



Unit 2: Output Devices

Schools of Specialized Excellence
Delhi Board of School Education



Session Plan
Output Devices

Teacher Name		Target Grade	10	Curriculum Component	Applied Learning Module
---------------------	--	---------------------	----	-----------------------------	-------------------------

Module Title	Output Devices		
Week Title	Output Devices	Week Number	1
Important Concepts	<ul style="list-style-type: none">- Introduction to Output Devices- Difference between Output Device & Input Device- Different types of Output Devices- Link b/w Output Devices and Arduino		

Learning Standards
<ol style="list-style-type: none"> 1. Explain the different Output Devices that can integrate with Controllers. 2. List out the type of Output Devices based on Applications. 3. Explain the Relay & their applications.

Inquiry Questions
<ol style="list-style-type: none"> 1. What do you know about Output Devices? 2. Give one Example of Output Devices of an Embedded System in our Day-to-day Life? 3. How do you choose Output Devices based on Applications?

Classroom Inquiry Process	
Day 1: Output Devices.	<p>Lesson Aims</p> <ol style="list-style-type: none"> 1. Explain the Output Devices Associated with Arduino Controllers. 2. Tabulate the difference between the output device & input device 3. Discuss the Different types of output devices 4. Demonstrate the Link Between Output Devices and Arduino <p>Activity Title:</p> <ol style="list-style-type: none"> 1. Ice-breaking on Output Devices(10 Mins) 2. Introduction to Output Devices(20 Mins) 3. Difference between Output Device & Input Device(30 Mins)



4. Different types of Output Devices (30 Mins)
5. Link b/w output Device and Arduino (10 Mins)
6. Doubt Clarification/Q & A Session (10 Mins)
7. Instructions for taking Home Assignment (10 Mins) (Individual)

Activity Description:

1. **Ice-breaking on Embedded System:** At the beginning, show them the Real-Life Application, which uses output devices like Relay, and RGB Led. Provide a brief description of the output device link between the output device & Arduino.
2. **Introduction to output device:** Brief introduction on the output device using an example. Explain How to identify the output device.

What is meant by an output device?

The Arduino output devices are the devices that take instructions from the Arduino and provide output in human-readable form. An example can understand the concept of an output device if an appliance works on the values coming from the sensor, the appliance will be an output device, and the sensor will be an input device. There are several output devices that we can interface with Arduino microcontrollers. So, we have listed some of the output devices that can use Arduino.

- LEDs
- Motors
- Liquid crystal displays (LCDs)
- Relays
- Seven segment displays
- Speaker and buzzer

3. **Difference between output device & input device:**

Introduce the **output device & input device using** live Electronic Appliances & Explain the Differences based on Features, Components, Working & Applications.

	Input Device	Output Device
	It is a hardware device used to key in the computer's data, instructions, or commands.	It is a hardware component that uses the data received from the computer to carry out a task.
	It can transfer data to another device but cannot receive any from it	Can obtain data from another device and produce output from it. However, it cannot transfer data to another device
	Necessary for the computer to receive user commands and data to process	Required if a computer has to share its results. They also help to prompt the users for additional information and commands.
	These are user-controlled	Computer manages them.
	Complex coding is used.	Users only need to see the results and are not required to learn the process
	Examples: Keyboard, webcam, microphone, joystick, and so on.	Examples: LCD Projection panels, printers, monitors, speakers, and more
<p>4. Different types of output devices:</p> <p>Light emitting diode (LED): The LEDs can be used as an output device with Arduino as they can serve various purposes in the projects; for example, the colour of the LED can be associated with any type of indication. The indication can be for turning on an appliance or a circuit or can use to indicate if there is any fault in the circuit. Similarly, there are various types of indications that can be made using the LED. Again, more than one LEDs can also be used for multiple indications in a project.</p>		



Motors:

Arduino boards can also be used to drive DC motors, which can run or control another device. For example, if water is to be pumped using Arduino, the DC motor will be an output device. The device will run when the water level falls below the level given in the Arduino program.

Similarly, motors can be used to drive fans and, in short, to move any device. We can use motors with Arduino.



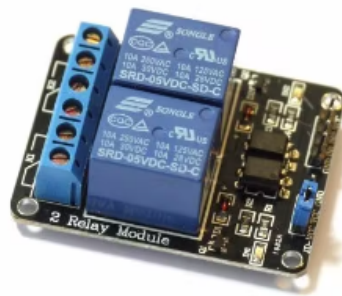
Liquid Crystal Displays (LCDs):

To display the output of the Arduino program, digital displays are interfaced with Arduino. These displays come in multiple sizes/resolutions; the most common is 16×2. This size is the most commonly used size by the students in their different projects. The purpose of the LCD is just to display the output data coming from the Arduino board or the input data given by any input device to the Arduino board.



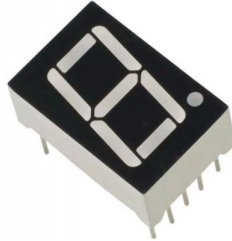
Relays:

Relays are generally used to protect the devices in the circuit as it isolates the faulty part of the circuit from the whole circuit. Another purpose of the relays is to use them to switch from one device to another. The Arduino boards mostly use relays as an output device, as they can switch from one device to another. For example, an Arduino board can switch between two or three motors based on the load, time interval, or voltage level. Similarly, it can shift from the regular supply to the backup supply in case of power failure.



Seven-Segment Displays:

The seven-segment displays are used when the output of Arduino is only in the form of numbers, for example, in digital counters, clocks, or electronic meters. A single module of the seven segments can display numbers from 0 to 9. It comes in two configurations: a common anode and a common cathode.



Speaker and Buzzer:

In Arduino projects, the speakers or buzzers are interfaced as an output device. These can be used for sounding an alarm or playing any melody on specific frequencies. Similarly, speakers can also be used for giving voice commands in case of the occurrence of an event.



5. Link b/w output devices and Arduino:

Features like Memory, Power Supply, Pin Configuration
Working like Processing Methods, Speed, and Performance
An application like Home Automation, Remote Controlled Toys

6. Doubt Clarification /Q & A Session- Do the Learning check using these Prompt Questions:

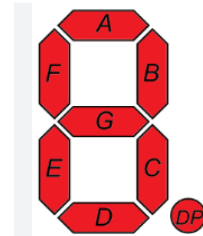
- Give an example of Output devices of Arduino Controllers used for Display Purposes.
- What is the difference between an Input Device & Output device based on Characteristics?
- List out the Application of Relay used in our Home Appliances.

7. Take-Home Assignment

Make a Table format Based on the Types of Input & Output Devices of Arduino Controllers in your Handout.

	<p>Refer to This Video: https://www.youtube.com/watch?v=NCAfju-ukbA</p> <p>References</p> <p>YouTube video https://youtu.be/sUcBKLxw-ro</p> <p>RC Car: https://www.youtube.com/watch?v=fddhx2KOKGo</p> <p>Home Automation: Home Automation Project using Arduino[Code Explained] Engineering Projects Arduino tutorial 22</p>
<p>Day 2:</p> <p>Output Devices - 7 Segment Display</p>	<p>Lesson Aims</p> <ol style="list-style-type: none"> 1. Explain about 7-segment display 2. Describe the Working of the 7-segment display 3. Discuss the types of 7-segment display 4. Integrate the 7-segment display with Arduino <p>Activity Title:</p> <ol style="list-style-type: none"> 1. Ice-breaking on 7-Segment Display (10 Mins) 2. Introduction to 7-Segment Display (20 Mins) 3. How does it work (30 Mins) 4. Seven Segment Types (30 Mins) 5. Link b/w 7-segment display and Arduino (10 Mins) 6. Doubt Clarification/Q & A Session (10 Mins) 7. Instructions for taking Home Assignment (10 Mins) (Individual) <p>Activity Description:</p> <ol style="list-style-type: none"> 1. Ice-breaking on Embedded System: At the beginning, Show them the Real-Life Application like Traffic Signals, Shopping Mall Car Parking, Token systems in Bank, Hospitals & Role of the 7-segment display in each Application. Also, Show Them the Latest output devices used for Display Purposes. Video: https://www.youtube.com/watch?v=A8c_Gpkpslk 2. Introduction to 7-segment display: Brief introduction on the 7-segment display using an example. <p>A seven-segment display is a form of an electronic display device for displaying decimal numerals that is an alternative to the more complex dot matrix displays.</p> <p>Seven-segment displays are widely used in digital clocks, electronic meters, basic calculators, and other electronic devices that display</p>

numerical information



3. How does it work?

Let's briefly discuss the characteristics and functionality of the 7-segment display before we connect it to an Arduino.

The 7-segment displays are just seven LEDs lined up in a particular pattern. In this case, the number '8' shape. Each of the seven LEDs is called a segment because when illuminated, the segment forms part of a numerical digit (both Decimal and Hex) to be displayed. An additional 8th LED is sometimes used to indicate a decimal point.

Each of the seven LEDs in the display is given a positional segment, with one of its connection pins being brought straight out of the rectangular plastic package. These individual LED pins are labelled from a through to g representing each LED. The other LED pins are connected and wired to form a common pin.

To turn a particular display part on and off, you set the appropriate pin HIGH or LOW, just like a regular LED.

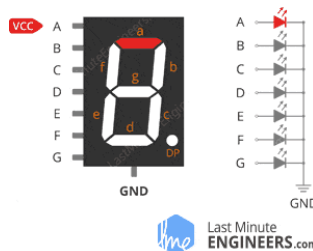
So that some segments will be light and others will be dark, allowing the desired character pattern of the number to be generated on display. This allows us to display each of the ten decimal digits 0 through to 9 on the same 7-segment display.

4. Seven Segment types:

Seven Segment displays are Common Cathode (CC) and Common Anode (CA). The Internal structure of both types is nearly the same. The difference is the polarity of the LEDs and the common terminal. As their name suggests, the common cathode has all the cathodes of the LEDs

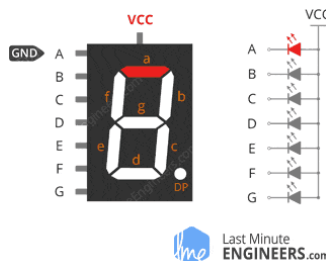
in a 7-segment connected, and the common anode has all the anodes of the LEDs in a 7-segment connected.

In the common cathode display, all the cathode connections of the LED segments are connected to 'logic 0' / GND. The individual segments are then illuminated by applying a HIGH / 'logic 1' signal to the individual Anode terminals (a-g).



Common Cathode 7 Segment Working:

In the common anode display, all the LED segments' anode connections are joined to logic "1". The individual segments are illuminated by applying a ground, logic "0" or "LOW" signal to the Cathode of the particular segment (a-g).



Common Anode 7 Segment Working:

How to know the type of the seven segments?

You can see the type by testing it; when you connect one of the centre pins from the seven segments to GND and connect one of the other pins to VCC, and it turns on, that means the seven segments are common cathodes. Otherwise, the seven-segment is a common anode.

5. Link b/w output devices and Arduino:

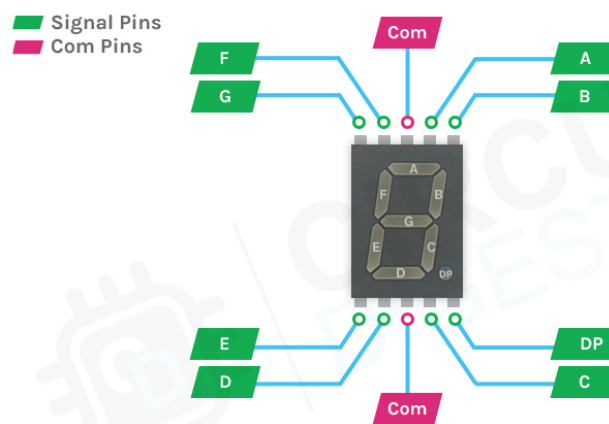
Objective: To Display Numbers between 0 to 9.

Components Required:

- Arduino UNO - 1 No

- Resistor 330 Ohm-8 Nos (7 Nos to Display Only Numbers)
It can also be done with one single 1k OHM resistance at a common pin attached to the ground rather than using eight separate resistances.
- Seven Segment Display-1 No
- Jumper Wires

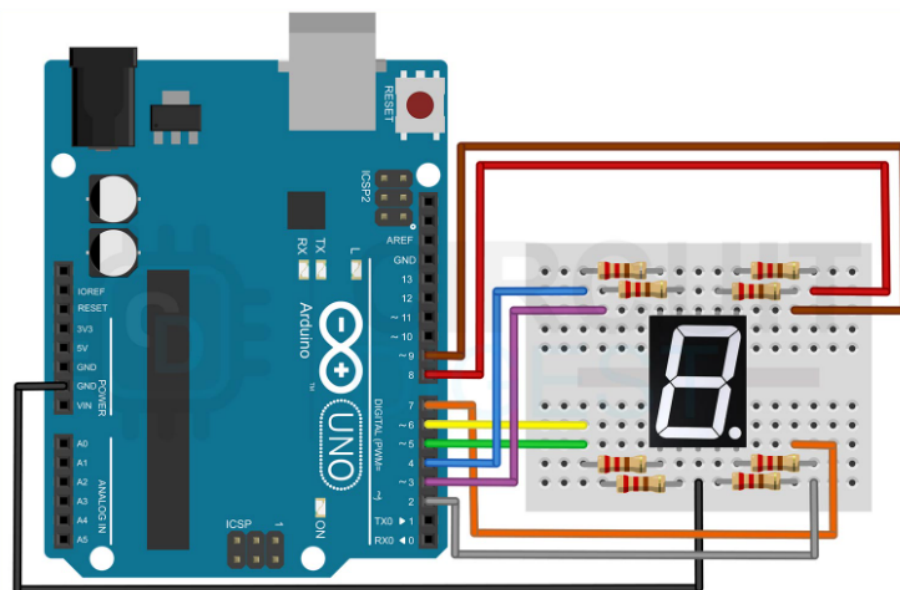
Seven Segment Display Pin Configuration:

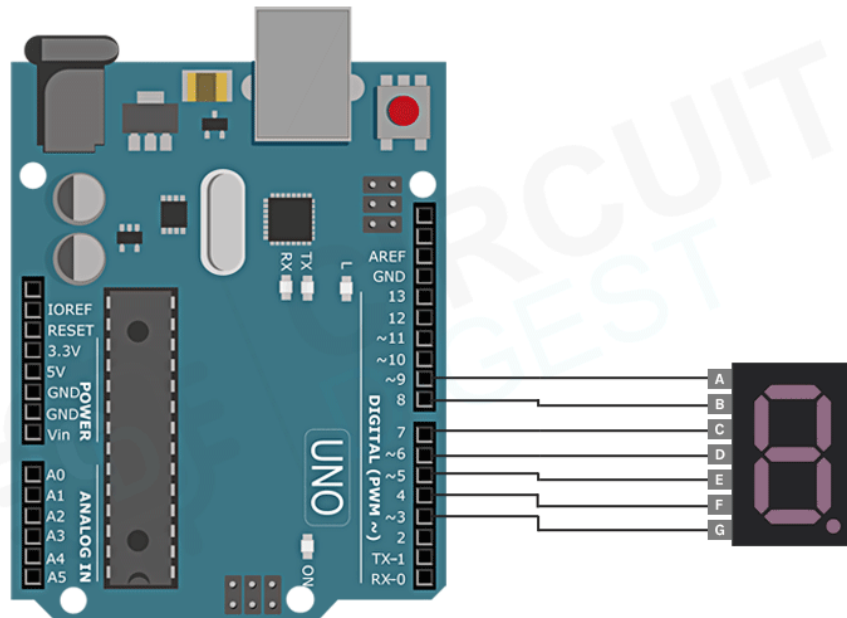


Seven segment Display with Arduino:

You can test them by connecting the centre to (GND to cathode / VCC to anode) and any other pin to (VCC to cathode / GND to anode).

The Schematic shows the connections between the seven segments and the Arduino.





Coding:

Download the Library Files using this link:

<https://github.com/DeanIsMe/SevSeg>

CODE FOR 7 SEGMENT DISPLAY

// DECLARING THE PIN NUMBER ON ARDUINO BOARD

int a=2;

int b=3;

int c=4;

int d=5;

int e=6;

int f=7;

int g=9;

void setup()

// DECLARING THE OUTPUT PINS

{

pinMode(a,OUTPUT);

pinMode(b,OUTPUT);

pinMode(c,OUTPUT);

pinMode(d,OUTPUT);

pinMode(e,OUTPUT);

```
pinMode(f,OUTPUT);
pinMode(g,OUTPUT);
}
void loop()
{
// TO DISPLAY NUMBER 9
digitalWrite(a,HIGH);
digitalWrite(b,HIGH);
digitalWrite(c,HIGH);
digitalWrite(d,HIGH);
digitalWrite(e,LOW);
digitalWrite(f,HIGH);
digitalWrite(g,HIGH);

// TO DISPLAY NUMBER 8
digitalWrite(a,HIGH);
digitalWrite(b,HIGH);
digitalWrite(c,HIGH);
digitalWrite(d,HIGH);
digitalWrite(e,HIGH);
digitalWrite(f,HIGH);
digitalWrite(g,HIGH);

//TO DISPLAY NUMBER 7
digitalWrite(a,HIGH);
digitalWrite(b,HIGH);
digitalWrite(c,HIGH);
digitalWrite(d,LOW);
digitalWrite(e,LOW);
digitalWrite(f,LOW);
digitalWrite(g,LOW);

// TO DISPLAY NUMBER 6
digitalWrite(a,HIGH);
digitalWrite(b,LOW);
digitalWrite(c,HIGH);
digitalWrite(d,HIGH);
digitalWrite(e,HIGH);
digitalWrite(f,HIGH);
digitalWrite(g,HIGH);

//TO DISPLAY NUMBER 5
```

	<pre>digitalWrite(a,HIGH); digitalWrite(b,LOW); digitalWrite(c,HIGH); digitalWrite(d,HIGH); digitalWrite(e,LOW); digitalWrite(f,HIGH); digitalWrite(g,HIGH); // TO DISPLAY NUMBER 4 digitalWrite(a,LOW); digitalWrite(b,HIGH); digitalWrite(c,HIGH); digitalWrite(d,LOW); digitalWrite(e,LOW); digitalWrite(f,HIGH); digitalWrite(g,HIGH); // TO DISPLAY NUMBER 3 digitalWrite(a,HIGH); digitalWrite(b,HIGH); digitalWrite(c,HIGH); digitalWrite(d,HIGH); digitalWrite(e,LOW); digitalWrite(f,LOW); digitalWrite(g,HIGH); //TO DISPLAY NUMBER 2 digitalWrite(a,HIGH); digitalWrite(b,HIGH); digitalWrite(c,LOW); digitalWrite(d,HIGH); digitalWrite(e,HIGH); digitalWrite(f,LOW); digitalWrite(g,HIGH); // TO DISPLAY NUMBER 1 digitalWrite(a,LOW); digitalWrite(b,HIGH); digitalWrite(c,HIGH); digitalWrite(d,LOW); digitalWrite(e,LOW); digitalWrite(f,LOW);</pre>
--	--



	<pre>digitalWrite(g,LOW); //TO DISPLAY NUMBER 0 digitalWrite(a,HIGH); digitalWrite(b,HIGH); digitalWrite(c,HIGH); digitalWrite(d,HIGH); digitalWrite(e,HIGH); digitalWrite(f,HIGH); digitalWrite(g,LOW);</pre> <p>6. Doubt Clarification /Q & A Session- Do the Learning check using these Prompt Questions:</p> <ul style="list-style-type: none"> • List out the Applications of Seven Segment LED. • What is the Reason Behind the Seven segment Display looks like Numeric 8? <p>7. Take-Home Assignment Make a one Page Learning Report (Components Required, Procedure & Learning Outcomes) on how to display Numeric 2 using Seven segment display with Arduino.</p> <p>Refer This Video: https://www.youtube.com/watch?v=3m4jhmaf8E</p> <p>References YouTube video: https://www.youtube.com/watch?v=fzGd0M-OmZE Working Video: https://drive.google.com/file/d/1cTwBwUOMHL0K2PW20AByx2J8Pr_rphN/view</p>
Day 3: Display Devices -16X2 LCD	<p>Lesson Aims</p> <ol style="list-style-type: none"> 1. Explain about 16X2 LCDs 2. Demonstrate the Working of 16X2 LCDs 3. Integrate 16X2 LCDs and Arduino 4. Interface I2C LCD with Arduino <p>Activity Title:</p> <ol style="list-style-type: none"> 1. Ice-breaking on 16X2 LCD (10 Mins) 2. Introduction to 16X2 LCD (10 Mins) 3. 16X2 LCD Pinout (30 Mins) 4. The interface between 16X2 LCD and Arduino (20 Mins)

5. Hands-on Activity (30 Mins)
6. Doubt Clarification / Q & A Session (10 Mins)
7. Instructions for taking Home Assignment (10 Mins) (Individual)

Activity Description:

1. **Ice-breaking on 16X2 LCD:** At the beginning, show them the Real-Life Application like Portable Video Games, Camera, and Camcorder & Explain the Role of the 16X2 LCD in each Application.

Videos: https://www.youtube.com/watch?v=KD_IBlyrVc

2. **Introduction to 16X2 LCD:**

An LCD is a unique type of display that can only output individual ASCII characters with a fixed size. Using these individual characters then, we can form a text.

Refer to this for ASCII Code/ Characters: <https://www.ascii-code.com/>.

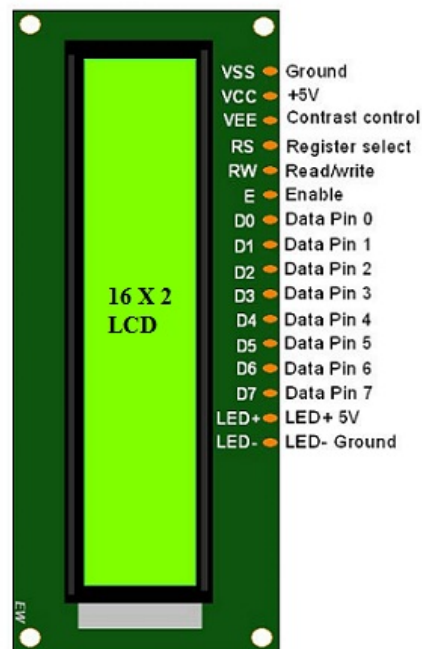
The number of rectangular areas defines the size of the LCD. The most popular LCD is the 16X2 LCD, which has two rows with 16 rectangular areas or characters. Of course, there are other sizes like 16X1, 16X4, 20X4, and so on, but they all work on the same principle. Also, these LCDs can have different backgrounds and text colours. Backgrounds and text colours.

3. **16X2 LCD Pinout**

- o It has 16 pins; the first one from left to right is the **Ground** pin. The second pin is the **VCC** which we connect the 5 volts pin on the Arduino Board.
- o Next is the Vo pin on which we can attach a potentiometer to control the display's contrast.
- o Next, The **RS** pin or register select pin is used for selecting whether we will send commands or data to the LCD. For example, if the RS pin is set to a low state or zero volts, we send commands to the LCD, like setting the cursor to a specific location, clearing the display, turning off the display, and so on.
- o If the RS pin is set to a High state or 5 volts, we send data or characters to the LCD.
- o Next comes the **R/W** pin, which selects the mode of whether we will read or write to the LCD. Here, the write mode is obvious and used for writing or sending commands and data to the LCD. The LCD uses the

read mode when executing the program, which we don't need to discuss in this tutorial.

- o Next is the **E** pin enables the writing to the registers or the next 8 data pins from D0 to D7. So, through these pins, we send the 8 bits data when we are writing to the registers, or for example, if we want to see the latter uppercase A on display, we will send 0100 0001 to the registers according to the ASCII table.
- o The last two pins, **A** and **K (LED+ & LED- Pins)** or anode and cathode, are for the LED backlight.



4. Link b/w output devices and Arduino:

Procedure to interface LCD Displays & Arduino Controller:

1. Gather components like Arduino Uno, LCD, 10K Potentiometer, Bread Board, 330-ohm resistor, Jumper wires
2. Connect LCD on Breadboard.
3. Connect stapler pins as a jumper to the LCD on a breadboard.
4. Connect the 330-ohm resistor with pin 15 to Vcc.
5. Connect pin 1 of the LCD to GND.
6. Connect pin 2 of LCD to +5v.
7. Connect pin 3 of the LCD to the 10K potentiometer middle pin.
8. Connect 1st pin of Pot to +5v and 2nd pin to GND.
9. Connect pin 4 of LCD to 12 on Arduino according to the code.

10. Connect pin 5 of the LCD to GND.
11. Connect pin 6 of LCD to 11 on Arduino according to the code.
12. Connect pin 11 of LCD to 5 on Arduino according to the code.
13. Connect pin 12 of LCD to 4 on Arduino according to the code.
14. Connect pin 13 of LCD to 3 on Arduino according to the code.
15. Connect pin 14 of LCD to 2 on Arduino according to the code.
16. Connect pin 15 of LCD to 330 ohms to +5v.
17. Connect pin 16 of LCD to GND.

5. Interface I2C LCD with Arduino:

- The character LCD is ideal for displaying text, numbers, and special characters. LCDs incorporate a small add-on circuit (backpack) mounted on the back of the LCD module.
- The module features a controller chip handling I2C communications and an adjustable potentiometer for changing the intensity of the LED backlight.
- An I2C LCD advantage is that wiring is straightforward, requiring only two data pins to control the LCD.

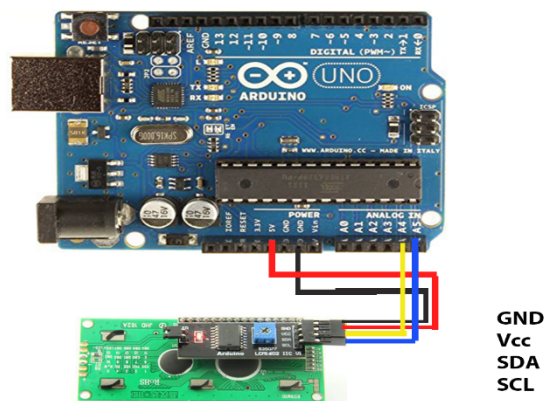
Connection Details:

I2C LCD Pin	Arduino UNO Pin
GND	GND
VCC	5V
SDA	A4
SCL	A5

Connecting the Arduino UNO to the I2C interface of the LCD requires only four connections. The connections include two for power and two for data.

Connection Diagram:

I2C Interface Diagram



Coding:

```
1 // -----
2 // Hello World! Demo - Updated for PCBoard.ca 2020-08-20
3 //
4 // This sketch displays a simple message to the LCD screen
5 // -----
6
7 #include <LiquidCrystal_I2C.h> // Driver Library for the LCD Module
8
9 // Wiring: Connect SDA pin to A4 and SCL pin to A5
10 // Connects to LCD via I2C, at address 0x27 (A0-A2 not jumpered)
11
12 LiquidCrystal_I2C lcd = LiquidCrystal_I2C(0x27,16,2); // Adjust to (0x27,20,4) for 20x4 LCD
13
14 void setup() {
15   // Initiate the LCD and turn on the backlight
16   lcd.init();           // Initiate the LCD module
17   lcd.backlight();      // Turn on the backlight
18 }
19
20 void loop() {
21   // Print 'Hello World!' on the first line of the LCD
22   lcd.setCursor(0, 0);   // Set the cursor on the first column and first row.
23   lcd.print("Hello World!"); // Print the string "Hello World!"
24
25   lcd.setCursor(1, 1);   //Set cursor to 2nd column and 2nd row (counting starts at 0)
26   lcd.print("www.PCBoard.ca");
27 }
28
```

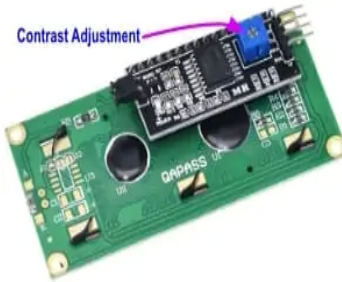
Library Required: LiquidCrystal_I2C library

Use this Link:

<https://github.com/fdebrabander/Arduino-LiquidCrystal-I2C-library>

First Step: Contrast Adjustment

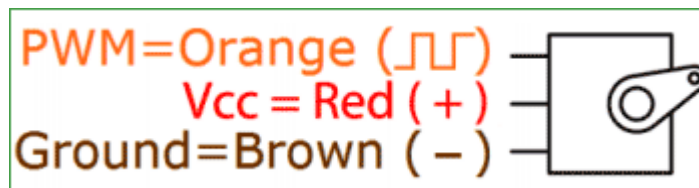
- Once you have the four connections to your LCD made, you can power your Arduino, which will provide power to the LCD.
- The LCD has an adjustment that needs to be approximately set to allow you to see characters on display.
- Located on the back of the LCD screen is the I2C interface board and an adjustable potentiometer on the interface. This adjustment is made with a small screwdriver.
- You will adjust the potentiometer until a series of rectangles appear – this will allow you to see your programming results.

	<ul style="list-style-type: none"> After your first program is loaded and you can see the output on the LCD, you can adjust the contrast control for optimal viewing.  <p>5. Hands-on Activity: Understand the Code and Print your name on the LCD Display. Refer to Activity Sheet.</p> <p>6. Doubt Clarification /Q & A Session- Do the Learning check using these Prompt Questions:</p> <ul style="list-style-type: none"> What is the Purpose of LED+ & LED- Pins of 16X2 LCD Display? Identify the Pin to do Contrast Control of the 16X2 LCD Display. What is the purpose of Using a Potentiometer in the Activity? <p>7. Take-Home Assignment: Write a code to Displaying moving(scrolling) text on 16x2 LCD with Arduino Uno.</p> <p>Refer to this for Coding: https://www.engineersgarage.com/moving-text-on-16x2-lcd-with-arduino/</p> <p>References Watch This video: https://www.youtube.com/watch?v=dZZynJLmTn8 :https://www.youtube.com/watch?v=np757xxsl0o</p>
Day 4: Motor Interface -	<p>Lesson Aims</p> <ol style="list-style-type: none"> 1. Explain the basics of Servo Motor 2. Describe the Servo motor working mechanism 3. Integrate the Servo Motor with Arduino



Servo Motor	<p>4. Demonstrate the Controlling of Servo Motor</p> <p>Activity Title:</p> <ol style="list-style-type: none"> 1. Ice-breaking on Servo Motor(10 Mins) 2. Introduction to Servo Motor(15 Mins) 3. Servo motor working mechanism(20 Mins) 4. Interfacing Servo Motor with Arduino(25 Mins) 5. Controlling Servo Motor -DemoActivity (30 Mins) 6. Doubt Clarification / Q & A Session (10 Mins) 7. Instructions for Taking Home Assignment (10 Mins) (Individual) <p>Activity Description:</p> <ol style="list-style-type: none"> 1. Ice-breaking on Servo Motor: At the beginning, show them the video having Significance & Real-Life Applications of Servo Motor like RC Helicopter and RC Car. Also, show any Robotic application video and explain How the Entire Robotics Domain Depends on Servo Motors. Video: Arduino Car Parking System 2. Introduction to Servo Motor: A Servomotor is a rotary or linear actuator that allows for precise control of angular or linear position, velocity, and acceleration. Servo Motor Working Mechanism consists of three parts: <ol style="list-style-type: none"> 1. Motor Controller 2. Output sensor 3. Feedback system <p>Servo Works with a closed-loop system that uses a positive feedback system to control motion.</p> <p>So, the main task of servomechanism is to maintain the output of a system at the desired value in the presence of noises.</p> 3. Interfacing Servo Motor with Arduino: Interfacing hobby Servo motors like SG90 servo motor with MCU is very easy. Servos have three wires coming out of them. Out of which, two will be used
--------------------	--

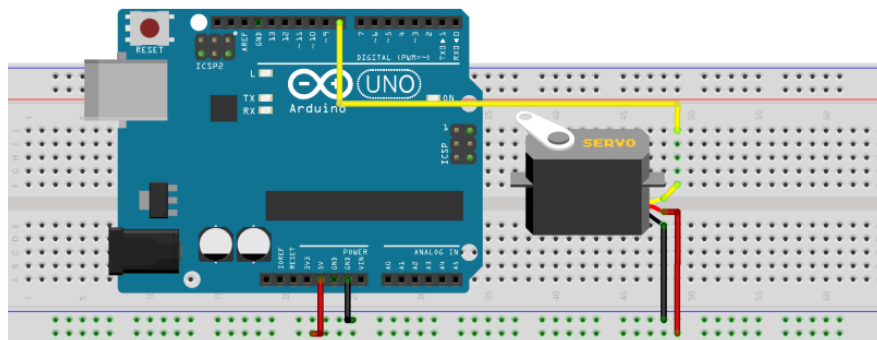
for Supply (positive and negative) and one for the signal to be sent from the MCU. An SG90 Servo Motor rotates from 0 to 180 degrees at each position of 90 degrees which is most used for RC cars, humanoid bots etc. The picture of SG90 is shown below:



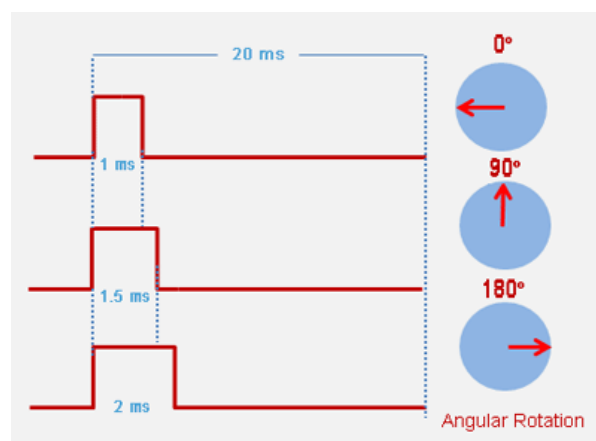
The color coding of your servo motor might differ; hence check for your respective datasheet.

All servo motors work directly with your +5V supply rails, but we have to be careful about the amount of current the motor would consume if you are planning to use more than two servo motors, a proper servo shield should be designed.

4. Controlling Servo Motor - Activity:



- The Servo motor is controlled by PWM (Pulse width Modulation), provided by the control wires. There is a minimum pulse, a maximum pulse, and a repetition rate. The Servo motor can turn 90 degrees from either direction from its neutral position.
- The servo motor expects to see a pulse every 20 milliseconds (ms); the pulse length will determine how far the motor turns. For example, a 1.5ms pulse will make the motor turn to the 90° position, such as if the pulse is shorter than 1.5ms shaft moves to 0°, and if it is longer than 1.5ms, then it will turn the servo to 180°.



Servo Motor Timing Diagram

Coding:

```

TO MOVE 90 DEGREE SERVO MOTOR
//LIBRARY CAN BE ADDED FOR SERVO MOTOR
#include<Servo.h>
Servo sl;
// Determine Which Pin of Arduino Is Connected
With Servo Motor
void setup()
{
  sl.attach(9);
}
void loop ()
{
  sl.write(90);
  delay(1000);

```

```
}  
Forward-Reverse Motion of Servo Motor  
#include<Servo.h>  
Servo sl;  
void setup()  
{  
sl.attach(9);  
}  
void loop ()  
{  
sl.write(0);  
delay(1000);  
sl.write(180);  
delay(1000);  
sl.write(0);  
}  
Smooth Forward & Reverse motion using for loop in servo motor  
#include<Servo.h>  
Servo s1;  
void setup()  
{  
s1.attach(9);  
}  
void loop()  
{  
for (int i=0; i<= 180; i+= 1)  
{  
s1.write(i);  
delay(15);  
}  
for (int i=180; i>=0; i-= 1)  
{  
s1.write(i);  
delay(15);  
}  
}  
Servo motor Control using potentiometer  
#include<Servo.h>  
Servo s1;  
int pot=A0;
```




	<pre>int x; int value; void setup() { pinMode(A0,INPUT); s1.attach(9); } void loop() { x=analogRead(pot); value=map(x,0,1023,0,180); s1.write(value); }</pre> <p>The Servo motor can be rotated from 0 to 180 degrees; after these, take a demo on the servo motor and show all turns from 0° to 180°.</p> <p>5. Doubt Clarification /Q & A Session- Do the Learning check using these Prompt Questions:</p> <ul style="list-style-type: none"> • What are the Advantages of the Servo motor over other motors? • Explain the Color Codes of SG90 Servo Motors. • The Servo motor can be rotated fromto..... degrees <p>6. Take-Home Assignment: Write a code to rotate the servo motor 90 to 180 degrees. Refer to the Above Coding And change the Parameters.</p> <p>References YouTube video https://youtu.be/LXURLvga8bQ how to control servo motor with arduino in tinkercad : Tinkercad</p>
Day 5: DC Motor and Motor Driver	<p>Lesson Aims:</p> <ol style="list-style-type: none"> 1. Explain the Basics of DC motor 2. Discuss the working of L298 Motor Driver 3. Describe the DC Motor Working Mechanism 4. Integrate the DC Motor with Arduino using Motor Driver <p>Activity Title:</p> <ol style="list-style-type: none"> 1. Ice-breaking on DC Motor (10 Mins) 2. Understanding DC motor (20 Mins) 3. Understanding motor driver (20 Mins)

4. Interfacing DC Motor with Arduino using a motor driver (20 Mins)
5. Demo Activity (30 Mins)
6. Doubt Clarification / Q & A Session (10 Mins)
7. Instructions for Taking Home Assignment (10 Mins) (Individual)

Activity Description:

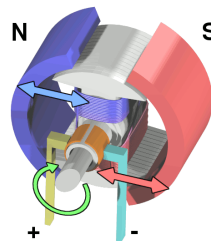
1. Ice-breaking on DC Motor:

At the beginning, show them the video having Significance & Real-Life Applications of DC Motor like DVD Player, Toy Car. Also, Show the video of any Conveyor System driven by DC Motor and India's Initiatives on Green Mobility "Electric Vehicle" having BLDC (Brushless DC) Motor.

Video Link: <https://www.youtube.com/watch?v=qnhReMd7dAk>

2. Understanding DC motor:

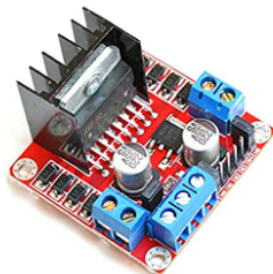
A DC motor is any of a class of rotary electrical motors that converts direct current (DC) electrical energy into mechanical energy. The most common types rely on the forces produced by induced magnetic fields due to flowing current in the coil. Nearly all types of DC motors have some internal mechanism, either electromechanical or electronic, to periodically change the current direction in a part of the motor.



DC Motor Working Mechanism

3. Understanding motor driver:

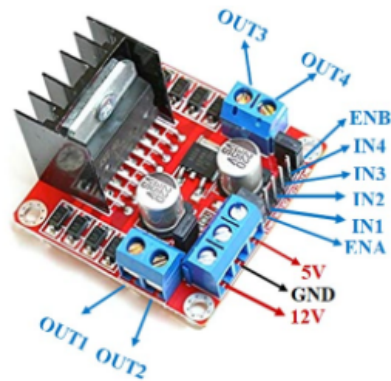
The L298N Motor Driver module consists of an L298 Motor Driver IC, 78M05 Voltage Regulator, resistors, capacitor, Power LED, 5V jumper in an integrated circuit.



L298N Motor Driver Module

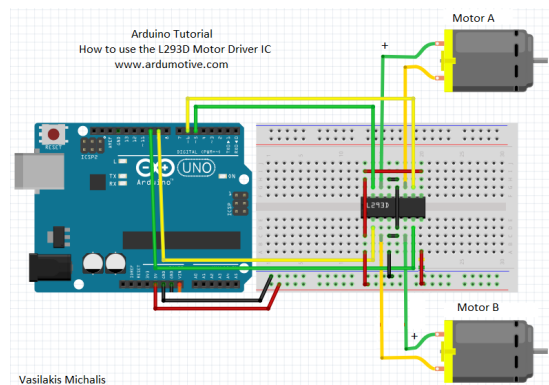
L298 Module Pinout Configuration

Pin Name	Description
IN1 & IN2	Motor A input pins. Used to control the spinning direction of Motor A
IN3 & IN4	Motor B input pins. Used to control the spinning direction of Motor B
ENA	Enables PWM signal for Motor A
ENB	Enables PWM signal for Motor B
OUT1 & OUT2	Output pins of Motor A
OUT3 & OUT4	Output pins of Motor B
12V	12V input from DC power Source
5V	Supplies power for the switching logic circuitry inside L298N IC
GND	Ground pin



L298N Motor Driver Module Pinout

2. Interfacing DC Motor with Arduino using a motor driver:



Warning – Do not drive the motor directly from Arduino board pins. This may damage the board. Use a driver Circuit or an IC.

3. Demo Activity:

Write the code to Control the Speed of DC motor and Explain the Procedure.

```
int motorPin = 9;

void setup() {
  pinMode(motorPin, OUTPUT);
  Serial.begin(9600);
  while (!Serial);
}
```

```
Serial.println("Speed 0 to 255");  
}  
  
void loop() {  
  if (Serial.available()) {  
    int speed = Serial.parseInt();  
    if (speed >= 0 && speed <= 255) {  
      analogWrite(motorPin, speed);  
    }  
  }  
}
```

4. Doubt Clarification /Q & A Session- Do the Learning check using these Prompt Questions:


- List any 2 differences between the DC motor & Servo Motor?
- How to interface the DC motor and Arduino using a Motor driver.
- Explain the Safety Precautions to be followed while doing this Activity.

5. Take-Home Assignment:

Write a code to rotate the DC motor clockwise and anticlockwise.

References

YouTube video: <https://www.youtube.com/watch?v=dyZolgNOomk>

:  Motor Driver Module With Arduino using Tinkercad (IOT)