Stepper Control by Using Wireless System

Varad Dhotre[1] Rahul Gunjker[2] Anuradha Nikam[3]
[1,2,3]Department of Electrical and Electronics Engineering
[1,2,3]Shri Shivaji Institute of Engineering and Management Studies Parbhani, India

Abstract — This paper focuses on controlling a stepper motor using infrared (IR) signals transmitted through an IR remote. The entire system is managed by the 8-bit microcontroller AT89S52. The system operates within a range of up to 10 meters. The microcontroller receives the IR signal from the transmitter, processes it, and uses it to control the motor's movement. A stepper motor is an electromechanical device that converts electrical pulses into discrete mechanical movements. The motor's shaft rotates in fixed increments when the proper sequence of electrical pulses is applied. The rotation direction of the motor is determined by the pulse sequence, while the speed is controlled by the frequency of the pulses. The motor's rotation length is determined by the number of pulses sent.

Keywords: IR-Infrared, MC-Microcontroller, EM-Electromechanical, PID-Proportional Integral Derivative, PI-Proportional Integral

I. INTRODUCTION

A stepper motor is an electromechanical device that converts electrical pulses into precise mechanical movements. The motor's shaft rotates in fixed step increments when electrical command pulses are applied in sequence. The direction of rotation depends on the pulse sequence, the speed of rotation is determined by the pulse frequency, and the distance moved by the motor is related to the number of pulses applied. The main objective of this project is to control a stepper motor using an infrared (IR) signal sent from an IR remote. The system is managed by an 8-bit microcontroller, the AT89S52, which receives and processes the IR signal and generates commands for the motor accordingly. The range of the system extends up to 10 meters. Stepper motors are used in various applications such as computer peripherals, business machines, process control, machine tools, and robotics. In fields like robotics, silicon processing, IC bonding, and laser trimming, remote control of stepper motors is essential for improving operational efficiency and safety. The IR-controlled stepper motor system allows operators to control machinery remotely, reducing the need for direct intervention and increasing convenience in controlled environments.

II. LITERATURE REVIEW

In the present work, a micro stepping technique using sine and cosine waves is employed to drive the stepper motor, instead of using discrete pulses. The micro stepper driver, based on the LM3S6965, is ideal for small to medium micro stepper motors and provides high resolution. Motor control for specific rotations, direction, and speed is achieved by adjusting pulse numbers and frequencies. The experiment demonstrated smooth operation, high precision, rapid response, and system stability.

The project also focuses on DC motor speed control and temperature sensing. The speed control system uses a PIC 16F877A microcontroller, allowing motor speed to be adjusted with a variable resistor. The temperature of the environment is sensed using the LM35, a precision temperature sensor, and displayed on an LCD screen. The microcontroller manages the motor speed and temperature sensing functions.

The application of high-performance motor drives is widespread in fields such as rolling mills, chemical processes, electric trains, robotic manipulators, and home appliances. DC motors offer the ability to control speed, torque, and direction in real-time to meet various requirements. This project aims to design a DC motor speed control system, using a microcontroller for closed-loop control. By varying the PWM signal to the motor driver, the system maintains the desired speed even under load variations. Temperature readings and set values are displayed on the LCD, and the system's performance was verified across different temperatures.

Position control of stepper motors is crucial for systems requiring precise control. In this study, a fuzzy logic controller combined with a PID controller is used to improve the motor's response. MATLAB/Simulink simulations show that this combination reduces the system's settling time and overshoot, leading to better control performance. The fuzzy logic controller adjusts the system based on error changes over time, minimizing errors and improving stepper motor positioning.

The project is part of a broader study on mechatronic systems, integrating theoretical studies, practical research, and optimization techniques. The system's behavior is modeled using the Runge-Kutta method for stepper motor motion and analyzed using MATLAB/Simulink. A step control technique was found to be effective for stepper motor applications, providing stable operation with minimal oscillations around the desired position. The dynamic stability was also studied using transfer functions and Bode diagrams in MATLAB/Simulink, which confirmed satisfactory system performance despite low vibrations.

III. HARDWARE AND SOFTWARE USED

A. *Hardware Components:*

1) 89S52 Microcontroller: The AT89S52 microcontroller is the central control unit, responsible for processing input signals and controlling the motor's speed and direction.
2) Voltage Regulator (7805): The 7805-voltage regulator provides a stable 5V output to power the microcontroller and other components, ensuring reliable performance.
3) Diode (IN4007): The 1N4007 diode is used for reverse voltage protection, safeguarding the circuit from incorrect power connections.
4) IR Remote: The IR remote sends commands in the form of modulated infrared light, which is detected by the IR receiver to control the stepper motor.
5) T-SOP (IR Receiver): The T-SOP infrared receiver detects the modulated IR signals from the remote and

All rights reserved by www.ijsrd.com

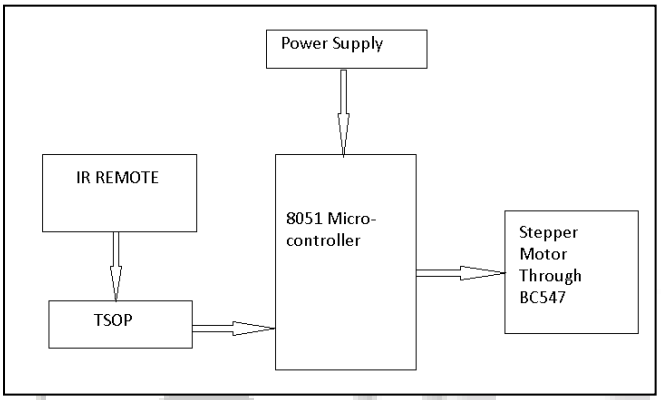
converts them into electrical signals, which are then processed by the microcontroller.

6) Stepper Motor: The stepper motor converts electrical pulses into precise mechanical movements. It is controlled by the microcontroller based on the pulses received from the IR remote.

B. Software Components:

1) Kiel u-Vision 3.0: Kiel u-Vision is used for developing embedded applications for the AT89S52 microcontroller. It assists in compiling, debugging, and programming the microcontroller.
2) PRO51 Programmer Software: This software is used for programming the 8051 microcontroller, allowing the transfer of the compiled code into the microcontroller for execution.

IV. BLOCK DIAGRAM



1) AVR Microcontroller (e.g., ATmega32): This microcontroller serves as the brain of the system. It takes input commands and generates appropriate control signals for the stepper motor driver.
2) Stepper Motor: A type of electric motor that rotates in precise steps, allowing for accurate control of position. Stepper motors are ideal for applications that require precise movement, such as robotics and automation.
3) Motor Driver (e.g., ULN2003): A driver IC, like the ULN2003, is used to amplify the control signals from the microcontroller and provide sufficient current to the stepper motor.
4) Power Supply: A regulated power supply is required to provide the appropriate voltage and current to both the AVR microcontroller and the stepper motor.
5) Control Interface (e.g., Push Button or IR Remote): This is used to input commands to change the speed and direction of the motor. An IR remote can be used for wireless control, while a push button can serve for basic on/off or speed control.

A. System Overview:

1) Speed Control: The speed of the stepper motor is directly related to the frequency of the pulses provided to it. By adjusting the interval between consecutive pulses, the microcontroller can change the motor's speed.
2) Direction Control: The direction of the stepper motor is controlled by changing the sequence of the pulses sent to the motor. Typically, the motor rotates clockwise when the pulses are applied in one order and counterclockwise when the order is reversed.
3) Pulse Generation: The AVR microcontroller generates a sequence of pulses using its I/O pins. These pulses are fed into the stepper motor driver, which controls the current flow to the motor coils in the correct sequence, making the motor rotate.

B. Operation:

1) Speed Control:

The AVR microcontroller generates pulse signals with a time delay between them. By increasing the time delay, the motor will rotate slower; by decreasing the delay, the motor will rotate faster. The delay time is calculated based on the desired speed.

2) Direction Control:

The direction is controlled by the sequence of pulses sent to the motor coils. The microcontroller can reverse the sequence to change the direction from clockwise to counterclockwise or vice versa.

3) User Input:

The user can input speed and direction commands via an IR remote or physical switch. For an IR remote, the signals are received by an IR receiver, which decodes the commands and passes them to the microcontroller for processing. For switches, the microcontroller detects the changes in the input pin state and adjusts the motor accordingly.

C. Software Implementation:

The software for controlling the stepper motor involves the following tasks:

1) IR Signal decoding (if using an IR remote): The microcontroller receives the IR signals and decodes them into commands for controlling the speed and direction of the motor.
2) Pulse Sequence Generation: The microcontroller generates a timed sequence of pulses that correspond to the required direction and speed of the stepper motor.
3) Control Algorithm: The software continuously monitors the user inputs (from switches or remote), adjusts the pulse frequency for speed control, and alters the pulse sequence for direction control.

V. EXAMPLE CODE OUTLINE (FOR AVR USING C LANGUAGE):

```
#define F_CPU 8000000UL // Set the microcontroller clock speed
#include <avr/io.h>
#include <util/delay.h>

void stepperMotorClockwise() {
   // Define the sequence for clockwise rotation
   PORTC = 0x01; // Step 1
   _delay_ms(10);
   PORTC = 0x02; // Step 2
   _delay_ms(10);
   PORTC = 0x04; // Step 3
   _delay_ms(10);
   PORTC = 0x08; // Step 4
   _delay_ms(10);
}
```

All rights reserved by *www.ijsrd.com*

```
void stepperMotorCounterClockwise() {
   // Define the sequence for counterclockwise rotation
   PORTC = 0x08; // Step 1
   _delay_ms(10);
   PORTC = 0x04; // Step 2
   _delay_ms(10);
   PORTC = 0x02; // Step 3
   _delay_ms(10);
   PORTC = 0x01; // Step 4
   _delay_ms(10);
}

int main() {
   DDRC = 0x0F; // Set PORTC as output for motor control
   while(1) {
      stepperMotorClockwise(); // Rotate motor clockwise
      _delay_ms(500); // Wait for some time
      stepperMotorCounterClockwise(); // Rotate motor counterclockwise
      _delay_ms(500); // Wait for some time
   }
}
```

VI. ADVANTAGES:

Stepper motors offer precise control with rotation proportional to input pulses and full torque at standstill. They provide excellent response to starting, stopping, and reversing, and are highly reliable due to the absence of brushes. The open-loop control simplifies operation and reduces cost. These motors can achieve low-speed synchronous rotation, handle a wide range of speeds, and are stable under various loads. They also eliminate the need for feedback systems and are inexpensive, with standardized frame sizes.

VII. DISADVANTAGES:

Stepper motors are less efficient, drawing power regardless of load, and experience torque loss at higher speeds. They can be prone to resonances and low accuracy, especially under full load. They also lack feedback to detect missed steps, have a low torque-to-inertia ratio, and tend to overheat at high performance levels. Momentary overloads can cause the motor to stall.

VIII. APPLICATIONS:

Stepper motors are widely used in industrial machines, medical devices (e.g., scanners and pumps), consumer electronics (e.g., cameras), and security systems.

IX. LIMITATIONS:

Stepper motors can suffer from resonance effects, long settling times, and rough performance at low speeds without micro stepping. They can consume current regardless of load, leading to overheating and noise. Additionally, they may experience position loss in open-loop systems and are limited in size and accuracy by their mechanical components.

X. CONCLUSION:

In this work, a micro stepping technique using sine/cosine waves is employed to drive the stepper motor, replacing discrete pulses. The LM3S6965-based micro stepper driver is suitable for small and medium motors, offering high micro stepping resolution. Motor control, including rotation count, direction, and speed, is achieved by adjusting the number and frequency of pulses. The experiment demonstrated smooth operation, high precision, rapid response, and stability in the system.

REFERENCES

[1] http://www.atmel.com/images/doc8017.pdf
[2] http://dspace.thapar.edu:8080/dspace/bitstream/10266/1591/1/Hirday+Pal+SIngh(800941013).pdf
[3] http://www.ijareeie.com/upload/2014/april/64_Position.pdf
[4] http://power.eecs.utk.edu/pubs/ChiassonCSTdec93.pdf
[5] http://www.wseas.us/e-library/conferences/2013/Vouliagmeni/MRME/MRME-14.pdf
[6] http://www.ijareeie.com/upload/2014/july/19_Design.pdf
[7] https://ojs.lboro.ac.uk/ojs/index.php/JDTE/article/viewFile/350/327
[8] http://www.ijsret.org/pdf/120548.pdf

All rights reserved by *www.ijsrd.com*