

IOT PARALYSIS PATIENT HEALTH CARE SYSTEM

Submitted in partial fulfilment of the requirements for the award of
Bachelor of Engineering Degree in
Electronics and Communication Engineering

By

U.M.S.R.CHOWDARY (38130228)

V.N.V.R.HARSHAVARDHAN (38130230)



**DEPARTMENT OF ELECTRONICS & COMMUNICATION
ENGINEERING
SCHOOL OF ELECTRICAL AND ELECTRONICS ENGINEERING**

SATHYABAMA

**INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)**

Accredited with Grade “A” by NAAC

JEPPIAAR NAGAR, RAJIV GANDHI SALAICHENNAI – 600119

MAY - 2022



SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY

(DEEMED TO BE UNIVERSITY)

Accredited "A" Grade by NAAC | 12B Status by UGC | Approved by AICTE

www.sathyabama.ac.in

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

BONAFIDE CERTIFICATE

This is to certify that Project Report is the bonafide work of **U.M.S.R. CHOWDARY (38130228)** and **V.N.V.R. HARSHAVARDHAN (38130230)** who carried out the Project entitled "**IOT PARALYSIS PATIENT HEALTH CARE SYSTEM**" under the guidance of **Dr. N.M. NANDHITHA, M.E., Ph.D.**, Dean, during the period Jan 2022 to March 2022.

Internal Guide

Dr. N.M. NANDHITHA, M.E., Ph.D.

Head of the Department

Dr. T. RAVI, M.E., Ph.D.

Submitted for Viva Voce held on

Internal Examiner

External Examiner

DECLARATION

We, **U.M.S.R. CHOWDARY (38130228)** and **V.N.V.R. HARSHAVARDHAN (38130230)** hereby declare that the Project report entitled **“IOT PARALYSIS PATIENT HEALTH CARE PROJECT”** is done under the guidance of done by us under the guidance of **Dr. N.M. NANDHITHA, M.E., Ph.D.**, Dean, is submitted in partial fulfilment of the requirements for the award of Bachelor of Engineering degree in **ELECTRONICS AND COMMUNICATION ENGINEERING**.

1)

DATE:

2)

PLACE: CHENNAI

SIGNATURE OF THE CANDIDATE

ACKNOWLEDGEMENT

I pleased to acknowledge my sincere thanks to board management of **SATHYABAMA INSTITUTE OF SCIENCE AND TECHNOLOGY** for their kind encouragement in doing this internship and for completing it successfully. I am grateful to them.

I convey my thanks to **Dr. N.M. NANDHITHA, M.E., Ph.D.**, Dean, school of Electrical and Electronics Engineering and **Dr. T. RAVI, M.E., Ph.D.**, Head of the Department, Electronics and Communication engineering for providing me necessary support and details at the right time during the progressive reviews.

I would like to express my sincere and deep sense of gratitude to my Project Guide **Dr. N.M. NANDHITHA, M.E., Ph.D.**, Dean, for her valuable guidance, suggestions and constant encouragement paved way for the successful completion of my Project.

I wish to express my thanks to all Teaching and Non-teaching staff members of the Department of Electronics and Communication Engineering who were helpful in many ways for the completion of the project.

ABSTRACT

We all know that the paralysis condition is a loss of muscle function in the body parts. It can affect any part of your body at any time, then probably you may won't feel the pain the affected area.

Technical and Therapeutic innovations are there to improve the quality of life Our goal is to develop a device which should be easy to use and should be affordable which consists of basic health care monitoring system with nursing care.

We know that these people can't able to convey their messages or needs. To overcome this, we come up with the system that helps these patients to display messages by very simple motion. This device can be designed to be mounted on the finger or to be inbuilt in their clothes.

TABLE OF CONTENTS

CHAPTER NO	TITLE	PAGE NO
	ABSTRACT	v
	LIST OF FIGURES	ix
	LIST OF ABBREVIATIONS	xi
1	INTRODUCTION	1
2	LITERATURE SURVEY	2
3	PROBLEM DEFINATION OBJECTIVES	4
	3.1 PROPOSED SYSTEM	4
	3.2 BLOCK DIAGRAM	5
	3.3 HARDWARE REQUIRED	6
4	EMBEDDED SYSTEMS	7
	4.1 OVERVIEW OF EMBEDDED SYSTEMS	7
	4.2 BLOCK DIAGRAM OF AN EMBEDDED SYSTEM	9
	4.2.1 Characteristics of Embedded Systems	9
	4.3 POWER SUPPLY	13
	4.4 TRANSFORMER	14
	4.5 RECTIFIER	14
	4.6 FILTERS	14
	4.7 REGISTERS	15
	4.8 REGULATORS	15
	4.9 FEATURES AND DESCRIPTION OF REGULATORS	16
5	INTRODUCTION ABOUT IOT	17
	5.1 WHAT IS AN INTERNET OF THINGS (IoT)	17

CHAPTER NO	TITLE	PAGE NO
5.2	HOW DOES INTERNET OF THING (IOT) WORK?	19
5.3	FEATURES OF IOT	19
5.4	ADVANTAGES AND DISADVANTAGES OF IoT	21
5.4.1	Advantages of IoT	21
5.4.2	Disadvantages of IoT	21
5.5	EMBEDDED DEVICES (SYSTEM) IN IOT	22
5.5.1	Embedded System Hardware	22
5.5.2	Embedded System Software	23
5.6	IoT ECOSYSTEM	23
5.7	IoT DECISION FRAMEWORK	25
5.7.1	Decision Area	26
5.8	IoT ARCHITECTURE	27
5.9	IoT DEVICES	31
5.10	ARDUINO DEVICE	33
5.11	PROPERTIES OF IoT DEVICES	35
5.12	IoT – PLATFORM	39
6	NODE MCU AND BRIEF ABOUT SENSORS	43
6.1	ESP8266 ARDUINO CORE	44
6.2	NODEMCU DEVELOPMENT BOARD/KIT V0.9 (VERSION1)	44
6.3	BASIC FLEX SENSOR CIRCUIT	50
6.4	CONTROL A SERVO USING ARDUINO AND FLEX SENSOR	55
6.4.1	Circuit Design	55
6.5	HEARTBEAT SENSOR	58
6.5.1	Two Ways to Measure a Heartbeat	58

CHAPTER NO	TITLE	PAGE NO
	6.5.2 Principle of Heartbeat Sensor	58
6.6	TEMPERATURE SENSOR	62
6.7	THERMOCOUPLE	63
6.8	SENSOR ICS	64
7	CODE AND OUTPUT RESULTS	68
7.1	SETUP	68
7.2	CODE	71
7.3	OUTPUT RESULTS	72
8.	CONCLUSION	77
	REFERENCES	78

LIST OF FIGURES

FIGURE NO	TITLE	PAGE NO
3.1	Block diagram	5
4.1	Block diagram of a typical embedded system	9
4.2	Embedded Systems having Heterogeneous Architectures	11
4.3	General Block of Power Supply Unit	14
5.1	Internet of Things	18
5.2	Example of IoT	18
5.3	IoT Architecture	19
5.4	Embedded Devices System in (IoT)	22
5.5	Basic Embedded System	23
5.6	IoT Ecosystem	24
5.7	The IoT Technology stack	25
5.8	Decision Area of the IoT Decision Framework	26
5.9	Stages of IoT Solutions Architecture	28
5.10	The 4 stage IoT Solutions Architecture	29
5.11	Interconnected among them and facilitate our daily life	30
5.12	Internet of Things Devices	31
5.13	IoT Devices and Technologies	32
5.14	Arduino Device	33
5.15	Intel Galileo Gen 2 Board	33
5.16	Samsung Gear Fit	34
5.17	Sensor	34
5.18	Bluetooth Low Energy	35
5.19	Raspberry Pi	36
5.20	Arduino boards	37
5.21	ESP8266	37
5.22	Raspberry Pi Sense HAT	38
5.23	Pin Diagram	39
5.24	IoT Platform	40

FIGURE NO	TITLE	PAGE NO
6.1	NodeMCU Development board	44
6.2	NodeMCU Development Board/kit v1.0 (Version2)	45
6.3	Flex Sensor	48
6.4	Bend angle	49
6.5	Types of Flex Sensors	49
6.6	Basic Flex Sensor Circuit	50
6.7	Arduino Flex Sensor Hook-up Guide	52
6.8	Circuit Design	53
6.9	Circuit diagram of Servo Motor control	55
6.10	Applications of Flex Sensor	57
6.11	Working of a Heartbeat Sensor	59
6.12	Heart Rate Sensor	60
6.13	Application Developing your own Heartbeat Sensor System	61
6.14	Thermocouple	63
6.15	Temperature sensor	65
7.1	Values verified	72
7.2	Finger_1	73
7.3	Finger_1 alert message	73
7.4	Finger_2	74
7.5	Finger_2 alert message	74
7.6	Pulse Sensor	75
7.7	Respiratory sensor	75
7.8	Temperature sensor	76

LIST OF ABBREVIATIONS

ADC	-	Analog-To-Digital Converter
AWS	-	Amazon Web Services
BLE	-	Bluetooth Low Energy
CG RAM	-	Character Generator RAM
DD RAM	-	Display Data RAM
DR	-	Data Register
IC	-	Integrated Circuit
IoT	-	Internet of Things
IR	-	Instruction Register
LAN	-	Local Area Network
LDR	-	Light Dependent Resistor
RTDs	-	Resistance Temperature Detectors
USB	-	Universal Serial Bus
UX	-	User Experience

CHAPTER 1

INTRODUCTION

Patient Monitoring system is a process of continuously tracking the patient's vital parameters.

Usually, the patient monitoring is carried out by attaching the sensors.

The components involved including the sensors network, Display Devices, Communication Wireless.

Nodes and other supporting components Sensors are nothing but the Transducers which is used to capture all the physical quantities of the patient.

Display devices are the devices which are used to accumulate the desired received signal and display in the appropriate content in the LCD or HMI displays.

Communication devices, are usually shorter distance communication devices.

Usually, the other supporting components involved in the patient monitoring system are the microcontroller units.

CHAPTER 2

LITERATURE SURVEY

Title: IOT BASED PARALYSIS PATIENT HEALTH CARE MONITORING SYSTEM

Author: Ms.N.ReneeSegridReddiyar, S.Remina, S.Sabrin, M.Subhashini

Year: June – 2021

Content: We all know that the paralysis condition is a loss of muscle function in the body parts. It can affect any part of your body at any time, then probably you may won't feel the pain the affected area. Technical and Therapeutic innovations are there to improve the quality of life Our goal is to develop a device which should be easy to use and should be affordable which consists of basic health care monitoring system with nursing care.

Title: IOT BASED AUTOMATED PARALYSIS PATIENT HEALTHCARE SYSTEM

Author: Chandan.V. Jha, Ms.Dipalee .M. Kate

Year: 2018 April

Content: We come across hospitals and NGO"s serving disabled people. Now these people are not capable of full body movement as compared to a normal person. In such a situation we propose a system that helps disabled person display a message by just simple motion of any part of his body.

Title: AUTOMATED PARALYSIS PATIENT HEALTH CARE SYSTEM

Author: Vidya Sarode, K.R Alex Rappai, Victor Thomas, Akash Dubey, Shashank Shukla

Year: May 2021

Content: The noble aim behind this project is to design a health care system which will be helpful for paralyzed and mute people. A Dumb individual all through the world uses gesture based communication for the correspondence. The progression in implanted framework can give a space to plan and build up an interpreter framework to change over the communication via gestures into discourse.

Title: AUTOMATED PARALYSIS PATIENT HEALTH CARE MONITORING SYSTEM

Author: Deepasri. T Gokulpriya. M. Arunkumar. GMohanraj. P Mrs. M. Shenbagapriya

Year: 03/03/2017

Content: Healthcare systems are a very important part of the economy of any country and for the public health. In this fast pace of life, it is difficult for people to be constantly available for their near ones who might need them while they are suffering from the difficult. Patient monitoring systems measure physiological characteristics either continuously or at regular intervals of time.

CHAPTER 3

PROBLEM DEFINATION OBJECTIVES

In existing system of methodology, only there exists the technology of monitoring the patient heart rate in the nearby terminal display unit and the remote monitoring is enabled only when there exists the WIFI connectivity.

1. In this monitoring system, the implementation of the module is for the paralyzed patients.
2. This module comprises of parameters include heart rate, respiration rate, temperature.
3. The primary function of this system is to monitor the heart rate, breathing rate, temperature of the paralyzed and the data collected by the sensors are sent to the cloud from there we can get alerts.

3.1 PROPOSED SYSTEM

In the proposed system Paralysis is the inability to move muscles on your own and with purpose.

It can be temporary or permanent. The most common causes are stroke, spinal cord injury, and multiple sclerosis.

Paralysis can be a complete loss of movement known as a significant weakness called paresis. Paralysis is most often caused by damage in the nervous system, especially the spinal cord.

Paralysis is caused by injury or disease affecting the central nervous system (brain and spinal cord) which means that the nerve signals sent to the muscles is interrupted.

Even though, there are innovative approaches for curing or treating paralysis patients.

But the aim of treatment is to help a person adapt to life with paralysis by making them as independent as possible.

Where we see a problem with these types of devices that are being developed is that they are very large and expensive machines.

Our goal is to make a device that will be able to retrain a patient's motion but have they are able to use the device themselves and have it be cheap enough for them to afford without much debt.

3.2 BLOCK DIAGRAM

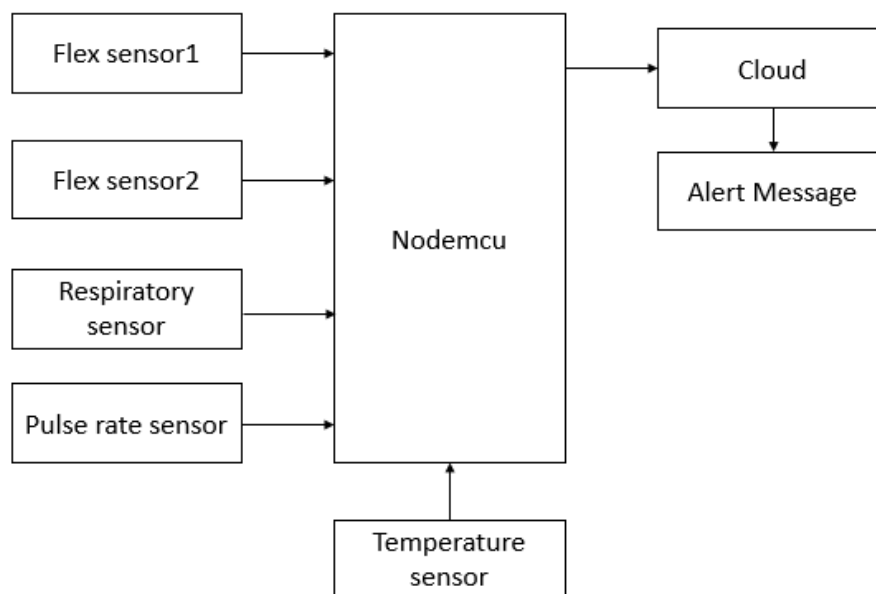


Fig. 3.1: Block diagram

3.3 HARDWARE REQUIRED

- Nodemcu
- Flex sensors
- Pulse rate sensor
- Temperature sensor
- Respiratory sensor

CHAPTER 4

EMBEDDED SYSTEMS

4.1 OVERVIEW OF EMBEDDED SYSTEMS

An embedded system is a special-purpose computer system designed to perform one or a few dedicated functions, often with real-time computing constraints. It is usually embedded as part of a complete device including hardware and mechanical parts. In contrast, a general-purpose computer, such as a personal computer, can do many different tasks depending on programming. Embedded systems have become very important today as they control many of the common devices we use.

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

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

In general, "embedded system" is not an exactly defined term, as many systems have some element of programmability. For example, Handheld computers share some elements with embedded systems — such as the operating systems and microprocessors which power them — but are not truly embedded systems, because they allow different applications to be loaded and peripherals to be connected.

Embedded systems provide several functions

- Monitor the environment; embedded systems read data from input sensors. This data is then processed and the results displayed in some format to a user or users
- Control the environment; embedded systems generate and transmit commands for actuators.
- Transform the information; embedded systems transform the data collected in some meaningful way, such as data compression / decompression

Although interaction with the external world via sensors and actuators is an important aspect of embedded systems, these systems also provide functionality specific to their applications. Embedded systems typically execute applications such as control laws, finite state machines, and signal processing algorithms. These systems must also detect and react to faults in both the internal computing environment as well as the surrounding electromechanical systems.

There are many categories of embedded systems, from communication devices to home appliances to control systems. Examples include;

- Communication devices
e.g.: modems, cellular phones
- Home Appliances
e.g.: CD player, VCR, microwave oven
- Control Systems
e.g.: Automobile anti-lock braking systems, robotics, satellite control

4.2 BLOCK DIAGRAM OF AN EMBEDDED SYSTEM

An embedded system usually contains an embedded processor. Many appliances that have a digital interface -- microwaves, VCRs, cars -- utilize embedded systems. Some embedded systems include an operating system. Others are very specialized resulting in the entire logic being implemented as a single program. These systems are embedded into some device for some specific purpose other than to provide general purpose computing. A typical embedded system is shown in Figure 4.1.

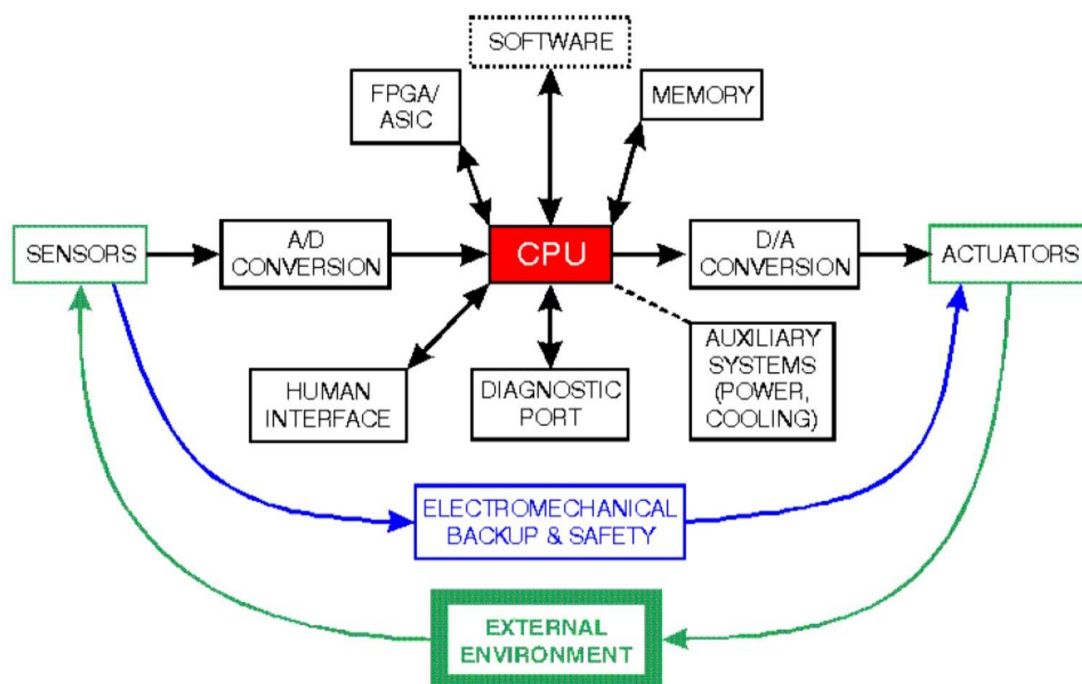


Fig. 4.1: Block diagram of a typical embedded system

4.2.1 Characteristics of Embedded Systems

Embedded systems are characterized by a unique set of characteristics. Each of these characteristics imposed a specific set of design constraints on embedded systems designers. The challenge to designing embedded systems is to conform to the specific set of constraints for the application.

Application Specific Systems

Embedded systems are not general-purpose computers. Embedded system designs are optimized for a specific application. Many of the job characteristics are known before the hardware is designed. This allows the designer to focus on the specific design constraints of a well-defined application. As such, there is limited user reprogram ability. Some embedded systems, however, require the flexibility of reprogram ability. Programmable DSPs are common for such applications.

Reactive Systems

As mentioned earlier, a typical embedded systems model responds to the environment via sensors and control the environment using actuators. This requires embedded systems to run at the speed of the environment. This characteristic of embedded system is called “reactive”. Reactive computation means that the system (primarily the software component) executes in response to external events. External events can be either periodic or aperiodic. Periodic events make it easier to schedule processing to guarantee performance. Aperiodic events are harder to schedule. The maximum event arrival rate must be estimated in order to accommodate worst case situations. Most embedded systems have a significant reactive component. One of the biggest challenges for embedded system designers is performing an accurate worst case design analysis on systems with statistical performance characteristics (e.g., cache memory on a DSP or other embedded processor). Real time system operation means that the correctness of a computation depends, in part, on the time at which it is delivered. Systems with this requirement must often design to worst case performance. But accurately predicting the worst case may be difficult on complicated architectures. This often leads to overly pessimistic estimates erring on the side of caution. Many embedded systems have a significant requirement for real time operation in order to meet external I/O and control stability requirements. Many real-time systems are also reactive systems.

Distributed Systems

A common characteristic of an embedded system is one that consists of communicating processes executing on several CPUs or ASICs which are connected by communication links. The reason for this is economy. Economical 4 8-bit microcontrollers may be cheaper than a 32-bit processors. Even after adding the cost of the communication links, this approach may be preferable. In this approach, multiple processors are usually required to handle multiple time-critical tasks. Devices under control of embedded systems may also be physically distributed.

Heterogeneous Architectures

Embedded systems often are composed of heterogeneous architectures (Figure 4.2). They may contain different processors in the same system solution. They may also be mixed signal systems. The combination of I/O interfaces, local and remote memories, and sensors and actuators makes embedded system design truly unique. Embedded systems also have tight design constraints, and heterogeneity provides better design flexibility.



Fig. 4.2: Embedded Systems having Heterogeneous Architectures

Harsh environment

Many embedded systems do not operate in a controlled environment. Excessive heat is often a problem, especially in applications involving combustion (e.g., many transportation applications). Additional problems can be caused for

embedded computing by a need for protection from vibration, shock, lightning, power supply fluctuations, water, corrosion, fire, and general physical abuse.

System safety and reliability

As embedded system complexity and computing power continue to grow, they are starting to control more and more of the safety aspects of the overall system. These safety measures may be in the form of software as well as hardware control. Mechanical safety backups are normally activated when the computer system loses control in order to safely shut down system operation. Software safety and reliability is a bigger issue. Software doesn't normally "break" in the sense of hardware. However software may be so complex that a set of unexpected circumstances can cause software failures leading to unsafe situations. Discussion of this topic is outside the scope of this book, but the challenges for embedded designers include designing reliable software and building cheap, available systems using unreliable components. The main challenge for embedded system designers is to obtain low-cost reliability with minimal redundancy.

Control of physical systems

One of the main reasons for embedding a computer is to interact with the environment. This is often done by monitoring and controlling external machinery. Embedded computers transform the analog signals from sensors into digital form for processing. Outputs must be transformed back to analog signal levels. When controlling physical equipment, large current loads may need to be switched in order to operate motors and other actuators. To meet these needs, embedded systems may need large computer circuit boards with many non-digital components. Embedded system designers must carefully balance system tradeoffs among analog components, power, mechanical, network, and digital hardware with corresponding software.

Small and low weight

Many embedded computers are physically located within some larger system. The form factor for the embedded system may be dictated by aesthetics.

For example, the form factor for a missile may have to fit inside the nose of the missile. One of the challenges for embedded systems designers is to develop non-rectangular geometries for certain solutions. Weight can also be a critical constraint. Embedded automobile control systems, for example, must be light weight for fuel economy. Portable CD players must be light weight for portability purposes.

Cost sensitivity

Cost is an issue in most systems, but the sensitivity to cost changes can vary dramatically in embedded systems. This is mainly due to the effect of computer costs have on profitability and is more a function of the proportion of cost changes compared to the total system cost.

Power management

Embedded systems have strict constraints on power. Given the portability requirements of many embedded systems, the need to conserve power is important to maintain battery life as long as possible. Minimization of heat production is another obvious concern for embedded systems.

4.3 POWER SUPPLY

All electronic circuits works only in low DC voltage, so we need a power supply unit to provide the appropriate voltage supply for their proper functioning. This unit consists of transformer, rectifier, filter & regulator. AC voltage of typically 230volts rms is connected to a transformer voltage down to the level to the desired ac voltage. A diode rectifier that provides the full wave rectified voltage that is initially filtered by a simple capacitor filter to produce a dc voltage. This resulting dc voltage usually has some ripple or ac voltage variation . A regulator circuit can use this dc input to provide dc voltage that not only has much less ripple voltage but also remains the same dc value even the dc voltage varies somewhat, or the load connected to the output dc voltages changes.

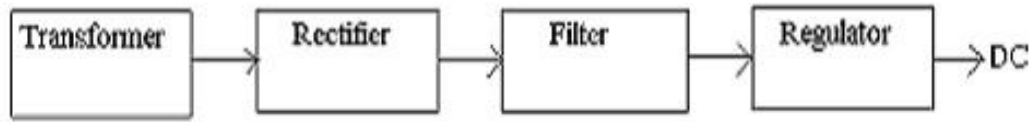


Fig. 4.3: General Block of Power Supply Unit

4.4 TRANSFORMER

A transformer is a static piece of which electric power in one circuit is transformed into electric power of same frequency in another circuit. It can raise or lower the voltage in the circuit, but with a corresponding decrease or increase in current. It works with the principle of mutual induction. In our project we are using a step down transformer to providing a necessary supply for the electronic circuits. Here we step down a 230volts ac into 12volts ac.

4.5 RECTIFIER

A dc level obtained from a sinusoidal input can be improved 100% using a process called full wave rectification. Here in our project for full wave rectification we use bridge rectifier. From the basic bridge configuration we see that two diodes(say D2 & D3) are conducting while the other two diodes (D1 & D4) are in off state during the period $t = 0$ to $T/2$. Accordingly for the negative cycle of the input the conducting diodes are D1 & D4 .Thus the polarity across the load is the same.

In the bridge rectifier the diodes may be of variable types like 1N4001, 1N4003, 1N4004, 1N4005, 1N4007 etc.... can be used. But here we use 1N4007, because it can withstand up to 1000v.

4.6 FILTERS

In order to obtain a dc voltage of 0 Hz, we have to use a low pass filter. So that a capacitive filter circuit is used where a capacitor is connected at the rectifier output & a dc is obtained across it. The filtered waveform is essentially a dc voltage with negligible ripples & it is ultimately fed to the load.

4.7 REGISTERS

The controller IC has two 8 bit registers, an instruction register (IR) and a data register (DR). The IR stores the instruction codes and address information for display data RAM (DD RAM) and character generator RAM (CG RAM). The IR can be written, but not read by the MPU.

The DR temporally stores data to be written to /read from the DD RAM or CG RAM. The data written to DR by the MPU, is automatically written to the DD RAM or CG RAM as an internal operation.

When an address code is written to IR, the data is automatically transferred from the DD RAM or CG RAM to the DR. data transfer between the MPU is then completed when the MPU reads the DR. likewise, for the next MPU read of the DR, data in DD RAM or CG RAM at the address is sent to the DR automatically. Similarly, for the MPU write of the DR, the next DD RAM or CG RAM address is selected for the write operation.

The dot-matrix liquid crystal display controller and driver LSI displays alphanumeric, Japanese kana characters, and symbols. It can be configured to drive a dot-matrix liquid crystal display under the control of a 4- or 8-bit microprocessor. Since all the functions such as display RAM, character generator, and liquid crystal driver, required for driving a dot-matrix liquid crystal display are internally provided on one chip, a minimal system can be interfaced with this controller/driver

4.8 REGULATORS

The output voltage from the capacitor is more filtered & finally regulated. The voltage regulator is a device, which maintains the output voltage constant irrespective of the change in supply variations, load variations & temperature changes. Here we use fixed voltage regulator namely LM7805. The IC LM7805 is a +5v regulator which is used for microcontroller.

4.9

FEATURES AND DESCRIPTION OF REGULATORS

- Output Current up to 1A
- Output Voltages of 5, 6, 8, 9, 10, 12, 15, 18, 24V
- Thermal Overload Protection
- Short Circuit Protection
- Output Transistor Safe Operating Area Protection

CHAPTER 5

INTRODUCTION ABOUT IOT

IoT stands for Internet of Things, which means accessing and controlling daily usable equipments and devices using Internet.

Our IoT tutorial includes all topics of IoT such as introduction, features, advantage and disadvantage, ecosystem, decision framework, architecture and domains, biometric, security camera and door unlock system, devices, etc.

5.1 WHAT IS AN INTERNET OF THINGS (IoT)

Let's us look closely at our mobile device which contains GPS Tracking, Mobile Gyroscope, Adaptive brightness, Voice detection, Face detection etc. These components have their own individual features, but what about if these all communicate with each other to provide a better environment? For example, the phone brightness is adjusted based on my GPS location or my direction.

Connecting everyday things embedded with electronics, software, and sensors to internet enabling to collect and exchange data without human interaction called as the Internet of Things (IoT).

The term "Things" in the Internet of Things refers to anything and everything in day to day life which is accessed or connected through the internet.

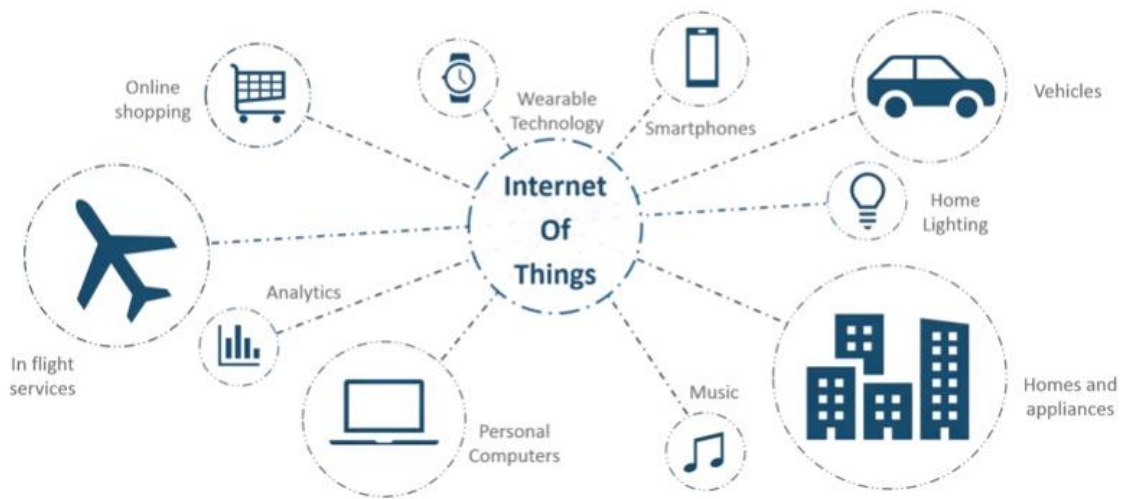


Fig. 5.1: Internet of Things

IoT is an advanced automation and analytics system which deals with artificial intelligence, sensor, networking, electronic, cloud messaging etc. to deliver complete systems for the product or services. The system created by IoT has greater transparency, control, and performance.

As we have a platform such as a cloud that contains all the data through which we connect all the things around us. For example, a house, where we can connect our home appliances such as air conditioner, light, etc. through each other and all these things are managed at the same platform. Since we have a platform, we can connect our car, track its fuel meter, speed level, and also track the location of the car.



Fig. 5.2: Example of IoT

If there is a common platform where all these things can connect to each other would be great because based on my preference, I can set the room temperature. For example, if I love the room temperature to to be set at 25 or 26-

degree Celsius when I reach back home from my office, then according to my car location, my AC would start before 10 minutes I arrive at home. This can be done through the Internet of Things (IoT).

5.2 HOW DOES INTERNET OF THING (IOT) WORK?

The working of IoT is different for different IoT echo system (architecture). However, the key concept of there working are similar. The entire working process of IoT starts with the device themselves, such as smartphones, digital watches, electronic appliances, which securely communicate with the IoT platform. The platforms collect and analyze the data from all multiple devices and platforms and transfer the most valuable data with applications to devices.

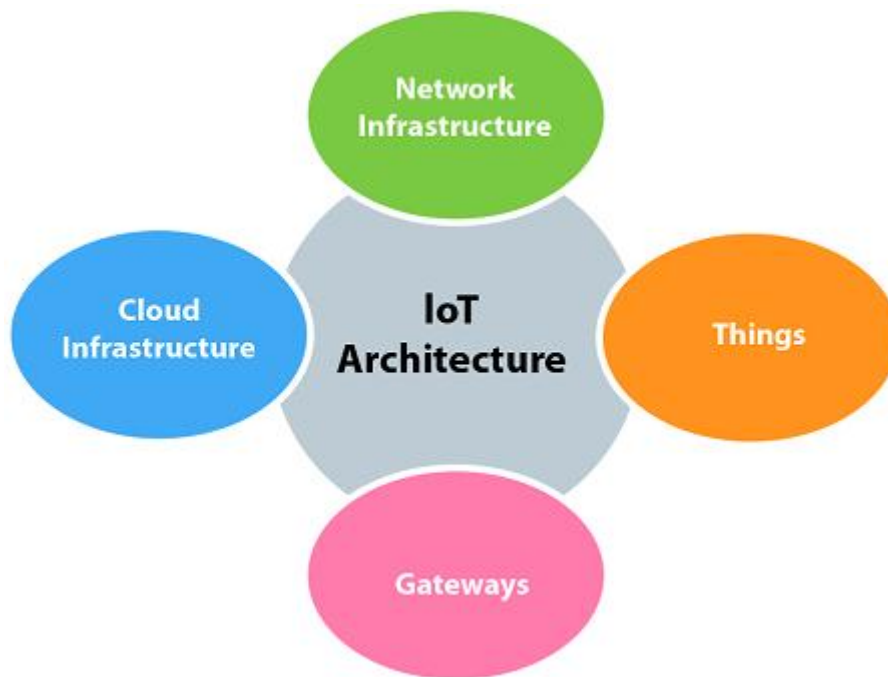


Fig. 5.3: IoT Architecture

5.3 FEATURES OF IOT

The most important features of IoT on which it works are connectivity, analyzing, integrating, active engagement, and many more. Some of them are listed below:

Connectivity: Connectivity refers to establish a proper connection between all the things of IoT to IoT platform it may be server or cloud. After connecting the IoT devices, it needs a high speed messaging between the devices and cloud to enable reliable, secure and bi-directional communication.

Analyzing: After connecting all the relevant things, it comes to real-time analyzing the data collected and use them to build effective business intelligence. If we have a good insight into data gathered from all these things, then we call our system has a smart system.

Integrating: IoT integrating the various models to improve the user experience as well.

Artificial Intelligence: IoT makes things smart and enhances life through the use of data. For example, if we have a coffee machine whose beans have going to end, then the coffee machine itself order the coffee beans of your choice from the retailer.

Sensing: The sensor devices used in IoT technologies detect and measure any change in the environment and report on their status. IoT technology brings passive networks to active networks. Without sensors, there could not hold an effective or true IoT environment.

Active Engagement: IoT makes the connected technology, product, or services to active engagement between each other.

Endpoint Management: It is important to be the endpoint management of all the IoT system otherwise, it makes the complete failure of the system. For example, if a coffee machine itself order the coffee beans when it goes to end but what happens when it orders the beans from a retailer and we are not present at home for a few days, it leads to the failure of the IoT system. So, there must be a need for endpoint management.

5.4 ADVANTAGES AND DISADVANTAGES OF IoT

Any technology available today has not reached to its 100 % capability. It always has a gap to go. So, we can say that **Internet of Things** has a significant technology in a world that can help other technologies to reach its accurate and complete 100 % capability as well.

Let's take a look over the major, advantages, and disadvantages of the Internet of Things.

5.4.1 *Advantages of IoT*

Internet of things facilitates the several advantages in day-to-day life in the business sector. Some of its benefits are given below:

- ***Efficient resource utilization:*** If we know the functionality and the way that how each device work we definitely increase the efficient resource utilization as well as monitor natural resources.
- ***Minimize human effort:*** As the devices of IoT interact and communicate with each other and do lot of task for us, then they minimize the human effort.
- ***Save time:*** As it reduces the human effort then it definitely saves out time. Time is the primary factor which can save through IoT platform.
- ***Enhance Data Collection:***
- ***Improve security:*** Now, if we have a system that all these things are interconnected then we can make the system more secure and efficient.

5.4.2 *Disadvantages of IoT*

As the Internet of things facilitates a set of benefits, it also creates a significant set of challenges. Some of the IoT challenges are given below:

5.5 EMBEDDED DEVICES SYSTEM IN IoT

It is essential to know about the embedded devices while learning the IoT or building the projects on IoT. The embedded devices are the objects that build the unique computing system. These systems may or may not connect to the Internet.

An embedded device system generally runs as a single application. However, these devices can connect through the internet connection, and able communicate through other network devices.

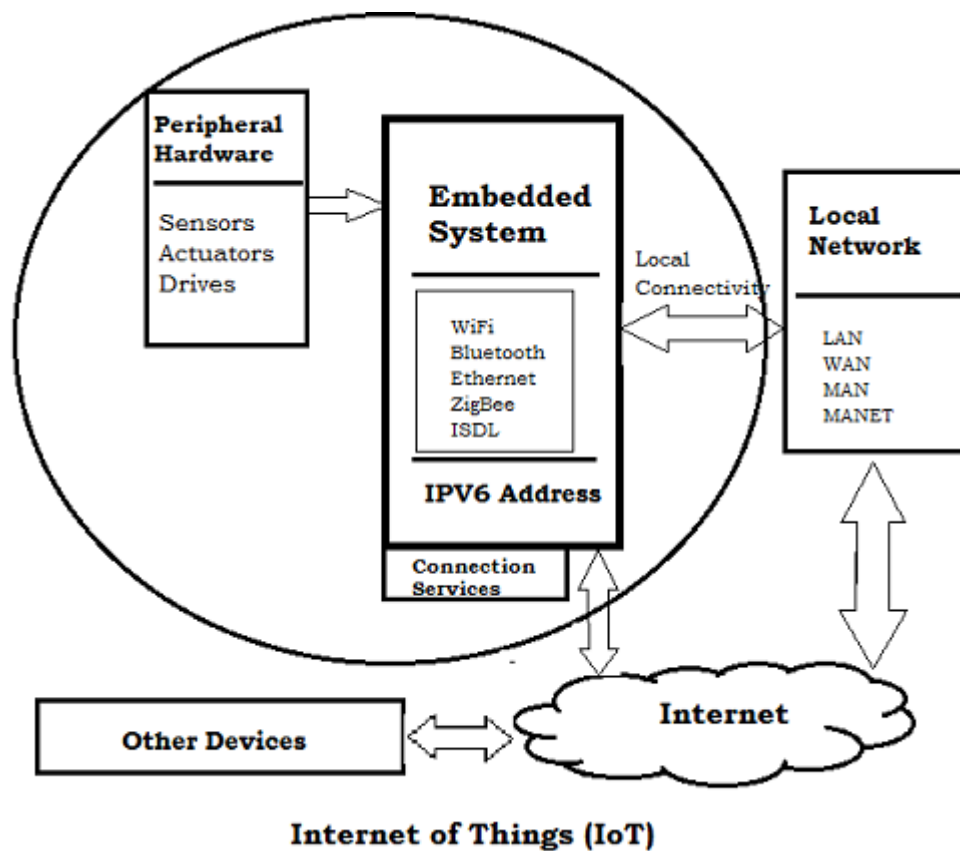


Fig. 5.4 : Embedded Devices System in (IoT)

5.5.1 Embedded System Hardware

The embedded system can be of type microcontroller or type microprocessor. Both of these types contain an integrated circuit (IC).

The essential component of the embedded system is a RISC family microcontroller like Motorola 68HC11, PIC 16F84, Atmel 8051 and many more. The

most important factor that differentiates these microcontrollers with the microprocessor like 8085 is their internal read and writable memory. The essential embedded device components and system architecture are specified below.

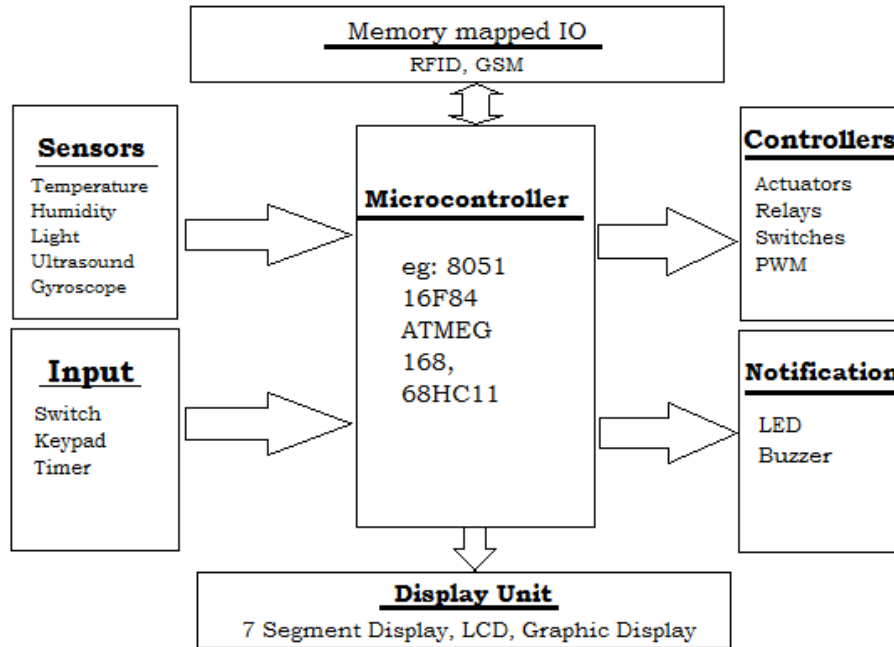


Fig. 5.5: Basic Embedded System

5.5.2 Embedded System Software

The embedded system that uses the devices for the operating system is based on the language platform, mainly where the real-time operation would be performed. Manufacturers build embedded software in electronics, e.g., cars, telephones, modems, appliances, etc. The embedded system software can be as simple as lighting controls running using an 8-bit microcontroller. It can also be complicated software for missiles, process control systems, airplanes etc.

5.6 IoT ECOSYSTEM

The IoT ecosystem is not easy to define. It is also difficult to capture its proper image due to the vastness and emerging possibility and the rapidity with which it is expanding in the entire sector. However, the IoT ecosystem is a connection of various kind of devices that sense and analyze the data and communicates with each other over the networks.

In the IoT ecosystem, the user uses smart devices such as smartphones, tablet, sensors, etc. to send the command or request to devices for information over the networks. The device response and performs the command to send information back to the user through networks after analyzed.

The typical IoT ecosystem is shown in below image, where the smarter devices send and receive data from the devices themselves in the environment that are integrate over network and Cloud Computing.

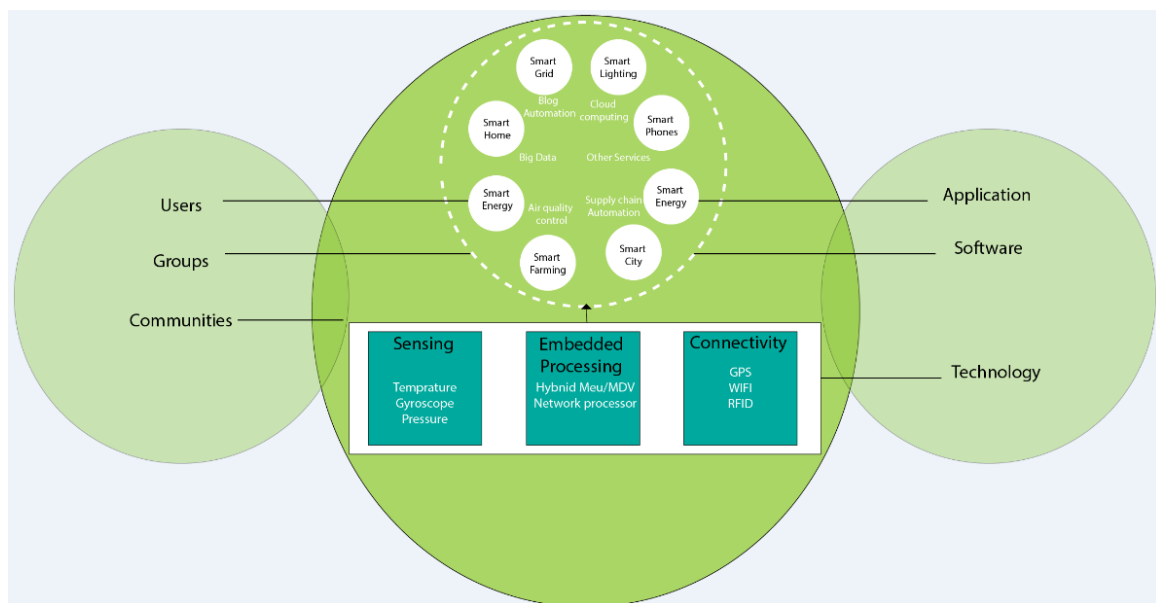


Fig. 5.6: IoT Ecosystem

The IoT is itself an ecosystem of network devices that transfer the data. It is also well interconnected with Big Data and Cloud Computing.

- **Sensing, Embedded processing, Connectivity:** The IoT ecosystem senses its surrounding like temperature, gyroscope, pressure, etc. and make the embedded processing using devices. These devices are connected through any type of devices such as GPS, WiFi, RFID, etc. over the networks.
- **Smart devices and environment, Cloud Computing, Big Data:** The data transfer or receive through smart devices and

environments are communicated through Cloud Computing or others Servers and stored as Big Data.

- **Technology, Software, Application:** The IoT ecosystem uses any of different technologies, software and application to communicate and connect with smart devices and environment.
- **Users or groups of community:** The product or services generated by the IoT ecosystem are consumed by the users or the group of communities to serve the smart life.

5.7 IoT DECISION FRAMEWORK

The IoT decision framework provides a structured approach to create a powerful IoT product strategy. The IoT decision framework is all about the strategic decision making. The IoT Decision Framework helps us to understand the areas where we need to make decisions and ensures consistency across all of our strategic business decision, technical and more.

The IoT decision framework is much more important as the product or services communicates over networks goes through five different layers of complexity of technology.

1. Device Hardware
2. Device Software
3. Communications
4. Cloud Platform
5. Cloud Application

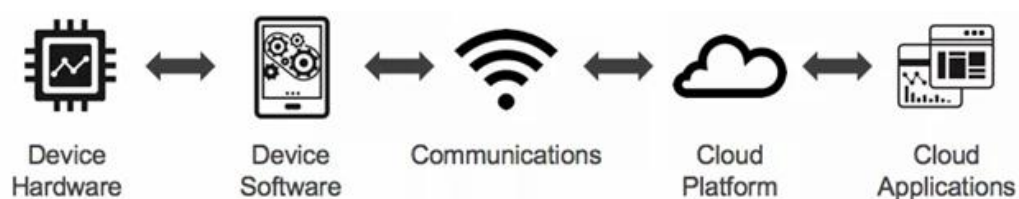


Fig. 5.7: The IoT Technology stack

5.7.1 Decision Area

The IoT decision framework pays attention to six key decision areas in any IoT product. These decision areas are:

1. User Experience (UX)
2. Data
3. Business
4. Technology
5. Security
6. Standards & Regulations

Each of these decision areas is evaluated at each of the IoT Technology Stack. The User Experience will be evaluated at Device Hardware, Device Software and so to provide the better user experience. Then at the next step Data Decision Area, we have to explore data considerations for all the stages of IoT Technology Stack.

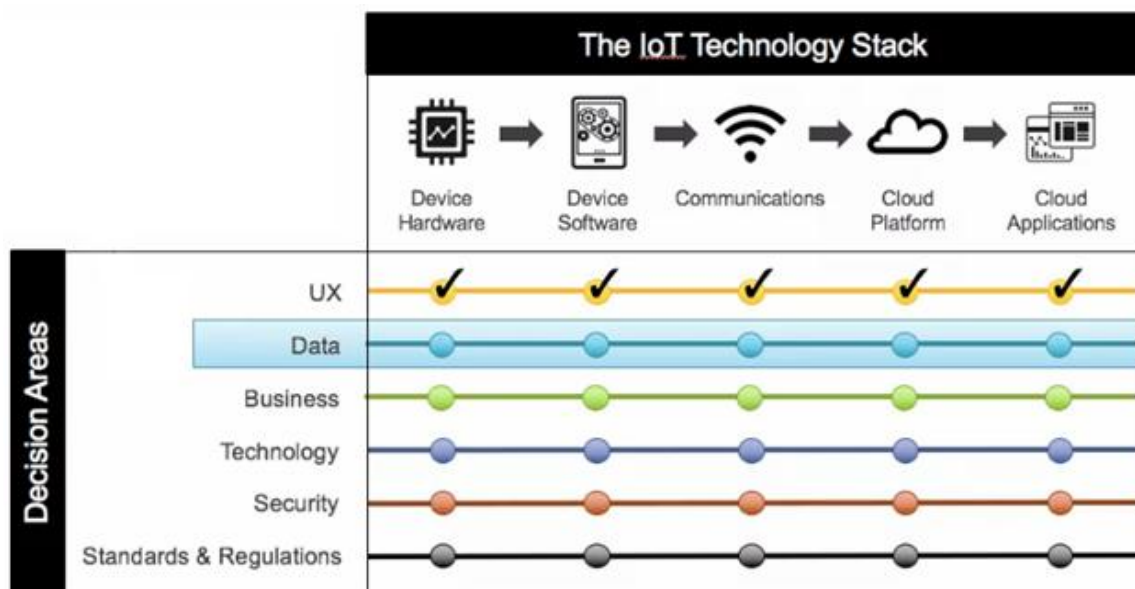


Fig. 5.8: Decision Area of the IoT Decision Framework

Let's see each of the Decision Area of IoT Decision Framework in detail:

1. **User Experience Decision Area:** This is the area where we concentrate about who are the users, what are their requirements and how to provide a great experience at each step of IoT stack without worrying about the technical details.
2. **Data Decision Area:** In this area, we make the overall data strategy such as the data flow over the entire IoT stack to fulfill the user's requirements.
3. **Business Decision Area:** Based on the previous decisions area, we make the decision how product or services will become financial potential. At each of the IoT Stack level are monetized about the costs of providing services.
4. **Technology Decision Area:** In this area, we work with the technology for each layer to facilitate the final solution.
5. **Security Decision Area:** After going through the implementation of technology it is important to decide and provide the security at each stage of the IoT Stack.
6. **Standards & Regulations Decision Area:** At the last stage of IoT Decision Area, we identify the standards and regulations of product or services that will affect your product at each layer of the IoT Stack.

5.8 IoT ARCHITECTURE

There is not such a unique or standard consensus on the Internet of Things (IoT) architecture which is universally defined. The IoT architecture differs from their functional area and their solutions. However, the IoT architecture technology mainly consists of four major components:

Components of IoT Architecture

- Sensors/Devices
- Gateways and Networks
- Cloud/Management Service Layer
- Application Layer

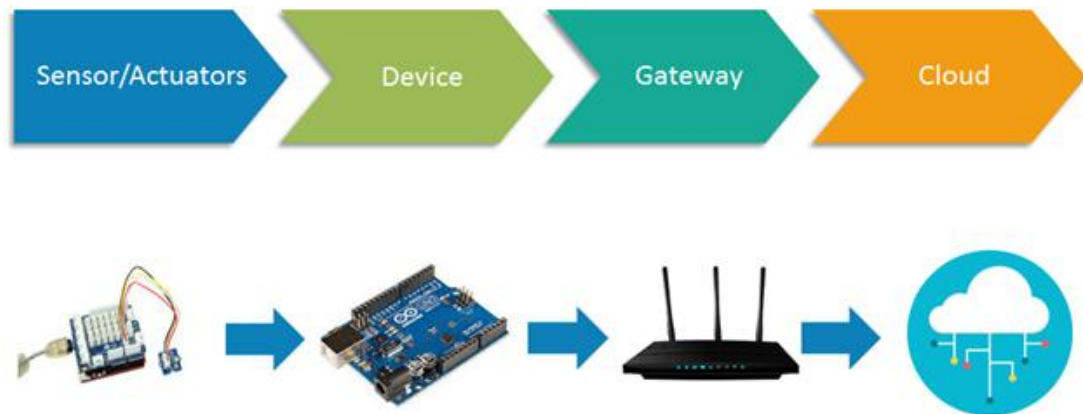


Fig. 5.9: Stages of IoT Solutions Architecture

There are several layers of IoT built upon the capability and performance of IoT elements that provides the optimal solution to the business enterprises and end-users. The IoT architecture is a fundamental way to design the various elements of IoT, so that it can deliver services over the networks and serve the needs for the future.

Following are the primary stages (layers) of IoT that provides the solution for IoT architecture.

1. **Sensors/Actuators:** Sensors or Actuators are the devices that are able to emit, accept and process data over the network. These sensors or actuators may be connected either through wired or wireless. This contains GPS, Electrochemical, Gyroscope, RFID, etc. Most of the sensors need connectivity through sensors gateways. The connection of sensors or actuators can be through a Local Area Network (LAN) or Personal Area Network.

2. **Gateways and Data Acquisition:** As the large numbers of data are produced by this sensors and actuators need the high-speed Gateways and Networks to transfer the data. This network can be of type Local Area Network (LAN such as WiFi, Ethernet, etc.), Wide Area Network (WAN such as GSM, 5G, etc.).
3. **Edge IT:** Edge in the IoT Architecture is the hardware and software gateways that analyze and pre-process the data before transferring it to the cloud. If the data read from the sensors and gateways are not changed from its previous reading value then it does not transfer over the cloud, this saves the data used.
4. **Data center/ Cloud:** The Data Center or Cloud comes under the Management Services which process the information through analytics, management of device and security controls. Beside this security controls and device management the cloud transfer the data to the end users application such as Retail, Healthcare, Emergency, Environment, and Energy, etc.

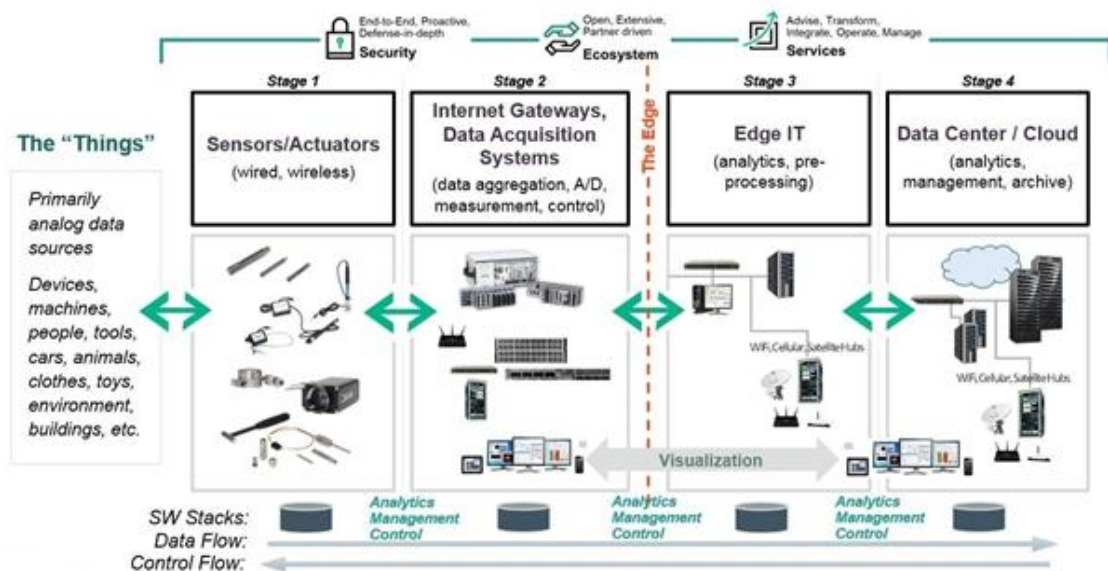


Fig. 5.10: The 4 stage IoT Solutions Architecture

What are Smarts Objects in IoT

The concept of smart in IoT is used for physical objects that are active, digital, networked, can operate to some extent autonomously, reconfigurable and has local control of the resources. The smart objects need energy, data storage, etc.

A **smart object** is an object that enhances the interaction with other smart objects as well as with people also. The world of IoT is the network of interconnected heterogeneous objects (such as smart devices, smart objects, sensors, actuators, RFID, embedded computers, etc.) uniquely addressable and based on standard communication protocols.

In a day to day life, people have a lot of object with internet or wireless or wired connection. Such as:

- Smartphone
- Tablets
- TV computer

These objects can be interconnected among them and facilitate our daily life (smart home, smart cities) no matter the situation, localization, accessibility to a sensor, size, scenario or the risk of danger.



Fig. 5.11: Interconnected among them and facilitate our daily life

Smart objects are utilized widely to transform the physical environment around us to a digital world using the Internet of things (IoT) technologies.

A smart object carries blocks of application logic that make sense for their local situation and interact with human users. A smart object sense, log, and interpret the occurrence within themselves and the environment, and intercommunicate with each other and exchange information with people.

The work of smart object has focused on technical aspects (such as software infrastructure, hardware platforms, etc.) and application scenarios. Application areas range from supply-chain management and enterprise applications (home and hospital) to healthcare and industrial workplace support. As for human interface aspects of smart-object technologies are just beginning to receive attention from the environment.

5.9 IoT DEVICES

Internet of Things Devices is non-standard devices that connect wirelessly to a network with each other and able to transfer the data. IoT devices are enlarging the internet connectivity beyond standard devices such as smartphones, laptops, tablets, and desktops. Embedding these devices with technology enable us to communicate and interact over the networks and they can be remotely monitored and controlled.

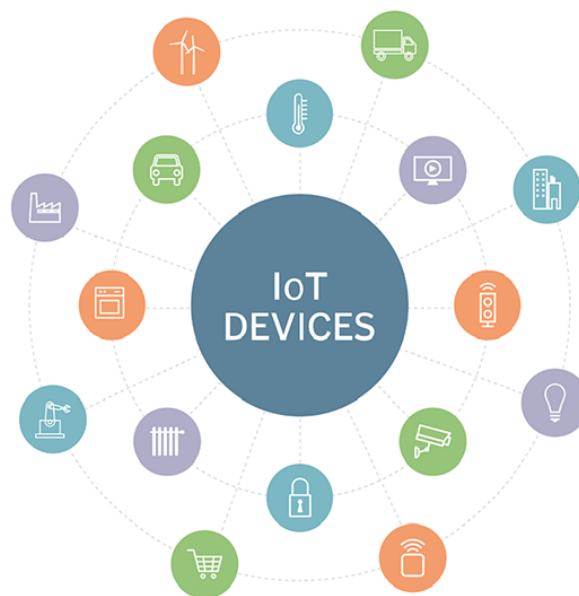


Fig.5.12: Internet of Things Devices

There are large varieties of IoT devices available based on IEEE 802.15.4 standard. These devices range from wireless motes, attachable sensor-boards to interface-board which are useful for researchers and developers.

IoT devices include computer devices, software, wireless sensors, and actuators. These IoT devices are connected over the internet and enabling the data transfer among objects or people automatically without human intervention.

Some of the common and popular IoT devices are given below:

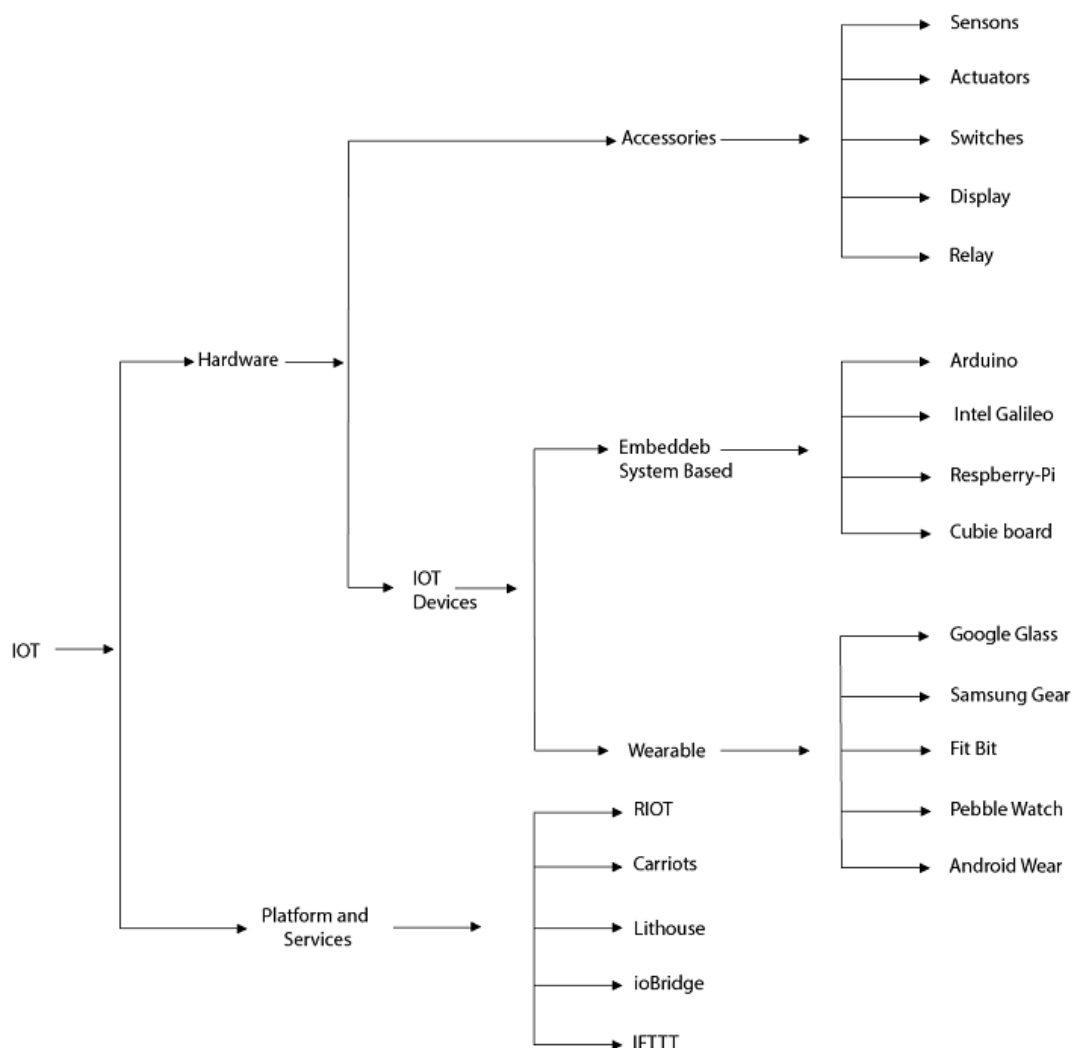


Fig. 5.13: IoT Devices and Technologies

5.10 ARDUINO DEVICE

Arduino devices are the microcontrollers and microcontroller kit for building digital devices that can be sense and control objects in the physical and digital world. Arduino boards are furnished with a set of digital and analog input/output pins that may be interfaced to various other circuits. Some Arduino boards include USB (Universal Serial Bus) used for loading programs from the personal computer.

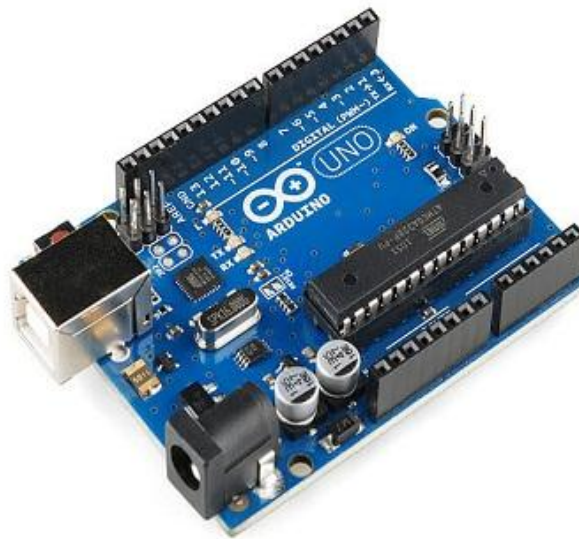


Fig. 5.14: Arduino Device

Intel Galileo

The Intel Galileo Gen 2 Board includes the parts such as Intel Quark SoC processor, 256MB RAM, multiple ports and supports for Arduino device.



Fig. 5.15: Intel Galileo Gen 2 Board

Samsung Gear Fit

A Samsung Gear Fit device is a dustproof, water-resistant with fitness tracker features, a curved display, and long-lasting battery. This device receives alerts about emails and text messages, and it integrates with Samsung's S Health app.



Fig. 5.16: Samsung Gear Fit

Sensor

A sensor is a device that reads the surrounding temperature, humidity, light, air quality control etc. There are different types of sensors available that reads different types of data. The sensors transmit these data over the networks or through which it is connected.



Fig. 5.17: Sensor

Bluetooth Low Energy (BLE) Intelligent Beacon

A Bluetooth low energy beacon device is used to track the object located at a real time. Many companies use it to track the location of employees, assets, patients, and more in real time. This service primarily focuses on manufacturing, retail, and healthcare services.



Fig. 5.18: Bluetooth Low Energy

5.11 PROPERTIES OF IoT DEVICES

Some of the essential properties of IoT devices are mention below:

- ***Sense:*** The devices that sense its surrounding environment in the form of temperature, movement, and appearance of things, etc.
- ***Send and receive data:*** IoT devices are able to send and receive the data over the network connection.
- ***Analyze:*** The devices can able to analyze the data that received from the other device over the internet networks.
- ***Controlled:*** IoT devices may control from some endpoint also. Otherwise, the IoT devices are themselves communicate with each other endlessly leads to the system failure.

Major IoT Boards in Market

There are several IoT boards available in the market to build the project. Some of the major IoT Boards are described below:

Raspberry Pi

Raspberry Pi is a much popular device used in building IoT project. The recently launched Raspberry Pi 3 includes built-in WiFi and Bluetooth making the most compact and standalone computer. It provides a powerful environment to install a variety of programming packages such as Python, Node.js, LAMP stack, Java and much more. Using 40 GPIO pins, and four USB ports you can connect many peripherals and accessories to the Pi.



Fig. 5.19: Raspberry Pi

Arduino

Arduino boards are the microcontrollers and microcontroller kit for building digital devices that can be sense and control objects in the physical and digital world. Arduino boards are furnished with a set of digital and analog input/output pins that may be interfaced to various other circuits. Some Arduino boards include USB (Universal Serial Bus) to load programs from the personal computer.

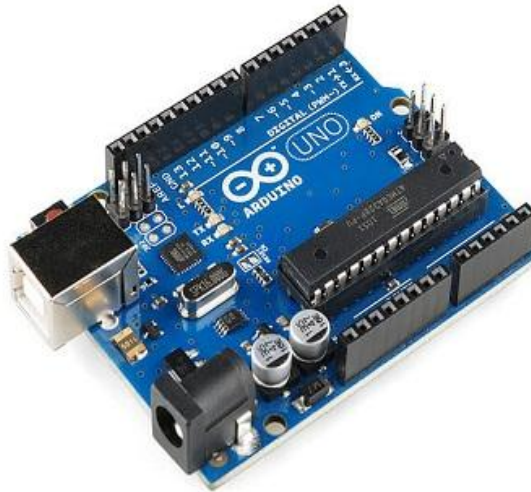


Fig. 5.20: Arduino boards

ESP8266

The ESP8266 is a low-cost Wi-Fi microchip with 32-bit microcontroller capability, standard digital peripheral interfaces. There are different types of ESP8266 boards available for different needs. The primary goal of this board is to deal with the built-in Wifi through AT commands if used as device module, but you can 'program' using Arduino board however it also read and controls input/output, digital and analog.



Fig. 5.21: ESP8266

Sense HAT 8x8 RGB LED matrix

Raspberry Pi Sense HAT is an integrated sensor that can measure humidity, temperature, acceleration, and pressure. The 8x8 LED matrix display data read from Raspberry Pi Sense HAT sensors. The Sense HAT has an 8×8 RGB LED matrix includes the following sensors:

1. Gyroscope
2. Accelerometer
3. Magnetometer
4. Temperature
5. Barometric pressure
6. Humidity



Fig. 5.22: Raspberry Pi Sense HAT

Bluetooth Module HC-05

Bluetooth Module HC-05 device is a 6 pins Bluetooth device that is used for wireless communication. Usually, this device connects small devices like mobile phones, PDAs and TVs using a short-range wireless connection to exchange data. It communicates with the microcontroller using the serial port (USART).

Pin Description

- **EN:** It is the enable pin, when it is connected to 3.3V then module is enabled.
- **+5V:** This is the supply pin for connecting +5V.
- **GND:** It is the ground pin.
- **TX:** It is the transmitter pin of the UART communication.
- **RX:** It is the receiver pin of UART communication.
- **STATE:** It indicates whether the module is connected or not. It acts as a status indicator.



Fig. 5.23: Pin Diagram

5.12 IoT - PLATFORM

As in IoT, all the IoT devices are connected to other IoT devices and application to transmit and receive information using protocols. There is a gap between the IoT device and IoT application. An IoT Platform fills the gap between the devices (sensors) and application (network). Thus we can say that **an IoT platform is an integrated service that fulfills the gap between the IoT device and application and offers you to bring physical object online.**

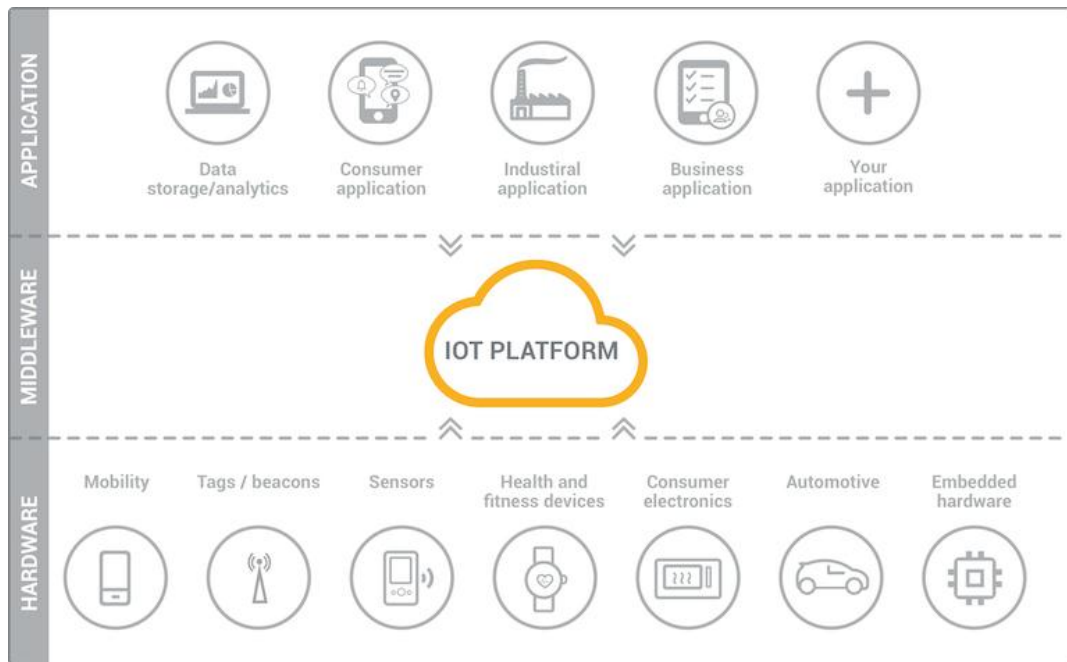


Fig. 5.24: IoT Platform

There are several IoT Platforms available that provides facility to deploy IoT application actively. Some of them are listed below:

Amazon Web Services (AWS) IoT platform: Amazon Web Service IoT platform offers a set of services that connect to several devices and maintain the security as well. This platform collects data from connected devices and performs real-time actions.

Microsoft Azure IoT platform: Microsoft Azure IoT platform offers strong security mechanism, scalability and easy integration with systems. It uses standard protocols that support bi-directional communication between connected devices and platform. Azure IoT platform has an Azure Stream Analytics that processes a large amount of information in real-time generated by sensors. Some common features provided by this platform are:

- Information monitoring
- A rules engine
- Device shadowing
- Identity registry

Google Cloud Platform IoT: Google Cloud Platform is a global cloud platform that provides a solution for IoT devices and applications. It handles a large amount of data using Cloud IoT Core by connecting various devices. It allows to apply BigQuery analysis or to apply Machine learning on this data. Some of the features provided by Google Cloud IoT Platform are:

- Cloud IoT Core
- Speed up IoT devices
- Cloud publisher-subscriber
- Cloud Machine Learning Engine

IBM Watson IoT platform: The IBM Watson IoT platform enables the developer to deploy the application and building IoT solutions quickly. This platform provides the following services:

- Real-time data exchange
- Device management
- Secure Communication
- Data sensor and weather data services

Artik Cloud IoT platform: Arthik cloud IoT platform is developed by Samsung to enable devices to connect to cloud services. It has a set of services that continuously connect devices to the cloud and start gathering data. It stores the incoming data from connected devices and combines this information. This platform contains a set of connectors that connect to third-party services.

Bosch IoT Suite:

Bosch cloud IoT Suit is based on Germany. It offers safe and reliable storing of data on its server in Germany. This platform supports full app development from prototype to application development.

How IoT platform help

- IoT Platform connects sensors and devices.
- IoT platform handles different software communication protocol and hardware.
- IoT platform provides security and authentication for sensors and users.
- It collects, visualizes, and analyzes the data gathered by the sensor and device.

CHAPTER 6

NODE MCU AND BRIEF ABOUT SENSORS

NodeMCU is an open source IoT platform.[†] It includes firmware which runs on the ESP8266 Wi-Fi SoC from Espressif Systems, and hardware which is based on the ESP-12 module. The term "NodeMCU" by default refers to the firmware rather than the development kits. The firmware uses the Lua scripting language. It is based on the eLua project, and built on the Espressif Non-OS SDK for ESP8266. It uses many open source projects, such as lua-cjson and SPIFFS.

NodeMCU was created shortly after the ESP8266 came out. On December 30, 2013, Espressif Systems began production of the ESP8266. The ESP8266 is a Wi-Fi SoC integrated with a Tensilica Xtensa LX106 core, widely used in IoT applications (see related projects). NodeMCU started on 13 Oct 2014, when Hong committed the first file of nodemcu-firmware to GitHub.^[11] Two months later, the project expanded to include an open-hardware platform when developer Huang R committed the gerber file of an ESP8266 board, named devkit v0.9. Later that month, Tuan PM ported MQTT client library from Contiki to the ESP8266 SoC platform, and committed to NodeMCU project, then NodeMCU was able to support the MQTT IoT protocol, using Lua to access the MQTT broker. Another important update was made on 30 Jan 2015, when Devsaurus ported the u8glib to NodeMCU project, enabling NodeMCU to easily drive LCD, Screen, OLED, even VGA displays.

In summer 2015 the creators abandoned the firmware project and a group of independent contributors took over. By summer 2016 the NodeMCU included more than 40 different modules. Due to resource constraints users need to select the modules relevant for their project and build a firmware tailored to their needs.

6.1 ESP8266 ARDUINO CORE

As Arduino.cc began developing new MCU boards based on non-AVR processors like the ARM/SAM MCU and used in the Arduino Due, they needed to modify the Arduino IDE so that it would be relatively easy to change the IDE to support alternate toolchains to allow Arduino C/C++ to be compiled for these new processors. They did this with the introduction of the Board Manager and the SAM Core. A "core" is the collection of software components required by the Board Manager and the Arduino IDE to compile an Arduino C/C++ source file for the target MCU's machine language. Some ESP8266 enthusiasts developed an Arduino core for the ESP8266 WiFi SoC, popularly called the "ESP8266 Core for the Arduino IDE".^[16] This has become a leading software development platform for the various ESP8266-based modules and development boards, including NodeMCUs.

Introduction

NodeMCU is an open source LUA based firmware developed for ESP8266 wifi chip. By exploring functionality with ESP8266 chip, NodeMCU firmware comes with ESP8266 Development board/kit i.e. NodeMCU Development board.

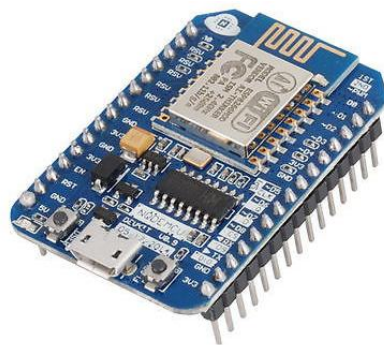


Fig. 6.1: NodeMCU Development board

6.2 NODEMCU DEVELOPMENT BOARD/KIT V0.9 (VERSION1)

Since NodeMCU is open source platform, their hardware design is open for edit/modify/build.

NodeMCU Dev Kit/board consist of ESP8266 wifi enabled chip. The **ESP8266** is a low-cost Wi-Fi chip developed by Espressif Systems with TCP/IP protocol. For more information about ESP8266, you can refer ESP8266 WiFi Module.

There is Version2 (V2) available for NodeMCU Dev Kit i.e. **NodeMCU Development Board v1.0 (Version2)**, which usually comes in black colored PCB.



Fig. 6.2: NodeMCU Development Board/kit v1.0 (Version2)

For more information about NodeMCU Boards available in market refer NodeMCU Development Boards.

NodeMCU Dev Kit has **Arduino like** Analog (i.e. A0) and Digital (D0-D8) pins on its board.

It supports serial communication protocols i.e. UART, SPI, I2C etc.

Using such serial protocols we can connect it with serial devices like I2C enabled LCD display, Magnetometer HMC5883, MPU-6050 Gyro meter + Accelerometer, RTC chips, GPS modules, touch screen displays, SD cards etc.

How to start with NodeMCU?

NodeMCU Development board is featured with wifi capability, analog pin, digital pins and serial communication protocols.

To get start with using NodeMCU for IoT applications first we need to know about how to write/download NodeMCU firmware in NodeMCU Development Boards. And before that where this NodeMCU firmware will get as per our requirement.

There is online NodeMCU custom builds available using which we can easily get our custom NodeMCU firmware as per our requirement.

To know more about how to build custom NodeMCU firmware online and download it refer Getting started with NodeMCU

How to write codes for NodeMCU?

After setting up ESP8266 with Node-MCU firmware, let's see the IDE (Integrated Development Environment) required for development of NodeMCU.

NodeMCU with ESPlorer IDE

Lua scripts are generally used to code the NodeMCU. Lua is an open source, lightweight, embeddable scripting language built on top of C programming language.

For more information about how to write Lua script for NodeMCU refer Getting started with NodeMCU using ESPlorerIDE

NodeMCU with Arduino IDE

Here is another way of developing NodeMCU with a well-known IDE i.e. Arduino IDE. We can also develop applications on NodeMCU using Arduino development environment. This makes easy for Arduino developers than learning new language and IDE for NodeMCU.

For more information about how to write Arduino sketch for NodeMCU refer Getting started with NodeMCU using ArduinoIDE

Difference in using ESPlorer and Arduino IDE

Well, there is a programming language difference we can say while developing application for NodeMCU using ESPlorer IDE and Arduino IDE.

We need to code in C\C++ programming language if we are using Arduino IDE for developing NodeMCU applications and Lua language if we are using ESPlorer IDE.

Basically, NodeMCU is Lua Interpreter, so it can understand Lua script easily. When we write Lua scripts for NodeMCU and send/upload it to NodeMCU, then they will get executes sequentially. It will not build binary firmware file of code for NodeMCU to write. It will send Lua script as it is to NodeMCU to get execute.

In Arduino IDE when we write and compile code, ESP8266 toolchain in background creates binary firmware file of code we wrote. And when we upload it to NodeMCU then it will flash all NodeMCU firmware with newly generated binary firmware code. In fact, it writes the complete firmware.

That's the reason why NodeMCU not accept further Lua scripts/code after it is getting flashed by Arduino IDE. After getting flashed by Arduino sketch/code it will be no more Lua interpreter and we got error if we try to upload Lua scripts. To again start with Lua script, we need to flash it with NodeMCU firmware.

Since Arduino IDE compile and upload/writes complete firmware, it takes more time than ESPlorer IDE.

A Brief Note on Flex Sensor

We have seen several **Types of Sensors** like Light, Temperature, Humidity, Proximity, Infrared, Heartbeat, Tilt, Flame etc. in various Electronics Hub projects. But there is one sensor which I haven't discussed or brought up until now. It is the Flex Sensor.

What is a Flex Sensor?

A Flex Sensor or sometimes called as Bend Sensor is a device that measures the amount of bend or angular deflection. Usually, a Flex Sensor is made up of a variable resistive surface and the amount of resistance is varied by bending the sensor.

The Flex Sensor used in this project is shown in the following image. It is about 0.6cm wide and 8cm long (including the two connectors).

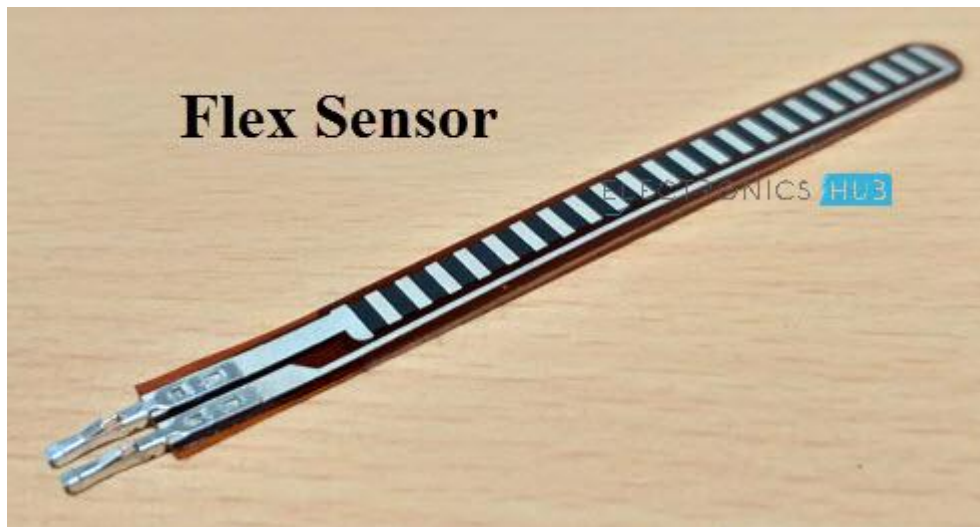


Fig. 6.3: Flex Sensor

Since a flex sensor is basically a resistor (whose resistance varies depending on its bend), it has two terminals (or leads).

Coming to the variable resistance part, an unflexed sensor i.e. a flex sensor as rest, exhibits a normal resistance value. In my case, the normal resistance of the Flex Sensor is around 60K Ω .

When I bend the sensor as shown in the following image, the resistance increases with the increase in the bend angle.



Fig. 6.4: Bend angle

At a bend angle of approximately 45° , the resistance of the flex sensor has increased to $65K\Omega$ and when I continue to bend it at an angle of 90° , the resistance further increases to $70K\Omega$. If I further bend it towards a complete 180° angle, the resistance of the flex sensor shoots up to around $75K\Omega$.

NOTE: The resistance values mentioned here are specific to the Flex Sensor which I am using and they are not universal.

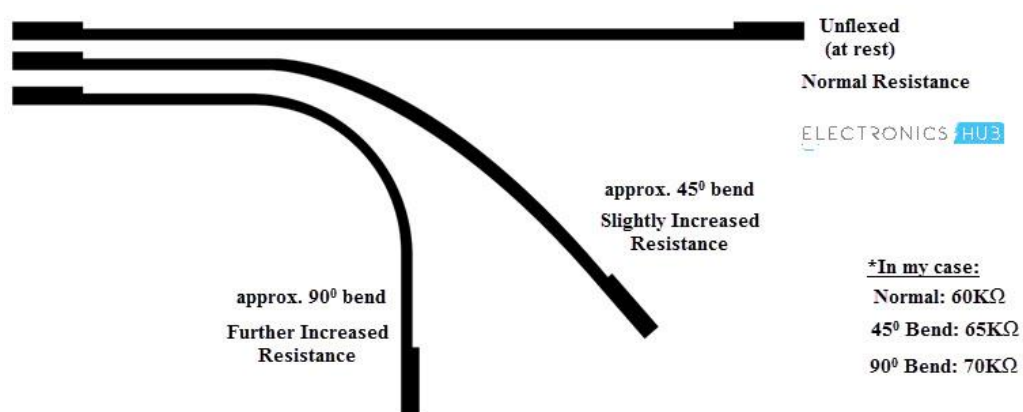


Fig. 6.5: Types of Flex Sensors

Depending on the type of resistive material used on the surface of the Flex Sensor, there can be different types of Flex Sensors. Some of them are:

- Carbon Resistive Flex Sensor
- Capacitive Flex Sensor
- Fiber optic Flex Sensor
- Conductive ink Flex Sensor

The Carbon Resistive Type Flex Sensor, which is sometimes called as Velostat Flex Sensor, is the commonly used Flex Sensor by DIYers and hobbyists.

6.3 BASIC FLEX SENSOR CIRCUIT

Since the underlying concept of Flex Sensor is a variable resistor, you can easily guess the Basic Flex Sensor Circuit that can be implemented. It is a simple voltage divider circuit.

The following image shows the basic flex sensor circuit consisting of a voltage divider (formed by the flex sensor itself and a 10KΩ Resistor) and an Operational Amplifier (Op-Amp).

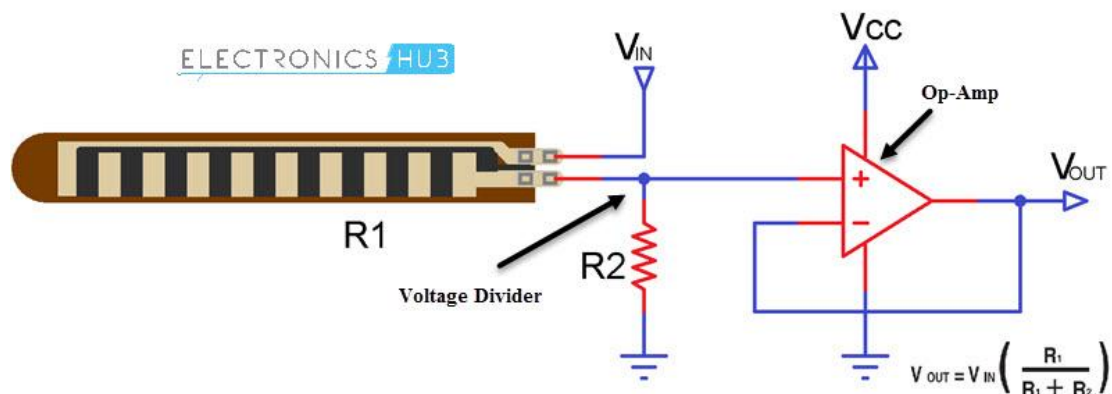


Fig. 6.6: Basic Flex Sensor Circuit

Interfacing a Flex Sensor with Arduino

Now that we have seen a little bit about what a flex sensor is and how a flex sensor works, we will now proceed to interfacing a flex sensor with Arduino UNO board.

I will design two simple circuits using Flex Sensor and Arduino, wherein the first circuit is just a basic hook-up guide of Flex Sensor with Arduino.

In the second circuit, using a similar Arduino Flex Sensor Interface, I will control a TowerPro SG90 Servo Motor. So, lets get started.

Components Required

- Arduino UNO
- Flex Sensor
- TowerPro SG90 Servo Motor
- 10K Ω Resistor
- LED
- 1K Ω Resistor
- Mini Breadboard
- Connecting wires

Arduino Flex Sensor Hook-up Guide

The following image shows the first of two circuits I am going to implement. It is a simple Arduino Flex Sensor Hook-up circuit.

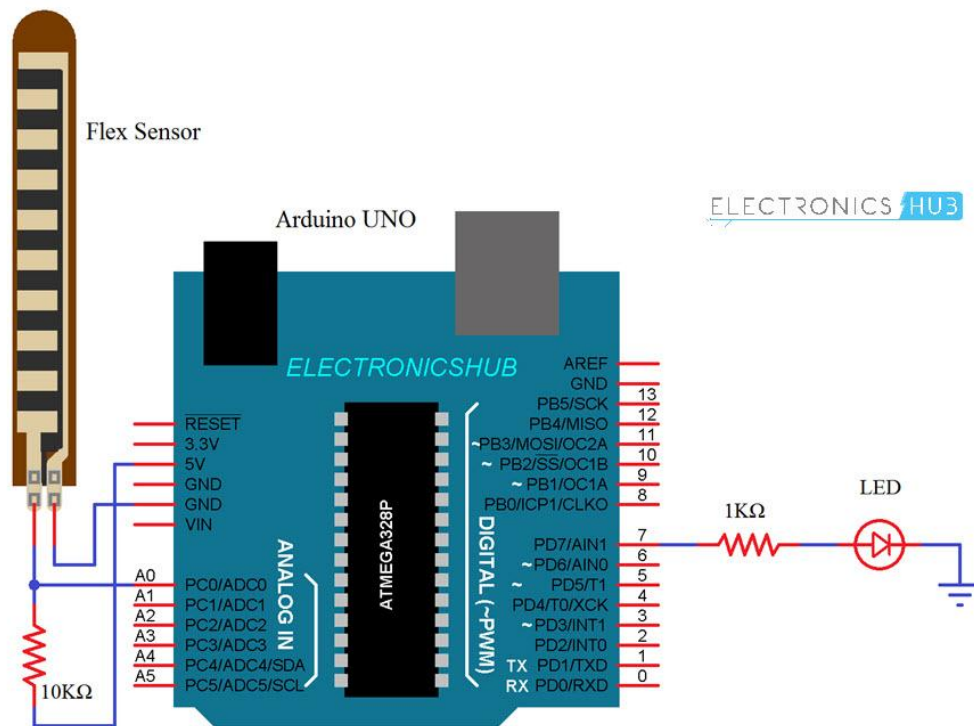


Fig. 6.7: Arduino Flex Sensor Hook-up Guide

Circuit Design

As I have already mentioned in the Basic Flex Sensor Circuit part that a flex sensor is nothing but a variable resistor and it can be configured in a voltage divider fashion along with another resistor to measure the analog voltage.

One end of flex sensor is connected to GND while the other end is connected to Analog Input A0 of Arduino. A 10KΩ resistor is connected between A0 and +5V. This connection means that the flex sensor and the 10KΩ resistor form a voltage divider.

Just to show some output, I have connected an LED (along with its current limiting resistor of course) to Digital I/O Pin 7 of Arduino UNO.

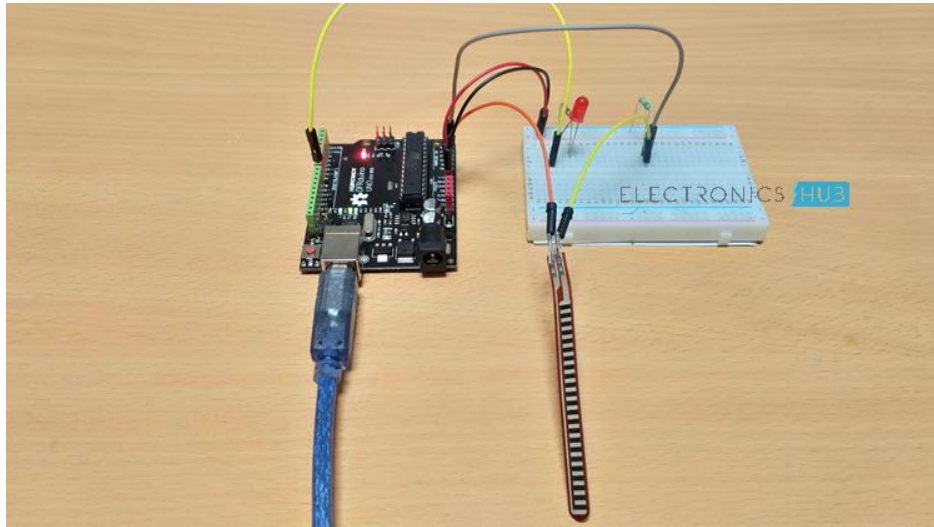


Fig. 6.8: Circuit Design

The following code is used to read the analog data from the Flex Sensor and display it on the serial monitor of Arduino. Additionally, if the value exceeds a certain value, the LED connected to Pin 7 will be turned ON.

```
const int flexPin = A0;  
const int ledPin = 7;
```

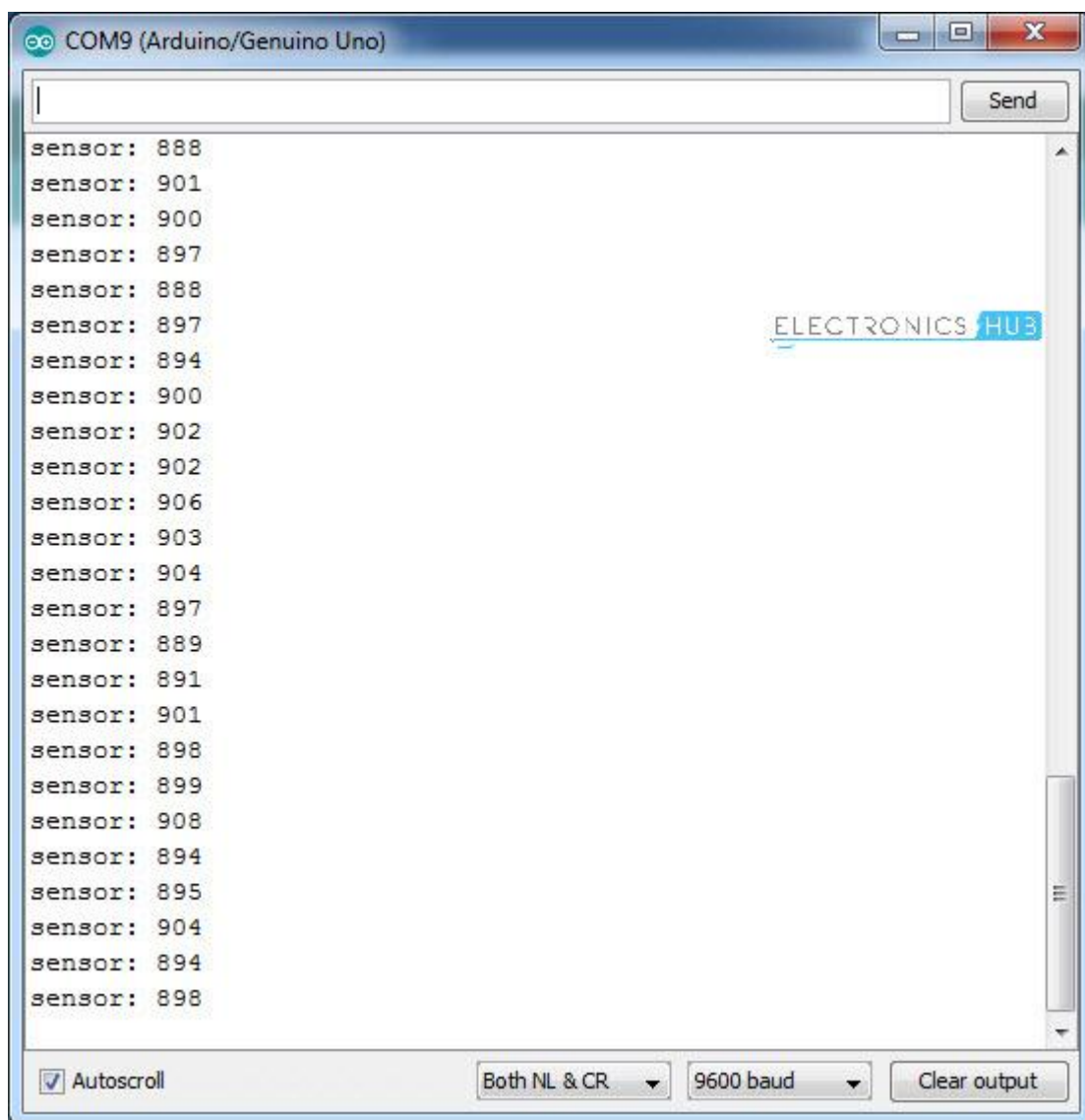
```
void setup()  
{  
  Serial.begin(9600);  
  pinMode(ledPin,OUTPUT);  
}
```

```
void loop()  
{  
  int flexValue;  
  flexValue = analogRead(flexPin);  
  Serial.print("sensor: ");  
  Serial.println(flexValue);
```

```
if(flexValue>890)
  digitalWrite(ledPin,HIGH);
else
  digitalWrite(ledPin,LOW);

delay(20);
}
```

view rawArduino_Flex_Sensor_LED.ino hosted with ♥ by GitHub



6.4 CONTROL A SERVO USING ARDUINO AND FLEX SENSOR

In the second circuit, which can be considered an extension to the first one, I will control a TowerPro SG90 Servo motor using the data obtained after interfacing Flex Sensor with Arduino.

The following image shows the circuit diagram of Servo Motor control using flex sensor and Arduino

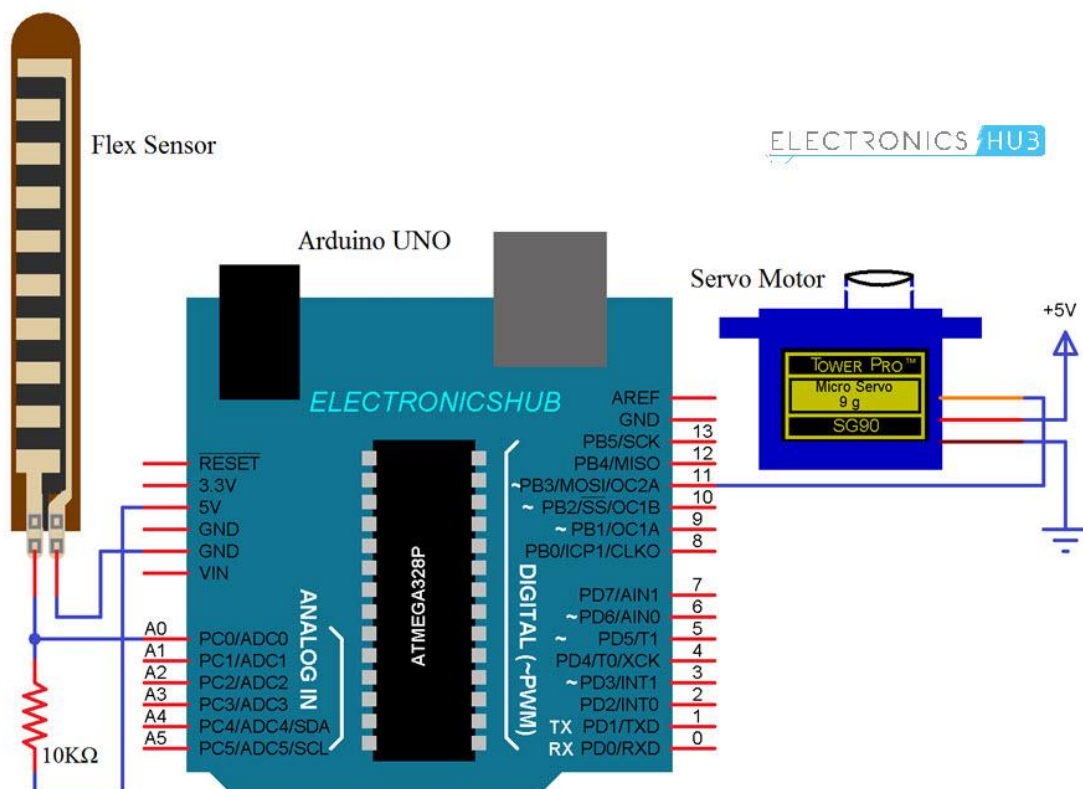


Fig. 6.9: Circuit diagram of Servo Motor control

6.4.1 Circuit Design

The Flex Sensor part of the circuit is same as the earlier circuit. Additionally, the servo motor (or rather its control pin – orange or yellow wire) is connected to Digital I/O pin 11 of Arduino UNO.

Code

The code for controlling a servo motor using flex sensor and Arduino is given below. Using this code, depending on the bend of the flex sensor, the servo motor will change its angle.

```
const
int
flexPin
= A0;

const int ledPin = 7;

void setup()

{

  Serial.begin(9600);

  pinMode(ledPin,OUTPUT);

}

void loop()

{

  int flexValue;

  flexValue = analogRead(flexPin);

  Serial.print("sensor: ");

  Serial.println(flexValue);
```

```
if(flexValue>890)
    digitalWrite(ledPin,HIGH);

else

    digitalWrite(ledPin,LOW);

delay(20);

}
```

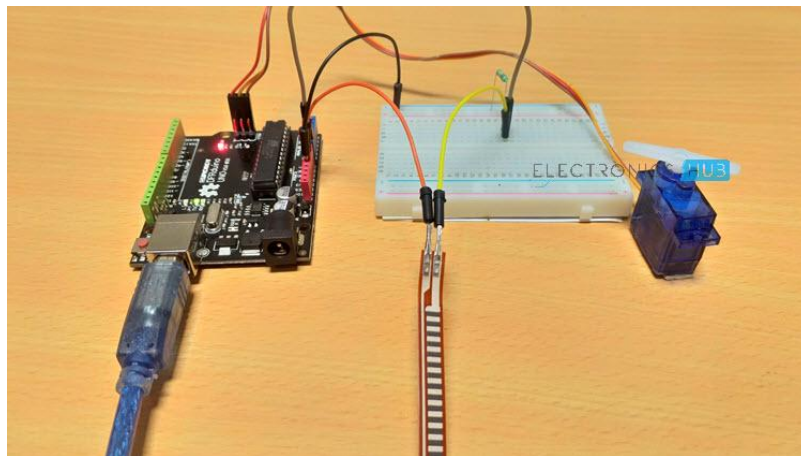


Fig. 6.10: Applications of Flex Sensor

Flex Sensors can be implemented in a variety of shapes and sizes. They can be used in applications where accurate measurement of motion or angle of a device is to be measured.

Some of the applications of Flex Sensor are mentioned below:

- Angular Displacement Measurement
- Robotics
- Automotive
- Medical Instruments
- Industrial Controls
- VR Equipment (VR – Virtual Reality)

- Computer Peripherals
- Motion Control
- Physical Therapy
- Musical Instruments
- Fitness Devices

6.5 HEARTBEAT SENSOR

A person's heartbeat is the sound of the valves in his/her's 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.

6.5.1 *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.

6.5.2 *Principle of Heartbeat Sensor*

The heartbeat sensor is based on the principle of photo plethysmography. 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.

There are two types of photoplethysmography:

Transmission: Light emitted from the light emitting device is transmitted through any vascular region of the body like earlobe and received by the detector.

Reflection: Light emitted from the light emitting device is reflected by the regions.

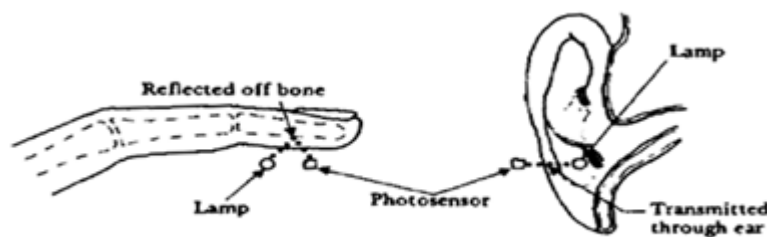
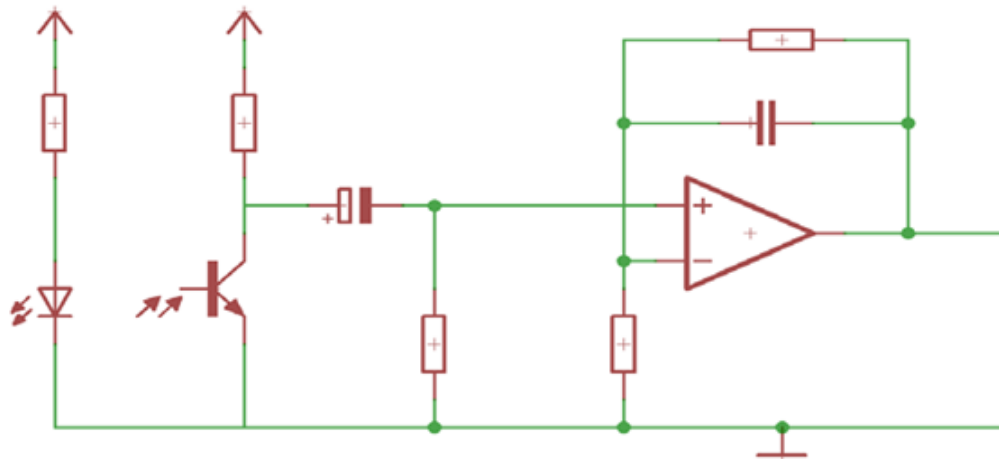


Fig. 6.11: Working of a Heartbeat Sensor

The basic heartbeat sensor consists of a light emitting diode and a detector like a light detecting resistor or a photodiode. The heart beat pulses causes 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 form of 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 from 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 heart beat rate, given by the formula-

$$\text{BPM(Beats per minute)} = 60 \cdot f$$

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 Idr embedded onto a clip like structure. The clip is attached to the organ (earlobe or the finger) with the detector part on the flesh.



Fig. 6.12: Heart Rate Sensor

Another example is **TCRT1000**, having 4 pins-

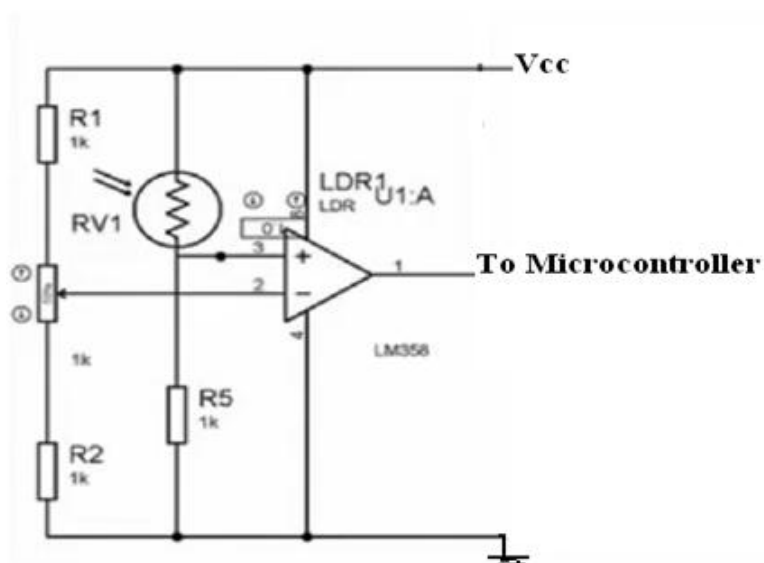
Pin1: To give supply voltage to the LED

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 the wrist and the output can be sent wirelessly (through Bluetooth) to the computer for processing.



Fig. 6.13: 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



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.

Here a comparator is used which compares the output voltage from the LDR to that of the threshold voltage. The threshold voltage is the voltage drop across the LDR when the light with fixed intensity, from the light source falls directly on it. The inverting terminal of the comparator LM358 is connected to the potential divider arrangement which is set to the threshold voltage and the non-inverting terminal is connected to the LDR. When a human tissue is illuminated using the light source, the intensity of the light reduces. As this reduced light intensity falls on the LDR, the resistance increases and as a result the voltage drop increases. When the voltage drop across the LDR or the non-inverting input exceeds that of the inverting input, a logic high signal is developed at the output of the comparator and in case voltage drop being lesser a logic low output is developed. Thus the output is a series of pulses. These pulses can be fed to the Microcontroller which accordingly processes the information to get the heart beat rate and this is displayed on the Display interfaced to the Microcontroller.

6.6 TEMPERATURE SENSOR

Temperature is the most often-measured environmental quantity. This might be expected since most physical, electronic, chemical, mechanical, and biological systems are affected by temperature. Certain chemical reactions, biological processes, and even electronic circuits perform best within limited temperature ranges. Temperature is one of the most commonly measured variables and it is therefore not surprising that there are many ways of sensing it. Temperature sensing can be done either through direct contact with the heating source, or remotely, without direct contact with the source using radiated energy instead. There

are a wide variety of temperature sensors on the market today, including Thermocouples, Resistance Temperature Detectors (RTDs), Thermistors, Infrared, and Semiconductor Sensors.

6.7 THERMOCOUPLE

It is a type of temperature sensor, which is made by joining two dissimilar metals at one end. The joined end is referred to as the HOT JUNCTION. The other end of these dissimilar metals is referred to as the COLD END or COLD JUNCTION. The cold junction is actually formed at the last point of thermocouple material. If there is a difference in temperature between the hot junction and cold junction, a small voltage is created. This voltage is referred to as an EMF (electro-motive force) and can be measured and in turn used to indicate temperature.

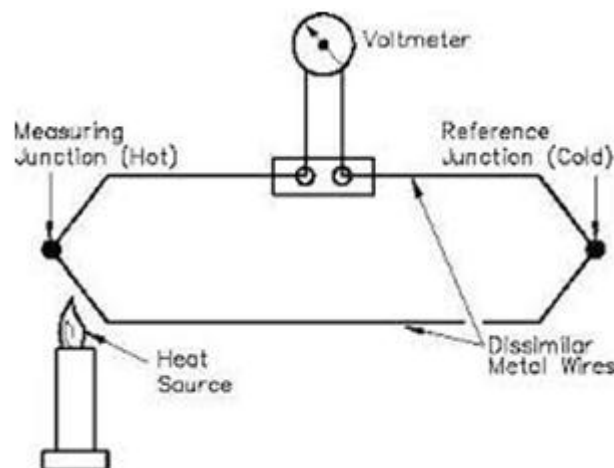


Fig. 6.14: Thermocouple

The RTD is a temperature sensing device whose resistance changes with temperature. Typically built from platinum, though devices made from nickel or copper are not uncommon, RTDs can take many different shapes like wire wound, thin film. To measure the resistance across an RTD, apply a constant current, measure the resulting voltage, and determine the RTD resistance. RTDs exhibit fairly linear resistance to temperature curves over their operating regions, and any nonlinearity are highly predictable and repeatable. The PT100 RTD evaluation board uses surface mount RTD to measure temperature. An external 2, 3 or 4-wire PT100 can also be associated with measure temperature in remote areas. The

RTDs are biased using a constant current source. So as to reduce self-heat due to power dissipation, the current magnitude is moderately low. The circuit shown in figure is the constant current source uses a reference voltage, one amplifier, and a PNP transistor.

Thermistors

Similar to the RTD, the thermistor is a temperature sensing device whose resistance changes with temperature. Thermistors, however, are made from semiconductor materials. Resistance is determined in the same manner as the RTD, but thermistors exhibit a highly nonlinear resistance vs. temperature curve. Thus, in the thermistors operating range we can see a large resistance change for a very small temperature change. This makes for a highly sensitive device, ideal for set-point applications.

Semiconductor sensors

They are classified into different types like Voltage output, Current output, Digital output, Resistance output silicon and Diode temperature sensors. Modern semiconductor temperature sensors offer high accuracy and high linearity over an operating range of about 55°C to +150°C. Internal amplifiers can scale the output to convenient values, such as 10mV/°C. They are also useful in cold-junction compensation circuits for wide temperature range thermocouples. A brief details about this type of temperature sensor are given below.

6.8 SENSOR ICS

There are a wide variety of temperature sensor ICs that are available to simplify the broadest possible range of temperature monitoring challenges. These silicon temperature sensors differ significantly from the above mentioned types in a couple of important ways. The first is operating temperature range. A temperature sensor IC can operate over the nominal IC temperature range of -55°C to +150°C. The second major difference is functionality.

A silicon temperature sensor is an integrated circuit, and can therefore include extensive signal processing circuitry within the same package as the sensor. There is no need to add compensation circuits for temperature sensor ICs. Some of these are analogue circuits with either voltage or current output. Others combine analogue-sensing circuits with voltage comparators to provide alert functions. Some other sensor ICs combine analogue-sensing circuitry with digital input/output and control registers, making them an ideal solution for microprocessor-based systems.

Digital output sensor usually contains a temperature sensor, analog-to-digital converter (ADC), a two-wire digital interface and registers for controlling the IC's operation. Temperature is continuously measured and can be read at any time. If desired, the host processor can instruct the sensor to monitor temperature and take an output pin high (or low) if temperature exceeds a programmed limit. Lower threshold temperature can also be programmed and the host can be notified when temperature has dropped below this threshold. Thus, digital output sensor can be used for reliable temperature monitoring in microprocessor-based systems.

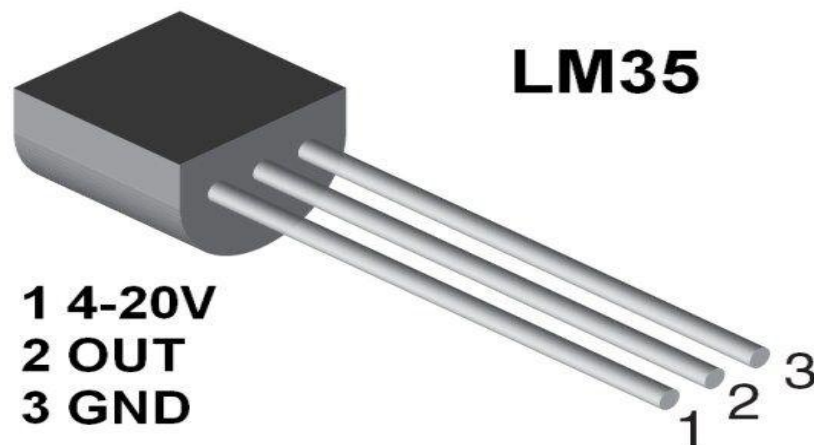


Fig. 6.15: Temperature sensor

Above temperature sensor has three terminals and required Maximum of 5.5 V supply. This type of sensor consists of a material that performs the operation according to temperature to vary the resistance. This change of resistance is sensed by circuit and it calculates temperature. When the voltage increases then the temperature also rises. We can see this operation by using a diode.

Temperature sensors directly connected to microprocessor input and thus capable of direct and reliable communication with microprocessors. The sensor unit can communicate effectively with low-cost processors without the need of A/D converters.

An example for a temperature sensor is **LM35**. The LM35 series are precision integrated-circuit temperature sensors, whose output voltage is linearly proportional to the Celsius temperature. The LM35 is operates at -55° to +120°C. The basic centigrade temperature sensor (+2°C to +150°C) is shown in figure below.

Features of LM35 Temperature Sensor

- Calibrated directly in ° Celsius (Centigrade)
- Rated for full I -55° to +150°C range
- Suitable for remote applications
- Low cost due to wafer-level trimming
- Operates from 4 to 30 volts
- Low self-heating,
- $\pm 1/4^\circ\text{C}$ of typical nonlinearity

Operation of LM35

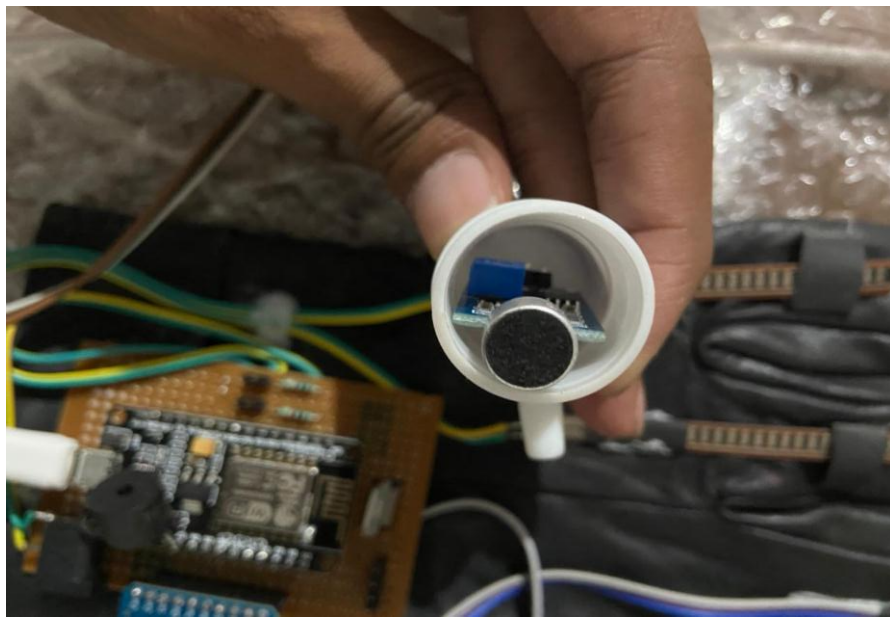
- The LM35 can be connected easily in the same way as other integrated circuit temperature sensors. It can be stuck or established to a surface and its temperature will be within around the range of 0.01°C of the surface temperature.
- This presumes that the ambient air temperature is just about the same as the surface temperature; if the air temperature were much higher or lower than the surface temperature, the actual temperature of the LM35 die would be at an intermediate temperature between the surface temperature and the air temperature.

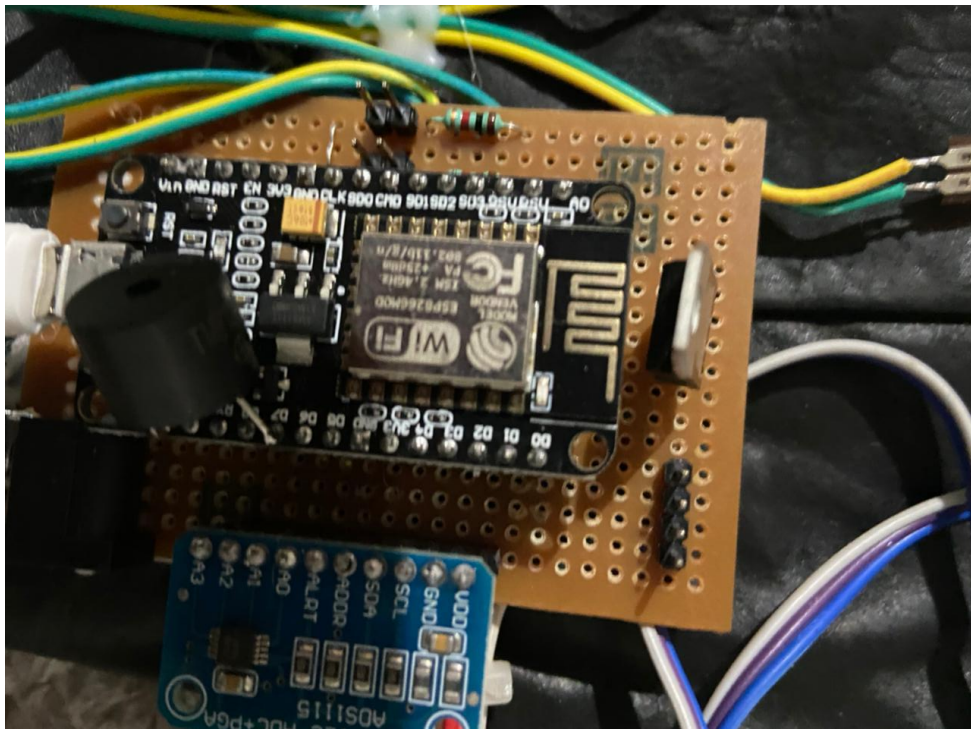
The temperature sensors have well known applications in environmental and process control and also in test, measurement and communications. A digital temperature is a sensor, which provides 9-bit temperature readings. Digital temperature sensors offer excellent precise accuracy, these are designed to read from 0°C to 70°C and it is possible to achieve $\pm 0.5^{\circ}\text{C}$ accuracy. These sensors completely aligned with digital temperature readings in degree Celsius.

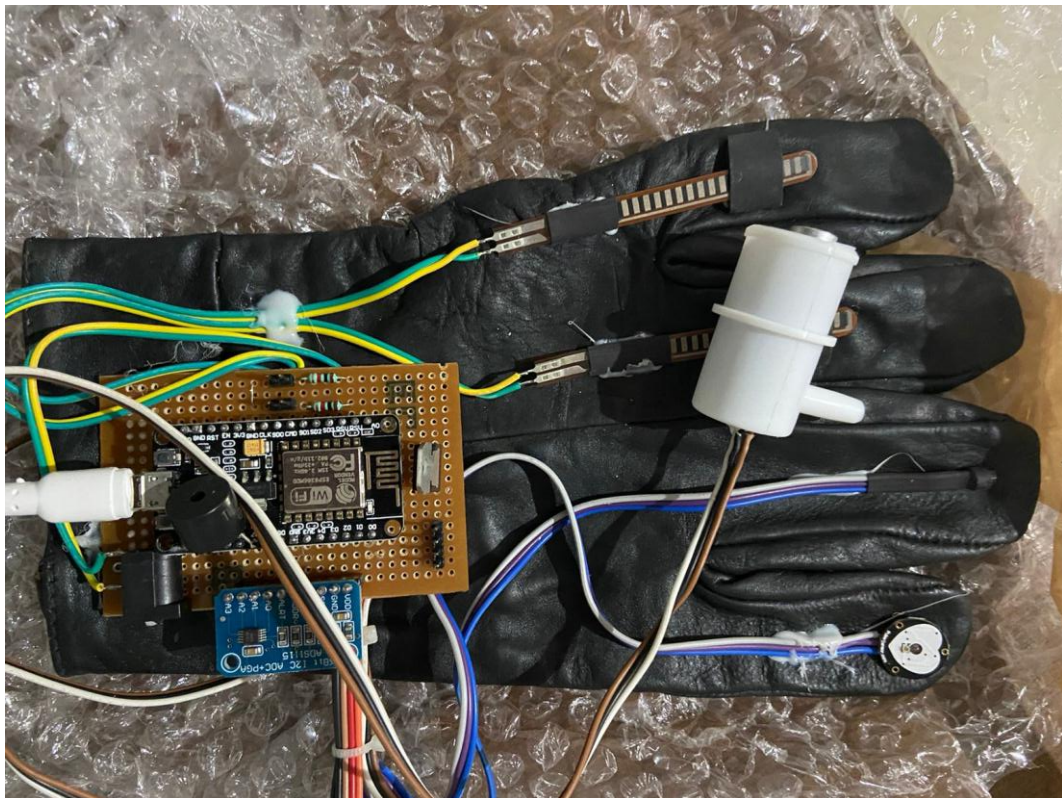
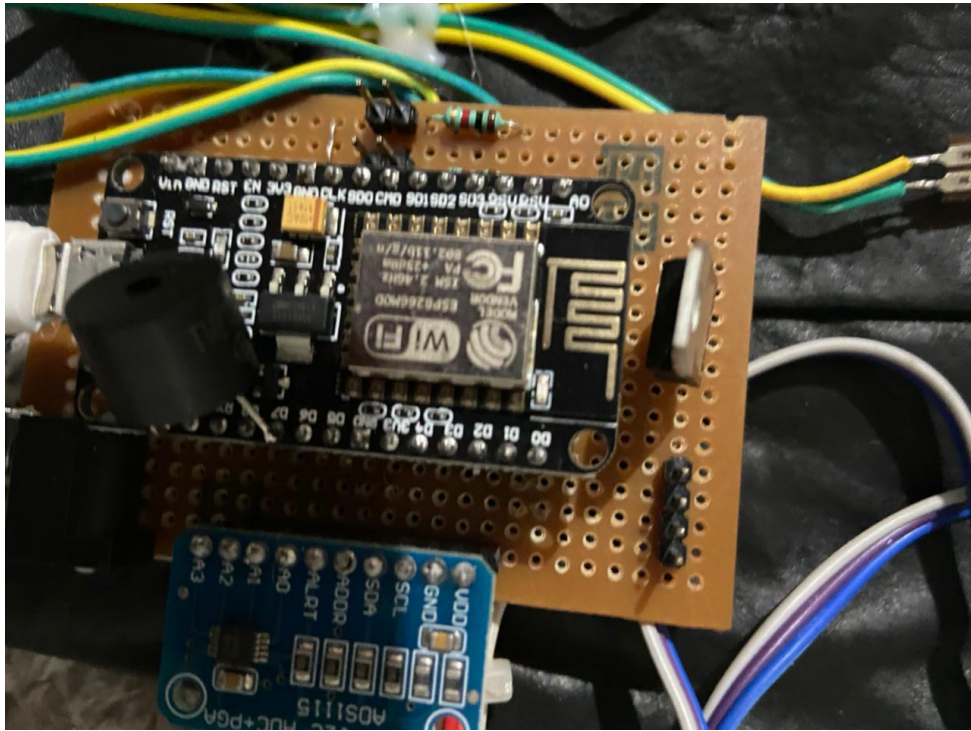
CHAPTER 7

CODE AND OUTPUT RESULTS

7.1 SETUP







7.2 CODE

```
030_paralysis_health | Arduino 1.8.19 (Windows Store 1.8.57.0)
File Edit Sketch Tools Help

030_paralysis_health
#include <Adafruit_ADS1X15.h>
#include "Ubidots.h"

/*****
 * Define Instances and Constants
 *****/

const char* UBIDOTS_TOKEN = "8BFF-CLMHAH8eRl6dqk77V7F7a7j9G49yk"; // Put here your Ubidots TOKEN
const char* WIFI_SSID = "paralysis"; // Put here your Wi-Fi SSID
const char* WIFI_PASS = "12345678"; // Put here your Wi-Fi password
Ubidots ubidots(UBIDOTS_TOKEN, UBI_HTTP);
Adafruit_ADS1015 ads; // Use this for the 12-bit version */

void setup(void)
{
  Serial.begin(9600);
  pinMode(D0, INPUT);
  pinMode(D7, OUTPUT);
  digitalWrite(D7, LOW);
  ubidots.wifiConnect(WIFI_SSID, WIFI_PASS);
  Serial.println("Hello!");

  Serial.println("Getting single-ended readings from AIN0..3");
  Serial.println("ADC Range: +/- 6.144V (1 bit = 3mV/ADS1015, 0.1875mV/ADS1115)");

  if (!ads.begin()) {
    Serial.println("Failed to initialize ADS.");
    while (1);
  }
}

void loop(void)
{
  int resp = digitalRead(D0);
  int16_t adc0, adc1, adc2, adc3;
  int i=0;
  int j=0;

  // ... (rest of the code) ...

  // Could not write preferences file: C:\Users\BOU\Documents\Arduino\data\preferences.txt (Access is denied)
  // Error downloading https://downloads.arduino.cc/packages/package_index.json
}
```

```
030_paralysis_health | Arduino 1.8.19 (Windows Store 1.8.57.0)
File Edit Sketch Tools Help

030_paralysis_health
ads0 = ads.readADC_SingleEnded(2);
ads3 = ads.readADC_SingleEnded(3);

float temperature = ads.computeVolts(ads0)*100;
if (ads2 < 30)
{
  alert();
  i++;
}
if (ads3 < 30)
{
  alert();
  j++;
}

Serial.println("-----");
Serial.print("AIN0: "); Serial.print(temperature); Serial.println(" ");
Serial.print("AIN1: "); Serial.print(adc1); Serial.println(" ");
Serial.print("AIN2: "); Serial.print(adc2); Serial.println(" ");
Serial.print("AIN3: "); Serial.print(adc3); Serial.println(" ");

ubidots.add("temperature", temperature);
ubidots.add("Pulse", random(78,88));
ubidots.add("Finger_1", i);
ubidots.add("Finger_2", j);
ubidots.add("respiratory", resp);
bool bufferSent = false;
bufferSent = ubidots.send(); // Will send data to a device label that matches the device id

if (bufferSent) {
  // Do something if values were sent properly
  Serial.println("Values sent by the device");
}
delay(1000);
}

void alert()
{
  // ... (rest of the code) ...

  // Could not write preferences file: C:\Users\BOU\Documents\Arduino\data\preferences.txt (Access is denied)
  // Error downloading https://downloads.arduino.cc/packages/package_index.json
}
```

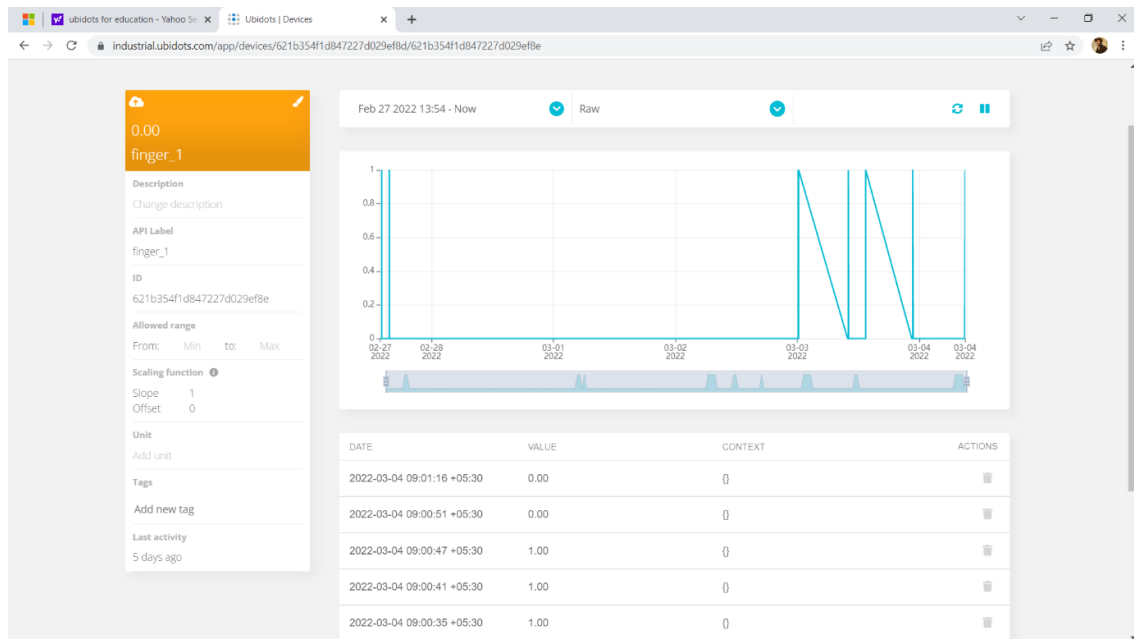



Fig. 7.2: Finger_1



Fig. 7.3: Finger_1 alert message

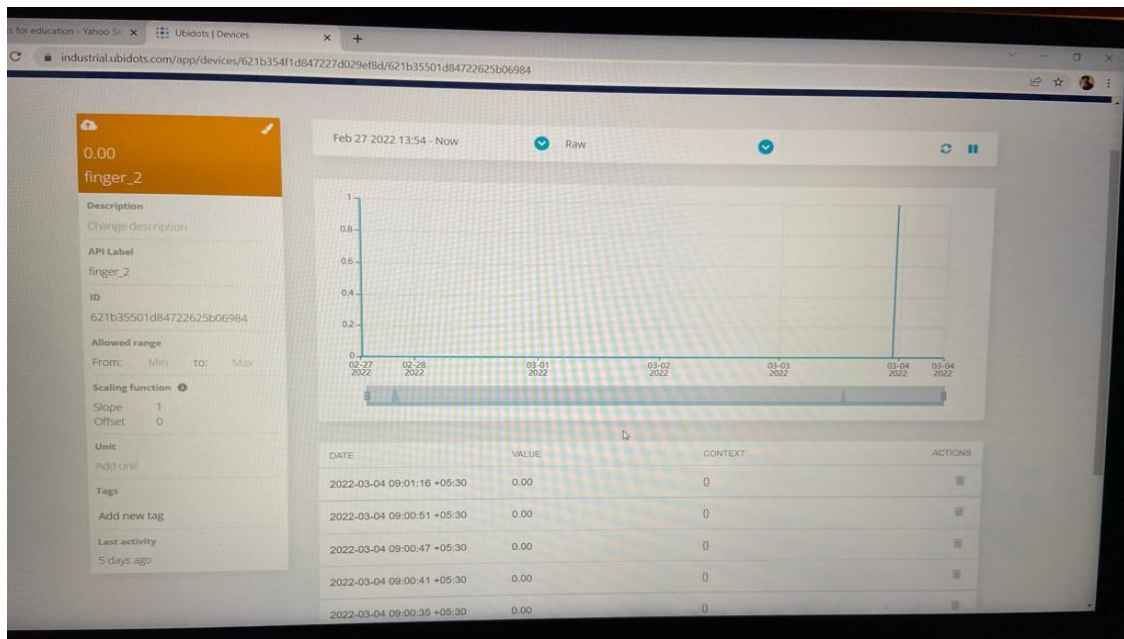


Fig.7.4: Finger_2

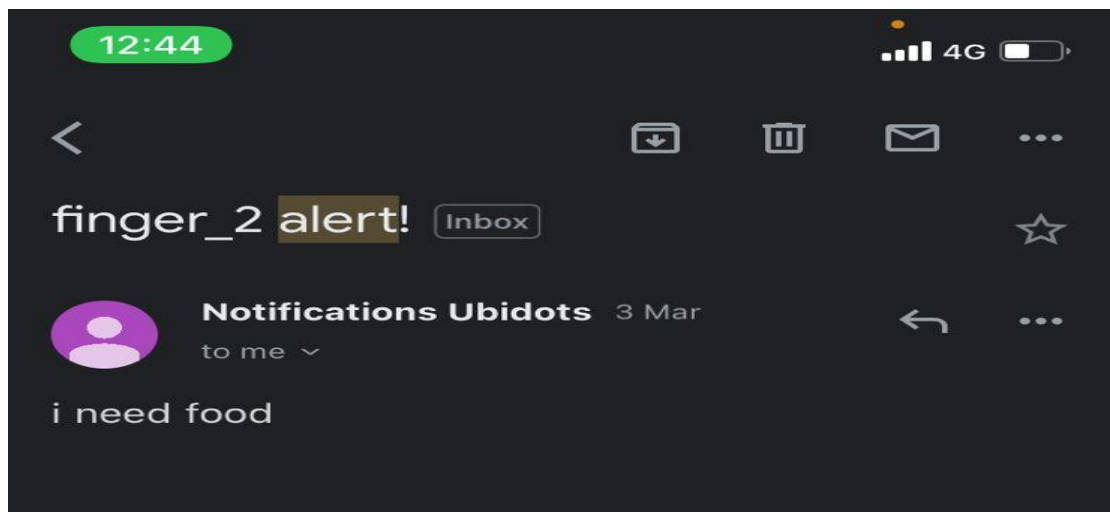


Fig. 7.5: Finger_2 alert message

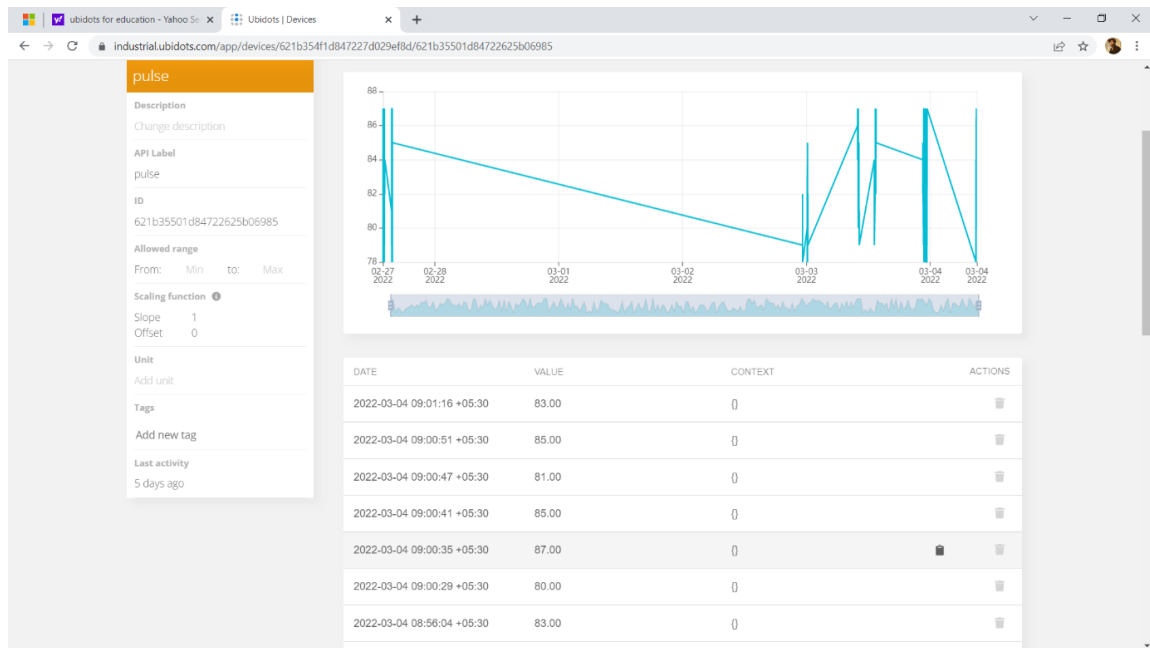


Fig. 7.6: Pulse Sensor

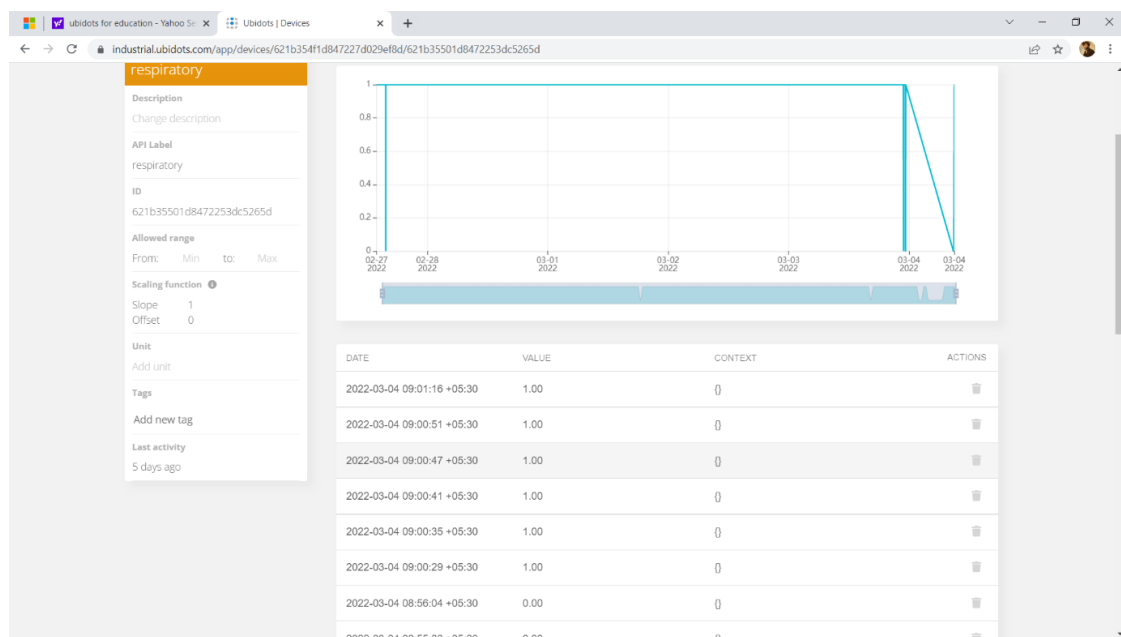


Fig. 7.7: Respiratory sensor

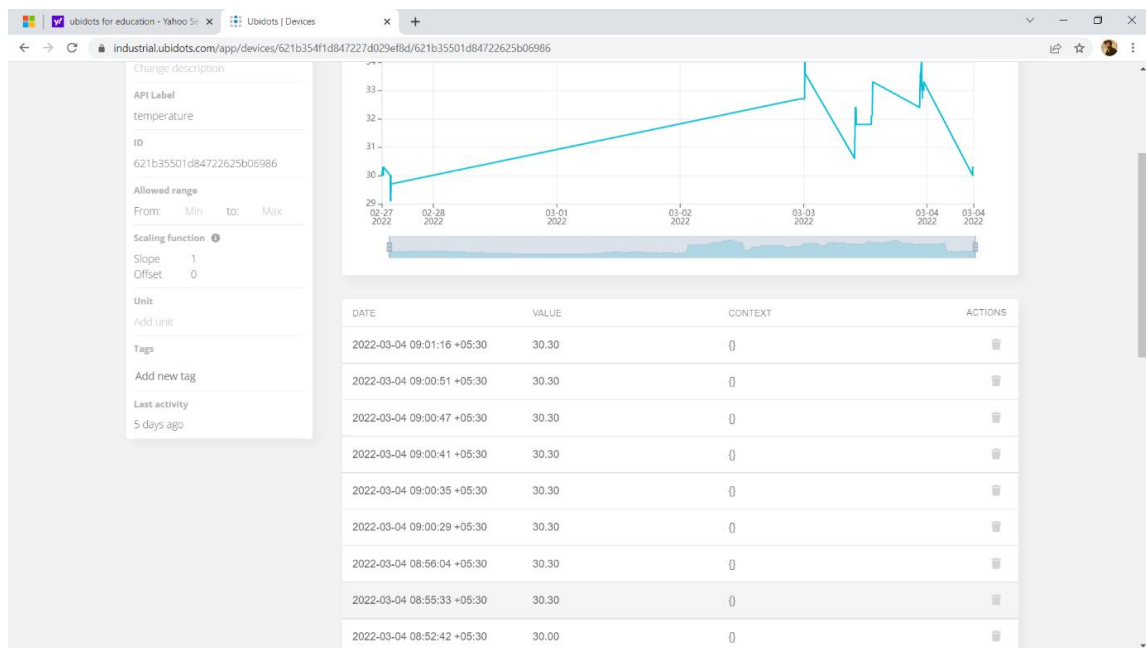


Fig. 7.8: Temperature sensor

CHAPTER 8

CONCLUSION

This system is really helpful for paralysed patients. When they need help then they can ask by using some movements they can also survive in this world like normal people by using this movement detection. This system is reliable and cheap and less weight.

So they can buy without debt. This system will make paralysed patients to achieve a independent of mobility. This is not a trivial task just because it varies from person to person in its nature and type. Therefore, different methods are essential to support these people, and it is our duty, as future engineers, to develop new technologies to help paralysed patients.

REFERENCES

- [1] Abhijeet Botre et al., "Assistance system for paralyzed", International Journal Of Innovative Research In Electrical, Electronics, Instrumentation And Control Engineering ,Vol 4, Issue 5, 2016.
- [2] Alexis Bell, Paul Rogers, Chris Farnell, Brett Sparkman, ScottC Smith, "Wireless Patient Monitoring System", HIC 2014 IEEE .
- [3] Alyson Matheus de Carvalho Souza et al., "Hand copter Game", Virtual and Augmented Reality (SVR), 2012.
- [4] Gomez-Vilda et al., "Vocal-fold paralysis patients treated with stem-cellgrafting", International Conference on Pattern Recognition Systems (ICPRS-16), 2016.
- [5] Hemakshi Pawar et al., "Assistive Interactive Device using Electro-Oculography", International Journal of Advanced Research in Computer Engineering and Technology ,Vol 4, Issue 1, 2015.
- [6] Herta Flor et al., "The Thought Translation Device (TTD) for Completely Paralyzed Patients", IEEE Transactions on Rehabilitation Engineering, Vol 8, No 2, 2000.
- [7] Massage Coll et al., "Treatment of Peripheral Facial Paralysis (PFP)", Photonics and Optoelectronics (SOPO), 2012.
- [8] Rolga Roy, "Methodologies to assist paralysed patients", International Journal Of Advanced Research In Electrical, Electronics And Instrumentation Engineering, Vol 5, Issue 3, 2016.