CONTROL SYSTEMS

PROJECT REPORT

TEAM MEMBERS:

B.VISHWA TEJA(17BEC0401)

V.YESWANTH(17BEC0179)

DC MOTOR CONTROL USING MATLAB(SIMULINK) AND ARDUINO

SUBMITTED TO :

PROF.RAJESH.R SIR

COMPONENTS REQUIRED:

- MATLAB(SIMULINK) installed Laptop (Preference: R2016a or above versions)
- Arduino UNO
- DC Motor
- L293D- motor driver IC
- Breadboard

ABSTRACT:

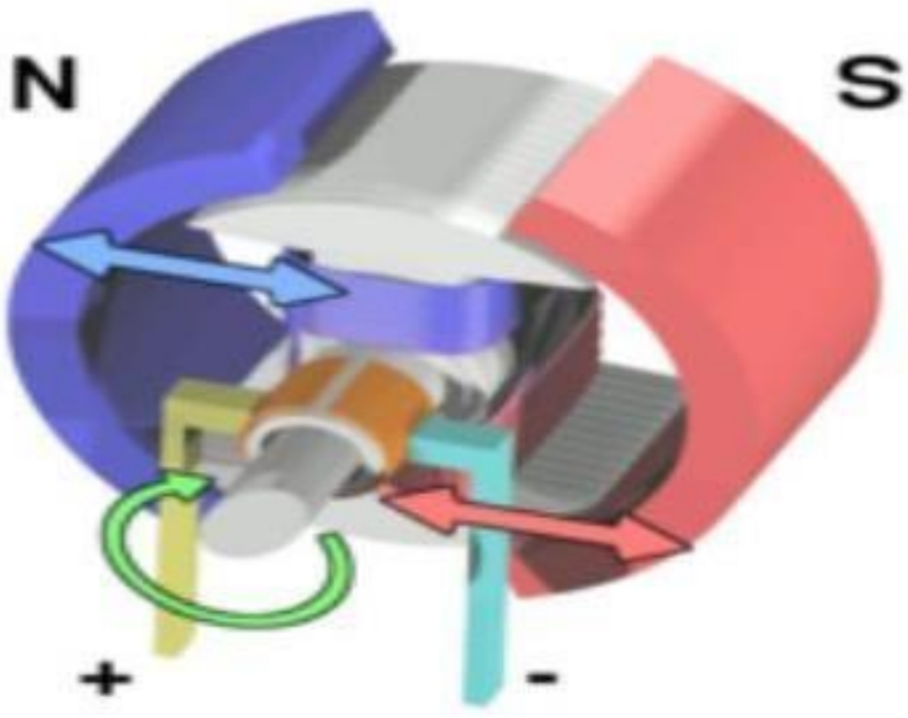
This project is based on dc motor which rotates in required direction(clockwise or anti clockwise) and also the speed of the motor rotating will be controlled according to the requirement of the instrument with specified matlab(simulink) using aurdino. Most of industrial machines and ac which requires both speed and direction to be controlled are managed by sing this project.

OBJECTIVE OF THE PROJECT:

It features

- Controlling the dc motor by using matlab and aurdino uno
- To control the dc motor in the required direction and parallely speed of the motor is also controlled by coding manually to the extent required.
- To find the application of this circuit in machines and automatic door systems.

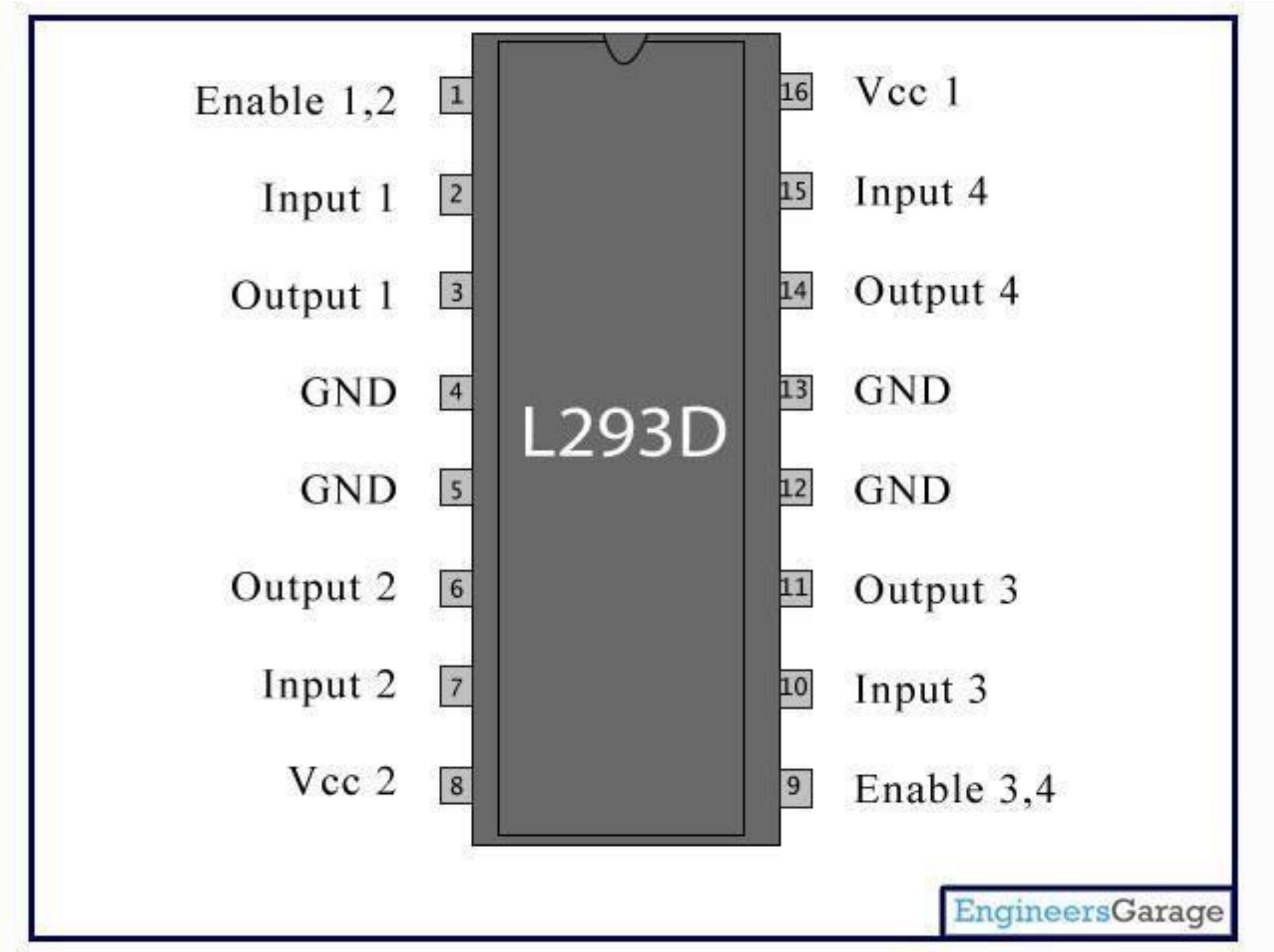
DC MOTOR WORKING PRINCIPLE:



WORKING PRINCIPLE:

- there are two types of dc motors they are brushed or brushless ,we are discussing brushed motor here.
- Workings of a brushed electric motor with a two-pole rotor (armature) and permanent magnet stator. "N" and "S" designate polarities on the inside axis faces of the magnets; the outside faces have opposite polarities. The + and - signs show where the DC current is applied to the commutator which supplies current to the armature coils

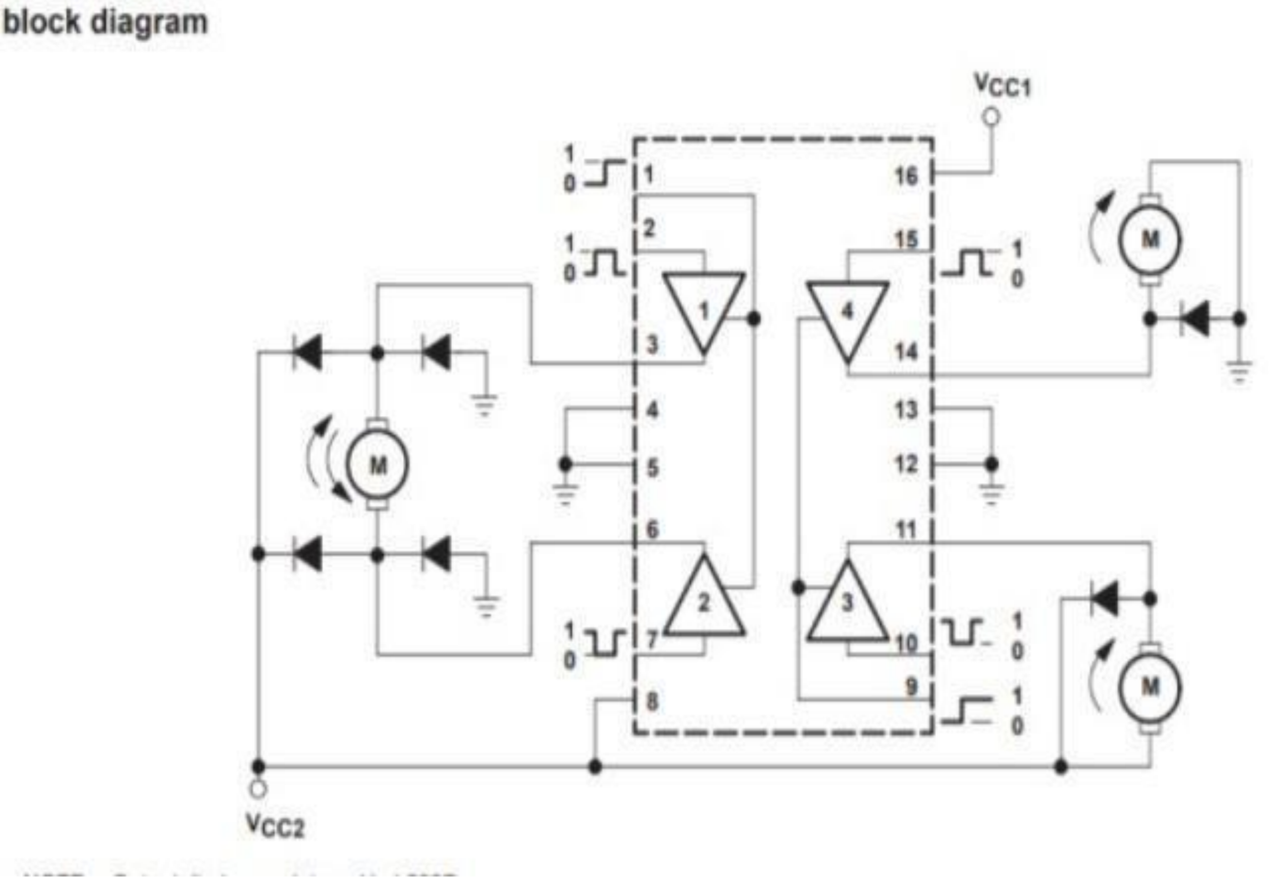
L293D MOTOR DRIVER IC:



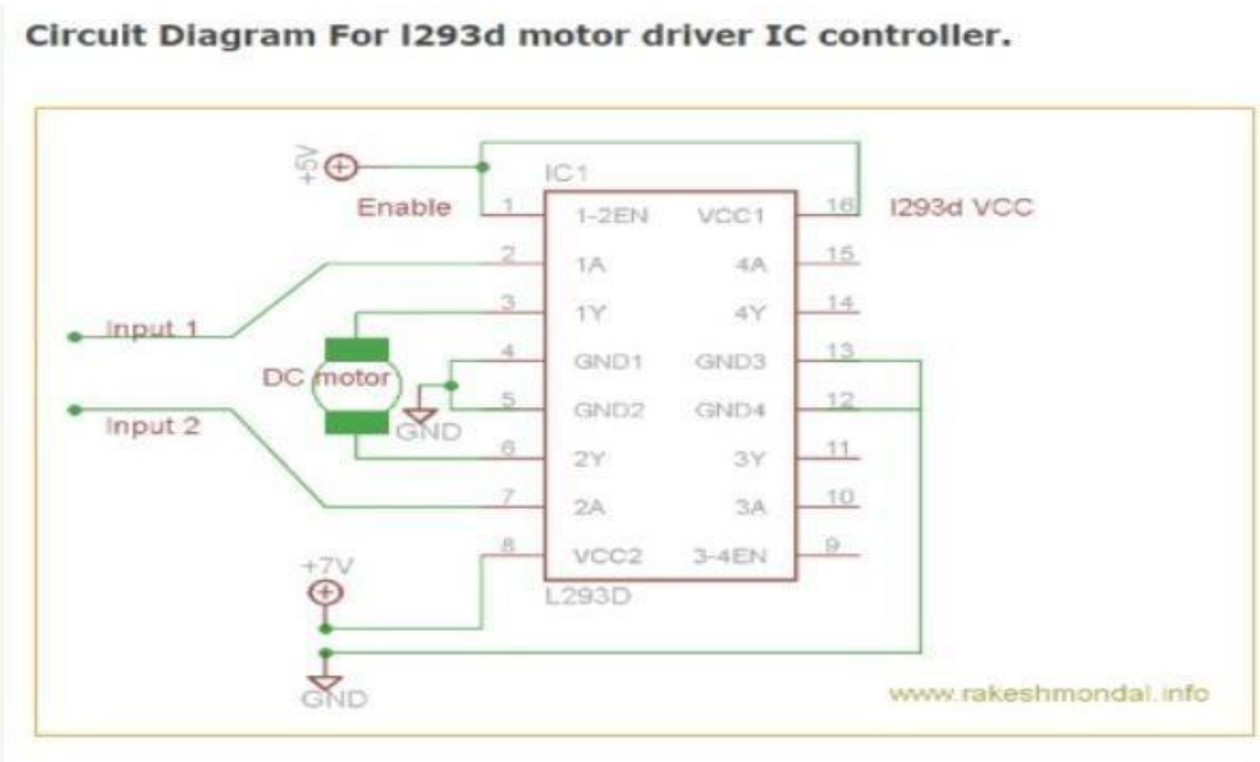
PIN CONFIGURATION FOR CONTROLLING DIRECTION:

- **Pin 2 = Logic 1** and **Pin 7 = Logic 0** | Clockwise Direction
- **Pin 2 = Logic 0** and **Pin 7 = Logic 1** | Anticlockwise Direction
- **Pin 2 = Logic 0** and **Pin 7 = Logic 0** | Idle [No rotation] [Hi-Impedance state]
- **Pin 2 = Logic 1** and **Pin 7 = Logic 1** | Idle [No rotation]

BLOCK DIAGRAM OF BASIC L293D:



CIRCUIT DESIGN OF L293D:



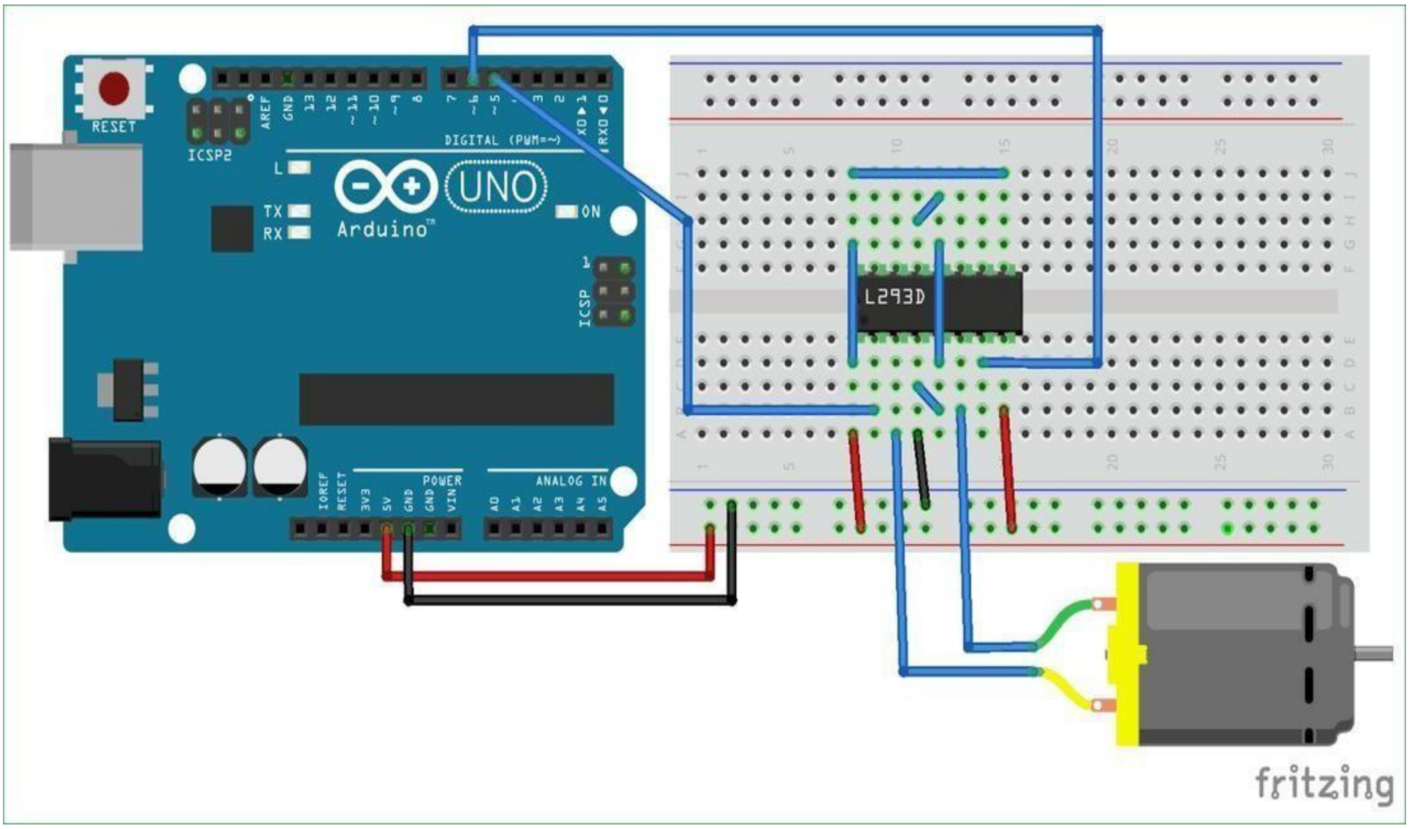
WORKING PRINCIPLE:

- L293D is a dual H-bridge motor driver integrated circuit (IC). Motor drivers act as current amplifiers since they take a low-current control signal and provide a higher-current signal. This higher current signal is used to drive the motors.
- L293D contains two inbuilt H-bridge driver circuits. In its common mode of operation, two DC motors can be driven simultaneously, both in forward and reverse direction. The motor operations of two motors can be controlled by input logic at pins 2 & 7 and 10 & 15. Input logic 00 or 11 will stop the corresponding motor. Logic 01 and 10 will rotate it in clockwise and anticlockwise directions, respectively.

APPLICATION OF THIS PROJECT IN SEVERAL WAYS:

- By using this controller we can control the direction and speed of multiple fans and ac's at a time by adjusting the code in matlab(simulink) and arduino.
- Reduces the complexity of industrial machines
- For example:
- We can control the direction and speed of the outlet of AC so that if the temperature of the room is high, then by increasing the speed of the fan the temperature can be get down easily by adjusting the software .

BLOCK DIAGRAM OF THE CIRCUIT:



METHODOLOGY:

- first connect the circuit using arduino uno
- then write the code in the matlab and draw the structures required for the operation of the DC motor like clockwise ,anticlockwise,speed controlling sliders so as to control the dc motor
- now connect the hardware and the hardware to laptop
- then vary the speed by using a slider and direction using bars.

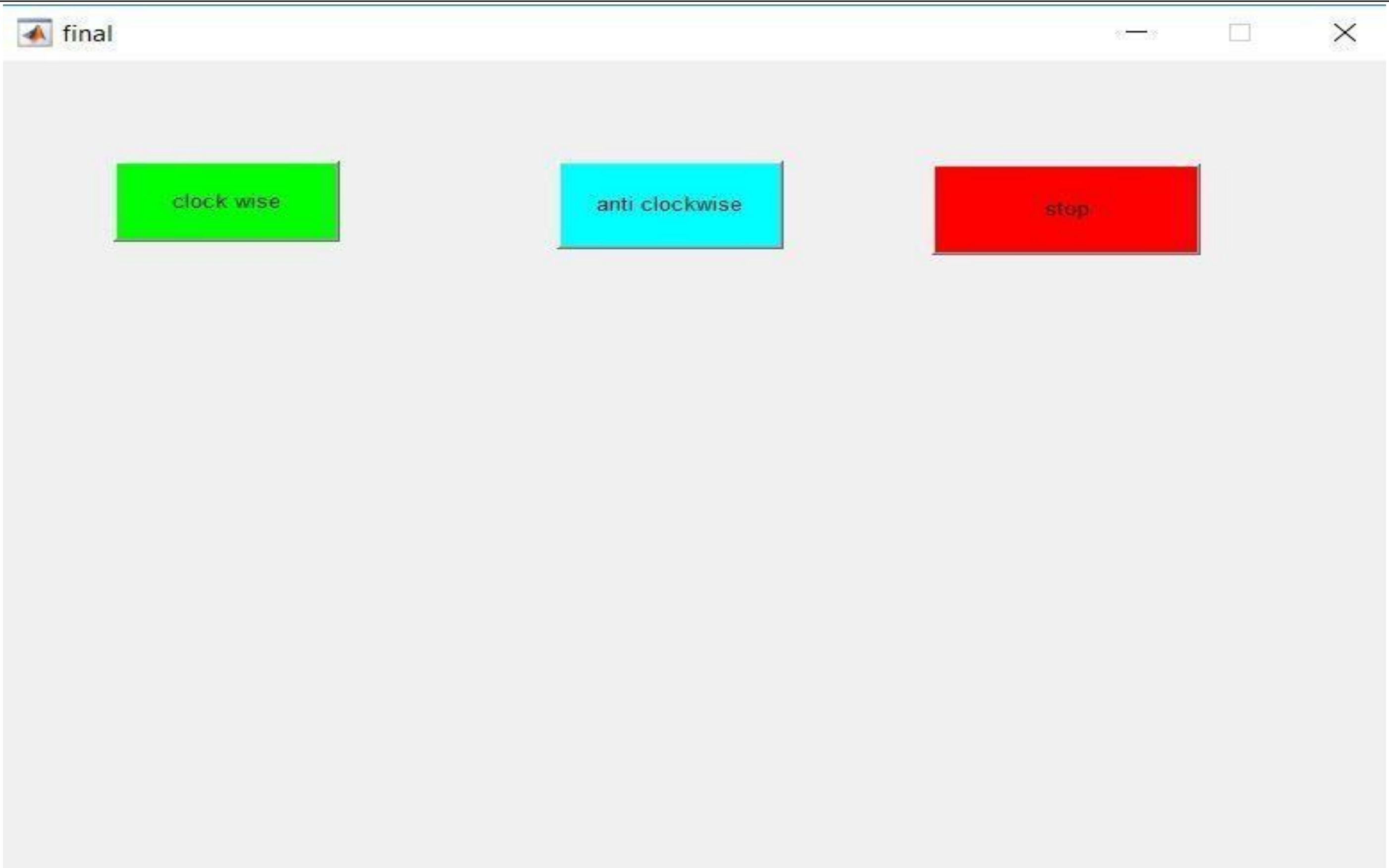
INPUT IN MATLAB:

MATLAB CODE:

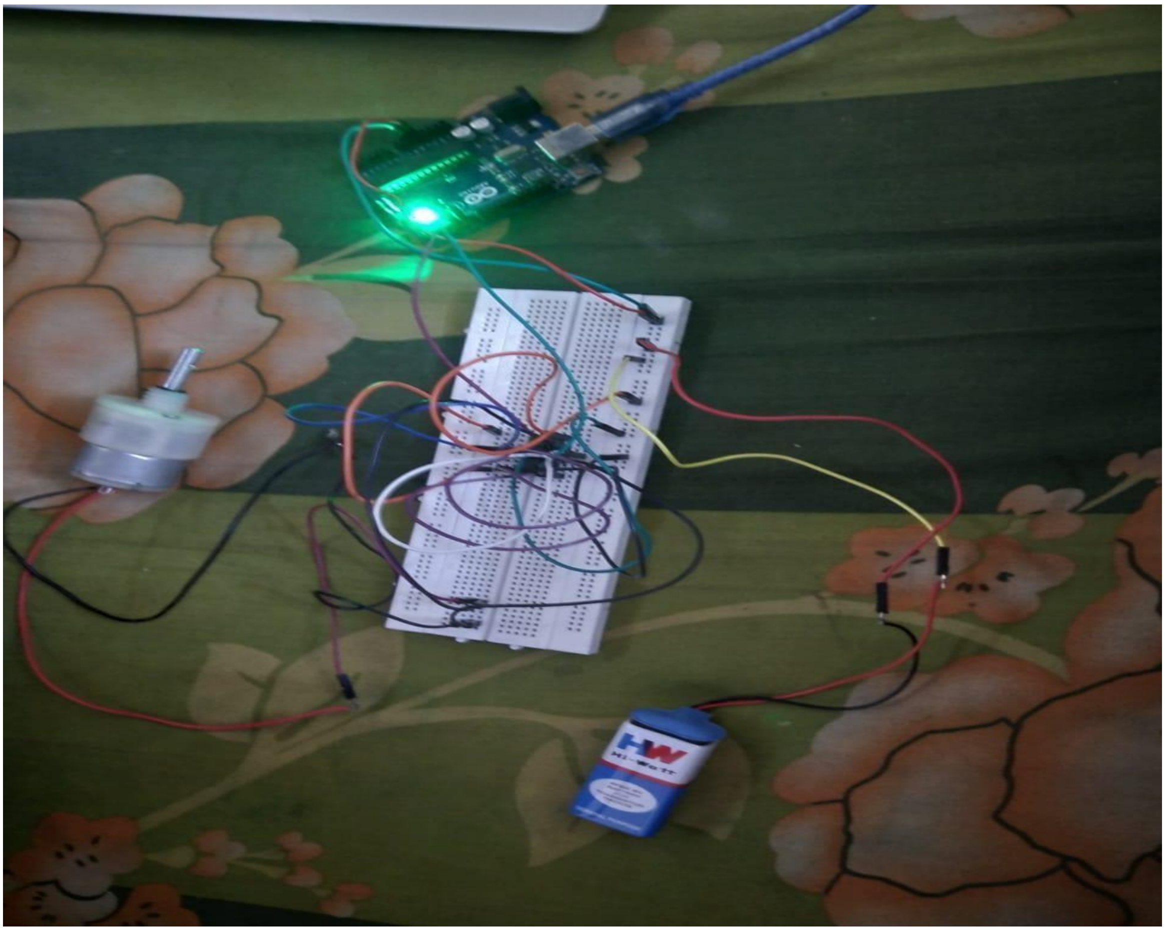
```
function varargout = gui2(varargin)
% GUI2 MATLAB code for gui2.fig
%      GUI2, by itself, creates a new GUI2 or raises the existing
%      singleton*.
%
%      H = GUI2 returns the handle to a new GUI2 or the handle to
%      the existing singleton*.
%
%      GUI2('CALLBACK',hObject,eventData,handles,...) calls the local
%      function named CALLBACK in GUI2.M with the given input arguments.
%
%      GUI2('Property','Value',...) creates a new GUI2 or raises the
%      existing singleton*.  Starting from the left, property value pairs are
%      applied to the GUI before gui2_OpeningFcn gets called.  An
%      unrecognized property name or invalid value makes property application
%      stop.  All inputs are passed to gui2_OpeningFcn via varargin.
%
%      *See GUI Options on GUIDE's Tools menu.  Choose "GUI allows only one
%      instance to run (singleton)".
%
% See also: GUIDE, GUIDATA, GUIHANDLES

% Edit the above text to modify the response to help gui2

% Last Modified by GUIDE v2.5 02-Nov-2019 20:42:36

% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @gui2_OpeningFcn, ...
                   'gui_OutputFcn',  @gui2_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT


% --- Executes just before gui2 is made visible.
function gui2_OpeningFcn(hObject, eventdata, handles, varargin)
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to gui2 (see VARARGIN)

% Choose default command line output for gui2
handles.output = hObject;
```

```
% Update handles structure
guidata(hObject, handles);

% UIWAIT makes gui2 wait for user response (see UIRESUME)
% uiwait(handles.figure1);


% --- Outputs from this function are returned to the command line.
function varargout = gui2_OutputFcn(hObject, eventdata, handles)
% varargout  cell array for returning output args (see VARARGOUT);
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Get default command line output from handles structure
varargout{1} = handles.output;
clear all;
global a;
a = arduino();


% --- Executes on button press in pushbutton1.
function pushbutton1_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
global a;
writeDigitalPin(a, 'D5', 0);
writeDigitalPin(a, 'D6', 1);
pause(0.5);


% --- Executes on button press in pushbutton2.
function pushbutton2_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
global a;
writeDigitalPin(a, 'D5', 1);
writeDigitalPin(a, 'D6', 0);
pause(0.5);


% --- Executes on button press in pushbutton3.
function pushbutton3_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton3 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
global a;
writeDigitalPin(a, 'D5', 0);
writeDigitalPin(a, 'D6', 0);
pause(0.5);
```

OUTPUT IN SIMULINK



OUTPUT HARDWARE:



SOFTWARE MODEL OF DC MOTOR

TYPES OF TORQUE:

T_a=motor torque
T_l=load torque
T_{lost}=loss torque due to friction
$T_a=T_{lost}+T_l$
Power(pout)=T_l*w
P_{out}=5HP=3730watts
w=2*pi*N/60
w=183.16rad/s
T_l=3730/183.16
T_l=20.36

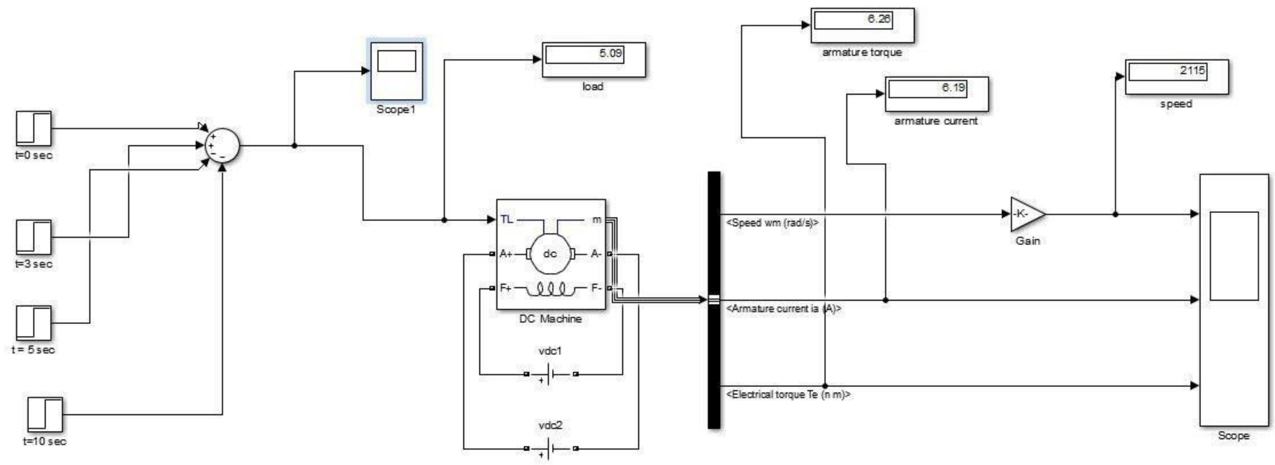
$T_{l/2}$=10.18

$T_{l/4}$=5.09

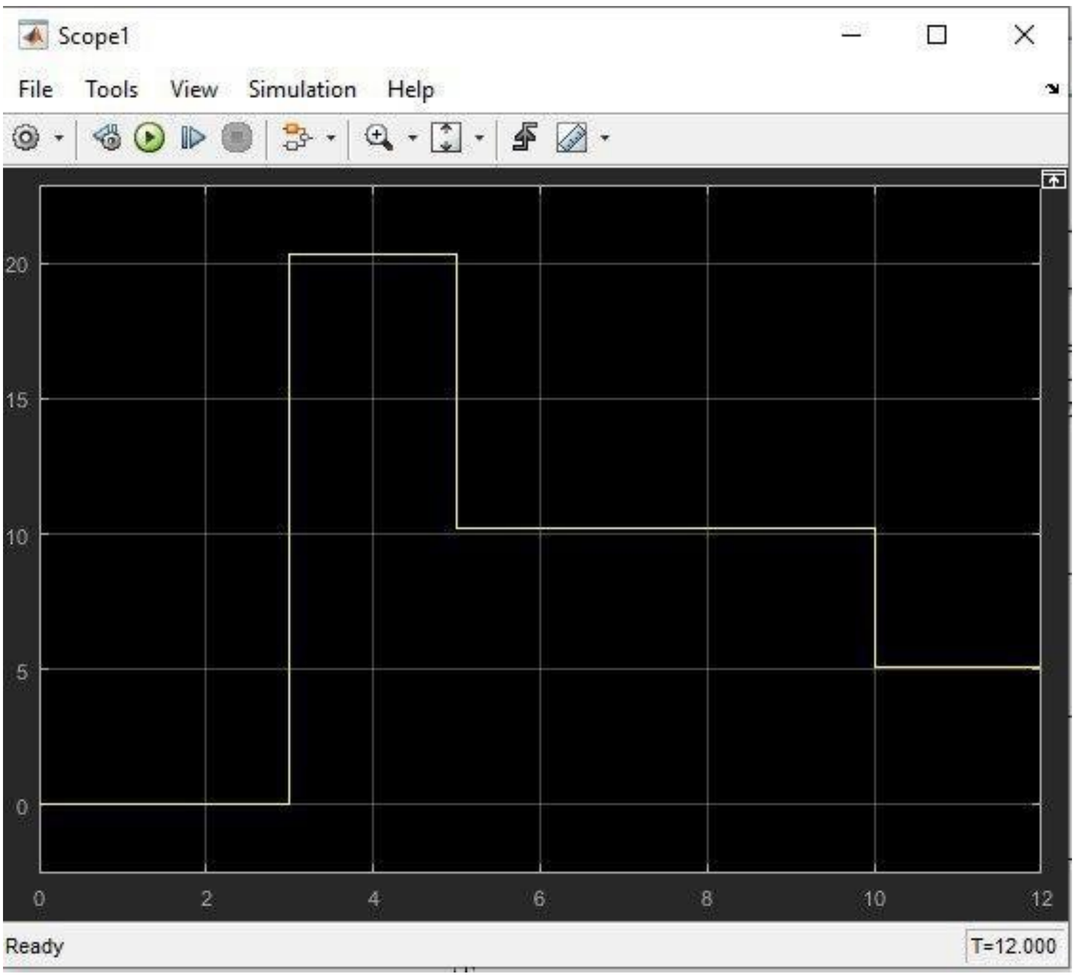
OBSERVATIONS:

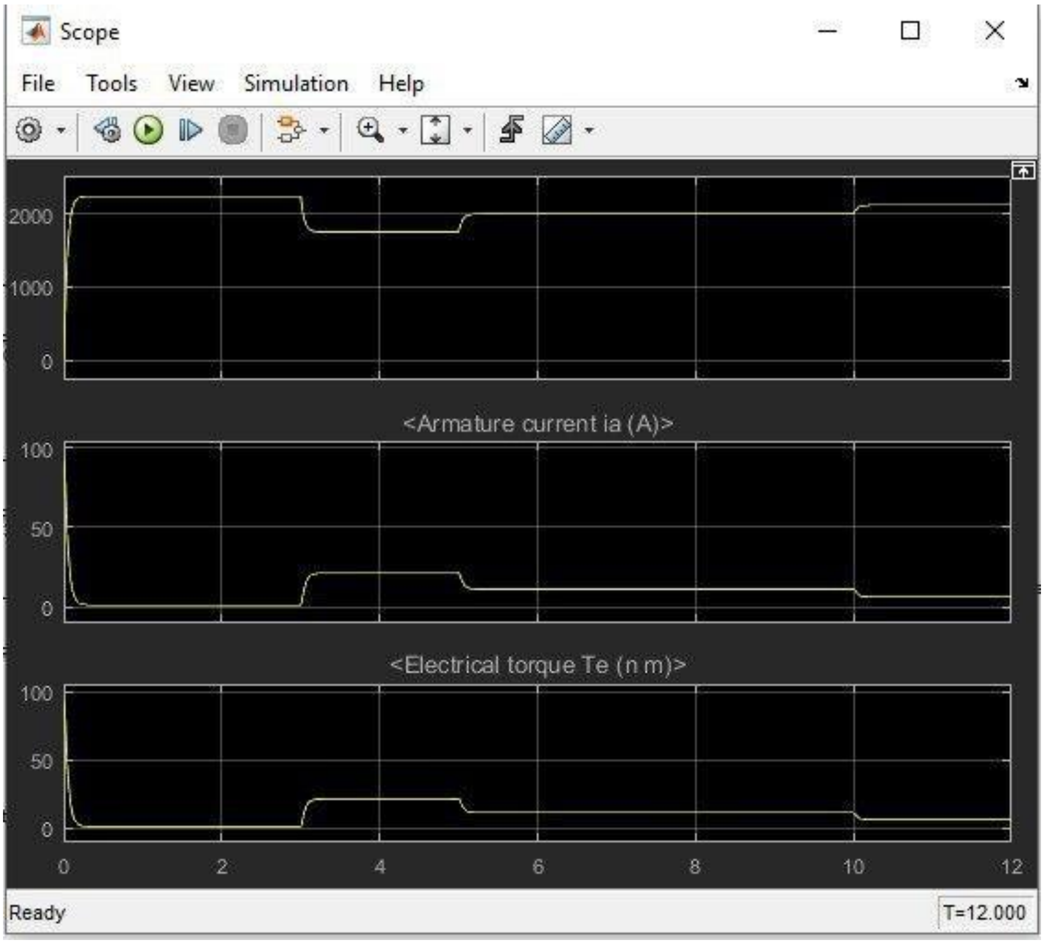
Tl value	Torque	Speed(rpm)
0	0	2150
Tl	20	1775
Tl/2	10.2	2000
Tl/4	6.26	2115

SIMULINK MODEL



Load varying with time:





This graph shows the output speed,armature current and torque of the DC motor while the load of the motor is time varied as shown in the previous figure.

- Speed increase with the reduction of road
- Current increase in the dc motor with the increase in the load value
- Torque as well increases in motor with increase in load

REFERENCES:

<u>RESEARCH PAPER 1 :</u>
simulation of linear brushless DC motor spped controlled system based on MATLAB/SIMULINK
BY:JUNYONG, DELIANG LIANG, XIANGYANG FENG

<u>RESEARCH PAPER 2:</u>
Brushless DC motor controller design using matlab applications
BY:HAYDER SALIM HAMEED

THANK YOU