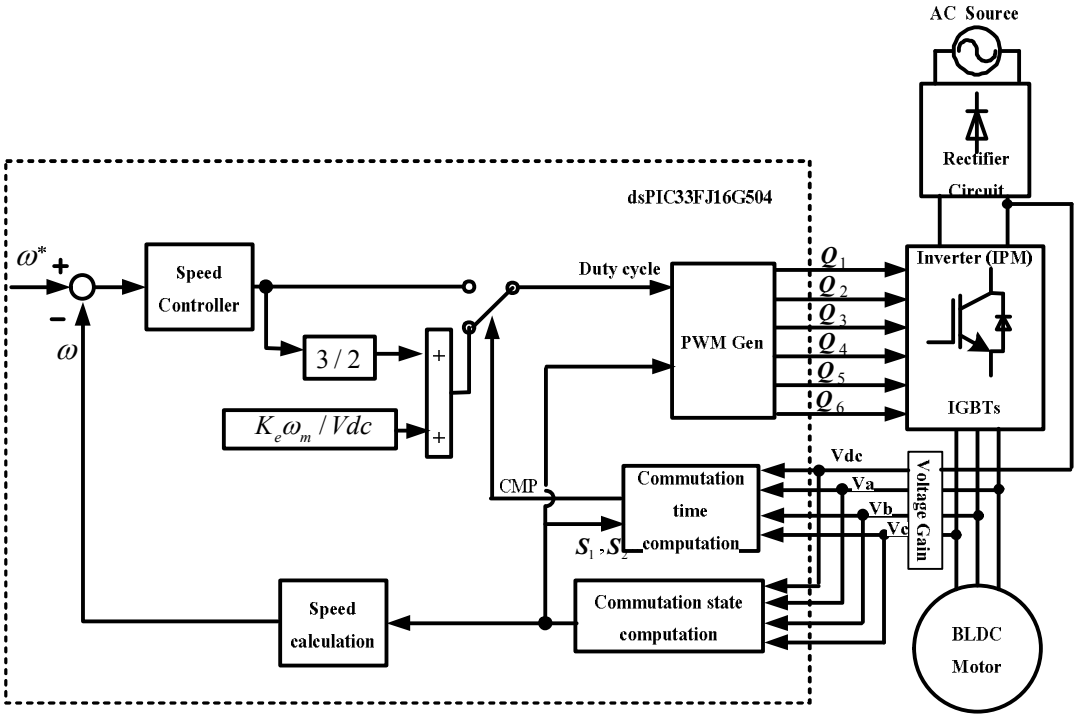
บทที่ 3 ขั้นตอนและวิธีการดำเนินงาน

ในบทที่ 3 กล่าวถึงขั้นตอนและวิธีการดำเนินงานวิจัย ซึ่งประกอบด้วย 2 ส่วนคือ ส่วนประกอบของซอฟแวร์และฮาร์ดแวร์ โครงสร้างและการทำงานของโดยรวมของการขับเคลื่อนมอเตอร์ไฟฟ้ากระแสตรงไร้แปรงถ่านแบบไร้ตัวตรวจจับตำแหน่งพร้อมทั้งลดการกระเพื่อมแรงบิดมอเตอร์ที่นำเสนอแสดงดังรูปที่ 3.1



รูปที่ 3.1 โครงสร้างและการทำงานของโดยรวมของการขับเคลื่อนมอเตอร์ไฟฟ้ากระแสตรงไร้แปรงถ่านแบบไร้ตัวตรวจจับตำแหน่งพร้อมทั้งลดการกระเพื่อมแรงบิดมอเตอร์ที่นำเสนอ

3.1 ส่วนประกอบของซอฟแวร์

ส่วนประกอบซอฟต์แวร์สำหรับงานวิจัยนี้แบ่งออกเป็น 2 ส่วน คือ 1) ซอฟต์แวร์สำหรับจำลองการทำงานของการขับเคลื่อนมอเตอร์ซึ่งถูกพัฒนาขึ้นด้วยภาษาซี (C code) โดยใช้โปรแกรม Microsoft Visual Studio 2005 และใช้โปรแกรม PSIM (Version 9.0.3) เพื่อจำลองผลการทำงานของมอเตอร์ และ 2) ซอฟต์แวร์สำหรับพัฒนาบนไมโครคอนโทรลเลอร์เบอร์ dsPIC33FJ16GS504 ในบอร์ดทดลองขับเคลื่อนมอเตอร์ โดยใช้โปรแกรม MPLAB Version 3.1 ที่ติดตั้งอยู่ภายในคอมพิวเตอร์เพื่อทำการเขียนชุดคำสั่งไปยังบอร์ดควบคุมที่มีตัวไมโครคอนโทรลเลอร์ติดตั้งอยู่ โดยมี REAL ICE เป็นตัวเชื่อมต่อระหว่างคอมพิวเตอร์และบอร์ดควบคุม

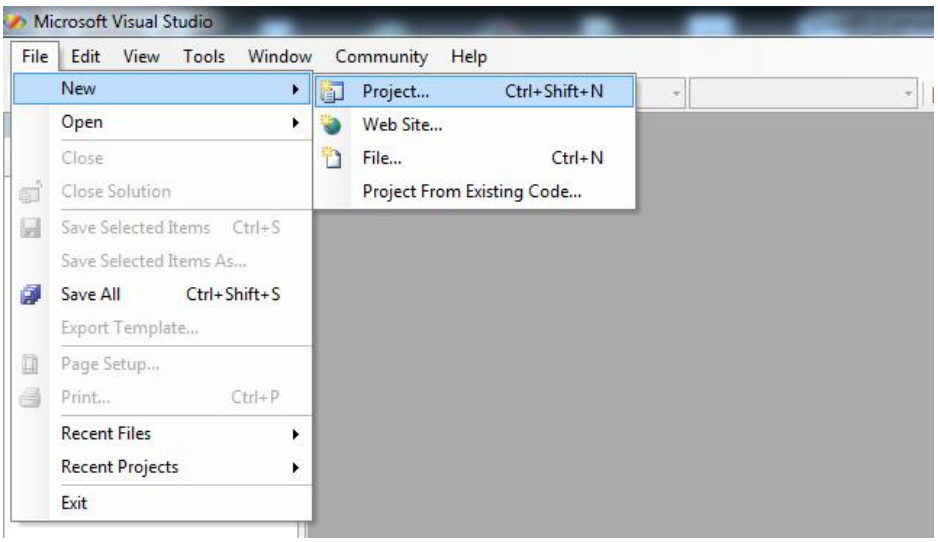
3.1.1 ซอฟต์แวร์สำหรับจำลองการทำงานของระบบขับเคลื่อนมอเตอร์

1. โปรแกรม Microsoft Visual Studio 2005

Microsoft Visual Studio 2005 คือ ชุดโปรแกรมที่ถูกพัฒนาขึ้นโดยบริษัทไมโครซอฟท์ ซึ่งเป็นเครื่องมือที่ช่วยนักพัฒนาซอฟต์แวร์พัฒนาโปรแกรมคอมพิวเตอร์ โดยในงานวิจัยนี้ได้เลือกใช้ภาษา C เพื่อทำการสร้างไลบรารีการเชื่อมโยงแบบพลวัต (DLL) เพื่อเป็นไฟล์สำหรับเชื่อมต่อกับโปรแกรม PSIM ในการจำลองการทำงานการขับเคลื่อนมอเตอร์ต่อไป ข้อดี สามารถใช้ชุดคำสั่งภาษา C ที่พัฒนาบนแบบจำลองกับไมโครคอนโทรลเลอร์ได้เลย ซึ่งช่วยลดเวลาในการพัฒนา

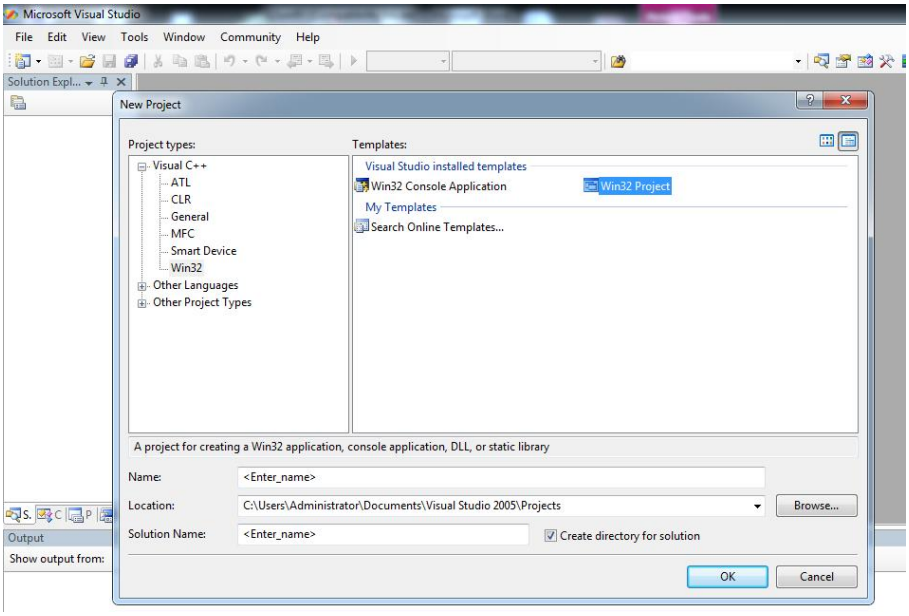
วิธีการสร้างโปรแกรมควบคุม

1) เปิดโปรแกรม Visual Studio 2005 เพื่อสร้างโปรเจคใหม่ ดังรูปที่ 3.2



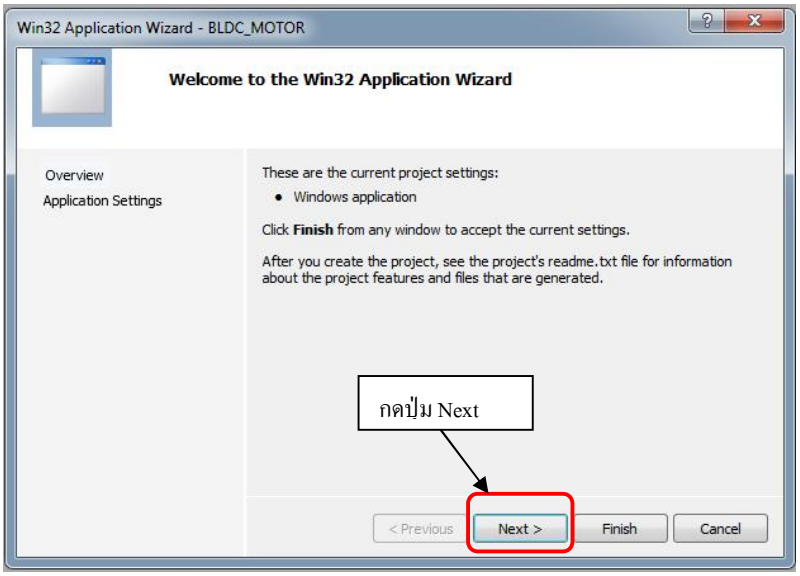
รูปที่ 3.2 วิธีการสร้างโปรเจคใหม่ในโปรแกรม Visual Studio 2005

2) กำหนดชนิดของโปรแกรมประเภท Win32 Project พร้อมทั้งตั้งชื่อ และกำหนดที่อยู่เก็บไฟล์โปรเจค ดังรูปที่ 3.3

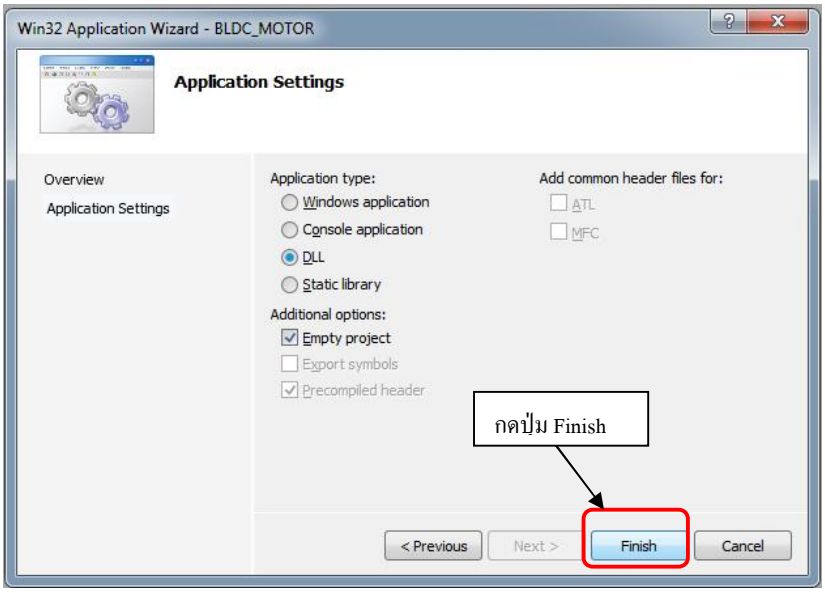


รูปที่ 3.3 การตั้งค่าโปรเจคในโปรแกรม Visual Studio 2005

3) ทำการกำหนดค่าโปรเจค Win32 เป็นชนิด DLL (Dynamic-link library) ดังรูปที่ 3.4-3.5

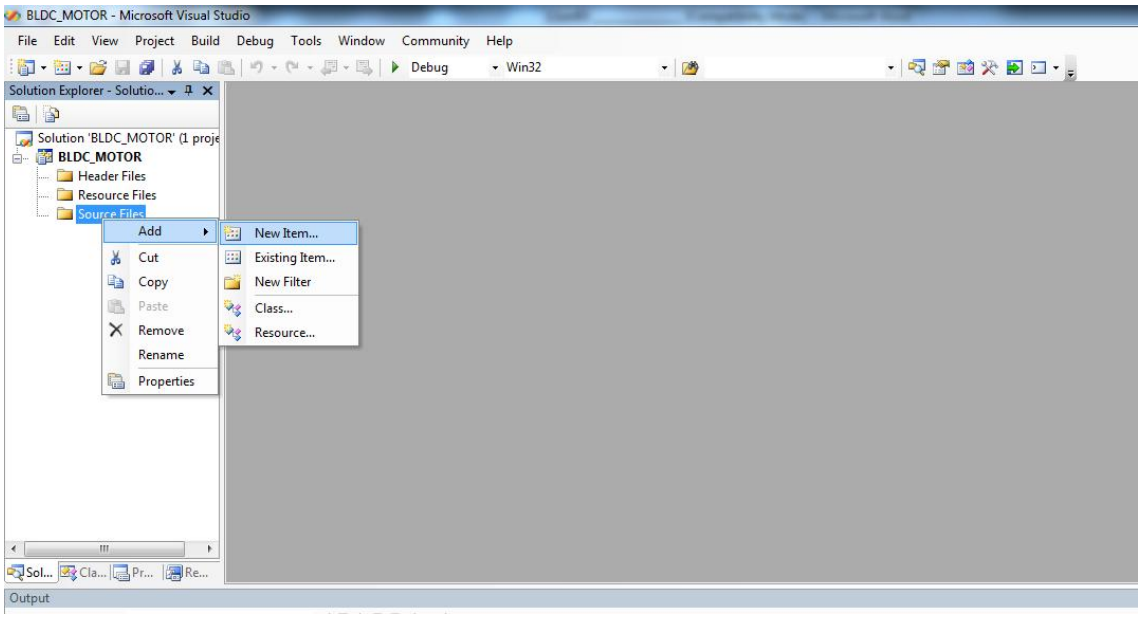


รูปที่ 3.4 การตั้งค่าโปรเจค Win32 ในโปรแกรม Visual Studio 2005

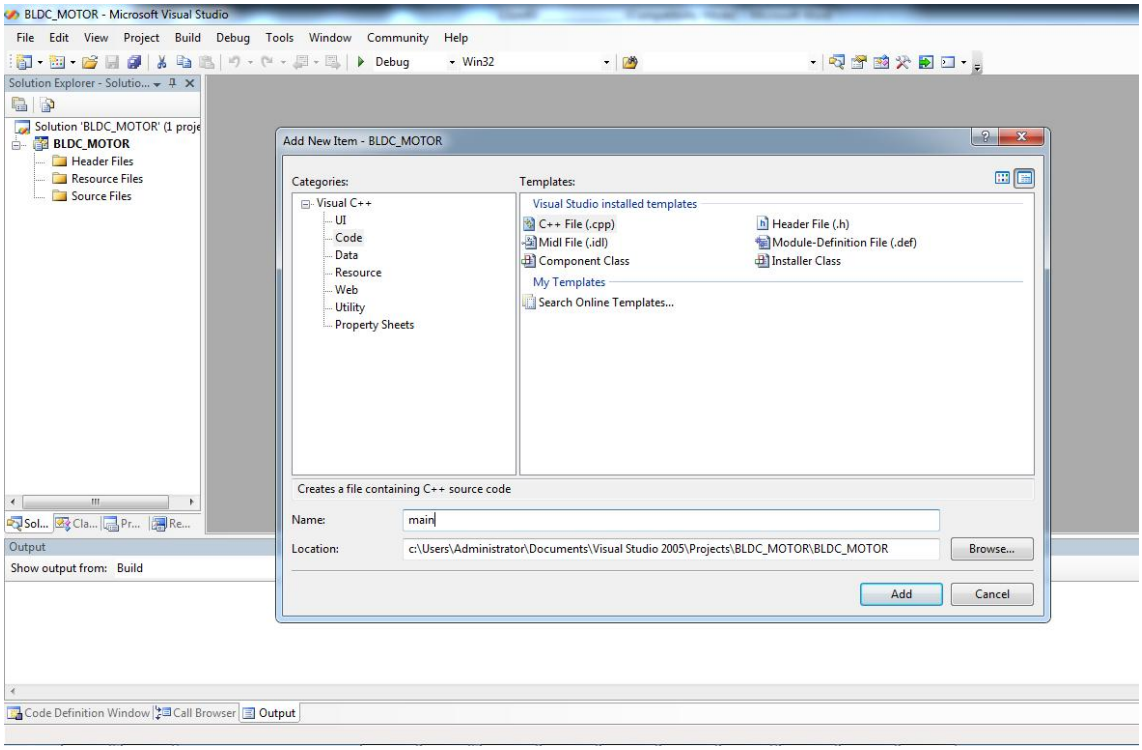


รูปที่ 3.5 การตั้งค่าโปรเจ็ค Win32 ชนิด DLL ในโปรแกรม Visual Studio 2005

4) ทำการเพิ่มไฟล์โดยคลิกขวาที่ Source Files จะปรากฏหน้าต่าง ดังแสดงรูปที่ 3.6 หลังจากนั้นกดปุ่ม New Item เพื่อสร้างไฟล์ main.cpp เพื่อใช้เป็นไฟล์หลักในการเขียนชุดคำสั่งภาษาซี (Source Code) ดังแสดงรูปที่ 3.7



รูปที่ 3.6 การเพิ่มไฟล์โค๊ดเข้าไปในโปรแกรม



รูปที่ 3.7 การสร้างไฟล์โค๊ดภาษาซี

ไฟล์ main.cpp เป็นโปรแกรมส่วนหลัก ซึ่ง Source Code ที่ใช้ต้องเขียนอยู่ภายในฟังก์ชัน simuser เพื่อเชื่อมต่อกับโปรแกรม PSIM ดังต่อแสดงรูปที่ 3.8

```
__declspec(dllexport) void simuser (double t, double delt, double *in, double
*out)
{
// Place your code here.....begin

// Place your code here.....end
}
```

รูปที่ 3.8 การสร้างฟังก์ชัน simuser

โดยที่ __declspec(dllexport) void simuser (double t, double delt, double *in, double *out) เป็นฟังก์ชันมาตรฐานที่กำหนดโดยโปรแกรม PSIM ไม่สามารถเปลี่ยนชื่อได้ โดยส่งผ่านค่าต่างๆผ่านตัวแปร ดังนี้

t คือ เวลาที่ผ่านไปจากโปแกรม PSIM

delt คือ สเต็ปเวลาของโปรแกรม PSIM

in คือ อาร์เรย์ป้อนข้อมูลจากโปรแกรม PSIM

out คือ อาร์เรย์เอาต์พุตส่งกลับไปยังโปรแกรม PSIM

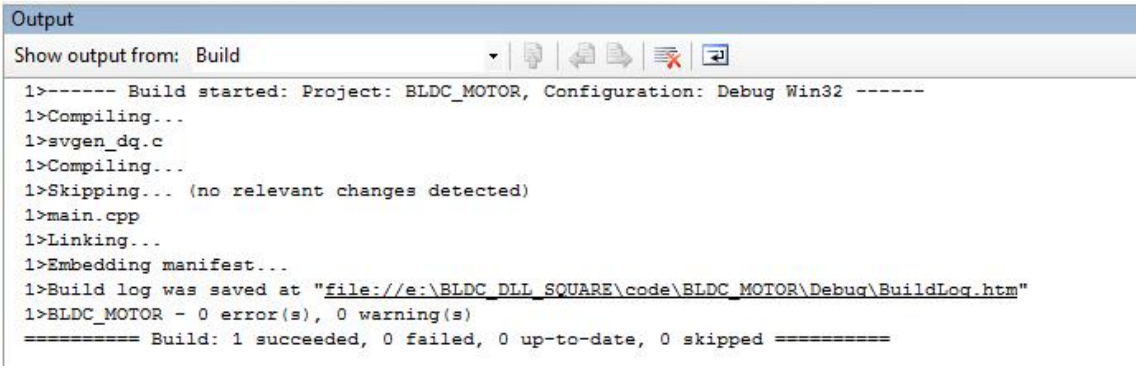
```
#include <math.h>
__declspec(dllexport) void simuser (double t, double delt,
double *in, double *out)
{
	static int Ncount = 0, flagSample=1, count= 0;
	Ncount=50e-6/delt; //คำนวณจำนวนครั้งในหนึ่งคาบ
	if (count == Ncount)
	{
		flagSample=1;
		count=0;  //Reset the counter to 0.
	}

if (flagSample == 1)	{
		flagSample=0;  //Reset the sampling flag
		//โปรแกรมที่ทำใช้สำหรับการขับเคลื่อนมอเตอร์ไฟฟ้ากระแสตรงแบบไร้แปรงถ่าน
```

รูปที่ 3.9 ตัวอย่างโปรแกรมกำหนดเวลาการเกิดอินเตอร์รัปต์

จากรูปที่ 3.9 แสดงตัวอย่างโปรแกรมที่กำหนดเวลา การชักสัญญาณ หรือการเกิดอินเตอร์รัป (Interrupt) โดยในงานวิจัยนี้ ได้กำหนดความถี่อินเตอร์รัป 20 kHz หรือ คาบเวลาเท่ากับ 50 µs และ PSIM มีความละเอียดในแต่ละสเต็ป 1 µs โดยมีตัวแปร Ncount เป็นตัวกำหนดจำนวนครั้ง ซึ่งจะได้จำนวนครั้ง 20 ครั้งที่ผ่านต่อการเกิดอินเตอร์รัป 1 ครั้ง ซึ่งส่วนของโปรแกรมขับเคลื่อนมอเตอร์ก็จะทำงานอยู่ภายในอินเตอร์รัปดังกล่าว

5) ทำการเขียนSource Code โปรแกรมเสร็จสมบูรณ์หลังจากนั้นทำการคอมไพล์โปรแกรมเพื่อตรวจสอบความถูกต้อง โดยกดที่ F7 หรือ เลือกที่ BUILD > Build Solution

6) หากโปรแกรมที่เขียนถูกต้อง และไม่มีข้อผิดพลาดจะแสดงข้อความ ดังรูปที่ 3.10

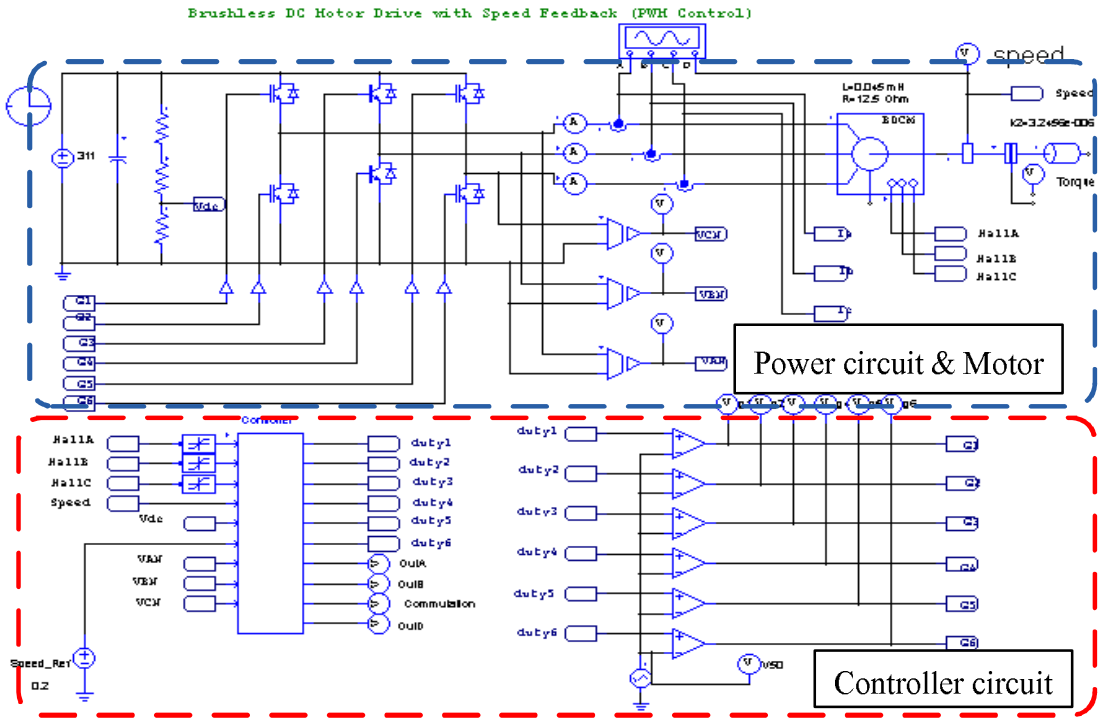


รูปที่ 3.10 แสดงผลการคอมไพล์ Source Code เพื่อสร้าง DLL ในโปรแกรม Visual Studio 2005 ที่สมบูรณ์

7) หลังจากคอมไพล์ Source Code ที่พัฒนาสมบูรณ์เรียบร้อยแล้ว ก็จะได้ไฟล์ DLL สำหรับเรียกใช้งานในโปรแกรม PSIM

2. โปรแกรม PSIM Version 9.0.3

โปรแกรม PSIM คือโปรแกรมที่ใช้ในการออกแบบวงจรอิเล็กทรอนิกส์กำลัง (Power Electronics) และวงจรขับมอเตอร์ (Motor Drive) ถูกพัฒนาโดยบริษัท Powersim Inc. ข้อดี มีความเร็วในการจำลอง ง่ายต่อการใช้งาน และสามารถเชื่อมต่อกับโปรแกรมและอุปกรณ์ภายนอกได้ โดยในงานวิจัยนี้ได้ใช้โปรแกรม PSIM จำลองการทำงานวงจรขับเคลื่อนมอเตอร์ไฟฟ้ากระแสตรงแบบไร้แปรงถ่านซึ่งแบ่งได้เป็น 2 ส่วน คือ ส่วนวงจรควบคุม (Controller circuit) และส่วนวงจรกำลังและมอเตอร์ (Power circuit and Motor) ดังรูปที่ 3.11

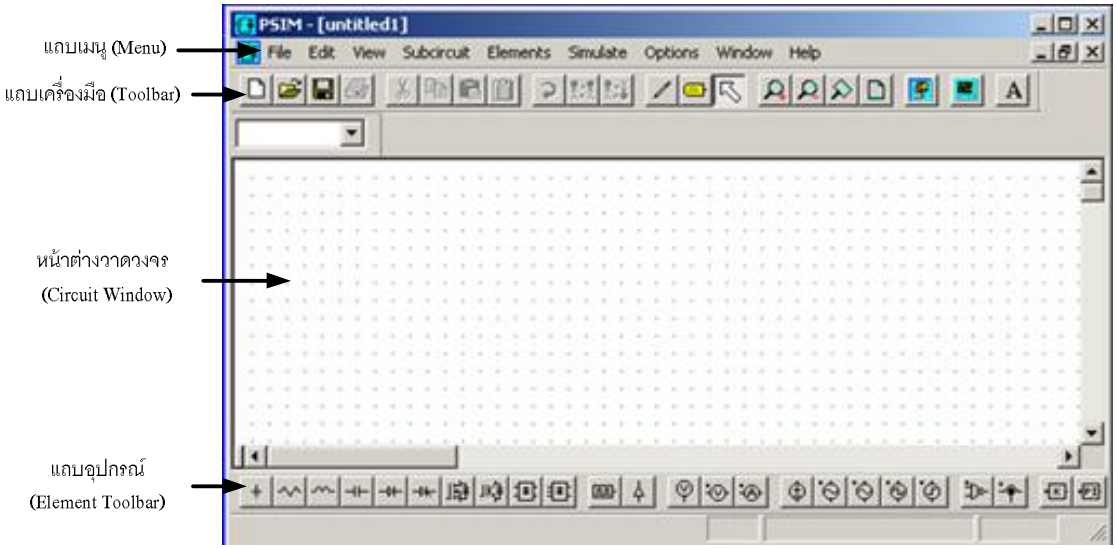


รูปที่ 3.11 วงจรจำลองการขับเคลื่อนมอเตอร์ไฟฟ้ากระแสตรงแบบไร้แปรงถ่าน

วิธีการสร้างวงจร และจำลองผลในโปรแกรม PSIM (Version 9.0.3)

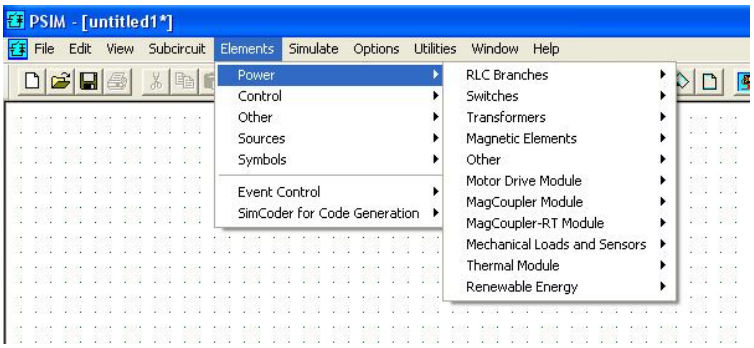
เพื่อให้ง่ายต่อการเข้าใจ จึงขอยกตัวอย่างในการสร้างวงจรจำลองการขับเคลื่อนมอเตอร์ไฟฟ้ากระแสตรงไร้แปรงถ่าน ซึ่งมีขั้นตอนในการสร้างวงจรดังนี้

1) เปิดโปรแกรม PSIM ทำการสร้างหน้าวาดวงจร โดยกดที่ File แล้วก็ New จะขึ้นหน้าต่างสำหรับวาดวงจร ดังรูปที่ 3.12



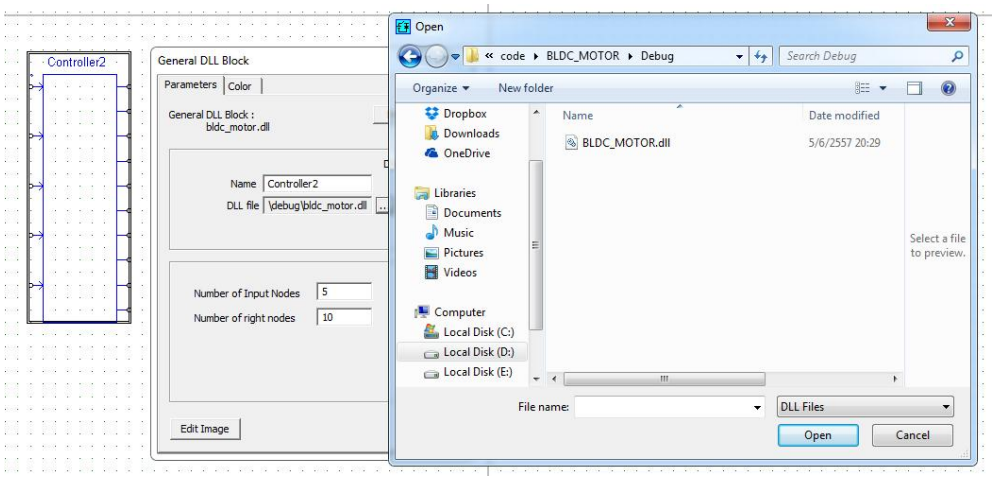
รูปที่ 3.12 หน้าต่างวาดวงจรในโปรแกรม PSIM

2) เลือกอุปกรณ์ที่ใช้สำหรับสร้างวงจร โดยไปที่แถบเมนู แล้วเลือกไปที่ Elements ซึ่งประกอบไปด้วยอุปกรณ์ไฟฟ้า ส่วนของวงจรควบคุม และแหล่งจ่าย ดังรูปที่ 3.13



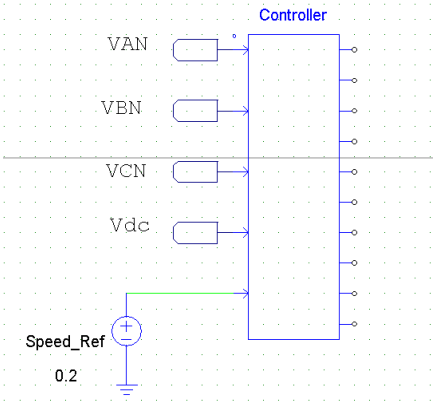
รูปที่ 3.13 การเลือกอุปกรณ์ที่ใช้สร้างวงจรในโปรแกรม PSIM

3) ทำการสร้างวงจรควบคุม โดยบล็อกที่ใช้จะอยู่ในส่วน Elements > Other > Functions Block > Generate DLL Block กำหนดจำนวนอินพุต และเอาต์พุต โดยการ Double Clicks ที่บล็อกทำการกำหนดจำนวนอินพุต (Number of input nodes) และจำนวนเอาท์พุต (Number of right nodes) หลังจากนั้นเลือกไฟล์ DLL ที่พัฒนาโดย Visual studio 2005 ดังรูปที่ 3.14

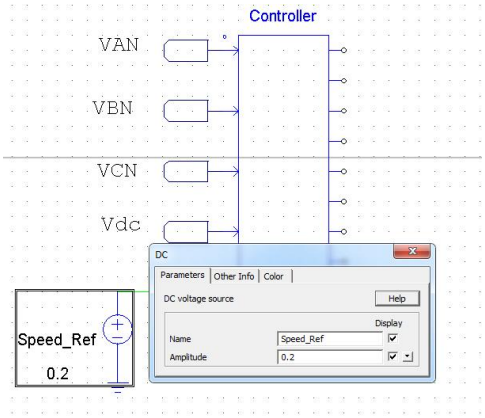


รูปที่ 3.14 การสร้างบล็อกควบคุม และการเลือก DLL file

4) สร้างส่วนสัญญาณอินพุตที่ติดต่อกับ Generate DLL Blocks ซึ่งรับสัญญาณอินพุตจาก ตัวตรวจจับแรงดันไฟฟ้าของมอเตอร์ทั้ง 3 เฟส Van , Vbn และ Vcn , แรงดัน DC Bus จากวงจรกำลัง และ ความเร็วรอบมอเตอร์อ้างอิง Speed Ref โดยกำหนดจากค่า DC source (Elements > Sources >Voltage> DC เพื่อเป็นค่าความเร็วรอบมอเตอร์อ้างอิงให้แก่ระบบหน่วยเป็น PU ดังรูปที่ 3.15 - 3.16

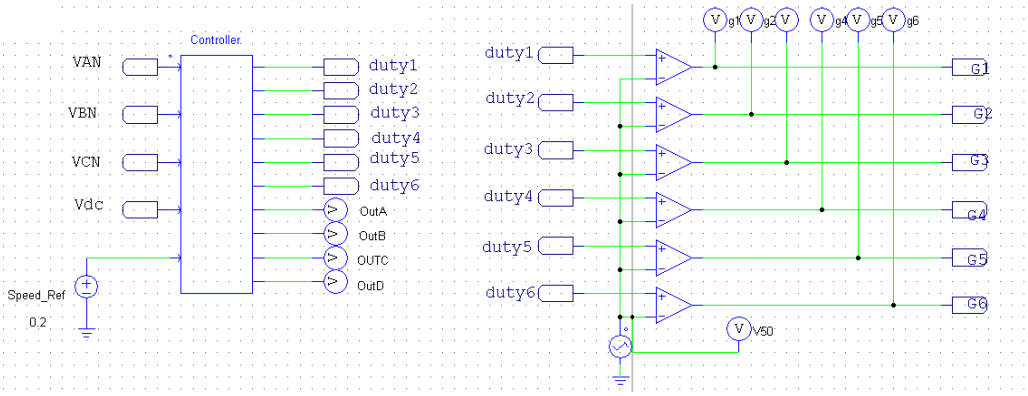


รูปที่ 3.15 ส่วนอินพุตที่ต่อกับ Generate DLL Block



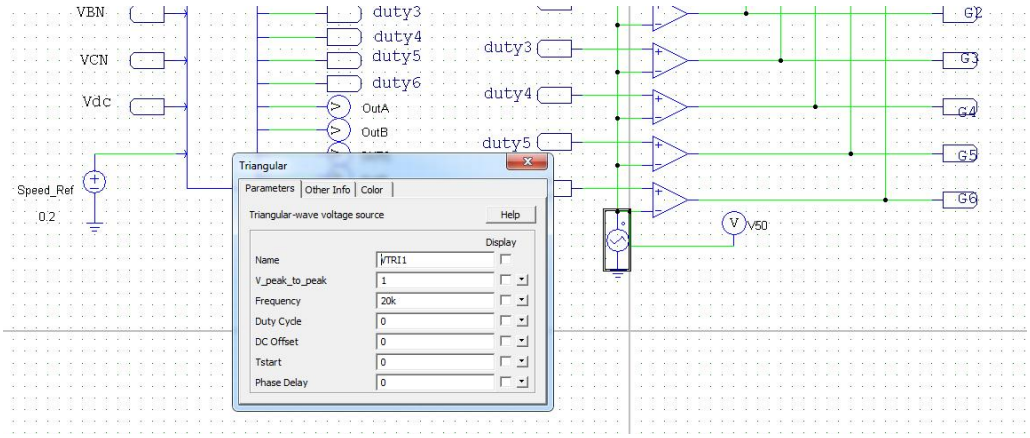
รูปที่ 3.16 การกำหนดค่าความเร็วรอบมอเตอร์อ้างอิง

5) สร้างอุปกรณ์เชื่อมต่อกับ Generate DLL Block ทางด้านเอาต์พุต โดยส่วนนี้เป็นส่วนที่สร้างสัญญาณพัลซ์วิดมอดูเลชั่น เพื่อไปควบคุมการทำงานสวิทช์อิเล็กทรอนิกส์กำลังในวงจรกำลัง โดยใช้ ออปแอมป์เปรียบเทียบแรงดันทั้งหมด 6 ตัว (กด Elements > Power>Other >OP.Amp.) เพื่อใช้ในการเปรียบเทียบสัญญาณเอาต์พุตจาก Generate DLL Block กับความถี่สวิตชิ่งที่สร้างจากแหล่งจ่ายแรงดันไฟฟ้าสามเหลี่ยม (กด Elements > Source > Voltage > Triangle) จากนั้นลากเส้นต่อวงจรตามลำดับ ดังรูปที่ 3.17



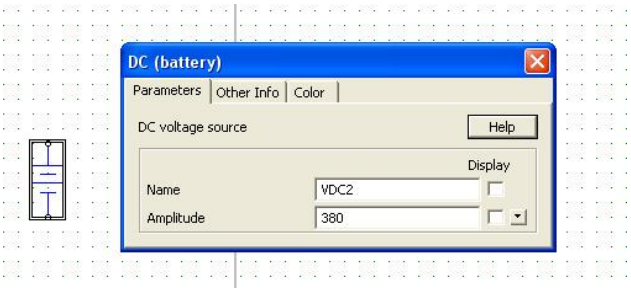
รูปที่ 3.17 ส่วนเอาต์พุตที่ต่อกับ Generate DLL Block

6) ใส่ค่าความถี่ของแหล่งจ่ายแรงดันไฟฟ้าสามเหลี่ยม เท่ากับ 20K ซึ่งเป็นค่าความถี่สวิตชิ่ง แสดงดังรูปที่ 3.18



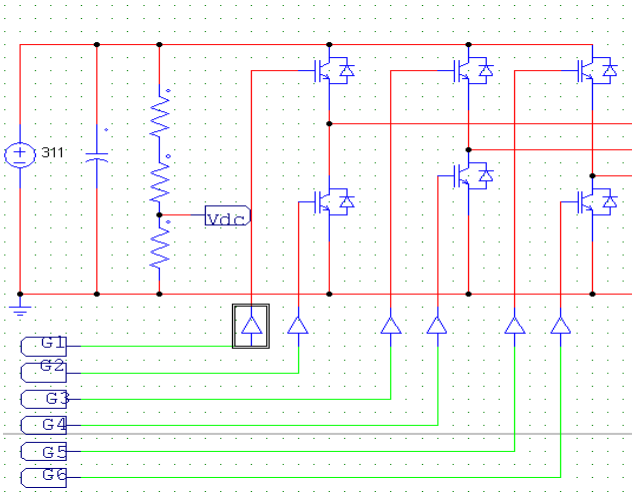
รูปที่ 3.18 การกำหนดค่าความถี่ของแหล่งจ่ายแรงดันไฟฟ้าสามเหลี่ยม

7) ทำการสร้างวงจรกำลัง เพื่อขับเคลื่อนมอเตอร์ โดยเพิ่มแหล่งจ่ายแรงดันไฟฟ้ากระแสตรง (DC) ที่มีแอมพลิจูดเท่ากับ 311 V (Elements > Sources > Voltage > DC battery) แสดงดังรูปที่ 3.19



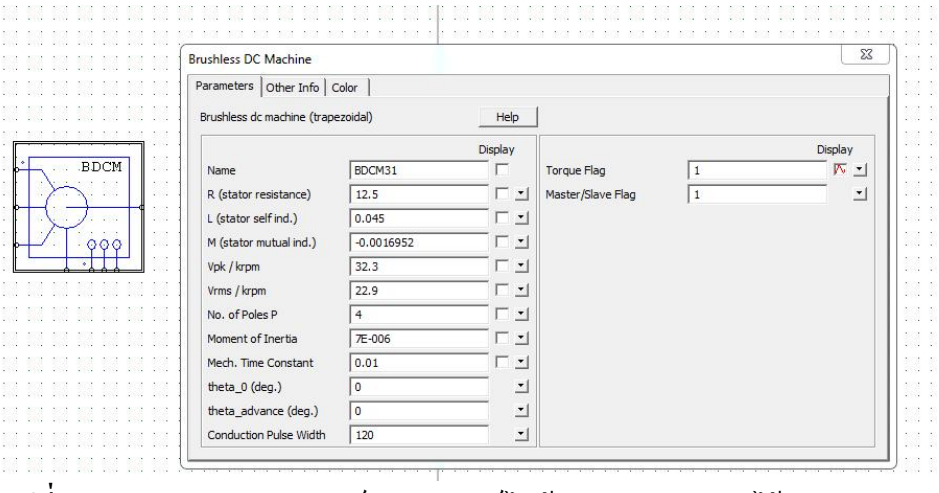
รูปที่ 3.19 กำหนดค่าแอมพลิจูดของแรงดันไฟฟ้ากระแสตรง

8) ทำการเพิ่มอุปกรณ์ตัวเก็บประจุไฟฟ้าที่ DC bus (Elements > Power > Capacitor) IGBT สวิทช์ (Elements > Switches > IGBT) และตัวควบคุมเปิด-ปิดสวิทช์ On-Off controller (Elements > Other > Switch controllers > On-Off controller) เพื่อรับสัญญาณควบคุมการทำงาน IGBT จากวงจรควบคุม แสดงดังรูปที่ 3.20



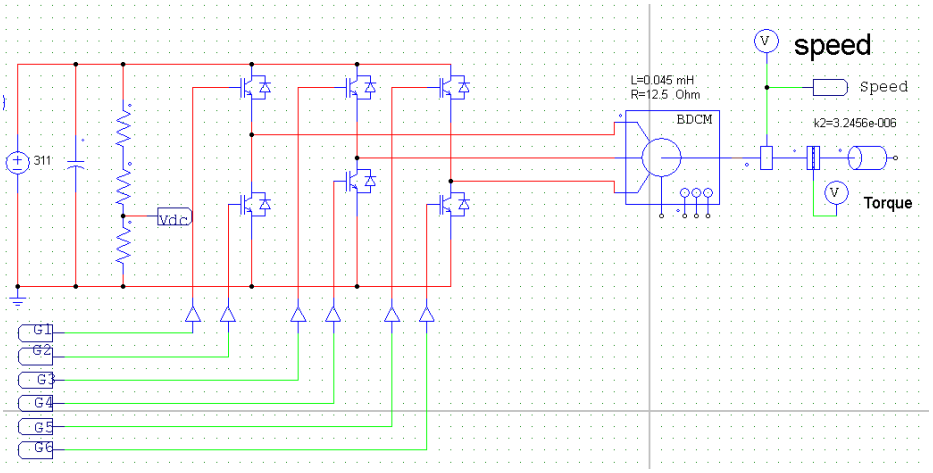
รูปที่ 3.20 วิธีการต่อวงจรอินเวอร์เตอร์ 3 เฟส

9) ทำการเพิ่มมอเตอร์ไฟฟ้ากระแสตรงแบบไร้แปรงถ่าน(Elements > Power > Motor Drive Module > Brushless DC Machine) หลังจากนั้นกำหนดค่าพารามิเตอร์มอเตอร์ที่ใช้ในการจำลอง เช่น ค่าความต้านทานขดลวดสเตเตอร์, ค่าความเหนี่ยวนำในขดลวดสเตเตอร์, ค่าสัมประสิทธิ์แรงเคลื่อนไฟฟ้าต้านกลับ และ จำนวนโพล เป็นต้น แสดงดังรูปที่ 3.21



รูปที่ 3.21 กำหนดค่าพารามิเตอร์ของมอเตอร์ไฟฟ้ากระแสตรงแบบไร้แปรงถ่าน

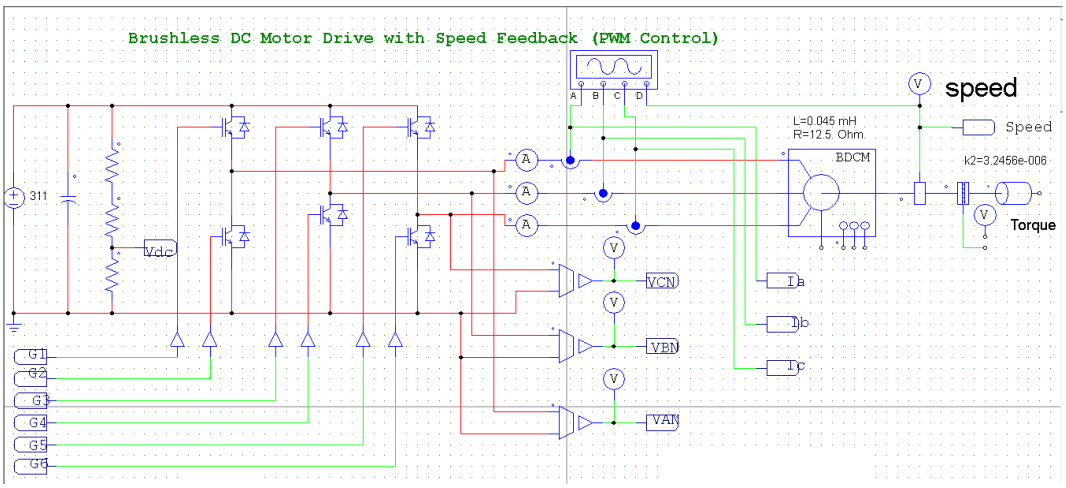
10) ใส่อุปกรณ์โหลดทางกล (Elements > Power > Mechanical Loads and Sensors > Mechanical Loads (general)) ตัวตรวจจับความเร็วรอบมอเตอร์ (Elements > Power > Mechanical Loads and Sensors > Speed Sensor) และตัวตรวจจับแรงบิดมอเตอร์ (Elements > Power > Mechanical Loads and Sensors > Torque Sensor) เชื่อมต่อเข้ากับวงจรอินเวอร์เตอร์ 3 เฟส กับมอเตอร์ ดังแสดงรูปที่ 3.22



รูปที่ 3.22 การต่ออุปกรณ์โหลดทางกลและมอเตอร์เข้ากับระบบขับเคลื่อนมอเตอร์ไฟฟ้ากระแสตรงแบบไร้แปรงถ่าน

11) ใส่อุปกรณ์ ตัวตรวจจับแรงดันไฟฟ้า (Elements > Other > Sensor > Voltage Sensor) เพื่อวัดแรงดันไฟฟ้ามอเตอร์ทั้ง 3 เฟสและเป็นสัญญาณอินพุตให้กับวงจรควบคุม โดยกำหนดค่า Gain เท่ากับ 1/311 เพื่อแปลงขนาดแรงดันไฟฟ้าให้เป็นหน่วย PU (Per Unit) ก่อนส่งให้วงจรควบคุม

12) ใส่อุปกรณ์ตัวตรวจจับกระแสไฟฟ้า (Elements > Other > Sensor > Current Sensor) เพื่อวิเคราะห์กระแสไฟฟ้าของมอเตอร์ แสดงดังรูปที่ 3.23



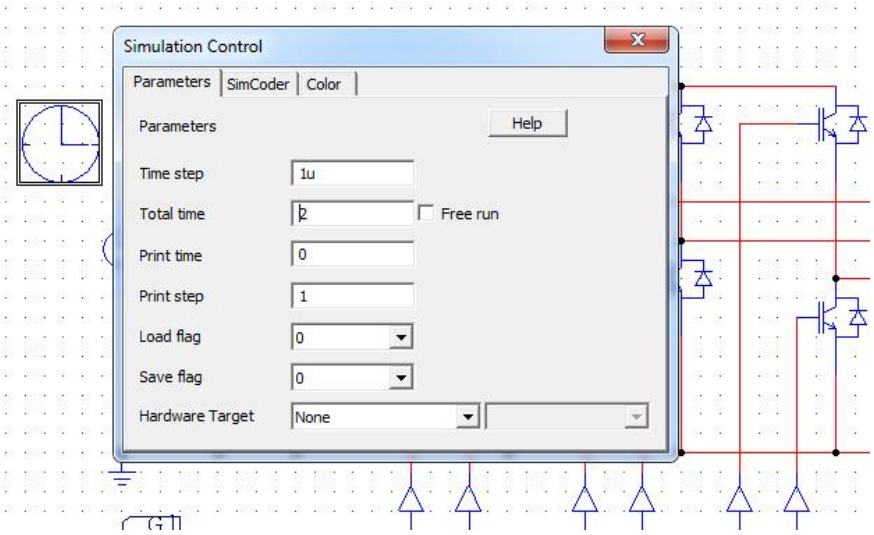
รูปที่ 3.23 การต่ออุปกรณ์ตัวตรวจจับแรงดันไฟฟ้า และตัวตรวจจับกระแสไฟฟ้า เข้ากับระบบขับเคลื่อนมอเตอร์ไฟฟ้ากระแสตรงแบบไร้แปรงถ่าน

13) การตั้งค่าการจำลองผลของวงจร Simulation control (Simulate > Simulation Control) หลังจากนั้นกำหนดค่า Time step โดยวิธีการกำหนดค่า Time Step พิจารณาจาก ความถี่สูงสุดของวงจร สำหรับแบบจำลองนี้คือ ความถี่สวิตชิ่ง 20 kHz และเพิ่มความละเอียดขั้นต่ำ 10 เท่าของความถี่สูงสุด

$$\text{Time step} \le \frac{1}{10} \times \frac{1}{20 \times 10^3} \tag{3.1}$$

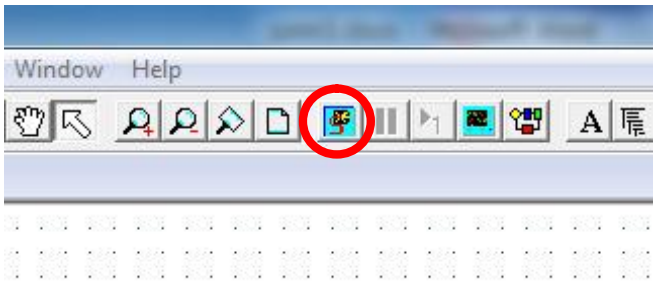
$$\text{Time step} \le 5\ \mu s$$

สำหรับในงานวิจัยนี้ได้กำหนดค่า 1 µs ให้แก่ Simulation Control แสดงดังรูปที่ 3.24



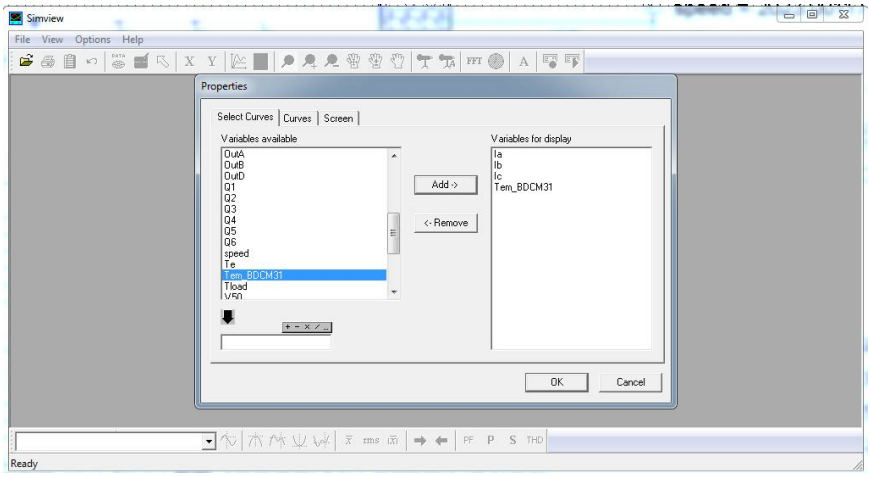
รูปที่ 3.24 การตั้งค่าการจำลองผลของวงจร

14) ทำการจำลองการทำงาน โดยกดที่ปุ่ม Run Simulation บนแถบเครื่องมือ ดังแสดงในรูปที่ 3.25

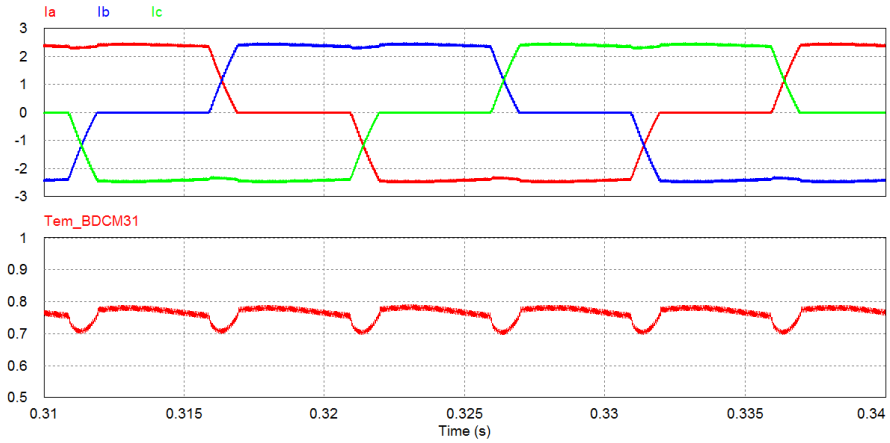


รูปที่ 3.25 สัญลักษณ์ปุ่ม Run Simulation

15) หลังจากการจำลองผลจากโปรแกรมเสร็จเรียบร้อย จะขึ้นหน้าต่างให้เลือกแสดงกราฟสัญญาณต่างๆตามจุดที่ติดตั้งในวงจร โดยทำการเลือกตัวแปรที่ต้องการให้แสดงผลกราฟ มาในช่องทางฝั่งขวา ด้วยการกดปุ่ม Add -> และลบตัวแปรที่ไม่ต้องการโดยการกด <- Remove ในที่นี้แสดงตัวอย่างการดูสัญญาณกระแสไฟฟ้ามอเตอร์(Ia, Ib, Ic) และ แรงบิดมอเตอร์ ดังรูปที่ 3.26 – 3.27



รูปที่ 3.26 หน้าต่าง Simview สำหรับดูกราฟสัญญาณ



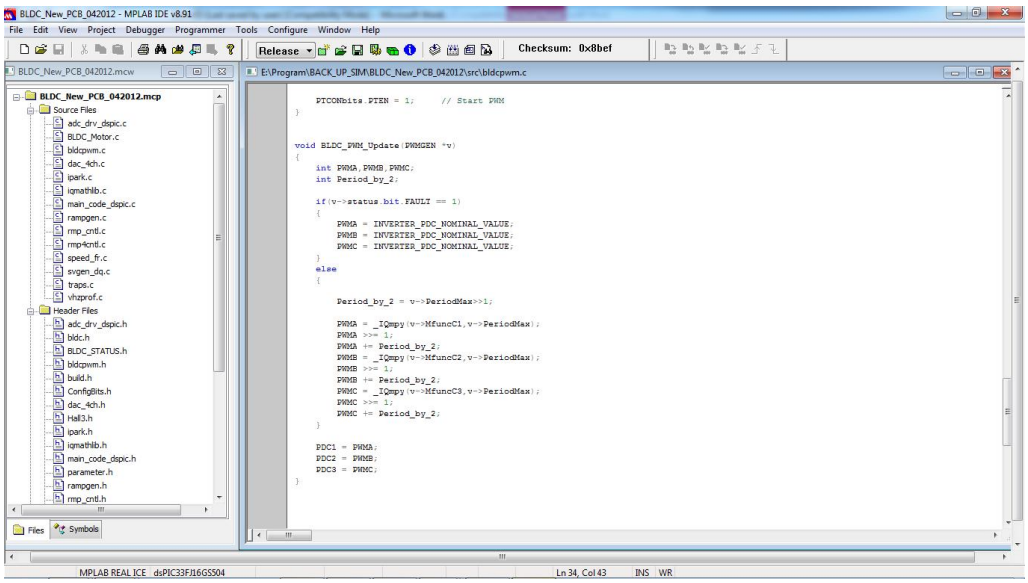
รูปที่ 3.27 กราฟสัญญาณกระแสไฟฟ้ามอเตอร์(Ia, Ib, Ic) และ แรงบิดมอเตอร์

3.1.2 ซอฟต์แวร์สำหรับพัฒนาบนไมโครคอนโทรเลอร์

1. โปรแกรม MPLAB Version 8.91

MPLAB เป็นโปรแกรมประเภท (Integrated development environment: IDE) ซึ่งพัฒนามาจากบริษัท Microchip ผู้ผลิตไมโครคอนโทรลเลอร์เบอร์ dsPIC33FJ16GS504 นั้นเอง โดยภาพรวมของโปรแกรม MPLAB Version 8.91 แสดงดังรูปที่ 3.28 ภายในโปรแกรมมีองค์ประกอบหรือตัวช่วยต่างๆ ที่จะคอยช่วยเหลือ นักพัฒนาโปรแกรม เพื่อเสริมให้เกิดความรวดเร็ว ถูกต้อง แม่นยำ ทำให้การพัฒนางานต่างๆมีความรวดเร็วมากขึ้น ซึ่งส่วนประกอบของโปรแกรมมีดังต่อไปนี้

1) Editor ทำหน้าที่ช่วยในการเขียนคำสั่งซอร์สโค้ด ที่เป็นภาษาซี
2) Compiler / Assembler ทำหน้าที่แปลซอร์สโค้ดให้อยู่ในรูป Hex ไฟล์ก่อนส่งไปโปรแกรมลงไมโครคอนโทรลเลอร์
3) Simulator ทำหน้าที่จำลองการทำงานของโปรแกรมก่อนที่จะโปรแกรมลงไมโครคอนโทรลเลอร์
4) Emulator (ต้องมีฮาร์ดแวร์เพิ่มต่อพ่วงภายนอก) ไว้จำลองการทำงานโดยการต่อเข้ากับฮาร์ดแวร์จริง โดยในงานวิจัยนี้ได้ใช้ อุปกรณ์ REAL ICE เชื่อมต่อโปรแกรมกับฮาร์ดแวร์
5) Programmer (ต้องต่อพ่วงเข้ากับฮาร์ดแวร์ภายนอก) เพื่อนำเอาไฟล์ Hex โปรแกรมลงตัวไมโครคอนโทรลเลอร์ ซึ่งรายละเอียดการใช้งานโปรแกรมสามารถดูได้ที่ www.microchip.com



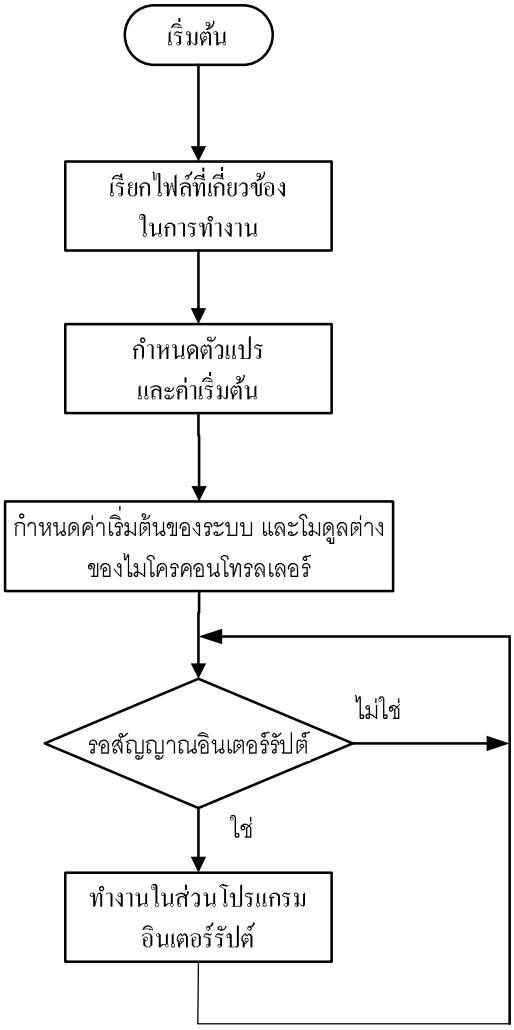
รูปที่ 3.28 ภาพรวมของโปรแกรม MPLAB Version 8.91

3.1.3 ขั้นตอนการทำงานของโปรแกรมการขับเคลื่อนมอเตอร์ไฟฟ้ากระแสตรงไร้แปรงถ่านแบบไร้ตัวตรวจจับตำแหน่งพร้อมทั้งลดการกระเพื่อมแรงบิดมอเตอร์

โปรแกรมการทำงานการขับเคลื่อนมอเตอร์ไฟฟ้ากระแสตรงไร้แปรงถ่านแบบไร้ตัวตรวจจับตำแหน่งพร้อมทั้งลดการกระเพื่อมแรงบิดมอเตอร์ที่นำเสนอถูกแบ่งออกเป็น 2 ส่วนดังนี้

1 โปรแกรมหลัก (Main Program)

ในส่วนของโปรแกรมหลักมีหน้าที่กำหนดค่าเริ่มต้นของระบบ เช่น ความถี่ในการทำงานของตัวไมโครคอนโทรลเลอร์ การทำงานของพอร์ต การทำงานฟังก์ชั่นพัลซ์วิดมอดูเลชั่นและกำหนดฐานเวลาการเกิดอินเตอร์รัปต์ เป็นต้น หลังจากนั้นทำการตรวจสอบว่า มีสัญญาณอินเตอร์รัปต์จากโมดูลไทเมอร์หรือไม่ ถ้ามีก็จะกระโดดไปทำงานยังโปรแกรมในส่วนอินเตอร์รัปต์ทันทีดังรูปที่ 3.29

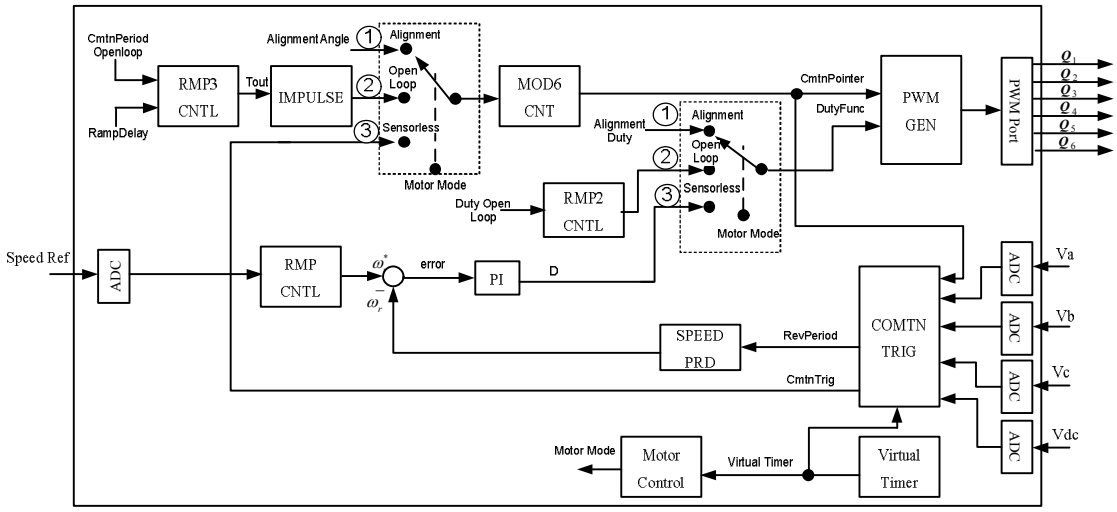


รูปที่ 3.29 การทำงานของโปรแกรมหลัก

2 โปรแกรมในส่วนอินเตอร์รัปต์ (Interrupt Service Routine : ISR)

ส่วนของโปรแกรมอินเตอร์รัปต์มีหน้าที่ควบคุมการขับเคลื่อนมอเตอร์ โดยจะเริ่มทำงานเมื่อได้รับสัญญาณอินเตอร์รัปต์จากโมดูลไทเมอร์ทุกๆ 62.5μS หรือเท่ากับ 16 KHz ซึ่งเป็นความถี่สวิตชิ่ง เมื่อทำงานในส่วนของโปรแกรมอินเตอร์รัปต์ เสร็จแล้วจะกลับไปทำงานในส่วนของโปรแกรมหลักต่อแล้วรอเวลาในการทำงานของโปรแกรมอินเตอร์รัปต์ต่อไป

ในการขับเคลื่อนมอเตอร์โดยไม่ใช้ตัวตรวจจับตำแหน่งแบ่งช่วงการทำงานเป็น 3 ช่วง 1) บังคับตำแหน่งของโรเตอร์ไปยังตำแหน่งเริ่มต้น (Rotor Alignment) 2) การขับเคลื่อนมอเตอร์แบบลูปเปิด (Open Loop) 3) การขับเคลื่อนมอเตอร์โดยไม่ใช้ตัวตรวจจับตำแหน่ง (Sensorless) ดังแสดงรูปที่ 3.30



รูปที่ 3.30 บล็อกไดอะแกรมการทำงานส่วนของโปรแกรมอินเตอร์รัปต์

จากรูปที่ 3.30 เริ่มต้นการทำงานส่วนของโปรแกรมอินเตอร์รัปต์ ฟังก์ชั่น Motor Control จะกำหนดให้มอเตอร์ทำงาน(Motor Mode) ในช่วง Rotor Alignment ซึ่งในช่วงนี้มีขั้นตอนการทำงานดังนี้

- กำหนดค่าดิวตี้ไซเคิลของแรงดันไฟฟ้าที่จ่ายให้มอเตอร์(Alignment Duty) และมุมโรเตอร์ที่ต้องการให้เคลื่อนที่ไปอยู่ในตำแหน่งนั้น (Alignment Angle)
- หลังจากนั้นนำค่าป้อนเข้าฟังก์ชั่น PWM Gen เพื่อสร้างสัญญาณควบคุมสวิทช์อิเล็กทรอนิกส์
- ถัดไปมีการหน่วงเวลาการจ่ายแรงดันไฟฟ้าเข้ามอเตอร์โดยมีฟังก์ชั่น Virtual Timer เป็นตัวนับเวลา ซึ่งระยะเวลาที่ใช้ในขั้นตอนนี้ขึ้นอยู่กับความเฉื่อยของมอเตอร์ เมื่อครบเวลาที่กำหนดฟังก์ชั่น Motor Control กำหนดให้มอเตอร์เปลี่ยนการทำงาน (Motor Mode) อยู่ในช่วงOpen Loop Speed Control

การทำงานช่วงOpen Loop มีขั้นตอนดังนี้

- เริ่มต้นอ่านค่าแรงดันไฟฟ้า Va , Vb , Vc และ Vdc และความเร็วรอบมอเตอร์อ้างอิง Speed Ref โดยผ่านโมดูลแปลงสัญญาณแอนะล็อกเป็นดิจิตอล
- หลังจากนั้นส่งค่าแรงดันไฟฟ้าให้ ฟังก์ชั่น COMTN_TRIG เพื่อหาจุดผ่านศูนย์ของแรงเคลื่อนไฟฟ้าต้านกลับ หลังจากนั้นทำการหน่วงเวลา 30 องศาไฟฟ้า เพื่อหาจุดคอมมิวเตชั่น และ คาบเวลาจุดผ่านศูนย์ของแรงเคลื่อนไฟฟ้าต้านกลับแล้วส่งให้ฟังก์ชั่น SPEED_PRD คำนวณหาความเร็วรอบมอเตอร์ ในช่วงนี้จุดคอมมิเตชั่นที่ได้ยังไม่ถูกนำไปใช้ในการขับเคลื่อนมอเตอร์ เนื่องจากในช่วงที่มอเตอร์ไฟฟ้าหยุดนิ่งหรือหมุนที่ความเร็วรอบต่ำ แรงเคลื่อนไฟฟ้าต้านกลับมีค่าน้อยเกินไป อาจทำให้ไม่ถูกต้อง ดังนั้นจึงทำการสร้างตำแหน่งคอมมิวเตชั่นขึ้นมา โดยไม่สนใจตำแหน่งจริงของโรเตอร์
- เรียกฟังก์ชั่น RMP3CNTRL เพื่อกำหนดเวลาการคอมมิวเตชั่น (Tout) ซึ่งค่านี้ขึ้นอยู่กับค่าความเร็วรอบอ้างอิงมอเตอร์ที่กำหนดในช่วง Open Loop
- หลังจากนั้นส่งค่าดังกล่าวไปที่ ฟังก์ชั่น Impulse เพื่อสร้างสัญญาณอิมพัลซ์ซึ่งมีคาบเวลาเท่ากับ Tout
- ฟังก์ชั่น MOD6CNT รับสัญญาณอิมพัลซ์เพื่อกำหนดเป็น State S1 ถึง S6 แล้วส่งให้ฟังก์ชั่น PWM_GEN
- หลังจากนั้น กำหนดค่าดิวตี้ไซเคิลสูงสุด (Duty Open loop) แล้วส่งให้ ฟังก์ชั่น RMP2CMTL ทำการเพิ่มค่าดิวตี้ไซเคิลเพื่อเร่งให้มอเตอร์หมุนด้วยความเร็วสูงขึ้น หลังจากนั้นส่งค่าดิวตี้ไซเคิลให้ฟังก์ชั่น PWM_GEN
- เรียกฟังก์ชั่น PWM_GEN เพื่อสร้างสัญญาณควบคุมสวิทช์อิเล็กทรอนิกส์ ตามค่าดิวตี้ไซเคิล (Duty Func) และ ตำแหน่งโรเตอร์ (CmtnPointer)
- เรียกฟังก์ชั่น Motor Control ทำการตรวจสอบว่าความเร็วรอบมอเตอร์ที่วัดได้มีค่ามากกว่าหรือเท่ากับความเร็วรอบอ้างอิงสูงสุดช่วง Open Loop Speed Control หรือไม่ ถ้าไม่ทำการปรับค่า Duty Open loop เพิ่มขึ้น ถ้าใช่ฟังก์ชั่น Motor Control กำหนดให้มอเตอร์เปลี่ยนการทำงาน (Motor Mode) อยู่ในช่วง Sensorless

การทำงานช่วง Sensorless มีขั้นตอนดังนี้

- เริ่มต้นอ่านค่าแรงดันไฟฟ้า Va , Vb , Vc และ Vdc และความเร็วรอบมอเตอร์อ้างอิง Speed Ref โดยผ่านโมดูลแปลงสัญญาณแอนะล็อกเป็นดิจิตอล
- หลังจากนั้นส่งค่าแรงดันไฟฟ้าให้ ฟังก์ชั่น COMTN_TRIG เพื่อหาจุดผ่านศูนย์ของแรงเคลื่อนไฟฟ้าต้านกลับ จุดคอมมิวเตชั่น และ คาบเวลา แล้วส่งให้ฟังก์ชั่น SPEED_PRD คำนวณหาความเร็วรอบมอเตอร์เหมือนกับช่วง Open Loop

- นำค่า Speed Ref ป้อนเข้า ฟังก์ชั่น RMPCNTL เพื่อเพิ่มค่าความเร็วรอบมอเตอร์อ้างอิง (ω_r^*) ตามความชันที่กำหนด
- หลังจากนั้นส่งค่าความเร็วรอบอ้างอิง (ω_r^*) และความเร็วป้อนกลับ (ω_r) ส่งให้ฟังก์ชั่น PI เพื่อทำการคำนวณหาค่าดิวตี้ไซเคิลที่เหมาะสมออกมา
- นำค่าค่าดิวตี้ไซเคิล และ จุดคอมมิวเตชั่น ส่งให้ฟังก์ชั่น PWM_GEN เพื่อสร้างสัญญาณควบคุมสวิทช์อิเล็กทรอนิกส์ ในการขับเคลื่อนมอเตอร์

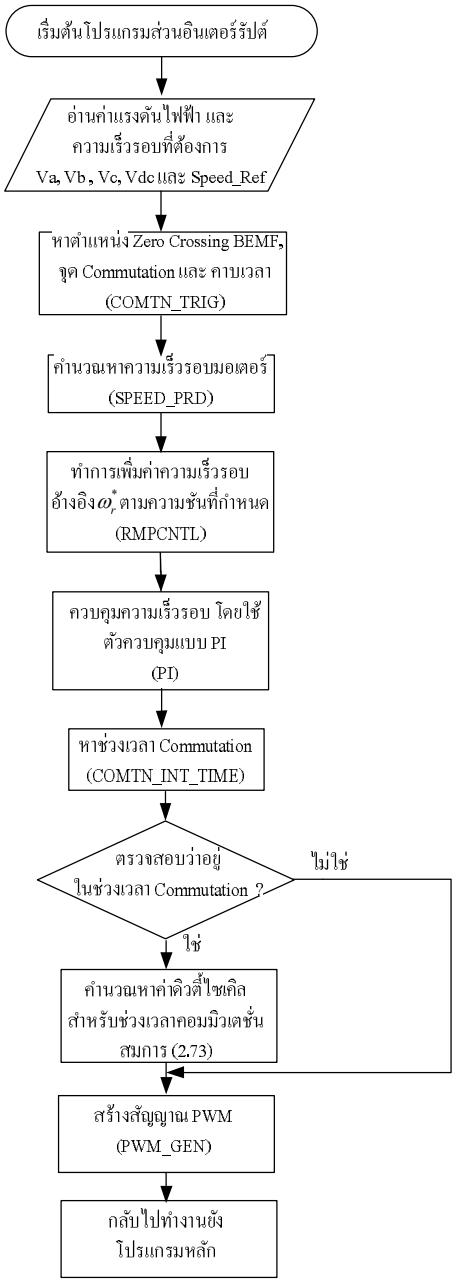
เนื่องจากในงานวิจัยนี้ได้เสนอ วิธีการลดการกระเพื่อมแรงบิดในการขับเคลื่อนมอเตอร์ไฟฟ้ากระแสตรงไร้แปรงถ่าน 2 วิธี คือ การขับเคลื่อนมอเตอร์ด้วยกระแสไฟฟ้าแบบควาซิ-สแคร์(Quasi-Square) ที่มีการชดเชยช่วงเวลาคอมมิวเตชั่น และ การขับเคลื่อนมอเตอร์ด้วยกระแสไฟฟ้าแบบควาซิ-ไซน์ (Quasi-Sine) ดังนั้นโปรแกรมในช่วง Sensorless จึงมี 2 แบบดังนี้

โปรแกรมขับเคลื่อนมอเตอร์ด้วยกระแสไฟฟ้าแบบควาซิ-สแคร์(Quasi-Square)ที่มีการชดเชยช่วงเวลาคอมมิวเตชั่น

ขั้นตอนการทำงานของโปรแกรมนี้เพิ่ม ฟังก์ชั่น COMTN_INT_TIME ซึ่งฟังก์ชั่นนี้รับแรงดันไฟฟ้ามาเพื่อหาช่วงเวลาคอมมิวเตชั่น หลังจากนั้นนำข้อมูลที่ได้ไปกำหนดค่าดิวตี้ไซเคิล โดยค่าดิวตี้ไซเคิลในระบบควบคุมมีอยู่ 2 ค่า คือ ดิวตี้ไซเคิล D ที่ได้จากฟังก์ชั่น PI และดิวตี้ไซเคิล D_{cmp} สำหรับช่วงเวลาคอมมิวเตชั่น เพื่อควบคุมแรงดันไฟฟ้าในเฟสที่ไม่ได้ทำคอมมิวเตชั่นให้มีขนาดแรงดันไฟฟ้าคงที่ ไม่เกิดการกระเพื่อมของขนาดกระแสไฟฟ้า แผงผังลำดับการทำงานของโปรแกรมนี้แสดงดังรูปที่ 3.31 รายละเอียดการทำงานของแต่ละขั้นตอนเรียงลำดับดังต่อไปนี้

- เริ่มต้นอ่านค่าแรงดันไฟฟ้า Va , Vb , Vc และ Vdc และความเร็วรอบมอเตอร์อ้างอิง Speed Ref โดยผ่านโมดูลแปลงสัญญาณแอนะล็อกเป็นดิจิตอล
- หลังจากนั้นส่งค่าแรงดันไฟฟ้าให้ ฟังก์ชั่น COMTN_TRIG เพื่อหาจุดผ่านศูนย์ของแรงเคลื่อนไฟฟ้าต้านกลับ จุดคอมมิวเตชั่น และ คาบเวลา แล้วส่งให้ฟังก์ชั่น SPEED_PRD คำนวณหาความเร็วรอบมอเตอร์
- นำค่า Speed Ref ป้อนเข้า ฟังก์ชั่น RMPCNTL เพื่อเพิ่มค่าความเร็วรอบมอเตอร์อ้างอิง (ω_r^*) ตามความชันที่กำหนด
- หลังจากนั้นส่งค่าความเร็วรอบอ้างอิง (ω_r^*) และความเร็วป้อนกลับ (ω_r) ส่งให้ ฟังก์ชั่น PI เพื่อทำการคำนวณหาค่าดิวตี้ไซเคิล D ที่เหมาะสมออกมา
- เรียกฟังก์ชั่น COMTN_INT_TIME เพื่อหาช่วงเวลาคอมมิวเตชั่น (CMP)

- ตรวจสอบว่าอยู่ในช่วงเวลาคอมมิวเตชั่นหรือไม่ ถ้าใช่ คำนวณหาค่าดิวตี้ไซเคิล D_{cmp} แล้วส่งให้ฟังก์ชั่น PWM_GEN
- นำค่าค่าดิวตี้ไซเคิล และ จุดคอมมิวเตชั่น ส่งให้ฟังก์ชั่น PWM_GEN เพื่อสร้างสัญญาณควบคุมสวิทช์อิเล็กทรอนิกส์ ตามรูปแบบสัญญาณพัลซ์วิมอดูเลชั่นที่นำเสนอ หลังจากนั้นกลับไปที่โปรแกรมหลักเพื่อรอสัญญาณอินเตอรัปต์ครั้งต่อไป

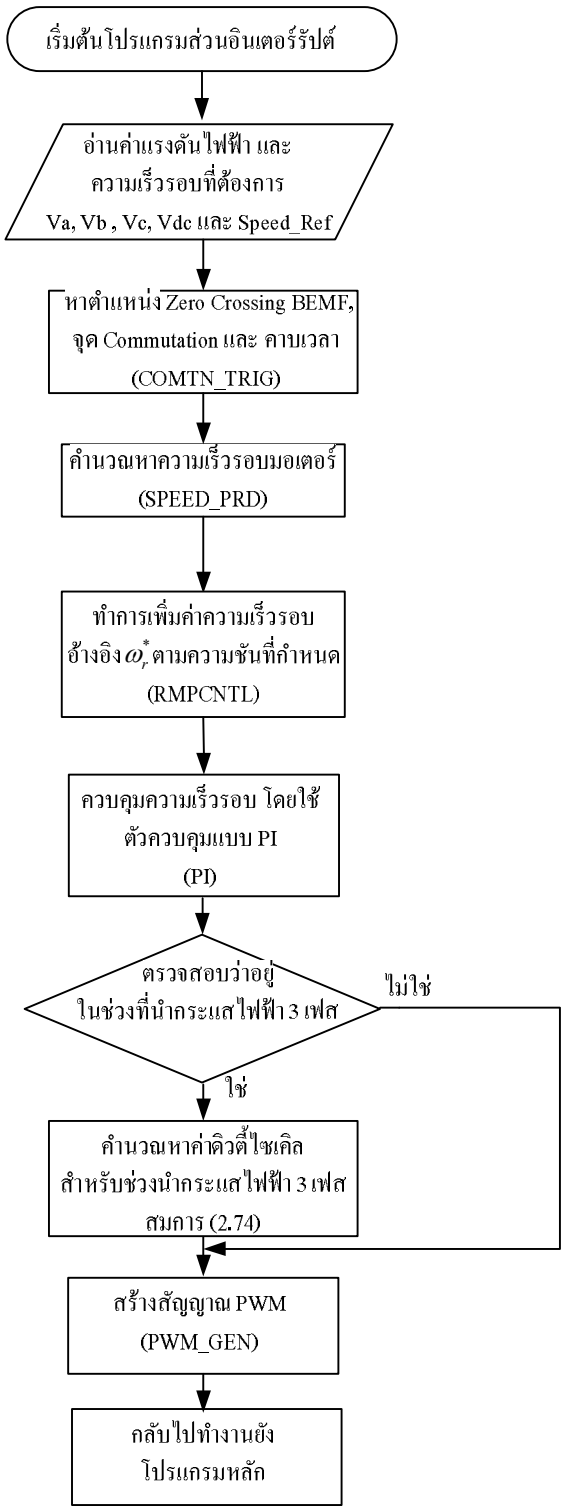


รูปที่ 3.31 แผนผังการทำงานส่วนของโปรแกรมขับเคลื่อนมอเตอร์ด้วยกระแสไฟฟ้าแบบควาซิ-สแคร์ที่มีการชดเชยช่วงเวลาคอมมิวเตชั่น

โปรแกรมสำหรับการขับเคลื่อนมอเตอร์ด้วยกระแสไฟฟ้าแบบควาซิ-ไซน์ (Quasi-Sine)

ขั้นตอนการทำงานของโปรแกรมนี้ ฟังก์ชั่น COMTN_TRIG มีการเปลี่ยนแปลงโดยกำหนดจุดผ่านศูนย์ของแรงเคลื่อนไฟฟ้าต้านกลับ เป็นจุดคอมมิวเตชั่น หลังจากนั้นทำการหน่วงเวลา 30 องศาไฟฟ้า เพื่อหาจุดคอมมิวเตชั่นอีกครั้ง ซึ่งทำให้ ช่วงคอมมิวเตชั่นมีทั้งหมด 12 ช่วง คือ S1 ถึง S12 แต่ละช่วงห่างกัน 30 องศาไฟฟ้า อีกทั้งค่าดิวตี้ไซเคิลในระบบควบคุมมีอยู่ 2 ค่า คือ ดิวตี้ไซเคิล D ที่ได้จากฟังก์ชั่น PI ใช้ในช่วงที่นำกระแสไฟฟ้า 2 เฟส และดิวตี้ไซเคิล D_{reduce} ใช้ในช่วงที่นำกระแสไฟฟ้า 3 เฟส เพื่อให้แรงดันไฟฟ้าแต่ละเฟสมีลักษณะใกล้เคียงกับรูปคลื่นไซน์ ซึ่งทำให้ได้กระแสไฟฟ้ามีลักษณะแบบควาซิ-ไซน์ แผงผังลำดับการทำงานของโปรแกรมนี้แสดงดังรูปที่ 3.32 รายละเอียดการทำงานของแต่ละขั้นตอนเรียงลำดับดังต่อไปนี้

- เริ่มต้นอ่านค่าแรงดันไฟฟ้า Va , Vb , Vc และ Vdc และความเร็วรอบมอเตอร์อ้างอิง Speed Ref โดยผ่านโมดูลแปลงสัญญาณแอนะล็อกเป็นดิจิตอล
- หลังจากนั้นส่งค่าแรงดันไฟฟ้าให้ ฟังก์ชั่น COMTN_TRIG เพื่อหาจุดผ่านศูนย์ของแรงเคลื่อนไฟฟ้าต้านกลับ จุดคอมมิวเตชั่น และ คาบเวลา แล้วส่งให้ฟังก์ชั่น SPEED_PRD คำนวณหาความเร็วรอบมอเตอร์
- นำค่า Speed Ref ป้อนเข้า ฟังก์ชั่น RMPCNTL เพื่อเพิ่มค่าความเร็วรอบมอเตอร์อ้างอิง (ω_r^*) ตามความชันที่กำหนด
- หลังจากนั้นส่งค่าความเร็วรอบอ้างอิง (ω_r^*) และความเร็วป้อนกลับ (ω_r) ให้ ฟังก์ชั่น PI เพื่อทำการคำนวณหาค่าดิวตี้ไซเคิล D ที่เหมาะสมออกมา
- ตรวจสอบว่าอยู่ในช่วงที่นำกระแสไฟฟ้า 3 เฟส หรือไม่ ถ้าใช่ คำนวณหาค่าดิวตี้ไซเคิล D_{reduce} แล้วส่งให้ ฟังก์ชั่น PWM_GEN
- นำค่าค่าดิวตี้ไซเคิล และ จุดคอมมิวเตชั่น ส่งให้ฟังก์ชั่น PWM_GEN เพื่อสร้างสัญญาณควบคุมสวิทช์อิเล็กทรอนิกส์ ตามรูปแบบสัญญาณพัลซ์วิมอดูเลชั่นที่นำเสนอ หลังจากนั้นกลับไปที่โปรแกรมหลักเพื่อรอสัญญาณอินเตอรัปต์ครั้งต่อไป



รูปที่ 3.32 แผนผังการทำงานส่วนของโปรแกรมขับเคลื่อนมอเตอร์ด้วยกระแสไฟฟ้าแบบควาซิ-ไซน์

3.2 โครงสร้างฮาร์ดแวร์ที่นำเสนอ

โครงสร้างทางฮาร์ดแวร์ชุดขับเคลื่อนมอเตอร์ไฟฟ้ากระแสตรงไร้แปรงถ่านแบบไร้ตัวตรวจจับตำแหน่งพร้อมทั้งลดการกระเพื่อมแรงบิดมอเตอร์ที่นำเสนอ แสดงดังรูปที่ 3.33 ภายในบอร์ดประกอบวงจรดังต่อไปนี้

หมายเลข 1 วงจรเรียงกระแส (Rectifier)

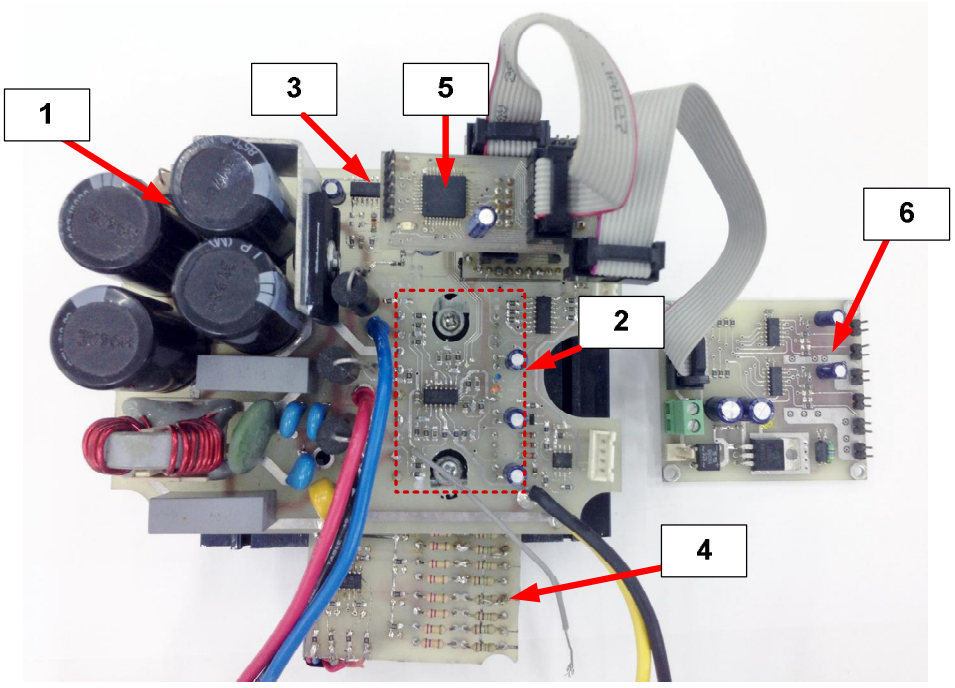
หมายเลข 2 วงจรอินเวอร์เตอร์ (Inverter)

หมายเลข 3 วงจรไฟเลี้ยง (Power Supply)

หมายเลข 4 วงจรปรับระดับสัญญาณของแรงดันไฟฟ้า

หมายเลข 5 ตัวไมโครคอนโทรลเลอร์

หมายเลข 6 วงจรแปลงสัญญาณดิจิตอลเป็นแอนะลอก (Digital-to-Analog converter)



รูปที่ 3.33 ชุดบอร์ดวงจรอินเวอร์เตอร์ขับมอเตอร์ไฟฟ้ากระแสตรงไร้แปรงถ่านแบบไร้ตัวตรวจจับตำแหน่งโรเตอร์

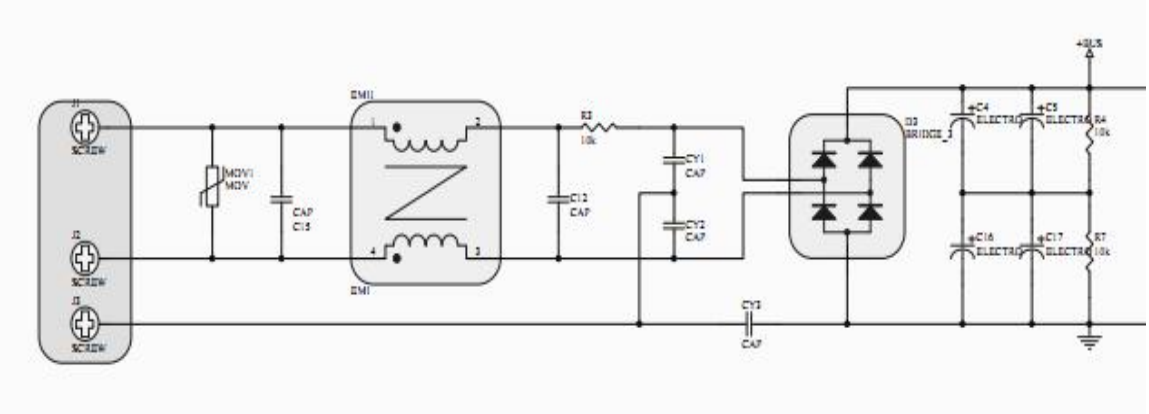
3.2.1 วงจรเรียงกระแส

ในส่วนวงจรเรียงกระแสที่ใช้ในงานวิจัยนี้ประกอบด้วยกัน 2 ส่วนคือ

1) ส่วนวงจรป้องกันสัญญาณรบกวนแม่เหล็กไฟฟ้า(Electromagnetic Interference; EMI) โดยในส่วนนี้ประกอบด้วย ตัวอุปกรณ์ป้องกันแรงดันไฟฟ้าเกินชั่วขณะ (Metal Oxide Varistor: MOV) ต่อคร่อมระหว่างสายเฟสกับสายนิวทรอล ซึ่งตัวMOV จะ Clamp แรงดันไฟฟ้าไม่ให้แรงดันไฟฟ้าตก

คร่อมตัวมันเกินกว่าค่าที่กำหนด ซึ่งจะต่อกับภาควงจรกรองความถี่ต่ำผ่าน LC low pass filter ซึ่งทำหน้าที่กรองความถี่ของแรงดันไฟฟ้าอินพุตและสัญญาณรบกวนอื่นๆ ก่อนจะส่งไปให้ส่วนวงจรเรียงกระแสต่อไป

2) ส่วนวงจรเรียงกระแส ทำหน้าที่แปลงแรงดันไฟฟ้ากระแสสลับระดับแรงดันไฟฟ้าที่ 220 V (rms) ความถี่ 50 Hz ให้เป็นแรงดันไฟฟ้ากระแสตรงเต็มคลื่น (Full Wave) ที่ระดับ 311 V โดยภายในวงจรเรียงกระแสประกอบด้วย บริดจ์ไดโอดซึ่งภายในประกอบด้วยไดโอดทั้งหมด 4 ตัว ต่ออนุกรมกับ ตัวเก็บประจุเพื่อกรองแรงดันไฟฟ้าให้เรียบ และจ่ายแรงดันไฟฟ้ากระแสตรงนี้ไปยังวงจรอินเวอร์เตอร์เพื่อทำการขับเคลื่อนมอเตอร์ต่อไป แสดงดังรูปที่ 3.34



รูปที่ 3.34 วงจรเรียงกระแส (Rectifier Circuit)

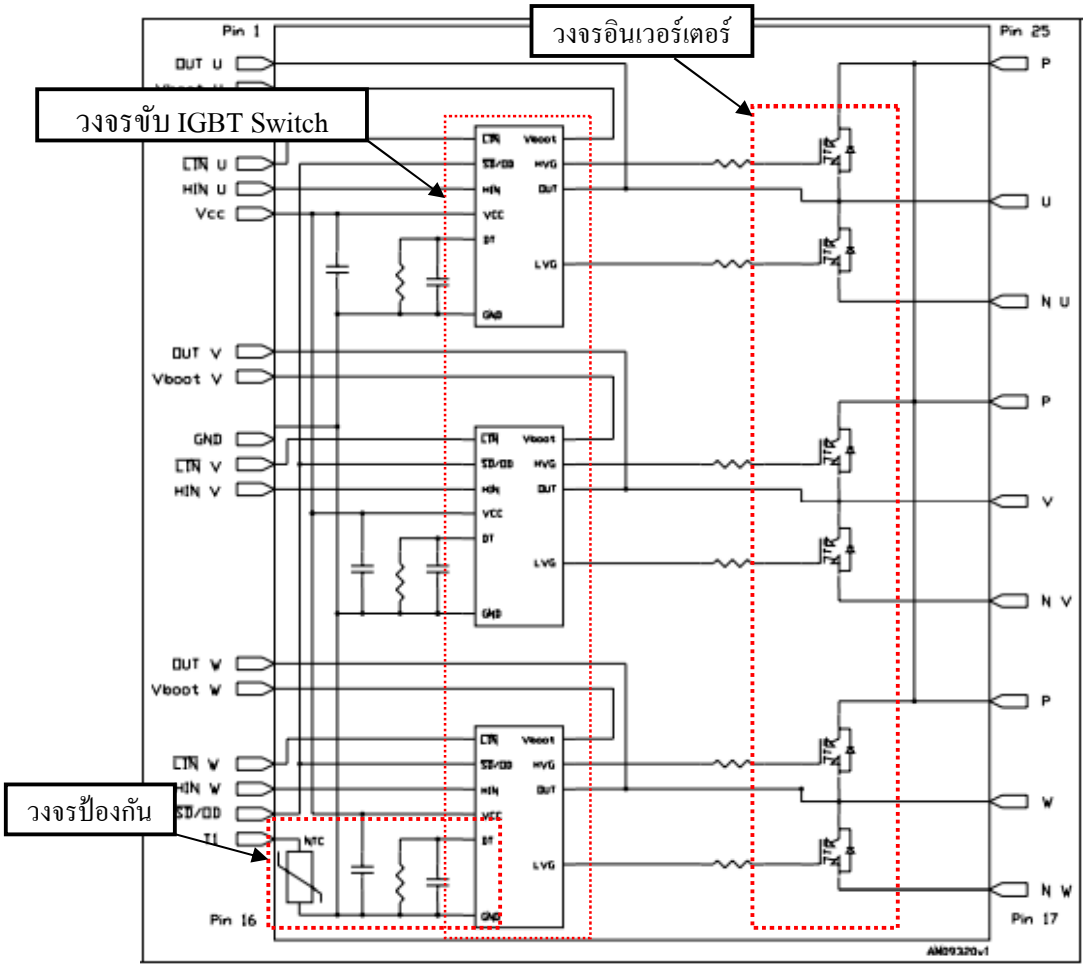
3.2.2 วงจรอินเวอร์เตอร์

วงจรอินเวอร์เตอร์ทำหน้าที่แปลงแรงดันไฟฟ้ากระแสตรงให้เป็นแรงดันไฟฟ้ากระแสสลับเพื่อขับเคลื่อนมอเตอร์ ในงานวิจัยนี้เลือกใช้ Intelligent Power Module (IPM) ของบริษัท ST เบอร์ STGIPS10K60T ซึ่งมีพิกัด กระแสไฟฟ้า 10A และ แรงดันไฟฟ้า 600V สามารถใช้งานกับมอเตอร์ไฟฟ้ากระแสสลับ 3 เฟส 220V ได้ Intelligent Power Module เป็นอุปกรณ์วงจรรวมชนิด Hybrid ที่รวมเอา วงจรอินเวอร์เตอร์ และชุดวงจรขับสวิตช์ไว้ภายในโมดูลเดียวกัน นอกจากนี้ยังมีวงจรป้องกันกระแสเกินและความร้อนเกินอีกด้วย ดังนั้นอุปกรณ์ IPM เบอร์ STGIPS10K60T จึงเป็นอุปกรณ์ที่สะดวกและง่ายในการออกแบบและประกอบวงจรอินเวอร์เตอร์ โดยลักษณะของอุปกรณ์ Intelligent Power Module เบอร์ STGIPS10K60T แสดงได้ดังรูปที่ 3.35

(ก) (ข)

รูปที่ 3.35 แสดงอินเวอร์เตอร์ (Intelligent Power Module: IPM) (ก) ด้านบน และ (ข) ด้านล่าง

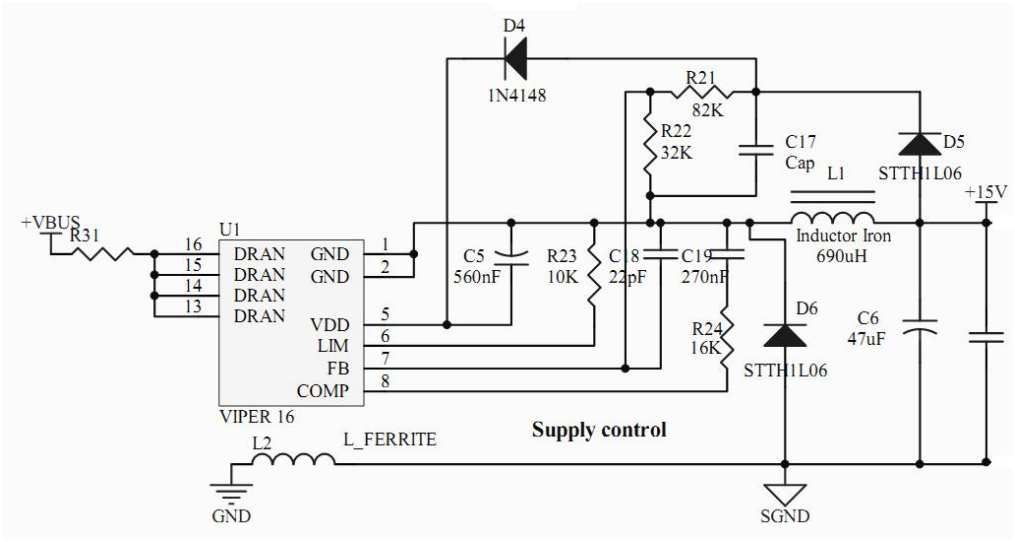
อุปกรณ์ Intelligent Power Module ใช้ไฟเลี้ยงวงจรระดับแรงดันไฟฟ้ากระแสตรง 15 V ภายในประกอบด้วย IGBT Switch จำนวนทั้งหมด 6 ตัว ชุดวงจรขับ IGBT Switch ซึ่งจะเชื่อมต่อกับขาสัญญาณพัลส์วิดมอดูเลชั่นจากตัวประมวลผล ชุดวงจรป้องกันกระแสเกินและความร้อนเกินโดยแสดงดังรูปที่ 3.36



รูปที่ 3.36 วงจรภายในของอุปกรณ์ IPM เบอร์ STGIPS10K60T

3.2.3 วงจรแหล่งจ่ายไฟฟ้ากระแสตรง

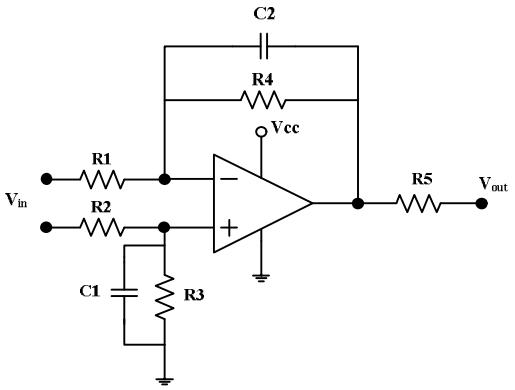
ในงานวิจัยนี้ใช้แหล่งจ่ายไฟฟ้าแบบสวิตช์ชิ่งชนิดบัคคอนเวอร์เตอร์(Buck Converter) เพื่อลดระดับแรงดันไฟฟ้ากระแสตรงจากวงจรเรียงกระแส 311 V ลงเหลือ 15 V เพื่อนำไปใช้เป็นไฟเลี้ยงให้กับ Intelligent Power Module ของชุดวงจรอินเวอร์เตอร์ โดยใช้ IC เบอร์ VIPER16 ของบริษัท ST แสดงดังรูปที่ 3.37 ส่วนไฟเลี้ยงตัวไมโครคอนโทรลเลอร์ และ วงจรปรับระดับสัญญาณของแรงดันไฟฟ้ารับจากแหล่งจ่ายไฟฟ้ากระแสตรงภายนอกระดับแรงดัน 3.3 V



รูปที่ 3.37 ชุดแหล่งจ่ายไฟฟ้ากระแสตรงหลัก (Main Switching Power Supply)

3.2.4 วงจรปรับระดับสัญญาณของแรงดันไฟฟ้า

เนื่องจากแรงดันไฟฟ้า V_a, V_b, V_c และ Vdc มีขนาดแรงดันไฟฟ้าสูงสุด 311 Vpeak ขณะที่การแปลงสัญญาณแอนะล็อกเป็นดิจิตอลในไมโครคอนโทรลเลอร์สามารถรับแรงดันไฟฟ้าได้เพียง 0-3.3 V เท่านั้น ดังนั้นจึงจำเป็นต้องมีการปรับระดับและขนาดสัญญาณให้เหมาะสม เพื่อที่จะให้ไมโครคอนโทรลเลอร์สามารถอ่านค่าได้อย่างถูกต้อง โดยงานวิจัยนี้ใช้วงจรขยายผลต่าง (Differential Amplifier) โดยได้ทำการออกแบบให้แรงดันไฟฟ้าเบส (V_{base}) อยู่ที่ 400 Vdc เป็นแรงดันไฟฟ้าขาเข้าของวงจรขยายผลต่างและแรงดันไฟฟ้าขาออก (V_{out}) คือ 3.3 V เพื่อให้ไมโครคอนโทรลเลอร์สามารถรับได้ เมื่อเลือกให้ค่าความต้านทานของ R1 กับ R2 เท่ากัน และ ค่าความต้านทานของ R3 กับ R4 เท่ากันแล้ว จะสามารถออกแบบวงจรขยายผลต่างได้ดังสมการที่ (3.2) โดยรูปวงจรขยายผลต่างแสดงดังรูปที่ 3.38



รูปที่ 3.38 วงจรขยายผลต่าง (Differential Amplifier)

$$V_{out} = \frac{R4}{R1} V_{in} \tag{3.2}$$

กำหนดให้ $V_{out} = 3.3V$, $V_{in} = V_{base} = 400V$ และ $R_4 = 10K\Omega$ แทนค่าดังกล่าวลงใน สมการ (3.2)
ดังนั้น $R1 = \frac{R4}{V_{out}} V_{in} = \frac{10K}{3.3V} 400 = 1.21M\Omega$

3.2.5 ไมโครคอนโทรลเลอร์

งานวิจัยนี้ใช้ไมโครคอนโทรเลอร์รุ่น dsPIC33FJ16GS504 ซึ่งมีฟังก์ชั่นการทำงานและคุณสมบัติที่สำคัญดังนี้

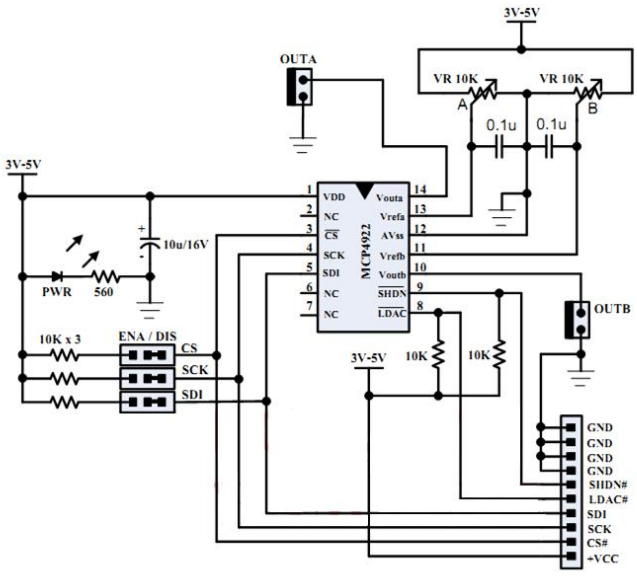
1.) ใช้ระดับแรงดันไฟฟ้ากระแสตรง 3.0 V ถึง 3.6 V สำหรับตัวประมวลและส่วนติดต่ออุปกรณ์ภายนอก
2.) หน่วยความจำแฟลต (Flash Memory) 64 Kbyte
3.) หน่วยความจำชั่วคราว (RAM) 16 Kbyte
4.) ตัวจับเวลา (Timer) 32 บิต
5.) รองรับการสื่อสารแบบอนุกรม 2 รูปแบบคือ SPI, SCI (UART)
6.) ฟังก์ชั่นการควบคุมมอเตอร์
    6.1 ฟังก์ชั่นการสร้างสัญญาณพัลส์วิดมอดูเลชั่น (PWM) 8 ช่อง
    6.2 ฟังก์ชั่นสำหรับเชื่อมต่อตัวตรวจจับตำแหน่งหรือเอ็นโคเดอร์ (Quadrature Encoder)
7.) ฟังก์ชั่นการแปลงสัญญาณอนาลอกเป็นดิจิตอล (Analog to Digital : ADC) ความละเอียด 12 บิต จำนวน 16 ช่อง ความเร็วการแปลงสัญญาณอนาลอกเป็นดิจิตอลสูงสุด 500 ksps

3.2.6 วงจรแปลงสัญญาณดิจิตอลเป็นแอนะลอก

วงจรแปลงสัญญาณดิจิตอลเป็นแอนะลอกในงานวิจัยนี้ใช้เพื่อทำการวัดและแสดงผล ค่าตัวแปรต่างๆ ในโปรแกรมที่จำเป็นในการควบคุมมอเตอร์ เช่น ความเร็วรอบที่วัดได้ ความเร็วรอบอ้างอิง ช่วงเวลาคอมมิวเตชั่น แรงดันไฟฟ้าแต่ละเฟสที่วัด จุด Zero Crossing และ จุดคอมมิวเตชั่นเป็นต้น โดยใช้ไอซีแปลงสัญญาณดิจิตอลเป็นแอนะลอก (Digital to Analog Converter : DAC) เบอร์ MCP4922 ของบริษัท Microchip ซึ่งมีคุณสมบัติดังนี้

1) สามารถรับข้อมูลดิจิตอลอินพุต มีความละเอียด 12 บิต
2) แรงดันอ้างอิง Vref เท่ากับ 5V
3) ใช้ระดับแรงดันไฟฟ้ากระแสตรง 2.7 V ถึง 5.5 V สำหรับทำงาน
4) มีช่องสัญญาณเอาท์พุตจำนวน 2 ช่อง

โดยงานวิจัยนี้ใช้ไอซี MCP4922 จำนวน 2 ตัว ทำให้มีช่องสัญญาณเอาท์พุตจำนวน 4 ช่องด้วยกัน การทำงานของไอซี MCP4922 รับค่าตัวแปรที่ต้องการแสดงผลจากตัวไมโครคอนโทรเลอร์ผ่านสัญญาณการสื่อสารแบบอนุกรม SPI เพื่อติดต่อกับไอซี MCP4922 หลังจากนั้นทำการแปลงสัญญาณดิจิตอลให้เป็นสัญญาณแอนะลอกเพื่อทำการวัดแสดงผล โดยวงจรแปลงสัญญาณดิจิตอลเป็นแอนะลอกแสดงดังรูปที่ 3.39



รูปที่ 3.39 วงจรแปลงสัญญาณดิจิตอลเป็นแอนะลอกโดยใช้ไอซี MCP4922