

Development of Smart Healthcare Monitoring System in IoT Environment

Submitted in Partial fulfillment of the Requirements for the Award of Bachelor of
Engineering in Electronics and Communication.

By Alan B George (38130008) AND M. Z. Arshak Mohideen (38130015).



DEPARTMENT OF ELECTRONICS AND COMMUNICATION 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



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 ELECTRONICS AND COMMUNICATION

ENGINEERING BONAFIDE CERTIFICATE

This is to certify that this Project Report is the bonafide work of Alan B George (38130008) and M. Z. Arshak Mohideen (38130015) who carried out the project entitled **Development of Smart Healthcare Monitoring System in IoT Environment**

under my / our supervision from December 2021 to May 2022

Internal Guide

External Guide (if Applicable)

Head of the Department

(Name in Capital letters with Seal & Signature)

Submitted for Viva voce Examination held on 02.05.2022

Internal Examiner

External Examiner

DECLARATION

We ALAN B GEORGE (3813008) and M. Z. ARSHAK MOHIDEEN (38130015) hereby declare that the Project Report entitled **Development of Smart Healthcare Monitoring System in IoT Environment** done by us under the guidance of Dr. T. VINO is submitted in partial fulfillment of the requirements for the award of Bachelor of Engineering in Electronics and Communication.

DATE: 02.05.2022

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. I am grateful to them. We convey our thanks to Dr. N.M. Nandita, Dean,

School of Electrical and Electronics Engineering and Dr. T. Ravi, Head of the Department, Dept. of Electronics and Communication 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. T. VINO for his 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 members of the Department of Electronics and Communication Engineering who were helpful in many ways for the completion of the project.

ABSTRACT

- The main objective for the Li-Fi is flexible data communication in the hospital environment systems
- A VLC system for indoor application is proposed. In this work, we focus on reducing cost, and for that, we had to make appropriate selection of system's components, e.g. modulation, coding, filtering.
- Our objective was to achieve acceptable bit error rate (BER) performance for indoor use, with a low cost system.
- The Microcontroller sends data from Sensor to the another microcontroller which is interfaced with Personal Computer (PC) by using Li-fi communication as wireless medium.
- For the high speed of the data communication between doctors and patients
- This type of communication will also reduce the risks off radiation hazards and can be used almost anywhere even at places where electronic devices are banned due to the fear of radiation.

Table of contents

Title	Page no
• Chapter 1:- Introduction	8
• Chapter 2:- literature survey	10
• ECG, Heartrate and Body temperature	12
• Chapter 3:- Existing system	16
• Chapter 4:- Hardware design	20
• IOT	41
• Chapter 5:- Software design	45
• Chapter 6:- Result and Conclusion	51
• References	53

List of figures

Title	Page no
• ECG graph	12
• Biological clock in humans	15
• Block diagram	18
• View of temperature sensor	35
• Heart beat sensor	36

CHAPTER 1

INTRODUCTION

1.1. PROJECT OBJECTIVE

In this chapter introduction of the **PATIENT MONITORING SYSTEM TO REMOTE DOCTORS USING GSM TECHNOLOGY** are discussed. It gives overall view of the project design and the related literature and the environment to be considered. Chapter wise organization of the thesis and the appendices is given at the end of this chapter. At first we discuss the main processing done using 8051 microcontroller is and then what is the process that can be automated which is within the scope of the work. Then we discuss the implementation aspects.

1.2. OVERVIEW

In case of emergency and dangerous situations we have to alert the doctor immediately. For this we are using a gsm based network for doctor to patient communication in the hospital and even to communicate and indicate the status of the patient through SMS. This way of communication is actually done with Zigbee network topology and with the GSM network. Each patient will be given this module and with the help of this module the patient health condition is monitored and if there is any change in the condition of the health then it immediately sends that changed data through Zigbee to the local system where the main module is connected to the computer to maintain the status of the patient.

The heart beat is monitored with the pulse rate of the body. The high intensity light sensor senses the expansion and contraction of the heart with the help of the nerves. That beam will transmit the signal to the receiver and the minute change in the pulse is noticed as the heart beat. If there is any change in the pulses then it is noticed as the change in the heart and then the controller will get a disturbed pulse count which indicates the fault or malfunction of the heart. The controller is fixed for a no. of pulses initially. If there is any change in the any of the pulse count then it considers as a malfunction of the heart and then it transmits the pulse count with the patients ID to the doctor in the hospital and at the same to it sends a sms to a fixed number in the

microcontroller. This is convenient process to monitor the patients health conditions form any of the distance we present. Since we are using both the networks like Zigbee and GSM this makes the user to communicate for internal system and as well as to the longer distances.

1.3. AIM OF THE PROJECT

The main processes involved in this type of control system are to monitor the patient's health status. Zigbee is a wireless connection network that is used to connect different devices at a frequency of 2.4GHz. For medical applications also this Zigbee is widely used. The Zigbee can communicate with the devices of about 1km. The other network is GSM network. This can be operated from any distance to any point of control. The communication is done with the help of local network support. This can get communicated to any part of the world which the network of the local system is applicable. Here we are using for the hospital communication for monitoring the patient.

1.4. LITERATURE SURVEY

The technical brilliance and development in different fields has led to a drastic in our lives, one among them is embedded systems. The application of these devices is to monitor the patient health status. Zigbee is a wireless connection network that is used to connect different devices at a frequency of 2.4GHz. For medical applications also this Zigbee is widely used. The Zigbee can communicate with the devices of about 75 m. The other network is GSM network. This can be operated from any distance to any point of control. The communication is done with the help of local network support. This can get communicated to any part of the world which the network of the local system is applicable. Here we are using for the hospital communication for monitoring the patient.

CHAPTER 2:

LITERATURE SURVEY:

Professor Harald Haas, from the [University of Edinburgh](#) in the UK, is widely recognised as the original founder of Li-Fi. He coined the term Li-Fi and is Chair of Mobile Communications at the [University of Edinburgh](#) and co-founder of pureLiFi.

The general term [visible light communication](#) (VLC), includes any use of the visible light portion of the electromagnetic spectrum to transmit information. The D-Light project at Edinburgh's Institute for Digital Communications was funded from January 2010 to January 2012. Haas promoted this technology in his 2011 [TED Global](#) talk and helped start a company to market it. PureLiFi, formerly pureVLC, is an [original equipment manufacturer](#) (OEM) firm set up to commercialize Li-Fi products for integration with existing [LED](#)-lighting systems.

In October 2011, companies and industry groups formed the [Li-Fi Consortium](#), to promote high-speed optical wireless systems and to overcome the limited amount of radio-based wireless spectrum available by exploiting a completely different part of the electromagnetic spectrum. A number of companies offer uni-directional VLC products which is not the same as Li-Fi.

VLC technology was exhibited in 2012 using Li-Fi. By August 2013, data rates of over 1.6 [Gbps](#) were demonstrated over a single color LED. In September 2013, a press release said that Li-Fi, or VLC systems in general, do not require line-of-sight conditions. In October 2013, it was reported Chinese manufacturers were working on Li-Fi development kits.

One part of VLC is modeled after communication protocols established by the [IEEE](#) workgroup. However, the IEEE 802.15.7 standard is out-of-date. Specifically, the standard fails to consider the latest technological developments in the field of optical wireless communications, specifically with the introduction of optical [orthogonal frequency-division multiplexing](#) (O-OFDM) modulation methods which have been optimized for data rates, multiple-access and energy

efficiency have. The introduction of O-OFDM means that a new drive for standardization of optical wireless communications is required.

Nonetheless, the IEEE 802.15.7 standard defines the [physical layer](#) (PHY) and [media access control](#) (MAC) layer. The standard is able to deliver enough data rates to transmit audio, video and multimedia services. It takes into account the optical transmission mobility, its compatibility with artificial lighting present in infrastructures, the deviance which may be caused by interference generated by the ambient lighting. The MAC layer allows to use the link with the other layers like the [TCP/IP](#) protocol.

The standard defines three PHY layers with different rates:

- The PHY I was established for outdoor application and works from 11.67 kbit/s to 267.6 kbit/s.
- The PHY II layer allows to reach data rates from 1.25 Mbit/s to 96 Mbit/s.
- The PHY III is used for many emissions sources with a particular modulation method called color shift keying (CSK). PHY III can deliver rates from 12 Mbit/s to 96 Mbit/s.

The modulation formats recognized for PHY I and PHY II are the coding [on-off keying](#) (OOK) and variable [pulse position modulation](#) (VPPM). The [Manchester coding](#) used for the PHY I and PHY II layers include the clock inside the transmitted data by representing a logic 0 with an OOK symbol "01" and a logic 1 with an OOK symbol "10", all with a DC component. The DC component avoids the light extinction in case of an extended line of logic 0.

The first Li-Fi [smartphone](#) prototype was presented at the [Consumer Electronics Show](#) in Las Vegas from January 7–10 in 2014. The phone uses Sun Partner's Wysips CONNECT, a technique that converts light waves into usable energy, making the phone capable of receiving and decoding signals without drawing on its battery.

ECG, HEART RATE AND BODY TEMPERATURE

2.1 HUMAN HEART

The human heart is a muscular organ that provides a continuous blood circulation through the cardiac cycle and is one of the most vital organs in the human body. The heart is divided into four main chambers: the two upper chambers are called the left and right atria and two lower chambers are called the right and left ventricles. There is a thick wall of muscle separating the right side and the left side of the heart called the septum. Normally with each beat the right ventricle pumps the same amount of blood into the lungs that the left ventricle pumps out into the body. Physicians commonly refer to the right atrium and right ventricle together as the right heart and to the left atrium and ventricle as the left heart.

The electric energy that stimulates the heart occurs in the sinoatrial node which produces a definite potential and then discharges, sending an impulse across the atria. In the atria the electrical signal move from cell to cell while in the ventricles the signal is carried by specialized tissue called the Purkinje fibers which then transmit the electric charge to the myocardium.

2.2 ELECTROCARDIOGRAPH (ECG)

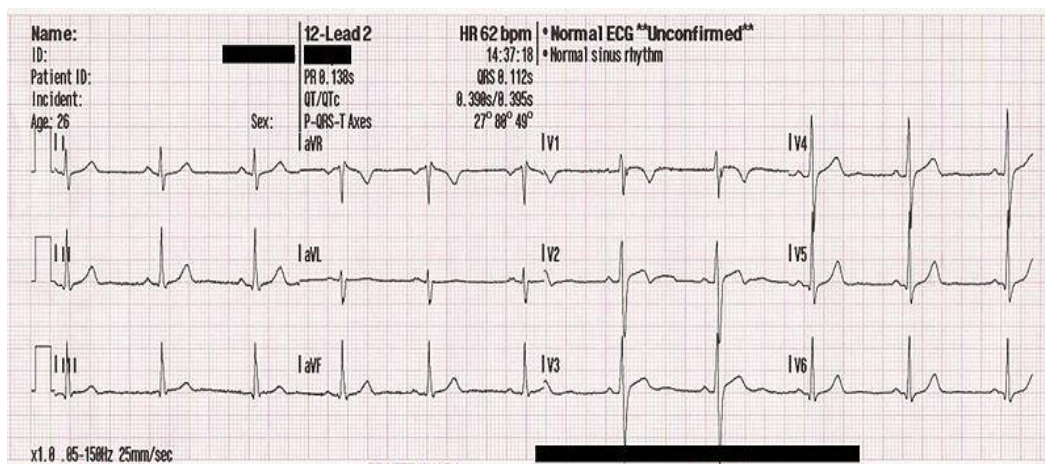


Fig 2.1: 12 Lead ECG of a 26-year-old male.

Electrocardiograph (ECG) is a transthoracic interpretation of the electrical activity of the heart over time captured and externally recorded by skin electrodes. It is a noninvasive recording produced by an electrocardiographic device.

The ECG works mostly by detecting and amplifying the tiny electrical changes on the skin that are caused when the heart muscle "depolarizes" during each heart beat. At rest, each heart muscle cell has a charge across its outer wall, or cell membrane reducing this charge towards zero is called de-polarization, which activates the mechanisms in the cell that cause it to contract. During each heartbeat a healthy heart will have an orderly progression of a wave of depolarization that is triggered by the cells in the sinoatrial node, spreads out through the atrium, passes through "intrinsic conduction pathways" and then spreads all over the ventricles. This is detected as tiny rises and falls in the voltage between two electrodes placed either side of the heart which is displayed as a wavy line either on a screen or on paper. This display indicates the overall rhythm of the heart and weaknesses in different parts of the heart muscle.

2.3 HEART RATE

Heart rate is the number of heartbeats per unit of time, typically expressed as **beats per minute** (bpm). Heart rate can vary as the body's need to absorb oxygen and excrete carbon dioxide changes, such as during exercise or sleep.

The measurement of heart rate is used by medical professionals to assist in the diagnosis and tracking of medical conditions. It is also used by individuals, such as athletes, who are interested in monitoring their heart rate to gain maximum efficiency from their training. The *R wave to R wave interval (RR interval)* is the inverse of the heart rate.

Heart rate is measured by finding the pulse of the body. This pulse rate can be measured at any point on the body where the artery's pulsation is transmitted to the surface by pressuring it with the index and middle fingers; often it is compressed against an underlying structure like bone. The thumb should not be used for measuring another person's heart rate, as its strong pulse may interfere with discriminating the site of pulsation.

The resting heart rate (HR_{rest}) is a person's heart rate when they are at rest, that is lying down but awake, and not having recently exerted themselves. The typical healthy resting heart rate in adults is 60–80 bpm, with rates below 60 bpm referred to as bradycardia, and rates above 100 bpm referred to as tachycardia. Note however that conditioned athletes often have resting heart rates below 60 bpm. And it is not unusual for people doing regular exercise to get below 50 bpm.

2.4 THERMO REGULATION

Thermoregulation is the ability of an organism to keep its body temperature within certain boundaries, even when the surrounding temperature is very different. This process is one aspect of homeostasis: a dynamic state of stability between an animal's internal environment and its *external* environment or If the body is unable to maintain a normal temperature and it increases significantly above normal, a condition known as hyperthermia occurs. This occurs when the body is exposed to constant temperatures of approximately 55° C, any prolonged exposure (longer than a few hours) at this temperature and up to around 70° C death is almost inevitable. The opposite condition, when body temperature decreases below normal levels, is known as hypothermia

Different parts of the body have different temperatures. Rectal and vaginal measurements, or measurements taken directly inside the body cavity, are typically slightly higher than oral measurements, and oral measurements are somewhat higher than skin temperature. The commonly accepted average core body temperature (taken internally) is 37.0 °C (98.6 °F). The typical oral (under the tongue) measurement is slightly cooler, at 36.8 ± 0.7 °C, or 98.2 ± 1.3 °F. In Russia and former Soviet countries, the commonly quoted value is 36.6 °C (97.9 °F), based on an armpit (auxiliary) reading. Although some people think of these numbers as representing the normal temperature, a wide range of temperatures has been found in healthy people. In samples of normal adult men and women, the observed range for oral temperature is 33.2–38.2 °C (92–101 °F), for rectal it is 34.4–37.8 °C (94–100 °F), for the Tympanic cavity it is 35.4–37.8 °C (96–100 °F) and for auxiliary it is 35.5–37.0 °C (96–99 °F).

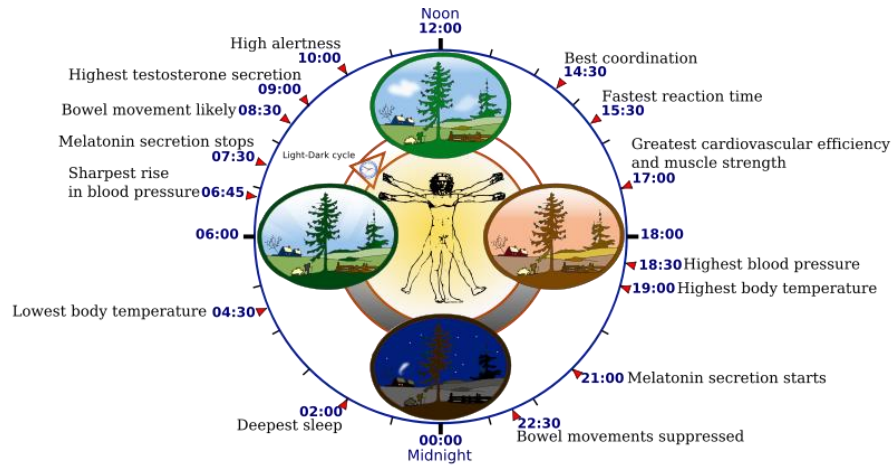


Figure 2.2: Overview of biological clock in humans.

CHAPTER 3

EXISTING SYSTEM

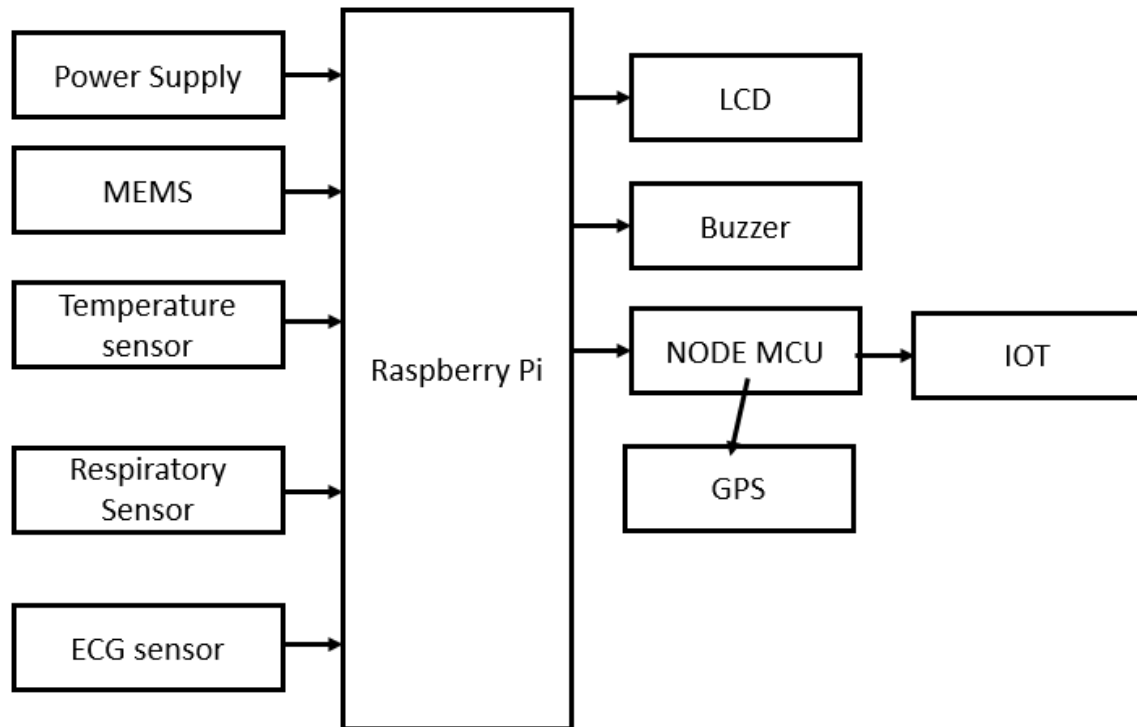
- Robust healthcare is a requirement for both developed countries, where the cost of healthcare is high and security and privacy are critical issues and developing countries like India,
- where there is a mass population to handle in hospitals and robust healthcare procedures are required. Doctors should visit the patient regularly and check the status. Situation may occur like nurse make mistake while noting the report.
- Doctors or Nurses will use the NFC card in card reader so that it sensors will send the details to microcontroller, from there through ZIGBEE we can view in PC.
- NFC tag can be employed for automation of identification of patient and doctors in hospital using smart phones and tablet computers that use the Android platform NFC can be used to explore new ways of real-time hospital workflows and data processing in the medical world.

DRAWBACKS

- In small hospitals this method is possible, but when it comes to big hospitals it is not possible.
- We cannot maintain a database of a patient in hardcopy if any accidents like fire or file misplacement occurs.
- As more and more users are tapped in with their devices, the clogged airwaves make it difficult to latch on a reliable signal.

PROPOSED SSYTEM:

- We have proposed a robust health monitoring system that is intelligent enough to monitor the patient automatically using IOT.
- It collects the status information through these systems which would include patient's temperature, body movement, heart rate, blood pressure and ECG and sends an emergency alert to patient's doctor as well as to the caretaker with his current status.
- We connected respiratory sensor, Temperature sensor, MEMS Sensor, and Heart beat Sensors to the Node MCU to monitor our home through IOT Technology.
- This would help the doctor and caretaker to monitor his patient from anywhere in the world.
- The system uses smart sensors that generates raw data information collected from each sensor and send it to a cloud server where the data can be further analyzed and statistically maintained to be used.

BLOCK DIAGRAM:**HARDWARE REQUIREMENTS**

- Power Supply
- ATmega328P Microcontroller
- Node mcu
- MEMS
- Temperature sensor
- Respiratory Sensor
- ECG sensor

SOFTWARE REQUIREMENTS

- ARDUINO IDE
- Language: Embedded C
- Proteus simulation

CHAPTER 4

HARDWARE DESIGN

ARDUINO NANO AND ITS PROGRAMMING

Arduino is a tool for making computers that can sense and control more of the physical world than your desktop computer. It's an open-source physical computing platform based on a simple microcontroller board, and a development environment for writing software for the board.

Arduino can be used to develop interactive objects, taking inputs from a variety of switches or sensors, and controlling a variety of lights, motors, and other physical outputs. Arduino projects can be stand-alone, or they can be communicate with software running on your computer. The boards can be assembled by hand or purchased preassembled; the open-source IDE can be downloaded for free.

The Arduino programming language is an implementation of Wiring, a similar physical computing platform, which is based on the Processing multimedia programming environment.

2.1 Overview

The Arduino microcontroller is an easy to use yet powerful single board computer that has gained considerable traction in the hobby and professional market. The Arduino is open-source, which means hardware is reasonably priced and development software is free. This guide is for students in ME 2011, or students anywhere who are confronting the Arduino for the first time. For advanced Arduino users, prowl the web; there are lots of resources.

This guide covers the Arduino Nano board (Spark fun DEV-09950, \$29.95), a good choice for students and educators. With the Arduino board, you can write programs and create interface circuits to read switches and other sensors, and to control motors and lights with very little effort. Many of the pictures and drawings in this guide were taken from the documentation on the

The Duemilanove board features an Atmel ATmega328 microcontroller operating at 5 V with 2 Kb of RAM, 32 Kb of flash memory for storing programs and 1 Kb of EEPROM for storing parameters. The clock speed is 16 MHz, which translates to about executing about 300,000 lines of C source code per second. The board has 14 digital I/O pins and 6 analog input pins. There is a USB connector for talking to the host computer and a DC power jack for connecting an external 6-20 V power source, for example a 9 V battery, when running a program while not connected to the host computer. Headers are provided for interfacing to the I/O pins using 22 g solid wire or header connectors.

The Arduino programming language is a simplified version of C/C++. If you know C, programming the Arduino will be familiar. If you do not know C, no need to worry as only a few commands are needed to perform useful functions.

An important feature of the Arduino is that you can create a control program on the host PC, download it to the Arduino and it will run automatically. Remove the USB cable connection to the PC, and the program will still run from the top each time you push the reset button. Remove the battery and put the Arduino board in a closet for six months. When you reconnect the battery, the last program you stored will run. This means that you connect the board to the host PC to develop and debug your program, but once that is done, you no longer need the PC to run the program.

The Arduino Nano is a microcontroller board based on the ATmega328. It has 14 digital input/output pins (of which 6 can be used as PWM outputs), 6 analog inputs, a 16 MHz ceramic resonator, a USB connection, a power jack, an ICSP header, and a reset button. It contains everything needed to support the microcontroller; simply connect it to a computer with a USB cable or power it with a AC-to-DC adapter or battery to get started.

The Nano differs from all preceding boards in that it does not use the FTDI USB-to-serial driver chip. Instead, it features the Atmega16U2 (Atmega8U2 up to version R2) programmed as a USB-to-serial converter.

2.2 Summary:

Microcontroller ATmega328

Operating Voltage 5V

Input Voltage (recommended) 7-12V

Input Voltage (limits) 6-20V

Digital I/O Pins 14 (of which 6 provide PWM output)

Analog Input Pins 6

DC Current per I/O Pin 40 mA

DC Current for 3.3V Pin 50 mA

Flash Memory 32 KB (ATmega328) of which 0.5 KB used by bootloader

SRAM 2 KB (ATmega328)

EEPROM 1 KB (ATmega328)

Clock Speed 16 MHz

2.3 Pin Configuration

The Arduino Nano can be powered via the USB connection or with an external power supply. The power source is selected automatically.

External (non-USB) power can come either from an AC-to-DC adapter (wall- wart) or battery. The adapter can be connected by plugging a 2.1mm center-positive plug into the board's power jack. Leads from a battery can be inserted in the Gnd and Vin pin headers of the POWER connector.

The board can operate on an external supply of 6 to 20 volts. If supplied with less than 7V, however, the 5V pin may supply less than five volts and the board may be unstable. If using more than 12V, the voltage regulator may overheat and damage the board. The recommended range is 7 to 12 volts.

The power pins are as follows:

- **VIN.** The input voltage to the Arduino board when it's using an external power source (as opposed to 5 volts from the USB connection or other regulated power source). You can supply voltage through this pin, or, if supplying voltage via the power jack, access it through this pin.
- **5V.** this pin outputs a regulated 5V from the regulator on the board. The board can be supplied with power either from the DC power jack (7 - 12V), the USB connector (5V), or the VIN pin of the board (7-12V). Supplying voltage via the 5V or 3.3V pins bypasses the regulator, and can damage your board.
- **3V3.** A 3.3 volt supply generated by the on-board regulator. Maximum current draw is 50 mA.
- **GND.** Ground pins.
- **IOREF.** This pin on the Arduino board provides the voltage reference with which the microcontroller operates. A properly configured shield can read the IOREF pin voltage and select the appropriate power source or enable voltage translators on the outputs for working with the 5V or 3.3V.

Memory

The ATmega328 has 32 KB (with 0.5 KB used for the bootloader). It also has 2 KB of SRAM and 1 KB of EEPROM.

Input and Output

Each of the 14 digital pins on the Nano can be used as an input or output, using `pinMode()`, `digitalWrite()`, and `digitalRead()` functions. They operate at 5 volts. Each pin can provide or receive a maximum of 40 mA and has an internal pull-up resistor (disconnected by default) of 20-50 kohms. In addition, some pins have specialized functions:

- **Serial: 0 (RX) and 1 (TX).** Used to receive (RX) and Transmit (TX) TTL serial data.

These pins are connected to the corresponding pins of the ATmega8U2 USB-to- TTL Serial chip.

- **External Interrupts: 2 and 3.** These pins can be configured to trigger an interrupt on a low value, a rising or falling edge, or a change in value. See the `attachInterrupt()` function for details.

- **PWM: 3, 5, 6, 9, 10, and 11.** Provide 8-bit PWM output with the `analogWrite()` function.

- **SPI: 10 (SS), 11 (MOSI), 12 (MISO), 13 (SCK).** These pins support SPI communication using the SPI library.

- **LED: 13.** There is a built-in LED connected to digital pin 13. When the pin is HIGH value, the

LED is on, when the pin is LOW, it's off.

The Nano has 6 analog inputs, labeled A0 through A5, each of which provide 10 bits of resolution (i.e. 1024 different values). By default they measure from ground to 5 volts, though it is possible to change the upper end of their range using the AREF pin and the `analogReference()` function. Additionally, some pins have specialized functionality:

- **TWI: A4 or SDA pin and A5 or SCL pin.** Support TWI communication using the `Wire` library.

There are a couple of other pins on the board:

- **AREF.** Reference voltage for the analog inputs. Used with `analogReference()`.

Reset. Bring this line LOW to reset the microcontroller. Typically used to add a reset button to shields which block the one on the board.

2.4 Communication

Microcontrollers depend on a host computer for developing and compiling programs. The software used on the host computer is known as an integrated development environment, or IDE. For the Arduino, the development environment is based on the open source Processing platform (www.processing.org) which is described by its creators as a “programming language and environment for people who want to program images, animation, and interactions.” The Arduino programming language leverages an open source project known as Wiring (wiring.org.co). The Arduino language is based on good old- fashioned C. If you are unfamiliar with this language, don't worry; it's not hard to learn, and the Arduino IDE provides some feedback when you make mistakes in your programs.

The Arduino Nano has a number of facilities for communicating with a computer, another

Arduino, or other microcontrollers. The ATmega328 provides UART TTL (5V) serial communication, which is available on digital pins 0 (RX) and 1 (TX). An ATmega16U2 on the board channels this serial communication over USB and appears as a virtual com port to software on the computer. The '16U2 firmware uses the standard USB COM drivers, and no external driver is needed. However, on Windows, a inf file is required. The Arduino software includes a serial monitor which allows simple textual data to be sent to and from the Arduino board. The RX and TX LEDs on the board will flash when data is being transmitted via the USB-to-serial chip and USB connection to the computer (but not for serial communication on pins 0 and 1).

A Software Serial library allows for serial communication on any of the Nano's digital pins. The ATmega328 also supports I2C (TWI) and SPI communication. The Arduino software includes a Wire library to simplify use of the I2C bus; see the documentation for details. For SPI communication, use the SPI library.

As you go through the list of programming statements available in the Arduino IDE (choose Help->Reference), you might think there isn't much power for doing things like running servos, operating stepper motors, reading potentiometers, or displaying text on an LCD. Like most any language based on C, the Arduino supports the notion of "libraries" code

Repositories that extend core programming functionality. Libraries let you re-use code without having to physically copy and paste it into all your programs. The standard Arduino software installation comes with several libraries you may use, and you can download others from the Arduino support pages and from third-party websites that publish Arduino library code. A good example of a library you'll use with the Robot and likely many other robot projects is

Servo. This library allows you to connect one or more hobby R/C servos to the Arduino's digital I/O pins. The Servo library comes with the standard Arduino installation package Library->Servo. This adds the line

```
#include <Servo.h>
```

Which tells the Arduino IDE that you wish to include the Servo library in your sketch. With the functionality of the library now available to you, you can use its various functions to control one or more servos. For example, you can use the write function to rotate a servo to a specific position, from 0 to 180 degrees. The following code

```
myServo.write(90);
```

Moves a servo to its midpoint, or 90 degree position. Structurally, Arduino sketches are very straightforward and are pretty easy to read and understand. The Arduino program contains two main parts: setup () and loop (). These are programming functions that do what their names suggest: setup () sets up the Arduino hardware, such as specifying which I/O lines you plan to use, and whether

They are inputs or outputs. The loop () function is repeated endlessly when the Arduino is operating.

Arduino IDE (Integrated development environment) is used to write the program and dump into the Arduino board

NodeMCU is an open-source Lua based firmware and **development board** specially targeted for IoT based Applications. It includes firmware that runs on the ESP8266 Wi-Fi SoC from Espressif Systems, and hardware which is based on the ESP-12 module

Programming NodeMCU ESP8266 with Arduino IDE

The NodeMCU Development Board can be easily programmed with Arduino IDE since it is easy to use.

Programming NodeMCU with the Arduino IDE will hardly take 5-10 minutes. All you need is the Arduino IDE, a USB cable and the NodeMCU board itself. You can check this [Getting Started Tutorial for NodeMCU](#) to prepare your Arduino IDE for NodeMCU.

NodeMCU is an open source firmware for which open source [prototyping](#) board designs are available. The name "NodeMCU" combines "[node](#)" and "MCU" ([micro-controller](#) unit).^[8] The term "NodeMCU" strictly speaking refers to the firmware rather than the associated [development kits](#).^[citation needed]

Both the firmware and prototyping board designs are [open source](#).^[8]

The firmware uses the [Lua](#) scripting language. The firmware is based on the eLua project, and built on the Espressif Non-OS SDK for ESP8266. It uses many open source projects, such as lua-cjson^[9] and [SPIFFS](#).^[10] Due to resource constraints, users need to select the modules relevant for their project and build a firmware tailored to their needs. Support for the 32-bit [ESP32](#) has also been implemented.

The prototyping hardware typically used is a circuit board functioning as a [dual in-line package](#) (DIP) which integrates a USB controller with a smaller surface-mounted board containing the MCU and antenna. The choice of the DIP format allows for easy prototyping on [breadboards](#). The design was initially based on the ESP-12 module of the [ESP8266](#), which is a Wi-Fi SoC integrated with a [Tensilica](#) Xtensa LX106 core, widely used in IoT applications (see [related projects](#)).

NodeMCU was created shortly after the [ESP8266](#) came out. On December 30, 2013, [Espressif Systems](#)^[6] began production of the ESP8266.^[11] NodeMCU started on 13 Oct 2015, when Hong committed the first file of nodemcu-firmware to GitHub.^[12] Two months later, the project expanded to include an open-hardware platform when developer Huang R committed the [gerber](#) file of an ESP8266 board, named devkit v0.9.^[13] Later that month, Tuan PM ported [MQTT](#) client library from [Contiki](#) to the ESP8266 SoC platform,^[14] and committed to NodeMCU project, then NodeMCU was able to support the MQTT IoT protocol, using Lua to access the MQTT broker. Another important update was made on 30 Jan 2015, when Devsaurus

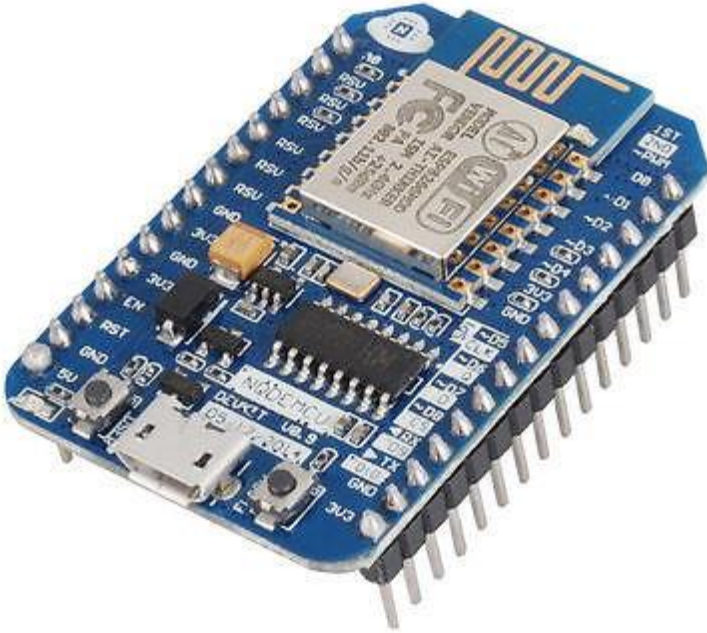
ported the `u8glib`^[15] to the NodeMCU project,^[16] enabling NodeMCU to easily drive LCD, Screen, OLED, even VGA displays.

In the summer of 2015 the original creators abandoned the firmware project and a group of independent contributors took over. By the summer of 2016 the NodeMCU included more than 40 different modules.

ESP8266 Arduino Core^[edit]

As [Arduino.cc](http://arduino.cc) began developing new MCU boards based on non-AVR processors like the ARM/SAM MCU and used in the Arduino Due, they needed to modify the Arduino IDE so that it would be relatively easy to change the IDE to support alternate toolchains to allow Arduino C/C++ to be compiled for these new processors. They did this with the introduction of the Board Manager and the SAM Core. A "core" is the collection of software components required by the Board Manager and the Arduino IDE to compile an Arduino C/C++ source file for the target MCU's machine language. Some ESP8266 enthusiasts developed an Arduino core for the ESP8266 WiFi SoC, popularly called the "ESP8266 Core for the Arduino IDE".^[17] This has become a leading software development platform for the various ESP8266-based modules and development boards, including NodeMCUs.

NodeMCU is an open-source LUA based firmware developed for the ESP8266 wifi chip. By exploring functionality with the ESP8266 chip, NodeMCU firmware comes with the ESP8266 Development board/kit i.e. NodeMCU Development board.

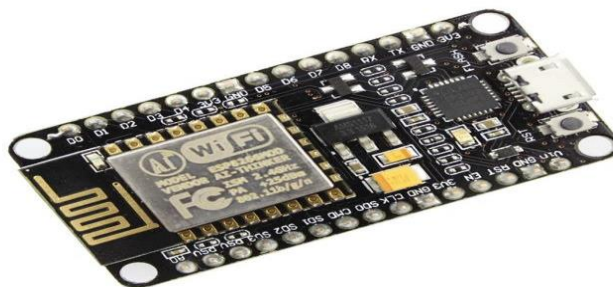


NodeMCU Development Board/kit v0.9 (Version1)

Since NodeMCU is an open-source platform, its hardware design is open for edit/modify/build.

NodeMCU Dev Kit/board consist of ESP8266 wifi enabled chip. The **ESP8266** is a low-cost Wi-Fi chip developed by Espressif Systems with TCP/IP protocol. For more information about ESP8266, you can refer to the ESP8266 WiFi Module.

There is Version2 (V2) available for NodeMCU Dev Kit i.e. **NodeMCU Development Board v1.0 (Version2)**, which usually comes in black colored PCB.



NodeMCU Development Board/kit v1.0 (Version2)

For more information about NodeMCU Boards available in the market refer to [NodeMCU Development Boards](#)

NodeMCU Dev Kit has **Arduino like** Analog (i.e. A0) and Digital (D0-D8) pins on its board.

It supports serial communication protocols i.e. UART, SPI, I2C, etc.

Using such serial protocols we can connect it with serial devices like I2C enabled LCD display, Magnetometer HMC5883, MPU-6050 Gyro meter + Accelerometer, RTC chips, GPS modules, touch screen displays, SD cards, etc.

How to start with NodeMCU?

NodeMCU Development board is featured with wifi capability, analog pin, digital pins, and serial communication protocols.

To get started with using NodeMCU for IoT applications first we need to know about how to write/download NodeMCU firmware in NodeMCU Development Boards. And before that where this NodeMCU firmware will get as per our requirement.

There are online NodeMCU custom builds available using which we can easily get our custom NodeMCU firmware as per our requirement.

To know more about how to build custom NodeMCU firmware online and download it refer to [Getting started with NodeMCU](#)

How to write codes for NodeMCU?

After setting up ESP8266 with Node-MCU firmware, let's see the IDE (Integrated Development Environment) required for the development of NodeMCU.

NodeMCU with ESPlorer IDE

Lua scripts are generally used to code the NodeMCU. Lua is an open-source, lightweight, embeddable scripting language built on top of C programming language.

For more information about how to write Lua script for NodeMCU refer to [Getting started with NodeMCU using ESPlorerIDE](#)

NodeMCU with Arduino IDE

Here is another way of developing NodeMCU with a well-known IDE i.e. Arduino IDE. We can also develop applications on NodeMCU using the Arduino development environment. This makes it easy for Arduino developers than learning a new language and IDE for NodeMCU.

For more information about how to write Arduino sketch for NodeMCU refer to [Getting started with NodeMCU using Arduino IDE](#)

The difference in using ESPlorer and Arduino IDE

Well, there is a programming language difference we can say while developing an application for NodeMCU using ESPlorer IDE and Arduino IDE.

We need to code in C/C++ programming language if we are using Arduino IDE for developing NodeMCU applications and Lua language if we are using ESPlorer IDE.

Basically, NodeMCU is Lua Interpreter, so it can understand Lua script easily. When we write Lua scripts for NodeMCU and send/upload it to NodeMCU, then they will get executed sequentially. It will not build a binary firmware file of code for NodeMCU to write. It will send Lua script as it is to NodeMCU to get executed.

In Arduino IDE when we write and compile code, the ESP8266 toolchain in the background creates a binary firmware file of code we wrote. And when we upload it to NodeMCU then it will flash all NodeMCU firmware with newly generated binary firmware code. In fact, it writes the complete firmware.

That's the reason why NodeMCU not accept further Lua scripts/code after it is getting flashed by Arduino IDE. After getting flashed by Arduino sketch/code it will be no more Lua interpreter and we got an error if we try to upload Lua scripts. To again start with Lua script, we need to flash it with NodeMCU firmware.

Since Arduino IDE compiles and upload/writes complete firmware, it takes more time than ESPlorer IDE.

NodeMCU ESP8266 Specifications & Features

- Microcontroller: Tensilica 32-bit RISC CPU Xtensa LX106
- Operating Voltage: 3.3V
- Input Voltage: 7-12V
- Digital I/O Pins (DIO): 16
- Analog Input Pins (ADC): 1
- UARTs: 1
- SPIs: 1
- I2Cs: 1
- Flash Memory: 4 MB
- SRAM: 64 KB
- Clock Speed: 80 MHz
- USB-TTL based on CP2102 is included onboard, Enabling Plug n Play
- PCB Antenna
- Small Sized module to fit smartly inside your IoT projects

TEMPERATURE SENSOR (LM35)

FEATURES DESCRIPTION

- Calibrated Directly in ° Celsius (Centigrade)
- Linear + 10 mV/°C Scale Factor
- 0.5°C Ensured Accuracy (at +25°C) LM35 has an advantage over linear temperature
- Rated for Full -55°C to +150°C Range
- Suitable for Remote Applications
- Low Cost Due to Wafer-Level Trimming
- Less than 60-μA Current Drain

- Low Self-Heating, 0.08°C in Still Air
- Nonlinearity Only $\pm\frac{1}{4}^{\circ}\text{C}$ Typical
- Low Impedance Output, 0.1 Ω for 1 mA Load

DESCRIPTION

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

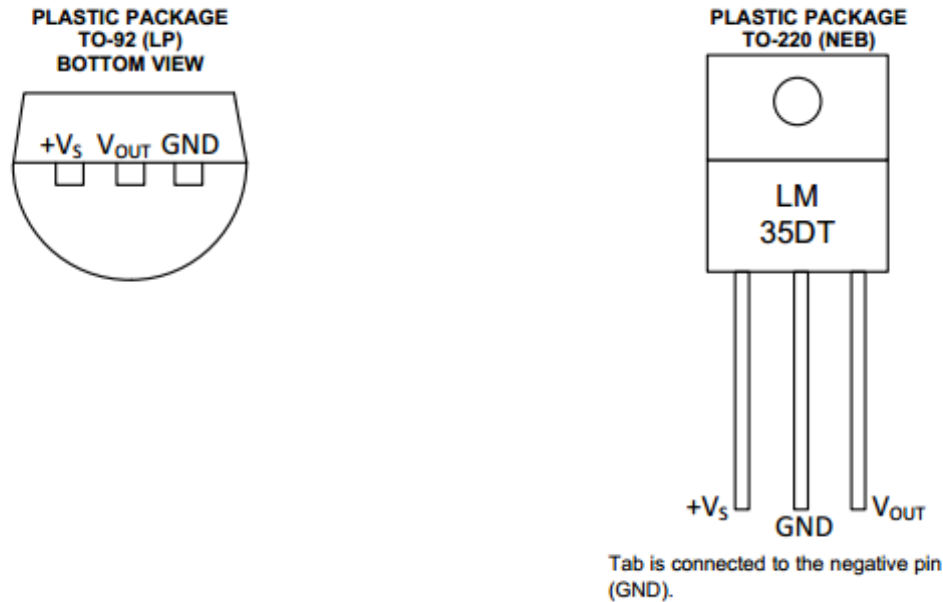


Figure. View of Temperature sensor

HEARTBEAT SENSOR

A person's heartbeat is the sound of the valves in his/her's heart contracting or expanding as they force blood from one region to another. The number of times the heart beats per minute (BPM), is the heart beat rate and the beat of the heart that can be felt in any artery that lies close to the skin is the pulse.

Two Ways to Measure a Heartbeat

Manual Way: Heart beat can be checked manually by checking one's pulses at two locations- wrist (the radial pulse) and the neck (carotid pulse). The procedure is to place the two fingers (index and middle finger) on the wrist (or neck below the windpipe) and count the number of pulses for 30 seconds and then multiplying that number by 2 to get the heart beat rate. However pressure should be applied minimum and also fingers should be moved up and down till the pulse is felt.

Using a sensor: Heart Beat can be measured based on optical power variation as light is scattered or absorbed during its path through the blood as the heart beat changes.

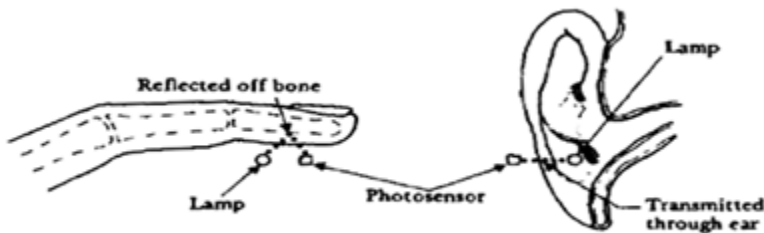
Principle of Heartbeat Sensor

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

There are two types of photoplethysmography:

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

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

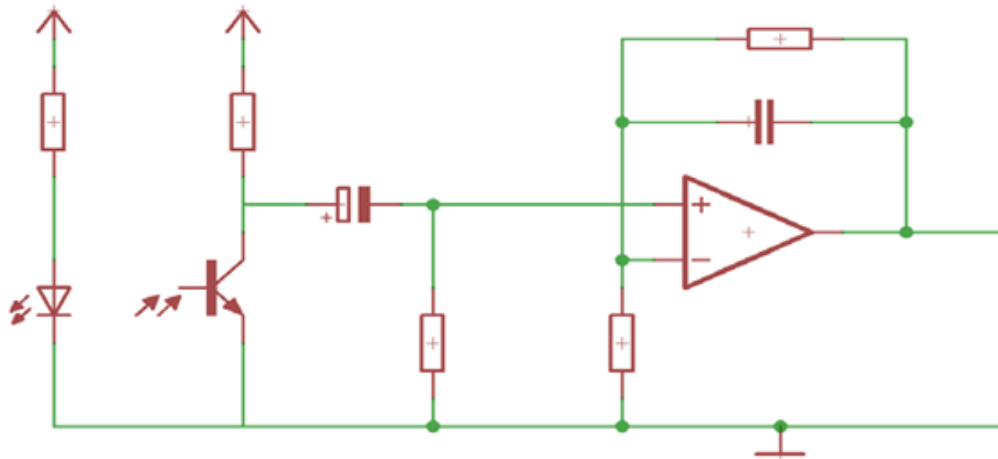


Working of a Heartbeat Sensor

The basic heartbeat sensor consists of a light emitting diode and a detector like a light detecting resistor or a photodiode. The heart beat pulses causes a variation in the flow of blood to different regions of the body. When a tissue is illuminated with the light source, i.e. light emitted by the led, it either reflects (a finger tissue) or transmits the light (earlobe). Some of the light is absorbed by the blood and the transmitted or the reflected light is received by the light detector. The amount of light absorbed depends on the blood volume in that tissue. The detector output is in form of electrical signal and is proportional to the heart beat rate.

This signal is actually a DC signal relating to the tissues and the blood volume and the AC component synchronous with the heart beat and caused by pulsatile changes in arterial blood

volume is superimposed on the DC signal. Thus the major requirement is to isolate that AC component as it is of prime importance.



To achieve the task of getting the AC signal, the output from the detector is first filtered using a 2 stage HP-LP circuit and is then converted to digital pulses using a comparator circuit or using simple ADC. The digital pulses are given to a microcontroller for calculating the heart beat rate,

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

Where f is the pulse frequency

Practical Heartbeat Sensor

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



Another example is TCRT1000, having 4 pins-

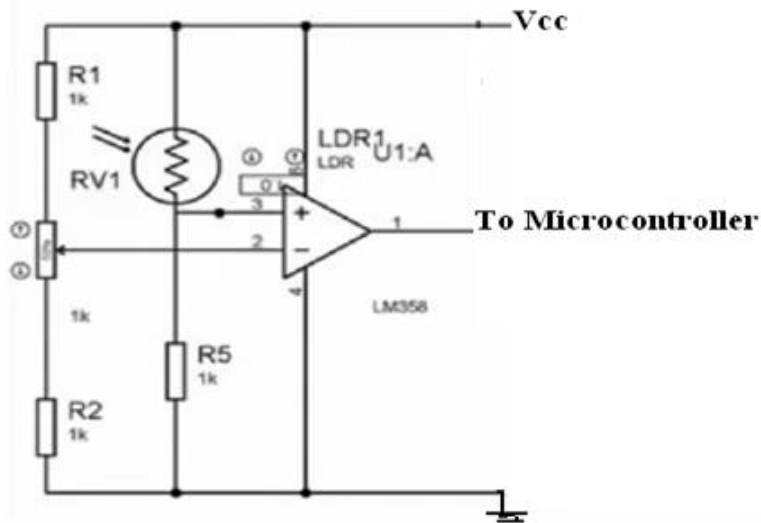
Pin1: To give supply voltage to the LED

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



Application Developing your own Heartbeat Sensor System

A basic Heartbeat Sensor system can also be built using basic components like a ldr, comparator IC LM358 and a Microcontroller as given below



As described above regarding the principle of heart beat sensor, when the finger tissue or the earlobe tissue is illuminated using a light source, the light is transmitted after getting modulated i.e. a part getting absorbed by the blood and the rest being transmitted. This modulated light is received by the light detector.

Here a Light Dependant Resistor (LDR) is used as a light detector. It works on the principle that when light falls on the resistor, its resistance changes. As the light intensity increases, the resistance decreases. Thus the voltage drop across the resistor decreases.

Here a comparator is used which compares the output voltage from the LDR to that of the threshold voltage. The threshold voltage is the voltage drop across the LDR when the light with fixed intensity, from the light source falls directly on it. The inverting terminal of the comparator LM358 is connected to the potential divider arrangement which is set to the threshold voltage and the non-inverting terminal is connected to the LDR. When a human tissue is illuminated using the light source, the intensity of the light reduces. As this reduced light intensity falls on the LDR, the resistance increases and as a result the voltage drop increases. When the voltage drop across the LDR or the non-inverting input exceeds that of the inverting input, a logic high signal is developed at the output of the comparator and in case voltage drop being lesser a logic low output is developed. Thus the output is a series of pulses. These pulses can be fed to the Microcontroller which accordingly processes the information to get the heart beat rate and this is displayed on the Display interfaced to the Microcontroller.

Micro-electromechanical systems (MEMS)

Micro-electromechanical systems (MEMS) is a technology that combines computers with tiny mechanical devices such as sensors, valves, gears, mirrors, and actuators embedded in semiconductor chips. MEMS or what he calls analog computing will be "the foundational technology of the next decade." MEMS is also sometimes called smart matter.



MEMS are already used as accelerometers in automobile air-bags. They've replaced a less reliable device at lower cost and show promise of being able to inflate a bag not only on the basis of sensed deceleration but also on the basis of the size of the person they are protecting. Basically, a MEMS device contains micro-circuitry on a tiny silicon chip into which some mechanical device such as a mirror or a sensor has been manufactured. Potentially, such chips can be built in large quantities at low cost, making them cost-effective for many uses.

Among the presently available uses of MEMS or those under study are:

- Global position system sensors that can be included with courier parcels for constant tracking and that can also sense parcel treatment en route
- Sensors built into the fabric of an airplane wing so that it can sense and react to air flow by changing the wing surface resistance; effectively creating a myriad of tiny wing flaps

- Optical switching devices that can switch light signals over different paths at 20-nanosecond switching speeds
- Sensor-driven heating and cooling systems that dramatically improve energy savings
- Building supports with imbedded sensors that can alter the flexibility properties of a material based on atmospheric stress sensing

Applications:

- Vibrations measurement and Tilt inclination (to adjust rotation speed, circuit breaker security, fault prediction),
- Professional and domestic Alarms
- Maintenance management and Asset Tracking for shocks monitoring
- Alarm and Anti Tamper function

IOT

The structure of this chapter is the following: Section 1.1 defines the motivation for this work. Section 1.2 shows the research objectives and questions. Section 1.3 presents the steps required to achieve our objectives. Section 1.4 shows the structure of the report.

1.1 Background

The Future Internet goal is to provide an infrastructure to have an immediate access to information about the physical world and its objects. Physical objects can be applicable to different application domains, such as e-health, warehouse management, etc. Each application domain may have different types of physical devices. Each physical device can have its own specifications, which is required to use in order to interact with it. To achieve the future Internet goal, a layered vision is required that can facilitate data access. Internet of Things (IoT) is a vision that aims to integrate the virtual world of information to the real world of devices through a layered architecture.

The term „Internet of Things“ consists of two words, namely *Internet* and *Things*. *Internet* refers to the global network infrastructure with scalable, configurable capabilities based on interoperable and standard communication protocols. *Things* are physical objects or devices, or virtual objects, devices or information, which have identities, physical attributes and virtual personalities, and use intelligent interfaces [1]. For instance, a virtual object can represent an abstract unit of sensor nodes that contains metadata to identify and discover its corresponding sensor nodes. Therefore, IoT refers to the *things* that can provide information from the physical environment through the Internet.

Middleware is as an interface between the hardware layer and the application layer, which is responsible for interacting with devices and information management [2]. The role of a middleware is to present a unified programming model to interact with devices. A middleware is in charge of masking the heterogeneity and distribution problems that we face when interacting with devices [3].

1.2 Motivation

IoT-based system is in charge of providing knowledge from an environment to a non-expert user. IoT-based system can be used in different environments, so it needs to be able to address many heterogeneous devices. Thus, a major concern within developing an IoT-based system is how to handle the interaction with the heterogeneous devices for non-expert users. This concern can be addressed by a middleware layer between devices and non-expert users. This layer is responsible to hide the diversity of devices from the user perspective, and provides access transparency to the devices for the end users.

The idea of creating abstractions of devices been addressed in the literature. The middleware we found in the literature can provide satisfaction by facilitating the interaction with devices, but they do not support low-level device configuration [4].

2.1. IoT definition

In this section, we explain some of the IoT definitions. Also, we explain the layered architecture for IoT.

Internet of Things (IoT) has increasingly gained attention in industry to interact with different types of devices. IoT can have influence on industry and society by integrating physical devices into information networks [8]. IoT impacts can be on different perspectives, namely for private and business users. From the perspective of a private user, IoT has effect on both working and personal fields, such as smart homes and offices, e-health and assisted living. From the aspect of a business user the impacts would be in fields such as automation and industrial manufacturing, logistics, business process management, intelligent transportation of people and goods [9]

IoT integrates physical things into information networks. IoT covers the overall infrastructure, including software, hardware and services, which is used to support these information networks. The integrated physical things can exchange data about the physical properties and information that they sense in their environment. To identify devices, we can use identification technologies like for example RFID, which allow each device be uniquely identified [10].

International Telecommunication Union (ITU)¹ defines IoT as “*A global infrastructure for the Information Society, enabling advanced services by interconnecting (physical and virtual) things based on, existing and evolving, interoperable information and communication technologies*”[11]

IoT has a layered architecture designed to answer the demands of various industries, enterprises and society. Fig. 2.1 shows a generic layered architecture for IoT that consist of five layers, which are discussed, in the following:[13]

- Edge Technology layer

This is a hardware layer that consists of embedded systems, RFID tags, sensor networks and all of the other sensors in different forms. This hardware layer can perform several functions, such as collecting information from a system or an environment, processing information and supporting communication.

- Access Gateway layer

This layer is concerned with data handling, and is responsible for publishing and

subscribing the services that are provided by the *Things*, message routing, and hovelling the communication between platforms.

- Middleware layer

This layer has some critical functionalities, such as aggregating and filtering the received data from the hardware devices, performing information discovery and providing access control to the devices for applications.

- Application layer

This layer is responsible for delivering various application services. These services are provided through the middleware layer to different applications and users in IoT-based systems. The application services can be used in different industries such as, logistics, retail, healthcare, etc.

CHAPTER 5**SOFTWARE DESIGN****ARDUINO SOFTWARE:****ARDUINO**

Arduino is a tool for making computers that can sense and control more of the physical world than your desktop computer. It's an open-source physical computing platform based on a simple microcontroller board, and a development environment for writing software for the board.

Arduino can be used to develop interactive objects, taking inputs from a variety of switches or sensors, and controlling a variety of lights, motors, and other physical outputs. Arduino projects can be stand-alone, or they can be communicate with software running on your computer (e.g. Flash, Processing, MaxMSP.) The boards can be assembled by hand or purchased preassembled; the open-source IDE can be downloaded for free.

The Arduino programming language is an implementation of Wiring, a similar physical computing platform, which is based on the Processing multimedia programming environment.

Why Ardui no?

There are many other microcontrollers and microcontroller platforms available for physical computing. Parallax Basic Stamp, Netmedia's BX-24, Phidgets, MIT's Handyboard, and many others offer similar functionality. All of these tools take the messy details of microcontroller programming and wrap it up in an easy-to-use package. Arduino also simplifies the process of working with microcontrollers, but it offers some advantage for teachers, students, and interested amateurs over other systems:

Inexpensive - Arduino boards are relatively inexpensive compared to other microcontroller

platforms. The least expensive version of the Arduino module can be assembled by hand, and even the pre-assembled Arduino modules cost less than \$50

Cross-platform - The Arduino software runs on Windows, Macintosh OSX, and Linux operating systems. Most microcontroller systems are limited to Windows.

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

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

PROTEUS

Proteus 8 is a best simulation software for various designs with microcontroller. It is mainly popular because of availability of almost all microcontrollers in it. So it is a handy tool to test programs and embedded designs for electronics hobbyist. Can simulate programming of microcontroller in Proteus 8 Simulation Software. After simulating circuit in Proteus 8 Software can directly make PCB design.

Proteus is a Virtual System Modelling (VSM) that combines circuit simulation, animated components and microprocessor models to co-simulate the complete microcontroller based designs. This is the perfect tool for engineers to test their microcontroller designs before constructing a physical prototype in real time. This program allows users to interact with the design using on-screen indicators and LED and LCD displays and, if attached to the PC, switches and buttons.

Proteus comes with extensive debugging features, single stepping and variable display for a neat design prior to hardware prototyping. Proteus is the program to use when we want to simulate the interaction between software running on a microcontroller and any analog or digital electronic device connected to it.

Proteus was initially created as a multiplatform (DOS, Windows, Unix) system utility, to manipulate text and binary files and to create CGI scripts. The language was later focused on Windows, by adding hundreds of specialized functions for: network and serial communication, database interrogation, system service creation, console applications, keyboard emulation, ISAPI scripting (for IIS). Most of these additional functions are only available in the Windows flavor of the interpreter, even though a Linux version is still available.

Proteus was designed to be practical (easy to use, efficient, complete), readable and consistent.

Its strongest points are:

- powerful string manipulation;
- comprehensibility of Proteus scripts;
- Availability of advanced data structures: arrays, queues (single or double), stacks, bit maps, sets, AVL trees.
- The language can be extended by adding user functions written in Proteus or DLLs created in C/C++.

Proteus has a fully functional, procedural approach; variables are untyped, do not need to be declared, can be local or public and can be passed by value or by reference; all the typical control structures are available (if-then-else; for-next; while-loop; repeat-until; switch-case); new functions can be defined and used as native functions.

Data types supported by Proteus are only three: integer numbers, floating point numbers and strings. Access to advanced data structures (files, arrays, queues, stacks, AVL trees, sets and so on) takes place by using handles, i.e. integer numbers returned by item creation functions.

Type declaration is unnecessary: variable type is determined by the function applied – Proteus converts on the fly every variable when needed and holds previous data renderings, to avoid performance degradation caused by repeated conversions.

There is no need to add parenthesis in expressions to determine the evaluation order, because the language is fully functional (there are no operators).

Proteus includes hundreds of functions for:

- accessing file system;
- sorting data;
- manipulating dates and strings;
- interacting with the user (console functions)
- Calculating logical and mathematical expressions.

Proteus supports associative arrays (called sets) and AVL trees, which are very useful and powerful to quickly sort and lookup values.

The functional approach and the extensive library of built-in functions allow to write very short but powerful scripts; to keep them comprehensible, medium-length keywords were adopted. The user, besides writing new high-level functions in Proteus, can add new functions in C/C++ by following the guidelines and using the templates available in the software development kit; the new functions can be invoked exactly the same way as the predefined ones, passing expressions by value or variables by reference. Proteus is an interpreted language: programs are loaded into memory, pre-compiled and run; since the number of built-in functions is large, execution speed is usually very good and often comparable to that of compiled programs. One of the most interesting features of Proteus is the possibility of running scripts as services or ISAPI scripts. Running a Proteus script as a service, started as soon as the operating system has finished loading, gives many advantages: no user needs to login to start the script; a service can be run with different privileges so that it cannot be stopped by a user. This is very useful to protect critical processes in industrial environments (data collection, device monitoring), or to avoid that the operator inadvertently closes a utility (keyboard emulation). The ISAPI version of Proteus can be used to create scripts run through Internet Information Services and is equipped with specific functions to cooperate with the web server.

For intellectual property protection Proteus provides:

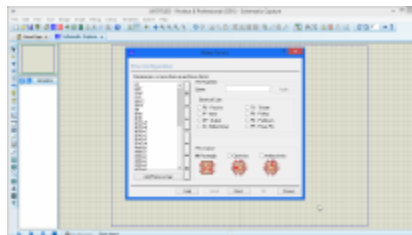
Digital signature of the scripts, by using the development key (which is unique); the option to enable or disable the execution of a script (or part of it) by using the key of the customer. Proteus is appreciated because it is relatively easy to write short, powerful and comprehensible scripts; the large number of built-in functions, together with the examples in the manual, keep low the learning curve.

The development environment includes a source code editor with syntax highlighting and a context-sensitive guide. Proteus does not need to be installed: the interpreter is a single executable (below 400 Kb) that does not require additional DLLs to be run on recent Windows systems.

GET STARTED TO PROTEUS

1. From start menu chose the PROTEUS, and then chose ISIS(blue).
2. To get a part click on Devices (P).
3. Write the name of the PIC16F84A.
4. To get LED write led then chose green led.
5. To get a resistor write 1k and chose the first part. Device Libraries

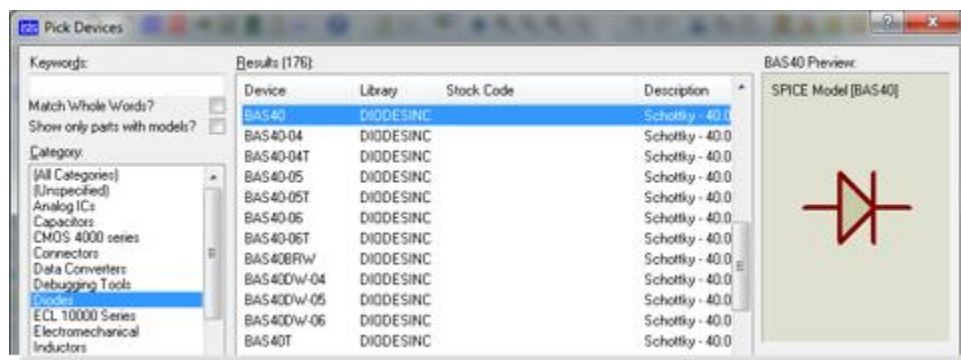
Proteus capture comes with pre-supplied libraries for TTL, CMOS, ECL, Microprocessor, Memory and Analogue ICs plus libraries with hundreds of named Bipolar, FET and Diode discrete semiconductors. Also included are manufacturer specific libraries from National Semiconductor, Philips, Motorols, Teccor, Texas Instruments, Dallas and Zetex amongst others - an expanding total, currently over 10,000 parts.



Importing Components from BSDL

Direct import of BSDL (Boundary Scan Description Library) files provides a semi-automatic way to quickly create and package larger components.

Libraries provided with Proteus also contain default properties for PCB packaging and/or simulator models. Both homogenous and heterogeneous multi-element devices are supported so can place a relay's coil and contacts on separate parts of the drawing. Connector pins can also be distributed. Device editing is performed directly on the drawing using the general editing tools. Devices may be constructed from lines, boxes, circles, arcs, text and special pin objects. Non-electrical symbols may also be created for use in small mechanical or block diagrams.



CHAPTER 6**RESULT AND CONCLUSION**

- As usage of DSP and DIP it will also be possible to transmit higher quality of data such as videos and sound apart from just text or simple bits.
- This type of communication will also reduce the risks off radiation hazards and can be used almost anywhere even at places where electronic devices are banned due to the fear of radiation.

The principle of operation is quite simple. Two dry sensors are used to detect and filter the EEG signals. The sensor tip detects electrical signals from the forehead of the brain. At the same time, the sensor picks up ambient noise generated by human muscle, computers, light bulbs, electrical sockets and other electrical devices. The second sensor, ear clip, is a grounds and reference, which allows Think Gear chip to filter out the electrical noise. The device measures the raw signal, power spectrum (alpha, beta, delta, gamma, theta), attention level, mediation level and blink detection. The raw EEG data received at a rate of 512 Hz. Other measured values are made every second. Therefore, raw EEG data is a main source of information on EEG signals using Mind Wave MW001.

In the Robot section we can read the value of raw EEG signal with the maximum frequency of 512 Hz. Sampling frequency is set on 512 Hz, and we control time delays in sampling. The value of the signal and time are written to the array data. The data which are stored in array will be compared with the threshold points given by the user. In this project, the MATLAB section waits for three consecutive blink in order to send the Robot activation signal. Then based on the attention level value Robot Move Forward Command will be send to the Robot module through ZigBee transmission. After three consecutive blink, the program will scan for a left blink and right blink to turn the Robot right and left respectively.

CONCLUSION

The brain-controlled mobile robots have a great deal of attention because they can help bring mobility back to people with devastating neuromuscular disorders and thus improve their quality of life. A comprehensive of the braincontrolled mobile robot, uses a brain wave sensor which can collect EEG based brain signals of different frequency and amplitude and it will convert these signals into packets and transmit through Wireless medium in to the level splitter section to check the attention level. Level splitter section (LSS) analyses the level and gives the robot movement for the person who is sitting in the wheel chair. The major difference between brain-controlled mobile robots and other brain-controlled devices is that these mobile robots require higher safety because they are used to transport disabled people.

The brain-controlled mobile robots can be applied in practice, including finding ways to improve the performance (especially robustness) of BCI systems, to improve the overall driving performance given the constraints of the BCI system.

REFERENCES

- [1] B. Rebsamen, C. Guan, H. Zhang, C. Wang, C. Teo, M. H. Ang, Jr., and E. Burdet, "A brain controlled wheelchair to navigate in familiar environments," *IEEE Trans. Neural Syst. Rehabil. Eng.*, vol. 18, no. 6, pp. 590–598, Dec. 2010.
- [2] J. d. R. Millan, R. Rupp, G. R. Muller-Putz, R. Murray-Smith, C. Giugliemma, M. Tangermann, C. Vidaurre, F. Cincotti, A. Kubler, R. Leeb, C. Neuper, K.-R. Muller, and D. Mattia, "Combining brain– computer interfaces and assistive technologies state-of-the-art and challenges," *Frontiers Neurosci.*, vol. 4, pp. 1–15, 2010.
- [3] X. Perrin, "Semi-autonomous navigation of an assistive robot using lo throughput interfaces," Ph.D. dissertation, ETHZ, Zurich, Switzerland, 2009.
- [4] A. Nijholt, D. Tan, G. Pfurtscheller, C. Brunner, J. del R. Millan, B. Allison, B. Graimann, F. Popescu, B. Blankertz, and K.-R. Muller, "Brain–computer interfacing for intelligent systems," *IEEE Intell. Syst.*, vol. 23, no. 3, pp. 72–79, May/Jun. 2008.
- [5] J. R. Wolpaw, D. J. McFarland, G. W. Neat, and C. A. Forneris, "An EEG-based brain– computer interface for cursor control," *Electroencephalogr. Clin. Neurophysiol.*, vol. 78, no. 3, pp. 252–259, Mar. 1991.
- [6] Y. Li, C. Wang, H. Zhang, and C. Guan, "An EEGbased BCI system for 2D cursor control," in *Proc. IEEE Int. Joint Conf. Neural Network.*, 2008, pp. 2214–2219.
- [7] E. Donchin, K. M. Spencer, and R. Wijesinghe, "The mental prosthesis: assessing the speed of a P300-based brain– computer interface," *IEEE Trans. Neural Syst. Rehabil. Eng.*, vol. 8, no. 2, pp. 174–179, Jun. 2000.
- [8] N. Birbaumer, N. Ghanayim, T. Hinterberger, I. Iversen, B. Kotchoubey, A. Kubler, J. Perelmouter, E. Taub, and H. Flor, "A spelling device for the paralyzed," *Nature*, vol. 398, pp. 297– 298, Mar. 1999.

- [9] K.-R. Muller and B. Blankertz, "Toward noninvasive brain-computer interfaces," *IEEE Signal Process. Mag.*, vol. 23, no. 5, pp. 125–128, Sep. 2006.
- [10] J. Williamson, R. Murray-Smith, B. Blankertz, M. Krauledat, and K.-R. Muller, "Designing for uncertain, asymmetric control: Interaction design for brain-computer interfaces," *Int. J. Human-Comput. Stud.*, vol. 67, no. 10, pp. 827–841, Oct. 2009.
- [11] Y. Li, H. Li, and C. Guan, "A self-training semi supervised SVM algorithm and its application in an EEG-based brain computer interface speller system," *Pattern Recognition. Lett.*, vol. 29, no. 9, pp. 1285–1294, 2008.
- [12] Y. Su, B. Wu, W. Chen, J. Zhang, J. Jiang, Y. Zhuang, and X. Zheng, "P300-based brain computer interface: Prototype of a Chinese speller," *J. Compute. Inf. Syst.*, vol. 4, no. 4, pp. 1515–1522, 2008.
- [13] B. Hong, F. Guo, T. Liu, X. Gao, and S. Gao, "N200-speller using motion onset visual response," *Clin. Neurophysiology.* vol. 120, no. 9, pp. 1658–1666, Sep. 2009.
- [14] A. A. Karim, T. Hinterberger, and J. Richter, "Neural internet: Web surfing with brain potentials for the completely paralyzed," *Neurorehabil. Neural Repair*, vol. 20, no. 4, pp. 508–515, 2006.
- [15] E. Mugler, M. Bensch, S. Halder, W. Rosenstiel, M. Bogdan, N. Birbaumer, and A. Kubler, "Control of an Internet browser using the P300 event-related potential," *Int. J. Bioelectromagnetic*, vol. 10, no. 1, pp. 56–63, 2008.
- [16] M. Bensch, A. A. Karim, J. Mellinger, T. Hinterberger, M. Tangermann, M. Bogdan, W. Rosenstiel, and N. N. Birbaumer, "An EEG controlled web browser for severely paralyzed patients," *Comput. Intell. Neurosci.*, vol. 2007, pp. 1–5, 2007.

- [17] R. Krepki, B. Blankertz, G. Curio, and K.-R. Müller, “The Berlin brain– computer interface (BCI): Towards a new communication channel for online control in gaming applications,” J. Multimedia ToolsAppl., vol. 33, no. 1, pp. 73–90, Apr. 2007.

- [18] EEG’s Signal Processing and Emotiv’s Neuro Headset by Andre Hoffmann
<http://data.text20.net/documentation/thesis.emotivsp.pdf>

- [19] Arduino – Arduino BoardNano
<http://arduino.cc/en/Main/ArduinoBoardNano>