Position Control using Stepper Motor

Experiment RB

In this experiment, you will study how a stepper motor is controlled through a microcontroller and what determines certain characteristics of a stepper motor like Error-Free Mode of Operation and Slewing Mode of Operation.

Stepper Motor

A stepper motor is a brushless DC motor which is controlled by electrical commutation. We will be using a unipolar type Stepper motor with a step angle of 3.6 degrees. This stepper motor has four coils and each coil can be powered on or off independently. The stepper motor has 6 wires. The first two wires are meant for connecting V_{CC} and the other four wires are connected to the ends of four coils which are named A, B, C and D.

There are two main schemes for moving a stepper motor – Full-step and Half-step.

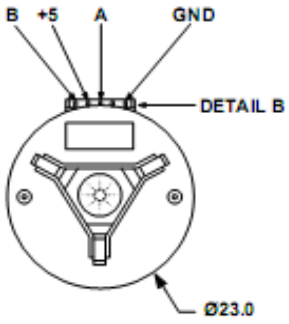
Full-step drive: In this mode there are only four sequences that complete a cycle: either in the sequence 1000-0100-0010-0001 or 1100-0110-0011-1001, where 1 corresponds to the coil being turned ON and 0 corresponds to coil being turned OFF. The direction of rotation of the motor is controlled by the sense of the sequence.

Half-step Drive: This mode combines the two sequences given above, resulting in the angular step size being half of the step angle.

The Stepper Motor driver circuit consists of 4 TIP31C Transistors. The collectors of these transistors are connected to the four coils A, B, C and D. The base of each transistor is connected to one of the 4 Port pins PB0, PB1, PB2 and PB3 through a 250Ω resistor. The two V_{CC} wires are connected to +5V. A transistor is turned OFF/ON if its base voltage is logic LOW/HIGH.

Shaft Encoder

The encoder used in this experiment is an Incremental Shaft Encoder having two channels, each providing 200 pulses per revolution (cpr), the two outputs being in phase quadrature. The Encoder Disc has been coupled to the Stepper motor shaft to enable exact measurement of the amount of rotation of the stepper motor shaft.



Shaft Encoder Connection diagram

Programming requirements:

1. In the main() routine, configure PORTB0-B3 as output and generate Full-step drive sequence. While switching between sequence 1 to sequence 2, a delay of a few milliseconds is necessary for coil A to power down and coil B to power up (the coils are inductive !!). Same delay should be given between any two sequences. Note that this delay decides the speed of rotation of the motor. Connect these PORT pins to the corresponding pins on motor driver circuit. You should see that the stepper motor rotates once the correct connections have been made.

    a. Start with a large delay so that the motor moves with clearly discernible steps. Visually count the number of steps the motor takes and verify that this is the same as the number of pulses applied to the coils.

    b. Now decrease this delay gradually and observe its effect on the speed of Stepper motor. Note down the minimum delay at which the stepper motor does not rotate at all.

    c. Repeat the above step for half-step drive and again find the minimum delay required between two sequences.

2. Write a routine to select External Interrupt 0, and configure it to trigger on falling edge of input.

3. Write an ISR (Interrupt service routine) named ISR(INT0_vect). In this routine increment a **global volatile integer** variable "count_tic" by 1. (We have declared this variable as **global** since it is being used both in Interrupt and main routine, declaring it **volatile** makes sure that the avr-gcc compiler does not optimize the code in such a fashion that "count_tic" is not incremented properly). The value of "count_tic" gives the number of count of the encoder. Use this value to find out by how much angle the rotor of stepper has rotated. Display the values on the LCD.

4. Write a loop in main() and generate one cycle of sequences in one iteration of the loop. After executing the loop "n" number of times, stop the stepper motor and display the value of "count_tic" and "n". Then execute the loop again.

5. In the main() routine use a delay in which there are discernible steps and find out how many sequences you have to generate in Full-step drive mode to make the motor rotate by 360^{o}. In this case the stepper motor rotates slowly and the rotor stops at each position before turning to the next position, this is the Error Free Mode of Operation. Gradually decrease the delay used till there is a significant difference between the reading of encoder and the expected reading of the stepper motor.

6. Keep reducing the delay used in step 5 and observe that the error between encoder reading and expected reading also increases. This is Slew Mode of operation. In this mode Stepper Motor rotates at higher rpm but there is an error in position control because even before the rotor 'locks' in one position it is pulled towards the next position and hence there is a 'slippage' of the rotor teethes. This mode of operation is used when the stepper motor is to be run continuously and the error in position is not of much concern.