Student Minor Research Project

IOT ENABLED HOME AUTOMATION SYSTEM

Under RUSA 2.0 Scheme

(Through Ch.S.D.St.Theresa's College for Women (Autonomous), Eluru, AP)

Submitted by

Mr K Mutyala Swamy , III B.Sc. MECs (Reg.No.11705051)

Mr D Leela Krishna, III B.Sc. MPE (Reg.No.11704009)

Mr T Ravi Teja, III B.Sc. MPE (Reg.No.11704033)

Under the guidance of

Mr K Vinaya Phaneendhra

Lecturer & Project Advisor

Department Of Electronics

SRI Y N COLLEGE

(AUTONOMOUS)

Thrice Accredited by NAAC at 'A' Grade

Recognized by UGC as "College with Potential for Excellence"

Narsapur-534275, AP, India

December-2019

Department Of Electronics

SRI Y N COLLEGE

(AUTONOMOUS)

Thrice Accredited by NAAC at 'A' Grade

Recognized by UGC as "College with Potential for Excellence"

Narsapur-534275, AP, India

CERTIFICATE

This is to certify that the project work entitled " **IoT Enabled Home Automation System** *" is bonafied work carried out by Mr K Mutyala Swamy (Reg.No: 11705051), Mr D Leela Krishna (Reg.No: 11704009), Mr T Ravi Teja (Reg.No: 11704033), submitted in Third Year of the degree B.Sc. in Electronics during the year 2019-20 is an authentic work under my supervision and guidance.*

To the best of my knowledge, the matter embodied in the project work has not been submitted to any other College/Institute.

Date: 29-12-2019

Mr K Vinaya Phaneendhra
Project Advisor
Department of Electronics

ACKNOWLEDGEMENT

We place on record and warmly acknowledge the continuous encouragement, invaluable supervision, timely suggestions and inspired guidance offered by our Project advisor, ***Mr K Vinaya Phaneendhra,*** *Lecturer, Department of Electronics,* ***Sri Y N College (Autonomous), Narsapur*** *in bringing this report to a successful completion.*

We are grateful to ***Dr K Venkateswarlu****, Head, Department of Electronics for permitting us to make use of the facilities available in the department to carry out the project successfully. Last but not the least we express our sincere thanks to all of our friends who have patiently extended all sorts of help for accomplishing this undertaking.*

Finally we extend our gratefulness to one and all who are directly or indirectly involved in the successful completion of this project work.

Mr. K Mutyala Swamy
III.B.Sc.MECs
Reg. No 11705051

Mr. D Leela Krishna
III.B.Sc.MPE
Reg. No.11704009

Mr. T Ravi Teja
III.B.Sc.MPE
Reg.No.11704033

DECLARATION

We, the undersigned, declare that the project entitled " **IoT Enabled Home Automation System** ", being submitted in Third Year of Bachelor of Science in Electronics, Sri Y N College (Autonomous), is the work carried out by us.

Mr. K Mutyala Swamy	**Mr. D Leela Krishna**	**Mr. T Ravi Teja**
III.B.Sc.MECs	III.B.Sc.MPE	III.B.Sc.MPE
Reg. No 11705051	Reg. No.11704009	Reg.No.11704033

Contents **Page No.**

List of Figures & Tables — Page No.

1. ABSTRACT

IoT or Internet of Things is an upcoming technology that allows us to control hardware devices through the internet. Here we propose to use IOT in order to control home appliances, thus automating modern homes through the internet. This system uses three loads to demonstrate as house lighting and a fan. Our user friendly interface allows a user to easily control these home appliances through the internet. For this system we use an AVR family microcontroller.

This microcontroller is interfaced with a WIFI modem to get user commands over the internet. Also we have an LCD display to display system status. Relays are used to switch loads. The entire system is powered by a 12 V transformer. After receiving user commands over the internet, microcontroller processes these instructions to operate these loads accordingly and display the system status on an LCD display. Thus this system allows for efficient home automation over the internet.

2. INTRODUCTION TO INTERNET OF THINGS (IoT)

The term *Internet of Things* (often abbreviated *IoT)* was coined more than ten years ago by industry researchers but has emerged into mainstream public view only more recently. Some claim the Internet of Things will completely transform how computer networks are used for the next 10 or 100 years, while others believe IoT is simply hype that won't much impact the daily lives of most people.

What Is IoT?

Internet of Things represents a general concept for the ability of network devices to sense and collect data from the world around us, and then share that data across the Internet where it can be processed and utilized for various interesting purposes.

Some also use the term *industrial Internet* interchangeably with IoT. This refers primarily to commercial applications of IoT technology in the world of manufacturing. The Internet of Things is not limited to industrial applications, however.

What the Internet of Things Can Do for Us

Some future consumer applications envisioned for IoT sound like science fiction, but some of the more practical and realistic sounding possibilities for the technology include:

- receiving warnings on your phone or wearable device when IoT networks detect some physical danger is detected nearby
- self-parking automobiles
- automatic ordering of groceries and other home supplies
- automatic tracking of exercise habits and other day-to-day personal activity including goal tracking and regular progress reports

Potential benefits of IoT in the business world include:

- location tracking for individual pieces of manufacturing inventory
- fuel savings from intelligent environmental modeling of gas-powered engines
- new and improved safety controls for people working in hazardous environments

Network Devices and the Internet of Things

All kinds of ordinary household gadgets can be modified to work in an IoT system. Wi-Fi network adapters, motion sensors, cameras, microphones and other instrumentation can be embedded in these devices to enable them for work in the Internet of Things. Home automation systems already implement primitive versions of this concept for things like light bulbs, plus other devices like wireless scales and wireless blood pressure monitorsthat each represent early examples of IoT gadgets. Wearable computing devices like watches and glasses are also envisioned to be key components in future IoT systems.

The same wireless communication protocols like Wi-Fi and Bluetooth naturally extend to the Internet of Things also.

Issues around IoT

Internet of Things immediately triggers questions around the privacy of personal data. Whether real-time information about our physical location, or updates about our weight and blood pressure that may be accessible by our health care providers, having new kinds and more detailed data about ourselves streaming over wireless networks and potentially around the world is an obvious concern.

Supplying power to this new proliferation of IoT devices and their network connections can be expensive and logistically difficult. Portable devices require batteries that someday must be replaced. Although many mobile devices are optimized for lower power usage, energy costs to keep potentially billions of them running remains high.

Numerous corporations and start-up ventures have latched onto the Internet of Things concept looking to take advantage of whatever business opportunities are available. While competition in the market helps lower prices of consumer products, in the worst case it also leads to confusing and inflated claims about what the products do.

IoT assumes that the underlying network equipment and related technology can operate semi-intelligently and often automatically. Simply keeping mobile devices connected to the Internet can be difficult enough much less trying to make them smarter. People have diverse needs that require an IoT system to adapt or be configurable for many different situations and preferences. Finally, even with all those challenges overcome, if people become too reliant on this automation and the technology is not highly robust, any technical glitches in the system can cause serious physical and/or financial damage.

3. INTRODUCTION TO EMBEDDED SYSTEMS

What is Embedded System?

An Embedded System is a combination of computer hardware and software, and perhaps additional mechanical or other parts, designed to perform a specific function. An embedded system is a microcontroller-based, software driven, reliable, real-time control system, autonomous, or human or network interactive, operating on diverse physical variables and in diverse environments and sold into a competitive and cost conscious market.

An embedded system is not a computer system that is used primarily for processing, not a software system on PC or UNIX, not a traditional business or scientific application. High-end embedded & lower end embedded systems. High-end embedded system - Generally 32, 64 Bit Controllers used with OS. Examples Personal Digital Assistant and Mobile phones etc .Lower end embedded systems - Generally 8,16 Bit Controllers used with an minimal operating systems and hardware layout designed for the specific purpose.

SYSTEM DESIGN CALLS:

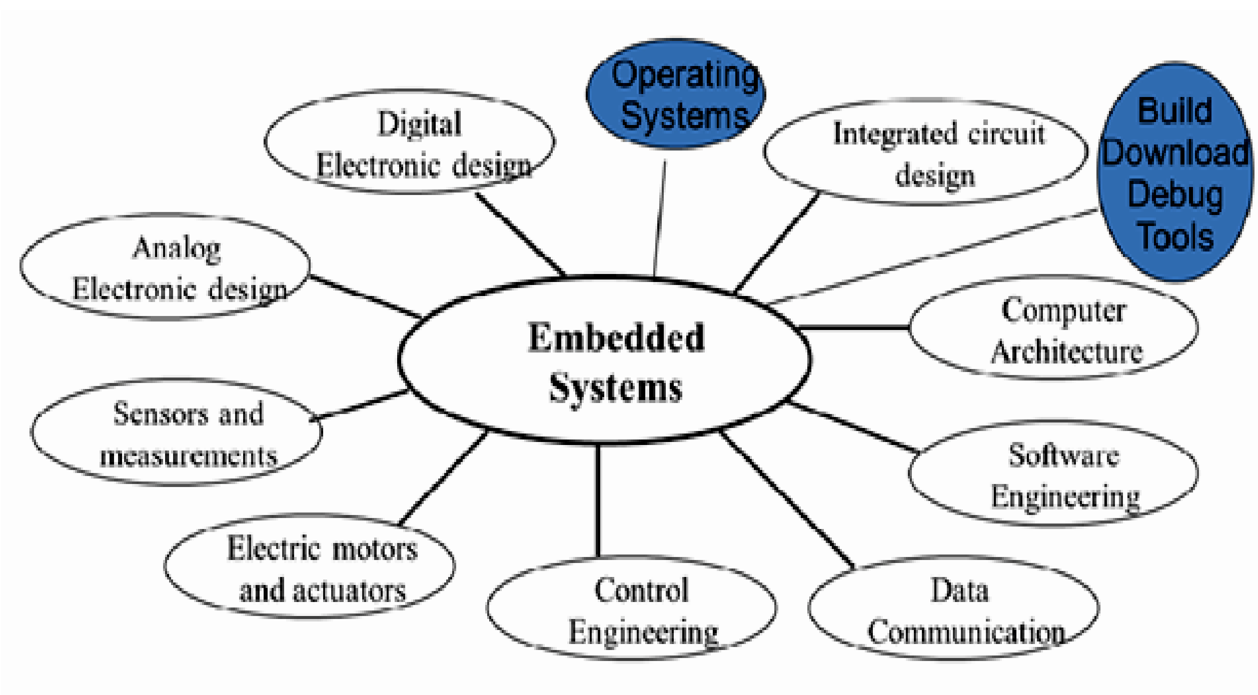


Fig 3(a): Embedded System Design calls on many disciplines

EMBEDDED SYSTEM DESIGN CYCLE

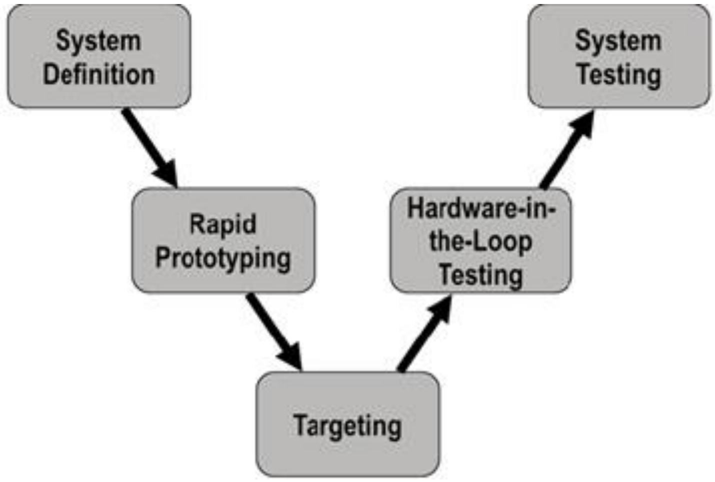


Fig:3(b) “V Diagram”(Embedded System Design Life Cycle)

Characteristics of Embedded System

- An embedded system is any computer system hidden inside a product other than a computer.
- They will encounter a number of difficulties when writing embedded system software in addition to those we encounter when we write applications.
  - Throughput – Our system may need to handle a lot of data in a short period of time.
  - Response–Our system may need to react to events quickly
  - Testability–Setting up equipment to test embedded software can be difficult.
  - Debugability–Without a screen or a keyboard, finding out what the software is doing wrong (other than not working) is a troublesome problem.
  - Reliability – embedded systems must be able to handle any situation without human intervention.
  - Memory space – Memory is limited on embedded systems, and you must make the software and the data fit into whatever memory exists.

– Program installation – you will need special tools to get your software into embedded systems.
– Power consumption – Portable systems must run on battery power, and the software in these systems must conserve power.
– Processor hogs – computing that requires large amounts of CPU time can complicate the response problem.
– Cost – Reducing the cost of the hardware is a concern in many embedded system projects; software often operates on hardware that is barely adequate for the job.

- Embedded systems have a microprocessor/ microcontroller and a memory. Some have a serial port or a network connection. They usually do not have keyboards, screens or disk drives.

APPLICATIONS

1) Military and aerospace embedded software applications
2) Communication Applications
3) Industrial automation and process control software
4) Mastering the complexity of applications.
5) Reduction of product design time.
6) Real time processing of ever increasing amounts of data.
7) Intelligent, autonomous sensors.

CLASSIFICATION

- Real Time Systems.
- RTS is one which has to respond to events within a specified deadline.
- A right answer after the dead line is a wrong answer.

RTS CLASSIFICATION

- Hard Real Time Systems
- Soft Real Time System

HARD REAL TIME SYSTEM

- "Hard" real-time systems have very narrow response time.

- Example: Nuclear power system, Cardiac pacemaker.

SOFT REAL TIME SYSTEM

- "Soft" real-time systems have reduced constrains on "lateness" but still must operate very quickly and repeatable.
- Example: Railway reservation system – takes a few extra seconds the data remains valid.

4. BLOCK DIAGRAM OF PROJECT

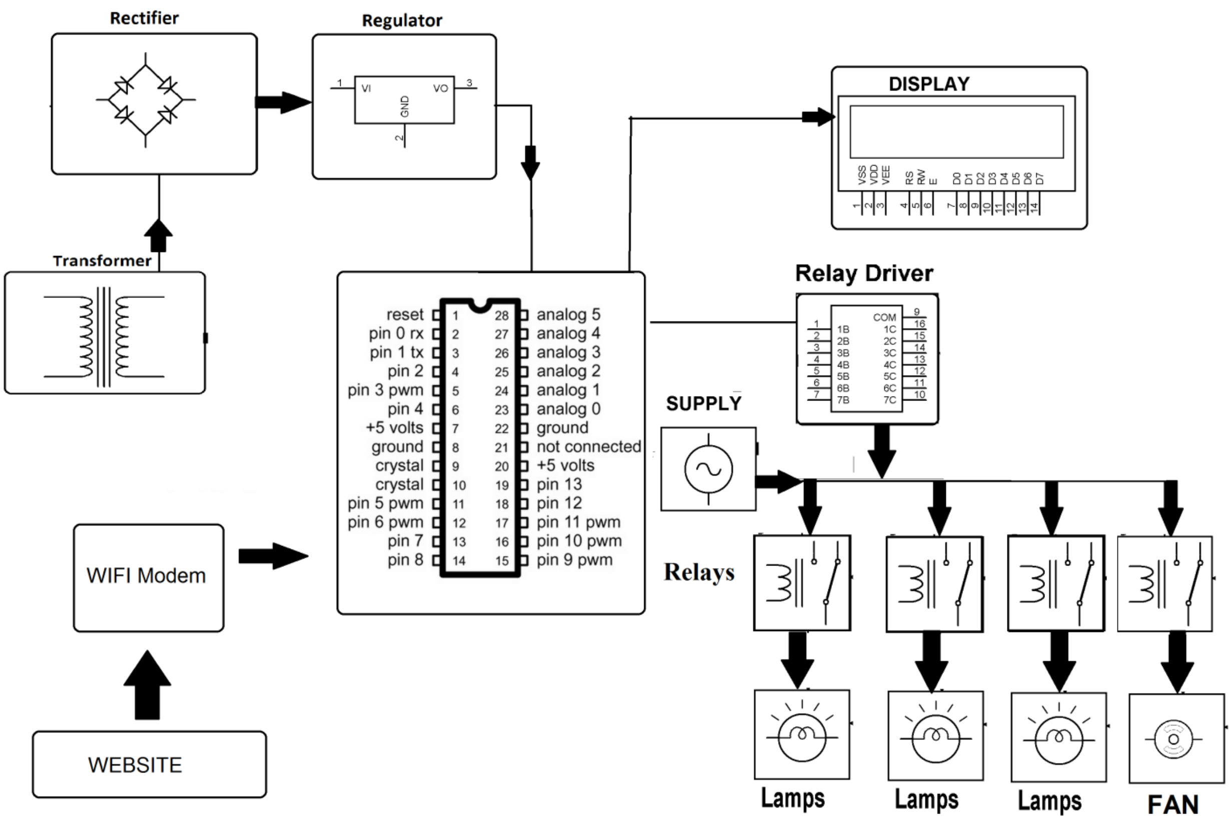


Fig (4) Block Diagram of Project

5. HARDWARE REQUIREMENTS

HARDWARE COMPONENT:

1. TRANSFORMER
2. BRIDGE RECTIFIER
3. FILTER
4. VOLTAGE REGULATOR (7805)
5. AVR FAMILY MICROCONTROLLER
6. WIFI MODULE
7. LCD
8. LEDs
9. RELAYS
10. RELAY DRIVER
11. LED BULBS
12. MOTOR/FAN
13. CRYSTAL
14. RESISTORS
15. CAPACITOR S
16. PUSH BUTTON

5.1. TRANSFORMERS

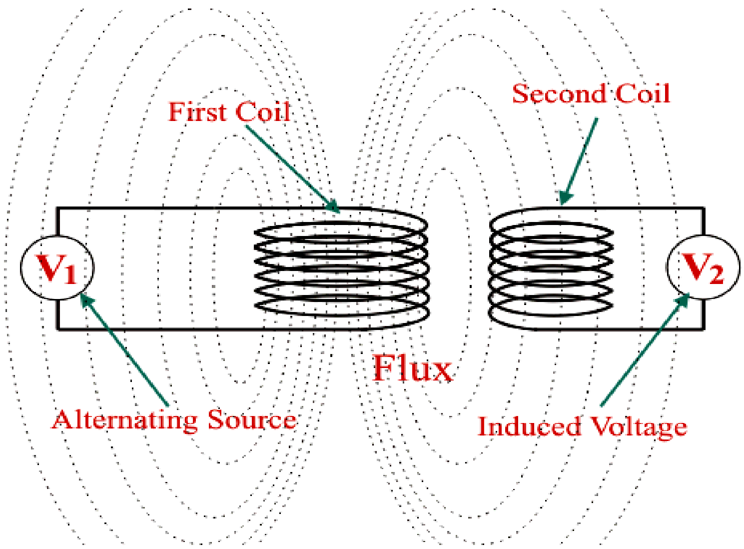
WORKING PRINCIPLE OF TRANSFORMER

The **working principle of a transformer** is very simple. Mutual induction between two or more windings (also known as coils) allows for electrical energy to be transferred between circuits. This principle is explained in further detail below.

Transformer Theory

We have one winding (also known as a coil) which is supplied by an alternating electrical source. The alternating current through the winding produces a continually changing and alternating flux that surrounds the winding. If another winding is brought close to this winding, some portion of this alternating flux will link with the second winding. As this flux is continually changing in its amplitude and direction, there must be a changing flux linkage in the second winding or coil.

According to Faraday's law of electromagnetic induction, there will be an EMF induced in the second winding. If the circuit of this secondary winding is closed, then a current will flow through it. This is the basic **working principle of a transformer**. Let us use electrical symbols to help visualize this. The winding which receives electrical power from the source is known as the 'primary winding'. In the diagram below this is the 'First Coil'.



The winding which gives the desired output voltage due to mutual induction is commonly known as the ‘secondary winding’. This is the ‘Second Coil’ in the diagram above.

A transformer that increases voltage between the primary to secondary windings is defined as a step-up transformer. Conversely, a transformer that decreases voltage between the primary to secondary windings is defined as a step-down transformer.

While the diagram of the transformer above is theoretically possible in an ideal transformer – it is not very practical. This is because in open air only a very tiny portion of the flux produced from the first coil will link with the second coil. So the current that flows through the closed circuit connected to the secondary winding will be extremely small (and difficult to measure).

The rate of change of flux linkage depends upon the amount of linked flux with the second winding. So ideally almost all of the flux of primary winding should link to the secondary winding. This is effectively and efficiently done by using a core type transformer. This provides a low reluctance path common to both of the windings.

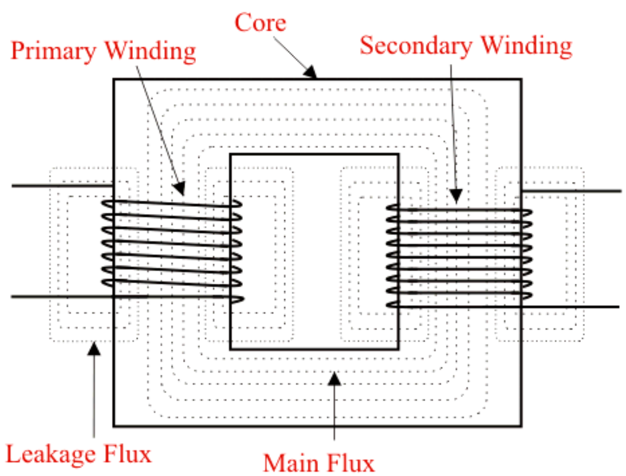


Fig 5.1(a) Transformer

The purpose of the transformer core is to provide a low reluctance path, through which the maximum amount of flux produced by the primary winding is passed through and linked with the secondary winding.

The current that intially passes through the transformer when it is switched on is known as the transformer inrush current.

5.2 VOLTAGE REGULATOR LM7805

Features

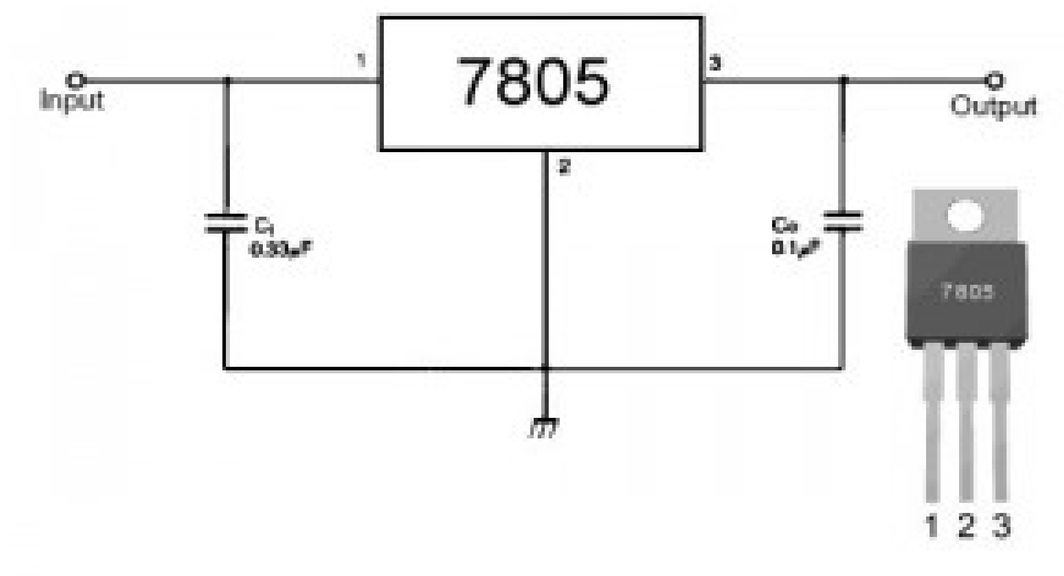
• Output Current up to 1A.

• Output Voltages of 5, 6, 8, 9, 10, 12, 15, 18, 24V.

• Thermal Overload Protection.

• Short Circuit Protection.

• Output Transistor Safe Operating Area Protection.



Description

The LM78XX/LM78XXA series of three-terminal positive regulators are available in the TO-220/D-PAK package and with several fixed output voltages, making them useful in a Wide range of applications. Each type employs internal current limiting, thermal shutdown and safe operating area protection, making it essentially indestructible. If adequate heat sinking is provided, they can deliver over 1A output Current. Although designed primarily as fixed voltage regulators, these devices can be used with external components to obtain adjustable voltages and currents.

Internal Block Diagram

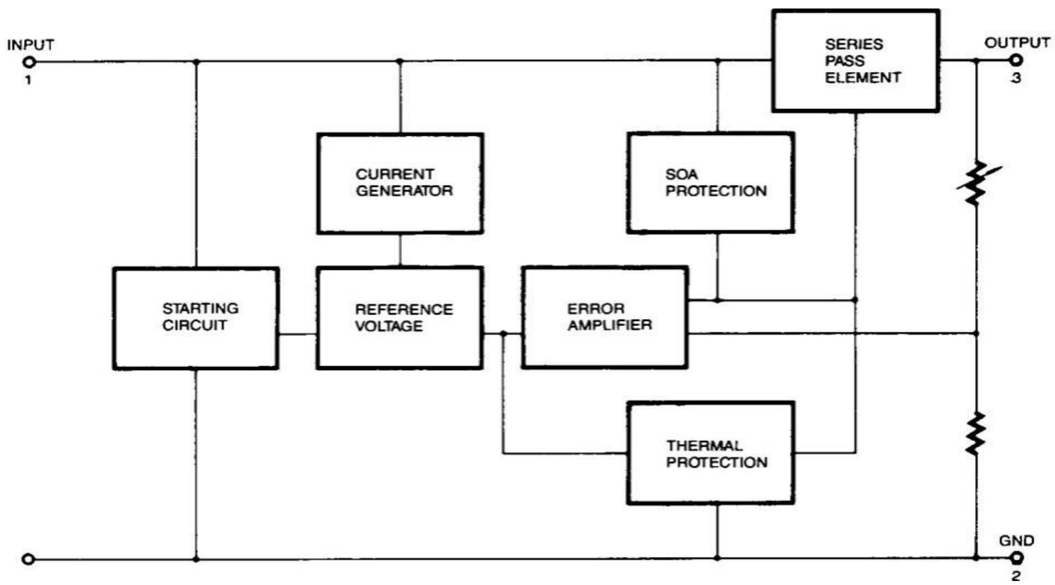


FIG 5.2(a): BLOCK DIAGRAM OF VOLTAGE REGULATOR

Absolute Maximum Ratings

Parameter	Symbol	Value	Unit
Input Voltage (for V_O = 5V to 18V) (for V_O = 24V)	V_I V_I	35 40	V V
Thermal Resistance Junction-Cases (TO-220)	$R_{\theta JC}$	5	°C/W
Thermal Resistance Junction-Air (TO-220)	$R_{\theta JA}$	65	°C/W
Operating Temperature Range (KA78XX/A/R)	T_{OPR}	0 ~ +125	°C
Storage Temperature Range	T_{STG}	-65 ~ +150	°C

TABLE 5.2(b): RATINGS OF THE VOLTAGE REGULATOR

5.3 IN4007

Diodes are used to convert AC into DC these are used as half wave rectifier or full wave rectifier. Three points must he kept in mind while using any type of diode.

1.Maximum forward current capacity

2.Maximum reverse voltage capacity

3.Maximum forward voltage capacity

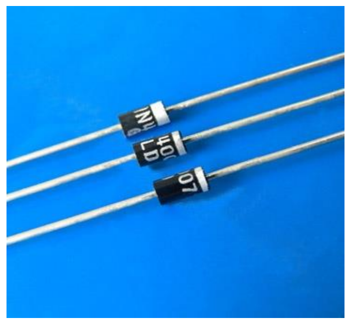


Fig5.3 (a): 1N4007 diodes

The number and voltage capacity of some of the important diodes available in the market are as follows:

- Diodes of number IN4001, IN4002, IN4003, IN4004, IN4005, IN4006 and IN4007 have maximum reverse bias voltage capacity of 50V and maximum forward current capacity of 1 Amp.
- Diode of same capacities can be used in place of one another. Besides this diode of more capacity can be used in place of diode of low capacity but diode of low capacity cannot be used in place of diode of high capacity. For example, in place of IN4002; IN4001 or IN4007 can be used but IN4001 or IN4002 cannot be used in place of IN4007.The diode BY125made by company BEL is equivalent of diode from IN4001 to IN4003. BY 126 is equivalent to diodes IN4004 to 4006 and BY 127 is equivalent to diode IN4007.

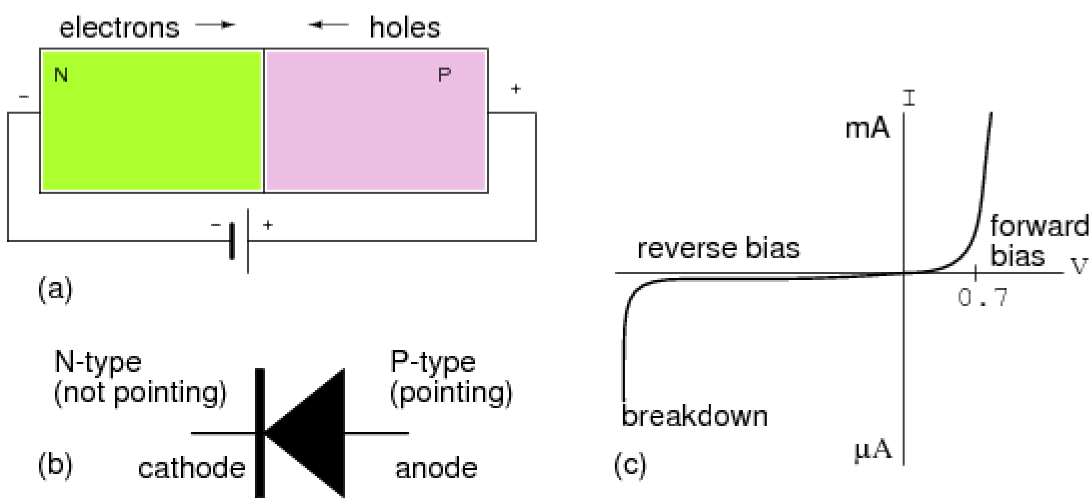


Fig 5.3(b):PN Junction diode

PN JUNCTION OPERATION

Now that you are familiar with P- and N-type materials, how these materials are joined together toform a diode, and the function of the diode, let us continue our discussion with the operation of the PNjunction. But before we can understand how the PN junction works, we must first consider current flow inthe materials that make up the junction and what happens initially within the junction when these twomaterials are joined together.

Current Flow in the N-Type Material

Conduction in the N-type semiconductor, or crystal, is similar to conduction in a copper wire. Thatis, with voltage applied across the material, electrons will move through the crystal just as current wouldflow in a copper wire. This is shown in figure 1-15. The positive potential of the battery will attract thefree electrons in the crystal. These electrons will leave the crystal and flow into the positive terminal ofthe battery. As an electron leaves the crystal, an electron from the negative terminal of the battery willenter the crystal, thus completing the current path. Therefore, the majority current carriers in the N-typematerial (electrons) are repelled by the negative side of the battery and move through the crystal towardthe positive side of the battery.

Current Flow in the P-Type Material

Current flow through the P-type material is illustrated. Conduction in the P material isby positive holes, instead of negative electrons. A hole moves from the positive terminal of the P materialto the negative terminal. Electrons from the external circuit enter the negative terminal of the material andfill holes in the vicinity of this terminal. At the positive terminal, electrons are removed from the covalentbonds, thus creating new holes. This process continues as the steady stream of holes (hole current) movestoward the negative terminal

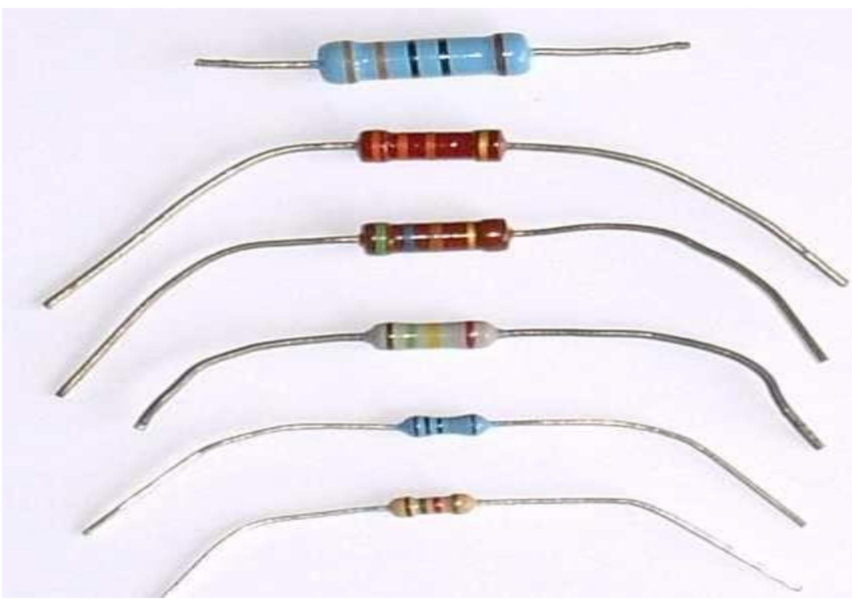
5.4 RESISTORS

A resistor is a two-terminal electronic component designed to oppose an electric current by producing a voltage drop between its terminals in proportion to the current, that is, in accordance with Ohm's law:

$$V = IR$$

Resistors are used as part of electrical networks and electronic circuits. They are extremely commonplace in most electronic equipment. Practical resistors can be made of various compounds and films, as well as resistance wire (wire made of a high-resistivity alloy, such as nickel/chrome).

The primary characteristics of resistors are their resistance and the power they can dissipate. Other characteristics include temperature coefficient, noise, and inductance. Less well-known is critical resistance, the value below which power dissipation limits the maximum permitted current flow, and above which the limit is applied voltage. Critical resistance depends upon the materials constituting the resistor as well as its physical dimensions; it's determined by design.
Resistors can be integrated into hybrid and printed circuits, as well as integrated circuits. Size, and position of leads (or terminals) are relevant to equipment designers; resistors must be physically large enough not to overheat when dissipating their power.



A resistor is a two-terminalpassiveelectronic component which implements electrical resistance as a circuit element. When a voltage V is applied across the terminals of a resistor, a current I will flow through the resistor in direct proportion to that voltage. The reciprocal of the constant of proportionality is known as the resistance R, since, with a given voltage V, a larger value of R further "resists" the flow of current I as given by Ohm's law:

$$I = \frac{V}{R}$$

Resistors are common elements of electrical networks and electronic circuits and are ubiquitous in most electronic equipment. Practical resistors can be made of various compounds and films, as well as resistance wire (wire made of a high-resistivity alloy, such as nickel-chrome). Resistors are also implemented within integrated circuits, particularly analog devices, and can also be integrated into hybrid and printed circuits.

The electrical functionality of a resistor is specified by its resistance: common commercial resistors are manufactured over a range of more than 9 orders of magnitude. When specifying that resistance in an electronic design, the required precision of the resistance may require attention to the manufacturing tolerance of the chosen resistor, according to its specific application. The temperature coefficient of the resistance may also be of concern in some precision applications. Practical resistors are also specified as having a maximum power rating which must exceed the anticipated power dissipation of that resistor in a particular circuit: this is mainly of concern in power electronics applications. Resistors with higher power ratings are physically larger and may require heat sinking. In a high voltage circuit, attention must sometimes be paid to the rated maximum working voltage of the resistor.

The series inductance of a practical resistor causes its behavior to depart from ohms law; this specification can be important in some high-frequency applications for smaller values of resistance. In a low-noise amplifier or pre-amp the noise characteristics of a resistor may be an issue. The unwanted inductance, excess noise, and temperature coefficient are mainly dependent on the technology used in manufacturing the resistor. They are not normally specified individually for a particular family of resistors manufactured using a particular technology. A family of discrete resistors is also characterized according to its form factor, that is, the size of the device and position of its leads (or terminals) which is relevant in the practical manufacturing of circuits using them.

Units

The ohm (symbol: Ω) is the SI unit of electrical resistance, named after Georg Simon Ohm. An ohm is equivalent to a volt per ampere. Since resistors are specified and manufactured over a very large range of values, the derived units of milliohm (1 mΩ = 10^{-3} Ω), kilohm (1 kΩ = 10^3 Ω), and megohm (1 MΩ = 10^6 Ω) are also in common usage.

The reciprocal of resistance R is called conductance G = 1/R and is measured in Siemens (SI unit), sometimes referred to as a mho. Thus a Siemens is the reciprocal of an ohm: $S = \Omega^{-1}$. Although the concept of conductance is often used in circuit analysis, practical resistors are always specified in terms of their resistance (ohms) rather than conductance.

Theory of operation

Ohm's law

The behaviour of an ideal resistor is dictated by the relationship specified in Ohm's law:

$$V = I \cdot R$$

Ohm's law states that the voltage (V) across a resistor is proportional to the current (I) passing through it, where the constant of proportionality is the resistance (R).

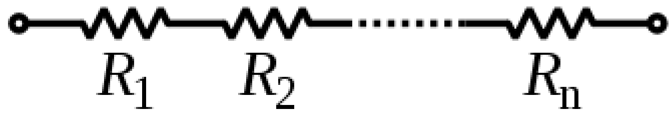
Equivalently, Ohm's law can be stated:

$$I = \frac{V}{R}$$

This formulation of Ohm's law states that, when a voltage (V) is present across a resistance (R), a current (I) will flow through the resistance. This is directly used in practical computations. For example, if a 300 ohm resistor is attached across the terminals of a 12 volt battery, then a current of 12 / 300 = 0.04 amperes (or 40 mill amperes) will flow through that resistor.

Series and parallel resistors

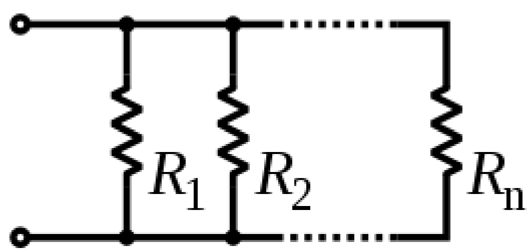
In a series configuration, the current through all of the resistors is the same, but the voltage across each resistor will be in proportion to its resistance. The potential difference (voltage) seen across the network is the sum of those voltages, thus the total resistance can be found as the sum of those resistances:



$$R_{\mathrm{eq}} = R_1 + R_2 + \cdots + R_n$$

As a special case, the resistance of N resistors connected in series, each of the same resistance R, is given by NR.

Resistors in a parallel configuration are each subject to the same potential difference (voltage), however the currents through them add. The conductance of the resistors then add to determine the conductance of the network. Thus the equivalent resistance (R_{eq}) of the network can be computed:



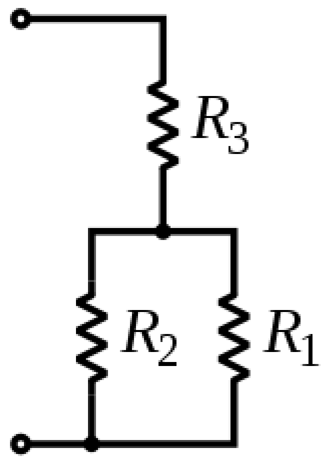
$$\frac{1}{R_{\mathrm{eq}}} = \frac{1}{R_1} + \frac{1}{R_2} + \cdots + \frac{1}{R_n}$$

The parallel equivalent resistance can be represented in equations by two vertical lines "||" (as in geometry) as a simplified notation. For the case of two resistors in parallel, this can be calculated using:

$$R_{\mathrm{eq}} = R_1 \| R_2 = \frac{R_1 R_2}{R_1 + R_2}$$

As a special case, the resistance of N resistors connected in parallel, each of the same resistance R, is given by R/N.

A resistor network that is a combination of parallel and series connections can be broken up into smaller parts that are either one or the other. For instance,

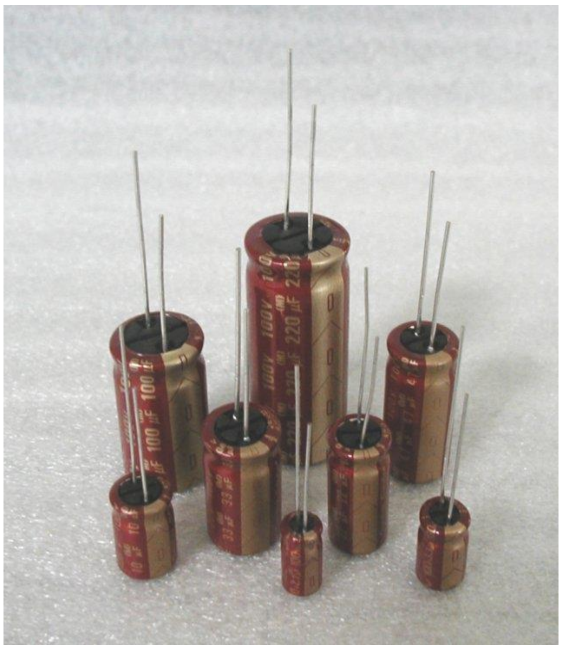


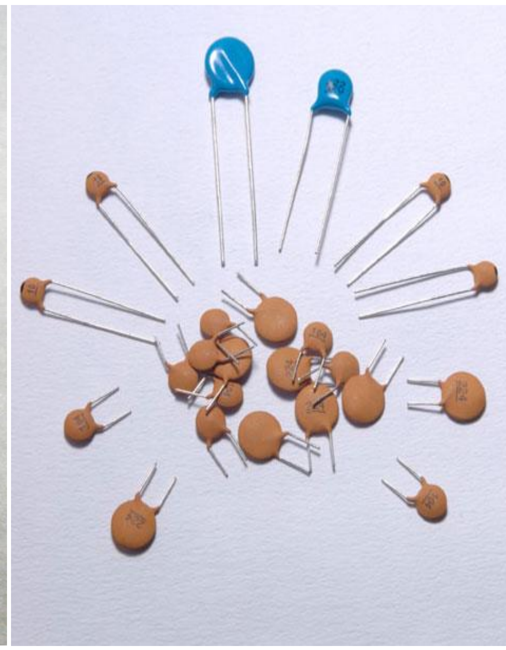
$$R_{\mathrm{eq}} = (R_1 \| R_2) + R_3 = \frac{R_1 R_2}{R_1 + R_2} + R_3$$

However, some complex networks of resistors cannot be resolved in this manner, requiring more sophisticated circuit analysis. For instance, consider a cube, each edge of which has been replaced by a resistor. What then is the resistance that would be measured between two opposite vertices? In the case of 12 equivalent resistors, it can be shown that the corner-to-corner resistance is ⅚ of the individual resistance. More generally, the Y-Δ transform, or matrix methods can be used to solve such a problem. One practical application of these relationships is that a non-standard value of resistance can generally be synthesized by connecting a number of standard values in series and/or parallel. This can also be used to obtain a resistance with a higher power rating than that of the individual resistors used. In the special case of N identical resistors all connected in series or all connected in parallel, the power rating of the individual resistors is thereby multiplied by N.

5.5 CAPACITORS

A capacitor or condenser is a passive electronic component consisting of a pair of conductors separated by a dielectric. When a voltage potential difference exists between the conductors, an electric field is present in the dielectric. This field stores energy and produces a mechanical force between the plates. The effect is greatest between wide, flat, parallel, narrowly separated conductors.





An ideal capacitor is characterized by a single constant value, capacitance, which is measured in farads. This is the ratio of the electric charge on each conductor to the potential difference between them. In practice, the dielectric between the plates passes a small amount of leakage current. The conductors and leads introduce an equivalent series resistance and the dielectric has an electric field strength limit resulting in a breakdown voltage.

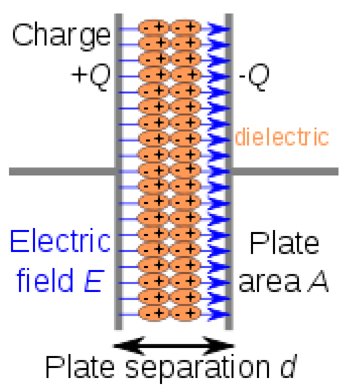
The properties of capacitors in a circuit may determine the resonant frequency and quality factor of a resonant circuit, power dissipation and operating frequency in a digital logic circuit, energy capacity in a high-power system, and many other important aspects.

A capacitor (formerly known as condenser) is a device for storing electric charge. The forms of practical capacitors vary widely, but all contain at least two conductors separated by a non-conductor. Capacitors used as parts of electrical systems, for example, consist of metal foils separated by a layer of insulating film.

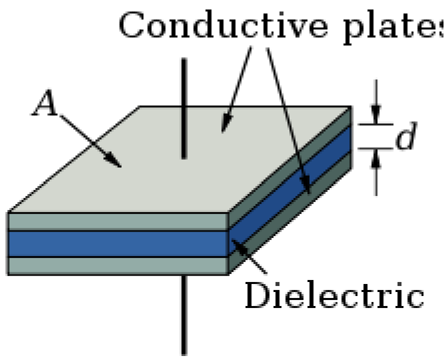
Capacitors are widely used in electronic circuits for blocking direct current while allowing alternating current to pass, in filter networks, for smoothing the output of power supplies, in the resonant circuits that tune radios to particular frequencies and for many other purposes.

A capacitor is a passiveelectronic component consisting of a pair of conductors separated by a dielectric (insulator). When there is a potential difference (voltage) across the conductors, a static electric field develops in the dielectric that stores energy and produces a mechanical force between the conductors. An ideal capacitor is characterized by a single constant value, capacitance, measured in farads. This is the ratio of the electric charge on each conductor to the potential difference between them.

The capacitance is greatest when there is a narrow separation between large areas of conductor, hence capacitor conductors are often called "plates", referring to an early means of construction. In practice the dielectric between the plates passes a small amount of leakage current and also has an electric field strength limit, resulting in a breakdown voltage, while the conductors and leads introduce an undesired inductance and resistance.



Parallel plate model



Dielectric is placed between two conducting plates, each of area A and with a separation of d.

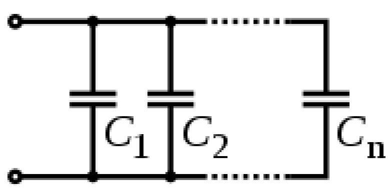
The simplest capacitor consists of two parallel conductive plates separated by a dielectric with permittivityε (such as air). The model may also be used to make qualitative predictions for other device geometries. The plates are considered to extend uniformly over an area A and a charge density $\pm\rho = \pm Q/A$ exists on their surface. Assuming that the width of the plates is much greater than their separation d, the electric field near the centre of the device will be uniform with the magnitude $E = \rho/\varepsilon$. The voltage is defined as the line integral of the electric field between the plates

$$V = \int_0^d E \mathrm{d}z = \int_0^d \frac{\rho}{\varepsilon} \mathrm{d}z = \frac{\rho d}{\varepsilon} = \frac{Qd}{\varepsilon A}.$$

Solving this for C = Q/V reveals that capacitance increases with area and decreases with separation

$$C = \frac{\varepsilon A}{d}.$$

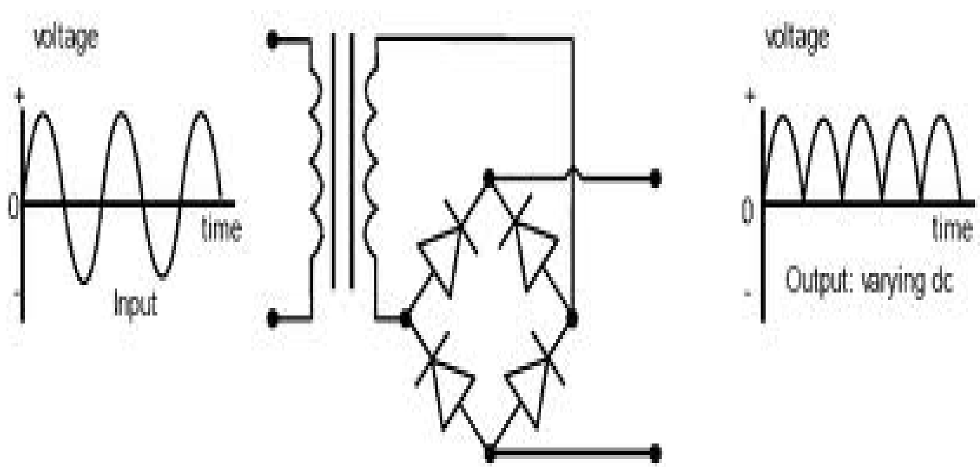
The capacitance is therefore greatest in devices made from materials with a high permittivity.



Several capacitors in parallel

5.6 RECTIFIERS

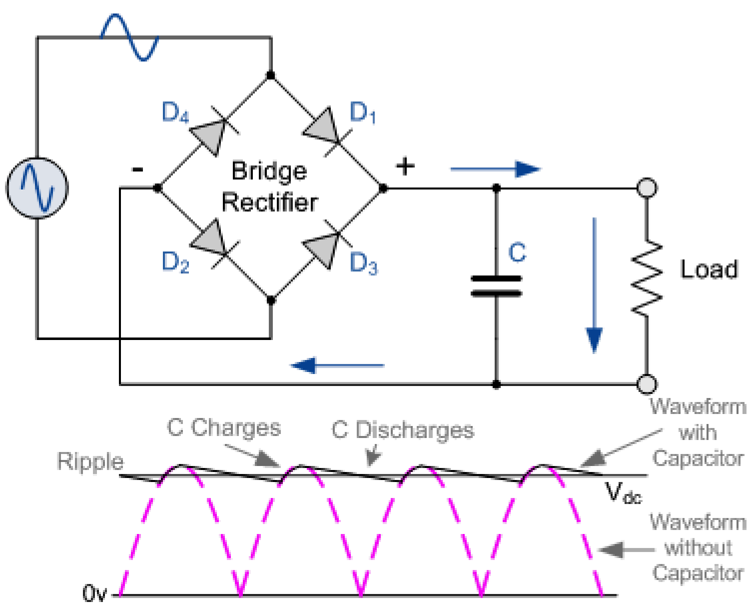
A rectifier is an electrical device that converts alternating current (AC), which periodically reverses direction, to direct current (DC), current that flows in only one direction, a process known as rectification. Rectifiers have many uses including as components of power supplies and as detectors of radio signals. Rectifiers may be made of solid statediodes, vacuum tube diodes, mercury arc valves, and other components. The output from the transformer is fed to the rectifier. It converts A.C. into pulsating D.C. The rectifier may be a half wave or a full wave rectifier. In this project, a bridge rectifier is used because of its merits like good stability and full wave rectification. In positive half cycleonly two diodes (1 set of parallel diodes) will conduct, in negative half cycle remaining two diodes will conduct and they will conduct only in forward bias only.



5.7 FILTERS

Capacitive filter is used in this project. It removes the ripples from the output of rectifier and smoothens the D.C. Output received from this filter is constant until the mains voltage and load is maintained constant. However, if either of the two is varied, D.C. voltage received at this point changes. Therefore a regulator is applied at the output stage.

The simple capacitor filter is the most basic type of power supply filter. The use of this filter is very limited. It is sometimes used on extremely high-voltage, low-current power supplies for cathode-ray and similar electron tubes that require very little load current from the supply. This filter is also used in circuits where the power-supply ripple frequency is not critical and can be relatively high. Below figure can show how the capacitor charges and discharges.



Resultant Output Waveform

6. INTRODUCTION TO ATMEGA328

1. Introduction:

The computer on one hand is designed to perform all the general purpose tasks on a single machine like you can use a computer to run a software to perform calculations or you can use a computer to store some multimedia file or to access internet through the browser, whereas the microcontrollers are meant to perform only the specific tasks, for e.g., switching the AC off automatically when room temperature drops to a certain defined limit and again turning it ON when temperature rises above the defined limit.

There are number of popular families of microcontrollers which are used in different applications as per their capability and feasibility to perform the desired task, most common of these are 8051, AVR and PIC microcontrollers. In this we will introduce you with AVR family of microcontrollers.

History of AVR

AVR was developed in the year 1996 by Atmel Corporation. The architecture of AVR was developed by Alf-Egil Bogen and Vegard Wollan. AVR derives its name from its developers and stands for Alf-Egil Bogen Vegard Wollan RISC microcontroller, also known as Advanced Virtual RISC.

AVR microcontrollers are available in three categories:

Tiny AVR – Less memory, small size, suitable only for simpler applications

Mega AVR – These are the most popular ones having good amount of memory (up-to 256 KB), higher number of in-built peripherals and suitable for moderate to complex applications.

Xmega AVR – Used commercially for complex applications, which require large program memory and high speed?

2. Features:

- RISC Architecture with CISC Instruction set
- Powerful C and assembly programming
- Scalable
- Same powerful AVR microcontroller core
- Low power consumption
- Both digital and analog input and output interfaces

3. Description:

The Atmel ATmega48/88/328 is a low-power CMOS 8-bit microcontroller based on the AVR enhanced RISC architecture. By executing powerful instructions in a single clock cycle, the ATmega48/88/328 achieves throughputs approaching 1 MIPS per MHz allowing the system designed to optimize power consumption versus processing speed.

The Atmel ATmega48/88/328 provides the following features: 4K/8K/16K bytes of In-System Programmable Flash with Read-While-Write capabilities, 256/512/512 bytes EEPROM, 512/1K/1K bytes SRAM, 23 general purpose I/O lines, 32 general purpose working registers, three flexible Timer/Counters with compare modes, internal and external interrupts, a serial programmable USART, a byte-oriented 2-wire Serial Interface, an SPI serial port, a 6-channel 10-bit ADC (8 channels in TQFP and QFN/MLF packages), a programmable Watchdog Timer with internal Oscillator, and five software selectable power saving modes. The Idle mode stops the CPU while allowing the SRAM, Timer/Counters, USART, 2-wire Serial Interface, SPI port, and interrupt system to continue functioning. The Power-down mode saves the register contents but freezes the Oscillator, disabling all other chip functions until the next interrupt or hardware reset. In Power-save mode, the asynchronous timer continues to run, allowing the user to maintain a timer base while the rest of the device is sleeping. The ADC Noise Reduction mode stops the CPU and all I/O modules except asynchronous timer and ADC, to minimize switching noise during ADC conversions. In Standby mode, the crystal/resonator Oscillator is running while the rest of the device is sleeping. This allows very fast start-up combined with low power consumption.

The ATmega48, ATmega88 and ATmega328 differ only in memory sizes, boot loader support, and interrupt vector sizes. Table 2-1 summarizes the different memory and interrupts vector sizes for the three devices.

Device	Flash	EEPROM	RAM	Interrupt vector size
ATmega48	4Kbytes	256Bytes	512Bytes	1 instruction word/vector
ATmega88	8Kbytes	512Bytes	1Kbytes	1 instruction word/vector
ATmega168	16Kbytes	512Bytes	1Kbytes	2 instruction words/vector

ATmega88 and ATmega328 support a real Read-While-Write Self-Programming mechanism. There is a separate Boot Loader Section, and the SPM instruction can only execute from there. In ATmega48, there is no Read-While-Write support and no separate Boot Loader Section. The SPM instruction can execute from the entire Flash.

4. Processor architecture

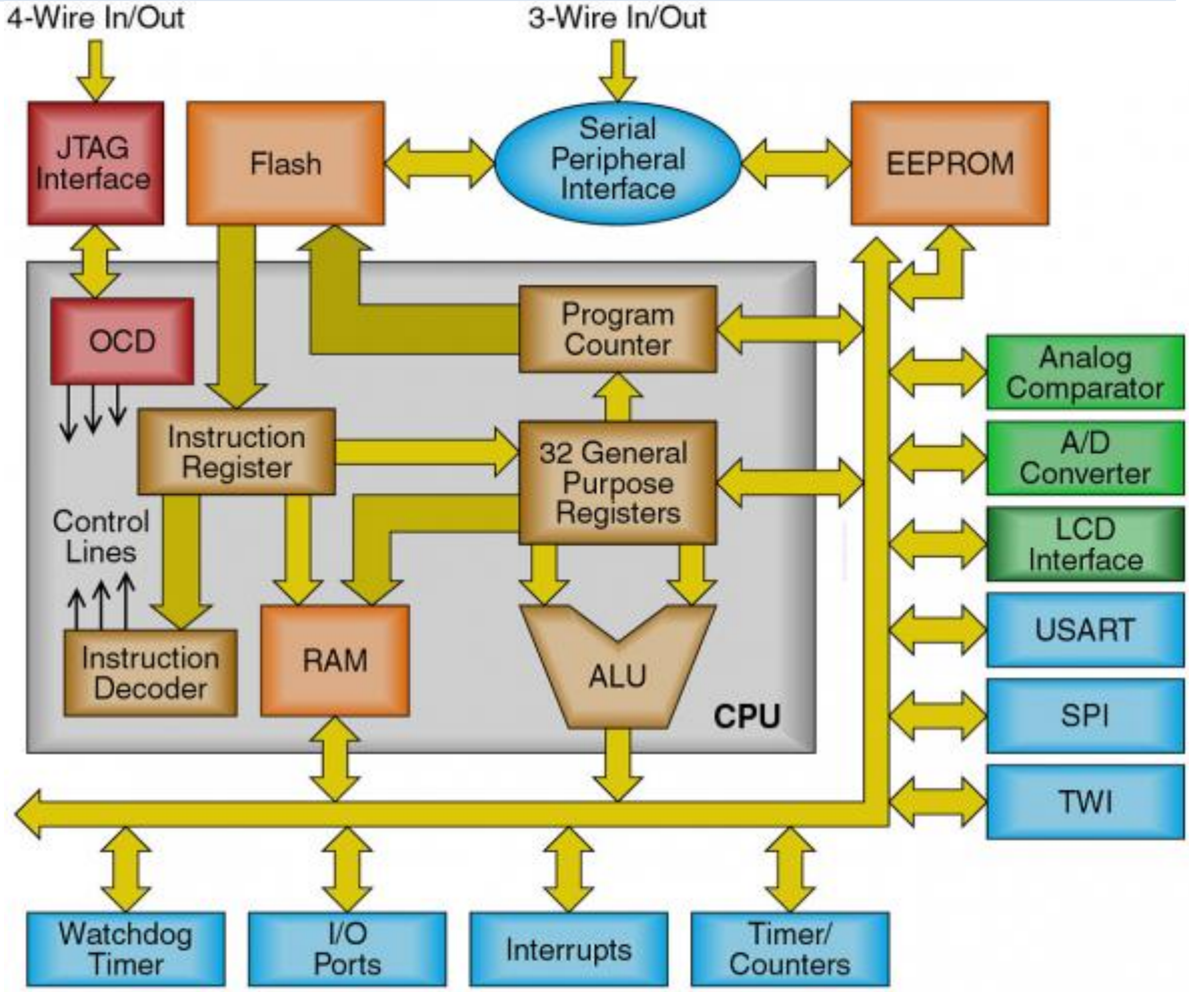


Fig 6(a) Architecture of ATmega328

AVR follows Harvard Architecture format in which the processor is equipped with separate memories and buses for Program and the Data information. Here while an instruction is being executed, the next instruction is pre-fetched from the program memory.

ALU: The high-performance AVR ALU operates in direct connection with all the 32 general purpose working registers. Within a single clock cycle, arithmetic operations between general purpose registers or between a register and an immediate are executed. The ALU operations are divided into three main categories – arithmetic, logical, and bit-functions. Some implementations of the architecture also provide a powerful multiplier supporting both signed/unsigned multiplication and fractional format.

In-system reprogrammable flash program memory: The ATmega48/88/328 contains 4K/8K/16K bytes On-chip In-System Reprogrammable Flash memory for program storage. Since all AVR instructions are 16 or 32 bits wide, the Flash is organized as 2K/4K/8K × 16. For software security, the Flash Program memory space is divided into two sections, Boot Loader Section and Application Program Section in ATmega88 and ATmega328.

EEPROM data memory: The Atmel ATmega48 /88/328 contains 256/512/512 bytes of data EEPROM memory. It is organized as a separate data space e, in which single bytes can be read and written. The EEPROM has an endurance of at least 100,000 write/erase cycles. The access between the EEPROM and the CPU is described in the following, specifying the EEPROM Address Registers, the EEPROM Data Register, and the EEPROM Control Register.

Program counter: A program counter is a register in a computer processor that contains the address (location) of the instruction being executed at the current time. As each instruction gets fetched, the program counter increases its stored value by 1. After each instruction is fetched, the program counter points to the next instruction in the sequence. When the computer restarts or is reset, the program counter normally reverts to 0. In computing, a program is a specific set of ordered operations for a computer to perform. An instruction is an order given to a computer processor by a program. Within a computer, an address is a specific location in memory or storage. A register is one of a small set of data holding places that the processor uses. Program counter is very important feature in the microcontrollers.

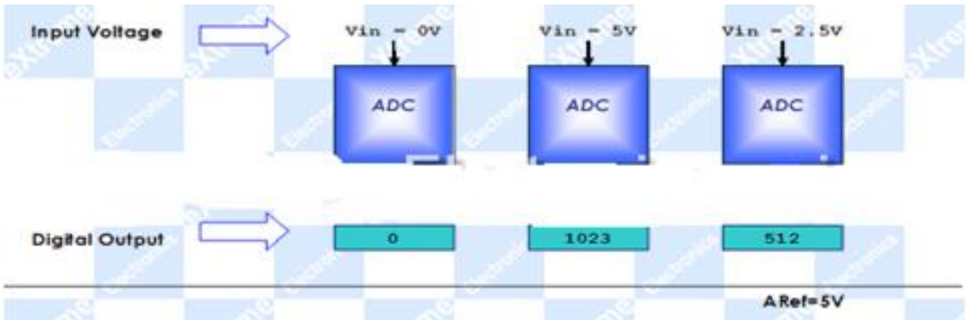
RAM: RAM stands for random access memory. This type of memory storage is temporary and volatile. You might have heard that if your system is working slowly you say that increase the RAM processing will increase. Let us understand in detail. Let us consider two cases to execute a task first the complete task is execute at one place(A), second the task is distributed in parts and the small tasks are executed at different places(A,B C)and finally assembled. It is clear the work will be finished in second case earlier. The A, B, C basically represent different address allocation for temporary processing. This is the case with RAM also if you increase the RAM the address basically increases for temporary processing so that no data has to wait for its turn. On major importance of the RAM is address allocations. However the storage is temporary every time u boot your system the data is lost but when you turn on the system The BIOS fetch number of addresses available in the RAM. This memory supports read as well as write operations both.

Instruction execution section (IES). It has the most important unit—instruction register and instruction decoder to control the flow of the instruction during the processing's.

INPUT/OUTPUT PORTS: To interact with the physical environment there are different input and output ports in every system like in PC we have VGA port to connect the monitor, USB port for flash memory connections and many more ports. Similarly ATMEGA 328 has its input and output ports with different configurations depending on the architecture like only input, only output and bi-directional input output ports. The accessing of this port is referred as input output interface design for microcontrollers. IT has analog input port, analog output port, digital input port ,digital output port, serial communication pins, timer execution pins etc.

Analog Comparator & A/D converters: The major question is that how a controller manage to detect variation of voltage in-spite it could not understand the voltage but understand only digital sequence

Most of the physical quantities around us are continuous. By continuous we mean that the quantity can take any value between two extreme. For example the atmospheric temperature can take any value (within certain range). If an electrical quantity is made to vary directly in proportion to this value (temperature etc) then what we have is Analogue signal. Now we have we have brought a physical quantity into electrical domain. The electrical quantity in most case is voltage. To bring this quantity into digital domain we have to convert this into digital form. For this a ADC or analog to digital converter is needed. Most modern MCU including AVRs has an ADC on chip. An ADC converts an input voltage into a number. An ADC has a resolution. A 10 Bit ADC has a range of 0-1023. (2^10=1024) The ADC also has a Reference voltage (ARef). When input voltage is GND the output is 0 and when input voltage is equal to ARef the output is 1023. So the input range is 0-ARef and digital output is 0-1023.



- **Inbuilt ADC of AVR**

  Now you know the basics of ADC let us see how we can use the inbuilt ADC of AVR MCU. The ADC is multiplexed with PORTA that means the ADC channels are shared with PORTA. The ADC can be operated in single conversion and free running more. In single conversion mode the ADC does the conversion and then stop. While in free it is continuously converting. It does a conversion and then start next conversion immediately after that.

- **ADC Pre-scalar.**

  The ADC needs a clock pulse to do its conversion. This clock generated by system clock by dividing it to get smaller frequency. The ADC requires a frequency between 50 KHz to 200 KHz. At higher frequency the conversion is fast while a lower frequency the conversion is more accurate. As the system frequency can be set to any value by the user (using internal or externals oscillators) (In board™ a 16MHz crystal is used). So the Pre-scalar is provided to produces acceptable frequency for ADC from any system clock frequency. System clock can be divided by 2, 4,16,32,64,128 by setting the Pre-scalar.

- **ADC Channels**

  The ADC in ATmega328 has 6 channels that mean you can take samples from eight different terminals. You can connect up to 8 different sensors and get their values separately.

5. Pin diagram & Description

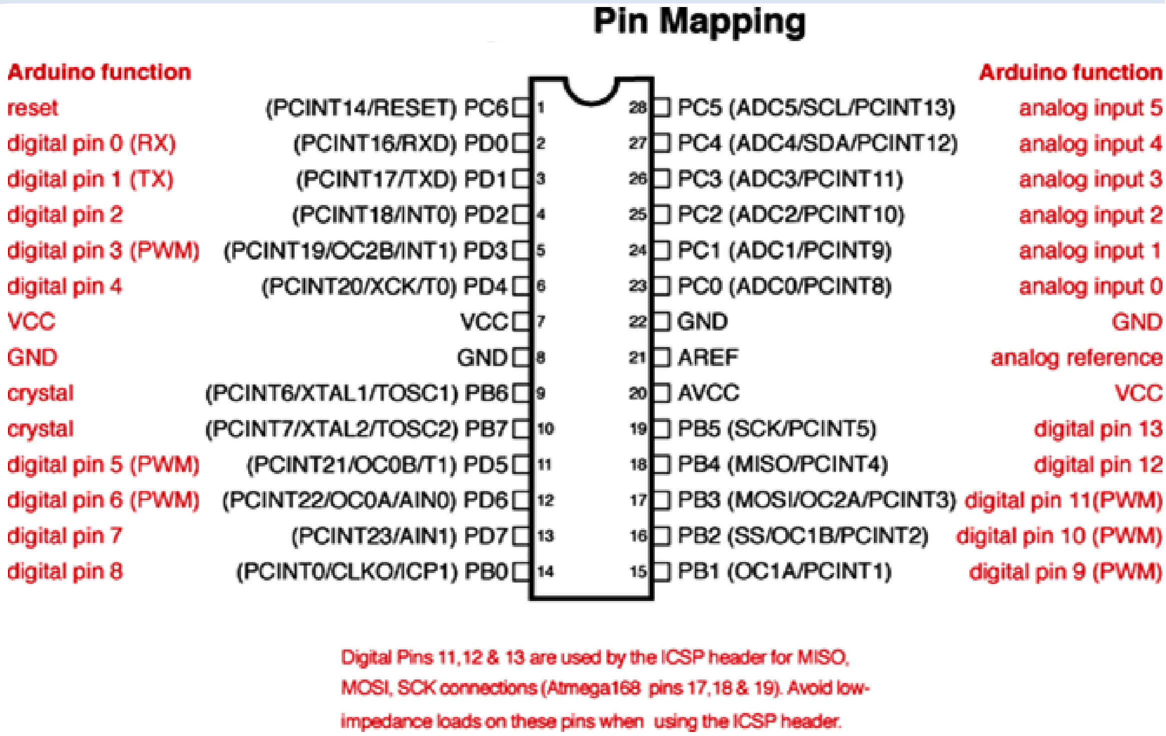


Fig 6(b) Pin Diagram of ATmega328

PIN DESCRIPTION:

VCC: Digital supply voltage.

GND: Ground.

Port B (PB7:0) XTAL1/XTAL2/TOSC1/TOSC2

Port B is an 8-bit bi-directional I/O port with internal pull- up resistors (selected for each bit). The Port B output buffers have symmetrical drive characteristics with both high sink and source Capability. As inputs, Port B pins that are externally pulled low will source current if the pull-up resistors are activated. The Port B pins are tri-stated when a reset condition becomes active, even if the clock is not running. Depending on the clock selection fuse settings, PB6 can be used as input to the inverting Oscillator amplifier and input to the internal clock operating circuit. Depending on the clock selection fuse settings, PB7 can be used as output from the inverting Oscillator amplifier. If the Internal

Calibrated RC Oscillator is used as chip clock source, PB7.6 is used as TOSC2.1 input for the Asynchronous Timer/Counter2 if the AS2 bit in ASSR is set.

Port C (PC5:0)

Port C is a 7-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). The PC5.0 output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, Port C pins that are externally pulled low will source current if the pull-up resistors are activated. The Port C pins are tri-stated when a reset condition becomes active, even if the clock is not running.

PC6/RESET:

If the RSTDISBL Fuse is programmed, PC6 is used as an I/O pin. Note that the electrical characteristics of PC6 differ from those of the other pins of Port C. If the RSTDISBL Fuse is un-programmed, PC6 is used as a Reset input. A low level on this pin for longer than the minimum pulse length will generate a Reset, even if the clock is not running. Shorter pulses are not guarantee to generate a reset.

Port D (PD7:0):

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

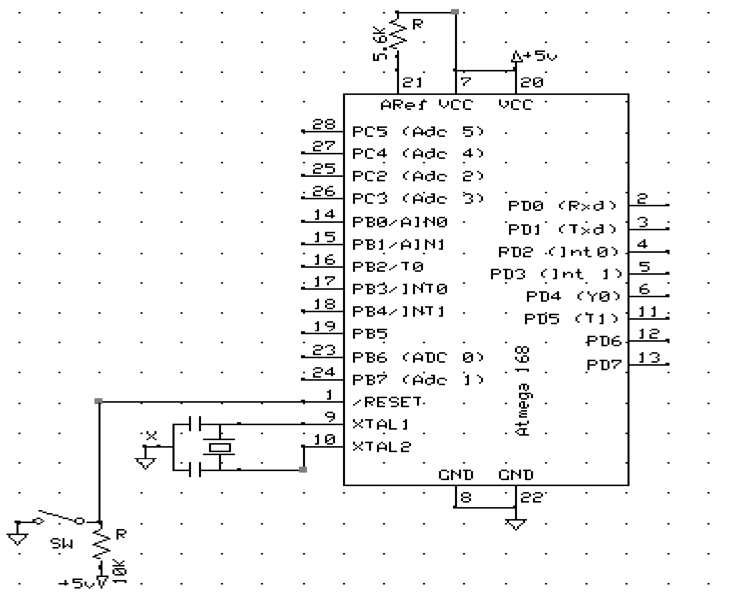
AVCC:

AVCC is the supply voltage pin for the A/D Converter PC3:0, and ADC7:6. It should be externally connected to VCC, even if the ADC is not used. If the ADC is used, it should be connected to VCC through a low-pass filter. Note that PC6.4 use digital supply voltage, VCC.

AREF: AREF is the analog reference pin for the A/D Converter.

6. Minimum Interface circuit for Atmega328 controller:

According the minimum interface discussed for the microcontrollers earlier, the minimum interface circuit for ATMEGA328 is:



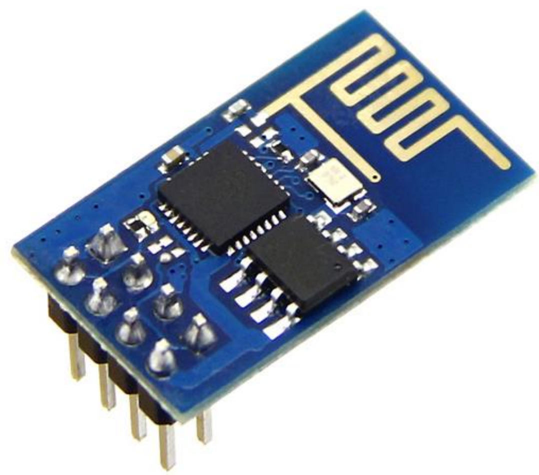
7. Things to remember about ATMEGA328 controllers:

The ATMEGA controllers are strong controllers but you have to take some small points in mind always like:

- When you go for the programming of atmega328 consider the pin no. as configures in red color in Pin diagram shown before (like controllers pin number 2 is digital pin number 0 for input or output ,pin23 is analog pin A0). Sao you will address the pin according to that number.
- Use the proper pin for proper input output interface that analog input should be configured at analog pin analog output should be configured on PWM pins and likewise the digital inputs and outputs

7. WI-FI MODEM

The ESP8266 WiFi Module is a self contained SOC with integrated 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. Each ESP8266 module comes pre-programmed with an AT command set firmware. The ESP8266 module is an extremely cost effective board with a huge, and ever growing, community.



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!

Note: The ESP8266 Module is not capable of 5-3V logic shifting and will require an external Logic Level Converter. Please do not power it directly from your 5V dev board.

Features:

802.11 b/g/n

Wi-Fi Direct (P2P), soft-AP

Integrated TCP/IP protocol stack

Integrated TR switch, balun, LNA, power amplifier and matching network

Integrated PLLs, regulators, DCXO and power management units

+19.5dBm output power in 802.11b mode

Power down leakage current of <10uA

1MB Flash Memory

Integrated low power 32-bit CPU could be used as application processor

SDIO 1.1 / 2.0, SPI, UART

STBC, 1×1 MIMO, 2×1 MIMO

A-MPDU & A-MSDU aggregation & 0.4ms guard interval

Wake up and transmit packets in < 2ms

Standby power consumption of < 1.0mW (DTIM3)

8. LIQUID CRYSTAL DISPLAY (LCD)

Description:

This is the example for the Parallel Port. This example doesn't use the Bi-directional feature found on newer ports, thus it should work with most, if not all Parallel Ports. It however doesn't show the use of the Status Port as an input for a 16 Character x 2 Line LCD Module to the Parallel Port. These LCD Modules are very common these days, and are quite simple to work with, as all the logic required running them is on board.

Pros:

- Very compact and light
- Low power consumption
- No geometric distortion
- Little or no flicker depending on backlight technology
- Not affected by screen burn-in
- No high voltage or other hazards present during repair/service
- Can be made in almost any size or shape
- No theoretical resolution limit

LCD Background:

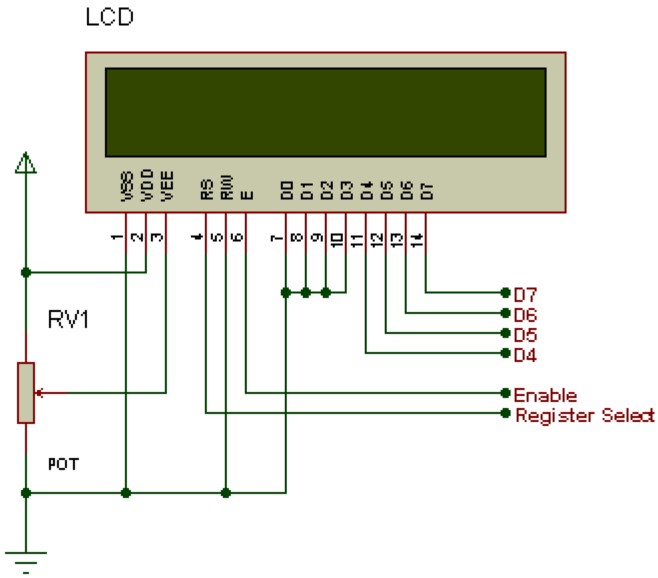
Frequently, an 8051 program must interact with the outside world using input and output devices that communicate directly with a human being. One of the most common devices attached to an 8051 is an LCD display. Some of the most common LCDs connected to the 8051 are 16x2 and 20x2 displays. This means 16 characters per line by 2 lines and 20 characters per line by 2 lines, respectively.

Fortunately, a very popular standard exists which allows us to communicate with the vast majority of LCDs regardless of their manufacturer. The standard is referred to as HD44780U, which refers to the controller chip which receives data from an external source (in this case, the 8051) and communicates directly with the LCD.

Fig 8(a): LCD

44780 LCD BACKGROUND

The 44780 standard requires 3 control lines as well as either 4 or 8 I/O lines for the data bus. The user may select whether the LCD is to operate with a 4-bit data bus or an 8-bit data bus. If a 4-bit data bus is used the LCD will require a total of 7 data lines (3 control lines plus the 4 lines for the data bus). If an 8-bit data bus is used the LCD will require a total of 11 data lines (3 control lines plus the 8 lines for the data bus).



The three control lines are referred to as EN, RS, and RW.

The EN line is called "Enable." This control line is used to tell the LCD that you are sending it data. To send data to the LCD, your program should make sure this line is low (0) and then set the other two control lines and/or put data on the data bus. When the other lines are completely ready, bring EN high (1) and wait for the minimum amount of time required by the LCD datasheet (this varies from LCD to LCD), and end by bringing it low (0) again.

The RS line is the "Register Select" line. When RS is low (0), the data is to be treated as a command or special instruction (such as clear screen, position cursor, etc.). When RS is high (1), the data being sent is text data which should be displayed on the screen. For example, to display the letter "T" on the screen you would set RS high.

The RW line is the "Read/Write" control line. When RW is low (0), the information on the data bus is being written to the LCD. When RW is high (1), the program is effectively querying (or reading) the LCD. Only one instruction ("Get LCD status") is a read command. All others are write commands--so RW will almost always be low .Finally, the data bus consists of 4 or 8 lines (depending on the mode of operation selected by the user). In the case of an 8-bit data bus, the lines are referred to as DB0, DB1, DB2, DB3, DB4, DB5, DB6, and DB7.

8.1 LED

A Light-Emitting Diode (LED) is a semiconductor light source. LEDs are used as indicator lamps in many devices, and are increasingly used for lighting. When a light-emitting diode is forward biased (switched on), electrons are able to recombine with holes within the device, releasing energy in the form of photons.

This effect is called electroluminescence and the color of the light (corresponding to the energy of the photon) is determined by the energy gap of the semiconductor. An LED is often small in area (less than 1 mm^2), and integrated optical components may be used to shape its radiation pattern. LEDs present many advantages over incandescent light sources including lower energy consumption, longer lifetime, improved robustness, smaller size, faster switching, and greater durability and reliability.

Types of LED'S

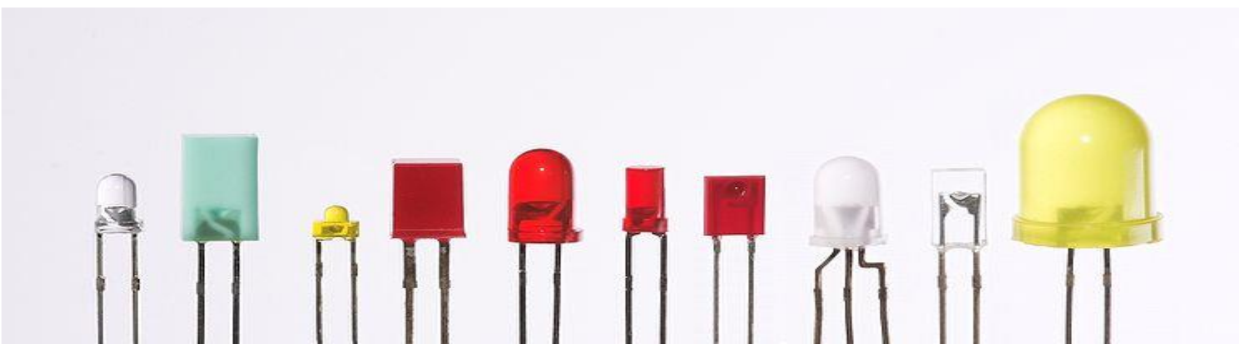


Fig 8.1(a): Types of LEDs

Light-emitting diodes are used in applications as diverse as replacements for aviation lighting, automotive lighting as well as in traffic signals. The compact size, the possibility of narrow bandwidth, switching speed, and extreme reliability of LEDs has allowed new text and video displays and sensors to be developed, while their high switching rates are also useful in advanced communications technology.

Electronic Symbol:

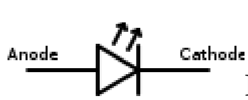


Fig 8.1(b): Symbol of LED

White LED'S

Light Emitting Diodes (LED) have recently become available that are both white and bright, so bright that they seriously compete with incandescent lamps in lighting applications. They are still pretty expensive as compared to a GOW lamp but draw much less current and project a fairly well focused beam.

When run within their ratings, they are more reliable than lamps as well. Red LEDs are now being used in automotive and truck tail lights and in red traffic signal lights. You will be able to detect them because they look like an array of point sources and they go on and off instantly as compared to conventional incandescent lamps.

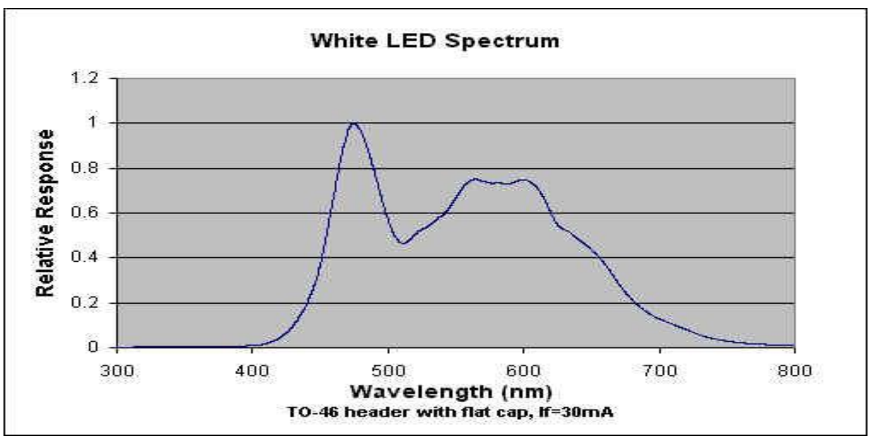
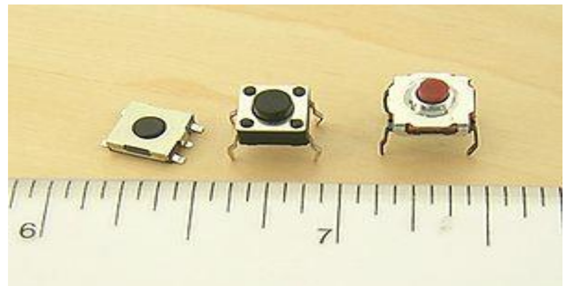


Fig 8.1(c): White LED spectrum

9. PUSH BUTTONS

A push-button (also spelled pushbutton) or simply button is a simple switch mechanism for controlling some aspect of a machine or a process. Buttons are typically made out of hard material, usually plastic or metal. The surface is usually flat or shaped to accommodate the human finger or hand, so as to be easily depressed or pushed. Buttons are most often biased switches, though even many un-biased buttons (due to their physical nature) require a spring to return to their un-pushed state. Different people use different terms for the "pushing" of the button, such as press, depress, mash, and punch.



Uses:

In industrial and commercial applications push buttons can be linked together by a mechanical linkage so that the act of pushing one button causes the other button to be released. In this way, a stop button can "force" a start button to be released. This method of linkage is used in simple manual operations in which the machine or process have no electrical circuits for control.

Pushbuttons are often color-coded to associate them with their function so that the operator will not push the wrong button in error. Commonly used colors are red for stopping the machine or process and green for starting the machine or process.

Red pushbuttons can also have large heads (mushroom shaped) for easy operation and to facilitate the stopping of a machine. These pushbuttons are called emergency stop buttons and are mandated by the electrical code in many jurisdictions for increased safety. This large mushroom shape can also be found in buttons for use with operators who need to wear gloves for their work and could not actuate a regular flush-mounted push button. As an aid for operators and users in industrial or commercial applications, a pilot light is commonly added to draw the attention of the user and to provide feedback if the button is pushed. Typically this light is included into the center of the pushbutton and a lens replaces the pushbutton hard center disk.

The source of the energy to illuminate the light is not directly tied to the contacts on the back of the pushbutton but to the action the pushbutton controls. In this way a start button when pushed will cause the process or machine operation to be started and a secondary contact designed into the operation or process will close to turn on the pilot light and signify the action of pushing the button caused the resultant process or action to start.

In popular culture, the phrase "the button" refers to a (usually fictional) button that a military or government leader could press to launch nuclear weapons.

Push to ON button:

Fig 9(a): Push ON button

Initially the two contacts of the button are open. When the button is pressed they become connected. This makes the switching operation using the push button.

10. SCHEMATIC DIAGRAM

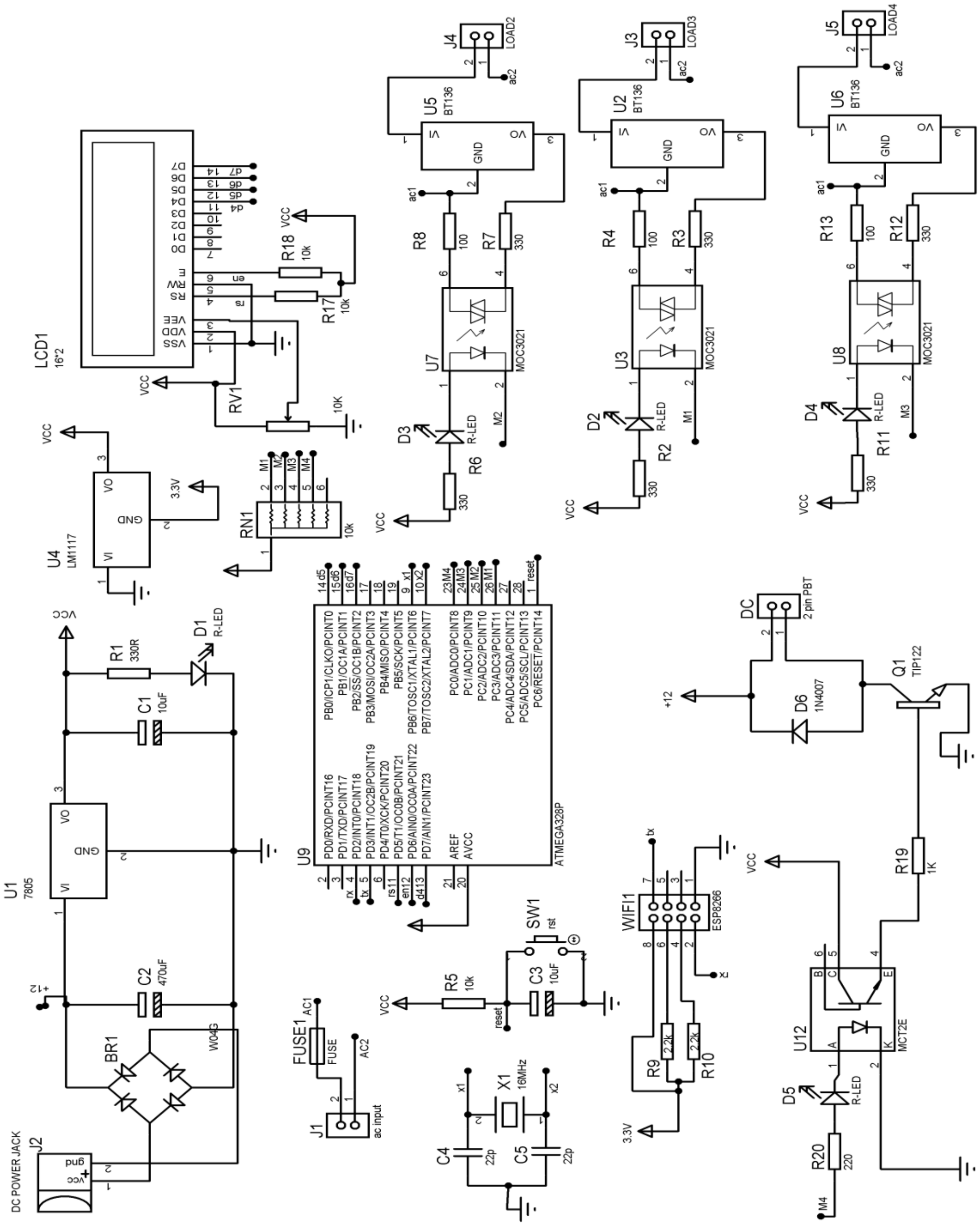


Fig (10) Schematic diagram

10.1 LAYOUT DIAGRAM

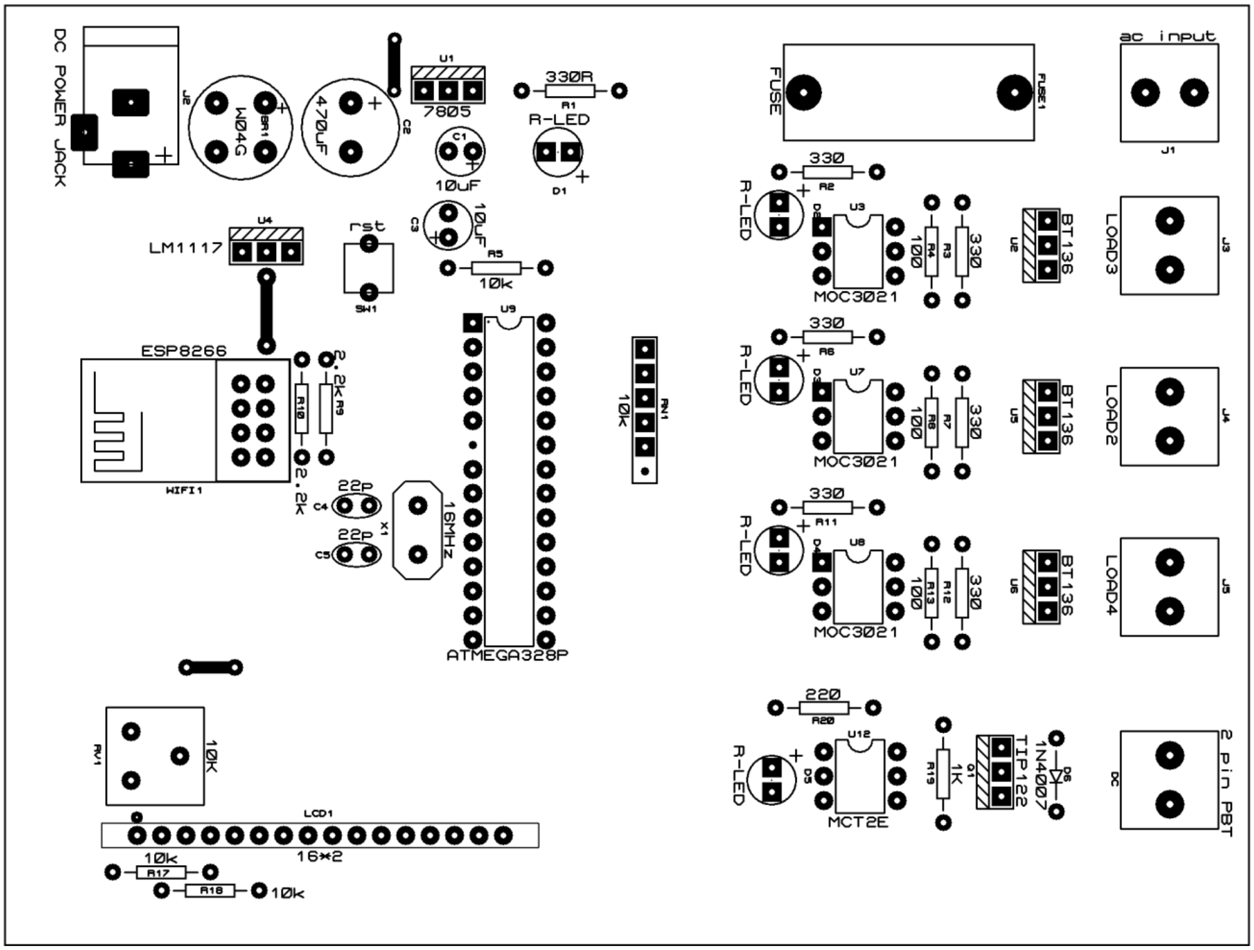


Fig (10.1) Layout Diagram

10.2 DESCRIPTION

The essential components for IoT Enabled Home Automation System can be listed as a AVR Microcontroller, Wi-Fi module, Relays, Relay Driver, Resistors, Capacitors, diode, Voltage Regulator, loads (Lamps & Fan), LCD, LEDs.

In this project Lamps & Fan can be controlled and monitored using a web page with user configurable front end. The user can send commands through the allotted IP and these commands are fed to Wi-Fi module. The Wi-Fi module is configured to access internet using any nearby wireless modem. The commands received by a Wi-Fi module are executed by a program within a Wi-Fi module. The Wi-Fi module is interfaced to Relays through the loads are turned ON & OFF based on commands. The load status (ON or OFF) will be displayed on the web page.

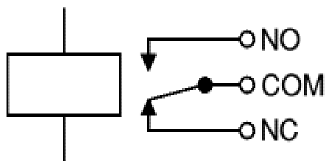
POWER SUPPLY

The circuit uses standard power supply comprising of a step-down transformer from 230Vto 12V and 4 diodes forming a bridge rectifier that delivers pulsating dc which is then filtered by an electrolytic capacitor of about 470µF to 1000µF. The filtered dc being unregulated, IC LM7805 is used to get 5V DC constant at its pin no 3 irrespective of input DC varying from 7V to 15V. The input dc shall be varying in the event of input ac at 230volts section varies from 160V to 270V in the ratio of the transformer primary voltage V1 to secondary voltage V2 governed by the formula V1/V2=N1/N2. As N1/N2 i.e. no. of turns in the primary to the no. of turns in the secondary remains unchanged V2 is directly proportional to V1.Thus if the transformer delivers 12V at 220V input it will give 8.72V at 160V.Similarly at 270V it will give 14.72V.Thus the dc voltage at the input of the regulator changes from about 8V to 15V because of A.C voltage variation from 160V to 270V the regulator output will remain constant at 5V.

The regulated 5V DC is further filtered by a small electrolytic capacitor of 10µF for any noise so generated by the circuit. One LED is connected of this 5V point in series with a current limiting resistor of 330Ω to the ground i.e., negative voltage to indicate 5V power supply availability. The unregulated 12V point is used for other applications as and when required.

10.3 RELAYS

A relay is an electrically operated switch. Many relays use an electromagnet to operate a switching mechanism mechanically, but other operating principles are also used. Relays are used where it is necessary to control a circuit by a low-power signal (with complete electrical isolation between control and controlled circuits), or where several circuits must be controlled by one signal.





A relay is an electrically operated switch. Current flowing through the coil of the relay creates a magnetic field which attracts a lever and changes the switch contacts. The coil current can be on or off so relays have two switch positions and most have double throw (changeover) switch contacts as shown in the diagram.

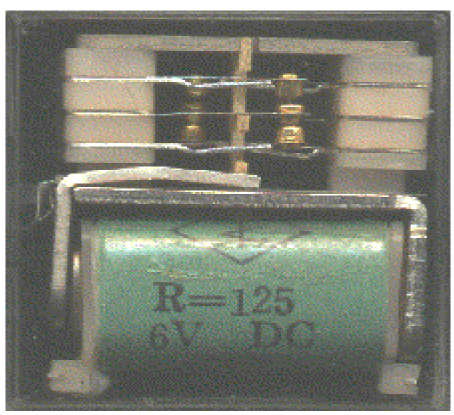


Fig (10.3) Relay showing coil and switch contacts

Relays allow one circuit to switch a second circuit which can be completely separate from the first. For example a low voltage battery circuit can use a relay to switch a 230V AC mains circuit. There is no electrical connection inside the relay between the two circuits; the link is magnetic and mechanical.

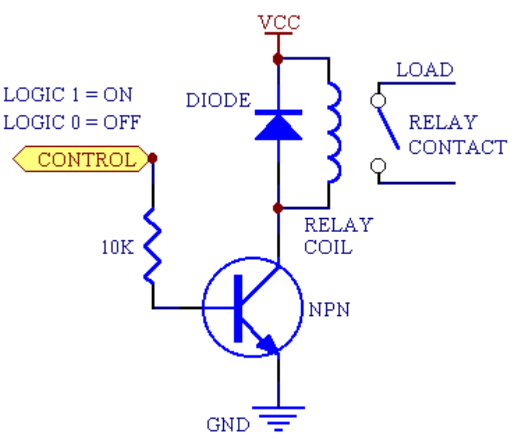
The coil of a relay passes a relatively large current, typically 30mA for a 12V relay, but it can be as much as 100mA for relays designed to operate from lower voltages. Most ICs (chips) cannot provide this current and a transistor is usually used to amplify the small IC current to the larger value required for the relay coil. The maximum output current for the popular 555 timer IC is 200mA so these devices can supply relay coils directly without amplification.

Relays are usually SPDT or DPDT but they can have many more sets of switch contacts, for example relays with 4 sets of changeover contacts are readily available. For further information about switch contacts and the terms used to describe them please see the page on switches.

Most relays are designed for PCB mounting but you can solder wires directly to the pins providing you take care to avoid melting the plastic case of the relay.

The supplier's catalogue should show you the relay's connections. The coil will be obvious and it may be connected either way round. Relay coils produce brief high voltage 'spikes' when they are switched off and this can destroy transistors and ICs in the circuit. To prevent damage you must connect a protection diode across the relay coil.

The figure shows a relay with its coil and switch contacts. You can see a lever on the left being attracted by magnetism when the coil is switched on. This lever moves the switch contacts.



There is one set of contacts (SPDT) in the foreground and another behind them, making the relay DPDT.

The relay's switch connections are usually labelled COM, NC and NO:

- COM = Common, always connect to this; it is the moving part of the switch.
- NC = Normally Closed, COM is connected to this when the relay coil is off.
- NO = Normally Open, COM is connected to this when the relay coil is on.

Applications of relays

Relays are used to and for:

- Control a high-voltage circuit with a low-voltage signal, as in some types of modems or audio amplifiers.
- Control a high-current circuit with a low-current signal, as in the starter solenoid of an automobile.
- Detect and isolate faults on transmission and distribution lines by opening and closing circuit breakers.
- 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.

10.4 RELAY DRIVER (ULN2003)

RELAY DRIVER:

ULN2003 is a high voltage and high current Darlington transistor array.

DESCRIPTION:

The ULN2003 is a monolithic high voltage and high current Darlington transistor arrays. It consists of seven NPN Darlington pairs that feature high-voltage outputs with common-cathode Clamp diode for switching inductive loads. The collector-current rating of a single Darlington pair is 500mA. The Darlington pairs may be paralleled for higher current capability. Applications include relay drivers, hammer drivers, lamp drivers, display drivers (LED gas discharge), line drivers, and logic buffers.

The ULN2003 has a 2.7kW series base resistor for each Darlington pair for operation directly with TTL or 5V CMOS devices.

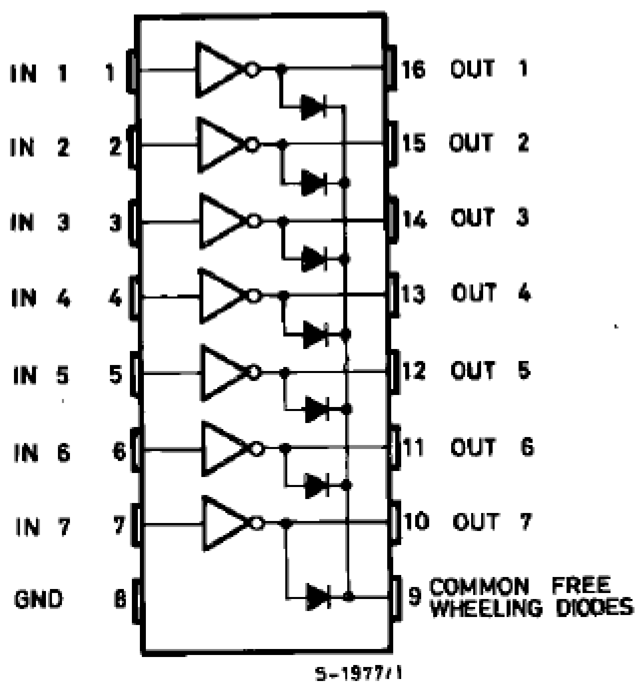
Fig 10.4(a): ULN 2003

FEATURES:

- Pin no.:16
- Temperature, Operating Range:-20°C to +85°C
- Transistor Polarity:NPN
- Transistors, No. of:7

- Case Style:DIP-16
- Temp, Op. Min:-20°C
- Temp, Op. Max:85°C
- Base Number:2003
- Channels, No. of:7
- Current, Output Max:500mA
- Device Marking:ULN2003A
- IC Generic Number:2003
- Input Type:TTL, CMOS 5V
- Logic Function Number:2003
- Output Type: Open Collector
- Transistor Type: Power Darlington
- Voltage, Input Max:5V
- Voltage, Output Max:50V

Pin Diagram:



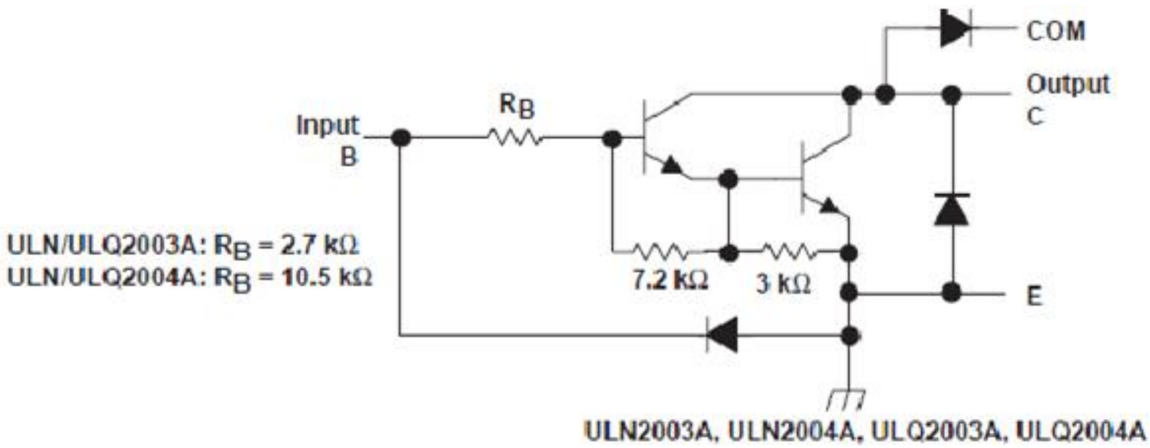


Fig 10.4(b): Schematics of Darlington's pair

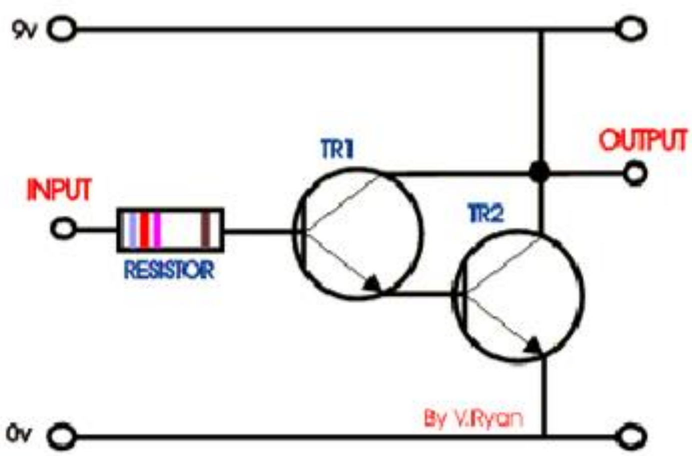


Fig 10.4(c): Darlington pair

Darlington pairs are back to back connection of two transistors with somesource resistorsand when they are arranged as shown in the circuit they are used to amplify weak signals.The amount by which the weak signal is amplified is called the 'GAIN'. .

FEATURES OF DRIVER:

- Seven Darlington's per package
- Output currents500mA per driver(600mA peak)
- Integrated suppression diodes for inductive loads
- Outputs can be paralleled for high currents
- TTL/CMOS/PMOS/DTL compatible inputs.
- Inputs pinned opposite to outputs
- Simplified layout

Figure shows the Darlington pair connection of transistor. The circuit above is a 'Darlington Pair' driver. The first transistor's emitter feeds into the second transistor's base and as a result the input signal is amplified by the time it reaches the output. The important point to remember is that the Darlington Pair is made up of two transistors

FEATURES

* 500mA rated collector current (Single output).

* High-voltage outputs: 50V.

* Inputs compatible with various types of logic.

* Relay driver application.

10.5 CODING

```
#include <SoftwareSerial.h>

#include <LiquidCrystal.h>

//#include <PS2Keyboard.h>



#define DEBUG true

#define FALSE false



const int load1 = A2;

const int load2 = A3;

const int load3 = A4;

const int load4 = A5;

const int tempPin = A1;



int reading;

int tempC;
```

```
//const int DataPin = A0;

//const int IRQpin =  3;


int load1_st = 0, load2_st = 0, load3_st = 0, load4_st = 0;

//int load1_prev = 0, load2_prev = 0, load3_prev = 0, load4_prev = 0;

int update_st = 0;


//char c;

//String ssid="";

//String pass="";


SoftwareSerial esp8266(2,3); // make RX Arduino line is pin 2, make TX Arduino line is pin 3.

                        // This means that you need to connect the TX line from the esp to the Arduino's pin 2

                        // and the RX line from the esp to the Arduino's pin 3

LiquidCrystal lcd( 12, 4 , 8, 9, 10, 11);
```

```
//PS2Keyboard keyboard;

String response = "";

int cmd=1; int i = 0;

long int looptime = 0;



void disp_update();

//int index = 0;

void setup()

{

  Serial.begin(9600);

  esp8266.begin(9600); // your esp's baud rate might be different

  lcd.begin(16, 2);

  //analogReference(INTERNAL);

  pinMode(load1,OUTPUT);

  digitalWrite(load1,HIGH);



  pinMode(load2,OUTPUT);
```

```
  digitalWrite(load2,HIGH);

  pinMode(load3,OUTPUT);
  digitalWrite(load3,HIGH);

  pinMode(load4,OUTPUT);
  digitalWrite(load4,HIGH);

  lcd.clear();
  lcd.print("IOT - home");
  lcd.setCursor(0,1);
  lcd.print("   Automation");
  //temp();

  sendData(F("AT+RST\r\n"),2000,FALSE); // reset module
  sendData(F("AT+CIPMUX=1\r\n"),1000,FALSE); // configure for multiple connections
```

```
  sendData(F("AT+CWMODE=3\r\n"),1000,FALSE); // configure as access point

  sendData(F("AT+CWJAP=\"iothome\",\"project1234\"\r\n"),10000,FALSE);

//  sendData("AT+CIFSR\r\n",3000,DEBUG); // get ip address

}


void loop()

{

  reading = analogRead(tempPin);

  tempC = reading / 2;



sendData(F("AT+CIPSTART=2,\"TCP\",\"www.nevemtech.com\",80\r\n"),5000,F
ALSE);

   /* if(response[2] == 'b' && response[3] == 'u' && response[4] == 's' &&
response[5] == 'y')    delay(2000); */

    sendData(F("AT+CIPSEND=2,80\r\n"),50,FALSE);

   // delay(5000);
```

```
//      if(response[2] == 'b' && response[3] == 'u' && response[4] == 's' &&
response[5] == 'y')    delay(5000);

    sendData(F("GET /IOTtest.aspx?d1=5644&d2=xy56&tmp="),0,FALSE);

    sendData(String(tempC),0,FALSE);

    sendData(F(" HTTP/1.1\r\n"),100,FALSE);

    delay(100);

//      if(response[2] == 'b' && response[3] == 'u' && response[4] == 's' &&
response[5] == 'y')    delay(5000);

    sendData(F("Host: www.nevemtech.com\r\n\r\n\r\n"),5000,FALSE);

//      if(response[2] == 'b' && response[3] == 'u' && response[4] == 's' &&
response[5] == 'y')    delay(5000);

//    delay(5000);



    //delay(100);

    i=0;

    looptime = millis();

    while(cmd)

    {
```

```
   // if(response[i++] == 'd')
      if(response[i++] == 'p')
        if(response[i++] == 'i')
          //if(response[i++] == 'n')
          //if(response[i++] == '=')
          {
            i = i+2;
//              load1_prev = load1_st;
//              load2_prev = load2_st;
//              load3_prev = load3_st;
 //             load4_prev = load4_st;

            load1_st = (response[i++] - 48);
            load2_st = (response[i++] - 48);
            load3_st = (response[i++] - 48);
            load4_st = (response[i++] - 48);
```

```
        cmd = 0;

        update_st = 1;

       }


  if(millis() - looptime > 3000)

  {

   cmd = 0;

   sendData(F("AT+RST\r\n"),2000,DEBUG);

   sendData(F("AT+CWMODE=3\r\n"),3000,DEBUG); // configure as access point

   sendData(F("AT+CIPMUX=1\r\n"),3000,DEBUG); // configure for multiple connections

   update_st = 0;

 }

 }

  sendData(F("AT+CIPCLOSE=4\r\n\r\n"),5000,FALSE);

 cmd = 1;
```

```
// temp();
lcd.begin(16,2);
 lcd.clear();
 disp_update();

 if(update_st == 1)
 {
  if(load1_st == 1)
  {
  digitalWrite(load1, LOW); // toggle pin
  }
  else
  {
    digitalWrite(load1, HIGH);
  }
  delay(500);
```

```
if(load2_st == 1)
{
digitalWrite(load2, LOW); // toggle pin
}
else
{
  digitalWrite(load2,HIGH);
}
delay(500);
if(load3_st == 1)
{
  digitalWrite(load3, LOW); // toggle pi
}
else
{
  digitalWrite(load3, HIGH);
}
```

```
    delay(500);
    if(load4_st == 1)
    {
      digitalWrite(load4, LOW); // toggle pin
    }
    else
    {
      digitalWrite(load4, HIGH);
    }
   }
//   delay(500);
   sendData(F("AT+CWJAP=\"iothome\",\"project1234\"\r\n"),10000,FALSE);
}


/*
* Name: sendData
* Description: Function used to send data to ESP8266.
```

* Params: command - the data/command to send; timeout - the time to wait for a response; debug - print to Serial window?(true = yes, false = no)

* Returns: The response from the esp8266 (if there is a reponse)

*/

```
String sendData(String command, const int timeout, boolean debug)
{
    response = "";

    esp8266.print(command); // send the read character to the esp8266

    long int time = millis();

    while( (time+timeout) > millis())
    {
      while(esp8266.available())
      {
```

```
        // The esp has data so display its output to the serial window

        char c = esp8266.read(); // read the next character.

        response+=c;

      }

    }



    if(debug)

    {

      Serial.print(response);

    }





    return response;

}



void disp_update()

{
```

```
if(update_st == 1)
{
  lcd.setCursor(0,0);
  if(load1_st==1)
    lcd.print("L1 ON");
  else
    lcd.print("L1 OFF");

  lcd.setCursor(10,0);
  if(load2_st==1)
    lcd.print("L2 ON");
  else
    lcd.print("L2 OFF");

  lcd.setCursor(0,1);
  if(load3_st==1)
     lcd.print("L3 ON");
```

```
    else
       lcd.print("L3 OFF");


    lcd.setCursor(10,1);
    if(load4_st==1)
      lcd.print("L4 ON");
    else
      lcd.print("L4 OFF");
  }
  else
  lcd.print("Connecting...");
}
//void temp()
//{
//  reading = analogRead(tempPin);
//  tempC = reading / 2;
//  dtostrf(tempC, 0, 0, char1);
```

```
//  str=String(char1);

//  delay(10);

//}
```

11. SOFTWARE REQUIREMENTS

11.1 INTRODUCTION TO AVR STUDIO 5

AVR Studio 5 is a free integrated development environment (IDE) for programming AVRs offered by Atmel for Atmel's 8-bits, 32-bits and ARM Cortex-M families of AVR microcontrollers. AVR Studio 5 works with the WinAVR avr-gcc compiler and contains built-in support for AVR ISP programming.

- ✓ It is a full software development environment with an editor, simulator, programmer, etc.
- ✓ It comes with its own integrated C compiler the **AVR GNU C Compiler (GCC)**. As such you do not need a third party C compiler.
- ✓ It provides a single environment to develop programs for both the 8-bits, 32-bits and ARM Cortex-M AVR series of microcontrollers.
- ✓ It also integrates fully QTouch studio.
- ✓ Provides support for several programmers including the STK500, AVR Dragon, etc.

Installation

Requirements Windows 98/NT/2000/XP XP x64/VISTA/WIN 7 Internet Explorer 6.0 or later (Latest version is recommended) Recommended hardware: ~ Intel Pentium 200MHz processor or equivalent ~ 1024x768 screen (minimum 800x600 screen) ~ 256 MB memory ~ 100 MB free hard disk space

11.2 CONCEPT OF COMPILER

Compilers are programs used to convert a High Level Language to object code. Desktop compilers produce an output object code for the underlying microprocessor, but not for other microprocessors. I.E the programs written in one of the HLL like ‘C’ will compile the code to run on the system for a particular processor like x86 (underlying microprocessor in the computer). For example compilers for Dos platform is different from the Compilers for Unix platform So if one wants to define a compiler then compiler is a program that translates source code into object code.

The compiler derives its name from the way it works, looking at the entire piece of source code and collecting and reorganizing the instruction. See there is a bit little difference between compiler and an interpreter. Interpreter just interprets whole program at a time while compiler analyses and execute each line of source code in succession, without looking at the entire program.

The advantage of interpreters is that they can execute a program immediately. Secondly programs produced by compilers run much faster than the same programs executed by an interpreter. However compilers require some time before an executable program emerges. Now as compilers translate source code into object code, which is unique for each type of computer, many compilers are available for the same language.

11.3 CONCEPT OF CROSS COMPILER

A cross compiler is similar to the compilers but we write a program for the target processor on the host processors (like computer of x86). It means being in one environment you are writing a code for another environment is called cross development. And the compiler used for cross development is called cross compiler. So the definition of cross compiler is a compiler that runs on one computer but produces object code for a different type of computer.

11.4 KEIL CROSS COMPILER

WinAVR™ (pronounced "whenever") is a suite of executable, open source software development tools for the Atmel AVR series of RISC microprocessors hosted on the Windows platform. It includes the GNU GCC compiler for C and C++.

WinAVR™ contains all the tools for developing on the AVR. This includes avr-gcc (compiler), avrdude (programmer), avr-gdb (debugger), and more! WinAVR is used all over the world from hobbyists sitting in their damp basements, to schools, to commercial projects.

WinAVR™ is comprised of many open source projects. If you feel adventurous, volunteers are always welcome to help with fixing bugs, adding features, porting, writing documentation and many other tasks on a variety of projects.

11.5 BUILDING AN APPLICATION IN AVR STUDIO

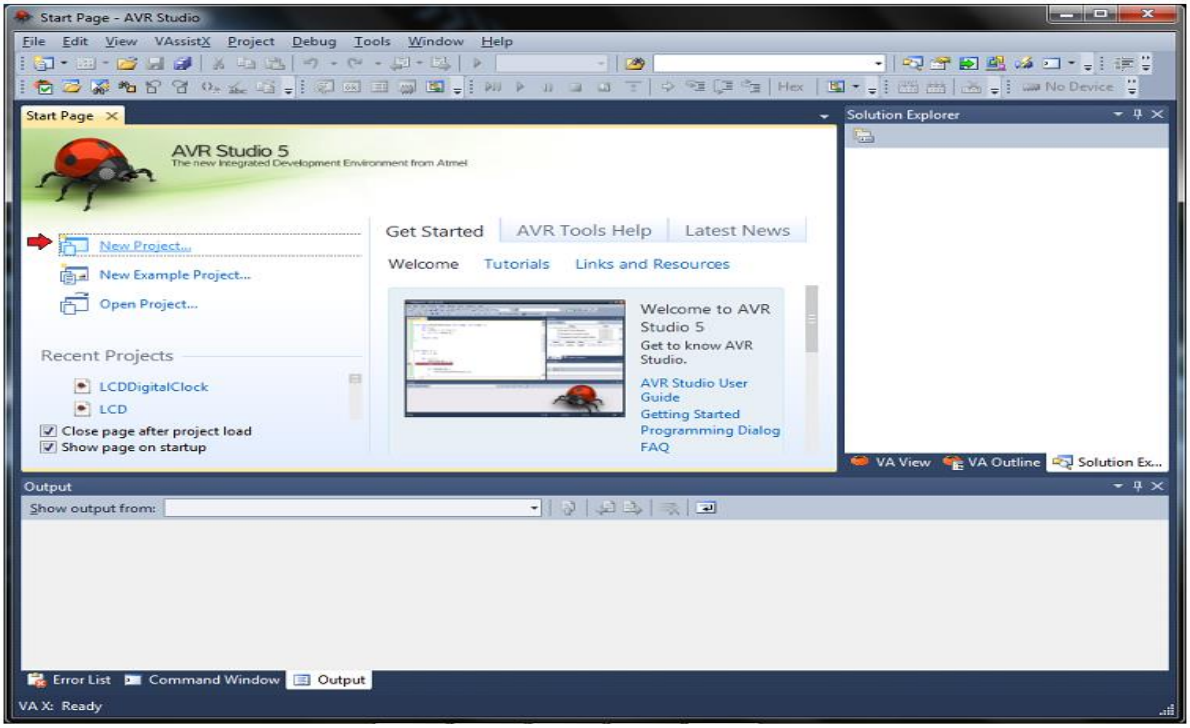
To build (compile, assemble, and link) an application in AVR Studio, you must:

1. Select Project
2. Select Build - Rebuild all target files or Build target.
3. AVR Studio compiles, assembles, and links the files in your project.

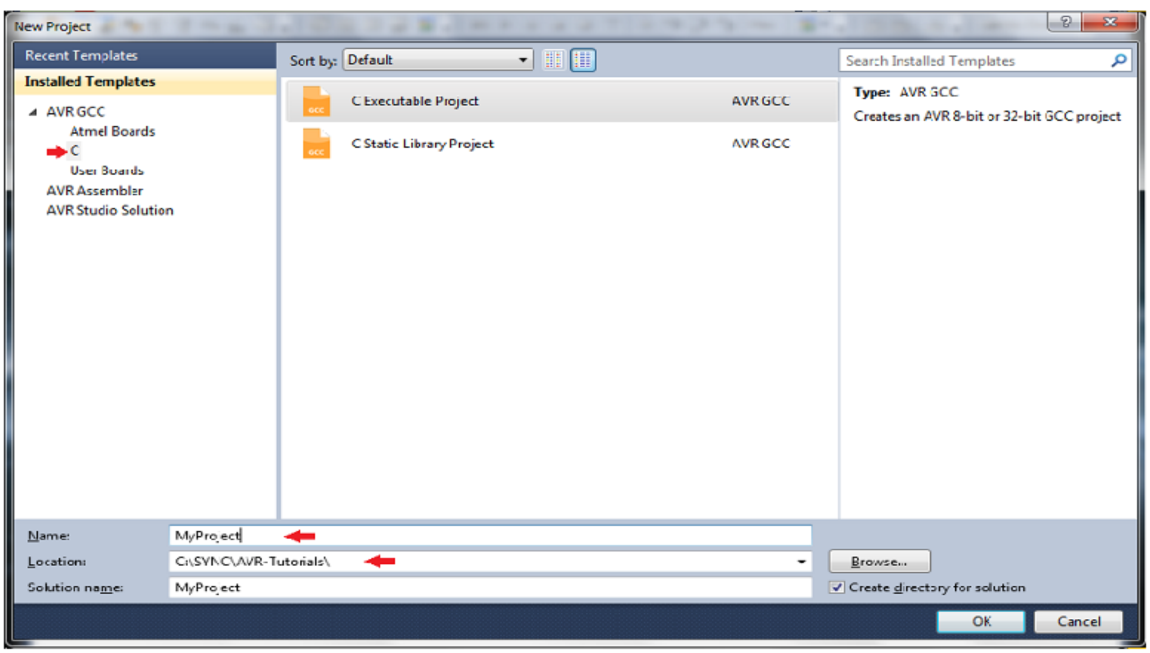
11.6 CREATING OWN APPLICATION / NEW PROJECT IN AVR STUDIO

To create a new project in AVR Studio, you must:

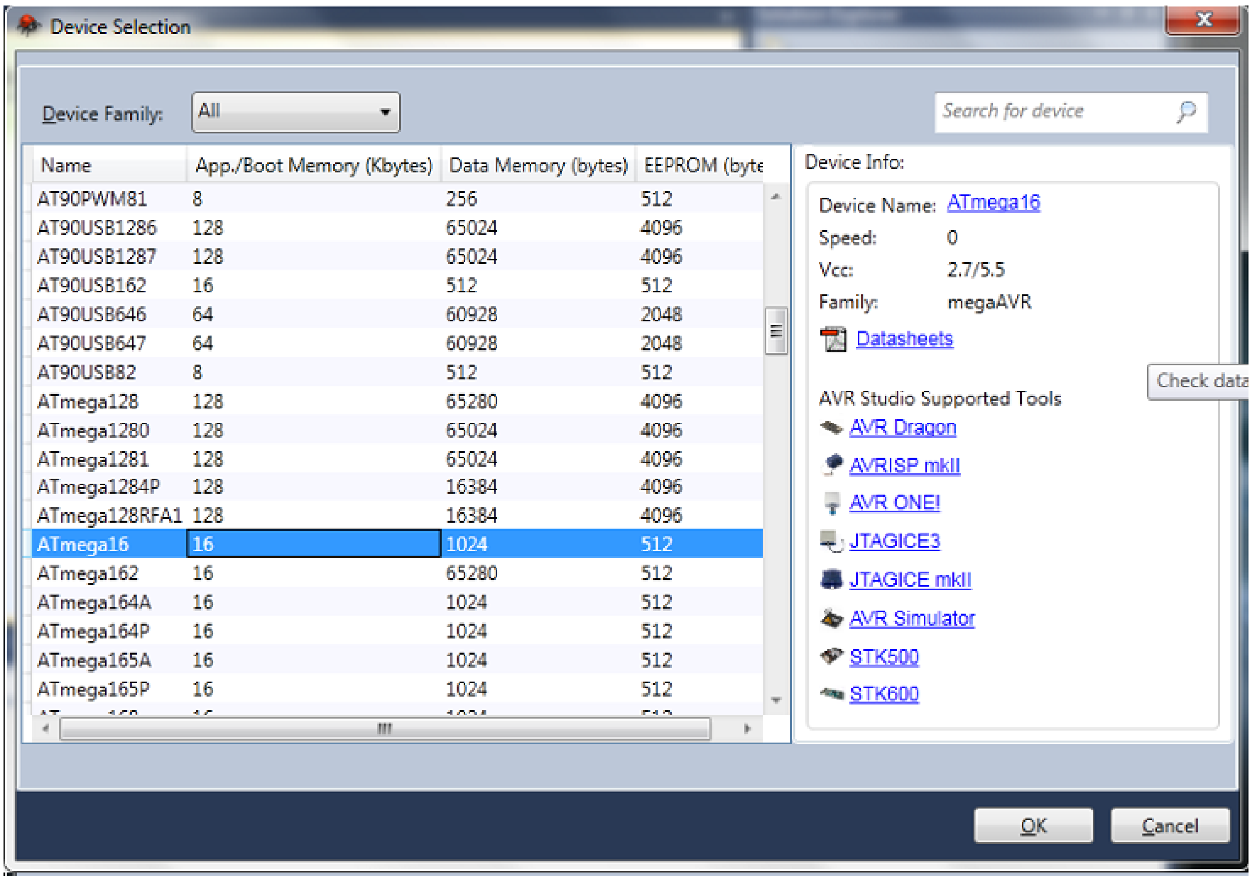
1. Open AVR Studio 5



2. Select Project - New Project.

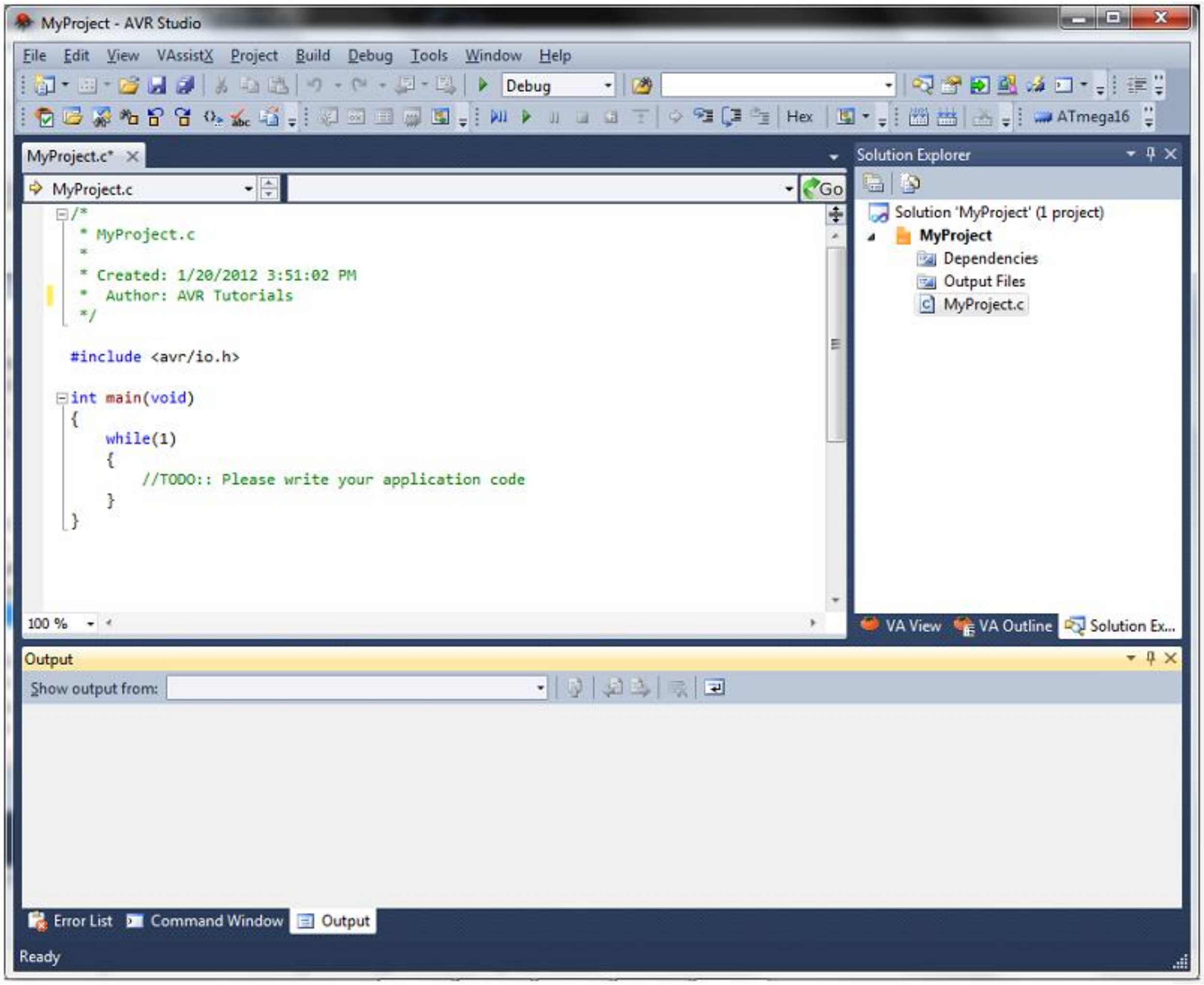


3. Write name of Project in the space given on the right hand side of label Name.
4. Similarly, write the location of where you want the project to be saved (Browsing to the location and selecting it will also do)
5. Click on OK if you are sure of the above said steps have been done and are correctly done.
6. Below is the device selection screen for AVR Studio 5. Scroll down and select the microcontroller you will be using.



7. This is the AVR studio 5 editor where you type your C program. The editor starts your C program for you by providing you with the structure shown in the editor of the figure below.
8. Save the project once done with programming.
9. Build the target to generate the machine understandable hex code and other binaries.

10. If all the above steps have been followed correctly and after building the target no errors are coming then the necessary above said files get generated. Otherwise one has to debug the code.



11.7 DEBUGGING A PROJECT IN AVR STUDIO 5

The AVR Studio 5 Simulator has the following features:

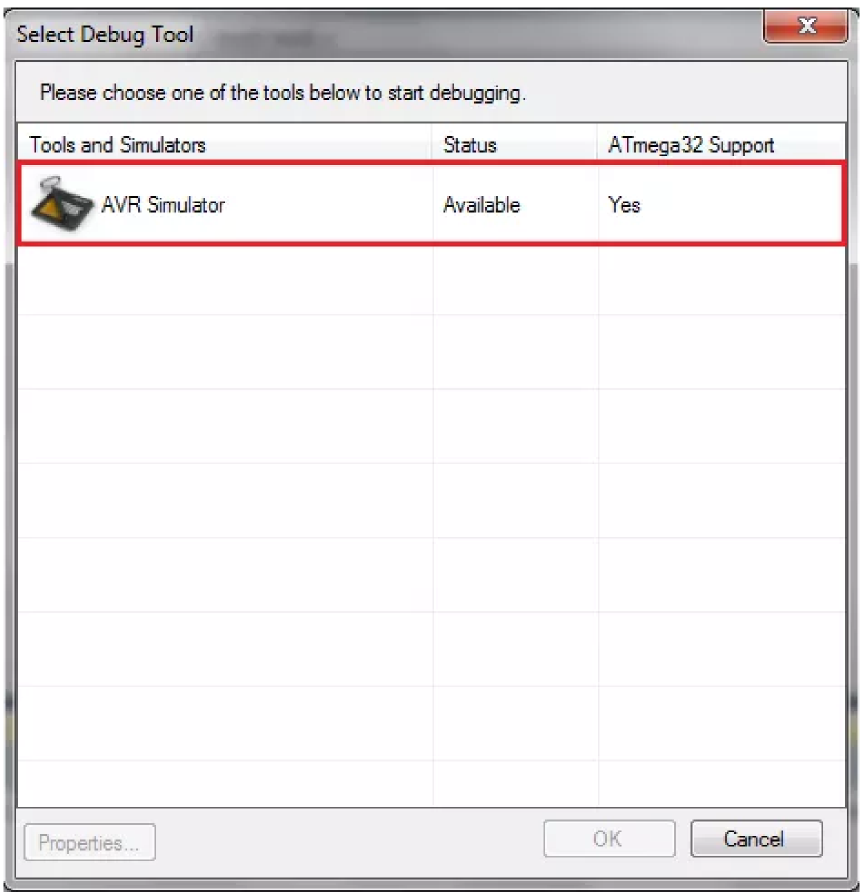
- It supports software emulation of any real AVR device without actually connecting it.
- It gives access to all the peripherals of the real MCU but no external devices.
- So if you want to give external signals, you need to do it yourself, either by manually updating the registers or creating a stimuli file.

Now, let's take the following code example to explain the functionality of the AVR Simulator.

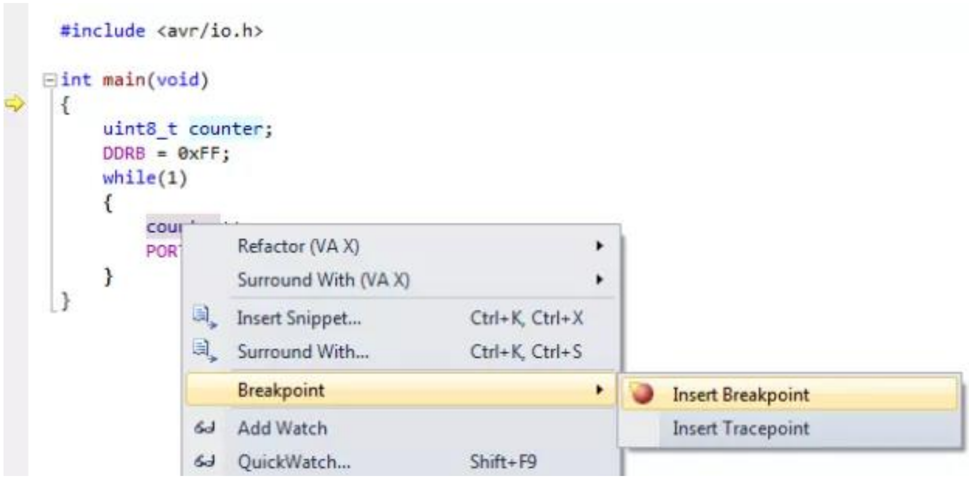
```
#include <avr/io.h>

intmain(void)
{
   uint8_t counter;
   DDRB = 0xFF;
   while(1)
   {
      counter++;    // insert breakpoint here <-----
      PORTB = counter;
   }
}
```

Debugging the code using Simulator:

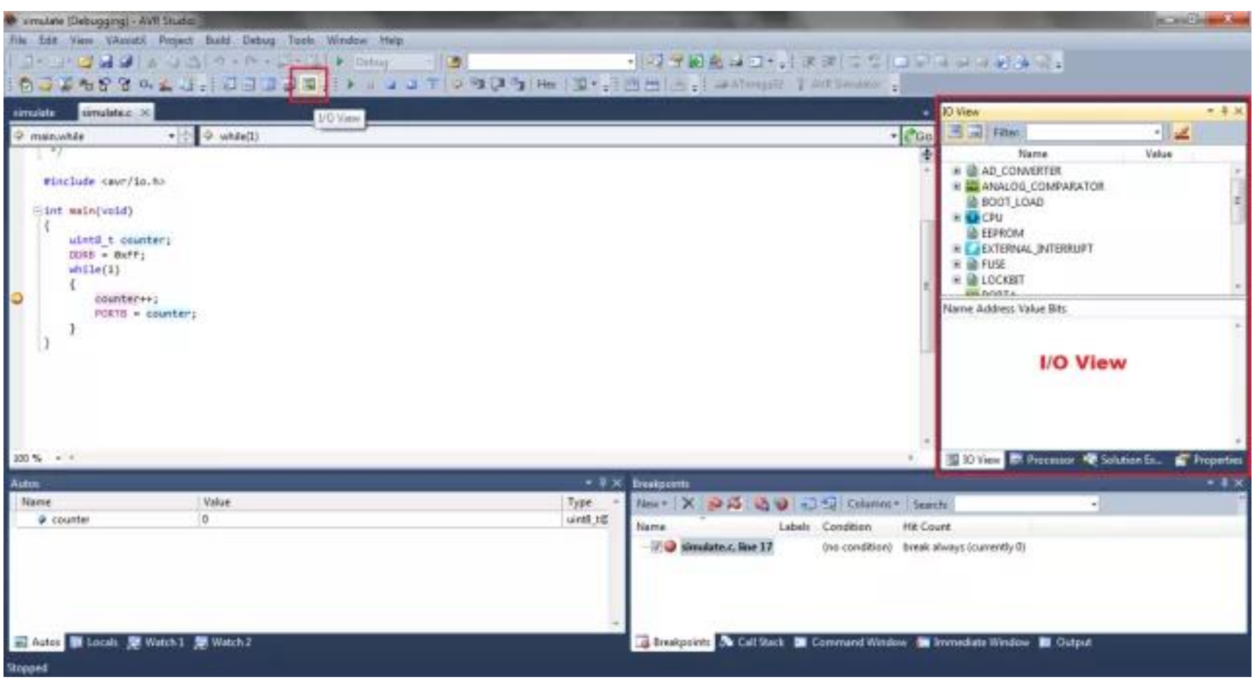
- Now, click on the **Debug** menu and then click on **Start Debugging and Break**. If initially no debugger is chosen, AVR Studio 5 will ask you to choose a **Debug Tool**. **AVR Simulator** is always an option there. Choose it and click **OK**.



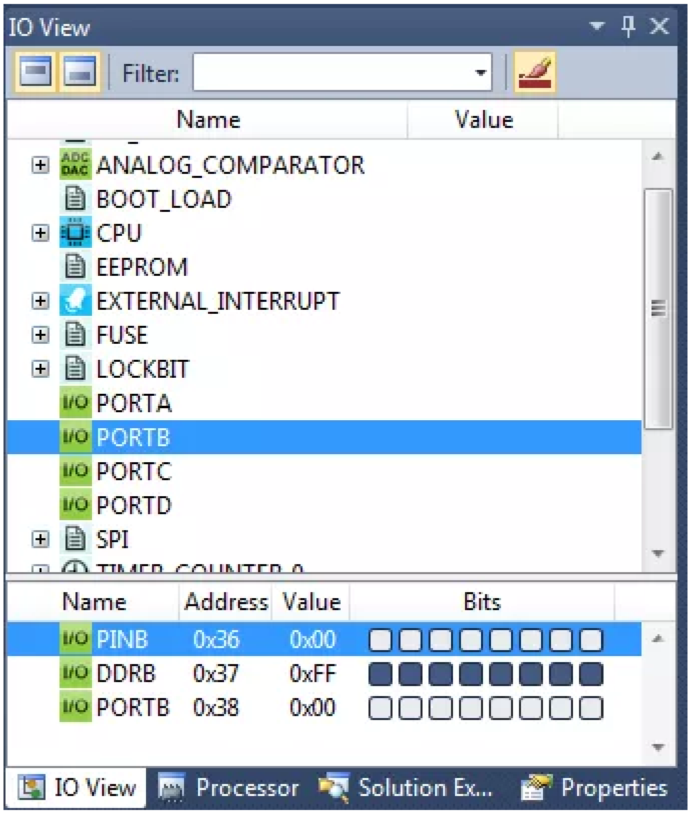
- After this, debugging starts and halts in the beginning of main(). You can see a yellow arrow mark determining the current executing line.
  Let's place a breakpoint in the main and start execution. Highlight the variable counter in counter++, right click it, go to **Breakpoint** and then click on **Insert Breakpoint**



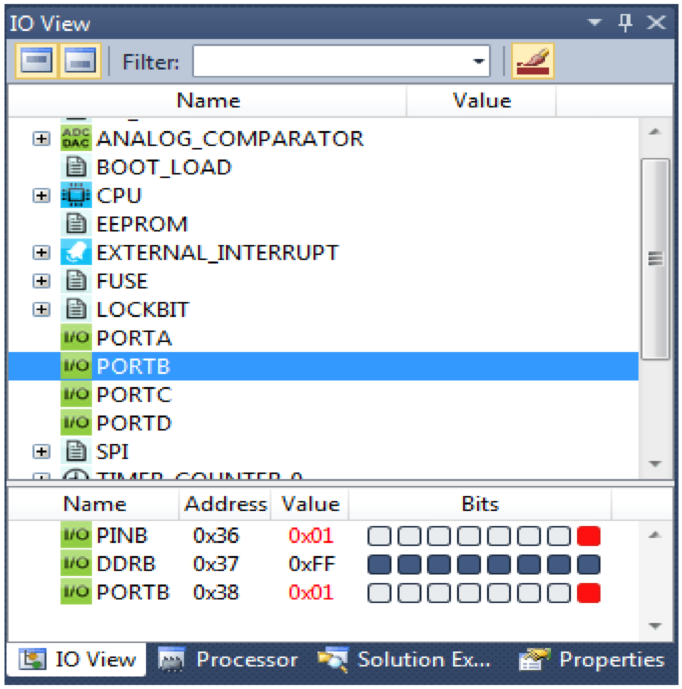
- Now press the **play** button (**F5**) or click on **Continue** from the **Debug**menu to run to the breakpoint.
- Now look at the affected registers in the **I/O view**. If you don't have the I/O View open, you can select it from the **Debug** toolbar or from the**Debug** windows menu.



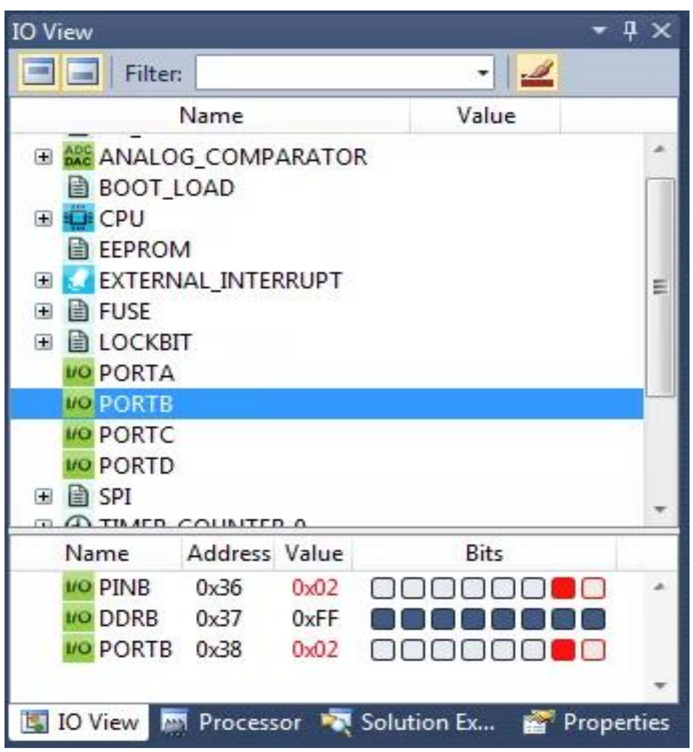
- All the peripheral features are mentioned over here. We can monitor any changes from the software and also manipulate the values to provide input.
- Now, since the counter changes the value of PORTB, scroll down in the I/O View and click on PORTB.

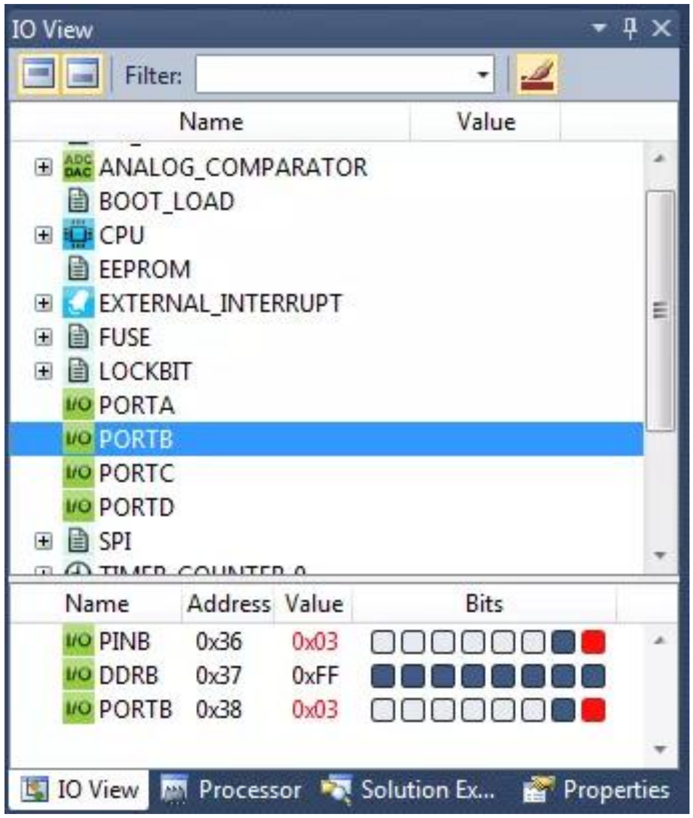


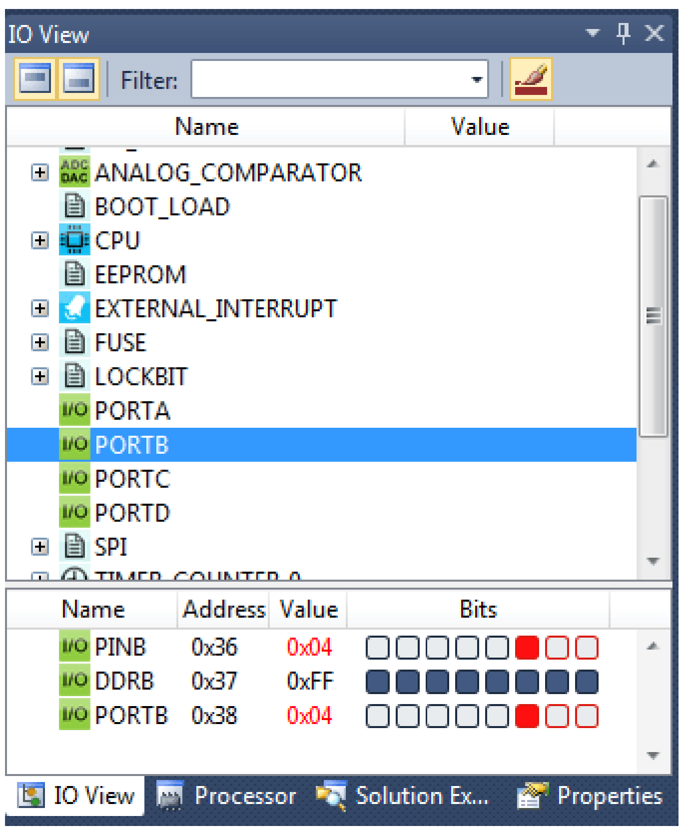
- Upon clicking PORTB, you can see the three registers assigned for PORTB operations, PINB, DDRB and PORTB. You can also view their current values.
- A solid block represents ‘1’ whereas a blank block represents ‘0’.
- Since it the beginning of main(), we defined DDRB = 0xFF, all the blocks are filled. You can also look at its value there.
- Now, press the play button. The loop iterates once and stops at the breakpoint. You can see that values of PINB and PORTB have changed to 0x01. This is because after one iteration, counter = 1.



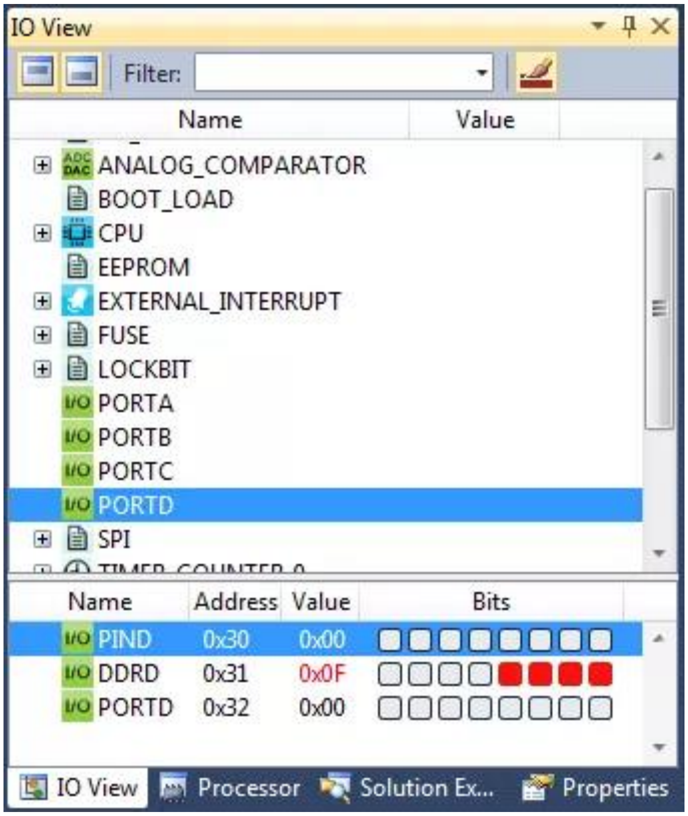
- The red block indicates that there has been a change in the value of the bit. If it's a solid red block, a change has been there from 0 to 1. If it's just a red outline, it's the other way round.
- Once again click on play. You will be able to see the following sequences.




IO View
Filter:
Name
Value
ANALOG_COMPARATOR
BOOT_LOAD
CPU
EEPROM
EXTERNAL_INTERRUPT
FUSE
LOCKBIT
PORTA
PORTB
PORTC
PORTD
SPI
Name
Address
Value
Bits
PINB
0x36
0x03
DDRB
0x37
0xFF
PORTB
0x38
0x03
IO View
Processor
Solution Ex...
Properties


IO View
Filter:
Name
Value
ANALOG_COMPARATOR
BOOT_LOAD
CPU
EEPROM
EXTERNAL_INTERRUPT
FUSE
LOCKBIT
PORTA
PORTB
PORTC
PORTD
SPI
Name
Address
Value
Bits
PINB
0x36
0x04
DDRB
0x37
0xFF
PORTB
0x38
0x04
IO View
Processor
Solution Ex...
Properties

- Now, if you want to change some other registers (apart from the ones changed by the code), simply click on the corresponding register and change its value.
- Say for example you want to change the value of DDRD. Click on PORTD and then give any value you want. You can also click on the corresponding bits to toggle the values.



So now we are done with the basics of AVR Studio 5. There are more advanced features of debugging in AVR Studio 5 which includes In-System Debugging which is a kind of runtime debugging unlike the software emulation that we learnt in this post.

11.8 EMBEDDED C

Use of embedded processors in passenger cars, mobile phones, medical equipment, aerospace systems and defense systems is widespread, and even everyday domestic appliances such as dish washers, televisions, washing machines and video recorders now include at least one such device.

Because most embedded projects have severe cost constraints, they tend to use low-cost processors like the 8051 family of devices considered in this book. These popular chips have very limited resources available most such devices have around 256 bytes (not megabytes!) of RAM, and the available processor power is around 1000 times less than that of a desktop processor. As a result, developing embedded software presents significant new challenges, even for experienced desktop programmers. If you have some programming experience - in C, C++ or Java - then this book and its accompanying CD will help make your move to the embedded world as quick and painless as possible.

12. PROGRAMMING CODE

USBasp is awesome USB based programmer for the AVR. In this tutorial we will see how to use AVRdude for burning hex files into AVR microcontroller using USBasp.

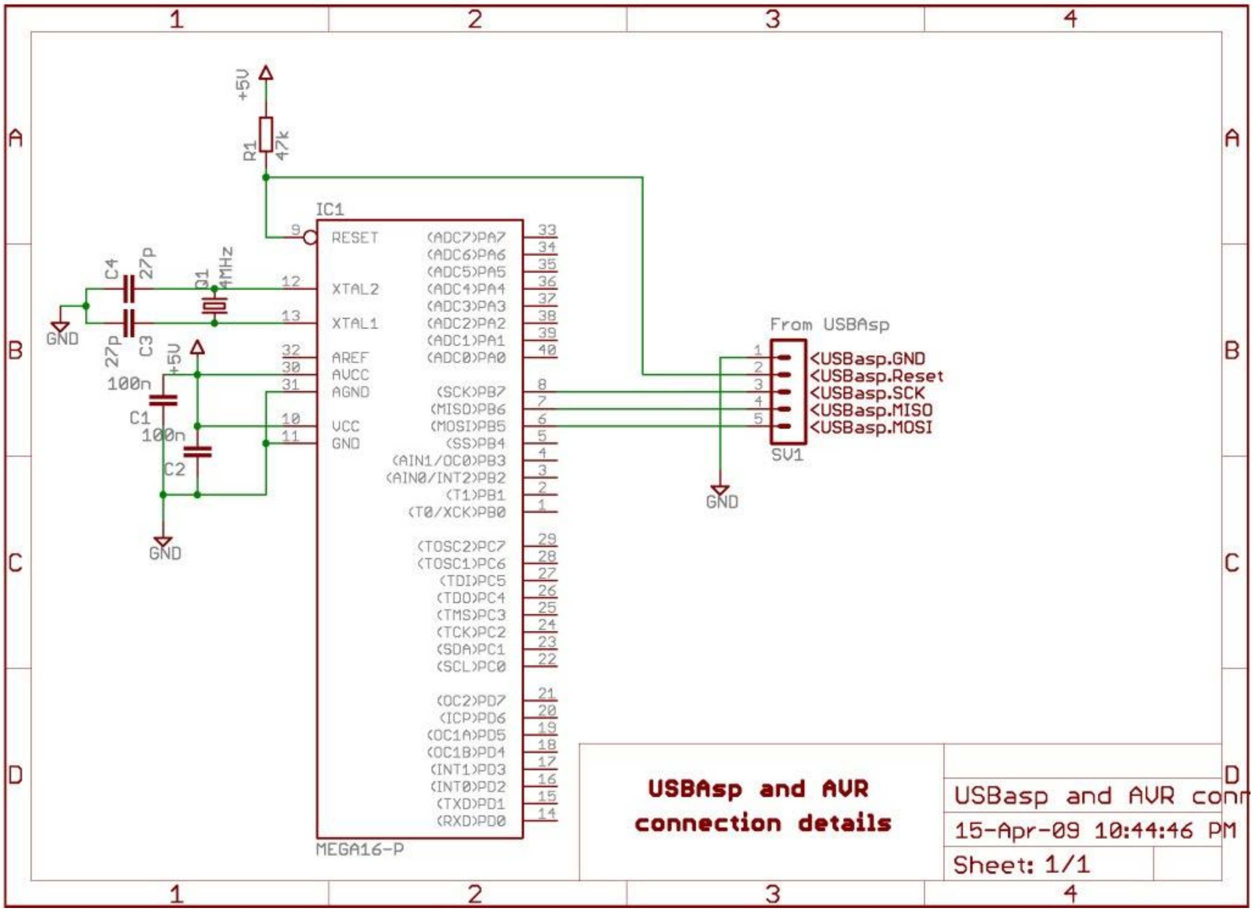
AVRdude is not readily available for windows installation. Windows version of AVRdude is bundled with WinAVR compiler

Introduction:

In order to program any microcontroller you need the .HEX file. It is nothing but the machine code for the microcontroller. This file is generated by the corresponding assembler software, which converts assembly code into machine code. Assembly can be produced by third party cross compiler software or can be handwritten.

Connections:

- ✓ Connect the USBasp to PC.
- ✓ Connect SPI programming pins of USBasp to the AVR microcontroller. Following figure shows sample schematic diagram, if you have different AVR, then connect MOSI,MISO, SCK, RESET and GND pins of that uC to corresponding pins of USBasp.



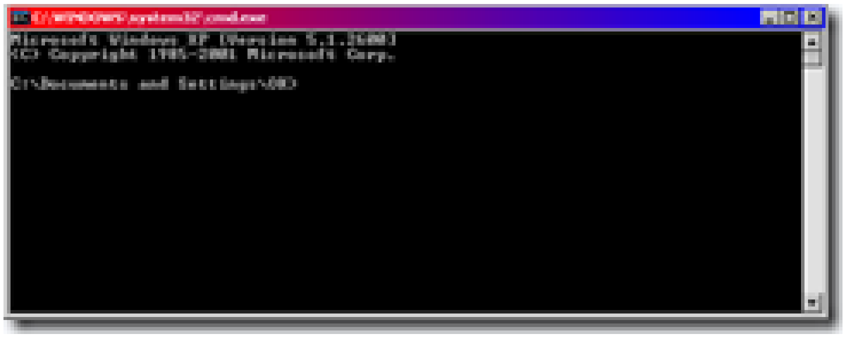
- ✓ Give +5V supply to the microcontroller.
- ✓ **VIMP:** If you are burning a fresh microcontroller, close the Slow Serial Clock jumper of USBasp. Since many brand new microcontrollers are factory programmed for internal 1MHz oscillator. USBasp uses very high speed serial clock for faster programming. Thus you will have to specifically tell USBasp to use slow serial clock. This setting is done by above mentioned jumper.

NOTE: If you have uC which has internal oscillator enabled and after the programming you are not planning to change its fuse bits back to external clock setting, then you can skip the crystal.

Executing AVRdude:

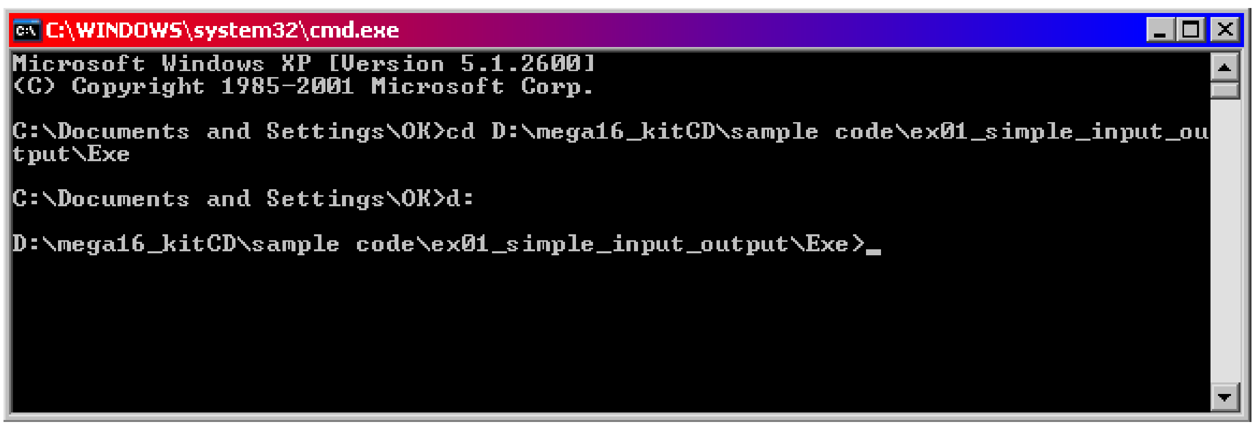
Fortunately AVRdude is command line tool, so that you can be very sure of what you are doing with your uC Or Unfortunately AVRdude is command line tool, so you will have to spend little time to get familiar with it .

- ✓ Open the command prompt. (Press WinKey + R. Run dialogbox will appear. Type *cmd* and press enter.)



- ✓ Navigate to the directory where .hex file is located. For example :

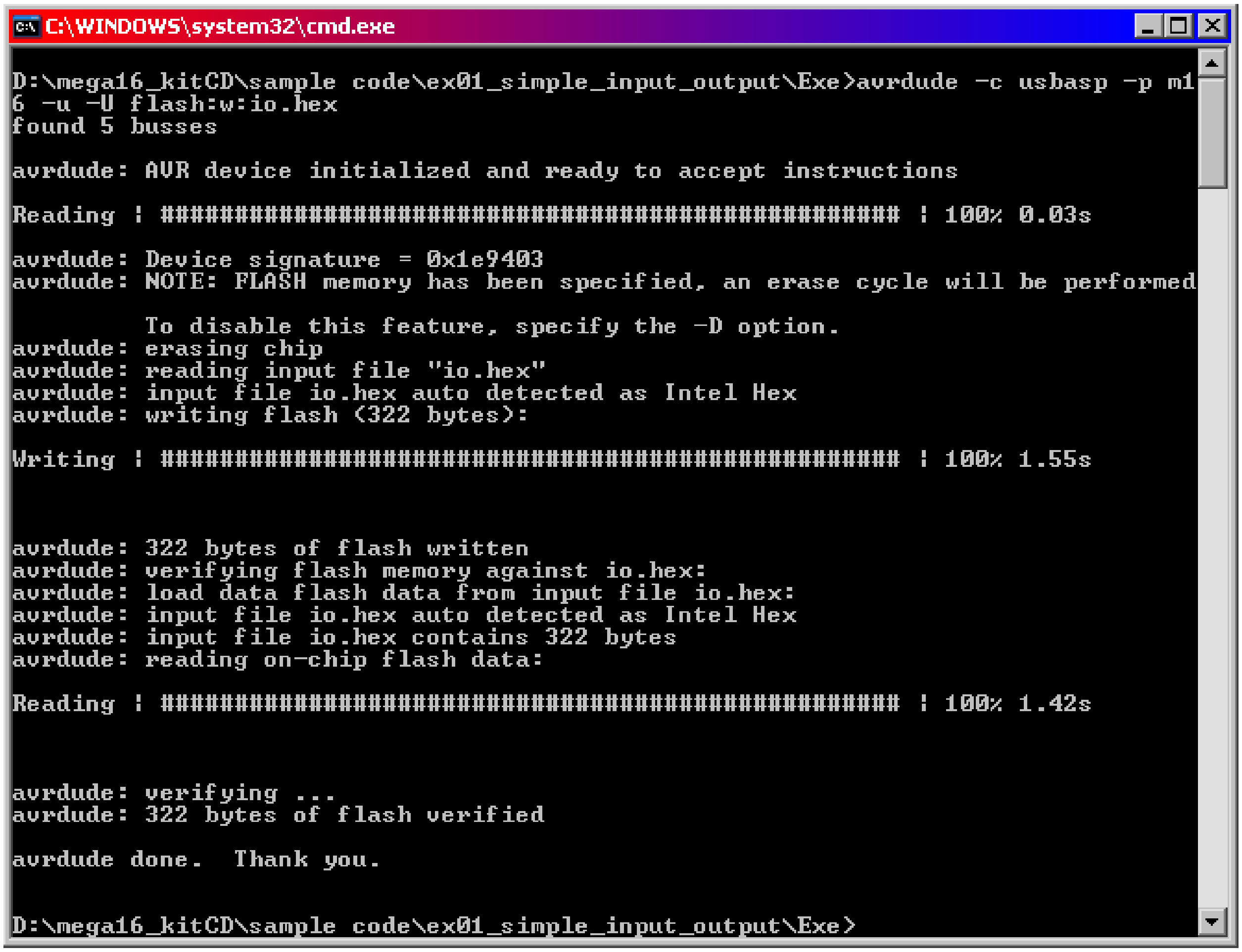
> cd D:mega16_kitCDsample codeex01_simple_input_outputExe



  - To burn the hex file enter following command. Consider for example name of my hex file is*io.hex* :

> avrdude –c usbasp –p m16 –u –U flash:w:io.hex

You should see something like this:



```
C:\WINDOWS\system32\cmd.exe

D:\mega16_kitCD\sample code\ex01_simple_input_output\Exe>avrdude -c usbasp -p m1
6 -u -U flash:w:io.hex
found 5 busses

avrdude: AVR device initialized and ready to accept instructions

Reading | ################################################## | 100% 0.03s

avrdude: Device signature = 0x1e9403
avrdude: NOTE: FLASH memory has been specified, an erase cycle will be performed

         To disable this feature, specify the -D option.
avrdude: erasing chip
avrdude: reading input file "io.hex"
avrdude: input file io.hex auto detected as Intel Hex
avrdude: writing flash (322 bytes):

Writing | ################################################## | 100% 1.55s



avrdude: 322 bytes of flash written
avrdude: verifying flash memory against io.hex:
avrdude: load data flash data from input file io.hex:
avrdude: input file io.hex auto detected as Intel Hex
avrdude: input file io.hex contains 322 bytes
avrdude: reading on-chip flash data:

Reading | ################################################## | 100% 1.42s



avrdude: verifying ...
avrdude: 322 bytes of flash verified

avrdude done.  Thank you.


D:\mega16_kitCD\sample code\ex01_simple_input_output\Exe>
```

avrdude –c usbasp –p m16 –u –U flash:w:io.hex

-c : Indicates the programmer type. Since we are using the USBasp programmer, argument "usbasp" is mentioned.

-p : Processor. We are using ATmega16, hence "m16". Note ATmega16 has two variants, one is "ATmega16L" (slow speed version) and "ATmega16" normal 16MHz version. However their device signature is same and hence you will have to use "m16" as parameter for both the AVRs. This applies to all AVRs having "L" variants.

-u : Disables the default behavior of reading out the fuses three times before programming, then verifying at the end of programming that the fuses have not changed. Always use this option. Many times it happens that we forget to switch on the AVR's +5V power supply, then at the end of programming cycle, avrdude detects inconsistent fuses and tries to reprogram them. Since there is no power supply, fuses gets programmed incorrectly and entire microcontroller gets screwed up(means becomes useless). Thus always use this option.

-U : memtype:op:filename[:format]

Perform a memory operation. Multiple '-U' options can be specified in order to operate on multiple memories on the same command-line invocation.

13. HARDWARE TESTING

12.1 CONTINUITY TEST:

In electronics, a continuity test is the checking of an electric circuit to see if current flows (that it is in fact a complete circuit). A continuity test is performed by placing a small voltage (wired in series with an LED or noise-producing component such as a piezoelectric speaker) across the chosen path. If electron flow is inhibited by broken conductors, damaged components, or excessive resistance, the circuit is "open".

Devices that can be used to perform continuity tests include multi meters which measure current and specialized continuity testers which are cheaper, more basic devices, generally with a simple light bulb that lights up when current flows.

An important application is the continuity test of a bundle of wires so as to find the two ends belonging to a particular one of these wires; there will be a negligible resistance between the "right" ends, and only between the "right" ends.

This test is the performed just after the hardware soldering and configuration has been completed. This test aims at finding any electrical open paths in the circuit after the soldering. Many a times, the electrical continuity in the circuit is lost due to improper soldering, wrong and rough handling of the PCB, improper usage of the soldering iron, component failures and presence of bugs in the circuit diagram. We use a multi meter to perform this test. We keep the multi meter in buzzer mode and connect the ground terminal of the multi meter to the ground. We connect both the terminals across the path that needs to be checked. If there is continuation then you will hear the beep sound.

12.2 POWER ON TEST:

This test is performed to check whether the voltage at different terminals is according to the requirement or not. We take a multi meter and put it in voltage mode. Remember that this test is performed without microcontroller. Firstly, we check the output of the transformer, whether we get the required 12 v AC voltage.

Then we apply this voltage to the power supply circuit. Note that we do this test without microcontroller because if there is any excessive voltage, this may lead to damaging the controller. We check for the input to the voltage regulator i.e., are we getting an input of 12v and an output of 5v. This 5v output is given to the microcontrollers' 40^{th} pin. Hence we check for the voltage level at 40^{th} pin. Similarly, we check for the other terminals for the required voltage. In this way we can assure that the voltage at all the terminals is as per the requirement.

14. RESULT

We have implemented project with 100% accuracy, after successfully connecting the Project setup to the web server through Wi-Fi the data of sensor are sent to the web server for monitoring of the system. Web server page which will allow us to monitor and control the Home Automation system. By entering the assigned IP address in the web browser this web server page will appear. The web server gives the information about the Home Appliances in different places of the house, which we can control remotely.

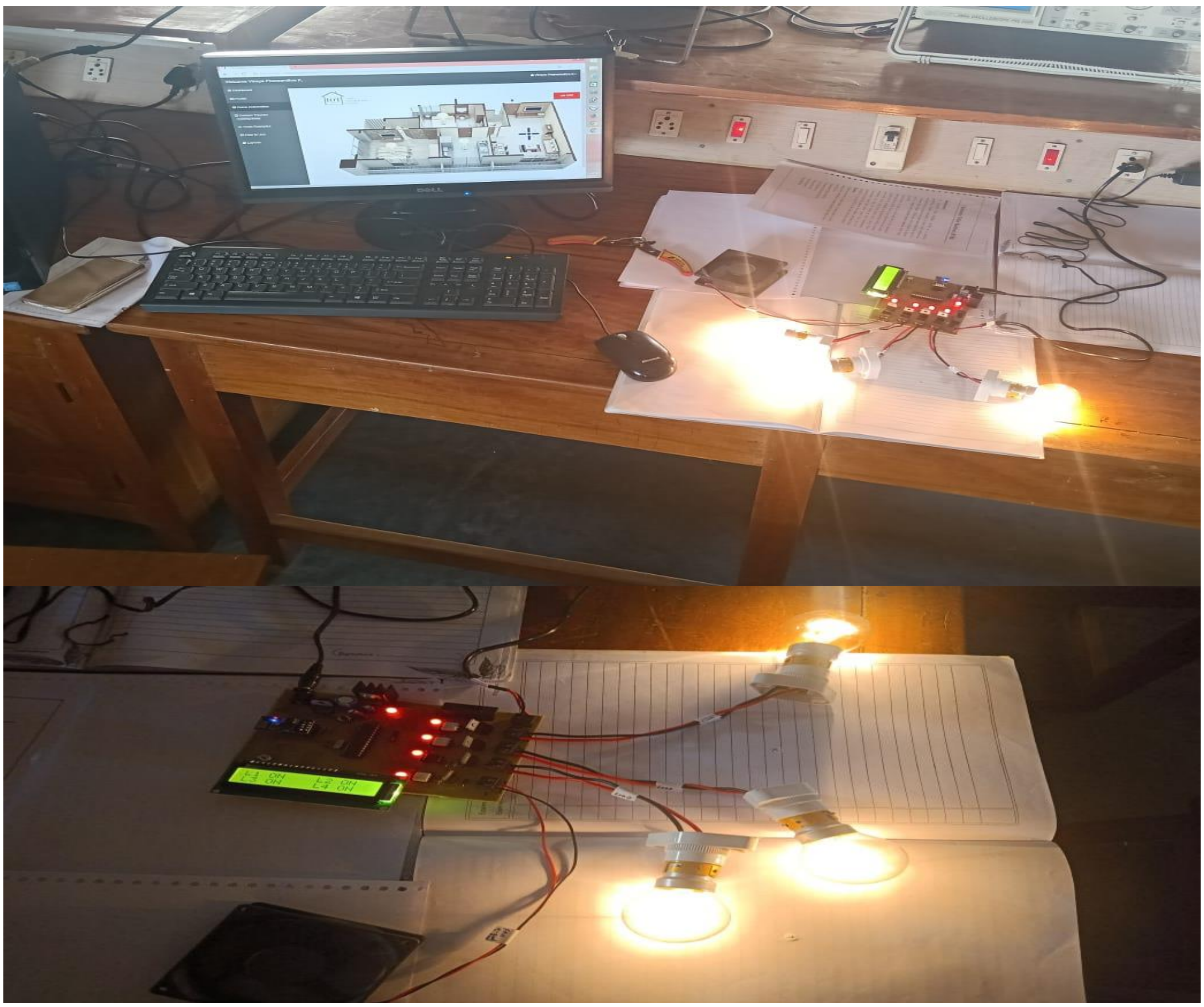


Fig (14) Prototype of the Project

15. CONCLUSION

The Project which we designed has its focal point on controlling different home appliances providing 100% efficiency. Due to advancement in technology, Wi-Fi network is easily available in all places like home, Office Building and Industrial Building so proposed wireless network easily controlled using any Wi-Fi network through web server. The wiring cost is reduced. Since less wiring is required for the switches. This also eliminates power consumption inside the building when the loads were in off conditions. This system is also platform independent allowing any web browser in any platform to connect ESP8266. The system is fully functional through android application known as “ESP8266 Wi-Fi control”. The delay to turn ON is 3 sec and turn OFF is 2 sec for any load.

16. BIBLIOGRAPHY

REFERNCES:

1. Internet of Things: A Hands-On Approach by Arsheep Bahga
2. AVR Microcontroller and Embedded Systems: Using Assembly and C, 1e
   by Mazidi / Naimi / Naimi
3. The AVR Microcontroller and Embedded Systems Using Assembly and C: Using Arduino Uno and Atmel Studio by Sarmad Naimi.
4. Programming and Customizing the AVR Microcontroller by Dhananjay Gadre.
5. Programming & Interfacing Atmel AVR Microcontrollers by Kevin Schultz
6. ATMEGA328 Data Sheets.

WEBSITES

- www.atmel.com
- www.beyondlogic.org
- www.wikipedia.org
- www.alldatasheets.com