****

**รายงานวิชา Pre-Project รหัสวิชา 01216747**

**จัดทำโดย**

**นาย ศิวคุป ควรพันธ์ รหัสนักศึกษา 60010987**

**นางสาว สุชานรี กองพลพรหม รหัสนักศึกษา 60011083**

**นาย สุวิจักขณ์ ชัยเจริญลักษณ์ รหัสนักศึกษา 60011114**

**เสนอ**

**ผศ.ดร.อุดม จันทร์จรัสสุข**

**ภาคเรียนที่ 2 ปีการศึกษา 2562**

**สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง**

**กิตติกรรมประกาศ**

โครงงานนี้สำเร็จลุล่วงด้วยความกรุณาจาก ผศ.ดร.อุดม จันทร์จรัสสุข อาจารย์ที่ปรึกษาโครงงานที่ได้ให้ความรู้ คำแนะนำ การเสนอแนะ ตลอดจนการแก้ไขข้อบกพร่องต่างๆ มาโดยตลอดจนโครงงานเล่มนี้เสร็จสมบูรณ์ได้ด้วยดี ผู้จัดทำจึงขอกราบขอพระคุณเป็นอย่างสูง

ท้ายสุดนี้คณะผู้จัดทำหวังเป็นอย่างยิ่งว่าโครงงานเล่มนี้จะเป็นประโยชน์ต่อผู้สนใจในเรื่องการทำหุ่นยนต์ การเขียนโปรแกรมภาษา C / C++ / C# หรือเรื่องการต่อวงจรต่างๆ

คณะผู้จัดทำ

**บทคัดย่อ**

ในปัจจุบันอุตสาหกรรมการผลิตยานยนต์และเครื่องมืออิเล็กทรอนิกส์เป็นหนึ่งในอุตสาหกรรมที่สำคัญ นั่นทำให้โรงงานอุตสาหกรรมต่างๆ ต้องการคนงานจำนวนมาก ทว่าด้วยปัญหาด้านการขาดแคลนแรงงาน และงานบางประเภทมีการทำงานซ้ำซ้อนกัน หรืองานเสี่ยงและอันตราย นวัตกรรมหุ่นยนต์ที่สามารถทำงานอย่างต่อเนื่อง มีความแม่นยำ และรวดเร็วกว่ามนุษย์ อีกทั้งยังอดทนต่อสภาพแวดล้อม จึงถูกนำมาใช้แทนที่ และให้มนุษย์เป็นผู้ควบคุมแทน ซึ่งในปัจจุบันเม็ดเงินลงทุนในอุตสาหกรรมที่เกี่ยวกับหุ่นยนต์ยังเพิ่มสูงขึ้น ทั้งการสั่งซื้อหุ่นยนต์และการจัดจ้างบุคลากร สอดคล้องกับความต้องการในตลาดที่เพิ่มกำลังการผลิต รวมถึงบำรุงรักษาหุ่นยนต์ให้พร้อมใช้งานอยู่เสมอ ยิ่งนวัตกรรมหุ่นยนต์ใหม่ๆ ออกมามากเท่าไหร่ ก็เป็นการเพิ่มความหลากหลายให้การทำงานและเปิดโอกาสให้ผู้ประกอบการผลิตชิ้นงานได้หลากหลายและรวดเร็วขึ้นเท่านั้น

โครงงานเล่มนี้จึงจัดทำขึ้นโดยมีจุดประสงค์เพื่อศึกษาหาความรู้เกี่ยวกับการเขียนโปรแกรม การออกแบบ หรือการใช้อุปกรณ์ต่างๆเพื่อใช้ในการทำหุ่นยนต์ ที่จะเป็นความรู้ให้ผู้จัดทำที่สามารถนำไปต่อยอดได้ในอนาคต และเป็นความรู้แก่ผู้ที่สนใจศึกษาในด้านนี้

**สารบัญ**

**1.บทนำ**

1.1 ความเป็นมาและความสำคัญของปัญหา 1

1.2 ปัญหา 1

1.3 วัตถุประสงค์ 2

1.4 ขอบเขตการทำโครงงาน 2

**2.ทฤษฎีที่เกี่ยวข้อง**

2.1 ภาษา C / C++ 3

2.2 ไมโครคอนโทรลเลอร์ 5

2.3 โมดูลเซ็นเซอร์แสงสำหรับตรวจจับวัตถุกีดขวาง 8

2.4 เซ็นเซอร์วัดระยะทาง 10

2.5 DC Motor Speed Control 11

2.6 TCRT5000 Infrared Reflective sensor 13

2.7 วงจร DC/DC Step-up (แรงดันปรับค่าได้) 15

2.8 ภาษา C# 16

**3.การออกแบบและการจัดทำโครงงาน**

3.1 กลยุทธ์ฝ่ายรุก 17

3.2 กลยุทธ์ฝ่ายรับ 18

3.3 การออกแบบการทำงานของวงจร 19

3.4 การออกแบบตัวหุ่นยนต์ 24

3.5 Programming Code 25

**4.เอกสารอ้างอิง**  34

**บทที่ 1**

**บทนำ**

**1.1) ความเป็นมาและความสำคัญของปัญหา**

หุ่นยนต์ คือเครื่องยนต์ชนิดหนึ่งที่มีลักษณะโครงสร้างและรูปร่างแตกต่างกันไปตามวัตถุประสงค์ หุ่นยนต์ในแต่ละประเภทจะมีหน้าที่การทำงานในด้านต่าง ๆ ตามการควบคุมโดยตรงของมนุษย์ หรืออาจมีการตั้งค่าให้หุ่นยนต์สามารถตัดสินใจได้เองในระดับใดระดับหนึ่ง การควบคุมระบบต่าง ๆ ในการสั่งงานระหว่างหุ่นยนต์และมนุษย์ สามารถทำได้โดยทางอ้อมและอัตโนมัติ โดยทั่วไปหุ่นยนต์ถูกสร้างขึ้นเพื่อสำหรับงานที่มีความยากลำบากหรืออันตรายเช่น งานสำรวจในพื้นที่บริเวณแคบ งานสำรวจในบริเวณที่เกิดภัยพิบัติ หรืองานสำรวจบนผิวของดวงจันทร์หรือดาวเคราะห์ที่ต่างๆ ปัจจุบันเทคโนโลยีของหุ่นยนต์เจริญก้าวหน้าอย่างรวดเร็ว หุ่นยนต์เริ่มเข้ามามีบทบาทกับชีวิตของมนุษย์ในหลากหลายด้าน เช่น ด้านอุตสาหกรรมการผลิต ด้านการแพทย์ ด้านงานสำรวจทั้งในโลกเราและงานสำรวจในอวกาศ หรือด้านการบันเทิง เช่นหุ่นยนต์ที่ถูกสร้างขึ้นเพื่อเป็นเครื่องเล่นของมนุษย์ ปัจจุบันนี้ได้มีการพัฒนาให้หุ่นยนต์นั้นมีลักษณะที่คล้ายมนุษย์มากขึ้นเพื่อผลทางจิตวิทยาในการอาศัยอยู่ร่วมกันกับมนุษย์ในชีวิตประจำวัน

หุ่นยนต์เริ่มมีบทบาททางด้านเทคโนโลยีอุตสาหกรรมในขณะที่งานด้านอุตสาหกรรม มีความต้องการด้านแรงงานเป็นอย่างมาก การจ้างแรงงานจำนวนมากเพื่อใช้ในงานอุตสาหกรรม ทำให้ต้นทุนการผลิตของแต่ละโรงงานอุตสาหกรรม เพิ่มจำนวนสูงขึ้น และงานอุตสาหกรรมบางงานไม่สามารถที่จะใช้แรงงงานเข้าไปทำได้ ซึ่งบางงานนั้นอันตรายและมีความเสี่ยงเป็นอย่างมาก หรือเป็นงานที่ต้องการความรวดเร็วและแม่นยำในการผลิตรวมทั้งเป็นการประหยัดระยะเวลา ทำให้หุ่นยนต์กลายเป็นทางออกของงานด้านอุตสาหกรรม [1]

**1.2) ปัญหา**

การแข่งขันหุ่นยนต์นั้นจะประกอบไปด้วยหุ่นยนต์ฝั่งละ 7 ตัวทั้ง 2 ฝ่ายสลับกันรุกและรับในการแข่งขันแต่ละรอบ โดยฝ่ายรุกจะต้องเข้าไปหาฝั่งตรงข้าม จนผ่านเส้นสีแดง แล้วกลับมาที่เดิมแบบปลอดภัยคือผ่านเส้นสีเหลืองเริ่มต้น โดยที่ไม่โดนจับได้ก็จะเป็นฝ่ายที่ชนะไปในรอบนั้น หุ่นยนต์ที่ถูกจับจะถูกนำออกจากการแข่งขัน ส่วนฝ่ายรับจะสามารถวิ่งสกัดอีกฝ่ายในพื้นที่ป้องกันเท่านั้น ถ้าวิ่งออกนอกพื้นที่จะถูกนำออกจากการแข่งขันในรอบนั้นด้วยเช่นกัน ถ้าไม่มีหุ่นยนต์ตัวไหนผ่านไปได้ฝ่ายรับก็จะชนะการแข่งขันแต่ละรอบจะจบเมื่อทีมรุกมาสารถผ่านเส้นไปได้ หรือทีมใดทีมหนึ่งไม่เหลือผู้เล่น

**1.3) วัตถุประสงค์**

1.) ฝึกฝนการทำหุ่นยนต์ของจริง เพื่อเป็นความรู้ไปใช้ต่อในอนาคต

2.) ศึกษาการใช้งานอุปกรณ์เครื่องมือต่างๆ เช่น อินฟราเรด , บอร์ด Arduino , เซนเซอร์ตรวจจับสิ่งกีดขวาง เป็นต้น

3.) ฝึกการเขียนโปรแกรมไมโครคอนโทรลเลอร์

4.) ได้รับความสนุกสนานเวลาแข่งกับฝ่ายตรงข้าม

**1.4 ) ขอบเขตการทำโครงงาน**

ทำการออกแบบตัวหุ่นยนต์ที่ใช้ในการแข่งขัน และวงจรสำหรับการควบคุมหุ่นยนต์ โดยต้องมีกลยุทธ์ให้หุ่นยนต์ 2 กลยุทธ์คือ กลยุทธ์รุก และ กลยุทธ์รับ ที่สามารถควบคุมการทำงานได้โดยไมโครคอนโทรลเลอร์

**บทที่ 2**

**ทฤษฎีที่เกี่ยวข้อง**

โครงงานนี้ได้จัดทำขึ้นเพื่อออกแบบวงจร และควบคุมหุ่นยนต์เพื่อแข่งขันกัน โดยในการทำโครงงานนี้มีทฤษฎีและหลักการที่เกี่ยวข้องดังต่อไปนี้

**2.1) ภาษา C / C++**

ภาษา Arduino (หรือ ภาษา C/C++) ใช้สำหรับเขียนโปรแกรมควบคุมไมโครคอนโทรลเลอร์

Arduino IDE เป็นเครื่องมือสำหรับเขียนโปรแกรมด้วยภาษา Arduino คอมไพล์โปรแกรม (Compile) และอัปโหลดโปรแกรมลงบอร์ด (Upload)

1) ฟังก์ชันหลัก (Structure)

เป็นฟังก์ชันหลักในการเขียนโปรแกรม จำเป็นต้องมีในทุกโปรแกรม

Setup () คือ ฟังก์ชันใช้ในการประกาศค่าเริ่มต้น ตำแหน่งพอร์ตที่ใช้งาน รวมถึงฟังก์ชันที่อยู่ไลบารีที่ใช้งาน เป็นฟังก์ชันที่ทำงานเพียงครั้งเดียว จะทำงานทุกครั้งที่มีการรีเซต หรือรีบูตเครื่องใหม่เท่านั้น

Loop () คือ ฟังก์ชันใช้ในการเขียนโค้ดโปรแกรมการทำงานของ Arduino เป็นฟังก์ชันการวนลูปไปเรื่อยๆ

2) ชุดคำสั่งในการควบคุม (Control Structures)

เป็นชุดคำสั่งในการใช้ในการตัดสินใจหาทางออก เพื่อใช้ในการทำงาน

If คือ คำสั่งในการตัดสินใจแบบตัวเลือกเดียว โดยใช้งานร่วมกับ and, or not, ==, !=, <, > เพื่อใช้ในการตัดสินใจในการหาคำตอบ

If...else คือ คำสั่งในการตัดสินใจแบบหลายตัวเลือก โดยใช้งานร่วมกับ and, or not, ==, !=, <, >เพื่อใช้ในการตัดสินใจในการหาคำตอบ

for คือ คำสั่งกำหนดเงื่อนไขเป็นจำนวนครั้งที่จะทำตามชุดคำสั่งต่าง ๆ ภายในลูป เหมาะที่จะใช้กับงานประเภทที่ไม่มีการเปลี่ยนแปลง

switch case คือ คำสั่งที่ใช้ในการจัดการเงื่อนไขหลายเงื่อนไข โดยเฉพาะการใช้งานโครงสร้าง การจำแนกเงื่อนไขไม่จำเป็นต้องอาศัยเฉพาะตัวแปรที่เก็บค่าจำนวนเต็มเท่านั้น ข้อมูลแบบอื่นก็สามารถใช้ได้เช่นกัน

while คือ คำสั่งเงื่อนไขที่จะทำการตรวจสอบว่าเป็นจริงหรือเท็จ ชุดคำสั่งก็คือ ส่วนที่ทำงานซ้ำ ๆ โดยจะต้องมีคำสั่งที่จะทำให้เงื่อนไขเป็นเท็จด้วย

#define คือ คำสั่งกำหนดค่านิพจน์ต่าง ๆ ให้กับชื่อของตัวคงที่

#include การกำหนดชื่อไฟล์ตามหลัง include จะใช้เครื่องหมาย <> ซึ่งจะเป็นการอ่านไฟล์จาก ไดเร็กทอรี หรือโฟลเดอร์ที่กำหนดไว้ก่อนแล้ว โดยปกติจะเป็นโฟลเดอร์ include แต่ถ้าใช้เครื่องหมาย “ ” เป็นการอ่านไฟล์จาก โฟลเดอร์ หรือไดเร็กทอรี ที่กำลังติดต่ออยู่และไฟล์ที่จะ include เข้ามานี้จะต้องไม่มีฟังก์ชัน main () โดยมากจะประกอบไปด้วยโปรแกรมย่อย ค่าคงที่ หรือข้อความต่าง ๆ

3) การแปลงค่า (Conversion)

char () คือ การแปลงค่าข้อมูลให้เป็น character

byte () คือ การแปลงค่าข้อมูลให้เป็น byte

int () คือ การแปลงค่าข้อมูลให้เป็น integer

4) Time

delay () คือ คำสั่งหยุดการทำงานโปรแกรมสำหรับจำนวนของเวลา (ใน milliseconds) Milliseconds = จำนวนมิลลิวินาทีในการหยุดการทำงานชั่วคราว

**Functions**

pinMode () ใช้ในกลุ่ม void setup () เพื่อกำหนดหน้าที่ขาของไมโครคอนโทรลเลอร์ให้เป็น ขารับสัญญาณ INPUT หรือขาส่งสัญญาณ OUTPUT

digitalWrite () คือ การส่งค่าลอจิก HIGH หรือ LOW (เปิด หรือปิด) ไปยังขา digital ที่กำหนดหมายเลขขาไอซีอาจกำหนดเป็นตัวแปรหรือค่าคงที่ [2]

**2.2) ไมโครคอนโทรลเลอร์ (Microcontroller)**

ไมโครคอนโทรลเลอร์ คือ เป็นคอมพิวเตอร์ขนาดเล็กบนวงจรรวมเดี่ยว ในคำศัพท์ที่ทันสมัยเป็นระบบในชิปหรือ SoC ไมโครคอนโทรลเลอร์มี CPU หนึ่งตัวหรือมากกว่า (แกนตัวประมวลผล) พร้อมกับหน่วยความจำและอุปกรณ์ต่อพ่วงอินพุต / เอาท์พุตที่ตั้งโปรแกรมได้ หน่วยความจำโปรแกรมในรูปแบบของ Ferroelectric RAM, NOR flash หรือ OTP ROM มักรวมอยู่ในชิปรวมทั้ง RAM ขนาดเล็ก ไมโครคอนโทรลเลอร์ได้รับการออกแบบมาสำหรับแอพพลิเคชันแบบฝังตัวซึ่งตรงกันข้ามกับไมโครโปรเซสเซอร์ที่ใช้ในคอมพิวเตอร์ส่วนบุคคลหรือแอพพลิเคชั่นวัตถุประสงค์ทั่วไปอื่น ๆ ซึ่งประกอบไปด้วยชิพแบบต่างๆไมโครคอนโทรลเลอร์จะใช้ในผลิตภัณฑ์และอุปกรณ์ที่มีการควบคุมโดยอัตโนมัติ โดยโครงสร้างของไมโครคอนโทรลเลอร์เป็นดังนี้

1.) หน่วยประมวลผลกลางหรือซีพียู (Central Processing Unit : CPU) ทำหน้าที่

เป็นศูนย์กลางควบคุมการทำงานของระบบคอมพิวเตอร์ทั้งหมด โดยจะนำข้อมูลจากอุปกรณ์ด้านอินพุตมา

ทำการประมวลผลข้อมูลตามคำสั่งของโปรแกรมและส่งผลลัพธ์ออกไปอุปกรณ์ด้านเอาต์พุต

BUS

2.) หน่วยความจำ (Memory) สามารถแบ่งออกเป็น 2 ส่วน คือ หน่วยความจำที่มี

ไว้สำหรับเก็บโปรแกรมหลัก (Program Memory) เปรียบเสมือนฮาร์ดดิสก์ของไมโครคอมพิวเตอร์ ข้อมูล

ใดๆ ที่ถูกเก็บไว้ในนี้จะไม่สูญหายไปแม้ไม่มีไฟเลี้ยง อีกส่วนหนึ่งคือหน่วยความจำข้อมูล (Data Memory)

ใช้เป็นเหมือนกระดาษทดในการคำนวณของซีพียูและเป็นที่พักข้อมูลชั่วคราวขณะทำงาน หากไม่มีไฟเลี้ยง

ข้อมูลก็จะหายไปคล้ายกับหน่วยความจำแบบแรมในไมโครคอมพิวเตอร์ทั่วไป สำหรับไมโครคอนโทรลเลอร์

สมัยใหม่ หน่วยความจำข้อมูลจะมีทั้งที่เป็นหน่วยความจำแบบแรม ซึ่งข้อมูลจะหายไปเมื่อไม่มีไฟเลี้ยง และ

เป็นอีอีพรอม (Erasable Electrically Read-Only Memory : EEPROM) ซึ่งสามารถเก็บข้อมูลได้แม้ไม่มี

ไฟเลี้ยง หรือเป็นหน่วยความจำแบบแฟลช (Flash Memory) ที่สามารถเขียนข้อมูลได้เป็นพันครั้ง

3.) หน่วยติดต่อกับอุปกรณ์ภายนอกหรือเรียกว่าพอร์ต (Port) มี 2 ลักษณะ ได้แก่

พอร์ตอินพุต (Input Port) และพอร์ตส่งสัญญาณหรือพอร์ตเอาต์พุต (Output Port) ส่วนนี้จะใช้ในการ

เชื่อมต่อกับอุปกรณ์ภายนอก ถือว่าเป็นส่วนที่สำคัญมาก ใช้ร่วมกันระหว่างพอร์ตอินพุตเพื่อรับสัญญาณ

อาจจะด้วยการกดสวิตช์ เพื่อนำไปประมวลผลและส่งไปพอร์ตเอาต์พุต เพื่อแสดงผล เช่น การติดสว่างของ

หลอดไฟ LED หรือจอแสดงผล เป็นต้น

4.) ช่องทางเดินของสัญญาณหรือบัส (BUS) คือ เส้นทางการแลกเปลี่ยนสัญญาณ

ข้อมูลระหว่างซีพียูกับหน่วยความจำและพอร์ต เป็นลักษณะของสายสัญญาณจำนวนมากที่อยู่ภายในตัวของ

ไมโครคอนโทรลเลอร์ โดยแบ่งเป็นบัสข้อมูล (Data Bus), บัสแอดเดรส (Address Bus) และบัสควบคุม

(Control Bus)

5.) วงจรกำเนิดสัญญาณนาฬิกา (Oscillator) เป็นส่วนประกอบสำคัญอีกส่วนหนึ่ง

เนื่องจากการทำงานที่เกิดขึ้นในตัวไมโครคอนโทรลเลอร์จะขึ้นอยู่กับการกำหนดจังหวะ ของวงจรกำเนิด

สัญญาณนาฬิกา หากสัญญาณนาฬิกามีความถี่สูง จังหวะการทำงานก็จะสามารถทำได้ถี่ขึ้นส่งผลให้การ

ทำงานของไมโครคอนโทรลเลอร์ตัวนั้น มีความเร็วในการประมวลผลสูงตามไปด้วย

นอกจากส่วนประกอบ 5 ส่วนที่กล่าวมา บริษัทผู้ผลิตไอซีไมโครคอนโทรลเลอร์ได้เพิ่มวงจรที่มี

คุณสมบัติพิเศษในตัวไมโครคอนโทรลเลอร์ เพื่อให้สะดวกต่อการนำไปใช้งานและการประยุกต์ใช้งานที่

หลากหลายทั้งกับอุปกรณ์ดิจิทัลและแอนะล็อก รวมทั้งอุปกรณ์ด้านการสื่อสาร ประกอบด้วยวงจรดังนี้

6.) วงจร ADC (Analog to Digital Converter) คือ วงจรส่วนอินพุตที่รับสัญญาณ

แอนะล็อกแล้วแปลงไปเป็นสัญญาณดิจิทัล เพื่อนำไปประมวลผล

7.) วงจร DAC (Digital to Analog Converter) คือ วงจรส่วนของอินพุตที่สามารถ

ประมวลผลสัญญาณแบบดิจิทัลแล้วแปลงไปเป็นสัญญาณแบบแอนะล็อก

8.) วงจร PWM (Pulse Width Modulation) คือ วงจรส่วนเอาต์พุตสำหรับสร้าง

สัญญาณพัลส์แบบสี่เหลี่ยม (Square wave) ที่สามารถปรับค่าความกว้าง (Duty Cycle) ของสัญญาณได้

เพื่อนำไปควบอุปกรณ์ต่าง ๆ เช่น ปรับความเร็วของมอเตอร์ เป็นต้น

เอกสารประกอบการเรียน วิชาไมโครคอนโทรลเลอร์(2105-2105) 11

หน่วยที่ 1 ความรู้เบื้องต้นเกี่ยวกับไมโครคอนโทรลเลอร์ เรียบเรียงโดยครูบุญเกิด สนธิพันธ์

9.) วงจร UART (Universal Asynchronous Receiver Transmitter) คือ วงจร

สื่อสารอนุกรมรับแบบอะซิงโครนัส (Asynchronous) สำหรับมาตรฐานการรับส่งข้อมูลแบบ RS-232

10.) วงจร I2C (Inter Integrate Circuit Bus) คือ วงจรสื่อสารอนุกรม

แบบซิงโครนัส (Synchronous) เพื่อใช้ติดต่อสื่อสารระหว่างไมโครคอนโทรลเลอร์กับอุปกรณ์ภายนอก ที่

พัฒนา ขึ้นโดยบริษัท Philips Semiconductors โดยใช้สายสัญญาณเพียง 2 เส้นเท่านั้น คือสาย Serial

Data (SDA) และสาย Serial Clock (SCL) ซึ่งสามารถเชื่อมต่ออุปกรณ์จำนวนหลายๆ ตัวเข้าด้วยกันได้

โดยใช้ขาพอร์ตเพียง 2 ขาเท่านั้น

11.) วงจร SPI (Serial Peripheral Interface) คือ วงจรการสื่อสารอนุกรมสำหรับ

รับส่งข้อมูลกับอุปกรณ์ภายนอกแบบซิงโครนัส (Synchronize) โดยที่มีสัญญาณนาฬิกา (Clock) ควบคุม

ระหว่างไมโครคอนโทรลเลอร์ (Microcontroller) กับอุปกรณ์ภายนอก การรับส่งข้อมูลแบบ SPI นี้จะมี

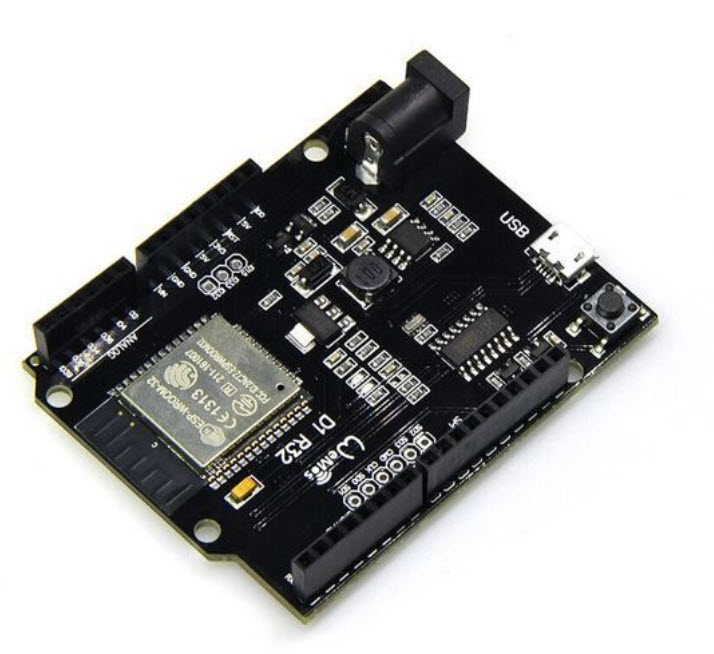
อุปกรณ์ที่ทำหน้าที่เป็น มาสเตอร์ (Master) ซึ่งโดยปกติแล้วจะเป็นไมโครคอนโทรลเลอร์ หรืออาจกล่าวได้

ว่าอุปกรณ์ Master จะควบคุมอุปกรณ์ Slave ได้ และปกติตัว Slave มักจะเป็นไอซีพิเศษต่าง ๆ เช่น ไอซี

วัดค่าอุณหภูมิ, ไอซีกำเนิดฐานเวลานาฬิกาจริง (Real-Time Clock) หรืออาจเป็นไมโครคอนโทรลเลอร์ที่

ทำหน้าที่ในโหมด Slave [3]

ไมโครคอนโทรลเลอร์ที่ผู้จัดทำใช้คือ Arduino wemos D1 R32 แสดงดังรูปที่ 1



รูปที่ 1 Arduino wemos D1 R32

คุณสมบัติของ คือ Arduino wemos D1 R32 มีดังนี้

- ไฟเลี้ยง VCC: 3.3-5 Vdc

- ดิจิตอลเอาต์พุต (0 หรือ 1)

- ระยะตรวจจับ สามารถปรับได้ตั้งแต่ 2-30 cm

- มุมในการตรวจจับ 35 องศา

- ขนาดบอร์ด 3.1 x 1.5 cm

**2.3) โมดูลเซ็นเซอร์แสงสำหรับตรวจจับวัตถุกีดขวาง IR Infrared Obstacle Avoidance Sensor Module**

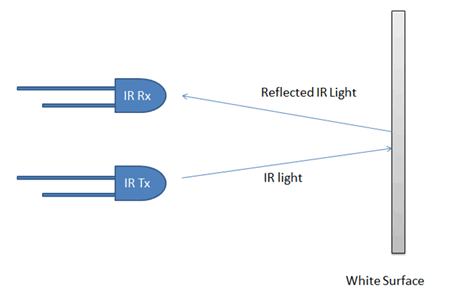
โดยโมดูลนี้สะแสดงดังภาพที่ 2 โมดูลนี้ จะมีตัวรับและตัวส่ง infrared ในตัว ตัวสัญญาณ(สีขาว) infrared จะส่งสัญญาณออกมา และเมื่อมีวัตถุมาบัง คลื่นสัญญาณ infrared ที่ถูกสั่งออกมาจะสะท้องกลับไปเข้าตัวรับสัญญาณ (สีดำ) สามารถนำมาใช้ตรวจจับวัตถุที่อยู่ตรงหน้าได้ และสามารถปรับความไว ระยะการตรวจจับ ใกล้หรือไกลได้



รูปที่ 2 เซนเซอร์ตรวจจับสิ่งกีดขวาง

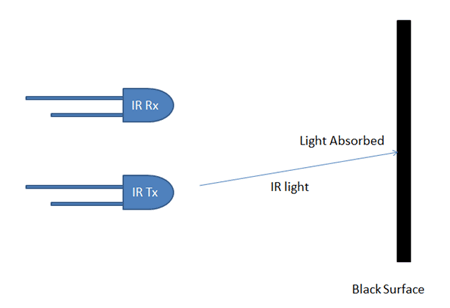
ภายตัวเซ็นเซอร์แบบนี้จะมีตัวส่ง Emitter และ ตัวรับ Receiver ติดตั้งภายในตัวเดียวกัน ทำให้ไม่จำเป็นต้องเดินสายไฟทั้งสองฝั่ง เหมือนแบบ Opposed Mode ทำให้การติดตั้งใช้งานได้ง่ายกว่า แต่อย่างไรก็ตามจำเป็นต้องติดตั้งตัวแผ่นสะท้อนหรือ Reflector ไว้ตรงข้ามกับตัวเซ็นเซอร์เอง โดยโฟโต้เซ็นเซอร์แบบที่ใช้แผ่นสะท้อนแบบนี้จะเหมาะสำหรับชิ้นงานที่มีลักษณะทึบแสงไม่เป็นมันวาว เนื่องจากอาจทำให้ตัวเซ็นเซอร์เข้าใจผิดว่าเป็นตัวแผ่นสะท้อน และ ทำให้ทำงานผิดพลาดได้

เซ็นเซอร์แบบนี้จะมีช่วงในการทำงาน หรือ ระยะในการตรวจจับจะได้ใกล้กว่าแบบ Opposed mode ซึ่งในสภาวะการทำงานปกติตัวรับ Receiver จะสามารถรับสัญญาณแสงจากตัวส่ง Emitter ได้ตลอดเวลา เนื่องจากลำแสงจะสะท้อนกับแผ่นสะท้อน Reflector อยู่ตลอดเวลา จะแสดงค่า เป็น 0



รูปที่ 3 แสดงการทำงานเมื่อตรวจพบวัตถุ

หน้าที่หลักของเซ็นเซอร์ชนิดนี้ จะคอยตรวจจับวัตถุที่เคลื่อนที่ตัดผ่านหน้าเซ็นเซอร์ เมื่อวัตถุ หรือ ชิ้นงานผ่านเข้ามาที่หน้าเซ็นเซอร์ แล้วจะการขวางลำแสงที่ส่งจากตัวส่ง Emitter ที่ส่งไปยังแผ่นสะท้อน จึงทำให้ตัวรับ Receiver ไม่สามารถรับลำแสงที่จะสะท้อนกลับมาได้ จะแสดงค่า เป็น 1



รูปที่ 4 แสดงการทำงานเมื่อตรวจพบวัตถุทึบแสง

หน้าที่หลักของเซ็นเซอร์ชนิดนี้ จะคอยตรวจจับวัตถุที่เคลื่อนที่ตัดผ่านหน้าเซ็นเซอร์ เมื่อวัตถุ หรือ ชิ้นงานผ่านเข้ามาที่หน้าเซ็นเซอร์ แล้วจะการขวางลำแสงที่ส่งจากตัวส่ง Emitter ที่ส่งไปยังแผ่นสะท้อน จึงทำให้ตัวรับ Receiver ไม่สามารถรับลำแสงที่จะสะท้อนกลับมาได้ จะแสดงค่า เป็น 1

ซึ่งจะทำให้วงจรภายในรับรู้ได้ว่า มีวัตถุหรือชิ้นงานขวางอยู่ ทำให้สถานะของเอาท์พุตของตัวรับเปลี่ยนแปลงไป โดยเราเรียกลักษณะการทำงานแบบนี้ว่า Dark On หรือ Dark Operate [4]

**2.4) เซ็นเซอร์วัดระยะทาง Ultrasonic Module**

เซนเซอร์วัดระยะทางด้วย Ultrasonic ใช้หลักการ ส่งคลื่นเสียงความถี่ต่ำ Ultrasonic ไปเมื่อคลื่นเสียงกระทบกับวัตถุจะมีการสะท้อนกลับมา เซนเซอร์จับเวลาที่ส่งคลื่นเสียงออกไปจนถึงคลื่นเสียงสะท้อนกลับมา เมื่อนำมาคำนวณกับเวลาที่เสียงเดินทางในอากาศ ก็จะได้ระยะทางออกมา

โมดูล Ultrasonic ตรวจจับวัตถุ คำนวณระยะทางโดยใช้คลื่น มีลักษณะเป็นกรวยและไม่ใช่เส้นตรง จึงเหมาะสำหรับใช้ตรวจจับสิ่งกีดขวางด้วย ระยะเวลาที่ได้จากการวัดช่วงเวลาการเดินทางไปและกลับนี้ เราสามารถนำค่าที่ได้ไปเปรียบเทียบกับอัตราเร็วที่เสียงสามารถเดินทางได้ไปในอากาศได้เลย โดยอัตราเร็วเสียงที่เดินทางได้ในอากาศสามารถหาได้ตามสูตร

อัตราเร็วของเสียงในอากาศ = 331 + (0.606 \* อุณหภูมิในหน่วยองศาเซลเซียส) m/s

สังเกตว่า อัตราเร็วของเสียงที่เดินทางในอากาศนั้น จะขึ้นอยู่กับอุณหภูมิ ณ ขณะนั้นด้วย ดังนั้นในเซ็นเซอร์อัลตร้าโซนิคบางรุ่น จึงมีเซ็นเซอร์วัดอุณหภูมิมาด้วย ทำให้สามารถวัดระยะทางได้แม่นยำมากยิ่งขึ้น สำหรับในรุ่นที่ไม่มีเซ็นเซอร์วัดอุณหภูมิ ท่านสามารถนำเซ็นเซอร์วัดอุณหภูมิมาต่อเพื่อแก้ค่าความผิดพลาดเองได้ หรือใช้ค่าอุณหภูมิเฉลี่ยทั้งปีของประเทศไทยได้ โดยอุณหภูมิเฉลี่ยของประเทศไทยทั้งปีจะอยู่ที่ 27 องศาเซลเซียส [5]



รูปที่ 5 เซนเซอร์วัดระยะทาง

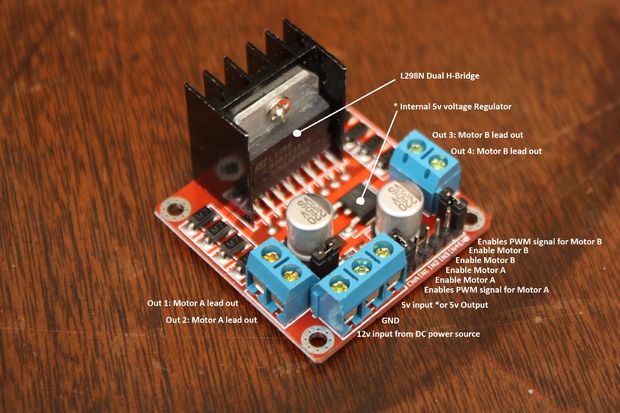
**2.5) DC Motor Speed Control**

ประกอบด้วย 2 ส่วนใหญ่ ๆ คือ 1) H-bridge Driver และ 2) Pulse-width modulation (PWM) สิ่งที่เป็นพลังขับเคลื่อนหลักให้กับหุ่นยนต์นั้น ก็คงจะไม่พ้น มอเตอร์ ซึ่งต้องการการควบคุม จาก Motor Driver ที่จะมาควบคุม ทั้ง ทิศทาง และ ความเร็ว ของมอเตอร์ซึ่งทางผู้จัดทำได้เลือกใช้ L298N Dual H-Bridge Motor Controller โดยหลักการทำงานวงจร H-Bridge ของ L298N จะขับกระแสเข้ามอเตอร์ ตามขั้วที่กำหนดด้วยลอจิกเพื่อควบคุมทิศทาง ส่วนความเร็วของมอเตอร์นั้นจะถูกควบคุมด้วย สัญญาณ PWM (Pulse Width Modulation) เป็นวิธีการควบคุมการจ่ายกำลังโดยการปรับความกว้างของสัญญาณ Pulse ด้วยความถี่สูงเพื่อให้ได้กำลังเฉลี่ยเป็นไปตามส่วนที่ต้องการ ซึ่งต้องมีการปรับความถี่ให้เหมาะสมกับเป็นพารามิเตอร์ที่ใช้กำหนดสัดส่วนการทำงาน (ON) ของ Load (มอเตอร์)

วงจร H-Bridge ของ L298N จะขับกระแสเข้ามอเตอร์ ตามขั้วที่กำหนดด้วยลอจิคเพื่อควบคุมทิศทาง

ส่วนความเร็วของมอเตอร์นั้นจะถูกควบคุมด้วย สัญญาณ (PWM Pulse Width Modulation)

ซึ่งต้องมีการปรับความถี่ให้เหมาะสมกับมอเตอร์ที่จะใช้



รูปที่ 6 ส่วนประกอบของ L298N Dual H-Bridge Motor Controller

**ขาต่างๆ**

Out 1: ช่องต่อขั้วไฟของมอเตอร์ A

Out 2: ช่องต่อขั้วไฟของมอเตอร์ A

Out 3: ช่องต่อขั้วไฟของมอเตอร์ B

Out 4: ช่องต่อขั้วไฟของมอเตอร์ B

12V: ช่องจ่ายไฟเลี้ยงมอเตอร์ 12V (ต่อได้ตั้งแต่ 5V ถึง 35V)

GND: ช่องต่อไฟลบ (Ground)

5V: ช่องจ่ายไฟเลี้ยงมอเตอร์ 5V (หากมีการต่อไฟเลี้ยงที่ช่อง 12V แล้ว ช่องนี้จะทำหน้าที่จ่ายไฟออก เป็น 5V Output สามารถต่อไฟจากช่องนี้ไปเลี้ยงบอร์ด Arduino ได้

ENA: ช่องต่อสัญญาณ PWM สำหรับมอเตอร์ A

IN1: ช่องต่อสัญญาณลอจิคเพื่อควบคุมทิศทางของมอเตอร์ A

IN2: ช่องต่อสัญญาณลอจิคเพื่อควบคุมทิศทางของมอเตอร์ A

IN3: ช่องต่อสัญญาณลอจิคเพื่อควบคุมทิศทางของมอเตอร์ B

IN4: ช่องต่อสัญญาณลอจิคเพื่อควบคุมทิศทางของมอเตอร์ B

ENB : ช่องต่อสัญญาณ PWM สำหรับมอเตอร์ B

**สเปกทางเทคนิค**

Dual H bridge Drive Chip : L298N

แรงดันสัญญาณลอจิค : 5V Drive voltage: 5V-35V

กระแสของสัญญาณลอจิค : 0-36mA

กระแสขับมอเตอร์ : สูงสุดที่ 2A (เมื่อใช้มอเตอร์เดียว)

กำลังไฟฟ้าสูงสุด : 25W

ขนาด : 43 x 43 x 26 มิลลิเมตร

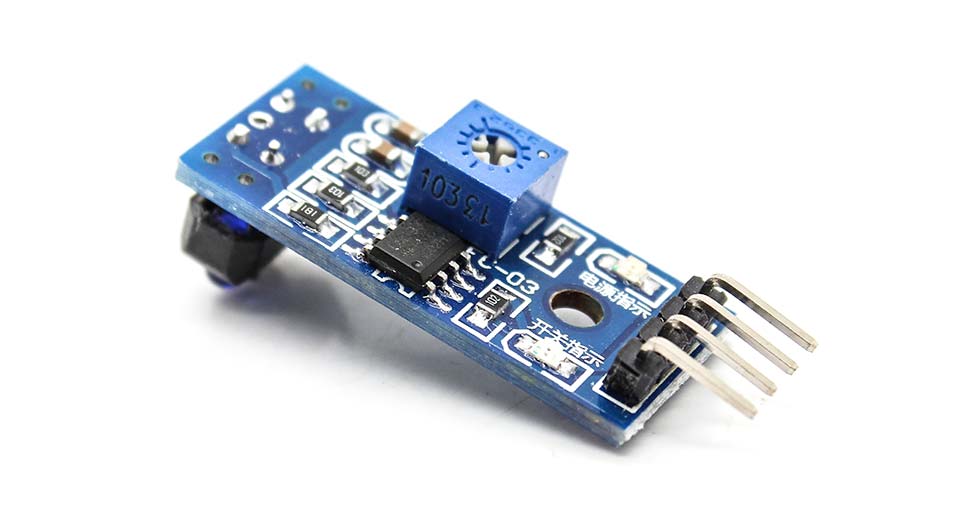
น้ำหนัก : 26 กรัม

มี Power Supply 5V ในตัว สามารถจ่ายไฟออกจากช่อง 5V (เพื่อจ่ายให้บอร์ด Arduino) ได้

เมื่อต่อไฟเลี้ยงเข้าที่ช่อง 12V [6]

**2.6) TCRT5000 Infrared Reflective sensor**

เป็นโมดูลตรวจจับวัตถุระยะใกล้ มีราคาถูก ขนาดเล็ก สะดวกในการนำไปใช้ติดตั้งกับงานจำพวก หุ่นยนต์, Smart car, หุ่นยนต์หลบสิ่งกีดขวาง เป็นต้น โดยการทำงานของตัวโมดูลนี้ เริ่มต้นโดยให้ หลอด Infrared LED ทำการส่งสัญญาณ เป็นแสงอินฟราเรดออกไปตกกระทบกับวัตถุที่ตรวจพบในระยะ และทำการสะท้อนกลับมายังตัวหลอดโฟโต้ไดโอดที่ทำหน้าที่รับแสงอินฟราเรด โดยส่วนมาก ตัวโมดูลจะให้ค่า output ออกมาเป็น Digital signal แต่สำหรับบางโมดูลอาจจะรองรับ output แบบ Analog signal ด้วย ส่วนตัว R ปรับค่านั้นใช้ในการปรับความไวต่อการตรวจจับแสงอินฟราเรด ซึ่งจะส่งผลต่อระยะในการตรวจพบวัตถุของตัวเซนเซอร์

  
รูปที่ 7 โมดูลตรวจจับวัตถุ TCRT5000

**คุณสมบัติของอุปกรณ์**

Package type: leaded

Detector type: phototransistor

Dimensions (L x W x H in mm): 10.2 x 5.8 x 7

Peak operating distance: 2.5 mm

Operating range within > 20 % relative collector current: 0.2 mm to 15 mm

Typical output current under test: IC = 1 mA

Daylight blocking filter

Emitter wavelength: 950 nm

Lead (Pb)-free soldering released

Compliant to RoHS directive 2002/95/EC and in accordance to WEEE 2002/96/EC [7]

**2.7) วงจร DC/DC Step-up (แรงดันปรับค่าได้)**

เป็นวงจรที่ใช้เพิ่มแรงดันทางไฟฟ้า โดยที่เป็นผู้จัดทำใช้รุ่น MT3608



รูปที่ 8 วงจร DC / DC step up รุ่น MT3608

**คุณสมบัติของรุ่น MT3608**

แรงดันไฟฟ้าขาเข้า 2V ถึง 24V พร้อมขาออกเพิ่ม 5V ถึง 28V ปรับได้กระแสไฟฟ้าขาออกสูงสุด: 2A

แรงดันไฟฟ้าเข้า: 2V ~ 24V

แรงดันไฟฟ้าขาเข้าสูงสุด: > 5V - 28V

ประสิทธิภาพ : > 93%

แรงดันไฟฟ้าขาเข้าไม่ควรเกินแรงดันไฟฟ้าสูงสุด

กระแสไฟขาออกสูงสุดในปัจจุบันไม่ควรเกิน 2A

แรงดันขาออกควรสูงกว่าแรงดันไฟฟ้าขาเข้าเสมอ

ขนาด : 36 mm \* 17 mm \* 14 mm [8]

**2.8) ภาษา C#**

C# คือ ภาษาคอมพิวเตอร์ประเภท object-oriented programming พัฒนาโดย Microsoft โดยมีจุดมุ่งหมายในการวมความสามารถการคำนวณของ C++ ด้วยการโปรแกรมง่ายกว่าของ Visual Basic โดย C# มีพื้นฐานจาก C++ และเก็บส่วนการทำงานคล้ายกับ Java

C# ได้รับการออกแบบให้ทำงานกับ .NET platform ของ Microsoft จุดมุ่งหมายคือ อำนวยความสะดวกในการแลกเปลี่ยนสารสนเทศและบริการผ่านเว็บ และทำให้ผู้พัฒนาสร้างโปรแกรมประยุกต์ในขนาดกระทัดรัด C# ทำให้โปรแกรมง่ายขึ้นผ่านการใช้ Extensible Markup Language (XML) และ Simple Object Access Protocol (SOAP) ซึ่งยอมให้เข้าถึงอ๊อบเจคของโปรแกรมหรือเมธอด โดยปราศจากความต้องการให้ผู้เขียนโปรแกรมเขียนคำสั่งเพิ่มในแต่ละขั้นตอน เนื่องจากผู้เขียนโปรแกรมสามารถสร้างบนคำสั่งที่มีอยู่ แทนที่การคัดลอกซ้ำ C# ภาษา C# ถูกพัฒนาขึ้นโดยเป็นส่วนหนึ่งในการพัฒนาโครงสร้างพื้นฐานของ .NET Framework เป็นการการนำข้อดีของภาษาต่างๆ (เช่นภาษา Delphi , ภาษา C++) มาปรับปรุงเพื่อให้มีความเป็น OOP (โปรแกรมเชิงวัตถุ) มากขึ้น ขณะเดียวกันก็ลดความซับซ้อนในโครงสร้างของภาษาลง (เรียบง่ายกว่าภาษา C++) และมีสิ่งที่เกินความจำเป็นน้อยลง (เมื่อเทียบกับ Java)

C# ถูกรับรองจากหน่วยงาน ECMA (หน่วยงานกำหนดมาตรฐานสากลด้านสารสนเทศ) และ ISO และปัจจุบันไมโครซอฟท์ยังพัฒนาภาษานี้อย่างต่อเนื่อง

**จุดเด่นหลักๆ ของภาษา C#**

1.Component oriented – เป็นภาษาที่เน้นชิ้นส่วนโดยถูกออกแบบมาเป็นอย่างดีทำให้สามารถนำมาใช้ต่อกันเป็นอะไรก็ได้

2.สิ่งต่าง ๆ ใน C# เป็นออบเจ็กต์ทั้งหมด

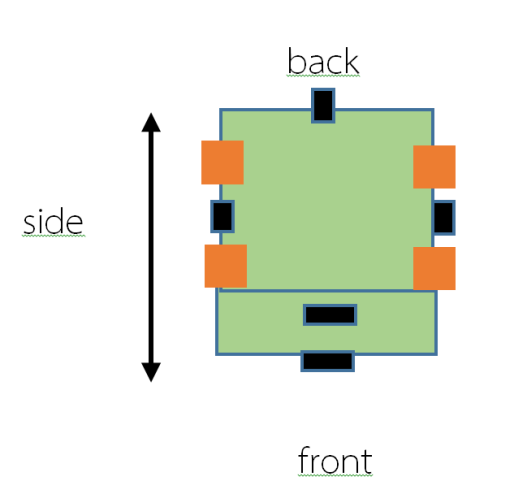
3.เป็นภาษา ที่ทนทาน (robust) - ทนต่อความผิดพลาด ไม่ทำให้ระบบแฮงก์หรือระบบทำงานช้า เพราะ C# มีข้อดีคือ garbage collection , exception , type-safety และ versioning

4.ภาษา C# จัดเตรียมกลไกไว้หลายอย่างที่ช่วยให้ผู้เขียนโปรแกรมสามารถนำโค้ดที่เขียนไว้ใน โปรเจค หนึ่งไปใช้กับอีกโปรเจคหนึ่งได้ง่าย นอกจากนั้นภาษา C# ยังสามารถเรียกใช้คลาสหลายพันคลาสใน .NET Framework ได้โดยตรง ทำให้ลดเวลาการพัฒนาซอฟท์แวร์ได้มาก [9]

**บทที่ 3**

**การออกแบบและการจัดทำโครงงาน**

**3.1) กลยุทธ์ฝ่ายรุก**



1.เช็ค sensor ล่างก่อน (ตรวจจับสี) : ทำงานคือพบสี/ไม่ทำงานคือไม่พบสี

1.1 ไม่ทำงาน : เดินหน้า ในกรณี sensor ด้านหน้า และ ด้านข้างไม่ทำงาน ( ไม่เจอรถฝั่งตรงข้ามใน

ระยะ 10 ซม. และ ไม่เจอรถที่มาจากทางด้านข้าง )

1.2 ไม่ทำงาน : break ในกรณี sensor ด้านหน้าทำงาน แต่ด้านข้างไม่ทำงาน ( เมื่อมีรถอยู่ด้านหน้าในระยะ 10 ซม. และไม่เจอรถที่อยู่ด้านข้าง )

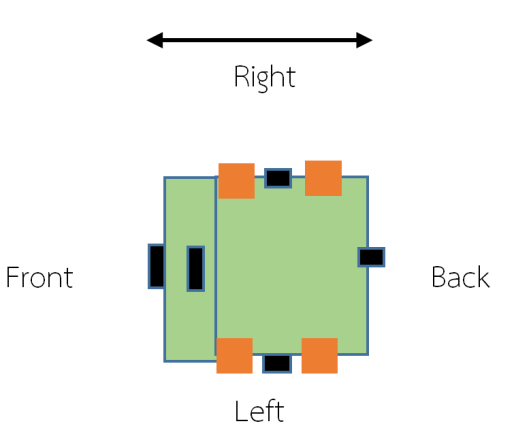
1.3 ไม่ทำงาน : ถอยหลัง ในกรณี sensor ด้านหน้า และ ด้านข้าง ทำงาน ( เจอรถข้างหน้าในระยะ 10 ซม. พร้อมกับ มีรถมาข้างๆ)

1.4 ทำงาน : กลับรถ ในกรณีตรวจจับสีแดงได้

1.5 ทำงาน : Stop ในกรณีตรวจจับสีเหลืองได้

หมายเหตุ : จากรูป สีดำคือ sensor , สีเขียว คือตัวโครงรถ และ สีส้มคือล้อ

**3.2) กลยุทธ์ฝ่ายรับ**

****

1. เช็ค sensor ด้านล่าง

1.1 ไม่ทำงาน : ถอยหลัง ในกรณี sensor ด้านหน้าทำงาน ( พบเจอรถในระยะ 5 ซม.) และ sensor ด้านหลังและด้านขวาหรือด้านซ้ายไม่ทำงาน

1.2 ไม่ทำงาน : เดินหน้า ในกรณี sensor ด้านหลังทำงาน ( พบเจอรถ ) และ sensor ด้านหน้าและด้านขวาหรือด้านซ้ายไม่ทำงาน

1.3 ไม่ทำงาน : Break รถ ในกรณี sensor ด้านขวาทำงานหรือด้านซ้าย (พบเจอรถศัตรู) และ sensor ด้านหน้าและด้านหลังไม่ทำงาน

1.4 ไม่ทำงาน : ถอยหลังอย่างช้าๆ ในกรณี ในกรณี sensor ด้านขวาหรือด้านซ้ายทำงาน (พบเจอรถศัตรู) และ sensor ด้านหน้าทำงาน (เจอรถเพื่อนในระยะ 5 ซม.) และด้านหลังไม่ทำงาน

1.5 ไม่ทำงาน : เดินหน้าอย่างช้าๆ ในกรณี ในกรณี sensor ด้านขวาทำงานหรือด้านซ้าย (พบเจอรถศัตรู) และ sensor ด้านหลังทำงาน (เจอรถเพื่อน) และด้านหน้าไม่ทำงาน

1.6 ทำงาน : กลับรถ ในกรณีตรวจจับเส้นสีดำ

หมายเหตุ : จากรูป สีดำคือ sensor , สีเขียว คือตัวโครงรถ และ สีส้มคือล้อ

**3.3) การออกแบบการทำงานของวงจร**

**อุปกรณ์**

1. Arduino board (wemos D1 R32 ) จำนวน 1 ชิ้น

2. 18650 Li-ion battery, Battery case, Li-ion Charging module จำนวนอย่างละ 1 ชิ้น

3. DC Geared-Motors จำนวน 2 ก้อน

4. H-bridge Driver จำนวน 1 ชิ้น

5. Breadboard จำนวน 1 ชิ้น

6. DC/DC Step-up Converter จำนวน 1 ชิ้น

7. TCRT5000 Infrared Reflective sensor จำนวน 1 ชิ้น

8. IR Infrared Obstacle Avoidance Sensor จำนวน 3 ชิ้น

9. Ultrasonic Sensor จำนวน 1 ชิ้น

10. LED จำนวน 2 ชิ้น

11. ตัวต้านทาน 1 ชิ้น

12. Push Button จำนวน 1 ชิ้น

**การทำงานของวงจร**

1. D/C Motor Driver Module

ENA: ช่องต่อสัญญาณ PWM สำหรับมอเตอร์ A โดยมี pinMode อยู่ที่ 7

IN1: ช่องต่อสัญญาณ logic เพื่อควบคุมทิศทางของมอเตอร์ A โดยมี pinMode อยู่ที่ 6

IN2: ช่องต่อสัญญาณ logic เพื่อควบคุมทิศทางของมอเตอร์ A โดยมี pinMode อยู่ที่ 5

IN3: ช่องต่อสัญญาณ logic เพื่อควบคุมทิศทางของมอเตอร์ B โดยมี pinMode อยู่ที่ 4

IN4: ช่องต่อสัญญาณ logic เพื่อควบคุมทิศทางของมอเตอร์ B โดยมี pinMode อยู่ที่ 3

ENB: ช่องต่อสัญญาณ PWM สำหรับมอเตอร์ B โดยมี pinMode อยู่ที่ 2

2. Obstacle Sensor (เซนเซอร์ตรวจจับวัตถุ)

มีการใช้แรงดันไฟฟ้าในการทำงาน 3V - 5.5V มีช่องที่ต่ออยู่กับไมโครคอนโทรเลอร์ เพื่อเป็นตัว input แรงดันไฟฟ้า ทำหน้าที่รับค่าจากการตรวจจับวัตถุ โดยผู้จัดได้มีการติดตั้งเซนเซอร์ตรวจจับวัตถุนี้ 3 ตำแหน่งด้วยกัน

1.) ตำแหน่งทางด้านซ้ายของรถ โดยมี pinMode อยู่ที่ 11

2.) ตำแหน่งทางด้านขวาของรถ โดยมี pinMode อยู่ที่ 10

3.) ตำแหน่งทางด้านหลังของรถ โดยมี pinMode อยู่ที่ 13

3. Ultrasonic

ตัว VCC จะถูกใช้เป็นตัวควบคุมการเริ่มทำงาน หรือ หยุดทำงานของอุปกรณ์ จากนั้นสัญญาณเริ่มต้นในการปล่อยพัลส์จะถูกส่งไปที่ ขา Trig และ สัญญาณ Echo จะถูกส่งกลับไปที่ตัว Arduino เพื่อคำนวณระยะทางต่อไป

Trig มี pinMode อยู่ที่ 9 , Echo มี pinMode อยู่ที่ 8

4. TCRT5000 (Infrared Reflectance Sensor)

Sensor จะใช้หลักการสะท้อนกลับของแสง กระดาษสีดำจะดูดกลืนแสง กระดาษสีขาว sensor จะสะท้อนแสงได้ดี ทำให้ค่าที่อ่านได้ต่างกันบนตัว sensor ที่ค่าสัญญาณ analog A0 และ ค่า สัญญาณ digital D0 หรือหมายถึงมีทั้งค่า Analog และ Digital

Analog มี pinMode อยู่ที่ A0

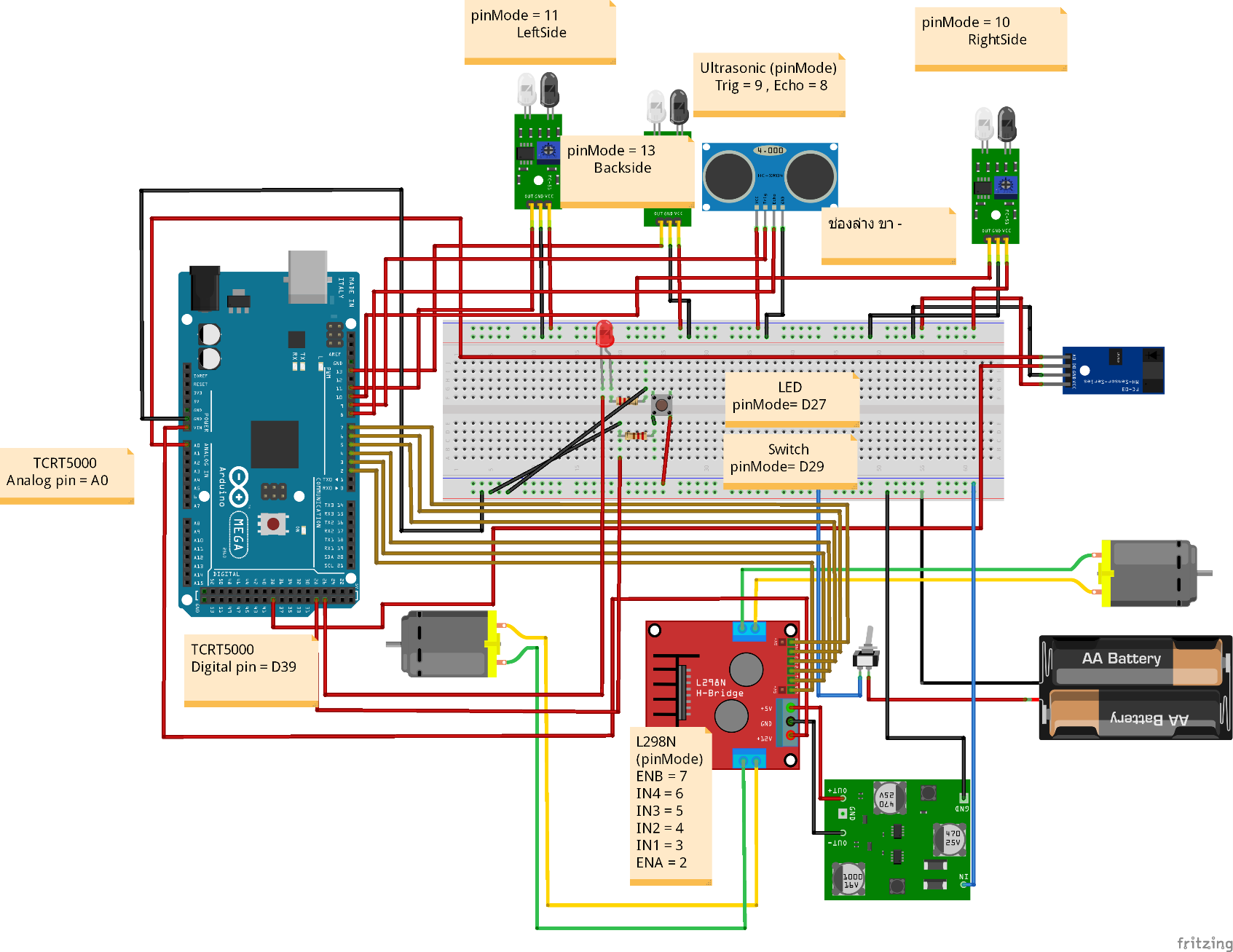
Digital มี pinMode อยู่ที่ D39

นอกจากนั้นยังมีการต่อ Push Button และ LED เพื่อสะดวกในการเปลี่ยนกลยุทธ์จากรุกเป็นรับ หรือ Offender เป็น Defender โดยการต่อ Push Button นั้น โดยทางผู้จัดทำนั้นได้มีการต่อวงจรแบบ Pull-Down Resistor

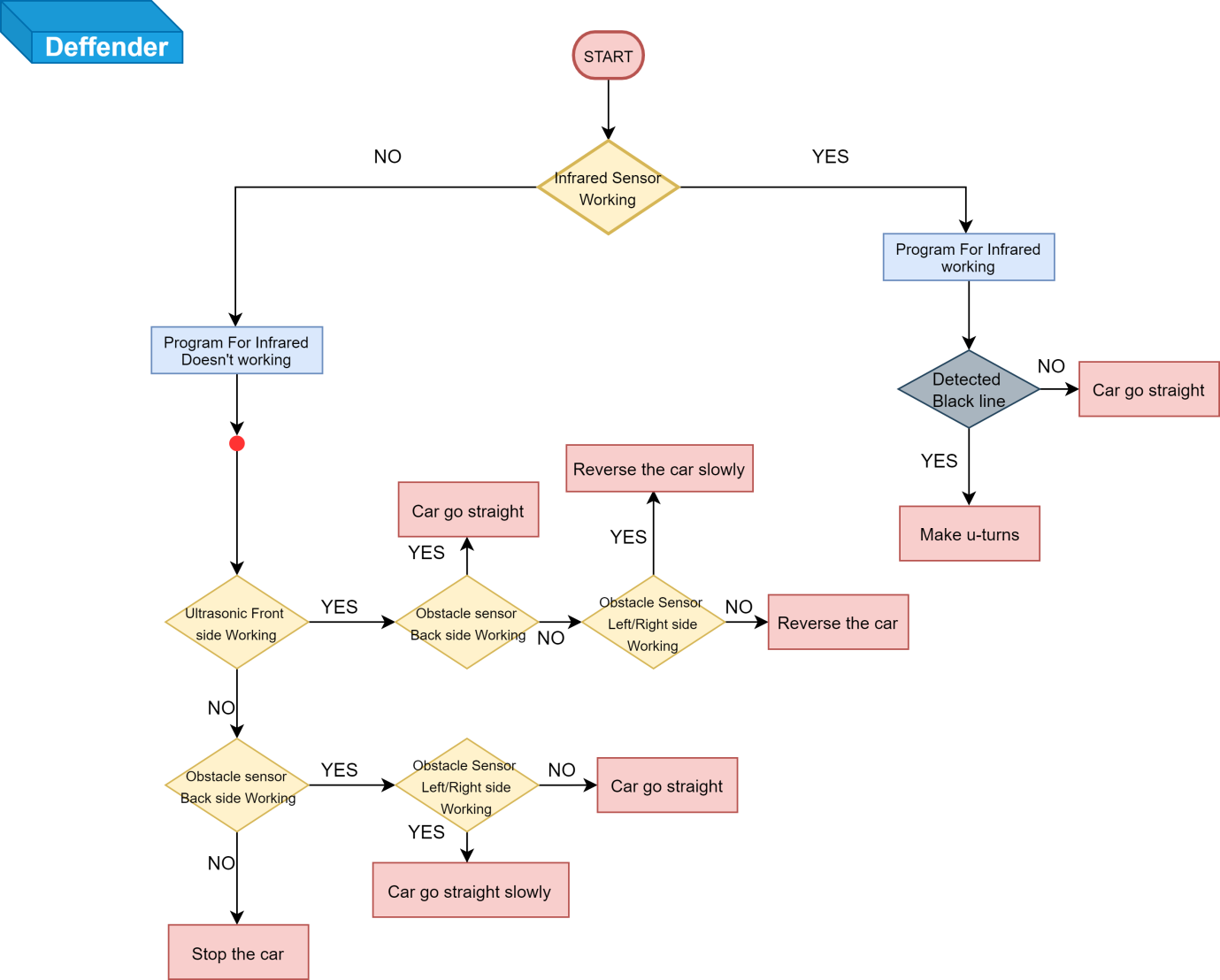
คือเมื่อไม่ได้กดปุ่ม จะอ่านค่าได้เป็น LOW และเมื่อกดปุ่ม จะอ่านค่าได้เป็น HIGH โดยทั้ง Push Button และ LED เป็นการรับส่งสัญญาณแบบ Digital และต้องมีตัว resistance ทั้ง 2 ตัวอีกด้วย

Push Button มี pinMode อยู่ที่ D29

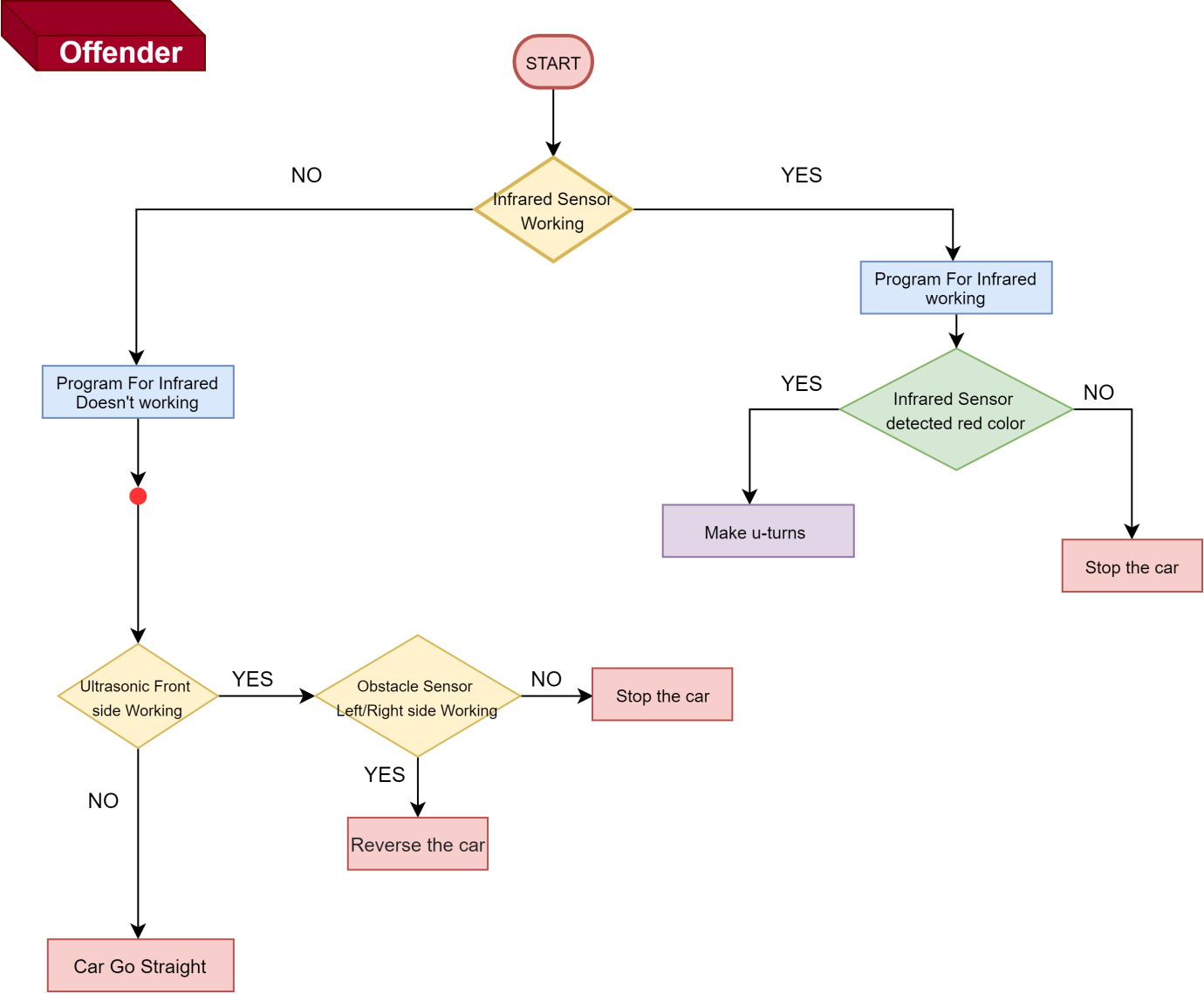
LED มี pinMode D27



รูปที่ 9 การออกแบบวงจรเพื่อเชื่อมต่ออุปกรณ์ต่าง ๆ ภายในรถ

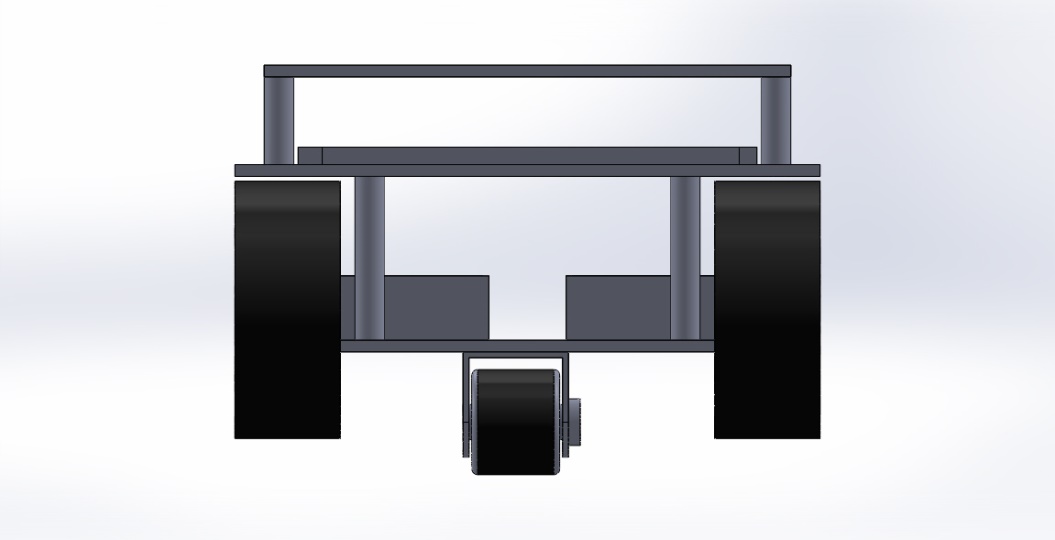


รูปที่ 10 Flowchart กลยุทธ์ฝ่ายรับ

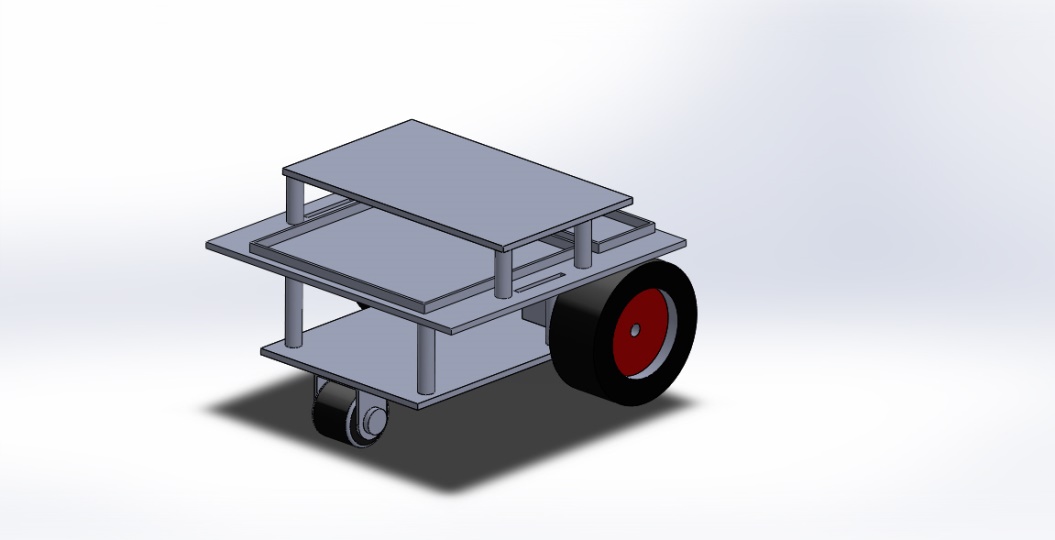


รูปที่ 11 Flowchart กลยุทธ์ฝ่ายรุก

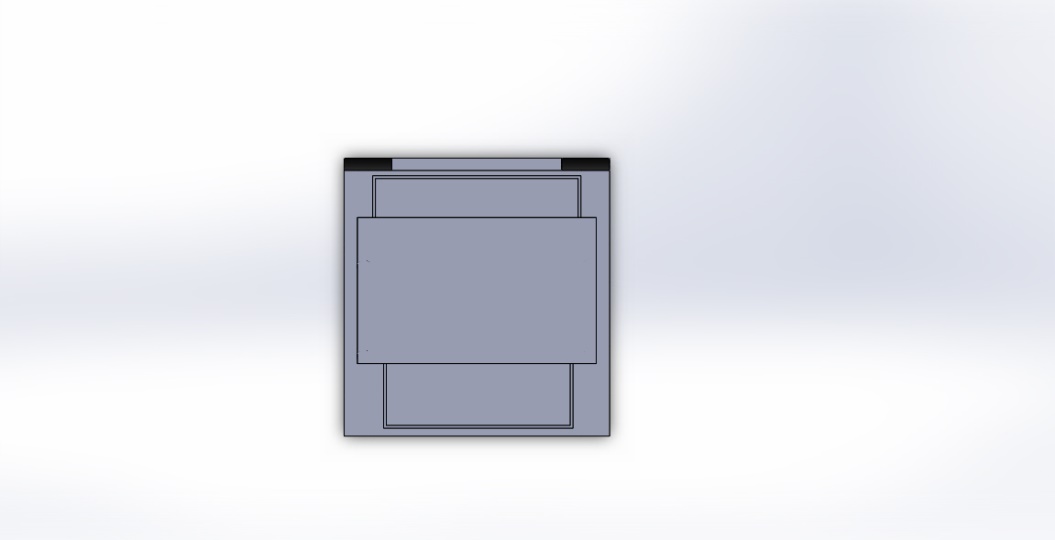
**3.4) การออกแบบตัวหุ่นยนต์**

****

รูปที่ 12 มุมมองด้านหน้า

****

รูปที่ 13 มุมมอง Isometric

****

รูปที่ 14 มุมมองด้านบน

**3.5) Programming Code**

#include <HCSR04.h>

HCSR04 hc(8,9); //initialisation class HCSR04 (trig,echo);

#define ia1 3 //motor 1 เดินหน้า

#define ia2 4 //moter 1 ถอยหลัง

#define ib1 5 //motor 2 เดินหน้า

#define ib2 6 //motor 2 ถอยหลัง

#define ls 11 //sensorซ้าย

#define rs 10 //sensorขวา

#define bs 13 //sensorหลัง

#define RED 1

#define YELLOW 2

#define BLACK 3

#define Number\_color 0

#define maxSpd 255 // motor max speed

int sensor\_color = A0;

int led1 = D27;

int buttonPin = D29;

int val=0;

int old\_val=0;

int state=0;

int color=0;

int getColor() {

int Number\_color = analogRead(sensor\_color);

if ((Number\_color>2200)&&(Number\_color<2500)) //สีแดง

return RED;

else if ((Number\_color>1900)&&(Number\_color<2200)) //สีเหลือง

return YELLOW;

else if ((Number\_color>3600)&&(Number\_color<3900)) //สีดำ

return BLACK;

else //ไม่พบสี

return Number\_color;

}

void setup() {

// put your setup code here, to run once:

pinMode(ls, INPUT);

pinMode(rs, INPUT);

pinMode(bs, INPUT);

pinMode(buttonPin, INPUT);

pinMode(led1, OUTPUT);

pinMode(ia1, OUTPUT);

pinMode(ia2, OUTPUT);

pinMode(ib1, OUTPUT);

pinMode(ib2, OUTPUT);

Serial.begin(9600);

}

void loop()

{

color = getColor();

val = digitalRead(buttonPin);

if( (val==HIGH) && (old\_val==LOW))

{

state=!state;

}

old\_val=val;

if (state==1) //เมื่อกดสวิทซ์ 1 ครั้ง ใช้กลยุทธ์รุก

{

digitalWrite(led1, HIGH);

if(color == Number\_color) // sensor ด้านล่างไม่ทำงาน

{

if((hc.dist()>10)&&((digitalRead(ls)==HIGH)||(digitalRead(rs)==HIGH))) // เดินหน้าเมื่อ sensor ด้านหน้า และ ด้านข้าง ไม่ทำงาน

{

aForward(maxSpd);

bForward(maxSpd);

}

if((hc.dist()<10)&&((digitalRead(ls)==HIGH)||(digitalRead(rs)==HIGH))) // break เมื่อ sensor ด้านหน้าทำงาน แต่ด้านข้างไม่ทำงาน

{

aBreakTime(1000);

bBreakTime(1000);

}

if((hc.dist()<10)&&((digitalRead(ls)==LOW)||(digitalRead(rs)==LOW))) // เดินถอยหลัง 3วิ เมื่อ sensor ด้านหน้าและข้างซ้ายขวาทำงาน

{

aBackwardTime(3000);

bBackwardTime(3000);

}

}

if(color == RED) // เดินกลับรถ เมื่อ sensor ด้านล่างตรวจจับเส้นสีแดงได้

{

aForwardTime(5000);

bBackwardTime(5000);

}

if(color == YELLOW) // รถหยุด เมื่อ sensor ด้านล่างตรวจจับเส้นสีเหลืองได้

{

aStop();

bStop();

}

}

else

{

digitalWrite (led1,LOW); // เข้าสู้ฝ่ายรับ ( DEFENDER )

if(color == Number\_color)

{

if((hc.dist()<5)&&((digitalRead(ls)==HIGH)||(digitalRead(rs)==HIGH))&&(digitalRead(bs)==HIGH)) // ถอยหลัง ในกรณี sensor ด้านหน้าทำงาน และด้านข้างและด้านหลังไม่ทำงาน

{

aBackward(maxSpd);

bBackward(maxSpd);

}

if((hc.dist()>5)&&((digitalRead(ls)==HIGH)||(digitalRead(rs)==HIGH))&&(digitalRead(bs)==LOW)) // เดินหน้าเมื่อ sensor ด้านหลังทำงาน แต่ด้านหน้าและด้านข้างไม่ทำงาน

{

aForwardTime(3000);

bForwardTime(3000);

}

if((hc.dist()>5)&&(digitalRead(ls)==LOW)||(digitalRead(rs)==LOW)&&(digitalRead(bs)==HIGH)) // เบรค เมื่อ sensor ด้านข้างซ้ายหรือขวาทำงาน แต่ ด้านหน้าและหลังไม่ทำงาน

{

aBreakTime(1000);

bBreakTime(1000);

}

if((hc.dist()<5)&&(digitalRead(ls)==LOW)||(digitalRead(rs)==LOW)&&(digitalRead(bs)==HIGH)) // ถอยหลังช้าๆ เมื่อ sensor ด้านขวาหรือซ้ายทำงาน และ ด้านหน้าทำงาน และด้านหลังไม่ทำงาน

{

BackwardSlowly(maxSpd);

}

if((hc.dist()>5)&&(digitalRead(ls)==LOW)||(digitalRead(rs)==LOW)&&(digitalRead(bs)==LOW)) // เดินหน้าช้าๆ เมื่อ sensor ด้านซ้ายหรือขวาทำงาน และ ด้านหลังทำงาน และด้านหน้าไม่ทำงาน

{

ForwardSlowly(maxSpd);

}

}

if(color == BLACK) // กลับรถ เมื่อ sensor ด้านล่างตรวจจับเส้นสีดำได้

{

aForwardTime(5000);

bBackwardTime(5000);

}

}

delay(20);

}

void aStop()

{

digitalWrite(ia1, LOW); // motor stop

digitalWrite(ia2, LOW);

}

void aBreak()

{

digitalWrite(ia1, HIGH); // motor break

digitalWrite(ia2, HIGH);

}

void bStop()

{

digitalWrite(ib1, LOW); // motor stop

digitalWrite(ib2, LOW);

}

void bBreak()

{

digitalWrite(ib1, HIGH); // motor break

digitalWrite(ib2, HIGH);

}

void aForward(int speed)

{

digitalWrite(ia2, LOW);

analogWrite(ia1, speed);

}

void bForward(int speed)

{

digitalWrite(ib2, LOW);

analogWrite(ib1, speed);

}

void aBackward(int speed)

{

digitalWrite(ia1, LOW);

analogWrite(ia2, speed);

}

void bBackward(int speed)

{

digitalWrite(ib1, LOW);

analogWrite(ib2, speed);

}

void aBackwardTime(int time)

{

digitalWrite(ia1, LOW);

analogWrite(ia2, maxSpd);

delay (time);

}

void bBackwardTime(int time)

{

digitalWrite(ib1, LOW);

analogWrite(ib2, maxSpd);

delay (time);

}

void aForwardTime(int time)

{

digitalWrite(ia2, LOW);

analogWrite(ia1, maxSpd);

delay (time);

}

void bForwardTime(int time)

{

digitalWrite(ib2, LOW);

analogWrite(ib1, maxSpd);

delay (time);

}

void aBreakTime(int time)

{

digitalWrite(ia1, HIGH);

digitalWrite(ia2, HIGH);

delay (time);

}

void bBreakTime(int time)

{

digitalWrite(ib1, HIGH);

digitalWrite(ib2, HIGH);

delay (time);

}

void ForwardSlowly(int speed) //goforwardwithSlowspeed

{

digitalWrite(ia2, LOW);

analogWrite(ia1, speed\*0.3);

digitalWrite(ib2, LOW);

analogWrite(ib1, speed\*0.3);

}

void BackwardSlowly(int speed) //gobackwardwithSlowspeed

{

digitalWrite(ia1, LOW);

analogWrite(ia2, speed\*0.3);

digitalWrite(ib1, LOW);

analogWrite(ib2, speed\*0.3);

}

**เอกสารอ้างอิง**

1. https://sites.google.com/site/robotprograming

2. http://marcuscode.com/lang/cpp/program-struct

3. https://www.lcdtvthailand.com/webboard

4. https://robotsiam.blogspot.com/2016/10

5. https://www.myarduino.net/article/66

6. http://naringroup.blogspot.com/2016/03

7. http://www.arduino.codemobiles.com/product/88.

8. https://www.ioxhop.com/product/238

9. https://www.mindphp.com