Lab 10

Mechatronics
MENG483 Fall 2006

Stepper Motors

1 Objectives

The objective of this laboratory exercise is to familiarize students with principles and control algorithms governing the operation of a stepper motor. A two-phase stepper motor is discussed with the associated driving circuitry and the step control sequences. Performance of a motor under the high-current condition and fast revolution rates is explored. A practical example of application of a stepper motor in the light intensity measurements is presented.

2 Theoretical background

2.1 Introduction

As a dc motor spins, the commutator reverses the direction of the current through the motor windings whenever the rotor has turned a certain amount. Without a commutator, the rotor of a dc motor would align itself with the magnetic field of the stator and then hold that position. Stepper motors are designed to use this holding property, and thus they do not have commutators. An external drive circuit provides the commutated signal.

Stepper motors come in many varieties. In this lab, we will only discuss the two-phase unipolar stepper motor. For more information on other varieties of stepper motors, see reference [1].

2.2 Internal Construction

Figure 1 depicts the basic construction of a two-phase unipolar motor. 'Two-phase' means that there are two primary motor windings. Each of these windings has a center-tap which allows the direction of current to be changed by switching between the two ends, *a* and *b*. The term 'unipolar' refers to this center-tap configuration. In the figure, winding 1 is distributed between the top and bottom motor poles while winding 2 is distributed between the left and right motor poles. The rotor has six poles, resulting in a step angle of 30 degrees. To have a smaller step angle, each winding would be distributed among more motor poles than just two, and the rotor would also have more poles.

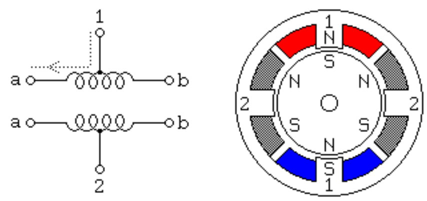


Figure 1: Two-phase unipolar motor [1]

Figure 2 is photo of the internals of the motor you will be using. The rotor has 50 poles and each motor winding is distributed among 4 poles. It has 200 steps per revolution, or 1.8 degrees per step.

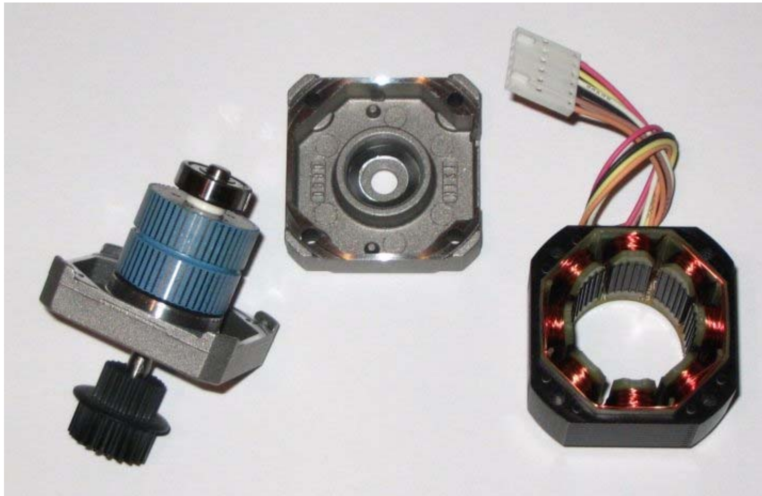


Figure 2: Shinano Kenshi STP-42D241 motor internals

2.3 Control Sequence and Circuitry

Figure 1 shows the stator poles when winding 1a is energized. North is on the top and south on the bottom. Let's say that winding 2a is then energized and north is on the left and south is on the right. The rotor would then move so that the closest south pole on the rotor would align with the stator's north pole, and the closest north pole on the rotor would align with the stator's south pole. Note that the stator poles have 'rotated' counterclockwise 90 degrees while the rotor has rotated clockwise 30 degrees. To continue rotating clockwise, winding 1b should be energized, followed by winding 2b. The control sequence is then:

Winding	2b	2a	1b	1a	HEX
t_0	0	0	0	1	0x01
t_1	0	1	0	0	0x04
t_2	0	0	1	0	0x02
t_3	1	0	0	0	0x08
…	0	0	0	1	0x01
bit	*3*	*2*	*1*	*0*	
sequence	4	3	2	1	

The HEX column indicates the control command that would be used if winding 1a were controlled by bit 0, winding 1b by bit 1, and so on.
The typical circuitry used to drive a unipolar stepper motor is given in Figure 3. The center-taps are connected to the supply voltage, and the control signals connect one of the half-windings to ground.

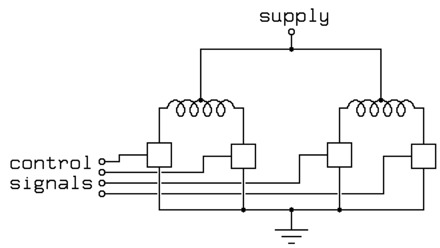


Figure 3: Unipolar stepper motor control circuit [1]

For this lab, we will be using a slightly different circuit which uses the L293D chip, which is a versatile chip that can be used in many motor applications. The schematic for the control circuit is shown in Figure 4. Note that the center-taps are connected to ground instead of the motor power supply.

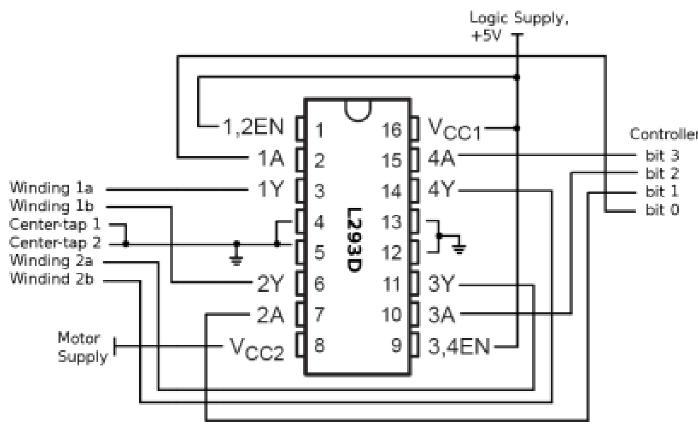
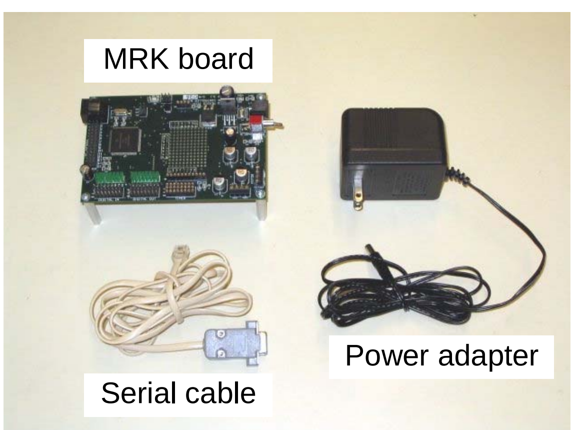


Figure 4: Stepper motor control circuit

3 Equipment



- MRK Controller Board
- Serial communication cable
- Power adapter
- Two-phase, unipolar stepping motor
- Stepper motor control circuit (Fig. 4)
- Light sensor circuit (Fig. 5)

4 Procedure

Exercise #1: In this exercise, you will use the keyboard to activate each motor winding. Pressing one of the number keys 1-4 will provide power to the corresponding winding, and pressing the number 0 will turn all windings off. The code that generates the control signals is given below. Power is provided through the Left Motor port. To use the motor port, you must enable the High Power switch on the MRK board and use the command `motor_out(LEFT, 255)` to supply a constant 15 V.

The drive circuit has been wired so that the correct drive sequence is {1, 3, 2, 4}. Press the number keys in this order and note how the shaft turns. What happens if you go out of order {1, 2, 3, 4}? What happens if you reverse the order {4, 2, 3, 1}?

```
#include "MRK.h"

void sci_isr(void) {
      char c;
      static char command = 0x00;
      c = fgetchar(SCI0);
      if (c == '0') command = 0x00;
      else if (c == '1') command = 0x01;
      else if (c == '2') command = 0x02;
      else if (c == '3') command = 0x04;
      else if (c == '4') command = 0x08;
      digital_out(OUT, ALL, command);
}

int main() {
      digital_out(OUTPUT,ALL);
      setup_sci(SCI0,INTERRUPT,&sci_isr);
      motor_out(LEFT, 255);

      while (1);
}
```

Exercise #2: For this exercise, the control program has been modified so that the less-than and greater-than keys (< and >) step the motor in either direction. What does the code fragment `step_sequence[state & 0x03]` do? Why won't `step_sequence[state]` work? How else could you write the code to produce the same effect?

```
#include "MRK.h"

void sci_isr(void) {
      char c;
      const char step_sequence[] = {0x01, 0x04, 0x02, 0x08};
```

```
      static int state = 0;
      c = fgetchar(SCI0);
      if (c == ',') state = state + 1;         // comma, the < key
      else if (c == '.') state = state - 1;    // period, the > key
      digital_out(OUT,ALL, step_sequence[state & 0x03]);
}

int main() {
      digital_out(OUTPUT,ALL);
      setup_sci(SCI0,INTERRUPT,&sci_isr);
      motor_out(LEFT,255);

      while (1);
}
```

Exercise #3: There is a limit to the rate at which a stepping motor can be stepped. In this exercise you will determine the approximate value of the minimum time between steps. The program below allows the delay between steps to be changed from 1 to 9 milliseconds by pressing the number keys 1 through 9.

```
#include "MRK.h"

int step_delay;

void sci_isr(void) {
      char c;
      c = fgetchar(SCI0);
      // change to number 0-9
      step_delay = c - '0';
      if (step_delay<0 || step_delay>9) step_delay=0;
}

int main() {
      const char step_sequence[] = {0x01, 0x04, 0x02, 0x08};
      int state = 1;
      digital_out(OUTPUT,ALL);
      setup_sci(SCI0,INTERRUPT,&sci_isr);
      motor_out(LEFT,255);

      step_delay = 0;
      while (1) {
            if (step_delay > 0) {
                  delay(step_delay);
                  state = state + 1;
                  digital_out(OUT,ALL, step_sequence[state & 0x03]);
            }
      }
}
```

Begin at a delay of 9 milliseconds and work down to 1 millisecond. At each speed, note how much torque the motor can provide (just get a rough idea by using your fingers). What minimum time between steps can the motor handle? What happens if the step command is slightly too fast? What happens if the step command is much too fast?

The torque of the motor is related to both the speed and the current. Use a wire to short-circuit the current limiting resistor, **but do not let the motor run too long at this higher current as it might damage the MRK board – remove the wire when you are done**. Can the motor run at a higher speed now? Give an explanation for your observations in this exercise.

Exercise #4: In this exercise, you will attach a light sensor to the stepper motor and write a program that searches for the brightest light source. As the motor rotates one full revolution, the program must remember the angle at which the light was brightest. Then, the motor should rotate back to that position.

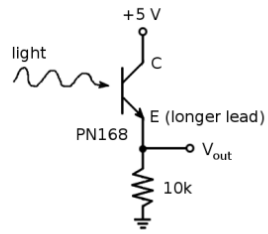


Figure 5: Light sensor circuit

5 Report and analysis requirements

5.1 Theory

Explain how the stepper motor works.

What is the difference between a permanent magnet (PM) and variable reluctance (VR) stepper motors?

Which type of stepper motor can retain the holding torque when the stator is not energized?

5.2 Results and analysis

Answer all questions in previous sections of these laboratory instructions.

Present and discuss you results.

Experimental procedures in this laboratory exercise utilize a full step rotation of a stepper motor that results in the shaft increment of 30 degrees. What would you suggest to consider in order to reduce this increment?

What is the fastest rate at which the stepper motor still can perform adequately? Discuss what happens if you run the stepper motor too fast.

What is the advantage of using a stepper motor versus DC and servo motors in position control applications?

References

1. “Control of Stepping Motors,” http://www.cs.uiowa.edu/~jones/step/

Last updated
November 1, 2006.