

HEALTH CARE MONITORING SYSTEM USING IOT

Submitted in partial fulfillment of the requirements for
the award of
Bachelor of Engineering Degree in Computer Science and Engineering

by

KUNAL R S B V RAGHAVA RAJU L (37110393)



**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
SCHOOL OF COMPUTING**

SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY

(DEEMED TO BE UNIVERSITY)

Accredited with Grade “A” by NAAC

**JEPPIAAR NAGAR, RAJIV GANDHI SALAI,
CHENNAI – 600119.**

MARCH 2021



SATHYABAMA
INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)
Accredited with Grade “A” by NAAC
JEPPIAAR NAGAR, RAJIV GANDHI SALAI,
Chennai - 600119
www.sathyabama.ac.in



DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

BONAFIDE CERTIFICATE

This is to certify that this Project Report is the bonafide work of **KUNAL R S B V RAGHAVA RAJU L (37110393)** who carried out the project entitled “**HEALTH CARE MONITORING SYSTEM USING IOT**” under our supervision from November 2020 to April 2021.

Internal Guide

Dr. A. PRAVI N M.E., Ph.D.,

Head of the Department

Dr. S. VIGNESHWARI M.E., Ph.D.,

Submitted for Viva voce Examination held on_____

Internal Examiner

External Examiner

DECLARATION

I, **KUNAL R S B V RAGHAVA RAJU L (37110393)** hereby declare that the Project Report entitled "**HEALTH CARE MONITORING SYSTEM USING IOT**" done by me under the guidance of **Dr. A. PRAVIN M.E., Ph.D.,** at Sathyabama Institute of Science and technology is submitted in partial fulfillment of the requirements of the award of Bachelor of Engineering degree in Computer Science and Engineering.

DATE:

PLACE:

SIGNATURE OF THE CANDIDATE

ACKNOWLEDGEMENT

I are pleased to acknowledge our sincere thanks to Board of management of **SATHYABAMA** for their kind encouragement in doing this project and for completing it successfully. I am grateful to them.

I convey our thanks to **Dr. T. SASIKALA M.E., Ph.D., Dean, School of Computing, Dr. S. VIGNESHWARI M.E., Ph.D. and Dr. L. LAKSHMANAN M.E., Ph.D., Heads of the Department,** Department of Computer Science and Engineering for providing us the necessary support and details at the right time during the progressive reviews.

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

I wish to express our thanks to all Teaching and Non-teaching staff members of the Department of **COMPUTER SCIENCE AND ENGINEERING** who were helpful in many ways for the completion of the project.

ABSTRACT

Technology plays the foremost vital role in aid not just for sensory devices however besides in communication, recording and show device. it's important to watch numerous medical parameters and post operational days. thus, the most recent development in aid communication methodology mistreatment IOT is customized.

Internet of things is a catalyst for the aid and plays distinguished role in wide selection of aid applications. during this project, the Nodemcu microcontroller is employed as an entree to speak to the varied detectors like temperature sensor, Respiration detector, Blood gas level and heartbeat detector.

The microcontroller picks up the detector knowledge and sends it to the network through IOT and thus provides real time observance of the health care parameters for doctors. the info is often access anytime by the doctor. The controller is additionally connected with buzzer to alert the caretaker concerning variation in detector output. apart from the most issue in remote patient observance system is that the info on be firmly transmitted to the destination finish and provision is formed to permit solely licensed user to access the info.

The security drawback is to be self-addressed by sending the info through the countersign protected web will which is able to} be encrypted by customary AES128 and the Doctors can access the records by work to the hypertext mark-up language webpage. Next to the instance of extremity alert message is sent to the doctor through IOT connected to the controller. For this reason, fast tentative medication is often simply done by this method. this method is capable with low power consumption capability, easy setup, high performance, and time to time response.

TABLE OF CONTENTS

CHAPTER NO	TITLE	PAGE NUMBER
	ABSTRACT	V
	LIST OF FIGURES	VIII
1	INTRODUCTION	1
2	LITERATURE SURVEY	2
3	SYSTEM DESIGN AND MODULE DISCRIPTION	6
	3.1 EXISTING SYSTEM	6
	3.1.1 DRAWBACKS OF EXISTING SYSTEM	6
	3.2 PROPOSED SYSTEM	6
	3.2.1 ADVANTAGES OF PROPOSED SYSTEM	7
	3.3 SYSTEM REQUIREMENTS	7
	3.3.1 HARDWARE REQUIRMENTS	6
	3.3.2 SOFTWARE REQUIRMENTS	7
	3.4 SOFTWARE DESCRIPTION	7
	3.4.1 ARDUINO IDE	7
	3.4.2 UBIDOTS	9
	3.5 NODEMCU	9
	3.5.1 CIRCUIT DIAGRAM DESCRIPTION	12
	3.5.2 CURCUIT DIAGRAM OF POWER SUPPLY	13
	3.6 HEARTBEAT SENSOR	13
	3.6.1 WORKING OF A HEARTBEAT SENSOR	15
	3.6.2 PRACTICAL HEARTBEAT SENSOR	16
	3.7 TEMPERATURE SENSOR	17
	3.8 PRESSURE SENSOR	19
	3.9 HUMIDITY SENSOR	20
	3.10 DHT11 SENSOR	20
4	IMPLIMENTATION	23
	4.1 OVERVIEW OF EMBEDDED SYSTEMS	23
	4.2 BLOCK DIAGRMA OF AN EMBEDDED SYSTEM	24
	4.2.1 CHARACTERISTICS OF EMBEDDED SYSTEM	25

	4.2.2 POWER SUPPLY	28
	4.3 TRANSFORMER	29
	4.3.1 RECTIFIER	29
	4.3.2 FILTERS	29
	4.3.3 REGISTERS	30
	4.3.4 REGULATORS	30
	4.4 DATA COLLECTION MODULE	30
5	RESULTS AND SCREENSHOTS	31
6	CONCLUSION AND FUTURE ENHANCEMENTS	34
	6.1 CONCLUSION	34
	6.2 FUTURE ENHANCEMENT	34
	REFERENCES	35
	APPENDIX - SAMPLE CODING	36

LIST OF FIGURES

S.NO	TITLE	PAGE NO.
3.1	ARDUINO IDE	8
3.2	UBIDOTS	9
3.3	NODEMCU	12
3.4	POWER SUPPLY OF NODEMCU	12
3.5	CIRCUIT DIAGRAM	13
3.6	HEART RATE SESNOR PRINCIPLE	14
3.7	CIRCUITE DIAGRAM OF SENSOR	15
3.8	PULSE SENSOR	16
3.9	TEMPARATURE SENSOR	18
3.10	PRESSURE SENSOR	19
3.11	DHT11 SENSOR	21
4.1	BLOCK DIAGRAM OF EMBEDDED SYSTEM	25
4.2	EMBEDDED SYSTEMS HAVING HETEROGENEOUS ARCHITECTURES	27
4.3	GENERAL BLOCK OF POWER SUPPLY UNIT	29
5.1	THE HARDWARE PART	31
5.2	WEBSITE IMAGE	31
5.3	PHONE TRACKING IMAGE	32
5.4	NOTIFICATION SENDING PART	32
5.5	GMIAL NOTIFICATION MESSAGE	33

Chapter 1

Introduction:

- It presents a prototype machine-to-machine (M2M) healthcare solution that combines mobile and IPv6 techniques in a wireless sensor network to monitor the health condition of patients and provide a wide range of effective, comprehensive, and convenient healthcare services.
- A low-power embedded wearable sensor measures the health parameters dynamically, and is connected, according to the concept of IPv6 over low-power wireless personal area network, to the M2M node for wireless transmission through the internet or external IP-enabled networks via the M2M gateway.
- A visualization module of the server program graphically displays the recorded biomedical signals on Android mobile devices used by patients and doctors at the end of the networks in real-time. Our approach for a global M2M healthcare solution is managed to process the large amount of biomedical signals through the extended network combining IPv6 technique and mobile technology for daily lifestyle to users appropriately.
- It presents a working model which incorporates sensors to measure all these parameters like body temperature and Heartbeat rate and transfer it to the ARM 11 SMARTPHONE, so that the patient condition can be analyzed to by doctors in any part of the hospital wherever they are.
- Thus it reduces doctors work load and also gives more accurate results, wherever there is an abnormality felt by the patient, we have also incorporated saline monitoring system which gives an alarm when the saline bottle about to empty.
- The experimental setup can be operated for monitoring from anywhere covered by the Cellular (GSM) service by exchanging SMS messages with the remote mobile device. At the consultation unit, a dedicated application software is required to manage the follow of SMS messages from the mobile and display the temperature and heart beat of the patient.

Chapter 2

LITERATURE SURVEY:

1. Title: Remote patient monitoring system

Author: Narasimha Rao Jasti Madhu

Year: (2010)

It provides the image-based system which acquires the ECG signal via digital camera; this information is performed on the tool like MATLAB and data sending through the internet network and stored in database. Then the original image is then availed to the doctor via Android mobiles. The purpose of this system is the vital signs and parameters from the ICU monitoring system and makes this data to be available to the doctor who may not be in the hospital and in the country. In case of any abnormality, the doctor is alerted by sending a notification from C2DM server to his mobile. The drawback of this paper is that, due to the slow internet connection the data will not be send to the doctor which is located remotely. The image is captured through the camera, which must be HD which cost a lot. a notification from C2DM server to his mobile. the downside of this paper is that, because of the slow net association the information won't be send to the doctor that is found remotely. The image is captured through the camera, that should be HD that price a great deal.

2. Title: Design of Family Health Care Monitoring System

Author: R.Aravind , Musthak Ahme

Year: (2013).

Health observation is repeatedly mentioned united of the most application areas for Pervasive computing. Mobile Health Care is that the integration of mobile computing and health observation. it's the appliance of mobile computing technologies for up communication among patients, physicians, and different health care employees. As mobile devices became AN indivisible a part of our life it will integrate health care a lot of seamlessly to our everyday life. It permits the delivery of correct medical data anytime anyplace by suggests that of mobile devices. Recent technological advances in sensors, low-power integrated circuits, and wireless communications have enabled the planning of cheap , miniature, light-weight and intelligent bio-sensor nodes. These nodes, capable of sensing, processing, and

communicating one or a lot of important signs, is seamlessly integrated into wireless personal or body space networks for mobile health observation. This paper, presents a mechanism for estimation of old well-being condition supported usage of household appliances connected through varied sensing units. 2 new eudaemonia functions are outlined to determine the standing of the old persons on activity essential daily activities. The developed system for monitoring and analysis of essential daily activities was tested at the homes of 4 totally different old persons living alone and also the results are encouraging in determinative eudaemonia of the old.

3. Title: Design and Implementation of Wearable ECG System

Author: Sub Choi

Year: (2013)

This system describes the planning and implementation a part of wearable graph with the good phone for the important time watching of health. during this system good shirt are developed with graph sensors and might be worn by any variety of patient for watching his or her health in real time and obtain needed treatment or prescription. These systems are principally developed considering elder individuals in mind as they live alone in their homes. thus, this method primarily monitors the senior individuals for self-diagnosing purpose. The results of this method were the system might monitor and diagnose patients' heart conditions in real time, after they wear a sports-shirt with a graph detector in it. additionally, to the current, the system conjointly provides graphical data with history management tools and an automatic emergency decision system to the patient to urge the desired treatment in time. the disadvantage of the system is that it solely concentrates on elder individuals and it includes shirt (ECG sensor) for carrying that price heaps.

4. Title: Smart Elderly Home Monitoring System with an Android Phone

Author: S.Selvakanmani

Year: (2015)

Here it is considers or takes into consideration bound facts, that square measure attack and stroke as they're the foremost reason behind hospitalization of the elder folks. there's a lot of probabilities of survival if the older folks get the treatment inside associate hour. it had conjointly been developed. associate robot good phone

with measuring device is employed to discover a fall of the carrier, and this robot device is understood as tending device. The robot phone is then connected to the observance system by mistreatment the TCP/IP protocol through Wi-Fi. due to this method, aged and inveterately unwell patients will keep severally within their house and secure in the information that they're being monitored. the downside of this method is that it solely considers elder folks as there's a lot of probabilities of fulminant occurrence in them.

5. Title: Hospital Health Care Monitoring System using WirelessSensors Network

Author: Jagdish W. Bakal , S.S. Jondhale

Year: (2016)

There's continuous observation of the patient's physiological parameters like vital sign of patient likewise as vital sign. this technique is principally helpful for pregnant ladies to live the assorted parameters like vital sign, heart beat and fatal movement to manage the health issue. this technique must observation over one patient at a time and simply able to sense the vital sign (BP) and vital sign of the patients. during this system, there's a device node hooked up to body of patient to live signals from the wireless sensors and sends these signals to the information. this technique will find the abnormal conditions of the patient, raise associate degree alarm to the patient and sends a SMS/Email to the doctor for treatment. the most advantage of this technique is to extend the liberty for enhancing patient's quality of life. The demerit of this technique is that during this the patients ought to get admitted within the hospital for continuous observation of the patient physiological parameters. This WSN gets difficult if range of patient is admitted within the hospital on the far side the required limit.

6. Title: RFID Based Centralized Patient Monitoring System and Tracking (RPMST)

Author: Ramon idle, Jackson

Year: (2017)

Radio Frequency Identification (RFID) though finds its use in many applications, still has to be accepted in a clear view. Talking in the perspective of a patient the only thing that strikes first is that lots of treatments undergone, many health records to be maintained, and after which remembering the concerned

doctor's name and other details. This becomes impossible at a certain point of time when the details have to be conveyed. To make it easier this concept of maintaining a centralized information system and sharing has been proposed through the use of RFID technology. RFID is known for its unique ID number. Using this advantage, in the first section the monitoring of a patient is done whenever he arrives at the hospital. In case of any shifts from the hospital the information is still available. Every patient is provided with a unique RFID number and all the details regarding the patient and treatments are stored in a centralized database which is retrieved by the server. The second section is patient tracking where in case of any emergency in a closed environment the patient is provided with assistance in a short span of time. The patient is regularly monitored by the temperature and heartbeat sensor. The moment the value crosses the normal range a message is sent through GSM/GPS to the nearest hospital with its location and also to a relative. Assistance is provided accordingly to the patient

Chapter 3

SYSTEM DESIGN AND MODULE DISCRIPTION

EXISTING SYSTEM

- The health care sensors are playing an essential role in hospitals. The patient monitoring system is one of the major developments because of its innovative technology. An automatic wireless health monitoring system is used to measure patient's body temperature and heartbeat by using embedded technology.
- The proposed system uses both the sensors like heartbeat sensor and temperature sensor. These sensors mainly involve in monitoring the condition of the patient.
- But the sensor data cannot be upload to cloud
- So each and every time doctor will go there and check the parameter
- In some conditions doctor may busy means they never monitoring the person.

DRAWBACKS

- It is used to continuously monitoring of the patient is very difficult by the doctor.
- The sensor data cannot be upload to cloud.

PROPSOED SYSTEM

- IOT patient monitoring has 3 sensors. First one is a temperature sensor, second is Heartbeat sensor and the third one is humidity sensor. This project is very useful since the doctor can monitor patient health parameters just by visiting website or URL. And nowadays many IOT apps are also being developed. So now the doctor or family members can monitor or track the patient health through the Android apps.
- To operate IOT based health monitoring system project, you need a Wi-Fi connection. The microcontroller connects to the Wi-Fi network using a Wi-Fi module. This project will not work without a working Wi-Fi network. You can create a Wi-Fi zone using a Wi-Fi module or you can even create a Wi-Fi zone using Hotspot on your smartphone.
- The NodeMCU board continuously reads input from these 3 senses. Then it sends this data to the cloud by sending this data to a particular URL/IP address. Then this action of sending data to IP is repeated after a particular interval of time. For example

in this project, we have sent data after every 30 seconds.

ADVANTAGES OF PROPOSED SYSTEM

- By this proposed system we can monitor our health any time with out going to doctor for every time. So, it will save so much of money and we can monitor our health in our hands only.
- Very easy to get the details from the person we want to know.
- Home application can be done in this way to get out health check in home itself.

SYSTEM REQUIREMENTS

HARDWARE REQUIRMENTS

- Nodemcu
- Temperature sensor
- Heartbeat sensor
- Respiratory sensor
- Pressure sensor

SOFTWARE REQUIRMENTS

- Arduino ide
- Ubidots account.

SOFTWARE DESCRIPTION

Arduino ide

Arduino ide is a software in which we use it to write code for the IOT purpose. Which it is written in function of c and c++. In which we write a code in this ide and upload it to the Arduino compatible boards. With the rising quality of Arduino as a software system platform, different vendors began to implement custom open supply compilers and tools (cores) which will build and transfer sketches to different microcontrollers that aren't supported by Arduino's official line of microcontrollers. In October 2019 the Arduino organization began providing early access to a brand-new

Arduino professional IDE with debugging and different advanced options.

Why Arduino?

Tanks to its simple and accessible user experience, Arduino has been used in thousands of different projects and applications. The Arduino software is easy-to-use for beginners, yet flexible enough for advanced users. It runs on Mac, Windows, and Linux. Teachers and students use it to build low-cost scientific instruments, to prove chemistry and physics principles, or to get started with programming and robotics. Designers and architects build interactive prototypes, musicians and artists use it for installations and to experiment with new musical instruments. Makers, of course, use it to build many of the projects exhibited at the Maker Faire, for example. Arduino is a key tool to learn new things. Anyone - children, hobbyists, artists, programmers - can start tinkering just following the step-by-step instructions of a kit, or sharing ideas online with other members of the Arduino community.

Inexpensive - Arduino boards are relatively inexpensive compared to other microcontroller platforms. The least expensive version of the Arduino module can be assembled by hand, and even the pre-assembled Arduino modules cost less than \$50 Cross-platform - The Arduino Software (IDE) runs on Windows, Macintosh OSX, and Linux operating systems. Most microcontroller systems are limited to Windows.

Simple, clear programming environment - The Arduino Software (IDE) is easy-to-use for beginners, yet flexible enough for advanced users to take advantage of as well. For teachers, it's conveniently based on the Processing programming environment, so students learning to program in that environment will be familiar with how the Arduino IDE works.

Open source and extensible software - The Arduino software is published as open-source tools, available for extension by experienced programmers. The language can be expanded through C++ libraries, and people wanting to understand the technical details can make the leap from Arduino to the AVR C programming language on which it's based. Similarly, you can add AVR-C code directly into your Arduino programs if you want to.



Fig .3.1 Arduino ide

Ubidots

Ubidots is an IoT Platform empowering innovators and industries to prototype and scale IoT projects to production. Use the Ubidots platform to send data to the cloud from any Internet-enabled device. You can then configure actions and alerts based on your real-time data and unlock the value of your data through visual tools. Ubidots offers a REST API that allows you to read and write data to the resources available: data sources, variables, values, events and insights. The API supports both HTTP and HTTPS and an API Key is required.

Your data will be protected with two more replication, encrypted storage and optional TLS/SSL data support. You can also customize permission groups to each module of the platform, making sure the right information is shown to the right user



Fig.3.2 UBIDOTS

which allows System Integrators to customize their own IoT application, manage customers and users, and even code their own HTML/JS visualizations on top of it. It is like cloud in which the data will be used in future also.

NODEMCU

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

NodeMCU was created shortly after the ESP8266 came out. On December 30, 2013, Espressif Systems began production of the ESP8266. The ESP8266 is a Wi-Fi SoC integrated with a Tensilica Xtensa LX106 core, widely used in IoT applications (see related projects). NodeMCU started on 13 Oct 2014, when Hong committed the first file of

nodemcu-firmware to GitHub. Two months later, the project expanded to include an open-hardware platform when developer Huang R committed the gerber file of an ESP8266 board, named devkit v0.9. Later that month, Tuan PM ported MQTT client library from Contiki to the ESP8266 SoC platform, and committed to NodeMCU project, then NodeMCU was able to support the MQTT IoT protocol, using Lua to access the MQTT broker. Another important update was made on 30 Jan 2015, when Devsaurus ported the u8glib to NodeMCU project, enabling NodeMCU to easily drive LCD, Screen, OLED, even VGA displays.

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

- s 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](http://ArduinoIDE) 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". This has become a leading software development platform for the various ESP8266-based modules and development boards, including NodeMCUs.

- IoT platform node MCU is open source . Language used in it is lua scripting language. It is based on the eLua project, and built on the ESP8266 SDK 0.9.5. It uses many open source projects, such as lua-cjson, and spiffs. It includes firmware which runs on the ESP8266 Wi-Fi SoC, and hardware which is based on the ESP-12 module. NodeMCU was created shortly after the ESP8266 came out. In December 30, 2013, Espressif systems began production of the ESP8266. The ESP8266 is a Wi-Fi SoC integrated with a Tensilica Xtensa LX106 core, widely used in IoT applications. NodeMCU started in 13 Oct 2014, when Hong committed the first file of NodeMCU - firmware to GitHub. 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 1.0. Later that month, Tuan PM ported MQTT client library from Contiki to the ESP8266 SoC platform, and committed to Node MCU project, then Node MCU was able to support the MQTT IoT protocol, using Lua to access the MQTT IoT

protocol, using Lua to access the MQTT broker. Another important update was made on 30 Jan 2015, when Devsaurus ported the u8glib to NodeMCU project, enabling NodeMCU to easily drive LCD, Screen, OLED, even VGA displays.

- The main objective of this project is to develop a home automation system using an Node MCU board with Internet being remotely controlled by any Android OS smart phone. As technology is advancing so houses are also getting smarter. Modern houses are gradually shifting from conventional switches to centralized control system, involving remote controlled switches. Presently, conventional wall switches located in different parts of the house makes it difficult for the user to go near them to operate. Even more it becomes more difficult for the elderly or physically handicapped people to do so. Remote controlled home automation system provides a most modern solution with smart phones.
- In order to achieve this, a relay module is interfaced to the Node MCU board at the receiver end while on the transmitter end, a GUI application on the cell phone sends ON/OFF commands to the receiver where loads are connected. By touching the specified location on the GUI, the loads can be turned ON/OFF remotely through this technology. The loads are operated by IOT board through Relay Module.
- NODEMCU (esp8266) has been selected as the controller for this system due to its compact size, compatibility, easy interfacing over several other type of controller including Programmable Integrated Circuit (PIC), Programmable Logic Controller (PLC) and others. ESP8266 is an open source firmware that is built on top of the chip manufacturer's proprietary SDK. The firmware provides a simple programming environment, which is a very simple and fast scripting language The ESP8266 chip incorporates on a standard circuit board. The board has a built-in USB port that is already wired up with the chip, a hardware reset button, Wi-Fi antenna, LED lights, and standard-sized GPIO (General Purpose Input Output) pins that can plug into a bread board..Figure-3 shows the diagram of NODEMCU (ESP8266).It has Processor called L106 32bit RISC microprocessor core based on the Tensilica Xtensa Diamond Standard 106Micro running at 80 MHz and has a memory of 32 Kbit instruction RAM ,32 Kbit instruction cache RAM, 80 Kbit user data RAM&16 KbitETS system data RAM. It has inbuilt Wi-Fi module of IEEE 802.11 b/g/n Wi-Fi.

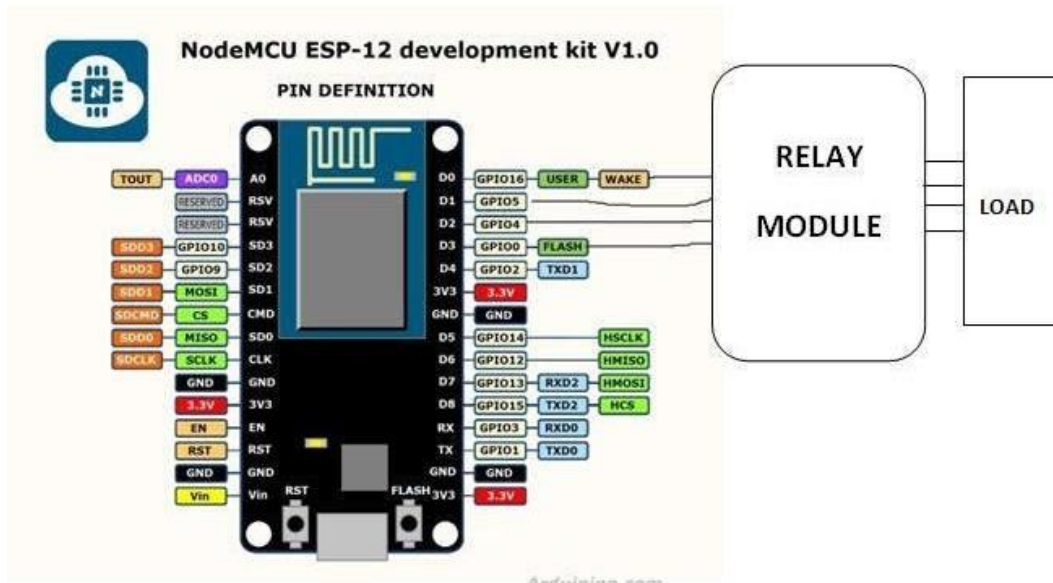


Fig no – 3.3 – Nodemcu

CIRCUIT DIAGRAM DESCRIPTION

- The operation of power supply circuits built using filters, rectifiers, and then voltage regulators. Starting with an AC voltage, a steady DC voltage is obtained by rectifying the AC voltage, Then filtering to a DC level, and finally, regulating to obtain a desired fixed DC voltage. The regulation is usually obtained from an IC voltage regulator Unit, which takes a DC voltage and provides a somewhat lower DC voltage, Which remains the same even if the input DC voltage varies, or the output Load connected to the DC voltage changes.

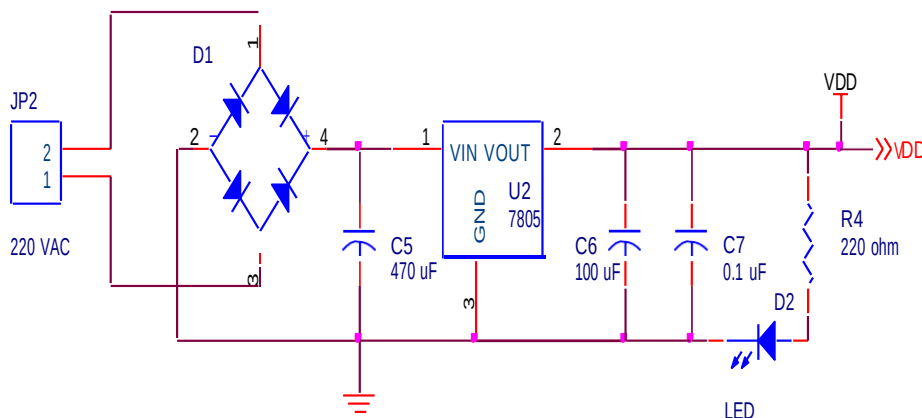


Fig 3.4 Power supply of Nodemcu

CIRCUIT DIAGRAM OF POWER SUPPLY

WORKING PRINCIPLE

The AC voltage, typically 220V rms, is connected to a transformer, which steps that ac voltage down to the level of the desired DC output. A diode rectifier then provides a full-wave rectified voltage that is initially filtered by a simple capacitor filter to produce a dc voltage. This resulting dc voltage usually has some ripple or ac voltage variation. A regulator circuit removes the ripples and also remains the same dc value even if the input dc voltage varies, or the load connected to the output dc voltage changes.

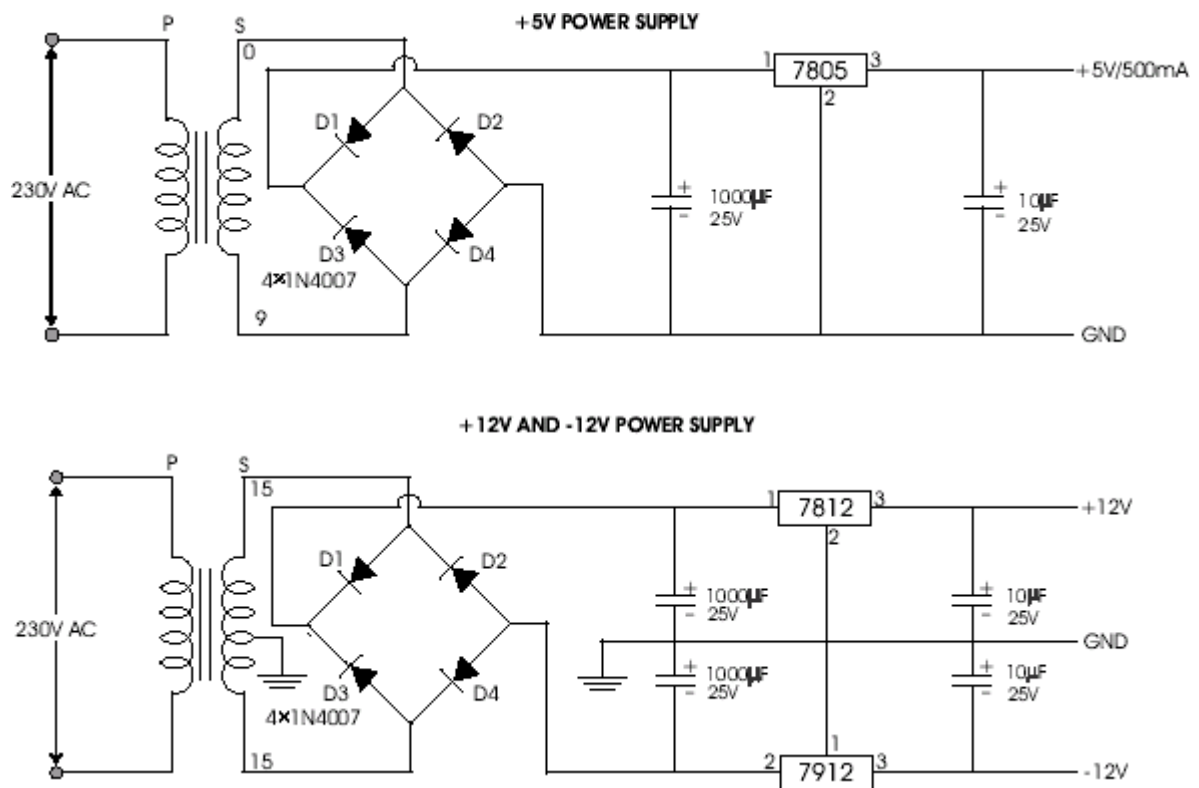


Fig.3.5 Circuit diagram

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 phlethysmography. 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 photophlethysmography:

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

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

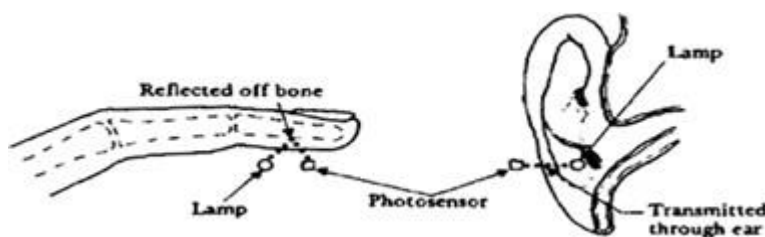


Fig .3.6 Heart rate sensor principle

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.

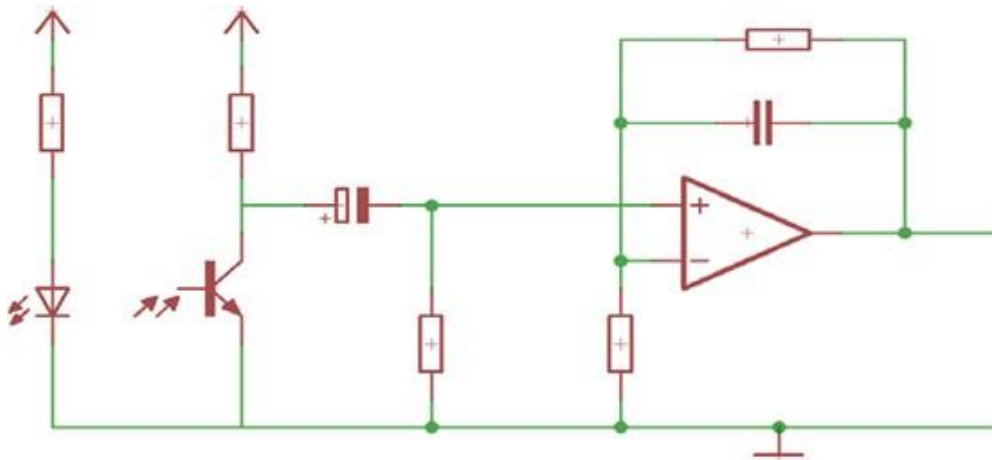


Fig3.7 Circuit diagram of sensor

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,

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

Where f is the pulse frequency

Practical Heartbeat Sensor

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



Fig.3.8 Pule Sensor

Another example is TCRT1000, having 4 pins-

Pin1: To give supply voltage to the LED

Pin2 and 3 are grounded. Pin 4 is the output. Pin 1 is also the enable pin and pulling it high turns the LED on and the sensor starts working. It is embedded on a wearable device which can be worn on the wrist and the output can be sent wirelessly (through Bluetooth) to the computer for processing.

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 Dependant 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.

TEMPERATURE SENSOR (LM35)

FEATURES DESCRIPTION

- Calibrated Directly in ° Celsius (Centigrade)
- Linear + 10 mV/°C Scale Factor
- 0.5°C Ensured Accuracy (at +25°C) LM35 has an advantage over linear temperature
- Rated for Full –55°C to +150°C Range
- Suitable for Remote Applications
- Low Cost Due to Wafer-Level Trimming
- Less than 60-μA Current Drain
- Low Self-Heating, 0.08°C in Still Air
- Nonlinearity Only $\pm 1/4^\circ\text{C}$ Typical
- Low Impedance Output, 0.1 Ω for 1 mA Load

DESCRIPTION

The LM35 series are precision integrated-circuit temperature sensors, with an output voltage linearly proportional to the Centigrade temperature. Thus the LM35 has an advantage over linear temperature sensors calibrated in ° Kelvin, as the user is not required to subtract a large constant voltage from the output to obtain convenient Centigrade scaling. The LM35 does not require any external calibration or trimming to provide typical accuracies of $\pm\frac{1}{4}^{\circ}\text{C}$ at room temperature and $\pm\frac{3}{4}^{\circ}\text{C}$ over a full -55°C to $+150^{\circ}\text{C}$ temperature range. Low cost is assured by trimming and calibration at the wafer level. The low output impedance, linear output, and precise inherent calibration of the LM35 make interfacing to readout or control circuitry especially easy. The device is used with single power supplies, or with plus and minus supplies. As the LM35 draws only $60\text{ }\mu\text{A}$ from the supply, it has very low self-heating of less than 0.1°C in still air. The LM35 is rated to operate over a -55°C to $+150^{\circ}\text{C}$ temperature range, while the LM35C is rated for a -40°C to $+110^{\circ}\text{C}$ range (-10° with improved accuracy). The LM35 series is available packaged in hermetic TO transistor packages, while the LM35C, LM35CA, and LM35D are also available in the plastic TO-92 transistor package. The LM35Dis also available in an 8-lead surface-mount small outline package and a plastic TO-220 package.

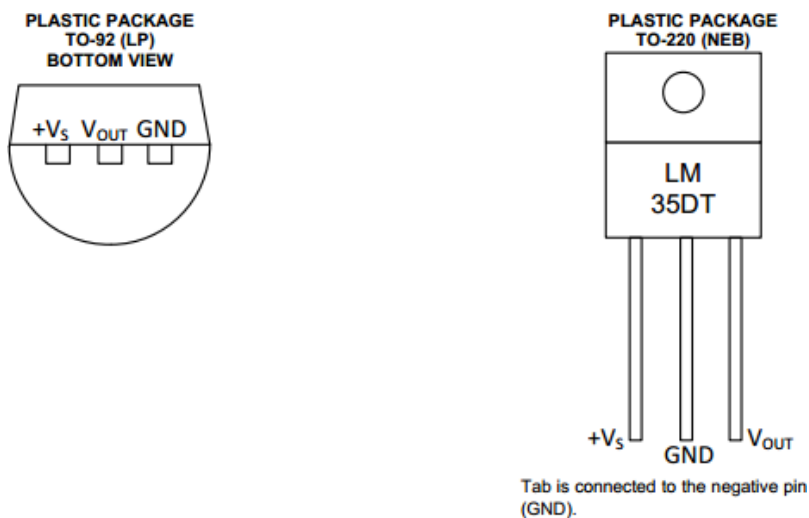


Fig.3.9 Temperature sesor

Pressure sensor

A pressure sensor measures pressure, typically of gases or liquids. Pressure is an expression of the force required to stop a fluid from expanding, and is usually stated in terms of force per unit area. A pressure sensor usually acts as a transducer; it generates a signal as a function of the pressure imposed. For the purposes of this article, such a signal is electrical.

Pressure sensors are used for control and monitoring in thousands of everyday applications.

Pressure sensors can also be used to indirectly measure other variables such as fluid/gas flow, speed, water level, and altitude. Pressure sensors can alternatively be called pressure

transducers, pressure transmitters, pressure senders, pressure

indicators, piezometers and manometers, among other names.

Pressure sensors can vary drastically in technology, design, performance, application suitability and cost. A conservative estimate would be that there may be over 50 technologies and at least 300 companies making pressure sensors worldwide.

There is also a category of pressure sensors that are designed to measure in a dynamic mode for capturing very high speed changes in pressure. Example applications for this type of sensor would be in the measuring of combustion pressure in an engine cylinder or in a gas turbine. These sensors are commonly manufactured out of piezoelectric materials such as quartz.

Some pressure sensors, such as those found in some traffic enforcement cameras, function in a binary (off/on) manner, i.e., when pressure is applied to a pressure sensor, the sensor acts to complete or break an electrical circuit. These types of sensors are also known as a pressure switch.

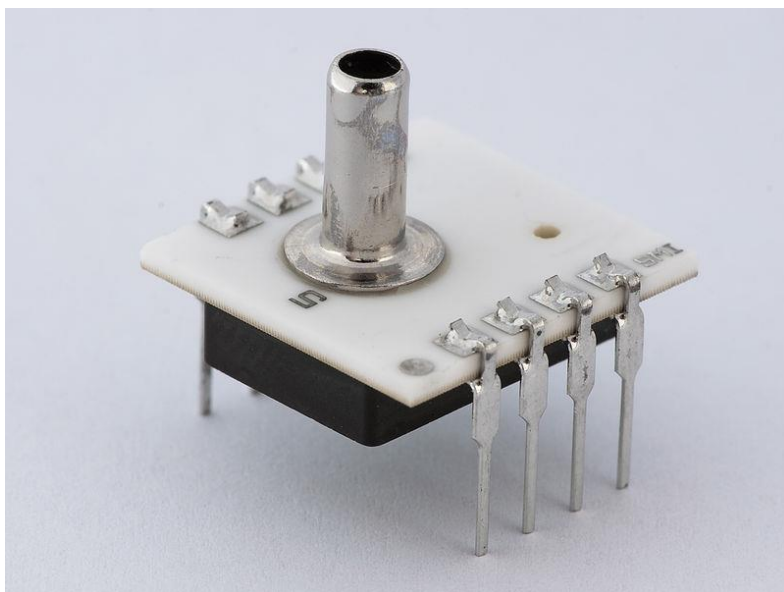


Fig.3.10 Pressure Sensor

Humidity sensor

Measurement and control of temperature and relative humidity finds applications in numerous areas. These days devices are available which have both temperature and humidity sensors with signal conditioning, ADC, calibration and communication interface all built inside them. The use of such smart sensors greatly simplify the design and reduces the overall cost. We discussed in past about [Humidity and temperature measurements with Sensirion's SHT1x/SHT7x sensors](#). These sensors are capable of measuring both temperature and relative humidity and provide fully calibrated digital outputs. While SHT1x/SHT7x are very accurate sensors, they are still expensive for hobbyists use. This articles discusses the DHT11 sensor which also provides calibrated digital outputs for temperature and humidity but is relatively lot cheaper than the Sensirion sensors. The DHT11 sensor uses a proprietary 1-wire protocol which we will be exploring here and implementing with the PIC16F628A microcontroller that will receive the temperature and humidity values from the sensor and display them on a 16×2 character LCD.

DHT11 sensor

The DHT11 sensor comes in a single row 4-pin package and operates from 3.5 to 5.5V power supply. It can measure temperature from 0-50 °C with an accuracy of $\pm 2^{\circ}\text{C}$ and relative humidity ranging from 20-95% with an accuracy of $\pm 5\%$. The sensor provides fully calibrated digital outputs for the two measurements. It has got its own proprietary 1-wire protocol, and therefore, the communication between the sensor and a microcontroller is not possible through a direct interface with any of its peripherals. The protocol must be

implemented in the firmware of the MCU with precise timing required by the sensor. The following timing diagrams describe the data transfer protocol between a MCU and the DHT11 sensor. The MCU initiates data transmission by issuing a “Start” signal. The MCU pin must be configured as output for this purpose. The MCU first pulls the data line low for at least 18 ms and then pulls it high for next 20-40 μ s before it releases it.

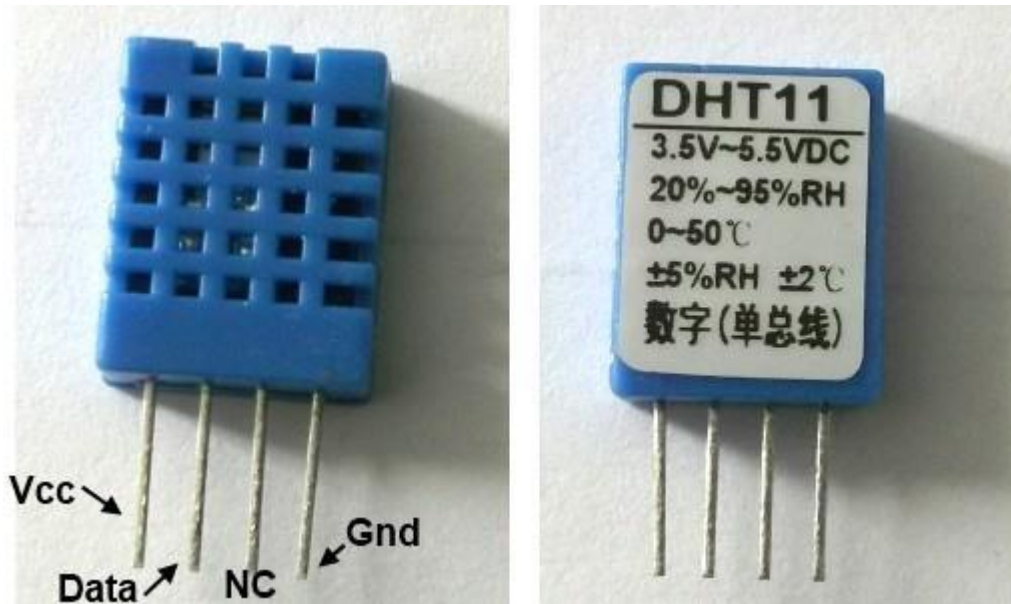
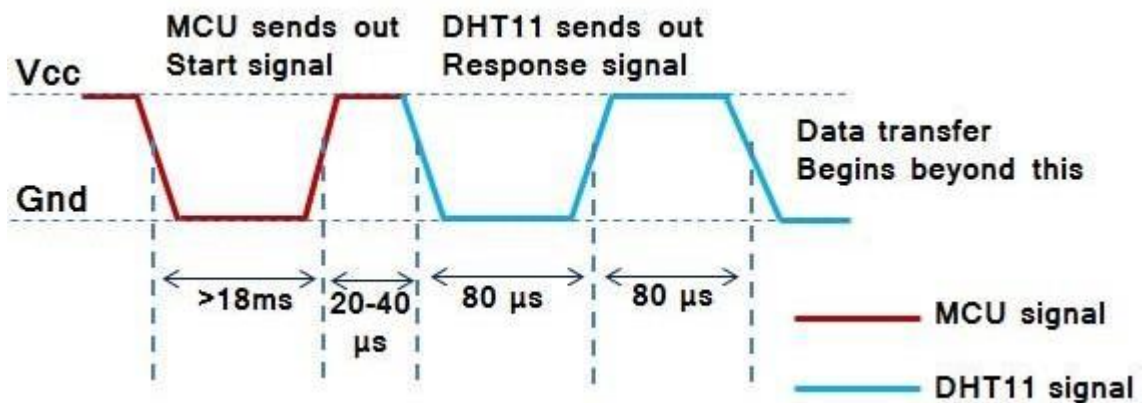


Fig. 3.11 DHT11 sensor comes in a single row 4-pin package

Next, the sensor responds to the MCU “Start” signal by pulling the line low for 80 μ s followed by a logic high signal that also lasts for 80 μ s. Remember that the MCU pin must be configured to input after finishing the “Start” signal. Once detecting the response signal from the sensor, the MCU should be ready to receive data from the sensor. The sensor then sends 40 bits (5 bytes) of data continuously in the data line. Note that while transmitting bytes, the sensor sends the most significant bit first.



"Start" and "Response" signals

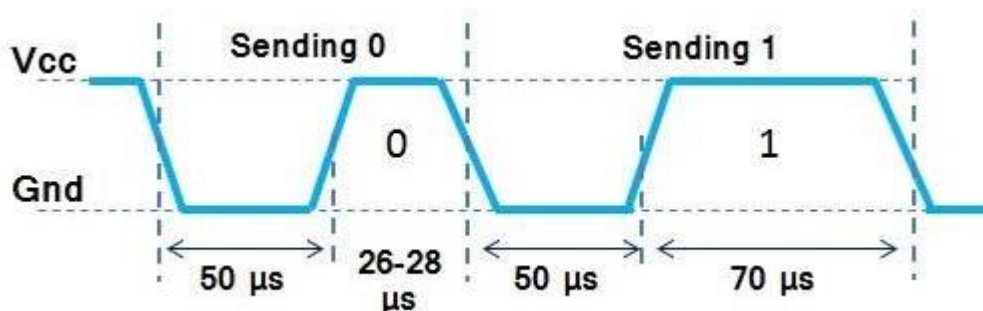
The 40-bit data from the sensor has the following structure.

Data (40-bit) = Integer Byte of RH + Decimal Byte of RH + Integer Byte of Temp. +
Decimal Byte of Temp. + Checksum Byte

For DHT11 sensor, the decimal bytes of temperature and humidity measurements are always zero. Therefore, the first and third bytes of received data actually give the numeric values of the measured relative humidity (%) and temperature (°C). The last byte is the checksum byte which is used to make sure that the data transfer has happened without any error. If all the five bytes are transferred successfully then the checksum byte must be equal to the last 8 bits of the sum of the first four bytes, i.e.,

Checksum = Last 8 bits of (Integer Byte of RH + Decimal Byte of RH + Integer Byte of Temp. + Decimal Byte of Temp.)

Now let's talk about the most important thing, which is signalling for transmitting "0" and "1". In order to send a bit of data, the sensor first pulls the line low for 50 µs. Then it raises the line to high for 26-28 µs if it has to send "0", or for 70 µs if the bit to be transmitted is "1". So it is the width of the positive pulse that carries information about 1 and 0.



Timing difference for transmitting "1s" and "0s"

At the end of the last transmitted bit, the sensor pulls the data line low for 50 µs and then releases it. The DHT11 sensor requires an external pull-up resistor to be connected between its Vcc and the data line so that under idle condition, the data line is always pulled high. After finishing the data transmission and releasing the data line, the DHT11 sensor goes to the low-power consumption mode until a new "Start" signal arrives from the MCU.

Chapter 4

IMPLEMENTATION

4. EMBEDDED SYSTEMS

Overview of embedded systems

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

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

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

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

Embedded systems provide several functions

- Monitor the environment; embedded systems read data from input sensors. This data is then

processed and the results displayed in some format to a user or users

- Control the environment; embedded systems generate and transmit commands for actuators.
- Transform the information; embedded systems transform the data collected in some meaningful way, such as data compression/decompression

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

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

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

Block diagram of an embedded system:

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

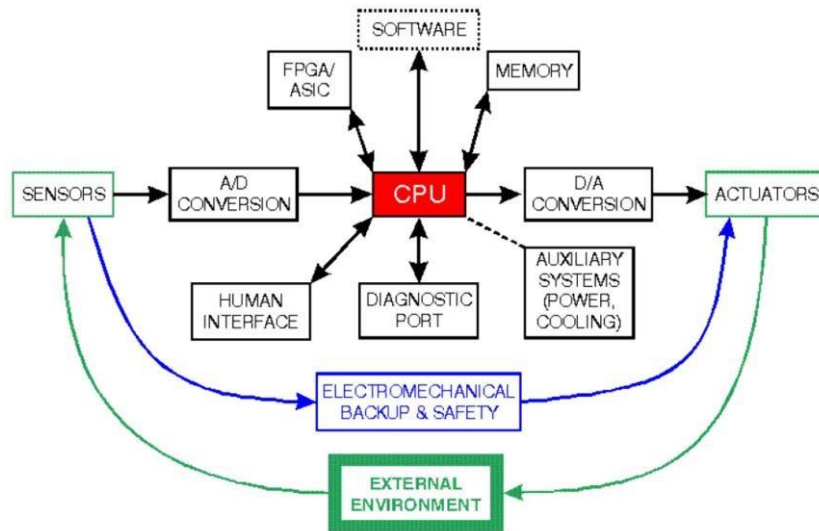


Fig.4.1 - Block diagram of a typical embedded system

Characteristics of embedded systems

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

Application Specific Systems:

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

Reactive Systems

As mentioned earlier, a typical embedded systems model responds to the environment via sensors and control the environment using actuators. This requires embedded systems to run at the speed of the environment. This characteristic of embedded system is called “reactive”. Reactive computation means that the system (primarily the software component) executes in

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

Distributed Systems

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

Heterogeneous Architectures

Embedded systems often are composed of heterogeneous architectures (Fig 1.2). They may contain different processors in the same system solution. They may also be mixed signal systems. The combination of I/O interfaces, local and remote memories, and sensors and actuators makes embedded system design truly unique. Embedded systems also have tight design constraints, and heterogeneity provides better design flexibility.

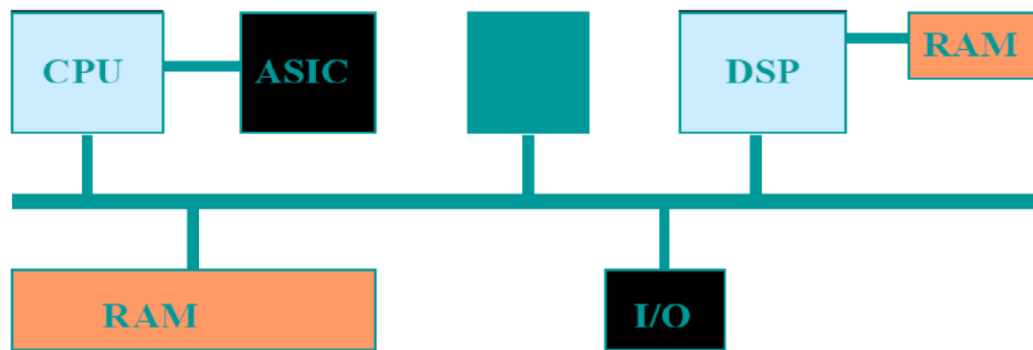


Fig .4.2 Embedded Systems having Heterogeneous Architectures

Harsh environment

Many embedded systems do not operate in a controlled environment. Excessive heat is often a problem, especially in applications involving combustion (e.g., many transportation applications). Additional problems can be caused for embedded computing by a need for protection from vibration, shock, lightning, power supply fluctuations, water, corrosion, fire, and general physical abuse.

System safety and reliability

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

Control of physical systems

One of the main reasons for embedding a computer is to interact with the environment. This is often done by monitoring and controlling external machinery. Embedded computers transform the analog signals from sensors into digital form for processing. Outputs must be transformed

back to analog signal levels. When controlling physical equipment, large current loads may need to be switched in order to operate motors and other actuators. To meet these needs, embedded systems may need large computer circuit boards with many non-digital components. Embedded system designers must carefully balance system tradeoffs among analog components, power, mechanical, network, and digital hardware with corresponding software.

Small and low weight

Many embedded computers are physically located within some larger system. The form factor for the embedded system may be dictated by aesthetics. For example, the form factor for a missile may have to fit inside the nose of the missile. One of the challenges for embedded systems designers is to develop non-rectangular geometries for certain solutions. Weight can also be a critical constraint. Embedded automobile control systems, for example, must be light weight for fuel economy. Portable CD players must be light weight for portability purposes.

Cost sensitivity

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

Power management

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

POWER SUPPLY

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

some what, or the load connected to the output dc voltages changes.

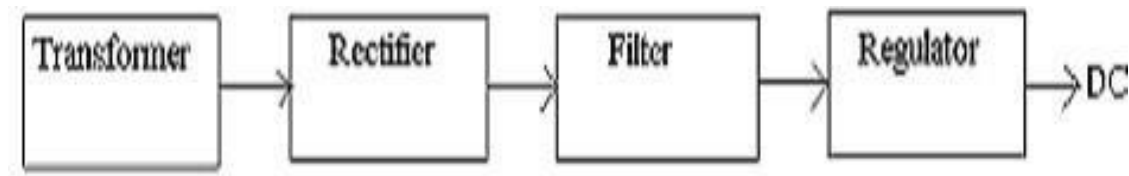


Fig.4.3 General Block Of Power Supply Unit

TRANSFORMER

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

RECTIFIER

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

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

FILTERS

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

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.

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

REGULATORS

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

DATA COLLECTION MODULE

In this module, we implement the collection of data from the sensors and get them connected to the nodemcu that there will be transfer of data from nodemcu to laptop. We will send a mail to a particular person whenever the parameter is greater than the expected one and will send location to your phone using ubidots application so it will help so many people in this way

Chapter 5

RESULTS AND SCREENSHOTS

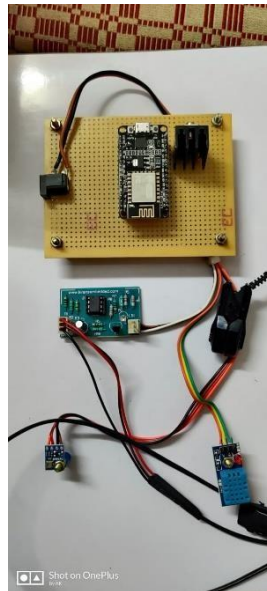


Fig.5.1 It is the hardware part of the project and it is the main.

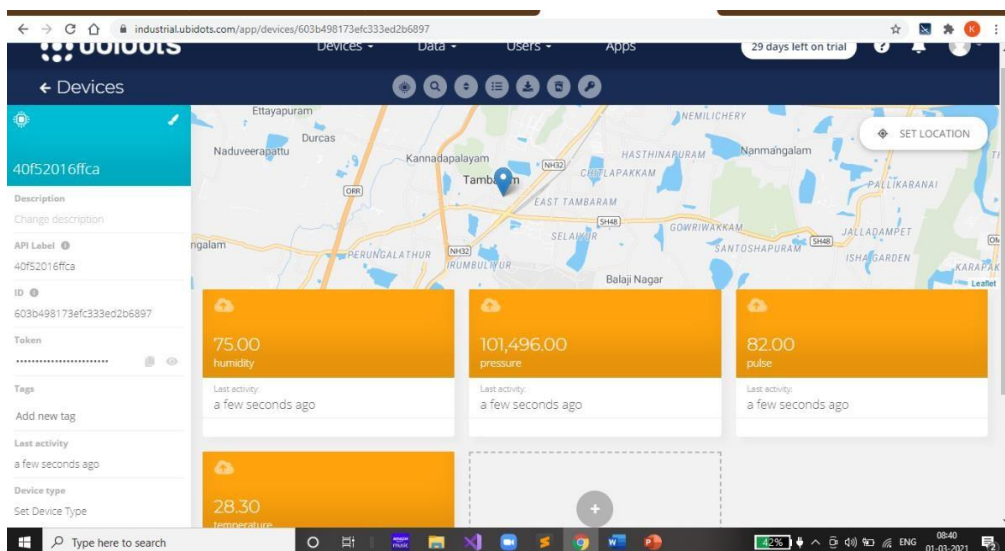


Fig.5.2 Website image

when we keep our finger on the pulse sensor everything will be calculated in

seconds and we can see everything in the page we have shown you in the screen shot.



Fig.5.3 Phone tracking image

in this screen shot we can know the location of the person in phone so that we can track the person easily when he is in danger.

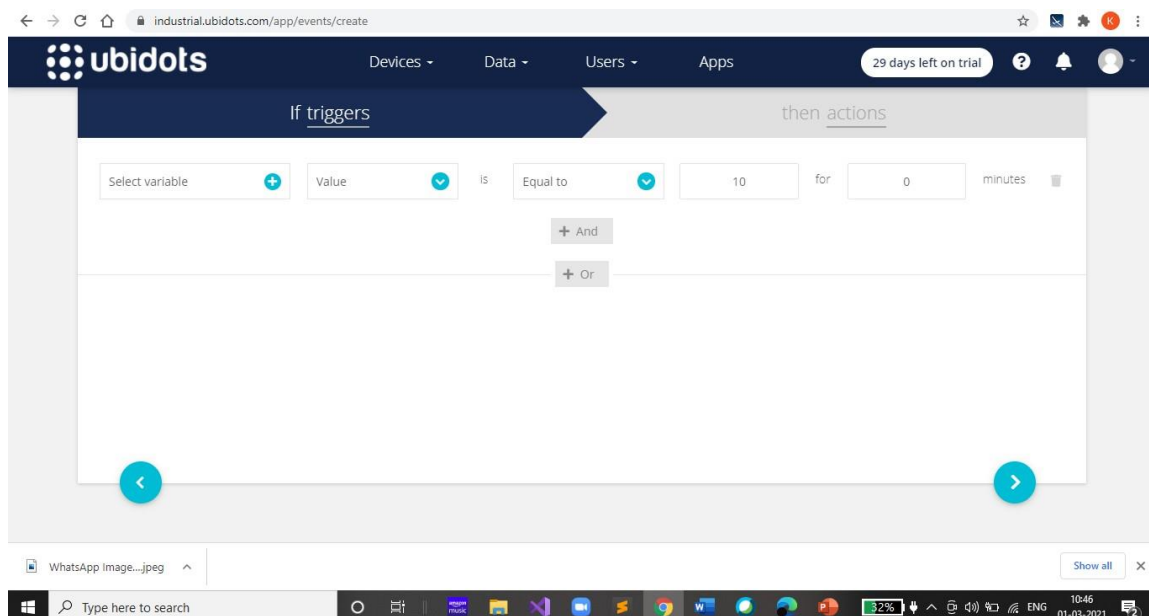


Fig.5.4 Notification sending part

In this part we can get the notification from this website when ever the person temperature or heart rate or any of the parameter is increasing rapidly we can

easily get to know the person is in danger.

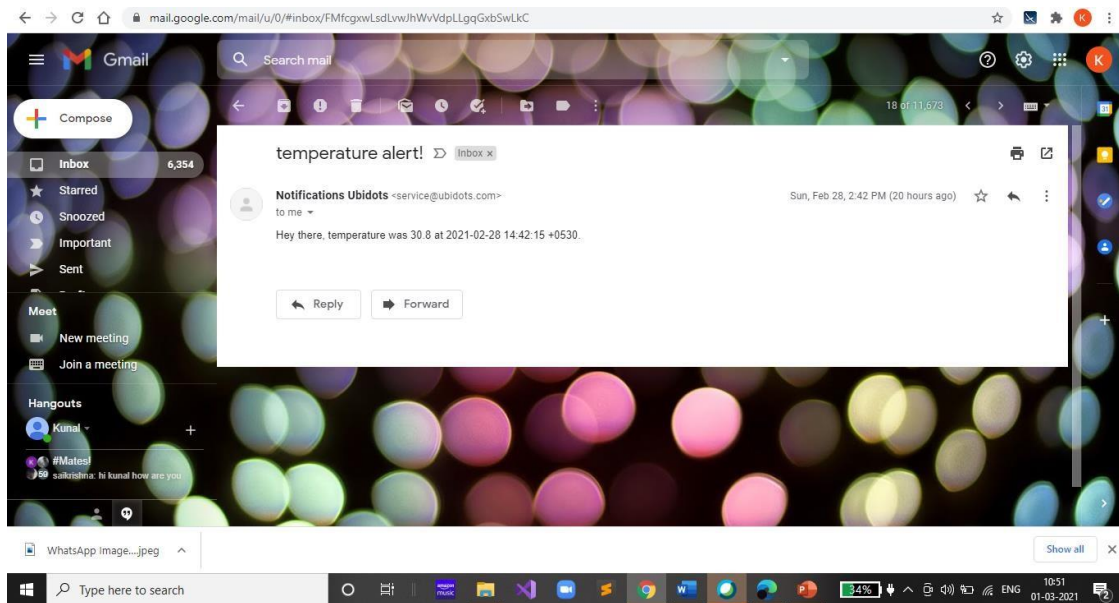


Fig.5.5 Gmail notification message

In this screen shot we have got the notification when the temperature is increasing rapidly the notification is sent to the mail itself.

Chapter 6

CONCLUSION AND FUTURE ENHANCEMENTS

CONCLUSION

- By the on top of matter we've got have to be compelled to apprehend that health is wealth. so as to take care of our body during a correct means we have a tendency to use some sensors to spot during this project.
- Some sensors like pulse, temperature GPS pursuit. once connecting every part during a correct manner we have a tendency to write code for the elements to attach one another by writing the code in Arduino ide once writing the code confirm there are not any errors within the code so we will connect the elements to laptop computer.
- once connecting the laptop computer we have a tendency to connect the nodemcu with same Wi-Fi that's connected within the laptop computer. afterward take a device and tag that adult male to your forefinger and you'll be able to see the output within the laptop computer and placement of the

FUTURE ENHANCEMENT

In future we can make some more interesting project like direct inking to the hospital or doctor phone so that sudden reaction can be taken so that we save a human life in further ways. Not only this we can also add some more informative sensor for extra health precision.

REFERENCES

1. D. Miorandi, S. Sicarib, F. De Pellegrinia and I. Chlamtac, “Internet of Things: Vision, applications and research challenges,” *Ad Hoc Networks* 10 (2012) 14971516.
2. Internet of Things, “European Research Cluster on the Internet of Things,” [Online]:<http://www.internet-of-things-research.eu/aboutiot.htm>
3. Mohammad S. Jassas, Abdullah A. Qasem, Qusay H. Mahmoud, “A Smart System Connecting e-Health Sensors and the Cloud,” *Proceeding of the IEEE 28th Canadian Conference on Electrical and Computer Engineering* Halifax, Canada, May 3-6, 2015.
4. Punit Gupta, Deepika Agrawal, Jasmeet Chhabra, Pulkit Kumar Dhir, “IoT based Smart HealthCare Kit,” *2016 International Conference on Computational Techniques in Information and Communication Technologies (ICCTICT)*.
5. Ravi Kishore Kodali, Govinda Swamy and Boppana Lakshmi, “An Implementation of IoT for Healthcare,” *2015 IEEE Recent Advances in Intelligent Computational Systems (RAICS)* — 10-12 December 2015 Trivandrum.
6. S. C. Mukhopadhyay, N. K. Suryadevara, “Internet of Things:

APPENDIX

SOURCE CODE

```
#include <SoftwareSerial.h>
SoftwareSerial mySerial(14, 12);

#include <Adafruit_Sensor.h>
#include <DHT.h>
#include <DHT_U.h>

#define DHTPIN D4

#define DHTTYPE  DHT11

DHT_Unified dht(DHTPIN, DHTTYPE);
uint32_t delayMS;

#include <Arduino.h>
#include <Wire.h>

#include <BMP180I2C.h>

#define I2C_ADDRESS 0x77

BMP180I2C bmp180(I2C_ADDRESS);

int j=0;
int b=0;

#include "Ubidots.h"

const char* UBIDOTS_TOKEN = "BBFF-
JG2tM3eaBEGoCoMG6K17lXtvfEoSv"; // Put here your Ubidots
TOKEN
const char* WIFI_SSID = "health_2021";    // Put here your Wi-Fi SSID
```

```

const char* WIFI_PASS = "12345678";    // Put here your Wi-Fi
password
Ubidots ubidots(UBIDOTS_TOKEN, UBI_HTTP);

void setup() {
  Serial.begin(9600);
  ubidots.wifiConnect(WIFI_SSID, WIFI_PASS);
  dht.begin();
  Serial.println(F("DHTxx Unified Sensor Example"));
  sensor_t sensor;
  dht.temperature().getSensor(&sensor);
  dht.humidity().getSensor(&sensor);
  delayMS = sensor.min_delay / 1000;
  while (!Serial);
  pinMode(D0, INPUT);

  Wire.begin();

  if (!bmp180.begin())
  {
    Serial.println("begin() failed. check your BMP180 Interface and I2C
Address.");
    while (1);
  }

  bmp180.resetToDefaults();

  bmp180.setSamplingMode(BMP180MI::MODE_UHR);

}

void loop() {
  int pulse = digitalRead(D0);
  if (pulse == HIGH) ubidots.add("Pulse", random(80,86));
  sensors_event_t event;
  dht.temperature().getEvent(&event);
  if (isnan(event.temperature)) {
    Serial.println(F("Error reading temperature!"));
  }
  else {
    Serial.print(F("Temperature: "));
    Serial.print(event.temperature);

```

```

    Serial.println(F("°C"));
}
// Get humidity event and print its value.
dht.humidity().getEvent(&event);
if (isnan(event.relative_humidity)) {
    Serial.println(F("Error reading humidity!"));
}
else {
    Serial.print(F("Humidity: "));
    Serial.print(event.relative_humidity);
    ubidots.add("Humidity", event.relative_humidity);
    Serial.println(F("%"));
}

if (!bmp180.measureTemperature())
{
    Serial.println("could not start temperature measurement, is a
measurement already running?");
    return;
}
do
{
    delay(100);
} while (!bmp180.hasValue());

Serial.print("Temperature: ");
Serial.print(bmp180.getTemperature());
ubidots.add("Temperature", bmp180.getTemperature());
Serial.println(" degC");

if (!bmp180.measurePressure())
{
    Serial.println("could not start perssure measurement, is a
measurement already running?");
    return;
}
do
{
    delay(100);
} while (!bmp180.hasValue());

Serial.print("Pressure: ");
Serial.print(bmp180.getPressure());

```

```

ubidots.add("Pressure", bmp180.getPressure());
Serial.println(" Pa");

    bool bufferSent = false;
    bufferSent = ubidots.send(); // Will send data to a device label that
    matches the device Id

    if (bufferSent) {
        Serial.println("Values sent by the device");
    }
    delay(2000);
// }
}
int sensorPin = A0;                // A0 is the input pin for the heart
rate sensor

float sensorValue = 0;             // Variable to store the value
coming from the sensor

int count = 9;

unsigned long starttime = 0;

int heartrate = 0;

boolean counted = false;


void setup (void) {

    pinMode (LED_BUILTIN, OUTPUT);    // D13 LED as
    Status Indicator

    Serial.begin (115200);            // Start Serial Communication
    @ 9600

}

```

```

void loop (){

    starttime = millis();

    while (millis()<starttime+10000)           // Reading pulse sensor for
    10 seconds

    {

        sensorValue = analogRead(sensorPin);

        if (sensorValue > 550 && counted == false) // Threshold value is 550 (~
        2.7V)

        {

            count++;

            Serial.print ("count = ");

            Serial.println (count);

            digitalWrite (LED_BUILTIN,HIGH);

            delay (50);

            digitalWrite (LED_BUILTIN, LOW);

            counted = true;

        }

        else if (sensorValue < 550)

        {

```

```

counted = false;

digitalWrite (LED_BUILTIN, LOW);

}

}

heartrate = count*6;           // Multiply the count by 6 to get
beats per minute

Serial.println ();

Serial.print ("BPM = ");

Serial.println (heartrate);    // Display BPM in the Serial
Monitor

Serial.println ();

count = 0;
}
#include <ESP8266WiFi.h>

const char *ssid = "kunal";    // replace with your wifi ssid and wpa2 key
const char *pass = "kunal1234";
int sensorPin = A0;           // A0 is the input pin for the heart
rate sensor

float sensorValue = 0;        // Variable to store the value
coming from the sensor

int count = 9;

unsigned long starttime = 0;

int heartrate = 0;

boolean counted = false;

```

```

WiFiClient client;

void setup()
{
  pinMode (LED_BUILTIN, OUTPUT);
  Serial.begin(115200);
  delay(10);
  Serial.println("hi kunal");

  Serial.println("Connecting to ");
  Serial.println(ssid);

  WiFi.begin(ssid, pass);
  while (WiFi.status() != WL_CONNECTED)
  {
    delay(500);
    Serial.print(".");
  }
  Serial.println("");
  Serial.println("WiFi connected");
}

void loop()
{
  starttime = millis();

  while (millis()<starttime+10000)           // Reading pulse sensor for
  10 seconds

  {

  sensorValue = analogRead(sensorPin);

  if (sensorValue > 550 && counted == false) // Threshold value is 550 (~
  2.7V)

  {

  count++;

```

```

Serial.print ("count = ");

Serial.println (count);

digitalWrite (LED_BUILTIN,HIGH);

delay (50);

digitalWrite (LED_BUILTIN, LOW);

counted = true;
}

else if (sensorValue < 550)
{
counted = false;

digitalWrite (LED_BUILTIN, LOW);
}

}
heartrate = count*6;           // Multiply the count by 6 to get
beats per minute

Serial.println ();

Serial.print ("BPM = ");

Serial.println (heartrate);    // Display BPM in the Serial
Monitor
Serial.println ();
count = 0;
}

```