

# Temperature\_Measurement -- Overview

## Temperature Measurement using LM35 Sensor



## OBJECTIVES

After performing this lab exercise, learner will be able to:

- Program Arduino board to measure temperature.
- Convert the temperature changes to an equivalent voltage value using sensor
- Measure RMS / Mean voltage of the captured signal using inbuilt functions of the scope

## EQUIPMENT

To carry out this experiment, you will need:

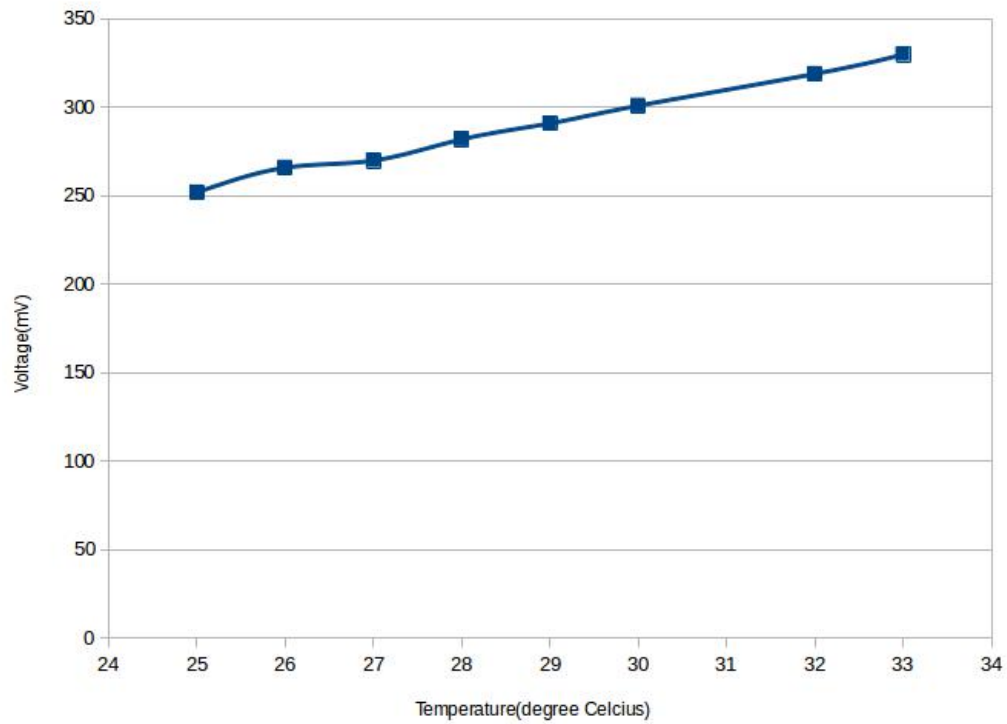
- TBS1KB - Digital Oscilloscope from Tektronix .
- Arduino Duemilanve or Uno board .
- Voltage probe (provided with oscilloscope) / BNC cables .
- Breadboard and connecting wires .
- LM35 Temperature Sensor.

## THEORY

- The LM35 series are precision integrated-circuit temperature sensors, with an output voltage linearly proportional to the Centigrade temperature.
- Thus the LM35 has an advantage over linear temperature sensors calibrated in ° Kelvin, as the user is not required to subtract a large constant voltage from the output to obtain convenient Centigrade scaling.
- The LM35 does not require any external calibration or trimming to provide typical accuracies of  $\pm 1/4^{\circ}\text{C}$  at room temperature and  $\pm 3/4^{\circ}\text{C}$  over a full  $-55^{\circ}\text{C}$  to  $+150^{\circ}\text{C}$  temperature range.
- RMS Value: Root Mean Square value of voltage.

## PLOT

Temperature Versus Voltage



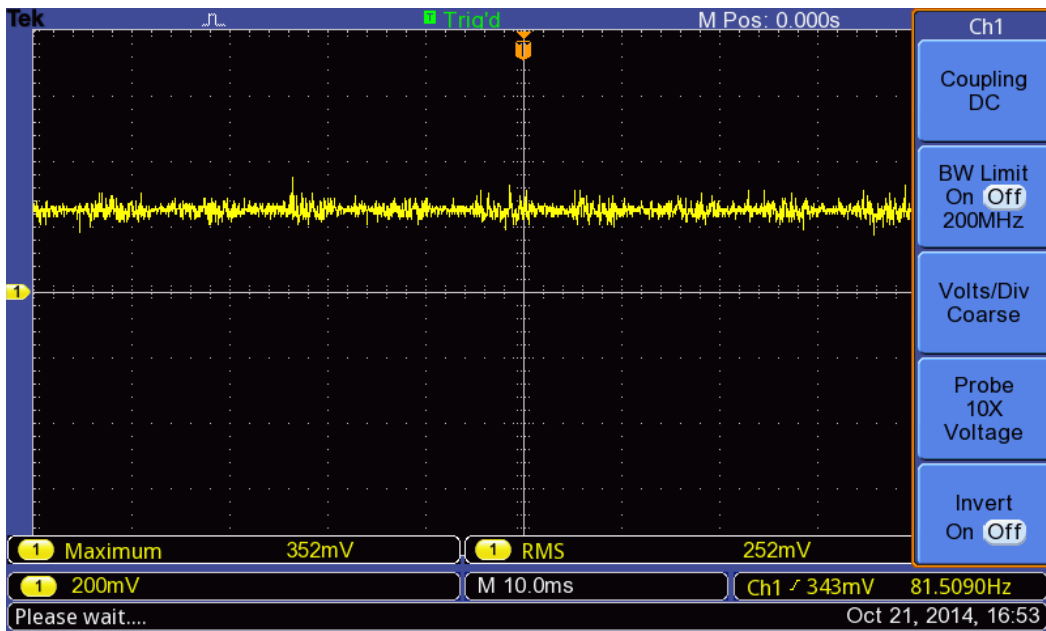
## ARDUINO CODE

```
void setup()
{
  pinMode(A0, INPUT);
  Serial.begin(9600);
}

void loop()
{
  int temp = analogRead(A0);
  float new_temp = (5*temp*100)/1023;
  Serial.println(new_temp);
  delay(1000);
}
```

## OUTPUT WAVEFORM

The output waveform would look like following, at Temperature = 25°C

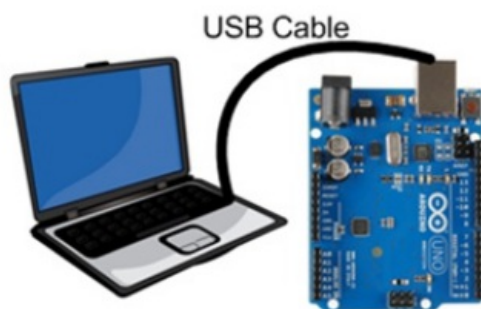


## Temperature\_Measurement -- Procedures

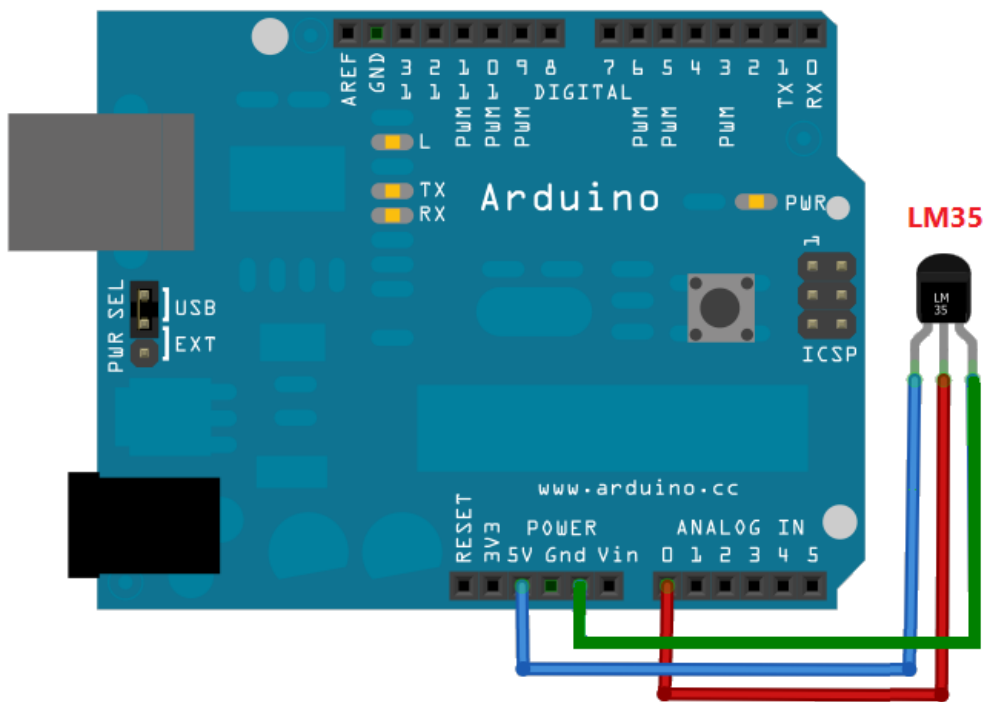
### Step 1

#### DUT / SOURCE SETUP

- Ensure you have Arduino IDE (software to program the Arduino boards) installed on your computer.
- Connect the Arduino board to PC using USB cable



- Program the Arduino board with given code
- Connect the LM35 sensor output to A<sub>0</sub> pin of Arduino



## Step 2

### MEASUREMENT / SCOPE SETUP

- Power ON the oscilloscope
- Connect the Channel 1 probe of the oscilloscope to  $A_0$  pin of Arduino
- Acquire the signal(s) from circuit on oscilloscope

## Step 3

- Do the Autoset on the scope to efficiently capture and view the signal
- If AUTOSSET feature is not enabled, then manually set the horizontal and vertical scale and trigger condition to view stable waveform without any clipping.

## Step 4

- From the measurement menu, configure RMS measurement on acquired channel

## Step 5

- Measure the RMS voltage at different temperatures.
- Tabulate the Temperature versus measured RMS voltage.

## Step 6

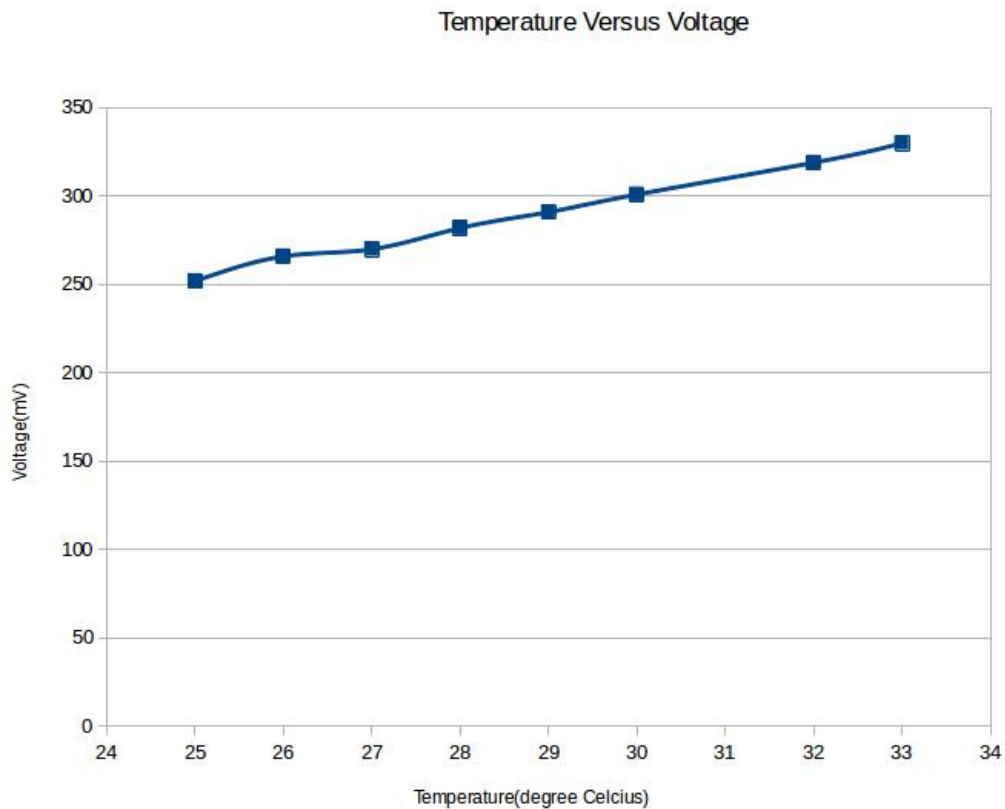
Measure the RMS voltage at different temperatures and fill out the observation table:

---

Temperature (degree Celcius)	Voltage (mV)
25	
26	
27	
28	
29	
30	
32	
33	

## Step 7

- From the tabulated Temperature versus measured RMS voltage, verify that it is a linear relationship. The plot would typically look like this:



# PIR -- Overview

## Human Movement Detection Using PIR Sensor



## OBJECTIVES

After performing this lab exercise, learner will be able to:

- Program Arduino board to detect movement.
- Interface PIR (Passive Infrared) Sensor with Arduino
- Measure MAXIMUM amplitude of the captured signal using inbuilt functions of the scope.

## EQUIPMENT

To carry out this experiment, you will need:

- TBS1KB - Digital Oscilloscope from Tektronix
- Arduino Duemilanve or Uno board
- Voltage probe (provided with oscilloscope) / BNC cables
- Breadboard and connecting wires
- PIR(Passive Infrared) Sensor

## THEORY

- PIR Motion detector module can detect human movement by detecting infrared emitted by human body.
- PIR motion detector module will output a high or low level or open collector signal when it detects human movement. So it is widely used on the motion activated lightening, security system, automatic sensory device and more.
- **Maximum Value:** Value of highest amplitude point in the acquired signal, measured in volts.

## ARDUINO CODE

```
void setup()  
{  
  Serial.begin(9600);  
  pinMode(2, INPUT);  
  pinMode(13, OUTPUT);  
}
```

```
void loop()
```

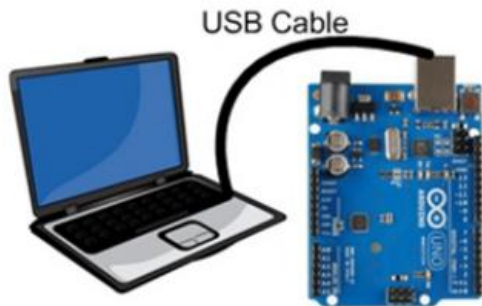
```
{  
  int senseVal = digitalRead(2);  
  Serial.println(senseVal);  
  digitalWrite(13, senseVal);  
}
```

## PIR -- Procedures

### Step 1

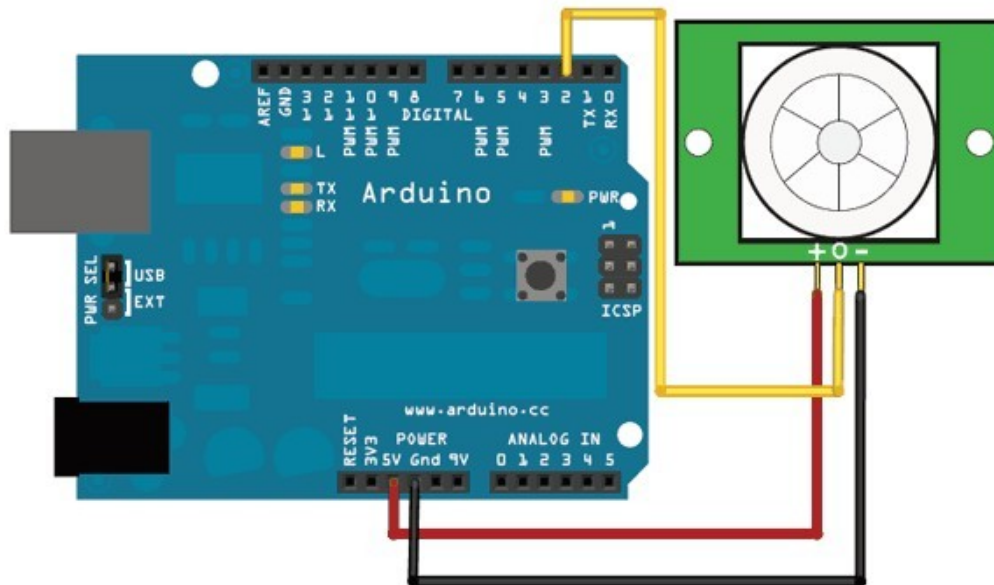
#### DUT / SOURCE SETUP

- Ensure you have Arduino IDE (software to program the Arduino boards) installed on your computer.
- Connect the Arduino board to PC using USB cable to program it



### Step 2

- Make the PIR sensor connection with Arduino as shown below



### Step 3

#### MEASUREMENT / SCOPE SETUP

- Power ON the oscilloscope

- Connect the Channel 1 probe of the oscilloscope to PIR sensor output (digital pin 2 of Arduino)
- Acquire the signal(s) from circuit on oscilloscope

## Step 4

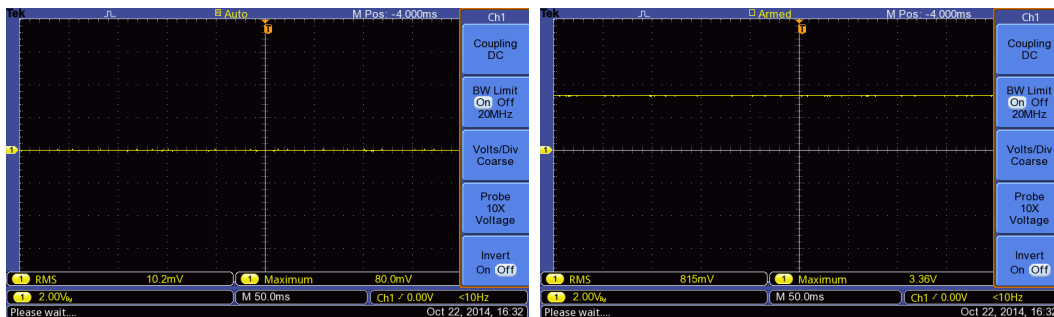
- Do the Autoset on the scope to efficiently capture and view the signal
- If AUTOSSET feature is not enabled, then manually set the horizontal and vertical scale, and trigger condition to view waveform without any clipping.
- Mostly we would be working on larger timebase ~ 20-50 ms/div

## Step 5

- From the measurement menu, configure MAXIMUM measurement on acquired channel

## Step 6

- Upload the code and check for human movement - without movement the signal will be low. When movement is detected, signal goes high and LED on pin 13 lights up.





# PWM -- Overview

## PWM Generation Using Arduino



## OBJECTIVES

After performing this lab exercise, learner will be able to:

- Program Arduino board to generate PWM output
- Control the duty cycle of the PWM output from Arduino
- Measure PWM (Pulse Width Modulation) duty cycle and positive width of the captured signal using inbuilt functions of the scope.

## EQUIPMENT

To carry out this experiment, you will need:

- TBS1KB - Digital Oscilloscope from Tektronix
- Arduino Duemilanve or Uno board
- Voltage probe (provided with oscilloscope) / BNC cables
- Breadboard and connecting wires

## THEORY

- Pulse-width modulation(PWM), or pulse-duration modulation (PDM), is a modulation technique that controls the width of the pulse, formally the pulse duration, based on modulator signal information.
- Maximum Value: Value of highest amplitude point in the acquired signal, measured in volts.

## ARDUINO CODE

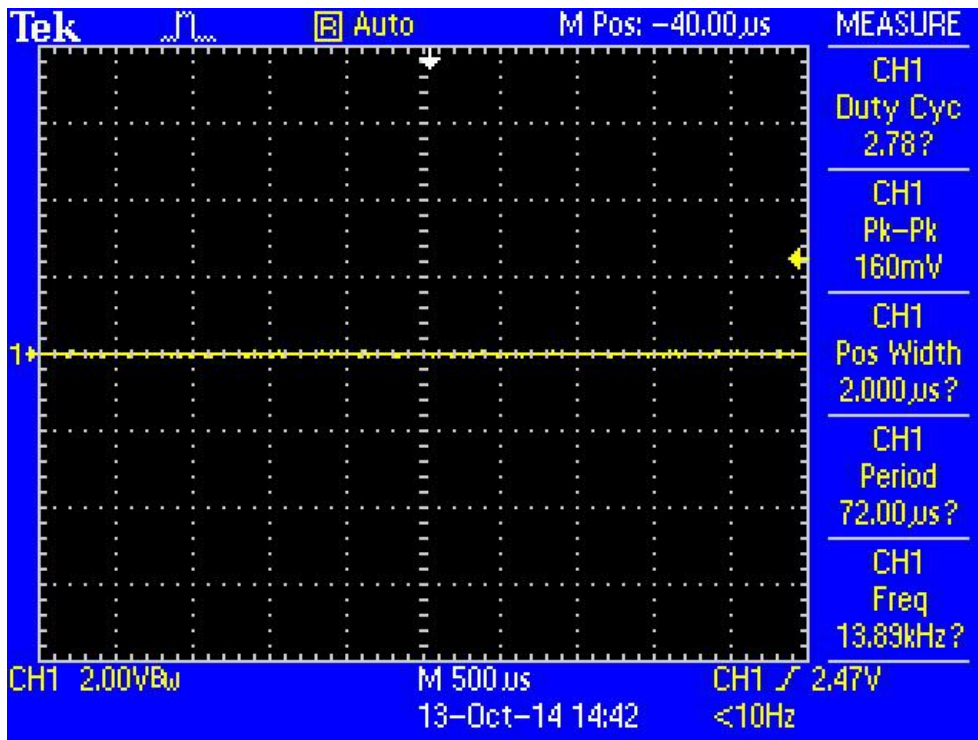
```
void setup()
{
  pinMode(9, OUTPUT);
}

void loop()
{
  analogWrite(9, value);//value can take values from 0 to 255
}
```

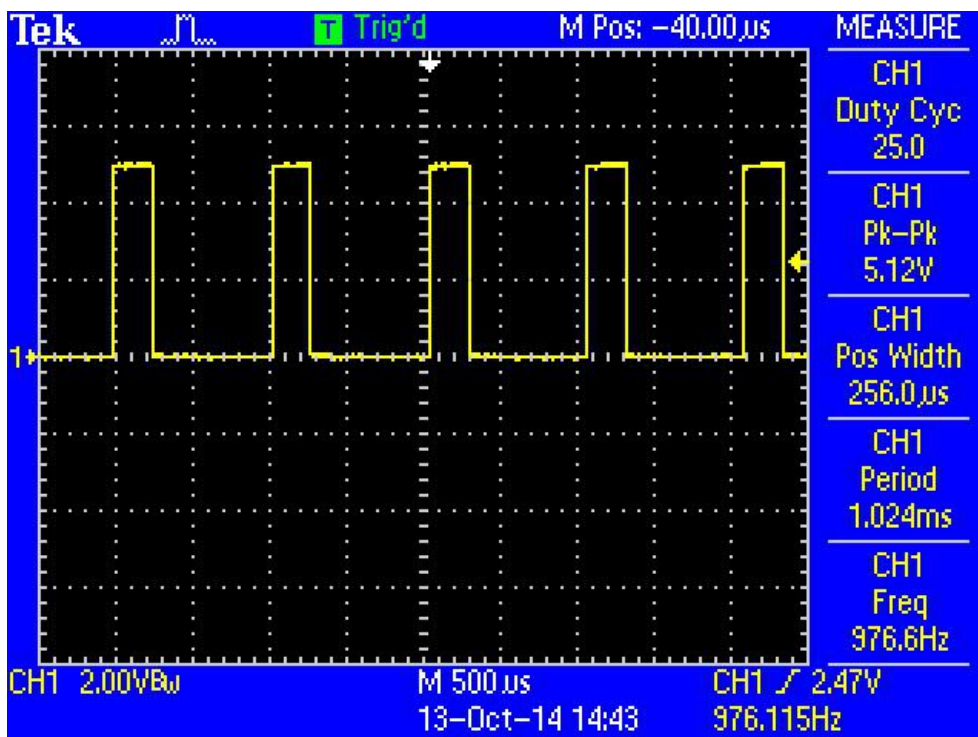
## OUTPUT WAVEFORM

The output would look like following,

- When value = 0 --> Duty Cycle expected =  $0/256 = 0\%$



- When value = 64 --> Duty Cycle expected =  $64/256 = 25\%$

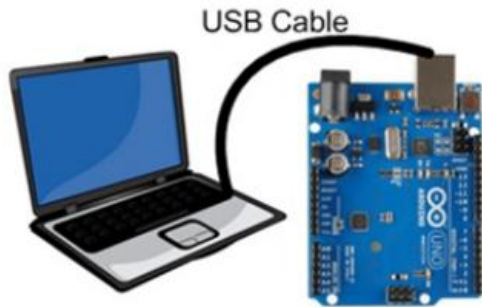


## PWM -- Procedures

### Step 1

## DUT / SOURCE SETUP

- Ensure you have Arduino IDE (software to program the Arduino boards) installed on your computer.
- Connect the Arduino board to PC using USB cable



## Step 2

## MEASUREMENT / SCOPE SETUP

- Power ON the oscilloscope
- Connect the Channel 1 probe of the oscilloscope to Pin # 9 of Arduino
- Acquire the signal(s) from circuit on oscilloscope

## Step 3

- Do the Autoset on the scope to efficiently capture and view the signal
- If AUTOSET feature is not enabled, then manually set the horizontal and vertical scale, and trigger condition to view 3-4 cycles of waveform without any clipping.

## Step 4

- From the measurement menu, configure **Duty Cycle** & **Positive Width** measurement on acquired channel

## Step 5

- Change the value of 'duty cycle' in the following line of the code: `analogWrite(9, value);` //value can take values from 0 to 255
- Upload the code and check the output pulse for different PWM values - Verify that you get the expected duty cycle.

# IR -- Overview

## Proximity Sensing Using IR Sensor



## OBJECTIVES

After performing this lab exercise, learner will be able to:

- Interface IR sensor with Arduino board
- Program Arduino board to measure small distances.
- Measure MAXIMUM amplitude of the captured signal using inbuilt functions of the scope.

## EQUIPMENT

To carry out this experiment, you will need:

- TBS1KB - Digital Oscilloscope from Tektronix
- Arduino Duemilanve or Uno board
- Voltage probe (provided with oscilloscope) / BNC cables
- Breadboard and connecting wires
- Sharp IR proximity sensor

## THEORY

- IR proximity sensor comprises of an IR emitter and a photodiode. IR emitter emits IR rays and photodiode detects the reflected radiations.
- Typically proximity sensors are used to measure distance of any object. As the object starts moving away from the sensor, the intensity of the reflected radiation keeps reducing.
- These radiations can be captured using a microcontroller and distance can be calculated accordingly.
- If the object is light coloured, it will reflect maximum radiations back to the sensor and if its dark coloured, it will absorb the majority of radiations and the sensor will detect a weak reflected radiation.
- **Maximum Value:** Value of highest amplitude point in the acquired signal, measured in volts.

## ARDUINO CODE

```
void setup()  
{  
  Serial.begin(9600);
```

```

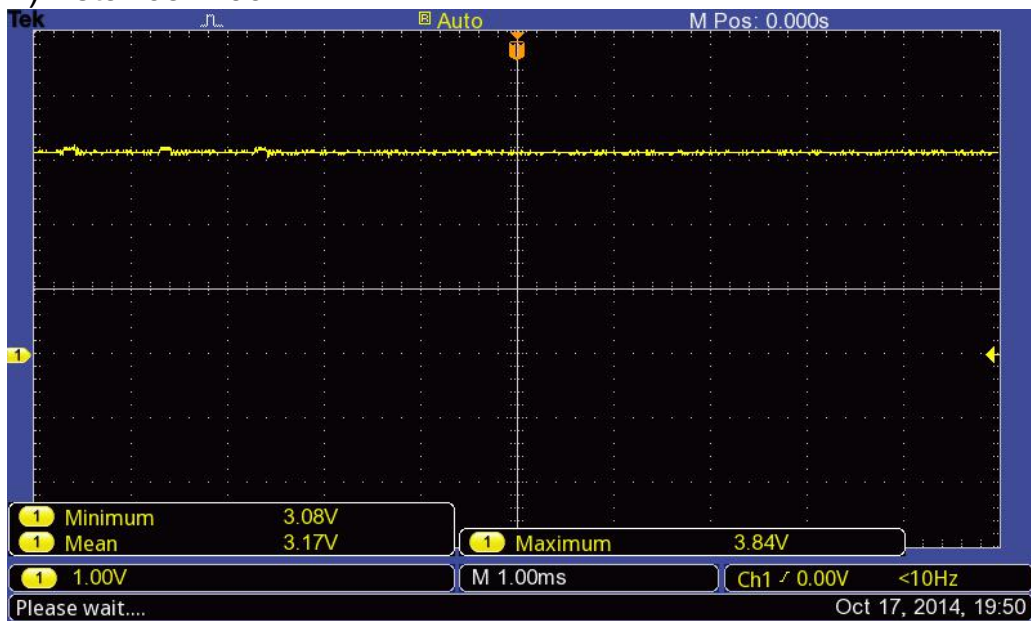
}

void loop()
{
  float volts = analogRead(A0)*0.0048828125;
  float distance = 65*pow(volts, -1.10);
  Serial.println(distance);
  delay(100);
}

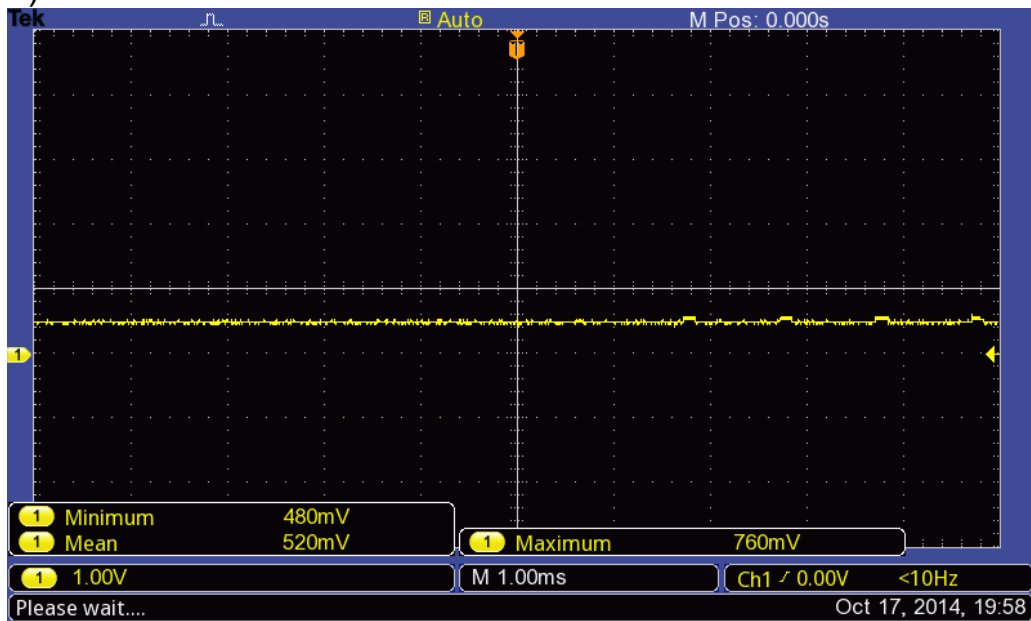
```

## OUTPUT WAVEFORM

1) Distance = 6cm



2) Distance = 55cm

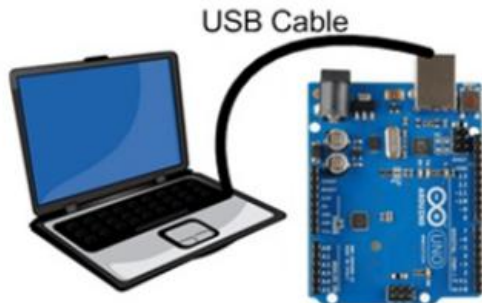


IR -- Procedures

## Step 1

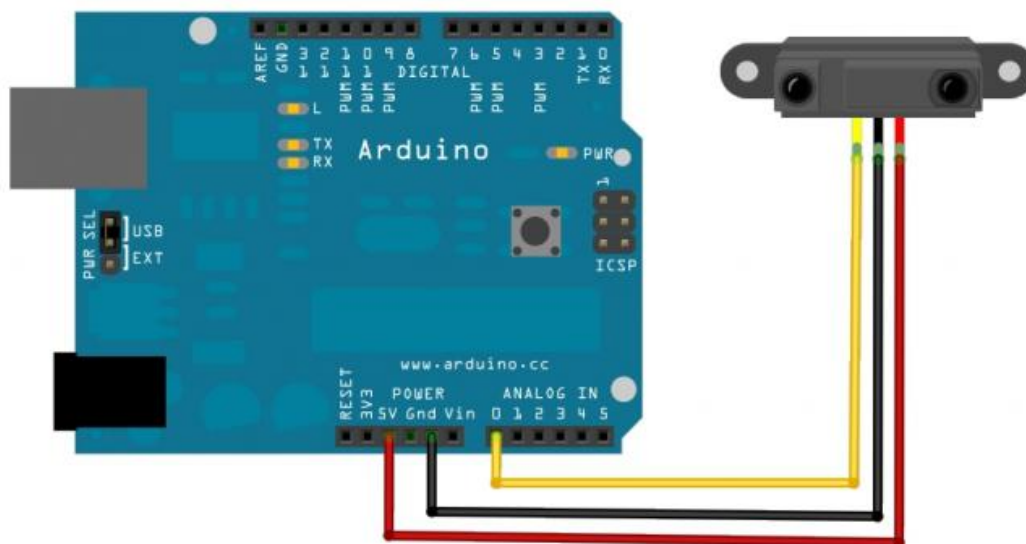
### DUT / SOURCE SETUP

- Ensure you have Arduino IDE (software to program the Arduino boards) installed on your computer.
- Connect the Arduino board to PC using USB cable



## Step 2

- Make the connections as shown below and fix the IR sensor at particular position.
- Keep a white body at a distance of 5cm from the sensor.
- Keep moving the object away from the sensor by 1cm and note the MAXIMUM measured voltage



## Step 3

### MEASUREMENT / SCOPE SETUP

- Power ON the oscilloscope
- Connect the Channel 1 probe of the oscilloscope to A0 pin of Arduino
- Acquire the signal(s) from circuit on oscilloscope

## Step 4

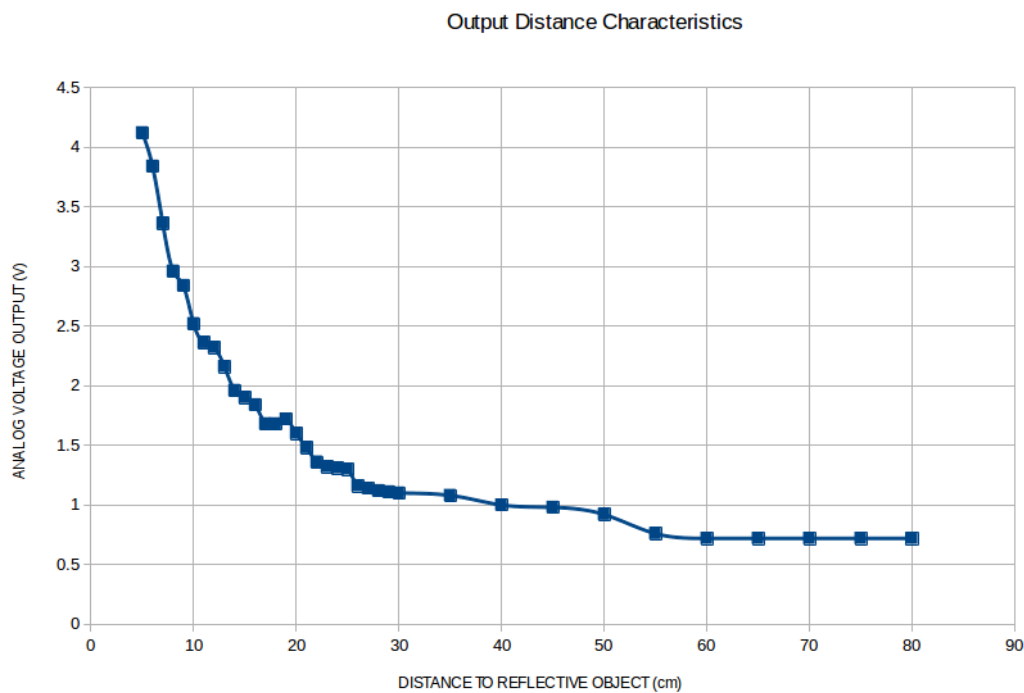
- Do the Autoset on the scope to efficiently capture and view the signal
- If AUTOSSET feature is not enabled, then manually set the horizontal and vertical scale, and trigger condition to view waveform without any clipping.

## Step 5

- From the measurement menu, configure MAXIMUM and MEAN measurements on acquired channel

## Step 6

- Vary the distance between object and IR sensor (5 to 40 cms with the step of 2 cms) and measure the voltage
- Tabulate the distance versus measured MAXIMUM voltage.
- Plot the voltage Vs distance graph and validate if it matches that of the datasheet of the IR sensor.



# POWER\_LED -- Overview

## Drive LED Using Transistor Driver Circuit



## OBJECTIVES

After performing this lab exercise, learner will be able to:

- Program Arduino board to generate PWM
- Design the Driver circuit using TIP127 TRANSISTOR.
- Measure MEAN & DUTY CYCLE of the captured signal using inbuilt functions of the scope.

## EQUIPMENT

To carry out this experiment, you will need:

- TBS1KB - Digital Oscilloscope from Tektronix
- Arduino Duemilanve or Uno board
- Voltage probe (provided with oscilloscope) / BNC cables
- Breadboard and connecting wires
- Resistor (1K & 10 Ohm)
- TIP127 Transistor

## THEORY

- Power Light Emitting Diode requires more current than the usual LED. Most of the GPIO pins of the controller can not supply such large current to drive LEDs. To provide the higher current we have to use driver circuit.
- Driver can be very simple, using little more than an NPN or PNP transistor to control the current. All that the low-power circuitry has to provide is enough base current to turn the transistor on and off.
- In our experiment NPN transistor is being used for driver. Series base resistor R is used to set the base current for transistor, so that the transistor is driven into saturation (fully turned on). By this way we can achieve minimal voltage drop across transistor.

## CIRCUIT DESIGN

- $R_b = (V_{CC} - V_{BE}) / I_B = (5 - 0.7) / (5 \times 10^{-3}) = 860 \Omega \approx 1K \Omega$
- $R_C = (V_{CC} - V_{CE}) / I_C = (9 - 3.6) / (350 \times 10^{-3}) = 15.42 \Omega$
- Power rating of collector resistance,



$$P_{RC} = (I_C)^2 \times R_C = (350 \times 10^{-3})^2 \times 55 \approx 1 \text{ W}$$

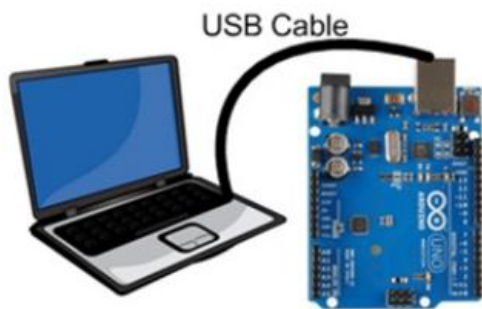
- PRC  $\approx$  1 Watt is the best choice based on availability.

## POWER\_LED -- Procedures

### Step 1

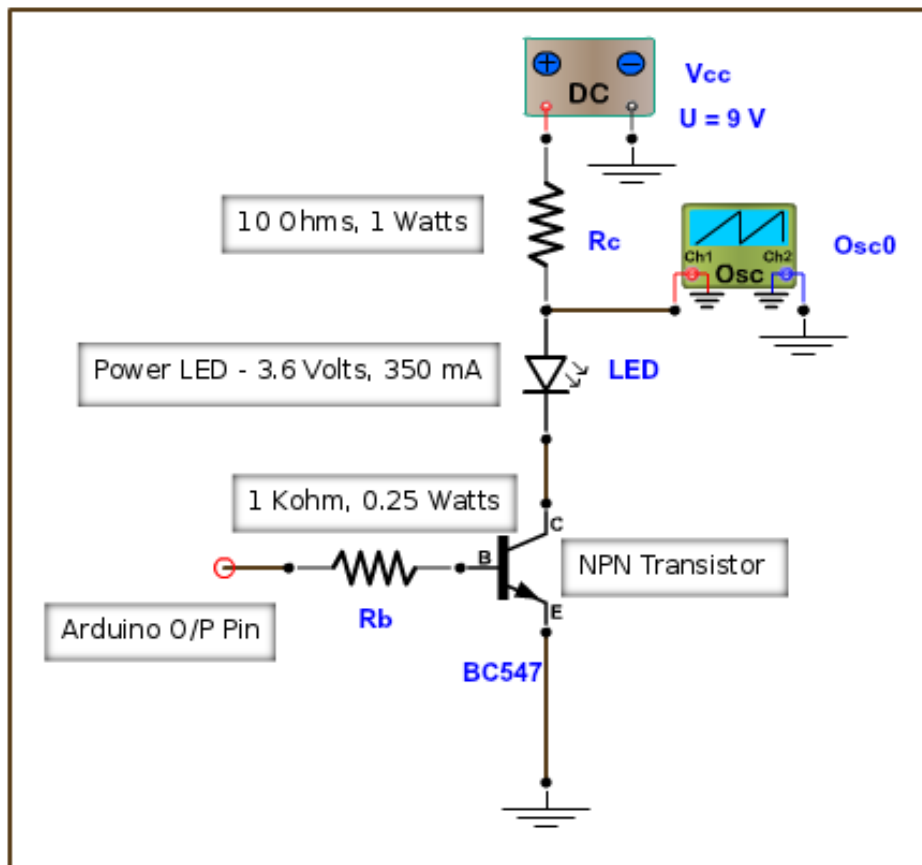
#### DUT / SOURCE SETUP

- Ensure you have Arduino IDE (software to program the Arduino boards) installed on your computer.
- Connect the Arduino board to PC using USB cable and program it with given code



### Step 2

- Rig up the circuit as shown.
- Measure the output across LED.



### Step 3

#### MEASUREMENT / SCOPE SETUP

- Power ON the oscilloscope
- Connect the Channel 1 probe of the oscilloscope to pin number 6 of Arduino and Channel 2 probe at the Anode of the LED

### Step 4

- Do the Autoset on the scope to efficiently capture and view the signal
- If AUTOSSET feature is not enabled, then manually set the horizontal and vertical scale, and trigger condition to view 3-4 cycles of waveform without any clipping.

### Step 5

- From the measurement menu, configure MEAN and DUTY CYCLE measurements on acquired channel - Higher the duty cycle, average value will be more - the LED will draw more current and give more brightness

# Preamp -- Overview

## Audio Signal Amplification Using LM741



## OBJECTIVES

After performing this lab exercise, learner will be able to:

- Understand the working of a basic audio amplifier
- Design an audio amplifier circuit using LM741 IC
- Convert the audio signal to electrical energy using MIC
- Measure MEAN & RMS amplitude of the captured signal using inbuilt functions of the scope.

## EQUIPMENT

To carry out this experiment, you will need:

- TBS1KB - Digital Oscilloscope from Tektronix
- LM741 Op-amp, Condensor MIC, resistors, capacitors
- Voltage probe (provided with oscilloscope) / BNC cables
- Breadboard and connecting wires

## THEORY

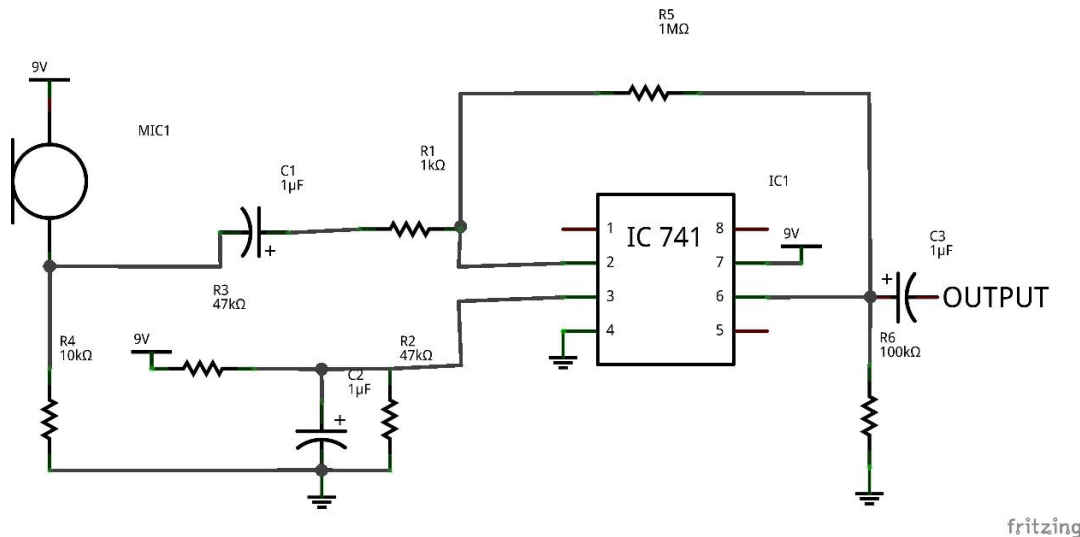
- A preamplifier (preamp) is an electronic amplifier that prepares a small electrical signal for further amplification or processing.
- A preamplifier is often placed close to the sensor to reduce the effects of noise and interference.
- Microphone detects the audio and converts them to electrical signals. Before sending the signal to amplifier, the signals are passed through the filters. Filtered signal is given to Op-Amp for amplification.

## Preamp -- Procedures

### Step 1

### DUT / SOURCE / CIRCUIT SET UP

- Rig up the circuit as shown in figure



## Step 2

### MEASUREMENT / SCOPE SETUP

- Power ON the oscilloscope
- Connect the Channel 1 probe of the oscilloscope to pin number 2 and Channel 2 probe to pin # 6 of LM741 IC
- Acquire the signal(s) from circuit on oscilloscope

## Step 3

- Do the Autoset on the scope to efficiently capture and view the signal
- If AUTOSSET feature is not enabled, then manually set the horizontal and vertical scale, and trigger condition to view 3-4 cycles of waveform without any clipping.

## Step 4

- From the measurement menu, configure MEAN and RMS measurement on acquired channels - Ch1 and Ch2

## Step 5

- Measure the initial MEAN and RMS value at Pin 2 (MIC output) and Pin 6 (Amplified signal) of LM741

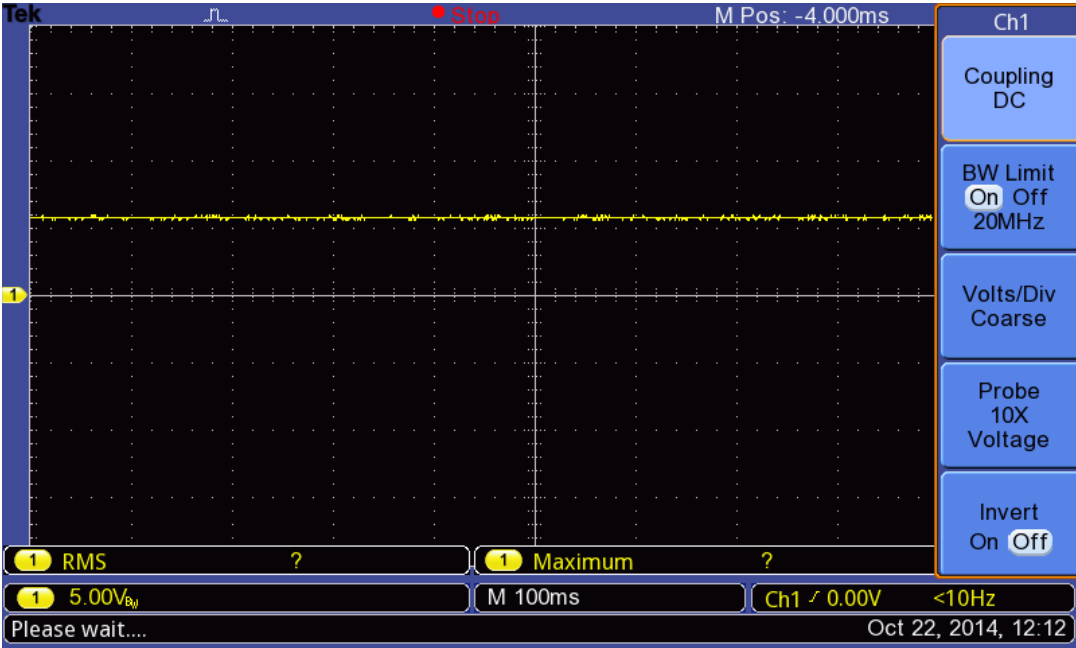
## Step 6

- Make some near mic and measure MEAN and RMS value at Pin 2 (MIC output) and Pin 6 (Amplified signal) of LM741
- Check if the amplification happens as expected or not.

## Step 7

# WAVEFORM

The output waveform would look like below,



# DCMOTOR\_PWM -- Overview

## DC Motor Speed Control Using PWM



## OBJECTIVES

After performing this lab exercise, learner will be able to:

- Program Arduino board to generate the PWM signals using Arduino UNO Board
- Interface DC motors using H-bridge circuit
- Control the speed of a 12V DC motor using PWM duty cycle

## EQUIPMENT

To carry out this experiment, you will need:

- 12 V 500rpm DC Motor
- H - Bridge
- ARDUINO Uno board
- Micro IR Sensor
- Resistor
- TBS1KB - Digital Oscilloscope from Tektronix
- Voltage probe (provided with oscilloscope) / BNC cables
- Breadboard and connecting wires

## THEORY

- PWM control is a very commonly used method for controlling the power across loads. This method is very easy to implement and has high efficiency.
- PWM signal is essentially a high frequency square wave (typically greater than 1KHz). The duty cycle of this square wave is varied in order to vary the power supplied to the load.
- Duty cycle is usually stated in percentage and it can be expressed using the equation :

$$\% \text{ Duty cycle} = 100 \times T_{\text{ON}} / (T_{\text{ON}} + T_{\text{OFF}})$$

Where  $T_{\text{ON}}$  is the time for which the square wave is high and  $T_{\text{OFF}}$  is the time for which the square wave is low

- When duty cycle is increased the power delivered to the load increases and when duty cycle is reduced, power across the load decreases.
- Control signal is what we give to the PWM controller as the

input. It might be an analog or digital signal according to the design of the PWM controller. The control signal contains information on how much power has to be applied to the load. The PWM controller accepts the control signal and adjusts the duty cycle of the PWM signal according to the requirements.

- The digital pins of the Arduino can provide the current of only up to 40 mA and hence cannot drive the motor directly. For fixing this problem, the L293D motor driver H-Bridge circuit is used to drive the motor according to the PWM signal available at digital pins.

## Measurement of motor speed using Magnetic Read Switch

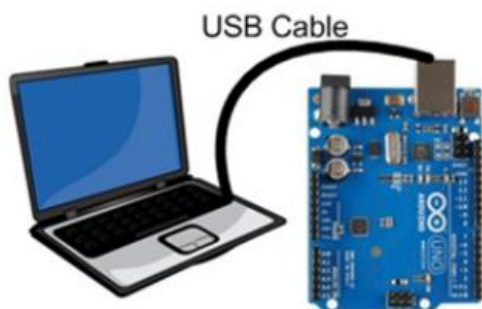
- The reed switch is an electrical operated switch by an applied external magnetic field.
- There are two contacts and it is kept open initially. If we apply external magnetic field nearby the contacts, then it gets closed. By using this principle, we have measured the speed of motor.
- Initially we have attached the robotic wheel in the shaft of motor and we have attached one small neodymium magnet in the outer surface of wheel. So, whenever the magnet comes near to the read switch, it gets closed and the signal has passed to the analog input of Arduino. We have measured the speed according to the time taken by one complete rotation of the wheel.

## DCMOTOR\_PWM -- Procedures

### Step 1

### DUT / CIRCUIT SETUP

- Ensure you have Arduino IDE (software to program the Arduino board) installed on your computer.
- Connect the Arduino board to PC using USB cable

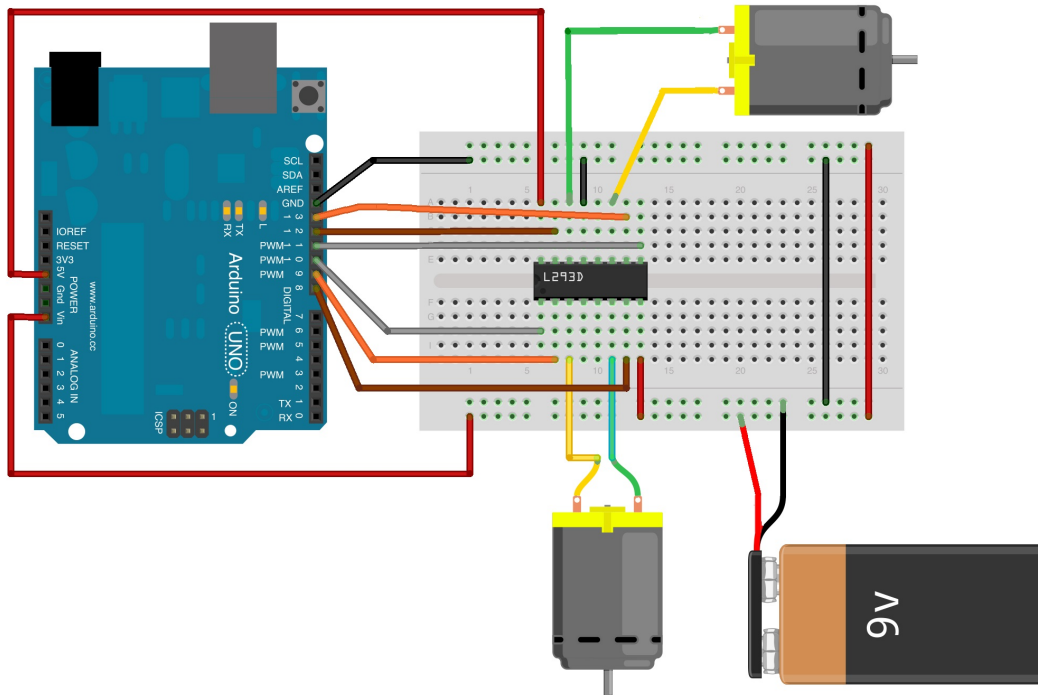


- Program the Arduino board with code

### Step 2

## DUT / CIRCUIT SETUP

- Build the circuit connection as given below:



### Step 3

- Power ON the oscilloscope
- Connect the Channel 1 probe and Channel 2 probe of the oscilloscope at the PWM output pin and ground pin of ARDUINO respectively.

### Step 4

- Do the Autoset on the scope to efficiently capture and view the signal
- If AUTOSSET feature is not enabled, then manually set the horizontal and vertical scale, and trigger condition to view waveform without any clipping

### Step 5

- Acquire the signal(s) from circuit on oscilloscope for different PWM pulses (say 25%, 50%, 75% & 100% duty cycle)

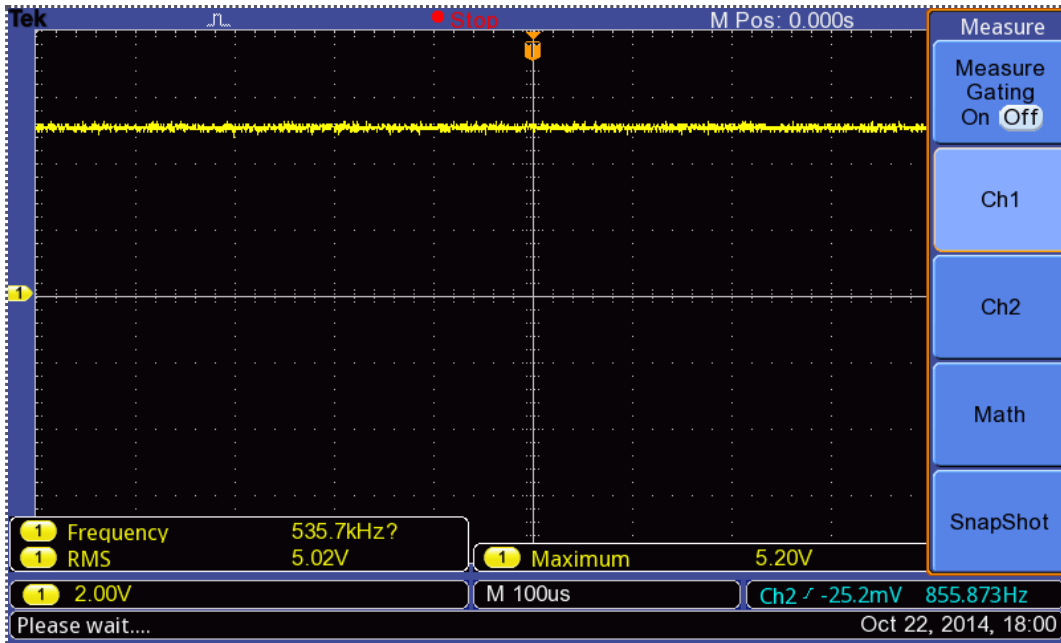
### Step 6

## OUTPUT WAVEFORM

The output waveform would look like,

100 % PWM duty cycle:





## Step 7

### RESULT

- Thus the design, construction and testing of the speed control of DC motor controlled by ARDUINO is completed successfully. The waveforms of the output voltage at different time delays were taken from the DSO.
- Relation between the PWM input and the speed of the motor is given in table below.

PWM Duty cycle	Motor speed (rpm)
100%	500
75%	375
50%	250
25%	125

# Relay\_12V -- Overview

## Interfacing 12V Relay with Arduino



## OBJECTIVES

After performing this lab exercise, learner will be able to:

- Design and construct the transistor driver circuit for 5 volts and 12 volts electromagnetic relays.
- Automate the relay control by programming a Arduino.

## EQUIPMENT REQUIRED

- TBS1KB - Digital Oscilloscope from Tektronix
- Relay - 5V / 12 V
- Resistor (1K-Ohm) – 2 Nos.
- Resistor (100K-Ohm) – 3 Nos.
- Diode (IN4007) – 2 Nos.
- 2N3906 NPN Transistor – 2Nos.
- DSO Probe
- USB Pendrive.

## THEORY

- Relays are components which allow a low-power circuit to switch a relatively high current on and off, or to control signals that must be electrically isolated from the controlling circuit itself.
- To make a relay operate, you have to pass a suitable 'pull-in' and 'holding' current (DC) through its energizing coil. And generally relay coils are designed to operate from a particular supply voltage often 12V or 5V, in the case of many of the small relays used for electronics work.
- In each case the coil has a resistance which will draw the right pull-in and holding currents when it's connected to that supply voltage. So the basic idea is to choose a relay with a coil designed to operate from the supply voltage you're using for your control circuit (and with contacts capable of switching the currents you want to control), and then provide a suitable 'relay driver' circuit so that your low-power circuitry can control the current through the relay's coil. Typically this will be somewhere between 25mA and 70mA.
- Often your relay driver can be very simple, using little more than an NPN or PNP transistor to control the coil current. All your low-power circuitry has to do is provide enough base current to turn

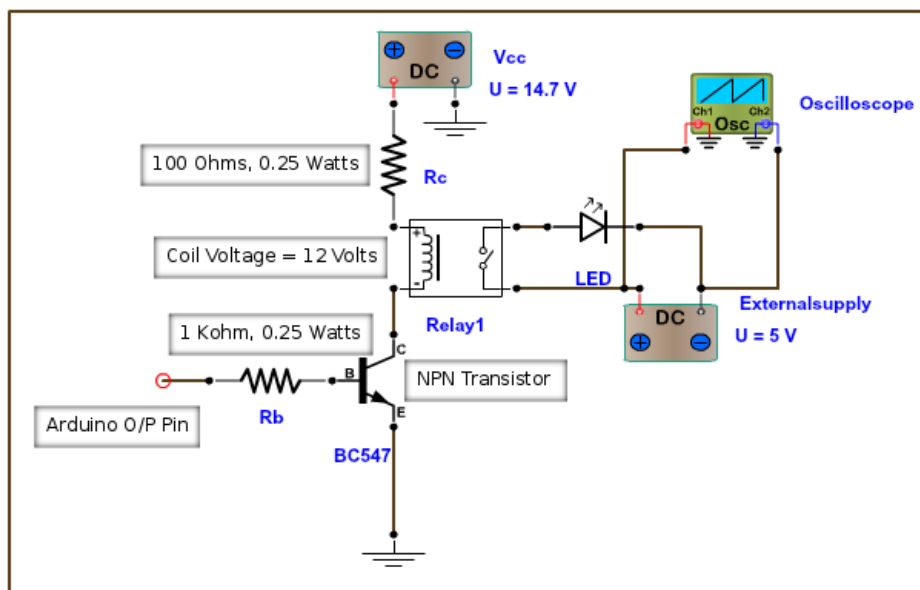
the transistor on and off.

- In A, NPN transistor Q1 (say 2N3904) is being used to control a relay (RLY1) with a 12V coil, operating from a +12V supply. Series base resistor R1 is used to set the base current for Q1, so that the transistor is driven into saturation (fully turned on) when the relay is to be energised. That way, the transistor will have minimal voltage drop, and hence dissipate very little power as well as delivering most of the 12V to the relay coil.

## CIRCUIT DESIGN

- $R_b = (V_{CC} - V_{BE}) / I_B = (5 - 0.7) / (5 \times 10^{-3}) = 860 \Omega \approx 1K \Omega$
- $R_C = (V_{CC} - V_{CE}) / I_C = (9 - 5) / (73 \times 10^{-3}) = 54.79 \Omega$
- Power rating of collector resistance,  $P_{RC} = I^2 R_C = (73 \times 10^{-3})^2 \times 55 = 0.2953 W$
- $P_{RC} \approx 0.25$  Watt is the best choice based on availability.

## CIRCUIT DIAGRAM



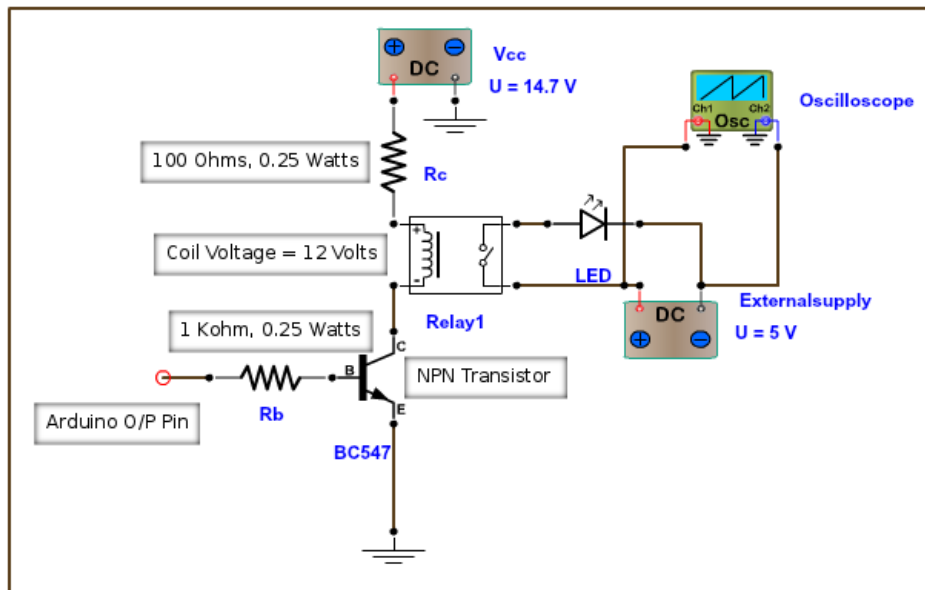
## Relay\_12V -- Procedures

### Step 1

- Ensure you have Arduino IDE (software to program the Arduino board) installed on your computer.
- Connect the Arduino board to PC using USB cable.
- Program the Arduino board with code

## Step 2

- Build the circuit connection as per the given circuit diagram.



## Step 3

### MEASUREMENT / SCOPE SETUP

- Power ON the oscilloscope.
- Connect the Channel 1 probe and Channel 2 probe of the oscilloscope across the relay contact terminals.

## Step 4

- Acquire the signal(s) from circuit on oscilloscope for different time delays.

## Step 5

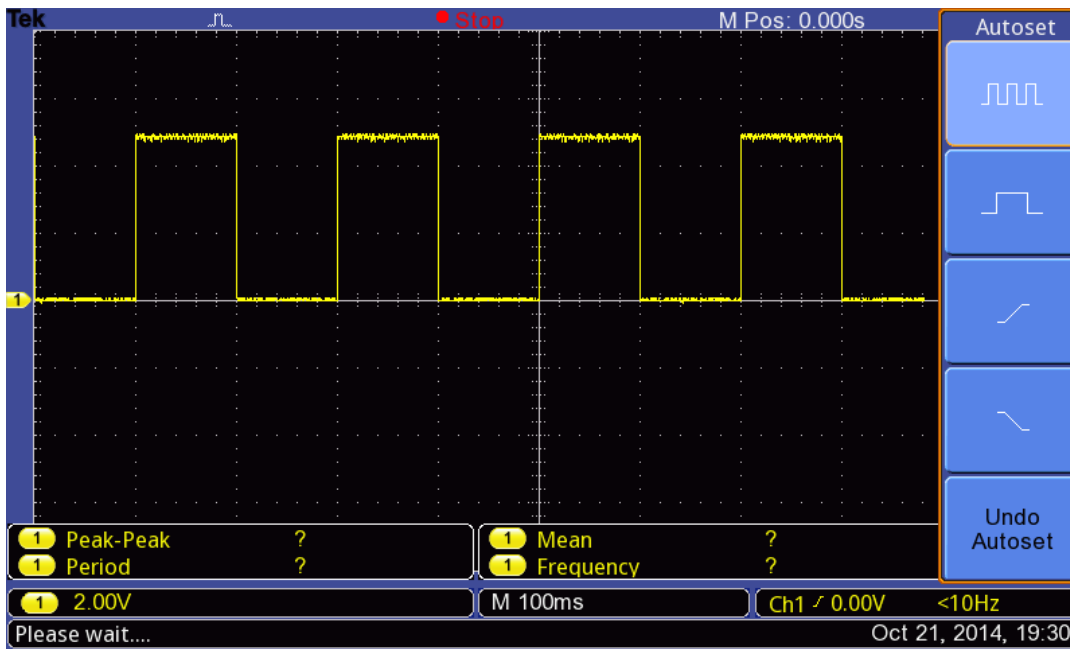
- Do the Autoset on the scope to efficiently capture and view the signal.
- If AUTOSSET feature is not enabled, then manually set the horizontal and vertical scale, and trigger condition to view waveform without any clipping.

## Step 6

### OUTPUT WAVEFORM

The output waveform would look like the following,

Time delay of 100 ms



## Step 7

### RESULT

Thus the design, construction and testing of transistor driver circuit to run relays (5V/12V) controlled by ARDUINO is completed successfully. The waveforms of the output voltage at different time delays were taken from the DSO.

# SERVO\_MOTOR -- Overview

## Controlling Servo Motor



## OBJECTIVE

After performing this lab exercise, learner will be able to:

- Program Arduino board to control Servo motor
- Measure Duty Cycle of the captured signal using inbuilt functions of the scope

## EQUIPMENT

- TBS1KB - Digital Oscilloscope from Tektronix
- Servo motor
- ARDUINO Uno board
- DSO Probes
- USB Pendrive (for saving measurement results / report)

## THEORY

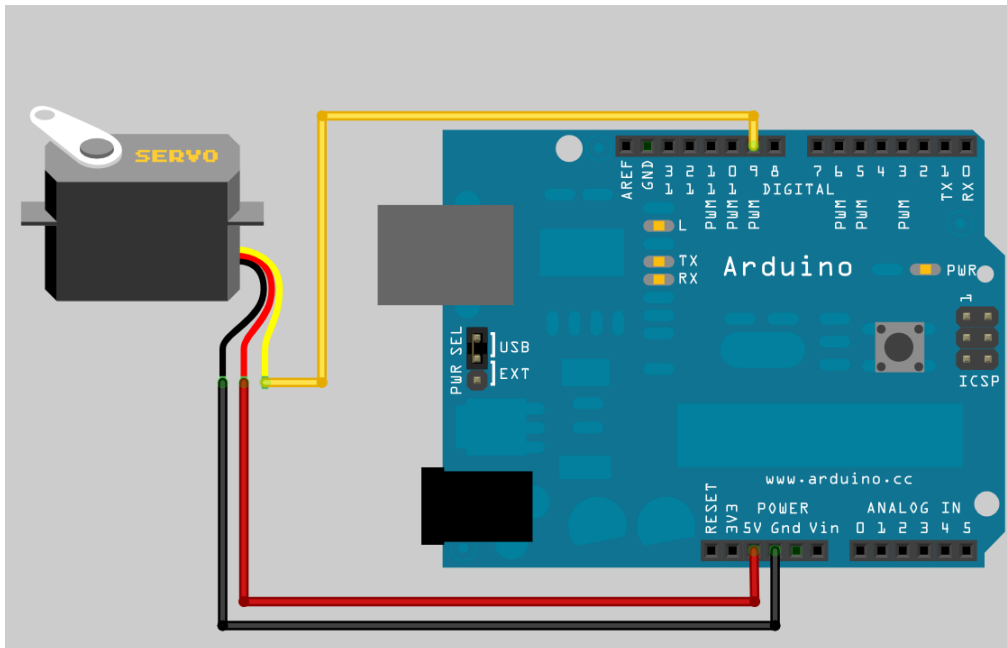
- A servomotor is a rotary actuator that allows for precise control of angular position, velocity and acceleration. It consists of a suitable motor coupled to a sensor for position feedback. It also requires a relatively sophisticated controller, often a dedicated module designed specifically for use with servomotors.
- Servo motors have three wires: power, ground, and signal. The power wire is typically red, and should be connected to the 5V pin on the Arduino board. The ground wire is typically black or brown and should be connected to a ground pin on the Arduino board. The signal pin is typically yellow, orange or white and should be connected to a digital pin on the Arduino board.
- Servos draw considerable power, so if you need to drive more than one or two, you'll probably need to power them from a separate supply (i.e. not the +5V pin on your Arduino). Be sure to connect the grounds of the Arduino and external power supply together.

## SERVO\_MOTOR -- Procedures

### Step 1

#### Circuit / DUT Setup

- Build the circuit connection as per the given circuit diagram



## Step 2

- Write and upload the program using ARDUINO IDE.

## Step 3

- Connect the oscilloscope across the digital pin 9 and ground pin of ARDUINO.

## Step 4

- Adjust the duty cycle of the PWM output to 20%, by tweaking the Arduino code. Run the updated code on Arduino.

## Step 5

- Do the Autoset on the scope to efficiently capture and view the signal.
- If AUTOSSET feature is not enabled, then manually set the horizontal and vertical scale, and trigger condition to view waveform without any clipping.

## Step 6

- Add DUTY CYCLE measurement on the scope for acquired channel

## Step 7

- Capture the PWM input to servo motor on the oscilloscope

- Observe the angle of the servo motor

## Step 8

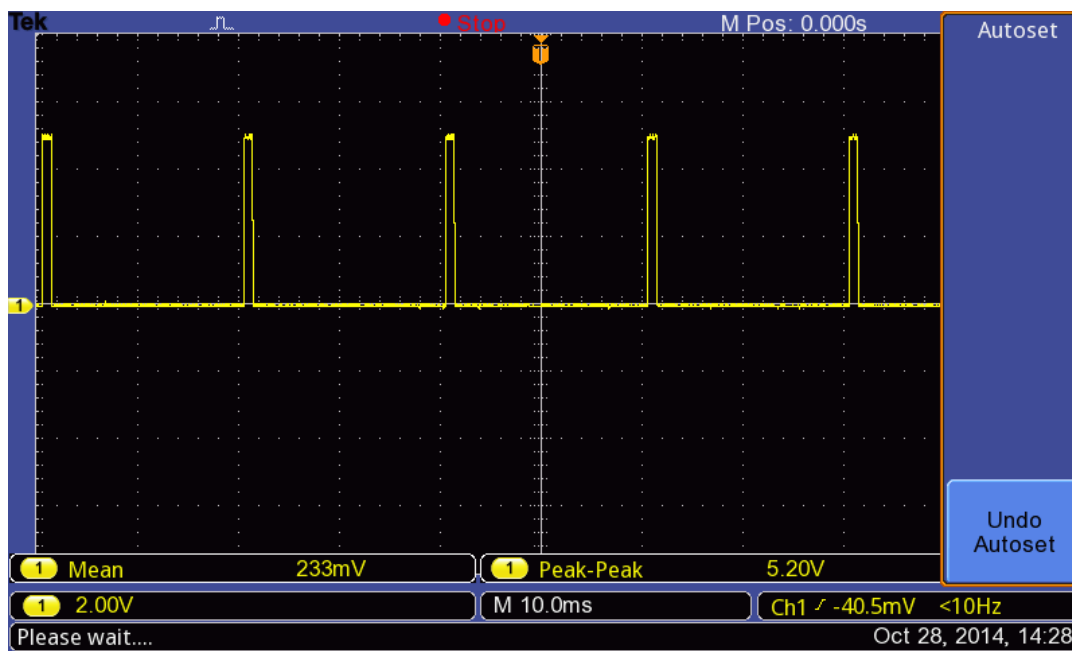
- Trying changing the PWM duty cycle and record the angle of servo motor for those input.

## Step 9

### OUTPUT WAVEFORM

The output waveform would look like the following,

Angle: 30degree



## Step 10

### RESULT

Plot the Angle (servo output) against the PWM duty cycle (%)



# AUDIO\_OUTPUT -- Overview



## OBJECTIVE

After performing this lab exercise, learner will be able to,

- Study sound intensity and frequency characteristics of buzzer with respect to variation in input voltage.

## EQUIPMENT REQUIRED

To carry out this experiment you will need:

- TBS1KB - Digital Storage Oscilloscope (DSO) from Tektronix
- DSO voltage probes / BNC Cable
- Resistor: 1K, 10K (2 nos.), 1M, 100K, 47K (2 nos.)
- Capacitor(0.1uF) – 2
- Buzzer – 1
- Microphone – 1
- Battery / DC Power Supply – 1
- Op-Amp(IC741)- 1
- Breadboard & Connecting wires

## THEORY

- Buzzer/speaker is an audio signalling device which may be mechanical, electromechanical or piezoelectric. Typical uses of buzzer/speaker include alarm devices, timers, confirmation of user input and even serves entertainment purpose.
- These audio signalling devices can be processed and used as source of application according to the user requirement. Processing of audio signal involves measuring of sound intensity and frequency. These two parameters are effectively used to process audio signal and at the same time these parameters vary in accordance to the input voltage given to them.
- In this experiment we will see how the output parameters change when variable input voltage is given to the buzzer. Since we are using buzzer, the output of microphone obtained is not sufficient to observe the changes in sound intensity and frequency. Hence we will use a pre-amplifier circuit which would take buzzer output as input and give the amplified signal as

output.

- Variable input voltage is obtained through voltage divider circuit built as given in circuit diagram.

## CIRCUIT DESIGN

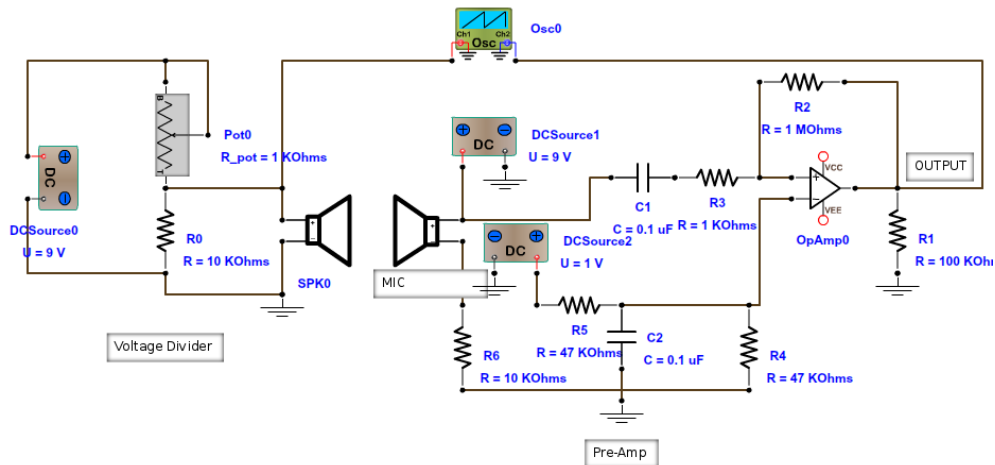
### a. Voltage Divider

- $V_{out} = V_{in} * R_2 / (R_1 + R_2)$
- $R_2 = 10K\text{ ohm}$   $R_1 = 1k\text{ POT}$

### b. Pre-amp circuit

- $\text{Gain} = V_{in} * (-R_f / R_1)$

## CIRCUIT DIAGRAM



## RESULT TABLE

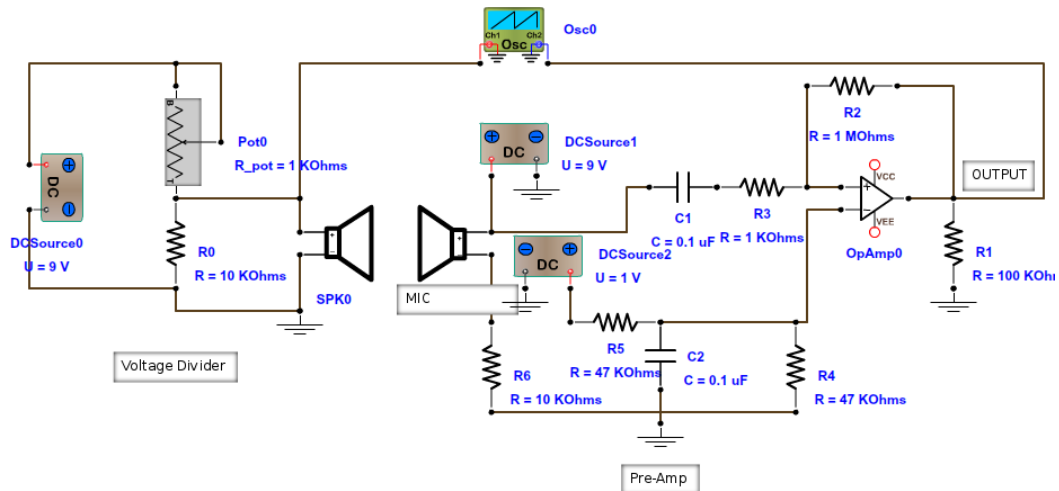
Input Voltage (V)	Output Voltage (V)	Frequency (kHz)
1.44	6.16	3.396
2.16	7.36	3.399
2.72	8.72	3.405
3.52	8.80	3.417
4.32	8.80	3.475
5.04	8.80	3.490

## AUDIO\_OUTPUT -- Procedures

## Step 1

### DUT / CIRCUIT SETUP

- Build the circuit connection as given in the circuit diagram



## Step 2

- Two channels of oscilloscope are required in order to have comparison of signals i.e input signal given to the voltage divider circuit and output signal across Pre-amp circuit , so that variation in sound intensity is observed accordingly

## Step 3

- At channel 1 connect probes across output of the voltage divider circuit and ground
- At channel 2 connect probes across output of Pre-Amp circuit i.e. PIN No 6 and ground

## Step 4

- Add RMS measurement for CH1 (input) and CH2 (output).
- Add frequency measurement for CH2 (output)

## Step 5

- Capture the input and output signal on scope.

## Step 6

- Observe the change in output waveform, i.e the change in sound intensity and frequency with respect to change in input voltage
- Save the output waveform in USB pen-drive.

## Step 7

# OUTPUT WAVEFORM

The output waveform would look like,

