout of pins, voltages, line impedance, cable specifications, signal timing and similar characteristics for connected devices and frequency (5 GHz or 2.4 GHz etc.) for wireless devices.

2) Data Link Layer: The data link layer provides node-to-node data transfer—a link between two directly connected nodes. It detects and possibly corrects errors that may occur in the physical layer. It defines the protocol to establish and terminate a connection between two physically connected devices. It also defines the protocol for flow control between them.

3) Network Layer: The network layer provides the functional and procedural means of transferring variable length data sequences (called datagrams) from one node to another connected in different networks.

4) Transport Layer: The transport layer provides the functional and procedural means of transferring variable-length data sequences from a source to a destination host, while maintaining the quality of service functions.

5) Application Layer: The application layer is the OSI layer closest to the end user, which means both the OSI application layer and the user interact directly with the software application. This layer interacts with software applications that implement a communicating component.

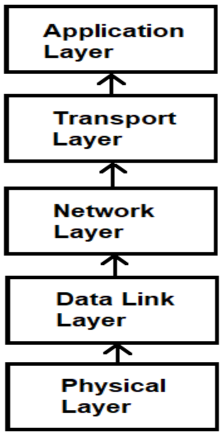


Fig 4.2: Layers of OSI

b) Protocols:

A protocol is a system of rules that allow two or more entities of a communications system to transmit information via any kind of variation of a physical quantity. The protocol defines the rules syntax, semantics and synchronization of communication and possible error recovery methods. Protocols may be implemented by hardware, software, or a combination of both. Protocols should therefore specify rules governing the transmission. In general, much of the following should be addressed:

1) Data formats for data exchange.
2) Address formats for data exchange.
3) Address mapping.
4) Routing.
5) Detection of transmission errors.
6) Acknowledgements.
7) Loss of information - timeouts and retries.
8) Direction of information flow.
9) Sequence control.
10) Flow control.

c) TCP/IP:

The Transmission Control Protocol (TCP) is one of the main protocols of the Internet protocol suite. It originated in the initial network implementation in which it complemented the Internet Protocol (IP).

Therefore, the entire suite is commonly referred to as TCP/IP. Major Internet applications such as the World Wide Web, email, remote administration, and file transfer rely on TCP.

The Transmission Control Protocol provides a communication service at an intermediate level between an application program and the Internet Protocol. It provides host-to-host connectivity at the Transport Layer of the Internet model. TCP is optimized for accurate delivery rather than timely delivery and can incur relatively long delays (on the order of seconds) while waiting for out-of-order messages or re-transmissions of lost messages. It is a reliable stream delivery service which guarantees that all bytes received will be identical with bytes sent and in the correct order. Since packet transfer by many networks is not reliable, a technique known as positive acknowledgement with re-transmission is used to guarantee reliability.

d) UDP:

The User Datagram Protocol (UDP) is one of the core members of the Internet protocol suite. With UDP, computer applications can send messages, in this case referred to as datagrams, to other hosts on an Internet Protocol (IP) network. Prior communications are not required in order to set up communication channels or data paths.

UDP is suitable for purposes where error checking and correction are either not necessary or are performed in the application; UDP avoids the overhead of such processing in the protocol stack. Time-sensitive applications often use UDP because dropping packets is preferable to waiting for packets delayed due to retransmission, which may not be an option in a real-time system.

e) Difference between TCP/IP and UDP:

Parameters	TCP/IP	UDP
Acronym	Transmission Control Protocol/Internet Protocol	User Datagram Protocol
Connection	TCP is a connection-oriented protocol.	UDP is a connectionless protocol.
Usage	TCP is suited for applications that require high reliability, and transmission time is relatively less critical.	UDP is suitable for applications that need fast, efficient transmission, such as games. UDP's stateless nature is also useful for servers that answer small queries from huge numbers of clients.
Weight	TCP is heavy-weight	UDP is light-weight.

Data Flow Control	TCP does Flow Control.	UDP does not have an option for flow control.
Error Checking	TCP does error checking and error recovery. Erroneous packets are retransmitted from the source to the destination.	UDP does error checking but simply discards erroneous packets. Error recovery is not attempted.

4.2 Details of the theoretical model & understandings:

Block Diagram:

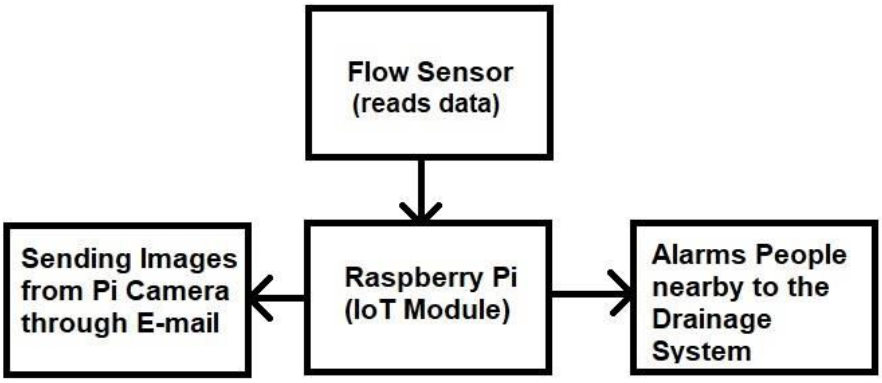


Fig 4.3: Block Diagram of the Project.

The System consists of a flow sensor which reads data whenever the system is blocked by water. When the system is blocked by water, clog gets happened. Then as the data is received by Raspberry Pi, here's the actual work begins. Firstly, The Pi Camera captures the image which is being saved in the Raspberry Pi as database, then after that it sends the image to the receiver's address which is given in the code. For low light mode, we use a Flash LED. After the image is being captured, the nearby will get notified by a sharp & alarming sound which is generated by a Piezoelectric Buzzer. Again, here we use a Red LED for our convenience, for better understanding to the nearby people. After both the processes, the nearby people as well as the user of the receiver's address may take further steps to how to overcome the problem.

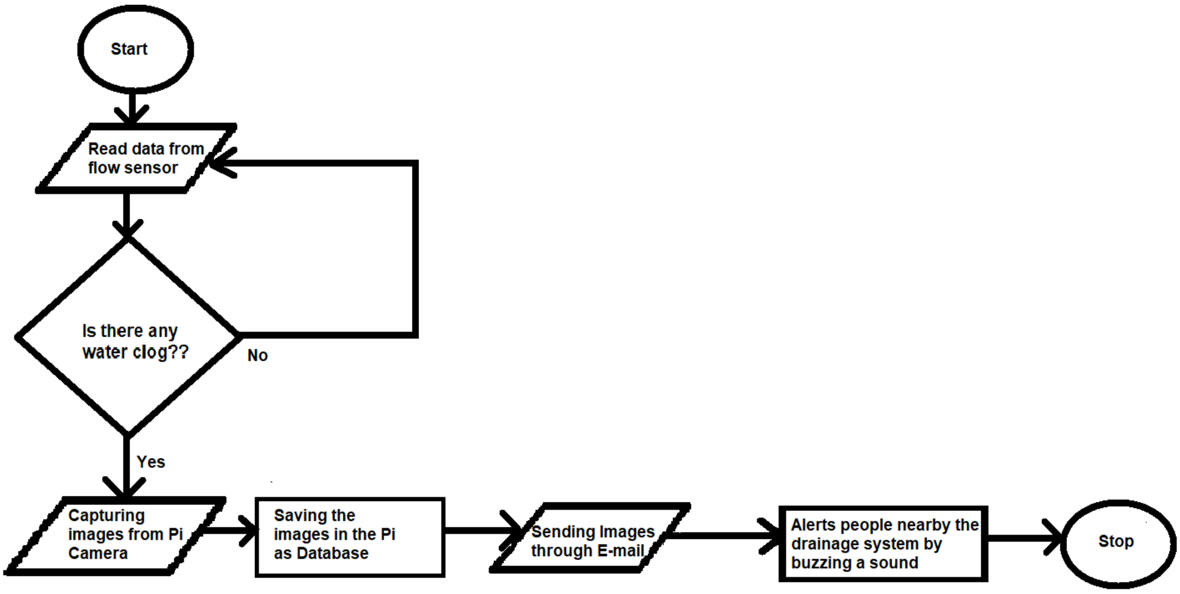


Fig 4.4: Flowchart of the project.

4.3 Introduction to Raspberry Pi 3:

The Raspberry Pi is a remarkable device: a fully functional computer in a tiny and low-cost package. Whether you're looking for a device you can use to browse the web or play games, are interested in learning how to write your own programs, or are looking to create your own circuits and physical devices, the Raspberry Pi – and it's amazing community will support you every step of the way. Raspberry Pi is a series of single-board computers developed by Raspberry Pi Foundation in the UK. Raspberry Pi is highly preferred for programming (especially Python) and electronics projects. But for high processing power than other IoT modules, we use Raspberry Pi 3 B as the main component. It is a computer having an HDMI port, 4 USB Ports, an Ethernet Port, a 3.5mm audio input jack and a micro SD card slot as the SD card being the system storage of the Raspberry Pi 3. And on the other hand, it works like a microcontroller by having 40 GPIO pins for implementing our projects. Unlike Windows, we use Raspbian, a Debian-Linux based OS provided by Raspberry Pi Foundation of UK. Other OSes include Ubuntu Mate, Windows IoT Core, RISC OS, etc.

Other Specifications of Raspberry Pi 3B+:

a) RAM: 1 GB SDRAM.
b) CPU: Quad cortex A53@1.2GHz.

c) GPU: 400 MHz video core IV.

d) Ethernet:10/100

e) Wireless: 802.11/Bluetooth 4.0

f) Video output: HDMI/Composite.

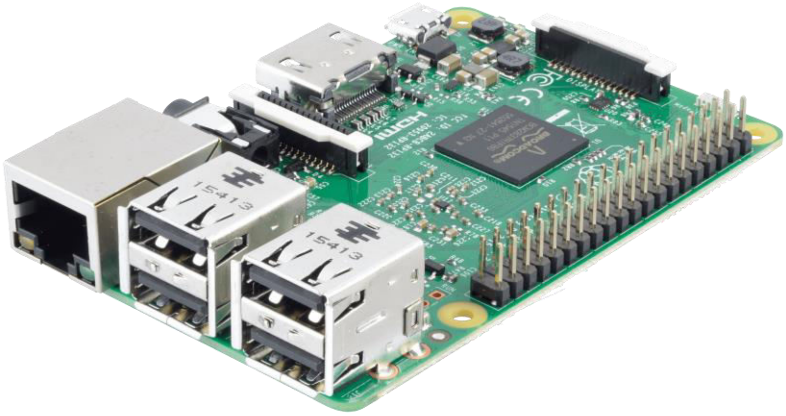


Fig 4.5: Raspberry Pi 3 B+ Model.

a) Installing Raspbian: A debian based-OS by Raspberry-Pi 3:

Being a microcomputer, we can install OSes on Raspberry Pi 3 like Raspbian, Windows IoT Core, RISC OS, Ubuntu Mate. But in this project, we installed Raspbian OS as it includes support for "hard float" code that takes advantage of the Raspberry Pi's hardware. That means software that use floating point operations should run much more quickly on Raspbian than they do on existing versions of Debian for the Raspberry Pi.

The Operating System can be obtained in two ways: 1) in the form of an Image file (.img) which is flashed in a MicroSD card and downloaded from the Internet (which will be the hard disk of Raspberry Pi 3 from where it will be booted.) and 2) in pre-loaded MicroSD Cards along with an SD Card Adapter which will be available in relevant market stores. Following are steps for installing Raspbian OS in Raspberry Pi 3 if you choose step 1. For this you require a MicroSD card (with SD Card Adapter), a laptop and few softwares that needed to be installed in your laptop:

1. Open any web browser in your laptop and type the URL link: "https://www.raspberrypi.org/downloads/raspbian/" to download the ZIP files of raspbian. The snapshot of the above link is given on Fig 4.6. In the snapshot, it has an option of downloading the file in two versions. You can choose any one of them. Just click either on "Download Torrent" or "Download ZIP" as per your desire.

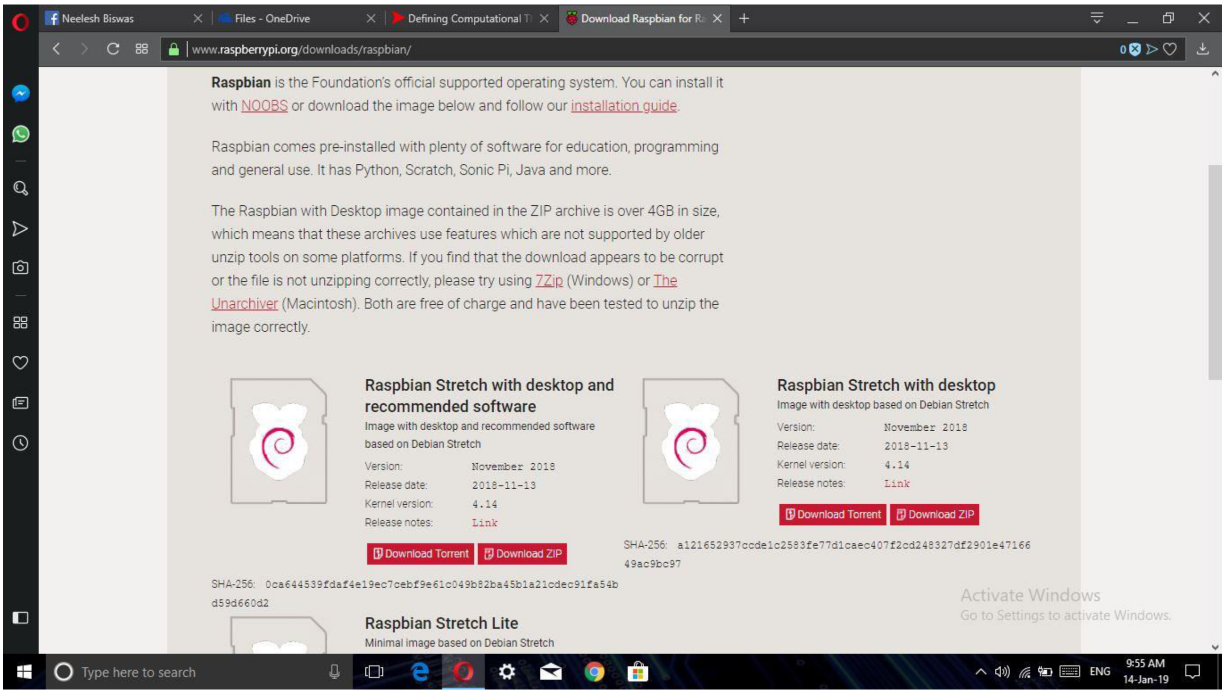


Fig 4.6: Snapshot of Downloading Raspbian.

2. After downloading Raspbian, download 7zip or any file unzipping software so that you can unzip the Raspbian image file by extracting the image file (.img).

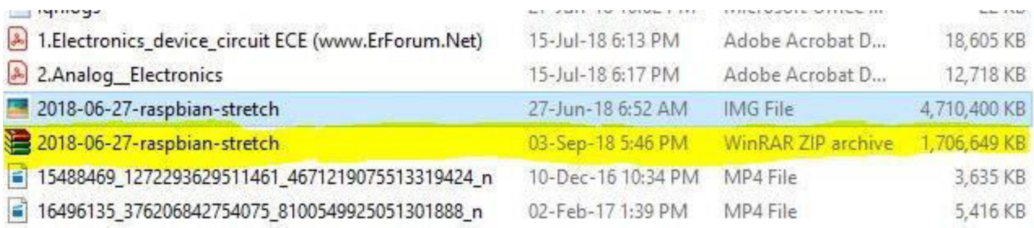


Fig 4.7: Snapshot of files a) Highlighted one is before extracting (or after downloading from Internet) Raspbian (WinRAR ZIP archive) and b) Selected one is after extracting Raspbian (IMG File).

3. Now insert your MicroSD Card on your laptop (make sure you've already inserted in a SD Card adapter so that you can insert it on the SD Slot of your laptop). The figure of a MicroSD Card with its adapter is given in Fig 4.8. Though the MicroSD card shown in the figure is of 2GB but we prefer the MicroSD Card should be above 8GB.

Fig 4.8: A Micro SD with SD Adapter.

4. Format the MicroSD card with SD Formatter (preferably) which is available on the Internet. The Drive is already selected, just select radio button "Quick format" and then click "Format" (See Fig. 4.9 for better understanding).

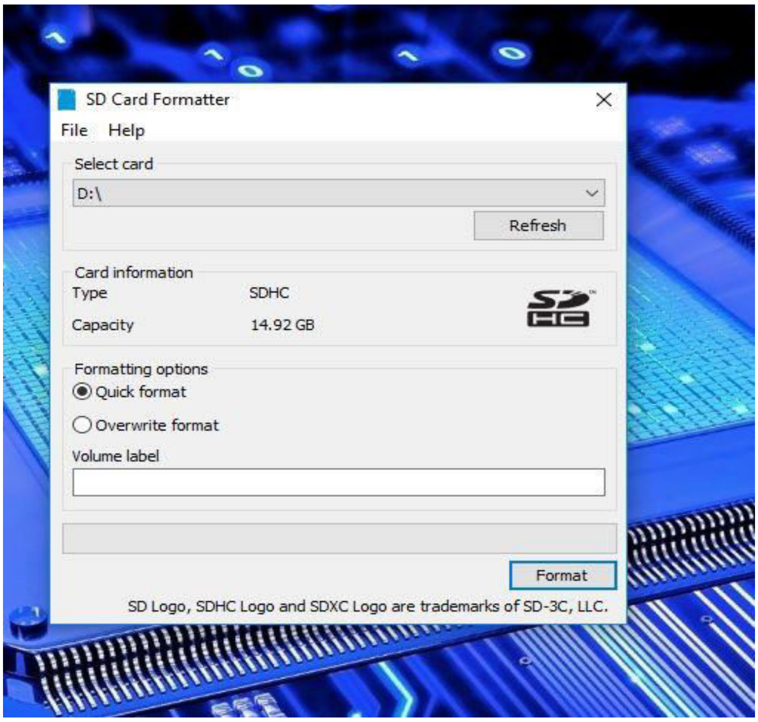
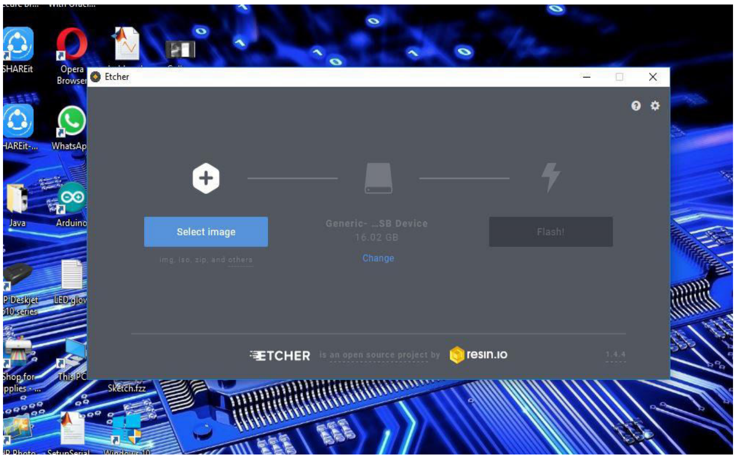


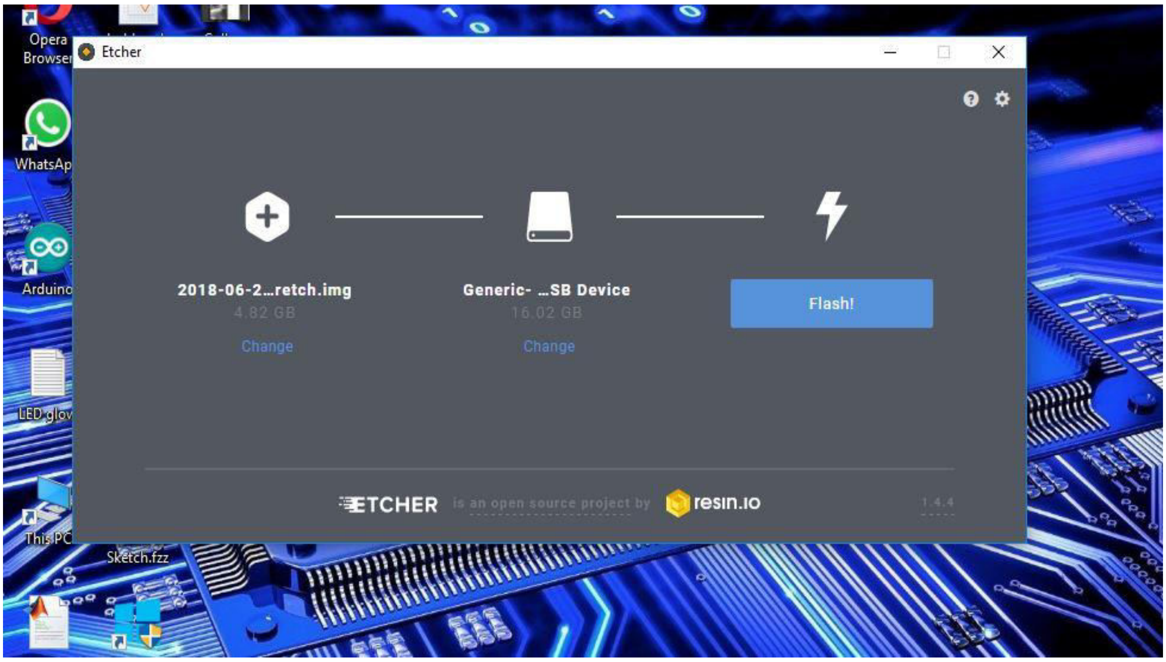
Fig 4.9: Snapshot of SD Card Formatter.

5. a) Download, install and open Etcher. After opening, a window will appear and it first it will ask to select the image file (you'll have to select the raspbian IMG file which is obtained in step 2).



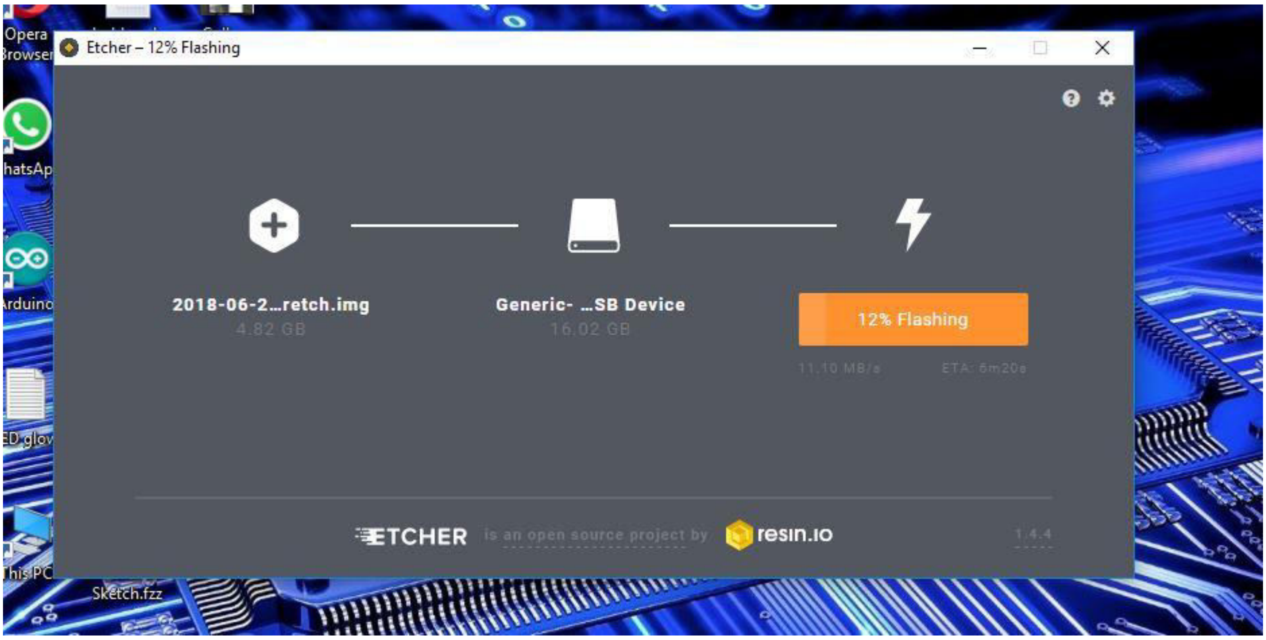
<u>Fig 4.10</u>: Snapshot of Etcher.

b) After that click on 'Select Image' after that the image file will be selected.

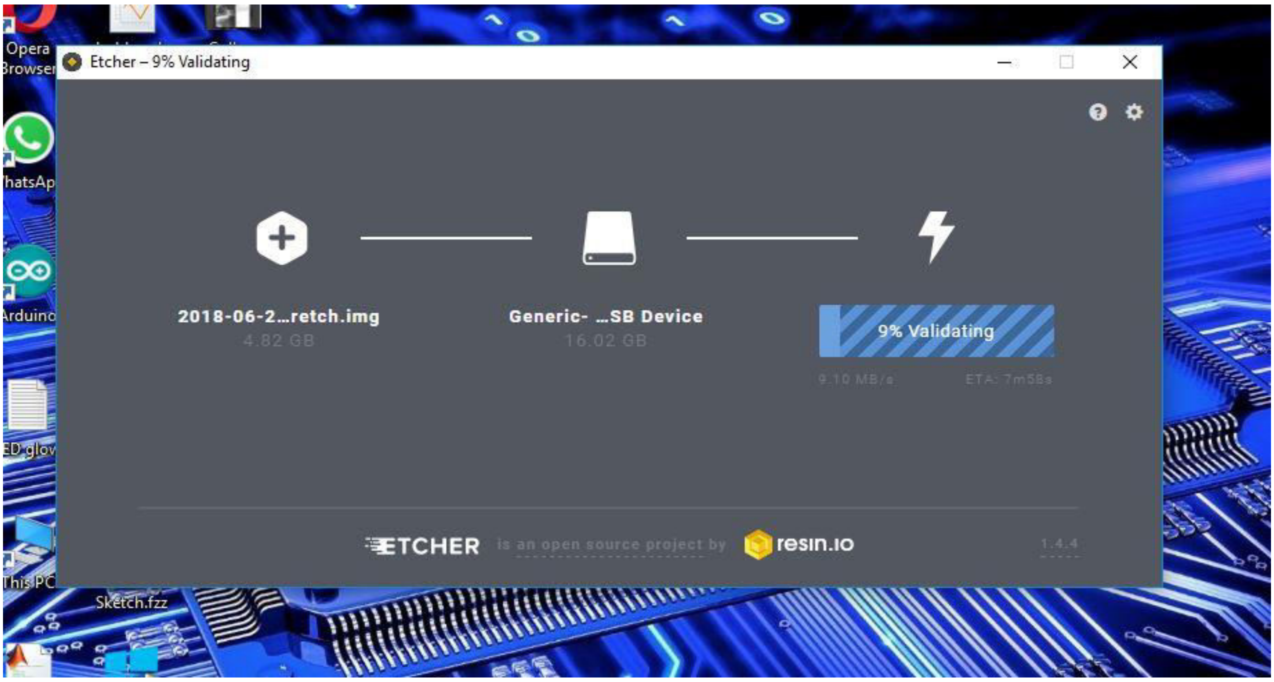


<u>Fig 4.11</u>: Snapshot of Etcher after selecting .img file.

c) After that Click on 'Flash' and the process will start flashing and after that it will validate the image file just to make the Raspbian bootable. Hence, the MicroSD Card is ready to use in the Raspberry Pi.

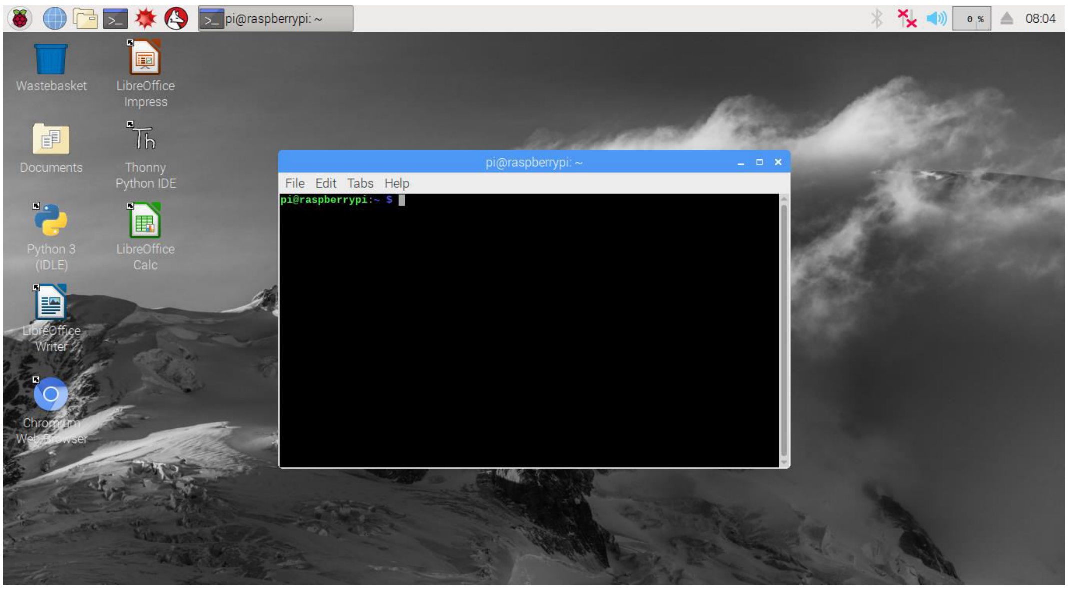


<u>Fig 4.12</u>: After click on the 'Flash' button.



<u>Fig 4.13</u>: After 'Flashing', the validation process will start which is the final step of installation.

6. Insert the MicroSD Card in your Raspberry Pi 3 and connect Power Supply and other peripherals like keyboard, mouse and monitor just to make Raspberry Pi 3 in use.



<u>Fig 4.14</u>: Raspbian OS Desktop booted on Raspberry Pi 3 with 'Raspberry Pi terminal' in the middle (similar to 'Command Prompt' in Windows).

b) <u>GPIO Pin Configuration of Raspberry Pi 3</u>:

A powerful feature of the Raspberry Pi is the row of GPIO (general-purpose input/output) pins along the top edge of the board. A 40-pin GPIO header is found on all current Raspberry Pi boards. One can make electronic circuits through these GPIO Pins. Making a Raspberry Pi part of the circuit is quite easy. At its most basic, it can provide power to a circuit, as well as a negative or ground end through the GPIO pins. Some pins are specifically always powered, mostly by 3.3 V, and always go to ground. Most of them can be programmed to create or recognise a HIGH or LOW signal, though; in the case of the Raspberry Pi, a HIGH signal is of 5 V (and/or 3.3 V) and a LOW signal is ground or 0 V. Wiring up a circuit to a Raspberry Pi is fairly simple. To create the physical circuit in the project, we're using a prototyping breadboard as it allows you to insert components and wires to connect them all together, without having to fix it permanently. One can modify and completely reuse your components because of this.

GPIO Zero was created to simplify the process of physical computing, helping new coders to learn. It's a Python library which builds upon the existing GPIO libraries RPi.GPIO, RPIO, and pigpio. However, while those libraries provide an interface to the GPIO pins themselves, GPIO Zero sits above them and provides a way to interface to the devices that you connect to those pins.

While writing programs, the GPIO pins can be written in any of the two modes, keeping in mind that you cannot operate both the modes at a time in a Python Script:

a) GPIO.BOARD: In this mode, you have to specify the I/O pin as its direct pin no. To set this mode we write, “`GPIO.setmode(GPIO.BOARD)`”. Now, for Example, if you want to set GPIO Pin 11 as Output Pin then you’ll have to write the following command in your Script “`GPIO.setup(11, GPIO.OUT)`”.

b) GPIO.BCM: Each GPIO pin in Raspberry Pi has its own specific name or GPIO No. To specify those pins by the numbers, we use BCM Mode or Broadcom Mode, named after Broadcom BCM2837B0 SoC (System on Chip), which is fabricated in Raspberry Pi. To set in this mode we write, “`GPIO.setmode(GPIO.BCM)`”. Now, for Example, if you want to set GPIO Pin 11 as Output Pin then first, keep in mind that Pin 11 has the alter name which is GPIO 17. Hence, you’ll have to write the following command in your Script “`GPIO.setup(17, GPIO.OUT)`”. In this project, since flow sensor cannot be operated in GPIO.BOARD mode, we are using BCM Mode.

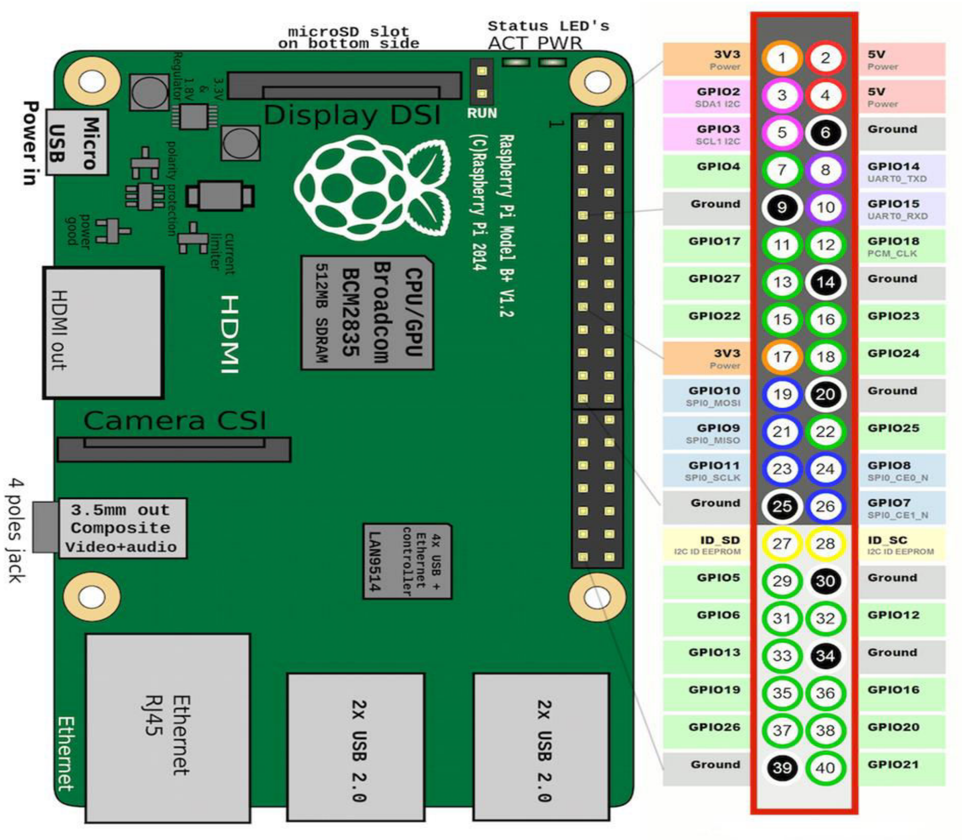


Fig 4.15: GPIO Pin Configuration of Raspberry Pi 3 B+ (right side) along with its top view (left side).

c) <u>Raspberry Pi Camera</u>:

In this project, we are also using a Raspberry Pi Camera module for capturing Images & recording videos. One of the most popular add-ons for the Raspberry Pi, the official Camera Module turns the single-board computer into a powerful digital camera. We can control the camera in two ways: (i) either the Raspberry Pi Terminal or (ii) by using Python commands in one of the programming applications pre-installed in the Raspberry Pi (Namely Scratch, Python, IDLE, Thonny Python, etc.).

Fig 4.16: Raspberry Pi Camera Module.

Note: Few things should be kept in mind while operating the Camera Module in the Raspberry Pi:

a) With the Pi switched off, locate its camera port. On most models, it's the one furthest away from the micro USB power connector, labelled 'CAMERA' or 'CSI'; on the Pi. Take hold of both ends of its plastic slider and pull it away from the Pi gently but firmly; it will move up a short distance, opening up the connector. Insert the ribbon cable, with its blue side facing the plastic slider. Push the slider back down, putting pressure on both sides, so that it gently clicks into place.

b) Plug the Raspberry Pi back in and turn it on. Once it has booted into the desktop, click on the Menu (top left) and go down to Preferences. Here you'll find the Raspberry Pi Configuration menu, which you should now click on. On the tab called Interfaces, you'll find an option to enable the camera; if this isn't set to Enabled, do so now. Raspbian doesn't have the camera enabled by default, so this is required. Now reboot your Pi.

c) The metal contacts on the ribbon cable should face away from the Ethernet socket (shown in Fig 4.15).

d) After rebooting the Pi, open the terminal and type "`raspistill -o firstpic.jpg`" to capture your first picture (just to make sure that the camera is working fine.). The image gets saved in the path " ".

4.4 Other Components used:

a) **<u>Water Flow Sensor/Hall Effect Sensor</u>:**

This is Water Flow Sensor / Fluid Flow meter Control Switch YF-S201. This sensor sits in line with your water line and contains a pinwheel sensor to measure how much liquid has moved through it. There's an integrated magnetic Hall Effect sensor that outputs an electrical pulse with every revolution. The Hall Effect sensor is sealed from the water pipe and allows the sensor to stay safe and dry. By counting the pulses from the output of the sensor, you can easily calculate water flow. Each pulse is approximately 2.25 millilitres.

<u>Fig 4.17</u>: Water Flow Sensor.

<u>Features</u>:

- Model: YF-S201.
- Working Voltage: 5 to 18V DC (min tested working voltage 4.5V).
- Max current draw: 15mA @ 5V.
- Output Type: 5V TTL.
- Working Flow Rate: 1 to 30 Liters/Minute.
- Working Temperature range: -25 to +80.
- Working Humidity Range: 35%-80% RH.
- Accuracy: ±10%.

- Maximum water pressure: 2.0 MPa.
- Output duty cycle: 50% +-10%.
- Output rise time: 0.04μs.
- Output fall time: 0.18μs.
- Flow rate pulse characteristics: Frequency (Hz) = 7.5 * Flow rate (L/min).
- Pulses per Liter: 450.
- Durability: minimum 300,000 cycles.
- Cable length: 15cm.
- 1/2" nominal pipe connections, 0.78" outer diameter, 1/2" of thread.
- Size: 2.5" x 1.4" x 1.4".

Note: This isn't a precision sensor, and the pulse rate does vary a bit depending on the flow rate, fluid pressure, and sensor orientation. It will need careful calibration if better than 10% precision is required. However, it's great for basic measurement tasks.

b) Piezo-electric Buzzer:

A piezo-electric buzzer is a sound producing device. The main working principle is based on the theory that, whenever an electric potential is applied across a piezoelectric material, a pressure variation is generated. Piezo buzzers generate a loud & sharp sound. So, they are typically used as alarm circuits.

Here we used the buzzer as the output of the circuit, such that whenever blockage appears or the water clogs in the drainage system, the buzzer generates sound so that the nearby people can get notified about the blockage in the drainage system. For convenience, we are using a Red LED along with the buzzer.

Fig 4.18: Piezoelectric Buzzer.

Features:

- Low frequency tone (2 kHz).
- Suitable for automatic radial taping machine (15mm-pitch).
- Operating temperature range: –10 to +70°C.
- Maximum input voltage: 30V0-P max.

c) LED:

A **light-emitting diode** (**LED**) is a semiconductor light source that emits light when current flows through it. Electrons in the semiconductor recombine with electron holes, releasing energy in the form of photons. This effect is called electroluminescence. The color of the light (corresponding to the energy of the photons) is determined by the energy required for electrons to cross the band gap of the semiconductor. White light is obtained by using multiple semiconductors or a layer of light-emitting phosphor on the semiconductor device. In the project, we used two LEDs, one White LED for capturing images from Pi Camera in Low-Light Mode and another one Red LED for convenience for nearby people.

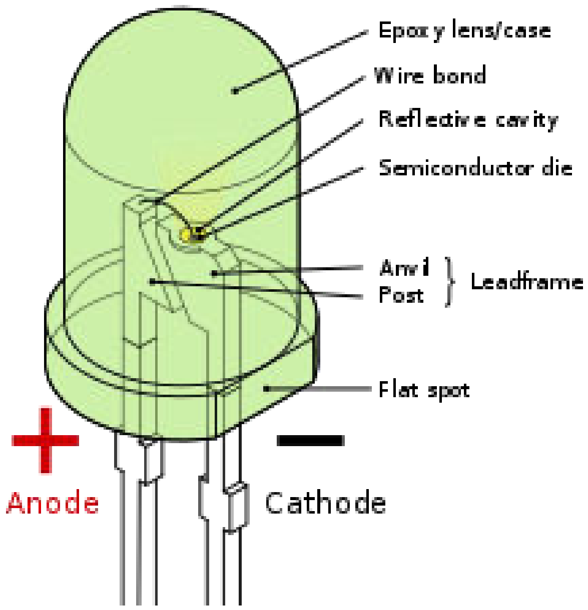


Fig 4.19: Internal Structure of LED.

Fig 4.20: LED as shown in Existence.

4.5 SMTP Protocol:

After assembling all the components, we will install software for sending e-mails whenever water gets blocked in the drainage system. For this, we use SMTP (Simple Mail Transfer Protocol). The Simple Mail Transfer Protocol (SMTP) is a communication protocol for electronic mail transmission. As an Internet standard, SMTP was first defined in 1982 by RFC 821, and updated in 2008 by RFC 5321 to Extended SMTP additions, which is the protocol variety in widespread use today. Mail servers and other message transfer agents use SMTP to send and receive mail messages. Proprietary systems such as Microsoft Exchange and IBM Notes and web mail systems such as Outlook.com, Gmail and Yahoo! Mail may use non-standard protocols internally, but all use SMTP when sending to or receiving email from outside their own systems. SMTP servers commonly use the TCP on port number 25.

SMTP is a connection-oriented, text-based protocol in which a mail sender communicates with a mail receiver by issuing command strings and supplying necessary data over a reliable ordered data stream channel, typically a TCP connection. An SMTP session consists of commands originated by an SMTP client (the initiating agent, sender, or transmitter) and corresponding responses from the SMTP server (the listening agent, or receiver) so that the session is opened, and session parameters are exchanged. A session may include zero or more SMTP transactions. An SMTP transaction consists of three command/reply sequences:

1. MAIL command, to establish the return address, also called return-path, reverse-path, bounce address, mfrom, or envelope sender. Blue arrows depict implementation of SMTP variations.

2. RCPT command, to establish a recipient of the message. This command can be issued multiple times, one for each recipient. These addresses are also part of the envelope.

3. DATA to signal the beginning of the message text; the content of the message, as opposed to its envelope. It consists of a message header and a message body separated by an empty line. DATA is actually a group of commands, and the server replies twice: once to the DATA command itself, to acknowledge that it is ready to receive the text, and the second time after the end-of-data sequence, to either accept or reject the entire message.

To install SMTP in a Raspberry Pi, we need to install two packages, namely ‘ssmtp’ & ‘mailutils’. Below are the further steps to install SMTP Protocol in a Raspberry Pi:

1) Open Raspberry Pi Terminal in after booting your Pi.
2) Type ‘`sudo apt-get install ssmtp`’ for installing SMTP protocol.
3) Type ‘`sudo apt-get install mailutils`’ for installing mailutils Package. (Mailutils is a collection of programs for managing, viewing and processing electronic mail. It contains both utilities and server domains and all operate in a protocol-agnostic way.)
4) Type ‘`sudo nano /etc/ssmtp/ssmtp.conf`’, to configure a file named ‘ssmtp.conf’ which will open in the terminal after executing the command (Shown in Fig 4.21).

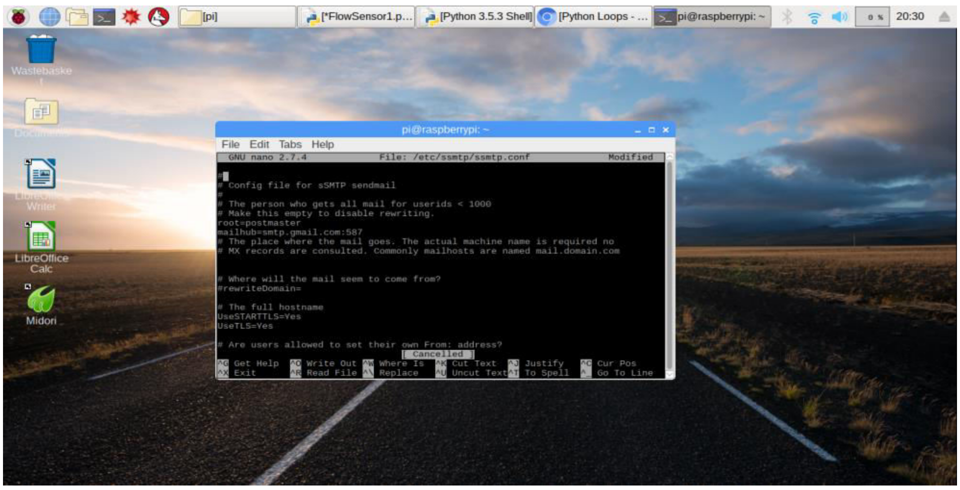


Fig 4.21: configuration of file ‘ssmtp.conf’ before entering the code given in step 5.

5) type the following in 'ssmtp.conf' and save the file:

```
root=postmaster
mailhub=smtp.gmail.com:587
hostname=raspberrypi
AuthUser=XXXX@gmail.com          #Sender's E-mail address
AuthPass=XXXX                    #Sender's E-mail password
FromLineOverride=YES
UseSTARTTLS=Yes
UseTLS=Yes
```

To save the file, press 'ctrl+x' then press 'y' & then press 'enter'. You will be redirected into the main terminal (which is shown in Fig. 4.14).

6) Type '`echo "Hello"|mail -s "Test" XXXX@gmail.com`' to send an e-mail through the terminal (Just to make sure that SMTP is working in Pi).

<u>Note</u>:

a) The address used in the last is the receiver address (Step 6). However the above steps will work only if the Sender's E-mail address is of Gmail Domain although, the Receiver's E-mail Address can be of any Domain.

b) Before implementing SMTP, domains like Gmail will try to obstruct sending E-mails from Pi Terminal, due to itself being third-party software. To avoid this, you should enable the access to third-party software for the sender's address, by first logging in into the sender's e-mail address & go to the security settings for further actions.

Chapter 5: Operation

5.1 Circuit Diagram:

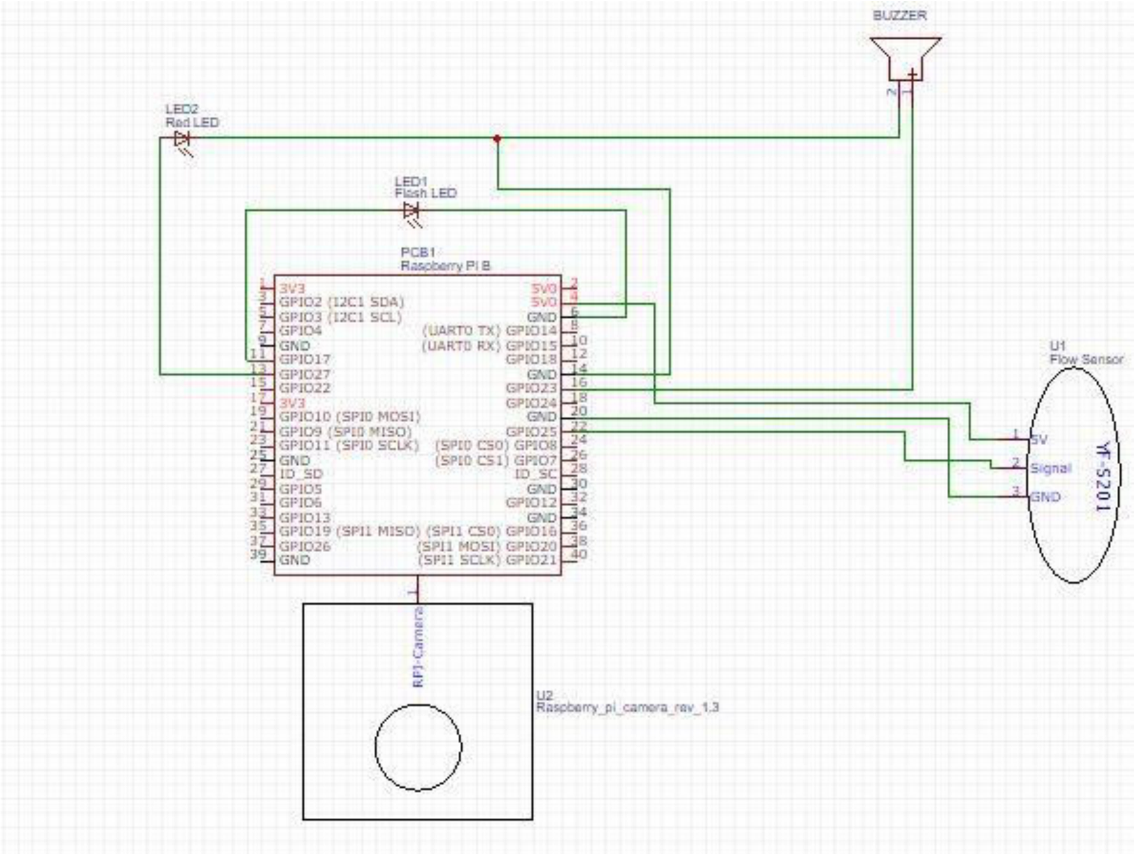


Fig 5.1: Schematic Circuit Diagram of the Whole System (Make Sure Pi Camera should be inserted into the CSI slot of Raspberry Pi).

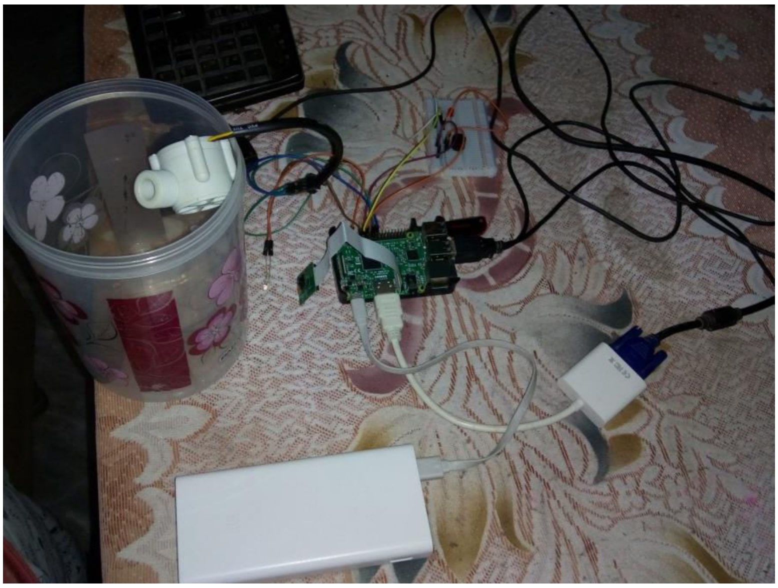


Fig 5.2: Prototype Circuit Diagram of the Whole System.

Table 5.1: Connection of Flow Sensor and Raspberry Pi:

Flow Sensor (Wires)	Raspberry Pi [GPIO Pin No. (BCM No.)]
Red Wire	4 (5V)
Black Wire	20 (GND)
Yellow Wire	22 (GPIO 25)

Table 5.2: Connection of Pi Camera & Flash LED to Raspberry Pi (unit for sending E-mails):

Component	Raspberry Pi [Port/GPIO Pin No.(BCM No.)]
Pi Camera	CSI Port
Flash LED (Positive Terminal)	11 (GPIO 17)
Flash LED (Negative Terminal)	6 (GND)

Table 5.3: Connection of Piezoelectric Buzzer & Red LED to Raspberry Pi (unit for nearby people):

<table>
<tr><th>Component</th><th>Raspberry Pi [GPIO Pin No.(BCM No.)]</th></tr>
<tr><td>Piezoelectric Buzzer (Positive Terminal)</td><td>13 (GPIO27)</td></tr>
<tr><td>Piezoelectric Buzzer (Negative Terminal)</td><td rowspan="2">14 (GND)
[Note: Both negative terminals should be connected in the same negative ground line on the breadboard]</td></tr>
<tr><td>Red LED (Negative Terminal)</td></tr>
<tr><td>Red LED (Positive Terminal)</td><td>16 (GPIO 23)</td></tr>
</table>

5.2 Programming Language used:

Raspberry Pi is mainly uses Python 3 as programming language. You don’t need to install Python just like in Windows or Linux, as Python 3 comes pre-installed in Raspbian OS. There are various platforms, you can write Python Scripts in Raspberry Pi such as Python 3 IDLE, Scratch, Thonny IDE, etc.

Why Python?:

- Python is a versatile language which is easy to script and easy to read.
- It doesn't support strict rules for syntax.
- Its installation comes with Integrated Development Environment (IDE) for programming.
- It supports interfacing with wide ranging hardware platforms.
- With open-source nature, it forms a strong backbone to build large applications.

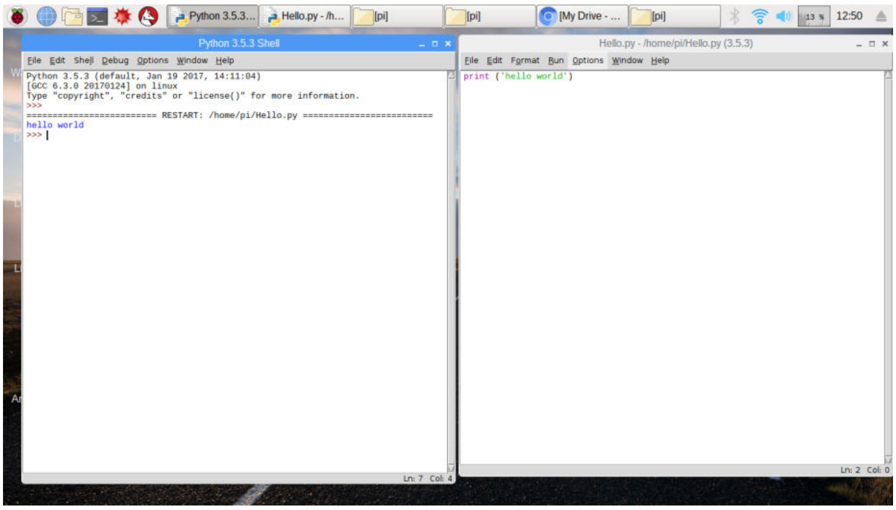


Fig 5.3: Python 3 IDLE Platform: 3.5.3 Shell (on left) & Editor (on right).

5.3 Programming Triggers:

1) To Read data from Flow Sensor:

To read data from flow Sensor, first we initialize a variable 'count' to zero and define a function 'countpulse1' which will keep track the no. of revolution, the pin wheel of flow sensor will rotate & store in 'count'. If there is no of revolution or in simpler words, if there is any water blockage, then the program terminates by printing 'count' as no. of revolutions the pinwheel rotates and goes to the next step.

```
#For counting the Number of revolution of Flow Sensor
pulse_pin = 25
GPIO.setmode(GPIO.BCM)
GPIO.setup(pulse_pin, GPIO.IN, pull_up_down = GPIO.PUD_UP)
count1=0
def countPulse1(channel):
        global count1
        print("Number of revolution of wheel of flow sensor:")
        count1+=1
        print(count1)
GPIO.add_event_detect(pulse_pin, GPIO.RISING, callback=countPulse1)
print("Inside while starting")
time.sleep(10)
```

2) **To click pictures and send those pictures via E-mail:**

a) First, we import modules for sending E-mail & attachments, and after that we specify the sender's E-mail address, sender's E-mail password & receiver's E-mail address as strings. Before doing, we made sure that the libraries 'ssmtp' and 'mailutlis' would have been installed in Raspberry Pi which has been discussed in chapter 4. After that we define a Directory & Prefix of the file name, as strings where the captured images would get saved in Pi.

```
# Importing modules for sending mail
from email.mime.multipart import MIMEMultipart
from email.mime.text import MIMEText
from email.mime.base import MIMEBase
from email import encoders

sender = 'XXXX@gmail.com'                    #Sender's E-mail address
password = 'XXXX'                            #Sender's E-mail password
receiver = 'XXXX@gmail.com'                  #Receiver's E-mail address

DIR = './Database/'
FILE_PREFIX = 'image'
```

.

b) After that we define a function named 'sendmail()' in which we first write the code to send the images & glow the flash LED while capturing images. Then, we use the above mentioned strings to send those images to receiver's e-mail address.

```
#defining a function to send E-mails
def send_mail():
    print ('Sending E-Mail')     # Create the directory if not exists
    if not os.path.exists(DIR):
        os.makedirs(DIR)
    # Find the largest ID of existing images.
    # Start new images after this ID value.

    files = sorted(glob.glob(os.path.join(DIR, FILE_PREFIX + '[0-9][0-9][0-9].jpg')))
    count = 0

    if len(files) > 0:
        # Grab the count from the last filename.
        count = int(files[-1][-7:-4])+1

    # Save image to file
    filename = os.path.join(DIR, FILE_PREFIX + '%03d.jpg' % count)
    # Capture the face
    with picamera.PiCamera() as camera:
        GPIO.setwarnings(False)
        GPIO.setmode(GPIO.BCM)
        GPIO.setup(17,GPIO.OUT)
        GPIO.output(17,True)
        time.sleep(10)
        GPIO.output(17,False)
        pic = camera.capture(filename)
```

```
# Sending mail
msg = MIMEMultipart()
msg['From'] = sender
msg['To'] = receiver
msg['Subject'] = 'Overflow in Drainage System'

body = "Sir/Ma'am, Overflow of water has blocked the drainage system of your area. Please kindly check this attachment"
msg.attach(MIMEText(body, 'plain'))
attachment = open(filename, 'rb')
part = MIMEBase('application', 'octet-stream')
part.set_payload((attachment).read())
encoders.encode_base64(part)
part.add_header('Content-Disposition', 'attachment; filename= %s' % filename)
msg.attach(part)
server = smtplib.SMTP('smtp.gmail.com', 587)
server.starttls()
server.login(sender, password)
text = msg.as_string()
server.sendmail(sender, receiver, text)
server.quit()
```

3) **<u>Code for Buzzer & Red LED</u>:**

Finally we create a loop & condition i.e., for exceeding the threshold value of the rate of theflow , the images will be sent to receiver and after that the buzzer will generate sound & the red LED will glow at the same time.

```
while True:
    if count1 < 100:  # When output from flow sensor doesn't exceed the threshold value
        print ("No Water clog")
        sleep(0.3)
    elif count1 >= 100:  # When output from flow sensor exceeds threshold value
        print ("Water clog detected in the drainage system")
        send_mail()
        for j in range(0,2):
            GPIO.setmode(GPIO.BCM)
            GPIO.setup(27,GPIO.OUT)
            GPIO.setup(23,GPIO.OUT)
            GPIO.output(27,True)
            GPIO.output(23,True)
            time.sleep(15)
            GPIO.output(27,False)
            GPIO.output(23,False)
            time.sleep(1)
            GPIO.cleanup()
```

5.4 Source Code (in Python 3):

```
import time, sys
import os
import glob
import picamera
import RPi.GPIO as GPIO
import smtplib
from time import sleep

# Importing modules for sending mail
from email.mime.multipart import MIMEMultipart
```

```
from email.mime.text import MIMEText
from email.mime.base import MIMEBase
from email import encoders

sender = 'neelesh.ece30@gmail.com'               #Sender's E-mail address
password = 'XXXX'                      #Sender's E-mail password
receiver = 'sonai.mgk30@gmail.com'            #Receiver's E-mail address

DIR = './Database/'
FILE_PREFIX = 'image'

#For counting the Number of revolution of Flow Sensor
pulse_pin = 25
GPIO.setmode(GPIO.BCM)
GPIO.setup(pulse_pin, GPIO.IN, pull_up_down = GPIO.PUD_UP)
count1=0
def countPulse1(channel):
        global count1
        print("Number of revolution of wheel of flow sensor:")
        count1+=1
        print(count1)
GPIO.add_event_detect(pulse_pin, GPIO.RISING, callback=countPulse1)
print("Inside while starting")
time.sleep(10)

#defining a function to send E-mails
def send_mail():
    print ('Sending E-Mail')    # Create the directory if not exists
    if not os.path.exists(DIR):
        os.makedirs(DIR)
    # Find the largest ID of existing images.
    # Start new images after this ID value.

    files = sorted(glob.glob(os.path.join(DIR, FILE_PREFIX + '[0-9][0-9][0-
9].jpg')))
    count = 0

    if len(files) > 0:
        # Grab the count from the last filename.
        count = int(files[-1][-7:-4])+1

    # Save image to file
    filename = os.path.join(DIR, FILE_PREFIX + '%03d.jpg' % count)
    # Capture the face
    with picamera.PiCamera() as camera:
        GPIO.setwarnings(False)
        GPIO.setmode(GPIO.BCM)
        GPIO.setup(17,GPIO.OUT)
```

```
        GPIO.output(17,True)
        time.sleep(10)
        GPIO.output(17,False)
        pic = camera.capture(filename)
    # Sending mail
    msg = MIMEMultipart()
    msg['From'] = sender
    msg['To'] = receiver
    msg['Subject'] = 'Overflow in Drainage System'

    body = "Sir/Ma'am, Overflow of water has blocked the drainage system of
your area. Please kindly check this attachment"
    msg.attach(MIMEText(body, 'plain'))
    attachment = open(filename, 'rb')
    part = MIMEBase('application', 'octet-stream')
    part.set_payload((attachment).read())
    encoders.encode_base64(part)
    part.add_header('Content-Disposition', 'attachment; filename= %s' %
filename)
    msg.attach(part)
    server = smtplib.SMTP('smtp.gmail.com', 587)
    server.starttls()
    server.login(sender, password)
    text = msg.as_string()
    server.sendmail(sender, receiver, text)
    server.quit()
while True:
    if count1 < 100:  # When output from flow sensor doesn't exceed the
threshold value
        print ("No Water clog")
        sleep(0.3)
    elif count1 >= 100:  # When output from flow sensor exceeds threshold
value
        print ("Water clog detected in the drainage system")
        send_mail()
        for j in range(0,2):
            GPIO.setmode(GPIO.BCM)
            GPIO.setup(27,GPIO.OUT)
            GPIO.setup(23,GPIO.OUT)
            GPIO.output(27,True)
            GPIO.output(23,True)
            time.sleep(15)
            GPIO.output(27,False)
            GPIO.output(23,False)
            time.sleep(1)
            GPIO.cleanup()
```

Chapter 6: Results and Discussion

6.1 Program Output:

1) After executing the program, the flow sensor tries to send data until some water in the drainage system flows & hence. It calculates the no. of revolutions of the flow sensor.

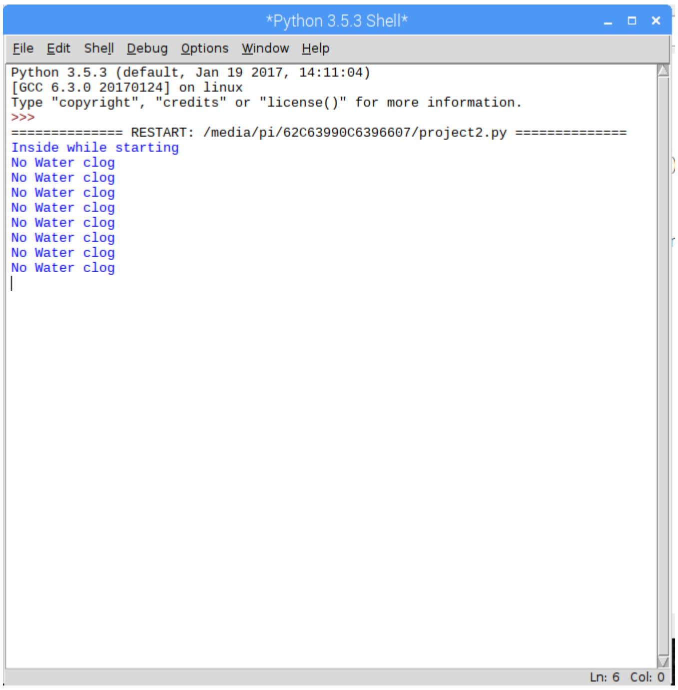


Fig 6.1: When the flow sensor tries to detect the data.

2) When the flow sensor receives data, the pinwheel rotates & sends the data through the principle of Hall effect to the Raspberry Pi.

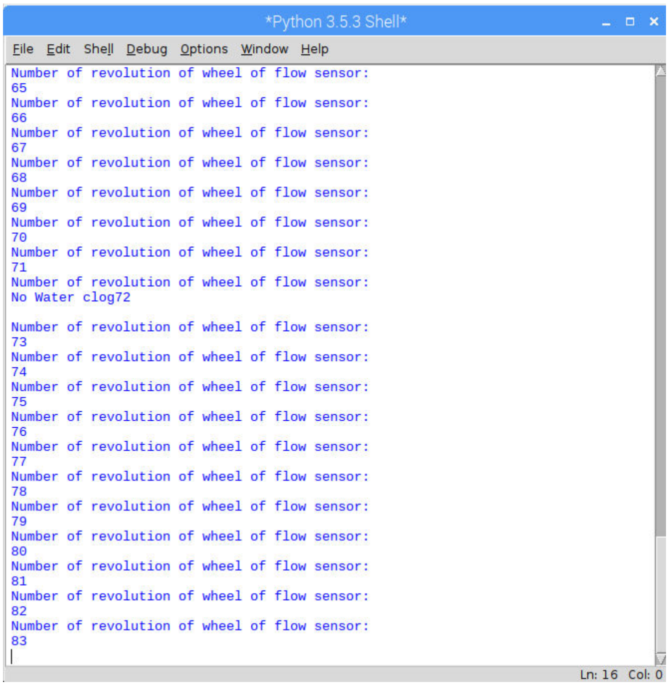
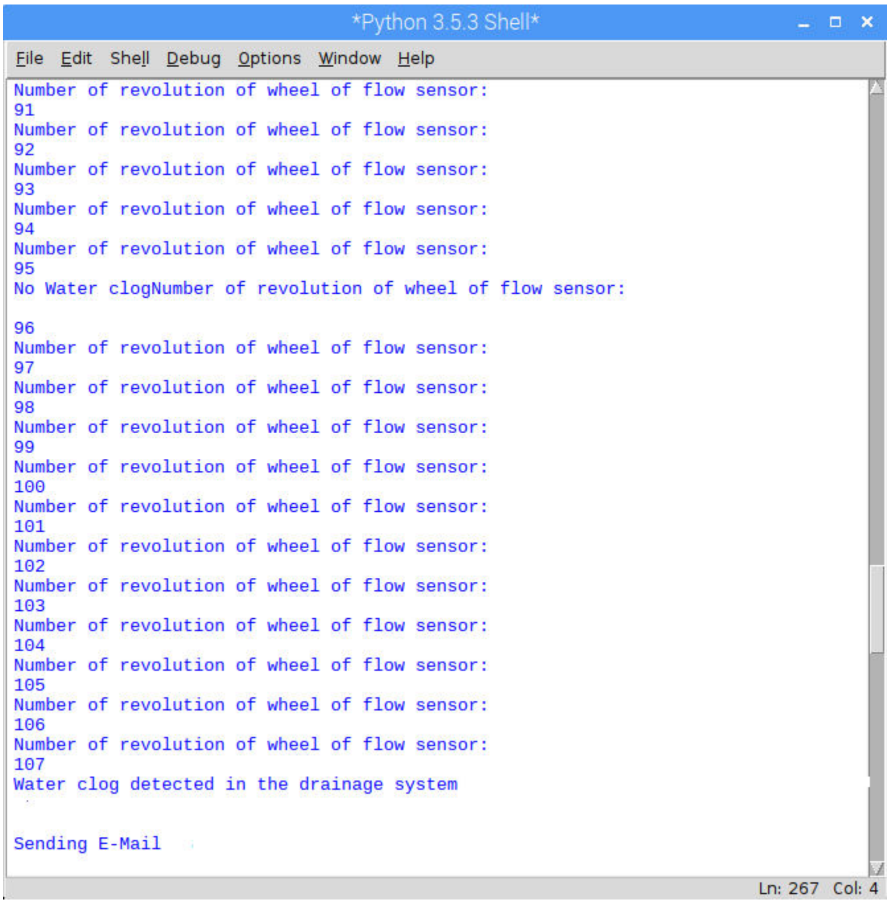


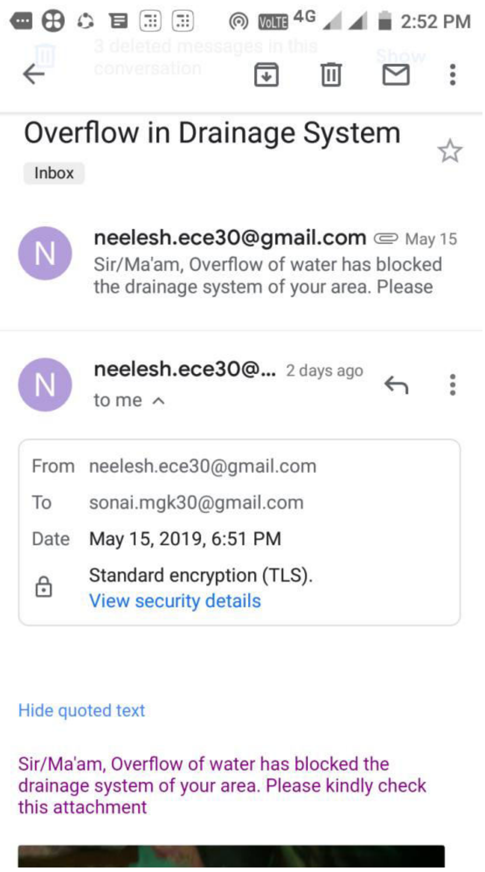
Fig 6.2: When the flow sensor receives data

3) When the water reaches the threshold value, the camera sends the captured image of the system via E-mail of the receiver's address provided in the code.



<u>Fig 6.3</u>: When water clog gets detected in the system.

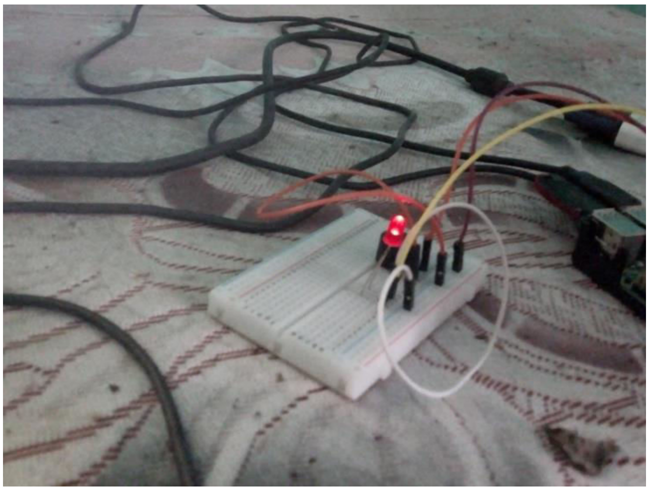
<u>Fig 6.4</u>: The Pi Camera when tries to capture the image.



<u>Fig 6.5</u>: Screenshot of the E-mail received by the user.

<u>Fig 6.6</u>: Image captured by Pi Camera.

4) As the E-mail is sent, the people living near to the drainage system, gets a sharp sound alarming that water blocked the drainage system, so that by hearing this, people will have to take further necessary actions.



<u>Fig 6.7</u>: Unit for the nearby people (Red LED glowing indicates that the buzzer is generating sound).

6.2 Advantages:

1. <u>Prevents water overflow</u>: It always gives the accurate data so we can easily handle this to being overflow.

2. <u>Decreases possibility of toxic materials</u>: There's a danger of toxic contamination whenever a flood occurs. If you have a sub drainage system installed, it will enable you to get rid of excess water and reduce the likelihood of contaminating your water supply.

3. <u>Decrease maintenance cost</u>: It gives the exact location where the blockage occurs. So we can easily rectify it. Due to this fact fuel and labour cost will reduced.

6.3 Disadvantages:

1 The flash LED will glow according in the day as well as night.

2 Since the system was supposed to encase in a transparent case, the picture quality of the camera may get poor.

6.4 Further Improvements:

Although we used only Flow Sensor in this project, some more sensors could be added, for example; Ultrasonic Sensor or water level sensor, which is used to measure the water level of the system, then after that an LDR Sensor which is convenient to glow flash LED so that the imaged could get captured only in night.

Chapter 7: Conclusion

In this project, we understood the working principle of modern gadgets like Raspberry Pi & Flow Sensor. The flow sensor is really helpful for the ones those who want to make further projects on drainage monitoring system or flow meters, etc., and what we feel that certain sensors could further be added for better enhancement of the project. And about Raspberry Pi, it's really one of the amazing inventions that we've ever seen in the era of modern technologies. From our personal point of view, we can say that if we use Raspberry Pi judiciously along with other embedded system modules like Arduino, Node MCU, etc., in the field of electronics & computer projects, then only the terms 'Smart Home', 'Smart City', 'Smart Town' will have a worth.

The experience of doing this project was extremely rewarding, the years spent in learning the theory was put to use. This felt fulfilling and self-reliant. From a personal outlook the project can be classified success for this reason. It could also be said successful in a professional point of view because of achieving most of the objectives required by the University. Some limitations and lack of validity in some areas have been discussed in previous chapters. This leaves a lot of room for improvement. The module itself is capable of getting the data, transmitting it and moving around unmanned. The module is easy to build, economical to manufacture, simple to control and guaranteed to finish the designated task.

3 Block Diagram

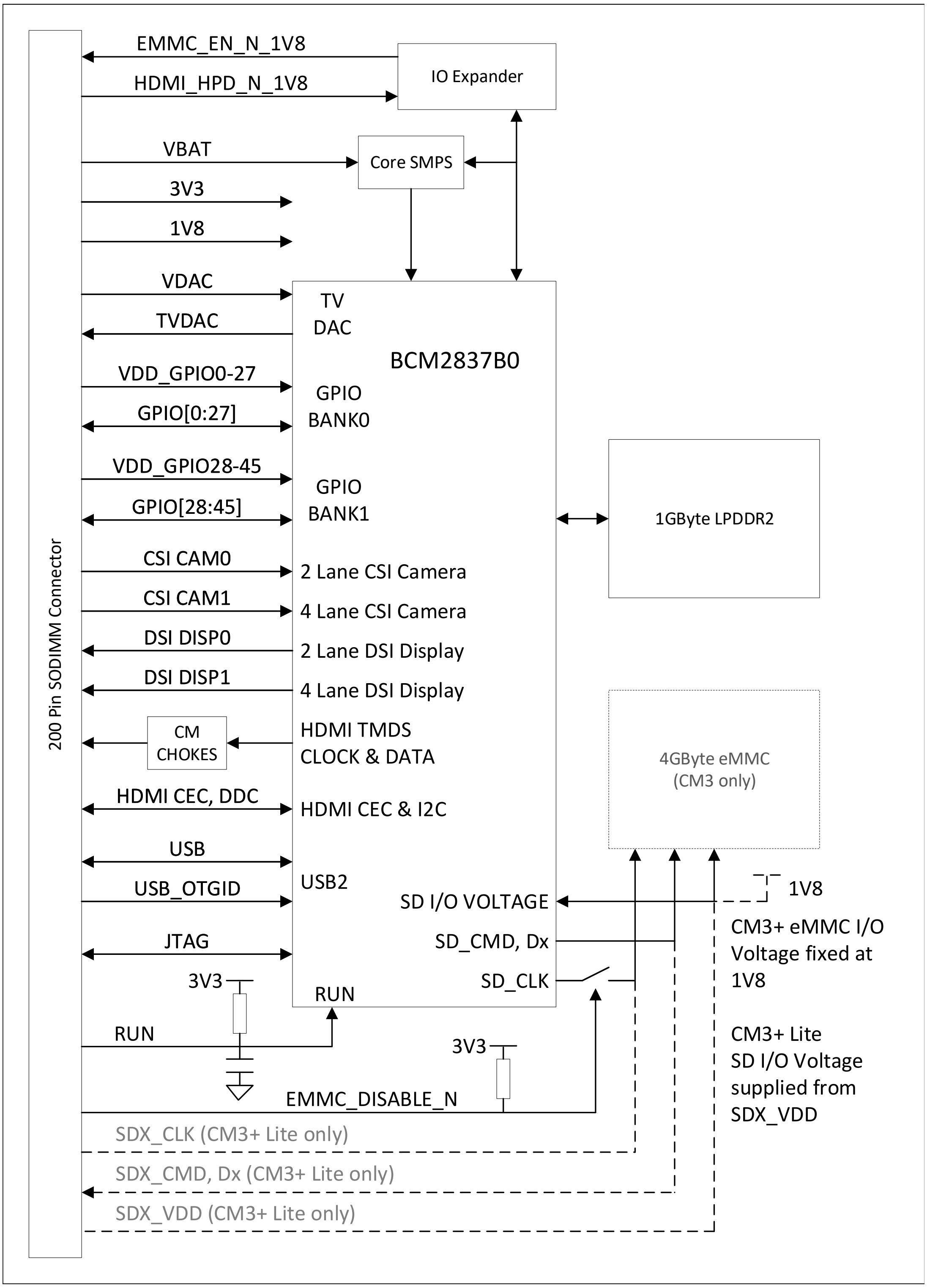


Figure 1: CM3+ Block Diagram

5 Pin Assignments

CM3+	CM3+ Lite	PIN		PIN	CM3+	CM3+ Lite
GND		1		2	EMMC_DISABLE_N	
GPIO0		3		4	NC	SDX_VDD
GPIO1		5		6	NC	SDX_VDD
GND		7		8	GND	
GPIO2		9		10	NC	SDX_CLK
GPIO3		11		12	NC	SDX_CMD
GND		13		14	GND	
GPIO4		15		16	NC	SDX_D0
GPIO5		17		18	NC	SDX_D1
GND		19		20	GND	
GPIO6		21		22	NC	SDX_D2
GPIO7		23		24	NC	SDX_D3
GND		25		26	GND	
GPIO8		27		28	GPIO28	
GPIO9		29		30	GPIO29	
GND		31		32	GND	
GPIO10		33		34	GPIO30	
GPIO11		35		36	GPIO31	
GND		37		38	GND	
GPIO0-27_VDD		39		40	GPIO0-27_VDD	
		KEY				
GPIO28-45_VDD		41		42	GPIO28-45_VDD	
GND		43		44	GND	
GPIO12		45		46	GPIO32	
GPIO13		47		48	GPIO33	
GND		49		50	GND	
GPIO14		51		52	GPIO34	
GPIO15		53		54	GPIO35	
GND		55		56	GND	
GPIO16		57		58	GPIO36	
GPIO17		59		60	GPIO37	
GND		61		62	GND	
GPIO18		63		64	GPIO38	
GPIO19		65		66	GPIO39	
GND		67		68	GND	
GPIO20		69		70	GPIO40	
GPIO21		71		72	GPIO41	
GND		73		74	GND	
GPIO22		75		76	GPIO42	
GPIO23		77		78	GPIO43	
GND		79		80	GND	
GPIO24		81		82	GPIO44	
GPIO25		83		84	GPIO45	
GND		85		86	GND	
GPIO26		87		88	HDMI_HPD_N_1V8	
GPIO27		89		90	EMMC_EN_N_1V8	
GND		91		92	GND	
DSI0_DN1		93		94	DSI1_DP0	
DSI0_DP1		95		96	DSI1_DN0	
GND		97		98	GND	
DSI0_DN0		99		100	DSI1_CP	
DSI0_DP0		101		102	DSI1_CN	
GND		103		104	GND	
DSI0_CN		105		106	DSI1_DP3	
DSI0_CP		107		108	DSI1_DN3	
GND		109		110	GND	
HDMI_CLK_N		111		112	DSI1_DP2	
HDMI_CLK_P		113		114	DSI1_DN2	
GND		115		116	GND	
HDMI_D0_N		117		118	DSI1_DP1	
HDMI_D0_P		119		120	DSI1_DN1	
GND		121		122	GND	
HDMI_D1_N		123		124	NC	
HDMI_D1_P		125		126	NC	
GND		127		128	NC	
HDMI_D2_N		129		130	NC	
HDMI_D2_P		131		132	NC	
GND		133		134	GND	
CAM1_DP3		135		136	CAM0_DP0	
CAM1_DN3		137		138	CAM0_DN0	
GND		139		140	GND	
CAM1_DP2		141		142	CAM0_CP	
CAM1_DN2		143		144	CAM0_CN	
GND		145		146	GND	
CAM1_CP		147		148	CAM0_DP1	
CAM1_CN		149		150	CAM0_DN1	
GND		151		152	GND	
CAM1_DP1		153		154	NC	
CAM1_DN1		155		156	NC	
GND		157		158	NC	
CAM1_DP0		159		160	NC	
CAM1_DN0		161		162	NC	
GND		163		164	GND	
USB_DP		165		166	TVDAC	
USB_DM		167		168	USB_OTGID	
GND		169		170	GND	
HDMI_CEC		171		172	VC_TRST_N	
HDMI_SDA		173		174	VC_TDI	
HDMI_SCL		175		176	VC_TMS	
RUN		177		178	VC_TDO	
DD_CORE (DO NOT CONNECT		179		180	VC_TCK	
GND		181		182	GND	
1V8		183		184	1V8	
1V8		185		186	1V8	
GND		187		188	GND	
VDAC		189		190	VDAC	
3V3		191		192	3V3	
3V3		193		194	3V3	
GND		195		196	GND	
VBAT		197		198	VBAT	
VBAT		199		200	VBAT	

Table 2: Compute Module 3+ SODIMM Connector Pinout

Table 2 gives the Compute Module 3+ pinout and Table 3 gives the pin functions.

Pin Name	DIR	Voltage Ref	PDN[a] State	If Unused	Description/Notes
RUN and Boot Control (see text for usage guide)					
RUN	I	3V3[b]	Pull High	Leave open	Has internal 10k pull up
EMMC_DISABLE_N	I	3V3[b]	Pull High	Leave open	Has internal 10k pull up
EMMC_EN_N_1V8	O	1V8	Pull High	Leave open	Has internal 2k2 pull up
GPIO					
GPIO[27:0]	I/O	GPIO0-27_VDD	Pull or Hi-Z[c]	Leave open	GPIO Bank 0
GPIO[45:28]	I/O	GPIO28-45_VDD	Pull or Hi-Z[c]	Leave open	GPIO Bank 1
Primary SD Interface[d,e]					
SDX_CLK	O	SDX_VDD	Pull High	Leave open	Primary SD interface CLK
SDX_CMD	I/O	SDX_VDD	Pull High	Leave open	Primary SD interface CMD
SDX_Dx	I/O	SDX_VDD	Pull High	Leave open	Primary SD interface DATA
USB Interface					
USB_Dx	I/O	-	Z	Leave open	Serial interface
USB_OTGID	I	3V3		Tie to GND	OTG pin detect
HDMI Interface					
HDMI_SCL	I/O	3V3[b]	Z[f]	Leave open	DDC Clock (5.5V tolerant)
HDMI_SDA	I/O	3V3[b]	Z[f]	Leave open	DDC Data (5.5V tolerant)
HDMI_CEC	I/O	3V3	Z	Leave open	CEC (has internal 27k pull up)
HDMI_CLKx	O	-	Z	Leave open	HDMI serial clock
HDMI_Dx	O	-	Z	Leave open	HDMI serial data
HDMI_HPD_N_1V8	I	1V8	Pull High	Leave open	HDMI hotplug detect
CAM0 (CSI0) 2-lane Interface					
CAM0_Cx	I	-	Z	Leave open	Serial clock
CAM0_Dx	I	-	Z	Leave open	Serial data
CAM1 (CSI1) 4-lane Interface					
CAM1_Cx	I	-	Z	Leave open	Serial clock
CAM1_Dx	I	-	Z	Leave open	Serial data
DSI0 (Display 0) 2-lane Interface					
DSI0_Cx	O	-	Z	Leave open	Serial clock
DSI0_Dx	O	-	Z	Leave open	Serial data
DSI1 (Display 1) 4-lane Interface					
DSI1_Cx	O	-	Z	Leave open	Serial clock
DSI1_Dx	O	-	Z	Leave open	Serial data
TV Out					
TVDAC	O	-	Z	Leave open	Composite video DAC output
JTAG Interface					
TMS	I	3V3	Z	Leave open	Has internal 50k pull up
TRST_N	I	3V3	Z	Leave open	Has internal 50k pull up
TCK	I	3V3	Z	Leave open	Has internal 50k pull up
TDI	I	3V3	Z	Leave open	Has internal 50k pull up
TDO	O	3V3	O	Leave open	Has internal 50k pull up

[a] The PDN column indicates power-down state (when RUN pin LOW)
[b] Must be driven by an open-collector driver
[c] GPIO have software enabled pulls which keep state over power-down
[d] Only available on Lite variants
[e] The CM will always try to boot from this interface first
[f] Requires external pull-up resistor to 5V as per HDMI spec

Table 3: Pin Functions

6 Electrical Specification

Caution! Stresses above those listed in Table 4 may cause permanent damage to the device. This is a stress rating only; functional operation of the device under these or any other conditions above those listed in the operational sections of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

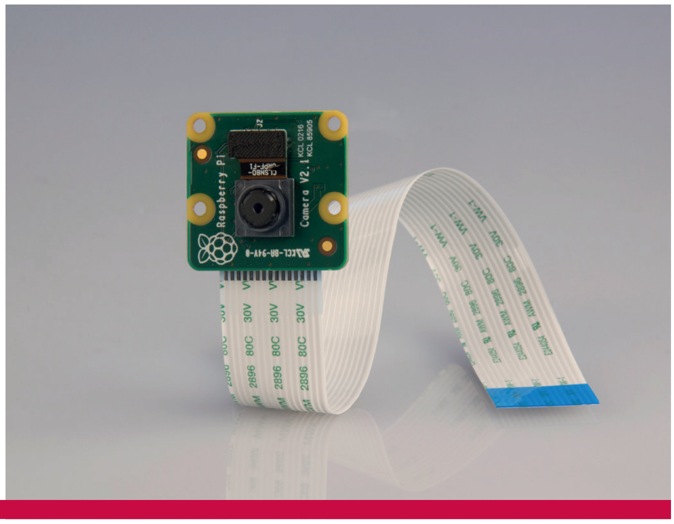
Symbol	Parameter	Minimum	Maximum	Unit
VBAT	Core SMPS Supply	-0.5	6.0	V
3V3	3V3 Supply Voltage	-0.5	4.10	V
1V8	1V8 Supply Voltage	-0.5	2.10	V
VDAC	TV DAC Supply	-0.5	4.10	V
GPIO0-27_VDD	GPIO0-27 I/O Supply Voltage	-0.5	4.10	V
GPIO28-45_VDD	GPIO28-45 I/O Supply Voltage	-0.5	4.10	V
SDX_VDD	Primary SD/eMMC Supply Voltage	-0.5	4.10	V

Table 4: Absolute Maximum Ratings

DC Characteristics are defined in Table 5

Raspberry Pi

Camera Module



Product Name	**Raspberry Pi Camera Module**
Product Description	High Definition camera module compatible with all Raspberry Pi models. Provides high sensitivity, low crosstalk and low noise image capture in an ultra small and lightweight design. The camera module connects to the Raspberry Pi board via the CSI connector designed specifically for interfacing to cameras. The CSI bus is capable of extremely high data rates, and it exclusively carries pixel data to the processor.
RS Part Numer	**913-2664**

Specifications

Image Sensor	Sony IMX 219 PQ CMOS image sensor in a fixed-focus module.
Resolution	8-megapixel
Still picture resolution	3280 x 2464
Max image transfer rate	1080p: 30fps (encode and decode) 720p: 60fps
Connection to Raspberry Pi	15-pin ribbon cable, to the dedicated 15-pin MIPI Camera Serial Interface (CSI-2).
Image control functions	Automatic exposure control Automatic white balance Automatic band filter Automatic 50/60 Hz luminance detection Automatic black level calibration
Temp range	Operating: -20° to 60° Stable image: -20° to 60°
Lens size	1/4″
Dimensions	23.86 x 25 x 9mm
Weight	3g

YF-S201 Water Flow Sensor



This sensor sit in line with your water line, and uses a pinwheel sensor to measure how much liquid has moved through it.

Rating: Not Rated Yet
Price:
Variant price modifier:

Base price with tax:

Price with discount: 800,00 ?

Salesprice with discount:

Sales price: 800,00 ?

Sales price without tax: 800,00 ?

Discount:

Tax amount:

Ask a question about this product

Description

Measure liquid/water flow for your solar, water conservation systems, storage tanks, water recycling home applications, irrigation systems and much more. The sensors are solidly constructed and provide a digital pulse each time an amount of water passes through the pipe. The output can easily be connected to a microcontroller for monitoring water usage and calculating the amount of water remaining in a tank etc.

Features:

- Model: YF-S201
- Working Voltage: 5 to 18V DC (min tested working voltage 4.5V)
- Max current draw: 15mA @ 5V
- Output Type: 5V TTL
- Working Flow Rate: 1 to 30 Liters/Minute
- Working Temperature range: -25 to +80?
- Working Humidity Range: 35%-80% RH
- Accuracy: ±10%
- Maximum water pressure: 2.0 MPa
- Output duty cycle: 50% +-10%
- Output rise time: 0.04us
- Output fall time: 0.18us
- Flow rate pulse characteristics: Frequency (Hz) = 7.5 * Flow rate (L/min)
- Pulses per Liter: 450
- Durability: minimum 300,000 cycles
- Cable length: 15cm
- 1/2" nominal pipe connections, 0.78" outer diameter, 1/2" of thread
- Size: 2.5" x 1.4" x 1.4"

ITEM INCLUDED:

1 x YF-S201 Water Flow Sensor

Reviews

There are yet no reviews for this product.

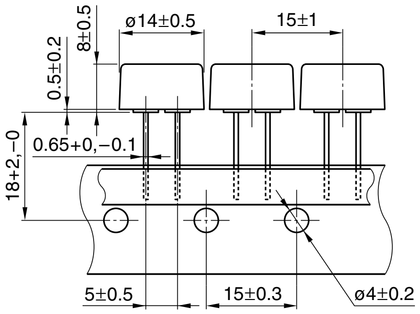
PS14 TYPE

PS1440P02BT

FEATURES

- High sound pressure.
- Miniature size(ø14×T8mm).
- Suitable for automatic radial taping machine(15mm-pitch).

SHAPES AND DIMENSIONS



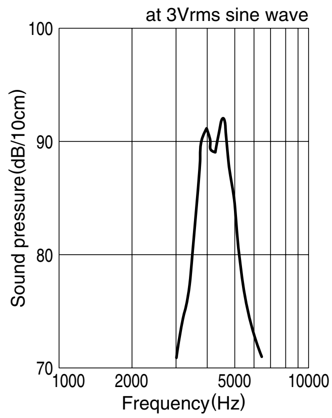
Dimensions in mm

SPECIFICATIONS AND CHARACTERISTICS

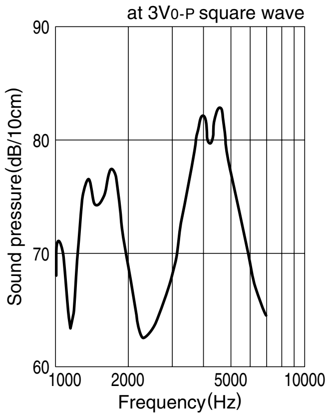
Sound pressure	75dBA/ 10cm min.	[at 4kHz, $3V_{0\text{-}P}$ rectangular wave, measuring temperature: 25±5°C, humidity: 60±10%]
Operating temperature range	–10 to +70°C	
Storage conditions	+5 to +40°C, 20 to 70%RH, please use within 6 months	
Maximum input voltage	$30V_{0\text{-}P}$ max.	[without DC bias]
Minimum delivery unit	1750 pieces	[350 pieces/1 reel×5 reels]

FREQUENCY SOUND PRESSURE CHARACTERISTICS

SINE WAVE DRIVE



SQUARE WAVE DRIVE

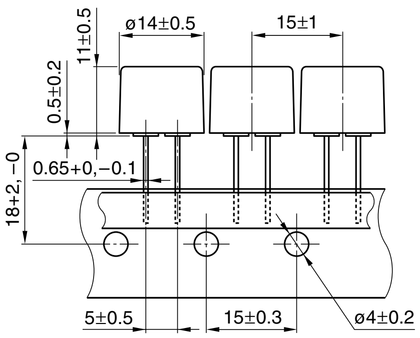


PS1420P02CT

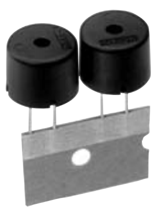
FEATURES

- Low frequency tone(2kHz).
- Suitable for automatic radial taping machine(15mm-pitch).

SHAPES AND DIMENSIONS



Dimensions in mm

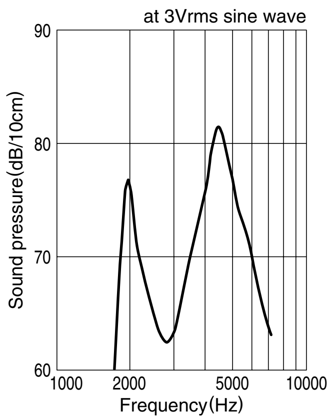


SPECIFICATIONS AND CHARACTERISTICS

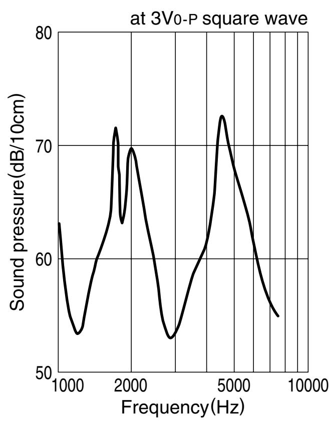
Sound pressure	70dBA/ 10cm min.	[at 2kHz, $5V_{0\text{-}P}$ rectangular wave, measuring temperature: 25±5°C, humidity: 60±10%]
Operating temperature range	–10 to +70°C	
Storage conditions	+5 to +40°C, 20 to 70%RH, please use within 6 months	
Maximum input voltage	$30V_{0\text{-}P}$ max.	[without DC bias]
Minimum delivery unit	1750 pieces	[350 pieces/1 reel×5 reels]

FREQUENCY SOUND PRESSURE CHARACTERISTICS

SINE WAVE DRIVE



SQUARE WAVE DRIVE



References

- IoT Based Controlling and Monitoring of Smart City, P. Sai Bhavani Prasad,Mahantesh H Arutagi, Sanket Bandi, Shridhar V. Nayak , Dr. Iranna Korachagaon, IJARET, Vol. 4, Issue 5 , May 2017, ISSN: 2350-0328.

- Smart Real-Time Drainage Monitoring System Using Internet of Things, Gaurang Sonawane, Chetan Mahajan, Anuja Nikale, Yogita Dalvi , Dept. of Computer Engineering, Late G. N. Sapkal College of Engineering, Nashik, MAY 2018, IRE Journals, Volume 1 Issue 11, ISSN: 2456-8880 .

- IOT based Smart Home Security System with Alert and Door Access Control using Smart Phone, Shaik Anwar, Prof.D. Kishore, Department of Electronics and Communication Engineering, Aditya College of Engineering and Technology, Surampallem, Andhra Pradesh, India, Published by International Journal of Engineering Research & Technology (IJERT), Vol. 5 Issue 12, December-2016, ISSN: 2278-0181.

- Effectiveness of Sewage Treatment System and its Irrigation Potential: the Case of Main Campus of Hawassa University, Ethiopia, Kannan Narayanan and Brook Abate Hawassa University, Institute of Technology, School of Biosystems and Environmental Engineering, Published International Journal of Engineering Research & Technology (IJERT), ISSN: 2278-0181, Vol. 6 Issue 10, October – 2017.

- The Internet of Things, Enabling Technologies and Use Cases, Pethuru Raj and Anupama C.Rajan, CRC Press.

- An NPTEL online certification course on ‘Introduction to Internet of Things’ by Prof. Sudip Misra, IIT Kharagpur (URL for video lectures: https://nptel.ac.in/courses/106105166/).

- The MagPi Essentials: Conquer the Command Line (The Raspberry Pi Terminal Guide) by Richard Smedley, 2nd Edition.

- The MagPi Essentials: The Camera Module Guide (Take Pictures and Capture Video with your Raspberry Pi) by Phil King.

- The MagPi Essentials: Simple Electronics with GPIO Zero (Take control of the real world with your Raspberry Pi) by Phil King.