

LEB 2: Actuators

Name

- นายสิรภพ วาทินชัย 67340500045
- นางสาวพัทธนันท์ รำพึงพงษ์ 67340500058
- นายพีรดนัย โพธิ์ศรีเมือง 67340500059
- นายศิววัฒน์ พิงส์ใส 67340500065

Objectives

- เพื่อออกแบบการทดลอง และตรวจสอบพฤติกรรมการทำงานของ Actuators (Dc Motor, Stepper Motor, Brushless DC Motor (BLDC)
- เพื่อให้เลือกใช้มอเตอร์ให้เหมาะสมกับการใช้งาน
- เพื่อให้เข้าใจหลักการทำงานของมอเตอร์

LAB 2.1: DC Motor

การทดลองที่ 1: การศึกษาความสัมพันธ์ระหว่าง Torque กับค่าต่างๆ

จุดประสงค์

เพื่อศึกษาพฤติกรรมของ motor และ Torque ที่เปลี่ยนไป

สมมติฐาน

Current จะเปลี่ยนแปลงตาม Torque ที่มากขึ้น

ตัวแปร

1. ตัวแปรต้น:
 - Torque
2. ตัวแปรตาม:
 - Current
3. ตัวแปรควบคุม:
 - ชุดบอร์ดการทดลอง

เอกสารและงานวิจัยที่เกี่ยวข้อง

1. หลักการทำงานของ DC Motor

DC Motor ทำงานโดยใช้กระแสไฟฟ้าไหลผ่านขดลวดเพื่อสร้างสนามแม่เหล็ก ซึ่งทำให้โรเตอร์เกิดแรงบิดและหมุนออกมาเป็นพลังงานกล ขณะที่หมุนจะเกิด Back-EMF ซึ่งแปรผันตรงกับความเร็รรอบและเป็นปัจจัยสำคัญในการควบคุม สมการพื้นฐาน เช่น $Torque = k_t I$ และ $Back - EMF = k_e \omega$ ใช้ในการอธิบายพฤติกรรมของมอเตอร์และช่วยให้เข้าใจการทำงานจริงได้ดีขึ้น

2. สมการ DC Motor ที่ใช้สร้าง Motor Characteristic Curve

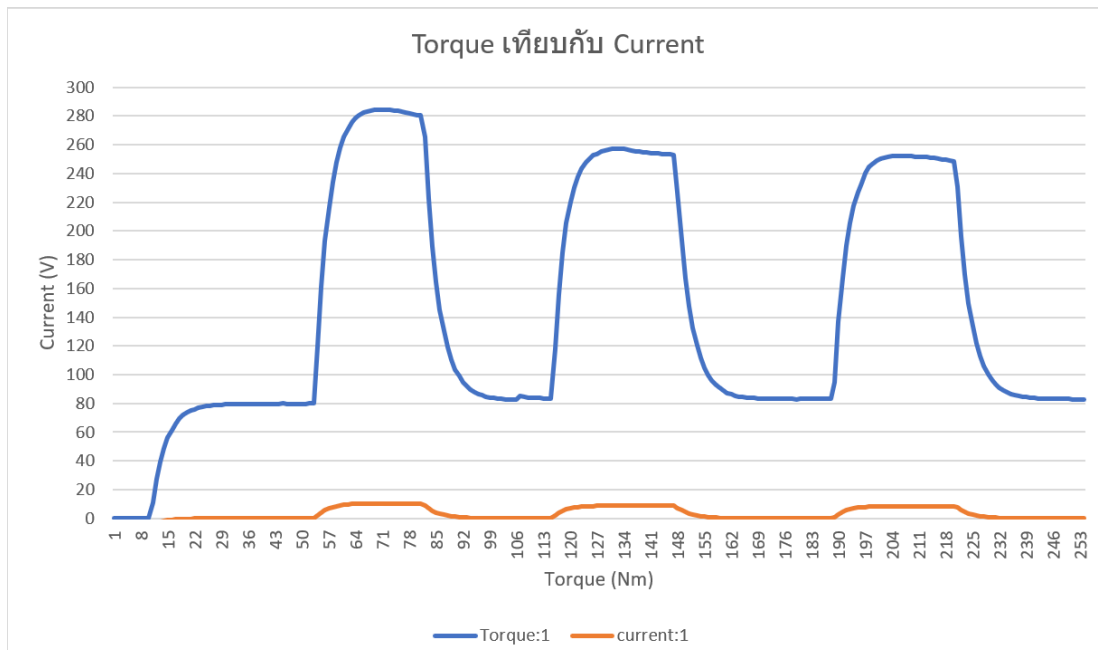
กราฟลักษณะของมอเตอร์ เช่น Speed-Torque และ Current-Torque สร้างขึ้นจากความสัมพันธ์ของสมการระหว่างแรงบิด ความเร็ว และกระแส เช่น ความเร็วที่ลดลงเมื่อแรงบิดเพิ่มขึ้น ($\omega = \omega_0 - (\omega_0/T_{stall}) \cdot T$) หรือกระแสที่เพิ่มตามแรงบิด ($I = I_0 + T/k_t$) รวมถึงการคำนวณกำลังและประสิทธิภาพ ซึ่งช่วยให้สามารถวิเคราะห์สมรรถนะของมอเตอร์ได้อย่างชัดเจน

ขั้นตอนการดำเนินงาน

1. ต่อชุดทดลองเข้ากับแหล่งจ่ายไฟ และคอมพิวเตอร์
2. Setup block ใน Simulink เพื่อใช้ในการอ่านและแปลงค่าต่างๆ
3. ทดลองปรับจูนค่าเพื่อให้ Load cell สามารถอ่านค่าออกมาเป็น KG ได้โดยการนำถ่วงทรายมาวางที่ละถ่วงเพื่อคำนวณค่า offset และ gain ที่ต้องใช้

- ทดลองปรับจูนค่าเพื่อให้ WCS1600 Hall Current Sensor สามารถอ่านค่า Current และ amp ที่ใกล้เคียงความเป็นจริงได้โดยการ ใช้ power supply ต่อขั้วบวกที่ supply ด้านนี้ และเอาคั้วลบต่อที่อีกด้านนี้และอ่านค่าที่ได้จาก sensor
- นำค่าที่ได้มาจาก sensor กับค่าจริงที่อ่านได้มาทำ Linearization เพื่อปรับจูนให้ได้ค่าที่ใกล้เคียงกับของจริงที่สุด
- สั่งให้ motor ทำงานแบบ max duty cycle เพื่อให้ 3D part ที่ได้มากดลงไปที่ load cell เพื่ออ่านค่าและนำไปคำนวณหา Torque และเก็บผลต่างๆ

ผลการทดลอง



รูปภาพ 1 กราฟแสดงการเปรียบเทียบระหว่าง Torque และ Current

สรุปผลการทดลอง

จากการทดลองทำให้เราทราบว่า จะหา K_t ได้จากการเอา $\frac{Torque_{Real}}{I(Amp)}$ และจะหา $Torque_{Real}$ ได้จากการคำนวณดังนี้ $Torque_{Real} = Wight * 9.81 * \text{ระยะแขนที่กด}$ จากการทดลองทำให้ทราบได้ว่ายิ่ง Motor ส่งออก Torque ที่มากขึ้นค่า I ก็จะเปลี่ยนไปตาม Torque ที่เปลี่ยนไปและเมื่อเปลี่ยนค่า Duty cycle จะทำให้ Torque เปลี่ยนไป เนื่องจากกระแสที่จ่ายเข้า motor ต่างไปจากเดิม เมื่อนำค่า K_t ที่ได้มาจะสามารถนำมาใช้เพื่อหาค่า Efficiency ได้และสามารถสร้าง Motor Characteristic ได้

อภิปรายผล

จากการวิเคราะห์การทดลองพบว่า WCS1600 Hall Current Sensor หรือ sensor อ่านค่าไฟฟ้าจากคลื่นแม่เหล็กมีความคลาดเคลื่อนมากถ้าไม่ทำ Signal Conditioning ทางเราจะได้ใช้การ ทำสมาการ

Linearization เพื่อให้ได้ค่าที่เข้าใกล้ค่าความเป็นจริงมากที่สุด แต่เนื่องจาก WCS1600 Hall Current Sensor เป็น sensor ที่ sensitive มากทำให้ค่าต่างๆที่อ่านออกมาได้เกิดการคลาดเคลื่อนได้ และค่า torque ที่ได้มาต่างไปจากความเป็นจริง

ข้อเสนอแนะ

อ้างอิง

1. Hughes, A. (2006). Electric Motors and Drives.
2. Maxon Motor (2020). DC Motor Characteristics.

การทดลองที่ 2: การศึกษาความสัมพันธ์ระหว่าง Duty cycle ที่เปลี่ยนไป

จุดประสงค์

เพื่อศึกษาพฤติกรรมของ motor เมื่อเปลี่ยนไป Duty cycle

สมมติฐาน

Motor จะเปลี่ยนแปลงความเร็วตาม Duty ที่เปลี่ยนไป

ตัวแปร

1. ตัวแปรต้น:
 - Frequency
2. ตัวแปรตาม:
 - Duty cycle
3. ตัวแปรควบคุม:
 - Current ชุดบอร์ดการทดลอง

เอกสารและงานวิจัยที่เกี่ยวข้อง

1. ค่าที่ทำให้มอเตอร์มีประสิทธิภาพสูงสุดที่ 12V

จากทฤษฎีมอเตอร์ DC จะทำงานได้มีประสิทธิภาพสูงสุด เมื่อการสูญเสียพลังงานในรูปความร้อนต่ำที่สุดหรือก็คือเมื่อทำงานที่ ความเร็วประมาณ 70–80% ของความเร็วขณะไม่มีโหลด และอยู่ในช่วงแรงบิดปานกลาง ซึ่งเป็นจุดที่มีการสูญเสียพลังงานน้อยที่สุด เมื่อนำมอเตอร์มาทำงานที่แรงดัน 12V ก็สามารถใช้หลักการนี้ช่วยหาดำเนินการทำงานที่ดีที่สุดได้ (Maxon, 2020).

2. หลักการทำงานของ H-Bridge

H-Bridge เป็นวงจรที่ประกอบด้วยสวิตช์ 4 ตัวจัดเป็นรูปตัว H ทำหน้าที่ควบคุมทิศทางการหมุนของมอเตอร์ DC โดยที่วงจรนี้จะทำหน้าที่เหมือนสะพานที่สามารถ สลับทิศทางการไหลของกระแสไฟฟ้า ที่เข้าสู่มอเตอร์ได้ ทำให้เราสามารถสั่งการให้มอเตอร์:

- หมุนไปข้างหน้า (Forward)
- หมุนถอยหลัง (Reverse)
- สั่งเบรก (Brake) โดยการช็อตขั้วมอเตอร์เข้าหากัน
- ปลปล่อยให้หมุนอิสระ (Freewheeling)

วงจรนี้เป็นพื้นฐานของ Motor Driver เช่น Cytron MD20A ที่ใช้ในงานทดลอง

3. การเปรียบเทียบโหมด Sign-Magnitude และ Locked Anti-Phase

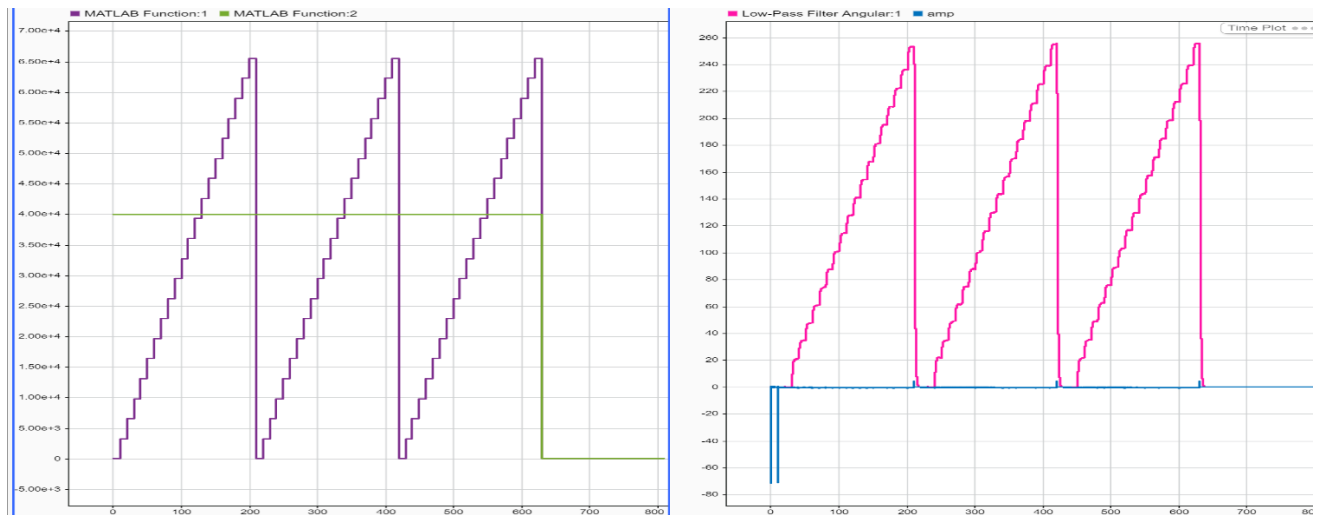
ในการควบคุมความเร็วมอเตอร์ผ่าน H-Bridge มักจะใช้สัญญาณ PWM โดยมีโหมดการควบคุมหลัก ๆ สองแบบคือ Sign-Magnitude และ Locked Anti-Phase โหมด Sign-Magnitude ใช้ PWM ควบคุม

ความเร็ว และใช้สัญญาณแยกกำหนดทิศทาง ทำให้ควบคุมง่ายและสูญเสียพลังงานน้อย ส่วนโหมด Locked Anti-Phase ใช้เพียงสัญญาณ PWM เดียว โดยค่าที่ต่ำกว่า 50% หมายถึงหมุนถอยหลัง เท่ากับ 50% คือหยุด และมากกว่า 50% คือหมุนไปข้างหน้า แม้จะตอบสนองไวกว่า แต่ทำให้มอเตอร์ร้อนง่ายและใช้พลังงานมากกว่า

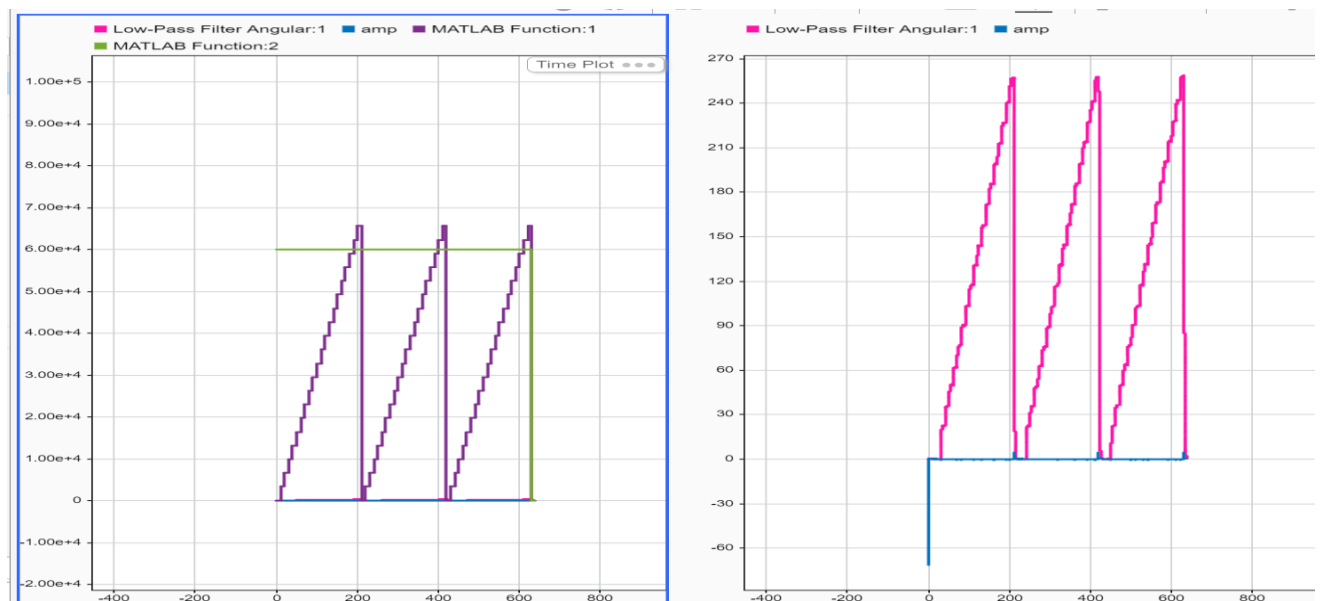
ขั้นตอนการดำเนินงาน

1. ต่อชุดทดลองเข้ากับแหล่งจ่ายไฟ และคอมพิวเตอร์
2. Setup block ใน Simulink เพื่อใช้ในการอ่านและแปลงค่าต่างๆ
3. เก็บค่าที่ Duty Cycle ต่างๆ ทุกๆ 10 duty cycle 3 ครั้งเพื่อนำค่าที่ได้มาสร้างกราฟและเปลี่ยน Frequency

ผลการทดลอง



รูปภาพ 2 ผลตอบสนองของ DC Motor ต่อการเปลี่ยนแปลงค่า Duty Cycle



รูปภาพ 3 ผลตอบสนองของ DC Motor ต่อการเปลี่ยนแปลงค่า Duty Cycle

สรุปผลการทดลอง

จากการทดลองทำให้เราทราบว่า เมื่อค่า Duty cycle เปลี่ยนไปจะทำให้ค่าต่างๆเปลี่ยนไปเนื่องจากการส่งคลื่นสัญญาณแบบ Duty cycle จะเป็นการส่ง Current แบบติดต่อกัน ทำให้เหมือนเราจ่าย Current ที่น้อยลงในการจ่าย duty ที่น้อยลง

อภิปรายผล

จากการวิเคราะห์การทดลองพบว่า angular Velocity จะแปรเปลี่ยนไปตาม Duty cycle ที่เปลี่ยนไปเนื่องจาก Current จะมีผลต่อค่าความเร็วและ Frequency ที่ motor ทำได้โดยตรง ทำให้ที่แต่ละ Duty cycle ที่ต่างกันจะมี Frequency ที่ต่างกัน

และจากการวิเคราะห์ current จะเห็นได้ว่า current จะกระชากในตอนแรกเนื่องจากคดลวดในตัว motor และกลับมาใช้งานปกติในช่วงที่ motor ทำงานคงที่แล้ว และเมื่อ motor หยุดทำงานไฟจะไหลย้อนกลับเนื่องจาก back EMF ของขดลวดใน motor

ข้อเสนอแนะ

อ้างอิง

1. Texas Instruments (2015). Understanding H-Bridge Motor Drivers.
2. Infineon (2018). DC Motor Control Fundamentals.

LAB 2.2: Stepper Motor

การทดลองที่ 1: การศึกษาพฤติกรรมของ Stepper Motor ในโหมดการขับเคลื่อนต่างๆ

จุดประสงค์

1. เพื่อศึกษารูปแบบการขับเคลื่อน (Drive Modes) แบบ Wave Drive, Full Step, Half Step และ Microstep
2. เพื่อวิเคราะห์ความสัมพันธ์ระหว่างความถี่ของสัญญาณ Pulse (Frequency) กับความเร็วรอบของมอเตอร์ (Speed) ในแต่ละโหมด
3. เพื่อศึกษาปรากฏการณ์การสูญเสียตำแหน่ง (Loss Step) และวิเคราะห์ปัจจัยที่ส่งผลให้เกิด Loss Step
4. เพื่อเปรียบเทียบข้อดีและข้อเสียของการขับเคลื่อนแต่ละรูปแบบ (Drive Modes) เพื่อเลือกใช้ให้เหมาะสมกับงานจริง

สมมติฐาน

ในการทำงานแต่ละโหมดของ Stepper Motor จะส่งผลให้การหมุนของมอเตอร์ใช้สัญญาณความถี่ที่สูงขึ้น เพื่อให้ได้ความเร็วเท่าเดิม และความเร็วรอบของ Stepper Motor จะแปรผันตรงกับความถี่ของสัญญาณ

ตัวแปร

3. ตัวแปรต้น:
 - ความถี่สัญญาณ (Stepper Frequency),
 - โหมดการขับเคลื่อน (Drive Mode: Full, Half, Micro 1/8, 1/16)
 - อัตราเร่ง (Acceleration)
4. ตัวแปรตาม:
 - ความเร็วรอบของมอเตอร์ (Motor Angular Velocity)
 - จุดที่เกิด Loss Step
5. ตัวแปรควบคุม:
 - แรงดันไฟฟ้าขาเข้า
 - มอเตอร์ไดรเวอร์ (DRV8825)
 - ภาระโหลดของมอเตอร์ (No Load)

เอกสารและงานวิจัยที่เกี่ยวข้อง

1. หลักการทำงานของ Stepper Motor

Stepper Motor คือมอเตอร์ไฟตรง ประเภทหนึ่งที่มีการหมุนแบ่งออกเป็นสเต็ป หรือขั้นย่อย ๆ ที่เท่ากัน โดยสามารถควบคุมให้มอเตอร์หมุนได้ตามจำนวนสเต็ปที่ต้องการ และยังสามารถทำให้มอเตอร์คงตำแหน่งนั้นไว้ด้วยการจ่ายกระแสไฟฟ้า ในการควบคุมตำแหน่งของมอเตอร์ประเภทนี้

หลักการทำงาน คือจ่ายไฟให้แก่ขดลวดที่ติดตั้งอยู่บน stator เพื่อให้ส่งสนามแม่เหล็กออกมา และ rotor จะมีแกนเป็นแม่เหล็กถาวร การจ่ายกระแสเข้าไปที่ขดลวดทำให้เกิดแรงดึงดูดจากสนามแม่เหล็ก ทำให้ rotor หมุน โดยเมื่อเราทำการจ่ายกระแสให้กับขดลวดแต่ละขดอย่างต่อเนื่องจะทำให้มอเตอร์หมุนอย่างต่อเนื่อง และหากเราต้องการให้มอเตอร์หมุนไปตามจำนวนสเต็ปที่กำหนดก็สามารถทำได้โดยการจ่ายพัลส์ กระแสผ่านขดลวดตามจำนวนสเต็ปที่ต้องการ โดยเมื่อจ่ายไฟให้แก่ขดลวดชุด A-A ก็จะทำให้แกนหมุนถูกผลักอยู่ที่ตำแหน่งนั้น ไม่เคลื่อนที่ไปไหนแม้มีแรงกระทำจากภายนอก โดยการหมุนในทิศตามเข็มนาฬิกาจะเกิดขึ้นเมื่อมีการป้อนกระแสให้เกิดสนามแม่เหล็กหมุนวนตามลำดับ A-B-C-D และหากต้องการให้หมุนในทิศทวนเข็มนาฬิกา ก็สามารถทำได้เช่นกัน โดยการกลับลำดับการป้อนกระแสให้แก่ขดลวดบนมอเตอร์

2. ประเภทของ Stepper Motor

จะแบ่งตามการต่อสายไฟได้ 2 ประเภท ดังนี้ Bipolar Stepper Motor (ไบโพลาร์) มี 2 ขดลวด (4 สาย) ไม่มีแท็ปกลาง และ Unipolar Stepper Motor (ยูนิโพลาร์) มีขดลวดพร้อมแท็ปกลาง (6-8 สาย)

แบ่งตามโครงสร้างภายใน ได้ 3 ประเภท ดังนี้

1. แบบแม่เหล็กถาวร (PERMANENT MAGNET-PM)

สเต็ปมอเตอร์แบบ PM จะมี STATOR ที่พันขดลวดไว้หลายๆ โพล โดยมี ROTOR เป็นรูปทรงกระบอกฟันเลื่อย และ ROTOR ทำด้วยแม่เหล็กถาวร เพื่อป้อนไฟกระแสตรง ให้กับขด STATOR จะทำให้เกิดแรงแม่เหล็กไฟฟ้าผลักต่อ ROTOR ทำให้มอเตอร์หมุนมอเตอร์แบบ PM จะเกิดแรงดูดยึดให้ ROTOR หยุดอยู่กับที่แม้จะไม่ได้ป้อนไฟเข้าขดลวด

2. แบบแปรค่ารีลักแตนซ์ (VARIABLE RELUCTANCE-VR)

สเต็ปมอเตอร์แบบ VR จะมีการหมุน ROTOR ได้อย่างอิสระ แม้จะไม่ได้จ่ายไฟให้ ROTOR ทำจากสารเฟอร์โรแมกเนติก กำลังอ่อน มีลักษณะเป็นฟันเลื่อย รูปทรงกระบอกโดยจะมีความสัมพันธ์ โดยตรงกับจำนวน โพล STATOR แรงบิดที่เกิดขึ้นจะไปหมุน ROTOR ไปในเส้นทางของอำนาจแม่เหล็กที่มีค่ารีลักแตนซ์ต่ำที่สุด ตำแหน่งที่จะเกิดแน่นอนและมีเสถียรภาพแต่จะเกิดขึ้นได้หลายๆ จุดดังนั้นเมื่อป้อนไฟเข้าขดลวดต่างๆ ในมอเตอร์แตกต่างกันไป ก็ทำให้มอเตอร์ หมุนไปตำแหน่งต่างๆ กัน ROTOR ของ VR จะมีความเฉื่อยของ ROTOR น้อยจึงมีความเร็วรอบสูงกว่ามอเตอร์แบบ PM

3. แบบผสม (HYBRID-H)

สเต็ปมอเตอร์แบบ H จะเป็นลูกผสมของ VR กับ PM โดยจะมี STATOR คล้ายกับที่ใช้ใน VR ROTOR มีหมวกหุ้ม ปลายซึ่งมีลักษณะของสารแม่เหล็กที่มีกำลังสูง โดยการควบคุมขนาดรูปร่างของหมวกแม่เหล็กอย่างดีทำให้ได้มุม การหมุนและครั้งน้อยและแม่นยำ ข้อดีก็คือ ให้แรงบิดสูงและมีขนาดกระทัดรัด และให้แรงดูดยึด ROTOR นิ่งกับที่ตอนไม่จ่ายไฟ

3. Drive Mode (Wave, Full, Half, Microstep)

โหมดขับเคลื่อน	จำนวนขดลวดที่ทำงาน	แรงบิดสัมพัทธ์	ความละเอียดสัมพัทธ์	ความราบรื่น
Wave Drive	1	ต่ำสุด	1x (มาตรฐาน)	ต่ำ
Full Step	2	สูงสุด	1x (มาตรฐาน)	ต่ำ
Half Step	1 สลับ 2	ปานกลาง (ไม่สม่ำเสมอ)	2x (สองเท่า)	ปานกลาง
Microstep	2 (ปรับกระแส)	สูง (แต่ลดที่ความเร็วสูง)	สูงมาก (เช่น 16x, 32x)	สูงมาก

4. การเกิด Loss Step

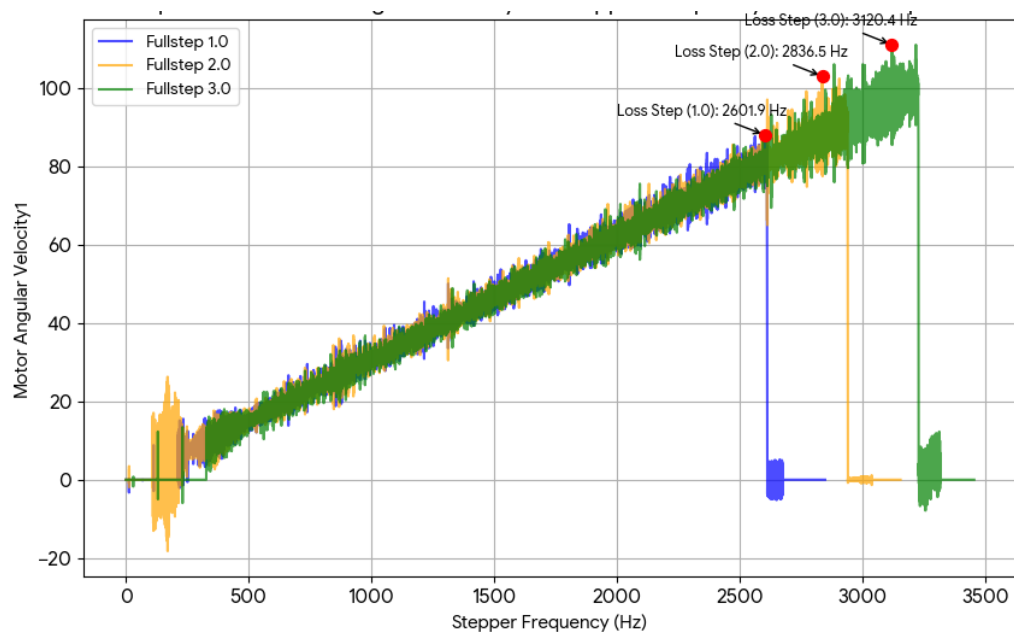
Loss step คือสภาวะที่มอเตอร์ไม่หมุนไปตามคำสั่งที่ถูกต้อง ทำให้ตำแหน่งสุดท้ายผิดพลาด เกิดจาก แรงบิดไม่พอ (Torque Deficiency), การป้อนไฟไม่เหมาะสม, ความเร็วสูงเกินไป, หรือโหลดหนักเกินที่มอเตอร์รับได้ ซึ่งทำให้สนามแม่เหล็กของสเตเตอร์ไม่สามารถจับโรเตอร์ให้อยู่ในตำแหน่งที่ต้องการได้ทันเวลา ส่งผลกระทบให้ระบบหยุดทำงาน

ขั้นตอนการดำเนินงาน

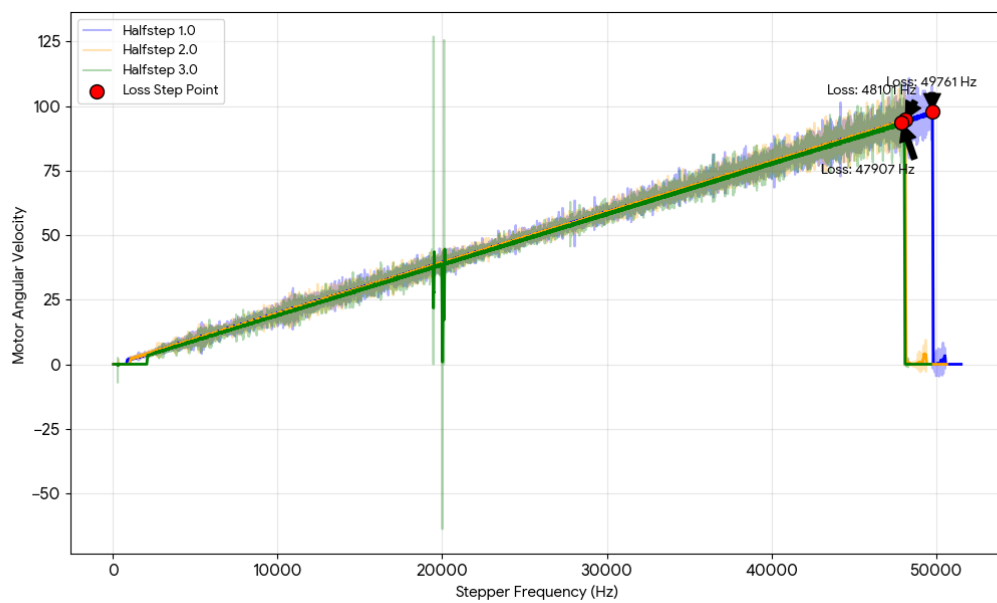
1. โดยเชื่อมต่อ STM32G474RE เข้ากับ Stepper Motor Driver (DRV8825) และ Encoder
2. อัปโหลดไฟล์ motorxplorer.bin ลงในบอร์ด และเปิดโปรแกรม Simulink
3. กำหนดโหมดการขับเคลื่อน (M0, M1, M2) สำหรับ Full Step, Half Step, และ Microstep ผ่าน Simulink Block
4. ทำการทดลองโดยปล่อยสัญญาณ Pulse แบบ Ramp-up
5. บันทึกค่าความเร็วรอบ (Angular Velocity) เทียบกับความถี่ (Frequency) และสังเกตจุดที่ความเร็วตกลงสู่ศูนย์ทันที (Loss Step)

ผลการทดลอง

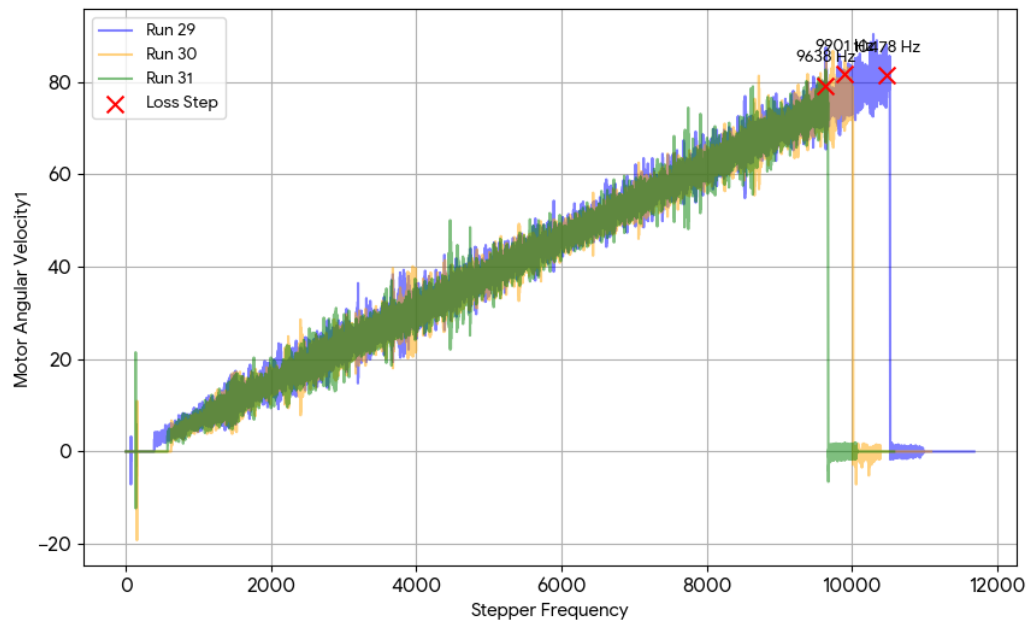
จากการทดลองทำซ้ำในแต่ละ Drive mode ของ Stepper Motor ได้กราฟการเปรียบเทียบ Frequency vs Speed ของแต่ละ Mode (Full, Half, Micro-step) ออกมาดังนี้



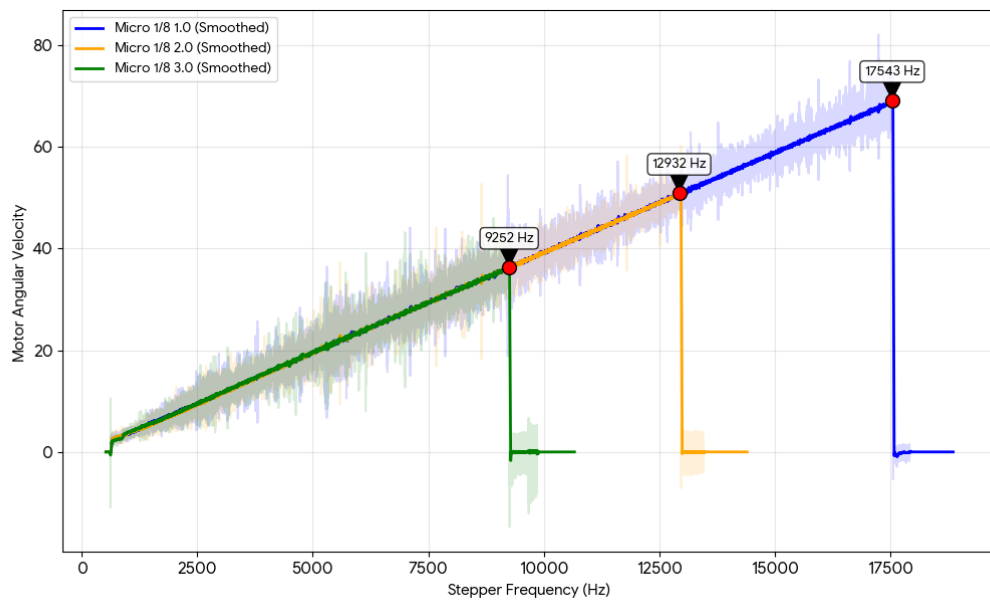
รูปภาพ 4 Full Step Mode



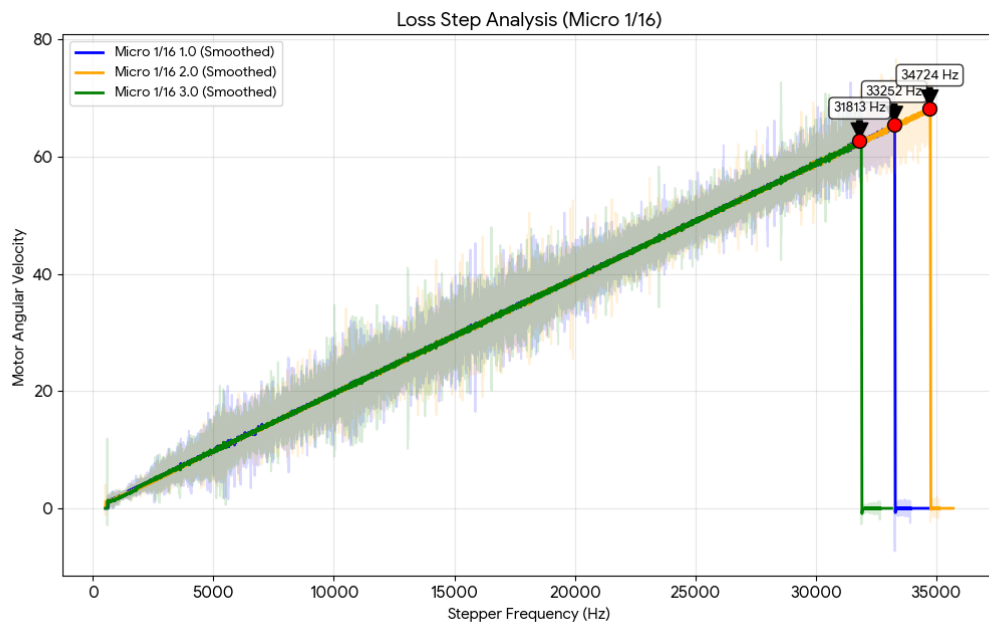
รูปภาพ 5 Half Step Mode



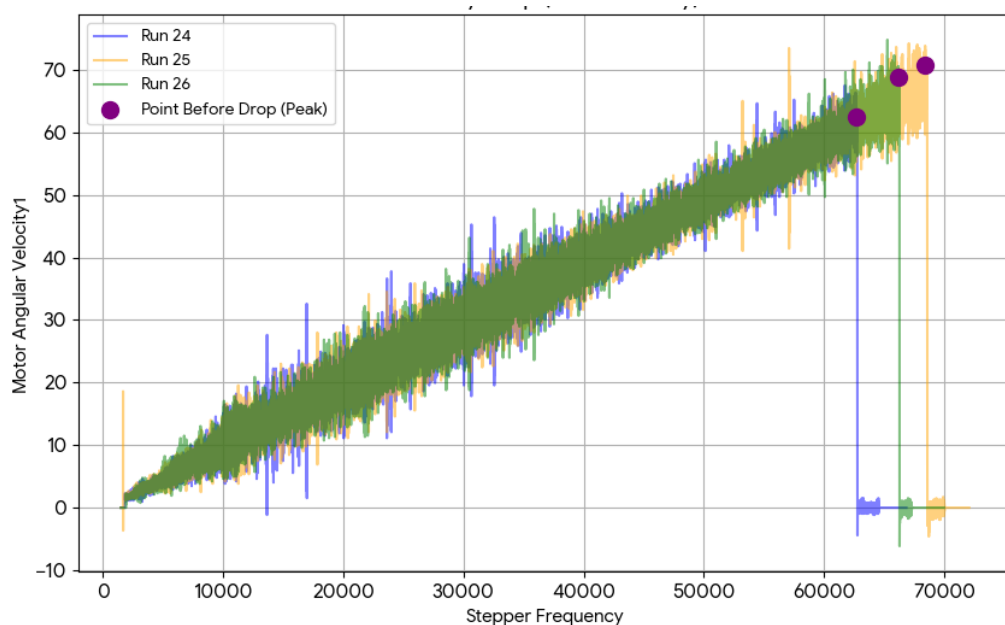
รูปภาพ 6 Microstep 1/4 Step Mode



รูปภาพ 7 Microstep 1/8 Step Mode



รูปภาพ 8 Microstep 1/16 Step Mode



รูปภาพ 9 Microstep 1/32 Step Mode

สรุปผลการทดลอง

จากผลการทดลองพบว่า ความเร็วรอบของ Stepper Motor แปรผันตรงกับความถี่สัญญาณ โดยในโหมด Full Step สามารถทำความเร็วสูงได้ที่มีความถี่ต่ำแต่มีความสั่นสะเทือนขณะหมุนสูง ในขณะที่โหมด Microstep เคลื่อนที่ราบรื่นและลดแรงสั่นสะเทือนได้อย่างชัดเจน แต่ใช้ความถี่สัญญาณที่สูงขึ้นเป็นทวีคูณเพื่อให้ได้ความเร็วรอบเท่าเดิม ซึ่งสอดคล้องกับจุด Loss Step ที่ขยับไปสู่ย่านความถี่สูงขึ้นตามความละเอียด

ของ Microstep การกำหนดอัตราเร่ง (Acceleration) ที่สูงเกินไป ส่งผลให้มอเตอร์เกิดภาวะ Loss Step ได้เร็วขึ้นที่ความเร็วรอบต่ำ เนื่องจากแรงบิดของมอเตอร์ไม่เพียงพอที่จะเอาชนะความเฉื่อยของระบบได้ทันที

อภิปรายผล

1. พฤติกรรมความชันของกราฟ (Slope Characteristics)

สาเหตุที่ความชันของกราฟลดลงเมื่อเพิ่มความละเอียดของ Microstep เกิดจากขนาดของมุมก้าว (Step Angle) ที่เล็กลง โดยในโหมด Full Step มอเตอร์หมุน 1.8 องศาต่อ Pulse แต่ในโหมด Microstep 1/16 มอเตอร์จะหมุนเพียง $1.8/16 = 0.1125$ องศาต่อ Pulse ดังนั้นเพื่อให้ได้ความเร็วรอบเชิงมุม (ω) เท่ากัน โหมด Microstep จึงต้องการความถี่ที่สูงกว่าดังสมการ $\omega = k \times f$ โดยค่า k จะลดลงเมื่อตัวหารของ Microstep เพิ่มขึ้น ข้อมูลจากกราฟยืนยันความสัมพันธ์นี้ โดยความถี่สูงสุดของ Microstep 1/32 (ประมาณ 68 kHz) มีค่าประมาณ 2 เท่าของ Microstep 1/16 (ประมาณ 34 kHz)

2. การเกิด Loss Step และความสัมพันธ์ของแรงบิด (Torque-Speed Relationship)

การเกิด Loss Step ณ จุดสิ้นสุดของกราฟ เกิดจากการที่แรงบิดที่มอเตอร์สร้างได้ (Pull-out Torque) ลดลงเมื่อความเร็วรอบเพิ่มขึ้น เนื่องจากผลของแรงดันต้านกลับ (Back EMF) และค่าความเหนี่ยวนำ (Inductance) ที่สูงขึ้นที่ความถี่สูง ทำให้กระแสไหลเข้าขดลวดไม่ทัน เมื่อแรงบิดที่ผลิตได้น้อยกว่าแรงเสียดทานและความเฉื่อยของโรเตอร์ มอเตอร์จึงหยุดหมุนกะทันหัน

3. ผลกระทบของความเร่ง (Inertia & Acceleration)

จากสมการพลศาสตร์ $T_{total} = J \cdot \alpha + T_{load}$ การทดลองที่ใช้อัตราเร่งสูงต้องการแรงบิดรวมมากกว่าเพื่อเอาชนะความเฉื่อย ส่งผลให้แรงบิดสำรองเหลือลดลง จึงทำให้จุดตัดที่แรงบิดไม่เพียงพอเกิดขึ้นเร็วที่ความถี่ต่ำกว่าการทดลองที่ใช้อัตราเร่งน้อยดังที่ปรากฏชัดเจนในกราฟ Microstep 1/8

4. ผลกระทบของอุณหภูมิ (Temperature Effect)

อุณหภูมิสะสมของ Stepper Motor เป็นอีกปัจจัยสำคัญที่ส่งผลต่อการเกิด Loss Step เมื่อมอเตอร์ทำงานต่อเนื่อง และมีอุณหภูมิสูงขึ้น ค่าความต้านทานไฟฟ้าของขดลวด (Winding Resistance) จะเพิ่มขึ้นตามคุณสมบัติของตัวนำ ทำให้กระแสไฟฟ้าขับเคลื่อนไหลผ่านได้น้อยลง ส่งผลให้แรงบิด (Torque) ที่มอเตอร์ผลิตได้ลดต่ำลง นอกจากนี้ความร้อนที่สูงเกินไปอาจส่งผลให้เกิดการลดลงของสนามแม่เหล็กชั่วคราว (Thermal Demagnetization) ทำให้ Torque Margin ของระบบลดลง ส่งผลให้มอเตอร์ที่มีความร้อนสะสมมีโอกาสเกิด Loss Step ได้ง่ายขึ้นหรือเกิดที่ความเร็วรอบต่ำกว่าเมื่อเทียบกับมอเตอร์ในสภาวะปกติ

ข้อเสนอแนะ

1. เพื่อป้องกันการเกิด Loss Step ในช่วงออกตัวหรือหยุด ควรใช้รูปแบบการเร่งความเร็วแบบ S-Curve หรือ Trapezoidal แทนการเร่งแบบ Linear หรือกระชากทันที เพื่อลดแรงบิดที่ต้องใช้ขณะความเฉื่อย
2. การใช้งานที่ความถี่สูงหรือโหลดหนักต่อเนื่อง อาจทำให้ Driver และ Motor ร้อนจัด ควรติดตั้ง Heat Sink หรือพัดลมระบายความร้อนให้เหมาะสม

3. ใช้บอร์ดทดลองที่มี Misalignment น้อยเพื่อให้ได้ผลการทดลองที่มีประสิทธิภาพการขับเคลื่อนที่ราบรื่นมากกว่านี้

อ้างอิง

1. <https://suwitkiravittaya.eng.chula.ac.th/B2i2019BookWeb/stepmotor.html>
2. [https://scma.co.th/blog/post/stepper-motor#:~:text=%E0%B8%AB%E0%B8%A5%E0%B8%B1%E0%B8%81%E0%B8%81%E0%B8%B2%E0%B8%A3%E0%B8%97%E0%B8%B3%E0%B8%87%E0%B8%B2%E0%B8%99%E0%B8%82%E0%B8%AD%E0%B8%87%20Stepper%20Motors,%E0%B9%84%E0%B8%9F%E0%B8%9F%E0%B9%89%E0%B8%B2%E0%B9%84%E0%B8%AB%E0%B8%A5%E0%B8%9C%E0%B9%88%E0%B8%B2%E0%B8%99%20%E0%B8%AA%E0%B9%88%E0%B8%A7%E0%B8%99%E0%B9%82%E0%B8%A3%E0%B9%80%E0%B8%95%E0%B8%AD%E0%B8%A3%E0%B9%8C%20\(](https://scma.co.th/blog/post/stepper-motor#:~:text=%E0%B8%AB%E0%B8%A5%E0%B8%B1%E0%B8%81%E0%B8%81%E0%B8%B2%E0%B8%A3%E0%B8%97%E0%B8%B3%E0%B8%87%E0%B8%B2%E0%B8%99%E0%B8%82%E0%B8%AD%E0%B8%87%20Stepper%20Motors,%E0%B9%84%E0%B8%9F%E0%B8%9F%E0%B9%89%E0%B8%B2%E0%B9%84%E0%B8%AB%E0%B8%A5%E0%B8%9C%E0%B9%88%E0%B8%B2%E0%B8%99%20%E0%B8%AA%E0%B9%88%E0%B8%A7%E0%B8%99%E0%B9%82%E0%B8%A3%E0%B9%80%E0%B8%95%E0%B8%AD%E0%B8%A3%E0%B9%8C%20()

Leb 2.3: Brushless DC Motor (BLDC)

การทดลองที่ 1: การศึกษาความสัมพันธ์ระหว่างความถี่ของสัญญาณ Back-EMF กับความเร็วรอบของมอเตอร์ BLDC

จุดประสงค์

1. เพื่อศึกษาหลักการทำงานของ Brushless DC Motor (BLDC)
2. เพื่อวิเคราะห์ลักษณะสัญญาณ Back EMF (BEMF) แบบ Trapezoidal และอธิบายการเกิด Phase Shift ระหว่างสัญญาณแต่ละ Phase
3. เพื่อวิเคราะห์ความสัมพันธ์ระหว่างความถี่ของสัญญาณ Back-EMF กับความเร็วรอบของมอเตอร์ (RPM)

สมมติฐาน

ความถี่ของสัญญาณไฟฟ้า Back-EMF ที่วัดได้ จะแปรผันตรงกับความเร็วรอบของมอเตอร์ (RPM)

ตัวแปร

6. ตัวแปรต้น:
 - ค่าความเร็วรอบของมอเตอร์ (RPM) ที่กำหนดผ่านโปรแกรม MC Workbench
7. ตัวแปรตาม:
 - ความเร็วรอบจริง
 - ลักษณะความถี่ของสัญญาณที่วัดได้จาก Oscilloscope
8. ตัวแปรควบคุม:
 - แรงดันไฟฟ้าขาเข้า
 - BLDC Motor รุ่น มอเตอร์รุ่น A2212 13T 1000KV
 - บอร์ดควบคุม Nucleo STM32G474RE
 - Driver X-NUCLEO-IHM08M1

เอกสารและงานวิจัยที่เกี่ยวข้อง

1. หลักการทำงานของ BLDC Motor

หลักการทำงานของ BLDC Motor เป็นกระบวนการที่อาศัยหลักการเหนี่ยวนำแม่เหล็กไฟฟ้า โดยการใช้ระบบอิเล็กทรอนิกส์ในการควบคุมทิศทางการไหลของกระแสไฟฟ้า องค์ประกอบสำคัญในการทำงานคือ Stator ซึ่งเป็นส่วนที่อยู่กับที่ ประกอบด้วยขดลวดทองแดง (Coil/Winding) ซึ่งปกติจะมี 3 ชุด (A, B, C หรือ U, V, W) พันอยู่รอบแกนเหล็ก และ Rotor ซึ่งเป็นส่วนที่หมุนได้ประกอบด้วยแม่เหล็กถาวร (Permanent Magnets) ติดตั้งอยู่รอบแกนกลาง เมื่อจ่ายกระแสไฟฟ้ากระแสตรง (DC) ไปยังชุดขดลวดใน Stator ทำให้ขดลวดนั้นกลายเป็นแม่เหล็กไฟฟ้าชั่วคราว สนามแม่เหล็กไฟฟ้าที่เกิดขึ้นนี้จะทำปฏิกิริยากับสนามแม่เหล็ก

ถาวรบน Rotor เกิดเป็นแรงดูดและแรงผลัก ซึ่งสร้างแรงบิดทำให้ Rotor เริ่มหมุนเพื่อพยายามจัดเรียง ขั้วแม่เหล็กให้ตรงกัน สิ่งที่ทำให้มอเตอร์หมุนอย่างต่อเนื่องคือกระบวนการ "การสับเปลี่ยนขั้วแม่เหล็ก" (Commutation) ซึ่งเป็นการที่วงจรควบคุมสลับการจ่ายกระแสไฟไปยังขดลวดชุดถัดไปก่อนที่ขั้วแม่เหล็กของ Rotor และ Stator จะเรียงตัวตรงกันเพื่อให้แรงบิดยังคงทิศทางการหมุนต่อไป โดยการที่กระบวนการจะดำเนินการได้อย่างถูกต้องระบบต้องทราบตำแหน่งที่แน่นอนของ Rotor ซึ่งทำได้สองวิธี คือ ติดตั้งเซนเซอร์ Hall Effect เพื่อตรวจจับตำแหน่งแม่เหล็กถาวรโดยตรง และอาศัยการวัดค่าแรงดันไฟฟ้าที่เกิดขึ้นในขดลวดที่ไม่ได้ใช้งานชั่วคราว หรือที่เรียกว่า Back EMF เพื่อคำนวณหาตำแหน่ง Rotor แทน

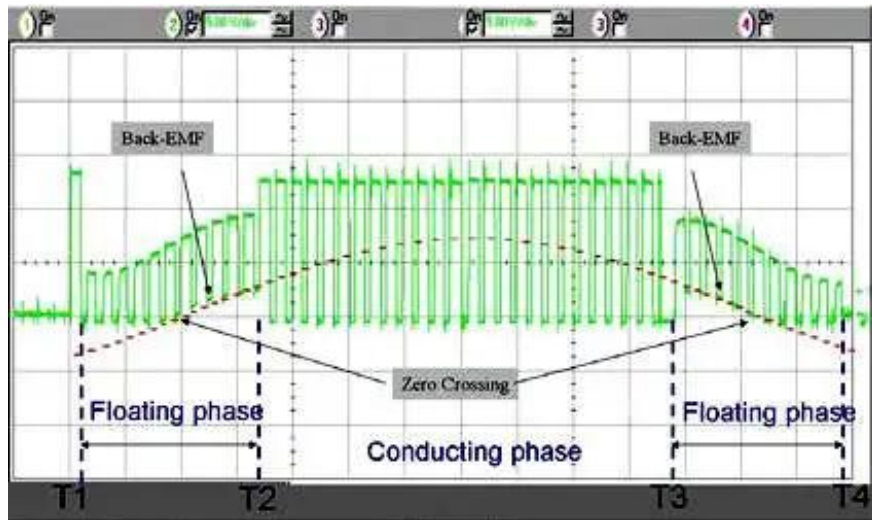
2. ความแตกต่างระหว่าง BLDC และ PMSM

BLDC (Brushless DC) และ PMSM (Permanent Magnet Synchronous Motor) ใช้แม่เหล็กถาวรเป็นแหล่งกำเนิดสนามแม่เหล็กบนโรเตอร์เหมือนกันทั้งคู่ ความแตกต่างของซึ่งเป็นตัวกำหนดลักษณะการทำงาน และการใช้งาน คือรูปคลื่น Back-EMF และวิธีการควบคุมของมอเตอร์แต่ละชนิด โดย BLDC ลักษณะของรูปคลื่น Back-EMF จะมีลักษณะเป็นรูปสี่เหลี่ยมคางหมูหรือขั้นบันไดซึ่งได้รับการออกแบบจากการพันขดลวดหรือโครงสร้างแม่เหล็ก ส่วน PMSM ออกแบบอย่างละเอียดเพื่อให้สร้างรูปคลื่น Back-EMF ที่เป็นไซน์ (Sinusoidal wave) ที่ราบเรียบเรียบ และต่อเนื่อง ถัดมาคือความแตกต่างด้านวิธีการควบคุม โดย BLDC จะเป็นการสับเปลี่ยนแบบหกจังหวะ (Six-step commutation) ซึ่งเป็นการจ่ายกระแสไฟเข้าไปที่ละคู่ขดลวดเป็นจังหวะ ๆ มักใช้ Hall effect ในการระบุตำแหน่งโรเตอร์ ส่วน PMSM ใช้การควบคุมแบบเวกเตอร์ (Field-Oriented Control หรือ FOC) ซึ่งใช้เซนเซอร์ตำแหน่งที่มีความแม่นยำสูง เช่น Encoder เพื่อคำนวณ และปรับทิศทางการของสนามแม่เหล็กอย่างต่อเนื่องตลอดเวลา ด้วยข้อแตกต่างเหล่านี้ส่งผลให้ BLDC มีความเรียบง่ายของตัวควบคุม และราคาถูกกว่า แต่ PMSM สามารถทำงานได้แม่นยำ และเงียบกว่า เนื่องจากไม่มีแรงบิดจากการสับเปลี่ยนจังหวะ

3. Floating และ Conducting Period (ช่วงลอยตัวและช่วงนำกระแส)

ในการควบคุมแบบ 6 จังหวะ (six-step commutation) สถานะของขดลวดสตาเตอร์ ทั้งสามเฟสจะสลับไปมาระหว่างสองสถานะ Conducting Period (ช่วงนำกระแส) คือช่วงเวลาที่มีการจ่ายกระแสไฟฟ้าไหลผ่านขดลวดเฟสนั้นๆ เพื่อสร้างสนามแม่เหล็กไฟฟ้าไปขับเคลื่อน Rotor และ Floating Period (ช่วงลอยตัว) คือสถานะของขดลวดเฟสที่สามที่ไม่ได้ถูกต่อเข้ากับแหล่งจ่ายไฟโดยตรง ในช่วงนี้จะไม่มีกระแสไหลผ่านเฟสที่ลอยตัว แต่เป็นช่วงใช้สำหรับวัดค่าแรงดันไฟฟ้าเหนี่ยวนำ (Back EMF) ที่เกิดขึ้น เพื่อใช้กำหนดจังหวะการสับเปลี่ยนกระแส (commutation) ครั้งต่อไป

4. การวิเคราะห์ Zero Crossing



รูปภาพ 10 เฟสในระหว่างการสลับ PWM ปกติ

การวิเคราะห์จุดตัดศูนย์ (Zero Crossing Point: ZCP) ช่วยให้วงจรควบคุมทราบตำแหน่งที่แน่นอนของ Rotor โดยไม่ต้องใช้เซ็นเซอร์ Hall Effect ในระหว่างที่เฟสที่สามอยู่ในช่วง Floating Period วงจรควบคุมจะทำการวัดแรงดันไฟฟ้าที่ขั้วของเฟสนั้นเทียบกับจุดกลางของมอเตอร์ จุด ZCP คือจังหวะเวลาที่แรงดัน Back EMF ของเฟสที่ลอยตัวมีค่าเท่ากับแรงดันจุดกลางอ้างอิง จุด ZCP นี้เป็นตัวบ่งชี้ตำแหน่ง Rotor เมื่อตรวจพบจุด ZCP แล้ว ระบบควบคุมจะส่งช่วงเวลาออกไป 30 องศาไฟฟ้าก่อนจะทำการสับเปลี่ยนการจ่ายกระแส (Commutation) ไปยังเฟสถัดไปอย่างแม่นยำ

5. การวิเคราะห์สัญญาณ 3 Phase

การวิเคราะห์สัญญาณ 3 เฟสเป็นขั้นตอนสำคัญในการยืนยันคุณลักษณะของมอเตอร์ และสร้างแบบจำลองทางคณิตศาสตร์สำหรับระบบควบคุม ซึ่งความสัมพันธ์ระหว่างความถี่ของสัญญาณ Back EMF และความเร็วยรอบของมอเตอร์มีความสัมพันธ์โดยตรง และเป็นสัดส่วนกัน โดยสัญญาณ Back-EMF ที่วัดได้ในแต่ละเฟสจะมีลักษณะเป็นรูปสี่เหลี่ยมคางหมูที่ชัดเจน ซึ่งเป็นคุณสมบัติเด่นของมอเตอร์ BLDC โดยสัญญาณทั้งสามเฟสจะเหลื่อมกัน 120 องศาทางไฟฟ้า และควรมีความสมมาตรกันทั้งขนาดของแรงดันสูงสุด และความกว้างของช่วงราบ โดยสามารถวิเคราะห์ ความสัมพันธ์ระหว่างความเร็วยรอบของมอเตอร์ และความถี่ของสัญญาณ Back-EMF ได้โดยการเก็บข้อมูลความเร็วยรอบที่วัดได้จาก Tachometer และความถี่ที่วัดจากออสซิลโลสโคป แล้วนำมาพล็อตเป็นกราฟ ซึ่งจะได้ความสัมพันธ์เป็นกราฟเส้นตรงที่ผ่านจุดกำเนิด

ขั้นตอนการดำเนินงาน

1. ต่อชุดทดลองเข้ากับแหล่งจ่ายไฟ และคอมพิวเตอร์
2. นำสาย Probe ของ Oscilloscope ต่อเข้ากับจุดวัดสัญญาณเฟสของมอเตอร์ (Phase A, B, C)
3. เปิดโปรแกรม MC Workbench โดยสั่งการผ่าน Motor Pilot ให้มอเตอร์หมุนด้วยความเร็วคงที่
4. ปรับ Time/Div และ Volts/Div บนหน้าปัด Oscilloscope จนปรากฏรูปคลื่นสัญญาณ Back-EMF ครบ 1 ลูกคลื่นอย่างชัดเจนครบทั้ง 3 เฟส

5. เลือกเมนู Cursor บนหน้าจอ Oscilloscope จากนั้นเลือกโหมด Vertical Cursor เพื่อทำการวัดค่าทางเวลา ซึ่งจะปรากฏเส้นแนวตั้ง 2 เส้น (X1 และ X2) บนหน้าจอ
6. ทำการเลื่อนเส้น X1 ไปไว้ที่จุดเริ่มต้นของคาบสัญญาณ และเลื่อนเส้น X2 ไปไว้ที่จุดสิ้นสุดของลูกคลื่นในรอบเดียวกัน เพื่อให้ครอบคลุม 1 คาบเวลาทางไฟฟ้า
7. อ่าน และบันทึกค่าโดยค่า dX (Delta X) คือ ค่าคาบเวลา (Period, T) และค่า $1/dX$ (Frequency, f_{elec}) คือค่าความถี่ทางไฟฟ้า (Electrical Frequency)
8. นำค่าที่อ่านได้มาวิเคราะห์ ความเร็วรอบตามทฤษฎีจากสมการ $Speed(RPM) = \frac{120 \times f_{elec}}{P}$ โดย P มีค่าเท่ากับ 14 ตามที่ระบุไว้ใน Data sheet เปรียบเทียบค่าที่คำนวณได้กับค่าที่อ่านได้จากโปรแกรม และตรวจสอบค่าความคลาดเคลื่อนจากสูตร $\%Error = \left| \frac{Reference - \text{ค่าคำนวณ}}{Reference} \right| \times 100$

ผลการทดลอง

ตารางที่ 1 แสดงผลการทดลองครั้งที่ 1

การทดลองครั้งที่ 1				
Reference Speed (RPM)	ความถี่ BEMF ที่วัดได้ ($1/\Delta x$) (Hz)	Mechanical Speed (RPM)	ความเร็วรอบจากการคำนวณ (RPM)	ความคลาดเคลื่อน (%)
0	0	0	0	0
1998	331.1	1908	2838	48.74
3996	684.9	4050	5870.57	44.95
6000	1020	6060	8742.86	44.27
7998	1333	8064	11425.71	41.69
9864	1493	10440	12797.14	22.58

ตารางที่ 2 แสดงผลการทดลองครั้งที่ 2

การทดลองครั้งที่ 2				
Reference Speed (RPM)	ความถี่ BEMF ที่วัดได้ ($1/\Delta x$) (Hz)	Mechanical Speed (RPM)	ความเร็วรอบจากการคำนวณ (RPM)	ความคลาดเคลื่อน (%)
0	0	0	0	0
1998	343.6	1968	2945.14	49.65
3996	684.9	3930	5870.57	49.38
6000	1000	6114	8571.43	40.19
7998	1282	8328	10988.57	31.95
9864	1562	9870	13825.71	37.57

ตารางที่ 3 แสดงผลการทดลองครั้งที่ 1

การทดลองครั้งที่ 3				
Reference Speed (RPM)	ความถี่ BEMF ที่วัดได้ ($1/\Delta x$) (Hz)	Mechanical Speed (RPM)	ความเร็วรอบจากการคำนวณ (RPM)	ความคลาดเคลื่อน (%)
0	0	0	0	0
1998	342.5	1992	2935.71	47.37
3996	684.9	3960	5870.57	48.25
6000	1042	6114	8931.43	46.08
7998	1333	7920	11425.71	44.26
9864	1613	10050	13825.71	37.57

สรุปผลการทดลอง

จากการทดลองพบว่า สัญญาณ Back-EMF ที่วัดได้มีลักษณะเป็นรูปสี่เหลี่ยมคางหมู (Trapezoidal) สอดคล้องกับการควบคุมแบบ 6-Step และเมื่อเพิ่มความเร็วรอบมอเตอร์ (Mechanical Speed) ค่าความถี่ของสัญญาณ BEMF เพิ่มขึ้นตามลำดับ เมื่อนำค่าความถี่มาคำนวณกลับเป็นความเร็วรอบ พบว่ามีค่าสูงกว่าความเร็วรอบจริงอย่างมีนัยสำคัญ ส่งผลให้เกิดความคลาดเคลื่อน (Error) ในช่วง 37% - 48%

อภิปรายผล

จากการวิเคราะห์ผลการทดลองพบว่าเกิดการคลาดเคลื่อนกันอย่างมากระหว่าง Reference Speed และความเร็วรอบจากการคำนวณโดยสาเหตุอาจเกิดจากหลายสาเหตุ โดยทางผู้จัดทำมีความเห็นว่าเกิดจากความผิดพลาดจากการวัดคาบเวลา: จากการกำหนดช่วง (Cursor) ไม่ครอบคลุม 1 คาบเวลาทางไฟฟ้าที่สมบูรณ์ของสัญญาณ เนื่องจากการคำนวณได้ค่า RPM ที่สูงกว่าความเป็นจริงประมาณ 1.5 เท่า แสดงว่าค่าความถี่ที่อ่านได้สูงเกินจริง หรือวัดช่วงเวลาสั้นกว่าความเป็นจริง เช่นค่าที่ได้จากความเร็วในการทดลองครั้งที่ 3 คือ 1992 RPM ความถี่ที่ควรจะเป็น คือ $f_{elec} = \frac{1992 \times 14}{120}$ จะได้ค่าประมาณ 232.4 Hz ซึ่งสังเกตได้ว่าค่าในตารางที่ได้นั้นสูงกว่าทฤษฎีมาก

ข้อเสนอแนะ

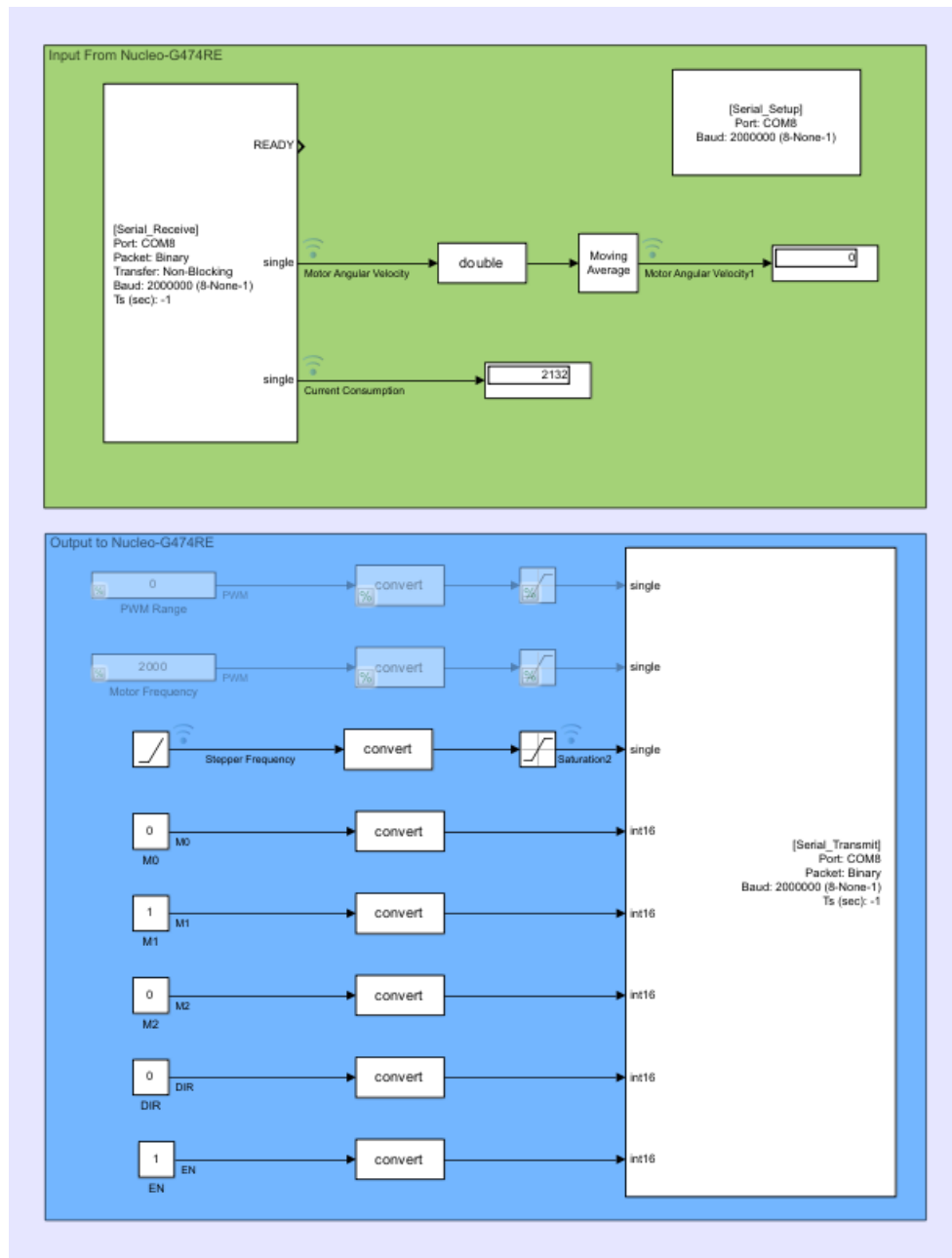
1. การวัดความถี่ด้วย Oscilloscope ควรปรับ Time/Div ให้เห็นลูกคลื่นครบอย่างน้อย 2-3 ลูกคลื่น และวาง Cursor วัดระยะห่างระหว่างจุดเดียวกันของลูกคลื่นเพื่อความแม่นยำสูงสุด

อ้างอิง

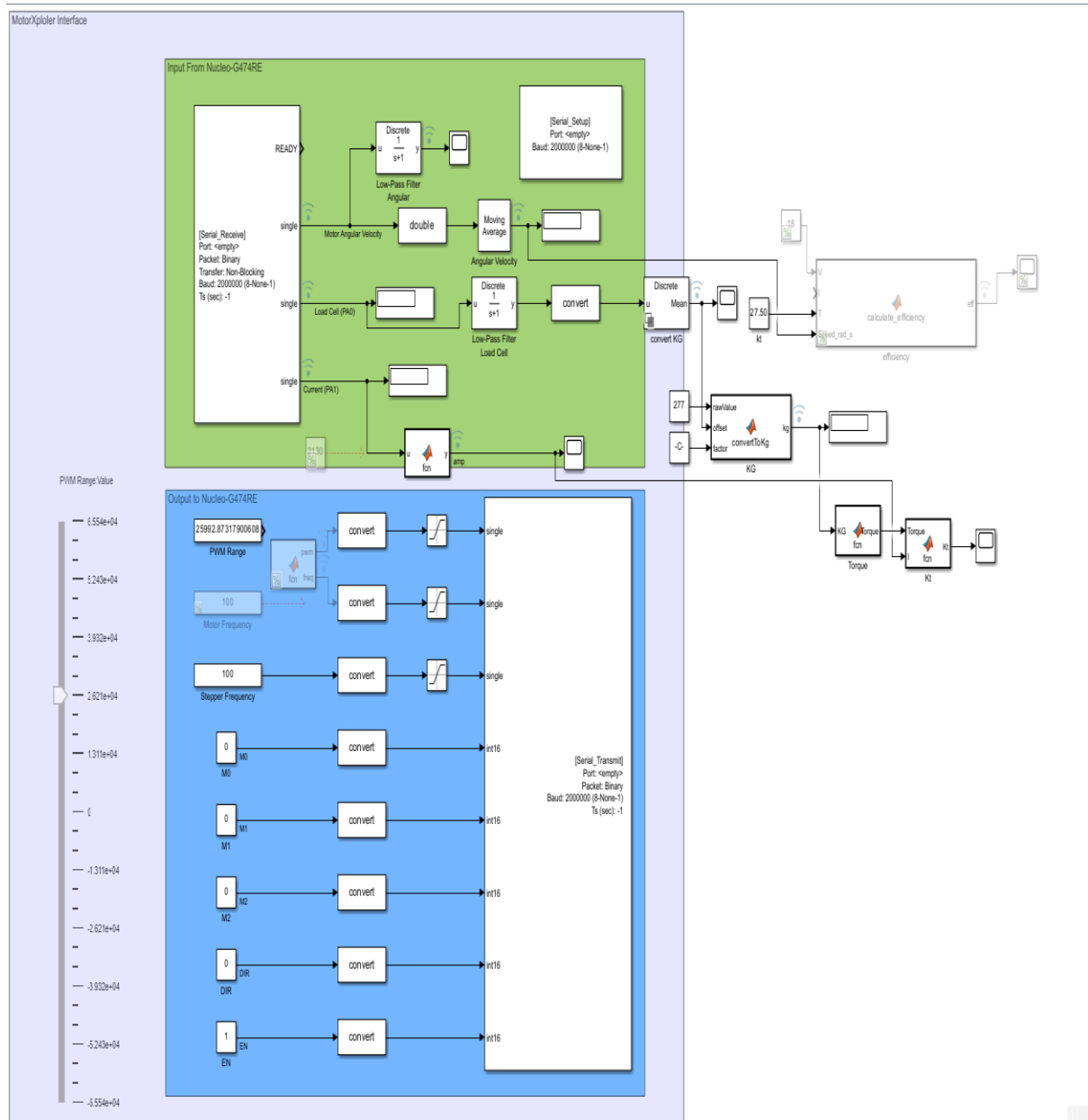
1. <https://www.atlantis-press.com/article/19902.pdf>
2. <https://share.google/rqz3MO9fHArYRNJE2>

3. <https://www.digikey.pl/pl/articles/controlling-sensorless-bldc-motors-via-back-emf>
4. <https://ww1.microchip.com/downloads/en/AppNotes/AN857-Brushless-DC-Motor-Control-00000857C.pdf>

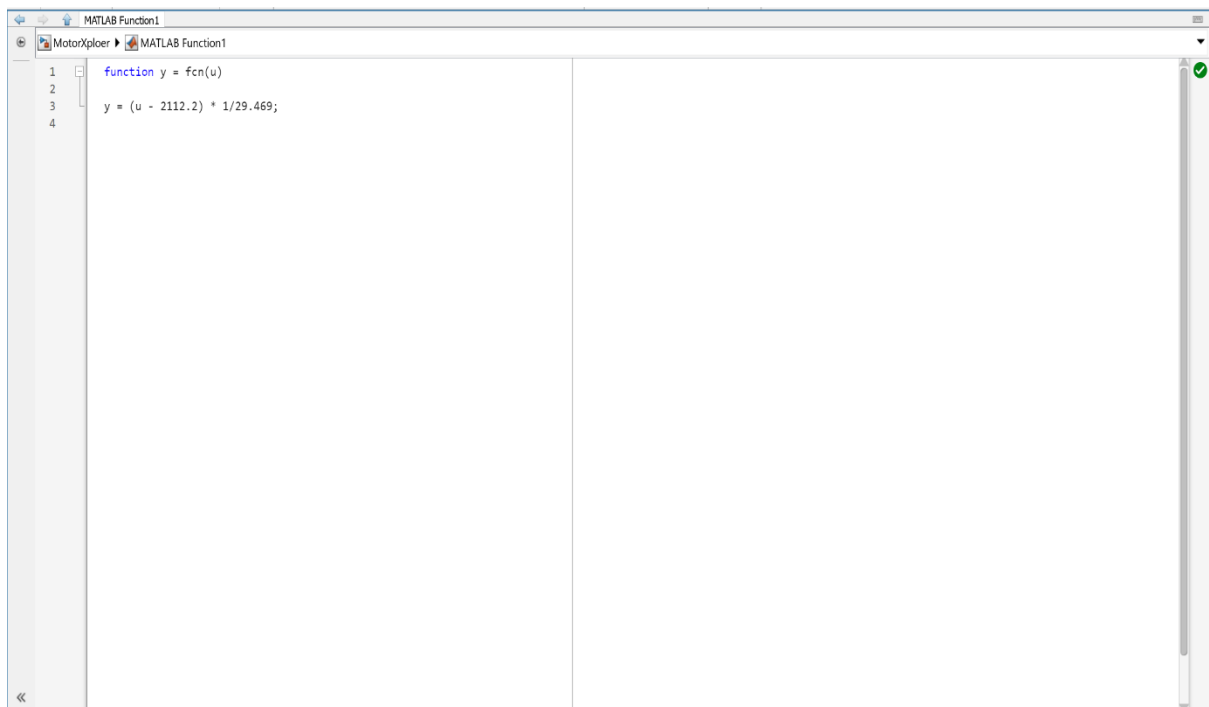
ภาคผนวก



หน้า Simulink สำหรับทดสอบ Stepper Motor โดยส่งคำสั่งควบคุมความเร็ว ผ่าน Ramp, Frequency, ทิศทาง, และโหมดการขับ ไปยังบอร์ด STM32 และในขณะเดียวกันก็อ่านค่า Feedback (ความเร็วที่วัดได้จริง และกระแสไฟฟ้าที่ใช้)



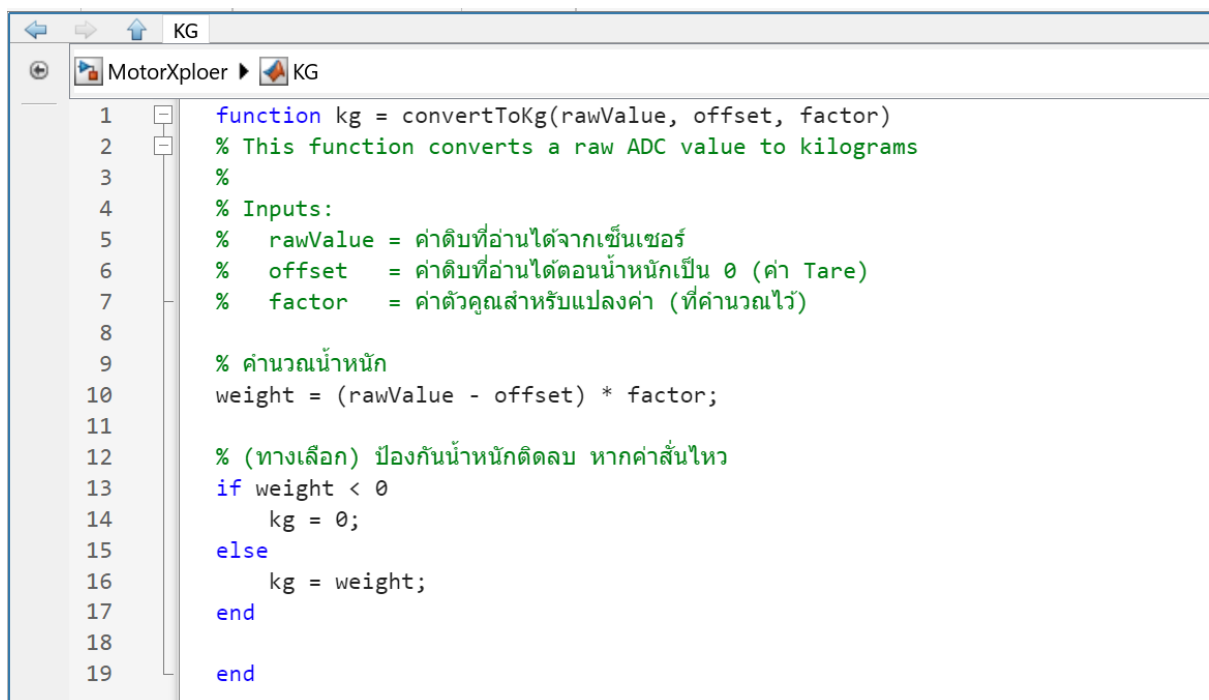
Block Simulink ที่ใช้



The image shows a MATLAB Function Editor window titled 'MATLAB Function1'. The editor contains the following code:

```
1 function y = fcn(u)
2
3 y = (u - 2112.2) * 1/29.469;
4
```

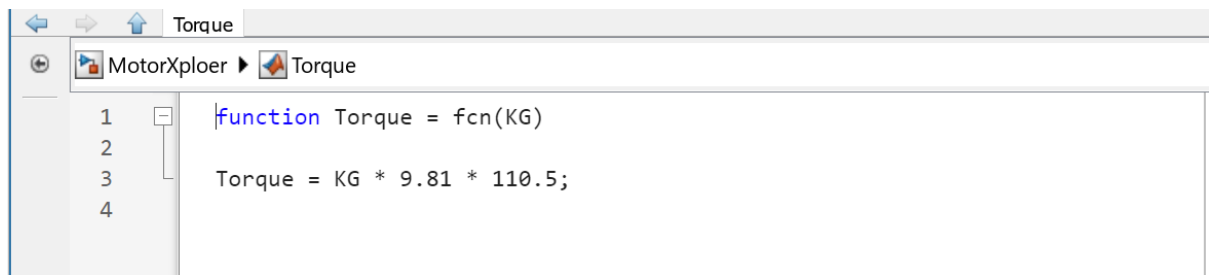
สมการ Linearization ที่ใช้



The image shows a MATLAB Function Editor window titled 'KG'. The editor contains the following code:

```
1 function kg = convertToKg(rawValue, offset, factor)
2 % This function converts a raw ADC value to kilograms
3 %
4 % Inputs:
5 % rawValue = ค่าดิบที่อ่านได้จากเซ็นเซอร์
6 % offset   = ค่าดิบที่อ่านได้ตอนน้ำหนักเป็น 0 (ค่า Tare)
7 % factor   = ค่าตัวคูณสำหรับแปลงค่า (ที่คำนวณไว้)
8
9 % คำนวณน้ำหนัก
10 weight = (rawValue - offset) * factor;
11
12 % (ทางเลือก) ป้องกันน้ำหนักติดลบ หากค่าสั้นไหว
13 if weight < 0
14     kg = 0;
15 else
16     kg = weight;
17 end
18
19 end
```

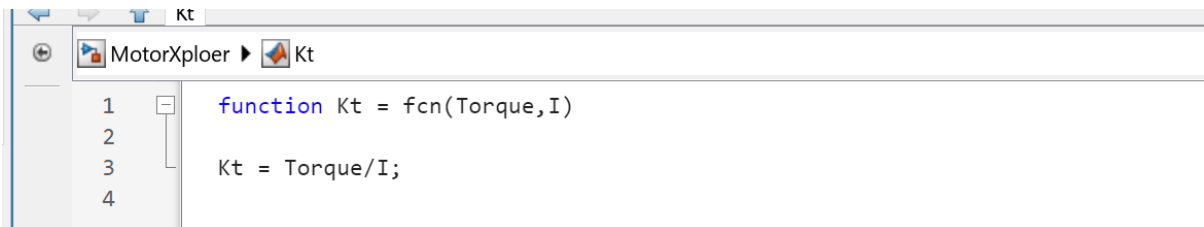
โปรแกรมที่ใช้แปลงค่าจาก Load cell เป็น KG



The image shows a MATLAB script editor window titled 'Torque'. The script defines a function 'Torque = fcn(KG)' and calculates the torque value based on the input KG. The script is as follows:

```
1 function Torque = fcn(KG)
2
3 Torque = KG * 9.81 * 110.5;
4
```

โปรแกรมที่ใช้หา $Torque_{Real}$



The image shows a MATLAB script editor window titled 'Kt'. The script defines a function 'Kt = fcn(Torque,I)' and calculates the Kt value based on the input Torque and I. The script is as follows:

```
1 function Kt = fcn(Torque,I)
2
3 Kt = Torque/I;
4
```

โปรแกรมที่ใช้หา K_t

```
MATLAB Function
MotorXploer MATLAB Function

1 function [pwm, freq] = fcn()
2
3 persistent freqList pwmStep sampleCount freqIndex pwmIndex roundCount;
4
5 % Initialize
6 if isempty(freqList)
7     freqList = [80000 80000 80000];
8     pwmStep = 0:0.05:1.0;
9     %pwmStep = 0:0.1:1.0;
10    sampleCount = 0;
11    freqIndex = 1;
12    pwmIndex = 1;
13    roundCount = 1;
14 end
15
16 % Stop after 4 rounds
17 if roundCount > 4
18     pwm = 0;
19     freq = 0;
20     return;
21 end
22
23 % Output
24 freq = freqList(freqIndex);
25 pwm = pwmStep(pwmIndex) * (65535); % <-- ตอนนี้เป็น double แล้ว
26
27 % Update counter
28 sampleCount = sampleCount + 1;
29
30 if sampleCount >= 10000
31     sampleCount = 0;
32     pwmIndex = pwmIndex + 1;
33
34     if pwmIndex > length(pwmStep)
35         pwmIndex = 1;
36         freqIndex = freqIndex + 1;
37
38         if freqIndex > length(freqList)
39             freqIndex = 1;
40             roundCount = roundCount + 1;
41         end
42     end
43 end
```

โปรแกรมที่ใช้ในการไล่ Duty cycle ไปเรื่อยๆ