

TRANSMITTING PATIENT VITALS OVER RELIABLE ZIGBEE MESH NETWORK

Submitted in partial fulfillment of the requirements for the award of
Bachelor of Engineering degree in Electrical and Electronics Engineering

by

AARTHI A (Register No. 38140001)



**DEPARTMENT OF ELECTRICAL AND ELECTRONICS ENGINEERING
SCHOOL OF ELECTRICAL AND ELECTRONICS ENGINEERING**

SATHYABAMA

**INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)**

Accredited with Grade "A" by NAAC

JEPPIAAR NAGAR, RAJIV GANDHI SALAI, CHENNAI - 600 119

MARCH 2022



SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)

Accredited with Grade "A" by NAAC

JEPPIAAR NAGAR, RAJIV GANDHI SALAI, CHENNAI - 600 119

www.sathyabama.ac.in



DEPARTMENT OF ELECTRICAL AND ELECTRONICS ENGINEERING

BONAFIDE CERTIFICATE

This is to certify that this Project Report is the bonafide work of **AARTHI A** (Reg. No. 38140001) who carried out the project entitled "**TRANSMITTING PATIENT VITALS OVER RELIABLE ZIGBEE MESH NETWORK**" under our supervision from November 2021 to April 2022.

Dr. D. SUSITRA
Internal Guide

Dr. V. SIVACHIDAMBARANATHAN
Head of the Department

Submitted for Viva voce Examination held on _____

Name:

Signature:

Internal Examiner

External Examiner

DECLARATION

I **AARTHI A** (Reg. No. 38140001) hereby declare that the Project Report entitled **“TRANSMITTING PATIENT VITALS OVER RELIABLE ZIGBEE MESH NETWORK”** done by us under the guidance of **Dr. D. SUSITRA** is submitted in partial fulfillment of the requirements for the award of Bachelor of Engineering degree in Electrical and Electronics Engineering.

DATE: 21.04.2022

PLACE: CHENNAI



SIGNATURE OF THE CANDIDATE

ACKNOWLEDGEMENT

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

We convey our thanks to **Dr. N. M. Nandhita, Dean, School of Electrical and Electronics Engineering** and **Dr. V. Sivachidambaranathan, Head of the Department, Dept. of Electrical and Electronics Engineering** for providing us necessary support and details at the right time during the progressive reviews.

We would like to express our sincere and deep sense of gratitude to our Project Guide **Dr. D. Susitra** for her valuable guidance, suggestions and constant encouragement paved way for the successful completion of our project work.

We wish to express our thanks to all Teaching and Non-teaching staff member of the Department of Electrical and Electronics Engineering who were helpful in many ways for the completion of the project.

ABSTRACT

Real- time measures and display of vitals are integral part of patient's health monitoring with limited resources. Wireless detector networks that communicate in a mesh acquire and transmit similar critical parameters, making a medical terrain effective. With limited resources, both equipment and medical staff, the system must be dependable, and able of dynamic updating on patient status. System must be suitable to sustain at least single contingency. Presenting a new IoT- grounded health monitoring approach in which collected medical detector data is transferred to an analysis module via low- cost, low- power and secure communication links handed by a Zigbee WAN network structure. Mainly focusing on covering ECG, pulse and temperature in rural areas where cellular network content is either absent or doesn't allow data transmission. The main ideal is to reduce the burden of long passages for people living in these areas to visit healthcare installations, while minimizing the communication cost. Testing was performed to measure the reliability of transmitting data over a wide range of distances through high business areas (both human and Wi-Fi). To show that similar systems can give enhanced reliability in a limited resourced environment and have the potential to be deployed in disaster recovery situations.

TABLE OF CONTENTS

CHAPTER NO.	TITLE	PAGENO.
	ABSTRACT	v
	LIST OF FIGURES	viii
	LIST OF TABLES	ix
1	INTRODUCTION	
	1.1 GENERAL INTRODUCTION	1
2	LITERATURE SURVEY	
	2.1 SURVEY	2
	2.2 LIMITATIONS OBSERVED FROM SURVEY	3
3	OBJECTIVE AND SCOPE	
	3.1 OBJECTIVE OF THE PROJECT	4
	3.2 SCOPE OF THE PROJECT	4
	3.3 EXISTING SYSTEM	4
	3.3.1 DRAWBACK	4
	3.4 PROPOSED SYSTEM	5
	3.4.1 ADVANTAGE	5
4	EXPERIMENTAL AND METHODS USED	
	4.1 EMBEDDED SYSTEM	6
	4.1.1 OVERVIEW	6
	4.1.2 FUNCTIONS	6

4.2	BLOCK DIAGRAM	7
4.2.1	HARDWARE REQUIREMENT	8
4.2.2	SOFTWARE REQUIREMENT	8
4.3	HARDWARE COMPONENTS	9
4.3.1	BASIC ARDUINO	9
4.3.2	ZIGBEE	14
4.3.3	TEMPERATURE SENSOR	17
4.3.4	ECG SENSOR	18
4.3.5	PULSE SENSOR	18
4.3.6	NODE MCU	19
4.3.7	COMPLETE HARDWARE	20
4.4	SOFTWARE COMPONENTS	20
4.4.1	ARDUINO SOFTWARE IDE	20
4.4.2	EMBEDDED C	31
5	RESULTS AND PERFORMANCE ANALYSIS	
5.1	TX CODE	33
5.2	RX CODE	34
5.3	RESULTS	35
6	CONCLUSION	
6.1	CONCLUSION	38

LIST OF FIGURES

FIGURE NO.	TITLE	PAGE NO.
4.1	TRANSMITTER NODE	7
4.2	RECEIVER NODE	7
4.3	ARDUINO UNO	9
4.4	DESCRIPTION OF ARDUINO	13
4.5	TYPES OF ARDUINO UNO	13
4.6	ZIIGBEE MODULE	16
4.7	MESH NETWORK	16
4.8	DHT11 SENSOR	17
4.9	ECG SENSOR	18
4.10	PULSE SENSOR	18
4.11	NODE MCU	19
4.12	COMPLETE HARDWARE	20
4.13	PREFERENCE	30
5.1	TX CODE	33
5.2	RX CODE	35
5.3	OUTPUT FROM SERIAL MONITOR COM4	35
5.4	OUTPUT AT VARIOUS TEMPERATURES	36
5.5	GRAPHICAL OUTPUT 1	37
5.6	GRAPHICAL OUTPUT 2	37

LIST OF TABLES

FIGURE NO.	TITLE	PAGE NO.
4.1	ARDUINO DETAILS	10
4.2	COMPARISON OF PROTOCOL	15
4.3	NODE MCU DETAILS	19
4.4	SOFTWARE ICONS	21

CHAPTER 1

INTRODUCTION

1.1 GENERAL INTRODUCTION

In a clinical setting, the diagnosis and treatment of cases involves several natural parameters that need to be measured on a real-time base. Pulse rate, temperature, blood pressure, electrocardiogram (ECG), and blood oxygen level are some of these crucial measures. Wireless detector networks that communicate in a mesh collect and transmit similar critical parameters, making a medical terrain more dependable and effective through their self-healing nature. With limited resources, both equipment and medical staff, the system must be dependable, and able of dynamic updating on patient status. System must be suitable to sustain at least single contingency.

Originally, a primary station acting as a router must be supported by a hot standby station in case of system failure. Secondly, medical staff (doctors and nurses) looking after cases must have real-time contextually sensitive terrain that informs the applicable staff the real-time status update of test/ diagnostics presently being performed of what case, when and where the event is being, and by which medical staff. Further, equipment vital to the health and well being of a patient must contain tone-individual capabilities to notify staff of fault. For illustration, a low battery suggestion on a detector that monitors heart rate can notify staff that the battery must be changed. Testing was performed to measure the reliability of transmitting data over a wide range of distances through high business areas. We show that similar systems can handed enhanced reliability in a limited resourced terrain and have the eventuality to be stationed in disaster recovery situations.

CHAPTER 2

LITERATURE SURVEY

2.1 SURVEY

Nitin Vijaykumar Swami, Vilas Malappa Jarali in 2017 proposed “Performance Dimension of Ray Communication Prototype in Free Space for Water Dam using ATmega328 and Comparison with ZigBee Technology”. Considering increased demands of the RF channels makes major chain in developing new wireless technologies. The old systems like Bluetooth, IR, Wi-Fi, fryer, ZigBee, etc. are unfit to fulfill all the conditions of dispatches links between bias. It has been demonstrated that we must now shift our focus to the underused RF spectrum, i.e. visible light. Laser communication is one of the types of establishing network using visible light with the help of ray.

Himanshu Sharma, Rajesh M in 2017 proposed “Design and Simulation of WSN for Zigbee grounded communication in Multi-Robot System”. Communication plays a vital part in multi robot systems addressing real time operations. Because of high performance of Wireless Local Area Network (WLAN), its use for wireless communication among multi robots has become an effective system. Still, in a system with a great number of robots, it becomes hard for the robots to interchange data at a time because of their limited communication capabilities. In similar case, an ad-hoc robot networking system is more promising.

Deepakfranklin.P, Krishnamoorthi.M in 2018 proposed “Monitoring Multiple Biomedical parameter to Automate Anesthesia Injector using FPGA”. This paper aids anesthetist to monitor level of consciousness of the cases by covering multiple biomedical parameters. Operation of an anesthetic medicine injector machine grounded on multiple biomedical parameters of case through hardware reconfigurable device called field programmable gate array which reduces chances of over dosage through effective monitoring and parallel processing.

Huang-Chen Lee, Kai-Hsiang Ke in 2018 proposed “Design and Evaluation of a Zigbee Wireless Mesh Network System for Monitoring Large-Area IoT Detectors”. Although numerous ways exist to transfer data from the extensively distributed detectors that make up the Internet of Effects (IoT) (e.g., using 3G/ 4G networks or lines), these styles are associated with prohibitively high costs, making them impracticable for real- life operations. Several new wireless technologies have recently been proposed for IoT detectors to provide long-range communication. Among these, Zigbee's long-range performance has been investigated.

Simone Aiassa, Francesca Stradolini, Abuduwaili Tuoheti, Sandro Carrara, Danilo Demarchi in 2019 proposed “Quasi-Digital Biosensor-Interface for a Portable Pen to Cover Anesthetics Delivery”. Monitoring of patient response to the anesthetic medicines is a seductive enhancement for achieving a correct balance of sedation position, adding the chance of success in the right procedure of anesthesia. Currently, there are no marketable tools suitable to offer real-time monitoring of anesthetics; indeed, there's still a lack in seeing technologies suitable to maintain high performances in long term monitoring within a movable miniaturized hardware system. To overcome these limitations, we are then presenting the innovative conception of a movable pen- device suitable to smell anesthetic composites over time.

2.2 LIMITATIONS OBSERVED FROM THE LITERATURE SURVEY

- ▶ Expensive implementation
- ▶ If the direction is not intended over a period of time, the ErrPs generated by discrete occurrences might be employed as a safety mechanism to stop or change the direction of the movement
- ▶ Because the hardware is implemented using FPGA, the cost will be expensive
- ▶ More research will be needed to figure out how to reduce data transmission latency from the nodes
- ▶ Monitoring is restricted

CHAPTER 3

OBJECTIVE AND SCOPE OF THE PRESENT INVESTIGATION

3.1 OBJECTIVE OF THE PROJECT

We mainly focus on monitoring ECG, pulse and temperature in rural areas where cellular network coverage is either absent or does not allow data transmission. The major goal is to lessen the hardship of long visits to healthcare institutions for people living in these locations while lowering communication costs.

3.2 SCOPE OF THE PROJECT

Several experiments have been conducted to evaluate the area covered by the zigbee network and the power consumption of our system. When the Zigbee Gateway is placed outside at a 12 meter altitude, the findings show that the average area covered is roughly 33 km².

3.3 EXISTING SYSTEM

- ▶ Remote clinics
- ▶ Manual diagnosis
- ▶ IoT based health monitoring

3.3.1 DRAWBACK

- ▶ Very costly
- ▶ Very hard to reach remote villages or forest areas
- ▶ No network connectivity for IoT

3.4 PROPOSED SYSTEM

- ▶ Monitoring health parameters through sensors
- ▶ Arduino Uno used as the central node
- ▶ It can be programmed using embedded C program
- ▶ Collected data is send through zigbee for further analysis and for diagnosing the disease
- ▶ Uses ECG and pulse sensor for heart related issues and temperature sensor for fever

3.4.1 ADVANTAGE

- ▶ Anyone laymen can use
- ▶ Very cost effective
- ▶ Zigbee -long range communication, low data rate, less power
- ▶ Can be used in places where there is no network connectivity

CHAPTER 4

EXPERIMENTAL OR MATERIALS AND METHODS USED

4.1 EMBEDDED SYSTEM

4.1.1 OVERVIEW OF EMBEDDED SYSTEM

A special-purpose computing system built to execute one or a couple of dedicated functions, frequently with real-time operation limitations, are understood as an embedded system. It is usually embedded as part of a larger gadget that includes both hardware and mechanical components. A general-purpose computer, such as a personal computer, on the other hand, can do a wide range of functions depending on the programming. Embedded systems have become increasingly significant in today's world, as they manage many of the items we use on a daily basis. Because the embedded system is dedicated to a single mission, design engineers can optimise it to reduce the product's size and cost while also improving its dependability and performance. Embedded systems are sometimes mass-produced to take advantage of economies of scale.

4.1.2 FUNCTIONS OF EMBEDDED SYSTEM

- Monitor the environment: Data from input sensors is read by embedded systems. The data is subsequently processed, and the results are presented to a user or users in some manner.
- Control the environment: Actuator commands are generated and sent by embedded systems.
- Transform the information: The data gathered by embedded systems is transformed in some way, such as data compression and decompression.

4.2 BLOCK DIAGRAM

There are two parts in this project, one is transmitter node and another one is receiver node. Transmitter side consists of Arduino UNO, sensors, zigbee transmitter. Receiver side consists of Node MCU, zigbee receiver.

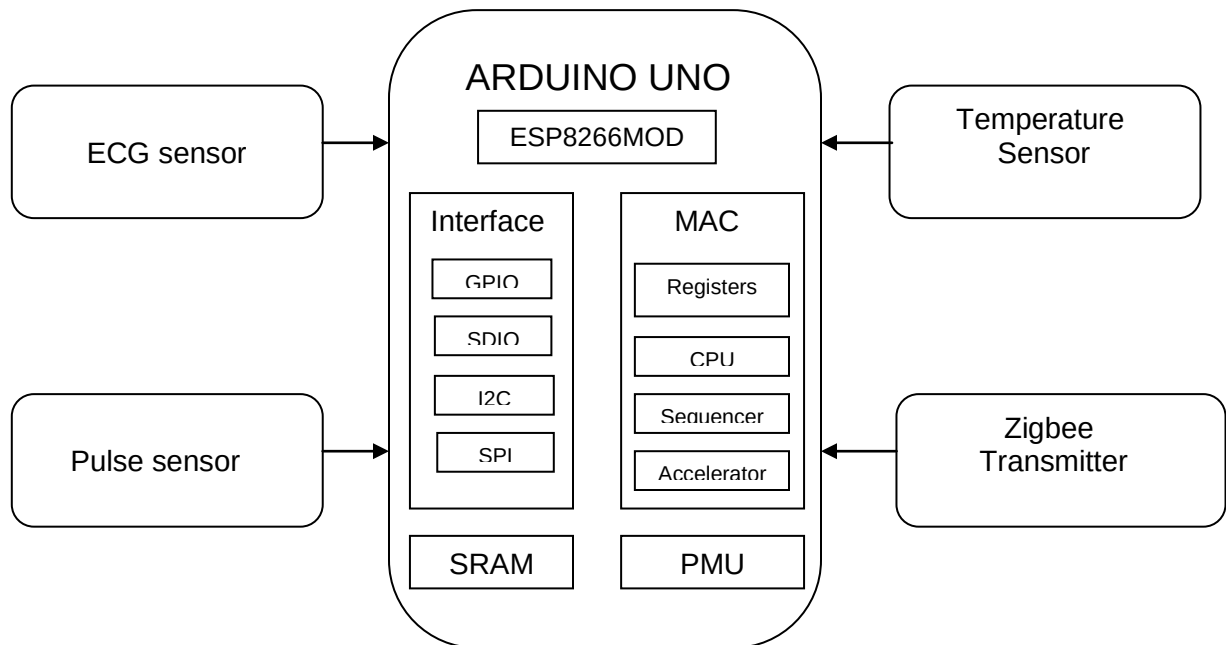


Fig 4.1 Transmitter Node

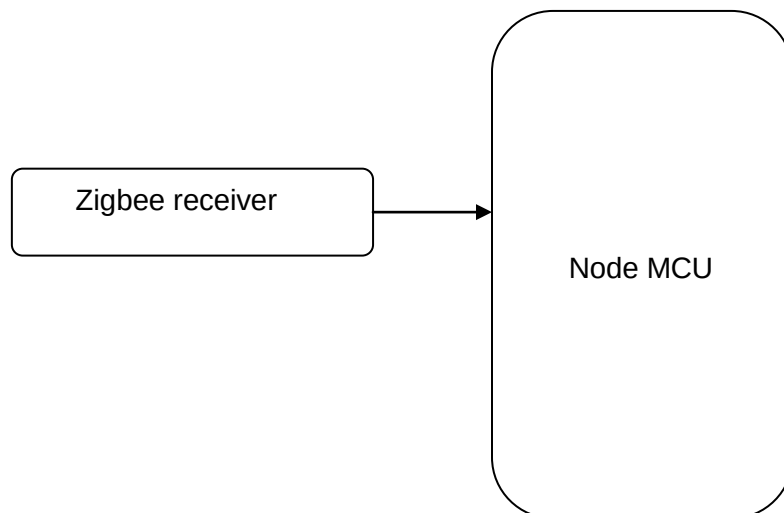


Fig 4.2 Receiver Node

4.2.1 HARDWARE REQUIREMENT

- ▶ Arduino Uno
- ▶ Zigbee module (transmitter, receiver)
- ▶ DHT11 sensor
- ▶ Pulse sensor
- ▶ ECG sensor

MODULE 1

- ▶ In this project we are using sensors that have inputs are Pulse sensors, ECG sensors and temperature sensors.
- ▶ Pulse sensor is for sensing the pulse of human body
- ▶ ECG is for recording electrical signal in your body.
- ▶ DHT11 sensor is used to sense the temperature in our body.

MODULE 2

- ▶ In this project we are using output that has a zigbee module and nodemcu.
- ▶ Zigbee module is used to act as a receiver and sending data from arduino
- ▶ Node MCU is used to take data from zigbee.

4.2.2 SOFTWARE REQUIREMENT

- ▶ Programming platform: Arduino IDE
- ▶ Programming language: Embedded C

MODULE 3

- ▶ The software for Arduino is called Arduino IDE (Integrated Development Environment). It is a notepad-style text editor with a lot of features. It is used to write code, compile it to see if there are any issues, and then upload the code to the Arduino.
- ▶ Microcontroller-based applications are written in the embedded C programming language. Embedded C is a C programming language extension that adds functionality including addressing I/O, fixed-point arithmetic, and multiple-memory addressing. Specific compilers are used in the embedded C language.

4.3 HARDWARE COMPONENTS

Arduino UNO, Zigbee module, DHT11 sensor, pulse sensor, ECG sensor, Node MCU are the hardware components used.

4.3.1 BASIC ARDUINO

The ATmega328P microprocessor is used. It contains 14 digital input/output pins, 6 analog inputs, a 16 MHz ceramic resonator (CSTCE16M0V53-R0), a USB connection, a power connector, an ICSP header, and a reset button. To get started, just connect it to a computer through USB or power it with an AC-to-DC converter or battery.



Fig 4.3 Arduino uno

Table 4.1 Arduino Details

MICROCONTROLLER	ATmega328P
OPERATING VOLTAGE	5V
INPUT VOLTAGE (RECOMMENDED)	7-12V
INPUT VOLTAGE (LIMIT)	6-20V
DIGITAL I/O PINS	14 (6 PWM output)
PWM DIGITAL I/O PINS	6
ANALOG INPUT PINS	6
DC CURRENT PER I/O PIN	20 Ma
DC CURRENT FOR 3.3V PIN	50 mA
FLASH MEMORY	32 KB
SRAM	2 KB (ATmega328P)
EEPROM	1 KB (ATmega328P)
CLOCK SPEED	16 MHz
LED_BUILTIN	13
LENGTH	68.6 mm
WIDTH	53.4 mm
WEIGHT	25 g

POWER

The Arduino Uno board may be powered either by USB or by an external power source. The power source is automatically selected. An AC-to-DC adapter can provide external (non-USB) power. A 2.1mm centre-positive plug may be plugged into the board's power port to connect the adapter. Battery leads can be placed into the POWER connector's GND and Vin pin headers.

To power the board, 6 to 20 volt external power supply can be used. If the 5V pin is provided with less than 7V, it may produce fewer than five volts, causing the board to become unstable. If more than 12V is utilized, the voltage regulator may overheat and kill the board. The suggested range is 7 to 10.

The power pins are as follows:

- ▶ Vin: When utilizing an external power source, the Arduino board's input voltage (as opposed to 5 volts from the USB connection or other regulated power source).
- ▶ 5V: Powered by the DC power connector (7-12V), USB (5V), or the VIN pin on the board (7-12V). Using the 5V or 3.3V pins to supply power bypasses the regulator and may cause harm to your board. It is not recommended.
- ▶ 3V3: A 3.3 volt supply is provided by the on-board regulator.
- ▶ GND: Ground pins.
- ▶ IOREF: This pin on the Arduino board provides the voltage reference with which the microcontroller runs. A properly designed shield can read the IOREF pin voltage and then select the appropriate power source or enable voltage translators on the outputs to work with 5V or 3.3V.

MEMORY

The ATmega328 has 32 KB of memory, 2 KB of SRAM and 1 KB of EEPROM on board is presented. The EEPROM library is a type of memory that retains its contents even after the board is switched off (like a tiny hard drive). You may read and write those bytes with the help of this library.

INPUT AND OUTPUT

Using the `pinMode()`, `digitalWrite()`, and `digitalRead()` routines, each of the Uno's 14 digital pins may be utilised as an input or output. They are powered by 5 volts. Each pin includes a 20-50k internal pull-up resistor (disconnected by default) and can deliver or receive 20mA as a suggested operating state. To avoid irreversible damage to the microcontroller, a limit of 40mA must not be exceeded on any I/O pin.

In addition, some pins have specialized functions:

- ▶ Serial: 0 (RX) and 1 (TX). TTL serial data is received (RX) and sent (TX) using this device. These pins are linked to the ATmega8U2 USB-to-TTL Serial chip's equivalent pins.
- ▶ External Interrupts: 2 and 3. On a low value, a rising or falling edge, or a change in value, these pins can be programmed to generate an interrupt. For further information, `attachInterrupt()` method is used.
- ▶ PWM: The `analogWrite()` method generates an 8-bit PWM signal(3, 5, 6, 9, 10, and 11).
- ▶ SPI: Supports SPI communication (10 (SS), 11 (MOSI), 12 (MISO), 13 (SCK)).
- ▶ LED: 13. It will act like a switch; the LED is turned on when the pin is high and vice versa.

The Uno includes six analog inputs, labelled A0 through A5, each with a resolution of 10 bits (i.e. 1024 different values). They measure from ground to 5 volts by default, but the AREF pin and the `analogReference()` function can be used to adjust the upper end of their range. On the board, there are a few more pins:

- ▶ AREF: The analog inputs have a reference voltage. Used in conjunction with `analogReference()`.
- ▶ Reset: Make LOW to reset the microcontroller. Typically used to add a reset button to shields that are in the way of the board's reset button.

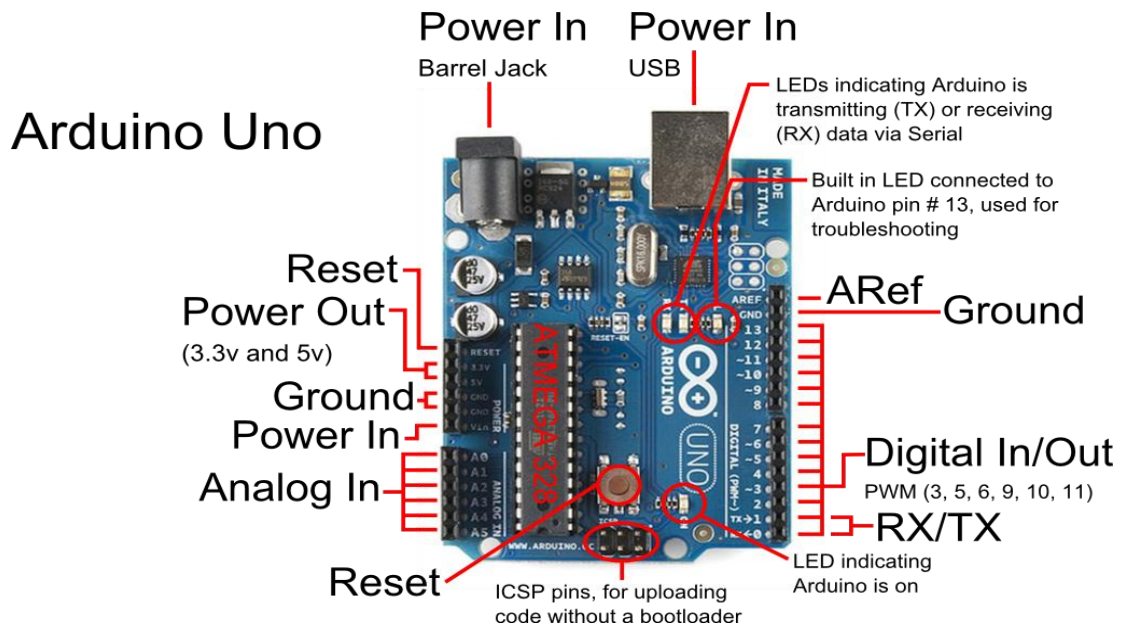


Fig 4.4 Description of Arduino

These boards employ the same microcontroller as the ones above, but in a different packaging. The Lily-pad is designed to be used with conductive thread rather than wire, whereas the Arduino Mini is a stripped-down Arduino with no USB, barrel jack, or power outputs.

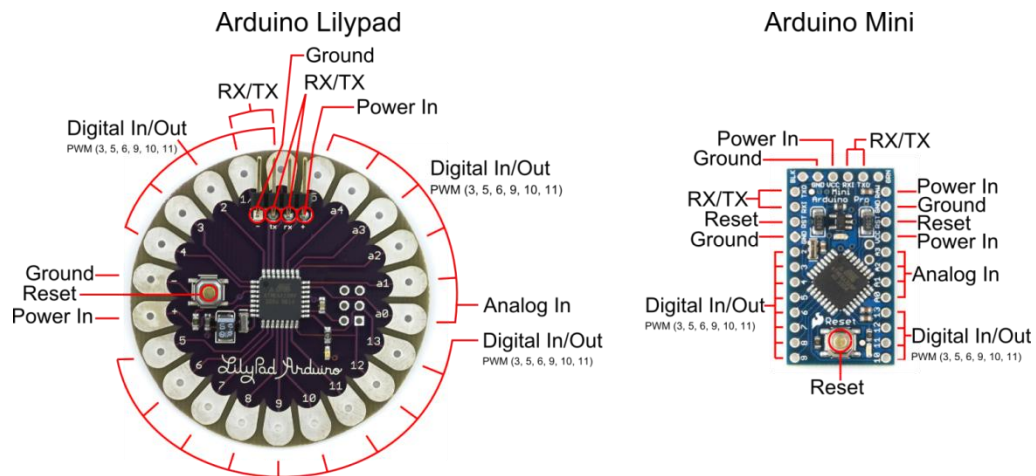


Fig 4.5 Types of Arduino UNO

4.3.2 ZIGBEE

The ZigBee Alliance designed the IEEE 802.15.4 standard in 1998, standardised it in 2003, and amended it in 2006. The protocol's principal benefit is that it may be used to establish wireless personal area networks using low-power digital radios. In the 2.4 GHz frequency spectrum, often known as the ISM band, ZigBee runs across 16 channels. This band is utilised for Industrial, Scientific, and Medical (ISM) purposes and does not need licence, making it available to the general public for free. ZigBee is the only standard-based wireless technology that can meet the particular requirements of low-cost, low-power wireless sensor and control networks in almost any application. ZigBee also integrates the greatest range of devices into simple networks, allowing users unparalleled control over the gadgets they use on a daily basis.

The ZigBee Alliance, which has over 100 members, is a non-profit organisation made up of corporations, colleges, and government organisations from all around the world. The major purpose of the ZigBee Alliance is to develop products that will aid in the development of ZigBee modules for low-power applications. The alliance's members have complete access to the development of ZigBee hardware and documentation. The eleven ZigBee promoters, who play a key role in the standard's development and promotion, provide the most significant contribution.

ZigBee is utilised in a variety of applications that demand long-range communication and long-lasting battery power. ZigBee also provides a number of low-cost modules that aid in the research and development of a variety of future applications. Reliability, low power consumption, and cost-effective development and implementation are common requirements for ZigBee applications.

ZIGBEE PROTOCOL FEATURES

- ▶ Multiple network topologies, including point-to-point, point-to-multipoint, and mesh networks, are supported.
- ▶ Low duty cycle means longer battery life.
- ▶ Low latency
- ▶ Direct Sequence Spread Spectrum (DSSS)
- ▶ Up to 65,000 nodes per network
- ▶ Secure data communications with 128-bit AES encryption
- ▶ Collision avoidance, retries and acknowledgements

OTHER WIRELESS PROTOCOLS

In comparison to other wireless protocols, ZigBee provides benefits such as extended battery life and a limitless network capacity, as illustrated in Table 4.2. Another useful characteristic of the ZigBee standard is its capacity to convey data by passing it along to other ZigBee devices until it reaches its final destination. This is accomplished by deploying ZigBee coordination nodes, which greatly extend the network's transmission range.

Table 4.2 Comparison of protocol

	ZigBee	Wi-Fi	Bluetooth
Standard	802.15.4	802.11b	802.15.1
Application Focus	Monitoring & Control	Web, Email, Video	Cable Replacement
Battery Life (days)	100–1000+	0.5–5	1–7
Network Size	Unlimited (2^{64})	32	7
Max Data Rate (Kb/s)	20–250	11,000+	720
Transmit Range (M)	1–100+	1–100	1–10+
Advantage	Reliability, Power, Cost	Speed, Flexibility	Cost, Convenience

TRANSMITTER AND RECEIVER

Transmitter node will act as a slave node which collects the data from the sensor to transmit to the receiver. Receiver node will act as a Master node in which displays the data in the monitor by receiving it from the slave node

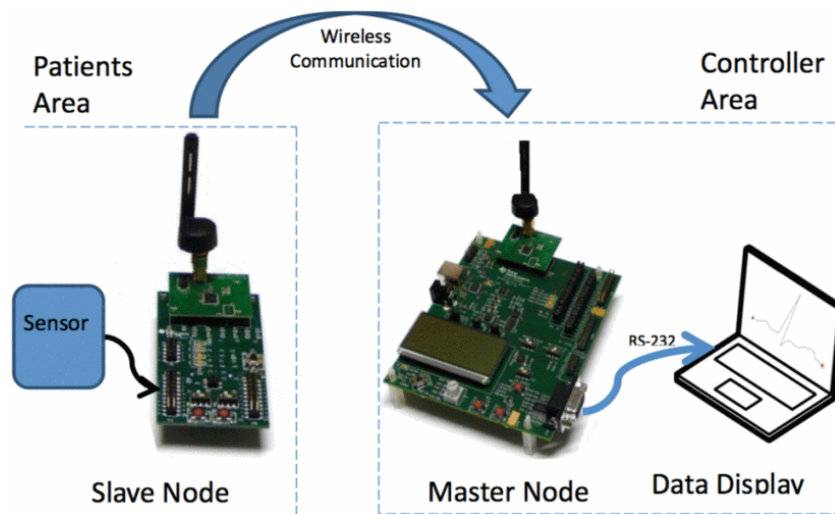


Fig 4.6 Zigbee Module

MESH NETWORKS

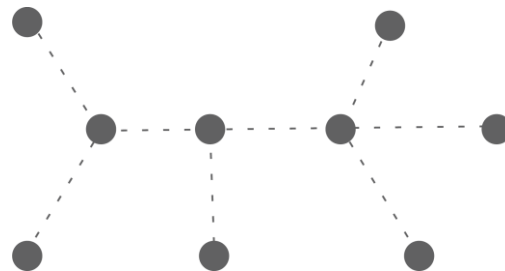


Fig 4.7 Mesh Network

The ability to enable mesh networking is an important feature of the Zigbee protocol. Nodes in a mesh network are coupled with other nodes so that each node may be reached through numerous paths. Through a complex, built-in mesh routing table, connections between nodes are dynamically updated and optimised.

Mesh networks are decentralised in nature, with each node in the network capable of self-discovery. The mesh architecture also allows nodes to adjust routing patterns depending on the changing network structure when nodes leave the network. Mesh architecture and ad-hoc routing features give increased stability in changing situations or single node failure.

ZIGBEE APPLICATIONS

With low-cost, low-power solutions, Zigbee enables widespread deployment of wireless networks. It can run for years on cheap batteries, making it ideal for a variety of monitoring and control applications. Smart energy/smart grid, AMR (Automatic Meter Reading), lighting controls, building automation systems, tank monitoring, HVAC management, medical devices, and fleet applications are just a few of the areas where Zigbee technology is advancing rapidly.

4.3.3 TEMPERATURE SENSOR

The DHT11 sensor may be used to monitor both temperature and humidity levels in the air. It has a temperature range of 0 to 50 degrees Celsius, with a ± 2 degree precision.

Temperature sensors are utilized in a wide range of applications, including food processing, HVAC environmental management, medical equipment, chemical handling, and under-the-hood monitoring in automobiles.

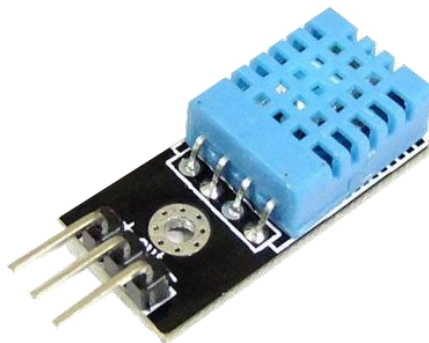


Fig 4.8 DHT11 sensor

4.3.4 ECG SENSOR

An ECG can detect the following:

- ▶ Arrhythmia occurs when heartbeats that is too slow, too fast, or irregular.
- ▶ Coronary heart disease occurs when a build-up of fatty substances blocks or interrupts the heart's blood flow.
- ▶ A heart attack occurs when the heart's blood supply is unexpectedly cut off.

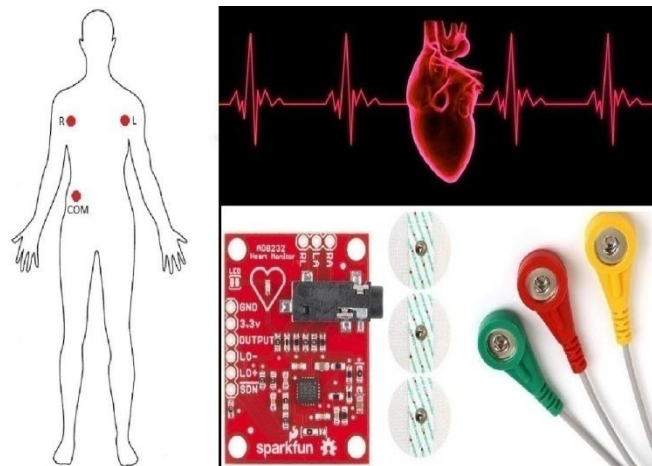


Fig 4.9 ECG sensor

4.3.5 PULSE SENSOR

Pulse waves, or changes in the volume of a blood artery caused by the heart pumping blood, are measured by an optical heart rate sensor. An optical sensor and a green LED are used to detect pulse waves by monitoring volume changes.



Fig 4.10 Pulse sensor

4.3.6 NODE MCU

- ▶ Open source prototype board designs are available for Node MCU, which is an open source firmware. The term "Node MCU" is a combination of the words "node" and "MCU" (micro-controller unit). The name "Node MCU" refers to the firmware rather than the development kits that go with it.
- ▶ The Node MCU IoT platform is an open source IoT platform that contains firmware for the ESP8266 Wi-Fi SoC.

Table 4.3 Node MCU Details

DEVELOPER	ESP8266 Open source Community
TYPE	Single-board microcontroller
OPERATING SYSTEM	XTOS
CPU	ESP8266(LX106)
MEMORY	128kBytes
STORAGE	4MBytes
POWER	USB

PIN CONFIGURATION

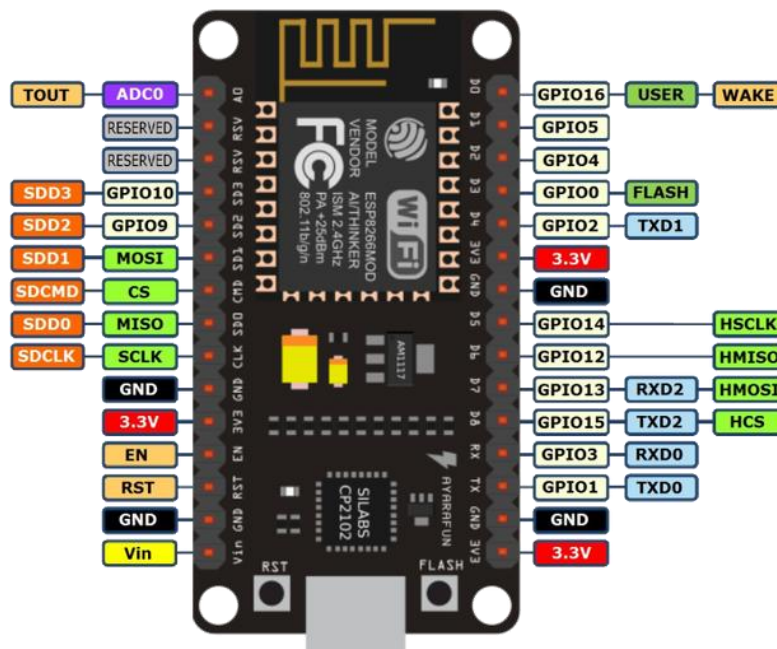


Fig 4.11 Node MCU

4.3.7 COMPLETE HARDWARE

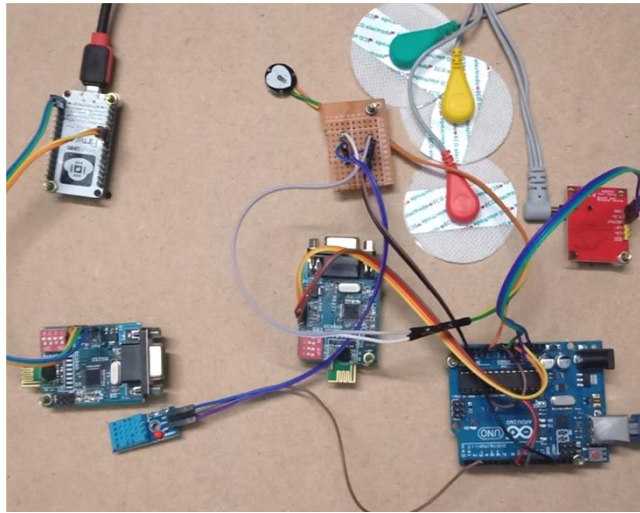


Fig 4.12 Hardware

4.4 SOFTWARE COMPONENTS

To execute and get the results by using an software called Arduino software (IDE) which is also known as Arduino Integrated Development Environment. Embedded C is a programming language which is used to carry forward the work.

4.4.1 ARDUINO SOFTWARE (IDE)

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

A text editor for writing code, a message area, a text terminal, a toolbar with buttons for basic operations, and a series of menus are all included in the Arduino Integrated Development Environment. It communicates with the Arduino and Genuino devices by connecting to them and uploading code.

Writing sketches

Programs are created with the Arduino Software (IDE) using sketches. These drawings are stored with the [.ino] file extension and written with a text editor. Cutting, pasting, replacing and searching text are present in the editor. While storing and exporting, the message section shows errors and gives feedback. The Arduino Software (IDE) outputs text to the console, which includes detailed error messages and other information. In the window's bottom right corner, the configured board and serial port are displayed. Can validate and upload programs, generate, save, and save drawings, and open the serial monitor using the toolbar buttons.

NB: NB: Previous versions of the Arduino Software (IDE) used the [.pde] extension to store sketches. Version 1.0 can open these files, but you'll be requested to save the drawing with the [.ino] suffix when you save it.

Table 4.4 SOFTWARE ICONS

ICONS	DESCRIPTION
	<u>Verify</u> Checks your code for errors compiling it.
	<u>Upload</u> Compiles and uploads your code to the specified board. For more information, see the section on uploading below. Note: If you're using an external programmer with your board, you may use this icon while holding down the "shift" key on your computer. "Upload using Programmer" will be the new text.

	<u>New</u> Creates a new sketch.
	Open Displays a menu with all of your sketchbook's sketches. When you click one, it will open in the current window, overwriting the content. Note that this menu does not scroll owing to a Java problem; if you need to access a drawing that is farther down the list, use the File Sketchbook option instead.
	<u>Save</u> Saves your sketch.
	<u>Serial Monitor</u> Opens the serial monitor to display the output.

The five menus File, Edit, Sketch, Tools, and Help provide more commands. The menus are context sensitive, which means they only show things that are relevant to the job being done at the time.

File

- ▶ New
Creates a new instance of the editor, complete with the very minimal structure of a sketch.
- ▶ Open
Allows you to browse through your computer's files and folders to find a sketch file.
- ▶ Open Recent
Gives you a quick list of the most current drawings, all of which are ready to be opened.

‣ Sketchbook

The current drawings are displayed within the sketchbook folder hierarchy; selecting any name opens the matching sketch in a new editor instance.

‣ Examples

This menu option displays any example offered by the Arduino Software (IDE) or library. All of the examples are organized in a tree, making it simple to find what you're looking for by topic or library.

‣ Close

Closes the Arduino Software instance from which it was selected.

‣ Save

The drawing is saved with the current name. If the file hasn't been given a name yet, one will be suggested in the "Save as..." box.

‣ Save as...

Allows you to save the current sketch under a new name.

‣ Page Setup

It displays the printing Page Setup window.

‣ Print

Sends the current sketch to the printer according to the parameters provided in Page Setup.

‣ Preferences

Opens the Preferences window, where several IDE settings, such as the language of the IDE interface, can be adjusted.

‣ Quit

All IDE windows are closed. When you choose Quit, the same drawings that were open before will be automatically reopened the next time you start the IDE.

Edit

‣ Undo/Redo

Goes back in time to one or more steps you took while editing; after you've gone back, you may use Redo to move ahead.

- ▶ Cut
The chosen text is removed from the editor and pasted into the clipboard.
- ▶ Copy
Duplicates the editor's chosen text and pastes it onto the clipboard.
- ▶ Copy for Forum
Copies your sketch's code to the clipboard in a format suited for publishing to the forum, replete with comments.
- ▶ Copy as HTML
Copies your sketch's code to the clipboard as HTML, which may be pasted into web sites.
- ▶ Paste
The contents of the clipboard are pasted into the editor at the cursor point.
- ▶ Select All
Selects and highlights all of the editor's material.
- ▶ Comment/Uncomment
The / comment marker is added or removed at the beginning of each specified line.
- ▶ Increase/Decrease Indent
Adds or removes a space at the start of each chosen line, advancing the text one space to the right or removing a space at the start.
- ▶ Find
Opens the Find and Replace box, where you may search for text inside the current drawing using a variety of criteria.
- ▶ Find Next
The next occurrence of the string selected as the search item in the Find window, if any, is highlighted relative to the cursor point.
- ▶ Find Previous
The prior occurrence of the string supplied as the search item in the Find window, if any, is highlighted relative to the cursor point.

Sketch

- ▶ Verify/Compile

Checks for mistakes when generating your sketch and displays memory use for code and variables in the terminal window.

- ▶ Upload

Compiles the binary file and loads it onto the specified board through the defined Port.

- ▶ Upload Using Programmer

This will overwrite the board's boot loader; you'll need to restore it with Tools > Burn Boot loader before you can upload to the USB serial port again. It does, however, allow you to use the whole Flash memory for your sketch. Please keep in mind that this order will not cause the fuses to blow. To do so, go to Tools -> Burn Boot Loader and run the command.

- ▶ Export Compiled Binary

Saves a.hex file, which may be saved as an archive or delivered to the board with other tools.

- ▶ Show Sketch Folder

Opens the current sketch folder.

- ▶ Include Library

Inserts #include statements at the start of your code to add a library to your drawing. You may also open the Library Manager from this menu item and import new libraries from.zip files.

- ▶ Add File...

This command adds a source file to the drawing (it will be copied from its current location). In the sketch window, the new file displays in a new tab. The tab menu, which can be accessed by clicking on the little triangle icon below the serial monitor on the right side of the toolbar, may be used to delete files from the sketch.

Tools

- ▶ Auto Format

This indents your code such that the opening and closing curly braces line up, and the statements within curly braces are indented further.

- ▶ Archive Sketch

In.zip format, save a copy of the current drawing. The archive is saved alongside the drawing in the same location.

- ▶ Fix Encoding & Reload

Fixes any inconsistencies between the editor's char map encoding and that of other operating systems.

- ▶ Serial Monitor

The serial monitor window is opened, and data is exchanged with any connected board on the presently specified Port. If the board supports it, this normally resets the board. Override the serial ports opening with a reset.

- ▶ Board

Select the board that you're using. The descriptions of the various boards may be found below.

- ▶ Port

This menu lists all of your machine's serial devices, including real and virtual. When you access the top-level tools menu, it should automatically refresh.

- ▶ Programmer

When programming a board or chip with a hardware programmer rather than the inbuilt USB-serial connection. You won't normally require this, but you will if you're burning a boot loader to a new microcontroller.

- ▶ Burn Boot loader

The entries in this menu allow you to burn a boot loader onto an Arduino board's microcontroller. This is not necessary for usual Arduino or Genuino board operation, but it will come in handy if you buy a new AT mega microcontroller (without a boot loader). This instruction also activated the appropriate fuses.

Help

You may discover a variety of documentation that come with the Arduino Software here (IDE). Without an internet connection, you can view Getting Started, Reference, this IDE tutorial, and other materials locally. The documents are a local duplicate of the online versions, and they may include a link to our website.

► Find in Reference

This is the Help menu's single interactive feature: it picks the appropriate page from the local copy of the Reference for the function or command beneath the cursor.

Sketchbook

The Arduino Software (IDE) is based on the idea of a sketchbook, which is a standard location for storing your applications (or sketches). The Open button on the toolbar or the File > Sketchbook menu may be used to open the sketches in your sketchbook. The Arduino software will generate a directory for your sketchbook the first time you run it. With the Preferences dialog, you can see or modify the location of the sketchbook.

Files are stored with a .ino extension starting with version 1.0. The .pde extension was used in previous versions. In version 1.0 and later, you may still open .pde files; however, the program will automatically rename the extension to .ino.

Tabs, Multiple Files, and Compilation

Allows you to organize drawings from several files (each of which appears in its own tab). These can be plain Arduino code files (no extension shown), C files (.c extension), or C++ files (.cpp), or header files (.h).

Uploading

The relevant things from the Tools > Board and Tools > Port menus before uploading your drawing must be picked. The boards are discussed in the following sections. It's most likely COM1 or COM2 (for a serial board) or COM4, COM5, COM7, or above (for a USB board) on Windows; to find out, check for USB serial device in the Windows Device Manager's ports area. It should be /dev/ttyACMx, /dev/ttyUSBx, or something similar on Linux. Press the upload button in the toolbar or pick the Upload item from the Sketch menu once you've selected the right serial port and board. Current Arduino boards will immediately reboot and start uploading. You'll need to hit the reset button on the board right before commencing the upload for earlier boards (pre-Diecimila) that don't have auto-reset. As the sketch is uploaded, the RX and TX LEDs on most boards will blink. When the upload is complete, the Arduino Software (IDE) will display a message or an error.

When you upload a sketch, you're utilizing the Arduino boot loader, which is a little software put onto your board's microcontroller. It enables you to upload code without the need for any further hardware. When the board resets, the boot loader is active for a few seconds before starting whichever sketch was most recently uploaded to the microcontroller. When the boot loader starts, it will blink the on-board (pin 13) LED (i.e. when the board resets).

Libraries

Libraries add further functionality to drawings, such as the ability to interface with hardware or manipulate data. Select a library from the Drawing > Import Library menu to utilize it in a sketch. This will build the library with your drawing and add one or more `#include` statements to the top of the sketch. Libraries take up more space on the board than your sketch since they are submitted alongside it. Remove the `#include` statements from the top of your code if a sketch no longer requires a library.

The Arduino software comes with a few libraries. Others are available for download from a variety of places, including the Library Manager. You may import a library from a zip file and use it in an open sketch starting with version 1.0.5 of the IDE. To install a third-party library, follow these instructions.

Third-Party Hardware

Support for third-party hardware may be added to your sketchbook directory's hardware directory. Board definitions (which display in the board menu), core libraries, boot loaders, and programmer definitions are some of the platforms that may be installed. Create the hardware directory first, and then unzip the third-party platform into its own subdirectory. (If you use "arduino" as the sub-directory name, the built-in Arduino platform will be overridden.) Simply remove the program's directory to uninstall it. The Arduino IDE 1.5 3rd party Hardware standards have more information on developing packages for third-party hardware.

Serial Monitor

The serial data transmitted from the Arduino or Genuino board is shown (USB or serial board). Enter text and hit enter or click the "send" button to transmit data to the board. Select the baud rate that corresponds to the rate supplied to Serial from the drop-down menu. Start with your drawing. When you connect with the serial monitor on Windows, Mac, or Linux, the Arduino or Genuino board will reset.

Preferences

In the preferences dialog, you may set some options (found under the Arduino menu on the Mac, or File on Windows and Linux). The remainder is in the preferences file, whose location is indicated in the preferences dialog.

Language Support

The Arduino Software (IDE) has been translated into 30+ languages since version 1.0.1. The IDE starts up in the language that your operating system has picked by default. (Note: on Windows and potentially Linux, the locale option, which governs currency and date formats, determines this, not the language in which the operating system is presented).

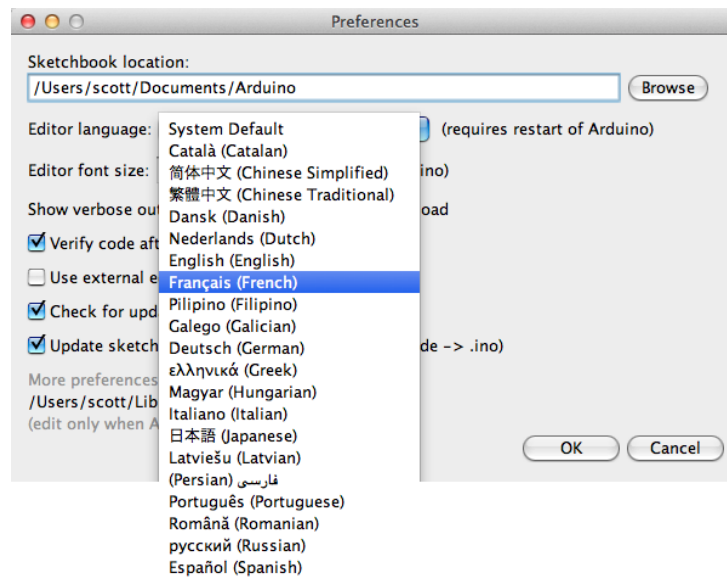


Fig 4.13 Preference

If you want to change the language manually, open the Preferences box in the Arduino (IDE). A dropdown choice of presently supported languages appears next to the Editor Language. Choose your favorite language from the menu, and then restart the software to utilize it. The Arduino Software (IDE) will default to English if your operating system language is not supported.

Select System Default from the Editor Language drop-down to revert the software to its default option of picking its language depending on your operating system. When you restart the Arduino Software, this adjustment will take effect (IDE). Similarly, you must restart the Arduino Software (IDE) after changing your operating system's settings to update it to the new default language.

Boards

The board selection has two effects: it changes the parameters used to generate and upload drawings (such CPU speed and baud rate), and also the file and circuit settings being used by burn boot loader function. Some of the board definitions change solely in the latter, so double-check before burning the boot loader, even if you've been uploading successfully with a certain option. A comparison chart of the different boards may be found [here](#).

4.4.2 EMBEDDED C

Embedded C is one of the most popular and widely used programming languages for embedded systems development. There are several popular programming languages for designing embedded systems, such as Assembly, BASIC, C++, Python, and others, but Embedded C remains popular owing to its efficiency, short development time, and portability.

The Embedded C Programming Language is an extension of the C Programming Language that is frequently used in the creation of embedded systems. The Embedded C Programming Language follows the same syntax and semantics as the C Programming Language, including the main function, data type declarations, variable definitions, loops, functions, and statements, among other things. I/O Hardware Addressing, fixed point arithmetic operations, and accessing address spaces are all extensions in Embedded C from the conventional C Programming Language.

DIFFERENCE BETWEEN C AND EMBEDDED C

- ▶ Apart from a few extensions and the operating environment, there isn't much difference between C and Embedded C. Both C and Embedded C are ISO Standards with almost identical syntax, data types, functions, and other features.

- ▶ Embedded C is an extension to the Standard C Programming Language that adds features like Addressing I/O, multiple memory addressing, and fixed-point arithmetic, among others.
- ▶ The C Programming Language is typically used to create desktop applications, whereas Embedded C is used to create Microcontroller-based applications.

BENEFITS OF EMBEDDED C

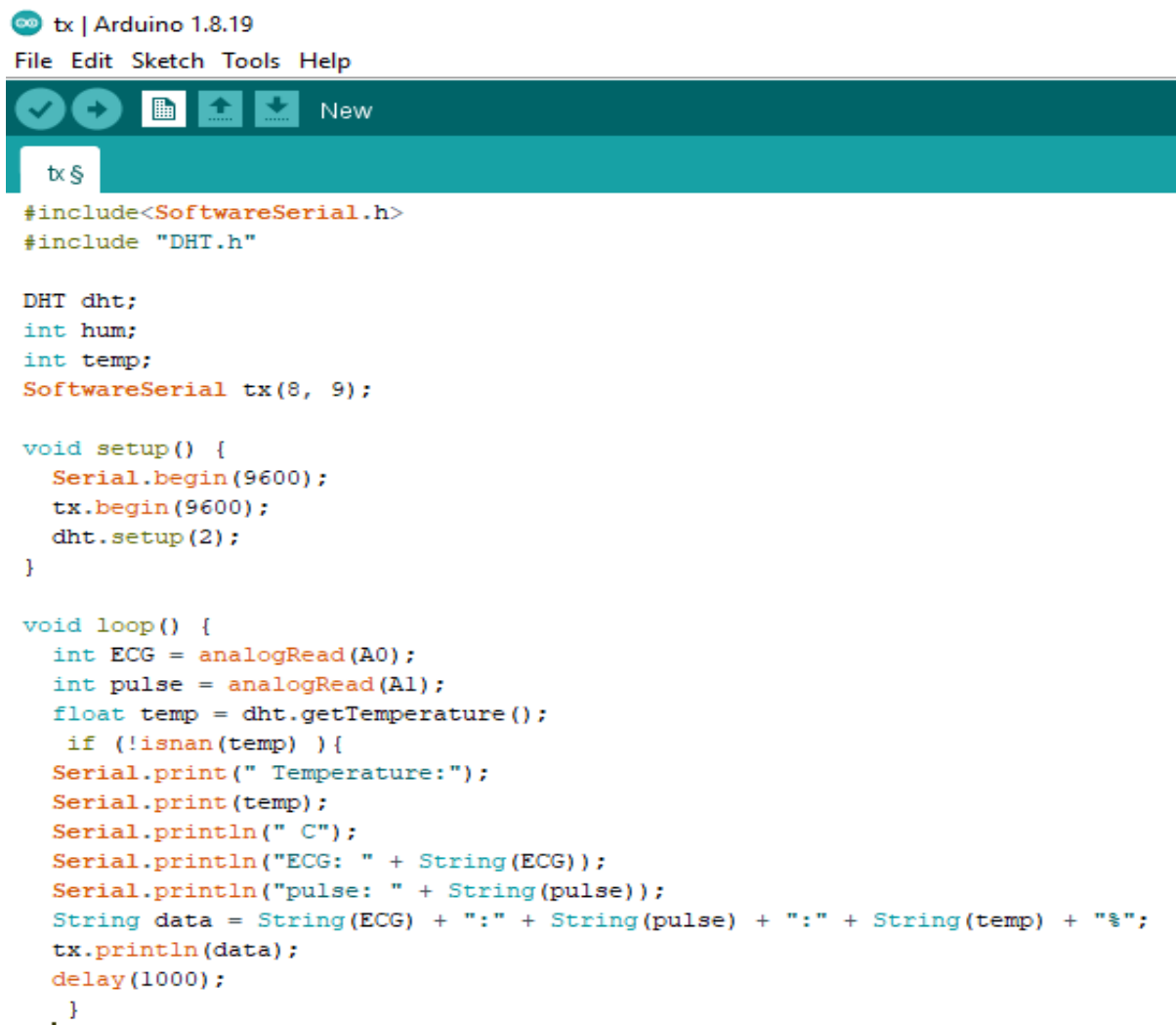
- ▶ It's a lot easier to develop code in C.
- ▶ When compared to Assembly, it takes less time.
- ▶ Code maintenance (modifications and upgrades) is very simple.
- ▶ To decrease the complexity of the main code, use library functions.
- ▶ With very little changes, the code may be simply ported to various architectures.

CHAPTER 5

RESULTS AND PERFORMANCE ANALYSIS

5.1 TX CODE

To verify the result we are getting on the receiver side, we can run the output on the Transmitter side (TX) to double check the result.



```
tx | Arduino 1.8.19
File Edit Sketch Tools Help

tx$
#include<SoftwareSerial.h>
#include "DHT.h"

DHT dht;
int hum;
int temp;
SoftwareSerial tx(8, 9);

void setup() {
  Serial.begin(9600);
  tx.begin(9600);
  dht.setup(2);
}

void loop() {
  int ECG = analogRead(A0);
  int pulse = analogRead(A1);
  float temp = dht.getTemperature();
  if (!isnan(temp) ){
    Serial.print(" Temperature:");
    Serial.print(temp);
    Serial.println(" C");
    Serial.println("ECG: " + String(ECG));
    Serial.println("pulse: " + String(pulse));
    String data = String(ECG) + ":" + String(pulse) + ":" + String(temp) + "%";
    tx.println(data);
    delay(1000);
  }
}
```

Fig 5.1 TX code

5.2 RX CODE

The actual output will be shown on the receiver side (RX). The Serial Monitor has to be clicked to print the output. And for showing the graphs of ECG, pulse Serial Plotter can be used.



```
rx | Arduino 1.8.19
File Edit Sketch Tools Help

#include<SoftwareSerial.h>
SoftwareSerial tx(D6,D7);

String getStringPartByNr(String data, char separator, int index)
{
    // splitting a string and return the part nr index
    // split by separator
    int stringData = 0;        //variable to count data part nr
    String dataPart = "";      //variable to hold the return text

    for(int i = 0; i<data.length(); i++) {    //Walk through the text one letter at a time
        if(data[i]==separator) {
            //Count the number of times separator character appears in the text
            stringData++;

        }else if(stringData==index) {
            //get the text when separator is the right one
            dataPart.concat(data[i]);

        }else if(stringData>index) {
            //return text and stop if the next separator appears - to save CPU-time
            return dataPart;
            break;
        }
    }
    //return text if this is the last part
    return dataPart;
}
```

```
data, char separator, int index)

turn the part nr index

//variable to count data part nr
//variable to hold the return text

); i++) {    //Walk through the text one letter at a time

times separator character appears in the text

{
separator is the right one
};

{
```



35

Zigbee Transmitter module will collect and send all the data to receiver side and thus connected to the Node MCU which in turn connected to the display monitor (laptop). We can select COM4 under serial monitor option while being in RX code.

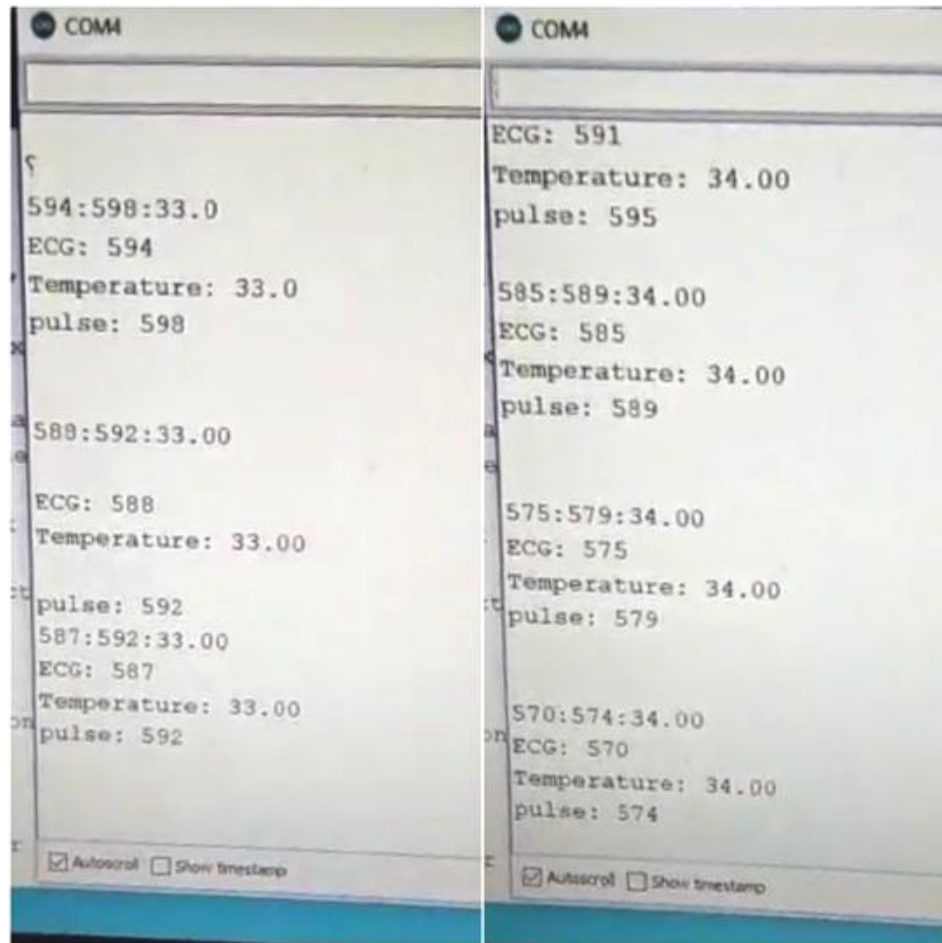


Fig 5.4 COM4 output at different temperatures

As we can see, the three values are kept changing and it is shown in the monitor. The values are analog values as we get analog output from the sensor. We can also get the graph result by selecting serial plotter in the tools option.

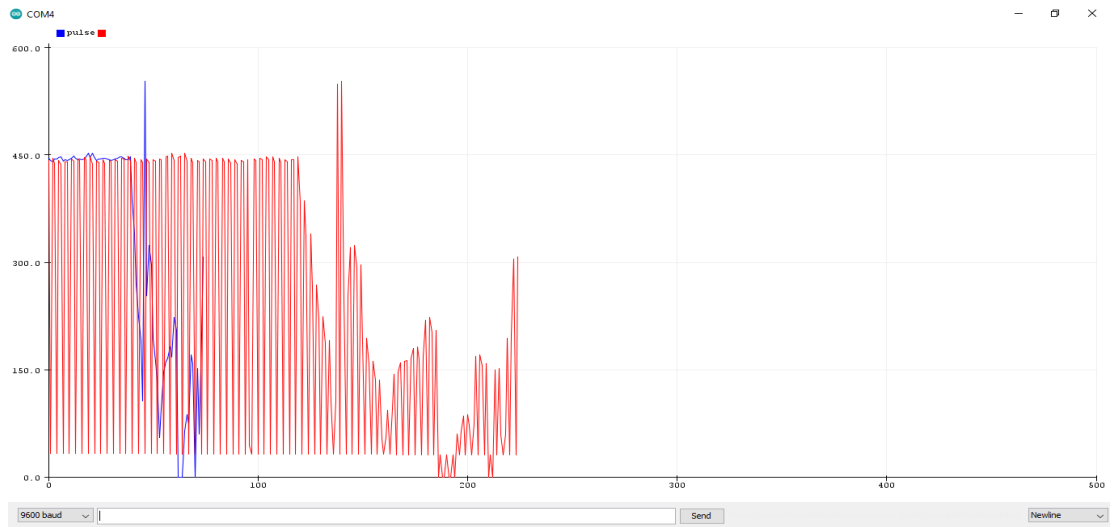


Fig 5.5 Graphical output 1

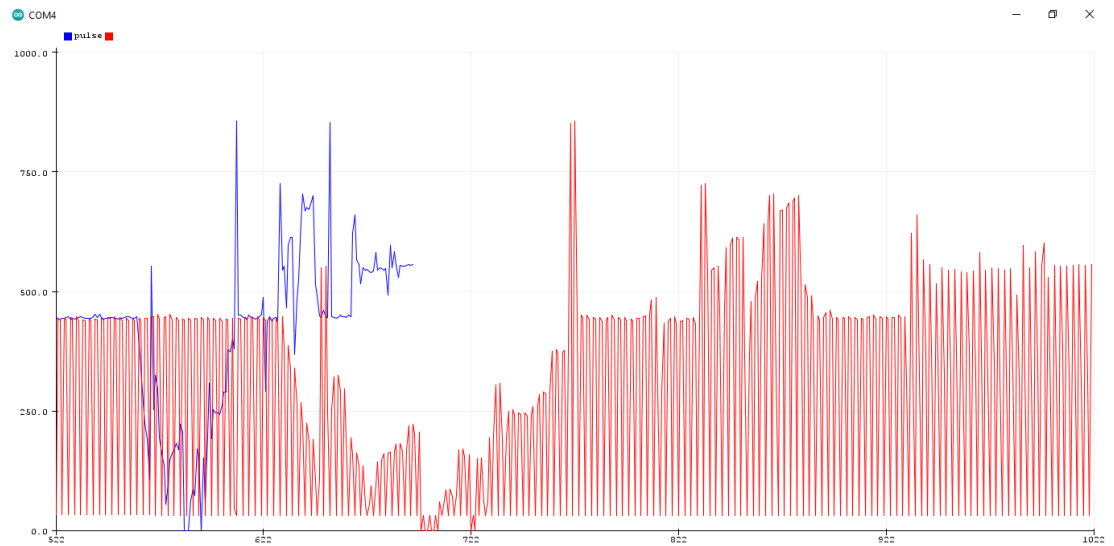


Fig 5.6 Graphical output 2

CHAPTER 6

CONCLUSION

6.1 CONCLUSION

Many components of the Medical Zigbee Network have been improved to satisfy the needs of medical staff. Although the Medical ZigBee Network was created with simple sensors, there's room for development in many various aspects of the sensor. ZigBee module is employed to transmit information between the sign monitoring devices and therefore the healthcare gateway. The sign monitoring devices include ECG, SPO2, vital sign, glucose, and blood heat. The info is then relayed to a healthcare service information platform. As Zigbee are often achieved by low cost implementation of low power devices with low data, it's well ok topology in comparison to devices like wifi, blue-tooth. The appliance of Zigbee in medical applications reduces the workload of hospitals, and therefore the doctors remotely can monitor patients through the network, and supply, in real time, a diagnosis or an advice to their patients.

REFERENCES

- [1] Gartner “21 Billion IoT Devices to Invade By 2020”, 2015. [Online]
Available: <http://www.informationweek.com/mobile/mobile-devices/gartner-21-billion-iot-devices-to-invade-by-2020/d/d-id/1323081>
- [2] Research and Markets “Global Forecast to 2021”, 2016. [Online] Available:
http://www.researchandmarkets.com/research/gsixb5/internet_of_things
- [3] Active Telecoms “Chinese manufacturers tipped to spend \$127b annually on IoT”, 2016. [Online] Available: <http://activetelecoms.com/news/internet-of-things/chinese-manufacturers-tipped-to-spend-127b-annually-on-iot>
- [4] Xiaodong Song and Xiang Wang, “Extended AODV Routing Method Based on Distributed Minimum Transmission (DMT) for WSN,” Intl. J. Electronics &Comms, vol.69, no.1, pp.371-381, 2015.
- [5] Degan Zhang, Guang Li, and Ke Zheng, “An energy-balanced routing method based on forward-aware factor for Wireless Sensor Network,” IEEE Trans. Ind. Infor., vol. 10, no. 1, pp. 766-773, 2014.
- [6] Si Liu and Ting Zhang. “Novel Unequal Clustering Routing Protocol Considering Energy Balancing Based on Network Partition & Distance for Mobile Education,” J. Net. & Comp. Appls., vol. 88, no. 15, pp. 1-9, 2017.
- [7] K. Navya and M. Murthy, "A zigbee based patient health monitoring system", Int. Journal of Engineering Research and Applications, vol. 3, no. 5, 2013.
- [8] ZigBee Alliance, "Zigbee smart energy profile specification", Zigbee Doc. 075356r16ZB rev 16, 2011.

[9] <https://store-usa.arduino.cc/products/arduino-uno-rev3/>

[10] <https://www.electronicshub.org/basics-of-embedded-c-program/>