

# SMART HEALTHCARE MONITOR USING IOT

<sup>1</sup>SK. MD. Mansoor Ali, <sup>2</sup>CH.Rohan Rao, <sup>3</sup>U. Satish <sup>4</sup>Mrs. M. Revathi

<sup>1,2,3</sup>Students, <sup>4</sup>Guide

Department of Computer Science and Engineering  
Bharath Institute of Higher Education and Research

**Abstract-** Healthcare monitoring system in hospitals and many other health centers has experienced significant growth, and portable healthcare monitoring systems with emerging technologies are becoming of great concern to many countries worldwide nowadays. The advent of Internet of Things (IoT) technologies facilitates the progress of healthcare from face-to-face consulting to telemedicine. This paper proposes a smart healthcare system in IoT environment that can monitor a patient's basic health signs as well as the room condition where the patients are now in real-time. In this system, five sensors are used to capture the data from hospital environment named heart beat sensor, body temperature sensor, room temperature sensor. The error percentage of the developed scheme is within a certain limit (<5%) for each case. The condition of the patients is conveyed via a portal to medical staff, where they can process and analyze the current situation of the patients. The developed prototype is well suited for healthcare monitoring that is proved by the effectiveness of the system.

## Objective

One of the goals of smart healthcare is to help users by educating them about their medical status and keeping them health-aware. Smart healthcare empowers users to self-manage some emergency situations. It provides an emphasis on improving the quality and experience of the user.

## Introduction

Now a days, the internet has become a vital part of our daily life. It has changed how people live, work, play and learn. Internet serves for numerous ideas such as education, science, industries, entertainment, social networking, shopping, ecommerce etc. The next innovative mega trend of Internet is Internet of Things (IoT). The IoT connects smart objects to the Internet. It can facilitate an exchange of data and bring users processed data in a more reliable and secured way. The Internet of Things (IoT) is one of the most vital and transformative technologies ever invented. The Internet of Things (IoT) is a megatrend in next-generation technologies that can culminate the complete business gamut and can be thought of as the interconnection of uniquely identifiable smart devices within today's internet infrastructure with extended benefits. These benefits basically include the advanced concatenation of the devices, systems, and services that go beyond machine-to-machine (M2M) scenarios. Therefore, initiating automation is feasible in nearly every domain.

The increased use of mobile technologies and smart devices in the area of health has caused great impact on the world. Health experts are increasingly taking advantage of the benefits these technologies bring, thus generating a significant improvement in health care in clinical settings. Likewise, countless ordinary users are being served from the advantages of the M-Health (Mobile Health) applications and E-Health (health care supported by ICT) to improve, help and assist their health.

According to the constitutions of World Health Organization (WHO) the highest attainable standard of health is a fundamental right for an individual. As we are truly inspired by this, we attempt to propose an innovative system that puts forward a smart patient health tracking system that uses sensors to track patient vital parameters and uses internet to update the doctors so that they can help in case of any issues at the earliest preventing death rates. Patient Health monitoring using IoT is a technology to enable monitoring of patients outside of conventional clinical settings (e.g. in the home), which may increase access to care and decrease healthcare delivery costs. This can significantly improve an individual's quality of life. It allows patients to maintain independence, prevent complications, and minimize personal costs. This system facilitates these goals by delivering care right to the home. In addition, patients and their family members feel comfort knowing that they are being monitored and will be supported if a problem arises.

## LITERATURE SURVEY

### Title: IoT Based Portable ECG Monitoring Device for Smart Healthcare

Title: Prof. Prachi Kamble, Ashish Birajdar

Measuring the Electrocardiogram (ECG) signal is an important method for the identification of Heart diseases. The ECG signal has the knowledge of the degree of how much heart perform its function. The existing devices cannot store the ECG information, which results in loss of ECG information. In this paper, using Internet-of-things (IoT) we propose a new methodology for ECG recording and monitoring. A wearable monitoring node gathers the ECG information and using Wi-Fi technology is transmitted directly to IoT cloud and is stored on SD Card for offline storage. The ECG wave is displayed through the local LCD and developed Web Interface/Mobile-Application. The ECG information can be conveniently acquired using smart devices with a web browser, which has diminished the cross-platform issue.

**Title-2 IoT based Healthware and Healthcare Monitoring System in India****Author-Rakhi Bhardwaj, Shiv Narain Gupta, Manish Gupta, Priyesh Tiwari**

The Indian healthcare scenario has been gradually changing in terms of the use of advance healthcare and healthware systems. In accordance with the recent surrounding conditions and increasing health issues rate, accelerate the use of healthware and healthcare system. The growth of healthcare systems in India is obstructed due by lack of medical professionals, hospitals, poor accessibility to medicines, and unavailability of quality healthcare in distant and rural areas. The Internet of Things (IoT) can play a vital role in improving the poor healthcare system in India by providing a proficient, integrated, well connected, patient-centered system. The IoT technology has emerges with various IOT based healthware, medical monitoring system, and smart homes for elderly patients suffering from chronic disease which serves as an efficient solution to qualitative, effective and accessible healthcare system. The present study provides a review of the available healthware and healthcare system in India and their efficacy for healthcare space.

**Title-3 Cognitive Smart Healthcare for Pathology Detection and Monitoring****Author-SYED UMAR AMIN 1 , M. SHAMIM HOSSAIN 2 , (Senior Member, IEEE), GHULAM MUHAMMAD 1 , (Member, IEEE), MUSAED ALHUSSEIN 1 , AND MD. ABDUR RAHMAN 3**

We propose a cognitive healthcare framework that adopts the Internet of Things (IoT)-cloud technologies. This framework uses smart sensors for communications and deep learning for intelligent decision-making within the smart city perspective. The cognitive and smart framework monitors patients' state in real time and provides accurate, timely, and high-quality healthcare services at low cost. To assess the feasibility of the proposed framework, we present the experimental results of an EEG pathology classification technique that uses deep learning. We employ a range of healthcare smart sensors, including an EEG smart sensor, to record and monitor multimodal healthcare data continuously. The EEG signals from patients are transmitted via smart IoT devices to the cloud, where they are processed and sent to a cognitive module. The system determines the state of the patient by monitoring sensor readings, such as facial expressions, speech, EEG, movements, and gestures. The real-time decision, based on which the future course of action is taken, is made by the cognitive module. When information is transmitted to the deep learning module, the EEG signals are classified as pathologic or normal. The patient state monitoring and the EEG processing results are shared with healthcare providers, who can then assess the patient's condition and provide emergency help if the patient is in a critical state. The proposed deep learning model achieves better accuracy than the state-of-the-art systems.

**Title-4 Development and Implementation of Smart Healthcare Bidet****Author-Sunghil Heo ,Suyong Jeong ,JunDong Lee ,Hwan-Seog Kim**

In this paper, various health indicators are calculated by measuring and analyzing bio-signals in a non-constrained, unrecognized environment through healthcare bidet using contact sensor, non-contact sensor, wireless communication, and AI SW technology. This article describes the development and implementation of smart healthcare bidet that can be used to prevent various diseases. IoT(Internet of things) based medical bidet platform and App software were developed to monitor users' health information through several contact-type sensors (i.e., oxygen, pressure, and thermometer sensors). In particular, this bidet platform is intended to be used for the early detection of human diseases. The pressure, oxygen, and thermometer sensors had accuracies of ~ 4.1, 0.6, and 1.06 % under waterproof conditions, confirming the proper operation of the developed medical sensors attached to the bidet platform. In addition, we developed an app service that works on smartphones.

**Existing system**

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.

This system of methodology forms the drawback of data missing when there is interruption in WIFI network and hence there need to be a constant and stable WIFI network in order to have a constant connectivity to transmit the collected data to the remote server.

**Disadvantages**

Privacy of Data. Privacy is the biggest challenge with IoT, as all the connected devices transfer data in real-time. Personal data can be hacked if this end to end connection is not secure. ...

Accuracy. Accuracy issues may come due to handling such massive data in real-time.

Cost.

**Proposed system:**

The proposed system aims to cover an end-to-end smart, efficient and innovative health application that can be built up with two functional building blocks. However, the main function of the rest building block is to gather all sensory data that are related to the monitoring of the patients, whereas the second block function is to store, process and present the resulted information on the server where the doctors can access. The vibration sensor used in here senses the shaking of the body movements and hence we use it here to monitor whether the patient is shivering so that proper aid can be given.

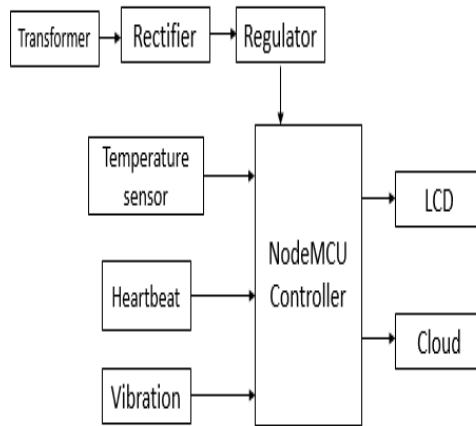
In this proposed system we used NodMCU Controller. Heartbeat and temperature can measure by using sensors. If any of the sensor value getting the threshold level it will intimate family as well as doctor and also send to the Cloud.

**Advantages**

Improving Patient Care and Doctor Response Time. ...

Staff Monitoring and Compliance.

**Block Diagram:**



**Hardware requirements**

- Nodemcu Controller
- Heartbeat sensor
- Temperature sensor
- Vibration sensor
- LCD Display
- Power supply

**Software Requirements**

- Arduino IDE
- Embedded C

**Hardware used**

**IOT (Internet of Things):**

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.

**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.



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.

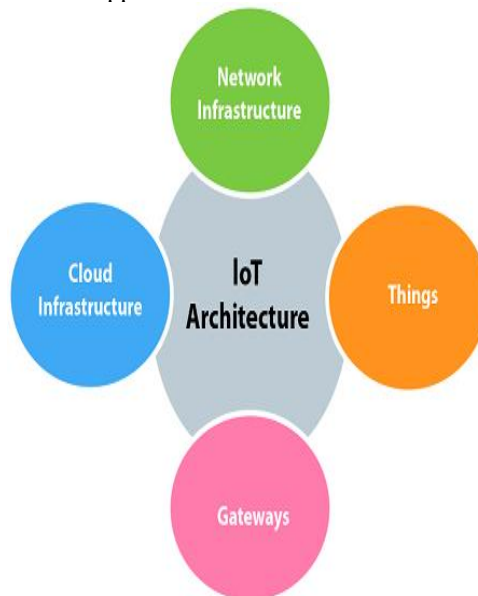
As we have a platform such as a cloud that contains all the data through which we connect all the things around us. For example, a house, where we can connect our home appliances such as air conditioner, light, etc. through each other and all these things are managed at the same platform. Since we have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



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 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).

### 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.



### 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.

**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.

**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.

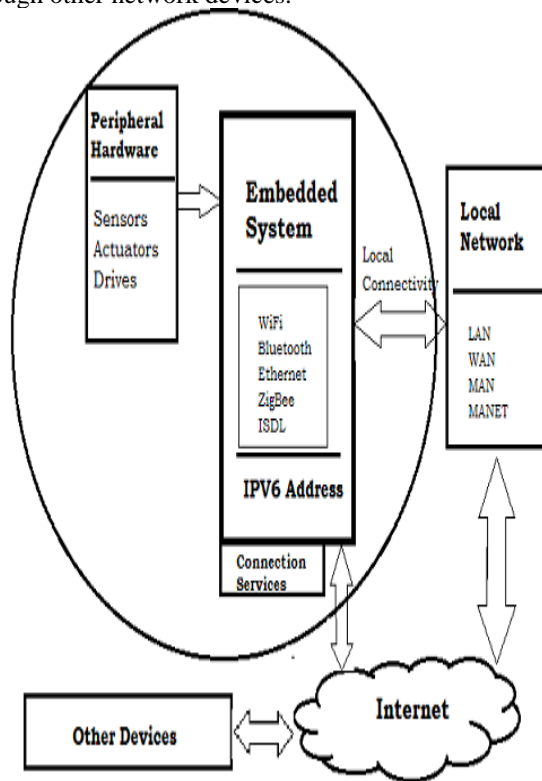
**Disadvantages of IoT**

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:

**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.

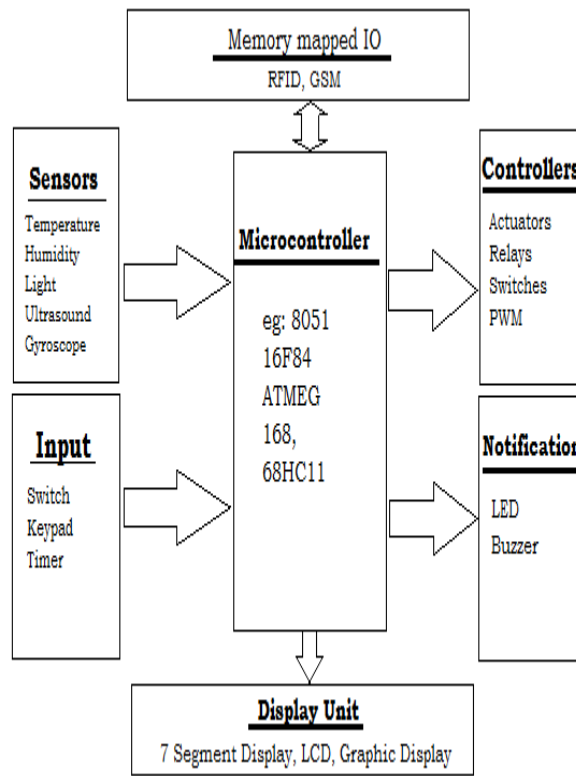
An embedded device system generally runs as a single application. However, these devices can connect through the internet connection, and able communicate through other network devices.



**Internet of Things (IoT)**

**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: Basic Embedded System**

**Embedded System Software**

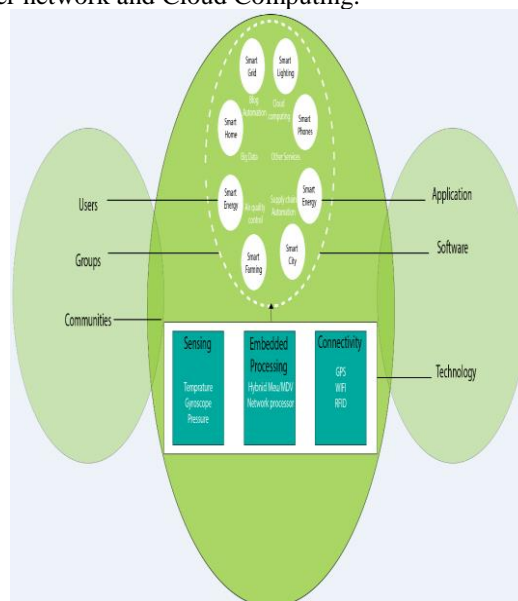
The embedded system that uses the devices for the operating system is based on the language platform, mainly where the real-time operation would be performed. Manufacturers build embedded software in electronics, e.g., cars, telephones, modems, appliances, etc. The embedded system software can be as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

**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.



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



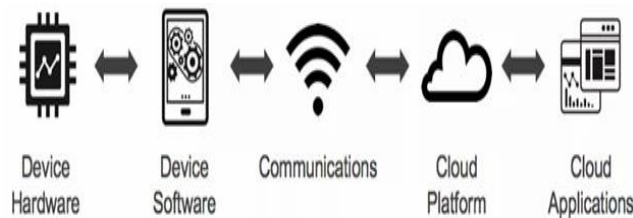
- **Sensing, Embedded processing, Connectivity:** The IoT ecosystem senses its surrounding like temperature, gyroscope, pressure, etc. and make the embedded processing using devices. These devices are connected through any type of devices such as GPS, WiFi, RFID, etc. over the networks.
- **Smart devices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

**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



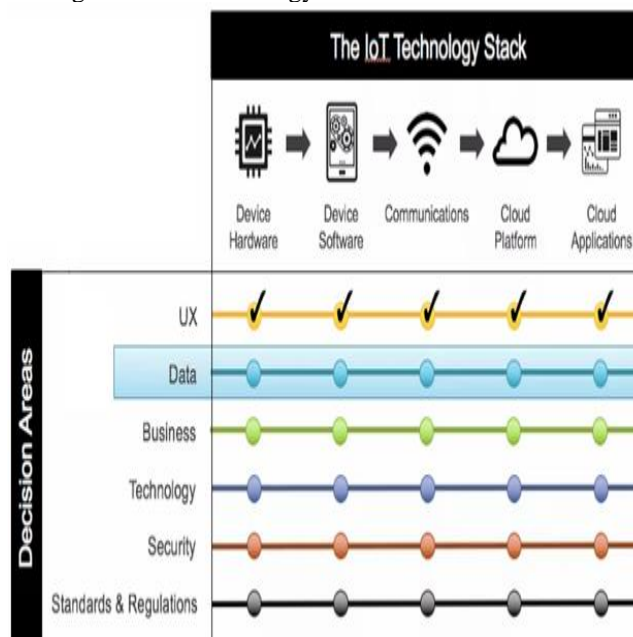
**The IoT Technology Stack**

**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.



### Decision Area of the IoT Decision Framework

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

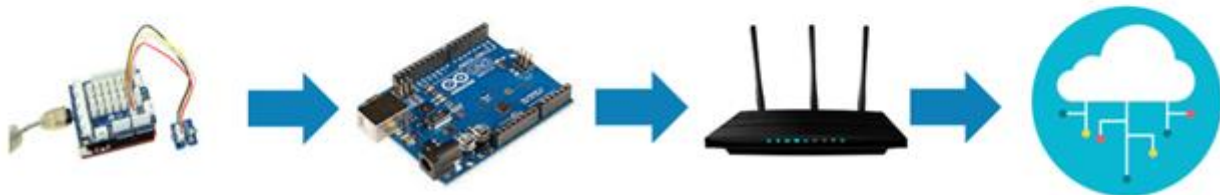
1. **User Experience Decision Area:** This is the area where we concentrate about who are the users, what are their requirements and how to provide a great experience at each step of IoT stack without worrying about the technical details.
2. **Data Decision Area:** In this area, we make the overall data 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.

### 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



### Stages of IoT Solutions Architecture

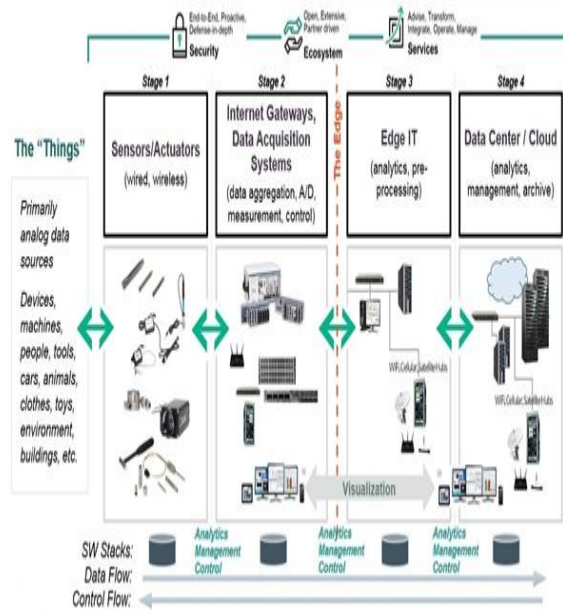
There are several layers of IoT built upon the capability and performance of IoT elements that provides the optimal solution to the business enterprises and end-users. The IoT architecture is a fundamental way to design the various elements of IoT, so that it can deliver services over the networks and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



### The 4 Stage IoT Solutions Architecture



#### What are Smart 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.



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

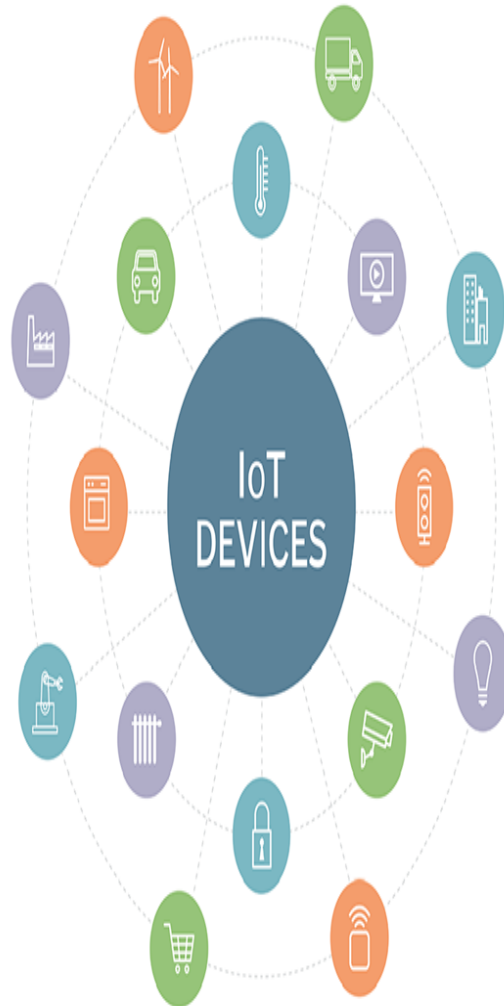
A smart object carries blocks of application logic that make sense for their local situation and interact with human users. A smart object sense, log, and interpret the occurrence within themselves and the environment, and intercommunicate with each other and exchange information with people.

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.

#### IoT Devices

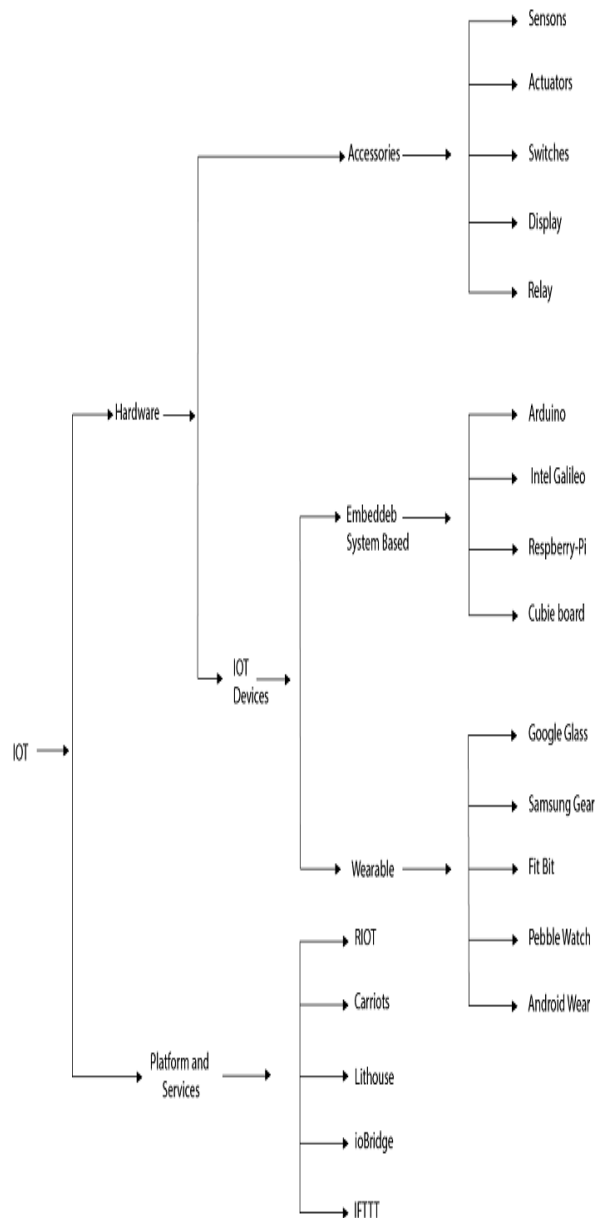
Internet of Things Devices is non-standard devices that connect wirelessly to a network with each other and able to transfer the data. IoT devices are enlarging the internet connectivity beyond standard devices such as smartphones, laptops, tablets, and desktops. Embedding these devices with technology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



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:



IoT Devices and Technologies

**NODE MCU**

NodeMCU is an open source IoT platform.<sup>[4][5]</sup> It includes firmware which runs on the ESP8266 Wi-Fi SoC from Espressif Systems, and hardware which is based on the ESP-12 module.<sup>[6][7]</sup> 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<sup>[8]</sup> and SPIFFS

NodeMCU was created shortly after the ESP8266 came out. On December 30, 2013, Espressif Systems<sup>[6]</sup> began production of the ESP8266.<sup>[10]</sup> The ESP8266 is a Wi-Fi SoC integrated with a Tensilica Xtensa LX106 core,<sup>[citation needed]</sup> 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.<sup>[11]</sup> 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.<sup>[12]</sup> Later that month, Tuan PM ported MQTT client library from Contiki to the ESP8266 SoC platform,<sup>[13]</sup> 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<sup>[14]</sup> to NodeMCU project,<sup>[15]</sup> 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.

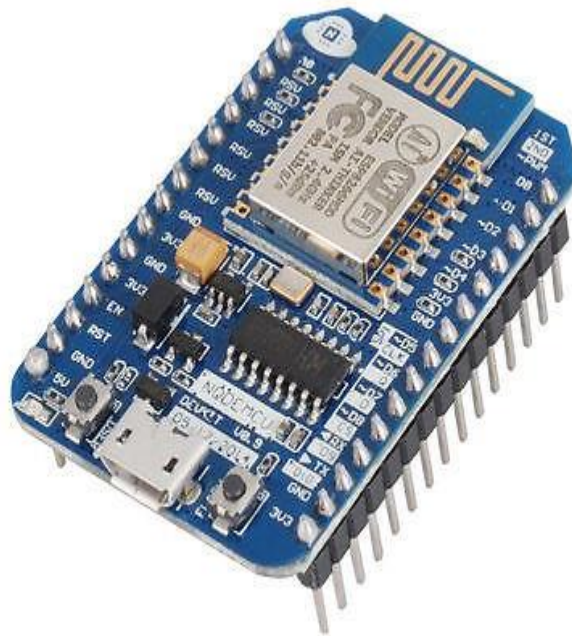
**ESP8266 Arduino Core**<sup>[edit]</sup>

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".<sup>[16]</sup> 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.

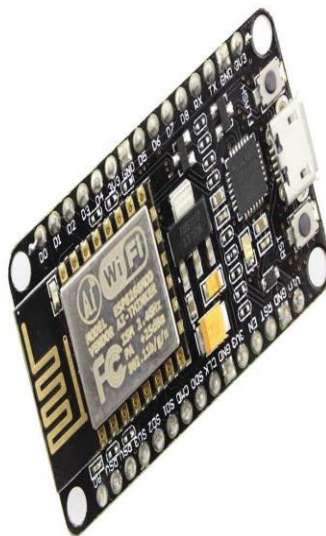


#### NodeMCU Development Board/kit v0.9 (Version1)

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.



### **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++ 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.

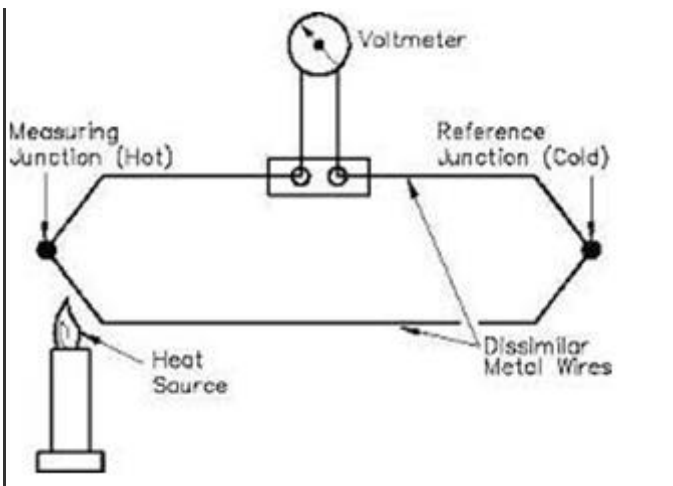
### **TEMPERATURE SENSOR**

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.

#### **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.





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:

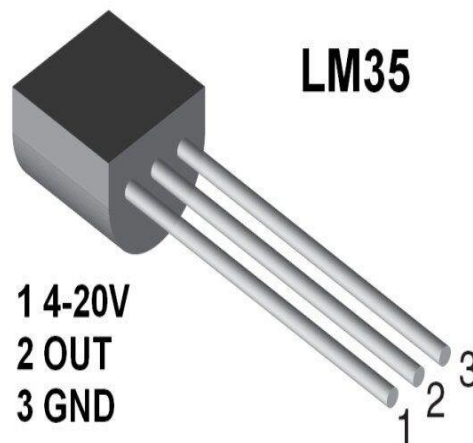
They are classified into different types like Voltage output, Current output, Digital output, Resistance output silicon and Diode temperature sensors. Modern semiconductor temperature sensors offer high accuracy and high linearity over an operating range of about  $55^{\circ}\text{C}$  to  $+150^{\circ}\text{C}$ . Internal amplifiers can scale the output to convenient values, such as  $10\text{mV}/^{\circ}\text{C}$ . 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.

#### 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  $-55^{\circ}\text{C}$  to  $+150^{\circ}\text{C}$ . 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.



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  $-55^{\circ}$  to  $+120^{\circ}\text{C}$ . The basic centigrade temperature sensor ( $+2^{\circ}\text{C}$  to  $+150^{\circ}\text{C}$ ) is shown in figure below.

#### Features of LM35 Temperature Sensor:

- Calibrated directly in  $^{\circ}$  Celsius (Centigrade)
- Rated for full  $1 -55^{\circ}$  to  $+150^{\circ}\text{C}$  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  $0.01^{\circ}\text{C}$  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  $0^{\circ}\text{C}$  to  $70^{\circ}\text{C}$  and it is possible to achieve  $\pm 0.5^{\circ}\text{C}$  accuracy. These sensors completely aligned with digital temperature readings in degree Celsius.

## 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.

#### 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.

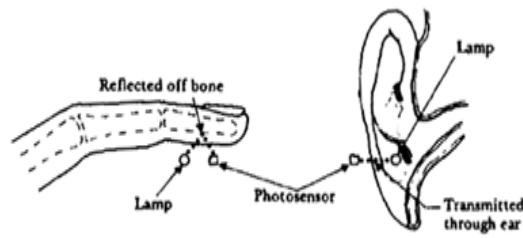
#### Principle of Heartbeat Sensor

The heartbeat sensor is based on the principle of 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:

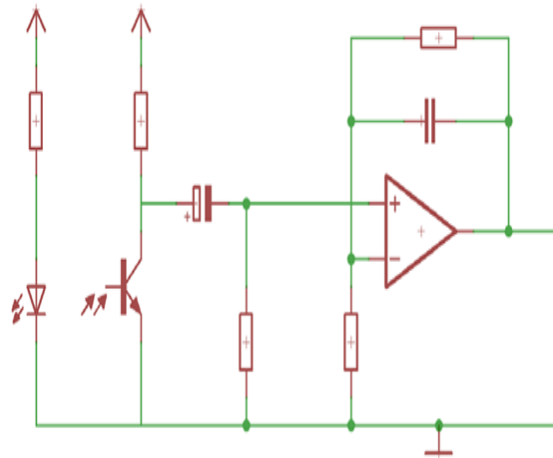
**Transmission:** Light emitted from the light emitting device is transmitted through any vascular region of the body like earlobe and received by the detector.

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



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-

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

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.



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

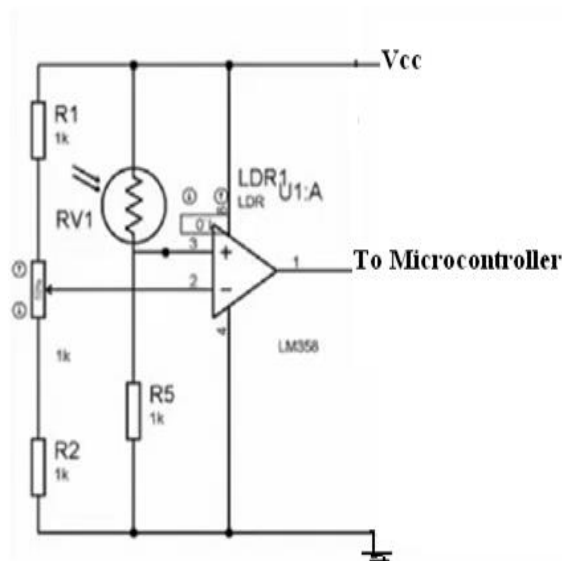
Pin1: To give supply voltage to the LED

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.



### 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.

### VIBRATION SENSORS

The piezoelectric sensor is used for flex, touch, vibration and shock measurement. Its basic principal, at the risk of oversimplification, is as follows: whenever a structure moves, it experiences acceleration. A piezoelectric shock sensor, in turn, can generate a charge when physically accelerated. This combination of properties is then used to modify response or reduce noise and vibration.

Why is that important? Because vibration and shock can shorten the life of any electronic and electromechanical system. Delicate leads and bond wires can be stressed, especially after exposure to long term vibration. Solder joints can break free and PCB traces can ever so slightly tear from impact and impulse shock, creating the hardest type of system failure to debug; an intermittent failure.

This article discusses piezoelectric shock and vibration sensors and sensor technology, focusing on available products (all parts

mentioned here can be found on the Digi-Key website — links are provided), as well as design issues and design techniques.

### HOW IT WORKS:

The piezoelectric effect was discovered by Pierre and Jacques Curie in the latter part of the 19th century. They discovered that minerals such as tourmaline and quartz could transform mechanical energy into an electrical output. The voltage induced from pressure (Greek: piezo) is proportional to that applied pressure, and piezoelectric devices can be used to detect single-pressure events as well as repetitive events.

Still, the ability of certain crystals to exhibit electrical charges under mechanical loading was of no practical use until very-high-input impedance amplifiers enabled engineers to amplify the signals produced by these crystals.

Several materials can be used to make piezoelectric sensors, including tourmaline, gallium phosphate, salts, and quartz. Most electronic applications use quartz since its growth technology is far along, thanks to development of the reverse application of the piezoelectric effect; the quartz oscillator.

Sensors based on the piezoelectric effect can operate from transverse, longitudinal, or shear forces, and are insensitive to electric fields and electromagnetic radiation. The response is also very linear over wide temperature ranges, making it an ideal sensor for rugged environments. For example, gallium phosphate and tourmaline sensors can have a working temperature range of 1,000°C.

The physical design of the piezoelectric sensor depends on the type of sensor you wish to create. For example, the configuration of a pressure sensor, or a shock (impulse) sensor, would arrange a smaller, but well-known mass of the crystal in a transverse configuration, with the loading deformation along the longest tracks to a more massive base (Figure 1). This assures that the applied pressure will load the base from only one direction.

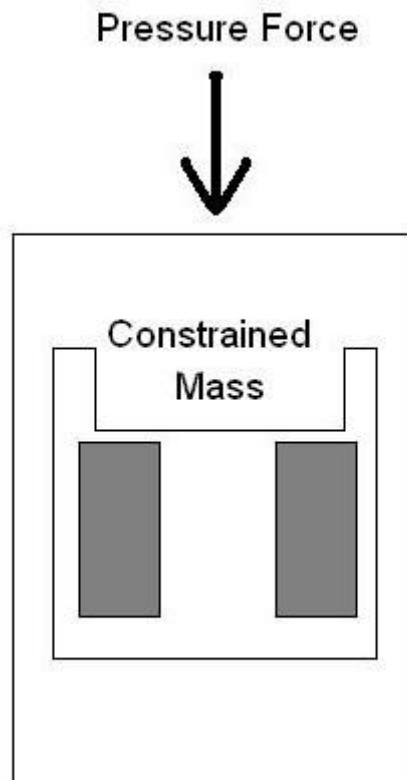


Figure 1: A constrained mass is allowed to deform the crystal sensor in one axis. This configuration is good for force and pressure. An accelerometer based on the piezoelectric effect, would use a known mass to deform the sensing crystal part in either a positive or negative direction depending on the excitation force (Figure 2). It should be noted that you need a known modulus of elasticity in the sensor substrate.



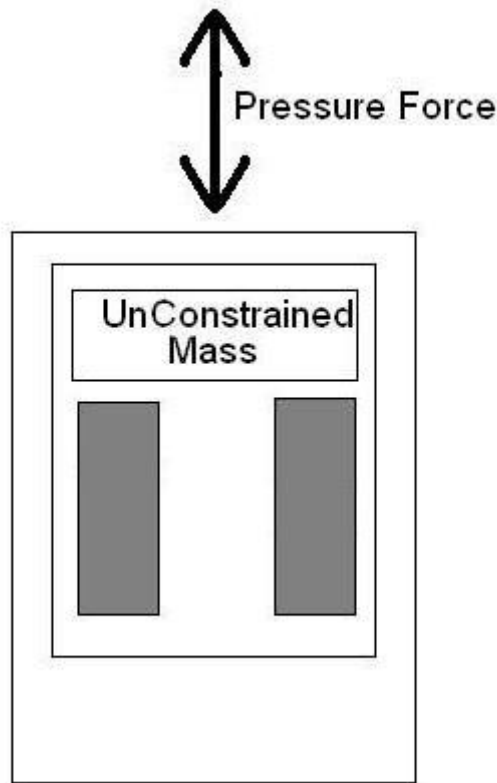


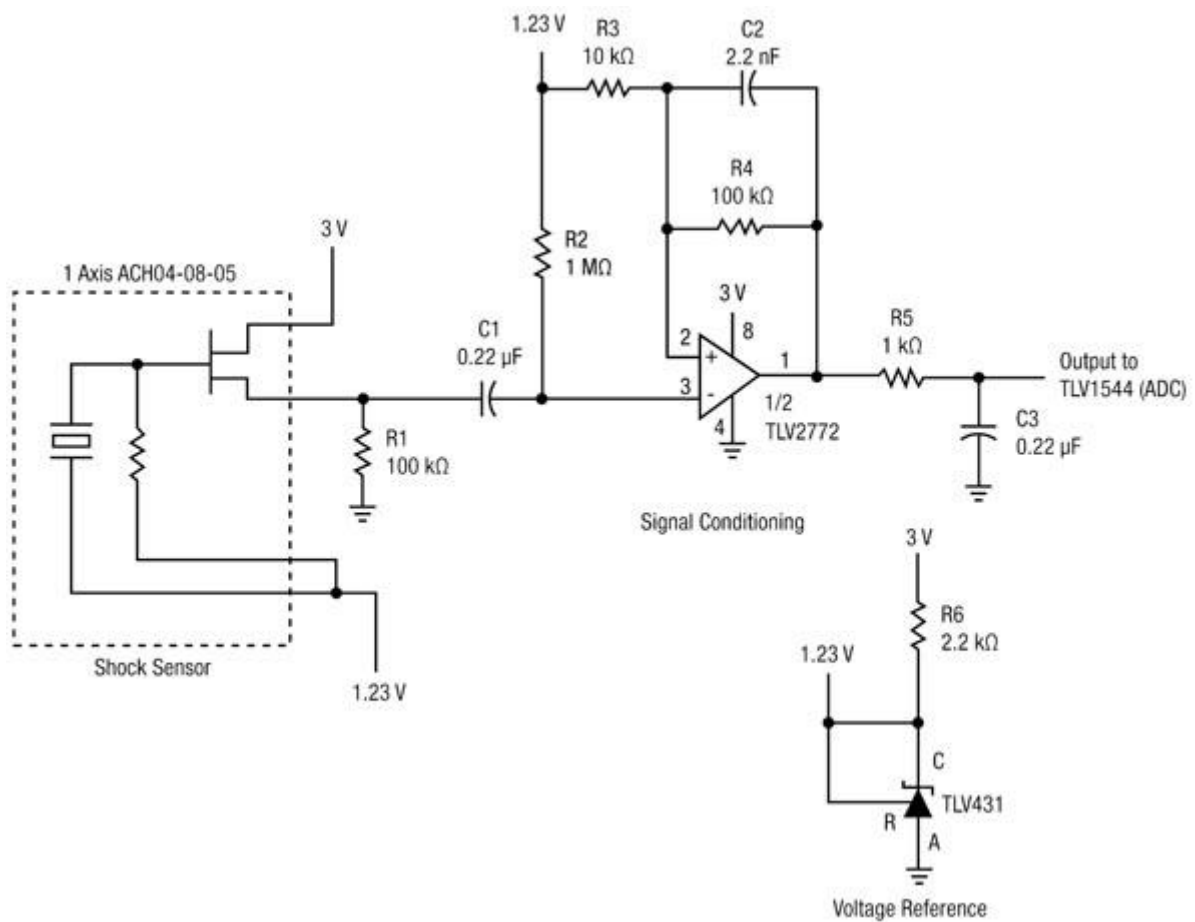
Figure 2: Because the modulus of elasticity is known for a substrate material, the unconstrained mass is allowed to move with vibration making this type of piezoelectric sensor ideal for detecting shock and vibration.

#### Designing with piezoelectric sensors

Piezoelectric sensors require some precautions when connecting to sensitive electronic components. First and foremost, the voltage levels created by hard shock can be very high, even around 100-V spikes.

More than likely, an op amp will be used to interface these sensors to an A/D converter, either discrete or on a microcontroller. One tip is to choose a high-input-impedance op amp to minimize current. One possible candidate is the Linear Technology JFET input dual op amp. It has  $10^{12} \Omega$  input resistance and a 1 MHz gain bandwidth product, good enough to easily handle the vibration ranges of piezoelectric sensors.

Another suitable part is the TLV2771 from Texas Instruments. This rail-to-rail low-power op-amp also has a  $10^{12} \Omega$  differential input resistance and a 5 MHz unity-gain bandwidth. Signal conditioning in a single stage can prepare the input from the shock sensor directly into an A/D converter.



Op amps such as the TI TLV2772 feature high input impedances to help minimize current from the potentially high-voltage inputs from the piezoelectric sensors.

Op-amp circuits can be designed to operate in voltage mode or charge mode. Charge mode is used when the amplifier is remote to the sensor. Voltage mode is used when the amplifier is very close to the sensor.

Another tip is to attenuate the input signal and use the op amp’s gain to bring into the desired range. Be aware that you may need snubbing protection on the inputs of the op amp, especially if the design could be subjected to harsh hits.

Also note that you may think that a pressure sensor would generate only a positive voltage, but, in reality, the signal from the sensor can ring and introduce negative voltage spikes (Figure 4). This means that you may need to squelch negative voltage levels on the op-amp inputs, especially if using only a single rail power supply on the op amp.



Figure 4: Care must be taken when using single rail op amps since shock can cause negative voltage spike ringing that can damage op-amp inputs if not squelched.

### Parts for the design

Many off the shelf piezoelectric sensors are readily available to use in your designs. A case in point is the Parallax 605-00004, which is a piezo vibra tab sensor capable of acting as a switch, or as a vibration sensor (Figure 5). A polymer film laminate uses crimped contacts and features a sensitivity of 50 mV/g.



Figure 5: The flexible through-hole LDTO polymer film piezoelectric sensors can be hard mounted or free floating to detect strain, shock, or vibration.

You should be aware that adding mass to a piezoelectric sensor can change its resonant frequency as well as change its baseline sensitivity. Many piezoelectric sensors like the 605-00004 are characterized to be used this way and provide supporting tables and graphs.

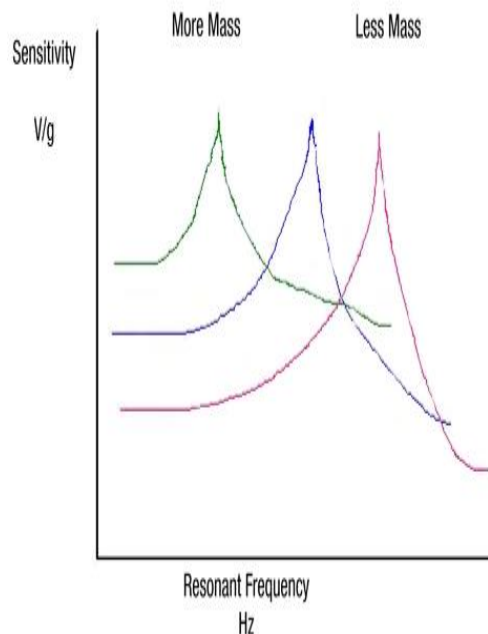


Figure 6: Loading a sensor with mass changes its resonant frequency and sensitivity in a predictable and repeatable way.

Another part worth considering is the Measurement Specialties 0-1002794-0 cantilever piezo film sensor. This is also a vibra tab sensor capable of hard mounting to a surface, floating in an axis of inertia, or mass loaded to prebias and calibrate. The output voltage swings can directly trip a FET or CMOS input, and a multi-axis response can be obtained by offsetting the mass center.

### Other uses:

In addition to sensing vibration and shock, a piezoelectric device can also be used to extract ambient energy. Take for example the MideV22BL, which is a hermetically sealed piezoelectric sensor capable of sensing from 26 to 100 Hz vibrations.

It can be evaluated using the company's VR001 data logger, which is a portable rechargeable data logger that can measure acceleration and vibration in three axes (Figure 7). As a USB peripheral, it can be configured and have its data transferred to a PC or other USB host device. It can also be used to help you develop your own designs based on the piezoelectric sensors you choose. A Mide training module is available on the Digi-Key website to help bring you up to speed quickly when starting a new design.

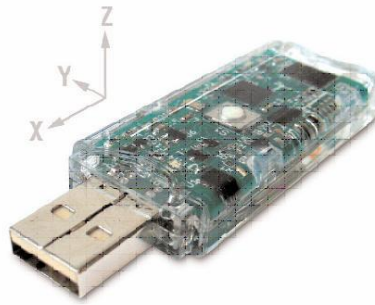


Figure 7: The USB plug in three-axis sensor can be used as a development tool and evaluation tool when learning or starting a new design.

Finally, before you select a piezoelectric sensor for shock or vibration duty, evaluate the criteria specific to your intended application (we've provided some examples here), then, spend some time with the manufacturer data sheets and use the links above to get further information from the product and technology pages on the Digi-Key website.

### LCD Display

Liquid crystal displays (LCDs) have materials which combine the properties of both liquids and crystals. Rather than having a melting point, they have a temperature range within which the molecules are almost as mobile as they would be in a liquid, but are grouped together in an ordered form similar to a crystal.

An LCD consists of two glass panels, with the liquid crystal material sandwiched in between them. The inner surface of the glass plates are coated with transparent electrodes which define the character, symbols or patterns to be displayed. Polymeric layers are present in between the electrodes and the liquid crystal, which makes the liquid crystal molecules to maintain a defined orientation angle.

On each polariser is pasted outside the two glass panels. These polarisers would rotate the light rays passing through them to a definite angle, in a particular direction.

When the LCD is in the off state, light rays are rotated by the two polarisers and the liquid crystal, such that the light rays come out of the LCD without any orientation, and hence the LCD appears transparent.

When sufficient voltage is applied to the electrodes, the liquid crystal molecules would be aligned in a specific direction. The light rays passing through the LCD would be rotated by the polarisers, which would result in activating / highlighting the desired characters.

The LCD's are lightweight with only a few millimeters thickness. Since the LCD's consume less power, they are compatible with low power electronic circuits, and can be powered for long durations.

The LCD's don't generate light and so light is needed to read the display. By using backlighting, reading is possible in the dark. The LCD's have long life and a wide operating temperature range.

Changing the display size or the layout size is relatively simple which makes the LCD's more customer friendly.

The LCDs used exclusively in watches, calculators and measuring instruments are the simple seven-segment displays, having a limited amount of numeric data. The recent advances in technology have resulted in better legibility, more information displaying capability and a wider temperature range. These have resulted in the LCDs being extensively used in telecommunications and entertainment electronics. The LCDs have even started replacing the cathode ray tubes (CRTs) used for the display of text and graphics, and also in small TV applications.

Crystalline dot-matrix (alphanumeric) liquid crystal displays are available in TN, STN types, with or without backlight. The use of C-MOS LCD controller and driver ICs result in low power consumption. These modules can be interfaced with a 4-bit or 8-bit micro processor /Micro controller.

The built-in controller IC has the following features:

- Correspond to high speed MPU interface (2MHz)
- 80 x 8 bit display RAM (80 Characters max)
- 9,920 bit character generator ROM for a total of 240 character fonts. 208 character fonts (5 x 8 dots) 32 character fonts (5 x 10 dots)
- 64 x 8 bit character generator RAM 8 character generator RAM 8 character fonts (5 x 8 dots) 4 character fonts (5 x 10 dots)
- Programmable duty cycles
  - 1/8 – for one line of 5 x 8 dots with cursor
  - 1/11 – for one line of 5 x 10 dots with cursor
  - 1/16 – for one line of 5 x 8 dots with cursor
- Wide range of instruction functions display clear, cursor home, display on/off, cursor on/off, display character blink, cursor shift, display shift.

### FUNCTIONAL DESCRIPTION OF THE CONTROLLER IC

#### REGISTERS:

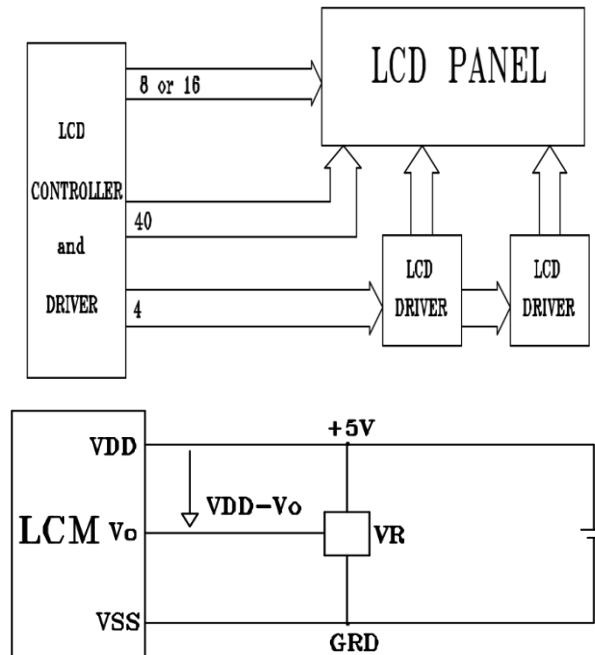
The controller IC has two 8 bit registers, an instruction register (IR) and a data register (DR). The IR stores the instruction codes and address information for display data RAM (DD RAM) and character generator RAM (CG RAM). The IR can be written, but not read by the MPU.

The DR temporarily 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.

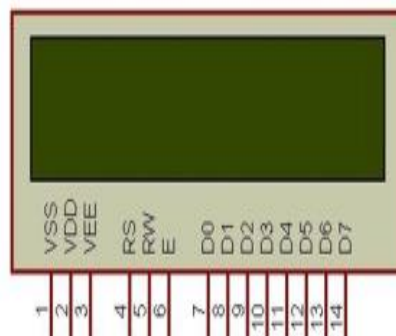
When an address code is written to IR, the data is automatically transferred from the DD RAM or CG RAM to the DR. data transfer between the MPU is then completed when the MPU reads the DR. likewise, for the next MPU read of the DR, data in DD RAM or CG RAM at the address is sent to the DR automatically.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

**Block Diagram**



VDD-V<sub>o</sub>: LCD DRIVING VOLTAGE  
VR: 10K-20K.



**Absolute Maximum Ratings**

Item	Symbol	Min	Max	Unit
Power Voltage	VDD -VSS	0	7.0	V
Input Voltage	V <sub>in</sub>	V <sub>SS</sub>	VDD	
Operating Temperature Range	TOP	0	+50	T
Storage Temperature Range	T <sub>ST</sub>	-20	+60	

**Software requirements**

**ARDUINO UNO AND ITS PROGRAMMING**



Arduino is a tool for making computers that can sense and control more of the physical world than your desktop computer. It's an open-source physical computing platform based on a simple microcontroller board, and a development environment for writing software for the board.

Arduino can be used to develop interactive objects, taking inputs from a variety of switches or sensors, and controlling a variety of lights, motors, and other physical outputs. Arduino projects can be stand-alone, or they can be communicate with software running on your computer. The boards can be assembled by hand or purchased preassembled; the open-source IDE can be downloaded for free.

The Arduino programming language is an implementation of Wiring, a similar physical computing platform, which is based on the Processing multimedia programming environment.

## 2.1 Overview

The Arduino microcontroller is an easy to use yet powerful single board computer that has gained considerable traction in the hobby and professional market. The Arduino is open-source, which means hardware is reasonably priced and development software is free. This guide is for students in ME 2011, or students anywhere who are confronting the Arduino for the first time. For advanced Arduino users, prowl the web; there are lots of resources.

This guide covers the Arduino Uno board (Spark fun DEV-09950, \$29.95), a good choice for students and educators. With the Arduino board, you can write programs and create interface circuits to read switches and other sensors, and to control motors and lights with very little effort. Many of the pictures and drawings in this guide were taken from the documentation on the

This is what the Arduino ide looks like.



The Duemilanove board features an Atmel ATmega328 microcontroller operating at 5 V with 2 Kb of RAM, 32 Kb of flash memory for storing programs and 1 Kb of EEPROM for storing parameters. The clock speed is 16 MHz, which translates to about executing about 300,000 lines of C source code per second. The board has 14 digital I/O pins and 6 analog input pins. There is a USB connector for talking to the host computer and a DC power jack for connecting an external 6-20 V power source, for example a 9 V battery, when running a program while not connected to the host computer. Headers are provided for interfacing to the I/O pins using 22 g solid wire or header connectors.

The Arduino programming language is a simplified version of C/C++. If you know C, programming the Arduino will be familiar. If you do not know C, no need to worry as only a few commands are needed to perform useful functions.

An important feature of the Arduino is that you can create a control program on the host PC, download it to the Arduino and it will run automatically. Remove the USB cable connection to the PC, and the program will still run from the top each time you push the reset button. Remove the battery and put the Arduino board in a closet for six months. When you reconnect the battery, the last program you stored will run. This means that you connect the board to the host PC to develop and debug your program, but once that is done, you no longer need the PC to run the program.

The Arduino Uno is a microcontroller board based on the ATmega328. It has 14 digital input/output pins (of which 6 can be used as PWM outputs), 6 analog inputs, a 16 MHz ceramic resonator, a USB connection, a power jack, an ICSP header, and a reset button. It contains everything needed to support the microcontroller; simply connect it to a computer with a USB cable or power it with a AC-to-DC adapter or battery to get started.

The Uno differs from all preceding boards in that it does not use the FTDI USB-to-serial driver chip. Instead, it features the Atmega16U2 (Atmega8U2 up to version R2) programmed as a USB-to-serial converter.

## Arduino Software (IDE)







- Writing Sketches
  - File
  - Edit
  - Sketch
  - Tools
  - Help
- Sketchbook
- Tabs, Multiple Files, and Compilation
- Uploading
- Libraries
- Third-Party Hardware
- Serial Monitor
- Preferences
- Language Support
- Boards

The Arduino Integrated Development Environment - or Arduino Software (IDE) - contains a text editor for writing code, a message area, a text console, a toolbar with buttons for common functions and a series of menus. It connects to the Arduino and Genuino hardware to upload programs and communicate with them.

### Writing Sketches

Programs written using Arduino Software (IDE) are called **sketches**. These sketches are written in the text editor and are saved with the file extension `.ino`. The editor has features for cutting/pasting and for searching/replacing text. The message area gives feedback while saving and exporting and also displays errors. The console displays text output by the Arduino Software (IDE), including complete error messages and other information. The bottom righthand corner of the window displays the configured board and serial port. The toolbar buttons allow you to verify and upload programs, create, open, and save sketches, and open the serial monitor.

**NB: Versions of the Arduino Software (IDE) prior to 1.0 saved sketches with the extension `.pde`. It is possible to open these files with version 1.0, you will be prompted to save the sketch with the `.ino` extension on save.**

-  **Verify**  
Checks your code for errors compiling it.
-  **Upload**  
Compiles your code and uploads it to the configured board. See uploading below for details.  
Note: If you are using an external programmer with your board, you can hold down the "shift" key on your computer when using this icon. The text will change to "Upload using Programmer"
-  **New**  
Creates a new sketch.
-  **Open**  
Presents a menu of all the sketches in your sketchbook. Clicking one will open it within the current window overwriting its content.  
Note: due to a bug in Java, this menu doesn't scroll; if you need to open a sketch late in the list, use the **File | Sketchbook** menu instead.
-  **Save**  
Saves your sketch.
-  **Serial** Monitor  
Opens the serial monitor.

Additional commands are found within the five menus: **File, Edit, Sketch, Tools, Help**. The menus are context sensitive, which means only those items relevant to the work currently being carried out are available.

- File**
  - **New**  
Creates a new instance of the editor, with the bare minimum structure of a sketch already in place.
  - **Open**  
Allows to load a sketch file browsing through the computer drives and folders.
  - **Open** Recent  
Provides a short list of the most recent sketches, ready to be opened.
  - **Sketchbook**  
Shows the current sketches within the sketchbook folder structure; clicking on any name opens the corresponding sketch in a new editor instance.
  - **Examples**  
Any example provided by the Arduino Software (IDE) or library shows up in this menu item. All the examples are structured in a tree that allows easy access by topic or library.

- Close  
Closes the instance of the Arduino Software from which it is clicked.
- Save  
Saves the sketch with the current name. If the file hasn't been named before, a name will be provided in a "Save as.." window.
- Save as...  
Allows to save the current sketch with a different name.
- Page Setup  
It shows the Page Setup window for printing.
- Print  
Sends the current sketch to the printer according to the settings defined in Page Setup.
- Preferences  
Opens the Preferences window where some settings of the IDE may be customized, as the language of the IDE interface.
- Quit  
Closes all IDE windows. The same sketches open when Quit was chosen will be automatically reopened the next time you start the IDE.
- Edit
- Undo/Redo  
Goes back of one or more steps you did while editing; when you go back, you may go forward with Redo.
- Cut  
Removes the selected text from the editor and places it into the clipboard.
- Copy
- Copy for Forum  
Copies the code of your sketch to the clipboard in a form suitable for posting to the forum, complete with syntax coloring.
- Copy as HTML  
Copies the code of your sketch to the clipboard as HTML, suitable for embedding in web pages.
- Paste  
Puts the contents of the clipboard at the cursor position, in the editor.
- Select All  
Selects and highlights the whole content of the editor.
- Comment/Uncomment  
Puts or removes the // comment marker at the beginning of each selected line.
- Increase/Decrease Indent  
Adds or subtracts a space at the beginning of each selected line, moving the text one space on the right or eliminating a space at the beginning.
- Find
- Find Next  
Opens the Find and Replace window where you can specify text to search inside the current sketch according to several options.
- Find  
Highlights the next occurrence - if any - of the string specified as the search item in the Find window, relative to the cursor position.
- Find Previous  
Highlights the previous occurrence - if any - of the string specified as the search item in the Find window relative to the cursor position.
- Sketch
- Verify/Compile  
Checks your sketch for errors compiling it; it will report memory usage for code and variables in the console area.
- Upload
- Upload Using Programmer  
Compiles and loads the binary file onto the configured board through the configured Port.
- Upload Using Programmer  
This will overwrite the bootloader on the board; you will need to use Tools > Burn Bootloader to restore it and be able to Upload to USB serial port again. However, it allows you to use the full capacity of the Flash memory for your sketch. Please note that this command will NOT burn the fuses. To do so a Tools -> Burn Bootloader command must be executed.
- Export Compiled Binary  
Saves a .hex file that may be kept as archive or sent to the board using other tools.
- Show Sketch Folder  
Opens the current sketch folder.
- Include Library  
Adds a library to your sketch by inserting #include statements at the start of your code. For more details, see libraries below. Additionally, from this menu item you can access the Library Manager and import new libraries from .zip files.
- Add File...  
Adds a source file to the sketch (it will be copied from its current location). The new file appears in a new tab in the sketch window. Files can be removed from the sketch using the tab menu accessible clicking on the small triangle icon below the serial monitor one on the right side of the toolbar.

## Tools

- **Auto** Format  
This formats your code nicely: i.e. indents it so that opening and closing curly braces line up, and that the statements inside curly braces are indented more.
- **Archive** Sketch  
Archives a copy of the current sketch in .zip format. The archive is placed in the same directory as the sketch.
- **Fix Encoding &** Reload  
Fixes possible discrepancies between the editor char map encoding and other operating systems char maps.
- **Serial** Monitor  
Opens the serial monitor window and initiates the exchange of data with any connected board on the currently selected Port. This usually resets the board, if the board supports Reset over serial port opening.
- **Board**  
Select the board that you're using. See below for descriptions of the various boards.
- **Port**  
This menu contains all the serial devices (real or virtual) on your machine. It should automatically refresh every time you open the top-level tools menu.
- **Programmer**  
For selecting a hardware programmer when programming a board or chip and not using the onboard USB-serial connection. Normally you won't need this, but if you're burning a bootloader to a new microcontroller, you will use this.
- **Burn** Bootloader  
The items in this menu allow you to burn a bootloader onto the microcontroller on an Arduino board. This is not required for normal use of an Arduino or Genuino board but is useful if you purchase a new ATmega microcontroller (which normally come without a bootloader). Ensure that you've selected the correct board from the **Boards** menu before burning the bootloader on the target board. This command also set the right fuses.
- **Help**  
Here you find easy access to a number of documents that come with the Arduino Software (IDE). You have access to Getting Started, Reference, this guide to the IDE and other documents locally, without an internet connection. The documents are a local copy of the online ones and may link back to our online website.
- **Find in** Reference  
This is the only interactive function of the Help menu: it directly selects the relevant page in the local copy of the Reference for the function or command under the cursor.

## Sketchbook

The Arduino Software (IDE) uses the concept of a sketchbook: a standard place to store your programs (or sketches). The sketches in your sketchbook can be opened from the **File > Sketchbook** menu or from the **Open** button on the toolbar. The first time you run the Arduino software, it will automatically create a directory for your sketchbook. You can view or change the location of the sketchbook location from with the **Preferences** dialog.

**Beginning with version 1.0, files are saved with a .ino file extension. Previous versions use the .pde extension. You may still open .pde named files in version 1.0 and later, the software will automatically rename the extension to .ino.**

## Tabs, Multiple Files, and Compilation

Allows you to manage sketches with more than one file (each of which appears in its own tab). These can be normal Arduino code files (no visible extension), C files (.c extension), C++ files (.cpp), or header files (.h).

## Uploading

Before uploading your sketch, you need to select the correct items from the **Tools > Board** and **Tools > Port** menus. The boards are described below. On the Mac, the serial port is probably something like `/dev/tty.usbmodem241` (for an Uno or Mega2560 or Leonardo) or `/dev/tty.usbserial-1B1` (for a Duemilanove or earlier USB board), or `/dev/tty.USA19QW1b1P1.1` (for a serial board connected with a Keyspan USB-to-Serial adapter). On Windows, it's probably COM1 or COM2 (for a serial board) or COM4, COM5, COM7, or higher (for a USB board) - to find out, you look for USB serial device in the ports section of the Windows Device Manager. On Linux, it should be `/dev/ttyACMx`, `/dev/ttyUSBx` or similar. Once you've selected the correct serial port and board, press the upload button in the toolbar or select the **Upload** item from the **Sketch** menu. Current Arduino boards will reset automatically and begin the upload. With older boards (pre-Diecimila) that lack auto-reset, you'll need to press the reset button on the board just before starting the upload. On most boards, you'll see the RX and TX LEDs blink as the sketch is uploaded. The Arduino Software (IDE) will display a message when the upload is complete, or show an error.

When you upload a sketch, you're using the Arduino **bootloader**, a small program that has been loaded on to the microcontroller on your board. It allows you to upload code without using any additional hardware. The bootloader is active for a few seconds when the board resets; then it starts whichever sketch was most recently uploaded to the microcontroller. The bootloader will blink the on-board (pin 13) LED when it starts (i.e. when the board resets).

## Libraries

Libraries provide extra functionality for use in sketches, e.g. working with hardware or manipulating data. To use a library in a sketch, select it from the **Sketch > Import Library** menu. This will insert one or more **#include** statements at the top of the sketch and compile the library with your sketch. Because libraries are uploaded to the board with your sketch, they increase the amount of space it takes up. If a sketch no longer needs a library, simply delete its **#include** statements from the top of your code.

There is a list of libraries in the reference. Some libraries are included with the Arduino software. Others can be downloaded from a variety of sources or through the Library Manager. Starting with version 1.0.5 of the IDE, you do can import a library from a zip file and use it in an open sketch. See these instructions for installing a third-party library.

To write your own library, see this tutorial.

#### Third-Party Hardware

Support for third-party hardware can be added to the **hardware** directory of your sketchbook directory. Platforms installed there may include board definitions (which appear in the board menu), core libraries, bootloaders, and programmer definitions. To install, create the **hardware** directory, then unzip the third-party platform into its own sub-directory. (Don't use "arduino" as the sub-directory name or you'll override the built-in Arduino platform.) To uninstall, simply delete its directory.

For details on creating packages for third-party hardware, see the Arduino IDE 1.5 3rd party Hardware specification.

#### Serial Monitor

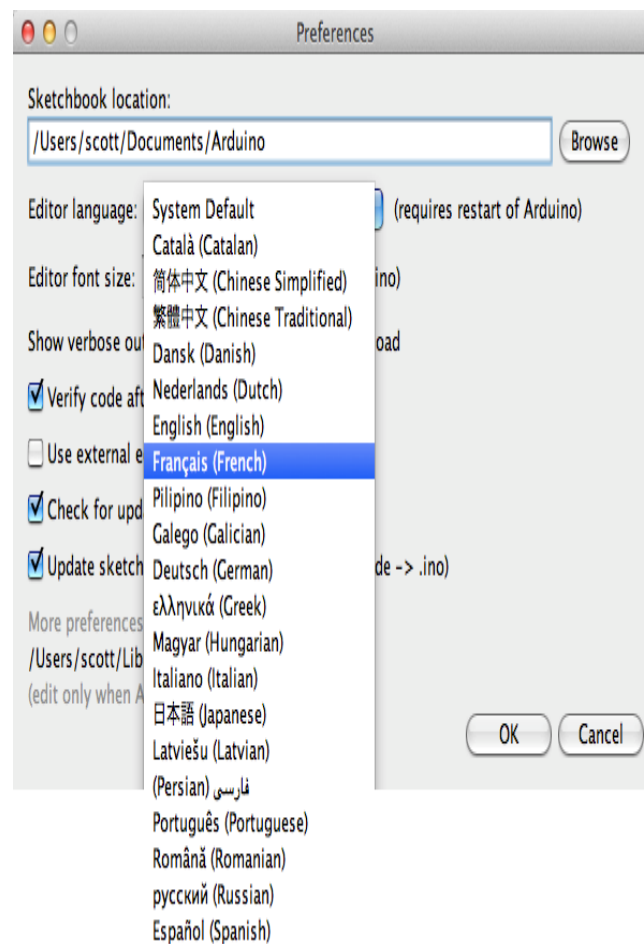
This displays serial sent from the Arduino or Genuino board over USB or serial connector. To send data to the board, enter text and click on the "send" button or press enter. Choose the baud rate from the drop-down menu that matches the rate passed to **Serial.begin** in your sketch. Note that on Windows, Mac or Linux the board will reset (it will rerun your sketch) when you connect with the serial monitor. Please note that the Serial Monitor does not process control characters; if your sketch needs a complete management of the serial communication with control characters, you can use an external terminal program and connect it to the COM port assigned to your Arduino board.

You can also talk to the board from Processing, Flash, MaxMSP, etc (see the interfacing page for details).

#### Preferences

Some preferences can be set in the preferences dialog (found under the **Arduino** menu on the Mac, or **File** on Windows and Linux). The rest can be found in the preferences file, whose location is shown in the preference dialog.

#### Language Support



Since version 1.0.1, the Arduino Software (IDE) has been translated into 30+ different languages. By default, the IDE loads in the language selected by your operating system. (Note: on Windows and possibly Linux, this is determined by the locale setting which controls currency and date formats, not by the language the operating system is displayed in.)

If you would like to change the language manually, start the Arduino Software (IDE) and open the **Preferences** window. Next to the **Editor Language** there is a dropdown menu of currently supported languages. Select your preferred language from the menu, and restart the software to use the selected language. If your operating system language is not supported, the Arduino Software (IDE) will default to English.

You can return the software to its default setting of selecting its language based on your operating system by selecting **System Default** from the **Editor Language** drop-down. This setting will take effect when you restart the Arduino Software (IDE).



Similarly, after changing your operating system's settings, you must restart the Arduino Software (IDE) to update it to the new default language.

#### Boards

The board selection has two effects: it sets the parameters (e.g. CPU speed and baud rate) used when compiling and uploading sketches; and sets and the file and fuse settings used by the burn bootloader command. Some of the board definitions differ only in the latter, so even if you've been uploading successfully with a particular selection you'll want to check it before burning the bootloader. You can find a comparison table between the various boards here.

Arduino Software (IDE) includes the built in support for the boards in the following list, all based on the AVR Core. The Boards Manager included in the standard installation allows to add support for the growing number of new boards based on different cores like Arduino Due, Arduino Zero, Edison, Galileo and so on.

- Arduino Yùn  
An ATmega32u4 running at 16 MHz with auto-reset, 12 Analog In, 20 Digital I/O and 7 PWM.
- Arduino/Genuino Uno  
An ATmega328P running at 16 MHz with auto-reset, 6 Analog In, 14 Digital I/O and 6 PWM.
- Arduino Diecimila or Duemilanove w/ ATmega168  
An ATmega168 running at 16 MHz with auto-reset.
- Arduino Nano w/ ATmega328P  
An ATmega328P running at 16 MHz with auto-reset. Has eight analog inputs.
- Arduino/Genuino Mega 2560  
An ATmega2560 running at 16 MHz with auto-reset, 16 Analog In, 54 Digital I/O and 15 PWM.
- Arduino Mega  
An ATmega1280 running at 16 MHz with auto-reset, 16 Analog In, 54 Digital I/O and 15 PWM.
- Arduino Mega ADK  
An ATmega2560 running at 16 MHz with auto-reset, 16 Analog In, 54 Digital I/O and 15 PWM.
- Arduino Leonardo  
An ATmega32u4 running at 16 MHz with auto-reset, 12 Analog In, 20 Digital I/O and 7 PWM.
- Arduino/Genuino Micro  
An ATmega32u4 running at 16 MHz with auto-reset, 12 Analog In, 20 Digital I/O and 7 PWM.
- Arduino Esplora  
An ATmega32u4 running at 16 MHz with auto-reset.
- Arduino Mini w/ ATmega328P  
An ATmega328P running at 16 MHz with auto-reset, 8 Analog In, 14 Digital I/O and 6 PWM.
- Arduino Ethernet  
Equivalent to Arduino UNO with an Ethernet shield: An ATmega328P running at 16 MHz with auto-reset, 6 Analog In, 14 Digital I/O and 6 PWM.
- Arduino Fio  
An ATmega328P running at 8 MHz with auto-reset. Equivalent to Arduino Pro or Pro Mini (3.3V, 8 MHz) w/ ATmega328P, 6 Analog In, 14 Digital I/O and 6 PWM.
- Arduino BT w/ ATmega328P  
ATmega328P running at 16 MHz. The bootloader burned (4 KB) includes codes to initialize the on-board bluetooth module, 6 Analog In, 14 Digital I/O and 6 PWM..
- LilyPad Arduino USB  
An ATmega32u4 running at 8 MHz with auto-reset, 4 Analog In, 9 Digital I/O and 4 PWM.
- LilyPad Arduino  
An ATmega168 or ATmega132 running at 8 MHz with auto-reset, 6 Analog In, 14 Digital I/O and 6 PWM.
- Arduino Pro or Pro Mini (5V, 16 MHz) w/ ATmega328P  
An ATmega328P running at 16 MHz with auto-reset. Equivalent to Arduino Duemilanove or Nano w/ ATmega328P; 6 Analog In, 14 Digital I/O and 6 PWM.
- Arduino NG or older w/ ATmega168  
An ATmega168 running at 16 MHz without auto-reset. Compilation and upload is equivalent to Arduino Diecimila or Duemilanove w/ ATmega168, but the bootloader burned has a slower timeout (and blinks the pin 13 LED three times on reset); 6 Analog In, 14 Digital I/O and 6 PWM.
- Arduino Robot Control  
An ATmega328P running at 16 MHz with auto-reset.
- Arduino Robot Motor  
An ATmega328P running at 16 MHz with auto-reset.
- Arduino Gemma  
An ATtiny85 running at 8 MHz with auto-reset, 1 Analog In, 3 Digital I/O and 2 PWM.

For instructions on installing support for other boards, see third-party hardware above.



Last revision 2015/09/07 by SM

The text of the Arduino getting started guide is licensed under a Creative Commons Attribution-ShareAlike 3.0 License. Code samples in the guide are released into the public domain.

## CONCLUSION

In this paper, we have presented and proved the prototype for an automatic system that guarantees a constant monitoring of various health parameters and prediction of any kind of disease or disorder that prevents the patient from the pain of paying frequent visits to the hospitals. The proposed system can be set-up in the hospitals and massive amount of data can be obtained and stored in the online database. Even the results can be made to be accessed from mobile through an application. The system can be further improved further by adding artificial intelligence system components to facilitate the doctors and the patients. The data, consisting medical history of many patients' parameters and corresponding results, can be explored using data mining, in search of consistent patterns and systematic relationships in the disease. For instance, if a patient's health parameters are changing in the same pattern as those of a previous patient in the database, the consequences can also be estimated. If the similar patterns are found repeatedly, it would be easier for the doctors and medical researchers to find a remedy for the problem.

## REFERENCES:

- 1) Prof. Prachi Kamble 2 Ashish Birajdar, "IoT Based Portable ECG Monitoring Device for Smart Healthcare", 978-1-7281-1599-3/19/\$31.00 ©2019 IEEE
- 2) Rakhi Bhardwaj, Shiv Narain Gupta, Manish Gupta, Priyesh Tiwari, "IoT based Healthware and Healthcare Monitoring System in India", 978-1-7281-7741-0/20/\$31.00 ©2021 IEEE | DOI: 10.1109/ICACITE51222.2021.9404633
- 3) SYED UMAR AMIN 1, M. SHAMIM HOSSAIN 2, (Senior Member, IEEE), GHULAM MUHAMMAD 1, (Member, IEEE), MUSAED ALHUSSEIN 1, AND MD. ABDUR RAHMAN 3, (Senior Member, IEEE), "Cognitive Smart Healthcare for Pathology Detection and Monitoring" Digital Object Identifier 10.1109/ACCESS.2019.2891390
- 4) Sungphil Heo, Suyong Jeong, JunDong Lee, Hwan-Seog Kim, "Development and Implementation of Smart Healthcare Bidet", 978-1-7281-5584-5/19/\$31.00 ©2019 IEEE DOI 10.1109/CSCI49370.2019.00294
- 5) S. B. Baker, W. Xiang, and I. Atkinson, "Internet of things for smart healthcare: Technologies, challenges, and opportunities," IEEE Access, vol. 5, pp. 26521-26544, 2017.
- 6) P. Strojnik and P. H. Peckham, The Biomedical Engineering Handbook. Boca Raton, FL, USA: CRC Press, 2000.
- 7) S. P. Heo, D. H. Noh, C. B. Moon, and D. S. Kim, "Trend of IoT-based Healthcare Service," IEMEK J. Embedded Syst. Appl., vol. 10, no. 4, pp. 221-231, 2015.
- 8) L. Johnson, "A new method for pulse oximetry processing inherent insensitivity to artifact," IEEE Trans. Biomed. Eng., vol. 5, pp. 2110-2118, 2001.