EXP. No 1. INTRODUCTION TO ARDUINO AND BUZZER

Introduction:

We have often seen that the tanks which are used for industrial and household purposes overflow. The flow into the overhead (typically) tanks has to be cut off manually. Many times, due to ignorance and human errors large amount of freshwater is wasted. Water, being scarce, such heavy losses cannot be afforded. Such losses can be reduced by significant amount, using integrated circuits which control the flow through pumps.
Integrated Circuits can be classified into many parts some of which are Microcontrollers, Microprocessors, etc.

Integrated circuits

In layman terms, an integrated circuit (IC), sometimes called a chip or microchip, is a semiconductor wafer on which thousands or millions of tiny resistors, capacitors, and transistors are fabricated. An IC can function as an amplifier, oscillator, timer, counter, computer memory and much, much more.
Usually the information about a particular IC is written on top of the IC which is essentially the model name.

Microcontroller

(Also called as an 'embedded controller')
Microcontroller is an Integrated Circuit which contains a Processing Unit, Read Only Memory (ROM) which stores the instructions, Read Write Memory (RAM) which is used as temporary storage during execution, Input/output unit to interface with the external world. Most programmable microcontrollers that are used today are built in other products like phones, automobiles and household appliances for computer systems.

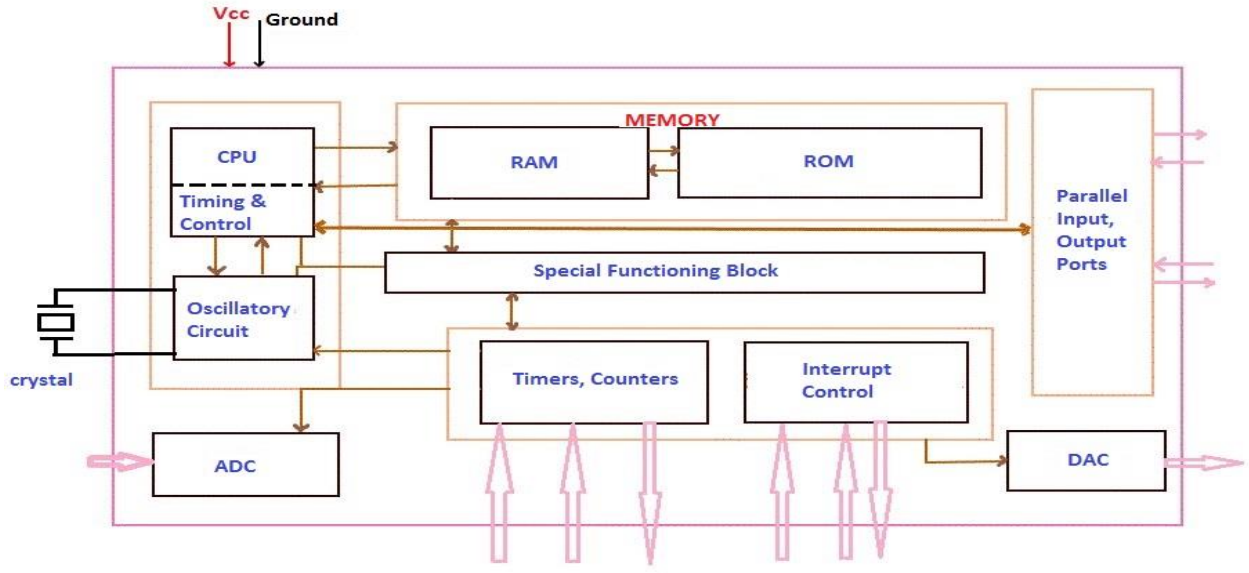


Fig. – Microcontroller Block Diagram

Microcontroller is basically an electronic device which contains one or more of the following things:

1. Central Processing Unit (CPU)
2. Random Access Memory (RAM)
3. Read Only Memory (ROM)
4. Input, Output Ports
5. Timers, Counters
6. Interrupt Controls
7. Analog to Digital convertors (ADC), Digital to Analog Convertors(DAC)
8. Serial Interfacing Ports
9. Oscillatory Circuits

CPU

It is the brain of the microcontroller. It connects every part of the microcontroller to one another forming a single system. Its primary function is to obtain information from the program memory and to decode it.

Memory

The Memory unit is used to store Data and Programs in a microcontroller.

Input, Output Ports

These are used to drive various devices like LCDs, memories, printers to a microcontroller.

Timers, Counters

The Timers, Counters in a microcontroller provide all timing and counting functions in a microcontroller. Their functions include performing clock functions, pulse generation, frequency measurements, etc. They can also be used for counting external pulses. A microcontroller may have one or more than one timer.

Analog to Digital Converter (ADC)

They are used to convert analog signal to digital signals. The input signal to them should be on analog form and the output given out is in digital form

Digital to Analog Converter (DAC)

They perform reverse function of ADC. They convert digital signal to analog signal.

Interrupt Control

The interrupt control used for providing delay for a working program. The delay may be external (i.e. given by an external pin) or internal (given as an instruction during programming).

Thus, microcontroller is basically the logic of a microcomputer put inside an integrated circuit. Thus, if proper input and output devices are connected to a microcontroller it can be converted into a microcomputer.

Microprocessor

A microprocessor is a computer processor which incorporates the functions of a computer's central processing unit on a single integrated circuit. It is a digital integrated

circuit containing both combinational and sequential digital logic. It accepts binary input and gives output after processing it according to instructions stored in its memory. Thus, in layman terms, microprocessor is a chip which performs billions of mathematical calculations per second.

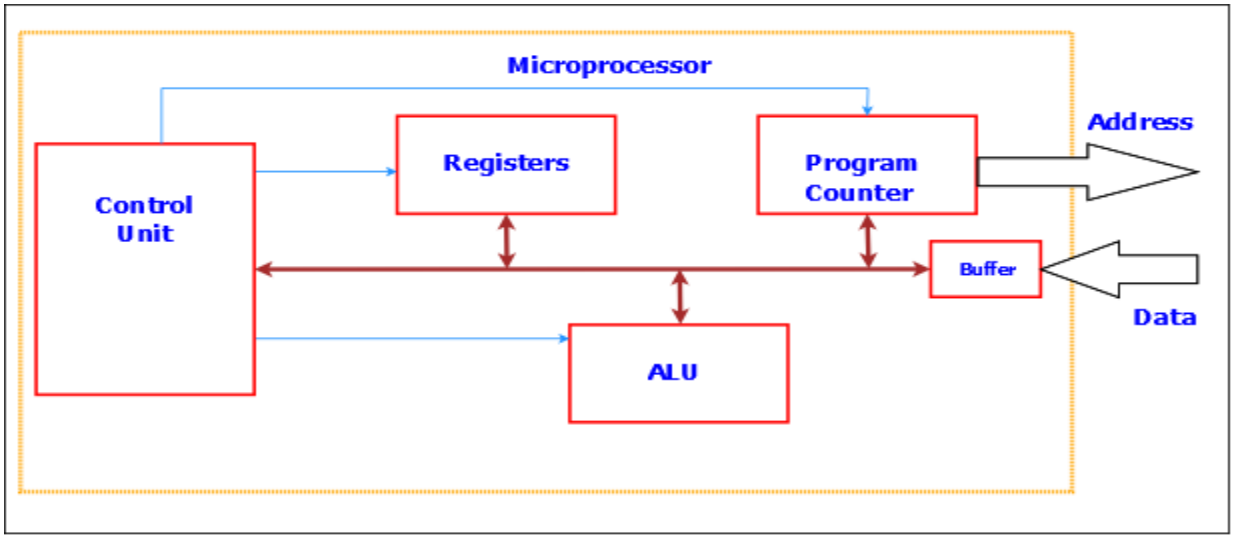


Fig.– Microprocessor Block Diagram

ALU

ALU is a basic functionality unit which consists of arithmetic (addition, subtraction, multiplication, division) and logic (AND, OR, NOT).

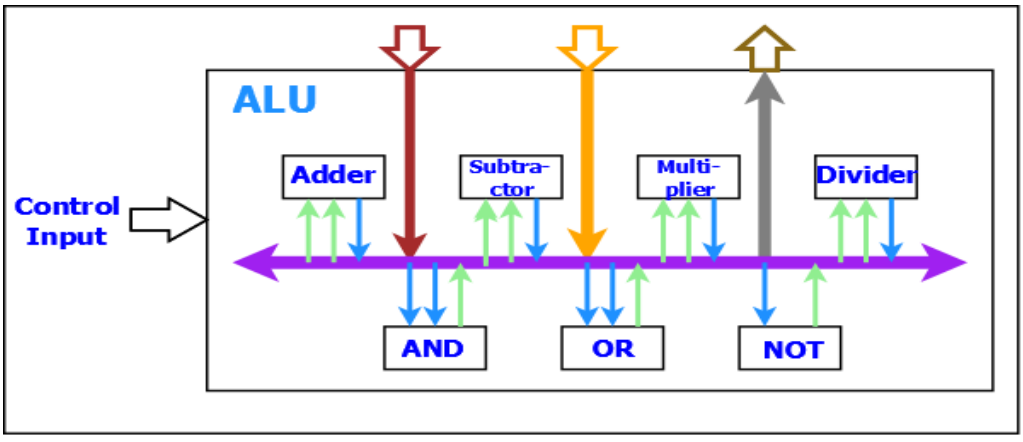


Fig. - ALU Block Diagram

The Control input gives information to the ALU about what operations to perform. Thus, based on the control input only the ALU can perform operations. The control input is given out by the control unit.

Control Unit

The control unit gives out this output based on the operation code. That is, the input given to the Control unit is an operation code and the output given by the unit is the control output. The operation code is a specific code (for every specific operation) stored in the memory by the user.

Memory unit

The memory unit gives out the operation code to the control unit. The memory unit has 2 lines, namely address line and Data line.

ARDUINO:

Arduino is an open-source platform used for building electronics projects. Arduino consists of both a physical programmable circuit board (which is essentially a microcontroller) and a software or IDE (Integrated Development Environment) that runs on a computer. This IDE is used to write an upload code to the physical board.

In completely layman terms, a code is written on a computer, uploaded to the microcontroller on the Arduino board, and then the microcontroller performs some specific tasks based on the code which has been uploaded.

Arduino does not require a separate piece of hardware to load a new code on the board. Instead a simple USB cable can be used

There are various types of Arduino:

1. Arduino Uno
2. Arduino Nano
3. Arduino Due
4. Arduino Mega
5. Arduino Leonardo
6. Lily Pad Arduino Board
7. Red Board Arduino Board

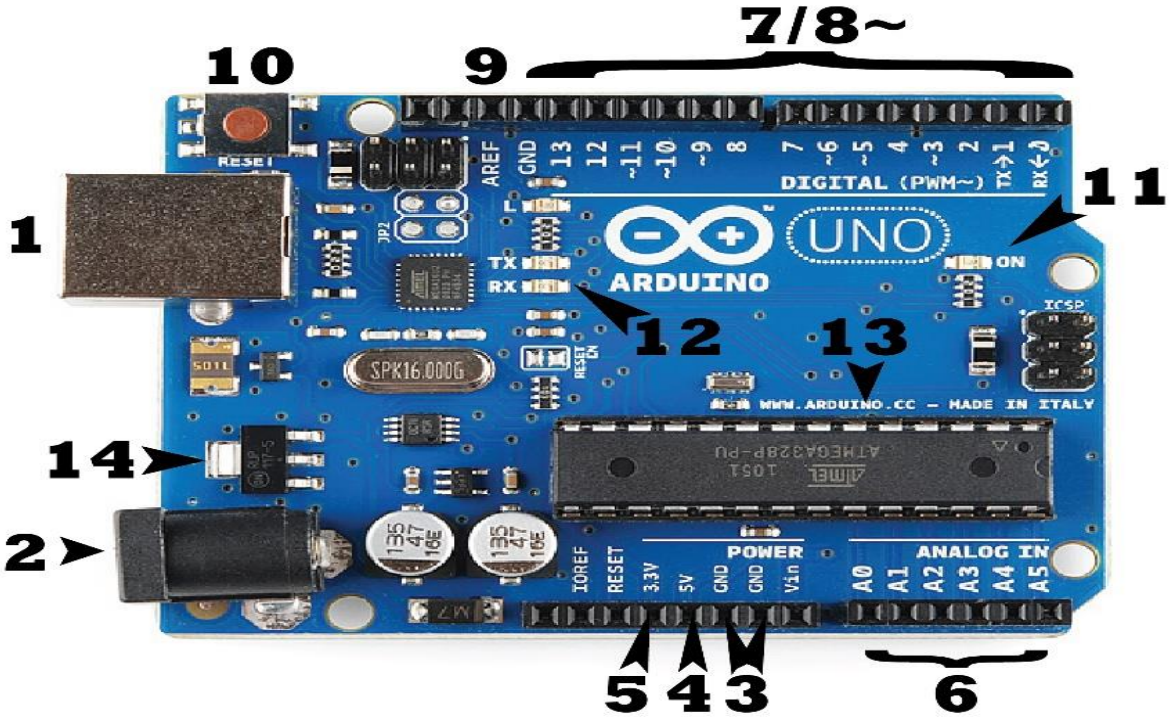


Fig. – ARDUINO UNO Board

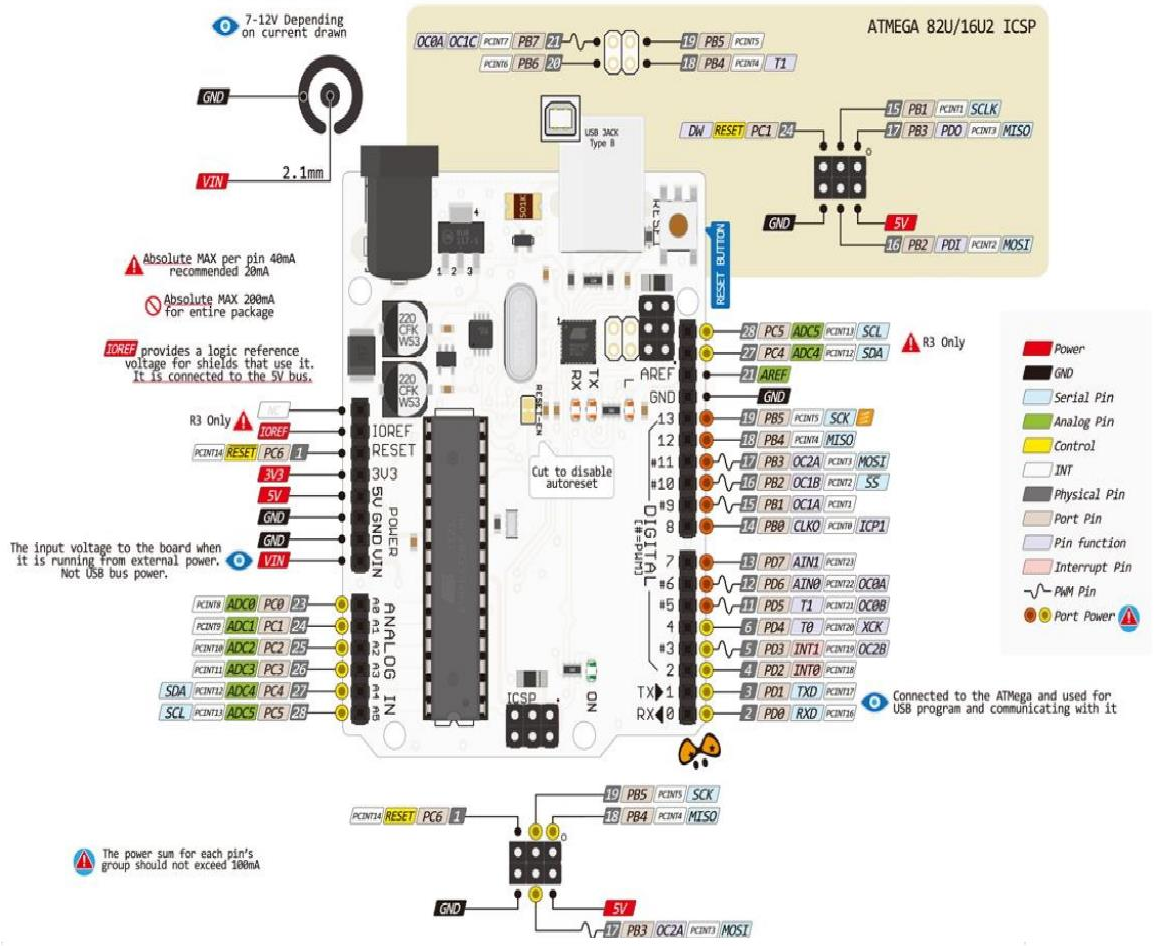


Fig. –Internal ARDUINO UNO Board

1. USB connection
2. Barrel Jack
3. Ground
4. 5V (gives 5V output)
5. 3.3V (gives 3.3 V output)
6. Analog pins
7. Digital Pins
8. Pulse Wave Modulation (PWM)
9. AREF (Analog Reference)
10. Reset Button
11. LED (Power indication)
12. Transmit, Receive LEDs
13. Integrated Circuit
14. Voltage Regulator

Components of the Arduino Uno Board:

Power
The Arduino UNO is powered from a USB cable coming from a computer or a wall power supply terminated in a barrel jack. The code is uploaded to the board through the USB connection.
The working input voltage range of most of the Arduinos is between 6 – 12 Volts.

Pins
Pins are the places where wires are connected to the Arduino board. The Arduino kit usually has wires which have black headers which allows to insert a wire right into the board. The various kinds of pins which are present on the board are given below:

Analog:
These pins are A0 to A5. These pins are essentially Analog to Digital Converters (ADC).

Digital:
0 – 13 pins on the Arduino board are digital pins. These pins can be used for both digital input (telling if a particular signal is given) as well as digital output (powering an LED).

PWM:
The pins 3, 5, 6, 9, 10, and 11 on the UNO board have a tilde (~) next to them. These pins can also be used for something called Pulse Width Modulation.

AREF (Analog Reference):
These pins are used to set external reference voltage as the upper limit for analog input pins.

Reset Button
This button temporarily connects the reset pin to the ground and **restarts** any code loaded on the Arduino.

Power LED Indicator
This LED lights up whenever any power source is connected to the Arduino.

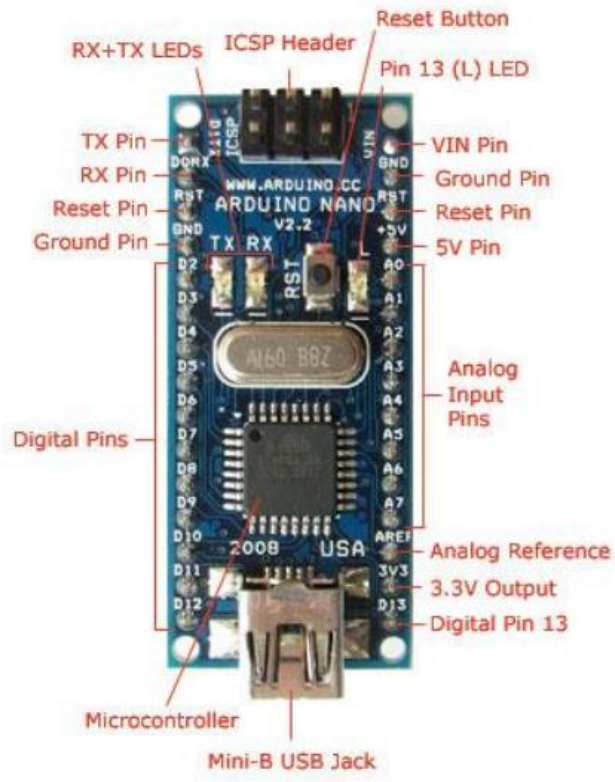
TX, RX LEDs
Tx stands for **transmit**, **Rx** for **Receive**. These LEDs give us an idea whether the Arduino is transmitting and receiving data. In case of UNO, these LEDs are present at two places.

IC
It is the brain of the Arduino. It is essential to know the type of the IC before loading any program on to the Arduino board.

Voltage Regulator
The voltage regulator controls the amount of voltage that is let into the Arduino board. It only lets voltage below a certain value to pass through.

Arduino Nano

Over view The Arduino Nano is a small, complete, and breadboard-friendly board based on the ATmega328 (Arduino Nano 3.0) or ATmega168 (Arduino Nano 2.x). It has more or less the same functionality of the Arduino Duemilanove, but in a different package. It lacks only a DC power jack, and works with a Mini-B USB cable instead of a standard one. The Nano was designed and is being produced by Gravitech.



Specifications :

Microcontroller	Atmel ATmega168 or ATmega328
Operating Voltage (logic level)	5 V
Input Voltage (recommended)	7-12 V
Input Voltage (limits)	6-20 V
Digital I/O Pins	14 (of which 6 provide PWM output)
Analog Input Pins	8
DC Current per I/O Pin	40 mA
Flash Memory	16 KB (ATmega168) or 32 KB (ATmega328) of which 2 KB used by bootloader
SRAM 1	KB (ATmega168) or 2 KB (ATmega328)
EEPROM	512 bytes (ATmega168) or 1 KB (ATmega328)
Clock Speed	16 MHz Dimensions 0.73" x 1.70"

Power : The Arduino Nano can be powered via the Mini-B USB connection, 6-20V unregulated external power supply (pin 30), or 5V regulated external power supply (pin 27). The power source is automatically selected to the highest voltage source.

The FTDI FT232RL chip on the Nano is only powered if the board is being powered over USB. As a result, when running on external (non-USB) power, the 3.3V output (which is supplied by the FTDI chip) is not available and the RX and TX LEDs will flicker if digital pins 0 or 1 are high.

Memory The ATmega168 has 16 KB of flash memory for storing code (of which 2 KB is used for the bootloader); the ATmega328 has 32 KB, (also with 2 KB used for the bootloader). The ATmega168 has 1 KB of SRAM and 512 bytes of EEPROM (which can be read and written with the EEPROM library); the ATmega328 has 2 KB of SRAM and 1 KB of EEPROM.

Input and Output

Each of the 14 digital pins on the Nano can be used as an input or output, using pinMode(), digitalWrite(), and digitalRead() functions. They operate at 5 volts. Each pin can provide or receive a maximum of 40 mA and has an internal pull-up resistor (disconnected by default) of 20K-50K Ohms. In addition, some pins have specialized functions:

Serial: 0 (RX) and 1 (TX). Used to receive (RX) and transmit (TX) TTL serial data. These pins are connected to the corresponding pins of the FTDI USB-to-TTL Serial chip.

External Interrupts: 2 and 3. These pins can be configured to trigger an interrupt on a low value, a rising or falling edge, or a change in value. See the attachInterrupt() function for details.

PWM: 3, 5, 6, 9, 10, and 11. Provide 8-bit PWM output with the analogWrite() function

.

SPI: 10 (SS), 11 (MOSI), 12 (MISO), 13 (SCK). These pins support SPI communication, which, although provided by the underlying hardware, is not currently included in the Arduino language.

LED: 13. There is a built-in LED connected to digital pin 13. When the pin is HIGH value, the LED is on, when the pin is LOW, it's off.

The Nano has 8 analog inputs, each of which provide 10 bits of resolution (i.e. 1024 different values). By default they measure from ground to 5 volts, though is it possible to change the upper end of their range using the **analogReference()** function. Additionally, some pins have specialized functionality:

I 2C: 4 (SDA) and 5 (SCL). Support I2C (TWI) communication using the Wire library (documentation on the Wiring website).

There are a couple of other pins on the board:

AREF. Reference voltage for the analog inputs. Used with analogReference().

Reset. Bring this line LOW to reset the microcontroller. Typically used to add a reset button to shields which block the one on the board. See also the mapping between Arduino pins and ATmega168 ports.

Communication The Arduino Nano has a number of facilities for communicating with a computer, another Arduino, or other microcontrollers. The ATmega168 and ATmega328 provide UART TTL (5V) serial communication, which is available on digital pins 0 (RX) and 1 (TX). An FTDI FT232RL on the board channels this serial communication over USB and the FTDI drivers (included with the Arduino software) provide a virtual com port to software on the computer. 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 FTDI chip and USB connection to the computer (but not for serial communication on pins 0 and 1).

A Software Serial library allows for serial communication on any of the Nano's digital pins. The ATmega168 and ATmega328 also support I2C (TWI) and SPI communication. The Arduino software includes a Wire library to simplify use of the I2C bus; see the documentation for details. To use the SPI communication, please see the ATmega168 or ATmega328 datasheet.

Programm ing

The Arduino Nano can be programmed with the Arduino software (download). Select "Arduino Diecimila, Duemilanove, or Nano w/ ATmega168" or "Arduino Duemilanove or Nano w/ ATmega328" from the Tools
> **Board** menu (according to the microcontroller on your board). For details, see the reference and tutorials.

The ATmega168 or ATmega328 on the Arduino Nano comes pre burned with a bootloader that allows you to upload new code to it without the use of an external hardware programmer. It communicates using the original STK500 protocol (reference, C header files).

You can also bypass the bootloader and program the microcontroller through the ICSP (In-Circuit Serial Programming) header; see these instructions for details.

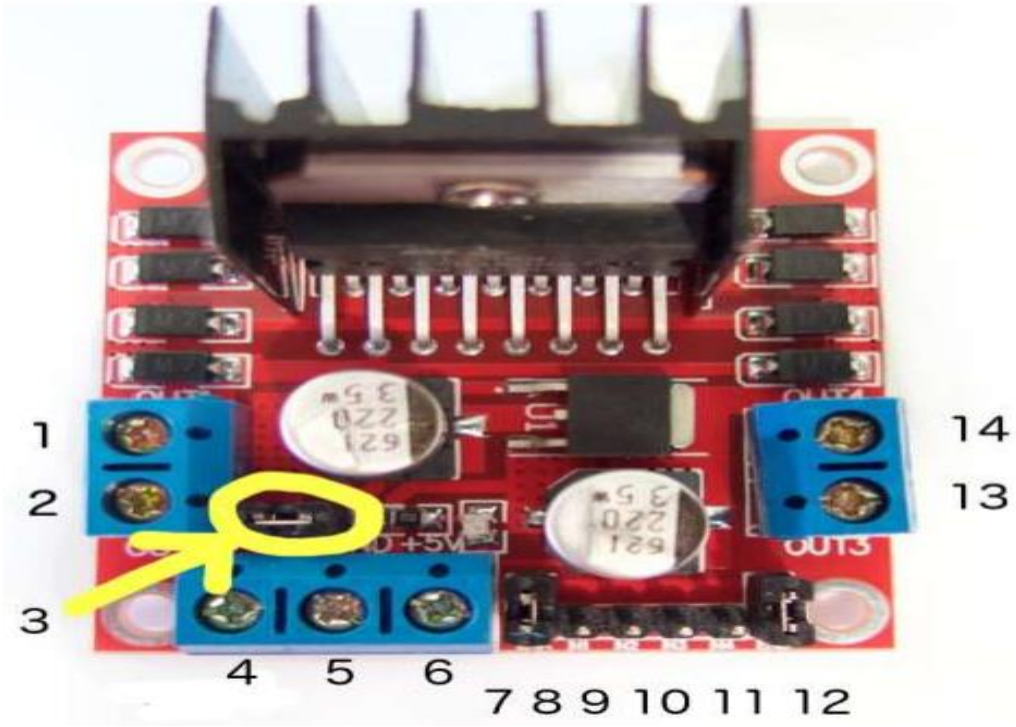
Automat ic (Software) Reset

Rather then requiring a physical press of the reset button before an upload, the Arduino Nano 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 FT232RL is connected to the reset line of the ATmega168 or ATmega328 via a 100-nanofarad capacitor. When this line is asserted (taken low), the reset line drops long enough to reset the chip. The Arduino software uses this capability to allow you to upload code by simply pressing the upload button in the Arduino environment. This means that the bootloader can have a shorter timeout, as the lowering of DTR can be well coordinated with the start of the upload.

This setup has other implications. When the Nano is connected to either a computer running Mac OS X or Linux, it resets each time a connection is made to it from software (via USB). For the following half-second or so, the bootloader is running on the Nano. 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.

L298N Dual Motor Controller

L298N H-bridge is a Dual Motor Controller Module 2A with Arduino. This allows you to control the speed and direction of two DC motors, or control one bipolar stepper motor with ease. The L298N H-bridge module can be used with motors that have a voltage of between 5 and 35V DC.
There is also an onboard 5V regulator, so if your supply voltage is up to 12V you can also source 5V from the board.



1. DC motor 1 "+" or stepper motor A+
2. DC motor 1 "-" or stepper motor A
3. 12V jumper - remove this if using a supply voltage greater than 12V DC. This enables power to the onboard 5V regulator
4. Connect your motor supply voltage here, maximum of 35V DC. Remove 12V jumper if >12V DC
5. GND
6. 5V output if 12V jumper in place, ideal for powering your Arduino (etc)
7. DC motor 1 enable jumper. Leave this in place when using a stepper motor. Connect to PWM output for DC motor speed control.
8. IN1
9. IN2
10. IN3
11. IN4
12. DC motor 2 enable jumper. Leave this in place when using a stepper motor. Connect to PWM output for DC motor speed control.
13. DC motor 2 "+" or stepper motor B+ 14. DC motor 2 "-" or stepper motor B

Controlling DC Motors

To control one or two DC motors is quite easy. First, connect each motor to the A and B connections on the L298N module. If you're using two motors for a robot (etc) ensure that the polarity of the motors is the same on both inputs. Otherwise, you may need to swap them over when you set both motors to forward and one goes backwards!

Next, connect your power supply - the positive to pin 4 on the module and negative/GND to pin 5. If you supply is up to 12V you can leave in the 12V jumper (point 3 in the image above) and 5V will be available from pin 6 on the module. This can be fed to your Arduino's 5V pin to power it from the motors' power supply. Don't forget to connect Arduino GND to pin 5 on the module as well to complete the circuit.
Now you will need six digital output pins on your Arduino, two of which need to be PWM (pulse-width modulation) pins. PWM pins are denoted by the tilde ("~") next to the pin number, for example:

Finally, connect the Arduino digital output pins to the driver module.The motor direction is controlled by sending a HIGH or LOW signal to the drive for each motor (or channel) and a LOW and HIGH will cause it to turn in the other direction.
However the motors will not turn until a HIGH is set to the enable pin (7 for motor one, 12 for motor two). And they can be turned off with a LOW to the same pin(s). However if you need to control the speed of the motors, the PWM signal from the digital pin connected to the enable pin can take care of it.

Ultrasonic Distance Sensor

Ultrasonic sensor is a device that measures the distance to an object, boundary or layer with the help of high frequency sound waves.

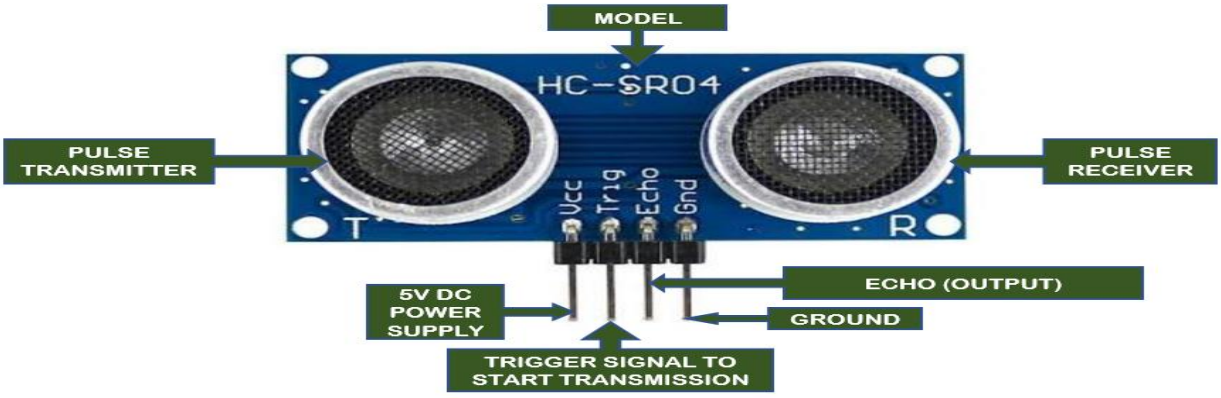


Fig. - Ultrasonic Distance Sensor Diagram

How Does It Work?

Ultrasonic ranging module HC - SR04 provides 2cm - 400cm non-contact measurement function, the ranging accuracy can reach to 3mm. The modules includes ultrasonic transmitters, receiver and control circuit. The basic principle of work:
(1) Using IO trigger for at least 10us high level signal,
(2) The Module automatically sends eight 40 kHz and detect whether there is a pulse signal back. (3) IF the signal back, through high level , time of high output IO duration is the time from sending ultrasonic to returning.

$$Test\ signal = \frac{(\text{high level time} \times \text{velocity of sound (340M/S)}}{2}$$

Wire connecting direct as following:

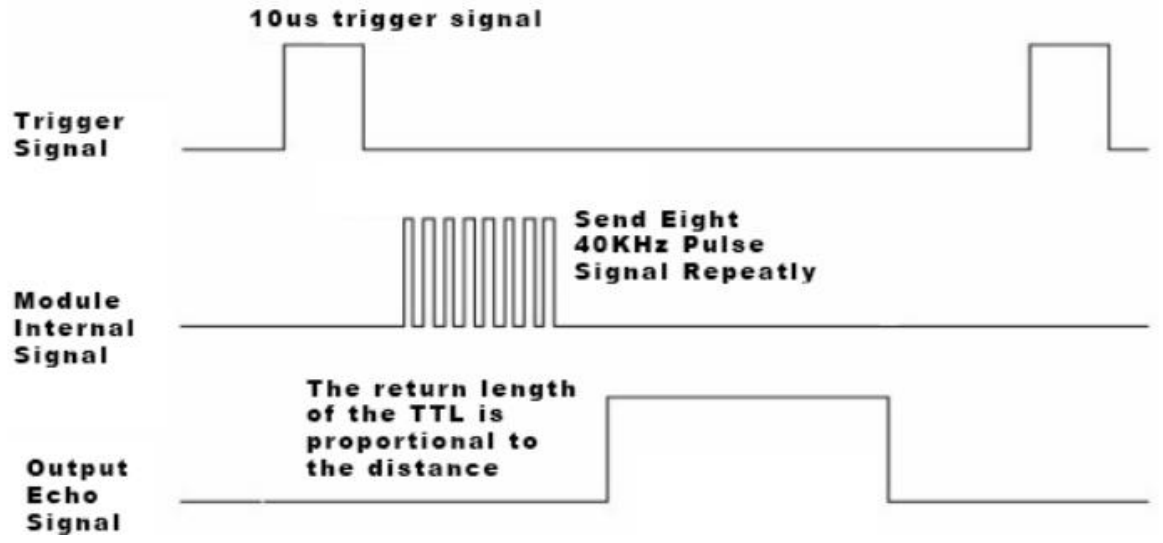
- 5V Supply
- Trigger Pulse Input
- Echo Pulse Output
- 0V Ground

Electric Parameter

Working Voltage	DC 5 V
Working Current	15mA
Working Frequency	40Hz
Max Range	4m
Min Range	2cm
MeasuringAngle	15 degree
Trigger Input Signal	10uS TTL pulse
Echo Output Signal	Input TTL lever signal and the range in proportion
Dimension	45*20*15mm

Timing diagram

The Timing diagram is shown below. You only need to supply a short 10uS pulse to the trigger input to start the ranging, and then the module will send out an 8 cycle burst of ultrasound at 40 kHz and raise its echo. The Echo is a distance object that is pulse width and the range in proportion .You can calculate the range through the time interval between sending trigger signal and receiving echo signal. Formula: uS / 58 = centimeters or uS / 148 =inch; or: the range = high level time * velocity (340M/S) / 2; we suggest to use over 60ms measurement cycle, in order to prevent trigger signal to the echo signal.



Why/ When To Use Ultrasonic Sensors?

1) Gives accurate and automatic distance measurement in normal and difficult environments.
2) Particularly suitable for environments where optical sensors are unusable such as smoke, dust and similar.
3) Usable for large ranges.

Attention:

- The module is not suggested to connect directly to electric, if connected electric, the GND terminal should be connected the module first, otherwise, it will affect the normal work of the module.

- When tested objects, the range of area is not less than 0.5 square meters and the plane requests as smooth as possible, otherwise ,it will affect the results of measuring.

What Parameters Does It Detect?

- Distance
- Level
- Diameter
- Presence
- Position
- Motion

Material, surface, light, dust, mist and vapour doesn't affect the functioning of ultrasonic sensor.

Where Is This Device And Concept Used?

1) SONAR (sound navigation and ranging) uses this principle to detect depth of sea surface and its form. It also detects deep trenches.
2) Parking sensors in cars.
3) Animals like bats use echolocation as a substitute of their weak vision.

Relay Module

The relay module is a separate hardware device used for switching on/off a remote device. Medium of operation may be a network or internet. It can control computers, peripherals or other powered devices.

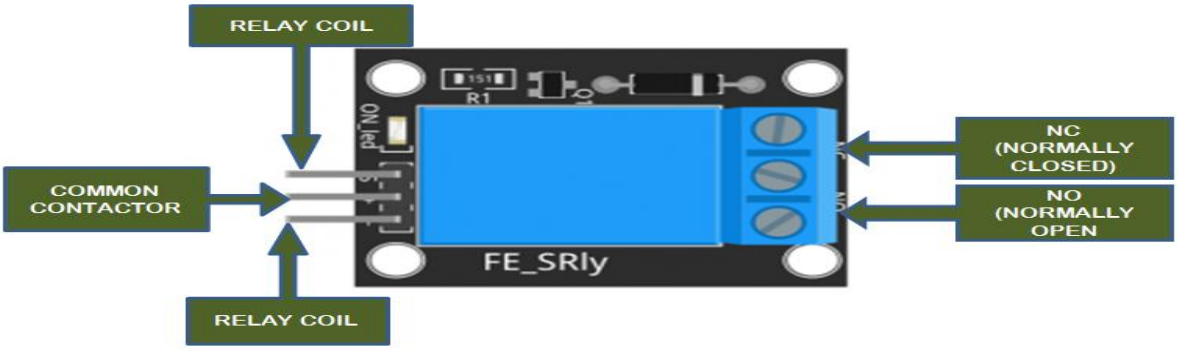


Fig. – Relay Module Diagram

Working Of Relay Module:

These active semiconductor devices use light instead of magnetism to actuate a switch. The light comes from an LED, or light emitting diode. When control power is applied to the device's output, the light General-Purpose Relay is turned on and shines across an open space. On the load side of this space, a part of the device senses the presence of the light, and triggers a solid-state switch that either opens or closes the circuit under control.

Why Solid-State Relay Module?

Often, solid state relays are used where the circuit under control must be protected from the introduction of electrical noises. Advantages of Solid State Relays include low EMI/RFI, long life, no moving parts, no contact bounce, and fast response.

What Affects Relay Module's working?

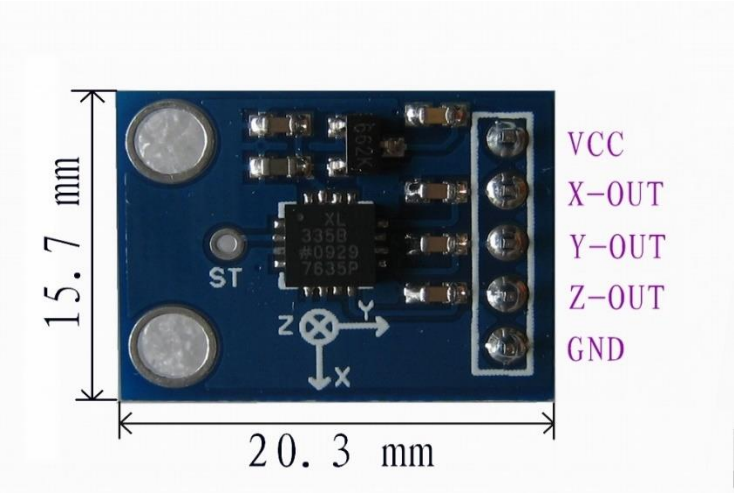
The contacts are the most important constituent of a relay. Their characteristics are significantly affected by factors such as the material of the contacts, voltage and current values applied to them (especially, the voltage and current waveforms when energizing and de-energizing the contacts), the type of load, operating frequency, and bounce. If any of these factors fail to satisfy a predetermined value, problems such as metal degradation between contacts, contact welding, wear, or a rapid increase in the contact resistance may occur. The quantity of electrical current that flows through the contacts directly influences the contacts' characteristics. To prolong its life, a contact safety circuit is recommended

GY-61 ADXL335 3-Axis Accelerometer Module

GY-61 DXL335 3-Axis Accelerometer Module is a three-axis accelerometer sensor module based on ADXL335 integrated circuit. The ADXL335 is a triple axis accelerometer with extremely low noise and power consumption. The sensor has a full sensing range of +/-3g. It can measure the static acceleration of gravity in tilt-sensing applications, as well as dynamic acceleration resulting from motion, shock, or vibration.

There is an on-board 3.3V voltage regulator to power the ADXL335 so power provided should be between 3.3V and 6V DC.

Model: GY-61 Three-axis magnetic field accelerometer module Compact size, low power supply Used for game systems, mobile devices, etc



General Specifications

- ADXL335 3-axis Accelerometer
- On-board 3.3V Voltage Regulator
- Analog voltage output centered at 1.65V
- Suitable for connection to 5V and 3.3V systems

Technical Specifications

- Sensor Chip : ADXL335
- Operating Voltage Range : 3V ~ 5V
- Supply Current : 400uA
- Interface : Analog quantity output
- Full scale range : +/-3g
- Operating Temperature : -40'C~ +85'C
- Sensitivity : 300mv /g;
- Sensitivity of accuracy (%) : +/- 10
- Application : Various electronic products or DIY project
- Material : PCB + Brass
- Dimensions : 21 x 16 x 10 mm / 0.83 x 0.63 x 0.39 inch
- Weight : 2 g / 0.07 oz
- Color : Blue

Pin Definitions:

1. VCC: 3.3V or 5V

2. X_OUT: Analog Output

3. Y_OUT: Analog Output

4. Z_OUT: Analog Output

5. GND: Ground

Light Dependent Resistor LDR

LDR Resistor | LDR Pinout

The cadmium sulfide (CdS) or light dependent resistor (LDR) whose resistance is inversely dependent on the amount of light falling on it, is known by many names including the photo resistor, photoresistor, photoconductor, photoconductive cell, or simply the photocell.

A typical structure for a photoresistor uses an active semiconductor layer that is deposited on an insulating substrate. The semiconductor is normally lightly doped to enable it to have the required level of conductivity. Contacts are then placed either side of the exposed area.

the photoresistors are available in 5mm, 12mm and 20mm sizes, the conformably epoxy or hermetical package offer high quality performance for applications that require quick response and good characteristic of spectrum.

Features –

Quick Response

Reliable Performance

- Epoxy or hermetical package

- Good Characteristic of Spectrum

Applications –

Photo switch

Photoelectric Control

Auto Flash for Camera

Electronic Toys,

Industrial Control

Light Resistance : Measured at 10 lux with standard light A (2854K-color temperature) and 2hr. preillumination at 400-600 lux prior testing.

Dark Resistance : Measured at 10th seconds after closing 10 lux.

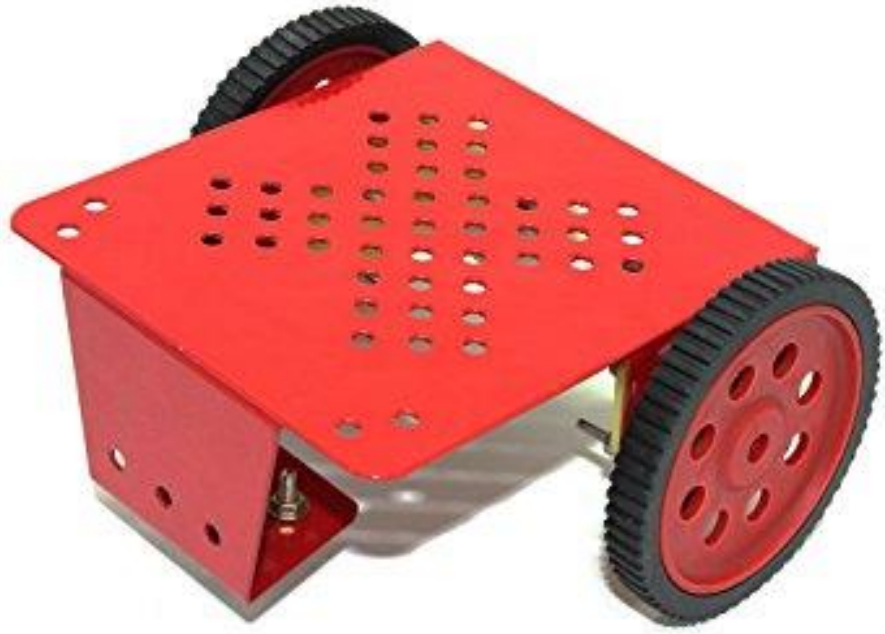
Gamma characteristic : Under 10 lux and 100 lux and given by $\gamma = \log(R10/R100) / \log(100/10) = \log(R10/R100)$ R10, R100: resistance at 10 lux and 100 lux. The tolerance of γ is ±0.1.

DC Motors:

A DC motor is a rotatory engine that converts electrical energy into mechanical energy. They have two leads one positive and the other negative. Connecting these two leads to the battery the motor starts rotating, on switching the leads the motor rotates in opposite direction.

Gear Motor

Chasis and Wheels:



Ball Caster:

Breadboard

A breadboard is a construction base for prototyping of electronic circuits. Because the solderless breadboard does not require soldering, it is reusable, which makes it easy to use for creating temporary prototypes and experimenting with circuit design.

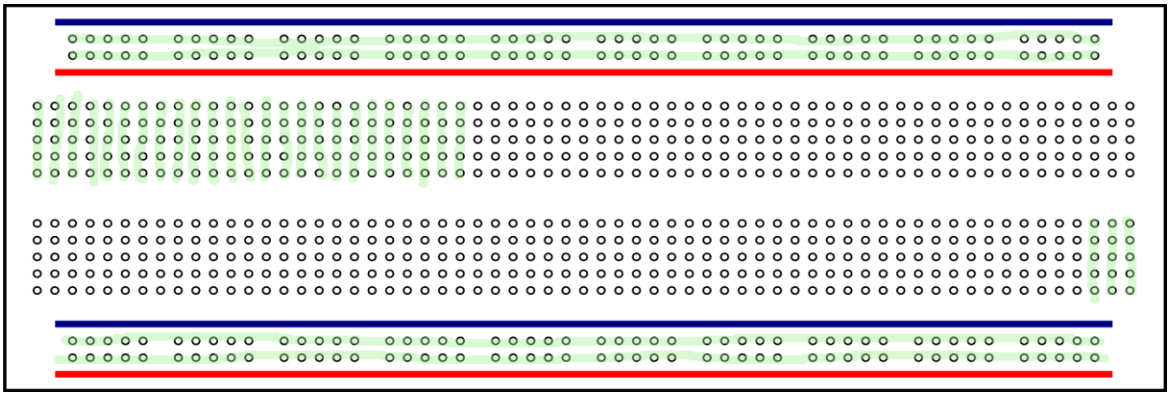


Fig.1: Schematic of the Bread Board

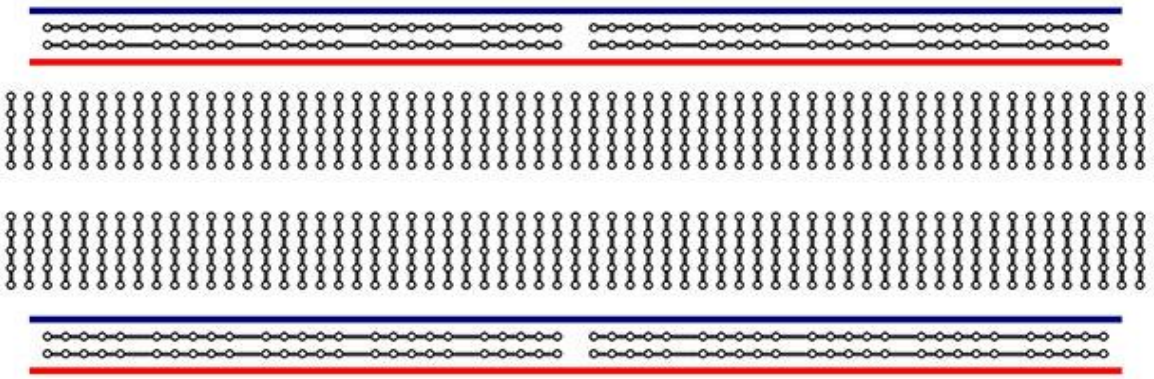


Fig.2: Equipotential Lines

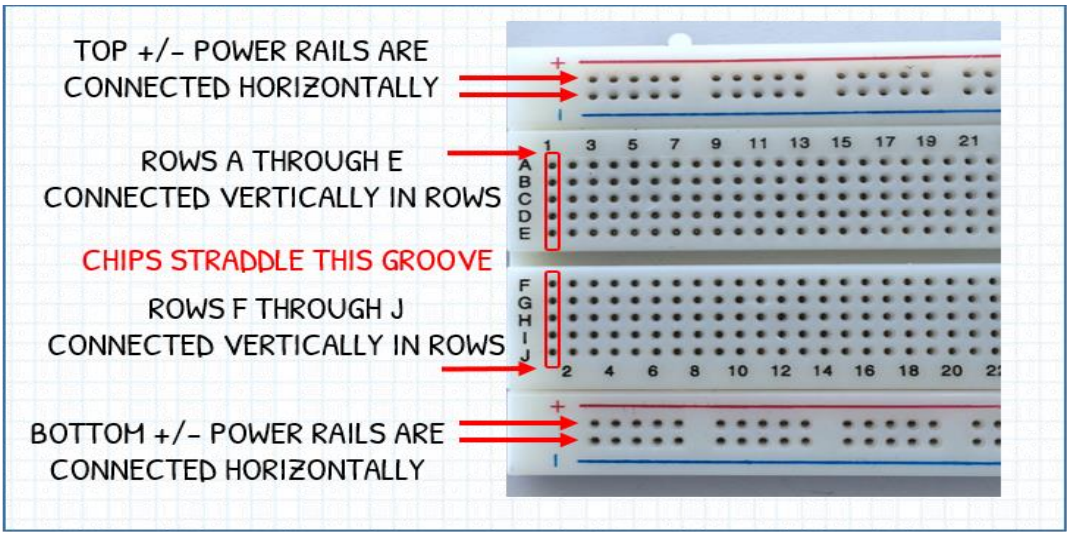
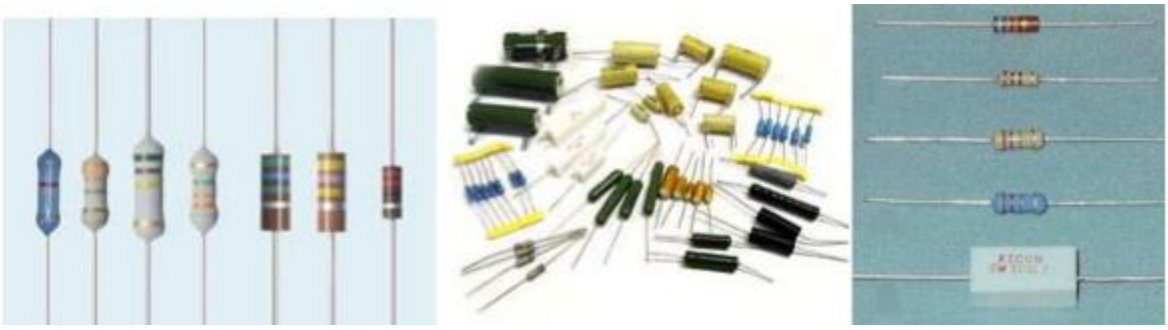


Fig.3: How to use breadboard

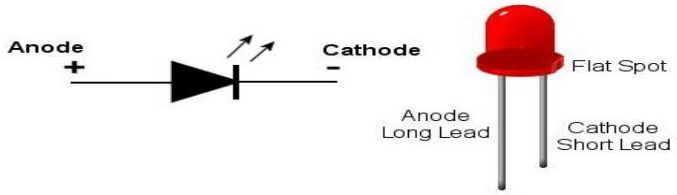
RESISTORS

A resistor is a passive two-terminal electrical component that implements electrical resistance as a circuit element. The current through a resistor is in direct proportion to the voltage across the resistor's terminals. This relationship is represented by Ohm's law. A device used in electrical circuits to maintain a constant relation between current flow and voltage. Resistors are used to step up or lower the voltage at different points in a circuit and to transform a current signal into a voltage signal or vice versa, among other uses. The electrical behavior of a resistor obeys Ohm's law for a constant resistance; however, some resistors are sensitive to heat, light, or other variables. Resistors are one of the most used components in a circuit. Most are color coded, but some have their value in Ohms and their tolerance printed on them. A multi-meter that can check resistance can also be helpful, providing the resistor is already removed from the board (measuring it while still soldered in can give inaccurate results, due to connections with the rest of the circuit). They are typically marked with an “R” on a circuit board.



LED

Another common type of diode is a light-emitting diode, or LED. Light emitting diodes generate visible light when current passes between the anode and cathode though a space called the p-n junction. The electrical charge in this space produces light in different colors depending on the charge and materials used in the diode.



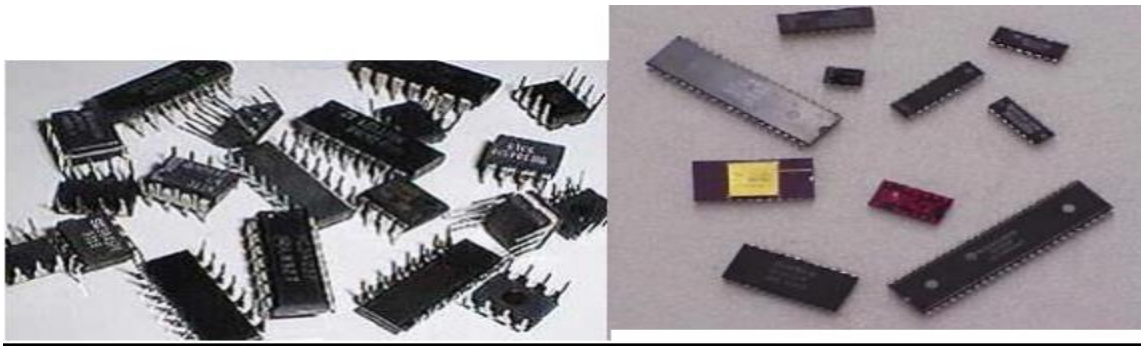
BATTERIES

In electricity, a battery is a device consisting of one or more electrochemical cells that convert stored chemical energy into electrical energy. Batteries are also pretty easy to identify, and are well marked with their specification.

Fig.: Different types of Practical Batteries

INTEGRATED CIRCUITS

An integrated circuit or monolithic integrated circuit (also referred to as an IC, a chip, or a microchip) is a set of electronic circuits on one small plate ("chip") of semiconductor material, normally silicon. This can be made much smaller than a discrete circuit made from independent components. Integrated circuits are used in virtually all electronic equipment today and have revolutionized the world of electronics. Computers, mobile phones, and other digital home appliances are now inextricable parts of the structure of modern societies, made possible by the low cost of producing integrated circuits. Integrated Circuits (typically marked with an “U” or “IC” on a circuit board)

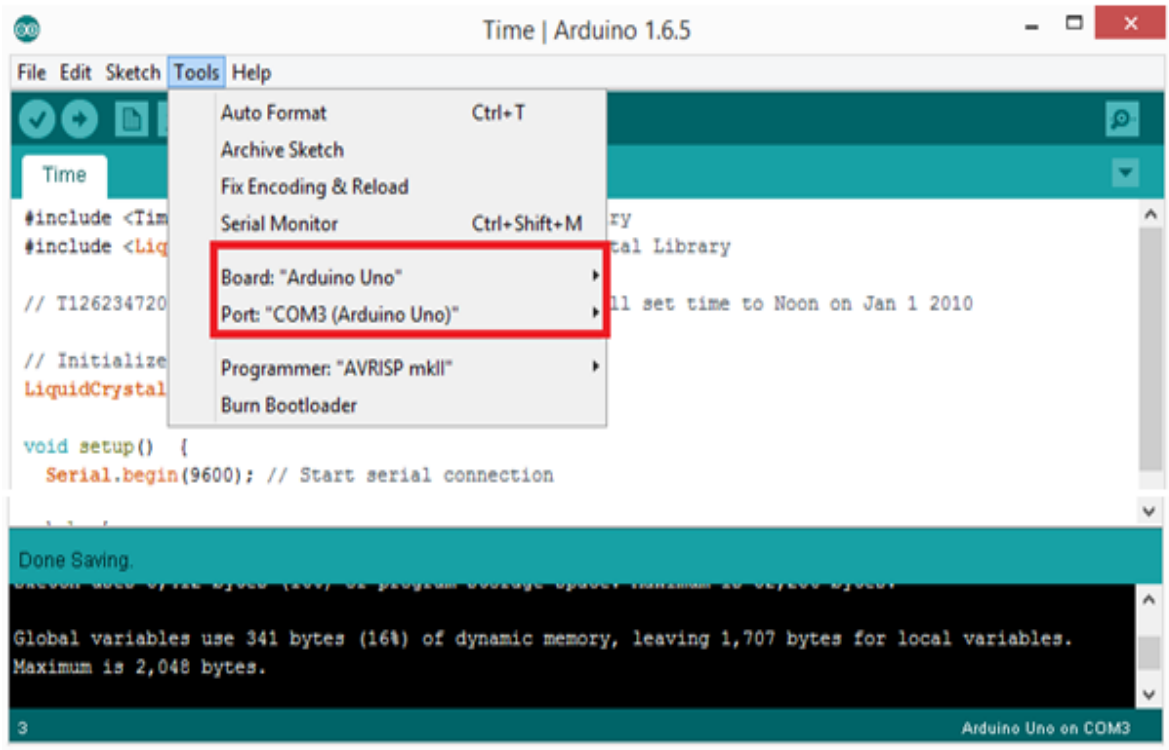


Introduction to “Arduino” software

The Arduino Web Editor is one of the Arduino Create platform's tools. The open-source Arduino Software (IDE) makes it easy to write code and upload it to the board. It runs on Windows, Mac OS X, and Linux. The environment is written in Java and based on Processing and other open-source software.

Initial steps before starting any project:-

1. Connect the Arduino to the USB port of your laptop/system,
2. Open the Arduino software.
3. Go to Tools=>select Board: “Arduino Uno”=>Port: COMx (where x is predefined by the laptop/system)



Language Reference

Arduino programming language can be divided in three main parts: structure, values (variables and constants), and functions.

FUNCTIONS

For controlling the Arduino board and performing computations.

Digital I/O digitalRead() digitalWrite() pinMode() Analog I/O analogRead() analogReference() analogWrite() Zero, Due & MKR Family analogReadResolution() analogWriteResolution() Advanced I/O noTone() pulseIn() pulseInLong() shiftIn() shiftOut() tone()	Time delay() delayMicroseconds() micros() millis() Math abs() constrain() map() max() min() pow() sq() sqrt() Trigonometry cos() sin() tan()	Characters isAlpha() isAlphaNumeric() isAscii() isControl() isDigit() isGraph() isHexadecimalDigit() isLowerCase() isPrintable() isPunct() isSpace() isUpperCase() isWhitespace() Random Numbers random() randomSeed()	Bits and Bytes bit() bitClear() bitRead() bitSet() bitWrite() highByte() lowByte() Interrupts interrupts() noInterrupts() Communication Serial stream USB Keyboard Mouse

VARIABLES

Arduino data types and constants.

Constants	Data Types	Variable Scope & Qualifiers
Floating Point Constants Integer Constants HIGH | LOW INPUT | OUTPUT | INPUT_PULLUP LED_BUILTIN true | false Conversion byte() char() float() int() long() word()	String String() array bool boolean byte char double float int long short unsigned char unsigned int unsigned long void word	const scope static volatile Utilities PROGMEM sizeof()

STRUCTURE

The elements of Arduino (C++) code.

Sketch

loop()
setup()

Control Structure

break
continue
do...while
else
for
goto
if...else
return
switch...case
while

Further Syntax

#define (define)
#include (include)
/* */ (block comment)
// (single line comment)
; (semicolon)
{} (curly braces)

Arithmetic Operators

% (remainder)
* (multiplication)
+ (addition)
- (subtraction)
/ (division)
= (assignment operator)

Comparison Operators

!= (not equal to)
< (less than)
<= (less than or equal to)
== (equal to)
> (greater than)
>= (greater than or equal to)

Boolean Operators

! (logical not)
&& (logical and)
|| (logical or)

Pointer Access Operators

& (reference operator)
* (dereference operator)

Compound Operators

&= (compound bitwise and)
*= (compound multiplication)
++ (increment)
+= (compound addition)
-- (decrement)
-= (compound subtraction)
/= (compound division)
^= (compound bitwise xor)
|= (compound bitwise or)

Bitwise Operators

& (bitwise and)
<< (bitshift left)
>> (bitshift right)
^ (bitwise xor)
| (bitwise or)
~ (bitwise not)

Reference

https://www.arduino.cc/

http://www.farnell.com/datasheets/1682238.pdf