Raspberry Pi-Smart
Health Tracker Alert Band

A Main Project submitted
in partial fulfillment of the requirements
for the award of the degree of

BACHELOR OF TECHNOLOGY

In

COMPUTER SCIENCE AND ENGINEERING

Submitted by

1. **P. Dikshetha (20PA1A05D5)**
2. **S. Hemanth Kumar (20PA1A05G0)**
3. **U. Mahesh (20PA1A05H7)**
4. **S. Prashanth Nayak (20PA1A05E2)**
5. **V. Uttej (20PA1A05H9)**

Under the esteemed guidance of
J.N.S.S JANARDHANA NAIDU
Assistant Professor

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

VISHNU INSTITUTE OF TECHNOLOGY
(Autonomous)
(Approved by AICTE, Accredited by NBA & NAAC and permanently affiliated to JNTU Kakinada)
BHIMAVARAM – 534202

2023 – 2024

VISHNU INSTITUTE OF TECHNOLOGY

(Autonomous)

(Approved by AICTE, accredited by NBA & NAAC, and permanently affiliated to JNTU Kakinada)

BHIMAVARAM-534202

2023-2024

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CERTIFICATE

This is to certify that the project entitled "**Raspberry Pi-Smart Health Tracker Alert Band**", is being submitted by ***P.DIKSHETHA, S.HEMANTH KUMAR, U.MAHESH, S.PRASHANTH NAYAK, V.UTTEJ***, bearing the **REGD.NOS: 20PA1A05D5, 20PA1A05G0, 20PA1A05H7, 20PA1A05E2, 20PA1A05H9** submitted in fulfillment for the award of the degree of **"BACHELOR OF TECHNOLOGY"** in **"COMPUTER SCIENCE AND ENGINEERING"** is a record of work carried out by them under my guidance and supervision during the academic year 2023-2024 and it has been found worthy of acceptance according to the requirements of university.

Internal Guide | **Head of the Department**

J.N.S.S. Janardhana Naidu | Dr. Sumit Gupta

External Examiner

ACKNOWLEDGEMENT

It is natural and inevitable that the thoughts and ideas of other people tend to drift into the subconscious due to various human parameters, where one feels the need to acknowledge the help and guidance derived from others. We express our gratitude to those who have contributed to the fulfillment of this project.

We take the opportunity to extend our sincere thanks to **Dr. D. Suryanaryana**, the director of VIT, Bhimavaram, whose guidance from time to time helped us complete this project successfully.

We also express our sincere gratitude to **Dr. M. Venu**, the principal of VIT, Bhimavaram, for his continuous support in helping us complete the project on time.

We are thankful to **Dr. Sumit Gupta**, Head of the Department of Computer Science and Engineering, for his continuous and unwavering support and guidance. We acknowledge our gratitude for his valuable guidance and support extended to us from the conception of the idea to the completion of this project.

We are very thankful to **J.N.S.S. Janardhana Naidu**, Assistant Professor, our internal guide, whose guidance from time to time helped us complete this project successfully.

Project Associates:

P. Dikshetha (20PA1A05D5)

S. Hemanth Kumar (20PA1A05G0)

U. Mahesh (20PA1A05H7)

S. Prashanth Nayak (20PA1A05E2)

V. Uttej (20PA1A05H9)

ABSTRACT

In the recent development of, Internet of Things (IoT) makes all objects interconnected and it has been recognized as the next technical revolution. Some of the applications of Internet of Things are smart parking, smart home, smart environment, industrial places and health monitoring process. One such application is in healthcare to monitor the patient's health status. The Internet of Things makes medical equipment more efficient by allowing real time monitoring of patient health, in which sensors acquire data of patients. In the Internet of Things patient parameters get transmitted through medical devices via a gateway, where it is stored and analyzed. The significant challenge in the implementation of Internet of Things for healthcare applications is monitoring all patients from various places. Thus, the Internet of Things in the medical field brings out the solution for effective patient monitoring at reduced cost and also reduces the trade-off between patient outcome and disease management. In this project we discuss monitoring a patient's body temperature, surrounding humidity, temperature and heart beat using Raspberry Pi board. A buzzer is utilized to alert when there is sudden increase or decrease in heart rate, and an alert message is sent to specified mobile numbers. And the data is analyzed and the results will display as a dashboard in the website Ubidots and a web page created by us using PHP.

TABLE OF CONTENTS

LIST OF DIAGRAMS

LIST OF TABLES

CHAPTER 1

INTRODUCTION:

The unpredictable growth of the "Internet of Things" is changing the world and the rapid drop in price for typical IoT components is allowing the public to innovate new designs and products at home. IoT can be used in monitoring patient's health.The unexpected occurrence in patient health is monitored using IoT. In this paper specialized sensors are used to monitor a patient's heart rate, body temperature. One of the key learning platforms for IoT is the Raspberry Pi. The Raspberry Pi is a popular platform because it offers a complete Linux server in a tiny platform for a very low cost. The Raspberry Pi also allows interfacing services and actuators through the general purpose I/O pins. The combination of Raspberry Pi and IoT becomes a new innovation technology in the healthcare system. Raspberry Pi acts as a small clinic after connecting sensors.Raspberry Pi will collect data from sensors and then it transfers wirelessly to the IoT website. Raspberry Pi board is connected to the internet, that board MAC address is registered to the internet. After that in the IoT website, add the MAC address of this board. Then the sensor's output is connected to the IoT website.

1.1 PROJECT OVERVIEW:

Patient Monitoring in Real Time Using IoT Based Cloud The proposed system helps to monitor the health condition of the patient using sensors which are connected to the network. The biological behaviors of patients can be gathered by using different sensors thereby sending the biological data to the IoT based cloud. The critical condition of a patient can be detected by processing the data inside the sensor and also providing instant push notification to doctors or patients family members, thus making it an intelligent patient monitoring system. This system helps to observe the patient remotely even without visiting them. The hardware components include sensors for temperature and humidity, and heart rate. The temperature and humidity sensors are used for measuring the body temperature and humidity. The heart rate sensor helps to measure the pulse. And this data is analyzed and based on the heart rate message is sent to the specified phone numbers. Using this application, even without visiting the ICU unit, the doctors or nurses can monitor their patients remotely. The paper is able to monitor the real-time patient in ICU thereby improving the efficiency and quality of service. The advantage is that the system is fully automatic and hence provides constant remote monitoring facilities. Also, the proposed system is less prone to errors and is therefore more accurate.

1.2 MOTIVATION:

IoT-based systems are in charge of providing knowledge from an environment to a non-expert user.
IoT-based systems can be used in different environments, so it needs to be able to address many heterogeneous devices. Thus, a major concern within developing an IoT-based system is how to handle the interaction with the heterogeneous devices for non-expert users. This concern can be addressed by a middleware layer between devices and non-expert users. This layer is responsible for hiding the diversity of devices from the user perspective, and provides access transparency to the devices for the end users.
The idea of creating abstractions of devices has been addressed in the literature. The middleware we found in the literature can provide satisfaction by facilitating the interaction with devices, but they do not support low-level device configuration.

1.3 BACKGROUND:

The Future Internet goal is to provide an infrastructure to have immediate access to information about the physical world and its objects. Physical objects can be applicable to different application domains, such as e-health, warehouse management, etc. Each application domain may have different types of physical devices. Each physical device can have its own specifications, which is required to use in order to interact with it. To achieve the future Internet goal, a layered vision is required that can facilitate data access. Internet of Things (IoT) is a vision that aims to integrate the virtual world of information to the real world of devices through a layered architecture.

The term "Internet of Things‟ consists of two words, namely Internet and Things. Internet refers to the global network infrastructure with scalable, configurable capabilities based on interoperable and standard communication protocols. Things are physical objects or devices, or virtual objects, devices or information, which have identities, physical attributes and virtual personalities, and use intelligent interfaces. For instance, a virtual object can represent an abstract unit of sensor nodes that contains metadata to identify and discover its corresponding sensor nodes. Therefore, IoT refers to the things that can provide information from the physical environment through the Internet.

IOT DEFINITION:

In this section, we explain some of the IoT definitions. Also, we explain the layered architecture for IoT.The Internet of Things (IoT) has increasingly gained attention in industry to interact with different types of devices. IoT can have influence on industry and society by integrating physical devices into information networks. IoT impacts can be on different perspectives, namely for private and business users. From the perspective of a private user, IoT has an effect on both working and personal fields, such as smart homes and offices, e-health and assisted living. From the aspect of a business user the impacts would be in fields such as automation and industrial manufacturing, logistics, business process management, intelligent transportation of people and goods.

The Internet of Things (IoT) refers to the network of physical devices, embedded with sensors, software, and other technologies, that are able to connect and exchange data with each other over the internet. These devices can range from ordinary household objects like refrigerators and thermostats to sophisticated industrial tools like robots and sensors in manufacturing plants.

IoT integrates physical things into information networks. IoT covers the overall infrastructure, including software, hardware and services, which is used to support these information networks. The integrated physical things can exchange data about the physical properties and information that they sense in their environment. To identify devices, we can use identification technologies like for example RFID, which allow each device to be uniquely identified. These services are provided through the middleware layer to different applications and users in IoT-based systems. The application services can be used in different industries such as, logistics, retail, healthcare, etc

Characteristics of the Internet of Things:

Connectivity: Devices communicate and exchange data wirelessly or through wired connections.

Data collection: Sensors embedded in devices collect data about their environment, such as temperature, pressure, or location.

Data analysis: The collected data can be analyzed to gain insights, improve efficiency, and automate tasks.

Automation: Devices can be programmed to take actions based on the data they collect, enabling automation in various fields.

The growth of the IoT is fueled by several factors:

Decreasing cost of sensors and computing power: Makes it more affordable to embed intelligence into everyday objects.Advancements in wireless technologies: Enables seamless connectivity for a wider range of devices.Cloud computing: Provides a platform for storing and analyzing the massive amount of data generated by IoT devices.

CHAPTER 2

LITERATURE SURVEY:

Health has become one of the global challenges for humanity. Cardiac diseases, Lung failures and heart related diseases are increasing at a rapid rate. Monitoring the health of elderly people at home or patients at hospitals is necessary but it requires constant observation of Practitioners and Doctors. Information Technology (IT) and its growing applications are performing a major role in making human life easier. The Internet of Things (IoT) is transforming healthcare and the role of IT in healthcare. IoT consists of physical devices, such as sensors and monitoring devices for patients (glucose, blood pressure, heart rate & activity monitoring, etc) to connect to the internet and transform information from the physical world into the digital world. The proposed system, with the help of IoT's such features, will help to keep the necessary details and reports of a patient organized and available to all actors in the system. IoT devices like low power sensors will be used to collect data from patients and it will be displayed using LCD and stored on any personal computer and also on the cloud so that any actor in the system can refer to it.

The world of medical science is an emerging area that has accelerated with new technologies and this is the time when the vision of “The internet of things (IoT)” has turned into reality. IoT can play a significant role in the healthcare domain by managing chronic diseases at one end as well as preventing diseases on the other hand. People demand more care at reduced clinical costs, remote health monitoring (using IoT) is one of the possible solutions to this demand. Remote health monitoring can be best utilized provided the device is wearable to facilitate self-monitoring. In this paper, we propose a system for monitoring the pulse rate, body temperature (vital body parameters) of the person with dedicated sensors along with Raspberry pi and IoT. A system is wearable and also supports remote health monitoring. Remote Health monitoring is attained by storing the collected data to Bluemix cloud, this data can be retrieved by the doctor for analysis anywhere and any aberrancy will be timely detected. Bluemix Uses MQTT(Message Queuing Telemetry Transport) protocol. Along with remote monitoring and wearability of the system, accuracy and cost cannot be ignored. A perfect tradeoff between accuracy and cost of the system is accomplished by choosing appropriate sensors which are DS18B20 (temperature sensor) and KG011 (heart rate sensor). The pulse rate and temperature of a person at different time instants are measured by the sensors. The readings are shown in the form of graphs at IBM Watson IoT platform.

In the recent development of, Internet of Things (IoT) makes all objects interconnected and it has been recognized as the next technical revolution. Some of the applications of Internet of Things are smart parking, smart home, smart city, smart environment, industrial places, agriculture fields and health monitoring process. One such application is in healthcare to monitor the patient's health status. The Internet of Things makes medical equipment more efficient by allowing real time monitoring of patient health, in which sensors acquire data of patient's and reduces human error. In the Internet of Things, a patient's parameters get transmitted through medical devices via a gateway, where it is stored and analyzed. The significant challenges in the implementation of Internet of Things for healthcare applications is monitoring all patient's from various places. Thus the Internet of Things in the medical field brings out the solution for effective patient monitoring at reduced cost and also reduces the trade-off between patient outcome and disease management. In this paper, we discuss monitoring a patient's body temperature, respiration rate, heart beat and body movement using a Raspberry Pi board.

CHAPTER 3

3.1 HARDWARE AND SOFTWARE REQUIREMENTS:

Hardware Requirements:

- Power Supply
- Raspberry PI
- Buzzer
- Temperature sensor
- Heart Beat sensor

Software Requirements:

- ARDUINO IDE
- Language: Embedded C
- Python

3.2 EXISTING SYSTEM:

In the traditional approach the healthcare professionals play the major role. They need to visit the patient's ward for necessary diagnosis and advising. There are two basic problems associated with this approach. Firstly, the healthcare professionals must be present on site of the patient all the time.

Secondly, the patient remains admitted in a hospital, bedside biomedical instruments, for a period of time. In order to solve these two problems, the patients are given knowledge and information about disease diagnosis and prevention. Secondly, a reliable and readily available patient monitoring system (PMS) is required.

DRAWBACKS:

- In small hospitals this method is possible, but when it comes to big hospitals it is not possible.
- We cannot maintain a database of a patient in hardcopy if any accidents like fire or file misplacement occurs.

- As more and more users are tapped in with their devices, the clogged airwaves make it difficult to latch on a reliable signal.

3.3 PROPOSED SYSTEM:

In this proposed idea the sensors gather the medical information of the patient that includes the patient's heart rate, Temperature. Then monitoring patients through the Raspberry kit and this information is sent to the Internet and stored in a server and analyzed.

We have proposed a robust health monitoring system that is intelligent enough to monitor the patient automatically using IOT. It collects the status information through these systems which would include patient temperature, heart rate, blood pressure and sends an emergency alert to the patient doctor as well as to the caretaker with his current status.

This would help the doctor and caretaker to monitor his patient from anywhere in the world. The system uses smart sensors that generate raw data information collected from each sensor and send it to a cloud server where the data can be further analyzed and statistically maintained to be used. The proposed method of the patient monitoring system monitors the patient's health parameters using Raspberry Pi.

After connecting the internet to the Raspberry Pi it acts as a server. Then the server automatically sends data to the website. Using an IP address anybody can monitor the patient’s health status anywhere in the world using laptops, tablets and smartphones.

3.4 FEASIBILITY STUDY:

Economic Feasibility:

This project involves assessing the cost-effectiveness of developing and maintaining the health tracker with Raspberry Pi, considering potential benefits and market value.

Technical Feasibility:

Seamless integration of heart rate, and temperature sensors with Raspberry Pi and Edge Computing.

- Implementation of a robust IoT framework for real-time data transmission and remote monitoring.

CHAPTER 4

SYSTEM DESIGN:

4.1 Block Diagram:

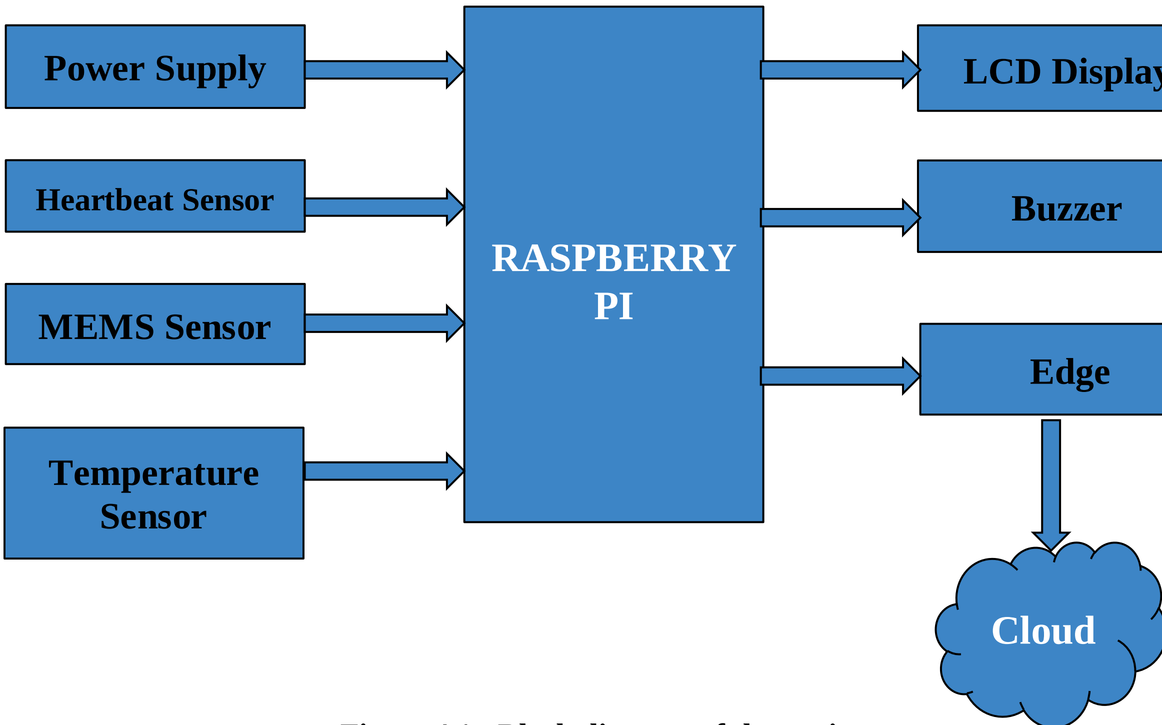


Figure 4.1 : Block diagram of the project

4.2 About Raspberry Pi:

Raspberry Pi is defined as a minicomputer the size of a credit card that is interoperable with any input and output hardware device like a monitor, a television, a mouse, or a keyboard – effectively converting the set-up into a full-fledged PC at a low cost. This article discusses the Raspberry Pi models available today, their key features, and use cases. Raspberry Pi is a small single board computer. By connecting peripherals like Keyboard, mouse, display to the Raspberry Pi, it will act as a mini personal computer.

Raspberry Pi is popularly used for real time Image/Video Processing, IoT based applications and Robotics applications.

Raspberry Pi is slower than laptop or desktop but is still a computer which can provide all the expected features or abilities, at a low power consumption.

Raspberry Pi Foundation:

A listed institutional charity (registration number 1129409) located in the UK is the Raspberry Pi Foundation. The target of the Foundation is to emphasize on the education of adults, and children.

The Foundation of Raspberry Pi is an aid based in the United Kingdom that aims to get the authority of computing and digital development into the hands of people in the world. The target is for more people to leverage the power of computing and emerging technology for work, solve and express themselves creatively in order to solve concerns that really matter.

OS for Raspberry Pi:

Raspberry Pi OS (formerly Raspbian) is a Unix-like operating system based on the Debian GNU/Linux distribution for the Raspberry Pi family of compact single-board computers. First developed independently in 2012, it has been produced as the primary operating system for these boards since 2013, distributed by the Raspberry Pi Foundation.

The Raspberry Pi operates in the open source ecosystem: it runs Linux (a variety of distributions), and its main supported operating system, Pi OS, is open source and runs a suite of open source software. The Raspberry Pi Foundation contributes to the Linux kernel and various other open source projects as well as releasing much of its own software as open source.

Raspberry Pi Foundation officially provides Debian based Raspbian OS. Also, they provide NOOBS OS for Raspberry Pi. We can install several Third-Party versions of OS like Ubuntu, Archlinux, RISC OS, Windows 10 IOT Core, etc.

Raspbian OS is an official Operating System available for free to use. This OS is efficiently optimized to use with Raspberry Pi. Raspbian has a GUI which includes tools for Browsing, Python programming, office, games, etc.

Raspberry Pi is more than computer as it provides access to the on-chip hardware i.e. GPIOs for developing an application. By accessing GPIO, we can connect devices like LED, motors, sensors, etc and can control them too.

Working of Raspberry Pi

Raspberry Pi is a programmable device. It comes with all the critical features of the motherboard in an average computer but without peripherals or internal storage. To set up the Raspberry computer, you will need an SD card inserted into the provided space. The SD card should have the operating system installed and is required for the computer to boot. Raspberry computers are compatible with Linux OS. This reduces the amount of memory needed and creates an environment for diversity.

After setting up the OS, one can connect Raspberry Pi to output devices like computer monitors or a High-Definition Multimedia Interface (HDMI) television. Input units like mice or keyboards should also be connected. This minicomputer's exact use and applications depend on the buyer and can cover many functions.

Raspberry Pi processor

The BCM2835 is the Broadcom chip used in the Raspberry Pi 1 Models A, A+, B, B+, the Raspberry Pi Zero, the Raspberry Pi Zero W, and the Raspberry Pi Compute Module 1. Some details of the chip can be found in the peripheral specification document. It contains a single-core ARM1176JZF-S processor.

Figure 4.2 : Raspberry pi processor

It has ARM based Broadcom Processor SoC along with on-chip GPU (Graphics Processing Unit).

- The CPU speed of Raspberry Pi varies from 700 MHz to 1.2 GHz. Also, it has on-board SDRAM that ranges from 256 MB to 1 GB. All Raspberry Pi models use **ARM processors**, known for their efficient power consumption and versatility.
- The processor type and speed have significantly increased over generations, reflecting advancements in technology and the need for more demanding applications.
- The choice of Raspberry Pi model depends on your project's requirements. For basic tasks, an older model might suffice, while performance-intensive projects might benefit from the latest models with more powerful processors.

Raspberry Pi also provides on-chip SPI, I2C, I2S and UART modules

CHAPTER 5

MODULE DESCRIPTIONS:

5.1 VERSIONS OF RASPBERRY PI MODELS:

There are different versions of raspberry pi available as listed below:

1. **Raspberry Pi 1 Model A**
2. **Raspberry Pi 1 Model A+**
3. **Raspberry Pi 1 Model B**
4. **Raspberry Pi 1 Model B+**
5. **Raspberry Pi 2 Model B**
6. **Raspberry Pi 3 Model B**
7. **Raspberry Pi Zero**

Out of the above versions of Raspberry Pi, more prominently use Raspberry Pi and their features are as follows:

Features	**RaspberryPi Model B+**	**Raspberry Pi 2 Model B**	**Raspberry Pi 3 Model B**	**RaspberryPi zero**
SoC	BCM2835	BCM2836	BCM2837	BCM2835
CPU	ARM11	Quad Cortex A7	Quad Cortex A53	ARM11
Operating Freq.	700 MHz	900 MHz	1.2 GHz	1 GHz
RAM	512 MB SDRAM	1 GB SDRAM	1 GB SDRAM	512 MB SDRAM
GPU	250 MHz Videocore IV	250MHz Videocore IV	400 MHz Videocore IV	250MHz Videocore IV
Storage	micro-SD	Micro-SD	micro-SD	micro-SD
Ethernet	Yes	Yes	Yes	No

Wireless	WiFi and Bluetooth	No	No	No

Table 5.1 : Versions of Raspberry pi

5.2 RASPBERRY PI BOARD:

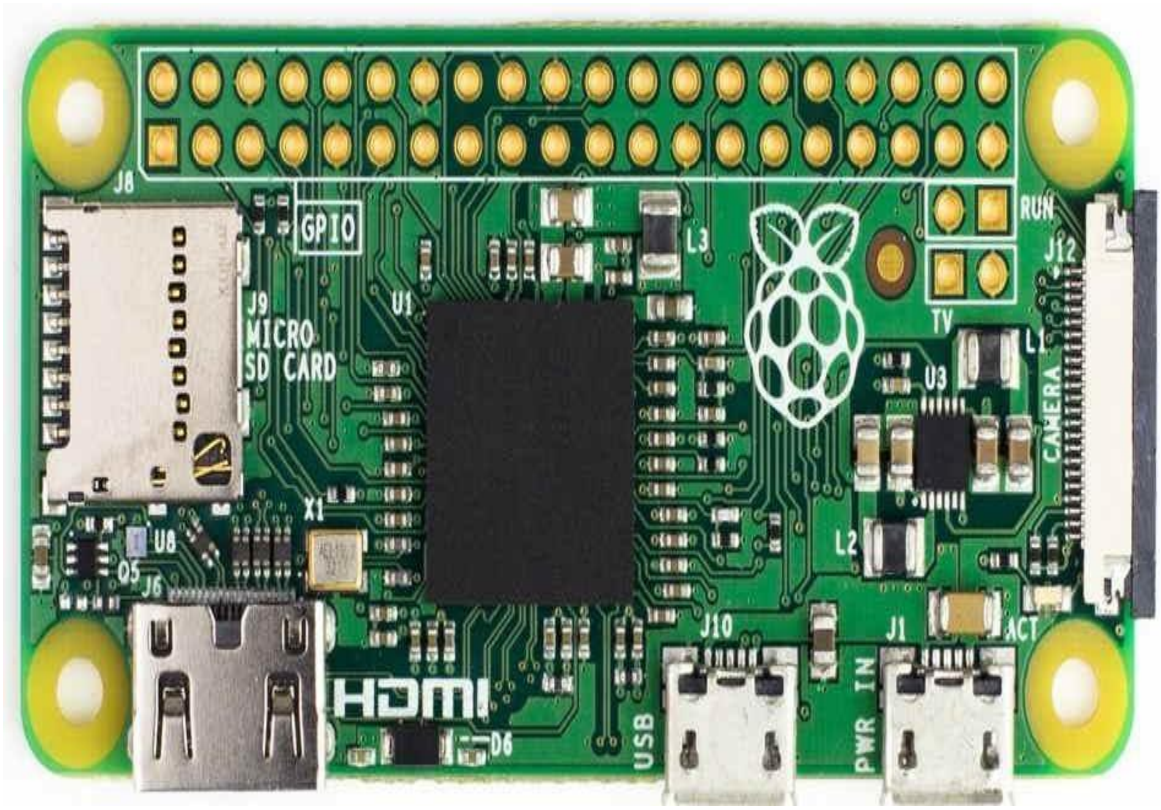


Figure 5.1:Raspberry pi Board

Raspberry Pi 3 Hardware Details

The On-chip hardware of Raspberry Pi 3 (here) is as shown in below figure,

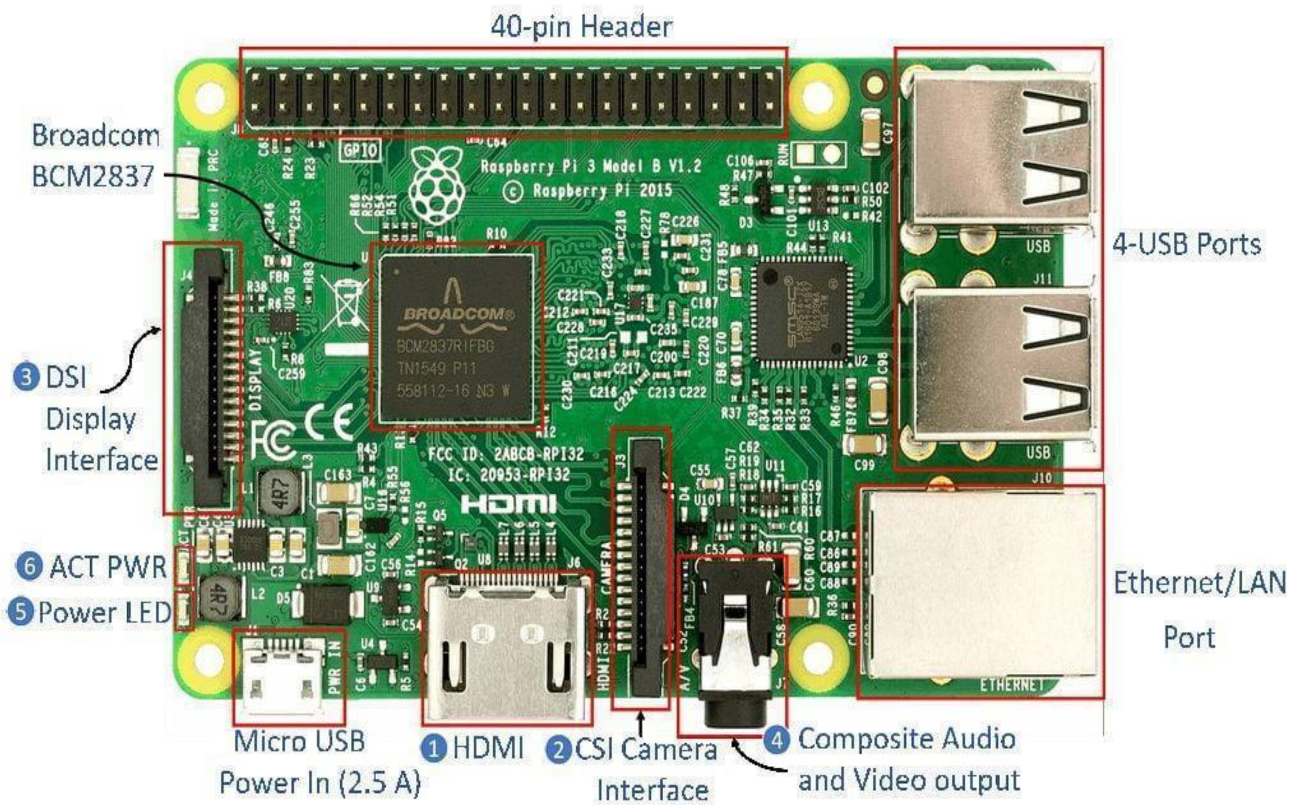


Figure 5.2:Raspberry pi hardware details

Some Hardware Components shown above are mention below:

1. **HDMI (High-Definition Multimedia Interface):** It is used for transmitting uncompressed video or digital audio data to the Computer Monitor, Digital TV, etc. Generally, this HDMI port helps to connect Raspberry Pi to the Digital television.
2. **CSI Camera Interface:** CSI (Camera Serial Interface) interface provides a connection in between Broadcom Processor and Pi camera. This interface provides electrical connections between two devices.
3. **DSI Display Interface:** DSI (Display Serial Interface) Display Interface is used for connecting LCD to the Raspberry Pi using 15-pin ribbon cable. DSI provides a fast High-resolution display interface specifically used for sending video data directly from GPU to the LCD display.

4. **Composite Video and Audio Output:** The composite Video and Audio output port carries video along with audio signal to the Audio/Video systems.
5. **Power LED:** It is a RED colored LED which is used for Power indication. This LED will turn ON when Power is connected to the Raspberry Pi. It is connected to 5V directly and will start blinking whenever the supply voltage drops below 4.63V.
6. **ACT PWR:** ACT PWR is Green LED which shows the SD card activity.

5.3 RASPBERRY PI GPIO:

GPIO (General Purpose Input Output) pins can be used as input or output and allows raspberry pi to connect with general purpose I/O devices.

- Raspberry pi 3 model B took out 26 GPIO pins on board.
- Raspberry pi can control many external I/O devices using these GPIO's.
- These pins are a physical interface between the Pi and the outside world.
- We can program these pins according to our needs to interact with external devices. For example, if we want to read the state of a physical switch, we can configure any of the available GPIO pins as input and read the switch status to make decisions. We can also configure any GPIO pin as an output to control LED ON/OFF.
- Raspberry Pi can connect to the Internet using on-board Wi-Fi or Wi-Fi USB adapter. Once the Raspberry Pi is connected to the Internet then we can control devices, which are connected to the Raspberry Pi, remotely.

WM (Pulse Width Modulation) is a modulation technique by which the width of pulse is varied while keeping the frequency constant.

- Through PWM technique, we can control the power delivered to the load by using ON-OFF signal.
- The PWM signals can be used for applications such as controlling the speed of DC motors, changing intensity of an LED, controlling Servo motors, etc.
- The GIF shown below depicts the use of PWM for intensity control of an LED.

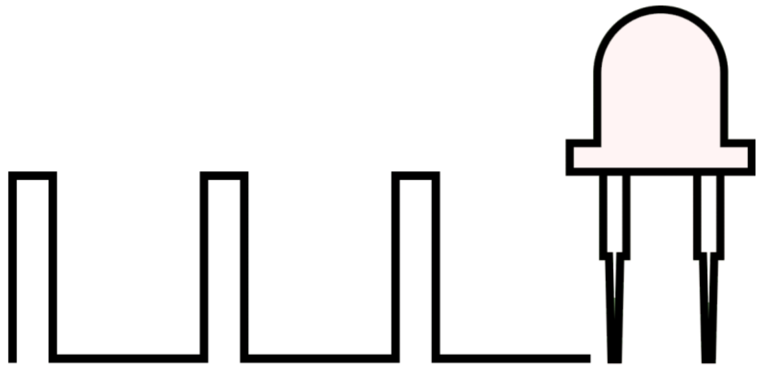


Figure 5.3: LED bulb ON/OFF

Working: To use GPIO pins in your projects, you'll need to:

Connect your external devices to the relevant GPIO pins using jumper wires and potentially additional components like resistors (depending on the device).

Write code using programming languages like Python and libraries like RPi.GPIO to control the pins as inputs or outputs. These libraries provide functions to read data from input pins and write data to output pins.

Remember, using GPIO pins involves some level of electronics knowledge and programming expertise. However, numerous online resources and tutorials can guide you through the process, making it a great way to learn about electronics and physical computing with your Raspberry Pi.

5.4 RASPBERRY PI PWM:

- Raspberry Pi has two PWM channels i.e. PWM0 and PWM1.
- PWM pins for the two PWM channels on 40-pin P1 header are as follows:

GPIO Pin	**PWM0/PWM1**
GPIO12	PWM0
GPIO18	PWM0
GPIO13	PWM1
GPIO19	PWM1

Table 5.2 : GPIO pins and PWM channels

- The PWM pins on Raspberry Pi 40-pin P1 Header is shown in below figure,

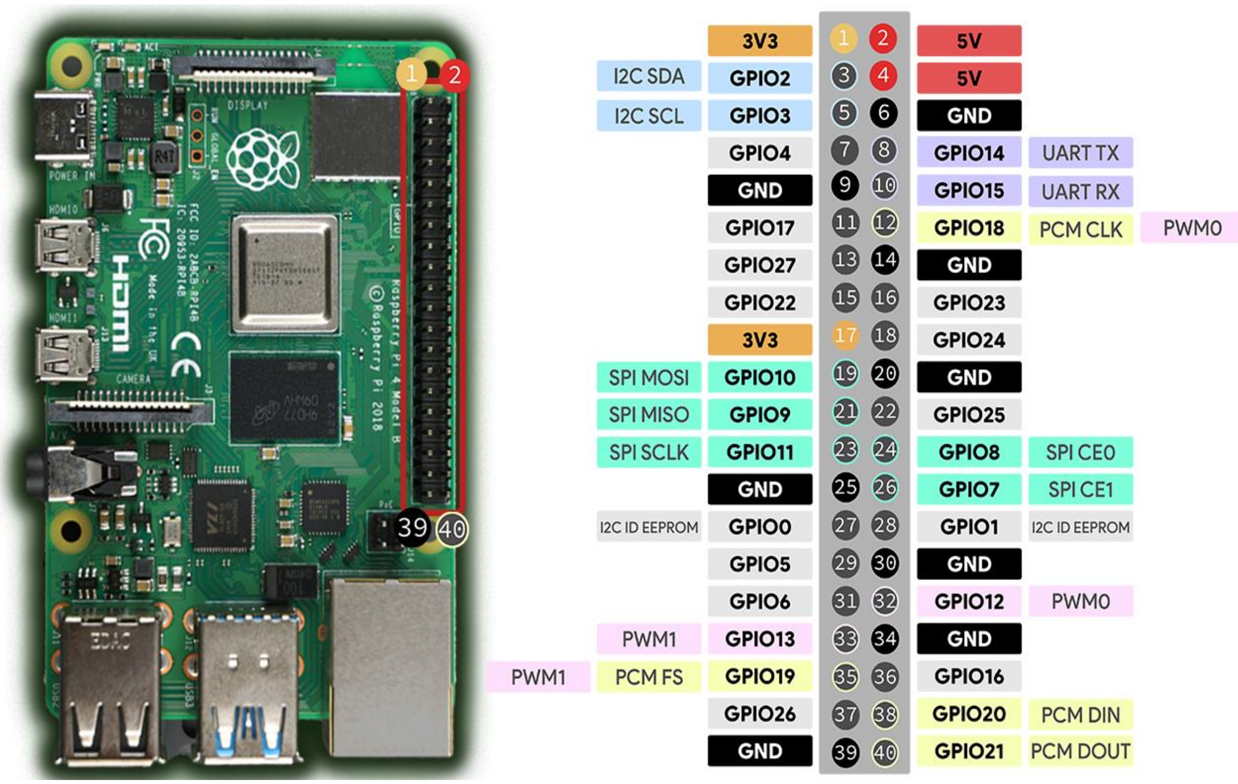


Figure 5.4 : Raspberry pi board GPIO pins

UART (Universal Asynchronous Receiver/Transmitter) is a serial communication protocol in which data is transferred serially i.e. bit by bit. Asynchronous serial communication is widely used for byte oriented transmission. In Asynchronous serial communication, a byte of data is transferred at a time.

UART serial communication protocol uses a defined frame structure for their data bytes. Frame structure in Asynchronous communication consists:

- **START bit:** It is a bit with which indicates that serial communication has started and it is always low.
- **Data bits packet**: Data bits can be packets of 5 to 9 bits. Normally we use an 8-bit data packet, which is always sent after the START bit.
- **STOP bit**: This usually is one or two bits in length. It is sent after a data bits packet to indicate the end of the frame. Stop bit is always logic high.

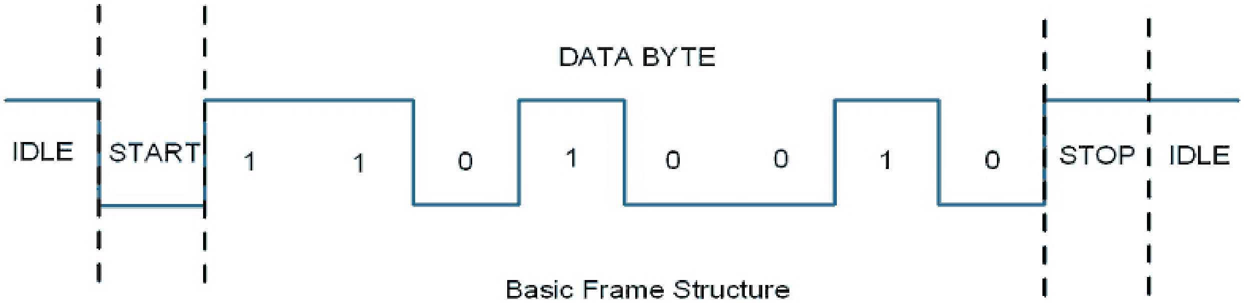


Figure 5.5 : Basic Frame structure in UART

Usually, an asynchronous serial communication frame consists of a START bit (1 bit) followed by a data byte (8 bits) and then a STOP bit (1 bit), which forms a 10-bit frame as shown in the figure above. The frame can also consist of 2 STOP bits instead of a single bit, and there can also be a PARITY bit after the STOP bit.

5.5 RASPBERRY PI UART:

UART: Universal Asynchronous Receiver/Transmitter

Raspberry Pi has two in-built UART which are as follows:

- **PL011 UART**
- **mini UART**

PL011 UART is an ARM based UART. This UART has better throughput than **mini** UART.

In Raspberry Pi 3, mini UART is used for Linux console output whereas PL011 is connected to the On-board Bluetooth module.

And in the other versions of Raspberry Pi, PL011 is used for Linux console output. Mini UART uses the frequency which is linked to the core frequency of the GPU. So as the GPU core frequency changes, the frequency of UART will also change which in turn will change the baud rate for UART. This makes the mini UART unstable which may lead to data loss or corruption.

The PL011 is a stable and high performance UART. For better and effective communication use PL011 UART instead of mini UART.

It is recommended to enable the UART of Raspberry Pi for serial communication. Otherwise, we are not able to communicate serially as UART ports are used for Linux console output and Bluetooth module.

Functionality:

Enables communication between devices using only two wires: one for transmitting data (Tx) and another for receiving data (Rx).

Utilizes asynchronous communication, meaning there's no shared clock signal to synchronize data transmission.

Each byte of data is transmitted in a frame consisting of:

Start bit: Signals the beginning of data transmission.

Data bits: The actual data being sent, typically 8 bits per byte.

Parity bit (optional): Used for basic error detection.

Stop bits: Indicates the end of data transmission.

- I2C (Inter Integrated Circuit) is a synchronous serial protocol that communicates data between two devices.
- It is a master-slave protocol which may have one master or many masters and many slaves whereas SPI has only one master.
- It is generally used for communication over short distances.
- The I2C device has a 7-bit or 10-bit unique address. So, to access these devices, the master should address them by the 7-bit or 10-bit unique address.
- It is also called the Two Wire Interface (TWI) protocol.

5.6 RASPBERRY PI I2C:

The Raspberry Pi is a versatile single-board computer well-suited for various projects, and I2C (Inter-Integrated Circuit) is a valuable communication protocol that expands its capabilities significantly.

I2C is a serial communication protocol that allows multiple devices to communicate with a single master device using only two wires:

Serial Clock (SCL): Synchronizes data transmission between devices.

Serial Data (SDA): Carries the actual data bits being transmitted.

- Raspberry Pi has Broadcom processor having Broadcom Serial Controller (BSC) which is a master, fast-mode (400Kb/s) BSC controller. The BSC bus is compliant with the Philips I2C bus.
- It supports both 7-bit and 10-bit addressing.
- It also has a BSC2 master which is dedicatedly used with the HDMI interface and should not be accessed by the user.
- The I2C bus/interface is used to communicate with the external devices like RTC, MPU6050, Magnetometer, etc with only 2 lines. We can connect more devices using the I2C interface if their addresses are different.

5.7 TEMPERATURE SENSOR & HUMIDITY SENSOR:

TEMPERATURE:

Features Description

- Calibrated Directly in ° Celsius (Centigrade)
- Linear + 10 mV/°C Scale Factor
- 0.5°C Ensured Accuracy (at +25°C) LM35 has an advantage over linear temperature
- Rated for Full −55˚C to +150˚C Range
- Suitable for Remote Applications
- Low Cost Due to Wafer-Level Trimming
- Less than 60-µA Current Drain

- Low Self-Heating, 0.08°C in Still Air
- Nonlinearity Only ±¼°C Typical
- Low Impedance Output, 0.1 Ω for 1 mA Load

The LM35 series are precision integrated-circuit temperature sensors, with an output voltage linearly proportional to the Centigrade temperature. Thus the LM35 has an advantage over linear temperature sensors calibrated in ° Kelvin, as the user is not required to subtract a large constant voltage from the output to obtain convenient Centigrade scaling.

The LM35 does not require any external calibration or trimming to provide typical accuracies of $\pm\frac{1}{4}$°C at room temperature and $\pm\frac{3}{4}$°C over a full −55°C to +150°C temperature range. Low cost is assured by trimming and calibration at the wafer level. The low output impedance, linear output, and precise inherent calibration of the LM35 make interfacing to readout or control circuitry especially easy.

The device is used with single power supplies, or with plus and minus supplies. As the LM35 draws only 60 µA from the supply, it has very low self-heating of less than 0.1°C in still air. The LM35 is rated to operate over a −55°C to +150°C temperature range, while the LM35C is rated for a −40°C to +110°C range (−10° with improved accuracy).

The LM35 series is available packaged in hermetic TO transistor packages, while the LM35C, LM35CA, and LM35D are also available in the plastic TO-92 transistor package. The LM35Dis also available in an 8-lead surface-mount small outline package and a plastic TO-220 package.

A temperature sensor is a device that detects and measures the temperature of an object or its surroundings. They are widely used in various applications, including:

- **Monitoring and controlling** environments in homes, buildings, and industrial settings.
- **Regulating temperatures** in appliances like refrigerators, freezers, ovens, and air conditioners.
- **Measuring body temperature** for medical purposes.
- **Detecting overheating** in electronic devices to prevent malfunctions.

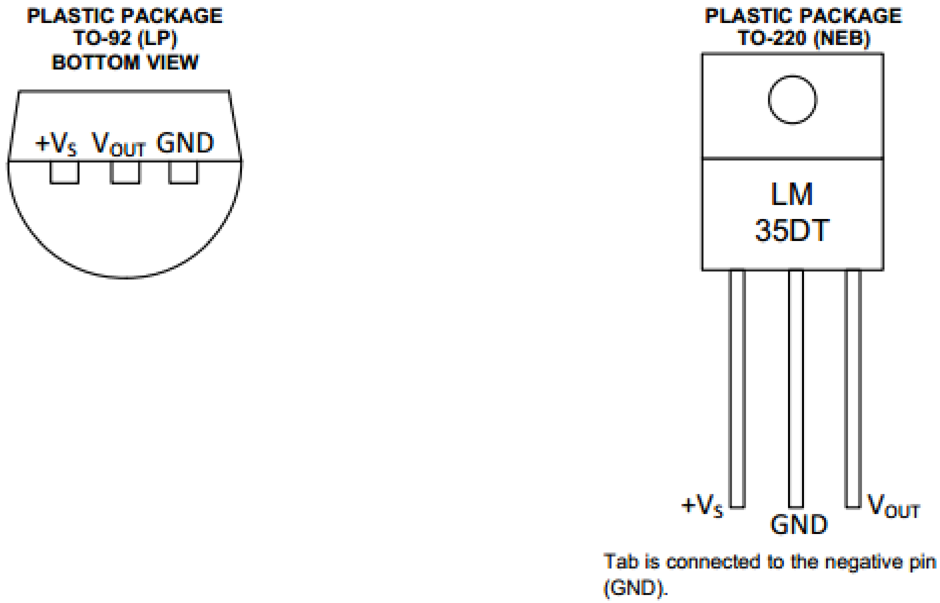


Figure 5.6 : View of temperature sensor

HUMIDITY:

A humidity sensor is a device that measures the amount of water vapor present in the air or other gasses. There are different types of humidity sensors, but most of them work by measuring changes in electrical resistance, capacitance, or temperature caused by moisture in the air. The accuracy of a humidity sensor varies depending on the type of sensor, the calibration, and the conditions under which it is used. Most commercial sensors have an accuracy of around ±2% relative humidity. The range of a humidity sensor depends on the type of sensor and the manufacturer. Some sensors have a range of 0-100% relative humidity, while others have a more limited range. Humidity sensors should be kept clean and free of dust and debris. They should also be calibrated regularly to ensure accuracy.

Sometimes, after rain, the air feels moist. The water seems to have suspended in the air. However, in certain AC's you click some buttons and the atmosphere brightens up. How and why does it all happen?

Moisture forms up in the air, resulting in humidity. However, the humidity sensor in your AC picks it up and cleans it up for you. Isn't that wonderful? Let's take a look at how it does that.

Figure 5.7: Humidity sensor

A humidity sensor (or hygrometer) senses, measures, and reports both moisture and air temperature. The ratio of moisture in the air to the highest amount of moisture at a particular air temperature is called relative humidity. Relative humidity becomes an important factor when looking for comfort. A humidity sensor is an electronic device that measures the humidity in its environment and converts its findings into a corresponding electrical signal. Humidity sensors vary widely in size and functionality; some humidity sensors can be found in handheld devices (such as smartphones), while others are integrated into larger embedded systems (such as air quality monitoring systems). Humidity sensors are commonly used in the meteorology, medical, automobile, HVAC and manufacturing industries.

Humidity sensors can be divided into two groups, as each category uses a different method to calculate humidity: relative humidity (RH) sensors and absolute humidity (AH) sensors. Relative humidity is calculated by comparing the live humidity reading at a given temperature to the maximum amount of humidity for air at the same temperature. RH sensors must therefore measure temperature in order to determine relative humidity. In contrast, absolute humidity is measured without reference to temperature.

The two most common RH sensors are the capacitive and resistive humidity sensors. Capacitive sensors use two electrodes to monitor the capacitance (i.e. the ability to store an electric charge) of a thin metal strip placed between them. The metal's capacitance increases or decreases at a rate that is directly proportional to the change of humidity in the sensor's environment. The difference in charge (voltage) generated by an increase in humidity is then amplified and sent to the embedded computer for processing. Resistive humidity sensors operate on a different principle. These sensors utilize a small polymer comb that increases and decreases in size as the humidity changes, which directly affects the system’s ability to store charge.

Humidity sensors work by detecting changes that alter electrical currents or temperature in the air.

5.8 HEARTBEAT SENSOR:

A person’s heartbeat is the sound of the valves in his/her heart contracting or expanding as they force blood from one region to another. The number of times the heart beats per minute (BPM), is the heart beat rate and the beat of the heart that can be felt in any artery that lies close to the skin is the pulse.

A heartbeat sensor, also known as a pulse sensor, is a device that detects changes in blood volume in a person's body. These changes are caused by the heart pumping blood throughout the body.

There are two main types of heartbeat sensors:

- Photoplethysmography (PPG) sensors**:** These sensors use light to measure blood volume changes. They typically consist of a light-emitting diode (LED) that emits light and a photodiode that detects the light that is reflected or transmitted back. When the heart beats, the blood volume in the tissue being illuminated by the LED changes, which in turn, changes the amount of light that is detected by the photodiode.

- Electrocardiogram (ECG) sensors: These sensors measure the electrical activity of the heart. They typically consist of multiple electrodes that are placed on the chest, arms, and legs. The electrodes detect the tiny electrical signals produced by the heart as it beats.

- Heartbeat sensors are used in a variety of applications, including:

- Fitness trackers and smartwatches: These devices use heartbeat sensors to track a person's heart rate during exercise or other activities.

- Medical devices: Heartbeat sensors are used in a variety of medical devices, such as pulse oximeters, which measure blood oxygen levels, and heart rate monitors, which monitor a person's heart rate continuously.

- Research: Heartbeat sensors are used in research studies to study the heart and cardiovascular system.

Two Ways to Measure a Heartbeat

Manual Way: Heart beat can be checked manually by checking one's pulses at two locations- wrist (the radial pulse) and the neck (carotid pulse). The procedure is to place the two fingers (index and middle finger) on the wrist (or neck below the windpipe) and count the number of pulses for 30 seconds and then multiplying that number by 2 to get the heart beat rate. However pressure should be applied minimum and also fingers should be moved up and down till the pulse is felt.

Using a sensor: Heart Beat can be measured based on optical power variation as light is scattered or absorbed during its path through the blood as the heart beat changes.

Principle of Heartbeat Sensor

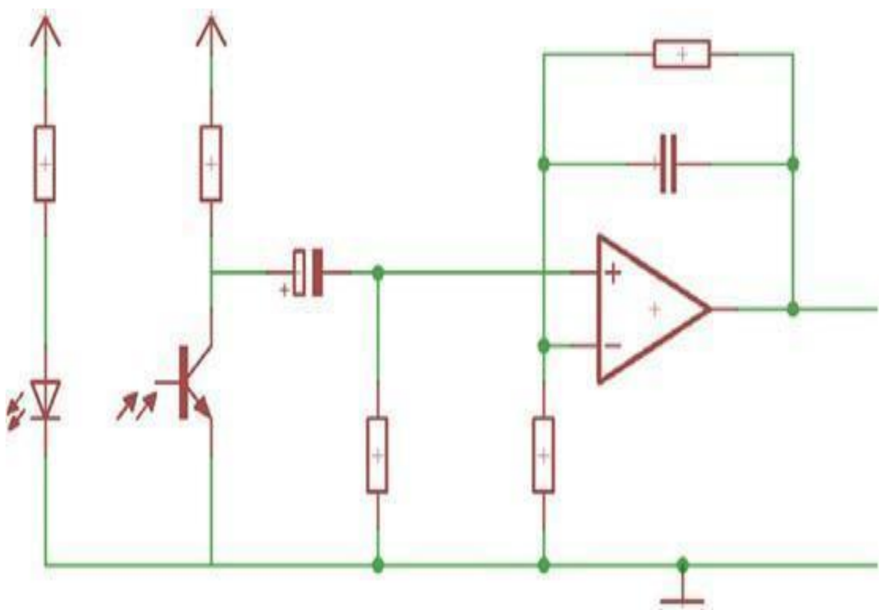
The heartbeat sensor is based on the principle of photoplethysmography. It measures the change in volume of blood through any organ of the body which causes a change in the light intensity through that organ (a vascular region). In case of applications where heart pulse rate is to be monitored, the timing of the pulses is more important. The flow of blood volume is decided by the rate of heart pulses and since light is absorbed by blood, the signal pulses are equivalent to the heart beat pulses.

Working of a Heartbeat Sensor

The basic heartbeat sensor consists of a light emitting diode and a detector like a light detecting resistor or a photodiode. The heartbeat pulses cause a variation in the flow of blood to different regions of the body. When a tissue is illuminated with the light source, i.e. light emitted by the led, it either reflects (a finger tissue) or transmits the light (earlobe). Some of the light is absorbed by the blood and the transmitted or the reflected light is received by the light detector. The amount of light absorbed depends on the blood volume in that tissue. The detector output is in the form of an electrical signal and is proportional to the heart beat rate.

This signal is actually a DC signal relating to the tissues and the blood volume and the AC component synchronous with the heart beat and caused by pulsatile changes in arterial blood volume is superimposed on the DC signal. Thus the major requirement is to isolate that AC component as it is of prime importance. To achieve the task of getting the AC signal, the output The detector is first filtered using a 2 stage HP-LP circuit and is then converted to digital pulses using a comparator circuit or using simple ADC. The digital pulses are given to a microcontroller for calculating the heat beat rate,

Figure 5.8: Circuit diagram for Heart beat sensor



BPM(Beats per minute) = 60*f , where f is the pulse frequency

Practical Heartbeat Sensor:

Practical heartbeat Sensor examples are Heart Rate Sensor (Product No PC-3147). It consists of an infrared led and an ldr embedded onto a clip-like structure. The clip is attached to the organ (earlobe or the finger) with the detector part on the flesh.

Figure 5.9 : Practical Heartbeat sensor

Another example is TCRT1000, having 4 pins-

Pin1: To give supply voltage to the LED

Pin2 and 3 are grounded. Pin 4 is the output. Pin 1 is also the enable pin and pulling it high turns the LED on and the sensor starts working. It is embedded on a wearable device which can be worn on

Figure 5.10 : Heart beat sensor with wire

the wrist and the output can be sent wirelessly (through Bluetooth) to the computer for processing. A heartbeat sensor, also known as a pulse oximeter, works based on a principle called photoplethysmography (PPG). Here's the breakdown of its working:

Components:

- Light source: Typically a Light-Emitting Diode (LED), emitting either red or infrared light.

- Light detector: Either a photodiode or a light-dependent resistor (LDR).

Working principle:

1. Light emission: The LED shines light onto the user's fingertip or earlobe, where blood vessels are close to the skin's surface.
2. Blood volume change: With each heartbeat, the volume of blood in the vessels changes slightly. This is due to the pumping action of the heart pushing blood through the body.
3. Light absorption: More blood in the vessels absorbs more of the emitted light. Conversely, less blood absorbs less light.
4. Detection: The light detector (photodiode or LDR) receives the light that passes through or reflects off the tissue. As the blood volume changes, the amount of light reaching the detector fluctuates.
5. Signal processing: The sensor circuit amplifies and filters these fluctuations in the received light signal.
6. Output: This processed signal, representing the heart rate, can be displayed on a screen, sent to a microcontroller for further analysis, or used to control other functions.
7. The accuracy of heartbeat sensors can be affected by factors like movement, poor placement, and skin tone.
8. These sensors are mainly used for fitness tracking and general health monitoring and should not be used for medical diagnosis.

For a deeper understanding, you can explore terms like "photoplethysmography" and "light

Figure 5.11 : Detailed view of Heart beat sensor

absorption in tissues".

Application Developing your own Heartbeat Sensor System:

A basic Heartbeat Sensor system can also be built using basic components like a ldr, comparator IC LM358 and a Microcontroller as given below

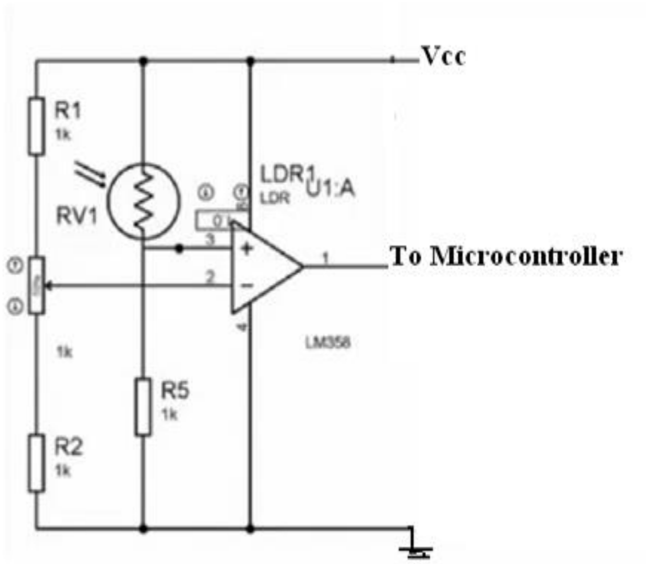


Figure 5.12 : Circuit diagram for LM358 and microcontroller

As described above regarding the principle of heart beat sensor, when the finger tissue or the earlobe tissue is illuminated using a light source, the light is transmitted after getting modulated i.e. a part getting absorbed by the blood and the rest being transmitted. This modulated light is received by the light detector.

Here a Light Dependent Resistor (LDR) is used as a light detector. It works on the principle that when light falls on the resistor, its resistance changes. As the light intensity increases, the resistance decreases. Thus the voltage drop across the resistor decreases.

Circuit Assembly: Connect the heartbeat sensor to the Raspberry Pi according to the sensor's datasheet and wiring diagrams.

Python Script: Write a Python script that:
Imports necessary libraries (RPi.GPIO, adafruit_max30102 or equivalent).
Initializes the GPIO pins and the heartbeat sensor.
Continuously reads data from the sensor using appropriate library functions.

Processes the data, potentially filtering noise and calculating the heart rate.

5.9 Micro-electromechanical systems (MEMS):

Micro-electromechanical systems (MEMS) is a technology that combines computers with tiny mechanical devices such as sensors, valves, gears, mirrors, and actuators embedded in semiconductor chips.

Figure 5.13 : MEMS sensor

MEMS or what he calls analogy computing will be "the foundational technology of the next decade

MEMS is also sometimes called smart matter.

Among the presently available uses of MEMS or those under study are:

- Global position system sensors that can be included with courier parcels for constant tracking and that can also sense parcel treatment en route.
- Sensors built into the fabric of an airplane wing so that it can sense and react to air flow by changing the wing surface resistance; effectively creating a myriad of tiny wing flaps.
- Optical switching devices that can switch light signals over different paths at 20-nanosecond switching speeds.
- Sensor-driven heating and cooling systems that dramatically improve energy savings.
- Building supports embedded sensors that can alter the flexibility properties of a material based on atmospheric stress sensing.

Applications:

- Vibrations measurement and Tilt inclination (to adjust rotation speed, circuit breaker security, fault prediction),
- Professional and domestic Alarms
- Maintenance management and Asset Tracking for shocks monitoring
- Alarm and Anti Tamper function

CHAPTER 6

IMPLEMENTATION:

The Pi can run a large range of systems, including the official Raspbian OS, Ubuntu Mate, Snappy Ubuntu Core, the Kodi-based media centers OSMC and LibreElec, the non-Linux based Risc OS (one for fans of 1990s Acorn computers). It can also run Windows 10 IoT Core, which is very different to the desktop version of Windows, as outlined below.

The implementation of a Raspberry Pi involves several steps, from setting up the hardware to installing the software and utilizing it for your desired application. Here's a breakdown of the general process:

HARDWARE SETUP:

Gather the essentials: You'll need your Raspberry Pi board, a power supply (appropriate for your Pi model), a microSD card (with enough storage for your chosen operating system), an HDMI cable, a monitor, a keyboard, and a mouse. Depending on your project, additional peripherals like sensors, cameras, or other components might be necessary.

Power Up: Connect the power supply to the Raspberry Pi and plug it into an outlet.

Connect peripherals: Connect the HDMI cable to the Pi and your monitor, and then plug in the keyboard and mouse.

Prepare the microSD card: You can use the Raspberry Pi Imager tool to download and flash the chosen operating system (like Raspberry Pi OS) onto the microSD card.

SOFTWARE SETUP:

Boot up and initial configuration: Insert the prepared microSD card into the Raspberry Pi and power it on. The Pi will boot up automatically and might guide you through the initial setup process, including language selection, Wi-Fi configuration, and password creation. Use raspi-config tool for further configuration option.

Software installation: Once booted, you can access the desktop environment and use the built-in package manager to install additional software needed for your project. This might involve programming languages, libraries, specific applications, or tools depending on your goals.

IMPLEMENTATION AND USAGE:

Coding and development: If your project involves coding, you can use the Raspberry Pi as a development platform to write your code using various available text editors and compilers.

Interfacing with hardware: The Raspberry Pi provides GPIO pins (General Purpose Input/Output) that allow you to connect various sensors, actuators, and other hardware components. You'll need to learn about programming these pins and using libraries to interact with the connected hardware.

Running your project: Once you have completed the hardware setup, software installation, and any necessary coding, you can run your project by executing your code or launching the specific application you've installed.
However, these are just the officially recommended operating systems,

- Hands-On: First impressions of the Raspberry Pi 3
- Hands-on with Raspbian GNU/Linux Stretch for Raspberry Pi

Upton says he feels the Pi 4 also likely has the power to run a full desktop version of Windows on Arm, but that any decision to port Windows to the Pi 4 rests with Microsoft.

- How to set up Windows 11 on the Raspberry Pi
- Windows 11 on the Raspberry Pi: What you need to know

The Pi 4 can run Windows desktop apps, although it requires an awful lot of effort to do so, and even then, apps will only run poorly. It used to be possible to do so using the ExaGear Desktop software, although this is no longer on sale. There are free alternatives, however, such as Pi386.

Whichever approach you use, performance will be sub-par, with the tools needed to run Windows apps on the Pi requiring so much processing power that you're basically restricted to running 20-year-old Windows apps and games, and simple modern text editors. Basically, while it's technically possible, it's not something you'll probably want to do.

- How to run Windows apps on the Raspberry Pi
- Yes, Windows apps will run on the Raspberry Pi: But why would you bother?

It can run Ubuntu with various desktops, with the Raspberry Pi Foundation highlighting Ubuntu Mate and Ubuntu Snappy Core as standouts.

- **System-on-a-chip:** Broadcom BCM2711
- **CPU:** Quad-core 1.5GHz Arm Cortex-A72 based processor
- **GPU:** VideoCore VI
- **Memory:** 1/2/4GB LPDDR4 RAM
- **Connectivity:** 802.11ac Wi-Fi / Bluetooth 5.0, Gigabit Ethernet
- **Video and sound:** 2 x micro-HDMI ports supporting 4K@60Hz displays via HDMI 2.0, MIPI DSI display port, MIPI CSI camera port, 4 pole stereo output and composite video port
- **Ports:** 2 x USB 3.0, 2 x USB 2.0
- **Power:** 5V/3A via USB-C, 5V via GPIO header
- **Expandability:** 40-pin GPIO header

It's worth investing in a case to protect the Pi from damage, especially if you're going to be carrying the Pi with you. Note the Pi 4 doesn't fit earlier Pi cases due a change in its layout.

It's also sensible to shell out for a high-speed micro SD card, as outlined below, if performance is important to you.

While the Pi can run many operating systems, if you're after stability and performance then the official Raspbian operating system is a good choice, having been tuned to get the most from the Pi, and thanks to bundling a fast web browser and a decent selection of office and programming software.

One tip if you didn't install the Raspbian OS using the NOOBS installer, and you're running out of space: you can go into the terminal and type 'sudo raspi-config' and then select the option to 'Expand root partition to fill SD card', which will ensure you're using the available space on the card.
If you're running the Pi's official Raspbian operating system then keeping the Pi up to date is relatively straightforward. Just open the terminal and type "*sudo apt-get update*".

Once the update is complete, then type "***sudo apt-get dist-upgrade***".

And then type "***sudo apt install apache2 -y***" for installing apache in the system.

According to tests, the peak power consumption of the Pi 4 is about 7.6W under load and 3.4W when idle. There's no shortage of Raspberry Pi kits available, which add everything from speech recognition, to robotic arms to build-it-yourself laptops for kids to virtual assistants to the $35 board. Due to the success of the Pi, if you've got an idea for a project, there's probably a kit out there to suit your needs. The best choice is the official Raspberry Pi Foundation USB Type-C power supply, which is rated at 5.1V/3A.If you're installing the official Raspbian OS you'll need at least an 8GB micro SD card, whereas for the Raspbian Lite you'll need a minimum of 4GB.Yes, the board supports 802.11ac Wireless LAN (throughput of around 100 Mbps) and Bluetooth 5.0.

we type in terminal these below commands which will go to place of that files and execute them,

- cd /var/www/html
- /var/www/html $ ls -al
- /var/www/html $ hostname -I
- sudo apt install php -y
- sudo rm index.html
- sudo service apache2 restart
- ls -lh /var/www/
- sudo chown -R pi:www-data /var/www/html/
- sudo chmod -R 770 /var/www/html/
- ls -lh /var/www

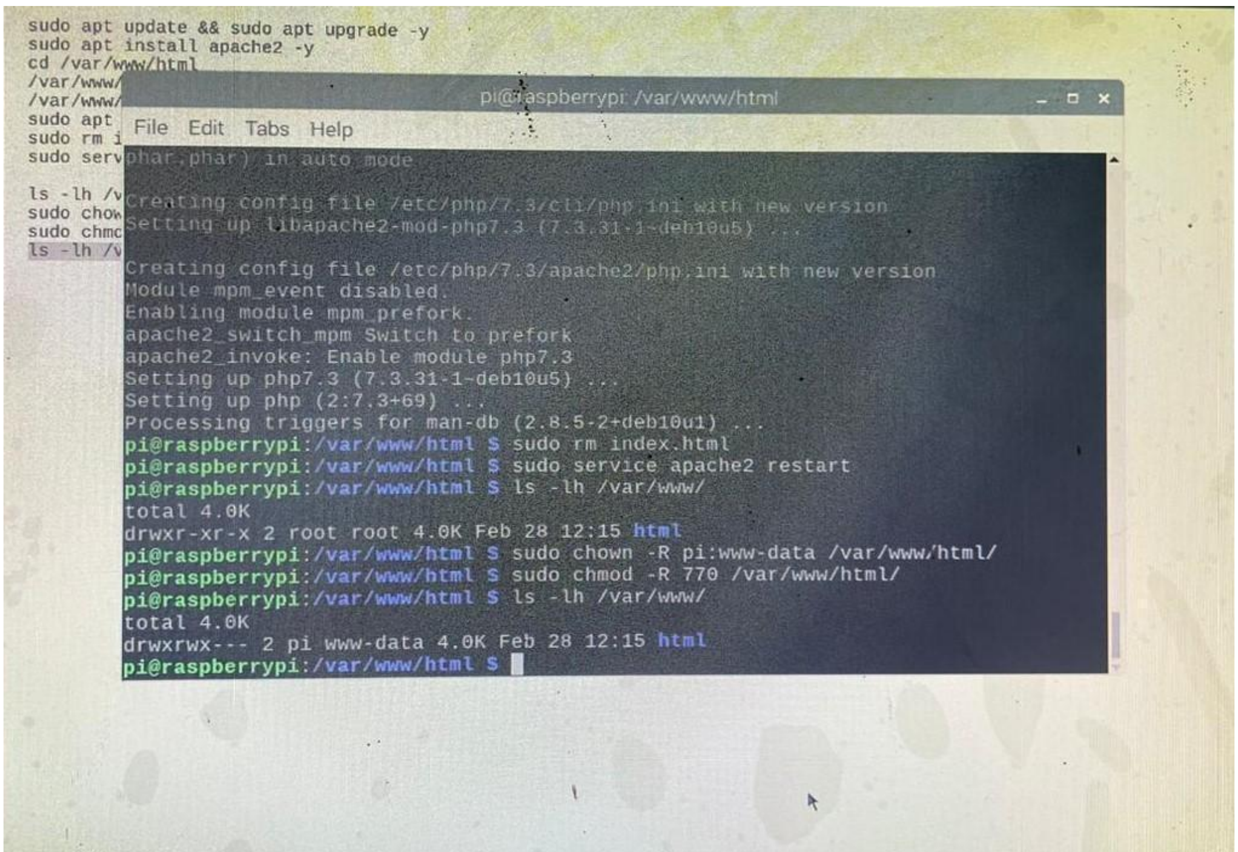


Figure 6.1 : Commands for installing apache and running program

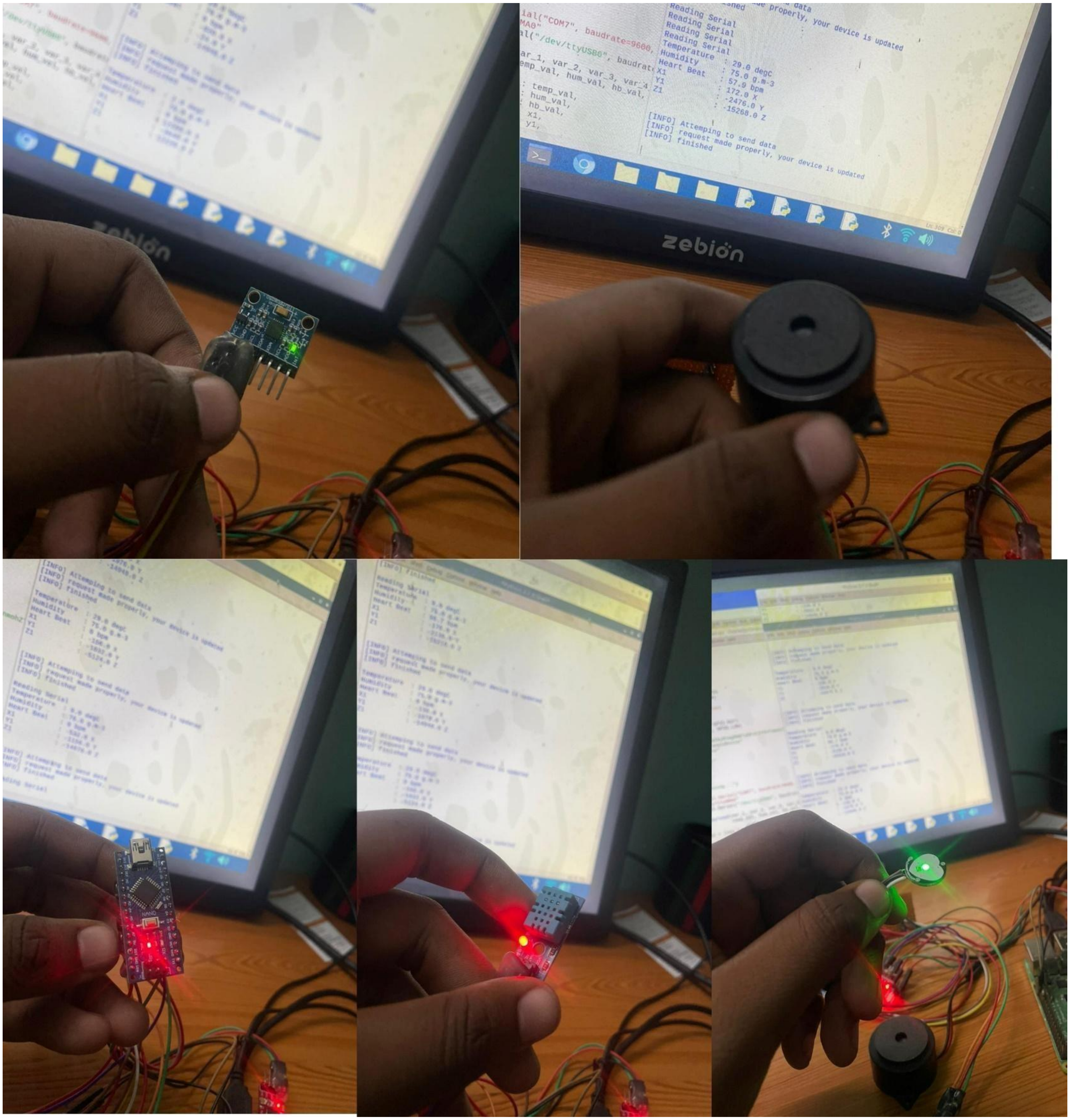


Figure 6.2 : Sensors used in our project

CHAPTER 7

TESTING:

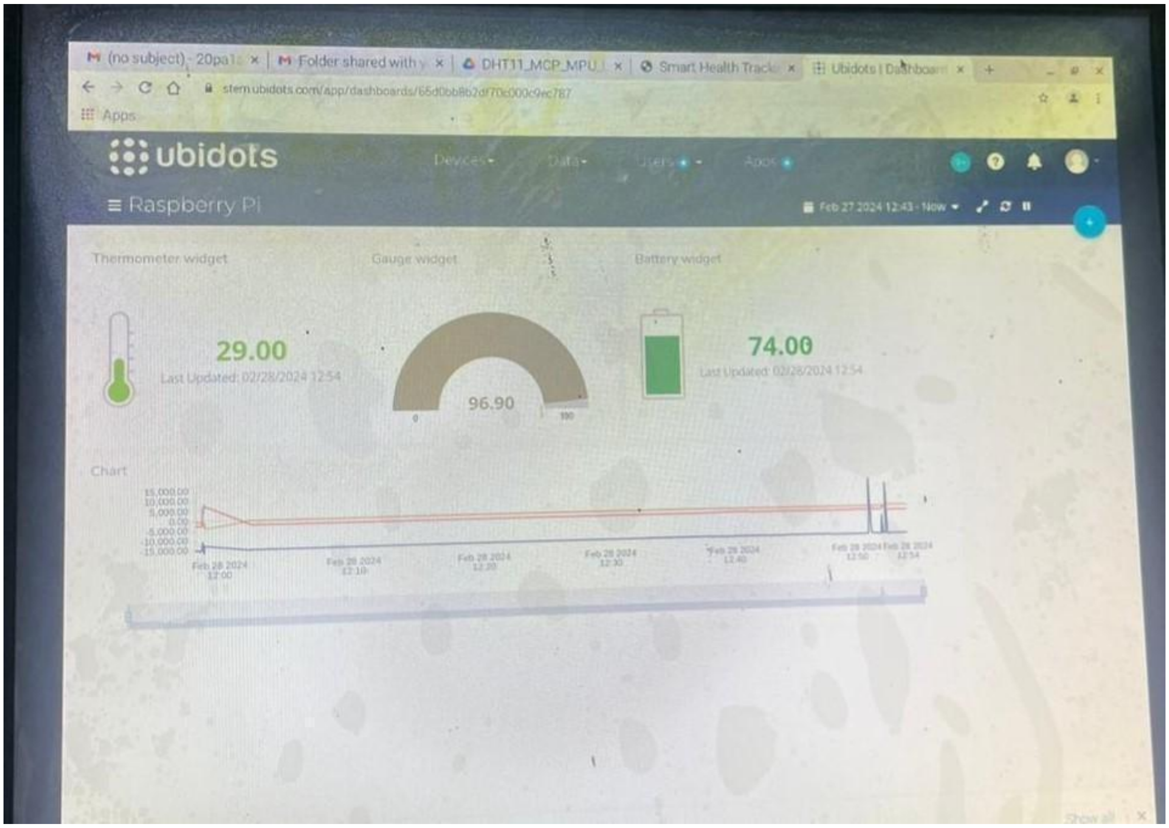


Figure 7.1 : Ubidots dashboard of the project

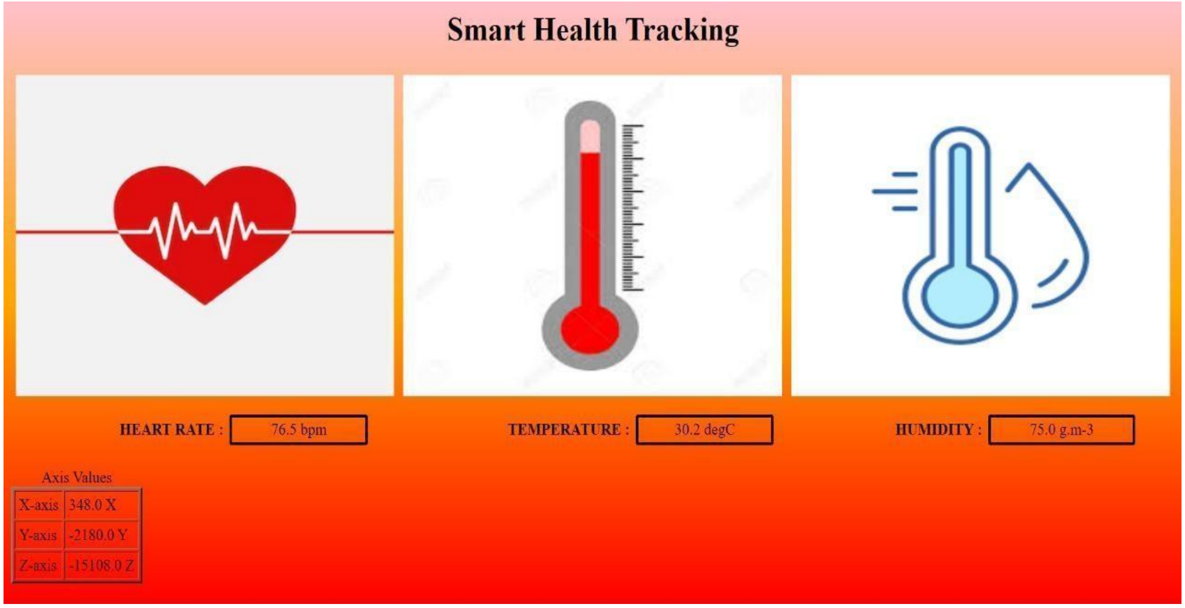


Figure 7.2 : Local website designed using PHP

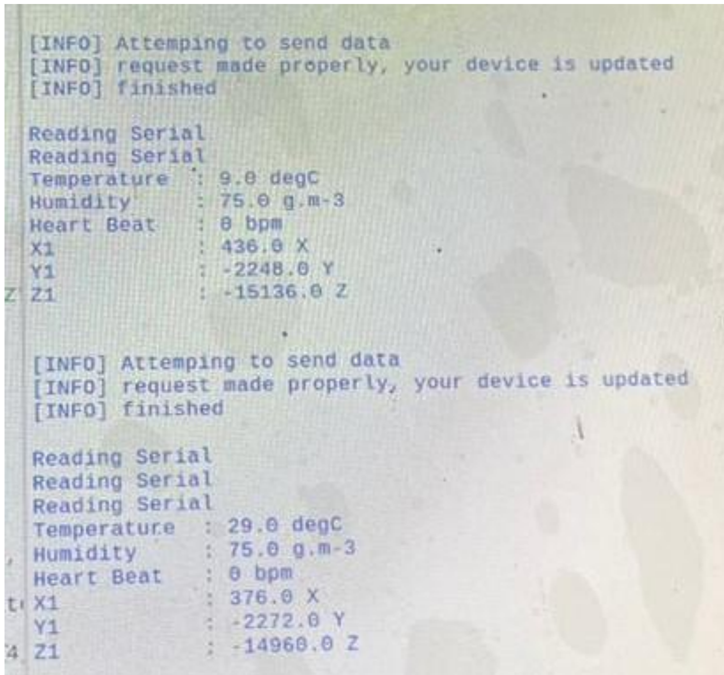


Figure 7.3 : Output

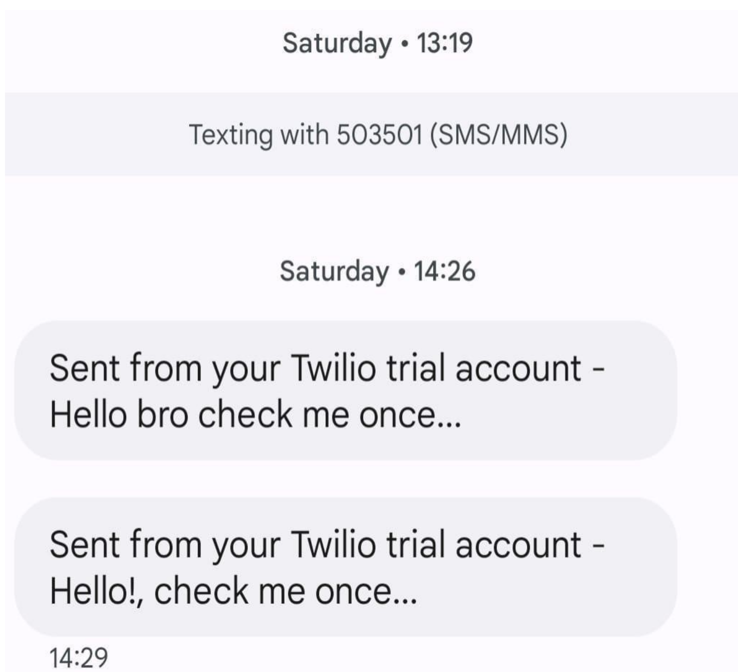


Figure 7.4 : Immediate messaging initiated for any critical event

CONCLUSION

To monitor the health of a patient, we need to take readings from different sensors such as heart rate, temperature and humidity, and MEMS. The heart rate sensor will calculate the bpm of the body, while temperature and humidity sensors will inform us about the current temperature and humidity of the surroundings. The MEMS sensor will measure physical quantities such as acceleration. We collected all these sensors and an Arduino Uno is connected to a Raspberry Pi board, and a memory card is kept on that board for the Raspberry Pi OS. You can access the output by connecting a monitor, keyboard, and mouse to the board. After running the code, the data is stored and analyzed by the Raspberry Pi, and you can view the output on the screen directly, there are some other ways also. The first way is to use an established website, Ubidots, which provides a dashboard for analyzing the data. This will be useful for family members to keep an eye on the patient all the time, and we can access this from anywhere. The second way is to create a local website using PHP code to display the output. And we used the Twilio website which provided us communication APIs for SMS to mobile numbers. You can stay informed if there is a sudden increase or decrease in heartbeats as the system will send an alert message right away to your designated phone numbers, so they can take action quickly and efficiently.

BIBLIOGRAPHY

[1] A. Gupta, R. Kumar, S. Sharma, "IoT-based Wearable Health Monitoring Device with Cloud Connectivity", IEEE International Conference on Internet of Things and Cloud Computing (IoTCC), pp. 105-110, 2023.

[2] K. Patel, H. Shah, P. Mehta, "Smart IoT-enabled Home Healthcare System for Remote Patient Monitoring", IEEE International Conference on Smart Technologies for Healthcare (ICSTH), pp. 150-155, 2023.

[3] S. Kumar, A. Mishra, R. Singh, "Real-time Patient Monitoring System using IoT and AI", International Conference on Internet of Medical Things and Healthcare (IoMT), pp. 80-85, 2023.

[4] R. Verma, S. Mishra, A. Singh, "Smart IoT-based Healthcare Monitoring System for Disease Prediction", IEEE International Conference on Intelligent Systems and Data Engineering (ISDE), pp. 220-225, 2022.

[5] N. Sharma, P. Jain, R. Gupta, "IoT-enabled Smart Health Monitoring Bracelet for Real-time Fitness Tracking", International Conference on Communication, Computing, and IoT (CCIoT), pp. 95-100, 2022.

[6] M. Ali, S. Rahman, K. Ahmed, "Wireless Sensor Network-based Health Monitoring System for Elderly People using IoT", IEEE International Conference on Innovations in Electrical and Electronics Engineering (ICIEEE), pp. 45-50, 2022.

[7] J. Smith, A. Patel, B. Johnson, "IoT-enabled Health Monitoring System for Remote Patient Care", International Conference on Internet of Things (IoT), pp. 120-125, 2021.

[8] S. Gupta, R. Sharma, K. Singh, "Smart Wearable Device for Continuous Blood Pressure Monitoring Using IoT", IEEE International Conference on Wearable and Implantable Body Sensor Networks (BSN), pp. 78-83, 2021.

[9] A. Kumar, B. Das, C. Mishra, "An IoT-based Real-time Health Monitoring System for Elderly Care", IEEE International Conference on Internet of Things and Intelligence Systems (IoTaIS), pp. 210-215, 2021.

[10] N. Jain, S. Gupta, A. Sharma, "Low-cost IoT-based Health Monitoring System for Rural Areas", International Conference on Sustainable Technologies for Rural Development (STRD), pp. 130-135, 2023.

[11] R. Gupta, S. Kumar, K. Patel, "Smart IoT-based Health Monitoring Bracelet with Emergency Alert System", IEEE International Conference on Intelligent Systems and Applications (ISA), pp. 75-80, 2023.

[12] K. Singh, A. Kumar, S. Chauhan, "Wireless Sensor Network for Real-time Health Monitoring in IoT Environment", IEEE International Conference on Communication and Network Security (ICoNS), pp. 80-85, 2022.

[13] H. Patel, S. Gupta, R. Sharma, "IoT-based Smart Health Monitoring System for Remote Diagnosis", IEEE International Conference on Advances in Internet of Things (ICIOT), pp. 65-70, 2021.

[14] R. Das, S. Roy, M. Ghosh, "A Survey of IoT-based Health Monitoring Systems: Challenges and Opportunities", IEEE International Conference on Internet of Medical Things (IoMT), pp. 40-45, 2021.

APPENDIX

Code that we should type in terminal

sudo apt update && sudo apt upgrade -y
sudo apt install apache2 -y
cd /var/www/html
/var/www/html $ ls -al
/var/www/html $ hostname -I
sudo apt install php -y
sudo rm index.html
sudo service apache2 restart
ls -lh /var/www/
sudo chown -R pi:www-data /var/www/html/
sudo chmod -R 770 /var/www/html/
ls -lh /var/www

➢ Python code for Raspberry pi

```
import time
import numpy as np
import os
import serial
import requests
import math
import random
import RPi.GPIO as GPIO

GPIO.setmode(GPIO.BCM)
GPIO.setwarnings(False)
buzzer = 26      #26 is pin
GPIO.setup(buzzer, GPIO.OUT)
GPIO.output(buzzer, GPIO.LOW)

print ("Processing...")

port = serial.Serial("/dev/ttyUSB0", baudrate=9600, timeout=1.0)


def Get_Serial_Data():
    Byte_Range = 1024
    for Idx in range(Byte_Range):
        Serail_Data = port.readline()
        #print(Serail_Data)
        Serail_Data_Decode = Serail_Data.decode('utf-8')
        if len(Serail_Data_Decode) > 5:
            Info_Set = True
            break
    else:
        Byte_Range = Byte_Range + 1
    while Info_Set:
```

```
        sensor_value = Serail_Data_Decode.split('*')

        temp_val  = sensor_value[0]
        hum_val   = sensor_value[1]
        hb_val    = sensor_value[2]
        x1        = sensor_value[3]
        y1        = sensor_value[4]
        z1        = sensor_value[5].rstrip()

        temp_val  = float(temp_val)
        hum_val   = float(hum_val)
        hb_val    = (float(hb_val) * 100) / 1000
        if hb_val < 48:
            hb_val = 0
        x1        = float(x1)
        y1        = float(y1)
        z1        = float(z1)
        Info_Set = False
        return temp_val, hum_val, hb_val, x1, y1, z1

if __name__ == '__main__':
    while True:
        try:
            temp_val, hum_val, hb_val, x1, y1, z1 = Get_Serial_Data()

            sensor_txt_path = r'/var/www/html/sensor_data.txt'
            sensor_f = open(sensor_txt_path, "w")
            sensor_f.write(str(temp_val) + '*' + str(hum_val) + '*' +
str(hb_val) + '*' + str(x1) + '*' + str(y1) + '*' + str(z1))
            sensor_f.close()

            print('Temperature  : ' + str(temp_val)  + ' degC')
            print('Humidity     : ' + str(hum_val) + ' g.m-3')
            print('Heart Beat   : ' + str(hb_val)   + ' bpm')
            print('X1           : ' + str(x1)   + ' X')
            print('Y1           : ' + str(y1)   + ' Y')
            print('Z1           : ' + str(z1)   + ' Z')
            print('\n')
            time.sleep(3)
            port.flushInput()
            if(hb_val>110 or hb_val<50):
                account_sid = 'AC2189c832f54015bd052c0123763fb61f'
                auth_token = 'fa0f4ec7d5de3b41eff07bd83c92e360'
                client = Client(account_sid, auth_token)

                message = client.messages.create(
                from_='+18582958642',
                body='Hello!, check me once...',
                to='+917382225656'
                )

                GPIO.output(buzzer, GPIO.HIGH)
            else:
                GPIO.output(buzzer, GPIO.LOW)
```

```
        except:
            print('Reading Serial')
```

Code to post the data to ubidots website through url

```
import time
import numpy as np
import os
import serial
import time
import requests
import math
import random
import RPi.GPIO as GPIO
GPIO.setmode(GPIO.BCM)
GPIO.setwarnings(False)
buzzer = 26
GPIO.setup(buzzer, GPIO.OUT)
GPIO.output(buzzer, GPIO.LOW)
TOKEN = "BBUS-ENjmIby81eg8bBTuDPzEIYhhfnmohZ" # Token
DEVICE_LABEL = "raspidevice"
VL1 = "temperature"
VL2 = "humidity"
VL3 = "heartbeat"
VL4 = "x1"
VL5 = "y1"
VL6 = "z1"
print ("Processing...")
#port = serial.Serial("COM7", baudrate=9600, timeout=1.0)
#port = "/dev/ttyAMA0"
port = serial.Serial("/dev/ttyUSB0", baudrate=9600, timeout=1.0)
def build_payload(var_1, var_2, var_3, var_4, var_5, var_6,
                  temp_val, hum_val, hb_val, x1, y1, z1):
    payload = {var_1: temp_val,
               var_2: hum_val,
               var_3: hb_val,
               var_4: x1,
               var_5: y1,
               var_6: z1}
    return payload
def post_request(payload):
    # Creates the headers for the HTTP requests
    url = "http://industrial.api.ubidots.com"
    url = "{}/api/v1.6/devices/{}".format(url, DEVICE_LABEL)
    headers = {"X-Auth-Token": TOKEN, "Content-Type": "application/json"}
    # Makes the HTTP requests
    status = 400
    attempts = 0
    while status >= 400 and attempts <= 5:
        req = requests.post(url=url, headers=headers, json=payload)
        status = req.status_code
```

```
          attempts += 1
          time.sleep(1)

def Get_Serial_Data():
        Byte_Range = 1024
        for Idx in range(Byte_Range):
            Serail_Data = port.readline()
            #print(Serail_Data)
            Serail_Data_Decode = Serail_Data.decode('utf-8')
            if len(Serail_Data_Decode) > 5:
                Info_Set = True
                break
            else:
                Byte_Range = Byte_Range + 1

        while Info_Set:
            sensor_value = Serail_Data_Decode.split('*')
            temp_val = sensor_value[0]
            hum_val  =  sensor_value[1]
            hb_val = sensor_value[2] x1
            = sensor_value[3]
            y1    = sensor_value[4]
            z1        = sensor_value[5].rstrip()
            temp_val  = float(temp_val)
            hum_val   = float(hum_val)
            hb_val    = (float(hb_val) * 100) / 1000
            if hb_val < 48:
                hb_val = 0
            x1        = float(x1)
            y1        = float(y1)
            z1        = float(z1)
            Info_Set = False
            return temp_val, hum_val, hb_val, x1, y1, z1

    if _name____ == '_main_':
        while True:
          try:
                temp_val, hum_val, hb_val, x1, y1, z1 = Get_Serial_Data()
                print('Temperature  : ' + str(temp_val)  + ' degC')
                print('Humidity     : ' + str(hum_val) + ' g.m-3')
                print('Heart Beat   : ' + str(hb_val)   + ' bpm')
                print('X1           : ' + str(x1)   + ' X')
                print('Y1           : ' + str(y1)   + ' Y')
                print('Z1           : ' + str(z1)   + ' Z')
                print('\n')
                time.sleep(3)
                port.flushInput()
```

```
        payload = build_payload(
        VL1, VL2, VL3, VL4, VL5, VL6,
        temp_val, hum_val, hb_val, x1, y1, z1)


    print("[INFO] Attempting to send data")
    post_request(payload)
    print("[INFO] finished\n")
    if temp_val > 38 or hb_val > 110:
        GPIO.output(buzzer, GPIO.HIGH)
    else:
        GPIO.output(buzzer, GPIO.LOW)
except:
    print('Reading Serial')
```