**EMBEDDED SYSTEMS**

**INTRODUCTION TO EMBEDDED SYSTEMS**

An embedded system can be defined as a computing device that does a specific focused job. Appliances such as the air-conditioner, VCD player, DVD player, printer, fax machine, mobile phone etc. are examples of embedded systems. Each of these appliances will have a processor and special hardware to meet the specific requirement of the application along with the embedded software that is executed by the processor for meeting that specific requirement. The embedded software is also called “firm ware”. The desktop/laptop computer is a general purpose computer. You can use it for a variety of applications such as playing games, *word* processing, accounting, software development and so on. In contrast, the software in the embedded systems is always fixed listed below:

Embedded systems do a very specific task, they cannot be programmed to do different things. . Embedded systems have very limited resources, particularly the memory. Generally, they do not have secondary storage devices such as the CDROM or the floppy disk. Embedded systems have to work against some deadlines. A specific job has to be completed within a specific time. In some embedded systems, called real-time systems, the deadlines are stringent. Missing a deadline may cause a catastrophe-loss of life or damage to property. Embedded systems are constrained for power. As many embedded systems operate through a battery, the power consumption has to be very low.

Some embedded systems have to operate in extreme environmental conditions such as very high temperatures and humidity.

**Application Areas**

Nearly 99 per cent of the processors manufactured end up in embedded systems. The embedded system market is one of the highest growth areas as these systems are used in very market segment- consumer electronics, office automation, industrial automation, biomedical engineering, wireless communication, data communication, telecommunications, transportation, military and so on.

**Consumer appliances**:

At home we use a number of embedded systems which include digital camera, digital diary, DVD player, electronic toys, microwave oven, remote controls for TV and air-conditioner, VCO player, video game consoles, video recorders etc. Today’s high-tech car has about 20 embedded systems for transmission control, engine spark control, air-conditioning, navigation etc. Even wristwatches are now becoming embedded systems. The palmtops are powerful embedded systems using which we can carry out many general-purpose tasks such as playing games and word processing.

**Office automation:** The office automation products using em embedded systems are copying machine, fax machine, key telephone, modem, printer, scanner etc.

**Industrial automation**: Today a lot of industries use embedded systems for process control. These include pharmaceutical, cement, sugar, oil exploration, nuclear energy, electricity generation and transmission. The embedded systems for industrial use are designed to carry out specific tasks such as monitoring the temperature, pressure, humidity, voltage, current etc., and then take appropriate action based on the monitored levels to control other devices or to send information to a centralized monitoring station. In hazardous industrial environment, where human presence has to be avoided, robots are used, which are programmed to do specific jobs. The robots are now becoming very powerful and carry out many interesting and complicated tasks such as hardware assembly.

**Medical electronics**: Almost every medical equipment in the hospital is an embedded system. These equipments include diagnostic aids such as ECG, EEG, blood pressure measuring devices, X-ray scanners; equipment used in blood analysis, radiation, colonoscopy, endoscopy etc. Developments in medical electronics have paved way for more accurate diagnosis of diseases.

**Computer networking**: Computer networking products such as bridges, routers, Integrated Services Digital Networks (ISDN), Asynchronous Transfer Mode (ATM), X.25 and frame relay switches are embedded systems which implement the necessary data communication protocols. For example, a router interconnects two networks. The two networks may be running different protocol stacks. The router’s function is to obtain the data packets from incoming pores, analyze the packets and send them towards the destination after doing necessary protocol conversion. Most networking equipments, other than the end systems (desktop computers) we use to access the networks, are embedded systems

**Telecommunications**: In the field of telecommunications, the embedded systems can be categorized as subscriber terminals and network equipment. The subscriber terminals such as key telephones, ISDN phones, terminal adapters, web cameras are embedded systems. The network equipment includes multiplexers, multiple access systems, Packet Assemblers Dissemblers (PADs), sate11ite modems etc. IP phone, IP gateway, IP gatekeeper etc. are the latest embedded systems that provide very low-cost voice communication over the Internet.

**Wireless technologies**: Advances in mobile communications are paving way for many interesting applications using embedded systems. The mobile phone is one of the marvels of the last decade of the 20’h century. It is a very powerful embedded system that provides voice communication while we are on the move. The Personal Digital Assistants and the palmtops can now be used to access multimedia services over the Internet. Mobile communication infrastructure such as base station controllers, mobile switching centers are also powerful embedded systems.

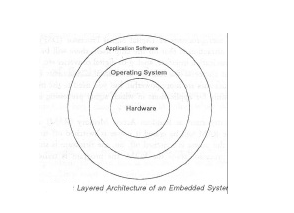
**Insemination:** Testing and measurement are the fundamental requirements in all scientific and engineering activities. The measuring equipment we use in laboratories to measure parameters such as weight, temperature, pressure, humidity, voltage, current etc. are all embedded systems. Test equipment such as oscilloscope, spectrum analyzer, logic analyzer, protocol analyzer, radio communication test set etc. are embedded systems built around powerful processors. Thank to miniaturization, the test and measuring equipment are now becoming portable facilitating easy testing and measurement in the field by field-personnel.

**Security:** Security of persons and information has always been a major issue. We need to protect our homes and offices; and also the information we transmit and store. Developing embedded systems for security applications is one of the most lucrative businesses nowadays. Security devices at homes, offices, airports etc. for authentication and verification are embedded systems. Encryption devices are nearly 99 per cent of the processors that are manufactured end up in~ embedded systems. Embedded systems find applications in every industrial segment- consumer electronics, transportation, avionics, biomedical engineering, manufacturing, process control and industrial automation, data communication, telecommunication, defense, security etc. Used to encrypt the data/voice being transmitted on communication links such as telephone lines. Biometric systems using fingerprint and face recognition are now being extensively used for user authentication in banking applications as well as for access control in high security buildings.

**Finance**: Financial dealing through cash and cheques are now slowly paving way for transactions using smart cards and ATM (Automatic Teller Machine, also expanded as Any Time Money) machines. Smart card, of the size of a credit card, has a small micro-controller and memory; and it interacts with the smart card reader! ATM machine and acts as an electronic wallet. Smart card technology has the capability of ushering in a cashless society. Well, the list goes on. It is no exaggeration to say that eyes wherever you go, you can see, or at least feel, the work of an embedded system!

**Overview of Embedded System Architecture**

Every embedded system consists of custom-built hardware built around a Central Processing Unit (CPU). This hardware also contains memory chips onto which the software is loaded. The software residing on the memory chip is also called the ‘firmware’. The embedded system architecture can be represented as a layered architecture as shown in Fig.

The operating system runs above the hardware, and the application software runs above the operating system. The same architecture is applicable to any computer including a desktop computer. However, there are significant differences. It is not compulsory to have an operating system in every embedded system. For small appliances such as remote control units, air conditioners, toys etc., there is no need *for* an operating system and you can write only the software specific to that application. For applications involving complex processing, it is advisable to have an operating system. In such a case, you need to integrate the application software with the operating system and then transfer the entire software on to the memory chip. Once the software is transferred to the memory chip, the software will continue to run *for* a long time you don’t need to reload new software.

Now, let us see the details of the various building blocks of the hardware of an embedded system. As shown in Fig. the building blocks are;

· Central Processing Unit (CPU)

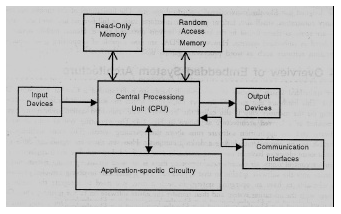
· Memory (Read-only Memory and Random Access Memory)

· Input Devices

· Output devices

· Communication interfaces

· Application-specific circuitry



**Central Processing Unit (CPU):**

The Central Processing Unit (processor, in short) can be any of the following: microcontroller, microprocessor or Digital Signal Processor (DSP). A micro-controller is a low-cost processor. Its main attraction is that on the chip itself, there will be many other components such as memory, serial communication interface, analog-to digital converter etc. So, for small applications, a micro-controller is the best choice as the number of external components required will be very less. On the other hand, microprocessors are more powerful, but you need to use many external components with them. D5P is used mainly for applications in which signal processing is involved such as audio and video processing.

**Memory:**

The memory is categorized as Random Access 11emory (RAM) and Read Only Memory (ROM). The contents of the RAM will be erased if power is switched off to the chip, whereas ROM retains the contents even if the power is switched off. So, the firmware is stored in the ROM. When power is switched on, the processor reads the ROM; the program is program is executed.

**Input devices**:

Unlike the desktops, the input devices to an embedded system have very limited capability. There will be no keyboard or a mouse, and hence interacting with the embedded system is no easy task. Many embedded systems will have a small keypad-you press one key to give a specific command. A keypad may be used to input only the digits. Many embedded systems used in process control do not have any input device *for* user interaction; they take inputs *from* sensors or transducers 1’fnd produce electrical signals that are in turn fed to other systems.

**Output devices**:

The output devices of the embedded systems also have very limited capability. Some embedded systems will have a *few* Light Emitting Diodes (LEDs) *to* indicate the health status of the system modules, or *for* visual indication of alarms. A small Liquid Crystal Display (LCD) may also be used to display *some* important parameters.

**Communication interfaces**:

The embedded systems may need to, interact with other embedded systems at they may have to transmit data to a desktop. To facilitate this, the embedded systems are provided with one or a *few* communication interfaces such as RS232, RS422, RS485, Universal Serial Bus (USB), IEEE 1394, Ethernet etc.

**Application-specific circuitry**:

Sensors, transducers, special processing and control circuitry may be required fat an embedded system, depending on its application. This circuitry interacts with the processor to carry out the necessary work. The entire hardware has to be given power supply either through the 230 volts main supply or through a battery. The hardware has to design in such a way that the power consumption is minimized.

**Microcontroller**

**Arduino Microcontroller**

The Arduino Uno is a microcontroller board based on the ATmega328 (datasheet). It has 14 digital input/output pins (of which 6 can be used as PWM outputs), 6 analog inputs, a 16 MHz ceramic resonator, a USB connection, a power jack, an ICSP header, and a reset button. It contains everything needed to support the microcontroller; simply connect it to a computer with a USB cable or power it with a AC-to-DC adapter or battery to get started.

**Features:**

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

**"Uno"** means one in Italian and is named to mark the upcoming release of Arduino 1.0. The Uno and version 1.0 will be the reference versions of Arduino, moving forward. The Uno is the latest in a series of USB Arduino boards, and the reference model for the Arduino platform; for a comparison with previous versions, see the index of Arduino boards.

**Summary**

Microcontroller ATmega328

Operating Voltage 5V

Input Voltage (recommended) 7-12V

Input Voltage (limits) 6-20V

Digital I/O Pins 14 (of which 6 provide PWM output)

Analog Input Pins 6

DC Current per I/O Pin 40 mA

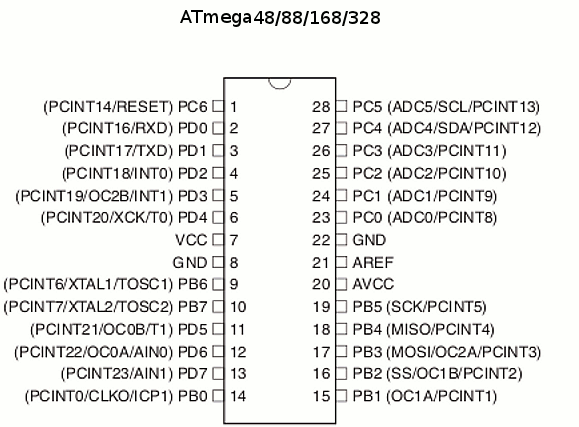
DC Current for 3.3V Pin 50 mA

Flash Memory 32 KB (ATmega328) of which 0.5 KB used by boot loader

SRAM 2 KB (ATmega328)

EEPROM 1 KB (ATmega328)

Clock Speed 16 MHz

**PIN DIAGRAM:**

**Power**

The Arduino Uno can be powered via the USB connection or with an external power supply. The power source is selected automatically.

External (non-USB) power can come either from an AC-to-DC adapter (wall-wart) or battery. The adapter can be connected by plugging a 2.1mm center-positive plug into the board's power jack. Leads from a battery can be inserted in the Gnd and Vin pin headers of the POWER connector.

The board can operate on an external supply of 6 to 20 volts. If supplied with less than 7V, however, the 5V pin may supply less than five volts and the board may be unstable. If using more than 12V, the voltage regulator may overheat and damage the board. The recommended range is 7 to 12 volts.

**The power pins are as follows:**

**VIN.** The input voltage to the Arduino board when it's using an external power source (as opposed to 5 volts from the USB connection or other regulated power source). You can supply voltage through this pin, or, if supplying voltage via the power jack, access it through this pin.

**5V**.This pin outputs a regulated 5V from the regulator on the board. The board can be supplied with power either from the DC power jack (7 - 12V), the USB connector (5V), or the VIN pin of the board (7-12V). Supplying voltage via the 5V or 3.3V pins bypasses the regulator, and can damage your board. We don't advise it.

**3V3**. A 3.3 volt supply generated by the on-board regulator. Maximum current draw is 50 mA.

**GND**. Ground pins.

**Memory**

The ATmega328 has 32 KB (with 0.5 KB used for the bootloader). It also has 2 KB of SRAM and 1 KB of EEPROM (which can be read and written with the EEPROM library).

**Input and Output**

**Pin Descriptions**

**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 unprogrammed, 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.

**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.

**ADC7:6 (TQFP and QFN/MLF Package Only)**

In the TQFP and QFN/MLF package, ADC7:6 serve as analog inputs to the A/D converter.

These pins are powered from the analog supply and serve as 10-bit ADC channels.**Communication**

The Arduino Uno has a number of facilities for communicating with a computer, another Arduino, or other microcontrollers. The ATmega328 provides UART TTL (5V) serial communication, which is available on digital pins 0 (RX) and 1 (TX). An ATmega16U2 on the board channels this serial communication over USB and appears as a virtual com port to software on the computer. The '16U2 firmware uses the standard USB COM drivers, and no external driver is needed. However, on Windows, a .inf file is required. The Arduino software includes a serial monitor which allows simple textual data to be sent to and from the Arduino board. The RX and TX LEDs on the board will flash when data is being transmitted via the USB-to-serial chip and USB connection to the computer (but not for serial communication on pins 0 and 1).

A Software Serial library allows for serial communication on any of the Uno's digital pins.

The ATmega328 also supports I2C (TWI) and SPI communication. The Arduino software includes a Wire library to simplify use of the I2C bus; see the documentation for details. For SPI communication, use the SPI library.

**Automatic (Software) Reset**

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

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

The Uno contains a trace that can be cut to disable the auto-reset. The pads on either side of the trace can be soldered together to re-enable it. It's labeled "RESET-EN". You may also be able to disable the auto-reset by connecting a 110 ohm resistor from 5V to the reset line; see this forum thread for details.

USB Over current Protection

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

**POWER SUPPLY**

**POWER SUPPLY:**

The input to the circuit is applied from the regulated power supply. The a.c. input i.e., 230V from the mains supply is step down by the transformer to 12V and is fed to a rectifier. The output obtained from the rectifier is a pulsating d.c voltage. So in order to get a pure d.c voltage, the output voltage from the rectifier is fed to a filter to remove any a.c components present even after rectification. Now, this voltage is given to a voltage regulator to obtain a pure constant dc voltage.

**230V AC**

**50Hz**

D.C Output

Regulator

# Filter

Bridge Rectifier

Step down transformer

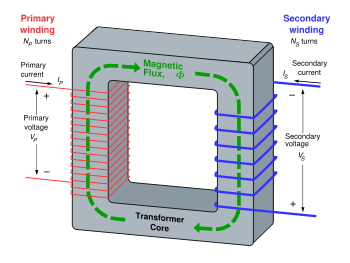
**Fig: Power supply**

**Transformer:** Usually, DC voltages are required to operate various electronic equipment and these voltages are 5V, 9V or 12V. But these voltages cannot be obtained directly. Thus the a.c input available at the mains supply i.e., 230V is to be brought down to the required voltage level. This is done by a transformer. Thus, a step down transformer is employed to decrease the voltage to a required level.

A transformer is an electrical device that transfers energy from one circuit to another by magnetic coupling with no moving parts. A transformer comprises two or more coupled windings, or a single tapped winding and, in most cases, a magnetic core to concentrate magnetic flux. A changing current in one winding creates a time-varying magnetic flux in the core, which induces a voltage in the other windings.

Basic principles:

A simple transformer consists of two electrical conductors called the primary winding and the secondary winding. Energy is coupled between the windings by the time-varying magnetic flux that passes through (links) both primary and secondary windings. Whenever the amount of current in a coil changes (including when the current is switched on or off), a voltage is induced in the neighboring coil. The effect, called mutual inductance, is an example of electromagnetic induction.

[](http://en.wikipedia.org/wiki/Image:Transformer3d_col3.svg)

If a time-varying voltage {v_P}\,is applied to the primary winding of N_P\,turns, a current will flow in it producing a magneto motive force (MMF). Just as an electromotive force (EMF) drives current around an electric circuit, so MMF tries to drive magnetic flux through a magnetic circuit. The primary MMF produces a varying magnetic flux \Phi_P\,in the core, and, with an open circuit secondary winding, induces a back electromotive force (EMF) in opposition to{v_P}\,. In accordance with Faraday's law of induction, the voltage induced across the primary winding is proportional to the rate of change of flux:

{v_P} = {N_P} \frac {d \Phi_P}{dt}     and     {v_S} = {N_S} \frac {d \Phi_S}{dt}

where

vP and vS are the voltages across the primary winding and secondary winding,

NP and NS are the numbers of turns in the primary winding and secondary winding,

dΦP / dt and dΦS / dt are the derivatives of the flux with respect to time of the primary and secondary windings.

Saying that the primary and secondary windings are perfectly coupled is equivalent to saying that \Phi_P = \Phi_S\,. Substituting and solving for the voltages shows that:

\frac{v_P}{v_S}=\frac{N_P}{N_S}

where

vp and vs are voltages across primary and secondary,

Np and Ns are the numbers of turns in the primary and secondary, respectively.

Hence in an ideal transformer, the ratio of the primary and secondary voltages is equal to the ratio of the number of turns in their windings, or alternatively, the voltage per turn is the same for both windings.

**Transformer losses arise from:**

**Winding resistance**

Current flowing through the windings causes resistive heating of the conductors (I2 R loss). At higher frequencies, skin effect and proximity effect create additional winding resistance and losses.

**Eddy currents**

Induced eddy currents circulate within the core, causing resistive heating. Silicon is added to the steel to help in controlling eddy currents. Adding silicon also has the advantage of stopping aging of the electrical steel that was a problem years ago.

**Hysteresis losses**

Each time the magnetic field is reversed, a small amount of energy is lost to hysteresis within the magnetic core. The amount of hysteresis is a function of the particular core material.

**Magnetostriction**

Magnetic flux in the core causes it to physically expand and contract slightly with the alternating magnetic field, an effect known as magnetostriction. This in turn causes losses due to frictional heating in susceptible ferromagnetic cores.

**Mechanical losses**

In addition to magnetostriction, the alternating magnetic field causes fluctuating electromagnetic forces between the primary and secondary windings. These incite vibrations within nearby metalwork, creating a familiar humming or buzzing noise, and consuming a small amount of power.

Stray losses

Not all the magnetic field produced by the primary is intercepted by the secondary. A portion of the leakage flux may induce eddy currents within nearby conductive objects, such as the transformer's support structure, and be converted to heat.

**Rectifier:**

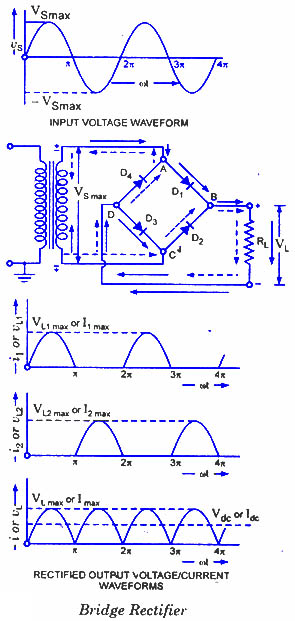
The output from the transformer is fed to the rectifier. It converts A.C. into pulsating D.C. The rectifier may be a half wave or a full wave rectifier. In this project, a bridge rectifier is used because of its merits like good stability and full wave rectification.



The Bridge rectifier is a circuit, which converts an ac voltage to dc voltage using both half cycles of the input ac voltage. The Bridge rectifier circuit is shown in the figure. The circuit has four diodes connected to form a bridge. The ac input voltage is applied to the diagonally opposite ends of the bridge. The load resistance is connected between the other two ends of the bridge.

For the positive half cycle of the input ac voltage, diodes D1 and D3 conduct, whereas diodes D2 and D4 remain in the OFF state. The conducting diodes will be in series with the load resistance RL and hence the load current flows through RL.

For the negative half cycle of the input ac voltage, diodes D2 and D4 conduct whereas, D1 and D3 remain OFF. The conducting diodes D2 and D4 will be in series with the load resistance RL and hence the current flows through RL in the same direction as in the previous half cycle. Thus a bi-directional wave is converted into a unidirectional wave.

****

**Filter:**

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.

**Voltage regulator:**

****As the name itself implies, it regulates the input applied to it. A voltage regulator is an electrical regulator designed to automatically maintain a constant voltage level. In this project, power supply of 5V and 12V are required. In order to obtain these voltage levels, 7805 and 7812 voltage regulators are to be used. The first number 78 represents positive supply and the numbers 05, 12 represent the required output voltage levels. The L78xx series of three-terminal positive regulators is available in TO-220, TO-220FP, TO-3, D2PAK and DPAK packages and several fixed output voltages, making it useful in a wide range of applications. These regulators can provide local on-card regulation, eliminating the distribution problems associated with single point regulation. Each type employs internal current limiting, thermal shut-down and safe area protection, making it essentially indestructible. If adequate heat sinking is provided, they can deliver over 1 A output current. Although designed primarily as fixed voltage regulators, these devices can be used with external components to obtain adjustable voltage and currents.

**INTRODUCTION TO SONAR**

****

**INTRODUCTION:**

Sonar:   
Sonar stands for **So**und **Na**vigation **R**anging. Sonar is used in navigation, forecasting weather, and for tracking aircraft, ships, submarines, and missiles. Sonar devices work by bouncing sound waves off objects to determine their location. A sonar unit consists of an ultrasonic transmitter and a receiver. On boats, the receiver is mounted on the bottom of the ship. To measure water depth, for instance, the transmitter sends out a short pulse of sound, and later, the receiver AT89S52ks up the reflected sound. The water depth is determined from the time elapsed between the emission of the ultrasonic sound and the reception of its reflection off the sea-floor. In the diagram below, a ship sends out ultrasonic waves (green) in order to detect schools of fish swimming beneath. The waves reflect off the fish (white), and return to the ship where they are detected and the depth of the fish is determined

**Ultrasonic Waves**

Humans can normally hear sound frequencies between 20 and 20,000 Hz (20 kHz). When a sound wave's frequency lies above 20 kHz, it is called an ultrasonic wave. While we cannot hear ultrasonic waves, we apply them in various technologies such as sonar systems, sonograms, surgical tools, and cleaning systems. Some animals also use ultrasonic waves in a specialized technique called echolocation that allows them to pinpoint objects and other animals, even in the dark.

The project uses 5 standard transistors to receive and transmit the ultrasound and a comparator to set the threshold echo detection level - so there are no special components other than the microcontroller. The ultrasonic transducers are standard 40 kHz types. Note that the internal oscillator of the MC micro is used and this saves two pins - that can be used for normal I/O,

Ultrasonic sensors (also known as transceivers when they both send and receive) work on a principle similar to radar or sonar which evaluate attributes of a target by interpreting the echoes from radio or sound waves respectively. Ultrasonic sensors generate high frequency sound waves and evaluate the echo which is received back by the sensor. Sensors calculate the time interval between sending the signal and receiving the echo to determine the distance to an object.

This technology can be used for measuring: wind speed and direction (anemometer), fullness of a tank and speed through air or water. For measuring speed or direction a device uses multiple detectors and calculates the speed from the relative distances to particulates in the air or water. To measure the amount of liquid in a tank, the sensor measures the distance to the surface of the fluid. Further applications include: [humidifiers](http://en.wikipedia.org/wiki/Humidifier), [sonar](http://en.wikipedia.org/wiki/Sonar), [medical ultrasonography](http://en.wikipedia.org/wiki/Medical_ultrasonography), burglar alarms and [non-destructive testing](http://en.wikipedia.org/wiki/Non-destructive_testing).

Systems typically use a transducer which generates sound waves in the ultrasonic range, above 18,000 hertz, by turning electrical energy into sound, then upon receiving the echo turn the sound waves into electrical energy which can be measured and displayed.

The technology is limited by the shapes of surfaces and the density or consistency of the material. For example foam on the surface of a fluid in a tank could distort a reading.

**Ultra sonic Range Finding:**

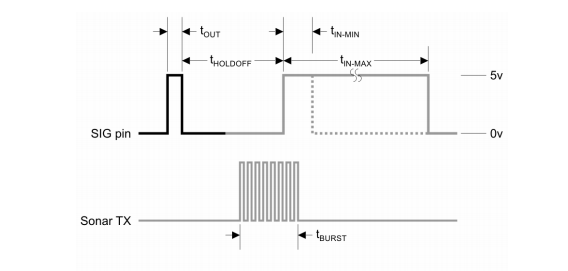
A common use of ultrasound is in [range finding](http://en.wikipedia.org/wiki/Range_finding); this use is also called [SONAR](http://en.wikipedia.org/wiki/SONAR), (sound navigation and ranging). This works similarly to [RADAR](http://en.wikipedia.org/wiki/RADAR) (radio detection and ranging): An ultrasonic pulse is generated in a particular direction. If there is an object in the path of this pulse, part or all of the pulse will be reflected back to the transmitter as an [echo](http://en.wikipedia.org/wiki/Echo_(phenomenon)) and can be detected through the receiver path. By measuring the difference in time between the pulse being transmitted and the echo being received, it is possible to determine how far away the object is.

The measured travel time of SONAR pulses in water is strongly dependent on the temperature and the salinity of the water. Ultrasonic ranging is also applied for measurement in air and for short distances. Such method is capable for easily and rapidly measuring the layout of rooms.

Although range finding underwater is performed at both sub-audible and audible frequencies for great distances (1 to several kilometers), ultrasonic range finding is used when distances are shorter and the accuracy of the distance measurement is desired to be finer. Ultrasonic measurements may be limited through barrier layers with large salinity, temperature or vortex differentials. Ranging in water varies from about hundreds to thousands of meters, but can be performed with centimeters to meters accuracy.

**Theory:**

The Ping sensor detects objects by emitting a short ultrasonic burst and then "listening" for the echo. Under control of a host microcontroller (trigger pulse), the sensor emits a short 40 kHz (ultrasonic) burst.

This burst travels through the air at about 1130 feet per second, hits an object and then bounces back to the sensor. The PING sensor provides an output pulse to the host that will terminate when the echo is detected; hence the width of this pulse corresponds to the distance to the target.

**Two Ultrasonic Sensor Types**

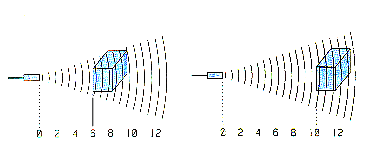
The following diagrams summarize the distinctions between proximity and ranging ultrasonic sensors:

**Proximity Detection**

An object passing anywhere within the preset range will be detected and generate an output signal. The detect point is independent of target size, material, or degree of reflectivity.

http://www.migatron.com/image/introwmf0.gif  
Objected detected - YES         Objected detected - NO

**Ranging Measurement:**

Precise distance(s) of an object moving to and from the sensor are measured via time intervals between transmitted and reflected bursts of ultrasonic sound. The example shows a target detected at six inches from sensor and moving to 10 inches. The distance change is continuously calculated and outputted.  


**Calculation for target finding:**

The time from transmission of the pulse to reception of the echo is the time taken for the signal energy to travel through the air to the object and back again. Since the speed of signal is constant through air measuring the echo reflection time lets you calculate the distance to the object using the DST equation:

Distance = (s \* t)/2 (in meters)

You need to divide by 2 as the distance is the round trip distance i.e. from transmitter to object and back again.

Where:

|  |  |
| --- | --- |
| s [m/s] | the speed of sound in air |
| t [s] | The round trip echo time. |

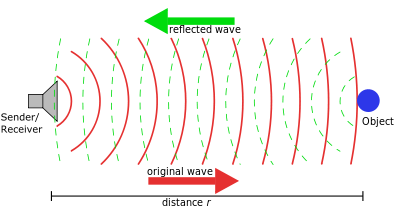
Some delay times:

|  |  |
| --- | --- |
| Round trip echo time | Distance |
| t = 588us | 10cm |
| t = 5.8ms | 1m |

The speed of sound in air is more or less constant at 330m/s (@ 0ºC) - it varies mainly with temperature (~340m/s @ 20ºC). In this project I am using a value of 340m/s i.e. it is assumed that the project is used indoors. You can change it to whatever you like by modifying the code.

You can get ultrasonic transducers optimized for 25 kHz, 32 kHz, 40 kHz or wide bandwidth transducers. This project uses a 40 kHz transducer but it will still work with the others if you make simple changes to the software (where it generates the 40kz signal). The receiver and generator circuits will work as they are. If you use a different transducer you must change the software to generate the correct frequency for the transducer as they only work at their specific operating frequency. The 40kz signal is easily generated by the microcontroller but detection requires a sensitive amplifier. I have used a three transistor amplifier for the receiver.

This is followed by a peak detector and comparator which set the sensitivity threshold so that false reflections (weaker signals) are ignored.





****

**Time mode**

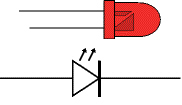
If you store the value of timer 1 and then the counter value is starts counting u to the Rx echo completion or as soon as an ultrasonic echo is received. Counter value gives the time delay in machine cycles. Since the project uses a 4MHz main clock then the time delay will be measured in micro-seconds.

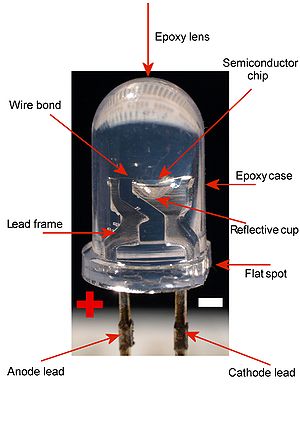
The minimum distance of this scheme is about 5cm. looking at the output of the first receiver amplifier shows that it should be more accurate at lower distances - it is inaccurate by about 2cm which is still quite good. Probably the addition of amplifiers for the longer range stops accurate short range operation. The maximum distance is limited by the sensitivity, gain and noise performance of the receive amplifier and also the transmit power and duration of transmission. For this circuit the maximum distance is about 3m.

#### MC Sonar Specification

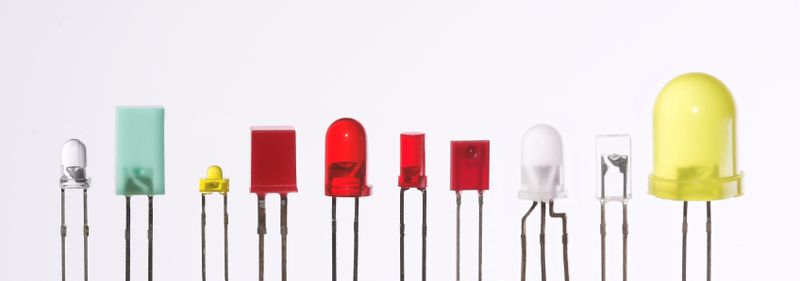
|  |  |
| --- | --- |
| Range | ~0cm - 4m |
| Accuracy | 100 % |
| Transducer frequency | 40kHz |
| internal oscillator frequency | 4MHz |

#### LIGHT-EMITTING DIODE (LED):

The longer lead is the anode (+) and the shorter lead is the cathode (&minus). In the schematic symbol for an LED (bottom), the anode is on the left and the cathode is on the right. Lighemitting diodes are elements for light signalization in electronics.  Description: 118px-LED_symbol



They are manufactured in different shapes, colors and sizes. For their low price, low consumption and simple use, they have almost completely pushed aside other light sources- bulbs at first place.



It is important to know that each diode will be immediately destroyed unless its current is limited. This means that a conductor must be connected in parallel to a diode. In order to correctly determine value of this conductor, it is necessary to know diode’s voltage drop in forward direction, which depends on what material a diode is made of and what colors it is. Values typical for the most frequently used diodes are shown in table below: As seen, there are three main types of LEDs. Standard ones get full brightness at current of 20mA. LowCurrent diodes get full brightness at ten time’s lower current while Super Bright diodes produce more intensive light than Standard ones.

**Buzzer**

**What does it do?**

|  |  |
| --- | --- |
| The buzzer subsystem produces an audible tone when powered. | Description: http://www.electronicsinschools.org/images/data%20sheets%20rev/Buzzer/buz001.gif |

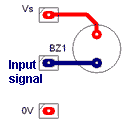
**How does it operate?**

|  |  |
| --- | --- |
| [Description: http://www.electronicsinschools.org/images/data%20sheets%20rev/Buzzer/buz002.gif](http://www.ectinschools.org/lvw/buzzer.lvw) Buzzer circuit  . | Buzzers come in a variety of voltages and currents. The power supply for the buzzer (which can be separate from the supply for the rest of the electronics) must provide the voltage needed by the buzzer.  Piezo sounders are a type of buzzer. They should not be confused with Piezo transducers – which require an a.c. input voltage to drive them.  Some process units provide enough current to drive buzzers. Typical buzzers require currents in the range 10 – 35mA.  If [CMOS ICs](http://www.ectinschools.org/page.php?ps=2&p=894) or a higher current buzzer are used then a driver ([transistor](http://www.ectinschools.org/page.php?ps=2&p=45), [Darlington](http://www.ectinschools.org/page.php?ps=2&p=11) or [MOFET](http://www.ectinschools.org/page.php?ps=2&p=44)) is needed to boost the current. The circuit on the left shows the circuit needed with a driver. |
| [Description: http://www.electronicsinschools.org/images/data%20sheets%20rev/Buzzer/buz003.gif](http://www.ectinschools.org/lvw/buzzer-2.lvw) Buzzer curcuit for use with higher current process units  . | PICs, 555 Timer ICs and the LM324 op-amp can provide higher currents and can drive some buzzers directly.  Check the data for the buzzer and the process unit to make sure that the process unit can provide more current than is needed by the buzzer.  If this is possible, the buzzer is connected to the 0V rail (as on the left) rather than to +Vs. |
|  | Buzzers can either be PCB-mounted or connected to the circuit with flying leads. Usually it is neater to mount them on the PCB. |

**Applications**

* Making a warning sound
* Signaling that something has happened

**Making**

Buzzers have a positive and a negative terminal, marked on their case. The positive terminal should be connected to the positive voltage supply. The negative terminal should be connected to the signal from the driver.

The graphic on the left shows how part of the PCB might look for a PCB-mounted buzzer connected to a driver.

How part of the PCB might look

If a buzzer with flying leads is used then a terminal block is mounted on the PCB and wires from this are connected to the buzzer.

Build and test the unit that will provide the driving input signal before adding the buzzer.

**Testing**

Make sure that the buzzer switches on and off as power is applied from the driver unit.

**Source Code:**

#define MP 9

#define MN 10

int red=11,yel=12,grn=13;

int trigPin = 5;

int echoPin = 6;

int buz = 7;

int duration, distance;

void setup()

{

Serial.begin(9600);

pinMode (trigPin, OUTPUT);

pinMode(echoPin, INPUT);

pinMode(MP, OUTPUT);

pinMode(MN, OUTPUT);

pinMode(red,OUTPUT);

pinMode(yel,OUTPUT);

pinMode(grn,OUTPUT);

pinMode(buz,OUTPUT);

delay(1000);

}

void loop()

{

int l;

digitalWrite(red,1);

delay(3000);

l=density();

digitalWrite(red,0);

digitalWrite(yel,1);

delay(500);

digitalWrite(yel,0);

digitalWrite(grn,1);

if(l<5)

{

digitalWrite(buz,1);

delay(3000);

digitalWrite(buz,0);

digitalWrite(MP,1);

digitalWrite(MN,0);

delay(500);

digitalWrite(MP,0);

digitalWrite(MN,0);

delay(3000);

digitalWrite(MP,0);

digitalWrite(MN,1);

delay(500);

digitalWrite(MP,0);

digitalWrite(MN,0);

delay(5000);

digitalWrite(grn,0);

}

else

{

delay(2000);

digitalWrite(grn,0);

}

delay(1500);

}

int density()

{

digitalWrite (trigPin, HIGH);

delay(50);

digitalWrite (trigPin, LOW);

duration=pulseIn(echoPin,HIGH);

distance=(duration\*0.034)/2;

return distance;

}

**SOFTWARE TOOLS**

**SOFTWARE TOOLS**

**Arduino and Arduino Software and Drivers Installation**

This describes the installation of the Arduino IDE Development software and drivers for the Windows Operating System. The images and description is based on installation under Windows XP, but the process should be similar for Vista and Windows 7.

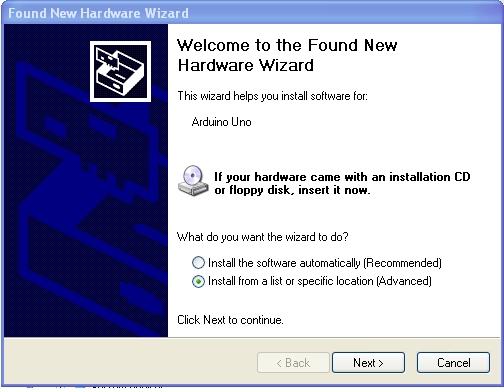
First we need to get the latest version of the Arduino software this can be downloaded from the Arduino website

**STEP 1:**

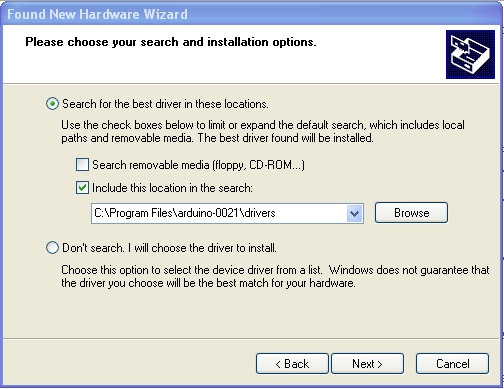
Next, plug in your Arduino board to your computer with a USB cable and wait while Windows detects the new device.

Windows will fail to detect the device as it doesn't know where the drivers are stored. You will get an error similar to the one right.

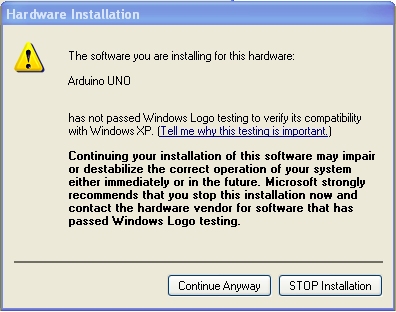
Select the **Install from a list or specific location (Advanced)** option and click Next



**STEP 2:**

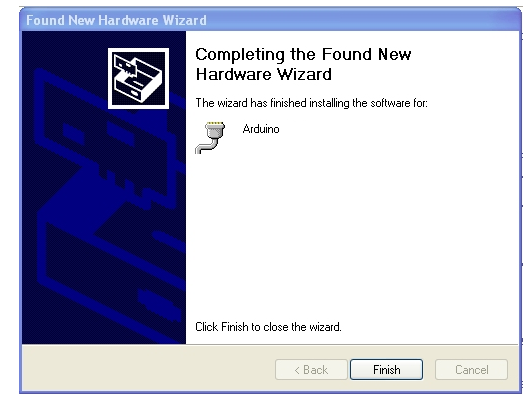
Now choose the location that the Arduino drivers are stored in. This will be in a subfolder called **drivers** in your arduino directory

**STEP3:** After selecting Next you may get a message like the one shown right.   
Select **Continue Anyway**



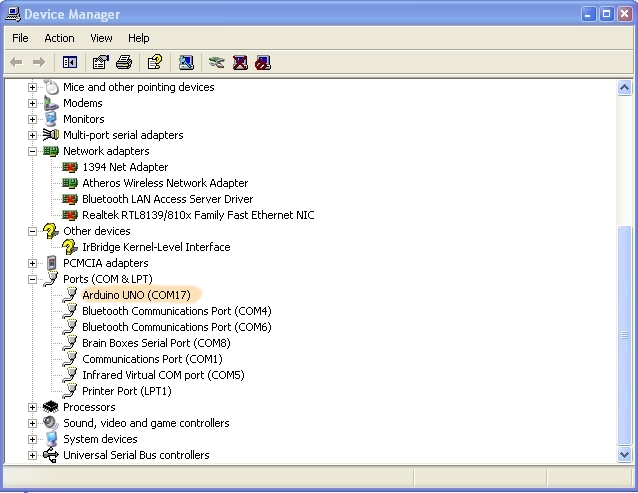
**STEP 4:**

Windows should now have found the Arduino drivers. Click **Finish** to complete the installation

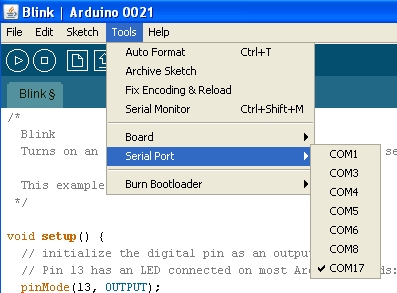


**STEP 5:**

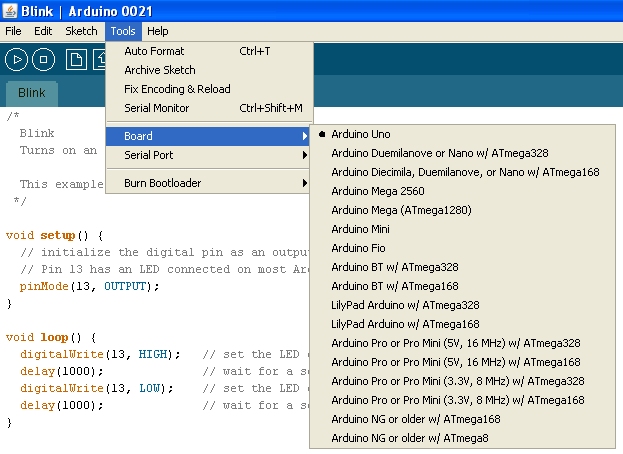
The computer communicates with the Arduino board via a special serial port chip built into the Arduino board. The Arduino IDE software needs to know the serial port number that Windows has just allocated to it Open the Windows **Control Panel** and select the **System** app.  Click on the **Hardware** tab and then on the **Device Manager** button. Click on the **Ports (COM and LPT)** option and note what com port has been allocated to the Arduino Board



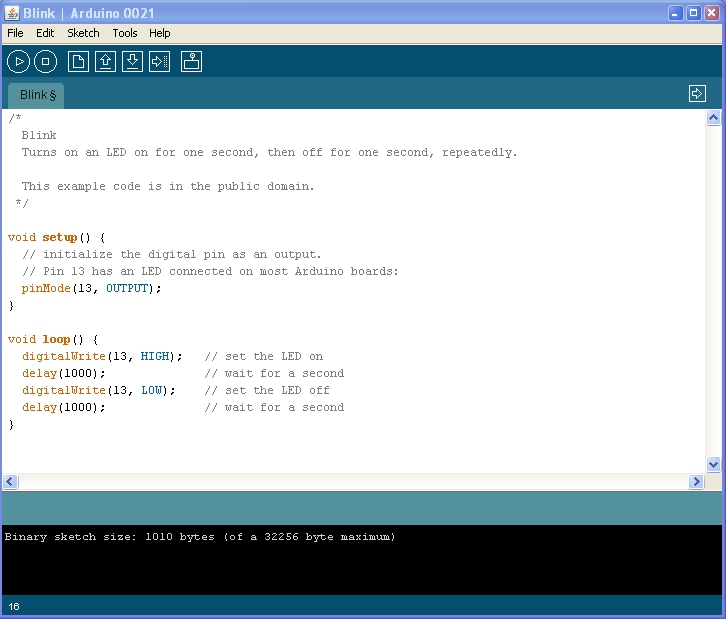
**STEP 6:**

Next, run the Arduino IDE application, which will be in c:\program files\arduino-0021 or similar  
  
Click on Tools | Serial Port and select the port number from above

**STEP 7:**

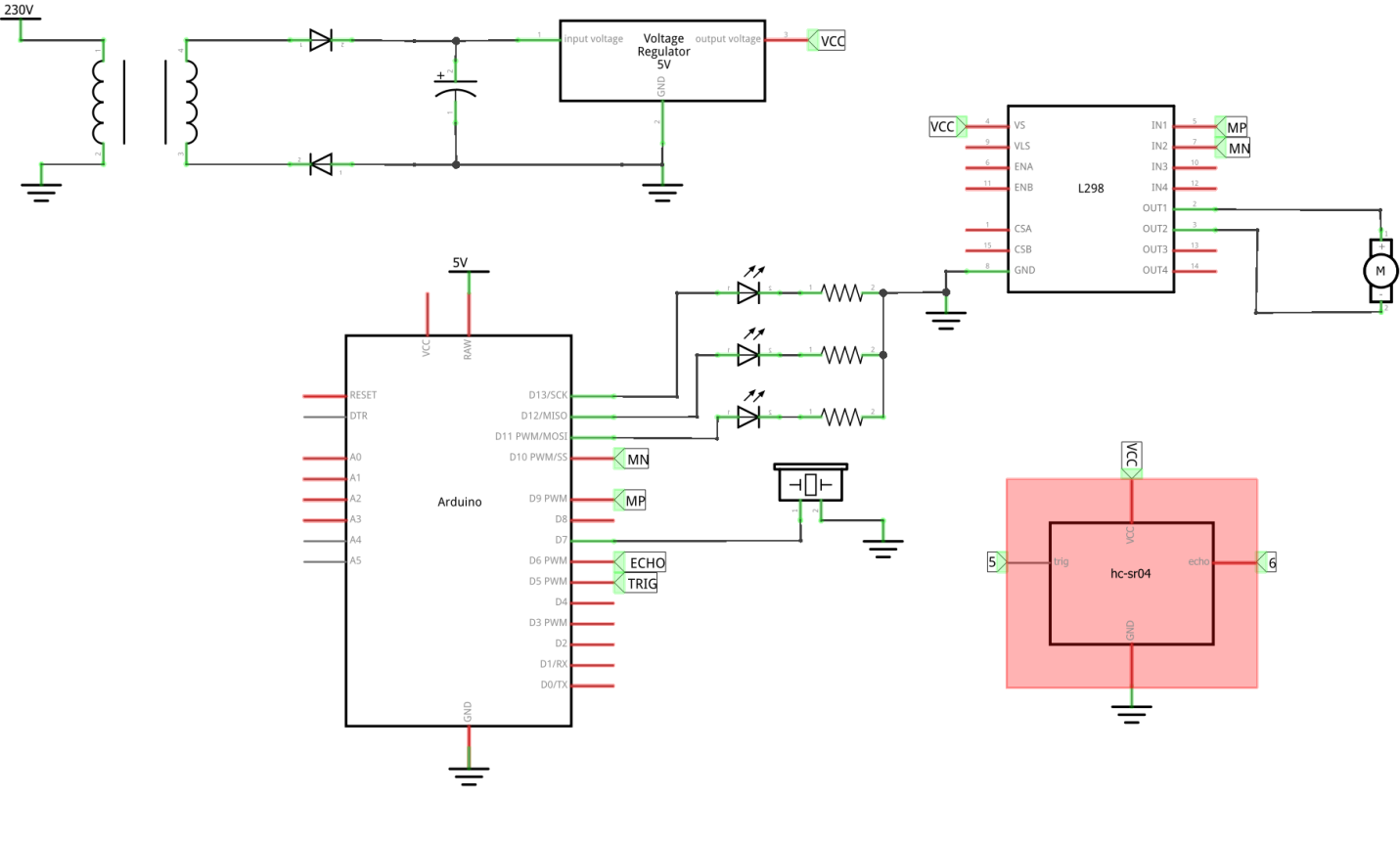
Next click on Tools | Board and select the type of board that you have

**STEP 8:**

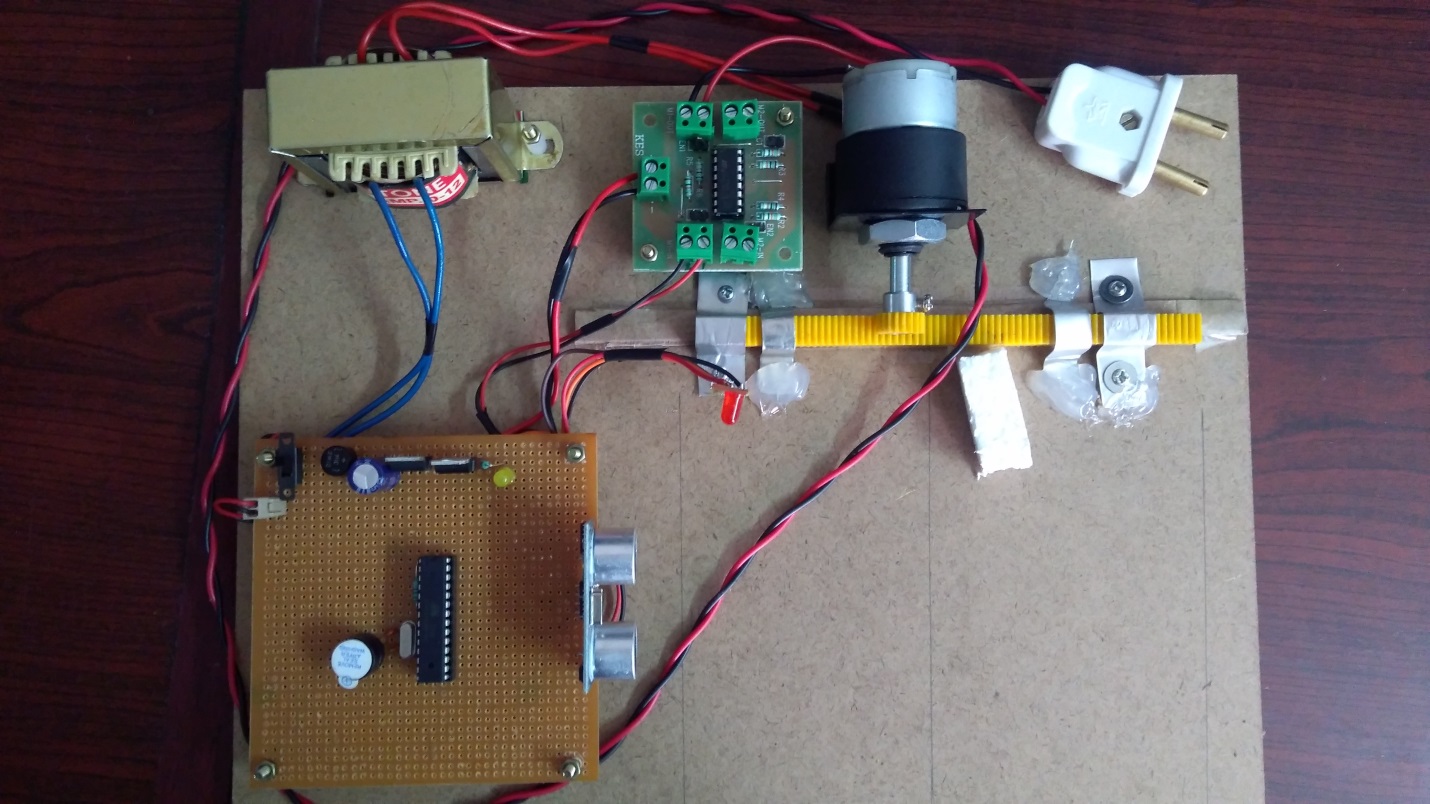
 Now try opening the Simple program from the example directory within the Arduino IDE, Verify/Compile it and upload it to your board. You should see the TX and RX led’s on the board flash showing you that it is working. Finally the built in LED connected to Pin 13 will flash. That’s your first program running.

Create a shortcut to the Arduino IDE and place it on your desktop

**SHEMATIC DIAGRAM:**



**KIT SNAPSHOT:**

****