Stepper motor using 8086 microprocessor

Project Report

Submitted by: Group 5

Members: Mohamed Ragab, ID: 216432

Abdelraouf Naser, ID: 197724

Sara Ahmed, ID: 213802

Youssef Emad El-din Othman, ID: 208906

Module Name: Digital Design

Module code: 22COMP05C

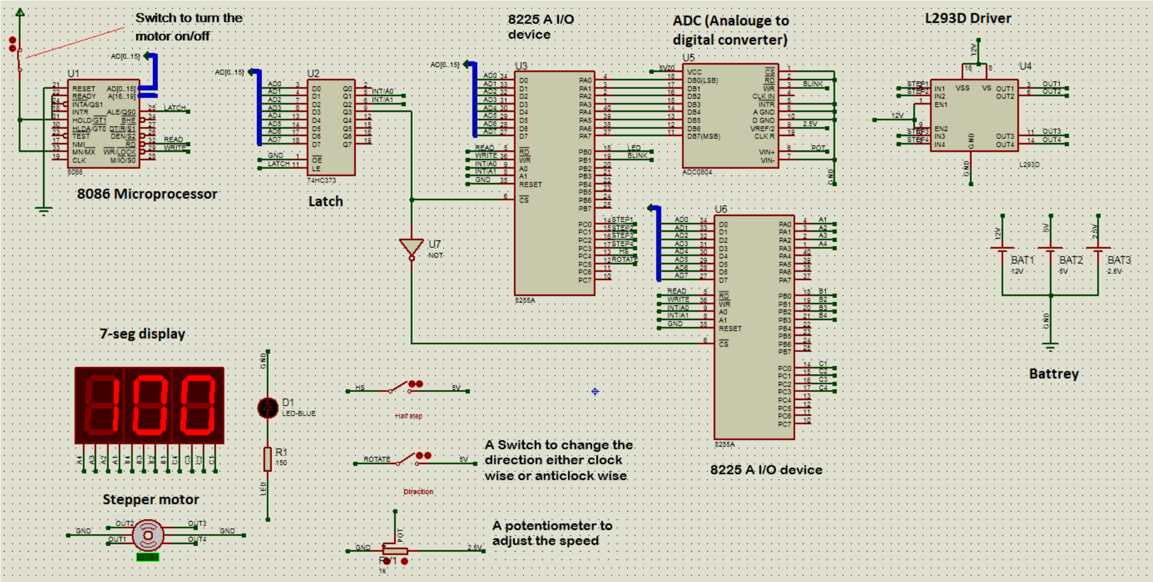
Computer Engineering Department, Degree Year 2

Submitted to: Dr. Mervat Mahmoud,

October 2022

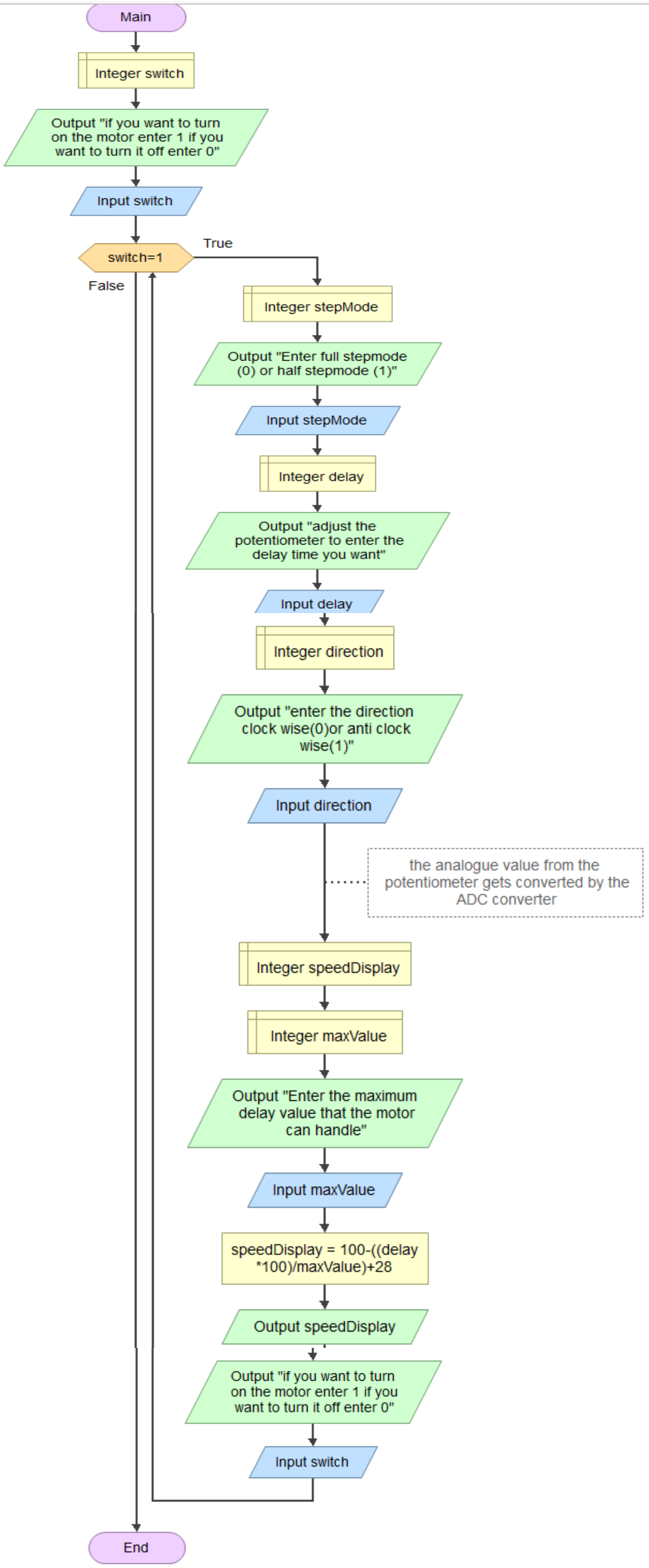
Table of Contents

Circuit Diagram



Flow chart

The basic idea is to use 8086 Microprocessor to control stepper motor direction and speed then showing the current speed value on 7-seg display.


Main
Integer switch
Output "if you want to turn on the motor enter 1 if you want to turn it off enter 0"
Input switch
switch=1
True
False
Integer stepMode
Output "Enter full stepmode (0) or half stepmode (1)"
Input stepMode
Integer delay
Output "adjust the potentiometer to enter the delay time you want"
Input delay
Integer direction
Output "enter the direction clock wise(0)or anti clock wise(1)"
Input direction
the analogue value from the potentiometer gets converted by the ADC converter
Integer speedDisplay
Integer maxValue
Output "Enter the maximum delay value that the motor can handle"
Input maxValue
speedDisplay = 100-((delay *100)/maxValue)+28
Output speedDisplay
Output "if you want to turn on the motor enter 1 if you want to turn it off enter 0"
Input switch
End

8086 Microprocessor description

- 8086 is a microprocessor that have 20 bit for address bus and 16 bit for data bus. It’s used here to control all the devices including the stepper motor, led, display
- AD--->address/data bus. AD0-AD15 for data bus, AD0-AD19 for address bus
- ALE---> address enable latch, a positive pulse generated each time the processor begins any operation. This signal indicates the availability of a valid address on the address/data lines
- RD--->used to read signal
- WR--->write signal. Used to write the data into the memory or the output device
- RESET---> used ot restart the execution. It causes the processor to immediately terminate its present activity. This signal is active high for the first 4 clock cycles to reset the microprocessor
- READY---> indicates that the device is ready to transfer data. When it's low it indicates wait state
- HOLD--->indicates that external devices are requested to access the address/data buses
- MN/MX---> minimum/ maximum. Indicates what mode the processor is to operate in. When high it operates in minimum mode

Stepper motor description

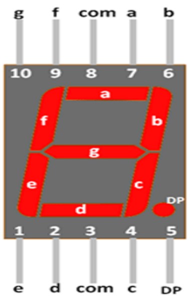
- DC motor that move in discrete steps giving the illusion of rotation
- It divides the complete rotation into number of steps, each stepper motor will have fixed step angle.
- This motor is drived by L293D Motor Driver.

  We will use unipolar stepper motor in this project which has 5 or 6 wires, this happens by tying each 2 coils from one end then there are 2 common wires as shown, unipolar can be used as bipolar as we did in the project.

The input of the motor: 12V

The output: motor is moving in discrete steps complete rotation divides into number of steps, each step has fixed angle.

7 Segment Display Pinout Configuration



Pin Number	Pin Name	Description
1	e	Controls the left bottom LED of the 7-segment display
2	d	Controls the bottom most LED of the 7-segment display
3	Com	Connected to Ground/Vcc based on type of display
4	c	Controls the right bottom LED of the 7-segment display
5	DP	Controls the decimal point LED of the 7-segment display
6	b	Controls the top right LED of the 7-segment display
7	a	Controls the top most LED of the 7-segment display
8	Com	Connected to Ground/Vcc based on type of display
9	f	Controls the top left LED of the 7-segment display
10	g	Controls the middle LED of the 7-segment display

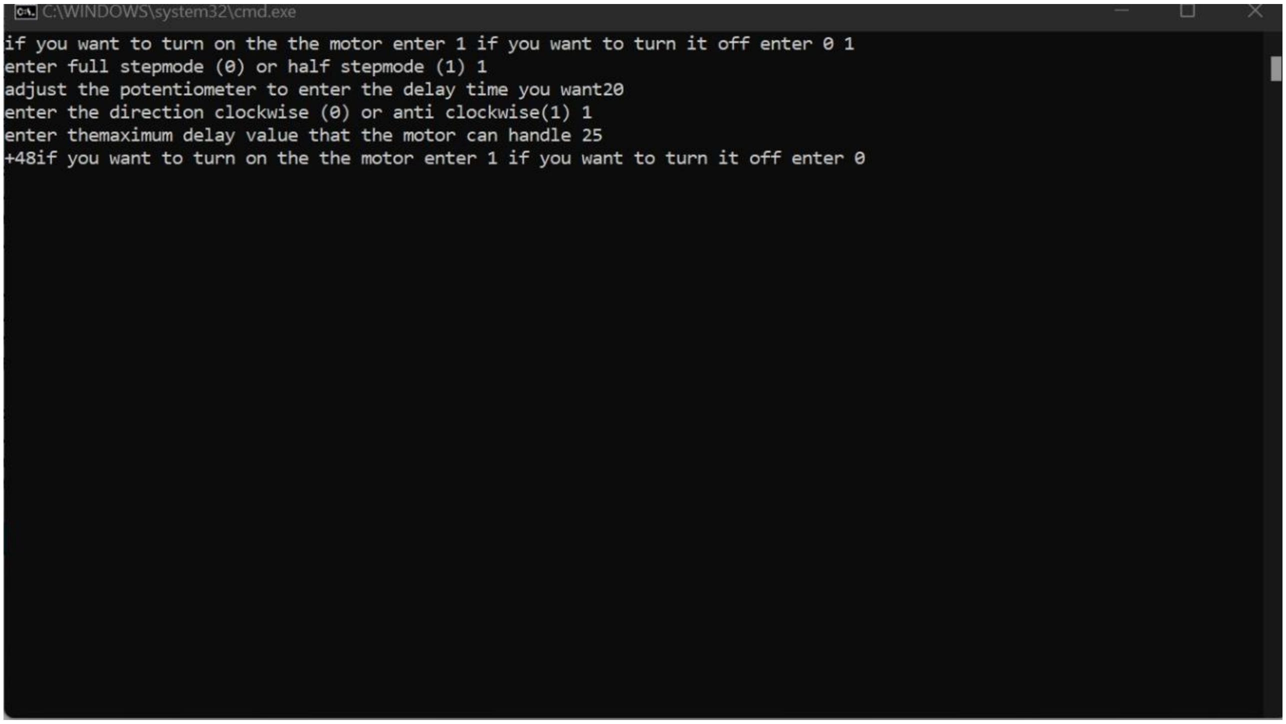
Assembly Code Using Irvine

```
TITLE MASM Template

INCLUDE Irvine32.inc

.data
```

```
myMessage BYTE "if you want to turn on the the motor enter 1 if you want to turn it off enter 0 ",0
myMessage1 BYTE "enter full stepmode (0) or half stepmode (1) ",0
myMessage2 BYTE "adjust the potentiometer to enter the delay time you want",0
myMessage3 BYTE "enter the direction clockwise (0) or anti clockwise(1) ",0
myMessage4 BYTE "enter themaximum delay value that the motor can handle ",0

stepMode DWORD ?
DelayTime DWORD ?
switch DWORD ?
direction DWORD ?
maxValue DWORD ?
speedDisplay DWORD ?
.code
main proc

mov edx, offset myMessage
call writestring
call readint
mov switch, eax

beginwhile:
.IF switch == 0
jnl endwhile
.ENDIF

mov edx, offset myMessage1
call writestring
call readint
mov stepMode, eax

mov edx, offset myMessage2
call writestring
call readint
mov DelayTime, eax

mov edx, offset myMessage3
call writestring
call readint
mov direction, eax

mov edx, offset myMessage4
call writestring
call readint
mov maxValue, eax

mov eax, DelayTime
imul eax, 100
mov edx,0
mov ecx ,maxValue
div ecx
imul eax,-1
add eax, 128
call writeint

mov edx, offset myMessage
```

```
call writestring

call readint

mov switch, eax

.IF switch == 1

jmp beginwhile

.ENDIF

endwhile:


exit

main endp

end main
```

Screenshot for Code Execution



```
if you want to turn on the the motor enter 1 if you want to turn it off enter 0 1
enter full stepmode (0) or half stepmode (1) 1
adjust the potentiometer to enter the delay time you want20
enter the direction clockwise (0) or anti clockwise(1) 1
enter themaximum delay value that the motor can handle 25
+48if you want to turn on the the motor enter 1 if you want to turn it off enter 0
```