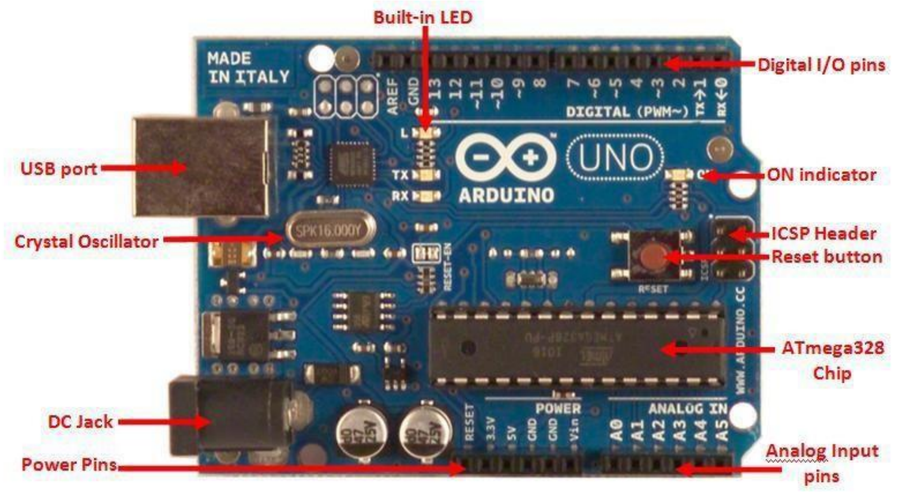
**Design and analysis of measurement applications using Arduino**

1. Introduction to Arduino, blink LED using external circuit.
2. Detection of motion using PIR sensor
3. Measurement of range using Ultrasonic sensor

## 1. Introduction to Arduino, blink LED using external circuit.

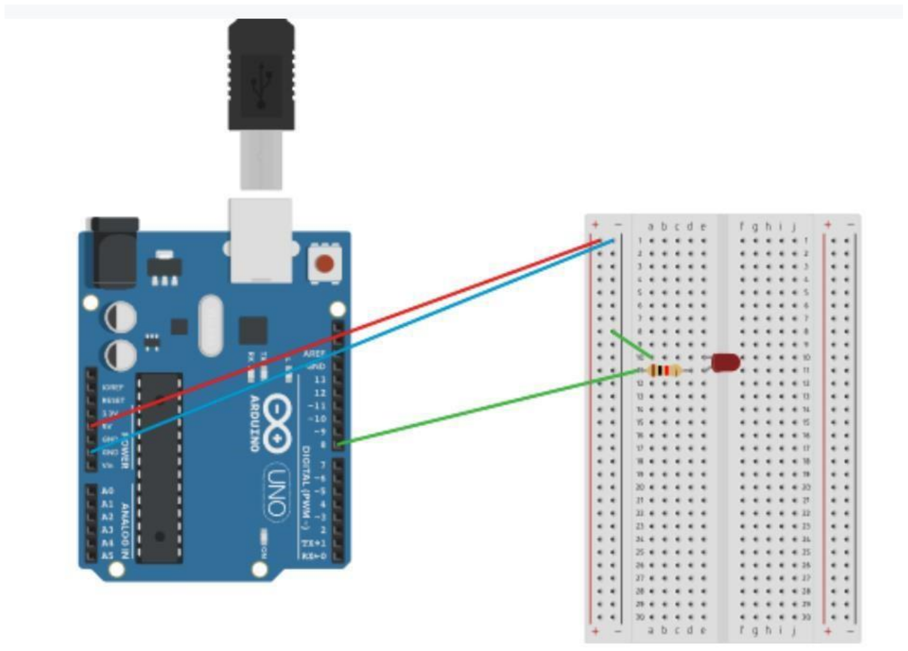
The Arduino (Uno) Board is a micro- controller board that was created to house the ATmega328 chip. The chip is a high performance and low power 8-bit microcontroller that has 23 programmable I/O lines, 32K bytes of flash memory (of which 0.5KB is already used for the Boot loader), 1k bytes of EEPROM and 2k bytes of
RAM. The Arduino Uno board provides the user with 6 analog input pins, 14 digital I/O pins of which 6 of them can also be used for PWM outputs, a power jack, a USB port, an ICSP header, a reset button, a small LED connected to digital pin 13, and a 16MHz crystal oscillator.

A pictorial view of the Arduino Uno board's peripherals is shown below:



**Analog input pins:** pins (A0-A5) that take-in analog values to be converted to be represented with a number range 0-1023 through an Analog to Digital Converter (ADC).
**ATmega328 chip:** 8-bit microcontroller that processes the sketch you programmed.
**Built-in LED:** in order to gain access or control of this pin, you have to change the configuration of pin 13 where it is connected to. **Crystal Oscillator:** clock that has a frequency of 16MHz
**DC Jack:** where the power source (AC-to-DC adapter or battery) should be connected. It is limited to input values between 6-20V but recommended to be around 7-12V.
**Digital I/O pins:** input and output pins (0-13) of which 6 of them (3, 5, 6, 9, 10 and 11) also provide PWM (Pulse Width Modulated) output by using the analogWrite() function. Pins (0 (RX) and 1 (TX)) are also used to transmit and receive serial data.
**ICSP Header:** pins for "In-Circuit Serial Programming" which is another method of programming.

**ON indicator:** LED that lights up when the board is connected to a power source. **Power Pins:** pins that can be used to supply a circuit with values VIN (voltage from DC Jack), 3.3V and 5V.
**Reset Button:** a button that is pressed whenever you need to restart the sketch programmed in the board.
**USB port:** allows the user to connect with a USB cable the board to a PC to upload sketches or provide a voltage supply to the board. This is also used for serial communication through the serial monitor from the Arduino software.



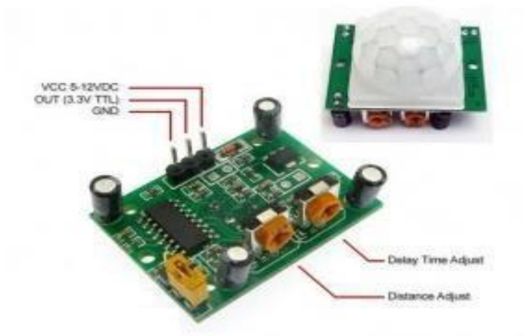
**Procedure:** Make necessary connections. Write, compile, and dump the code onto Arduino board using IDE. Test the circuit.

### Source code

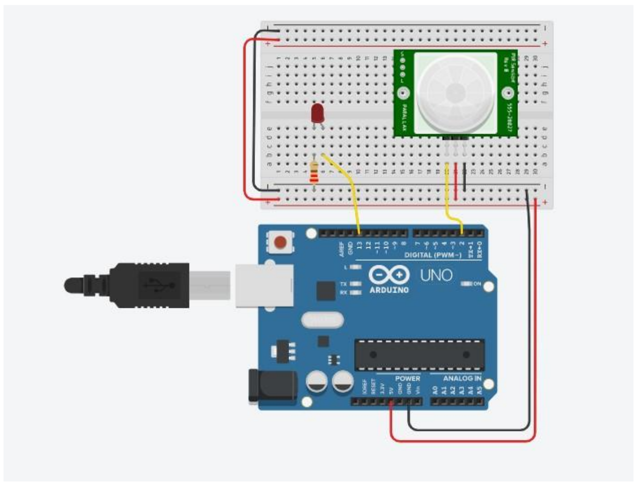
```
// the setup function runs once when you press reset or power
the board  void setup() {
      // initialize digital pin 8 as an output.
      pinMode(8, OUTPUT);
}
// the loop function runs over and over again forever  void loop()
{  digitalWrite(8, HIGH); // turn the LED on (HIGH is the voltage
level)  delay(500); // wait for half second
      (500ms)  digitalWrite(8, LOW); // turn the
      LED off by making the voltage
LOW  delay(500); // wait for half second(500ms)
}
```

## 2. Detection of motion using PIR sensor

A PIR sensor is generally known as **motion sensor** or **motion detector**. The working of PIR sensor is based on the fact that all objects emit heat energy in the form of radiation. All objects with a temperature above absolute zero (absolute zero is -273.15$_0$C or zero kelvin) emit heat energy in the form of radiation at infrared wavelengths (invisible to human eyes). These emitted infrared radiations can be detected with the help of PIR sensor. A PIR sensor do not emit any kind of radiation for detection purposes but they just measure the infrared radiation emitted by other objects inside its field or range of measurement.



PIR sensor consists of two sensing elements pyroelectric sensor and Fresnel lens. The sensor's Fresnel lens focuses a thermal image on the sensing elements which absorb the thermal energy from the image and convert it into heat. Further, this heat is converted into a minute electric current by the pyroelectric crystalline material.



**Procedure:**

1. Connect first (GND) pin of PIR Sensor to GND pin of Arduino
2. Connect second (Vout) pin of PIR Sensor to any Digital I/O pin (say pin 2) of Arduino
3. Connect third (Vcc) pin of PIR Sensor to 5V pin of Arduino 4. Connect positive terminal of LED to one lead of the resistor.
5. Connect other lead of the resistor to any digital I/O pin say pin 7 of Arduino
6. Write the Code and upload it to the Arduino board
7. Note the output by moving some object near to the PIR sensor **Source**

**Code:**

```
int led = 13;                // the pin that the LED is atteched to
int sensor = 2;              // the pin that the sensor is atteched
to
int state = LOW;             // by default, no motion detected int
val = 0;                 // variable to store the sensor status
(value)  void
setup() {
  pinMode(led, OUTPUT);      // initalize LED as an output
pinMode(sensor, INPUT);    // initialize sensor as an input
  Serial.begin(9600);        // initialize serial
}  void
loop(){
  val = digitalRead(sensor);   // read sensor value   if
(val == HIGH) {            // check if the sensor is HIGH
digitalWrite(led, HIGH);   // turn LED ON
    delay(500);                // delay 100 milliseconds

    if (state == LOW) {
      Serial.println("Motion detected!");
      state = HIGH;       // update variable state to HIGH
    }
}
else {
      digitalWrite(led, LOW); // turn LED OFF
      delay(500);             // delay 200 milliseconds

      if (state == HIGH){
        Serial.println("Motion stopped!");
        state = LOW;       // update variable state to LOW
    }
  }
}
```

## 3. Measurement of range using Ultrasonic sensor

**Theory:**

Ultrasonic sensors are great tools to measure distance without actual contact and used at several places like water level measurement, distance measurement etc. This is an efficient way to measure small distances precisely. Basic principal of ultrasonic distance measurement is based on ECHO. When sound waves are transmitted in environment then waves are return back to origin as ECHO after striking on the obstacle. So, by calculating the travelling time of both sounds means outgoing time and returning time to origin after striking on the obstacle. we can calculate the distance by following formula:

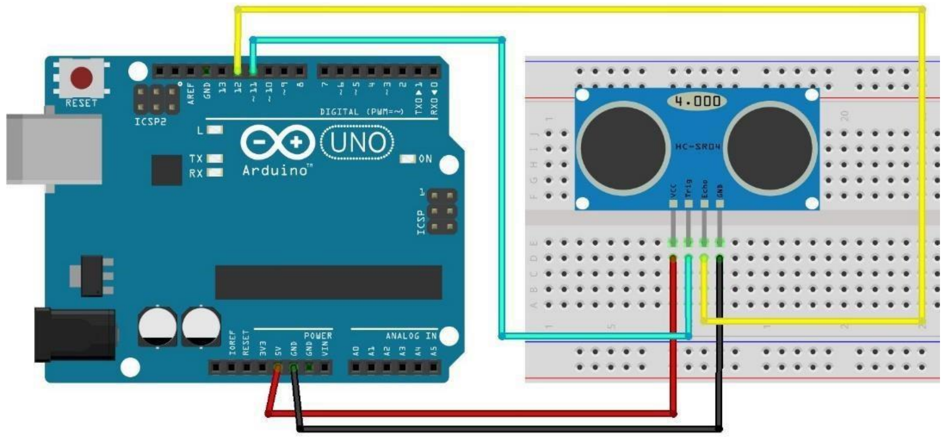
**Distance = (**travel time/2) X speed of sound in air (340 $\text{ms}_{-1}$)

The Ultrasonic sensor module consists of ultrasonic transmitter, receiver and the control circuit. The working principle of ultrasonic sensor is as follows:

 High level signal is sent for 10us using Trigger.

 The module sends eight 40 KHz signals automatically, and then detects whether pulse is received or not.

 If the signal is received, then it is through high level. The time of high duration is the time gap between sending and receiving the signal.



Procedure:

1. Connect GND pin of sensor to GND pin of Arduino
2. Connect TRIGGER pin of Sensor to any Digital I/O pin (say pin 11) of Arduino
3. Connect ECHO pin of Sensor to any Digital I/O pin (say pin 12) of Arduino
4. Connect VCC pin of sensor to 5V pin of Arduino
5. Write the Code and upload it to the Arduino board
6. Note the output by moving some object near to the PIR sensor

**Source code:**

```
#fdefine echoPin 2 // attach pin D2 Arduino to pin Echo of
HCSR04
#define trigPin 3 //attach pin D3 Arduino to pin Trig of HC-
SR04

// defines variables
long duration; // variable for the duration of sound wave
travel
int distance; // variable for the distance measurement
 void setup()
{
  pinMode(trigPin, OUTPUT); // Sets the trigPin as an OUTPUT
pinMode(echoPin, INPUT); // Sets the echoPin as an INPUT
Serial.begin(9600); // // Serial Communication is starting
with 9600 of baudrate speed
  Serial.println("Ultrasonic Sensor HC-SR04 Test"); // print
some text in Serial Monitor
  Serial.println("with Arduino UNO R3");
} void loop()
{
  // Clears the trigPin condition
digitalWrite(trigPin, LOW);   delayMicroseconds(2);
  // Sets the trigPin HIGH (ACTIVE) for 10 microseconds
digitalWrite(trigPin, HIGH);   delayMicroseconds(10);
digitalWrite(trigPin, LOW);
  // Reads the echoPin, returns the sound wave travel time in
microseconds
  duration = pulseIn(echoPin, HIGH);
// Calculating the distance
  distance = duration * 0.034 / 2; // Speed of sound wave
divided by 2 (go and back)
  // Displays the distance on the Serial Monitor
  Serial.print("Distance: ");
  Serial.print(distance);
Serial.println(" cm");
}
```