



รายงานการวิจัย

65900

การรักษาความเร็วของเครื่องทดสอบความคงตัวของน้ำยางด้วยฟัซซี่ลอจิก
Speed Control Regulator for Latex Stability Test with Fuzzy logic

สัญญา ผาสุก

Sunya Pasuk

629.89

สุภชัย อรุณพันธ์

Supachai Arunphan

8 213

พรชัย แคล้วอ้อม

Pornchai Khlawoam

1554

สาขาวิศวกรรมไฟฟ้า คณะวิศวกรรมศาสตร์
มหาวิทยาลัยเทคโนโลยีราชมงคลศรีวิชัย

ได้รับการสนับสนุนทุนวิจัยจากมหาวิทยาลัยเทคโนโลยีราชมงคลศรีวิชัย
งบประมาณเงินรายได้ประจำปี พ.ศ. 2554

บทคัดย่อ

โครงการนี้เป็นการควบคุมความเร็วมอเตอร์ไฟฟ้ากระแสตรง 220 โวลต์ 1.4 แอมแปร์ 14000 รอบ/นาที โดยใช้ทฤษฎีการควบคุมแบบพีซีซึ่งล่อจิกเข้ามาใช้ในการควบคุมเพื่อให้สามารถควบคุมความเร็วมอเตอร์ไฟฟ้ากระแสตรงให้คงที่ที่ 14000 รอบ ต่อนาที โดยนำเอาไมโครคอนโทรลเลอร์ (MICROCONTROLLER) ตระกูล PIC เบอร์ 16F877 เป็นตัวประมวลผล เขียนโปรแกรมด้วยภาษา C ซึ่งไมโครคอนโทรลเลอร์เป็นตัวควบคุมการส่งสัญญาณทริกให้แก่ขาเกตของ IGBT ในวงจร CHOPPER ซึ่งทำหน้าที่เป็นสวิตซ์ ON - OFF เพื่อเปลี่ยนแปลง DUTY CYCLE ในการควบคุมแรงดันขั้วของมอเตอร์ การตรวจจับความเร็วของมอเตอร์โครงการนี้ใช้ TACHO GENERATOR เป็นตัวตรวจจับแล้วส่งสัญญาณเป็นแรงดันไฟฟ้ากระแสตรงตั้งแต่ 0-2 โวลต์ กลับไปยังไมโครคอนโทรลเลอร์ และแสดงผลด้วยจอ LCD



Abstract

This research was to design and construction of Fuzzy Logic Controller for Controlling speed of 220 volt 1.4 amp 140000 rpm Separately Excited DC motor. The Fuzzy Logic Controller is consisted of microcontroller PIC16F877 . Software is programmed in C language. Fuzzy Logic Controller will change duty cycle of signal to control IGBT in chopper circuit for changing terminal voltage of DC motor



สารบัญ

	หน้า
บทคัดย่อภาษาไทย	ข
บทคัดย่อภาษาอังกฤษ	ค
สารบัญ	จ
สารบัญภาพ	ช
บทที่ 1. บทนำ	1
1.1 ความสำคัญและที่มาของการวิจัย	1
1.2 วัตถุประสงค์	3
1.3 ขอบเขตของงานวิจัย	3
1.4 ประโยชน์ที่คาดว่าจะได้รับ	3
บทที่ 2. การควบคุมระบบฟิชชีลอจิก	4
2.1 ระบบฟิชชีลอจิก	3
2.2 มอเตอร์ไฟฟ้ากระแสตรง	6
2.2.1 หลักการทำงานของมอเตอร์ไฟฟ้ากระแสตรง	6
2.2.2 ชนิดของมอเตอร์ไฟฟ้ากระแสตรง	6
2.2.3 ตัวแปรที่มีผลต่อความเร็วของมอเตอร์ไฟฟ้ากระแสตรง	8
2.2.4 การควบคุมความเร็วของมอเตอร์ไฟฟ้ากระแสตรง	9
2.2.5 การเปรียบเทียบคุณลักษณะของมอเตอร์	10
2.3 หลักการของวงจรถอปเปอร์	11
บทที่ 3. การออกแบบและสร้างตัวควบคุมฟิชชีลอจิก	17
3.1 การออกแบบระบบตัวควบคุมความเร็ว โดยอาศัยทฤษฎีฟิชชีลอจิก	17
3.2 การออกแบบโปรแกรมและฮาร์ดแวร์	26
3.2.1 ส่วนของโปรแกรม	26
3.2.2 ส่วนของฮาร์ดแวร์	28
บทที่ 4. การทดลอง	39
4.1 การทดลองวงจรถอปเปอร์	39
4.2 การทดลองตัวตรวจจับกระแส	40
4.3 การทดลองความเร็วควบคุมความเร็วของมอเตอร์	42

สารบัญ (ต่อ)

บทที่ 5. สรุปผล	46
บรรณานุกรม	47
ภาคผนวก	48



สารบัญภาพ

ภาพประกอบที่	หน้า
1.1 เครื่องวัดความคงตัวของน้ำยาง	2
2.1 แผนผังแสดงการทำงานของทฤษฎีฟิซซึลลจิก	5
2.2 โครงสร้างการทำงานของตัวควบคุมแบบฟิซซึลลจิกในการควบคุมความเร็วมอเตอร์	5
2.3 การต่อวงจรอาร์เมเจอร์และวงจรมอเตอร์ของ DC Machine	7
2.4 วงจรสมมูลของมอเตอร์ไฟฟ้ากระแสตรงแบบแยกฟิลด์กระตุ้น	8
2.5 การควบคุมความเร็วโดยการปรับแรงดันป้อนอาร์เมเจอร์และปรับแต่งค่ากระแสฟิลด์	10
2.6 แสดงการเปรียบเทียบคุณลักษณะของแรงบิด	11
2.7 วงจรชอปเปอร์พื้นฐานโหลด R	12
2.8 รูปคลื่นแรงดันอินพุตไฟตรง (VS) สัญญาณเปิด-ปิดสวิตช์	12
2.9 วงจรชอปเปอร์พื้นฐานโหลด RL	14
2.10 การทำงานในโหมด 1 และ 2 ของชอปเปอร์	14
2.11 รูปคลื่นกระแสโหลด (IO) และแรงดันคร่อมโหลดเมื่อโหลด RL	14
3.1 แสดงฟังก์ชันการเป็นสมาชิกฟิซซึลลจิก	19
3.2 แสดงฟังก์ชันการเป็นสมาชิกฟิซซึลลจิก 2	20
3.3 ฟังก์ชันการเป็นสมาชิกของเอาต์พุตฟิซซึลลจิกในระบบควบคุมความเร็ว	20
3.4 โฟลวชาร์ตการทำงานของไมโครคอนโทรลเลอร์ PIC16F877	27
3.5 วงจรแหล่งจ่ายไฟเลี้ยงของดิจิตอลคอนโทรลเลอร์	28
3.6 วงจรชุดไมโครคอนโทรลเลอร์ PIC 16F877	29
3.7 แสดงความสัมพันธ์ของวงจรถ้าเนตสัญญาณ PWM	29
3.8 แสดงลักษณะการต่อของไอซีเบอร์ IR2110	30
3.9 แสดงวงจรภายในของไอซีเบอร์ IR2110	31
3.10 การต่อไอซีเบอร์ IR2110	31
3.11 วงจรขับเคลื่อน	32
3.12 แสดงการต่อวงจรภาคกำลัง	34

สารบัญญภาพ (ต่อ)

	หน้า	
3.13	วงจรแปลงแรงดันเป็นไฟกระแสตรง จ่ายให้กับขดลวดฟิวส์	34
3.14	วงจรแปลงแรงดันเป็นไฟกระแสตรง จ่ายให้กับขดลวดอาร์เมเจอร์	35
3.15	วงจรรวมทั้งหมดของภาคกำลัง	35
3.16	แสดงการต่อวงจรรวมของฮาร์ดแวร์	36
3.17	ชุดมอเตอร์ปั่นน้ำยางข้น	37
3.18	ชุดควบคุมความเร็วของมอเตอร์กระแสตรง	38
3.19	ชุดควบคุมและชุดขับมอเตอร์กระแสตรง	39



บทที่ 1

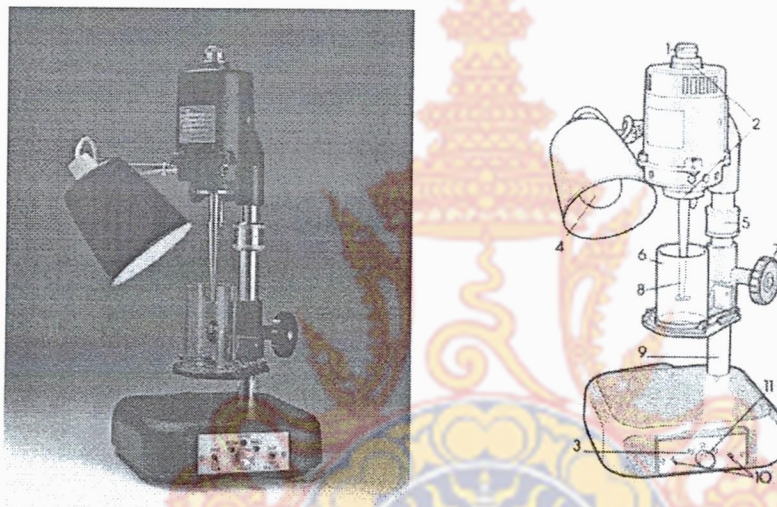
บทนำ

1.1 ความสำคัญและที่มาของการวิจัย

ประเทศไทยเป็นประเทศผู้ผลิตและส่งออกน้ำยางข้นอันดับหนึ่งของโลก และมีสัดส่วนการส่งออกเป็นลำดับ 3 รองจากยางแท่ง และยางแผ่นรมควันซึ่งมีการส่งออกสูงเป็นอันดับหนึ่งของประเทศ โดยมีมูลค่าการส่งออกสูงถึง 30,000 ล้านบาท หรือคิดเป็นร้อยละ 0.4 ของ GDP การผลิตน้ำยางข้นของประเทศไทยมีการขยายตัวเพิ่มสูงขึ้นอย่างรวดเร็วตลอดระยะเวลา 5 ปีที่ผ่านมา (2543-2547) จาก 0.29 ล้านตัน ในปี 2543 เป็น 0.50 x ล้านตัน ในปี 2547 หรือเพิ่มขึ้นถึง 72.41% เนื่องจากการขยายการผลิตของอุตสาหกรรมถุงมือยาง และอุตสาหกรรมผลิตยางอนามัย ที่เข้ามาใช้ประเทศไทยเป็นฐานการผลิตของโลก

น้ำยางข้น ผลิตขึ้นจากการนำน้ำยางสด จากส่วนที่มีปริมาณเนื้อยาง (Dry Rubber Content) เฉลี่ยประมาณ 35% สารละลายที่ไม่ใช่ยาง (Non-rubber solid) 5% และน้ำ (Watery) มาผ่านกระบวนการแปรรูปให้อยู่ในรูปของน้ำยางข้นที่มีเนื้อยางแห้งอย่างน้อย 60% โดยใช้วิธีการปั่นแยกด้วยเครื่องปั่นความเร็วสูงประมาณ 14000 รอบต่อนาที เพื่อแยกน้ำ และสารอื่นๆ ที่ละลายอยู่ ออกไปบางส่วน น้ำยางที่ได้จะเรียกว่า Centrifuged Latex ซึ่งมีน้ำยางข้น 60% กับหางน้ำยาง จากนั้นจะต้องทำการรักษาคุณภาพด้วยการเติมแอมโมเนีย เพื่อป้องกันมิให้น้ำยางจับตัว ซึ่งแบ่งได้เป็น 2 ชนิดหลักๆ ได้แก่ 1) น้ำยางข้นชนิดแอมโมเนียสูง (High ammonia: HA) จะใช้แอมโมเนีย 0.7% ต่อน้ำหนักยาง และ 2) น้ำยางข้นชนิดแอมโมเนียต่ำ (Low ammonia: LA) จะใช้แอมโมเนีย 0.2% ต่อน้ำหนักยาง และเติมสารละลายอื่นๆ ซึ่งทำให้ได้น้ำยางที่เหมาะสมสำหรับการเก็บไว้ใช้ประโยชน์ หรือเข้าสู่กระบวนการผลิตเพื่อทำผลิตภัณฑ์ที่มีคุณภาพตามความต้องการของผู้ใช้ น้ำยางข้นที่ผลิตในประเทศไทยจะต้องผลิตภายใต้มาตรฐานคุณภาพ ตามข้อกำหนดมาตรฐานน้ำยางข้นไทย (มอก. 980-2533) ซึ่งเป็นไปตามมาตรฐาน ISO: 1997 (E) ที่กำหนดมาตรฐานคุณภาพและวิธีการตรวจสอบคุณภาพของน้ำยางไว้อย่างชัดเจน ทั้งนี้ทางสมาคมน้ำยางข้นได้กำหนดมาตรฐานคุณภาพน้ำยางข้นที่มีคุณสมบัติที่สูงกว่ามาตรฐาน ISO เพื่อยกระดับมาตรฐานน้ำยางข้นของประเทศให้เป็นที่ยอมรับทั่วโลก ซึ่งได้รับการรับรองจากสถาบันวิจัยยางให้ใช้เป็นมาตรฐานสำหรับการซื้อขายในตลาดสินค้าเกษตรล่วงหน้าได้

การปฏิบัติการหาคุณสมบัติของน้ำยางชั้นนั้นจะดำเนินการนำตัวอย่างน้ำยางมาทดสอบในห้องปฏิบัติการ โดยการนำน้ำยางมาปั่นด้วยความเร็วสูงด้วยเครื่องวัดความคงตัวของน้ำยาง (Latex Stability Test) ที่ความเร็วรอบ 14000 รอบต่อนาที ซึ่งต้องรักษาระดับความเร็วให้คงที่เพื่อทำการแยกน้ำยางกับน้ำและสิ่งเจือปนต่างๆ ซึ่งเครื่องดังกล่าวนี้จะมีการใช้งานเป็นประจำซึ่งจะทำให้ชุดควบคุมความเร็วของเครื่องจะชำรุดได้บ่อย ซึ่งการซ่อมแซมจะทำให้ยากและหากจะเปลี่ยนชุดควบคุมความเร็วจะต้องนำเข้าจากต่างประเทศและมีราคาแพง จึงเป็นแนวทางในการดำเนินการวิจัยเพื่อสร้างบอร์ดอิเล็กทรอนิกส์ที่ทำงานด้วยระบบไมโครคอนโทรลเลอร์ทำการรักษาระดับความเร็วไปทดแทนบอร์ดอิเล็กทรอนิกส์เดิมที่ชำรุด



ภาพที่ 1.1 เครื่องวัดความคงตัวของน้ำยาง (Latex Stability Test)

การควบคุมความเร็วของเครื่องวัดความคงตัวของน้ำยางสามารถทำได้โดยใช้ทฤษฎีของพีชชีลอจิก ที่จะนำค่าของค่าความผิดพลาดระหว่างค่าที่กำหนดกับค่าที่วัดได้มาทำการปรับผลทางด้านเอาต์พุตเพื่อควบคุมความเร็วให้คงที่ตามที่กำหนด ซึ่งเป็นองค์ความรู้ที่ทางมหาวิทยาลัยมีอยู่แล้ว ดังนั้นงานวิจัยนี้จะเป็นประโยชน์ในการทดแทนการนำเข้าชุดควบคุมความเร็วของเครื่องวัดความคงตัวของน้ำยางจากต่างประเทศและเป็นการนำองค์ความรู้ไปประยุกต์ใช้ให้เกิดประโยชน์กับสังคม

1.2 วัตถุประสงค์

1.2.1 พัฒนาค้นแบบบอร์ดอิเล็กทรอนิกส์รักษาระดับความเร็วของเครื่องวัดความคงตัวของน้ำยางด้วยพีซีลोजิกที่ระดับความเร็ว 14000 รอบต่อนาที

1.2.2 เป็นอะไหล่ทดแทนการนำเข้าบอร์ดอิเล็กทรอนิกส์ของเครื่องวัดความคงตัวของน้ำยาง

1.2.3 ได้องค์ความรู้เกี่ยวกับการรักษาระดับความเร็วของเครื่องวัดความคงตัวของน้ำยางด้วยพีซีลोजิก

1.3 ขอบเขตงานวิจัย

1.3.1 สามารถรักษาระดับความเร็วของเครื่องวัดความคงตัวของน้ำยางให้คงที่ที่ความเร็ว 14000 รอบต่อนาที

1.3.2 สามารถใส่แทนบอร์ดอิเล็กทรอนิกส์เดิมของชุดควบคุมความเร็วของเครื่องวัดความคงตัวของน้ำยาง

1.3.3 สามารถจับเวลาการทำงานของเครื่อง

1.4 ประโยชน์ที่คาดว่าจะได้รับ

1.4.1 ได้ต้นแบบระบบเครื่องควบคุมความเร็วของเครื่องวัดความคงตัวของน้ำยาง

1.4.2 ได้องค์ความรู้เกี่ยวกับการควบคุมเครื่องกำเนิดไฟฟ้าด้วยระบบพีซีลोजิก

1.4.3 สามารถลดการนำเข้าชุดควบคุมความเร็วของเครื่องวัดความคงตัวของน้ำยาง (Latex Stability Test) จากต่างประเทศ

1.4.4 สามารถนำเสนอแก่หน่วยงานภายนอกที่สนใจและพัฒนาต่อในเชิงพาณิชย์ได้

บทที่ 2

การควบคุมระบบฟัซซี่ลอจิก

2.1 ระบบฟัซซี่ลอจิก

ระบบควบคุมแบบฟัซซี่ลอจิก (Fuzzy Logic Control System) คือ ระบบกฎพื้นฐานอันเป็นกฎฟัซซี่ (Fuzzy rule) ที่สามารถแทนส่วนที่ใช้แทนการตัดสินใจทางกล (Human mechanical) เพื่อให้ได้คำตอบที่แน่นอนของระบบ จุดประสงค์หลักของระบบควบคุมแบบฟัซซี่ลอจิกคือ ใช้แทนส่วนระบบปฏิบัติการของมนุษย์ (Human operator) ด้วยกฎพื้นฐานของฟัซซี่ (Fuzzy Rule-Based System) โดยทั่วไประบบควบคุมฟัซซี่ลอจิกจะมีส่วนประกอบ 3 ส่วน คือ

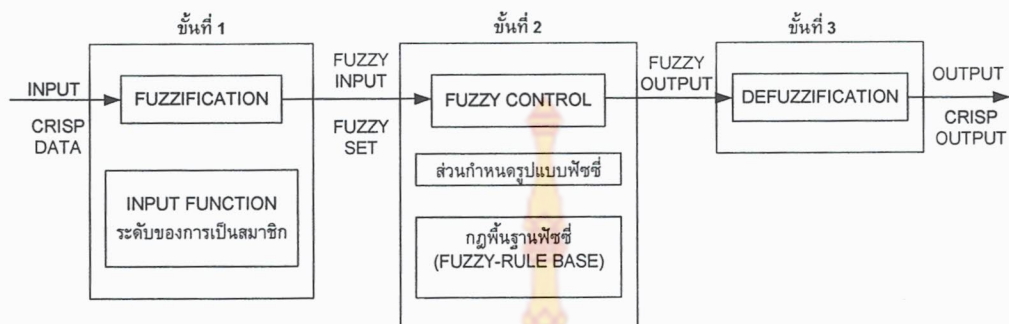
ส่วนที่ 1 เรียกว่า ฟัซซิฟิเคชัน (Fuzzification) ทำหน้าที่แปลงข้อมูลที่ได้รับเข้ามาจากระบบควบคุมซึ่งเป็นค่าข้อมูลอินพุต (มักจะเรียกว่า ข้อมูลคริสป : Crisp Data) ให้เป็นอินพุตของระบบควบคุมแบบฟัซซี่ (Fuzzy input) คือมีความหมายเดียวกับระบบฟัซซี่ ซึ่งจะอยู่ในรูปแบบของค่า

ส่วนที่ 2 ในฟัซซี่เซตเมื่อข้อมูลอินพุตผ่านส่วนที่ 1 แล้วจะเข้าสู่ส่วนของการประมวลผล ซึ่งจะมีการตีความและวิเคราะห์อินพุตตามเงื่อนไขที่ได้มีการกำหนดไว้ หรือเข้าสู่กฎพื้นฐานของฟัซซี่ (Fuzzy - Rule Based) ระบบควบคุมแบบฟัซซี่จะเป็นตามแบบของการป้อนกลับด้วยเหตุนี้การป้อนกลับในระบบฟัซซี่จะเกิดขึ้นในส่วนกฎพื้นฐานนี้เอง

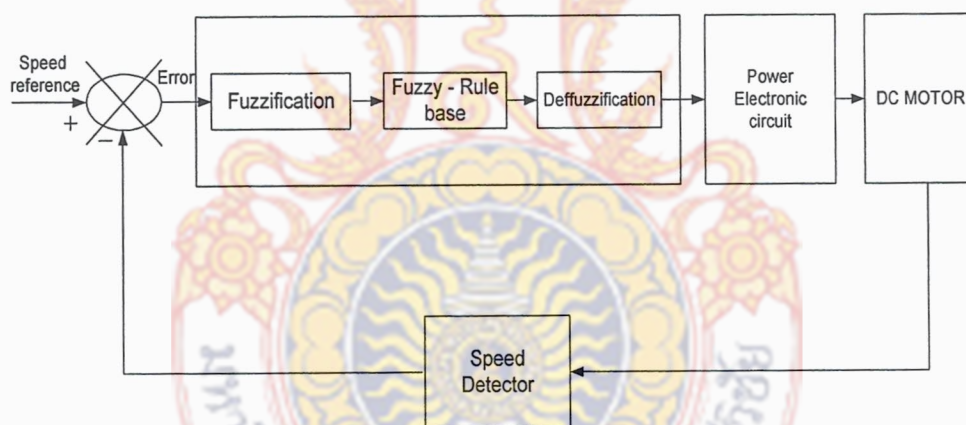
ส่วนที่ 3 เป็นส่วนสุดท้าย เรียกว่า ดีฟัซซิฟิเคชัน ซึ่งเป็นขั้นตอนที่ทำการเปรียบเทียบเอาต์พุตฟัซซี่ (Fuzzy output) ทั้งหมดเข้ากับผลลัพธ์รวม ปัจจุบันนิยมใช้วิธีหาศูนย์กลางพื้นที่ (Center Of Area : COA) เช่นวิธีการหาจุดเซนทรอยด์ (Centroid Center of Gravity) เพราะสามารถถ่วงน้ำหนักของเอาต์พุตฟัซซี่ของแต่ละกฎได้อย่างทั่วถึง ได้ค่าเอาต์พุตอันเป็นค่าโดยประมาณของระบบการทำงานจริง

สิ่งที่ได้กล่าวมาข้างต้น คือ ทฤษฎีการควบคุมแบบฟัซซี่ลอจิก การสร้างความสัมพันธ์แบบฟัซซี่ (Fuzzy relations) เป็นไปในลักษณะ IF.....THEN ประกอบด้วยส่วนประกอบสำคัญ 2 ส่วน คือ ส่วนที่เกิดขึ้นก่อน (Antecedent, If part) และผลที่ตามมา (Consequent, then part) พร้อมทั้งมีฟังก์ชันการเป็นสมาชิก (Membership Function) ของแต่ละส่วน ผลที่ตามมาของแต่ละกฎจะถูกรวมกันด้วยวิธีการทางวิทยาศาสตร์ เช่น ยูเนียน และการหาจุดเซนทรอยด์ เพื่อให้ได้ค่าเอาต์พุตจริงเพียงค่าเดียว (ค่าตรรกศาสตร์ในช่วงตั้งแต่ 0 - 1) ภายในกรอบการทำงานของระบบ

ผู้เชี่ยวชาญจะมีการสร้างกฎแต่ละข้อ ซึ่งได้มาจากความรู้และประสบการณ์ของการปฏิบัติทางมนุษย์



ภาพที่ 2.1 แผนผังแสดงการทำงานของทฤษฎีฟัซซี่ลอจิก



ภาพที่ 2.2 โครงสร้างการทำงานของตัวควบคุมแบบฟัซซี่ลอจิกในการควบคุมความเร็วมอเตอร์

จากภาพที่ 2.2 มีการรับค่าความเร็วรอบของมอเตอร์กระแสตรงในส่วนของการป้อนกลับจากนั้นก็เข้าสู่ทฤษฎีการควบคุมแบบฟัซซี่ลอจิก ซึ่งแบ่งเป็น 3 ส่วน ดังนี้

1. การฟัซซี่ฟิเคชัน (Fuzzification) หาอินพุตของฟัซซี่ คือค่าความผิดพลาด (Error) โดยหาได้จากความเร็วรอบของมอเตอร์ที่ผลิตขึ้นมากับค่าความเร็วรอบอ้างอิง

2. การเข้าสู่กฎพื้นฐานของฟัซซี่ เป็นหัวใจหลักของการควบคุมเปรียบเสมือนการตัดสินใจของมนุษย์ ซึ่งจะมีการตีความและวิเคราะห์อินพุตตามเงื่อนไขที่ได้มีการกำหนดไว้

3. การดีฟัซซิฟิเคชัน (Defuzzification) หาค่าเอาต์พุตจริงเนื่องจากในแต่ละอินพุตจะได้ค่าเอาต์พุตฟัซซี่ออกมาหลายค่า เราจึงต้องหาค่าแม่นยำตรงเพียงค่าเดียวเพื่อให้ได้ค่าเอาต์พุตที่แท้จริง ของตัวควบคุมฟัซซี่ลอจิก ซึ่งจะใช้กรรมวิธีการหาค่าศูนย์กลางเพื่อหาค่าเซนทรอยด์ เมื่อได้ค่าเอาต์พุตจริงแล้ว นำค่าที่ได้ไปพิจารณาหาค่าสัญญาณควบคุมที่เหมาะสมเพื่อให้ความเร็วรอบมอเตอร์เข้าสู่ความเร็วรอบอ้างอิง

จากหลักการทฤษฎีการควบคุมแบบฟัซซี่ลอจิกที่ได้กล่าวมาข้างต้น ได้นำมาประยุกต์ใช้ในโครงการ โดยทำการควบคุมความเร็วของมอเตอร์ชนิดยูนิเวอร์ซัล ให้ได้ที่มีความเร็ว 14000 รอบต่อนาที

2.2 มอเตอร์ไฟฟ้ากระแสตรง

2.2.1 หลักการทำงานของมอเตอร์ไฟฟ้ากระแสตรง

เครื่องจักรไฟฟ้ากระแสตรง มีหลักการทำงาน คือ เมื่อมีกระแสไฟฟ้าไหลผ่านตัวนำ และสนามแม่เหล็กไฟฟ้า จะเป็นไปตามกฎการเหนี่ยวนำเนื่องจากอำนาจแม่เหล็กตามหลักของ “ฟาราเดย์” ปฏิกริยาของมอเตอร์เป็นไปตามกฎของ “แอมแปร์” แรงที่เกิดขึ้นในลวดตัวนำที่มีกระแสไหล และวางอยู่ในสนามแม่เหล็กทิศทางของแรงที่เกิดขึ้นจะมีทิศเป็นไปตามกฎมือซ้าย และสามารถหาค่าแรงได้จาก

$$F = Bli \quad (2-1)$$

โดยที่

F คือ แรงที่เกิดขึ้นในขดลวดตัวนำ

B คือ ความหนาแน่นของฟลักแม่เหล็ก

l คือ ความยาวของขดลวดตัวนำ

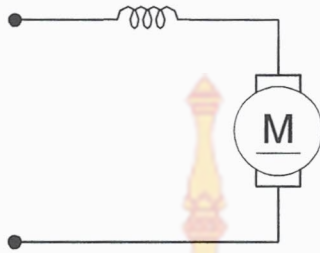
i คือ กระแสไฟฟ้าที่ไหลผ่านขดลวดตัวนำ

2.2.2 ชนิดของมอเตอร์ไฟฟ้ากระแสตรง

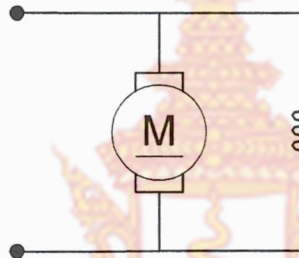
ในการเลือกใช้งานนั้นเราต้องเลือกใช้มอเตอร์กระแสตรงให้เหมาะสมกับงานที่จะทำ ซึ่งมอเตอร์กระแสตรงมีหลายชนิด เช่น

1. มอเตอร์กระแสตรงชนิดอนุกรม (Series motor)

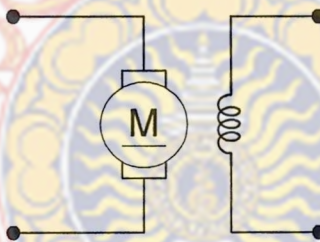
2. มอเตอร์กระแสตรงชนิดขนาน (Shunt motor)
3. มอเตอร์กระแสตรงชนิดกระตุ้นแยก (Separately excited)
4. มอเตอร์กระแสตรงชนิดผสม (Compound motor)



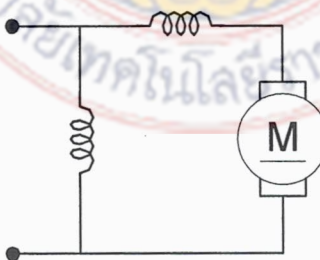
(ก) มอเตอร์กระแสตรงชนิดอนุกรม



(ข) มอเตอร์กระแสตรงชนิดขนาน



(ค) มอเตอร์กระแสตรงชนิดกระตุ้นแยก



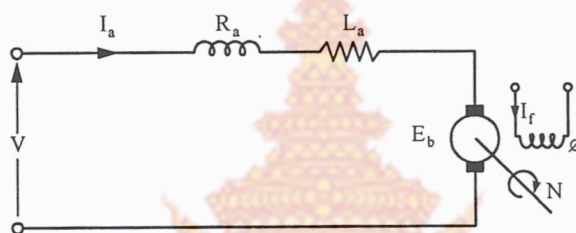
(ง) มอเตอร์กระแสตรงชนิดผสม

ภาพที่ 2.3 การต่อวงจรอาร์เมเจอร์และวงจรฟิลด์ของ DC Machine

2.2.3 ตัวแปรที่มีผลต่อความเร็วของมอเตอร์ไฟฟ้ากระแสตรง

ในการที่จะศึกษาถึงวิธีการควบคุมความเร็วของมอเตอร์ไฟฟ้ากระแสตรง (DC Motor) นั้นก่อนอื่นเราจะต้องทราบเสียก่อนว่า ตัวแปรต่าง ๆ ที่จะมีอิทธิพลต่อความเร็วของมอเตอร์ไฟฟ้ากระแสตรงนั้นมีอะไรบ้าง และตัวแปรแต่ละตัวที่มีอิทธิพลต่อความเร็วของมอเตอร์ไฟฟ้ากระแสตรงที่เราทราบนั้น จะมีข้อดี ข้อเสียอย่างไรบ้างในการที่จะนำไปควบคุมความเร็วของมอเตอร์ไฟฟ้ากระแสตรง

ในการที่จะหาตัวแปรที่มีผลต่อความเร็วของมอเตอร์ไฟฟ้ากระแสตรงได้นั้น เราจะเริ่มพิจารณาจากวงจรสมมูลของมอเตอร์ไฟฟ้ากระแสตรงแบบฟีดแบ็คแยกกระตุ้น (Separately excited DC Motor) ดังภาพที่ 4



ภาพที่ 2.4 วงจรสมมูลของมอเตอร์ไฟฟ้ากระแสตรงแบบแยกฟีดแบ็คกระตุ้น

จากภาพที่ 2.4 พิจารณาที่ Steady state เราจะได้สมการ

$$V = E_b + I_a R_a \quad (2-2)$$

จากสมการของแรงเคลื่อนไฟฟ้าเหนี่ยวนำจะได้ว่า

$$E_b = K \cdot \omega \cdot N \quad (2-3)$$

ดังนั้นจะเห็นได้ว่า

$$V = K \omega N + I_a R_a \quad (2-4)$$

โดยที่

- V = แรงเคลื่อนไฟฟ้าที่มอเตอร์ได้รับ (โวลต์)
- E_b = แรงเคลื่อนไฟฟ้าต้านกลับ (Back emf) (โวลต์)
- = แรงเคลื่อนไฟฟ้าที่เหนี่ยวนำขึ้นในอาร์เมเจอร์ (โวลต์)
- I_a = กระแสไฟฟ้าที่ไหลในอาร์เมเจอร์ (แอมแปร์)
- R_a = ค่าความต้านทานของวงจรรีอาร์เมเจอร์ (โอห์ม)
- K = ค่าคงที่

$$Z = \frac{2P}{60A}$$

Z = จำนวนตัวนำทั้งหมดในอาร์เมเจอร์ที่ต่ออนุกรมกัน
 P = จำนวนขั้วแม่เหล็กของมอเตอร์
 a = จำนวนแผง (Path) ที่ขนานกัน
 \emptyset = จำนวนเส้นแรงแม่เหล็กต่อขั้วของมอเตอร์ (เวเบอร์)
 N = ความเร็วรอบของมอเตอร์ (รอบ/นาที)

จากสมการ (2-4) จะได้ว่า

$$K\phi \cdot N = V - I_a \cdot R_a \quad (2-5)$$

เราสามารถหาความเร็วของมอเตอร์ได้ดังนี้

$$N = \frac{V - I_a R_a}{K\phi} \quad (2-6)$$

จากสมการ (2-6) จะเห็นได้ว่าการเปลี่ยนแปลงความเร็วของมอเตอร์ขึ้นอยู่กับ

1. ค่าความต้านทานในวงจรอาร์เมเจอร์ (R_a) การควบคุมแบบนี้จะเรียกว่าการควบคุมโดยใช้ความต้านทาน (Rheostatic control)
 2. จำนวนเส้นแรงแม่เหล็กต่อขั้ว (\emptyset) ซึ่งเป็นสัดส่วนผกผันกับความเร็ว ซึ่งการควบคุมแบบนี้จะเรียกว่า การควบคุมโดยใช้เส้นแรงแม่เหล็ก (Flux control)
 3. การเคลื่อนที่ไฟฟ้าที่จ่ายให้กับมอเตอร์ (V) ซึ่งเป็นสัดส่วนโดยตรงกับความเร็วการควบคุมโดยวิธีนี้เรียกว่า การควบคุมแรงเคลื่อนไฟฟ้าที่จ่ายให้กับมอเตอร์ (Voltage Control)
- การควบคุมความเร็วทั้ง 3 วิธี ข้างต้นสามารถนำไปใช้ควบคุมความเร็วมอเตอร์ไฟฟ้ากระแสตรงทั้งแบบขนาน อนุกรม และผสมได้

2.2.4 การควบคุมความเร็วของมอเตอร์ไฟฟ้ากระแสตรง

การควบคุมความเร็วในการทำงานนั้นจะแบ่งออกเป็น 2 ช่วง คือ

1. Constant Torque Mode คือ การควบคุมความเร็วที่มีค่าต่ำกว่า “ ความเร็วฐาน ” (Base Speed) โดยแรงหมุนจะมีค่าคงที่ แต่แรงม้าจะเปลี่ยนแปลงไปตามความเร็ว
2. Constant Horse Mode คือ การควบคุมความเร็วที่มีค่าสูงกว่า “ ความเร็วฐาน ” (Base Speed) โดยแรงม้าจะมีค่าคงที่แต่แรงหมุนจะมีค่าเปลี่ยนแปลงไปตามความเร็ว

การควบคุมความเร็วมอเตอร์กระแสตรงแบบกระตุ้นแยก มีการกระตุ้นวงจรถาวรฟิวด์อย่างถาวร (Permanent Magnet) ซึ่งเป็นมอเตอร์ที่ง่ายต่อการควบคุมที่สุด เพราะวงจรถาวรแม่เหล็กแยกจากวงจรรีเมเจอร์ โดยการควบคุมความเร็วของมอเตอร์ในช่วงความเร็วต่ำกว่าความเร็วมาตรฐาน (Constant Torque Mode) ทำได้โดยการปรับเปลี่ยนแรงดันขั้วเนื่องจากแรงดันขั้วจะแปรผันโดยตรงกับความเร็วรอบ โดยที่ฟลักซ์ที่ขดลวดสนามแม่เหล็กมีค่าคงที่ จะให้ความสัมพันธ์ระหว่าง E_a กับ ω ดังต่อไปนี้

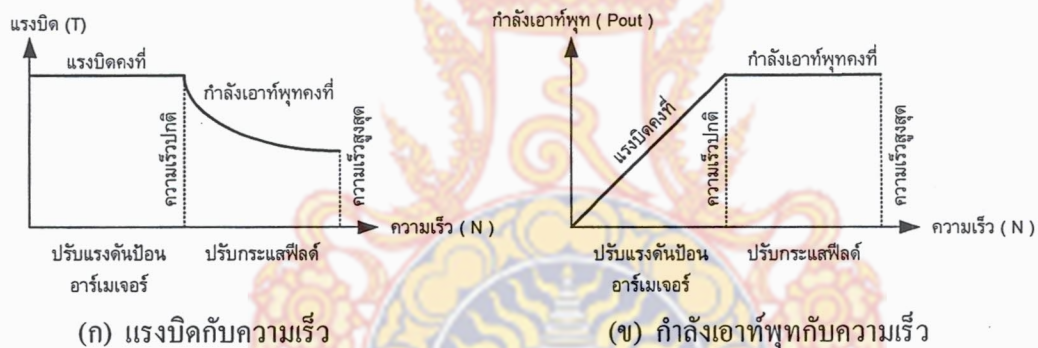
$$E_a = K_b \omega(t) \quad (2-7)$$

โดยที่ E_a คือ แรงดันเหนี่ยวนำด้านกลับของมอเตอร์ (Back – emf)

K_b คือ ค่าคงที่

$\omega(t)$ คือ ความเร็วเชิงมุมของมอเตอร์

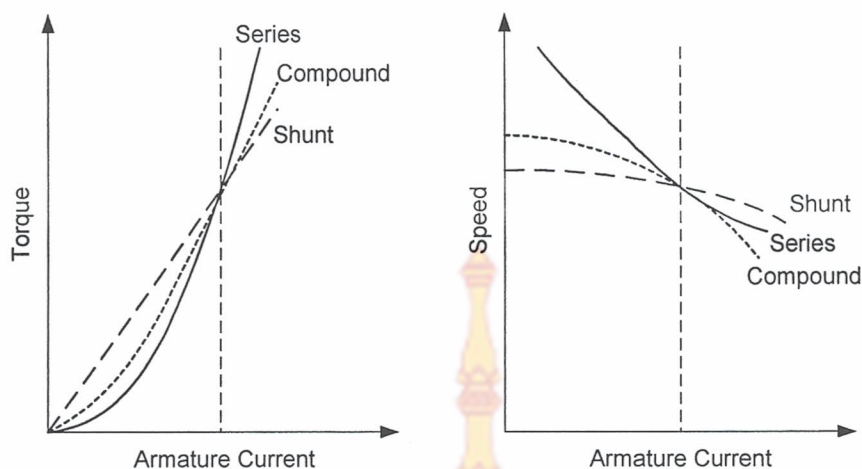
เครื่องกลไฟฟ้ากระแสตรงได้มีการผลิตมาก่อนศตวรรษ ไฟฟ้ากระแสตรงก็ยังเป็นที่นิยมใช้กันอยู่ทุกวันนี้ เพราะคุณสมบัติของมอเตอร์กระแสตรงมีมากในการนำไปใช้งาน ที่มีภาระที่ต้องการเปลี่ยนแปลงความเร็วค่าต่าง ๆ หลายค่า



ภาพที่ 2.5 การควบคุมความเร็วโดยการปรับแรงดันขั้วอาร์เมเจอร์และปรับแต่งค่ากระแสฟิวด์

2.2.5 การเปรียบเทียบคุณลักษณะของมอเตอร์

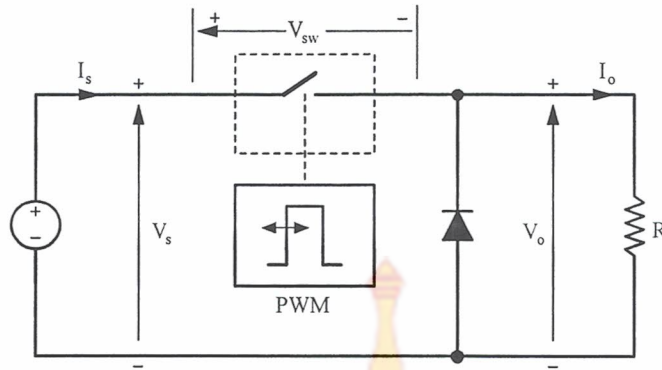
ในการเลือกใช้มอเตอร์กับงานแต่ละประเภท เราจะต้องรู้คุณลักษณะของมอเตอร์ชนิดต่างๆ การเปรียบเทียบคุณลักษณะของมอเตอร์ในภาพที่ 2.6 จึงจะมีประโยชน์อย่างมากในการทำให้เราทราบว่ามอเตอร์ชนิดใดเหมาะกับงานของเรามากที่สุด



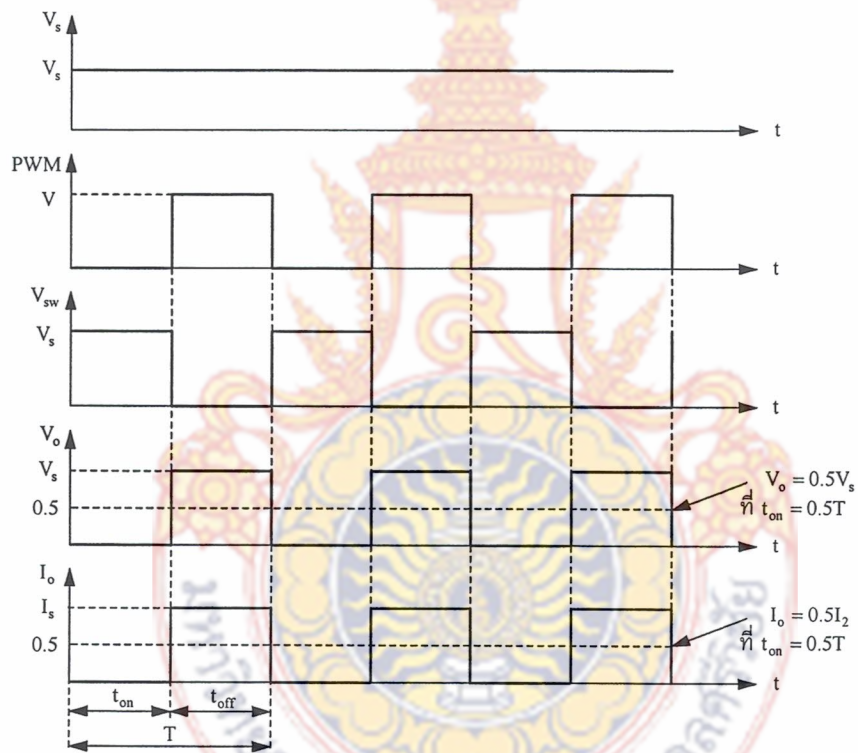
ภาพที่ 2.6 แสดงการเปรียบเทียบคุณลักษณะของแรงบิด (Torque Characteristics) และคุณลักษณะของความเร็วรอบ (Speed Characteristics) ของมอเตอร์ไฟฟ้า

2.3 หลักการของวงจรถอปเปอร์

วงจรถอปเปอร์หรือวงจรถอปเปอร์กระแสตรง (DC Chopper) อาจเรียกอีกชื่อหนึ่งว่า วงจรสับไฟฟ้ากระแสตรง เป็นวงจรที่ทำหน้าที่แปลงผันพลังงานไฟฟ้าจากแหล่งจ่ายกระแสตรงที่มีแรงดันคงที่ให้เป็นพลังงานไฟฟ้ากระแสตรงที่ปรับค่าได้ โดยใช้สวิตซ์อิเล็กทรอนิกส์กำลังเป็นตัวเปิด-ปิด สวิตซ์อิเล็กทรอนิกส์กำลังที่นิยมใช้ในวงจรถอปเปอร์ เช่น ทรานซิสเตอร์กำลัง มอสเฟตกำลัง ไอจีบีที และ จีทีโอ (ไอจีบีที ย่อมาจาก IGBT หรือ Insulated Gate Bipolar Transistor) เป็นต้น เนื่องจากสวิตซ์อิเล็กทรอนิกส์กำลังเหล่านี้ เปิด-ปิดได้ด้วยสัญญาณควบคุมที่ป้อนเข้าทางอินพุต สัญญาณควบคุมนี้คือ สัญญาณมอดูเลตความกว้างพัลส์ที่สามารถปรับค่า ton ในขณะที่ T คงที่ได้ ลักษณะของวงจรถอปเปอร์แสดงในภาพที่ 2.7 และสัญญาณควบคุม การเปิด-ปิดสวิตซ์ของชอปเปอร์เทียบกับแรงดันคร่อมโหลด และแรงดันคร่อมสวิตซ์แสดงในภาพที่ 2.8



ภาพที่ 2.7 วงจรชอปเปอร์พื้นฐาน โหลด R



ภาพที่ 2.8 รูปคลื่นแรงดันอินพุตไฟตรง (VS) สัญญาณเปิด-ปิดสวิตช์ (PWM) แรงดันคร่อมสวิตช์ (VSW) แรงดันเอาต์พุต (VO) และกระแสเอาต์พุต (IO) ของวงจรชอปเปอร์ที่มีโหลด R

เมื่อกำหนดให้ $D =$ วัฏจักรหน้าที่การสวิตช์ (Duty cycle,%)

$$D = \frac{t_{on}}{T} \quad \text{หรือ} \quad = \frac{t_{on}}{T} \times 100\%$$

ดังนั้น

$$V_o = DV_s$$

จากภาพที่ 8 เมื่อกำหนดให้

$$t_{on} = \frac{T}{2} = 0.5T$$

$$D = \frac{0.5T}{T} = 0.5 = 50\%$$

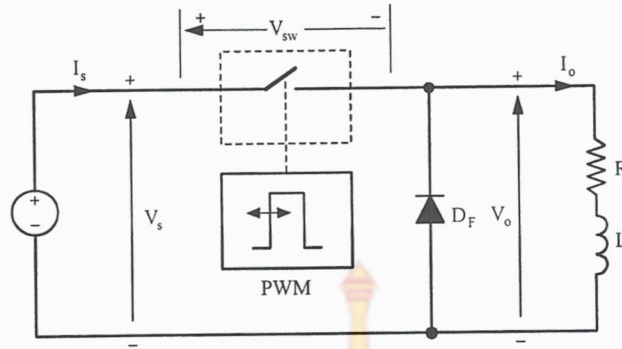
แทนค่า

$$V_o = 0.5V_s$$

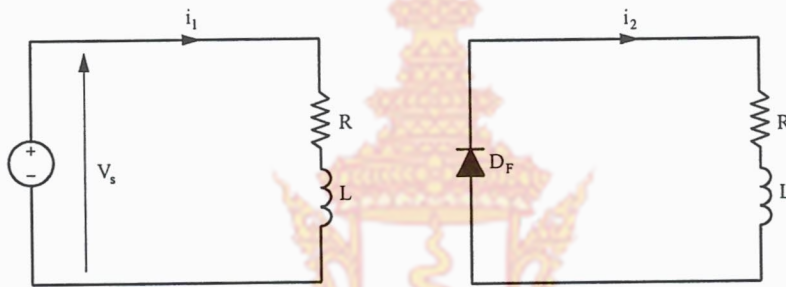
ในการทำงานเดียวกัน

$$I_o = DI_s$$

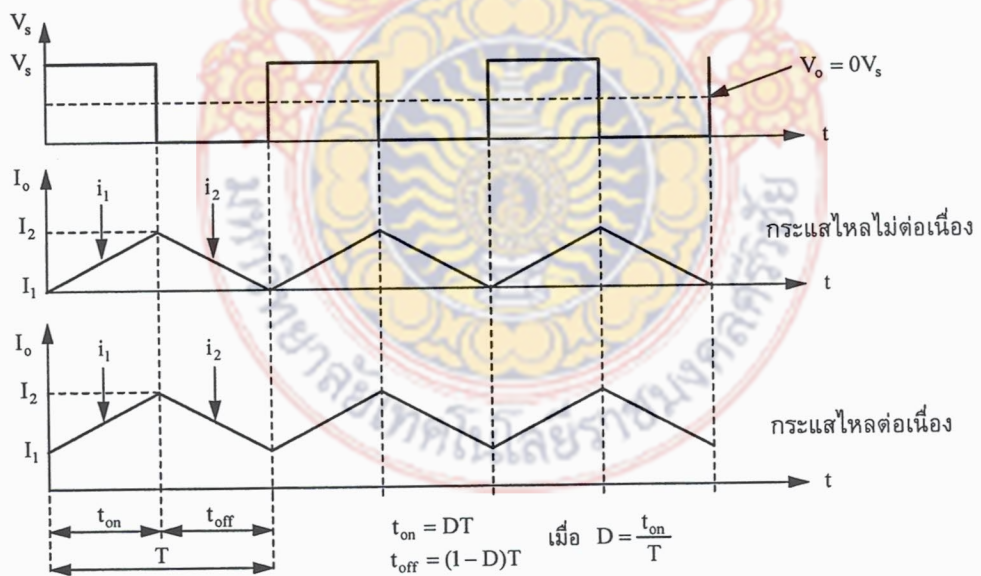
จะเห็นว่าจากสมการแรงดันเอาต์พุต (V_o) จะแปรค่าไปตามค่าวัฏจักรหน้าที่ (D) และถ้า $D = 0.5T$ จะได้ค่า $V_o = 0.5V_s$ ถ้าค่า D สูงขึ้นค่า V_o จะสูงขึ้นด้วย ในทำงานเดียวกัน ถ้า $t_{on} = T$ ค่า $V_o = V_s$ และเมื่อ $t_{on} = 0$ ค่า $V_o = 0$ หลักการของวงจรชอปเปอร์คือควบคุมการเปิด-ปิดสวิตช์อิเล็กทรอนิกส์กำลัง เพื่อให้สามารถปรับค่าแรงดันเอาต์พุตคร่อมโหลดได้ วงจรชอปเปอร์ลักษณะนี้เรียกว่า Step-down chopper เพราะว่าแรงดัน V_o จะมีค่าสูงสุดไม่เกินค่าแรงดันแหล่งจ่ายอินพุต เมื่อวงจรชอปเปอร์พื้นฐานมีโหลดเป็น RL ดังภาพที่ 9 วงจรชอปเปอร์จะทำงานได้ 2 โหมดคือ โหมด 1 เมื่อสวิตช์ปิด (Closed) กระแส i_1 จะไหลจากแหล่งจ่าย V_s เข้าสู่โหลด โดยผ่านสวิตช์อิเล็กทรอนิกส์กำลัง ที่เวลา $t = DT$ และ โหมด 2 เมื่อสวิตช์เปิด (Open) กระแส i_2 จะไหลจากโหลดผ่านไดโอดฟูลวีล (DF) ที่เวลา $t = (1-D)T$ ซึ่งช่วงเวลานี้กระแสแหล่งจ่ายจะไม่ไหลผ่านโหลดรูปคลื่นกระแส i_1 และ i_2 แสดงในภาพที่ 10 ซึ่งเกิดขึ้นได้ 2 กรณีคือ เมื่อกระแสโหลดไหลต่อเนื่อง เมื่อค่า D มีค่าน้อยๆ และกระแสไหลไม่ต่อเนื่อง เมื่อค่า D มีค่ามากๆ



ภาพที่ 2.9 วงจรขอปเปอร์พื้นฐานโหลด RL



ภาพที่ 2.10 การทำงานในโหมด 1 และ 2 ของขอปเปอร์



ภาพที่ 2.11 รูปคลื่นกระแสโหลด (IO) และแรงดันคร่อมโหลดเมื่อโหลด RL

$$V_s = Ri_1 + L \frac{di_1}{dt} [i_1(t=0) = I_1] \quad (2-8)$$

$$i_1(t) = I_1 e^{-tR/L} + \frac{V_s}{R} (1 - e^{-tR/L}) \quad (2-9)$$

การทำงานในโหมดนี้ต้องมีค่า t ในย่าน $0 < t < t_{on} (= DT)$ และเมื่อสิ้นสุดการทำงานในโหมด 1 กระแส $i_1(t_{on}) = I_2$

$$\text{การทำงานในโหมด 2} \quad 0 = Ri_2 + L \frac{di_2}{dt} [i_2(t=0) = I_2] \quad (2-10)$$

$$\text{ดังนั้น} \quad i_2(t) = I_2 e^{-tR/L} - \frac{1}{R} (1 - e^{-tR/L}) \quad (2-11)$$

การทำงานในโหมดนี้ต้องมีค่า t ในย่าน $0 < t < t_{off} = ((1-D)T)$ และเมื่อสิ้นสุดการทำงานในโหมด 2 กระแส $i_2(t_{off}) = I_1$ (ในสภาวะคงตัว) จากสมการที่ (2-9) และสมการ $i_1(t_{on}) = I_2$ หาค่า I_2 ได้ว่า

$$I_2 = I_1 e^{-TR/L} + \frac{V_s}{R} (1 - e^{-DTR/L}) \quad (2-12)$$

ในทำนองเดียวกัน จากสมการที่ (2-11) และสมการ $i_2(t_{off}) = I_1$ หาค่า I_1 ได้ว่า

$$I_1 = I_2 e^{-(1-D)TR/L} - \frac{1}{R} (1 - e^{-(1-D)TR/L}) \quad (2-13)$$

สมการค่ากระแสพลิวอดถึงยอด (Ripple current peak to peak)

$$\Delta I = I_2 - I_1$$

แทนค่าสมการที่ (2-12) และ (2-13)

$$\Delta I = \frac{V_s}{R} \left[\frac{1 - e^{-DTR/L} + e^{-TR/L} - e^{-(1-D)TR/L}}{1 - e^{-TR/L}} \right] \quad (2-14)$$

กรณีที่ค่า ΔI มีค่าสูงสุด นั่นคือ $\frac{d(\Delta I)}{dD} = 0$

$$\text{ดังนั้นค่า} \quad e^{-DTR/L} - e^{-(1-D)TR/L} = 0$$

$$\text{หรือ} \quad -D = -(1-D)$$

หรือ $D = 0.5$

นั่นหมายความว่าค่า ΔI สูงสุดเมื่อ $D = 0.5$ ดังนั้น

$$\Delta I_{\max} = \frac{V_s}{R} \tanh \frac{R}{4fL} \quad (2-15)$$

เมื่อ $4fL \gg R$ ค่าของ $\tanh \theta \approx \theta$ ค่าโดยประมาณของ ΔI_{\max} คือ

$$\Delta I_{\max} = \frac{R_s}{4fL} \quad (2-16)$$

หมายเหตุ สมการที่ (2-8) ถึง (2-16) เป็นจริงเมื่อคลื่นกระแส I_o ของวงจรชอปเปอร์ไหลอย่างต่อเนื่องเท่านั้น หรือกระแสไหลต่อเนื่องเมื่อ $L/R \gg T$ หรือ $Lf \gg R$ กรณีกระแส I_o ไหลไม่ต่อเนื่อง นั่นคือ $I_1 = 0$ ค่า $i_1(t)$ หาได้จากสมการที่ (2-9) รูปสมการจะเปลี่ยนเป็น

$$i_1(t) = \frac{V_s}{R} (1 - e^{-tR/L}) \quad (2-17)$$

สมการคำนวณค่ากระแสเฉลี่ยที่โหลด $I_{O(\text{avg})}$ เมื่อพิจารณาจากรูปคลื่นกระแส I_o ในภาพที่ 2.11 จะได้ว่า

$$I_{O(\text{avg})} = (I_2 + I_1)/2 \quad (2-18)$$

และกระแสเฉลี่ยจากแหล่งไฟฟ้า หาได้จากสมการ (2-19)

$$I_{S(\text{avg})} = DI_{O(\text{avg})} \quad (2-19)$$

บทที่ 3

การออกแบบและการสร้างตัวควบคุมพีชชีลอจิก

3.1 การออกแบบระบบตัวควบคุมความเร็ว โดยอาศัยทฤษฎีพีชชีลอจิก

ขั้นตอนที่ 1 การพีชชีพีเคชั่นของอินพุต เป็นการหาอินพุตของระบบ หาช่วงพีชชีของอินพุต และทำการสร้างระดับความเป็นสมาชิกที่เป็นไปได้

ในที่นี่จะกำหนดให้มีอินพุตของระบบ 2 ตัว คือ

1. ค่าความผิดพลาด (Error , E) ซึ่งหาได้จากความแตกต่างระหว่างค่าความเร็วรอบจากมอเตอร์ไฟฟ้ากระแสตรง กับความเร็วรอบอ้างอิง

$$\text{Error}(E) = N_{ref}(\text{Reference Speed}) - N_p(\text{Process Speed})$$

2. ค่าการเปลี่ยนแปลงค่าความผิดพลาด (Error Change , E_{ch}) ซึ่งหาได้จากความแตกต่างระหว่างค่าความผิดพลาดล่าสุด (Late Error) กับค่าความผิดพลาดก่อนหน้านั้น (Previous Error)

$$\text{Error Change}(E_{ch}) = E_l(\text{Late Error}) - E_p(\text{Previous Error})$$

โดยตัวแปรทางอินพุตทั้ง 2 ตัวนี้ต้องนำมาผ่านขั้นตอนของการพีชชีพีเคชั่นในรูปแบบของพีชชีของอินพุต โดยการเปรียบเทียบค่าของตัวแปรทางอินพุตกับตารางที่ 3.1 ซึ่งเป็นค่าควอนไทซ์(Quantized) ที่กำหนดขึ้น ทำให้สามารถจำนวนของอินพุตที่จะนำมาทำพีชชีพีเคชั่นได้

ตารางที่ 3.1 ค่าควอนไทซ์ของ E และ E_{ch} ของระบบควบคุมความเร็ว

Error Speed (E,rpm)	Error Change Speed (Ech,rpm)	Quantized Value
$E \geq 45$	$E \geq 45$	5
$35 \leq E < 45$	$35 \leq E < 45$	4
$25 \leq E < 35$	$25 \leq E < 35$	3
$15 \leq E < 25$	$15 \leq E < 25$	2
$5 \leq E < 15$	$5 \leq E < 15$	1
$-5 \leq E < 5$	$-5 \leq E < 5$	0
$-15 \leq E < -5$	$-15 \leq E < -5$	-1
$-25 \leq E < -15$	$-25 \leq E < -15$	-2
$-35 \leq E < -25$	$-35 \leq E < -25$	-3
$-45 \leq E < -35$	$-45 \leq E < -35$	-4
$E \leq -45$	$E \leq -45$	-5

ขั้นตอนที่ 2 การกำหนดตัวแปรเอาต์พุตฟัซซี่ การสร้างระดับการเป็นสมาชิกที่เป็นไปได้ ในแต่ละช่วงจะโดยจะกำหนด ตัวแปรทางเอาต์พุตฟัซซี่ 1 ตัว เพื่อที่จะนำไปหาค่าการเปลี่ยนแปลง ดิวตี้ไซเคิล (Duty Cycle Change, D_{ch}) ที่ใช้ในวงจรชอปเปอร์ (Chopper) เพื่อเปลี่ยนค่าแรงดันที่ไปตกคร่อมขดลวดอาร์เมเจอร์ (ควบคุมการเปลี่ยนแปลงแรงดันขั้วของมอเตอร์ไฟฟ้ากระแสตรงเพื่อให้ความเร็วคงที่ตามที่แรงดันอ้างอิง) และมีการกำหนดค่าจุดศูนย์กลาง (Assigned Value) ของเอาต์พุตฟัซซี่เพื่อให้สามารถทำการดีฟัซซี่ฟิเคชัน ดังแสดงในตารางที่ 3.2

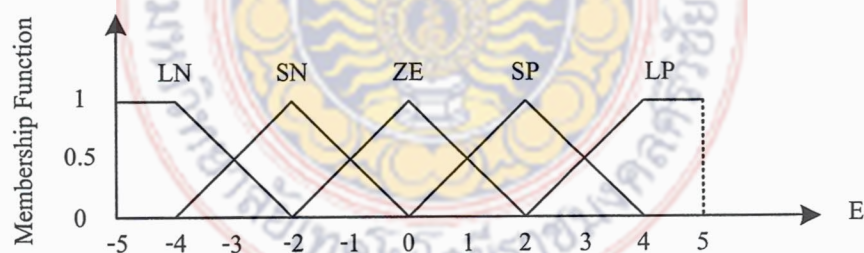
ตารางที่ 3.2 ค่าจุดศูนย์กลางของเอาต์พุตฟัซซี่ของระบบควบคุมความเร็ว

Output Status	Assigned Value
LN	-0.3
SN	-0.15
ZE	0
SP	0.15
LP	0.3

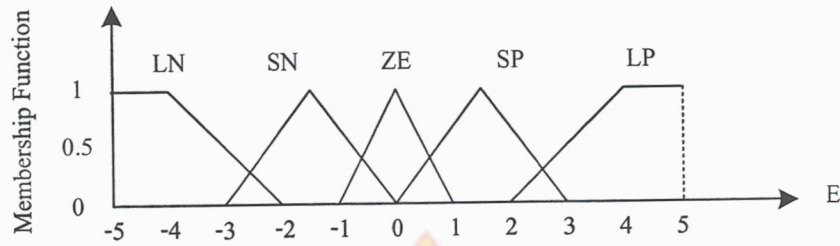
ขั้นตอนที่ 3 การกำหนดขอบเขตเชิงภาษา (Fuzzy Linguistic) และการกำหนดค่าความเป็นสมาชิก (Grade of Membership Function) ของ Fuzzy Subset ต่างๆ ที่ระดับการควอนไทซ์ที่แตกต่างกัน การกำหนดขอบเขตเชิงภาษาซึ่งเป็นภาษาที่ใช้ในระบบการควบคุมแบบฟัซซี่ลอจิกในที่นี้สามารถแบ่งระดับของฟัซซี่ลอจิกที่ใช้กับตัวแปรอินพุตและเอาต์พุตได้ 5 ภาษา ดังนี้

1. Large Negative(LN) : ลบมาก
2. Small Negative(SN) : ลบน้อย
3. Zero(ZE) : ศูนย์
4. Small Positive(SP) : บวกน้อย
5. Large Positive(LP) : บวกมาก

โดยการกำหนดรูปร่างกราฟของฟังก์ชันการเป็นสมาชิกของอินพุตทั้ง 2 ตัว ดังนี้



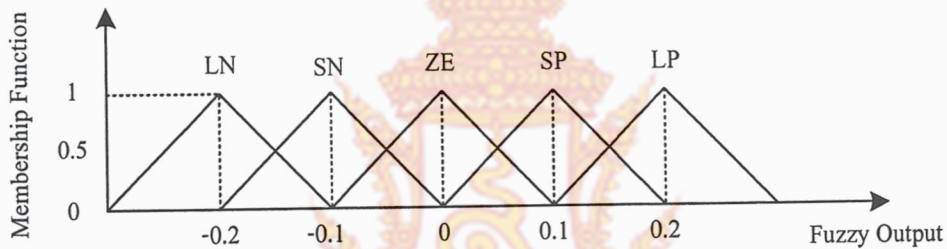
ภาพที่ 3.1 แสดงฟังก์ชันการเป็นสมาชิก 1: Error Frequency (E) ในระบบควบคุมความเร็ว



ภาพที่ 3.2 แสดงฟังก์ชันการเป็นสมาชิก 2: Error Change Frequency (E_{ch})

ในระบบควบคุมความเร็ว

การกำหนดรูปร่างกราฟของฟังก์ชันการเป็นสมาชิกของเอาต์พุตฟัซซี่ลอจิกนั้นทำได้โดยการนำค่าจุดศูนย์กลาง (Assigned Value) จากตารางที่ 3.2 มานั่นเอง ดังรูปที่ 3.3



ภาพที่ 3.3 ฟังก์ชันการเป็นสมาชิกของเอาต์พุตฟัซซี่ลอจิกในระบบควบคุมความเร็ว

ขั้นตอนที่ 4 การสร้างเงื่อนไขหรือกฎการควบคุม หรือการหากฎเกณฑ์และเงื่อนไข ที่สามารถนำค่าจากอินพุตมาประมวลผลออกไปได้ เมื่อได้ทำขั้นตอนในข้างต้นแล้ว ก็จะเป็นการสร้างเงื่อนไขที่จะนำมาประยุกต์ใช้ในการควบคุมของตัวควบคุมฟัซซี่ลอจิก เพื่อให้สามารถควบคุมความเร็วมอเตอร์ไฟฟ้ากระแสตรงได้ตามจุดประสงค์ ขั้นตอนนี้จึงเป็นขั้นตอนที่มีความสำคัญมาก เพราะจะเป็นการจำลองพฤติกรรมของระบบที่เป็นไปได้หรือเป็นการกำหนดให้ตัวควบคุมฟัซซี่ลอจิกปฏิบัติการควบคุม เมื่อมีการเปลี่ยนแปลงความเร็วมอเตอร์ไฟฟ้ากระแสตรงนั่นเอง โดยจะได้ค่าเอาต์พุตฟัซซี่ลอจิกเพื่อนำไปหาค่าการเปลี่ยนแปลงความเร็วตัวเอง ว่าจะมีค่าเท่าไรนั้นก็ขึ้นอยู่กับค่าอินพุตทั้ง 2 ตัว ที่เปลี่ยนแปลงไปตามตารางที่ 3.3

ตารางที่ 3.3 แสดงเงื่อนไขที่กำหนดขึ้นของเอาต์พุตฟัซซี่ในระบบควบคุมความเร็ว
เมื่ออินพุตมีการเปลี่ยนแปลง

Ech \ E	LN	SN	ZE	SP	LP
LN	LN	LN	LN	SN	SN
LP	SP	SP	LP	LP	LP
SN	LN	SN	SN	ZE	ZE
SP	ZE	ZE	SP	SP	LP
ZE	SN	SN	ZE	ZE	SP

หรือสามารถเขียนให้อยู่ในรูปของ IF.....THEN ได้ดังนี้

1. IF E = LN AND Ech = LN THEN Fuzzy Output = LN
2. IF E = LN AND Ech = SN THEN Fuzzy Output = LN
3. IF E = LN AND Ech = ZE THEN Fuzzy Output = LN
4. IF E = LN AND Ech = SP THEN Fuzzy Output = SN
5. IF E = LN AND Ech = LP THEN Fuzzy Output = SN
6. IF E = SN AND Ech = LN THEN Fuzzy Output = LN
7. IF E = SN AND Ech = SN THEN Fuzzy Output = SN
8. IF E = SN AND Ech = ZE THEN Fuzzy Output = SN
9. IF E = SN AND Ech = SP THEN Fuzzy Output = ZE
10. IF E = SN AND Ech = LP THEN Fuzzy Output = ZE
11. IF E = ZE AND Ech = LN THEN Fuzzy Output = SN
12. IF E = ZE AND Ech = SN THEN Fuzzy Output = SN
13. IF E = ZE AND Ech = ZE THEN Fuzzy Output = ZE
14. IF E = ZE AND Ech = SP THEN Fuzzy Output = ZE
15. IF E = ZE AND Ech = LP THEN Fuzzy Output = SP

16. IF E = SP AND Ech = LN THEN Fuzzy Output = ZE
17. IF E = SP AND Ech = SN THEN Fuzzy Output = ZE
18. IF E = SP AND Ech = ZE THEN Fuzzy Output = SP
19. IF E = SP AND Ech = SP THEN Fuzzy Output = SP
20. IF E = SP AND Ech = LP THEN Fuzzy Output = LP
21. IF E = LP AND Ech = LN THEN Fuzzy Output = SP
22. IF E = LP AND Ech = SN THEN Fuzzy Output = SP
23. IF E = LP AND Ech = ZE THEN Fuzzy Output = LP
24. IF E = LP AND Ech = SP THEN Fuzzy Output = LP
25. IF E = LP AND Ech = LP THEN Fuzzy Output = LP

ในที่นี้ใช้ตัวดำเนินการ “AND” โดยใช้ค่าต่ำสุด (Minimum) ของตัวแปรเงื่อนไข (Condition) ในการคำนวณผลลัพธ์รวมของฟัซซี่ลอจิก ซึ่งการเลือกค่าต่ำสุดจะทำให้ความแข็งแกร่งของกฎที่ได้เลือกอยู่นั้นขึ้นกับค่าตัวแปร ส่วนเงื่อนไขที่มีค่าความเป็นสมาชิกน้อยที่สุด

เช่น กรณีที่ 1 : IF E = LN AND Ech = LN THEN Fuzzy Output = LN

มีความหมายว่า ถ้าความผิดพลาด (E) มีค่าเป็นลบมาก (Large Negative) คือมีค่ามากกว่าแรงดันอ้างอิงมาก และมีค่าการเปลี่ยนแปลงค่าผิดพลาด (E_{Δ}) มีค่าเป็นลบมาก (Large Negative) ด้วย คือมีค่าการเปลี่ยนแปลงของความเร็วรอบเพิ่มขึ้นไปจากครั้งที่แล้วมาก ค่าเอาต์พุตฟัซซี่จึงมีค่าเป็นลบมากด้วย ทำให้ค่าดิฟเฟอเรนเชียลของแรงดันที่ไปตกคร่อมขดลวดอาร์เมเจอร์ลดลง ซึ่งก็คือแรงดันที่ไปตกคร่อมขดลวดอาร์เมเจอร์จะลดลง จึงทำให้ความเร็วมอเตอร์ไฟฟ้ากระแสตรงมีค่าลดลงตามไปด้วย

ขั้นตอนที่ 5 การดิฟเฟอเรนเชียลของเอาต์พุต และการกำหนดวิธีการรวมผลของฟัซซี่ลอจิก เพื่อให้ได้ค่าเอาต์พุตที่เป็นค่าจริงในรูปของครีสปเซต (Crisp Set) จากเงื่อนไขในขั้นตอนที่ 3 เมื่อพิจารณาความเป็นไปได้ของค่าเอาต์พุต จะเห็นได้ว่าในแต่ละอินพุตจะได้ค่าเอาต์พุตฟัซซี่ออกมาหลายค่า เราจึงต้องหาค่าที่แม่นยำที่สุดเพียงค่าเดียว เพื่อให้ได้เอาต์พุตที่แท้จริงของตัวควบคุม ฟัซซี่ลอจิก ซึ่งจะใช้กรรมวิธีการหาค่าศูนย์ถ่วงเพื่อหาค่าเซนทรอยด์ โดยสามารถหาได้จากสูตร

$$Y = \frac{\sum_{n=1}^n (\mu_n * Y_n)}{\sum \mu_n}$$

เมื่อ Y = Crisp Output

μ = Membership Function



n = Number of Membership = 1, 2, 3,

ตารางที่ 3.4 แสดงค่าเอาต์พุตฟัซซี่ลอจิกทั้งหมดของทุกกรณีที่เกิดขึ้นในระบบการควบคุมความเร็ว ซึ่งได้จากการคำนวณ โดยกฎฟัซซี่ลอจิก

E \ E _n	-5	-4	-3	-2	-1	0	1	2	3	4	5
-5	-0.3	-0.3	-0.3	-0.3	-0.3	-0.3	-0.15	-0.15	-0.15	-0.15	-0.15
-4	-0.3	-0.3	-0.3	-0.3	-0.3	-0.3	-0.15	-0.15	-0.15	-0.15	-0.15
-3	-0.3	-0.3	-0.3	-0.225	-0.225	-0.225	-0.075	-0.075	-0.075	-0.075	-0.075
-2	-0.3	-0.3	-0.3	-0.15	-0.15	-0.15	0	0	0	0	0
-1	-0.225	-0.225	-0.225	-0.15	-0.15	-0.075	0	0	0.075	0.075	0.075
0	-0.15	-0.15	-0.15	-0.15	-0.15	0	0	0	0.15	0.15	0.15
1	-0.075	-0.075	-0.075	-0.075	-0.075	0.075	0.075	0.075	0.225	0.225	0.225
2	0	0	0	0	0	0.15	0.15	0.15	0.3	0.3	0.3
3	0.075	0.075	0.075	0.075	0.075	0.225	0.225	0.225	0.3	0.3	0.3
4	0.15	0.15	0.15	0.15	0.15	0.3	0.3	0.3	0.3	0.3	0.3
5	0.15	0.15	0.15	0.15	0.15	0.3	0.3	0.3	0.3	0.3	0.3

ตัวอย่างการคำนวณค่าของเอาต์พุตฟัซซี่ลอจิกในตารางที่ 3.4

จากกรณีค่า E = 3 มีค่า $\mu_1(SP) = 0.5$, $\mu_2(LP) = 0.5$ และค่า $E_{ch} = -3$ มีค่า $\mu(LN) = 0.5$ ซึ่งจากตารางที่ 3.3 พบว่าเอาต์พุตฟัซซี่ลอจิกสามารถเกิดขึ้นได้ 2 กรณี คือ

- เงื่อนไขในกรณีที่ 1 คือ

IF E = SP AND E_{ch} = LN THEN Fuzzy Output = ZE

จากเงื่อนไขและตารางที่ 3.3 จะทราบว่าเอาต์พุตฟัซซี่ลอจิกอยู่ในสับเซต ZE โดยมีค่าเอาต์พุตฟัซซี่ลอจิกในกรณีที่ 1 = 0 และ $\mu(\text{Fuzzy Output}_1)$ คือ

$$\begin{aligned}
 \mu(\text{Fuzzy Output}_1) &= \min[\mu_1(E), \mu(E_{ch})] \\
 &= \min[\mu_1(SP), \mu(LN)] \\
 &= \min[0.5, 0.5] \\
 &= 0.5
 \end{aligned}$$

- เงื่อนไขในกรณีที่ 2 คือ

IF E = LP AND Ech = LN THEN Fuzzy Output = SP

จากเงื่อนไขและตารางที่ 3.3 จะทราบว่าเอาต์พุตฟัซซี่ลอจิกอยู่ในสับเซต SP โดยมีค่าเอาต์พุตฟัซซี่ลอจิกในกรณีที่ 2 = 0 และ $\mu(\text{Fuzzy Output}_2)$ คือ

$$\begin{aligned}\mu(\text{Fuzzy Output}_2) &= \min[\mu_2(E), \mu(E_{ch})] \\ &= \min[\mu_2(LP), \mu(LN)] \\ &= \min[0.5, 0.5] \\ &= 0.5\end{aligned}$$

จากค่าเอาต์พุตฟัซซี่ลอจิกทั้ง 2 ตัว นำมาหาค่าเอาต์พุตฟัซซี่ลอจิกที่แท้จริง โดยใช้สูตรการหาค่าศูนย์ถ่วงจะได้

$$\text{Output} = \{[\mu(\text{Fuzzy Output}_2) * \text{Fuzzy Output}_1] + [\mu(\text{Fuzzy Output}_1) * \text{Fuzzy Output}_2]\} /$$

$$\begin{aligned}& [\mu(\text{Fuzzy Output}_1) + \mu(\text{Fuzzy Output}_2)] \\ &= \{[0.5 * (0)] + [0.5 * (0.15)]\} / (0.5 + 0.5) \\ &= 0.075 \text{ ซึ่งมีค่าตรงกับตารางที่ 3.4}\end{aligned}$$

ขั้นตอนที่ 6 เปลี่ยนแปลงค่าเอาต์พุตฟัซซี่ลอจิกที่หาได้จากตารางที่ 3.4 เป็นค่าการเปลี่ยนแปลงของคิวดัไอเซลล์หลังจากที่ได้ทำการคิฟิซซี่ฟิเคชันเสร็จแล้ว ก็จะได้ผลลัพธ์ในการคำนวณในรูปแบบเอาต์พุตฟัซซี่ ซึ่งค่าเอาต์พุตฟัซซี่นี้ยังไม่สามารถนำไปใช้ในการควบคุมความเร็วของมอเตอร์ไฟฟ้ากระแสตรงได้โดยตรง จึงต้องนำค่านี้มาผ่านกระบวนการเปลี่ยนให้ค่าของเอาต์พุตฟัซซี่นั้นสามารถควบคุมความเร็วของมอเตอร์ไฟฟ้ากระแสตรงได้ ซึ่งวิธีการเปลี่ยนก็คือนำค่าของเอาต์พุตฟัซซี่ไปเปรียบเทียบกับตารางเอาต์พุต (ตารางที่ 3.5) ซึ่งเป็นกระบวนการกลับกันกับการหาอินพุตของฟัซซี่ เพื่อหาค่าการเปลี่ยนแปลงคิวดัไอเซลล์ เมื่อได้ทำการเปลี่ยนค่าเอาต์พุตเป็นค่าการเปลี่ยนแปลงของคิวดัไอเซลล์แล้ว ก็จะสามารถนำค่าการเปลี่ยนแปลงของคิวดัไอเซลล์ไปควบคุมความเร็วของมอเตอร์ไฟฟ้ากระแสตรงได้ต่อไป

ตารางที่ 3.5 แสดงค่าความสัมพันธ์ของเอาต์พุตพีชชีกับค่าการเปลี่ยนแปลงของคิวตี้ไซเคิล ของความเร็ว

เอาต์พุต	ค่าการเปลี่ยนแปลงของคิวตี้ไซเคิล
[0. 226, 0.3]	7
[0. 151 , 0. 225]	3
[0. 076, 0. 15]	1
[0. 01 , 0. 075]	0
0	0
0	
-1	
-3	
-7	

เมื่อได้ค่าของอินพุตครบทั้ง 2 ตัวแล้วก็สามารถหาค่าของเอาต์พุตพีชชีได้จากตารางที่ 3.4 แล้วนำค่ามาเปรียบเทียบกับช่วงของเอาต์พุตเพื่อหาค่าการเปลี่ยนแปลงของคิวตี้ไซเคิลในตารางที่ 3.5 สมมติเมื่อเราต้องการให้มอเตอร์หมุนที่ 1500 รอบต่อนาที มีค่าคิวตี้ไซเคิลประมาณ 68 % ถ้ามอเตอร์หมุนด้วยความเร็ว 1400 รอบต่อนาที มีผลทำให้

$$\text{ค่าความผิดพลาด (Error)} = 14000 - 13900 = 100$$

$$\text{ค่าการเปลี่ยนแปลงความผิดพลาด (Error Change)} = 100 - 0 = 100$$

จากตาราง เราจะได้ค่า $E = 4$ และค่า $E_{ch} = 4$ ซึ่งจะได้ค่าเอาต์พุตของระบบเท่ากับ 0.3 นำค่าที่ได้ไปเปรียบเทียบกับค่าการเปลี่ยนแปลงของคิวตี้ไซเคิลในตารางที่ 3.5 จะได้ค่าการเปลี่ยนแปลงของคิวตี้ไซเคิลเท่ากับ 7 นั่นเอง ดังนั้นค่าคิวตี้ไซเคิลจะค่าเท่ากับ $68 + 7 = 75$ ซึ่งเมื่อค่าคิวตี้ไซเคิลเพิ่มขึ้นจะทำให้แรงดันที่ไปตกคร่อมขดลวดอาร์เมเจอร์มีค่าเพิ่มขึ้น ทำให้มอเตอร์ไฟฟ้ากระแสตรงมีความเร็วเข้าใกล้ความเร็วอ้างอิง ซึ่งจากการยกตัวอย่างมีเท่ากับ 14000 รอบต่อนาที

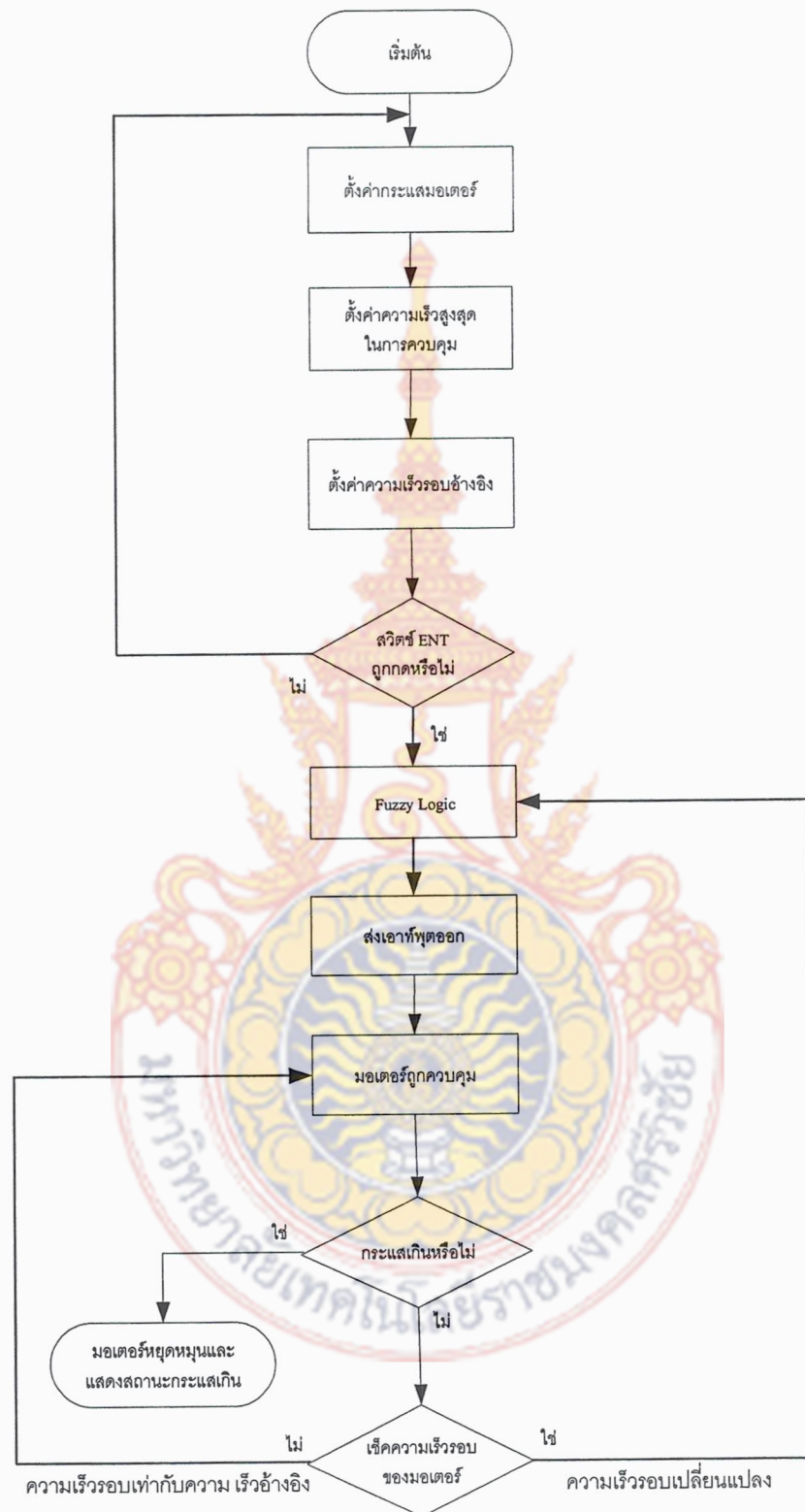
3.2 การออกแบบโปรแกรมและฮาร์ดแวร์

การออกแบบตัวควบคุม เพื่อทำหน้าที่ควบคุมให้ระบบหรือกระบวนการที่ต้องการควบคุม มีสัญญาณด้านออกหรือผลตอบสนองตามที่ต้องการในรูปแบบของระบบควบคุมแบบป้อนกลับ (Feedback Control System) นั้น คือ การนำเอาสัญญาณด้านออกกลับมาเปรียบเทียบกับสัญญาณด้านเข้า ซึ่งการป้อนกลับแบบนี้จะต้องเป็นการป้อนกลับแบบลบ (Negative Feedback) จากนั้น นำค่าความแตกต่างระหว่างด้านเข้ากับด้านออก ส่งต่อไปยังส่วนสร้างสัญญาณของตัวควบคุม เพื่อสร้างสัญญาณควบคุมที่เหมาะสมที่จะทำให้สัญญาณด้านออกของระบบที่ถูกควบคุมมีค่าตาม ที่ต้องการ ซึ่งในการสร้างคอนโทรลเลอร์สามารถจำแนกออกเป็น 2 ส่วนหลัก ๆ ได้แก่ ส่วนของโปรแกรม (Soft Ware) และ ส่วนของฮาร์ดแวร์ (Hard Ware)

3.2.1 ส่วนของโปรแกรม (Soft Ware)

อินพุตของดิจิตอล คอนโทรลเลอร์หรือสัญญาณด้านเข้านั้นแบ่งเป็น 2 ส่วน ได้แก่ อินพุตอ้างอิง (Reference Input) หรือค่าที่ตั้งไว้ (Set Point) และ อินพุตที่เป็นตัวป้อนกลับมาเปรียบเทียบกับ (Feedback Input) ซึ่งเป็นสัญญาณอนาล็อก พร้อมกับสัญญาณอนาล็อกจากเคอร์เร็นทรานสดิวเซอร์ (Current Transducer) ดังนั้นจะต้องแปลงสัญญาณอนาล็อกเป็นดิจิตอล โดยเรียกใช้งาน โมดูลแปลงสัญญาณอนาล็อกเป็นดิจิตอล



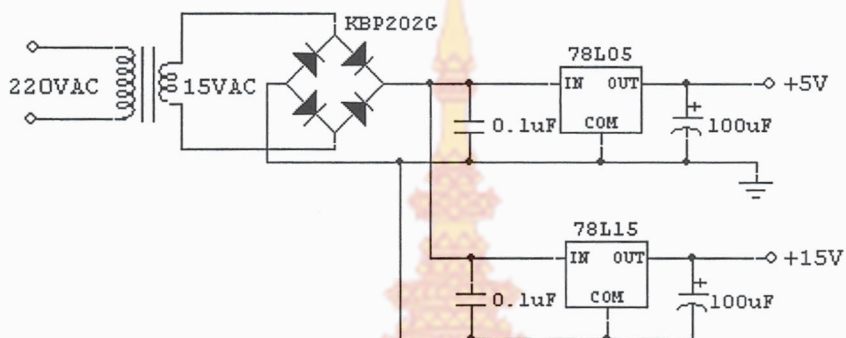


ภาพที่ 3.4 โฟลวชาร์ตการทำงานของไมโครคอนโทรลเลอร์ PIC16F877

3.2.2 ส่วนของฮาร์ดแวร์ (Hard Ware)

1. วงจรแหล่งจ่ายไฟเลี้ยงของคอนโทรลเลอร์

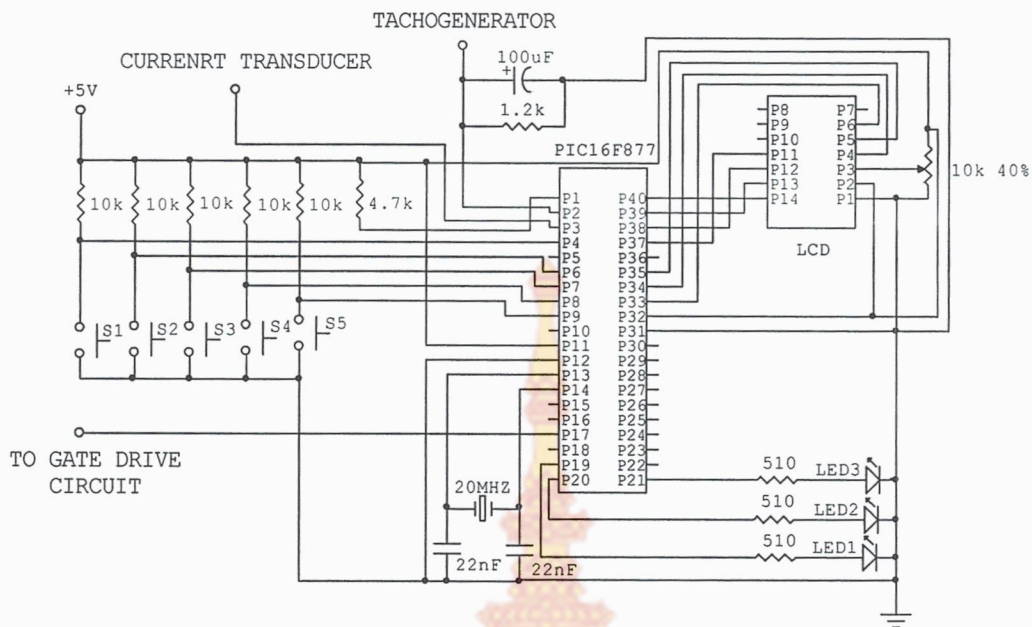
ในวงจรแหล่งจ่ายไฟเลี้ยงชุดคอนโทรลเลอร์แบ่งออกเป็น 2 ส่วนหลัก ๆ ได้แก่ ชุดจ่ายไฟเลี้ยง 5 Vdc สำหรับไมโครคอนโทรลเลอร์ PIC 16F877 ชุดจ่ายไฟเลี้ยง 15 Vdc สำหรับทรานซิสเตอร์ BD139 และ ไอซีเบอร์ IR2110 แสดงการต่อวงจรดังภาพที่ 3.5



ภาพที่ 3.5 วงจรแหล่งจ่ายไฟเลี้ยงของดิจิทัลคอนโทรลเลอร์

2. วงจรชุดไมโครคอนโทรลเลอร์ PIC 16F877

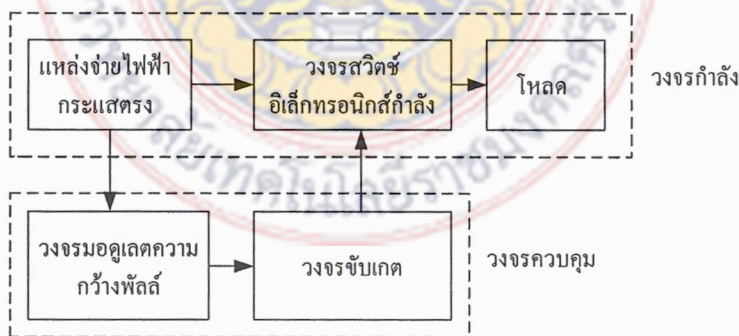
วงจรชุดนี้เป็นวงจรหลักของคอนโทรลเลอร์ เพื่อทำหน้าที่ประมวลผลการคำนวณและส่งค่าเอาต์พุตออก โดยมีไมโครคอนโทรลเลอร์ตระกูล PIC16F877 ที่ใช้ไฟเลี้ยง 5 Vdc ใช้สัญญาณนาฬิกา 20 MHz ในส่วนของการป้อนค่าต่างๆ นั้นจะใช้สวิทช์ปุ่มกด (Push Button) เป็นอินพุต แสดงผลด้วยจอ LCD 16x2 และหลอด LED ซึ่งสามารถแสดงการต่อใช้งานดังภาพที่ 3.6



ภาพที่ 3.6 วงจรชุดไมโครคอนโทรลเลอร์ PIC 16F877

3. วงจรขับเคลื่อน

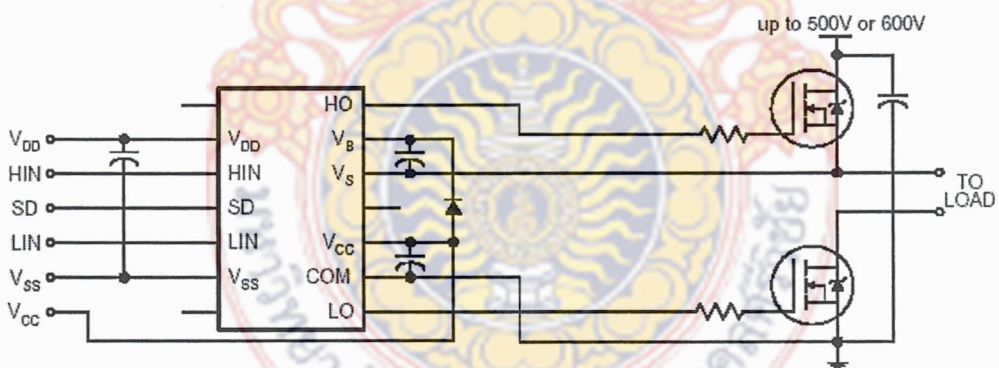
วงจรขับเคลื่อน (Gate Drive) เป็นวงจรส่วนที่ทำหน้าที่เชื่อมโยงสัญญาณพัลส์แบบมอดูเลตความกว้างพัลส์จากวงจรรวมกำเนิดสัญญาณแบบต่างๆ กับสวิทช์อิเล็คทรอนิกส์กำลังไฟตรง เช่น ทรานซิสเตอร์กำลัง มอสเฟตกำลัง หรือไอจีบีที เป็นต้น โดยทั่วไปวงจรขับเคลื่อนจะทำหน้าที่แยก (Isolate) ความเชื่อมโยงทางไฟฟ้าระหว่างสัญญาณพัลส์ควบคุมกับสวิทช์อิเล็คทรอนิกส์กำลังในวงจรกำลัง และทำหน้าที่ตัดสัญญาณควบคุมสวิทช์อิเล็คทรอนิกส์กำลัง ในกรณีที่วงจรกำลังกำลังอยู่ในสภาวะกระแสไหลเกินหรือแรงดันสูงกว่าพิกัด เป็นต้น ลักษณะของวงจรขับเคลื่อนจะรวมกำเนิดสัญญาณมอดูเลตความกว้างพัลส์ และสวิทช์อิเล็คทรอนิกส์กำลัง



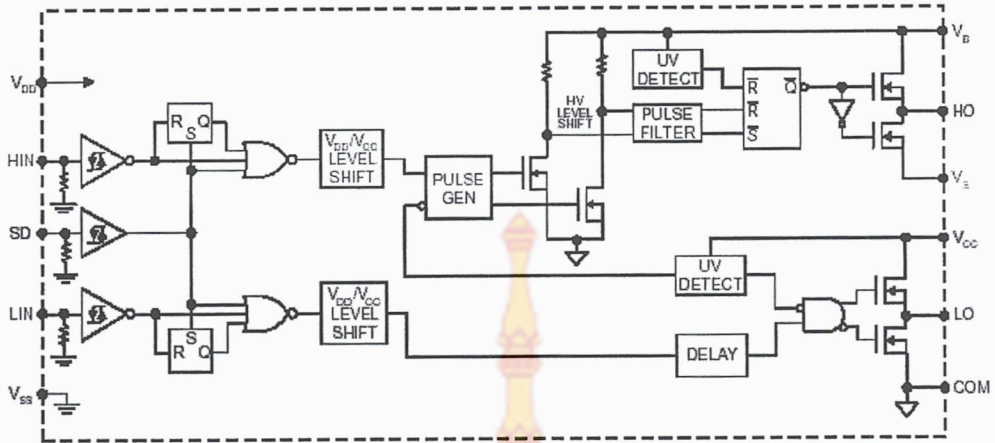
ภาพที่ 3.7 แสดงความสัมพันธ์ของวงจรกำเนิดสัญญาณ PWM วงจรขับเคลื่อน และวงจรสวิทช์อิเล็คทรอนิกส์กำลัง

การเลือกวงจรขับเคลื่อน ต้องเลือกให้เหมาะสมกับสวิตช์อิเล็กทรอนิกส์กำลังที่ใช้ เช่น วงจรขับเคลื่อนสำหรับทรานซิสเตอร์กำลัง ซึ่งมีการสวิตช์ที่ความถี่ไม่สูงมากนักประมาณ ± 15 kHz อาจใช้วงจรขับเคลื่อนที่แยกสัญญาณด้วยตัวเชื่อมโยงทางแสง หรือสำหรับวงจรขับเคลื่อนมอสเฟตกำลัง และไอจีบีที ซึ่งเป็นสวิตช์อิเล็กทรอนิกส์กำลังที่ทำงานในย่านความถี่สูงๆ ในช่วง 10 kHz - 20 kHz นิยมใช้วงจรขับเคลื่อนที่เป็นวงจรรวมแบบไฮบริดจ์ ซึ่งทำหน้าที่แยกสัญญาณ (Isolate) และมีวงจรป้องกันมอสเฟตกำลัง และไอจีบีทีที่นิยมใช้นั้นมีหลายเบอร์ เช่น IR 2110, IR 2213 หรือ EXB8XX ของ Fuji Electric เป็นต้น

วงจรขับเคลื่อนจะทำหน้าที่ขับให้ไอจีบีทีในส่วนของวงจรภาคเพาเวอร์ แต่ถ้าเราจะสร้างวงจรขับเคลื่อนจะต้องใช้อุปกรณ์หลายชิ้น ทำให้ส่วนของวงจรขับเคลื่อนมีขนาดใหญ่โตมาก จึงไม่สะดวกในการนำไปใช้งานและเอาท์พุทของวงจรขับเคลื่อนที่ได้จะไม่แน่นอน เพราะจะขึ้นอยู่กับการทำงานของอุปกรณ์หลายตัวด้วยกัน และนอกจากนี้ผลของการเปลี่ยนแปลงของอุณหภูมิและสถานะแวดล้อม จะมีผลต่อการทำงานของวงจรขับเคลื่อนด้วย ซึ่งอาจจะทำให้การทำงานของวงจรผิดพลาดไปด้วย ดังนั้นจึงใช้ IR2110 เป็นตัวขับเคลื่อน เนื่องจากมีคุณสมบัติที่เหมาะสมได้แก่ มีขนาดเล็กให้แรงดันเอาท์พุทที่เหมาะสมกับการนำไปขับไอจีบีที โดย IR2110 จะมีอินพุตสองอินพุตและเอาท์พุทสองเอาท์พุท ดังนั้น IR2110 เพียงตัวเดียวสามารถนำไปขับไอจีบีทีได้ถึง 2 ตัว เพราะให้เอาท์พุท 2 ค่า โดยเอาท์พุททั้งสองนี้ จะขึ้นอยู่กับอินพุตที่ป้อนซึ่งลักษณะการต่อ IR2110 เพื่อนำไปขับเคลื่อน



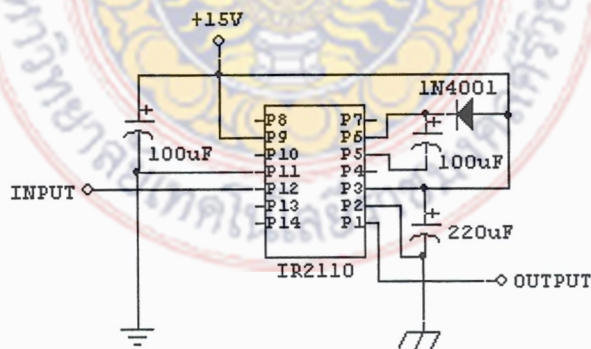
ภาพที่ 3.8 แสดงลักษณะการต่อของไอซีเบอร์ IR2110



ภาพที่ 3.9 แสดงวงจรภายในของไอซีเบอร์ IR2110

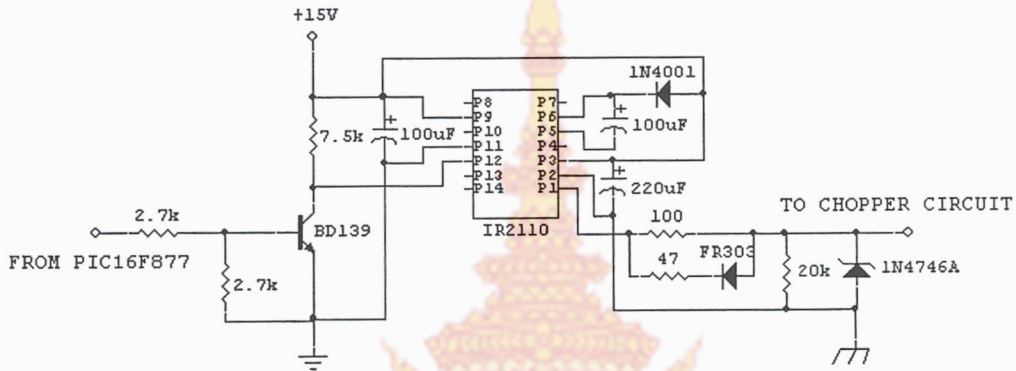
วงจรขับเคลื่อนที่นำมาขับเพาเวอร์มอสเฟสหรือไอจีบีทีนั้น ถ้าเราจะสร้างวงจรขับเองจะต้องใช้อุปกรณ์หลายชิ้นทำให้ส่วนของวงจรขับเคลื่อนมีขนาดใหญ่มาก จึงไม่สะดวกในการนำไปใช้งาน และเอาต์พุตของวงจรขับเคลื่อนที่ได้จะไม่แน่นอน เพราะจะขึ้นอยู่กับการทำงานของอุปกรณ์หลายตัวด้วยกัน และนอกจากนี้ผลของการเปลี่ยนแปลงของอุณหภูมิและสภาพแวดล้อม จะมีผลต่อการทำงานของวงจรขับเคลื่อนด้วยซึ่งอาจจะทำให้การทำงานของวงจรเพาเวอร์มอสเฟส หรือไอจีบีทีผิดพลาดไปด้วย จึงใช้ IR2110 เป็นตัวขับเคลื่อน เนื่องจากมีคุณสมบัติที่เหมาะสมได้แก่ มีขนาดเล็ก ให้แรงดันเอาต์พุตที่เหมาะสมกับการนำไปขับเพาเวอร์มอสเฟส หรือไอจีบีที

เนื่องจากไอซีเบอร์ IR2110 นั้นมีอินพุตสองอินพุตและเอาต์พุต แต่ในโครงการนี้ใช้เพียงอินพุตและเอาต์พุตเดียว ซึ่งแสดงลักษณะการต่อได้ดังภาพที่ 3.10



ภาพที่ 3.10 การต่อไอซีเบอร์ IR2110

จากภาพที่ 3.11 จะเห็นว่าอินพุตของไอซี IR2110 จะมาจากทรานซิสเตอร์เบอร์ BD139 ซึ่งทำหน้าที่ขยายสัญญาณจากไมโครคอนโทรลเลอร์ PIC 16F877 ที่อยู่ในรูปของสัญญาณ PWM (Pulse – width modulation) ระดับแรงดัน 0–5 Vdc ให้เพิ่มขึ้นเพื่อขับไอซี IR2110 และใช้ซีเนอร์ไดโอด 18 V รักษาระดับแรงดัน V_{GS} ใช้ไดโอดเบอร์ FR303 ต่อรีเวิร์สไบแอสอนุกรมกับความต้านทาน 47 Ω และคร่อมความต้านทาน 100 Ω เพื่อให้ได้เวลาในการเริ่มต้นการทำงานที่เหมาะสม



ภาพที่ 3.11 วงจรขับเกด

4. การสร้าง Reactor (L)

เมื่อไอจีบีทีมีขีดจำกัด อัตราการเปลี่ยนแปลงของกระแส $\frac{di}{dt} = 120 \text{ A}/\mu\text{sec}$ และทำการให้ไอจีบีทีเริ่มนำกระแสซึ่งเป็นไฟฟ้ากระแสตรง 220 Vdc

$$120 \times 10^6 = \frac{220}{L}$$

$$L = 1.83 \mu\text{H}$$

ดังนั้นเพื่อไม่ให้อัตราเปลี่ยนแปลงของกระแสมากกว่าขีดจำกัด 120 แอมแปร์ต่อไมโครวินาที ($\text{A}/\mu\text{sec}$) จึงต้องใช้อินดักเตอร์มากกว่า 1.83 μH โดยให้ 1 mH ต่ออนุกรมกับโหลด

Reactor (L) ที่ใช้ในวงจร Chopper สามารถสร้างขึ้นโดยการพันลวดทองแดงรอบแกนอากาศ (Air Core) เพราะไม่ต้องการให้สนามแม่เหล็กเกิดการอิ่มตัว (Saturation) โดยค่ารีแอคแตนซ์ที่เรากำหนดไว้คือ 1 mH

จากสมการ

$$L = \frac{N^2 \mu A}{l}$$

- เมื่อ L = ค่าอินดักแตนซ์ (Inductance) มีหน่วยเป็นเฮนรี (Henry)
 N = จำนวนรอบของการพันขดลวด (Number of turn)
 μ = Permeability (H/m)
 A = พื้นที่หน้าตัดของแกน (m^2)
 l = ความยาวของแกน (m)

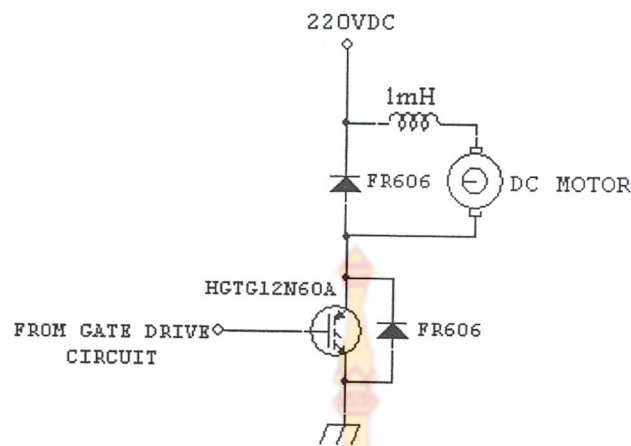
สำหรับ Reactor ที่เราต้องการสร้างนั้นมีขนาด $L = 1$ mH, มีความยาวของแกน $l = 0.09$ m, รัศมีของแกนเท่ากับ 2.5 cm. ดังนั้นจะได้ขนาดพื้นที่หน้าตัด (A) เท่ากับ $\pi (2.5 \times 10^{-2})^2 m^2$ และจากการที่เป็นแกนอากาศจึงได้ค่า Permeability (μ) = $4\pi \times 10^{-7}$ H/m แทนค่าพารามิเตอร์ในสมการที่ 2.1 เพื่อหาจำนวนรอบของการพันขดลวด (N) ดังนั้นจะได้

$$N = \sqrt{\frac{1 \times 10^{-3} \times 0.09}{4\pi \times 10^{-7} \times \pi (2.5 \times 10^{-2})^2}} = 191 \text{ รอบ}$$

5. วงจรชอปเปอร์ภาคกำลัง

ออกแบบโดยใช้วงจรชอปเปอร์ หรือวงจรชอปเปอร์กระแสตรง (DC Chopper) อาจเรียกอีกชื่อหนึ่งว่าวงจรสับไฟฟ้้ากระแสตรง เป็นวงจรที่ทำหน้าที่แปลงผันพลังงานไฟฟ้าจากแหล่งจ่ายกระแสตรงที่มีแรงดันคงที่ ให้เป็นพลังงานไฟฟ้ากระแสตรงที่ปรับค่าได้ โดยใช้สวิตช์อิเล็กทรอนิกส์กำลังเป็นตัวเปิด - ปิดสวิตช์อิเล็กทรอนิกส์กำลังที่นิยมใช้ในวงจรชอปเปอร์ เช่น ไอจีบีที เนื่องจากสวิตช์อิเล็กทรอนิกส์กำลังเหล่านี้ เปิด-ปิดได้ด้วยสัญญาณควบคุมที่ป้อนเข้าทางอินพุต สัญญาณควบคุมนี้คือสัญญาณมอดูเลตความกว้างพัลส์ที่สามารถปรับค่า t_{on} ในขณะที่ T คงที่ได้

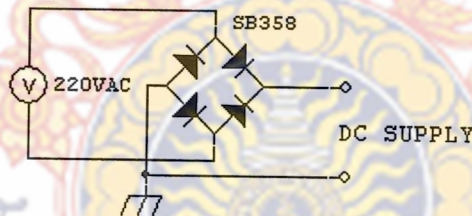
ในวงจรชอปเปอร์จะใช้ไอจีบีที ซึ่งสามารถทนกระแสได้ 12 A และทนแรงดันได้ 600 V โดยไอจีบีที จะทำงานเมื่อมีสัญญาณเข้ามาที่ขาเกตของไอจีบีที ซึ่งสัญญาณที่เข้ามาที่ขาเกตของไอจีบีทีนั้นจะเป็นสัญญาณพัลส์ ซึ่งส่งมาจากวงจรขับเคลื่อนโดยใช้ไอซีเบอร์ IR2110 ซึ่งเป็นไอซีที่ใช้ในการขับเพาเวอร์มอสเฟส และไอจีบีทีโดยเฉพาะ



ภาพที่ 3.12 แสดงการต่อวงจรภาคกำลัง

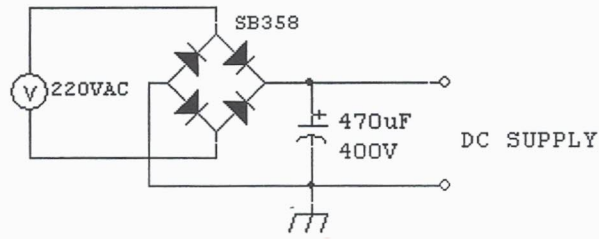
6. วงจรรักษาแรงดันภาคจ่ายไฟ

วงจรจ่ายไฟเลี้ยงให้แก่ขดลวดฟิลต์นั้น จะแปลงแรงดันไฟฟ้ากระแสสลับ 220 V ซึ่งจะแปลงเป็นแรงดันไฟฟ้ากระแสตรง 220 V ที่มีค่าคงที่ เพื่อจ่ายให้แก่ขดลวดฟิลต์ของมอเตอร์ไฟฟ้ากระแสตรง โดยจะใช้บริดจ์ที่ทนแรงดันได้สูง โดยจะใช้บริดจ์ SB358 ซึ่งทนแรงดันได้ 400 V ดังแสดงในภาพที่ 3.13



ภาพที่ 3.13 วงจรแปลงแรงดันเป็นไฟกระแสตรง จ่ายให้กับขดลวดฟิลต์

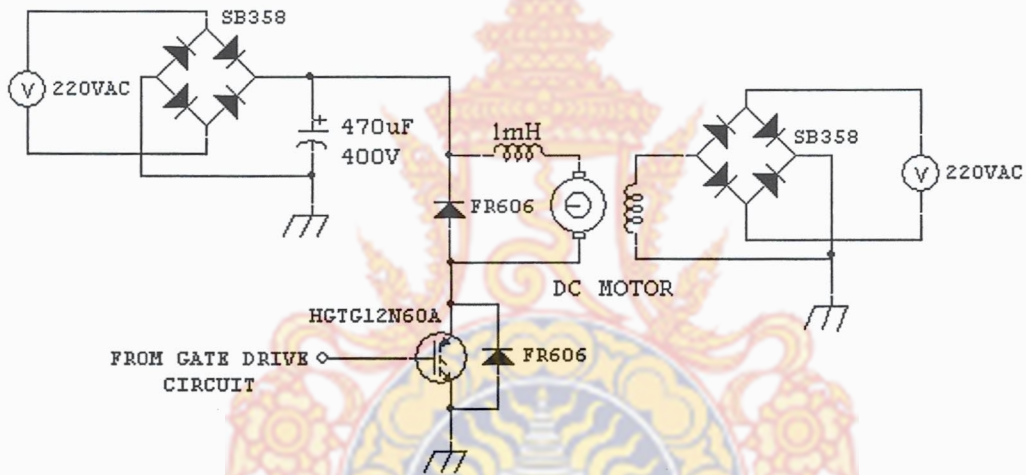
วงจรจ่ายไฟเลี้ยงให้แก่ขดลวดอาร์เมเจอร์จะเหมือนกับวงจรจ่ายไฟเลี้ยงให้แก่ขดลวดฟิลต์ แต่จะมีการต่อคาปาซิเตอร์ที่มีอัตราการทำงานแรงดันสูง ขนทานเข้ามาเพื่อยกระดับแรงดันให้เพิ่มขึ้น และจ่ายให้กับมอเตอร์ ดังแสดงในภาพที่ 3.14



ภาพที่ 3.14 วงจรแปลงแรงดันเป็นไฟกระแสตรง จ่ายให้กับขดลวดอาร์เมเจอร์

7. วงจรโดยรวมของวงจรภาคกำลัง

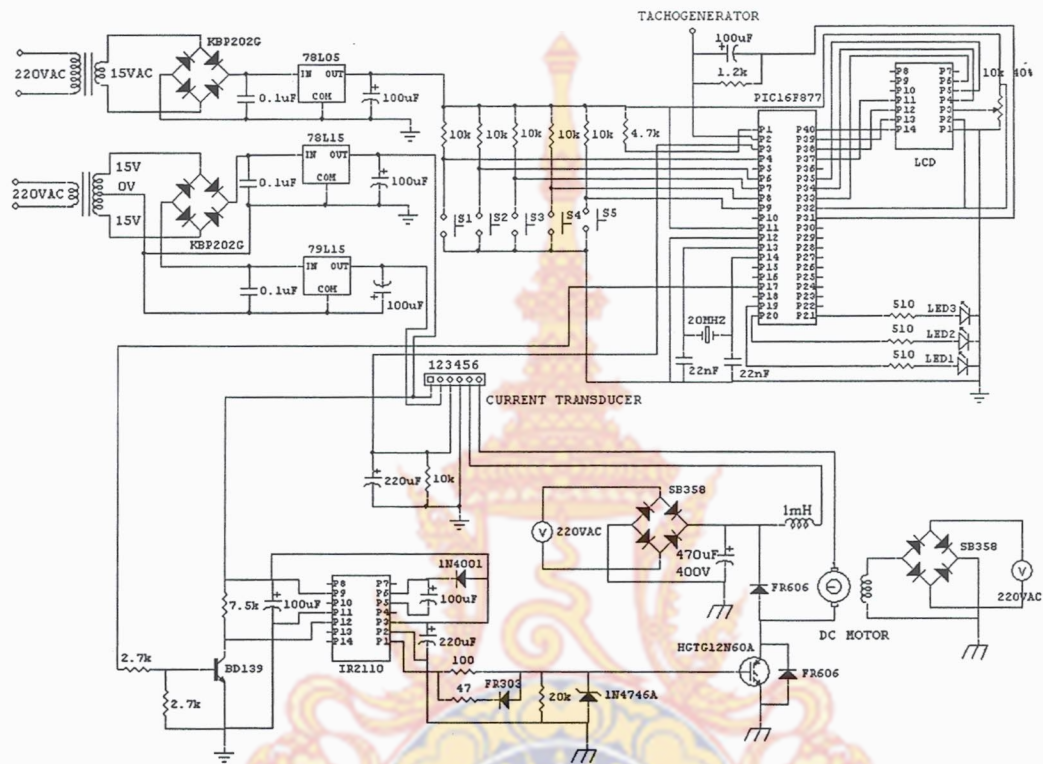
วงจรโดยรวมของวงจรภาคกำลัง จะประกอบด้วยวงจรชอปเปอร์ วงจรไฟเลี้ยง ขดลวดฟิลด์ของมอเตอร์ และไฟเลี้ยงขดลวดอาร์เมเจอร์ของมอเตอร์



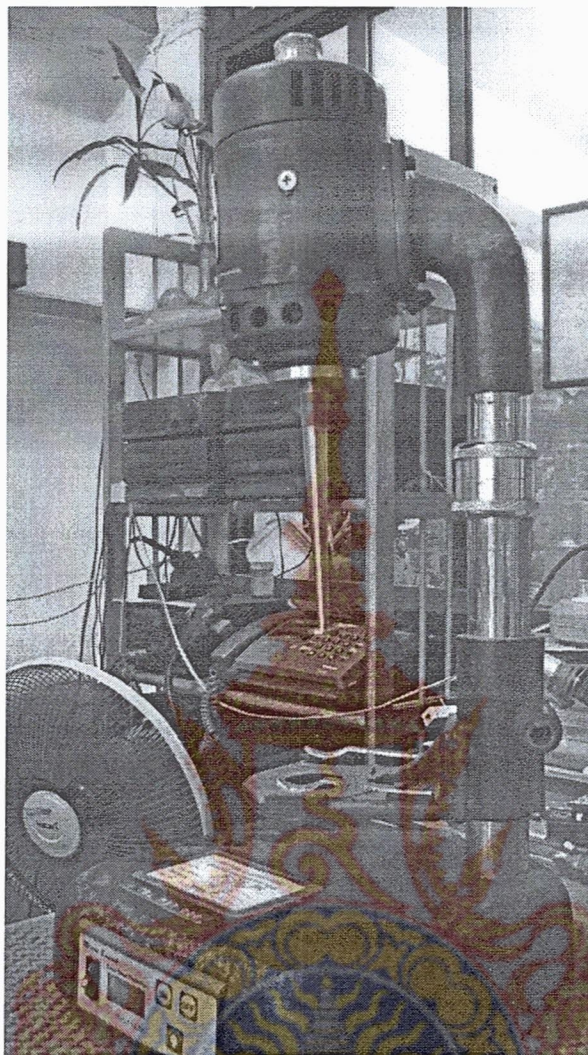
ภาพที่ 3.15 วงจรรวมทั้งหมดของภาคกำลัง

9. วงจรโคจรรวมของฮาร์ดแวร์

วงจรโคจรรวมของฮาร์ดแวร์ที่ออกแบบมาทั้งหมด ซึ่งประกอบด้วยชุดคอนโทรลเลอร์ และชุดเพาเวอร์ ดังแสดงในภาพที่ 3.18



ภาพที่ 3.16 แสดงการต่อวงจรรวมของฮาร์ดแวร์



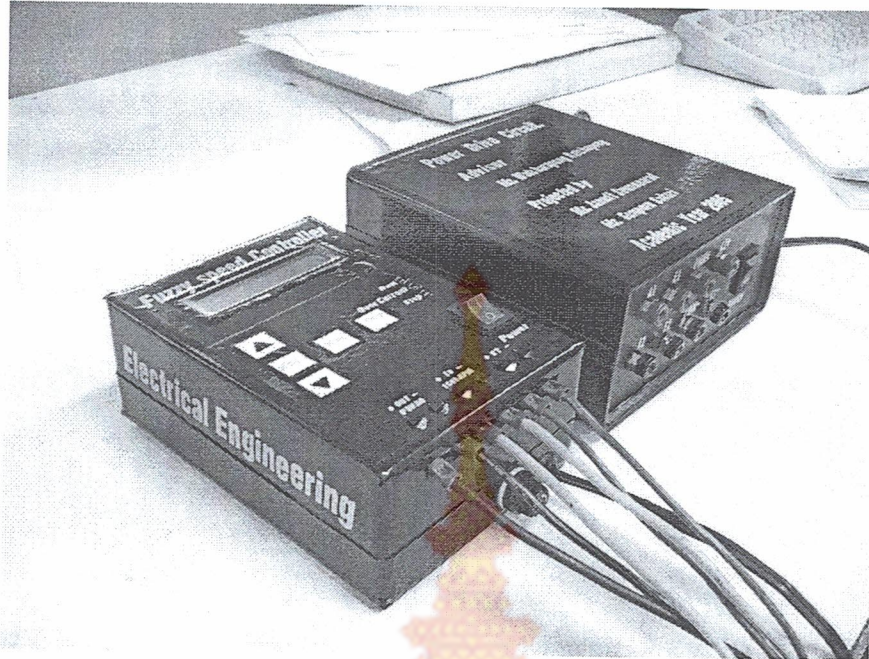
ภาพที่ 3.17 ชุดมอเตอร์ปั่นน้ำยางข้น





ภาพที่ 3.18 ชุดควบคุมความเร็วของมอเตอร์กระแสตรง





ภาพที่ 3.19 ชุดควบคุมและชุดขับมอเตอร์กระแสตรง

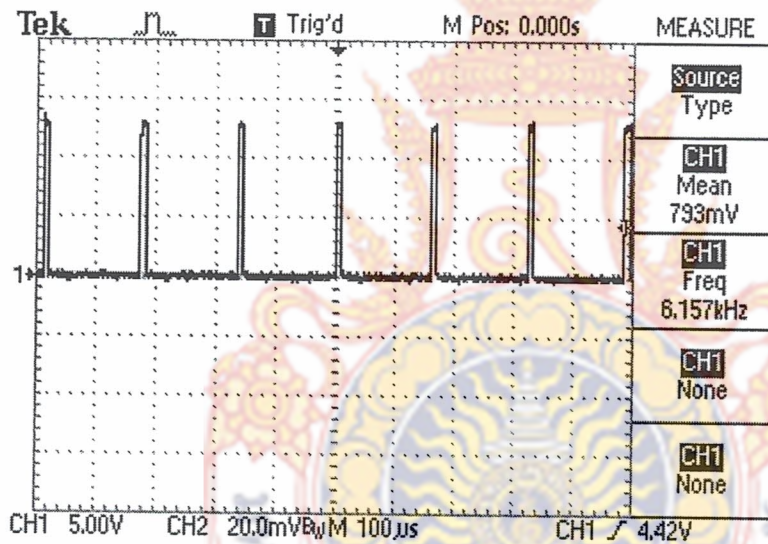


บทที่ 4

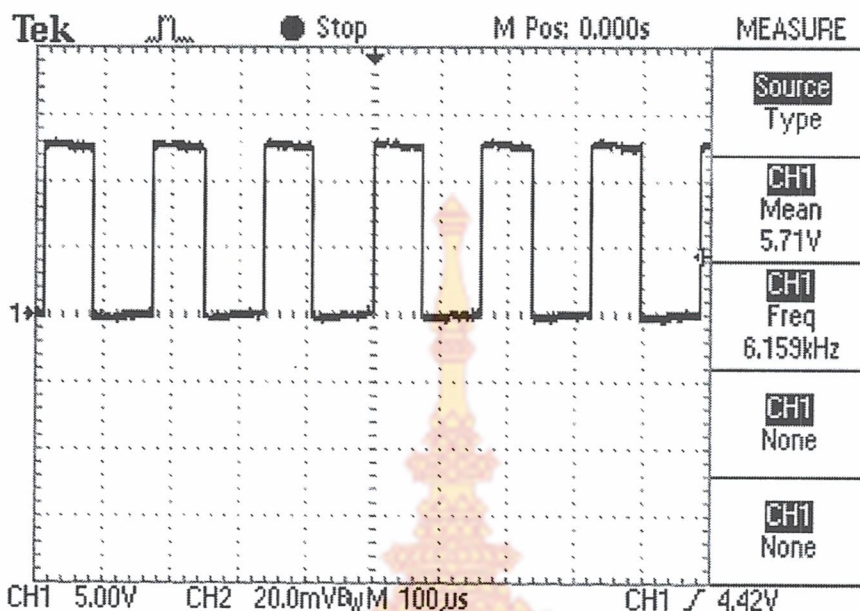
การทดลอง

4.1 การทดลองวงจรขับเคลื่อน

การทดลองนี้ทำการทดลองโดยกำหนดค่าตัวชี้แจงเกิดจากไมโครคอนโทรลเลอร์ แล้วบันทึกสัญญาณเอาต์พุตเพื่อขับเคลื่อนของไอจีบีทีเพื่อจ่ายแรงดันแบบพัลส์วิดท์ให้แก่มอเตอร์ โดยลักษณะของสัญญาณขับเคลื่อนไอจีบีทีของไมโครคอนโทรลเลอร์ ด้วยความถี่ 6.158 kHz เป็นดังในภาพที่ 4.1 และในภาพที่ 4.2



ภาพที่ 4.1 สัญญาณขับเคลื่อนไอจีบีทีของไมโครคอนโทรลเลอร์ ที่ค่าตัวชี้แจงเกิด 10%



ภาพที่ 4.2 สัญญาณขับเคลื่อนไอจีบีทีของไมโครคอนโทรลเลอร์ ที่ค่าความถี่ 50%

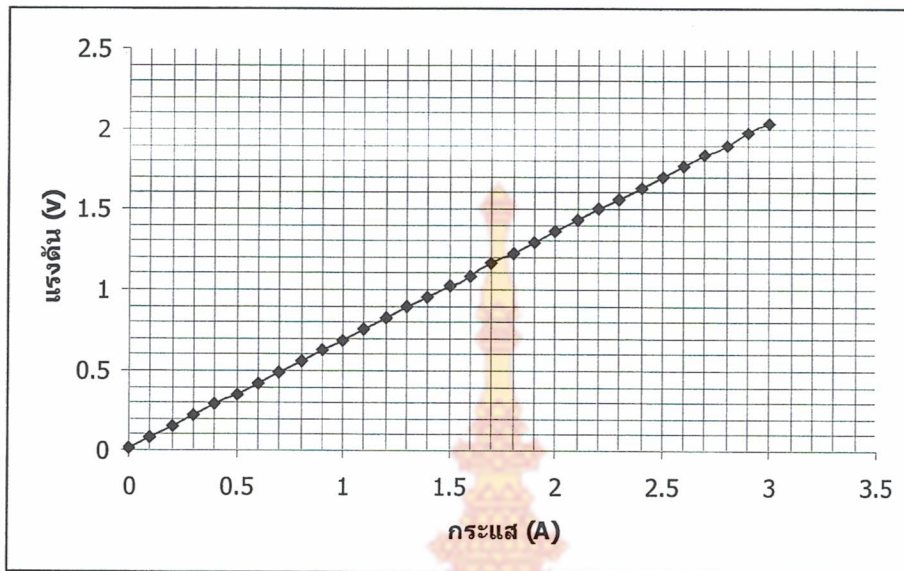
จากทดลองเมื่อกำหนดค่าความถี่ของไมโครคอนโทรลเลอร์ให้เพิ่มขึ้นครั้งละ 10% ก็จะได้สัญญาณเอาต์พุตเพื่อไปขับไอจีบีทีเพิ่มขึ้นเรื่อยๆ ทำให้สามารถควบคุมความเร็วของมอเตอร์ได้ตามต้องการตามปริมาณแรงดันไฟฟ้าที่จ่ายให้แก่มอเตอร์

4.2 การทดลองตัวตรวจจับกระแส

การทดลองนี้ทำโดยนำตัวตรวจจับกระแส (Current Transducer) มาทดสอบเพื่อสังเกตการเปลี่ยนแปลงของเอาต์พุตเมื่อกระแสไหลมีการเปลี่ยนแปลง ซึ่งทำได้โดยจ่ายกระแสไหลผ่านตัวตรวจจับกระแส (Current Transducer) โดยใช้หลอดไส้เป็นโหลดในการเพิ่มกระแส แล้วใช้ออสซิลโลสโคปวัดรูปคลื่นของเอาต์พุตจากตัวตรวจจับกระแส (Current Transducer) แล้วบันทึกค่าแรงดันที่อ่านได้ลงในตารางบันทึกผลการทดลอง หลังจากนั้นนำค่าที่ได้บันทึกลงในตารางผลการทดลองมาพล็อตกราฟเพื่อดูลักษณะการเปลี่ยนแปลงของกระแสไหลต่อเอาต์พุตของตัวตรวจจับกระแส (Current Transducer) จากการทดลองวัดแรงดันที่ Current Transducer ได้ดังตารางที่ 4.1 เมื่อนำมาพล็อตกราฟจะได้ดังภาพที่ 4.3

ตารางที่ 4.1 แรงดันที่วัดได้จาก Current Transducer

กระแสที่อ่านจากมิเตอร์เข็ม (ampere)	แรงดันที่อ่านจากตัวตรวจจับกระแส (volt)
0	0.016
0.1	0.084
0.2	0.15
0.3	0.218
0.4	0.287
0.5	0.352
0.6	0.418
0.7	0.486
0.8	0.554
0.9	0.624
1	0.69
1.1	0.756
1.2	0.824
1.3	0.89
1.4	0.958
1.5	1.026
1.6	1.087
1.7	1.16
1.8	1.22
1.9	1.288
2	1.361
2.1	1.428
2.2	1.498

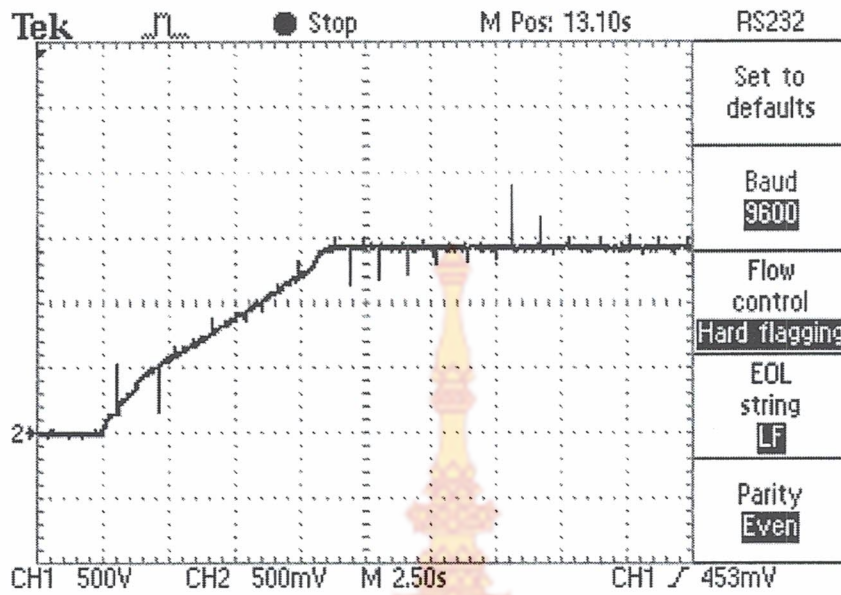


ภาพที่ 4.3 กราฟแรงดันเอาต์พุตที่วัดได้จากตัวตรวจจับกระแส (Current Transducer)

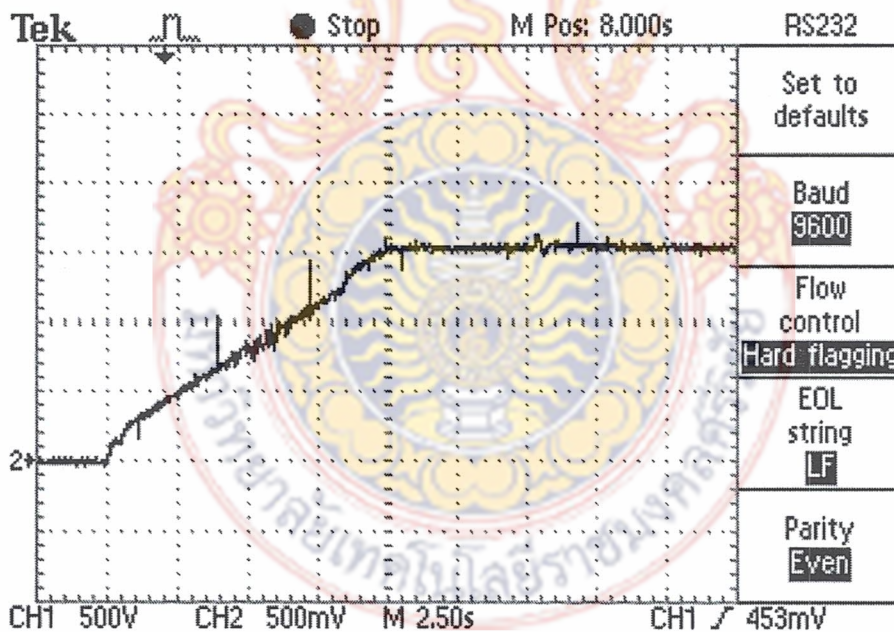
จากการทดลองวัดแรงดันที่ได้จากตัวตรวจจับกระแส (Current Transducer) เมื่อมีกระแสไหลผ่านค่าแรงดันที่ได้จะมีลักษณะที่แปรผันตามค่าของกระแสที่ไหลผ่าน เมื่อนำมาพล็อตกราฟจะได้เป็นเส้นตรง แต่สัญญาณที่วัดได้ไม่ค่อยเรียบ ดังนั้นเมื่อใช้ไมโครคอนโทรลเลอร์อ่านค่าแล้วทำให้ค่าที่อ่านเพื่อตรวจจับปริมาณกระแสที่ไหลเข้าสู่มอเตอร์ เพื่อป้องกันความเสียหายแก่มอเตอร์หากกระแสเกินพิกัด ซึ่งจะเป็นการป้องกันมอเตอร์จากกรณีโหลดเกิน

4.3 การทดลองความเร็วควบคุมความเร็วของมอเตอร์

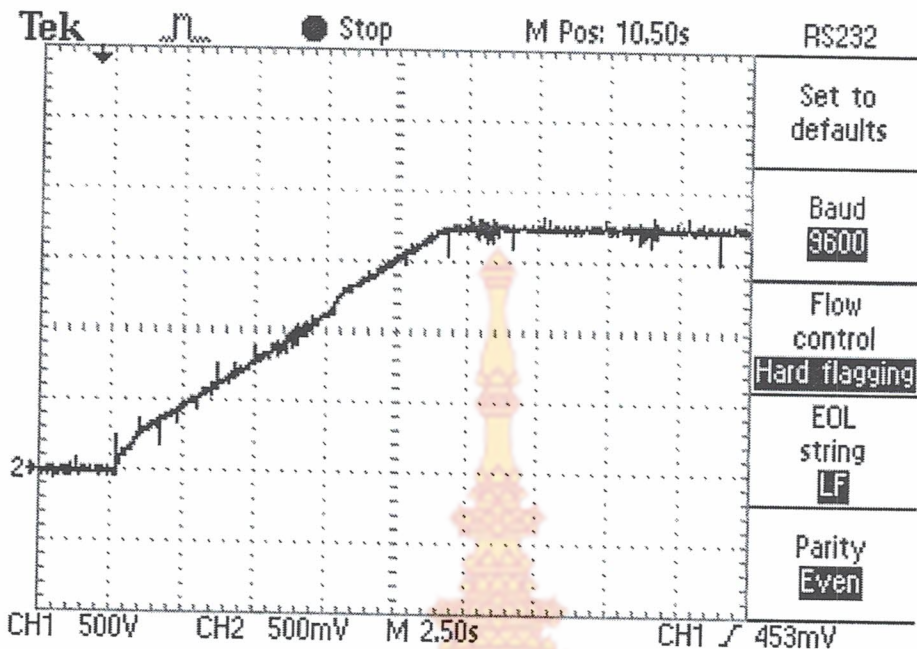
การทดลองนี้ทำโดยนำดีซีมอเตอร์พิกัด 220 V_{DC} 1.4 A 14000 rpm มาทดลองโดยการควบคุมความเร็วในขณะที่ไม่มีโหลด เพื่อดูผลตอบสนองของมอเตอร์เมื่อไม่มีโหลด แล้วใช้ออสซิลโลสโคปวัดรูปคลื่นของเอาต์พุตจากตัวตรวจจับความเร็ว (Tacho Generator) เพื่อดูลักษณะการเริ่มเดินของมอเตอร์และการรักษาความเร็วของมอเตอร์ให้คงที่ ที่ความเร็ว 14000 รอบต่อนาที โดยมีลักษณะของความเร็วดังภาพที่ 4.4



ภาพที่ 4.4 ผลตอบสนองขั้นบันได (Step Response) ที่ความเร็วอ้างอิง 13000 rpm ของระบบควบคุมความเร็วมอเตอร์กระแสตรง



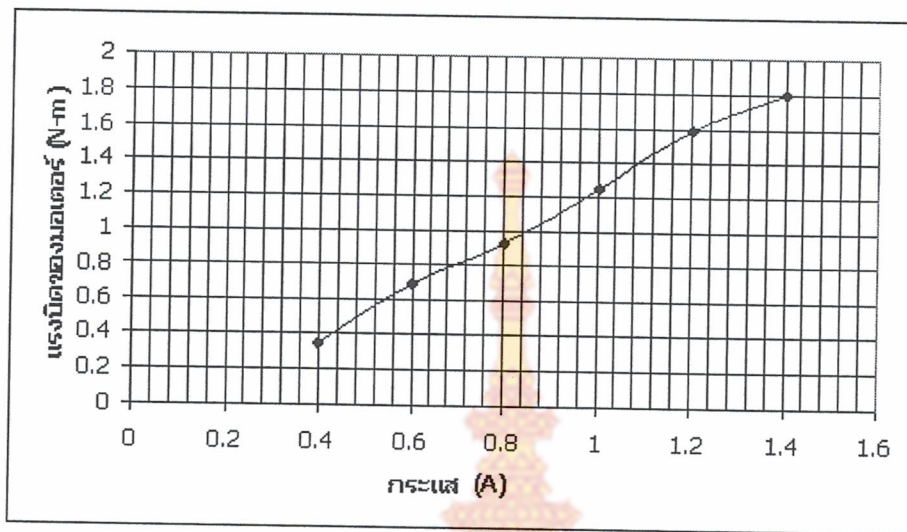
ภาพที่ 4.5 ผลตอบสนองขั้นบันได (Step Response) ที่ความเร็วอ้างอิง 14000 rpm



ภาพที่ 4.6 ผลตอบสนองขั้นบันได (Step Response) ที่ความเร็วอ้างอิง 15000 rpm ของระบบควบคุมความเร็วมอเตอร์กระแสตรง

ตารางที่ 4.2 ผลการทดลอง เมื่อขับที่ความเร็วอ้างอิง 14000 รอบ

Desired Speed (rpm)	Voltage (volt)	Current (ampere)	Field Current (ampere)	Load (N.m)	%Error Speed
14000	223	0.4	0.14	0.28	0.15
	225	0.6		0.65	0.15
	229	0.8		0.95	0.15
	231	1.0		1.25	0.25
	235	1.2		1.51	0.15
	239	1.4		1.81	0.15



ภาพที่ 4.7 กราฟแสดงความสัมพันธ์ระหว่างกระแสอาร์เมเจอร์และแรงบิดของมอเตอร์ ที่ความเร็วอ้างอิง 14000 รอบ/นาที

จากการทดลองเมื่อนำฟuzzyลอจิกคอนโทรลเลอร์ (Fuzzy logic Controller) มาควบคุมความเร็วของมอเตอร์ 220 V 14000 rpm 1.4 A ผลการทดลอง สามารถควบคุมความเร็วของมอเตอร์ให้คงที่ที่ความเร็ว 14000 รอบต่อนาที แต่อาจจะมีมีความคลาดเคลื่อนบ้างเล็กน้อย เมื่อมีโหลดเข้ามา จะทำให้ความเร็วของมอเตอร์ลดลงตามโหลดที่เพิ่มเข้ามา คอนโทรลเลอร์ก็จะทำการควบคุมความเร็วของมอเตอร์ให้อยู่ในค่าความเร็วปกติหรือความเร็วอ้างอิง ถึงแม้ว่าโหลดจะมีการเปลี่ยนแปลงเพิ่มขึ้นหรือลดลง อย่างไรก็ตามฟuzzyลอจิกคอนโทรลเลอร์ ยังมีค่าผิดพลาดเล็กน้อยซึ่งเป็นค่าที่ยังสามารถยอมรับได้

บทที่ 5

สรุปผล

ในงานวิจัยนี้ได้นำเอาไมโครคอนโทรลเลอร์มาเป็นตัวประมวลผลแบบพีซีลจิก คำนวณค่าแรงดันไฟฟ้าที่เหมาะสมเพื่อควบคุมความเร็วของมอเตอร์ไฟฟ้ากระแสตรง เพื่อรักษาความเร็วให้คงที่ที่ 14000 รอบต่อนาที เพื่อเป็นการแยกความเป็นน้ำกับน้ำยางให้ออกจากกันใน ห้องทดลอง เพื่อหาคุณสมบัติของน้ำยางในทางกระบวนการวิทยาศาสตร์ต่อไป โดยมีวงจร ตรวจสอบความเร็วของมอเตอร์ เพื่อนำไปเปรียบเทียบกับความเร็วที่ต้องการ และวงจรควบคุมระดับ แรงดันไฟฟ้ากระแสตรง โดยใช้หลักการของวงจรดีซีชอปเปอร์ ซึ่งจะควบคุมมุมทริกด้วย ซอร์ฟแวร์ซึ่งประมวลผลด้วยหลักการของพีซีลจิก เห็นได้ว่าสามารถรักษาระดับความเร็วของ มอเตอร์ไฟฟ้ากระแสตรง ตามที่กำหนดไว้ได้

ระบบควบคุมที่สร้างขึ้นนี้สามารถควบคุมความเร็วของมอเตอร์ให้คงที่ที่ความเร็ว 14000 รอบต่อนาที แต่อาจจะมีความคลาดเคลื่อนบ้างเล็กน้อย เมื่อมีโหลดเข้ามา จะทำให้ความเร็วของ มอเตอร์ลดลงตามโหลดที่เพิ่มเข้ามา คอนโทรลเลอร์ก็จะทำการควบคุมความเร็วของมอเตอร์ให้อยู่ ในค่าความเร็วปกติหรือความเร็วอ้างอิง ถึงแม้โหลดจะมี การเปลี่ยนแปลงเพิ่มขึ้นหรือ ลดลง อย่างไรก็ตามพีซีลจิกคอนโทรลเลอร์ ยังมีค่าผิดพลาดเล็กน้อยซึ่งเป็นค่าที่ยังสามารถ ยอมรับได้



บรรณานุกรม

- [1] ตำราเรียนรู้ด้วยตนเอง อิเล็กทรอนิกส์กำลัง. โครงการความร่วมมือทางวิชาการระหว่างสถาบันเทคโนโลยีราชมงคลกับสถาบันเทคโนโลยีพระจอมเกล้าพระนครเหนือ.
กรุงเทพมหานคร : สำนักพัฒนาเทคนิคศึกษา สถาบันเทคโนโลยีพระจอมเกล้าพระนครเหนือ.
- [2] ธวัชชัย อรรถวิบูลย์กุล. เครื่องกลไฟฟ้า 3. กรุงเทพมหานคร : ศูนย์ส่งเสริมอาชีวะ, 2527.
- [3] ประจัน พลังสันติกุล. เรียนรู้และใช้งาน CCS C คอมไพเลอร์ เขียนโปรแกรมภาษา C ควบคุมไมโครคอนโทรลเลอร์ PIC. กรุงเทพมหานคร : บริษัท อินโนโวลทิฟ เอ็กเพอริเมนต์ จำกัด.
- [4] พีซีซีลอจิกคอนโทรล. [ออนไลน์] เข้าถึงได้จาก <http://www.geocities.com/pisitp/logic.htm>




```

#define _PIC16F877_           // Use PIC16F877 device

#include <16F877.h>           // Standard Header file for the PIC16F877 device
#define TxD    PIN_C6        // Define Transmitted Data
#define RxD    PIN_C7        // Define Received Data
#define CLOCK_SP 2000000    // Clock Speed(Hz)
#define MAX_DATA_EEPROM 256 // data EEPROM in PIC16F877
#define RA0_RA1_ANALOG 0x84
#define device ADC=10
#define use fast_io(A)
#define Vbe 0.0048875855327468230694037145650049
#define fuses HS              // Oscillator mode HS
#define fuses NOLVP, NOWDT    // No Low Voltage Program, No Watchdog timer
#define fuses NOPROTECT       // Code no protection
#define use delay (clock=CLOCK_SP) // Use built-in function: delay_ms() & delay_us()
#define use rs232(baud=9600, xmit=TxD,rcv=RxD) // Use serial I/O port (RS232)
#define use_porth_lcd
#include "lcd.c"
#define STRING_SIZE 17
/*****
* FUNCTION: Main
* DESCRIPTION: This is the main entry point for the program.
* PARAMETERS: nothing
* RETURNED: nothing
*****/

int a,i,c,v,d,e;
char MSG[17];
int16 value;
int16 val;

```

```

float volt,amp,CUR;
float y,ERROR,ERRORCH,error_old,TEST,Vref;
int8 err,errch,position,output;
int8 UT;
void Fuzzy(void);
void data_fuzzy (void);
void WriteIntEEPROM(byte addr,byte value10);
void LCD_Command (int cm);
void strcpy (char *s1 ,char *s2);
void LCD_String(char *s);
void LCD_ShiftLeft(void);
void LCD_String1(char *s, int dly);
void OVER_CURRENT(void);
void DATA_MOTOR(void);
void DATA_CURRENT(void);
void READ_CURRENT(void);
void PRESS(void);
void SETSPEED(int a);
void SHOW(void);
void SHOW(void){
if (i<50){
    if (Vref<1){
        LCD_Command (0x80);
        strcpy(MSG,"S.SET = 0 RPM");
        LCD_String(MSG);
        sprintf(MSG,"%3.0f",((Vref*100)+1-1));
        lcd_gotoxy(8,1);
        LCD_String(MSG);
    }
}

```

```
else{
LCD_Command (0x80);
strcpy(MSG,"S.SET = 0 RPM");
LCD_String(MSG);
sprintf(MSG,"%4.0f",((Vref*100)+1-1));
lcd_gotoxy(8,1);
LCD_String(MSG);
}
}
else
{

LCD_Command (0x80);
strcpy(MSG,"CURRENT= AMP");
LCD_String(MSG);
sprintf(MSG,"%1.1f",(amp));
lcd_gotoxy(9,1);
LCD_String(MSG);
}

LCD_Command (0xc0);
strcpy(MSG,"SPEED = RPM");
LCD_String(MSG);
sprintf(MSG,"%4.0f", (volt*1000));
lcd_gotoxy(8,2);
LCD_String(MSG);
}
}
```

```
/****** // DATA// *****/
```

```
void data_fuzzy (void){
```

```
int i;
```

```
for (i=11; i<=16; i++){
```

```
write_eeprom(i,52);
```

```
}
```

```
for (i=17; i<=21; i++){
```

```
write_eeprom(i,46);
```

```
}
```

```
for (i=22; i<=27; i++){
```

```
write_eeprom(i,52);
```

```
}
```

```
for (i=28; i<=32; i++){
```

```
write_eeprom(i,46);
```

```
}
```

```
for (i=33; i<=35; i++){
```

```
write_eeprom(i,52);
```

```
}
```

```
for (i=36; i<=38; i++){
```

```
write_eeprom(i,49);
```

```
}
```

```
for (i=39; i<=43; i++){
```

```
write_eeprom(i,43);
```

```
}
```

```
for (i=44; i<=46; i++){
```

```
write_eeprom(i,52);
```

```
}
```



```
for (i=47; i<=49; i++){
write_eeprom(i,46);
}
for (i=50; i<=54; i++){
write_eeprom(i,0);
}
for (i=55; i<=57; i++){
write_eeprom(i,49);
}
for (i=58; i<=59; i++){
write_eeprom(i,46);
}
for (i=60; i<=60; i++){
write_eeprom(i,43);
}
for (i=61; i<=62; i++){
write_eeprom(i,0);
}
for (i=63; i<=65; i++){
write_eeprom(i,43);
}
for (i=66; i<=70; i++){
write_eeprom(i,46);
}
for (i=71; i<=73; i++){
write_eeprom(i,0);
}
for (i=74; i<=76; i++){
write_eeprom(i,6);
}
}
```



```
for (i=77; i<=81; i++){
write_eeprom(i,43);
}
for (i=82; i<=84; i++){
write_eeprom(i,3);
}
for (i=85; i<=87; i++){
write_eeprom(i,9);
}
for (i=88; i<=92; i++){
write_eeprom(i,0);
}
for (i=93; i<=95; i++){
write_eeprom(i,6);
}
for (i=96; i<=98; i++){
write_eeprom(i,12);
}
for (i=99; i<103; i++){
write_eeprom(i,3);
}
for (i=104; i<=106; i++){
write_eeprom(i,9);
}
for (i=107; i<=109; i++){
write_eeprom(i,12);
}
```



```

for (i=110; i<=114; i++){
write_eprom(i,6);
}
for (i=115; i<=120; i++){
write_eprom(i,12);
}
for (i=121; i<=125; i++){
write_eprom(i,6);
}
for (i=126; i<=131; i++){
write_eprom(i,12);
}
}

/***** // Fuzzy *****/

void Fuzzy(void){

if (ERROR>=45)   err=11;           // 5
else if ((ERROR>=35)&&(ERROR<45))   err=10;           // 4
else if ((ERROR>=25)&&(ERROR<35))   err=9;            // 3
else if ((ERROR>=15)&&(ERROR<25))   err=8;            // 2
else if ((ERROR>=5)&&(ERROR<15))    err=7;            // 1
else if ((ERROR>=-5)&&(ERROR<5))    err=6;            // 0
else if ((ERROR>=-15)&&(ERROR<-5))  err=5;           // -1
else if ((ERROR>=-25)&&(ERROR<-15)) err=4;           // -2
else if ((ERROR>=-35)&&(ERROR<-25)) err=3;           // -3
else if ((ERROR>=-45)&&(ERROR<-35)) err=2;           // -4
else   err=1;                       // -5

```

```
if (ERRORCH>=45)  errch=10;           // 5
else if ((ERRORCH>=35)&&(ERRORCH<45))  errch=9;    // 4
else if ((ERRORCH>=25)&&(ERRORCH<35))  errch=8;    // 3
else if ((ERRORCH>=15)&&(ERRORCH<25))  errch=7;    // 2
else if ((ERRORCH>=5)&&(ERRORCH<15))   errch=6;    // 1
else if ((ERRORCH>=-5)&&(ERRORCH<5))    errch=5;    // 0
else if ((ERRORCH>=-15)&&(ERRORCH<-5))  errch=4;    // -1
else if ((ERRORCH>=-25)&&(ERRORCH<-15)) errch=3;    // -2
else if ((ERRORCH>=-35)&&(ERRORCH<-25)) errch=2;    // -3
else if ((ERRORCH>=-45)&&(ERRORCH<-35)) errch=1;    // -4
else  errch=0;                          //- 5
```

```
position=((err*11)+errch);
output=read_eeprom(position);
```

```
if (output==12) { UT=(UT+7); }
if (output==9)  { UT=(UT+3); }
if (output==6)  { UT=(UT+1); }
if (output==3)  { UT=(UT); }
if (output==0)  { UT=(UT); }
if (output==43) { UT=(UT); }
if (output==46) { UT=(UT-1); }
if (output==49) { UT=(UT-3); }
if (output==52) { UT=(UT-7); }

}
```

```
void PRESS(void){
i=0;
for (i=0;i<5;i++){
```



```
if (!input(PIN_E1)) {
    set_pwm1_duty(200);
    LCD_Command (0x01);
    e=1;
    break;
}

LCD_Command (0x01);
delay_ms(250);
LCD_Command (0x80);
strcpy(MSG,"READY!!!!");
lcd_gotoxy(5,1);
LCD_String(MSG);
LCD_Command (0xC0);
strcpy(MSG,"PRESS RUN ");
lcd_gotoxy(5,2);
LCD_String(MSG);
if (!input(PIN_E0)) {
    break;}
delay_ms(250);
LCD_Command (0x01);
i=0;}
}

void READ_CURRENT(void){

    if ((amp)>CUR){
        OVER_CURRENT();
    }
}
```

```
void LCD_Command (int cm) {  
    lcd_send_byte (0, cm);  
}
```

```
void strcpy (char *s1 ,char *s2){  
    while(*s1++ = *s2++);  
}
```

```
void LCD_String(char *s){  
    while(*s!=0) {  
  
        lcd_putc(*s++);  
    }  
}
```

```
void LCD_ShiftLeft(void) {  
  
    lcd_send_byte(0,0x18);  
}
```

```
void LCD_String1(char *s, int dly) {  
    while(*s!=0) {  
        if (!input(PIN_A2)) {  
            break;}  
            lcd_putc(*s++);  
            delay_ms(dly);  
    }  
}
```



```
void WriteIntEEPROM(byte addr,byte value10){
write_eeprom(addr,value10); // write eeprom
}
```

```
void sendlcd (void){

    lcd_gotoxy(6,1);
    LCD_String(MSG);}
```

```
void SETSPEED(int a){
int8 b,ss;
```

```
/****** // ADJUST VREF *****/
```

```
LCD_Command (0x01);
LCD_Command (0x80);
strcpy(MSG,"SET S.= 0 RPM");
LCD_String(MSG);
```

```
sprintf(MSG,"%4.0f",((Vref*100)+1-1));
lcd_gotoxy(8,1);
LCD_String(MSG);
```

```
delay_ms(200);
```

```
while (a==3){
```

```
    if (!input(PIN_A4)) {
        delay_ms(200);
```

```
if (v==1 && Vref>0.99) {
    Vref=1;}
if (v==2 && Vref>1.99) {
    Vref=2;}
if (v==3 && Vref>2.99) {
    Vref=3;}
if (v==4 && Vref>3.99) {
    Vref=4;}
if (v==5 && Vref>4.99) {
    Vref=5;}
else
    { Vref=Vref+0.01; }
sprintf(MSG,"%4.0f",((Vref*100)+1-1));
lcd_gotoxy(8,1);
LCD_String(MSG);
if ((Vref)<1.0){
    strcpy(MSG,"0 RPM");
    lcd_gotoxy(11,1);
    LCD_String(MSG);
}
}

if (!input(PIN_A5)) {
    delay_ms(200);
    if (Vref<0.001) {
        Vref=0;}
    else
        { Vref=Vref-0.01; }
    sprintf(MSG,"%4.0f",((Vref*100)+1-1));
```

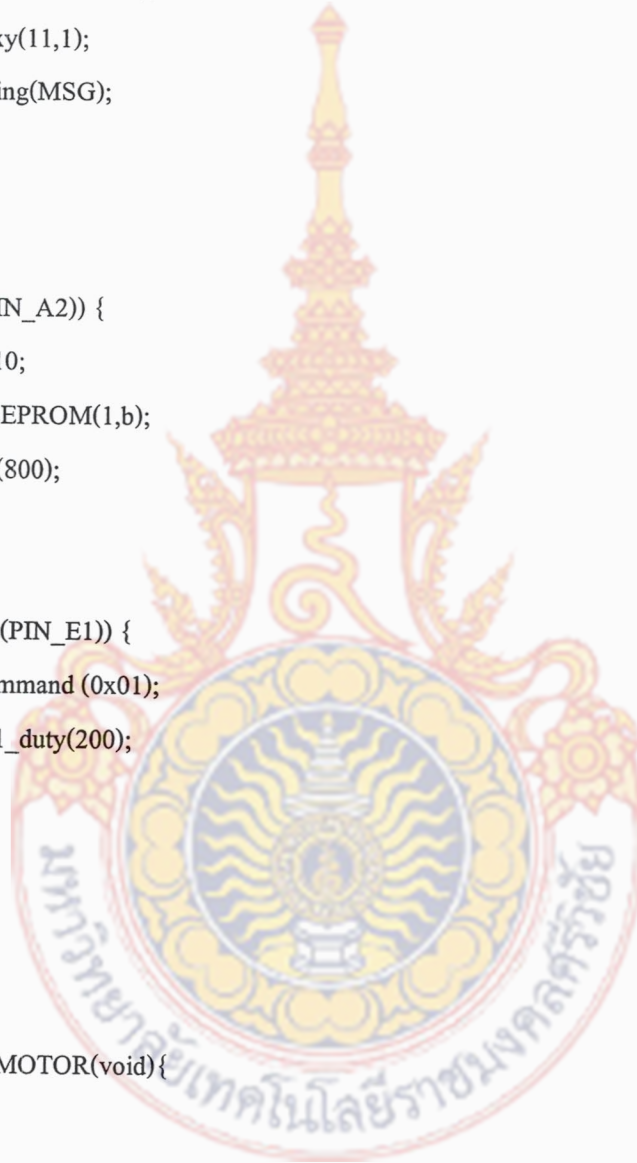


```
lcd_gotoxy(8,1);
LCD_String(MSG);

if ((Vref)<1.0){
strcpy(MSG,"0 RPM");
lcd_gotoxy(11,1);
LCD_String(MSG);
}
}

if (!input(PIN_A2)) {
b= Vref*10;
WriteIntEEPROM(1,b);
delay_ms(800);
break;
}
if (!input(PIN_E1)) {
LCD_Command (0x01);
set_pwm1_duty(200);
e=1;
break;
}
}
}

void DATA_MOTOR(void){
int8 z1,j;
z1=0;
v = read_eeprom(5);
LCD_Command (0x01);
LCD_Command (0x80);
```



```
strcpy(MSG,"MAX . SPEED");
LCD_String(MSG);
LCD_Command (0xC0);
strcpy(MSG,"===== 1000 RPM");
LCD_String(MSG);

while(z1==0) {

if (!input(PIN_E1)) {
    LCD_Command (0x01);
    set_pwm1_duty(200);
    e=1;
    break;
}

if (!input(PIN_A4)) {
    delay_ms(200);
    v=v+1;
    if (v>4){v=5;}
}

if (!input(PIN_A5)) {
    delay_ms(200);
    v=v-1;
    if (v<2){v=1;}
}

if(v==1){
    delay_ms(100);
    LCD_Command (0xC0);
    strcpy(MSG,"===== 1000 RPM");
    LCD_String(MSG);}
```



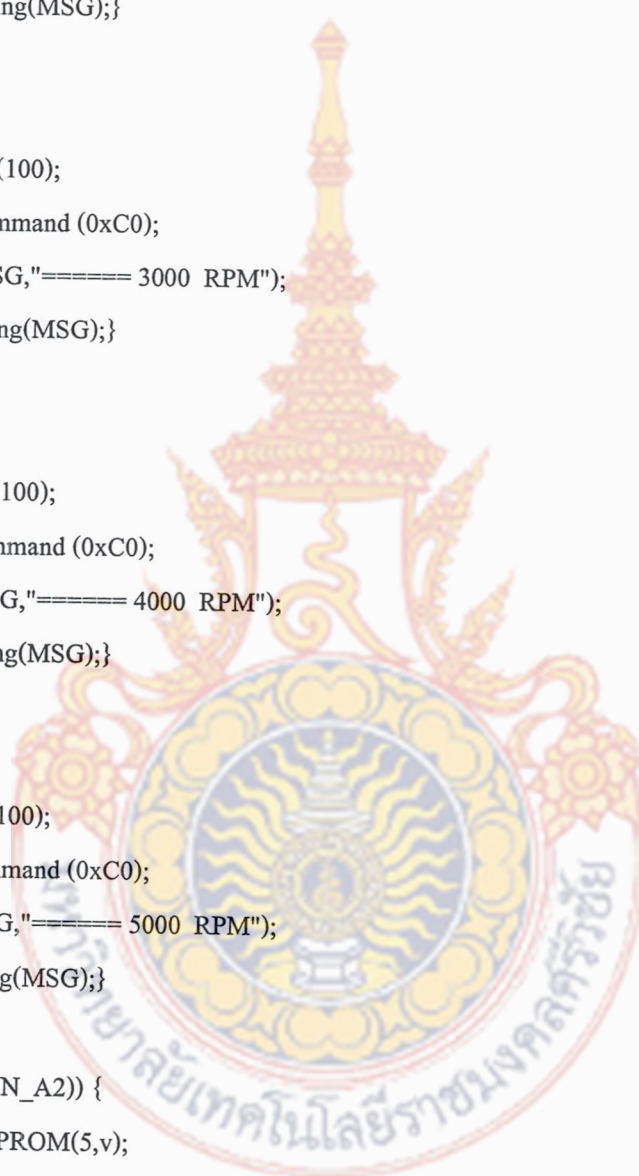
```
if(v==2){  
    delay_ms(100);  
    LCD_Command (0xC0);  
    strcpy(MSG,"===== 2000 RPM");  
    LCD_String(MSG);}
```

```
if(v==3){  
    delay_ms(100);  
    LCD_Command (0xC0);  
    strcpy(MSG,"===== 3000 RPM");  
    LCD_String(MSG);}
```

```
if(v==4){  
    delay_ms(100);  
    LCD_Command (0xC0);  
    strcpy(MSG,"===== 4000 RPM");  
    LCD_String(MSG);}
```

```
if(v==5){  
    delay_ms(100);  
    LCD_Command (0xC0);  
    strcpy(MSG,"===== 5000 RPM");  
    LCD_String(MSG);}
```

```
if(!input(PIN_A2)) {  
    WriteIntEEPROM(5,v);  
    break;}  
}  
}
```



```
void DATA_CURRENT(void){
int8 z1,j;
z1=0;
v=1;
LCD_Command (0x80);
strcpy(MSG,"CURRENT PROTECT");
LCD_String(MSG);
LCD_Command (0xC0);
strcpy(MSG,"CURRENT=  A");
LCD_String(MSG);
sprintf(MSG,"%3.1f", CUR );
lcd_gotoxy(10,2);
LCD_String(MSG);

while(z1==0) {
if (!input(PIN_A4)) {
delay_ms(200);
if (CUR<5)
CUR=CUR+0.1;
else
CUR=5;
sprintf(MSG,"%3.1f", CUR);
lcd_gotoxy(10,2);
LCD_String(MSG);
}
if (!input(PIN_A5)) {
delay_ms(200);
if (CUR>0)
CUR=CUR-0.1;
else
```



```
    CUR=0;
    sprintf(MSG,"%3.1f", CUR);
    lcd_gotoxy(10,2);
    LCD_String(MSG);

}

if (!input(PIN_A2)) {
    j=CUR*10;
    WriteIntEEPROM(2, j);
    delay_ms(100);
    break;
}

if (!input(PIN_E1)) {
    set_pwm1_duty(200);
    LCD_Command (0x01);
    e=1;
    break; }
}

void OVER_CURRENT(void){
int8 j;
    UT= 0 ;
    output_low(pin_D2);
    set_pwm1_duty(200-(UT)); // Clear PWM_Val

for (j=0;j<5;j++){
```

```
    if (!input(PIN_E1)) {
        LCD_Command (0x01);
        e=1;
        break;
    }
    output_high(pin_D1);
    LCD_Command (0x01);
    LCD_Command (0x80);
    strcpy(MSG,"OVER CURRENT!!");
    lcd_gotoxy(3,1);
    LCD_String(MSG);
    delay_ms(500);

    output_low(pin_D1);
    LCD_Command (0x01);
    j=1;
    delay_ms(1000);
}
}

/***** //RUN *****/

void main(void) {

    int8 j,w;

    data_fuzzy();
    delay_ms(200);

    labelSTOP:
```

```
lcd_init();
value=0;
val=0;
i=0;
c=0;
e=0;

Vref=(float)(read_eeprom(1))/10;
CUR= (float)(read_eeprom(2))/10;

set_tris_a(0xFF); // All inputs
set_tris_e(0xFF); // All inputs
set_tris_d(0x00); // All outputs
output_d(0x00);
output_high(pin_D0);
output_low(pin_D2);

setup_adc_ports( RA0_RA1_ANALOG);
setup_adc(ADC_CLOCK_INTERNAL);
setup_ccp1(CCP_PWM); // Configure CCP1 as a PWM
setup_timer_2(T2_DIV_BY_4, 202, 1); // 1.2kHz

UT= 0 ;
set_pwm1_duty(200); // Clear PWM_Val

for(i=0; i<2; i++) {

strcpy(MSG,"FUZZY CONTROLLER ");
LCD_String1(MSG,100);
LCD_Command (0xC0);
```

```
strcpy(MSG," PRESS ENTER ");
LCD_String1(MSG,100);
LCD_Command (0x01);
i=0;
if (!input(PIN_A2)) {
i=3;
break;}
}
delay_ms(800);

DATA_CURRENT();

if (e==1){
goto labelSTOP;}
delay_ms(500);
DATA_MOTOR();

if (e==1){
goto labelSTOP;}

a = SETSPEED(3);

if (e==1){
goto labelSTOP;}

PRESS();

if (e==1){
goto labelSTOP;}
LCD_Command (0x01);
```



```
while (input(PIN_E0)); { // RUN

    label: // interrupt manu

    if (e==1){
        goto labelSTOP;}
    output_low(pin_D0);
    output_low(pin_D1);
    output_high(pin_D2);

    for(i=0; i<100; i++) {

        SHOW();

        LCD_Command (0xc0);
        strcpy(MSG,"SPEED = RPM");
        LCD_String(MSG);
        sprintf(MSG,"%4.0f", (volt*1000));
        lcd_gotoxy(8,2);
        LCD_String(MSG);

        set_adc_channel(0);
        delay_ms(50);
        value = Read_ADC();
        volt = Vbe * (float)value;

        error_old = ERROR;
        ERROR = (Vref*1000) - (volt*1000) ;
        ERRORCH = ERROR - error_old;
```

```
Fuzzy();

if (UT > 200){
  UT=200;}

set_pwm1_duty(202-UT);//UT

if (!input(PIN_A2)) {
  a = SETSPEED(3);
}

if (!input(PIN_E1)) {
  output_high(pin_D0);
  UT=(Vref*1000);
  goto labelSTOP;
}

set_adc_channel(1); // CHECK CURRENT
delay_ms(50);
val = Read_ADC();
amp = (((Vbe * (float)val)*1.443127));

READ_CURRENT);

if (i==98){
  i=1;}

}}}
```

