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

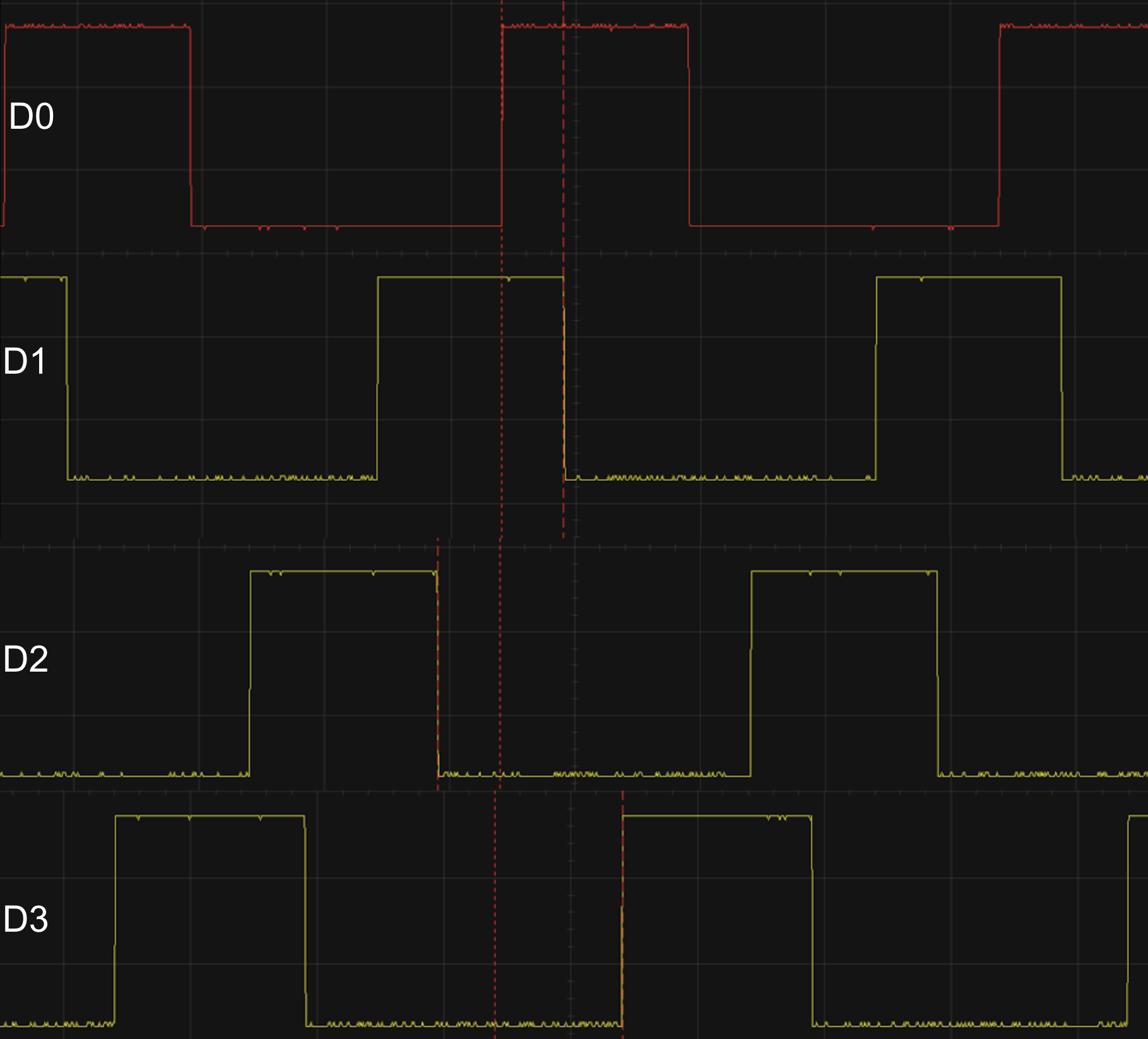
The College of New Jersey

2000 Pennington Road, Ewing, NJ 08618, USA

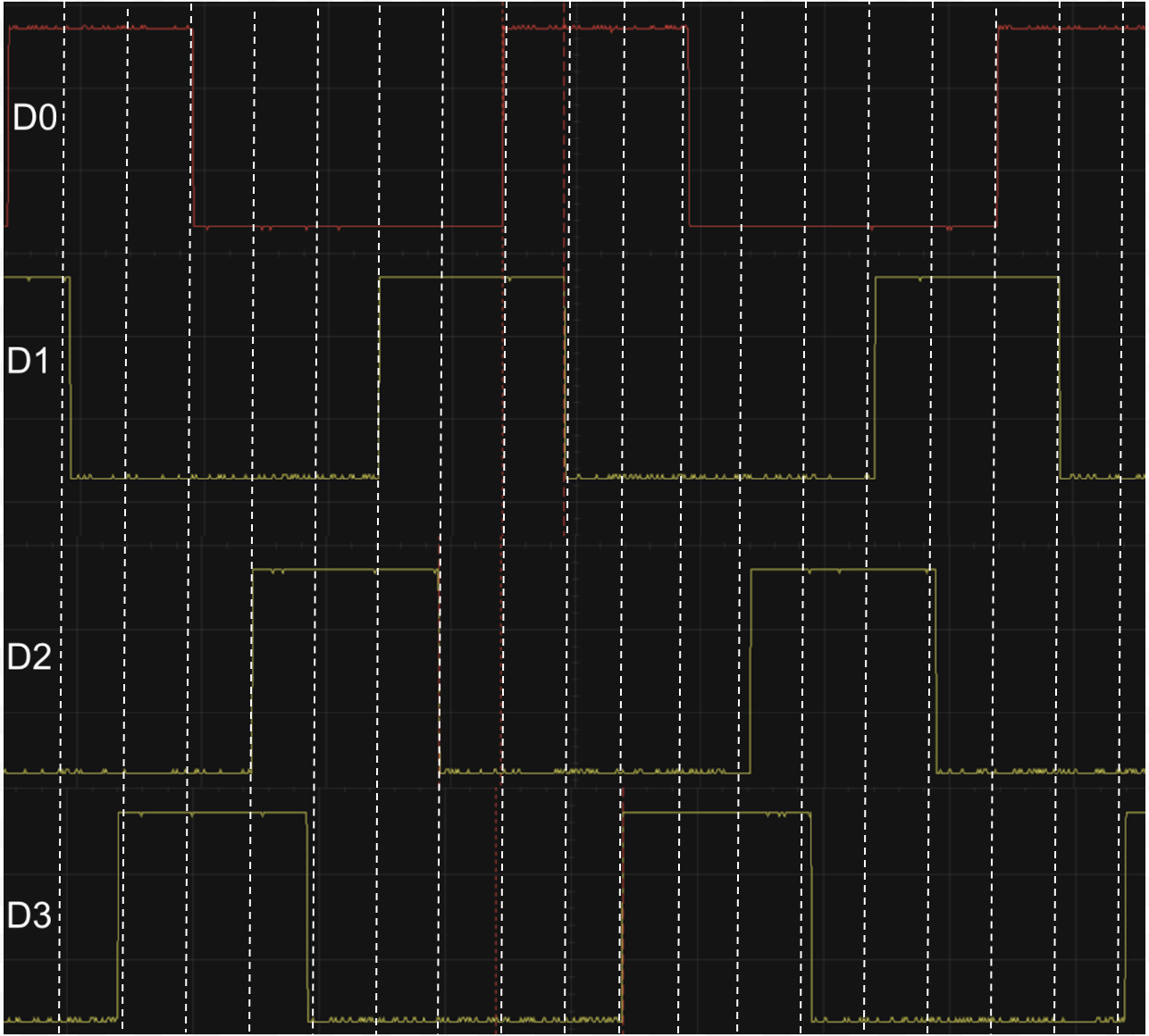
(adiea1, mastrom7) @tcnj.edu

I. Results

*A. Measurements and Observations*



**Figure 1:** Four square waveforms depicting the four coilsfor the stepper motor [20 /div | 2V/div].



**Figure 2:** Four square waveforms depicting the four coilsfor the stepper motor with vertical segments which allow better visualization of the phases [20 /div | 2V/div].

# II. DISCUSSION

The team measured the winding resistance to be

28.8 Ohms.

1. During operation, the shaft goes almost a full 90 degrees before changing direction.
2. Since the software went through 7 steps before changing direction, a full revolution would require 14 steps.
3. I = Io/sqrt2 where Io equals peak current. V/R =I. I = 0.174. Io = 0.246 A = 246 mA
4. Since the maximum value of the GPIO current is 41 mA. 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**

**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;

}

Commented and Debugged Code

#include "project.h"  
  
int main(void)  
{  
 //Initialize varibles  
 int phase = 0;  
 int dir = 1;  
 int count = 0;  
   
 for (;;)  
{  
 for (count = 0; count < 2048; ++count)  
 {  
 switch(dir)  
 {   
 case 1:  
 //Generate State Cases for Driving 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;  
 }   
 break;  
 case (-1):  
 //Generate State Cases for Driving Motor  
 switch(phase)  
 {  
 case 7:  
 DIG\_OUT\_Pin0\_Write(1);  
 DIG\_OUT\_Pin1\_Write(0);  
 DIG\_OUT\_Pin2\_Write(0);  
 DIG\_OUT\_Pin3\_Write(0);  
 break;  
  
 case 6:  
 DIG\_OUT\_Pin0\_Write(1);  
 DIG\_OUT\_Pin1\_Write(1);  
 DIG\_OUT\_Pin2\_Write(0);  
 DIG\_OUT\_Pin3\_Write(0);  
 break;  
   
 case 5:  
 DIG\_OUT\_Pin0\_Write(0);  
 DIG\_OUT\_Pin1\_Write(1);  
 DIG\_OUT\_Pin2\_Write(0);  
 DIG\_OUT\_Pin3\_Write(0);  
 break;  
  
 case 4:  
 DIG\_OUT\_Pin0\_Write(0);  
 DIG\_OUT\_Pin1\_Write(1);  
 DIG\_OUT\_Pin2\_Write(1);  
 DIG\_OUT\_Pin3\_Write(0);  
 break;  
  
 case 3:  
 DIG\_OUT\_Pin0\_Write(0);  
 DIG\_OUT\_Pin1\_Write(0);  
 DIG\_OUT\_Pin2\_Write(1);  
 DIG\_OUT\_Pin3\_Write(0);  
 break;  
  
 case 2:  
 DIG\_OUT\_Pin0\_Write(0);  
 DIG\_OUT\_Pin1\_Write(0);  
 DIG\_OUT\_Pin2\_Write(1);  
 DIG\_OUT\_Pin3\_Write(1);  
 break;  
  
 case 1:  
 DIG\_OUT\_Pin0\_Write(0);  
 DIG\_OUT\_Pin1\_Write(0);  
 DIG\_OUT\_Pin2\_Write(0);  
 DIG\_OUT\_Pin3\_Write(1);  
 break;  
  
 case 0:  
 DIG\_OUT\_Pin0\_Write(1);  
 DIG\_OUT\_Pin1\_Write(0);  
 DIG\_OUT\_Pin2\_Write(0);  
 DIG\_OUT\_Pin3\_Write(1);  
 break;  
 }  
 break;  
   
 }   
 phase = (phase + 1) & 7;  
 CyDelay(10);  
 dir = ~dir;  
 }  
   
 //Delay 500 ms  
 CyDelay(500);  
  
 }  
 count = 0;  
}