Department of the Electronics and Communication Engineering.

Aftabnagar, Dhaka , Bangladesh.

August,2015

Title of the Project:

Design and Development of Real Time Temperature and Solar Radiation Measurement System.

Submitted by:

Md. Abu Hemjal (2012-1-55-006)

Abdullah Al-Mahbub (2012-1-55-027)

Supervised by:

Dr. Md. Habibur Rahman

The Project Report Presented in Partial Fulfillment of the Requirement of the Degree of Bachelor of Science (Engineering) in Electronics and Telecommunication

Engineering.

Declaration

We hereby declare that this project is based on the results found by ourselves. We have completed the project on the topic entitled "**Design and Development of Real Time Temperature and Solar Radiation Measurement System**" as well as prepared as research report to the Department of Electronic and Communication Engineering of East West University in partial fulfillment of the requirement for the degree of B.Sc. in Electronic and Telecommunication Engineering, under the supervision of Dr. Md. Habibur Rahman, Professor, Department of Electronics and Electrical Engineering, University of Dhaka.

Signature

.....
Dr. Md. Habibur Rahman
Professor, EEE
University of Dhaka

Signature

.....
Abdullah Al-Mahbub
ID No.:2012-1-55-027

.....
Md. Abu Hemjal
ID No.:2012-1-55-006

Acceptance

This project has been prepared and submitted by Md. Abu Hemjal, ID. 2012-1-55-006 and Abdullah Al -Mahbub, ID. 2012-1-55-027. This project report presented to the Department of Electronics and Communication Engineering, East West University is submitted in partial fulfillment of the requirement for the degree of B.sc. In Electronics and Telecommunication Engineering, Under complete supervision of the undersigned.

Signature

.....

Dr. Md. Habibur Rahman

Professor, EEE,

University of Dhaka

Acknowledgement

First of all we wish to convey our heart full thanks to **Almighty Allah** because we completed this project successfully. This project consumed huge amount of work, research and dedication. Still, implementation would not have been possible if we did not have a support of many individuals and organizations. Therefore we would like to extend our sincere gratitude to all of them. First of all we are thankful to our supervisor Dr**. Md. Habibur Rahman** for his logistical support and for providing necessary guidance concerning projects implementation. Without his superior knowledge and experience, the Project would like in quality of outcomes, and thus his support had been essential. We would like to express our sincere thanks towards volunteer researchers on online and our senior researchers who devoted their time and knowledge in the implementation of this project.

Nevertheless, we express our gratitude toward our families and project partners for their kind co-operation and encouragement which help us in completion of this project.

Abstract

Solar energy is one of the solutions to the energy crisis as it is renewable and has no Environmental hazards. Thus, we investigated the dye sensitized solar cell aiming at introducing a clean and cheap energy resource measuring system. We started our investigation using the conditions recommended in the literature, i.e. using ATmega328p microcontroller based Arduino Uno board, Real Time Clock (DS3231), LM35 Temperature sensor, Solar cell with an operational amplifier.
Measuring solar radiation is essential for determining the viability of a solar system in a particular surface.

Content:

Chapter One

INTRODUCTION

1.1 Motivation:

The Sun is the star at the center of the Solar System and is by far the most important source of energy for life on Earth. It is a nearly perfect spherical ball of hot plasma, with internal convective motion that generates a magnetic field via a dynamo process. Its diameter is about 109 times that of Earth, and it has a mass about 330,000 times that of Earth, accounting for about 99.86% of the total mass of the Solar System. Chemically, about three quarters of the Sun's mass consists of hydrogen; the rest is mostly helium, with much smaller quantities of heavier elements, including oxygen, carbon, neon and iron. Solar energy can be used to heat and cool your home, but it has almost no impact on the global climate. By comparison, electricity generated by power plants produces carbon dioxide emissions that scientists say pose serious threats to the environment. The sun is a sphere of intensity warm gaseous matter with a diameter $1.39*10^{9}$m and $1.5*10^{11}$m distance from earth. So, we have to motivated to work about solar energy and radiation measurement and to learning, knowing and experiencing about sun power.

1.2 Aim of the project:

Every project must have a definite aim in its way. An aimless way is just like a boat without a rudder. That means the boat without rudder cannot reach ashore. The aim of the project is to find the solar radiation in real time and temperature in real time, how much energy will produce by solar in a particular area and how to display those data.

1.3 Project in Brief:

1.3.1 Development of circuit:

At first we have known about all the components, related to the work. Then we have gained the basic knowledge. Finally we have applied our merit and creative technical knowledge to develop the devices and the system.

1.3.2 Documentation of Data:

Here, all data those had been collected which have recorded and stored. We have maintained sincerity during taking and

storing the data. After approval from the supervisor, all the data have been implemented into the final project report.

1.4 Outline of the Project:

There are five chapters in this book that describes in details. Chapter one is the introduction. It contains the motivation, aim of the project, development of the circuit, documentation of the data, and outline of the project. Chapter two contains the theoretical framework of the project.

This chapter includes various important theoretical frameworks such as solar radiation, solar constant, measuring solar radiations, pyranometer, LM35 dz temperature sensor, DS3231 real time clock module, solar cell, microcontroller, Arduino Uno device overview, LCD display, etc. Chapter three contains system design, hardware design, and software design. Chapter four focuses the results of the project. Chapter five focuses on the overview and conclusion of the project and included appendix in the last section.

Chapter Two

THEORETICAL FRAMEWORK

2.1 Solar Radiation:

Solar radiation is radiant energy emitted by the sun, particularly electromagnetic energy. About half of the radiation is in the visible short-wave part of the electromagnetic spectrum. The other half is mostly in the near-infrared part, with some in the ultraviolet part of the spectrum.

It is an important source of renewable energy and its technologies are broadly characterized as either passive solar or active solar depending on the way they capture and distribute solar energy or convert it into solar power. Active

solar techniques include the use of photovoltaic systems, concentrated solar power and solar water heating to harness the energy. Passive solar techniques include orienting a building to the Sun, selecting materials with favorable thermal mass or light dispersing properties, and designing spaces that naturally circulate air.

The large magnitude of solar energy available makes it a highly appealing source of electricity. The United Nations Development Programme in its 2000 World Energy Assessment found that the annual potential of solar energy was 1575 – 49387 EJ. This is larger than the total world energy consumption (the total energy consumed by the world within a year) in 2012, which was 559.8 EJ.

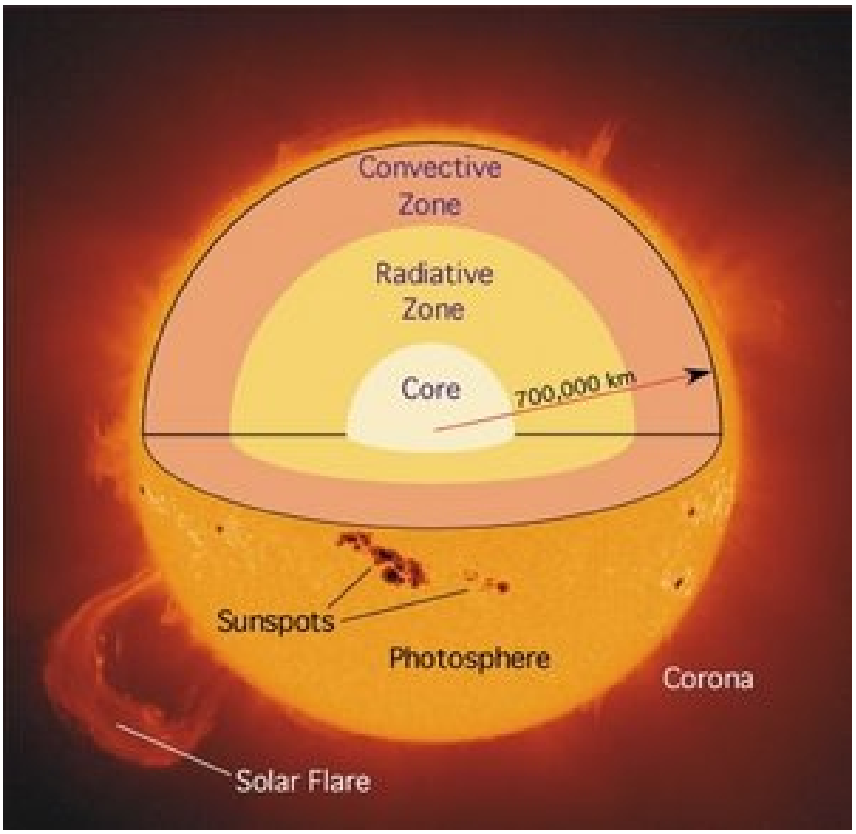


Figure: Source of the energy in sun

The Earth receives 174,000 terawatts (TW) of incoming solar

radiation (insolation) at the upper atmosphere. Approximately 30% is reflected back to space while the rest is absorbed by clouds, oceans and land masses. The spectrum of solar light at the Earth's surface is mostly spread across the visible and near-infrared ranges with a small part in the near-ultraviolet. Most people around the world live in areas with insolation levels of 150 to 300 watt per square meter or 3.5 to 7.0 kWh/m-2 per day.

Earth's land surface, oceans and atmosphere absorb solar radiation, and this raises their temperature. Warm air containing evaporated water from the oceans rises, causing atmospheric circulation or convection. When the air reaches a high altitude, where the temperature is low, water vapor condenses into clouds, which rain onto the Earth's surface, completing the water cycle. The latent heat of water condensation amplifies convection, producing atmospheric phenomena such as wind, cyclones and anti-cyclones. Sunlight absorbed by the oceans and land masses keeps the surface at an average temperature of 14 °C. By photosynthesis green plants convert solar energy into chemical energy, which produces food, wood and the biomass from which fossil fuels are derived.

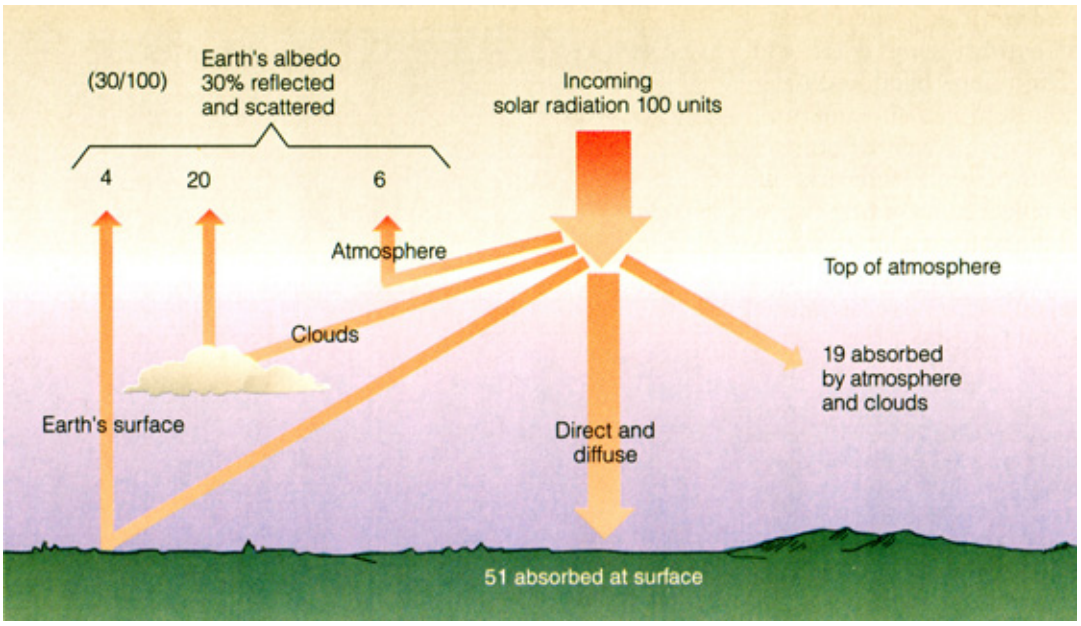


Figure: Solar Energy on the earth

2.2 Measuring solar radiation

Two types of basic instruments are used for measuring solar

radiation. They are Pyranometer and Pyrheliometer.

2.3 Pyranometer

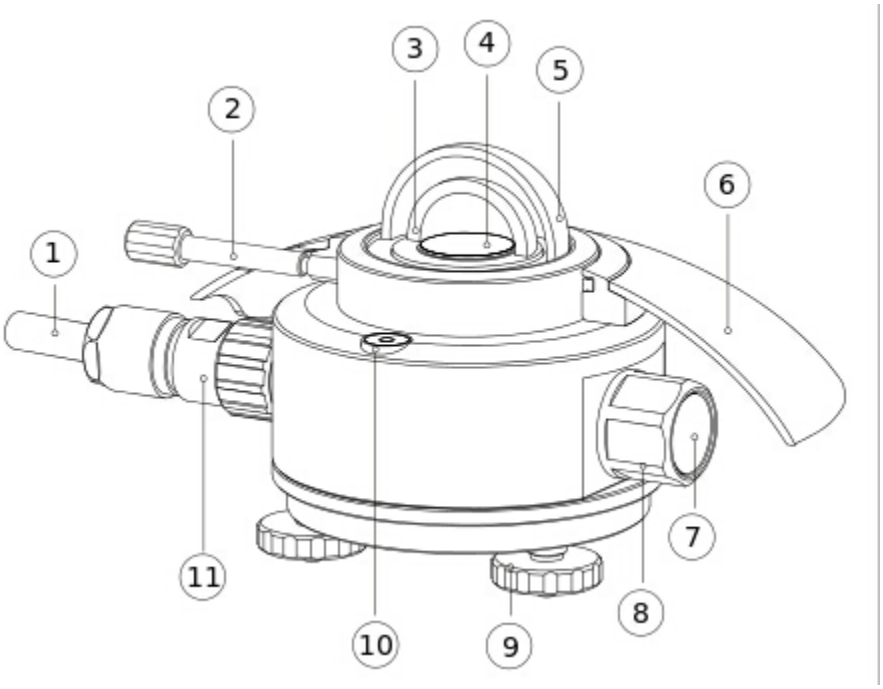
A pyranometer is a type of actinometer used to measure broadband solar irradiance on a planar surface and is a sensor that is designed to measure the solar radiation flux density (W/m^2) from a field of view of 180 degrees.

Fig: Pyranometer

The solar radiation spectrum extends approximately from 300 to 2,800 nm. Pyranometer usually cover that spectrum with a spectral sensitivity that is as flat as possible. Above

Pyranometer does not require any power, it supplies a low voltage of 0-20 mV in relation to the amount of incoming radiation.

2.4 Design of Pyranometer



The line drawing showing pyranometer properties and functions (1) cable, (3) glass inner dome, (4) thermopile sensor, (5) glass outer dome, (7) humidity indicator with desiccant, (11) connector.

In order to attain the proper directional and spectral characteristics, a pyranometer's main components are:

- A thermopile sensor with a black coating. This sensor absorbs all solar radiation, has a flat spectrum covering the 300 to 50,000 nanometer range, and has a near-perfect cosine response.

- A glass dome. This dome limits the spectral response from 300 to 2,800 nanometers (cutting off the part above 2,800 nm), while preserving the 180 degrees field of view. Another function of the dome is that it shields the thermopile sensor from convection.

The black coating on the thermopile sensor absorbs the solar radiation. This radiation is converted to heat. The heat flows through the sensor to the pyranometer housing. The thermopile sensor generates a voltage output signal that is proportional to the solar radiation.

2.4.A Uses of Pyranometer

Pyranometers are frequently used in meteorology, climatology, solar energy studies and building physics. They can be seen in many meteorological stations - typically installed horizontally and next to solar panels - typically mounted with the sensor surface in the plane of the panel.

2.5 Solar cell

A solar cell, or photovoltaic cell, is an electrical device that converts the energy of light directly into electricity by the photovoltaic effect, which is a physical and chemical phenomenon. It is a form of photoelectric cell, defined as a device whose electrical characteristics, such as current, voltage, or resistance, vary when exposed to light. Solar cells are the building blocks of photovoltaic modules, otherwise known as solar panels.

Solar cells are described as being photovoltaic irrespective of whether the source is sunlight or an artificial light. They are used as a photodetector detecting light or other

electromagnetic radiation near the visible range, or measuring light intensity.

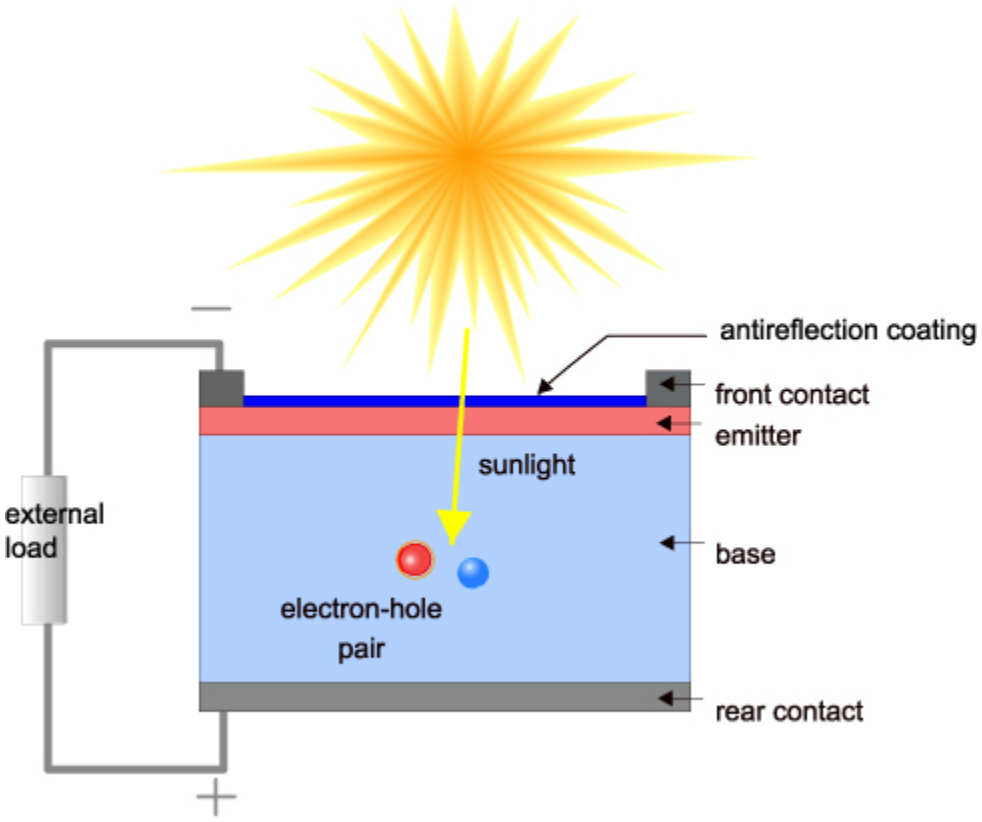


Fig : Solar cell

The basic steps in the operation of a solar cell are:

- the generation of light-generated carriers
- the collection of the light-generated carries to generate a current
- the generation of a large voltage across the solar cell
- the supply of power in the load.

2.6 Microcontroller

2.6.A what is microcontroller

A microcontroller is a small computer on a single integrated circuit containing a processor core, memory, and programmable input/output peripherals. A microcontroller is a self-contained system with peripherals, memory and a processor that can be used as an embedded system. Most programmable microcontrollers that are used today are embedded in other consumer products or machinery including phones, peripherals, automobiles and household appliances for computer systems. Due to that, another name for a microcontroller is "embedded controller." Some embedded systems are more sophisticated, while others have minimal requirements for memory and programming length and a low software complexity. Input and output devices include solenoids, LCD displays, relays, switches and sensors for data like humidity, temperature or light level, amongst others.

2.6.B How we select a microcontroller?

Amount of RAM and ROM.
Reliable sources of MCU.
Cost effective.
Maximum speed of microcontroller.
Avaibility of software and hardware.
A timer module that allow working on real time.
A serial I/O port.
An ADC to allow acceptance of analog input data for processing

2.7 Arduino Uno
2.7.A Device overview

Arduino is a tool for making computers that can sense and control more of the physical world than our 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.

Fig: Arduino Uno.

The Arduino platform has become quite popular with people just starting out with electronics, and for good reason. Unlike most previous programmable circuit boards, the Arduino does not need a separate piece of hardware (called a programmer) in order to load new code onto the board – you can simply use a USB cable. Additionally, the Arduino IDE uses a simplified version of C++, making it easier to learn to

program. Finally, Arduino provides a standard form factor that breaks out the functions of the micro-controller into a more accessible package.

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 communicate with software running on our computer (e.g. Flash, Processing, MaxMSP.) The boards can be assembled by hand or purchased preassembled; the open-source IDE can be downloaded for free.

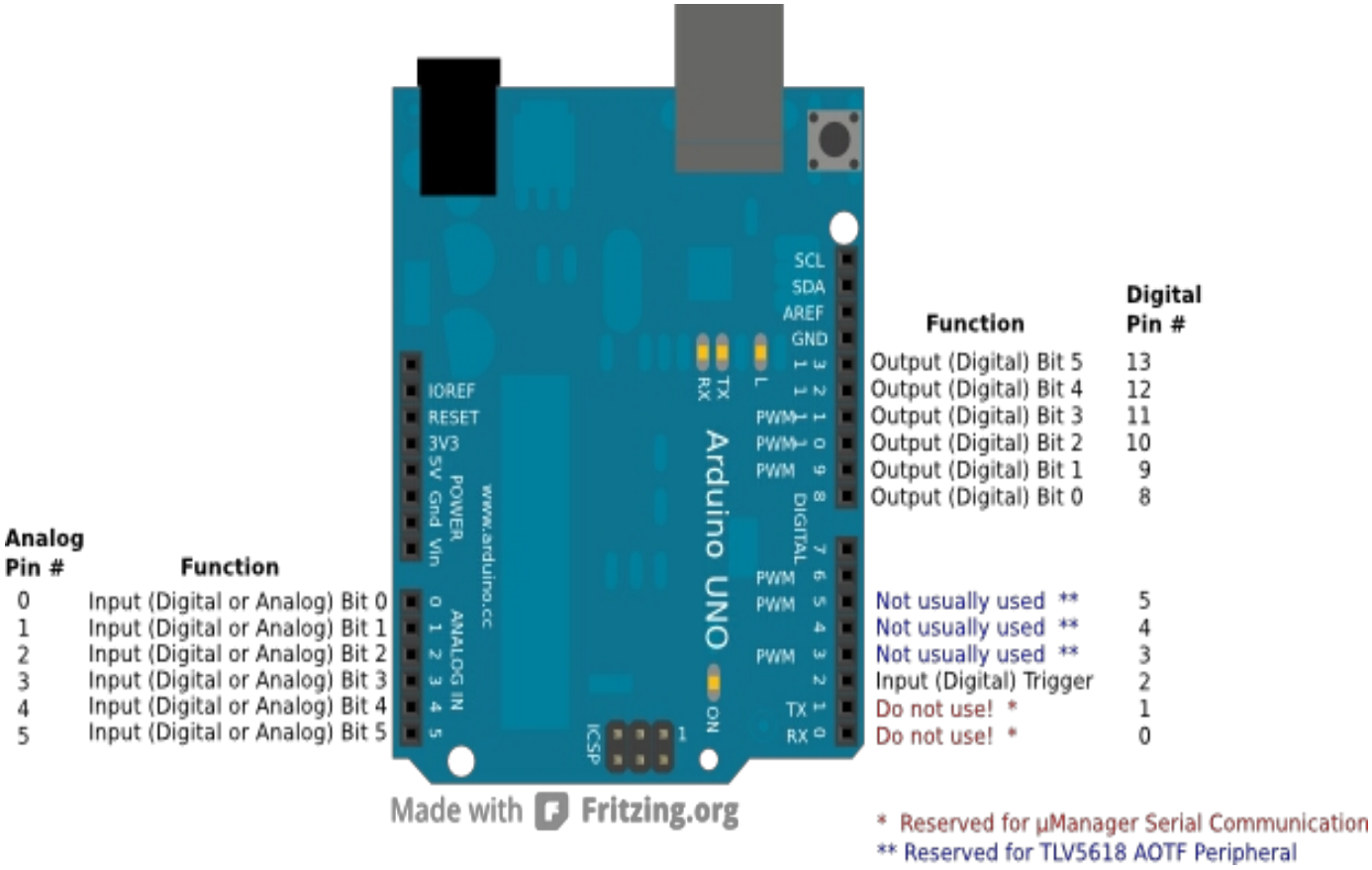
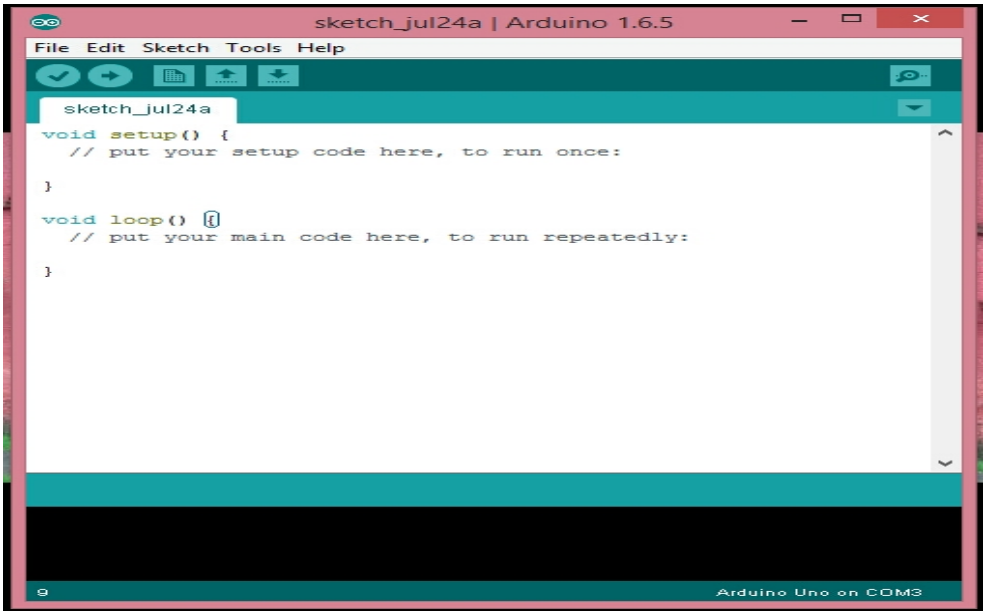


Fig: A typical Arduino Uno board showing all of its I/O ports.

2.7.B Arduino IDE

The Arduino Integrated Development Environment - or Arduino Software (IDE) - contains a text editor for writing code, a message area, a text console, a toolbar with buttons for common functions and a series of menus. It connects to the Arduino and Genuino hardware to upload programs and communicate with them. The open-source Arduino Software (IDE) makes it easy to write code and upload it to the board. It runs on Windows, Mac OS X, and Linux. The environment is written in Java and based on Processing and other open-source software.



1. Upload: Compiles our code

and uploads it to the configured board.

2. Verify: Checks our code for errors compiling it.

3. New: Creates a new sketch.

4. Open: Presents a menu of all the sketches in our sketchbook. Clicking one will open it within the current window overwriting its content.

5. Save: Saves our sketch.

6. Serial Monitor: Opens the serial monitor.

Additional commands are found within the five menus: File, Edit, Sketch, Tools, Help. The menus are context sensitive, which means only those items relevant to the work currently being carried out are available.

2.8 LCD display (2*16)

LCD (Liquid Crystal Display) screen is an electronic display module and find a wide range of applications. A 16x2 LCD display is very basic module and is very commonly used in various devices and circuits. These modules are preferred over seven segments and other multi segment LEDs. The reasons being: LCDs are economical; easily programmable; have no limitation of displaying special & even custom

characters, animations and so on.
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.
The command register stores the command instructions given to the LCD. A command is an instruction given to LCD to do a predefined task like initializing it, clearing its screen, setting the cursor position, controlling display etc. The data register stores the data to be displayed on the LCD. The data is the ASCII value of the character to be displayed on the LCD. Click to learn more about internal structure of a LCD.

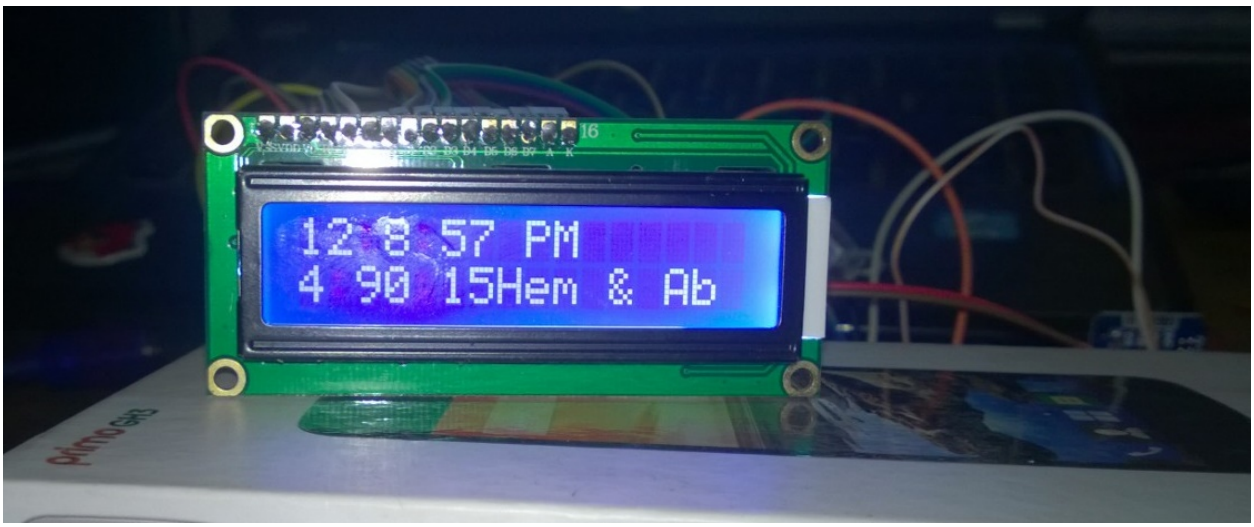
Fig: A simple LCD display(16x2)

Pin Description:

Pin No	Function	Name
1	Ground (0V)	Ground
2	Supply voltage; 5V (4.7V – 5.3V)	Vcc
3	Contrast adjustment; through a variable resistor	V_{EE}
4	Selects command register when low; and data register when high	Register Select
5	Low to write to the register; High to read from the register	Read/write
6	Sends data to data pins when a high to	Enable

	low pulse is given	
7-14	8-bit data pins	DB0
15	Backlight V_{CC} (5V)	Led+
16	Backlight Ground (0V)	Led-

Our first attempt to write a program and showing it on lcd successfully to understand how to control lcd display and understand how the codes works on the display:



2.9 DS3231 IC:

The DS3231 is a low-cost, extremely accurate I2C real-time clock (RTC) with an integrated temperature-compensated crystal oscillator (TCXO) and crystal. The device incorporates a battery input, and maintains accurate timekeeping when main power to the device is interrupted. The integration of the crystal resonator enhances the long-term accuracy of the device as well as reduces the piece-part count in a manufacturing line. The DS3231 is available in commercial and industrial temperature ranges, and is offered in a 16-pin,

300-mil SO package.

The RTC maintains seconds, minutes, hours, day, date, month, and year information. The date at the end of the month is automatically adjusted for months with fewer than 31 days, including corrections for leap year. The clock operates in either the 24-hour or 12-hour format with an AM/PM indicator. Two programmable time-of-day alarms and a programmable square-wave output are provided. Address and data are transferred serially through an I2C bidirectional bus.

Please take note that here we had used a DS3231 module that comes with the built in power back up system to avoid extra hassle.

Fig: RTC module.

Benefits and Features :

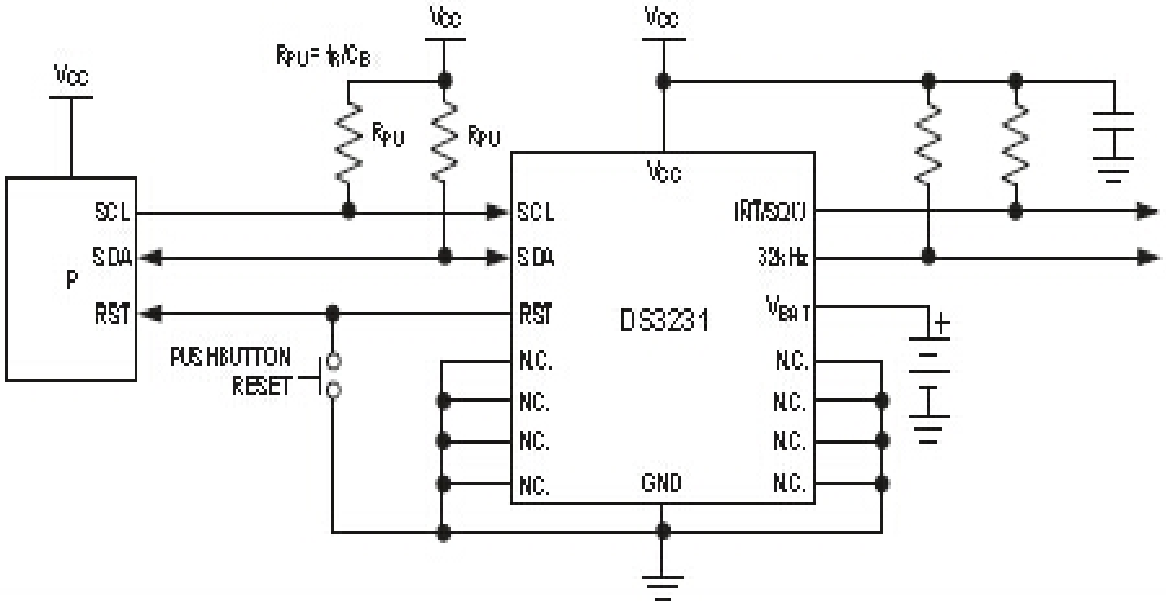
1. Highly Accurate RTC Completely Manages All Timekeeping Functions Real-Time Clock Counts Seconds, Minutes, Hours, Date of the Month, Month, Day of the Week, and Year, with Leap-Year Compensation Valid Up to 2100
2. Accuracy ±2ppm from 0°C to +40°C
3. Accuracy ±3.5ppm from -40°C to +85°C
4. Digital Temp Sensor Output: ±3°C Accuracy Register for

Aging Trim
5. RST Output/Pushbutton Reset Denounce Input
6. Two Time-of-Day Alarms
7. Programmable Square-Wave Output Signal
8. Simple Serial Interface Connects to Most Microcontrollers Fast (400kHz) I2C Interface
9. Battery-Backup Input for Continuous Timekeeping Low Power Operation Extends Battery-Backup Run Time3.3V Operation.
10. Operating Temperature Ranges: Commercial (0°C to +70°C) and Industrial (-40°C to +85°C) Underwriters Laboratories® (UL) Recognized.

Applications :

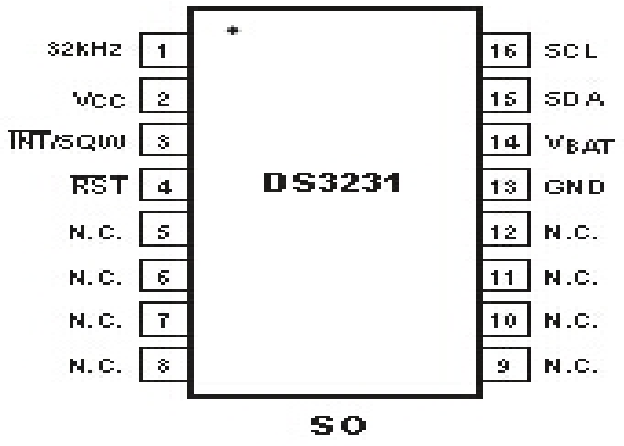
1.Servers
2.Telematics
3.Utility Power Meters

Typical Operating Circuit:

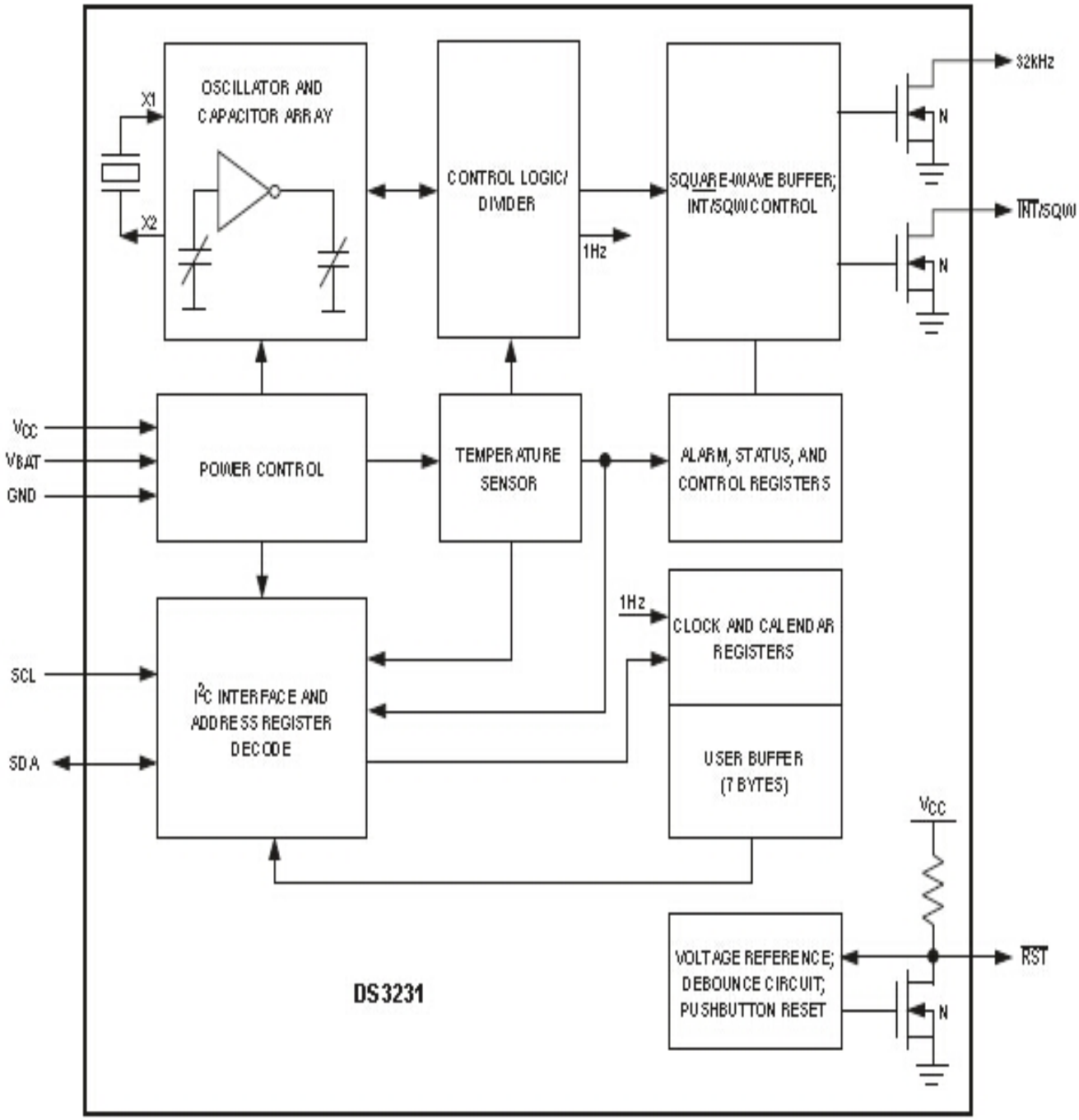


Pin configurations:

TOP VIEW

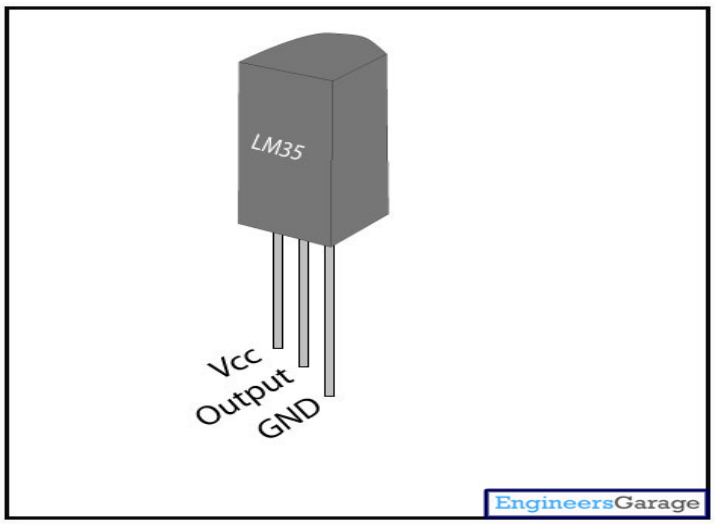


Block diagram:


OSCILLATOR AND CAPACITOR ARRAY
X1
X2
CONTROL LOGIC/ DIVIDER
1Hz
SQUARE-WAVE BUFFER; INT/SQW CONTROL
32kHz
INT/SQW
N
N
VCC
VBAT
GND
POWER CONTROL
TEMPERATURE SENSOR
ALARM, STATUS, AND CONTROL REGISTERS
1Hz
CLOCK AND CALENDAR REGISTERS
SCL
SDA
I2C INTERFACE AND ADDRESS REGISTER DECODE
USER BUFFER (7 BYTES)
VCC
VOLTAGE REFERENCE; DEBOUNCE CIRCUIT; PUSHBUTTON RESET
RST
N
DS3231

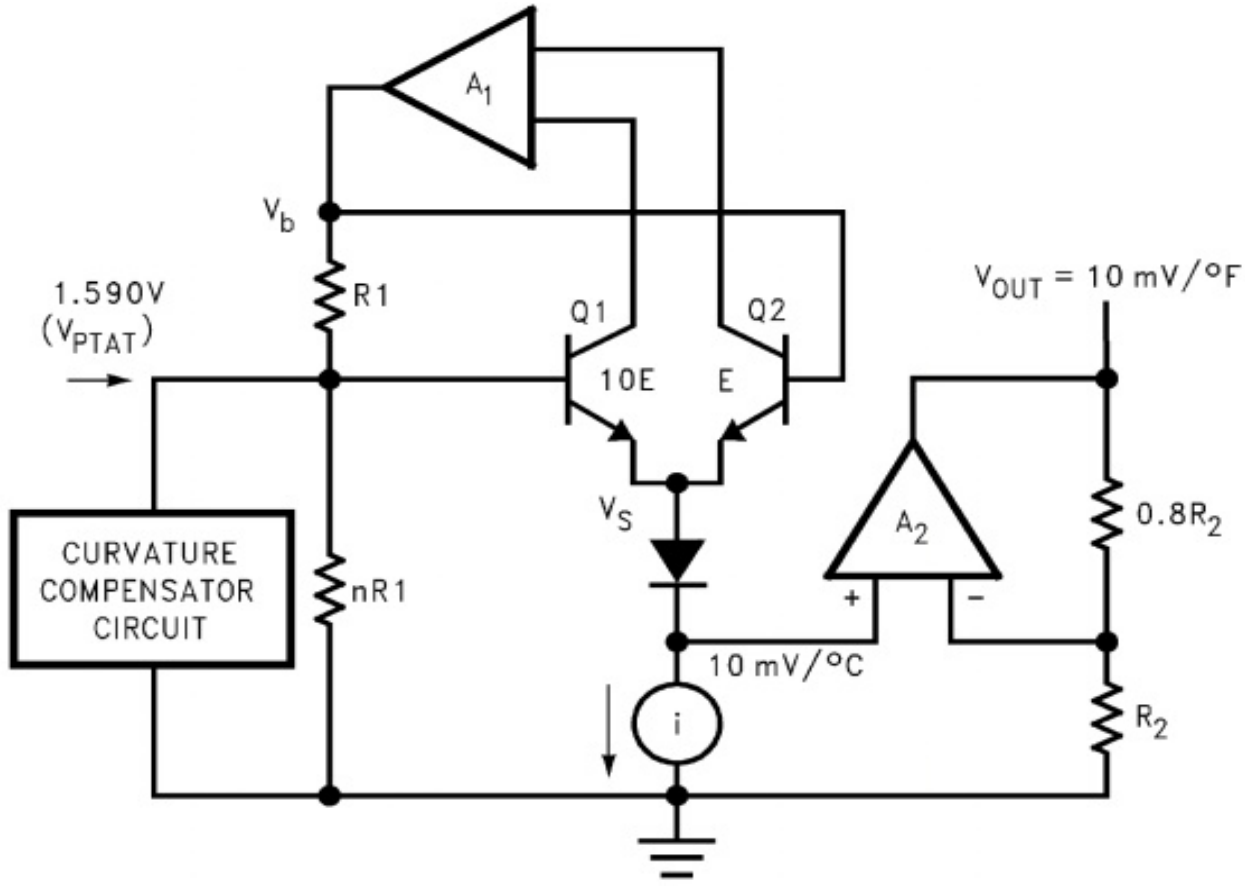
2.10 LM 35

LM 35 device is used as the temperature sensor. It is a simple IC and it has only three pins.



The LM35-series devices are precision integrated-circuit temperature sensors, with an output voltage linearly proportional to the Centigrade temperature. The LM35 device has an advantage over linear temperature sensors calibrated in Kelvin, as the user is not required to subtract a large constant voltage from the output to obtain convenient Centigrade scaling. The LM35 device does not require any external calibration or trimming to provide typical accuracies of ± ¼ °C at room temperature and ± ¾ °C over a full −55°C to 150°C temperature range. Lower cost is assured by trimming and calibration at the wafer level. The low output impedance, linear output, and precise inherent calibration of the LM35 device makes interfacing to readout or control circuitry especially easy. The device is used with single power supplies, or with plus and minus supplies. As the LM35 device draws only 60 µA from the supply, it has

very low self-heating of less than 0.1°C in still air. The LM35 device is rated to operate over a −55°C to 150°C temperature range, while the LM35C device is rated for a −40°C to 110°C range (−10° with improved accuracy). The temperature-sensing element is comprised of a delta-V BE architecture. The temperature-sensing element is then buffered by an amplifier and provided to the VOUT pin. The amplifier has a simple class A output stage with typical 0.5-Ω output impedance as shown in the Functional Block Diagram. Therefore the LM35DZ can only source current and it's sinking capability is limited to 1 µA.



Chapter Three

SYSTEM DESIGN

3.1 Hardware design

3.1.A Sensor

The sensor converts into current to voltage. As regards the circuit's inputs, we prefer current to voltage as an input quantity. Only some devices has a current input. In these cases, we connect a current to voltage converter before the device; as a result, it acquires a current input and voltage output.

3.1.B Signal conditioning circuit

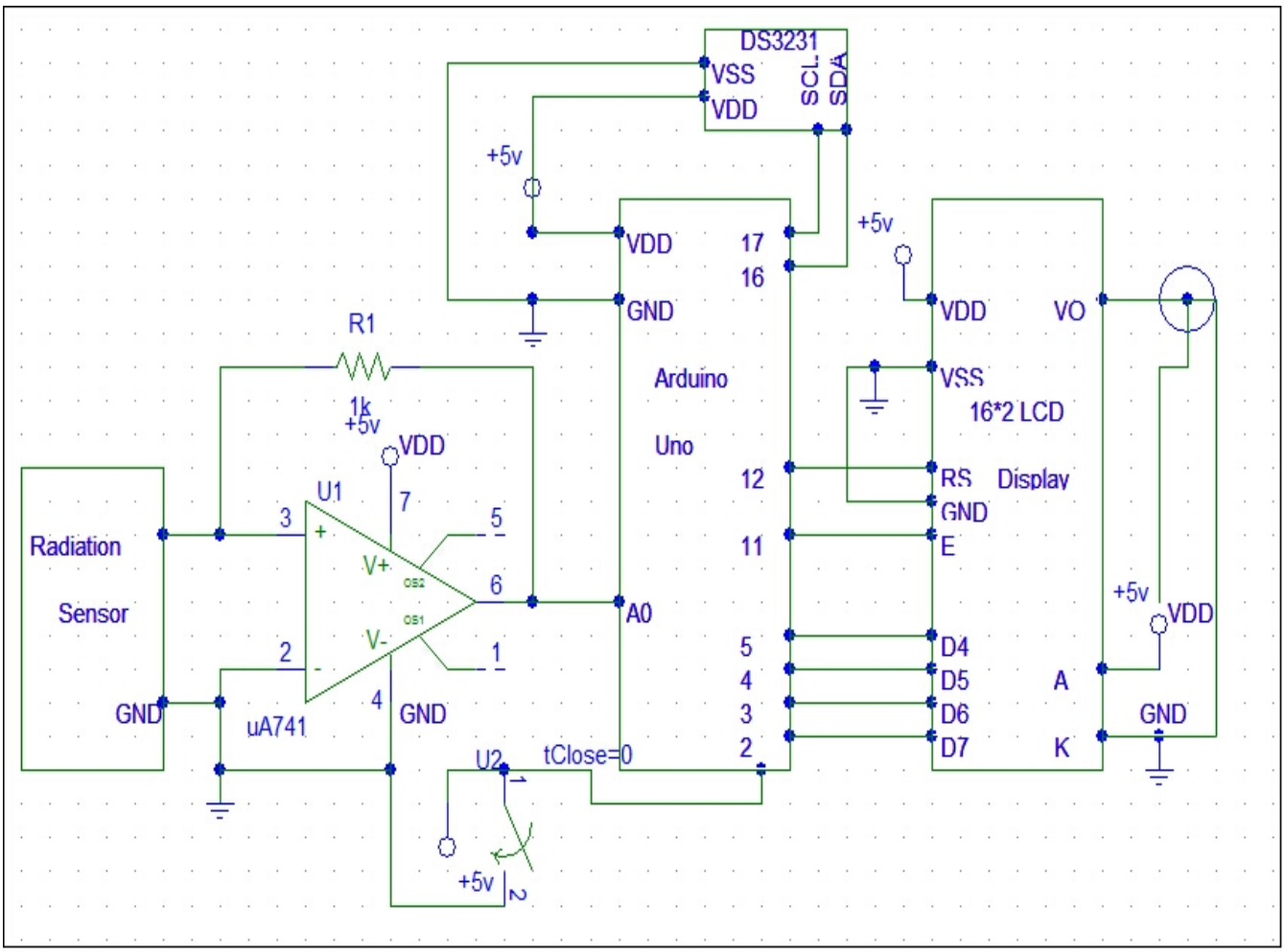
This circuit converts an input current to a proportional amount of voltage. Use the two switches to select an input current. This current flows across an 1k resistor. The op-amp outputs a voltage equal to the voltage drop across the resistor in order to ensure that the terminal is at ground, which means that the output voltage is proportional to the resistor voltage.

3.1.C ADC and Control:

Analog to digital converted to the radiation data and control the programming the system of total overview of system.Two modes works here for high mode it will calculate the instantaneous value and for low mode it will calculate the integrated value.

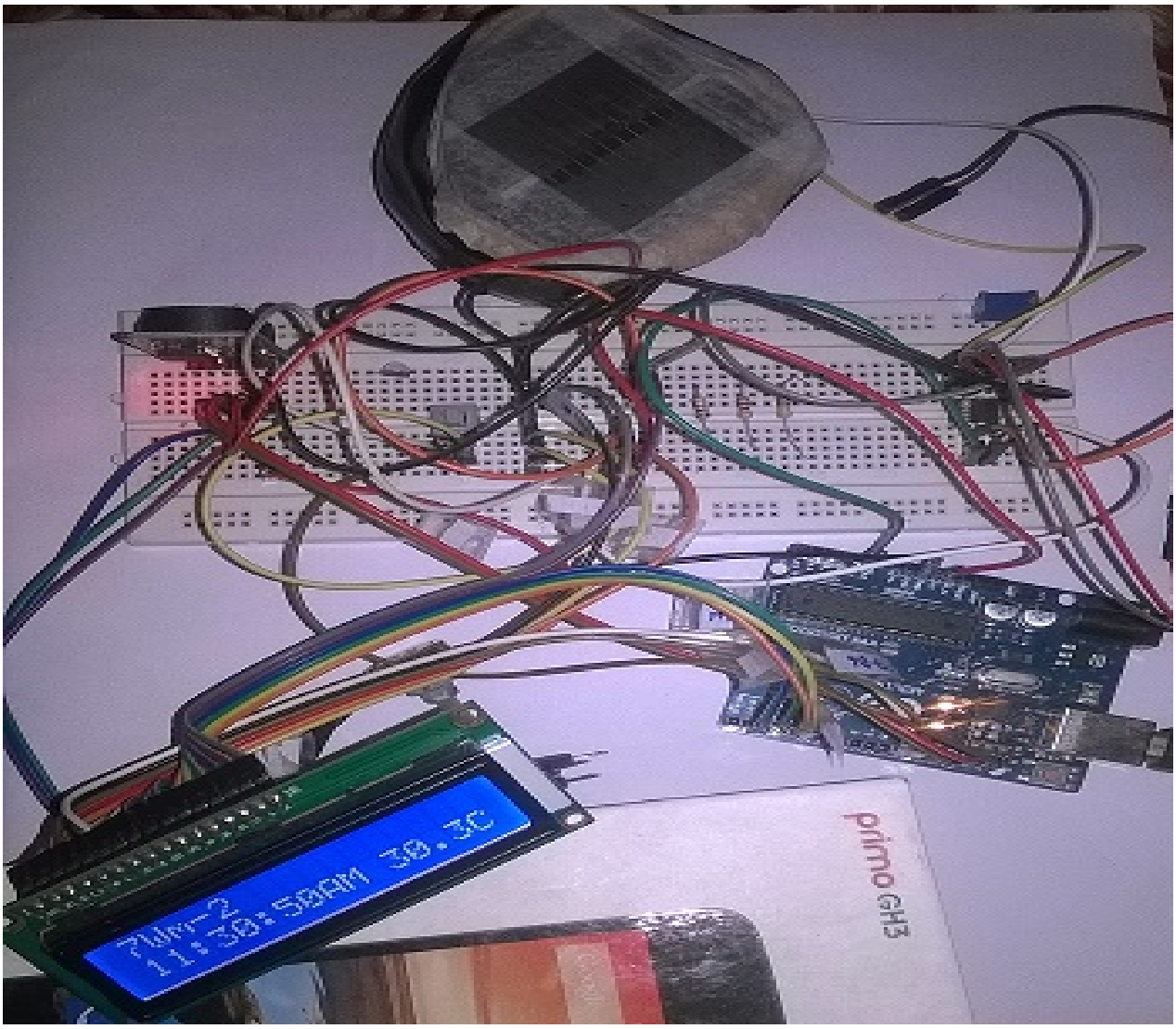
3.1.D Schematic diagram of circuit

The circuit connected into one switch one sensor and one op Amp and one Arduino and one LCD. The sensor converts the solar current into the voltage and the op-amp amplifies it and the Arduino takes the value and analyze it and shows output in the LCD.

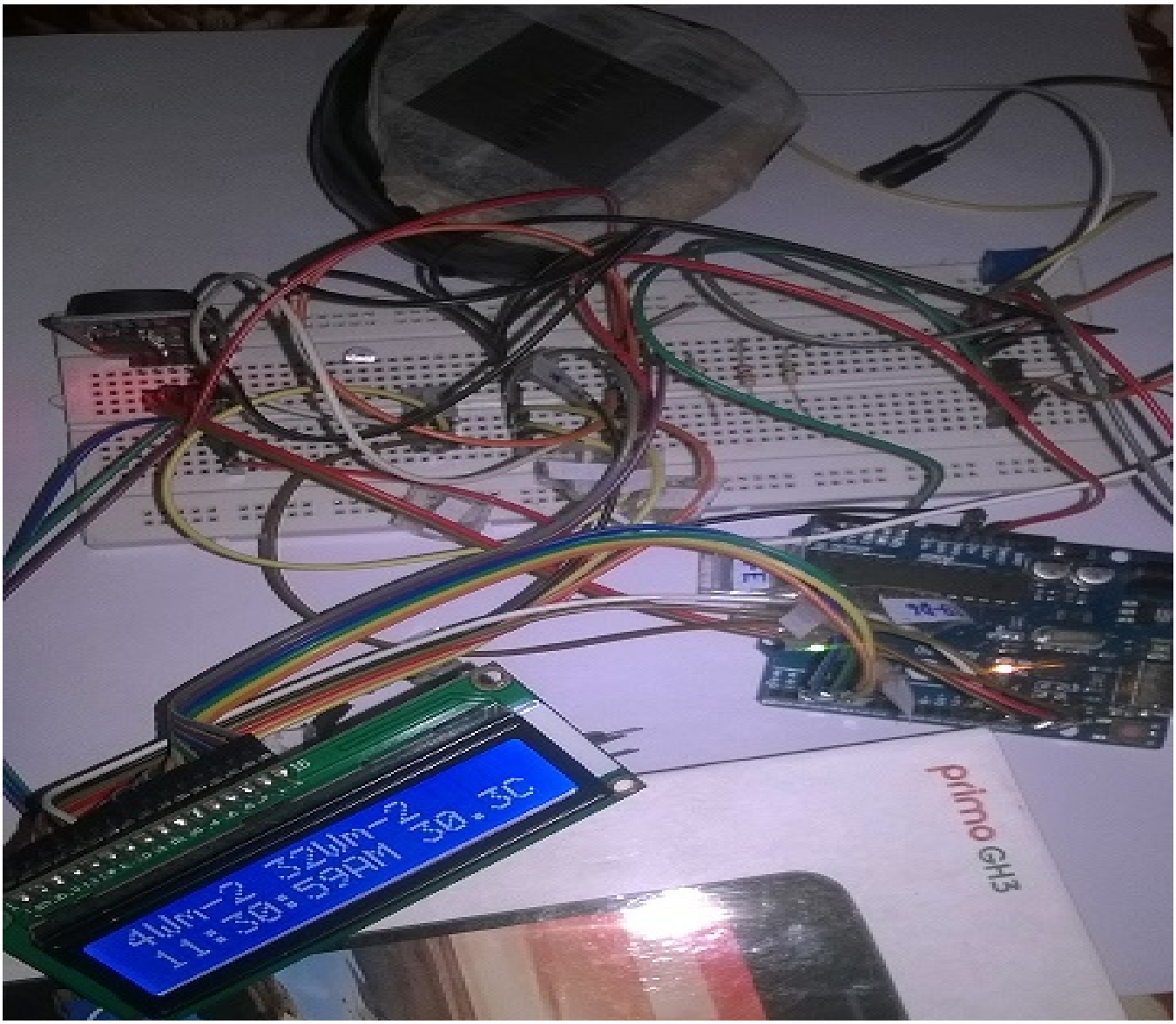


3.1.E System Circuit

Instantaneous Mode



Integrated Mode:



3.1.F System Connection

Connection between LCD,Arduino and RTC

LCD PIN	ARDUINO PIN	RTC PIN
1	GND	
2	5v	

3	wiper	
4	12	
5	GND	
6	11	
11	5	
12	4	
13	3	
14	2	
15	5V	VCC
16	GND	GND
	SDA	SDA
	SCL	SCL

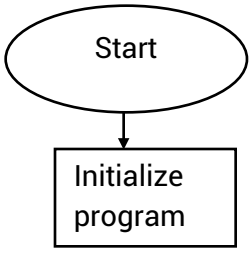
3.2 Software Design

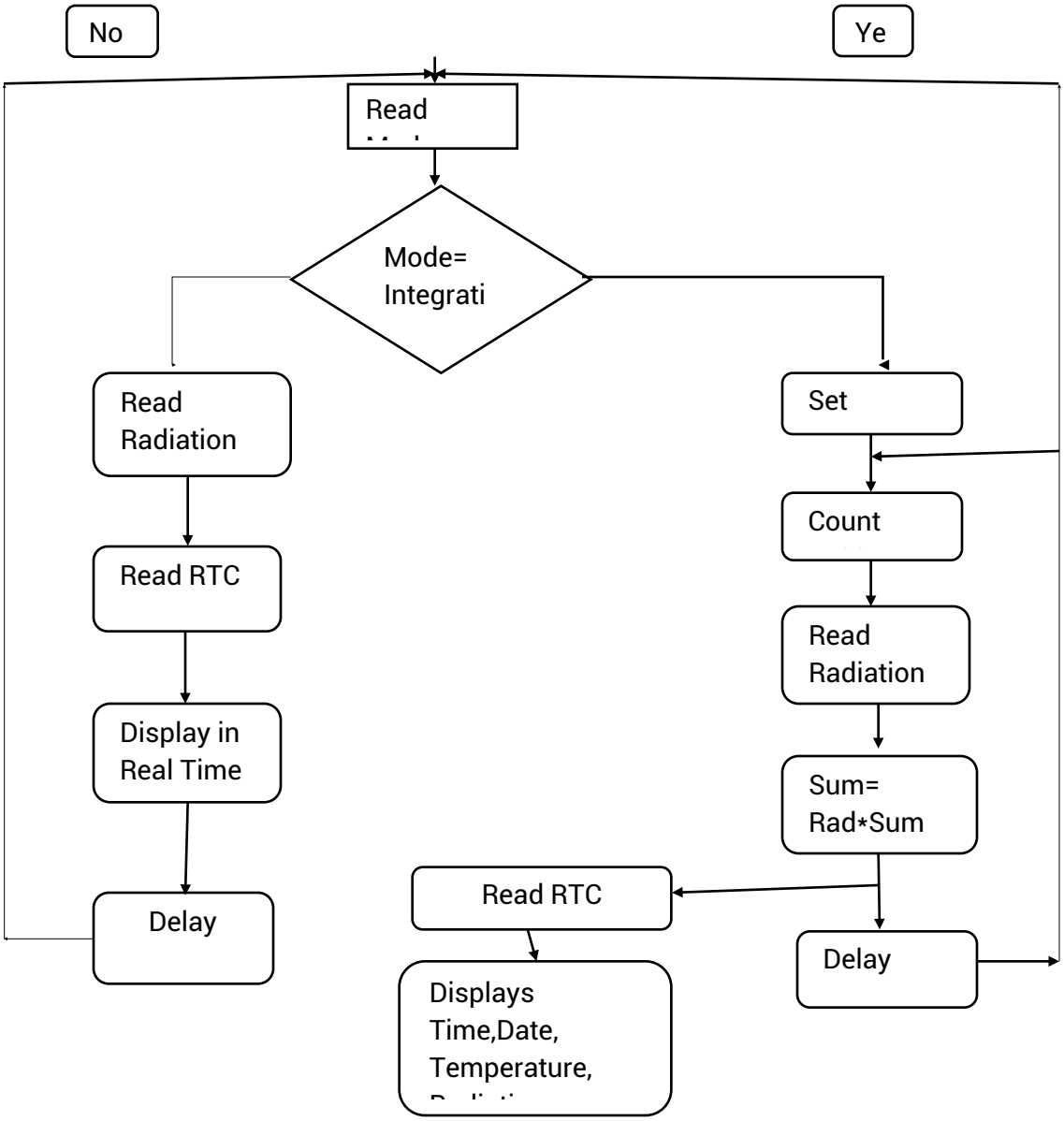
3.2.A Programming language

Without programming we cannot interface with the Arduino

and cannot get any value. Here we had used the Arduino C programming. Before doing the programming the programming code, logical concept has applied very carefully.

3.2.B Flow chart of the program





3.2.C Program code

Library Function:

```
#include <Wire.h>

#include "ds3231.h"

#include "rtc_ds3231.h"

#include<LiquidCrystal.h>
```

Arduino Digital I/O pins connected with LCD:

```
LiquidCrystal lcd(12,11,5,4,3,2);
```

Maximum size:

```
#define BUFF_MAX 128

uint8_t time[8];

char recv[BUFF_MAX];
```

Long Value:

```
unsigned int recv_size = 0;

unsigned long prev;
```

Time Interval or pulse:

```
 interval = 1000;
```

Input :

```
int analogInPin=0;
```

Setup:

```
void setup()

{

   Serial.begin(9600);

   Wire.begin();

   DS3231_init(DS3231_INTCN);

   memset(recv, 0, BUFF_MAX);
```

```
  Serial.println("GET time");
  lcd.begin(16,2);
  lcd.clear();
  serial.println("Setting Time");
  parse_cmd("T105910415072015",16);
```

//**Time setup=** Seconds (10), Minutes(59), Hours(10), Day of the week(4), Day of the month(15), Month of the year(07), Year(2015).

```
}
void loop()
{
```

To Display on LCD monitor:

```
lcd.print(Function);
lcd.print("Comment");
```

To Display on Serial monitor:

```
Serial.print(Function);
Serial.print("Comment");
```

Print New line:

```
Serial.println(Function);
Serial.println("Comment");
```

Space Between Two values:

```
lcd.print(" ");
  }
```

Using Switch Case:

```
void printMonth ( int month)
```

```
{
  switch(month)
  {
    case 1: lcd.print(" January "); break;
    case 2: lcd.print(" February "); break;
    case 3: lcd.print(" March "); break;
    case 4: lcd.print(" April "); break;
    case 5: lcd.print(" May "); break;
    case 6: lcd.print(" June "); break;
    case 7: lcd.print(" July "); break;
    case 8: lcd.print(" August "); break;
    case 9: lcd.print(" September"); break;
    case 10: lcd.print(" October "); break;
    case 11: lcd.print(" November "); break;
    case 12: lcd.print(" December "); break;
  }
}
```

Real Time Temperature/Radiation Measuring Code:

```
/*
```

```
Simple Temperature uses the lm35 in the basic centigrade temperature configuration
*/
float temp;
int tempPin = 2; // analog input pin A2
int sampleTime = 1000; // 1 second dafault

void setup()
{
  Serial.begin(9600);// serial port communicate with temp sensor
}

void loop()
{
  //gets and prints the raw data from the lm35
  temp = analogRead(tempPin);
  Serial.print("Main DATA: ");
  Serial.print (temp);
  temp = temp * 0.48828125/20;
  Serial.println(" ");
  Serial.println((byte)temp);
  //converts raw data into degrees celsius and prints it out
  // 500mV/1024=.48828125

  Serial.print("CELSIUS: ");
  Serial.print(temp);
  Serial.println("*C ");
  //converts celsius into fahrenheit
  temp = temp *9 / 5;
  temp = temp + 32;
  Serial.print("FAHRENHEIT: ");
  Serial.print(temp);
  Serial.println("*F");
  delay(sampleTime);

}
```

Chapter Four

RESULTS

4.1.A Data Analysis

Instantaneous Mode:

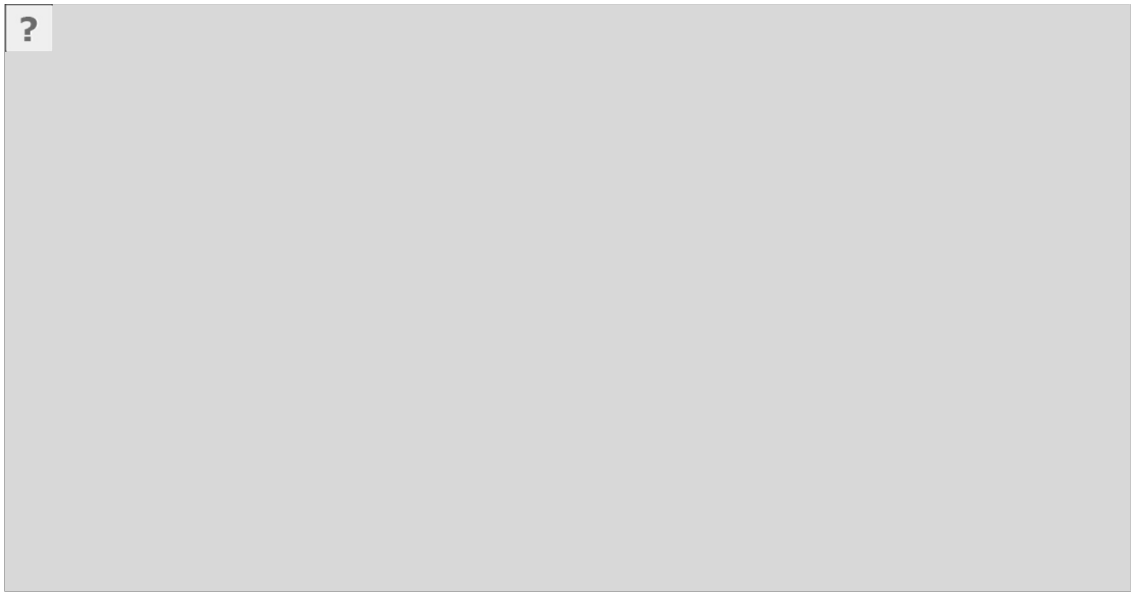
Day time	Day 1 (Wm^{-2})	Day 2(Wm^{-2})
6:00AM	33	35
7:00AM	95	56
8:00AM	190	98
9:00AM	210	201
10:00AM	398	350
11:00AM	450	412
12:00PM	499	401
1:00PM	512	550
2:00PM	490	570
3:00PM	320	389
4:00PM	254	150
5:00PM	115	120
6:00PM	45	48

Integrated Mode:

Day time	Day 1 (Wm^{-2})	Day 2(Wm^{-2})
8:00AM	318	189
9:00AM	528	390
10:00AM	926	740
11:00AM	1376	1152
12:00PM	1875	1553
1:00PM	2387	2103
2:00PM	2877	2673

4.2.B Graphical representation

Instantaneous Mode:



Integrated Mode:

4.3 Observation

Atmospheric conditions can increase or decrease the intensity of the sunlight.
Shadows from the obstacle can decrease the efficiency of the system.
At the noon the value is highest where in the morning and evening the value is lowest.

4.4 Future Work:

☐Data transfer system can be improved by adding wifi

file transfer system, Bluetooth device or any handheld mobile devices.

- ☐Another high accuracy microcontroller can be used instead of Arduino Uno.
- ☐Portable power system can be integrated for better handling of this system.
- ☐Remote control system can be added.

4.5 Conclusion

In this project work we have invented a better method to measure the solar radiation in an easy way. We tried our best to measure the solar radiation in the more sophisticated and accurate way. The device loss degrades the system efficiency in a small amount. Atmospheric conditions and shadows can reduce the efficiency of the system. Here we used a real time clock to measure the accurate time of observation. Note that the accuracy of this real time clock is very high. We

combined the clock with the system to measure the radiation in real time and integrated time. The sensor original value is very low. We had used an amplifier to amplify the output voltage and integrate it with our microcontroller based system. We used an Arduino based programming language to implement the whole system and shows it to the liquid crystal display. Now we can measure the radiation of the sun in avery cost effective way. Note that The device loss degrades the system efficiency in a small amount. Atmospheric conditions and shadows can reduce the efficiency of the system.

4.6 References:

1. https://www.arduino.cc/en/Main/arduinoBoardUno
2. http://www.techopedia.com/definition/3641/microcontroller
3. http://www.delta-t.co.uk/product-display.asp?id=SPN1%20Product&div=Meteorology%20and%20Solar
4. https://en.wikipedia.org/wiki/Solar_cell
5. www.microchip.com

6. Let us C by Yashavant kanetkar
7. Teach yourself C++ by Sheild.
8. http://www.maximintegrated.com/en/products/digital/real-time-clocks/DS3231.html
9. http://www.instructables.com/id/DS3231-OLED-clock-with-2-button-menu-setting-and-t/
10. http://www.ti.com/product/lm35
11. http://www.mikroe.com/chapters/view/64/chapter-1-introduction-to-microcontrollers/
12. http://stackoverflow.com/questions/1663993/what-is-the-realtime-setting-for-for-process-priority
13. https://processing.org/examples/clock.html
14. http://www.st.com/web/en/catalog/sense_power/FM151/CL1410?sc=rtc
15. http://www.facstaff.bucknell.edu/mastascu/elessonsHTML/Sensors/TempLM35.html

Appendixes

Programing Code For Whole System:

```
#include <Wire.h>
#include "ds3231.h"
#include "rtc_ds3231.h"
#include<LiquidCrystal.h>
//#include<EEPROM.h>
```

```
LiquidCrystal lcd(12,11,5,4,3,2);

#define BUFF_MAX 128

uint8_t time[8];
char recv[BUFF_MAX];
unsigned int recv_size = 0;
unsigned long prev, interval = 1000;

int mode=0;

int n=0;//Start reading from the first byte(address=0) of the eeprom

unsigned long  delayValue;
//float rad;
//int analogInPin=0;

int rad;
//int radPin = 0; // analog input pin 0
unsigned long sampleTime = 1000; // 1 second dafault
//int lcdPin=13; // lcd will connected to pin 13

void setup()
{
   Serial.begin(9600);
   Wire.begin();
   DS3231_init(DS3231_INTCN);
   memset(recv, 0, BUFF_MAX);
   Serial.println("GET time");



   lcd.begin(16,2);

   lcd.clear();

   //serial.println("Setting Time");
   //parse_cmd("T105910415072015",16);
}

void loop()
{
   char in;
   char C;
   char tempF[3];
   float temperature;
   char buff[BUFF_MAX];
   unsigned long now = millis();
   struct ts t;
```

```
// show time once in a while
if ((now - prev > interval) && (Serial.available() <= 0)) {
    DS3231_get(&t);
    parse_cmd("C",1);
    temperature = DS3231_get_treg();
    dtostrf(temperature, 5, 1, tempF);
    lcd.clear();

    lcd.setCursor(0,1);
    if(t.hour==13)
    {
     lcd.print("1");
    }
    else if(t.hour==14)
    {
      lcd.print("2");
    }
    else if(t.hour==15)
    {
      lcd.print("3");
    }
    else if(t.hour==16)
    {
      lcd.print("4");
    }
    else if(t.hour==17)
    {
      lcd.print("5");
    }
    else if(t.hour==18)
    {
      lcd.print("6");
    }
    else if(t.hour==19)
    {
      lcd.print("7");
    }
    else if(t.hour==20)
    {
      lcd.print("8");
    }
    else if(t.hour==21)
    {
      lcd.print("9");
    }
    else if(t.hour==22)
    {
      lcd.print("10");
```

```
      }
      else if(t.hour==23)
      {
        lcd.print("11");
      }
       else if(t.hour==0)
      {
        lcd.print("12");
      }
      else lcd.print(t.hour);


      lcd.print(":");
      if(t.min<10)
      {
        lcd.print("0");
      }
      lcd.print(t.min);
      lcd.print(":");
      if(t.sec<10)
      {
        lcd.print("0");
      }
      lcd.print(t.sec);
      if (t.hour<12)
      {
        lcd.print("AM");
      }
      else
      {
        lcd.print("PM");
      }

      lcd.print(tempF);

       lcd.print("C");

/*
      lcd.setCursor(0,C);

      lcd.print(t.mday);
      printMonth(t.mon);

      lcd.print(t.year);*/


      // there is a compile time option in the library to include unixtime support
#ifdef CONFIG_UNIXTIME
      snprintf(buff, BUFF_MAX, "%d.%02d.%02d %02d:%02d:%02d %ld", t.year,
```

```
          t.mon, t.mday, t.hour, t.min, t.sec, t.unixtime);
#else
      snprintf(buff, BUFF_MAX, "%d.%02d.%02d %02d:%02d:%02d", t.year,
          t.mon, t.mday, t.hour, t.min, t.sec);
#endif

      Serial.println(buff);
      prev = now;
   }

   if (Serial.available() > 0) {
      in = Serial.read();

      if ((in == 10 || in == 13) && (recv_size > 0)) {
         parse_cmd(recv, recv_size);
         recv_size = 0;
         recv[0] = 0;
      } else if (in < 48 || in > 122) {;       // ignore ~[0-9A-Za-z]
      } else if (recv_size > BUFF_MAX - 2) {   // drop lines that are too long
         // drop
         recv_size = 0;
         recv[0] = 0;
      } else if (recv_size < BUFF_MAX - 2) {
         recv[recv_size] = in;
         recv[recv_size + 1] = 0;
         recv_size += 1;
      }

   }

//  char C;
  int sampleTime = 1000; // 1 second dafault
 rad = analogRead(A0);
  rad= rad - 60;
  mode=digitalRead(13);
if(mode==HIGH){

  char C;

  Serial.print("Main DATA: ");
  Serial.print (rad);
   Serial.println("Wm-2");
   lcd.setCursor(0,C);
  lcd.print(rad);
  //temp = temp * 0.48828125/20;
  Serial.println(" ");

  lcd.print("Wm-2 ");
  //Serial.println((byte)temp);
```

```
   /* lcd.print(temp);
    lcd.print("C");
      temp = temp *9 / 5;
  temp = temp + 32;
  Serial.print("FAHRENHEIT: ");
  Serial.print(temp);
  Serial.println("F");
lcd.print(" ");
lcd.print(temp);
lcd.print("F");
   // tempi=temp;*/

  delay(sampleTime);
}
else if(mode==LOW){
  int i, x,  cn=0;
  unsigned long sum=0;
//isFirst=0;
for(int count=1;count<1000;count++)
{
sum= sum + rad;
cn++;

Serial.print("Current data: ");

Serial.print(rad);
 Serial.println("Wm-2");
Serial.print("Time ( no. Data): ");
Serial.println(count);
 lcd.setCursor(0,C);
 lcd.print(rad);
 lcd.print("Wm-2");
 Serial.print("Integrated data: ");
 Serial.print(sum);
 Serial.println("Wm-2");
 Serial.println("   ");
 //lcd.print("C");
 lcd.print(" ");
 //lcd.print(count);

// lcd.setCursor(0,1);
//lcd.print(" ");
lcd.print(sum);
lcd.print("Wm-2");
//lcd.print("C");

delay(sampleTime);
```

```
}
}
}

void parse_cmd(char *cmd, int cmdsize)
{
   uint8_t i;
   uint8_t reg_val;
   char buff[BUFF_MAX];
   struct ts t;


   // TssmmhhWDDMMYYYY aka set time
   if (cmd[0] == 84 && cmdsize == 16) {
      //T355720619112011
      t.sec = inp2toi(cmd, 1);
      t.min = inp2toi(cmd, 3);
      t.hour = inp2toi(cmd, 5);
      t.wday = cmd[7] - 48;
      t.mday = inp2toi(cmd, 8);
      t.mon = inp2toi(cmd, 10);
      t.year = inp2toi(cmd, 12) * 100 + inp2toi(cmd, 14);
      DS3231_set(t);
      Serial.println("OK");
   } else if (cmd[0] == 49 && cmdsize == 1) {  // "1" get alarm 1
      DS3231_get_a1(&buff[0], 59);
      Serial.println(buff);
   } else if (cmd[0] == 50 && cmdsize == 1) {  // "2" get alarm 1
      DS3231_get_a2(&buff[0], 59);
      Serial.println(buff);
   } else if (cmd[0] == 51 && cmdsize == 1) {  // "3" get aging register
      Serial.print("aging reg is ");
      Serial.println(DS3231_get_aging(), DEC);
   } else if (cmd[0] == 65 && cmdsize == 9) {  // "A" set alarm 1
      DS3231_set_creg(DS3231_INTCN | DS3231_A1IE);
      //ASSMMHHDD
      for (i = 0; i < 4; i++) {
         time[i] = (cmd[2 * i + 1] - 48) * 10 + cmd[2 * i + 2] - 48; // ss, mm, hh, dd
      }
      uint8_t flags[5] = { 0, 0, 0, 0, 0 };
      DS3231_set_a1(time[0], time[1], time[2], time[3], flags);
      DS3231_get_a1(&buff[0], 59);
      Serial.println(buff);
   } else if (cmd[0] == 66 && cmdsize == 7) {  // "B" Set Alarm 2
      DS3231_set_creg(DS3231_INTCN | DS3231_A2IE);
      //BMMHHDD
      for (i = 0; i < 4; i++) {
         time[i] = (cmd[2 * i + 1] - 48) * 10 + cmd[2 * i + 2] - 48; // mm, hh, dd
      }
```

```
      uint8_t flags[5] = { 0, 0, 0, 0 };
      DS3231_set_a2(time[0], time[1], time[2], flags);
      DS3231_get_a2(&buff[0], 59);
      Serial.println(buff);
    } else if (cmd[0] == 67 && cmdsize == 1) {  // "C" - get temperature register
      Serial.print("temperature reg is ");
      Serial.println(DS3231_get_treg(), DEC);
    } else if (cmd[0] == 68 && cmdsize == 1) {  // "D" - reset status register alarm flags
      reg_val = DS3231_get_sreg();
      reg_val &= B11111100;
      DS3231_set_sreg(reg_val);
    } else if (cmd[0] == 70 && cmdsize == 1) {  // "F" - custom fct
      reg_val = DS3231_get_addr(0x5);
      Serial.print("orig ");
      Serial.print(reg_val,DEC);
      Serial.print("month is ");
      Serial.println(bcdtodec(reg_val & 0x1F),DEC);
    } else if (cmd[0] == 71 && cmdsize == 1) {  // "G" - set aging status register
      DS3231_set_aging(0);
    } else if (cmd[0] == 83 && cmdsize == 1) {  // "S" - get status register
      Serial.print("status reg is ");
      Serial.println(DS3231_get_sreg(), DEC);
    } else {
      Serial.print("unknown command prefix ");
      Serial.println(cmd[0]);
      Serial.println(cmd[0], DEC);
    }
}
void printMonth ( int month)
{
  switch(month)
  {
    case 1: lcd.print(" January "); break;
    case 2: lcd.print(" February "); break;
    case 3: lcd.print(" March "); break;
    case 4: lcd.print(" April "); break;
    case 5: lcd.print(" May "); break;
    case 6: lcd.print(" June "); break;
    case 7: lcd.print(" July "); break;
    case 8: lcd.print(" August "); break;
    case 9: lcd.print(" September"); break;
    case 10: lcd.print(" October "); break;
    case 11: lcd.print(" November "); break;
    case 12: lcd.print(" December "); break;
  }
}
```