A project Report

"Patient Health Monitoring System Using Cloud Based Bylnk Technology"

Submitted in partial fulfilment of the requirements for the award of the degree of

BACHELOR OF TECHNOLOGY

IN

ELECTRONICS & COMMUNICATION ENGINEERING

By

P. SAI KRISHNA	**20MH1A04B5**
K. DIVYA	**20MH1A04A1**
K. MURALI KARTHIK	**21MH5A0436**
L. JAI SAI SANTOSH	**20MH1A04A0**

Under the Esteemed Guidance of

Dr. R. Raman, M.Tech, Ph.D

Professor

Department of Electronics & Communication Engineering

ADITYA COLLEGE OF ENGINEERING (A)

(*Approved by AICTE, New Delhi & Permanently Affiliated to JNTUK, Kakinada, accredited by NBA & NAAC A++)*

Recognized by UGC under sections 2(f) & 12(b) of UGC Act, 1956

Aditya Nagar, ADB road, Surampalem, E.G.Dt, A.P-533437

VISION & MISSION OF THE INSTITUTE

VISION

To induce higher planes of learning by imparting technical education with

- International standards
- Applied research
- Creative Ability
- Value based instruction and to emerge as a premiere institute.

MISSION

Achieving academic excellence by providing globally acceptable technical education by forecasting technology through

- Innovative Research And development
- Industry Institute Interaction
- Empowered Manpower

VISION & MISSION OF THE DEPARTMENT

VISION

"To be a center of excellence and renowned for Electronics &Communication Engineering education and research"

MISSION

M1:Enlighten the graduates in the basic concepts underlying the principles of analog and digital electronics, communication systems and advanced technologies.

M2:Provide state of the art infrastructure and research facilities

M3:Organizing industrial programs and social activities in collaboration with industries, NSS to disseminate knowledge

ADITYA COLLEGE OF ENGINEERING

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

BONAFIDE CERTIFICATE

This is to certify that the Project Report entitled

"Patient Health Monitoring System Using Cloud Based Bylnk Technology"

being submitted by

P. SAI KRISHNA	**20MH1A04B5**
K. DIVYA	**20MH1A04A1**
K. MURALI KARTHIK	**21MH5A0436**
L. JAI SAI SANTOSH	**20MH1A04A0**

In partial fulfilment of the requirements for the award of the Degree of Bachelor of Technology in Electronics and Communication Engineering, at Aditya College of Engineering is a bonafide work carried out by them under my guidance and supervision of Dr. R Raman M. Tech, Ph. D

Project Guide
Dr. R Raman M.Tech, Ph. D
Professor
Department of ECE
Aditya College of Engineering

Head of the Department
Ch. Janaki Devi M. Tech, (Ph.D)
Associate Professor & HOD
Department of ECE
Aditya College of Engineering

2020-2024

DECLARATION

I hereby declare that the entire project work embodied in this dissertation entitled “Patient Health Monitoring System Using Cloud Based Bylnk Technology” has been independently carried out by us. As per my knowledge, no part of this work has submitted for any degree in any institution, university and organization previously. I hereby boldly state that to the best of my knowledge my work is free from plagiarism.

Yours sincerely,

P. SAI KRISHNA	20MH1A04B5
K. DIVYA	20MH1A04A1
K. MURALI KARTHIK	21MH5A0436
L. JAI SAI SANTOSH	20MH1A04A0

ACKNOWLEDGEMENT

We express our sincere gratitude and heartful thanks to the under stated person for the successful completion of our final project on "**Patient Health Monitoring System Using Cloud Based Bylnk Technology**".

First and foremost, we wish to thank our beloved guide **Dr. R RAMAN,** Professor for his kind guidance, valuable advices and utmost care at every stage of our final project.

We would like to thank **Ch Janaki Devi,** Associate Professor Head of the Department of E.C.E who have provided vital information, which was necessary for success of the project.

We also wish to convey our sincere thanks to **Dr. A. RAMESH** sir, principal of **ADITYA COLLEGE OF ENGINEERING** for providing appropriate environment required for our project.

We own our sincere gratitude to all other faculty members of **ADITYA COLLEGE OF ENGINEERING** for their help directly & indirectly on completion of this project.

Yours sincerely,

P. SAI KRISHNA	20MH1A04B5
K. DIVYA	20MH1A04A1
K. MURALI KARTHIK	21MH5A0436
L. JAI SAI SANTOSH	20MH1A04A0

ABSTRACT

This project explores the development of an "**Patient Health Monitoring System Using Cloud Based Bylnk Technology**". The primary objective is COVID-19 pandemic has highlighted the importance of self-isolation and physical distancing in healthcare. The Internet of Things (IoT) has been proposed as a potential application for monitoring health parameters, such as body temperature, pulse rate, body movement, and blood oxygen saturation, to provide real-time health systems. This project uses an IoT-based wireless communication system to monitor patients remotely, allowing doctors to view their health data through an Android application. The system uses a WIFI Module – ESP8266, Heartbeat Pulse sensor, Accelerometer, and LM35 temperature sensor, which are processed by an Arduino Uno. The Blynk Android application allows paramedical staff to view patient data. The system also includes an LCD module for direct patient data viewing. The power supply setup includes a step-down transformer, bridge rectifier, 7805 voltage regulator, and filter capacitor to remove ripple from DC voltages.

INDEX

CONTENTS	PAGE NO.
CHAPTER 1: INTRODUCTION	**1-4**
1.1 INTRODUCTION	1
1.2 INTERNET OF THINGS	2
1.3 PROPOSED SYSTEM/WORKING	3
1.4 OBJECTIVES	3
1.5 ADVANTAGES	4
CHAPTER 2: HARDWARE COMPONENTS	**5-22**
2.1 ARDUINO IDE	5
2.1.1 PIN DESCRIPTION	7
2.2 HEART BEAT SENSOR	9
2.3 LM35 TEMPERATURE SENSOR	10
2.4 16*2 LCD DISPLAY	11
2.4.1 PIN DISCRIPTION	12
2.5 ACCELEROMETER	13
2.6 NODE MUC-ESP266	14
2.7 BUZZER	17
2.8 TRANSFORMER	18
2.9 VOLTAGE REGULATORS	19
2.10 SWITCH	20
2.11 PRINTED CIRCUIT BOARD(PCB)	21
CHAPTER 3: SOFTWARE COMPONENTS	**23-26**
3.1 ARDUINO IDE	23
3.1.1 PROGRAM EXECUTION	24
3.2 BLYNK SERVER AND IOT PLATFORM	24
3.3 EXECUTION	26

CHAPTER 4: DESIGN AND IMPLEMENTATION **27-29**

4.1 BLOCK DIAGRAM 27

4.2 CIRCUIT DIAGRAM 28

CHAPTER 5: RESULTS & VALIDATIONS **30-31**

5.1 PROTOTYPE OF THE PROPOSED SYSTEM 30

CHAPTER 6: CONCLUSION & FUTURE SCOPE **32**

6.1 CONCLUSION 32

6.2 FUTURE SCOPE 32

REFERENCES **33**

APPENDIX CODE **34-42**

LIST OF FIGURES

FIG.NO	NAME OF THE FIGURE	PAGE NO.
2.1	ARDUINO UNO	5
2.2	HEART BEAT SNESOR	9
2.3	TEMPERATURE SENSOR	10
2.4	CIRCUIT OF LM35	10
2.5	16*2 LCD DISPLAY	11
2.6	ACCELEROMETER	14
2.7	NODE MCU	15
2.8	NODEMCU DEV KIT	16
2.9	BUZZER	17
2.10	TRANSFORMER	18
2.11	VOLTAGE REGULATOR	19
2.12	PUSH BUTTON	20
2.13	PRINTED CIRCUIT BOARD	21
3.1	ARDUINO IDE	24
4.1	BLOCK DIAGRAM	27
4.2	CIRCUIT DIAGRAM	28
5.1	PROTOTYPE MODEL	30
5.2	ACCELEROMETER OUTPUT	30
5.3	OUTPUT	31
5.4	LCD DISPLAY OUTPUT	31

CHAPTER 1

1.1 INTRODUCTION

In the early months of the COVID-19 pandemic with no designated cure or vaccine, the only way to break the infection chain is self-isolation and maintaining the physical distancing. In this article, we present a potential application of the Internet of Things (IoT) in healthcare and physical distance monitoring for pandemic situations. The IoT node tracks health parameters, including body temperature, Pulse Rate, Body Movement and blood oxygen saturation, and then updates the Smartphone app to display the user health conditions. Patient monitoring systems are considered as a part of M-health technology. These can also be named as m-health or mobile health. These systems are used for practice of medical and public health with the help of mobile devices. These monitoring systems can be used onsite or remotely. Currently, the primary usage of the IoT in healthcare can be categorized as remote monitoring and real-time health systems. Controlling and managing dire situations, such as the one in 2020 when the corona virus disease (COVID-19) took over the world, can be achieved with the help of IoT systems, without imposing severe restrictions on people and industries. Internet of Things (IoT) is an ecosystem of connected physical objects that are accessible through the internet. Internet of Things can connect devices embedded in various systems to the internet. When devices/objects can represent themselves digitally, they can be controlled from anywhere. The connectivity then helps us capture more data from more places, ensuring more ways of increasing efficiency and improving safety and IoT security. In the Patient Monitoring also needs to detect emergencies and inform medical personnel when they occur. Conventional Patient Monitoring monitors the physiological signals constantly but they are not provided to the medical personnel in real-time and in some hospital patient monitoring system not use, the problem found in such hospitals is that continuous monitoring of physiological parameters is done for ICU patients, but the monitors are local to the room in which the patient is admitted. Physician has to frequently visit the patient and asses his/her condition by analysing the measured parameter such as temperature, accelerometer, heart rate. In case of emergencies, the nurse intimates the Doctor through some means of communication like mobile phone. This can be a serious problem during emergencies because the patient’s life may be in danger if immediate attention is not provided. Another problem with conventional Patient Monitoring is that most are bulky standalone machines. Some models are connected to networks but they are usually hard-wired.

1.2 INTERNET OF THINGS

The Internet of things (IoT) is the inter-networking of physical devices, vehicles (also referred to as "connected devices" and "smart devices"), buildings, and other items— embedded with electronics, software, sensors, actuators, and network connectivity that enable these objects to collect and exchange data. In 2013 the Global Standards Initiative on Internet of Things (IoT-GSI) defined the IoT as "the infrastructure of the information society." The IoT allows objects to be sensed or controlled remotely across existing network infrastructure, creating opportunities for more direct integration of the physical world into computer-based systems, and resulting in improved efficiency, accuracy and economic benefit in addition to reduced human intervention. When IoT is augmented with sensors and actuators, the technology becomes an instance of the more general class of cyber-physical systems, which also encompasses technologies such as smart grids, smart homes, intelligent transportation and smart cities. Each thing is uniquely identifiable through its embedded computing system but is able to interoperate within the existing Internet infrastructure. Experts estimate that the IoT will consist of almost 50 billion objects by 2020. Typically, IoT is expected to offer advanced connectivity of devices, systems, and services that goes beyond machine-to-machine (M2M) communications and covers a variety of protocols, domains, and applications. The interconnection of these embedded devices (including smart objects), is expected to usher in automation in nearly all fields, while also enabling advanced applications like a smart grid, and expanding to areas such as smart cities. "Things," in the IoT sense, can refer to a wide variety of devices such as heart monitoring implants, biochip transponders on farm animals, electric clams in coastal waters, automobiles with built-in sensors, DNA analysis devices for environmental/food/pathogen monitoring or field operation devices that assist firefighters in search and rescue operations. Legal scholars suggest to look at "Things" as an "inextricable mixture of hardware, software, data and service". These devices collect useful data with the help of various existing technologies and then autonomously flow the data between other devices. Current market examples include home automation (also known as smart home devices) such as the control and automation of lighting, heating (like smart thermostat), ventilation, air conditioning (HVAC) systems, and appliances such as washer/dryers, robotic vacuums, air purifiers, ovens or refrigerators/freezers that use Wi-Fi for remote monitoring. As well as the expansion of Internet-connected automation into a plethora of new application areas, IoT is also expected to generate large amounts of data from diverse locations, with the consequent necessity for quick aggregation of the data, and an increase in the need to index,

store, and process such data more effectively. IoT is one of the platforms of today's Smart City, and Smart Energy Management Systems.

1.3 PROPOSED SYSTEM/WORKING

In this proposed system a Heartbeat Pulse sensor, Accelerometer and LM35 temperature sensor is used to monitor the patient Health. This sensor values are fed into Arduino Uno for Processing. Paramedical staff is provided with a Blynk Android Application to view each patient's Heart Beat, Blood Oxygen Body Movement and Body Temperature. This Health Monitoring device automatically connects with Local Internet Wi-Fi Router at Initial Startup and at every 15 Sec, its updates the data to Blynk based Cloud Server. An LCD Module is also provided with this system to directly view the patient health data. If the patient body temperature, crosses the predefined value or Body movement detected than a High Alert buzzer sound will be produced. Embedded C Language is used to Program this Hardware. Arduino Ide compiler along with Onboard USB Programmer used to Upload Code. Digital and Analog Pins are used to Connect all Input and Output peripherals. The power supply setup of the system contains a step-down transformer of 230/12V, used to step down the voltage to 12VAC. To convert it to DC, a bridge rectifier is used. 7805 voltage regulator is used regulate 12V Dc to +5V that will be needed for microcontroller and other components operation. Filter Capacitor are used to remove ripple from DC Voltages.

The project is designed and developed for monitoring patients remotely using a IoT based wireless communication system. The main aim of this project is to monitor the HEART Beat, Blood Oxygen content, Body Movement and Body Temperature of the patient and display the same to the doctor through Internet based server using Thing Speak cloud Server. In hospitals, patients' body temperature must need to be monitored constantly, which is usually done by doctors or other paramedical staff. They observe the body temperature and Heart Beat of patients' constantly and maintain a record of it.

OBJECTIVES

Here the main objective is to design a Remote Patient Health Monitoring System to diagnose the health condition of the patients. Giving care and health assistance to the bedridden patients at critical stages with advanced medical facilities have become one of the major problems in the modern hectic world. In hospitals where many patients whose physical conditions must be monitored frequently as a part of a diagnostic procedure, the need for a cost-effective and fast responding alert mechanism is inevitable. Proper implementation of such systems can provide

timely warnings to the medical staffs and doctors and their service can be activated in case of medical emergencies. Present-day systems use sensors that are hardwired to a PC next to the bed. 5 The use of sensors detects the conditions of the patient and the data is collected and transferred using a microcontroller. In addition to this, use of multiple microcontrollers based intelligent system provides high-level applicability in hospitals where many patients must be frequently monitored. For this, here we use the idea of network technology with wireless applicability, providing each patient a unique ID by which the doctor can easily identify the patient and his/her status of health parameters.

ADVANTAGES

- IOT Monitoring proves really helpful when we need to monitor & record and keep track of changes in the health parameters of the patient over the period of time. So, with the IOT health monitoring, we can have the database of these changes in the health parameters. Doctors can take the reference of these changes or the history of the patient while suggesting the treatment or the medicines to the patient.
- Hospital stays are minimized due to Remote Patient Monitoring.
- Hospital visits for normal routine checkups are minimized.

HARDWARE REQUIREMENTS

- ARDUINO Microcontroller Board
- LCD 2X16 Module
- Temperature Sensor
- HEART Beat Sensor
- Accelerometer
- Buzzer
- Pulse Oximeter
- WIFI Module – ESP8266
- Transformer
- Voltage Regulators
- Filter Capacitors
- Other Misc. Electronics Component

SOFTWARE REQUIREMENTS

- Embedded C Programming Language
- Arduino IDE Compiler
- Blynk Cloud Server

CHAPTER 2
HARDWARE COMPONENTS

2.1 ARDUINO UNO

Arduino is common term for a software company, project, and user community that designs and manufactures computer open-source hardware, open-source software, and microcontroller-based kits for building digital devices and interactive objects that can sense and control physical devices.

Fig 2.1 ARDUINO UNO

The project is based on microcontroller board designs, produced by several vendors, using various microcontrollers. These systems provide sets of digital and analog I/O pins that can interface to various expansion boards (termed shields) and other circuits. The boards feature serial communication interfaces, including Universal Serial Bus (USB) on some models, for loading programs from personal computers. For programming the microcontrollers, the Arduino project provides an integrated development environment (IDE) based on a programming language named Processing, which also supports the languages C and C++.

The first Arduino was introduced in 2005, aiming to provide a low cost, easy way for novices and professionals to create devices that interact with their environment using sensors and actuators. Common examples of such devices intended for beginner hobbyists include simple robots, thermostats, and motion detectors.

Arduino programs may be written in any programming language with a compiler that produces

binary machine code. The Arduino project provides the Arduino integrated development environment (IDE), which is a cross-platform application written in the programming language Java. It originated from the IDE for the languages Processing and Wiring. 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 provides simple one-click mechanism to compile and load programs to an Arduino board. A program written with the IDE for Arduino is called a "sketch".

The Arduino IDE supports the languages C and C++ using special rules to organize code. The Arduino IDE supplies a software library called Wiring from the Wiring project, which provides many common input and output procedures. A typical Arduino C/C++ sketch consist of two functions that are compiled and linked with a program stub main() into an executable cyclic executive program:

- *setup(): a function that runs once at the start of a program and that can initialize settings.*
- *loop(): a function called repeatedly until the board powers off.*

After compiling and linking with the GNU toolchain, also included with the IDE distribution, the Arduino IDE employs the program argued to convert the executable code into a text file in hexadecimal coding that is loaded into the Arduino board by a loader program in the board's firmware.

Applications

See also: List of open-source hardware projects

- Xoscillo, an open-source oscilloscope
- Scientific equipment such as the Chemduino
- Arduinome, a MIDI controller device that mimics the Monome
- OBDuino, a trip computer that uses the on-board diagnostics interface found in most modern cars • Ardupilot, drone software and hardware
- ArduinoPhone, a do-it-yourself cellphone
- GertDuino, an Arduino mate for the Raspberry Pi
- Water quality testing platform
- Homemade CNC using Arduino and DC motors with close loop control by Homofaciens

2.1.1 PIN DESCRIPTION

Power (USB / Barrel Jack)

Every Arduino board needs a way to be connected to a power source. The Arduino UNO can be powered from a USB cable coming from your computer or a wall power supply (like this) that is terminated in a barrel jack. In the picture above the USB connection is labelled (1) and the barrel jack is labeled (2).

The USB connection is also how you will load code onto your Arduino board. More on how to program with Arduino can be found in our Installing and Programming Arduino tutorial.

Pins (5V, 3.3V, GND, Analog, Digital, PWM, AREF)

The pins on your Arduino are the places where you connect wires to construct a circuit. They usually have black plastic 'headers' that allow you to just plug a wire right into the board. The Arduino has several different kinds of pins, each of which is labeled on the board and used for different functions.

GND (3): Short for 'Ground'. There are several GND pins on the Arduino, any of which can be used to ground your circuit.

5V (4) & 3.3V (5): As you might guess, the 5V pin supplies 5 volts of power, and the 3.3V pin supplies 3.3 volts of power. Most of the simple components used with the Arduino run happily off of 5 or 3.3 volts.

Analog (6): The area of pins under the 'Analog In' label (A0 through A5 on the UNO) are Analog In pins. These pins can read the signal from an analog sensor (like a temperature sensor) and convert it into a digital value that we can read.

Digital (7): Across from the analog pins are the digital pins (0 through 13 on the UNO). These pins can be used for both digital input (like telling if a button is pushed) and digital output (like powering an LED).

PWM (8): You may have noticed the tilde (~) next to some of the digital pins (3, 5, 6, 9, 10, and 11 on the UNO). These pins act as normal digital pins, but can also be used for something called Pulse-Width Modulation (PWM). We have a tutorial on PWM, but for now, think of these pins as being able to simulate analog output (like fading an LED in and out).

AREF (9): Stands for Analog Reference. Most of the time you can leave this pin alone. It is sometimes used to set an external reference voltage (between 0 and 5 Volts) as the upper limit for the analog input pins.

Reset Button

Just like the original Nintendo, the Arduino has a reset button (10). Pushing it will temporarily connect the reset pin to ground and restart any code that is loaded on the Arduino. This can be very useful if your code doesn't repeat, but you want to test it multiple times. Unlike the original Nintendo however, blowing on the Arduino doesn't usually fix any problems.

Power LED Indicator

Just beneath and to the right of the word "UNO" on your circuit board, there's a tiny LED next to the word 'ON' (11). This LED should light up whenever you plug your Arduino into a power source. If this light doesn't turn on, there's a good chance something is wrong. Time to re-check your circuit!

TX RX LEDs

TX is short for transmit; RX is short for receive.

Main IC

The black thing with all the metal legs is an IC, or Integrated Circuit (13). Think of it as the brains of our Arduino. The main IC on the Arduino is slightly different from board type to board type, but is usually from the Atmega line of ICs from the ATMEL company. This can be important, as you may need to know the IC type (along with your board type) before loading up a new program from the Arduino software. This information can usually be found in writing on the top side of the IC. If you want to know more about the difference between various IC's, reading the datasheets is often a good idea.

Voltage Regulator

The voltage regulator (14) is not actually something you can (or should) interact with on the Arduino. But it is potentially useful to know that it is there and what it's for. The voltage regulator controls the amount of voltage that is let into the Arduino board. Think of it as a kind of gatekeeper; it will turn away an extra voltage that might harm the circuit. Of course, it has its limits, so don't hook up your Arduino to anything greater than 20 volts.

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

There are two types of photoplethysmography:

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

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

Here we are using Transmission type Heart beat sensor

Heart beat sensor is designed to give digital output of heat beat when a finger is placed on it. When the heart beat detector is working, the beat LED flashes in unison with each heartbeat. This digital output can be connected to microcontroller directly to measure the Beats Per Minute (BPM) rate. It works on the principle of light modulation by blood flow through finger at each pulse.

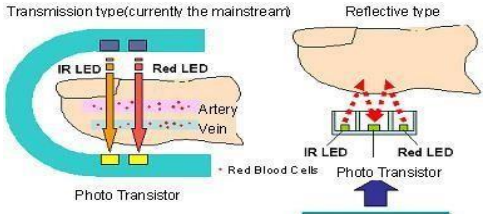


FIG 2.2 HEART BEAT SENSOR

The sensor consists of a Infrared Transmitter and Infrared Receiver. Now, when the heart pumps a pulse of blood through the blood vessels, the finger becomes slightly more opaque and so less light reached the detector. With each heart pulse the detector signal varies. This variation is converted to electrical pulse. This signal is amplified and triggered through an amplifier which outputs +5V logic level signal. The output signal is also indicated by a LED which blinks on each heartbeat.

2.3 LM35 TEMPERATURE SENSOR

LM35 is a precision IC **temperature sensor** with its output proportional to the temperature (in ºC). The sensor circuitry is sealed and therefore it is not subjected to oxidation and other processes. With **LM35**, temperature can be measured more accurately than with a Thermistor. It also possesses low self-heating and does not cause more than 0.1 ºC temperature rise in still air.

The operating temperature range is from -55°C to 150°C. The output voltage varies by 10mV in response to every ºC rise/fall in ambient temperature, *i.e.*, its scale factor is 0.01V/ ºC.

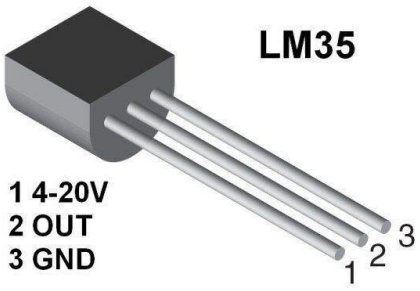


FIG 2.3 LM35 TEMPERATURE SENSOR

Pin Description:

Pin No	Function	Name
1	Supply voltage; 5V (+35V to -2V)	Vcc
2	Output voltage (+6V to -1V)	Output
3	Ground (0V)	Ground

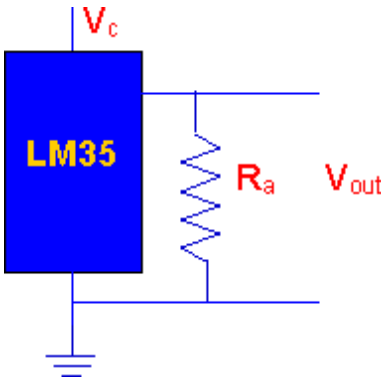


FIG 2.4 CIRCUIT OF LM35

2.4 16*2 LCD DISPLAY

A **liquid-crystal display** (**LCD**) is a flat-panel display or other electronic visual display that uses the light-modulating properties of liquid crystals. Liquid crystals do not emit light directly. LCDs are used in a wide range of applications including computer monitors, televisions, instrument panels, aircraft cockpit displays, and signage. They are common in consumer devices such as DVD players, gaming devices, clocks, watches, calculators, and telephones, and have replaced cathode ray tube (CRT) displays in nearly all applications. They are available in a wider range of screen sizes than CRT and plasma displays, and since they do not use phosphors, they do not suffer image burn-in. LCDs are, however, susceptible to image persistence.

The LCD screen is more energy-efficient and can be disposed of more safely than a CRT. Its low electrical power consumption enables it to be used in battery-powered electronic equipment more efficiently than CRTs. It is an electronically modulated optical device made up of any number of segments controlling a layer of liquid crystals and arrayed in front of a light source (backlight) or reflector to produce images in color or monochrome.

LCD (Liquid Crystal Display) screen is an electronic display module and find a wide range of applications. A 16x2 LCD display is very basic module and is very commonly used in various devices and circuits. These modules are preferred over seven segments and other multi segment LEDs. The reasons being: LCDs are economical; easily programmable; have no limitation of displaying special & even custom characters (unlike in seven segments), animations and so on.

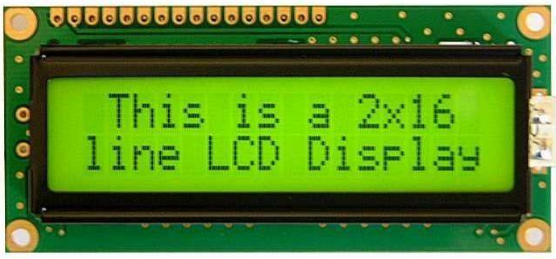


FIG 2.5 16*2 LCD DISPLAY

A **16x2 LCD** means it can display 16 characters per line and there are 2 such lines. In this LCD each character is displayed in 5x7 pixel matrix. This LCD has two registers, namely, Command and Data.

The command register stores the command instructions given to the LCD. A command is an instruction given to LCD to do a predefined task like initializing it, clearing its screen, setting the cursor position, controlling display etc. The data register stores the data to be displayed on the LCD. The data is the ASCII value of the character to be displayed on the LCD.

2.4.1 PIN DESCRIPTION

Pin No	Function	Name
1	Ground (0V)	Ground
2	Supply voltage; 5V (4.7V – 5.3V)	Vcc
3	Contrast adjustment; through a variable resistor	VEE
4	Selects command register when low; and data register when high	Register Select
5	Low to write to the register; High to read from the register	Read/write
6	Sends data to data pins when a high to low pulse is given	Enable
7	8-bit data pins	DB0
8		DB1
9		DB2
10		DB3
11		DB4
12		DB5
13		DB6
14		DB7
15	Backlight V_{CC} (5V)	Led+
16	Backlight Ground (0V)	Led-

2.5 ACCELEROMETER

An accelerometer is a device that measures proper acceleration.[1] Proper acceleration, being the acceleration (or rate of change of velocity) of a body in its own instantaneous rest frame, is not the same as coordinate acceleration, being the acceleration in a fixed coordinate system. For example, an accelerometer at rest on the surface of the Earth will measure an acceleration due to Earth's gravity, straight upwards (by definition) of $g \approx 9.81\ m/s^2$. By contrast, accelerometers in free fall (falling toward the center of the Earth at a rate of about $9.81\ m/s^2$) will measure zero.

Accelerometers have multiple applications in industry and science. Highly sensitive accelerometers are components of inertial navigation systems for aircraft and missiles. Accelerometers are used to detect and monitor vibration in rotating machinery. Accelerometers are used in tablet computers and digital cameras so that images on screens are always displayed upright. Accelerometers are used in drones for flight stabilisation. Coordinated accelerometers can be used to measure differences in proper acceleration, particularly gravity, over their separation in space; i.e., gradient of the gravitational field. This gravity gradiometry is useful because absolute gravity is a weak effect and depends on local density of the Earth which is quite variable.

Single- and multi-axis models of accelerometer are available to detect magnitude and direction of the proper acceleration, as a vector quantity, and can be used to sense orientation (because direction of weight changes), coordinate acceleration, vibration, shock, and falling in a resistive medium (a case where the proper acceleration changes, since it starts at zero, then increases). Micromachined microelectromechanical systems (MEMS) accelerometers are increasingly present in portable electronic devices and video game controllers, to detect the position of the device or provide for game input.

Conceptually, an accelerometer behaves as a damped mass on a spring. When the accelerometer experiences an acceleration, the mass is displaced to the point that the spring is able to accelerate the mass at the same rate as the casing. The displacement is then measured to give the acceleration.

In commercial devices, piezoelectric, piezoresistive and capacitive components are commonly used to convert the mechanical motion into an electrical signal. Piezoelectric accelerometers rely on piezoceramics (e.g. lead zirconate titanate) or single crystals (e.g. quartz, tourmaline).

They are unmatched in terms of their upper frequency range, low packaged weight and high temperature range. Piezoresistive accelerometers are preferred in high shock applications. Capacitive accelerometers typically use a silicon micro-machined sensing element. Their performance is superior in the low frequency range and they can be operated in servo mode to achieve high stability and linearity.

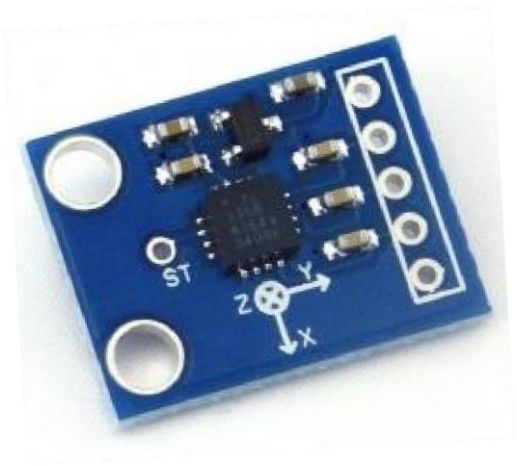


FIG 2.6 ACCELEROMETER

Modern accelerometers are often small *micro electro-mechanical systems* (MEMS), and are indeed the simplest MEMS devices possible, consisting of little more than a cantilever beam with a proof mass (also known as *seismic mass*). Damping results from the residual gas sealed in the device. As long as the Q-factor is not too low, damping does not result in a lower sensitivity.

2.6 NODE MCU – ESP8266

NodeMCU is an open-source LUA based firmware developed for ESP8266 Wi-Fi chip. By exploring functionality with ESP8266 chip, NodeMCU firmware comes with ESP8266 Development board/kit i.e. NodeMCU Development board. Since NodeMCU is open-source platform, their hardware design is open for edit/modify/build. NodeMCU Dev Kit/board consist of ESP8266 Wi-Fi enabled chip. The ESP8266 is a low-cost Wi-Fi chip developed by Espressif Systems with TCP/IP protocol. For more information about ESP8266, you can refer ESP8266 Wi-Fi Module. There is Version2 (V2) available for NodeMCU Dev Kit i.e. NodeMCU Development Board v1.0 (Version2), which usually comes in black colored PCB.

Fig 2.7 NODE MCU

NodeMCU Dev Kit has Arduino like Analog (i.e. A0) and Digital (D0-D8) pins on its board. It supports serial communication protocols i.e. UART, SPI, I2C etc. Using such serial protocols, we can connect it with serial devices like I2C enabled LCD display, Magnetometer HMC5883, MPU-6050 Gyro meter + Accelerometer, RTC chips, GPS modules, touch screen displays, SD cards etc. NodeMCU Development board is featured with wifi capability, analog pin, digital pins and serial communication protocols. To get start with using NodeMCU for IoT applications first we need to know about how to write/download NodeMCU firmware in NodeMCU Development Boards. And before that where this NodeMCU firmware will get as per our requirement. There is online NodeMCU custom builds available using which we can easily get our custom NodeMCU firmware as per our requirement.

DEVELOPMENT BOARD

NodeMCU Development Kit/Board consist of ESP8266 Wi-Fi chip. ESP8266 chip has GPIO pins, serial communication protocol, etc. features on it.

ESP8266 is a low-cost Wi-Fi chip developed by Espressif Systems with TCP/IP protocol. For more information about ESP8266, you can refer ESP8266 Wi-Fi Module.

The features of ESP8266 are extracted on NodeMCU Development board. NodeMCU (LUA based firmware) with Development board/kit that consist of ESP8266 (Wi-Fi enabled chip) chip combines NodeMCU Development board which make it stand-alone device in IoT applications.

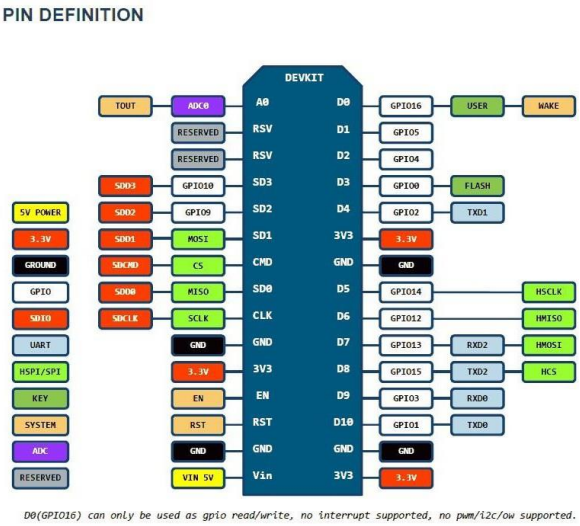


FIG. 2.8 NODEMCU DEV KIT

NodeMCU Dev Kit v1.0 pin descriptions

GPIO (General Purpose Input Output) Pins:

NodeMCU has general purpose input output pins on its board as shown in above pinout diagram. We can make it digital high/low and control things like LED or switch on it. Also, we can generate PWM signal on these GPIO pins.

ADC (Analog to Digital Converter) channel (A0):

NodeMCU has one ADC channel/pin on its board.

SPI (Serial Peripheral Interface) Pins:

NodeMCU based ESP8266 has Hardware SPI (HSPI) with four pins available for SPI communication. It also has SPI pins for Quad-SPI communication. With this SPI interface, we can connect any SPI enabled device with NodeMCU and make communication possible with it.

I2C (Inter-Integrated Circuit) Pins:

NodeMCU has I2C functionality support on ESP8266 GPIO pins. Due to internal functionality on ESP-12E we cannot use all its GPIOs for I2C functionality. So, do tests before using any GPIO for I2C applications.

UART (Universal Asynchronous Receiver Transmitter) Pins:

NodeMCU based ESP8266 has two UART interfaces, UART0 and UART1. Since UART0 (RXD0 & TXD0) is used to upload firmware/codes to board, we can’t use them in applications while uploading firmware/codes.

2.7 BUZZER

A buzzer or beeper is an audio signaling device, which may be mechanical, electromechanical, or piezoelectric. Typical uses of buzzers and beepers include alarm devices, timers and confirmation of user input such as a mouse click or keystroke.

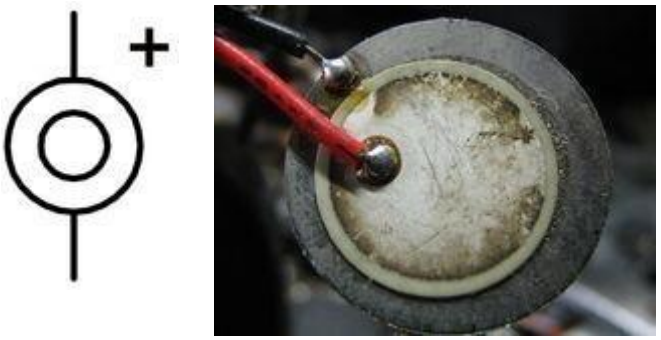
TYPE OF BUZZERS

ELECTROMECHANICAL

Early devices were based on an electromechanical system identical to an electric bell without the metal gong. Similarly, a relay may be connected to interrupt its own actuating current, causing the contacts to buzz. Often these units were anchored to a wall or ceiling to use it as a sounding board. The word "buzzer" comes from the rasping noise that electromechanical buzzers made.

MECHANICAL

A joy buzzer is an example of a purely mechanical buzzer. They require drivers.



Piezoelectric disk beeper Piezoelectric

FIG. 2.9 BUZZER

A piezoelectric element may be driven by an oscillating electronic circuit or other audio signal source, driven with a piezoelectric audio amplifier. Sounds commonly used to indicate that a button has been pressed are a click, a ring or a beep.

2.8 TRANSFORMER

A transformer is an electrical device that transfers electrical energy between two or more circuits through electromagnetic induction. Electromagnetic induction produces an electromotive force across a conductor which is exposed to time varying magnetic fields. Commonly, transformers are used to increase or decrease the voltages of alternating current in electric power applications.

A varying current in the transformer's primary winding creates a varying magnetic flux in the transformer core and a varying magnetic field impinging on the transformer's secondary winding. This varying magnetic field at the secondary winding induces a varying electromotive force (EMF) or voltage in the secondary winding due to electromagnetic induction.

FIG .2.10 TRANSFORMER

A Transformer takes in electricity at a higher voltage and lets it run through lots of coils wound around an iron core. ". A single-phase Transformer can operate to either increase or decrease the voltage applied to the primary winding. Because the current is alternating, the magnetism in the core is also alternating. Also around the core is an output wire with fewer coils. The magnetism changing back and forth makes a current in the wire. Having fewer coils means less voltage. When it is used to "decrease" the voltage on the secondary winding with respect to the primary it is called a Step-down Transformer. When a Transformer is used to "increase" the voltage on its secondary winding with respect to the primary, it is called a Step-up Transformer.

Applications

Transformers are used to increase (or step-up) voltage before transmitting electrical energy over long distances through wires. Wires have resistance which loses energy through joule heating at a rate corresponding to square of the current. By transforming power to higher voltage transformers enable economical transmission of power and distribution. Consequently, transformers have shaped the electricity supply industry, permitting generation to be located remotely from points of demand. All

but a tiny fraction of the world's electrical power has passed through a series of transformers by the time it reaches the consumer.

2.9 VOLTAGE REGULATORS

A voltage regulator is designed to automatically maintain a constant voltage level. A voltage regulator may be a simple "feed-forward" design or may include negative feedback control loops. It may use an electromechanical mechanism, or electronic components. Depending on the design, it may be used to regulate one or more AC or DC voltages.

FIG .2.11 VOLTAGE REGULATOR

Electronic voltage regulators are found in devices such as computer power supplies where they stabilize the DC voltages used by the processor and other elements. In automobile alternators and central power station generator plants, voltage regulators control the output of the plant. In an electric power distribution system, voltage regulators may be installed at a substation or along distribution lines so that all customers receive steady voltage independent of how much power is drawn from the line.

The 78xx (sometimes L78xx, LM78xx, MC78xx...) is a family of self-contained fixed linear voltage regulator integrated circuits. The 78xx family is commonly used in electronic circuits requiring a regulated power supply due to their ease-of-use and low cost. For ICs within the family, the xx is replaced with two digits, indicating the output voltage (for example, the 7805 has a 5-volt output, while the 7812 produces 12 volts). The 78xx line are positive voltage regulators: they produce a voltage that is positive relative to a common ground. There is a related line of 79xx devices which are complementary negative voltage regulators. 78xx and 79xx ICs can be used in combination to provide positive and negative supply voltages in the same circuit.

78xx ICs have three terminals and are commonly found in the TO-220 form factor, although they are

available in surface-mount, TO-92, and TO-3 packages. These devices support an input voltage anywhere from around 2.5 volts over the intended output voltage up to a maximum of 35 to 40 volts depending on the model, and typically provide 1 or 1.5 amperes of current (though smaller or larger packages may have a lower or higher current rating).

Pin Description:

Pin No	Function	Name
1	Input voltage (5V-18V)	Input
2	Ground (0V)	Ground
3	Regulated output; 5V (4.8V-5.2V)	Output

2.10 SWITCH

A Switch is an electrical component that can disconnect or connect the conducting path in an electrical circuit, interrupting the electric current or diverting it from one conductor to another. The most common type of switch is an electromechanical device consisting of one or more sets of movable electrical contacts connected to external circuits. When a pair of contacts is touching current can pass between them, while when the contacts are separated no current can flow.

FIG.2.12 PUSH BUTTON

Switches are made in many different configurations; they may have multiple sets of contacts controlled by the same knob or actuator, and the contacts may operate simultaneously, sequentially, or alternately. A switch may be operated manually, for example, a light switch or a keyboard button, or may function as a sensing element to sense the position of a machine part, liquid level, pressure, or temperature, such as a thermostat. Many specialized forms exist, such as the toggle switch, rotary switch, mercury switch, pushbutton switch, reversing switch, relay, and circuit breaker. A common use is control of lighting, where multiple switches may be wired into one circuit to allow convenient control of light fixtures. Switches in high-powered circuits must have special construction to prevent destructive arcing when they are opened.

A push button is a momentary or non-latching switch which causes a temporary change in the state of an electrical circuit only while the switch is physically actuated. An automatic mechanism (i.e. a spring) returns the switch to its default position immediately afterwards, restoring the initial circuit condition. There are two types:

- A push to make switch allows electricity to flow between its two contacts when held in. When the button is released, the circuit is broken. This type of switch is also known as a Normally Open (NO) Switch. (Examples: doorbell, computer case power switch, calculator buttons, individual keys on a keyboard)
- A push to break switch does the opposite, i.e. when the button is not pressed, electricity can flow, but when it is pressed the circuit is broken. This type of switch is also known as a Normally Closed (NC) Switch. (Examples: Fridge Light Switch, Alarm Switches in Fail-Safe circuits)

Many Push switches are designed to function as both push to make and push to break switches. For these switches, the wiring of the switch determines whether the switch functions as a push to make or as a push to break switch.

2.11 PRINTED CIRCUIT BOARD (PCB)

A printed circuit board (PCB) mechanically supports and electrically connects electrical or electronic components using conductive tracks, pads and other features etched from one or more sheet layers of copper laminated onto and/or between sheet layers of a non-conductive substrate. Components are generally soldered onto the PCB to both electrically connect and mechanically fasten them to it.

Printed circuit boards are used in all but the simplest electronic products. They are also used in some electrical products, such as passive switch boxes.

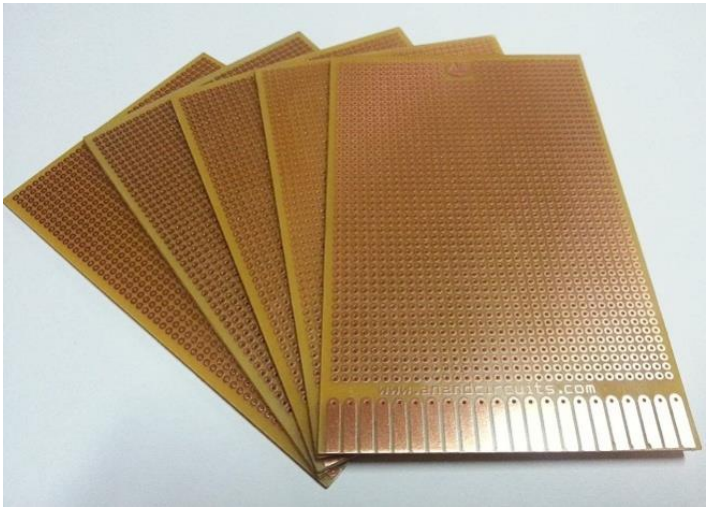


FIG.2.13 PRINTED CIRCUIT BOARD

Alternatives to PCBs include wire wrap and point-to-point construction, both once popular but now rarely used. PCBs require additional design effort to lay out the circuit, but manufacturing and assembly can be automated. Electronic computer-aided design software is available to do much of the work of layout. Mass-producing circuits with PCBs is cheaper and faster than with other wiring methods, as components are mounted and wired in one operation. Large numbers of PCBs can be fabricated at the same time, and the layout only has to be done once. PCBs can also be made manually in small quantities, with reduced benefits.

make repair, analysis, and field modification of circuits much more difficult and usually impractical.

A printed circuit board can have multiple copper layers. A two-layer board has copper on both sides; multi-layer boards sandwich additional copper layers between layers of insulating material. Conductors on different layers are connected with vias, which are copper-plated holes that function as electrical tunnels through the insulating substrate. Through-hole component leads sometimes also effectively function as vias. After two-layer PCBs, the next step up is usually four-layer. Often two layers are dedicated as power supply and ground planes, and the other two are used for signal wiring between components.

"Through hole" components are mounted by their wire leads passing through the board and soldered to traces on the other side. "Surface mount" components are attached by their leads to copper traces on the same side of the board. A board may use both methods for mounting components. PCBs with only through-hole mounted components are now uncommon. Surface mounting is used for transistors, diodes, IC chips, resistors and capacitors. Through-hole mounting may be used for some large components such as electrolytic capacitors and connectors.

CHAPTER 3
SOFTWARE COMPONENTS

3.1 ARDUINO IDE

The Arduino integrated development environment (IDE) is a crossplatform application (for Windows, macOS, Linux) that is written in the programming language Java. It is used to write and upload programs to Arduino compatible boards, but also, with the help of 3rd party cores, other vendor development boards.

The source code for the IDE is released under the GNU General Public License, version 2.The Arduino IDE supports the languages C and C++ using special rules of code structuring. The Arduino IDE supplies a software library from the Wiring project, which provides many common input and output procedures. User-written code only requires two basic functions, for starting the sketch and the main program loop, that are compiled and linked with a program stub main() into an executable cyclic executive program with the GNU tool chain, also included with the IDE distribution. The Arduino IDE employs the program argued to convert the executable code into a text file in hexadecimal encoding that is loaded into the Arduino board by a loader program in the board's firmware.

Arduino is a prototype platform (open-source) based on an easy-to-use hardware and software. It consists of a circuit board, which can be programmed (referred to as a microcontroller) and a ready-made software called Arduino IDE (Integrated Development Environment), which is used to write and upload the computer code to the physical board.

Arduino provides a standard form factor that breaks the functions of the micro-controller into a more accessible package.

Audience This tutorial is intended for enthusiastic students or hobbyists. With Arduino, one can get to know the basics of micro-controllers and sensors very quickly and can start building prototype with very little investment.

This tutorial is intended to make you comfortable in getting started with Arduino and its various functions.

Prerequisites Before you start proceeding with this tutorial, we assume that you are already familiar with the basics of C and C++. If you are not well aware of these concepts, then we will suggest you go through our short tutorials on C and C++. A basic understanding of microcontrollers and electronics is also expected.

3.1.1 PROGRAM EXECUTION

Window will look something like this

- Click the upload button or Ctrl+U to compile the program and load on the Raspberry Pi Pico board.

- Click the serial monitor button. If all has gone well, the monitor window will show your message and look something like this program, paying attention to where semi-colons appear at the end of command lines.

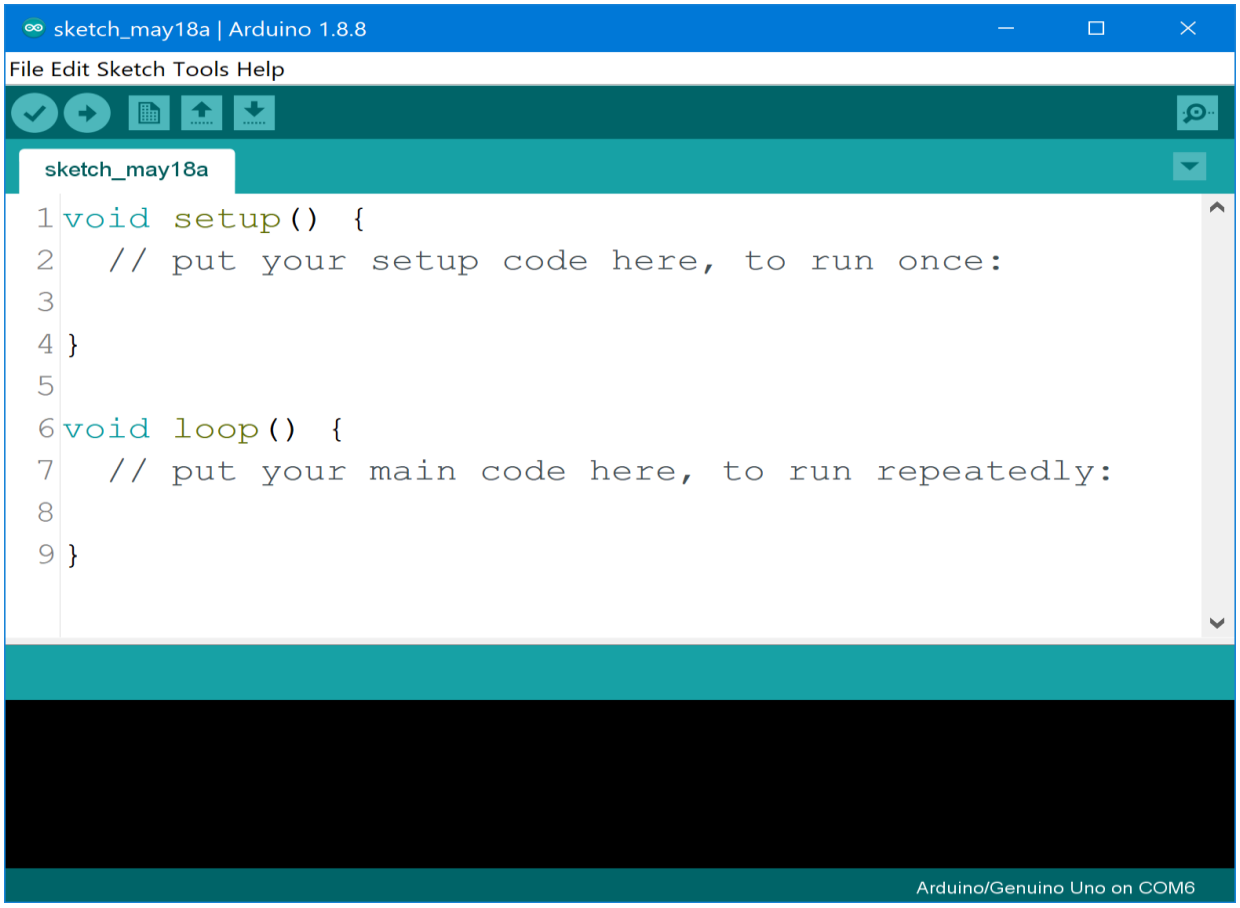


FIG. 3.1 ARDUINO IDE

3.2 Blynk Server and IoT Platform

Blynk is a Platform with iOS and Android apps to control Arduino, Raspberry Pi and the likes over the Internet.

It's a digital dashboard where you can build a graphic interface for your project by simply dragging and dropping widgets.

It's really simple to set everything up and you'll start tinkering in less than 5 mins.

Blynk is not tied to some specific board or shield. Instead, it's supporting hardware of your choice. Whether your Arduino or Raspberry Pi is linked to the Internet over Wi-Fi, Ethernet or this new ESP8266 chip, Blynk will get you online and ready for the Internet Of Your Things.

Create a Blynk Project

Click the “Create New Project” in the app to create a new Blynk app. Give it any name.

Blynk works with hundreds of hardware models and connection types. Select the Hardware type. After this, select connection type. In this project we have select WiFi connectivity.

The Auth Token is very important – you’ll need to stick it into your ESP8266’s firmware. For now, copy it down or use the “E-mail” button to send it to yourself.

Widgets To The Project

Then you’ll be presented with a blank new project. To open the widget box, click in the project window to open.

We are selecting a button to control Led connected with NodeMCU.

1.Click on Button.

2.Give name to Button say led.

3.Under OUTPUT tab- Click pin and select the pin to which led is connected to NodeMCU, here it is digital pin 2, hence select digital and under pin D2. And Click continue.

Under MODE tab- Select whether you want this button as "push button" or "Switch".

You have successfully created a GUI for Arduino

Upload The Firmware

Now that your Blynk project is set-up, open Arduino and navigate to the ESP8266_Standalone example in the File > Examples > Blynk > Boards WIFI> ESP8266_Standalone menu.

Stand Alone Programming Code:

Before uploading, make sure to paste your authorization token into the auth [] variable. Also make sure to load your Wi-Fi network settings into the Blynk.begin(auth, "ssid", "pass") function.

```
#define BLYNK_PRINT Serial
#include <ESP8266WiFi.h>
#include <BlynkSimpleEsp8266.h>

// You should get Auth Token in the Blynk App.
// Go to the Project Settings (nut icon).
char auth[] = "YourAuthToken";
```

```
// Your WiFi credentials.
// Set password to "" for open networks.
char ssid[] = "YourNetworkName";
char pass[] = "YourPassword";
void setup()
{
  // Debug console
  Serial.begin(9600);
  Blynk.begin(auth, ssid, pass);
}
void loop()
{
  Blynk.run();
}
```

3.3 Execution

After the app has uploaded, open the serial monitor, setting the baud rate to 9600. Wait for the “Ready” message.

Then click the “Run” button in the top right corner of the Blynk app. Press the button and watch the LED

Then add more widgets to the project. They should immediately work on the ESP8266 without uploading any new firmware.

CHAPTER 4
DESIGN AND IMPLEMENTATION

4.1 BLOCK DIAGRAM

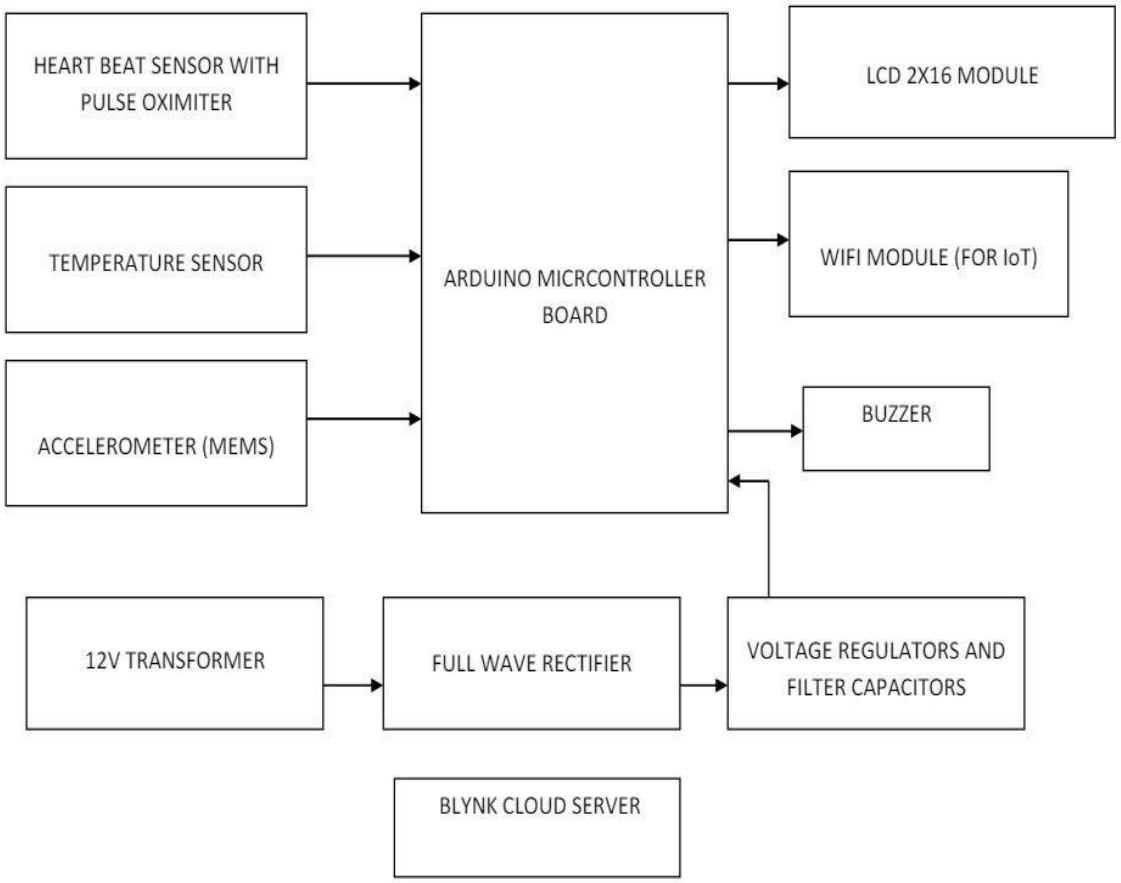


FIG. 4.1 BLOCK DIAGRAM

The above block diagram shows a patient health monitoring system. The system uses various sensors to collect health data from the patient, which is then processed by a microcontroller and transmitted wirelessly via Wi-Fi to a cloud server.

Here's a breakdown of the components listed in the block diagram:

- **Heart beat sensor with pulse oximeter:** This sensor measures the patient's heart rate and blood oxygen levels.
- **Temperature sensor:** This sensor measures the patient's body temperature.
- **Accelerometer:** This sensor measures the patient's movement and activity levels.

- **Microcontroller board (Arduino):** The Arduino microcontroller collects data from the various sensors, processes the data, and transmits it wirelessly.
- **Full wave rectifier:** This circuit component converts alternating current (AC) from the power source into direct current (DC) for the rest of the circuit.
- **Voltage regulators and filter capacitors:** These components provide a clean and stable power supply for the various parts of the circuit.
- **LCD display:** The LCD display shows the patient's health data collected by the sensors.
- **Wi-Fi module:** This module enables the microcontroller to transmit data wirelessly to a cloud server.
- **BLYNK cloud server:** This is a cloud platform that stores and displays the patient's health data.

This type of patient health monitoring system can be used in a variety of settings, including hospitals, clinics, and home care. The system can help healthcare providers to monitor patients remotely and intervene quickly in case of any emergencies.

4.2 CIRCUIT DIAGRAM

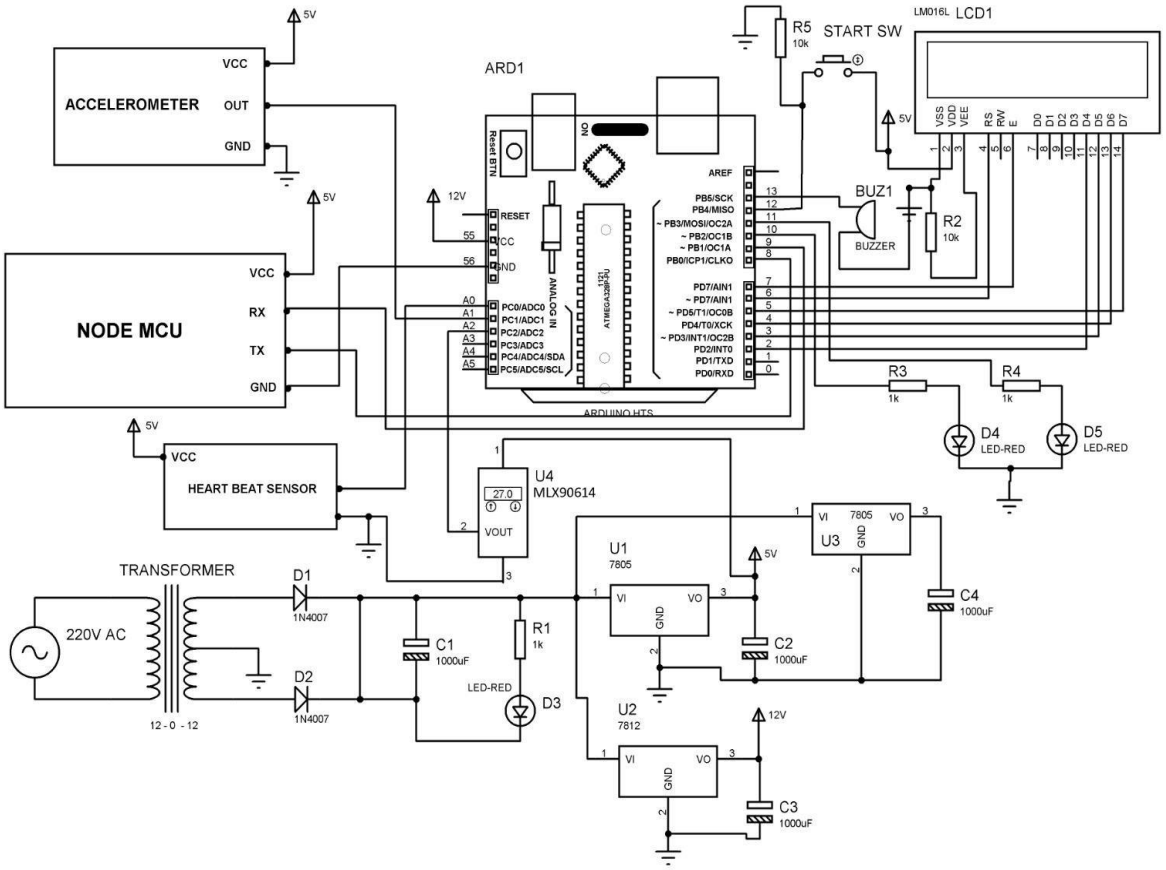


FIG. 4.2 CIRCUIT DIAGRAM

The above diagram shows the circuit diagram for a patient health monitoring system with temperature sensor, accelerometer, LCD display, and Wi-Fi module, but it uses an Arduino. Here's a breakdown of the connections between the components based on the labels in the image:

- **Microcontroller (ATmega328):**
    - The heartbeat sensor connects to analog input pin A0.
    - The temperature sensor connects to analog input pin A1.
    - The accelerometer connects to analog input pin A2.
    - The LCD display (LCD1) connects to digital pins D4 through D7.
    - Wi-Fi module (ESP-8266) connects to digital pins RX (D0) and TX (D1).
- **Voltage regulators (U2, U3):** These 7805 voltage regulators provide regulated 5V power to the circuit.
- **Voltage regulator (U3):** The 7812 voltage regulator provides regulated 12V power to the circuit, possibly for the Wi-Fi module.
- **Reset button (START SW):** This button is used to reset the microcontroller.
- **Crystal (XTAL):** The crystal provides a precise clock signal for the microcontroller.
- **Resistors:** There are several resistors (R1-R5) with various values throughout the circuit. They are used to limit current flow in various parts of the circuit.
- **Capacitors:** There are several capacitors (C1-C4) with various values throughout the circuit. They store electrical energy and help to smooth out the DC voltage from the rectifier diodes.
- **Power Supply:** The circuit is powered by a transformer (T1) that steps down the AC mains voltage, likely 220V AC, to a lower voltage. Rectifier diodes (D1 and D2) convert the AC voltage from the transformer into DC voltage. Capacitors (C1, C2, C3 and C4) store electrical energy and help to smooth out the DC voltage from the rectifier diodes.

CHAPTER 5
RESULTS &VALIDATIONS

5.1 PROTOTYPE OF THE PROPOSED SYSTEM

The prototype is successfully designed and tested. The objectives of the project are satisfactorily realized. Following are the major results obtained. An advanced system is designed for patient health monitoring system to continuously monitor the patient health parameters.

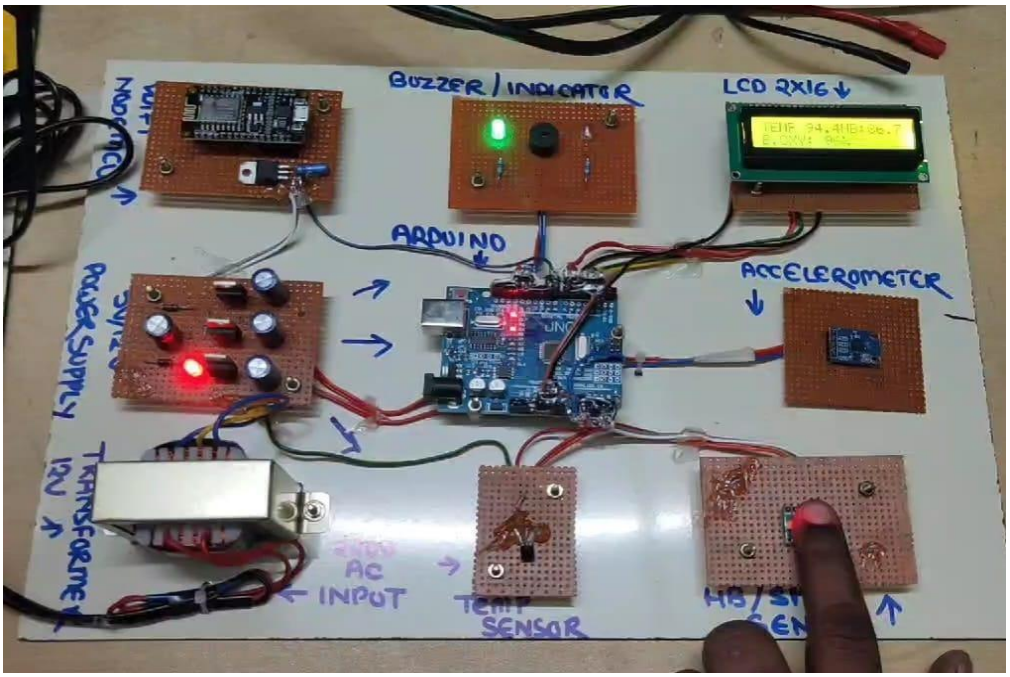


FIG. 5.1 PROTOTYPE MODEL

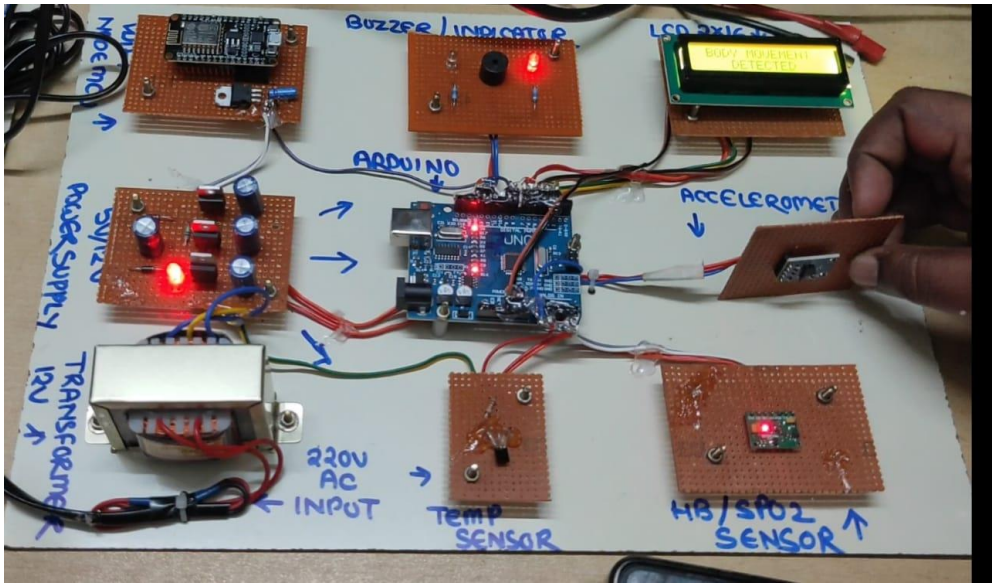


FIG. 5.2 ACCELEROMETER OUTPUT

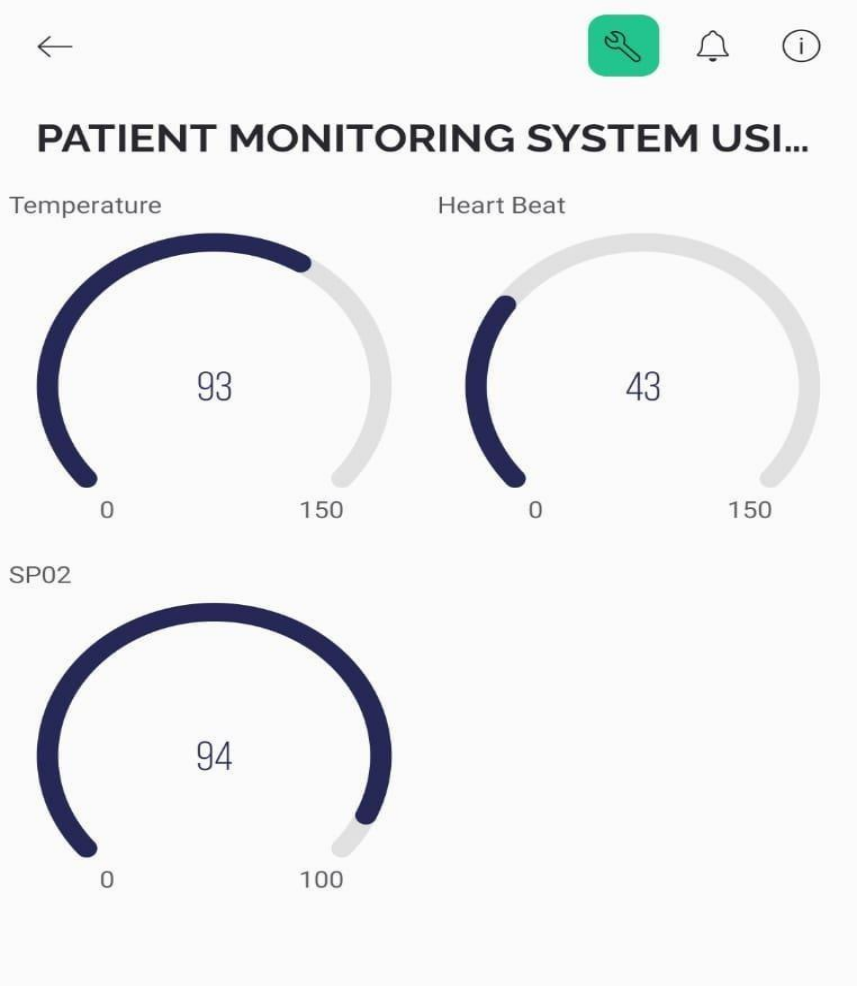


FIG. 5.3 OUTPUT

FIG. 5.4 LCD DISPLAY OUTPUT

CHAPTER 6
CONCLUSION &FUTURE SCOPE

6.1 CONCLUSION

We looked at an internet with a health monitoring device focused on microcontrollers. Certain anomalies in health problems can be detected and reported by the consumer using IoT technologies. The Internet of Things is now generally considered one of the most realistic solutions for remote value control, especially in the field of health monitoring. It enables the safe storage of cloud data on human health parameters, shorter visits to hospitals for periodic exams and, most importantly, disease diagnosis and assessment by any doctor from anywhere. This paper established an Internet of Things health surveillance scheme. Body temperature, pulse rate and body acceleration and temperature were measured by the sensors, with LCD results seen. The IoT technology is then used to give Blynk Applications these sensor values. This knowledge is then sent to a mobile phone connecting to an IoT network which belonging to an approved user. The doctor detects the illness and determines the patient's actual wellbeing on the basis of the evidence.

6.2 FUTURE SCOPE

This system can further be extended by adding a GSM mode to it. This will help to send SMS alert to the Doctors in Emergency Conditions.

REFERENCES

[1] Vikram Patel, Somnath Chatterji, et.al “India: Towards Universal Health CoverageChronic diseases and injuries in India”, Vol. 377, pp 413-428, January 29, 2011.

[2] Giovanni Acampora, Diane J. Cook, Parisa Rashidi and Athanasios V. Vasilakos, “A Survey on Ambient Intelligence in Healthcare”, Proceedings of the IEEE , Vol. 101, No. 12, pp 2470-2494, , December 2013.

[3] Ramakrishnan Iyer and Radharaman Mishra, “Building Intelligent Internet of Things Applications using Microsoft Stream lnsight”, IGATE Global Solution, pp 1-7, April 2014.

[4] Chonggang Wang and Mahmoud Daneshmand, “Guest Editorial Special Issue on Internet of Things (IoT): Architecture, Protocols and Services”, IEEE Sensors Journal, Vol. 13, No. 10, pp 3505-3510, October 2013

Appendix Code

IDE CODE:

```
#include <Wire.h>
#include "MAX30100_PulseOximeter.h"

#include<SoftwareSerial.h>
SoftwareSerial mySerial(8, 9); // NODE MCU

#include<LiquidCrystal.h>
LiquidCrystal lcd(6, 7, 2, 3, 4, 5);

float tempf = 0, temp = 0;
int acc = 0;
int count = 0;
String finalCode;
String a1, b1, c1, d1;
int m = 0;
#define REPORTING_PERIOD_MS     1000

PulseOximeter pox;
uint32_t tsLastReport = 0;

void onBeatDetected()
{
  Serial.println("Beat!");
}

void setup()
{
  Serial.begin(9600);
  Serial.println("Patient Monitoring using IoT");

  mySerial.begin(9600); // NODE MCU - BAUD RATE - 115200
```

```
pinMode(13, OUTPUT);
digitalWrite(13, LOW);

pinMode(12, OUTPUT);
digitalWrite(12, HIGH);

lcd.begin(16, 2);
lcd.clear();
lcd.setCursor(0, 0);
lcd.print(" PATIENT HEALTH ");
lcd.setCursor(0, 1);
lcd.print("MONITORING SYSTM");
delay(1000);

lcd.clear();
lcd.setCursor(0, 0);
lcd.print("     USING      ");
lcd.setCursor(0, 1);
lcd.print(" IOT TECHNOLOGY ");
delay(1000);

lcd.clear();
lcd.setCursor(0, 0);
lcd.print("TEMP: ");
lcd.setCursor(9, 0);
lcd.print("HB: ");
lcd.setCursor(0, 1);
lcd.print("B.OXY: ");
delay(1000);

if (!pox.begin()) {
  Serial.println("FAILED");
  for (;;);
```

```
  } else {
    Serial.println("SUCCESS");
  }
  // epox.setIRLedCurrent(MAX30100_LED_CURR_7_6MA);
  pox.setIRLedCurrent(MAX30100_LED_CURR_11MA);
  // Register a callback for the beat detection
  pox.setOnBeatDetectedCallback(onBeatDetected);
}

void loop()
{
  temp = analogRead(A1);
  acc = analogRead(A0);
  temp = (temp * 500) / 1023;
  tempf = (temp * 1.8) + 32;
  tempf = tempf + 7;
  Serial.println(acc);


  // Make sure to call update as fast as possible
  pox.update();
  if (millis() - tsLastReport > REPORTING_PERIOD_MS) {
    Serial.print("Heart rate:");
    Serial.print(pox.getHeartRate());
    Serial.print("bpm / SpO2:");
    Serial.print(pox.getSpO2());
    Serial.println("%");

    lcd.clear();
    lcd.setCursor(0, 0);
    lcd.print("TEMP ");
    lcd.setCursor(5, 0);
    lcd.print(tempf);
```

```
  lcd.setCursor(9, 0);
  lcd.print("HB: ");
  lcd.setCursor(12, 0);
  lcd.print(pox.getHeartRate());

  lcd.setCursor(0, 1);
  lcd.print("B.OXY: ");
  lcd.setCursor(7, 1);
  lcd.print(pox.getSpO2());
  lcd.print("%");

  senddata();

  if (tempf > 103)
  {
    digitalWrite(13, HIGH);
    digitalWrite(12, LOW);
    lcd.clear();
    lcd.setCursor(0, 0);
    lcd.print("BODY TEMP HIGH.");
    lcd.setCursor(0, 1);
    lcd.print(" HIGH ALERT ...");
    senddata();
    senddata();
    senddata();
    senddata();
    senddata();
    while (1);
  }

  if ((acc < 280) | (acc > 400))
  {
    digitalWrite(13, HIGH);
    digitalWrite(12, LOW);
```

```
      lcd.clear();
      lcd.setCursor(0, 0);
      lcd.print(" BODY MOVEMENT ");
      lcd.setCursor(0, 1);
      lcd.print("    DETECTED   ");
     // m = 1;
     // senddata();
      while (1);
    }


    tsLastReport = millis();
  }
}

void senddata()
{
  // SEND DATA TO NODE MCU
  if(count == 5)
  {
  a1 = String(tempf); // TEMPERATURE
  b1 = String(pox.getHeartRate()); //  HBEAT
  c1 = String(pox.getSpO2()); //  SPO2
  //d1 = String(m); //  M

  finalCode = a1 + "@" + b1 + "*" + c1 + "#";
  mySerial.print(finalCode);
  Serial.println(finalCode);
  count = 0;
  }
  count++
```

NODEMCU

```
#include <ESP8266WiFi.h>
#include <ESP8266HTTPClient.h>

HTTPClient http;
WiFiClient client;// NEW LINE ADDED FOR NEW VERSION OF LIBRARY

int a,b,c,d,e;
int count;
char f; String data;
void setup()
{

WiFi.disconnect();
delay(2000);

WiFi.begin("project","12345678"); // HOTSPOT ID AND PASSWORD
while ((!(WiFi.status() == WL_CONNECTED)))
{
delay(300);
}

Serial.begin(9600);
Serial.println("Wifi- Connected");
}

void loop()
{

 while(Serial.available())
  {
    delay(10);
    f = Serial.read();
    if (f == '@')
```

```
 {
  a = data.toInt();
  data="";
 }
 else if (f == '*')
 {
  b = data.toInt();
  data="";
 }
 else if (f == '#')
 {
  c = data.toInt();
  data="";
 }
 else if (f == '-')
 {
  d = data.toInt();
  break;
 }

 else
 {
 data +=f;
 }
 }

if (data.length() > 0)
{
 Serial.println(a); // TEMP
 Serial.println(b); // conduct
 Serial.println(c); // TURBIDITY
 Serial.println(d); // PH
 Serial.println("");
 upload();
```

```
   data="";
  }
}

void upload()
{
   String link="http://blynk.cloud/external/api/update?token=pTmpaYtJEeyG8ku8HNaVf-
Ym0QmuhQ4M&v0="; //
   link=String(link+a);
   http.begin(client,link); // MODIFY SYNTAX FOR NEW VERSION OF LIBRARY ESP AND
BLYNK
   http.GET();
   http.end();
   delay(300);

   String link1="http://blynk.cloud/external/api/update?token=pTmpaYtJEeyG8ku8HNaVf-
Ym0QmuhQ4M&v1="; //
   link1=String(link1+b);
   http.begin(client,link1); // MODIFY SYNTAX FOR NEW VERSION OF LIBRARY ESP AND
BLYNK
   http.GET();
   http.end();
   delay(300);

   String link2="http://blynk.cloud/external/api/update?token=pTmpaYtJEeyG8ku8HNaVf-
Ym0QmuhQ4M&v2="; //
   link2=String(link2+c);
   http.begin(client,link2); // MODIFY SYNTAX FOR NEW VERSION OF LIBRARY ESP AND
BLYNK
   http.GET();
   http.end();
   delay(300);
```

```
   String link3="http://blynk.cloud/external/api/update?token=pTmpaYtJEeyG8ku8HNaVf-
Ym0QmuhQ4M&v3="; //
   link3=String(link3+d);
   http.begin(client,link3); // MODIFY SYNTAX FOR NEW VERSION OF LIBRARY ESP AND
BLYNK
   http.GET();
   http.end();
   delay(300)
```