PROJECT REPORT

2025

ES314-LAB

Submitted by**: Hamza Safwan**

Registration No.: **2022198**

Faculty: **FES**

“On my honour, as student of Ghulam Ishaq Khan Institute of Engineering Sciences and Technology, I have neither given nor received unauthorized assistance on this academic work.”

Submitted to:

Engr. Hussain Tariq

2025

Serial-Controlled Stepper Motor

Table of Contents

I. INTRODUCTION

Stepper motors are electromechanical devices that convert electrical pulses into discrete mechanical movements. They are widely used in control systems, robotics, disk drives, printers, and CNC machines where precise position control is required. Most stepper motors used in microcontroller-based systems are of the unipolar or four-phase type, with four coil windings and a common terminal. Each coil's energizing pattern allows the rotor to align itself with a new position, thus rotating the motor in fixed angles called step angles.

II. COMPONENTS USED

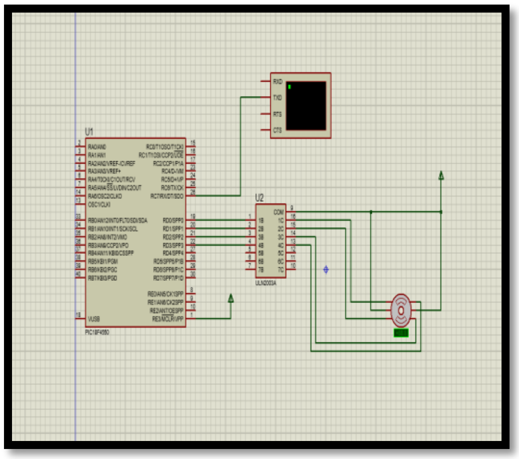


Figure 1. Proteus Simulation Circuit

The circuit used in the simulation is shown in Figure 1 below:

Component	**Purpose**
PIC18F4550 Microcontroller	Logic control and UART communication.
-ULN2003A Driver IC	current amplifier to safely drive the stepper motor coils
4-Phase Unipolar Stepper Motor	Converts electrical pulses into mechanical rotation.
Virtual Terminal	allowing the user to input commands via UART.
DC Power Supply	Powers the PIC, ULN2003, and the motor.
Connecting wires	Establish connections between components

III. METHODOLOGY

The project utilizes a PIC18F4550 microcontroller programmed using XC8 in MPLAB X IDE. The stepper motor is driven using an 8-step half-step sequence sent to PORTD (RD0–RD3), interfaced via a ULN2003 driver IC. Serial communication is established using the built-in UART module of the PIC. The TX pin of the Proteus Virtual Terminal is connected to the RX pin (RC7) of the PIC. Upon receiving ASCII characters 'F', 'B', or 'S', the code respectively initiates clockwise rotation, counterclockwise rotation, or stop condition. The user enters commands via the Virtual Terminal to control direction in real-time.

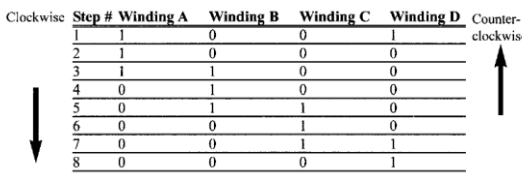
A. Theory

I. Step Angle and Step Speed:

The step angle is the smallest angle a motor can rotate in response to one control pulse. Smaller angles allow finer control Common step angles include 1.8°, 2.5°, and 5°. The speed of the motor is determined by the switching frequency of the control signals outputs. Each step is followed by a delay to make motion visible.

$$\text{Steps per second} = \frac{\text{rpm} \times \text{steps per revolution}}{60}$$

II. *Half Step Sequence*



Clockwise

Step #	Winding A	Winding B	Winding C	Winding D
1	1	0	0	1
2	1	0	0	0
3	1	1	0	0
4	0	1	0	0
5	0	1	1	0
6	0	0	1	0
7	0	0	1	1
8	0	0	0	1

Counter-clockwis

Figure 2. Half Step Sequence

The 8-step half-step sequence is a common stepping method for controlling unipolar stepper motors with higher precision. In this technique, the motor energizes coils in a mixed pattern of single and dual-phase steps, allowing finer control of angular motion. Each step in the table alternates between activating a single winding and a pair of windings, effectively halving the motor’s step angle. This not only improves positioning accuracy but also enables smoother rotation. The clockwise direction follows steps 1 through 8, while the counter-clockwise direction follows them in reverse, as illustrated by the directional arrows.

III. *ULN2003A Working*

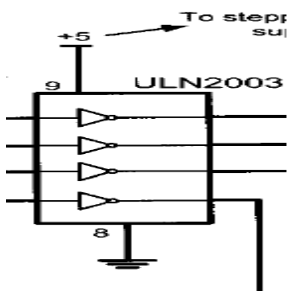


Figure 1. ULN2003A

The ULN2003A is a high-voltage, high-current Darlington transistor array used to interface low-power microcontrollers like the PIC18F4550 with high-power loads such as stepper motors. Each input pin of the ULN2003 controls a pair of transistors in a Darlington configuration, which amplifies the input signal. When a logic high signal is applied at the input (from the microcontroller), the corresponding Darlington pair turns on, pulling the output to ground and allowing current to flow through the connected stepper motor coil.

Pin 9 (COM) is connected to the motor's supply voltage (+5V in this case), while Pin 8 is connected to ground. The internal flyback diodes protect the circuit by safely dissipating the back EMF generated by the motor's inductive windings. This IC ensures safe and reliable operation of the motor without damaging the microcontroller's I/O pins.

IV. *Serial Communication Configure*

$$SPBRG = \frac{Fosc}{(16 \times Baud\ Rate)} - 1$$

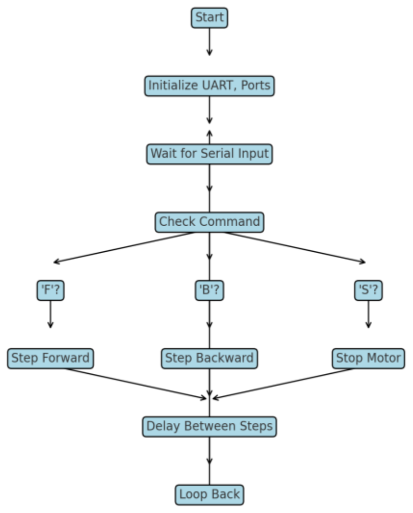
$$SPBRG = \frac{4{,}000{,}000}{(16 \times 9600)} - 1 \approx 25$$

In this project, a standard baud rate of **9600 bps** was used for UART serial communication between the PIC18F4550 microcontroller and the virtual terminal. This baud rate ensures reliable and error-free data transmission. The value of the SPBRG register was calculated based on the oscillator frequency (4 MHz) using the high-speed asynchronous mode formula. By setting BRGH = 1 and SPBRG = 25, accurate baud rate generation was achieved, facilitating smooth communication with external devices.

B. Proteus Simulation Setup

The simulation includes the PIC18F4550 microcontroller connected to the ULN2003A driver IC. The driver controls a 4-phase unipolar stepper motor based on outputs from PORTD. A Virtual Terminal is used to send characters ('F', 'B', 'S') via RC7 (UART RX). The simulation allows real-time control of the motor.

Here is a flowchart Representation of the project:



C. Embedded C Code Overview

The code is written in MPLAB XC8 and uses UART to receive character commands. Depending on the command received, it executes a specific index from an 8-step half-step sequence that is output to PORTD. The stepper motor is driven accordingly using the ULN2003.

IV. CONCLUSION

The integration of serial communication and stepper motor control demonstrates how microcontrollers like the PIC18F4550 can be effectively used for precision actuation tasks. Using the ULN2003 provides safe and reliable power interfacing, while UART allows dynamic user interaction without extra hardware.

V. REFERENCES

M. A. Mazidi, R. D. McKinlay, and D. Causey, *PIC Microcontroller and Embedded Systems: Using Assembly and C for PIC18*. Pearson Education, International Edition, 1st ed., 2008.

Stepper Motor interfacing with PIC18F4550 | PIC Controllers

Microcontrollers Lab, "Serial Controlled Stepper Motor using PIC Microcontroller," YouTube, https://youtu.be/DWz2Srch460?si=whNfubDY1lpnh7ZV

VI. APPENDIX

```
#define _XTAL_FREQ 4000000


#include <xc.h>

#include <stdint.h>


#pragma config FOSC = INTOSC_EC

#pragma config WDT = OFF

#pragma config LVP = OFF

#pragma config PBADEN = OFF
```

```
const uint8_t step_sequence[8] =
{0x09, 0x08, 0x0C, 0x04, 0x06, 0x02,
0x03, 0x01};


uint8_t index = 0;

char command = 'S';


void UART_Init(void) {

    TRISCbits.TRISC7 = 1;

    TRISCbits.TRISC6 = 0;

    SPBRG = 25;

    TXSTAbits.BRGH = 1;

    RCSTAbits.SPEN = 1;

    RCSTAbits.CREN = 1;

    TXSTAbits.TXEN = 1;

}


char UART_Read(void) {

    while (!PIR1bits.RCIF);

    return RCREG;

}


void step_delay(void) {

    __delay_ms(100);

}

void main(void) {


    OSCCON = 0x60;

    UART_Init();


    TRISD = 0x00;

    LATD = 0x00;

    while (1) {

        if (PIR1bits.RCIF) {

            command = UART_Read();

        }

        if (command == 'F' ) {

            LATD =
step_sequence[index];

            index = (index + 1) % 8;

            step_delay();

        }

        else if (command == 'B' ) {

            if (index == 0) index =
7;

            else index--;

            LATD =
step_sequence[index];

            step_delay();

        }

        else if (command == 'S' ) {

            LATD = 0x00;

        }

    }

}
```