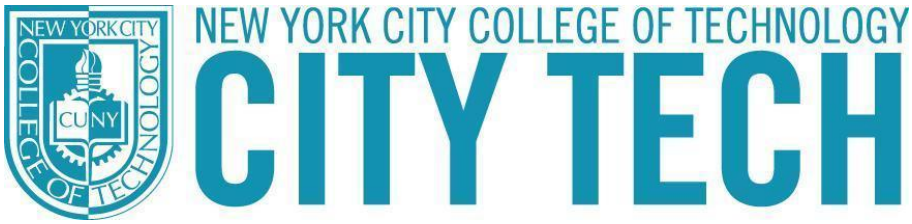


Department of Computer Engineering Technology

Project Title (Proposal): DC Motor Control Using MATLAB and Arduino

Course: CET 3625, Applied Analysis Laboratory
Section: HD23
Semester: Spring 2022
Instructor: Prof. Edward Morton, P.E.
Student(s): Touheda Khanom
Date: 05/10/2022

Table of Contents

Objective

In this project, I am going to show how to control the rotation of a DC motor using MATLAB and Arduino. The main purpose of this project is to use the L293D chip and the MATLAB Graphical User Interface to control the DC Motor. I learned how to set up hardware support for Arduino in MATLAB software and to control an Arduino using MATLAB code.

List of Material and Equipment

- Arduino Uno
- L293D- motor driver
- Breadboard
- jumper wires
- Matlab
- DC Motor
- Connecting wires / Jumper Wires

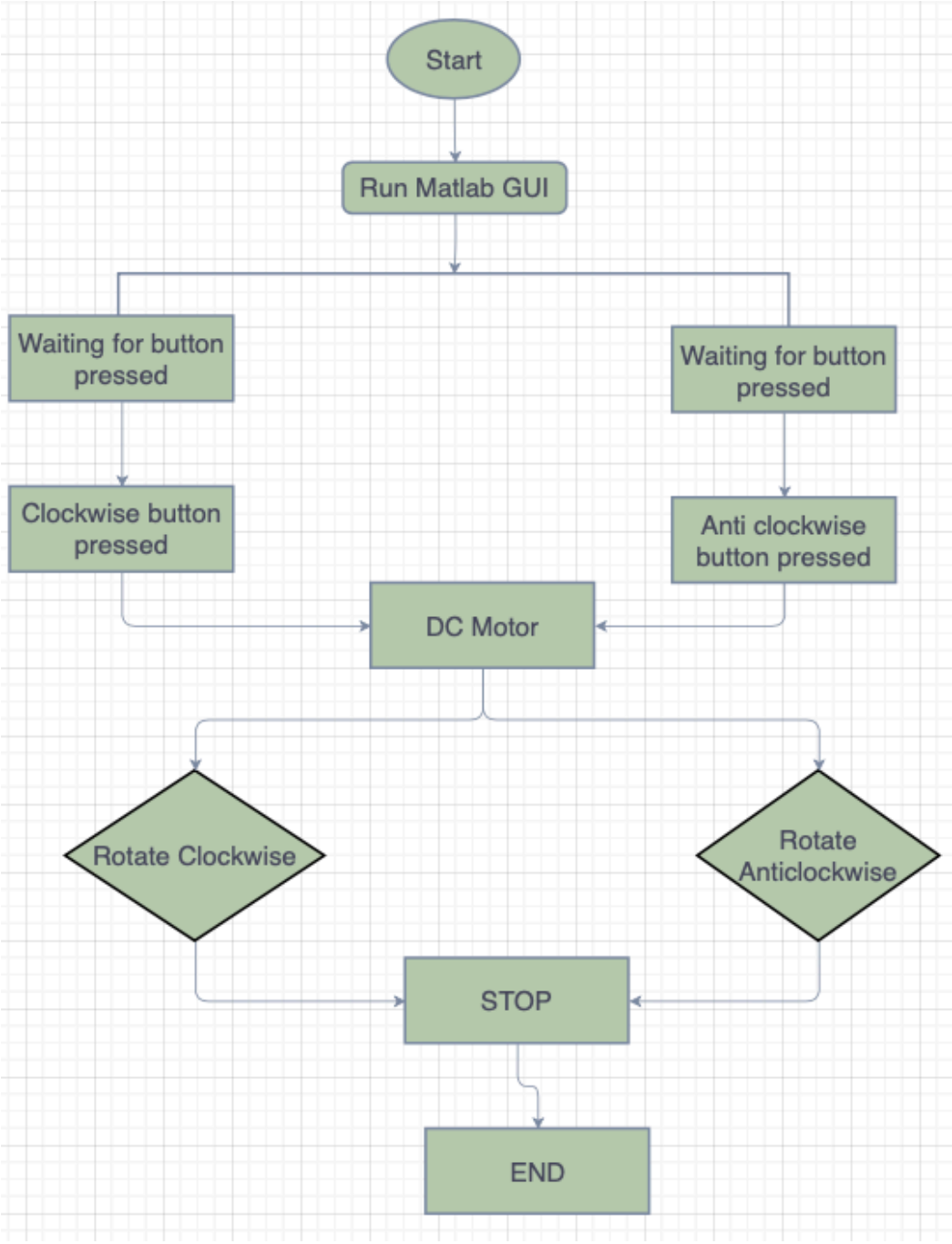
Test Plan/Procedure

1) Gather all the necessary materials for the project
2) Construct the circuit, and observe the data being collected using TinkerCad or Fritzing.
3) Download and install MATLAB
4) Start Matlab
5) In the Environment section, select Add-Ons > Get Hardware Support Packages
6) Install MATLAB Support Package for Arduino Hardware
7) Setup the hardware using L293D and Arduino
8) Connect Arduino to PC using USB
9) After finishing setup with Arduino for MATLAB, build GUI (Graphical User Interface) to control the DC motor
10) To launch the GUI, type the below command in the command window
11) Choose three push buttons for Clockwise rotation, Anti-clockwise rotation and STOP
12) After setup the hardware according to circuit diagram, just click on the run button to run the edited code in .m file
13) When everything is ready, click on the clockwise or anticlockwise button to rotate the motor. When you press Clockwise button current flow through Pin 6 to Pin 5 and motor will rotate in clockwise direction while in anti-clockwise current flows through Pin 5 to Pin 6 and motor will rotate in clockwise direction will rotate in clockwise direction. To stop the rotation of the DC motor press the STOP button.
14) Test whether the output signal changes appropriately in response to the input.
15) Troubleshoot the program if necessary, until the output is the one that you expect

to get.

System Design

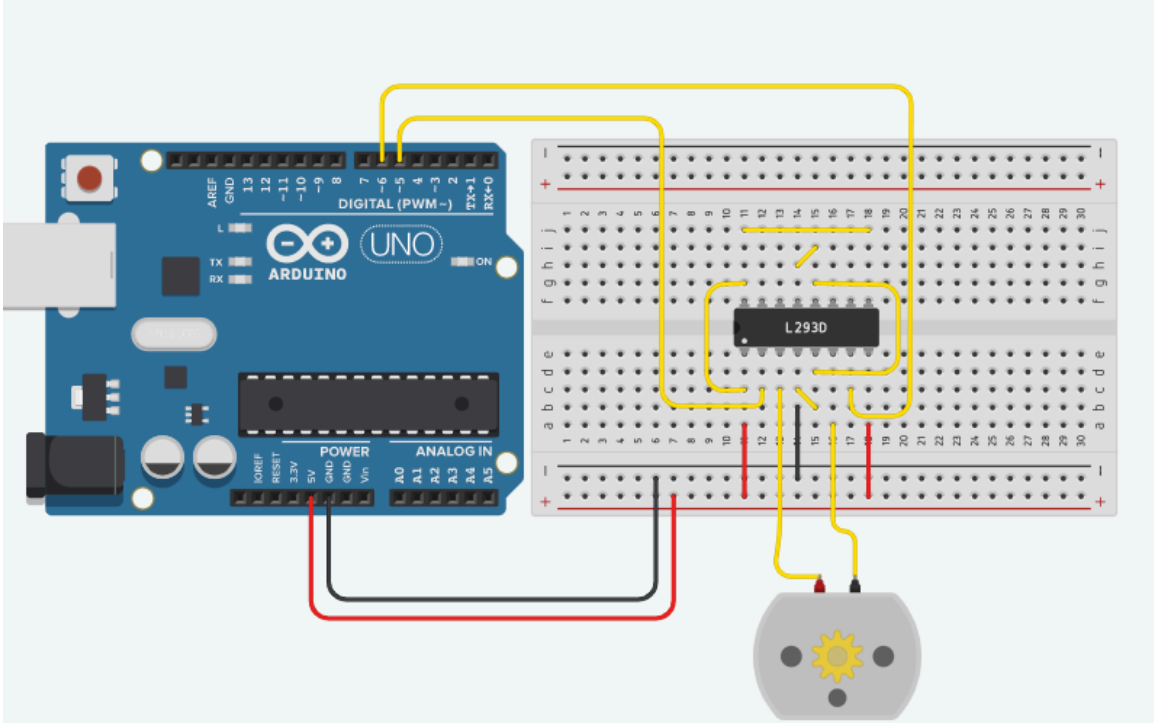
Flow Control



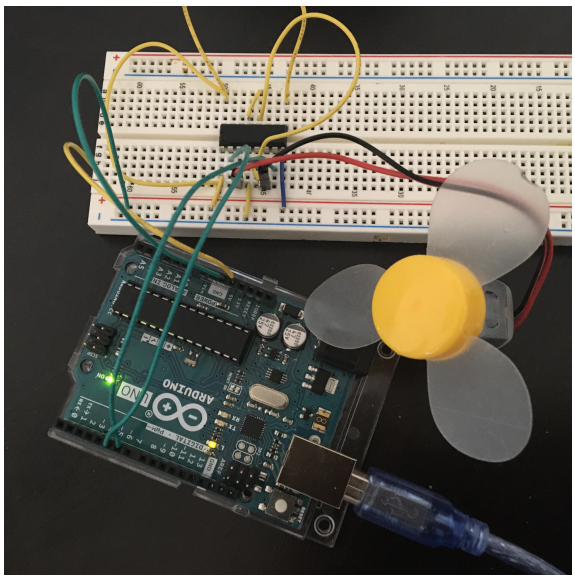
Results

Schematic Wiring

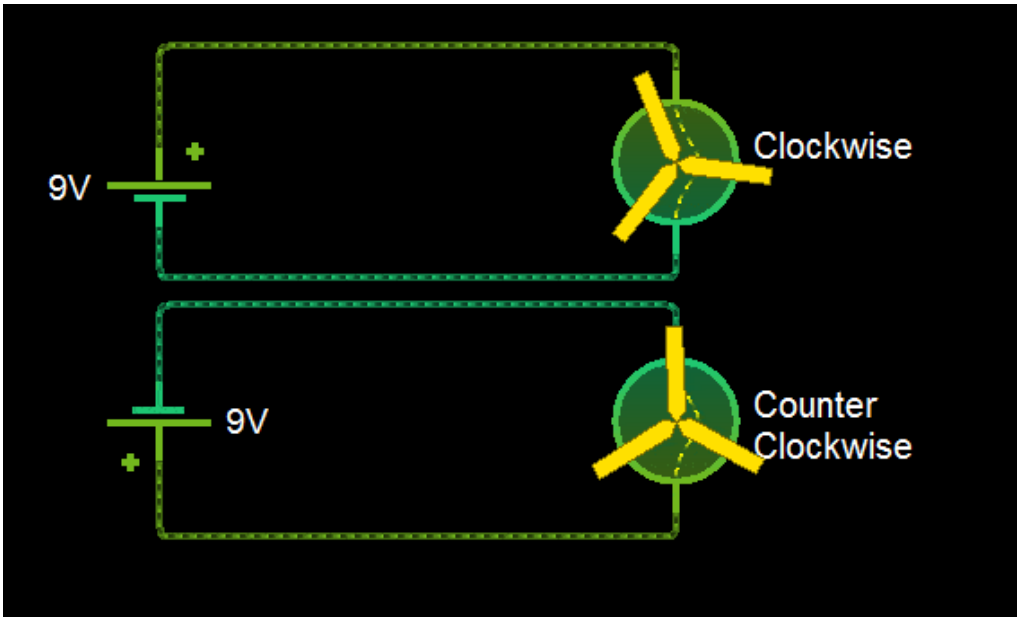
DC Motor Control Using MATLAB and Arduino



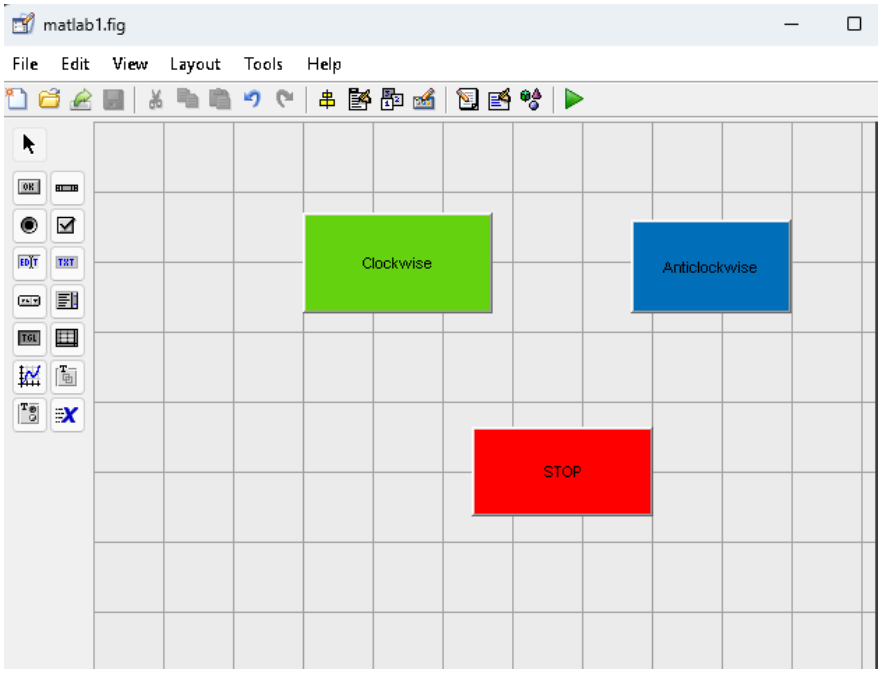
Circuit Diagram



Interfacing of Arduino with DC Motor



Created MATLAB Graphical User Interface for controlling DC Motor



Matlab Code for controlling DC Motor with Arduino

```
function varargout = matlab1(varargin)
%MATLAB1 MATLAB code file for matlab1.fig
%      MATLAB1, by itself, creates a new MATLAB1 or raises the existing
%      singleton*.
%
```

```
%      H = MATLAB1 returns the handle to a new MATLAB1 or the handle to
%      the existing singleton*.
%
%      MATLAB1('Property','Value',...) creates a new MATLAB1 using the
%      given property value pairs. Unrecognized properties are passed via
%      varargin to matlab1_OpeningFcn.  This calling syntax produces a
%      warning when there is an existing singleton*.
%
%      MATLAB1('CALLBACK') and MATLAB1('CALLBACK',hObject,...) call the
%      local function named CALLBACK in MATLAB1.M with the given input
%      arguments.
%
%      *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 matlab1

% Last Modified by GUIDE v2.5 08-May-2022 22:53:38

% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @matlab1_OpeningFcn, ...
                   'gui_OutputFcn',  @matlab1_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 matlab1 is made visible.
function matlab1_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   unrecognized PropertyName/PropertyValue pairs from the
%            command line (see VARARGIN)

% Choose default command line output for matlab1
handles.output = hObject;

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

```
% UIWAIT makes matlab1 wait for user response (see UIRESUME)
% uiwait(handles.figure1);

% --- Outputs from this function are returned to the command line.
function varargout = matlab1_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 Clockwise.
function Clockwise_Callback(hObject, eventdata, handles)
% hObject    handle to Clockwise (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 Anticlockwise.
function Anticlockwise_Callback(hObject, eventdata, handles)
% hObject    handle to Anticlockwise (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 STOP.
function STOP_Callback(hObject, eventdata, handles)
% hObject    handle to STOP (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);
```

```
function varargout = matlab1_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 Clockwise.
function Clockwise_Callback(hObject, eventdata, handles)
% hObject    handle to Clockwise (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 Anticlockwise.
function Anticlockwise_Callback(hObject, eventdata, handles)
% hObject    handle to Anticlockwise (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 STOP.
function STOP_Callback(hObject, eventdata, handles)
% hObject    handle to STOP (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);
```

Project Schedule and Activities

	Task Name	Start	Finish	Duration
1	Procure Materials	3/15/2022	3/22/2022	1 week

2	Construct Circuit	3/23/2022	4/5/2022	2 weeks
3	Develop Software	4/12/2022	4/19/2022	1 week
4	Test and Troubleshoot	4/20/2022	4/26/2022	1 week
5	Complete Circuit and Software	4/27/2022	5/3/2022	1 week
6	Presentation and Demo	5/4/2022	5/17/2022	2 week

Capabilities and Qualifications

I am knowledgeable in programming languages such as Python, C++, and Arduino IDE, and circuit design. I have done some previous projects using the Arduino hardware and the Arduino IDE programming language. I am part of a robotics research program therefore I have worked with many sensors.

References

1. Arduino Project Hub. 2022. *Temperature based Fan Speed Control*. [online] Available at:<https://create.arduino.cc/projecthub/embeddedlab786/temperature-based-fan-speed-control-945f9d> [Accessed 16 March 2022].
2. MacroAir Fans. 2022. *3 benefits of fans with automated temperature control*. [online] Available at: <https://macroairfans.com/blog /3-benefits-automated-fan-control/ #:~:text=Using%20the%20automated%20fan%20control,and%20create%20a%20cooling%20effect.> [Accessed 16 March 2022].