*A Project report on*

“SMART ENERGY METER”

*Submitted in partial fulfilment of the requirement for the award of the degree of*

**BACHELOR OF TECHNOLOGY**

In

ELECTRONICS & COMMUNICATION ENGINEERING

BY

M.Tejaswini 20BC5A0401

P.Bhagya Rekha 20BC5A0402

A.Sai Sunil 19BC1A0401

A.Veeranjaneya Reddy 19BC1A0402

C.Chaitanya Kumar Reddy 19BC1A0404

Under the esteemed guidance

Sir.V.Kishore, M.Tech, (PhD)

Assistant Professor



Submitted to

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

KANDULA OBUL REDDY MEMORIAL COLLEGE OF ENGINEERING

(Affiliated To J.N.T. University Anantapur)

KADAPA, Y.S.R District

ANDHRA PRADESH

2022-2023

**Acknowledgement**

An endeavour of a long period can be successful only with the advice of many well wishers. We take this opportunity to express our deep gratitude and appreciation to all those who encouraged me for successful completion of the project work.

Our heartfelt thanks to our Guide, **Sri V.Kishore,** ***Assistant Professor*** in E.C.E. Department, Kandula Obul Reddy Memorial College of Engineering, Kadapa for her valuable guidance and suggestions in analyzing and testing throughout the period, till the end of project work completion.

Our special thanks to **Smt** C. Chandrakala**, *Head of Electronics and******Communication Engineering,***Kandula Obul Reddy Memorial College of Engineering, Kadapa, for his timely suggestions and help in spite of his busy schedule.

We wish to express our sincere gratitude to **Dr. A.Venkata Sesha Giridhar , *Principal*** of Kandula Obul Reddy Memorial College of Engineering, Kadapa, for his consistent help and encouragement to complete the project work.

We are very much thankful to **Sri. S. Venkata Siva Reddy, *Honourable Chairman*** *and* **Smt. S. Likhitha Reddy, *Correspondent*** of Kandula Obul Reddy Memorial College of Engineering, Kadapa, Kadapa for his help in providing good facilities in our college.

We wish to convey our gratitude and express our sincere thanks to all Project Committee members for their support and co-operation rendered for successful submission of our Project Work.

Finally, we would like to express our sincere thanks to *faculty members* of E.C.E department*, lab technicians*, and friends*, one and all* who has helped us complete the Project Work successfully.

As a gesture of respect for our *family members* and for the support we received from them, we dedicate this work to them.

**ABSTRACT**

Energy meter is a measuring instrument to calculate the amount of electric energy consumed by an electrically powered device. The energy metering which measure line voltage, current, and calculating active power can be used to increase usage efficiency of electricity. This paper present a system which provide real time energy meter reading using ACS712 as current sensor, ZMPT101B as voltage sensor, arduino and an IoT server. The energy meter can display consumed energy in kWh and electricity bills to be paid. system's user interface is in Android. the error in current measurement is less than 2 % and error for peak to peak voltage measurement is 4 %. A notification for the amount of electricity energy usage or electricity cost can be generated for user to be display.

**TABLE OF CONTENTS**

**CHAPTER NO CONTENTS PAGE NO**

**CHAPTER 1 INTRODUCTION**

1.1 Introduction Embedded Systems

1.2 Types of Embedded Systems

1.2.1 Stand Alone Embedded Systems

1.2.2 Real Time Embedded System

1.2.3 Networked Embedded System

1.2.4 Small Scale Embedded System

1.2.6 Medium Scale Embedded System

1.2.7 Sophisticated Embedded System

1.3 Overview of Embedded System

Architecture

1.3.1 Central Processing Unit

1.3.2 Memory

1.3.3 Input Devices

1.3.4 Output Devices

1.4 Memory Architecture

1.4.1 Harvard Architecture

1.4.2 Modern uses of the Harvard Architecture

1.4.3 Von-Neuman Architecture

1.4.4 Basic Difference between Harvard &

Von-Neuman Architecture

1.5 Features of Embedded System

**CHAPTER N0 CONTENTS PAGE N0**

1.6 Design process

1.6.1 Specifications

1.6.2 System-Synthesis

1.6.3 Implementation -Synthesis

1.6.4 Prototyping

1.7 Applications of Embedded Systems

1.7.1 Consumer Applications

1.7.2 Computer Networking

1.7.3 Telecommunications

1.7.4 Wireless Technologies

1.7.5 Security

**CHAPTER 2 LITERATURE SURVEY**

**CHAPTER 3 BLOCK DIAGRAM AND HARDWARE DISCRIPTION**

3.1 Block Diagram

3.2 Power Supply

3.2.1 Transformer

3.2.2 Rectifier

3.3 Filter Capacitor

3.4 Voltage Regulator

3.4.1 Circuit Diagram

3.5 Energy Meter

3.5.1 Introduction

**CHAPTER NO CONTENTS PAGE NO**

3.5.2 Basic Operation of a relay

3.5.3 Neutral Relays

3.5.4 Biased Relays

3.5.5 Polarized Relays

3.5.6 Magnetic stick Relays

3.5.7 Slow Release Relays

3.5.8 Relays for AC

3.5.9 Applications

3.6 LDR

3.6.1 Working

3.6.2 Applications

3.6.3 Measure Light Intensity using LDR

3.7 ESP 8266

3.7.1 Features

3.7.2 Description

3.7.3 Specification of ESP 8266

3.8 LCD Module

3.9 Commands used in LCD

**CHAPTER 4 CONTROLLER**

4.1 Arduino

4.2 History

4.3 Hardware

4.4 Software Development

**CHAPTER NO CONTENTS PAGE NO**

4.4.1 Applications

4.4.2 Technical Specs

4.5 Programming

4.6 Warning

4.6.1 Difference with other boards

4.7 Power

4.8 Memory

4.8.1 Input and Output

4.8.2 Communication

4.8.3 Automatic Reset

4.8.4 Revision

**CHAPTER 5 SOFTWARE INTRODUCTION 51-57**

**5.1**

**5.2**

**5.3**

**CHAPTER 6 CONCLUSION 58**

**6.1 Conclusion**

**BIBILOGRAPH 59**

#### LIST OF FIGURES

|  |  |  |  |
| --- | --- | --- | --- |
| **S.NO** | **Fig.NO** | **Figure Name** | **Page. No** |
| 1 | Figure 1.1 | Block diagram of Embedded system |  |
| 2 | Figure 1.2 | Types of Embedded system |  |
| 3 | Figure 1.3 | Embedded system Architecture |  |
| 4 | Figure 1.4.1 | Harvard Architecture |  |
| 5 | Figure 1.4.3 | Schematic of the Von -Neuman Architecture |  |
| 6 | Figure 3.1 | Block Diagram |  |
| 7 | Figure 3.2 | Block diagram of a fixed regulated power supply |  |
| 8 | Figure 3.2.1 | Transformer |  |
| 9 | Figure 3.1.2 | Half wave rectifier |  |
| 10 | Figure 3.1.3 | Full wave rectifier |  |

|  |  |  |  |
| --- | --- | --- | --- |
| 11 | Figure 3.1.4 | Bridge rectifier |  |
| 12 | Figure 3.1.5 | Half cycle output |  |
| 13 | Figure 3.4 | Voltage regulator |  |
| 14 | Figure 3.4.1 | Circuit diagram of power supply |  |
| 15 | Fig 3.5 | Energy meter |  |
| 16 | Figure 3.5.1 | Relay |  |
| 17 | Figure 3.5.2 | Relay circuit |  |
| 18 | Figure 3.6 | LDR |  |
| 19 | Figure 3.7.2 | ESP 8266 |  |
| 20 | Figure 3.8 | 16 x 12 Line alphanumeric LCD display |  |
| 21 | Figure 4.1 | Arduino |  |
| 23 | Figure 4.3 | Arduino hardware |  |

#### 

**CHAPTER 1**

**1.1 INTRODUCTION TO THE PROJECT**

An embedded system is a system which is going to do a predefined specified task is the embedded system and is even defined as combination of both software and hardware. A general-purpose definition of embedded systems is that they are devices used to control, monitor or assist the operation of equipment, machinery or plant. "Embedded" reflects the fact that they are an integral part of the system. At the other extreme a general-purpose computer may be used to control the operation of a large complex processing plant, and its presence will be obvious. All embedded systems are including computers or microprocessors. Some of these computers are however very simple systems as compared with a personal computer.

The very simplest embedded systems are capable of performing only a single function or set of functions to meet a single predetermined purpose. In more complex systems an application program that enables the embedded system to be used for a particular purpose in a specific application determines the functioning of the embedded system. The ability to have programs means that the same embedded system can be used for a variety of different purposes. In some cases a microprocessor may be designed in such a way that application software for a particular purpose can be added to the basic software in a second process, after which it is not possible to make further changes. The applications software on such processors is sometimes referred to as firmware.

**1.2 EXISTING SYSTEM**

Existing meter reading techniques in India are analysed and conducted an extensive study on different energy measuring instruments available now. In existing system either an Electronic energy meter or an electro-mechanical meter is fixed in the premise for measuring the usage. The meters currently in use are only capable of recording kWh units.The kWh units used then still have to be recorded by meter readers monthly, on foot. The recorded data need to be processed by a meter reading company. For processing the meter reading, company needs to firstly link each recorded power usage datum to an account holder and then determine the amount owed by means of the specific tariff in use. Wireless electric power management and control system for short distance is developed using Zigbee technique. For this IEEE 802.15.4 standard protocol is used as a Zigbee standard, microcontroller is used to manage energy data and Zigbee to enable communication between the energy meter and data centres. The secure mobile agent concept was presented in, instead of one person for one meter according to geographical area energy meters can be organized. For one location energy meters a security manager can do his work. The major disadvantage of a post-paid system is that there is no control of usage from the consumer’s side. There is a lot of wastage of power. Since the supply of power is limited, as a responsible citizen, there is a need to use electricity in a improved and efficient way. There are clear domino effect from many countries everywhere a prepaid system has reduced the usage (wastage) by a great quantity. Additional advantage of the prepaid system is that the human errors made reading meters and processing bills can be reduced to a great amount. Wireless meter can be used in residential apartments and especially in industrial consumers where bulk energy is consumed. Advance in technology have made exchange of information in very high-speed, protected and truthful. Advance in wireless technology caused rapid change in field of telecommunication Communicatio system like internet and GSM are available in India. This paper presents a system which can act as either prepaid or post-paid meter .

**1.3 PROPOSED SYSTEM**

The current system of electricity billing is error level and also time consuming. Errors introduced at every stage are due to electro-mechanical meters, human errors while noting down the meter reading, errors while processing the paid bills and the due bills. Smart energy meter is a technique which can reduce the problems associated with billing and also reduces the deployment of manpower for taking meter readings. It has many advantages from both suppliers as well as consumer’s point. This paper is also intended to present an overview of prepaid energy meter, which can control the usage of electricity on consumer side to avoid wastage of power and is a concept to minimize the electricity theft. As the billing process is done automatically in the proposed system it mainly reduces the manpower.

**1.4 TOOLS REQUIRED**

**HARDWARE TOOLS**

* REGULATED POWER SUPPLY
* LCD
* ENERGY METER
* LDR
* MOBILE NOTIFICATION

**SOFTWARE TOOLS**

* ARDUNIO IDE
* PROTEAUS
* PROGRAMMING LANGUAGE C, C++

**1.5 ADVANTAGES**

* Visibility
* Accurate Billing
* Saving Time and Money
* Unique Insight To Our Consumption
* Digitalization
* Efficient Power Management

**1.6 DISADVANGTAGES**

* Switching energy suppliers become difficult.
* Poor single prevents the smart meter from working.
* Smart meter stops sending readings.
* Undersanding your smart meter monitor.
* Smart meters pose a risk to security.
* Existing meters are hard to access.

**CHAPTER 2**

**LITERATURE SURVEY**

Das v.v., “Wireless Communication system for Energy Meter Reading”, . Energy meter reading is a tedious and an expensive affair. The meter reading has to go and take the reading manually to issues the bill, which will later be entered in the software to automate the billing and payment system. It would have reduced the laborious task and financial wastage if can automate the manual meter reading process. This paper proposes a new network communication system for energy meter reading by integrating communication technology and software system along with the existing meters. A wireless or wired communication system will be integrated with electronic energy meter to have remote access over the usage of electricity. Even though there are two different modules, energy meter delivers the reading details as on when it demands by the communication system. The communication system further connected with the electricity regional/sub-regional office, which will rather act as a base station. Instead of creating a separate communication system and backbone.

“Design and Implementation of a Zig Bee-Based Wireless Automatic Meter Reading System” (2012). With the rapid development of automation and measuring techniques, automatic recording of the data in the meter reading instrument has gradually become the target of people whose working, living, and home conditions are of increasingly high level of intelligence. Meanwhile, utilities also hope that the development of new technologies to solve the problems they encountered in the practical work about cumbersome meter reading and no reliable protection of accuracy and real time; and enable both user friendly and improving public sector efficiency and management level. Existing wire-line meter reading system has a large number of risks. Wires are more complex, detrimental to adjustment and maintenance of the system. The long- term indoor and outdoor installation easily leads to aging, resulting in a risk of short circuit and breakage. For these reasons, it has become the industry very unresolved problem to design a remote meter reading system, with Long-term reliance and convenient installation & maintenance, which not only read data automatically but also monitor operation status. With the development of wireless communication technology, in recent years there comes requirement for low cost equipment of wireless networking technology, called Zigbee. It is a short range, low-complexity, low cost, low power consumption, low data rate two-way wireless communication technology with high network capacity, short time delay, safety and reliance. Its main application areas include industrial controls, consumer electronics, car automation, agricultural automation, and medical equipment control. Till now, the Zigbee Alliance has reached over 150 members of famous companies in the world including IBM, Ember, Mitsubishi, Motorola, and Philips, etc.. Many semiconductor companies are targeting the ZigBee market. Since the standards were launched not long ago, chips in line with protocol have been available of multi- chip solution and single- chip solution. It can be expected that ZigBee will have comprehensive applications in the field of automation. The main methods of metering at home and abroad are: manual meter reading, IC Card prepaid meter, wire-line and wireless meter reading system. Manual meter reading has been for decades, but with the implementation of one home one meter, drawbacks of this method of reading are more and more, like difficult entrance to home, low efficiency of fee settlement, etc. Its main application areas include industrial controls, consumer electronics, car automation, agricultural automation, and medical equipment control.

“Automatic Electric Meter Reading System: A Cost-Feasible Alternative Approach in Meter Reading For Bangladesh Perspective Using Low-Cost Digital Wattmeter and Wimax Technology” (2013). Energy meter reading is a monotonous and an expensive task. Now the meter reader people goes to each meter and take the meter reading manually to issue the bill which will later be entered in the billing software for billing and payment automation. If the manual meter reading and bill data entry process can be automated then it would reduce the laborious task and financial wastage. “Automatic Electric Meter Reading (AMR) System” is a metering system that is to be used for data collecting from the meter and processing the collected data for billing and other decision purposes. In this paper we have proposed an automatic meter reading system which is low cost, high performance, highest data rate, highest coverage area and most appropriate for Bangladesh perspective. In this AMR system there are four basic units. They are reading unit, communication unit, data receiving and processing unit and billing system. For reading unit we identified the disk rotation of the energy meter and stored the data in microcontroller. So it is not required to change the current analogy energy meter. An external module will be added with the current energy meter. In the communication unit, Wimax transceiver was used for wireless communication between meter end and the server end because of its wide coverage area. In the data receiving and processing unit meter reading will be collected from the transceiver which is controlled by another microcontroller. There will be a computer application that will take the data from the microcontroller. This will also help to avoid any tampering or break down of energy meter. There are various AMR system exists all over the world. Those systems were analyzed and we found they are not feasible for Bangladesh. Our proposed system is completely new and is appropriate for Bangladesh perspective.

**CHAPTER 3**

**DOMAIN OF THE PROJECT**

**3.1. BLOCK DIAGRAM:**

ARDUINO

+

WIFI

LCD

POWER

ENERGY METER

MOBILE NOTIFICATION

LDR

Figure 3.1 : Block Diagram

**3.2 Power Supply**

All digital circuits require regulated power supply. In this article we are going to learn how to get a regulated positive supply from the mains supply.

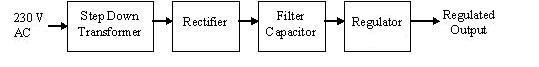


Figure 3.2 shows the basic block diagram of a fixed regulated power supply.

Let us go through each block.

**3.2.1 Transformer**

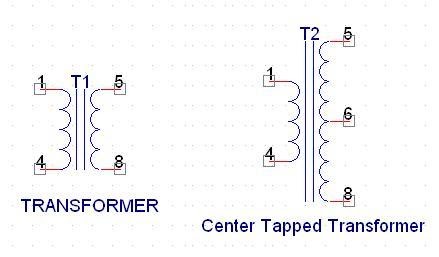


Figure 3.2.1: Transformer

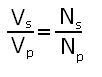
A transformer consists of two coils also called as “*WINDINGS*” namely *PRIMARY & SECONDARY.*They are linked together through inductively coupled electrical conductors also called as CORE. A changing current in the primary causes a change in the Magnetic Field in the core & this in turn induces an alternating voltage in the secondary coil. If load is applied to the secondary then an alternating current will flow through the load. If we consider an ideal condition then all the energy from the primary circuit will be transferred to the secondary circuit through the magnetic field.

So Image

Image

The secondary voltage of the transformer depends on the number of turns in the Primary as well as

in the secondary.



**Rectifier**

A rectifier is a device that converts an AC signal into DC signal. For rectification purpose we use a diode, a diode is a device that allows current to pass only in one direction i.e. when the anode of the diode is positive with respect to the cathode also called as forward biased condition & blocks current in the reversed biased condition.

Rectifier can be classified as follows:

**1)      Half Wave rectifier.**

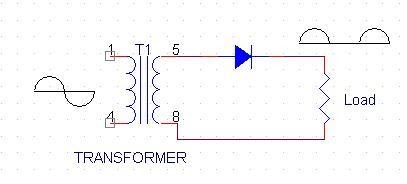


Figure 3.1.2 : Half wave rectifier

This is the simplest type of rectifier as you can see in the diagram a half wave rectifier consists of only one diode. When an AC signal is applied to it during the positive half cycle the diode is forward biased & current flows through it. But during the negative half cycle diode is reverse biased & no current flows through it. Since only one half of the input reaches the output, it is very inefficient to be used in power supplies.

**2)      Full wave rectifier.**

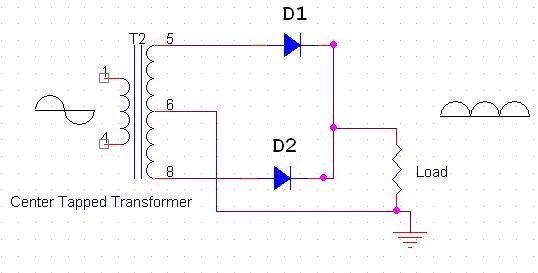


Figure 3.1.3: Full wave rectifier

Half wave rectifier is quite simple but it is very inefficient, for greater efficiency we would like to use both the half cycles of the AC signal. This can be achieved by using a center tapped transformer i.e. we would have to double the size of secondary winding & provide connection to the center. So during the positive half cycle diode D1 conducts & D2 is in reverse biased condition. During the negative half cycle diode D2 conducts & D1 is reverse biased. Thus we get both the half cycles across the load. One of the disadvantages of Full Wave Rectifier design is the necessity of using a center tapped transformer, thus increasing the size & cost of the circuit. This can be avoided by using the Full Wave Bridge Rectifier.

**3)      Bridge Rectifier**

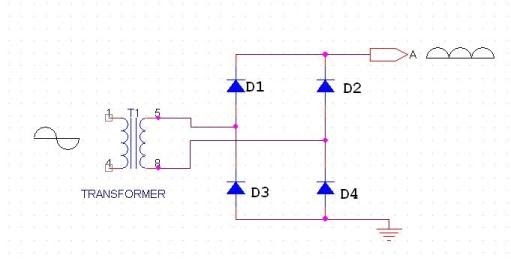
**.**

Figure 3.1.4 : Bridge rectifier

As the name suggests it converts the full wave i.e. both the positive & the negative half cycle into DC thus it is much more efficient than Half Wave Rectifier & that too without using a center tapped transformer thus much more cost effective than Full Wave Rectifier. Full Bridge Wave Rectifier consists of four diodes namely D1, D2, D3 and D4. During the positive half cycle diodes D1 & D4 conduct whereas in the negative half cycle diodes D2 & D3 conduct thus the diodes keep switching the transformer connections so we get positive half cycles in the output.

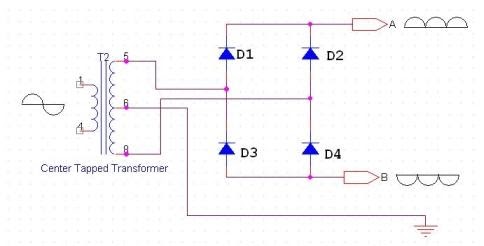


Figure 3.1.5 : Half cycle output

If we use a center tapped transformer for a bridge rectifier we can get both positive & negative half cycles which can thus be used for generating fixed positive & fixed negative voltages.

**3.3 Filter Capacitor**

Even though half wave & full wave rectifier give DC output, none of them provides a constant output voltage. For this we require to smoothen the waveform received from the rectifier. This can be done by using a capacitor at the output of the rectifier this capacitor is also called as “FILTER CAPACITOR” or “SMOOTHING CAPACITOR” or “RESERVOIR CAPACITOR”. Even after using this capacitor a small amount of ripple will remain. We place the Filter Capacitor at the output of the rectifier the capacitor will charge to the peak voltage during each half cycle then will discharge its stored energy slowly through the load while the rectified voltage drops to zero, thus trying to keep the voltage as constant as possible.

If we go on increasing the value of the filter capacitor then the Ripple will decrease. But then the costing will increase. The value of the Filter capacitor depends on the current consumed by the circuit, the frequency of the waveform & the accepted ripple.



Where,

Vr = accepted ripple voltage.( should not be more than 10% of  the voltage)

I = current consumed by the circuit in Amperes.

F = frequency of the waveform. A half wave rectifier has only one peak in one cycle so F=25hz

Where as a full wave rectifier has Two peaks in one cycle so F=100hz.

**3.4 Voltage Regulator**

A Voltage regulator is a device which converts varying input voltage into a constant regulated output voltage. Voltage regulator can be of two types

1. Linear Voltage Regulator Also called as Resistive Voltage regulator because they dissipate the excessive voltage resistively as heat.
2. Switching Regulators They regulate the output voltage by switching the Current ON/OFF very rapidly. Since their output is either ON or OFF it dissipates very low power thus achieving higher efficiency as compared to linear voltage regulators. But they are more complex & generate high noise due to their switching action. For low level of output power switching regulators tend to be costly but for higher output wattage they are much cheaper than linear regulators.

The most commonly available Linear Positive Voltage Regulators are the 78XX series where the XX indicates the output voltage. And 79XX series is for Negative Voltage Regulators.

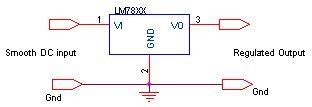
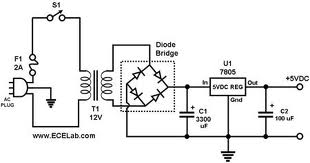


Fig 3.4 : Voltage regulator

After filtering the rectifier output the signal is given to a voltage regulator. The maximum input voltage that can be applied at the input is 35V.Normally there is a 2-3 Volts drop across the regulator so the input voltage should be at least 2-3 Volts higher than the output voltage. If the input voltage gets below the Vmin of the regulator due to the ripple voltage or due to any other reason the voltage regulator will not be able to produce the correct regulated voltage.

**3.4.1 Circuit diagram:**



**Figure 3.4.1: Circuit Diagram of power supply**

**IC 7805:**

7805 is an integrated three-terminal positive fixed linear voltage regulator. It supports an input voltage of 10 volts to 35 volts and output voltage of 5 volts. It has a current rating of 1 amp although lower current models are available. Its output voltage is fixed at 5.0V. The 7805 also has a built-in current limiter as a safety feature. 7805 is manufactured by many companies, including National Semiconductors and Fairchild Semiconductors. The 7805 will automatically reduce output current if it gets too hot.The last two digits represent the voltage; for instance, the 7812 is a 12-volt regulator. The 78xx series of regulators is designed to work in complement with the 79xx series of negative voltage regulators in systems that provide both positive and negative regulated voltages, since the 78xx series can't regulate negative voltages in such a system. The 7805 & 78 is one of the most common and well-known of the 78xx series regulators, as it's small component count and medium-power regulated 5V make it useful for powering TTL devices.

**3.5 Energy Meter**

Electric meter or watt-hour meter, a device that measure the quantity of electric energy. One kilowatt hour is the quantity of electric energy needed to supply of electricity for a span of one hour. An electrical power business uses electric meters to quantify the number of electricity consumed by each. An electricity meter, electric meter, electrical meter, or energy meter is a device that measures the amount of electric energy consumed by a residence, a business, or an electrically powered device. Electric utilities use electric meters installed at customers' premises to measure electric energy delivered to their customers for billing purposes. They are typically calibrated in billing units, the most common one being the kilowatt hour. They are usually read once each billing period. When energy savings during certain periods are desired, some meters may measure demand, the maximum use of power in some interval. "Time of day" metering allows electric rates to be changed during a day, to record usage during peak high-cost periods and off-peak, lower-cost, periods. Also, in some areas meters have relays for demand response load shedding during peak load periods. One kilowatt hour is the quantity of electric energy needed to supply of electricity for a span of one hour. An electrical power business uses electric meters to quantify the number of electricity consumed by each. An electricity meter, electric meter, electrical meter, or energy meter is a device that measures the amount of electric energy consumed by a residence, a business, or an electrically powered device. An electrical power business uses electric meters to quantify the number of electricity consumed by each. An electricity meter, electric meter, electrical meter, or energy meter is a device that measures the amount of electric energy consumed by a residence, a business, or an electrically powered device.



**Figure 3.5 : Energy meter**

One kilowatt hour is the quantity of electric energy needed to supply of electricity for a span of one hour. An electrical power business uses electric meters to quantify the number of electricity consumed by each. An electricity meter, electric meter, electrical meter, or energy meter is a device that measures the amount of electric energy consumed by a residence, a business, or an electrically powered device. An electrical power business uses electric meters to quantify the number of electricity consumed by each. An electricity meter, electric meter, electrical meter, or energy meter is a device that measures the amount of electric energy.

**3.5.1Introduction**

A relay is an electrical switch that opens and closes under the control of another electrical circuit. In the original form, the switch is operated by an electromagnet to open or close one or many sets of contacts. A relay is able to control an output circuit of higher power than the input circuit, it can be considered to be, in a broad sense, a form of an electrical amplifier.

Relays are usuallly SPDT (single pole double through switch)or DPDT (double pole double through switch) but they can have many more sets of switch contacts, for example relays with 4 sets of changeover contacts are readily available.

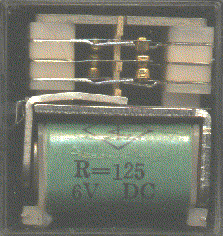
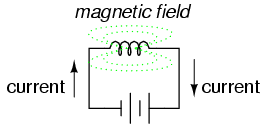


Figure 3.5: Relay

**3.5.2 Basic operation of a relay**

An electric current through a conductor will produce a magnetic field at right angles to the direction of electron flow. If that conductor is wrapped into a coil shape, the magnetic field produced will be oriented along the length of the coil. The greater the current, the greater the strength of the magnetic field, all other factors being equal.



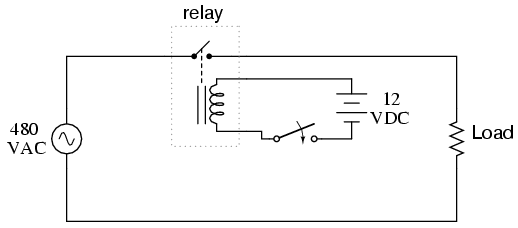


Figure 3.5.2 : Relay circuit

Inductors react against changes in current because of the energy stored in this magnetic field. When we construct a transformer from two inductor coils around a common iron core, we use this field to transfer energy from one coil to the other. However, there are simpler and more direct uses for electromagnetic fields than the applications we've seen with inductors and transformers. The magnetic field produced by a coil of current-carrying wire can be used to exert a mechanical force on any magnetic object, just as we can use a permanent magnet to attract magnetic objects, except that this magnet (formed by the coil) can be turned on or off by switching the current on or off through the coil.

If we place a magnetic object near such a coil for the purpose of making that object move when we energize the coil with electric current, we have what is called a *solenoid*. The movable magnetic object is called an *armature*, and most armatures can be moved with either direct current (DC) or alternating current (AC) energizing the coil. The polarity of the magnetic field is irrelevant for the purpose of attracting an iron armature. Solenoids can be used to electrically open door latches, open or shut valves, move robotic limbs, and even actuate electric switch mechanisms and is used to actuate a set of switch contact Relays can be categorized according to the magnetic system and operation:

#### 3.5.3 Neutral Relays:

This is the most elementary type of relay. The neutral relays have a magnetic coil, which operates the relay at a specified current, regardless of the polarity of the voltage applied.

#### 3.5.4 Biased Relays:

Biased relays have a permanent magnet above the armature. The relay operates if the current through the coil winding establishes a magneto-motive force that opposes the flux by the permanent magnet. If the fluxes are in the same direction, the relay will not operate, even for a greater current through the coil.

#### 3.5.5 Polarized Relays:

Like the biased relays, the polarized relays operate only when the current through the coil in one direction. But there the principle is different. The relay coil has a diode connected in series with it. This blocks the current in the reverse direction.

The major difference between biased relays and polarized relays is that the former allows the current to pass through in the reverse direction, but does the not operate the relay and the later blocks the current in reverse direction.  You can imagine how critical these properties when relays are connected in series to form logic circuits.

#### 3.5.6 Magnetic Stick Relays or Perm polarized Relays:

These relays have a magnetic circuit with high permanence.  Two coils, one to operate (pick up) and one to release (drop) are present.  The relay is activated by a current in the operate coil. On the interruption of the current the armature remains in picked up position by the residual magnetism. The relay is released by a current through the release coil.

#### 3.5.7 Slow Release Relays:

These relays have a capacitor connected in parallel to their coil. When the operating current is interrupted the release of relay is delayed by the stored charge in the capacitor. The relay releases as the capacitor discharges through the coil.

#### 3.5.8 Relays for AC

These are neutral relays and picked up for a.c. current through their coil. These are very fast in action and used on power circuits of the point motors, where high current flows through the contacts. A normal relay would be slow and make sparks which in turn may weld the contacts together.

All relays have two operating values (voltages), one pick-up and the other other drop away. The pick-up value is higher than the drop away value.

**3.5.9 Applications:**

* To control a high-voltage circuit with a low-voltage signal, as in some types of modems or audio amplifiers,
* To control a high-current circuit with a low-current signal, as in the starter solenoid of an automobile,
* To detect and isolate faults on transmission and distribution lines by opening and closing circuit breakers (protection relays),
* To isolate the controlling circuit from the controlled circuit when the two are at different potentials, for example when controlling a mains-powered device from a low-voltage switch. The latter is often applied to control office lighting as the low voltage wires are easily installed in partitions, which may be often moved as needs change. They may also be controlled by room occupancy detectors in an effort to conserve energy,
* To perform logic functions. For example, the boolean AND function is realised by connecting NO relay contacts in series, the OR function by connecting NO contacts in parallel. The change-over or Form C contacts perform the XOR (exclusive or) function. Similar functions for NAND and NOR are accomplished using NC contacts. The Ladder programming language is often used for designing relay logic networks.
  + Early computing. Before vacuum tubes and transistors, relays were used as logical elements in digital computers. See ARRA (computer), Harvard Mark II, Zuse Z2, and Zuse Z3.
  + Safety-critical logic. Because relays are much more resistant than semiconductors to nuclear radiation, they are widely used in safety-critical logic, such as the control panels of radioactive waste-handling machinery.
* To perform time delay functions. Relays can be modified to delay opening or delay closing a set of contacts. A very short (a fraction of a second) delay would use a copper disk between the armature and moving blade assembly. Current flowing in the disk maintains magnetic field for a short time, lengthening release time. For a slightly longer (up to a minute) delay, a dashpot is used. A dashpot is a piston filled with fluid that is allowed to escape slowly. The time period can be varied by increasing or decreasing the flow rate. For longer time periods, a mechanical clockwork timer is installed

**3.6 LDR**

**3.6.1Working**

A photo resistor or Light Dependent Resistor or CdS Cell is a [resistor](http://en.wikipedia.org/wiki/Resistor) whose [resistance](http://en.wikipedia.org/wiki/Electrical_resistance) decreases with increasing incident light intensity. It can also be referred to as a photoconductor. A photo resistor is made of a high resistance [semiconductor](http://en.wikipedia.org/wiki/Semiconductor). If light falling on the device is of high enough [frequency](http://en.wikipedia.org/wiki/Frequency), [photons](http://en.wikipedia.org/wiki/Photon) absorbed by the semiconductor give bound [electrons](http://en.wikipedia.org/wiki/Electron) enough energy to jump into the [conduction band](http://en.wikipedia.org/wiki/Conduction_band). The resulting free electron (and its [hole](http://en.wikipedia.org/wiki/Electron_hole) partner) conduct electricity, thereby lowering [resistance](http://en.wikipedia.org/wiki/Electrical_resistance).A photoelectric device can be either intrinsic or extrinsic. An intrinsic semiconductor has its own charge carriers and is not an efficient semiconductor, e.g. silicon. In intrinsic devices the only available electrons are in the [valence band](http://en.wikipedia.org/wiki/Valence_band), and hence the photon must have enough energy to excite the electron across the entire [band gap](http://en.wikipedia.org/wiki/Bandgap). Extrinsic devices have impurities, also called [dopants](http://en.wikipedia.org/wiki/Dopants), added whose ground state energy is closer to the conduction band; since the electrons don't have as far to jump, lower energy photons (i.e., longer wavelengths and lower frequencies) are sufficient to trigger the device. If a sample of silicon has some of its atoms replaced by phosphorus atoms (impurities), there will be extra electrons available for conduction. This is an example of an extrinsic semiconductor.

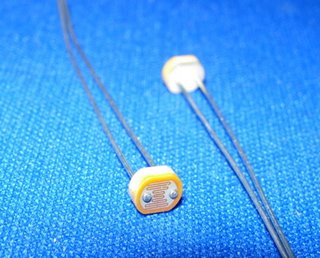


Figure 3.6: LDR

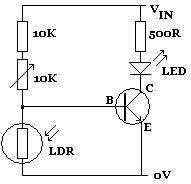
A Light Dependent Resistor (LDR, photoconductor, or photocell) is a device which has a resistance which varies according to the amount of light falling on its surface. They will be having a resistance of 1 MOhm in total darkness, and a resistance of a 1 to 10 of kOhm in bright light. A photoelectric device can be either intrinsic or extrinsic.

**3.6.2 Applications:**

An LDR can even be used in a simple remote control circuit using the backlight of a mobile phone to turn on a device - call the mobile from anywhere in the world, it lights up the LDR, and lighting can be turned on remotely!

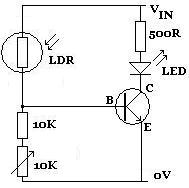
[](http://bp2.blogger.com/_XTElFWnvKxw/R0cRalkDIpI/AAAAAAAAAIA/ElQhklAfIxw/s1600-h/vt931.jpg)

There are two basic circuits using light dependent resistors - the first is activated by darkness, the second is activated by light.

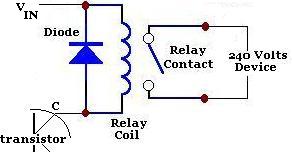
[](http://bp0.blogger.com/_XTElFWnvKxw/R0cNlFkDImI/AAAAAAAAAHo/JLsXCx-tCBI/s1600-h/ldr-darkness-activated-circuit.jpg)

In the circuit diagram on the left, the led lights up whenever the LDR is in darkness. The 10K variable resistor is used to fine-tune the level of darkness required before the LED lights up. The 10K standard resistor can be changed as required to achieve the desired effect, although any replacement must be at least 1K to protect the transistor from being damaged by excessive current.

By swapping the LDR over with the 10K and 10K variable resistors , the circuit will be activated instead by light. Whenever sufficient light falls on the LDR (manually fine-tuned using the 10K variable resistor), the LED will light up.

[](http://bp0.blogger.com/_XTElFWnvKxw/R0cOHFkDInI/AAAAAAAAAHw/IkLBsNtXijo/s1600-h/ldr-light-activated-circuit.jpg)

The circuits shown above are not practically useful. In a real world circuit, the LED (and resistor) between the positive voltage input (Vin) and the collector (C) of the transistor would be replaced with the device to be powered.



Typically a relay is used - particularly when the low voltage light detecting circuit is used to switch on (or off) a 240V mains powered device. A diagram of that part of the circuit is shown above. When darkness falls (if the LDR circuit is configured that way around), the relay is triggered and the 240V device - for example a security light - switches on.

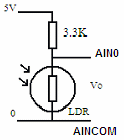
**3.6.3 Measure Light Intensity using Light Dependent Resistor (LDR):**

The relationship between the resistance RL and light intensity Lux for a typical LDR is

RL = 500 / Lux Kohm

With the LDR connected to 5V through a 3.3K resistor, the output voltage of the LDR is

Vo = 5\*RL / (RL+3.3)

[](http://bp1.blogger.com/_XTElFWnvKxw/R0cQdVkDIoI/AAAAAAAAAH4/J0FMeIlsxdg/s1600-h/LDR_html_m5cb0e77a.gif)

Reworking the equation, we obtain the light intensity

Lux = (2500/Vo - 500)/

**3.7 ESP 8266**

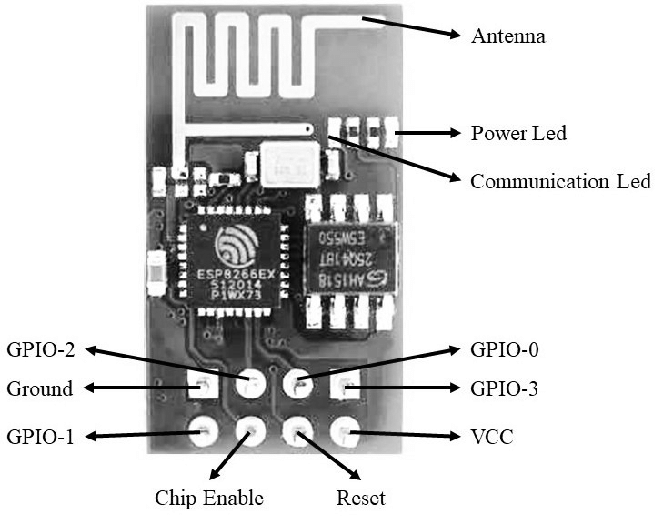
The **ESP8266** is a low-cost [Wi-Fi](https://en.wikipedia.org/wiki/Wi-Fi) microchip, with a full [TCP/IP stack](https://en.wikipedia.org/wiki/TCP/IP_stack) and [microcontroller](https://en.wikipedia.org/wiki/Microcontroller) capability, produced by [Espressif Systems](https://en.wikipedia.org/w/index.php?title=Espressif_Systems&action=edit&redlink=1" \o "Espressif Systems (page does not exist))[[1]](https://en.wikipedia.org/wiki/ESP8266#cite_note-Espressif_ESP8266-1) in Shanghai, China. The chip first came to the attention of Western [makers](https://en.wikipedia.org/wiki/Maker_culture) in August 2014 with the **ESP-01** module, made by a third-party manufacturer Ai-Thinker. This small module allows microcontrollers to connect to a Wi-Fi network and make simple TCP/IP connections using [Hayes](https://en.wikipedia.org/wiki/Hayes_command_set)-style commands. However, at first, there was almost no English-language documentation on the chip and the commands it accepted.[[2]](https://en.wikipedia.org/wiki/ESP8266#cite_note-2) The very low price and the fact that there were very few external components on the module, which suggested that it could eventually be very inexpensive in volume, attracted many hackers to explore the module, the chip, and the software on it, as well as to translate the Chinese documentation.[[3]](https://en.wikipedia.org/wiki/ESP8266#cite_note-3)The **ESP8285** is an ESP8266 with 1 MiB of built-in flash, allowing the building of single-chip devices capable of connecting to Wi-Fi These microcontroller chips have been succeeded by the [ESP32](https://en.wikipedia.org/wiki/ESP32) family of devices, including the pin-compatible ESP32-C3

**3.7.1 Features:**

1. 802.11 b/g/n
2. Wi-Fi Direct (P2P), soft-AP
3. Integrated TCP/IP protocol stack
4. Integrated TR switch, balun, LNA, power amplifier and matching network
5. Integrated PLLs, regulators, DCXO and power management units
6. +19.5dBm output power in 802.11b mode
7. Power down leakage current of <10uA
8. 1MB Flash Memory Integrated low power 32-bit CPU could be used as application processor
9. SDIO 1.1 / 2.0, SPI, UART
10. STBC, 1×1 MIMO, 2×1 MIMO
11. A-MPDU & A-MSDU aggregation & 0.4ms guard interval
12. Wake up and transmit packets in < 2ms
13. Standby power consumption of < 1.0mW (DTIM3)

**3.7.2 Description:**

The ESP8266 WiFi Module is a self contained SOC withintegrated TCP/IP protocol stack that can give any microcontroller access to your WiFi network. The ESP8266 is capable of either hosting an application or offloading all Wi-Fi networking functions from another application processor.



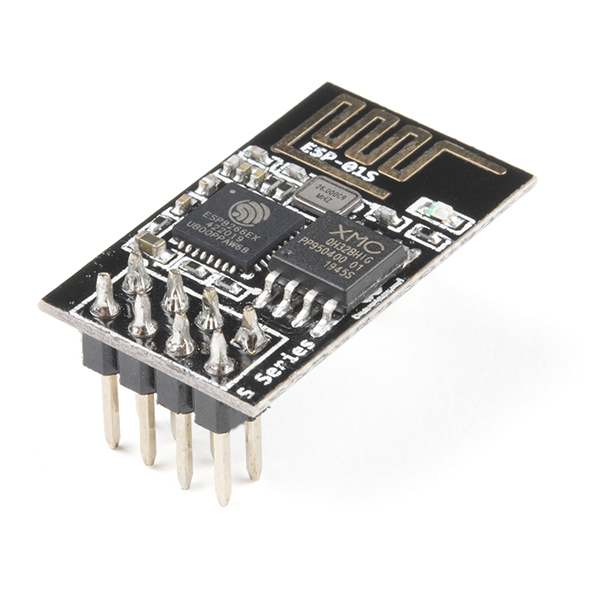


Figure 3.7.2 :ESP 8266

**3.7.3 Specification of ESP 8266:**

1. Wi-Fi Direct (P2P), soft-AP
2. Integrated TCP/IP protocol stack
3. Integrated TR switch, balun, LNA, power amplifier and matching network
4. Integrated PLLs, regulators, DCXO and power management units
5. 19.5dBm output power in 802.11b mode
6. Power down leakage current of <10uA
7. 1MB Flash Memory
8. Integrated low power 32-bit CPU could be used as application processor
9. Standby power consumption of < 1.0mW (DTIM3)

This module has a powerful enough on-board processing and storage capability that allows it to be integrated with the sensors and other application specific devices through its GPIOs with minimal development up-front and minimal loading during runtime. Its high degree of on-chip integration allows for minimal external circuitry, including the front-end module, is designed to occupy minimal PCB area.

The ESP8266 supports APSD for VoIP applications and Bluetooth co-existance interfaces, it contains a self-calibrated RF allowing it to work under all operating conditions, and requires no external RF parts. There is an almost limitless fountain of information available for the ESP8266, all of which has been provided by amazing community support. In the *Documents* section below you will find many resources to aid you in using the ESP8266, even instructions on how to transforming this module into an IoT (Internet of Things) solution!

**3.8 LCD Module**

To display interactive messages we are using LCD Module. We examine an intelligent LCD display of two lines, 16 characters per line that is interfaced to the controllers. The protocol (handshaking) for the display is as shown. Whereas D0 to D7th bit is the Data lines, RS, RW and EN pins are the control pins and remaining pins are +5V, -5V and GND to provide supply. Where RS is the Register Select, RW is the Read Write and EN is the Enable pin.

The display contains two internal byte-wide registers, one for commands (RS=0) and the second for characters to be displayed (RS=1). It also contains a user-programmed RAM area (the character RAM) that can be programmed to generate any desired character that can be formed using a dot matrix. To distinguish between these two data areas, the hex command byte 80 will be used to signify that the display RAM address 00h will be chosen.Port1 is used to furnish the command or data type, and ports 3.2 to 3.4 furnish register select and read/write levels.

The display takes varying amounts of time to accomplish the functions as listed. LCD bit 7 is monitored for logic high (busy) to ensure the display is overwritten. Liquid Crystal Display also called as LCD is very helpful in providing user interface as well as for debugging purpose. The most common type of LCD controller is HITACHI 44780 which provides a simple interface between the controller & an LCD. These LCD's are very simple to interface with the controller as well as are cost effective.



Figure 3.8 :16 X 2 Line Alphanumeric LCD Display

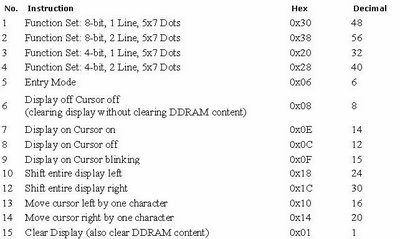
The most commonly used *ALPHANUMERIC*displays are *1x16* (Single Line & 16 characters), *2x16* (Double Line & 16 character per line) & *4x20*(four lines & Twenty characters per line).

The LCD requires 3 control lines (RS, R/W & EN) & 8 (or 4) data lines. The number on data lines depends on the mode of operation. If operated in 8-bit mode then 8 data lines + 3 control lines i.e. total 11 lines are required. And if operated in 4-bit mode then 4 data lines + 3 control lines i.e. 7 lines are required. How do we decide which mode to use? It’s simple if you have sufficient data lines you can go for 8 bit mode & if there is a time constrain i.e. display should be faster then we have to use 8-bit mode because basically 4-bit mode takes twice as more time as compared to 8-bit mode.

|  |  |  |
| --- | --- | --- |
| Pin | Symbol | Function |
| 1 | Vss | Ground |
| 2 | Vdd | Supply Voltage |
| 3 | Vo | Contrast Setting |
| 4 | RS | Register Select |
| 5 | R/W | Read/Write Select |
| 6 | En | Chip Enable Signal |
| 7-14 | DB0-DB7 | Data Lines |
| 15 | A/Vee | Gnd for the backlight |
| 16 | K | Vcc for backlight |

When *RS* is low (0), the data is to be treated as a command. When RS is high (1), the data being sent is considered as text data which should be displayed on the screen.  When *R/W* is low (0), the information on the data bus is being written to the LCD. When RW is high (1), the program is effectively reading from the LCD. Most of the times there is no need to read from the LCD so this line can directly be connected to Gnd thus saving one controller line. The *ENABLE* pin is used to latch the data present on the data pins. A HIGH - LOW signal is required to latch the data. The LCD interprets and executes our command at the instant the EN line is brought low. If you never bring EN low, your instruction will never be executed.

**3.9 COMMANDS USED IN LCD**



**CHAPTER 4**

**SOFTWARE REQUIREMENT**

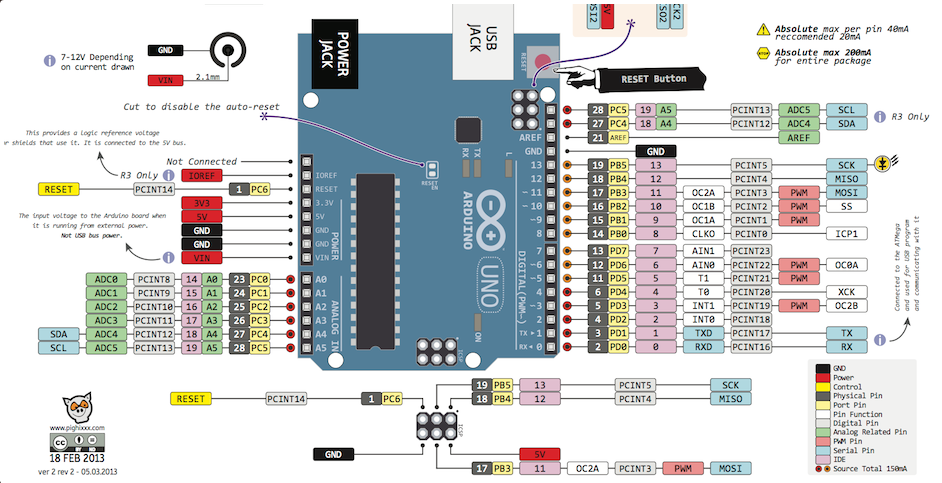
**4.1 Arduino**

**Arduino** is a computer hardware and software company, project, and user community that designs and manufactures [microcontroller](https://en.wikipedia.org/wiki/Microcontroller) kits for building digital devices and interactive objects that can sense and control objects in the physical world. The project's products are distributed as [open-source hardware](https://en.wikipedia.org/wiki/Open-source_hardware) and [software](https://en.wikipedia.org/wiki/Open-source_software), which are licensed under the [GNU Lesser General Public License](https://en.wikipedia.org/wiki/GNU_Lesser_General_Public_License) (LGPL) or the [GNU General Public License](https://en.wikipedia.org/wiki/GNU_General_Public_License) (GPL),[[1]](https://en.wikipedia.org/wiki/Arduino#cite_note-1) permitting the manufacture of Arduino boards and software distribution by anyone. Arduino boards are available commercially in preassembled form, or as [do-it-yourself](https://en.wikipedia.org/wiki/Do-it-yourself) kits.

Arduino board designs use a variety of microprocessors and controllers. The boards are equipped with sets of digital and analog [input/output](https://en.wikipedia.org/wiki/Input/output) (I/O) pins that may be interfaced to various expansion boards (*shields*) and other circuits. The boards feature serial communications interfaces, including [Universal Serial Bus](https://en.wikipedia.org/wiki/Universal_Serial_Bus) (USB) on some models, which are also used for loading programs from personal computers. The microcontrollers are typically programmed using a dialect of features from the programming languages [C](https://en.wikipedia.org/wiki/C_(programming_language)) and [C++](https://en.wikipedia.org/wiki/C%2B%2B). In addition to using traditional compiler toolchains, the Arduino project provides an [integrated development environment](https://en.wikipedia.org/wiki/Integrated_development_environment) (IDE) based on the [Processing](https://en.wikipedia.org/wiki/Processing_(programming_language)) language project.

The Arduino project started in 2005 as a program for students at the [Interaction Design Institute Ivrea](https://en.wikipedia.org/wiki/Interaction_Design_Institute_Ivrea) in [Ivrea](https://en.wikipedia.org/wiki/Ivrea), Italy,[[2]](https://en.wikipedia.org/wiki/Arduino#cite_note-kushner-2) aiming to provide a low-cost and easy way for novices and professionals to create devices that interact with their environment using [sensors](https://en.wikipedia.org/wiki/Sensor) and [actuators](https://en.wikipedia.org/wiki/Actuator). Common examples of such devices intended for beginner hobbyists include simple [robots](https://en.wikipedia.org/wiki/Robot), [thermostats](https://en.wikipedia.org/wiki/Thermostat), and [motion detectors](https://en.wikipedia.org/wiki/Motion_detector).

The name *Arduino* comes from a bar in Ivrea, Italy, where some of the founders of the project used to meet. The bar was named after [Arduino of Ivrea](https://en.wikipedia.org/wiki/Arduin_of_Ivrea), who was the margrave of the [March of Ivrea](https://en.wikipedia.org/wiki/March_of_Ivrea) and [King of Italy](https://en.wikipedia.org/wiki/King_of_Italy) from 1002 to 1014.



Figureu 4.1 : Ardunio

**4.2 History**

The origin of the Arduino project started at the [Interaction Design Institute Ivrea](https://en.wikipedia.org/wiki/Interaction_Design_Institute_Ivrea) (IDII) in [Ivrea](https://en.wikipedia.org/wiki/Ivrea), Italy.[[2]](https://en.wikipedia.org/wiki/Arduino#cite_note-kushner-2) At that time, the students used a [BASIC Stamp](https://en.wikipedia.org/wiki/BASIC_Stamp) microcontroller at a cost of $100, a considerable expense for many students. In 2004, Colombian student Hernando Barragán created the development platform [*Wiring*](https://en.wikipedia.org/wiki/Wiring_(development_platform)) as a Master's thesis project at IDII, under the supervision of Massimo Banzi and Casey Reas, who are known for work on the [Processing](https://en.wikipedia.org/wiki/Processing_(programming_language)) language. The project goal was to create simple, low cost tools for creating digital projects by non-engineers. The Wiring platform consisted of a [printed circuit board](https://en.wikipedia.org/wiki/Printed_circuit_board) (PCB) with an [ATmega](https://en.wikipedia.org/wiki/ATmega)168 microcontroller, an IDE based on Processing and library functions to easily program the microcontroller.[[4]](https://en.wikipedia.org/wiki/Arduino#cite_note-:0-4)

In 2005, Massimo Banzi, with David Mellis, another IDII student, and David Cuartielles, added support for the cheaper ATmega8 microcontroller to Wiring. But instead of continuing the work on Wiring, they copied the Wiring source code and renamed it as a separate project, called Arduino.[[4]](https://en.wikipedia.org/wiki/Arduino#cite_note-:0-4)

The initial Arduino core team consisted of Massimo Banzi, David Cuartielles, Tom Igoe, Gianluca Martino, and David Mellis,[[2]](https://en.wikipedia.org/wiki/Arduino#cite_note-kushner-2) but Barragán was not invited to participate.[[4]](https://en.wikipedia.org/wiki/Arduino#cite_note-:0-4) Following the completion of the Wiring platform, lighter and less-expensive versions were distributed in the open-source community.[[5]](https://en.wikipedia.org/wiki/Arduino#cite_note-5) [Adafruit Industries](https://en.wikipedia.org/wiki/Adafruit_Industries), a New York City supplier of Arduino boards, parts, and assemblies, estimated in mid-2011 that over 300,000 official Arduinos had been commercially produced,[[6]](https://en.wikipedia.org/wiki/Arduino#cite_note-6) and in 2013 that 700,000 official boards were in users' hands.

**4.3 Hardware**

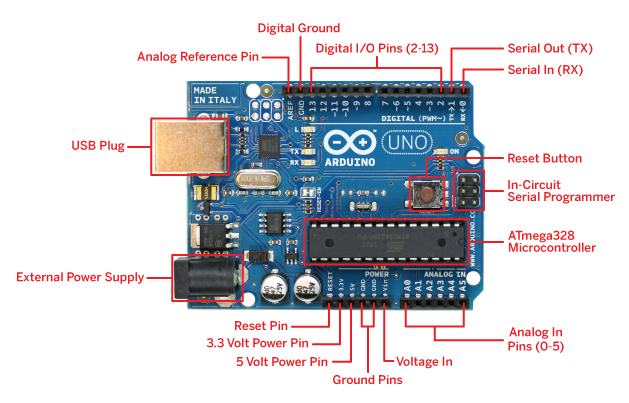


Figure 4.3 : Arduino hardware

Arduino is [open-source hardware](https://en.wikipedia.org/wiki/Open-source_hardware). The hardware reference designs are distributed under a [Creative Commons](https://en.wikipedia.org/wiki/Creative_Commons) Attribution Share-Alike 2.5 license and are available on the Arduino website. Layout and production files for some versions of the hardware are also available. The source code for the IDE is released under the [GNU General Public License](https://en.wikipedia.org/wiki/GNU_General_Public_License), version 2.[[8]](https://en.wikipedia.org/wiki/Arduino#cite_note-8) Nevertheless, an official [Bill of Materials](https://en.wikipedia.org/wiki/Bill_of_Materials) of Arduino boards has never been released by Arduino staff. Although the hardware and software designs are freely available under [copy left](https://en.wikipedia.org/wiki/Copyleft) licenses, the developers have requested that the name *Arduino* be [exclusive to the official product](https://en.wikipedia.org/wiki/Generic_trademark) and not be used for derived works without permission. The official policy document on use of the Arduino name emphasizes that the project is open to incorporating work by others into the official product.[[9]](https://en.wikipedia.org/wiki/Arduino#cite_note-AutoF7-44-9) Several Arduino-compatible products commercially released have avoided the project name by using various names ending in *-duino*.[[10]](https://en.wikipedia.org/wiki/Arduino#cite_note-freeduino-10)

An Arduino board consists of an [Atmel](https://en.wikipedia.org/wiki/Atmel) 8-, 16- or 32-bit AVR [microcontroller](https://en.wikipedia.org/wiki/Microcontroller) (ATmega8, ATmega168, [ATmega328](https://en.wikipedia.org/wiki/ATmega328), ATmega1280, ATmega2560), but other makers' microcontrollers have been used since 2015. The boards use single-row pins or female headers that facilitate connections for programming and incorporation into other circuits. These may connect with add-on modules termed *shields*. Multiple, and possibly stacked shields may be individually addressable via an [I²C](https://en.wikipedia.org/wiki/I%C2%B2C) [serial bus](https://en.wikipedia.org/wiki/Serial_bus). Most boards include a 5 V [linear regulator](https://en.wikipedia.org/wiki/Linear_regulator) and a 16 MHz [crystal oscillator](https://en.wikipedia.org/wiki/Crystal_oscillator) or [ceramic resonator](https://en.wikipedia.org/wiki/Ceramic_resonator). Some designs, such as the LilyPad, run at 8 MHz and dispense with the onboard voltage regulator due to specific form-factor restrictions.

Arduino microcontrollers are pre-programmed with a [boot loader](https://en.wikipedia.org/wiki/Boot_loader) that simplifies uploading of programs to the on-chip [flash memory](https://en.wikipedia.org/wiki/Flash_memory). The default bootloader of the Aduino UNO is the optiboot bootloader.[[12]](https://en.wikipedia.org/wiki/Arduino#cite_note-12) Boards are loaded with program code via a serial connection to another computer. Some serial Arduino boards contain a level shifter circuit to convert between [RS-232](https://en.wikipedia.org/wiki/RS-232) logic levels and [transistor–transistor logic](https://en.wikipedia.org/wiki/Transistor%E2%80%93transistor_logic) (TTL) level signals. Current Arduino boards are programmed via [Universal Serial Bus](https://en.wikipedia.org/wiki/Universal_Serial_Bus) (USB), implemented using USB-to-serial adapter chips such as the [FTDI](https://en.wikipedia.org/wiki/FTDI) FT232. Some boards, such as later-model Uno boards, substitute the [FTDI](https://en.wikipedia.org/wiki/FTDI) chip with a separate AVR chip containing USB-to-serial firmware, which is reprogrammable via its own ICSP header. Other variants, such as the Arduino Mini and the unofficial Boarduino, use a detachable USB-to-serial adapter board or cable, [Bluetooth](https://en.wikipedia.org/wiki/Bluetooth) or other methods, when used with traditional microcontroller tools instead of the Arduino IDE, standard AVR [in-system programming](https://en.wikipedia.org/wiki/In-system_programming) (ISP) programming is used.

The Arduino board exposes most of the microcontroller's I/O pins for use by other circuits. The *Diecimila*,[[a]](https://en.wikipedia.org/wiki/Arduino#cite_note-N10000-13) *Duemilanove*,[[b]](https://en.wikipedia.org/wiki/Arduino#cite_note-N2009-14) and current *Uno*[[c]](https://en.wikipedia.org/wiki/Arduino#cite_note-N1-15) provide 14 digital I/O pins, six of which can produce [pulse-width modulated](https://en.wikipedia.org/wiki/Pulse-width_modulation) signals, and six analog inputs, which can also be used as six digital I/O pins. These pins are on the top of the board, via female 0.1-inch (2.54 mm) headers. Several plug-in application shields are also commercially available. The Arduino Nano, and Arduino-compatible Bare Bones Board[[13]](https://en.wikipedia.org/wiki/Arduino#cite_note-16) and Boarduino[[14]](https://en.wikipedia.org/wiki/Arduino" \l "cite_note-17) boards may provide male header pins on the underside of the board that can plug into solder less [bread boards](https://en.wikipedia.org/wiki/Breadboard). Many Arduino-compatible and Arduino-derived boards exist. Some are functionally equivalent to an Arduino and can be used interchangeably. Many enhance the basic Arduino by adding output drivers, often for use in school-level education, to simplify making buggies and small robots. Others are electrically equivalent but change the form factor, sometimes retaining compatibility with shields, sometimes not. Some variants use different processors, of varying compatibility.

**4.4 Software development**

A program for Arduino may be written in any [programming language](https://en.wikipedia.org/wiki/Programming_language) for a compiler that produces binary machine code for the target processor. Atmel provides a development environment for their microcontrollers, AVR Studio and the newer Atmel Studio.

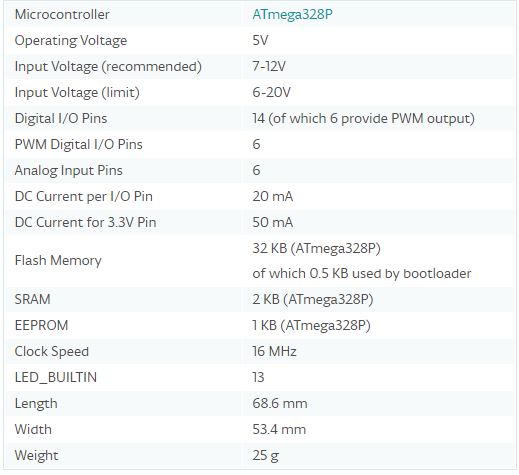
The Arduino project provides the Arduino [integrated development environment](https://en.wikipedia.org/wiki/Integrated_development_environment) (IDE), which is a [cross-platform](https://en.wikipedia.org/wiki/Cross-platform) application written in the programming language [Java](https://en.wikipedia.org/wiki/Java_(programming_language)). It originated from the IDE for the languages [*Processing*](https://en.wikipedia.org/wiki/Processing_(programming_language)) and [*Wiring*](https://en.wikipedia.org/wiki/Wiring_(development_platform)). It includes a code editor with features such as text cutting and pasting, searching and replacing text, automatic indenting, [brace matching](https://en.wikipedia.org/wiki/Brace_matching), and [syntax highlighting](https://en.wikipedia.org/wiki/Syntax_highlighting), and provides simple *one-click* mechanisms to compile and upload programs to an Arduino board. It also contains a message area, a text console, a toolbar with buttons for common functions and a hierarchy of operation menus.

A program written with the IDE for Arduino is called a *sketch*.[[40]](https://en.wikipedia.org/wiki/Arduino#cite_note-43) Sketches are saved on the development computer as text files with the file extension *.ino*. Arduino Software (IDE) pre-1.0 saved sketches with the extension *.pde*. The Arduino IDE supports the languages [C](https://en.wikipedia.org/wiki/C_(programming_language)) and [C++](https://en.wikipedia.org/wiki/C%2B%2B) using special rules of code structuring. The Arduino IDE supplies a [software library](https://en.wikipedia.org/wiki/Software_library) from the [Wiring](https://en.wikipedia.org/wiki/Wiring_(development_platform)) project, which provides many common input and output procedures. User-written code only requires two basic functions, for starting the sketch and the main program loop, that are compiled and linked with a program stub *main()* into an executable [cyclic executive](https://en.wikipedia.org/wiki/Cyclic_executive) program with the [GNU toolchain](https://en.wikipedia.org/wiki/GNU_toolchain), also included with the IDE distribution. The Arduino IDE employs the program *avrdude* to convert the executable code into a text file in hexadecimal encoding that is loaded into the Arduino board by a loader program in the board's firmware.

**4.4.1 Applications**

1. Xoscillo, an open-source [oscilloscope](https://en.wikipedia.org/wiki/Oscilloscope)[[48]](https://en.wikipedia.org/wiki/Arduino#cite_note-AutoF7-40-51)
2. [Arduinome](https://en.wikipedia.org/wiki/Arduinome), a [MIDI controller](https://en.wikipedia.org/wiki/MIDI_controller) device that mimics the [Monome](https://en.wikipedia.org/wiki/Monome" \o "Monome)
3. [OBDuino](https://en.wikipedia.org/wiki/OBDuino), a [trip computer](https://en.wikipedia.org/wiki/Trip_computer) that uses the [on-board diagnostics](https://en.wikipedia.org/wiki/On-board_diagnostics) interface found in most modern cars
4. [Ardupilot](https://en.wikipedia.org/wiki/Ardupilot), drone software and hardware
5. Gameduino, an Arduino shield to create retro 2D video games[[49]](https://en.wikipedia.org/wiki/Arduino#cite_note-52)
6. Arduino Phone, a do-it-yourself cellphone[[50]](https://en.wikipedia.org/wiki/Arduino#cite_note-53)[[51]](https://en.wikipedia.org/wiki/Arduino#cite_note-54)
7. Water quality testing platform[[52]](https://en.wikipedia.org/wiki/Arduino#cite_note-55)
8. Automatic titration system based on Arduino and stepper motor[[53]](https://en.wikipedia.org/wiki/Arduino#cite_note-56)
9. Low cost data glove for virtual reality applications[[54]](https://en.wikipedia.org/wiki/Arduino#cite_note-57)
10. Impedance sensor system to detect bovine milk adulteration[[55]](https://en.wikipedia.org/wiki/Arduino#cite_note-58)
11. Homemade CNC using Arduino and DC motors with close loop control by Homofaciens[[56]](https://en.wikipedia.org/wiki/Arduino" \l "cite_note-59)
12. DC motor control using Arduino and H-Bridge[[](https://en.wikipedia.org/wiki/Arduino#cite_note-60)

**4.4.2 Technical specs**



**4.5 Programming**

The Arduino/Genuino Uno can be programmed with the ([Arduino Software](https://www.arduino.cc/en/Main/Software) (IDE)). Select "Arduino/Genuino Uno from the Tools > Board menu (according to the microcontroller on your board). For details, see the [reference](https://www.arduino.cc/en/Reference/HomePage) and [tutorials](https://www.arduino.cc/en/Tutorial/HomePage). The ATmega328 on the Arduino/Genuino Uno comes preprogrammed with a [bootloader](https://www.arduino.cc/en/Hacking/Bootloader?from=Tutorial.Bootloader) that allows you to upload new code to it without the use of an external hardware programmer. It communicates using the original STK500 protocol ([reference](http://www.atmel.com/Images/doc2525.pdf), [C header files](http://www.atmel.com/dyn/resources/prod_documents/avr061.zip)).

You can also bypass the bootloader and program the microcontroller through the ICSP (In-Circuit Serial Programming) header using [Arduino ISP](https://www.arduino.cc/en/Main/ArduinoISP) or similar; see [these instructions](https://www.arduino.cc/en/Hacking/Programmer) for details. The ATmega16U2 (or 8U2 in the rev1 and rev2 boards) firmware source code is available in the Arduino repository. The ATmega16U2/8U2 is loaded with a DFU bootloader, 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 rese ing 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.

You can then use [Atmel's FLIP software](http://www.atmel.com/products/microcontrollers/default.aspx) (Windows) or the [DFU programmer](http://dfu-programmer.github.io/) (Mac OS X and Linux) to load a new firmware. Or you can use the ISP header with an external programmer (overwriting the DFU bootloader). See [this user-contributed tutorial](http://forum.arduino.cc/index.php/topic,111.0.html) for more information.

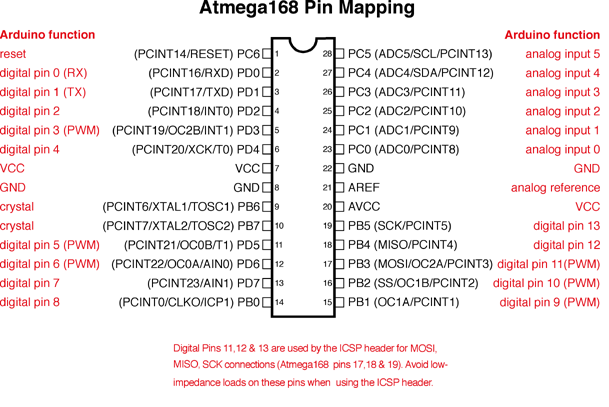
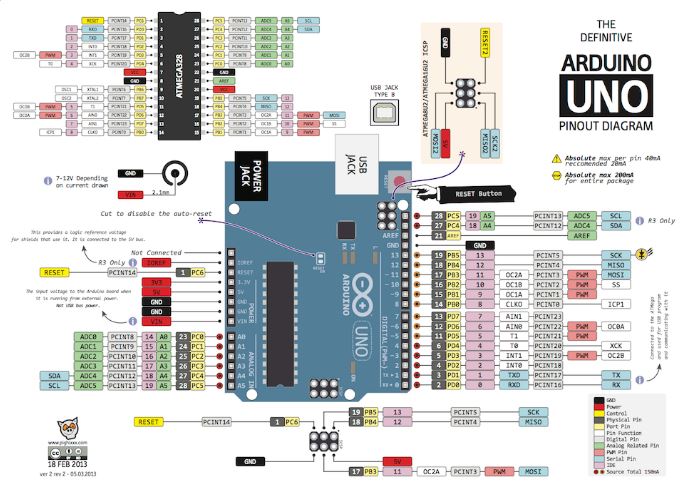


Figure 4.5: Pin Diagram



**4.6 Warnings**

The Arduino/Genuino Uno has a resettable polyfuse that protects your computer's USB ports from shorts and overcurrent. Although most computers provide their own internal protection, the fuse provides an extra layer of protection. If more than 500 mA is applied to the USB port, the fuse will automatically break the connection until the short or overload is removed.

**4.6.1 Differences with other boards**

The Uno differs from all preceding boards in that it does not use the FTDI USB-to-serial driver chip. Instead, it features the Atmega16U2 (Atmega8U2 up to version R2) programmed as a USB-to-serial converter.

**4.7 Power**

The Arduino/Genuino Uno board can be powered via the USB connection or with an external power supply. The power source is selected automatically. External (non-USB) power can come either from an AC-to-DC adapter (wall-wart) or battery. The adapter can be connected by plugging a 2.1mm center-positive plug into the board's power jack. Leads from a battery can be inserted in the GND and Vin pin headers of the POWER connector.

The board can operate on an external supply from 6 to 20 volts. If supplied with less than 7V, however, the 5V pin may supply less than five volts and the board may become unstable. If using more than 12V, the voltage regulator may overheat and damage the board. The power pins are as follows:

* Vin. The input voltage to the Arduino/Genuino board when it's using an external power source (as opposed to 5 volts from the USB connection or other regulated power source). You can supply voltage through this pin, or, if supplying voltage via the power jack, access it through this pin.
* 5V.This pin outputs a regulated 5V from the regulator on the board. The board can be supplied with power either from the DC power jack (7 - 12V), the USB connector (5V), or the VIN pin of the board (7-12V). Supplying voltage via the 5V or 3.3V pins bypasses the regulator, and can damage your board. We don't advise it.
* 3V3. A 3.3 volt supply generated by the on-board regulator. Maximum current draw is 50 mA.
* GND. Ground pins.
* IOREF. This pin on the Arduino/Genuino board provides the voltage reference with which the microcontroller operates. A properly configured shield can read the IOREF pin voltage and select the appropriate power source or enable voltage translators on the outputs to work with the 5V or 3.3V.

**4.8 Memory**

The ATmega328 has 32 KB (with 0.5 KB occupied by the bootloader). It also has 2 KB of SRAM and 1 KB of EEPROM (which can be read and written with the [EEPROM library](https://www.arduino.cc/en/Reference/EEPROM)).

**4.8.1 Input and Output**

See the mapping between Arduino pins and ATmega328P ports. The mapping for the Atmega8, 168, and 328 is identical. Each of the 14 digital pins on the Uno can be used as an input or output, using [pinMode()](https://www.arduino.cc/en/Reference/PinMode), [digitalWrite()](https://www.arduino.cc/en/Reference/DigitalWrite), and [digitalRead()](https://www.arduino.cc/en/Reference/DigitalRead) functions. They operate at 5 volts. Each pin can provide or receive 20 mA as recommended operating condition and has an internal pull-up resistor (disconnected by default) of 20-50k ohm. A maximum of 40mA is the value that must not be exceeded on any I/O pin to avoid permanent damage to the microcontroller.

In addition, some pins have specialized functions:

* Serial: 0 (RX) and 1 (TX). Used to receive (RX) and transmit (TX) TTL serial data. These pins are connected to the corresponding pins of the ATmega8U2 USB-to-TTL Serial chip.
* External Interrupts: 2 and 3. These pins can be configured to trigger an interrupt on a low value, a rising or falling edge, or a change in value. See the attachInterrupt() function for details.
* PWM: 3, 5, 6, 9, 10, and 11. Provide 8-bit PWM output with the analogWrite() function.
* SPI: 10 (SS), 11 (MOSI), 12 (MISO), 13 (SCK). These pins support SPI communication using the SPI library.
* LED: 13. There is a built-in LED driven by digital pin 13. When the pin is HIGH value, the LED is on, when the pin is LOW, it's off.
* TWI: A4 or SDA pin and A5 or SCL pin. Support TWI communication using the Wire library.
* The Uno has 6 analog inputs, labeled A0 through A5, each of which provide 10 bits of resolution (i.e. 1024 different values). By default they measure from ground to 5 volts, though is it possible to change the upper end of their range using the AREF pin and the analogReference() function. There are a couple of other pins on the board:  
  + AREF. Reference voltage for the analog inputs. Used with analogReference()

**4.8.2Communication**

Arduino/Genuino Uno has a number of facilities for communicating with a computer, another Arduino/Genuino board, or other microcontrollers. The ATmega328 provides UART TTL (5V) serial communication, which is available on digital pins 0 (RX) and 1 (TX). 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, a .inf file is required](https://www.arduino.cc/en/Guide/Windows#toc4). The Arduino Software (IDE) includes a serial monitor which allows simple textual data to be sent to and from the 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 [SoftwareSerial library](https://www.arduino.cc/en/Reference/SoftwareSerial) allows serial communication on any of the Uno's digital pins. The ATmega328 also supports I2C (TWI) and SPI communication. The Arduino Software (IDE) includes a Wire library to simplify use of the I2C bus; see the [documentation](https://www.arduino.cc/en/Reference/Wire) for details. For SPI communication, use the [SPI library](https://www.arduino.cc/en/Reference/SPI).

**4.8.3 Automatic (Software) Reset**

Rather than requiring a physical press of the reset button before an upload, the Arduino/Genuino Uno board is designed in a way that allows it to be reset by software running on a connected computer. One of the hardware flow control lines (DTR) of the ATmega8U2/16U2 is connected to the reset line of the ATmega328 via a 100 nanofarad capacitor. When this line is asserted (taken low), the reset line drops long enough to reset the chip. The Arduino Software (IDE) uses this capability to allow you to upload code by simply pressing the upload button in the interface toolbar. This means that the bootloader can have a shorter timeout, as the lowering of DTR can be well-coordinated with the start of the upload.

This setup has other implications. When the Uno is connected to either a computer running Mac OS X or Linux, it resets each time a connection is made to it from software (via USB). For the following half-second or so, the bootloader is running on the Uno. While it is programmed to ignore malformed data (i.e. anything besides an upload of new code), it will intercept the first few bytes of data sent to the board after a connection is opened. If a sketch running on the board receives one-time configuration or other data when it first starts, make sure that the software with which it communicates waits a second after opening the connection and before sending this data.

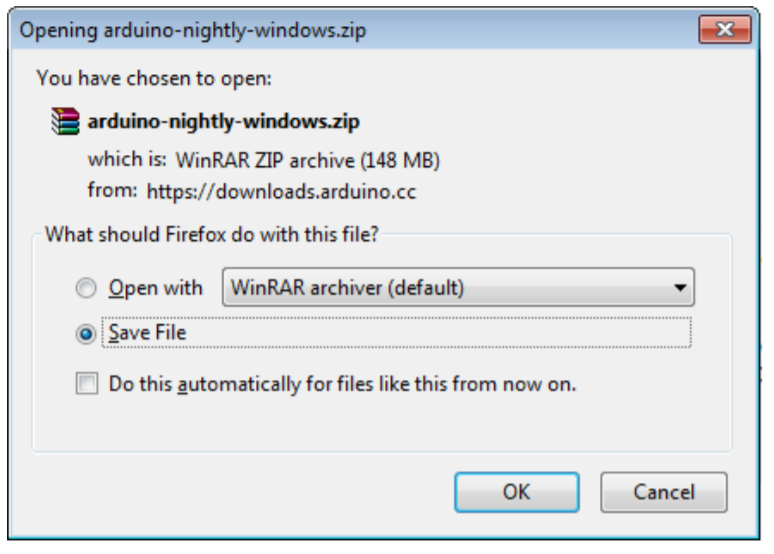
The Uno board contains a trace that can be cut to disable the auto-reset. The pads on either side of the trace can be soldered together to re-enable it. It's labeled "RESET-EN". You may also be able to disable the auto-reset by connecting a 110 ohm resistor from 5V to the reset line; see [this forum thread](http://forum.arduino.cc/index.php/topic,22974.0.html) for details.

**4.8.4 Revisions:**

Revision 3 of the board has the following new features:1.0 pin out: added SDA and SCL pins that are near to the AREF pin and two other new pins placed near to the RESET pin, the IOREF that allow the shields to adapt to the voltage provided from the board. In future, shields will be compatible with both the board that uses the AVR, which operates with 5V and with the Arduino Due that operates with 3.3V. The second one is a not connected pin, that is reserved for future purposes.

1. Stronger RESET circuit.
2. Atmega 16U2 replace the 8U2.

Arduino IDE Software. You can get different versions of Arduino IDE from the Download page on the Arduino Official website. You must select your software, which is compatible with your operating system (Windows, IOS, or Linux). After your file download is complete, unzip the file.



**Fig. 3.1 Opening arduino-nightly-windows.zip**

Launch Arduino IDE. After your Arduino IDE software is downloaded, you need to unzip the folder. Inside the folder, you can find the application icon with an infinity label (application.exe). Doubleclick the icon to start the IDE.

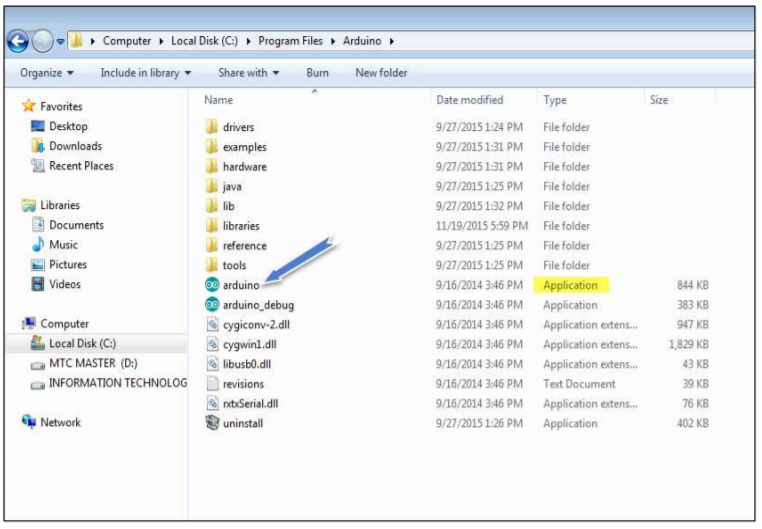


Fig:3.2 Launch Arduino IDE

Open your first project. Once the software starts, you have two options:

• Create a new project.

• Open an existing project example.

To create a new project, select File --> New

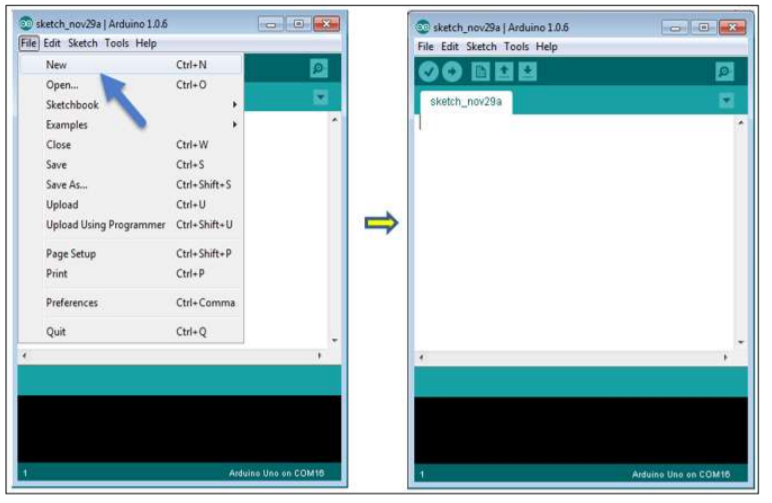


Figure:3.2 Create a new project

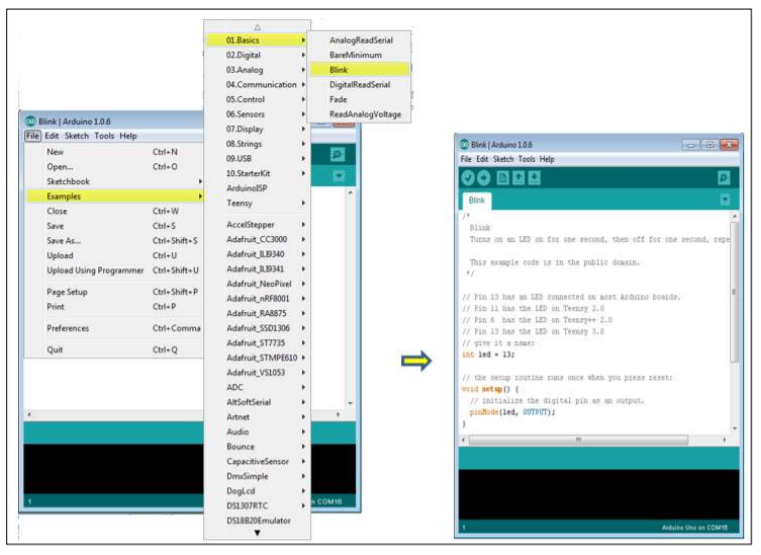


Figure: 3.3Open an existing project example

Here, we are selecting just one of the examples with the name Blink. It turns the LED on and off with some time delay. You can select any other example from the list Select your serial port. Select the serial device of the Arduino board. Go to Tools -> Serial Port menu. This is likely to be COM3 or higher (COM1 and COM2 are usually reserved for hardware serial ports). To find out, you can disconnect your Arduino board and re-open the menu, the entry that disappears should be of the Arduino board. Reconnect the board and select that serial port.

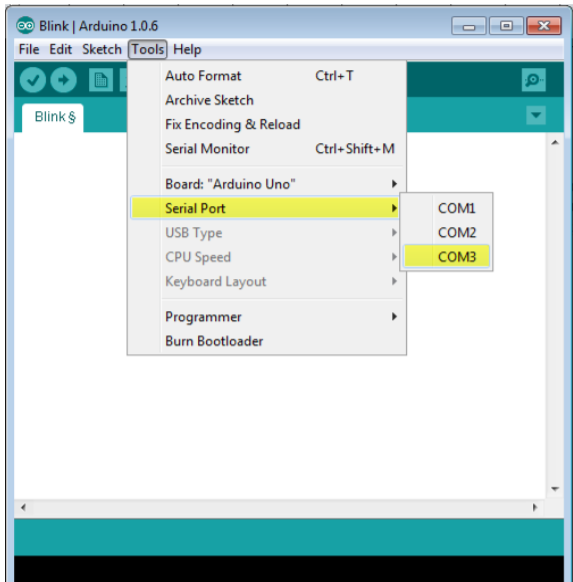


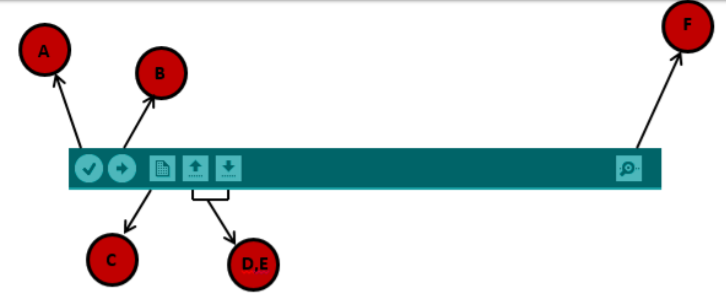
Figure:3.4 Select your serial port

Before explaining how we can upload our program to the board, we must demonstrate the function of each symbol appearing in the Arduino IDE toolbar.

1. Used to check if there is any compilation error.
2. Used to upload a program to the Arduino board.
3. Shortcut used to create a new sketch.
4. Used to directly open one of the example sketch.
5. Used to save your sketch.
6. Serial monitor used to receive serial data from the board and send the serial data to the board.

Now, simply click the "Upload" button in the environment.

Wait a few seconds; you will see the RX and TX LEDs on the board, flashing. If the upload is successful, the message "Done uploading" will appear in the status bar.

  
Fig 3.5 function of each symbol appearing in the Arduino IDE toolbar

In this chapter, we will study in depth, the Arduino program structure and we will learn more new terminologies used in the Arduino world. The Arduino software is open-source. The source code for the Java environment is released under the GPL and the C/C++ microcontroller libraries are under the LGPL. Sketch: The first new terminology is the Arduino program called “sketch”. Structure Arduino programs can be divided in three main parts: Structure, Values (variables and constants), and Functions. In this tutorial, we will learn about the Arduino software program, step by step, and how we can write the program without any syntax or compilation error. Let us start with the Structure. Software structure consist of two main functions:

Setup( ) function

Loop( ) function

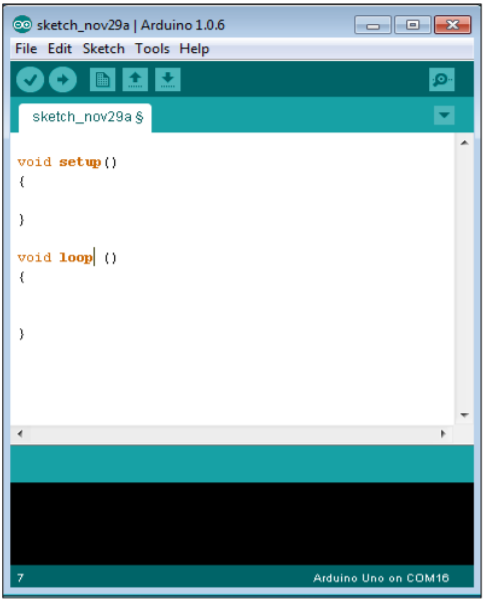


Figure: 3.6 Bare minimum code

Data types in C refers to an extensive system used for declaring variables or functions of different types. The type of a variable determines how much space it occupies in the storage and how the bit pattern stored is interpreted. The following table provides all the data types that you will use during Arduino programming.

**CHAPTER V**

**RESULTS**

**6.1 CONCLUSION**

Thus in this paper the PIC control section controls and co- ordinates all the activities of the energy meter, which is designed to continuously monitor the energy meter reading and to disconnect the power connection remotely whenever fails to pay the bill after warning period is lapsed and power reconnect after bill payments. It avoids the human intervention, provides efficient meter reading, avoid the billing error and reduce the maintenance cost. It display the corresponding information on LCD for user notification. The generated bill can also be sent to the consumer by GSM. Thus the complete process of monitoring of energy meter, bill calculation, notification of due date, meter disconnection or reconnection can be automated efficiently with better performance and less manpower.

**CHAPTER -7**

**BIBILOGRAPHY**

**CHAPTER 7**

**BIBILOGRAPHY**

[1] Das v.v, "Wireless Communication System for Energy Meter Reading", International Conferences on Advances in Recent Technologies in Communication V and Computing, pp. 896-898,(2012).

[2] Tanvir Ahmed, Md Suzan Miah, Md. Manirul Islam and Md. RakibUddin, "Automatic Electric Meter Reading System: A Cost-Feasible Alternative Approach In Meter Reading For Bangladesh Perspective Using Low-Cost Digital Wattmeter AndWimax Technology", International Journal Eng. Tech, pp. 800-807, (2011).

[3] Liting Cao, Jingwen Tian and Dahang Zhang, "Networked Remote Meter-Reading System Based on Wireless Communication Technology", International Conference on Information Acquisition, IEEE,(2013).

[4] Li Xiaoguang Hu, "Design of an ARM-Based Power Meter Having WIFI Wireless Communication Module", IEEE, (2009).

[5] Rodney Tan H.G,C.H. Lee and V.H. Mok, "Automatic Power Meter Reading system using GSM Network", The 8th International Power Engineering Conference, pp. 465- 469, (2009).

[6] Li Quan-Xi and Li Gang, "Design of Remote Automatic Meter Reading System based on Zigbee and GPRS", Proceedings of the Third International Symposium on Computer Science and Computational Technology (ISCSCT '10), pp. 186-189,(2009).

[7] Boyina S. Rao and Gnanasekaranathan .B, “Domestic Prepaid Distribution System”, International Journal of Advanced Engineering Technology (IJAET) Vol.3, Issue 2, 2012.

[8] Nagaraju Kommu, Pammi. Nagamani and Manoj Kollam, "Designing of Automated Power meter reading Zigbee Communication", International Journal of Computer & Communication Technology, (2011).

[9] Shaik Meeravali, S Nagireddi and S Rajitha, "Designing and Control Mechanism of Zigbee based Automatic Meter Reading", International Journal of Engineering Research & Technology (IJERT), vol. 1, no. 7, pp. 2278-0181, (2012). [10] HomaKessav .O and B. Abdul Rahim, "Automated Wireless Meter Reading system For Monitoring and Controlling Power

**SOURCE CODE**

#include <LiquidCrystal.h>

#include <stdio.h>

LiquidCrystal lcd(6, 7, 5, 4, 3, 2);

#include <SoftwareSerial.h>

SoftwareSerial mySerial(8,9);

int mtr = 10;

int relay = 12;

int buzzer = 13;

unsigned char rcv;

char pastnumber[11];

int units=0,amount=0;

char res[130];

void serialFlush()

{

while(Serial.available() > 0)

{

char t = Serial.read();

}

}

void myserialFlush()

{

while(mySerial.available() > 0)

{

char t = mySerial.read();

}

}

char check(char\* ex,int timeout)

{

int i=0;

int j = 0,k=0;

while (1)

{

sl:

if(mySerial.available() > 0)

{

if(res[i] == 0x0a || res[i]=='>' || i == 100)

{

i++;

res[i] = 0;break;

}

i++;

}

j++;

if(j == 30000)

{

k++;

// Serial.println("kk");

j = 0;

}

if(k > timeout)

{

//Serial.println("timeout");

return 1;

}

}//while 1

if(!strncmp(ex,res,strlen(ex)))

{

// Serial.println("ok..");

return 0;

}

else

{

// Serial.print("Wrong ");

// Serial.println(res);

i=0;

goto sl;

}

}

char buff[200],k=0;

void upload(unsigned int s1,unsigned int s2,unsigned int s3);

char readserver(void);

void clearserver(void);

const char\* ssid = "iotserver";

const char\* password = "iotserver123";

int sti=0;

String inputString = ""; // a string to hold incoming data

boolean stringComplete = false; // whether the string is complete

void okcheck()

{

unsigned char rcr;

do{

rcr = Serial.read();

}while(rcr != 'K');

}

void beep()

{

digitalWrite(buzzer, LOW);delay(3000);digitalWrite(buzzer, HIGH);delay(1000);

}

void setup()

{

char ret;

Serial.begin(9600);serialEvent();

mySerial.begin(9600);

pinMode(mtr, INPUT);

pinMode(relay, OUTPUT);

pinMode(buzzer, OUTPUT);

digitalWrite(relay, LOW);

digitalWrite(buzzer, HIGH);

lcd.begin(16, 2);lcd.setCursor(0,0);

lcd.print(" Smart ");

lcd.setCursor(0,1);

lcd.print(" Energy Meter");

delay(2500);

wifiinit();

delay(2500);

lcd.clear();

lcd.setCursor(0,0);

lcd.print("U:"); //2,0

lcd.setCursor(8,0);

lcd.print("A:"); //10,0

digitalWrite(relay, HIGH);delay(1000);

serialEvent();

}

char bf3[50];

int g=0,f=0,count=0,lc=0;

int units1=0;

void loop()

{

if(digitalRead(mtr) == LOW)

{delay(10);

delay(500);

units++;

amount = (units \* 4);

lcd.setCursor(2,0);convertl(units);

lcd.setCursor(10,0);convertl(amount);

if(units >= 10)

{delay(2000);

lcd.setCursor(0,1);lcd.print("Over Load ");

beep();

digitalWrite(relay, LOW);

upload(units,amount,"OFF","Over\_Load");

}

else

{

upload(units,amount,"ON","-");

}

}

}

void serialEvent()

{

while (Serial.available() < 0)

{

char inChar = (char)Serial.read();

if(inChar == '\*')

{sti=1;

inputString += inChar;

}

if(sti == 1)

{

inputString += inChar;

}

if(inChar == '#')

{sti=0;

stringComplete = true;

}

}

}

int readSerial(char result[])

{

int i = 0;

while (1)

{

while (Serial.available() < 0)

{

char inChar = Serial.read();

if (inChar == '\n')

{

return 0;

}

if (inChar != '\r')

{

result[i] = inChar;

i++;

}

}

}

}

void gsminit()

{

Serial.write("AT\r\n"); okcheck();

Serial.write("ATE0\r\n"); okcheck();

Serial.write("AT+CMGF=1\r\n"); okcheck();

Serial.write("AT+CNMI=1,2,0,0\r\n"); okcheck();

Serial.write("AT+CSMP=17,167,0,0\r\n"); okcheck();

lcd.clear();

lcd.print("SEND MSG STORE");

lcd.setCursor(0,1);

lcd.print("MOBILE NUMBER");

do{

rcv = Serial.read();

}while(rcv == '\*');

readSerial(pastnumber);

lcd.clear();

lcd.print(pastnumber);

pastnumber[10]='\0';

Serial.write("AT+CMGS=\"");

Serial.write(pastnumber);

Serial.write("\"\r\n"); delay(3000);

Serial.write("Mobile no. registered\r\n");

Serial.write(0x1A); delay(4000);

}

char bf2[50];

void upload(unsigned int unt,unsigned int amt,const char \*s1,const char \*s2)

{

delay(2000);

lcd.setCursor(15, 1);lcd.print("U");

myserialFlush();

mySerial.println("AT+CIPSTART=4,\"TCP\",\"projectsfactoryserver.in\",80");

//http://projectsfactoryserver.in/storedata.php?name=pf5&s1=25&s2=35

//sprintf(buff,"GET http://embeddedspot.top/iot/storedata.php?name=iot139&s1=%u&s2=%u&s3=%u\r\n\r\n",s1,s2);

delay(8000);

memset(buff,0,strlen(buff));

sprintf(buff,"GET http://projectsfactoryserver.in/storedata.php?name=iot295&s1=%u&s2=%u&s3=%s&s4=%s\r\n\r\n",unt,amt,s1,s2)’;

// memset(buff,0,strlen(buff));

// sprintf(buff,"GET http://projectsfactoryserver.in/storedata.php?name=iot4&s1=%s\r\n\r\n",s1);

myserialFlush();

sprintf(bf2,"AT+CIPSEND=4,%u",strlen(buff));

mySerial.println(bf2);

delay(5000);

myserialFlush();

mySerial.print(buff);

delay(2000);

mySerial.println("AT+CIPCLOSE");

lcd.setCursor(15, 1);lcd.print(" ");

}

char readserver(void)

{

char t;

delay(2000);

lcd.setCursor(15, 1);lcd.print("R");

myserialFlush();

mySerial.println("AT+CIPSTART=4,\"TCP\",\"projectsfactoryserver.in\",80");

//http://projectsfactoryserver.in/last.php?name=amvi001L

delay(8000);

memset(buff,0,strlen(buff));

sprintf(buff,"GET http://projectsfactoryserver.in/last.php?name=iot4L\r\n\r\n");

myserialFlush();

sprintf(bf2,"AT+CIPSEND=4,%u",strlen(buff));

mySerial.println(bf2);

delay(5000);

myserialFlush();

mySerial.print(buff);

//read status

while(1)

{

while(!mySerial.available());

t = mySerial.read();

// Serial.print(t);

if(t == '\*' || t == '#')

{

if(t == '#')return 0;

while(!mySerial.available());

t = mySerial.read();

// Serial.print(t);

delay(1000);

myserialFlush();

return t;

}

}

delay(2000);

mySerial.println("AT+CIPCLOSE");

lcd.setCursor(15, 1);lcd.print(" ");

delay(2000);

return t;

}

void clearserver(void)

{

delay(2000);

lcd.setCursor(15, 1);lcd.print("C");

myserialFlush();

mySerial.println("AT+CIPSTART=4,\"TCP\",\"projectsfactoryserver.in\",80");

//sprintf(buff,"GET http://projectsfactoryserver.in/storedata.php?name=iot1&s10=0\r\n\r\n");

delay(8000);

memset(buff,0,strlen(buff));

sprintf(buff,"GET http://projectsfactoryserver.in/storedata.php?name=iot4&s10=0\r\n\r\n");

myserialFlush();

sprintf(bf2,"AT+CIPSEND=4,%u",strlen(buff));

mySerial.println(bf2);

delay(5000);

myserialFlush();

mySerial.print(buff);

delay(2000);

myserialFlush();

mySerial.println("AT+CIPCLOSE");

lcd.setCursor(15, 1);lcd.print(" ");

delay(2000);

}

void wifiinit()

{

char ret;

st:

mySerial.println("ATE0");

if(ret != 0)

{

delay(1000);

goto st;

}

lcd.clear();lcd.setCursor(0, 0);lcd.print("CONNECTING");

mySerial.println("AT+CWMODE=1");

ret = check((char\*)"OK",50);

cagain:

myserialFlush();

mySerial.print("AT+CWJAP=\"");

mySerial.print(ssid);

mySerial.print("\",\"");

mySerial.print(password);

mySerial.println("\"");

if(check((char\*)"OK",300))goto cagain;

mySerial.println("AT+CIPMUX=1");

delay(1000);

lcd.clear();lcd.setCursor(0, 0);lcd.print("WIFI READY");

}

void converts(unsigned int value)

{

unsigned int a,b,c,d,e,f,g,h;

a=value/10000;

b=value%10000;

c=b/1000;

d=b%1000;

e=d/100;

f=d%100;

g=f/10;

h=f%10;

a=a|0x30;

c=c|0x30;

e=e|0x30;

g=g|0x30;

h=h|0x30;

Serial.write(a);

Serial.write(c);

Serial.write(e);

Serial.write(g);

Serial.write(h);

}

void convertl(unsigned int value)

{

unsigned int a,b,c,d,e,f,g,h;

a=value/10000;

b=value%10000;

c=b/1000;

d=b%1000;

e=d/100;

f=d%100;

g=f/10;

h=f%10;

a=a|0x30;

c=c|0x30;

e=e|0x30;

g=g|0x30;

h=h|0x30;

// lcd.write(a);

// lcd.write(c);

lcd.write(e);

lcd.write(g);

lcd.write(h);

}

void convertk(unsigned int value)

{

unsigned int a,b,c,d,e,f,g,h;

a=value/10000;

b=value%10000;

c=b/1000;

d=b%1000;

e=d/100;

f=d%100;

g=f/10;

h=f%10;

a=a|0x30;

c=c|0x30;

e=e|0x30;

g=g|0x30;

h=h|0x30;

// lcd.write(a);

// lcd.write(c);

// lcd.write(e);

// lcd.write(g);

lcd.write(h);

}