บทที่ 3
วิธีดำเนินการวิจัย

3.1 บทนำ

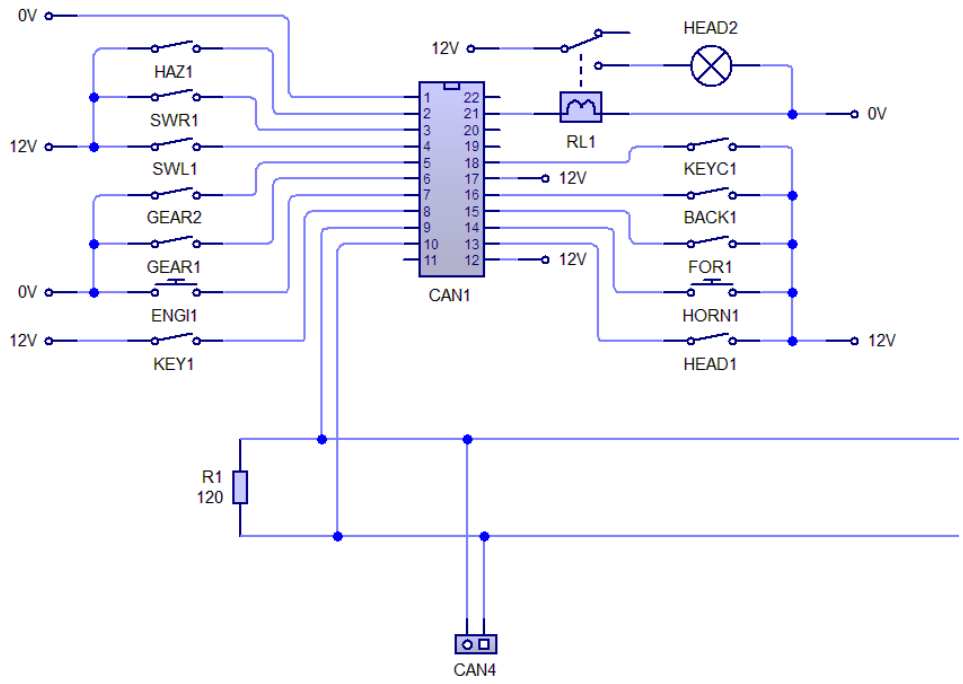
ในการศึกษานี้ กระบวนการและขั้นตอนการดำเนินการวิจัยได้รับการวางแผนและดำเนินการอย่างเป็นระบบ โดยครอบคลุมถึงการออกแบบการทดลอง การเก็บรวบรวมข้อมูล การวิเคราะห์ข้อมูล ตลอดจนการประยุกต์ใช้เครื่องมือและอุปกรณ์ต่าง ๆ เพื่อให้ผลลัพธ์ที่ได้มีความถูกต้องและน่าเชื่อถือ วิธีการดำเนินการวิจัยถูกกำหนดขึ้นอย่างมีแบบแผน เพื่อให้สามารถนำผลลัพธ์ไปใช้ในการพัฒนาและแก้ไขปัญหาตามวัตถุประสงค์ของการศึกษา งานวิจัยนี้ใช้เทคโนโลยี CAN BUS ร่วมกับอุปกรณ์ CAN I/O PLC และซอฟต์แวร์ MRS Developers Studio ในการสื่อสารข้อมูลภายในระบบยานยนต์ โดยอ้างอิงมาตรฐาน SAE J1939 ซึ่งใช้ควบคุมระบบสื่อสารและกำหนดที่อยู่ของข้อมูล นอกจากนี้ยังมีการตรวจสอบและทดสอบการทำงานของระบบเพื่อให้มั่นใจว่าข้อมูลที่ส่งผ่านระบบมีความถูกต้องและสามารถนำไปใช้งานได้จริง เนื่องจาก กล่อง CAN I/O PLC ไม่สามารถส่งสัญญาณอนาล็อก ได้โดยตรง จึงใช้หลักการ Low-pass filter เพื่อแปลงสัญญาณ PWM ให้เป็นสัญญาณอนาล็อก เพื่อให้สามารถสื่อสารกับกล่องควบคุมมอเตอร์ได้ นอกจากนี้ยังมีการวัดกำลังไฟฟ้าผ่าน Power meter ร่วมกับ V-Box และส่งข้อมูลไปยัง V-Net เพื่อบันทึกข้อมูลการใช้พลังงานแบบเรียลไทม์ เพื่อนำไปวิเคราะห์ผลการทดลอง สำหรับการแสดงผลข้อมูล จอสัมผัส ATD3.5-S3 ขนาด 3.5 นิ้ว ซึ่งเชื่อมต่อกับ บอร์ด ESP32-S3 ถูกนำมาใช้เพื่อแสดงผลข้อมูลแบบเรียลไทม์จากระบบ CAN BUS ข้อมูลที่แสดงรวมถึงสถานะการทำงานของระบบ เช่น สัญญาณจากปุ่มสตาร์ท สัญญาณไฟเลี้ยว (ซ้าย-ขวา-ฉุกเฉิน) สัญญาณคันเร่ง และระบบเบรกของยานพาหนะ นอกจากนี้ จอ ATD3.5-S3 ยังสามารถแสดงการสื่อสารระหว่างโมดูล CAN ทั้งสามโมดูล ที่ติดตั้งในตัวรถ ซึ่งช่วยให้การตรวจสอบและทดสอบการทำงานของระบบ CAN BUS มีความแม่นยำและละเอียดรอบคอบมากขึ้น เพื่อให้มั่นใจว่าข้อมูลที่ได้รับและผลลัพธ์ของการวิจัยมีความถูกต้องและสามารถนำไปใช้ได้อย่างมีประสิทธิภาพ

3.2 การออกแบบระบบสื่อสารในยานยนต์ CAN BUS (Controller Area Network)

งานวิจัยนี้ได้ใช้มาตรฐาน SAE J1939 ในการออกแบบระบบสื่อสารภายในยานยนต์ โดยเลือกใช้ High-Speed CAN เป็นรูปแบบในการเชื่อมต่อระหว่างโมดูลต่าง ๆ ซึ่งระบบนี้สามารถรองรับ

การเชื่อมต่อโมดูล CAN ได้สูงสุดถึง 30 โมดูล และจำกัดความยาวของวงจรสื่อสารไม่เกิน 40 เมตร เพื่อให้เหมาะสมกับการใช้งาน งานวิจัยนี้ได้นำโมดูล CAN I/O PLC จำนวน 3 โมดูลมาใช้ในการจำลองระบบสื่อสาร โดยแต่ละโมดูลมีหน้าที่แตกต่างกันไปในการจัดการข้อมูลและสัญญาณต่าง ๆ ภายในระบบ โดยมีการกำหนดหน้าที่ของแต่ละโมดูลแตกต่างกันออกไป ดังนี้

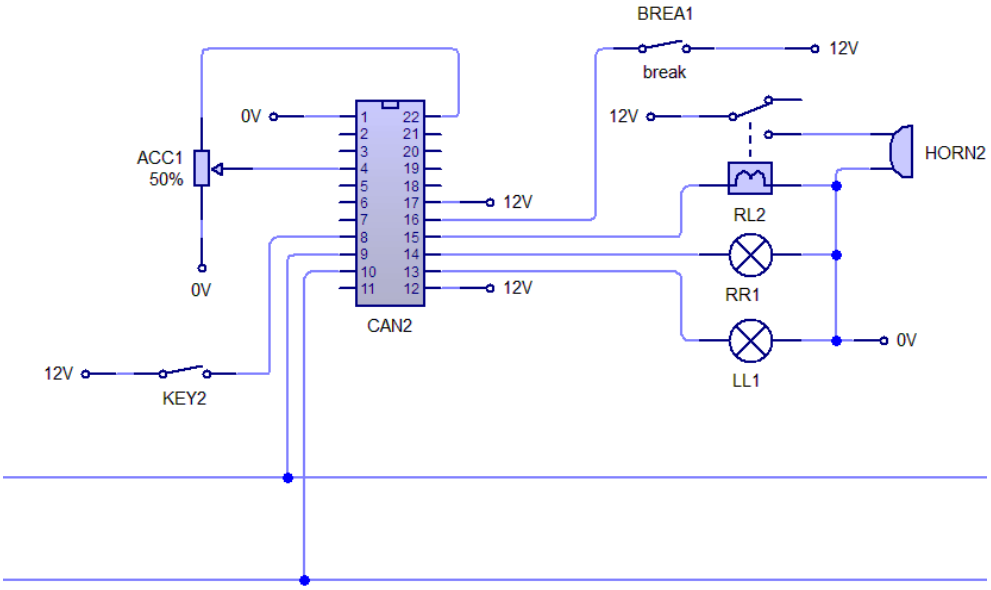
โมดูลที่ 1 จะถูกติดตั้งภายในตัวรถและมีหน้าที่สำคัญในการรับสัญญาณจากสวิตช์ควบคุมต่าง ๆ ที่ติดตั้งในระบบของรถ ATV โมดูลนี้จะทำการรับข้อมูลจากสวิตช์ควบคุมที่ใช้ในการปรับการทำงานของระบบต่าง ๆ ภายในรถ เช่น การเปิด-ปิดไฟหน้า สัญญาณไฟเลี้ยว สัญญาณเดินหน้าถอยหลัง โหมดการทำงาน และการสั่งงานฟังก์ชันอื่น ๆ ที่เกี่ยวข้อง จากนั้นโมดูลจะส่งข้อมูลเหล่านี้ไปยังโมดูลอื่น ๆ ภายในระบบผ่านช่องทางสื่อสาร CAN โดยรูปที่ 3.1 แสดงวงจรการเชื่อมต่อของโมดูลนี้ ซึ่งจะสามารถแสดงให้เห็นถึงการเชื่อมต่อระหว่างสวิตช์ควบคุมต่าง ๆ และโมดูล CAN I/O PLC เพื่อให้ระบบทำงานได้อย่างถูกต้อง



รูปที่ 3.1 แผนผังการต่อสายไฟ ของโมดูล CAN ที่ 1

โมดูลที่ 2 จะถูกติดตั้งที่ด้านหน้าของรถ ATV และมีบทบาทสำคัญในการรับสัญญาณจากเซนเซอร์คันเร่งและเซนเซอร์เบรก โดยจะทำการตรวจจับสถานะต่าง ๆ ของระบบ เช่น การกดคันเร่งหรือการเบรก เพื่อส่งข้อมูลไปยังโมดูลอื่น ๆ ในระบบเพื่อให้สามารถประมวลผลและดำเนินการต่อไป

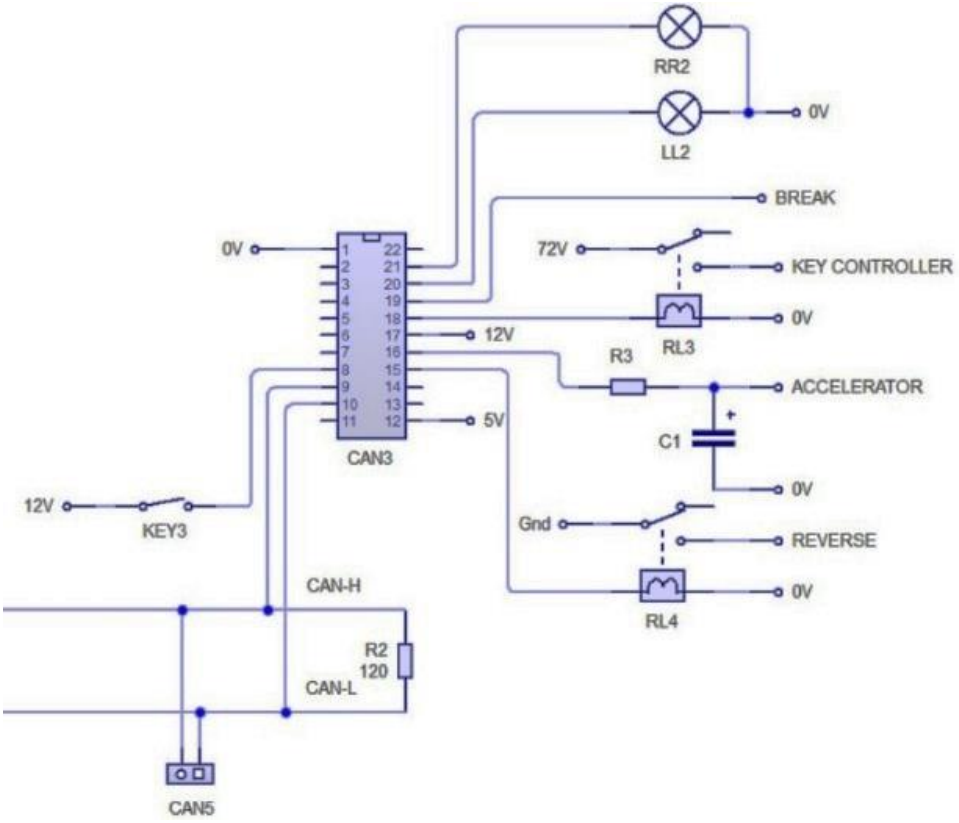
ได้อย่างถูกต้อง นอกจากนี้ โมดูลที่ 2 ยังมีหน้าที่ในการส่งออกสัญญาณไฟเลี้ยวและแตรที่ได้รับค่าจากโมดูลที่ 1 เพื่อให้การแสดงผลของไฟเลี้ยวและการทำงานของแตรตรงตามการควบคุมของผู้ขับขี่ การเชื่อมต่อวงจรของโมดูลนี้จะแสดงในรูปที่ 3.2 ซึ่งจะช่วยให้เห็นการเชื่อมต่อระหว่างเซนเซอร์ต่าง ๆ ที่รับสัญญาณจากคันเร่งและเบรก รวมถึงการรับส่งสัญญาณไฟเลี้ยวและแตรจากโมดูลที่ 1 ผ่านระบบ CAN อย่างไรก็ตาม ในกรณีของแตรนั้นมีการใช้กระแสไฟฟ้าสูง ซึ่งทำให้โมดูล CAN ไม่สามารถจ่ายไฟฟ้าให้กับแตรได้โดยตรง ดังนั้นจึงจำเป็นต้องใช้รีเลย์ในการช่วยสั่งงานแตร เพื่อป้องกันไม่ให้โมดูล CAN ได้รับความเสียหายจากการจ่ายไฟที่มากเกินไป



รูปที่ 3.2 แผนผังการต่อสายไฟ ของโมดูล CAN ที่ 2

โมดูลที่ 3 จะถูกติดตั้งที่ด้านท้ายของรถ ATV และมีหน้าที่สำคัญในการส่งสัญญาณที่ได้รับจากทั้งโมดูลที่ 1 และโมดูลที่ 2 เพื่อนำไปควบคุมกล่องควบคุมมอเตอร์และอุปกรณ์ต่าง ๆ ภายในระบบของรถ เช่น การควบคุมคันเร่ง ระบบเบรก สวิตช์กุญแจ เกียร์เดินหน้า-ถอยหลัง โหมดการทำงาน และไฟเลี้ยวด้านหลัง โมดูลนี้จะทำหน้าที่ประมวลผลข้อมูลที่ได้รับจากทั้งสองโมดูล ก่อนที่จะส่งสัญญาณไปยังอุปกรณ์ที่เกี่ยวข้อง เพื่อให้สามารถทำงานได้ตามที่ผู้ขับขี่ต้องการ ในส่วนของสัญญาณคันเร่งนั้น เนื่องจากโมดูล CAN ไม่สามารถส่งคำสั่งเป็นสัญญาณอนาล็อกได้ แต่สามารถส่งเป็นสัญญาณ PWM (Pulse Width Modulation) ได้ จึงต้องมีการแปลงรูปสัญญาณจาก PWM เป็นสัญญาณอนาล็อก โดยการใช้วงจร Low Pass Filter เพื่อทำการแปลงสัญญาณดังกล่าวให้เหมาะสมกับการควบคุมอุปกรณ์ที่ต้องการสัญญาณอนาล็อก การเชื่อมต่อวงจรของโมดูลนี้จะแสดงในรูปที่ 3.3

ซึ่งจะช่วยให้เห็นการเชื่อมต่อระหว่างโมดูลที่ 3 กับอุปกรณ์ต่าง ๆ ที่อยู่ในระบบของรถ เพื่อให้ระบบสามารถทำงานได้สอดคล้องกับการควบคุมของผู้ขับขี่

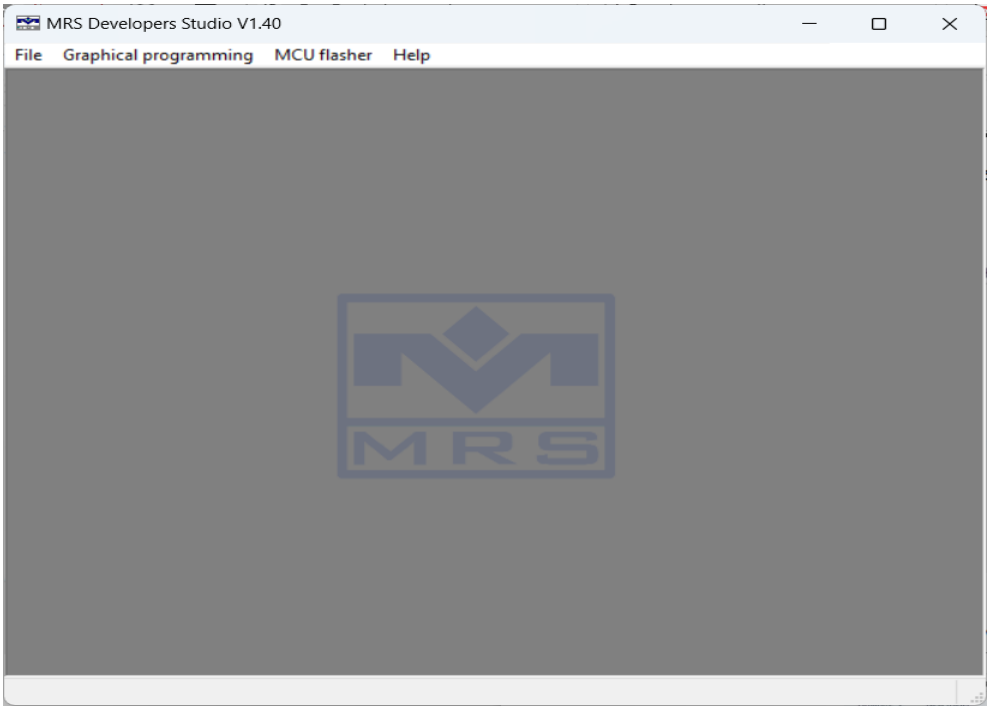


รูปที่ 3.3 แผนผังการต่อสายไฟ ของโมดูล CAN ที่ 3

การออกแบบระบบสื่อสารนี้ช่วยให้โมดูลทั้งสามสามารถทำงานร่วมกันได้อย่างมีประสิทธิภาพภายใต้มาตรฐาน SAE J1939 ซึ่งเป็นมาตรฐานที่รองรับการสื่อสารในระบบยานยนต์ โดยการเลือกใช้เทคโนโลยี High-Speed CAN ช่วยให้ระบบสามารถส่งข้อมูลระหว่างโมดูลต่าง ๆ ได้อย่างรวดเร็วและมีเสถียรภาพสูง การกำหนดข้อจำกัดทางด้านความยาววงจรและจำนวนโมดูลที่เชื่อมต่อได้ ทำให้ระบบสามารถทำงานร่วมกันได้อย่างเหมาะสมและไม่เกิดปัญหาความล่าช้าหรือการสูญเสียข้อมูล ระบบนี้ช่วยให้การทำงานของอุปกรณ์ต่าง ๆ ในรถ ATV เป็นไปอย่างราบรื่นและมีความแม่นยำ เนื่องจากโมดูลทั้งสามมีการประมวลผลข้อมูลและสั่งงานได้อย่างมีประสิทธิภาพ รวมถึงการเชื่อมต่อที่เสถียรและรองรับการทำงานที่หลากหลาย เช่น การควบคุมคันเร่ง เบรก สวิตช์กุญแจ เกียร์ ไฟเลี้ยว และอื่น ๆ ได้อย่างถูกต้องและทันเวลา ทำให้การทำงานของระบบสื่อสารภายในรถ ATV มีความเสถียรและเหมาะสมต่อการใช้งานในสภาพแวดล้อมต่าง ๆ

3.3 การออกแบบการทำงานของยานยนต์

การออกแบบวงจรการทำงานของแต่ละโมดูล CAN I/O PLC ได้รับการกำหนดให้มีการทำงานที่แตกต่างกันไปตามหน้าที่เฉพาะของแต่ละโมดูล ซึ่งจำเป็นต้องมีการเขียนโปรแกรมควบคุมเพื่อทำให้การทำงานของโมดูลเหล่านี้สอดคล้องกันและทำงานได้อย่างมีประสิทธิภาพ ผ่านซอฟต์แวร์ MRS Developers Studio เครื่องมือที่ใช้ในการตั้งค่าและเขียนโปรแกรมสำหรับโมดูล CAN I/O PLC ยี่ห้อ MRS โดยซอฟต์แวร์ดังกล่าวช่วยให้สามารถตั้งค่าการทำงานของแต่ละโมดูลได้อย่างละเอียดและตรงตามความต้องการของการใช้งานในแต่ละกรณี โครงสร้างของซอฟต์แวร์ที่ใช้งานได้แสดงอยู่ในรูปที่ 3.4



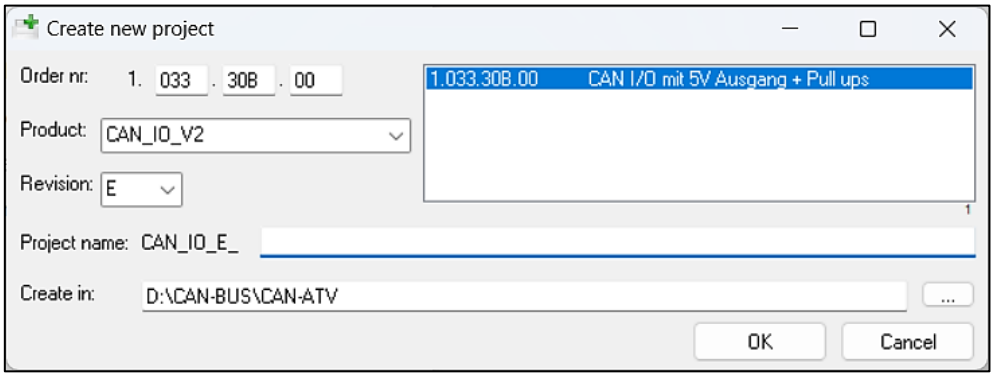
รูปที่ 3.4 หน้าต่างโปรแกรม MRS Developers Studio

3.3.1 ขั้นตอนการสร้างโปรเจกต์ใหม่ในการออกแบบการควบคุมระบบการทำงาน ATV

การเริ่มต้นสร้างโปรเจกต์ใหม่สำหรับ CAN I/O PLC ยี่ห้อ MRS จำเป็นต้องเริ่มจากการกำหนดรุ่นของโมดูลที่ใช้ให้ถูกต้อง โดยในกรณีนี้จะต้องระบุรุ่นเป็น Order nr: 1.033.30B.00 ซึ่งเป็นรุ่นที่กำหนดให้ใช้ Product: CAN_IO_V2 และ Revision: E ตามที่แสดงในรูปที่ 3.5 การกำหนดค่ารุ่นเหล่านี้มีความสำคัญอย่างยิ่ง เนื่องจากแต่ละรุ่นอาจมีคุณสมบัติทางเทคนิคหรือข้อกำหนดที่แตกต่างกัน การตั้งค่าเริ่มต้นให้ตรงกับฮาร์ดแวร์ที่ใช้งานจริงจะช่วยลดความผิดพลาดที่

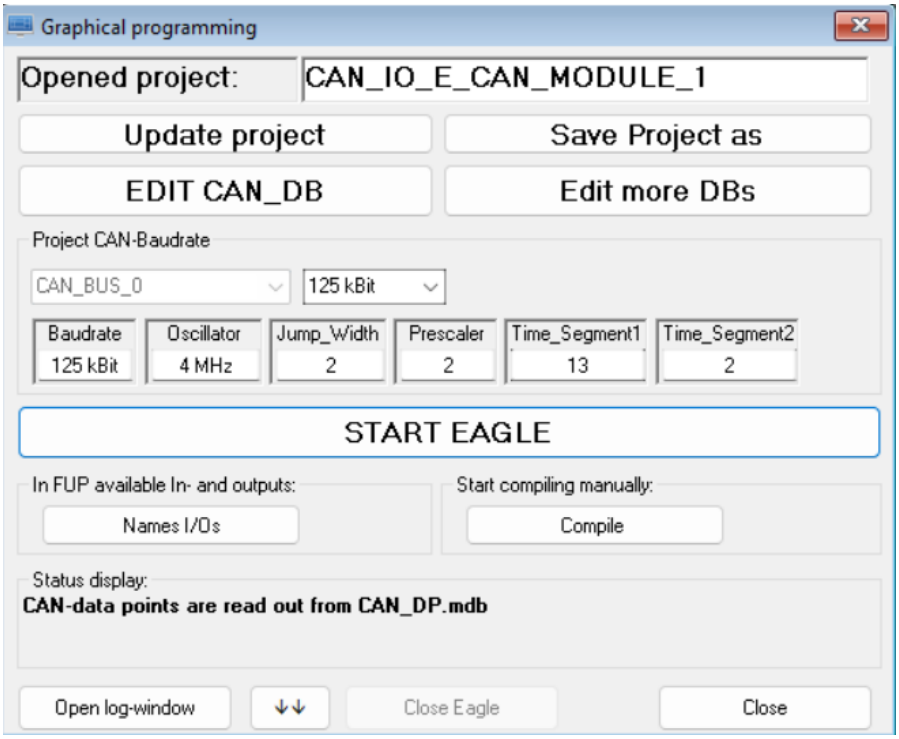
อาจเกิดขึ้นในการเขียนโปรแกรม และช่วยให้การเชื่อมต่อกับระบบ CAN BUS เป็นไปอย่างถูกต้องและมีเสถียรภาพ

นอกจากนี้ การกำหนดค่าที่ตรงตามฮาร์ดแวร์ยังช่วยให้ซอฟต์แวร์ MRS Developers Studio สามารถประมวลผลคำสั่งและควบคุมฟังก์ชันต่าง ๆ ของโมดูลได้อย่างแม่นยำ ซึ่งส่งผลให้การทำงานของระบบเป็นไปตามแผนที่วางไว้



รูปที่ 3.5 หน้าต่างการตั้งค่าโมดูล PLC CAN I/O

เมื่อสร้างโปรเจกต์ใหม่ในซอฟต์แวร์ MRS Developers Studio หน้าต่าง Graphical Programming จะปรากฏขึ้น แสดงส่วนต่าง ๆ ของโปรเจกต์ที่กำลังเปิดใช้งาน ดังรูป 3.6 ผู้ใช้งานสามารถออกแบบและกำหนดโปรแกรมผ่านบล็อกคำสั่งต่าง ๆ ได้อย่างสะดวก



รูปที่ 3.6 หน้าต่าง Graphical Programming

การทำงานสามารถตั้งค่าโปรแกรมได้ดังนี้

1) Opened Project: แสดงชื่อของโปรเจกต์ที่กำลังเปิดใช้งานอยู่ เช่น "CAN_IO_E_CAN _MODULE_1"

2) Update Project / Save Project as: ใช้เพื่ออัพเดตหรือบันทึกโปรเจกต์ปัจจุบันในชื่อใหม่

3) Edit CAN_DB / Edit more DBs: ใช้สำหรับแก้ไขข้อมูลฐานข้อมูล CAN หรือฐานข้อมูลอื่นที่เกี่ยวข้อง

4) Project CAN-Baud rate: กำหนดอัตราความเร็วในการสื่อสารของ CAN BUS (Baud rate) ซึ่งในตัวอย่างคือ 125 Kbit

5) Baud rate / Oscillator / Jump Width / Pre Scaler / Time Segment1 / Time Segment2: เป็นพารามิเตอร์ที่ใช้ในการตั้งค่าความเร็วและการสื่อสารของ CAN BUS

6) Start Eagle: ใช้เพื่อเริ่มการทำงานของโปรเจกต์ที่สร้างขึ้น

7) Names I/O: ชื่อของอินพุต/เอาต์พุตที่ใช้งานในโปรแกรม

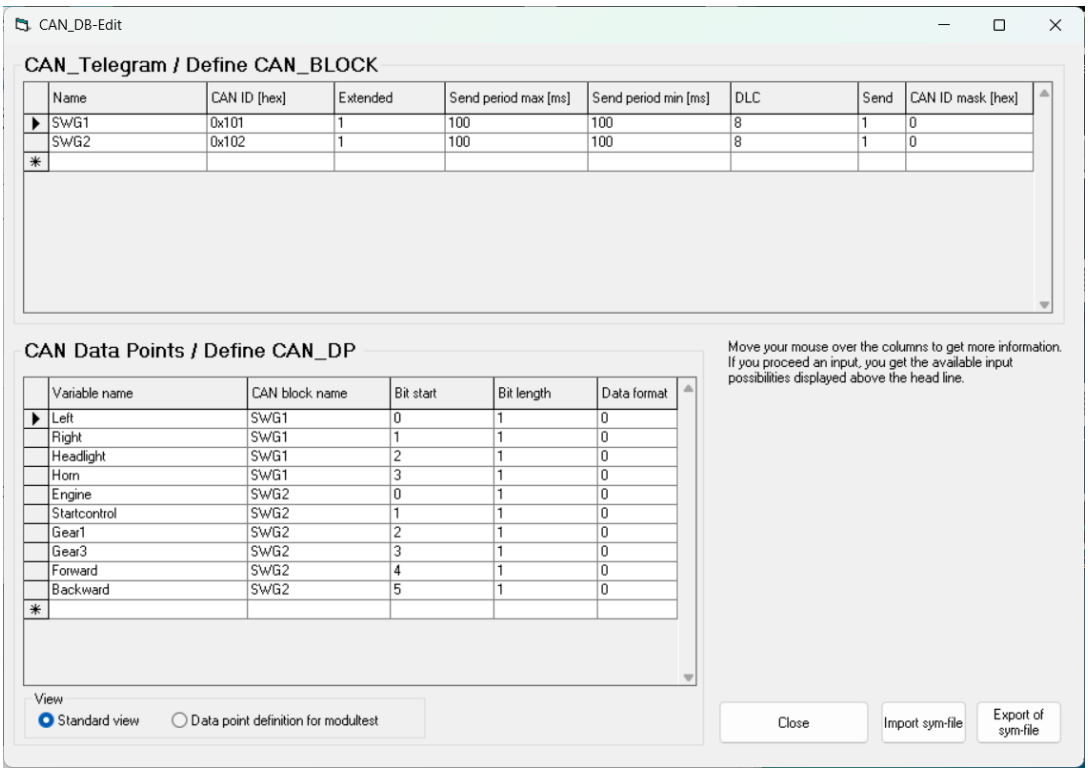
8) Compile: ใช้เพื่อคอมไพล์โปรแกรมก่อนที่จะทำการอัปโหลด

9) Status Display: แสดงสถานะปัจจุบันของข้อมูลที่ถูกอ่านจากฐานข้อมูล

3.3.2 การตั้งค่า CAN-ID และตัวแปรสำหรับการรับส่งข้อมูล

การพัฒนาโปรแกรมควบคุมโมดูล CAN I/O PLC จำเป็นต้องดำเนินการตั้งค่าการรับส่งข้อมูลในหน้า Edit CAN_DB ซึ่งเป็นหน้าต่างสำหรับการกำหนดข้อมูลที่ต้องการรับหรือส่งผ่านระบบ CAN BUS โดยผู้ใช้งานต้องสร้างตัวแปรที่ใช้ในการจัดเก็บข้อมูล เพื่อให้ระบบสามารถประมวลผลและจัดเก็บข้อมูลการสื่อสารได้อย่างถูกต้องและแม่นยำ หลังจากการตั้งค่าข้อมูลและตัวแปรเรียบร้อยแล้ว จึงสามารถดำเนินการเขียนโปรแกรมและตั้งค่าฟังก์ชันที่จำเป็นสำหรับการควบคุมระบบของรถ ATV

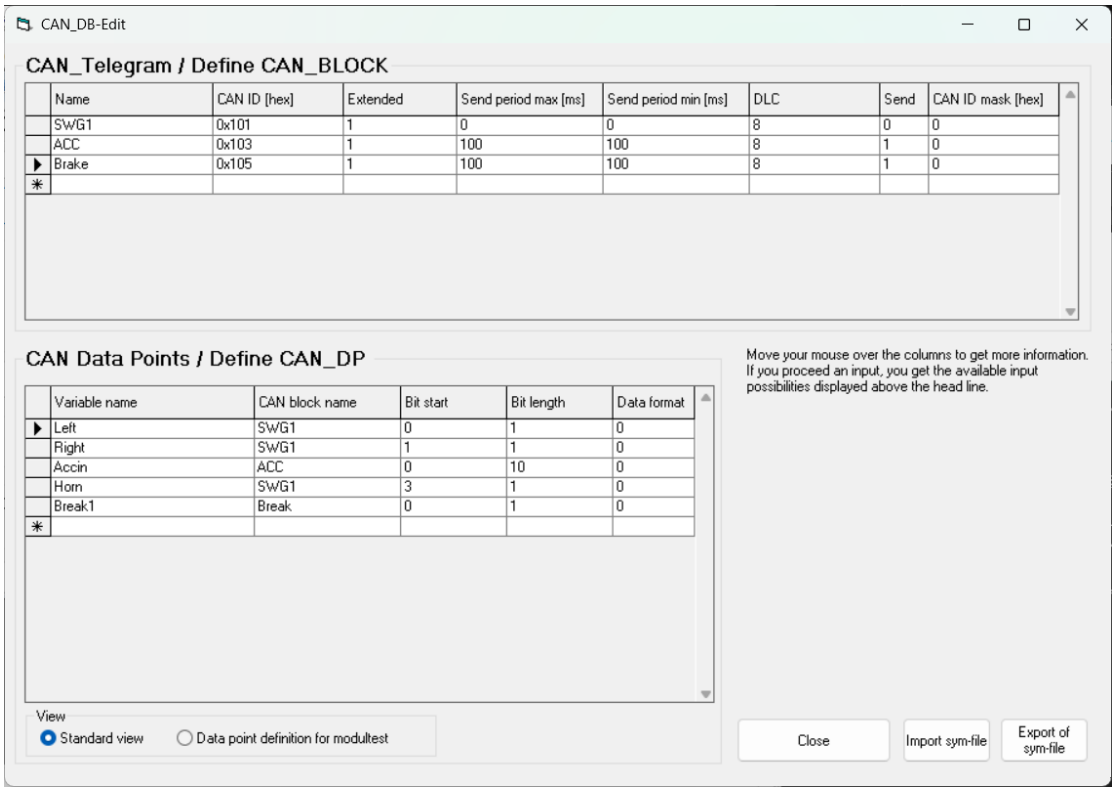
ในส่วนของการตั้งค่า CAN-ID และตัวแปรสำหรับการรับส่งข้อมูลในโมดูล CAN ที่ 1 ซึ่งรับค่าจากสวิตช์เพียงอย่างเดียว จะถูกแบ่งออกเป็นหมวดหมู่ของสวิตช์ทั้งหมด 2 กลุ่ม โดยกลุ่มแรกมีชื่อว่า SWG1 ซึ่งถูกกำหนดชื่อตัวแปรแทน CAN-ID: 0X101 โดยมีการกำหนดความเร็วในการส่งข้อมูลที่ 100 ms ในตัวแปร SWG1 หรือ CAN-ID: 0X101 จะเก็บข้อมูลที่เป็นส่วนของสวิตช์ควบคุมสัญญาณไฟภายในตัวรถ เช่น สวิตช์ไฟเลี้ยว ไฟหน้า แตร เป็นต้น ข้อมูลเหล่านี้จะถูกรวมอยู่ใน CAN-ID นี้ เพื่อแยกเป็นกลุ่มของข้อมูลที่เกี่ยวข้องกับการควบคุมไฟต่าง ๆ ภายในรถ และในกลุ่มที่ 2 มีชื่อว่า SWG2 ซึ่งถูกกำหนดชื่อตัวแปรแทน CAN-ID: 0X102 จะเก็บข้อมูลที่เป็นส่วนของสวิตช์ควบคุมการทำงานของมอเตอร์ขับเคลื่อน เช่น สวิตช์สตาร์ท สวิตช์เลือกโหมด และสวิตช์เดินหน้า-ถอยหลังเป็นต้น ซึ่งสวิตช์แต่ละตัวก็จะถูกตัวตัวแปรย่อยเพื่อไว้เก็บข้อมูลแต่ละตัวต่างกันออกไป การตั้งค่า CAN-ID และรูปแบบการรับส่งข้อมูลการสื่อสารแสดงในโมดูล CAN ที่ 1 แสดงดังรูปที่ 3.7



รูปที่ 3.7 หน้าต่าง Edit CAN_DB สำหรับการตั้งค่าการสื่อสารในโมดูล CAN ที่ 1

ในส่วนของการตั้งค่า CAN-ID และตัวแปรสำหรับการรับส่งข้อมูลใน โมดูล CAN ที่ 2 โมดูลนี้จะมีการกำหนดให้รับค่าของสวิตช์ไฟเลี้ยวที่ได้รับจาก โมดูล CAN ที่ 1 โดยโมดูล CAN ที่ 2 จะรับค่าจาก CAN-ID: 0X101 ซึ่งเป็นที่อยู่ของข้อมูลสวิตช์ไฟเลี้ยวที่ส่งมาจากโมดูล CAN ที่ 1 นั่นหมายความว่าโมดูล CAN ที่ 2 จะต้องสามารถรับข้อมูลจากโมดูล CAN ที่ 1 ซึ่งมีการกำหนดค่า CAN-ID เป็น 0X101 เพื่อให้สามารถรับข้อมูลที่ถูกส่งจากโมดูลนั้นได้ นอกจากนี้ โมดูล CAN ที่ 2 ยังมีการสร้าง CAN-ID ใหม่ ขึ้นมาสำหรับการสื่อสารกับโมดูล CAN อื่น ๆ ในระบบ โดยมี CAN-ID: 0X103 ซึ่งใช้ชื่อว่า ACC สำหรับส่งข้อมูลเกี่ยวกับการควบคุมคันเร่ง และ CAN-ID: 0X105 ในชื่อ Brake เพื่อใช้ในการส่งข้อมูลที่เกี่ยวข้องกับสัญญาณเบรก โมดูล CAN ที่ 2 มีหน้าที่รับข้อมูลจากโมดูล CAN ที่ 1 และส่งข้อมูลไปยังโมดูล CAN อื่น ๆ โดยผ่าน CAN-ID ที่ตั้งไว้ ในกรณีของ ตัวแปรย่อย Accin ซึ่งเป็นตัวแปรที่ใช้เก็บข้อมูลเกี่ยวกับคันเร่ง ค่าของคันเร่งนั้นเป็น สัญญาณอนาล็อก ซึ่งมีลักษณะเป็นค่าต่อเนื่อง ไม่ใช่แค่เปิดหรือปิดเหมือนสวิตช์ ทำให้ต้องใช้พื้นที่ในการเก็บข้อมูลมากขึ้น ตัวแปร Accin จึงถูกกำหนดให้สามารถเก็บข้อมูลได้มากถึง 10 ตำแหน่ง เพื่อรองรับการส่งข้อมูลที่มีหลายค่าในตัวแปรเดียว เมื่อเทียบกับข้อมูลจากสวิตช์ที่มีแค่สถานะเปิดหรือปิด ซึ่งใช้พื้นที่ในการเก็บข้อมูลเพียงแค่

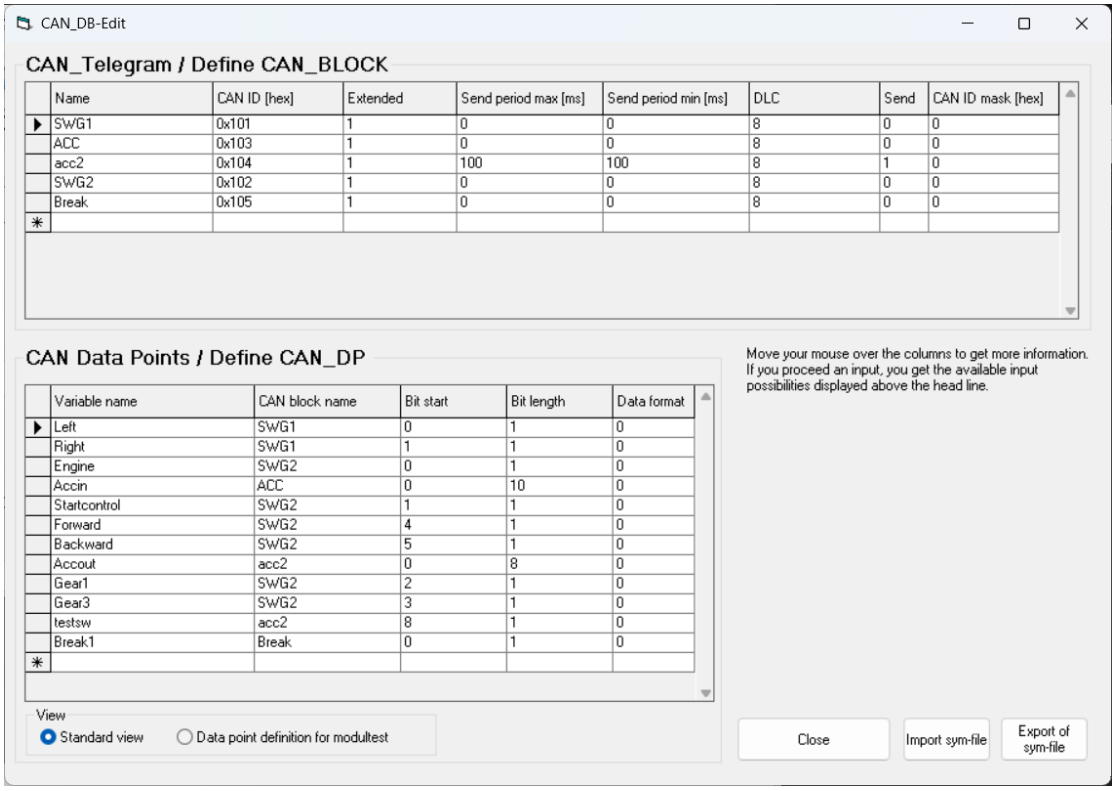
1 ตำแหน่ง การตั้งค่า CAN-ID และรูปแบบการรับส่งข้อมูลต่าง ๆ ในการสื่อสารของโมดูล CAN ที่ 2 จะถูกแสดงในรูปที่ 3.8



รูปที่ 3.8 หน้าต่าง Edit CAN_DB สำหรับการตั้งค่าการสื่อสารในโมดูล CAN ที่ 2

ในส่วนของการตั้งค่า CAN-ID และตัวแปรสำหรับการรับส่งข้อมูลใน โมดูล CAN ที่ 3 โมดูลนี้จะไม่มีการรับค่าจากสวิตช์หรือเซ็นเซอร์ แต่จะทำหน้าที่ในการส่งข้อมูลออกจากตัวโมดูลเพื่อไปควบคุมการทำงานของกล่องควบคุมมอเตอร์ โมดูล CAN ที่ 3 จะทำการรับคำสั่งจาก โมดูล CAN ที่ 1 และ โมดูล CAN ที่ 2 มาเพื่อทำการประมวลผลก่อนที่จะส่งข้อมูลไปยังกล่องควบคุมมอเตอร์ โดยการทำงานในลักษณะนี้ทำให้โมดูล CAN ที่ 3 จะเป็นตัวกลางในการประสานข้อมูลจากโมดูลที่ 1 และ 2 เพื่อควบคุมการทำงานของมอเตอร์ การตั้งค่า CAN-ID และตัวแปรสำหรับการรับส่งข้อมูลใน โมดูล CAN ที่ 3 จะมีการรับข้อมูลจากโมดูลอื่น ๆ เท่านั้น โดยจะรับค่า CAN-ID: 0X101 และ CAN-ID: 0X102 จากโมดูล CAN ที่ 1 ซึ่งเป็นค่าของสวิตช์ควบคุมภายในตัวรถ เช่น สวิตช์ไฟเลี้ยว ไฟหน้า หรือแตร และจะเก็บข้อมูลเหล่านี้ไว้ในตัวแปรต่าง ๆ จากนั้นโมดูล CAN ที่ 3 ยังจะรับค่า CAN-ID: 0X103 และ CAN-ID: 0X105 จากโมดูล CAN ที่ 2 ซึ่งเป็นค่าคันเร่งและสัญญาณเบรก และเก็บข้อมูลเหล่านี้ไว้ในตัวแปรเช่นเดียวกัน การตั้งค่า CAN-ID และรูปแบบการรับส่งข้อมูลในการสื่อสารของโมดูล CAN

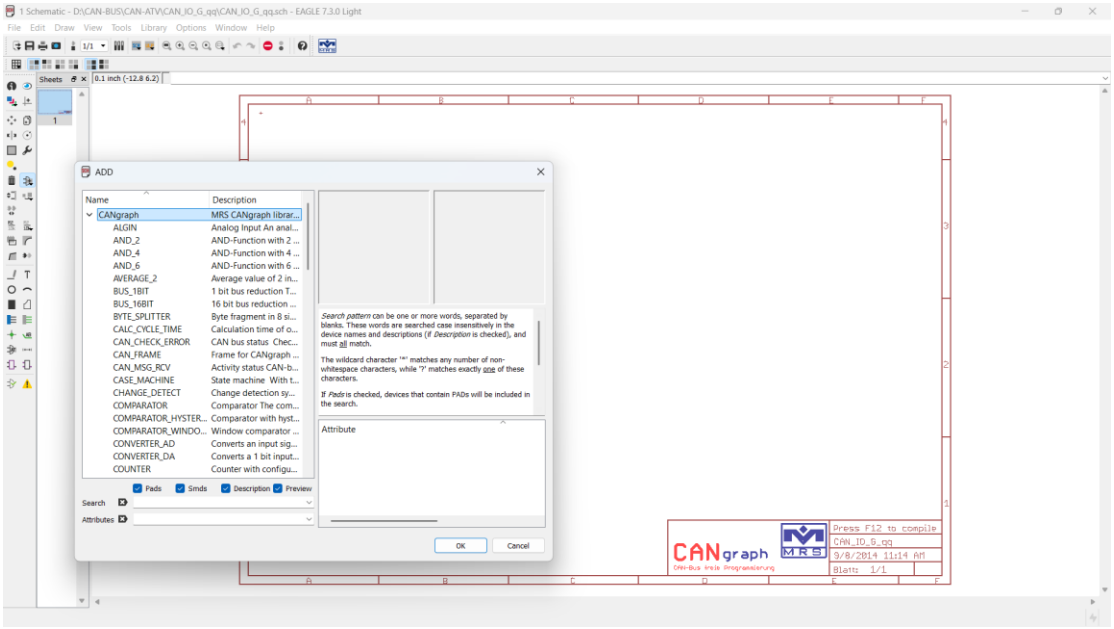
ที่ 3 จะแสดงในรูปที่ 3.9 โดยหลังจากการตั้งค่าต่าง ๆ เสร็จสิ้นแล้ว ข้อมูลที่ถูกเก็บไว้ในตัวแปรเหล่านี้จะถูกนำไปใช้ในการเขียนโปรแกรมควบคุมการทำงานของมอเตอร์ในขั้นตอนถัดไป



รูปที่ 3.9 หน้าต่าง Edit CAN_DB สำหรับการตั้งค่าการสื่อสารในโมดูล CAN ที่ 3

3.3.3 การเขียนโปรแกรมควบคุม CAN I/O PLC

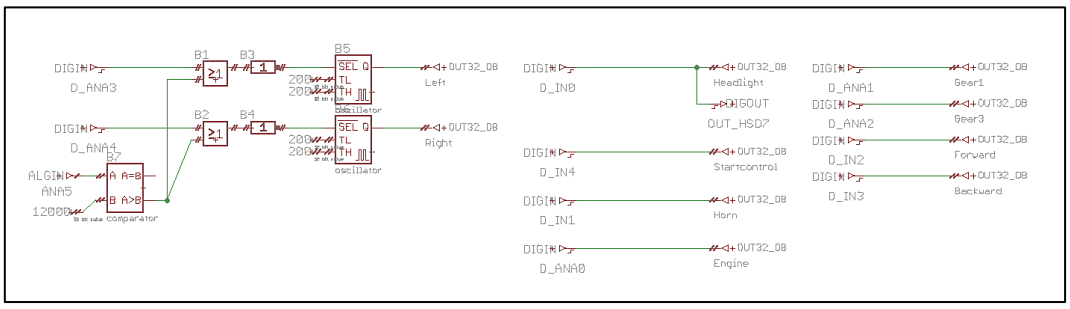
จากตัวอย่างในรูปที่ 3.10 การเขียนโปรแกรมควบคุมโมดูล CAN I/O PLC ใช้การประมวลผลผ่านโปรแกรม EAGLE 7.30 Light ซึ่งเป็นเครื่องมือที่ช่วยให้สามารถเชื่อมต่อและควบคุมการทำงานของอุปกรณ์ต่าง ๆ ในระบบ CAN BUS ได้อย่างมีประสิทธิภาพ โปรแกรมนี้ช่วยให้ผู้ใช้งานสามารถกำหนดและออกแบบกระบวนการทำงานของระบบได้ง่ายขึ้น โดยใช้วิธีการลากและจัดเรียงเครื่องมือหรือฟังก์ชันตามลำดับการทำงานที่ต้องการ การเขียนโปรแกรมจะเริ่มต้นจากการเลือกฟังก์ชันต่าง ๆ ที่จำเป็นสำหรับการประมวลผลคำสั่ง เช่น การรับส่งข้อมูลจากเซ็นเซอร์ การสั่งการอุปกรณ์ หรือการประมวลผลสัญญาณจากโมดูลต่าง ๆ ในระบบ CAN โดยผู้ใช้งานสามารถวางเครื่องมือในลำดับที่เหมาะสมเพื่อให้การทำงานของระบบเป็นไปตามที่ต้องการ ฟังก์ชันเหล่านี้จะช่วยให้การควบคุมอุปกรณ์ต่าง ๆ เป็นไปได้อย่างถูกต้อง



รูปที่ 3.10 หน้าต่างโปรแกรม EAGLE 7.30 Light

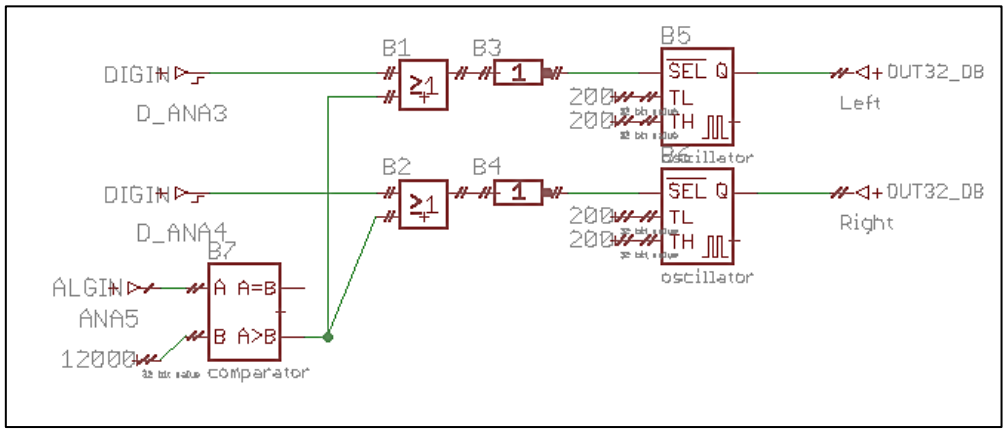
3.3.4 โปรแกรมควบคุมการทำงานของรถATV

โปรแกรมควบคุมการทำงานของรถ ATV โมดูลที่ 1 ได้ถูกตั้งโปรแกรมให้รับสัญญาณจากสวิตช์ควบคุมต่าง ๆ ภายในตัวรถ เช่น สวิตช์กุญแจ สวิตช์สตาร์ท สวิตช์แตร สวิตช์เดินหน้า-ถอยหลัง และสวิตช์โหมดการทำงาน จากนั้นจะมีการประมวลผลในส่วนของสัญญาณไฟเลี้ยวก่อนที่จะส่งข้อมูลไปยังโมดูลที่ 2 และโมดูลที่ 3 ผ่านระบบ CAN BUS โดยที่โมดูลที่ 2 และโมดูลที่ 3 จะรอรับข้อมูลดังกล่าวเพื่อนำไปสั่งการให้ไฟและอุปกรณ์ต่าง ๆ ทำงานสอดคล้องตามข้อมูลที่ได้รับจากโมดูลที่ 1 เพื่อให้ระบบทั้งหมดทำงานประสานกัน รูปแบบการเขียนโปรแกรมแสดง ดังรูปที่ 3.11



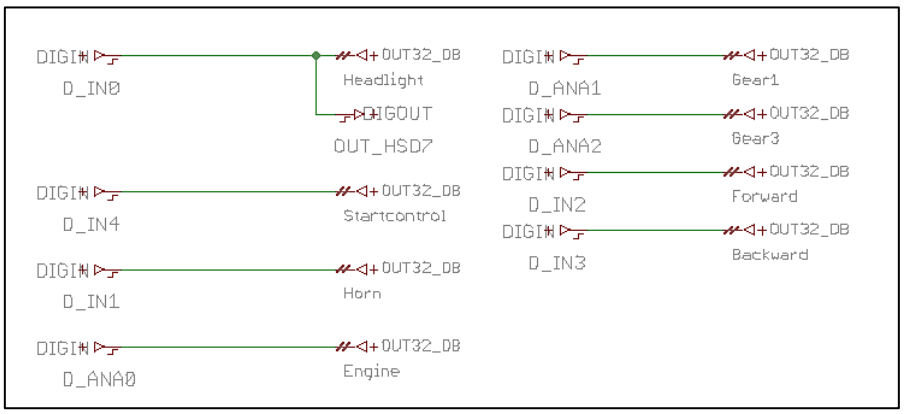
รูปที่ 3.11 โปรแกรมควบคุมการทำงานของรถATV โมดูลที่ 1

ในการประมวลผลของสัญญาณไฟเลี้ยว เมื่อมีการรับค่าสวิตช์ไฟเลี้ยวแล้ว โปรแกรมจะทำการประมวลผลให้เกิด สัญญาณกระพริบ และส่งคำสั่งไปยัง โมดูล CAN ที่ 2 และ โมดูล CAN ที่ 3 เพื่อหลีกเลี่ยงการเขียนโปรแกรมซ้ำซ้อนในส่วนของโปรแกรมควบคุมอื่น ๆ การส่งสัญญาณไฟกระพริบนี้ค่อนข้างซับซ้อน เนื่องจากความสามารถของขารับสัญญาณใน โมดูล CAN นั้นแตกต่างกันไป ตัวอย่างเช่น ขารับสัญญาณไฟเลี้ยวซ้าย D_ANA3 และ ขารับสัญญาณไฟเลี้ยวขวา D_ANA4 จะรับสัญญาณแบบ PULL_DOWN ซึ่งทำให้ค่าเริ่มต้นที่ออกมาคือ 0 ตลอดเวลาจนกว่าจะมีการเปิดสัญญาณไฟ การทำงานนี้ทำให้เครื่องมือ Oscillator เมื่อรับค่าเป็น 0 จะสร้างสัญญาณไฟกระพริบขึ้นมา ซึ่งไม่ตรงกับความต้องการ เพื่อแก้ไขปัญหานี้ จำเป็นต้องเพิ่มเครื่องมือ NOT (B3 และ B4) เพื่อทำการกลับค่าสัญญาณจากขา D_ANA3 และ D_ANA4 ก่อนที่จะส่งสัญญาณออกไปยังโมดูล CAN อื่น ๆ ซึ่งจะใช้ชื่อของตัวแปรที่ได้ตั้งไว้ก่อนหน้านี้ การใช้เครื่องมือ NOT นี้จะช่วยให้สามารถควบคุมและส่งสัญญาณไฟกระพริบไปยังโมดูล CAN ได้ตามต้องการ และในการสร้างสัญญาณไฟผ่าหมาก โมดูลจะรับค่าจาขารับสัญญาณ ANA5 ซึ่งขานี้จะรับสัญญาณแบบอนาล็อก โดยการเทียบค่าแรงดันไฟฟ้า 1 mV เท่ากับ 1 digit ดังนั้น เมื่อกดสัญญาณไฟผ่าหมากจะมีแรงดันที่ 12.3V หรือ 12300 digit การทำงานนี้จะใช้เครื่องมือ Comparator ช่วยในการส่งค่าเปิดไฟผ่าหมาก โดยเครื่องมือ Comparator จะเทียบค่าสัญญาณเมื่อเปิดแล้วว่า สัญญาณไฟมีค่ามากกว่าค่าที่กำหนดไว้ 12000 digit เมื่อถึงค่าดังกล่าวแล้ว ระบบจะใช้เครื่องมือ AND (B1 และ B2) เพื่อให้สัญญาณส่งค่าออกไปโดยในวงจรการสื่อสาร CAN BUS นี้จะใช้คำสั่ง Out32_DB เพื่อส่งค่าที่ได้ไปเก็บไว้ในตัวแปรย่อยที่ได้กำหนดไว้ก่อนหน้านี้ ซึ่งเป็นตัวแปรที่ใช้ในการควบคุมการทำงานของอุปกรณ์ต่าง ๆ หลังจากนั้น เมื่อโมดูลอื่น ๆ รับค่าแล้ว ก็จะสามารถทำงานพร้อมกันตามคำสั่งที่ส่งไปจากโมดูลที่มีการประมวลผล รูปแบบการเขียนโปรแกรมควบคุมไฟเลี้ยว แสดงดังรูปที่ 3.12



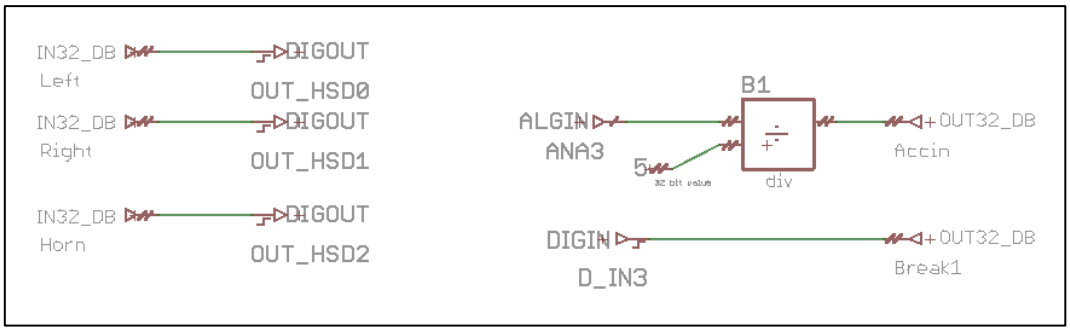
รูปที่ 3.12 โปรแกรมควบคุมการทำงานของไฟเลี้ยวในรถATV

ในอีกส่วนหนึ่งของการประมวลผลที่รับสัญญาณมาจากสวิตช์ ยังไม่ได้มีการประมวลผลทันที เนื่องจากจำเป็นต้องทำการประมวลผลร่วมกับสัญญาณที่ได้รับจากโมดูล CAN ที่ 2 เพื่อให้สามารถทำการควบคุมและประมวลผลข้อมูลได้อย่างถูกต้อง ดังนั้นในขั้นตอนนี้สัญญาณจากสวิตช์จะถูกส่งเข้าไปเก็บไว้ในตัวแปรย่อย ที่ได้มีการกำหนดไว้ก่อนหน้านี้เพื่อรอการประมวลผลในขั้นตอนถัดไป โดยการเก็บข้อมูลในตัวแปรนี้จะช่วยให้สามารถนำข้อมูลไปใช้งานร่วมกับสัญญาณจากโมดูล CAN ที่ 2 ในภายหลัง รูปแบบการเขียนโปรแกรมเพื่อรับค่าจากสวิตช์และเก็บข้อมูลไว้ในตัวแปรย่อยจะถูกแสดงใน รูปที่ 3.13



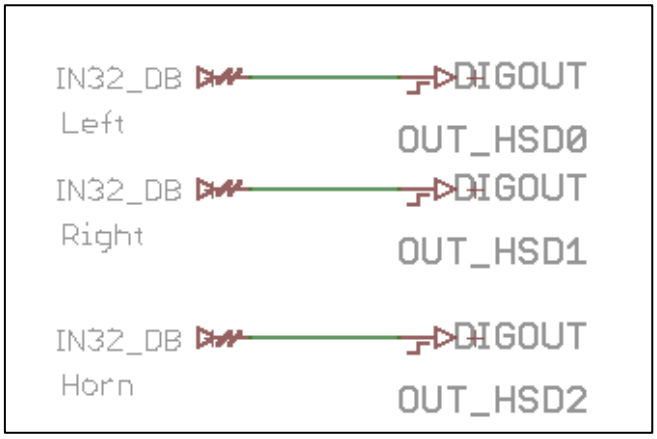
รูปที่ 3.13 โปรแกรมรับค่าสวิตช์และเก็บข้อมูลไว้ในตัวแปรย่อยในโมดูล CAN ที่ 1

โปรแกรมควบคุมการทำงานของรถ ATV สำหรับโมดูลที่ 2 ได้ถูกตั้งโปรแกรมให้รับสัญญาณ CAN BUS จากโมดูลที่ 1 เช่น สัญญาณไฟเลี้ยว และแตร โดยโมดูลที่ 2 จะทำการสั่งการให้หลอดไฟเลี้ยวซ้าย-ขวาและแตร ทำงานตามคำสั่งที่ได้รับจากโมดูลที่ 1 ทำให้การควบคุมอุปกรณ์ต่าง ๆ ในระบบไฟฟ้าภายในรถ ATV สามารถทำได้ตามต้องการ นอกจากนี้ โมดูลที่ 2 ยังมีหน้าที่ในการรับ สัญญาณจากเซ็นเซอร์เบรก และคันเร่ง เพื่อนำข้อมูลเหล่านี้ส่งต่อไปยัง โมดูลที่ 3 ซึ่งจะทำการควบคุมการทำงานของกล่องคอนโทรลเลอร์หลักในระบบรถ ATV โดยข้อมูลจากเซ็นเซอร์เบรกและคันเร่งจะถูกใช้ในการประมวลผลเพื่อควบคุมการทำงานของมอเตอร์และระบบต่าง ๆ ที่เกี่ยวข้อง ในภายหลังรูปแบบการเขียนโปรแกรมในการรับสัญญาณและการส่งต่อข้อมูลจากโมดูลที่ 2 แสดงในรูปที่ 3.14



รูปที่ 3.14 โปรแกรมควบคุมการทำงานของรถ ATV โมดูลที่ 2

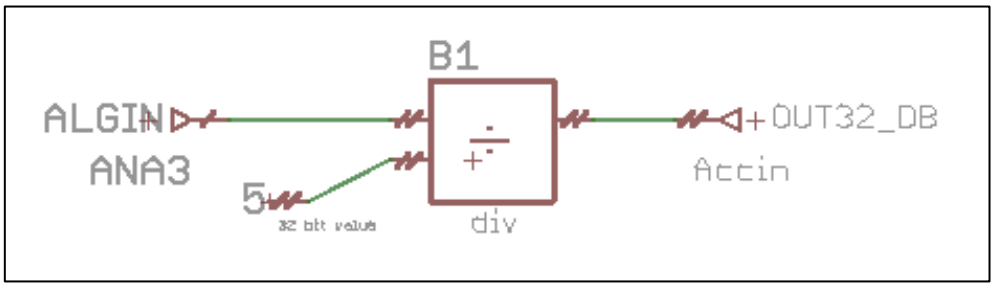
ในส่วนของการรับค่าสัญญาณนั้น โปรแกรมจะทำการเรียกตัวแปรย่อยที่ได้กำหนดไว้ก่อนหน้านี้ โดยใช้คำสั่ง IN32_DB เพื่อดึงข้อมูลที่เก็บไว้ในตัวแปรย่อยที่เกี่ยวข้องออกมา เมื่อข้อมูลถูกดึงมาแล้ว โปรแกรมจะทำการส่งค่าออกไปยังขาสัญญาณที่เชื่อมต่อกับอุปกรณ์ต่าง ๆ อยู่ โดยใช้คำสั่ง DIGOUT เพื่อสั่งให้อุปกรณ์ที่เชื่อมต่อกับขาสัญญาณ เช่น หลอดไฟ หรือ รีเลย์ นั้นทำงานตามคำสั่งที่ได้รับ การส่งออกสัญญาณนี้จะขึ้นอยู่กับการควบคุมจากโมดูล CAN ที่ 1 ซึ่งรับคำสั่งจากผู้ใช้งาน เพื่อทำให้หลอดไฟหรือรีเลย์ทำงานตามที่ต้องการในระบบการควบคุม รูปแบบการเขียนโปรแกรมดึงค่าจากตัวแปรมาใช้งาน แสดงในรูปที่ 3.15



รูปที่ 3.15 โปรแกรมดึงค่าจากตัวแปรมาใช้งานของรถ ATV โมดูลที่ 2

ในอีกส่วนหนึ่งของโปรแกรมการทำงานของ โมดูล CAN ที่ 2 คือการรับค่าสัญญาณจาก คันเร่ง ซึ่งคันเร่งจะส่งออกสัญญาณอนาล็อกที่มีค่าตั้งแต่ 0 ถึง 5V โมดูล CAN จะรับค่าได้ในรูปแบบอนาล็อก ซึ่งจะมีค่าตั้งแต่ 0 ถึง 5000 digit โดยที่ 1 mV เท่ากับ 1 digit นั่นหมายความว่า

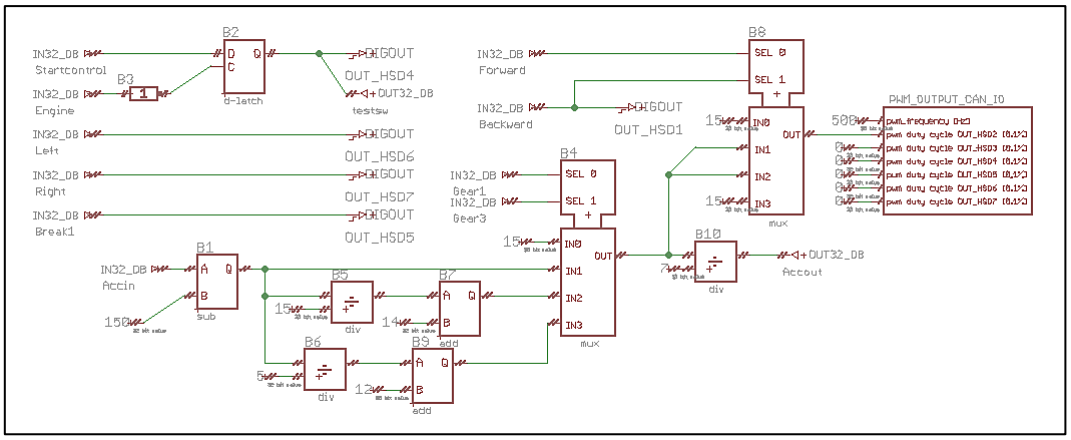
เมื่อคันเร่งส่งค่า 5V ก็จะถูกแปลงเป็น 5000 digit ซึ่งเท่ากับ 5V หรือค่าที่สูงสุดที่โมดูล CAN สามารถรับได้อย่างไรก็ตาม การส่งค่า 5000 digit นั้นจะต้องใช้พื้นที่ในการจัดเก็บข้อมูลมากถึง 13 bit ซึ่งมีข้อมูลที่เยอะเกินไป ทำให้การส่งข้อมูลในระบบ CAN BUS ช้าลง เนื่องจากต้องใช้เวลาในการส่งข้อมูลมากขึ้นเพื่อแก้ไขปัญหานี้ จึงทำการหารค่าที่อ่านได้จากคันเร่ง เพื่อลดขนาดข้อมูลที่ส่งไปยังโมดูลอื่น ๆ วิธีการนี้ช่วยให้ลดความยาวของข้อมูล และทำให้การส่งข้อมูลเป็นไปได้รวดเร็วขึ้น ตัวอย่างการเขียนโปรแกรมเพื่อรับค่าคันเร่ง และลดขนาดข้อมูลจะแสดงใน รูปที่ 3.15



รูปที่ 3.16 โปรแกรมรับค่าสัญญาณคันเร่งของโมดูล CAN ที่ 2

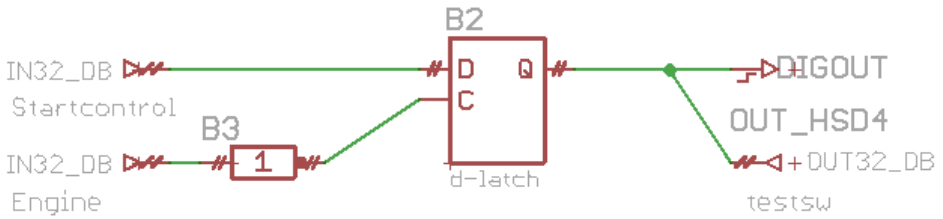
โปรแกรมควบคุมการทำงานของรถ ATV ถูกออกแบบให้โมดูลที่ 3 รับสัญญาณ CAN BUS จากโมดูลที่ 1 และโมดูลที่ 2 เพื่อนำมาประมวลผลและควบคุมระบบต่าง ๆ เช่น สวิตช์สตาร์ท และ สวิตช์เดินหน้า-ถอยหลัง โหมดการทำงาน โดยกำหนดเงื่อนไขเพื่อเพิ่มความปลอดภัยต่อการใช้งาน โดยผู้ใช้ต้องกดสวิตช์สตาร์ทเพื่อเปิดระบบมอเตอร์ก่อน จากนั้นต้องเลือกโหมดเดินหน้าหรือถอยหลังก่อนที่คันเร่งจะสามารถสั่งงานให้มอเตอร์หมุนได้ นอกจากนี้ มีการเขียนโปรแกรมให้มี 3 โหมดการขับขี่ เพื่อปรับการทำงานของคันเร่งให้เหมาะสมกับการใช้งาน ได้แก่ โหมด Eco, โหมด Normal และโหมด Sport โดยโหมด Eco จำกัดกำลังคันเร่งไว้ที่ 10% ของความสามารถสูงสุดของกล่องควบคุมมอเตอร์ แม้ว่าผู้ใช้จะเหยียบคันเร่งเต็มที่ ส่วนโหมด Normal จำกัดกำลังคันเร่งไว้ที่ 20% และโหมด Sportให้คันเร่งทำงานที่ 100%

นอกจากนี้ โมดูลที่ 3 ยังควบคุมคันเร่งให้ทำงานตามโหมดที่ได้รับคำสั่ง โดยโมดูล CAN I/O PLC ส่งสัญญาณ PWM ไปยังวงจร Low Pass Filter เพื่อแปลงสัญญาณเป็นอนาล็อคไปควบคุมกล่องควบคุมมอเตอร์ รวมถึงควบคุมระบบเบรกและไฟเลี้ยวซ้าย-ขวา เพื่อให้การขับขี่ของรถ ATV เป็นไปตามการควบคุมที่ต้องการ รูปแบบการเขียนโปรแกรมในการรับสัญญาณจากโมดูลที่ 2 แสดงในรูปที่ 3.17



รูปที่ 3.17 โปรแกรมควบคุมการทำงานของรถATV โมดูลที่ 3

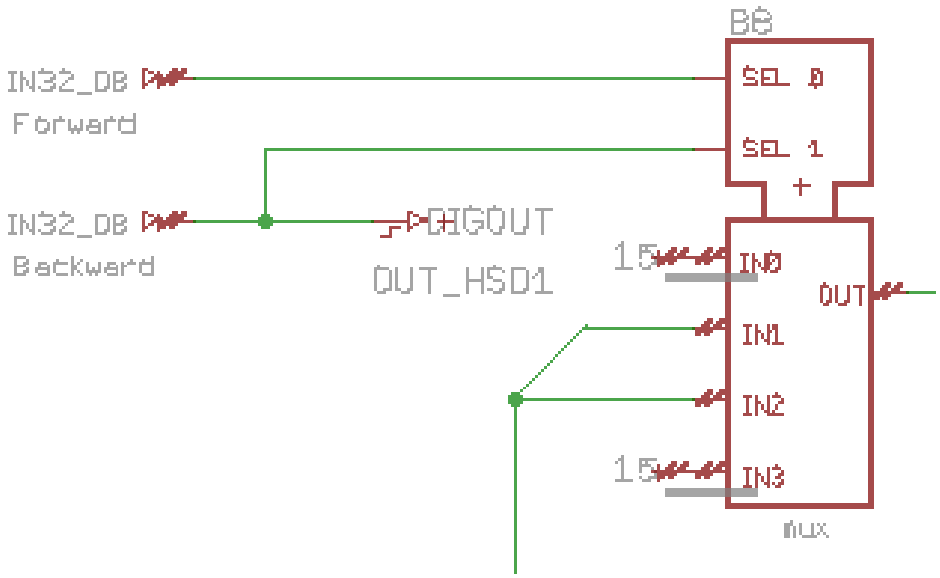
ในการเขียนโปรแกรมเงื่อนไขด้านความปลอดภัยของการใช้งานนั้น ได้ถูกกำหนดเงื่อนไขการใช้งานไว้ว่า เมื่อเปิดสวิตช์กุญแจแล้วรถยังไม่สามารถขับเคลื่อนได้จนกว่าจะกด ปุ่ม Engine Start เพื่อป้องกันการหลงลืมของผู้ใช้งาน และเพื่อความปลอดภัยในการใช้งานรถ ATV โดยใช้เครื่องมือ d-latch เพื่อคงค่าสถานะของ ปุ่ม Engine Start ซึ่งจะช่วยให้สถานะการกดปุ่ม Engine Start คงอยู่แม้ผู้ใช้งานจะปล่อยมือจากปุ่ม เมื่อกดปุ่ม Engine Start แล้ว รถจะสามารถขับเคลื่อนได้ตามปกติ แต่หากผู้ใช้งาน ปิดกุญแจ แล้วเปิดใหม่ รถจะไม่สามารถขับเคลื่อนได้จนกว่าจะกด ปุ่ม Engine Start อีกครั้งเพื่อเริ่มการใช้งานใหม่ ซึ่งเป็นการเพิ่มความปลอดภัยในการใช้งานรถอย่างมีประสิทธิภาพรูปแบบการเขียนโปรแกรมด้านความปลอดภัยของสวิตช์ Engine Start แสดงในรูปที่ 3.18



รูปที่ 3.18 โปรแกรมเงื่อนไขด้านความปลอดภัยของการใช้งาน Engine Start

อีกหนึ่งเงื่อนไขด้านความปลอดภัยของการใช้งานคือ สวิตช์สั่งงานเดินหน้า, ถอยหลัง และเกียร์ว่าง ซึ่งมีการเขียนเงื่อนไขในส่วนนี้เพิ่มเข้ามาเพื่อเพิ่มความปลอดภัยในการใช้งาน เนื่องจากเดิมกล่องควบคุมมอเตอร์นั้นไม่มีการกำหนดเกียร์ว่าง เมื่อเหยียบคันเร่งรถก็จะเคลื่อนที่ทันที

ซึ่งอาจเกิดอันตรายได้ในบางกรณี เพื่อป้องกันปัญหานี้ จึงมีการเขียนโปรแกรมให้ตัดการทำงานของคันเร่ง ในกรณีที่ไม่ได้อยู่ในเกียร์ที่เลือกไว้ เช่น เกียร์เดินหน้าหรือถอยหลัง โดยเมื่อเหยียบคันเร่งแล้วรถจะไม่สามารถเคลื่อนที่ได้ ถ้าไม่ได้อยู่ในตำแหน่งเกียร์ที่ถูกต้อง โปรแกรมนี้ใช้เครื่องมือ MUX ซึ่งจะช่วยเลือกค่าของคันเร่งตามคำสั่งที่ได้รับจากการสั่งงานเดินหน้าหรือถอยหลัง ก่อนที่จะส่งค่าดังกล่าวไปควบคุมการทำงานของกล่องควบคุมมอเตอร์ การใช้เครื่องมือ MUX จะช่วยให้โปรแกรมสามารถเลือกค่าได้อย่างถูกต้องตามคำสั่งและทำให้การทำงานของมอเตอร์เป็นไปตามที่ต้องการ รูปแบบการเขียนโปรแกรมด้านความปลอดภัยของ สวิตช์เดินหน้า, ถอยหลัง และเกียร์ว่าง จะแสดงในรูปที่ 3.19



รูปที่ 3.19 โปรแกรมด้านความปลอดภัยของ สวิตช์เดินหน้า, ถอยหลัง และเกียร์ว่าง

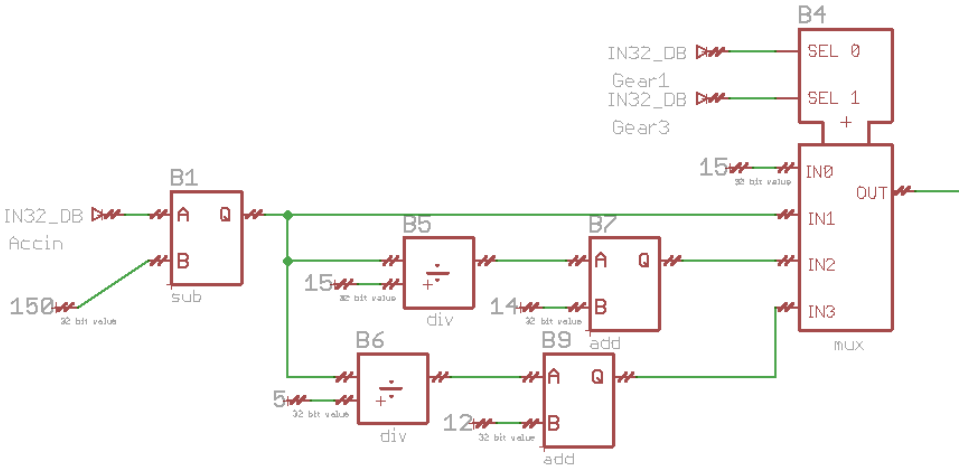
ในส่วนของโหมดการทำงาน โปรแกรมได้รับการออกแบบให้สามารถจัดการการทำงานของ คันเร่งให้ทำงานแตกต่างกันในแต่ละโหมด โดยมีการกำหนดโหมดหลัก ๆ คือ Eco, Normal, และ Sport โปรแกรมจะเริ่มต้นจากการรับค่าคันเร่ง เข้ามาก่อนที่จะทำการประมวลผล การประมวลผลค่าคันเร่งนั้นจะใช้เครื่องมือ MUX เพื่อเลือกและคำนวณค่าของคันเร่งตามโหมดที่เลือกไว้

โหมด ECO: ค่าคันเร่งที่รับเข้ามาจะถูกหารด้วย 15 ผลลัพธ์ที่ได้จะเป็นเพียง 10% ของค่าคันเร่งที่ได้รับ ซึ่งจะช่วยลดการตอบสนองของคันเร่งเพื่อประหยัดพลังงานและเพิ่มความประหยัดในโหมดนี้

โหมด Normal: ค่าคันเร่งจะถูก หารด้วย 5 ผลลัพธ์ที่ได้จะออกมาเพียง 20% ของค่าคันเร่งที่ได้รับ ซึ่งจะทำให้การตอบสนองของคันเร่งเป็นปกติ สามารถขับขี่และควบคุมรถได้ง่าย

โหมด Sport: ในโหมดนี้ ค่าคันเร่งจะไม่ถูกหารแต่จะส่งออกไปโดยตรง ซึ่งทำให้การตอบสนองของคันเร่งเต็มที่และรวดเร็วที่สุด

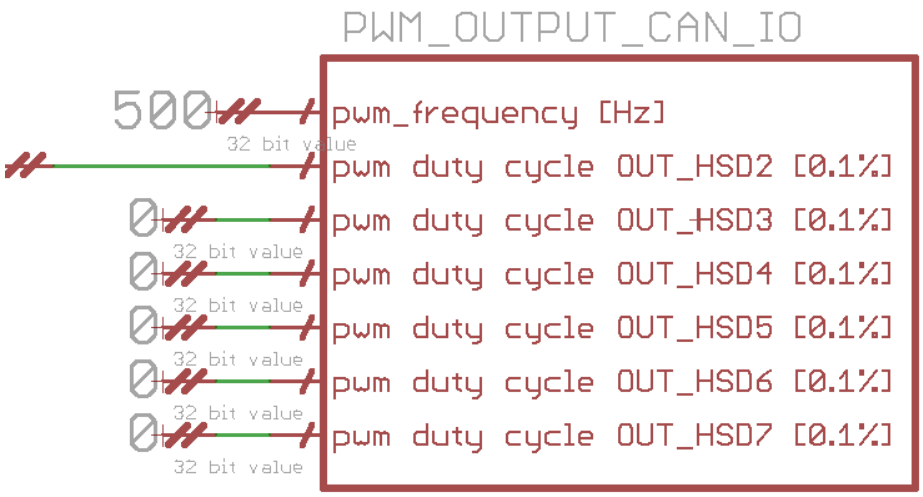
การประมวลผลในแต่ละโหมดจะช่วยให้การขับขี่มีความหลากหลายและเหมาะสมกับสภาพการขับขี่หรือความต้องการของผู้ขับขี่ รูปแบบการเขียนโปรแกรมเลือกโหมดการใช้งานแสดงดังในที่ 3.20



รูปที่ 3.20 โปรแกรมเลือกโหมดการใช้งาน

การส่งค่าคันเร่งเพื่อควบคุมการทำงานของกล่องควบคุมมอเตอร์นั้น โปรแกรมจะทำการนำค่าคันเร่งที่ได้รับการประมวลผลจากการเลือกโหมด (Eco, Normal, Sport) แล้ว เพื่อสร้างเป็น สัญญาณ PWM โดยใช้เครื่องมือ PWM_OUTPUT_CAN_IO ซึ่งเครื่องมือนี้จะช่วยในการสร้างสัญญาณ PWM ที่ต้องการ การสร้างสัญญาณ PWM นี้จะต้องมีการกำหนดให้สัญญาณออกที่ขาส่งสัญญาณที่กำหนดไว้ในระบบ และต้องตั้งค่าความถี่ของสัญญาณ PWM โดยความถี่ที่เหมาะสมสำหรับการใช้งานนี้คือ 500 Hz ซึ่งเหมาะสมสำหรับการควบคุมความเร็วของมอเตอร์ในระบบ อย่างไรก็ตาม เนื่องจากกล่องควบคุมมอเตอร์ไม่สามารถใช้สัญญาณ PWM ในการควบคุมการทำงานได้โดยตรง จึงจำเป็นต้องมีการใช้วงจร Low Pass Filter เพื่อแปลงสัญญาณ PWM ที่ได้รับมาให้กลายเป็นสัญญาณอนาล็อกที่สามารถใช้งานได้กับกล่องควบคุมมอเตอร์ การทำงานของ Low Pass Filter จะช่วยกรองสัญญาณ PWM และให้ผลลัพธ์เป็นสัญญาณอนาล็อกที่คงที่ ซึ่งสามารถส่งไปควบคุมการทำงานของ

มอเตอร์ได้อย่างแม่นยำและเหมาะสม รูปแบบการเขียนโปรแกรมสร้างสัญญาณPWM แสดงดังในรูปที่ 3.21



รูปที่ 3.21 โปรแกรมสร้างสัญญาณPWM

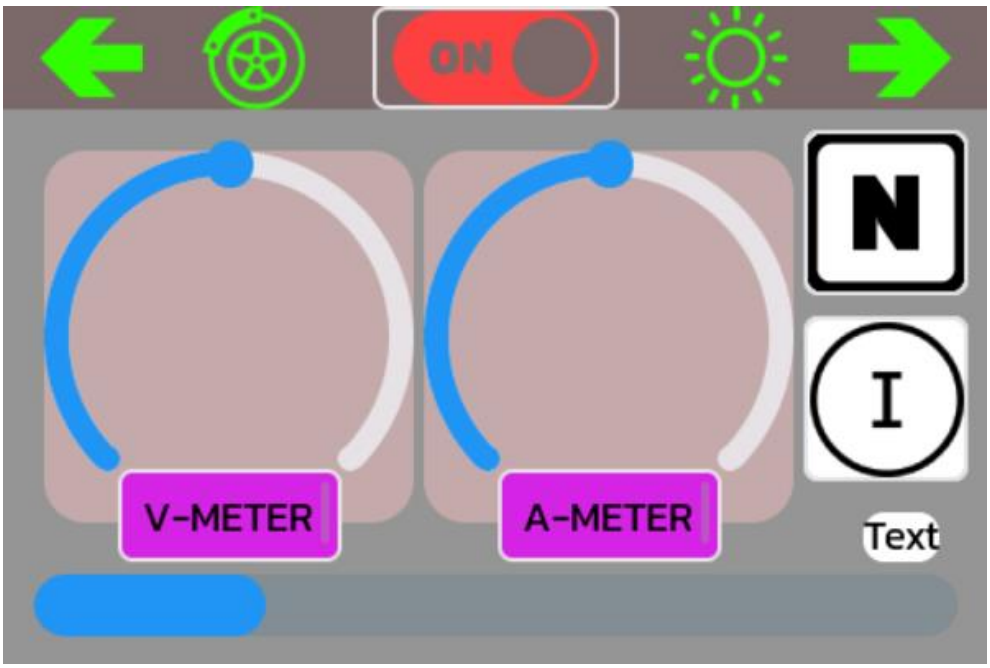
3.4 การกรองสัญญาณ PWM เป็นสัญญาณอนาล็อก

เนื่องจากโมดูล CAN I/O PLC ไม่สามารถส่งสัญญาณอนาล็อก ออกไปควบคุมกล่องควบคุมมอเตอร์ได้ แต่สามารถส่งสัญญาณ Pulse Width Modulation (PWM) ได้ ดังนั้น จึงจำเป็นต้องแปลงสัญญาณ PWM ให้เป็นสัญญาณอนาล็อก เพื่อให้กล่องควบคุมมอเตอร์สามารถรับและประมวลผลได้อย่างถูกต้องการแปลงสัญญาณนี้ใช้ หลักการของ Low Pass Filter (LPF) ซึ่งทำหน้าที่กรองความถี่สูงของสัญญาณ PWM ออกไป ทำให้สัญญาณที่ได้เป็นแรงดันเฉลี่ยที่สามารถใช้แทนสัญญาณอนาล็อก ได้ ในงานวิจัยนี้ได้คำนวนวนจากตัวแปลงที่มีอยู่ คือ สัญญาณ PWM แรงดัน 5V ความถี่ที่ 500 Hz จะใช้ความต้านที่ 4,700 โอห์ม และคาปาซิเตอร์ 10 ไมโครฟารัต การต่อวงจรกรองความถี่ Low Pass Filter แสดงในรูปที่ 3.22 วิธีนี้ช่วยให้กล่องควบคุมมอเตอร์สามารถรับค่าและตอบสนองต่อการควบคุมจากโมดูล CAN I/O PLC ได้อย่างถูกต้องและราบรื่น

รูปที่ 3.22 การต่อวงจรกรองความถี่ Low Pass Filter

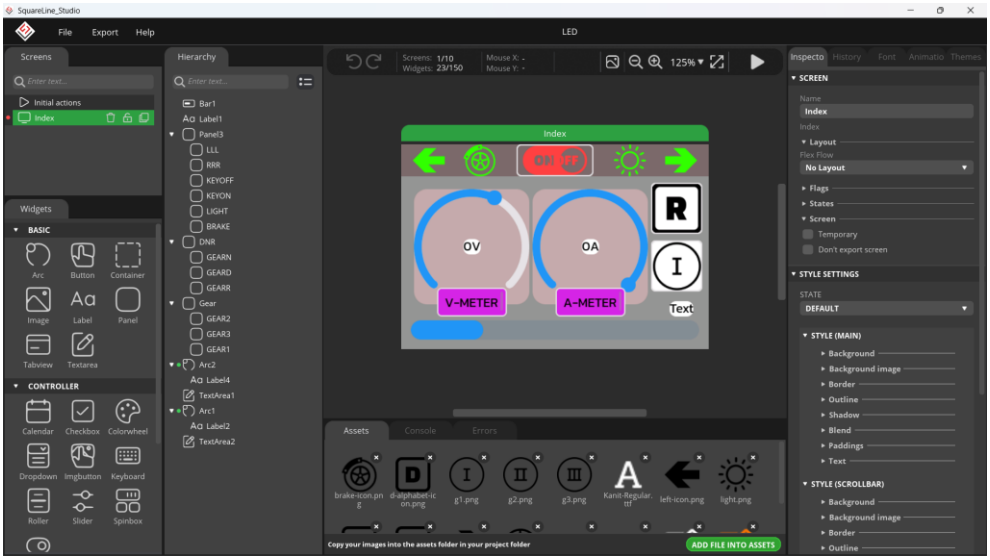
3.5 จอแสดงผลของระบบ CAN BUS

จอแสดงผลของระบบ CAN BUS ถูกเลือกใช้จอ ATD3.5-S3 ร่วมกับ CAN BUS Shield โดยเริ่มจากการออกแบบอินเทอร์เฟซผ่านโปรแกรม Square Line Studio ซึ่งเป็นเครื่องมือที่ช่วยในการสร้างอินเทอร์เฟซและสามารถออกแบบได้ตามต้องการ ในหน้าจอนี้ถูกออกแบบให้แสดงสถานะการทำงานต่าง ๆ ภายในตัวรถ ข้อมูลทั้งหมดจะถูกส่งและแสดงผลแบบเรียลไทม์บนจอแสดงผล ซึ่งการออกแบบหน้าจอแสดงในรูปที่ 3.23 หลังจากออกแบบหน้าจอเสร็จแล้ว การเขียนโปรแกรมควบคุมดำเนินการผ่าน Visual Studio Code พร้อมติดตั้ง Platform IO Extension โดยใช้ไลบรารี LVGL เพื่อสร้างหน้าจอตามที่ออกแบบใน Square Line Studio และใช้ไลบรารี twai เพื่ออ่านค่าจาก CAN BUS และนำข้อมูลมาแสดงผลบนหน้าจอ

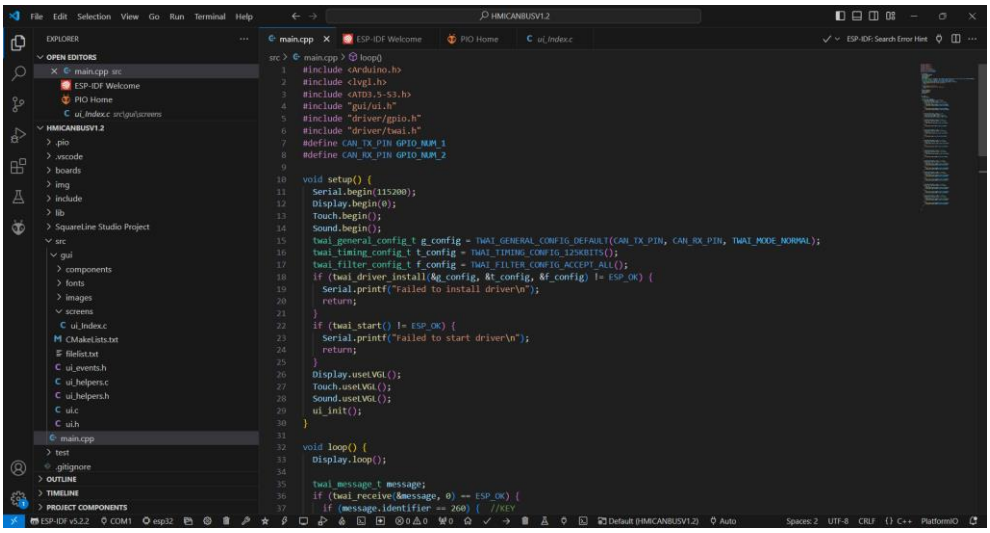


รูปที่ 3.23 ตัวอย่างจอแสดงผลของระบบ CAN BUS

ในการสร้างหน้าจอแสดงผลนั้น ขั้นตอนแรกคือการออกแบบหน้าจอแสดงผล ในขั้นตอนนี้จะทำการออกแบบหน้าจอโดยใช้โปรแกรม Square Line Studio ซึ่งเป็นเครื่องมือที่ช่วยให้ผู้ใช้งานสามารถออกแบบ อินเทอร์เฟซ ได้อย่างสะดวก โดยเริ่มจากการสร้าง ไอคอน และ องค์ประกอบต่าง ๆ ที่จะใช้แสดงผลบนหน้าจอให้ครบถ้วนตามที่ต้องการ เช่น การแสดงสถานะของ คันเร่ง เบรก ไฟเลี้ยว โหมดการทำงาน หรือสถานะของมอเตอร์ รวมไปถึงกราฟิกต่าง ๆ ที่จะช่วยให้ผู้ใช้งานเข้าใจข้อมูลได้ง่าย โปรแกรมการออกแบบหน้าจอแสดงในรูปที่ 3.24



รูปที่ 3.23 โปรแกรม Square Line Studio



รูปที่ 3.13 โปรแกรมคำสั่งจอรับข้อมูลจาก CAN BUS

3.6 การวัดกำลังฟ้า โดยใช้ V-BOX ร่วมกับ Power Meter

เนื่องจาก BMS ของแบตเตอรี่ไม่สามารถสื่อสารผ่าน CAN BUS ได้โดยตรง จึงใช้ DC Power Meter ในการอ่านค่าพลังงานที่ใช้จากแบตเตอรี่ จากนั้นใช้ V-BOX ที่เชื่อมต่อกับ DC Power Meter เพื่อส่งค่าพลังงานเข้าสู่ V-Net ซึ่งทำหน้าที่เก็บข้อมูลการใช้กำลังไฟฟ้าแบบเรียลไทม์ แสดงในรูปที่ 3.14 ข้อมูลที่ได้จาก V-Net ถูกนำมาใช้ในการวิเคราะห์ผลการทดลอง เพื่อประเมินแนวโน้มการใช้

กำลังไฟฟ้าของรถ ATV ในแต่ละโหมดการทำงาน ซึ่งช่วยให้สามารถวิเคราะห์ประสิทธิภาพและปรับปรุงระบบให้เหมาะสมกับการใช้งานได้

รูปที่ 3.14 ตัวอย่างข้อมูลการวัดกำลังไฟฟ้าที่ใช้ในรถ ATV

3.3 ขั้นตอนการทดลอง

ในการทดสอบระบบ CAN BUS สำหรับควบคุมการทำงานของรถ ATV ได้ดำเนินการตามลำดับขั้นตอนดังต่อไปนี้ โดยเริ่มจากการทดสอบระบบ CAN BUS ซึ่งทำการตรวจสอบสัญญาณการสื่อสารระหว่างโมดูลทั้งสามโมดูล โดย โมดูลที่ 1 ทำหน้าที่รับข้อมูลจากสวิตช์ควบคุมต่าง ๆ เช่น สวิตช์กุญแจและไฟเลี้ยว จากนั้นส่งข้อมูลไปยังโมดูลอื่น โมดูลที่ 2 ทำหน้าที่รับสัญญาณจากคันเร่งและเบรก เพื่อนำไปแสดงผลและส่งต่อไปยัง โมดูลที่ 3 ซึ่งทำหน้าที่รับค่าการทำงานของเกียร์และโหมดการทำงาน พร้อมทั้งแสดงสถานะต่าง ๆ เช่น เกียร์เดินหน้า ถอยหลัง หรือเกียร์ว่าง นอกจากนี้ยังได้ทดสอบการแสดงผลแบบเรียลไทม์ของค่าต่าง ๆ บนหน้าจอ ATD3.5-S และทำการวิเคราะห์ค่าที่แสดงเพื่อพิจารณาความถูกต้องของข้อมูลที่รับ-ส่งผ่านระบบ CAN BUS หลังจากทดสอบระบบ CAN BUS แล้ว จึงดำเนินการทดสอบการควบคุมมอเตอร์ไฟฟ้า โดยเริ่มจากการส่งสัญญาณจากโมดูล CAN BUS ไปยังกล่องควบคุมมอเตอร์ไฟฟ้า จากนั้นทำการทดสอบการควบคุมคันเร่งในโหมดการทำงานต่าง ๆ ได้แก่ Eco, Normal และ Sport ซึ่งในขั้นตอนนี้ได้มีการใช้งานวงจร Low Pass Filter เพื่อแปลงสัญญาณ PWM ให้เป็นสัญญาณอนาล็อก เพื่อให้การควบคุมมอเตอร์ไฟฟ้ามีความแม่นยำยิ่งขึ้น

สุดท้ายได้ทำการบันทึกค่าความเร็วสูงสุดและกำลังไฟฟ้าที่ใช้ในแต่ละโหมด เพื่อเปรียบเทียบการทำงานของระบบและวิเคราะห์ผลการทดสอบ