Design Assignment 6: Stepper Motor

Alexis Adie and Madison Mastroberte
ELC 411-01: Embedded Systems

Submitted:
November 24th 2017

Design Assignment 6: Stepper Motor

Alexis Adie and Madison Mastroberte
Department of Electrical and Computer Engineering
The College of New Jersey
2000 Pennington Road, Ewing, NJ 08618, USA
(adiea1, mastrom7) @tcnj.edu

I. Results

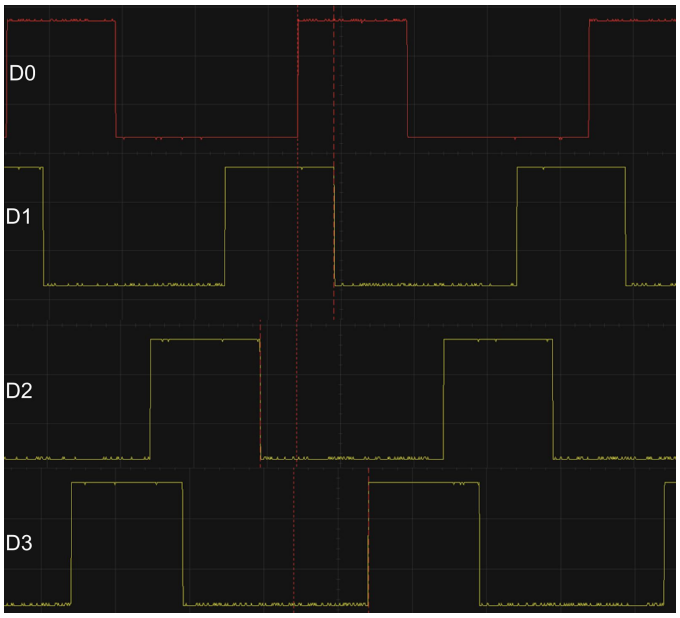

Figure 1: Four square waveforms depicting the four coils (A, $\overline{A}$, B, $\overline{B}$) for the stepper motor [20 ms/div | 2V/div].

Figure 1 demonstrates all the internal motor outputs on a single time axis. Each data pin stays on for 30 ms and this lasts over three continuous phases. For example, D0 is on for Phase 7, 0, 1. Each phase lasts 10 ms, which is shown in Figure 4 by the white dotted lines.

Utilizing Figure 2 and 3, which depicts the motor connections and overall architecture, the pins and connections are labeled. D0 represents the waveform output of the blue coil, B. D1 represents the waveform output of the pink coil, $\overline{A}$. D2 represents the waveform output of the yellow coil, $\overline{B}$. D3 represents the waveform output of the orange coil, A.

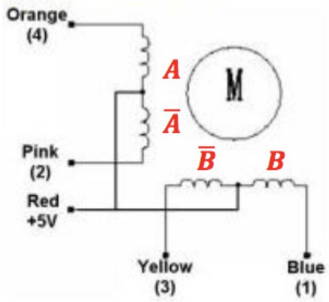

Figure 2: Coils of the stepper motor with corresponding A, $\overline{A}$, B, $\overline{B}$ labels.

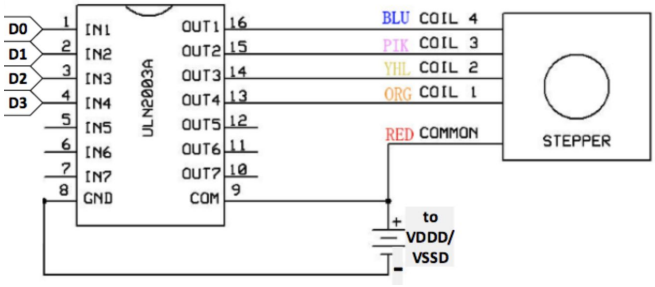

Figure 3: Motor drawing depicting coils and corresponding outputs from the PSoC GPIO.

In Figure 4, the motor begins on phase P1 and cycles through all of the phases three times. In other words, the motor goes through P0 - P7 three times, however at the the time of observation, the cycle is CCW. Therefore, the cycles consist of {P1, P0, P7, P6 … P1, P0}.

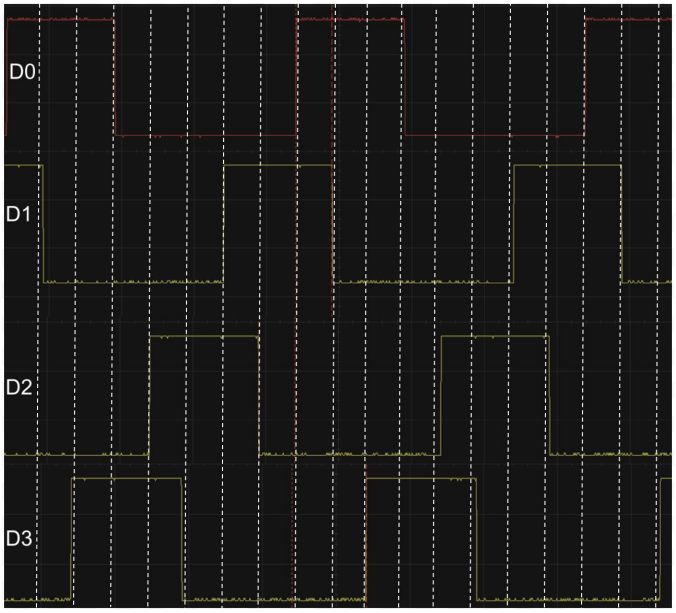

Figure 4: Four square waveforms depicting the four coils (A, $\overline{A}$, B, $\overline{B}$) for the stepper motor with vertical segments which allow better visualization of the phases [20 ms/div | 2V/div].

II. DISCUSSION

The team measured the winding resistance to be $28.8\,\Omega$, between the common center tap ground and motor terminal coil.

A. Shaft Length

The shaft of the motor goes through the entirety of the motor since the shaft connects to the network of gears inside the motor. The shaft has a magnet around its lower portion that is manipulated by the changes in electromagnetic fields due to the teeth and active coils inside the motor.

B. Full Revolution of Steps

Utilizing the datasheet of the motor it was determined that with a gear ration of 64 and a stride angle of $5.625\circ$, the motor has 4096 steps for one revolution. Since the count in the lab procedure is 2048, this means that the shaft is only going $180\circ$. Then once it hits this limit, it changes directions and goes another $180\circ$ before changing again. Therefore, during operation for 20 ms, the shaft goes a full 180 degrees before changing direction.

C. Peak Current

Since there is a 5V assumption and a resistance of $28.8\,\Omega$:

$$V = IR = \frac{V}{R}$$

$$I = \frac{5}{28.8}$$

The current is equivalent to 0.1736 mA.

Since peak current is:

$$I = \frac{I_p}{\sqrt{2}}$$

The peak current is equivalent to **0.246 mA**.

D. Current Driven

Utilizing the data sheet, I_{GPIO} has a maximum value of 41 mA. Since the peak current is lower than the maximum current, the current cannot be driven directly through this pin.

VI. Appendix

Madison Mastroberte's Code

```
#include "project.h"
int main(void)

{
    //Initialize variables
    int phase = 0;
    int direction = 1;
    int count = 0;
    int i = 0;

    //Generate State Cases for Driving Motor
    switch(phase)
        {
        case 0:
            DIG_OUT_Pin0 = 1;
            DIG_OUT_Pin1 = 0;
            DIG_OUT_Pin2 = 0;
            DIG_OUT_Pin3 = 0;
            break;

        case 1:
            DIG_OUT_Pin0 = 1;
            DIG_OUT_Pin1 = 1;
            DIG_OUT_Pin2 = 0;
            DIG_OUT_Pin3 = 0;
            break;

        case 2:
            DIG_OUT_Pin0 = 0;
            DIG_OUT_Pin1 = 1;
            DIG_OUT_Pin2 = 0;
            DIG_OUT_Pin3 = 0;
            break;

        case 3:
            DIG_OUT_Pin0 = 0;
            DIG_OUT_Pin1 = 1;
            DIG_OUT_Pin2 = 1;
            DIG_OUT_Pin3 = 0;
```

```
        break;

    case 4:
        DIG_OUT_Pin0 = 0;
        DIG_OUT_Pin1 = 0;
        DIG_OUT_Pin2 = 1;
        DIG_OUT_Pin3 = 0;
        break;

    case 5:
        DIG_OUT_Pin0 = 0;
        DIG_OUT_Pin1 = 0;
        DIG_OUT_Pin2 = 1;
        DIG_OUT_Pin3 = 1;
        break;

    case 6:
        DIG_OUT_Pin0 = 0;
        DIG_OUT_Pin1 = 0;
        DIG_OUT_Pin2 = 0;
        DIG_OUT_Pin3 = 1;
        break;

    case 7:
        DIG_OUT_Pin0 = 1;
        DIG_OUT_Pin1 = 0;
        DIG_OUT_Pin2 = 0;
        DIG_OUT_Pin3 = 1;
        break;
    }
//Increment Count
for (i = 0; i < 2048, ++i)

{
    count = count + i;
}


//10ms Delay
CyDelay(10);

//Increment phase, since count is less than 2048
if (count <= 2048)
{
```

```
        //Increment phase by 1
        phase = phase + 1;
        return phase;
    }

    //Delay 500 ms
    CyDelay(500);

    //Change Direction
    dir = ~dir;
}
```

Alexis Adie's Code

```
#include "project.h"

int main(void)
{
   //Initialize varibles
   int phase = 0;
   int direction = 1;
   int count = 0;

 for (;;)
{
    //loop which continues until count reaches 2048
    for (count = 0; count < 2048; ++count)
      {

            //case statments for each state of the motor
            switch(phase)
            {
               case "0":
                  DIG_OUT_Pin0_Write(1);
                  DIG_OUT_Pin1_Write(0);
                  DIG_OUT_Pin2_Write(0);
                  DIG_OUT_Pin3_Write(0);
                  break;

               case "1":
                  DIG_OUT_Pin0_Write(1);
```

```
    DIG_OUT_Pin1_Write(1);
    DIG_OUT_Pin2_Write(0);
    DIG_OUT_Pin3_Write(0);
    break;

case "2":
    DIG_OUT_Pin0_Write(0);
    DIG_OUT_Pin1_Write(1);
    DIG_OUT_Pin2_Write(0);
    DIG_OUT_Pin3_Write(0);
    break;

case "3":
    DIG_OUT_Pin0_Write(0);
    DIG_OUT_Pin1_Write(1);
    DIG_OUT_Pin2_Write(1);
    DIG_OUT_Pin3_Write(0);
    break;

case "4":
    DIG_OUT_Pin0_Write(0);
    DIG_OUT_Pin1_Write(0);
    DIG_OUT_Pin2_Write(1);
    DIG_OUT_Pin3_Write(0);
    break;

case "5":
    DIG_OUT_Pin0_Write(0);
    DIG_OUT_Pin1_Write(0);
    DIG_OUT_Pin2_Write(1);
    DIG_OUT_Pin3_Write(1);
    break;

case "6":
    DIG_OUT_Pin0_Write(0);
    DIG_OUT_Pin1_Write(0);
    DIG_OUT_Pin2_Write(0);
    DIG_OUT_Pin3_Write(1);
    break;
```

```
            case "7":
               DIG_OUT_Pin0_Write(1);
               DIG_OUT_Pin1_Write(0);
               DIG_OUT_Pin2_Write(0);
               DIG_OUT_Pin3_Write(1);
               break;
            }


         }

    }
    phase = (phase + 1) & 7;
    CyDelay(10);
    direction = ~direction;
    }

    //Delay 500 ms
    CyDelay(500);

  }
  count = 0;
}
```

Commented and Debugged Code

```
#include "project.h"

int main(void)
{
    //Initialize variables
    int phase = 0;
    int dir = 1;
    int count = 0;

    for (;;)
    {
        //Increment the count and switch cases until count reaches 2047
        //This is at the 20second mark when turning clockwise
        for (count = 0 ; count < 2047; ++count)
        {
            //Switch cases to change the state of the motor
```

```
switch(phase)
{
case 0:
    DIG_OUT_Pin0_Write(1);
    DIG_OUT_Pin1_Write(0);
    DIG_OUT_Pin2_Write(0);
    DIG_OUT_Pin3_Write(0);
    break;

case 1:
    DIG_OUT_Pin0_Write(1);
    DIG_OUT_Pin1_Write(1);
    DIG_OUT_Pin2_Write(0);
    DIG_OUT_Pin3_Write(0);
    break;

case 2:
    DIG_OUT_Pin0_Write(0);
    DIG_OUT_Pin1_Write(1);
    DIG_OUT_Pin2_Write(0);
    DIG_OUT_Pin3_Write(0);
    break;

case 3:
    DIG_OUT_Pin0_Write(0);
    DIG_OUT_Pin1_Write(1);
    DIG_OUT_Pin2_Write(1);
    DIG_OUT_Pin3_Write(0);
    break;

case 4:
    DIG_OUT_Pin0_Write(0);
    DIG_OUT_Pin1_Write(0);
    DIG_OUT_Pin2_Write(1);
    DIG_OUT_Pin3_Write(0);
    break;

case 5:
    DIG_OUT_Pin0_Write(0);
    DIG_OUT_Pin1_Write(0);
    DIG_OUT_Pin2_Write(1);
    DIG_OUT_Pin3_Write(1);
    break;
```

```
            case 6:
                DIG_OUT_Pin0_Write(0);
                DIG_OUT_Pin1_Write(0);
                DIG_OUT_Pin2_Write(0);
                DIG_OUT_Pin3_Write(1);
                break;

            case 7:
                DIG_OUT_Pin0_Write(1);
                DIG_OUT_Pin1_Write(0);
                DIG_OUT_Pin2_Write(0);
                DIG_OUT_Pin3_Write(1);
                break;

            }
            //If spinning clockwise (indicated by 1)
            if (dir == 1)
            {
                //The phase will increment by 1, while being masked by 4b0111
                phase = (phase + 0b001) & 7;
                CyDelay(10);
            }
            //If spinning counter-clockwise (indicated by 0)
            else if (dir == 0)
            {
                //The phase will decrement by 1, while being masked by 4b0111
                phase = (phase - 0b001) & 7;
                CyDelay(10);
            }
        }
        //This delay acts as the half second break prior to CW changing to
CCW
        CyDelay(500);

        //This sets the direction to be counter-clockwise and
        //then is masked by 1 bit to only set the value as either 0 or 1
        dir = (~dir) & 0x1;

    }
        //Resets the count
        count = 0;
}
```