Project Report

on

ARDUINO BASED ALARM CLOCK

BY

Devvart (2k20/IT/41)

Dharamveer (2k20/IT/42)

Under the Supervision of

Prof. Seba Susan

Department of Information Technology

Delhi Technological University

Jan – May, 2022

CERTIFICATE

I hereby certify that the work which is being presented in this project report entitled "**ARDUINO BASED ALARM CLOCK**", in partial fulfilment of the requirements for the award of the **Bachelor of Technology in Information Technology** is an authentic record of my own work carried out during a period from Jan 2022 to May 2022 under the supervision of Professor Seba Susan, Department of Information Technology.

The matter presented in this project report has not been submitted for the award of any other degree elsewhere.

Devvart (2k20/IT/41)

Dharamveer (2k20/IT/42)

Signature of Candidate

This is to certify that the above statement made by the candidates is correct to the best of my knowledge.

Date:26-04-2022 *Signature of Supervisor*

Prof. Seba Susan

Department of Information technology

Delhi Technological University, Delhi

TABLE OF CONTENTS

ABSTRACT

This Arduino based Real time clock is a digital clock to display real time using a RTC IC DS1307 which works on I2C protocol. Real time clock means it runs even after power failure. When power is reconnected, it displays the real time irrespective to the time and duration it was in off state. In this Arduino alarm clock project we have used a 16x2 LCD module to display the time in - "hour, minute, seconds, date, month and year" format. An Alarm option is also added and we can set up the alarm time. Once alarm time it saved in internal EEPROM of Arduino, it remains saved even after reset or electricity failure. Real time clocks are commonly used in our computers, houses, offices and electronics device for keeping them updated with real time.

INTRODUCTION

In this Arduino alarm clock project, we have used a 16x2 LCD module to display the time in - "hour, minute, seconds, date, month and year" format. An Alarm option is also added and we can set up the alarm time. Once alarm time it saved in internal EEPROM of Arduino, it remains saved even after reset or electricity failure. Real time clocks are commonly used in our computers, houses, offices and electronics device for keeping them updated with real time.

receives a message from the authorized mobile phone and the message is extracted by the microcontroller from the BLUETOOTH module and is displayed on the MATRIX display board. Serial to parallel communication is used for the entire process from WIFI module to Microcontroller and from microcontroller to the matrix display. And for the acknowledgement LCD display is used. This proposed system in this paper has many upcoming applications in educational institutions and organizations, crime prevention, traffic management, railways, advertisements etc. Been user friendly, long range and faster means of conveying information are major bolsters for this application. By using this proposed methodology we can enhance the security system and also make awareness of the emergency situations and avoid many dangers.

MATERIALS REQUIRED

1) Arduino UNO

2)LCD 16x2

3) RTC Module DS1307

4) BUZZER

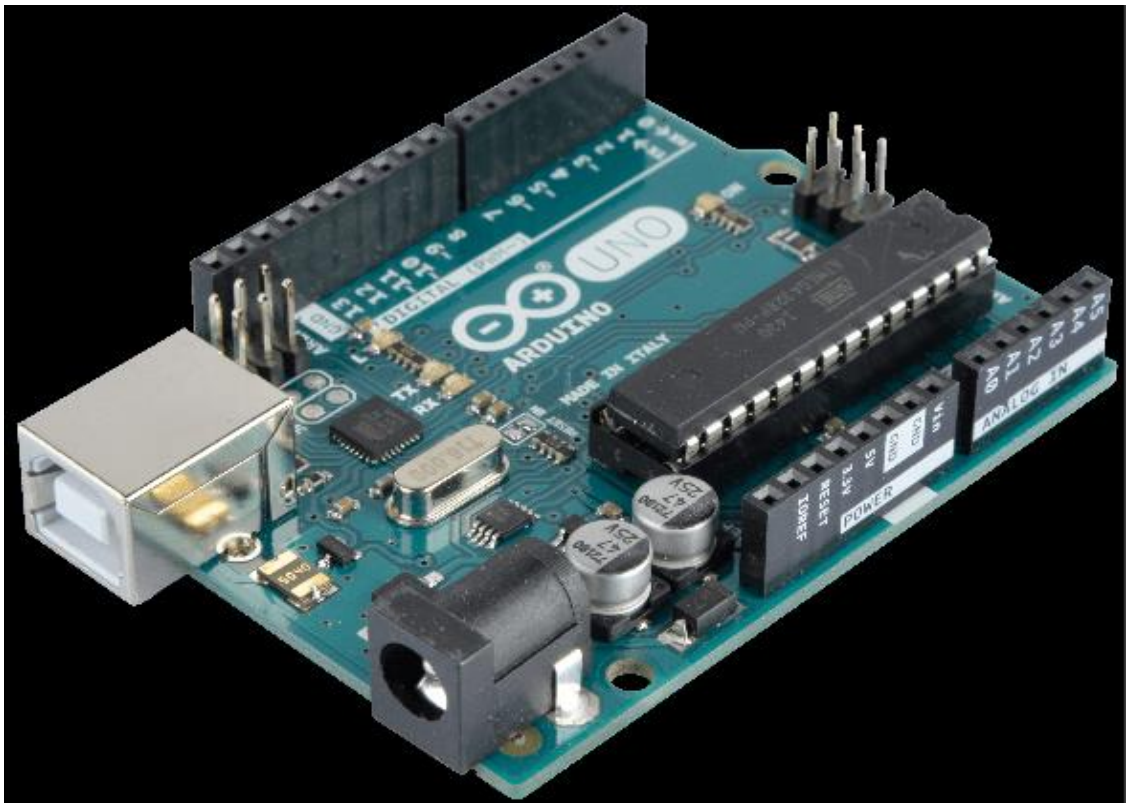
5) LEDs

6) WIRES

7) PUSH BUTTONS

ARDUINO UNO

The Arduino Uno is an open-source microcontroller board, developed by Arduino.cc.
The board is equipped with sets of digital and analogue input/output (I/O) pins that may be interfaced to various expansion boards and other circuits. The board has 14 digital I/O pins, 6 analogue I/O pins, and is programmable with the Arduino IDE (Integrated Development Environment), via a type B USB cable. It can be powered by the USB cable or by an external 9-volt battery, though it accepts voltages between 7 and 20 volts.



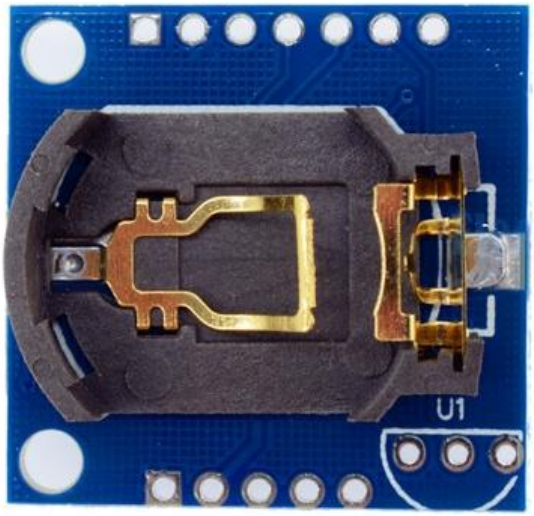
LCD 16x2

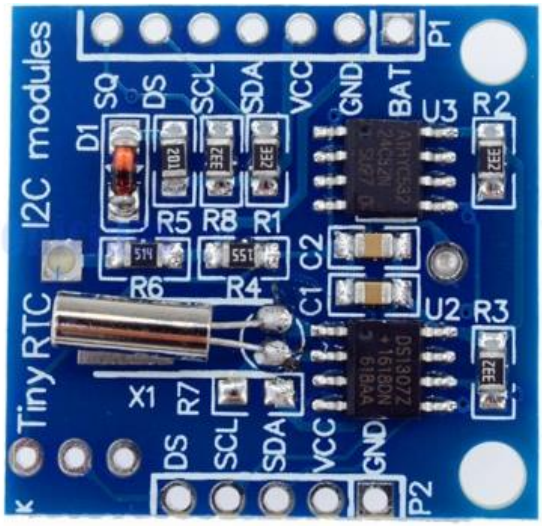
The term LCD stands for liquid crystal display. It is one kind of electronic display module used in an extensive range of applications like various circuits & devices like mobile phones, calculators, computers, TV sets, etc. These displays are mainly preferred for multi-segment light-emitting diodes and seven segments. The main benefits of using this module are inexpensive; simply programmable, animations, and there are no limitations for displaying custom characters, special and even animations, etc.

DS1307 RTC Module

Real Time Clock or RTC is a battery powered clock that measures time even when there is no external power or the microcontroller is reprogrammed.

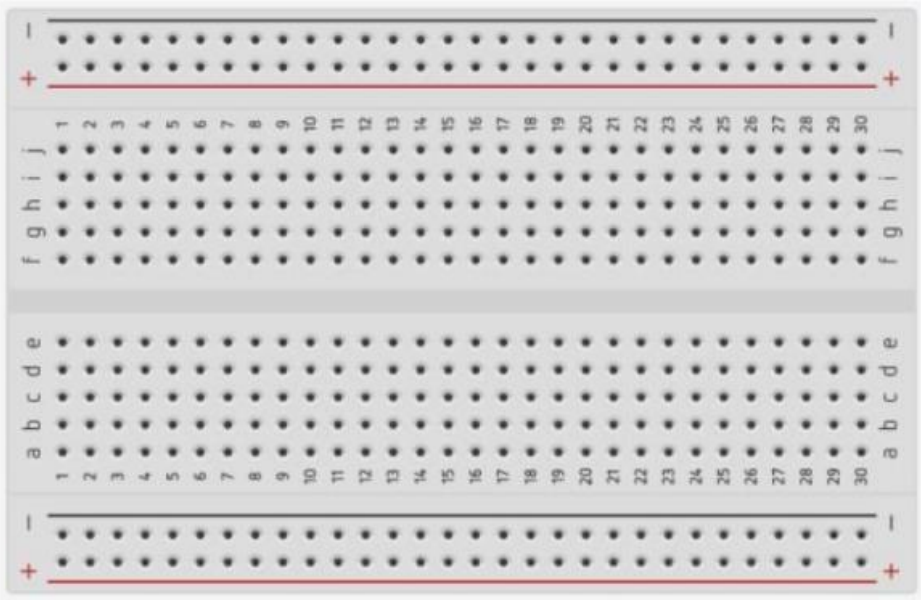
An RTC displays clock and calendar with all timekeeping functions. The battery, which is connected to the RTC is a separate one and is not related or connected to the main power supply



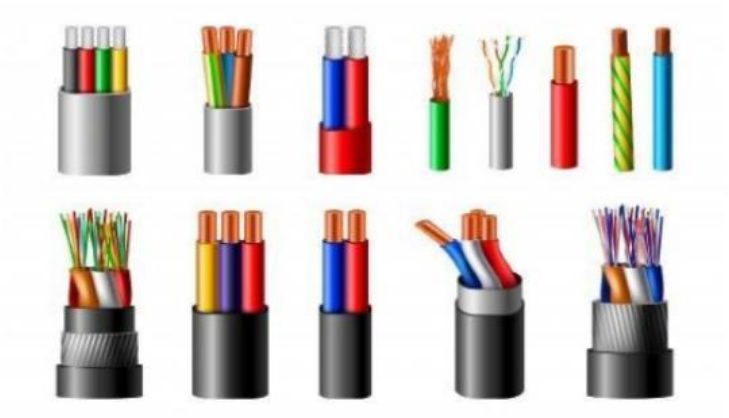


OTHER COMPONENTS REQUIRED

● Breadboard



● Connecting Wires



IMPLEMENTATION

In this Arduino based digital clock circuit, we have used three major components which are IC DS1307, Arduino Pro Mini Board and 16x2 LCD module.
Here arduino is used for reading time from ds1307 and display it on 16x2 LCD. DS1307 sends time/date using 2 lines to arduino. A buzzer is also used for alarm indication, which beeps when alarm is activated
As you can see in the circuit diagram, DS1307 chip pin SDA and SCL are connected to arduino pins SDA and SCL with pull up resistor that holds default value HIGH at data and clock lines.
16x2 LCD is connected with arduino in 4-bit mode. Control pin RS, RW and En are directly connected to arduino pin 2, GND and 3. And data pin D0-D7 is connected to 4, 5, 6, 7 of arduino. A buzzer is connected with arduino pin number 13.
Three buttons namely set, INC and Next are used for setting alarm to pin 12, 11 and 10 of arduino in active low mode. When we press set, alarm set mode activates and now we need to set alarm by using INC button and Next button is used for moving to digit.
I2C protocol is a method to connect two or more devices using two wires to a single system, and so this protocol is also called as two wire protocol. It can be used to communicate 127 devices to a single device or processor. Most of I2C devices run on100Khz frequency.

Steps for data writing master to slave (slave receiving mode)

1. Sends START condition to slave.
2. Sends slave address to slave.
3. Send write bit (0) to slave.
4. Received ACK bit from slave
5. Sends words address to slave.
6. Received ACK bit from slave
7. Sends data to slave.
8. Received ACK bit from slave.
9. And last sends STOP condition to slave.

Steps for data reading from slave to master (slave transmitting mode)

1. Sends START condition to slave.
2. Sends slave address to slave.
3. Send read bit (1) to slave.
4. Received ACK bit from slave
5. Received data from slave
6. Received ACK bit from slave.
7. Sends STOP condition to slave.

CIRCUIT DESIGN

The communication between microcontroller and RTC IC DS1307 is serial I2C bidirectional bus. I2C protocol is a method of communication between a faster device (Microcontroller or Arduino in this case) in master mode and a slower device (RTC) in slave mode.

There are two pins on Arduino for I2C communication. Analog pins 4 and 5 will act as SDA (Serial Data) and SCL (Serial Clock).

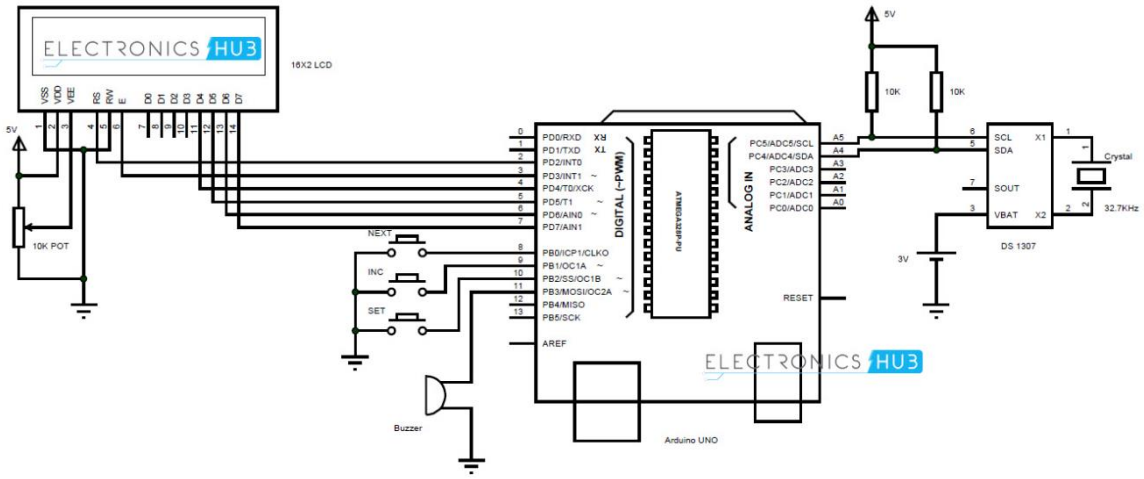
These are connected to respective SDA and SCL pins of RTC. Both these pins of RTC are pulled high using 10KΩ resistors.

An LCD is used to display the clock. 6 pins of LCD must be connected to Arduino. RS, E, D4, D5, D6 and D7 (Pins 4, 6, 11, 12, 13 and 14) of LCD are connected to pins 2, 3, 4, 5, 6 and 7 of Arduino.

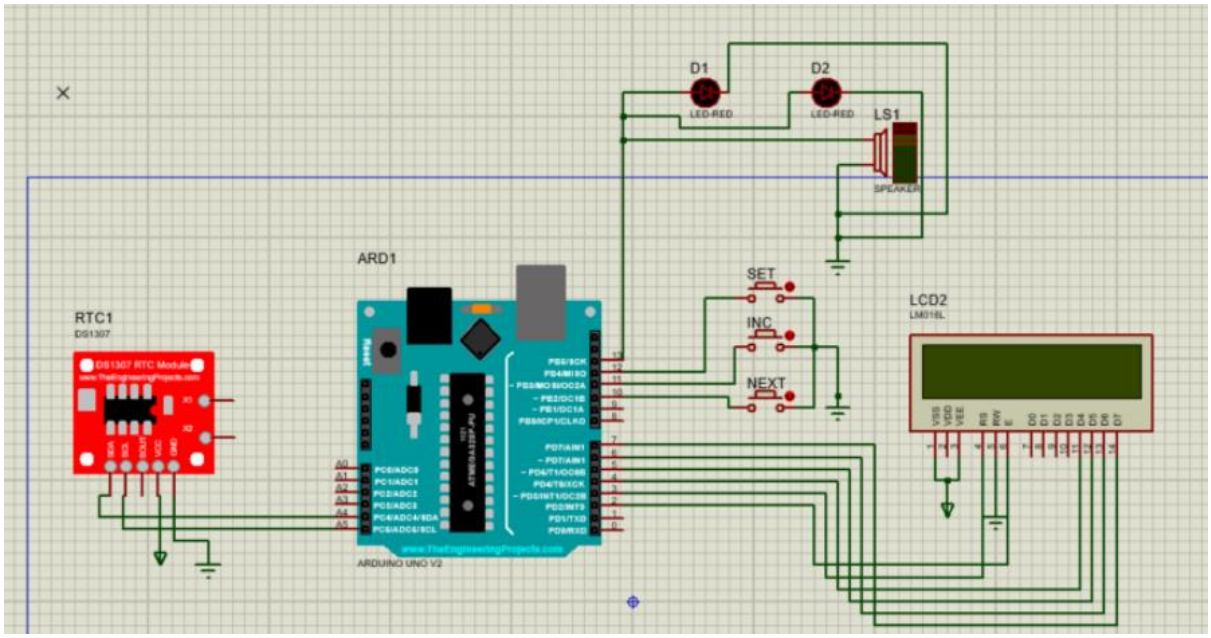
Three buttons are used to set the alarm. These buttons are connected to pins 8, 9 and 10 of Arduino. A buzzer is connected to pin 11 of Arduino that acts as an

alarm.

CIRCUIT DIAGRAM



SIMULATOR DIAGRAM



HOW DOES THE PROJECT WORKS?

The aim of this project is to create a real time clock along with an alarm feature. The working of the project is explained below.

All the connections are made as per the shown circuit diagram. The code for Arduino is uploaded and the LCD displays the current date and time.

In order to set the alarm, we press the set button. It'll go to alarm mode and asks for hours with current time being displayed. The increment button must be pressed must be pressed to change the hours.

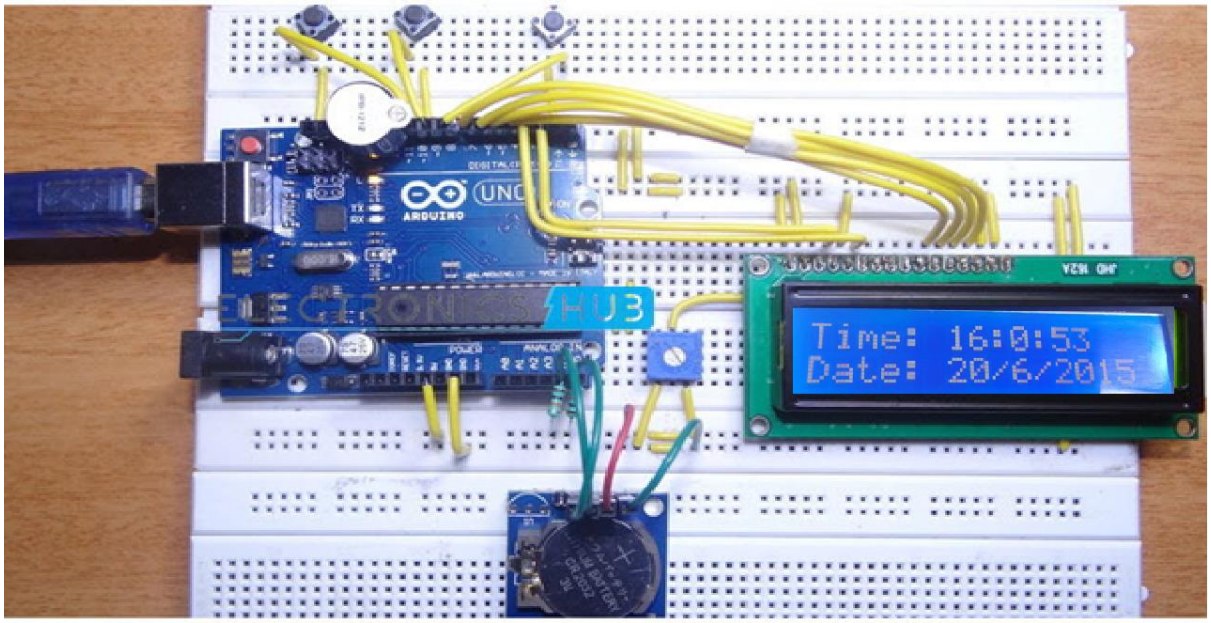
As the clock is in 24 hour format, the hours will be incremented between 0 and 23. Once the hours of the alarm is set, we must press the next button to go to minutes tab.

Again increment button is pressed to change the minutes. Once the alarm time is entered, set button is pressed and the alarm is set.

The values entered as alarm are stored in the EEPROM of the Arduino. These values are continuously compared with the present time.

When the stored values and current value match, the buzzer for the alarm will be triggered. In order to stop the alarm, the next button is pressed.

HARDWARE IMPLEMENTATION



CONCLUSION

In this project, we designed an Arduino based Real Time Clock with alarm. A Real Time Clock or RTC is a battery powered clock that measures time even when there is no external power or the microcontroller is reprogrammed.

An RTC displays clock and calendar with all timekeeping functions. The battery, which is connected to the RTC is a separate one and is not related or connected to the main power supply.

When the power is restored, RTC displays the real time irrespective of the duration for which the power is off. Such Real Time Clocks are commonly found in computers and are often referred to as just CMOS (Complementary Metal Oxide Semiconductor).

Most microcontrollers and microprocessors have built in timers for keeping time. But they work only when the microcontroller is connected to power supply.
When the power is turned on, the internal timers reset to 0. Hence, a separate RTC chip is included in applications like data loggers for example, which doesn't reset to 0 when the power is turned off or reset.

Real Time Clocks are often useful in data logging applications, time stamps, alarms, timers, clock builds etc. In this project, a Real Time Clock, which displays accurate time and date along with an alarm feature is designed.

One of the frequently used RTC ICs DS1307 is used in this project along with Arduino. The circuit, design and working are explained in the following sections.

REFERENCES

- https://papers.ssrn.com/sol3/papers.cfm?abstract_id=3915584

- https://codemint.net/electrical-engineering/design-and-implemetation-of-arduino-based-digital-clock-with-alarm/index.html
- https://www.engineersgarage.com/diy-arduino-based-alarm-clock/

SOURCE CODE

```
#include<Wire.h>
#include<EEPROM.h>
#include<RTClib.h>
#include<LiquidCrystal.h>

LiquidCrystal lcd(3,2,4,5,6,7);
RTC_DS1307 RTC;
int temp,inc,hours1,minut,add=11;
```

```
int next=10;
int INC=11;
int set_mad=12;

#define buzzer 13

int HOUR,MINUT,SECOND;

void setup()
{
Wire.begin();
RTC.begin();
lcd.begin(16, 2);
pinMode (INC, INPUT);
pinMode (next, INPUT);
pinMode (set_mad, INPUT);
pinMode (buzzer, OUTPUT);
digitalWrite (next, HIGH);
digitalWrite (set_mad, HIGH);
digitalWrite (INC, HIGH);

   lcd.setCursor(0,0);
   lcd.print("Real Time Clock");
   lcd.setCursor(0,1);
   lcd.print("Devvart IT-41");
   delay (2000);
    lcd.setCursor(0,1);
    lcd.print("Dharamveer IT-42");
   delay (2000);

 if(!RTC.isrunning())
 {
 RTC.adjust (DateTime (__DATE__,__TIME__));
 }
}

void loop()
```

```
{
  int temp=0, val=1, temp4;
  DateTime now = RTC.now();
  if (digitalRead (set_mad) == 0)    //set Alarm time
  {

   lcd.clear();
   lcd.setCursor(0,0);
   lcd.print(" Set Alarm ");
  delay (2000);
  defualt();
  time();
  delay(1000);
  lcd.clear();
  lcd.setCursor(0,0);
  lcd.print(" Alarm time ");
  lcd.setCursor(0,1);
  lcd.print(" has been set");
  delay (2000);
}

 lcd.clear();
 lcd.setCursor(0,0);
 lcd.print("Time: ");
 lcd.setCursor (6, 0);
 lcd.print (HOUR=now.hour(), DEC);
 lcd.print(":");
 lcd.print (MINUT=now.minute(), DEC);
 lcd.print(":");
 lcd.print (SECOND=now.second(), DEC);
 lcd.setCursor(0,1);
 lcd.print("Date: ");
 lcd.print (now.day(), DEC);
 lcd.print("/");
 lcd.print (now.month(), DEC);
 lcd.print("/");
 lcd.print (now. year(), DEC);
```

```
 match();
 delay(200);
}

void defualt ()
{
 lcd.setCursor(0,1);
 lcd.print (HOUR);
 lcd.print(":");
 lcd.print (MINUT);
 lcd.print(":");
 lcd.print(SECOND);
}



/*Function to set alarm time and feed time into Internal eeprom*/

void time()
{
 int temp=1, minuts=0, hours=0, seconds=0;
  while (temp==1)
  {
   if(digitalRead(INC)==0)
   {
    HOUR++;
    if(HOUR==24)
    {
     HOUR=0;
    }
    while (digitalRead(INC)==0);
   }
   lcd.clear();
    lcd.setCursor(0,0);
   lcd.print("Set Alarm Time ");
  //lcd.print(x);
   lcd.setCursor(0,1);
```

```
lcd.print (HOUR);
lcd.print(":");
lcd.print (MINUT);
lcd.print(":");
lcd.print (SECOND);
delay (100);
if(digitalRead (next)==0)
{
  hours1=HOUR;
  EEPROM.write(add++, hours1);
 temp=2;
 while (digitalRead (next)==0);
}
}
while (temp==2)
{
 if (digitalRead(INC)==0)
 {
  MINUT++;
  if(MINUT==60)
  {MINUT=0;}
  while(digitalRead (INC)==0);
 }
 // lcd.clear();
lcd.setCursor(0,1);
lcd.print (HOUR);
lcd.print(":");
lcd.print (MINUT);
lcd.print(":");
lcd.print (SECOND);
delay(100);
  if(digitalRead(next)==0)
  {
   minut=MINUT;
   EEPROM.write(add++,minut);
   temp=0;
   while(digitalRead(next)==0);
```

```
    }
  }
   delay(1000);
}


/* Function to chack alarm set time */

void match()
{
  int tem[17];
  for(int i=11;i<17;i++)
  {
    tem[i]=EEPROM.read(i);
  }
  if(HOUR == tem[11] && MINUT == tem[12])
  {
   beep ();
   beep ();
   beep ();
   beep ();
   lcd.clear();
   lcd.print("Wake Up.......");
   lcd.setCursor(0,1);
   lcd.print("Wake Up......");
   beep();
   beep();
   beep();
   beep();
  }
}

/* function to buzzer indication */

void beep()
{
   digitalWrite(buzzer,HIGH);
```

```
  delay(500);
  digitalWrite(buzzer,LOW);
  delay(500);
}
```