การแยกมนุษย์ และสัตว์ ผ่านประตูเลื่อนอัตโนมัติ โดยUltrasonic Sensor

Isolation of human and animal through the automatic sliding doors by Ultrasonic Sensor.

โดย

นายธนทรัพย์ ตนะทิพย์ รหัสนักศึกษา B5047383

รายงานนี้เป็นส่วนหนึ่งของการศึกษาวิชา 427499 โครงงานวิศวกรรมโทรคมนาคม

และวิชา 427494 โครงงานศึกษาวิศวกรรมโทรคมนาคม

หลักสูตรวิศวกรรมศาสตรบัณฑิต สาขาวิชาวิศวกรรมโทรคมนาคม หลักสูตรวิศวกรรม

โทรคมนาคม-2546

สำนักวิชาวิศวกรรมศาสตร์ มหาวิทยาลัยเทคโนโลยีสุรนารี

ประจำภาคการศึกษาที่ 3 ปีการศึกษา 2555

การแยกมนุษย์และสัตว์ผ่านประตูเลื่อนอัตโนมัติ โดย Ultrasonic Sensor

คณะกรรมการสอบโครงงาน

(ผู้ช่วยศาสตราจารย์ ดร.พีระพงษ์ อุฑารสกุล)
กรรมการ/อาจารย์ที่ปรึกษาโครงงาน

(ผู้ช่วยศาสตราจารย์ ดร.ชุติมา พรหมมาก)
กรรมการ

(อ.เศรษฐวิทย์ ภูฉายา)
กรรมการ

มหาวิทยาลัยเทคโนโลยีสุรนารี อนุมัติให้นับรายงานโครงงานฉบับนี้ เป็นส่วนหนึ่งของการศึกษาระดับปริญญาตรี สาขาวิชาวิศวกรรมโทรคมนาคม รายวิชา 427499 โครงงานวิศวกรรมโทรคมนาคม และรายวิชา 427494 โครงงานศึกษาวิศวกรรมโทรคมนาคม ประจำปีการศึกษา 2555

โครงงาน	การแยกมนุษย์ และสัตว์ ผ่านประตูเลื่อนอัตโนมัติ โดยUltrasonic Sensor (Isolation of human and animal through the automatic sliding doors by Ultrasonic Sensor.)
จัดทำโดย	นายธนทรัพย์ ตนะทิพย์
อาจารย์ที่ปรึกษา	ผศ.ดร.พีระพงษ์ อุฑารสกุล
สาขาวิชา	วิศวกรรมโทรคมนาคม
ภาคการศึกษาที่	3/ 2555

บทคัดย่อ
(Abstract)

ในปัจจุบันมีการนำเครื่องมือเซนเซอร์ ทั้งหลายมาใช้วัดค่าทางไฟฟ้า ทางกล และทางตรรกะบ้าง ซึ่งมีวางขายตามท้องตลาดมากมาย แต่เมื่อนำมาใช้แล้วประสิทธิภาพขึ้นอยู่ตามราคา แต่ที่แก้ไขปรับปรุงได้ก็ราคาสูงมาก ดังนั้นเมื่อเราได้ศึกษาและหาความรู้ด้วยตนเอง ก็สามารถสร้างขึ้นได้ และดัดแปลงให้สามารถนำมา ใช้ให้มีประสิทธิภาพมากขึ้น จากโครงงานเรื่องการแยกมนุษย์และสัตว์ ผ่านประตูเลื่อนอัตโนมัติ โดยอุลตร้าโซนิคเซนเซอร์(Isolation of human and animal through the automatic sliding doors by Ultrasonic Sensor.) ที่เป็นปัญหาสำหรับทุกร้านค้าที่ คอยต้อนรับสัตว์ตัวน้อยๆ ที่ผ่านเข้ามาอย่างง่ายดาย ซึ่งวัดจากรังสีอินฟราเรดที่แผ่จากตัวเราเอง และสัตว์เลือดอุ่นทั้งหลาย

ด้วยเหตุนี้จึงทำการศึกษาอุปกรณ์อิเล็กทรอนิกส์ หลักการทำงานของ Board Arduino mega 2560 R3 รวมไปถึงตรรกะ ในการเขียนโปรแกรมสำหรับ Arduino 1.0.4 มีลักษณะคล้ายภาษา C++ ซึ่งง่ายต่อการประยุกต์ใช้งาน คำนวณ และทำเป็นสมการให้ง่ายต่อการจัดการตัวแปร เช่นความสูงของประตู สามารถหามุมที่ดีที่สุดของสองตัวเซนเซอร์ที่มีประสิทธิภาพในการแยกมนุษย์และสัตว์ในการผ่านเข้าประตูไปได้

กิตติกรรมประกาศ

(Acknowledgement)

จากการที่ข้าพเจ้าได้รับมอบหมายให้ทำโครงงานเรื่อง การแยกมนุษย์ และสัตว์ ผ่านประตูเลื่อนอัตโนมัติ โดยอุลตร้าโซนิคเซนเซอร์(Isolation of human and animal through the automatic sliding doors by Ultrasonic Sensor.) ส่งผลให้ข้าพเจ้าได้รับความรู้และประสบการณ์ต่างๆ เกี่ยวกับการเขียนโปรแกรมด้วยโปรแกรม Arduino 1.0.4 (มีความคล้ายคลึงกับโปรแกรมภาษา C++) บัดนี้โครงงานดังกล่าวพร้อมทั้งรายงานได้สำเร็จลงแล้ว ทั้งนี้ด้วยความร่วมมือและสนับสนุนจาก อาจารย์ที่ปรึกษาโครงงาน ผศ.ดร. พีระพงษ์ อุฑารสกุล คณะอาจารย์ประจำสาขาวิชาวิศวกรรมโทรคมนาคม สำนักวิชาวิศวกรรมศาสตร์ ทุกท่าน

สุดท้ายนี้ข้าพเจ้าขอขอบพระคุณผู้ที่มีส่วนเกี่ยวข้องทุกท่านที่มีส่วนร่วมในการให้ข้อมูลเป็นที่ปรึกษาในการทำรายงานฉบับนี้จนเสร็จสมบูรณ์ ตลอดจนให้การดูแลและให้ความเข้าใจเกี่ยวกับพื้นฐานแนวคิดต่างๆ รวมไปถึงศูนย์เครื่องมือวิทยาศาสตร์และเทคโนโลยี มหาวิทยาลัยเทคโนโลยีสุรนารี ที่ได้ส่งเสริมทุนอุดหนุน และให้ใช้ห้องปฏิบัติการทำโครงงานนี้จนสำเร็จ

ข้าพเจ้าขอกราบขอบพระคุณเป็นอย่างสูง ณ โอกาสนี้

นายธนทรัพย์ ตนะทิพย์

สารบัญ

สารบัญ(ต่อ)

สารบัญ(ต่อ)

สารบัญภาพ

สารบัญภาพ(ต่อ)

สารบัญตาราง

บทที่ 1
บทนำ

ความต้องการใช้เซนเซอร์ในงานตรวจวัดและควบคุมสาขาต่าง ๆ กำลังเพิ่มขึ้น ตามแนวโน้มของปัจจัยนำ (Drivers and Trends)ในด้านการพัฒนาพลังงาน ความปลอดภัย สุขภาพ สิ่งแวดล้อม และสังคมยูบิควิตัส (Ubiquitous Society) ซึ่งล้วนส่งผลต่อการพัฒนาภาคอุตสาหกรรมและภาคบริการของประเทศ โปรแกรมเทคโนโลยีเซนเซอร์ จึงถูกออกแบบให้เป็นโปรแกรมการวิจัยและพัฒนาที่มุ่งเน้นให้ความสำคัญกับการ สร้างขีดความสามารถด้านเทคโนโลยีเซนเซอร์ เพื่อสร้างโอกาสใหม่ ๆ ในการพัฒนาประเทศจากเทคโนโลยีดังกล่าว และนำไปสู่การลดพึ่งพาเทคโนโลยีที่เกินความจำเป็นจากต่างประเทศ ตลอดจนเป็นการยกระดับขีดความสามารถอุตสาหกรรมและบริการที่มีประสิทธิภาพยิ่งขึ้น

ด้วยเหตุนี้การเปิดปิดประตูอัตโนมัติ จึงมีเซ็นเซอร์ตรวจจับความเคลื่อนไหว (Motion Sensor) ซึ่งเป็นอุปกรณ์ที่แปลงการตรวจจับความเคลื่อนไหวเป็นสัญญาณไฟฟ้า โดยทั่วไปเซ็นเซอร์ตรวจจับความเคลื่อนไหวมี 3 ประเภทคือ

1. Passive infrared sensors (PIR) เป็นเซ็นเซอร์ที่รับความร้อนจากร่างกายเมื่อเคลื่อนที่ ไม่มีการปล่อยพลังงานออกมาจากเซ็นเซอร์
2. Ultrasonic เป็นเซ็นเซอร์ที่มีการปล่อยคลื่นอัลตร้าโซนิกออกมาและตรวจวัดการสะท้อนของคลื่นเมื่อวัตถุเคลื่อนที่
3. Microwave เป็นเซ็นเซอร์ที่มีการปล่อยคลื่นไมโครเวฟออกมาและตรวจวัดการสะท้อนของคลื่นเมื่อวัตถุเคลื่อนที่

ในโครงงานนี้ได้เลือกใช้เซ็นเซอร์ตรวจจับความเคลื่อนไหวประเภท Ultrasonic sensors

1.1 ความเป็นมาและความสำคัญของปัญหา

การเพิ่มระบบอัตโนมัติเปิดปิดประตูบริการให้ลูกค้าในบริษัท ห้างร้านต่างๆ รวมไปถึง การใช้งานสำหรับในบ้านด้วย เป็นสิ่งที่ทำให้มีความสะดวกสบาย โดยใช้ตัวเซนเซอร์จับความเคลื่อนไหว ชนิดต่างๆ แต่ก็มีขีดจำกัด ในการดัดจับสิ่งที่ไม่พึ่งประสงค์ให้ผ่านเข้ามาได้ เช่น สุนัข แมว สัตว์เลื้อยคลานต่างๆ เป็นต้น ด้วยเหตุนี้จึงเป็นปัญหาอย่างมากสำหรับผู้ใช้อุปกรณ์ ที่ไว้จับความเคลื่อนไหวเพียงอย่างเดียว เราจึงคิดค้นหาวิธีการโดยใช้ Ultrasonic sensors เพียง 2 ตัวเท่านั้น ซึ่งสามารถจับความเคลื่อนไหว พร้อมแยกประเภท มนุษย์ และสัตว์

1.2 วัตถุประสงค์ ขอบเขต และเป้าหมายของโครงงาน

1.2.1 วัตถุประสงค์

- เพื่อศึกษาระบบการทำงานของ Arduino mega 2560 r3 Board
- เพื่อศึกษาการและออกแบบวงจรร่วมกับอุปกรณ์ module ultrasonic sensor hc-sr04 , ic(integrated circuit) CD4050BE , keypad 4 x 4 , หลอดLED และจอ Graphic LCD 84x48 Nokia 5110
- เพื่อศึกษาการใช้ Sensor 2 ตัว ในการวัดส่วนสูงและความยาวฐานในแนวแกน y และ x
- ให้มีความรู้การต่อวงจรอุปกรณ์และประยุกต์ใช้งานกับบอร์ด arduino โดยการเขียนโปรแกรม arduino-1.0.4 (มีลักษณ์คล้ายภาษาC++) เพื่อประมวลผลให้สามารถแยกมนุษย์และสัตว์ได้
- เพื่อศึกษาการการติดตั้งอุปกรณ์ ให้สัมพันธ์กันกับการประมวล Arduino mega 2560 r3 Board

1.2.2 ขอบเขตงาน

1. ศึกษาลักษณะการสะท้อนกลับของคลื่น และเวลาที่ใช้ในการสะท้อนกลับ
2. ออกแบบกลไกในการประมูลผลในการวัดส่วนสูงและความยาวฐาน ในแกน y และ x
3. ออกแบบวงจรเชื่อมต่อเพื่อใช้งานระหว่าง Board และSensor แกน x และ y
4. เขียนโปรแกรมควบคุม ลง Board และให้Sensor วัดค่าได้
5. เขียนโปรแกรมประมวลผลค่าการวัดแกน x และ y และแยกประเภทมนุษย์และสัตว์ได้ โดยสัมพันธ์กับการเปิดปิดประตูอัตโนมัติได้
6. สร้างอุปกรณ์ต้นแบบทั้งหมดและทดสอบเพื่อให้ได้ตามวัตถุประสงค์

1.2.3 เป้าหมาย

โครงงานนี้เมื่อเสร็จสมบูรณ์ดีแล้ว สามารถนำไปใช้ในระบบเปิดปิดประตูอัตโนมัติให้สามารถแยกมนุษย์ และสัตว์ ให้มีประสิทธิภาพมากขึ้น และนำความรู้ที่ได้ไปต่อยอดประดิษฐ์อุปกรณ์ต่างๆ นำมาใช้ในชีวิตประจำวัน

บทที่ 2
รายละเอียดของการพัฒนา

2.1 การออกแบบระบบโดยรวม

ในขั้นตอนการพัฒนาโครงการนักศึกษาได้ทำการออกแบบภาพรวมทั้ง หมดของโครงการทั้งหมดประกอบไปด้วย

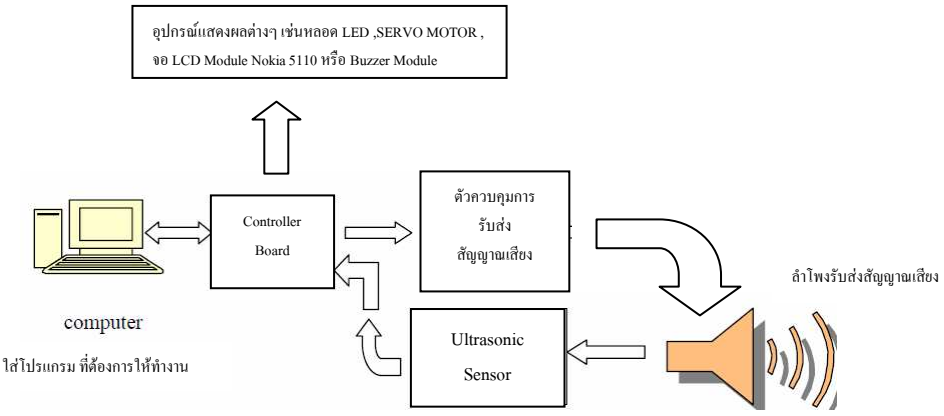
- **อุปกรณ์ที่ใช้ในการพัฒนาทำโครงงานดังนี้**

1. Arduino mega 2560 r3 Board 1 ตัว
2. คอมพิวเตอร์ 1 เครื่อง
3. ตัวควบคุมการรับส่งสัญญาณเสียงหรือ Ultrasonic Sensor 2 ตัว สำหรับอ้างอิงตำแหน่งที่ระบุแน่นอนหนึ่งตัว และ ปรับองศาได้อีกหนึ่งตัวเพื่อความยืดหยุ่นในการทำงาน ให้มีละเอียดการจับวัตถุมากน้อยตามที่ต้องการ
4. วัสดุทำแนวแกน x และ y
5. Adapter (9V 1A) สายไฟหรือสายส่งสัญญาณ สกรูนอต เหล็กฉาก
6. อุปกรณ์แสดงผลต่างๆ เช่นหลอด LED ,SERVO MOTOR , จอ LCD Module Nokia 5110 หรือ Buzzer Module เพื่อให้สื่อสารหรือสนองต่อ ผู้ควบคุมให้ทราบ

- **เครื่องมือ**

1. ค้อน คีมปากแหลม ไขควง ชนิดต่างๆ เช่น ปากแฉก ปากแบน เป็นต้น
2. โปรแกรม arduino-1.0.4 (ใช้ภาษาคอมพิวเตอร์ภาษา C++)

ซึ่งทำงานร่วมกันตามรูปที่ 1



รูปที่ 1 แสดงผังไดอะแกรมโครงงาน การศึกษาระบบทำงานของ Ultrasonic Sensor

2.2 ทฤษฎีที่เกี่ยวข้อง

2.2.1 ระบบอุลตร้าโซนิค (Ultrasonic)

คลื่นเสียงที่มีความถี่สูงเกินกว่าที่หูมนุษย์จะได้ยิน โดยทั่วไปแล้วหูของมนุษย์โดยเฉลี่ยจะได้ยินเสียงสูงถึงเพียงแค่ประมาณ 15 เท่านั้น แต่พวกที่อายุยังน้อย ๆ อาจจะได้ยินเสียงที่มีความถี่สูงกว่านี้ได้ ดังนั้นโดยปกติแล้วคำว่าอุลตร้าโซนิคจึงมักจะหมายถึงคลื่นเสียงที่มีความถี่สูงกว่า 20 ขึ้นไป จะสูงขึ้นจนถึงเท่าใดไม่ได้ระบุจำกัดเอาไว้

เป็นคลื่นทีมีทิศทางทำให้เราสามารถเล็งคลื่นเสียงไปยังเป้าหมายที่ต้องการได้โดยเจาะจง เรื่องนี้เป็นคุณสมบัติของคลื่นอย่างหนึ่ง ยิ่งคลื่นมีความถี่สูงขึ้นความยาวคลื่นก็จะยิ่งสั้นลง ถ้าความยาวคลื่นยาวกว่าช่องเปิด (ที่ให้เสียงนั้นออกมา)ของตัวกำเนิดเสียงความถี่นั้นเช่น คลื่นความถี่ 300 Hz ในอากาศจะมีความยาวถึงประมาณ 1 เมตรเศษ ๆ ซึ่งจะยาวกว่าช่องที่ให้คลื่นเสียงออกมาจากตัวกำเนิดเสียงโดยทั่วไปมากมายคลื่นจะหักเบนที่ขอบด้านนอกของตัวกำเนิดเสียงทำให้เกิดการกระจายทิศทางคลื่นแต่ถ้าความถี่สูงขึ้นมาอยู่ในย่านอุลตร้าโซนิค อย่างเช่น 40 จะมีความยาวคลื่นในอากาศเพียงประมาณ 8 มม. เท่านั้นซึ่งเล็กกว่ารูเปิดของตัวที่ให้กำเนิดเสียงความถี่นี้มากคลื่นเสียงจะไม่มีการเลี้ยวเบนที่ขอบจึงพุ่งออกมาเป็นลำแคบ ๆ หรือที่เราเรียกว่า มีทิศทาง

การมีทิศทางของคลื่นเสียงย่านอุลตร้าโซนิคทำให้เรานำไปใช้งานได้หลายอย่าง เช่น นำไปใช้ในเครื่องควบคุมระยะไกล (Ultrasonic remote control) เครื่องล้างอุปกรณ์ (Ultrasonic cleaner) โดยให้น้ำสั่นที่ความถี่สูง เครื่องวัดความหนาของวัตถุโดยสังเกตระยะเวลาที่คลื่นสะท้อนกลับมา เครื่องวัดความลึกและทำแผนที่ใต้ท้องทะเล ใช้ในเครื่องหาตำแหน่งอวัยวะบางส่วนในร่างกาย ใช้ทดสอบการรั่วไหลของท่อ เป็นต้น โดยความถี่ที่ใช้ขึ้นอยู่กับการใช้งาน เช่น คลื่นเสียงต้องเดินทางผ่านอากาศแล้ว ความถี่ที่ใช้ก็มักจะจำกัดอยู่เพียงไม่เกิน 50 เพราะที่ความถี่สูงขึ้นกว่านี้อากาศจะดูดกลืนคลื่นเสียงเพิ่มขึ้นมาก ทำให้ระดับความแรงของคลื่นเสียงที่ระยะห่างออกไปลดลงอย่างรวดเร็ว ส่วนการใช้งานด้านการแพทย์ซึ่งต้องการรัศมีทำการสั้น ๆ ก็อาจใช้ความถี่ในช่วง 1 MHz ถึง 10 MHz ขณะที่ความถี่เป็น GHz (10^9 Hz) ก็มีใช้กันในหลาย ๆการใช้งานที่ตัวกลางที่คลื่นเสียงเดินทางผ่านไม่ใช่อากาศ

อุปกรณ์ที่สามารถแปลงพลังงานในรูปอื่นให้มาเป็นพลังงานทางกลโดยการสั่นไปมา ซึ่งทำให้เกิดคลื่นเสียงย่านอุลตร้าโซนิคกระจายไปในอากาศได้หรือแปลงพลังงานทางกลให้มาเป็นพลังงานในรูปอื่นได้นั้น มีชื่อเรียกว่า อัลตร้าโซนิคทรานสดิวเซอร์ (Ultrasonic Transducer) ในปัจจุบันอุลตร้าโซนิคทรานสดิวเซอร์มีหลายแบบขึ้นอยู่กับหลักการที่ใช้

แบบที่นิยมใช้กันมากได้แก่

แบบเปียโซอิเลคทริก (Piezo-electric Transducer) ซึ่งแปลงไปมาระหว่างพลังงานไฟฟ้าและพลังงานทางกล โดยมีความถี่เรโซแนนซ์คงที่อยู่ค่าหนึ่ง

แบบแมกนีโตสตริคทีฟ (Magnetostrictive Transducer) ซึ่งแปลงไปมาระหว่างพลังงานไฟฟ้าในขดลวดกับตำแหน่งความยาวของแกนเหล็กที่สวมขดลวดนั้นอยู่

แบบอิเลคโตรสตริคทีฟ (Electrostrictive Transducer) ซึ่งแปลงไปมาระหว่างพลังงานไฟฟ้ากับพลังงานทางกล

ในการทำโครงงานนี้เราใช้ ทรานสดิวเซอร์แบบเปียโซอิเลคทริค ภายในตัวอุลตร้าโซนิคทรานสดิวเซอร์แบบเปียโซอิเลคทริค แบบที่มีใช้กันในปัจจุบันซึ่งได้รับการพัฒนากันมาในระดับหนึ่งแล้วจะประกอบด้วยชิ้นสารเซรามิคสี่เหลี่ยมซึ่งมีผิวโลหะเงินฉาบอยู่ทั้ง 2 หน้าเพื่อให้ต่อสายไฟออกมาเป็นขา 2 ขา ชิ้นสารเซรามิคนี้ประกอบขึ้นจากสารเซรามิค 2 ชิ้น ประกบกันอยู่โดยวางให้ขั้วไดโพลทางไฟฟ้าภายในอะตอมของมันมีทิศทางตรงข้ามกันดังรูปที่ 2

รูปที่ 2

(ก).โครงสร้างภายในตัวอุลตร้าโซนิคทรานสดิวเซอร์แบบเปียโซอิเลคทริคที่ใช้สารเซรามิค

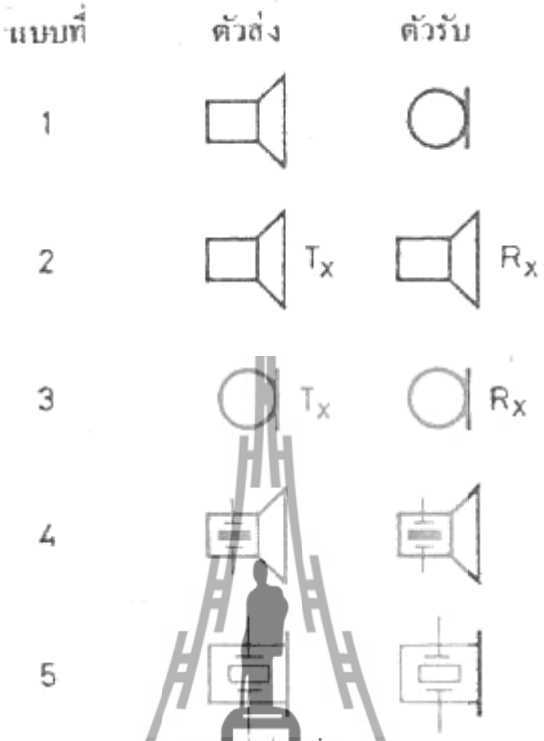
(ข).เมื่อป้อนแรงดันให้แก่ตัวมันจะทำให้ชิ้นสารเซรามิคโก่งงอไปมาทำให้เกิดคลื่นเสียงอุลตร้าโซนิคกระจายไปในอากาศ

ชิ้นสารเซรามิคถูกยึดติดภายในตัวถังอย่างดีเพื่อไม่ให้การสั่นขณะที่มันทำงานอยู่ได้รับผลกระทบกระเทือนจากภายนอกตัวถังมักจะเป็นรูปทรงกระบอกที่มีเส้นผ่าศูนย์กลางและมีความสูงประมาณ 1 ถึง 2.5 ซม. ด้านหน้าทำเป็นช่องเปิดมีตะแกรงติดอยู่เพื่อให้คลื่นอุลตร้าโซนิคเข้ามาหรือออกจากช่องเปิดได้โดยสะดวก ถ้าตัวถังทำมาจากโลหะก็ควรต่อตัวถังลงกราวด์เพื่อทำหน้าที่ชีลด์ สำหรับบางยี่ห้อเขาจะต่อขาหนึ่งติดกับตัวถังมาให้เลย เมื่อพลิกดูขา 2 ขาที่โผล่ออกมาจากตัวถังจะเห็นมีขาหนึ่งติดกับตัวถัง เมื่อมีสัญญาณแรงดันมาตกคร่อมขั้วทั้งสองของชิ้นสารเซรามิคดังรูป (ข) จะทำให้ชิ้นสารโก่งงอมากหรือน้อยหรือในทิศทางใดตามขนาดและทิศทางการเปลี่ยนแปลงขนาดของสัญญาณนั้น ๆ ทำให้เกิดการกดอัดอากาศโดยรอบเกิดเป็นคลื่นเสียงที่มีความถี่เดียวกับสัญญาณนั้นออกไป โดยทั่ว ๆ ไปกำลังเอาท์พุทที่ออกมาจะตกประมาณ 10% ของกำลังไฟฟ้าที่ป้อนเข้าไป แต่กำลังเอาท์พุท์จะสูงสุดที่ค่าประมาณนี้ต่อเมื่อความถี่ของสัญญาณตรงกับความถี่เรโซแนนซ์ซึ่งเป็นความถี่ทางกลตามธรรมชาติของชิ้นสารเซรามิคนั้น ๆ ส่วนที่ความถี่อื่น ๆ กำลังเอาท์พุทจะลดลงกว่านี้มาก

ในทำนองกลับกันเมื่อมีคลื่นเสียงที่มีความถี่ตรงกับความถี่เรโซแนนซ์ของชิ้นสารเซรามิคเข้ามาจะทำให้ชิ้นสารโก่งงอไปมาและเกิดสัญญาณแรงดันซึ่งมีขนาดเล็กขึ้นมาคร่อมขั้วทั้งสองของตัวมันเองได้ คุณสมบัติโดยทั่วไปของอุลตร้าโซนิคทรานสดิวเซอร์แบบเปียโซอิเลคตริคก็คือมีค่าความต้านทานไฟตรงสูงมากอาจสูงถึง 100 MW เรียกว่าถ้าเอามัลติมิเตอร์ธรรมดามาตั้งสเกลวัดค่าความต้านทานสูง ๆ เข็มจะไม่กระดิกเลย แต่ในขณะที่มันทำงานความต้านทานทางด้านไฟสลับจะลดลง

ตัวส่งและตัวรับ ทรานสดิวเซอร์แบบเปียโซอิเลคตริคที่ใช้สารเซรามิค (หรือที่ผู้ผลิตบางรายเรียกว่าอุลตร้าโซนิคทรานสดิวเซอร์แบบเซรามิค) จะมีอยู่ 2 อย่าง คือ ตัวส่งหรือ Transmitter และ ตัวรับ (เสียง) หรือ Receiver

- ตัวส่งคืออุลตร้าโซนิคทรานสดิวเซอร์ที่ถูกออกแบบเจาะจงมาให้แปลงสัญญาณไฟฟ้าที่ให้แก่ตัวมัน ให้ออกมาเป็นคลื่นเสียงย่านอุลตร้าโซนิค หน้าที่ของตัวส่งจึงคล้าย ๆ กับเป็นลำโพง
- ตัวรับคืออุลตร้าโซนิคทรานสดิวเซอร์ที่ถูกออกแบบเจาะจงมาให้แปลงคลื่นเสียงย่านอุลตร้าโซนิคที่มาตกกระทบตัวมันให้ออกมาเป็นสัญญาณไฟฟ้า หน้าที่ของตัวรับจึงคล้าย ๆ กับเป็นไมโครโฟน ด้วยเหตุนี้เวลาเขียนสัญลักษณ์ของอุลตร้าโซนิคทรานสดิวเซอร์จึงนิยมเขียนตามหน้าที่ของมันคือถ้าเป็นตัวส่งก็เขียนสัญลัษณ์เป็นลำโพง ถ้าเป็นตัวรับก็เขียนสัญลักษณ์เป็นไมโครโฟน ดังรูปที่ 3



รูปที่ 3

แสดงตัวอย่างการเขียนสัญลักษณ์ของอุลตร้าโซนิคทรานสดิวเซอร์แบบต่างๆ กัน

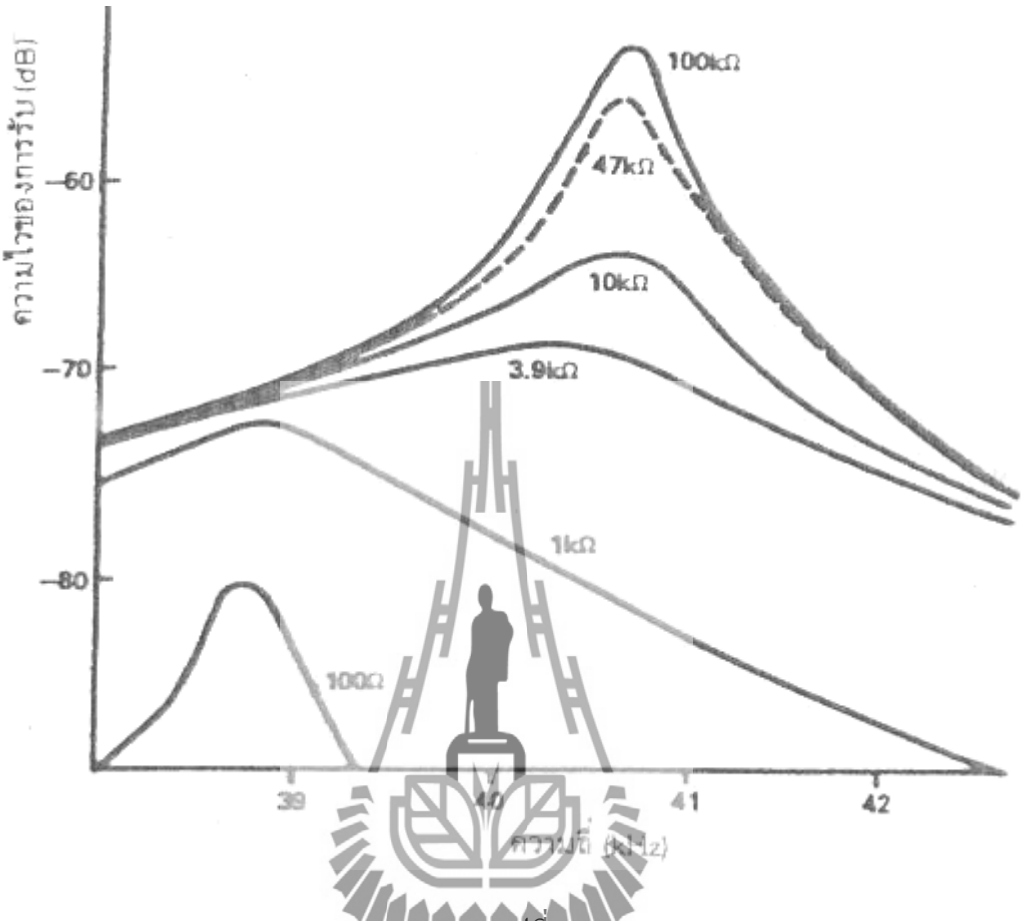
อุลตร้าโซนิคทรานสดิวเซอร์แบบเซรามิคที่มีจำหน่ายทั่วไปจะมีค่าความถี่เรโซแนนซ์ให้เลือกตั้งแต่ 23 ขึ้นไปจนถึง 40 แต่ที่พบเห็นกันบ่อยก็มี 23, 25, และ 40 โดยความถี่ 40 เป็นรุ่นที่นิยมใช้กันมากที่สุดเพราะมีทิศทางดีกว่า

2.2.2 ข้อควรรู้ในการใช้งานตัวส่งและตัวรับ

เนื่องจากสเปคตลอดจนรายละเอียดต่าง ๆ ของอุลตร้าโซนิคทรานสดิวเซอร์หาได้ยาก ดังนั้นจึงสามารถที่จะสรุปสิ่งที่ควรรู้ในขั้นต้นของอุปกรณ์อุลตร้าโซนิคทรานสดิวเซอร์เพื่อเป็นแนวทางในการใช้งานดังนี้

1. ไม่ควรให้ตัวทรานสดิวเซอร์ได้รับการกระแทกหรือตกจากที่สูง เพื่อป้องกันโครงสร้างภายในมิให้เสียหาย
2. ทรานสดิวเซอร์ที่มีขายกันโดยทั่วไปจะทนแรงดันตกคร่อมตัวมันสูงสุดได้ไม่เกินกว่า 20 Vrms ดังนั้นขนาดของสัญญาณที่จะป้อนให้กับทรานสดิวเซอร์ก็ควรจะอยู่ภายในขีดจำกัดอันนี้

3. ความถี่เรโซแนนซ์ (ความถี่ที่ตัวมันทำงานได้อย่างมีประสิทธิภาพสูงสุด) ของทรานสดิวเซอร์ 40 ที่มีขายกันโดยทั่วไปจะผิดพลาดไปไม่เกิน ± 1 และมีแถบความถี่ (Bandwidth) ประมาณ 4.5 สำหรับตัวส่ง และมีแถบความถี่ประมาณ 5.0 สำหรับตัวรับ จะเห็นได้ว่าแถบความถี่ของตัวรับจะกว้างกว่าของตัวส่งอยู่เล็กน้อย เพื่อให้แน่ใจว่าตัวรับจะสามารถรับความถี่ทั้งหมดที่ออกมาจากตัวส่งได้
4. อุณหภูมิใช้งานของตัวทรานสดิวเซอร์ควรอยู่ภายในช่วง –20 ° C ถึง +60° C
5. ทั้งตัวส่งและตัวรับจะมีทิศทางคล้ายคลึงกันมากกล่าวคือ ที่ตำแหน่งเบนจากแนวแกนของตัวส่งไปประมาณ 30° ความแรงของคลื่นเสียงที่ถูกส่งออกไปจะลดลงจากแนวแกนประมาณ 10 dB ในทำนองเดียวกันถ้าคลื่นเสียงพุ่งเข้ามาในแนวที่เบี่ยงเบนไปจากแนวแกนของตัวรับไปประมาณ 30° ความไวหรือขนาดแรงดันที่ออกมาก็ลดลงไปประมาณ 10 dB ด้วยเช่นกัน ดังนั้นในการใช้งานที่เป็นการควบคุมระยะไกลในที่โล่งแจ้งจึงควรพยายามให้ทั้งตัวรับและตัวส่งอยู่ในแนวที่พุ่งตรงเข้าหากันให้มากที่สุด อย่างไรก็ตามในกรณีที่อยู่ในห้องอาจจะเบี่ยงเบนจากกันได้มากหน่อย เพราะคลื่นเสียงอุลตร้าโซนิคสามารถสะท้อนกับกำแพง พื้น และวัตถุที่อยู่ภายในห้อง ทำให้คลื่นเสียงเข้าไปหาตัวรับได้หลายทาง
6. ในกรณีที่ใช้งานตัวรับจะต้องมีตัวต้านทานต่อขนานกับตัวรับเพื่อทำหน้าที่เป็นโหลด ตามปกติแล้วตัวต้านทานตัวนี้ควรมีค่าอยู่ในช่วงจาก 10 kW - 100 kW จากการทดลองพบว่าถ้าเปลี่ยนโหลดจาก 100 kW มาเป็น 10 kW ความไวจะลดลงประมาณ 10 ถึง 20 dB แต่แถบความถี่จะกว้างขึ้น ถ้าใช้ค่าความต้านทานต่ำลงไปอีก ความถี่เรโซแนนซ์ (ความถี่กลาง) จะลดลงมากกระทั่งว่า ถ้าการใช้งานมีสัญญาณรบกวนมากควรใช้โหลดที่มีความต้านทานสูงสักหน่อย เพื่อให้ตัวส่งมีความไวสูงและมีแถบความถี่แคบ ตัวอย่างการทดสอบแสดงไว้ดังรูปที่ 4
7. ตามปกติแล้วเราสามารถนำเอาตัวส่งและรับมาใช้งานแทนกันได้ในการใช้งานส่วนใหญ่ และตัวส่งหรือตัวรับของยี่ห้อใด รุ่นใด ก็สามารถที่จะนำมาใช้แทนกันได้ในงานส่วนใหญ่ ขอเพียงแต่ให้มีความถี่เรโซแนนซ์เดียวกันเท่านั้นเอง อย่างไรก็ตามในบางกรณีอาจต้องเปลี่ยนแปลงค่าความต้านทานสมมูลย์ทางด้านไฟสลับเพื่อให้ลักษณะผลตอบสนองทางความถี่สอดคล้องกับของเดิม



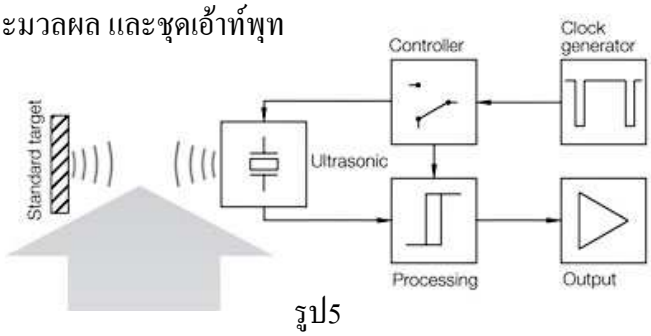
รูปที่ 4

แสดงผลการทดลองตัวรับตัวหนึ่งโดยลองเปลี่ยนโหลดเป็นค่าต่าง ๆ กัน

แล้วป้อนคลื่นเสียงความถี่ต่างๆกันเข้ามา

2.2.3 อุลตร้าโซนิคเซ็นเซอร์หน้าที่และการทำงาน

รูปแบบต่าง ๆ ของอุลตร้าโซนิคเซ็นเซอร์ประกอบด้วย ตัวตรวจจับด้วยคลื่นอุลตร้าโซนิค ชุดส่งสัญญาณ ชุดประมวลผล และชุดเอ้าท์พุท



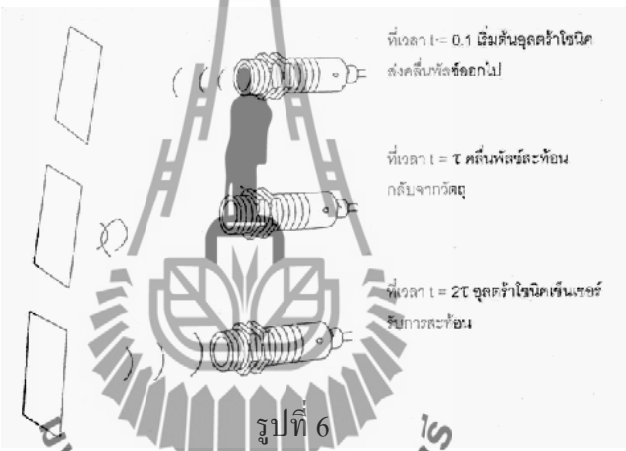
รูป5

หลักการทำงานของอุลตร้าโซนิค

มักจะใช้เป็นภาครับ และ ภาคส่ง อาจมีระบบซึ่งประกอบด้วยส่วนหลัก ๆ แยกกันอยู่ 2 ส่วน ในระหว่างการทำงาน เซ็นเซอร์จะทำการส่งสัญญาณเสียงซึ่งเรียกว่า “ซาวด์พาร์เซลส์” (Sound parcels) ให้ขบวนการทางอิเล็กทรอนิกส์ ของเวลาทำงานไปเรื่อย ๆ จนกระทั่งมีการ รับการสะท้อนครั้งแรกเกิดขึ้น

2.2.4 วงจรส่งผ่าน / รับ

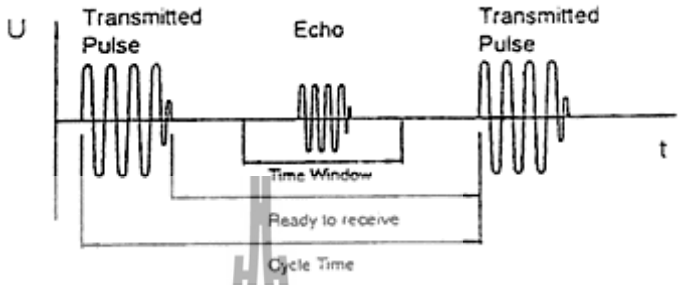
สำหรับการทำงานเป็นวงจรของอุลตร้าโซนิคเซ็นเซอร์ จะส่งผ่านคลื่นพัลซ์เสียงที่ช่วงเวลาสม่ำเสมอ หรือช่วงเวลาที่เปลี่ยนแปลง คลื่นเสียงที่ปล่อยออกไปจะถูกสะท้อนได้โดยวัตถุที่เหมาะสม โดยเซ็นเซอร์ และระบบการทำงานจะรับการสะท้อนของคลื่นเสียงที่สะท้อนกลับมา (ดังแสดงในรูปที่ 6) ความกว้างของคลื่นพัลซ์ของเสียงอยู่ในช่วง 2.-200 ไมโครเซท



รูปที่ 6
อุลตร้าโซนิคเซ็นเซอร์ วงจรส่งผ่านและรับ

เวลาในการเดินทางของคลื่นพัลซ์ของคลื่นเสียงเป็นการวัดระยะห่างจากวัตถุ ทั้งนี้ขึ้นอยู่กับชนิดของเซ็นเซอร์ ระยะห่างนี้นำไปแสดงในรูปของ สัญญาอนาล็อก (Analague Signal) (เช่น 0-20 mA) สัญญาณลอจิก (Logic Signal) (เช่น สัญญาณลอจิก 8 bit) ตลอดทั้ง ซีเรียลอินเตอร์เฟส (Serial Interface) (RS232) หรือการเปรียบเทียบกับค่าอ้างอิงในรูปของสวิสซ์พัลซ์ที่เรียกว่า ไทม์เฟรม (Time Frame) เนื่องจากขบวนการดำเนินไปตามเวลาที่คลื่นสะท้อนเดินทาง ไม่ใช่เป็นไปตามความเข้มของคลื่นสะท้อน จึงจัดได้ว่าอุลตร้าโซนิคเซ็นเซอร์ มีข้อดีเหนือกว่าเซ็นเซอร์แบบออปติคอล (Optical Sensor) เวลาที่คลื่นสะท้อนการเดินทางจะ ทำให้ขบวนการดำเนินโดยไม่ขึ้นกับความเข้มของคลื่นสะท้อน ตราบเท่าที่วัตถุยังคงสะท้อนคลื่นที่สามารถตรวจจับได้ออกมา ดังนั้นคุณลักษณะการสวิทซ์ไม่เปลี่ยนไป แม้ในสภาวะที่การสะท้อนเป็นไปอย่างไม่ดีคลื่นสะท้อนที่อ่อนจะมีผลต่อความถูกต้องในการตรวจจับวัตถุ ซึ่งอาจทำให้ไม่สามารถทำการตรวจจับวัตถุได้เลย

ความเร็วที่เปลี่ยนไปของคลื่นพัลซ์ของเสียง มีผลกระทบต่อพิสัย การทำงานของสวิทช์ (ระยะทาง) โดยตรงเซ็นเซอร์ทำงานด้วยวงจรเวลาที่คงที่ (เช่น t = 20 ms) จะส่งคลื่นเสียงออกมาอย่างสม่ำเสมอ (ดังแสดงในรูปที่ 7)ดังนั้นวงจรเวลาจะเป็นตัวกำหนดช่วงและวงจรการทำงานของสวิทซ์ของเซ็นเซอร์



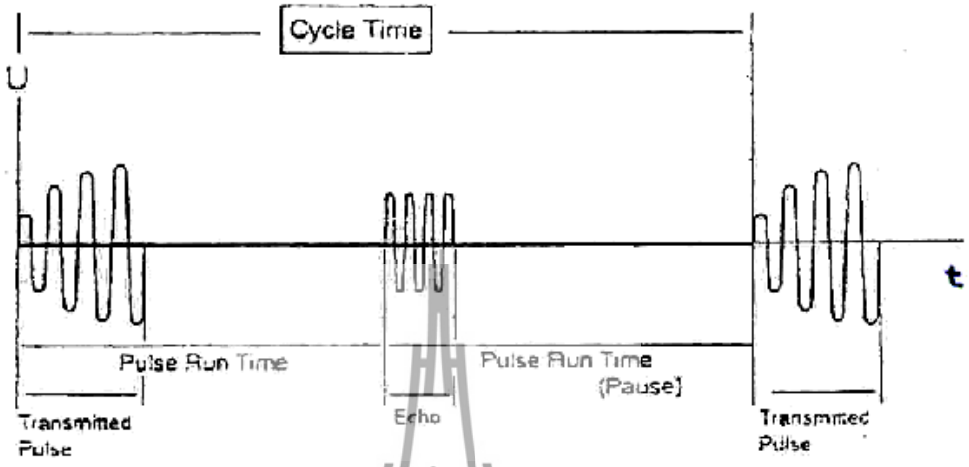
รูปที่ 7

อุลตร้าโซนิคเซ็นเซอร์ , วงจรเวลาคงที่

ยกตัวอย่าง เช่นคลื่นเสียงที่มีความเร็ว v = 340 m/s (20'"c) ในช่วงเวลา t= 20 ms (50 Hz) จะเดินทางS= Vxt = 6.8 m เนื่องจากระยะห่างระหว่างเซ็นเซอร์ และวัตถุที่ได้จากการทำงานของเซ็นเซอร์ คิดที่ไปและกลับจึงได้ระยะทางจริงสูงสุดสำหรับวงจรเวลานี้เป็น 3.4 m แอมปลิจูดของส่วนของคลื่นเสียงและซิทิวิตี้ (Sensitivity) ของตัวรับต้องมีการพิจารณาเลือกใช้ เพื่อให้คลื่นสะท้อนที่เดินทางมาถึงหลังจากเลยเวลาของวงจรเวลาที่

กำหนดไปแล้วจะไม่ได้รับการตรวจจับเนื่องจากคลื่นสะท้อนนั้นอ่อนมากซึ่งสัญญาณคลื่นนี้จะทำให้เซ็นเซอร์สวิทส์ มีการทำงานผิดพลาดหรือให้ข้อมูลที่ผิด เซ็นเซอร์แบบ อนาลอก (Analog Sensor) เพื่อให้การตรวจจับวัตถุเป็นไปอย่างถูกต้อง วัตถุต้องอยู่นิ่งเป็นเวลาที่พอเพียงสำหรับสำหรับสะท้อนอย่างน้อย 1 ส่วน ของคลื่นเสียงภายในขอบเขตที่เซ็นเซอร์จะทำงานได้รอบมากที่สุดของการสวิทซ์ ซึ่งจะเปลี่ยนแปลงกับอัตราส่วนของวัตถุต่อที่ว่าง และจะพิจารณาให้มีค่าน้อยกว่ารอบของความถี่ที่จุดนี้ เวลาที่ขยายจะสิ้นสุดระหว่างการส่งผ่านของพัลซ์ และการรับคลื่นสะท้อนแรกจะถูกนำไปใช้วัดสำหรับวงจรเวลา เมื่อเวลาดำเนินไปเท่ากับเวลาที่คลื่นสะท้อนเดินทางไป และกลับสิ้นสุด ส่วนของคลื่นสะท้อนต่อไปจะถูกส่งออก การหยุดลงชั่วขณะของเวลาพิเศษที่คลื่นเดินทางทำขึ้นเพื่อลดสัญญาณรบกวน (Noise) ที่ดำเนินมาจากตรวจจับวัตถุมากกว่าหนึ่งระยะ โดยเซ็นเซอร์สามารถถูกปรับให้เหมาะสมตามสภาพแวดล้อมซึ่งหมายถึง สำหรับการตรวจจับวัตถุที่อยู่ไกลเวลาการเดินทางจะนานเป็นผลให้ต้องการความถี่ต่ำในทางตรงกันข้าม ความถี่ของวงจรจะ

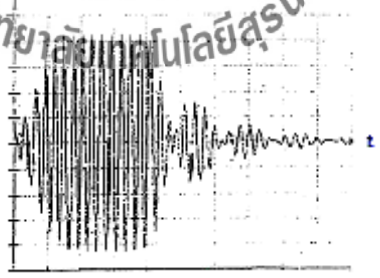
เพิ่มขึ้นเมื่อวัตถุเข้าใกล้เซ็นเซอร์ทำงานให้วงจรเวลาสั้นลง และพลังงานที่ส่งออกไปสามารถปรับในช่วงเวลาของคลื่นเสียงที่ปล่อยไป เวลาที่เพิ่มขึ้นของแอมปลิจูดเมื่อมีการสวิทซ์



รูปที่ 8

อุลตร้าโซนิคเซ็นเซอร์ , วงจรที่เปลี่ยนแปลงได้

ทรานซ์ดิวเซอร์จะถูกใช้สำหรับพลังงานที่ส่งออกไป (ดังแสดงในรูปที่9) ดังนั้นคลื่นสะท้อนเบื้องหลังสามารถควบคุมได้ด้วย การลดพลังงานในการส่งผ่านวัตถุที่อยู่ใกล้เซ็นเซอร์



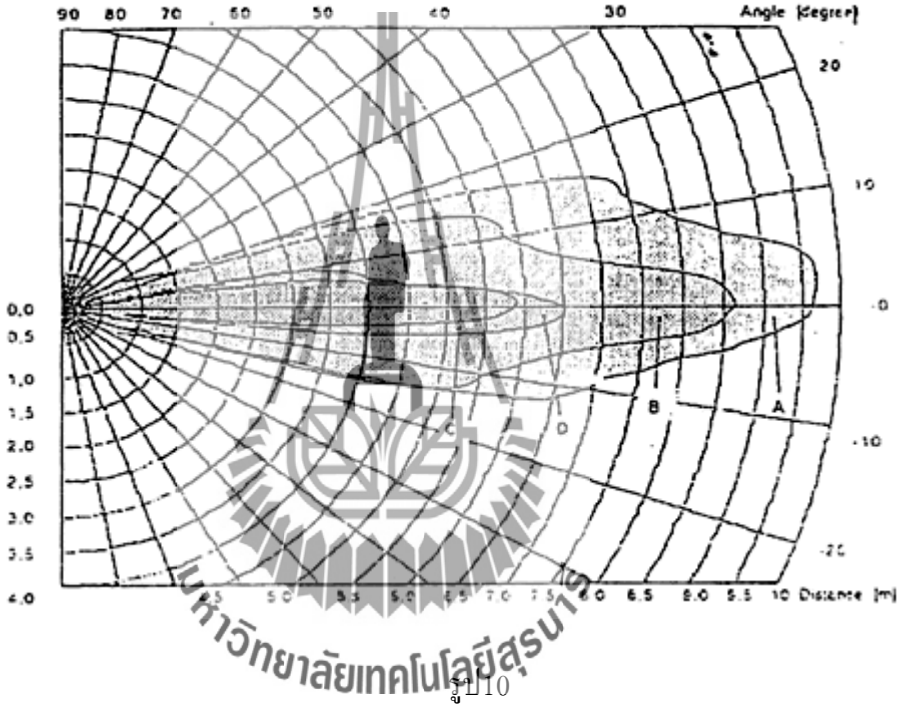
รูป9

อุลตร้าโซนิคเซ็นเซอร์ทรานซ์ดิวเซอร์

แบบชิ้นส่วนการอิมพัลซ์ (Impulse) (แตกออก)ที่170 MHz

2.2.5 การลดสัญญาณรบกวน และสภาวะการทำงาน

ผลที่เกิดจากคลื่นรบกวน และการทรอดแทรกในการประยุกต์ใช้อุลตร้าโซนิคเซ็นเซอร์คือ การตรวจจับวัตถุได้แต่ระยะที่ใกล้กับเซ็นเซอร์ และไม่สามารถตรวจจับวัตถุที่มีการสะท้อนได้ไม่ดี เนื่องจากความจริงที่ว่า คลื่นอุลตร้าโซนิคจะสะท้อนได้จากวัตถุเกือบทุกชนิด และง่ายต่อการเบียงเบน วัตถุเหล่านั้นจะทำให้สวิทซ์เปิด-ปิด เมื่อเข้าใกล้บริเวณที่เซ็นเซอร์สามารถตรวจจับได้ ดังกราฟทำนายคุณลักษณะของเซ็นเซอร์ (ดังรูปที่ 10)



รูป10

อุลตร้าโซนิคเซ็นเซอร์ , คุณลักษณะการตรวจจับ

เพื่อหาคุณลักษณะของวัตถุชนิดต่าง ๆ จะวางในตำแหน่งของวัตถุในระยะห่างเท่า ๆ กันที่มุมตั้งฉากกับแนวแกนของเซ็นเซอร์ จุดที่สวิทช์ทำงานก็จะถูกกำหนดขึ้น ตัวอย่างวัสดุที่ใช้คือ

A : แผ่นจานขนาด 700 x 700 mm. ขอบเขตที่อยู่ด้านนอกส่วนโค้งชั้นนี้โดยปกติจะไม่มีวัตถุตรวจจับได้

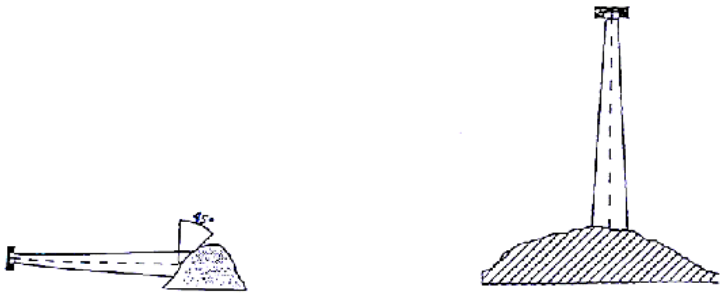
B : แผ่นจานขนาด 100 x 100 mm. แผ่นจานอ้างอิงมาตรฐานกำหนดโดยข้อมูลทางเทคนิคทั่วไป

C : ท่อพลาสติก ขนาดเส้นผ่าศูนย์กลาง 160 mm. คลุมด้วยสักหลาด ใช้เป็นตัวแทนมาตราฐาน

D : แท่นไม้ ขนาดเส้นผ่าศูนย์กลาง 25 mm. วัสดุทดสอบ เช่นระยะความปลอดภัยย้อนกลับในยานพาหนะ เพื่อให้ปราศจากปัญหาในการทำงาน จะไม่มีวัสดุอื่นใดที่ไม่ใช่เป้าหมายในขอบเขตนอกสุดในทางกลับกันวัตถุเป้าหมายต้องอยู่ภายในบริเวณพื้นที่ที่สามารถตรวจจับได้ทั้งขนาด รูปร่าง เพื่อป้องกันปัญหาการตรวจจับคลื่นเสียงพื้นผิวของวัตถุควรมีขนาดใหญ่เท่าที่จะเป็นได้ราบเรียบ และมีมุมเอียงไม่เกิน 3 “ กับแกนของเซ็นเซอร์ (ดังแสดงในรูปที่ 11 และ12)จากข้อกำหนดดังกล่าว เมื่อทำการตรวจจับวัตถุทรงกลม หรือวัตถุผิวไม่เรียบ (ของเหลว, ของผสม) ก็จะเกิดปัญหาขึ้น

รูปที่ 11

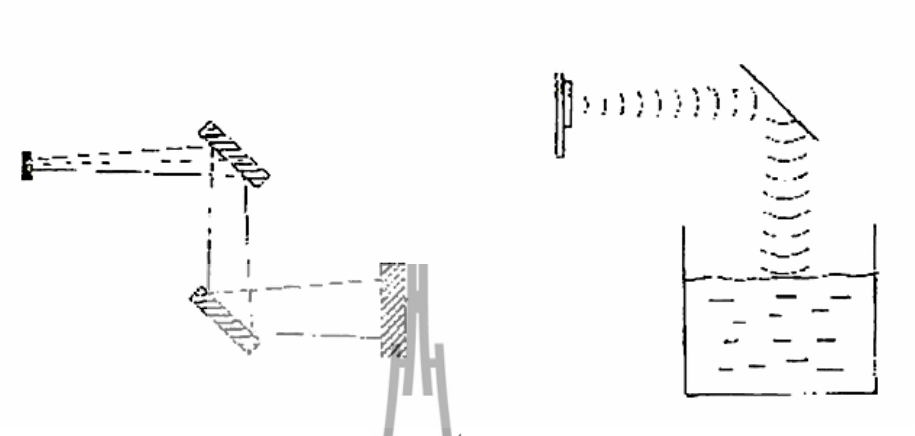
อุลตร้าโซนิคเซ็นเซอร์,พื้นราบตรง



รูป12

อุลตร้าโซนิคเซ็นเซอร์,การตรวจจับสิ่งของ

อุลตร้าซาวด์สามารถนำมาเบี่ยงเบนด้วยตัวสะท้อนอย่างง่าย ที่ทำจากวัสดุใด ๆ (ดังแสดงในรูป13) พื้นที่ตรวจจับยังคงเท่าเดิม ทำให้ใช้กับตัวสะท้อนขนาดใหญ่ได้โดยใช้ตัวเบี่ยงเบนไม่เกิน 2 ตัว ติดตั้งภายในทางเดินของคลื่นเสียงในแนวทางเดินตั้งฉากอย่างถูกต้อง



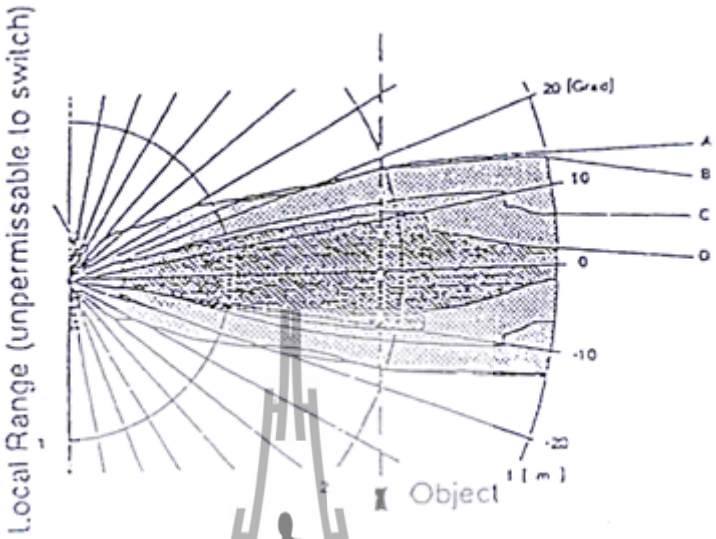
รูป13

อุลตร้าโซนิคเซ็นเซอร์ , การเบี่ยงเบนคลื่นเสียง

ด้วยวิธีนี้สามารถนำไปใช้ป้องกันป้องกันเซ็นเซอร์ จากการใช้งานในสิ่งแวดล้อมที่เป็นอันตรายต่อเซ็นเซอร์และทำให้ลดคลื่นสะท้อนที่ไม่ต้องการจากวัตถุ การรวมคลื่นเสียงผ่านม่านกั้นหรือท่อโดยปกติไม่สามารถทำได้ เนื่องจากการหักเหของแสง บนขอบ และกำแพง เพื่อหลีกเลี่ยงการสอดแทรกจากเครื่องมือที่ให้กำเนิดคลื่นเสียงอันอื่น ๆ สัญญาณที่รับได้จะถูกทดสอบความถี่ในตัวเอง วิธีการนี้ไม่สามารถทำให้สำเร็จได้เมื่อใช้เซ็นเซอร์ชนิดเดียวกัน (ทรานซ์ดิวเซอร์และความถี่ชนิดเดียวกัน) หรือมีย่านรบกวนกว้าง (เช่น ในอากาศ) สอดแทรกกัน เพื่อหลีกเลี่ยงผลกระทบระหว่างเซ็นเซอร์จะต้องมีการติดตั้งที่ระยะปลอดภัย คลื่นรบกวนกับแอมปลิจูดขนาดใหญ่ สามารถปิดกั้นเซ็นเซอร์ไม่ให้รับคลื่นสะท้อนที่อ่อนกว่าคลื่นรบกวนได้

เซ็นเซอร์บางชนิดแก้ปัญหาการสอดแทรกนี้ โดยสัญญาณเตือนที่เอ้าท์พุทที่แยกต่างหากเพื่อเป็นการชดเชยการเปลี่ยนแปลงความเร็วของเสียงจากอุณหภูมิที่ขึ้น ๆ ลง ๆ เซ็นเซอร์บางชนิดจึงรวมเอาเซ็นเซอร์อุณหภูมิเข้าไปด้วย โดยจะทำการวัดอุณหภูมิบริเวณเซ็นเซอร์ และวัตถุ (ระยะสูงสุด 6 m) ขบวนการหลาย ๆ ขบวนการของสัญญาณที่ช่วยในการเกิดคลื่นรบกวนที่เกิดขึ้นเป็นครั้งคราว เช่น คลื่นรบกวนเนื่องจากการเคลื่อนที่ของอากาศ หรือคลื่นรบกวน กับคลื่นอุลตร้าซาวด์ที่มีองค์ประกอบมาก (ในอากาศ , เครื่องจักรกล) ด้วยวิธีการนี้เอาท์พุทที่ได้จากการเซ็นเซอร์จะถูกกระตุ้นเมื่อจำนวนของคลื่นสะท้อนมีความเข้มข้นเดียวกัน และในการเดินทางเท่ากันได้ถูกรับเอาไว้ ข้อเสียคือ ความถี่ของการทำงานค่อนข้างลดน้อยลงเมื่อทำการตรวจจับวัตถุที่เคลื่อนที่ผ่านอย่าง

รวดเร็วผ่านจุดที่ทำมุมตั้งฉากกับแกนของเซ็นเซอร์ ในกรณีนี้ความเร็วของวัตถุที่ยอมให้มีได้กำหนดมาจากขนาดของวัตถุ และระยะห่างจากเซ็นเซอร์ (ดังแสดงในรูปที่ 14)



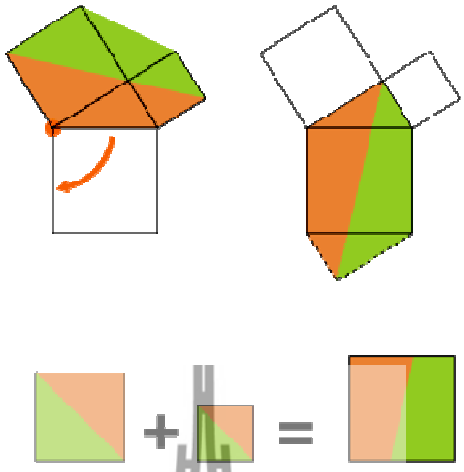
รูปที่ 14

อุลตร้าโซนิคเซ็นเซอร์, ความเร็วของวัตถุที่ยอมให้ได้

หากมีวัตถุมีรูปร่างเป็นสี่เหลี่ยมจัตุรัสขนาดความยาวของแต่ละด้านเท่ากับ 100 mm. (แผ่นจานอ้างอิงมาตราฐานส่วนโค้ง B) ในระยะห่าง 2 m จากเซ็นเซอร์พร้อมกับย่านความไว 3 m นั้น ต้องการตรวจจับด้วยเซ็นเซอร์ จะต้องมีเวลาปฏิกิริยา 280 ms การพิจารณาประยุกต์ใช้งานดังนี้ ระยะเวลาช่องวัตถุในการตรวจจับ (ดังแสดงในกลางด้านบน) : 1.24 m. เวลาน้อยที่สุดที่ภายในช่วงการทำงานของเซ็นเซอร์ (เวลาปฏิกิริยา) : 280 ms. ความเร็วสูงสุด v = s/t = 1240 mm/ 280 ms = 4.43 m/s เพื่อที่จะตรวจจับแท่งไม้ที่มีขนาดเส้นผ่าศูนย์กลาง 25 mm. (ส่วนโค้ง D) ได้ในระยะ 3 m เราใช้การคำนวณดังนี้ s = 0.19 m , t = 280 ms , v = 0.68 m/s

2.2.6 ทฤษฎีบทพีทาโกรัส Euclidean distance และความเร็วเสียง

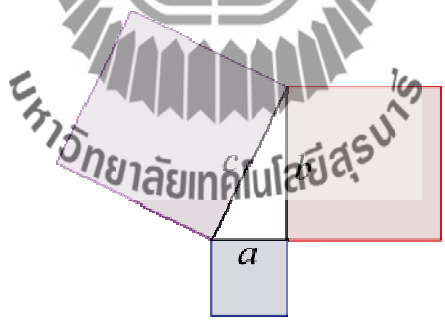
2.2.6.1 ทฤษฎีบทพีทาโกรัส แสดงความสัมพันธ์ใน เรขาคณิตแบบยุคลิด ระหว่างด้านทั้งสามของสามเหลี่ยมมุมฉาก ทฤษฎีนี้ ถูกตั้งชื่อเพื่อเป็นเกียรติแก่พีทาโกรัส นักคณิตศาสตร์ชาวกรีก แม้ว่าความจริงแล้ว ทฤษฎีได้มีการคิดค้นไว้ก่อนหน้าที่เขาจะมีชีวิตอยู่ โดยชาว อินเดีย, ชาวกรีก, ชาวจีน และ ชาวบาบิโลน



รูปที่ 15 วิธีพิสูจน์ทฤษฎีบทพีทาโกรัสของเลโอนาร์โด ดา วินชี

ทฤษฎีบทพีทาโกรัส กล่าวไว้ว่า

"ผลรวมของพื้นที่ของรูปสี่เหลี่ยมจัตุรัสบนด้านประชิดมุมฉากทั้งสอง จะเท่ากับ พื้นที่ของรูปสี่เหลี่ยมจัตุรัสบนด้านตรงข้ามมุมฉาก"



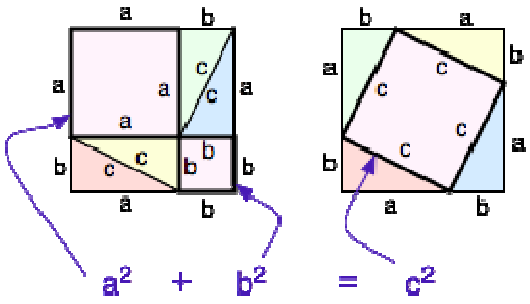
รูปที่ 16

จากรูปที่ 16 จะสังเกตว่า ผลรวมของพื้นที่ของสี่เหลี่ยมสีน้ำเงินและสีแดง จะเท่ากับ พื้นที่ของสี่เหลี่ยมสีม่วง เราสามารถเขียนทฤษฎีบทนี้ให้อยู่ในรูป สมการ

$$c^2 = a^2 + b^2$$

โดยที่ a และ b เป็นความยาวด้านประชิดมุมฉากทั้งสองของสามเหลี่ยมมุมฉาก และ c เป็นความยาวด้านตรงข้ามมุมฉาก

วิธีการพิสูจน์อีกแบบแสดงได้ดังรูปที่ 17 ด้านล่าง



รูปที่ 17

2.2.6.2 Euclidean distance เป็นค่าระยะทางระหว่างจุดสองจุด ซึ่งขยายผลมาจากทฤษฎีบทพีทาโกรัส ซึ่งมีคำนิยามดังนี้

Euclidean Distance ระหว่างจุด $P(p_1, p_2, ..., p_n)$ และ $Q(q_1, q_2, ..., q_n)$ ใน Euclidean n-Space คือ

$$\sqrt{(p_1 - q_1)^2 + (p_2 - q_2)^2 + \cdots + (p_n - q_n)^2} = \sqrt{\sum_{i=1}^{n} (p_i - q_i)^2}.$$

- one-dimension distance

  สำหรับระยะห่างใน 1 มิติระหว่าง 2 จุด $P(p_x)$ และ $Q(q_x)$ จะได้ระยะทางเป็น

  $$\sqrt{(p_x - q_x)^2} = |p_x - q_x|$$

- Two-dimension distance

  สำหรับระยะห่างใน 2 มิติระหว่าง 2 จุด $P(p_x, p_y)$ และ $Q(q_x, q_y)$, จะได้ระยะทางเป็น

  $$\sqrt{(p_x - q_x)^2 + (p_y - q_y)^2}$$

- Three-dimension distance

  สำหรับระยะห่างใน 3 มิติระหว่าง 2 จุด $P(p_x, p_y, p_z)$, และ $Q(q_x, q_y, q_z)$, จะได้ระยะทางเป็น

  $$\sqrt{(p_x - q_x)^2 + (p_y - q_y)^2 + (p_z - q_z)^2}.$$

- N-dimension distance

  สำหรับระยะห่างใน 3 มิติระหว่าง 2 จุด $P(p_1, p_2, p_3)$ และ $Q(q_1, q_2, q_3)$, จะได้ระยะทางเป็น

  $$\sqrt{(p_1 - q_1)^2 + (p_2 - q_2)^2 + ... + (p_n - q_n)^2}.$$

โดยในโครงงานนี้ใช้ Euclidean distance ที่ 2 มิติ คือใช้ Euclidean distance 2

2.2.6.3 อัตราเร็วเสียง

เราอาจหา อัตราเร็วเสียงได้จาก

$v = s/t$ หรือ $v = f\lambda$

เมื่อ v = อัตราเร็ว (m/s) s = ระยะทางที่เสียงเคลื่อนที่ได้ (m)

t = เวลา (s) f = ความถี่เสียง (Hz)

λ = ความยาวคลื่น (m)

ในโครงงานนี้ ต้องการได้ระยะห่างจากวัตถุจริง s' เพราะเสียงเดินทางไปกลับ

$$s' = \frac{vt}{2}$$

ปัจจัยที่มีผลต่ออัตราเร็วเสียง

1. ความหนาแน่นของตัวกลาง อัตราเร็วในตัวกลางที่มีความหนาแน่นมากกว่า จะมีค่ามากกว่าในตัวกลางที่มี ความหนาแน่นน้อยกว่า

ตารางแสดงอัตราเร็วของเสียงในตัวกลางต่าง ๆ ที่อุณหภูมิ 25 องศาเซลเซียส

ตัวกลาง	อัตราเร็ว (m/s)
อากาศ	346
น้ำ	1,498
น้ำทะเล	1,531
เหล็ก	5,200

2. อุณหภูมิ อัตราเร็วเสียง จะแปรผันตรงกับรากที่ 2 ของอุณหภูมิเคลวิน เพราะอุณหภูมิสูงขึ้นจะทำให้โมเลกุล มีพลังงานจลน์มากขึ้น การอัดตัวและขยายตัวเร็ว ทำให้เสียงเคลื่อนที่ได้เร็วขึ้นด้วย

จึงได้ว่า $v \propto \sqrt{T}$

และสำหรับในอากาศนั้น เราสามารถหาอัตราเร็วเสียงที่อุณหภูมิต่าง ๆ ได้ โดยอาศัยสมการ $v = v_0 + 0.6\,t$ หรือ $v = 331 + 0.6\,t$

เมื่อ vo = อัตราเร็วเสียงที่อุณหภูมิ 0 ˚C = 331 m/s

t = อุณหภูมิ (˚C)

2.2.7 บอร์ด Arduino mega 2560 R3 และ ultrasonic sensor(hc-sr04)

ในโครงงานนี้เราได้เลือกใช้ บอร์ด Arduino mega 2560 R3 เป็นตัวทำงานหลักของการประมวลผล เชื่อมต่อกับคอมพิวเตอร์ ซึ่งมี module ที่จำเป็นคือ ultrasonic sensor (hc-sr04) และใช้โปรแกรม arduino-1.0.4 ในการควบคุม การเชื่อมต่อ module กับบอร์ด Arduino ใช้คำสั่ง จะอธิบายในหัวข้อต่อไป

2.3 เครื่องมือที่ใช้ในการพัฒนา

- โปรแกรม arduino-1.0.4
- บอร์ด Arduino mega 2560 R3 เป็นตัวควบคุม และ module ultrasonic sensor (hc-sr04) เป็นตัวรับค่าและส่งค่า

2.4 รายละเอียดโปรแกรม

Input/output specification:

- input เป็นคำสั่งเพื่อความสามารถการส่งสัญญาณให้กับ sensor หรือคำสั่งเพื่อให้ sensor สามารถบอกความแกน x และ y ได้
- output เป็นการระบุว่าสิ่งนั้นเป็นมนุษย์ หรือสัตว์

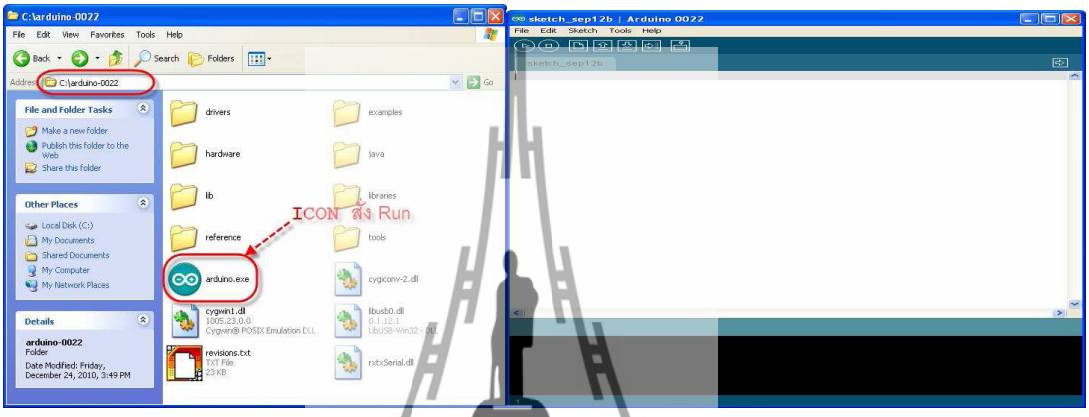
Functional specification

- function สำหรับการรับส่งข้อมูลผ่าน port pin ต่างๆ ในการแสดงผล และควบคุมการทำงาน อุปกรณ์ module เพิ่มเติม เช่นหลอด LED ,SERVO MOTOR , จอ LCD Module Nokia 5110 หรือ Buzzer Module
- function สำหรับการระบุสิ่งที่ผ่านเข้ามาเป็น มนุษย์หรือสัตว์
- function สำหรับการควบคุมsensor

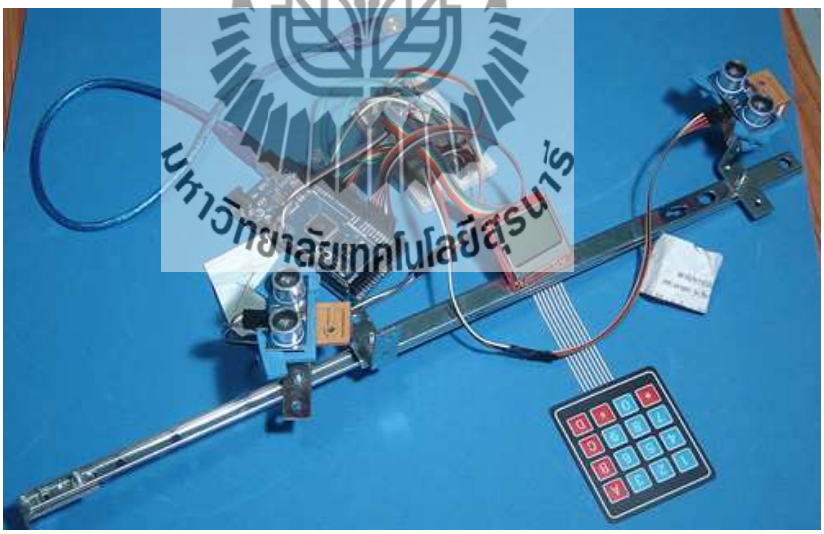
2.4.1 การออกแบบโปรแกรม

arduino-1.0.4 เป็นซอร์ฟแวร์สำหรับเชื่อมต่อ บอร์ด Arduino mega 2560 R3 กับ module ultrasonic sensor (hc-sr04) และmodule อื่นๆ ซึ่งติดต่อกับคอมพิวเตอร์ผ่านทาง Serial Port ซอร์ฟแวร์มีลักษณะคล้ายกับรีโมทคอนโทรลที่สามารถส่งคำสั่งเพื่อควบคุม

module ultrasonic sensor ทำหน้าที่วัด และมีบอร์ด Arduino mega 2560 R3 เป็นส่วนของโปรแกรมควบคุม และแสดงผลระบุสิ่งนั้นเมื่อผ่าน ตัว sensor ซึ่งแสดงดังรูปที่ 18



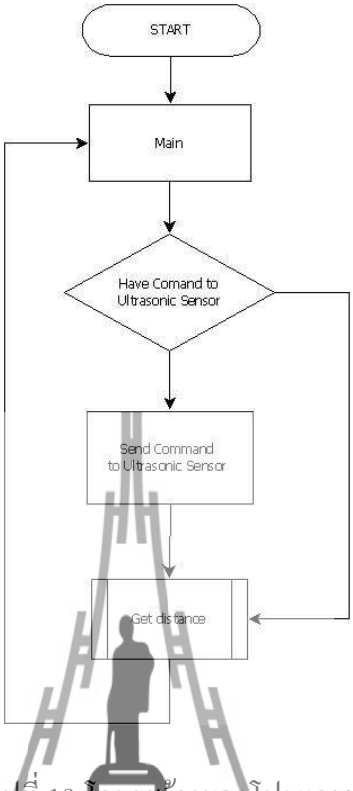
ก. รูปของ icon โปรแกรม arduino-1.0.4 ข. รูปของน่าตาโปรแกรม arduino-1.0.4



ค. วงจรรวมของระบบการทำงานของ module sensor และmodule ต่างๆ

รูปที่ 18 (ก ข และค)

การทำงานของโปรแกรม อุปกรณ์ module sensor และ module ต่างๆ ตามลำดับ

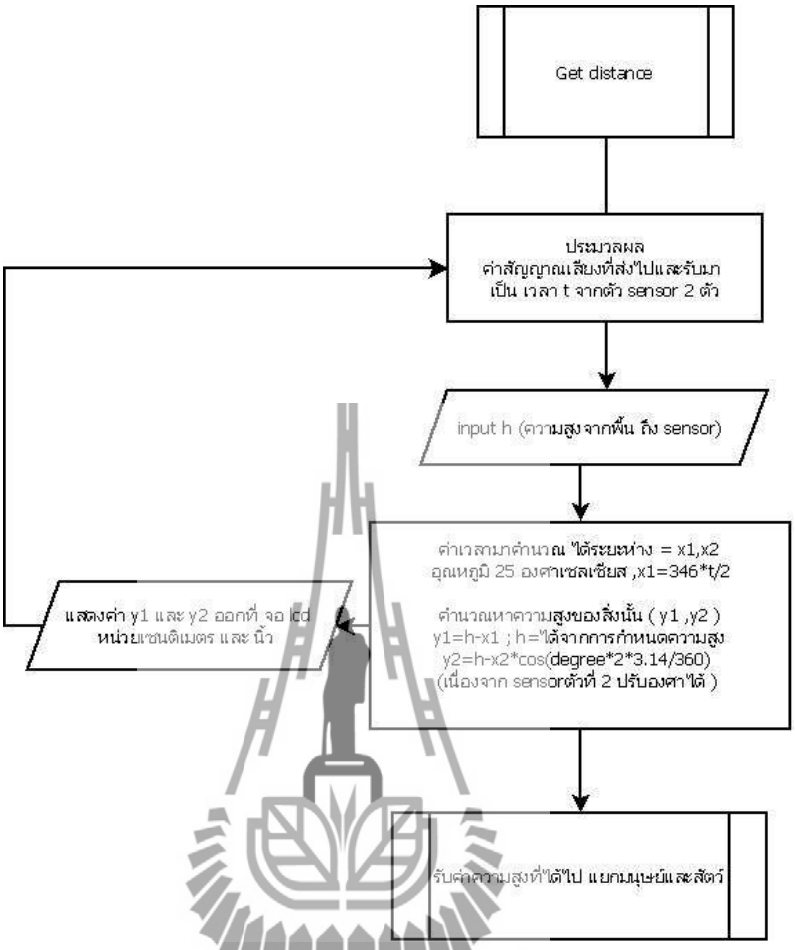


รูปที่ 19 โครงสร้างของโปรแกรม

โครงสร้างการทำงานของโปรแกรมจะเป็นการตรวจสอบว่ามี Input จากผู้ใช้ว่าต้องการส่งคำสั่งไปควบคุม Ultrasonic Sensor หรือไม่ และคอยทำการ Update ค่าของ distance (ระยะทาง ของสัญญาณเสียงที่ส่งและรับมา)

2.4.2 การหาระยะทางความห่างของ Ultrasonic Sensor กับสิ่งที่ผ่านเข้ามา

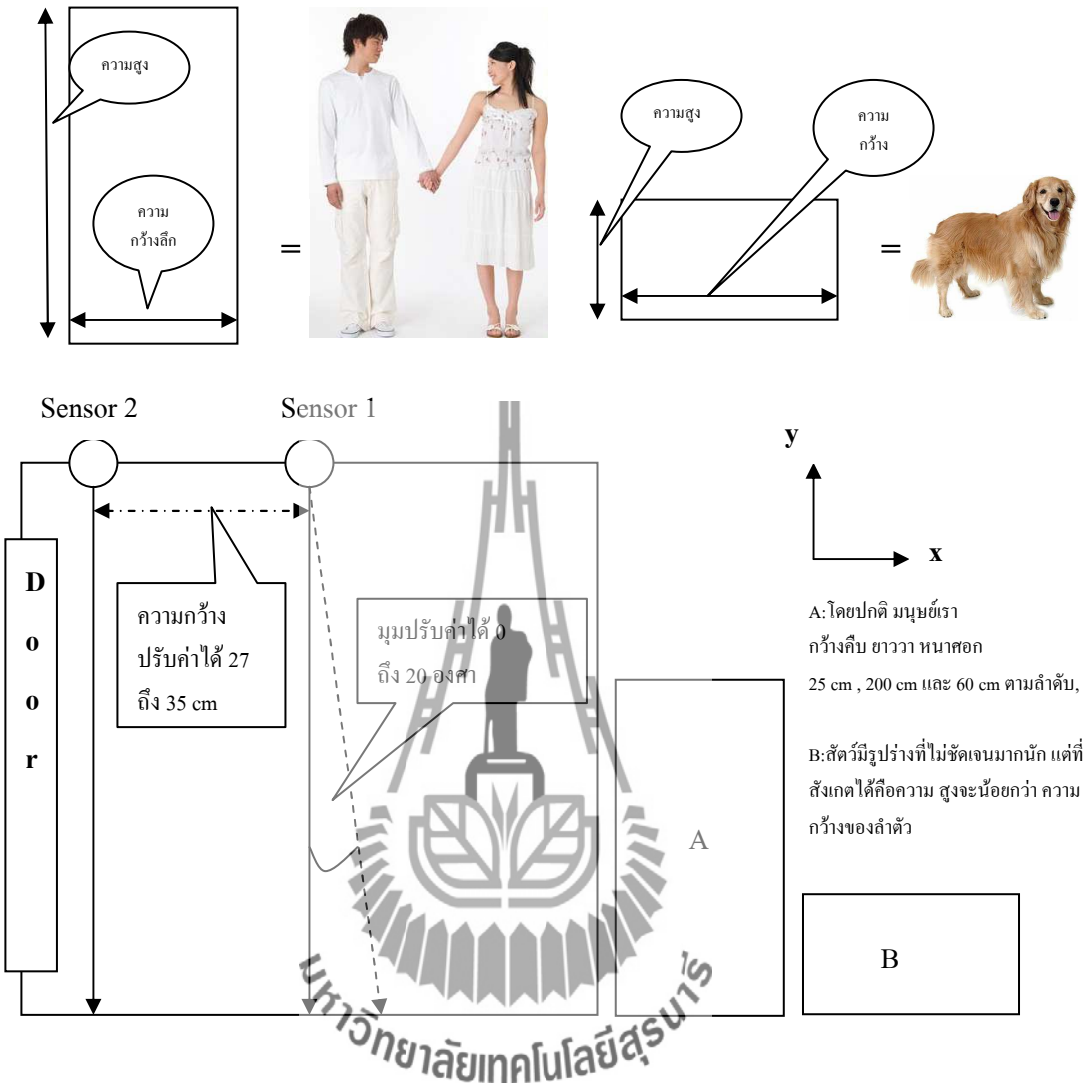
วิธีการที่ผู้พัฒนาโครงงานใช้สำหรับการหาระยะทาง คือ ทำการจับเวลาขอสัญญาณเสียงที่ปล่อยไปตกกระทบใส่วัตถุ หรือสิ่งของ แล้วสะท้อนกลับมาที่ ตัวเซนเซอร์ ค่าเวลาที่ได้ส่งมายัง board arduino นำค่าเวลามาคำนวณหาระยะทางของเซนเซอร์กับวัตถุได้ ซึ่งสามารถ หาความสูงของวัตถุนั้นได้ ถ้าหาก sensor อยู่แนวตั้งฉากกับพื้น หรือเอียงทำมุมองศากับแนวตั้งฉากกับพื้นได้ โดยกำหนด input ในตัวโปรแกรมเอง หรือสร้างโปรแกรมให้รับ input จากภายนอก เช่น keypad เป็นต้น ซึ่งวัดจากความสูงของพื้นถึงตัวเซนเซอร์



รูปที่ 20 โครงสร้างของ Function การวัดความสูงของวัตถุหรือสิ่งของ

2.4.3 การแยกประเภทมนุษย์ และสัตว์

ในส่วนของการออกแบบซึ่งยังคงเป็นการทำงานหลักของ board arduino อยู่ แต่เพิ่มฟังก์ชันการทำงานอีกอย่างคือ การเทียบความสูงของทั้ง 2 ตัวเซนเซอร์ มีแนวคิดมาจาก มนุษย์เรา มีร่างกายที่ตั้งฉากกับพื้น ความสูงมากกว่า ความกว้างของร่างกาย ส่วนสัตว์เลื้อยคลานเกือบทุกชนิด มีร่างกาย ที่ขนานกับพื้น ความสูงน้อยกว่า ความกว้างของร่างกาย ดังรูปที่ 19



รูปที่ 21 โครงสร้างของมนุษย์และสัตว์ ในการผ่าน sensor ทั้งสองตัว

ซึ่ง Y1=h-sensor2 คือความสูงของวัตถุ

Y2=h-sensor1xcos(2x3.14xองศา/360) ซึ่งเป็นมุม radian ในการทำงานในโปรแกรม

X=ความห่างของ sensor ทั้ง 2 ตัว สามารถปรับความละเอียดการแยกวัตถุได้ โดยถ้าน้อยลงจะวัดสัตว์ที่มีขนาดกว้างไม่ต่ำเกิน 27 cm ได้ หรือมาขึ้นก็จะสามารถทำให้การผ่านเข้าประตูสะดวกมากขึ้น เพราะความกว้างมากขึ้น ทำให้สัตว์บางชนิดผ่านเข้ามาได้

ซึ่งมีหลักการทำงานดังนี้

- **กรณีที่ วัตถุความสูงมากกว่า 120 cm**

  Sensor ตัวที่ 1 และ ตัวที่ 2 ทำงาน ถ้าตัวใดตัวหนึ่งวัดความสูงวัตถุ เกิน 120 cm จะทำการแสดงผล “open” ที่จอ lcd (หรือต่อกับ module อื่นๆ ให้สามารถสื่อสารได้)

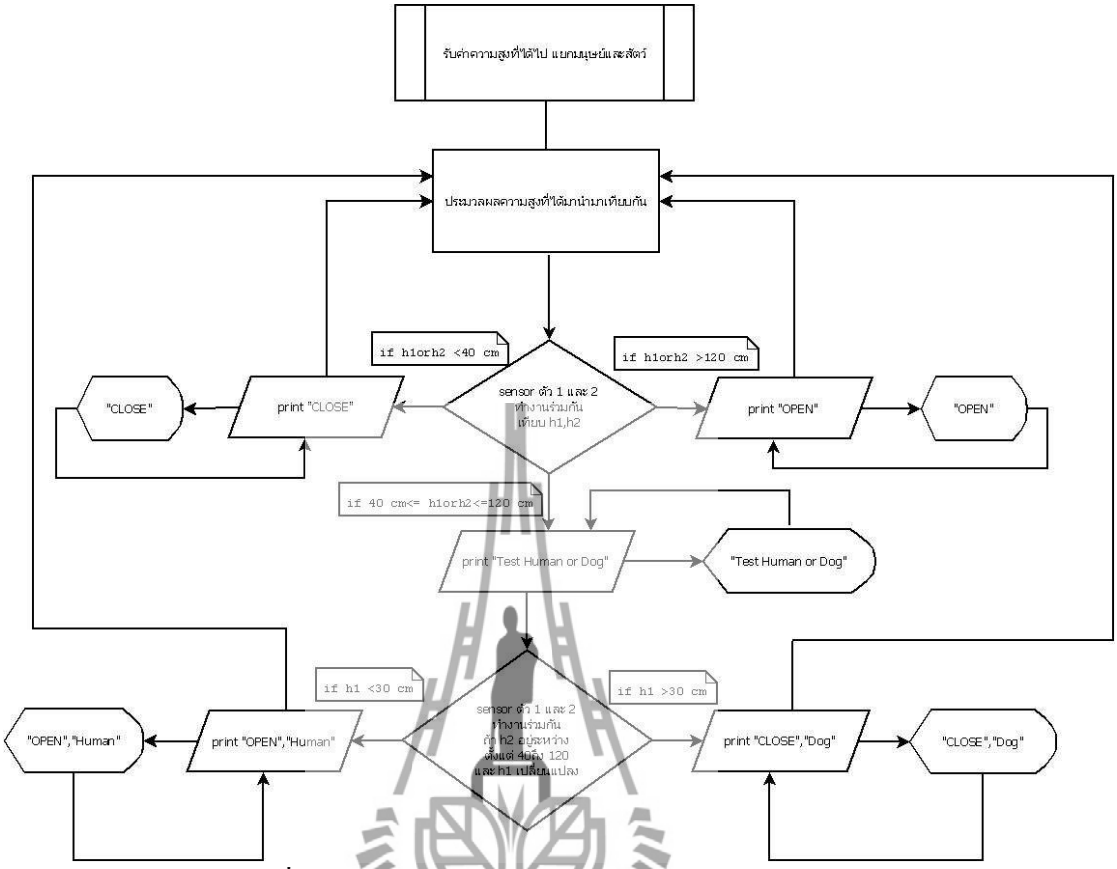
- **กรณีที่ วัตถุความสูงต่ำกว่า 40 cm**

  Sensor ตัวที่ 1 และ ตัวที่ 2 ทำงาน ถ้าตัวใดตัวหนึ่งวัดความสูงวัตถุ ต่ำกว่า 40 cm จะทำการแสดงผล “Close” ที่จอ lcd (หรือต่อกับ module อื่นๆ ให้สามารถสื่อสารได้)

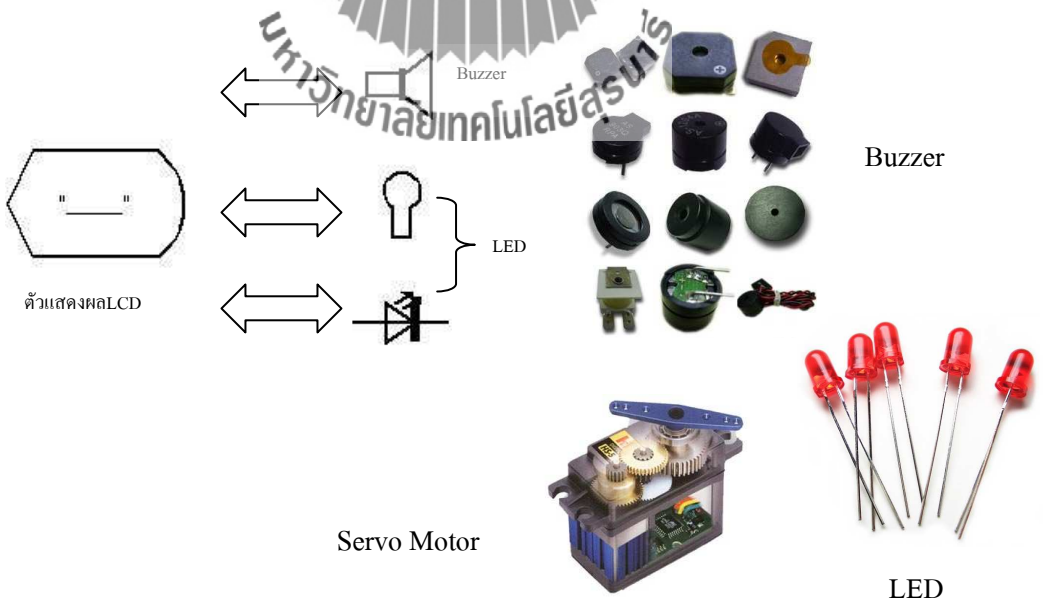
- **กรณีที่ วัตถุความสูง อยู่ระหว่าง 40 ถึง 120**

  แยก กรณีย่อยได้อีก 2 กรณี คือ

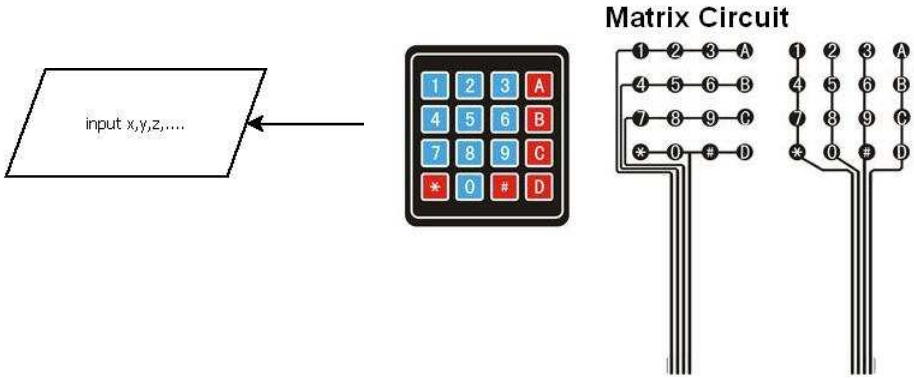
  1. sensor ตัวที่ 2 (ตัวที่ติดกับประตูที่สุด) วัดความสูงได้ช่วงระหว่าง 40-120 cm sensor ตัวที่ 1 (ตัวที่ห่างจากประตูมากสุด) วัดความสูงได้ มากกว่า 30 cm จะทำการแสดงผล “CLOSE” พร้อมกับระบุว่าเป็นสุนัข “Dog” ที่จอ lcd (หรือต่อกับ module อื่นๆ ให้สามารถสื่อสารได้)
  2. sensor ตัวที่ 2 (ตัวที่ติดกับประตูที่สุด) วัดความสูงได้ช่วงระหว่าง 40-120 cm sensor ตัวที่ 1 (ตัวที่ห่างจากประตูมากสุด) วัดความสูงได้ น้อยกว่า 30 cm จะทำการแสดงผล “OPEN” พร้อมกับระบุว่าเป็นมนุษย์ “Human” ที่จอ lcd (หรือต่อกับ module อื่นๆ ให้สามารถสื่อสารได้)



รูปที่ 22 โครงสร้างของ Function การแยกมนุษย์และสัตว์



รูปที่ 23 module output ที่สามารถทำงานร่วมกับ function ดังกล่าวมา ในการสื่อสาร สามารถแทนที่ กับตัวแสดงผล LCD ได้



รูปที่ 24 module input นอกเหนือจาก ultrasonic sensor hc-sr04 ที่สามารถทำงานร่วมกับ function ดังกล่าวมา สามารถเพิ่มเติมแทน input ในโปรแกรมได้

2.5 Code ภาษา arduino (คล้ายกับ ภาษา C++) ที่ได้มาจากการออกแบบโปรแกรม

ซึ่งมีความสัมพันธ์ board arduino กับการสื่อสารของอุปกรณ์ให้ทำงานร่วมกัน

```
// LCD5110_NumberFonts (C)2013 Henning Karlsen
// web: http://www.henningkarlsen.com/electronics
// This program is a demo of the included number-fonts
// and how to use them.
// This program requires a Nokia 5110 LCD module.
// It is assumed that the LCD module is connected to
// the following pins using a levelshifter to get the
// correct voltage to the module.
// SCK - Pin 8
// MOSI - Pin 9
// DC - Pin 10
// RST - Pin 11
// CS - Pin 12
```

ส่วนของ ผู้ร่วมพัฒนาโปรแกรม ซึ่งทางเรานำมาพัฒนาต่อยอดแก้ไขให้สามารถใช้กับโครงงานนี้ได้ ทำงานร่วมกับจอ Nokia 5110 LCD module

```
#include <LCD5110_Basic.h>              // นำเข้า function Lcd5110
//keyyyyyyyyyyyyyyyypaddddddddddddddddddd
#include <Keypad.h>                     // นำเข้า function keypad
const byte ROWS = 4;                    //four rows 4 แถว
const byte COLS = 4;                    //four columns 4 คอลัม
//define the cymbols on the buttons of the keypads
char hexaKeys[ROWS][COLS] = {
{'1','2','3','A'},
{'4','5','6','B'},
{'7','8','9','C'},
{'*','0','#','D'}
};
byte rowPins[ROWS] = {14, 15, 16, 17}; //connect to the row pinouts of the key
byte colPins[COLS] = {18, 19, 20, 21}; //connect to the column pinouts of the
//initialize an instance of class NewKeypad
Keypad kpd = Keypad( makeKeymap(hexaKeys), rowPins, colPins, ROWS, COLS);
//keyyyyyyyyyyyyyyyypaddddddddddddddddddd
/*LCD5110 myGLCD(8,9,10,11,12);*/  // pin เดิมของผู้ร่วมพัฒนา
LCD5110 myGLCD(3,4,5,6,7); // กำหนด pin ใหม่สำหรับทำโครงงานชิ้นนี้
extern uint8_t SmallFont[];
extern uint8_t MediumNumbers[];
extern uint8_t BigNumbers[];
//codenew2-------ultrasonic
const int triggerPin = 8;
const int echoPin = 9;
const int triggerPin2 = 10;
const int echoPin2 = 11;
```

เซตค่าพารามิเตอร์สำหรับ keypad กำหนด pin ที่เชื่อต่อกับ board arduino

เซตค่าพารามิเตอร์สำหรับ Nokia 5110 LCD กำหนด pin ที่เชื่อต่อกับ board arduino

เซตค่า พารามิเตอร์ สำหรับ ultrasonic sensor กำหนด pin ที่ เชื่อต่อกับ board arduino

```
void setup()
{
myGLCD.InitLCD();
pinMode(triggerPin, OUTPUT);
pinMode(echoPin, INPUT);
pinMode(triggerPin2, OUTPUT);
pinMode(echoPin2, INPUT);
pinMode(12, OUTPUT);
}
void loop()
{
char customKey1 = kpd.getKey();          // รับค่าจากkey นำมาไว้ที่ตัวแปร customKey1
if(customKey1=='A'){                     //กำหนดเงื่อนไขถ้ามีกดkeypad  'A'
//keeeeeeeeeeeeeeeyyyyyyyyyyyyyyy
int a;
int b;
int d;
int g;
int h;
int q;
int r;
int customKey ;
do{
myGLCD.clrScr();                         // คำสั่งล้างหน้าจอ
myGLCD.setFont(SmallFont);               //กำหนดขนาด font ให้มีขนาดเล็ก
myGLCD.print("New 3No.(cm.)!", CENTER, 8);//เขียนข้อความไว้ที่ตำแหน่งกลาง และลงมา8 จุด
myGLCD.print("Door's height", CENTER, 16);
delay(50);                               // กำหนดค่า .ให้หยุดเป็น เวลา 50/1000 วินาที
customKey = kpd.getKey();                // นำค่าที่ได้จากการกด keypad มาไว้ที่ customKey
a=customKey-48;                          // คำนวณ ค่าที่ได้มา-48 แล้วเก็บไว้ที่ a
```

ส่วนการเตรียมการเริ่มต้นใช้งาน LCD5510 ,ultrasonic sensor กำหนด pin ขาที่ 12ที่เชื่อต่อกับ board arduino

ส่วนการทำงานหลัก ที่มี module LCD5510 ,ultrasonic sensor กำหนด pin ขาที่ 12, keypad ที่เชื่อต่อกับ board arduino

เซตค่าตัวแปรต่างๆ ให้เป็นตัวแปรจำนวนเต็ม int หรือ integer

ส่วนของฟังก์ชัน if และ do while รับค่าจาก keypad และแสดงผลจอ LCD

```
if(a>-48){                          //กำหนดเงื่อนไข ถ้า a มีค่า มากกว่า -48
myGLCD.printNumI(a , 6, 0);         // ให้ส่งค่าไปที่ lcd 5110 ให้แสดงผลค่า a ที่lcd
}
}while(a<0);                        // ถ้า a <0 เป็นจริงก็วนลูปที่เดิม ถ้าเป็นเท็จก็ออกจากลูปไป
                                    คำสั่งถัดไป ทำอย่างนี้จดครบคำสั่งทั้งหมด
 do{
customKey = kpd.getKey();
b=customKey-48;
if(b>-48){
myGLCD.printNumI(b , 12, 0);
}
}while(b<0);
do{
customKey = kpd.getKey();
d=customKey-48;
if(d>-48){
myGLCD.printNumI(d , 18, 0);
delay(300);
}
}while(d<0);
do{
myGLCD.clrScr();
myGLCD.setFont(SmallFont);
myGLCD.print("2No. far(cm.)!", CENTER, 8);
myGLCD.print("2 ultrasonic", CENTER, 16);
customKey = kpd.getKey();
g=customKey-48;
if(g>-48){
myGLCD.printNumI(g , 12, 0);
}
```

```
}while(g<0);
do{
customKey = kpd.getKey();
h=customKey-48;
if(h>-48){
myGLCD.printNumI(h , 18, 0);
delay(300);
}
}while(h<0);
do{
myGLCD.clrScr();
myGLCD.setFont(SmallFont);
myGLCD.print("2No.degree(')!", CENTER, 8);
myGLCD.print("ultrasonic t", CENTER, 16);
customKey = kpd.getKey();
q=customKey-48;
if(q>-48){
myGLCD.printNumI(q , 12, 0);
}
}while(q<0);
do{
customKey = kpd.getKey();
r=customKey-48;
if(r>-48){
myGLCD.printNumI(r , 18, 0);
delay(300);
}
}while(r<0);
```

```
int r1=a;
int r2=b;
int r3=d;
int c=r1*100+r2*10+r3;
int o=g*10+h;
float plottx=o+((c/cos(2*3.14*5/360))*sin(2*3.14*5/360));
float degree=q*10+r;
delay(100);
if(c&&o&&degree){                    //กำหนดเงื่อนไขถ้าได้รับค่าครบแล้ว
myGLCD.clrScr();
myGLCD.setFont(SmallFont);
myGLCD.print("far degree height", CENTER, 0);
myGLCD.print("(cm.)( ' )(cm.) ", CENTER, 8);
myGLCD.printNumI(c, RIGHT , 24);
myGLCD.printNumI(degree, CENTER , 24);
myGLCD.printNumI(o, LEFT , 24);
for (int i=0; i<12; i++)             //ทำลูป คำสั่ง for สำหรับ "\\" และดี delay 400/1000 วินาที
{
myGLCD.print("\\", 6+(i*6), 40);
delay(400);
}
for (int i=0; i<16; i++)
{
myGLCD.clrScr();
myGLCD.setFont(MediumNumbers);       //ส่วนของจอ lcd แสดงผลค่าที่เป็น string ,int
myGLCD.printNumI(c, RIGHT , 24);
myGLCD.printNumI(plottx, LEFT , 24);
myGLCD.setFont(SmallFont);
myGLCD.print("Door's", RIGHT, 8);
myGLCD.print("Height", RIGHT, 16);
```

การคำนวณ
นำค่าจาก keypad มา
คำนวณ ได้ความสูง
องศาของ sensorตัวที่ 1
และระยะความกว้างของ
สุนัข

```
myGLCD.print("a Dog", LEFT, 0);
myGLCD.print("Minimum", LEFT, 8);
myGLCD.print("width", LEFT, 16);
myGLCD.setFont(SmallFont);
myGLCD.print("(cm.)", RIGHT, 40);
myGLCD.print("(cm.)", LEFT, 40);
myGLCD.print(">>", 6+(i*6), 0);
delay(450);
myGLCD.clrScr();
}
}
//keeeeeeeeeeeeeeeyyyyyyyyyyyyyyy
/*for (int i=0; i<=10000; i++)
{
myGLCD.setFont(MediumNumbers);
myGLCD.printNumF(float(i)/3, 2, RIGHT, 0);
myGLCD.setFont(BigNumbers);
myGLCD.printNumI(i, RIGHT, 24);
}
myGLCD.setFont(SmallFont);
myGLCD.print("| |", CENTER, 16);
delay(500);
for (int i=0; i<12; i++)
{
myGLCD.print("\\", 6+(i*6), 16);
delay(500);
}
myGLCD.clrScr();*/
//codenew2-------ultrasonic
if(c){
```

ส่วนของ code เดิมของผู้ร่วมพัฒนา คำสั่งเชื่อมกับ จอ Lcd 5110 ซึ่งเราได้พัฒนาโปรแกรมขึ้นมาใหม่ สำหรับโครงงานนี้

```
do{
//เป็นการกำหนด การส่งและรับ สัญญาณเสียง Ultrasonic sensor Hc-Sr04
// กำหนดพารามิเตอร์สำหรับ สัญญาณเสียง Ultrasonic sensor Hc-Sr04
// establish variables for duration of the ping,  // and the distance result in inches and centimeters:
float duration, inches, cm, inches2, cm2;
// The device is triggered by a HIGH pulse of 2 or more microseconds.
// Give a short LOW pulse beforehand to ensure a clean HIGH pulse:
delay(500);
digitalWrite(triggerPin, LOW);
delayMicroseconds(2);
digitalWrite(triggerPin, HIGH);
delayMicroseconds(5);
digitalWrite(triggerPin, LOW);
// The echo pin is used to read the signal from the device: a HIGH
// pulse whose duration is the time (in microseconds) from the sending
// of the ping to the reception of its echo off of an object.
duration = pulseIn(echoPin, HIGH);
// convert the time into a distance
inches = microsecondsToInches(duration);
cm = microsecondsToCentimeters(duration);
delay(1200);
digitalWrite(triggerPin2, HIGH);
delayMicroseconds(5);
digitalWrite(triggerPin2, LOW);
delayMicroseconds(2);
digitalWrite(triggerPin2, HIGH);
delayMicroseconds(5);
digitalWrite(triggerPin2, LOW);
// The echo pin is used to read the signal from the device: a HIGH
// pulse whose duration is the time (in microseconds) from the sending
```

```
// of the ping to the reception of its echo off of an object.
duration = pulseIn(echoPin2, HIGH);
// convert the time into a distance
inches2 = microsecondsToInches(duration);
cm2 = microsecondsToCentimeters(duration);
float inchess=c*0.39-inches;
float inchess2=c*0.39-inches2;
float t=c-cm*cos(2*3.14*degree/360);      //กำหนดค่า t คือความสูงของวัตถุเมื่อ ตัว sensor ตัวที่ 1
                                          //เอียงทำมุมกับแนวตั้งฉากพื้น
float t2=(c-cm2);
/*for (int i=0; i<=10000; i++)
{
myGLCD.setFont(MediumNumbers);
myGLCD.printNumF(float(i)/3, 2, RIGHT, 0);
myGLCD.setFont(BigNumbers);*/
/* if(cm<20){
myGLCD.print("New Number Please!", CENTER, 32);
for (int i=0; i<12; i++)
{
myGLCD.print("\\", 6+(i*6), 40);
delay(200);
}
myGLCD.clrScr();
}
*/
myGLCD.clrScr();
myGLCD.setFont(SmallFont);
myGLCD.printNumI(t, LEFT, 0);
myGLCD.print("CM.", 28, 0);
myGLCD.print("{t}", RIGHT, 0);
```

ส่วนของ code เดิมของผู้ร่วมพัฒนา
คำสั่งเชื่อมกับ จอ Lcd 5110
ซึ่งเราได้พัฒนาโปรแกรมขึ้นมาใหม่
สำหรับโครงงานนี้

```
myGLCD.printNumI(inchess, LEFT, 8);
myGLCD.print("Inc.", 28, 8);
myGLCD.print("{t}", RIGHT, 8);
myGLCD.printNumI(t2, LEFT, 24);
myGLCD.print("CM.", 28, 24);
myGLCD.print("{t2}", RIGHT, 24);
myGLCD.printNumI(inchess2, LEFT, 32);
myGLCD.print("Inc.", 28, 32);
myGLCD.print("{t2}", RIGHT, 32);
myGLCD.print("\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\", 0, 40);
delay(200);
// myGLCD.clrScr();
//กำหนดเงื่อนไขสำหรับ แยกมนุษย์ และสัตว์ ถ้า sensor 1 หรือ 2
//มีค่าน้อยกว่า 60 cm ให้ปิดประตู ในที่นี้จะแสดงผล lcd 5110
//ข้อความว่า “Close” ที่ ตำแหน่งกลาง ลงมา 40 จุด
if(t2<60||t<60){
myGLCD.print(">>>CLOSE<<<", CENTER, 40);
delay(150);
}
if(t2>120||t>120){ //กำหนดเงื่อนไขสำหรับ แยกมนุษย์ และสัตว์ ถ้า sensor 1 หรือ 2 ถ้ามาก
                  //ว่า 120 cm ให้เปิดประตู ในที่นี้จะแสดงผล lcd 5110
                  //ข้อความว่า “Human”,”OPEN”  ที่ ตำแหน่งกลาง ลงมา 16, 40 จุด ตามลำดับ
myGLCD.print(">>>>>Human<<<<<", CENTER, 16);
myGLCD.print(">>>OPEN<<<", CENTER, 40);
delay(300);
}
//กำหนดเงื่อนไขสำหรับ แยกมนุษย์ และสัตว์ ถ้า sensor ตัวที่ 2 (อยู่ติดกับประตูที่สุด) วัดได้ความ
//สูง ระหว่าง 40 ถึง 120 จะทำการตรวจสอบ มนุษย์และสัตว์
else if(t2>=40&&t2<=120){
myGLCD.print("Test Human&DOG", CENTER, 16);
```

```
delay(500);
//กำหนดเงื่อนไขสำหรับ แยกมนุษย์ และสัตว์ ถ้า sensor ตัวที่ 1 (อยู่ห่างประตูที่สุด) วัดความสูงน้อย
//กว่า 40 แสดงว่าเป็น มนุษย์ ให้เปิดประตู ในที่นี้จะแสดงผล ผ่านจอ lcd  “Human”,”OPEN”
if(t<40){
myGLCD.print(">>>>>Human<<<<<", CENTER, 16);
myGLCD.print(">>>OPEN<<<", CENTER, 40);
delay(300);
}
//กำหนดเงื่อนไขสำหรับ แยกมนุษย์ และสัตว์ ถ้า sensor ตัวที่ 1 (อยู่ห่างประตูที่สุด) วัดความสูงมาก
//กว่าหรือเท่ากับ 40 cm แสดงว่าเป็นสัตว์ ให้เปิดประตู ในที่นี้จะแสดงผล ผ่านจอ lcd
“Dog”,”Close”
else if(t>=40){
myGLCD.print(">>>THE DOG!<<<", CENTER, 16);
myGLCD.print(">>>CLOSE<<<", CENTER, 40);
//ส่วนของการทำงานของหลอด led ที่ขา pin ที่ 12 ของ arduino board .ให้ติดเมื่อเป็น high ให้ดับ
//เมื่อเป็น low และหยุดระหว่างการสลับ high low เวลา 150/1000 วินาที
digitalWrite(12, HIGH); // set the LED on
delay(150);
digitalWrite(12, LOW); // set the LED off
delay(150);
digitalWrite(12, HIGH); // set the LED on
delay(150);
digitalWrite(12, LOW); // set the LED off
delay(150);
digitalWrite(12, HIGH); // set the LED on
delay(150);
digitalWrite(12, LOW); // set the LED off
delay(150);
}
}
```

```
customKey1 = kpd.getKey();
}while(customKey1=='1'||customKey1=='2'||customKey1=='3'||customKey1==!'||customKey1=='6
'||customKey1==!'B'||customKey1=='7'||customKey1=='8'||customKey1==
||customKey1=='0'||customKey1=='#'||customKey1==!'D');
}
//เป็นฟังก์ชันเพิ่มเติมสำหรับควบคุมการเปิดปิดประตู แต่ในที่นี้จะแสดงผลผ่านหน้าจอ lcd
//ถ้ากดkeypad 'B' จะ ปิดประตูชั่วคราว แต่จะแสดงผลว่า "Temporarily closed"
}
if(customKey1=='B'){
myGLCD.clrScr();
do{ myGLCD.print("Temporarily", CENTER, 16);
myGLCD.print("closed!", CENTER, 24);
delay(150);
customKey1 = kpd.getKey();
}while(customKey1=='1'||customKey1=='2'||customKey1=='3'||customKey1==!'||customKey1=='6
'||customKey1==!'B'||customKey1=='7'||customKey1=='8'||customKey1==
||customKey1=='0'||customKey1=='#'||customKey1==!'D');
}
//เป็นฟังก์ชันเพิ่มเติมสำหรับควบคุมการเปิดปิดประตู แต่ในที่นี้จะแสดงผลผ่านหน้าจอ lcd
//ถ้ากดkeypad 'C' จะ เปิดประตูชั่วคราว แต่จะแสดงผลว่า "Temporarily OPENED"

if(customKey1=='C'){
myGLCD.clrScr();
do{ myGLCD.print("Temporarily", CENTER, 16);
myGLCD.print("OPENED!", CENTER, 24);
delay(150);
customKey1 = kpd.getKey();
}while(customKey1=='1'||customKey1=='2'||customKey1=='3'||customKey1==!'||customKey1=='6
'||customKey1==!'B'||customKey1=='7'||customKey1=='8'||customKey1==
||customKey1=='0'||customKey1=='#'||customKey1==!'D');
```

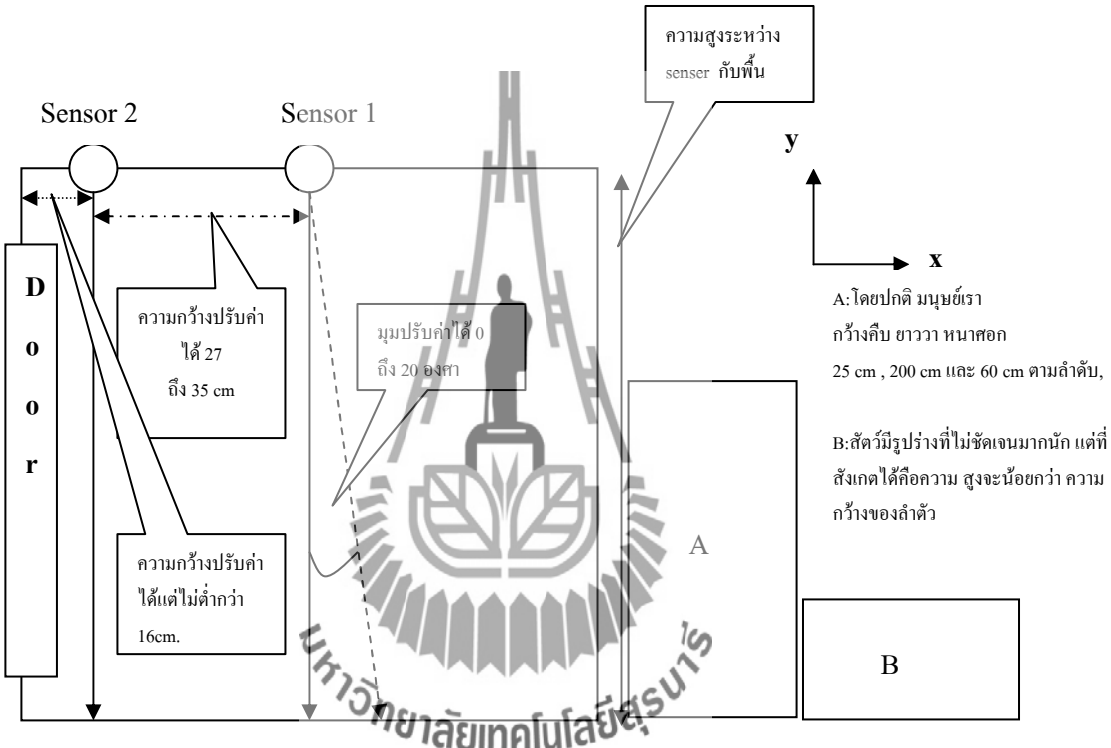
```
}
/*}*/
/*Serial.print("in, ");
Serial.print(cm);
Serial.print("cm");
Serial.print(inches2);
Serial.print("in, ");
Serial.print(cm2);
Serial.print("cm");
Serial.println();*/
}
//codenew2-------ultrasonic
//เป็นฟังก์ชันการคำนวณ จากเวลาให้เป็น ระยะทางความห่างตัว sensor กับวัตถุ เก็บค่าเป็น float ตัว
//แปรที่มีจุดทศนิยม
float microsecondsToInches(float microseconds)
{
// *** THIS NEEDS TO BE CHECKED FOR THE HC-SR04 ***
return microseconds / 74 / 2;
}
float microsecondsToCentimeters(float microseconds)
{
// *** THIS NEEDS TO BE CHECKED FOR THE HC-SR04 ***
return microseconds / 29 / 2;
}
//codenew2-------ultrasonic
```

ส่วนของ code เดิมของผู้ร่วมพัฒนา
คำสั่งเชื่อมกับ Ultrasonic sensor
ซึ่งเราได้พัฒนาโปรแกรมขึ้นมาใหม่
สำหรับโครงงานนี้

บทที่ 3
ผลการทดสอบโครงงาน

3.1 การติดตั้งSensor อุปกรณ์ต่างๆ เพื่อทำการวัดค่า พารามิเตอร์ต่างๆ

การติดตั้งอุปกรณ์บางครั้งขึ้นที่สูงต้องใช้ความระมัดระวังมากขึ้น ทำการติดตั้งดังรูปอ้างอิงจากรูปที่ 19

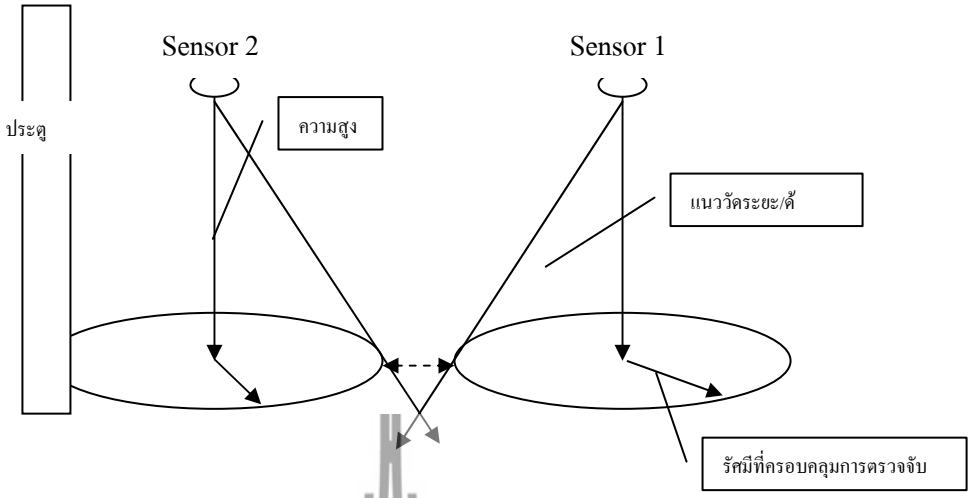


รูปที่ 25 อ้างอิงรูปที่ 19 กำหนดค่าต่างๆ ในแนวแกน x , y

กำหนดพารามิเตอร์ในการวัด

ค่าตัวแปรปรับค่าได้	:	ความสูง (h) ,ระยะหาของตัว sesor 2 ตัว(ตัวแปร l) และมุมองศาที่ปรับได้ของsensor ตัวที่ 1
ค่าคงที่	:	ความห่างของประตูกับตัวsensor ตัวที่ 2 (16 cm.)

เพื่อหาว่า ความกว้างของมนุษย์ที่ผ่านเข้าไปที่ตัวเซนเซอร์ ไม่ถูกระบุว่าเป็นสัตว์ คือหาช่องว่างระหว่าง เซนเซอร์2 ตัว ที่จับมนุษย์ไม่ได้ ในความสูงที่ต่างกัน เพื่อกำหนดค่า l และมุมองศาที่ดีที่สุด ซึ่งตามทฤษฎีแล้ว เซนเซอร์ มีรัศมีจับวัตถุที่ 20 องศา แต่วัตถุต้องเป็นผิวราบเรียบตั้งฉากกับแกนไม่เกิน 3 องศา ที่จะสามารถสะท้อนกับมาที่เดิมได้ ข้อจำกัดนี้จึงอยากหาผลสรุป



รูปที่ 26 รูปร่างของรัศมีการตรวจจับ ในแนวแกน x , y

จะเห็นว่า รัศมีมีความมากขึ้นถ้าความสูงจากพื้นถึงsenesr มีความสูงขึ้น จึงไม่สามารถระบุประเภทของ มนุษย์และสัตว์ได้ชัดเจน แต่ถ้าวางห่างกันเกินไปก็ไม่สามารถระบุว่าเป็นสัตว์ได้เช่นกัน ดังนั้นจึงหาค่าที่ดีพอ ที่สามารถวัดแยกประเภทได้ชัดเจนมาที่สุด

3.2 คำนวณและวิเคราะห์ผลจากตาราง

3.2.1 ปรับเปลี่ยนแก้ไข

ให้ Sensorตัวที่ 1 มุมที่คงค่าไว้ เปลี่ยนค่าให้ทำมุมกับ แนวตั้งฉากเป็น 15 องศา

ตารางที่ 7.1.1 ที่ความสูงของเซนเซอร์ถึงพื้น 250 cm

ความสูงของวัตถุที่ผ่านเข้ามาของมนุษย์(cm)	การตั้งค่าsensor ที่ทำมุมตั้งฉากกับพื้น ให้เป็น 15 องศา ถ้ามากว่าจะไม่สามารถระบุมนุษย์หรือสัตว์ได้
40 – 80	ระบุไม่ได้ที่ 28 cm
81-120	ระบุไม่ได้ที่ 28 cm

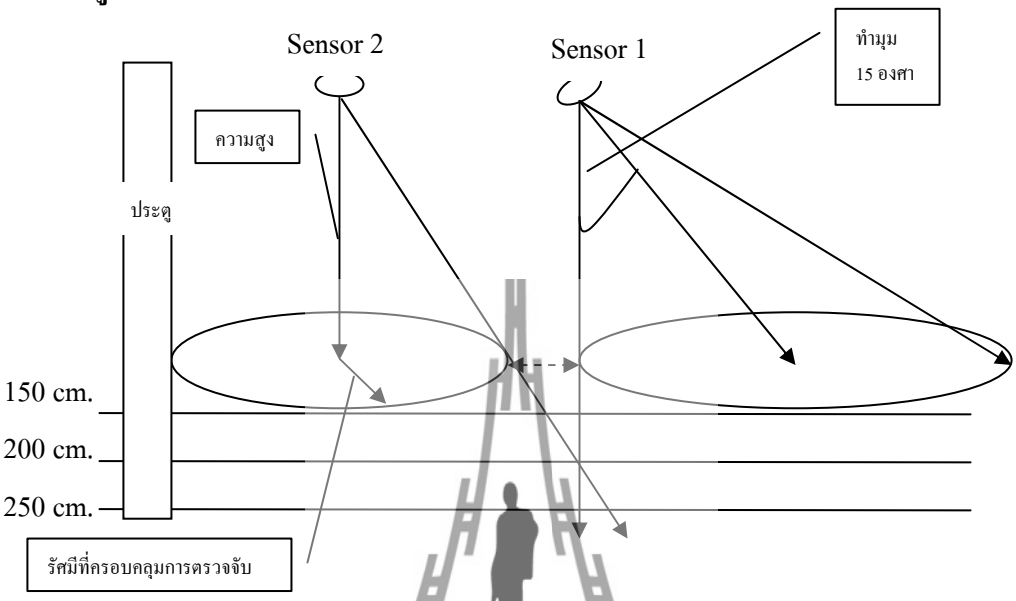
ตารางที่ 7.1.2 ที่ความสูงของเซนเซอร์ถึงพื้น 200 cm

ความสูงของวัตถุที่ผ่านเข้ามาของมนุษย์(cm)	การตั้งค่าsensor ที่ทำมุมตั้งฉากกับพื้น ให้เป็น 15 องศา ถ้ามากว่าจะไม่สามารถระบุมนุษย์หรือสัตว์ได้
40 – 80	ระบุไม่ได้ที่ 15 cm
81-120	ระบุไม่ได้ที่ 15 cm

ตารางที่ 7.1.3 ที่ความสูงของเซนเซอร์ถึงพื้น 150 cm

ความสูงของวัตถุที่ผ่านเข้ามาของมนุษย์(cm)	การตั้งค่าsensor ที่ทำมุมตั้งฉากกับพื้น ให้เป็น 15 องศา ถ้ามากว่าจะไม่สามารถระบุมนุษย์หรือสัตว์ได้
40 – 80	ระบุได้ ที่ 15 cm และ 28 cm เป็นมนุษย์
81-120	ระบุได้ ที่ 15 cm และ 28 cm เป็นมนุษย์ แต่ในความเป็นจริง ความต่างน้อยมาก จึงแยกได้บางคนที่ ความกว้างด้านข้างลำตัว น้อยกว่า 13 cm ซึ่งเป็นไปไม่ได้

ทำให้ได้รูปดังกล่าว



รูปที่ 27 รูปร่างของรัศมีการตรวจจับ ในแนวแกน x , y ที่ความสูงต่างกัน

3.2.2 ปรับเปลี่ยนแก้ไขให้

Sensorตัวที่ 2 มุมที่คงค่าไว้ เปลี่ยนค่าให้ทำมุมกับ แนวตั้งฉากเป็น 10 องศา

ตารางที่ 7.2.1ที่ความสูงของเซนเซอร์ถึงพื้น 250 cm

ความสูงของวัตถุที่ผ่านเข้ามาของมนุษย์(cm)	ปรับตั้งค่าsensor ที่ทำมุมตั้งฉากกับพื้น ให้เป็น 15 องศา ถ้ามากว่าจะไม่สามารถระบุมนุษย์หรือสัตว์ได้
40 – 80 81-120	ระบุได้ที่ 8 cm และ 28 cm เป็นมนุษย์ ระบุได้ที่ 8 cm และ 28 cm เป็นมนุษย์ ช่วงความกว้างด้านลึกของมนุษย์ ต้องมีไม่เกิน 20 cm.

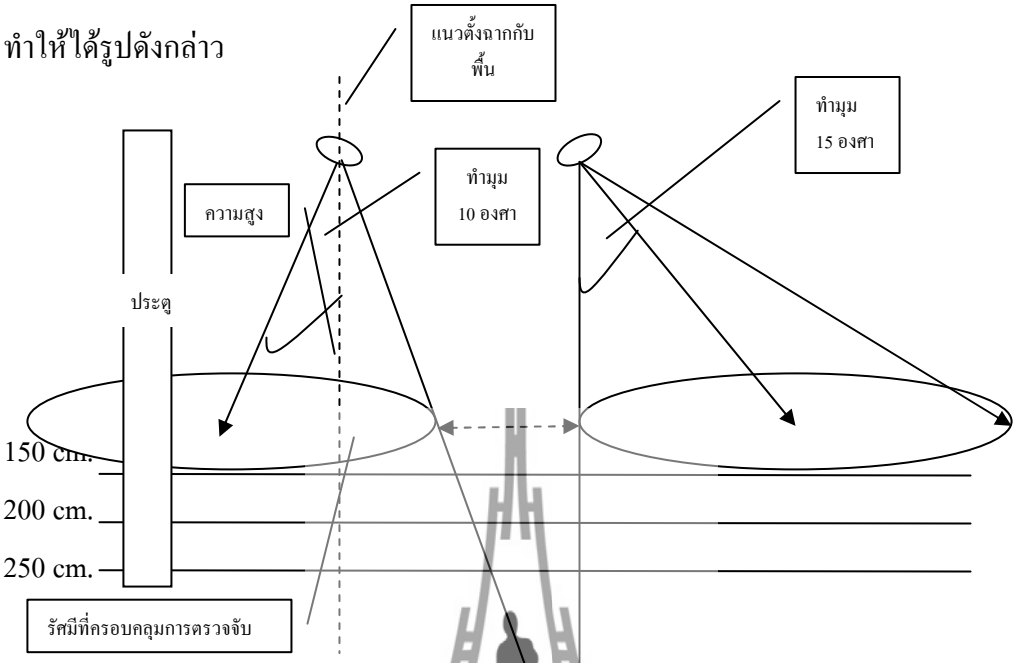
ตารางที่ 7.2.2ที่ความสูงของเซนเซอร์ถึงพื้น 200 cm

ความสูงของวัตถุที่ผ่านเข้ามาของมนุษย์(cm)	การตั้งค่าsensor ที่ทำมุมตั้งฉากกับพื้น ให้เป็น 15 องศา ถ้ามากว่าจะไม่สามารถระบุมนุษย์หรือสัตว์ได้
40 – 80	ระบได้ที่ 5 cm. และ 28 cm.
81-120	ระบุได้ที่ 5 cm. และ 28 cm. ช่วงความกว้างด้านลึกของมนุษย์ ต้องมีไม่เกิน 23 cm.

ตารางที่ 7.2.3ที่ความสูงของเซนเซอร์ถึงพื้น 150 cm

ความสูงของวัตถุที่ผ่านเข้ามาของมนุษย์(cm)	การตั้งค่าsensor ที่ทำมุมตั้งฉากกับพื้น ให้เป็น 15 องศา ถ้ามากว่าจะไม่สามารถระบุมนุษย์หรือสัตว์ได้
40 – 80	ระบุได้ ที่ 0 cm เป็นมนุษย์ และ 28 cmเป็นสัตว์
81-120	ระบุได้ ที่ 0 cm เป็นมนุษย์ และ 28 cmเป็นสัตว์ ช่วงความกว้างด้านลึกของมนุษย์ ต้องมีไม่เกิน 28 cm ซึ่งถือว่าใช้ได้ ถ้าเกินกว่านี้ sensor อาจทำงานผิดพลาดได้

ทำให้ได้รูปดังกล่าว



รูปที่ 28 รูปร่างของรัศมีการตรวจจับ ในแนวแกน x , y ที่ความสูงต่างกัน

3.3 ผลการทดสอบการติดตั้งในแบบต่างๆ

เมื่อเราได้พัฒนาโครงงานให้สามารถแยกมนุษย์และสัตว์ได้ จึงนำมาวัดค่าความเหมาะสมและวัดประสิทธิภาพ ของระบบ และเพื่อนำไปปรับปรุงแก้ไขข้อบกพร่องของระบบ โดยการทดลอง ความกว้างของมนุษย์ที่ผ่านเข้าไปที่ตัวเซนเซอร์ ไม่ถูกระบุว่าเป็นสัตว์ คือหาช่องว่างระหว่าง เซนเซอร์2 ตัว ที่จับมนุษย์ไม่ได้ ในความสูงที่ต่างกัน เพื่อกำหนดค่า l และมุมองศาที่ดีที่สุด ซึ่งตามทฤษฎีแล้ว เซนเซอร์ มีรัศมีจับวัตถุที่ 20 องศา แต่วัตถุต้องเป็นผิวราบเรียบตั้งฉากกับแกนไม่เกิน 3 องศา ที่จะสามารถสะท้อนกับมาที่เดิมได้

กำหนดให้ดังนี้

H= 200 cm l=27 cm α= 0 องศา

H= 220 cm l=30 cm α= 15 องศา

H= 240 cm l=35 cm

H= 260 cm

ตาราง ก หาความห่างที่ sensor ตัวที่ 2 จับค่าไม่ได้ α= 0 องศา ที่ความสูง 40 cm. จากพื้น

ความห่าง l(cm.) / ความสูง H(cm.)	27	30	35
200	2	3	7
220	1	3	4
240	1	2	3
260	0	0	1

ตาราง ข หาความห่างที่ sensor ตัวที่ 2 จับค่าไม่ได้ α= 0 องศา ที่ความสูง 120 cm. จากพื้น

ความห่าง l(cm.) / ความสูง H(cm.)	27	30	35
200	1	2	6
220	0	1	3
240	0	1	1
260	0	0	0

ตาราง ค หาความห่างที่ sensor ตัวที่ 1 จับค่าไม่ได้ α=15 องศา ที่ความสูง 40 cm. จากพื้น

ความห่าง l(cm.) / ความสูง H(cm.)	27	30	35
200	6	10	26
220	4	3	22
240	0	2	18
260	0	0	10

ตาราง ง หาความห่างที่ sensor ตัวที่ 2 จับค่าไม่ได้ α= 15 องศา ที่ความสูง 120 cm. จากพื้น

ความห่าง l(cm.) / ความสูง H(cm.)	27	30	35
200	3	2	10
220	2	1	8
240	0	0	4
260	0	0	2

บทที่ 4
อภิปรายผลการทดสอบโครงงาน

4.1 อภิปราย

จากตารางจะเห็นว่า

1. ความห่าง แปรผันตรงกับ ระยะทางจาก sensor ตัวที่สองถึงจุดไม่สามารถระบุได้ว่าเป็นมนุษย์หรือสัตว์

2. ความสูง แปรผกผันกับ ระยะทางจาก sensor ตัวที่สองถึงจุดไม่สามารถระบุได้ว่าเป็นมนุษย์หรือสัตว์

ได้สมการออกมาดังนี้

P= lxK1 สมการที่ 1

P=K2x(1/H) สมการที่ 2

P = ระยะทางจาก sensor ตัวที่สอง ถึงจุดไม่สามารถระบุได้ว่าเป็นมนุษย์หรือสัตว์

K1= เป็นค่าคงที่ตัวที่หนึ่ง ที่เกี่ยวข้องกับ l

K2= เป็นค่าคงที่ตัวที่หนึ่ง ที่เกี่ยวข้องกับ H

จากสมาการที่ (1)=(2)

จะได้ว่า lxK1=K2x(1/H)

lxH=K2/K1 สมาการที่ 3

ซึ่งจะเห็นว่าความสูงของวัตถุที่ผ่านเข้ามาในตัว sensor มีความสัมพันธ์กับ ตำแหน่งที่แยกไม่ได้ว่าเป็นมนุษย์หรือสัตว์ ซึ่งมีความสอดคล้องกับตารางที่ 7.1และ7.2 ดังนั้น มุมที่ที่ดีที่สุดจากการทดลองของ sensor ตัวที่ หนึ่งคือ 15 องศา sensor ตัวที่ สองคือ 10 องศา

4.2 ปัญหาและอุปสรรค

Software และ Hardware โปรแกรมจำเป็นต้องหา Arduino Libraries มาเพิ่มเติมซึ่งกินเวลาการทำงานนานอาจจะทำการทดลองหรืออะไรต่างๆ ล่าช้าลงไปได้ มีจำนวน pin ของบอร์ด ที่ต่อเข้ากับอุปกร์น้อย ทำให้ไม่สามารถแต่งเติมให้มาขึ้นได้จึงจำเป็นต้องมีตัว module เพิ่มเติมเข้าไปแทนที่จุดนั้น ความไม่แน่นอนของตัว module เองที่อาจเกิดความเสียหายได้ง่าย จำเป็นต้องสั่งซื้อใหม่ทำให้การทดลองทำงานโครงงานล่าช้าอีกด้วย

4.3 แนวทางการพัฒนาและการประยุกต์ร่วมกับงานอื่น

การนำแนวคิดที่ได้จากการทำโครงงานไปต่อยอดอุปกรณ์เซนเซอร์แบบอื่น เช่น ตัวจับความเร็ว แรงสั้นสะเทือน หรือการวัดค่าน้ำในอ่างเก็บน้ำ เป็นต้น หรือพัฒนาฟังก์ชันการทำงานให้ดีขึ้นให้สามารถใช้งานได้สะดวกรวดเร็ว และมีประสิทธิภาพของการระบุตัวมนุษย์และสัตว์ให้ดีขึ้น

4.4 ขอบเขตของโครงงาน และข้อจำกัดด้านอื่นๆ

4.4.1 ขอบเขตของโครงงาน

1. ศึกษาลักษณะการสะท้อนกลับของคลื่น และเวลาที่ใช้ในการสะท้อนกลับ
2. ออกแบบกลไกในการประมูลผลในการวัดส่วนสูงและความยาวฐาน ในแกน y และ x
3. ออกแบบวงจรเชื่อมต่อเพื่อใช้งานระหว่าง Controller Board และSensor แกน x และ y
4. เขียนโปรแกรมควบคุม ลง Controller Board และให้Sensor วัดค่าได้
5. เขียนโปรแกรมประมวลผลค่าการวัดแกน x และ y และแยกประเภทมนุษย์และสัตว์ได้ โดยสัมพันธ์กับการเปิดปิดประตูอัตโนมัติได้
6. สร้างอุปกรณ์ต้นแบบทั้งหมดและทดสอบเพื่อให้ได้ตามวัตถุประสงค์

4.4.2 ข้อจำกัดด้านอื่นๆ

Software : โปรแกรม arduino-1.0.4 (มีลักษณ์คล้ายภาษาC++) และใช้ในการวิเคราะห์ผล

Hardware : Arduino mega 2560 r3 Board , module ultrasonic sensor hc-sr04 , ic(integrated circuit) CD4050BE , keypad 4 x 4 , หลอดLED และจอ Graphic LCD 84x48 Nokia 5110

4.5 สรุป เสนอแนะ และกลุ่มผู้ใช้งาน

4.5.1 สรุป

การใช้งานตัวอุปกรณ์ module ultrasonic sensor hc-sr04 , ic(integrated circuit)CD4050BE , keypad 4 x 4 , หลอดLED และจอ Graphic LCD 84x48 Nokia 5110 รวมถึงหัวใจสำคัญคือ arduino board พัฒนาขึ้นให้สามารถทำงานความสะดวกสบายให้กับผู้ใช้ได้ดังต่อไปนี้

- ทำการแยกประเภท คือ มนุษย์และสัตว์ ในการผ่านเข้าประตู
- กำหนดความละเอียดของสัตว์ที่สามารถผ่านเข้าประตูได้

4.5.2 ข้อเสนอแนะ

สำหรับผู้ที่สนใจใช้อุปกรณ์ตรวจจับ แยกประเภท มนุษย์และสัตว์ให้สามารถผ่านเข้าประตูได้ และต้องการนำไปพัฒนาต่อไปสามารถเพิ่มประสิทธิภาพในการแยกประเภท ให้ดีขึ้นโดยเพิ่มตัว sensor ultrasonic ในแกนต่างๆได้ แต่ต้องคิดถึงเรื่องราคาด้วย หรือมีสูตรวิธีหาคำนวณอื่นๆ

4.6 กลุ่มผู้ใช้งาน

ห้างร้านค้าสะดวกซื้อ บริษัทต่างๆ ในครัวเรือนทั่วไปที่ต้องความสะดวกสบาย และกลุ่มอุตสาหกรรมในการแยกย่อยวัตถุสิ่งของต่างๆ ออกจากกัน ซึ่งsoftware และ hardware สามารถนำไปต่อยอดได้

ประวัติผู้เขียน

นายธนทรัพย์ ตนะทิพย์ เกิดเมื่อวันที่ 8 กรกฎาคม พ.ศ. 2530 ภูมิลำเนาอยู่ที่ ตำบลจอมพระ อำเภอท่าวังผา จังหวัดน่าน สำเร็จการศึกษาระดับมัธยมปลายจากโรงเรียนท่าวังผาพิทยาคม อำเภอท่าวังผา จังหวัดน่าน เมื่อปี พ.ศ. 2548 ปัจจุบันเป็นนักศึกษาชั้นปีที่ 6 สาขาวิชาวิศวกรรมโทรคมนาคม สำนักวิชาวิศวกรรมศาสตร์ มหาวิทยาลัยเทคโนโลยีสุรนารี

เอกสารอ้างอิง

1. keypad

http://www.parallax.com/Portals/0/Downloads/docs/prod/hardware/27899-4x4matrixmembranekeypad-v1.2.pdf

2. lcdNokia5110

https://www.sparkfun.com/datasheets/LCD/Monochrome/Nokia5110.pdf

3. IC CD4050BE/BC

http://pdf1.alldatasheet.com/datasheet-pdf/view/50859/FAIRCHILD/CD4050/+_0J5-UwYhRDpKxHepKwY.z+/datasheet.pdf

4. 3 mm Round LED (T-1)

http://www.promelec.ru/pdf/204-15UTC-S400-X9.pdf

4. หลักการทำงาน board เพิ่มเติมที่

http://arduino.cc/en/Main/arduinoBoardMega2560

5. datasheet board

http://www.atmel.com/Images/doc2549.pdf

6. หลักการทำงาน board

http://arduino.cc/en/uploads/Main/arduino-mega2560_R3-schematic.pdf

7. Ultrasonic ranging module : HC-SR04

http://jaktek.com/wp-content/uploads/2011/12/HC-SR04.pdf

8. คู่มือใช้ dia

http://202.28.72.148/~picnic/dia/Dia_Manual.pdf

9. การเขียน flowchart

http://meewebfree.com/site/start-website-builder/25-flowchart-for-programming

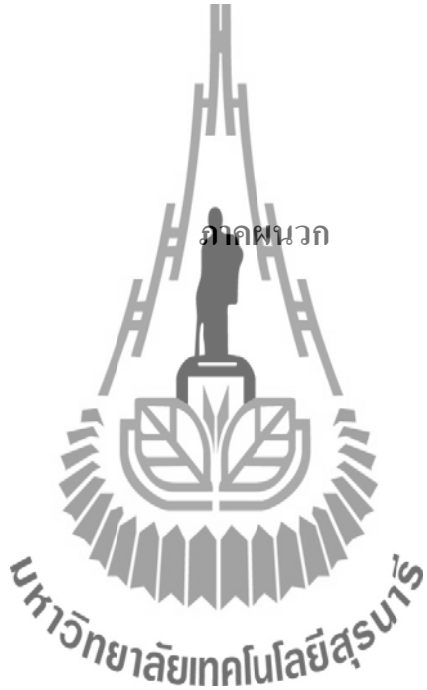
10. คำสั่ง ตัวแปร ที่สำคัญโปรแกรม arduino

http://www.logicthai.net/node/10

11. ทฤษฎีเสียงและการได้ยิน

http://www.elec-cm.com/srt/attachments/article/55/ ใบความรู้ที่%201เสียงและการได้ยิน.pdf

ภาคผนวก



ภาคผนวก ก

Datasheet อุปกรณ์ IC module ต่างๆ และอุปกรณ์ที่ใช้งานสำหรับทำโครงงาน

CD4050BC หรือ CD4050BE Datasheet

FAIRCHILD SEMICONDUCTOR TM

October 1987
Revised January 1999

CD4049UBC • CD4050BC
Hex Inverting Buffer • Hex Non-Inverting Buffer

General Description

The CD4049UBC and CD4050BC hex buffers are monolithic complementary MOS (CMOS) integrated circuits constructed with N- and P-channel enhancement mode transistors. These devices feature logic level conversion using only one supply voltage (V_{DD}). The input signal high level (V_{IH}) can exceed the V_{DD} supply voltage when these devices are used for logic level conversions. These devices are intended for use as hex buffers, CMOS to DTL/TTL converters, or as CMOS current drivers, and at V_{DD} = 5.0V, they can drive directly two DTL/TTL loads over the full operating temperature range.

Features

- Wide supply voltage range: 3.0V to 15V
- Direct drive to 2 TTL loads at 5.0V over full temperature range
- High source and sink current capability
- Special input protection permits input voltages greater than V_{DD}

Applications

- CMOS hex inverter/buffer
- CMOS to DTL/TTL hex converter
- CMOS current "sink" or "source" driver
- CMOS HIGH-to-LOW logic level converter

Ordering Code:

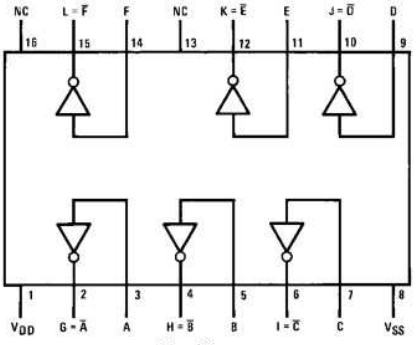
Order Number	Package Number	Package Description
CD4049UBCM	M16A	16-Lead Small Outline Integrated Circuit (SOIC), JEDEC MS-012, 0.150" Narrow
CD4049UBCN	N16E	16-Lead Plastic Dual-In-Line Package (PDIP), JEDEC MS-001, 0.300" Wide
CD4050BCM	M16A	16-Lead Small Outline Integrated Circuit (SOIC), JEDEC MS-012, 0.150" Narrow
CD4050BCN	N16E	16-Lead Plastic Dual-In-Line Package (PDIP), JEDEC MS-001, 0.300" Wide

Devices also available in Tape and Reel. Specify by appending the suffix letter "X" to the ordering code.

Connection Diagrams

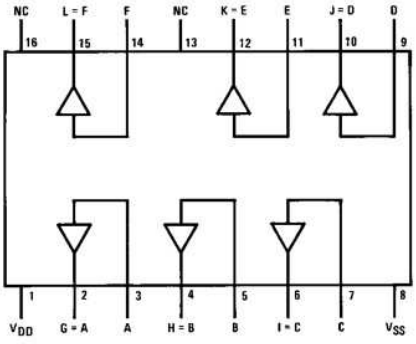
Pin Assignments for DIP

CD4049UBC



Top View

CD4050BC



Top View

Nokia 5110

PCD854448 ´ 84 pixels matrix LCD

48 × 84 pixels matrix LCD controller/driver PCD8544

1 FEATURES

- Single chip LCD controller/driver
- 48 row, 84 column outputs
- Display data RAM 48 × 84 bits
- On-chip:
  - Generation of LCD supply voltage (external supply also possible)
  - Generation of intermediate LCD bias voltages
  - Oscillator requires no external components (external clock also possible).
- External $\overline{RES}$ (reset) input pin
- Serial interface maximum 4.0 Mbits/s
- CMOS compatible inputs
- Mux rate: 48
- Logic supply voltage range V_{DD} to V_{SS}: 2.7 to 3.3 V
- Display supply voltage range V_{LCD} to V_{SS}
  - 6.0 to 8.5 V with LCD voltage internally generated (voltage generator enabled)
  - 6.0 to 9.0 V with LCD voltage externally supplied (voltage generator switched-off).
- Low power consumption, suitable for battery operated systems
- Temperature compensation of V_{LCD}
- Temperature range: −25 to +70 °C.

2 GENERAL DESCRIPTION

The PCD8544 is a low power CMOS LCD controller/driver, designed to drive a graphic display of 48 rows and 84 columns. All necessary functions for the display are provided in a single chip, including on-chip generation of LCD supply and bias voltages, resulting in a minimum of external components and low power consumption.

The PCD8544 interfaces to microcontrollers through a serial bus interface.

The PCD8544 is manufactured in n-well CMOS technology.

3 APPLICATIONS

- Telecommunications equipment.

4 ORDERING INFORMATION

TYPE NUMBER	PACKAGE		
	NAME	DESCRIPTION	VERSION
PCD8544U	–	chip with bumps in tray; 168 bonding pads + 4 dummy pads	–

5 BLOCK DIAGRAM

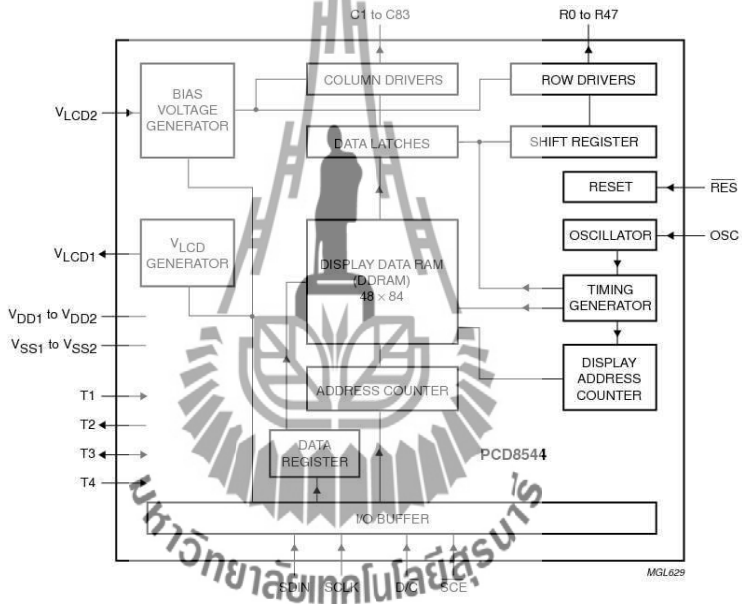


Fig.1 Block diagram.

6 PINNING

SYMBOL	DESCRIPTION
R0 to R47	LCD row driver outputs
C0 to C83	LCD column driver outputs
V_{SS1}, V_{SS2}	ground
V_{DD1}, V_{DD2}	supply voltage
V_{LCD1}, V_{LCD2}	LCD supply voltage
T1	test 1 input
T2	test 2 output
T3	test 3 input/output
T4	test 4 input
SDIN	serial data input
SCLK	serial clock input
D/$\overline{C}$	data/command
$\overline{SCE}$	chip enable
OSC	oscillator
$\overline{RES}$	external reset input
dummy1, 2, 3, 4	not connected

Note

1. For further details, see Fig.18 and Table 7.

6.1 Pin functions

6.1.1 R0 TO R47 ROW DRIVER OUTPUTS

These pads output the row signals.

6.1.2 C0 TO C83 COLUMN DRIVER OUTPUTS

These pads output the column signals.

6.1.3 V_{SS1}, V_{SS2}: NEGATIVE POWER SUPPLY RAILS

Supply rails V_{SS1} and V_{SS2} must be connected together.

6.1.4 V_{DD1}, V_{DD2}: POSITIVE POWER SUPPLY RAILS

Supply rails V_{DD1} and V_{DD2} must be connected together.

6.1.5 V_{LCD1}, V_{LCD2}: LCD POWER SUPPLY

Positive power supply for the liquid crystal display. Supply rails V_{LCD1} and V_{LCD2} must be connected together.

6.1.6 T1, T2, T3 AND T4: TEST PADS

T1, T3 and T4 must be connected to V_{SS}, T2 is to be left open. Not accessible to user.

6.1.7 SDIN: SERIAL DATA LINE

Input for the data line.

6.1.8 SCLK: SERIAL CLOCK LINE

Input for the clock signal: 0.0 to 4.0 Mbits/s.

6.1.9 D/$\overline{C}$: MODE SELECT

Input to select either command/address or data input.

6.1.10 $\overline{SCE}$: CHIP ENABLE

The enable pin allows data to be clocked in. The signal is active LOW.

6.1.11 OSC: OSCILLATOR

When the on-chip oscillator is used, this input must be connected to V_{DD}. An external clock signal, if used, is connected to this input. If the oscillator and external clock are both inhibited by connecting the OSC pin to V_{SS}, the display is not clocked and may be left in a DC state. To avoid this, the chip should always be put into Power-down mode before stopping the clock.

6.1.12 $\overline{RES}$: RESET

This signal will reset the device and must be applied to properly initialize the chip. The signal is active LOW.

4x4matrixmembranekeypad-v1.2

PARALLAX

Web Site: www.parallax.com
Forums: forums.parallax.com
Sales: sales@parallax.com
Technical: support@parallax.com

Office: (916) 624-8333
Fax: (916) 624-8003
Sales: (888) 512-1024
Tech Support: (888) 997-8267

4x4 Matrix Membrane Keypad (#27899)

This 16-button keypad provides a useful human interface component for microcontroller projects. Convenient adhesive backing provides a simple way to mount the keypad in a variety of applications.

Features

- Ultra-thin design
- Adhesive backing
- Excellent price/performance ratio
- Easy interface to any microcontroller
- Example programs provided for the BASIC Stamp 2 and Propeller P8X32A microcontrollers

Key Specifications

- Maximum Rating: 24 VDC, 30 mA
- Interface: 8-pin access to 4x4 matrix
- Operating temperature: 32 to 122 °F (0 to 50°C)
- Dimensions:
  Keypad, 2.7 x 3.0 in (6.9 x 7.6 cm)
  Cable: 0.78 x 3.5 in (2.0 x 8.8 cm)

Application Ideas

- Security systems
- Menu selection
- Data entry for embedded systems

How it Works

Matrix keypads use a combination of four rows and four columns to provide button states to the host device, typically a microcontroller. Underneath each key is a pushbutton, with one end connected to one row, and the other end connected to one column. These connections are shown in Figure 1.

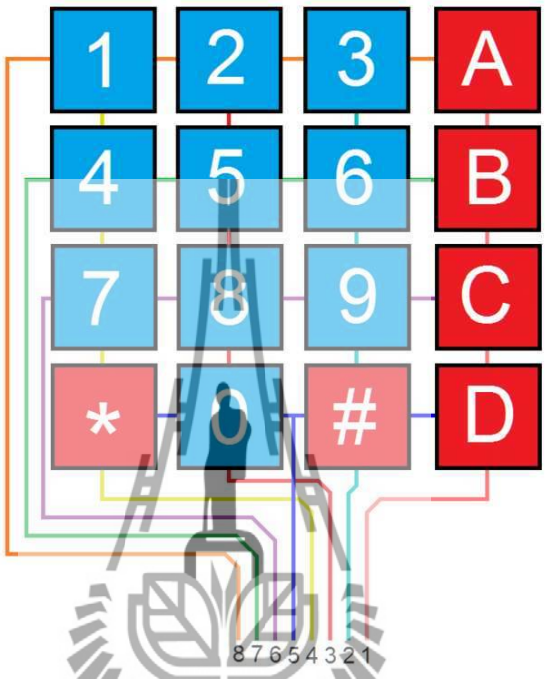


Figure 1: Matrix Keypad Connections

In order for the microcontroller to determine which button is pressed, it first needs to pull each of the four columns (pins 1-4) either low or high one at a time, and then poll the states of the four rows (pins 5-8). Depending on the states of the columns, the microcontroller can tell which button is pressed.

For example, say your program pulls all four columns low and then pulls the first row high. It then reads the input states of each column, and reads pin 4 high. This means that a contact has been made between column 4 and row 1, so button 'A' has been pressed.

3 mm Round LED (T-1)

EVERLIGHT

Technical Data Sheet
3 mm Round LED (T-1)

204-15UTC/S400-X9

Features

- Popular T-1 colorless 3mm package.
- High luminous power.
- Typical chromaticity coordinates x=0.29, y=0.28 according to CIE1931.
- Bulk, available taped on reel.
- Pb free.
- ESD-withstand voltage: up to 4KV
- The product itself will remain within RoHS compliant version.

Descriptions

- The series is designed for application required high luminous intensity.
- The phosphor filled in the reflector converts the blue emission of InGaN chip to ideal white.

Applications

- Outdoor Displays
- Optical Indicators
- Backlighting
- Marker Lights

Device Selection Guide

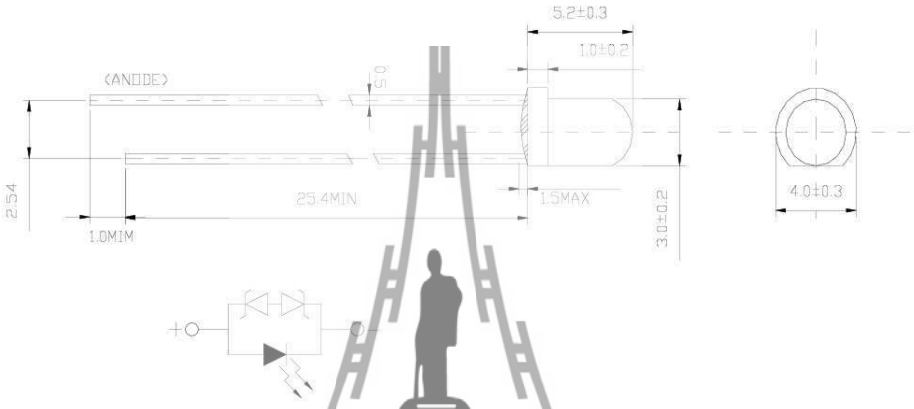
PART NO.	Chip		Lens Color
	Material	Emitted Color	
204-15UTC/S400-X9	InGaN	White	Water Clear

EVERLIGHT

Technical Data Sheet
3 mm Round LED (T-1)

204-15UTC/S400-X9

Package Dimensions



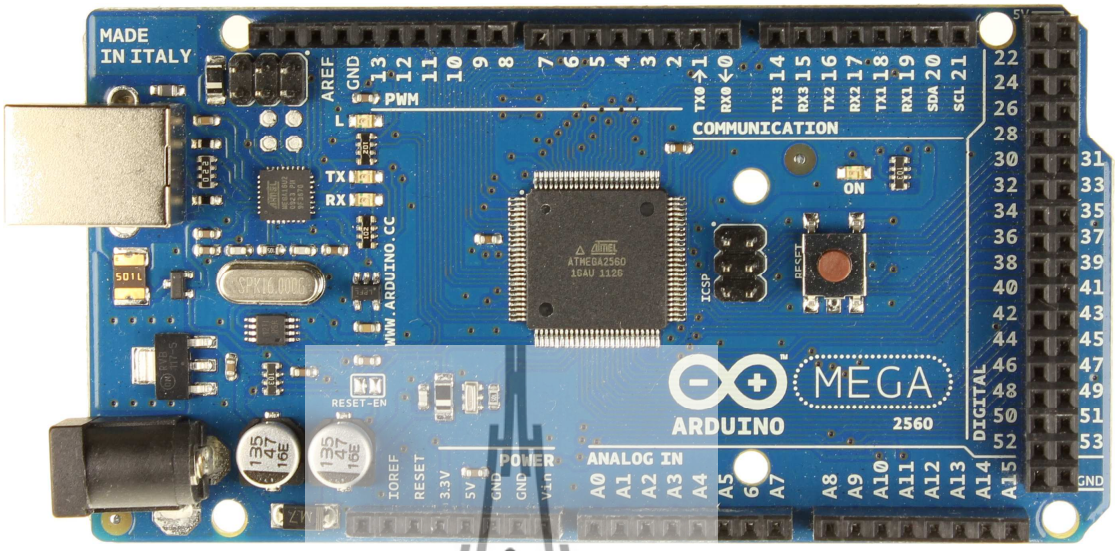
Notes:

1.All dimensions are in millimeters, and tolerance is 0.25mm except being specified.
2.Lead spacing is measured where the lead emerges from the package.
3.Protruded resin under flange is 1.5mm Max. LED.

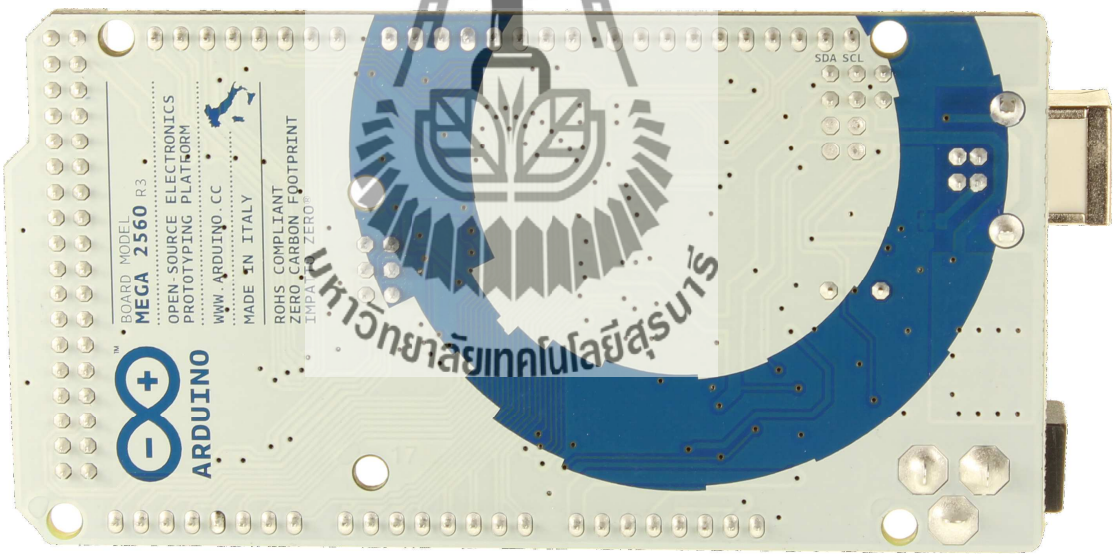
Absolute Maximum Ratings (Ta=25℃)

Parameter	Symbol	Rating	Unit
Continuous Forward Current	I_F	25	mA
Peak Forward Current(Duty 1/10 @ 1KHZ)	I_{FM}	100	mA
Reverse Voltage	V_R	5	V
Operating Temperature	T_{opr}	-40 ~ +85	℃
Storage Temperature	T_{stg}	-40 ~ +100	℃
Soldering Temperature (T=5 sec)	T_{sol}	260 ± 5	℃
Power Dissipation	P_d	100	mW
Zener Reverse Current	Iz	100	mA
Electrostatic Discharge	ESD	4K	V

arduino-mega2560_R3

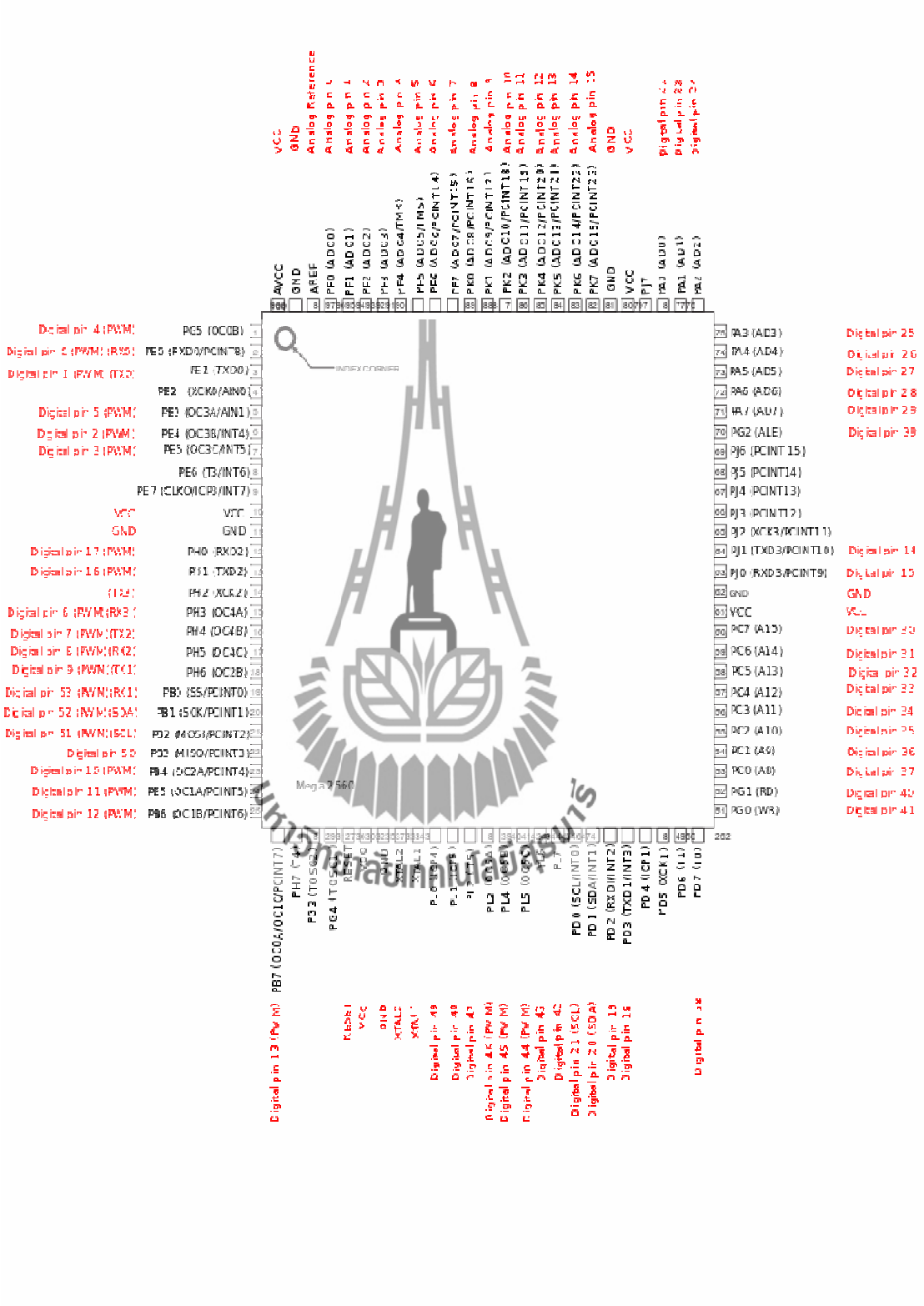


ArduinoMega2560_R3_Front



ArduinoMega2560_R3_Back

Pin Mapping arduino-mega2560_R3



Summary arduino-mega2560_R3

Microcontroller	ATmega2560
Operating Voltage	5V
Input Voltage (recommended)	7-12V
Input Voltage (limits)	6-20V
Digital I/O Pins	54 (of which 15 provide PWM output)
Analog Input Pins	16
DC Current per I/O Pin	40 mA
DC Current for 3.3V Pin	50 mA
Flash Memory	256 KB of which 8 KB used by bootloader
SRAM	8 KB
EEPROM	4 KB
Clock Speed	16 MHz

Ultrasonic ranging module : HC-SR04

Ultrasonic ranging module : HC-SR04

Specifications:

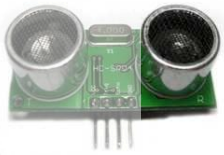
power supply :5V DC

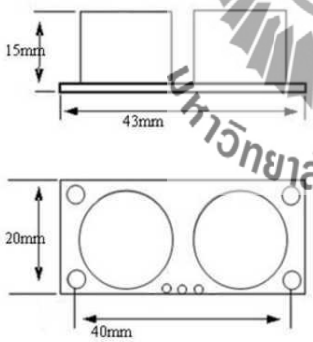
quiescent current : <2mA

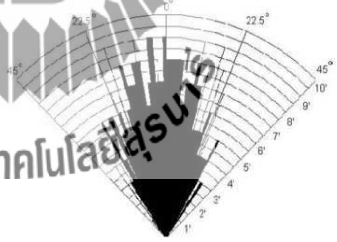
effectual angle: <15°

ranging distance : 2cm – 500 cm

resolution : 0.3 cm

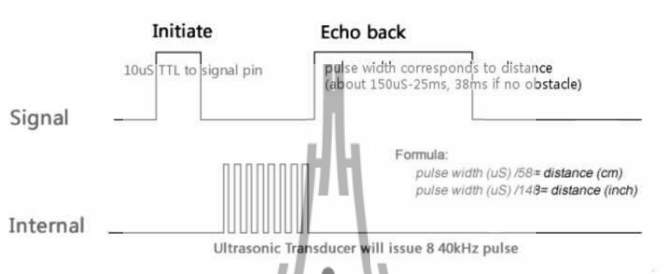






Practical test of performance,
Best in 30 degree angle

Sequence chart



A short ultrasonic pulse is transmitted at the time 0, reflected by an object. The senor receives this signal and converts it to an electric signal. The next pulse can be transmitted when the echo is faded away. This time period is called cycle period. The recommend cycle period should be no less than 50ms. If a 10μs width trigger pulse is sent to the signal pin, the Ultrasonic module will output eight 40kHz ultrasonic signal and detect the echo back. The measured distance is proportional to the echo pulse width and can be calculated by the formula above. If no obstacle is detected, the output pin will give a 38ms high level signal.

Library:
http://iteadstudio.com/store/images/produce/Robot/HCSR04/Ultrasonic.rar

อุปกรณ์อื่นๆที่เกี่ยวข้องกับโครงงาน

- Jump Wire (Male to Male) ยาว 20cm. จำนวน 40 เส้น
- Jump Wire (Male to Female) ยาว 20cm. จำนวน 40 เส้น
- Jump Wire (Female to Female) ยาว 20cm. จำนวน 40 เส้น

รูปที่ Jump Wire

Module อื่นๆ ที่อยากแนะนำมาใช้ board arduino-mega2560_R3

Servo specfication:

- Model:DS04-NFC
- Weight:38g
- Size:40.8 x 20 x 39.5 mm
- Torque:5.5kg/cm(at 4.8V)
- Speed:0.22sec/60°(at 4.8V)
- Voltage:4.8v-6v
- Temperature:0℃-60℃
- Current: < 1000mA

Buzzer Module

Description

Introduction: Buzzer Module use DC power supply, as voice device widely used in electronic products like computer, printer, duplicator, alarm, electronic toy, auto electronic equipment, telephone, timer .Active buzzer directly connected to 5V rated supply, this can be continuous sounding .The module combining with Arduino sensor shield can do simple circuit design, "plug and play."

ภาคผนวก ข

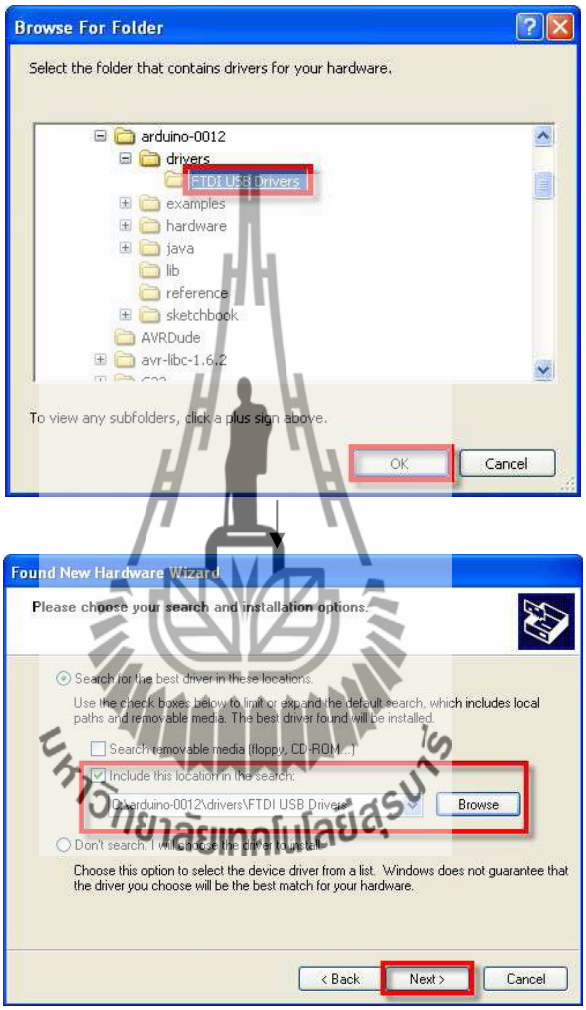
คู่มือการใช้งานโปรแกรม Arduino -1.0.4 (มีลักษณ์คล้ายภาษาC++)

การติดตั้ง Driver ของ USB Bridge ของบอร์ด ET-MEGA2560-ADKบอร์ด ET-MEGA2560-ADK จะใช้ชิพ USB Bridge ของ FTDI เป็นตัวกลางในการเชื่อมต่อกับคอมพิวเตอร์ PC โดย USB Bridge ของ FTDI จะทำหน้าที่เป็นตัวกลางในการเชื่อมต่อและติดต่อสื่อสารระหว่างเครื่องคอมพิวเตอร์ PC กับ MCU ATmega2560 ของบอร์ด ET-MEGA2560-ADK ในรูปแบบของพอร์ตอนุกรม (Visual Com Port) โดยโปรแกรม Application ต่างๆที่ทำงานอยู่บนคอมพิวเตอร์ PC รวมทั้งโปรแกรม Arduino จะมองเห็น พอร์ต USB ที่เชื่อมต่อกับบอร์ด ET-MEGA2560-ADK เป็นพอร์ตสื่อสารอนุกรม (Com Port) ช่องหนึ่งเท่านั้น ซึ่งถ้าเครื่องคอมพิวเตอร์ของผู้ใช้เคยทำการติดตั้ง Driver สำหรับ USBBridge ของ FTDI ไว้ก่อนแล้ว เมื่อทำการเชื่อมต่อสาย USB ของบอร์ด ET-MEGA2560-ADK เข้ากับ USBHUB ของเครื่องคอมพิวเตอร์ PC แล้ว Windows จะทำการติดตั้ง Driver ให้เองโดยอัตโนมัติ แต่ถ้าเครื่องคอมพิวเตอร์ PC ยังไม่เคยติดตั้ง Driver ของ FTDI ไว้ก่อนก็จะต้องทำการติดตั้ง Driver ให้กับบอร์ดให้เรียบร้อยเสียก่อนซึ่งมีลำดับขั้นตอนดังนี้

1. เตรียมแผ่น CD ROM ที่บรรจุ Driver ของ FTDI ไว้ให้พร้อม หรือ ในกรณีที่ผู้ใช้ได้ทำการติดตั้งโปรแกรมของ Arduino ไว้เรียบร้อยแล้ว ภายในโฟล์เดอร์ของโปรแกรม Arduino ก็จะมี Driver ของFTDI จัดเตรียมไว้ให้เรียบร้อยแล้ว โดยจะอยู่ที่ **“C:\arduino-0012\drivers\FTDI USB Drivers\”**

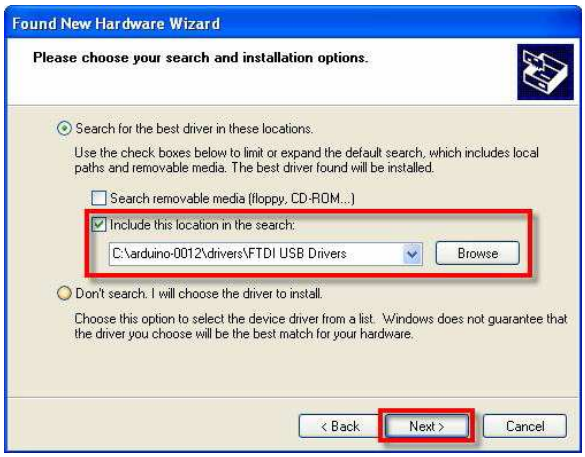
2. ทำการเสียบสาย USB ของบอร์ด ET-MEGA2560-ADK เข้ากับพอร์ต USB HUB ของเครื่องคอมพิวเตอร์ PC ซึ่ง Windows จะตรวจพบอุปกรณ์ใหม่ โดยเป็น **“FT232R USB UART”** และแจ้งให้ผู้ใช้ทำการติดตั้ง Driver ให้กับอุปกรณ์ ดังรูป

3. ให้เลือก Install from list or specific location(Advanced) แล้วเลือก Next ซึ่ง Windows ก็จะแจ้งให้ผู้ใช้ระบุตำแหน่งโฟลเดอร์ที่บรรจุไฟล์ Driver ของ FTDI ไว้ ก็ให้เลือกที่ Browse และเลือกไปยังDrive และ โฟลเดอร์ที่เก็บไฟล์ Driver ไว้ ซึ่งถ้าผู้ใช้ได้ทำการติดตั้งโปรแกรมของ Arduinoไว้แล้ว ก็ให้เลือกไปที่ “C:\arduino-0012\drivers\FTDI USB Drivers” แล้วเลือก Next ดังรูป



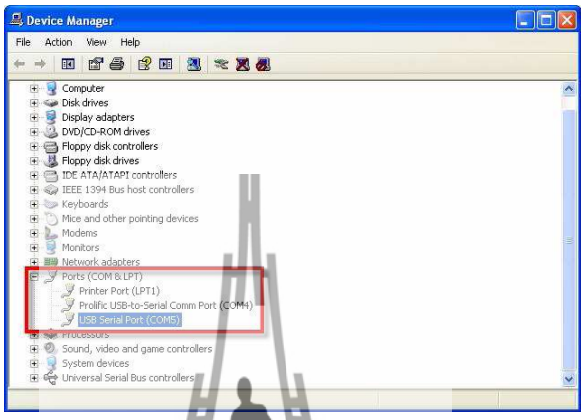
4. ในขั้นตอนนี้โปรแกรม Windows จะทำการค้นหาและติดตั้ง Driver ให้กับอุปกรณ์ ให้รอสักครู่จนการทำงานเสร็จเรียบร้อย แล้วเลือก Finish ดังรูป

5. หลังจากทำการติดตั้ง Driver ของฮาร์ดแวร์เรียบร้อยแล้ว Windows ก็จะตรวจพบว่ามีอุปกรณ์ใหม่ถูกเชื่อมต่ออยู่ โดยเป็นอุปกรณ์ประเภท “USB Serial Port” และแจ้งให้ผู้ใช้ทำการติดตั้ง Driverให้กับอุปกรณ์ใหม่ที่ระบุเป็น “USB Serial Port” อีกครั้งหนึ่ง ซึ่งก็ให้เลือกระบุตำแหน่งโฟลเดอร์ที่เก็บไฟล์ Driver ไว้ ซึ่งให้เลือกเหมือนขั้นตอนในหัวข้อที่ 3 ดังรูป



6. ในขั้นตอนนี้โปรแกรม Windows จะทำการค้นหาและติดตั้ง Driver ให้กับอุปกรณ์ ให้รอสักครู่จนการทำงานเสร็จเรียบร้อย แล้วเลือก Finish ดังรูป

7. หลังจากทำการติดตั้ง Driver เรียบร้อยแล้ว ก็สามารถใช้งานอุปกรณ์ได้แล้ว แต่เพื่อความถูกต้องในครั้งแรกนี้ควรต้องเข้าไปทำการตรวจสอบและปรับแต่งค่าให้กับอุปกรณ์ก่อน โดยในขั้นตอนนี้ให้ไปที่ “My Computer > Control Panel > System > Hardware > Device Manager” แล้วทำการตรวจสอบที่ Ports (COM&LPT) แล้วดูที่ชื่อของ “USB Serial Port” ซึ่งให้ผู้ใช้จดจำหมายเลขของ Com Port ของอุปกรณ์ดังกล่าวไว้ เพื่อใช้อ้างอิงถึงในการเรียกใช้งาน ดังรูป

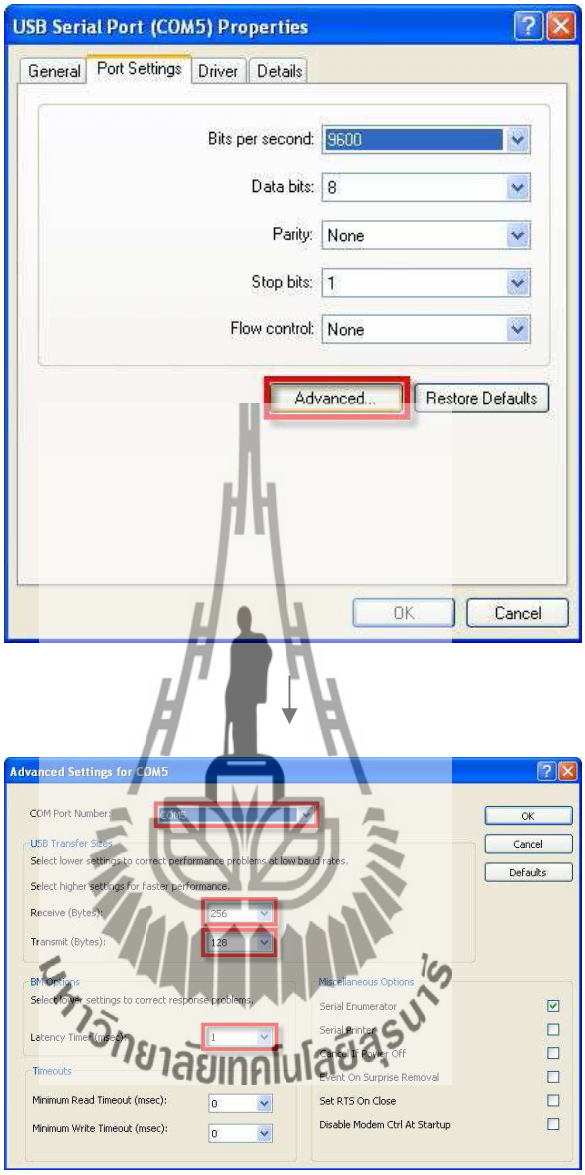


8. ในขั้นตอนนี้ให้คลิกเมาส์ที่เครื่องหมาย (+) หน้าหัวข้อ Ports(COM&LPT) แล้วมองหาอุปกรณ์ที่ชื่อ“USB Serial Port” ตามที่เราได้ทำการติดตั้ง Driver ไว้เรียบร้อยแล้ว หรือ ถ้าไม่แน่ใจว่าอุปกรณ์ดังกล่าวใช่อุปกรณ์ที่เป็นของบอร์ด “ET-MEGA2560-ADK” หรือไม่ ให้ทดสอบด้วยการถอดสายUSB ออก รายชื่ออุปกรณ์ดังกล่าวจะต้องหายไป แต่เมื่อเสียบสาย USB กลับเข้ามาใหม่ รายชื่อของอุปกรณ์ดังกล่าวก็จะต้องปรากฏให้เห็นอีกครั้ง ถ้าทุกอย่างถูกต้อง ก็ให้ทำการคลิกเมาส์ที่ Tabรายชื่อของอุปกรณ์ดังกล่าว เมื่อปรากฏหน้าต่าง USB Serial Port Properties ขึ้นมาแล้วให้เลือกที่ Port Setting แล้วเลือก Advance เพื่อเข้าไปกำหนดค่าให้กับอุปกรณ์ให้เรียบร้อย ดังนี้

a. USB Transfer Size > Receive (Bytes) ให้กำหนดเป็น 256

b. USB Transfer Size > Transmit (Bytes) ให้กำหนดเป็น 128

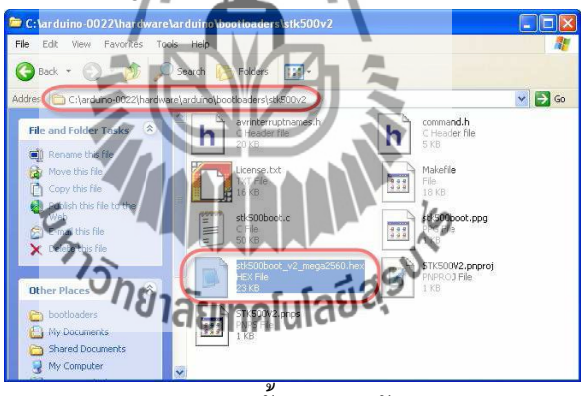
c. BM Option > Latency Timer (mSec) ให้กำหนดเป็น 1


USB Serial Port (COM5) Properties
General
Port Settings
Driver
Details
Bits per second: 9600
Data bits: 8
Parity: None
Stop bits: 1
Flow control: None
Advanced...
Restore Defaults
OK
Cancel
Advanced Settings for COM5
COM Port Number: COM5
OK
Cancel
Defaults
USB Transfer Sizes
Select lower settings to correct performance problems at low baud rates.
Select higher settings for faster performance.
Receive (Bytes): 256
Transmit (Bytes): 128
BM Options
Select lower settings to correct response problems.
Latency Timer (msec): 1
Timeouts
Minimum Read Timeout (msec): 0
Minimum Write Timeout (msec): 0
Miscellaneous Options
Serial Enumerator
Serial Printer
Cancel If Power Off
Event On Surprise Removal
Set RTS On Close
Disable Modem Ctrl At Startup

การพัฒนาโปรแกรมของ Arduino mega 2560 r3 Board ด้วย arduino-1.0.4

ตามปรกติแล้วบอร์ด Arduino mega 2560 r3 จะทำการ ติดตั้งโปรแกรม Bootloader ไว้ให้กับ MCUเป็นที่เรียบร้อยแล้ว โดยใช้ Bootloader ชื่อ “stk500boot_v2_mega2560.hex” ซึ่งเป็น Bootloaderมาตรฐานจาก Arduino โดยโปรแกรม Bootloader นี้จะใช้สำหรับติดต่อสื่อสารเพื่อสั่ง Upload Code จากคอมพิวเตอร์ PC ให้กับ MCU ในบอร์ดทำงาน โดยไม่ต้องใช้เครื่องโปรแกรมภายนอกให้ยุ่งยาก ซึ่งคุณสมบัติของ Bootloader รุ่น Arduino-0022 มีคุณสมบัติการทำงานเป็นดังนี้

- สื่อสารกับโปรแกรมภายนอกด้วย Protocol แบบ stk500v2
- ใช้ความเร็ว Baudrate 115200 โดยใช้ความถี่ XTAL 16 MHz
- โปรแกรม Bootloader มีขนาด 8KByte ทำงานที่ตำแหน่ง 0x3E000-0x3FFFF
- ใช้ LED ที่ต่อกับขา Digital-13 เป็นตัวแสดงสถานะในขณะที่ Bootloader ทำงาน
- โปรแกรมใน Bootloader จะทำงานโดยอัตโนมัติทุกครั้งหลังการรีเซ็ต โดย MCU จะเริ่มต้นทำงานใน Bootloader นี้ก่อนเสมอ เพื่อรอการติดต่อสื่อสารจากโปรแกรมสำหรับสั่งให้ทำการ UploadCode ให้กับ MCU แต่ถ้าไม่มีการติดต่อสื่อสารเข้ามาภายในเวลาที่กำหนดไว้ ก็จะกระโดดไปทำงานตามโปรแกรมที่ผู้ใช้โหลดไว้ให้ทันที

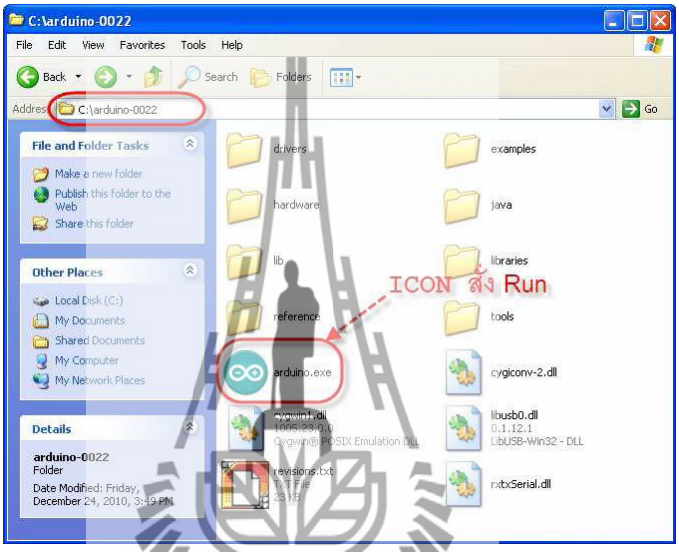


สำหรับบอร์ด ET-MEGA2560-ADK นั้น จะรองรับการ Reset MCU แบบอัตโนมัติจาก USB Bridge(FT232RL) โดยใช้ขา DTR จาก FT232RL เป็นขาควบคุมการรีเซ็ต MCU

การติดตั้งโปรแกรม Arduino

สำหรับโปรแกรม Arduino นั้น ได้รับการพัฒนาขึ้นมาให้สามารถใช้งานกับระบบปฏิบัติการแบบต่างๆ ได้หลาย Platform ซึ่งปัจจุบัน (เดือน กันยายน พศ.2554) โปรแกรมของ Arduino ได้รับการปรับปรุงเป็นรุ่น เวอร์ชั่น “Arduino-0022” แล้ว โดยมีโปรแกรมให้เลือกใช้งาน 4 Platform ทั้ง Windows,Mac OSxและ Linux โดยผู้อ่านสามารถเข้าไป ตรวจสอบ หรือ Download โปรแกรมรุ่นใหม่ๆของ Arduino มาใช้งานได้ฟรีโดยไม่เสียค่าใช้จ่ายใดๆจาก “http://arduino.cc/”

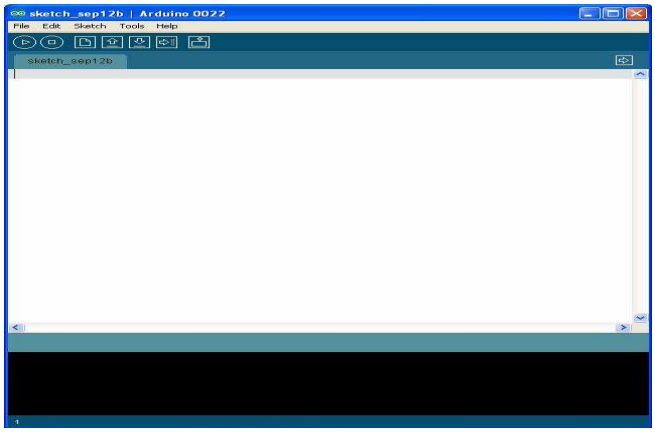
หรือ “http://arduino.cc/en/Main/Software” ซึ่งเป็นเวปไซต์ที่ได้รวบรวมรายละเอียดและข่าวคราวความเคลื่อนไหวต่างๆ เกี่ยวกับ Arduino มากมาย ซึ่งข้อมูลต่างๆจะได้รับการปรับปรุงอย่างต่อเนื่องเป็นประจำโดยในการติดตั้งโปรแกรมของ Arduino นั้นให้ทำการ Unzip แล้ว Copy ไปติดตั้งไว้ในตำแหน่งโฟลเดอร์ “c:\arduino-0022” ดังตัวอย่าง



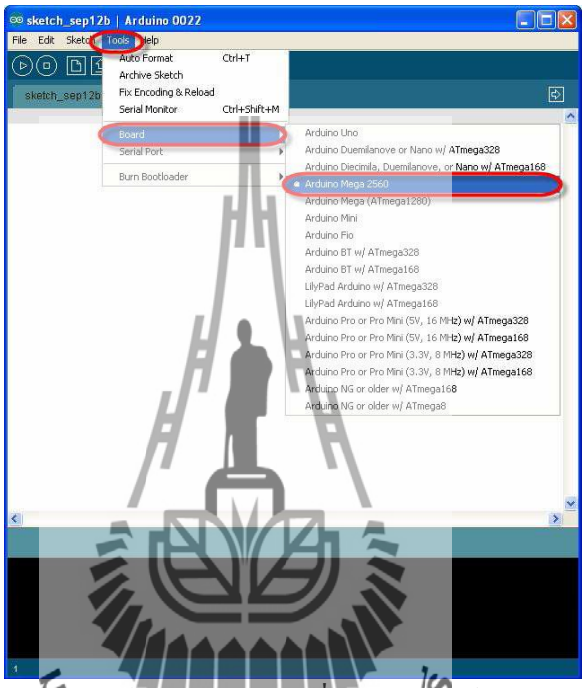
ทดสอบเขียนโปรแกรมใช้งานด้วย Arduino

หลังจากที่เราได้ทำการติดตั้งโปรแกรม Arduino เป็นที่เรียบร้อยแล้ว ก็เป็นอันเสร็จสิ้นขั้นตอนของการเตรียมการแล้ว สำหรับขั้นตอนต่อจากนี้เป็นต้นไป ก็เป็นเรื่องของการใช้งาน การเขียนโปรแกรม และการศึกษาเรียนรู้ต่างๆตามความต้องการแล้ว แต่ก่อนอื่นเราจะต้องทำการติดตั้งโปรแกรมของ Arduino เพื่อใช้เป็นโปรแกรมสำหรับศึกษาเรียนรู้ ซึ่งมีลำดับขั้นตอนดังต่อไปนี้

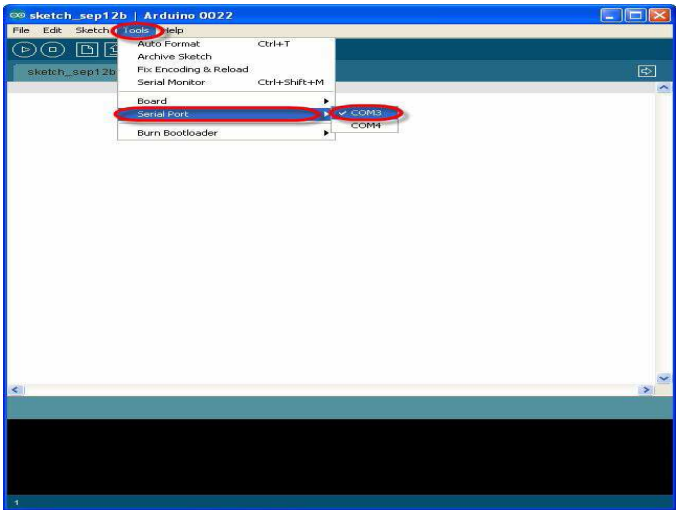
1. ทำการสั่ง Run โปแกรม “arduino.exe” จะได้ผลดังรูป



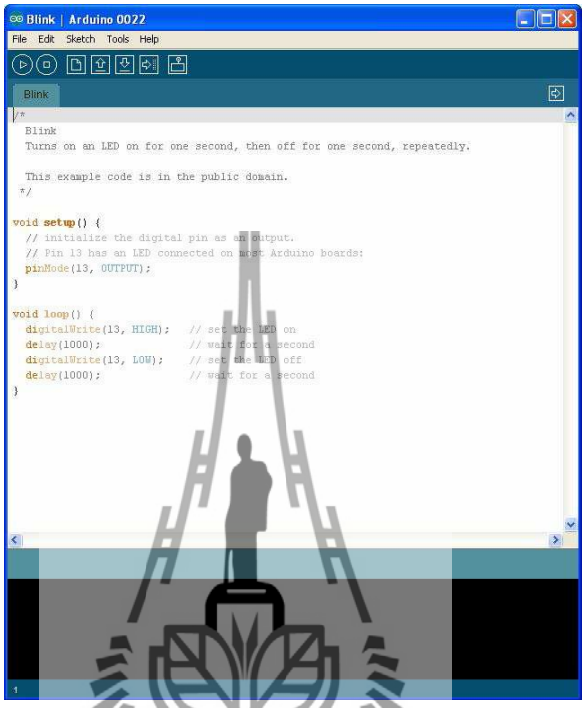
2. ในครั้งแรกของการเรียกใช้งานโปรแกรม ให้ทำการกำหนดระบบฮาร์ดแวร์ที่จะใช้งานกับโปรแกรมของ Arduino ให้เรียบร้อยเสียก่อน เนื่องจากในปัจจุบันนี้ มีการออกแบบวงจรและสร้างฮาร์ดแวร์บอร์ดแบบต่างๆสำหรับนำมาใช้งานร่วมกับโปรแกรมพัฒนาของ Arduino ไว้มากมายหลายรุ่น โดยในกรณีของบอร์ด ET-MEGA2560-ADK ให้ทำการเลือกกำหนดชื่อบอร์ดเป็น “Arduino Mega”โดยคลิกเมาส์ที่ “Tools > Board > “Arduino Mega” ดังรูป



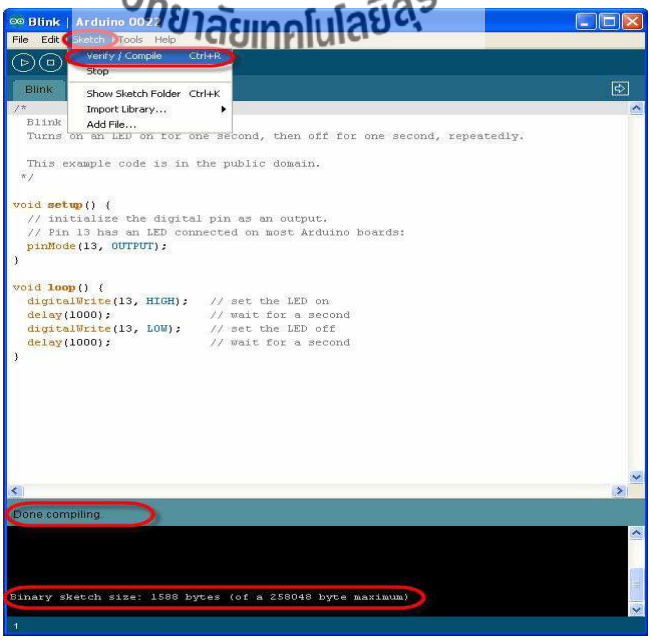
3. เลือกกำหนดหมายเลขพอร์ต สำหรับติดต่อสื่อสารกับบอร์ด ให้ตรงกับหมายเลข Comport ที่ต่อใช้งานไว้จริงในเครื่องคอมพิวเตอร์ PC เช่น กำหนดหมายเลข Comport ของเครื่องคอมพิวเตอร์ PC เป็นCOM3 ให้คลิกเมาส์ที่ Tools > Serial Port > COM3 ดังรูป



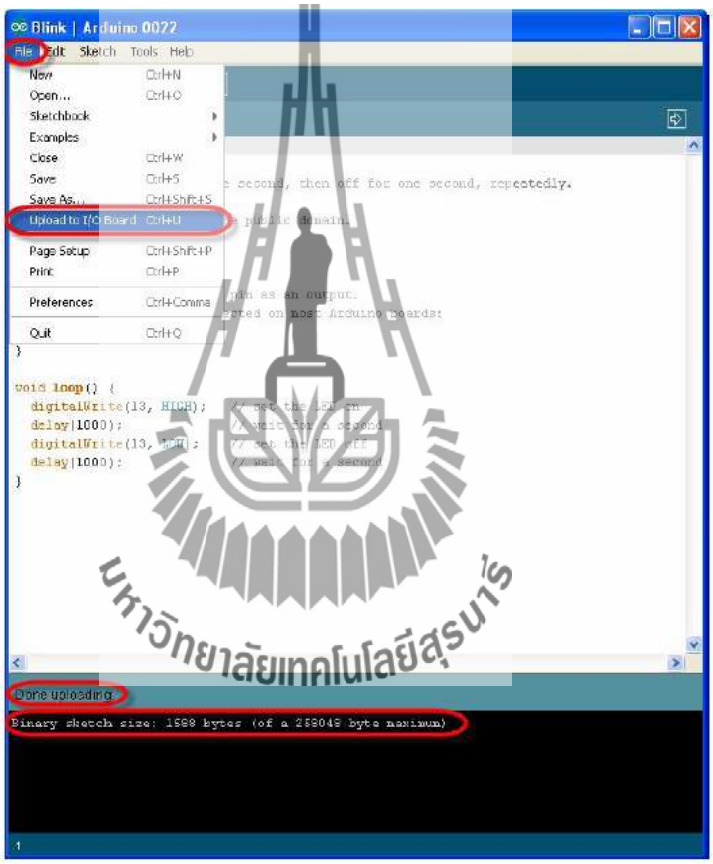
4. ทดสอบเขียนโปรแกรม โดยคลิกเมาส์ที่ File > New แล้วพิมพ์โปรแกรมทดสอบ หรืออาจใช้การสั่งเปิดไฟล์ตัวอย่างที่สร้างไว้แล้วขึ้นมาแทนก็ได้ โดยในที่นี้ขอแนะนำให้ทดสอบด้วยโปรแกรมไฟกระพริบ โดยให้เลือก “File > sketchbook > Examples > Digital > Blink” ซึ่งจะได้ดังรูป



5. สั่งแปลโปรแกรม โดยคลิกเมาส์ที่ “Sketch > Verify/Compile” เพื่อตรวจสอบคำสั่งต่างๆ ในโปรแกรมว่าถูกต้องหรือไม่ ดังตัวอย่าง



6. สั่ง Download Code ให้กับบอร์ด โดยคลิกเมาส์เลือกที่ “File > Upload to I/O Board” แล้วรอสักครู่จนโปรแกรมทำงานเสร็จ หลังจากที่ทำการ Upload Code ให้กับบอร์ดเป็นที่เรียบร้อยแล้วบอร์ดก็จะเริ่มต้นทำงานตามคำสั่งที่เขียนไว้ในโปรแกรมทันที โดยจะสังเกตเห็น LED กระพริบติดและ ดับ สลับกันไปมา ด้วยความเร็วประมาณ 1 วินาที ตลอดเวลา ซึ่งควรได้ผลดังรูป



ภาคผนวก ค
คำสั่ง และตัวแปรต่างๆที่ใช้งานกับโปรแกรม arduino-1.0.4

คำสั่งใช้งาน

contents	true/false
structure	high/low
structure	input/output
setup()	flow control
loop()	if
functions	if... else
{} curly braces	for
; semicolon	while
/*... */ block comments	do... while
// line comments	digital i/o
variables	pinMode(pin, mode)
variables	digitalRead(pin)
variables declaration	digitalWrite(pin, value)
variable scope	analog i/o
datatype	analogRead(pin)
byte	analogWrite(pin, value)
int	time
long	delay(ms)
float	millis()
arrays	math
arithmetic	min(x,y)
arithmetic	max(x,y)
compound assignments	random
comparison operators	randomSeed(seed)
logical operators	random(min, max)
constants	serial
constants	Serial.begin(rate)

Serial.println(data)

appendix

digital output

digital input

high current output

pwm output

potentiometer input

variable resistor input

servo output

โครงสร้างของโปรแกรม

การโปรแกรม Arduino board มีส่วนประกอบอย่างง่ายๆแบ่งเป็นโปรแกรมย่อยหรือฟังชั่นสองส่วน


```
void setup()
{
statements;
}
void loop()
{
statements;
}
```

โปรแกรมย่อยในส่วน setup() คือส่วนการเตรียมการเริ่มต้นใช้งาน ส่วน loop() เป็นส่วนการทำงานหลัก ซึ่งทั้งสองส่วนมีความสำคัญในการทำงานของบอร์ด

ฟังชั่นในส่วน setup() ควรจะมีส่วนที่กำหนดค่าตัวแปรที่ส่วนหัวของโปรแกรม ซึ่งมันจะทำงานเพียงครั้งเดียวและทำงานก่อนโปรแกรมอื่นๆ ส่วนมากโปรแกรมในส่วนนี้จะใช้สำหรับการกำหนดหน้าที่ขาของตัว AVR ไมโครคอนโทรลเลอร์ หรือตั้งค่าการติดต่อแบบ serial กับอุปกรณ์อื่น โปรแกรม loop จะทำงานต่อจากโปรแกรม setup และมันจะทำงานอย่างต่อเนื่องต่อไป เช่นอ่าน

ค่า input ส่งสัญญาณ triggering outputs เป็นต้น โปรแกรมนี้เป็นโปรแกรมทำงานหลักของ Arduino และทำงานส่วนใหญ่ของระบบ

SETUP()

ฟังชั่นนี้จะถูกเรียกใช้งานเพียงครั้งเดียวเมื่อโปรแกรมถูกเริ่มใช้งาน ใช้สำหรับการกำหนดค่าของขาของตัว AVR หรือตั้งค่าการติดต่อแบบ serial โปรแกรมนี้เป็นส่วนจำเป็นที่จะต้องมีถึงแม้มันจะไม่มีคำสั่งภายในตัวมันก็ตาม

```
void setup()
{
pinMode(pin,OUTPUT); // set the 'pin' as output
}
```

LOOP()

หลังจากเรียกโปรแกรม setup() แล้ว โปรแกรม loop() จะทำงานต่อและจะทำงานเป็นวนลูป เพื่อให้โปรแกรมทำงานตามสถานะต่างๆ

```
void loop()
{
digitalWrite(pin, HIGH); // turns 'pin' on
delay(1000); // pauses for one second
degitalWrite(pin, LOW) // turns 'pin' off
delay(1000);
}
```


FUNCTION

ฟังชั่นเป็นกลุ่มคำสั่งที่มีชื่อเรียกและกลุ่มคำสั่งซึ่งจะทำงานเมื่อฟังชั่นถูกเรียกใช้ นอกจากฟังชั่น void setup() และ void loop() แล้วยังมีฟังชั่นอื่นๆที่พร้อมใช้งาน(build-in) ซึ่งจะได้อธิบายต่อไป

ฟังชั่นที่ทำงานเฉพาะสามารถเขียนขึ้นเพื่อให้ทำงานที่ซ้ำๆกันเพื่อลดความยาวของคำสั่ง ในการสร้างฟังชั่นจะเริ่มต้นด้วยการกำหนดชนิดค่าที่ฟังชั่นจะคืนกลับให้โปรแกรมที่เรียกใช้ เช่น int สำหรับค่า integer ถ้าฟังชั่นไม่มีการคืนค่า จะใช้คำว่า void ต่อจากชนิดค่าที่ฟังชั่นคืนกลับจะเป็นชื่อฟังชั่นและตามด้วยวงเล็บซึ่งจะเป็นค่าพารามิเตอร์ที่จะส่งค่าให้กับฟังชั่น

```
type functionName(parameters)
{
statements;
}
```

ตัวอย่างด้านล่างนี้เป็นค่า integer type function delayVal() ที่ใช้ตั้งค่าหน่วงเวลาในโปรแกรมโดยอ่านค่าของ potentiometer การทำงานของฟังชั่นจะกำหนดตัวแปร v และนำค่าของ potentiometer ซึ่งมีค่า 0-1023 มาเก็บไว้ใน v จากนั้นหารค่านั้นด้วย 4 เพื่อให้ได้ค่า 0-255 สุดท้ายจึงส่งค่ากลับไปยัง main program

```
int delayVal()
{
int v; // create temporary variable 'v'
v = analogRead(pot); // read potentiometer value
v /= 4; // converts 0 - 1023 to 0 - 255
return v;
}
```

{} curly braces

วงเล็บปีกกาใช้เป็นเครื่องหมายเพื่อแสดงจุดเริ่มต้นและจุดสิ้นสุดของกลุ่มคำสั่งของฟังชั่น และกลุ่มคำสั่งเช่น void loop() function และกลุ่มคำสั่ง for และ if statement


```
type function()
{
statements;
}
```

วงเล็บปีกกาเปิด { จะต้องตามด้วยวงเล็บปีกกาปิด } เสมอ ในโปรแกรมอาจมีวงเล็บเปิดและปิดหลายๆอันแต่จะต้องมีจำนวนเท่ากัน ถ้าวงเล็บเปิดปิดไม่เท่ากันจะทำให้เกิด compiler error และบางครั้งทำให้หาข้อผิดพลาดได้ยากในโปรแกรมที่มีความซับซ้อน

สำหรับ โปรแกรม Arduino environment จะมีความสามารถในการตรวจความสมดุลย์ของวงเล็บปีกกา โดย คลิกที่ วงเล็บปีกกา หรือคลิกด้านข้างขวาของวงเล็บปีกกาที่ต้องการตรวจสอบ โปรแกรมจะแสดงวงเล็บปีกกาที่เป็นคู่ของมันทันที

; SEMICOLON

เครื่องหมายเซมิโคลอน ; จะต้องใช้ที่ส่วนท้ายของคำสั่งและใช้แยกส่วนของโปรแกรมออกจากกัน นอกจากนี้เซมิโคลอนยังใช้แยกส่วนของ for loop ออกเป็นส่วนๆด้วย

```
int x = 13; // declare variable 'x' as the integer 13
```

หมายเหตุ: ถ้าลืมใส่เซมิโคลอนที่ท้ายบรรทัดจะทำให้ compiler error ซึ่งการ error อาจมาจากหลายสาเหตุแต่การลืมใส่เซมิโคลอนก็เป็นสาเหตุหนึ่ง ให้ตรวจสอบบรรทัดที่ไกล้จุดที่ compiler error ว่าใส่เซมิโคลอนหรือเปล่า

/*....*/ BLOCK COMMENTS

คำอธิบายโปรแกรม สามารถใส่ไว้ในเครื่องหมาย /*.....คำอธิบาย....*/ คำอธิบายที่อยู่ในเครื่องหมาย/*...*/ นี้ตัว compiler จะไม่นำมาประมวลผล ซึ่งจะมีประโยชน์ในการอธิบายความมุ่งหมายของคำสั่ง เพื่อให้เข้าใจการทำงานแต่ละส่วนของโปรแกรม ซึ่งจะเริ่มด้วยเครื่องหมาย /* และจบด้วยเครื่องหมาย */ และสามารถเขียน comment ได้หลายๆบรรทัดถ้าต้องการ

```
/* this is an enclosed block comment
don't forget the closing comment.
they have to be balanced!
*/
```

เนื่องจากการเขียน คำอธิบายไม่ได้เอาไปใช้ในโปรแกรมทำให้ไม่สิ้นเปลืองหน่วยความจำโปรแกรม ดังนั้นควรเขียนคำอธิบายให้ชัดเจนและทำให้เป็นนิสัย อีกอย่างที่เครื่องหมาย comment มีประโยชน์คือใช้ตัดส่วนของโปรแกรมที่ไม่ต้องการออกไปสำหรับการตรวจสอบความผิดพลาดของโปรแกรม

// LINE COMMENTS

การเขียนคำอธิบายโปรแกรมบรรทัดเดียวใช้เครื่องหมาย // เป็นเครื่องหมายเริ่มต้นและสิ้นสุดด้วยการขึ้นบรรทัดใหม่ และเช่นเดียวกับ block comment คือมันจะไม่เปลืองเนื้อที่โปรแกรมใช้งาน เนื่องจาก compiler จะไม่นำไปใช้ในโปรแกรม

// this is a single line comment

คำอธิบายโปรแกรมบรรทัดเดียวใช้บ่อยเพื่ออธิบายผลของกลุ่มคำสั่ง หรืออาจเป็นจุดเตือนในการเขียนคำสั่งต่อๆไป

VARIABLES

variable หรือ ตัวแปรเป็นชื่อและที่เก็บค่าของตัวเลขสำหรับใช้งานในโปรแกรม ตัวแปรมีค่าที่เปลี่ยนแปลงได้ซึ่งตรงกันข้ามกับ constant หรือค่าคงที่ ที่ไม่สามารถเปลี่ยนแปลงค่าได้ การใช้งานตัวแปรจะต้องมีการกำหนดชนิดและอาจกำหนดค่าเริ่มต้นให้กับตัวแปรนั้น ตัวอย่างด้านล่างนี้เป็นการกำหนดตัวแปรชื่อ inputVariable หลังจากนั้นนำค่าที่ได้จาก analog input pin 2 มาเก็บไว้

```
int inputVariable = 0; // declares a variable and assigns value of 0
inputVariable = analogRead(2); // set variable to value of anlog pin 2
```

ค่าของตัวแปร inputVariable จะเปลี่ยนแปลงได้ บรรทัดแรกจะกำหนดให้มันเป็นชนิด int หรือตัวเลขจำนวนเต็มโดยมีค่าเริ่มต้นเป็น 0 บรรทัดที่สองจะนำค่าจาก analog pin 2 มาเก็บไว้ ซึ่งจะทำให้ค่าของ pin 2 จะถูกเรียกใช้ใน โปรแกรมได้

เมื่อตัวแปรถูกสร้างขึ้น, หรือใส่ค่าใหม่ให้มัน, คุณสามารถตรวจสอบค่าของมันว่ามันมีค่าตามที่ต้องการ, หรือคุณจะนำค่านั้นไปใช้เลยก็ได้ ตัวอย่างด้านล่างนี้เป็นสามตัวอย่างที่ใช้งานตัวแปร, คือตรวจสอบค่าตัวแปร inputVariable ว่ามีค่าต่ำกว่า 100 หรือไม่ ถ้ามีค่าต่ำกว่า 100 ให้ใส่ค่า 100 ให้กับตัวแปร หลังจากนั้นตั้งค่าหน่วงเวลาตามค่าตัวแปร inputVariable ซึ่งก็คือค่า 100 เป็นอย่างต่ำ

```
if (inputVariable < 100) // test variable if less than 100
{
inputVariable = 100; // if true assigns value of 100
}
delay(inputVariable); // uses variables as delay
```

หมายเหตุ: ตัวแปรควรจะมีชื่อที่สื่อถึงหน้าที่ใช้งาน, เพื่อให้โปรแกรมสามารถอ่านได้ง่าย ชื่อตัวแปรเช่น tiltSensor หรือ pushButton จะช่วยให้โปรแกรมเมอร์หรือคนอื่นที่อ่านโปรแกรมสามารถเข้าใจได้ว่าตัวแปรนั้นเอาไว้ทำอะไร คุณสามารถตั้งชื่ออะไรก็ได้ที่ไม่ตรงกับชื่อเฉพาะที่ใช้ใน Arduino language

variable declaration

ตัวแปรที่จะนำไปใช้งานได้จะต้องถูกตั้งค่าตัวแปรก่อนนำไปใช้ การตั้งค่าตัวแปรหมายถึงการระบุชนิดตัวแปร ให้เป็น int, long, float, เป็นต้น การกำหนดชื่อให้ตัวแปรมีชื่อเรียก และอาจมีการกำหนดค่าเริ่มต้นให้กับตัวแปร ซึ่งการตั้งค่านี้จะทำเพียงครั้งเดียวในโปรแกรม แต่ค่าของตัวแปรจะเปลี่ยนแปลงไปตามการทำงานของโปรแกรม

ตัวอย่างต่อไปนี้ตั้งค่าตัวแปร inputVariable ให้เป็นตัวแปรจำนวนเต็ม int,หรือ integer, และใส่ค่าเริ่มต้นให้เป็นศูนย์ ซึ่งเป็นการตั้งค่าตัวแปรแบบง่าย

```
int inputVariable = 0;
```

ตัวแปรสามารถตั้งค่าในแบบระบุตำแหน่งในหน่วยความจำ

variable scope

ตัวแปรสามารถตั้งค่าตอนเริ่มต้นโปรแกรมก่อน void setup(), หรือตั้งค่าตัวแปรภายในฟังชั่น,และบางครั้งก็ตั้งค่าตัวแปรภายในกลุ่มคำสั่ง for loop ซึ่งการตั้งค่าตัวแปรในแบบต่างๆ มีผลถึงขอบเขตการใช้ตัวแปร, หรืออีกนัยหนึ่งการที่โปรแกรมจะสามารถใช้ตัวแปรนั้น

ตัวแปรแบบ global เป็นตัวแปรที่โปรแกรมมองเห็นและใช้งานได้จากทุกฟังชั่นและทุกกลุ่มคำสั่งในโปรแกรม ตัวแปรนี้จะตั้งค่าที่ตอนเริ่มต้นโปรแกรม, ก่อน setup() function

ตัวแปรแบบ local เป็นตัวแปรที่ตั้งค่าภายในฟังชั่นหรือภายในกลุ่มคำสั่ง for loop ตัวแปรนี้จะมองเห็นและใช้งานได้เฉพาะภายในฟังชั่นที่มันตั้งค่า ชื่อตัวแปรแบบนี้อาจมีชื่อซ้ำกันในแต่ละฟังชั่น แต่ในการทำงานฟังชั่นแต่ละตัวจะใช้ตัวแปรตัวนั้นเฉพาะที่ตั้งค่าภายในตัวมันเท่านั้น

ตัวอย่างด้านล่างนี้แสดงการตั้งค่าตัวแปรและขอบเขตการใช้งานของตัวแปร

```
int value; // 'value' is visible to any function
void setup();
{
// no setup neededd
```

```
}
void loop()
{
for (int i=0; i<20;) // 'i is only visible inside the for - loop
{
i++;
}
float f; // 'f' is only visible inside loop
}
```

byte

ตัวแปร byte เก็บตัวเลข 8 bit ไม่มีทศนิยม มีค่า 0 - 255

```
byte someVariable = 180; // declares 'someVariable' as a byte type
```

int

integer เป็นตัวแปรพื้นฐานที่เก็บตัวเลข โดยไม่มีจุดทศนิยม และเก็บค่า 16 bit มีค่าระหว่าง 32,767 ถึง -32,768

```
int someVariable = 1500; // declares 'someVariable' as an integer type
```

หมายเหตุ: ตัวแปร integer จะมีค่าเป็นลบหรือ roll over ถ้ามันถูกใส่ค่าเกินจากค่าสูงสุดที่มันรับได้ เช่นถ้า x = 32767 และคำสั่งถัดมาให้เพิ่มค่า 1 ให้กับ x , x = x + 1 หรือ x++, ค่า x จะล้น หรือที่เรียกว่า roll over เป็นค่า -32768

long

เป็นตัวแปรจำนวนเต็มแบบขยายโดยไม่มีจุดทศนิยม เก็บค่าแบบ 32 bit มีค่าระหว่าง 2,147,483,647 ถึง -2,147,483,648

```
long someVariable = 90000; // declares 'someVariable' as a long type
```

float

ตัวแปรชนิด floating-point หรือตัวแปรที่มีจุดทศนิยม ตัวแปรนี้มีค่ามากกว่าค่าของตัวแปรจำนวนเต็ม โดยใช้เนื้อที่เก็บ 32 bit มีค่าระหว่าง 3.4028235E+38 ถึง -3.4028235E+38

float someVariable = 3.14; // declares 'someVariable' as a floatint -point type

หมายเหตุ: ค่าตัวเลขทศนิยมเป็นค่าที่ไม่เต็มจำนวน และอาจส่งผลแปลกๆเมื่อนำมาเปรียบเทียบกับค่าอื่น และการคำนวนเลขทศนิยมใช้เวลามากว่าการคำนวนเลขจำนวนเต็ม, ดังนั้นควรหลีกเลี่ยงใช้งานถ้าทำได้

arrays

ตัวแปร arrays หรือตัวแปรหลายมิติเป็นตัวแปรที่สามารถเข้าถึงได้ด้วยค่าตัวชี้หรือ index ค่าตัวแปรใน array อาจเรียกใช้โดยระบุชื่อ array และระบุตัวชี้ index number ตัวแปร array จะมี index เริ่มต้นจาก 0 ตัวแปร array จะต้องตั้งค่า ก่อนจะนำไปใช้งาน และอาจกำหนดค่าเริ่มต้นหรือไม่ก็ได้

int myArray[] = {value0, value1, value2...}

เราอาจตั้งค่า array โดยกำหนดชนิดและขนาด แล้วค่อยใส่ค่าลงไปทีหลังก็ได้

int myArrays[5]; // declares integer array w/ 6 positions

myArray[3] = 10; // assigns the 4th index the value 10

การดึงค่าจาก array ใช้การเรียกค่าจากตัวแปร array และระบุตำแหน่งตัวชี้ index

x = myArray[3]; // x now equals 10

ตัวแปร arrays ใช้บ่อยใน for loops, ซึ่งจะมีการเพิ่มค่าตัวนับ และใช้ตัวนับเป็นตัวชี้ตำแหน่งของ arrays แต่ละตัว ตัวอย่างต่อไปนี้เป็นการใช้ array ในการสั่ง LED ติดดับ โดยใช้ for loop, ตัวนับจะเริ่มจาก 0, เขียนค่าที่ได้จาก array ตำแหน่งที่ 0 ของ array flicker[], ในที่นี้คือค่า 180, ไปยัง PWM pin 10, หยุดเป็นเวลา 200 ms, แล้วจึงย้ายไปเอาค่าในตำแหน่งถัดไปของ array

int ledPin = 10; // LED on pin 10

byte flicker[] = {180, 30, 255, 200, 10, 90, 150, 60}; // array of 8 different values

```
viod setup()
{
pinMode(ledPin, OUTPUT); // sets OUTPUT pin
}
viod loop()
{
for(int i=0; i<7; i++) // loop equals number of values in array
{
analogWrite(ledPin, flicker[i]); // write index value
delay(200);
}
}
```

arithmetic

ตัวกระทำการทางคณิตศาสตร์หมายรวมถึง การบวก, ลบ, คูณ, และหาร ซึ่งมันจะให้ค่าผลรวม, ผลต่าง, ผลคูณ, หรือผลการหาร ของสองตัวดำเนินการ

```
y = y + 3;
x = x - 7;
i = j * 6;
r = r / 5
```


การทำงานของโปรแกรมจะเป็นการนำขนิดข้อมูลตัวแปรของตัวดำเนินการมาใช้, ดังนั้น, จากตัวอย่าง 9/4 ผลลัพท์จะเท่ากับ 2 แทนที่จะเท่ากับ 2.25 เนื่องจาก 9 และ 4 เป็นตัวแปรจำนวนเต็ม หรือ integer ซึ่งไม่สามารถเก็บค่าทศนิยมได้ และถ้าผลลัพท์ที่ได้มีค่ามากกว่าที่ตัวแปรจะรับได้มันจะเกิดการ overflow

ถ้ามีการคำนวนกับชนิดตัวแปรที่ต่างกัน, ตัวแปรที่มีขนาดใหญ่กว่าจะถูกนำมาใช้ เช่นการบวก ลบเลขจำนวนเต็มกับเลขทศนิยม การบวกลบทศนิยมจะถูกนำมาใช้ควรเลือกตัวแปรที่มีขนาดที่ใหญ่พอที่จะเก็บผลลัพท์การคำนวน ตรวจสอบจุดที่ตัวแปรจะมีค่าย้อนกลับหรือ roll over เช่น 0-1 หรือ 0-32768 สำหรับการคำนวนที่เป็นเลขเศษส่วน ใช้ตัวแปรที่มีจุดทศนิยม แต่ควรระวังผลข้างเคียงถ้าตัวเลขมากจะทำให้ความเร็วในการคำนวนช้าลง

หมายเหตุ: การ cast ตัวแปรคือการเปลี่ยนชนิดตัวแปรจากชนิดหนึ่งไปยังอีกชนิดหนึ่งคือ (int)myFloat เช่น i = (int)3.6 จะเป็นการแปลงค่าให้ i มีค่าเป็น 3

compound assignment

การเขียนคำสั่งโปรแกรมแบบย่อหมายถึงการทำงานกับตัวแปรแล้วนำผลลัพท์ที่ได้ไปเก็บในตัวแปรนั้น จะพบมากใน for loop การเขียนคำสั่งแบบย่อหรือ compound assignment ที่ใช้ทั่วไปคือ

x ++ // same as x = x + 1, or increments x by +1

x -- // same as x = x - 1, or decrements x by -1

x += y // same as x = x + y, or increments x by +y

x -= y // same as x = x - y, or decrements x by -y

x *= y // same as x = x * y, or multiply x by y

x /= y // same as x = x / y, or devides x by y

หมายเหตุ: ตัวอย่างเช่น x *= 3 จะคูณค่า x เป็นสามเท่าแล้วนำผลลัพท์เก็บใน x

comparison operator

การเปรียบเทียบตัวแปรหรือค่าคงที่กับค่าอื่นๆ จะใช้บ่อยในกลุ่มคำสั่ง if เพื่อตรวจค่าว่ามีสถานะตรงตามที่ต้องการหรือไม่ ตัวอย่างต่อไปนี้เป็นคำสั่งเปรียบเทียบตัวแปรสองตัว

x == y // x is equal to y ??

x != y // x is not equal to y ??

x < y // x is less than y ??

x > y // x is greater than y ??

x <= y // x is less than or equal to y ??

x >= y // x is greater than or equal to y ??

logical operators

การตรวจสอบทาง logic จะใช้เปรียบเทียบตัวดำเนินการสองตัวและให้ผลลัพท์ จริง หรือ เท็จ(TRUE OR FALSE) ขึ้นอยู่กับตัวดำเนินการ ตัวดำเนินการทางลอจิกมี 3 ตัวคือ AND, OR และ NOT ซึ่งใช้บ่อยใน กลุ่มคำสั่ง if ดังตัวอย่างต่อไปนี้

Logical AND:

if (x > 0 && x < 5) // true only if both expression are true

// เป็นจริงเมื่อทั้งสองส่วนเป็นจริง

Logical OR:

if (x > 0 || y < 5) // true only if either expression are true

// เป็นจริงเมื่อตัวใดตัวหนึ่งเป็นจริง

constants

ในภาษา Arduino มีค่าที่กำหนดไว้แล้วไม่มาก เราเรียกค่าพวกนี้ว่า constant หรือค่าคงที่ ซึ่งมีไว้เพื่อให้โปรแกรมอ่านง่าย ค่าคงที่เหล่านี้จะแบ่งออกได้หลายกลุ่ม

true/false

ค่าเหล่านี้เป็นค่าคงที่ boolean ซึ่งบอกสถานะระดับลอจิก FALSE หมายถึง 0(ศูนย์) ในขณะที่ TRUE จะหมายถึง 1, หรืออะไรก็ได้ที่ไม่ใช่ ศูนย์ ดังนั้นในทางลอจิกแล้ว -1, 2, -200 จะหมายถึง TRUE

if (b == TRUE);

{

doSomething;

}

high/low

ค่าคงที่เหล่านี้แสดงถึงระดับลอจิกที่ขาไอซีว่าเป็น HIGH หรือ LOW และใช้เมื่อมีการอ่านหรือเขียนไปที่ขาไอซี HIGH จะแทนระดับลอจิก 1, ON, หรือ 5 volts ในขณะที่ LOW คือระดับลอจิก 0, OFF, หรือ 0 volts

digitalWrite(13, HIGH)

input/output

ค่าคงที่ใช้ในฟังชั่น pinMode() ในการกำหนดหน้าที่ของ digital pin ว่าจะให้เป็น INPUT หรือ OUTPUT

pinMode(13, OUTPUT);

if

คำสั่ง if จะใช้ตรวจหาสถานะที่ต้องการว่าเกิดขึ้นหรือยัง, เช่นค่า analog ที่อ่านได้มีค่าเกินจำนวนที่ตั้งไว้หรือยัง และทำงานตามคำสั่งในวงเล็บปีกกาถ้าสถานะเป็นจริง(TRUE) แต่ถ้าสถานะเป็นเท็จ(FALSE)โปรแกรมจะข้ามกลุ่มคำสั่งนั้นไป ดังตัวอย่าง

```
if (someVariable ?? value)
{
doSomething;
}
```

ตัวอย่างข้างบนเป็นการเปรียบเทียบ someVariable กับค่าอื่น, ซึ่งอาจเป็นตัวแปรหรือค่าคงที่ ถ้าผลลัพท์เป็นจริง คำสั่งในวงเล็บปีกกาจะถูกทำงาน แต่ถ้าไม่ โปรแกรมจะข้ามไปทำงานต่อที่หลังวงเล็บปีกกาปิด

หมายเหตุ: ระมัดระวังการใช้ '=' ในรูปคำสั่ง if (x=10), คำสั่งนี้ถูกต้องตามรูปแบบคือนำค่า 10 ไปใส่ในตัวแปร x และจะทำให้ผลลัพท์เป็นจริงตลอด อาจทำให้เกิดผลที่ไม่คาดคิด ดังนั้นควรใช้ '==' เช่น if (x==10), ซึ่งเพียงแต่ตรวจค่า x ว่ามีค่าเท่ากับ 10 หรือไม่ เท่านั้น จำไว้ว่า '=' หมายถึง 'เท่ากับ' ตรงกันข้ามกับ '==' ที่หมายถึง 'เท่ากับใช่หรือไม่'

if... else

if... else เป็นการเพิ่มทางเลือก 'ถ้าไม่ใช่สถานะที่ต้องการ' ให้มีทางเลือกอีกทาง ตัวอย่างเช่น ถ้าคุณต้องการตรวจสอบ digital input และทำงานอย่างหนึ่งถ้าอินพุทเป็น HIGH หรือทำงานอีกอย่างถ้าอินพุทเป็น LOW คุณจะเขียนคำสั่งในลักษณะนี้

```
if (inputPin == HIGH)
{
doThingA;
}
else
{
doThingB;
}
```

else

สามารถขยายเงื่อนไขต่อเนื่องออกไปได้อีก ทำให้เราสามารถทดสอบเงื่อนไขได้หลายเงื่อนไขในเวลาเดียวกัน เราสามารถตั้งเงื่อนไขสำหรับการทดสอบได้อย่างไม่จำกัด แต่จำไว้ว่าเงื่อนไขทั้งหมดจะมีกลุ่มคำสั่งเดียวที่ทำงานเนื่องจากสถานะที่ทดสอบตรงตามเงื่อนไข

```
if (inputPin < 500)
{
doThingA;
}
else if (inputPin >= 1000)
{
doThingB;
}
else
{
doThingC;
}
```

หมายเหตุ: คำสั่ง if จะทดสอบเงื่อนไขในวงเล็บว่าจริงหรือเท็จ ซึ่งเงื่อนไขจะเป็นคำสั่งในภาษา C ดังตัวอย่างแรก, if (inputPin == HIGH) ในตัวอย่างนี้ คำสั่ง if จะตรวจสอบว่า inputPin เป็น HIGH หรือ +5V

for

คำสั่ง for จะใช้เพื่อทำกลุ่มคำสั่งในวงเล็บปีกกาหลายๆครั้งเป็นจำนวนที่ต้องการ ปกติจะใช้ตัวนับจำนวนเพิ่มจำนวนครั้งที่ทำงานจนถึงจำนวนที่ต้องการแล้วจึงออกจาก loop เงื่อนไขของการทำงานใน loop จะกำหนดโดยคำสั่งที่ส่วนหัวของ loop ซึ่งมีสามส่วนแยกกันด้วยเซมิโคลอน

```
for (initialization; condition; expression)
{
doSomething;
}
```

ส่วนของการเริ่มต้นค่าตัวแปรหรือที่เรียกว่า initialization จะทำงานก่อนและทำงานเพียงครั้งเดียว และเมื่อทำงานครบหนึ่งรอบ ส่วนทดสอบรอบการทำงานหรือส่วน condition จะถูกทดสอบถ้าเงื่อนไขเป็นจริง กลุ่มคำสั่ง expression จะถูกทำงานและเงื่อนไขจะถูกทดสอบอีกครั้ง ถ้าเงื่อนไขเป็นเท็จ จึงจะออกจาก loop

ตัวอย่างต่อไปนี้จะตั้งค่าเริ่มต้นให้ตัวแปร i เป็นศูนย์, ทดสอบว่ามันยังมีค่าต่ำกว่า 20 และถ้าเป็นจริง, เพิ่มค่า i ด้วย 1 และทำคำสั่งในวงเล็บปีกกา

```
for (int i=0; i < 20; i++) // declares i, test if less than 20, increment i by 1
{
degitalWrite(13, HIGH); // turn pin 13 on
delay(250); // pauses for 1/4 second
degitalWrite(13, LOW); //turn pin 13 off
delay(250); // pauses for 1/4 second
}
```

หมายเหตุ: คำสั่ง for loop ในภาษา C มีความซับซ้อนกว่า for loop ในภาษาคอมพิวเตอร์อื่นบางภาษารวมทั้งภาษาเบสิก บางส่วนของส่วนหัวอาจไม่มีกลุ่มคำสั่งแต่ก็ต้องใส่เครื่องหมายเซมิโคลอนไว้ คำสั่งของกลุ่มคำสั่งในส่วนหัวอันได้แก่ส่วน initialization, condition, และ expression จะเป็นคำสั่งภาษา C ที่มีส่วนตัวแปรที่ไม่เกี่ยวข้องกันก็ได้ ขึ้นอยู่กับการออกแบบของโปรแกรมเมอร์

while

while loops จะทำงานอย่างต่อเนื่องตลอดไป จนกระทั่งเงื่อนไขในวงเล็บจะเป็นเท็จ ดังนั้นจะต้องมีเหตุการณ์บางอย่างมาเปลี่ยนแปลงค่าตัวแปรที่ใช้ทดสอบ หรือการทำงานใน while loop อาจทำงานไปตลอดโดยไม่มีการออกจาก loop ขึ้นอยู่กับคำสั่งที่คุณเขียน เช่นเพิ่มค่าตัวแปร, หรือทดสอบสถานะภายนอก, เช่นตรวจสอบ sensor เป็นต้น

```
while (someVariable ?? value) // test if less than 200
{
doSomething; // executes enclosed statements
someVariable++; // increments variable by 1
}
```

do... while

do loop จะทำงานและทดสอบสถานะการทำงานที่ด้านท้ายของ loop คล้ายกับการทำงานของ while loop ยกเว้นการตรวจสอบสถานะเพื่อออกจาก loop จะทำที่ส่วนท้ายของ loop ดังนั้น do loop จะทำงานอย่างน้อยหนึ่งครั้งเสมอ

```
do
{
doSomething;
} while (someVariable ?? value);
```

ตัวอย่างต่อไปนี้อ่านค่า readSensor() แล้วเก็บไว้ที่ตัวแปร 'x', หยุดเป็นเวลา 50 มิลลิวินาที, และทำงานวนลูปจนกระทั่ง 'x' มีค่าน้อยกว่า 100

```
do
{
x = readSensor(); // assigns the value of readSensors() to x
delay(50); // pauses 50 milliseconds
} while (x < 100); // loop if x is less than 100
```

pinMode(pin, mode)

ใช้ในกลุ่ม void setup() เพื่อกำหนดหน้าที่ขาของไมโครคอนโทรลเลอร์ให้เป็น ขารับสัญญาน INPUT หรือขาส่งสัญญาน OUTPUT

```
pingMode(pin, OUTPUT); // sets 'pin' to output
```

ขาของ Arduino digital ถูกตั้งให้เป็นขารับอินพุทโดยปริยาย ดังนั้นจึงไม่จำเป็นต้องสั่งให้มันทำหน้าที่เป็นอินพุทด้วยคำสั่ง pinMode() ขาที่ถูกตั้งให้เป็นอินพุทจะมีคุณสมบัติเป็นขาที่มีอิมพีแดนท์สูง

นอกจากนี้ยังมีความต้านทานพูลอัพ 20K ที่พร้อมใช้งานในตัว Atmega chip ซึ่งสามารถสั่งให้ทำงานได้ด้วยซอฟท์แวร์ โดยคำสั่งดังนี้

```
pinMode(pin, INPUT); // set 'pin' to input
digitalWrite(pin, HIGH); // turn on pullup resistors
```

ความต้านทานพูลอัพปกติจะใช้ต่อกับอุปกรณ์อินพุทเช่นสวิทซ์ ตัวอย่างข้างบนไม่ได้ทำให้ขาเปลี่ยนเป็นขา output แต่เป็นวิธีตั้งให้ความต้านทางพูลอัพทำงานเท่านั้น

ขาที่กำหนดให้เป็น OUTPUT เรียกได้ว่าเป็นขาที่ทำให้มีอิมพีแดนท์ต่ำและสามารถจ่ายกระแสได้ 40 มิลลิแอมป์ให้อุปกรณ์อื่น ซึ่งเพียงพอที่จะขับ LED ให้ติด(อย่าลืมใส่ความต้านทานจำกัดกระแสด้วย) อย่างไรก็ตามมันไม่สามารถขับรีเลย์, โซลีนอยด์, หรือมอเตอร์ได้โดยตรง

การลัดวงจรบนขาของ Arduino และการจ่ายกระแสมากเกินไปอาจทำความเสียหายให้กับขา output หรืออาจทำให้ตัวไอซีเสียหาย ดังนั้นควรต่อความต้านทานค่า 470 โอห์มหรือ 1k โอห์มอนุกรมกับอุปกรณ์ภายนอก

digitalRead(pin)

คำสั่งนี้อ่านค่าจาก ขาไอซีที่ถูกกำหนดให้เป็น digital pin ซึ่งจะได้ผลลัพท์เป็น HIGH หรือ LOW หมายเลขขาไอซีอาจกำหนดเป็นตัวแปรหรือค่าคงที่ (0-13)

```
value = digitalRead(pin); // sets 'value' equal to the input pin
```

digitalWrite(pin, value)

ส่งค่าลอจิก HIGH หรือ LOW (เปิด หรือ ปิด) ไปยังขา digital ที่กำหนด หมายเลขขาไอซีอาจกำหนดเป็นตัวแปรหรือค่าคงที่ (0-13)

```
digitalWrite(pin, HIGH); // sets 'pin' to high
```

ตัวอย่างต่อไปนี้อ่านค่าจากปุ่มกดที่ต่อกับ digital input และเปิด LED ที่ต่อกับ digital output เมื่อปุ่มถูกกด

```
int led = 13; // connect LED to pin 13
int pin = 7; // connect pushbutton to pin 7
int value = 0; // variable to store the read value
void setup()
{
pinMode(led, OUTPUT); // sets pin 13 as output
pinMode(pin, INPUT); // sets pin 7 as input
}
```

```
void loop()
{
value = digitalRead(pin); // sets 'vaue' equal to the input pin
digitalWrite(led, value); // sets 'led' to the button's value
}
```

analogRead(pin)

คำสั่งนี้อ่านค่าจากขา Analog จะได้ค่า 10 bit คำสั่งนี้จะทำงานกับขา analog input (0-5) เท่านั้น และได้ผลลัพท์เป็นเลขจำนวนเต็มค่า 0 - 1023

```
value = analogRead(pin); // sets 'value' equal to 'pin'
```

หมายเหตุ: ขา Analog ไม่เหมือนกับ ขา digital, ไม่ต้องกำหนดตอนเริ่มต้นว่าเป็น INPUT หรือ OUTPUT

analogWrite(pin, value)

เป็นคำสั่งเขียนค่า analog เทียมโดยใช้ hardware enabled pulse width mdulation(PWM) ไปยังขา outut ที่สามารถทำ PWM ได้ ใน Arduino รุ่นใหม่ที่ใช้ชิพ Atmega168 คำสั่งนี้จะทำงานกับขา 3, 5, 6, 9, 10, และ 11 ส่วน Arduino รุ่นเก่าที่ใช้ Atmega8 จะรองรับเพียงขา 9, 10 และ 11 ค่าที่เขียนสามารถใช้เป็นตัวแปรหรือค่าคงที่จาก 0 – 255


```
analogWrite(pin, value); // writes 'value' to analog 'pin'
```

ค่า 0 ที่เขียนไปยังขา analog จะทำให้เกิด 0 โวลท์คงที่ที่ขานั้น ส่วนค่า 255 จะทำให้เกิด 5 โวลท์คงที่ที่ขานั้นเช่นกัน สำหรับค่าระหว่าง 0 - 255 ขาจะเปลี่ยนไปเป็นค่าที่อยู่ระหว่าง 0 และ 5 โวลท์ ค่ายิ่งมากแรงดันที่ปรากฏก็ยิ่งเข้าใกล้ HIGH(5 โวลท์) ตัวอย่างเช่น ค่า 64 จะทำให้เกิด 0 โวลท์เป็นสามในสี่ของเวลาทั้งหมด และเป็น 5 โวลท์หนึ่งในสี่ที่เหลือ ถ้าค่าที่เขียนคือ 128 จะทำให้เกิด 0 ครึ่งหนึ่งของคาบเวลาและ 255 อีกครึ่งคาบเวลา ถ้าเป็นค่า 192 จะทำให้เกิด 0 โวลท์หนึ่งในสี่และ 5 โวลท์ สามในสี่ของคาบเวลา

เนื่องจากคำสั่งนี้เป็นคำสั่งให้ hardware ในตัวไอซีทำงาน ขาไอซีจะผลิตคลื่นที่มีคาบเวลาตามค่าที่เขียนอย่างคงที่หลังจากทำคำสั่ง analogWrite ไปเรื่อยๆ จนกระทั่งมีการเรียกใช้คำสั่ง analogWrite ครั้งใหม่ (หรือการเรียกใช้คำสั่ง digitalRead หรือ digitalWrite บนขาไอซีขาเดียวกัน)

หมายเหตุ: ขา Analog ไม่เหมือนกับ ขา digital, ไม่ต้องกำหนดตอนเริ่มต้นว่าเป็น INPUT หรือ OUTPUT

ตัวอย่างต่ไปนี้ อ่านค่า analog จากขา analog input pin, เปลี่ยนค่าที่อ่านได้ด้วยการหาร 4 และส่งค่าที่ได้เป็นค่า PWM ออกทางขา PWM

```
int led = 10; // LED with 220 resistor on pin 10
int pin = 0; // potentiometer on analog pin 0
int value; // value for reading
void setup(){} // no setup needed
void loop()
{
value = analogRead(pin); // sets 'value' equal to 'pin'
value /= 4; // converts 0 - 1023 to 0 - 255
analogWrite(led, value); // outputs PWM signal to led
}
```


delay(ms)

คำสั่งนี้จะหยุดการทำงานของโปรแกรมเป็นเวลาตามที่กำหนดเป็นมิลลิวินาที ซึ่ง 1000 วินาทีเท่ากับ 1 วินาที

```
delay(1000); // waits for one second
```

millis()

คำสั่งนี้จะได้ผลลัพท์ค่าเวลาเป็นมิลลิวินาทีแสดงค่าที่ Arduino board1 เริ่มต้นทำโปรแกรมปัจจุบัน ค่าที่ได้เป็นค่า unsigned long ขนาด 32 bit

```
value = missis(); // sets 'value' equal to missis()
```

หมายเหตุ: ถ้า run โปรแกรมอย่างต่อเนื่องตัวเลขจำนวนนี้จะมีค่าย้อนกลับเป็นศูนย์ หลังจากเวลาผ่านไปประมาณ 9 ชั่วโมง

min(x,y)

คำสั่งนี้คำนวนหาค่าที่น้อยกว่าของค่าที่ให้มาในวงเล็บและคืนค่าที่น้อยก่วา

value = min(value, 100); // sets 'value to the smaller of 'value' or 100, ensureing that

// it never gets above 100.

max(x,y)

คำสั่งนี้คำนวนหาค่าที่มากกว่าของค่าที่ให้มาในวงเล็บและคืนค่าที่มากกว่า

value = max(value, 100); // sets 'value to the larger of 'value' or 100, ensureing that

// it is at least 100.

randomSeed(seed)

กำหนดค่าเริ่มต้นของฟังชั่น random

randomSeed(value); // sets 'value' as the random seed

เนื่องจาก Arduino ไม่สามารถสร้างตัวเลขสุ่มได้เอง คำสั่ง randomSeed จะอนุญาตให้เราใส่ค่าตัวแปร, ค่าคงที่, หรือค่าจากฟังชั่นอื่นลงไปในฟังชั่น random เพื่อให้มันสามารถสร้างตัวเลขสุ่มได้อย่างมีประสิทธิภาพ ค่าจากฟังชั่นที่จะใส่ลงไปมีทั้งค่าจาก millis() หรือ analogRead() เป็นต้น

random(max)

random(min, max)

เป็นฟังชั่นที่ให้ค่าตัวเลขสุ่มที่มีค่าตัวเลขระหว่างตัวเลขจ้อยกับตัวเลขมากที่กำหนด

value = random(100, 200); // sets 'value' to a random number between 100-200

หมายเหตุ: ควรใช้คำสั่งนี้หลังจากใช้คำสั่ง randomSeed()

ตัวอย่างต่อไปนี้จะสร้างตัวเลขสุ่มมีค่าระหว่าง 0 - 255 และส่งตัวเลขสุ่มนี้ไปสร้างสัญญาณ PWM ออกไปที่ขา PWM

int randNumber; // variable to store the random value

int led = 10; // LED with 220 resistor on pin 10

```
void setup(){} // no setup needed
void loop()
{
randomSeed(millis()); // sets millis() asa seed
randNumber = random(255); // random number from 0 - 255
analogWrite(led, randNumber); // outputs PWM signal
delay(500); // pauses for half a second
}
```

Serial.begin(rate)

คำสั่งนี้จะเปิดพอร์ท serial และตั้งค่าความเร็วในการรับส่ง ปกติการติดต่อกับเครื่องพีซีจะตั้งไว้ที่ ความเร็ว 9600 แม้ว่ามันสามารถรับส่งได้ที่ความเร็วมากกว่านี้ก็ตาม

```
void setup()
{
Serial.begin(9600); // opens serial port set data rate to 9600 bps
}
```

หมายเหตุ: เมื่อใช้พอร์ทสำหรับการติดต่อ serial communication ,ขา digital pin 0 (RX) และ digital pin 1 (TX) จะนำไปใช้งานอย่างอื่นไม่ได้

Serial.println(data)

คำสั่งนี้จะส่งข้อมูลให้กับ serial port, ตามด้วยขึ้นบรรทัดใหม่โดยอัตโนมัติ คำสั่งนี้จะมีผลเหมือนกับคำสั่ง Serial.print() แต่สามารถดูข้อมูลได้ที่ Serial monitor ได้ด้วย

```
Serial.println(analogvalue); // sends the value of 'analogValue'
```

หมายเหตุ: ข้อมูลเพิ่มเติมของการใช้ ฟังชั่น Serial.println() และ Serial.print() ดูได้จากเวบไซด์ Aruduino

ตัวอย่างต่อไปนี้เป็นคำสั่งอ่านค่าจากขา analog pin 0 และส่งข้อมูลให้กับเครื่องคอมพิวเตอร์ทุกๆ 1 วินาที

```
viod setup()
{
Seral.begin(9600); // sets serial to 9600 bps
}
void loop()
{
Seral.println(analogRead(0)); // sends analog value
delay(1000); // pauses for 1 second
}
```

คำสั่งสำคัญ

คำว่าคำสั่งนั้นแบ่งออกเป็นสองประเภทคือ

1. คำสั่งภายใน
2. คำสั่งภายนอก

คำสั่งภายนอกส่วนใหญ่จะเขียนสำเร็จรูปไว้แล้ว ทั้งใน Complier เองและในการอ้างมาจากที่อื่น(จาก CLass)ซึ่งจะอยู่ในรูปแบบของ Function นั้นเองครับ

ตัวอย่างการอ้างมาจากข้างนอก

```
#include <LiquidCrystal.h>
```

สั่งเกตุว่าเราจะใช้คำสั่ง #include แล้วตามด้วยไฟล์ที่อยู่ใน Folder ชื่อ Libraries นั้นเอง วิธีการเพิ่ม Library ลงใน Arduino หากเราต้องการให้ Arduino มีความสามารถมากขึ้นให้เราก๊อป Lib folder มาจากที่อื่น แล้วให้เอามาวางลงใน Folder นี้แล้วรีสตาร์ท IDE มันก็จะเข้ามาอยู่ในเมนูทั้ง File->Example และ Sketch->Import Library แล้วครับ

Class และ Object

ก่อนอื่นผมจะออกตัวก่อนว่าจะบรรยายเพียงวิธีการนำไปใช้งานเท่านั้นนะครับ แต่คงไม่กล่าวถึงการสร้างคลาสเพราะยาวมากครับ คงต้องเขียนหนังสือเป็นเล่มแน่ๆกว่าจะเข้าใจกัน Class คือพิมพ์เขียว Object ครับ หมายถึง Class ไม่สามารถนำเอาไปใช้ได้ทันที จะต้องนำไปสร้างเป็น Object ก่อนถึงจะสามารถใช้ได้ครับ ยกตัวอย่างของการผลิตรถ คลาสรถก็คือพิมพ์เขียวและงานออกแบบต่างๆซึ่งจะบอกว่ารถคันนั้นมีคุณสมบัติอย่างไร ทำงานอย่างไรคงไม่มีใครเอา

กระดาษพิมพ์เขียวมาขับจริงมั้ยครับ จากนั้นเราก็จะนำพิมพ์เขียวต่างๆเข้าในโรงงาน(Factory) แล้วสร้างตามแบบที่ออกแบบเอาไว้สุดท้ายก็จะออกมาเป็นรถยนต์ตามที่ต้องการ (Object) นั้นเอง

การสร้าง Object ทำได้ดังนี้

[ชื่อ Class] [ชื่อ Object];

หรือกรณีมีตัวแปรเริ่มต้น(Constructor) ให้กำหนดแบบนี้

[ชื่อ Class] [ชื่อ Object](param1,param2...);

ตัวอย่าง

```
BettleCar car1;
```

หรือแบบมี Constructor

```
LiquidCrystal lcd(12, 11, 5, 4, 3, 2);
```

เท่านี้เราก็สามารถที่จะเอา Object ไปใช้งานได้เลยครับ

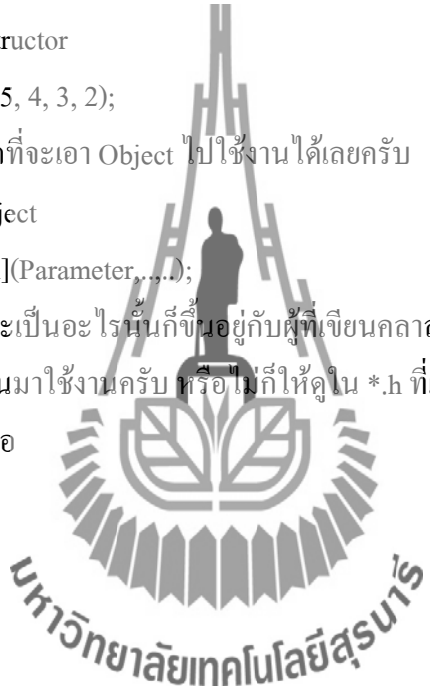
วิธีการใช้งาน Object

[ชื่อ Object].[ชื่อ Function](Parameter,.....);

ซึ่งชื่อ Function จะเป็นอะไรนั้นก็ขึ้นอยู่กับผู้ที่เขียนคลาสครับ เราสามารถดูได้จากเวปต่างๆที่เราไปก๊อปคลาสนั้นมาใช้งานครับ หรือไม่ก็ให้ดูใน *.h ที่เราก๊อปมาก็ได้ครับ

ตัวอย่างการนำไปใช้งานคือ

```
car1.Run(30);
```



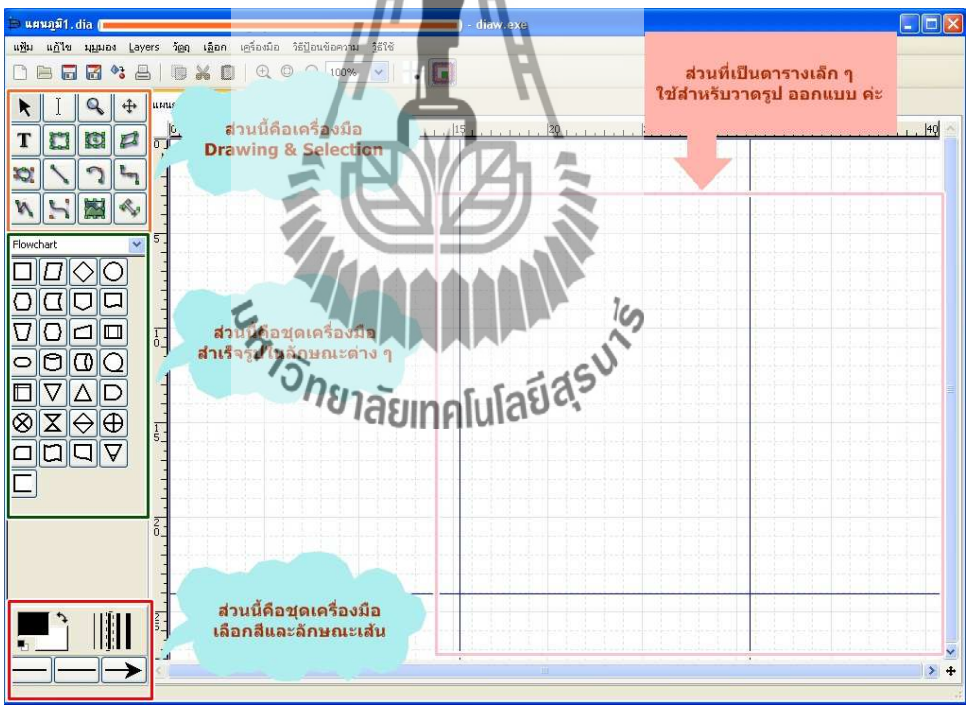
ภาคผนวก ง

คู่มือการใช้งาน Dia - Diagram Drawing

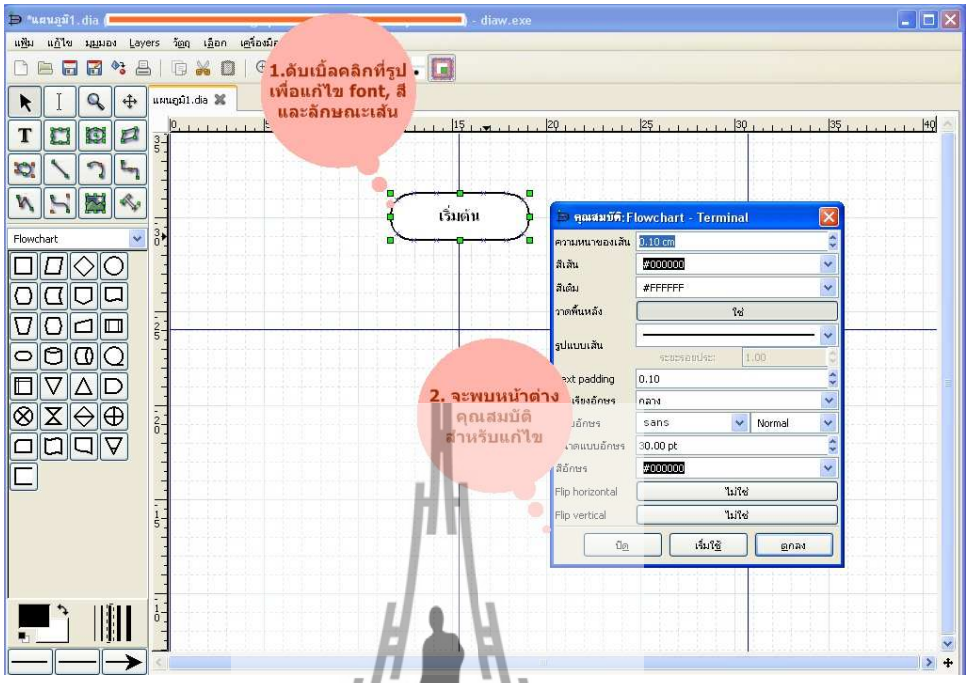
สร้างและออกแบบแผนผังงานง่าย ๆ ด้วย Dia

มาแล้วสำหรับโปรแกรมโอเพนซอร์สดี ๆ ฟรี ๆ ในวันนี้จะแนะนำให้ท่านรู้จักโปรแกรม Dia ซึ่งเป็นโปรแกรมสำหรับเขียนแผนผังงานต่าง ๆ ไม่ว่าจะเป็น Flowchart ของขั้นต้อนการปฏิบัติงาน, แผนที่, วิธีการใช้เครื่องมือต่าง ๆ หรือใช้วาดแผนผังห้อง ก็สามารถทำได้ ซึ่งเท่าที่สังเกตดูพบว่าส่วนมากจะใช้ Word บ้าง Excel บ้าง ในการเขียนแผนผัง ซึ่งดิฉันก็เคยใช้ แต่ใช้แล้วก็รู้สึกว่าทำไมเส้นต่าง ๆ มันไม่ค่อยจะตรงกันเลยนะ เพ่งเท่าไหร่ก็ไม่ตรงสักที ทำให้ค่อนข้างใช้เวลานานในการจัดให้มันตรงกัน ซึ่งโปรแกรม Dia เป็นโปรแกรมในกลุ่ม Opensource ท่านสามารถดาวน์โหลดได้ที่ http://dia-installer.de/download.html

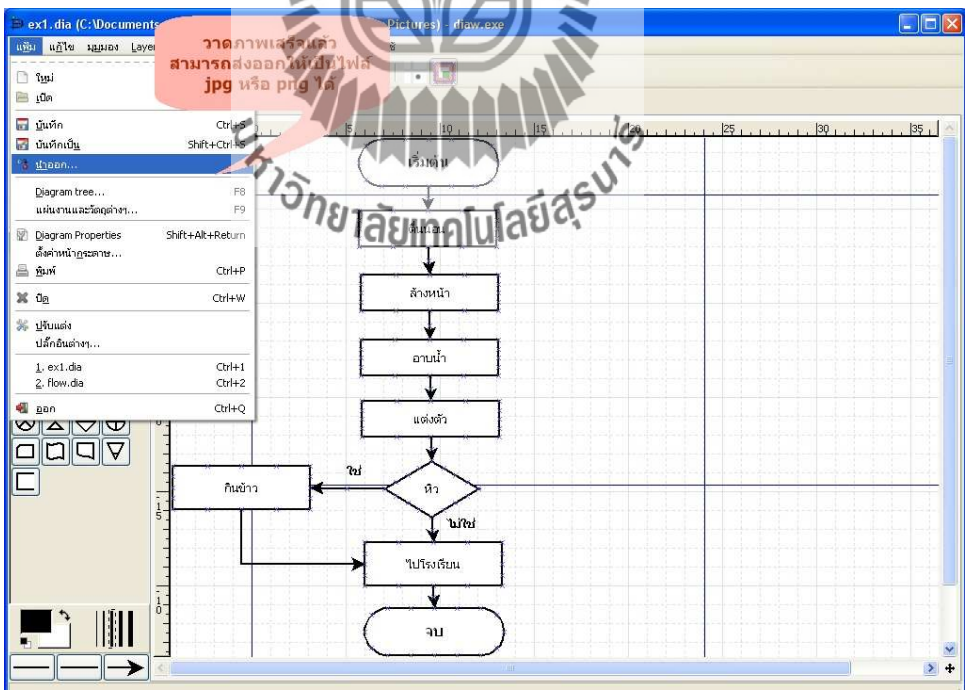
เรามารู้จักหน้าตาของเจ้าตัวโปรแกรมนี้กัน



มาเริ่มวาดภาพกันเลย ขอยกตัวอย่างกิจวัตรประจำวันของเรากันนะ

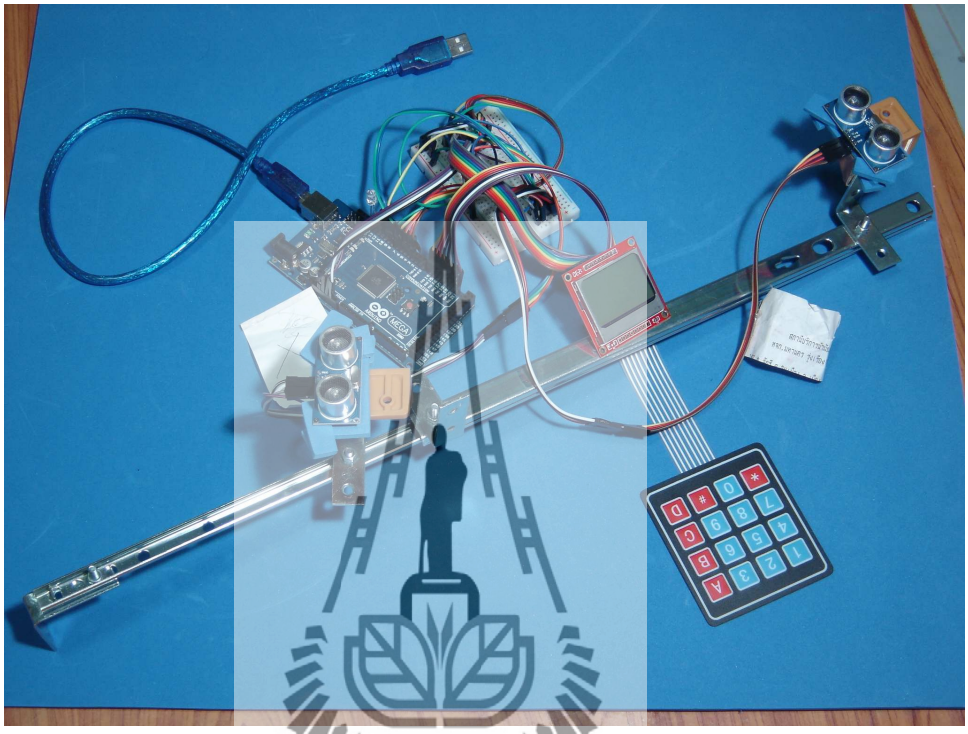


เสร็จแล้ว พร้อมโชว์ สามารถส่งออกเป็น ไฟล์ได้หลายนามสกุลนะ เช่น jpg หรือ png เป็นต้น

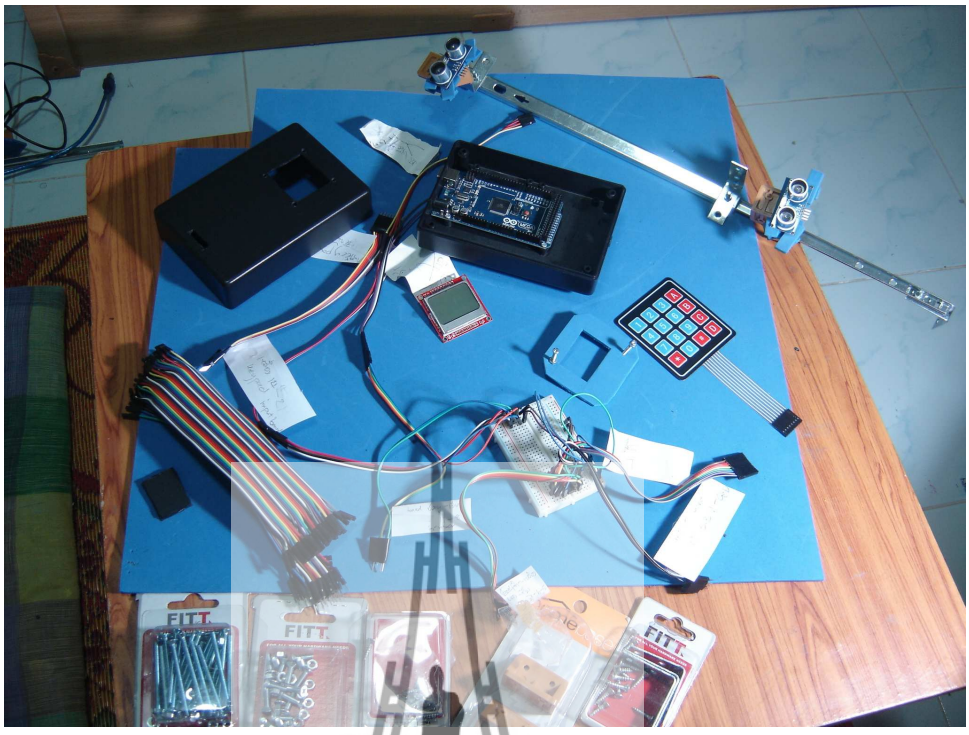


สามารถศึกษาเพิ่มเติม http://202.28.72.148/~picnic/dia/Dia_Manual.pdf (ที่มาเอกสารศูนย์เทคโนโลยีอิเล็กทรอนิกส์และคอมพิวเตอร์แห่งชาติ)

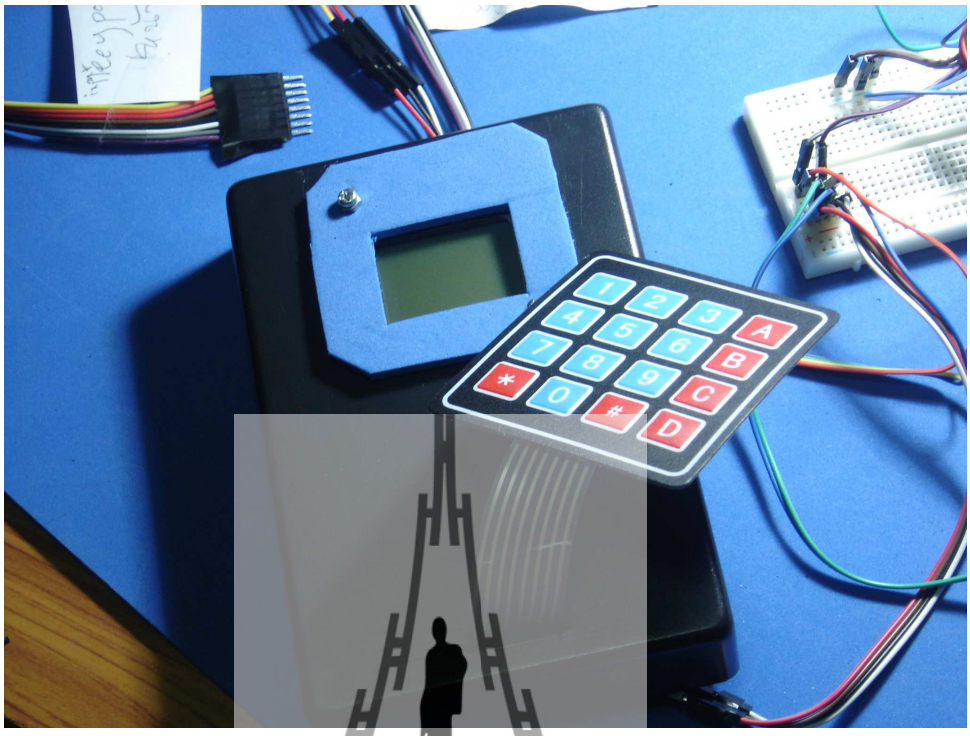
ภาคผนวก จ
รูปการทดลองการประกอบชิ้นสอนอุปกรณ์ และวัดระยะวัตถุ



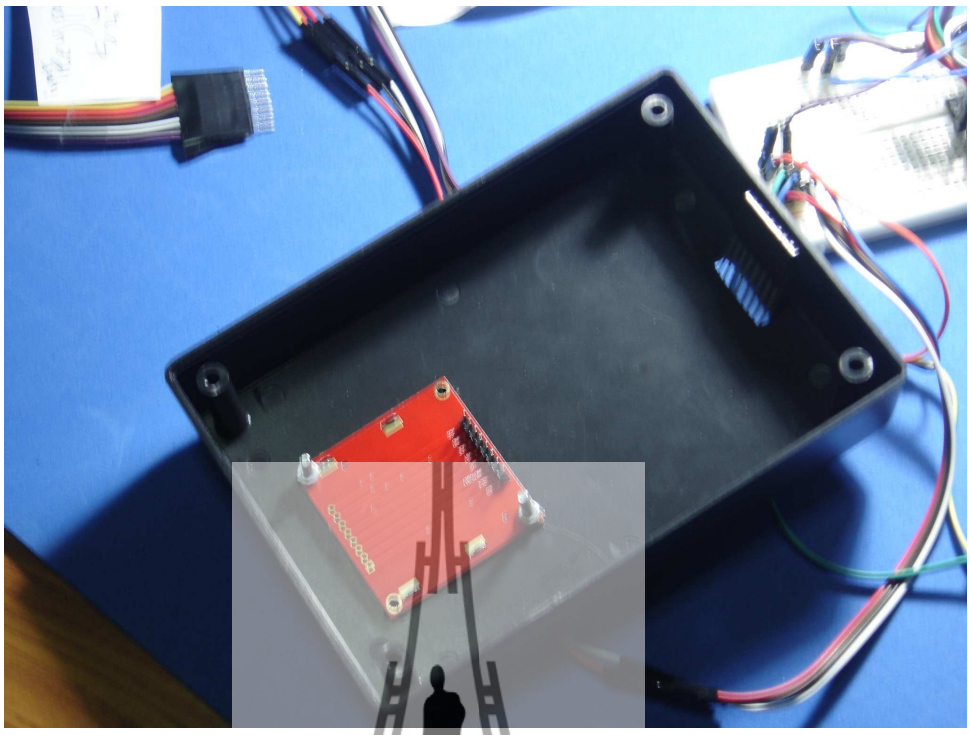
รูป จ.1
อุปกรณ์การทำงานทั้งหมดแบบไม่ได้ประกอบให้ดูสวยงาม



รูป จ.2
อุปกรณ์ และวัสดุที่ใช้ทำโครงงาน แบบแยกชิ้นส่วน

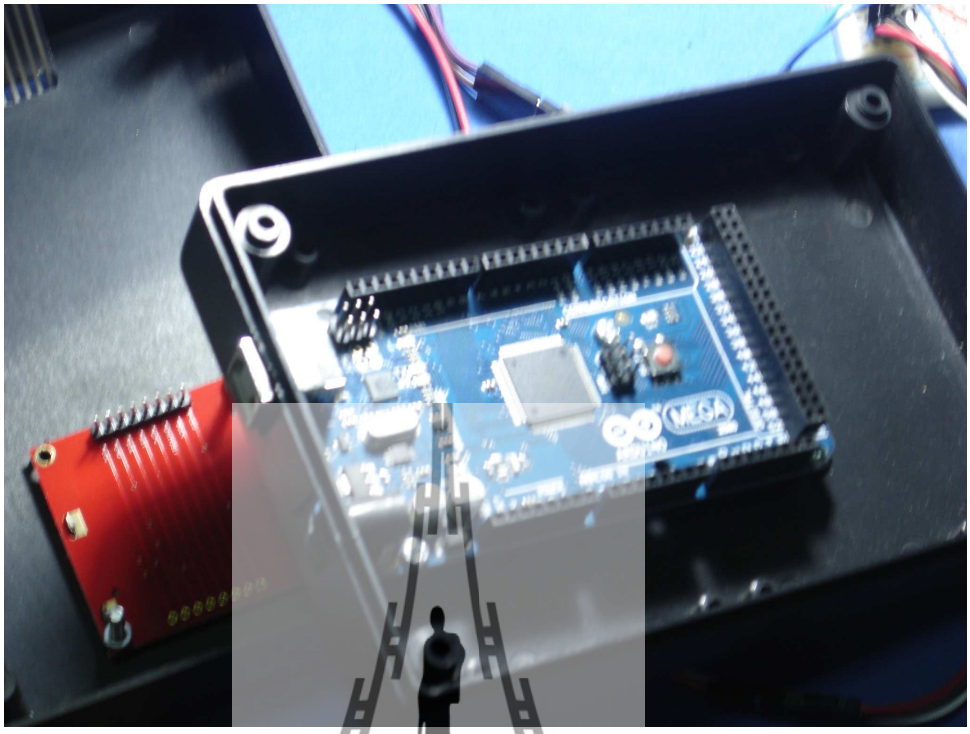


รูป จ.3
ด้านหน้าของ อุปกรณ์เมื่อประกอบ ยังไม่เสร็จ



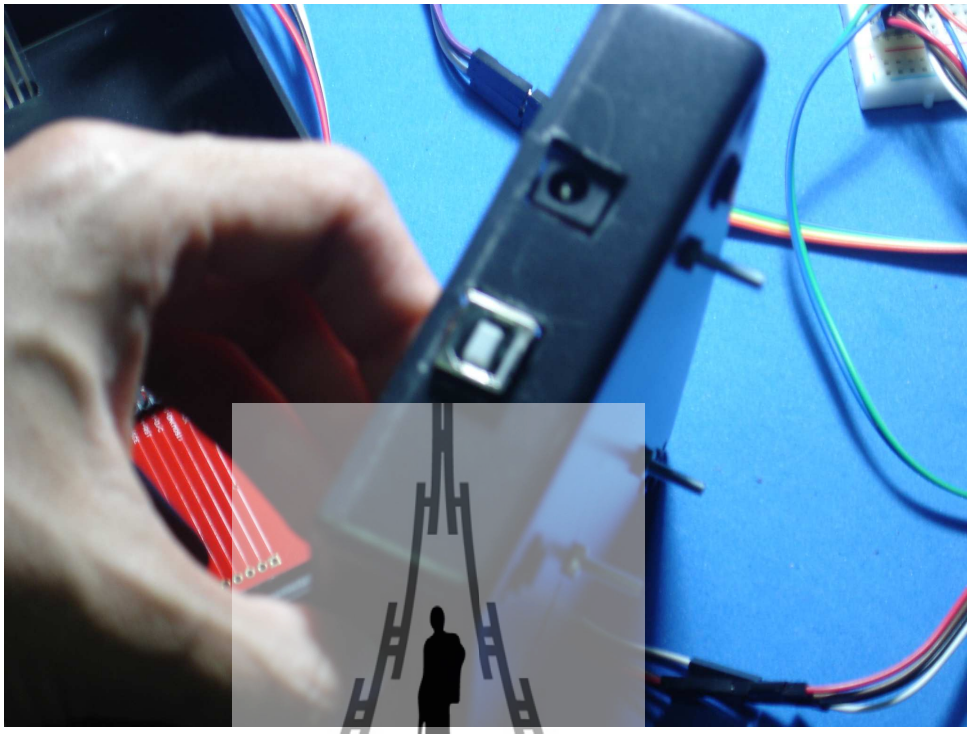
รูป จ.4

ส่วนด้านในของกล่องอเนกประสงค์ เจาะรูและใส่ตัว module เข้าไป



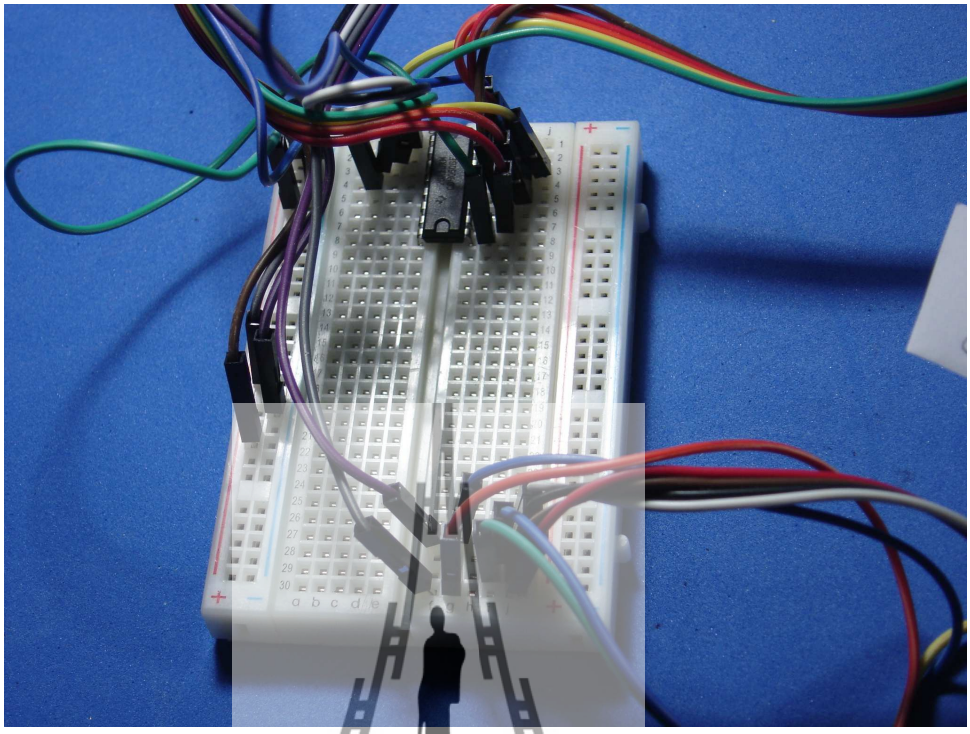
รูป จ.5

ของ board Arduino Mega 2560 ประกอบติดกับตัวกล่อง

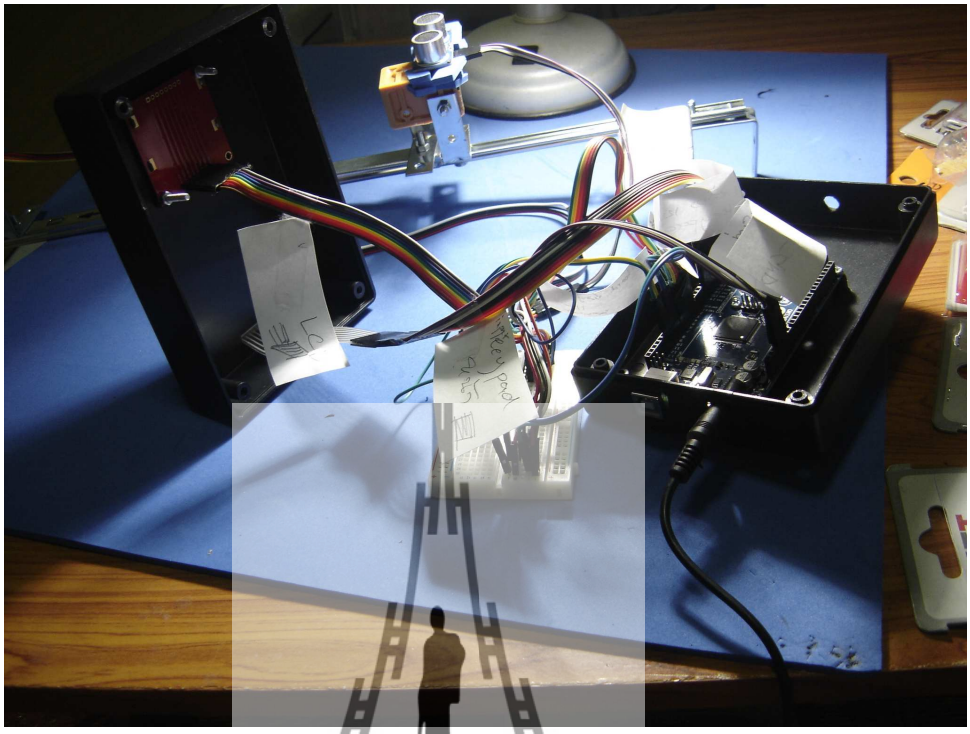


รูป จ.6

Slot ช่องเสียบที่เชื่อม ไฟ VCC (DC) และ Serial Port (ไว้รับส่งข้อมูล)



รูป จ.7
Breadboard 400 holes สายเชื่อม และ IC DC4050BE



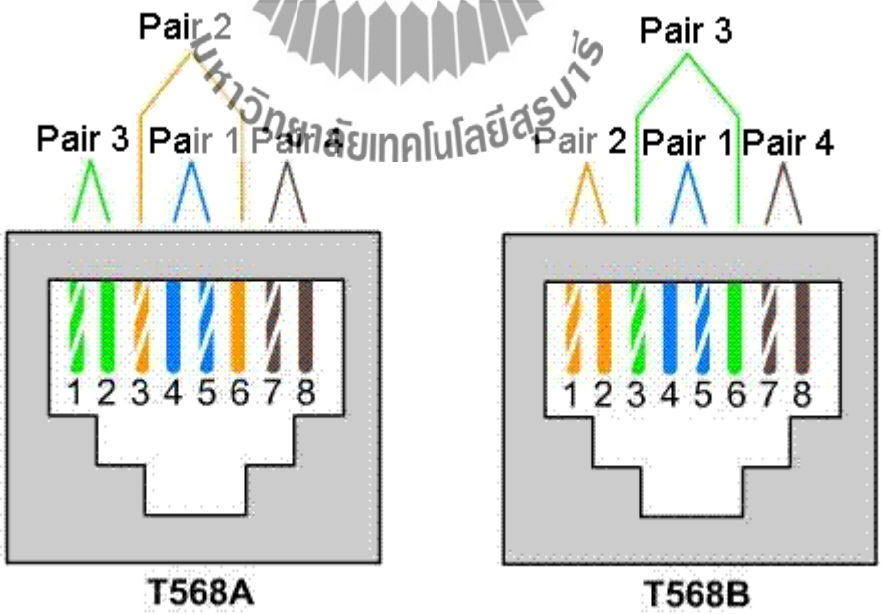
รูป จ.8

เมื่อเชื่อมต่อสาย module IC และ Board Arduino mega 2560 เข้าด้วยกัน

ประกอบในกล่องเอนกประสงค์

รูป จ.9

เมื่อประกอบเสร็จแล้วพร้อมใช้งาน เสียบปลั๊กโดยเพิ่มสาย UTP มาเชื่อมระหว่าง วงจรกับ Module Ultrasonic Sensor HC-SR04 ให้มีความยาวมากขึ้นง่ายต่อการวัดค่า



รูป จ.10

จำนวนสาย 8 เส้น และสีของสาย UTP

รูป จ.11

ทำการทดลองหาค่าความเหมาะสมของ การติดตั้ง อุปกรณ์

ประวัติผู้เขียน

นายธนทรัพย์ ตนะทิพย์ เกิดเมื่อวันที่ 8 กรกฎาคม พ.ศ. 2530 ภูมิลำเนาอยู่ที่ ตำบลจอมพระ อำเภอท่าวังผา จังหวัดน่าน สำเร็จการศึกษาระดับมัธยมปลายจากโรงเรียนท่าวังผาพิทยาคม อำเภอท่าวังผา จังหวัดน่าน เมื่อปี พ.ศ. 2548 ปัจจุบันเป็นนักศึกษาชั้นปีที่ 6 สาขาวิชาวิศวกรรมโทรคมนาคม สำนักวิชาวิศวกรรมศาสตร์ มหาวิทยาลัยเทคโนโลยีสุรนารี