IoT Based Smart Grid System

SONARGAON UNIVERSITY (SU)

Supervised By

Md. Rais Uddin Mollah

Lecturer and Asst. Coordinator
Department of EEE
Sonargaon University (SU)

Submitted by

Md. Abdul Kader	**ID: EEE1802014105**
Md. Faysal Mahmud	**ID: EEE1802014108**
Foysal Miah	**ID: EEE1802014010**
Razaul Hoque	**ID: EEE1802014124**
Al Amin Kazi	**ID: EEE1802014077**
Md. Kawser Hamid	**ID: EEE1903018043**

Department of Electrical & Electronic Engineering (EEE)
Sonargaon University (SU)
147/I, Panthopath,
Dhaka-1215, Bangladesh.

Date of submission: 23/01/2022

Declaration

We declare that this project work entitled **"IoT Based Smart Grid System"** is the result of our own work as cited in the references. This project has not been accepted for any degree and is not concurrently submitted in candidature for any other degree or diploma elsewhere.

_____	_____
Md. Abdul Kader	Razaul Hoque
_____	_____
Md. Faysal Mahmud	Alamin Kazi
_____	_____
Foysal Miah	Md. Kawser Hamid

Under Supervision of

Md. Rais Uddin Mollah

Lecturer and Asst. Coordinator

Department of EEE

Sonargaon University

Certification

This is to certify that this project entitled "**IoT Based Smart Grid System**" is done by the following students under my direct supervision. This project work has been carried out by them in the laboratories of the Department of Electrical and Electronic Engineering under the Faculty of Engineering, Sonargaon University (SU) in partial fulfillment of the requirements for the degree of Bachelor of Science in Electrical and Electronic Engineering.

Supervisor

……

Md. Rais Uddin Mollah

Lecturer and Asst. Coordinator

Department of Electrical and Electronic Engineering (EEE)

Sonargaon University (SU)

ACKNOWLEDGEMENT

The report titled as on "**IoT Based Smart Grid System**" has been prepared to fulfill the requirement of our practicum program. In the process of doing and preparing our practicum report, we would like to pay our gratitude to some persons for their enormous help and vast co-operation.

At first, we would like to show our gratitude to the University authority to permit us to do our practicum. Specially, we would like to thank to our honorable teacher **Md. Rais Uddin Mollah**, Lecturer and Asst. Coordinator, Department of Electrical & Electronics Engineering, SU–Sonargaon University, Dhaka, for his valuable and patient advice, sympathetic assistance, co-operation, contribution of new idea. Deep theoretical and hardware knowledge & keen interest of our supervisor in this field influenced us to carry out this project. His endless patience, scholarly guidance, continual encouragement, constant and energetic supervision, constructive criticism, valuable advice, reading many inferior draft and correcting them at all stage have made it possible to complete this project.

Finally, we would like to thanks again to the respected **Vice- Chancellor of SU, Professor Dr. Md. Abul Bashar also thanks to Head of Department of SU, Electrical & Electronics Engineering, Professor Dr. M. Bashir Uddin** because they are designated such an environment for learning through which we got the opportunity to acquire knowledge under Bsc in EEE program, and that will be very helpful for our prospective career. We are, indeed, grateful to all those from whom we got sincere cooperation and help for the preparation of this report.

Abstract

An electric network, electric grid, or electricity network is an integrated electricity supply network for producers to consumers. It consists of electricity producing stations. The main objective of this study is to monitor the electricity grid system process, disclose this system at a dangerous level, monitor the current line, and reduce conventional systems expenses. From anywhere on the Internet, we can monitor. We can do it also if a system is enabled or disabled. It uses an electrical microcontroller to monitor an electrical device using microcontroller to read sensor voltage and current and then communicate measured data via a new Android application for wireless monitoring. It enables the monitoring of several basic power quality parameters of basic watt and ampere. Here we use a Node MCU for controlling this system, use current and voltage sensor to measure and monitor user watt and ampere level and the main power will be come from solar panel.

TABLE OF CONTENTS

TOPIC		PAGE
Declaration		ii
Certificate		iii
Acknowledgement		iv
Abstract		v
Table of Content		vi-vii
List of Figures		viii-ix
List of Tables		x
CHAPTER-1	**INTRODUCTION**	**1-7**
1.1 Background		1-2
1.2 Literature Review		3-4
1.3 Smart Grid		4-5
1.4 Application of IoT		5
1.5 Objects		5
1.6 Methodology		6
1.7 Advantages		6
1.8 Application		6
1.9 Limitations		7
1.10 Future Scope of Work		7
1.11 Structurer of the Project		7
CHAPTER-2	**HARDWARE ANALYSIS**	**8-27**
2.1 Introduction		8
2.2 Node MCU		8-11
2.3 Solar Panel		11-12

2.4 Solar Charge Controller 12-14
2.5 12V Battery 14-15
2.6 AC Current Sensor 15-16
2.7 Voltage Sensor 16-17
2.8 Relay 18-21
2.9 100W Inverter Circuit 22-23
2.10 Capacitor 23-25
2.11 Resistor 26-27

CHAPTER-3 SOFTWARE IMPLEMENTATION 28-41
3.1 Arduino Software 28-38
3.2 Proteus Software 39
3.3 Adafruit Server 40-41

CHAPTER-4 SYSTEM ARCHITECTURE 42-47
4.1 Block Diagram 42
4.2 Circuit Diagram 43
4.3 Working Principle 43-44
4.4 The project prototype 44
4.5 Result 44-47
4.6 Cost Analysis 47

CHAPTER-5 DISCUSSION & CONCLUSION 48
5.1 Discussion 48
5.2 Conclusion 48

Reference **49-50**

Appendix **51-56**

LIST OF FIGURES

FIGURE NO	FIGURE NAME	PAGE NO
1.1	Smart Grid System	5
2.1	Node MCU	9
2.2	Node MCU Schematic Diagram	9
2.3	Node MCU Pin Out	10
2.4	Solar Panel	12
2.5	Solar Panel Schema Diagram	12
2.6	Solar Sell System Curve	13
2.7	Solar Charger Controller Circuit	13
2.8	12V Battery	15
2.9	Voltage Sensor	16
2.10	Voltage Sensor	17
2.11	Relay	18
2.12	Transistor Switching Circuit	19
2.13	Relay Module	20
2.14	Pin Diagram of Relay Module	20
2.15	Main Voltage Connection	21
2.16	Inverter Circuit	22
2.17	Capacitor Pin Out	23
2.18	Capacitor Operation	25
3.1	Arduino Software Interface IDE	29
3.2	Proteus Software Interface	39
4.1	Block Diagram	42
4.2	Circuit Diagram	43
4.3	Project Prototype	44

4.4	One Load on this Project	45
4.5	Two Load's on this project	46
4.6	Two load's Ampere reading in Apps	46
4.7	Two Load's Watt Reading in Apps	47

List of Tables

Figure no	Figure name	Page no
01	Cost of Components with Price	47

CHAPTER 1

INTRODUCTION

1.1 Background

Computerization shows a main part in the industrial processes and IT application. Energies manufacturers provide all houses with energy through intermediary power-controlled hubs known as the Electricity Grid. Sometimes the failure of the electrical grid causes difficulties that lead to blackout of a whole region supplied by this specific grid. The project seeks to solve the problem through the use of IoT for communication and to address a number of additional challenges that an intelligent system can tackle to minimize excessive energy losses IoT's intelligent energy grid is built the at mega series controller that manages the different system operations [1]. IoT gradually becomes an essential element of our lives that can be felt within us. With Wi-Fi technology the system communicates via the internet. The most important component facilitating this endeavor is the reconnecting of the operational grid transmission system. When an energy grid fails and a different energy grid occurs, the system shifts transmission lines to the grid, making it easier for a particular location whose energy grid is OFF to supply electricity [2].

And this information, from which the grid is actively updated via IoT smartphone applications, may be used by the authorities and updates. In addition, atomic and nanostructures take use of speeds that were previously unimaginable for storing, detecting, and computing. For this reason, many researchers are currently trying to develop the new types of materials for connecting them it. Comprehensive research investigations in scholarly arts and print and online media reports illustrating the inherent efficacy and application of IoT changes have been carried out and accessible. In the shape of print resources [3]. In addition to grid monitoring, this project also includes the advanced energy consumption tracking capabilities and even electricity stealing. The major goal of this work is 'IoT' power grid control for the basic objective is to develop an intelligent system to take advantage of a project [4].

The major goal is to shut down the Remote-Control System, to monitor the power line current, to minimize the cost of traditional systems at unsafe levels. We did frequency and voltage monitoring in this project. Besides measuring the data current. From all across the internet we can check. This could also be done by enabling or disabling the system [5]. Fresh information transmission by internet of things (IoT) and storage revolution. Goals that are identifiable and intelligent by making judgments in connection with events or allowing them to be contextualized. You can pass information on yourself. They can have access or be components of other services to information utilized by others. Internet of Things (IoT) is an interconnected system for the transmission of data across a network and the transfer of information through a network, without human or human interaction [6].

It includes interconnected computer devices, mechanical or alphanumeric apparatuses, matters, natures, and persons with unique identifiers. Due to the confluence of many technologies, real-time analysis, machine education, commodity and embedded systems, the Internet of Things concept has changed. In future development of electricity grid sensors, actuators and transducers, real-time energy tracking and monitored services are anticipated to play a significant role. IoT has become a technology that offers new answers to the power grid system problems [7]. The IoT enabled sensors are utilized to communicate information via the internet and mobile applications throughout the grid system, which enables enhanced grid management. We suggested an IoT aided power monitoring and control system because of the above-mentioned advancement in IoT and the use of it in power networks. It gives customers and utilities the benefit of analyzing and managing their resources. This article presents IoT-based power monitoring with blink software. A literature analysis highlights existing SG, IoT and IoT research. SG supports. Our additions to this article include: the integration of an open-source IoT platform that delivers information analyzes is supplied for the deployment of the IoT aided power monitoring system [8].

1.2 Literature Review

IoT can be considered as a worldwide network infrastructure consisting of various connected devices that depend on different processing technologies such as sensory, communication, networking and information. In order for IoT to provide high quality services to end users, technical standards is needed to define different specifications for information exchange, processing, and communications between things. The future Success highly depends on standardization, which provides various characteristics like interoperability, compatibility, effective operation, and reliability on world scale. Many countries in the world are looking forward to the development of IoT standards because it can bring huge economic benefits in future.

"Landi, C, Dipt. di Ing. dell"Inf, Seconda University di Napoli, Aversa, Italy; Merola, P, Ianniello, G", titled "ARM-based management system using smart meter and Web server",2011. In this paper it is describes the low-cost real-time ARM-based energy management system. It is devised as a part of a distributed system which measures the main power system quantities and gives the possibility to control the whole power plant. An integrated Web Server allows the system to collect the statistics of power consumptions, power quality and is being able to interface the devices for load displacement. This device is distinguished with an easy access to information and the combination of a smart meter and data communication capability which allows local and remote access. In this way it is possible to manage the power consumption of the power system leads to an overall reduction in power consumption and billing costs.

"AMR perspective to save energy in Smart Grids ", 2012. In this paper, an AMR solution provides the enhanced end-to-end application. It is completely based on an energy meter with low-power microcontroller MSP430FE423A and the Power Line Communication standards. The microcontroller comprises an energy metering module ESP430CEl. The conclusion of this paper is to realize a real time pricing. This solution leads to great interest in low cost and low carbon society point of view,The internet of things (IOT) has touched many different actors and received more appreciation one year since the last edition of the cluster book 2012. Smart cities (and regions), smart cars and mobility, intelligent homes and helped lives, smart industries, public safety, energy protection,

agriculture and tourism have received great interest from prospective internet in the field of application of belongings in the situation to a future IoT-Ecosystem. In accordance with this trend, the Internet of Things currently looks like an area of innovation and growth for most governments in Europe, Asia and America. While greater companies are still not able to see the potential in some application areas, many of them pay high attention or even speed up the process by formulating new words for IoT and adding new components. In addition, individual and corporate end-users today have considerable skill in the area of smart gadgets and networked apps [9].

The ongoing growth on the Internet of Things estimates greater possibilities by combining related technology and ideas such as cloud applications, the future Network, large-scale information technology, robots, and conceptual technology [10]. Naturally the notion is not new as such, but now becomes apparent when these connected concepts are first uncovered by merging synergies. The Internet of Things remains mature, however, in particular because of many limitations limiting the full use of the IOT. The following appears to be most important among these factors [11]. - Not clear methodology for the use of unique IDs and numbering spaces on a global scale for various types of permanent and volatile items. - No accelerated application and future IOT architecture development.

1.3 Smart Grid System

The smart grid is decentralized system where power flows in both directions, from generation end to consumer end and vice versa. Smart grids are based on communication between provider and consumer. It is energy consumption monitoring and measuring system. With the help of smart grid consumer and owner get daily electricity consumption reading and owner can cut electricity through internet if bill is not paid Smart grid can be complement of traditional electric grid system by including renewable energy resources, such as wind, solar and biomass, which is environmentally cleaner as compared to fossils fuels.

Figure 1.1: Smart Grid System

1.4 Application of IoT

Applications related to IoT are still in early stages, but the use of it is rapidly evolving. Only a few applications are considered for being developed and deployed in different industries such as environmental monitoring, food supply chain, security and surveillance, etc. Some of IoT applications related to industries are: Using IoT in healthcare service industries. Using IoT in food supply chain. Using IoT for safer mining production. Using IoT for transportation and logistics. Using IoT in firefighting. Using IoT in smart environment monitoring. Using IoT in smart agriculture.

1.5 Objective

The objective of this work is:

- Design & Construction of "**IoT Based Smart Grid System**"
- Learn the Implementation of Smart grid system monitoring line ampere and watt of two house.
- Implementation of all data will record in a data logger system.

1.6 Methodology

Our used methodology for the project:

- Creating an idea for design and construction of "**Iot Based Smart Grid System**". And designing a block diagram & circuit diagram to know which components we need to construct it.
- Collecting all the components and programming the microcontroller to control the system.
- Setting up all the components in a PCB board & then soldering. Then assembling all the blocks in a board and finally running the system & checking.

1.7 Advantages

There are certainly many advantages of our project and some of the major ones have been given below:

- By using this system, it is possible to reduce human physical work.
- It's a wireless notification system which is suitable in modern technology.
- This Smart Grid is an automatic protection system from unwanted situations.
- It's easy to install this system.
- Very cost effective.

1.8 Applications

Our project has many application areas and actually we need to use it in many places to verified the soil situation, air moisture and it will be works from far way with a device. Some of the application areas of the project has been pointed out below:

- It can be used in anywhere etc.
- It can be use in Power Grid.
- It Can be Use in Home. & Industry.

1.9 Limitations

We are thinking about adding many advantages to our project but this system has some limitations. Some of the limitations are given below:

- IoT Signal may late cause it’s a demo project.

1.10 Future Scope of Work

The model can be improved by making some changes in the program and components. Some suggestions are given below.

- In the future we will add more sensor which will be detect unwanted situation and protect our system.

1.11 Structure of the Project

This project book consists of five chapter. The first chapter contains the statement of the introduction, our background study for the project, objectives of the study in the project and the project organization. Chapter two contains design of this project, block diagram and circuit diagram, working principle. Chapter three describes the background and real project, details of component and instrument details of the whole project. Chapter four deals with the result and discussion and shows the complete prototype of the project that we have built. In the final chapter, we discuss about future scope and conclusion of our project.

CHAPTER 2

HARDWARE ANALYSIS

2.1 Introduction

This Project has worked on two things, Hardware and Software. In this Chapter we will discuss about instrument specification, application and working procedure.

Software

- Proteus 8.9
- Arduino IDE
- Adafruit

Hardware

- Solar Panel
- Solar Charger Control
- Battery
- Inverter
- Relay
- Current Sensor
- Voltage Sensor

Hardware Description

2.2 Node MCU

Node MCU is an open-source firmware for which open-source prototyping board designs are available. The name "Node MCU" combines "node" and "MCU" (micro-controller unit). The term "Node MCU" strictly speaking refers to the firmware rather than the associated development kits. Both the firmware and prototyping board designs are open source. The firmware uses the Lua scripting language.

Figure 2.1: Node MCU

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

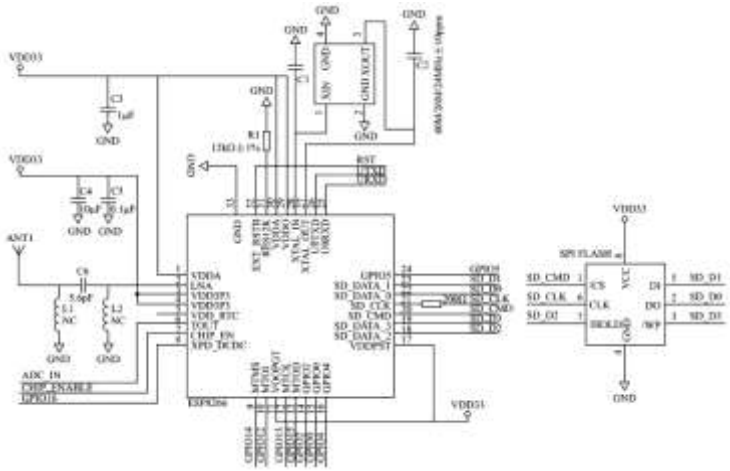

Figure 2.2: Node MCU Schematic Diagram

This an open source IoT platform. It includes firmware which runs on the ESP8266 Wi-Fi SoC from Express if Systems, and hardware which is based on the ESP-12 module. The term "Node MCU" by default refers to the firmware rather than the development kits. The firmware uses the Luascripting language. It is based on the eLua project, and built on

the Espress if Non-OS SDK for ESP8266. Node MCU was created shortly after the ESP8266 came out. On December 30, 2013, Espressif Systems began production of the ESP8266. The ESP8266 is a Wi-Fi SoC integrated with a Tensilica Xtensa LX106 core, widely used in IoT applications (see related projects). Node MCU started on 13 Oct 2014, when Hong committed the first file of nodemcu-firmware to GitHub.

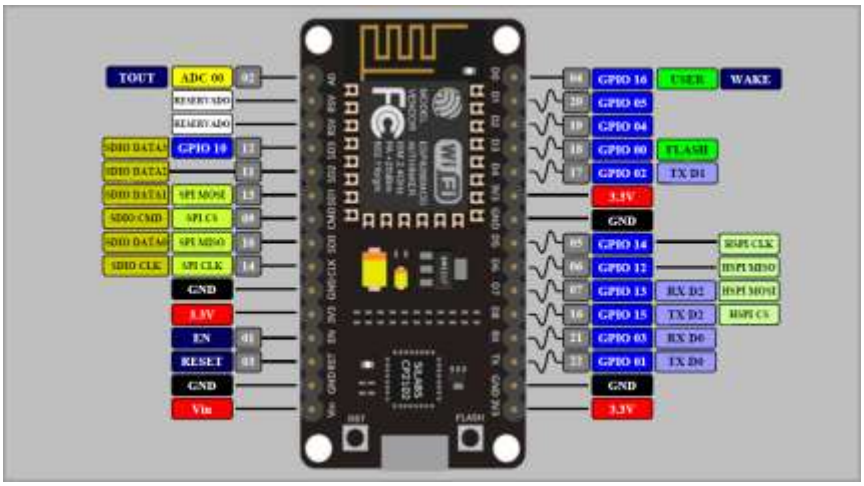

Figure 2.3: Node MCU Pin Out

Node MCU V3 ESP8266 ESP-12E is Wi-Fi development board that helps you to prototype your IoT product with few Lua script lines, or through Arduino IDE. The board is based on ESP8266 ESP-12E variant, unlike other ESP-12E, you won't need to buy a separate breakout board, USB to serial adapter, or even solder it to a PCB to get started, you will only need a USB cable (Micro USB).

Features

1. Communication interface voltage: 3.3V.
2. Antenna type: Built-in PCB antenna is available.
3. Wireless 802.11 b/g/n standard
4. Wi-Fi at 2.4GHz, support WPA / WPA2 security mode
5. Support STA/AP/STA + AP three operating modes

6. Built-in TCP/IP protocol stack to support multiple TCP Client connections (5 MAX)
7. D0 ~ D8, SD1 ~ SD3: used as GPIO, PWM, IIC, etc., port driver capability 15mA
8. AD0: 1 channel ADC
9. Power input: 4.5V ~ 9V (10VMAX), USB-powered
10. Current: continuous transmission: ≈70mA (200mA MAX), Standby: <200uA
11. Transfer rate: 110-460800bps
12. Support UART / GPIO data communication interface
13. Remote firmware upgrade (OTA)
14. Flash size: 4MByte.

2.3 Solar Panel

A solar panel is a set of solar photovoltaic modules electrically connected and mounted on a supporting structure. A photovoltaic module is a packaged, connected assembly of solar cells. The solar panel can be used as a component of a larger photovoltaic system to generate and supply electricity in commercial and residential applications. Each module is rated by its DC output power under standard test conditions (STC), and typically ranges from 100 to 320 watts. The efficiency of a module determines the area of a module given the same rated output - an 8% efficient 230 watt module will have twice the area of a 16% efficient 230 watt module. A single solar module can produce only a limited amount of power; most installations contain multiple modules.

A photovoltaic system typically includes a panel or an array of solar modules, an inverter, and sometimes a battery and/or solar tracker and interconnection wiring. Solar cell modules produce electricity only when the sun is shining. They do not store energy, therefore to ensure flow of electricity when the sun is not shinning, it is necessary to store some of the energy produced. The most obvious solution is to use batteries, which chemically store electric energy. Batteries are groups of electro chemical cells (devices that convert chemical energy to electrical energy) connected in series.

Figure 2.4: Solar Panel

Battery cells are composed of two electrodes immersed in electrolyte solution which produce an electric current when a circuit is formed between them. The current is caused by reversible chemical reactions between the electrodes and the electrolyte within the cell. Batteries that are re-chargeable are called secondary or accumulator batteries. As the battery is being charged, electric energy is stored as chemical energy in the cells. When being discharged, the stored chemical energy is being removed from the battery and converted to electrical energy. In East-Africa, the most common type of secondary battery is the Lead-acid battery.

2.4 Solar Charger Controller

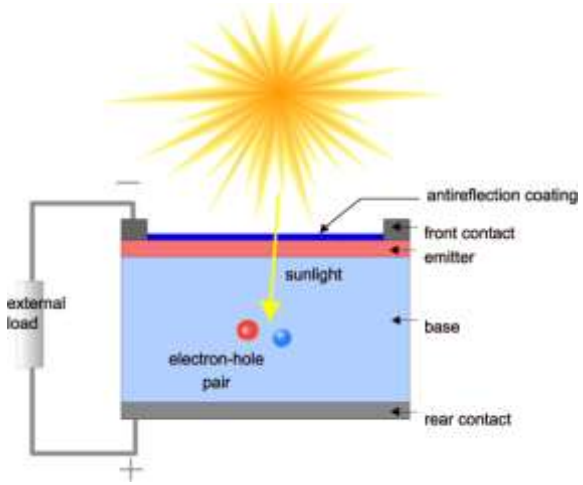

Figure 2.5: Solar Panel Schema Diagram

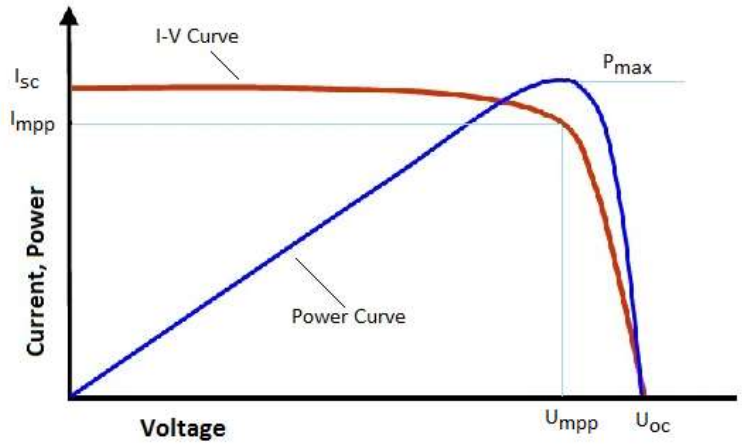

Figure 2.6: Solar Sell System Curve

Here is a solar charger circuit that is used to charge Lead Acid or Ni-Cd batteries using the solar energy power. The circuit harvests solar energy to charge a 6-volt 4.5 Ah rechargeable battery for various applications. The charger has voltage and current regulation and over voltage cut-off facilities.

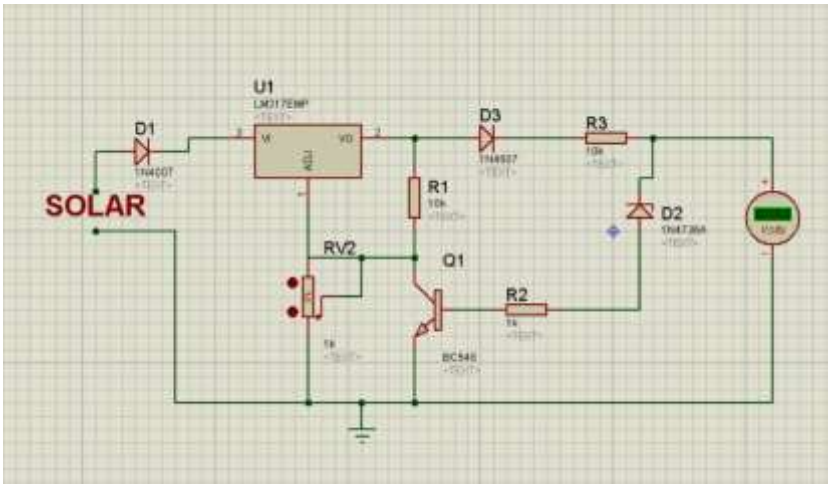

Figure 2.7: Solar Charger Controller Circuit

The circuit uses a 12 volt solar panel and a variable voltage regulator IC LM 317. The solar panel consists of solar cells each rated at 1.2 volts. 12 volt DC is available from the panel to charge the battery. Charging current passes through D1 to the voltage regulator IC LM 317. By adjusting its Adjust pin, output voltage and current can be regulated. VR

is placed between the adjust pin and ground to provide an output voltage of 9 volts to the battery. Resistor R3 Restrict the charging current and diode D2 prevents discharge of current from the battery. Transistor T1 and Zener diode ZD act as a cutoff switch when the battery is full. Normally T1 is off and battery gets charging current. When the terminal voltage of the battery rises above 6.8 volts, Zener conducts and provides base current to T1. It then turns on grounding the output of LM317 to stop charging.

2.5 12v Battery

A twelve-volt battery has six single cells in series producing a fully charged output voltage of 12.6 volts. A battery cell consists of two lead plates a positive plate covered with a paste of lead dioxide and a negative made of sponge lead, with an insulating material (separator) in between.

Quick Details

- Brand Name: DOUBLE TECH(rechargeable 12v dc battery pack)
- Model Number: DBG12-200(12V 200AH)
- Nominal Capacity: 200AH
- Place of Origin: Fujian, China
- Weight: 57kgs
- Certification: ISO9001,CE,MSDS
- rechargeable 12v dc battery pack: 1PCS/CTN(according to the actual situation)
- Production Capacity: rechargeable 12v dc battery pack:50000PCS/Month
- Lifetime: 8-12years
- OEM/ODM: 8 years experience
- Total Workers: 300
- QC Stuffs: 12
- Technical Engineers: 5
- Battery Production Lines: 5
- Factory space: 6000m2

- Maintenance Type: Free
- Voltage: 12V
- Size: 522*240*220/225mm
- Sealed Type: Sealed
- Usage: AGM,UPS

Figure 2.8: 12V Battery

2.6 AC Current Sensor

The module is equipped with the ZMCT103C series of small high-precision current transformers and high-precision op amp circuits for accurate sampling and proper compensation of signals. It is convenient for signal acquisition of AC power within 5A. The corresponding output analog AC signal can be adjusted. The required output voltage can be adjusted according to the potentiometer (adjusting the amplification ratio, the amplification range is 0-100 times), but the maximum voltage at the output (OUT) will not exceed 1/2 VCC

Product Description:

- On-board micro-precision current transformer
- Onboard sampling resistor

- Modules 5A can be measured within an alternating current, the analog output corresponding to 5A/5mA
- Rated input current: 5A
- Rated output current: 5mA
- Change: 1000: 1
- The linear range: 0 ~ 10A (100 ohm)
- Precision rating: 0.2
- Uses isolation voltage: 3000V Measurement
- Sealing material: epoxy resin
- Operating temperature: - 40 Celsius - + 70 Celsius

Figure 2.9: AC Current Sensor

2.7 Voltage Sensor

ZMPT101B AC Single Phase voltage sensor module is based on a high precision ZMPT101B voltage Transformer. ZMPT101B AC Voltage Sensor is the best for the purpose of the DIY project, where we need to measure the accurate AC voltage with a voltage transformer. This is an ideal choice to measure the AC voltage using Arduino /ESP8266 /Raspberry Pi like an open-source platform. In many electrical projects, engineer directly deals with measurements with few basic requirements like High galvanic isolation, Wide Range, High accuracy, Good Consistency.

Onboard precision miniature voltage transformer, The active phase AC output voltage transformer module. Onboard precision op-amp circuit, the signal sampling and appropriate compensation for precise functions. Modules can be measured within 250V AC voltage, the corresponding analog output can be adjusted. It is brand new, good quality high performance.

FEATURES of ZMPT101B AC Single Phase Voltage Sensor Module

- Voltage up to 250 volts can be measured
- Light weight with on-board micro-precision voltage transformer
- High precision on-board op-amp circuit
- Operating temperature: 40ºC ~ + 70ºC
- Supply voltage 5 volts to 30 volts

ADVANTAGES of ZMPT101B AC Single Phase Voltage Sensor Module

- Analog output corresponding quantity can be adjusted.
- Pcb board size: 49.5 (mm) x19.4 (mm)
- Good consistency, for voltage and power measurement
- Very efficient and accuracy

Figure 2.10: Voltage Sensor

2.8 Relay

A relay is an electrically operated switch. Many relays use an electromagnet to mechanically operate a switch, but other operating principles are also used, such as solid-state relays. Relays are used where it is necessary to control a circuit by a separate low-power signal, or where several circuits must be controlled by one signal. The first relays were used in long distance telegraph circuits as amplifiers: they repeated the signal coming in from one circuit and re-transmitted it on another circuit. Relays were used extensively in telephone exchanges and early computers to perform logical operations.

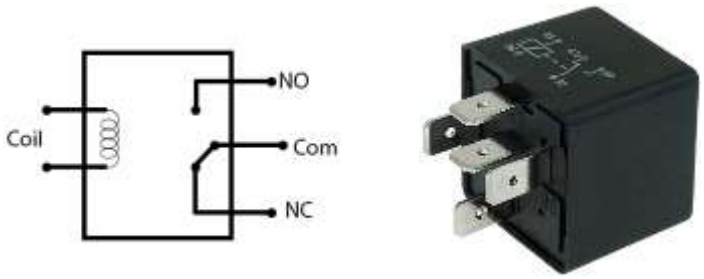

Figure 2.11: Relay

A type of relay that can handle the high power required to directly control an electric motor or other loads is called a contactor. Solid-state relays control power circuits with no moving parts, instead using a semiconductor device to perform switching. Relays with calibrated operating characteristics and sometimes multiple operating coils are used to protect electrical circuits from overload or faults; in modern electric power systems these functions are performed by digital instruments still called "protective relays".

Magnetic latching relays require one pulse of coil power to move their contacts in one direction, and another, redirected pulse to move them back. Repeated pulses from the same input have no effect. Magnetic latching relays are useful in applications where interrupted power should not be able to transition the contacts. Magnetic latching relays can have either single or dual coils. On a single coil device, the relay will operate in one direction when power is applied with one polarity, and will reset when the polarity is

reversed. On a dual coil device, when polarized voltage is applied to the reset coil the contacts will transition. AC controlled magnetic latch relays have single coils that employ steering diodes to differentiate between operate and reset commands.

The circuit above is called a low-side switch, because the switch – our transistor – is on the low (ground) side of the circuit. Alternatively, we can use a PNP transistor to create a high-side switch: Similar to the NPN circuit, the base is our input, and the emitter is tied to a constant voltage.

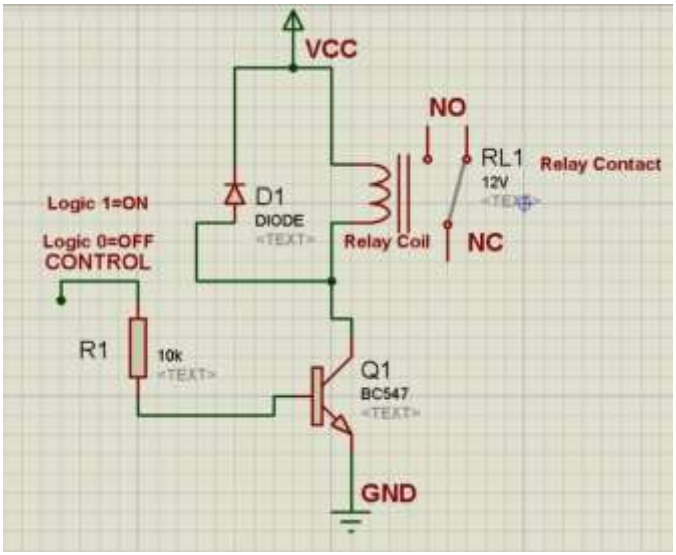

Figure 2.12: Transistor Switching Circuit.

A relay is an electrically operated switch of mains voltage. It means that it can be turned on or off, letting the current go through or not. Controlling a relay with the Arduino is as simple as controlling an output such as an LED. The relay module is the one in the figure below.

Figure 2.13: Relay Module.

This module has two channels (those blue cubes). There are other varieties with one, four and eight channels.

Mains voltage connections:

In relation to mains voltage, relays have 3 possible connections:

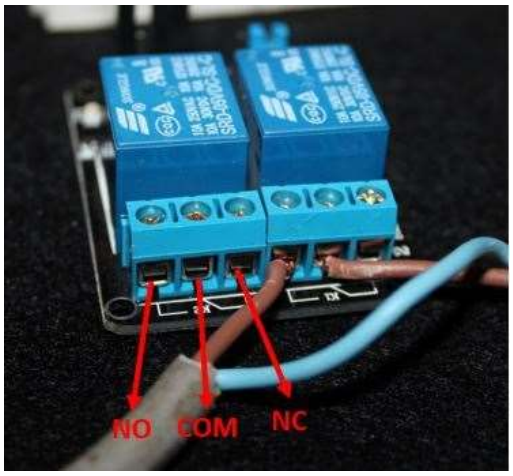

Figure 2.14: Pin diagram of Relay Module

- **COM**: common pin
- **NO (Normally Open):** there is no contact between the common pin and the normally open pin. So, when you trigger the relay, it connects to the COM pin and supply is provided to a load

- **NC (Normally Closed):** there is contact between the common pin and the normally closed pin. There is always connection between the COM and NC pins, even when the relay is turned off. When you trigger the relay, the circuit is opened and there is no supply provided to a load. If you want to control a lamp for example, it is better to use a normally-open circuit, because we just want to light up the lamp occasionally.

Pin wiring:

The connections between the relay module and the Arduino are really simple:

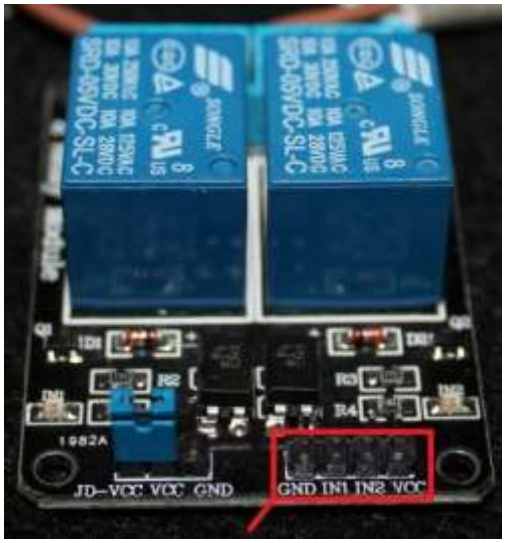

Figure 2.15: Main Voltage Connection

- **GND**: goes to ground
- **IN1**: controls the first relay (it will be connected to an Arduino digital pin)
- **IN2**: controls the second relay (it should be connected to an Arduino digital pin if you are using this second relay. Otherwise, you don't need to connect it)
- **VCC**: goes to 5V

2.9 100W Inverter Circuit:

- Power: 100W
- Input: DC12V
- Output: 0-110-220V
- Static no-load current: 0.1A or so
- Output frequency waveform: about 15KHZ high-frequency square wave
- Protection: No protection

Figure 2.16: Inverter Circuit

Features:

1. This board is a high-frequency square wave AC output; it is generally not with the emotional electrical!" Such as: motor fan coil transformer, etc." Another: simple resistance-type buck-type small electrical appliances, cheap resistive-capacity buck LED lighting, etc. may not take

2. Load the electronic products (such as: energy-saving lamps LED lights mobile phone charger DVD set-top boxes, etc.) It is recommended that the board of the high-frequency rectifier short pieces of unplug or low-voltage gear, such as: 110V use

3. In use, to consider the power supply should not be a long time continuously in full load working condition. For continuous long-term working conditions, it is recommended to run 80% of the efficiency of switching power supply. Leaving room for use, to ensure long-term power supply, or it may cause artificial failure in the early.

4. The power supply cannot be reversed, and the other power supply and power cable must be large enough

5. Do not overload because there is no protection output Do not short-circuit.

6. Output high-frequency square-wave AC voltage General multimeter will not measure or show a lot of low or high (is not correct, not the actual voltage)

2.10 Capacitor

Capacitor is an electronic component that stores electric charge. The capacitor is made of 2 close conductors (usually plates) that are separated by a dielectric material. The plates accumulate electric charge when connected to power source. One plate accumulates positive charge and the other plate accumulates negative charge. The capacitance is the amount of electric charge that is stored in the capacitor at voltage of 1 Volt. The capacitance is measured in units of (F).The capacitor disconnects current in direct current (DC) circuits and short circuit in alternating current (AC) circuits.

Figure 2.17: Capacitor Pin Out

Pin Configuration

The **Electrolytic Capacitors** have polarity. Meaning they have a positive and negative pin. The pin which is long is the positive pin and the pin which is short is the negative pin**.** You can also identify the polarity using the negative strip on the capacitor label. As shown in the picture above the negative pin will be directly under the negative symbol.

Features

- Capacitor Type - Electrolytic
- Has a high range of capacitance value starting from 0.01uF to 10000uF
- Has a high range of voltage value starting from 16V to 450V
- Can withstand a maximum of 105°C temperature

Other types of Capacitors

Ceramic Capacitor, Box Capacitor, Variable Capacitor.

Capacitor parameters selection

Ever wondered about the types of Electrolytic capacitors available in market and how to select one for your project? Electrolytic Capacitors can be classified based on two main parameters. One is their **Capacitance(C-Farad)** itself and the other is its **Voltage (V-Volts)** rating.

Capacitor is a passive component which can store a charge (Q). This charge (Q) will be a product of the value of capacitance (C) and the voltage (V) applied to it. The value of the capacitance and Voltage of a capacitor will be mentioned on its label.

Hence the amount of charge a capacitor can be found using the value of Voltage (V) and Capacitance (C) of the capacitor.

$$\mathbf{C = Q \times V}$$

Precaution

While using an Electrolytic capacitor care should always be taken to connect the positive pin to the positive of the circuit and the negative pin to the negative of the circuit. Also the voltage appearing across the capacitor terminals should always be less than the rated capacitor voltage (V). Failing to do so will lead to abnormal heating of the capacitor and might even burst.

Capacitor in series and parallel

In most of the circuits the value of the capacitance need not be exactly the same value specified in the circuit. A higher value of capacitance will generally not affect the performance of the circuit. However, the value of voltage should be the same or higher than the specified value to prevent the risk mentioned in precaution above. In that case, if you do not have the exact value you can use to capacitors in series or parallel to attain the desired value.

When two capacitors are *connected in parallel* then, the value of the capacitance(C) gets directly added up and the rated voltage (V) is remains the same in parallel as shown in the picture below.

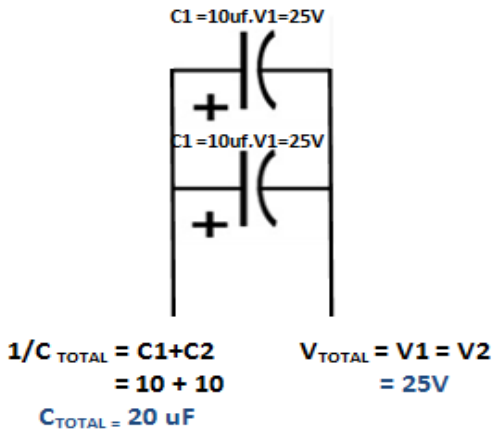

Figure 2.18: Capacitor Operation

2.11 Resistor

A **resistor** is a passive two-terminal electrical component that implements electrical resistance as a circuit element. In electronic circuits, resistors are used to reduce current flow, adjust signal levels, to divide voltages, bias active elements, and terminate transmission lines, among other uses. High-power resistors that can dissipate many watts of electrical power as heat, may be used as part of motor controls, in power distribution systems, or as test loads for generators. Fixed resistors have resistances that only change slightly with temperature, time or operating voltage. Variable resistors can be used to adjust circuit elements (such as a volume control or a lamp dimmer), or as sensing devices for heat, light, humidity, force, or chemical activity.

Two typical schematic diagram symbols are as follows:

(a) resistor, (b) rheostat (variable resistor), and (c) potentiometer

IEC resistor symbol

The notation to state a resistor's value in a circuit diagram varies.

One common scheme is the RKM code following IEC 60062. It avoids using a decimal separator and replaces the decimal separator with a letter loosely associated with SI prefixes corresponding with the part's resistance. For example, *8K2* as part marking code, in a circuit diagram or in a bill of materials (BOM) indicates a resistor value of 8.2 kΩ. Additional zeros imply a tighter tolerance, for example *15M0* for three significant digits. When the value can be expressed without the need for a prefix (that is, multiplicator 1), an "R" is used instead of the decimal separator. For example, *1R2* indicates 1.2 Ω, and *18R* indicates 18 Ω.

Specifications:

- Resistance: 220 Ohms
- Power (Watts): 0.25W, 1/4W
- Temperature Coefficient: 350ppm/Celcius
- Tolerance: +/- 5%
- Case: Axial

CHAPTER 3

SOFTWARE IMPLEMENTATION

3.1 Arduino Software

The digital microcontroller unit named as Arduino Nano can be programmed with the Arduino software IDE. There is no any requirement for installing other software rather than Arduino. Firstly, Select "Arduino Nano from the Tools, Board menu (according to the microcontroller on our board). The IC used named as ATmega328 on the Arduino Nano comes pre burned with a boot loader that allows us to upload new code to it without the use of an external hardware programmer.

Communication is using the original STK500 protocol (reference, C header files). We can also bypass the boot loader and programs the microcontroller through the ICSP (In Circuit Serial Programming) header. The ATmega16U2 (or 8U2 in the rev1 and rev2 boards) firmware source code is available. The ATmega16U2/8U2 is loaded with a DFU boot loader, which can be activated by:

On Rev1 boards: connecting the solder jumper on the back of the board (near the map of Italy) and then resetting the 8U2. On Rev2 or later boards: there is a resistor that pulling the 8U2/16U2 HWB line to ground, making it easier to put into DFU mode.

The Arduino Nano is one of the latest digital microcontroller units and has a number of facilities for communicating with a computer, another Arduino, or other microcontrollers. The ATmega328 provides UART TTL at (5V) with serial communication, which is available on digital pins 0 -(RX) for receive the data and pin no.1 (TX) for transmit the data. An ATmega16U2 on the board channels this serial communication over USB and appears as a virtual com port to software on the computer. The '16U2 firmware uses the standard USB COM drivers, and no external driver is needed. However, on Windows, an .in file is required. The Arduino software includes a serial monitor which allows simple textual data to be sent to and from the Arduino board.

The RX and TX LEDs on the board will flash when data is being transmitted via the USB-to-serial chip and USB connection to the computer (but not for serial Communication on pins 0 and 1). A Software Serial library allows for serial communication on any of the Nano's digital pins. The ATmega328 also supports I2C (TWI) and SPI communication. The Arduino software includes a Wire library to simplify use of the I2C bus. Arduino programs are written in C or C++ and the program code written for Arduino is called sketch. The Arduino IDE uses the GNU tool chain and AVR Lab to compile programs, and for uploading the programs it uses argued. As the Arduino platform uses Atmel microcontrollers, Atmel's development environment, AVR Studio or the newer Atmel Studio, may also be used to develop software for the Arduino.

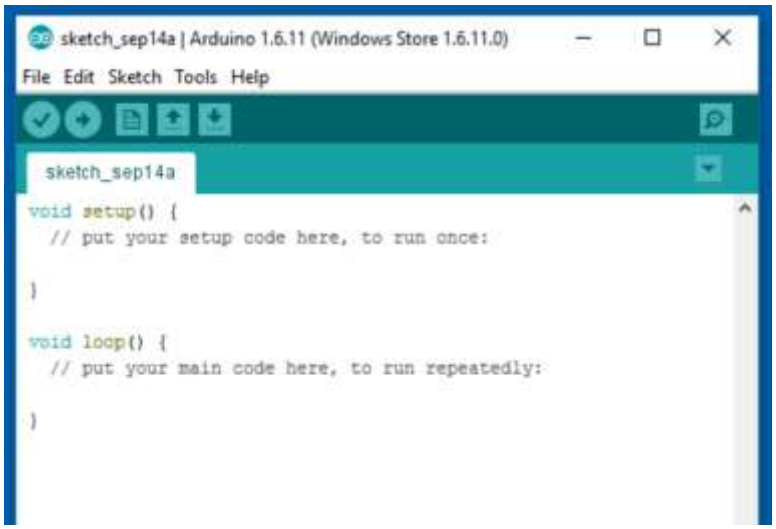

Figure 3.1: Arduino Software Interface IDE

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.

Writing Sketches

Programs written using Arduino Software (IDE) are called sketches. These sketches are written in the text editor and are saved with the file extension .ino. The editor has features for cutting/pasting and for searching/replacing text. The message area gives feedback while saving and exporting and also displays errors. The console displays text output by the Arduino Software (IDE), including complete error messages and other information. The bottom right hand corner of the window displays the configured board and serial port. The toolbar buttons allow you to verify and upload programs, create, open, and save sketches, and open the serial monitor.

NB: Versions of the Arduino Software (IDE) prior to 1.0 saved sketches with the extension .pde. It is possible to open these files with version 1.0, you will be prompted to save the sketch with the .ino extension on save.

Verify

Checks your code for errors compiling it.

Upload

Compiles your code and uploads it to the configured board. See uploading below for details.

Note: If you are using an external programmer with your board, you can hold down the "shift" key on your computer when using this icon. The text will change to "Upload using Programmer"

New

Creates a new sketch.

Open

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

Note: due to a bug in Java, this menu doesn't scroll; if you need to open a sketch late in the list, use the File | Sketchbook menu instead.

Save

Saves your sketch.

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.

File

New

Creates a new instance of the editor, with the bare minimum structure of a sketch already in place.

Open

Allows to load a sketch file browsing through the computer drives and folders.

Open Recent

Provides a short list of the most recent sketches, ready to be opened.

Sketchbook

Shows the current sketches within the sketchbook folder structure; clicking on any name opens the corresponding sketch in a new editor instance.

Examples

Any example provided by the Arduino Software (IDE) or library shows up in this menu item. All the examples are structured in a tree that allows easy access by topic or library.

Close

Closes the instance of the Arduino Software from which it is clicked.

Save

Saves the sketch with the current name. If the file hasn't been named before, a name will be provided in a "Save as.." window.

Save as...

Allows to save the current sketch with a different name.

Page Setup

It shows the Page Setup window for printing.

Print

Sends the current sketch to the printer according to the settings defined in Page Setup.

Preferences

Opens the Preferences window where some settings of the IDE may be customized, as the language of the IDE interface.

Quit

Closes all IDE windows. The same sketches open when Quit was chosen will be automatically reopened the next time you start the IDE.

Edit

Undo/Redo

Goes back of one or more steps you did while editing; when you go back, you may go forward with Redo.

Cut

Removes the selected text from the editor and places it into the clipboard.

Copy

Duplicates the selected text in the editor and places it into the clipboard.

Copy for Forum

Copies the code of your sketch to the clipboard in a form suitable for posting to the forum, complete with syntax coloring.

Copy as HTML

Copies the code of your sketch to the clipboard as HTML, suitable for embedding in web pages.

Paste

Puts the contents of the clipboard at the cursor position, in the editor.

Select All

Selects and highlights the whole content of the editor.

Comment/Uncomment

Puts or removes the // comment marker at the beginning of each selected line.

Increase/Decrease Indent

Adds or subtracts a space at the beginning of each selected line, moving the text one space on the right or eliminating a space at the beginning.

Find

Opens the Find and Replace window where you can specify text to search inside the current sketch according to several options.

Find Next

Highlights the next occurrence - if any - of the string specified as the search item in the Find window, relative to the cursor position.

Find Previous

Highlights the previous occurrence - if any - of the string specified as the search item in the Find window relative to the cursor position.

Sketch

Verify/Compile

Checks your sketch for errors compiling it; it will report memory usage for code and variables in the console area.

Upload

Compiles and loads the binary file onto the configured board through the configured Port.

Upload Using Programmer

This will overwrite the bootloader on the board; you will need to use Tools > Burn Bootloader to restore it and be able to Upload to USB serial port again. However, it allows you to use the full capacity of the Flash memory for your sketch. Please note that this command will NOT burn the fuses. To do so a Tools -> Burn Bootloader command must be executed.

Export Compiled Binary

Saves a .hex file that may be kept as archive or sent to the board using other tools.

Show Sketch Folder

Opens the current sketch folder.

Include Library

Adds a library to your sketch by inserting #include statements at the start of your code. For more details, see libraries below. Additionally, from this menu item you can access the Library Manager and import new libraries from .zip files.

Add File...

Adds a source file to the sketch (it will be copied from its current location). The new file appears in a new tab in the sketch window. Files can be removed from the sketch using the tab menu accessible clicking on the small triangle icon below the serial monitor one

on the right side o the toolbar.

Tools

Auto Format

This formats your code nicely: i.e. indents it so that opening and closing curly braces line up, and that the statements inside curly braces are indented more.

Archive Sketch

Archives a copy of the current sketch in .zip format. The archive is placed in the same directory as the sketch.

Fix Encoding & Reload

Fixes possible discrepancies between the editor char map encoding and other operating systems char maps.

Serial Monitor

Opens the serial monitor window and initiates the exchange of data with any connected board on the currently selected Port. This usually resets the board, if the board supports Reset over serial port opening.

Board

Select the board that you're using. See below for descriptions of the various boards.

Port

This menu contains all the serial devices (real or virtual) on your machine. It should automatically refresh every time you open the top-level tools menu.

Programmer

For selecting a hardware programmer when programming a board or chip and not using the onboard USB-serial connection. Normally you won't need this, but if you're burning a boot loader to a new microcontroller, you will use this.

Burn Boot loader

The items in this menu allow you to burn a boot loader onto the microcontroller on an Arduino board. This is not required for normal use of an Arduino or Genuino board but is useful if you purchase a new ATmega microcontroller (which normally come without a bootloader). Ensure that you've selected the correct board from the Boards menu before burning the bootloader on the target board. This command also set the right fuses.

Help

Here you find easy access to a number of documents that come with the Arduino Software (IDE). You have access to Getting Started, Reference, this guide to the IDE and other documents locally, without an internet connection. The documents are a local copy of the online ones and may link back to our online website.

Find in Reference

This is the only interactive function of the Help menu: it directly selects the relevant page in the local copy of the Reference for the function or command under the cursor.

Sketchbook

The Arduino Software (IDE) uses the concept of a sketchbook: a standard place to store your programs (or sketches). The sketches in your sketchbook can be opened from the File > Sketchbook menu or from the Open button on the toolbar. The first time you run the Arduino software, it will automatically create a directory for your sketchbook. You can view or change the location of the sketchbook location from with the Preferences dialog.

Beginning with version 1.0, files are saved with a .ino file extension. Previous versions use the .pde extension. You may still open .pde named files in version 1.0 and later, the software will automatically rename the extension to .ino.

Tabs, Multiple Files, and Compilation

Allows you to manage sketches with more than one file (each of which appears in its own tab). These can be normal Arduino code files (no visible extension), C files (.c extension), C++ files (.cpp), or header files (.h).

Uploading

Before uploading your sketch, you need to select the correct items from the Tools > Board and Tools > Port menus. The boards are described below. On the Mac, the serial port is probably something like /dev/tty.usbmodem241 (for an Uno or Mega2560 or Leonardo) or /dev/tty.usbserial-1B1 (for a Duemilanove or earlier USB board), or /dev/tty.USA19QW1b1P1.1 (for a serial board connected with a Keyspan USB-to-Serial adapter). On Windows, it's probably COM1 or COM2 (for a serial board) or COM4, COM5, COM7, or higher (for a USB board) - to find out, you look for USB serial device in the ports section of the Windows Device Manager. On Linux, it should be /dev/ttyACMx , /dev/ttyUSBx or similar. Once you've selected the correct serial port and board, press the upload button in the toolbar or select the Upload item from the Sketch menu. Current Arduino boards will reset automatically and begin the upload. With older boards (pre-Diecimila) that lack auto-reset, you'll need to press the reset button on the board just before starting the upload. On most boards, you'll see the RX and TX LEDs blink as the sketch is uploaded. The Arduino Software (IDE) will display a message when the upload is complete, or show an error.

When you upload a sketch, you're using the Arduino bootloader, a small program that has been loaded on to the microcontroller on your board. It allows you to upload code without using any additional hardware. The bootloader is active for a few seconds when the board resets; then it starts whichever sketch was most recently uploaded to the microcontroller. The bootloader will blink the on-board (pin 13) LED when it starts (i.e. when the board resets).

Libraries

Libraries provide extra functionality for use in sketches, e.g. working with hardware or manipulating data. To use a library in a sketch, select it from the Sketch > Import Library menu. This will insert one or more #include statements at the top of the sketch and compile the library with your sketch. Because libraries are uploaded to the board with your sketch, they increase the amount of space it takes up. If a sketch no longer needs a library, simply delete its #include statements from the top of your code.

There is a list of libraries in the reference. Some libraries are included with the Arduino software. Others can be downloaded from a variety of sources or through the Library Manager. Starting with version 1.0.5 of the IDE, you do can import a library from a zip file and use it in an open sketch. See these instructions for installing a third-party library.

Third-Party Hardware

Support for third-party hardware can be added to the hardware directory of your sketchbook directory. Platforms installed there may include board definitions (which appear in the board menu), core libraries, bootloaders, and programmer definitions. To install, create the hardware directory, then unzip the third-party platform into its own sub-directory. (Don't use "arduino" as the sub-directory name or you'll override the built-in Arduino platform.) To uninstall, simply delete its directory.

For details on creating packages for third-party hardware, see the Arduino IDE 1.5 3rd party Hardware specification.

Serial Monitor

This displays serial sent from the Arduino or Genuino board over USB or serial connector. To send data to the board, enter text and click on the "send" button or press enter. Choose the baud rate from the drop-down menu that matches the rate passed to Serial.begin in your sketch. Note that on Windows, Mac or Linux the board will reset (it will rerun your sketch) when you connect with the serial monitor. Please note that the Serial Monitor does not process control characters; if your sketch needs a complete management of the serial communication with control characters, you can use an external terminal program and connect it to the COM port assigned to your Arduino board.

You can also talk to the board from Processing, Flash, MaxMSP, etc (see the interfacing page for details).

3.2 Proteus Software

The Proteus Design Suite is a proprietary software tool suite used primarily for electronic design automation. The software is used mainly by electronics design engineers and technicians to create schematics and electronics prints for manufacturing printed circuit boards.

The first version of what is now the Proteus Design Suite was called PC-B and was written by the company chairman, John Jameson, for DOS in 1988. Schematic Capture support followed in 1990 with a port to the Windows environment shortly thereafter. Mixed mode SPICE Simulation was first integrated into Proteus in 1996 and microcontroller simulation then arrived in Proteus in 1998. Shape based auto routing was added in 2002 and 2006 saw another major product update with 3D Board Visualization. More recently, a dedicated IDE for simulation was added in 2011 and MCAD import/export was included in 2015. Support for high speed design was added in 2017. Feature led product releases are typically biannual, while maintenance based service packs are released as required.

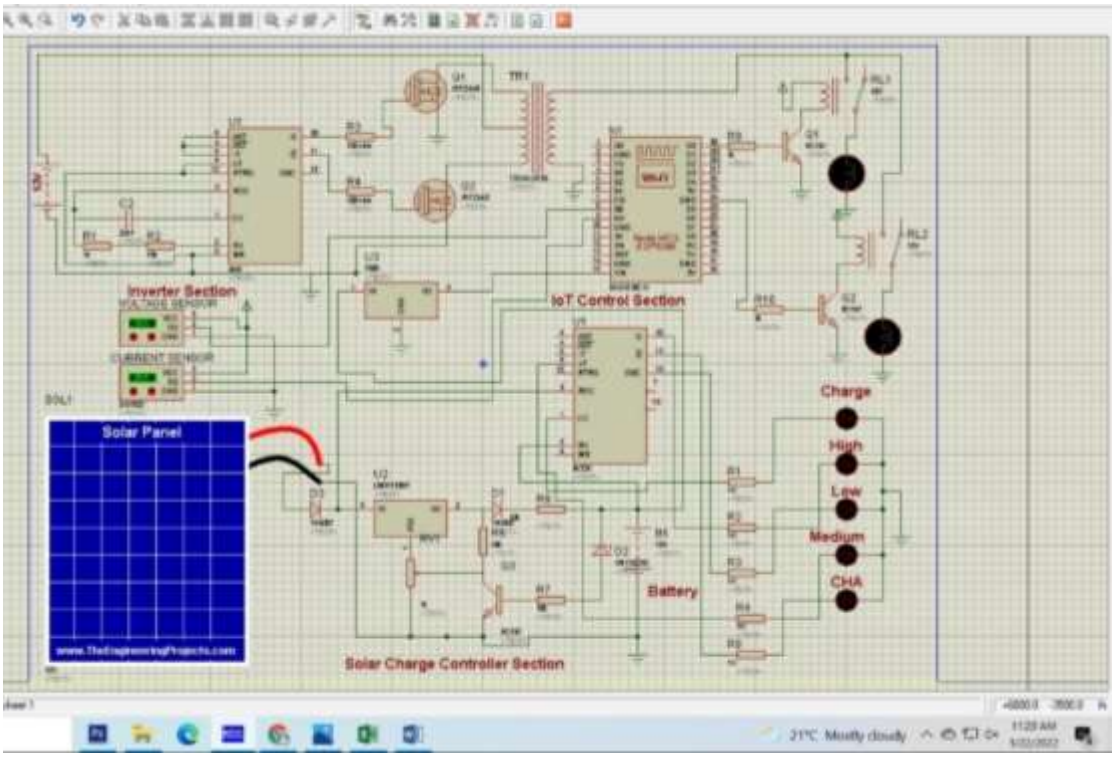

Figure 3.2: Our Project Design in Proteus Software

3.3 Adafruit Server:

Adafruit IO is a system that makes data useful. Our focus is on ease of use, and allowing simple data connections with little programming required. On an Arduino there are two different libraries you can use to access Adafruit IO. One library is based on the REST API, and the other library is based on the MQTT API. The difference between these libraries is that MQTT keeps a connection to the service open so it can quickly respond to feed changes. The REST API only connects to the service when a request is made so it's a more appropriate choice for projects that sleep for a period of time (to reduce power usage) and wake up only to send/receive data. If you aren't sure which library to use, try starting with the Adafruit MQTT library below.

Adafruit MQTT Client Library To use Adafruit IO with the MQTT protocol on an Arduino you can use the Adafruit MQTT Arduino library (https://adafru.it/fp6). This is a general-purpose MQTT library for Arduino that's built to use as few resources as possible so that it can work with platforms like the Arduino Uno. Unfortunately, platforms like the Trinket 3.3V or 5V (based on the ATtiny85) have too little program memory to use the library--stick with a Pro Trinket or better.

Adafruit IO

Having a better understanding about the Internet of Things and Cloud computing, let’s now go over what Adafruit IO is about, and how it works. With the rise in digital transformations, IoT deployments in the cloud have become more popular. By deploying IoT solutions on the cloud, we have the following benefits:

- **Cost** - Reduces the cost of computing and storage by using various cloud services.
- **Scalability** - The “pay-as-you-go” pricing model allows a flexible pay model, also allowing scalability of the application.
- **Data control** - Data backup and recovery with high security.
- **Server uptime** - Allows very minimum or no downtime, with high server availability.

- Adafruit IO is one such cloud provider focusing more on IoT deployments on the cloud. Adafruit IO supports different hardware like Raspberry PI, ESP2866, and Arduino. IoT developers prefer Adafruit IO over other IoT cloud providers for the following reasons:
- **Powerful API** - Provides us libraries for various programming languages, which also provides the built-in user interface support.
- **Dashboard -** Understanding data via charts and graphs enables us to make better decisions.
- **Privacy -** Data is secured in the cloud platform with better encryption algorithms.
- **Documentation & Community -** Many blogs with amazing community support allows continuous developments of the products.

Objective

we worked with NodeMCU on blinking lights of built-in LEDs. In this article, to demonstrate Adafruit IO works, we will send (publish) the LED brightness readings to the Adafruit IO cloud via Arduino IDE (written in C), and receive (subscribe) them via a Python server.

Installation

Follow the installation guide to get setup:

- Sign up by creating a new account in Adafruit IO.
- Make note of your private key by heading over to the “My Key” section.
- Then, “Create a new dashboard” with the desired name.
- In local setup, install a Python package called Adafruit_IO using pip install Adafruit_IO.
- Similarly, make sure to go through the previous article on how to set up the existing development environment.

CHAPTER 4

SYSTEM ARCHITECTURE

4.1 Block Diagram

In this project, we have used a Solar panel for main power source. Battery will store all the power of solar panel which will create with the help of sun ray. Battery will provide the energy to run the home load. All kind of instrument will connect with the micro controller. Because controller will control this whole system

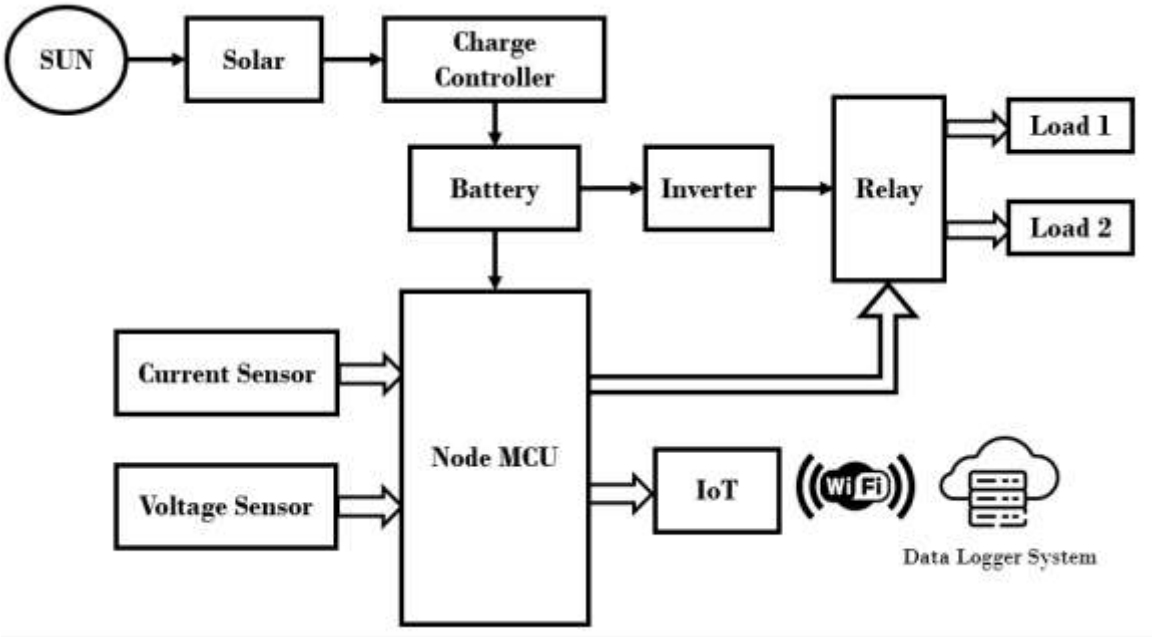

Figure 4.1: Block Diagram of IoT Based Smart Grid System

4.2 Circuit Diagram

The schematic diagram here is representing the electrical circuit and the components of the project. Here we have used standardized symbols and lines.

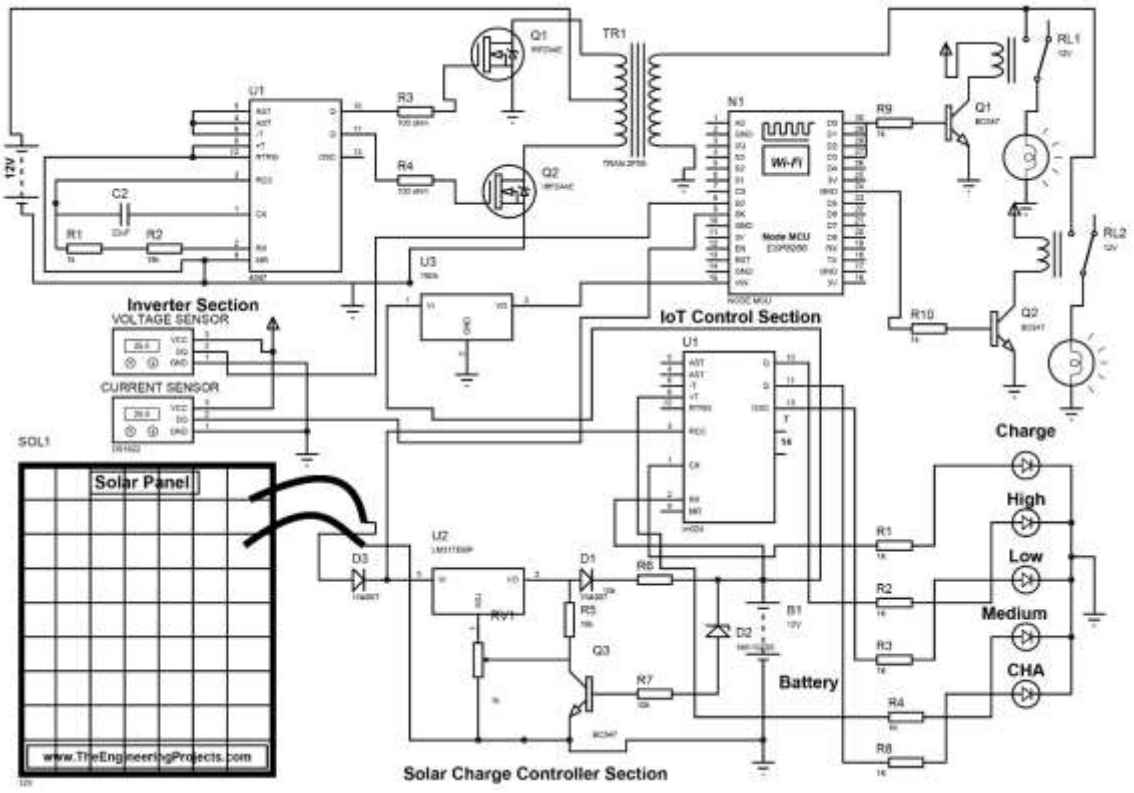

Figure 4.2: Circuit Diagram of IoT Based Smart Grid System.

4.3 Working Principle

This is IoT Based Smart Grid System. The main brain of our system is the Node MCU. The way of whole project works is that we take solar power store in a battery. The main power of our system comes from solar cells. Solar Power will store in battery for subsequent supply. With Node MCU this system can be monitored via internet. This battery system supplies 12 volts, an inverter converts it DC to AC and supplies 220V AC in this system.

The first load of this system is 40 Watt and the second load is 25 watts. Now if this system can get more than 30Watt power at second load then the whole system will automatically shut down for 5 seconds. After 5 seconds the system will be on automatic again if the situation is OK. If it doesn't pass the previous total watt, the system will be turned off again. The system ampere and watt can be seen through mobile.

4.4 The Project Prototype

The complete prototype of our project is shown below:

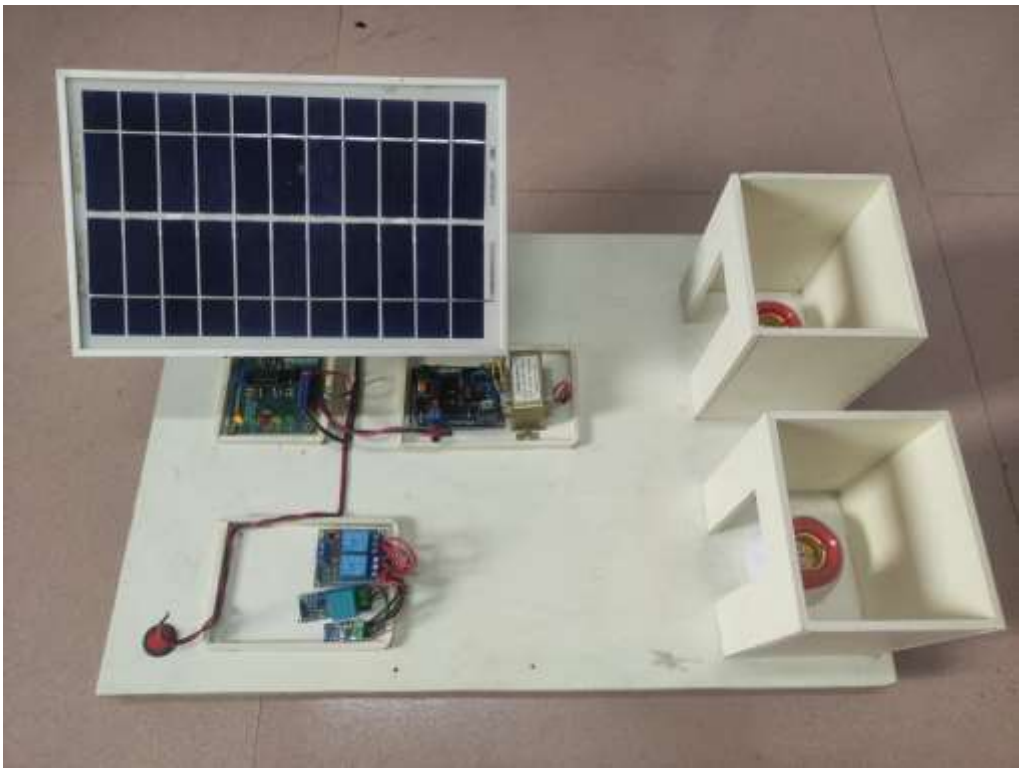

Figure 4.3: The Complete Prototype of the Project

4.5 Result

This chapter contains the results obtained and discussion about the full project. Our project is IoT Based Smart Grid System. In our project making we used PVC boards for total

hardware making. After finally completing this project, we ran it & we observed the output of this project. We can see that it is working well as expected. After making our project we observe it very careful. It works as we desire. Our project give output perfectly and all equipment are work perfectly. We check how much it works and we get perfect output from this project.

- Finally, we have completed our project successfully & check our project its run accurately according to our objective.
- Firstly, solar absorb the sun ray and store its power in the battery.
- Inverter circuit convert DC to AC for AC load.
- Firstly, load will be power from battery.
- If the second load cross over 30 Watts, then system will be shut automatically.
- All data of this project (Watt and Ampere) will show in mobile apps.

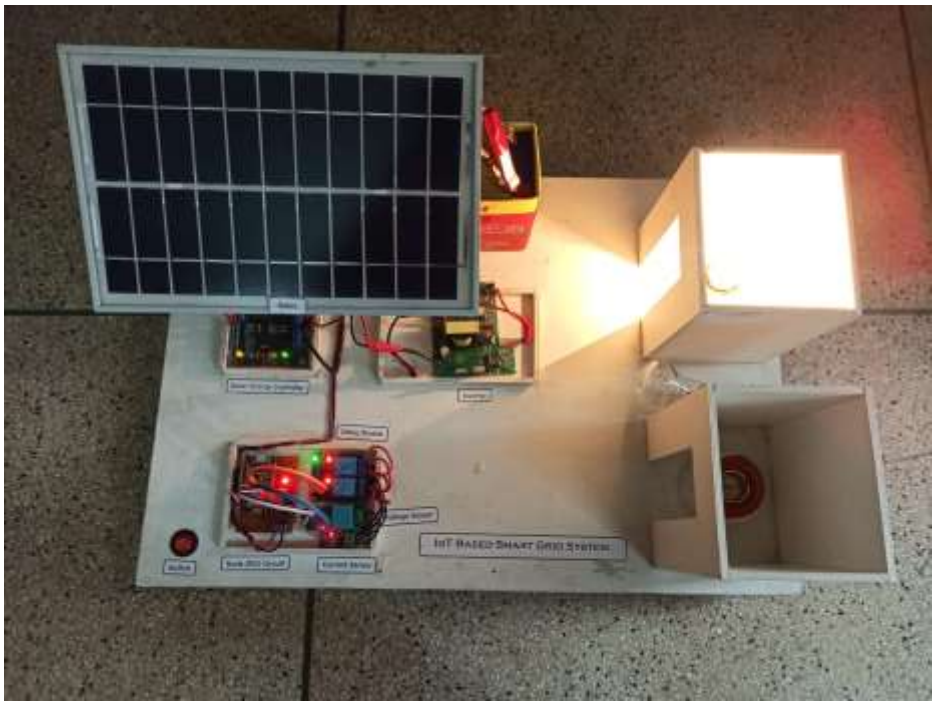

Figure 4.4: One Load on in this project.

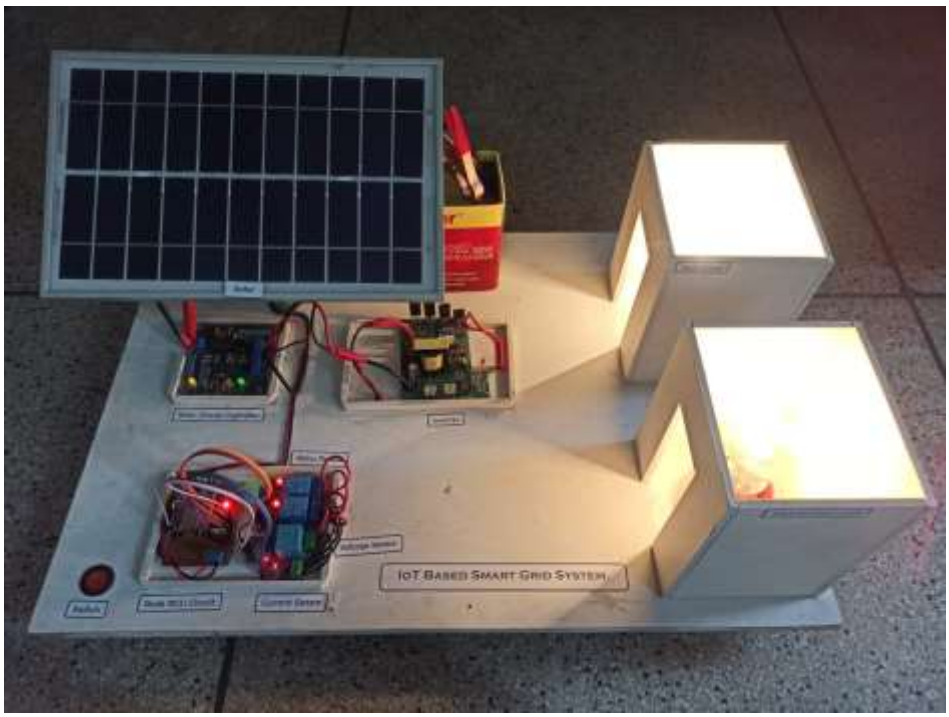

Figure 4.5: Two Load's on in this project.

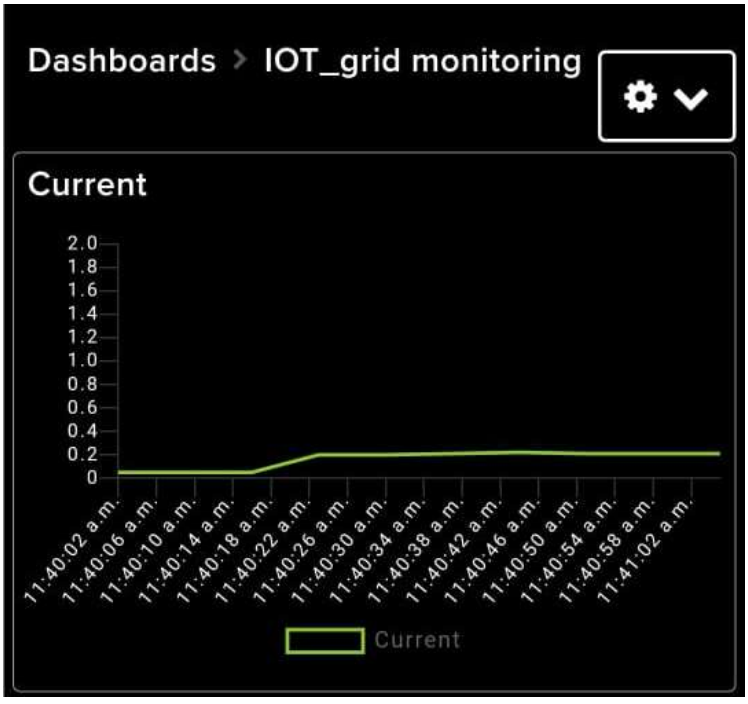

Figure 4.6: Two Load's Ampere reading in apps

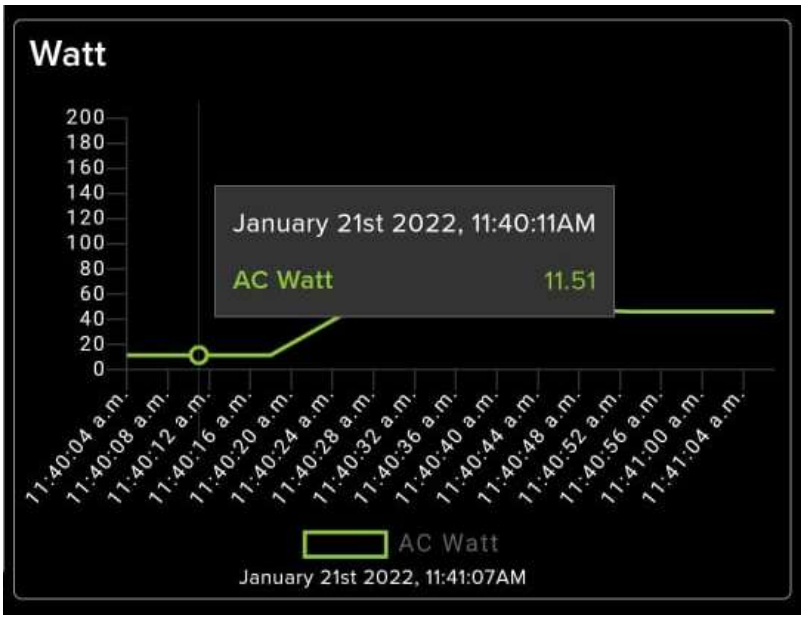

Figure 4.7: Two Load's Watt Reading in apps.

4.6 Cost Analysis

In the below table we have summarized our project expenditure.

Table 01: Cost of Components with Price

No.	Product Name	Specification	Qty. Price	Unit Price (Taka)	Total Price (Taka)
01	Solar	6Watt	1	550	550
02	Inverter	500Watt	1	850	850
03	Charge Controller		1	280	280
04	Battery	12V	1	1050	1050
05	Node MCU	ESP-8266	1	580	580
07	Relay	5V (2 Channel)	1	150	150
08	Current Sensor		1	320	320
09	Voltage Sensor		1	350	350
10	Others				1250
				Total =	5380/=

CHAPTER 5
DISCUSSION & CONCLUSION

5.1 Discussion

In this project we collect all of the instruments, make a circuit design and connect all of these elements sequentially. IoT based smart grid system for the automatic protection between 2 home or user, which is mainly used in automatic daily life. This project has some advantages, that are it's an ecofriendly project, low management cost, installation cost is low. Anywhere you can install by little effort.

5.2 Conclusion

The proposed system makes the energy unit reading to be handy and calculates the energy and the energy consumption of household along. Hence it creates awareness on power consumption and a way to save and manage power by each individual consumer. It has a vast importance in the field of energy management and monitoring and it provides the interaction between customer and the service providers. The proposed project with its real time application for power management and monitoring provides a reliable, comfortable and important application at the same time it avoids human intervention. Advancements in the technology further make the implementation of this system easier in future. The proposed project with its real time application for power management and monitoring provides a reliable, comfortable and important application at the same time it avoids human intervention. Advancements in the technology further make the implementation of this system easier in future.

Reference

[1] Avere Sarvesh, et al. “IOT based power grid monitoring & control system,” VIVA-Tech International Journal for Research and Innovation, vol. 1.4, pp. 1-6, 2021.

[2] J. U. Duncombe, “Infrared navigation—Part I: An assessment of feasibility,” IEEE Trans. Electron Devices, vol. ED-11, pp. 34-39, Jan. 1959.

[3] Karthick, T., and K. Chandrasekaran, “Design of IoT based smart compact energy meter for monitoring and controlling the usage of energy and power quality issues with demand side management for a commercial building,” Sustainable Energy, Grids and Networks, vol. 26, 100454, 2021.

[4] Mehmood, M. Yasir, et al., “Edge computing for IoT-enabled smart grid,” Security and Communication Networks, 2021.

[5] Ahmed, S. Rafeek, et al., “Smart IOT based Short Term Forecasting of Power Generation Systems and Quality Improvement Using Resilient Back Propagation Neural Network,”revista geintec-gestao inovacao e tecnologias, vol. 11.3, pp. 1200-1211, 2021.

[6] Avancini Danielly B., et al. “A new IoT-based smart energy meter for smart grids,” International Journal of Energy Research, vol. 45.1, pp. 189-202, 2021.

[7] Hazarika K., Gauri Katiyar, and Noorul Islam. “IOT Based Transformer Health Monitoring System: A Survey,” 2021 International Conference on Advance Computing and Innovative Technologies in Engineering (ICACITE). IEEE, 2021.

[8] Onawola Hassan Jimoh, et al. “A Conceptual Model for Mitigating Security Vulnerabilities in IoT-Based Smart Grid Electric Energy Distribution Systems,”

International Journal of Engineering Research in Africa, vol. 55. Trans Tech Publications Ltd, 2021.

[9] Ahmad Shamim, et al. “A low SAR in-body antenna for wireless monitoring purpose of pacemaker system,” 2019 4th International Conference on Electrical Information and Communication Technology (EICT). IEEE, 2019.

[10] Al Rakib, Md Abdullah, et al. “Design and Simulation-Based Parametric Studies of a Compact Ultra-Wide Band Antenna for Wireless Capsule Endoscopy System at Inside Body Environment,” International Journal of Electrical and Electronic Engineering & Telecommunications, 2021.

[11] Al Rakib, Md Abdullah, et al. “Dry and Wet Waste Segregation and Management System,” European Journal of Engineering and Technology Research, vol. 6.5, pp. 129-133, 2021.

[12] Al Rakib, Md Abdullah, et al. “Energy Harvesting Technology from Human Motion,” 2020 2nd International Conference on Advanced Information and Communication Technology (ICAICT). IEEE, 2020.

[13] Mahamud, Md Sadad, et al. “Mouchak-An IoT Basted Smart Beekeeping System Using MQTT,” 2019 4th International Conference on Robotics and Automation Engineering (ICRAE). IEEE, 2019.

Appendix

```
// adafruit Faysal_Mahmud 12345689

#include <LiquidCrystal_I2C.h>
LiquidCrystal_I2C lcd(0x27 ,20,4);

#include <ESP8266WiFi.h>
#include "Adafruit_MQTT.h"
#include "Adafruit_MQTT_Client.h"
#define WLAN_SSID       "abcde"
#define WLAN_PASS       "123456789"
#define AIO_SERVER      "io.adafruit.com"
#define AIO_SERVERPORT  1883
#define AIO_USERNAME    "Faysal_Mahmud"
#define AIO_KEY         "aio_QltA44MVlLygWaA2mkDIhk5DZ4V5"
// Create an ESP8266 WiFiClient class to connect to the MQTT server.
WiFiClient client;
// or... use WiFiFlientSecure for SSL
//WiFiClientSecure client;
// Setup the MQTT client class by passing in the WiFi client and MQTT server and login
details.
Adafruit_MQTT_Client mqtt(&client, AIO_SERVER, AIO_SERVERPORT,
AIO_USERNAME, AIO_KEY);
Adafruit_MQTT_Publish photocell = Adafruit_MQTT_Publish(&mqtt,
AIO_USERNAME "/feeds/Current");
Adafruit_MQTT_Publish photocell1 = Adafruit_MQTT_Publish(&mqtt,
AIO_USERNAME "/feeds/AC Watt");
Adafruit_MQTT_Publish photocell2 = Adafruit_MQTT_Publish(&mqtt,
AIO_USERNAME "/feeds/volt");
```

```
const int sensorIn = A0;
int mVperAmp = 66; //185mV for 5Amp module , 100mV for 10A , 66mv for 20 & 30 Amp module
double Voltage = 0;
double Vp = 0;
double Vrms = 0;
double Irms = 0;
double watt = 0;
double data =0.00;
int voltage = 220;
int load_share  = D5;

int voltage1;
// Bug workaround for Arduino 1.6.6, it seems to need a function declaration
// for some reason (only affects ESP8266, likely an arduino-builder bug).
void MQTT_connect();
void setup() {
  Serial.begin(9600);
 pinMode(D5,OUTPUT);
  WiFi.begin(WLAN_SSID, WLAN_PASS);
  while (WiFi.status() != WL_CONNECTED)
  {
    Serial.println("connecting....");
    delay(1000);
  }
  Serial.println("Connected");
}
void loop() {
  MQTT_connect();
  Voltage = getVPP();
    Vrms = (Voltage / 5.0) * 0.707; // sq root
```

```
  Irms = ((Vrms * 1000) / mVperAmp) ;
  watt = Irms*voltage;
  Serial.println(" Amps: ");
  Serial.print(Irms);
    Serial.println(" watt: ");
Serial.print(watt);

  if(watt < 86){
digitalWrite(D5,HIGH);
  }
    if(watt > 94){
digitalWrite(D5,LOW);
delay(5000);
digitalWrite(D5,HIGH);
  }

  if (! photocell2.publish(voltage1))
  {
    Serial.println("Failed");
  }
  else
  {
    Serial.println("OK!");
  }

delay(2000);

  if(Irms >0.03){
  if (! photocell.publish(Irms))
  {
    Serial.println("Failed");
```

```
  }
  else
  {
    Serial.println("OK!");
  }
}
delay(2000);

  if (! photocell1.publish(watt))
  {
    Serial.println("Failed");
  }
  else
  {
    Serial.println("OK!");
  }



delay(2000);

}
double getVPP()
{
  float result;
  int readValue; //value read from the sensor
  int maxValue = 0; // store max value here
  int minValue = 1024; // store min value here
  uint32_t start_time = millis();
  while ((millis() - start_time) < 1000) //sample for 1 Sec
  {
    readValue = analogRead(sensorIn);
```

```
    // see if you have a new maxValue
    if (readValue > maxValue)
    {
      /*record the menimum sensor value*/
      maxValue = readValue;
    }
    if (readValue < minValue)
    {
      /*record the menimum sensor value*/
      minValue = readValue;
    }
  }
  // Subtract min from max
  result = ((maxValue - minValue) * 5) / 1024.0;
  return result;
}
void MQTT_connect() {
  int8_t ret;
  // Stop if already connected.
  if (mqtt.connected()) {
    Serial.println("MQTT_Already_connected");
    return;
  }
  Serial.print("Connecting to MQTT... ");
  uint8_t retries = 3;
  while ((ret = mqtt.connect()) != 0) { // connect will return 0 for connected
    Serial.println(mqtt.connectErrorString(ret));
    Serial.println("Retrying MQTT connection in 5 seconds...");
    mqtt.disconnect();
    delay(5000);  // wait 5 seconds
    retries--;
```

```
    if (retries == 0) {
      // basically die and wait for WDT to reset me
      while (1);
    }
  }
  Serial.println("MQTT Connected!");
}

void voltt(){



}
```