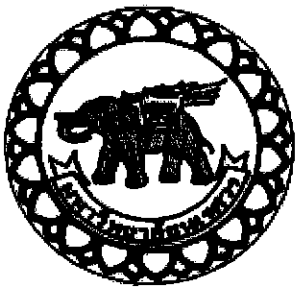


เครื่องวัดระยะทางโดยใช้อุลตร้าโซนิค

DISTANCE METER USING ULTRASONIC SENSOR

นายวัชรพงค์ เชื้อสะอาด รหัส 48280123

I 5078135 ค.2
...สมุดคณะวิศวกรรมศาสตร์
5200034
93794
2551.

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต

สาขาวิชาวิศวกรรมไฟฟ้า ภาควิชาวิศวกรรมไฟฟ้าและคอมพิวเตอร์

คณะวิศวกรรมศาสตร์ มหาวิทยาลัยนเรศวร

ใบรับรองโครงงานวิศวกรรม

หัวข้อโครงงาน	เครื่องวัดระยะทางโดยใช้อุลตร้าโซนิค
ผู้ดำเนินโครงงาน	นายวัชรพงค์ เชื้อสะอาด รหัส 48280123
อาจารย์ที่ปรึกษา	ดร.อัครพันธ์ วงศ์กังแห
อาจารย์ที่ปรึกษาร่วม	ดร.ชูศักดิ์ ธนวัฒโน
สาขาวิชา	วิศวกรรมไฟฟ้า
ภาควิชา	วิศวกรรมไฟฟ้าและคอมพิวเตอร์
ปีการศึกษา	2551

.....

คณะวิศวกรรมศาสตร์ มหาวิทยาลัยนเรศวร อนุมัติให้โครงงานฉบับนี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรวิศวกรรมศาสตรบัณฑิต สาขาวิชาวิศวกรรมไฟฟ้า

คณะกรรมการการสอบโครงงานวิศวกรรม

..... ประธานกรรมการ
(ดร.อัครพันธ์ วงศ์กังแห)

..... กรรมการ
(ดร.ชูศักดิ์ ธนวัฒโน)

..... กรรมการ
(ดร.ชัยรัตน์ พินทอง)

หัวข้อโครงงาน	เครื่องวัดระยะทางโดยใช้อุลตร้าโซนิค
ผู้ดำเนินโครงงาน	นายวัชรพงค์ เชื้อสะอาด รหัส 48280123
อาจารย์ที่ปรึกษา	ดร.อัครพันธ์ วงศ์กังแห
อาจารย์ที่ปรึกษาร่วม	ดร.ชูศักดิ์ ธนวัฒโน
สาขาวิชา	วิศวกรรมไฟฟ้า
ภาควิชา	วิศวกรรมไฟฟ้าและคอมพิวเตอร์
ปีการศึกษา	2551

บทคัดย่อ

โครงงานเป็นการศึกษาและพัฒนาเครื่องวัดระยะทางโดยใช้อุลตร้าโซนิค ซึ่งมีความรวดเร็วและแม่นยำในการวัดระยะโดยเลือกใช้อุลตร้าโซนิค Ultra-Sonic Ranger (SRF05) ซึ่งใช้ความถี่ 40 kHz ในการส่งคลื่นออกไปกระทบวัตถุแล้วอาศัยการสะท้อนกลับของคลื่นตามระยะเวลา แล้วทำการเปรียบเทียบเป็นระยะทางออกมา ด้วยการเขียนโปรแกรมบน AVR Studio4 แบบใช้Timer/Counter ช่วยในการสร้างสัญญาณพลัส แล้วผ่านการเทียบสัญญาณที่ส่งจาก Echo pulse และ Trigger pulse และใช้ไมโครคอนโทรลเลอร์ควบคุมการทำงานของ SRF05 โดยใช้ไมโครคอนโทรลเลอร์ตระกูล AVR (ATmega48) แล้วทำการแสดงผลออกทาง LCD

Project title	Distance meter using ultrasonic sensor
Name	Mr.Watcharapong Chuesa-ard ID. 48280123
Project Advisor	Dr.Akaraphunt Vongkunghae
Co-Project Advisor	Dr. Chusak Thanawattano
Major	Electrical Engineering
Department	Electrical and Computer Engineering
Academic Year	2008

Abstract

The project presents a design and an implementation of a distance measurement system using ultrasonic sensors. The Ultra-Sonic Ranger (SRF05) module and AVR Atmega48 are used in this project. The SRF05 module controls the ultrasonic speaker and monitors the reflected wave. The AVR Atmega48 takes control of the displaying and triggering signals for measurement. The range of this ultrasonic range meter is from 1 to 4000 centimeters.

กิตติกรรมประกาศ

ในการจัดทำโครงงานเรื่อง เครื่องวัดระยะทางโดยใช้อุลตร้าโซนิค ทางคณะผู้จัดทำใคร่ขอแสดงความขอบคุณ ดร.อัครพันธ์ วงศ์กังแห ให้เกียรติเป็นที่ปรึกษา ให้ความรู้และแนวทางในการทำโครงงาน และขอขอบคุณ ดร.ชูศักดิ์ ธนวัฒโน ที่ให้เกียรติเป็นที่ปรึกษาร่วมโครงงานวิศวกรรม ซึ่งได้แนะแนวทางแก้ไขปัญหาเกี่ยวกับโครงงาน และข้อเสนอแนะต่างๆที่เป็นประโยชน์ต่อการทำงานครั้งนี้ให้ลุล่วงไปได้ และแหล่งทุนศูนย์เทคโนโลยีอิเล็กทรอนิกส์และคอมพิวเตอร์แห่งชาติที่ให้ทุนสนับสนุน โดย ดร.ชูศักดิ์ ธนวัฒโน และขอบคุณอาจารย์ทุกท่านในภาควิชาวิศวกรรมไฟฟ้าและคอมพิวเตอร์ มหาวิทยาลัยนเรศวร ที่ได้สั่งสอนและให้ความรู้ จนสามารถนำมาประยุกต์ใช้ในการทำโครงงานครั้งนี้

นายวัชรพงค์ เชื้อสะอาด

สารบัญ

หน้า

บทคัดย่อภาษาไทย ..... ก
บทคัดย่อภาษาอังกฤษ ..... ข
กิตติกรรมประกาศ ..... ค
สารบัญ ..... ง
สารบัญตาราง ..... ฉ
สารบัญรูป ..... ช

บทที่ 1 บทนำ

1.1 ที่มาและความสำคัญของโครงงาน 1
1.2 วัตถุประสงค์ของโครงงงาน 1
1.3 ขอบข่ายของโครงงาน 1
1.4 ขั้นตอนการดำเนินงาน 1
1.5 แผนการดำเนินงาน 2
1.6 ผลที่คาดว่าจะได้รับ 2
1.7 งบประมาณของโครงงาน 3

บทที่ 2 หลักการและอุปกรณ์ที่เกี่ยวข้อง

2.1 ระบบอุลตร้าโซนิค (ULTRASONIC) 4
2.2 อุลตร้าโซนิคเซนเซอร์และการทำงาน 9
2.3 อุลตร้าโซนิคแบบ SRF05 16

สารบัญตาราง

ตารางที่	หน้า
3.1 การทำงานของขาพอร์ต LCD	23
4.1 อ่านค่าวัดระยะทางที่วัดโดยตลับเมตรเทียบกับการวัดระยะทางจากอุลตร้าโซนิค และค่าเปอร์เซนต์ความแตกต่าง	36
ก Clock Select Bit Description ภาคผนวก ข	42

สารบัญรูป

รูปที่	หน้า
(ก) โครงสร้างภายในตัวอุลตร้าโซนิคทรานสดิวเซอร์แบบเปียโซอิเลคทริค ที่ใช้สารเซรามิค	5
(ข) เมื่อป้อนแรงดันให้แก่ตัวมันจะทำให้ชิ้นสารเซรามิคโก่งงอไปมาทำให้เกิดคลื่นเสียง อุลตร้าโซนิคกระจายไปในอากาศ	5
2.1 แสดงตัวอย่างการเขียนสัญลักษณ์ของอุลตร้าโซนิคทรานสดิวเซอร์แบบต่างๆกัน	7
2.2 แสดงผลการทดลองตัวรับตัวหนึ่งโดยลองเปลี่ยนโหลดเป็นค่าต่างๆกัน แล้วป้อนคลื่นเสียงความถี่ต่างๆกันเข้ามา	8
2.3 หลักการทำงานของอุลตร้าโซนิค	9
2.4 อุลตร้าโซนิคเซ็นเซอร์ วงจรส่งผ่านและรับ	10
2.5 อุลตร้าโซนิคเซ็นเซอร์, วงจรเวลาคงที่	11
2.6 อุลตร้าโซนิคเซ็นเซอร์, วงจรที่เปลี่ยนแปลงได้	12
2.7 อุลตร้าโซนิคเซ็นเซอร์แบบชิ้นส่วนการอิมพัลซ์ (Impulse) แตกออกที่170 MHz	12
2.8 อุลตร้าโซนิคเซ็นเซอร์, คุณลักษณะการตรวจจับ	13
2.9 อุลตร้าโซนิคเซ็นเซอร์, พื้นผิวตรง	14
2.10 อุลตร้าโซนิคเซ็นเซอร์, การตรวจจับสิ่งของ	14
2.11 อุลตร้าโซนิคเซ็นเซอร์, การเบี่ยงเบนคลื่นเสียง	14
2.12 อุลตร้าโซนิคเซ็นเซอร์, ความเร็วของวัตถุที่ยอมให้ได้	15
2.13 แสดงหลักการตรวจจับวัตถุโดยใช้สัญญาณความถี่เหนือเสียงหรืออุตร้าโซนิค	16
2.14 การแสดงขาสัญญาณของ SRF05 และโหมดการทำงานต่างๆ	17
2.15 แสดงบอร์ด ADX-SRF05 และสาย PCB3A	18
2.16 แสดงการใช้งานของบอร์ด ADX-SRF05	19
3.1 รูปวงจรของบอร์ดทดลองที่ออกแบบเอง	21
3.2 วงจรไมโครคอนโทรลเลอร์ที่เชื่อมกับ SRF05	23
3.3 วงจรไมโครคอนโทรลเลอร์ที่เชื่อมต่อกับ LCD	24

สารบัญรูป (ต่อ)

รูปที่		หน้า
3.4	การต่อวงจรเพื่อใช้งานจริง	24
3.5	ไดอะแกรมเวลาแสดงการส่งสัญญาณไปและการสะท้อนสัญญาณกลับของ SRF05 ในกรณีติดต่อแบบ 2 สัญญาณ	25
3.6	โปรแกรม AVR Studio	27
3.7	เมนู Project	27
3.8	เมนู Project	27
3.9	AVR GCC พัฒนาโปรแกรมด้วยภาษา C	28
3.10	กำหนดเครื่องมือในการดีบักและเบอร์ AVR ที่ใช้งาน	28
3.11	หน้าต่างเขียนโค้ดโปรแกรมและหน้าต่างคอมไพล์โปรแกรม	29
3.12	หน้าต่างเขียนโค้ดโปรแกรม	29
3.13	หน้าต่างคอมไพล์โปรแกรม	30
3.14	Configuration Option	30
3.15	การกำหนดค่าความถี่ (Frequency)	30
3.16	หน้าต่าง Project Option	31
3.17	แสดงไฟล์ที่ได้จากการคอมไพล์ .HEX	32
3.18	เมนู Program AVR	32
3.19	การเลือก Platform และ Port	32
3.20	หัวข้อ Program และส่วนใช้งาน	33
3.21	จากไดเรกทอรีที่ได้เก็บไฟล์ .HEX	33
4.1	อ่านค่าวัดระยะทางที่วัดโดยตลับเมตรเทียบกับอุลตร้าโซนิค	35
4.2	อ่านค่าวัดระยะทางที่วัดโดยไม้บรรทัดเทียบกับอุลตร้าโซนิค	35
4.3	ระยะทางที่ได้จากการวัดตลับเมตรเทียบกับระยะทางการวัดที่ได้จากอุบตร้าโซนิค	37

บทที่ 1
บทนำ

1.1 ที่มาและความสำคัญ

การวัดระยะทาง โดยทั่วไปจะใช้เครื่องวัดแบบตลับเมตร หรือสายเมตรในการวัดระยะทางแบบปกติจะใช้ตลับเมตรดึงจากจุดที่ตั้งเป็นหลักแล้วดึงซึ่งการวัดโดยวิธีเหล่านี้จะมีข้อด้อยอยู่ คือ จะต้องใช้เวลาในการวัดและกินพื้นในการวัด ต่อการวัด 1 ครั้ง ซึ่งหากเป็นทางหรือช่องแคบที่ไม่สามารถวัดระยะทางได้จึงจะต้องมีวิธีการวัดระยะทางที่รวดเร็วและแม่นยำ

1.2 วัตถุประสงค์ของโครงงาน

1.2.1 ค้นคว้าเกี่ยวกับการวัดระยะทางต่างๆ ที่มีอยู่ในปัจจุบัน

1.2.1 ศึกษาทำความเข้าใจหาสาเหตุเกี่ยวกับคลื่นความถี่ ที่มนุษย์สามารถนำไปใช้งานได้จากความถี่เหนือเสียงที่มนุษย์สามารถได้ยิน

1.2.3 ออกแบบและสร้างเครื่องวัดระยะทางโดยใช้อุลตร้าโซนิค

1.3 ขอบข่ายของโครงงาน

1.3.1 ออกแบบและสร้างเครื่องวัดระยะทางโดยใช้อุลตร้าโซนิค

1.3.2 เครื่องวัดระยะทางด้วยอุลตร้าโซนิคที่พัฒนาขึ้น สามารถแสดงค่าออกมาได้

1.3.3 เครื่องวัดระยะทางด้วยอุลตร้าโซนิค แสดงค่าที่ได้ออกมาให้มีความผิดพลาดน้อยที่สุด แสดงผลออกทาง LCD Module

1.4 ขั้นตอนการดำเนินงาน

1.4.1 ศึกษาหลักการวัดระยะทางแบบใช้คลื่นความถี่

1.4.2 ค้นคว้าเกี่ยวกับการเพรีคลื่นความถี่ของคลื่นอุลตร้าโซนิค

1.4.3 ศึกษาและคัดเลือกอุปกรณ์ที่จะนำมาสร้างเครื่องวัดระยะทาง

1.4.4 ออกแบบเครื่องวัดระยะทางและวงจรอิเล็กทรอนิกส์ที่เกี่ยวข้อง

1.4.5 เขียนโปรแกรมและทำการดีบัคโปรแกรมควบคุมการทำงานของเครื่องวัดระยะทาง

1.4.6 ทดสอบการทำงานและแก้ไขข้อผิดพลาดและทำ comment ไว้

1.4.7 สรุปผลการทดลอง ข้อเสนอแนะ และจัดทำรูปเล่มโครงงาน

1.5 แผนการดำเนินงาน

กิจกรรม	2551							2552	
	มิ.ย.	ก.ค.	ส.ค.	ก.ย.	ต.ค.	พ.ย.	ธ.ค.	ม.ค.	ก.พ.
1. ศึกษาหลักการวัดระยะแบบต่างๆ	■								
2. ค้นคว้าเกี่ยวกับส่งคลื่นความถี่จากวัตถุ		■							
3. ศึกษาและคัดเลือกอุปกรณ์ที่จะนำมาสร้างวัดระยะทาง			■						
4. ออกแบบเครื่องวัดระยะและวงจรอิเล็กทรอนิกส์ที่เกี่ยวข้อง				■	■	■			
5. เขียนโปรแกรม และทำการ ดีบัค โปรแกรม ควบคุมการทำงานของเครื่องวัดระยะ					■	■	■		
6. ทดสอบการทำงานและแก้ไขข้อผิดพลาดและทำ comment ไว้						■	■	■	
7. สรุปผลการทดลอง ข้อเสนอแนะ และจัดทำรูปเล่มโครงงาน							■	■	■

1.6 ผลที่คาดว่าจะได้รับ

1.6.1 เครื่องวัดระยะทางด้วยอุลตร้าโซนิค สามารถวัดระยะทางได้รวดเร็วแม่นยำถูกต้อง และมีฟังก์ชันการทำงานอื่นๆ ที่เกี่ยวข้องครบถ้วน

1.6.2 มีความรู้ความเข้าใจในการเขียนโปรแกรมควบคุมไมโครคอนโทรลเลอร์ และการออกแบบวงจรอิเล็กทรอนิกส์

1.7 งบประมาณของโครงงาน

1.7.1 ค่าอุปกรณ์พัฒนาเครื่องวัดระยะทางด้วยอุลตร้าโซนิค	4500 บาท
1.7.2 ค่าจัดทำรูปเล่มโครงงาน	500 บาท
รวมเป็นเงิน	5000 บาท

บทที่ 2
หลักการและอุปกรณ์ที่เกี่ยวข้อง

2.1 ระบบอุลตร้าโซนิค (ULTRASONIC)

หมายถึง คลื่นเสียงที่มีความถี่สูงเกินกว่าที่หูมนุษย์จะได้ยิน โดยทั่วไปแล้วหูของมนุษย์โดยเฉลี่ยจะได้ยินเสียงสูงถึงเพียงแค่ประมาณ 15 KHz เท่านั้น แต่พวกที่อายุยังน้อย ๆ อาจจะได้ยินเสียงที่มีความถี่สูงกว่านี้ได้ ดังนั้นโดยปกติแล้วคำว่าอุลตร้าโซนิคจึงมักจะหมายถึงคลื่นเสียงที่มีความถี่สูงกว่า 20 KHz ขึ้นไป จะสูงขึ้นจนถึงเท่าใดไม่ได้ระบุจำกัดเอาไว้ สาเหตุที่มีการนำเอาคลื่นย่านอุลตร้าโซนิคมาใช้ก็เพราะว่า เป็นคลื่นทีมีทิศทางทำให้เราสามารถเล็งคลื่นเสียงไปยังเป้าหมายที่ต้องการได้โดยเจาะจง เรื่องนี้เป็นคุณสมบัติของคลื่นอย่างหนึ่ง ยิ่งคลื่นมีความถี่สูงขึ้นความยาวคลื่นก็จะยิ่งสั้นลง ถ้าความยาวคลื่นยาวกว่าช่องเปิด (ที่ให้เสียงนั้นออกมา) ของตัวกำเนิดเสียงความถี่นั้นเช่น คลื่นความถี่ 300 Hz ในอากาศจะมีความยาวถึงประมาณ 1 เมตรเศษ ๆ ซึ่งจะยาวกว่าช่องที่ให้คลื่นเสียงออกมาจากตัวกำเนิดเสียงโดยทั่วไปมากมาย คลื่นจะหักเบนที่ขอบด้านนอกของตัวกำเนิดเสียง ทำให้เกิดการกระจายทิศทางคลื่นแต่ถ้าความถี่สูงขึ้นมาอยู่ในย่านอุลตร้าโซนิคอย่างเช่น 40 KHz จะมีความยาวคลื่นในอากาศเพียงประมาณ 8 มม. เท่านั้นซึ่งเล็กกว่ารูเปิดของตัวที่ให้กำเนิดเสียงความถี่นี้มากคลื่นเสียงจะไม่มีการเลี้ยวเบนที่ขอบจึงพุ่งออกมาเป็นลำแคบ ๆ หรือที่เราเรียกว่า มีทิศทาง

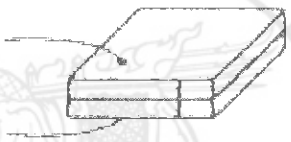
การมีทิศทางของคลื่นเสียงย่านอุลตร้าโซนิคทำให้เรานำไปใช้งานได้หลายอย่าง เช่น นำไปใช้ในเครื่องควบคุมระยะไกล (Ultrasonic remote control) เครื่องล้างอุปกรณ์ (Ultrasonic cleaner) โดยให้น้ำสั่นที่ความถี่สูง เครื่องวัดความหนาของวัตถุโดยสังเกตระยะเวลาที่คลื่นสะท้อนกลับมา เครื่องวัดความลึกและทำแผนที่ใต้ท้องทะเล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

แบบเปียโซอิเลคทริก (Piezo-electric Transducer) ซึ่งแปลงไปมาระหว่างพลังงานไฟฟ้าและพลังงานทางกล โดยมีความถี่เรโซแนนซ์คงที่อยู่ค่าหนึ่ง

แบบแมกนีโตสตริคทีฟ (Magnetostrictive Transducer) ซึ่งแปลงไปมาระหว่างพลังงานไฟฟ้าในขดลวดกับตำแหน่งความยาวของแกนเหล็กที่สวมขดลวดนั้นอยู่

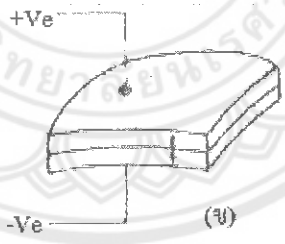
แบบอิเลคโตรสตริคทีฟ (Electrostrictive Transducer) ซึ่งแปลงไปมาระหว่างพลังงานไฟฟ้ากับพลังงานทางกล

ทรานสดิวเซอร์แบบเปียโซอิเลคทริค ภายในตัวอุลตร้าโซนิคทรานสดิวเซอร์แบบเปียโซอิเลคทริค แบบที่มีใช้กันในปัจจุบันซึ่งได้รับการพัฒนากันมาในระดับหนึ่งแล้วจะประกอบด้วยชิ้นสารเซรามิคสี่เหลี่ยมซึ่งมีผิวโลหะเงินฉาบอยู่ทั้ง 2 หน้าเพื่อให้ต่อสายไฟออกมาเป็นขา 2 ขา ชิ้นสารเซรามิคนี้ประกอบขึ้นจากสารเซรามิค 2 ชิ้น ประกบกันอยู่โดยวางให้ขั้วไดโพลทางไฟฟ้าภายในอะตอมของมันมีทิศทางตรงข้ามกันดังรูป



(ก)

รูป (ก) โครงสร้างภายในตัวอุลตร้าโซนิคทรานสดิวเซอร์แบบเปียโซอิเลคทริคที่ใช้สารเซรามิค



(ข)

รูป (ข) เมื่อป้อนแรงดันให้แก่ตัวมันจะทำให้ชิ้นสารเซรามิคโก่งงอไปมาทำให้เกิดคลื่นเสียงอุลตร้าโซนิคกระจายไปในอากาศ

ชิ้นสารเซรามิคถูกยึดติดภายในตัวถังอย่างดีเพื่อไม่ให้การสั่นขณะที่มันทำงานอยู่ ได้รับผลกระทบกระเทือนจากภายนอกตัวถังมักจะเป็นรูปทรงกระบอกที่มีเส้นผ่าศูนย์กลางและมีความสูงประมาณ 1 ถึง 2.5 ซม. ด้านหน้าทำเป็นช่องเปิดมีตะแกรงติดอยู่เพื่อให้คลื่นอุลตร้าโซนิคเข้ามาหรือออกจากช่องเปิดได้โดยสะดวก ถ้าตัวถังทำมาจากโลหะก็ควรต่อตัวถังลงกราวด์เพื่อทำหน้าที่ชีลด์ สำหรับบางยี่ห้อเขาจะต่อขาหนึ่งติดกับตัวถังมาให้เลย เมื่อพลิกดูขา 2 ขาที่โผล่ออกมาจากตัวถังจะเห็นมีขาหนึ่งติดกับตัวถัง

เมื่อมีสัญญาณแรงดันมาตกคร่อมขั้วทั้งสองของชิ้นสารเซรามิคดังรูป (ข) จะทำให้ชิ้นสารโก่งงอมากหรือน้อยหรือในทิศทางใดตามขนาดและทิศทางการเปลี่ยนแปลงขนาดของสัญญาณนั้น ๆ ทำให้เกิดการกดอัดอากาศโดยรอบเกิดเป็นคลื่นเสียงที่มีความถี่เดียวกับสัญญาณนั้นออกไป โดยทั่ว ๆ ไปกำลังเอาท์พุทที่ออกมาจะตกประมาณ 10% ของกำลังไฟฟ้าที่ป้อนเข้าไป แต่กำลังเอาท์พุทจะสูงสุดที่ค่าประมาณนี้ต่อเมื่อความถี่ของสัญญาณตรงกับความถี่เรโซแนนซ์ ซึ่งเป็นความถี่ทางกลตามธรรมชาติของชิ้นสารเซรามิคนั้น ๆ ส่วนที่ความถี่อื่น ๆ กำลังเอาท์พุทจะลดลงกว่านี้มาก

ในทำนองกลับกันเมื่อมีคลื่นเสียงที่มีความถี่ตรงกับความถี่เรโซแนนซ์ของชิ้นสารเซรามิคเข้ามาจะทำให้ชิ้นสารโก่งงอไปมาและเกิดสัญญาณแรงดันซึ่งมีขนาดเล็กขึ้นมาคร่อมขั้วทั้งสองของตัวมันเองได้ คุณสมบัติโดยทั่วไปของอุลตร้าโซนิคทรานสดิวเซอร์แบบเปียโซอิเลคตริคก็คือมีค่าความต้านทานไฟตรงสูงมากอาจสูงถึง 100 MΩ เรียกว่าถ้าเอามัลติมิเตอร์ธรรมดามาตั้งสเกลวัดค่าความต้านทานสูง ๆ เข็มจะไม่กระดิกเลย แต่ในขณะที่มันทำงานความต้านทานทางด้านไฟสลับจะลดลง

2.1.1 ตัวส่งและตัวรับ

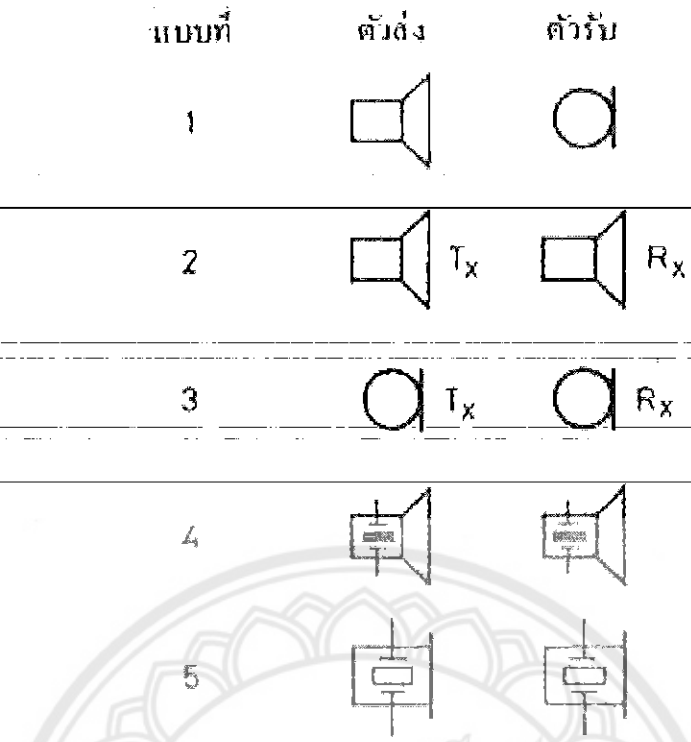
ทรานสดิวเซอร์แบบเปียโซอิเลคทริคที่ใช้สารเซรามิค (หรือที่ผู้ผลิตบางรายเรียกว่าอุลตร้าโซนิคทรานสดิวเซอร์แบบเซรามิค) จะมีอยู่ 2 อย่าง คือ ตัวส่งหรือ Transmitter และ ตัวรับ (เสียง) หรือ Receiver

2.1.1.1 ตัวส่ง (Transmitter)

ก็คืออุลตร้าโซนิคทรานสดิวเซอร์ที่ถูกออกแบบเจาะจงมาให้แปลงสัญญาณไฟฟ้า ที่ให้แก่ตัวมัน ให้ออกมาเป็นคลื่นเสียงย่านอุลตร้าโซนิค หน้าที่ของตัวส่งจึงคล้าย ๆ กับเป็นลำโพง

2.1.1.2 ตัวรับ (Receiver)

ก็คืออุลตร้าโซนิคทรานสดิวเซอร์ ที่ถูกออกแบบเจาะจงมาให้แปลงคลื่นเสียงย่านอุลตร้าโซนิคที่มาตกกระทบตัวมันให้ออกมาเป็นสัญญาณไฟฟ้า หน้าที่ของตัวรับจึงคล้าย ๆ กับเป็นไมโครโฟน ด้วยเหตุนี้เวลาเขียนสัญลักษณ์ของอุลตร้าโซนิคทรานสดิวเซอร์จึงนิยมเขียนตามหน้าที่ของมันคือถ้าเป็นตัวส่งก็เขียนสัญลักษณ์เป็นลำโพง ถ้าเป็นตัวรับก็เขียนสัญลักษณ์เป็นไมโครโฟน ดังรูป 2.1



รูปที่ 2.1 แสดงตัวอย่างการเขียนสัญลักษณ์ของอุลตร้าโซนิคทรานสดิวเซอร์แบบต่างๆกัน

อุลตร้าโซนิคทรานสดิวเซอร์แบบเซรามิคที่มีจำหน่ายกันจะมีค่าความถี่เรโซแนนซ์ให้เลือกตั้งแต่ 23 KHz ขึ้นไปจนถึง 40 KHz แต่ที่พบเห็นกันบ่อยก็มี 23 KHz, 25 KHz, และ 40 KHz โดยความถี่ 40 KHz เป็นรุ่นที่นิยมใช้กันมากที่สุดเพราะมีทิศทางดีกว่า

2.1.2 ข้อควรรู้ในการใช้งานตัวส่งและตัวรับ

เนื่องจากสเปคตลอดจนรายละเอียดต่างๆของอุลตร้าโซนิคทรานสดิวเซอร์หาได้ยาก ดังนั้นจึงสามารถที่จะสรุปสิ่งที่ควรรู้ในขั้นต้นของอุปกรณ์อุลตร้าโซนิคทรานสดิวเซอร์ เพื่อเป็นแนวทางในการใช้งานดังนี้

- ไม่ควรให้ตัวทรานสดิวเซอร์ได้รับการกระแทกหรือตกจากที่สูง เพื่อป้องกันโครงสร้างภายในมิให้เสียหาย

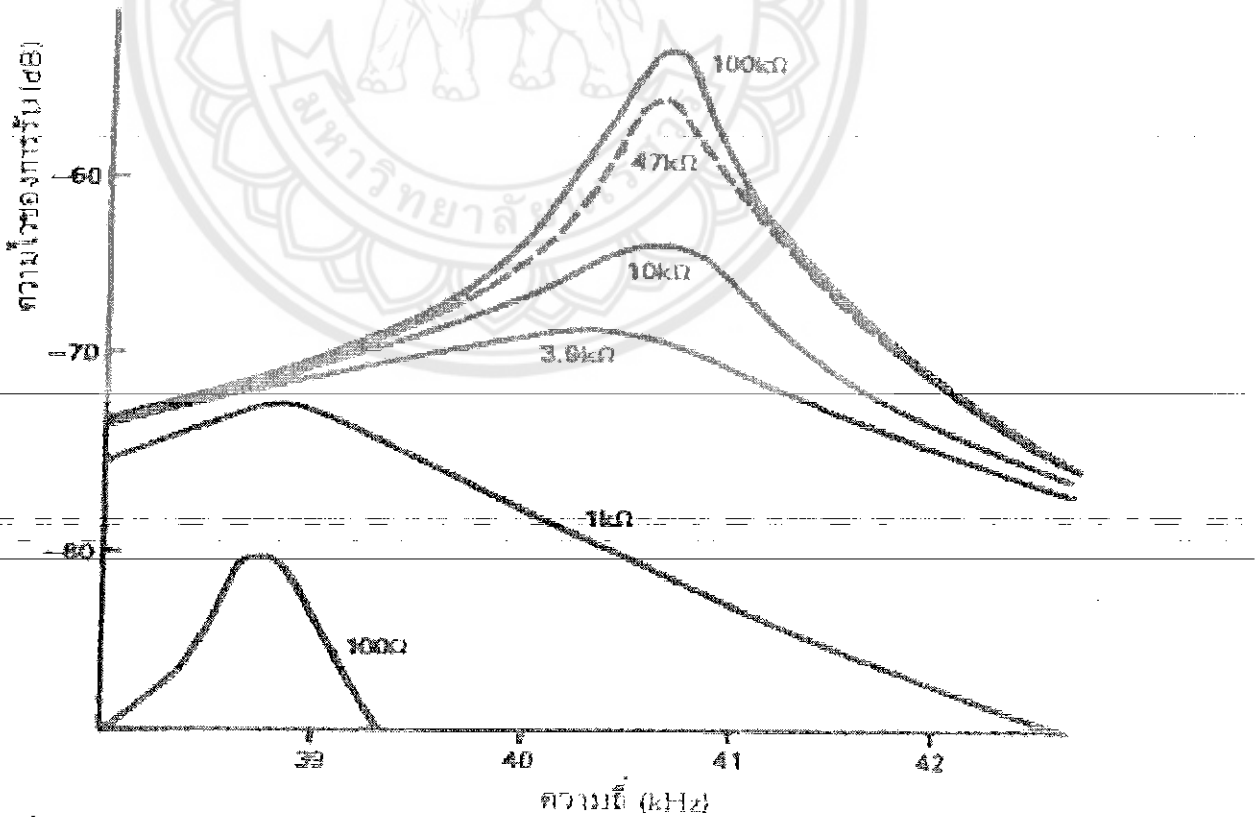
- ทรานสดิวเซอร์ที่มีขายกันโดยทั่วไปจะทนแรงดันตกคร่อมตัวมันสูงสุด ได้ไม่เกินกว่า 20 V_{rms} ดังนั้นขนาดของสัญญาณที่จะป้อนให้กับทรานสดิวเซอร์ก็ควรจะอยู่ภายในขีดจำกัด

- ความถี่เรโซแนนซ์ (ความถี่ที่ตัวมันทำงานได้อย่างมีประสิทธิภาพสูงสุด) ของทรานสดิวเซอร์ 40 KHz ที่มีขายกันโดยทั่วไปจะผิดพลาดไปไม่เกิน ? 1 KHz และมีแถบความถี่ (Bandwidth) ประมาณ 4.5 KHz สำหรับตัวส่ง และมีแถบความถี่ประมาณ 5.0 KHz สำหรับตัวรับ จะเห็นได้ว่าแถบความถี่ของตัวรับจะกว้างกว่าของตัวส่งอยู่เล็กน้อยเพื่อให้แน่ใจว่าตัวรับจะสามารถรับความถี่ทั้งหมดที่ออกมาจากตัวส่งได้

- อุณหภูมิใช้งานของตัวทรานสดิวเซอร์ควรอยู่ภายในช่วง –20 ? C ถึง +60? C

- ทั้งตัวส่งและตัวรับจะมีทิศทางคล้ายคลึงกันมากกล่าวคือ ที่ตำแหน่งเบนจากแนวแกนของตัวส่งไปประมาณ 30? ความแรงของคลื่นเสียงที่ถูกส่งออกไปจะลดลงจากแนวแกนประมาณ 10 dB ในทำนองเดียวกันถ้าคลื่นเสียงพุ่งเข้ามาในแนวที่เบี่ยงเบนไปจากแนวแกนขอตัวรับไปประมาณ 30? ความไวหรือขนาดแรงดันที่ออกมาก็ลดลงไปประมาณ 10 dB ด้วยเช่นกัน ดังนั้นในการใช้งานที่เป็นการควบคุมระยะไกลในที่โล่งแจ้ง จึงควรพยายามให้ทั้งตัวรับและตัวส่งอยู่ในแนวที่พุ่งตรงเข้าหากันให้มากที่สุด อย่างไรก็ตามในกรณีที่อยู่ในห้องอาจจะเบี่ยงเบนจากกันได้มากหน่อย เพราะคลื่นเสียงอุลตร้าโซนิคสามารถสะท้อนกับกำแพง พื้น และวัตถุที่อยู่ภายในห้อง ทำให้คลื่นเสียงเข้าไปหาตัวรับได้หลายทาง

- ในกรณีที่ใช้งานตัวรับจะต้องมีตัวต้านทานต่อขนานกับตัวรับ เพื่อทำหน้าที่เป็นโหลดตามปกติแล้วตัวต้านทานตัวนี้ควรมีค่าอยู่ในช่วงจาก 10 kΩ - 100 kΩ จากการทดลองพบว่าถ้าเปลี่ยนโหลดจาก 100 kΩ มาเป็น 10 kΩ ความไวจะลดลงประมาณ 10 dB ถึง 20 dB แต่แถบความถี่จะกว้างขึ้น ถ้าใช้ค่าความต้านทานต่ำลงไปอีก ความถี่เรโซแนนซ์ (ความถี่กลาง) จะลดลงไปจากที่ระบุไว้ ถ้าการใช้งานมีสัญญาณรบกวนมากควรใช้โหลดที่มีความต้านทานสูงสักหน่อยเพื่อให้ตัวส่งมีความไวสูงและมีแถบความถี่แคบ ตัวอย่างการทดสอบแสดงไว้ดังรูป 2.2

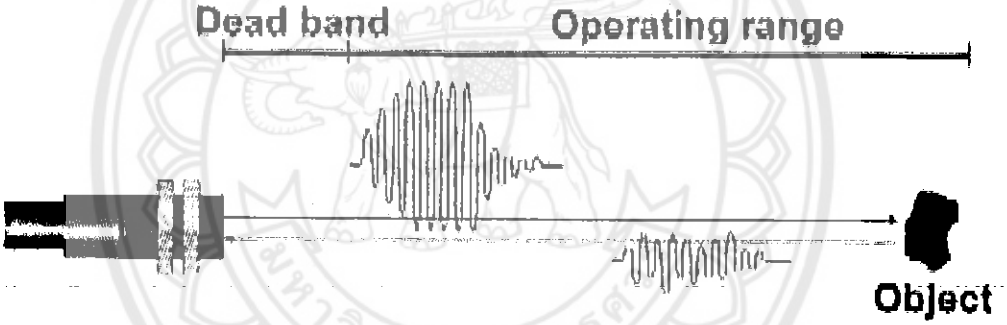


รูปที่ 2.2 แสดงผลการทดลองตัวรับตัวหนึ่งโดยลองเปลี่ยนโหลดเป็นค่าต่างๆกัน แล้วป้อนคลื่นเสียงความถี่ต่างๆกันเข้ามา

- ตามปกติแล้วเราสามารถนำเอาตัวส่งและรับมาใช้งานแทนกันได้ ในการใช้งานส่วนใหญ่ และตัวส่งหรือตัวรับของยี่ห้อใดรุ่นใดก็ได้ที่สามารถจะนำมาใช้แทนกันได้ในงานส่วนใหญ่ ขอเพียงแต่ให้มีความถี่เรโซแนนซ์เดียวกันเท่านั้นเอง อย่างไรก็ตามในบางกรณีอาจต้องเปลี่ยนแปลงค่าความต้านทานสมมูลย์ทางด้านไฟสลับ เพื่อให้ลักษณะผลตอบสนองทางความถี่สอดคล้องกับของเดิม

2.2 อุลตร้าโซนิคเซ็นเซอร์หน้าที่และการทำงาน

รูปแบบต่าง ๆ ของอุลตร้าโซนิคเซ็นเซอร์ประกอบด้วย ตัวตรวจจับด้วยคลื่นอุลตร้าโซนิค ชุดส่งสัญญาณ ชุดประมวลผล และชุดเอ้าท์พุทดังรูป 2.3

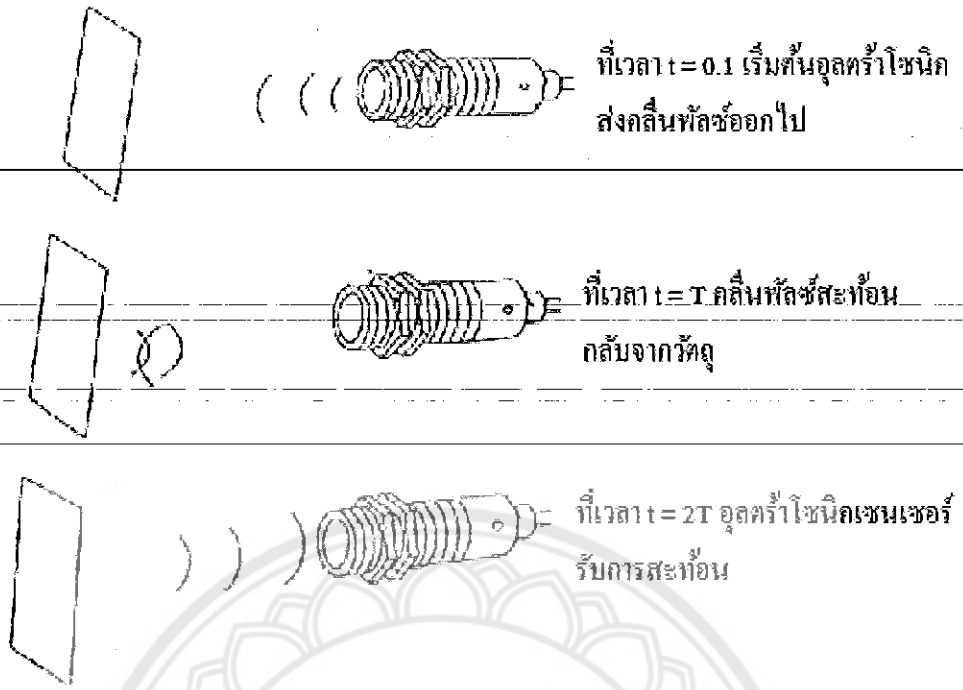


รูปที่ 2.3 หลักการทำงานของอุลตร้าโซนิค

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

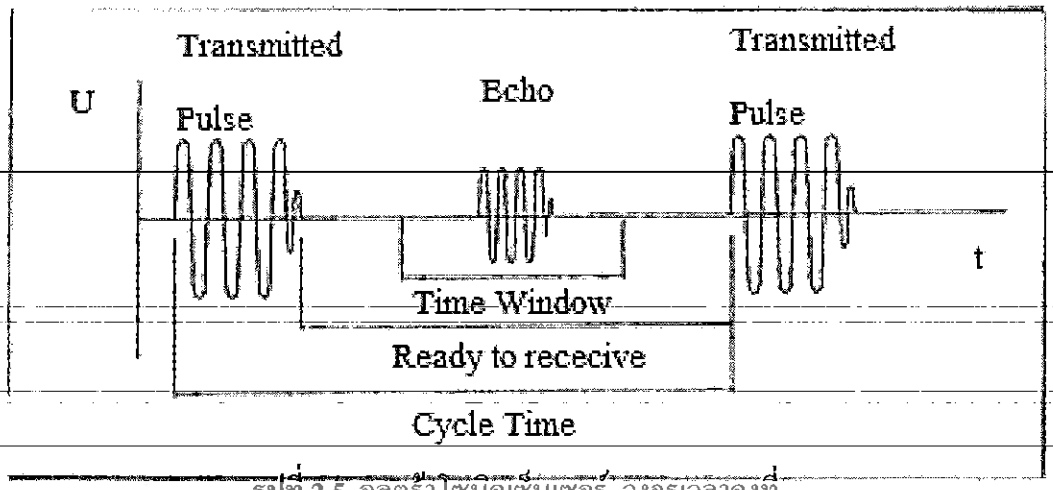
2.2.1 วงจรส่งผ่านและรับ

สำหรับการทำงานเป็นวงจรของอุลตร้าโซนิคเซ็นเซอร์ จะส่งผ่านคลื่นพัลซ์เสียงที่ช่วงเวลาสม่ำเสมอ หรือช่วงเวลาที่เปลี่ยนแปลง คลื่นเสียงที่ปล่อยออกไปจะถูกสะท้อนได้โดยวัตถุที่เหมาะสม โดยเซ็นเซอร์ และระบบการทำงานจะรับการสะท้อนของคลื่นเสียงที่สะท้อนกลับมา ดังแสดงในรูป 2.4 ความกว้างของคลื่นพัลซ์ของเสียงอยู่ในช่วง 2.-200 ไมโครเซท



รูปที่ 2.4 อุลตร้าโซนิคเซ็นเซอร์ วงจรส่งผ่านและรับ

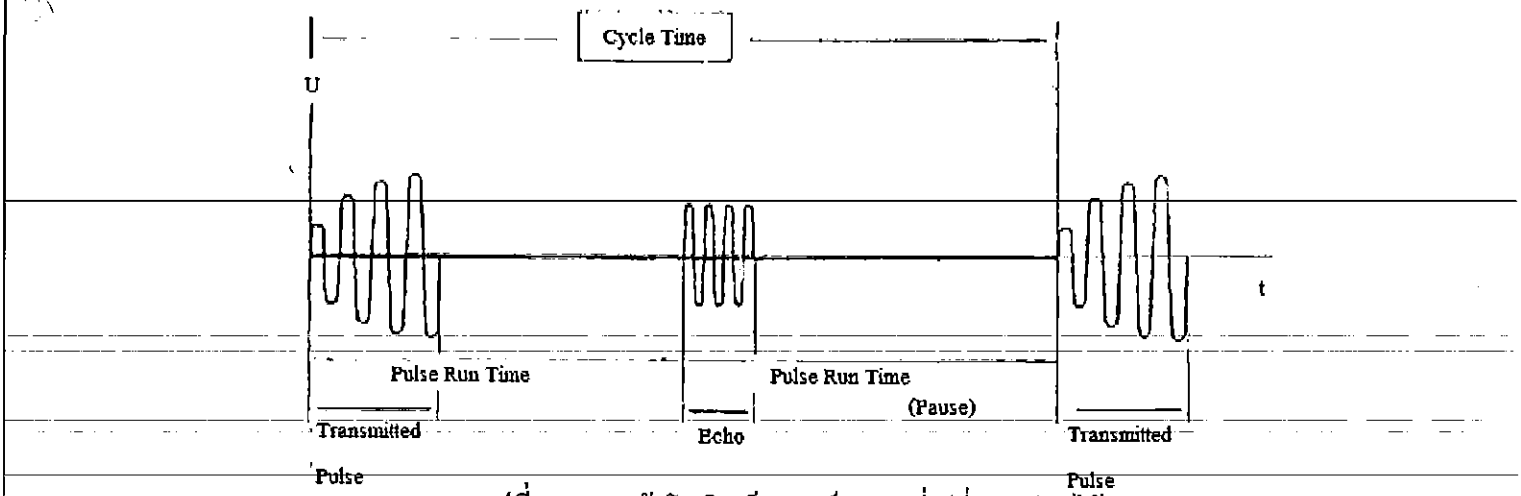
เวลาในการเดินทางของคลื่นพัลซ์ของคลื่นเสียงเป็นการวัดระยะห่างจากวัตถุ ทั้งนี้ขึ้นอยู่กับชนิดของเซ็นเซอร์ ระยะห่างนี้นำไปแสดงในรูปของ สัญญาอนาล็อก (Analog Signal) (เช่น 0-20 mA) สัญญาณลอจิก (Logic Signal) (เช่น สัญญาณลอจิก 8 bit) ตลอดทั้ง ซีเรียลอินเตอร์เฟส (Serial Interface) (RS232) หรือการเปรียบเทียบกับค่าอ้างอิงในรูปของสวิสซ์พัลซ์ที่เรียกว่า ไทม์เฟรม (Time Frame) เนื่องจากขบวนการดำเนินไปตามเวลาที่คลื่นสะท้อนเดินทาง ไม่ใช่เป็นไปตามความเข้มของคลื่นสะท้อน จึงจัดได้ว่าอุลตร้าโซนิคเซ็นเซอร์ มีข้อดีเหนือกว่าเซ็นเซอร์แบบออปติคอล (Optical Sensor) เวลาที่คลื่นสะท้อนการเดินทางจะ ทำให้ขบวนการดำเนินโดยไม่ขึ้นกับความเข้มของคลื่นสะท้อน ตราบเท่าที่วัตถุยังคงสะท้อนคลื่นที่สามารถตรวจจับได้ออกมา ดังนั้นคุณลักษณะการสวิทซ์ไม่เปลี่ยนไป แม้ในสภาวะที่การสะท้อนเป็นไปอย่างไม่ดีคลื่นสะท้อนที่อ่อนจะมีผลต่อความถูกต้องในการตรวจจับวัตถุ ซึ่งอาจทำให้ไม่สามารถทำการตรวจจับวัตถุได้เลยความเร็วที่เปลี่ยนไปของคลื่นพัลซ์ของเสียง มีผลกระทบต่อพิสัย การทำงานของสวิทช์ (ระยะทาง) โดยตรง เซ็นเซอร์ทำงานด้วยวงจรเวลาที่คงที่ (เช่น t = 20 ms) จะส่งคลื่นเสียงออกมาอย่างสม่ำเสมอ ดังแสดงในรูป 2.5 ดังนั้นวงจรเวลาจะเป็นตัวกำหนดช่วงและวงจรการทำงานของสวิทซ์ของเซ็นเซอร์



รูปที่ 2.5 อุลตร้าโซนิคเซ็นเซอร์, วงจรเวลาคงที่

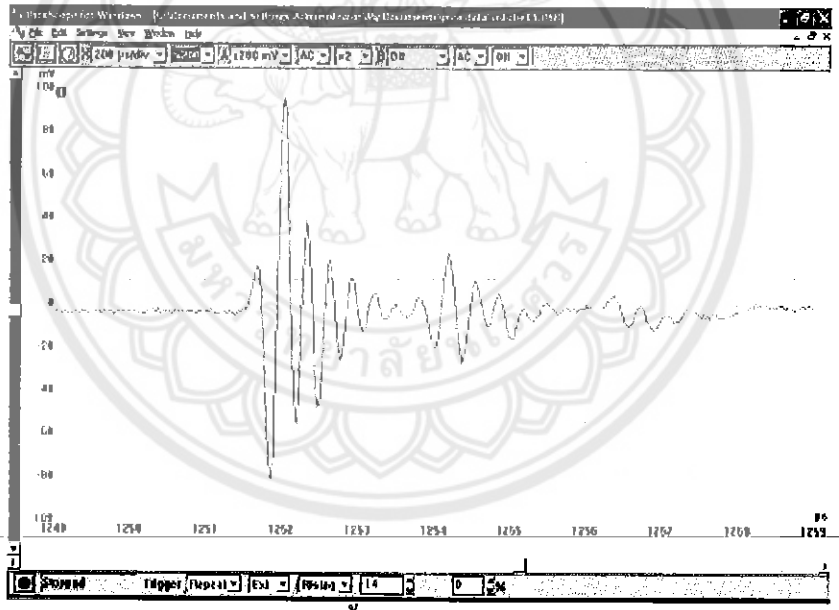
ยกตัวอย่าง เช่นคลื่นเสียงที่มีความเร็ว v = 340 m/s ในช่วงเวลา t= 20 ms (50 Hz) จะเดินทาง S= Vxt = 6.8 m เนื่องจากระยะห่างระหว่างเซ็นเซอร์ และวัตถุที่ได้จากการทำงานของเซ็นเซอร์ คิดที่ไปและกลับจึงได้ระยะทางจริงสูงสุดสำหรับวงจรเวลานี้เป็น 3.4 m แอมปลิจูดของส่วนของคลื่นเสียงและซิทิวิตี้ (Sensitivity) ของตัวรับต้องมีการพิจารณาเลือกใช้ เพื่อให้คลื่นสะท้อนที่เดินทางมาถึง หลังจากเลยเวลาของวงจรเวลาที่กำหนดไปแล้ว จะไม่ได้รับการตรวจจับ เนื่องจากคลื่นสะท้อนนั่นอ่อนมากซึ่งสัญญาณคลื่นนี้ จะทำให้เซ็นเซอร์สวิทส์มีการทำงานผิดพลาดหรือให้ข้อมูลที่ผิด

เซ็นเซอร์แบบอนาถอก (Analog Sensor) เพื่อให้การตรวจจับวัตถุเป็นไปอย่างถูกต้อง วัตถุต้องอยู่นิ่งเป็นเวลาที่พอเพียงสำหรับสำหรับสะท้อนอย่างน้อย 1 ส่วน ของคลื่นเสียงภายในขอบเขตที่เซ็นเซอร์จะทำงานได้รอบมากที่สุดของการสวิทซ์ ซึ่งจะเปลี่ยนแปลงกับอัตราส่วนของวัตถุต่อที่ว่าง และจะพิจารณาให้มีค่าน้อยกว่ารอบของความถี่ที่จุดนี้ เวลาที่ขยายจะสิ้นสุดระหว่างการส่งผ่านของพัลซ์ และการรับคลื่นสะท้อนแรกจะถูกนำไปใช้วัดสำหรับวงจรเวลา เมื่อเวลาดำเนินไปเท่ากับเวลาที่คลื่นสะท้อนเดินทางไป และกลับสิ้นสุด ส่วนของคลื่นสะท้อนต่อไปจะถูกส่งออก การหยุดลงชั่วขณะของเวลาพิเศษที่คลื่นเดินทางทำขึ้นเพื่อลดสัญญาณรบกวน (Noise) ที่ดำเนินมาจากตรวจจับวัตถุมากกว่าหนึ่งระยะ โดยเซ็นเซอร์สามารถถูกปรับให้เหมาะสมตามสภาพแวดล้อมซึ่งหมายถึง สำหรับการตรวจจับวัตถุที่อยู่ไกลเวลาการเดินทางจะนานเป็นผลให้ต้องการความถี่ต่ำในทางตรงกันข้าม ความถี่ของวงจรจะเพิ่มขึ้นเมื่อวัตถุเข้าใกล้เซ็นเซอร์ทำงานให้วงจรเวลาสั้นลง และพลังงานที่ส่งออกไปสามารถปรับในช่วงเวลาของคลื่นเสียงที่ปล่อยไป เวลาที่เพิ่มขึ้นของแอมปลิจูดเมื่อมีการสวิทซ์



รูปที่ 2.6 อุลตร้าโซนิคเซ็นเซอร์, วงจรที่เปลี่ยนแปลงได้

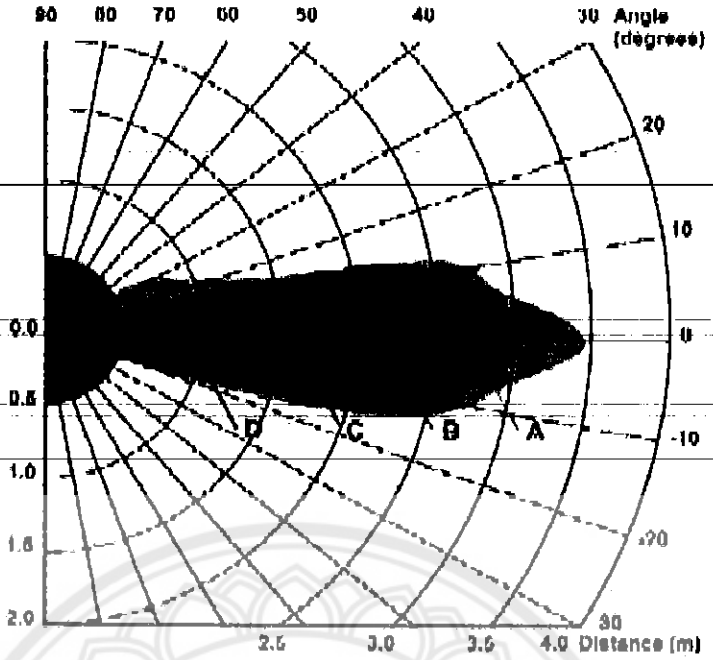
ทรานซ์ดิวเซอร์จะถูกใช้สำหรับพลังงานที่ส่งออกไป ดังแสดงในรูป 2.6 ดังนั้นคลื่นสะท้อนเบื้องหลังสามารถควบคุมได้ด้วย การลดพลังงานในการส่งผ่านวัตถุที่อยู่ใกล้เซ็นเซอร์



รูปที่ 2.7 อุลตร้าโซนิคเซ็นเซอร์แบบชิ้นส่วนการอิมพัลซ์ (Impulse) แตกออกที่170 MHz

2.2.2 การลดสัญญาณรบกวน และสภาวะการทำงาน

ผลที่เกิดจากคลื่นรบกวน และ การทรอดแทรกในการประยุกต์ใช้อุลตร้าโซนิคเซ็นเซอร์คือการตรวจจับวัตถุได้แต่ระยะที่ใกล้กับเซ็นเซอร์ และไม่สามารถตรวจจับวัตถุที่มีการสะท้อนได้ไม่ดี เนื่องจากความจริงที่ว่า คลื่นอุลตร้าโซนิคจะสะท้อนได้จากวัตถุเกือบทุกชนิด และง่ายต่อการเบียงเบน วัตถุเหล่านั้นจะทำให้สวิทซ์เปิด-ปิด เมื่อเข้าใกล้บริเวณที่เซ็นเซอร์สามารถตรวจจับได้ ดังกราฟทำนายคุณลักษณะของเซ็นเซอร์ ดังรูป 2.8



รูปที่ 2.8 อุลตร้าโซนิคเซ็นเซอร์, คุณลักษณะการตรวจจับ

เพื่อหาคุณลักษณะของวัตถุชนิดต่าง ๆ จะวางในตำแหน่งของวัตถุในระยะห่างเท่า ๆ กันที่มุมตั้งฉากกับแนวแกนของเซ็นเซอร์ จุดที่สวิทช์ทำงานก็จะถูกกำหนดขึ้น ตัวอย่างวัสดุที่ใช้คือ

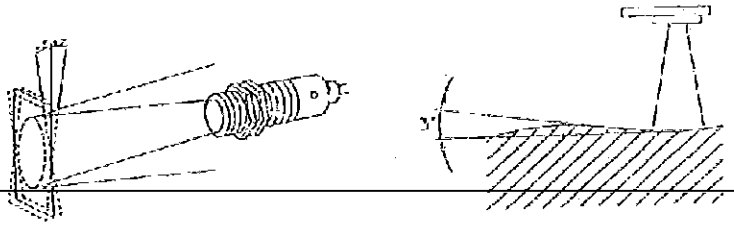
A: แผ่นจานขนาด 700 x 700 mm. ขอบเขตที่อยู่ด้านนอกส่วนโค้งชั้นนี้โดยปกติจะไม่มีวัตถุตรวจจับได้

B: แผ่นจานขนาด 100 x 100 mm. แผ่นจานอ้างอิงมาตรฐานกำหนดโดยข้อมูลทางเทคนิคทั่วไป

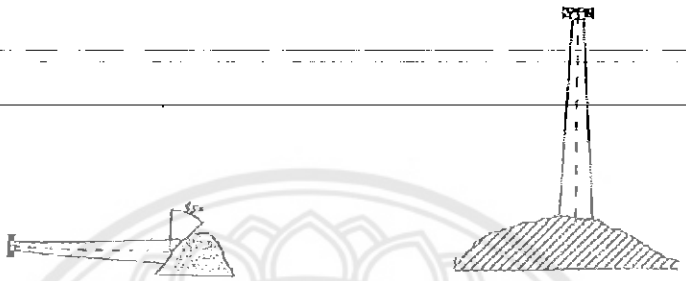
C: ท่อพลาสติก ขนาดเส้นผ่าศูนย์กลาง 160 mm. คลุมด้วยสักหลาด ใช้เป็นตัวแทนมาตราฐาน

D: แท่นไม้ขนาดเส้นผ่าศูนย์กลาง 25 mm. วัสดุทดสอบ เช่นระยะความปลอดภัยย้อนกลับในยานพาหนะ

เพื่อให้ปราศจากปัญหาในการทำงาน จะไม่มีวัสดุอื่นใดที่ไม่ใช่เป้าหมายในขอบเขตนอกสุด ในทางกลับกันวัตถุเป้าหมายต้องอยู่ภายในบริเวณพื้นที่ที่สามารถตรวจจับได้ทั้งขนาด รูปร่าง เพื่อป้องกันปัญหาการตรวจจับคลื่นเสียง พื้นผิวของวัตถุควรมีขนาดใหญ่เท่าที่จะเป็นได้ราบเรียบและมีมุมเอียงไม่เกิน 3องศา กับแกนของเซ็นเซอร์ ดังแสดงในรูป2.9 จากข้อกำหนดดังกล่าว เมื่อทำการตรวจจับวัตถุทรงกลม หรือวัตถุผิวไม่เรียบ (ของเหลว, ของผสม) ก็จะเกิดปัญหาขึ้น

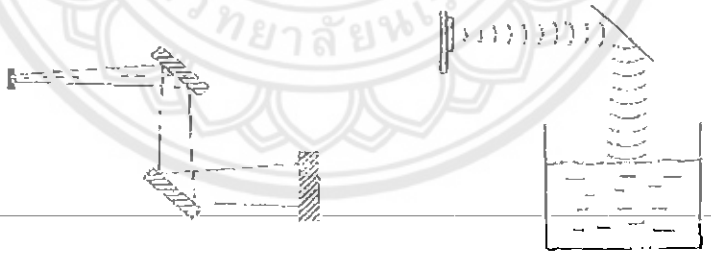


รูปที่ 2.9 อุลตร้าโซนิคเซ็นเซอร์, พื้นผิวตรง



รูปที่ 2.10 อุลตร้าโซนิคเซ็นเซอร์, การตรวจจับสิ่งของ

อุลตร้าซาวด์สามารถนำมาเบี่ยงเบนด้วยตัวสะท้อนอย่างง่าย ที่ทำจากวัสดุใด ๆ ดังแสดงใน รูป 2.10 พื้นที่ตรวจจับยังคงเท่าเดิม ทำให้ใช้กับตัวสะท้อนขนาดใหญ่ได้โดยใช้ตัวเบี่ยงเบนไม่เกิน 2 ตัว ติดตั้งภายในทางเดินของคลื่นเสียงในแนวทางเดินตั้งฉากอย่างถูกต้อง



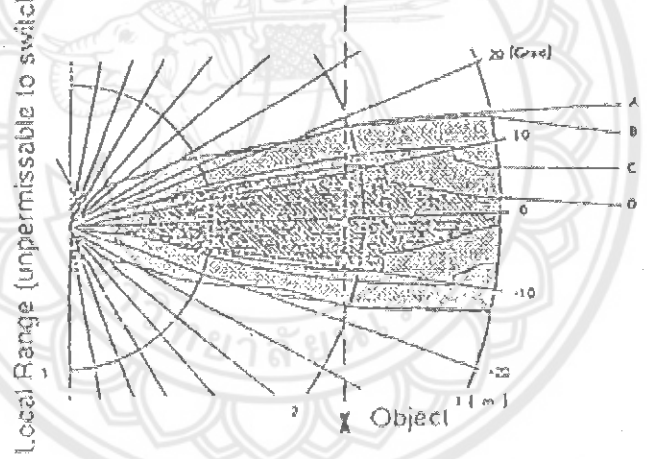
รูปที่ 2.11 อุลตร้าโซนิคเซ็นเซอร์, การเบี่ยงเบนคลื่นเสียง

ด้วยวิธีนี้สามารถนำไปใช้ป้องกันป้องกันเซ็นเซอร์ จากการใช้งานในสิ่งแวดล้อมที่เป็นอันตรายต่อเซ็นเซอร์และทำให้ลดคลื่นสะท้อนที่ไม่ต้องการจากวัตถุ การรวมคลื่นเสียงผ่านม่านกั้นหรือท่อโดยปกติไม่สามารถทำได้ เนื่องจากการหักเหของแสง บนขอบ และกำแพง เพื่อหลีกเลี่ยงการสอดแทรกจากเครื่องมือที่ให้กำเนิดคลื่นเสียงอันอื่น ๆ สัญญาณที่รับได้จะถูกทดสอบความถี่ในตัวเอง วิธีการนี้ไม่สามารถทำให้สำเร็จได้เมื่อใช้เซ็นเซอร์ชนิดเดียวกัน (ทรานซ์ดิวเซอร์และความถี่ชนิดเดียวกัน) หรือมีย่านรบกวนกว้าง (เช่น ในอากาศอัด) สอดแทรกกัน เพื่อหลีกเลี่ยงผลกระทบ

ระหว่างเซ็นเซอร์จะต้องมีการติดตั้งที่ระยะปลอดภัย 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 ความถี่ของการทำงานค่อนข้างลดน้อยลงเมื่อทำการตรวจจับวัตถุที่เคลื่อนที่ผ่านอย่างรวดเร็วผ่านจุดที่ทำมุมตั้งฉากกับแกนของเซ็นเซอร์ ในกรณีนี้ความเร็วของวัตถุที่ยอมให้มีได้กำหนดมาจากขนาดของวัตถุ และระยะห่างจากเซ็นเซอร์ ดังแสดงในรูปที่ 2.11



รูปที่ 2.12 อุลตร้าโซนิคเซ็นเซอร์, ความเร็วของวัตถุที่ยอมให้ได้

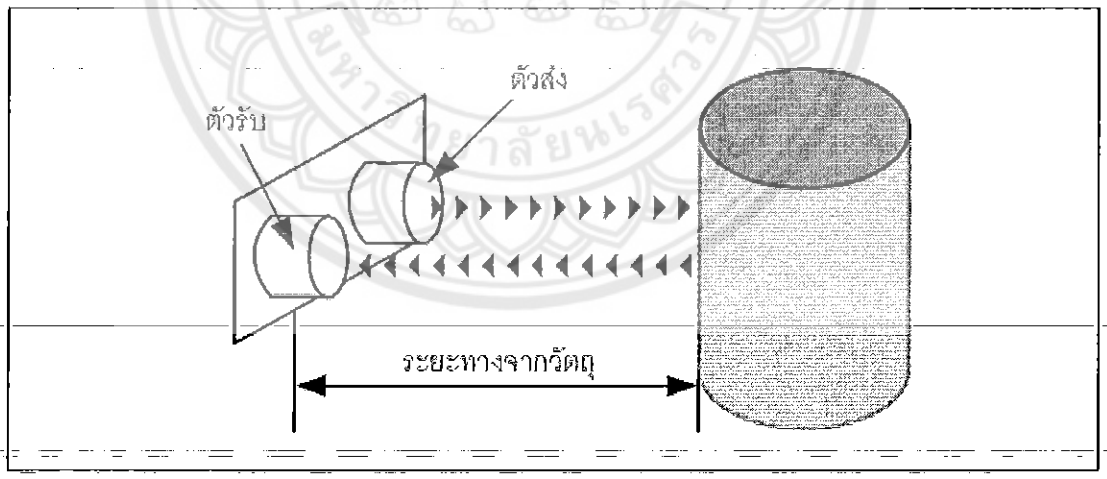
หากมีวัตถุมีรูปร่างเป็นสี่เหลี่ยมจัตุรัสขนาดความยาวของแต่ละด้านเท่ากับ 100 mm. (แผ่นจานอ้างอิงมาตราฐานส่วนโค้ง B)ในระยะห่าง 2 m จากเซ็นเซอร์พร้อมกับย่านความไว 3 m นั้นต้องการตรวจจับด้วยเซ็นเซอร์ จะต้องมีเวลาปฏิกิริยา 280 ms การพิจารณาประยุกต์ใช้งานดังนี้ ระยะเวลาช่องวัตถุในการตรวจจับ ดังแสดงไว้ในรูปด้านบน 2.12 : 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

2.3 อุลตร้าโซนิคแบบ SRF05

โครงงานนนี้เลือกใช้อุลตร้าโซนิคแบบ SRF05 เป็นแผงวงจรตรวจจับระยะทางด้วยคลื่นอุลตร้าโซนิคที่มีความเที่ยงตรงสูงโดยสามารถตรวจระยะทางได้ตั้งแต่ 1 เซนติเมตรไปจนถึง 4 เมตร SRF05 จะทำการส่งสัญญาณอุลตร้าโซนิค ออกไป และวัดระยะทางที่มีสัญญาณสะท้อนตอบกลับมา เอาต์พุตที่ได้จาก SRF05 จะอยู่ในรูปของความกว้างพลัลส์ซึ่งสัมพันธ์กับระยะทางของวัตถุที่ตรวจจับได้ ความถี่ของสัญญาณอุลตร้าโซนิคของ SRF05 คือ 40 kHz ถูกส่งออกไปในอากาศด้วยความเร็ว 1.125 ฟุตต่อมิลลิวินาที (ประมาณ 346 เมตรต่อวินาที) ดังนั้นเมื่อทราบความเร็วในการเคลื่อนที่ของคลื่น, เวลาเริ่มต้นส่งคลื่น และเวลาที่รับเสียงสะท้อนกลับมา จึงสามารถคำนวณหาระยะทางที่วัดได้ ดังแสดงหลักในการตรวจจับดังรูปที่ 2.13

ระยะทางที่ได้นั้นจะต้องมีการคำนวณค่ากลับทางคณิตศาสตร์ก่อน เมื่อใช้กับไมโครคอนโทรลเลอร์แล้วถือว่าเป็นเรื่องที่ยุ่งยากพอสมควร ดังนั้น SRF05 จึงประมวลผลค่าทางคณิตศาสตร์ต่างๆ เหล่านี้ไว้เรียบร้อยแล้ว จากนั้นส่งผลที่วัดได้ออกมาเป็นพลัลส์ที่มีความกว้างสัมพันธ์กับระยะทางที่วัดได้

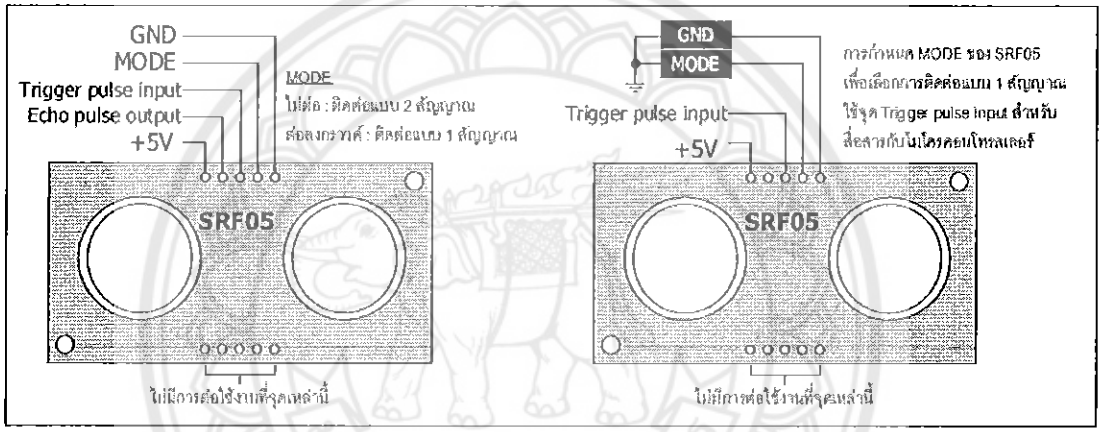
การส่งผลลัพธ์ที่วัดได้ออกมาในเชิงความกว้างของสัญญาณพลัลส์ อาจจะดูยุ่งยากกว่าการส่งข้องมูลเป็นดิจิตอลออกมาก แต่การส่งข้อมูลดิจิตอลอาจต้องใช้สายสัญญาณจำนวนมากซึ่งทำให้ต้องใช้ขาพอร์ตในการเชื่อมสัญญาณจำนวนมากตามไปด้วย ดังนั้นหากผลลัพธ์ออกมาในรูปของสัญญาณพลัลส์จะใช้สายสัญญาณเพียงเส้นเดียว จึงสะดวกในการเชื่อมต่อกับไมโครคอนโทรลเลอร์



รูปที่ 2.13 แสดงหลักการตรวจจับวัตถุโดยใช้สัญญาณความถี่เหนือเสียงหรืออุตร้าโซนิค

2.3.1 คุณสมบัติของ SRF05

- ใช้ไฟเลี้ยง 5V ต้องการกระแส 30mA
- ใช้ตัวรับและส่งคลื่นอุลตร้าโซนิค ใช้ความถี่ 40kHz
- วัดระยะทางในช่วง 1 เซนติเมตรถึง 4 เมตร
- สัญญาณพลัลส์สำหรับกระตุ้นการทำงานต้องมีความกว้างอย่างน้อย 10 ไมโครวินาที
- ให้ผลลัพธ์จากการวัดระยะเป็นค่าความกว้างพลัลส์ซึ่งเป็นสัดส่วนกับระยะทางที่วัดได
- สื่อสารกับไมโครคอนโทรลเลอร์ยอดนิยมได้ทุกตะกูล เช่น เบสิกแสตมป์ 2SX/2P, Pic, MCS-51, PSoC, 68HC11, AVR-MEGA128
- สามารถติดต่อได้ 2 แบบคือ แบบ 2 สัญญาณ (Echo กับ Trigger) เหมือนกับ SRF04 และแบบอนุกรมสัญญาณเส้นเดียวสามารถใช้แทน SRF04 ได้



รูปที่ 2.14 การแสดงขาสัญญาณของ SRF05 และโหมดการทำงานต่างๆ

2.3.2 จุดต่อใช้งานของ SRF05

มีจุดต่อสำหรับการใช้งานทั้งหมด 5 จุดตามรูปที่ 2.14

ขาไฟเลี้ยง (+5V)

สำหรับต่อไฟเลี้ยงแรงดัน +5V

ขา Echo Pulse Output (ECHO)

เป็นขาเอาต์พุตสำหรับส่งสัญญาณพลัลส์ออกจาก SRF05 ซึ่งการใช้งานจะนำขานี้ไปต่อเข้ากับพอร์ตอินพุตของไมโครคอนโทรลเลอร์ เพื่อตรวจจับความกว้างของสัญญาณพลัลส์ที่ส่งออกมาเพื่อแปลความหมายออกเป็นระยะทางอีกครั้งหนึ่ง

ขา Trigger Pulse Input (TRIGGER)

เป็นขาอินพุตรับสัญญาณพลัลส์ที่มีความกว้างอย่างน้อย 10 ไมโครวินาที เพื่อกระตุ้นสร้างคลื่นอุลตร้าโซนิคที่มีความถี่ 40 kHz ออกสู่อากาศจากตัวส่ง ดังนั้นเมื่อความถี่ดังกล่าวนี้เคลื่อนที่กระทบสิ่งกีดขวางที่อยู่เบื้องหน้าก็จะเกิดการสะท้อนกลับมายังตัวรับ และถูกแปลงออกมาเป็น

ความกว้างของสัญญาณพลัสที่ส่งออกไปทางขา Echo Pulse Output นอกจากนี้ยังมีในโหมด 1 สัญญาณ จะใช้จุดนี้เป็นจุดสื่อสารข้อมูลอนุกรมเพื่อรับส่งค่าการวัดกับไมโครคอนโทรลเลอร์

ขา MODE สำหรับเลือกรูปแบบการติดต่อกับ SRF05

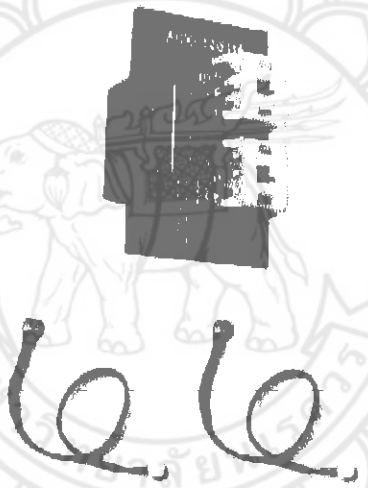
ปล่อยไว้ (NC) : เลือกให้ติดต่อแบบ 2 สัญญาณผ่านจุดต่อ ECHO และ TIGGER

ต่อลงกราวด์ (GND) : เลือกให้ติดต่อแบบ 1 สัญญาณ ผ่านจุดต่อ TRIGGER

ขา GND สำหรับต่อลงกราวด์

2.3.3 อุปกรณ์เสริม

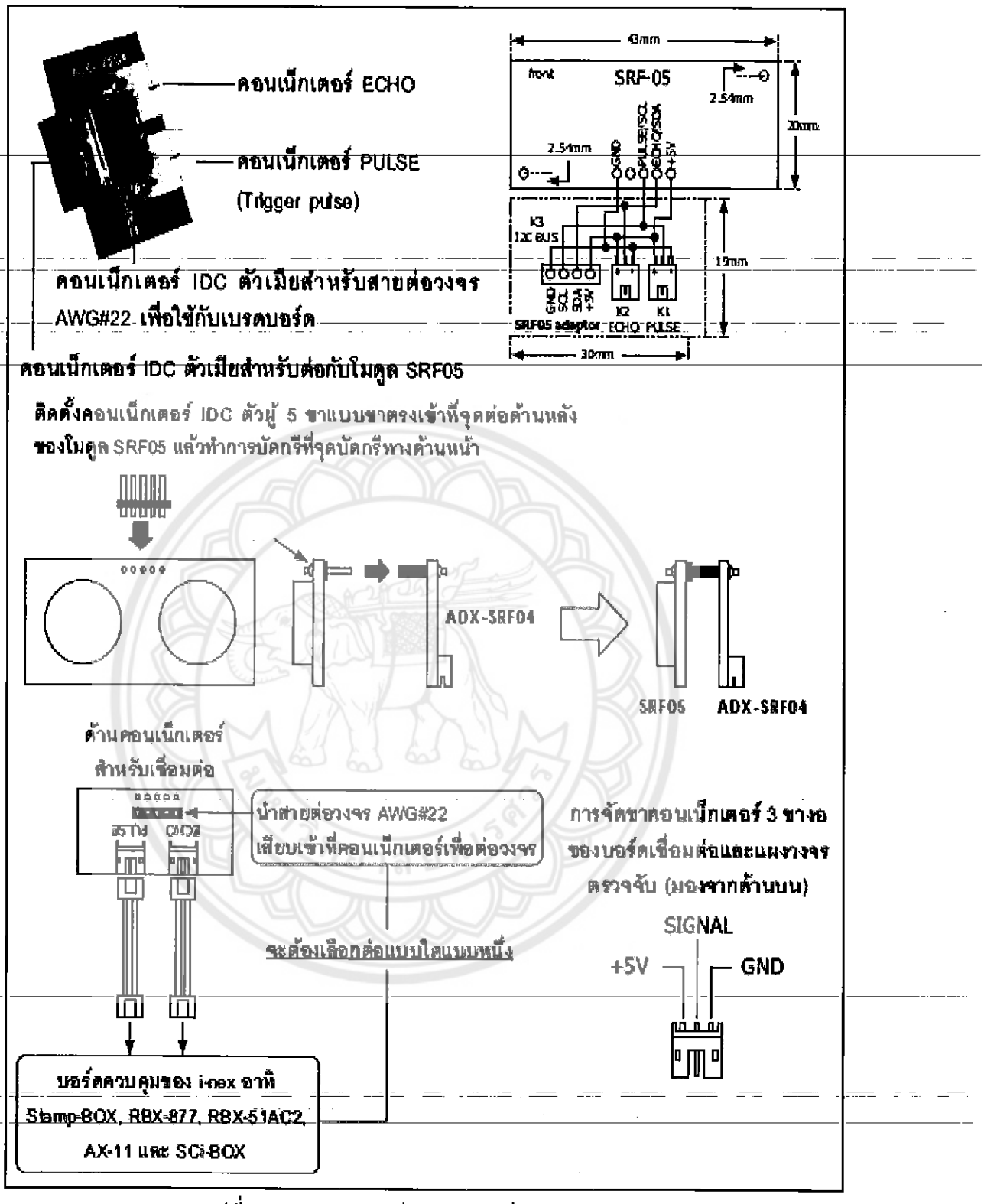
- บอร์ดADX-SRF05 ซึ่งเป็นบอร์ดอะแดปเตอร์สำหรับอำนวยความสะดวก ในการเชื่อมกับบอร์ดไมโครคอนโทรลเลอร์และแผงต่อวงจรบอร์ด
- สาย PCB3A สำหรับเชื่อมต่อกับบอร์ดควบคุมหุ่นยนต์ของ i-nex



รูปที่ 2.15 แสดงบอร์ด ADX-SRF05 และสาย PCB3A

2.3.4 การใช้งานบอร์ด ADX-SRF05

เพื่ออำนวยความสะดวกแก่ผู้ใช้งานกับบอร์ดควบคุม จึงได้สร้างบอร์ด ADX-SRF05 ขึ้นมา โดยบนบอร์ด ADX-SRF05 ได้จัดเตรียมคอนเน็กเตอร์ PCB3 ขาตัวผู้ 2 ตัวแยกกันระหว่างสัญญาณ ECHO และ TRIGGER สำหรับเชื่อมต่อกับบอร์ดควบคุม และคอนเน็กเตอร์ IDC ตัวเมียแถวเดียว 4 ขาสำหรับเสียบสายต่อวงจรเบอร์ AWG#22 เพื่อต่อกับแผงวงจรหรือบอร์ดโดยวงจรของบอร์ด ADX-SRF05 และการติดตั้งเพื่อใช้งานแสดงได้ดังรูป 2.16



รูปที่ 2.16 แสดงการใช้งานของบอร์ด ADX-SRF05

บทที่ 3
การพัฒนาเครื่องวัดระยะด้วยเซนเซอร์ชนิดอุลตร้าโซนิค

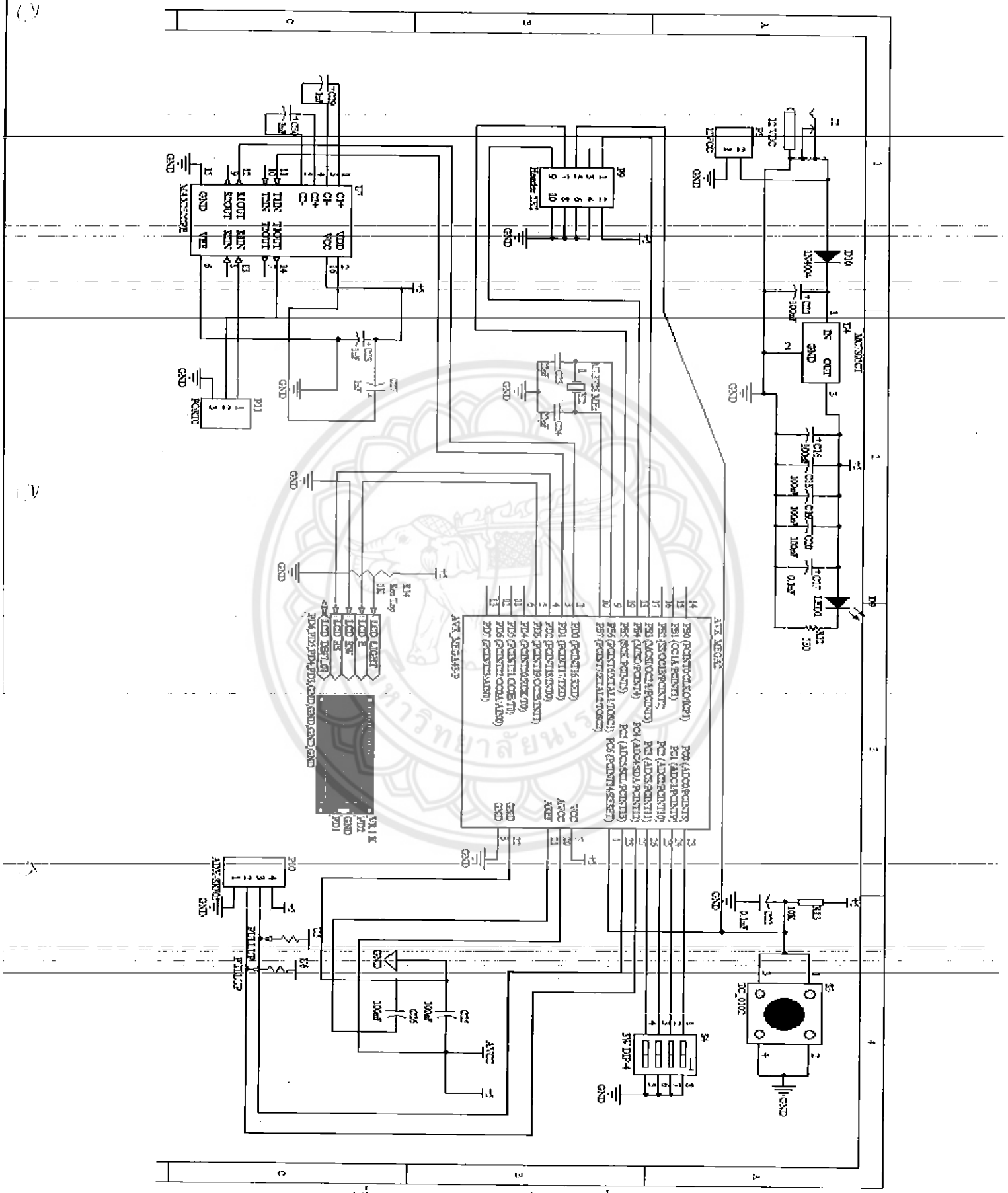
3.1 ไมโครคอนโทรลเลอร์

3.1.1 บอร์ดทดลอง

ส่วนประกอบภายในเครื่องวัดระยะ จะมีไมโครคอนโทรลเลอร์เป็นหัวใจหลักในการทำงาน โครงงานนี้เลือกทำบอร์ดที่ออกแบบเอง ซึ่งมีคุณสมบัติดังนี้

- ใช้กับ CPU ตระกูลAVR เบอร์ ATmega48
- CRYSTAL 16 MHz
- RS232 1 CHANNEL
- LCD PORT 16 PIN (8 BIT INTERFACE)
- +5 VOLT REGULATOR ON BOARD
- POWER ON RESET
- HEADER 4 PIN FOR SRF05
- VR FOR VALUE BLACK LIGHT LCD

บอร์ดนี้ออกแบบมาเพื่อใช้กับการใช้งาน ULTRA-SONIC RANGER (SRF05) โดยเฉพาะ ได้ทำการออกแบบ และ Interface ระบบเอง แสดงแบบ Schematic ใช้การออกแบบจากโปรแกรม Altium Designer 6 ดังรูป 3.1



รูปที่ 3.1 รูปวงจรของบอร์ดทดลองที่ออกแบบเอง

3.1.2 ไมโครคอนโทรลเลอร์ ATmega48

ไมโครคอนโทรลเลอร์ AVR เป็นหนึ่งในไมโครคอนโทรลเลอร์ตระกูลใหม่จาก ATMEL (ผู้นำทางด้านไมโครคอนโทรลเลอร์ตระกูล MCS-51) AVR จัดเป็นไมโครคอนโทรลเลอร์ตระกูลใหม่จาก ATMEL มีสถาปัตยกรรมแบบ RISC (Advance RISC architecture) คือหนึ่งคำสั่งการทำงานใช้สัญญาณนาฬิกาเพียง 1 ลูก (instructions in a single clock cycle) เป็นไมโครคอนโทรลเลอร์ที่มีประสิทธิภาพและความสามารถสูง แบ่งออกเป็นหลายอนุกรม ในแต่ละอนุกรมยังแบ่งออกเป็นหลายเบอร์ เพื่อรองรับความต้องการที่แตกต่างจากผู้ใช้งาน ในขณะที่ยังคงประสิทธิภาพเท่ากัน และคุณสมบัติของไมโครคอนโทรลเลอร์ ATmega48 มีคุณสมบัติที่สำคัญดังนี้

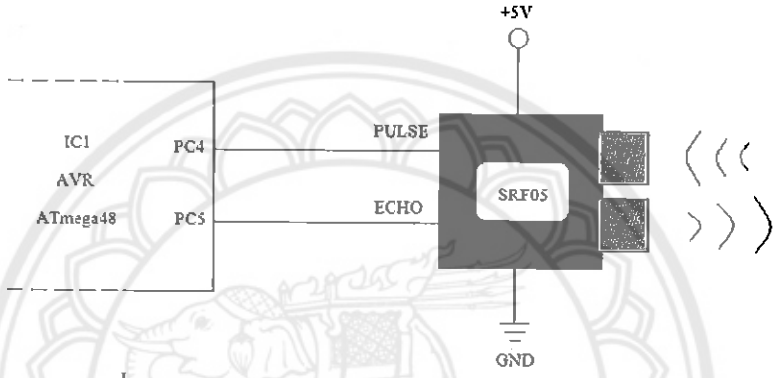
- สถาปัตกรรมภายในเป็นแบบ Advance RISC (Reduce Instruction Set Computer)
- มีคำสั่งการทำงานมากกว่า 100 คำสั่ง โดยมีความเร็วในการประมวลผล 1 คำสั่งต่อ 1 สัญญาณนาฬิกา (1 MIP/1 MHz)
- มีรีจิสเตอร์ใช้งานทั่วไปขนาด 8บิต จำนวน 32ตัว (ทำให้สะดวกต่อการพัฒนาโปรแกรมด้วยภาษา C เป็นอย่างมาก)
- ความเร็วในการทำงาน 1 MIP ต่อ 1 MHz และมากถึง 16 MIPS เมื่อใช้ความถี่ 16 MHz (ความ สามารถในการใช้งานความถี่สัญญาณนาฬิกาขึ้นอยู่กับเบอร์ที่เลือกใช้งาน)
- หน่วยความจำ ROM แบบ Flash (มีโหมดป้องกันหน่วยความจำ) ขนาด 16 กิโลไบต์ (เขียน/ลบได้ 10,000 ครั้ง)
- หน่วยความจำข้อมูลแบบ EEPROM (มีโหมดป้องกันหน่วยความจำ) ขนาด 512 ไบต์ (เขียน/ลบได้ 100,000 ครั้ง)
- หน่วยความจำข้อมูลแบบ SRAM 1 กิโลไบต์
- ไทเมอร์/เค้าเตอร์ทั้งแบบ 8 บิตและ 16 บิตพร้องปรีสเกลเลอร์
- มีระบบตรวจสอบความผิดพลาดในการทำงานซอฟต์แวร์ (Watchdog Timer with On-Chip Oscillator)
- โมดูลสร้างสัญญาณ PWM (Pulse Width Modulator) มีจำนวน 4 ช่อง
- โมดูลแปลงสัญญาณอนาลอกเป็นดิจิตอล (ADC) ขนาด 10 บิต มากถึง 8 ช่อง
- โมดูลเปรียบเทียบแรงดันอนาลอก (Analog Comparator)
- การสื่อสารข้อมูลอนุกรมมีทั้งแบบ UART (Universal Asynchronous Receiver Transmitters) หรือแบบ RS232, SPI (Serial Peripheral Interface) และแบบ I^2C
- พอร์ตอินพุเอ้าพุตขึ้นอยู่กับเบอร์ AVR ที่เลือกใช้งานมีตั้งแต่ 8 ขา จนมากกว่า 100 ขา พอร์ต (ATmega 48 มีขาพอร์ตอินพุตเอาท์พุต 23 ขา)
- แรงดันและไฟเลี้ยงในการทำงานขึ้นอยู่กับ เบอร์ที่ใช้งาน

3.2 วงจรการทำงานแต่ละส่วน

จากโครงงานที่ทำนี้ได้แบ่งออกเป็น 2 ส่วนหลักๆ คือ ส่วนที่ทำการวัดระยะทาง เชื่อมต่อกับโมดูล SRF05 และส่วนที่แสดงผลเชื่อมกับ LCD

3.2.1 วงจรของ SRF05 กับ Microcontroller

การต่อวงจรไมโครคอนโทรลเลอร์ เข้ากับ SRF05 ใช้แค่ 2 ขาพอร์ตในการเชื่อมต่อ ในโครงงานนี้ใช้ขาพอร์ต PC4 และ PC5 ในการเชื่อมกับโมดูล SRF05 เป็นขา TRIGGER และ ECHO แสดงได้ดังรูปที่ 3.2



รูปที่ 3.2 วงจรไมโครคอนโทรลเลอร์ที่เชื่อมกับ SRF05

3.2.2 วงจรของ LCD

การแสดงผลออกทาง LCD ในโครงงานนี้ใช้ LCD ที่แสดงผลแบบ 20 ตัวอักษร 4 แถว และมี Backlight และการใช้งานของ LCD แสดงได้ดังตารางต่อไปนี้

ตารางที่ 3.1 การทำงานของขาพอร์ต LCD

No.	Symbol	Function
1	VSS	Power ground (0V)
2	VDD	Power supply for Logic(+5V)
3	V0	Power supply for LCD drive
4	RS	Register selection (H: Data register , L :Instruction register)
5	R/W	Read/write selection (H: Read , L: Write)
6	E	Enable signal.
7-14	DB0~DB7	Data Bus line
15	LEDA	Power supply for Backlight(+5V)
16	LEDK	Power supply for Backlight(0V)

MISSING

ห้องสมุดคณะวิศวกรรมศาสตร์ มหาวิทยาลัยนเรศวร

I 6078685 C. 2

3.3 การใช้งานกับไมโครคอนโทรลเลอร์

3.3.1 ฟังก์ชันโปรแกรมภาษา C สำหรับติดต่อระหว่าง ATmega48 กับโมดูล SRF05 ในโหมดการติดต่อแบบ 2 สัญญาณ

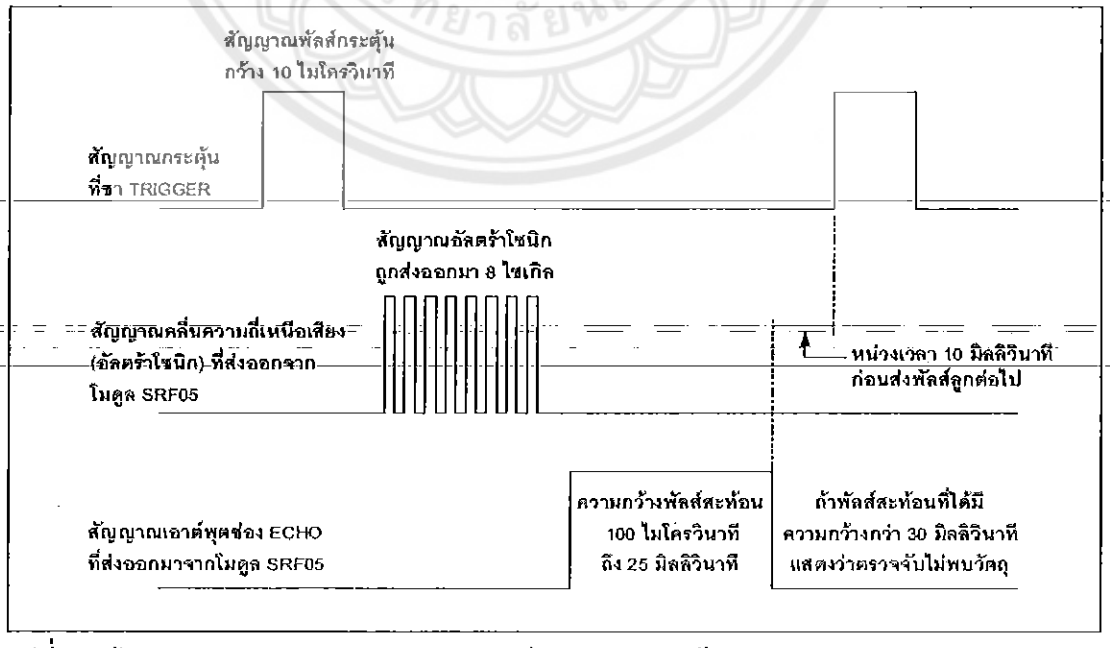
ว.ม. วิ377ค 2551

ในการเขียนโปรแกรมภาษา C สำหรับไมโครคอนโทรลเลอร์ ATmega48 เพื่อใช้งานโมดูล SRF05 สิ่งที่ควรทำคือ การสร้างฟังก์ชั่นในการเชื่อมต่อเพื่ออ่านค่าและประมวลผล สำหรับที่ทำในบทนี้คือ ฟังก์ชั่น Distance

ฟังก์ชั่นที่นำมาใช้อ่านค่าระยะทางที่วัดได้จากโมดูล SRF05 เริ่มต้นด้วยการกำหนดให้ขาพอร์ต PC5 ของไมโครคอนโทรลเลอร์ ATmega48 ต่อกับขา ECHO ของโมดูล SRF05 และกำหนดให้เป็นพอร์ตอินพุต โดยมีการกำหนดลอจิก "1" เพื่อเริ่มต้นการทำงาน ส่วนขา TRIGGER ของโมดูล SRF05 ต่อเข้ากับขาพอร์ต PC4 ดังนั้นจึงต้องกำหนดเป็นเอาท์พุต เพื่อสร้างสัญญาณพลัสบวกที่มีความกว้าง 10 ไมโครวินาที เพื่อเป็นการกระตุ้นโมดูล SRF05 เริ่มต้นทำการวัดระยะทาง

ทันทีที่โมดูล SRF05 ได้รับสัญญาณพลัสบวกเข้าที่ขา TRIGGER โมดูล SRF05 จะดำเนินการสร้างขบวนการพลัสความถี่ 40 kHz ออกสู่อากาศ ขบวนการพลัสความถี่เหนือเสียงหรืออุลตร้าโซนิคกลุ่มนี้จะเดินทางเป็นเส้นตรง จนกระทั่งกระทบวัตถุที่อยู่เบื้องหน้า ทำให้ขบวนการอุลตร้าโซนิคสะท้อนกลับเข้ามายังตัวรับของโมดูล SRF05 หลังจากนั้นหน่วยประมวลผลภายในโมดูล SRF05 จะวิเคราะห์และให้สัญญาณพลัสเอาท์พุตออกมาเป็นความกว้างพลัสบวกที่เป็นสัดส่วนโดยตรงกลับระยะทางที่ตรวจจับได้ ดังรูป 3.5

5200034



รูปที่ 3.5 ไดอะแกรมเวลาแสดงการส่งสัญญาณไปและการสะท้อนสัญญาณกลับของ SRF05 ในกรณีติดต่อแบบ 2 สัญญาณ

สำหรับการนับความกว้างที่ขา ECHO จะใช้ไทเมอร์ 1 ในโหมด 16 บิต เป็นตัวนับเวลาความกว้างของสัญญาณพลัลส์ โดยเริ่มต้นนับเมื่อพบว่ามีการเปลี่ยนแปลงลอจิกแบบขาขึ้น จาก "0" ไปเป็น "1" และหยุดการนับเมื่อพบการเปลี่ยนแปลงลอจิกแบบขาลง จาก "1" ไปเป็น "0"

หลังจากนั้นนำค่าที่ได้ไปหารด้วย 114 จะเป็นค่าประมาณที่ใกล้เคียงจากการเปลี่ยนแปลงข้อมูลระยะทางที่ได้จากโมดูล SRF05 ไทเมอร์ของไมโครคอนโทรลเลอร์ที่ใช้นั้นจะอ้างอิงสัญญาณนิกา ความถี่ 16 MHz และไมโครคอนโทรลเลอร์ ATmega48 ทำงานด้วยความเร็ว ดังสูตรต่อไปนี้

จาก

$$T = 1 / f$$
$$= 1 / (16\ \text{MHz})$$

ที่ความถี่คริสตอลเท่ากับ 16 MHz ใช้ค่าปรีสเกลเลอร์เท่ากับ 8 เพื่อให้การนับ 1 ครั้งใช้เวลาเท่ากับ 0.5 ไมโครวินาที

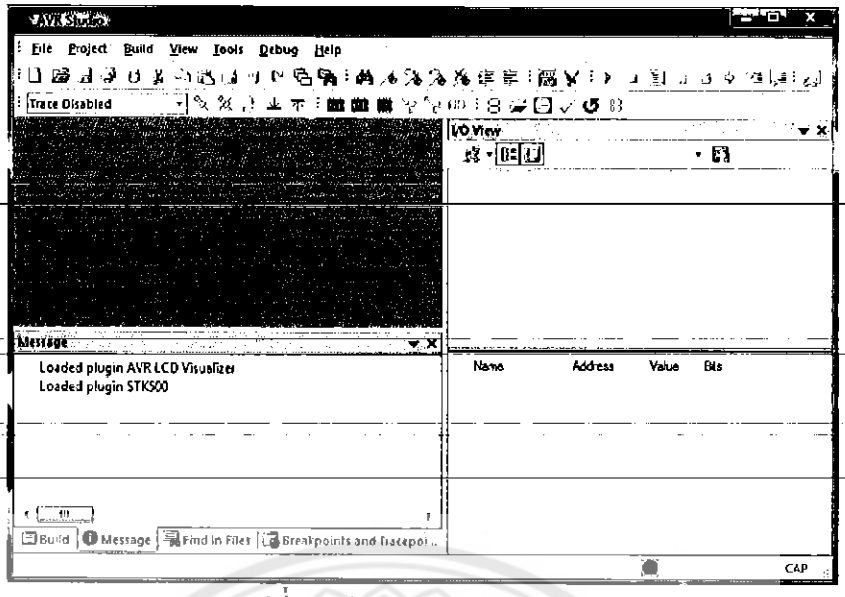
$$T = 1 / (16\ \text{MHz} / 8)$$
$$= 1 / (2\ \text{MHz})$$
$$= 0.5 \quad \text{ไมโครวินาที}$$

3.3.2 การพัฒนาโปรแกรมด้วย AVR Studio4

AVR Studio 4.xx เป็นเครื่องมือในการพัฒนาโปรแกรมภาษาแอสแซมบลี (Assembly) สำหรับไมโครคอนโทรลเลอร์ AVR ในรูปแบบ IDE (รวมรูปแบบเครื่องมือพัฒนาทั้งแอวแซมเบอร์ ดีบักเกอร์ และอิดิเตอร์ ไว้ในตัวเดียวกัน) และเมื่อนำมาทำงานร่วมกับ WinAVR ก็จะรวมคอมไพเลอร์ภาษา C สำหรับ AVR (GCC compiler for AVR) เข้าไปด้วยเนื่องจาก AVR Studio ได้เปิดโอกาสให้ผู้พัฒนาคอมไพเลอร์ สามารถนำโปรแกรมหรือตัวคอมไพเลอร์ภาษาอื่นๆเข้าไปใช้งานร่วมกับ AVR Studio ได้เรียกว่าการผนวกเครื่องมือจากภายนอก (Plug-in) เมื่อรวม AVR Studio และ WinAVR เข้าด้วยกันแล้ว เราก็สามารถพัฒาโปแกรมภาษา C ผ่าน AVR Studio ได้เช่นเดียวกับคอมไพเลอร์ภาษา C สำหรับ AVR ที่มีจำหน่าย เช่น CodeVision AVR, ImageCraft AVR เป็นต้น

3.3.3 การใช้งาน AVR Studio/WinAVR

หลังจากที่ได้ติดตั้งโปรแกรม AVR Studio และ WinAVR เรียบร้อยแล้ว เมื่อเรียกใช้งาน AVR Studio จะได้หน้าต่างดังรูปที่ 3.6 และมีขั้นตอนการใช้งานดังนี้

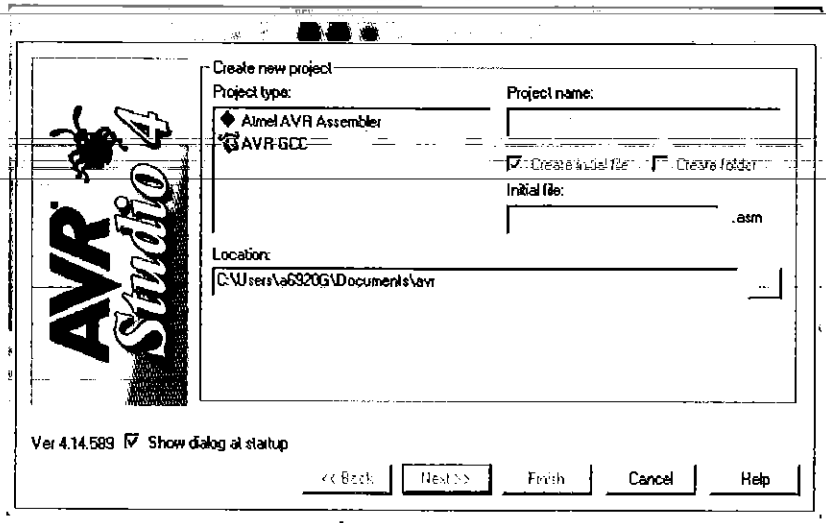


รูปที่ 3.6 โปรแกรม AVR Studio

3.3.4 สร้างโปรเจกต์ใหม่

- ในเมนูโปรเจกต์ (Project) เลือกรายการ New Project ดังรูปที่ 3.7 จะได้หน้าต่าง New Project ดังรูปที่ 3.8

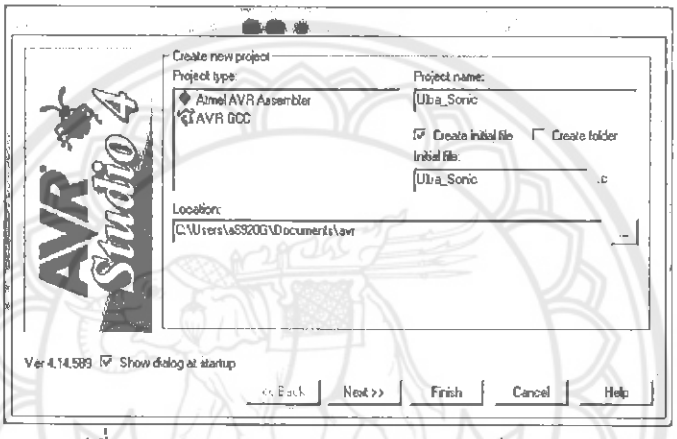
รูปที่ 3.7 เมนู Project



รูปที่ 3.8 เมนู Project

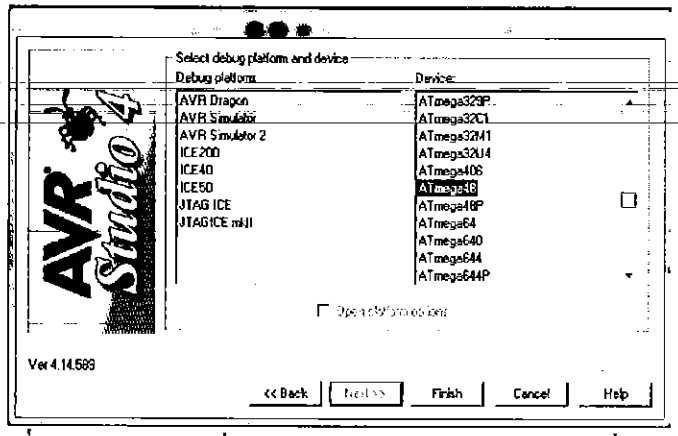
ในส่วนของ Create new project ช่อง Project type ผู้ใช้งานสามารถเลือกการพัฒนาโปรแกรมด้วยภาษาแอแซมบลี (Atmel AVR Assembly) หรือ ภาษา C (AVR GCC) อย่างใดอย่างหนึ่งได้ ในที่นี้เลือก AVR GCC ดังรูปที่ 3.9

- คลิกเลือกไปที่ AVR GCC จากนั้นป้อนชื่อไฟล์โปรเจกต์ที่ต้องการสร้างในช่อง Project name: และไฟล์เริ่มต้น (Initial file) จะเป็นชื่อเดียวกับไฟล์โปรเจกต์ โดยไฟล์โปรเจกต์และไฟล์โปรแกรม (.c) จะถูกเก็บอยู่ในไดเรกทอรีและโฟลเดอร์ที่ช่อง Location (ช่อง Initial file และ Location สามารถเปลี่ยนแปลงได้ตามความต้องการ) คลิกปุ่มเช็คบอกซ์ Create initial file และ Create folder ให้มีเครื่องหมายถูกด้านหน้าคำสั่งนั้น จากนั้นให้คลิกปุ่มที่คำสั่ง Next



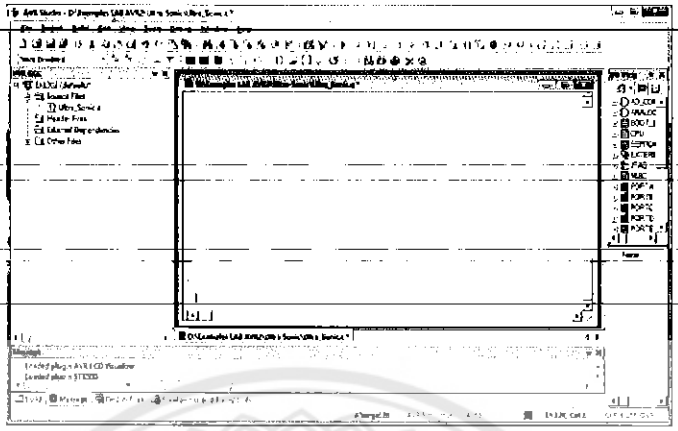
รูปที่ 3.9 AVR GCC พัฒนาโปรแกรมด้วยภาษา C

- หลังจากคลิกปุ่มคำสั่ง Next แล้วจะได้หน้าต่างกำหนดเครื่องมือในการดีบักโปรแกรม (Debug platform) พร้อมกำหนดเบอร์ AVR (Device) ที่ต้องการดังรูปที่ 3.10 จากโปรแกรมที่ใช้งานนี้เลือกไปที่ AVR Simulator และ ATmega48 (หากผู้ใช้งานมีเครื่องมือดีบักและเบอร์ที่ใช้งานแตกต่างกันจากที่เลือกให้ใช้เลือกตรงกับแบบที่ใช้งานแทน) จากนั้นคลิกปุ่มคำสั่ง Finish



รูปที่ 3.10 กำหนดเครื่องมือในการดีบักและเบอร์ AVR ที่ใช้งาน

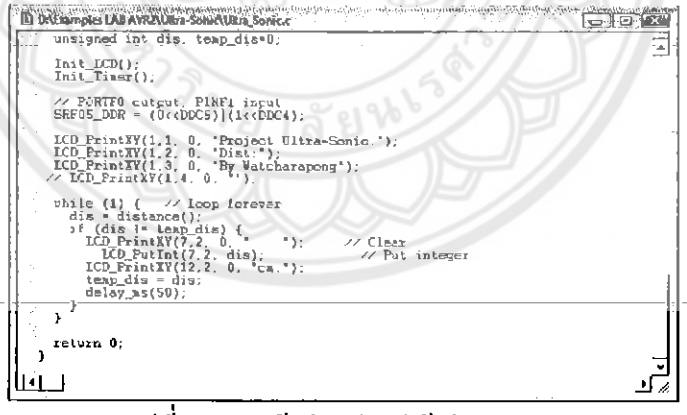
- เมื่อคลิกคำสั่งที่ปุ่ม Finish แล้วจะได้หน้าต่างสำหรับเขียนโค้ดโปรแกรม แสดงดังรูปที่ 3.11



รูปที่ 3.11 หน้าต่างเขียนโค้ดโปรแกรมและหน้าต่างคอมไพล์โปรแกรม

3.3.5 เขียนโค้ดโปรแกรม

หลังจากที่ได้หน้าต่างเขียนโค้ดโปรแกรม แล้วให้เขียนโค้ดโปรแกรมในการใช้งานโดยเขียนแล้วจะได้โค้ดโปรแกรมดังรูปที่ 3.12 โดยโปรแกรมที่เขียนจะทดสอบโปรแกรมผ่าน Simulation ของ AVR Studio แทนการเขียนโปรแกรมลงไปในชิบจริง



```
unsigned int dis, temp_dis=0;

Init_LCD();
Init_Timer();

// PORTF0 output, PINF1 input
SREG5_DDR = (0<<DDC5)|(1<<DDC4);

LCD_PrintXY(1,1, 0, "Project Ultra-Sonic ");
LCD_PrintXY(1,2, 0, "Dist:");
LCD_PrintXY(1,3, 0, "By Watcharapong");
// LCD_PrintXY(1,4, 0, " ");

while (1) {    // Loop forever
  dis = distance();
  if (dis != temp_dis) {
    LCD_PrintXY(7,2, 0, "    ");      // Clear
      LCD_PutInt(7,2, dis);           // Put integer
    LCD_PrintXY(12,2, 0, "cm.");
    temp_dis = dis;
    delay_ms(50);
  }
}

return 0;
}
```

รูปที่ 3.12 หน้าต่างเขียนโค้ดโปรแกรม

3.3.5 การคอมไพล์โค้ดโปรแกรม

การคอมไพล์โค้ดโปรแกรมให้เลือกไปที่เมนู Build หรือที่ไอคอนบาร์ดังรูปที่ 3.13 ใน AVR Studio สามารถเลือกได้ 3 รูปแบบคือ

1. การคอมไพล์โปรเจกต์ไฟล์ (Build Active Configuration หรือคีย์ F7) การคอมไพล์ลักษณะนี้จะคอมไพล์ที่อยู่ในโปรเจกต์ไฟล์ทั้งหมดพร้อมกับสร้างไฟล์ทำงาน หรือไฟล์นามสกุล .HEX

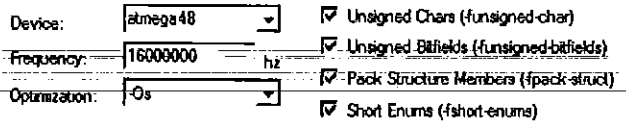
2. การคอมไพล์โปรเจกต์ไฟล์พร้อมทำงาน (Build and Run หรือคีย์ Ctrl + F7) การคอมไพล์ลักษณะนี้จะคอมไพล์ที่อยู่ในโปรเจกต์ไฟล์ทั้งหมดพร้อมกับสร้างไฟล์ทำงาน และเข้าสู่โหมดการดีบักโปรแกรม

3. การคอมไพล์ไฟล์ที่เปิดอยู่ (Compile Active file หรือคีย์ Alt + F7) การคอมไพล์ลักษณะนี้จะคอมไพล์เฉพาะไฟล์ที่เปิดทำงานอยู่ (หน้าต่างปัจจุบัน) เท่านั้น

รูปที่ 3.13 เมนูการคอมไพล์โปรแกรม

ในที่นี่จะแสดงการเขียนโปรแกรมที่ใช้งานกับ SRF05 และการโปรแกรมลงไปในชิบจริง โดยเลือกไปที่เมนู Project และเลือกรายการย่อย Configuration Option ดังรูปที่ 3.14 แล้วกำหนดค่าความถี่ Frequency เพื่อใช้ในการดีบัคโปรแกรมในที่นี้ใช้ Frequency = 16 MHz (16000000) ดังรูปที่ 3.15

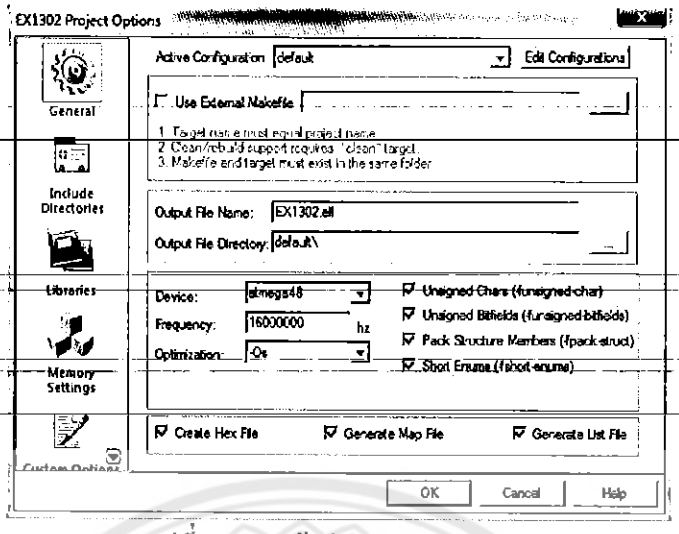
รูปที่ 3.14 เมนู Configuration Option



รูปที่ 3.15 การกำหนดค่าความถี่ (Frequency)

3.3.6 กำหนดรายละเอียดการคอมไพล์โปรเจกต์

การคอมไพล์โค้ดโปรแกรมสามารถที่จะกำหนดรายละเอียดและเงื่อนไขการคอมไพล์โค้ดได้โดยไปที่เมนู Project และเลือกรายการ Configuration Option จะได้หน้าต่างดังรูปที่ 3.16



รูปที่ 31.6 หน้าต่าง Project Option

โดยปกติเราจะใช้ค่าที่ AVR Studio กำหนดไว้แล้วเป็นค่าเบื้องต้น (Default) ส่วนที่ผู้เขียนแนะนำให้แก้มีรายละเอียดดังนี้

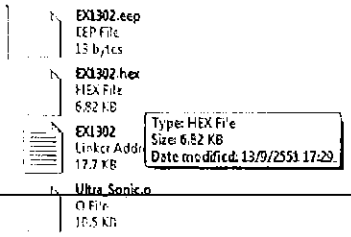
- กำหนดสัญญาณความถี่ (Frequency) ให้ตรงกับบอร์ดที่ใช้งาน โดยเฉพาะเมื่อทำงานในโหมดดีบักโปรแกรม (Debugging)

- การบีบอัดโค้ดโปรแกรม Hex ไฟล์ให้มีขนาดเล็ก (Optimization) โดยแบ่งออกเป็น 4 ระดับคือ -O1, -O2, -O3 และ –Os ให้เลือก -O3 ดีที่สุดทั้งระดับการบีบอัดโค้ดและความเร็วในการทำงาน (หากใช้ออปชันนี้แล้วเกิดปัญหากับโค้ดโปรแกรมให้ลดระดับการบีบอัดโปรแกรมลง) การใช้งานอปชันบีบอัดโปรแกรมจะทำให้โปรแกรมมีขนาดเล็กลง จะเห็นผลชัดเจน เมื่อมีการเรียกใช้งานฟังก์ชันไลบารีที่มาพร้อมกับ WinAVR

- เมื่อทำงานในโหมดดีบักแล้วไม่เกิดผลใดๆ ให้ยกเลิกการบีบอัดโปรแกรมกลับไปที่ -O0

- ข้อควรระวัง การคอมไพล์โค้ดโปรแกรม ต้องแน่ใจว่าเลือก Device ตรงกับเบอร์ที่ใช้งานจริง จากรูปก่อนหน้านี้ เลือก Device เป็น ATmega48 เนื่องจากไมโครคอนโทรลเลอร์ที่ใช้งานเป็นเบอร์ ATmega48 (หากไม่ตรงกับฮาร์ดแวร์แล้วนำไฟล์ที่ได้จากการคอมไพล์ไปใช้งาน อาจได้ผลลัพธ์ที่ผิดพลาดได้) กำหนดความถี่ (Frequency) ให้ตรงกับบอร์ดที่ใช้งาน แล้วก็กดคำสั่ง OK แล้วกลับสู่หน้าหลักรอการคอมไพล์ต่อไป

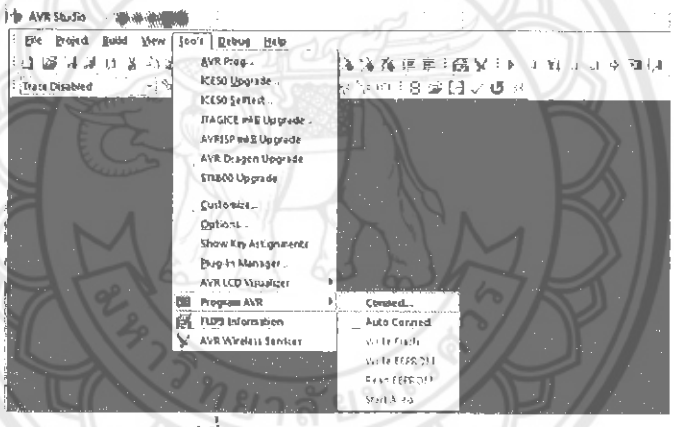
จากนั้นเลือกรายการ Build Active Configuration หรือกดคีย์ F7 โปรแกรม AVR Studio จะทำการดีบัคโปรแกรมถ้าคอมไพล์ผ่าน ก็จะไม่แสดงค่า error ขึ้นมา และจะได้ไฟล์นามสกุล .HEX ดังรูปที่ 3.17 เพื่อทำการโปรแกรมลงชิปในขั้นต่อไป



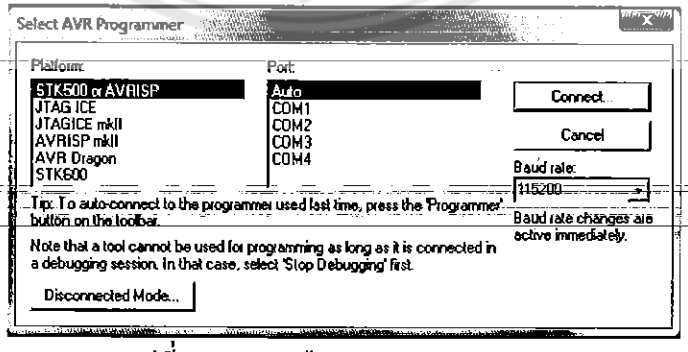
รูปที่ 3.17 แสดงไฟล์ที่ได้จากการคอมไพล์ .HEX

3.3.7 การโปรแกรมลงในชิบ

การโปรแกรมลงในชิบเราจะไปที่ Tools และเลือกคำสั่ง Program AVR และไปที่ Connect... ดังแสดงในรูปที่ 3.18 จากนั้นจะปรากฏหน้าต่างใหม่ขึ้นมาให้เลือก หัวข้อ Platform คลิกที่ STK500 or AVRISP และ ตรง Port ตั้งไว้ที่ Auto แสดงดังรูปที่ 3.19 หลังจากนั้นให้คลิก Connect...



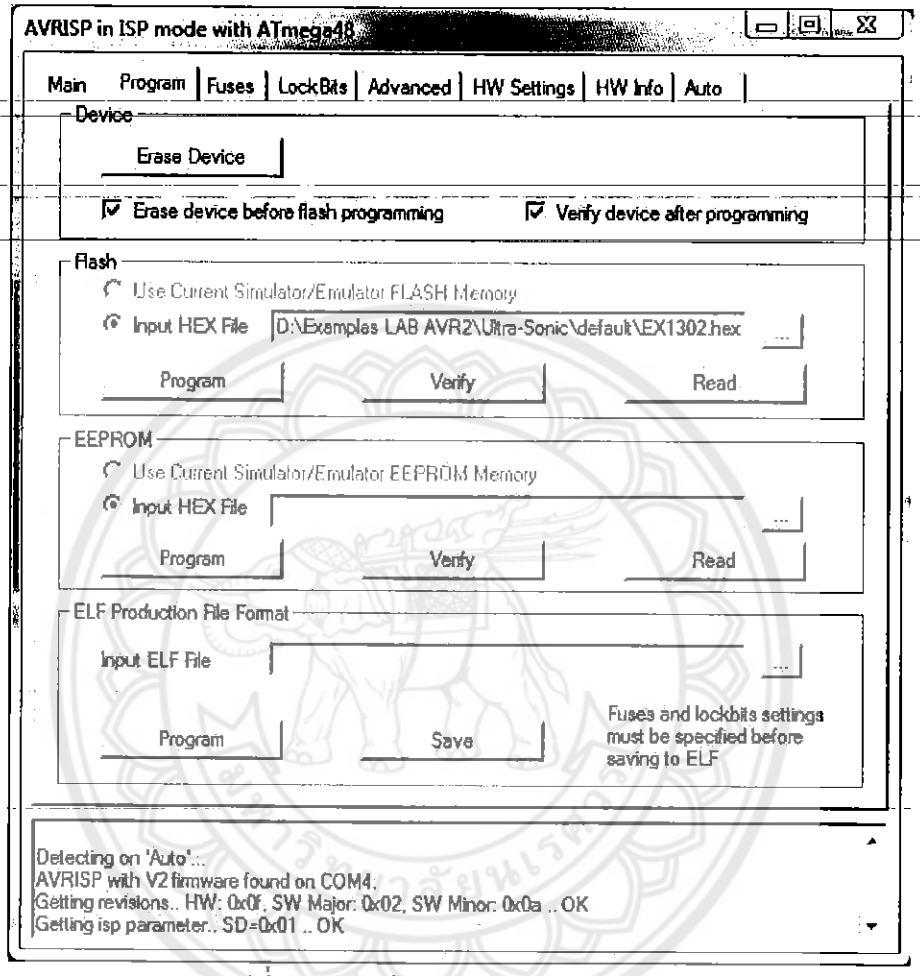
รูปที่ 3.18 เมนู Program AVR



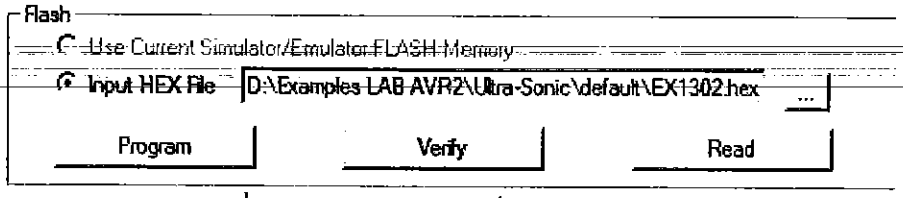
รูปที่ 3.19 การเลือก Platform และ Port

แล้วจะได้หน้าต่างใหม่ขึ้นมา AVRISP in ISP mode with ATmega48 โดยขึ้นหัวข้อ Program แสดงดังรูปที่ 3.20 ในนี้จะแบ่งอยู่เป็นสองส่วนหลักที่ใช้งานบ่อยๆ คือ Device คำสั่ง Erase Device คือการลบข้อมูลออกจากชิบ และ Flash จะมีกรอบที่ Input HEX File ก็จะทำการเลือก

ไฟล์นามสกุล .HEX ที่เราได้สร้างไว้แล้วจากการคอมไพล์โปรแกรม จากไดเรกทอรีที่ได้เก็บไฟล์ .HEX ไว้ แสดงดังรูปที่ 3.21 และคำสั่ง Program คือการโปรแกรมลงในชิบ ถ้าไม่เกิดข้อผิดพลาดใดๆ ก็จะบอกว่า Programming Successful…



รูปที่ 3.20 หัวข้อ Program และส่วนใช้งาน



รูปที่ 3.21 จากไดเรกทอรีที่ได้เก็บไฟล์ .HEX

3.4 อธิบายรหัสโปรแกรม Ultra_Sonic และ LIB_LCD.C

3.4.1 รายละเอียดของโปรแกรม Ultra-Sonic Ranger (SRF05)

บรรทัดที่ 21 ถึง 24 กำหนดรายละเอียดสำหรับใช้ในการเชื่อมต่อกับโมดูล SRF05 โดยกำหนดให้ขา PC5 ต่อเข้ากับขาสัญญาณ TRIGGER ใช้เป็นขาเอาท์พุต และขา PC4 ต่อกับขาสัญญาณ ECHO ใช้เป็นขาอินพุตรับสัญญาณ

บรรทัดที่ 37 ฟังก์ชันติดตั้งการใช้งานไมเมอร์/เคาน์เตอร์ 1 กำหนดให้ทำงานในโหมดไทเมอร์ พร้อมค่าของปรีสเกลเลอร์เท่ากับ 8

บรรทัดที่ 44 ฟังก์ชั่นการสร้างสัญญาณพลัลส์ตามเงื่อนไขที่โมดูล SRF05 ต้องการ คือ สัญญาณระหว่าง High และ Low ห่างกัน 10 ไมโครวินาที

บรรทัดที่ 53 ฟังก์ชันอ่านค่าระยะทางที่ได้จากโมดูล SRF05 โดยมีหลักการดังนี้

- บรรทัดที่ 57 และ 58 เคลียร์ขา PC4 หรือขาสัญญาณ PULSE จากนั้นสร้างสัญญาณพลัลส์ด้วยฟังก์ชัน trigger_pulse()
- บรรทัดที่ 60 วนลูปพร้อมกับอ่านค่าสัญญาณ ECHO หากมีค่าเป็นหนึ่ง แสดงว่าโมดูลเริ่มต้นทำงานแล้ว
- บรรทัดที่ 61 เคลียร์ค่าตัวนับ (TCNT1) เป็นศูนย์
- บรรทัดที่ 62 วนลูปพร้อมกับอ่านค่าสัญญาณ ECHO หากมีค่าเป็นศูนย์ แสดงว่าโมดูลทำการแปลงค่าสัญญาณเสร็จสิ้นแล้ว
- บรรทัดที่ 63 อ่านค่าการนับจากรีจีสเตอร์ TCNT1 กำหนดให้กับตัวแปร ct จากนั้นหน่วงเวลาอย่างน้อย 10 มิลลิวินาที
- บรรทัดที่ 66 ส่งค่าเป็นระยะทางในหน่วยเซนติเมตรกลับออกมาจากฟังก์ชัน

บรรทัดที่ 87 เรียกใช้งานฟังก์ชัน distance() เมื่ออ่านค่าระยะทางได้แล้วจะกำหนดให้กับตัวแปร dis นำค่าตัวแปร dis เปรียบเทียบกับตัวแปร temp_dis หากมีค่าไม่เท่ากันจะนำค่าระยะทางไปแสดงที่หน้าจอโมดูล LCD หากมีค่าเท่ากันจะข้ามส่วนการแสดงผลไป ด้วยหลักการนี้หากค่าไม่มีการเปลี่ยนแปลงก็ไม่ต้องเข้าส่วนแสดงผล

3.4.2 Header Interface "LIB_LCD.C"

Code ที่ใช้เป็น header file ที่ทำขึ้นเพื่อความสะดวกสะบายในการเขียนโปรแกรมส่วนหลักสามารถดึงส่วนที่เป็น header file นี้ไปใช้งาน

บทที่ 4

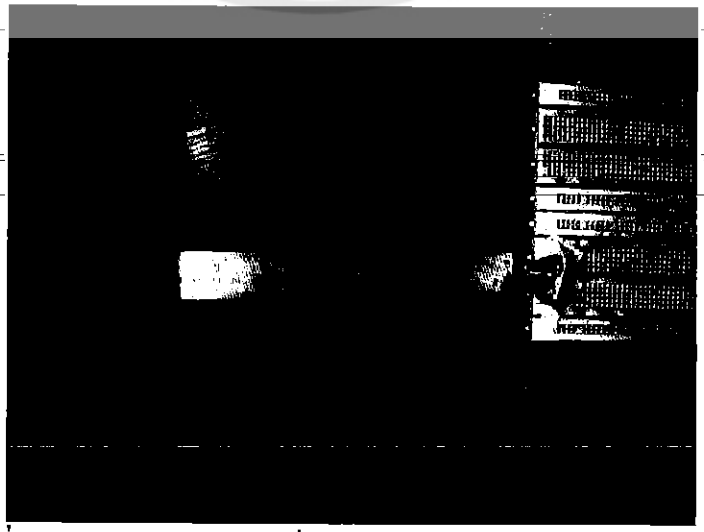
ผลการทดลอง

4.1 การทดลองวัดระยะทางด้วยอุลตร้าโซนิค

การทดลองวัดระยะทางด้วยอุลตร้าโซนิค จะทำการวัดและแสดงผลการทดลองเทียบกับตลับเมตร โดยทำการวัดระยะทางตั้งแต่ 1 เซนติเมตร จนถึง 4 เมตร จากนั้นทำการวัดโดยอุลตร้าโซนิค กับตลับเมตรดังแสดงในรูปที่ 4.1 และ รูปที่ 4.2

บันทึกค่าวัดระยะทางที่อ่านได้จากอุลตร้าโซนิค พร้อมทั้งหาค่าเปอร์เซนต์ความแตกต่างระหว่างระยะทางทั้ง 2 ค่า บันทึกลงในตารางที่ 4.1

รูปที่ 4.1 อ่านค่าวัดระยะทางที่วัดโดยตลับเมตรเทียบกับอุลตร้าโซนิค

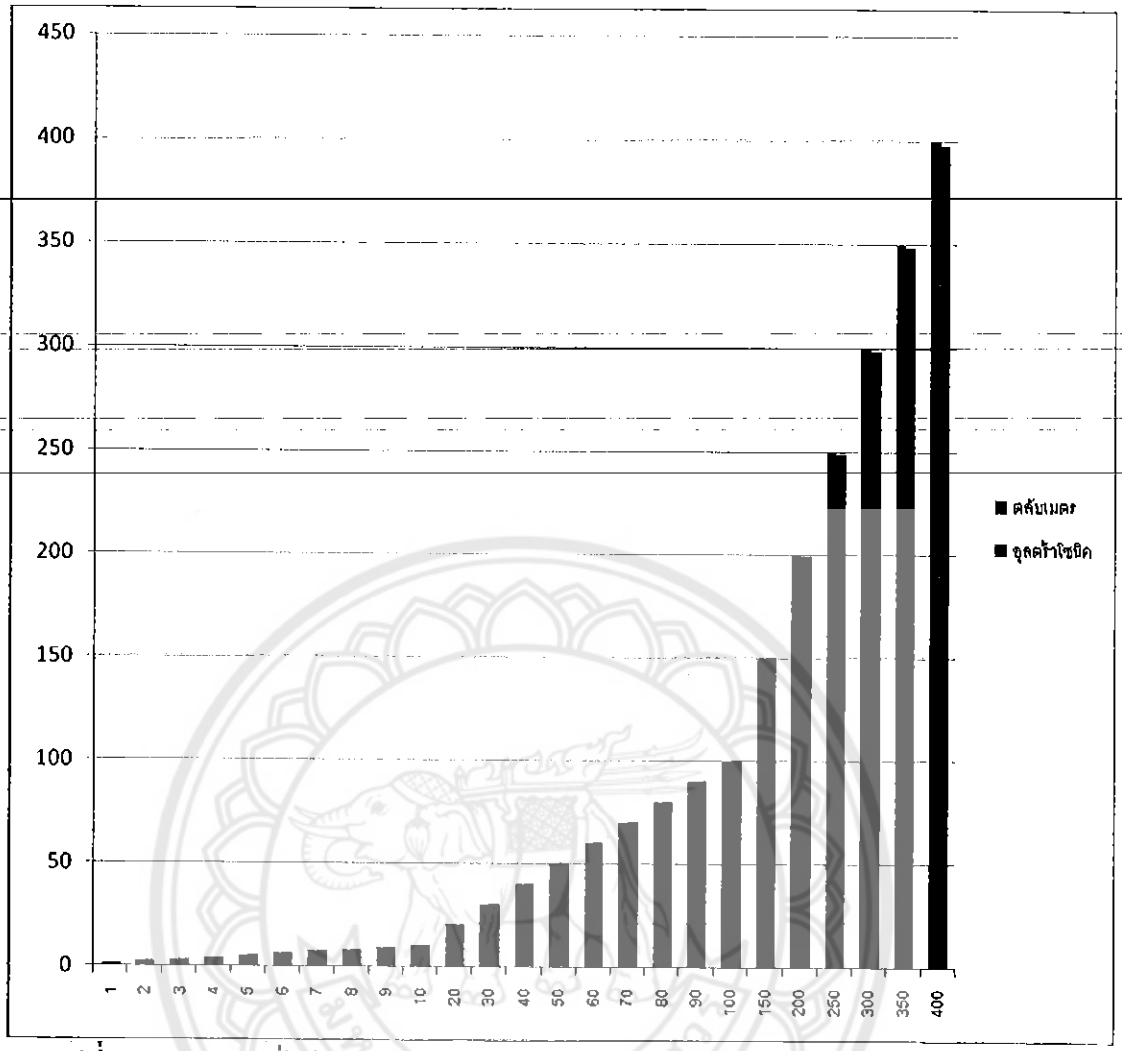


รูปที่ 4.2 อ่านค่าวัดระยะทางที่วัดโดยไม้บรรทัดเทียบกับอุลตร้าโซนิค

ตารางที่ 4.1 อ่านค่าวัดระยะทางที่วัด โดยตลับเมตรเทียบกับอุลตร้าโซนิค และค่าเปอร์เซนต์ความแตกต่าง

ตลับเมตร (cm.)	อุลตร้าโซนิค (cm.)	เปอร์เซนต์ความแตกต่าง (%)
1	1	0
2	2	0
3	3	0
4	4	0
5	5	0
6	6	0
7	7	0
8	8	0
9	9	0
10	10	0
20	20	0
30	30	0
40	40	0
50	50	0
60	60	0
70	70	0
80	80	0
90	90	0
100	100	0
150	149-150	0.33
200	199	0.5
250	248-249	0.6
300	297-299	0.66
350	347-349	0.57
400	396-398	0.75

จากข้อมูลตามตารางที่ 4.1 นำมาเขียนกราฟเปรียบเทียบระหว่างการวัดระยะทางโดยตลับเมตร และค่าที่วัดระยะทางได้จากอุตร้าโซนิค ได้ดังรูปที่ 4.3



รูปที่ 4.3 ระยะทางที่ได้จากการวัดตลับเมตรเทียบกับระยะทางการวัดที่ได้จากอุบตร้าโซนิค

บทที่ 5
สรุปผลและข้อเสนอแนะ

5.1 ผลการทดลอง

จากผลการทดลองสรุปได้ว่า ในการวัดระยะโดยระบบอุลตร้าโซนิค สามารถทำการวัดระยะได้แต่ช่วงการทำงานตั้งแต่ 1 เซนติเมตรถึง 4 เมตร พอวัดใกล้ หรือ ไกล กว่านั้นค่าหรือผลลัพทธ์ที่ได้ออกมาจะเพี้ยน และช่วงค่าที่วัดเปรียบเทียบกับสเกลที่วัดกับไม้บรรทัดและตลับเมตร ผลปรากฏว่า ค่าความผิดเพี้ยนที่สรุปในตาราง ช่วง 1 เซนติเมตรถึง 4 เมตร ค่าความผิดเพี้ยนอยู่ที่ 0.5683% ตามค่าการส่งสัญญาณในแต่ละระยะทาง ยิ่งส่งไกลค่าความกว้างของคลื่นก็มากขึ้นและการสูญเสียของสัญญาณก็มาก แต่ก็ไม่มีผลกระทบกับการวัดมากเท่าไหร่

5.2 ปัญหาการทดลอง

5.2.1 การเขียนโปรแกรมเพื่อทำการแสดงผลมีปัญหาเกี่ยวกับการแสดงผลออกทาง LCD และ แก้ register ที่บันทึกค่าเก็บไว้ จะต้องหา Library ที่ใช้งานกับ ไมโครคอนโทรลเลอร์เฉพาะรุ่น จึงต้องทำการสร้างเป็น Header file ที่เป็นไฟล์เก็บไว้ที่เดียวกับโปรเจกต์โปรแกรมที่เขียนไว้ เพื่อเวลาเขียนโปรแกรมสามารถดึงออกมาใช้ได้โดยไม่ต้องเขียนโปรแกรมเพื่อควบคุม LCD ซ้ำอีกในการเขียนครั้งต่อไป

5.2.2 การเปลี่ยนความถี่ คริสตรอลความถี่กับค่าปรีสเกลเลอร์ เพื่อให้ได้ค่าที่ต้องการตามคู่มือ SRF05 ต้องทำการคำนวณให้ได้ค่าที่ต้องการ คือ 0.5 ไมโครวินาที ที่ความถี่ 16 MHz กับค่าปรีสเกลเลอร์ 8 สามารถหารลงตัวได้ แต่เมื่อต้องการนำค่าออก RS232 เพื่อเชื่อมต่อกับคอมพิวเตอร์ ใช้ความถี่ 16 MHz เพื่อเซตค่า Baud rate 9600 ใส่ค่า USART UBRRn เพื่อให้ค่าส่งที่ดีที่สุด Optimization Baud rate ที่ UBRRn 103 ค่า error 0.2 % แต่ถ้าใช้ความที่ 7.3728 MHz จะใช้ UBRRn 47 ค่า error 0.0 % ไม่มีค่าความผิดพลาด แต่เมื่อนำมาคิดที่ทำการแสดงผลของ SRF05 จะหาค่าที่ต้องการ 0.5 ไมโครวินาที ต่อการประมวลผล 1 ครั้งไม่ได้ เพราะค่าที่ได้หารไม่ลงตัว 7.3728 หารค่าปรีสเกลเลอร์ไม่ลงตัว เพราะถ้านำมาคิดใน SRF05 จะทำให้ค่าที่ใช้ประมวลผลผิดเพี้ยนได้

5.2.3 การเขียนโปรแกรมเพื่อทำการ Simulation ต้องกำหนดสถานะของเบอร์ไมโครคอนโทรลเลอร์ให้ชัดเจนและตั้งค่า ให้ถูกต้องกับเบอร์ ปัญหาหลักของการเขียนโปรแกรมกำหนดค่าไม่ตรงกับเบอร์ที่ใช้งานและค่ารีจิสเตอร์ของแต่ละเบอร์เป็นค่าเฉพาะ เพื่อการทำงานที่ถูกต้อง

5.2.4 การออกแบบวงจรก็มีปัญหาแรงดันตกและแรงดันที่จ่ายให้กับ SRF05 ไม่คงที่ จึงต้องสร้างอีกแหล่งจ่ายหนึ่งเพื่อสร้างเป็น แหล่งจ่ายแรงดัน Pull up ดึงแรงดันที่จ่ายให้กับ SRF05 แรงดันไม่ตกและคงที่

5.2.5 การวัดที่มีระยะทางไกลๆ มีผลต่อการวัดที่ผิดเพี้ยน หากมีวัตถุมาบังการกระจายคลื่นออกไป ในมุมองศาที่ตกระทบของคลื่นทำให้ค่าออกมาไม่ตรงกับค่าจริงที่ใช้อุปกรณ์วัดได้

5.2.6 ค่าที่ระบุใน SRF05 ว่าหาร 114 เพื่อให้ได้ค่าที่ใกล้เคียงกับระยะทางมากที่สุดซึ่งในการเขียนโปรแกรมแล้วค่าได้มาอาจจะไม่ตรง ที่ต้องการ ในโครงงานนี้ต้องใช้ 114/16 เพื่อให้ได้ค่าเป็นอัตราส่วนในเทียบเป็นหน่วยเซนติเมตร

5.3 ข้อเสนอแนะ

5.3.1 การออกแบบวงจรควรทำการ Isolate เพื่อการลดสัญญาณรบกวนและ เพื่อความเป็นระเบียบสวยงาม และคำนึงถึงการทำงานแต่ละส่วนด้วย

5.3.2 จากโครงงานเรื่องนี้สามารถนำไปประยุกต์ใช้กับผู้ป่วยได้ เช่น เครื่องตรวจจับอุล ตร้าโซนิคใช้สำหรับผู้สูงวัยและผู้พิการ ในบ้านพักคนชรา หรือ โรงพยาบาล

5.3.3 จากโครงงานเรื่องนี้สามารถนำไปใช้กับการอำนวยความสะดวก เช่น การส่งสัญญาณเตือนเมื่อรถถอยหลังเข้าใกล้กำแพงมากเกินไป

5.3.4 จากโครงงานเรื่องนี้สามารถนำไปประยุกต์ใช้กับผู้พิการทางสายตา เช่น เครื่องวัดระยะทางสำหรับผู้พิการทางสายตาโดยระบบสั่นเตือนเมื่อเข้าใกล้วัตถุ

เอกสารอ้างอิง

[1] ประจิน พลังสันติกุล. การประยุกต์ใช้งานภาษา C สำหรับไมโครคอนโทรลเลอร์ AVR เล่ม 1. กรุงเทพมหานคร : 2551

[2] ประจิน พลังสันติกุล. การประยุกต์ใช้งานภาษา C สำหรับไมโครคอนโทรลเลอร์ AVR เล่ม 2. กรุงเทพมหานคร : 2551

[3] Data Sheet (SRF05). "Ultra-Sonic Ranger." [Online], เข้าถึงได้จาก : http://hendawan.files.wordpress.com/2008/07/srf05.pdf. 2551

[4] Document Word UltraSonic. "ระบบอุลตร้าโซนิค." [Online], เข้าได้ถึงจาก : http://student.nu.ac.th/electronic/Part01/Ultrasonic.doc. 2551

ภาคผนวก ก

ตารางการใช้ค่าปรีสเกลเลอร์ในไมโครคอนโทรลเลอร ATmega48

ตาราง ก Clock Select Bit Description

CS02	CS01	CS00	Description
0	0	0	No clock source (Timer/Counter stopped)
0	0	1	clkI/O/(No prescaling)
0	1	0	clkI/O/8 (From prescaler)
0	1	1	clkI/O/64 (From prescaler)
1	0	0	clkI/O/256 (From prescaler)
1	0	1	clkI/O/1024 (From prescaler
1	1	0	External clock source on T0 pin. Clock on falling edge.
1	1	1	External clock source on T0 pin. Clock on rising edge.

If external pin modes are used for the Timer/Counter0, transitions on the T0 pin will clock the counter even if the pin is configured as an output. This feature allows software control of the counting.

ภาคผนวก ข

รหัสต้นฉบับของการควบคุม LCD Header file "LIB_LCD.C"

```
/************************************************************************
*
Author:                 prajin palangsantikul
Company:                appsofttech co.,ltd.
Filename:               LIB_LCD.c
Purpose:                LCD Character 16x2 Module 4bit Mode
Ref..:                  TG_4bitLCD.c [Thorsten Godau, thorsten.godau@gmx.de]
Date:                   15/10/2005
************************************************************************
*/

/************************************************************** Defines */

/*
* LCD PIN
* PORT Control : Low Nibble
* PORT Data : High Nibble
*/

#include <stdlib.h>

#ifdef LCD_PORTD
#define LCD_DATA    PORTD     // Dataport of LCD-Display (D4..D7)
#define LCD_OUT         DDRD                    // direction port
#define LCD_RS      PD2       // Register Select of LCD-Display
#define LCD_E       PD3       // Enable of LCD-Display
//#define LCD_RW              PC3                     // Read/Write LCD-Display
#else
```

```
#define LCD_DATA    PORTB      // Dataport of LCD-Display (D4..D7)
#define LCD_OUT          DDRB                    // direction port
#define LCD_RS      PB0        // Register Select of LCD-Display
#define LCD_E       PB2        // Enable of LCD-Display
//#define LCD_RW    PB3                         // Read/Write LCD-Display
#endif


#define BLINK    0x01                // Alias for blinking cursor
#define NOBLINK     0x00             // Alias for non blinking cursor
#define SHOW        0x02             // Alias for cursor on
#define HIDE        0x00             // Alias for cursor off
#define ON          0x04             // Alias for display on
#define OFF         0x00             // Alias for display off
#define CLS         0x01          // Clear LCD


#define SET_LCD_RS          sbi(LCD_DATA,LCD_RS)        // Select data register
#define CLS_LCD_RS          cbi(LCD_DATA,LCD_RS)        // Select instruction register


#define SET_LCD_E    sbi(LCD_DATA,LCD_E)                // Enable of LCD-Display
#define CLS_LCD_E    cbi(LCD_DATA,LCD_E)                // Disable of LCD-Display
//#define READ_LCD_RW               sbi(LCD_DATA,LCD_RW)       // Read Mode
//#define WRITE_LCD_RW     cbi(LCD_DATA,LCD_RW)        // Write Mode


/*
* Table to select DD-RAM address (including instruction and address)
* 0x00..0x0F -> Line 1, 0x40..0x4F -> Line 2
*/
static unsigned char LOCATION[4] [20] = { {0x80,0x81,0x82,0x83,0x84,0x85,0x86,0x87,
                                0x88,0x89,0x8A,0x8B,0x8C,0x8D,0x8E,0x8F,
                                        0x90,0x91,0x92,0x93},
                                        {0xC0,0xC1,0xC2,0xC3,0xC4,0xC5,0xC6,0xC7,
                                0xC8,0xC9,0xCA,0xCB,0xCC,0xCD,0xCE,0xCF,
```

```
                                        0xD0,0xD1,0xD2,0xD3},
                                        {0x94,0x95,0x96,0x97,0x98,0x99,0x9A,0x9B,
                0x9C,0x9D,0x9E,0x9F,0xA0,0xA1,0xA2,0xA3,
                                        0xA4,0xA5,0xA6,0xA7},
                                        {0xD4,0xD5,0xD6,0xD7,0xD8,0xD9,0xDA,0xDB,
                0xDC,0xDD,0xDE,0xDF,0xE0,0xE1,0xE2,0xE3,
                                        0xE4,0xE5,0xE6,0xE7} };

/*static unsigned char LOCATION[2] [20] = { {0x80,0x81,0x82,0x83,0x84,0x85,0x86,0x87,
                0x88,0x89,0x8A,0x8B,0x8C,0x8D,0x8E,0x8F,
                                        0x90,0x91,0x92,0x93},
                                        {0xC0,0xC1,0xC2,0xC3,0xC4,0xC5,0xC6,0xC7,
                0xC8,0xC9,0xCA,0xCB,0xCC,0xCD,0xCE,0xCF,
                                        0xD0,0xD1,0xD2,0xD3} };
*/

/** Timer dependend LCD_Delay-routine */
void LCD_Delay(unsigned int ms);
/**  Writes a character to display */
void LCD_PutChar(unsigned char value);
/**  Writes a string to display */
void LCD_PutStr(unsigned int dly, unsigned char *text);
/**  Command LCD */
void LCD_Command(unsigned char cmd);
/**  Prints a text to x/y position */
void LCD_PrintXY(unsigned char x,unsigned char y, unsigned int dly, unsigned char *text);
/** Controls the display */
void LCD_Control(unsigned char dsp,unsigned char blink,unsigned char cursor);
/** Sets LCD write position */
void LCD_GotoXY(unsigned char x,unsigned char y);
/** Initialize LCD Module */
void Init_LCD(void);
```

```
/************************************************************** LCD Delay */
/*
* Function : LCD_Delay
* Parameters : i - unsigned int
* Returned : nothing
*/
void LCD_Delay(unsigned int i)
{
 for (;i > 0; i--)
   _delay_ms(1);
}

/************************************************************** LCD_Control */
/*
* Function    : LCD_Control(dsp,blink,cursor)
* Input : unsigned char dsp    = ON     -> Display on
*                       OFF    -> Display off
*       unsigned char blink  = BLINK  -> Cursor blinks
*                       NOBLINK-> Cursor not blinks
*       unsigned char cursor = SHOW   -> Cursor visible
*                       HIDE   -> Cursor invisible
*
*/
void LCD_Control(unsigned char dsp,unsigned char blink,unsigned char cursor)
{
 unsigned char control;              // variable to generate instruction byte

 control = (0x08 + blink + cursor + dsp); // Cursor control
 LCD_Command(control);
 return;
```

```
}

/************************************************************** LCD_GotoXY
*/
/*
* Function    : LCD_GotoXY(x,y)
* Description : Sets the write position of the LCD display
*
*             (1,1)       (16,1)
*               |           |
*               ################  -> line 1
*               ################  -> line 2
*               |           |
*             (1,2)       (16,2)
* Input       : unsigned char x    -> x position (horizontal)
*             unsigned char y    -> y position (vertical)
*/
void LCD_GotoXY(unsigned char x,unsigned char y)
{
 LCD_Delay(20);          // Wait 20ms
 CLS_LCD_E;                          // Disable
 CLS_LCD_RS;           // Switch to inruction register
//     WRITE_LCD;

 /** Set LCD_DATA to high nibble of position table value */
 LCD_DATA = (LCD_DATA&0x0F)|((LOCATION[y-1][x-1])&0xF0);
 SET_LCD_E;  CLS_LCD_E;   // Write data to display
 LCD_Delay(1);                    // Wait 1ms
 /** Set LCD_DATA to lower nibble of position table value */
 LCD_DATA = (LCD_DATA&0x0F)|(((LOCATION[y-1][x-1])<<4)&0xF0);
 SET_LCD_E;  CLS_LCD_E;   // Write data to display
```

```
 LCD_Delay(1);          // Wait 1ms
 return;
}

/************************************************************** LCD_PrintXY
*/
/*
* Function : LCD_PrintXY(x,y,*text)
* Description : Prints text to position x/y of the display
* Input      : unsigned char x     -> x position of the display
*              unsigned char y     -> y position of the display
*              unsigned char *text -> pointer to text
*                                     unsigned int dly        -> LCD_Delay time
*/
void LCD_PrintXY(unsigned char x,unsigned char y, unsigned int dly, unsigned char *text)
{
 LCD_GotoXY(x,y);         // Set cursor position
 while ( *text ) {        // while not end of text
  LCD_PutChar(*text++);   // Write character and increment position
  LCD_Delay(dly);                    // time LCD_Delay print text
 }
 return;
}

/************************************************************** LCD_PutChar
*/
/*
* Function    : LCD_PutChar(value)
* Description : Writes the character value to the display
*/
void LCD_PutChar(unsigned char value)
{
```

```
  CLS_LCD_E;                          // Disable
  SET_LCD_RS;          // Switch to data register

  /** Set LCD_DATA to high nibble of value */
  LCD_DATA = (LCD_DATA&0x0F)|(value&0xF0);
  SET_LCD_E;  CLS_LCD_E;  // Write data to display
        LCD_Delay(1);                 // Wait 1ms
  /** Set LCD_DATA to lower nibble of value */
  LCD_DATA = (LCD_DATA&0x0F)|((value<<4)&0xF0);
  SET_LCD_E;  CLS_LCD_E;  // Write data to display

  LCD_Delay(1);                       // Wait 1ms
  return;
}

/*********************************************************************** LCD_PutStr */
/*
* Function    : LCD_PutStr(value)
* Description : Writes the string value to the display
*/
void LCD_PutStr(unsigned int dly, unsigned char *text)
{
  while (*text) {          //while not end of text
    LCD_PutChar(*text++);     // Write character and increment position
    LCD_Delay(dly);                  // time LCD_Delay print text
  }
  return;
}

/*********************************************************************** LCD_PutStr */
/*
* Function    : LCD_PutInt(x, y, value)
```

```
* Description : Writes the value to the display
*/
void LCD_PutInt(unsigned char x,unsigned char y,unsigned int num)
{
 unsigned char buf[10];

 sprintf(buf,"%d",num);

 LCD_GotoXY(x,y);        // Set cursor position
 LCD_PutStr(0,buf);

 return;
}

/************************************************************** LCD_Command
*/
void LCD_Command(unsigned char cmd)
{
 LCD_Delay(20);          // Wait 20ms
 CLS_LCD_E;                      // Disable
 CLS_LCD_RS;          // Switch to inruction register

 /** Set LCD_DATA to high nibble of Display On/Off Control */
 LCD_DATA = (LCD_DATA&0x0F)|(cmd&0xF0);
 SET_LCD_E; CLS_LCD_E;  // Write data to display
 LCD_Delay(1);               // Wait 1ms
 /** Set LCD_DATA to lower nibble of Display On/Off Control */
 LCD_DATA = (LCD_DATA&0x0F)|((cmd<<4)&0xF0);
 SET_LCD_E;  CLS_LCD_E;  // Write data to display

 LCD_Delay(1);               // Wait 1ms
 return;
```

```
}

/************************************************************************ Init_LCD
*/
// Initialize LCD Module
void Init_LCD(void)
{
 LCD_OUT = 0xFF;      // Set output port

 /** Set LCD Module 4 bit Mode */
 LCD_Command(0x33);          // set 4 bit mode
 LCD_Command(0x32);          // set 4 bit mode
 LCD_Command(0x28);          // 2 line 5x7 dot
 LCD_Command(0x01);          // clear screen lcd

 // Display on, cursor hidden and no blinking
 LCD_Control(ON,NOBLINK,HIDE);
}
```

รหัสต้นฉบับของการควบคุม Ultra_Sonic.C

```
01: /*-------------------------------------
02: File       : Ultra-Sonic Ranger (SRF05)
03: Purpose    : Ultrasonic Distance Detector
04: Compiler   : AVR Studio/WinAVR
05: Target     : ATmega48
06: -------------------------------------*/
07:
08: //-------------------------------------:INCLUDE
09:
10: #include <avr/io.h>          // AVR device-specific IO
definitions
11: #include <avr/interrupt.h>  // AVR interrupt
12: #include <compat/deprecated.h>  // Use sbi(), cbi() function
13:
14: #define F_CPU 16000000UL     // XTAL 16 MHz
15: #include <util/delay.h>      // header file implement simple
delay 16: loops
17:
18: #define LCD_PORTD            // Use PORTC for LCD Port
19: #include "LIB_LCD.C"         // LCD Library
20:
21: #define SRF05_DDR     DDRC
22: #define PULSE_PORT    PORTC
23: #define PULSE_PIN     PINC4
24: #define ECHO_PIN      (PINC&(1<<PINC5))
25:
26:
27: //-------------------------------------:FUNCTION
28:
29: // delay_ms
30: void delay_ms(unsigned int i)
31:{
32:  for (; i>0; i--)
33:    _delay_ms(1);
34:}
35:
36: // Init Timer
37: void Init_Timer(void)
38: {
39:   // Timer/Counter1: Prescaler 8
40:   TCCR1B = (0<<CS12)|(1<<CS11)|(0<<CS10);
41: }
42:
43: // Trigger pulse for start process
44: void trigger_pulse(void)
45: {
46:   sbi(PULSE_PORT,PULSE_PIN);
47:  _delay_us(10);
49:   cbi(PULSE_PORT,PULSE_PIN);
50: }
51:
52: // distance reading
53: unsigned int distance(void)
54: {
55:   unsigned int ct;
56:
```

```
57:    cbi(PULSE_PORT,PULSE_PIN);
58:    trigger_pulse();
59:
60:    while(!ECHO_PIN);      // Wait for start
61:    TCNT1 = 0;             // Clear Timer/Counter1 register
62:    while(ECHO_PIN);       // Wait for count
63:    ct = TCNT1;
64:
65:    _delay_ms(10);
66:    return (ct/(116/16));   //114
67: }
68:
69: //---------------------------------------:MAIN
70:
71: int main(void)
72: {
73:    unsigned int dis, temp_dis=0;
74:
75:    Init_LCD();
76:    Init_Timer();
77:
78: // PORTF0 output, PINF1 input
79: SRF05_DDR = (0<<DDC5)|(1<<DDC4);
80:
81: LCD_PrintXY(1,1, 0, "Project Ultra-Sonic.");
82: LCD_PrintXY(1,2, 0, "Dist:");
83: LCD_PrintXY(1,3, 0, "By Watcharapong");
84: // LCD_PrintXY(1,4, 0, "");
85:
86: while (1) {  // Loop forever
87:  dis = distance();
88:    if (dis != temp_dis) {
89:      LCD_PrintXY(7,2, 0, "    ");      // Clear
90:       LCD_PutInt(7,2, dis);             // Put integer
91:      LCD_PrintXY(12,2, 0, "cm.");
92:     temp_dis = dis;
93:      delay_ms(50);
94:    }
95: }
96:
97: return 0;
98: }
```

ประวัติผู้เขียนโครงงาน

ชื่อ นายวัชรพงค์ เชื้อสะอาด

ภูมิลำเนา 56 ม.4 ต.สันกำแพง อ.สันกำแพง จ.เชียงใหม่ 50130

169 ม.1 ต.หย่วน อ.เชียงคำ จ.พะเยา 56110

ประวัติการศึกษา

- จบระดับมัธยมศึกษาต้นโรงเรียนปรินส์รอยแยลส์วิทยาลัย
- จบระดับมัธยมศึกษาปลายโรงเรียนเชียงคำวิทยาคม
- ปัจจุบันกำลังศึกษาในระดับปริญญาตรี ชั้นปีที่ 4

สาขาวิชาวิศวกรรมไฟฟ้า คณะวิศวกรรมศาสตร์

มหาวิทยาลัยนเรศวร

E-mail : **takeachi@hotmail.com**