Project Stage II Report

On

"IoT Based Patient Health Monitoring using ESP8266 & Arduino"

Submitted in the fulfillment of the requirements
For the Degree of
Bachelor of Technology Semester VIII
in
Electronics & Telecommunication Engineering

By

Krishna Mani Raj (1814110507)
Kaushal Kumar (1814110505)
Manan Agrawal (1814110516)

Under the guidance of

Guide Name :-Prof. Prasad Dhondiram Kadam

Department of Electronics & Telecommunication Engineering
Bharati Vidyapeeth (Deemed To Be University)
College of Engineering,Pune – 4110043

Academic Year 2021-22

BHARATI VIDYAPEETH (DEEMED TO BE UNIVERSITY)
COLLEGE OF ENGINEERING, PUNE – 4110043
DEPARTMENT OF ELECTRONICS& TELECOMMUNICATION ENGINEERING

CERTIFICATE

Certified that the project report entitled, **"IoT Based Patient Health Monitoring using ESP8266 & Arduino"** is a bonafied work done **by Krishna Mani Raj , Kaushal Kumar, Manan Agrawal** in fulfillment of the requirements for the award of degree of Bachelor of Technology in Electronics & Telecommunication Engineering.

Date:

Prof. P.D Kadam	Prof. D.K Ray	Prof. S.K.Oza
(Project Guide)	(Project Coordinator)	(H.O.D)

ACKNOWLEDGEMENT

We would like to extend our sincere gratitude to the Principal "**Dr. Vidula Sohoni"**, Head of Department Electronics & Telecommunication, "**Prof. S.K.Oza",** for nurturing a congenial yet competitive environment, which motivates all the students not only to pursue goals but also to elevate the Humanitarian level.

Inspiration and guidance are invaluable in every aspect of life, which we have received from our respected project guide "**Prof. Prasad Dhondiram Kadam",** who gave us his careful and ardent guidance because of which we are able to complete this project. More words won't suffice to express our gratitude to his untiring devotion. He undoubtedly belongs to the members of the artistic gallery who are masters in all aspects.

We would also like to thank all the faculty members who directly or indirectly helped us from time to time with their invaluable inputs.

Table of Contents

LIST OF FIGURES

ABSTRACT

With tons of new healthcare technology start-ups, IoT is rapidly revolutionizing the healthcare industry. In this project, we have designed the IoT Based Patient Health Monitoring System using ESP8266 & Arduino. The IoT platform used in this project is Thing Speak. Thing Speak is an open-source Internet of Things (IoT) application and API to store and retrieve data from things using the HTTP protocol over the Internet or via a Local Area Network. This IoT device could read the pulse rate and measure the surrounding temperature. It continuously monitors the pulse rate and surrounding temperature and updates them to an IoT platform.

Monitoring your beloved ones becomes a difficult task in the modern day life. Keeping track of the health status of your patient at home is a difficult task. Especially old age patients should be periodically monitored and their loved ones need to be informed about their health status from time to time while at work. So we propose an innovative system that automated this task with ease. Our system puts forward a smart patient health monitoring system that uses Sensors to track patient health and uses internet to inform their loved ones in case of any issues.

Our system uses temperature , heartbeat rate sensing to keep track of patient health. The sensors are connected to a microcontroller to track the status which is in turn interfaced Wi-Fi connection in order to transmit alerts. If system detects any abrupt changes in patient heartbeat rate, body temperature, the system automatically alerts the user about the patients status over IOT and also shows details of heartbeat, temperature of patient live over the internet.

IOT based patient health monitoring system effectively uses internet to monitor patient health stats and save lives on time. An Arduino board is used for analyzing the inputs from the patient and any abnormality felt by the patient causes the monitoring system to give an alarm.

Also all the process parameters within an interval selectable by the user are recorded online. This is very useful for future analysis and review of patient's health condition. For more versatile medical applications, this project can be improvised, by incorporating blood pressure monitoring systems, dental sensors and annunciation systems, thereby making it useful in hospitals as a very efficient and dedicated patient care system.

Chapter 1

1.1 Introduction

A Patient Health Monitoring System is an extension of a hospital medical system where a patient's vital body state can be monitored remotely. Traditionally the detection systems were only found in hospitals and were characterized by huge and complex circuitry which required high power consumption. Continuous advances in the semiconductor technology industry have led to sensors and microcontrollers that are smaller in size, faster in operation, low in power consumption and affordable in cost.

According to research, we found that approximately 2000 people died monthly due to the only carelessness of their health. This is because they don"t have time for themselves and forget about their health management due to a heavy workload. The reason behind to make this project is the growing world of technology and people forget their health checkup which is needed to be done monthly or quarterly. As we all know that internet of things make our life easier. So, we have decided to make an internet of things based healthcare project for people who provide them all the personal information about their health on their mobile and they can check their all historical health data.

The best part of this project is that it can be used by everyone and make our health management easier than available systems. It provides a solution for measurement of body parameters like ECG, Temperature, Moisture, and Heartbeat. It also detects the body condition and location of the patients. This system also generates an alert when it required that means at the time of any critical conditions and notifications about the medicines, location change, conditions etc.

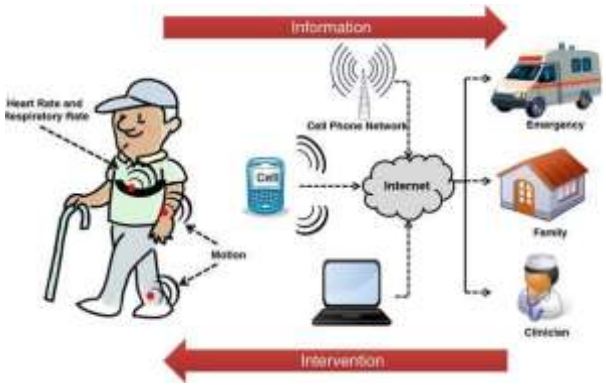

Fig1.1 : Introduction to the system

1.2 Introduction to IOT

The 'Thing' in IoT can be any device with any kind of built-in-sensors with the ability to collect and transfer data over a network without manual intervention. The embedded technology in the object helps them to interact with internal states and the external environment, which in turn helps in decisions making process.

Fig 1.2 : IOT

In a nutshell, IoT is a concept that connects all the devices to the internet and let them communicate with each other over the internet. IoT is a giant network of connected devices – all of which gather and share data about how they are used and the environments in which they are operated.

By doing so, each of your devices will be learning from the experience of other devices, as humans do. IoT is trying to expand the interdependence in human-
i.e interact, contribute and collaborate to things.

CHAPTER 2

PROBLEM DEFINITION & OBJECTIVE

Patient Health Monitoring can provide useful physiological information in the home. This monitoring is useful for elderly or chronically ill patients who would like to avoid a long hospital stay. Wireless sensors are used to collect and transmit signals of interest and a processor is programmed to receive and automatically analyze the sensor signals. In this project you are to choose appropriate sensors according to what you would like to detect and design algorithms to realize your detection.

The objective of the project was to come up with a system that can monitor and provide physiological information remotely in the home. The monitoring system would be useful for elderly or chronically ill patients who would like to avoid a long costly hospital stay. Wireless sensors would be used to collect and transmit signals of interest and a microcontroller was programmed to receive and automatically analyze the sensor signals.
For the devices that require instant intervention by a specialist doctor it was important that they be autonomous, non-invasive to the patient/users everyday life activities. In this way they were to be easy to use, minimal in size and weight, consume less power for maximum use on a single charge, and functional – able to withstand physical shock in the case of fall detection.

In both cases for accurate physiological signal detection, the circuitry in the detection system was crucial. To be able to accurately collect and manage the signal information Integrated circuits and microprocessors were implemented. This was done to minimize the drift voltages and any white noise that could be picked by the detection system.

Objective :-

- To develop health monitoring system i.e. it measures body temperature and heart rate,etc.
- To design a system to store the patient data over a period of time using cloud.
- To do analysis of collected data of sensors.

CHAPTER 3

REVIEW OF LITERATURE

1.) Patient-Monitoring Systems , Reed M. Gardner & M.Michael Shabot , Year 2014

To meet the increasing demands for more acute and intensive care required by patients with complex disorders, new organizational units— the ICUs—were established in hospitals beginning in the 1950s. The earliest units were simply postoperative recovery rooms used for prolonged stays after open-heart surgery. Intensive-care units proliferated rapidly during the late 1960s and 1970s. The types of units include burn, coronary, general surgery, open-heart surgery, pediatric, neonatal, respiratory, and multipurpose medical-surgical units. Today there are an estimated 75,000 adult, pediatric, and neonatal intensive care beds in the United States.

2.) IoT-Based Health Monitoring System for Active and Assisted Living , Ahmed Abdelgawad , School of Engineering and Technology, Central Michigan University, Mt. Pleasant, MI 48859, USA , Year 2017.

The Internet of Things (IoT) platform offers a promising technology to achieve the aforementioned healthcare services, and can further improve the medical service systems [1]. IoT wearable platforms can be used to collect the needed information ofthe user and its ambient environment and communicate such information wirelessly,where it is processed or stored for tracking the history of the user [2]. Such a connectivity with external devices and services will allow for taking preventive measure (e.g., upon foreseeing an upcoming heart stroke) or providing immediate care (e.g., when a user falls down and needs help). Recently, several IoT systems have been developed for IoT healthcare and assisted living applications.

3.) **IOT based health monitoring systems , Nayna Gupta & Sujata Pandey Year 2012**.

In this fast pace world, managing work and health simultaneously have become a matter of concern for most of the people. Long waiting hours at the hospitals or ambulatory patient monitoring are well known issues. The issues demands for a health monitoring system which can monitor the daily routine health parameters and heart rate monitoring seamlessly and can report the same to the concerned person with the help of GSM module.

CHAPTER 4

PROPOSED METHODOLOGY

Our system continuously monitoring patient's vital signs and sense abnormalities. The monitored data is delivered to medical staff. Upon encountering abnormalities, the system alerts the medical staff about the abnormal parameter. Thus, reduces the need for manual monitoring done by the medical staff. Our proposed system uses Arduino with esp8266 to send data from sensors to cloud platform that is thing speak. Arduino has been programed with ESP8266 module which includes the API key provided by on the things peak site. Any number users can see the medical record recorded on the thing speak using the thing speak access key.

MEASUREMENT OF RESPIRATORY RATE:

Thermister is used for the measurement of body temperature and respiratory temperature. This thermister is a passive transducer and it's resistance depends on the beat being applied on it. We have arranged the sensor in the potential divider circuit. This sensor exhibits a large change in resistance with a change in body temperature. The respiratory rate is determined by holding the sensor near the nose. The temperature sensor part is attached to the patient whose temperature has to be measured, which changes the values and thus the corresponding change in the temperature is displayed on the monitor graphically. Also all temperature measurements are updated in the patients database. Here in our project we use bead temperature sensor.

HEART BEAT MONITOR:

The patient's heart beat rate is monitored using photoelectric sensor which can sense the patient's pulse rate. This method of tracking the heart rate is more efficient than the traditional method which derives the same from ECG graph.

MEASUREMENT OF BODY TEMPERATURE

TEMPERATURE SENSOR (LM35) series are precision integrated circuit temperature sensors, whose output voltage is linearly proportional to the Celsius (Centigrade) temperature. The LM35 does not require any external calibration or trimming to provide typical accuracies of +-1/4 degree Celsius at room temperature and +-3/4 degree Celsius over a full -55 to +150 degree Celsius temperature range. Less to operates from 4 to 30 volt. Less than 60uA current drain.

PULSE SENSOR:

The Pulse Sensor is a plug-and-play heart-rate sensor for Arduino. It can be used by students, artists, athletes, makers, and game & mobile developers who want to easily incorporate live heart-rate data into their projects. The essence is an integrated optical amplifying circuit and noise eliminating circuit sensor. Clip the Pulse Sensor to your earlobe or fingertip and plug it into your Arduino, you can ready to read heart rate. Also, it has an Arduino demo code that makes it easy to use The pulse sensor has three pins: VCC, GND &Analog Pin.

WI-FI MODULE:

The ESP8266 wi-fi module is a self-contained SOC with incorporated TCP/IP protocol stack that can offer any controller access to wi-fi network. It uses 802.11 b/g/n protocols. Standby power consumption is less than 0.1mW.

An ESP8266 Wi-Fi module is a SOC microchip mainly used for the development of end-point IoT (Internet of things) applications. It is referred to as a standalone wireless transceiver, available at a very low price. It is used to enable the internet connection to various applications of embedded systems.

It can work as either a slave or a standalone application. If the ESP8266 Wi-Fi runs as a slave to a microcontroller host, then it can be used as a Wi-Fi adaptor to any type of microcontroller using UART or SPI. If the module is used as a standalone application, then it provides the functions of the microcontroller and Wi-Fi network.

The ESP8266 Wi-Fi module is highly integrated with RF balun, power modules, RF transmitter and receiver, analog transmitter and receiver, amplifiers, filters, digital baseband, power modules, external circuitry, and other necessary components.

BLOCK DIAGRAM

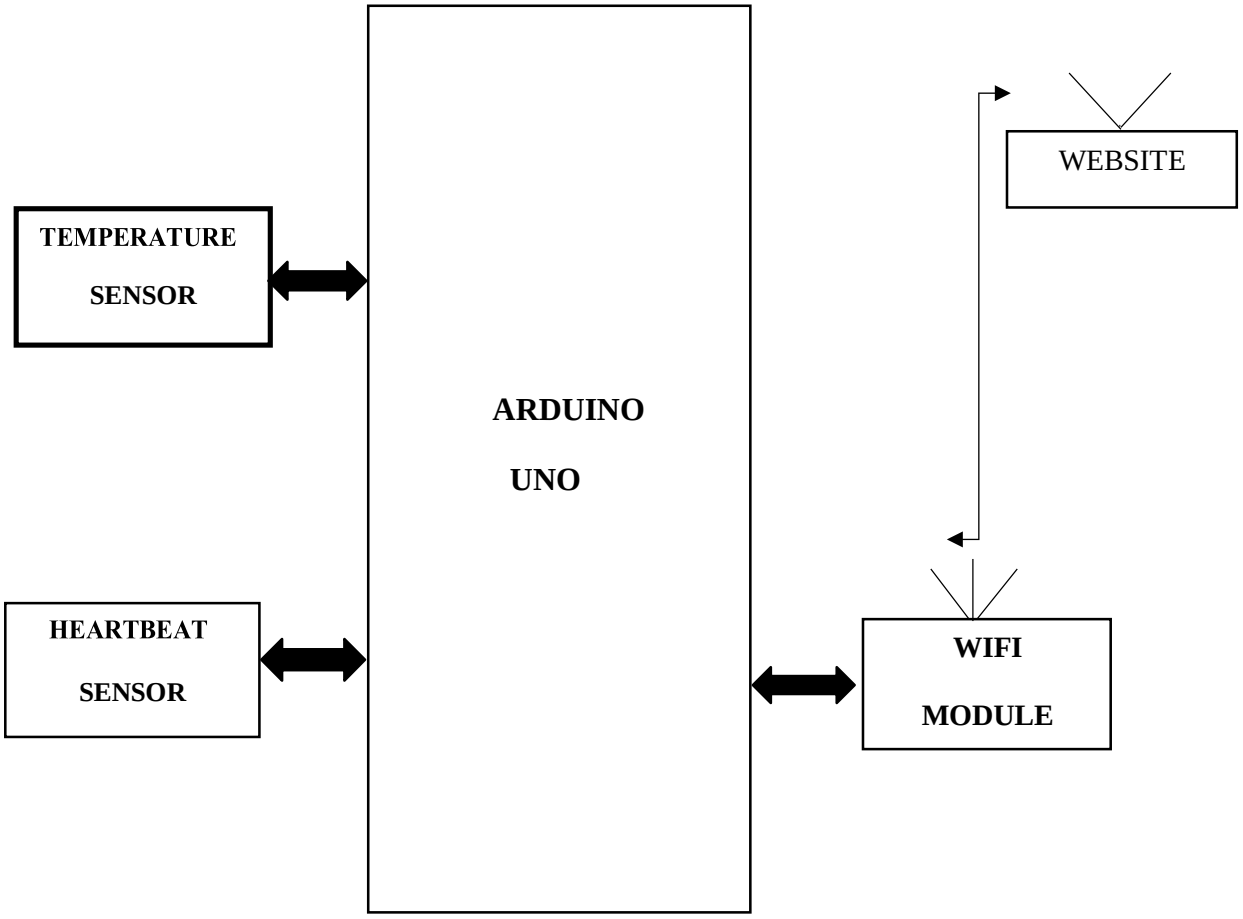

Fig .no. 4.1 Block Diagram

CHAPTER 5

TOOLS TO BE USED

5.1 Arduino Uno

The **Arduino Uno** is an open-source microcontroller board based on the Microchip ATmega328P microcontroller and developed by Arduino.cc. The board is equipped with sets of digital and analog input/output (I/O) pins that may be interfaced to various expansion boards (shields) and other circuits. The board has 14 digital I/O pins (six capable of PWM output), 6 analog I/O pins, and is programmable with the Arduino IDE (Integrated Development Environment), via a type B USB cable. It can be powered by the USB cable or by an external 9-volt battery, though it accepts voltages between 7 and 20 volts. It is similar to the Arduino Nano and Leonardo. The hardware reference design is distributed under a Creative Commons Attribution Share-Alike 2.5 license and is available on the Arduino website. Layout and production files for some versions of the hardware are also available.

The word "uno" means "one" in Italian and was chosen to mark the initial release of Arduino Software. The Uno board is the first in a series of USB-based Arduino boards; it and version 1.0 of the Arduino IDE were the reference versions of Arduino, which have now evolved to newer releases. The ATmega328 on the board comes preprogrammed with a bootloader that allows uploading new code to it without the use of an external hardware programmer.

While the Uno communicates using the original STK500 protocol, it differs from all preceding boards in that it does not use the FTDI USB-to-serial driver chip. Instead, it uses the Atmega16U2 (Atmega8U2 up to version R2) programmed as a USB-to-serial converter.

The power pins are as follows:-

● VIN. The input voltage to the Arduino board when it's using an external power source (as opposed to 5 volts from the USB connection or other regulated power source). You can supply voltage through this pin, or, if supplying voltage via the power jack, access it through this pin.

● 5V. The regulated power supply used to power the microcontroller and other components on the board. This can come either from VIN via an on-board regulator,or be supplied by USB or another regulated 5V supply.

● 3V3. A 3.3 volt supply generated by the on-board regulator. Maximum current draw is 50 mA.

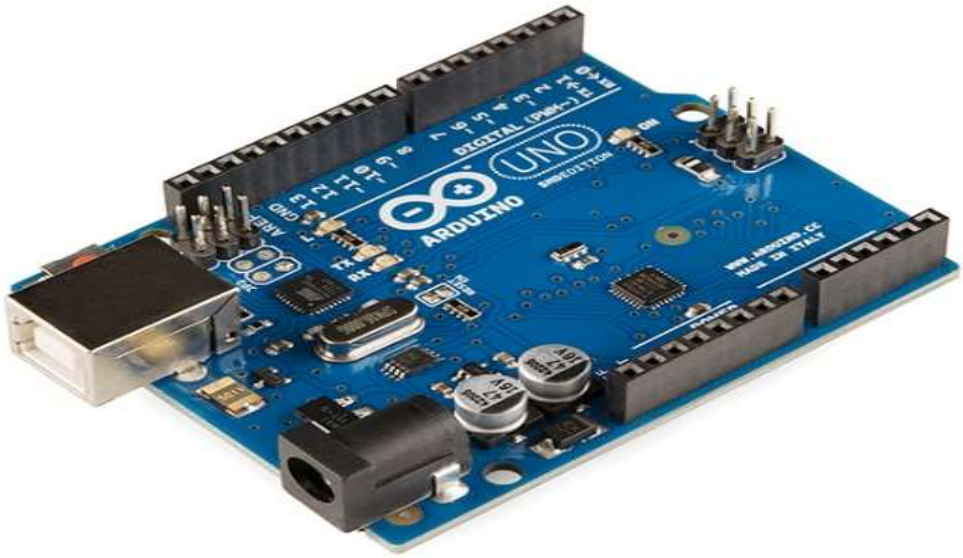

Fig.no.5.1 Arduino Uno

5.2 Wifi Module

ESP8266 is Wi-Fi enabled system on chip (SoC) module developed by Espressif system. It is mostly used for development of IoT (Internet of Things) embedded applications.

ESP8266 comes with capabilities of

- 2.4 GHz Wi-Fi (802.11 b/g/n, supporting WPA/WPA2),
- general-purpose input/output (16 GPIO),
- Inter-Integrated Circuit (I^2C) serial communication protocol,
- analog-to-digital conversion (10-bit ADC)
- Serial Peripheral Interface (SPI) serial communication protocol,
- I^2S (Inter-IC Sound) interfaces with DMA(Direct Memory Access) (sharing pins with GPIO),
- UART (on dedicated pins, plus a transmit-only UART can be enabled on GPIO2), and
- pulse-width modulation (PWM).

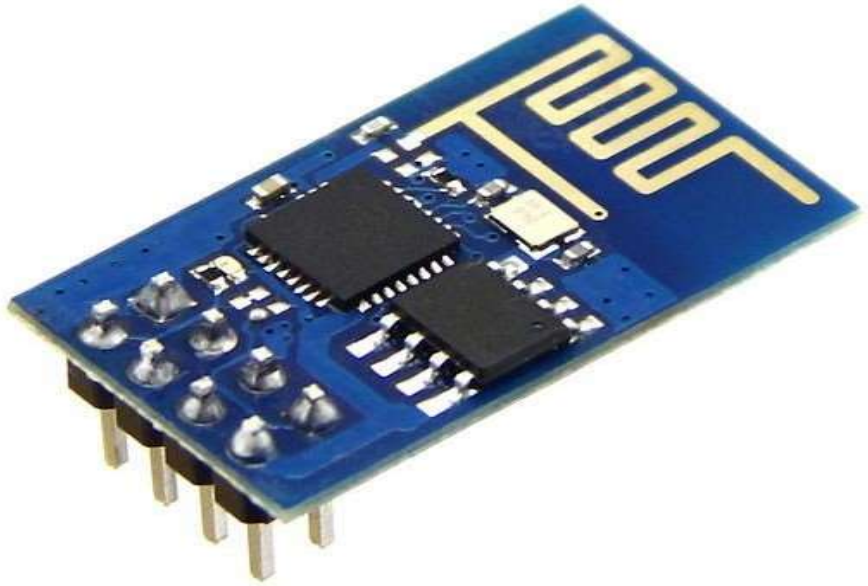

Fig.no.5.2 Wifi Module

5.3. Heartbeat Sensor

Heartbeat sensor provides a simple way to study the function of the heart which can be measured based on the principle of psycho-physiological signal used as a stimulus for the virtual reality system. The amount of the blood in the finger changes with respect to time. The sensor shines a light lobe (a small very bright LED) through the ear and measures the light that gets transmitted to the Light Dependent Resistor. The amplified signal gets inverted and filtered, in the Circuit. In order to calculate the heart rate based on the blood flow to the fingertip, a heartrate sensor is assembled with the help of LM358 OP-AMP for monitoring the heartbeat pulses.

Heartbeat Sensor is an electronic device that is used to measure the heart rate i.e. speed of the heartbeat. Monitoring body temperature, heart rate and blood pressure are the basic things that we do in order to keep us healthy.

Fig.no.5.3 Heartbeat Sensor

5.4. Temperature Sensor(DS18B20)

The DS18B20 digital thermometer provides 9-bit to 12-bit Celsius temperature measurements and has an alarm function with nonvolatile user-programmable upper and lower trigger points. The DS18B20 communicates over a 1-Wire bus that by definition requires only one data line (and ground) for communication with a central microprocessor. In addition, the DS18B20 can derive power directly from the data line ("parasite power"), eliminating the need for an external power supply.

Each DS18B20 has a unique 64-bit serial code, which allows multiple DS18B20s to function on the same 1-Wire bus. Thus, it is simple to use one microprocessor to control many DS18B20s distributed over a large area. Applications that can benefit from this feature include HVAC environmental controls, temperature monitoring systems inside buildings, equipment, or machinery, and process monitoring and control systems.

Key Features

- Unique 1-Wire® Interface Requires Only One Port Pin for Communication
- Reduce Component Count with Integrated Temperature Sensor and EEPROM
- Measures Temperatures from -55°C to +125°C (-67°F to +257°F)

- ±0.5°C Accuracy from -10°C to +85°C
- Programmable Resolution from 9 Bits to 12 Bits
- No External Components Required
- Parasitic Power Mode Requires Only 2 Pins for Operation (DQ and GND)
- Simplifies Distributed Temperature-Sensing Applications with Multidrop Capability
- Each Device Has a Unique 64-Bit Serial Code Stored in On-Board ROM
- Flexible User-Definable Nonvolatile (NV) Alarm Settings with Alarm Search Command Identifies Devices with Temperatures Outside Programmed Limits
- Available in 8-Pin SO (150 mils), 8-Pin µSOP, and 3-Pin TO-92 Packages

Fig.no.5.4 Temperature Sensor

Chapter 6

PRACTICAL IMPLEMENTATION

6.1 Circuit Diagram

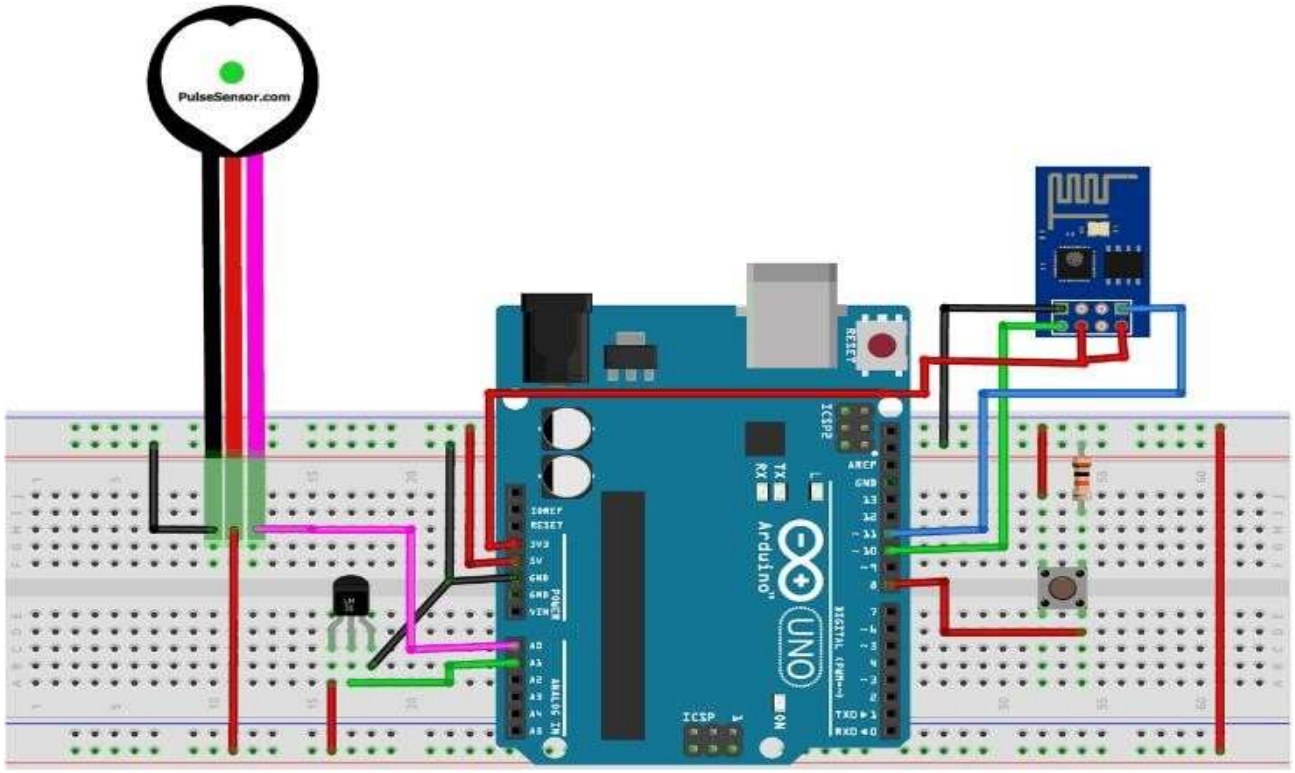

Fig.no. 6.1: Circuit Diagram

In this, we have used Arduino as microcontroller unit. All the components and modules are connected as give in a circuit diagram.

For this system we required two types of power supply 5V or 3.3 V because of some components are operated in 3.3 V. All the sensors, which generates the analog output is connected to an analog pin of Arduino's analog pins. And Digital sensors like (Temp.,) there are connected to digital pins. The Ethernet Shield (HANRUN) is used in this system for internet connectivity. It can be replaced by GSM or Wi-Fi module also.

The analog/digital data are processed by Arduino and with the help of internet connectivity all the monitored data sended to the cloud (Here we have used ThingSpeak). ThingSpeak cloud provides the MATLAB Visualization or MATLAB Code. By this we can plot graph and getting desirable data.

6.2 Schematic designed using EasyEDA software

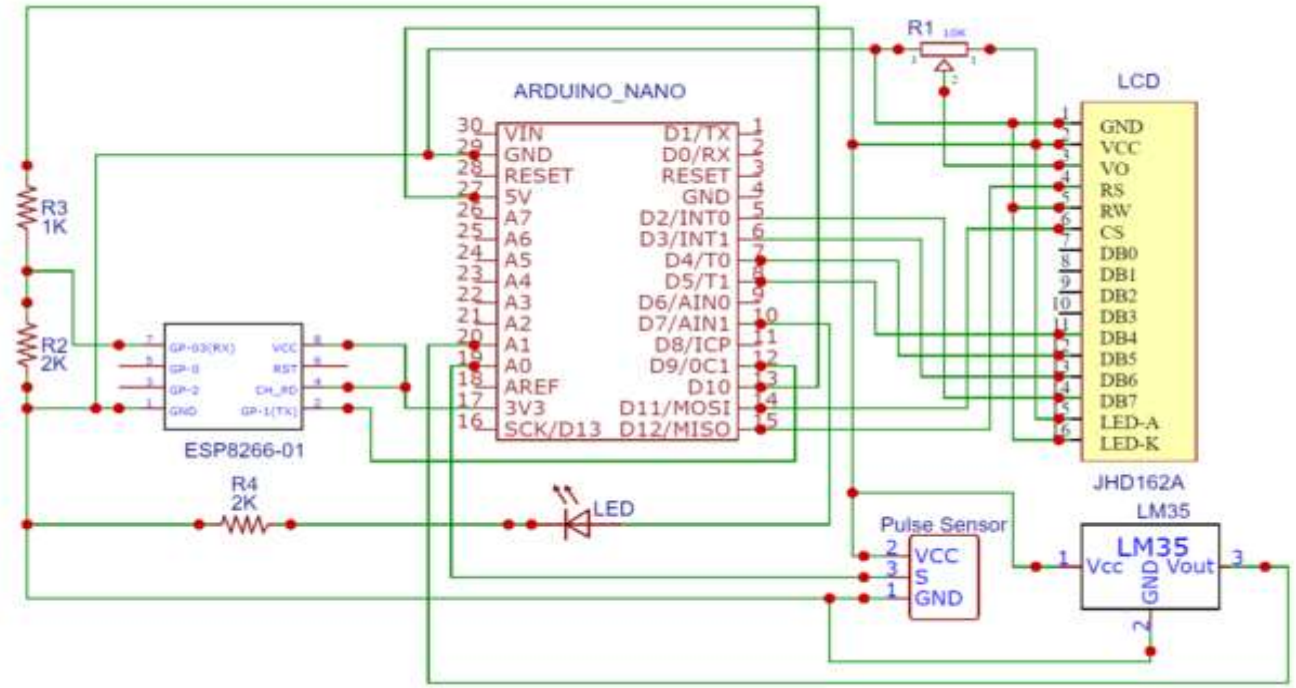

Fig.no. 6.2: Schematic designed using EasyEDA software

6.3 Interfacing ESP8266 With Arduino Uno

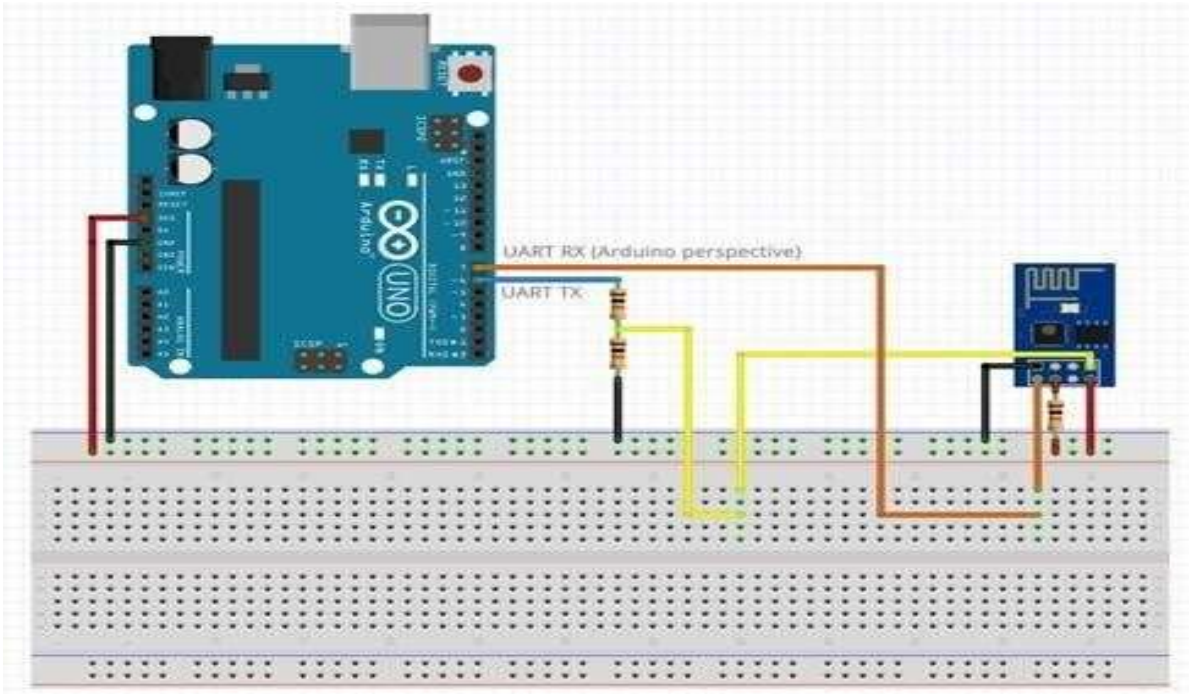

Figure 6.3: ESP interfacing with Arduino

ESP826 can be interfaced with Arduino, although the logic connection between ESP and Arduino is very simple. ESP-Rx goes to Arduino Tx, ESP-Tx goes to Arduino Rx. However, all ESP-8266 run on 3.3V, while Arduino pins run on 5V. Before connecting them, you shall provide a way to adapt these voltages, or you could damage your ESP. Given below the circuit showing interface between ESP and Arduino.
Following is steps needed to take care while interfacing

- VCC shall be connected to the 3.3V power supply.
- GPIO0 and GPIO2 are general purposes digital ports. GPIO0 also controls the module mode (programming or normal operation). In our case (normal operation), it shall be connected to 3.3V (high). We put it on 3.3V to simplify the connections
- CH_PD: Chip enable. Keeping it on high (3.3V) for normal operation
- RST: Reset. Keeping it on high (3.3V) for normal operation. Put it on 0V to reset the chip.
- Tx: Goes to Arduino Rx
- Rx: Goes to Arduino Tx (But needs a voltage adjusting) and GND is ground

6.4 Cloud computing:

Cloud computing, also known as on-the-line computing, is a kind of Internet-based computing that provides shared processing resources and data to computers and other devices on demand. It is a model for enabling ubiquitous, on-demand access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications and services), which can be rapidly provisioned and released with minimal management effort. Cloud computing and storage solutions provide users and enterprises with various capabilities to store and process their data in third-party data centers. It relies on sharing of resources to achieve coherence and economy of scale, similar to a utility (like the electricity grid) over a network.

Cloud storage:

Cloud storage is a model of data storage in which the digital data is stored in logical pools, the physical storage spans multiple servers (and often locations), and the physical environment is typically owned and managed by a hosting company. These cloud storage providers are responsible for keeping the data available and accessible, and the physical environment protected and running. People and organizations buy or lease storage capacity from the providers to store user, organization, or application data.

- https://thingspeak.com
- Send sensor data to the cloud

ThingSpeak Channel:

Channels store all the data that a ThingSpeak application collects. Each channel includes eight fields that can hold any type of data, plus three fields for location data and one for status data. Once we collect data in a channel, we can use ThingSpeak apps to analyze and visualize it.

Channel Settings

- Channel Name: Enter a unique name for the ThingSpeak channel.
- Description: Enter a description of the ThingSpeak channel.
- Field#: Check the box to enable the field, and enter a field name. Each ThingSpeak channel can have up to 8 fields.
- Metadata: Enter information about channel data, including JSON, XML, or CSV data.
- Tags: Enter keywords that identify the channel. Separate tags with commas.
- Latitude: Specify the position of the sensor or thing that collects data in decimal degrees. For example, the latitude of the city of London is 51.5072.
- Longitude: Specify the position of the sensor or thing that collects data in decimal degrees. For example, the longitude of the city of London is -0.1275.
- Elevation: Specify the position of the sensor or thing that collects data in meters. For example, the elevation of the city of London is 35.052.
- Make Public: If you want to make the channel publicly available, check this box.
- URL: If you have a website that contains information about your ThingSpeak channel, specify the URL.
- Video ID: If you have a YouTube™ or Vimeo® video that displays your channel information, specify the full path of the video URL.

Using the Channel

We can get data into a channel from a device, website, or another ThingsSpeak channel. You can then visualize data and transform it using ThingSpeak Apps.

ü Channel detail:-

Channel name-Health care system

Channel ID: 125XXX

Author: acd@gmail.com

Access: Private, Public

6.5 Using the Arduino IDE

The Arduino integrated development environment (IDE) is a cross-platform application written in Java, and is derived from the IDE for the Processing programming language and the Wiring projects. It is designed to introduce programming to artists and other newcomers unfamiliar with software development. It includes a code editor with features such as syntax highlighting, brace matching, and automatic indentation, and is also capable of compiling and uploading programs to the board with a single click. A program or code written for Arduino is called a "sketch".

Arduino programs are written in C or C++. The Arduino IDE comes with a software library called "Wiring" from the original Wiring project, which makes many common input/output operations much easier. Users only need define two functions to make a run able cyclic executive program:

- Setup(): a function run once at the start of a program that can initialize settings
- Loop(): a function called repeatedly until the board powers off.

Open the Arduino IDE software and select the board in use. To select the board:

- Go to Tools.
- Select Board.
- Under board, select the board being used, in this case Arduino Uno.
- Go to Tools and to Port and select the port at which the arduino board is connected

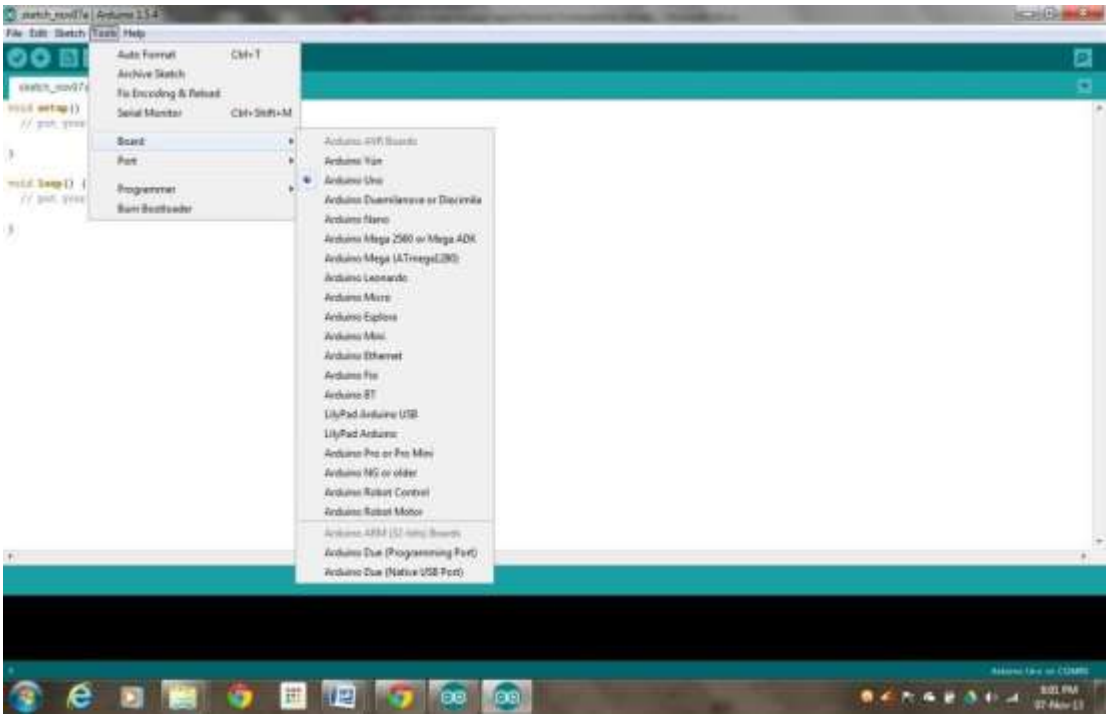

Fig.no 6.5: Arduino IDE

6.6 Using Thingspeak cloud service

ThingSpeaK is an IoT analytics platform service that allows you to aggregate, visualize and analyze live data streams in the cloud. ThingSpeak provides instant visualizations of data posted by your devices to ThingSpeak. With the ability to execute MATLAB® code in ThingSpeak you can perform online analysis and processing of the data as it comes in. ThingSpeak is often used for prototyping and proof of concept IoT systems that require analytics.

What is IoT?

Internet of Things (IoT) describes an emerging trend where a large number of embedded devices (things) are connected to the Internet. These connected devices communicate with people and other things and often provide sensor data to cloud storage and cloud computing resources where the data is processed and analyzed to gain important insights. Cheap cloud computing power and increased device connectivity is enabling this trend.

IoT solutions are built for many vertical applications such as environmental monitoring and control, health monitoring, vehicle fleet monitoring, industrial monitoring and control, and home automation.

At a high level, many IoT systems can be described using the diagram below:

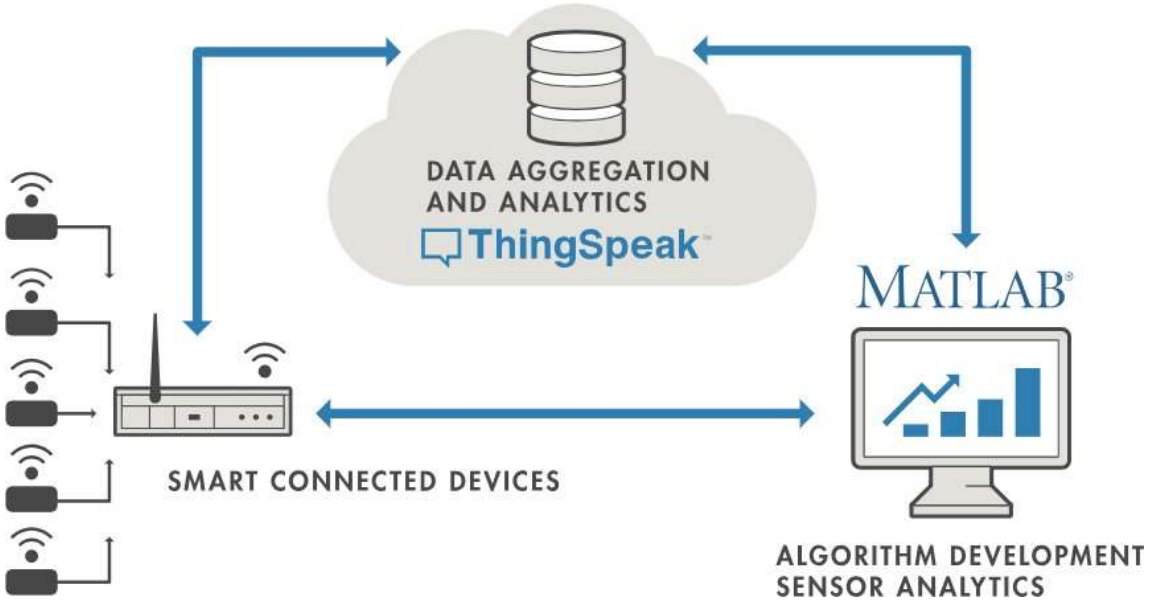

Fig.no. 6.6 : IOT & its applications

On the left, we have the smart devices (the "things" in IoT) that live at the edge of the network. These devices collect data and include things like wearable devices, wireless temperatures sensors, heart rate monitors, and hydraulic pressure sensors, and machines on the factory floor.

In the middle, we have the cloud where data from many sources is aggregated and analyzed in real time, often by an IoT analytics platform designed for this purpose.

The right side of the diagram depicts the algorithm development associated with the IoT application. Here an engineer or data scientist tries to gain insight into the collected data by performing historical analysis on the data. In this case, the data is pulled from the IoT platform into a desktop software environment to enable the engineer or scientist to prototype algorithms that may eventually execute in the cloud or on the smart device itself.

An IoT system includes all these elements. ThingSpeak fits in the cloud part of the diagram and provides a platform to quickly collect and analyze data from internet connected sensors.

ThingSpeak Key Features:-

ThingSpeak allows you to aggregate, visualize and analyze live data streams in the cloud. Some of the key capabilities of ThingSpeak include the ability to:

- Easily configure devices to send data to ThingSpeak using popular IoT protocols.
- Visualize your sensor data in real-time.
- Aggregate data on-demand from third-party sources.
- Use the power of MATLAB to make sense of your IoT data.
- Run your IoT analytics automatically based on schedules or events.
- Prototype and build IoT systems without setting up servers or developing web software.
- Automatically act on your data and communicate using third-party services like Twilio® or Twitter®.

6.7 Code

```
#define USE_ARDUINO_INTERRUPTS true
#define DEBUG true
#define SSID "krishna"          // "SSID-WiFiname"
#define PASS "12345678"   // "password"
#define IP "184.106.153.149"      // thingspeak.com ip
```

```
#include <SoftwareSerial.h>
#include "Timer.h"
#include <PulseSensorPlayground.h>     // Includes the PulseSensorPlayground Library.
Timer t;
PulseSensorPlayground pulseSensor;

String msg = "GET /update?key=your api key";
SoftwareSerial esp8266(10,11);

//Variables
const int PulseWire = A0;       // PulseSensor PURPLE WIRE connected to ANALOG PIN 0
const int LED13 = 13;          // The on-board Arduino LED, close to PIN 13.
int Threshold = 550;           //for heart rate sensor
float myTemp;
int myBPM;
String BPM;
String temp;
int error;
int panic;
int raw_myTemp;
float Voltage;
float tempC;
void setup()
{

  Serial.begin(9600);
  esp8266.begin(115200);
  pulseSensor.analogInput(PulseWire);
  pulseSensor.blinkOnPulse(LED13);       //auto-magically blink Arduino's LED with heartbeat.
  pulseSensor.setThreshold(Threshold);

  // Double-check the "pulseSensor" object was created and "began" seeing a signal.
   if (pulseSensor.begin()) {
    Serial.println("We created a pulseSensor Object !");  //This prints one time at Arduino power-up,  or
on Arduino reset.
  }
```

```
  Serial.println("AT");
  esp8266.println("AT");

  delay(3000);

  if(esp8266.find("OK"))
  {
    connectWiFi();
  }
  t.every(10000, getReadings);
   t.every(10000, updateInfo);
}

void loop()
{
  panic_button();
start: //label
    error=0;
   t.update();
    //Resend if transmission is not completed
    if (error==1)
    {
      goto start; //go to label "start"
    }
 delay(4000);
}

void updateInfo()
{
  String cmd = "AT+CIPSTART=\"TCP\",\"";
  cmd += IP;
  cmd += "\",80";
  Serial.println(cmd);
  esp8266.println(cmd);
  delay(2000);
  if(esp8266.find("Error"))
  {
```

```
    return;
  }
  cmd = msg ;
  cmd += "&field1=";    //field 1 for BPM
  cmd += BPM;
  cmd += "&field2=";  //field 2 for temperature
  cmd += temp;
  cmd += "\r\n";
  Serial.print("AT+CIPSEND=");
  esp8266.print("AT+CIPSEND=");
  Serial.println(cmd.length());
  esp8266.println(cmd.length());
  if(esp8266.find(">"))
  {
    Serial.print(cmd);
    esp8266.print(cmd);
  }
  else
  {
    Serial.println("AT+CIPCLOSE");
    esp8266.println("AT+CIPCLOSE");
    //Resend...
    error=1;
  }
}

boolean connectWiFi()
{
  Serial.println("AT+CWMODE=1");
  esp8266.println("AT+CWMODE=1");
  delay(2000);
  String cmd="AT+CWJAP=\"";
  cmd+=SSID;
  cmd+="\",\"";
  cmd+=PASS;
  cmd+="\"";
  Serial.println(cmd);
```

```
  esp8266.println(cmd);
  delay(5000);
  if(esp8266.find("OK"))
  {
   return true;
  }
  else
  {
   return false;
  }
}

void getReadings(){
  raw_myTemp = analogRead(A1);
  Voltage = (raw_myTemp / 1023.0) * 5000; // 5000 to get millivots.
  tempC = Voltage * 0.1;
  myTemp = (tempC * 1.8) + 32; // conver to F
  Serial.println(myTemp);
  int myBPM = pulseSensor.getBeatsPerMinute();  // Calls function on our pulseSensor object that returns BPM as an "int".
                                                // "myBPM" hold this BPM value now.
if (pulseSensor.sawStartOfBeat()) {            // Constantly test to see if "a beat happened".
Serial.println(myBPM);                         // Print the value inside of myBPM.
}

  delay(20);
    char buffer1[10];
     char buffer2[10];
    BPM = dtostrf(myBPM, 4, 1, buffer1);
    temp = dtostrf(myTemp, 4, 1, buffer2);
  }

void panic_button(){
  panic = digitalRead(8);
    if(panic == HIGH){
    Serial.println(panic);
      String cmd = "AT+CIPSTART=\"TCP\",\"";
```

```
  cmd += IP;
  cmd += "\",80";
  Serial.println(cmd);
  esp8266.println(cmd);
  delay(2000);
  if(esp8266.find("Error"))
  {
   return;
  }
  cmd = msg ;
  cmd += "&field3=";
  cmd += panic;
  cmd += "\r\n";
  Serial.print("AT+CIPSEND=");
  esp8266.print("AT+CIPSEND=");
  Serial.println(cmd.length());
  esp8266.println(cmd.length());
  if(esp8266.find(">"))
  {
   Serial.print(cmd);
   esp8266.print(cmd);
  }
  else
  {
   Serial.println("AT+CIPCLOSE");
   esp8266.println("AT+CIPCLOSE");
   //Resend...
   error=1;
  }
 }
}
```

Chapter 7

RESULTS & DISCUSSIONS

This system can be used to transmit the patient vital parameter information in real-time to remote location and can be seen by the care taker. The sensors are connected to the Arduino Mega board. The sensed values are transmitted wirelessly to the Arduino board receiver which is connected to the central station personal computer. The captured data can be seen over the ThingSpeak Cloud service using a unique username and password. The following are some results from the system.

Fig.no.7 Graphs of Sensor Output on the thing speak

Chapter 8

CONCLUSION AND FUTURE ENHANCEMENTS

We've demonstrated a smart sensor system that captures a variety of physiological data and continuously monitors IoT, allowing patients to manage their data in real time. This device includes nonintrusive, high-accuracy sensors such as an ECG and a temperature sensor. Furthermore, by eliminating the need for intrusive equipment or the usage of a laptop to see biological data, it simplifies the problems of wearable technology, allowing patients to have significantly less limits. This smart sensor system might be integrated with a wide-area network system in the future to give physicians and healthcare workers access to patient data for more accurate and speedier diagnosis and treatment options.

The Patient Health Care system utilizes these concepts to come up with a system for better quality of life for people in society. From an engineering perspective, the project has seen concepts acquired through the computer science and embedded study period being practically applied. The Electric circuit analysis knowledge was used during design and fabrication of the individual modules. Electromagnetic fields analysis used in the wireless transmission between microcontrollers and Software programming used during programming of the microcontrollers to come up with a final finished circuit system.

The whole health monitoring system, which we have proposed can be integrated into a small compact unit as small as a cell phone or a wrist watch. This will help the patients to easily carry this device with them wherever they go. The VLSI technologies will greatly come handy in this regard. In this paper, tele-monitoring application is presented which allows the doctor to view the patient’s vital parameters remotely and dynamically in a Web page in real time and doesn’t need to have any special requirement on the PC; all he needs is an Internet access.

For the patient side, a home based Arduino IDE application which is embedded in home PC is required. In future this work can be extended by adding the Blood pressure sensors to the existing set-up. This work is done based on single person’s data collection and in future this can be extended to multiple people.

Using the Arduino Uno and the WiFi module project, we may add a GPS module to IoT patient monitoring. Using the longitude and latitude obtained, this GPS module will determine the patient's position or location. The Wi-Fi module will then broadcast this position to the cloud, which is the IOT. Doctors can then determine the patient's status in case they need to take any preventative measures.

Chapter 9

REFERENCES

- S.H. Almotiri, M. A. Khan, and M. A. Alghamdi. Mobile health (m- health) system in the context of IoT. In 2016 IEEE 4th International Conference on Future Internet of Things and Cloud Workshops (FiCloudW), pages 39–42, Aug 2016.
- Gulraiz J. Joyia, Rao M. Liaqat, Aftab Farooq, and Saad Rehman, Internet of Medical Things (IOMT): Applications, Benefits and Future Challenges in Healthcare Domain, Journal of Communications Vol. 12, No. 4, April 2017
- B. G. Ahn, Y. H. Noh, and D. U. Jeong. Smart chair based on multi heart rate detection system. In 2015 IEEE SENSORS, pages 1–4, Nov 2015.
- A. Shah and B. Guru, "Poverty in Remote Rural Areas in India: A Review of Evidence and Issues", *SSRN Electron. J.*, pp. 1-58, 2012.
- A. Mdhaffar, T. Chaari, K. Larbi, M. Jmaiel and B. Freisleben, "IoT-based health monitoring via LoRaWAN", *17th IEEE International Conference on Smart Technologies EUROCON 2017 - Conference Proceedings*, pp. 519-524, 2017.
- H. Al-Hamadi and I. R. Chen, "Trust-Based Decision Making for Health IoT Systems", *IEEE Internet Things J.*, vol. 4, no. 5, pp. 1408-1419, Oct. 2017.
- M. Thangaraj, P. P. Ponmalar and S. Anuradha, "Internet of Things (IOT) enabled smart autonomous hospital management system - A real world health care use case with the technology drivers", *2015 IEEE International Conference on Computational Intelligence and Computing Research ICCIC 2015*, 2016.
- Y. YIN, Y. Zeng, X. Chen and Y. Fan, "The internet of things in healthcare: An overview", *J. Ind. Inf. Integr.*, vol. 1, pp. 3-13, 2016.
- Shubham Banka, Isha Madan and S.S. Saranya, Smart Healthcare Monitoring using IoT. International Journal of Applied Engineering Research ISSN 0973-4562 Volume 13, Number 15, pp. 11984-11989, 2018.
- K. Perumal, M. Manohar, A Survey on Internet of Things: Case Studies, Applications, and Future Directions, In Internet of Things: Novel Advances and Envisioned Applications, Springer International Publishing, (2017) 281-297.
- S.M. Riazulislam, Daehankwak, M.H.K.M.H., Kwak, K.S.: The Internet of Things for Health Care: A Comprehensive Survey. In: IEEE Access (2015)