



รายงานวิชา Pre-Project รหัสวิชา 01216747

จัดทำโดย

นางสาวปรียาพร สุทธิแพทย์ รหัสนักศึกษา 60010592

นางสาวศรวณีย์ อ่อนน้อม รหัสนักศึกษา 60010953

นายสหรัฐ สาแก้ว รหัสนักศึกษา 60011046

เสนอ

ผศ.ดร.อุดม จันทร์จรัสสุข

ภาคเรียนที่ 2 ปีการศึกษา 2562

คณะวิศวกรรมศาสตร์ ภาควิชาวิศวกรรมอุตสาหกรรม
สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

คำนำ

รายงานฉบับนี้เป็นส่วนหนึ่งของวิชา Pre-Project รหัสวิชา 01216747 ซึ่งประกอบไปด้วยส่วนของแนวคิดในการทำโปรเจกต์ กลยุทธ์ในเชิงรุกและรับ วงจรไฟฟ้า การออกแบบตัวรถ และ Programming Code ในการควบคุมการทำงาน

โดยโปรเจกต์นี้ใช้โปรแกรม Arduino ในการเขียนโปรแกรมควบคุมการทำงานของตัวรถ ใช้โปรแกรม Solidwork ในการออกแบบตัวรถ และใช้โปรแกรม EdrawMax ในการจำลองวงจรไฟฟ้า โดยทางคณะผู้จัดทำขอขอบคุณ ผศ.ดร.อุดม จันทร์จรัสสุข ในการให้คำปรึกษาและช่วยเหลือในการทำโปรเจกต์และรายงานฉบับนี้ให้สำเร็จไปได้ด้วยดี

คณะผู้จัดทำหวังเป็นอย่างยิ่งว่ารายงานฉบับนี้จะมีประโยชน์แก่ผู้ที่สนใจบ้างไม่มากนักน้อย และหากรายงานฉบับนี้มีข้อผิดพลาดประการใด ทางคณะผู้จัดทำขออภัยไว้ ณ ที่นี้

คณะผู้จัดทำ

สารบัญ

เรื่อง	หน้า
ปัญหาหรือโจทย์	1
ความรับผิดชอบของสมาชิกในทีมงาน	1
แนวคิดในการแก้ปัญหา	1
แนวคิดและเบื้องหลังที่จำเป็นในการทำโครงงาน	2
Circuit	9
Mechanical Design	12
Flow Chart	14
Programming Codes	16
เอกสารอ้างอิง	22

ปัญหาหรือโจทย์

การแข่งขันหุ่นยนต์มีลักษณะคล้ายกับการเล่น บอลลุนดำน หรือ เล่นเตย โดยแบ่งเป็นทีมรุกและทีมรับสลับกัน ในการแข่งแต่ละรอบ โดยทีมหนึ่งจะประกอบด้วยหุ่นยนต์ 7 ตัว ฝ่ายทีมรุกจะต้องวิ่งไปหาฝั่งตรงข้าม จนผ่านเส้นแดง แล้วกลับมาอย่างปลอดภัย(ผ่านเส้นสีเหลือง) โดยที่ไม่ถูกทีมรับจับได้ ก็จะเป็นฝ่ายชนะในการแข่งขันรอบนั้น หุ่นยนต์ที่ถูกจับได้จะถูกตัดออกจากการแข่งขันในรอบนั้น ส่วนทีมรับ จะสามารถวิ่งสกัดกั้นฝ่ายตรงข้ามในพื้นที่ป้องกันเท่านั้น ถ้าวิ่งออกนอกพื้นที่ก็就会被ตัดออกจากการแข่งขันในรอบนั้นเช่นกัน ถ้าไม่มีหุ่นยนต์ตัวไหนสามารถผ่านด่านได้ ทีมรับจะเป็นฝ่ายชนะ การแข่งขันของแต่ละรอบจะยุติเมื่อทีมรุกสามารถผ่านด่านได้สำเร็จ หรือเมื่อทีมใดทีมหนึ่งไม่เหลือผู้เล่น

ความรับผิดชอบของสมาชิกในทีมงาน

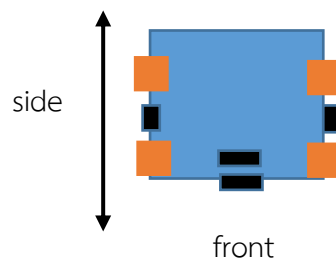
นางสาวปรียาพร สุทธิแพทย์ รหัสนักศึกษา 60010592 Mechanical Design

นางสาวศรวณีย์ อ่อนน้อม รหัสนักศึกษา 60010953 Circuit

นายสหัสรัฐ साแก้ว รหัสนักศึกษา 60011046 Programming Codes

แนวคิดในการแก้ปัญหา

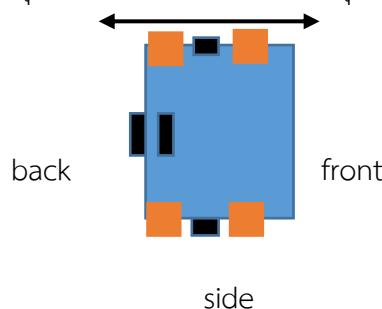
กลยุทธ์รุก



1. เดินหน้า เมื่อ sensor ด้านหน้า ด้านข้าง และด้านล่าง ไม่ทำงาน (เมื่อไม่มีรถอยู่ด้านหน้าในระยะ 10 ซม. และไม่มีรถมาด้านข้าง)
2. break เมื่อ sensor ด้านหน้าทำงาน แต่ด้านข้างและด้านล่างไม่ทำงาน (เมื่อมีรถอยู่ด้านหน้าในระยะ 10 ซม. แต่ไม่มีรถมาด้านข้าง)
3. เดินถอยหลัง 3วินาที เมื่อ sensor ด้านหน้าและด้านข้างซ้ายหรือขวาทำงาน แต่ด้านล่างไม่ทำงาน (เมื่อมีรถอยู่ด้านหน้าในระยะ 10 ซม. และมีรถมาด้านข้าง)
4. กลับรถ เมื่อ sensor ด้านล่างตรวจจับเส้นสีแดงได้
5. รถหยุด เมื่อ sensor ด้านล่างตรวจจับเส้นสีเหลืองได้

กลยุทธ์รับ

กำหนดให้ด้านหลังของรถในกลยุทธ์รุก เป็นด้านหน้าของรถในกลยุทธ์รับ



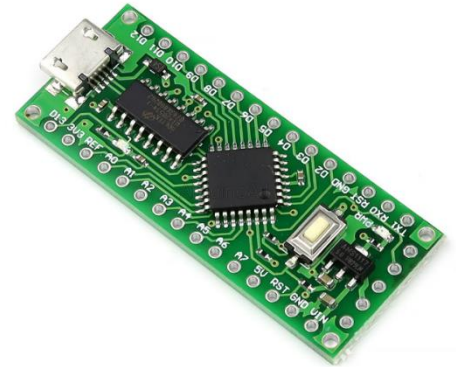
1. เดินหน้าเมื่อ sensor ด้านหลัง ด้านข้าง และด้านล่าง ไม่ทำงาน (เมื่อไม่มีรถอยู่ด้านหลังในระยะ 5 ซม. และไม่มีรถมาด้านข้าง)
2. ถอยหลังเมื่อ sensor ด้านหลังทำงาน แต่ด้านข้างและล่างไม่ทำงาน (เมื่อมีรถอยู่ด้านหลังในระยะ 5 ซม. และไม่มีรถมาด้านข้าง)
3. break เมื่อ sensor ด้านข้างซ้ายหรือขวาทำงาน แต่ล่างไม่ทำงาน (เมื่อมีรถมาด้านข้าง)
4. กลับรถ เมื่อ sensor ด้านล่างตรวจจับเส้นสีดำได้

หมายเหตุ: sensor ด้านล่าง คือ sensor ตรวจจับสี

แนวคิดและเบื้องหลังที่จำเป็นในการทำโครงงาน

Arduino board (LGT8F328P)

Feature	LGT8F328P
DAC output	Yes (D4)
ADC	12bit (9 channel)
ADC Sampling rate Max.	500KSPS
Analog Comparator	2
unique ID	Yes
Internal reference resoltuion	$\pm 0.5\%$
PWM dead zone control	Yes
High current push - pull PWM	Yes
Computing Accelerator (DSC)	Yes
Stacking expansion system	Yes
Speed	32Mhz
OUTPUT	27 pin
INPUT	30 pin

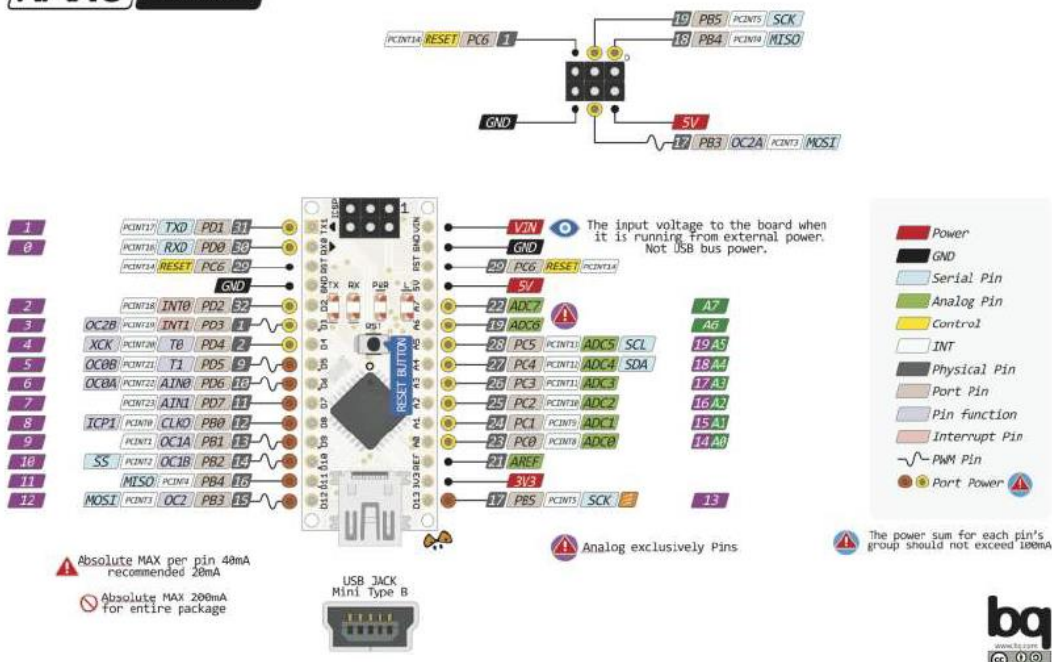


I/O port interface

- Port คืออุปกรณ์ที่ทำหน้าที่เชื่อมต่อคอมพิวเตอร์หรือ microcontroller กับอุปกรณ์ภายนอก
- Input port เป็นช่องทางในการรับสัญญาณจากอุปกรณ์ภายนอก
- Output port เป็นช่องทางในการส่งสัญญาณไปให้อุปกรณ์ภายนอก
- การเชื่อมต่อกับอุปกรณ์ภายนอกเรียกว่า I/O interface
- อุปกรณ์อาจจะมีทั้ง Input/ Output ports ในอุปกรณ์เดียวกัน
- Port ส่วนใหญ่สามารถถูกโปรแกรมให้เป็น Input หรือ Output ได้
- GPIO = General Purpose Input/ Output

I/O pin mapping (Arduino Nano)

NANO PINOUT



นั่นจะทำให้ cell เสียหายถาวร ไม่สามารถนำกลับมาใช้ใหม่ได้ (เครื่องชาร์จจะไม่ยอมชาร์ตหากแรงดันใน Cell ต่ำกว่าที่กำหนด) ฉะนั้น วงจรป้องกัน Protected PCB จึงเป็นสิ่งสำคัญอย่างมากในการใช้งานร่วมกับอุปกรณ์ทั่วไป โดยผู้ใช้ไม่ต้องกังวลว่าจะใช้งานจากแบตเตอรี่หมดจน cell พัง อีกทั้งหากมีการดึงกระแสเกินกำหนด หรือมีการใช้แรงดันชาร์ตเกินพิกัด วงจรจะทำการตัดการทำงานอัตโนมัติ ส่วน 18650 แบบไม่มีวงจร นิยมใช้กับอุปกรณ์อิเล็กทรอนิกส์ที่ถูกออกแบบมาเฉพาะซึ่งมีวงจรควบคุมอยู่ภายนอกแล้ว เช่น วงจร BMS หรือ PCM ซึ่งมีหลักการทำงานคล้ายกัน จึงไม่จำเป็นต้องมีวงจรป้องกันในตัว cell ทำให้ประหยัดต้นทุนในการผลิต มักพบเห็นใน Battery Pack

DC Geared-Motors

มอเตอร์ไฟฟ้ากระแสตรง หรือดีซีมอเตอร์ (DC Motor) เป็นอุปกรณ์ที่แปลงพลังงานไฟฟ้าให้เป็นพลังงานกล โครงสร้างภายใน DC motor ประกอบด้วยส่วนหลักๆ สองส่วน ได้แก่ แม่เหล็กถาวรและแกนขดลวด นอกจากนี้ยังมีแปรงถ่าน (Brush) ซึ่งเป็นส่วนเชื่อมต่อเพื่อรับพลังงานไฟฟ้าภายนอกไปยังขดลวดของมอเตอร์ เมื่อขดลวดได้รับไฟฟ้ากระแสตรง จะมีถูกเหนี่ยวนำให้เกิดสนามแม่เหล็กรอบ ๆ รอบขดลวด



ลักษณะภายนอกของมอเตอร์ไฟฟ้ากระแสตรง สังเกตได้จากสายของมอเตอร์จะมีเพียงสองเส้น เมื่อเราต่อมอเตอร์กับแหล่งจ่ายไฟกระแสตรงภายนอก เช่น ถ่านหรือแบตเตอรี่ มอเตอร์จะหมุน หากเราต่อไฟสลับขั้ว มอเตอร์จะหมุนในทิศตรงกันข้าม หากต้องการลดความเร็วของมอเตอร์ เราเพียงปรับแรงดันของแหล่งจ่ายไฟ เนื่องจากมอเตอร์ไฟฟ้ากระแสตรงมีราคาถูกและใช้งานง่าย เราจึงพบการนำมอเตอร์ไฟฟ้ากระแสตรง มาใช้งานได้หลากหลาย เช่น ของเล่นขนาดเล็ก จักรยานไฟฟ้า แขนกลหุ่นยนต์และเครื่องจักรต่าง ๆ ในโรงงานอุตสาหกรรม เนื่องจาก DC motor ต้องใช้กระแสสูงในการทำงาน ดังนั้น Microcontroller จะไม่สามารถเชื่อมต่อโดยตรง กับ DC Motor ได้ จึงต้องมีชุดขับกระแส

การทำงานของ DC Geared-Motors

in1	in2	Motor Operation
0	1	Forward
1	0	Reward
0	0	Stop
1	1	Break

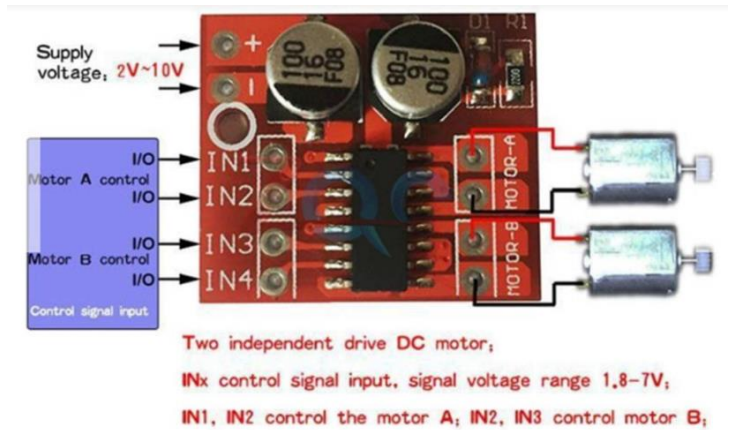
H-bridge Driver

ลักษณะ: Supply voltage: 2-10V

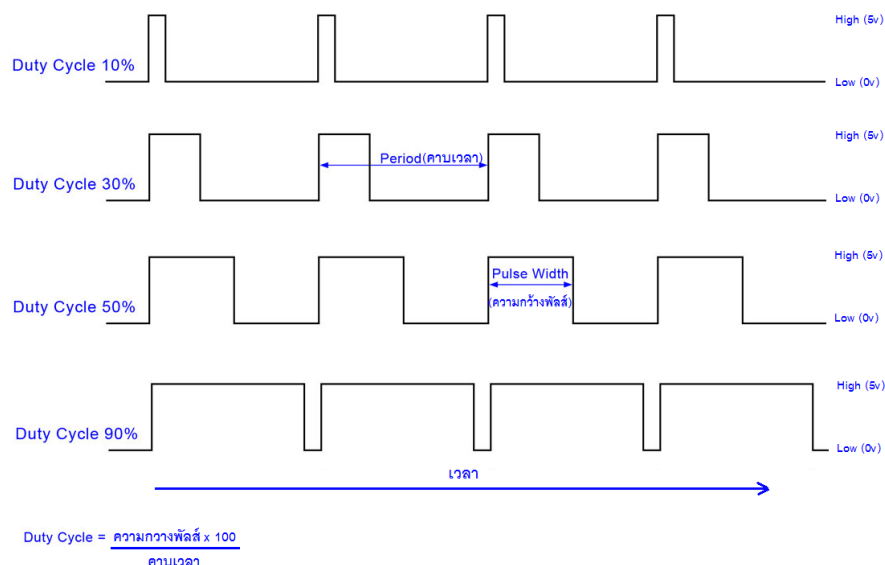
Signal input voltage: 1.8-7V

Max output current: $1.5A \times 2$

Control signal: PWM



PWM หมายถึง Pulse Width Modulation เป็นเทคนิคที่ Arduino ใช้ในการควบคุมวงจรและเขียนค่าแบบ Analog ด้วยพอร์ต Digital โดยปกติแล้วพอร์ต Digital สามารถมีได้แค่ 2 สถานะ คือ HIGH (5 V) กับ LOW (0 V) เท่านั้น จึงทำให้สร้างค่าสัญญาณ logic ได้เพียงเปิดหรือปิด (1 หรือ 0, มีไฟหรือไม่มีไฟ) ซึ่งการใช้เทคนิค PWM นั้น จะเป็นการทำให้พอร์ต Digital สามารถเขียนค่าได้มากกว่า HIGH หรือ LOW โดยทำให้สามารถเขียนค่าเป็นแบบ Analog ได้ (อาจเป็น 0-255 หรือ 0-1023) โดยวิธีการนั้น จะใช้การปรับสถานะของสัญญาณ logic HIGH / LOW สลับกันไปมาด้วยคาบเวลาหนึ่งๆ โดยค่าที่ได้นั้นจะขึ้นอยู่กับสัดส่วนเวลาของสัญญาณในช่วงเวลาที่มีสถานะเป็น HIGH กับช่วงเวลาที่ เป็น LOW โดย ช่วงเวลาทั้งหมดที่สัญญาณมีสถานะเป็น HIGH นั้นจะเรียกว่าเป็น "ความกว้าง Pulse (Pulse Width)" โดยสัญญาณพัลส์ เมื่อเทียบ % ของช่วงเวลาที่ เป็น HIGH (หรือก็คือ % ของ Pulse Width)กับ % ของคาบเวลา (Period) ของพัลส์ลูกนั้น ๆ จะเรียกว่า Duty Cycle

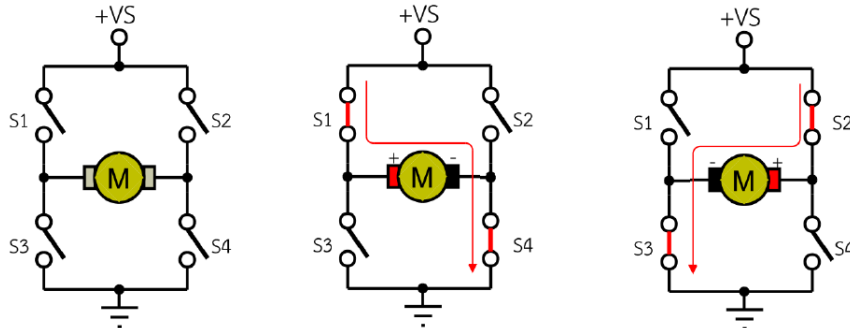


วงจรขับมอเตอร์ไฟฟ้ากระแสตรง

การขับมอเตอร์ไฟฟ้ากระแสตรง ทำได้ 2 ลักษณะคือ การควบคุมทิศทางหมุน และการควบคุมความเร็วในการหมุน ทั้งนี้ขึ้นอยู่กับลักษณะของวงจรขับมอเตอร์ด้วย

วงจรขับเคลื่อนมอเตอร์ไฟฟ้ากระแสตรงด้วยสวิตช์

เป็นวงจรขับเคลื่อนมอเตอร์ไฟฟ้ากระแสตรงแบบพื้นฐาน สำหรับควบคุมทิศทางการหมุนของมอเตอร์โดยใช้สวิตช์ควบคุม 4 ตัวเรียกว่าวงจรขับเคลื่อนมอเตอร์ไฟฟ้ากระแสตรงแบบ H-Bridge เนื่องจากลักษณะของวงจรคล้ายกับตัวอักษร H ในภาษาอังกฤษ และมีการใช้อุปกรณ์ควบคุม 4 ตัว ลักษณะวงจรและการทำงานแสดงดังรูป

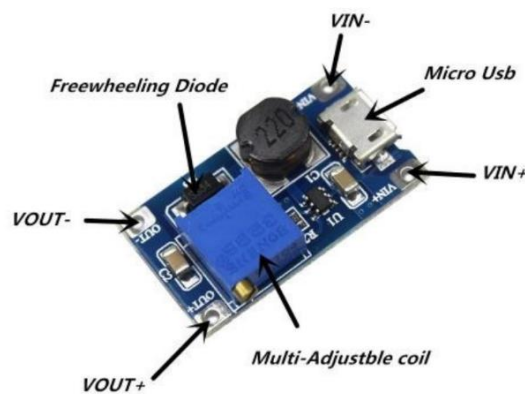


(ก) มอเตอร์ไม่หมุน (ข) มอเตอร์หมุนตามเข็มนาฬิกา (ค) มอเตอร์หมุนทวนเข็มนาฬิกา

จากรูป (ก) มอเตอร์ไม่ทำงานเนื่องจากไม่มีการต่อวงจรไฟฟ้าให้กับมอเตอร์ ส่วนรูป (ข) มอเตอร์หมุนตามเข็มนาฬิกา เนื่องจากเมื่อต่อวงจรสวิตช์ S1 กับ S4 กระแสจะไหลผ่านมอเตอร์จากทางด้านซ้ายมือไปด้านขวามือครบวงจร และรูป (ค) มอเตอร์หมุนทวนเข็มนาฬิกา เนื่องจากเมื่อต่อวงจรสวิตช์ S2 กับ S3 กระแสจะไหลผ่านมอเตอร์จากทางด้านขวามือไปด้านซ้ายมือครบวงจร ดังนั้นสามารถควบคุมทิศทางการหมุนของมอเตอร์ไฟฟ้ากระแสตรงด้วยการกลับขั้วของแรงดันไฟฟ้าที่จ่ายให้กับมอเตอร์

DC/DC Step-up Converter

เป็นวงจรที่ทำหน้าที่เพิ่มแรงดันไฟฟ้า ถ่าน Li-ion ให้แรงดันที่ 3.7-4.2 v ซึ่งไม่เพียงพอกับความต้องการของบอร์ด Arduino แก้ปัญหาโดยใช้วงจร step-up



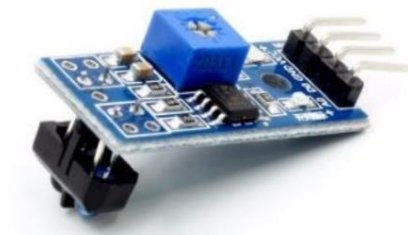
ข้อควรระวัง: เมื่อแรงดันเพิ่ม กระแสที่จ่ายได้จะลดลง



ถ้าประสิทธิภาพอยู่ที่ 80% กระแส output จะเหลือ 0.4A

TCRT5000 Infrared Reflective sensor

เป็นโมดูลตรวจจับวัตถุระยะใกล้ มีราคาถูก ขนาดเล็ก สะดวกในการนำไปใช้ติดตั้งกับงานจำพวก หุ่นยนต์, Smart car, หุ่นยนต์หลบสิ่งกีดขวาง เป็นต้น โดยการทำงานของตัวโมดูลนี้ เริ่มต้นโดยให้หลอด Infrared LED ทำการส่งสัญญาณ เป็นแสงอินฟราเรดออกไปตกกระทบกับวัตถุที่ตรวจพบในระยะ และทำการสะท้อนกลับมายังตัวหลอดโฟโตไดโอดที่ทำหน้าที่รับแสงอินฟราเรด โดยส่วนมากตัวโมดูลจะให้ค่า output ออกมาเป็น Digital signal แต่สำหรับบางโมดูลอาจจะรองรับ output แบบ Analog signal ด้วย ส่วนตัว R ปรับค่านั้นใช้ในการปรับความไวต่อการตรวจจับแสงอินฟราเรด ซึ่งจะส่งผลต่อระยะในการตรวจพบวัตถุของตัวเซนเซอร์



IR Infrared Obstacle

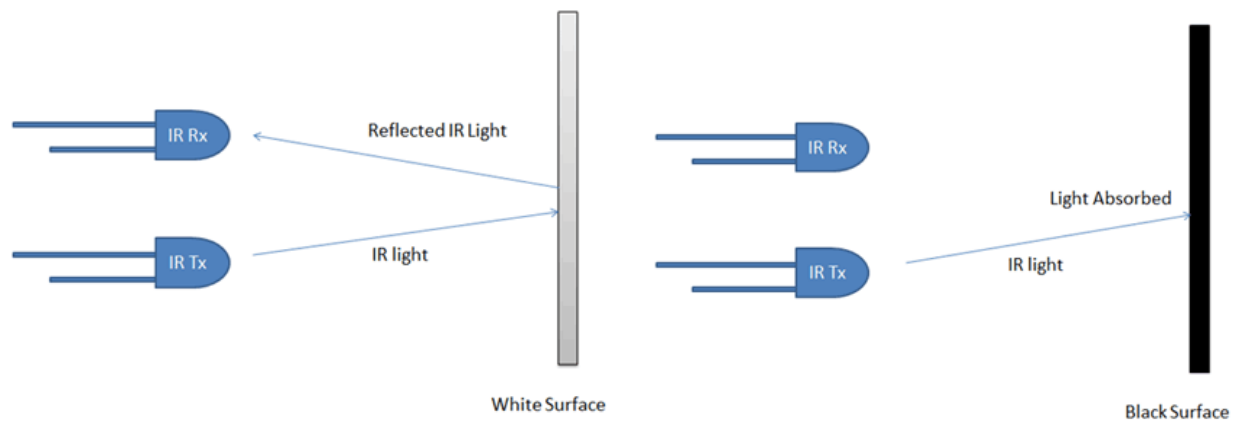
Avoidance Sensor

โมดูลเซ็นเซอร์แสงสำหรับตรวจจับวัตถุกีดขวาง IR Infrared Obstacle Avoidance Sensor Module โดยโมดูลนี้ จะมีตัวรับและตัวส่ง infrared ในตัว ตัวสัญญาณ(สีขาว) infrared จะส่งสัญญาณออกมา และเมื่อมีวัตถุมาบัง คลื่นสัญญาณ infrared ที่ถูกส่งออกมาจะสะท้อนกลับเข้าไปในตัวรับสัญญาณ (สีดำ) สามารถนำมาใช้ตรวจจับวัตถุที่อยู่ตรงหน้าได้ และสามารถปรับความไว ระยะการตรวจจับ ใกล้หรือไกลได้

ภายในตัวเซ็นเซอร์แบบนี้จะมีตัวส่ง Emitter และ ตัวรับ Receiver ติดตั้งภายในตัวเดียวกัน ทำให้ไม่จำเป็นต้องเดินสายไฟทั้งสองฝั่ง เหมือนแบบ Opposed Mode ทำให้การติดตั้งใช้งานได้ง่ายกว่า แต่อย่างไรก็ตาม จำเป็นต้องติดตั้งตัวแผ่นสะท้อนหรือ Reflector ไว้ตรงข้ามกับตัวเซ็นเซอร์เอง โดยโฟโต้เซ็นเซอร์แบบนี้ใช้แผ่นสะท้อนแบบนี้จะเหมาะสำหรับชิ้นงานที่มีลักษณะทึบแสงไม่เป็นมันวาว เนื่องจากอาจทำให้ตัวเซ็นเซอร์เข้าใจผิดว่าเป็นตัวแผ่นสะท้อน และทำให้ทำงานผิดพลาดได้

เซ็นเซอร์แบบนี้จะมีช่วงในการทำงาน หรือ ระยะในการตรวจจับจะได้ใกล้กว่าแบบ Opposed mode ซึ่งในสภาวะการทำงานปกติตัวรับ Receiver จะสามารถรับสัญญาณแสงจากตัวส่ง Emitter ได้ตลอดเวลาเนื่องจากลำแสงจะสะท้อนกับแผ่นสะท้อน Reflector อยู่ตลอดเวลาจะแสดงค่า เป็น 0



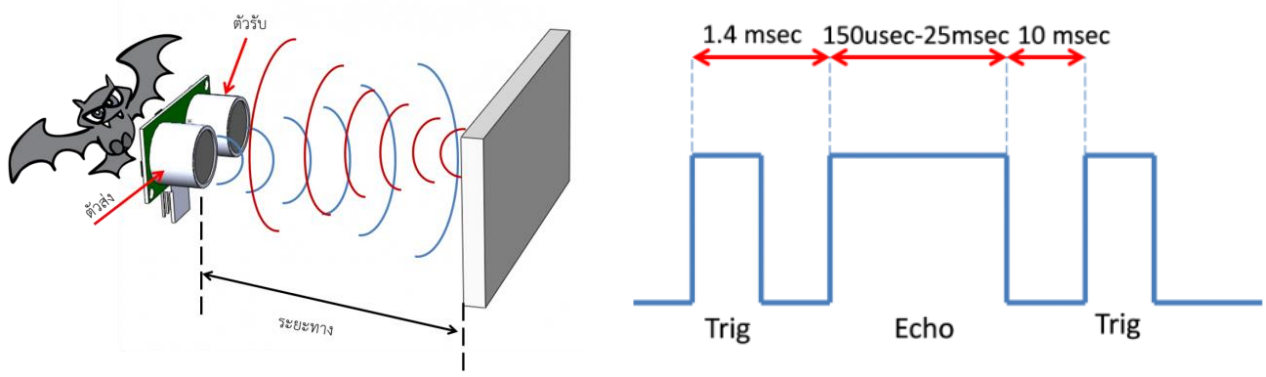


Ultrasonic Sensor

HC-SR04 เป็นเซนเซอร์โมดูลสำหรับตรวจจับวัตถุและวัดระยะทางแบบไม่สัมผัส โดยใช้คลื่นอัลตราโซนิก ซึ่งเป็นคลื่นเสียงความถี่สูงเกินกว่าการได้ยินของมนุษย์ วัดระยะได้ตั้งแต่ 2 – 400 เซนติเมตร หรือ 1 – 156 นิ้ว สามารถต่อใช้งานกับไมโครคอนโทรลเลอร์ได้ง่าย ใช้พลังงานต่ำ เหมาะกับการนำไปประยุกต์ใช้งานด้านระบบควบคุมอัตโนมัติ หรืองานด้านหุ่นยนต์ หลักการทำงาน จะเหมือนกับการตรวจจับวัตถุด้วยเสียงของค้างคาว โดยจะประกอบไปด้วยตัว รับ-ส่ง อัลตราโซนิก ตัวส่งจะส่งคลื่นความถี่ 40 kHz ออกไปในอากาศด้วยความเร็วประมาณ 346 เมตรต่อวินาที และตัวรับจะคอยรับสัญญาณที่สะท้อนกลับจากวัตถุ เมื่อทราบความเร็วในการเคลื่อนที่ของคลื่น, เวลาที่ใช้ในการเดินทางไป-กลับ (t) ก็จะสามารถคำนวณหาระยะห่างของวัตถุ (S) ได้จาก

$$S = 346 \times 0.5t$$

เพื่อให้การคำนวณหาระยะเป็นไปด้วยความง่าย โมดูลเซนเซอร์นี้จึงได้ประมวลผลให้เรียบร้อยแล้ว และส่งผลลัพธ์ของการคำนวณเป็นสัญญาณพัลส์ที่มีความกว้างสัมพันธ์กับระยะทางที่วัดได้

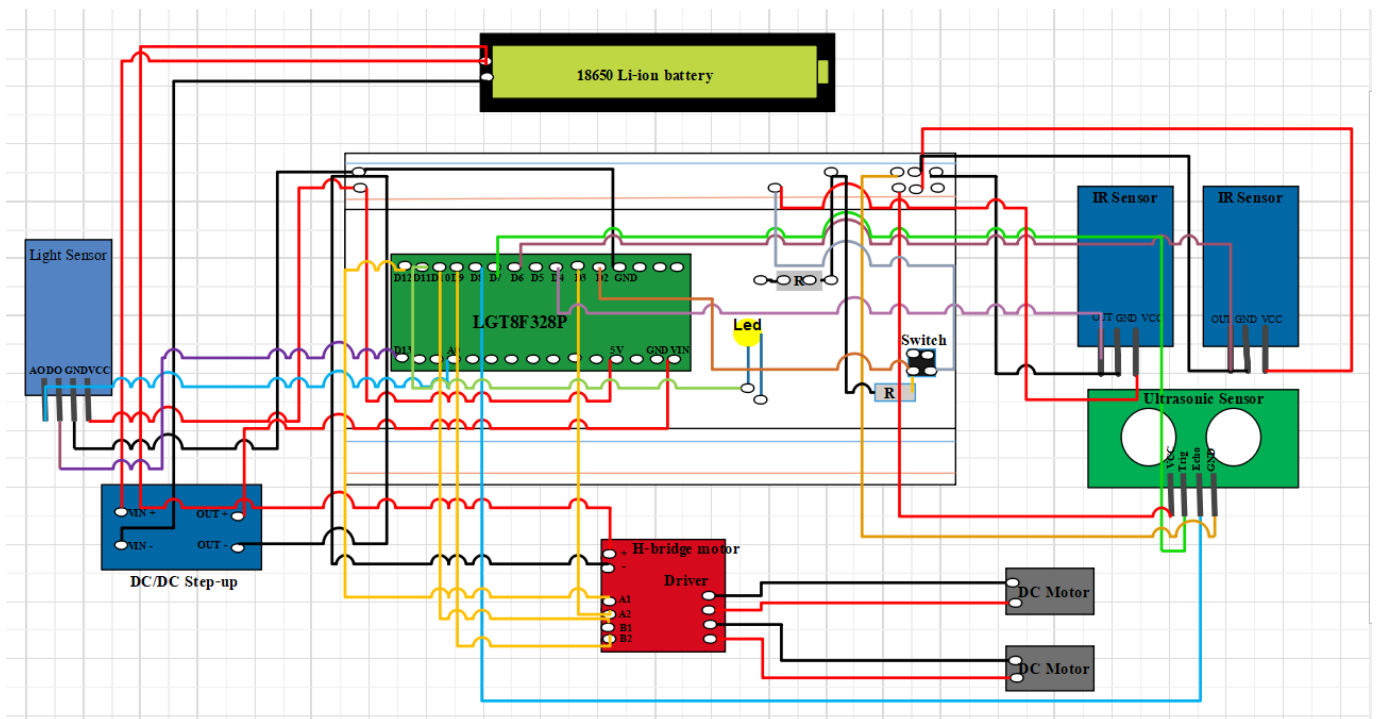


ตามคุณลักษณะของเซนเซอร์ จะต้องสร้างสัญญาณพัลส์ความกว้างไม่น้อยกว่า 10 msec ป้อนเข้าที่ขา Trig หลังจากนั้นอีกประมาณ 1.4 msec จึงจะเริ่มมีสัญญาณพัลส์เกิดขึ้นที่ขา Echo มีความกว้างของสัญญาณตั้งแต่ 150 usec – 25 msec ซึ่งถ้าหากกว้างกว่านี้จะถือว่าตรวจไม่พบวัตถุ หลังจากนั้นควรหน่วงเวลาออกไปอีก 10 mS จึงจะส่งสัญญาณ Trig ออกไปอีกรอบ

Circuit

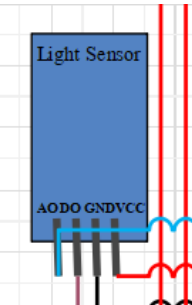
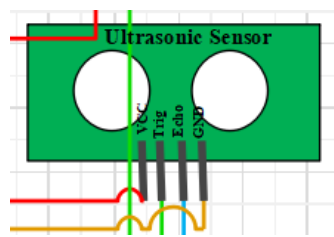
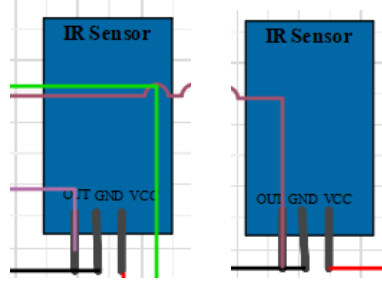
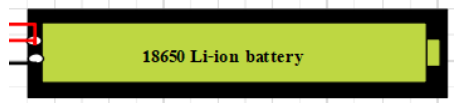
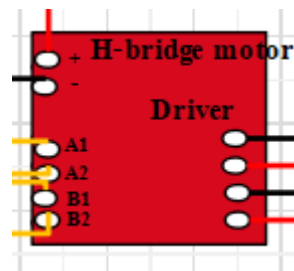
อุปกรณ์

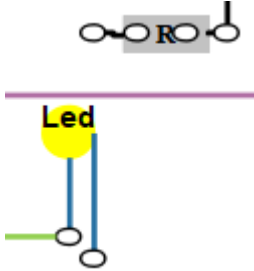
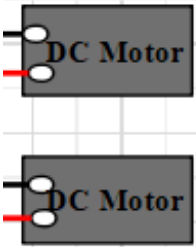
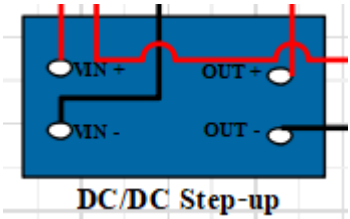
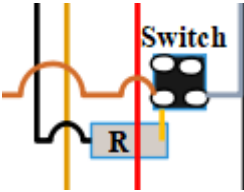
1. Arduino board (LGT8F328P) จำนวน 1 ชิ้น
2. 18650 Li-ion battery, Battery case, Li-ion Charging จำนวนอย่างละ 1 ชิ้น
3. DC Geared-Motors จำนวน 2 ก้อน
4. H-bridge Driver จำนวน 1 ชิ้น
5. Breadboard จำนวน 1 ชิ้น
6. DC/DC Step-up Converter จำนวน 1 ชิ้น
7. TCRT5000 Infrared Reflective sensor จำนวน 1 ชิ้น
8. IR Infrared Obstacle Avoidance Sensor จำนวน 2 ชิ้น
9. Ultrasonic Sensor จำนวน 1 ชิ้น
10. LED และตัวต้านทาน จำนวนอย่างละ 1 ชิ้น
11. Switch และตัวต้านทาน จำนวนอย่างละ 1 ชิ้น



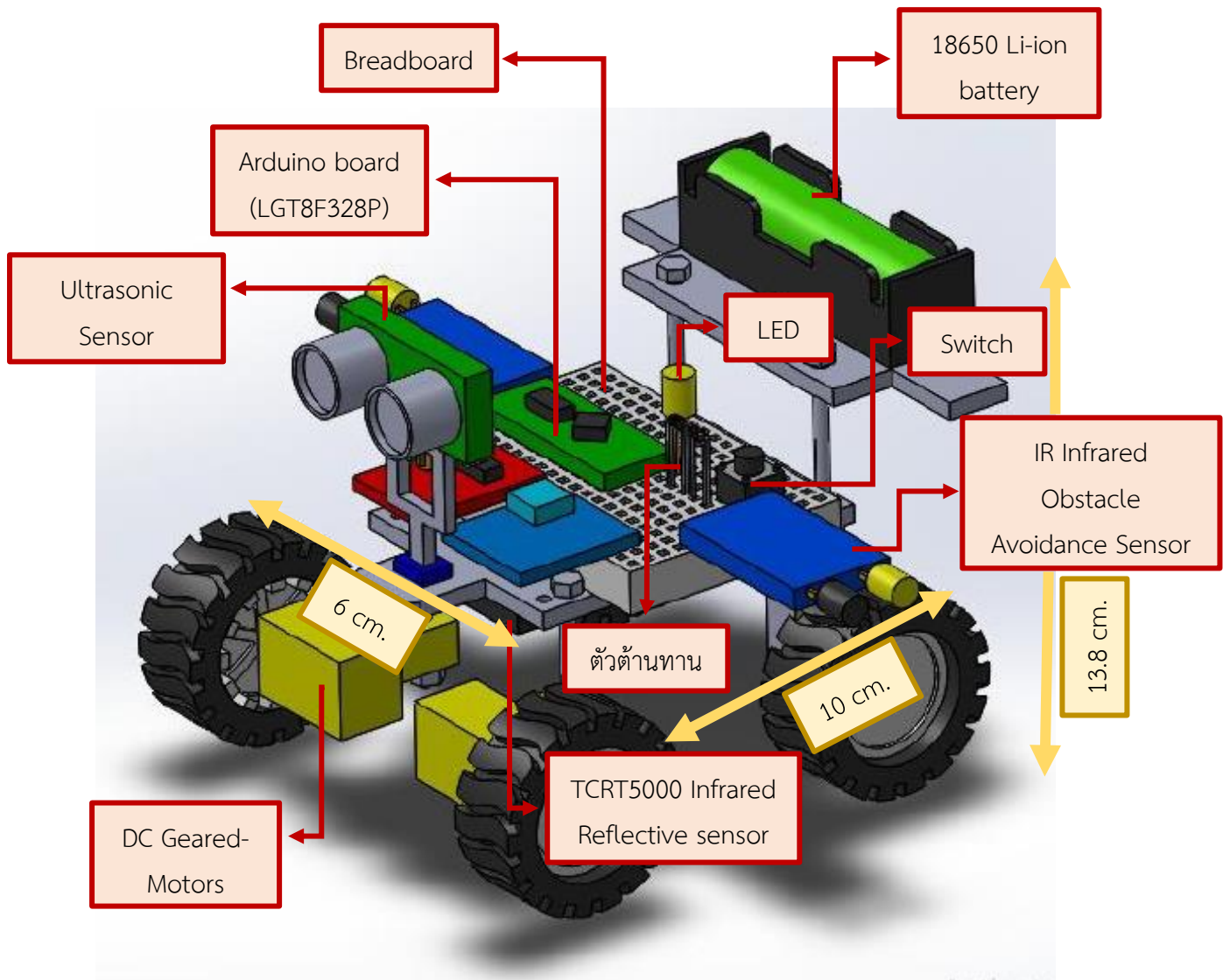
รูปแสดงวงจรโดยรวม

โดยมีรายละเอียดการต่อวงจรดังนี้

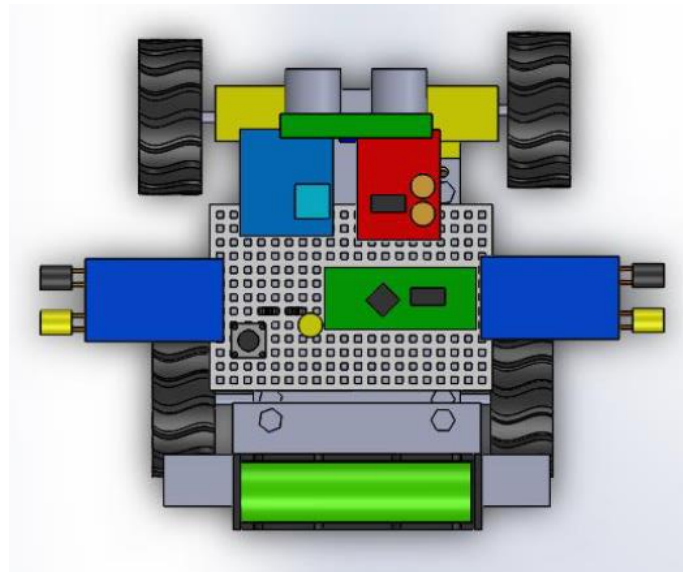
อุปกรณ์	รูปภาพ	การต่อ
TCRT5000 Infrared Reflective sensor		VCC: 5V GND: Ground D0: Digital output (0/1): D13 A0: Analog output: A0
Ultrasonic Sensor		VCC: 5V Trig: Digital output (0/1): D7 Echo: Digital Input (0/1): D8 GND: Ground
IR Infrared Obstacle Avoidance Sensor		OUT: Digital Input (0/1): D4, D6 GND: Ground VCC: 5V
18650 Li-ion battery, Battery case, Li-ion Charging		เข้าบวก: VIN+ ของ DC/DC Step-up Converter และบวกของ H-bridge Driver เข้าลบ: VIN- ของ DC/DC Step-up Converter
H-bridge Driver		บวก: เข้าบวกของ 18650 Li-ion battery ลบ: Ground A1: D12 A2: D3 B1: D10 B2: D9 4ช่องด้านขวา: Motor 1 and 2

LED และตัวต้านทาน		<p>ขาบวกของ LED: D11</p> <p>ขาลบของ LED: ขาบวกตัวต้านทาน</p> <p>ขาบวกตัวต้านทาน: ขาลบของ LED</p> <p>ขาลบตัวต้านทาน: Ground</p>
DC Geared-Motors		<p>ต่อเข้ากับ H-bridge Driver</p>
DC/DC Step-up Converter		<p>Vin: V battery (-) Ground</p> <p>Vin+: V battery (+)</p> <p>Out-: Ground</p> <p>Out+: VIN (Arduino)</p>
Switchและตัวต้านทาน		<p>ขาต้านซ้ายของswitch: D2และตัวต้านทาน</p> <p>ขาต้านขวาของSwitch: 5V</p> <p>ขาบวกของตัวต้านทาน: ขาต้านซ้ายของswitch</p> <p>ขาลบของตัวต้านทาน: Ground</p>

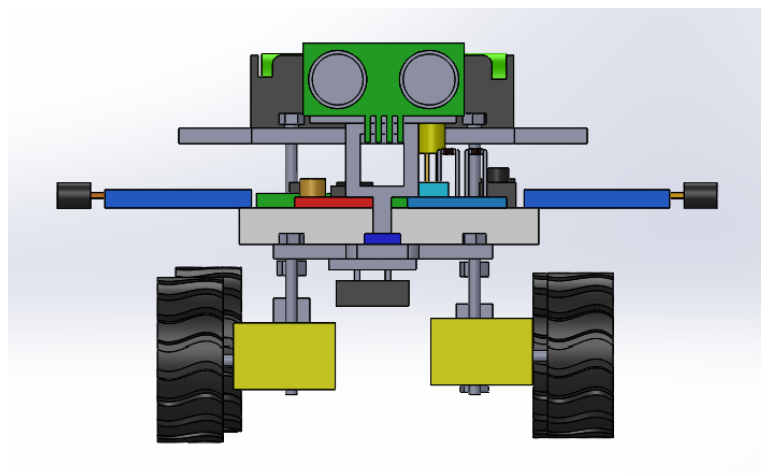
Mechanical Design



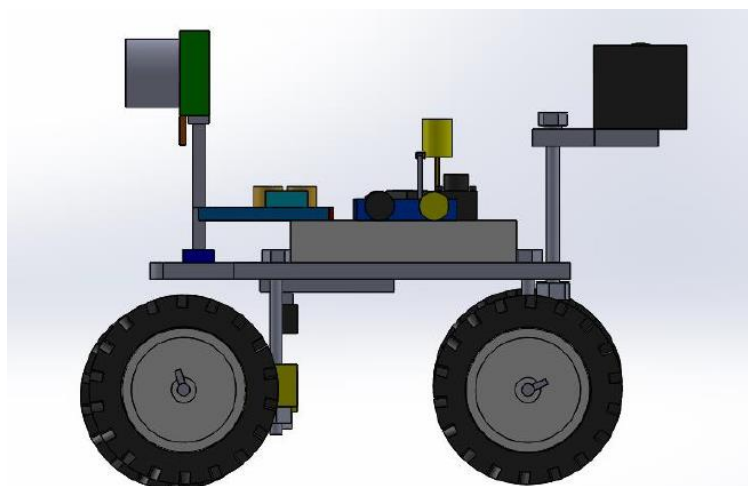
Isometric



Top

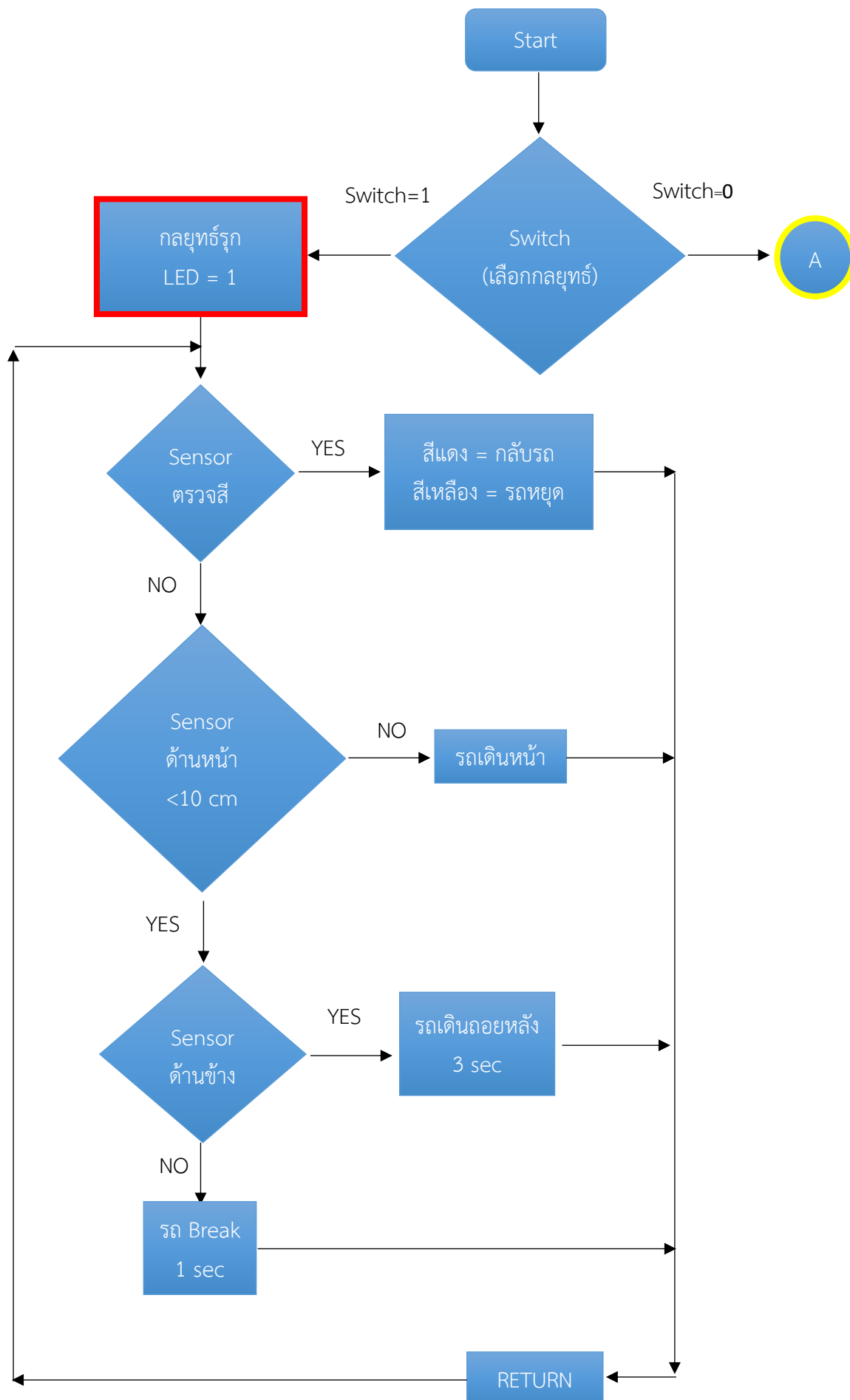


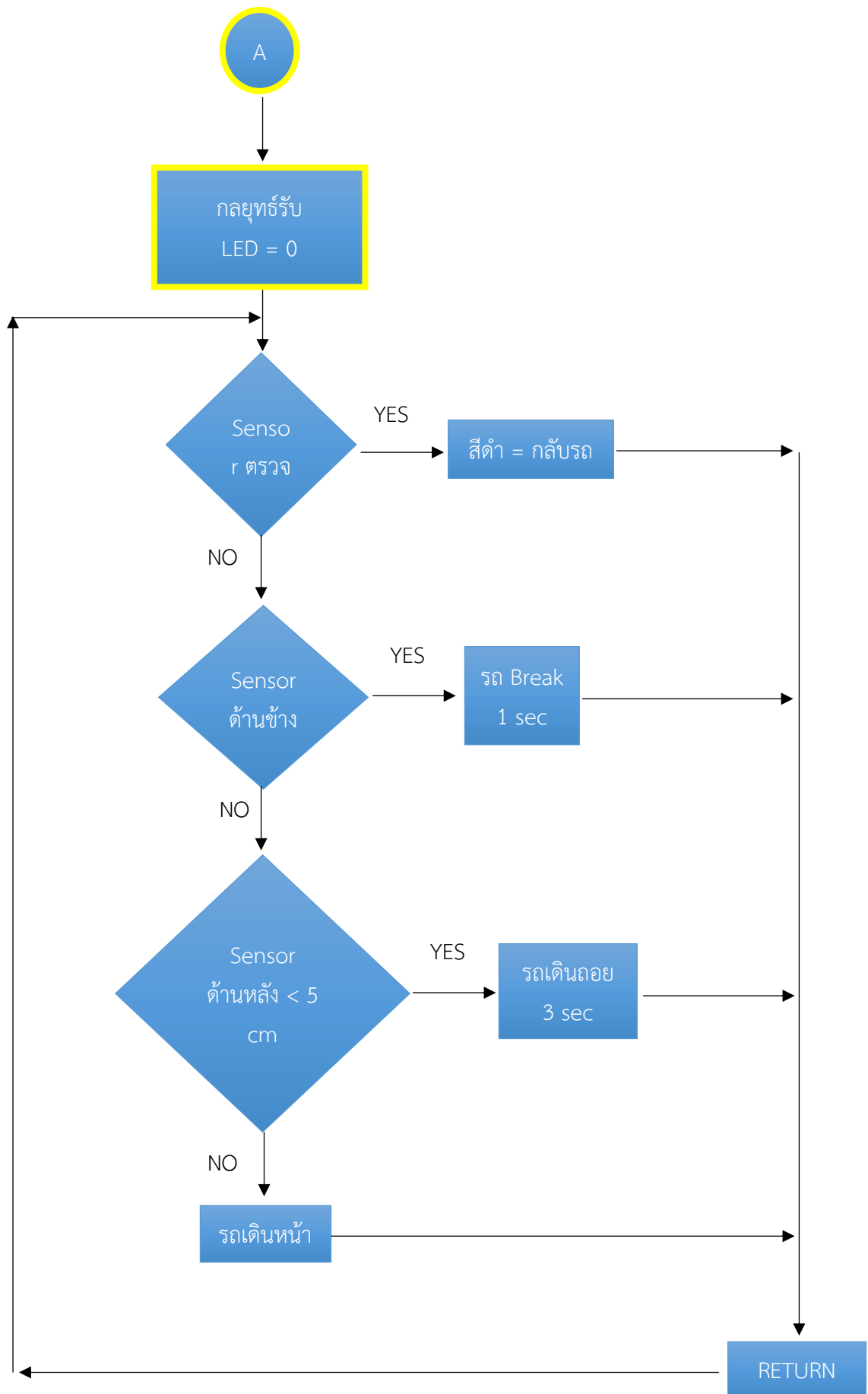
Front



Side

Flowchart





หมายเหตุ: เมื่อใช้กลยุทธิ์รับ ด้านหน้าของรถในกลยุทธิ์รูกจะเป็นด้านหลังของกลยุทธิ์รับ

Programming Codes

```
#define ia1 D12 //motor a เดินหน้า
#define ia2 D3 //motor a ถอยหลัง
#define ib1 D10 //motor b เดินหน้า
#define ib2 D9 //motor b ถอยหลัง
#define ls D4 //sensorซ้าย
#define rs D6 //sensorขวา
#define RED 1
#define YELLOW 2
#define BLACK 3
#define NOCOLOR 0
#define maxSpd 255 // motor max speed
#include <HCSR04.h>
HCSR04 hc(D7,D8); //initialisation class HCSR04 (trig,echo);
int analogPin = A5; //ประกาศตัวแปร ให้ analogPin แทนขา analog ขาที่5
int led1 = D11;
int buttonPin = D2;
int val=0;
int old_val=0;
int state=0;
int color=0;
int getColor() {
int NO_color = analogRead(analogPin); //อ่านค่าสัญญาณ analog ขา5 ที่ต่อกับ TCRT5000
if ((NO_color>2200)&&(NO_color<2500)) //สีแดง
return RED;
else if ((NO_color>1900)&&(NO_color<2200)) //สีเหลือง
return YELLOW;
else if ((NO_color>3600)&&(NO_color<3900)) //สีดำ
return BLACK;
else //ไม่พบสี
return NOCOLOR;
}
void setup() {
// put your setup code here, to run once:
pinMode(ls, INPUT);
pinMode(rs, INPUT);
pinMode(buttonPin, INPUT);
```

```

pinMode(led1, OUTPUT);
pinMode(ia1, OUTPUT);
pinMode(ia2, OUTPUT);
pinMode(ib1, OUTPUT);
pinMode(ib2, OUTPUT);
Serial.begin(115200);
}

void loop()
{
color = getColor();
val = digitalRead(buttonPin);
if( (val==HIGH) && (old_val==LOW))
{
state=!state;
}
old_val=val;
if (state==1) //เมื่อกดสวิตช์ 1 ครั้ง ใช้กลยุทธ์รุก
{
digitalWrite(led1, HIGH);
if((hc.dist())>10)&&((digitalRead(ls)==HIGH)||((digitalRead(rs)==HIGH))&&(color == NOCOLOR)) //
เดินหน้าเมื่อ sensor ด้านหน้า ด้านข้าง และด้านล่างไม่ทำงาน
{
aForward(maxSpd);
bForward(maxSpd);
}
if((hc.dist())<10)&&((digitalRead(ls)==HIGH)||((digitalRead(rs)==HIGH))&&(color == NOCOLOR)) //
break เมื่อ sensor ด้านหน้าทำงาน แต่ด้านข้างและด้านล่างไม่ทำงาน
{
aBreakTime(1000);
bBreakTime(1000);
}
if((hc.dist())<10)&&((digitalRead(ls)==LOW)||((digitalRead(rs)==LOW))&&(color == NOCOLOR)) //
เดินถอยหลัง 3วิ เมื่อ sensor ด้านหน้าและข้างซ้ายขวาทำงาน แต่ด้านล่างไม่ทำงาน
{
aRewardTime(3000);
bRewardTime(3000);
}
}

```

```

if(color == RED) // เดินกลับรถ เมื่อ sensor ด้านล่างตรวจจับเส้นสีแดงได้
{
aForwardTime(5000);
bRewardTime(5000);
}
if(color == YELLOW) // รถหยุด เมื่อ sensor ด้านล่างตรวจจับเส้นสีเหลืองได้
{
aStop();
bStop();
}
else
{
digitalWrite (led1,LOW);
if((hc.dist()>5)&&((digitalRead(ls)==HIGH)||((digitalRead(rs)==HIGH))&&(color == NOCOLOR)) //
เดินหน้าเมื่อ sensor ด้านหลัง ด้านข้าง และด้านล่าง ไม่ทำงาน
{
aReward(maxSpd);
bReward(maxSpd);
}
if((hc.dist()<5)&&((digitalRead(ls)==HIGH)||((digitalRead(rs)==HIGH))&&(color == NOCOLOR)) //
ถอยหลังเมื่อ sensor ด้านหลังทำงาน แต่ด้านข้างและล่างไม่ทำงาน
{
aForwardTime(3000);
bForwardTime(3000);
}
if((digitalRead(ls)==LOW)||((digitalRead(rs)==LOW)&&(color == NOCOLOR)) // เบรก เมื่อ sensor
ด้านข้างซ้ายหรือขวาทำงาน แต่ล่างไม่ทำงาน
{
aBreakTime(1000);
bBreakTime(1000);
}
if(color == BLACK) // กลับรถ เมื่อ sensor ด้านล่างตรวจจับเส้นสีดำได้
{
aForwardTime(5000);
bRewardTime(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 aReward(int speed)  
{  
digitalWrite(ia1, LOW);  
analogWrite(ia2, speed);
```

```
}  
void bReward(int speed)  
{  
    digitalWrite(ib1, LOW);  
    analogWrite(ib2, speed);  
}  
void aRewardTime(int time)  
{  
    digitalWrite(ia1, LOW);  
    analogWrite(ia2, maxSpd);  
    delay (time);  
}  
void bRewardTime(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);    }
```


เอกสารอ้างอิง

GitHub: pre-project62-SuayCherd

Folder: pre project final