AMBULANCE CONTROLLED TRAFFIC SYSTEM USING RFID

A Project Report

Submitted by

D.SRIDHARANI	**Roll. No.:13BEC148**
S.V.SUVETHANAYAKI	**Roll. No.:13BEC153**
S.VIDHYA	**Roll. No.:13BEC165**
A.VINODHA	**Roll.No.:13BEC172**

in partial fulfillment for the award of the degree

of

BACHELOR OF ENGINEERING

IN

ELECTRONICS AND COMMUNICATION

ENGINEERING

KUMARAGURU COLLEGE OF TECHNOLOGY

COIMBATORE - 641 049

(An Autonomous Institution Affiliated to Anna University, Chennai)

APRIL 2017

BONAFIDE CERTIFICATE

Certified that this project report titled **"AMBULANCE CONTROLLED TRAFFIC SYSTEM USING RFID"** is the bonafied work of "**S.VIDHYA , D.SRIDHARANI , S.V.SUVETHANAYAKI,A.VINODHA"** who carried out the project work under my supervision.

SIGNATURE	**SIGNATURE**
Ms.Shiji Shajahan	Dr.K.Malarvizhi
Assistant Professor/E.C.E	HEAD OF THE DEPARTMENT
Kumaraguru College of Technology	Electronics & Communication Engineering
Coimbatore.	Kumaraguru College of Technology
	Coimbatore.

The candidates with Register numbers 13BEC148,13BEC153,13BEC165 and 13BEC172 are examined by us in the project viva-voce examination held on …………………….

INTERNAL EXAMINER **EXTERNAL EXAMINER**

ACKNOWLEDGEMENT

First we would like to express our praise and gratitude to the Lord, who has showered his grace and blessing enabling us to complete this project in an excellent manner. He has made all things in beautiful in his time.

We express our sincere thanks to our beloved Joint Correspondent, **Shri. Shankar Vanavarayar** for his kind support and for providing necessary facilities to carry out the project work.

We would like to express our sincere thanks to our beloved Principal **Dr.R.S.Kumar M.E., Ph.D.,** who encouraged us with his valuable thoughts.

We would like to express our sincere thanks and deep sense of gratitude to our HOD **Dr.K.Malarvizhi Ph.D.,** for her valuable suggestions and encouragement which paved way for the successful completion of the project.

We are greatly privileged to express our deep sense of gratitude to the Project Coordinator , Assistant Professor (SRG), for his continuous support throughout the course.

In particular, We wish to thank and express our everlasting gratitude to the Supervisor **Ms.Shiji Shajahan**, Assistant Professor for her expert counselling in each and every steps of project work and we wish to convey our deep sense of gratitude to all teaching and non-teaching staff members of ECE Department for their help and cooperation.

Finally, we thank our parents and our family members for giving us the moral support in all of our activities and our dear friends who helped us to endure our difficult times with their unfailing support and warm wishes.

ABSTRACT

Traffic management on the road has become a severe problem of today's society because of growth of the urbanization. This leads to traffic jam at the traffic junctions which in turn causes delay to ambulances. In order to overcome this problem, this project presents a simple ambulance controlled traffic system. The main objective of this system is to control the traffic, allowing an ambulance to arrive at a particular location without it having to stop anywhere until the destination is reached. This system includes RFID technology and ZIGBEE communication. An RFID reader in the ambulance reads the ID number from the corresponding RFID tag. The RFID readers provide the information so that it compares the received ID with default ID’s stored in its memory. If the obtained ID gets matched with any of the IDs, then a green signal is given along the path of the ambulance or else no change in the signal takes place. Additionally, the information about the nearest hospitals in that route is provided by ZIGBEE which makes it easy for the ambulance to reach the nearest emergency centre quickly. ZIGBEE is placed in the ambulance which acts as receiver and the one in the traffic signal acts as transmitter .The transmitter sends the information about the hospitals on receiving the input from the driver.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF ABBREVATIONS

RFID	Radio Frequency Identification
PIC	Peripheral Interface Controller
LED	Light Emitting Diode
LCD	Liquid Crystal Display

LIST OF FIGURES

1. INTRODUCTION

India is the second most populous country in the world and is a fast growing economy. Because of more population the growth in the number of vehicles is increasing day by day. There are many issues related to increasing traffic such as accidents, numerous type of pollution, time wastage and health related problems. The major reasons for traffic problems are increase in the number of vehicles, violation of traffic rules and increase in number of accidents which creates problems for other vehicles. This in turn has an adverse effect in the lives due to ambulance getting stuck in traffic jams. Due to all these problems the increase in congestion level, especially at peak hours is one of the challenging works for the transportation specialists. But the existing methods for traffic management are not efficient in terms of the performance, cost and the effort needed for maintenance and support. There is no efficient method to recognize and transport emergency vehicles like ambulances so that they could reach the hospitals on time. This in turn causes harm to the patient who is inside the ambulance. To solve this problem, traffic is to be controlled whenever ambulance arrives at the junction so that a green signal is to be given along its path. This project includes RFID which detects presence of the ambulance and Zigbee which acts as a transceiver module which operates at low power to transmit and receive data. RFID is an emerging wireless technology that uses radio frequency electromagnetic energy to identify objects from a distance without requiring direct line of sight. The objects are identified with the tag where the information of the object is prestored. This system produces a smart and fully automatic traffic control system that will detect and control the congestion in real time and also displays the direction of the hospital which is at the shortest distance to the ambulance.

2. HARDWARE DESCRIPTION:

2.1 PROJECT DESCRIPTION:

The main objective of this project is that allowing an ambulance to arrive at a particular location without it having to stop anywhere until the destination is reached. Apart from unblocking the signal, the nearest hospital information on that particular route is also shared with the ambulance.

RFID readers are placed in the ambulance whereas RFID tags on the roadways at a particular distance from the traffic signal say 5m.So, whenever the ambulance crosses the path where tags are placed; the reader detects the radio frequency signal from the tag and generates corresponding identification code. This code is in turn sent to micro controller. Here, microcontroller compares the received ID with the default ID numbers stored in its memory during programming. If any of the ID gets matched with the received ID, then microcontroller changes the state of signal by giving green signal along the path of the ambulance.

The signal remains green for a particular time so that the ambulance can cross the signal. Additionally the nearest hospital's direction is provided to the driver using Zigbee. Zigbee is placed both in the ambulance and in the traffic signal. When the input is received by the driver, the microcontroller receives the control and the Zigbee in the traffic signal sends the information to the ambulance which is viewed in a LCD.

2.2 BLOCK DIAGRAM:

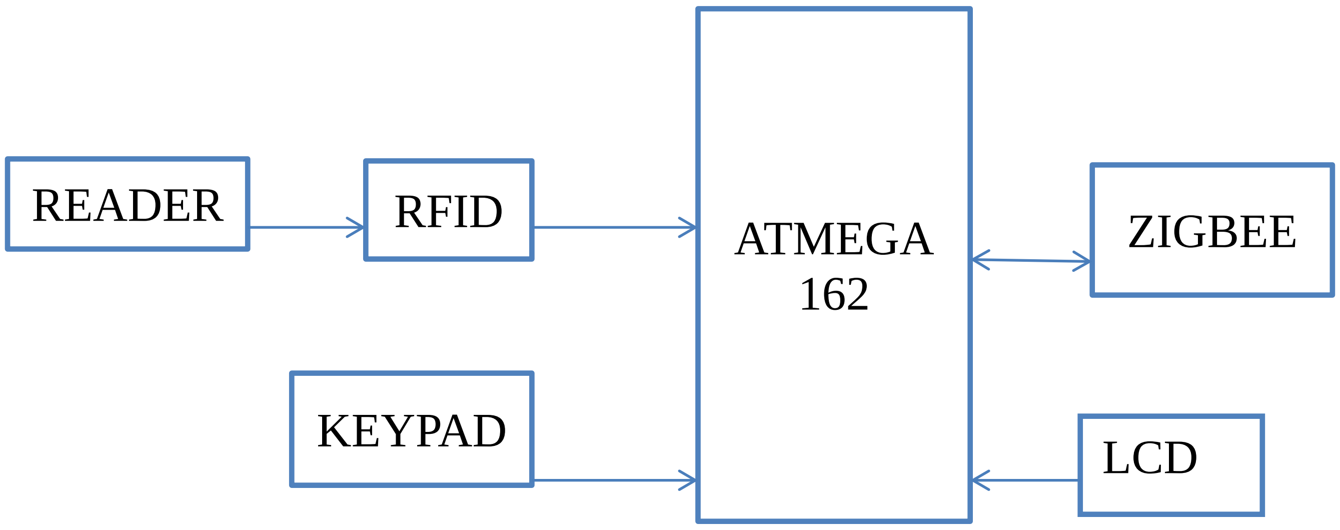

Fig:2.1-Ambulance Unit

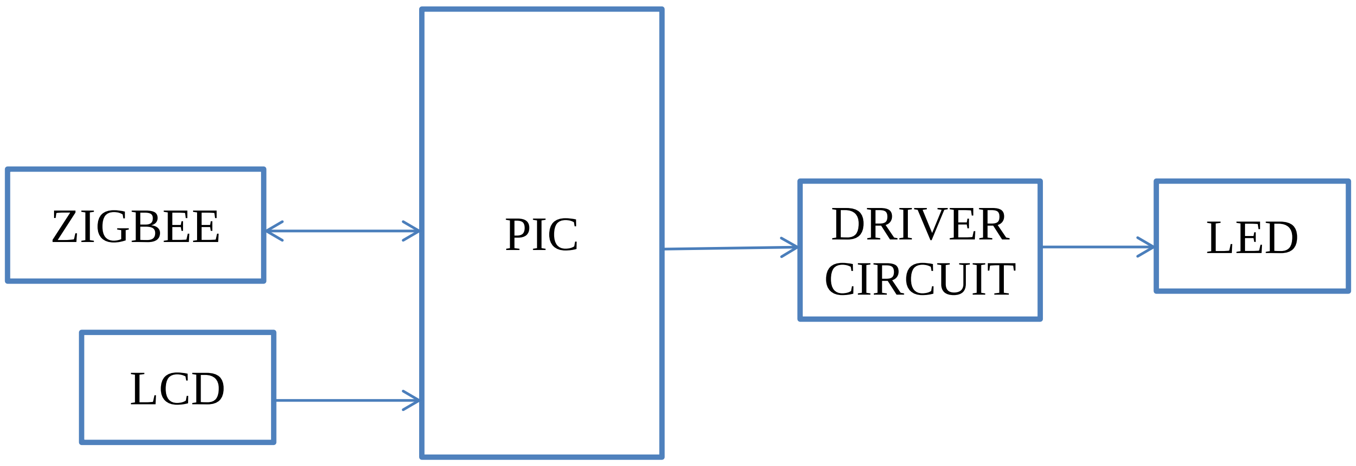

Fig:2.2-Traffic Unit

2.3 COMPONENTS AND IT'S USES:

Components	Uses
Atmega 162	Used in ambulance unit
PIC	Used in traffic unit
Zigbee	Used in communication between traffic and ambulance unit
RFID reader	Used to read ambulance information
RFID tags	Ambulance information is stored Electronically
LCD	Used for displaying tag information
LED	Used as traffic signal

2.4 HARDWARE DESCRIPTION:

The hardware components used in this project are PIC 16F877A, Atmega162, RFID reader and tag (passive), LCD, power supply, Zigbee, LED's.

2.4.1 Power supply:

A power supply supplies electric energy to an electric load. All power supplies have a power input, which receives energy from the energy source, and a power output that delivers energy to the load.

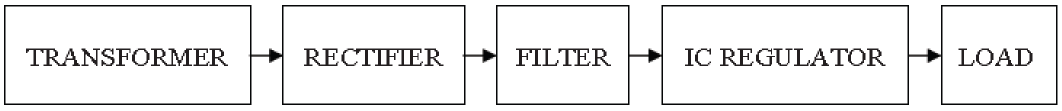

Fig:2.3-Block Diagram

The ac voltage, typically 220V rms, is connected to a transformer, which steps that ac voltage down to the level of the desired dc output. A diode rectifier then provides a full-wave rectified voltage that is initially filtered by a simple capacitor filter to produce a dc voltage. This resulting dc voltage usually has some ripple or ac voltage variation.

2.4.2 Transformer:

The potential transformer will step down the power supply voltage (0-230V) to (0-9Vand 15-0-15) level. If the secondary has less turns in the coil then the primary, the secondary coil's voltage will decrease and the current or AMPS will increase or decreased depend upon the wire gauge. This is called a STEP-DOWN transformer.Then the secondary of the potential transformer will be connected to the rectifier.

2.4.3 Bridge rectifier:

When four diodes are connected, the circuit is called as bridge rectifier. The input to the circuit is applied to the diagonally opposite corners of the network, and the output is taken from the remaining two corners.

2.4.4 Positive 12 and Negative 12 Volt circuit:

The unregulated AC/DC power supply part of the circuit consists of a transformer that steps down 230VAC to 15 volts across a center tapped secondary winding 15V AC individually across the two halves of the secondary winding with opposite polarities, diodes (D5) to (D8) that rectify the AC appearing across the secondary with (D5) and (D7) providing ‘full wave rectification to produce a positive output, (D6) and (D8), providing full wave rectification to produce a negative output, capacitors (C2) and (C3) providing the filtering action.. 7812 is a fixed output positive three terminal regulator whereas 7912 is a fixed output negative three terminal regulator.

2.4.5 Filter:

If a capacitor is added in parallel with the load resistor of a rectifier to form a simple Filter Circuit, the output of the rectifier will be transformed into a more stable DC Voltage. At first, the capacitor is charged to the peak value of the rectified Waveform. Beyond the peak, the capacitor is discharged through the load resistor until the time at which the rectified voltage exceeds the capacitor voltage. Then the capacitor is charged again and the process repeats itself.

2.4.6 IC voltage regulators:

Voltage regulators comprise a class of widely used ICs. Regulator IC units contain the circuitry for reference source, comparator amplifier, control device, and overload protection all in a single IC. IC units provide regulation of either a fixed positive voltage, a fixed negative voltage, or an adjustably set voltage. The regulators can be selected for operation with load currents from hundreds of milli amperes to tens of amperes, corresponding to power ratings from milli watts to tens of watts. A fixed three-terminal voltage regulator has an unregulated dc input voltage, applied to one input terminal, a regulated dc output voltage, from a second terminal, with the third terminal connected to ground.

The series 78 regulators provide fixed positive regulated voltages from 5 to 24 volts. Similarly, the series 79 regulators provide fixed negative regulated voltages from 5 to 24 volts. This is a regulated power supply circuit using the 78xx IC series. These regulators can deliver current around 1A to 1.5A at a fix voltage levels. The common regulated voltages are 5V, 6V, 8V, 9V, 10V, 12V, 15V, 18V, and 24V. It is important to add capacitors across the input and output of the regulator IC to improve the regulation. In this power supply circuit we get 5, 12 and -12Volt output.

A regulator circuit removes the ripples and also remains the same dc value even if the input dc voltage varies. This voltage regulation is usually obtained using one of the popular voltage regulator IC units. The voltage regulator used here is L7805cv.

2.4.7 PIC Microcontroller 16F877A:

Microcontroller is a general purpose device, which integrates a number of the components of a microprocessor system on to single chip. It is a self-contained system with CPU, memory and peripherals which act as a mini computer. PIC microcontroller is the first RISC based microcontroller fabricated in CMOS (complimentary metal oxide semiconductor) that uses separate bus for instruction and data allowing simultaneous access of program and data memory.

Various microcontrollers offer different kinds of memories. EEPROM, EPROM, FLASH etc. are some of the memories of which FLASH is the most recently developed. Technology that is used in pic16F877 is flash technology, so that data is retained even when the power is switched off.

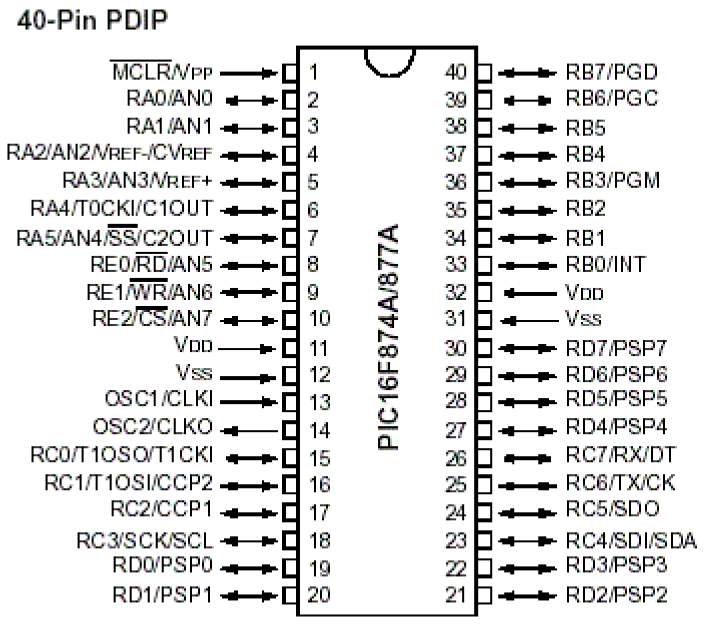

Fig:2.4-Pin Diagram

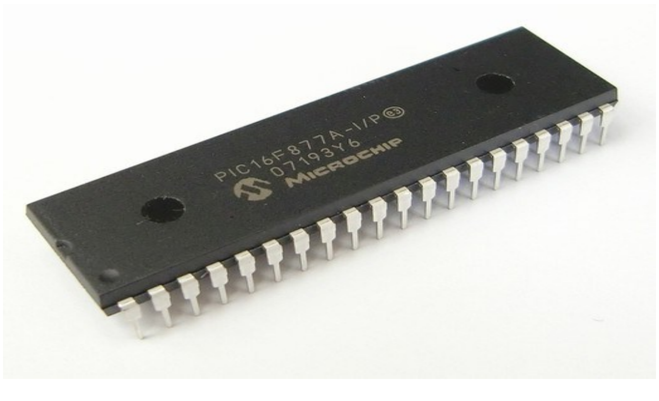

Fig:2.5-PIC 16F877A

Features:

- Only 35 single word instructions
- to learn
- All instructions are single cycle (1µs) except for program branches
- Operating speed: DC - 20MHz clock input
- 8 kBytes Flash Program Memory
- 368 Byte RAM Data Memory
- 256 Byte EEPROM Data Memory
- In-circuit Serial Programming
- Interrupt Capability (up to 10 sources)

2.4.10 ATMEGA 162:

The ATmega162 is a low-power CMOS 8-bit microcontroller based on the AVR enhanced RISC architecture. By executing powerful instructions in a single clock cycle, the ATmega162 achieves throughputs approaching 1 MIPS per MHz allowing the system designed to optimize power consumption versus processing speed.

In order to maximize performance and parallelism, the AVR uses Harvard architecture with separate memories and buses for program and data. Instructions in the program memory are executed with a single level pipelining. While one instruction is being executed, the next instruction is pre-fetched from the program memory. This concept enables instructions to be executed in every clock cycle.

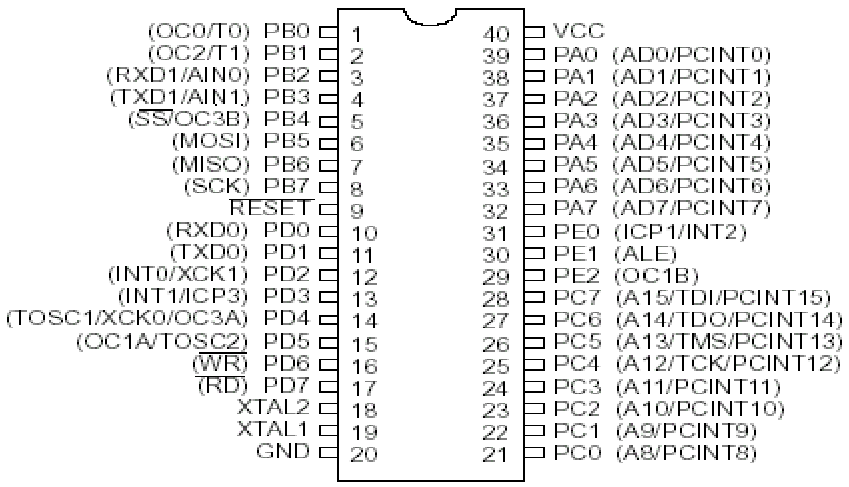

Fig:2.6-Pin diagram

Port A (PA7-PA0) Port A is an 8-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). The Port A output buffers have symmetrical drive characteristics with both high sink and source capability. When pins PA0 to PA7 are used as inputs and are externally pulled low, they will source current if the internal pull-up resistors are activated. The Port A pins are tri-stated when a reset condition becomes active, even if the clock is not running.

Port B (PB7-PB0) Port B is an 8-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). The Port B output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, Port B pins that are externally pulled low will source current if the pull-up resistors are activated. The Port B pins are tri-stated when a reset condition becomes active, even if the clock is not running.

Port C (PC7-PC0) Port C is an 8-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). The Port C output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, Port C pins that are externally pulled low will source current if the pull-up resistors are activated. The Port C pins are tri-stated when a reset condition becomes active, even if the clock is not running. If the JTAG interface is enabled, the pull-up resistors on pins PC7(TDI), PC5(TMS) and PC4(TCK) will be activated even if a Reset occurs.

Port D (PD7-PD0) Port D is an 8-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). The Port D output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, Port D pins that are externally pulled low will source current if the pull-up resistors are activated. The Port D pins are tri-stated when a reset condition becomes active, even if the clock is not running.

Port E(PE2-PE0) Port E is an 3-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). The Port E output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, Port E pins that are externally pulled low will source current if the pull-up resistors are activated. The Port E pins are tri-stated when a reset condition becomes active, even if the clock is not running.

Features:

- High-performance, Low-power AVR® 8-bit Microcontroller.
- Advanced RISC Architecture.
- 131 Powerful Instructions – Most Single-clock Cycle Execution.
- 32 x 8 General Purpose Working Registers.
- Up to 16 MIPS Throughput at 16 MHz.
- On-chip 2-cycle Multiplier.
- Non-volatile Program and Data Memories.
- 16K Bytes of In-System Self-programmable Flash.
- Endurance: 100,000 Write/Erase Cycles.
- 1K Bytes Internal SRAM.
- Up to 64K Bytes Optional External Memory Space.
- Programming Lock for Software Security.
- JTAG (IEEE std. 1149.1 Compliant) Interface.
- Boundary-scan Capabilities According to the JTAG Standard.
- Extensive On-chip Debug Support.
- Programming of Flash, EEPROM, Fuses, and Lock Bits through the JTAG.

2.4.11 RFID reader and RFID tag:

Radio-frequency identification (RFID) uses electromagnetic fields to automatically identify and track tags attached to objects. There are different kinds of RFID tags. In this project we use passive tag which collects energy from nearby RFID readers interrogating radio waves. The tags contain electronically stored information.

Radio-frequency identification (RFID) uses tags attached to objects to be identified. Two way radio transmitter-receivers called interrogators or readers send a signal to the tag and read its response.

The specification of the card used here is it works in the 125 kHz RF range. These tags come with a unique 32-bit ID and are not re-programmable. Card is blank, smooth, and mildly flexible. In these projects the reader reads the tag placed on the road side with the prestored information.

Fig:2.7-Reader Module

Features:

- The tag is typically much less expensive to manufacture.
- The tag is much smaller (some tags are the size of a grain of rice). These tags have almost unlimited applications in consumer goods and other areas.
- Tags can be read through a variety of substances such as snow, fog, ice, paint, crusted grime, and other visually and environmentally challenging conditions, where barcodes or other optically read technologies would be useless.
- RFID tags can also be read in challenging circumstances at remarkable speeds, in most cases responding in less than 100 millisecond.

Fig:2.8-RFID Tag

Features:

- Low-cost method for reading passive RFID EM4100 family transponder tags Reading Distance 10-15CM of the reader (Depend card shape).
- 125kHz read frequency.
- 9600 baud RS232 serial interface.
- Standard 2.54mm Pitch Bergstrip connector.
- Bread Board compatible.
- Low power Requirement 7-9V @ 100mA.
- Small Size.
- Built in Antenna .

- No components at PCB bottom side (easy to stick to any surface like wood,glass etc).
- Status LED for card detection.
- On-Board Power LED.

2.4.12 Zigbee:

Zigbee is an open global standard for wireless technology designed to use low power digital radio signals. Zigbee operates on IEEE 802.15.4 specification .It is a cost and energy-efficient wireless network standard.

Its low power consumption limits transmission distances to 10–100 meters line-of-sight, depending on power output and environmental characteristics. 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.

CC2500 is wireless transmitter receiver developed by Texas instruments which isused in 2400-2483.5 MHz ISM/SRD band systems. The CC2500 RF module is a low-cost 2.4 GHz transceiver used in very low power wireless applications. The RF transceiver is integrated with a highly configurable baseband modem. It supports OOK, 2-FSK, GFSK, and MSK modulations. It works in voltage range of 1.8 - 3.6V. Two AA batteries are enough to power it. It is always used with microcontroller which supports SPI communication. In our project Zigbee is used for the transmission of the information about the nearby hospitals to the ambulance.

Fig:2.9-Zigbee

Features:

- Low current consumption.
- Easy for application.
- Efficient spi interface.
- Operating temperature range : -40 ~+85
- Operating voltage :1. 8~ 3. 6 volts.
- Vailable frequency at : 2. 4-2. 483 ghz.
- Programmable output power (up to +1dbm).
- Hi sensitivity (-101dbm @10kbps).
- Small footprint in a 17x17mm,8pinpinout.
- Powerful digital features allow building a high-performance RF system.
- Using an inexpensive microcontroller.
- Wake-on-radio functionality for automatic low-power Rx polling.
- Burst mode data transmission with high over-the-air datarate.
- Reduces current consumption.
- Programmable data rate from1.2-500kbps.
- Robust solution with excellent selectivity and blocking performance.
- Ideal for multi-channels operation (50-800khz channels).
- Full packet handling including preamble generation, sync.
- Word insertion/detection,add check, flexible packet length.
- Programmable carrier sense indicator and digital rssi output enables.

2.4.13 Traffic LED:

A LED is a semiconductor device that emits visible light when an electric current passes through it. The LED consists of a chip of semiconducting material doped with impurities to create a pn-junction. As in other diodes, current flows easily from the p-side, or anode, to the n-side, or cathode, but not in the reverse direction. Charge-carriers electrons and holesflow into the junction from electrodes with different voltages. When an electron meets a hole, it falls into a lower energy level and releases energy in the form of a photon.

In silicon or germanium diodes, the electrons and holes usually recombine by a non-radiative transition, which produces no optical emission, because these are indirect band gap materials. The materials used for the LED have a direct band gap with energies corresponding to near-infrared, visible, or near-ultraviolet light.LEDs have many advantages like longer lifetime, smaller size, lower energy consumption and faster switching .Here it is used for traffic lights.

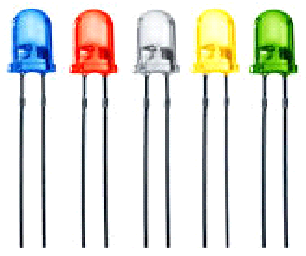

Fig:2.10-Led

2.4.14 LCD:

LCD (Liquid Crystal Display) is the technology used for displays which is much thinner than cathode ray tube technology. LCDs do not emit light directly. They are usually more compact, lightweight, portable, less expensive, more reliable, and easier on the eyes. They are available in a wider range of screen sizes than CRT and plasma displays, and since they do not use phosphors, they cannot suffer image burn-in when a static image is displayed on a screen for a long time. A 16x2 LCD means it can display 16 characters per line and there are 2 such lines. In this LCD each character is displayed in 5x7 pixel matrix. This LCD has two registers, namely, Command and Data. Small LCD screens are common in portable consumer devices such as digital cameras, watches, calculators, and mobile telephones, including smartphones.

LCD screens are also used on consumer electronics products such as DVD players, video game devices and clocks. IT has replaced heavy, bulky cathode ray tube (CRT) displays in nearly all applications.LCDs are, however, susceptible to image persistence.The LCD screen is more energy-efficient and can be disposed of more safely than a CRT can. By 2008, annual sales of televisions with LCD screens exceeded sales of CRT units worldwide, and the CRT became obsolete for most purposes.Here LCD is used to give input and to display the hospital information as output.

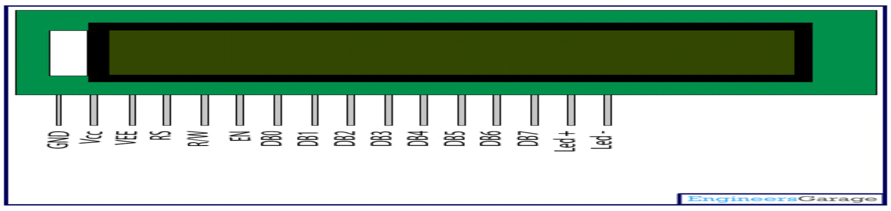

Fig:2.11-LCD Display

3. Working model:

The working model of the project along with the components is shown below.

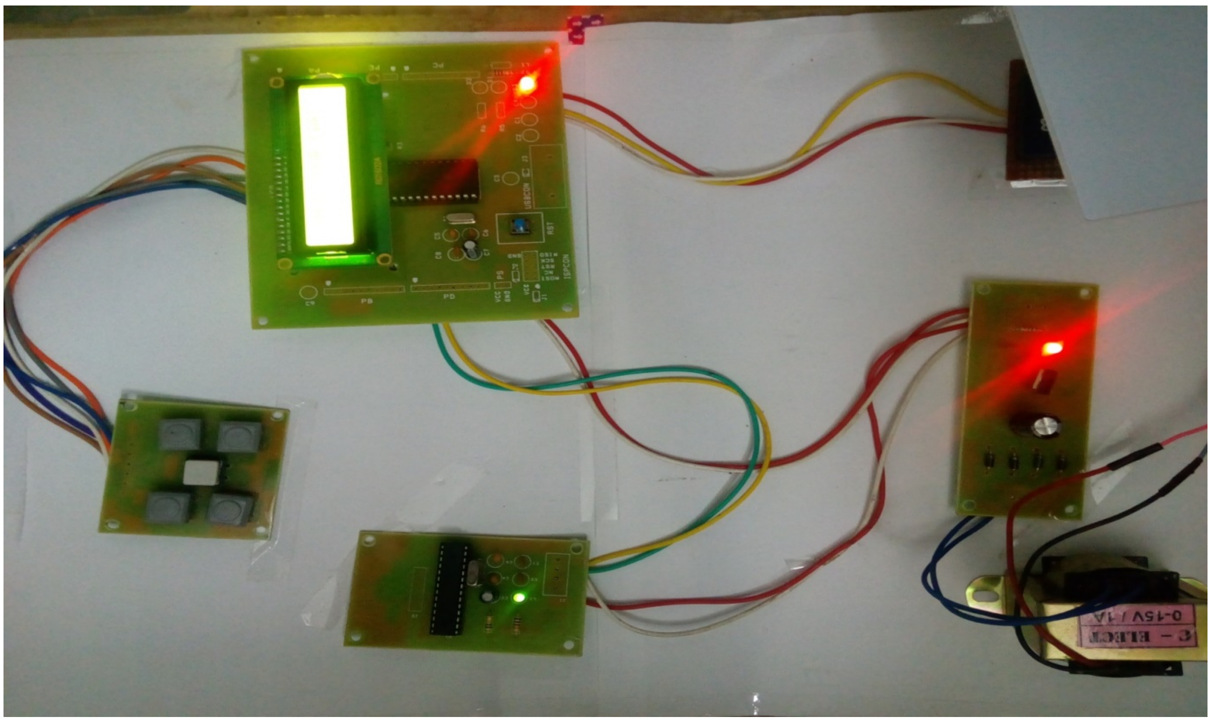

Fig:2.12-Ambulance Unit

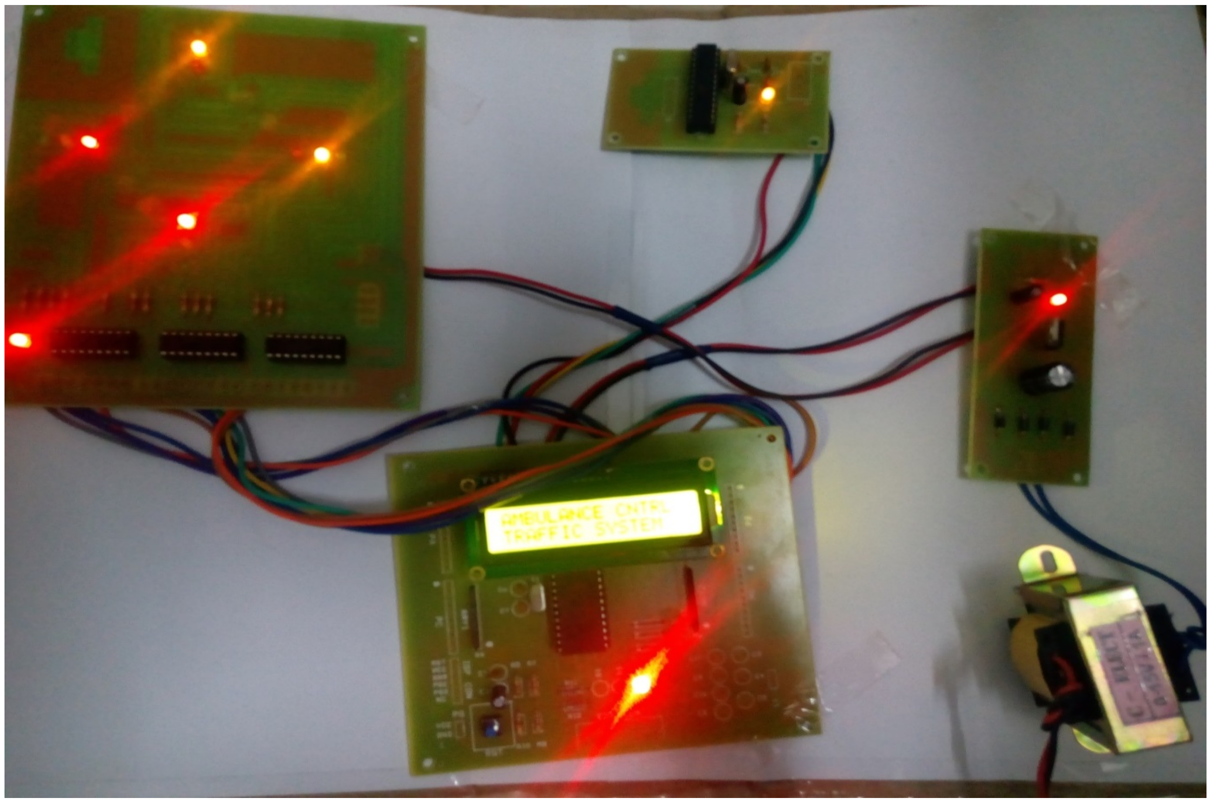

Fig:2.13-Traffic unit

4. SOFTWARE TOOLS:

Tools used:

MPLAB LAB FOR PIC

AVRDude FOR ATMEGA 162

4.1 MPLAB:

MPLAB IDE is an integrated development environment that provides development engineers with the flexibility to develop and debug firmware for various Microchip devices

MPLAB IDE is a Windows-based Integrated Development Environment for the Microchip Technology Incorporated PIC microcontroller (MCU) and dsPIC digital signal controller (DSC) families. In the MPLAB IDE, you can:

- Create source code using the built-in editor.
- Assemble, compile and link source code using various language tools. An assembler, linker and librarian come with MPLAB IDE.
- Debug the executable logic by watching program flow with a simulator, such as MPLAB SIM, or in real time with an emulator, such as MPLAB ICE.
- Make timing measurements.
- View variables in Watch windows.
- Program firmware into devices with programmers such as PICSTART Plus or PRO MATE II.
- Find quick answers to questions from the MPLAB IDE on-line Help.

4.2 MPLAB SIMULATOR:

MPLAB SIM is a discrete-event simulator for the PIC microcontroller (MCU) families. It is integrated into MPLAB IDE integrated development environment. The MPLAB SIM debugging tool is designed to model operation of Microchip Technology's PIC microcontrollers to assist users in debugging software for these devices.

4.3 IC PROG:

The PRO MATE II is a Microchip microcontroller device programmer. Through interchangeable programming socket modules, PRO MATE II enables you to quickly and easily program the entire line of Microchip Pismire microcontroller devices and many of the Microchip memory parts.

PRO MATE II may be used with MPLAB IDE running under supported Windows OS's (see Read me for PRO MATE II.txt for support list), with the command-line controller PROCMD or as a stand-alone programmer.

4.4 COMPILER-HIGH TECH C:

A program written in the high level language called C; which will be converted into Pismire MCU machine code by a compiler. Machine code is suitable for use by a Pismire MCU or Microchip development system product like MPLAB IDE.

4.5 PIC START PLUS PROGRAMMER:

The PIC start plus development system from microchip technology provides the product development engineer with a highly flexible low cost microcontroller design tool set for all microchip PIC micro devices. Thepica start plus development system includes PIC start plus development programmer and MPLAB IDE.

The PIC start plus programmer gives the product developer ability to program user software in to any of the supported microcontrollers. The PIC start plus software running under MPLAB provides for full interactive control over the programmer.

4.6 AVRDude:

The AVRDude is excellent program for burning HEX code in to Atmel AVR microcontroller. AVRDUDE is a command line program, so you'll have to type in all the commands.

4.7 Commands:

- -p <partno>: This is just to tell it what microcontroller its programming. For example, if you are programming an ATtiny2313, use attiny2313 as the partnumber.
- -b <baudrate>: This is for overriding the serial baud rate for programmers like the STK500.

- -B <bitrate>: This is for changing the bitrate, which is how fast the programmer talks to the chip.
- -C <config-file>: The config file tells avrdude about all the different ways it can talk to the programmer. There’s a default configuration file.
- -c <programmer>: Here is where we specify the programmer type, if you're using an STK500 use stk500, if you're using a DT006 programmer use dt006, etc.
- -D: This disables erasing the chip before programming.
- -P <port>: This is the communication port to use to talk to the programmer. It might be COM1 for serial or LPT1 for parallel or USB for, well, USB.
- -F: This overrides the signature check to make sure the chip you think you're programming is. The test is strongly recommended as it tests the connection.
- -e: This erases the chip, in general we don't use this because we auto-erase the flash before programming.
- -U <memtype>:r|w|v:<filename>[:format]: It is the one that actually does the programming. The <memtype> is either flash or eeprom (or hfuse, lfuse or efuse for the chip configuration fuses, but we aren't going to mess with those). ther|w|v means you can use r (read) w (write) or v (verify) as the command. The <filename> is, well, the file that you want to write to or read from. and [:format] means there’s an optional format flag. We will always be using "Intel Hex" format, so use i
  So, for example. If you wanted to write the file test.hex to the flash memory, you would use -U flash:w:test.hex:i. If you wanted to read the eeprom memory into the file "eedump.hex" you would use -U eeprom:r:eedump.hex:i.
- -n: This means you don't actually write anything, its good if you want to make sure you don't send any other commands that could damage the chip, sort of a 'safety lock'.
- -V: This turns off the auto-verify when writing. We want to verify when we write to flash so don't use this.
- -u: If you want to modify the fuse bits, use this switch to tell it you really mean it.

- -t: This is a 'terminal' mode where you can type out commands in a row.
- -E: This lists some programmer specifications, don't use it.
- -v: This gives you 'verbose' output...in case you want to debug something.
- -q: This is the opposite of the above, makes less output.

Source code:

Atmega

```
#define F_CPU 8000000UL
#include <avr/io.h>
#include <util/delay.h>
#include <avr/interrupt.h>
#include <avr/eeprom.h>
#include <string.h>

#include "ATMEGA_LCD4_NEW.h"
#include "ATM_serial.h"


#define set (PIND & (1<<2))
#define mov (PIND & (1<<3))
#define inc (PIND & (1<<4))
#define dec (PIND & (1<<5))
#define ent (PIND & (1<<6))

int count,sec,j;
unsigned char rfid[15];


ISR(USART1_RXC_vect) // RFID
{
	rfid[j++]=UDR1;
}




void rf_id()
{
	if(j>=10)
	{
		Lcd4_Display(0xC0,rfid,10);
```

```
                _delay_ms(2000);
                j=0;
                Lcd4_Command(0x01);
                if(strncmp(rfid,"010094318A",10)==0)
                {
                        Lcd4_Display(0x80,"   AMBULANCE   ",16);
                        Lcd4_Display(0xC0,"     ROAD1     ",16);
                        Serial0_Conout("*1\r",3);
                }

                else if(strncmp(rfid,"0800956B72",10)==0)
                {
                        Lcd4_Display(0x80,"   AMBULANCE   ",16);
                        Lcd4_Display(0xC0,"     ROAD2     ",16);
                        Serial0_Conout("*2\r",3);
                }

                else if(strncmp(rfid,"090073B87C",10)==0)
                {
                        Lcd4_Display(0x80,"   AMBULANCE   ",16);
                        Lcd4_Display(0xC0,"     ROAD3     ",16);
                        Serial0_Conout("*3\r",3);
                }

                else if(strncmp(rfid,"09007406FD",10)==0)
                {
                        Lcd4_Display(0x80,"   AMBULANCE   ",16);
                        Lcd4_Display(0xC0,"     ROAD4     ",16);
                        Serial0_Conout("*4\r",3);
                }
                _delay_ms(2000);_delay_ms(2000);
                Lcd4_Display(0x80," SHOW YOUR CARD ",16);
                Lcd4_Display(0xC0,"                ",16);

        }

}

void Serial_Decimal4(unsigned int val)
{
        unsigned int Lcd_th,Lcd_thr,Lcd_h,Lcd_hr,Lcd_t,Lcd_o;

        val = val%10000;
        Lcd_th=val/1000;
        Lcd_thr=val%1000;
        Lcd_h=Lcd_thr/100;
        Lcd_hr=Lcd_thr%100;
        Lcd_t=Lcd_hr/10;
        Lcd_o=Lcd_hr%10;

        Serial0_Out(Lcd_th+0x30);
        Serial0_Out(Lcd_h+0x30);
        Serial0_Out(Lcd_t+0x30);
        Serial0_Out(Lcd_o+0x30);
}
void key()
{
        if(!set) {_delay_ms(500);Serial0_Conout("*1\r",3);}
        else if(!mov) {_delay_ms(500);Serial0_Conout("*2\r",3);}
        else if(!inc) {_delay_ms(500);Serial0_Conout("*3\r",3);}
        else if(!dec) {_delay_ms(500);Serial0_Conout("*4\r",3);}
}
int main()
{
        cli();
        DDRA=0xFF;
        DDRB=0x08; //
(0<<0)|(0<<1)|(0<<2)|(1<<3)|(0<<4)|(0<<5)|(0<<6)|(0<<7);
        DDRC=0xDF; //
(1<<0)|(1<<1)|(1<<2)|(1<<3)|(1<<4)|(0<<5)|(0<<6)|(0<<7);
        PORTC=0xff;
        DDRD=0x02; //
(0<<0)|(1<<1)|(0<<2)|(0<<3)|(0<<4)|(0<<5)|(0<<6)|(0<<7);
        PORTD |= (1<<2)|(1<<3)|(1<<4)|(1<<5)|(1<<6);

        Lcd4_Init();
        Lcd4_Display(0x80,"AMBULANCE CTRL  ",16);
        Lcd4_Display(0xC0,"TRAFFIC SYSTEM  ",16);
        _delay_ms(1500);
        Serial0_Init(9600);_delay_ms(100); //ZIGBEE
        Serial1_Init(9600);_delay_ms(100); //RFID
        _delay_ms(500);
        sei();
        _delay_ms(500);
        Receive0(1);
        Receive1(1);


        Lcd4_Command(0x01);
        Lcd4_Display(0x80," SHOW YOUR CARD ",16);
        Lcd4_Display(0xC0,"                ",16);


        while(1)
        {
                rf_id();
                key();
        }
}

void delay()
{
        _delay_ms(500);_delay_ms(500);
        _delay_ms(500);_delay_ms(500);
}


void Serial_Decimal3(unsigned int val)
{
        unsigned int Lcd_h,Lcd_hr,Lcd_t,Lcd_o;

        Lcd_h=val/100;
        Lcd_hr=val%100;
        Lcd_t=Lcd_hr/10;
        Lcd_o=Lcd_hr%10;

        Serial0_Out(Lcd_h+0x30);
        Serial0_Out(Lcd_t+0x30);
        Serial0_Out(Lcd_o+0x30);
}
```

PIC Microcontroller:

```
#include <pic.h>
#define _XTAL_FREQ 4000000
#include "PIC_LCD_NEW.h"
```

```
#include "pic_timer.h"
#include "pic_serial.h"

unsigned int count1,wait;
char rec[5],r;

#define s1_red RB2
#define s1_yellow RB1
#define s1_green RB0

#define s2_red RB5
#define s2_yellow RB4
#define s2_green RB3

#define s3_red RC0
#define s3_yellow RB7
#define s3_green RB6

#define s4_red RC3
#define s4_yellow RC2
#define s4_green RC1



unsigned char sec,dummy;


void second(int val)
{
        int t;
        t=val*20;
        wait=0;
        while(wait<=t);
}

void traffic()
{
                if(sec<=7)
                {
                        s1_red=0;   s1_yellow=0;       s1_green=1;
                        s2_red=1;   s2_yellow=0;       s2_green=0;
                        s3_red=1;   s3_yellow=0;       s3_green=0;
                        s4_red=1;   s4_yellow=0;       s4_green=0;
                }
                else if((sec>7)&&(sec<=10))
                {
                        s1_red=0;   s1_yellow=1;      s1_green=0;
                        s2_red=0;   s2_yellow=1;      s2_green=0;
                        s3_red=1;   s3_yellow=0;      s3_green=0;
                        s4_red=1;   s4_yellow=0;      s4_green=0;
                }
                else if((sec>10)&&(sec<=17))
                {
                        s1_red=1;   s1_yellow=0;      s1_green=0;
                        s2_red=0;   s2_yellow=0;      s2_green=1;
                        s3_red=1;   s3_yellow=0;      s3_green=0;
                        s4_red=1;   s4_yellow=0;      s4_green=0;
                }
                else if((sec>17)&&(sec<=20))
                {
                        s1_red=1;   s1_yellow=0;      s1_green=0;
                        s2_red=0;   s2_yellow=1;      s2_green=0;
                        s3_red=0;   s3_yellow=1;      s3_green=0;
                        s4_red=1;   s4_yellow=0;      s4_green=0;
                }
                else if((sec>20)&&(sec<=27))
                {
                        s1_red=1;   s1_yellow=0;      s1_green=0;
                        s2_red=1;   s2_yellow=0;      s2_green=0;
                        s3_red=0;   s3_yellow=0;      s3_green=1;
                        s4_red=1;   s4_yellow=0;      s4_green=0;
                }
                else if((sec>27)&&(sec<=30))
                {
                        s1_red=1;   s1_yellow=0;      s1_green=0;
                        s2_red=1;   s2_yellow=0;      s2_green=0;
                        s3_red=0;   s3_yellow=1;      s3_green=0;
                        s4_red=0;   s4_yellow=1;      s4_green=0;
                }
                else if((sec>30)&&(sec<=37))
                {
                        s1_red=1;   s1_yellow=0;      s1_green=0;
                        s2_red=1;   s2_yellow=0;      s2_green=0;
                        s3_red=1;   s3_yellow=0;      s3_green=0;
                        s4_red=0;   s4_yellow=0;      s4_green=1;
                }
```

```
                else if((sec>37)&&(sec<=40))
                {
                        s1_red=0;   s1_yellow=1;        s1_green=0;
                        s2_red=1;   s2_yellow=0;        s2_green=0;
                        s3_red=1;   s3_yellow=0;        s3_green=0;
                        s4_red=0;   s4_yellow=1;        s4_green=0;
                }
                else {}
}

void off()
{
        s1_red=s1_yellow=s1_green=0;
        s2_red=s2_yellow=s2_green=0;
        s3_red=s3_yellow=s3_green=0;
        s4_red=s4_yellow=s4_green=0;
}

void yellow()
{
        if(sec<=7)
        {
                s1_yellow=1;
                s1_red=0;
        }
        else if((sec>10)&&(sec<=17))
        {
                s2_yellow=1;
                s2_red=0;
        }
        else if((sec>20)&&(sec<=27))
        {
                s3_yellow=1;
                s3_red=0;
        }
        else if((sec>30)&&(sec<=37))
        {
                s4_yellow=1;
                s4_red=0;
        }
        else {}
}
void main()
```

```
{
	int i;
	TRISB=0x00;
	TRISC=0x80;
	TRISA=0x00;
	ADCON1=0x06;
	TRISD=0x00;

	PORTB=0x00;
	PORTC=0x00;

	s1_red=s1_yellow=s1_green=0;
	s2_red=s2_yellow=s2_green=0;
	s3_red=s3_yellow=s3_green=0;
	s4_red=s4_yellow=s4_green=0;
	Lcd8_Init();
	Lcd8_Display(0x80,"AMBULANCE CNTRL ",16);
	Lcd8_Display(0xc0,"TRAFFIC SYSTEM  ",16);
	Serial_Init(9600);

	Timer0_50ms();
	Receive(1);



	while (1)
	{
		//Lcd8_Decimal2(0xc0,r);
		//Lcd8_Decimal2(0x80,sec);
		if(r>=2)
		{
			r=0;
			if(rec[1]=='1')
			{
				Lcd8_Display(0x80,"HOSPITAL IS ON  ",16);
				Lcd8_Display(0xc0,"   RIGHT SIDE   ",16);
				dummy=sec;
				off();
				s1_yellow=1;	s2_yellow=1;	s3_yellow=1;
	s4_yellow=1;
				second(3);
				s1_yellow=0;	s2_yellow=0;	s3_yellow=0;
	s4_yellow=0;
```

```
				s1_green=s2_red=s3_red=s4_red=1;
				second(7);
				s1_yellow=1;s1_green=0;
				sec=dummy;
				yellow();
				second(3);
				sec=dummy;
			}
			else if(rec[1]=='2')
			{
				Lcd8_Display(0x80,"HOSPITAL IS ON  ",16);
				Lcd8_Display(0xc0,"   LEFT SIDE    ",16);
				dummy=sec;
				off();
				s1_yellow=1;	s2_yellow=1;	s3_yellow=1;
s4_yellow=1;
				second(3);
				s1_yellow=0;	s2_yellow=0;	s3_yellow=0;
s4_yellow=0;
				s2_green=s1_red=s3_red=s4_red=1;
				second(7);
				sec=dummy;
				s2_yellow=1;s2_green=0;
				yellow();
				second(3);
				sec=dummy;
			}
			else if(rec[1]=='3')
			{
				Lcd8_Display(0x80,"HOSPITAL IS ON  ",16);
				Lcd8_Display(0xc0,"  STRAIGHT ROAD ",16);
				off();
				s1_yellow=1;	s2_yellow=1;	s3_yellow=1;
s4_yellow=1;
				second(3);
				s1_yellow=0;	s2_yellow=0;	s3_yellow=0;
s4_yellow=0;
				s3_green=s2_red=s1_red=s4_red=1;
				second(7);
				sec=dummy;
				s3_yellow=1;s3_green=0;
				yellow();
				second(3);
```

```
                        sec=dummy;
                }
                else if(rec[1]=='4')
                {
                        Lcd8_Display(0x80,"HOSPITAL IS ON  ",16);
                        Lcd8_Display(0xc0,"   BACK SIDE    ",16);
                        dummy=sec;
                        off();
                        s1_yellow=1;        s2_yellow=1;        s3_yellow=1;
        s4_yellow=1;
                        second(3);
                        s1_yellow=0;        s2_yellow=0;        s3_yellow=0;
        s4_yellow=0;
                        s4_green=s2_red=s3_red=s1_red=1;
                        second(7);
                        sec=dummy;
                        s4_yellow=1;s4_green=0;
                        yellow();
                        second(3);
                        sec=dummy;
                }
                Delay(65000);
                Lcd8_Display(0x80,"AMBULANCE CNTRL ",16);
                Lcd8_Display(0xc0,"TRAFFIC SYSTEM  ",16);

            }
            else
            {
                traffic();
            }
        }
}

void interrupt isr()
{
        if(T0IF==1)
        {
                T0IF=0;
                TMR0 = 61;
                count1++;wait++;
                if(count1==20)
                {
                        sec++;
```

```
				if(sec==40) sec=0;
				count1=0;
			}
		}
		if(RCIF==1)
		{
			rec[r]=RCREG;
			if(rec[0]=='*') r++;
			RCIF=0;
		}
	}
```

5. FUTURE PLAN:

The system can be extended by introducing GSM module, which sends alert intimation to the respected hospital authorities stating that ambulance is near to the hospital so that hospital can make necessary arrangements for the treatment.

6. ADVANTAGES:

Provides priority to emergency vehicles. Helps in reaching the destination of the ambulance faster, that could save the lives of the people. Additionally, hospital informations are also provided. The traffic signal is controlled automatically upon ambulance arrival. On the other hand, driver can also manually control it in case of any error in the system function by using keypad.

7. CONCLUSION:

This project helps in controlling the traffic signal for ambulances thus saving many lives in case of emergencies. Even a second plays a vital role in saving human lives which induced us to design a system to control traffic signal and give way to nearest hospitals using controllers, Zigbee for communication, RFID and some basic components. This system has an automatic control of the traffic signal by detecting the presence of the ambulance.

8. REFERENCES:

- Chen Wenjie, Chen Lifeng, Chen Zhanglong, TuShiliang, “A realtime dynamic traffic control system based on wireless sensor network”, 34th International Conference on Parallel Processing Workshops (ICPP 2005 Workshops), pp. 258-264, 2005.
- M.Collotta ,A.Messineo ,G.Nicolosi, G.Pau, “A Self-Powered Bluetooth Network for Intelligent Traffic Light Junction Management, WSEAS Transactions on Information Science and Applications”,Vol. 11, pp. 12-23, 2014.
- M. Collotta, M. Denaro, G. Scat`a, A. Messineo, G. Nicolosi, “A self-powered wireless sensor network for dynamic management of queues at traffic lights”, Transport and Telecommunication,Vol. 15, Issue 1, pp. 42-52, 2014.
- Wen and Yang, “A dynamic and automatic traffic light control system for solving the road congestion problem” WIT Transactions on the Built Environment (Urban Transport). Vol.89, 2006, pp 307-316
- D.T. Dissanayake, S.M.R Senanayake, H.K.D.W.M.M.R Divarathne, B.G.L.T. Samaranayake, “Real-Time Dynamic Traffic LightTiming Adaptation Algorithm and Simulation Software”, International Conference on Industrial and Information Systems (ICIIS), pp. 563-567,2009.