

TRAFFIC CONTROL AND AMBULANCE MONITORING SYSTEM

Submitted in partial fulfillment of the requirements for the award of

**Bachelor of Engineering Degree in
Electronics and Communication Engineering**

BY

CHANDANA MURALI (38130037)

VARSHITHA BEZAWADA (38130030)



**DEPARTMENT OF ELECTRONICS AND COMMUNICATION
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

APRIL – 2022



SATHYABAMA

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

Accredited with Grade "A" by NAAC | 12B Status by UGC | Approved by AICTE
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 **CHANDANA MURALI (38130037)**, and **VARSHITHA BEZAWADA (38130030)** who carried out the project entitled "**TRAFFIC CONTROL AND AMBULANCE MONITORING SYSTEM**" under my supervision from November 2021 to April 2022.

Internal Guide

Dr. V. VIJAYA BASKAR, M.E., Ph.D.

Head of the Department

Dr. T. RAVI., M.E., Ph.D.,

Submitted for Viva voce Examination held on _____

Internal Examiner

External Examiner

DECLARATION

We, **CHANDANA MURALI (38130037)**, and **VARSHITHA BEZAWADA (38130030)** hereby declare that the Project Report entitled “**TRAFFIC CONTROL AND AMBULANCE MONITORING SYATEM**” done by us under the guidance of **Dr. V. VIJAYA BASKAR, M.E., Ph.D.** at Sathyabama Institute of Science and Technology is submitted in partial fulfilment of the requirements for the award of Bachelor of Engineering degree in Electronics and Communication Engineering.

DATE:

1. CHANDANA MURALI

PLACE:

2. BEZAWADA VARSHITHA

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 my thanks to **Dr. N.M. NANDHITHA, M.E., Ph.D.**, Dean, School of Electrical and Electronics Engineering and **Dr. T. RAVI., M.E., Ph.D.**, Head of the Department, Dept. of Electronics and Telecommunication Engineering for providing me necessary support and details at the right time during the progressive reviews.

We would like to express my sincere and deep sense of gratitude to my Project Guide **Dr. V. VIJAYA BASKAR, M.E., Ph.D.**, for his valuable guidance, suggestions and constant encouragement paved way for the successful completion of my project work.

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

ABSTRACT

Road traffic congestion becomes a major issue for highly crowded metropolitan cities. India is the second most populated country in the world and is a fast growing economy. It is facing terrible road congestion in the cities. According to Times of India about 30 percent of deaths are caused due to delayed ambulance to reach at hospital. In proposed system we are trying to reduce the delay for the ambulance. To smoothen the ambulance movement we come up with “Traffic Control And Ambulance Monitoring System”

Congestion in traffic is a serious issue. In existing system signal timings are fixed and they are independent of traffic density. Large red light delays leads to traffic congestion.

This IOT based traffic control system is implemented in which signal timings are updated based on transceiver module it transmits the message of the current system to the traffic signal. There will be a screen displayed and a blue light will be shown which indicates the arrival of ambulance which can be controlled by the driver.

TABLE OF CONTENTS

CHAPTER NO	TITLE	PAGE NO
	ABSTRACT	v
	LIST OF FIGURES	viii
	LIST OF ABBREVIATIONS	ix
1	INTRODUCTION	1
	1.1 TRAFFIC MANAGEMENT AND CONGESTION	2
	1.2 PROBLEM DESCRIPTION	2
2.	LITERATURE SURVEY	4
3.	AIM AND SCOPE	6
	3.1 AIM	6
	3.2 SCOPE	6
4	EXISTING SYSTEM	7
5	PROPOSED SYSTEM	8
	5.1 ADVANTAGES	8
	5.2 WORKING	10
6	HARDWARE COMPONENTS	11
	6.1 SYSTEM DESIGN NODE MCU	11
	6.1.1 ESP8266 Arduino Core	12
	6.1.2 Role of NodeMCU	13
	6.1.3 NodeMCU Development Board/kit v0.9 (Version1)	13
	6.1.4 NodeMCU Development Board/kit v1.0 (Version2)	14
	6.1.5 NodeMCU with ESPlorer IDE	14
	6.1.6 NodeMCU with Arduino IDE	15
	6.2 LCD DISPLAY	15

CHAPTER NO	TITLE	PAGE NO
6.3	ARDUINO UNO AND ITS PROGRAMMING	17
6.3.1	Working of Arduino	18
6.3.2	Role of Arduino in this Project	19
6.3.3	Arduino Summary	20
7	SOFTWARE REQUIRED	21
7.1	ARDUINO IDE	21
7.1.1	Arduino Software	23
7.2	ZIGBEE IEEE 802.15.4	26
7.2.1	Mesh Networks	27
7.2.2	ZigBee Applications	28
7.2.3	Digi ZigBee Technology	28
8	IoT	29
8.1	FUTURE OF IoT	29
8.2	MOTIVATION	30
8.3	IoT DEFINITION	30
8.4	CHALLENGES	32
8.5	IoT BASED MIDDLEWARE	34
8.6	INTERFACE PROTOCOLS	35
8.7	DEVICE ABSTRACTION (DA)	35
8.8	CENTRAL CONTROL, CONTEXT DETECTION & MANAGEMENT (CCM)	37
8.9	APPLICATION ABSTRACTION	37
8.10	CONCLUSION	38
	8.10.1 Review of IoT-based Middleware	38
9.	CONCLUSION	40
9.1	SCREENSHOTS OF OUTPUT	41
	REFERENCES	44

LIST OF FIGURES

FIGURE NO	TITLE	PAGE NO
1.1	Traffic Congestion	2
4.1	Ambulance in traffic	7
5.1	Block Diagram Drivers side (transmitter side)	9
5.2	Block Diagram Traffic Signal Side (Receiver Side)	9
5.3	Working (Flowchart)	10
6.1	ESP8266 Arduino Core	12
6.2	NodeMCU	13
6.3	Working of LCD	16
6.4	LCD showing ambulance approach	16
6.5	Arduino uno	17
6.6	Arduino uno board	18
7.1	Arduino software	21
7.2	Arduino	22
7.3	USB Cable	22
7.4	Arduino IDE page	23
7.5	Selection of COM Port from tools	24
7.6	Required Arduino board selection	24
7.7	Sketch of Arduino IDE	25
7.8	Compilation	25
7.9	Pin diagram	26
8.1	Layered architecture of the Internet of Things (IoT)	32
8.2	Functional components of a middleware for IoT-based Systems	34
9.1	Circuit diagram-Arrival of ambulance	41
9.2	Circuit diagram-Departure of ambulance	41
9.3	Traffic signals with LED	42
9.4	Receiver side circuit	42
9.5	Transmitter side circuit	43

LIST OF ABBREVIATIONS

AI	-	Artificial Intelligence
AMR	-	Automatic Meter Reading
DA	-	Device Abstraction
DSSS	-	Direct Sequence Spread Spectrum
ETSI	-	European Telecommunications Standards Institute
IDE	-	Integrated Development Environment
IEEE	-	Institute of Electrical and Electronics Engineers
IoT	-	Internet of Things
ITU	-	International Telecommunication Union
TLS	-	Traffic Light System
WPANs	-	Wireless Personal Area Networks
WSN	-	Wireless Sensor Network

CHAPTER 1

INTRODUCTION

Human life is more valuable than anything else, timely help is more important than lending a helping hand. This project is one among those which is designed in a way to save human lives in a timely manner. The traffic system acts as a cornerstone to world's economy. With the increasing rate of population, the numbers of vehicles on roads are increasing simultaneously. This results in increase of rate of traffic jams on daily basis Congestion in traffic is a serious issue. In existing system signal timings are fixed and they are independent of traffic density.

Thus we propose a new design which automatically communicates with the traffic signals, and help the ambulance to pass through the way. The proposed system is capable to switch the traffic signal to blue, when the vehicle approaches the traffic signal. In this proposed, IOT based traffic control system is implemented in which signal timings are updated based on the vehicle counting.

This system consists of ZigBee transceiver module it transmits the data to the traffic signal. The emergency vehicles such as ambulances, fire engines and police vehicles are given equal priority over other vehicles and hence get stuck up in this traffic congestion. In an emergency every minute counts. But in urban areas, ambulances, and other rescue vehicles, often have difficulty making headway due to dense traffic or construction sites. When using sirens and flashing lights, the rescue teams do not have to stop at red lights; however, this can be dangerous for other road users at the intersection. A methodology for priority based vehicle detection based on embedded techniques is proposed in this model. If an emergency vehicle is arriving on the road, the lane in which this vehicle is will be given higher priority over all other lanes.

1.1 TRAFFIC MANAGEMENT AND CONGESTION

It has become very easy for a common person to own a vehicle with affordability and higher purchasing power. It creates a problem in terms of road congestion and increasing traffic in big cities though this has led to comfortable lifestyles. Smart traffic management system is an alternative way to solve road traffic problem and this system will support existing operation system. Traffic congestion is a severe problem in almost every modern cities around the world. Traffic congestion has been causing many critical problems and challenges in the major and most populated cities. It is becoming more difficult and time consuming to travel to different places within the city. Due to these congestion problems, people lose time, miss opportunities, and get frustrated. Due to traffic congestion, there is a loss in productivity from workers, trade opportunities are lost, delivery gets delayed, and thereby the costs goes on increasing.



Fig. 1.1: Traffic Congestion

1.2 PROBLEM DESCRIPTION

It has become very easy for common man to own a vehicle with affordability and higher purchasing power. It creates a problem in terms of road

congestion and increasing traffic in big cities though this has lead to comfortable lifestyles .This becomes a problem for the emergency vehicles like ambulances to reach its destination on time. Traffic control and ambulance monitoring system is an alternative way to solve road traffic problem related to ambulance services and this system will support existing operation system.

Traffic congestion is a severe problem in almost every modern cities around the world. Traffic congestion has been causing many critical problems and challenges in the major and most populated cities. It is becoming more difficult and time-consuming to travel to different places within the city. Due to these problems of heavy traffic, emergency vehicles remain stuck in the traffic and hence cannot reach their destination in time this will lead to loss of lives of people. According to Times of India about 146,133 people were killed in road accidents in India in the year 2016. Unfortunately about 30% of deaths are caused due to delayed ambulance.

CHAPTER 2

LITERATURE SURVEY

This Chapter discuss about the literature survey of different existing technologies. There are various researches on the technology of object detection as well as of navigating the path for the visually impaired persons. The discussion about the five literature surveys are given below.

Title: Intelligent Traffic Information System Based on Integration of Internet of Things and Agent

Author: Hasan Omar Al-Sakran(2015)

Explanation: The Internet of Things is based on the Internet, network wireless sensing and detection technologies to realize the intelligent recognition on the tagged traffic object, tracking, monitoring, managing and processed automatically.

Drawbacks: Hard to implementation, Less security, Take more time to execute.

Title: Traffic Monitoring and Reporting System Using IOT

Author: Chiranjeev V. Tapas 1, Narendra Chaudhar.(2017)

Explanation: Although many traffic laws are introduced, we still don't see any fall in traffic rule violations or death caused by such negligence. The factors causing such failure can either be people or the system.

Drawbacks: No data base management, Not economical.

Title: A Review of Traffic Management System Using IoT

Author: Nikita Tendulkar, Komal Sonawane, Darshana vakt. (2018)

Explanation: We can improve the traffic efficiency. In this paper we describe the things using Internet to control the traffic well.

Drawbacks: Traffic congestion is occurred, Not reliable.

Title: Intelligent Traffic Monitoring Using Internet of Things (IoT) with Semantic Web

Author: Manuj Darbarir, Diwakar Yagyasen, Anurag Tiwari (2020)

Explanation: The paper highlights the issue of Intelligent Traffic monitoring system using the technologies like IoT, MultiAgent system and Semantic Web.

Drawbacks: Zig bee protocol and traffic movements are continuously giving errors data.

Title: Development of IoT device for traffic management system

Author: Hon Fong Chong; Danny Wee Kiat Ng (2016)

Explanation: Fixed cycle traffic light system (TLS) is first introduced on road intersection to solve traffic congestion.

Drawbacks: Increase queue length and waiting time on the road. 12

Title: An IoT framework for intelligent vehicle monitoring system Sign In or Purchase

Author: Rahul B. Pendor, P. P. Tasgaonkar.(2016)

Explanation: In this paper we present a design and development of low cost and reliable Internet of things framework which consists of an array of RFID sensors for the real time tracking.

Drawbacks: Resultant data are not valued, Less Accuracy

CHAPTER 3

AIM AND SCOPE

3.1 AIM

The AIM of this project is save lives of people who are dying due to the congestion of traffic on road in ambulance. This helps the travelling of ambulance through the traffic in very smooth way and in less cost.

3.2 SCOPE

India is a populated country, like people the number of vehicles are also high. The traffic in Indian cities are not less. And accidents and other severe health problems like heart attacks are also increasing day by day. So it is necessary that the hospitals or public ambulance services (108) are always necessary for the help of people. And there are a lot of cases that people are dying in ambulance due to the traffic congestion. It is necessary that traffic should be controlled in such a way that the ambulance can pass through the signals smoothly. Due to this there is a major scope for this project so that the lives will b saved and the path of ambulance will be cleared.

CHAPTER 4

EXISTING SYSTEM

Congestion on roads eventually results in slow moving traffic, which increases the time of travel, thus stands out as one of the major issues in metropolitan cities. The existing system was used to provide clearance to any emergency vehicle by turning all the red lights to green on the path of the emergency vehicle, hence providing a complete green wave to the desired vehicle. In the existing system there is no screen which displays that emergency vehicle is passing through and the blue light also will not be displayed. Signal timings are fixed and they are independent of traffic density. Large red light delays leads to traffic congestion. The biggest disadvantage of green waves is that, when the wave is disturbed, the disturbance can cause traffic problems that can be exacerbated by the synchronization.



Fig. 4.1: Ambulance in traffic

CHAPTER 5

PROPOSED SYSTEM

This system discusses the IOT based traffic control system in which signal timings are updated based on the vehicle density on the road. The proposed system is implemented using Nodemcu and Arduino. Proposed system gives good performance and helps to reduce traffic congestion. Results shows vehicle counting in real time.

The proposed system is suitable for real time vehicle management. This proposed system a priority based traffic control system using embedded technology which will give high priority to all emergency vehicles thereby helping them to pass by the traffic light smoothly. We are gonna implement a concept in which that when ambulance starts going in roads all the vehicles will leave properly to the ambulance. For this in traffic signal an additional blue signal will be emerged. If the emergency vehicle driver press the emergency button it will send information to the traffic signal blue light will turn on, which leads to co-operate and leave way to the ambulance.

5.1 ADVANTAGES

A quick medical support for the accident victims can be given. Traffic chaos can be controlled. Other vehicles on the road can know that, there is an emergency with the help of display screen and blue light. Travelling to nearby hospital can be made easy.

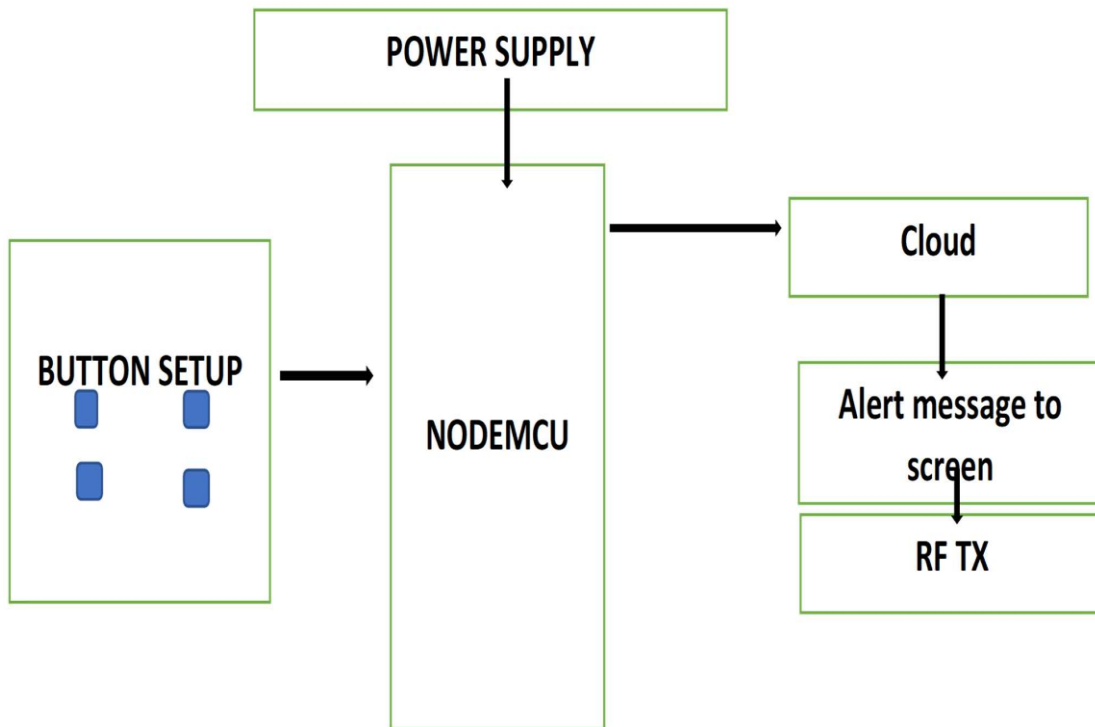


Fig. 5.1: Block Diagram Drivers side (transmitter side)

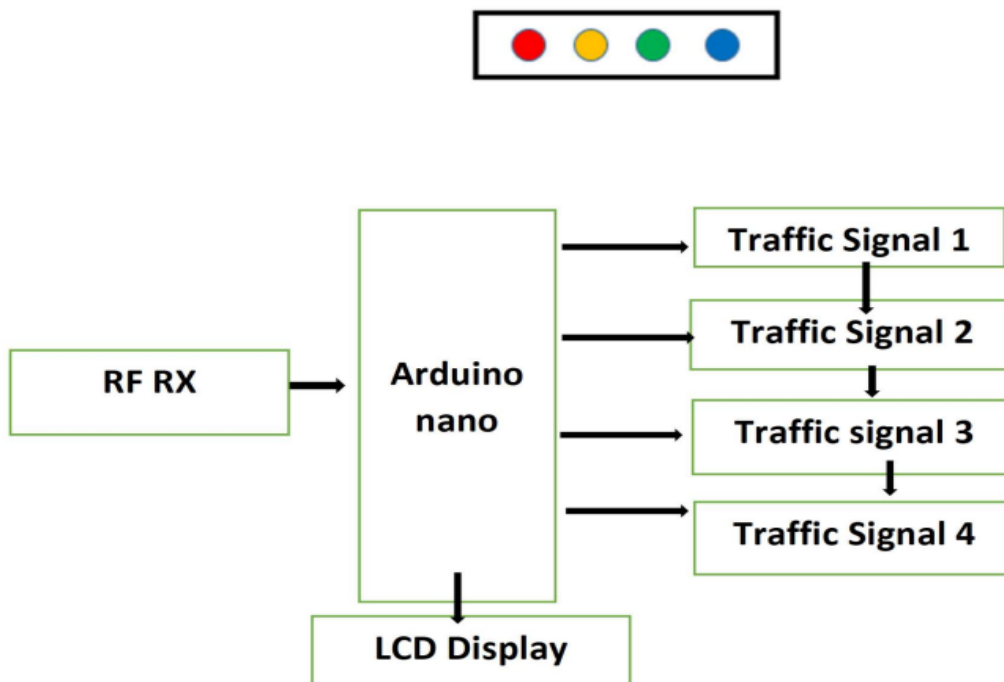


Fig. 5.2: Block Diagram Traffic Signal Side (Receiver Side)

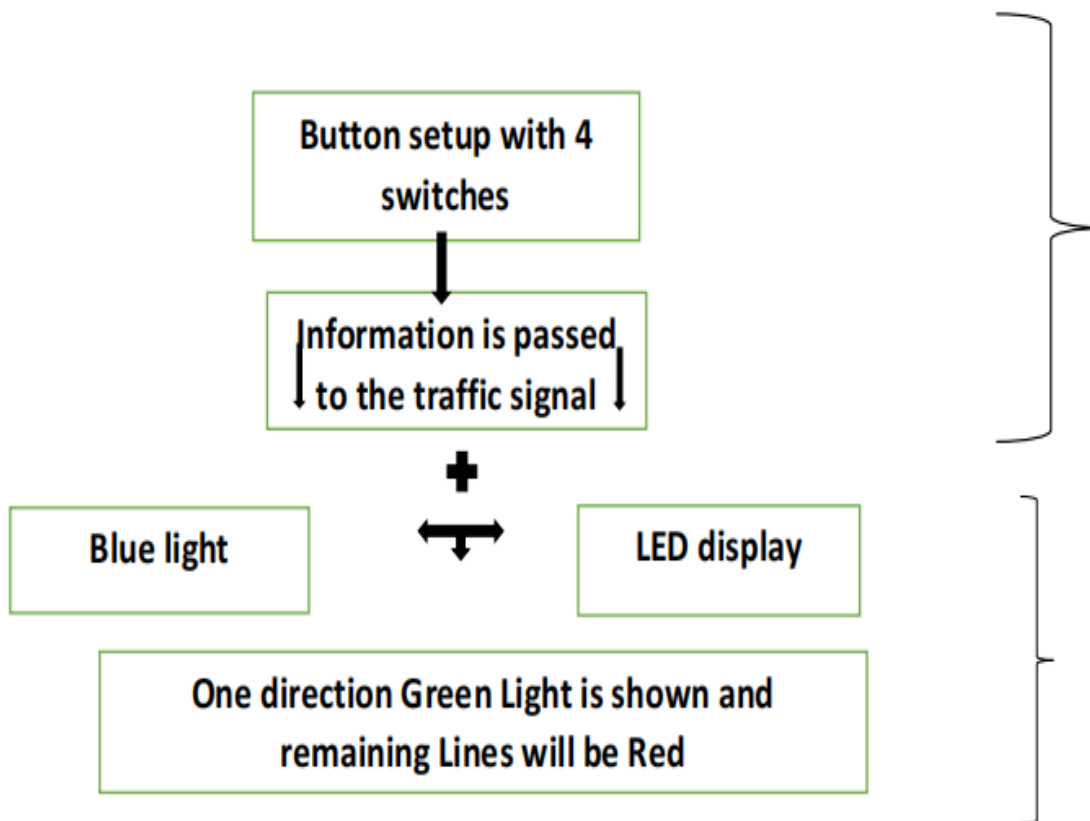


Fig. 5.3: Working (Flowchart)

5.2 WORKING

In this Project the main objective is to save the lives of people, by controlling the traffic signal to reduce the traffic congestion. In this we have a receiver side and a transmitter side through which the signal is passed.

In transmitter side is the driver's side which has the four switches which control the four sides of traffic signals, a power supply ,and through cloud the alert message will be send to the receiver side using nodeMUC . In Transmitter side that is the signal side, the Aurduino nano will be connected to four sides and an extra blue light will be also there which will indicate the presence of ambulance along with the LCD screen display which is connected along with the lights to the Arduino board.

CHAPTER 6

HARDWARE COMPONENTS

Traffic light board

TX and RX

Nodemcu

Arduino nano

Button setup

6.1 SYSTEM DESIGN NODE MCU

NodeMCU

It is an open source IoT platform. It includes firmware which runs on the ESP8266 Wi Fi SoC from Espressif Systems, and hardware which is based on the ESP-12 module. The term "NodeMCU" by default refers to the firmware rather than the development kits. The firmware uses the Lua scripting language. It is based on the eLua project, and built on the Espressif Non-OS SDK for ESP8266. It uses many open source projects, such as lua-cjson and SPIFFS. NodeMCU was created shortly after the ESP8266 came out. On December 30, 2013, Espressif Systems began production of the ESP8266. The ESP8266 is a Wi-Fi SoC integrated with a Tensilica Xtensa LX106 core,[*citation needed*] widely used in IoT applications (see related projects). NodeMCU started on 13 Oct 2014, when Hong committed the first file of nodemcu-firmware to GitHub. Two months later, the project expanded to include an open-hardware platform when developer Huang R committed the gerber file of an ESP8266 board, named devkit v0.9. Later that month, Tuan PM ported MQTT client library from Contiki to the ESP8266 SoC platform, and committed to NodeMCU project, then NodeMCU was able to support the MQTT IoT protocol, using Lua to access the MQTT broker.

Another important update was made on 30 Jan 2015, when Devsaurus ported the uglib to NodeMCU project, enabling NodeMCU to easily drive LCD, Screen, OLED, even VGA displays. In summer 2015 the creators abandoned the firmware project and a group of independent contributors took over. By summer 2016 the NodeMCU included more than 40 different modules. Due to resource constraints users need to select the modules relevant for their project and build a firmware tailored to their needs.

6.1.1 ESP8266 Arduino Core

As 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 tool chains to allow Arduino C/C++ to be compiled for these new processors. They did this with the introduction of the Board Manager and the SAM Core. A "core" is the collection of software components required by the Board Manager and the Arduino IDE to compile an Arduino C/C++ source file for the target MCU's machine language. Some ESP8266 enthusiasts developed an Arduino core for the ESP8266 WiFi SoC, popularly called the "ESP8266 Core for the Arduino IDE". This has become a leading software development platform for the various ESP8266-based modules and development boards, including NodeMCUs. NodeMCU is an open source LUA based firmware developed for ESP8266 wifi chip. By exploring functionality with ESP8266 chip, NodeMCU firmware comes with ESP8266 Development board/kit i.e. NodeMCU Development board.

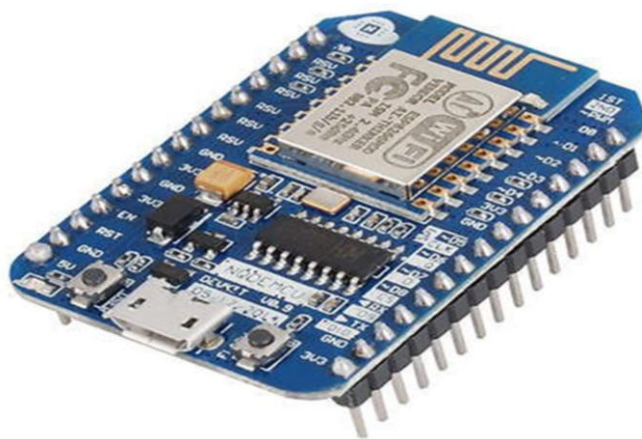


Fig. 6.1: ESP8266 Arduino Core

6.1.2 *Role of NodeMCU*

It is technically referred as ESP8266 which is used for providing WI-Fi connectivity to the entire Module. It is connected to control the traffic signal by the ambulance driver when the ambulance is almost near the traffic signal which will prevent the traffic congestion and the death of patient.

6.1.3 *NodeMCU Development Board/kit v0.9 (Version1)*

Since NodeMCU is open source platform, their hardware design is open for edit/modify/build. NodeMCU Dev Kit/board consist of ESP8266 wifi enabled chip. The **ESP8266** is a lowcost Wi-Fi chip developed by Espressif Systems with TCP/IP protocol. For more information about ESP8266, you can refer 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.



Fig. 6.2: NodeMCU

6.1.4 *NodeMCU Development Board/kit v1.0 (Version2)*

For more information about NodeMCU Boards available in market refer 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 start with using NodeMCU for IoT applications first we need to know about how to write/download NodeMCU firmware in NodeMCU Development Boards. And before that where this NodeMCU firmware will get as per our requirement. There is online NodeMCU custom builds available using which we can easily get our custom NodeMCU firmware as per our requirement. To know more about how to build custom NodeMCU firmware online and download it refer 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 development of NodeMCU.

6.1.5 *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 Getting started with NodeMCU using ESPlorerIDE

6.1.6 *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 Arduino development environment. This makes easy for Arduino developers than learning new language and IDE for NodeMCU. For more information about how to write Arduino sketch for NodeMCU refer Getting started with NodeMCU using Arduino IDE. Difference in using ESPlorer and Arduino IDE Well, there is a programming language difference we can say while developing 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. 22 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 executes sequentially. It will not build binary firmware file of code for NodeMCU to write. It will send Lua script as it is to NodeMCU to get execute. In Arduino IDE when we write and compile code, ESP8266 toolchain in background creates 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 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 compile and upload/writes complete firmware, it takes more time than ESPlorer IDE.

6.2 LCD DISPLAY

A liquid Cristal Display is a flat panel display or other electronically modulated optical device that uses light modulating properties of liquid. LCD is the combination of two states of matter, the solid and the liquid, the solid part is crystal and the liquid and the crystal part together makes the visible image. LCD consist of

two layers which are two polarized panels – filters and electrodes. LCD screen works by blocking the light rather than emitting the light.

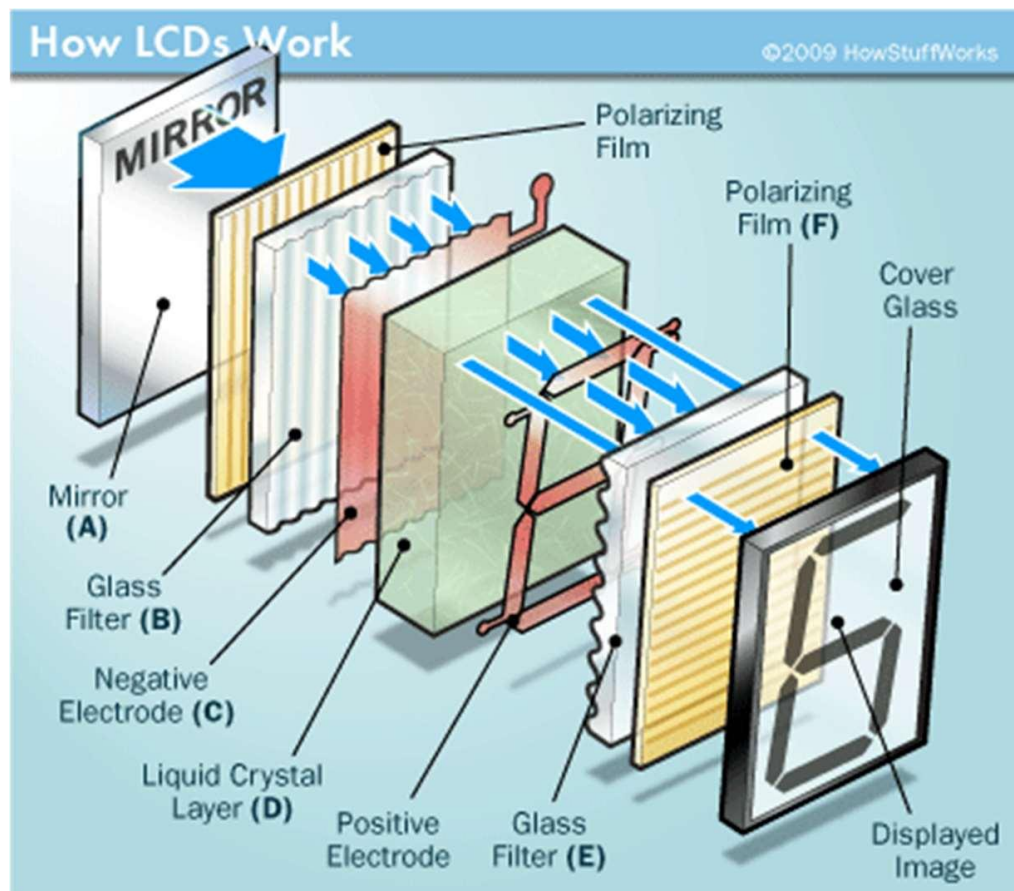


Fig. 6.3: Working of LCD



Fig. 6.4: LCD showing ambulance approach

6.3 ARDUINO UNO 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 opensource 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.



Fig. 6.5: Arduino uno

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 This is what the Arduino board looks like.

6.3.1 Working of Arduino

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.



Fig. 6.6: Arduino uno board

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 Uno 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 Uno 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.

6.3.2 *Role of Arduino in this Project*

We have used one Arduino MEGA where all the four LEDs (as signals) and eight servo motors (a pair for each road) acts as barricades connected to it to handle the traffic in a road junction. Let us consider a four number of roads (R1, R2, R3 and R4) to form a road junction and blue light (B1) indicates the arrival of ambulance. Now, for road R1, When the signal for road R1 becomes green, it indicates that R1 can go through the signal junction and the signals for roads R2, R3 and R4 becomes red so that no vehicles can break the signal and also it will

help to reduce traffic congestion and number of accidents. Now, after a particular time, the signal of road R1 turns yellow and then red so that the road will be closed and again no vehicles can break the signal. Also, signal for road R2 becomes green and that road will be opened. This operation will be continued for all the remaining roads according to the arrival of ambulances from different lane.

6.3.3 *Arduino 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

CHAPTER 7

SOFTWARE REQUIRED

7.1 ARDUINO IDE

Introduction

This tutorial will walk you through downloading, installing, and testing the Arduino software (also known as the Arduino IDE - short for Integrated Development Environment). Before you jump to the page for your operating system, make sure you've got all the right equipment.



Fig. 7.1: Arduino software

Required Materials

To follow along with this tutorial, you will need the following materials. You may not need everything though depending on what you have. Add it to your cart, read through the guide, and adjust the cart as necessary.

- A computer (Windows, Mac, or Linux)
- An Arduino-compatible microcontroller (anything from this guide should work)

- A USB A-to-B cable, or another appropriate way to connect your Arduino compatible microcontroller to your computer (check out this [USB buying guide](#) if you're not sure which cable to get)



Fig. 7.2: Arduino



Fig. 7.3: USB Cable

7.1.1 *Arduino Software*

1. *Open Arduino IDE as shown below*

Open Arduino IDE

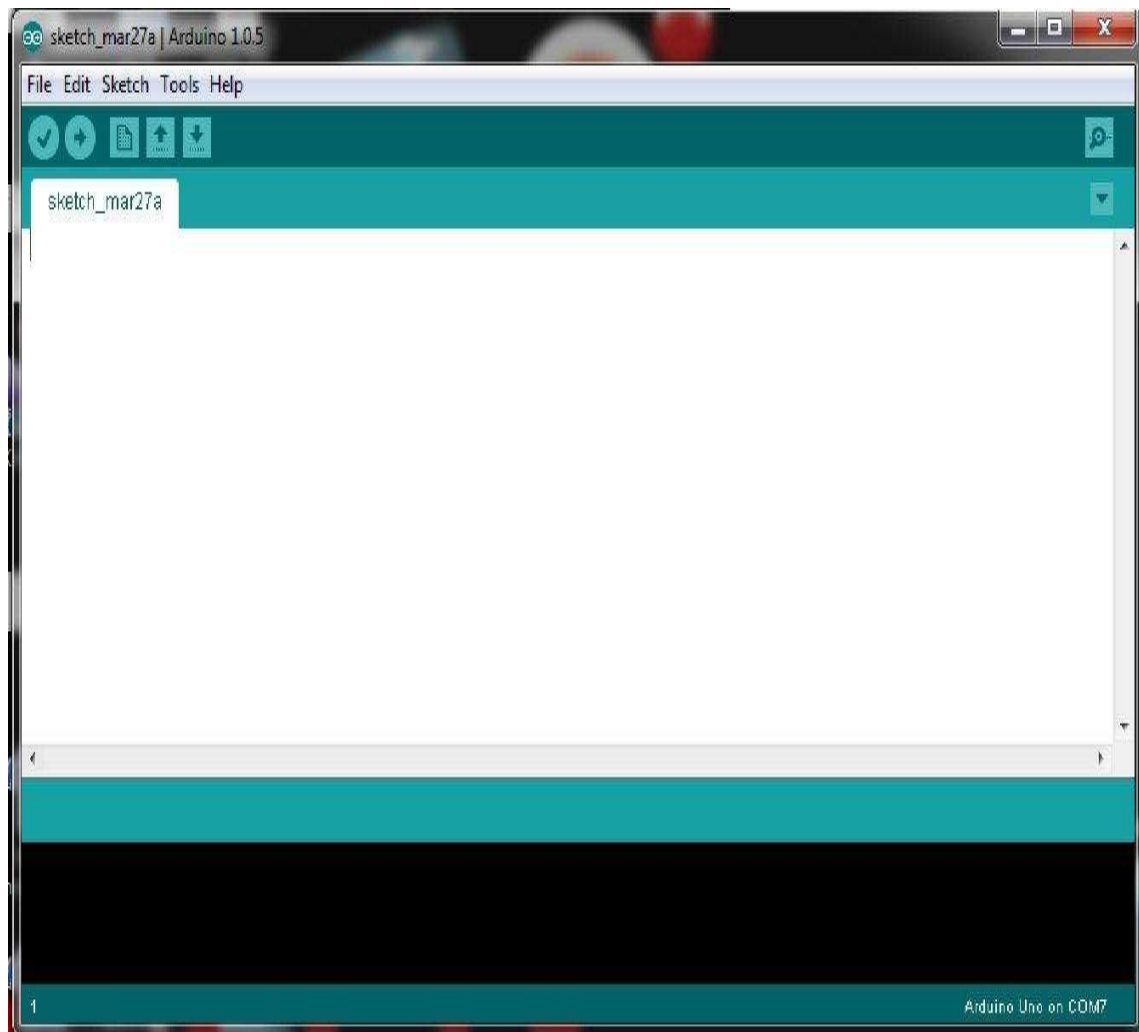


Fig. 7.4: Arduino IDE page

2. *Select the COM Port from tool*

Select the COM Port

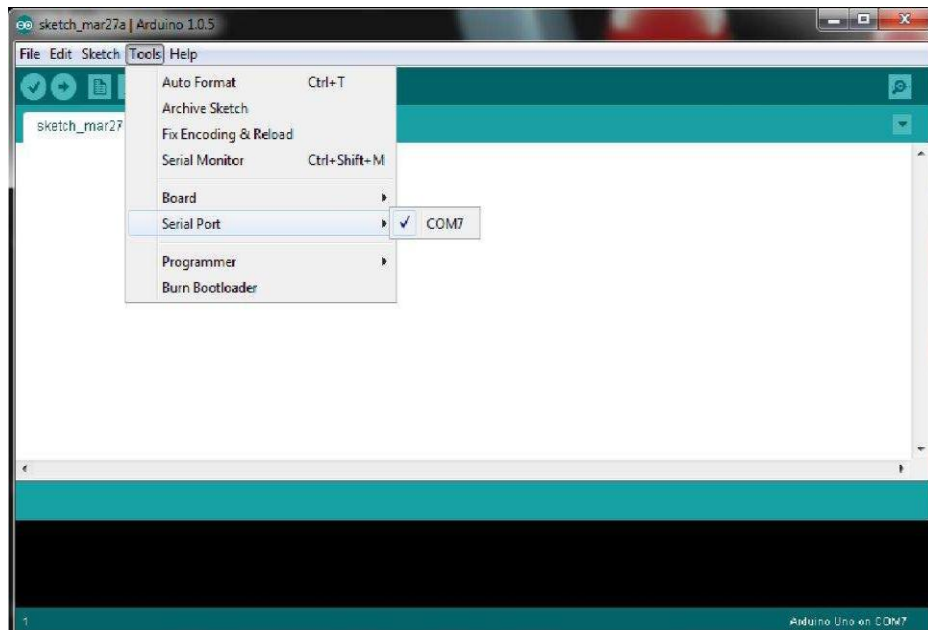


Fig. 7.5: Selection of COM Port from tools

3. *Select the required Arduino board from Tools as shown below*

Select the required Arduino board

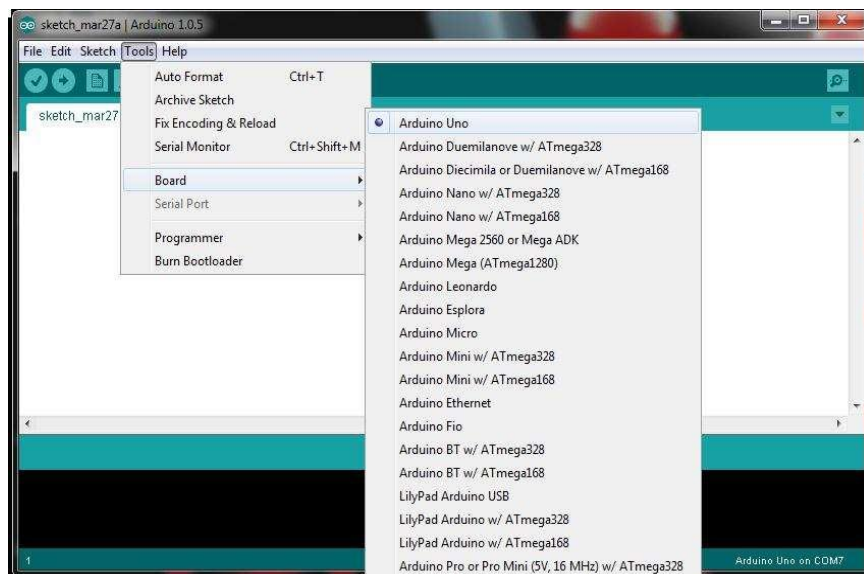


Fig. 7.6: Required Arduino board selection

4. *Write the sketch in Arduino IDE*

Sketch in Arduino IDE

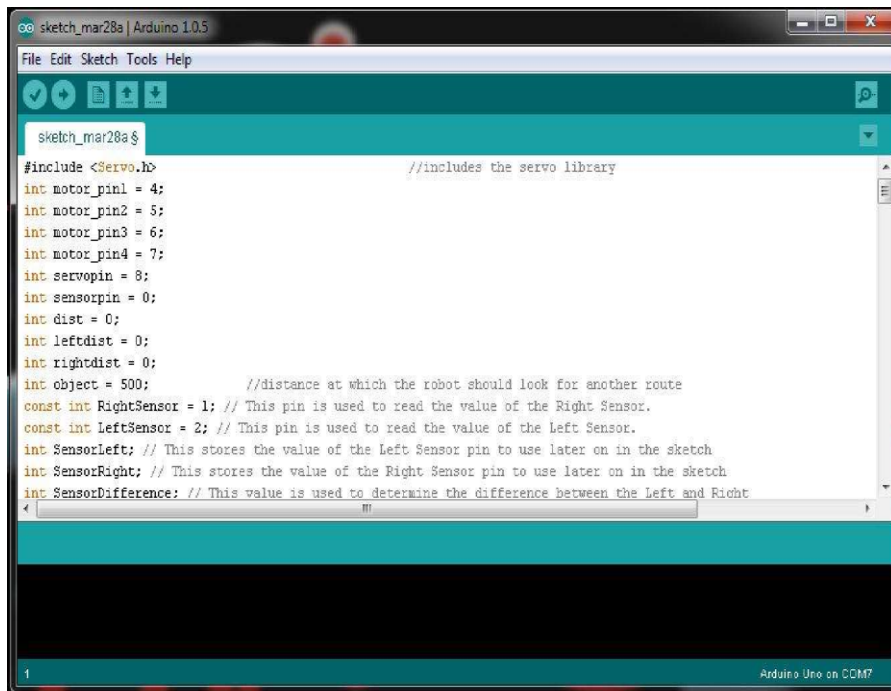


Fig. 7.7: Sketch of Arduino IDE

5. *Compile and upload the Sketch to Arduino board upload the Sketch to Arduino Board*

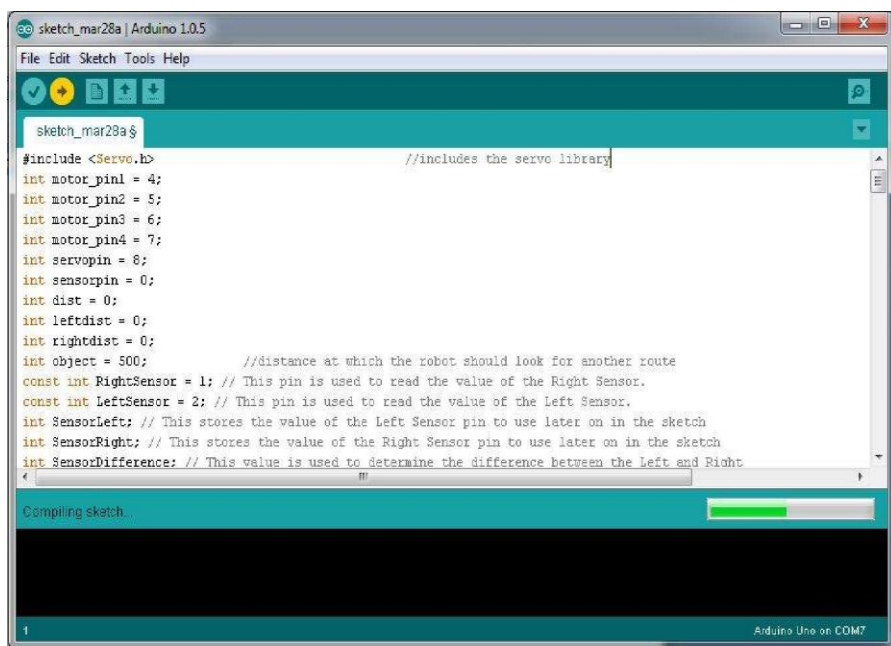


Fig. 7.8: Compilation

7.2 ZIGBEE IEEE 802.15.4

ZigBee is a wireless technology developed as an open global standard 802.15.4 – ZigBee Physical Layer to address the unique needs of low-cost, low-power wireless M2M networks. The ZigBee standard operates on the IEEE 802.15.4 physical radio specification and operates in unlicensed bands including 2.4 GHz, 900 MHz and 868 MHz. The 802.15.4 specification upon which the ZigBee stack operates gained ratification by the Institute of Electrical and Electronics Engineers (IEEE) in 2003. The specification is a packet-based radio protocol intended for low-cost, battery operated devices. The protocol allows devices to communicate in a variety of network topologies and can have battery life lasting several years.

The ZigBee Protocol

The ZigBee protocol has been created and ratified by member companies of the ZigBee Alliance. Over 300 leading semiconductor manufacturers, technology firms, OEMs and service companies comprise the ZigBee Alliance membership. The ZigBee protocol was designed to provide an easy-to-use wireless data solution characterized by secure, reliable wireless network architectures.

The ZigBee Advantage

The ZigBee protocol is designed to communicate data through hostile RF environments that are common in commercial and industrial applications.

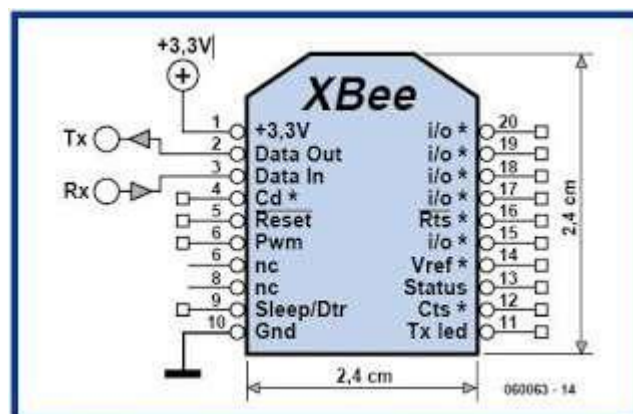


Fig. 7.9: Pin diagram

ZigBee is a specification for a suite of high-level communication protocols used to create personal area networks built from small, low-power digital radios. ZigBee is based on an IEEE 802.15.4 standard. Though its low power consumption limits transmission distances to 10–100 meters line-of-sight, depending on power output and environmental characteristics,[1] ZigBee devices can transmit data over long distances by passing data through a mesh network of intermediate devices to reach more distant ones. ZigBee is typically used in low data rate applications that require long battery life and secure networking (ZigBee networks are secured by 128 bit symmetric encryption keys.) ZigBee has a defined rate of 250 kbit/s, best suited for intermittent data transmissions from a sensor or input device.

Applications include wireless light switches, electrical meters with in-home displays, traffic management systems, and other consumer and industrial equipment that requires short-range low-rate wireless data transfer. The technology defined by the ZigBee specification is intended to be simpler and less expensive than other wireless personal area networks (WPANs), such as Bluetooth or Wi-Fi.

ZigBee protocol features include ;

- Support for multiple network topologies such as point-to-point, point-to-multipoint and mesh networks
- Low duty cycle – provides long battery life
- Low latency
- Direct Sequence Spread Spectrum (DSSS)
- Up to 65,000 nodes per network
- 128-bit AES encryption for secure data connections
- Collision avoidance, retries and acknowledgements

7.2.1 Mesh Networks

A key component of the ZigBee protocol is the ability to support mesh networking. In a mesh network, nodes are interconnected with other nodes so that

multiple pathways connect each node. Connections between nodes are dynamically updated and optimized through sophisticated, built-in mesh routing table.

Mesh networks are decentralized in nature; each node is capable of self-discovery on the network. Also, as nodes leave the network, the mesh topology allows the nodes to reconfigure routing paths based on the new network structure. The characteristics of mesh topology and ad-hoc routing provide greater stability in changing conditions or failure at single nodes.

7.2.2 *ZigBee Applications*

ZigBee enables broad-based deployment of wireless networks with low-cost, low power solutions. It provides the ability to run for years on inexpensive batteries for a host of monitoring and control applications. Smart energy/smart grid, AMR (Automatic Meter Reading), lighting controls, building automation systems, tank monitoring, HVAC control, medical devices and fleet applications are just some of the many spaces where ZigBee technology is making significant advancements.

7.2.3 *Digi ZigBee Technology*

Digi is a member of the ZigBee Alliance and has developed a wide range of networking solutions based on the ZigBee protocol. XBee and XBee-PRO modules and other XBee-enabled devices provide an easy-to-implement solution that provides functionality to connect to a wide variety of devices.

CHAPTER 8

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.

8.1 FUTURE OF IoT

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.

8.2 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 user. The idea of creating abstractions of devices has 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.

8.3 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. 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. 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.

International Telecommunication Union (ITU) 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” Internet of Things-Architecture 2 (IoT-A) defines it as “The idea of a globally interconnected continuum of devices, objects and things in general emerged with the RFID technology, and this concept has considerably been extended to the current vision that envisages a plethora of heterogeneous objects interacting with the physical environment.”

IoT has a layered architecture designed to answer the demands of various industries, enterprises and society. Fig. 6.1 shows a generic layered architecture for IoT that consist of five layers, which are discussed, in the following:

- *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.

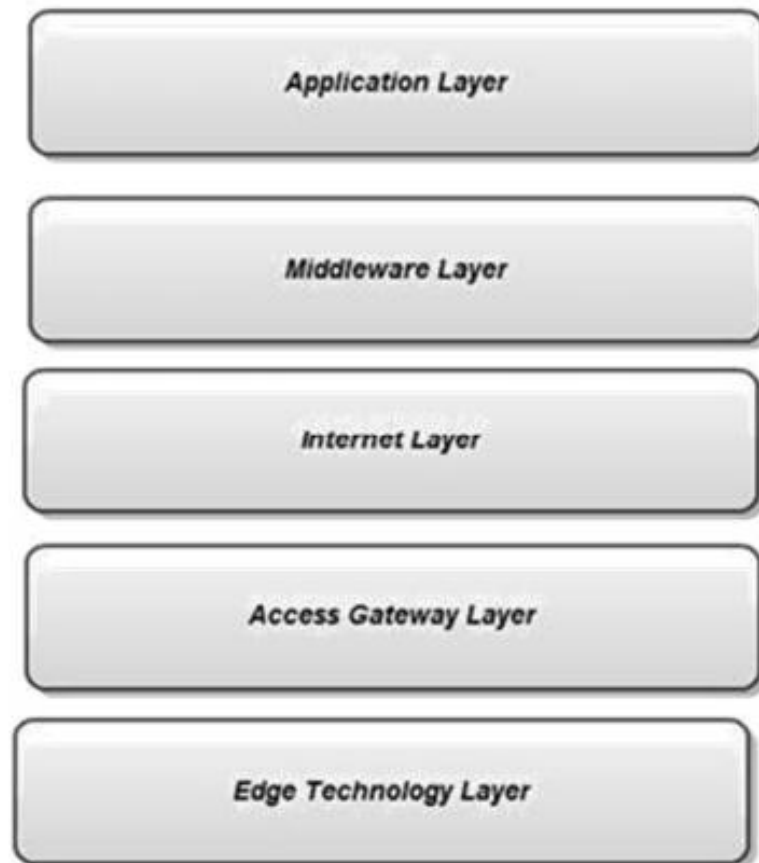


Fig. 8.1: Layered architecture of the Internet of Things (IoT)

8.4 CHALLENGES

IoT-based systems can be used for different purposes and areas, so that, we have to face different challenges. In this section, we explain some of the challenges that need to be considered in research activities:

- *Edge technology*

At the hardware level, more research efforts are required to develop the technology of embedded devices, sensors, actuators and (passive and active) identification, since an IoT-based system must be able to gather sufficient information about the real world by employing a large variety of devices and environments. Thus, there more work is still required to connect heterogeneous devices and deploy them in IoT applications, and to provide support for new devices.

- *Networking technologies*

In IoT, *things* are connected through different kinds of networks, i.e. mobile, wired and wireless network. These networks support bi-directional communication at different levels among the real world objects, applications services that are employed by the IoT applications. This highly distributed 46 structure should provide interconnection with low energy consumption, while distributed data can cause privacy issues.

- *Middleware system*

In IoT, we have heterogeneous devices and networks. Their heterogeneity can potentially increase with new technologies. To facilitate the use of these devices by IoT applications, we should shield their heterogeneity. Therefore, we need to develop a secure, scalable and semantically enriched service-oriented middleware to cope with the heterogeneity of devices.

- *Platform services*

They support a superior management of all involved devices in an integrated way, ensuring scalability, high availability, and the safe and secure execution of the requested functionality from devices. In continue, we focus on the middleware challenges, because we are looking for the functionalities that a middleware can provide for the application layer in the IoT-based systems.

8.5 IoT BASED MIDDLEWARE

Bandyopadhyay, S. et al. have studied the middleware systems that have been applied in IoT-based systems [1]. They classify the required functionality of middleware to manage interaction with a variety of devices in four functional components, namely (1) interface protocols, (2) device abstraction, (3) central control, context detection & management, and (4) application abstraction (shown in Fig. 8.2). In the following, we explain these components in details.

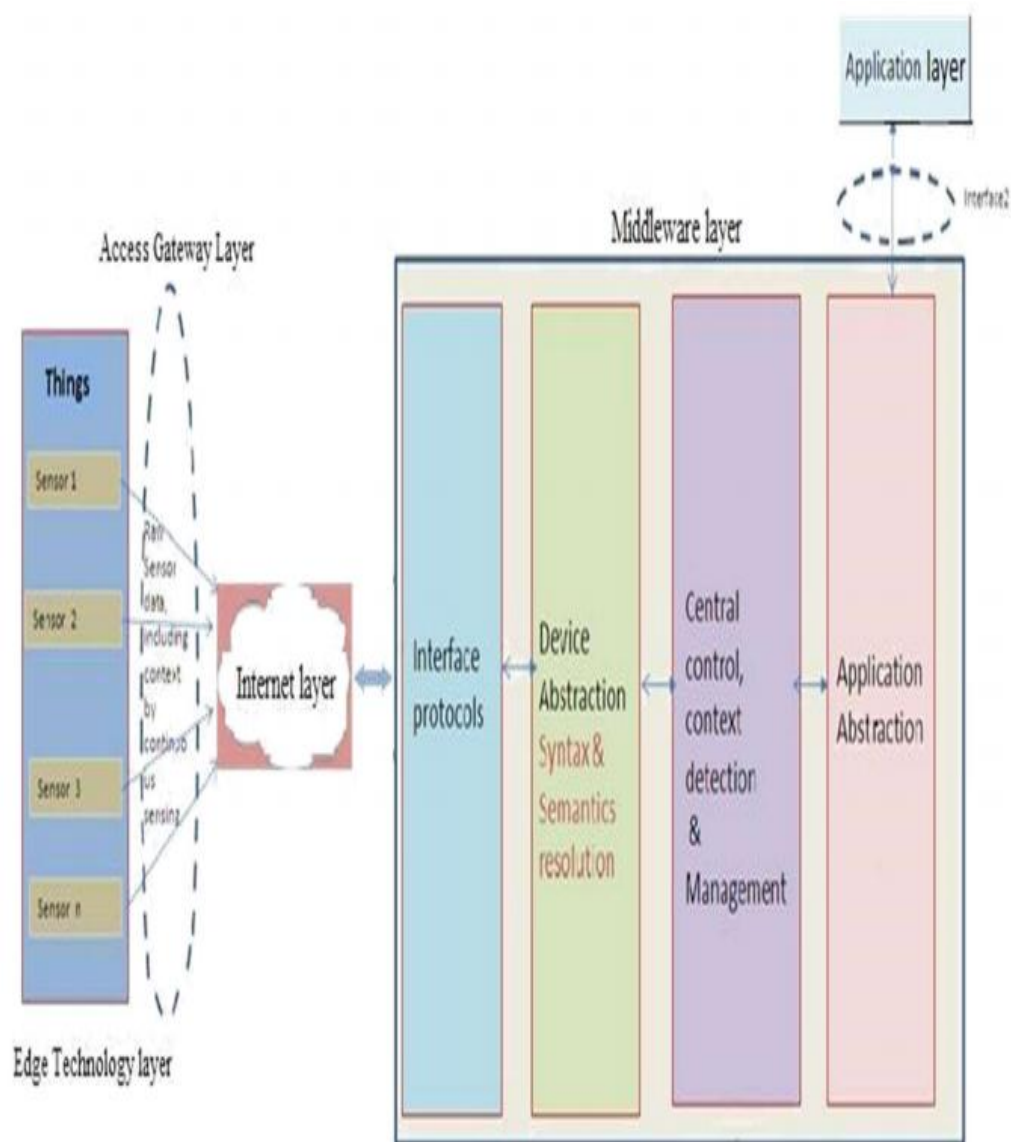


Fig. 8.2: Functional components of a middleware for IoT-based Systems

8.6 INTERFACE PROTOCOLS

This component is in charge of providing technical interoperability. Interoperability in the context of Interface protocols means: the ability of two systems to interoperate by using the same communication protocols. According to ETSI (European Telecommunications Standards Institute) technical interoperability is defined as the association of hardware or software components, systems and platforms that enable machine-to-machine communication to take place. This kind of interoperability is often centered on (communication) protocols and the infrastructure needed for those protocols to operate.

The Interface Protocol component defines protocols for exchanging information among different networks that may work based on different communication protocols, in order to allow technical interoperability. This component is responsible for handling basic connectivity in the physical and data link, network, transport, and sometimes the application layer of the TCP/IP stack. To cope with the heterogeneity of devices, we can use a wrapper for each device to translate the protocol supported by the device to a common protocol. This wrapper can be placed either on the device side or the middleware side. If we want to have direct interaction with devices, we should place the wrapper in the middleware side. Devices usually have limited capability of computational process, so this would be a reason to implement wrapper on the middleware side. In contrast, in case of indirect interaction with devices we can develop an intermediary wrapper between the middleware and the devices. The interface protocol component is responsible to allow the middleware to support both direct and indirect interactions.

8.7 DEVICE ABSTRACTION (DA)

This component is responsible for providing an abstract format to facilitate the interaction of the application components with devices. This abstraction provides *syntactic* and *semantic* interoperability, which are defined by ETSI as follows:

- *Syntactic interoperability* is associated with data formats. The messages transferred by communication protocols must have a well-

defined syntax and encoding format, which can be represented by using high-level transfer syntaxes such as, HTML and XML.

- *Semantic interoperability* is usually associated with the meaning of the content of message which is understandable for human. Thus, interoperability on this level means that there is a common understanding among people on the meaning of the content (information) being exchanged among them. However, since DA does not communicate directly with human, semantic interoperability in the context of DA is in charge of this common understanding for applications.

Semantic interoperability relies on semantic models which tends to be domain specific. For example, one way to provide semantic interoperability in Service Oriented (SOA) based middleware is by using Devices Profile for Web Services (DPWS) . In this context, each device type refers to a distinguished service type DA component provide two general functionalities: (1) to ask devices to perform some functionality and (2) to define and configure devices DPWS uses the XML format that is shown in Code 2.1.

```
<dpws:ThisModel>
  <dpws:Manufacturer>ACME Manufacturing</dpws:Manufacturer>
  <dpws:ModelName xml:lang="en-GB" >
    ColourBeam 9
  </dpws:ModelName>
  <dpws:ModelName xml:lang="en-US" >
    ColorBeam 9
  </dpws:ModelName>
</dpws:ThisModel>
</wsx:MetadataSection>
<wsx:MetadataSection
  dialect="http://docs.oasis-open.org/ws-dd/ns/dpws/2009/01/ThisDevice
  >
  <dpws:ThisDevice>
    <dpws:FriendlyName xml:lang="en-GB" >
      ACME ColourBeam Printer
    </dpws:FriendlyName>
    <dpws:FriendlyName xml:lang="en-US" >
      ACME ColorBeam Printer
    </dpws:FriendlyName>
  </dpws:ThisDevice>
```

Code 2.1 XML code to define a device

8.8 CENTRAL CONTROL, CONTEXT DETECTION & MANAGEMENT (CCM)

Context characterizes the situation of an entity, which can be a place, a person or an object that is relevant to the user, applications and their interactions [1]. The CCM functional component is responsible to support context-aware computation that is a computational style that take to account the context of the entities that interact with the system. A middleware for IoT-based systems must be contextaware to work in smart environments [1]. Smart environments refer to a physical world that is richly and invisibly interwoven with sensors, actuators, displays, and computational elements, embedded seamlessly in the everyday objects of our lives, and connected through a continuous network]. Context-awareness includes two functionalities:

- 1) Context detection, which consists of collecting data from resources, and selecting the information that can have an impact on the computation.
- 2) Context processing, to use the gathered information to perform a task or make a decision.

8.9 APPLICATION ABSTRACTION

This functional component provides an interface for both high-level applications and end-users to interact with devices. For instance, this interface can be a RESTful interface or can be implemented with some query-based language. A. Malatras. et. al. propose a SOA-based middleware to perform data management and data monitoring services. This middleware uses a RESTful interface to facilitate the interaction of high-level applications with sensors, which can communicate with Wireless Sensor Network (WSN) through the HTTP operations: (1) GET to issue a query on an existing resource, (2) DELETE to remove an existing resource, (3) POST to create a new resource, and (4) PUT to update an existing resource. For instance, a client application gets the result of a domain task by sending a GET request through the URL: „http :/{ hostname}/REST/ {version}/ DomaintaskResult/id’. In this URI that reference to an „id“ leads to a unique result.

In the TinyDB middleware, an end-user can use a query language to interact with devices. For instance, a user can send the following query to get the id of a sensor (nodeid) nearby and temperatures that are sensed by this sensor during the past 10 seconds before executing the query: `SELECT nodeid, temp FROM sensors SAMPLE PERIOD 1s FOR 10s`

8.10 CONCLUSION

In this chapter, we defined IoT and identified common IoT layers. Furthermore, we discussed a reference middleware architecture for IoT-based systems. This architecture has been proposed by Bandyopadhyay et. al based on a study on the existing middleware frameworks for IoT-based systems.

8.10.1 Review of IoT-based Middleware

The term *Internet of Things (IoT)* refers to a wide set of applications and research areas such as, distributed computing and knowledge management. Much of the IoT research we found in the literature is related to the field of *pervasive computing*. World Wide Web Consortium (W3C) [3] defines *pervasive computing* as *a vision about our future computing life style. In this vision, computer systems will be seamlessly integrated into our everyday life, anticipating our needs and providing relevant services and information for us in an anytime-and-anywhere fashion. Pervasive Computing is all about making the human's life easier by exploiting the available computing technologies.* This motivates scientists to develop new technologies for the everyday people's lives.

Since the vision of IoT is almost similar to Pervasive computing, we reviewed the literatures that address both IoT and Pervasive systems. To select the middleware to discuss we had the following approaches:

- 1) We looked at typical IoT middleware such as ISMP, ASPIRE, GSN. We selected HYDRA to review, because it is the most popular and well-documented middleware in comparison to the mentioned middleware. Also, based on the study of Bandyopadhyay, S. et. al,

those middleware does not support the discussed functional components in Chapter 3, while HYDRA supports them.

- 2) We looked at typical Pervasive system middleware. We selected the following middleware to review:
- 3) AURA because this middleware focus on elaboration and manipulation of the gathered data from devices. We aim to provide ease of configuration and deployment for end user and developer. Since to meet this goal, we require to facilitate the gathered data manipulation, we reviewed AURA that was the only middleware among the reviewed paper that considered the data manipulation.

TinyDB focus on gathering data from devices. Since in an IoT-based system we need to gather data from environment through different devices, we reviewed TinyDB that is a popular middleware. WiseMID is the only middleware among the reviewed middleware that is specific for energy saving purpose. As saving the energy of devices is important issue, we reviewed this middleware.

In this chapter, we the mentioned middleware in more detail. Also, we analyze the discussed middleware with respect to the functional components mentioned.

CHAPTER 9

CONCLUSION

Internet of Things is the concept in which the virtual world of information connected to the real world of things. Internet of Things is a smart technology that helps all connected devices to update themselves according to changes in the surrounding environment and to be able to be adopted and work in any other strange environment with high accuracy.

The future of IoT has the potential to be limitless. Advances to the industrial internet will be accelerated through increased network agility, integrated artificial intelligence (AI) and the capacity to deploy, automate, orchestrate and secure diverse use cases at hyperscale. In this world of busy roads, traffic signal plays the vital factor in saving person's life.

In this project we are not only tried to provide a way to ambulance but also tried to provide a ready situation in hospital to do the treatment of patient in time. We implemented two solutions like storing the medical history of all citizens, suggesting the nearest hospitals and giving an optimal path to reach hospital in time. In this project we have learned that This project represents an example of an intelligent ambulance traffic control with IoT. If it is fixed the ambulance will not wait for a signal in traffic. And also it is the way to easier to reach the hospital for quickly and save the patient's life. By this project the life is saved. In future, this project will be enhanced by using automated approach for fast and accurate performance, the emergency vehicles experience less congestion and reach faster to the destination and thus many life's were been saved, and there are advantages like A quick medical support for the accident victims can be given, Traffic chaos can be controlled, Other vehicles on the road can know that, there is an emergency with the help of display screen and blue light. Travelling to nearby hospital can be made easy.

9.1 SCREENSHOTS OF OUTPUT

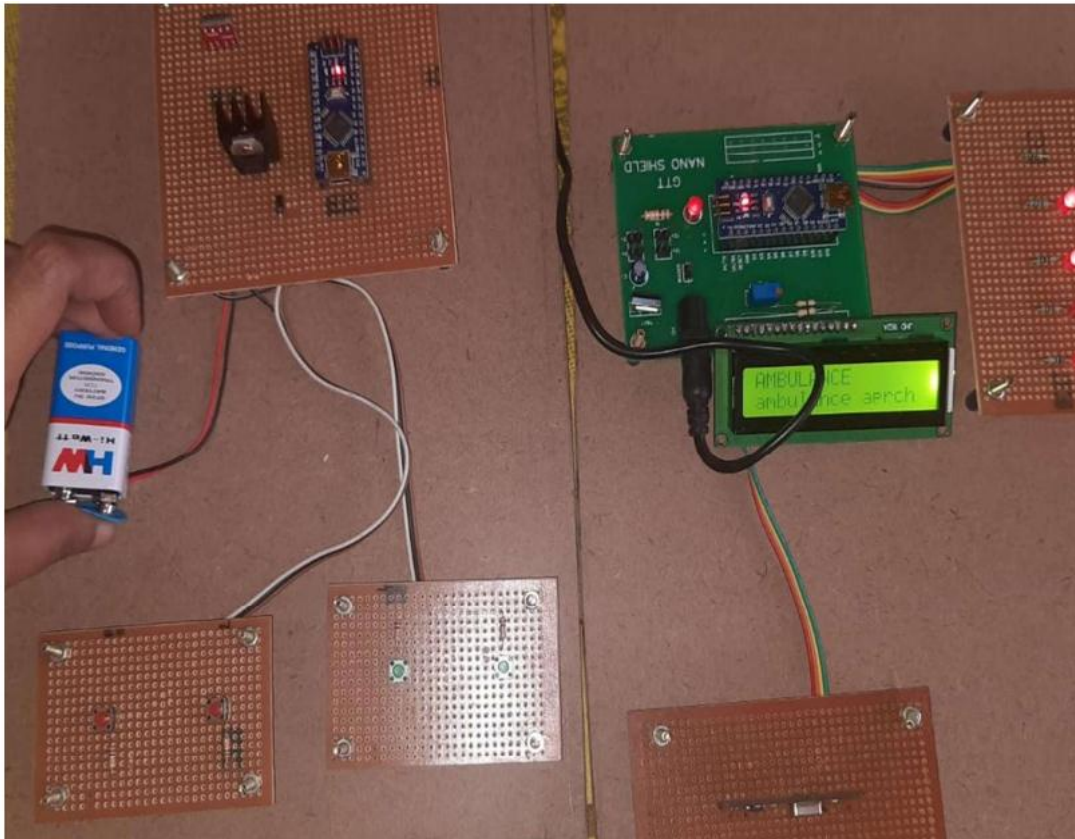


Fig. 9.1: Circuit diagram-Arrival of ambulance

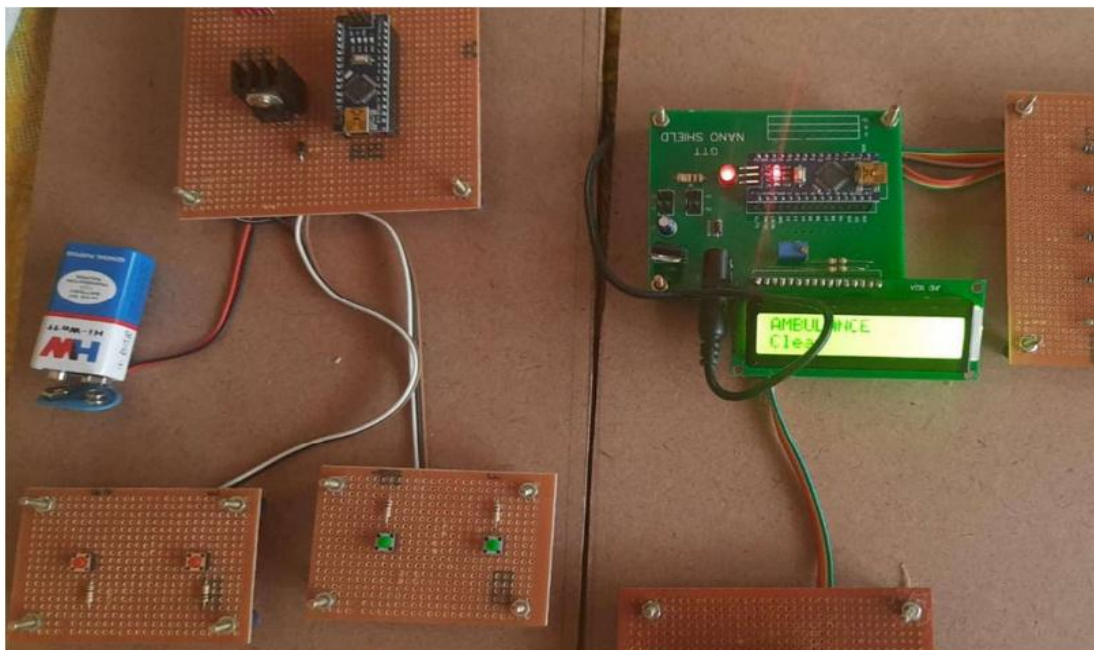


Fig. 9.2: Circuit diagram-Departure of ambulance

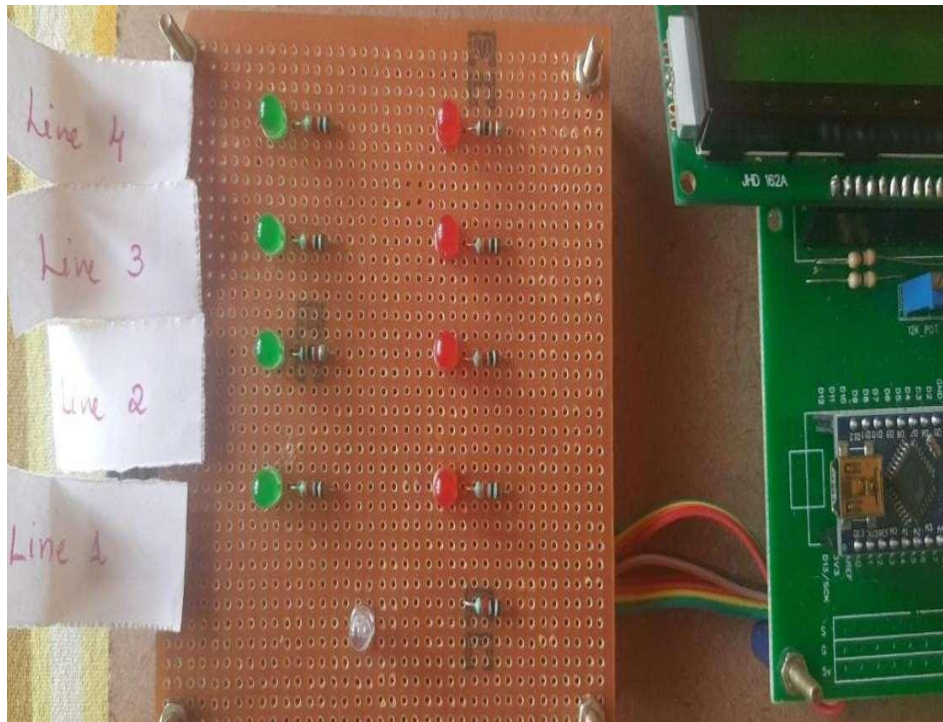


Fig. 9.3: Traffic signals with LED

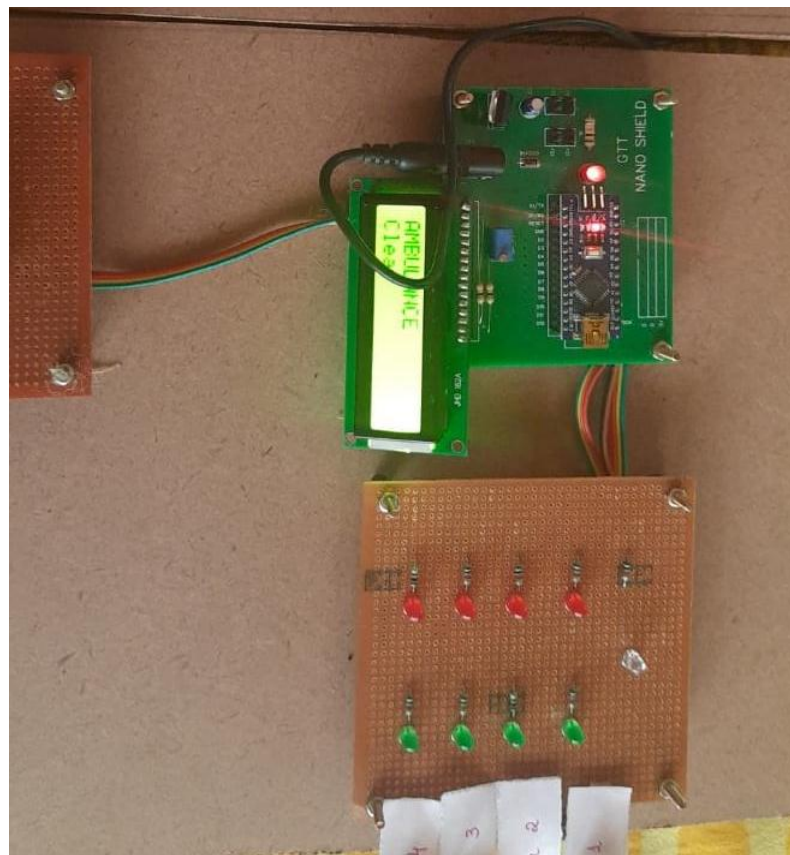


Fig. 9.4: Receiver side circuit

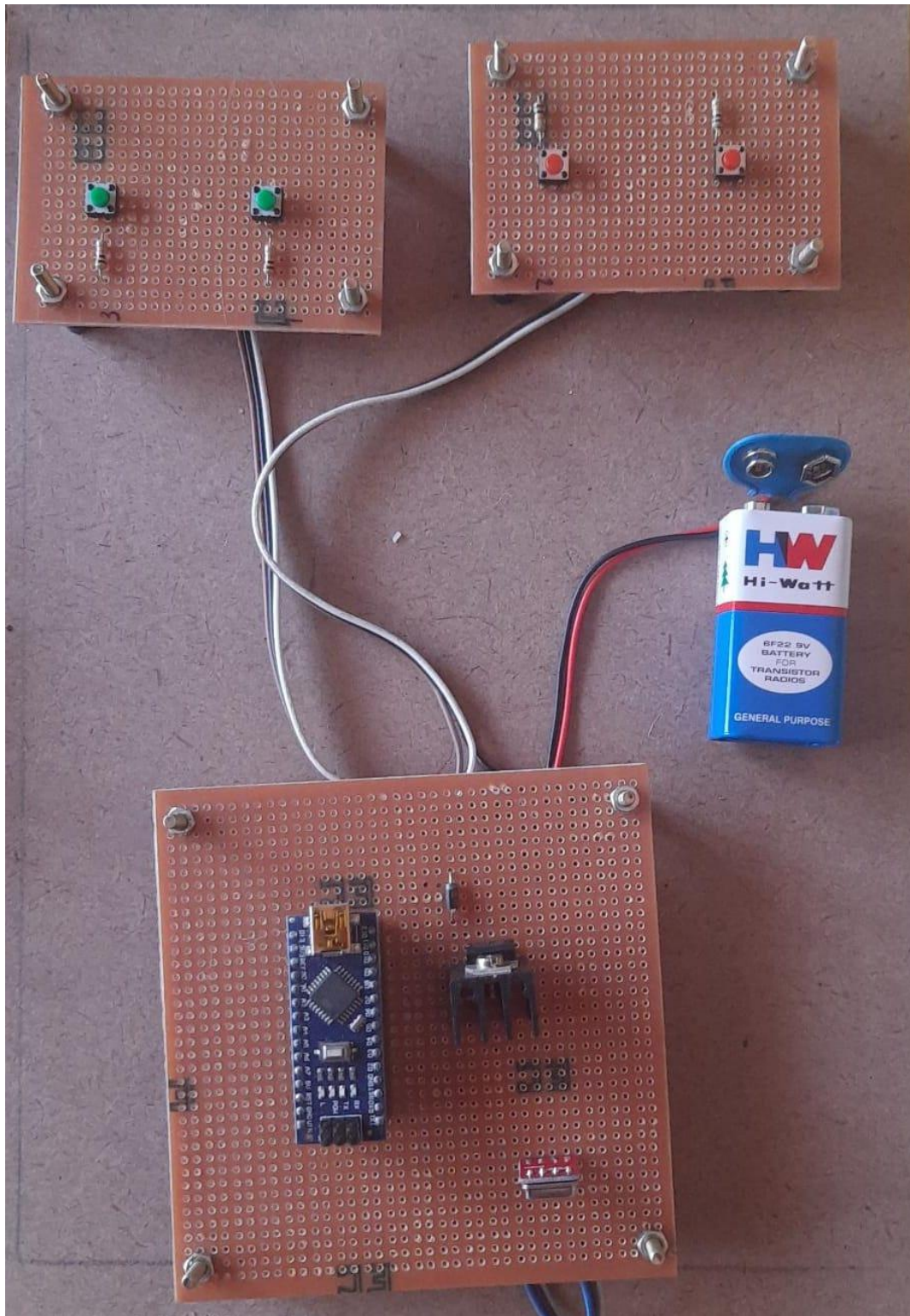


Fig. 9.5: Transmitter side circuit

REFERENCE

- [1] Abid Khan, Ravi Mishra, "Zigbee – GSM Based Tracking System", International Journal of Engineering Trends and Technology, Vol. 3, Issue2, pp: 161-169, 2012.
- [2] Bhumkar, S.P., Deotare, V.V., and Babar, R.V., "Intelligent Car System for Accident Prevention Using Arduino-7", International Journal of Emerging Technology and Advanced Engineering, Vol 2, Issue 4, pp: 56-78, 2012.
- [3] Nagaraja, B.G., Ravi Rayappa, Mahesh, M., Chandrasekhar M Patil, and Manjunath, T.C., "Design and Development of GSM based vehicle theft control system" Advanced Computer Control ICACC '09 International conference, pp148.
- [4] Xu Li, Wei Shu, Minglu Li, Hong-Yu Huang, Pei-En Luo, and MinYou Wu, "Performance Evaluation of Vehicle-Based Mobile Sensor Networks for Traffic Monitoring", IEEE 2009.
- [5] Young, Kar-Keung D. "Automated navigation and mobile vehicle control using wireless sensor network technology," Proceedings of the IEEE International Conference on Industrial Technology, 2008.