**A Mini Project Report On**

# ARDUINO-IOT POWERED WEARABLE VITAL SIGN MONITORING WITH PREDICTIVE HEALTH ALERTS

# ABSTRACT

This project presents an ARDUINO-powered wearable device designed for continuous vital sign monitoring and predictive health analysis. The system integrates a DHT11 sensor to measure body temperature and humidity, and an SpO2 sensor to track blood oxygen levels and heart rate. Data collected is wirelessly transmitted via Wi-Fi to a cloud platform for real-time monitoring and analysis. Using predictive algorithms, the system identifies abnormal trends, enabling early detection of potential health issues. The compact and wearable design ensures ease of use and mobility. Power-efficient operation extends usability for daily wear. The device is ideal for remote patient monitoring and personal health tracking. This solution bridges low- cost hardware with smart healthcare innovations.



**Vision of the Institute**

To emerge as a premier institute of high-quality professional graduates who can contribute to economic andsocial developments of the Nation.

##### Mission of the Institute

Mission Statement

**IM1** To have holistic approach in curriculum and pedagogy through industry interface to meet the needs of global competency.

**IM2** To develop student with knowledge, attitude, employability skills, entrepreneurship, research potential and professionally ethical citizens.

**IM3** To contribute to advancement of Engineering and Technology that would help to satisfy societal needs.

**IM4** To preserve, promote cultural heritage, humanistic values and spiritual valuesthus helping in peace and harmony in the society.



**Vision of the Department**

To impart quality technical education in Electronics & Communication with accent on creativity, innovation and research thereby producing competent engineers who can meet global challenges withsocietal commitment.

**Mission of the Department**

Mission Statement

**DM1** To impart quality education to students in Basic Sciences, Mathematics, Electronics and communication Engineering through innovative teaching-learning process.

**DM2** To facilitate students to define, design, and solve engineering problems in the field of Electronics and Communications Engineering using various Electronic Design Automation (EDA) tools.

**DM3** To encourage research culture among faculty and students thereby facilitating them to be creative and innovative through constant interaction with R & D organizations and Industry.

**DM4** To include qualities, professional ethics and social responsibilities instudents and faculty.



##### Program Educational Objectives

**PEO1:** Graduates with fundamental and advanced knowledge in Engineering Sciences, Mathematics and Engineering course of Electronics and Communication Engineering along with allied domains of Engineering will be able to exhibit the contemporary skills and become globally competent with a flair for life-long learning.

**PEO2:** Graduates capable in design, develop creative and innovative technologies in the field of Electronics and Communication Engineering, enabling them to work in multi-disciplinary teams to meet the societal needs.

**PEO3:** Graduates with professional ethics and values, positive attitude, effective communication skills and knowledge in cutting edge technologies will ensure sustainable engineering and able to demonstrate top notch technology in their professional practices.



##### Program Outcomes (POs)

Engineering Graduates will be able to:

**PO1. Engineering knowledge**: Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems

**PO2. Problem analysis**: Identify, formulate, review research literature, and analyse complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.

**PO3. Design/development of solutions**: Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.

**PO4. Conduct investigations of complex problems**: Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.

**PO5. Modern tool usage**: Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modelling to complex engineering activities with an understanding of the limitations.

**PO6. The engineer and society**: Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.

**PO7. Environment and sustainability**: Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.

**PO8. Ethics**: Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.

**PO9. Individual and team work**: Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.

**PO10. Communication**: Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions

**PO11. Project management and finance**: Demonstrate knowledge and understanding of the engineering and management principles and apply these to one’s own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.

**PO12. Life-long learning**: Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.



##### Program Specific Outcomes (PSOs)

**PSO1:** Apply the knowledge of mathematics, basic sciences and Engineering Sciences with a multidisciplinary approach to analyse and design electronic circuits, communication systems using contemporary skills.

**PSO2:** Conduct experiments with the state-of-the-art technology and carryout investigation in research as a team or individual and interpretant the data to synthesize various electronic and communication systems in the perspective of societal concerns, environment and economy with sustainable engineering practices.

**PSO3:** Demonstrate professional ethics and implement the principles of project management, business and public policy in the design of a system and carrying out simulation studies, and eventually communicate with effective presentation for its implementation with a due note for the scope of further studies to continue life-long learning.

# CONTENTS

**Certificate** i

[Acknowledgement ii](#_TOC_250040)

[Abstract iii](#_TOC_250039)

CHAPTER 1: INTRODUCTION 1

CHAPTER 2: LITERATURE SURVEY 4

CHAPTER 3: EXISTING SYSTEMS 7

CHAPTER 4: PROPOSED SYSTEMS 8

* 1. [Block diagram Explanation 9](#_TOC_250038)

CHAPTER 5: HARDWARE DESCRIPTION 10

* 1. [Microcontroller 10](#_TOC_250037)
     1. [Introduction to Microcontrollers 10](#_TOC_250036)
     2. [AVR- Arduino Microcontroller 11](#_TOC_250035)
     3. [Crystal oscillator 11](#_TOC_250034)
     4. [Architecture 12](#_TOC_250033)
     5. [Pin diagram 14](#_TOC_250032)
     6. [Applications 17](#_TOC_250031)
  2. [Regulated Power Supply 17](#_TOC_250030)
     1. [Introduction 17](#_TOC_250029)
     2. [Block Diagram 18](#_TOC_250028)
     3. [Transformers 19](#_TOC_250027)
     4. [Battery Power Supply 20](#_TOC_250026)
     5. [Rectifiers 21](#_TOC_250025)
     6. [Capacitors 24](#_TOC_250024)
     7. [Voltage Regulator 26](#_TOC_250023)
     8. [Resistors 26](#_TOC_250022)
  3. [LED 28](#_TOC_250021)
     1. [Working 28](#_TOC_250020)
  4. [LCD Display 30](#_TOC_250019)
     1. [LCD background 30](#_TOC_250018)
     2. [Pin Description 30](#_TOC_250017)
     3. [Circuit Description 32](#_TOC_250016)
     4. [Applications 32](#_TOC_250015)
  5. [Buzzer 33](#_TOC_250014)
  6. [Heart Beat Sensor 34](#_TOC_250013)
     1. [Features 35](#_TOC_250012)
     2. [Working 35](#_TOC_250011)
  7. DHT 11 37
  8. [IOT esp8266 module 40](#_TOC_250010)
     1. [Introduction 40](#_TOC_250009)
     2. [Technology Overview 40](#_TOC_250008)
     3. [Specifications 41](#_TOC_250007)
     4. [Applications 44](#_TOC_250006)

CHAPTER 6: SOFTWARE SIMULATION 45

* 1. [Arduino IDE compiler 45](#_TOC_250005)

CHAPTER 7: CODE 50

CHAPTER 8: RESULTS 56

CHAPTER 9: ADVANTAGES&APPLICATIONS 58

* 1. [Advantages 58](#_TOC_250004)
  2. [Applications 58](#_TOC_250003)

CHAPTER 10: CONCLUSION&FUTURE SCOPE 59

* 1. [Conclusion 59](#_TOC_250002)
  2. [Future Scope 59](#_TOC_250001)

PROJECT DETAILS 60

[REFERENCES 61](#_TOC_250000)

# LIST OF FIGURES

|  |  |  |
| --- | --- | --- |
| **F.NO** | **TITLE** | **PAGE NO** |
| 4 | Block diagram | 8 |
| 5.1 | Microcontroller | 10 |
| 5.1.1 | Architecture |  |
| 5.1.2 | Arduino development board | 11 |
| 5.1.3 | Crystal oscillator | 11 |
| 5.1.4 | AVR Architecture | 13 |
| 5.1.5 | (a)Pin diagram of Atmega328 | 14 |
| 5.1.5 | (b)Pin diagram of Arduino uno board | 15 |
| 5.2.2 | (a)Regulated power supply | 18 |
| 5.2.2 | (b)Circuit diagram of regulated power supply with LED connection | 18 |
| 5.2.3 | Step down transformer | 19 |
| 5.2.4 | (a)Hi-watt 9V battery | 20 |
| 5.2.4 | (b)Pencil battery of 1.5V | 21 |
| 5.2.5 | (a)Bridge rectifier: a full wave rectifier using 4 diodes | 22 |
| 5.2.5 | (b)DB107 | 23 |
| 5.2.6 | (a)Construction of a capacitor | 24 |
| 5.2.6 | (b)Electrolytic capacitor | 24 |
| 5.2.7 | Voltage regulator | 26 |
| 5.2.8 | (a)Resistor | 27 |
| 5.2.8 | (b)Colour branch in resistor | 27 |
| 5.3 | (a)Inside a LED | 28 |
| 5.3 | (b)Parts of an LED | 28 |
| 5.3.1 | Electrical symbol& Polarities of LED | 28 |
| 5.4.1 | LCD pin diagram | 30 |
| 5.5 | (a)Buzzer | 33 |
| 5.5 | (b)Buzzer logic circuit | 34 |
| 5.6.2 | (a)Sensor construction | 35 |
| 5.6.2 | (b)Signal view | 36 |

|  |  |  |
| --- | --- | --- |
| 5.6.2 | (c)Amplifier circuit | 36 |
| 5.7 | DHT 11 sensor | 37 |
| 5.8 | IOT module | 40 |
| 5.8.4 | IOT module pin diagram | 44 |
| 6.1 | (a)The Arduino software IDE | 46 |
| 6.1 | (b)Selecting a board | 47 |
| 6.1 | (c)Selecting the port | 47 |
| 6.1 | (d)Opening an example | 48 |
| 6.1 | (e)The Arduino software IDE | 49 |
| 6.1 | (f)Selection of board and port | 49 |
| 8 | (a)Implementation of Arduino-IOT powered wearable vital sign monitoring with Predictive Health Monitoring | 56 |
| 8 | (b)LCD display when system in ON state | 57 |
| 8 | (c)Output of the system on LCD | 57 |
| 8 | (d)Temp values monitored on Thing Speak | 57 |
| 8 | (e)Humidity values monitored on Thing Speak | 57 |

**LIST OF TABLES**

|  |  |  |
| --- | --- | --- |
| **S.NO** | **TITLE** | **PAGE NO** |
| 5.4.2 | Pin description | 30 |
| 5.8.4 | Current consumption | 41 |

### CHAPTER

1

#### INTRODUCTION

*This chapter discusses about the objectives, research gap, background and identification of problem*

##### 1. Introduction:

In recent years, the healthcare industry has undergone a transformative shift with the emergence of wearable technology and IoT-enabled health monitoring systems. These advancements have paved the way for real-time, remote, and non-invasive patient monitoring solutions that significantly improve healthcare accessibility, efficiency, and outcomes. Among the most critical health parameters, vital signs such as body temperature, heart rate, and blood oxygen saturation (SpO2) are key indicators of an individual’s physical condition. Continuous monitoring of these vital signs provides early warning of medical conditions, allowing for prompt intervention and reducing the chances of complications. Traditional healthcare setups are often limited by physical infrastructure, periodic checkups, and manual data recording, leading to delayed diagnosis and inefficient treatment planning. This limitation is especially prevalent in rural and remote areas where access to qualified healthcare professionals and diagnostic tools is scarce. The integration of wireless communication, embedded systems, and biosensors into compact wearable devices has revolutionized how health data is collected, analysed, and acted upon.

This project proposes a wearable health monitoring system that employs a ARDUINO microcontroller as the core processor, coupled with a DHT11 sensor for capturing environmental temperature and humidity, and an SpO2 sensor to measure pulse rate and blood oxygen saturation levels. The system continuously collects physiological data and transmits it to a cloud-based platform using Wi-Fi module linked with ARDUINO. This allows caregivers, doctors, and even patients themselves to access real-time data remotely via a smartphone or

computer. The inclusion of predictive health monitoring algorithms enables the system to analyse historical and current data to forecast health trends, detect anomalies, and issue early alerts for conditions such as hypoxia, fever, dehydration, or possible respiratory distress. This form of predictive analytics not only improves patient outcomes through early detection but also reduces hospital readmissions, emergency visits, and long-term healthcare costs. The wearable unit is designed to be lightweight, portable, and energy-efficient, making it ideal for day-to-day use. Patients, especially those with chronic illnesses, the elderly, or individuals recovering from surgery, can benefit immensely from continuous health tracking without the need for hospital visits. The ARDUINO's open-source nature and low cost make it an excellent choice for scalable and customizable health solutions. The DHT11 sensor, though basic, provides adequate ambient sensing for evaluating environmental conditions that can impact health, such as excessive humidity or temperature. On the other hand, the SpO2 sensor plays a crucial role in monitoring cardiopulmonary functions, which is particularly vital during pandemics like COVID-19, where oxygen saturation is a primary indicator of infection severity. Together, these sensors offer a basic but reliable overview of a patient’s health status.

From a technical perspective, the ARDUINO offers excellent connectivity features, robust support for multiple programming environments (like Arduino IDE and Lua), and seamless integration with cloud platforms such as Thing Speak, Firebase, and Blynk. These features allow for scalable data visualization, storage, and processing. The data collected can be visualized on dashboards in the form of time-series graphs, analysed for patterns using machine learning models, or used to trigger real-time alerts via SMS, email, or push notifications. This integration of hardware and software lays the foundation for smart healthcare systems where automation, machine intelligence, and medical data converge to support clinicians and empower patients.

Additionally, the predictive health aspect of the system is implemented using threshold-based rules and machine learning techniques. By analysing patterns in data such as fluctuations in SpO2 levels or irregular temperature spikes, the system can predict potential health deteriorations before critical thresholds are crossed. This approach helps in proactive rather than reactive healthcare, enabling timely interventions. Moreover, the system's modular architecture allows for future expansion, such as integrating ECG, motion sensors, or GPS for fall detection and location tracking.

The wearable device can thus evolve into a complete personal health assistant that not only monitors but also advises based on real-time data and learned behaviour.

Security and privacy are also critical aspects of any health monitoring system. To protect user data, the system can incorporate data encryption methods during transmission and secure user authentication on cloud platforms. This ensures compliance with healthcare data regulations and builds trust among users. Furthermore, the system can include features like local storage for offline monitoring and data synchronization when the network is available, ensuring uninterrupted functionality in remote areas with poor connectivity.

In conclusion, the proposed ARDUINO-powered wearable vital sign monitoring device presents a reliable, cost-effective, and scalable solution for continuous and predictive health tracking. It bridges the gap between traditional healthcare and digital wellness by combining sensor technology, IoT, and predictive analytics. The implementation of such wearable systems holds great potential for transforming patient care, improving quality of life, and enabling smarter, data-driven healthcare delivery across urban and rural landscapes alike. As the technology matures and sensor accuracy improves, such systems are expected to play a central role in the evolution of next-generation e-health and telemedicine platforms, contributing significantly to global health initiatives and personalized medical care.

### CHAPTER

2

#### LITERATURE SURVEY

*This chapter discusses about a concise review of existing research, technologies, and methods relevant to the project topic.*

A study by **Patel et al. (2020)** introduced a smart wearable health device using ARDUINO and pulse sensors to detect heart rate and temperature, transmitting the data to a cloud server. The system demonstrated cost efficiency and real-time tracking capabilities. It highlighted the potential for remote health surveillance in rural areas.[1].

An IoT-based temperature and heartbeat monitoring system using Arduino and GSM. The system successfully sends alerts via SMS to caregivers during abnormal readings. This prototype laid the foundation for mobile health tracking systems.[2]

A wearable device using optical sensors for continuous SpO2 monitoring in post-operative patients. It emphasized low-power design and wireless data transmission.[3]

**Ramesh et al. (2021)** implemented ARDUINO-based patient monitoring system with DHT11 and MAX30100 sensors. The device collected vital parameters and uploaded data to Thing Speak for visualization. It proved effective in reducing hospital visits for elderly patients.[4]

**Kumar and Singh (2020)** presented an IoT-based real-time health tracking system for chronic patients. They used cloud analytics to predict health abnormalities based on prior datasets. Their predictive module improved early diagnosis and patient engagement.[5]

**Ahmed et al. (2017)** designed a low-cost health monitoring system using ESP8266 and various biosensors. They integrated a mobile app for live data access. The system ensured accessibility for underserved populations.[6]

Using an SpO2 and temperature sensors with Firebase integration the real-time alert system is developed. The wearable device monitored symptoms of respiratory infections. It demonstrated

scalability and affordability.[7]

**Joshi et al. (2019)** demonstrated a wearable device using ARDUINO and machine learning models for anomaly detection. They used historical data to predict potential health issues. Their results showed 88% accuracy in early alerts.[8]

A smart health patch using flexible sensors and Wi-Fi modules. The patch provides accurate SpO2, pulse rate, and temperature monitoring. It contributed to wearable material innovation and miniaturization.[9]

**Choudhury et al. (2020)** developed a predictive health IoT model combining heart rate, oxygen saturation, and user behaviour data. They used cloud-based analysis for risk profiling. The model showed potential in pre-diagnostic stages.[10]

**Goyal et al. (2018)** introduced a wearable pulse oximeter with an OLED display and Bluetooth communication. It proved effective in fitness monitoring and sleep aponia detection. Their device was integrated with Android for user feedback.[11]

**Bhatnagar and Mishra (2021)** implemented real-time vitals monitoring using DHT11 and MAX30102 sensors. Their system used ARDUINO to push alerts to an Android app. The project focused on remote diagnostics in home-care setups.[12]

**Kim et al. (2019)** explored continuous wearable SpO2 monitoring in high-altitude environments using energy-efficient algorithms. Their findings contributed to personalized medicine for mountain climbers and athletes. The device sustained long battery life.[13]

**Anwar et al. (2021)** studied a cloud-integrated system using ARDUINO and biomedical sensors for rural health centres. The focus was on reducing response time in emergencies. They emphasized low-bandwidth data transfer optimization.[14]

**Singh and Desai (2017)** developed a health monitoring glove with integrated sensors to detect pulse rate and oxygen levels. The glove was embedded with wireless modules. It targeted industrial workers exposed to hazardous environments.[15]

**Paul et al. (2022)** presented an AI-powered wearable device that detected fever and low SpO2 using cloud prediction. The system warned users before symptoms worsened. It helped in proactive COVID-19 surveillance.[16]

**Rao and Patil (2020)** designed ARDUINO-based wearable health band interfacing with cloud dashboards. Their band monitored elderly patients for signs of fever or breathlessness. They used Firebase for real-time visualization.[17]

**Narayan and Kulkarni (2019)** proposed an IoT-enabled pulse and temperature monitoring kit using ESP8266 and DHT11. Their system was integrated into a school health surveillance network. The kit enabled prompt attention to student health issues.[18]

**Gupta et al. (2021)** explored predictive analytics using heart rate variability data. They combined wearable sensor data with regression models. Their system detected fatigue and dehydration in soldiers.[19]

**Verma and Jha (2022)** developed a hybrid model using ML and wearable sensors to detect COVID-19-like symptoms. Their system included temperature, SpO2, and coughing patterns. The model was tested with real-time hospital datasets.[20]

**Lee et al. (2020)** built a smart shirt embedded with bio-sensors and ARDUINO for health monitoring during exercise. The system measured sweat levels, heart rate, and oxygen levels. Data was sent via Wi-Fi to a mobile dashboard.[21]

**Srinivas and Reddy (2018)** implemented a low-cost wearable system using DHT11 for elderly patient care. They focused on fall detection and health monitoring. The system was connected to caregivers through mobile alerts.[22]

**Kaur and Mehta (2021)** proposed a real-time oxygen and heart rate monitor using MAX30100 with ARDUINO. They created an alert mechanism using IFTTT. The solution aimed to assist individuals in remote quarantine zones.[23]

**Rahman et al. (2019)** introduced a wearable SpO2 tracking system that used solar-powered sensors. Their study emphasized sustainable design in medical wearables. It proved beneficial in outdoor monitoring.[24] **Deshmukh and Kulkarni (2021)** used IoT and AI to create a system for early health risk detection. They employed time-series forecasting on sensor data. Their results showed improved alert precision compared to threshold-only models.[25]

### CHAPTER

3

#### EXISTING SYSTEMS

*This chapter describes previously developed solutions, technologies, or models that address similar problems or objectives.*

The existing health monitoring systems primarily rely on periodic manual checkups or hospital- based monitoring devices that require physical presence and are often bulky or expensive. Many traditional systems lack real-time data transmission and are limited to storing health records locally or on a computer. Basic wearable health gadgets, such as fitness bands, monitor only selective parameters like heart rate or steps and are not equipped for clinical-grade vital sign tracking. Additionally, most available solutions do not integrate predictive analytics, thereby failing to provide early alerts or warnings. The absence of connectivity with cloud platforms restricts their usefulness for remote monitoring. These systems often depend on Bluetooth rather than Wi-Fi, which limits long-range transmission. Data visualization, if available, is typically done via proprietary apps without customization. Moreover, integration of multiple sensors is minimal, and energy optimization is rarely considered. Hence, the existing solutions fall short of providing a comprehensive, affordable, and intelligent health monitoring ecosystem.

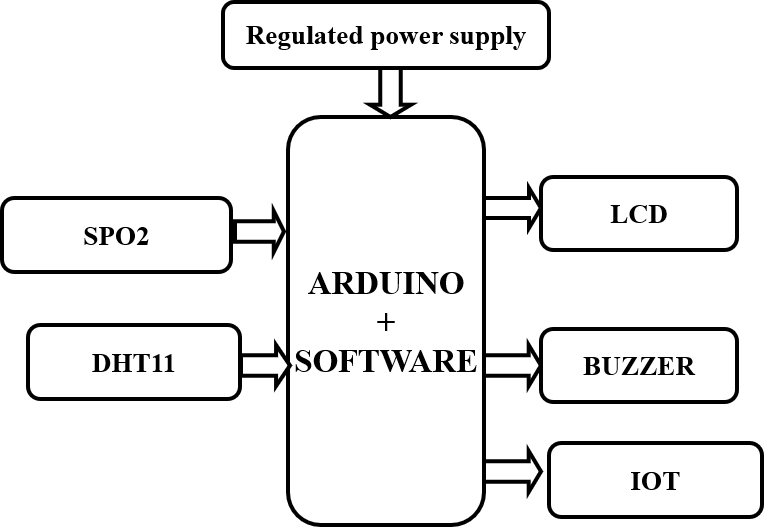
### CHAPTER

4

#### PROPOSED SYSTEMS

*This chapter outline the new or improved solution being suggested to overcome the limitations of existing systems*.

The proposed system is an Arduino-based wearable device that monitors SpO2, heart rate, temperature, and humidity using integrated sensors. It uses a Wi-Fi module to transmit real- time data to cloud platforms like Thing Speak or Firebase for remote access. Predictive analytics detect abnormal health trends and send alerts to caregivers or doctors. The device is compact, low-power, and comfortable for daily use, with long battery life. A modular design allows adding sensors like ECG, GPS, or motion detectors. Custom dashboards provide easy- to-read health data visualization. Secure data transmission ensures privacy and authorized access. This system enables continuous health monitoring and reduces the burden on hospitals.



**Fig:4 Block Diagram**

##### Block Diagram Explanation:

1. Regulated Power Supply:

This block provides a stable voltage source to all components in the system. It ensures that the microcontroller and sensors receive clean and consistent power, typically 5V or 3.3V, essential for accurate sensor readings and stable communication.

1. Arduino + Software (ARDUINO in practical implementation):

This is the central processing unit of the system. It collects data from the SpO2 and DHT11 sensors, processes it through embedded code, and then performs decision-making tasks like displaying data, sounding alarms, or sending data over the internet.

1. SpO2 Sensor:

This sensor measures the blood oxygen saturation (SpO2) and heart rate of the user. It sends real-time biometric data to the microcontroller, which is crucial for monitoring respiratory and cardiovascular health.

1. DHT11 Sensor:

This sensor measures ambient temperature and humidity levels. It's useful for tracking body temperature trends and environmental conditions affecting patient health. Data is sent to the Arduino for further processing.

1. LCD Display:

The LCD provides a local display for vital readings such as temperature, heart rate, and SpO2 levels. It allows the user or caretaker to see the real-time status without needing an external device.

1. Buzzer:

The buzzer acts as an alert system. If any vital sign goes beyond the predefined safety range (like low SpO2 or high temperature), the buzzer is triggered to provide an audible alarm for immediate attention.

1. IoT Module (Wi-Fi using ARDUINO or ESP):

This component sends the processed health data to an online cloud platform such as Thing Speak or Firebase. It allows for real-time remote monitoring through smartphones or computers and can be used for data logging and predictive analysis.

CHAPTER

5

# HARDWARE DESCRIPTION

*This chapter provides details of all physical components used, including their specifications, functions, and interconnections****.***

## Micro controller:

****

**Fig: 5.1 Microcontrollers**

## Introduction to Microcontrollers:

Circumstances that we find ourselves in today in the field of microcontrollers had their beginnings in the development of technology of integrated circuits. This development has made it possible to store hundreds of thousands of transistors into one chip.

That was a prerequisite for production of microprocessors, and the first computers were made by adding external peripherals such as memory, input-output lines, timers and other. Further increasing of the volume of the package resulted in creation of integrated circuits. These integrated circuits contained both processor and peripherals. That is how the first chip containing a microcomputer, or what would later be known as a microcontroller came about. Microprocessors and microcontrollers are widely used in embedded systems products.

Microcontroller is a programmable device. A microcontroller has a CPU in addition to a fixed amount of RAM, ROM, I/O ports and a timer embedded all on a single chip. The fixed amount of on-chip ROM, RAM and number of I/O ports in microcontrollers makes them ideal for many applications in which cost and space are critical.

##### AVR-ARDUINO microcontroller:

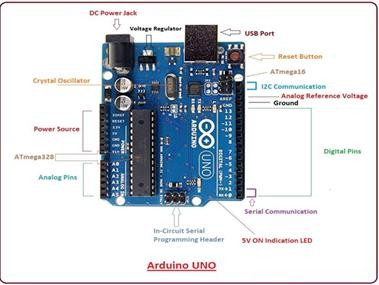
****

Fig 5.1.2: ARDUINO Development Board

The AVR is a modified Harvard architecture 8-bit RISC single chip microcontroller which was developed by Atmel in 1996. The AVR was one of the first microcontroller families to use on-chip flash memory for program storage, as opposed to One-Time Programmable ROM, EPROM, or EEPROM used by other microcontrollers at the time.

##### Crystal Oscillator:

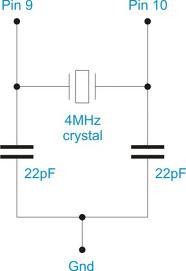
****

Fig 5.1.3: Crystal oscillator

XTAL1 and XTAL2 are the input and output of an inverting amplifier used for an on-chip oscillator. A quartz crystal or ceramic resonator can be connected to generate the clock signal. The CKOPT fuse determines the oscillator mode: programmed for full swing, unprogrammed for low swing. When CKOPT is programmed, the output has rail-to-rail swing, suitable for noisy environments or driving other buffers. Unprogrammed CKOPT reduces the output swing, which lowers power consumption but limits frequency and drive capability. The maximum frequency for resonators is 8 MHz (CKOPT off) and 16 MHz (CKOPT on). C1 and C2 capacitors should be equal in value for both crystals and resonators. Use manufacturer-recommended capacitor values for ceramic resonators. The oscillator supports three different modes, each suited for a specific frequency range. Mode selection is done using fuse bits CKSEL3..1 in the microcontroller.

##### Architecture:

**Memory:** It has **8 Kb** of Flash program memory (10,000 Write/Erase cycles durability), 512 Bytes of EEPROM (100,000 Write/Erase Cycles), 1Kbyte Internal SRAM.

**I/O Ports:** 23 I/ line can be obtained from three ports; namely Port B, Port C and Port D.

**Interrupts:** Two External Interrupt source, located at port D. 19 different interrupt vectors supporting 19 events generated by internal peripherals.

**Timer/Counter:** Three Internal Timers are available, two 8 bit, one 16 bit, offering various operating modes and supporting internal or external clocking.

**SPI (Serial Peripheral interface):** ATmega8 holds three communication devices integrated. One of them is Serial Peripheral Interface. Four pins are assigned to Atmega8 to implement this scheme of communication.

**USART:** One of the most powerful communication solutions is [USART](http://www.circuitstoday.com/how-to-establish-a-pc-micro-controller-usart-communication) and ATmega8 supports both synchronous and asynchronous data transfer schemes. It has three pins assigned for that. In many projects, this module is extensively used for PC-Micro controller communication.

**TWI (Two Wire Interface):** Another communication device that is present in ATmega8 is Two Wire Interface. It allows designers to set up a commutation between two devices using just two wires along with a common ground connection, As the TWI output is made by means of open

collector outputs, thus external pull up resistors are required to make the circuit.

**Analog Comparator:** A comparator module is integrated in the IC that provides comparison facility between two voltages connected to the two inputs of the Analog comparator via External pins attached to the micro controller.

**Analog to Digital Converter:** Inbuilt analog to digital converter can convert an analog input signal into digital data of **10bit** resolution. For most of the low-end application, this much resolution is enough.

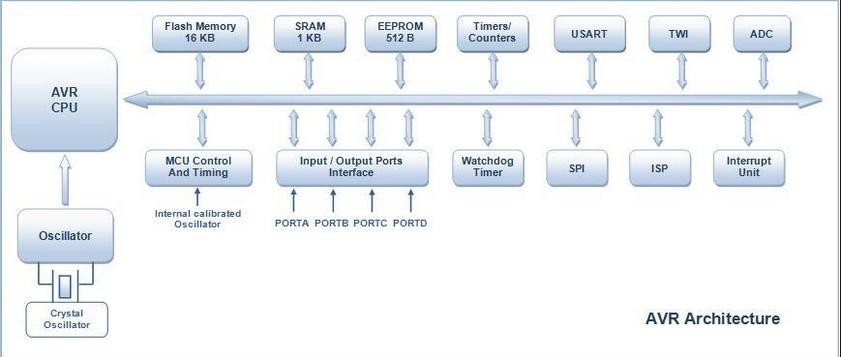


fig 5.1.4: AVR Architecture

**Microcontroller:** Microcontroller can be termed as a single on chip computer which includes number of peripherals like RAM, EEPROM, Timers etc., required to perform some predefined tasks.

The computer on one hand is designed to perform all the general purpose tasks on a single machine like you can use a computer to run a software to perform calculations or you can use a computer to store some multimedia file or to access [internet](http://www.engineersgarage.com/articles/what-is-internet-history-working) through the browser, whereas the microcontrollers are meant to perform only the specific tasks, for e.g., switching the AC off automatically when room temperature drops to a certain defined limit and again turning it ON when temperature rises above the defined limit.

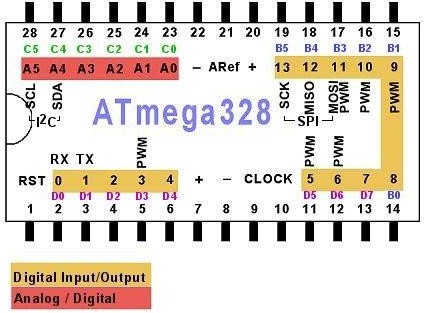
There are number of popular families of microcontrollers which are used in different applications as per their capability and feasibility to perform the desired task, most common of

these are [8051,](http://www.engineersgarage.com/8051-microcontroller) AVR and [PIC](http://www.engineersgarage.com/articles/pic-microcontroller-tutorial) microcontrollers. In this article we will introduce you with **AVR** family of microcontrollers.

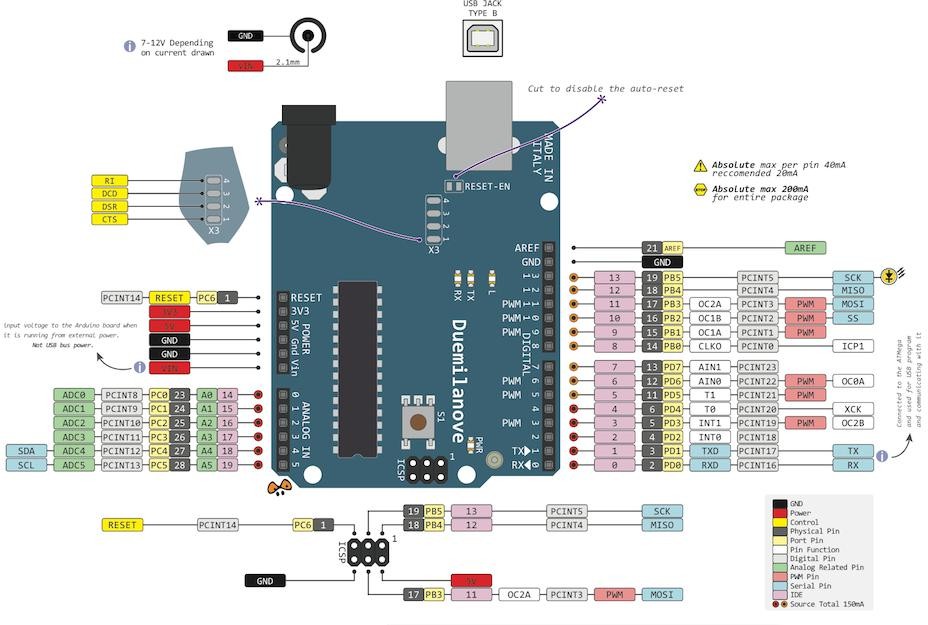
AVR was developed in 1996 by Atmel Corporation, designed by Alf-Egil Bogen and Vegard Wollan. The name AVR stands for Alf-Egil Bogen Vegard Wollan RISC or Advanced Virtual RISC.The first AVR-based microcontroller was AT90S8515; AT90S1200 was the first commercially released in 1997.AVR microcontrollers come in three families: TinyAVR, MegaAVR, and XmegaAVR.TinyAVR is compact with limited memory for basic applications. MegaAVR has up to 256 KB flash, many peripherals, and is suited for moderate to complex tasks. XmegaAVR is for advanced, high-speed commercial applications requiring large memory. Flash, EEPROM, and SRAM are integrated on-chip, minimizing the need for external memory. Program memory is non-volatile flash; each instruction is one or two 16-bit words. All AVR code executes from on-chip flash; off-chip code execution is not supported (except FPSLIC). AVR has 32 general-purpose 8-bit registers mapped into low memory addresses. XMEGA’s I/O registers occupy the first 4KB, while SRAM starts at 0x2000.EEPROM is non- volatile, slower, and usually not memory-mapped; writing should be minimized. The architecture supports 130 instructions, most executing in a single clock cycle.AVR features include 8KB flash, 1KB SRAM, 512B EEPROM, timers, ADCs, PWM, USART, SPI, TWI,

and comparator.

##### Pin diagram:

****

**Fig.5.1.5 (a): Pin diagram of Atmega328**



**fig.5.1.5(b): Pin Diagram of Arduino UNO board**

###### VCC

Digital supply voltage magnitude of the voltage range between 4.5 to 5.5 V for the ATmega8 and 2.7 to 5.5 V for ATmega8L

###### GND

Ground Zero reference digital voltage supply.

###### PORTB (PB7. PB0)

PORTB is an 8-bit bidirectional I/O port with selectable internal pull-up resistors and symmetrical output drive capability. During RESET, PORTB pins enter a tri-state condition, even if the clock is inactive.

###### PORTC (PC5. PC0)

PORTC is a 7-bit bidirectional I/O port with optional internal pull-up resistors and symmetrical output drive. Its pins enter a tri-state condition during RESET, even if the clock is not running.

###### PC6/RESET

If the RSTDISBL fuse is programmed, PC6 functions as a regular I/O pin with altered characteristics. If not programmed, PC6 acts as a RESET input, triggering a reset on a LOW signal lasting at least 1.5 microseconds, even without a running clock.

###### PORTD (PD7. PD0)

PORTD is an 8-bit bidirectional I/O port with selectable internal pull-up resistors and symmetrical output drive capability. Its pins enter a tri-state condition during RESET, even if the clock is inactive.

###### RESET

Reset input pin. LOW signal on this pin with a minimum width of 1.5 microseconds will bring the microcontroller into reset condition, although the clock is not running. Signal with a width of less than 1.5 microseconds does not guarantee a Reset condition.

###### AVCC

AVCC is the supply voltage pin for the ADC, PC3. PC0, and ADC7.ADC6. This pin should be connected to VCC, even if the ADC is not used. If the ADC is used, AVCC should be connected to VCC through a low-pass filter to reduce noise.

Aref

Analog Reference pin for the ADC.

###### ADC7, ADC6

ADC analog input there is only on ATmega8 with TQFP and QFP packages / MLF.

###### PORTS

Term "port" refers to a group of pins on a microcontroller which can be accessed simultaneously, or on which we can set the desired combination of zeros and ones, or read from them an existing status. Physically, port is a register inside a microcontroller which is connected by wires to the pins of a microcontroller. Ports represent physical connection of Central Processing Unit with an outside world. Microcontroller uses them

The Atmega8 has 23 I/O ports which are organized into 3 groups:

* Port B (PB0 to PB7)
* Port C (PC0 to PC6)
* Port D (PD0 to PD7)

We will use mainly 3 registers known as **DDRX, PORTX** & **PINX**.

We have total four PORTs on my ATmega16. They are **PORTA, PORTB, PORTC** and **PORTD**. They are multifunctional pins. Each of the pins in each port (total 32) can be treated as input or output pin.

##### Applications

AVR microcontroller perfectly fits many uses, from automotive industries and controlling home appliances to industrial instruments, remote sensors, electrical door locks and safety devices. It is also ideal for smart cards as well as for battery supplied devices because of its low consumption.

EEPROM memory makes it easier to apply microcontrollers to devices where permanent storage of various parameters is needed (codes for transmitters, motor speed, receiver frequencies, etc.). Low cost, low consumption, easy handling and flexibility make ATmega8 applicable even in areas where microcontrollers had not previously been considered (example: timer functions, interface replacement in larger systems, coprocessor applications, etc.).

In System Programmability of this chip (along with using only two pins in data transfer) makes possible the flexibility of a product, after assembling and testing have been completed.

#### REGULATED POWER SUPPLY:

##### Introduction:

Power supply is a supply of electrical power. A device or system that supplies electrical or other types of energy to an output load or group of loads is called a power supply unit or PSU. The term is most commonly applied to electrical energy supplies, less often to mechanical ones, and rarely to others.

A power supply may include a power distribution system as well as primary or secondary sources of energy such as

* + - * Conversion of one form of electrical power to another desired form and voltage, typically involving converting AC line voltage to a well-regulated lower- voltage DC for electronic devices. Low voltage, low power DC power supply units are commonly integrated with the devices they supply, such as computers and household electronics.
      * Batteries.
      * Chemical fuel cells and other forms of energy storage systems.
      * Solar power.
      * Generators or alternators.

##### Block Diagram:

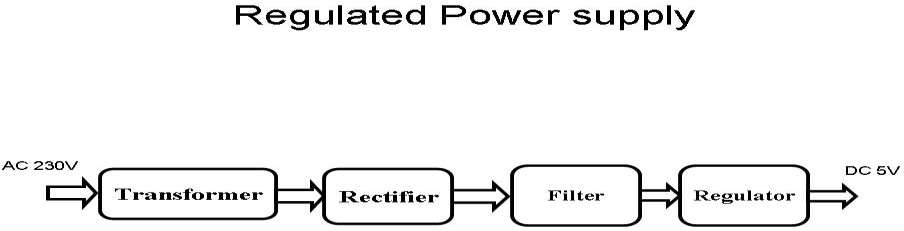
****

Fig 5.2.2(a): Regulated Power Supply

The basic circuit diagram of a regulated power supply (DC O/P) with led connected as load is shown in fig: 5.2.2(a).

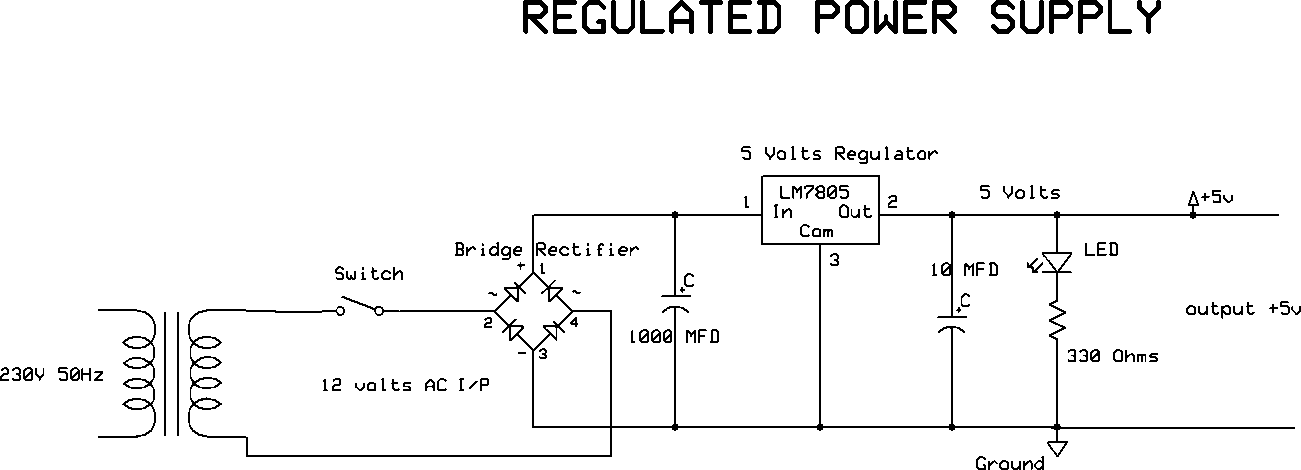


Fig 5.2.2(b): Circuit diagram of Regulated Power Supply with Led connection

The components mainly used in above figure are

* + - * 230V AC MAINS
      * TRANSFORMER
      * BRIDGE RECTIFIER(DIODES)
      * CAPACITOR
      * VOLTAGE REGULATOR (IC 7805)
      * RESISTOR
      * LED (LIGHT EMITTING DIODE)

The detailed explanation of each and every component mentioned above is as follows:

**Transformation:** The process of transforming energy from one device to another is called transformation. For transforming energy, we use transformers.

##### Transformers:

A transformer uses mutual induction to transfer electrical energy between two circuits without changing frequency. A varying current in the primary coil generates a changing magnetic field in the core. This magnetic field induces a voltage in the secondary coil through mutual induction. If a load is connected, current flows in the secondary, transferring energy.

The magnetic field behaves like that of a bar magnet, with moving lines inducing EMF. Figure 5.2.3 illustrates a step-down transformer operating on this principle.

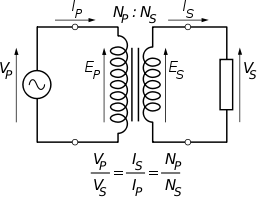


Fig 5.2.3: Step-Down Transformer

The voltage induced in the secondary is determined by the TURNS RATIO.



For example, if the secondary has half the primary turns; the secondary will have half the primary voltage.

Another example is if the primary has 5000 turns and the secondary has 500 turns, then the turn’s ratio is 10:1. If the primary voltage is 240 volts, then the secondary voltage will be x 10 smaller = 24 volts. Assuming a perfect transformer, the power provided by the primary must equal the power taken by a load on the secondary. If a 24-watt lamp is connected across a 24- volt secondary, then the primary must supply 24 watts.

To aid magnetic coupling between primary and secondary, the coils are wound on a metal CORE. Since the primary would induce power, called EDDY CURRENTS, into this core, the core is LAMINATED. This means that it is made up from metal sheets insulated from each other. Transformers to work at higher frequencies have an iron dust core or no core at all.

Note that the transformer only works on AC, which has a constantly changing current and moving field. DC has a steady current and therefore a steady field and there would be no induction.

Some transformers have an electrostatic screen between primary and secondary. This is to prevent some types of interference being fed from the equipment down into the mains supply, or in the other direction. Transformers are sometimes used for IMPEDANCE MATCHING.

We can use the transformers as step up or step down.

##### Battery power supply:

A [battery](http://en.wikipedia.org/wiki/Battery_(electricity)) is a type of linear power supply that offers benefits that traditional line-operated power supplies lack: mobility, portability and reliability. A battery consists of multiple electrochemical cells connected to provide the voltage desired.

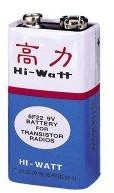


Fig 5.2.4(a): Hi-Watt 9V Battery

The most commonly used [dry-cell](http://en.wikipedia.org/wiki/Dry-cell) battery is the [carbon-zinc](http://en.wikipedia.org/wiki/Carbon-zinc) dry cell battery. Dry-cell batteries are made by stacking a carbon plate, a layer of electrolyte paste, and a zinc plate alternately until the desired total voltage is achieved. The most common dry-cell batteries have one of the following voltages: 1.5, 3, 6, 9, 22.5, 45, and 90. During the discharge of a carbon-zinc battery, the zinc metal is converted to a zinc salt in the electrolyte, and magnesium dioxide is reduced at the carbon electrode. These actions establish a voltage of approximately 1.5 V.

When the battery is charging, the lead sulfate is converted back to lead and lead dioxide A [nickel-cadmium](http://en.wikipedia.org/wiki/Nickel-cadmium) battery has become more popular in recent years. This battery cell is completely sealed and rechargeable. The electrolyte is not involved in the electrode reaction, making the voltage constant over the span of the batteries long service life. During the charging process, nickel oxide is oxidized to its higher oxidation state and cadmium oxide is reduced. The nickel-cadmium batteries have many benefits. They can be stored both charged and uncharged. They have a long service life, high current availabilities, constant voltage, and the ability to be recharged. Fig: 5.2.4(b) shows pencil battery of 1.5V.



Fig 5.2.4(b): Pencil Battery of 1.5V

**Rectification:**

The process of converting an alternating current to a pulsating direct current is called as rectification. For rectification purpose we use rectifiers.

##### Rectifiers:

A rectifier is an electrical device that converts alternating current (AC) to direct current (DC), a process known as rectification. Rectifiers have many uses including as components of power supplies and as detectors of radio signals. Rectifiers may be made of solid-state diodes, vacuum tube diodes, mercury arc valves, and other components.

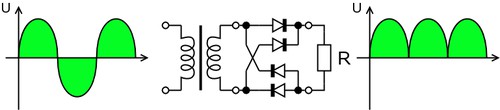
A device that it can perform the opposite function (converting DC to AC) is known as an inverter.

When only one diode is used to rectify AC (by blocking the negative or positive portion of the waveform), the difference between the term diode and the term rectifier is merely one of usage, i.e., the term rectifier describes a diode that is being used to convert AC to DC.

Almost all rectifiers comprise a number of diodes in a specific arrangement for more efficiently converting AC to DC than is possible with only one diode. Before the development of silicon semiconductor rectifiers, vacuum tube diodes and copper (I) oxide or selenium rectifier stacks were used.

**Bridge full wave rectifier:** The Bridge rectifier circuit is shown in fig: 5.2.5(a), which converts an ac voltage to dc voltage using both half cycles of the input ac voltage. The Bridge rectifier circuit is shown in the figure. The circuit has four diodes connected to form a bridge. The ac input voltage is applied to the diagonally opposite ends of the bridge. The load resistance is connected between the other two ends of the bridge. For the positive half cycle of the input ac voltage, diodes D1 and D3 conduct, whereas diodes D2 and D4 remain in the OFF state. The conducting diodes will be in series with the load resistance RL and hence the load current flows through RL. For the negative half cycle of the input ac voltage, diodes D2 and D4 conduct whereas, D1 and D3 remain OFF. The conducting diodes D2 and D4 will be in series with the load resistance RL and hence the current flows through RL in the same direction as in the previous half cycle. Thus, a bi-directional wave is converted into a unidirectional wave.

**Input Output**

****

**Fig 5.2.5(a): Bridge rectifier: a full-wave rectifier using 4 diodes**

###### DB107:

Now -a -days Bridge rectifier is available in IC with a number of DB107. In our project we are using an IC in place of bridge rectifier. The picture of DB 107 is shown in fig: 5.2.5(b).

Features:

* Good for automation insertion
* Surge overload rating - 30 amperes peak
* Ideal for printed circuit board
* Reliable low-cost construction utilizing molded
* Glass passivated device
* Polarity symbols molded on body
* Mounting position: Any



Fig 5.2.5(b): DB107

**Filtration:**

The process of converting a pulsating direct current to a pure direct current using filters is called as filtration.

Filters:

Electronic filters are electronic circuits, which perform signal-processing functions, specifically to remove unwanted frequency components from the signal, to enhance wanted ones.

##### Capacitors:

A capacitor is a passive component that stores energy as an electrostatic field between two electrically separated conductive plates.

When voltage is applied, a charging current flows, creating opposite charges on each plate until the capacitor is fully charged. Capacitors can be constructed in various forms, including the common electrolytic type, as shown in figures 5.2.6(a) and 5.2.6(b)

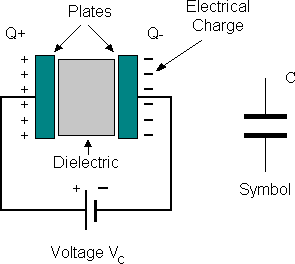
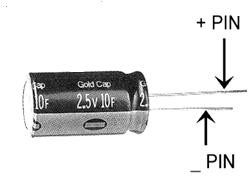


Fig 5.2.6(a): Construction Of a Capacitor

****

**Fig 5.2.6(b): Electrolytic Capacitor**

Units of Capacitance:

Microfarad (μF) 1μF = 1/1,000,000 = 0.000001 = 10-6 F

Nano-farad (nF) 1nF = 1/1,000,000,000 = 0.000000001 = 10-9 F

Pico farad (pF) 1pF = 1/1,000,000,000,000 = 0.000000000001 = 10-12 F

##### Operation of Capacitor:

Think of water flowing through a pipe. If we imagine a capacitor as being a storage tank with an inlet and an outlet pipe, it is possible to show approximately how an electronic capacitor works.

First, let's consider the case of a "coupling capacitor" where the capacitor is used to connect a signal from one part of a circuit to another but without allowing any direct current to flow.

If the current flow is alternating between zero and a maximum, our "storage tank" capacitor will allow the current waves to pass through.

However, if there is a steady current, only the initial short burst will flow until the "floating ball valve" closes and stops further flow.

So, a coupling capacitor allows "alternating current" to pass through because the ball valve doesn't get a chance to close as the waves go up and down. However, a steady current quickly fills the tank so that all flow stops.

A capacitor will pass alternating current but (apart from an initial surge) it will

not pass d.c.

Where a capacitor is used to decouple a circuit, the effect is to "smooth out ripples". Any ripples, waves or pulses of current are passed to ground while d.c. Flows smoothly.

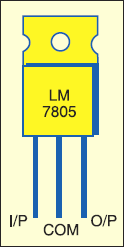
Regulation:

The process of converting a varying voltage to a constant regulated voltage is called as regulation. For the process of regulation, we use voltage regulators.

##### Voltage Regulator:

A voltage regulator (also called a ‘regulator’) with only three terminals appears to be a simple device, but it is in fact a very complex integrated circuit. It converts a varying input voltage into a constant ‘regulated’ output voltage. Voltage Regulators are available in a variety of outputs like 5V, 6V, 9V, 12V and 15V. The LM78XX series of voltage regulators are designed for positive input. For applications requiring negative input, the LM79XX series is

used. Using a pair of ‘voltage-divider’ resistors can increase the output voltage of a regulator circuit.

It is not possible to obtain a voltage lower than the stated rating. You cannot use a 12V regulator to make a 5V power supply. Voltage regulators are very robust. These can withstand over-current draw due to short circuits and also over-heating. In both cases, the regulator will cut off before any damage occurs. The only way to destroy a regulator is to apply reverse voltage to its input. Reverse polarity destroys the regulator almost instantly. Fig: 5.2.7shows voltage regulator.

**Fig 5.2.7: Voltage Regulator**

##### Resistors:

A resistor is a two-terminal electronic component that produces a voltage across its terminals that is proportional to the electric current passing through it in accordance with Ohm's law:

*V* = *IR*

Resistors are elements of electrical networks and electronic circuits and are ubiquitous in most electronic equipment. Practical resistors can be made of various compounds and films, as well as resistance wire (wire made of a high-resistivity alloy, such as nickel/chrome).

The primary characteristics of a resistor are the resistance, the tolerance, maximum working voltage and the power rating. Other characteristics include temperature coefficient, noise, and inductance. Less well-known is critical resistance, the value below which power dissipation limits the maximum permitted current flow, and above which the limit is applied voltage. Critical resistance is determined by the design, materials and dimensions of the resistor.

Resistors can be made to control the flow of current, to work as Voltage dividers, to dissipate power and it can shape electrical waves when used in combination of other components. Basic unit is ohms.

##### Theory of operation:

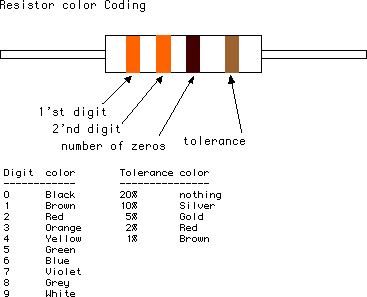
Ohm's law:

The behavior of an ideal resistor is dictated by the relationship specified in Ohm's law: V = IR.Ohm's law states that the voltage (V) across a resistor is proportional to the current (I) through it where the constant of proportionality is the resistance (R).

Power dissipation:

The power dissipated by a resistor (or the equivalent resistance of a resistor network) is calculated using the following:

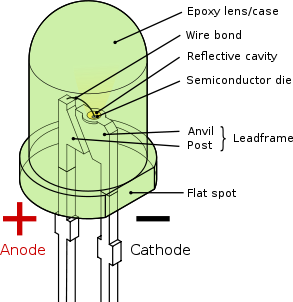
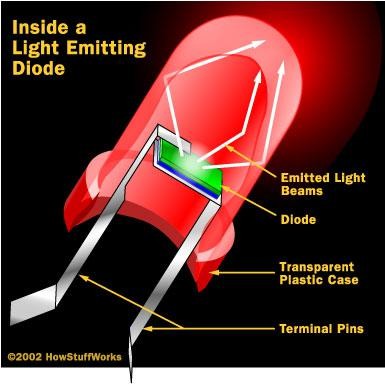
P = I^2 R = I V = \frac{V^2}{R}



**Fig 5.2.8(a): Resistor Fig 5.2.8(b): Color Bands in Resistor**

#### LED:

A light-emitting diode (LED) is a semiconductor light source. LEDs are used as indicator lamps in many devices, and are increasingly used for lighting. Introduced as a practical electronic component in 1962, early LEDs emitted low-intensity red light, but modern versions are available across the visible, ultraviolet and infrared wavelengths, with very high brightness. The internal structure and parts of a led are shown in figures 3.4.1 and 3.4.2 respectively.



**Fig 5.3(a): Inside a LED Fig 5.3(b): Parts of a LED**

##### Working:

The structure of an LED is simple yet robust, differing greatly from that of a traditional light bulb. It is built around a light-emitting semiconductor material that determines the colour of

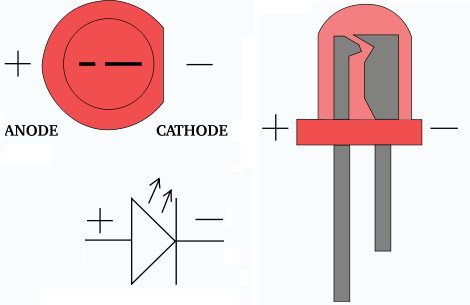


Fig 5.3.1: Electrical Symbol & Polarities of LED

light. LEDs are based on semiconductor diodes and operate on the principle of electroluminescence. When forward biased, electrons recombine with holes, releasing energy as photons (light). The colour of the light depends on the energy gap of the semiconductor material used. LEDs are usually all in size, often less than 1 mm².They include integrated optical components to shape light and enhance reflection. LEDs offer advantages such as low power consumption, long life, and durability. They also provide faster switching speeds and compact size compared to incandescent bulbs. However, LEDs are more expensive and require careful current and heat management. Despite the cost, they are increasingly used in automotive

lighting and traffic signals. Their small size supports new technologies like text and video displays. High switching rates make them ideal for communication systems. They outperform traditional sources in efficiency and reliability. The electrical symbol and polarities of an LED are typically illustrated in Fig: 5.3.1.

LED lights have a variety of advantages over other light sources:

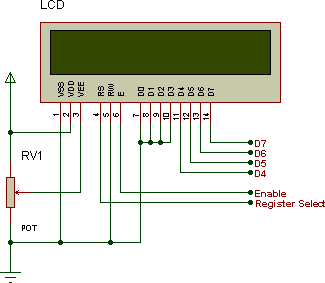
* + - * High-levels of brightness and intensity
      * High-efficiency
      * Low-voltage and current requirements
      * Low radiated heat
      * High reliability (resistant to shock and vibration)
      * No UV Rays
      * Long source life
      * Can be easily controlled and programmed Applications of LED fall into three major categories:
      * Visual signal application where the light goes more or less directly from the LED to the human eye, to convey a message or meaning.
      * Illumination where LED light is reflected from object to give visual response of these objects.
      * Generate light for measuring and interacting with processes that do not involve the human visual system.

#### LCD DISPLAY

##### LCD Background:

One of the most common devices attached to a micro controller is an LCD display. Some of the most common LCDs connected to the many microcontrollers are 16x2 and 20x2 displays. This means 16 characters per line by 2 lines and 20 characters per line by 2 lines, respectively.

**Basic 16x 2 Characters LCD**

****

**Figure 5.4.1: LCD Pin diagram**

##### Pin description:

Character LCD pins with Microcontroller

|  |  |  |
| --- | --- | --- |
| **Pin No.** | **Name** | **Description** |
| Pin no. 1 | **VSS** | Power supply (GND) |
| Pin no. 2 | **VCC** | Power supply (+5V) |
| Pin no. 3 | **VEE** | Contrast adjusts |
| Pin no. 4 | **RS** | 0 = Instruction input  1 = Data input |
| Pin no. 5 | **R/W** | 0 = Write to LCD module  1 = Read from LCD module |
| Pin no. 6 | **EN** | Enable signal |
| Pin no. 7 | **D0** | Data bus line 0 (LSB) |
| Pin no. 8 | **D1** | Data bus line 1 |
| Pin no. 9 | **D2** | Data bus line 2 |
| Pin no. 10 | **D3** | Data bus line 3 |

|  |  |  |
| --- | --- | --- |
| Pin no. 11 | **D4** | Data bus line 4 |
| Pin no. 12 | **D5** | Data bus line 5 |
| Pin no. 13 | **D6** | Data bus line 6 |
| Pin no. 14 | **D7** | Data bus line 7 (MSB) |

The LCD requires 3 control lines as well as either 4 or 8 I/O lines for the data bus. The user may select whether the LCD is to operate with a 4-bit data bus or an 8-bit data bus. If a 4- bit data bus is used the LCD will require a total of 7 data lines (3 control lines plus the 4 lines for the data bus). If an 8-bit data bus is used the LCD will require a total of 11 data lines (3 control lines plus the 8 lines for the data bus).

The three control lines are referred to as **EN**, **RS**, and **RW**.

The **EN** line is called "Enable." This control line is used to tell the LCD that we are sending it data. To send data to the LCD, our program should make sure this line is low (0) and then set the other two control lines and/or put data on the data bus. When the other lines are completely ready, bring **EN** high (1) and wait for the minimum amount of time required by the LCD datasheet (this varies from LCD to LCD), and end by bringing it low (0) again.

The **RS** line is the "Register Select" line. When RS is low (0), the data is to be treated as a command or special instruction (such as clear screen, position cursor, etc.). When RS is high (1), the data being sent is text data which should be displayed on the screen. For example, to display the letter "T" on the screen we would set RS high.

The **RW** line is the "Read/Write" control line. When RW is low (0), the information on the data bus is being written to the LCD. When RW is high (1), the program is effectively querying (or reading) the LCD. Only one instruction ("Get LCD status") is a read command. All others are write commands--so RW will almost always be low.

##### Circuit Description:

The LCD's Enable (EN) and Register Select (RS) lines are connected to the control port, which is open drain. External 10K pull-up resistors are added to ensure compatibility with systems lacking internal pull-ups. The R/W pin is hardwired to write mode, so we can't read the LCD’s Busy Flag.

To avoid timing issues, fixed software delays are added after sending instructions. A 10K potentiometer is used to adjust the LCD contrast. A 5V power supply with decoupling capacitors is recommended for reliable operation. The EN line signals the LCD to execute a command set on the data and control lines. The EN line must be cleared (LOW) before setting up data: CLR EN. After setup, EN is set (HIGH) with SETB EN, held briefly, then cleared again. Instruction execution occurs when EN is finally brought LOW, completing the communication.

Checking the busy status of the LCD:

As previously mentioned, it takes a certain amount of time for each instruction to be executed by the LCD. The delay varies depending on the frequency of the crystal attached to the oscillator input of the LCD as well as the instruction which is being executed.

While it is possible to write code that waits for a specific amount of time to allow the LCD to execute instructions, this method of "waiting" is not very flexible. If the crystal frequency is changed, the software will need to be modified. A more robust method of programming is to use the "Get LCD Status" command to determine whether the LCD is still busy executing the last instruction received.

The "Get LCD Status" command will return to us two tidbits of information; the information that is useful to us right now is found in DB7. In summary, when we issue the "Get LCD Status" command the LCD will immediately raise DB7 if it's still busy executing a command or lower DB7 to indicate that the LCD is no longer occupied. Thus, our program can query the LCD until DB7 goes low, indicating the LCD is no longer busy. At that point we are free to continue and send the next command.

##### Applications:

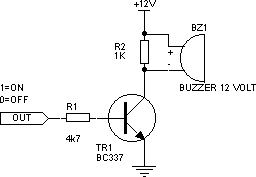
* + - * Medical equipment
      * Electronic test equipment
      * Industrial machinery Interface
      * Serial terminal
      * Advertising system
      * EPOS
      * Restaurant ordering systems
      * Gaming box
      * Security systems
      * R&D Test units

#### BUZZER

****

Fig 5.5: Buzzer

* A piezoelectric sound component uses a piezoelectric diaphragm as its sound source. The diaphragm consists of a ceramic plate with electrodes bonded to a metal plate. When DC voltage is applied, the piezoelectric effect causes mechanical distortion and bending.
* Applying AC voltage causes repeated bending, generating sound waves in the air. The metal plate remains rigid while the ceramic expands or contracts to produce vibration.
* To interface a buzzer the standard transistor interfacing circuit is used. Note that if a different power supply is used for the buzzer, the 0V rails of each power supply must be connected to provide a common reference.
* If a battery is used as the power supply, it is worth remembering that piezo sounders draw much less current than buzzers. Buzzers also just have one ‘tone’, whereas a piezo sounder is able to create sounds of many different tones.
* To switch on buzzer -high 1
* To switch off buzzer -low 1
* Notice (Handling) In Using Self Drive Method
* When the piezoelectric buzzer is set to produce intermittent sounds, sound may be heard continuously even when the self-drive circuit is turned ON / OFF at the "X" point shown in Fig. 5.5. This is because of the failure of turning off the feedback voltage.
* Build a circuit of the piezoelectric sounder exactly as per the recommended circuit shown in the catalog of the transistor and circuit constants are designed to ensure stable oscillation of the piezoelectric sounder.
* Design switching which ensures direct power switching.
* The self-drive circuit is already contained in the piezoelectric buzzer. So, there is no need to prepare another circuit to drive the piezoelectric buzzer.
* Rated voltage (3.0 to 20Vdc) must be maintained. Products which can operate with voltage higher than 20Vdc are also available.
* Do not place resistors in series with the power source, as this may cause abnormal oscillation. If a resistor is essential to adjust sound pressure, place a capacitor (about 1μF) in parallel with the piezo buzzer.
* Do not close the sound emitting hole on the front side of casing.
* Carefully install the piezo buzzer so that no obstacle is placed within 15mm from the sound release hole on the front side of the casing.



**Fig 5.5(b): Buzzer logic circuit**

## Heart Beat Sensor

This heart beat sensor is designed to give digital output of heat beat when a finger is placed inside it. When the heart detector is working, the top-most 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.

##### Features

* + - * Heat beat indication by LED
      * Instant output digital signal for directly connecting to microcontroller
      * Compact Size
      * Working Voltage +5V DC
      * Applications
      * Digital Heart Rate monitor
      * Bio-Feedback control of robotics and applications
      * Exercise machines

##### Working:

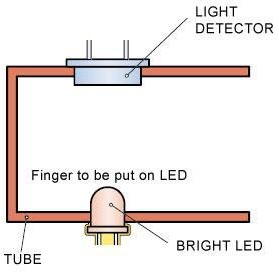
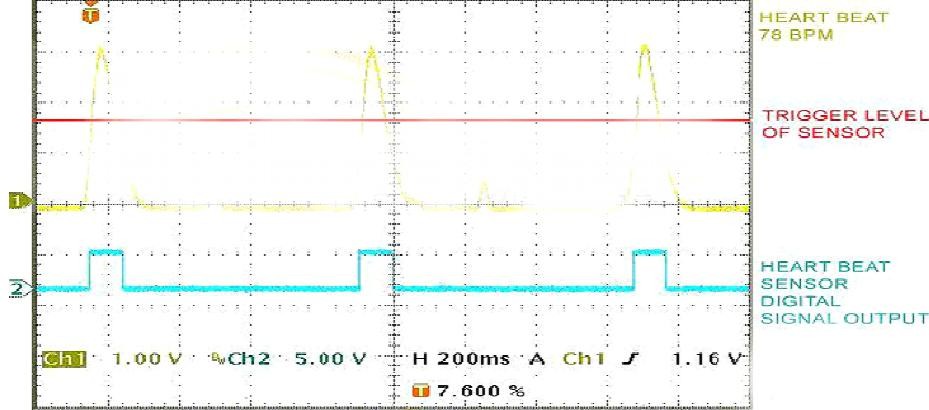
The sensor consists of a super bright red LED and light detector. The LED needs to be super bright as the light must pass through finger and detected at another end. Now, when the heart pumps a pulse of blood through the blood vessels, the finger becomes slightly opaquer and so less light reached the 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 on top by a LED which blinks on each heartbeat. Following figure shows signal of heart beat and sensor signal output graph.

Fig 5.6.2(a): Sensor construction



**fig 5.6.2(b): Signal view**

Fig.5.6.2(b) shows actual heart beat received by detector (Yellow) and the trigger point of sensor (Red) after which the sensor outputs digital signal (Blue) at 5V level.

###### AMPLIFIER CIRCUIT

For amplification, we use IC LM 358. Pulse rate is sensed by using a high intensity type LED and LDR. The finger is inserted in probe and red light from high intensity LED is allowed to fall on the finger. The amount of red light absorbed by finger varies according to the pulsatile blood flow in the finger. Therefore, the amount of light transmitted varies according to the blood flow.

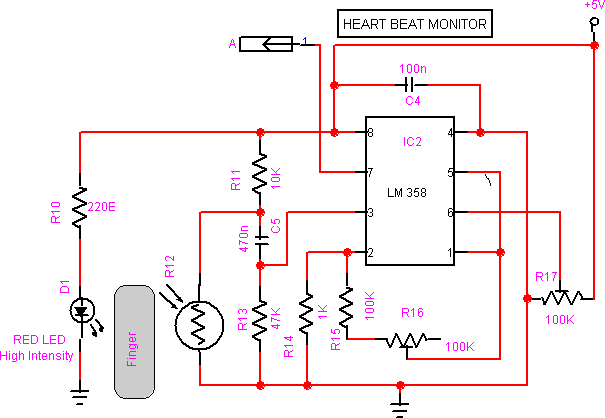


Fig 5.6.2(c): Amplifier circuit

An LDR placed opposite an LED detects transmitted light through a finger. Its resistance decreases with increased light intensity and increases when light decreases. A voltage divider converts the LDR resistance into a voltage signal. This signal contains both AC and DC components. DC components arise from static tissues and non-pulsatile blood, offering no pulse data.AC components represent pulsatile blood flow and contain the useful pulse information. Since DC is 100–1000 times stronger, a high-pass filter removes it.The filtered AC signal is amplified to volt-levels for processing. A comparator converts the signal into clean digital pulses based on a threshold. A PIC microcontroller calculates heart rate from pulse intervals and displays it on an LCD.

##### DHT11- Humidity and Temperature Sensor

DHT11 digital temperature and humidity sensor is a calibrated digital signal output of the temperature and humidity combined sensor. It uses dedicated digital modules capture technology and the temperature and humidity sensor technology to ensure that products with high reliability and excellent long-term stability. Humidity Sensor is one of the most important devices that has been widely in consumer, industrial, biomedical, and environmental etc. applications for measuring and monitoring Humidity.

Humidity is defined as the amount of water present in the surrounding air. This water content in the air is a key factor in the wellness of mankind. But if the temperature is 100C and the humidity is high i.e. the water content of air is high, then we will feel quite uncomfortable. Humidity is also a major factor for operating sensitive equipment like electronics, industrial equipment, electrostatic sensitive devices and high voltage devices etc. Such sensitive equipment must be operated in a humidity environment that is suitable for the device.

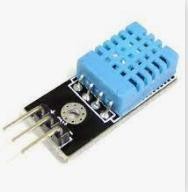


fig 5.7: DHT 11 sensor

Hence, sensing, measuring, monitoring and controlling humidity is a very important task. Some of the important areas of application for sensing, measuring and controlling Humidity are mentioned below.

**Domestic**: Sensing and controlling humidity in our homes and offices is important as higher humidity conditions will affect the blood flow. Other areas include cooking, indoor plantation etc.

**Industrial**: In industries like refineries, chemical, metal, or other industries where furnaces are used, high humidity will reduce the amount of oxygen in the air and hence reduces the firing rate. Other industries like food processing, textile, paper etc. also need control of humidity.

**Moisture**: Generally, the term Moisture means water content of any material or substance. But practically, the term Moisture refers to the water content in solids and liquids. The term Humidity refers to the water content in gases (air).

**Absolute Humidity:** Absolute Humidity (AH) is the ratio of mass of the water vapor to the volume of the air. If m is the mass of the water vapor and V is the total volume i.e. volume of air and water vapor mixture, then Absolute Humidity AH is given by

AH = m/V

Absolute Humidity doesn’t take temperature in to account but it changes with temperature and pressure.

**Relative Humidity:** Whenever we talk about measuring Humidity, it usually Relative Humidity that we are talking about (unless otherwise specified).

Relative Humidity or RH is the ratio of the actual water vapor pressure present in the air at a temperature to the maximum water vapor pressure present in the air at the same temperature.

In weather reports and forecasts, the probability of precipitation or dew or fog is indicated using Relative Humidity and hence, it is considered an important metric.

Relative Humidity takes both temperature and pressure in to consideration. Hence, the Humidity Sensors which measure Relative Humidity, measure both the moisture content as well as the air temperature.

**NOTE**: For temperatures above 1000C, measuring Relative Humidity (RH) is of no use as it would deliver misleading values.

**Specific Humidity:** Specific Humidity (SH) is the ratio of mass of the water vapor to the total mass of the air. Mixing Ratio or Humidity Ratio: Mixing Ratio is the ratio of mass of the water vapor to mass of the dry air.

###### [DHT11](http://www.adafruit.com/products/386)

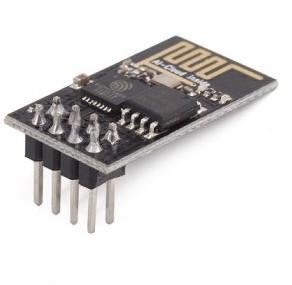
* Ultra-low cost
* 3 to 5V power and I/O
* 2.5mA max current use during conversion (while requesting data)
* Good for 20-80% humidity readings with 5% accuracy
* Good for 0-50°C temperature readings ±2°C accuracy
* No more than 1 Hz sampling rate (once every second)
* Body size 15.5mm x 12mm x 5.5mm

1. Working and Storage Conditions: Use of the sensor may cause up to 3% RH temporary drift, which will recover under normal conditions. To accelerate recovery, follow the “recovery process”. Avoid long-term exposure to condensation, dryness, and environments with smoke or corrosive gases like sulphur dioxide or HCl.

Recommended storage: Temperature 10–40℃, Humidity ≤ 60% RH2.

1. Effects of exposure to chemical substances Sensing resistive humidity sensor will be disturbed chemical vapor layer, the diffusion layer in the induction of chemicals may cause drift and measurement sensitivity. In a clean environment, slowly release contaminants out. The recovery process described below to accelerate the process.
2. Temperature Effect Relative humidity of the gas is largely dependent on temperature. Therefore, when measuring the humidity should be possible to ensure that the humidity sensor works at the same temperature. If you share a printed circuit board with electronic components heat released in the sensor should be installed as far as possible away from the electronic components, and installed at the bottom of the heat source, while maintaining a well-ventilated enclosure.
3. Light effects Prolonged exposure to sunlight or strong ultraviolet radiation, will reduce performance.
4. Recovery process Placed under extreme operating conditions or chemical vapor sensors, through the following process, you can return it to the state calibration. 45 ℃ and humidity under 70% RH conditions were maintained for more than 5 hours.

#### IOT ESP8266 MODULE

****

**fig 5.8: IOT module**

##### Introduction

Espressif Systems’ Smart Connectivity Platform (ESCP) of high performance wireless SOCs, for mobile platform designers, provides unsurpassed ability to embed Wi-Fi capabilities within other systems, at the lowest cost with the greatest functionality

##### Technology Overview

ESP8266 offers a complete and self-contained Wi-Fi networking solution, allowing it to either host the application or to offload all Wi-Fi networking functions from another application processor.

When ESP8266 hosts the application, and when it is the only application processor in the device, it is able to boot up directly from an external flash. It has integrated cache to improve the performance of the system in such applications, and to minimize the memory requirements.

Alternately, serving as a Wi-Fi adapter, wireless internet access can be added to any microcontroller-based design with simple connectivity through UART interface or the CPU AHB bridge interface.

ESP8266 on-board processing and storage capabilities allow it to be integrated with the sensors and other application specific devices through its GPIOs with minimal development up-front and minimal loading during runtime. With its high degree of on-chip integration, which includes the antenna switch balun, power management converters, it requires minimal external circuitry, and the entire solution, including front-end module, is designed to occupy minimal PCB area.

##### Features

* + - * 802.11 b/g/n protocol
      * Wi-Fi Direct (P2P), soft-AP
      * Integrated TCP/IP protocol stack
      * Integrated TR switch, balun, LNA, power amplifier and matching network
      * Integrated PLL, regulators, and power management units

 +19.5dBm output power in 802.11b mode

##### Specifications

Current Consumption

The following current consumption is based on 3.3V supply, and 25 C ambient, using internal regulators. Measurements are done at antenna port without SAW filter. All the transmitter’s measurements are based on 90% duty cycle, continuous transmit mode.

|  |  |  |
| --- | --- | --- |
| **Mode** | **Typ** | **Unit** |
| **Transmit 802.11b, CCK 1Mbps, POUT=+19.5dBm** | 215 | mA |
| **Transmit 802.11b, CCK 11Mbps, POUT=+18.5dBm** | 197 | mA |
| **Receive 802.11b, packet length=1024 byte, -80dBm** | 60 | mA |

|  |  |  |
| --- | --- | --- |
| **Receive 802.11g, packet length=1024 byte, -70dBm** | 60 | mA |
| **Receive 802.11n, packet length=1024 byte, -65dBm** | 62 | mA |
| **Standby** | 0.9 | mA |
| **Deep sleep** | 10 | uA |
| **Power save mode DTIM 1** | 1.2 | mA |
| **Power save mode DTIM 3** | 0.86 | mA |
| **Total shutdown** | 0.5 | uA |

CPU, Memory and Interfaces CPU

This chip embeds an ultra-low power Micro 32-bit CPU, with 16-bit thumb mode. This CPU can be interfaced using: code RAM/ROM interface (iBus) that goes to the memory controller, that can also be used to access external flash memory, data RAM interface (dBus), that also goes to the memory controller AHB interface, for register access, and JTAG interface for debugging

Memory Controller

The memory controller contains ROM, and SRAM. It is accessed by the CPU using the iBus, dBus and AHB interface. Any of these interfaces can request access to the ROM or RAM modules, and the memory controller arbiters serve these 3 interfaces on a first-come-first-serve basis.

AHB and AHB Blocks

The AHB blocks performs the function of an arbiter, controls the AHB interfaces from the MAC, SDIO (host) and CPU. Depending on the address, the AHB data requests can go into one of the two slaves: APB block, or flash controller (usually for standalone applications).

Data requests to the memory controller are usually high-speed requests, and requests to the APB block are usually register access.

The APB block acts as a decoder. It is meant only for access to programmable registers within ESP8266’s main blocks. Depending on the address, the APB request can go to the radio, SI/SPI, SDIO (host), GPIO, UART, real-time clock (RTC), MAC or digital baseband.

Master SI / SPI Control (Optional)

The master serial interface (SI) can operate in two, three or four-wire bus configurations to control the EEPROM or other I2C/SPI devices. Multiple I2C devices with different device addresses are supported by sharing the 2-wire bus. Multiple SPI devices are supported by sharing the clock and data signals, using separate software controlled GPIO pins as chip selects. The SPI can be used for controlling external devices such as serial flash memories, audio CODECs, or other slave devices. It is set up as a standard master SPI device with 3 different enable pins: SPI\_EN0, SPI\_EN1, SPI\_EN2.

Both SPI master and SPI slave are supported with the latter being used as a host interface. SPI\_EN0 is used as an enable signal to an external serial flash memory for downloading patch code and/or MIB-data to the baseband in an embedded application. In a host-based application, patch code and MIB-data can alternatively be downloaded via the host interface. This pin is active low and should be left open if not used. SPI\_EN1 is usually used for a user application,

e.g. to control an external audio codec or sensor ADC, in an embedded application. This pin is active low and should be left open if not used.

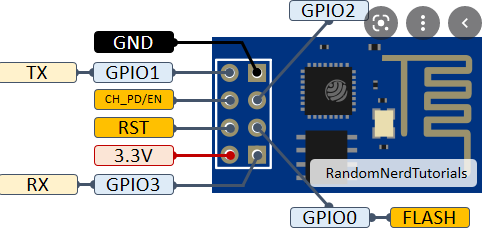
SPI\_EN2 usually controls an EEPROM to store individual data, such as MIB information, MAC address, and calibration data, or for general use. This pin is active low and should be left open if not used.

General Purpose IO

There are up to 16 GPIO pins. They can be assigned to various functions by the firmware. Each GPIO can be configured with internal pull-up/down, input available for sampling by a software register, input triggering an edge or level CPU interrupt, input triggering a level wakeup interrupt, open-drain or push-pull output driver, or output source from a software register, or a sigma-delta PWM DAC. These pins are multiplexed with other functions such as host interface, UART, SI, Bluetooth coexistence, etc.

Digital IO Pads

The digital IO pads are bidirectional, non-inverting and tri-state. It includes input and an output buffer with tristate control inputs. Besides this, for low power operations, the IO can also be set to hold. For instance, when we power down the chip, all output enable signals can be set to hold low.



**fig 5.8.4 IOT module pin diagram**

##### Applications

* + - * Wi-Fi Smart Hardware Converted from UART Serial Ports
      * Sensor
      * Smart Light
      * Smart Plug

Wi-Fi (Short for Wireless Fidelity) is a wireless technology that uses radio frequency to transmit data through the air. Wi-Fi has initial speeds of 1mbps to 2mbps. Wi-Fi transmits data in the frequency band of 2.4 GHz. It implements the concept of frequency division multiplexing technology. Range of Wi-Fi technology is 40-300 feet.

### CHAPTER

6

#### SOFTWARE SIMULATION

*This chapter involves using virtual tools or programs to model, test, and analyse the system's behaviour before actual implementation.*

The Arduino Software (IDE) makes it easy to write code and upload it to the board offline. We recommend it for users with poor or no internet connection. This software can be used with any Arduino board. There are currently two versions of the Arduino IDE, one of which is IDE 2.0.0.

##### 6.1 Arduino IDE – Compiler

Here are currently two versions of the Arduino IDE, one is the IDE 1.x.x and the other is IDE

2.x. The IDE 2.x is new major release that is faster and even more powerful to the IDE 1.x.x. In addition to a more modern editor and a more responsive interface it includes advanced features to help users with their coding and debugging.

The following steps can guide you with using the offline IDE (you can choose either IDE 1.x.x or IDE 2.x):

1. Download and install the Arduino Software IDE:
   * Arduino IDE 1.x.x ([Windows](https://www.arduino.cc/en/Guide/Windows), [Mac OS](https://www.arduino.cc/en/Guide/macOS), [Linux](https://www.arduino.cc/en/Guide/Linux), [Portable IDE](https://www.arduino.cc/en/Guide/PortableIDE) for Windows and Linux, [ChromeOS](https://chrome.google.com/webstore/detail/arduino-create/dcgicpihgkmccjigalccipmjlnjopdfe)).
   * [Arduino IDE 2.x](https://www.arduino.cc/en/Tutorial/getting-started-with-ide-v2/ide-v2-downloading-and-installing)
2. Connect your Arduino board to your device.
3. Open the Arduino Software (IDE).

The Arduino Integrated Development Environment - or Arduino Software (IDE) - connects to the Arduino boards to upload programs and communicate with them. Programs written using Arduino Software (IDE) are called sketches. These sketches are written in the text editor and

are saved with the file extension. Using the offline IDE 1.x.x

The editor contains the four main areas:

1. A Toolbar with buttons for common functions and a series of menus. The toolbar buttons allow you to verify and upload programs, create, open, and save sketches, and open the serial monitor.
2. The message area, gives feedback while saving and exporting and also displays errors.
3. The text editor for writing your code.
4. The text console displays text output by the Arduino Software (IDE), including complete error messages and other information.

The bottom right-hand corner of the window displays the configured board and serial port.

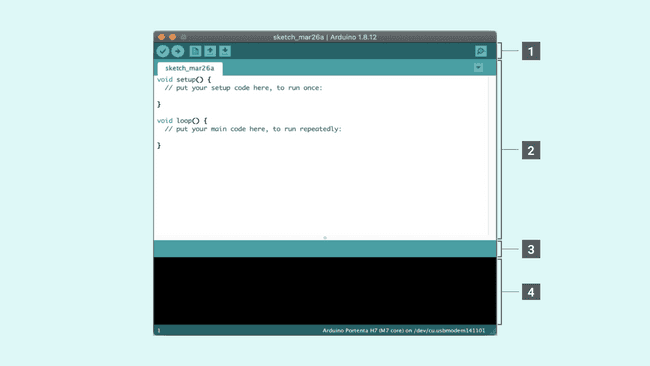


fig 6.1 (a): The Arduino Software IDE

Now that you are all set up, let’s try to make your board blink!

1. Connect your Arduino or Genuino board to your computer.
2. Now, you need to select the right core & board. This is done by navigating to Tools > Board

> Arduino AVR Boards > Board. Make sure you select the board that you are using. If you cannot find your board, you can add it from Tools > Board > Boards Manager.

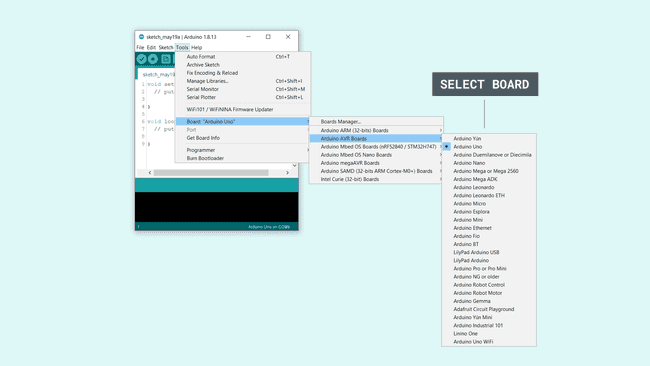


fig6.1(b): Selecting a board

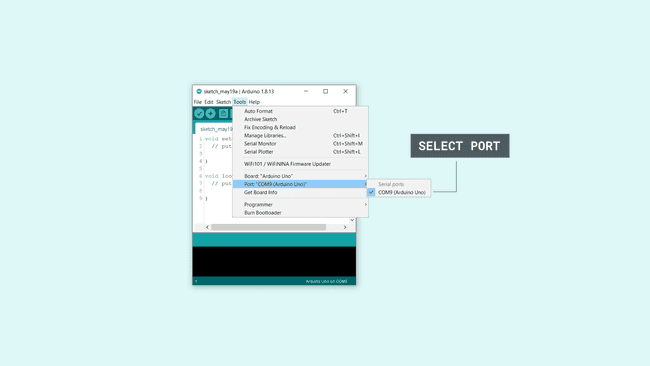
1. Now, let's make sure that your board is found by the computer, by selecting the port. This is simply done by navigating to Tools > Port, where you select your board from the list.

fig 6.1(c): Selecting the port

1. Let’s try an example: navigate to File > Examples >01.Basics> Blink.

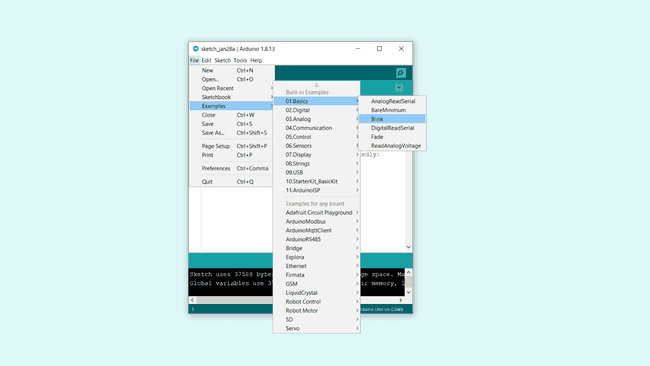


fig 6.1(d): Opening an example

1. To upload it to your board, simply click on the arrow in the top left corner. This process takes a few seconds, and it is important to not disconnect the board during this process. If the upload is successful, the message "Done uploading" will appear in the bottom output area.
2. Once the upload is complete, you should then see on your board the yellow LED with an L next to it start blinking. You can adjust the speed of blinking by changing the delay number in the parenthesis to 100, and upload the Blink sketch again. Now the LED should blink much faster.

The editor contains the four main areas:

1. A toolbar with buttons for common functions and a series of menus. The toolbar buttons allow you to verify and upload programs, create, open, and save sketches, choose your board and port and open the serial monitor.
2. The Sidebar for regularly used tools. It gives you quick access to board managers, libraries, debugging your board as well as a search and replacement tool.
3. The text editor for writing your code.
4. Console controls gives control over the output on the console.
5. The text console displays text output by the Arduino Software (IDE), including complete error messages and other information.

The bottom right-hand corner of the window displays the configured board and serial port.

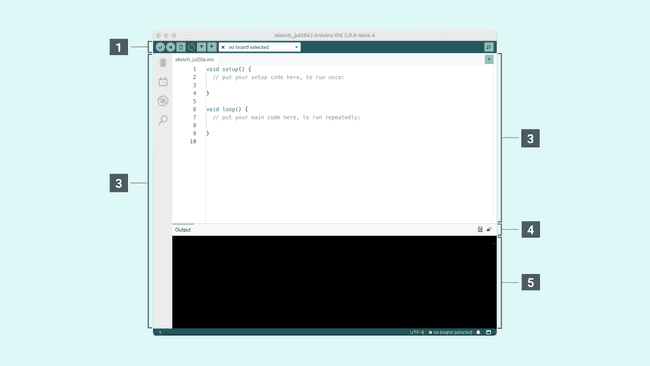


fig 6.1(e): The Arduino Software IDE

Now that you are all set up, let’s try to make your board blink!

1. Connect your Arduino or Genuino board to your computer.
2. Now, you need to select the right board & port. This is done from the toolbar. Make sure you select the board that you are using. If you cannot find your board, you can add it from the board manager in the sidebar.

Selecting a board & port

1. To upload it to your board, simply click on the arrow in the top left corner. This



Fig 6.1(f): Selection of board and port

Process takes a few seconds, and it is important to not disconnect the board during this process. If the upload is successful, "Done uploading" will appear in the bottom output

### CHAPTER

7

#### CODE

*This chapter refers to the set of written programming instructions that control the hardware and execute the desired functions of the system.*

#include <LiquidCrystal.h> #include <stdio.h> #include <Wire.h> #include "dht.h"

LiquidCrystal lcd(13, 12, 11, 10, 9, 8); unsigned char gchr='x',gchr1='x';

char rcv,count;

//char pastnumber[11]=""; #define dht\_apin 5

dht DHT; int fire = 4;

int buzzer = 7;

const int sensorPin = A0; // Analog pin A0 connected to the sensor signal int sensorValue = 0; // Variable to store the sensor reading

long lastPulseTime = 0; // To track the time of the last pulse long pulseDuration = 0; // Duration of each pulse

int pulseCount = 0; // Pulse count

int heartRate = 0; // Heart rate in beats per minute (BPM) float tempc=0,humc=0;

int hbv=0,hbv1=0;

int cntl=0,tempv=0,tempv1=0,x=0,y=0;

String inputString = ""; // a string to hold incoming databoolean stringComplete = false; // whether the string is complete void beep()

{

digitalWrite(buzzer, LOW);delay(1500);digitalWrite(buzzer, HIGH);

}

void okcheck(){

unsigned char rcr; do{

rcr = Serial.read();

}while(rcr != 'K');

}

void things\_send(){ unsigned char recr;

Serial.write("AT+CIPMUX=1\r\n");delay(2000); Serial.write("AT+CIPSTART=4,\"TCP\",\"184.106.153.149\",80\r\n"); delay(4000); //OK LINKED

Serial.write("AT+CIPSEND=4,77\r\n"); delay(3000);

Serial.write("GET https://api.thingspeak.com/update?api\_key=0PHK2UK998YJ7DFU&");

}

void things\_rcv()

{

unsigned char recr; Serial.write("AT+CIPSTART=4,\"TCP\",\"184.106.153.149\",80\r\n"); delay(4000);

Serial.write("AT+CIPSEND=4,73\r\n"); delay(3000);

Serial.write("GET https://api.thingspeak.com/channels/449069/fields/3.json?results=1");

}

void things\_done()

{

Serial.write("\r\n\r\n"); delay(4000);

}

void setup()

{

Serial.begin(9600);//serialEvent(); pinMode(fire, INPUT); pinMode(buzzer, OUTPUT); pinMode(sensorPin, INPUT); digitalWrite(buzzer, HIGH); lcd.begin(16, 2);

cd.print(" HEALTH"); lcd.setCursor(0,1);

lcd.print(" ALERT SYSTEM"); delay(1500);

delay(1500); wifiinit(); lcd.clear();

lcd.setCursor(0, 0); //column,row lcd.print("T:"); //2,0

lcd.setCursor(8, 0); //column,row lcd.print("H:"); //10,0 lcd.setCursor(0,1); lcd.print("HR:"); //2,1 lcd.setCursor(8,1);

// lcd.print("F:"); //10,1

}

void loop()

{

DHT.read11(dht\_apin); tempc = DHT.temperature; humc = DHT.humidity;

lcd.setCursor(2,0);convertl(tempc); lcd.setCursor(10,0);convertl(humc);

hbv = analogRead(A0); if(hbv > 700)

{

hbv1 = (hbv/10) - 1;

}

else

{

hbv1=0;

}

lcd.setCursor(3,1);convertl(hbv1); delay(3000);

if(tempc > 38)

{beep(); lcd.setCursor(13,1);lcd.print("1 "); things\_send(); Serial.write("field5="); converts(tempc);

things\_done(); lcd.setCursor(13,1);lcd.print(" "); for(cntl=0;cntl<15;cntl++)

{lcd.setCursor(13,1);convertl1(cntl);delay(900);} lcd.setCursor(13,1);lcd.print(" ");

}

if(humc > 90)

{beep(); lcd.setCursor(13,1);lcd.print("2 ");

things\_send(); Serial.write("field6="); converts(humc); things\_done();

lcd.setCursor(13,1);lcd.print(" "); for(cntl=0;cntl<15;cntl++)

{lcd.setCursor(13,1);convertl1(cntl);delay(900);} lcd.setCursor(13,1);lcd.print(" ");

}

/\* if(digitalRead(fire) == LOW)

{

lcd.setCursor(10,1);lcd.print("ON "); beep(); lcd.setCursor(13,1);lcd.print("3 "); things\_send(); Serial.write("field4=00100"); things\_done(); lcd.setCursor(13,1);lcd.print(" " for(cntl=0;cntl<15;cntl++)

{lcd.setCursor(13,1);convertl1(cntl);delay(900);} lcd.setCursor(13,1);lcd.print(" ");

}

if(digitalRead(fire) == HIGH)

{

lcd.setCursor(10,1);lcd.print("OFF");

}\*/

}

int readSerial(char result[])

{

int i = 0; while (1)

{

while (Serial.available() > 0)

{

char inChar = Serial.read(); if (inChar == '\n')

{

result[i] = '\0'; Serial.flush(); return 0;

}

if (inChar != '\r') { result[i] = inChar; i++;

}

}

}

}

void wifiinit(){

Serial.write("AT\r\n"); delay(2000); Serial.write("ATE0\r\n"); okcheck();delay(2000); Serial.write("AT+CWMODE=3\r\n"); delay(2000);

Serial.write("AT+CWJAP=\"project\",\"project1234\"\r\n"); okcheck(); Serial.write("AT+CIPMUX=1\r\n");delay(3000);// okcheck(); lcd.clear();

lcd.print("Connected"); delay(3000);

}

void converts(unsigned int value){ unsigned int a,b,c,d,e,f,g,h;

a=value/10000; b=value%10000; c=b/1000; d=b%1000;

e=d/100; f=d%100;

g=f/10; h=f%10;

a=a|0x30; c=c|0x30; e=e|0x30; g=g|0x30; h=h|0x30;

Serial.write(a); Serial.write(c); Serial.write(e); Serial.write(g); Serial.write(h);}

void convertl(unsigned int value){ unsigned int a,b,c,d,e,f,g,h;

a=value/10000; b=value%10000; c =b/1000; d=b%1000;

e=d/100; f=d%100;

g=f/10; h=f%10;

a=a|0x30; c=c|0x30; e=e|0x30; g=g|0x30; h=h|0x30;

// lcd.write(a); lcd.write(c);

lcd.write(e);

lcd.write(g);

lcd.write(h);

}

void convertl1(unsigned int value){ unsigned int a, b, c, d, e, f, g, h;

a=value/10000; b=value%10000; c=b/1000; d=b%1000;

e=d/100; f=d%100;

g=f/10; h=f%10;

a=a|0x30; c=c|0x30; e=e|0x30; g=g|0x30; h=h|0x30;

// lcd. write(a);

// lcd. write(c);

// lcd. write(e); lcd. write (g);

lcd. write (h);}

### CHAPTER

8

#### RESULTS

*This chapter presents the outcomes or data obtained from testing and running the system, demonstrating its performance and effectiveness.*

The project successfully demonstrated a real-time health monitoring system using Arduino and IoT technologies. Vital signs such as body temperature, heart rate, and oxygen saturation(SpO₂) were accurately measured using biomedical sensors. The data was continuously transmitted to an IoT platform (e.g., Thing Speak or Blynk), where it could be visualized and monitored remotely. In cases where the vital signs crossed predefined thresholds, instant alerts were generated and sent via SMS or app notifications to caregivers or medical personnel. The system proved to be reliable, cost-effective, and capable of providing timely health alerts, making it suitable for home care, elderly monitoring, and remote health services.

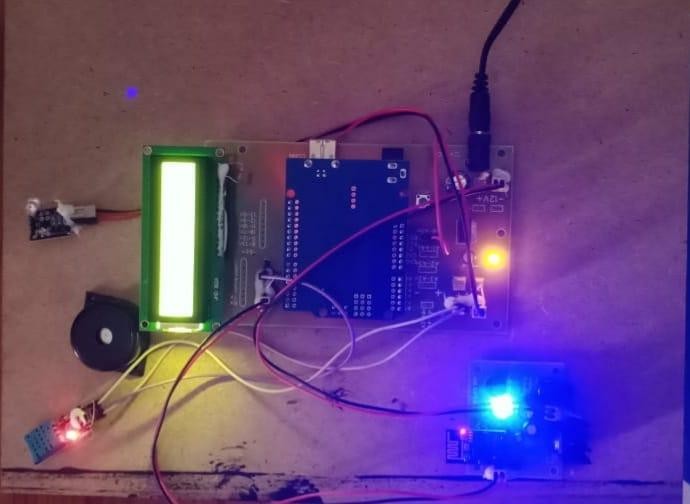
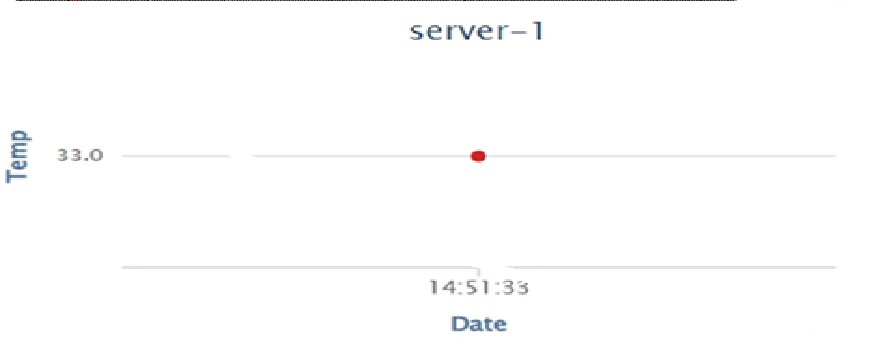


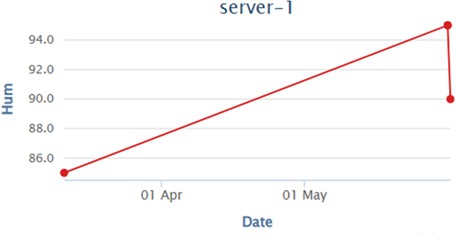
Fig 8(a): Implementation of Arduino-IOT powered wearable vital sign monitoring with Predictive Health Monitoring

**Fig 8(b): LCD display when system in ON state Fig 8(c):Output of the system on LCD**

****

**Fig 8(d): Temp values monitored on Think speak**

****

**Fig 8(e): Humidity values monitored on Thing speak**

### CHAPTER

9

#### ADVANTAGES &APPLICATIONS

*This chapter highlights the benefits and improvements offered by the proposed system over existing solutions.*

#### ADVANTAGES

* + - **Real-Time Health Monitoring** – Continuously tracks vital signs for immediate detection of health issues.
    - **Remote Access via IoT** – Allows doctors or caregivers to monitor patient data from anywhere.
    - **Instant Health Alerts** – Sends automatic notifications when abnormal readings are detected.
    - **Low-Cost and Scalable** – Uses affordable hardware, making it suitable for widespread use.
    - **User-Friendly Interface** – Provides easy-to-understand data visualization through mobile or web platforms.

#### APPLICATIONS

* + - **Home Health Monitoring** - Ideal for elderly or chronically ill patients to track their health conditions from home.
    - **Remote Patient Monitoring in Rural Areas** - Enables healthcare access in remote or underserved locations where medical facilities are limited.
    - **Hospital Patient Management** - Can be integrated into hospital systems to monitor multiple patients simultaneously and reduce workload.
    - **Post-Surgery Recovery Monitoring** - Assists in tracking patient recovery at home after surgery by keeping tabs on vital signs.

### CHAPTER

10

#### CONCLUSION&FUTURE SCOPE

*This chapter highlights the project's outcomes and impact, while the future scope suggests possible enhancements and extended applications.*

#### CONCLUSION

The Arduino-based wearable system offers a cost-effective solution for real-time health monitoring. It uses SpO2 and DHT11 sensors to track oxygen levels, heart rate, temperature, and humidity. Cloud connectivity enables remote access, while a buzzer alert warns during emergencies. Its lightweight design supports 24/7 use, ideal for elderly, chronic, and post-op care. By integrating IoT and predictive analytics, it ensures early detection and proactive healthcare.

#### FUTURE SCOPE

* + - Integration with Artificial Intelligence (AI):

AI algorithms can be used to predict potential health issues based on trends in vital sign data, enabling preventive healthcare.

* + - Emergency Response Automation:

The system can be linked directly to emergency services (ambulance, hospital hotlines) to provide faster medical assistance during critical conditions.

* + - Development of Wearable Health Devices:

The system can be miniaturized and embedded into wearable gadgets like smartwatches or fitness bands for continuous health tracking.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Project Details** | | | | |
| **Academic Year** | | 2024-25 | | |
| **Title of the Project** | | Arduino-IOT powered Wearable Vital Sign Monitoring with Predictive Health Monitoring | | |
| **Name of the Students and Hall Ticket No.** | | D. Shushrutha Reddy(22RA1A0447)  P. Rishika Reddy(22RA1A0438)  A. Shiva(22RA1A0444) | | |
| **Name of the Guide(s)** | | Dr. P. Anil Kumar | | |
| **Project PO Mapping** | | | | |
| **Name of Course from which Principles are applied in this project** | **Related Course Outcome Number** | **Description of the application** | **Page Number** | **Attained PO** |
| IOT(C31) | C31.2 | Introduction | 1-3 | PO1, PO3, PO4, PO11, PO12 |
|  |  | Literature survey | 4-7 | PO1, PO2, PO3, PO4, PO6, PO9, PO11, PO12 |
| MC(C31) | C31.2 | Design considerations | 8-11 | PO1, PO3, PO4, PO5, PO6, PO9, PO11, PO12 |
| IOT(C31) | C31.1 | Hardware description | 12-59 | PO1, PO3, PO5, PO6, PO9, PO11, PO12, PO12 |
| IOT(C31) | C31.4 | Software simulation | 60-64 | PO1, PO3, PO5, PO6, PO9, PO11, PO12, PSO2 |
| CS(C31) | C31.3 | Implementation | 65-72 | PO1, PO3, PO4, PO5, PO6, PO9, PO11, PO12 |
|  |  | Result | 73-74 | PO1, PO2, PO3, PO5, PO6, PO9, PO11, PSO1, PSO2, PSO3 |

Signature of the Students:

D. Shushrutha Reddy(22RA1A0447)

P. Rishika Reddy(22RA1A0438) **Signature of the Internal Guide**

A. Shiva(22RA1A0444) DEPT.OF ECE, KPRIT

#### REFERENCES

1. Patel A., et al., 2020 – Developed a ARDUINO-based health tracker for rural telehealth.
2. Sharma R., Bansal S., 2019 – Introduced GSM-based heartbeat and temperature monitoring system.
3. Zhou Y., Wang L., 2018 – Designed a wearable SpO2 monitor for post-operative care.
4. Ramesh P., et al., 2021 – Real-time health monitoring using MAX30100 and DHT11 with ESP32.
5. Kumar A., Singh V., 2020 – IoT-based system for chronic patient data tracking and prediction.
6. Ahmed S., et al., 2017 – ESP8266-based low-cost health monitoring prototype.
7. Rajput M., Verma T., 2022 – Real-time vitals monitoring with Firebase integration.
8. Joshi R., Kulkarni A., 2019 – Wearable system with predictive analytics for health anomalies.
9. Liang X., Wu H., 2021 – Smart patch using flexible sensors and Wi-Fi modules.
10. Choudhury D., et al., 2020 – Cloud-based predictive health monitoring model.
11. Goyal M., et al., 2018 – Bluetooth-enabled pulse oximeter for fitness and sleep monitoring.
12. Bhatnagar S., Mishra P., 2021 – DHT11 and MAX30102 integrated for IoT healthcare.
13. Kim J., et al., 2019 – Wearable SpO2 device for high-altitude physiology studies.
14. Anwar F., Khan I., 2021 – ESP32-based healthcare system for rural clinics.
15. Singh R., Desai N., 2017 – Sensor-embedded glove for industrial safety monitoring.
16. Paul D., et al., 2022 – AI-powered wearable for early COVID symptom detection.
17. Rao S., Patil K., 2020 – ESP32 health band with real-time dashboard.
18. Narayan R., Kulkarni N., 2019 – ESP8266 temperature and pulse tracker for school health.
19. Gupta T., et al., 2021 – Predictive modelling for soldier fatigue detection using sensors.
20. Verma K., Jha A., 2022 – Hybrid ML model for wearable COVID symptom detection.
21. Lee C., et al., 2020 – Smart shirt with Wi-Fi module for health tracking.
22. Srinivas M., Reddy P., 2018 – Wearable IoT for elderly fall and fever alerts.
23. Kaur H., Mehta R., 2021 – MAX30100 with ESP32 for quarantine zone alerting.
24. Rahman A., et al., 2019 – Solar-powered wearable SpO2 and vitals monitoring.
25. Deshmukh V., Kulkarni A., 2021 – Predictive IoT analytics for health risk alerts.