

# RMXplorer

## LAB2 : Brushed DC Motor, Stepper Motor and Brushless DC Motor Report

---

### สมาชิก

- นายกิตติภณ วงศ์เลขา 66340500003
- นางสาวรัณณุณภัส เพียรชูพัฒน์ 66340500020
- นายธีร์รัช กลมหาภัย 66340500021

### วัตถุประสงค์

- เพื่อให้เข้าใจถึงความแตกต่างของมอเตอร์ในแต่ละประเภท ได้แก่ Brushed DC Motor, Stepper Motor และ Brushless DC Motor ทั้งหมด 3 ประเภท
- เพื่อให้เข้าใจถึงลักษณะของมอเตอร์ทั้ง 3 ประเภท ในเรื่องของ Speed, Torque, Current, Power และ %Efficiency
- เพื่อให้เข้าใจถึงหลักการทำงานของมอเตอร์ทั้ง 3 ประเภท ในรูปแบบของ Full-Step และ Half-Step เพื่อควบคุมความเร็วและตำแหน่ง
- เพื่อให้สามารถออกแบบการทดลองเพื่อหาผลการทดลองที่สามารถตอบคำถามกับผลการเรียนรู้ย่อได้ ผ่านการใช้อุปกรณ์ที่ได้เรียนรู้รายในภาคเรียน เช่น MATLAB, Simulink เป็นต้น
- เพื่อให้สามารถใช้หลักการทางวิทยาศาสตร์เพื่อทำการทดลองผ่านการกำหนดตัวแปรต้น ตัวแปรตาม ตาม วิธีการดำเนินการทดลอง รวมไปถึงการบันทึกผลการทดลอง การสรุปผล โดยมีการใช้ทฤษฎี เพื่ออ้างอิงถึงหลักการต่าง ๆ และผลลัพธ์ที่ได้ต้องสอดคล้องกับตัวแปร และสมมติฐานที่กำหนดไว้
- เพื่อให้สามารถเขียนโปรแกรมผ่านการใช้ MATLAB และ Simulink ส่งไปควบคุมสั่งการกับบอร์ด Nucleo STM32G474RE

## การทดลองที่ 1 DC Motor with WCS1700 Hall Current Sensor

### จุดประสงค์

- เพื่อศึกษา และเข้าใจถึงหลักการทำงาน รวมถึงความสามารถของ Motor-Torque Constant และ Back-EMF Constant ของ DC Motor
- เพื่อศึกษา และเข้าใจถึงความสัมพันธ์ที่สอดคล้องกันของ Speed, Torque, Current, Power และ %Efficiency เมื่อมีการเปลี่ยนแปลงของ Magnetic Particle Clutches เมื่อปรับ Load Torque ที่กระทำต่อ DC Motor และ การเปลี่ยนแปลงของแรงดันไฟฟ้า เมื่อปรับ Duty Cycle และ Frequency ของ PWM ที่จ่ายเข้า DC Motor
- เพื่อศึกษา และเข้าใจถึงหลักการทำงานของ H-Bridge Drive Mode ทั้ง 3 Mode ได้แก่ Sign-Magnitude, Locked Anti-Phase และ Async Sign-Magnitude
- เพื่อศึกษา และเข้าใจถึงการควบคุม DC Motor ทั้ง 2 Mode ได้แก่ Sign-Magnitude และ Locked Anti-Phase
- เพื่อเข้าใจ และเข้าใจถึงกระบวนการเริ่มต้นของ Signal Conditioning และ Signal Processing เพื่ออธิบายถึง ที่มาของค่าจาก Incremental Encoder และ Hall Current Sensor รวมถึงวิธีการคำนวณ ขั้นตอนการทำ ทั้งหมดตั้งแต่ก่อนและ หลังการ Calibrate Sensor
- เพื่อศึกษา และเข้าใจถึงการหาสมการความสัมพันธ์ระหว่างกระแสไฟฟ้ากับแรงดันไฟฟ้าที่ออกจาก Hall Current Sensor และการอธิบายถึงการ Unwrap ค่า
- เพื่อศึกษา และเข้าใจถึงการเขียนโปรแกรมผ่านการใช้ MATLAB และ Simulink ส่งไปควบคุมสั่งการกับบอร์ด Nucleo STM32G474RE โดยรับค่า Input จากสัญญาณของ Incremental Encoder และ Hall Current Sensor เพื่อแสดงค่า Output เป็นกราฟการ Log สัญญาณ ใน Data Inspector ในโปรแกรม MATLAB Simulink เพื่อให้เห็นถึงค่า Output ที่ปรับแต่งตามค่า Output แบบ Real Time โดยกำหนดให้ Output เป็น ความเร็วเชิงมุม และกระแสไฟฟ้าในหน่วย SI derived

### สมมติฐาน

1. การใช้ Cytron MDD20A Motor Driver เป็น H-Bridge Drive นั้นสามารถทำให้มอเตอร์เลือกทิศทางในการหมุน ได้และยังสามารถเปลี่ยนโหมดในการควบคุมระหว่าง Lock Anti Phase และ Sign Magnitude
2. หากเพิ่มแรงต้านการหมุนให้กับมอเตอร์ มอเตอร์จะกินกระแสเพิ่มมากขึ้น และหากถ้าแรงหมุนกลับนั้นมากเกิน กว่าที่แรงบิดสูงสุดของมอเตอร์สามารถทำได้ มอเตอร์จะหยุดหมุน
3. การทำงานในโหมด Lock Anti Phase ของ Cytron MDD20A Motor Driver นั้นจะทำให้มอเตอร์หยุดนิ่งเร็วกว่า การเลือกใช้โหมด Sign Magnitude ที่มอเตอร์จะหยุดอย่างช้า ๆ ในกรณีที่เราสั่งให้มอเตอร์หมุนและหยุดการทำงาน (Off Time)

4. การใช้ PWM ที่มีร้อยละของ Duty Cycle สูงจะทำให้มอเตอร์มีแรงบิดและความเร็วที่มากขึ้นอย่างแปรผันตรงกัน
5. การหาค่าคงที่ ความเร็วตอบสนองไม่มีโหลด, แรงบิดสูงสุด, กระแสสูงสุด, กระแสขณะไม่มีโหลด และแรงดันไฟฟ้าที่จ่าย จะสามารถทำให้ผู้จัดทำหาจุดที่มีประสิทธิภาพที่สุดของระบบได้และยังสามารถสร้างกราฟสมการ Motor Characteristic ได้

## ตัวแปร

ตัวแปรต้น : Load และ Duty Cycle ที่เปลี่ยนไป

ตัวแปรตาม : ความเร็ว แรงบิด และกระแสไฟฟ้า

ตัวแปรควบคุม : แรงดันไฟฟ้าที่จ่ายเข้าบอร์ด, ชนิดของสายจัมเปอร์ที่ใช้เชื่อมต่อสายไฟ, ชนิดของบอร์ด Microcontroller และชนิดของ DC Motor

## นิยามคัพเพิลเจพะ

Rotor หมายถึง ส่วนที่หมุนของมอเตอร์ไฟฟ้า โดยมักประกอบด้วยชุดลวดหรือแม่เหล็กถาวรซึ่งทำหน้าที่สร้างแรงบิดเมื่อเกิดปฏิกิริยาแม่เหล็กของ Stator

Stator หมายถึง ส่วนที่อยู่กับที่ของมอเตอร์ไฟฟ้า ทำหน้าที่สร้างสนามแม่เหล็กโดยการจ่ายกระแสไฟฟ้าผ่านชุดลวดในโครงสร้างที่วางเรียงอยู่รอบ Rotor

Signal Conditioning หมายถึง การเปลี่ยนแปลงสัญญาณผ่านหลักการ เช่น Filtering, Amplification, Linearization ฯลฯ

Stall Torque หมายถึง แรงบิดที่มอเตอร์สามารถทำได้สูงสุด

No load หมายถึง สถานะของมอเตอร์เมื่อไม่มีแรงเสียดทานมาต้านการทำงานของมอเตอร์ที่หมุน

## นิยามเชิงปฏิบัติการ

DC Motor หมายถึง Nidec Components Geared DC Geared Motor, 12 V dc, 20 Ncm, 70 rpm, 6mm Shaft Diameter

Encoder หมายถึง Incremental Encoder AMT103-V

ไม่ หมายถึง ชิ้นส่วนพลาสติกชิ้นรูปจากเครื่องพิมพ์สามมิติ ที่รูปทรงเหมือนก้านไม้ที่ปลายมีชิ้นส่วนแบบตั้งฉากออกมากจากแกนหลักของชิ้นส่วน มีไว้สำหรับส่งแรงจากมอเตอร์ไปกด Load Cell

## เอกสารและงานวิจัยที่เกี่ยวข้อง

### 1. DC Motor Characteristic

DC motor มี characteristic ห้าหมวด 5 ค่า ได้แก่ Speed, Current, Efficiency, Power out และ Torque

1.1 Speed หรือความเร็วของมอเตอร์จะแปรผกผันกับ torque ในกรณีที่ไม่มี Load ความเร็วที่สามารถทำได้ของมอเตอร์จะมีมากที่สุด กลับกันหากมี Load มากเกินไป มอเตอร์จะไม่สามารถหมุนเพื่อสร้างความเร็วได้

1.2 Current หรือกระแสไฟฟ้าที่ผ่านมอเตอร์จะเพิ่มขึ้นตาม Torque ที่ต้องการ หากมี Load เพิ่มขึ้น กระแสจะเพิ่มผ่านจะเพิ่มขึ้นเพื่อสร้างแรงแม่เหล็กไฟฟ้าเพียงพอที่จะขับเคลื่อน Load เมื่อไร Load กระแสจะต่ำมาก เนื่องจากมอเตอร์ใช้เพียงกระแสเดือนอยู่เพื่ออาจน้ำความฝืดภายนอก

1.3 Efficiency หรือประสิทธิภาพของมอเตอร์คือสัดส่วนของกำลังงานกลที่ออกมานะกับกำลังงานไฟฟ้าที่ป้อนเข้า ประสิทธิภาพจะเพิ่มขึ้นตาม Load ในช่วงหนึ่ง เพราะพลังงานที่ป้อนเข้าจะถูกใช้เป็นงานที่มีประโยชน์มากขึ้นแต่เมื่อ Load สูงเกินไป ประสิทธิภาพจะลดลง เนื่องจากการสูญเสียพลังงาน เช่น ความร้อนจากความต้านทานที่เกิดขึ้นในมอเตอร์

1.4 Power หรือกำลังไฟฟ้าที่มอเตอร์สร้างขึ้นเท่ากับผลคูณของแรงบิดและความเร็ว กำลังงานที่ออกมานะเพิ่มขึ้นเมื่อ Load เพิ่มขึ้น จนถึงจุดที่มอเตอร์ริมเข้าใกล้จุดหยุดนิ่ง (Stall) ซึ่งความเร็วจะลดลงจนเหลือศูนย์

1.5 Torque หรือแรงบิดของมอเตอร์มีความสัมพันธ์โดยตรงกับกระแสที่มอเตอร์ใช้ Load ที่มากขึ้นจะต้องแรงบิดที่สูงขึ้น ดังนั้น กระแสที่ใช้จะเพิ่มขึ้นตาม หากแรงบิดสูงเกินไป มอเตอร์อาจหยุดการทำงาน (Stall)

### 2. Locked Anti-Phase

คือการ Drive มอเตอร์โดยการใช้สัญญาณ PWM ที่มีเฟสตรงกันข้าม (complementary signals) ขับทั้งสองฝั่งของ H-bridge เช่น ถ้าฝั่งหนึ่งได้ PWM Duty Cycle = 70% อีกฝั่งจะได้ PWM Duty Cycle = 30% (100% - 70%) มอเตอร์จะถูกขับไปข้างหน้าหรือถอยหลังขึ้นอยู่กับ Duty Cycle โดย 50% จะเป็นตำแหน่ง หยุด เนื่องจากแรงดันสองด้านเท่ากัน

### 3. Sign-Magnitude

คือการ Drive มอเตอร์โดยการใช้สัญญาณ PWM ขับเฉพาะฝั่งหนึ่งของ H-bridge ในขณะที่อีกฝั่งจะอยู่ในสถานะคงที่ (High หรือ Low) ใช้สัญญาณ Logic แยกอีกตัวหนึ่งเพื่อกำหนดทิศทาง (Direction) ของมอเตอร์ เช่น HIGH = ไปข้างหน้า LOW = ย้อนกลับ ควบคุมเฉพาะขนาดแรงดันที่ส่งไปยังมอเตอร์

## วิธีดำเนินการทดลอง

การทดลองเรื่อง DC Motor with WCS1700 Hall Current Sensor สามารถแบ่งการทดลองออกเป็น 3 ส่วน ส่วนแรก คือ การเตรียมอุปกรณ์ และโปรแกรมในการทำการทดลอง เช่น MATLAB และ Simulink เป็นต้น ส่วนที่สอง คือ การออกแบบการทดลอง และทำการทดลอง ซึ่งคือการ Calibrate Hall Effect Current Sensor, Fast Fourier Transform เป็นต้น ส่วนที่สาม คือ การเก็บบันทึกผล และวิเคราะห์ข้อมูลที่ได้ เพื่อมาทำกราฟ Motor Characteristic

## วัสดุอุปกรณ์

1. Nidec Components Geared DC Geared Motor, 12 V dc, 20 Ncm, 70 rpm, 6mm Shaft Diameter จำนวน 1 อัน
2. Incremental Encoder AMT103-V จำนวน 1 อัน
3. WCS1700 Hall Current Sensor จำนวน 1 อัน
4. Cytron MDD20A Motor Driver จำนวน 1 อัน
5. Nucleo STM32G474RE พร้อมสายอัปโหลด จำนวน 1 ชุด
6. MotorXplorer จำนวน 1 ชุด - ฐานสามารถบรรจุอิฐวงจรรวม, Breadboard, 3D-Print ใช้สำหรับการประกอบ กับ DC Motor

## ขั้นตอนการดำเนินงาน

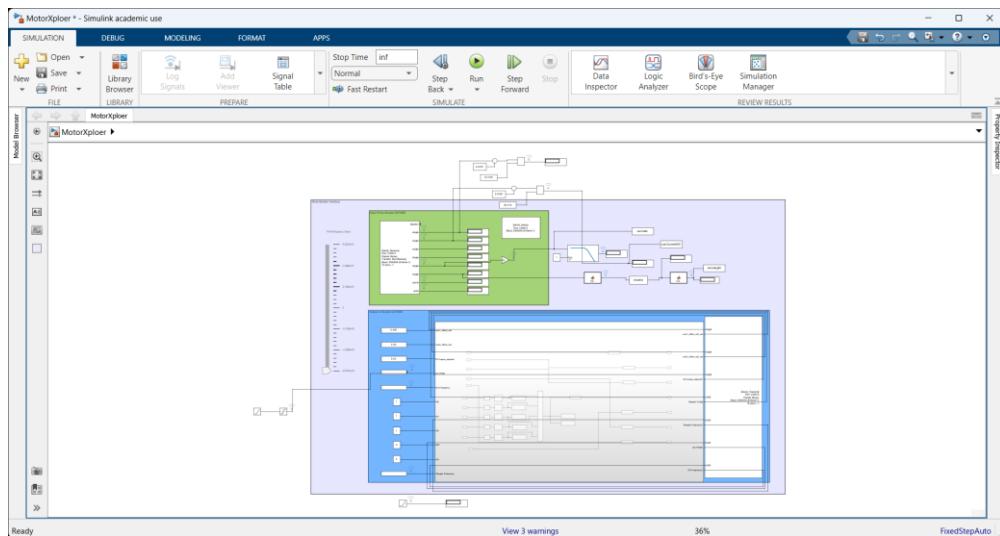
1. เชื่อมต่อ Nucleo STM32G474RE เข้ากับบอร์ด MotorXplorer และทำการจ่ายไฟ 24 V และเปลี่ยนรูปแบบ การใช้ไฟเป็น E5V
2. จ่ายไฟให้ DC Motor โดยใช้ Power Supply ที่จ่ายแรงดัน 15 V และกระแส 10 A
3. เชื่อมต่อสายไฟของ Sensor ต่างๆดังนี้
  1. Incremental Encoder AMT103-V
    - 5V -> 3V3
    - GND -> GND
    - A -> PA6
    - B -> PA7
  2. Magnetic Encoder
    - SCL -> PB8

- SDA -> PB9
- DIR -> GND
- VCC -> 3V3
- GND -> GND

### 3. Load Cell

- OUT -> A3

### 4. เข้า Simulink ใน MATLAB เพื่อทำการเปิดไฟล์สำหรับควบคุมบอร์ด MotorXploer.slx



รูปที่ 1 แสดงหน้าตาไฟล์ MotorXploer.slx

5. ติดตั้ง Extension ที่ชื่อ Waijang ลงใน Matlab
6. ทำการอัพโหลดไฟล์ bin ลงใน Nucleo STM32G474RE จากนั้นเลือก Com Port เพื่อการส่งข้อมูลให้ถูกต้อง
7. ปรับปรุงการรับค่าสัญญาณจาก WCS1700 Hall Current Sensor ผ่านขั้นตอน ดังนี้
  - 7.1 ทำ Signal Conditioning ซึ่งแต่ละ Sensor ก็จะมีค่าที่อ่านได้ไม่ตรงกัน จึงต้องใช้กราฟเส้นที่สามารถวัดได้จริงตั้งแต่ 0 A ไปจนถึง 3.2 A และวัดค่า ADC ที่ Sensor อ่านได้เพื่อสร้างสมการเชิงเส้น โดยการคำนวณหา ADC สามารถทำได้ตามสมการ ดังนี้

$$ADC = m \times Real\ Current + c$$

เมื่อ

$$ADC = \text{สัญญาณ Analog (Bit)}$$

$m$  = ค่าความชันของสมการเส้นตรง

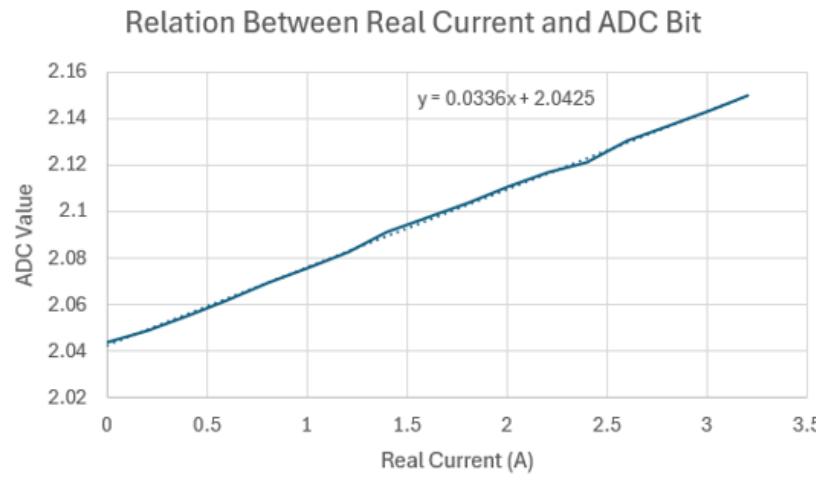
*Real Current* = กระแสที่ใช้จริง (A)

$c$  = ค่าที่เปลี่ยนเบนคงที่

จากนั้นจึงทำการย้ายข้างสมการเพื่อหากระแสไฟฟ้าที่ใช้จริงจะได้สมการ ดังนี้

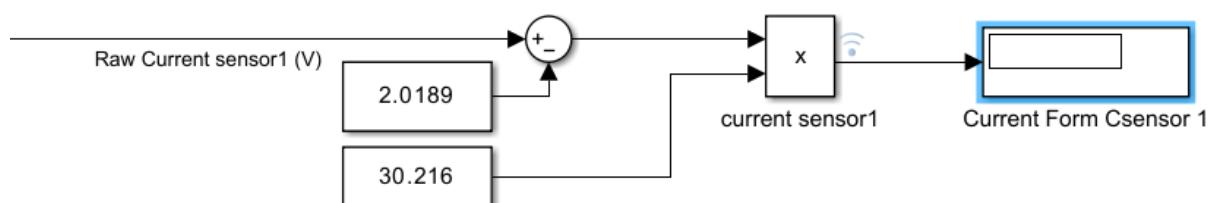
$$\text{Real Current} = \frac{(ADC - 2.0425)}{0.0336}$$

Real Current	ADC Value
0	2.0437
0.2	2.0485
0.4	2.0552
0.6	2.062
0.8	2.0691
1	2.0758
1.2	2.0824
1.4	2.0911
1.6	2.0973
1.8	2.1038
2	2.1103
2.2	2.117
2.4	2.1215
2.6	2.1303
2.8	2.1366
3	2.1432
3.2	2.1497



รูปที่ 2 แสดงการหาสมการเส้นตรงด้วยโปรแกรม Excel

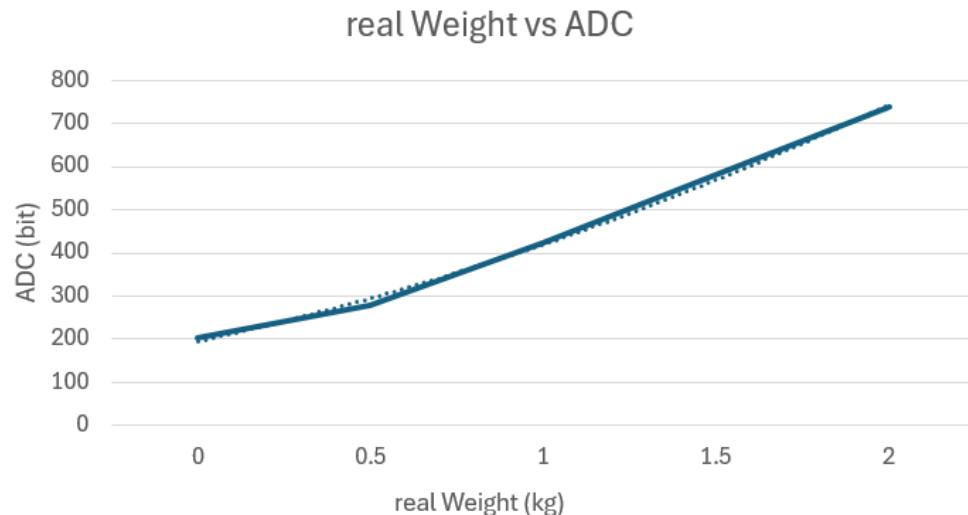
และการสร้าง Block สมการใน Simulink



รูปที่ 3 แสดง Block สมการเพื่อสร้างสมการเส้นตรงใน Simulink

7.2 Single Point Load Cell YZC-131A สามารถทำวิธีเดียวกันกับ Current Sensor ได้ แต่การทดลองดูค่า น้ำหนักมาตรฐานที่คณะผู้จัดทำรู้อยู่แล้ว กับค่า ADC จะเห็นว่ามีความสัมพันธ์ไม่เป็นสมการเชิงเส้น จึง

ต้องใช้สมการ Polynomial (สมการที่ตัวแปรมีมากกว่า หนึ่งดีกรี) เพื่อช่วยในการแปลงค่า ADC ให้เป็นน้ำหนักที่สามารถอ่านได้จาก Load Cell



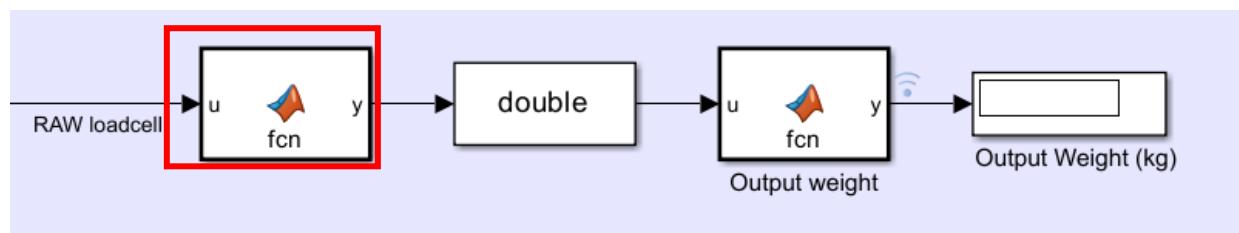
รูปที่ 4 แสดงกราฟระหว่างน้ำหนัก และค่า ADC ที่ถูกแปลงด้วยการใช้สมการ Polynomial

โดยสมการที่ได้คือ

$$Weight = (-2.2389 \times 10^{-6}) ADC^2 + 0.005698 ADC - 1.0055$$

นำสมการที่ได้มาใส่ใน MATLAB Function Block ใน Simulink

```
function y = fcn(u)
y = (- 2.2389*10^ -6) * u^2 + 0.005698*u - 1.0055;
end
```



รูปที่ 5 แสดงหน้าตา MATLAB Function Block ที่ต้องใส่สมการ Polynomial

7.3 Incremental Encoder AMT103-V เนื่องจากในไฟล์ MotorXploer.slx ทำการ QEI และ Unwarp ค่ามาแล้ว จึงสามารถอ่านค่า ความเร็วรอบ (Rad/s) ที่ออกมาจาก Incremental Encoder ได้เลย แต่

เนื่องจากสัญญาณที่ได้ยังมีสัญญาณรบกวนอยู่ จึงต้องทำการตัดสัญญาณที่เราไม่ได้สนใจออกโดยใช้ Varying Lowpass Filter โดยสามารถหาความถี่ของสัญญาณที่เราต้องการได้จากการสั่งงาน Motor ด้วยความถี่ 1 KHz และนำค่าความเร็ว Rad/s ที่อ่านได้ไปเข้าสูตรใน MATLAB โดยใช้คำสั่ง fft() เพื่อหา Frequency ที่มีในสัญญาณที่อุปกรณ์

```

signal = out.simout; % Assuming the second column contains the signal
Fs = 1000; % Sampling frequency (Hz)
T = 1 / Fs; % Sampling period
L = length(signal); % Length of signal
t = (0:L-1) * T; % Time vector

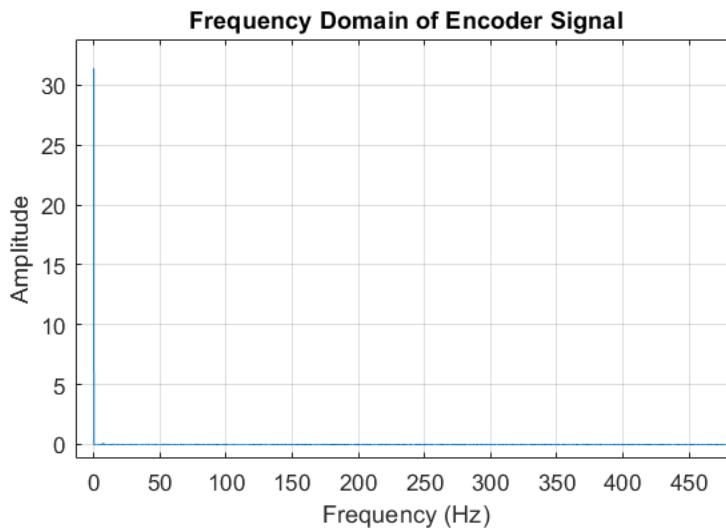
% Perform FFT
Y = fft(signal);
P2 = abs(Y / L); % Two-sided spectrum
P1 = P2(1:L/2+1); % Single-sided spectrum
P1(2:end-1) = 2 * P1(2:end-1); % Adjust for symmetry
f = Fs * (0:(L/2)) / L; % Frequency vector

% Plot Frequency Spectrum
figure;
plot(f, P1);
title('Frequency Domain of Encoder Signal');
xlabel('Frequency (Hz)');
ylabel('|P1(f)|');
grid on;

```

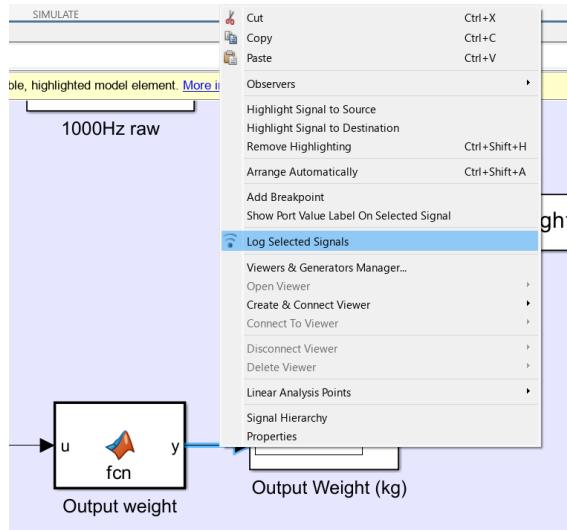
รูปที่ 6 แสดง Code การใช้ฟังก์ชัน Fast Fourier Transform และสร้างกราฟเพื่อดู Amplitude ในความถี่แต่ละช่วง

เราจะได้กราฟแสดง Frequency Domain ที่จะเห็นว่าช่วงความถี่ที่ สูงกว่า 1 Hz ขึ้นไปนั้นไม่จำเป็นต้องสนใจจึงสามารถใช้ Low Pass Filter ที่มี cutoff frequency อยู่ที่ 1 Hz ได้



รูปที่ 7 แสดงการกรองสัญญาณด้วย Fast Fourier Transform เพื่อแสดงให้เห็นถึงช่วงความถี่ของสัญญาณที่แท้จริง

## 8 เก็บข้อมูลเบื้องต้นด้วยการ Log Signal



รูปที่ 8 แสดงการ Log Signal ใน Simulink

8.1 เก็บค่าสัญญาณความเร็วรอบ (Rad/s) จาก Incremental Encoder

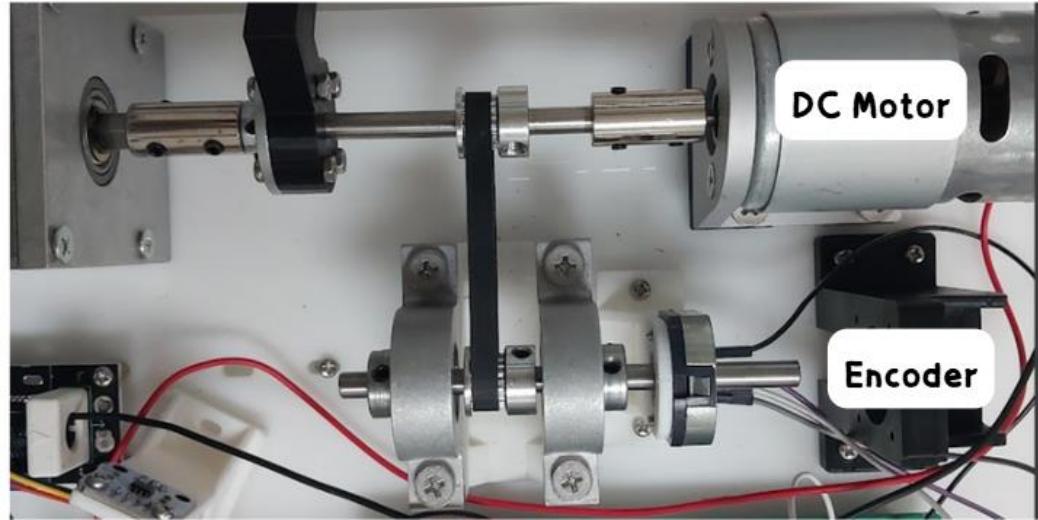
8.2 เก็บค่าสัญญาณกระแสไฟฟ้า (Current) ที่อ่านได้จาก Hall Current Sensor

8.3 เก็บค่าน้ำหนักหรือ Torque จาก Load Cell

8.4 เก็บค่าความถี่ PWM ที่สั่ง DC Motor

## 9 วัดความเร็วของ DC Motor และกระแสไฟฟ้า

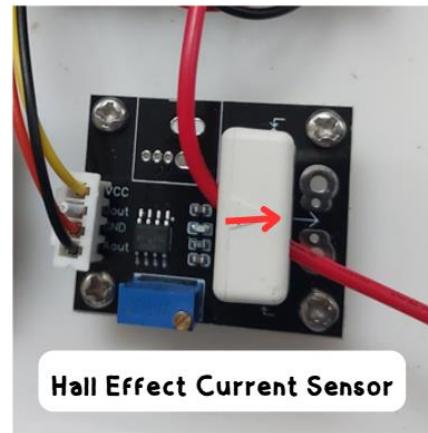
9.1 เชื่อมต่อ DC Motor กับ Pulley และเพลลา Incremental Encoder



รูปที่ 9 แสดงหน้าตาของ DC Motor และ Encoder

9.2 สั่ง PWM จากค่าลับสูงสุดจนถึง 0 และจาก 0 ไปจนถึงค่าบวกสูงสุด

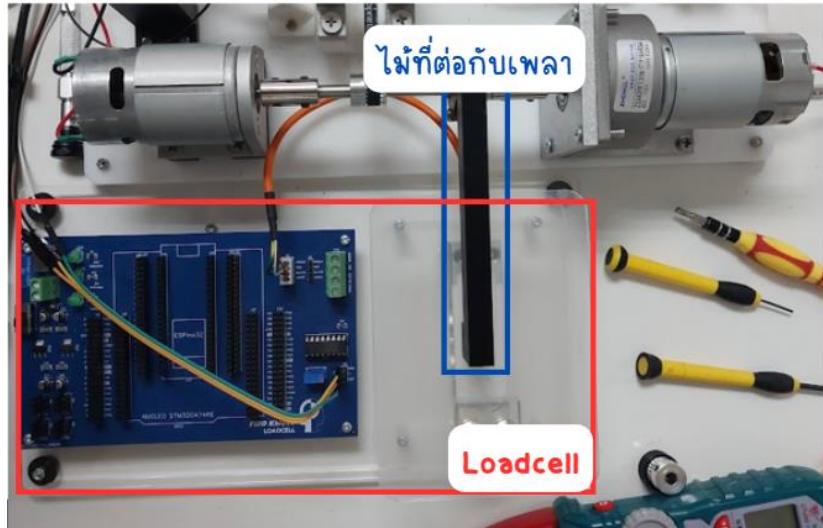
9.3 วัดค่า Speed (Rad/s) และ Current (A) ในทุกค่าของ PWM โดยนำสายไฟขึ้บวงก์ที่เข้า Motor ผ่าน  
ด้านที่มีลูกศรบน Hall Effect Current Sensor



รูปที่ 10 แสดงการต่อสายไฟขึ้บวงก์ผ่าน Hall Effect Current Sensor

10. วัด Torque และกระแสไฟฟ้า (Current)

10.1 ติดตั้งไม้ที่เพลา DC Motor เพื่อกด Load Cell



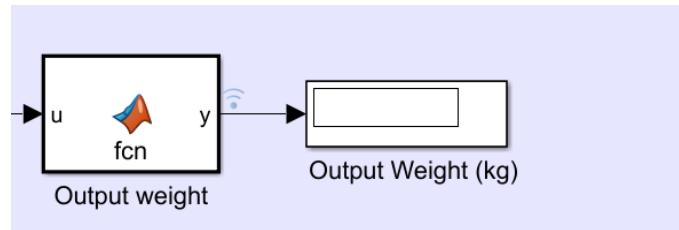
รูปที่ 11 แสดงหน้าตา Loadcell และไม้ที่ต่อ กับ เพลา

10.2 สั่ง PWM Frequency จาก 0 ถึงค่าบวกสูงสุดในทิศทางที่ไม่กด Load Cell



รูปที่ 12 แสดง Ramp Block ที่ต่อเข้า DC PWM

10.3 วัดค่า Torque และ Current ที่แต่ละค่าของ PWM



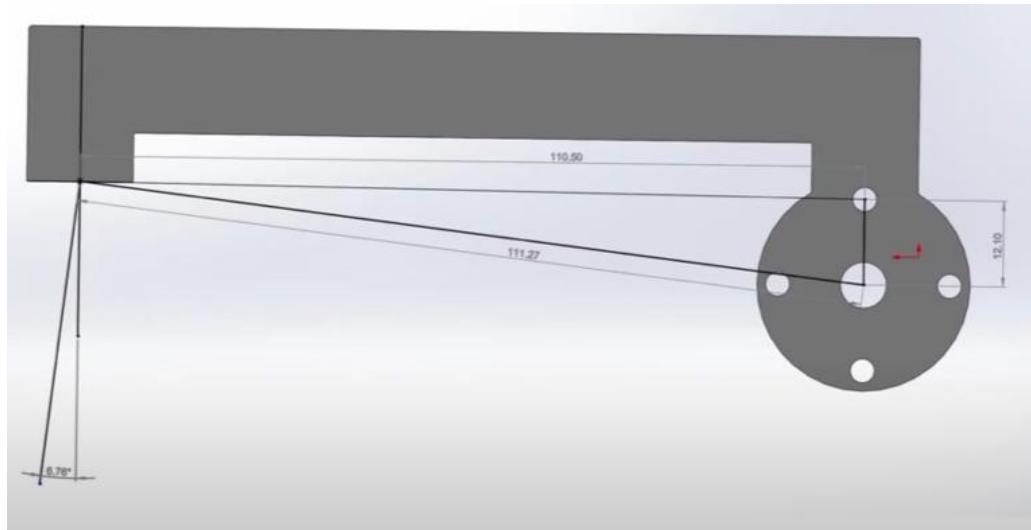
รูปที่ 13 แสดงการต่อสมการ MATLAB Function Block เพื่อแสดง Output Weight ใน Display Block

นำ Output Weight (kg) มาทำเป็น Torque โดยเริ่มจากเปลี่ยนกิโลกรัมเป็นนิวตัน โดยการคูณแรงโน้มถ่วงกับน้ำหนัก Output Weight (kg) ที่กดจากไม้ที่กดลงบน Load Cell

$$F = mg$$

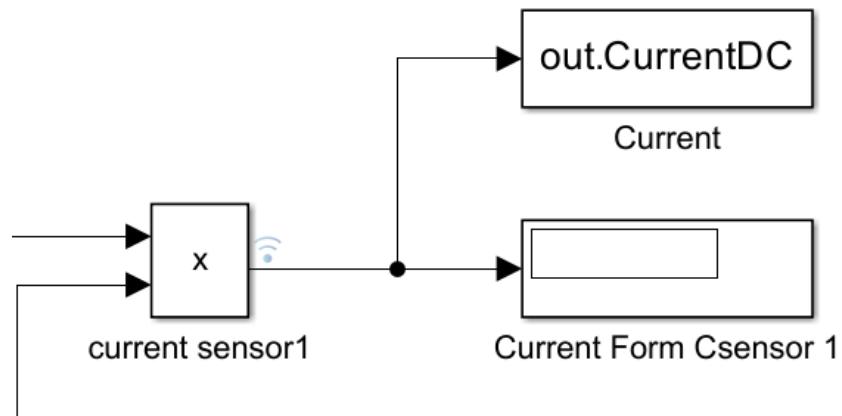
และเข้าสูตร Torque ที่มีการใช้รัศมี จากจุดศูนย์กลางเพลาถึงจุดที่แรงกีดการกด โดยรัศมี คือ 0.11127 เมตร และสูตรหา Torque คือ

$$\tau = Fr$$



รูปที่ 14 แสดงภาพ Drawing ของไม้ที่ติดกับเพลา

และการอ่านค่ากระแสที่ DC Motor กำลังใช้ได้จากการ Log Signal สัญญาณ Current Sensor 1



รูปที่ 15 แสดงการต่อ Block เพื่อแสดงสัญญาณ Current Sensor 1

## 11. วิเคราะห์และสร้างกราฟ Motor Characteristic

11.1 นำค่าที่ได้ทั้งหมดมาสร้างกราฟต่อไปนี้:

11.1.1 เทียบ Torque กับ Current

```
% Torque-Current Curve
xTorqueCurrent = [0, stallTorque]; % Torque range
yCurrent = [I_No_load, I_Stall]; % Current range
ratedCurrent = interp1(xTorqueCurrent, yCurrent, ratedTorque, 'linear'); % Current at rated torque
```

รูปที่ 16 แสดง Code ในการเปรียบเทียบ Torque กับ Current

11.1.2 เทียบ Torque กับ Speed

```
% Torque-Speed Curve
xTorqueSpeed = [0, stallTorque]; % Torque range
ySpeed = [maxSpeed, 0]; % Speed range
ratedSpeed = interp1(xTorqueSpeed, ySpeed, ratedTorque, 'linear'); % Speed at rated torque
```

รูปที่ 17 แสดง Code ในการเปรียบเทียบ Torque กับ Speed

11.1.3 เทียบ Torque กับ Power

```
% Power Curve
Torque2 = linspace(0, stallTorque, 1000); % Torque from 0 to Stall Torque
P = -(maxSpeed/stallTorque)*(Torque2).^2 + maxSpeed*Torque2;
```

รูปที่ 18 แสดง Code ในการเปรียบเทียบ Torque กับ Power

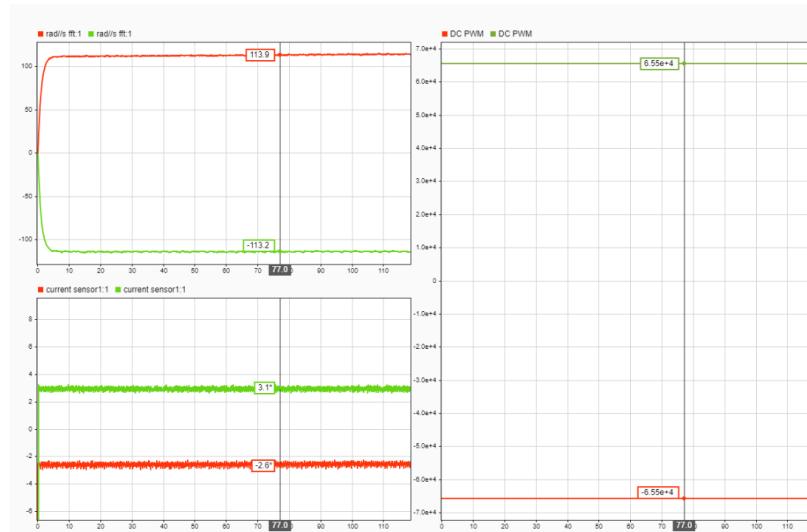
10.1.1 เทียบ Torque กับ Efficiency

```
% Efficiency Curve
Torque1 = linspace(0, stallTorque, 1000); % Torque from 0 to Stall Torque
n = ((-(maxSpeed/stallTorque)*(Torque1).^2) + maxSpeed*Torque1) ./ ...
(((I_Stall - I_No_load)/stallTorque)*Torque1*Vin + I_No_load*Vin)) * 100;
```

รูปที่ 19 แสดง Code ในการเปรียบเทียบ Torque กับ Efficiency

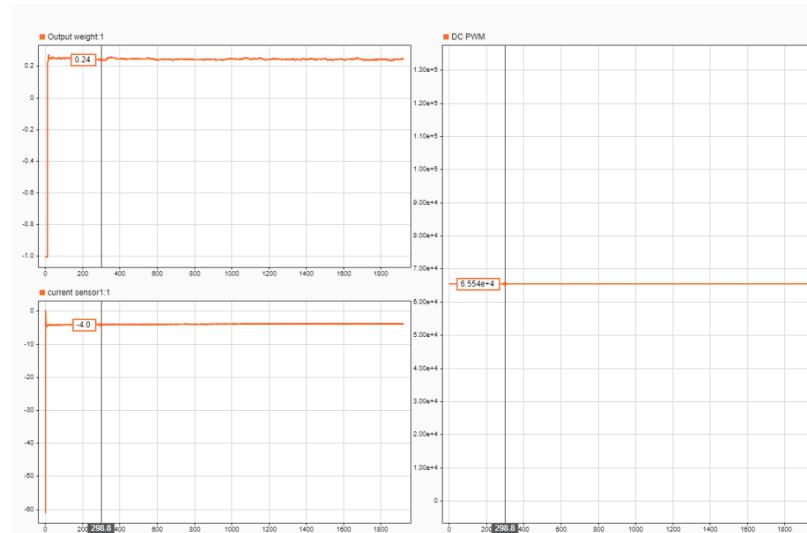
## ผลการทดลอง

ความเร็วที่วัดได้จาก Encoder และกระแสที่ใช้งานขณะ No load คือ 113.9 Rad/s และ 3.1 A ตามลำดับ

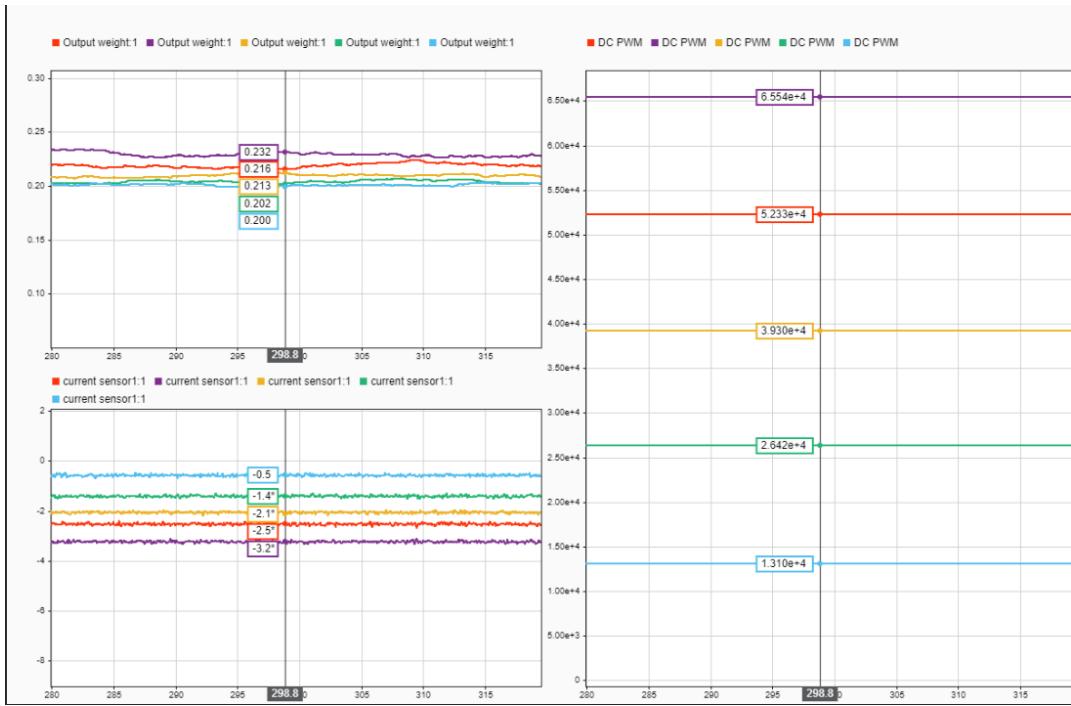


รูปที่ 20 แสดงกราฟความเร็วที่วัดได้จาก Encoder และกระแสที่ใช้งานขณะ No load

Stall Torque ที่วัดได้จากการ Signal Conditioning Loadcell คือ 0.24 Nm และกระแสที่ใช้งานขณะ Stall Torque คือ 4 A



รูปที่ 21 แสดงกราฟ Stall Torque ที่วัดได้จากการ Signal conditioning Loadcell และกระแสที่ใช้งานขณะ Stall Torque



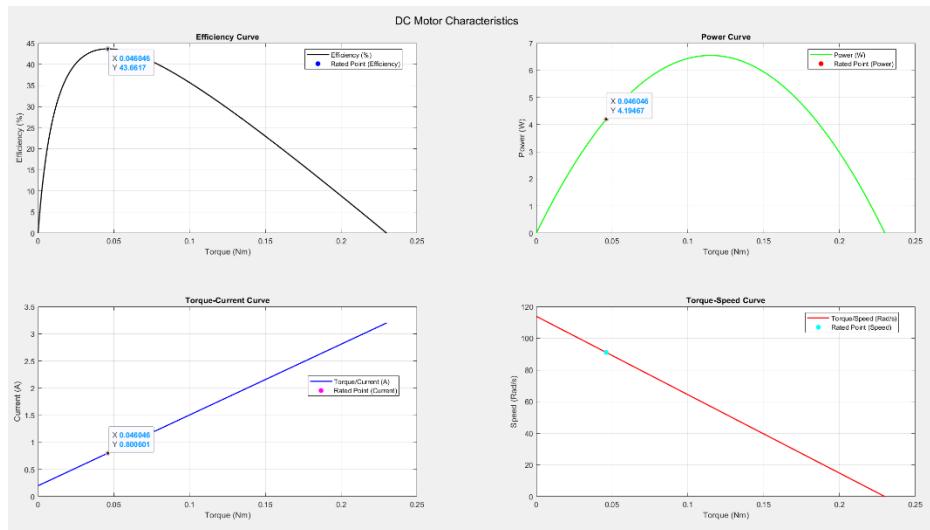
รูปที่ 22 แสดงกราฟกระแสที่วัดได้จาก Hall Current Sensor ที่ผ่านการ signal conditioning และ เทียบกับความถี่ที่สั่งงาน PWM

ในขณะ Stall Torque โดยเปลี่ยน Duty Cycle จะมี Torque ที่ทำได้ และกระแสที่เปลี่ยนไปดังนี้

Duty Cycle (%)	Torque (Nm)	Current (A)
0	0	0
20	0.200	0.5
40	0.202	1.4
60	0.213	2.1
80	0.216	2.5
100	0.232	3.2

ตารางที่ 1 แสดง Duty Cycle ต่อ Torque และกระแส

## เมื่อนำค่ามาพล็อตกราฟจะได้ออกมาดังนี้



รูปที่ 23 แสดงกราฟ Efficiency (ช้ายบ), Power (ขวบ), Current (ช้ายล่าง), Torque (ขวาล่าง) เทียบกับ Torque

### สรุปผล

ในขณะ No Load จะมีความเร็วสูงสุดอยู่ที่ 113.9 Rad/s และมีกระแสขณะ No Load ที่ 0.2 A

ในขณะ Stall Torque จะมี Torque ที่ทำได้คือ 0.24 Nm และมีกระแสขณะ Stall Torque ที่ 4 A

ในจุดที่มี Efficiency สูงที่สุด คือจุดที่มีแรงบิดที่ 0.046 Nm ซึ่งในจุดดังกล่าวมีพลังงานข้าอกอที่ 4.19 W มีกระแสที่ 0.8 A และมี Speed ที่ 90 Rad/s

### อภิปรายผล

ความเร็วที่วัดได้จาก Encoder และกระแสที่ใช้งานขณะ No load คือ 113.9 Rad/s และ 3.1 A ตามลำดับ มีความไม่สอดคล้องกับแนวโน้มที่ควรเป็นเนื่องจากค่ากระแสเมื่อยังไม่เกินจริง โดยไม่สามารถอธิบายได้ว่าเกิดจากอะไร แต่คาดว่าอาจมีการผลิตโดยผิดพลาดหรืออิเล็กทรอนิกส์ตัวต้านแรงไว้อยู่ทำให้เกิด Load มากกว่าที่ควรจะเป็น

Stall Torque ที่วัดได้จากการ Signal Conditioning Loadcell คือ 0.24 Nm และกระแสที่ใช้งานขณะ Stall Torque คือ 4 A สอดคล้องกับกราฟที่พล็อตได้

ในขณะ Stall Torque โดยเปลี่ยน Duty Cycle จะมี Torque ที่ทำได้ และกระแสที่เปลี่ยนไปแปรผันกับค่า Duty Cycle

## ข้อเสนอแนะ

1. ควรตรวจสอบความคลาดเคลื่อนของเซนเซอร์ทุกชนิดที่ใช้งานก่อนทดลอง

## เอกสารอ้างอิง(แนบ link)

อ้างอิงขั้นตอนและผลการทดลองร่วมกับกลุ่ม A12

- <https://www.regalrexnord.com/regal-rexnord-insights/electric-motor-terminology>
- <https://www.modularcircuits.com/blog/articles/h-bridge-secrets/sign-magnitude-drive/>
- <https://www.modularcircuits.com/blog/articles/h-bridge-secrets/sign-magnitude-drive/>
- <https://www.modularcircuits.com/blog/articles/h-bridge-secrets/lock-anti-phase-drive/>

## การทดลองที่ 2 Stepper Motor

### จุดประสงค์

- เพื่อศึกษา และเข้าใจถึงหลักการทำงาน รวมถึงความสามารถของ Stepper Motor รวมถึงความสัมพันธ์ของ สัญญาณที่เปลี่ยนแปลงไประหว่าง Speed และ Frequency
- เพื่อศึกษา และเข้าใจถึงหลักการทำงานของ Motor ในรูปแบบ Full-Step และ Half-Step เนื่องจากรูปแบบนั้น ส่งผลต่อความเร็ว และตำแหน่งของ Stepper Motor
- เพื่อเข้าใจ และเข้าใจถึงกระบวนการเริ่มถึงจบของ Signal Conditioning และ Signal Processing เพื่ออธิบายถึง ที่มาของค่าจาก Incremental Encoder และ Hall Current Sensor รวมถึงวิธีการคำนวณ ขั้นตอนการทำ ห้องหมัดตั้งแต่ก่อนและ หลังการ Calibrate Sensor
- เพื่อศึกษา และเข้าใจถึงการหาสมการความสัมพันธ์ระหว่างกระแสไฟฟ้ากับแรงดันไฟฟ้าที่ออกจากระ Hall Current Sensor และการอธิบายถึงการ Unwrap ค่า
- เพื่อศึกษา และเข้าใจถึงการเขียนโปรแกรมผ่านการใช้ MATLAB และ Simulink สู่ไปควบคุมสั่งการกับบอร์ด Nucleo STM32G474RE โดยรับค่า Input จากสัญญาณของ Incremental Encoder และ Hall Current Sensor เพื่อแสดงค่า Output เป็นกราฟการ Log สัญญาณ ใน Data Inspector ในโปรแกรม MATLAB Simulink เพื่อให้เห็นถึงค่า Output ที่แปรผันตามค่า Output แบบ Real Time โดยกำหนดให้ Output เป็น ความเร็วเชิงมุม และกระแสไฟฟ้าในหน่วย SI derived

### สมมติฐาน

ถ้าการการสั่งงาน Stepper Motor ด้วยความละเอียดที่ต่างกัน ส่งผลต่อค่า Frequency ที่วัดได้ก่อนเกิดการ Loss Step ดังนั้นยิ่งสั่งงาน Stepper Motor ด้วยความละเอียดที่สูงขึ้น จะทำให้ค่า Frequency ที่วัดได้นั้นสูงขึ้นตามด้วย

### ตัวแปร

ตัวแปรต้น : ความละเอียดที่ใช้สั่งงาน Stepper Motor (Full Step, Half Step, 1/4 Step, 1/8 Step, 1/16 Step และ 1/32 Step)

ตัวแปรตาม : ค่า Frequency ที่วัดได้จาก Encoder ก่อนเกิดการ Loss Step (Hz)

ตัวแปรควบคุม : แรงดันไฟฟ้าที่จ่ายเข้าบอร์ด, ชนิดของสายจัมเปอร์ที่ใช้เชื่อมต่อสายไฟ, ชนิดของบอร์ด Microcontroller และชนิดของ Stepper Motor

## นิยามคัพท์เฉพาะ

Stepper Motor หมายถึง มอเตอร์ชนิดหนึ่งที่ใช้การควบคุมการหมุนทีละขั้น (Step) ด้วยสนามแม่เหล็กที่สร้างจากการจ่ายกระแสไฟฟ้าไปยังชุดลวดของสเตเตอร์อย่างเป็นลำดับ การหมุนของโรเตอร์จะถูกควบคุมโดยจำนวน Pulse และลำดับของการจ่ายสัญญาณไฟ ทำให้สามารถกำหนดตำแหน่ง หมุนหมุน และความเร็วได้อย่างแม่นยำโดยไม่ต้องใช้เซนเซอร์ตรวจจับตำแหน่ง

Rotor หมายถึง ส่วนที่หมุนของมอเตอร์ไฟฟ้า โดยมักประกอบด้วยชุดลวดหรือแม่เหล็กถาวรซึ่งทำงานที่สร้างแรงบิดเมื่อเกิดปฏิกิริยาพันธ์กับสนามแม่เหล็กของ Stator

Stator หมายถึง ส่วนที่อยู่กับที่ของมอเตอร์ไฟฟ้า ทำงานที่สร้างสนามแม่เหล็กโดยการจ่ายกระแสไฟฟ้าผ่านชุดลวดในโครงสร้างที่วางเรียงอยู่รอบ Rotor

Loss Step หมายถึง สถานการณ์ที่มอเตอร์แบบ Stepper Motor ไม่สามารถรักษาตำแหน่งการหมุนที่ต้องการได้อันเนื่องมาจากการบิดไม่เพียงพอหรือการทำงานที่ความถี่สูงเกินไปจนเกิดการสูญเสียตำแหน่ง

Full-Step Drive หมายถึง รูปแบบการควบคุม Stepper Motor โดยใช้ชุดลวดทั้งสองชุดทำงานพร้อมกันในแต่ละขั้นตอน ทำให้เกิดแรงบิดสูงสุด แต่ตำแหน่งของ Rotor จะเคลื่อนที่ครั้งละ 1 Step เท่านั้น

Half-Step Drive หมายถึง รูปแบบการควบคุม Stepper Motor ที่สลับการเปิดใช้งานชุดลวดเพียงชุดเดียว กับการเปิดพร้อมกันสองชุดในแต่ละขั้นตอน ทำให้ได้จำนวนขั้นตอนเพิ่มขึ้นเป็นสองเท่า ลดการสั่น และเพิ่มความละเอียดในการเคลื่อนที่

Micro-Step Drive หมายถึง รูปแบบการควบคุม Stepper Motor ที่แบ่งการเคลื่อนที่ในแต่ละ Step ออกเป็นส่วนย่อยยิ่งขึ้นโดยการจ่ายกระแสไฟฟ้าบลําให้กับชุดลวด ทำให้การเคลื่อนที่ราบรื่นและแม่นยำมากขึ้น

## นิยามเชิงปฏิบัติการ

Stepper Motor หมายถึง Stepper Motor รุ่น HANPOSE Stepper Motor (Nema11)

Stepper Motor Driver หมายถึง Stepper Motor Driver DRV8825

Loss Step หมายถึง การที่ Stepper Motor ไม่สามารถทำงานต่อไปได้แล้ว และ Encoder อ่านค่าความเร็วขณะนั้นเป็น 0 Rad/s

ความละเอียดของ Stepper Motor หมายถึง การเลือก Mode การทำงานของ Stepper Motor ซึ่งในมอเตอร์รุ่นนี้มีการทำงานทั้งหมด 6 Mode ได้แก่ Full Step, Half Step, 1/4 Step, 1/8 Step, 1/16 Step และ 1/32 Step

Filter หมายถึง การกรองเอาสัญญาณที่ไม่ต้องการออกให้เหลือแต่สัญญาณที่ต้องการ

## เอกสารและงานวิจัยที่เกี่ยวข้อง

### 1. HANPOSE Stepper Motor (Nema11) แรงบิด 10 N.cm 1.0A (11HS3410)

เป็น Stepper Motor รุ่น 11HS3410 ซึ่งผลิตโดย Hanpose ขนาด NEMA 11 ซึ่งมีหน้าตัดขนาด 1.1 นิว (28 มม.) รุ่นนี้ให้แรงบิดสูงสุดประมาณ 10 N·cm และใช้กระแสไฟสูงสุด 1.0 A ต่อเฟส มีมุมก้าวที่ 1.8° ต่อ Step หรือ 200 Step ต่อ 1 รอบ



รูปที่ 24 แสดง Stepper Motor รุ่น 11HS3410

### 2. หลักการทำงาน Stepper Motor

Stepper Motor หมายถึง มอเตอร์ที่ออกแบบให้สามารถหมุนได้ทีละขั้น (Step) อย่างแม่นยำ โดยมีโครงสร้างหลัก 2 ส่วน คือ โรเตอร์ (Rotor) ซึ่งเป็นแกนหมุน และสเตเตอร์ (Stator) ซึ่งเป็นส่วนที่อยู่รอบนอกที่มีขดลวดพันอยู่ หลักการทำงานแบบละเอียดสามารถอธิบายได้ดังนี้

#### 2.1 การสร้างสนามแม่เหล็กใน Stator

ขดลวดบนสเตเตอร์จะถูกจัดเรียงเป็นเฟส เช่น เฟส A, B, C และ D การจ่ายไฟให้ขดลวดจะสร้างสนามแม่เหล็กซึ่งดึงดูดหรือผลักโรเตอร์ที่มีแม่เหล็กถาวรหรือวัสดุแม่เหล็กอ่อน (Soft Magnetic Material) ให้หมุนไปในตำแหน่งที่ต้องการ การจ่ายไฟนี้ใช้สัญญาณไฟฟ้าแบบดิจิทัลเป็นชุดพัลส์ที่เรียกว่า Step Signal โดยลำดับและทิศทางของการจ่ายไฟจะกำหนดทิศทางการหมุนของมอเตอร์

## 2.2 การหมุนทีละขั้น (Step Movement)

โรเตอร์จะหมุนทีละมุ่งที่แน่นอน เรียกว่า Step Angle ซึ่งขึ้นอยู่กับจำนวนขั้วแม่เหล็กบนโรเตอร์และจำนวนขดลวดบนสเตเตอเริร์

## 2.3 รูปแบบการขับเคลื่อน (Drive Modes)

Stepper Motor สามารถทำงานในหลายโหมดขึ้นอยู่กับการจ่ายไฟ สามารถแบ่งการจ่ายไฟออกเป็น 3 แบบ ได้แก่

1. Full-Step Drive: จ่ายกระแสไฟเต็มที่ให้กับขดลวดทีละเฟสหรือสองเฟสพร้อมกัน โรเตอร์จะหมุนในมุมที่กำหนด เช่น 1.8 องศา/Step
2. Half-Step Drive: จ่ายกระแสไฟสลับระหว่าง 1 เฟสและ 2 เฟส โรเตอร์จะเคลื่อนที่ครึ่งหนึ่งของมุม Step ปกติ (0.9 องศา/Step) ทำให้การหมุนละเอียดขึ้น
3. Micro-Step Drive: แบ่งการจ่ายกระแสไฟออกเป็นส่วนย่อยๆ ทำให้โรเตอร์เคลื่อนที่ได้ละเอียดและนิ่มนวลมากยิ่งขึ้น

## 2.4 การควบคุมตำแหน่ง

Stepper Motor ไม่ต้องใช้เซนเซอร์ตรวจจับตำแหน่ง เพราะตำแหน่งของโรเตอร์สามารถคำนวณได้จากจำนวน Pulse ที่ส่งไปยังมอเตอร์ หากจำนวน Pulse ที่ส่งไปตรงกับจำนวน Step ที่โรเตอร์หมุน ตำแหน่งจะถูกควบคุมได้อย่างแม่นยำ

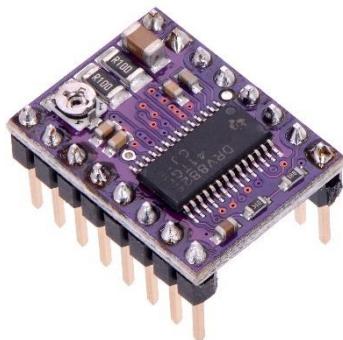
## 2.5 ความสัมพันธ์ระหว่างความถี่กับความเร็ว

ความเร็วของ Stepper Motor ขึ้นอยู่กับความถี่ของ Pulse หาก Pulse ถูกส่งเร็วขึ้น โรเตอร์จะหมุนเร็วขึ้น แต่หากเพิ่มความถี่เกินขีดจำกัดของมอเตอร์ อาจทำให้เกิดการ "Loss Step" หรือการพลาดตำแหน่ง

## 3. DRV8825

DRV8825 เป็น Driver สำหรับ Stepping Motor โดยบริษัทที่ออกแบบคือ Texas Instruments เพื่อใช้ควบคุม Stepping Motor แบบสองขั้ว (Bipolar Stepper Motor) โดยมีความสามารถในการขับเคลื่อนแบบ Microstepping ตั้งแต่ Full-Step จนถึง 1/32-Step ซึ่งช่วยให้การเคลื่อนไหวของมอเตอร์มีความละเอียดและนิ่มนวลมากขึ้น นอกจากนี้ DRV8825 ยังสามารถทำงานในช่วงแรงดันไฟฟ้าระหว่าง 8.2V ถึง 45V และขับกระแสได้สูงสุดถึง 2.5A ต่อเฟส หากมีการระบายความร้อนที่เหมาะสม การใช้งาน DRV8825 ยังรวมถึงฟังก์ชันการป้องกันต่าง ๆ เช่น การป้องกันกระแสเกิน, การปิดตัวเองเมื่อความร้อนสูงเกินไป, การล็อกแรงดันไฟฟ้าต่ำ และการแสดงสถานะข้อผิดพลาดผ่านพิน nFAULT นอกจากนี้ยังมีโหมดพลังงานต่ำที่ช่วยลดการใช้พลังงานเมื่อไม่ได้ใช้งาน Driver นี้เหมาะสม

สำหรับการใช้งานในอุปกรณ์อัตโนมัติต่าง ๆ เช่น เครื่องพิมพ์, สแกนเนอร์, กล้องรักษาความปลอดภัย, ระบบอัตโนมัติ ในสำนักงาน และหุ่นยนต์ เป็นต้น โดยมีการใช้อินเทอร์เฟซ STEP/DIR ที่เรียบง่ายสำหรับการควบคุม ซึ่งทำให้การใช้งานสะดวกและง่ายต่อการเชื่อมต่อกับวงจรควบคุมต่าง ๆ



รูปที่ 25 แสดงหน้าตาของ DRV8825

<b>MODE2</b>	<b>MODE1</b>	<b>MODE0</b>	<b>STEP MODE</b>
0	0	0	Full step (2-phase excitation) with 71% current
0	0	1	1/2 step (1-2 phase excitation)
0	1	0	1/4 step (W1-2 phase excitation)
0	1	1	8 microsteps/step
1	0	0	16 microsteps/step
1	0	1	32 microsteps/step
1	1	0	32 microsteps/step
1	1	1	32 microsteps/step

รูปที่ 26 แสดงถึงการสั่งงาน Stepper Motor ให้มีการทำงานในรูปแบบต่าง ๆ

### วิธีดำเนินการทดลอง

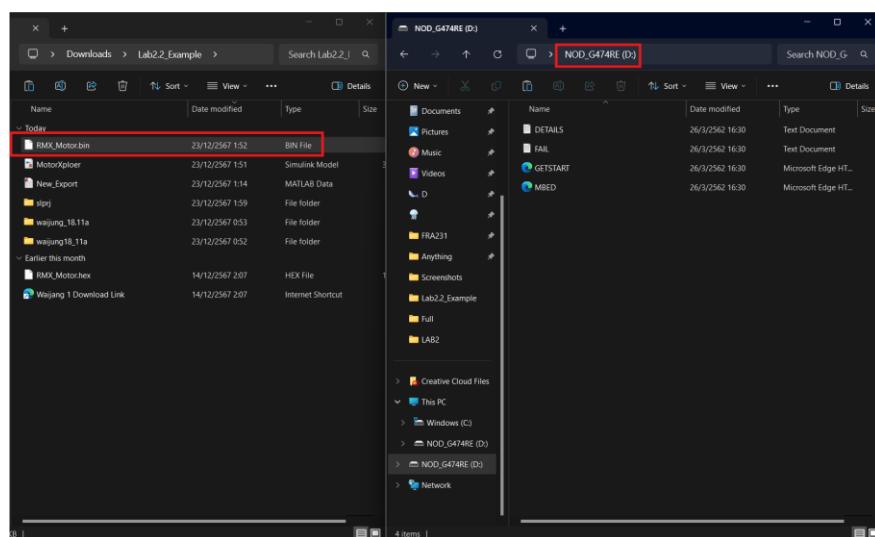
การทดลองเรื่อง Stepper Motor สามารถแบ่งการทดลองออกเป็น \*\* ส่วน ส่วนแรก คือ การติดตั้งโปรแกรมรวมถึงการตั้งค่าอุปกรณ์ต่าง ๆ เช่น มอเตอร์, บอร์ด, การควบคุม และโปรแกรม MATLAB และ Simulink ส่วนที่สอง คือ การออกแบบการทดลอง และทำการทดลอง ซึ่งคือการปรับความละเอียดของ Stepper Motor เพื่อเก็บค่า Frequency และความเร็วจาก Encoder ที่ผ่านการ Butterworth Low Pass Filter ส่วนที่สาม คือ การวิเคราะห์ถึงข้อมูลที่ได้มาจากการทดลอง โดยต้องวิเคราะห์ถึงความสมพนธ์ของ ความละเอียดของ Stepper Motor กับค่า Frequency, การอธิบาย Signal Conditioning ฯลฯ

## วัสดุอุปกรณ์

1. RS PRO Hybrid, Permanent Magnet Stepper Motor, 0.22Nm Torque, 2.8 V, 1.8°, 42.3 x 42.3mm Frame, 5mm Shaft จำนวน 1 อัน
2. Incremental Encoder AMT103-V จำนวน 1 อัน
3. Nucleo STM32G474RE พร้อมสายอัปโหลด จำนวน 1 ชุด
4. DRV8825 Stepper Motor Controller IC จำนวน 1 อัน
5. MotorXplorer จำนวน 1 ชุด - ฐานสามารถบรรจุบอร์ดควบคุม, Breadboard, 3D-Print ใช้สำหรับการประกอบกับ Stepper Motor

## ขั้นตอนการดำเนินงาน

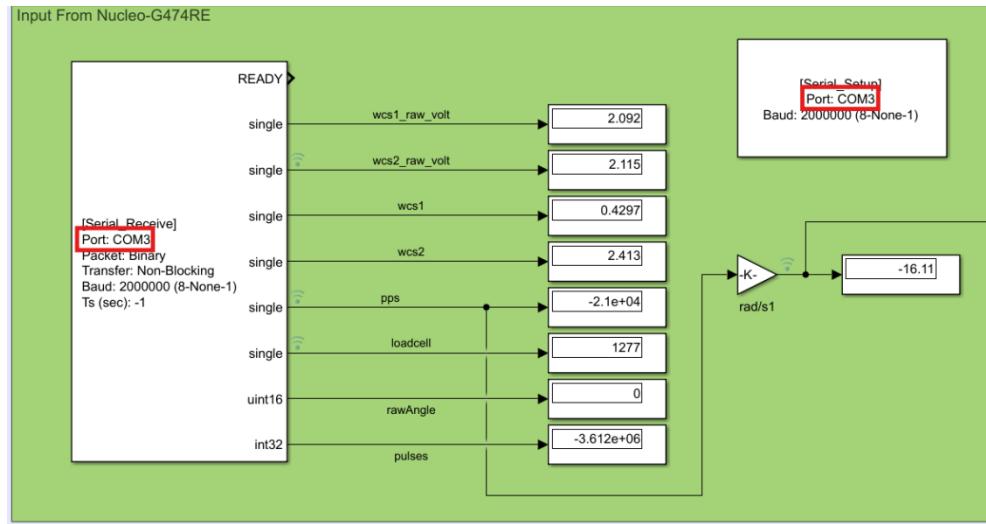
1. เชื่อมต่อสายอัปโหลดระหว่างคอมพิวเตอร์ และ Nucleo STM32G474RE
2. ย้ายไฟล์ RMX\_Motor.bin เข้าสู่พื้นที่จัดเก็บข้อมูลของบอร์ด Nucleo STM32G474RE



รูปที่ 27 แสดงการย้ายไฟล์ RMX\_Motor.bin เข้าสู่พื้นที่จัดเก็บของบอร์ด Nucleo STM32G474RE

4. ทำการเปิดไฟล์ Simulink เพื่อทำการตั้งค่าในการทำการทดลองเก็บค่าความเร็วของ Stepper Motor ที่ถูกวัดด้วย Encoder

5. เลือก Port ที่เชื่อมต่อเข้ากับ Nucleo STM32G474RE ให้ถูกต้อง



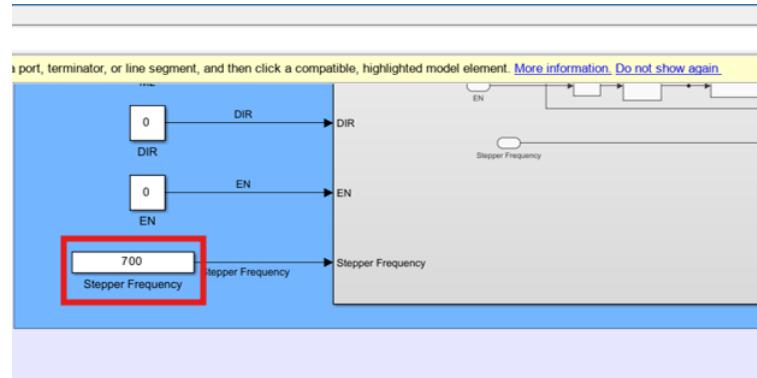
รูปที่ 28 แสดงการเลือก Port คอมพิวเตอร์ที่เชื่อมเข้ากับบอร์ด Nucleo STM32G474RE

6. สร้าง Real-Time Block เนื่องจาก Library Waijung ไม่สามารถกำหนดเวลาได้ จึงต้องใช้ Block นี้ เพื่อที่เวลาในการ Run คำนั้นตรง



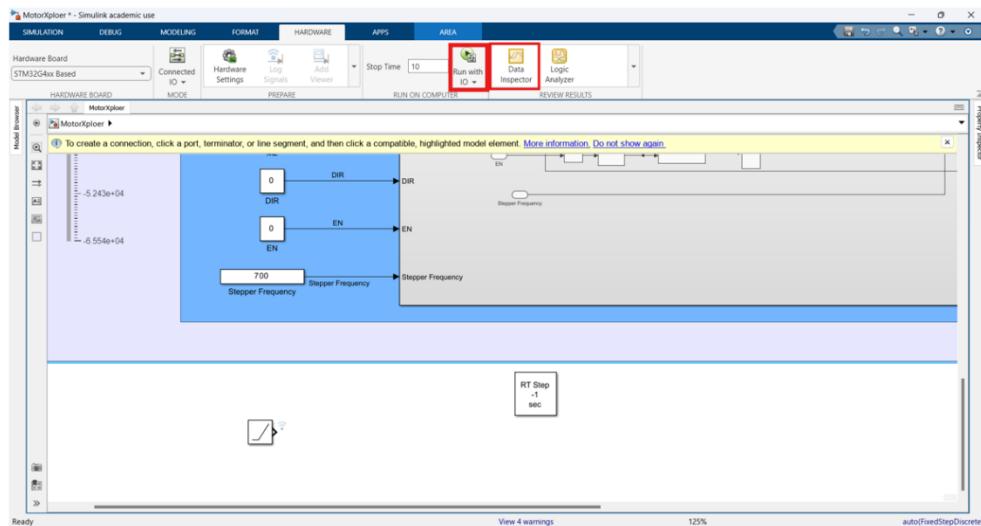
รูปที่ 29 แสดง Real-Time Block

7. ทำการจ่ายค่าสัญญาณ 700 Hz เพื่อให้มีความถี่คงที่



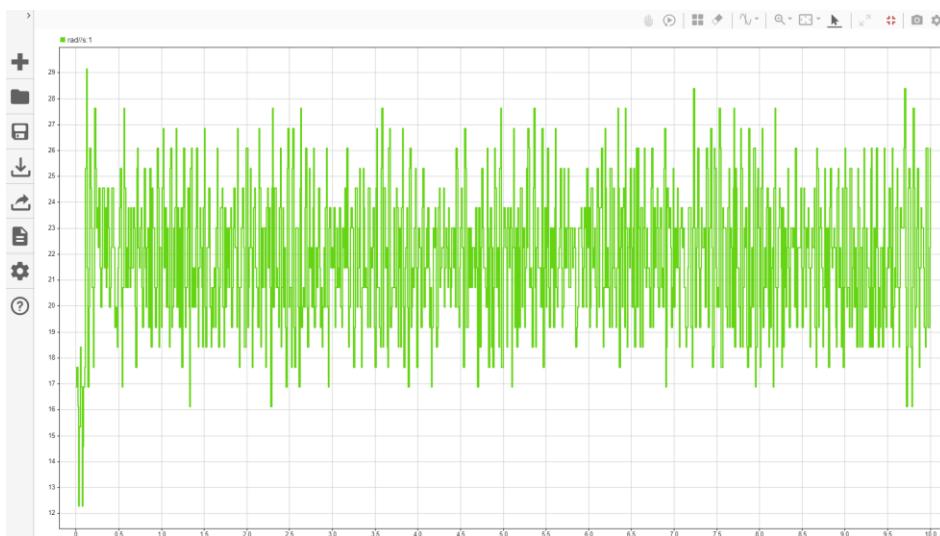
รูปที่ 30 แสดงการจ่ายค่าความถี่ 700 Hz และการเปิด Data Inspector

## 8. จากนั้นกด Run with IO และเปิด Data Inspector



รูปที่ 31 แสดงวิธีการกดปุ่ม Run with IO และการเปิด Data Inspector

## 9. เมื่อเปิดดู Data Inspector จะเห็นว่าสัญญาณมี Noise rob กวนจนยากต่อการเข้าใจถึงสัญญาณที่แท้จริง



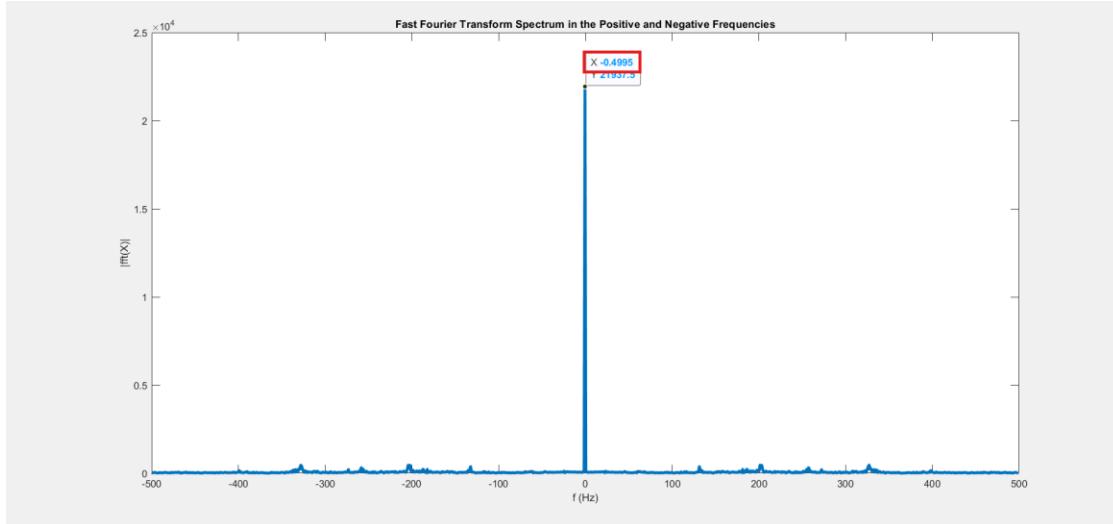
รูปที่ 32 แสดงค่าความเร็วที่รับได้จาก Encoder โดยมีสัญญาณ rob กวน

10. งานนั้นจึงนำค่าที่ได้ Export เข้าสู่ MATLAB เพื่อทำการ Filter เอาสัญญาณรบกวนออกไป

```
>> Frequency = 1000; % Sampling frequency  
T = 1/Frequency; % Sampling period  
L = 1001; % Length of signal  
  
Value = data.Data;  
Time = (linspace(0,10, 1001))';  
y = fft(Value);  
plot(Frequency/L*(-L/2:L/2-1),abs(fftshift(y)), "LineWidth",3)  
title("Fast Fourier Transform Spectrum in the Positive and Negative Frequencies")  
xlabel("f (Hz)")  
ylabel("|fft(X)|")  
>>
```

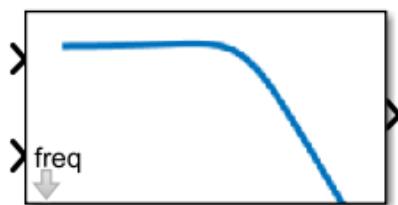
รูปที่ 33 แสดงโค้ดการใช้ Fast Fourier Transform เพื่อกรองสัญญาณรบกวน

จะเห็นได้ว่าช่วงความถี่ที่เป็นค่าสัญญาณที่แท้จริงนั้นอยู่ประมาณ 0.4995 Hz นอกนั้นจะเป็นสัญญาณรบกวน



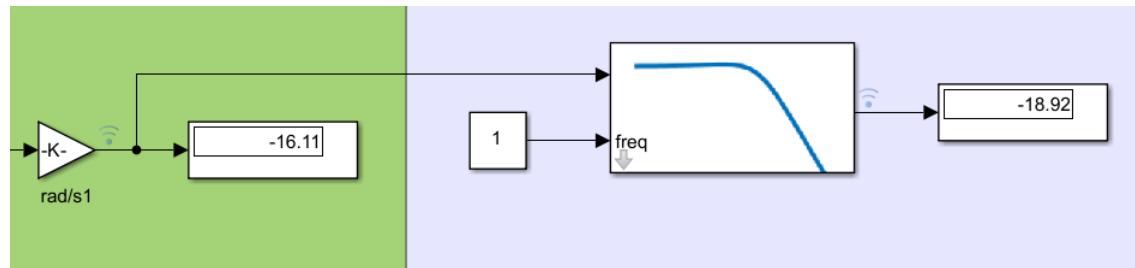
รูปที่ 34 แสดงการกรองสัญญาณด้วย Fast Fourier Transform เพื่อแสดงให้เห็นถึงช่วงความถี่ของสัญญาณที่แท้จริง

11. ทำให้คนผู้จัดทำต้องทำ Low Pass Filter เพื่อเลือกช่วงสัญญาณที่ต้องการผ่านการใช้ Varying Low Pass Filter Block ซึ่งภายในมีการใช้ Butterworth Low Pass Filter



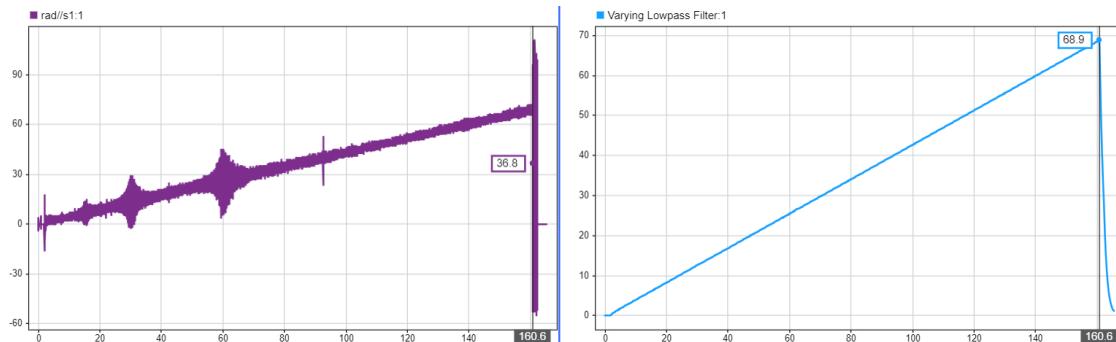
รูปที่ 35 แสดง Varying Low Pass Filter Block

โดยทำการต่อสัญญาณที่วัดค่าความเร็วจาก Stepper Motor ด้วย Encoder และทำการสร้าง Constant Block ที่มีค่า 1 เพื่อกำหนดเลือก Frequency ที่ต้องการได้



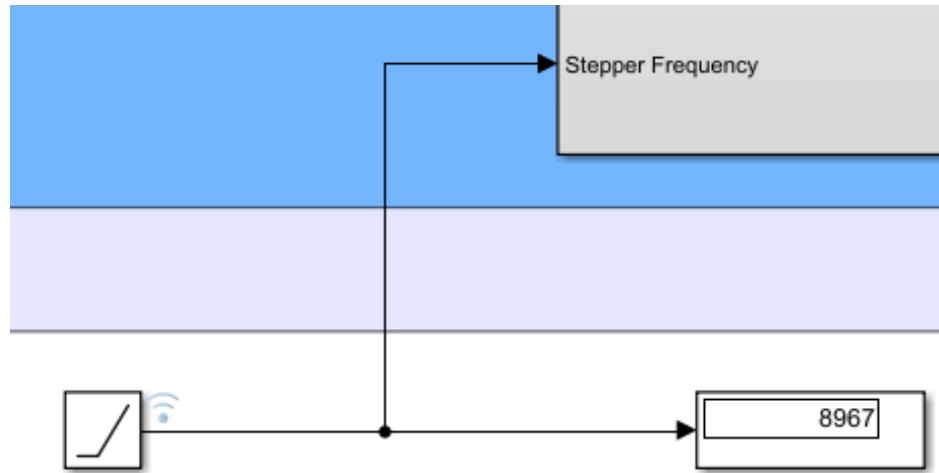
รูปที่ 36 แสดงให้เห็นถึงการใช้งาน Varying Low Pass Filter Block

- ทำการ Run ค่า และเปิด Data Inspector เพื่อดูกราฟสัญญาณที่ได้ จะเห็นได้ว่าก่อนทำการ Butterworth Low Pass Filter สัญญาณที่ได้มามีไม่ได้มีความเรียบเนียน (สีขาว) แต่เมื่อผ่านการทำกระบวนการนั้นแล้วสัญญาณมีความเรียบเนียนขึ้นอย่างเห็นได้ชัด (สีฟ้า)

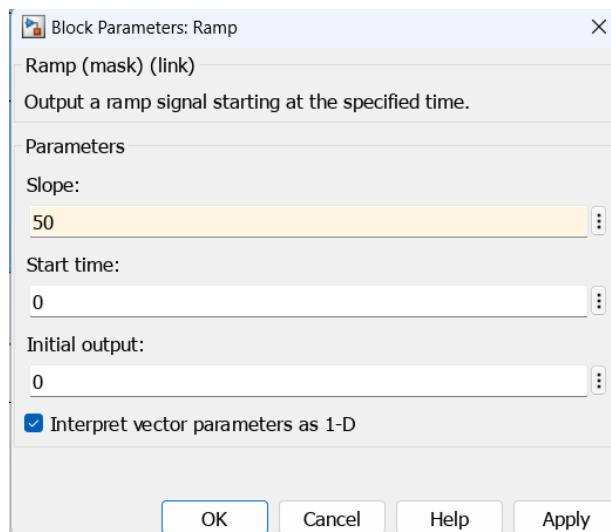


รูปที่ 37 แสดงการเปรียบเทียบก่อน และหลังการทำ Butterworth Low Pass Filter

11. จากนั้นจึงทำการต่อ Ramp Block เข้า Stepper Frequency และทำการตั้งค่า Slope เป็น 50 ซึ่งหมายความว่า Stepper Motor จะมีความถี่เพิ่มขึ้น 50 Hz ทุก ๆ 1 วินาที

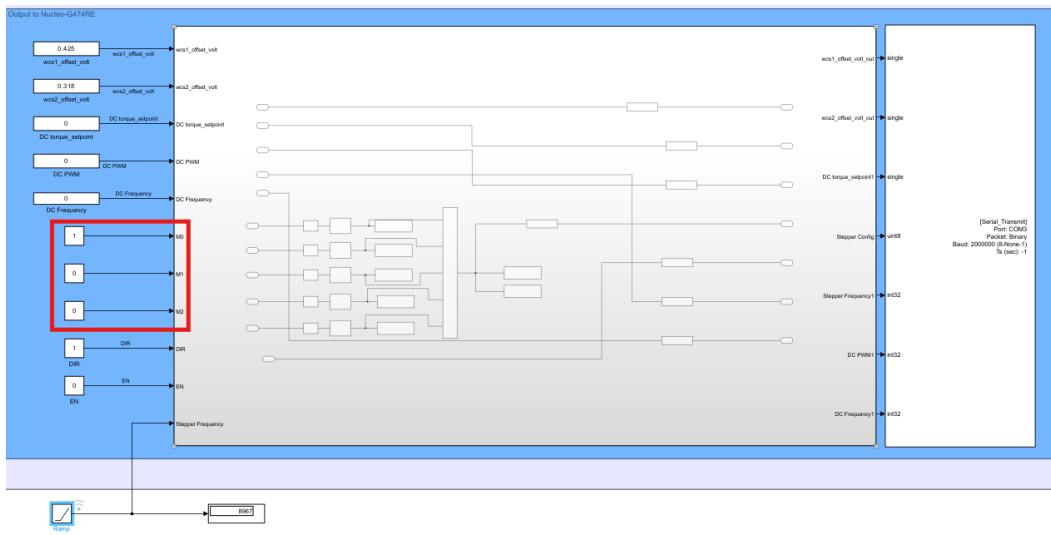


รูปที่ 38 แสดงการต่อ Ramp Block เข้า Stepper Frequency



รูปที่ 39 แสดงการตั้งค่า Slope ของ Ramp Block

12. กำหนด Mode การทำงานของ Stepper Motor ผ่านการตั้งค่า M0, M1 และ M2 เพื่อกำหนด Micro Step Resolution เนื่องจากการทำงาน 1/32 Step มีความเป็นไปได้ในการใช้งานที่เหมือนกัน ทางคณะผู้จัดทำจึงทำการเลือกมาสรุปแบบเดียวที่จะตั้งค่า และเก็บข้อมูล



รูปที่ 40 แสดง Block ที่ต่อเข้า M0, M1 และ M2 ซึ่งต้องทำการแก้ไข

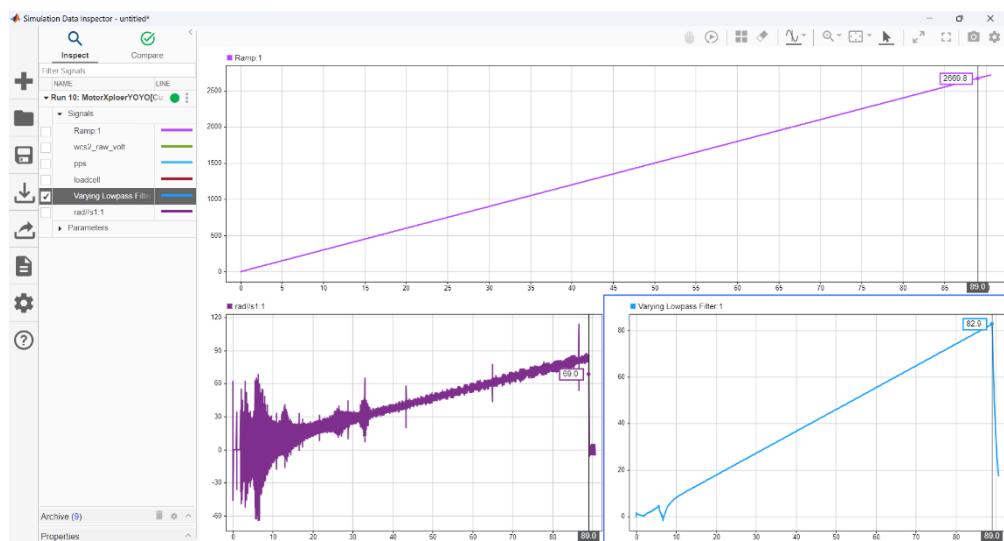
Micro Step Resolution	M0	M1	M2
Full Step	0	0	0
Half Step	1	0	0
1/4 Step	0	1	0
1/8 Step	1	1	0
1/16 Step	0	0	1
1/32 Step	1	0	1
1/32 Step	0	1	1
1/32 Step	1	1	1

ตารางที่ 2 แสดงการตั้งค่า M0, M1 และ M2 เพื่อกำหนด Micro Step Resolution

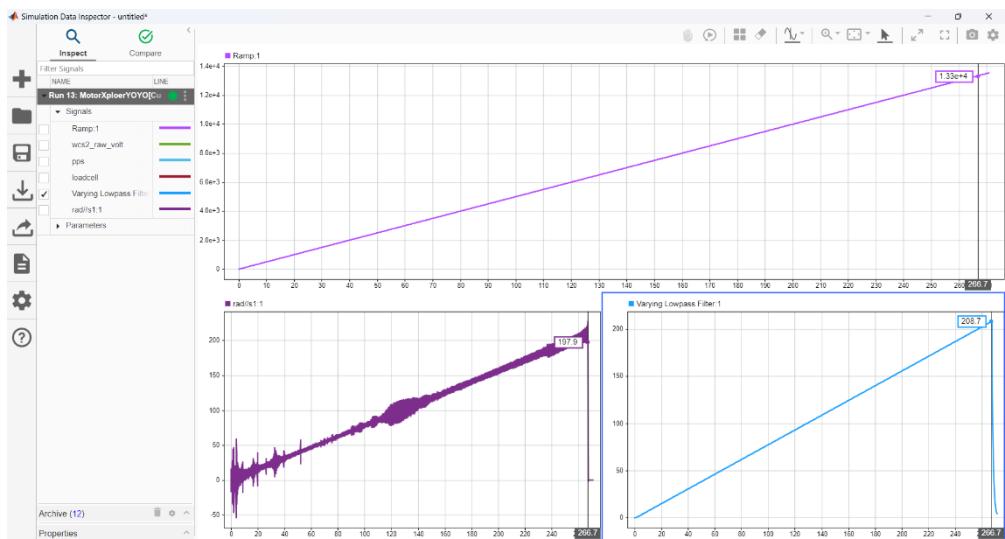
13. เมื่อทำการตั้งค่า Micro Step Resolution ในแต่ละความละเอียด จึงกด RUN with IO เพื่อเก็บข้อมูล จะ Stepper Motor มีการ Loss Step คือ Stepper Motor ไม่สามารถทำงานต่อไปได้แล้ว และ Encoder จะอ่านค่าได้ที่ 0 Rad/s ตลอด จึงกดหยุดการทำงานเชื่อมต่อ

14. เปิด Data Inspector เพื่อดูกราฟ Frequency, ความเร็วจาก Encoder ที่ยังไม่ได้ทำการ Filter และความเร็วจาก Encoder ที่ผ่านการ Butterworth Low Pass Filter
15. ใช้ One Cursor เพื่อทำการวัดช่วงสุดท้ายก่อนที่ Stepper Motor ไม่สามารถทำงานต่อไปได้แล้ว และบันทึกค่า Frequency (Hz) และความเร็วจาก Encoder ที่ผ่านการ Butterworth Low Pass Filter (Rad/s)
16. ทำขั้นตอน 13-15 แต่ต้องเปลี่ยน Micro Step Resolution ตามข้อที่ 12 และกำหนด Pin DIR
17. ใช้ Pin DIR ของ DRV8255 ในการเลือกทิศทางการหมุนของ Stepper Motor โดย DIR= 0 คือ การหมุนแบบ Clockwise และ DIR=1 คือ การหมุนแบบ Counterclockwise

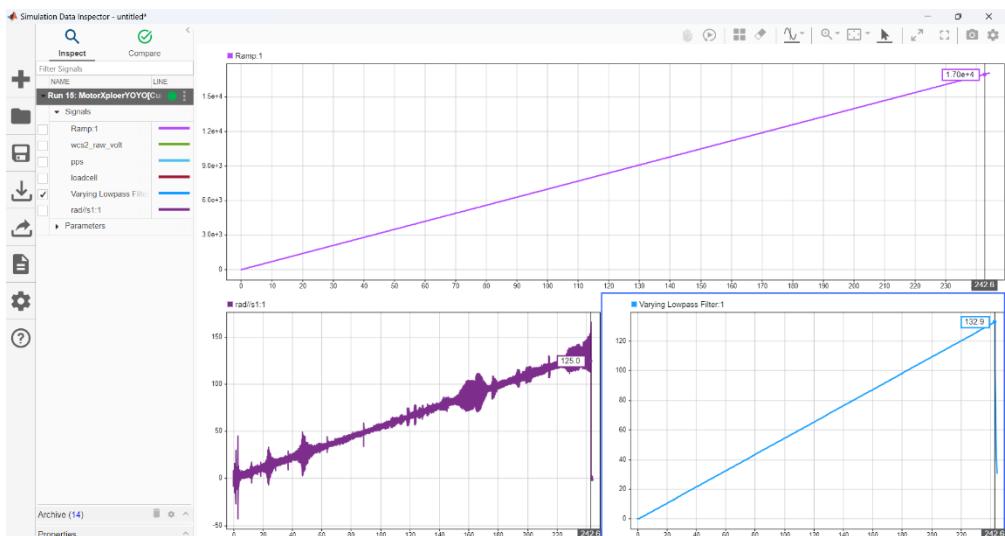
#### ผลการทดลอง



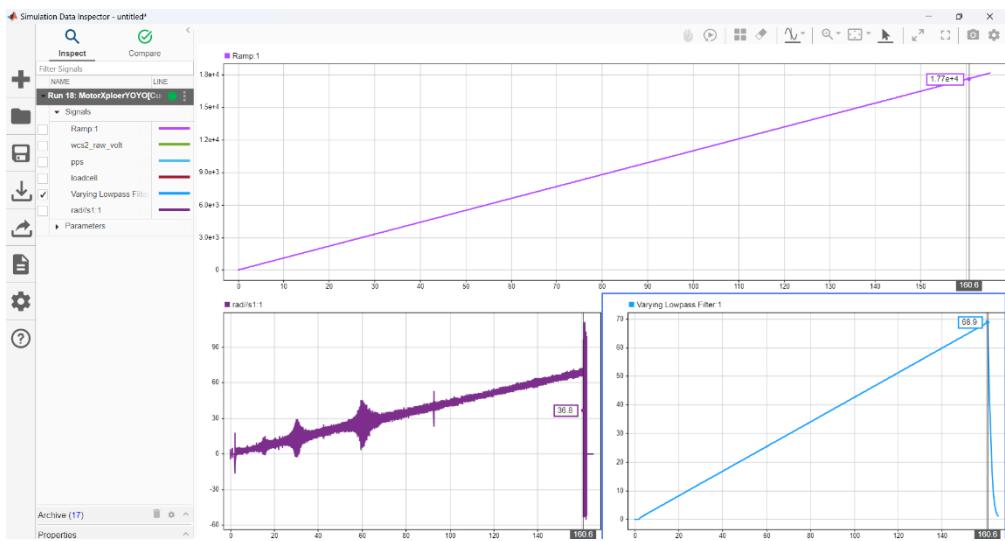
รูปที่ 41 แสดงการเปรียบเทียบ Frequency, ความเร็วจาก Encoder ที่ยังไม่ได้ทำการ Filter และความเร็วจาก Encoder ที่ผ่านการ Butterworth Low Pass Filter ที่การทำงาน Full Step



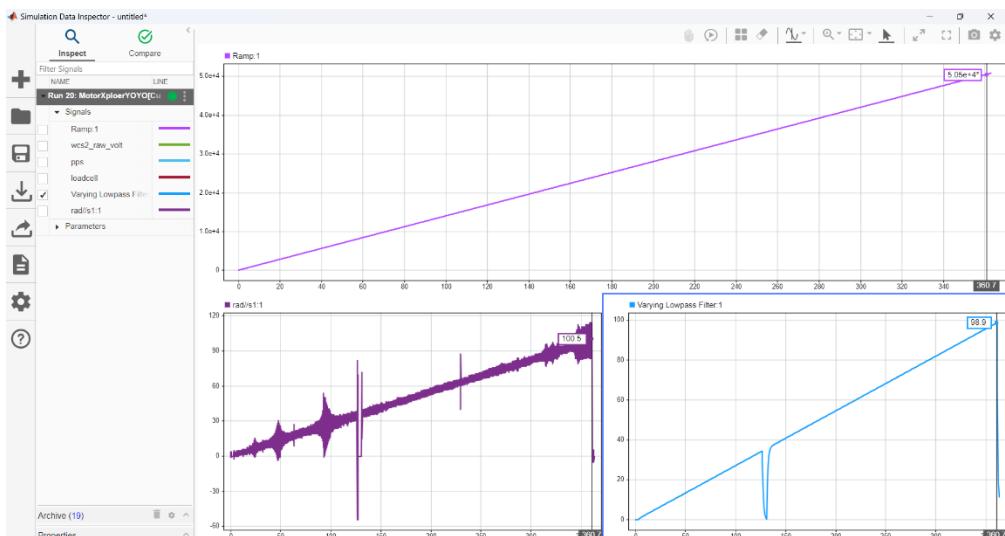
รูปที่ 42 แสดงการเปรียบเทียบ Frequency, ความเร็วจาก Encoder ที่ยังไม่ได้ทำการ Filter และความเร็วจาก Encoder ที่ผ่านการ Butterworth Low Pass Filter ทำการทำงาน Half Step



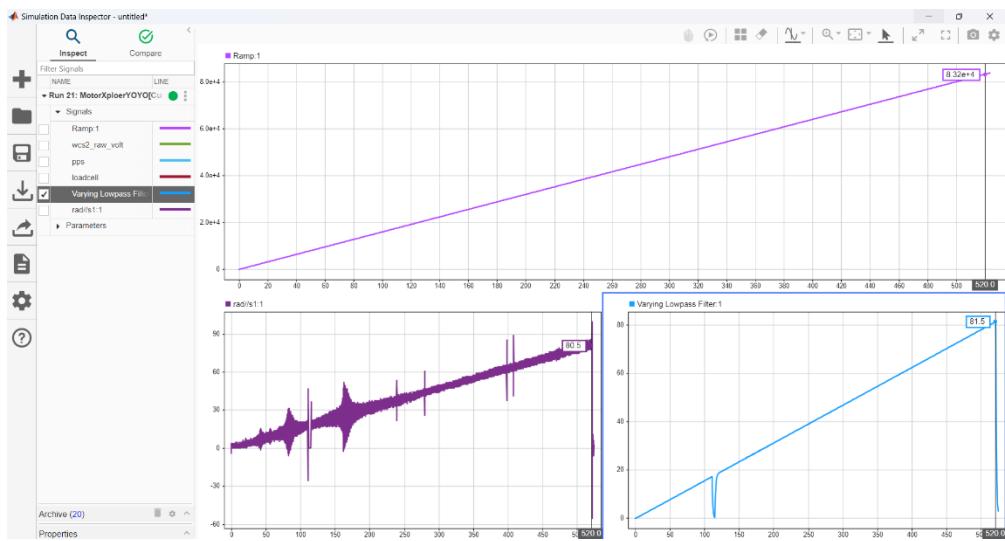
รูปที่ 43 แสดงการเปรียบเทียบ Frequency, ความเร็วจาก Encoder ที่ยังไม่ได้ทำการ Filter และความเร็วจาก Encoder ที่ผ่านการ Butterworth Low Pass Filter ทำการทำงาน 1/4 Step



รูปที่ 44 แสดงการเปรียบเทียบ Frequency, ความเร็วจาก Encoder ที่ยังไม่ได้ทำการ Filter และความเร็วจาก Encoder ที่ผ่านการ Butterworth Low Pass Filter ที่การทำงาน 1/8 Step



รูปที่ 45 แสดงการเปรียบเทียบ Frequency, ความเร็วจาก Encoder ที่ยังไม่ได้ทำการ Filter และความเร็วจาก Encoder ที่ผ่านการ Butterworth Low Pass Filter ที่การทำงาน 1/16 Step



รูปที่ 46 แสดงการเปรียบเทียบ Frequency, ความเร็วจาก Encoder ที่ยังไม่ได้ทำการ Filter และความเร็วจาก Encoder ที่ผ่านการ Butterworth Low Pass Filter ที่การทำงาน 1/32 Step

Micro Step Resolution	Rad/s	Frequency
Full Step	82.9	2669.8
Half Step	208.7	13300
1/4 Step	132.9	17000
1/8 Step	68.9	17700
1/16 Step	98.9	50500
1/32 Step	81.5	83200

ตารางที่ 3 แสดงการเก็บค่า Frequency และความเร็วจาก Encoder ที่ผ่านการ Butterworth Low Pass Filter ของแต่ละ Micro Step Resolution ในการหมุนแบบ Clockwise

Micro Step Resolution	Rad/s	Frequency
Full Step	-95.6	3097
Half Step	-197.4	12700
1/4 Step	-137.9	17700
1/8 Step	-68.9	17700
1/16 Step	-97.7	49900
1/32 Step	-81.5	83200

ตารางที่ 4 แสดงการเก็บค่า Frequency และความเร็วจาก Encoder ที่ผ่านการ Butterworth Low Pass Filter ของแต่ละ Micro Step Resolution ในการหมุนแบบ Counterclockwise

## สรุปผล

จากตารางที่ 3 แสดงให้เห็นว่าการทำงานของ Stepper Motor ในทิศทาง Clockwise โดย Micro Step Resolution ที่ Full Step นั้นมี Frequency น้อยที่สุด คือ 2669.8 Hz แต่ว่าที่ความละเอียด 1/32 มี Frequency มากที่สุด คือ 83200 Hz สามารถสื่อได้ว่าเมื่อมีความละเอียดเพิ่มมากขึ้น ค่า Frequency จะยิ่งเพิ่มขึ้นตาม สรุปได้ว่าความละเอียดของ Stepper Motor แปรผันตรงกับ Frequency ในการเกิด Loss Step

จากตารางที่ 4 แสดงให้เห็นว่าการทำงานของ Stepper Motor ในทิศทาง Counterclockwise ซึ่งความละเอียดของ Stepper Motor แปรผันตรงกับ Frequency ในการเกิด Loss Step เมื่อเทียบกับทิศทางการหมุนแบบ Clockwise เช่นกัน จะเห็นได้ว่าที่ Full Step มีค่า Frequency 3097 Hz และที่ความละเอียด 1/32 มีค่า Frequency 83200 Hz

## อภิปรายผล

ความเร็วของ Stepper Motor นั้นไม่สามารถที่จะเห็นถึงแนวโน้มได้ เพราะว่าบอร์ด MotorXplorer ที่ใช้สำหรับทำการทดลองมีแรงเสียดทานมากเกินไป ซึ่งทำให้ไม่สามารถที่จะทำให้เห็นแนวโน้มความสัมพันธ์ระหว่าง Speed และ Frequency

การการวิเคราะห์การทำงานของ Stepper Motor ในกรณีที่ใช้ความละเอียดของ Micro Step ต่างกัน แสดงให้เห็นถึงผลกระทบต่อการวัดค่าความเร็วจาก Encoder และค่าความถี่ที่เกิดขึ้นระหว่างการทำงาน โดยหลักการแล้ว เมื่อเพิ่มความละเอียดของ Micro Step ที่ใช้ในการควบคุม Stepper Motor ค่า Frequency จะเพิ่มสูงขึ้น เนื่องจาก Stepper Motor ต้องทำการหมุนหรือขยับตำแหน่งในจำนวนขั้นที่มากขึ้นภายในระยะเวลาเดียวกัน การเพิ่มความละเอียดของ Micro Step ทำให้ Stepper Motor สามารถทำงานได้อย่างนุ่มนวลและแม่นยำขึ้น โดยเฉพาะในงานที่ต้องการการควบคุมตำแหน่งอย่างละเอียด นอกจากนี้ ยังช่วยลดการสั่นสะเทือนในระหว่างการทำงาน ส่งผลให้ระบบมีเสถียรภาพและ

ประสิทธิภาพที่ดีขึ้น อย่างไรก็ตาม การเพิ่มความละเอียดนี้ต้องพิจารณาความสามารถของตัวขับมอเตอร์และตัวควบคุมว่า รองรับความถี่ที่สูงขึ้นได้หรือไม่ เพื่อให้ระบบทั้งหมดสามารถทำงานได้อย่างมีประสิทธิภาพและตอบสนองต่อความต้องการ ของการใช้งานได้อย่างเหมาะสม

การเลือกค่า Slope ใน Ramp Block เพื่อกำหนดค่าความถี่ที่จะเพิ่มขึ้นทุก ๆ 1 วินาทีนั้น ทางคณะผู้จัดทำได้ทำการทดสอบเพื่อศึกษาถึงความแตกต่างในการกำหนดค่า Slope ที่ต่างกัน โดยมีการใช้ค่า 50, 100 และ 150 ซึ่งเลือกทดสอบที่ความละเอียดแบบ Full Step ในทิศทาง Counterclockwise (DIR=1) และใช้ค่า 50, 100, 150 และ 200 ในความละเอียดแบบ Half Step ในทิศทาง Counterclockwise (DIR=1) จากตารางที่ เน้นไว้ว่า

Full Step		
Ramp	Rad/s	Frequency
150	-91.4	3059.7
100	-94.3	3103
50	-95.6	3097

ตารางที่ 55 แสดงการกำหนดค่า Slope ใน Ramp ที่ต่างกันเพื่อถูกค่าความเร็วจาก Encoder ที่ผ่านการ Butterworth Low Pass Filter และ Frequency ที่การทำงาน Full Step ในทิศทาง Counterclockwise

Half Step		
Ramp	Rad/s	Frequency
200	-125.6	8194.8
150	-205.7	13200.0
100	-208.1	13300.0
50	-150.3	96200.0

ตารางที่ 66 แสดงการกำหนดค่า Slope ใน Ramp ที่ต่างกันเพื่อถูกค่าความเร็วจาก Encoder ที่ผ่านการ Butterworth Low Pass Filter และ Frequency ที่การทำงาน Half Step ในทิศทาง Counterclockwise

### ข้อเสนอแนะ

1. ในการทดลองโดยใช้ Stepper Motor ไม่ควรใช้ติดต่อกันเป็นเวลานาน เนื่องจากมอเตอร์จะมีความร้อนมากเกินไป อาจจะต้องมีการติดตั้ง Heat Sink หรืออุปกรณ์ที่ช่วยระบายความร้อนในบอร์ดการทำงานเพิ่มเติม
2. การตั้งค่าความละเอียดในการทำงานของ Stepper Motor ควรเลือกค่าความถี่ใน Ramp Block ให้ดี เพื่อให้มอเตอร์ทำงานได้อย่างเหลี่ยม และมีประสิทธิภาพมากที่สุด

3. ควรจ่ายกระแสไฟฟ้า และแรงดันไฟฟ้าให้เหมาะสมตามที่ระบุ Datasheet เพื่อไม่ให้งจรเกิดความเสียหายเกิดขึ้น
4. ควรวางแผนการควบคุมในพื้นที่โล่ง และไม่ควรจับ Stepper Motor ขณะทำงาน เพราะว่าจะทำให้การเก็บค่านี้มีความผิดพลาดเกิดขึ้น
5. การย้ายไฟล์ RMX\_Motor.bin ต้องทำทันทีที่เขื่อมต่อสายอัปโหลดเข้าคอมพิวเตอร์
6. การประยุกต์ใช้การเพิ่มความถี่ (ความเร่งในการเพิ่มความถี่) ในลักษณะ S-Curve เพื่อเพิ่มความลื่นไหลในการสั่งงานและเพื่อให้ Stepper ทำงานได้โดยพยายามทำให้เกิด Loss Step ที่น้อยที่สุด

#### เอกสารอ้างอิง(แนบ link)

อ้างอิงขั้นตอนและผลการทดลองร่วมกับกลุ่ม A12

chrome-extension://efaidnbmnnibpcajpcglclefindmkaj/https://www.ti.com/lit/ds/symlink/drv8825.pdf

chrome-extension://efaidnbmnnibpcajpcglclefindmkaj/https://docs.rs-online.com/0330/A70000008880659.pdf

## การทดลองที่ 3 Brushless DC Motor

### จุดประสงค์

- เพื่อศึกษา และเข้าใจถึงหลักการทำงาน รวมถึงความสามารถของ Brushless DC Motor
- เพื่อศึกษา และเข้าใจถึงการใช้งานโปรแกรม Motor Workbench 6.3.2 ใน การตั้งค่า Stepper Motor ในเรื่องของ PWM Generation Frequency และ Speed Sensing
- เพื่อศึกษา และเข้าใจถึง BLDC Motor ในเรื่องของ Electrical parameters, Mechanical parameters และ Motor parameters
- เพื่อศึกษา และเข้าใจถึง BLDC motor control types ทั้ง 2 รูปแบบ ได้แก่ Sensorless Control และ Sensor-based Control
- เพื่อศึกษา และสามารถอธิบายถึงกราฟสัญญาณของ BLDC Motor ทั้ง 3 Phases ในเรื่องของพฤติกรรม, สาเหตุที่เกิดลักษณะกราฟนั้น ๆ เพื่อที่สามารถนำมำจำแนกถึงการควบคุมของ Motor โดยต้องอธิบายอย่างละเอียด และมีการวิเคราะห์ผ่านการคำนวนหรือทฤษฎีที่มีความน่าเชื่อถือประกอบผลการทดลองที่ต้องนำไปเปรียบเทียบกับพฤติกรรมการควบคุม PMSM แบบ FOC ซึ่งเนื้อหาที่นำมาอธิบาย และวิเคราะห์มีทั้งหมด 11 เรื่อง ได้แก่ Brushless DC (BLDC) Motor, Trapezoidal Back EMF, Permanent Magnet Synchronous Machine (PMSM), Trapezoidal Control, Field-Oriented Control (FOC), Pulse Width Modulation (PWM), Commutation Logic, Six-Step Commutation, Space Vector Modulation (SVM), Sensorless Control และ Hall Effect Sensors
- เพื่อศึกษา และเข้าใจถึงการคำนวนหาความเร็วของ BLDC Motor จากการดูกราฟ Frequency ผ่าน Oscilloscope
- เพื่อศึกษา และเข้าใจถึงการใช้งาน Oscilloscope 4 Channels ในการอ่านสัญญาณของ BLDC Motor ทั้ง 3 Phases

### สมมติฐาน

ถ้ามีการเปลี่ยนแปลงความเร็วของ Brushless DC Motor ในโปรแกรม Motor Control Workbench 6.3.2 จะสามารถคำนวนหาความเร็วข้อนกลับได้ตรงกับโปรแกรมด้วยการใช้ Oscilloscope เพื่อวัดหาค่า Frequency ของสัญญาณทั้ง 3 Phase

## ตัวแปร

ตัวแปรต้น : ความเร็วที่ตั้งค่าในโปรแกรม Motor Control Workbench 6.3.2 (RPM)

ตัวแปรตาม : ค่า Frequency ที่อ่านได้จาก Oscilloscope (Hz)

ตัวแปรควบคุม : แรงดันไฟฟ้าที่จ่ายเข้าบอร์ด, ชนิดของสายจัมเปอร์ที่ใช้เชื่อมต่อสายไฟ, ชนิดของบอร์ด Microcontroller และชนิดของ Oscilloscope

## นิยามศัพท์เฉพาะ

BLDC Motor หมายถึง Brushless DC Electric Motor หรือ มอเตอร์กระแสตรงแบบไร้แปรงถ่าน

สายจัมเปอร์ (Jumpers) หมายถึง คือสายที่ใช้สำหรับเชื่อมต่อระหว่าง Sensor เข้ากับบอร์ดทดลอง โมดูลต่าง ๆ หรือแหล่งจ่ายไฟเพื่อเชื่อมต่อวงจรเข้าด้วยกัน

บอร์ด Microcontroller หมายถึง อุปกรณ์ควบคุมขนาดเล็ก ซึ่งรวมเอาหน่วยประมวลผล, หน่วยความจำ และพอร์ตเข้า/ออกด้วยกัน เพื่อประมวลผลการทำงานที่สั่งงานคล้ายคลึงกับคอมพิวเตอร์

โรเตอร์ หมายถึง ส่วนที่หมุนของมอเตอร์

สเตเตอร์ หมายถึง ส่วนที่อยู่นิ่งของมอเตอร์ภายในการตอบสนองกระแสไฟฟ้าซึ่งเนี่ยวนำให้โรเตอร์หมุน

Commutator หมายถึง ส่วนที่เปลี่ยนทิศการไหลของกระแสไฟฟ้าภายในมอเตอร์ เพื่อให้มอเตอร์หมุนต่อเนื่องราบรื่น

แปรงถ่าน หมายถึง อุปกรณ์ในมอเตอร์ชนิดที่มีคอมมิวเตเตอร์ ใช้ในการส่งผ่านกระแสไฟฟ้าจากส่วนคงที่ของมอเตอร์ไปยังส่วนที่หมุนได้ โดยมักทำจากวัสดุคาร์บอนเพื่อให้ทนทานต่อการสึกหรอและลดการเกิดประกายไฟ

Torque หมายถึง แรงบิดที่เกิดจากแรงกระทำต่อวัตถุในระยะห่างจากจุดหมุน ทำให้เกิดการหมุนหรือการเคลื่อนไหวเชิงมุม มีหน่วยวัดเป็นนิวตันเมตร ( $N\cdot m$ )

สัญญาณ HIGH หมายถึง สถานะของสัญญาณไฟฟ้าที่มีแรงดันไฟฟ้าสูงกว่าค่ากำหนดในระบบดิจิทัล โดยทั่วไปเป็นค่าที่ใกล้เคียงกับแรงดันจ่ายไฟฟ้า เช่น 3.3V หรือ 5V

สัญญาณ LOW หมายถึง สถานะของสัญญาณไฟฟ้าที่มีแรงดันไฟฟ้าต่ำกว่าค่ากำหนดในระบบดิจิทัล โดยทั่วไปใกล้เคียงกับค่าศูนย์โวลต์ (0V)

Square-Wave หมายถึง สัญญาณไฟฟ้ารูปคลื่นสี่เหลี่ยมที่สลับระหว่างค่าสูงสุดและค่าต่ำสุดอย่างรวดเร็ว โดยมีลักษณะเป็นพัลส์ที่คุมชัดและใช้ในงานสัญญาณดิจิทัลหรือการควบคุมมอเตอร์

Sine Wave หมายถึง สัญญาณไฟฟ้ารูปคลื่นไอน์ที่เปลี่ยนแปลงอย่างราบรื่นตามเวลาซึ่งเป็นพื้นฐานของแรงดันไฟฟ้ากระแสสลับ (AC) ใช้ในการควบคุมมอเตอร์หรือระบบสื่อสาร

Hall State หมายถึง สถานะการทำงานของ Hall Sensor ที่ตรวจจับตำแหน่งหรือทิศทางการหมุนของมอเตอร์ผ่านสนามแม่เหล็ก โดยค่าที่ได้อาจเป็น HIGH หรือ LOW ขึ้นอยู่กับตำแหน่งแม่เหล็ก

Hall Sensor หมายถึง เซ็นเซอร์ตรวจจับสนามแม่เหล็กที่ใช้หลักการ Hall Effect เพื่อแปลงการเปลี่ยนแปลงของสนามแม่เหล็กเป็นสัญญาณไฟฟ้าสำหรับใช้ในมอเตอร์ BLDC หรือการตรวจจับตำแหน่ง

Oscilloscope หมายถึง อุปกรณ์วัดและแสดงผลรูปคลื่นสัญญาณไฟฟ้านานหน้าจอในรูปแบบกราฟของแรงดันไฟฟ้าตามเวลา ใช้สำหรับตรวจสอบสัญญาณและวิเคราะห์วงจร

แรงเฉียบ หมายถึง คุณสมบัติของวัตถุที่ต้านการเปลี่ยนแปลงการเคลื่อนที่ ไม่ว่าจะเป็นการเริ่มเคลื่อนที่ การหยุดหรือการเปลี่ยนแปลงความเร็ว

Motor Pilot หมายถึง ซอฟต์แวร์ที่พัฒนาโดย STMicroelectronics เพื่อช่วยในการควบคุมและกำหนดค่ามอเตอร์ในระบบที่ใช้ STM32 MCU ซอฟต์แวร์นี้มาพร้อมกับอินเทอร์เฟซแบบกราฟิกที่ช่วยให้ผู้ใช้สามารถปรับแต่งค่าพารามิเตอร์ เช่น การควบคุมแบบ FOC (Field Oriented Control), การรับPWM และการตรวจสอบข้อมูลมอเตอร์แบบเรียลไทม์ โดย Motor Pilot ถูกออกแบบมาเพื่อการทำงานร่วมกับ Library STM32 Motor Control SDK

## นิยามเชิงปฏิบัติการ

Brushless DC Motor หมายถึง Brushless DC Motor รุ่น A2212/13T/1000 KV

ความเร็ว洋洋กลับ หมายถึง การคำนวณหาความเร็วจากค่า Frequency ที่ได้จากการวัดด้วย Oscilloscope ค่าเฉลี่ย หมายถึง การหาค่าเฉลี่ยจากการหาความเร็ว洋洋กลับจำนวน 3 ค่า

เปอร์เซ็นต์ความผิดพลาด หมายถึง ค่าความคลาดเคลื่อนหรือผิดพลาดเป็นเปอร์เซ็นต์ของการคำนวณความเร็วย้อนกลับ เมื่อเทียบกับความเร็วที่กำหนดไว้จริง

Cursor หมายถึง สัญลักษณ์ในหน้าจอ Oscilloscope ที่ใช้ในการกดเลือกฟังก์ชันต่าง ๆ เพื่อควบคุมการใช้งานของ Oscilloscope ที่ทางคณะผู้จัดทำใช้เพื่อเลือกช่วงสัญญาณที่ต้องการ

กระเพื่อม หมายถึง สัญญาณที่ไม่ได้นิ่งหรือเป็นเส้นตรงเรียบ แต่เป็นสัญญาณที่มีการขึ้นลงเล็กน้อยตลอดเวลา

## เอกสารและงานวิจัยที่เกี่ยวข้อง

### 1. Datasheet A2212/13T/1000 KV Brushless DC Motor

A2212/13T/1000 KV Brushless DC Motor คือ DC Motor ชนิด Brushless ที่สามารถสร้างความเร็วรอบในการหมุนได้ 1000 RPM/V มอเตอร์มีเพลาเหล็กกล้าชุบแข็งขนาด 3.2 mm มีลูกปืนคู่ และมีขั้วต่อตัวผู้สปริงทองขนาด 3.5 mm ติดไว้ พร้อมทั้งมีขั้วต่อตัวเมีย 3 ตัวสำหรับควบคุมความเร็ว รูปด้านล่างจะแสดงถูกตอกย้ำว่ามีระยะห่างระหว่างจุดศูนย์กลาง 16 mm และ 19 mm และถูกเคลือบสำหรับสกรู 3 mm (M3)



รูปที่ 47 A2212/13T/1000 KV Brushless DC Motor

### 2. ลักษณะของ Brushless DC Motor

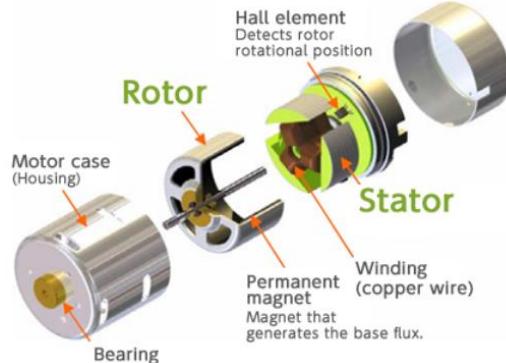
Brushless DC Motor เป็นมอเตอร์ที่ไม่มีแปรงถ่าน (Brush) ที่ใช้ในการส่งกระแสไฟฟ้าไปยัง Commutator เพื่อควบคุมการหมุนของมอเตอร์ ใน Brushless Motor จะใช้การควบคุมแบบอิเล็กทรอนิกส์แทนเพื่อควบคุมทิศทางการหมุน ความเร็ว และ Torque ของมอเตอร์ โดยมอเตอร์ชนิดนี้มีองค์ประกอบสำคัญ 2 ส่วน ได้แก่ โรเตอร์ (Rotor) และสเตเตอร์ (Stator)

โรเตอร์ หรือ ส่วนที่หมุนของมอเตอร์ ภายในจะประกอบไปด้วยแม่เหล็กถาวร (Permanent Magnet) เพื่อให้ตัวมันสามารถหมุนได้ตามสนามแม่เหล็กที่ถูกสร้างขึ้นจากสเตเตอร์ ซึ่งจะทำให้แม่เหล็กนั้นทำการจดจดหรือผลักกันจนทำให้โรเตอร์นั้นเกิดการเคลื่อนที่ขึ้น ซึ่งก็คือมอเตอร์ได้ทำงานแล้ว

สเตเตอร์ หรือ ส่วนที่อยู่นิ่ง ภายในจะประกอบไปด้วยชุดลวด ซึ่งเมื่อกระแสไฟฟ้าได้ไหลผ่านชุดลวดจะเกิดการสร้างสนามแม่เหล็กไฟฟ้า จนทำให้เกิดการดูดหรือผลักกันในตัวโรเตอร์เพื่อทำให้มอเตอร์ทำงาน

โดยสามารถแบ่งประเภทของ Brushless DC Motor ตามลักษณะการวางของโรเตอร์ได้ 2 แบบ ได้แก่

### 1. Outer Rotor



รูปที่ 48 แสดงลักษณะภายใน Outer Rotor

โรเตอร์จะอยู่ภายนอก สเตเตอร์จะอยู่ภายใน สามารถสังเกตได้ว่าขดลวดจะอยู่กับที่ภายใน ซึ่งข้อดีของประเภทนี้ มีดังนี้

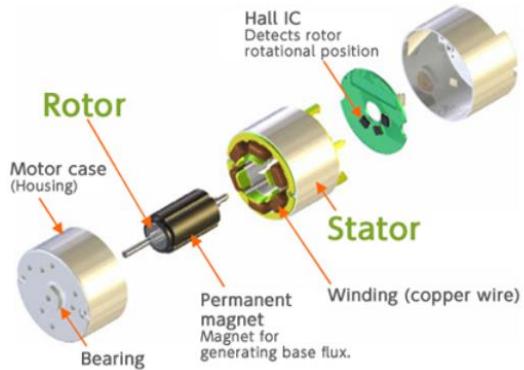
- สามารถรับทอร์คได้มากกว่าเนื่องจากขนาดของโรเตอร์จะใหญ่กว่า
- สร้างแรงบิดที่สูง
- มีแรงเฉียบสูงกว่า ทำให้เหมาะสมกับการทำงานที่ต่อเนื่อง

ในส่วนของข้อเสีย มีดังนี้

- เนื่องจากว่ามอเตอร์มีขนาดใหญ่ ทำให้น้ำหนักมากขึ้น
- มองเห็นไม่สามารถหมุนความเร็วสูงสุดได้เท่า Inner Rotor
- มีความเร็วรอบต่ำ
- มีการระบายความร้อนยากกว่า

โดยสรุปแล้ว Outer Rotor จะเหมาะสมกับการใช้งานในอุปกรณ์ที่ต้องการแรงบิดสูง แต่ความเร็วรอบต่ำ เช่น พัดลม เครื่องระบายอากาศ จักรยานไฟฟ้า เป็นต้น

## 2. Inner Rotor



รูปที่ 49 แสดงลักษณะภายใน Inner Rotor

โรเตอร์จะอยู่ภายใน สเตเตอร์จะอยู่ภายนอก สามารถสังเกตได้ว่าขนาดจะอยู่กับที่ภายนอก ซึ่งข้อดีของประเภทนี้ มีดังนี้

- เหมาะกับการทำงานในพื้นที่จำกัด
- มอเตอร์มีขนาดเล็ก และน้ำหนักเบา
- มีการตอบสนองที่ไวกว่า
- มีความเร็วรอบที่สูง
- มีการระบายความร้อนที่ดีกว่า

ในส่วนของข้อเสีย มีดังนี้

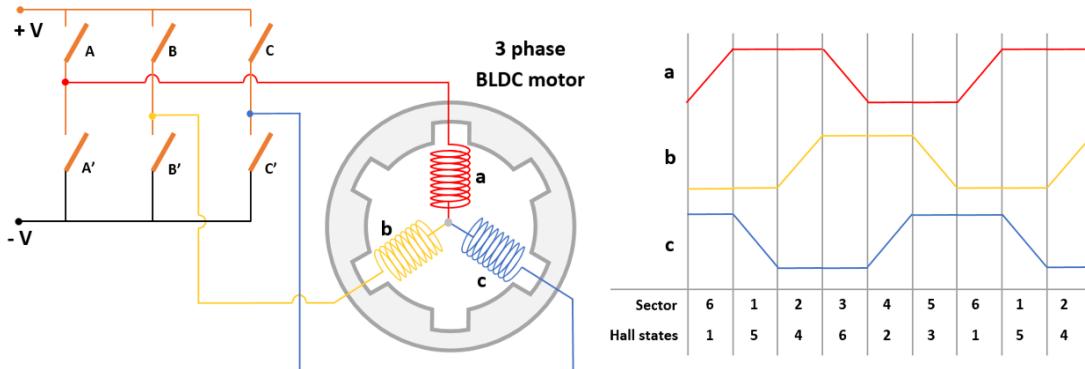
- ทำให้เกิดثارคุณสูงได้ยาก แม้เหล็กสามารถเสียหายได้จากแรงเหวี่ยงหนีศูนย์กลาง
- มีแรงเนื้อยต่ำกว่า ทำให้ต้องสนองต่อการทำงานได้อย่างรวดเร็ว
- สร้างแรงบิดต่ำ

โดยสรุปแล้ว Inner Rotor จะเหมาะสมกับการใช้งานในอุปกรณ์ที่ต้องการความเร็วรอบสูง แต่แรงบิดต่ำ และจำเป็นต้องการมอเตอร์ที่มีขนาดเล็กหรือน้ำหนักเบา เช่น โดรน สว่านไฟฟ้า เครื่องดูดฝุ่น เป็นต้น

### 3. วิธีการควบคุม Brushless DC Motor

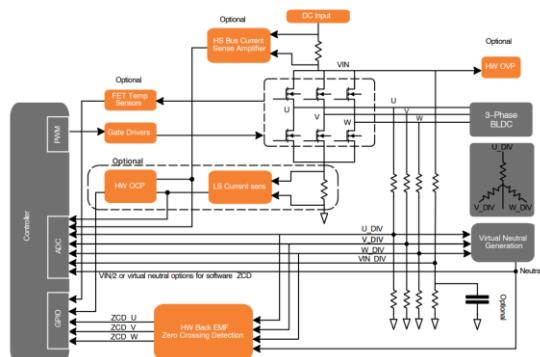
Brushless DC Motor สามารถแบ่งวิธีการควบคุมได้ 4 แบบ ได้แก่

#### 1. Trapezoidal Commutation



รูปที่ 50 แสดงลักษณะการควบคุม Brushless DC Motor แบบ Trapezoidal

Trapezoidal หรือ Six Step Control คือการควบคุม Brushless DC Motor ที่พื้นฐานที่สุด สามารถทำได้โดยการจ่ายกระแสไฟฟ้าในรูปแบบ Square-Wave ให้กับชด漉อดทีลิ่ง 2 จาก 3 ชด漉อดเพื่อควบคุมมอเตอร์ โดยมีรูปแบบการเปิดปิด MOSFET เพื่อควบคุมกระแสไฟฟ้าอยู่ 6 รูปแบบ สลับกันเป็นจังหวะเพื่อสร้างแรงบิดสูงสุด ในการควบคุมมอเตอร์ในรูปแบบนี้ จะต้องใช้เซนเซอร์ Hall Effect จำนวน 3 เซนเซอร์ที่ติดห่างกัน 120 องศาทางไฟฟ้าภายใน Brushless DC Motor เพื่อตรวจจับ Zero Crossing โดยการรับรู้สัญญาณ Back EMF ที่จะเกิดขึ้นจากชด漉อดที่ได้รับการจ่ายกระแสไฟฟ้า เพื่อให้รู้ตำแหน่งที่ถูกต้องในขณะนั้นของโรเตอร์ และนำมายังสร้างสัญญาณการจ่ายกระแสไฟฟ้าที่ถูกต้องต่อไป



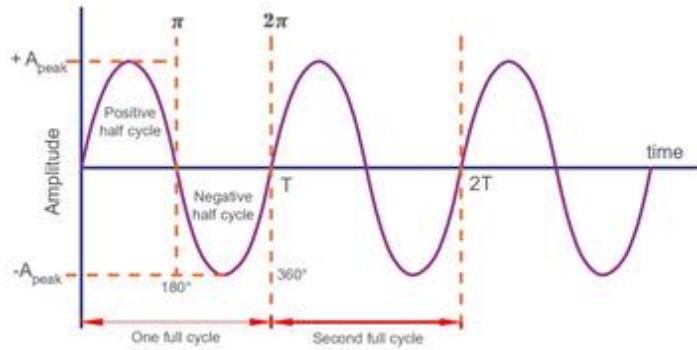
รูปที่ 51 แผนผังการควบคุม Brushless DC Motor แบบ Trapezoidal

การควบคุมแบบ Trapezoidal โดยการตรวจจับ Zero crossing ของ Back EMF สามารถทำได้โดยการทั้งใช้ซอฟต์แวร์ หรือฮาร์ดแวร์ ในกรณีของซอฟต์แวร์ การตรวจจับ Zero Crossing จะเป็นต้องมี ADC ที่มีอย่างน้อย 4 อินพุต (แรงดันไฟฟ้าในชุดลวดทั้ง 3 เฟส, VIN/2 หรือจุดอ้างอิงตรงกลาง) โดยแรงดันไฟฟ้าทั้งหมดต้องถูกแบ่งลงมาเพื่อให้เหมาะสมกับช่วงการวัดเต็มสเกลของ ADC ของคอนโทรลเลอร์ ในกรณีของฮาร์ดแวร์ การตรวจจับ Zero Crossing จะใช้ Comparator เพื่อเปรียบเทียบแรงดันไฟฟ้าในชุดลวดกับจุดอ้างอิง และส่งสัญญาณ Zero Crossing ไปยังคอนโทรลเลอร์โดยตรงผ่านขา GPIO กรณีหากใช้ฮาร์ดแวร์ เพื่อตรวจจับ Zero Crossing แนะนำให้กรองแรงดันไฟฟ้าในชุดลวดที่ถูกแบ่งไว้เพื่อกรองสัญญาณรบกวนจากการสลับ PWM และเพื่อป้องกันสัญญาณผิดพลาดใน ZCD ที่เกิดจากสัญญาณ Back EMF ที่ไม่สมบูรณ์และมีสัญญาณรบกวน อย่างไรก็ตามการกรองสัญญาณอาจทำให้เกิดความล่าช้าได้ ทำให้ความเร็วตอบสนองสูงสุดที่สามารถทำได้ของมอเตอร์ ถูกจำกัดลงมา

## 2. Field Oriented Control (FOC)

เป็นการควบคุมมอเตอร์ในระบบปิดที่ใช้สัญญาณกระแส และมุ่งมาจากเซนเซอร์ เพื่อนำมาคำนวณความเร็ว ตำแหน่ง และแรงบิดมอเตอร์ โดยต้องทำการควบคุมสนามแม่เหล็กทุกทิศทางผ่านการสลับ MOSFET เพื่อเปิดกระแสไฟฟ้าไฟล์ไปตามชุดลวด และสามารถเปิด MOSFET ทั้งสามตัว เพื่อสร้างสนามแม่เหล็กที่แรงกว่าการเปิดเพียงสองตัว มีการควบคุมความแรงของสนามแม่เหล็กผ่านการที่เปิด MOSFET ทั้งสามตัวแล้ว ชุดลวดจะไม่มีแรงดันไฟฟ้า เพราะชุดลวดเห็นยืนทำให้ไม่เกิดการเปลี่ยนแปลงกระแสไฟฟ้าทันทีทันใด แต่กระแสไฟฟ้าจะถลายตัวได้ตามอัตราที่กำหนดไว้ จากเดิมที่จะลดลงเป็นศูนย์ทันที มีการจัดตำแหน่งของโรเตอร์ และสนามแม่เหล็ก โดยการใช้เซนเซอร์วัดมุ่งเพื่อตรวจจับตำแหน่งที่โรเตอร์อยู่ในปัจจุบัน เพื่อสร้างสนามแม่เหล็กให้ตั้งฉากในทิศทางที่ถูกต้อง เพื่อให้โรเตอร์นั้นมีการดูด หรือผลักกันเกิดขึ้น ซึ่งเมื่อโรเตอร์มีการเคลื่อนที่ สนามแม่เหล็กจะพยายามทำให้ตั้งฉากกันอยู่เสมอ จากการควบคุมสนามแม่เหล็กทุกทิศทาง ความแรงของสนามแม่เหล็ก และจัดตำแหน่งของโรเตอร์กับสนามแม่เหล็กให้ตั้งฉากกัน ทำให้สามารถควบคุมมอเตอร์นี้ได้ เช่น หากเพิ่มความแรงสนามแม่เหล็กจะทำให้มอเตอร์หมุนเร็วขึ้น หากโรเตอร์เกิดการเปลี่ยนแปลงก็สามารถใช้สนามแม่เหล็กจัดตำแหน่งโรเตอร์ใหม่ เป็นต้น

### 3. Sinusoidal Commutation



รูปที่ 52 แสดงลักษณะการควบคุมแบบ Sinusoidal

เป็นการควบคุม Brushless DC Motor โดยใช้กระแสไฟฟ้าจ่ายเข้ามอเตอร์ในรูปแบบ Sine Wave หรือคลื่นไอน์ ที่มีระยะห่างในแต่ละลูกคลื่น 120 องศาทางไฟฟ้า เพื่อให้สามารถแม่เหล็กมีความไฟลีน ไม่กระชากเหมือนกับการใช้รูปแบบ Square Wave หรือคลื่นสี่เหลี่ยม เพราะมีการใช้ Space Vector Modulation (SVM) หรือ Sinusoidal PWM โดยการปรับความเร็วจะขึ้นอยู่กับการปรับความถี่ของคลื่นที่จ่ายกระแสไฟฟ้าไปยังชุดลวดในสเตเตอร์ ถ้าเพิ่มความถี่จะทำให้โรเตอร์หมุนเร็วขึ้น ลดความถี่จะทำให้โรเตอร์หมุนช้าลง นอกจากนี้มักจะมีการใช้เซนเซอร์ Hall Effect เพื่อรู้ถึงตำแหน่งของโรเตอร์ เพื่อควบคุมการจ่ายกระแสไฟฟ้าไปยังชุดลวดในสเตเตอร์

### 4. Back EMF

เป็นการควบคุมแบบไม่ใช้เซนเซอร์ Hall Effect แต่อาศัยการใช้ Back EMF เพื่อหาตำแหน่งของโรเตอร์ โดย Back EMF คือแรงดันไฟฟ้าที่เกิดขึ้นย้อนกลับ เมื่อมอเตอร์เกิดการหมุนจะสร้างแรงดันไฟฟ้าในทิศทางตรงกันข้ามกับแรงดันที่จ่ายเข้ามาในมอเตอร์ ซึ่งเกิดมาจากการเคลื่อนที่ของสนามแม่เหล็กที่ผ่านชุดลวดในมอเตอร์ ความแรงของ Back EMF จะแปรผันตรงกับความเร็วในการหมุนของโรเตอร์ ซึ่งจำเป็นต้องคำนวณค่า Back EMF ให้ถูกต้องเพื่อให้สามารถจ่ายกระแสไฟฟ้าได้อย่างเหมาะสม โดยมีสูตรคำนวณ ดังนี้

$$E_b = k_b \times \omega$$

โดยที่

$$E_b = \text{Back EMF}$$

$$k_b = \text{ค่าคงที่ของมอเตอร์}$$

$$\omega = \text{ความเร็วเฉลี่ยของโรเตอร์ (Rad/s)}$$

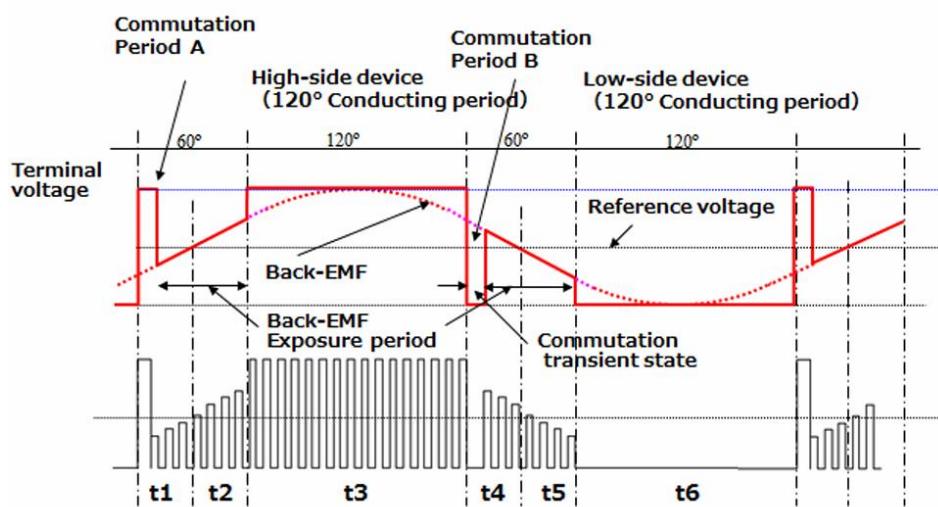
จากวิธีการควบคุมทั้ง 4 แบบ ได้แก่ Trapezoidal Commutation, Field Oriented Control (FOC), Sinusoidal Commutation และ Back EMF สามารถนำมาเปรียบเทียบถึงข้อดีข้อเสียในแต่ละประเภทได้ดังนี้

ประเภทการควบคุม	ข้อดี	ข้อเสีย
Trapezoidal Commutation	<ul style="list-style-type: none"> <li>- ใช้งานง่าย</li> <li>- ต้นทุนต่ำ</li> <li>- ระบบควบคุมง่าย</li> </ul>	<ul style="list-style-type: none"> <li>- แรงบิดไม่สม่ำเสมอ</li> <li>- มีการสั่นสะเทือน</li> <li>- เสียงดัง</li> </ul>
Field Oriented Control (FOC)	<ul style="list-style-type: none"> <li>- ความแม่นยำในแรงบิด และความเร็ว</li> <li>- ประสิทธิภาพสูง</li> <li>- ประหยัดพลังงาน</li> </ul>	<ul style="list-style-type: none"> <li>- ต้องใช้อุปกรณ์การควบคุม</li> <li>- ระบบซับซ้อน</li> <li>- ต้นทุนสูง</li> </ul>
Sinusoidal Commutation	<ul style="list-style-type: none"> <li>- ประสิทธิภาพสูง</li> <li>- แรงบิดสม่ำเสมอ</li> <li>- เสียงเบากว่า Trapezoidal</li> </ul>	<ul style="list-style-type: none"> <li>- มีการควบคุมที่ซับซ้อน</li> <li>- ต้นทุนสูง</li> </ul>
Back EMF	<ul style="list-style-type: none"> <li>- ต้นทุนต่ำ</li> <li>- ใช้งานง่าย</li> <li>- ใช้กับระบบที่ไม่มีเซนเซอร์</li> </ul>	<ul style="list-style-type: none"> <li>- ไม่สามารถใช้ได้กับทุกความเร็วในการหมุน</li> <li>- ข้อมูลอาจผิดพลาดได้</li> </ul>

ตารางที่ 7 แสดงการเปรียบเทียบข้อดี และข้อเสียของการควบคุม Brushless Dc Motor แต่ละประเภท

#### 4. การเกิด Back-EMF จากการควบคุมแบบ Trapezoidal

Back-EMF ถูกแบ่งออกเป็น 2 ส่วน คือช่วงที่ Permanent Magnet กำลังเคลื่อนที่เข้าใกล้ ชด漉วดที่พันรอบแกนสเตเตอร์ กับช่วงที่เคลื่อนที่ออกจากชด漉วดที่พันรอบแกนสเตเตอร์ โดยสัญญาณจะ ออกมาเป็น Sine Wave แต่ความเป็นจริงนั้นสัญญาณ Back-EMF ถูกทับด้วยสัญญาณ PWM ที่สั่นการ Brushless DC Motor ทำให้สัญญาณที่ออกมามีความคล้ายกับรูปสี่เหลี่ยมคงที่จึงถูกเรียกว่า Trapezoidal



รูปที่ 53 แสดงสัญญาณ Back-EMF ที่ซ่อนทับกับสัญญาณ PWM

#### 5. Synchronous Speed

Synchronous Speed คือค่าความเร็วรอบการหมุนของมอเตอร์ที่เกี่ยวข้องกับค่าความถี่ในการ สับเปลี่ยนขั้วของชด漉วดไฟฟ้าในมอเตอร์ โดยสามารถหา Synchronous Speed ในหน่วย RPM (Revolutions per Minute) ได้จากสมการ ดังนี้

$$N_s = 60 \frac{f}{P} = 120 \frac{f}{p}$$

และสามารถหา Synchronous Speed ในหน่วย Rad/s ได้จากสมการ

$$\omega_s = 2\pi \frac{f}{P} = 4\pi \frac{f}{p}$$

เมื่อ

$$f = \text{ความถี่การเปลี่ยนขั้วของกระแสไฟฟ้า}$$

$$p = \text{จำนวนขั้วของแม่เหล็กไฟฟ้า}$$

$$P = \text{จำนวนคู่ขั้วของแม่เหล็กไฟฟ้า}$$

## 6. Permanent Magnet Synchronous Machine (PMSM)

Permanent Magnet Synchronous Machine (PMSM) คือ คือเครื่องจักรไฟฟ้าชนิดหนึ่งที่ใช้แม่เหล็กถาวรเป็นตัวสร้างสนามแม่เหล็กในโรเตอร์ และทำงานในลักษณะของมอเตอร์หรือเครื่องกำเนิดไฟฟ้าแบบซิงโครนัส โดยที่ความเร็วของโรเตอร์จะสัมพันธ์กับความถี่ของกระแสไฟฟ้าในสเตเตอร์ (ทำให้ไม่เกิดการลื่นไถลเหมือนในมอเตอร์恒速 หมุน慢 ทั่วไป) PMSM มีคุณสมบัติเด่นด้านประสิทธิภาพที่สูงเนื่องจากการใช้แม่เหล็กถาวรซึ่งลดการสูญเสียพลังงานในรูปแบบความร้อน และสามารถสร้างแรงบิดที่สูงในขนาดเครื่องจักรที่เล็กกว่าเมื่อเทียบกับมอเตอร์ประเภทอื่น นอกจากนี้ PMSM ยังบำรุงรักษาง่าย เพราะไม่มีแปรงถ่านหรือคอมมิวเตอร์เหมือนในมอเตอร์กระแสตรง ทำให้มีความทนทานต่อการใช้งานต่อเนื่อง ส่วนการควบคุม PMSM มักใช้อินเวอร์เตอร์เพื่อจ่ายกระแสไฟฟ้าที่ควบคุมความถี่และเพลิดอย่างแม่นยำ ช่วยให้สามารถปรับความเร็วและแรงบิดได้ตามต้องการ PMSM นิยมใช้งานในหลากหลายอุตสาหกรรม เช่น ยานยนต์ไฟฟ้า (Electric Vehicles) เครื่องจักร CNC กังหันลม และระบบปั๊มที่ต้องการความแม่นยำและประสิทธิภาพสูง

## 7. Pulse Width Modulation (PWM)

Pulse Width Modulation (PWM) คือการสร้างสัญญาณ Pulse width ในรูปแบบดิจิตอล เพื่อควบคุมพลังงานเฉลี่ยที่ต้องการไปให้กับกรณีทางไฟฟ้า โดยมีหลักการในการควบคุมการจ่ายไฟโดยการสร้างสัญญาณคลื่นสี่เหลี่ยมสลับกันระหว่าง HIGH และ LOW ซึ่งสามารถควบคุมได้อย่างอนาคตจากความกว้างของสัญญาณ Pulse width โดยเรียกสัญญาณครบหนึ่งรอบว่า Time period และสามารถหาค่า Time period โดยใช้สมการ

$$Time period = On time + Off time$$

เมื่อ

*On time* = ช่วงเวลาที่สัญญาณเป็น HIGH

*Off time* = ช่วงเวลาที่สัญญาณเป็น LOW

ในการสั่งงาน PWM จะเป็นต้องใช้ความถี่ในการสั่งงานสัญญาณไฟฟ้าให้เหมาะสมกับอุปกรณ์ไฟฟ้า โดยความถี่ในการสับเปลี่ยนสัญญาณใน 1 Time period จะหาได้จาก สมการ

$$Frequency = \frac{1}{Time period}$$

เมื่อ

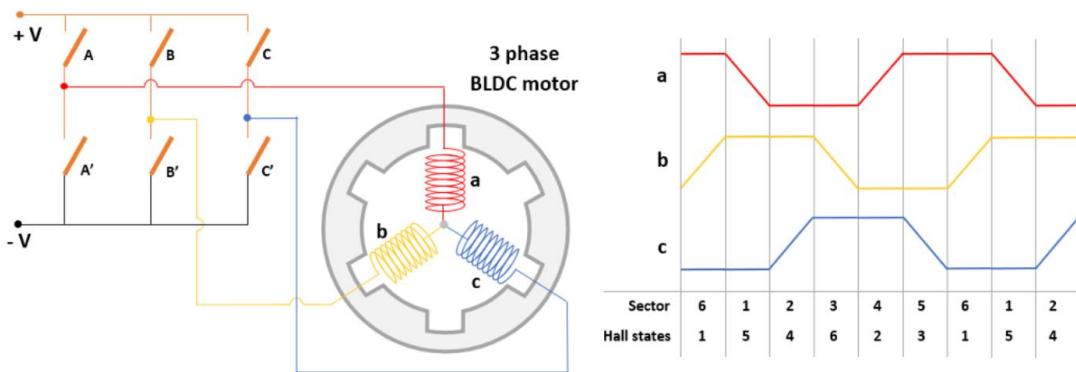
*Time period* = ช่วงเวลาที่สัญญาณจ่ายครบ 1 รอบ

ในการสั่งงาน PWM สัดส่วนการสั่งสัญญาณHIGH ใน 1 Time period มีผลต่อการทำงานของอุปกรณ์ไฟฟ้าคล้ายการปรับแรงดันในรูปแบบอนาล็อก อัตราส่วนระหว่างสัญญาณHIGH ใน 1 Time period เรียกว่า Duty cycle โดยมีสมการคือ

$$Duty cycle = \frac{On time}{Time period}$$

## 8. Commutation Logic & 6 Step Commutation

Commutation Logic หมายถึง โลจิคการทำงานของ Commutator เพื่อให้การส่งกระแสไฟฟ้าให้ขาด漉ดไฟฟ้าเป็นไปได้อย่างราบรื่นตามลำดับที่ควรจะเป็นในการทำงาน สำหรับการใช้งาน Brushless DC motor ที่ใช้ขาด漉ด 3 เพส เราจะใช้ Commutation Logic ในรูปแบบ 6 Step Commutation ซึ่งในการสั่งงานดังกล่าวจำเป็นต้องมี MOSFET ในการสั่งงาน 6 ชิ้น ประกอบด้วย A, B, C, A', B' และ C' ซึ่งจะเชื่อมต่อเข้ากับขาด漉ดไฟฟ้า A, B และ C ในลักษณะตามรูปที่ปรากฏ

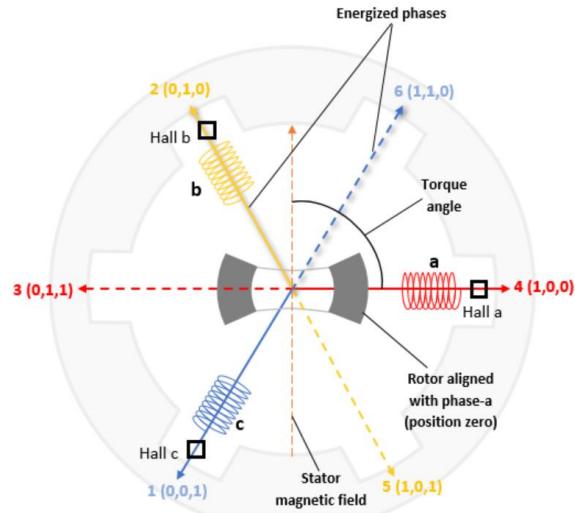


รูปที่ 54 แสดงถึงการเชื่อมต่อในรูปแบบ 6 Step Commutation และลำดับ Logic การสั่งงาน

เมื่อกำหนดให้ Hall State 1 คือตำแหน่งที่ Hall c, Hall State 2 คือตำแหน่งที่ Hall b, Hall State 3 คือตำแหน่งที่ระหว่าง Hall b และ Hall c, Hall State 4 คือตำแหน่งที่ Hall a, Hall State 5 คือตำแหน่งที่ระหว่าง Hall a และ Hall c , Hall State 6 คือตำแหน่งที่ระหว่าง Hall a และ Hall b จะได้ลำดับการสั่งงานที่ขึ้นอยู่กับองศาปัจจุบันของมอเตอร์ ดังตารางนี้

Hall State (Hall a, Hall b, Hall c)	ลำดับการสับเปลี่ยน (AA' BB' CC')		
	AA'	BB'	CC'
4 (100)	00	10	01
6 (110)	01	10	00
2 (010)	01	00	10
3 (011)	00	01	10
1 (001)	10	01	00
5 (101)	10	00	01

ตารางที่ 78 แสดงลำดับการสั่งงานสับเปลี่ยนที่ขึ้นอยู่กับองศาปัจจุบันของมอเตอร์



รูปที่ 55 แสดงภาพตำแหน่งหมายเลข Hall State ที่ขึ้นอยู่กับองศาปัจจุบันของมอเตอร์

## 9. Space Vector Modulation (SVM)

หมายถึง เทคนิคการควบคุมแรงดันไฟฟ้าในมอเตอร์แบบ 3 เฟส โดยการสร้างสัญญาณ PWM เพื่อควบคุมแรงดันที่จ่ายให้กับมอเตอร์ ซึ่ง SVM จะใช้วิเคราะห์ของการแทนแรงดันไฟฟ้าในรูปแบบ เวกเตอร์บนแกนเชิงซ้อน (Complex Plane) เพื่อให้ได้รูปคลื่นแรงดันที่ใกล้เคียงกับไชน์เฟที่สุด โดยแบ่ง การทำงานออกเป็น 6 โฉนหลักและแต่ละโฉนจะกำหนดค่าของสวิตช์ (Switching State) ที่เหมาะสมเพื่อลดการสูญเสียพลังงานและเพิ่มประสิทธิภาพของมอเตอร์

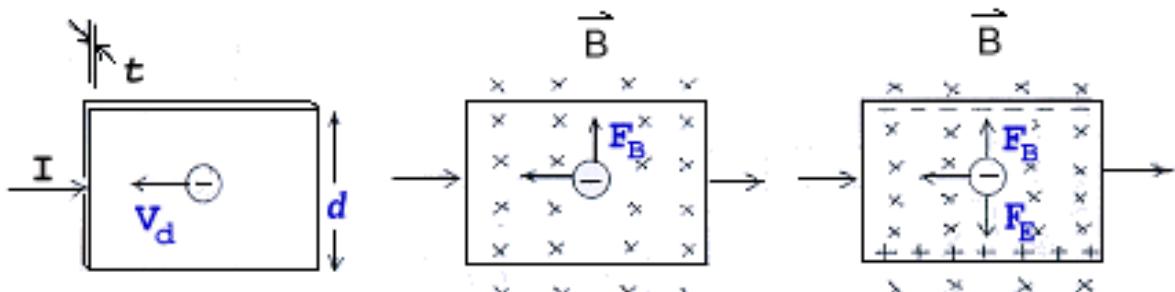
## 10. ปรากฏการณ์ฮอลล์ (Hall Effect)

หมายถึง การเกิดสนามไฟฟ้าในตัวนำบางในทิศทางที่ตั้งฉากกับทั้งกระแสไฟฟ้าและ สนามแม่เหล็ก ค้นพบในปีค.ศ. 1879 โดยคุณเอ็ดวิน ฮอลล์ (Edwin Hall) นักศึกษามหาวิทยาลัยจอห์น ฮอพคินส์ อายุ 24 ปี พบร่วมเมื่อนำแหนงตัวนำบางที่มีกระแสไฟฟ้าผ่านไปในบริเวณที่มีสนามแม่เหล็ก พาหะประจุ (Charge Carriers) ในตัวนำสามารถเบนไปจากแนวทางเดิมได้ และทำให้เกิดปรากฏการณ์ ฮอลล์

จากรูปด้านล่างแสดงแผลนตัวนำบางที่มีขนาดความกว้าง  $d$  หนา  $t$  และมีกระแสไฟฟ้า ขนาด  $I$  ผ่านในทิศจากด้านซ้ายไปด้านขวา พาหะประจุคืออิเล็กตรอนเคลื่อนที่ ในทิศตรงข้ามกับกระแสไฟฟ้า จากด้านขวาไปด้านซ้าย

ต่อมาเมื่อใส่สนามแม่เหล็ก  $B$  ในทิศพุ่งเข้าหาและตั้งฉากกับระนาบแผลนตัวนำบางหรือกระดาษ จะเกิดแรงแม่เหล็ก  $F_B$  กระทำกับอิเล็กตรอน ทำให้อิเล็กตรอนเบนไปทางขอบด้านบนของแผลนตัวนำบาง เมื่อเวลาผ่านไปจะมีจำนวนอิเล็กตรอนถูกผลักไปที่ขอบด้านบนมากขึ้น ส่วนขอบด้านล่างของกระดาษจะ

เกิดประจุไฟฟ้าบวกจำนวนมากขึ้นแทนที่อิเล็กตรอน การที่มีประจุไฟฟ้าต่างชนิดรวมตัวกันในขอบ ตำแหน่งทั้งสอง ทำให้เกิดสนามไฟฟ้า เรียกว่า สนามไฟฟ้าฮอลล์ (Hall Field) ในแผ่นตัวนำบางมีพิษจาก ขอบด้านล่างไปขึ้นด้านบน สนามไฟฟ้าจะทำให้เกิดแรงไฟฟ้า FE กระทำกับอิเล็กตรอนทำให้อิเล็กตรอน ถูกผลักไปทางขอบด้านล่าง เมื่อแรงไฟฟ้าและแรงแม่เหล็กมีขนาดเท่ากัน อิเล็กตรอนจะเคลื่อนที่ในทิศไป ทางซ้ายโดยไม่เบนออกข้าง



รูปที่ 56 ภาพแสดงการเกิดประภารณ์ฮอลล์

### 11. Zero Crossing

หมายถึง จุดที่สัญญาณไฟฟ้าผ่านค่าศูนย์ (0) โดยเปลี่ยนจากค่าเป็นบวกไปลบหรือจากค่าลบไป บวก ซึ่งเป็นเหตุการณ์สำคัญที่ใช้ในระบบตรวจจับเฟส (Phase Detection) และการควบคุมไฟฟ้า เช่น การสลับโหลดในวงจรไฟฟ้ากระแสสลับ (AC) เพื่อหลีกเลี่ยงการเกิดกระแสกระชาก (Surge Current) ขณะสลับโหลด

### 12. Op-Amp

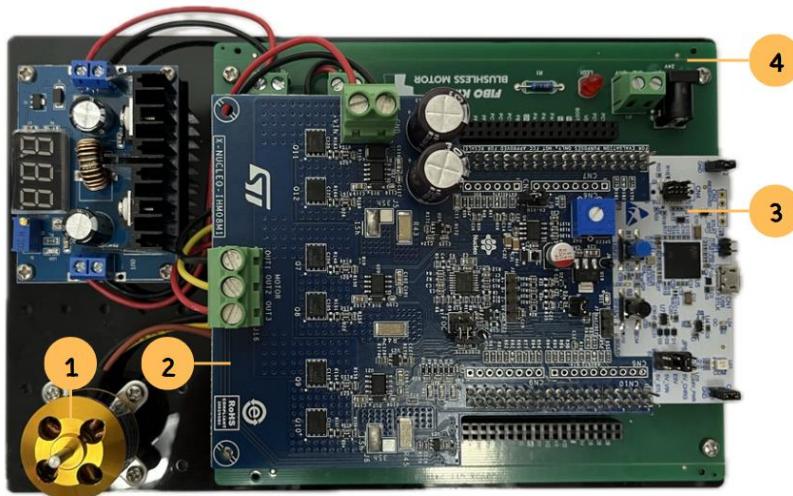
หมายถึง วงจรขยายสัญญาณไฟฟ้าที่มีอินพุตสองช่อง (Inverting และ Non-Inverting) และ เอ้าต์พุตหนึ่งช่อง โดยมีคุณสมบัติขยายแรงดันที่อินพุตไปยังเอ้าต์พุตตามค่าการตั้งค่าของวงจร สามารถใช้ งานได้หลากหลาย เช่น วงจรขยายสัญญาณ (Amplifier), วงกรองความถี่ (Filter), วงรบก-ลบ สัญญาณ (Summing) และวงเครื่องเปรียบเทียบ (Comparator)

## วิธีดำเนินการทดลอง

การทดลองเรื่อง Brushless DC Motor สามารถแบ่งการทดลองออกเป็น 3 ส่วน ส่วนแรก คือ การติดตั้งโปรแกรมรวมถึงการตั้งค่าอุปกรณ์ต่าง ๆ เช่น มอเตอร์, บอร์ด, การควบคุม และโปรแกรม Motor Control Workbench เป็นต้น ส่วนที่สอง คือ การออกแบบการทดลอง และทำการทดลอง ซึ่งคือการวัดสัญญาณด้วย Oscilloscope โดยที่เพิ่มหรือลดความเร็วตามที่กำหนดไว้ ส่วนที่สาม คือ การวิเคราะห์กราฟสัญญาณ เพื่อศึกษาถึงพฤติกรรมของ Brushless DC Motor ในความเร็วแต่ละช่วง เช่น Duty Cycle, Back-EMF, Phase Shift ฯลฯ ซึ่งในส่วนนี้จะมีการคำนวณเพื่อหาค่าที่ก่อร่อง เพื่อนำไปวิเคราะห์ต่อไป

## วัสดุอุปกรณ์

1. BLDC Motor จำนวน 1 ชิ้น
2. STMICROELECTRONICS X-NUCLEO-IHM08M1 จำนวน 1 ชิ้น
3. Nucleo STM32G474RE พร้อมสายอัปโหลด จำนวน 1 ชุด
4. BLDCXplorer จำนวน 1 ชุด



รูปที่ 57 แสดงองค์ประกอบภายในบอร์ดการทดลอง Brushless DC Motor

## ขั้นตอนการดำเนินงาน

- ติดตั้งโปรแกรม Motor Control Workbench 6.3.2



รูปที่ 58 แสดงหน้าตา Motor Control Workbench 6.3.2

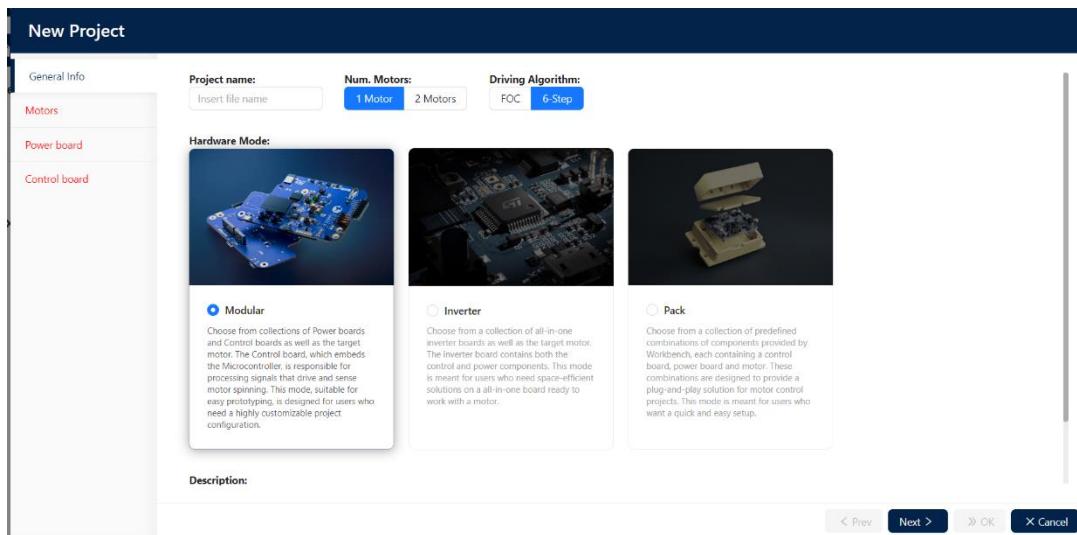
- ทำการกดสร้างไฟล์ใหม่ที่ New Project จากนั้นทำการใส่ข้อมูลในส่วน General Info ดังนี้

2.1 ตั้งชื่อไฟล์ที่ Project name

2.2 เลือกจำนวน Num Motor เป็น 1 Motor เพราะว่าในบอร์ดที่ได้มามี Brushless DC Motor 1 ตัว

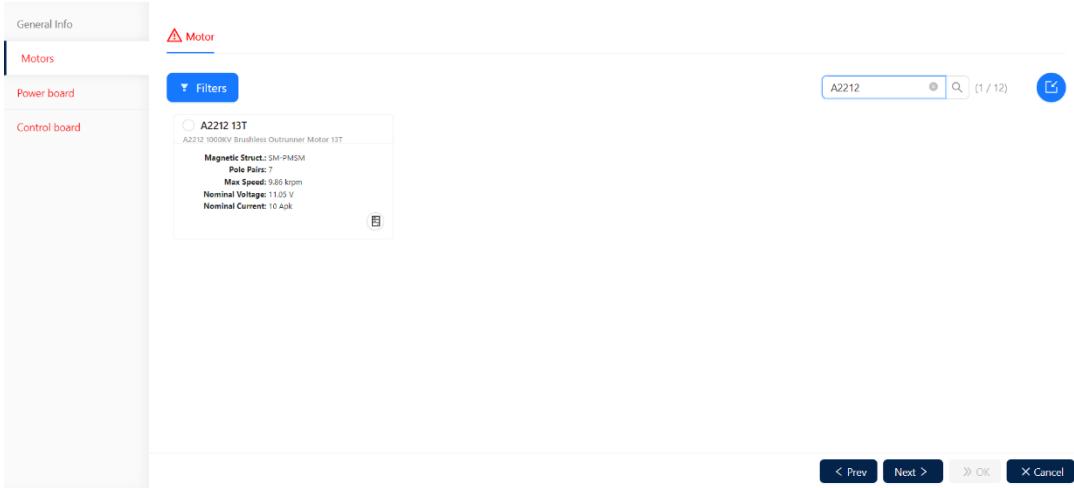
2.3 เลือก Driving Algorithm เป็นแบบ 6-Step

2.4 Hardware Mode เลือกเป็น Modular



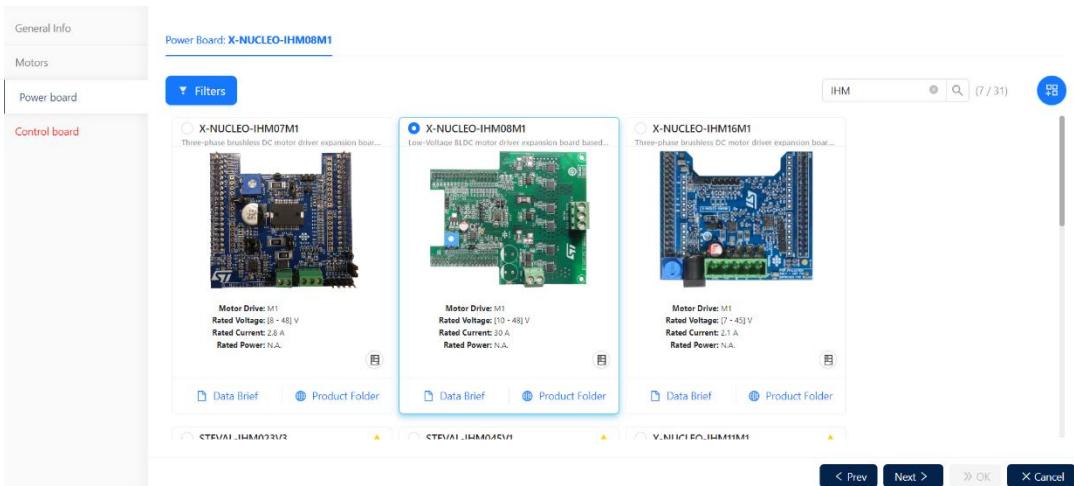
รูปที่ 59 แสดงการตั้งค่า General Info ในโปรแกรม Motor Control Workbench

3. ในส่วนของ Motors ให้เลือกเป็นรุ่นที่ใช้ในบอร์ดคือ A2212 13T



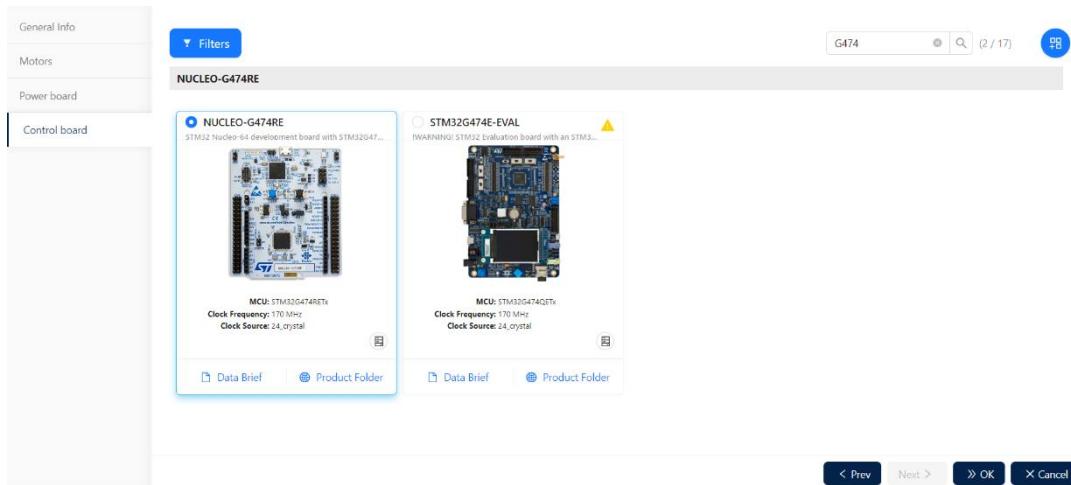
รูปที่ 60 แสดงการตั้งค่า Motors ในโปรแกรม Motor Control Workbench

4. ในส่วนของ Power board ให้เลือกเป็น X-NUCLEO-IHM08M1



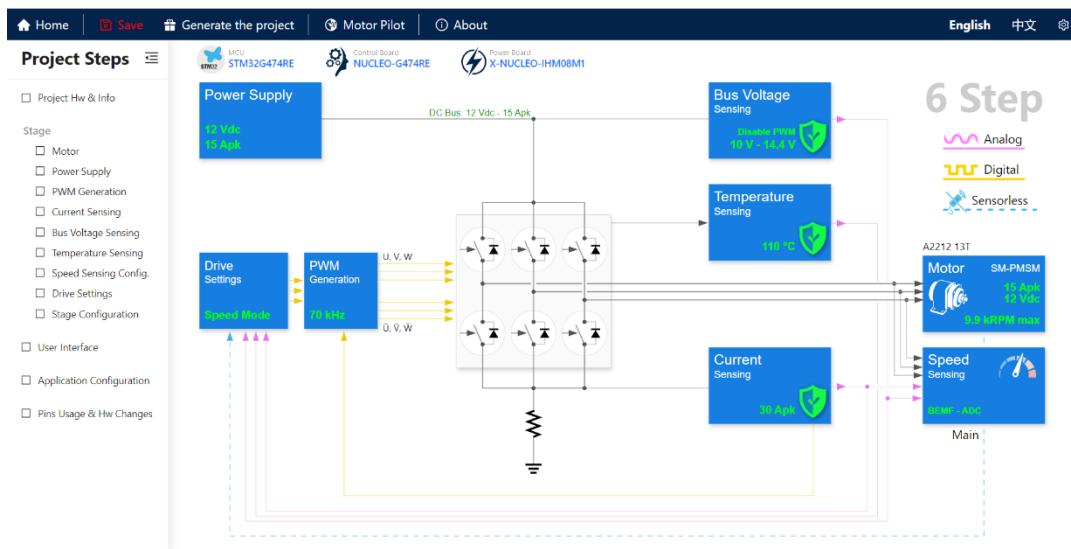
รูปที่ 61 แสดงการตั้งค่า Power board ในโปรแกรม Motor Control Workbench

## 5. ในส่วนของ Control board ให้เลือกเป็น NUCLEO-G474RE



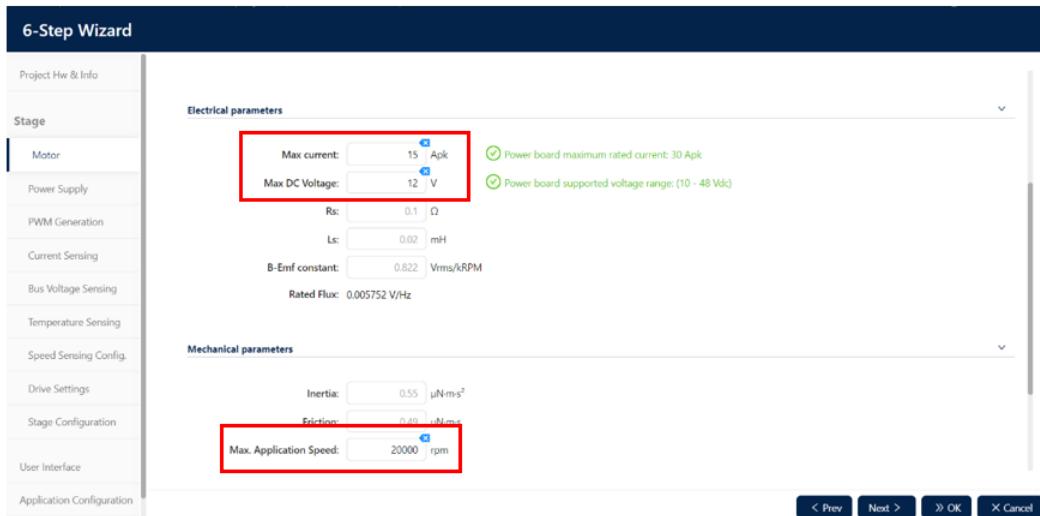
รูปที่ 62 แสดงการตั้งค่า Control board ในโปรแกรม Motor Control Workbench

## 6. เมื่อทำการตั้งค่าเสร็จสิ้นจึงกดปุ่ม OK โปรแกรมจะทำการโหลดไปยังหน้า Project Step



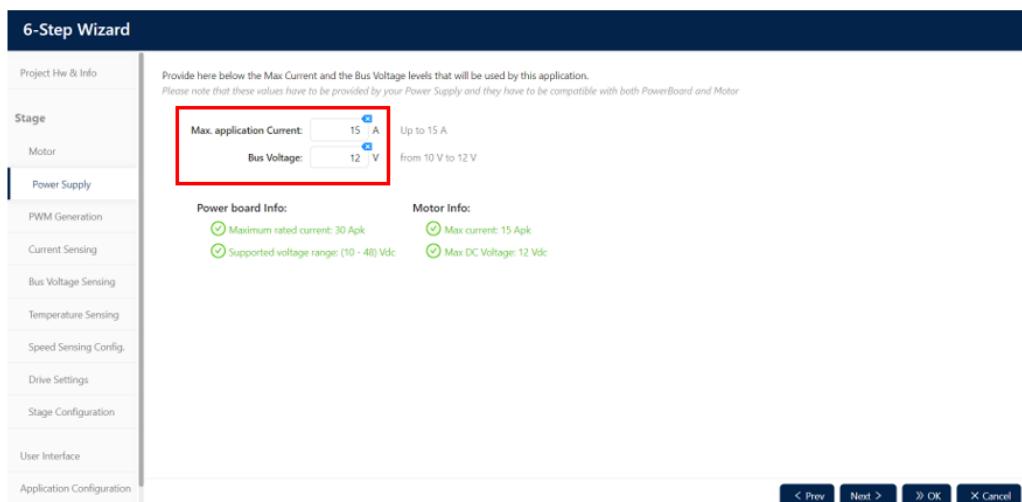
รูปที่ 63 แสดงหน้าตา Project Step หลังจากทำการตั้งค่าเรียบร้อย

7. ทำการตั้งค่าข้อมูลใน 6-Step Wizard โดยส่วน Motor ให้ตั้งค่า Max current เป็น 15 Apk, Max DC Voltage เป็น 12 V และ Max Application Speed เป็น 20000 RPM



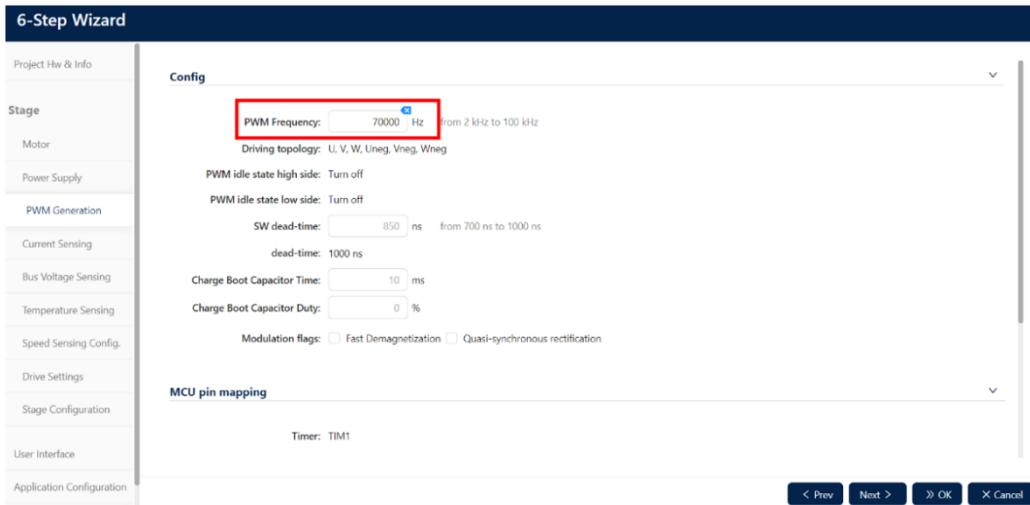
รูปที่ 64 แสดงการตั้งค่า Motor ใน 6-Step Wizard

8. ในส่วน Power Supply ตั้งค่า Max application Current เป็น 15 A และ Bus Voltage เป็น 12 V



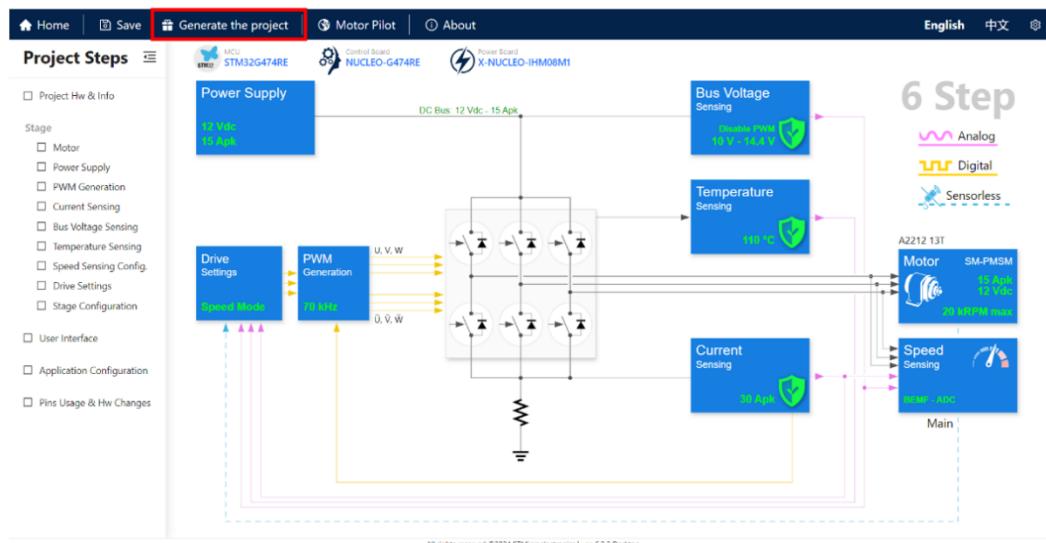
รูปที่ 65 แสดงการตั้งค่า Power Supply ใน 6-Step Wizard

9. ในส่วน PWM Generation ตั้งค่า PWM Frequency เป็น 70000 Hz



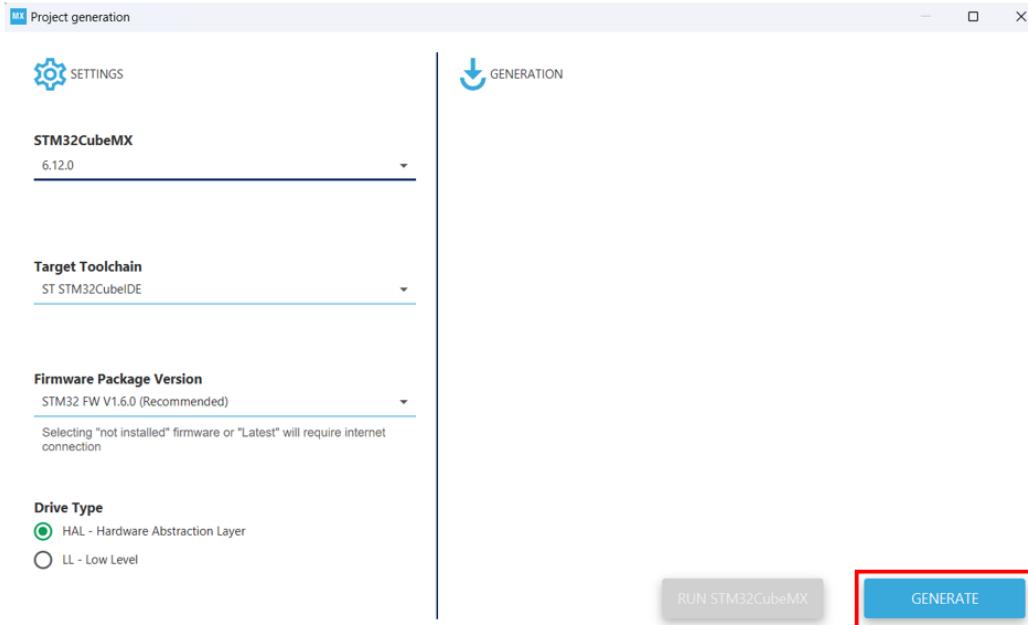
รูปที่ 66 แสดงการตั้งค่า PWM Generation ใน 6-Step Wizard

10. เมื่อทำการตั้งค่าเสร็จเรียบร้อย จึงกด OK และทำการ Generate the project



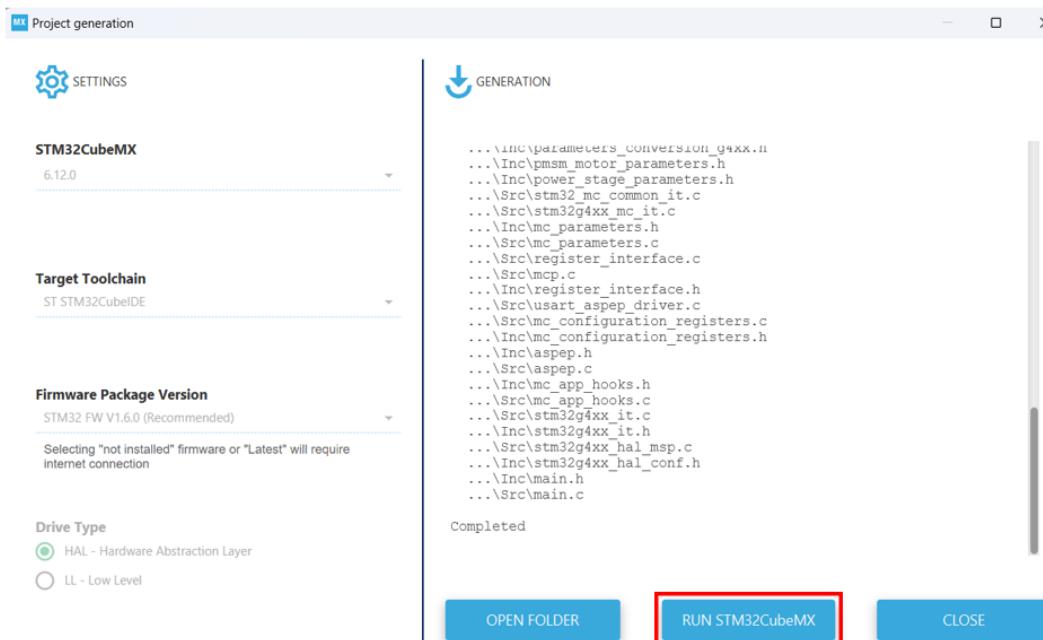
รูปที่ 67 แสดงหน้าตา Project Steps เมื่อทำการตั้งค่าเรียบร้อย

## 11. จากนั้นโปรแกรมจะทำการเปิดหน้าต่าง Project generation ให้กดปุ่ม Generate



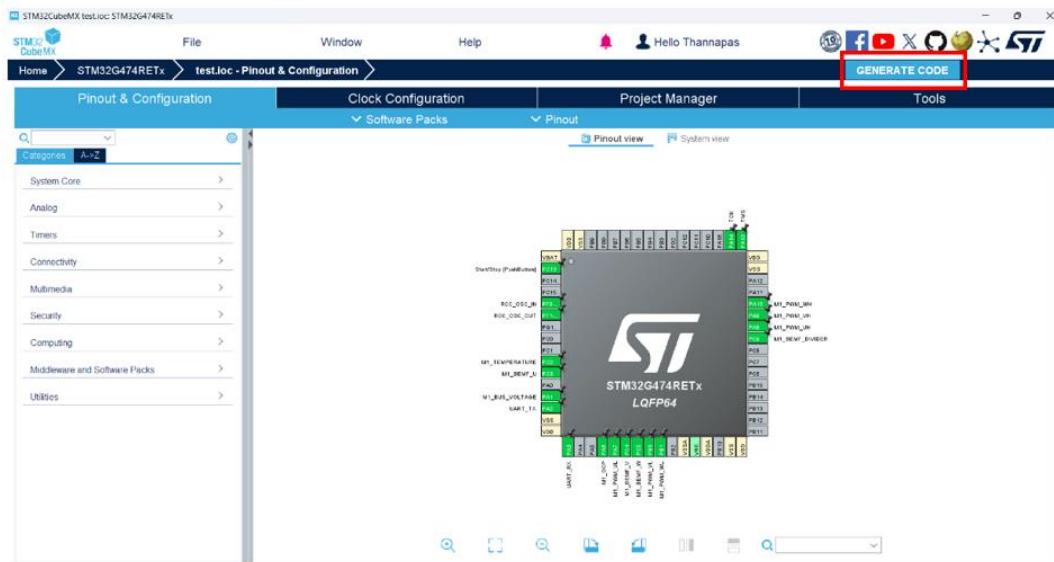
รูปที่ 68 แสดงหน้าตา Project generation

## 12. เมื่อโปรแกรมทำการติดตั้งเสร็จสิ้นจึงกดปุ่ม RUN STM32CubeMX



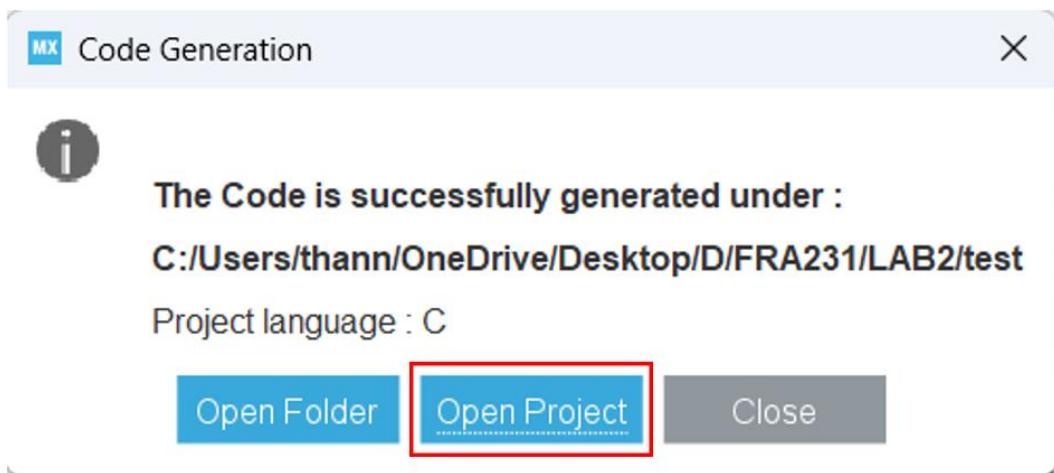
รูปที่ 69 แสดงหน้าต่างตอนติดตั้งข้อมูลใน STM32CubeMX เสร็จสิ้น

13. จากนั้นโปรแกรม STM32CubeMX จะถูกเปิด ให้ทำการกดปุ่ม GENERATE CODE



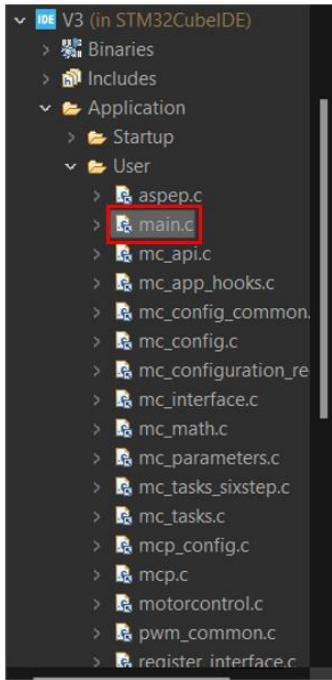
รูปที่ 70 แสดงโปรแกรม STM32CubeMX ที่ถูกตั้งค่าเรียบร้อย

14. จากนั้นจึงกด Open Project



รูปที่ 71 แสดงการกดปุ่ม Open Project

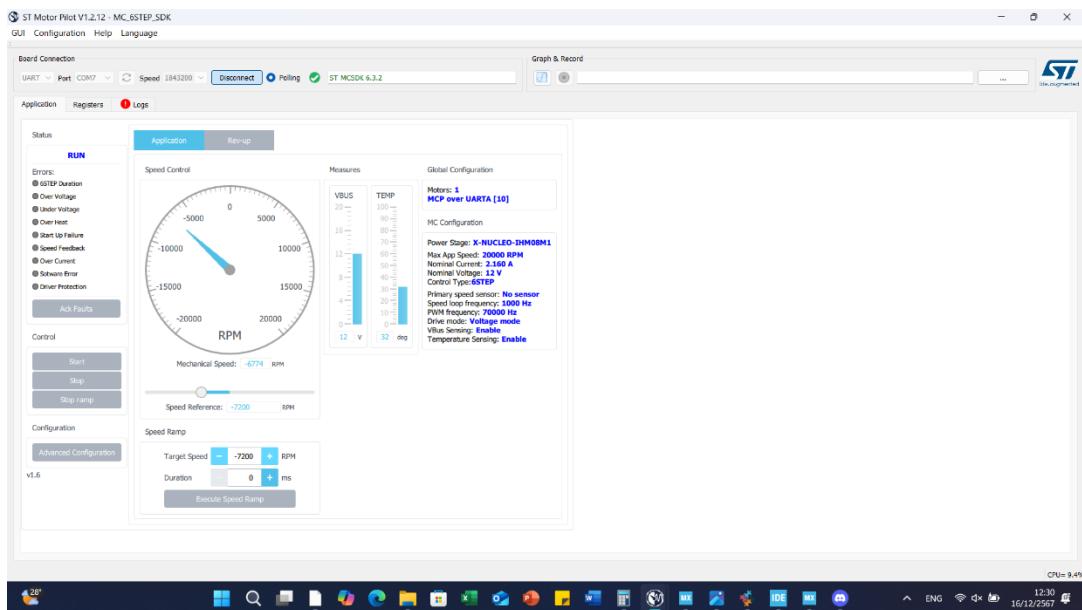
15. โปรแกรมจะทำการเปิด STM32CubeIDE จากนั้นให้ทำการหาไฟล์ main.c จากชื่อไฟล์ที่ตั้งไว้ ดังรูป



รูปที่ 72 แสดงการหาไฟล์ main.c ในโปรแกรม STM32CubeIDE

16. เปิดไฟล์ main.c และทำการเลือก Port ที่เชื่อมบอร์ด STM32G474RE จากนั้นจึงกด RUN

17. กลับเข้าโปรแกรม Motor Control Workbench กดปุ่มเปิด Motor Pilot

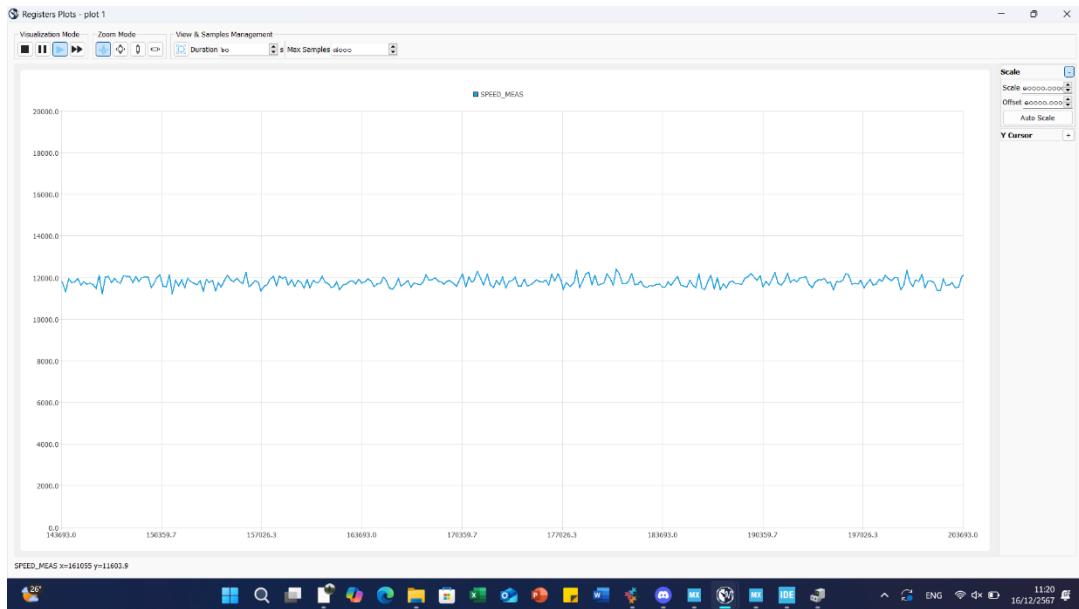


รูปที่ 73 แสดงหน้าตาของ Motor Pilot

18. ทำการตั้งค่า Target Speed เป็นค่าเดียวกันกับ Max Application Speed คือ 20000 RPM เพื่อทำการหาค่า Max Speed ของมอเตอร์ที่แท้จริง เพื่อนำไปใช้ในการทดลองต่อไป
19. กดปุ่ม Start เพื่อให้มอเตอร์ทำงาน
20. ทำการเปิดไปยังหน้า Registers หาชื่อ SPEED\_MEAS และกดปุ่ม Configure เพื่อทำการสร้างกราฟค่าเฉลี่ยความเร็วของมอเตอร์ที่ทำการวัดได้ในขณะนั้น

Poll	Id	Name	Value	Access	Unit	Type	Scope	Plot	Status	Description
	olen	STATUS	IDLE	R	LIBENUM	in	Configure	Int	Motor control subsystem state	
	olen	CONTROL_MODE	OBSERVING	RW	LIBENUM	in	Configure	Int	Control mode	
	olen	RUC_STAGE_NBR	0	None	UB	in	Configure	Int	Number of RevUp stages	
	olen	LOWSIDE_MODULATION	0	RW	UB	in	Configure	Int	Select which steps has a LOWSIDE enabling configuration	
	olen	QUASI_SYNCH	DISABLE	RW	LIBENUM	in	Configure	Int	Quasi-synchronous rectification enabler flag	
	olen	SPEED_KP	0	RW	s16A/ppm	\$16	Configure	Int	Speed PID regulator K_P parameter (Numerator)	
	olen	SPEED_KI	0	RW	s16A	\$16	Configure	Int	Speed PID regulator K_I parameter (Numerator)	
	olen	SPEED_KD	0	RW	s16A/ppm <sup>2</sup>	\$16	Configure	Int	Speed PID regulator K_D parameter (Numerator)	
	olen	BUS_VOLTAGE	0	R	V	U16	Configure	Int	Bus Voltage measurement	
	olen	HEATS_TEMP	0	R	deg	U16	Configure	Int	Heatsink temperature	
	olen	HALL_RL_ANGLE	0	None	s16degree	\$16	in	Int	Measured electrical angle (only available if Hall sensors are used either as main or auxiliary speed and position sensor)	
	olen	HALL_SPEED	0	None	s16speed	\$16	in	Int	Measured mechanical rotation speed (only available if Hall sensors are used either as main or auxiliary speed and position sensor)	
	olen	SPEED_KP_DIV	1	RW	N/A	U16NUM	Configure	Int	Speed PID regulator K_P divisor as power of 2	
	olen	SPEED_KI_DIV	1	RW	N/A	U16NUM	Configure	Int	Speed PID regulator K_I divisor as power of 2	
	olen	SPEED_KD_DIV	1	RW	N/A	U16NUM	Configure	Int	Speed PID regulator K_D divisor as power of 2	
	olen	MC_RHS_PULSE_VALUE	0	R	U16	in	Configure	Int	Pulse value of the HR timer	
	olen	FAULTS_FLAGS	0	R	U16T4AO1	in	Configure	Int	Faults flags	
	olen	SPEED_MEAS	0	R	RPM	\$32	Configure	Int	Average Mechanical Speed measure	
	olen	FW_NAME	0	R	STRING	o	Configure	Int	Hexadecimal representation of the FW name	
	olen	CTRL_STAGE_NAME	0	R	STRING	o	Configure	Int	Name of the control stage	
	olen	POWER_STAGE_NAME	0	R	STRING	o	Configure	Int	Name of the Power stage	
	olen	MOTOR_NAME	0	R	STRING	o	Configure	Int	Motor name	

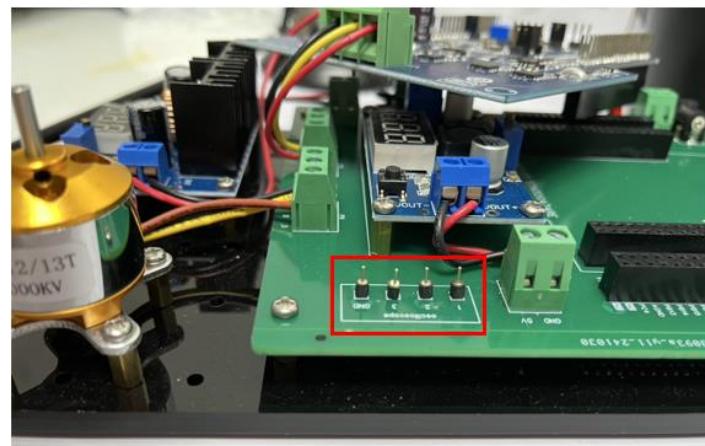
รูปที่ 74 แสดงการกดเลือกค่า SPEED\_MEAS เพื่อสร้างกราฟความเร็วขณะนั้น



รูปที่ 75 แสดงกราฟระหว่างเวลา กับ ความเร็วขณะนั้น

จากการภาพว่าเวลา กับความเร็วขณะนั้น สามารถเห็นได้ว่าถึงแม้จะตั้งค่า Target Speed 20000 RPM แต่ว่า เท่าจริงแล้วมอเตอร์นั้นมีความเร็วสูงสุดประมาณ 12000 RPM ทำให้คณะผู้จัดทำออกแบบการทดลองวัดค่า Brushless DC Motor ด้วย Oscilloscope จากความเร็วตั้งแต่ 0 – 12000 RPM โดยเพิ่มขึ้นครั้งละ 10 % หรือ 1200 RPM ทั้งหมดตามเข็มนาฬิกา และวนเข็มนาฬิกา

21. ต่อ Oscilloscope กับบอร์ดการทดลอง คือ GND U V W ตามลำดับ



รูปที่ 76 แสดงการต่อ Oscilloscope กับบอร์ดการทดลอง

22. เก็บค่า Frequency จากกราฟที่วัดได้จาก Oscilloscope จำนวน 3 ค่า โดยการวัดจะวัดการทำงานให้ครบ 6-Step Control เพื่อนำมาคำนวณหาความเร็วข้อนกลับ ค่าเฉลี่ย และเปอร์เซ็นต์ความผิดพลาด

22.1 การหาความเร็วข้อนกลับ (RPM) สามารถหาได้จากสมการ ดังนี้

$$RPM = \frac{f \times 60}{\text{number of pole pairs}}$$

22.2 การหาค่าเฉลี่ย (Average) สามารถหาได้จากสมการ ดังนี้

$$\text{Average} = \frac{RPM_1 + RPM_2 + RPM_3}{3}$$

22.3 การหาเปอร์เซ็นต์ความผิดพลาด (%Error) สามารถหาได้จากสมการ ดังนี้

$$\%Error = \left| \frac{\text{ค่าเฉลี่ย} - \text{ความเร็วที่ตั้งค่าไว้}}{\text{ความเร็วที่ตั้งค่าไว้}} \right| \times 100$$



รูปที่ 77 แสดงการทำงานแบบ 6-Step และค่า Frequency ที่หาได้จากการวัดด้วย Oscilloscope

23. ทำข้ามข้อ 22 โดยเพิ่มความเร็วที่ลิส 1200 RPM จนครบ 10 ค่า ได้แก่ 1200, 2400, 3600, 4800, 6000, 7200, 8400, 9600, 10800, 12000 ทำทั้งค่าบวก และลบ ซึ่งคือการที่มอเตอร์หมุนตามเข็มนาฬิกา และทวนเข็มนาฬิกา
24. นำข้อมูลการวัดสัญญาณ 3 Phase มาคำนวณหา Phase Shift ด้วยสมการ ดังนี้

$$\text{Phase Shift} = \frac{\Delta T}{T} \times 360^\circ$$

เมื่อ

$\Delta T$  = เวลาทั้งหมดจากปลายลูกคลื่นแรกถึงจีดเริ่มต้นลูกคลื่นถัดไป (ms)

$T$  = เวลาทั้งหมดจากปลายลูกคลื่นแรกถึงปลายลูกคลื่นถัดไป (ms)

จากรูปภาพที่ สามารถคำนวณหา Phase Shift ได้จากการวัดเวลาตัวแปร  $T$  คือ 2.390 ms แต่ว่าทางคณะผู้จัดทำไม่ได้ทำการวัดหา  $\Delta T$  ทำให้ต้องทำการคำนวณหาจาก  $T$  ที่วัดมาเท่านั้น โดยเริ่มจากการนับช่องที่เป็นจุดเล็ก ๆ ในหน้าจอ Oscilloscope จากปลายลูกคลื่นแรกถึงปลายลูกคลื่นถัดไปใน Phase U (สัญญาณสีเหลือง) นับได้ทั้งหมด 24 ช่อง จากนั้น จึงทำการคำนวณเวลาต่อหนึ่งช่อง ถึงค่อยทำการคูณเข้ากับจำนวนช่องของปลายลูกคลื่นแรกถึงต้นลูกคลื่นถัดไป คือ 8 ช่อง ซึ่งจะมีการคำนวณหาได้ ดังนี้

$$\begin{aligned}\Delta T &= \frac{T}{24} \times 8 \\ &= \frac{2.39 \text{ ms}}{24} \times 8 \\ &= 0.80 \text{ ms}\end{aligned}$$

จากนั้นจึงนำค่าที่ได้มาคำนวณหา Phase Shift ดังนี้

$$\text{Phase Shift} = \frac{\Delta T}{T} \times 360^\circ$$

$$= \frac{0.80 \text{ ms}}{2.39 \text{ ms}} \times 360^\circ$$

$$= 120^\circ$$

สามารถแสดงให้เห็นได้ว่าที่ Phase Shift มีค่า 120 องศา นั้น แสดงถึงเซนเซอร์ภายในสเตเตอร์ของ Brushless Motor ถูกวางไว้ห่างกัน 120 องศา จำนวน 3 ตัว



รูปที่ 78 แสดงการหาตัวแปรเพื่อนำมาคำนวณหา Phase Shift จากการทำงานแบบ 6-Step ที่ความเร็ว 3600 RPM

## ผลการทดสอบ

Speed	Frequency	RPM	Average	%Error	Speed	Frequency	RPM	Average	%Error
1200	139.7	1197.43	1197.43	0.21	7200	841.8	7215.43	7167.14	0.46
	139.7	1197.43				830.6	7119.43		
	139.7	1197.43				836.1	7166.57		
2400	281.7	2414.57	2407.71	0.32	8400	936.3	8025.43	8159.43	2.86
	280.9	2407.71				954.2	8178.86		
	280.1	2400.86				965.3	8274.00		
3600	418.4	3586.29	3596.29	0.10	9600	996	8537.14	8560.29	10.83
	421.9	3616.29				1008	8640.00		
	418.4	3586.29				992.1	8503.71		
4800	555.6	4762.29	4798.00	0.04	10800	1235	10585.71	10620.00	1.67
	565	4842.86				1232	10560.00		
	558.7	4788.86				1250	10714.29		
6000	704.2	6036.00	6002.57	0.04	12000	1374	11777.14	11785.71	1.79
	692.5	5935.71				1374	11777.14		
	704.2	6036.00				1377	11802.86		

ตารางที่ 89 แสดงการคำนวณหาความเร็วข้อมูล ค่าเฉลี่ย และเบอร์เซ็นต์ความผิดพลาด เมื่อมอเตอร์หมุนฟื้งบาก

Speed	Frequency	RPM	Average	%Error	Speed	Frequency	RPM	Average	%Error
-1200	143.7	1231.71	1227.14	2.26	-7200	822.4	7049.1429	7056.8571	1.9880952
	142.9	1224.86				822.4	7049.1429		
	142.9	1224.86				825.1	7072.2857		
-2400	275.5	2361.43	2365.71	1.43	-8400	950.6	8148	8171.4286	2.7210884
	277.8	2381.14				976.6	8370.8571		
	274.7	2354.57				932.8	7995.4286		
-3600	411.5	3527.14	3532.00	1.89	-9600	1092.0	9360	9291.4286	3.2142857
	411.5	3527.14				1068.0	9154.2857		
	413.2	3541.71				1092.0	9360		
-4800	560.5	4804.29	4758.57	0.86	-10800	1263.0	10825.714	10480	2.962963
	554.3	4751.14				1220.0	10457.143		
	550.7	4720.29				1185.0	10157.143		
-6000	679.3	5822.57	5808.00	3.20	-12000	1408.0	12068.571	11882.857	0.9761905
	686.8	5886.86				1385.0	11871.429		
	666.7	5714.57				1366.0	11708.571		

ตารางที่ 910 แสดงการคำนวณหาความเร็วข้อมูล ค่าเฉลี่ย และเบอร์เซ็นต์ความผิดพลาด เมื่อมอเตอร์หมุนผ่านลับ

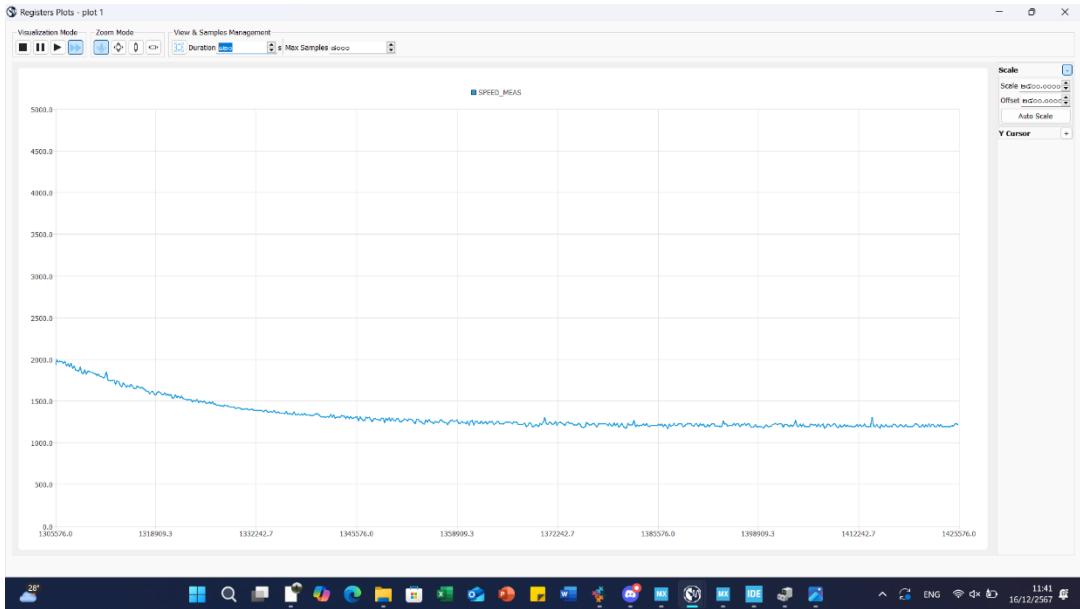
## สรุปผล

จากการที่ 9 สามารถเห็นได้ว่าการตั้งค่าความเร็วใน Motor Pilot ตั้งแต่ 1200 – 12000 RPM ในช่วงผิ่งบวก โดยเพิ่มความเร็วครั้งละ 1200 RPM เมื่อนำมาคำนวณหาความเร็วจากการหา Frequency เมื่อมอเตอร์ทำงานครบ 6-Step จะเห็นได้ว่าค่า้นนี้มีความใกล้เคียงกันจริง ตัวอย่างเช่น ค่าความเร็วที่ 1200 RPM สามารถคำนวณหาค่าเฉลี่ยความเร็วย้อนกลับได้ที่ 1197.43 RPM ซึ่งมีเปอร์เซ็นต์ความผิดพลาดอยู่ที่ร้อยละ 0.21 ซึ่งจากข้อมูลภายในกราฟส่วนใหญ่ ค่าเฉลี่ยความเร็วย้อนกลับที่ใกล้เคียงกัน และเปอร์เซ็นต์ความผิดพลาดไม่เกินร้อยละ 0.5 แต่จะมีค่าความเร็ว 8400 9600 10800 และ 12000 RPM มีค่าเฉลี่ยความเร็วย้อนกลับอยู่ที่ 8159.43, 8560.29, 10620.00 และ 11785.71 RPM ตามลำดับ มีเปอร์เซ็นต์ความผิดพลาดร้อยละ 2.86, 10.83, 1.67 และ 1.79 ซึ่งค่อนข้างเห็นได้ว่ามีความใกล้เคียงน้อยลง อย่างเห็นได้ชัดตั้งแต่ช่วงความเร็วมากกว่าหรือเท่ากับ 8400 RPM เป็นต้นไป

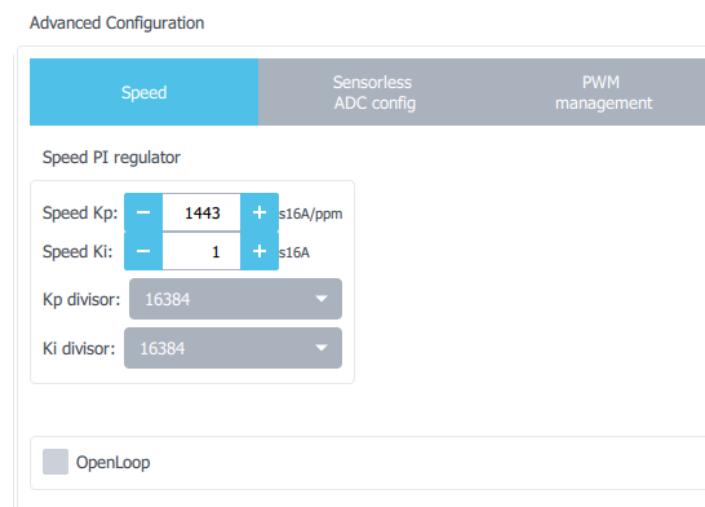
จากการที่ 10 สามารถเห็นได้ว่าการตั้งค่าความเร็วใน Motor Pilot ตั้งแต่ 1200 – 12000 RPM ในช่วงผิ่งลบ โดยเพิ่มความเร็วครั้งละ 1200 RPM ค่าความเร็วย้อนกลับที่คำนวณได้มีความใกล้เคียงกัน แต่มีเปอร์เซ็นต์ความผิดพลาดที่มากกว่าในช่วงผิ่งบวก ซึ่งเปอร์เซ็นต์ความผิดพลาดนั้นมีความใกล้เคียงกันอยู่ที่ประมาณร้อยละ 1- 3 โดยสามารถนำมาทำค่าเฉลี่ยเปอร์เซ็นต์ความผิดพลาดโดยรวมคือ ร้อยละ 2.15

## อภิรายผล

สาเหตุที่การคำนวณความเร็วย้อนกลับจาก 6-Step ด้วยค่า Frequency ที่อ่านได้จากการวัดสัญญาณ 3 Phase จาก Oscilloscope นั้นมีความคลาดเคลื่อนมาจากความผิดพลาดทางมนุษย์ตอนที่ทำการใช้ Cursor เลือกช่วง 6-Step ผ่านทางหน้าจอของ Oscilloscope ทำให้ค่า Frequency ที่ได้มาราจะคลาดเคลื่อนได้ นอกจากนี้ยังมีความผิดพลาดทางอุปกรณ์ ขณะที่สั่งงานมอเตอร์ให้หมุนไปที่ความเร็วหนึ่ง ค่าความเร็วที่ได้มามาไม่ได้เป็นค่าความเร็วนั้นจริง ๆ ซึ่งสามารถพิสูจน์ได้จากการเปิดกราฟ SPEED\_MEAS ดังรูปที่  จะเห็นได้ว่าเมื่อสั่งงานมอเตอร์ที่ความเร็ว 1200 RPM กราฟที่แสดงความเร็วอยู่ที่ประมาณ 1200 RPM นั้น เส้นกราฟไม่ได้อยู่เป็นเส้นตรง แต่กราฟมีความกระเพื่อม จากรูปที่  เห็นได้ว่า โปรแกรม Motor Pilot ส่วน Advanced Configuration มีการให้กำหนดค่า Speed Kp และ Speed Ki เพื่อเป็นการทำ PID ให้ค่า Frequency ที่วัดออกมานั้นไม่คงที่



รูปที่ 79 แสดงกราฟระหว่างเวลาและ SPEED\_MEAS เมื่อกำหนด Target Speed 1200 RPM



รูปที่ 80 แสดงการกำหนดค่า Speed Kp และ Speed Ki

การวิเคราะห์สัญญาณ 3 Phase ที่ได้จาก Oscilloscope หากสังเกตที่สัญญาณรูปทรง Trapezoidal จากรูปที่ X ด้านซ้าย เมื่อทำการขยายเข้าไปดูช่วงเปิด MOSFET ด้วยสัญญาณ PWM จะเห็นเป็นดังรูปที่ X ด้านขวา แสดงให้เห็นว่า Phase V สามารถคำนวณ Duty Cycle ได้จากสมการ ดังนี้

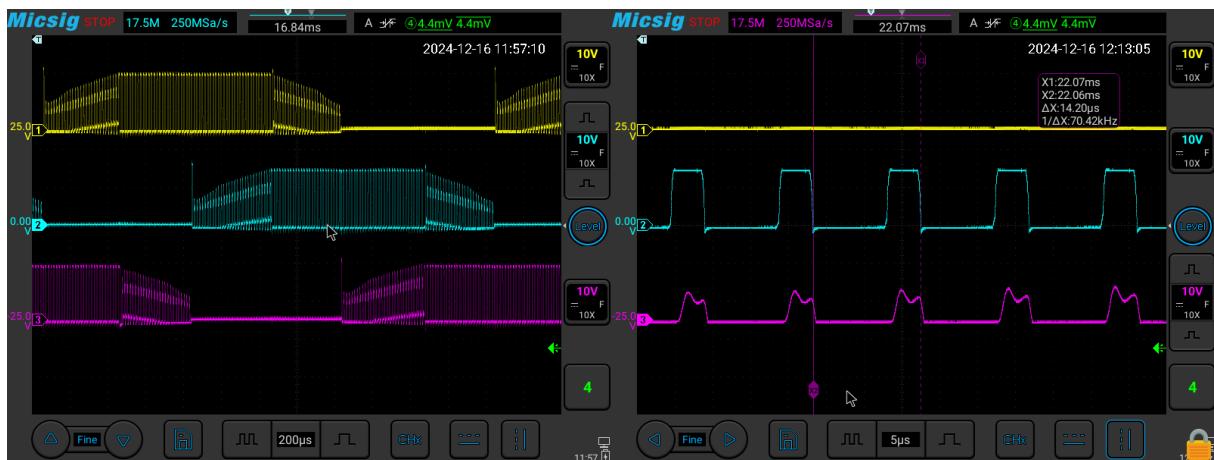
$$\text{Duty Cycle} = \frac{T_{on}}{T_{on} + T_{off}} \times 100$$

เมื่อ

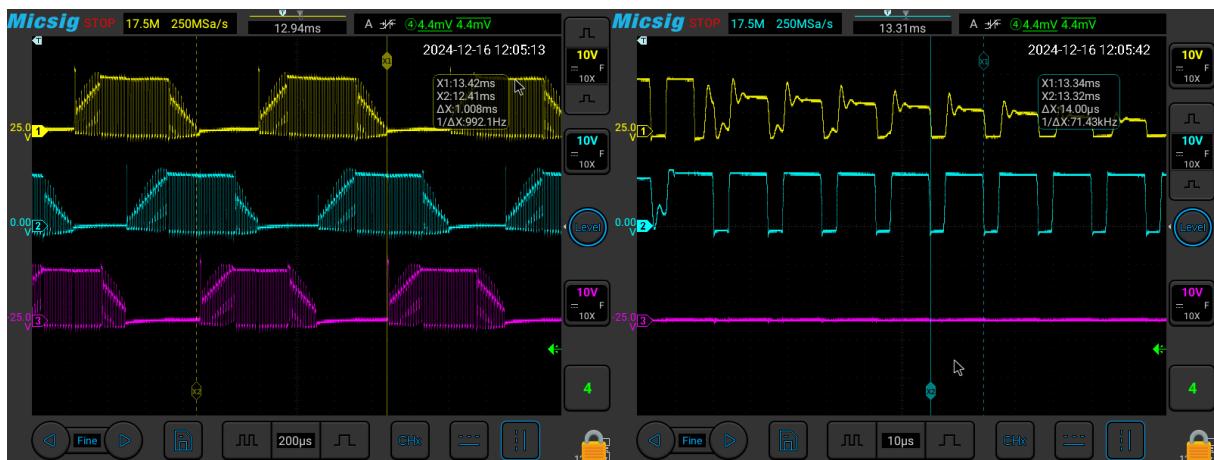
$T_{on}$  = ช่วงเปิดสัญญาณ (High)

$T_{off}$  = ช่วงปิดสัญญาณ (Low)

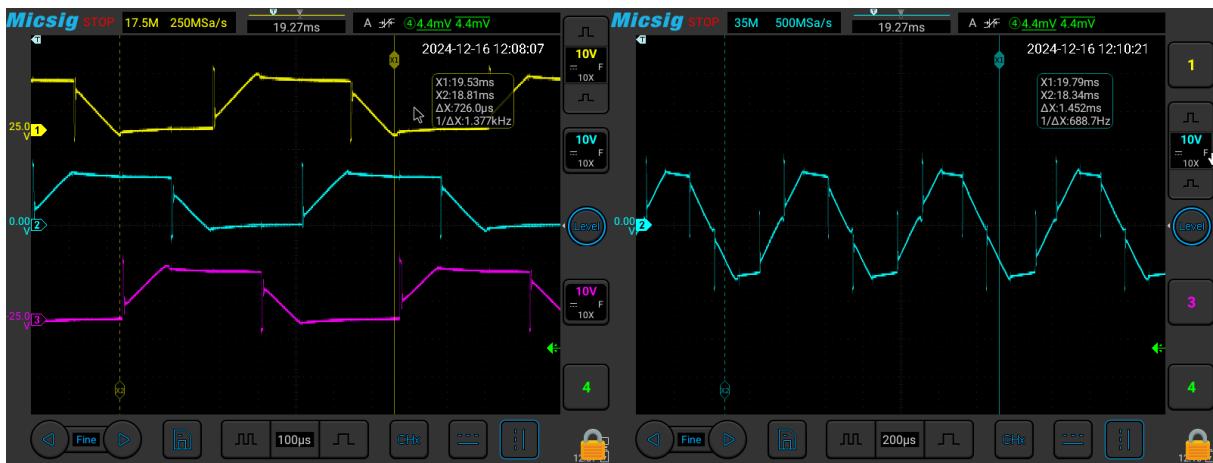
แสดงให้เห็นว่าที่ความเร็ว 3600 RPM มี Duty Cycle ร้อยละ 30.99 ที่ความเร็ว 9600 RPM มี Duty Cycle ร้อยละ 71.43 ที่ความเร็ว 12000 RPM มี Duty Cycle ร้อยละ 100 สามารถวิเคราะห์ได้ว่าเมื่อมีการปรับความเร็วเพิ่มขึ้น ค่า %Duty Cycle จะเพิ่มขึ้นด้วย แสดงให้เห็นว่าความเร็วนั้นแปรผันตรงกับ Duty Cycle



รูปที่ 81 แสดงสัญญาณ Trapezoidal จาก Phase V ที่ความเร็ว 3600 RPM จากการอ่านที่ 200 และ 5 Microsec

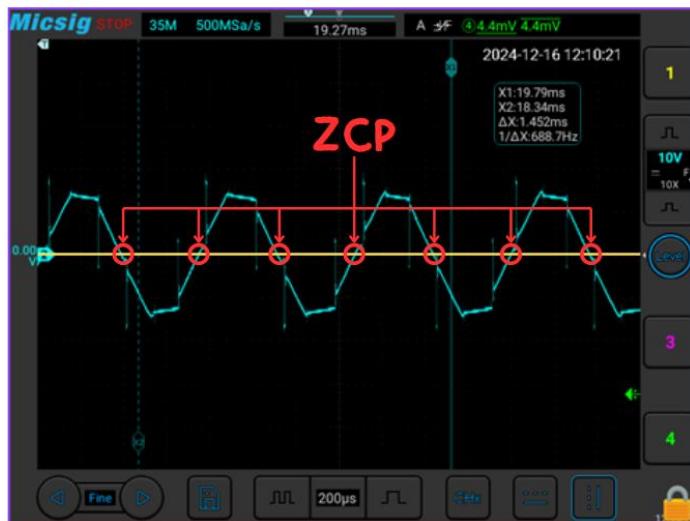


รูปที่ 82 แสดงสัญญาณ Trapezoidal จาก Phase V ที่ความเร็ว 9600 RPM จากการอ่านที่ 200 และ 5 Microsec



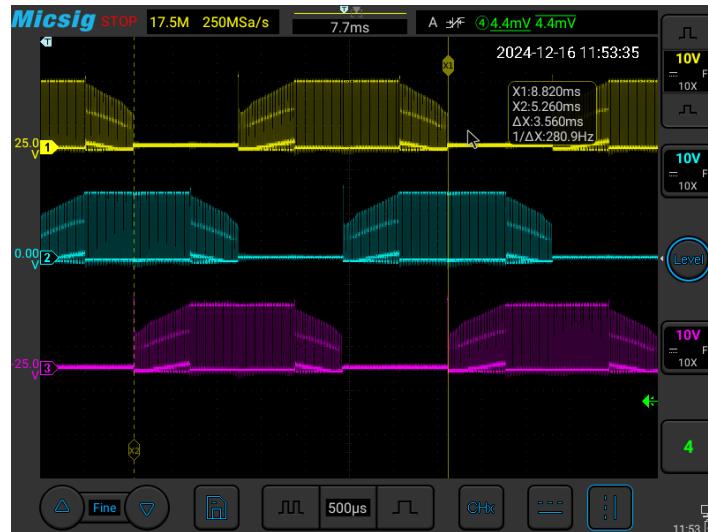
รูปที่ 83 แสดงสัญญาณ Trapezoidal จาก Phase V ที่ความเร็ว 12000 RPM จากการอ่านที่ 200 และ 5 Microsec

การวิเคราะห์จุด Zero Crossing (ZCP) จากรูปที่ สามารถสังเกตได้จากจุดที่เกิดขึ้นข้างบนหรือข้างล่าง เส้น 0.00 V ซึ่งคือ Ground Truth จากการวัด Oscilloscope เทียบ Phase U และ V ด้วยกัน สามารถหาจุด ZCP ได้ โดยการลากเส้นสมมติ Ground Truth และหากจุดไหนตัดกับเส้นนี้ จะถือว่าเป็นจุด ZCP เนื่องจากแรงดันไฟฟ้ามีพุ่ติกรรรมเคลื่อนที่ผ่านมากกว่าหรือน้อยกว่า Ground Truth หรือ 0 V



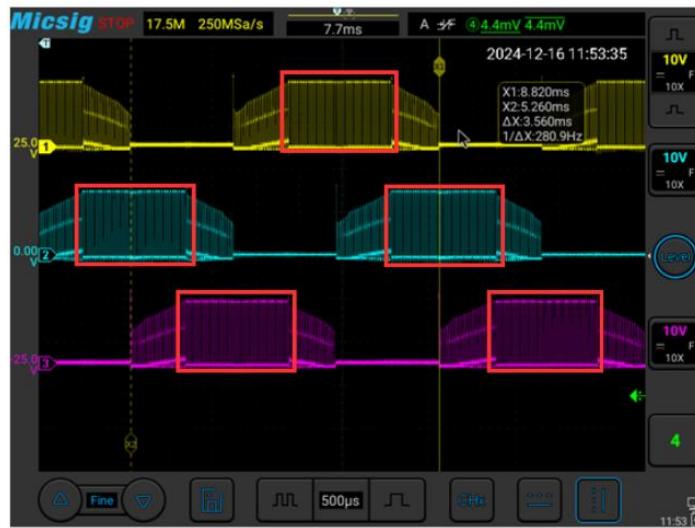
รูปที่ 84 แสดงจุด Zero Crossing Point จากสัญญาณขาออกที่เทียบระหว่าง Phase U และ V ณ ความเร็ว 12000 RPM  
จากการอ่านที่ 200 Microsec

การวิเคราะห์ Back-EMF จากรูปที่ 85 ที่เกิดขึ้นจากการวัดสัญญาณด้วย Oscilloscope ใน Phase V จะไม่สามารถเห็นได้ถึง Back-EMF ที่แท้จริง ดังที่กล่าวไว้ในเอกสารและงานวิจัยที่เกี่ยวข้อง ทำให้จากการวัดสัญญาณของมาสามารถแสดงให้เห็นช่วงสัญญาณของ Back-EMF ที่ถูกซ่อนทับเข้ากับ PWM



รูปที่ 85 แสดงสัญญาณ Trapezoidal ที่ความเร็ว 2400 RPM จากการอ่านที่ 500 Microsec

จากรูปที่ 86 กรอบสีเหลืองสีแดงนั้นแสดงถึงช่วงที่ Permanent Magnet กำลังเคลื่อนที่เข้าใกล้ชุดลดลาดสเตเตอร์ แต่ว่าสัญญาณนั้นถูกซ่อนทับเข้ากับ PWM ของ Brushless DC Motor ที่อยู่ในช่วงทำงาน (On Time)



รูปที่ 86 แสดงสัญญาณ Trapezoidal ช่วงที่ Permanent Magnet อยู่ต่างกับชุดลด

การวิเคราะห์ Field oriented control (FOC) เป็นเทคนิคการควบคุมมอเตอร์ไฟฟ้ากระแสสลับ ที่ช่วยปรับปรุงประสิทธิภาพและความแม่นยำในการควบคุมแรงบิดและความเร็วของมอเตอร์ เทคนิคนี้มักใช้ในมอเตอร์ซิงโครนัสแม่เหล็กถาวร (PMSM) และมอเตอร์เห็นี่ยวน้ำ ซึ่งใช้การแปลงสัญญาณทางคณิตศาสตร์เพื่อแยกแยะระหว่างสเตเตอเรอร์ (Stator) และโรเตอเรอร์ (Rotor) โดยมีขั้นตอนสำคัญคือการแปลงจากเฟสสามเป็นสอง (Clarke Transformation) การแปลงจากระนาบสแตติกเป็นระนาบทมุน (Park Transformation) การควบคุมในระนาบทมุน และการแปลงกลับเพื่อขับมอเตอร์ ข้อดีของ FOC คือสามารถควบคุมแรงบิดได้แม่นยำ มีความเสถียรสูง และเพิ่มประสิทธิภาพการทำงานของมอเตอร์ จึงมีการนำไปใช้ในหลายอุตสาหกรรม เช่น ยานยนต์ เครื่องจักรในภาคการผลิต และอุปกรณ์เครื่องใช้ไฟฟ้า ในการวิเคราะห์เชิงลึก FOC ใช้ Clarke Transformation และ Park Transformation เพื่อแปลงสัญญาณจากระบบสามเฟสไปยังระบบสองเฟสที่อยู่ในระนาบทมุน ทำให้สามารถควบคุมแรงบิดและฟลักช์แม่เหล็กได้อย่างมีประสิทธิภาพ Field Oriented Control จึงเป็นเทคนิคที่มีบทบาทสำคัญในการพัฒนาระบบควบคุมมอเตอร์ในปัจจุบัน ช่วยเพิ่มประสิทธิภาพการทำงาน ลดการสูญเสียพลังงาน และเพิ่มความเสถียรในการควบคุม

### ข้อเสนอแนะ

1. ควรตรวจสอบการต่อ Scope ของ Oscilloscope เข้ากับบอร์ดการทดลองให้ดี เพื่อให้ค่าที่วัดออกมากมีความแม่นยำมากที่สุด
2. ควรระวังบอร์ดการทดลองเคลื่อนที่เมื่อทำการสั่งงาน Brushless DC Motor ที่ความเร็วสูงมาก
3. หากต้องการคำนวณหาค่าความเร็วอันกลับให้แม่นยำ และมีความคลาดเคลื่อนน้อยลงต้องทำการหาค่าเฉลี่ยของความเร็วที่ได้จากโปรแกรม Motor Pilot เมื่อระบบกำลังทำการสั่งการ Brushless DC Motor จากการทดลองจะเห็นได้ว่ากราฟความเร็วที่ระบบสั่งงานจริงนั้น ไม่ได้มีความคงที่เป็นเส้นตรง แต่กราฟมีความกรวยเพื่อมอยู่ระหว่างค่าความเร็วที่ได้สั่งการไป เพราะว่ามอเตอร์ไม่ได้ทำงานที่ความเร็ว robust ที่
4. ควรตรวจสอบการเลือกช่วงคลื่นที่ต้องการจากสัญญาณ 3 Phase ที่วัดด้วย Oscilloscope ให้ดี เนื่องจากว่าค่า Frequency ที่ได้ออกมาอาจคลาดเคลื่อนได้หากไม่เช็คให้รอบคอบก่อน
5. ควรศึกษา Datasheet ของ Brushless DC Motor นั้น ๆ เพื่อไม่ให้เกิดความผิดพลาดในการจ่ายกระแสไฟฟ้าหรือแรงดันไฟฟ้า

## เอกสารอ้างอิง(แนบ link)

- [https://www.rhydolabz.com/documents/26/BLDC\\_A2212\\_13T.pdf?srsltid=AfmBOoqrQd9KG\\_VI4lbE0qNkTbdceNZ909Ragq8nb4Kp1HSBBq5VuzgK](https://www.rhydolabz.com/documents/26/BLDC_A2212_13T.pdf?srsltid=AfmBOoqrQd9KG_VI4lbE0qNkTbdceNZ909Ragq8nb4Kp1HSBBq5VuzgK)
- <https://th.huahaomotors.com/info/what-is-a-bldc-motor--90418688.html>
- [https://www.9engineer.com/index.php?m=article&a=print&article\\_id=2606](https://www.9engineer.com/index.php?m=article&a=print&article_id=2606)
- <https://greatpcb.com/th/foc-field-oriented-control-explained-for-beginners/>
- <https://bacancysystems.com/blog/trapezoidal-and-sinusoidal-bldc-motors>
- <https://www.celeramotion.com/frameless-motors/trapezoidal-vs-sinusoidal-commutation/>
- <https://mechtex.com/blog/back-emf-in-bldc-motors-a-complete-guide>
- [https://toshiba.semicon-storage.com/info/application\\_note\\_en\\_20180803\\_AKX00303.pdf?did=61176](https://toshiba.semicon-storage.com/info/application_note_en_20180803_AKX00303.pdf?did=61176)