Project 9 Stepper Motor

โจทย์

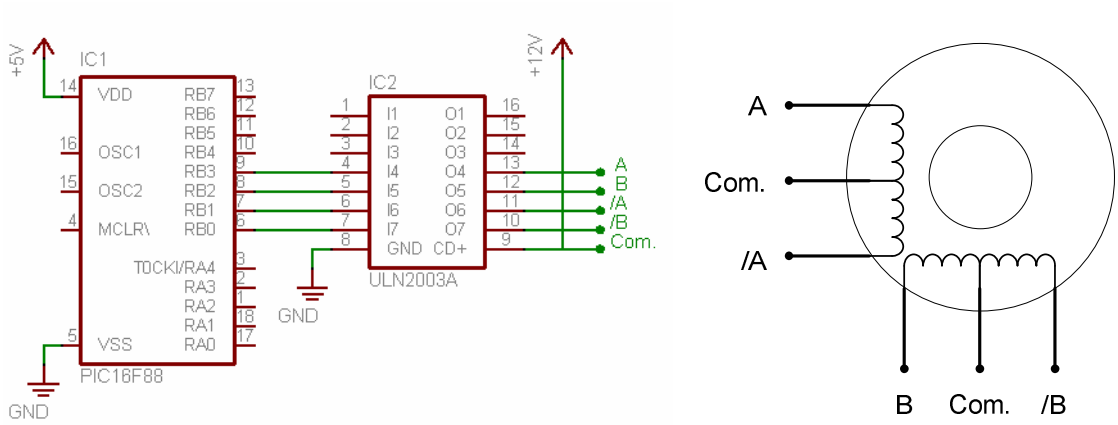
สร้างวงจรและเขียนโปรแกรมสำหรับ PIC16F88 เพื่อควบคุม stepper motor ทั้งแบบ unipolar และ bipolar

ขั้นตอนการปฏิบัติ

1) การออกแบบวงจร

วงจรสำหรับควบคุม Stepper Motor แบบ Unipolar

เนื่องจากมอเตอร์เป็นอุปกรณ์ที่ต้องการกระแสมากเกินกว่าที่ไมโครคอนโทรลเลอร์จะให้ได้ จึงต้องใช้วงจรขับ (driver) ช่วยในการจ่ายกระแสแทน ในโปรเจ็กนี้ใช้ไอซีเบอร์ ULN2003A ซึ่งมีวงจรขับ 7 ช่อง โดยแต่ละช่องสามารถจ่ายกระแสได้สูงสุด 0.5 A (สามารถดูรายละเอียดไอซีได้จาก datasheet) การควบคุมแบบ unipolar จะต้องใช้สัญญาณควบคุม 4 เส้น ซึ่งสามารถออกแบบวงจรได้ดังรูปที่ 1 โดยสาย common ของมอเตอร์จะต่ออยู่กับแรงดัน 12 V



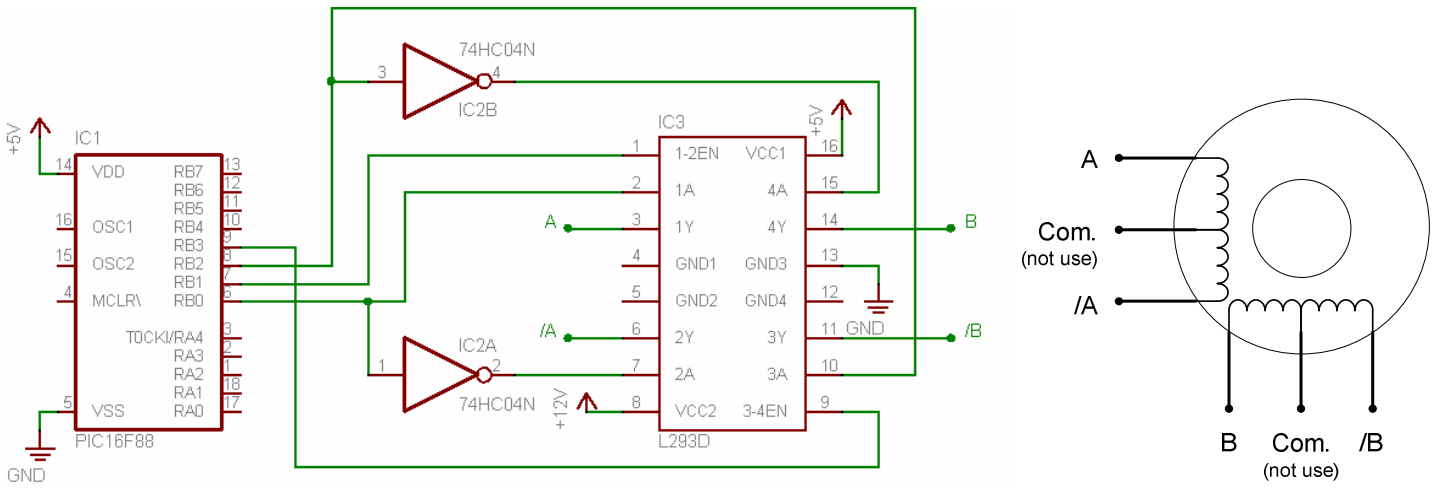
รูปที่ 1 แผนผังวงจรสำหรับใช้ควบคุม Stepper Motor แบบ unipolar

วงจรสำหรับควบคุม Stepper Motor แบบ Bipolar

การต่อสายแบบ bipolar จะไม่ใช้สาย common บนมอเตอร์ ดังนั้นจึงมีเพียงสายสัญญาณ 4 เส้น การควบคุมมอเตอร์จึงต้องใช้วิธีการสลับทิศทางการไหลของกระแส โดยจ่ายไฟบวกและไฟลบสลับกันระหว่างสาย A กับ /A และ ระหว่าง B กับ /B ซึ่งทำให้ต้องใช้วงจร H-bridge เพื่อช่วยในการสลับทิศทางการไหลของกระแส

ในที่นี้เราเลือกใช้ไอซีเบอร์ L293D ซึ่งมี H-bridge 2 ชุด โดยแต่ละชุดสามารถจ่ายกระแสได้สูงสุด 0.5 A H-bridge แต่ละชุดในไอซี L293D จะมีช่องสัญญาณ Enable เพื่อปิดเปิดการจ่ายไฟให้กับวงจรบริดจ์ ส่วนทิศทางการไหลของกระแสสามารถเลือกได้โดยการสลับแรงดันของแต่ละชุดโดยใช้ NOT gate (จากไอซี 74HC04) ดังแสดงในรูปที่ 2 ระดับสัญญาณที่จ่ายให้กับไอซี L293 สามารถสรุปได้ดังตารางต่อไปนี้

1-2EN / 3-4EN	1A / 3A	2A / 4A	**สถานะของวงจร H-bridge**
L	X	X	ไม่มีการจ่ายกระแส
H	L	H	จ่ายกระแสในทิศทางที่ 1
H	H	L	จ่ายกระแสในทิศทางที่ 2



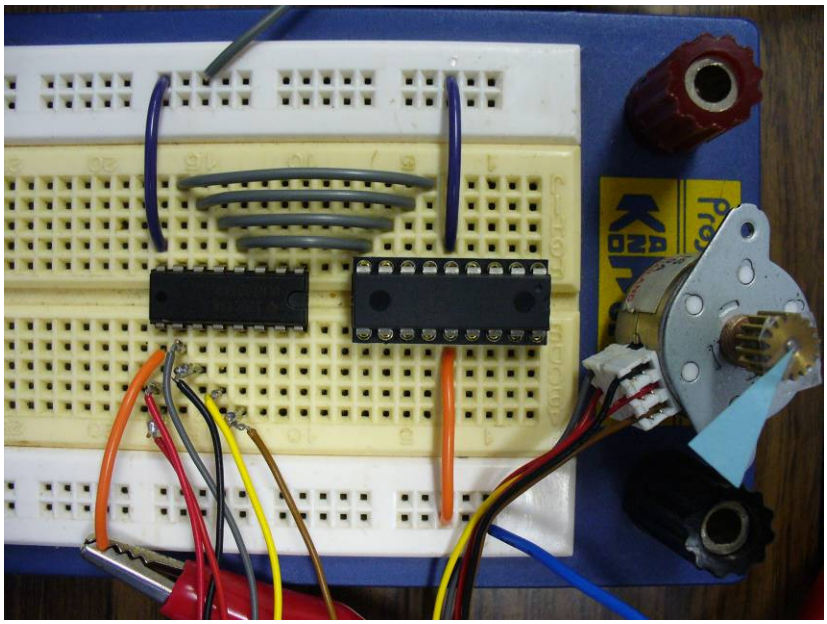
รูปที่ 2 แผนผังวงจรสำหรับใช้ควบคุม Stepper Motor แบบ bipolar

ไมโครคอนโทรลเลอร์ PIC16F88 ถูกต่อเข้ากับวงจรขับโดยใช้ขา RB0-RB3 โดย RB0 และ RB1 เป็นขา direction และ enable ของ H-Bridge ชุดที่ 1 และขา RB2 และ RB3 เป็นขา direction และ enable ของ H-Bridge ชุดที่ 2 โดยที่มอเตอร์จะต่อเข้ากับไอซี L293D ที่ตำแหน่ง A, /A, B, /B อีกทีหนึ่ง ดังแสดงในรูปที่ 2 ในที่นี้มอเตอร์จะถูกขับด้วยแรงดัน 12 V ตามที่จ่ายให้ที่ขา VCC2

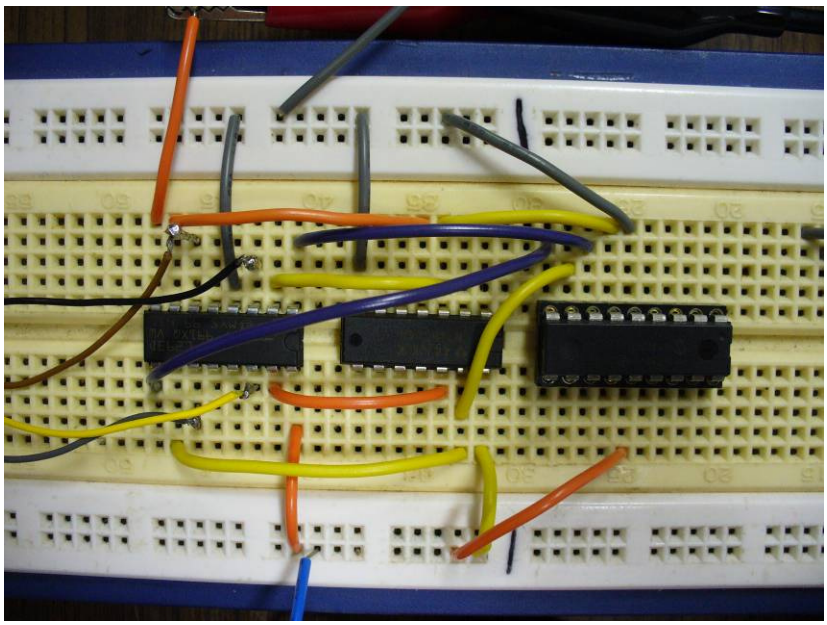
แม้ว่าการต่อมอเตอร์แบบ bipolar จะมีความซับซ้อนกว่าการต่อแบบ unipolar แต่ก็มีข้อดีคือ ทำให้สามารถจ่ายกระแสผ่านขดลวดทั้งสองชุดในเวลาเดียวกันได้ จึงทำให้เกิดสนามแม่เหล็กที่แรงกว่า และทำให้มอเตอร์มีแรงบิดที่มากขึ้นได้

2) การต่อวงจร

รูปที่ 3 และ รูปที่ 4 แสดงการต่อวงจรสำหรับควบคุม stepper motor แบบ unipolar และ bipolar ตามลำดับ



รูปที่ 3 รูปแสดงการต่อวงจรสำหรับใช้ควบคุม Stepper Motor แบบ unipolar



รูปที่ 4 รูปแสดงการต่อวงจรสำหรับใช้ควบคุม Stepper Motor แบบ bipolar

3) การเขียนโปรแกรม

<u>สำหรับแบบ Unipolar</u>

เริ่มด้วยการกำหนดให้พอร์ตที่ต้องการใช้ควบคุมมอเตอร์จำนวน 4 บิต เป็นพอร์ตเอาท์พุท จากนั้นใน main loop ให้ทำการ set หรือ clear สัญญาณของแต่ละพอร์ทตามลำดับของขดลวด ก็จะทำให้มอเตอร์หมุนไปได้เรื่อยๆ

ในกรณีที่ใช้พอร์ต RB0-RB3 เป็นสัญญาณควบคุมซึ่งต่อไปยังสาย A, B, /A, /B ตามลำดับ และมอเตอร์ถูกพันไว้ในลักษณะ unipolar สามารถลำดับสัญญาณที่จ่ายให้สายไฟแต่ละเส้นดังนี้

Step	Supply		PORTB
	+V	Gnd.	
1	Com.	A	0x01
2	Com.	B	0x02
3	Com.	/A	0x04
4	Com.	/B	0x08

เนื่องจากการ set หรือ clear สัญญาณแต่ละครั้งนั้นรวดเร็วมากจนกระแสไฟฟ้าไม่เพียงพอที่จะสร้างสนามแม่เหล็กเหนี่ยวนำแกนหมุน (rotor) ให้หมุนไปได้ จึงจำเป็นจะต้องใส่ฟังก์ชันหรือกระบวนการอื่นๆ เพื่อทำการยืดระยะเวลาระหว่างการ set และ clear สัญญาณแต่ละครั้งออกไปให้ห่างกันพอสมควร ซึ่งเป็นส่วนสำคัญในการกำหนดความเร็วการหมุนของมอเตอร์ด้วยหากระยะเวลาห่างเกินไปมอเตอร์ก็จะหมุนช้ามาก หากระยะเวลาสั้นเกินไปมอเตอร์ก็อาจจะไม่สามารถหมุนได้

ในกรณีที่เราต้องการให้มอเตอร์หมุนไปด้วยขนาดองศาที่เราต้องการ เราจะต้องรู้ว่ามอเตอร์มี stator ทั้งหมดกี่อัน โดยหมุนแกนของมอเตอร์จะรู้สึกได้ว่าเป็นจังหวะ ให้นับจำนวนจังหวะในหนึ่งรอบก็จะได้เท่ากับจำนวน stator หรืออีกวิธีหนึ่งคือเขียนโปรแกรมทดลองเพื่อให้มอเตอร์หมุนไปช้าๆ ก็จะสามารถเห็นจังหวะได้อย่างชัดเจนเช่นกัน ถ้ามี stator 20 อัน ใน 1 รอบการหมุน

ของมอเตอร์แสดงว่าเราสามารถกำหนดให้มอเตอร์หมุนไปด้วยความละเอียด = 360/20 = 18° หรือคือ 1 step ของการหมุนนั่นเอง

หมายเหตุ ในกรณีที่มอเตอร์ไม่มีไฟเลี้ยงอยู่ก่อนแล้วแกนหมุนอาจอยู่ในตำแหน่งระหว่าง stator 2 ตัว ทำให้การควบคุมองศาของมอเตอร์ในครั้งแรกมีความคลาดเคลื่อนไป 1 step (หากเป็นมอเตอร์ในตัวอย่างคือผิดไป 18°)

โปรแกรม project09a.c

```
#include<pic.h>
#include<pic.h>
#include "delay.h"

__CONFIG(UNPROTECT & LVPDIS & BOREN & MCLRDIS & PWRTDIS & WDTDIS & INTIO );

void main (void)
{
   int i;
   // Setup oscillator
   IRCF2   =  1;
   IRCF1 =  1;
   IRCF0 =  1;              // IRCF<2:0> = 111: internal RC oscillator frequency = 8 MHz
   SCS1  =  0;
   SCS0  =  0;              // SCS<1:0> = 00: Oscilator mode defined by FOSC<2:0>
   while(!IOFS);            // Wait for the frequency to become stable

   ANSEL = 0x00;            // Select every bit of port A as digital I/O
   TRISB = 0X00;            // Set port B as output.
   PORTB = 0X00;            // Clear data in port B before use.
   for(i=0;i<5;i++)         // 5 round of 4 steps = 1 round of motor's rotation.
   {
      PORTB = 0x01;         // Step 1 gennerate pulse to A
      DelayMs(100);
      PORTB = 0x02;         // Step 1 gennerate pulse to B
      DelayMs(100);
      PORTB = 0x04;         // Step 1 gennerate pulse to /A
      DelayMs(100);
      PORTB = 0x08;         // Step 1 gennerate pulse to /B
      DelayMs(100);
   }
   while(1);
}
```

คำอธิบายโปรแกรม

โปรแกรม project09a.c ประกอบด้วยการ set และ clear port ที่ port B ตามลำดับเช่นในตารางข้างต้น ทั้งนี้มอเตอร์ที่ใช้ในการทดลองครั้งนี้เป็นสเต็ปเปอร์มอเตอร์ที่มี stator 20 ตำแหน่ง ดังนั้นการวนลูปทั้งหมด 5 รอบจะทำให้มอเตอร์หมุนครบ 1 รอบพอดี โดยมีการกำหนดให้หน่วงเวลาระหว่างสเต็ปเป็นเวลา 100 ms จากนั้นจะเข้าสู่ infinity loop เพื่อไม่ให้โปรแกรมวนกลับไปเริ่มทำงานใหม่อีกครั้ง

สำหรับแบบ Bipolar

เนื่องจากเราต่อวงจรเพื่อควบคุมการหมุนโดยใช้สัญญาณ enable และ direction แทนการควบคุมสัญญาณที่ขา A, /A, B, /B โดยตรง ทำให้มีลำดับการส่งสัญญาณควบคุมเป็นดังตารางต่อไปนี้

Step	L293D				PORTB
	A direction (RB0)	A enable (RB1)	B direction (RB2)	B enable (RB3)	
1	1	1	X	0	0x03
2	X	0	1	1	0x0C
3	0	1	X	0	0x02
4	X	0	0	1	0x08

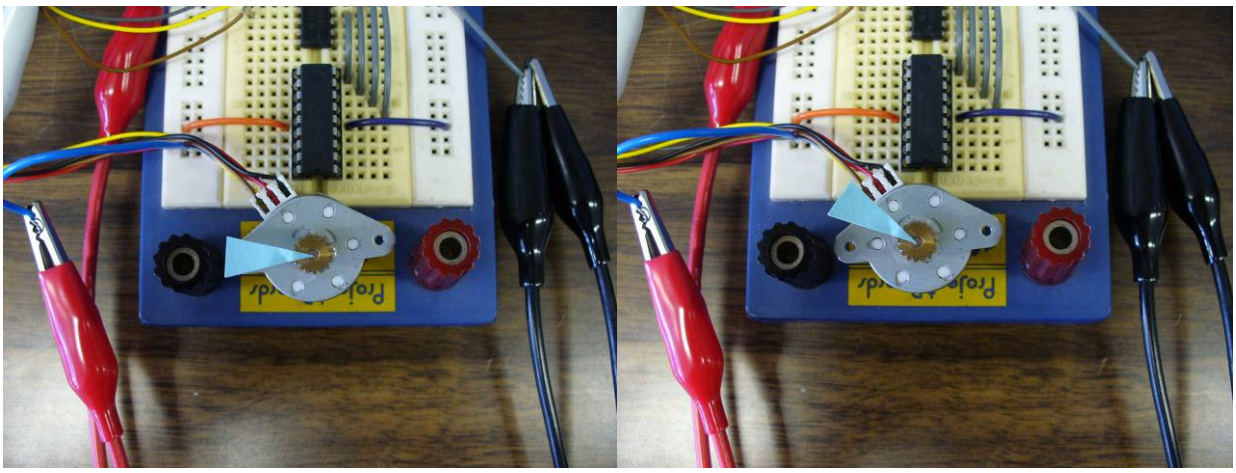
โปรแกรม project09b.c

```
#include<pic.h>
#include "delay.h"

__CONFIG(UNPROTECT & LVPDIS & BOREN & MCLRDIS & PWRTDIS & WDTDIS & INTIO );

static bank1 unsigned int an[4];

void main (void)
{
   int i;
   // Setup oscillator
   IRCF2   =  1;
   IRCF1 =  1;
   IRCF0 =  1;              // IRCF<2:0> = 111: internal RC oscillator frequency = 8 MHz
   SCS1  =  0;
   SCS0  =  0;              // SCS<1:0> = 00: Oscilator mode defined by FOSC<2:0>
   while(!IOFS);            // Wait for the frequency to become stable

   ANSEL = 0x00;            // Select every bit of port A as digital I/O
   TRISB = 0X00;            // Set port B as output.
   PORTB = 0X00;            // Clear data in port B before use.
   for(i=0;i<5;i++)
   {
      PORTB = 0x03;         // Step 1 gennerate current from A to /A
      DelayMs(100);
      PORTB = 0x0C;         // Step 2 gennerate current from B to /B
      DelayMs(100);
      PORTB = 0x02;         // Step 3 gennerate current from /A to A
      DelayMs(100);
      PORTB = 0x08;         // Step 4 gennerate current from /B to B
      DelayMs(100);
   }
   while(1)
   {}
}
```

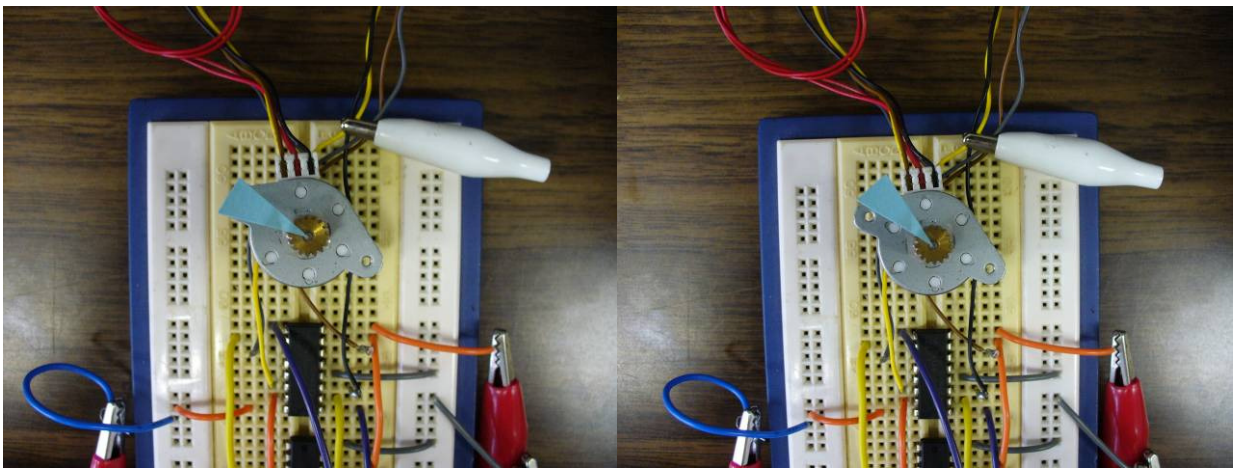
4) การทดสอบการทำงาน

ต่อสายสัญญาณจากมอเตอร์ให้ตรงตามคู่ สามารถต่อสลับกันระหว่างชุด A, /A กับ B, /B ได้ แต่จะทำให้การหมุนกลับทิศทางกัน และสามารถต่อสลับกันภายในชุด คือ A กับ /A และ B กับ /B ได้ โดยจะมีผลในช่วงแรกของการหมุนเท่านั้นที่จะทำให้ผิดจังหวะไปเล็กน้อย เมื่อวงจรเริ่มทำงานแล้วจะได้ผลดังรูปที่ 5 และ รูปที่ 6

ข้อควรระวัง เนื่องจากมอเตอร์แต่ละตัวอาจใช้แรงดันไฟฟ้าแตกต่างกัน ทั้งนี้ขึ้นอยู่กับรุ่นของมอเตอร์ที่ใช้ จึงควรตรวจสอบแรงดันไฟฟ้าให้ถูกต้องก่อนทำการจ่ายไฟ



รูปที่ 5 รูปแสดงผลการทดลองต่อวงจรควบคุมมอเตอร์แบบ unipolar



รูปที่ 6 รูปแสดงผลการทดลองต่อวงจรควบคุมมอเตอร์แบบ bipolar

จากรูปผลการทดลองจะเห็นว่าตำแหน่งการหมุนของมอเตอร์ที่คำนวณไว้สำหรับ 1 รอบ ไม่หมุนจนครบรอบพอดี เนื่องจากการเขียนโปรแกรมที่คำนวณตามจำนวนรอบพอดีทำให้เสียไปหนึ่งจังหวะการหมุนดังที่ได้กล่าวไปแล้ว หากเราเขียนโปรแกรมเพื่อให้หมุนหลายรอบ มอเตอร์ก็จะกลับมายังตำแหน่งดังรูปขวามือทุกครั้งเช่นกัน