ระบบตรวจวัดอุณหภูมิอัตโนมัติโดยผ่านเครือข่าย xbee โดยมีแหล่งจ่ายเป็นโซล่าเซลล์

โดย

นาย สุภนัย กระจายศรี B5402458

นาย ศณัฐขะพล อัตตวนิช B5424733

นาย กฤษฎา กลีบจำปา B5428212

รายงานนี้เป็นส่วนหนึ่งของรายวิชา 527499 โครงงานวิศวกรรมโทรคมนาคม

หลักสูตรวิศวกรรมศาสตรบัณฑิต สาขาวิชาวิศวกรรมโทรคมนาคม

หลักสูตรปรับปรุง พ.ศ. 2554

สำนักวิชาวิศวกรรมศาสตร์ มหาวิทยาลัยเทคโนโลยีสุรนารี

ประจำภาคการศึกษาที่ 1 ปีการศึกษา 2557

เรื่อง เครือข่ายเซนเซอร์ไร้สายเพื่อเฝ้าระวังนาข้าวโดยใช้เว็บบราวเซอร์

(Wireless Sensor Network for Monitoring Paddy Fields using Web Browser)

คณะกรรมการสอบโครงงาน

(ผู้ช่วยศาสตราจารย์ ดร. วิภาวี หัตถกรรม)

อาจารย์ที่ปรึกษาโครงงาน

(ผู้ช่วยศาสตราจารย์ ร.อ.ดร.ประโยชน์ คำสวัสดิ์)

กรรมการ

(อาจารย์ เศรษฐวิทย์ ภูฉายา)

กรรมการ

มหาวิทยาลัยเทคโนโลยีสุรนารี อนุมัติให้นับรายงานโครงงานฉบับนี้ เป็นส่วนหนึ่งของการศึกษาระดับปริญญาตรี สาขาวิชาวิศวกรรมโทรคมนาคม วิชา 527499 โครงงานวิศวกรรมโทรคมนาคมประจำปีการศึกษา 2558

โครงงาน	ระบบตรวจวัดอุณหภูมิอัตโนมัติโดยผ่านเครือข่าย xbee วัดอุณหภูมิโดยมีแหล่งจ่ายเป็นโซล่าเซลล์	
จัดทำโดย	นาย สุภนัย กระจายศรี	B5402458
	นาย ศณัฐขะพล อัตตวนิช	B5424733
	นาย กฤษฎา กลีบจำปา	B5428212
อาจารย์ที่ปรึกษา	ผู้ช่วยศาสตราจารย์ ดร. ชุติมา พรหมมาก	
สาขาวิชา	วิศวกรรมโทรคมนาคม	
ภาคการศึกษา	1/2557	

บทคัดย่อ
(Abstract)

โครงงานนี้เป็นการศึกษาโครงสร้างของระบบตรวจวัดอุณหภูมิอัตโนมัติโดยผ่านเครือข่าย xbee วัดอุณหภูมิโดยมีแหล่งจ่ายเป็นโซล่าเซลล์โดยเป็นการประยุกต์บอร์ดไมโครคอนโทรเลอร์ Arduino UNO R3 กับ sensor วัดอุณหภูมิ DS18B20 โดยใช้ภาษา C เป็นตัวสั่งการ โดยที่จะมีการใช้งาน 2 ระบบ คือจากภาคส่งและภาครับ โดยที่ภาคส่งจะทำการวัดอุณหภูมิโดยใช้วงจรเซนเซอร์ต่อกับบอร์ด arduino Uno R3 และใช้การส่งไร้สายจาก xbee และใช้แหล่งจ่ายจากโซล่าเซลล์ และส่งค่าข้อมูลเซนเซอร์ไปให้ภาครับ โดยที่ภาครับก็รับค่าจากการส่งไร้สายจาก Xbee ที่ต่อกับ arduino ด้วยเช่นกัน จากนั้นจะแสดงผลค่าอุณหภูมิที่จอ LCD และถ้าอุณหภูมิเกินค่าที่เราได้กำหนดไว้ จะเกิดการแจ้งเตือนของวงจรเสียง จากการทดสอบพบว่าปัจจุบันการใช้พลังงานไฟฟ้ามีมากขึ้น ทำให้สิ้นเปลืองพลังงาน จึงได้พลังงานทดแทนจากโซล่าเซลล์ เพื่อทำให้ไม่มีการสิ้นเปลืองพลังงาน เพราะพลังงานจากโซล่าเซลล์สามารถชาร์จเข้าแบตเตอรี่ได้ และทำให้วงจรภาคส่งสามารถทำงานตลอดเวลา

กิตติกรรมประกาศ
(Acknowledgement)

โครงงานเรื่องระบบตรวจวัดอุณหภูมิอัตโนมัติโดยผ่านเครือข่าย xbee วัดอุณหภูมิโดยมีแหล่งจ่ายเป็นโซล่าเซลล์สามารถสำเร็จลุล่วงไปได้ด้วยดีเนื่องจากคณะผู้จัดทำโครงงานได้รับความช่วยเหลือด้านต่างๆจากอาจารย์ที่ปรึกษา ผ.ศ. ดร. ชุติมา พรหมมาก ที่ได้ให้ความช่วยเหลือและให้คำปรึกษาในทุกๆ ด้านแก่คณะผู้จัดทำมาโดยตลอด และ ขอกราบขอบพระคุณ อาจารย์ ดร.ธนเสฏฐ์ ทศดีกรพัฒน์ ที่ช่วยให้คำปรึกษาและคำแนะนำเพิ่มเติมในการทำโครงงานครั้งนี้แก่คณะผู้จัดทำ และขอกราบขอบพระคุณคณาจารย์ บุคลากร และนักศึกษาบัณฑิตศึกษาสาขาวิชาวิศวกรรมโทรคมนาคมทุกท่านที่ให้ความช่วยเหลือแก่คณะผู้จัดทำ และสุดท้ายนี้คณะผู้จัดทำโครงงานขอขอบพระคุณบิดามารดา ที่ท่านทั้งสองให้การดูแลเอาใจใส่คอยให้กำลังใจและอยู่เคียงข้างมาโดยตลอด จนกระทั่งโครงงานนี้สำเร็จลุล่วงไปด้วยดี

คณะผู้จัดทำโครงงานใคร่ขอกราบขอบพระคุณทุกๆ ท่านที่ได้กล่าวไปแล้วไว้ ณ ที่นี้ สำหรับส่วนดีของโครงงานชิ้นนี้ ขออุทิศให้แก่คณาจารย์ทุกท่านที่ได้ประสิทธิ์ประสาทวิชาความรู้ให้แก่คณะผู้จัดทำโครงงาน หากโครงงานชิ้นดีมีข้อผิดพลาดประการใดทางคณะผู้จัดทำโครงงานใคร่ขอน้อมรับและขออภัยมา ณ ที่นี้

นาย สุภนัย กระจายศรี
นาย ศณัฐขะพล อัตตวนิช
นาย กฤษฎา กลีบจำปา

สารบัญ

เรื่อง	หน้า
บทคัดย่อ	ก
กิตติกรรมประกาศ	ข
สารบัญ	ค
สารบัญรูป	ฉ
สารบัญตาราง	ฎ
บทที่ 1 บทนำ	
1.1 บทนำ	1
1.2 หลักการและเหตุผล	1
1.3 วัตถุประสงค์	2
1.4 ขอบเขตงาน	2
1.5 ขั้นตอนการดำเนินงาน	3
1.6 ผลที่คาดว่าจะได้รับ	3
บทที่ 2 หลักการและทฤษฎีที่เกี่ยวข้อง	
2.1 บทนำ	4
2.2 ความร้อน และอุณหภูมิ	4
2.3 โซล่าเซลล์	7
2.3.1 ประวัติความเป็นมาของโซล่าเซลล์	7
2.3.2 ชนิดของโซล่าเซลล์	7
2.3.3 หลักการทำงานทั่วไปของโซล่าเซลล์	9
2.3.1 ลักษณะเด่นของโซล่าเซลล์	9
2.3.5 คุณสมบัติและตัวแปรที่สำคัญของโซล่าเซลล์	10
2.3.6 อุปกรณ์สำคัญของระบบการผลิตกระแสไฟฟ้าจากโซล่าเซลล์	11
2.3.7 การประยุกต์ใช้งานโซล่าเซลล์ในด้านต่างๆ	12
2.4 เซนเซอร์ตรวจวัดอุณหภูมิ (Temperature Sensor)	13
2.4.1 โมดูลเซ็นเซอร์วัดอุณหภูมิ	14
2.4.2 การใช้งานเซ็นเซอร์วัดอุณหภูมิ	14
2.4.3 บอร์ดไมโครคอนโทรเลอร์ Arduino	16
2.4.4 ความแตกต่างของ Arduino กับ ไมโครคอนโทรลเลอร์อื่นๆ	17
2.5 Zigbee and Xbee BASIC ตอน การใช้งาน Xbee เบื้องต้น	34
บทที่ 3 หลักการทำงานของอุปกรณ์และโปรแกรม ในการใช้งานระบบตรวจวัดอุณหภูมิอัตโนมัติโดยผ่านเครือข่าย xbee โดยมีแหล่งจ่ายเป็นโซล่าเซลล์	
3.1 บทนำ	43
3.2 โครงสร้างของระบบตรวจวัดอุณหภูมิอัตโนมัติโดยผ่านเครือข่ายxbee โดยมีแหล่งจ่ายเป็นโซล่าเซลล์	43

สารบัญ (ต่อ)

เรื่อง **หน้า**

3.3 อุปกรณ์ที่ใช้ในระบบตรวจวัดอุณหภูมิอัตโนมัติโดยผ่านเครือข่าย xbeeโดยมีแหล่งจ่ายเป็นโซล่าเซลล์ 46
3.3.1 เซนเซอร์วัดอุณหหูมิ DS18B20 46
3.3.2 บอร์ดไมโครคอนโทรลเลอร์ Arduino รุ่น Arduino UNO R3 50
3.3.3 จอแสดงผล LCD 60
3.3.4 วงจรเสียงแจ้งเตือน 64
3.3.5 โซล่าเซลล์ 64
3.3.6 Xbee 67
3.4 การเขียนโปรแกรมของระบบตรวจวัดอุณหภูมิอัตโนมัติโดยผ่านเครือข่าย Xbee โดยมีแหล่งจ่ายเป็นโซล่าเซลล์ 68
3.4.1 แผนผังการทำงานของโปรแกรม (Flow chart) 68
3.4.2 Code program และคำอธิบาย 74
3.4.3 การเขียนโปรแกรม Arduino IDE 77
3.4.4 การโหลดโปรแกรมลงบอร์ด 79

บทที่ 4 การทดสอบระบบตรวจวัดอุณหภูมิอัตโนมัติโดยผ่านเครือข่าย xbee โดยมีแหล่งจ่ายเป็นโซล่าเซลล์

4.1 บทนำ 85
4.2 การเชื่อมต่อวงจรของระบบในการทดลอง 85
4.2.1 เชื่อมต่อxbee เข้ากับคอมพิวเตอร์ 85
4.2.2 เชื่อมต่อวงจรภาคส่ง 88
4.2.3 เชื่อมต่อวงจรภาครับ 89
4.2.4 เชื่อมต่อวงจรภาครับกับภาคส่ง 90
4.3 ผลการทดสอบโครงสร้างของระบบตรวจวัดอุณหภูมิอัตโนมัติโดยผ่านเครือข่าย Xbee โดยมีแหล่งจ่ายเป็นโซล่าเซลล์ 91
4.3.1 การทดสอบการทำงานของวงจรแจ้งเตือนระบบตรวจวัดอุณหภูมิ 91
4.3.2 การทดสอบกรณีที่ 1 non line-of-sight 93
4.3.3 การทดสอบกรณีที่ 2 line-of-sight 97
4.3.4 ทดสอบการทำงานของเซนเซอร์วัดอุณหภูมิของกรณี non line-of-sight 99
4.3.5 ทดสอบการทำงานของเซนเซอร์วัดอุณหภูมิของกรณี line-of-sight 101
4.3.6 ทดสอบการทำงานของโซล่าเซลล์ 103
4.4 วิเคราะห์ผลการทดลอง 105
4.4.1 วิเคราะห์การทดสอบโครงสร้างระบบการวัดอุณหภูมิโดยผ่านเครือข่ายไร้สาย ZigBee 104

สารบัญ (ต่อ)

เรื่อง	หน้า
4.4.2 วิเคราะห์การทำงานของเซนเซอร์วัดอุณหภูมิ	104
4.4.3 วิเคราะห์การทำงานของโซล่าเซลล์	104
4.5 สรุปผลการทดลอง	105
4.5.1 สรุปผลการทดสอบระบบการส่งและรับอุณหภูมิโดยผ่านเครือข่ายไร้สาย ZigBee	105
4.5.2 สรุปการทำงานของเซนเซอร์วัดอุณหภูมิ	105
4.5.2 สรุปผลการทดลองการทำงานของโซล่าเซลล์	105
บทที่ 5 สรุปผลการทดลองและข้อเสนอแนะ	
5.1 บทนำ	106
5.2 บทสรุปผลของโครงงาน	106
5.3 ปัญหาและอุปสรรค	106
5.4 ข้อเสนอแนะ	107
บรรณานุกรม	108
ประวัติผู้เขียน	109
ภาคผนวก ก	110
ภาคผนวก ข	111

สารบัญรูป

รายการ	หน้า
รูปที่ 2.1 เปรียบเทียบสเกลอุณหภูมิทั้ง 3 ระบบ	5
รูปที่ 2.2 Single Crystalline Silicon Solar Cell	7
รูปที่ 2.3 Polycrystalline Silicon Solar Cell	7
รูปที่ 2.4 Amorphous Silicon Solar Cell	8
รูปที่ 2.5 หลักการทำงานทั่วไป	8
รูปที่ 2.6 อุปกรณ์ที่ใช้ในระบบโซล่าเซลล์	10
รูปที่ 2.7 เซนเซอร์ตรวจวัดอุณหภูมิชนิดต่างๆ	13
รูปที่ 2.8 Bridge-Nulling A/D Converter	13
รูปที่ 2.9 เซนเซอร์วัดอุณหภูมิ รุ่น DS18B20	14
รูปที่ 2.10 บอร์ดไมโครคอนโทรเลอร์ Arduino ชนิดต่างๆ	16
รูปที่ 2.11 โปรแกรม Arduino	18
รูปที่ 2.12 โครงสร้าง PIC	19
รูปที่ 2.13 บอร์ด Arduino	20
รูปที่ 2.14 การต่อใช้งานแบบต่างๆของบอร์ด Arduino	21
รูปที่ 2.15 รูปแบบการเชื่อมต่อบอร์ด	22
รูปที่ 2.16 เลือกรุ่นบอร์ด Arduino ที่ต้องการ upload	22
รูปที่ 2.17 เลือกหมายเลข Comport ของบอร์ด	22
รูปที่ 2.18 complie โปรแกรม	23
รูปที่ 2.19 ส่วนประกอบของบอร์ด Arduino	23
รูปที่ 2.20 Arduino r3	24
รูปที่ 2.21 Arduino Uno AMD	25
รูปที่ 2.22 Arduino Mega 2560 R3	25
รูปที่ 2.23 Arduino Mega ADK	25
รูปที่ 2.24 Arduino Leonardo	26
รูปที่ 2.25 Arduino Mini 05	27
รูปที่ 2.26 Arduino Pro Mini 328 3.3V	27
รูปที่ 2.27 Arduino Pro Mini 328 5V	28
รูปที่ 2.28 Arduino Ethernet with PoE module	28
รูปที่ 2.29 Arduino Ethernet without PoE module	29
รูปที่ 2.30 Arduino Due	29
รูปที่ 2.31 หน้าต่างโปรแกรม	30
รูปที่ 2.32 การเขียนโปรแกรม	31

สารบัญรูป(ต่อ)

รายการ	หน้า
รูปที่ 2.33 คอมไพล์โปรแกรม	31
รูปที่ 2.34 คอมไพล์เรียบร้อย	32
รูปที่ 2.35 ต่อบอร์ด Arduino UNO R3 เข้ากับคอมพิวเตอร์	33
รูปที่ 2.36 เลือก Serial Port	33
รูปที่ 2.37 โหลดโปรแกรมเข้าบอร์ด	34
รูปที่ 2.38 Serial Monitor	34
รูปที่ 2.39 เมื่อเปิด Serial Monitor	35
รูปที่ 2.40 หน้าตา Software X-CTU ที่ใช้ร่วมกับ Xbee (Free Download	36
รูปที่ 2.41 เลือก moderm	37
รูปที่ 2.42 ตั้ง parameter	37
รูปที่ 2.43 Mini Xbee USB Dongle V2	38
รูปที่ 2.44 Xbee Convert PIN to 2.54 Pitch	39
รูปที่ 2.45 Bluebee - Xbee Breakout board	39
รูปที่ 2.46 BlueBee Dongle	40
รูปที่ 2.47 set parameter ให้ติดต่อกันแบบ Point to Point	41
รูปที่ 2.47 set parameter	42
รูปที่ 3.1 โครงสร้างรวมของระบบ	43
รูปที่ 3.2 โครงสร้างระบบภาคส่ง	44
รูปที่ 3.3 โครงสร้างระบบภาครับ	44
รูปที่ 3.4 การเชื่อมต่อวงจรภาคส่ง	45
รูปที่ 3.5 การเชื่อมต่อวงจรภาคส่ง	45
รูปที่ 3.6 วงจรพื้นฐานของเซนเซอร์ DS8B20	46
รูปที่ 3.7 การเริ่มติดต่อสื่อสารแบบ 1-wire ด้วย Reset pulse และ Presence pulse	47
รูปที่ 3.8 การต่อใช้งาน DS18B20	47
รูปที่ 3.9 การเขียนข้อมูลจาก DS18B20	47
รูปที่ 3.10 การอ่านข้อมูลจาก DS18B20	48
รูปที่ 3.11 ขั้นตอนการแปลงและอ่านอุณหภูมิจาก DS18B20	49
รูปที่ 3.12 บอร์ดไมโครคอนโทรเลอร์ Arduino UNO R3	50
รูปที่ 3.13 โครงสร้างของบอร์ด Arduino UNO R3	50
รูปที่ 3.14 การใช้งานบอร์ด Arduino UNO R3	52
รูปที่ 3.15 เสียบสาย USB (ซึ่งต่ออยู่กับบอร์ด Arduino)	52
รูปที่ 3.16 ดู driver Arduino UNO R3	53

สารบัญรูป(ต่อ)

รายการ	หน้า
รูปที่ 3.17 Update Driver	53
รูปที่ 3.18 update driver software	54
รูปที่ 3.19 Browse เพื่อหาตำแหน่งเก็บไฟล์ Driver	54
รูปที่ 3.20 ตำแหน่ง driver	55
รูปที่ 3.21 next install program	55
รูปที่ 3.22 รอลงโปรแกรม	56
รูปที่ 3.23 Install เพื่อติดตั้ง Driver ลง window	56
รูปที่ 3.24 เสร็จสิ้นการติดตั้ง	57
รูปที่ 3.25 ดูใน Device Manager	57
รูปที่ 3.26 เลือกชนิด arduino	58
รูปที่ 3.27 เลือกport	58
รูปที่ 3.28 เลือกชนิด Programmer	59
รูปที่ 3.29 ตั้งค่าที่ Preferences	59
รูปที่ 3.30 หน้าต่างตั้งค่า	60
รูปที่ 3.31 ทดลองจากตัวอย่าง	60
รูปที่ 3.32 แสดงสถานะ	61
รูปที่ 3.33 จอแสดงผล LCD	61
รูปที่ 3.34 ตำแหน่งขาของ LCD	62
รูปที่ 3.35 การต่อ LCD เข้ากับบอร์ด Arduino	63
รูปที่ 3.36 วงจรแจ้งเตือน	64
รูปที่ 3.37 โซล่าเซลล์	64
รูปที่ 3.38 แสดง type และ การถ่ายเทพลังงาน	65
รูปที่ 3.39 Xbee Pro	66
รูปที่ 3.40 หน้าต่างโปรแกรม	79
รูปที่ 3.41 เลือกบอร์ด	80
รูปที่ 3.42 การเขียนโปรแกรม	81
รูปที่ 3.43 การ compile	81
รูปที่ 3.44 การ compile เสร็จเรียบร้อย	82
รูปที่ 3.45 การต่อ arduino กับ คอมพิวเตอร์	82
รูปที่ 3.46 การเลือก serialport	83
รูปที่ 3.47 วิธีการแสดงผลที่ serial monitor	83
รูปที่ 3.48 แสดงผลที่ serial monitor	84
รูปที่ 4.1 การปรับBaudrate	85

สารบัญรูป(ต่อ)

รายการ **หน้า**

รูปที่ 4.2 เป็นการเซ็ต Destination Address 86
รูปที่ 4.3 แสดงค่า SL SH ของ Xbee 86
รูปที่ 4.4 การเซ็ตช่องสัญญาณ ที่ใช้ในการสื่อสาร 87
รูปที่ 4.5 การเซ็ตกำลังงานในการส่งข้อมูล (Power Level 88
รูปที่ 4.6 แสดงการเชื่อมวงจรภาคส่ง 89
รูปที่ 4.7 เชื่อมต่อภาครับกับคอมพิวเตอร์ 89
รูปที่ 4.8 แสดงรูปการเชื่อมวงจรภาครับกับภาคส่ง 90
รูปที่ 4.9 แสดงผลการตรวจจับอุณหภูมิ 90
รูปที่ 4.10 ทดสอบวัดภาคส่งที่อุณหภูมิห้อง 91
รูปที่ 4.11 ทดสอบวัดภาคส่งที่อุณหภูมิภายนอก 91
รูปที่ 4.12 ภาครับรับอุณหภูมิจากภายนอกและมีการแจ้งเตือน 92
รูปที่ 4.13 ภาคส่งกลับมาวัดในอุณหภูมิห้องเหมือนเดิม 92
รูปที่ 4.14 แสดงตำแหน่งทดสอบในบริเวณอาคารเรียนรวมสอง 93
รูปที่ 4.15 ภาคส่งอยู่ที่หน้าสหกิจศึกษา 94
รูปที่ 4.16 ภาคส่งอยู่หน้าห้องอLabcom5 94
รูปที่ 4.17 ภาครับวางไว้อยู่ที่หน้าห้อง 600 95
รูปที่ 4.18 กราฟแสดงความสัมพันธ์ระหว่างสถานที่กับการส่งข้อมูล 96
รูปที่ 4.19 แสดงตำแหน่งทดสอบบริเวณรอบๆอาคารเรียนรวมสอง 97
รูปที่ 4.20 การวัดในสถานที่จริง 97
รูปที่ 4.21 กราฟแสดงความสัมพันธ์ 98
รูปที่ 4.22 กราฟแสดงความสัมพันธ์ระหว่างอุณหภูมิเวลากลางวันกับเวลากลางคืน 101
รูปที่ 4.23 กราฟแสดงความสัมพันธ์ระหว่างอุณหภูมิเวลากลางวันกับเวลากลางคืน 102
รูปที่ 4.24 แผงโซล่าเซลล์ที่ใช้ทดสอบ 103

สารบัญตาราง

รายการ	หน้า
ตารางที่ 3.1 ตำแหน่งของขาและหน้าที่การใช้งานของ LCD	62
ตารางที่ 4.3.1 การทดสอบภายในอาคารและรอบๆอาคารเสมือนเป็น non line-of-sight	93
ตารางที่ 4.3.2 การทดสอบภายนอกอาคารเสมือนเป็น line-of-sight	95
ตารางที่ 4.3.3 ค่าอุณหภูมิที่ตรวจจับได้ในเวลากลางวันที่เวลา 15.30 น. ในหน่วย (องศาเซลเซียส)	99
ตารางที่ 4.3.4 ค่าอุณหภูมิที่ตรวจจับได้ในเวลากลางคืนที่เวลา 19.00 น. ในหน่วย (องศาเซลเซียส)	100
ตารางที่ 4.3.5 ค่าอุณหภูมิที่ตรวจจับได้ในเวลากลางวันที่เวลา 14.40 น. ในหน่วย (องศาเซลเซียส)	101
ตารางที่ 4.3.6 ค่าอุณหภูมิที่ตรวจจับได้ในเวลากลางคืนที่เวลา 20.00 น. ในหน่วย (องศาเซลเซียส)	102
ตารางที่ 4.3.7 ทดสอบการใช้งานโซล่าเซลล์ขณะมีแสงแดดน้อยหรือความเข้มแสงน้อย	103
ตารางที่ 4.3.8 ทดสอบการใช้งานโซล่าเซลล์ขณะมีแสงแดดมากหรือความเข้มแสงมาก	103

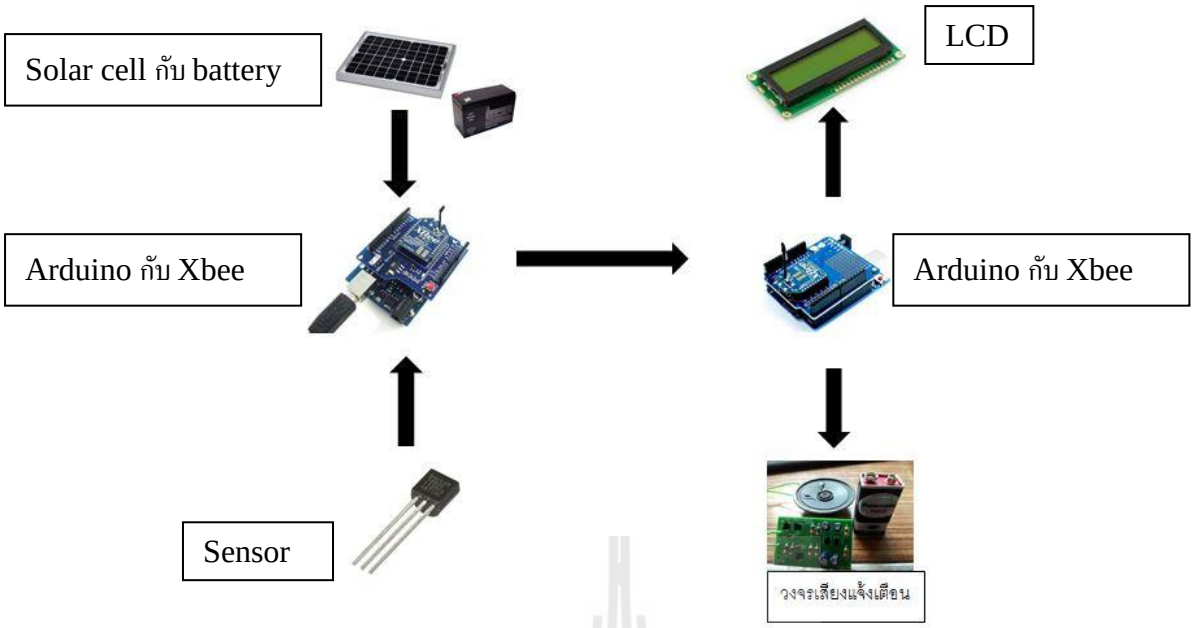
บทที่1
บทนำ

1.1บทนำ

ระบบเครือข่ายท้องถิ่นไร้สายในปัจจุบันนี้กำลังได้รับความนิยมอย่างมากในการนำมาใช้งาน เนื่องจากมีความสะดวก แต่ยังคงมีปัญหาในหลายๆเรื่อง เช่น ระยะทางในการส่ง สิ่งกีดขวางต่างๆ และในระบบตรวจวัดอุณหภูมิอัตโนมัติโดนผ่านเครือข่าย xbee ที่ใช้แหล่งจ่ายโซลล่าเซลล์ นั้นเป็นการติดสื่อสารในระยะทางสั้นๆ โดยติดต่อสื่อสารผ่าน xbee และยังมีการพัฒนาแหล่งจ่ายพลังงานที่ใช้ในภาคส่ง โดยการนำ โซลล่าเซลล์ เข้ามาช่วยในการประหยัดพลังงานและสามารถประหยัดค่าใช้ได้พอสมควร และยังเป็นการวางแผนการใช้พลังงานในระยะยาว เนื่องภาคส่งเราใช้พลังงานน้อย จึงทำให้เหมาะสำหรับการใช้พลังงานโซลล่าเซลล์ ได้อย่างเหมาะสมและโซลล่าเซลล์ก็สามารถผลิตพลังงานได้อย่างเพียงพอกับความต้องการในการงานของภาคส่ง ถึงแม้ว่าบางทีอาจจะเกิดปัญหาฝนตก ไม่มีแดด แต่ยังก็นำแบตเตอรี่มาใช้ในการเก็บพลังงานสำรองไว้ใช้ ในกรณีฉุกเฉินได้

1.2 หลักการและเหตุผล

การทำงานของ XBee จะเป็นการรับ-ส่งคลื่นสัญญาณข้อมูล ผ่านชิปเล็กจุดต่อจุดไปเรื่อยๆ จนถึงปลายทางที่ต้องการดาวน์โหลดข้อมูลลงในเครื่องคอมพิวเตอร์เพื่อใช้ในการวิเคราะห์ข้อมูข้อมูลที่ได้อาจจะเป็นการวัดอุณหภูมิ การเคลื่อนไหวของสิ่งมีชีวิต เป็นต้น ในที่นี้เราใช้ในการวัดอุณหภูมิ โดยที่เหล่งจ่ายที่ใช้ในการส่งใช้พลังงานจากโซล่าเซลล์ ซึ่งการทำงานของโซล่าเซลล์ เป็นขบวนการเปลี่ยนพลังงานแสงเป็นกระแสไฟฟ้าได้โดยตรง โดยเมื่อแสงซึ่งเป็นคลื่นแม่เหล็กไฟฟ้าและมีพลังงานกระทบกับสารกึ่งตัวนำ จะเกิดการถ่ายทอดพลังงานระหว่างกัน พลังงานจากแสงจะทำให้เกิดการเคลื่อนที่ของกระแสไฟฟ้า (อิเลคตรอน) ขึ้นในสารกึ่งตัวนำ จึงสามารถต่อกระแสไฟฟ้าดังกล่าวไปใช้งานได้ นอกจากนี้ผู้ใช้งานต้องรู้จักโปรแกรม X-CTU เพื่อใช้การปรับพารามิเตอร์ของ Xbee ในปัจจุบันการส่งข้อมูลโดยใช้บอร์ด ไมโครคอนโทรลเลอร์ ใช้ไฟจาก adapter ซึ่งถ้าทำงานบอร์ดไมโครคอนโทรลเลอร์ ทั้งวันจะเป็นการสูญเสียพลังงานไฟฟ้ามากเกินไป จากการเล็งเห็นปัญหาดังกล่าวมาแล้วข้างต้นผู้เสนอโครงงานมีความสนใจที่จะพัฒนาให้ใช้พลังงานจากพลังงานแสงอาทิตย์ โดยที่เวลาที่ไม่มีแสงอาทิตย์ เราสามารถใช้พลังงานจากแบตเตอรี่ได้มาควบคุมบอร์ดไมโครคอนโทรลเลอร์



รูปที่ 1 โครงสร้างโดยรวม

ดังรูปที่ 1 แสดงผังไดอะแกรมระบบตรวจวัดอุณหภูมิโดยผ่านเครือข่าย zigbee ที่ใช้แหล่งจ่ายจากโซล่าเซลล์ โดยใช้การควบคุมการส่งและการรับด้วยบอร์ดไมโครคอนโทรลเลอร์ (arduino Uno) ภาครับจะมีวงจรแจ้งเตือน (alarm) แล้วจะรับข้อมูลไปแสดงผลที่จอ LCD ขนาด 16 ตัวอักษร 2 บรรทัด

1.3 วัตถุประสงค์

1. เพื่อศึกษาการใช้งานของ บอร์ดไมโครคอนโทรลเลอร์ (arduino)
2. เพื่อศึกษาการใช้โปรแกรม X-ctu
3. เพื่อศึกษาการใช้ไฟฟ้าจากพลังงานแสงอาทิตย์
4. เพื่อสามารถใช้โปรแกรม Arduino IDE บนพื้นฐานภาษา C มาควบคุมการทำงานของบอร์ดไมโครคอนโทรลเลอร์ Arduino และ Sensor ได้
5. เพื่อพัฒนาระบบตรวจจับวัดอุณหภูมิที่มีต้นทุนถูกกว่าท้องตลาดได้
6. เพื่อฝึกการทำงานเป็นทีม การแก้ปัญหาต่างๆที่เกิดขึ้น

1.4 ขอบเขตงาน

1. ศึกษาโปรแกรม X-CTU เพื่อใช้ในการ set ค่าพารามิเตอร์ต่างๆของ xbee
2. ศึกษาโปรแกรม Circuit Wizard เพื่อใช้ในการออกแบบวงจรต่างๆ
3. ศึกษาการใช้งาน arduino เพื่อใช้ในการเขียนคำสั่งควบคุมต่างๆ
4. สร้างอุปกรณ์ที่จะใช้ในการทดสอบทั้งภาครับและภาคส่ง
5. นำอุปกรณ์ทั้งหมดมาประกอบรวมกัน
6. ทดสอบชิ้นงานที่ทำตามวัตถุประสงค์

1.5 ขั้นตอนการดำเนินงาน

1. ศึกษาค้นคว้าข้อมูล
2. เขียนโครงการและเสนอโครงงานกับอาจารย์ที่ปรึกษา
3. ศึกษาเกี่ยวกับไมโครคอนโทรลเลอร์ Arduino UNO R3 และ เซนเซอร์วัดอุณหภูมิ DS18B20
4. ศึกษาโปรแกรม x-ctu และโปรแกรม circuit wizard
5. เตรียมการซื้ออุปกรณ์
6. เชื่อมการทำงานของเซนเซอร์วัดอุณหภูมิ DS18B20 และบอร์ดไมโครคอนโทรลเลอร์ Arduino UNO R3 ด้วยโปรแกรม Arduino IDE บนพื้นฐานภาษา C
7. ทำการทดลองวิเคราะห์การทดลองและสรุปผลการทดลอง
8. นำเสนอโครงงาน

1.6 ผลที่คาดว่าจะได้รับ

1. สามารถนำความรู้ทางทฤษฎีที่ศึกษานำมาประยุกต์ใช้ในทางปฏิบัติ
2. สามารถนำความรู้มาใช้ในการประกอบวิชาชีพได้
3. สามารถแก้ปัญหาที่เกิดขึ้นจากการทำงานได้
4. รู้จักการทำงานเป็นทีม

บทที่ 2
หลักการและทฤษฎีที่เกี่ยวข้อง

2.1 บทนำ

เนื้อหาในบทนี้จะกล่าวถึงหลักการ แนวคิดและทฤษฎีที่เกี่ยวข้องกับระบบตรวจวัดอุณหภูมิอัตโนมัติโดยผ่านเครือข่าย xbee วัดอุณหภูมิโดยมีแหล่งจ่ายเป็นโซล่าเซลล์โดยเริ่มตั้งแต่ความรู้เบื้องต้นและคุณสมบัติของความอุณหภูมิ ตลอดจนเครื่องมือ อุปกรณ์และวิธีการทำเครื่องมือตรวจวัดอุณหภูมิที่สามารถใช้ได้ เพื่อนำมาศึกษาเป็นแนวทางไปสู่การพัฒนาระบบตรวจวัดอุณหภูมิอัตโนมัติโดยผ่านเครือข่าย xbee วัดอุณหภูมิโดยมีแหล่งจ่ายเป็นโซล่าเซลล์

2.2 ความร้อน และอุณหภูมิ

เชื้อเพลิงเปลี่ยนรูปเป็นพลังงานความร้อนในเครื่องยนต์ จากนั้นก็แปรเปลี่ยนเป็นพลังงานกลทำให้พลังงาน (energy) หมายถึง ความสามารถในการทำงาน ตัวอย่างเช่น พลังงานเคมีจากน้ำเชื้อเพลิง ทำให้รถยนต์เคลื่อนที่ พลังงานมีหลายรูปแบบ พลังงานสามารถเปลี่ยนจากรูปหนึ่งไปสู่อีกรูปหนึ่งเช่นพลังงานเคมีจากน้ำรถยนต์เคลื่อนที่

แบ่งพลังงานออกเป็น2ประเภทคือ

- พลังงานศักย์ (Potential energy) หมายถึง ศักยภาพที่จะทำให้เกิดงาน ซึ่งมีอยู่ในวัตถุที่หยุดนิ่งเช่นเชื้อเพลิงอาหาร
- พลังงานจลน์ (Kinetic energy) หมายถึง พลังงานซึ่งเกิดจากการเคลื่อนที่ ตัวอย่างเช่น เมื่อใช้ค้อนตอกตะปู ค้อนทำให้เกิดพลังงานจลน์ดันตะปูให้เคลื่อนที่ ยิ่งค้อนมีมวลมาก และมีความเร็วสูง พลังงานจลน์ก็ยิ่งมาก

ความร้อนและอุณหภูมิ

สสารทั้งหลายประกอบด้วย อะตอมรวมตัวกันเป็นโมเลกุล การเคลื่อนที่ของอะตอม หรือการสั่นของโมเลกุล ทำให้เกิดรูปแบบของพลังงานจลน์ ซึ่งเรียกว่า “ความร้อน” (Heat) พิจารณาพลังงานความร้อน (Heat energy) จากพลังงานทั้งหมดที่เกิดขึ้นจากการเคลื่อนที่ของอะตอมหรือโมเลกุลทั้งหมดของสสาร

อุณหภูมิ (Temperature) หมายถึง การวัดค่าเฉลี่ยของพลังงานจลน์ซึ่งเกิดขึ้นจากอะตอมแต่ละตัว หรือแต่ละโมเลกุลของสสาร เมื่อใส่พลังงานความร้อนให้กับสสาร อะตอมของจะเคลื่อนที่เร็วขึ้น ทำให้อุณหภูมิสูงขึ้น แต่เมื่อลดพลังงานความร้อน อะตอมของสสารจะเคลื่อนที่ช้าลงทำให้อุณหภูมิลดต่ำลง

หากต้มน้ำด้วยถ้วยและหม้อบนเตาเดียวกัน จะเห็นได้ว่าน้ำในถ้วยจะมีอุณหภูมิสูงกว่า แต่จะมีพลังงานความร้อนน้อยกว่าในหม้อ เนื่องจากปริมาณความร้อนขึ้นอยู่กับมวลทั้งหมดของสสาร แต่อุณหภูมิเป็นเพียงค่าเฉลี่ยของพลังงานในแต่ละอะตอม ดังนั้นบรรยากาศชั้นบนของโลก (ชั้นเทอร์โมสเฟียร์) จึงมีอุณหภูมิสูง แต่มีพลังงานความร้อนน้อย เนื่องจากมีมวลอากาศอยู่อย่างเบาบาง

สเกลอุณหภูมิ

- องศาฟาเรนไฮต์

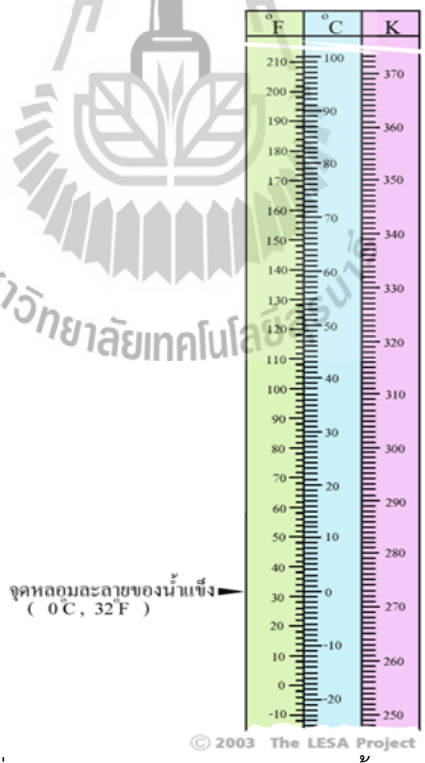
ในปี ค.ศ.1714 กาเบรียล ฟาเรนไฮต์ (Gabrial Fahrenheit) นักฟิสิกส์ชาวเยอรได้ประดิษฐ์เทอร์มอมิเตอร์ซึ่งบรรจุปรอทไว้ในหลอดแก้ว พยายามทำให้ปรอทลดต่ำสุด (0°F) โดยใช้น้ำแข็งและเกลือผสมน้ำ พิจารณาจุดหลอมละลายของน้ำแข็งเท่ากับ 32°F และจุดเดือดของน้ำเท่ากับ212°F

- องศาเซลเซียส

ในปี ค.ศ.1742 แอนเดอส์ เซลเซียส (Anders Celsius) นักดาราศาสตร์ชาวสวีเดน ได้ออกแบบสเกลเทอร์มอมิเตอร์ให้อ่านได้ง่ายขึ้น โดยมีจุดหลอมละลายของน้ำแข็งเท่ากับ 0°C และจุดเดือดของน้ำเท่ากับ100°C

- เคลวิน(องศาสัมบูรณ์)

ต่อมาในคริสศตวรรษที่ 19 ลอร์ด เคลวิน (Lord Kelvin) นักฟิสิกส์ชาวอังกฤษ ผู้ค้นพบความสัมพันธ์ระหว่างความร้อนและอุณหภูมิว่า ณ อุณหภูมิ -273°C อะตอมของสสารจะไม่มีการเคลื่อนที่ และจะไม่มีสิ่งใดหนาวเย็นไปกว่านี้ได้อีก จึงกำหนดให้ 0 K = -273°C (ไม่ต้องใช้เครื่องหมาย ° กำกับหน้าอักษร K) สเกลองศาสัมบูรณ์หรือเคลวิน เช่นเดียวกับองศาเซลเซียสทุกประการ เพียงแต่ +273 เข้าไป เมื่อต้องการเปลี่ยนเคลวินเป็นเซลเซียส ดังรูปที่ 2.1



รูปที่ 2.1 เปรียบเทียบสเกลอุณหภูมิทั้ง 3 ระบบ

ความสัมพันธ์ของสเกลอุณหภูมิ

ระยะสเกลฟาเรนไฮต์	= 212 °F – 32 °F	= 180 °F
ระยะสเกลเซลเซียส	= 100 °C – 0 °C	= 100 °C
สเกลทั้งสองมีความแตกต่างกัน	= 180/100	= 1.8

ความสัมพันธ์ของสเกลทั้งสองจึงเป็นดังนี้

°F	= (1.8 X °C) + 32
°C	= (°F -32) / 1.8

ตัวอย่าง: อุณหภูมิของร่างกายมนุษย์ 98.6°F คิดเป็นองศาเซลเซียสและเคลวินได้เท่าไร

แปลงเป็นองศาเซลเซียส=(°F-32)/1.8

=(98.6-32)/1.8

=37°C

แปลงเป็นองศาสัมบูรณ์=37+273K

= 310 K

2.3 โซล่าเซลล์

โซล่าเซลล์เป็นสิ่งประดิษฐ์กรรมทางอิเลคทรอนิกส์ ที่สร้างขึ้นเพื่อเป็นอุปกรณ์สำหรับเปลี่ยนพลังงานแสงอาทิตย์ให้เป็นพลังงานไฟฟ้า โดยการนำสารกึ่งตัวนำ เช่น ซิลิกอน ซึ่งมีราคาถูกที่สุดและมีมากที่สุดบนพื้นโลกมาผ่านกระบวนการทางวิทยาศาสตร์เพื่อผลิตให้เป็นแผ่นบางบริสุทธิ์ และทันทีที่แสงตกกระทบบนแผ่นเซลล์ รังสีของแสงที่มีอนุภาคของพลังงานประกอบที่เรียกว่า โฟตอน (Proton) จะถ่ายเทพลังงานให้กับอิเล็กตรอน (Electron) ในสารกึ่งตัวนำจนมีพลังงานมากพอที่จะกระโดดออกมาจากแรงดึงดูดของอะตอม (atom) และเคลื่อนที่ได้อย่างอิสระ ดังนั้นเมื่ออิเล็กตรอนเคลื่อนที่ครบวงจรจะทำให้เกิดไฟฟ้ากระแสตรงขึ้น เมื่อพิจารณาลักษณะการผลิตไฟฟ้าจากโซล่าเซลล์พบว่า โซล่าเซลล์จะมีประสิทธิภาพการผลิตไฟฟ้าสูงที่สุดในช่วงเวลากลางวัน ซึ่งสอดคล้องและเหมาะสมในการนำโซล่าเซลล์มาใช้ผลิตไฟฟ้า เพื่อแก้ไขปัญหาการขาดแคลนพลังงานไฟฟ้าในช่วงเวลากลางวัน

2.3.1 ประวัติความเป็นมาของโซล่าเซลล์

โซล่าเซลล์ถูกสร้างขึ้นมาครั้งแรกในปี ค.ศ. 1954 (พ.ศ. 2497) โดย แชปปิน (Chapin) ฟูลเลอร์ (Fuller) และเพียสัน (Pearson) แห่งเบลล์เทลเลโฟน (Bell Telephon) โดยทั้ง 3 นี้ได้ค้นพบเทคโนโลยีการสร้างรอยต่อ พี-เอ็น (P-N) แบบใหม่ โดยวิธีการแพร่สารเข้าไปในผลึกของซิลิกอน จนได้โซล่าเซลล์อันแรกของโลก ซึ่งมีประสิทธิภาพเพียง 6% ซึ่งปัจจุบันนี้โซล่าเซลล์ได้ถูกพัฒนาขึ้นจนมีประสิทธิภาพสูงกว่า 15% แล้ว ในระยะแรกโซล่าเซลล์ส่วนใหญ่จะใช้สำหรับโครงการด้านอวกาศ ดาวเทียมหรือยานอวกาศที่ส่งจากพื้นโลกไปโคจรในอวกาศ ก็ใช้แผงโซล่าเซลล์เป็นแหล่งกำเนิดพลังไฟฟ้า ต่อมาจึงได้มีการนำเอาแผงโซล่าเซลล์มาใช้บนพื้นโลกเช่นในปัจจุบันนี้ โซล่าเซลล์ในยุคแรกๆ

ส่วนใหญ่จะมีสีเทาดำ แต่ในปัจจุบันนี้ได้มีการพัฒนาให้โซล่าเซลล์มีสีต่างๆ กันไป เช่น แดง น้ำเงิน เขียว ทอง เป็นต้น เพื่อความสวยงาม

2.3.2 ชนิดของโซล่าเซลล์

แบ่งตามวัสดุที่ใช้เป็น 3 ชนิดหลักๆ คือ

1.โซล่าเซลล์ที่ทำจากซิลิคอน ชนิดผลึกเดี่ยว (Single Crystalline Silicon Solar Cell) หรือที่รู้จักกันในชื่อ MonocrystallineSilicon Solar Cell และชนิดผลึกรวม (Polycrystalline Silicon Solar Cell) ลักษณะเป็นแผ่นซิลิคอนแข็งและบางมาก

รูปที่ 2.2 Single Crystalline Silicon Solar Cell

2. โซล่าเซลล์ที่ทำจากอะมอร์ฟัสซิลิคอน (Amorphous Silicon Solar Cell) ลักษณะเป็นฟิล์มบางเพียง 0.5 ไมครอน (0.0005 มม.) น้ำหนักเบามาก และประสิทธิภาพเพียง 5-10%

รูปที่ 2.3 Polycrystalline Silicon Solar Cell

3. โซล่าเซลล์ที่ทำจากสารกึ่งตัวนำอื่นๆ เช่น แกลเลี่ยม อาร์เซไนด์, แคดเมียม เทลเลอไรด์ และคอปเปอร์ อินเดียม ไดเซเลไนด์ เป็นต้น มีทั้งชนิดผลึกเดี่ยว (Single Crystalline) และผลึกรวม (Polycrystalline) โซล่าเซลล์ที่ทำจากแกลเลี่ยม อาร์เซไนด์ จะให้ประสิทธิภาพสูงถึง 20-25%

รูปที่ 2.4 Amorphous Silicon Solar Cell

2.3.3 หลักการทำงานทั่วไปของโซล่าเซลล์

รูปที่ **2.5** หลักการทำงานทั่วไป

เมื่อมีแสงอาทิตย์ตกกระทบโซล่าเซลล์ จะเกิดการสร้างพาหะนำไฟฟ้าประจุลบและบวกขึ้น ได้แก่ อิเล็กตรอนและ โฮล โครงสร้างรอยต่อพีเอ็นจะทำหน้าที่สร้างสนามไฟฟ้าภายในเซลล์ เพื่อแยกพาหะนำไฟฟ้าชนิดอิเล็กตรอนไปที่ขั้วลบ และพาหะนำไฟฟ้าชนิดโฮลไปที่ขั้วบวก (ปกติที่ฐานจะใช้สารกึ่งตัวนำชนิดพี ขั้วไฟฟ้าด้านหลังจึงเป็นขั้วบวก ส่วนด้านรับแสงใช้สารกึ่งตัวนำชนิดเอ็น ขั้วไฟฟ้าจึงเป็นขั้วลบ) ทำให้เกิดแรงดันไฟฟ้าแบบกระแสตรงที่ขั้วไฟฟ้าทั้งสอง เมื่อต่อให้ครบวงจรไฟฟ้าจะเกิดกระแสไฟฟ้าไหลขึ้น

2.3.4 ลักษณะเด่นของโซล่าเซลล์

1.ใช้พลังงานจากธรรมชาติ คือ แสงอาทิตย์ ซึ่งสะอาดและบริสุทธิ์ ไม่ก่อปฏิกิริยาที่จะทำให้สิ่งแวดล้อมเป็นพิษ

2. เป็นการนำพลังงานจากแหล่งธรรมชาติมาใช้อย่างคุ้มค่าและไม่มีวันหมดไปจากโลกนี้

3.สามารถนำไปใช้เพื่อผลิตพลังงานไฟฟ้าได้ทุกพื้นที่บนโลก และได้พลังงานไฟฟ้าใช้โดยตรง

4.ไม่ต้องใช้เชื้อเพลิงอื่นใดนอกจากแสงอาทิตย์ รวมถึงไม่มีการเผาไหม้ จึงไม่ก่อให้เกิดมลภาวะด้านอากาศและน้ำ

5.ไม่เกิดของเสียขณะใช้งาน จึงไม่มีการปล่อยมลพิษทำลายสิ่งแวดล้อม

6. ไม่เกิดเสียงและไม่มีการเคลื่อนไหวขณะใช้งาน จึงไม่เกิดมลภาวะด้านเสียง

7.เป็นอุปกรณ์ที่ติดตั้งอยู่กับที่ และไม่มีชิ้นส่วนใดที่มีการเคลื่อนไหวขณะทำงาน จึงไม่เกิดการสึกหรอ

8.ต้องการการบำรุงรักษาน้อยมาก

9. อายุการใช้งานยืนยาวและประสิทธิภาพคงที่

10. มีน้ำหนักเบา ติดตั้งง่าย เคลื่อนย้ายสะดวกและรวดเร็ว

11.เนื่องจากมีลักษณะเป็นโมดูล จึงสามารถประกอบได้ตามขนาดที่ต้องการ

12.ช่วยลดปัญหาการสะสมของก๊าซต่างๆ ในบรรยากาศ เช่น คาร์บอนมอนอกไซด์, ซัลเฟอร์ไดออกไซด์, ไฮโดรคาร์บอน และก๊าซไนโตรเจนออกไซด์ ฯลฯ ซึ่งเป็นผลจากการเผาไหม้ของเชื้อเพลิงจำพวกน้ำ ถ่านหิน และก๊าซธรรมชาติ ล้วนแล้วแต่ส่งผลกระทบต่อสิ่งแวดล้อม เกิดปฏิกิริยาเรือนกระจก ทำให้โลกร้อนขึ้น เกิดฝนกรด และอากาศเป็นพิษ ฯลฯ

2.3.5 คุณสมบัติและตัวแปรที่สำคัญของโซล่าเซลล์

ตัวแปรที่สำคัญที่มีส่วนทำให้โซล่าเซลล์มีประสิทธิภาพการทำงานในแต่ละพื้นที่ต่างกัน และมีความสำคัญในการพิจารณานำไปใช้ในแต่ละพื้นที่ ตลอดจนการนำไปคำนวณระบบหรือคำนวณจำนวนแผงแสงอาทิตย์ที่ต้องใช้ในแต่ละพื้นที่ มีดังนี้

ความเข้มของแสง

กระแสไฟ (Current) จะเป็นสัดส่วนโดยตรงกับความเข้มของแสง หมายความว่าเมื่อความเข้มของแสงสูง กระแสที่ได้จากโซล่าเซลล์ก็จะสูงขึ้น ในขณะที่แรงดันไฟฟ้าหรือโวลต์แทบจะไม่แปรไปตามความเข้มของแสงมากนัก ความเข้มของแสงที่ใช้วัดเป็นมาตรฐานคือ ความเข้มของแสงที่วัดบนพื้นโลกในสภาพอากาศปลอดโปร่ง ปราศจากเมฆหมอกและวัดที่ระดับน้ำทะเลในสภาพที่แสงอาทิตย์ตั้งฉากกับพื้นโลก ซึ่งความเข้ม ของแสงจะมีค่าเท่ากับ 100 mW ต่อ ตร.ซม. หรือ 1,000 W ต่อ ตร.เมตร ซึ่งมีค่าเท่ากับ AM 1.5 (Air Mass 1.5) และถ้าแสงอาทิตย์ทำมุม 60 องศากับพื้นโลกความเข้มของแสง จะมีค่าเท่ากับประมาณ 75 mW ต่อ ตร.ซม. หรือ 750 W ต่อ ตร.เมตร ซึ่งมีค่าเท่ากับ AM2 กรณีของแผงโซล่าเซลล์นั้นจะใช้ค่า AM 1.5 เป็นมาตรฐานในการวัดประสิทธิภาพของแผง

อุณหภูมิ

กระแสไฟ (Current) จะไม่แปรตามอุณหภูมิที่เปลี่ยนแปลงไป ในขณะที่แรงดันไฟฟ้า (โวลท์) จะลดลงเมื่ออุณหภูมิสูงขึ้น ซึ่งโดยเฉลี่ยแล้วทุกๆ 1 องศาที่เพิ่มขึ้น จะทำให้แรงดันไฟฟ้าลดลง 0.5%

และในกรณีของแผงโซล่าเซลล์มาตรฐานที่ใช้กำหนดประสิทธิภาพของแผงแสงอาทิตย์คือ ณ อุณหภูมิ 25 องศา C เช่น กำหนดไว้ว่าแผงแสงอาทิตย์มีแรงดันไฟฟ้าที่วงจรเปิด (Open Circuit Voltage หรือ V oc) ที่ 21 V ณ อุณหภูมิ 25 องศา C ก็จะหมายความว่า แรงดันไฟฟ้าที่จะได้จากแผงแสงอาทิตย์เมื่อยังไม่ได้ต่อกับอุปกรณ์ไฟฟ้า ณ อุณหภูมิ 25 องศา C จะเท่ากับ 21 V ถ้าอุณหภูมิสูงกว่า 25 องศา C เช่น อุณหภูมิ 30 องศา C จะทำให้แรงดันไฟฟ้าของแผงแสงอาทิตย์ลดลง 2.5% (0.5% x 5 องศา C) นั่นคือ แรงดันของแผงแสงอาทิตย์ที่ V oc จะลดลง 0.525 V (21 V x 2.5%) เหลือเพียง 20.475 V (21V – 0.525V) สรุปได้ว่า เมื่ออุณหภูมิสูงขึ้น แรงดันไฟฟ้าก็จะลดลง ซึ่งมีผลทำให้กำลังไฟฟ้าสูงสุดของแผงแสงอาทิตย์ลดลงด้วย

2.3.6 อุปกรณ์สำคัญของระบบการผลิตกระแสไฟฟ้าจากโซล่าเซลล์

โซล่าเซลล์ผลิตไฟฟ้ากระแสตรง จึงนำกระแสไฟฟ้าไปใช้ได้เฉพาะกับอุปกรณ์ไฟฟ้ากระแสตรงเท่านั้น หากต้องการนำไปใช้กับอุปกรณ์ไฟฟ้าที่ใช้ไฟฟ้ากระแสสลับหรือเก็บสะสมพลังงานไว้ใช้ต่อไป จะต้องใช้ร่วมกับอุปกรณ์อื่นๆ อีก โดยรวมเข้าเป็นระบบที่ผลิตกระแสไฟฟ้าจากโซล่าเซลล์ อุปกรณ์สำคัญๆ มีดังนี้

รูปที่ 2.6 อุปกรณ์ที่ใช้ในระบบโซล่าเซลล์

1.แผงโซล่าเซลล์ (Solar Module) ทำหน้าที่เปลี่ยนพลังงานแสงอาทิตย์ให้เป็นพลังงานไฟฟ้า ซึ่งเป็นไฟฟ้ากระแสตรงและมีหน่วยเป็นวัตต์ (Watt) มีการนำแผงโซล่าเซลล์หลายๆ เซลล์มาต่อกันเป็นแถวหรือเป็นชุด (Solar Array) เพื่อให้ได้พลังงานไฟฟ้าใช้งานตามที่ต้องการ โดยการต่อกันแบบอนุกรม จะเพิ่มแรงดันไฟฟ้า และการต่อกันแบบขนาน จะเพิ่มพลังงานไฟฟ้า หากสถานที่ตั้งทางภูมิศาสตร์แตกต่างกัน ก็จะมีผลให้ปริมาณของค่าเฉลี่ยพลังงานสูงสุดในหนึ่งวันไม่เท่ากันด้วย รวมถึงอุณหภูมิก็มีผลต่อการผลิตพลังงานไฟฟ้า หากอุณหภูมิสูงขึ้น การผลิตพลังงานไฟฟ้าจะลดลง

2.เครื่องควบคุมการประจุ (Charge Controller) ทำหน้าที่ประจุกระแสไฟฟ้าที่ผลิตได้จากแผงโซล่าเซลล์เข้าสู่แบตเตอรี่ และควบคุมการประจุกระแสไฟฟ้าให้มีปริมาณเหมาะสมกับแบตเตอรี่

เพื่อยืดอายุการใช้งานของแบตเตอรี่ รวมถึงการจ่ายกระแสไฟฟ้าออกจากแบตเตอรี่ด้วย ดังนั้น การทำงานของเครื่องควบคุมการประจุ คือ เมื่อประจุกระแสไฟฟ้าเข้าสู่แบตเตอรี่จนเต็มแล้ว จะหยุดหรือลดการประจุกระแสไฟฟ้า (และมักจะมีคุณสมบัติในการตัดการจ่ายกระแสไฟฟ้าให้กับอุปกรณ์ไฟฟ้า กรณีแรงดันของแบตเตอรี่ลดลงด้วย) ระบบพลังงานแสงอาทิตย์จะใช้เครื่องควบคุมการประจุกระแสไฟฟ้าในกรณีที่มีการเก็บพลังงานไฟฟ้าไว้ในแบตเตอรี่เท่านั้น

3.แบตเตอรี่ (Battery) ทำหน้าที่เป็นตัวเก็บพลังงานไฟฟ้าที่ผลิตได้จากแผงโซล่าเซลล์ไว้ใช้เวลาที่ต้องการ เช่น เวลาที่ไม่มีแสงอาทิตย์ เวลากลางคืน หรือนำไปประยุกต์ใช้งานอื่นๆ แบตเตอรี่มีหลายชนิดและหลายขนาดให้เลือกใช้งานตามความเหมาะสม

4.เครื่องแปลงกระแสไฟฟ้า (Inverter) ทำหน้าที่แปลงพลังงานไฟฟ้าจากกระแสตรง (DC) ที่ผลิตได้จากแผงโซล่าเซลล์ ให้เป็นพลังงานไฟฟ้ากระแสสลับ (AC) เพื่อให้สามารถใช้ได้กับอุปกรณ์ไฟฟ้ากระแสสลับ แบ่งเป็น 2 ชนิด คือ Sine Wave Inverter ใช้ได้กับอุปกรณ์ไฟฟ้ากระแสสลับทุกชนิด และ Modified Sine Wave Inverter ใช้ได้กับอุปกรณ์ไฟฟ้ากระแสสลับที่ไม่มีส่วนประกอบของมอเตอร์และหลอดฟลูออเรสเซนต์ที่เป็น Electronic ballast

2.3.7การประยุกต์ใช้งานโซล่าเซลล์ในด้านต่างๆ

การนำพลังงานแสงอาทิตย์ซึ่งเป็นพลังงานจากธรรมชาติมาทดแทนพลังงานรูปแบบอื่นๆ ได้รับความสนใจและเป็นที่นิยมมากขึ้นสามารถนำมาใช้ให้เกิดประโยชน์อย่างมากมายในการดำรงชีวิต รวมถึงไม่เป็นการทำลายสิ่งแวดล้อม เช่น

บ้านพักอาศัย	ระบบแสงสว่างภายในบ้าน, ระบบแสงสว่างนอกบ้าน (ไฟสนาม, ไฟโรงจอดรถ และโคมไฟรั้วบ้าน ฯลฯ), อุปกรณ์ไฟฟ้าชนิดต่างๆ , ระบบเปิด-ปิดประตูบ้าน, ระบบรักษาความปลอดภัย, ระบบระบายอากาศ, เครื่องสูบน้ำ, เครื่องกรองน้ำ และไฟสำรองยามฉุกเฉิน ฯลฯ
ระบบสูบน้ำ	อุปโภค, สาธารณูปโภค, ฟาร์มเลี้ยงสัตว์, เพาะปลูก, ทำสวน-ไร่, เหมืองแร่ และชลประทาน ฯลฯ
ระบบแสงสว่าง	โคมไฟป้ายรถเมล์, ตู้โทรศัพท์, ป้ายประกาศ, สถานที่จอดรถ, แสงสว่างภายนอกอาคาร และไฟถนนสาธารณะ ฯลฯ
ระบบประจุแบตเตอรี่	ไฟสำรองไว้ใช้ยามฉุกเฉิน, ศูนย์ประจุแบตเตอรี่ประจำหมู่บ้านในชนบทที่ไม่มีไฟฟ้าใช้, แหล่งจ่ายไฟสำหรับใช้ในครัวเรือนและระบบแสงสว่างในพื้นที่ห่างไกล ฯลฯ
ทำการเกษตร	ระบบสูบน้ำ, พัดลมอบผลผลิตทางการเกษตร และเครื่องนวดข้าว ฯลฯ
เลี้ยงสัตว์	ระบบสูบน้ำ, ระบบเติมออกซิเจนในบ่อน้ำ (บ่อกุ้งและบ่อปลา) และแสงไฟ

	ดักจับแมลง ฯลฯ
อนามัย	ตู้เย็น/กล่องทำความเย็นเพื่อเก็บยาและวัคซีน, อุปกรณ์ไฟฟ้าทางการแพทย์สำหรับหน่วยอนามัย, หน่วยแพทย์เคลื่อนที่ และสถานีอนามัย ฯลฯ
คมนาคม	สัญญาณเตือนทางอากาศ, ไฟนำร่องทางขึ้น-ลงเครื่องบิน, ไฟประภาคาร, ไฟนำร่องเดินเรือ, ไฟสัญญาณข้ามถนน, สัญญาณจราจร, โคมไฟถนน และโทรศัพท์ฉุกเฉิน ฯลฯ
สื่อสาร	สถานีทวนสัญญาณไมโครเวฟ, อุปกรณ์โทรคมนาคม, อุปกรณ์สื่อสารแบบพกพา (เช่น วิทยุสนามของหน่วยงานบริการและทหาร) และสถานีตรวจสอบอากาศ ฯลฯ
บันเทิงและพักผ่อนหย่อนใจ	แหล่งจ่ายไฟฟ้าสำหรับบ้านพักตากอากาศในพื้นที่ห่างไกล, ระบบประจุแบตเตอรี่แบบพกพาติดตัวไปได้ และอุปกรณ์ไฟฟ้าที่ให้ความบันเทิง ฯลฯ
พื้นที่ห่างไกล	ภูเขา, เกาะ, ป่าลึก และพื้นที่สายส่งการไฟฟ้าเข้าไม่ถึง ฯลฯ
อวกาศ	ดาวเทียม

2.4 เซนเซอร์ตรวจวัดอุณหภูมิ (Temperature Sensor)

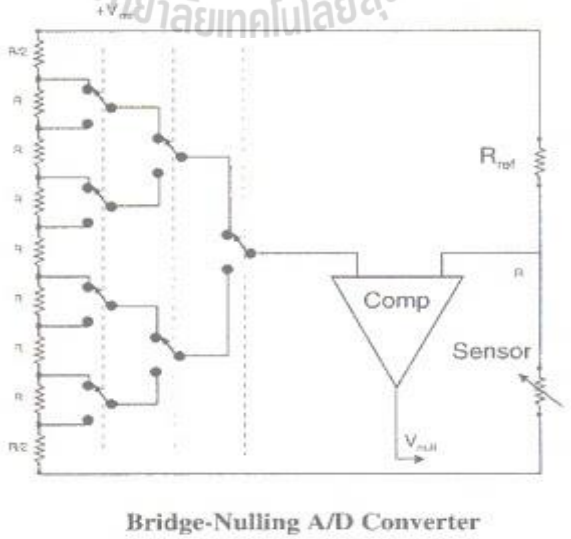
การตรวจวัดอุณหภูมิใช้รูปแบบ การเปลี่ยนแปลงระดับแรงดันไฟฟ้าจากสัญญาณอนาล็อกไปสู่ สัญญาณดิจิตอล โดยสัมพันธ์กับอุณภูมิ มีรูปแบบใหญ่ ๆ ของ เซนเซอร์ อยู่ด้วยกัน 3 รูปแบบ คือ Thermocouple คือ อุปกรณ์วัดอุณหภูมิโดยใช้หลักการเปลี่ยนแปลงอุณหภูมิเป็นแรงเคลื่อนไฟฟ้า ทำมาจากโลหะตัวนำที่ต่างชนิดกัน 2 ตัว มาเชื่อมต่อปลายทั้งสองเข้าด้วยกัน ที่ปลายด้านหนึ่ง เรียกว่า "จุดอุณหภูมิ" ส่วนปลายอีกด้านหนึ่งปล่อยเปิดไว้ เรียกว่า "จุดอ้างอิง" หากที่จุดวัดอุณหภูมิและจุดอ้างอิงมีอุณหภูมิต่างกันก็จะทำให้มีการนำกระแส ในวงจร Thermocouple Resistance Temperature Detector (RTD) คือ ตัวเซ็นเซอร์อุณหภูมิที่ใช้หลักการเปลี่ยนแปลงค่าความต้านทานของโลหะ ซึ่งค่าความต้านทานดังกล่าวจะมีค่าเพิ่มตามอุณหภูมิ ความต้านทานของโลหะที่เพิ่มขึ้นนี้ เรียกว่า “สัมประสิทธิ์การเปลี่ยนแปลงอุณหภูมิแบบบวก” นิยมนำไปใช้ในการวัดอุณหภูมิในช่วง -270 to 850 °C. วัสดุที่นำมาใช้จะเป็นโลหะที่มีความต้านทานจำเพาะต่ำ เช่น แพลทินัม, ทังสเตน และ นิกเกิล

รูปที่ 2.7 เซนเซอร์ตรวจวัดอุณหภูมิชนิดต่างๆ

Thermistor เป็น อุปกรณ์ความต้านทานชนิดที่สามารถเปลี่ยนค่าความต้านทานเมื่อได้รับความร้อน โดยที่ค่าความต้านทานจะเปลี่ยนแปลงแบบไม่เป็นเชิงเส้น กับอุณหภูมิ แบ่งเป็น 2 ลักษณะ คือ Positive Temperature Comitial (PTC) เป็น ชนิดที่ปกติจะมีค่าความต้านทานต่ำ เมื่อได้รับความร้อนจะทำให้มีค่าความต้านทานสูงขึ้นตามลำดับอุณหภูมิ นำไปใช้ตรวจสอบระดับความร้อน หรือทำให้เกิดความร้อนขึ้นเพื่อควบคุมการจ่ายแรงดันไฟฟ้าให้กับขดลวด เช่น วงจรล้างสนามแม่เหล็กอัตโนมัติของเครื่องรับโทรทัศน์สี(Degaussingcoil)เป็นต้น

Negative Temperature Comitial (NTC) เป็นชนิด ที่ปกติจะมีความต้านทานสูงเมื่อได้รับความร้อน ค่าความต้านทานจะต่ำลง ใช้งานด้านการตรวจสอบความร้อนเพื่อควบคุมระดับการทำงาน เช่น ในวงจรขยายเสียงที่ดีใช้ตรวจจับความร้อนที่เกิดจากการทำงานแล้วป้อนกลับไปลด การทำงานของวงจรให้น้อยลง เพื่ออุปกรณ์หลักจะไม่เกิดความร้อนมากจนเกินไป

วงจรการต่ออุปกรณ์



รูปที่ 2.8 Bridge-Nulling A/D Converter

2.4.1 โมดูลเซ็นเซอร์วัดอุณหภูมิ

โมดูลเซ็นเซอร์วัดอุณหภูมิเป็นอุปกรณ์ที่ทำหน้าที่เปลี่ยนระดับอุณหภูมิเป็นสัญญาณไฟฟ้าที่มีความ เที่ยงตรงสูง เช่น IC อุณหภูมิ เซนเซอร์แบบนี้จะให้ความเที่ยงตรงของค่าที่อ่านจากจากเทอมิสเตอร์ ไอซีตระกูล 335 เช่น LM135/LM235/LM335 มีความไวทางด้านเอาต์พุตเป็น 10 mV/° K , ไอซีตระกูล 34 เช่น เบอร์ LM34 จากบริษัท National Semiconductor

DS1820

รูปที่ 2.9 เซ็นเซอร์วัดอุณหภูมิรุ่น DS18B20

2.4.2 การใช้งานเซ็นเซอร์วัดอุณหภูมิ

เซ็นเซอร์วัดอุณหภูมิภายในอาคาร

เซ็นเซอร์วัดอุณหภูมิภายในบรรจุมาในกล่องสำหรับติดผนัง ซึ่งมาพร้อมกับแผงสวิตซ์มาตรฐาน ควรติดตั้งเซ็นเซอร์ไว้กึ่งกลางของบริเวณที่ต้องการวัด ห่างจากประตู หน้าต่าง ช่องลม เครื่องทำความร้อน และผนังด้านนอกที่อาจส่งผลต่อการอ่านค่าอุณหภูมิได้ โดยติดสูงจากพื้นสี่ถึงหกฟุตและยึดด้วยตะปูเกลียวที่ให้มาดังภาพ

เครื่องวัดอุณหภูมิสำหรับช่องแอร์และวอล์ค-อิน บ็อกซ์

เครื่องวัดอุณหภูมิแบบแท่งขนาด 12 นิ้วสามารถใช้วัดอุณหภูมิทั้งในช่องลมเข้าและออกของ AHU หรือ RHU เครื่องวัดสามารถยึดไว้ด้านใดของช่องแอร์ก็ได้ด้วยสกรูเกลียวปล่อย โดยเจาะรูเส้นผ่าศูนย์กลาง 0.250” รูป 30 แสดงภาพการติดตั้งเครื่องวัดแบบ

เซ็นเซอร์วัดอุณหภูมิภายนอกอาคาร

เซ็นเซอร์วัดอุณหภูมิโดยรอบหรือภายนอกอาคารควรติดตั้งไว้ทางด้านเหนือของอาคาร (ด้วยแคลมป์ยางรัดท่อทั้งนี้Emersonได้แถมกรอบอลูมิเนียมและ แคลมป์มาในชุดดังรูป 29 (ไม่รวมสายรัด)สำหรับซีกโลกเหนือ) ภายใต้ชายคาเพื่อป้องกันไม่ให้ไอร้อนจากแสงอาทิตย์ส่งผลต่อการวัดค่าอุณหภูมิ ส่วนสถานที่ในซีกโลกใต้ให้ติดตั้งเซ็นเซอร์ไว้ทางด้านใต้ของอาคารในลักษณะเดียวกัน โดยสามารถยึดเซ็นเซอร์ไว้

เครื่องวัดอุณหภูมิแบบแท่ง/จุ่ม

สามารถใช้กับของเหลวได้ ต่างจากเครื่องวัดแบบจุ่มที่ถูกออกแบบมาเพื่อวัดอุณหภูมิของเหลวซึ่งมีเครื่องวัดอุณหภูมิแบบแท่ง/จุ่มสามารถใช้ตรวจสอบอุณหภูมิ โดยเครื่องวัดแบบแท่งสำหรับใช้กับช่องแอร์เท่านั้นไม่ปลอกหุ้มอีกชั้น ในการติดตั้งให้เจาะรูเพื่อยึดเครื่องวัดแบบแท่งหรือปลอกเครื่องวัดแบบจุ่ม (ต้องใช้แท็ปยึด) เมื่อเจาะรูแล้ว ขันตะปูเกลียวเพื่อยึดเครื่องวัดแบบแท่งหรือปลอกเข้ากับรูที่เจาะไว้ โดยใช้วิธีเดียวกันในการยึดปลอก ตรวจดูให้แน่ใจว่าปลอกของเครื่องวัดชนิดจุ่มได้เคลือบปิดกันน้ำเป็นอย่างดีแล้ว เมื่อติดตั้งปลอกเครื่องวัดแบบจุ่มแล้ว ให้ขันตะปูเกลียวเพื่อยึดเครื่องวัดอุณหภูมิเข้ากับปลอกเป็นอันเสร็จ

เซ็นเซอร์แบบยึดท่อ

ลูกลอยหรือเซ็นเซอร์แบบยึดท่อที่ติดตั้งบนท่อสารทำความเย็นควรจะยึดไว้ด้วยสายเคเบิลอุณหภูมิต่ำแพนดวิท หมายเลข PLT2S-M120 หรือใกล้เคียง ในการติดตั้งเซ็นเซอร์แบบยึดท่อให้ทาบด้านโค้งของเซ็นเซอร์เข้ากับท่อ จากนั้นร้อยสายเคเบิลผ่านร่องด้านบนเซ็นเซอร์ และใช้สายเคเบิลอีกเส้นรัดสายไฟเข้ากับท่ออีกทีเพื่อความแน่นหนา (ผ่อนสายเคเบิลที่รัดช่วงสายไฟกับกล่องเซ็นเซอร์เล็กน้อยไม่ให้แน่นจนเกินไป) ทั้งนี้เซ็นเซอร์ที่ติดตั้งบนท่อสารทำความเย็นนั้นควรจะหุ้มฉนวนเพื่อป้องกันการเหนี่ยวนำจากอากาศรอบๆ โดยแนะนำให้ใช้ฉนวนชนิดมีกาวในตัวที่ไม่ดูดความชื้นเพื่อกันการจับตัวเป็นน้ำแข็งบริเวณเซ็นเซอร์

2.4.3 บอร์ดไมโครคอนโทรเลอร์ Arduino

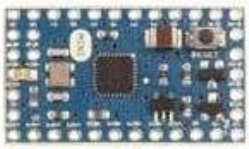
บอร์ด Arduino ถือว่าเป็น Open Hardware Platform ที่ถูกพัฒนาขึ้นโดยมี Micro-controller ของ Atmel เป็นหัวใจหลัก บอร์ด Arduino ที่ผลิตออกมาจำหน่ายในปัจจุบันมีทั้งหมด 20 รุ่น แต่ละรุ่นจะมีลักษณะภายนอกที่แตกต่างกัน รวมถึงคุณสมบัติต่างๆด้วย (บอร์ดบางรุ่นใช้ชื่อเดียวกันแต่มีรายละเอียดของ spec แตกต่างกัน เช่น บอร์ดรุ่น Arduino Pro กับ Arduino Nano, LilyPad Arduino)

Arduino Mega ADK

Arduino Ethernet

LilyPad Arduino USB

LilyPad Arduino Simple



Arduino Mini

Arduino Nano

LilyPad Arduino SimpleSnap

LilyPad Arduino

Arduino Micro

Arduino Pro Mini

Arduino Due

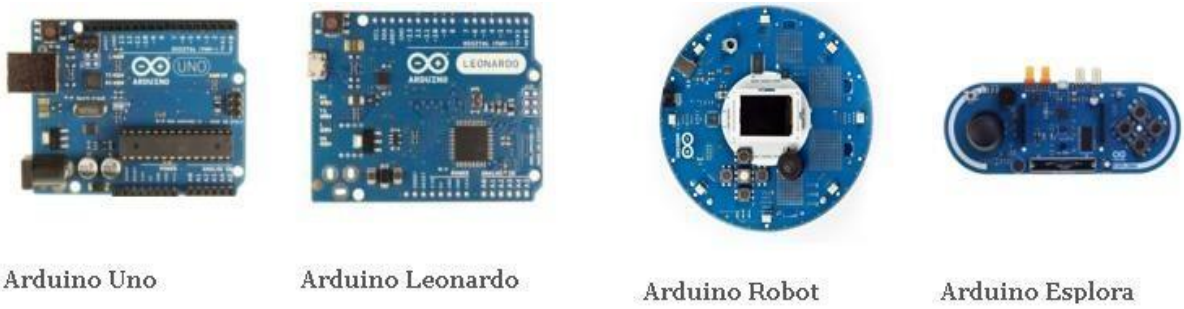
Arduino Yún

Arduino BT

Arduino Mega 2560

Arduino Pro

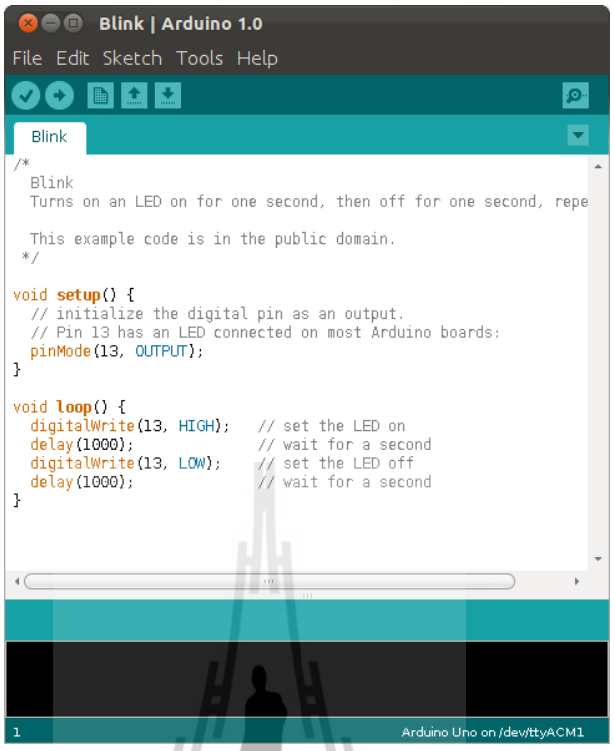
Arduino Fio



รูปที่ 2.10 บอร์ดไมโครคอนโทรเลอร์ Arduino ชนิดต่างๆ

2.4.4 ความแตกต่างของ Arduino กับ ไมโครคอนโทรเลอร์อื่นๆ

การใช้ไมโครคอนโทรเลอร์ไหนก็ตาม ก็ต้องมีบอร์ดที่จะใช้ ซึ่งบนบอร์ด จะมี MCU จะเป็นตัวใหนก็ตาม เช่น PIC, Atmel หรือ พวก MCS 51 ฯ เมื่อมีบอร์ดแล้ว ก่อนที่จะใช้ ก็ต้องเขียนโปรแกรมขึ้นมา บน PC โดยบน PC นี้ก็จะมีอุปกรณ์ (โปรแกรม หรือที่เรียกว่า Integrate Development Environment == IDE) ให้เขียน เช่นบน PIC ก็จะมีโปรแกรมที่ชื่อ MPLab ใน Atmel ก็จะมี Atmel Studio 6 (AS6) ให้มาใช้เพื่อพัฒนา สิ่งที่อยากจะให้ mcu ทำงานให้ โดยจะเขียนโปรแกรมขึ้นมา จะด้วยภาษาอะไรก็ตาม เช่น ภาษา C, Pascal หรือ ภาษา Basic ฯ และ IDE นี้แท้จริงแล้ว ก็คือโปรแกรมที่ทำมาให้ใช้ เพื่อความสดวก ในการพัฒนาโปรแกรม ตัวโปรแกรมเองก็ถูกพัฒนามาด้วยการเขียนด้วยภาษาอะไรก็ได้ เช่น ถูกพัฒนามาด้วย ภาษา C เช่น AS6 หรือ จะเป็นภาษา pascal (Delphi) หรือจะเป็น Java (Arduino) ก็ได้ รูปร่างหน้าตาขึ้นกับบริษัทที่ออกแบบมา



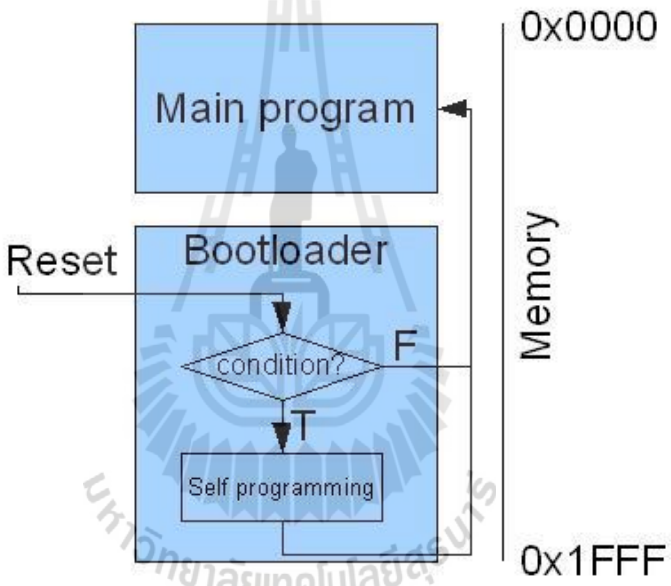
รูปที่ 2.11 โปรแกรม Arduino

เมื่อเขียนโปรแกรมขึ้นมาแล้ว ก็จะต้องเปลี่ยน (Compile) ให้เป็นภาษาเครื่อง เพื่อ MCU จะได้เอาไปทำงานตามที่ต้องการ เมื่อ compile เสร็จแล้วก็จะได้โปรแกรมที่เป็น ภาษาเครื่องขึ้นมา ที่เรียก file ที่ได้มาว่า hex file เมื่อได้มาแล้ว ก็ต้องเอา ภาษาเครื่อง (hex file) นี้ ไปบรรจุเข้าไป ในหน่วยความจำของ MCU ที่อยู่บนตัวบอร์ดที่ใช้ กรรมวิธีการเอาไปลงบน MCU อันนี้เรียกว่า "burn" hex file หรือ จะเรียกว่า "flash" hex หรือจะเรียกว่า "program" hex file เข้าไปใน flash memory กรรมวิธีในการที่จะ "burn" หรือ "flash" หรือจะ "program" hex file นี้ ก็ต้องอาศัย อุปกรณ์ hardware ที่เรียกว่า programmer ซึ่งเป็นอุปกรณ์ต่างหาก ที่ผู้จะใช้ mcu ต้องจัดหามา PIC อุปกรณ์ตัวนี้ก็มีอยู่หลายตัว เช่น PICKit 2 และ PICKit3 เป็นต้น ในสาย Atmel ก็จะมี programmer อยู่หลายตัวมาก ก็จะมี AVRISP MK II และ AVR Dragon เมื่อได้โปรแกรม และ compile ให้ได้เป็นภาษาเครื่องแล้ว ก็ยังต้องอาศัย programmer โดยจะไปคว้า hex file ที่ได้จาก โปรแกรม IDE (เช่น MPLab หรือ AVR Studio/Atmel Studio) เพื่อส่ง hex file นี้ ผ่าน programmer ไปยัง developement board ดัง diagram ข้างบน แค่นี้ mcu ก็จะทำงานให้ ตามที่ต้องการ นี่คือ หลักการทั่วไปของทั้ง PIC , Atmel or MCS 51 และอื่นๆ

Arduino ต่างจาก PIC หรือตัวอื่นๆอย่างไร

หน่วยความจำที่จะบรรจุ hex file เข้าไปใน MCU นั้น มีอยู่สองแบบ คือ หน่วยคำจำที่โปรแกรมเข้าไปอยู่ ซึ่งจะไปอยู่ส่วนล่างสุดของ หรือที่เรียกว่า program Section และอีกส่วนหนึ่งที่ปลายสุดโต่งบนสุดของ program memory นี้ ส่วนนี้จะทำหน้าที่พิเศษ ที่เรียกว่า bootloader Section ในส่วนนี้ เป็นที่อยู่ของ โปรแกรมพิเศษที่เรียกว่า bootloader

bootloader คืออะไร คือโปรแกรมซึ่งส่วนใหญ่ไม่ใหญ่โตมากนัก ที่ทำหน้าที่ทันทีที่เปิดเครื่องเมื่อ mcu ที่มีโปรแกรมชนิดที่เรียกว่า bootloader แล้ว ก็สามารถติดต่อกับ MCU โดยไม่ต้องอาศัย programmer เลย และสามารถติดต่อจาก PC โดยใช้ผ่าน serial communication เข้าไปยัง MCU เลยนี่คือ ลักษณะหนึ่งของบอร์ด arduino ซึ่งใช้ chip ของ Atmel กล่าวคือ ตัว MCU จะมีโปรแกรม bootloader ใส่เข้ามาให้เรียบร้อยแล้ว (ทำให้ประหยัดค่า programmer)

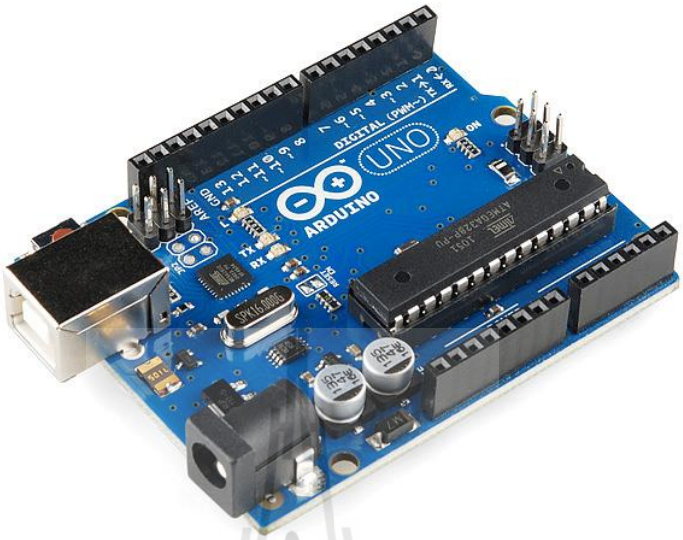


รูปที่ 2.12 โครงสร้าง PIC

อีกสิ่งหนึ่งที่เป็นจุดดึงดูดให้คนหันมาใช้ arduino มากก็คือ ให้ library เข้ามาใช้มากมาย ทำให้สามารถพัฒนาสิ่งที่อยากจะทำ จะใช้ ได้ง่าย หากดูจากสิ่งที่ผู้คนพัฒนาบน arduino นั้น จะเห็นว่ามีคนคิดประดิษฐ์ทำส่งของต่างๆแยะมาก ก็คือการที่ขาดองค์ความรู้เรื่องที่ต้องการจะทำมากกว่า และไม่ต้องกลัวว่า arduino จะรองรับไม่ได้

สรุปว่า Arduino คือการเอา Atmel chip คือต้ว Atmega 168/328 มาบรรจุ bootloader ซึ่งทำให้การ burn หรือ flash หรือจะprogramภาษาเครื่อง (hex file) ง่ายขึ้น

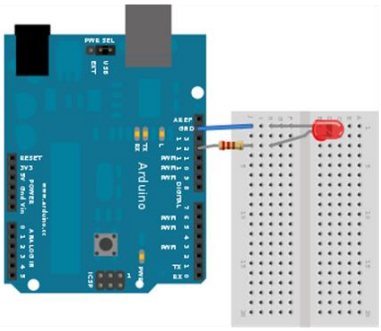
Arduino คืออะไร



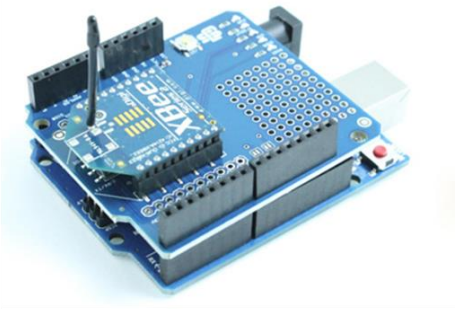
รูปที่ 2.13 บอร์ด Arduino

Arduino อ่านว่า (อา-ดู-อิ-โน่ หรือ อาดุยโน่) เป็นบอร์ดไมโครคอนโทรเลอร์ตระกูล AVR ที่มีการพัฒนาแบบ OpenSource คือมีการเปิดเผยข้อมูลทั้งด้าน Hardware และ Software ตัวบอร์ด Arduino ถูกออกแบบมาให้ใช้งานได้ง่าย ดังนั้นจึงเหมาะสำหรับผู้เริ่มต้นศึกษา ทั้งนี้ยังสามารถดัดแปลง เพิ่มเติม พัฒนาต่อยอดทั้งตัวบอร์ด หรือโปรแกรมต่อได้อีกด้วย

ความง่ายของบอร์ด Arduino ในการต่ออุปกรณ์เสริมต่างๆ คือสามารถต่อวงจรอิเล็กทรอนิคส์จากภายนอกแล้วเชื่อมต่อเข้ามาที่ขา I/O ของบอร์ด (ดูตัวอย่างรูปที่ 1) หรือเพื่อความสะดวกสามารถเลือกต่อกับบอร์ดเสริม (Arduino Shield) ประเภทต่างๆ (ดูตัวอย่างรูปที่ 2) เช่น Arduino XBee Shield, Arduino Music Shield, Arduino Relay Shield, Arduino Wireless Shield, Arduino GPRS Shield เป็นต้น มาเสียบกับบอร์ดบนบอร์ด Arduino แล้วเขียนโปรแกรมพัฒนาต่อได้เลย



รูปที่1 บอร์ด Arduino ต่อกับ LED



รูปที่2 บอร์ด Arduino ต่อกับบอร์ด XBee Shield

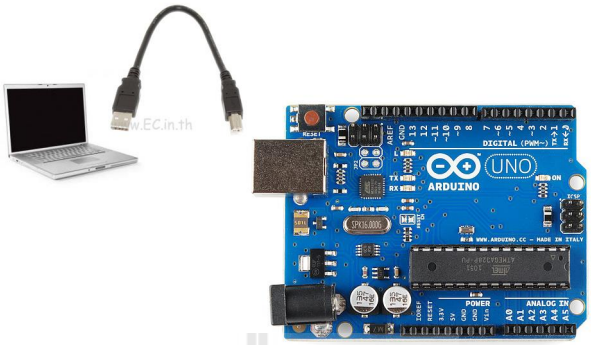
รูปที่ 2.14 การต่อใช้งานแบบต่างๆของบอร์ด Arduino

จุดเด่นที่ทำให้บอร์ด Arduino เป็นที่นิยม

1. ง่ายต่อการพัฒนา มีรูปแบบคำสั่งพื้นฐาน ไม่ซับซ้อนเหมาะสำหรับผู้เริ่มต้น
2. มี Arduino Community กลุ่มคนที่ร่วมกันพัฒนาที่แข็งแรง
3. Open Hardware ทำให้ผู้ใช้สามารถนำบอร์ดไปต่อยอดใช้งานได้หลายด้าน
4. ราคาไม่แพง
5. Cross Platform สามารถพัฒนาโปรแกรมบน OS ใดก็ได้

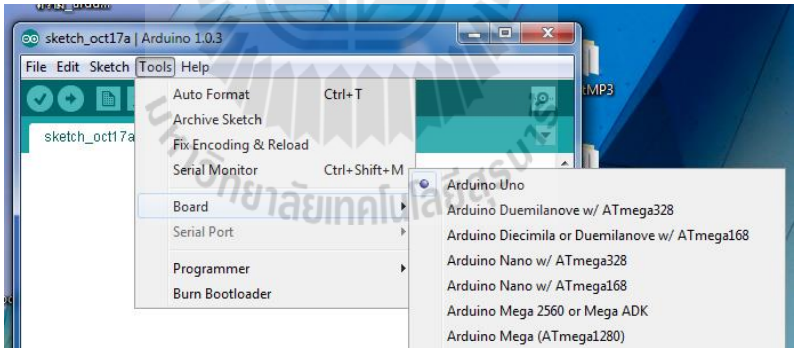
จะเห็นว่า บอร์ด Arduino คือบอร์ดไมโครคอนโทรลเลอร์ที่มี โปรแกรมภาษาที่ง่ายต่อการใช้งาน สามารถเรียกฟังก์ชั่นต่างๆ ของ AVR ออกมาใช้ทำงานได้อย่างง่ายดาย ไม่ว่าจะเป็น PWM , Serial , Port In Out , A to D, หรือแม้กระทั่งการใช้งานกับ LCD Graphic ต่างๆ ก็มีผู้คนทั่วโลกพัฒนาให้สามารถนำมาใช้งานต่อร่วมได้โดยง่าย อีกทั้งยังมี การเชื่อมต่อกับ คอมพิวเตอร์แบบ USB ทำให้การโปรแกรมต่างๆ นั้น ทำได้โดยสะดวก การเขียนโปรแกรมเป็นเรื่องง่ายมากกับการเทียบการใช้ไมโครคอนโทรลเลอร์ที่ต้องเขียนโปรแกรมเองทั้งหมดแต่ บอร์ด Arduino นี้เขียนโปรแกรมด้วยภาษาง่ายๆ สามารถทำความเข้าใจได้ง่ายและมีประสิทธิภาพสูงมากๆในเรื่องของโปรแกรม

รูปแบบการเขียนโปรแกรมบน Arduino

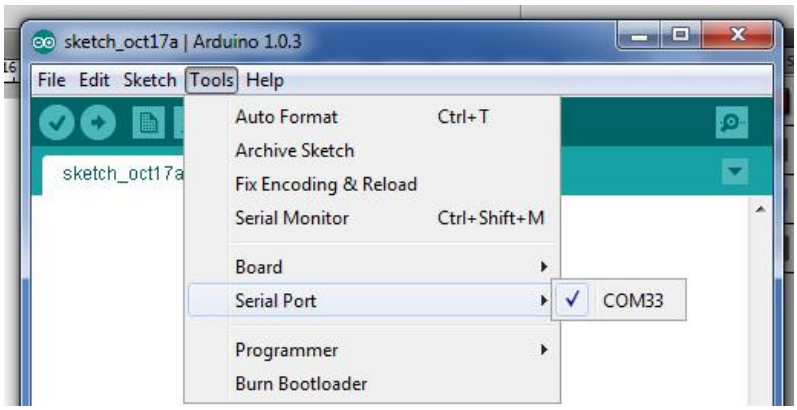


รูปที่ 2.15 รูปแบบการเชื่อมต่อบอร์ด

1. เขียนโปรแกรมบนคอมพิวเตอร์ผ่านทางโปรแกรม ArduinoIDE ซึ่งสามารถดาวน์โหลดได้จาก Arduino.cc/en/main/software
2. หลังจากที่เขียนโค้ดโปรแกรมเรียบร้อยแล้วให้เลือกรุ่นบอร์ด Arduino ที่ใช้และหมายเลข Com port

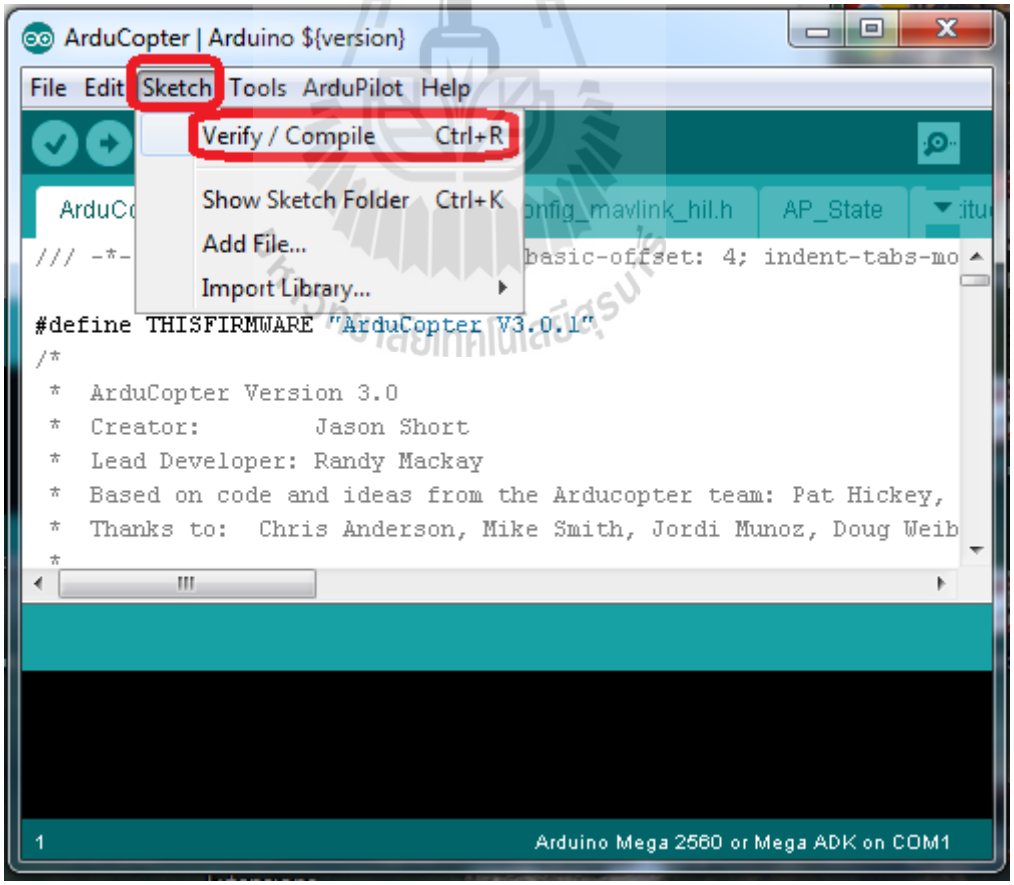


รูปที่ 2.16 เลือกรุ่นบอร์ด Arduino ที่ต้องการ upload



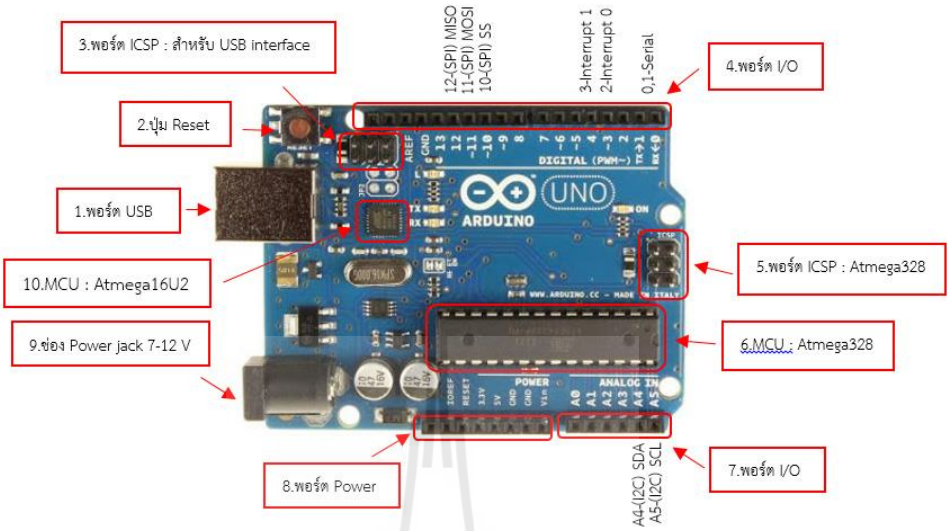
รูปที่ 2.17เลือกหมายเลข Comport ของบอร์ด

3. กดปุ่ม Verify เพื่อตรวจสอบความถูกต้องและ Compile โค้ดโปรแกรมจากนั้นกดปุ่ม Upload โค้ด โปรแกรมไปยังบอร์ด Arduino ผ่านทางสาย USB เมื่ออัปโหลดเรียบร้อยแล้ว จะแสดงข้อความแถบข้างล่าง “Done uploading” และบอร์ดจะเริ่มทำงานตามที่เขียนโปรแกรมไว้ได้ทันที



รูปที่ 2.18 Compile โปรแกรม

Layout & Pin out Arduino Board (Model: Arduino UNO R3)

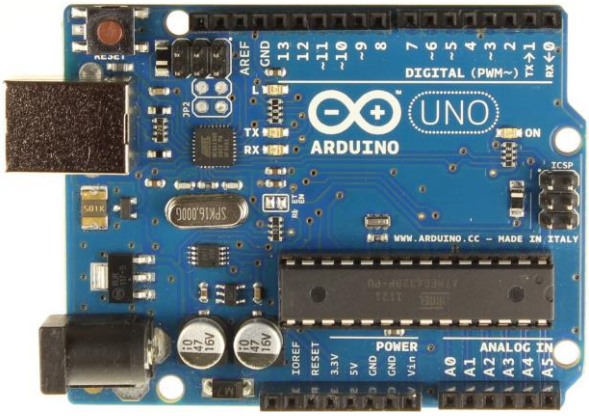


รูปที่ 2.19 ส่วนประกอบของบอร์ด Arduino

1.USBPort: ใช้สำหรับต่อกับ Computer เพื่ออัปโหลดโปรแกรมเข้า MCU และจ่ายไฟให้กับบอร์ด
2.Reset Button: เป็นปุ่ม Reset ใช้กดเมื่อต้องการให้ MCU เริ่มการทำงานใหม่
3.ICSP Port ของ Atmega16U2 เป็นพอร์ตที่ใช้โปรแกรม Visual Com port บน Atmega16U2
4. I/OPort:Digital I/O ตั้งแต่ขา D0 ถึง D13 นอกจากนี้ บาง Pin จะทำหน้าที่อื่นๆ เพิ่มเติมด้วย เช่น Pin0,1 เป็นขา Tx,Rx Serial, Pin3,5,6,9,10 และ 11 เป็นขา PWM
5.ICSP Port: Atmega328 เป็นพอร์ตที่ใช้โปรแกรม Bootloader
6. MCU: Atmega328 เป็น MCU ที่ใช้บนบอร์ด Arduino
7. I/OPort: นอกจากจะเป็น Digital I/O แล้ว ยังเปลี่ยนเป็น ช่องรับสัญญาณอนาล็อก ตั้งแต่ขา A0-A5
8.PowerPort ไฟเลี้ยงของบอร์ดเมื่อต้องการจ่ายไฟให้กับวงจรภายนอกประกอบด้วยขาไฟเลี้ยง +3.3 V, +5V, GND, Vin
9. Power Jack: รับไฟจาก Adapter โดยที่แรงดันอยู่ระหว่าง 7-12 V
10. MCU ของ Atmega16U2 เป็น MCU ที่ทำหน้าที่เป็น USB to Serial โดย Atmega328 จะติดต่อกับ Computer ผ่าน Atmega16U2

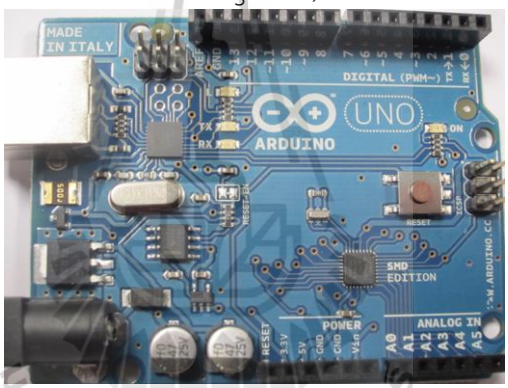
ประเภทของบอร์ดArduino แต่ละประเภท

1. Arduino Uno R3 เป็นบอร์ด Arduino ที่ได้รับความนิยมมากที่สุด เนื่องจากราคาไม่แพง ส่วนใหญ่โปรเจคและ Library ต่างๆ ที่พัฒนาขึ้นมา Support จะอ้างอิงกับบอร์ดนี้เป็นหลัก และข้อดีอีกอย่างคือ กรณีที่ MCU เสีย สามารถซื้อมาเปลี่ยนเองได้



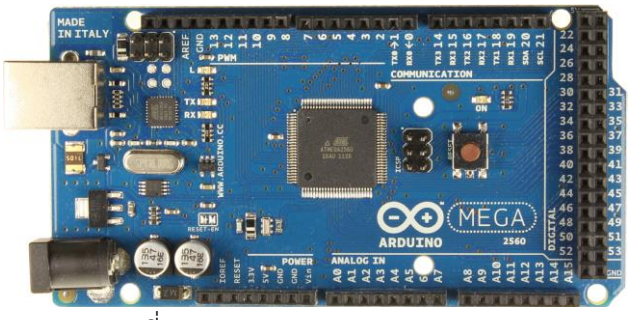
รูปที่ 2.20 Arduino r3

2. Arduino Uno SMD เป็นบอร์ดที่มีคุณสมบัติและการทำงานเหมือนกับบอร์ด Arduino UNO R3 ทุกประการ แต่จะแตกต่างกับที่Package ของ MCU ซึ่งบอร์ดนี้จะมี MCU ที่เป็น Package SMD (Arduino UNO R3 มี MCU ที่เป็น Package DIP)



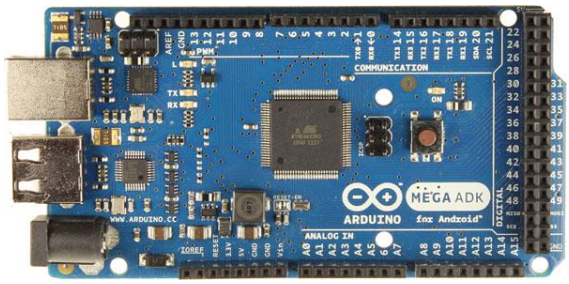
รูปที่ 2.21 Arduino Uno AMD

3. Arduino Mega 2560 R3 เป็นบอร์ด Arduino ที่ออกแบบมาสำหรับงานที่ต้องใช้ I/O มากกว่า ArduinoUnoR3 เช่นงานที่ต้องการรับสัญญาณจาก Sensor หรือควบคุมมอเตอร์ Servo หลายๆ ตัว ทำให้ Pin I/O ของบอร์ด Arduino Uno R3 ไม่สามารถรองรับได้ ทั้งนี้บอร์ด Mega 2560 R3 ยังมีความหน่วยความจำแบบ Flash มากกว่า Arduino Uno R3 ทำให้สามารถเขียนโค้ดโปรแกรมเข้าไปได้มากกว่า ในความเร็วของ MCU ที่เท่ากัน



รูปที่ 2.22 Arduino Mega 2560 R3

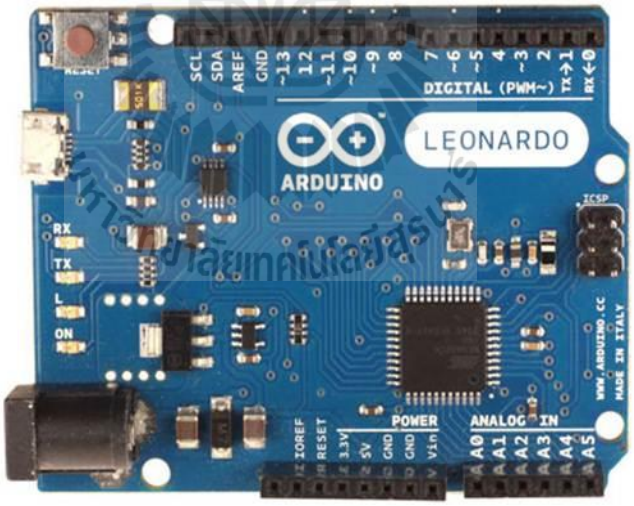
4. Arduino Mega ADK เป็นบอร์ดที่ออกแบบมาให้บอร์ด Mega 2560 R3 สามารถติดต่อกับอุปกรณ์ Android Deviceผ่านพอร์ตUSB Host ของบอร์ดได้



รูปที่ 2.23 Arduino Mega ADK

5. Arduino Leonardo การทำงานจะคล้ายกับบอร์ด Arduino Uno R3 แต่มีการเปลี่ยน MCU ตัวใหม่เป็น ATmega32U4 ซึ่งมีโมดูลพอร์ต USB มาด้วยบนชิป (แตกต่างจากบอร์ด Arduino UNO R3 หรือ Arduino Mega 2560 ที่ต้องใช้ชิป ATmega16U2 ร่วมกับAtmega328 ในการเชื่อมต่อกับพอร์ต USB)

ข้อควรระวัง: เนื่องจาก MCU เป็นคนละเบอร์กับ Arduino Uno R3 อาจจะทำให้บอร์ด Shield บางตัวหรือ Library ใช้ร่วมกันกับบอร์ดArduino Leonardo ไม่ได้ จำเป็นต้องตรวจสอบก่อนใช้งาน



รูปที่ 2.24 Arduino Leonardo

6. Arduino Mini 05 เป็นบอร์ด Arduino ขนาดเล็กที่ใช้ MCU เบอร์ ATmega328 เบอร์เดียวกับบอร์ด Arduino UNO R3

ข้อแตกต่าง: บอร์ด Arduino Mini 05 จะไม่มีพอร์ต USB มาให้ ต้องต่อกับบอร์ด USB to Serial Converter เพิ่มเมื่อต้องการโปรแกรมบอร์ด

รูปที่ 2.25 Arduino Mini 05

7. Arduino Pro Mini 328 3.3V เป็นบอร์ด Arduino ขนาดเล็ก ที่ใช้ MCU เบอร์ ATmega328 ซึ่งจะคล้ายกับบอร์ด Arduino Mini 05 แต่บนบอร์ดจะมี Regulator 3.3 V ชุดเดียวเท่านั้น ระดับแรงดันไฟที่ขา I/O คือ 3.3V

รูปที่ 2.26 Arduino Pro Mini 328 3.3V

8. Arduino Pro Mini 328 5V เป็นบอร์ด Arduino ขนาดเล็ก ที่ใช้ MCU เบอร์ ATmega328 เช่นเดียวกับบอร์ด Arduino Mini 05 แต่บนบอร์ดจะมี Regulator 5V ชุดเดียวเท่านั้นระดับแรงดันไฟที่ขา I/O คือ 5V

รูปที่ 2.27 Arduino Pro Mini 328 5V

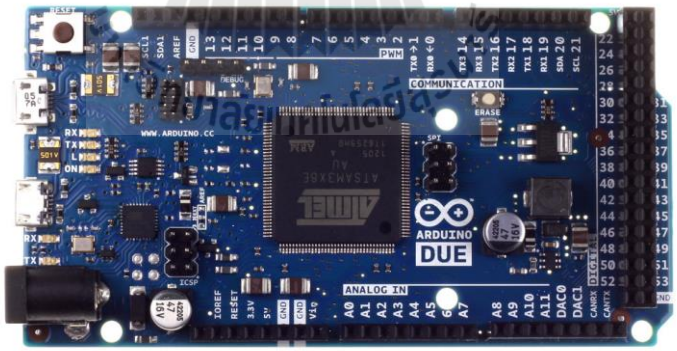
9. Arduino Ethernet with PoE module เป็นบอร์ด Arduino ที่ใช้ MCU เบอร์เดียวกับ Arduino Uno SMD ในบอร์ดมีชิป Ethernet และช่องสำหรับเสียบ SD Card รวมทั้งโมดูล POE ทำให้บอร์ดนี้สามารถใช้แหล่งจ่ายไฟจากสาย LAN ได้โดยตรง โดยไม่ต้องต่อ Adapter เพิ่ม แต่บอร์ด Arduino Ethernet with PoE module นี้จะไม่มีพอร์ต USB ทำให้เวลาโปรแกรมต้องต่อบอร์ด USB toSerial Converter เพิ่มเติม

รูปที่ 2.28 Arduino Ethernet with PoE module

10. Arduino Ethernet without PoE module บอร์ดนี้จะตัดโมดูล POE ออกไป ต้องใช้ไฟจากพอร์ต Power Jack เท่านั้น คุณสมบัติอื่นๆ จะเหมือนกับบอร์ด Arduino Ethernet with PoE module

รูปที่ 2.29 Arduino Ethernet without PoE module

11. Arduino Due เป็นบอร์ด Arduino ที่เปลี่ยนชิป MCU ใหม่ซึ่งจากเดิมเป็นตระกูล AVR เปลี่ยนเป็นเบอร์ AT91SAM3X8E (ตระกูลARMCortex-M3) แทนทำให้การประมวลผลเร็วขึ้นแต่ยังคงรูปแบบโค้ดโปรแกรมของ Arduino ที่ง่ายอยู่ ข้อควรระวัง: เนื่องจาก MCUเป็นคนละเบอร์กับ Arduino Uno R3 อาจะทำให้บอร์ด Shield บางตัวหรือ Library ใช้ร่วมกันกับบอร์ด Arduino Leonardo ไม่ได้ จำเป็นต้องตรวจสอบก่อนใช้งาน

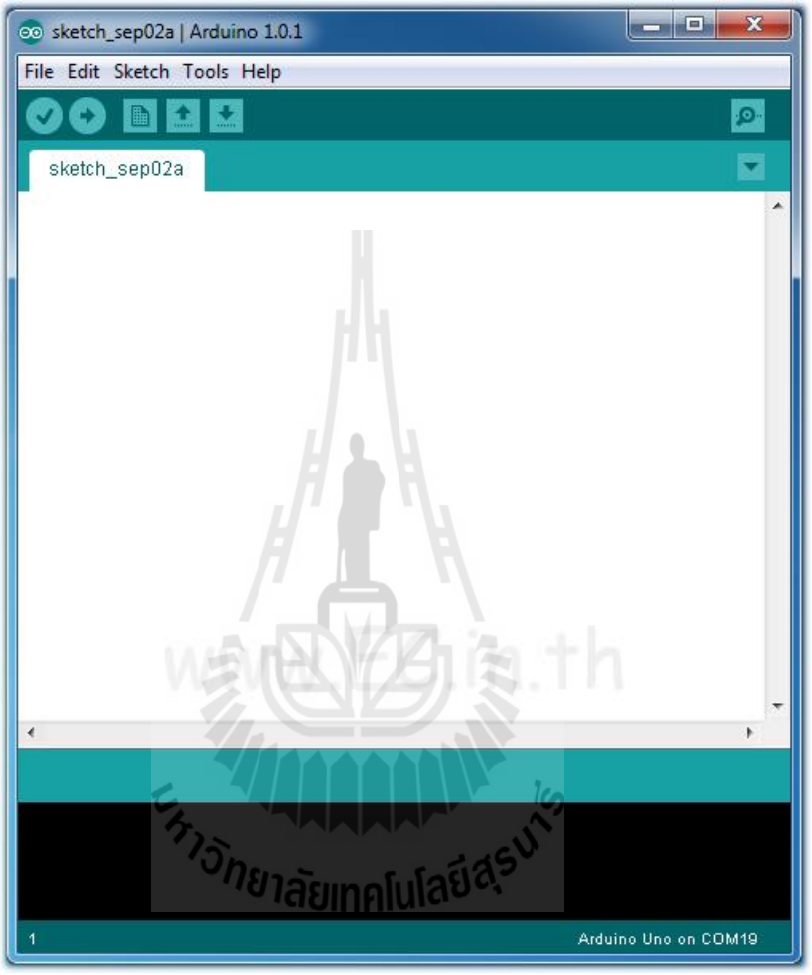


รูปที่ 2.30 Arduino Due

ตัวอย่างการใช้ Arduino กับบอร์ดไมโครคอนโทรลเลอร์

1. เปิดโปรแกรม Arduino IDE ที่ดาวน์โหลดมาจาก http://arduino.cc/en/Main/Software

2. เมื่อเปิดโปรแกรมแล้วจะพบกับหน้าต่างของ IDE ดังรูป



รูปที่ 2.31 หน้าต่างโปรแกรม

3. ไปที่ Tools->Board แล้วเลือกให้ตรงกับบอร์ดที่ใช้งาน สำหรับ Arduino UNO R3 ให้เลือกบอร์ด Arduino UNO

4. เขียนโปรแกรมดังข้อความด้านล่างนี้

```
/*
  Arduino "Hello world"

  This example code is in www.EC.in.th.
  */

// the setup routine runs once when you press reset:
void setup() {
   // initialize serial communication at 9600 bits per second:
   Serial.begin(9600);
}

// the loop routine runs over and over again forever:
void loop() {

   // print out the value you read:
   Serial.print("Hello World\n\r");
}
```

รูปที่ 2.32 การเขียนโปรแกรม

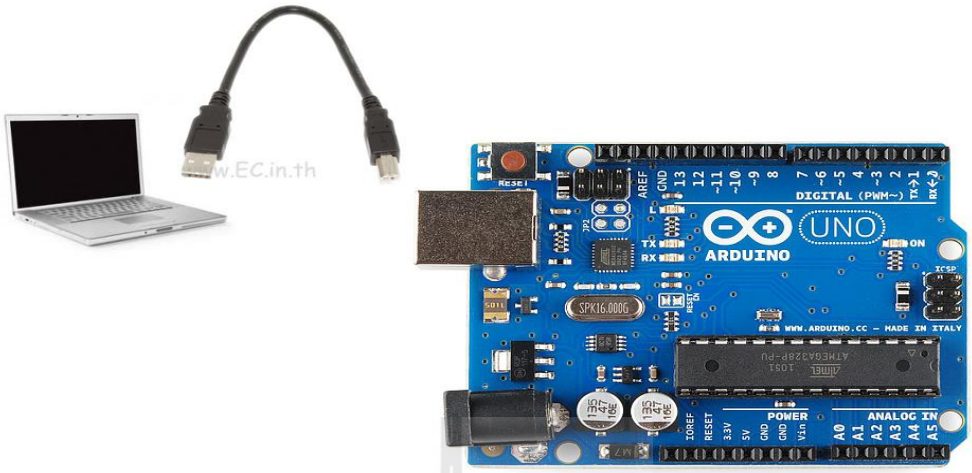
5. จากนั้นคอมไพล์โปรแกรมโดยไปที่ Sketch->Verify / Compile

รูปที่ 2.33 คอมไพล์โปรแกรม

6. เมื่อคอมไฟล์เรียบร้อยจะมีข้อความปรากฏดังรูป

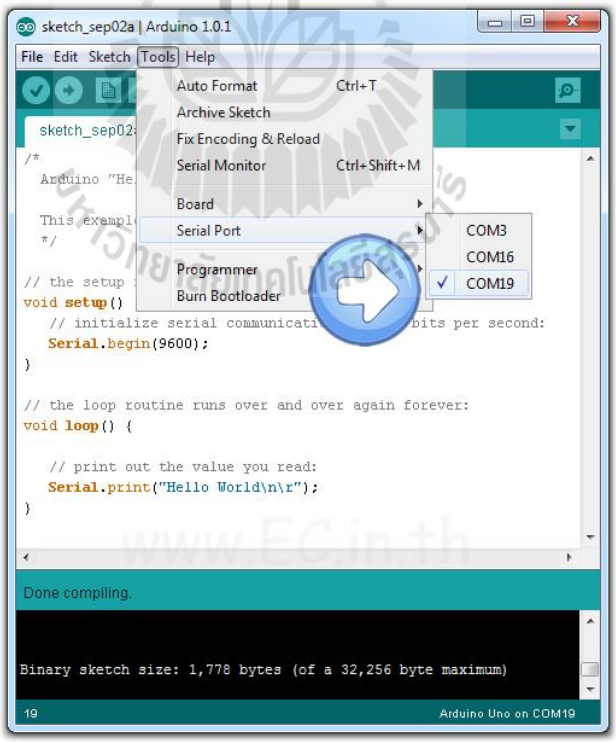
รูปที่ 2.34 คอมไพล์เรียบร้อย

7. ต่อบอร์ด Arduino UNO R3 เข้ากับคอมพิวเตอร์ผ่านทางพอร์ต USB



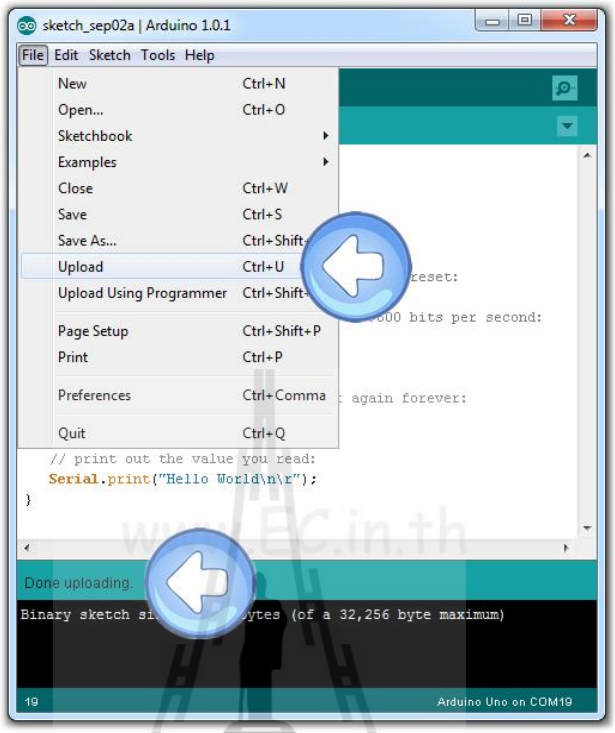
รูปที่ 2.35 ต่อบอร์ด Arduino UNO R3 เข้ากับคอมพิวเตอร์

8. จากนั้นให้ไปที่ Tools->Serial Port และเลือกให้ตรงกับบอร์ด Arduino UNO ที่ใช้งาน (สำหรับบอร์ด Arduino UNO R3 โปรแกรมจะเลือกให้อัตโนมัติ)



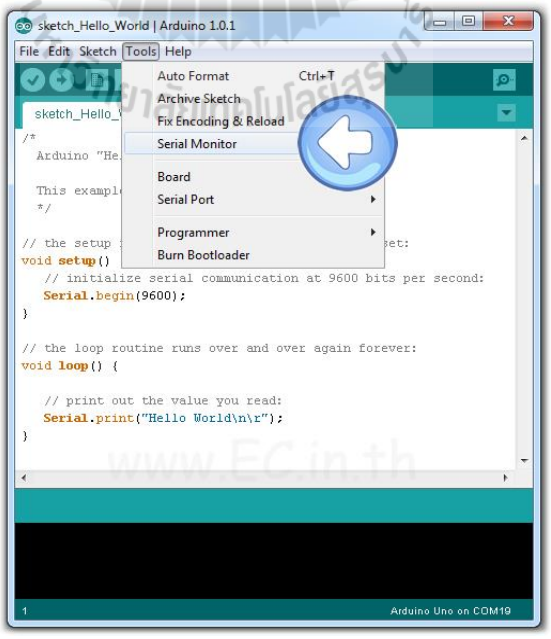
รูปที่ 2.36 เลือก Serial Port

9. โหลดโปรแกรมเข้าบอร์ด Arduino UNO โดยไปที่ File->Upload



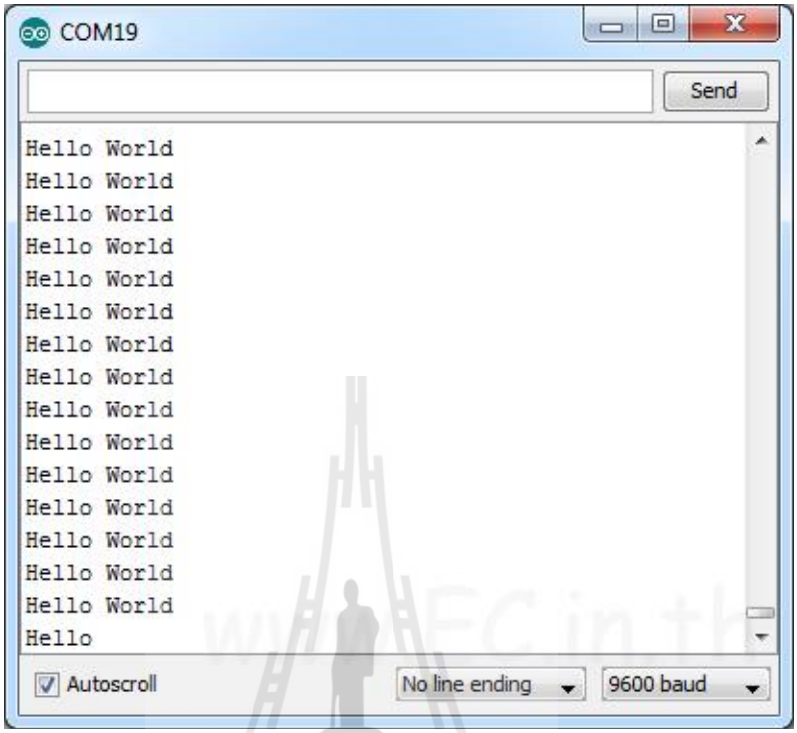
รูปที่ 2.37 . โหลดโปรแกรมเข้าบอร์ด

10. จากนั้นเปิด Serial Monitor ของ Arduino IDE โดยไปที่ Tools-> Serial Monitor



รูปที่ 2.38 Serial Monitor

11. เมื่อเปิด Serial Monitor จะได้ข้อความดังรูป

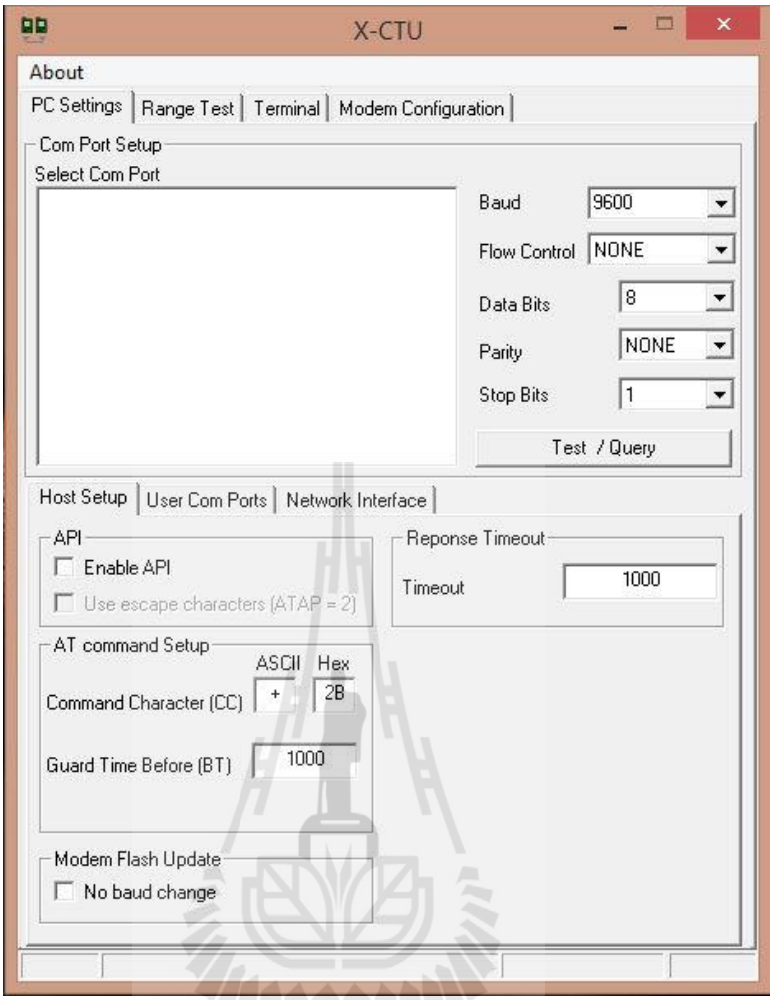


รูปที่ 2.39 เมื่อเปิด Serial Monitor

2.5 Zigbee and Xbee BASIC ตอน การใช้งาน Xbee เบื้องต้น

รู้จักX-CTU

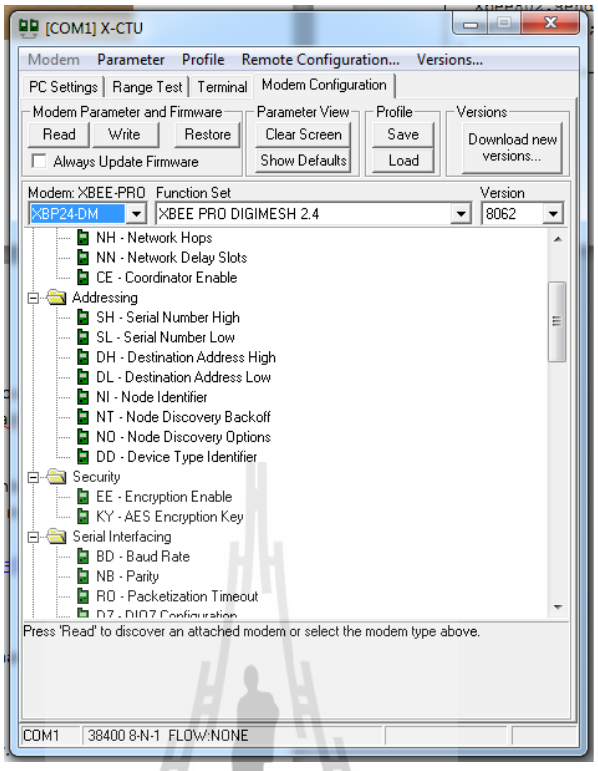
X-CTU เป็น software interface บนคอมพิวเตอร์ที่จะช่วยในการ update firmware หรือ ทดสอบการใช้งาน หรือ ปรับ parameter กับ Xbee โดยสามารถ download software มาได้ฟรี จาก Digi (X-CTU Software) สำหรับการใช้งาน สามารถอ่านจากคู่มือ X-CTU Configuration & Test Utility Software User Guide หลังจากที่ Download ตัว Software มาแล้ว การ Install จะมีการ Download Firmware ล่าสุดจาก Digi ผ่าน internet



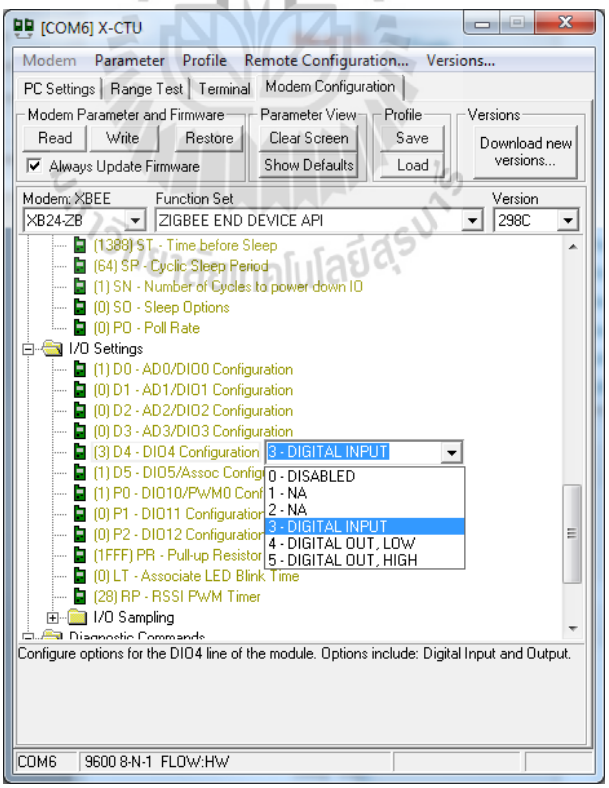
รูปที่ 2.40 หน้าตา Software X-CTU ที่ใช้ร่วมกับ Xbee (Free Download)

รู้จัก Firmware

สำหรับ Xbee นั้น ในแต่ละรุ่นที่ซื้อ จะมี firmware ที่โปรแกรมมาแล้วจากทางโรงงาน ซึ่งสามารถดึงค่ามาดูได้โดยไปที่ Tab Modem Configuration แล้วกดปุ่ม Read
หากซื้อ Xbee Pro Series 1 มา จะต้องเลือก firmware version ที่ใช้ร่วมกัน คือ 1081, 1082, 1083, 1084 , ... ซึ่ง firmware ล่าสุดจะถูก set ให้โดยอัตโนมัติ หากเลือก Xbee รุ่น จะต้องเลือก group XB24xxxx เป็นต้นไป ส่วนถ้าเลือกรุ่นPro มา ต้องเลือก XBP24xxxx สำหรับ ในแต่ละ series นั้น จะลงท้ายไม่เหมือนกัน เช่น series2 ZB จะเป็น XB24-ZB



รูปที่ 2.41 เลือก modem



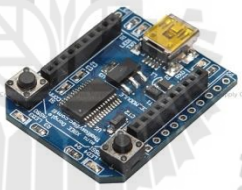
รูปที่ 2.42 ตั้ง parameter

สามารถ Set Parameter ใน Tab Modem Configuration แล้ว กด write Firmware สามารถทดลองใช้งานง่าย ๆ ด้วยการใช้อุปกรณ์ Dongle และ Xbee 1 คู่ สร้างเครือข่ายแบบ Point-to-Point ร่วมกับ X-CTU เพื่อกำหนด Parameter ให้กับ Xbee ผ่าน firmware และ X-CTU

การตั้งค่าอุปกรณ์

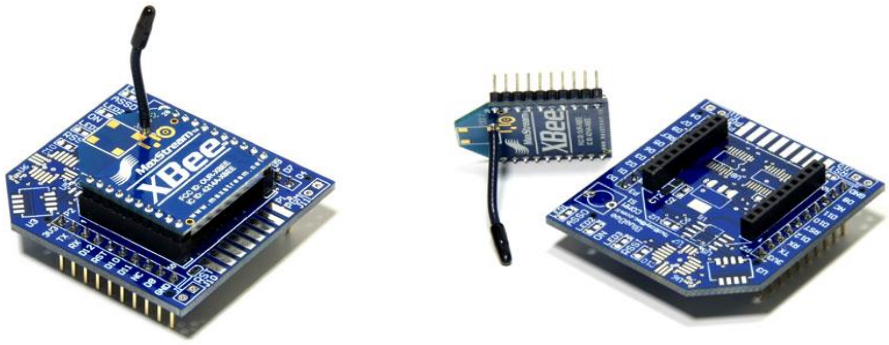
การทดสอบจะใช้อุปกรณ์ดังรูปด้านล่างคือประกอบด้วย Xbee Pro Series 1 และ Mini XbeeUSB DongleV2 สำหรับเสียบอุปกรณ์กับคอมพิวเตอร์ ทำการเปิด X-CTU 2 หน้าต่าง แล้วไปที่หน้า Tab Modem Configuration แล้วกด Read จะพบว่า X-CTU จะทำการ LoadFirmware ของ Xbee Pro Series1 ออกมาซึ่งจะเป็นค่าที่Setไว้แบบDefault

อุปกรณ์ที่ใช้งานในการทดสอบ



รูปที่ 2.43 Mini Xbee USB Dongle V2

MiniXbeeUSBDongleV2 (ETEE028A)
เป็น Tool ที่ใช้สำหรับทำงานร่วมกับ สำหรับ update firmware หรือ การ Config Parameter หรือ Update Firmware ผ่านคอมพิวเตอร์ ร่วมกับ Software X-CTU สามารถต่อ Xbee ร่วมกับ Mini Xbee USB Dongle ผ่าน USB Port ได้โดยตรง ซึ่งคอมพิวเตอร์จะมองเป็น Com Port (Serial UART) เนื่องจากใช้ IC FT232RL ซึ่งเป็น USB to Serial ในบอร์ดเดียว ไม่ต้องต่ออุปกรณ์ใด ๆ เพิ่มเติม สามารถใช้ร่วมกับ Xbee ได้ทุกรุ่น



รูปที่ 2.44 Xbee Convert PIN to 2.54 Pitch

Xbee Convert PIN to 2.54 Pitch (ETEE019A)

PCB สำหรับแปลงขาของ Xbee Series 1 และ Series 2 ให้สามารถเสียบกับ Protoboard ได้โดยตรง

รูปที่ 2.45 Bluebee - Xbee Breakout board

Bluebee - Xbee Breakout board (ETEE017)

BlueBee เป็น Xbee Breadboard ที่สามารถต่อสายใช้งานได้, สามารถเพิ่ม sensor ที่ต้องการได้เอง ตามที่ BlueBee จัด Footprint ไว้ให้ เหมาะกับการต่อโดยตรงเข้ากับ MCU

รูปที่ 2.46 BlueBee Dongle

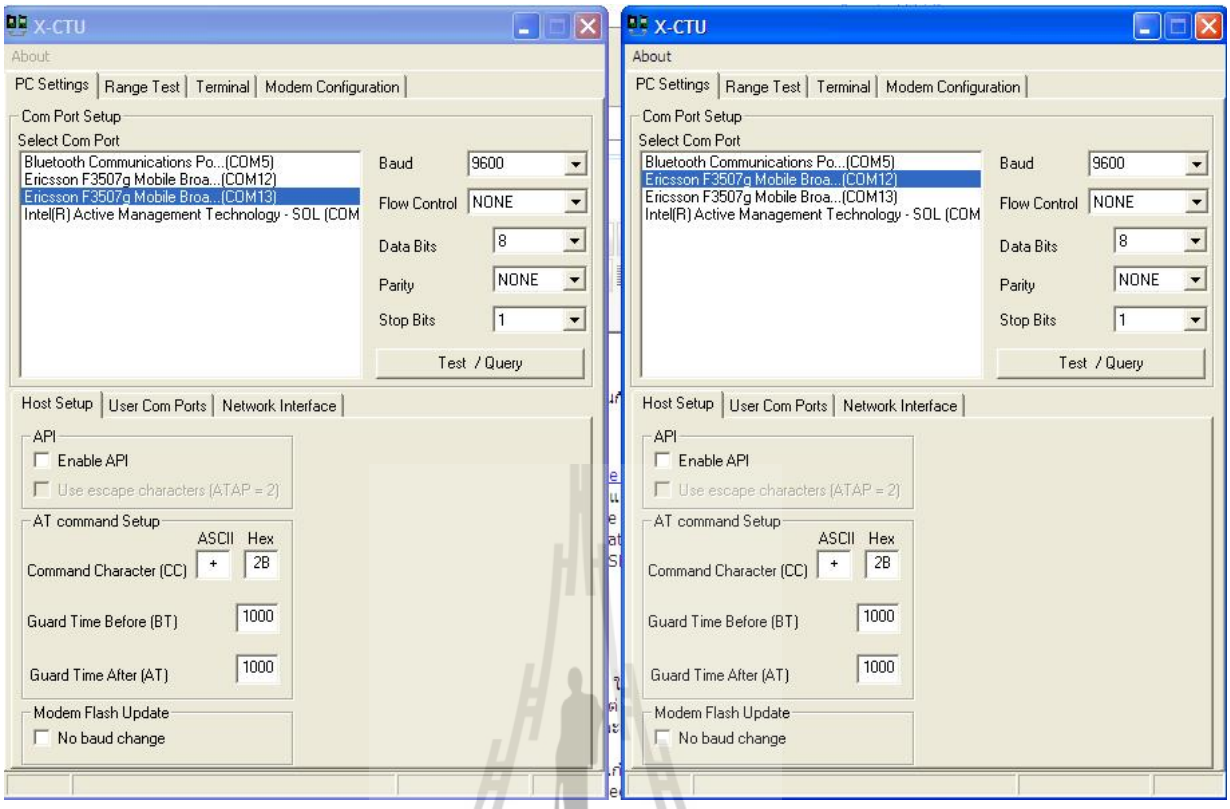
BlueBee Dongle (ETEE017D)

เป็น BlueBee - Xbee Breadboard ที่ใช้งานเป็น Xbee Dongle โดยต่อใช้งานร่วมกันกับ USB to Serial สำหรับ update firmware หรือ การ config parameter ผ่าน คอมพิวเตอร์

ทดสอบการใช้งาน Xbee และ Software X-CTU

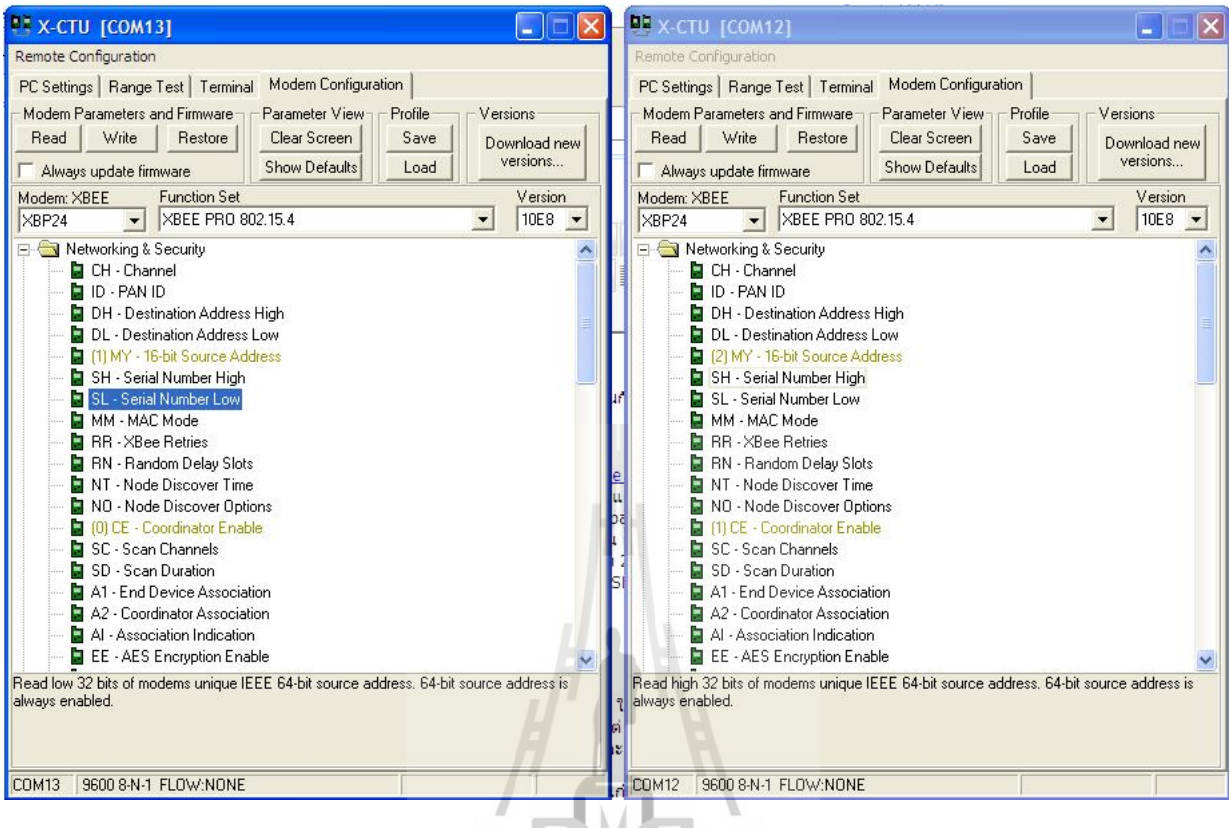
1. ทำการ set parameter ให้ติดต่อกันแบบ Point to Point เมื่อทำการ

ต่อ Xbee Pro Series1 และ Mini Xbee USB Dongle V2 เข้ากับ PC ทั้ง 2 ชุดเรียบร้อยแล้วให้เปิดโปรแกรม X-CTU ขึ้น 2 ชุดเช่นกัน แล้วทำการเลือก COM Port (UART) ในแต่ละชุดให้ถูกต้อง



รูปที่ 2.47 set parameter ให้ติดต่อกันแบบ Point to Point

2. ที่โปรแกรม X-CTU ให้เลือก Tab "Modem Configuration" แล้วทำการ set parameter โดยให้ฝั่งหนึ่งเป็น End Device ด้วยการ Set Parameter ในหมวด Networking & Security >> CE=0 และ ค่า MY = 1 และ ฝั่งอีกหนึ่ง เป็น Coordinator ด้วยการ Set Parameter ในหมวด Networking & Security >> (CE=1) และ ค่า MY = 2 ทั้งนี้ ได้ set baud rate ที่ตัว Xbee ทั้ง 2 ตัวที่ 9600 bps (BD=3)



รูปที่ 2.47 set parameter

3. ทำการ set ให้ parameter DH และ DL ของแต่ละฝั่งให้มีค่าเท่ากับ SH และ SL ของฝั่งตรงข้าม โดย SH SL เป็นค่า address ที่เปลี่ยนไม่ได้ (Read Only) เป็นค่าที่ใส่มาจากโรงงาน

วิธีการดูว่า Xbee ของมีค่า SH และ SL เท่าไหร่ สามารถดูได้ 2 วิธีคือ ให้กด Read เพื่อ Load ค่าของ Xbee เข้ามาจะทำให้ Click ดูค่า SH และ SL ได้และวิธีที่ 2 คือดูจากทางด้านใต้ของตัว Xbee จะมี Sticker ติดหมายเลข SH , SL มาให้แล้ว

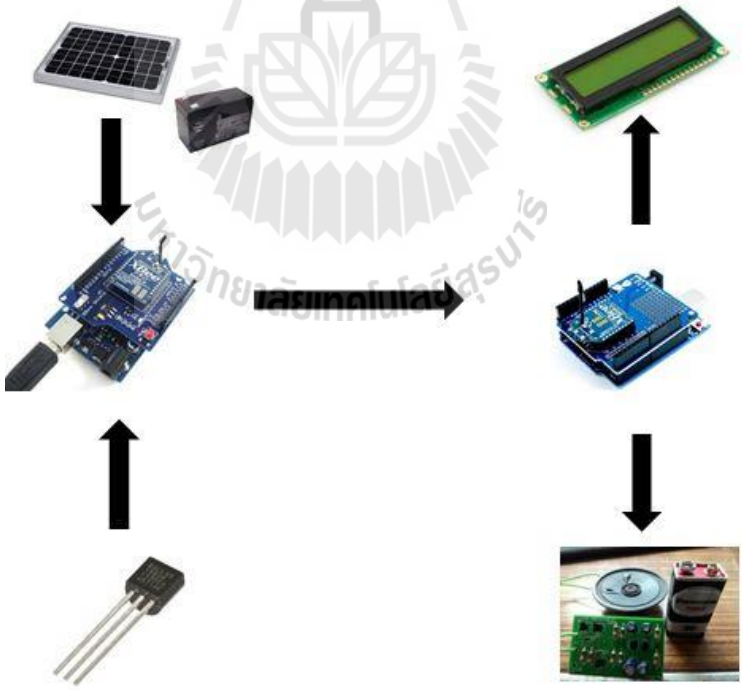
บทที่ 3
หลักการทำงานของอุปกรณ์และโปรแกรม ในการใช้งานระบบตรวจวัดอุณหภูมิอัตโนมัติโดยผ่านเครือข่าย xbee โดยมีแหล่งจ่ายเป็นโซล่าเซลล์

3.1 บทนำ

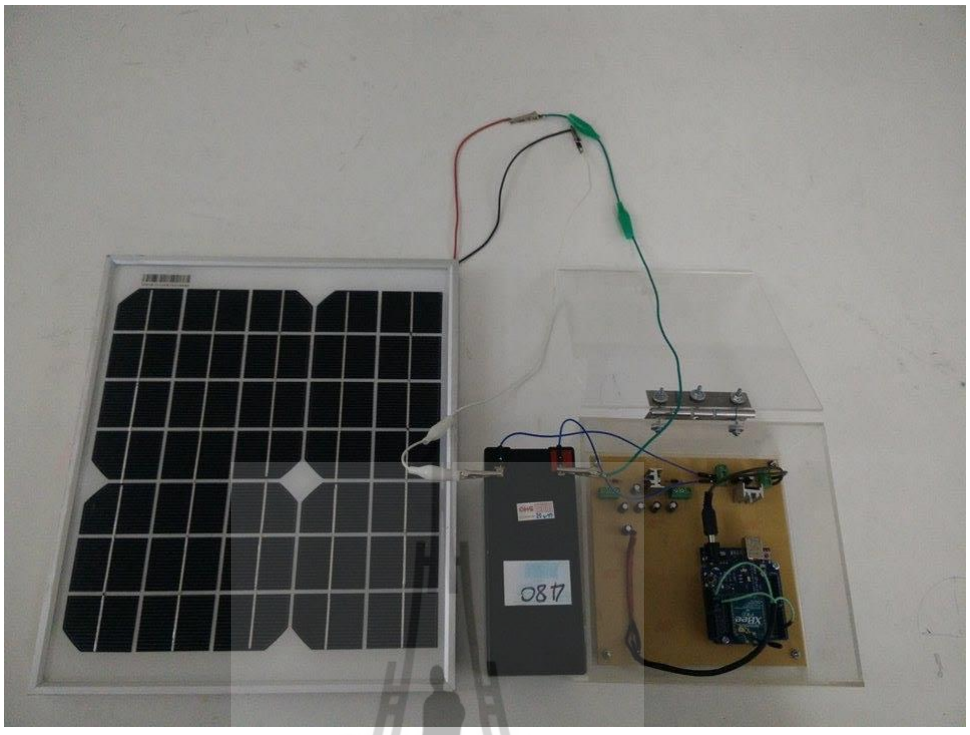
เนื้อหาในบทนี้จะกล่าวถึงโครงสร้างของระบบตรวจจับวัดอุณหภูมิของเซนเซอร์ หน้าที่และการใช้งานส่วนประกอบต่างๆ การเขียนโปรแกรมและใช้โปรแกรมควบคุมการทำงาน และการเชื่อมต่อส่วนประกอบต่างๆ ให้ทำงานร่วมกัน

3.2 โครงสร้างของระบบตรวจวัดอุณหภูมิอัตโนมัติโดยผ่านเครือข่าย xbeeโดยมีแหล่งจ่ายเป็นโซล่าเซลล์

ระบบตรวจจับวัดอุณหภูมิของเซนเซอร์นี้ มีโครงสร้างในการทำงานซึ่งประกอบไปด้วย เซนเซอร์วัดอุณหภูมิ ds18b20 บอร์ดไมโครคอนโทรลเลอร์ Arduino (รุ่น Arduino Uno R3) โปรแกรม Arduino IDE ที่ใช้เขียนโปรแกรมลงในบอร์ดไมโครคอนโทรลเลอร์ Arduino และโปรแกรม X-ctu ใช้ในการเขียนโปรแกรมของ Xbee อุปกรณ์แสดงผลในระบบตรวจจับวัดอุณหภูมิของเซนเซอร์ ประกอบไปด้วยจอแสดงผล LCD วงจรเสียง (Alarm) และแหล่งจ่ายภาคส่งที่ใช้โซล่าเซลล์

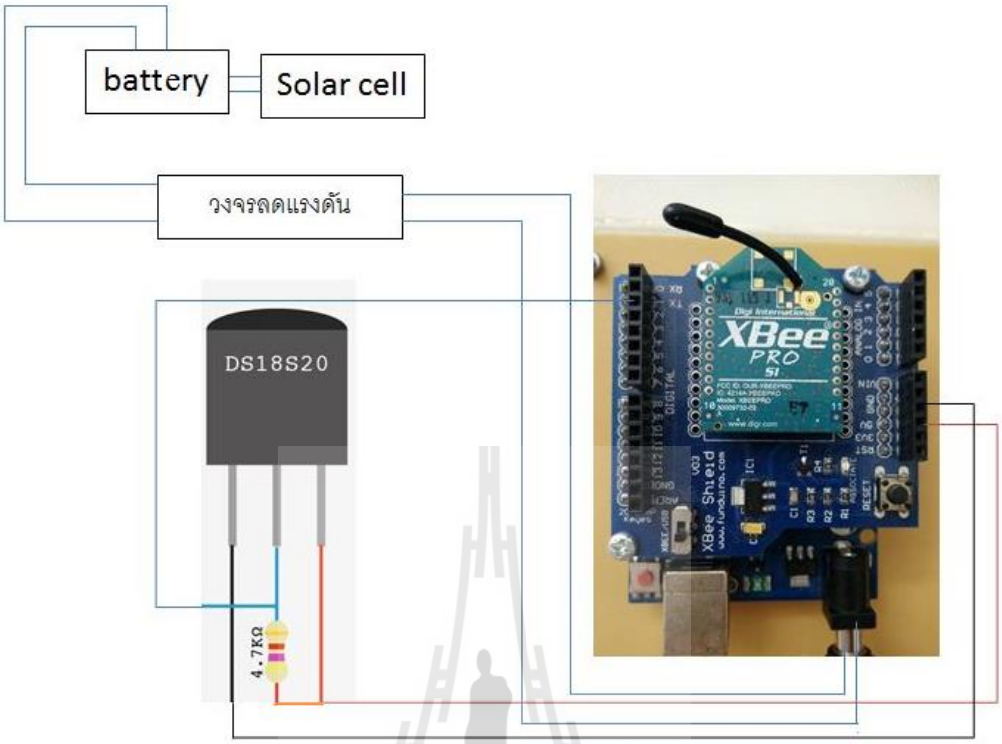


รูปที่ 3.1 โครงสร้างรวมของระบบ

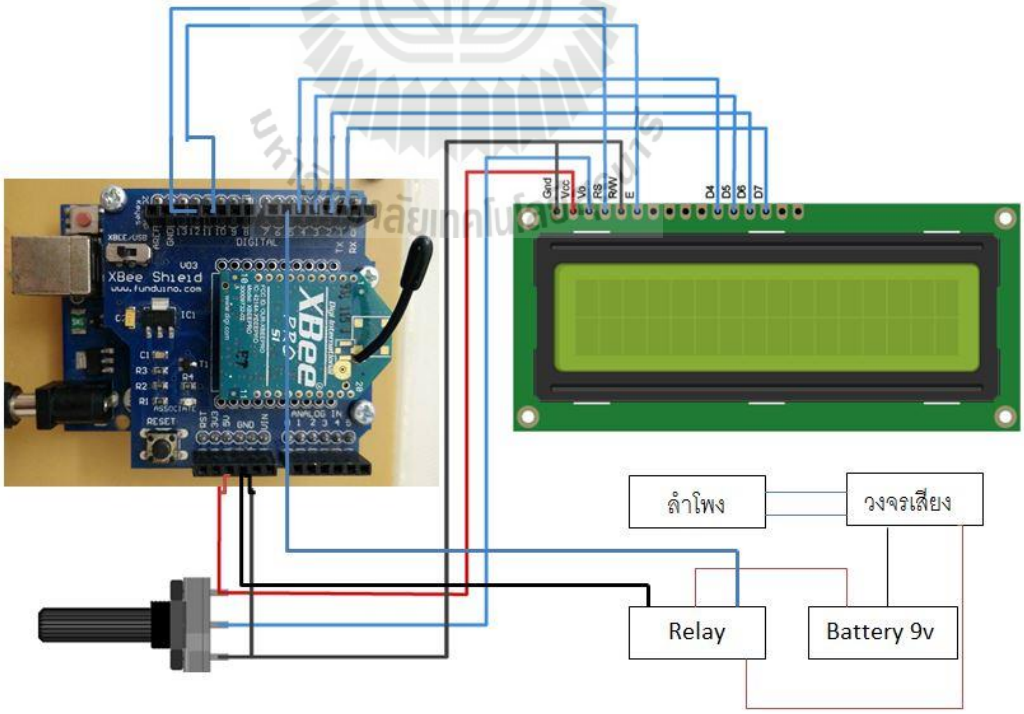


รูปที่ 3.2 โครงสร้างระบบภาคส่ง

รูปที่ 3.3 โครงสร้างระบบภาครับ



รูปที่ 3.4 การเชื่อมต่อวงจรภาคส่ง



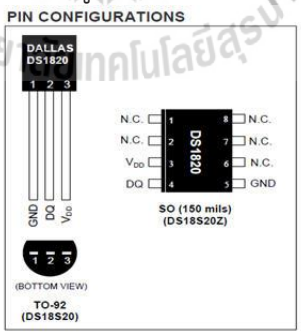
รูปที่ 3.5 การเชื่อมต่อวงจรภาครับ

3.3 อุปกรณ์ที่ใช้ในระบบตรวจวัดอุณหภูมิอัตโนมัติโดยผ่านเครือข่าย xbeeโดยมีแหล่งจ่ายเป็นโซล่าเซลล์

3.3.1 เซนเซอร์วัดอุณหูมิ DS18B20

ไอซี DS1820 เป็นไอซีที่มีระบบการสื่อสารข้อมูลอนุกรมแบบหนึ่งสายซึ่งถือได้ว่าเป็นระบบที่มีความชาญฉลาด และใช้จำนวนสายสัญญาณเพียง 1 เส้นเท่านั้น โดยไม่ต้องมีสายสัญญาณนาฬิกามาควบคุมจังหวะการถ่ายทอดข้อมูลเหมือนกับระบบสื่อสารข้อมูลอนุกรมในแบบอื่น สายข้อมูลจะทำหน้าที่เสมือนเป็นสายสัญญาณนาฬิกาในตัว ส่วนค่าของข้อมูลจะพิจารณาจากลักษณะของรูปสัญญาณที่ปรากฏบนสายสัญญาณในแต่ละช่องของเวลาซึ่งเรียกว่า ไทม์สล็อต (Time Slot) โดยคาบเวลาต่ำสุดและสูงสุดของสถานะต่าง ๆ ในการสื่อสารข้อมูลในแต่ละไทม์สล็อตมีการกำหนดขอบเขตไว้อย่างชัดเจนการถ่ายทอดข้อมูลจะเกิดขึ้นในแต่ละไทม์สล็อตนั้น รูปแบบการถ่ายทอด ข้อมูลจะเป็นแบบอะซิงโครนัสในระดับบิต ไม่มีการกำหนดความยาวของข้อมูลเป็นระดับไบต์ระบบสื่อสารแบบนี้เหมาะที่จะใช้ในการสื่อสารข้อมูลระหว่างไอซีแผงวงจรเดียวกัน

เหตุผลที่เลือก DS18S20 เนื่องจากสิ่งที่เราต้องการในการวัดอุณหภูมินี้เราต้องการระบบบัสที่สามารถเดินสายได้ในระยะไกลซึ่งเหมาะสำหรับระบบบัสแบบ 1-Wire ดูจะเหมาะกว่าในแง่ของความสะดวกในการใช้งานระยะห่างของแต่ละจุดที่วัดความแม่นยำของค่าที่วัดได้ใช้ไอซีเบอร์ DS18S20 ซึ่งมีระดับความผิดพลาดที่ 0.5องศาเซลเซียส DS18S20 เป็น IC วัดอุณหภูมิแบบดิจิตอลของ Dallas Semiconductor สามารถวัดอุณหภูมิเป็นหน่วยองศา C ในช่วง -55C ถึง 125 C ที่ความละเอียด 9-12 บิต และมีความแม่นยำอยู่ที่ 0.5C ในช่วง -10C ถึง 85 C ในกรณีที่เป็นตัวถังแบบ TO-92 นั้นจะมีโครงสร้าง และขาดังแสดงในรูปที่ 3.6

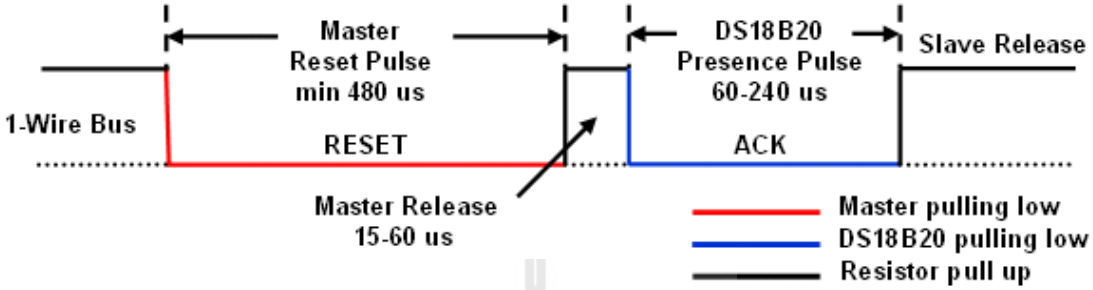


PIN	SYMBOL	Description
1	GND	Ground
2	DQ	Data Input/ Output pin
3	Vdd	Optional Vdd pin

รูปที่ 3.6 วงจรพื้นฐานของ เซนเซอร์ DS18B20

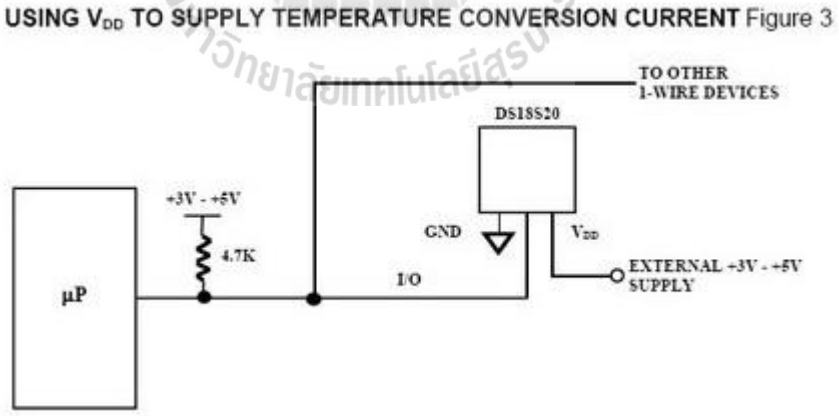
หลักการทำงานของเซนเซอร์วัดอุณหภูมิ DS18S20

ในกระบวนการเริ่มต้นการสื่อสารแบบ 1-wire ทั้งหมดนั้น อุปกรณ์ Master การสื่อสารด้วยการสร้าง Reset pulse ก่อน เมื่ออุปกรณ์ Slave ได้รับ Reset pulse ก็จะสร้าง Presence pulse เพื่อตอบรับการขอเริ่มการสื่อสารนั้น ซึ่งมีรายละเอียดของช่วงเวลาต่าง ๆ ดังแสดง



รูปที่ 3.7 การเริ่มติดต่อสื่อสารแบบ 1-wire ด้วย Reset pulse และ Presence pulse

จากรูปที่ 3.7 การสื่อสารแบบ 1-wire เป็นระบบบัสข้อมูลแบบ Half-duplex นั้นคือสามารถสื่อสารได้ 2 ทิศทาง แต่ไม่สามารถรับ และส่งข้อมูลพร้อมกันในช่วงเวลาเดียวกันได้ ระบบบัสมีการทำงานเป็นแบบ Master/Slave โดยอุปกรณ์ Master จะเป็นตัวควบคุมสถานะ และจังหวะการรับส่งของบัสข้อมูล ในขณะที่อุปกรณ์ Slave จะทำงานตามการควบคุมของอุปกรณ์ Master เท่านั้น ในการใช้งานบัสแบบ 1 wire นี้ สายสัญญาณข้อมูล DQ จะต้องมีสภาวะปกติที่ลอจิกสูง สามารถทำได้โดยการต่อตัวต้านทานประมาณ 5 กิโลโอห์ม พูลอัปไว้กับไฟเลี้ยง หรือในกรณีที่ใช้บัสแบบ 1 wire ต่อร่วมกับอุปกรณ์ DS18S20 หลายตัว ก็สามารถทำได้ดังแสดงในรูปที่ 3.8

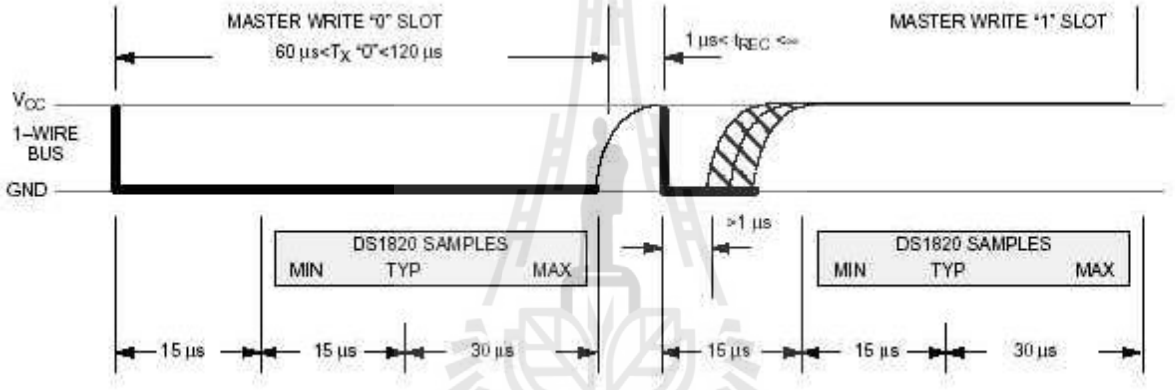


รูปที่3.8 การต่อใช้งาน DS18B20

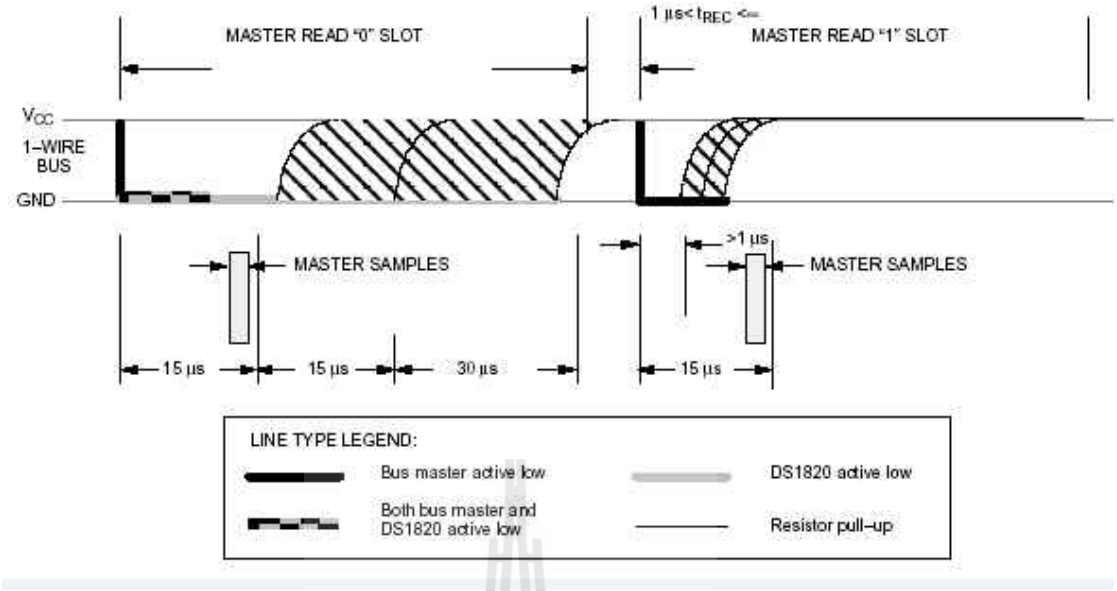
ในการเขียนข้อมูลแบ่งออกเป็น 2 ชนิดคือการเขียนข้อมูล “0” และการเขียนข้อมูล “1” ดังแสดงในรูปที่ 3.9 การเขียนข้อมูลลง DS18S20 ต้องใช้ช่วงเวลาของไทม์สล๊อตอย่างต่ำ 60 usec และต้องมีช่วงเวลาระหว่างไทม์สล๊อตอย่างต่ำ 1 usec

การเขียนข้อมูลทั้ง 2 ชนิด เริ่มแรกอุปกรณ์ Master ต้องดึงสัญญาณบนบัส 1-wire ลงมาให้อยู่ในสถานะลอจิกต่ำก่อน ในกรณีที่ต้องการเขียนข้อมูล “0” ลงใน DS18S20 อุปกรณ์ Master ต้องดึงสัญญาณบนบัสให้เป็นลอจิกต่ำต่อ จนกว่าจะครบช่วงเวลาไทม์สล๊อต (อย่างต่ำ 60 usec) ส่วนในกรณีที่ต้องการเขียนข้อมูล “1” ลง DS18S20 อุปกรณ์ Master ต้องปล่อยบัส เพื่อให้บัสกลับไปอยู่ในสถานะลอจิกสูงก่อนการ Sampling ของ DS18S20 ซึ่งจะอยู่ในช่วง 15 usec-60 usec หลังจากที่อุปกรณ์ Master ดึงสัญญาณบัส 1-wire ลงมาในการอ่านค่าภายใน SRAM ของ DS18S20 สามารถทำได้ก็ต่อเมื่ออุปกรณ์ Master ได้เขียนข้อมูลเพื่อขอทำการอ่านค่าใน SRAM (Read Scratchpad) ซึ่งมีค่าเป็น 0xBE ลงไปที่ DS18S20 เสียก่อน จากนั้นจึงเริ่มอ่านข้อมูลจากบัส 1-wire โดยไทม์สล๊อตของการอ่านต้องมีช่วงเวลาอย่างต่ำ 60 usec และต้องมีช่วงเวลาระหว่างไทม์สล๊อตอย่างต่ำ 1 usec ดังแสดงในรูปที่ 3.9

รูปที่ 3.9 การเขียนข้อมูลจาก DS18B20



การอ่านข้อมูลจากบัส 1-wire เริ่มแรกอุปกรณ์ Master จะต้องดึงบัส 1-wire ลงให้อยู่ในสถานะลอจิกต่ำเป็นช่วงเวลาอย่างน้อย 1 usec จากนั้นจึงค่อยปล่อยบัส ในกรณีที่ DS18S20 ส่งข้อมูล “0” DS18S20 จะดึงบัสให้เป็นลอจิกต่ำจนจนกว่าจะสิ้นสุดไทม์สล๊อตถึงจึงจะปล่อยบัสให้กลับไปอยู่ในสถานะลอจิกสูง ส่วนในกรณีที่ DS18S20 ส่งข้อมูล “1” DS18S20 จะปล่อยบัสให้อยู่ในสภานะลอจิกสูงตลอด ในการ Sample เพื่อรับข้อมูลจาก DS18S20 ควรทำภายใน 15 usec หลังจากจุดเริ่มของไทม์สล๊อต



รูปที่ 3.10 การหาข้อมูลจาก DS18B20

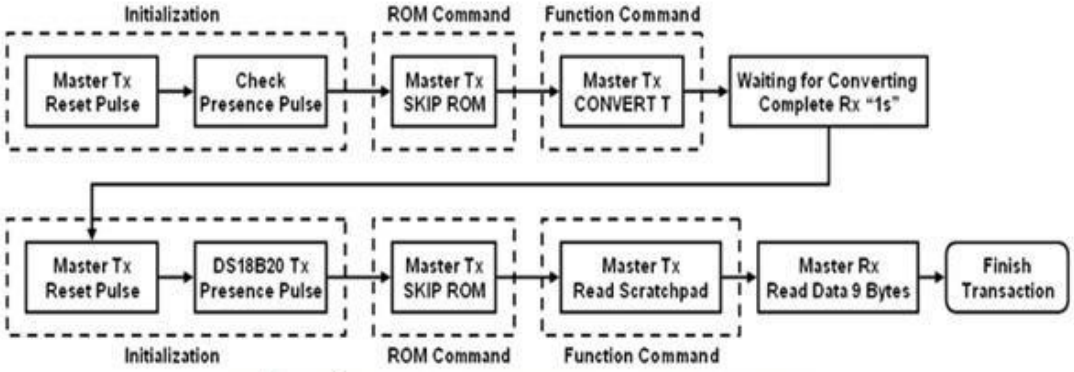
ขั้นตอนการใช้งาน DS18S20 มี 3 ขั้นตอนด้วยกันคือ

1.Initialization

2.ROMCommand

3.DS18S20FunctionCommand

ขั้นตอนการอ่านค่าอุณหภูมิจาก DS18S20นั้น แบ่งออกเป็น 2 ขั้นตอนใหญ่ ๆ คือ Converting Temperature ซึ่งเป็นการแปลงอุณหภูมิให้อยู่ในรูปข้อมูลดิจิตอลเก็บไว้ใน Scratchpad และ Read Scratchpad ซึ่งเป็นการอ่านข้อมูลที่เก็บใน Scratchpad นั้นออกมา ดังแสดงในรูปที่ 11 จะเห็นได้ว่าไมโครคอนโทรเลอร์ซึ่งเป็นอุปกรณ์ Master จะต้องทำการ Initialization จากนั้นจึงค่อยคำสั่ง SKIP ROM [CCh] ซึ่งเป็น ROM Command ที่ใช้ในกรณีที่ต่อใช้งาน DS18S20 เพียงตัวเดียวจากนั้นจึงค่อยส่ง Function Command CONVERT T [44h] เพื่อสั่งให้ DS18S20 ทำการแปลงอุณหภูมิให้อยู่ในรูปแบบดิจิตอลและเก็บไว้ใน Scratchpad กระบวนการแปลงนี้จะใช้เวลาประมาณ 750ms ซึ่งเราสามารถตรวจสอบสถานการณ์แปลงข้อมูลได้จากการเช็คสถานะของบัส 1-wire ในกรณีที่ DS18S20 ยังทำการแปลงข้อมูลอยู่บัส 1-wire จะมีสถานะเป็นลอจิกต่ำ แต่ถ้า DS18S20 ได้ทำการแปลงข้อมูลเสร็จเรียบร้อยแล้วบัส 1-wire จะกลับมาอยู่ในสถานะปกติคือลอจิกสูง



รูปที่ 3.11 ขั้นตอนการแปลง และอ่านอุณหภูมิจาก DS18S20

เมื่อ DS18S20 แปลงข้อมูลอุณหภูมิเป็นดิจิตอลเก็บไว้ใน Scratchpad แล้ว ไมโครคอนโทรลเลอร์สามารถอ่านข้อมูลภายใน Scratchpad นั้นโดยทำการ Initialization บัส 1-wire อีกครั้ง จากนั้นจึงส่งคำสั่ง SKIP ROM ซึ่งเป็น ROM Command และส่งคำสั่ง READ SCRATCHPAD ซึ่งเป็น Function Command เพื่อขออ่านข้อมูล Scratchpad ภายใน DS18S20 จากนั้นไมโครคอนโทรลเลอร์จึงค่อยทำการอ่านข้อมูลภายใน Scratchpad ของ DS18S20 มาทีละไบต์จนครบ 9 ไบต์ และแสดงผลข้อมูลอุณหภูมิที่อยู่ซึ่งเป็นข้อมูลในไบต์แรก และไบต์ 2 ที่อ่านมาจาก Scratchpad นั้นออกมาทางพอร์ตอนุกรม

3.3.2 บอร์ดไมโครคอนโทรลเลอร์ Arduino รุ่น Arduino UNO R3

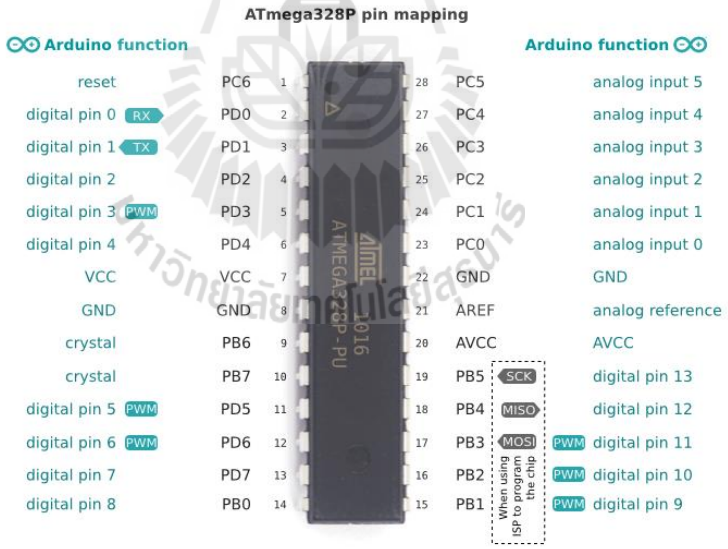
รูปที่ 3.12 บอร์ดไมโครคอนโทรเลอร์ Aduino UNO R3

Arduino UNO R3 เป็น ไมโครคอนโทรลเลอร์ board ที่ใช้ ATmega328 เป็น MCU หลัก ซึ่งตัวนี้จะมีขา Digital 14 ขา อินพุท/เอาท์พุท (สามารถทำเป็น PWM ได้ถึง 6 ขา) และมีขา Analog อินพุทได้อีก 6 ขา, รันที่ความถี่ 16 MHz มี USB Connector และ Power Jack DC ซึ่ง Concept

ของ Arduino Board นี้ทำมาเพื่อความสะดวก ง่ายในการเชื่อมต่อเข้ากับคอมพิวเตอร์ สามารถต่อ USB เข้ากับช่องคอมพอร์ตก็สามารถ Run โปรแกรมที่ Board ได้ มีรายละเอียดโดยสรุปดังนี้

บอร์ด Arduino UNO R3 มีคุณสมบัติดังนี้

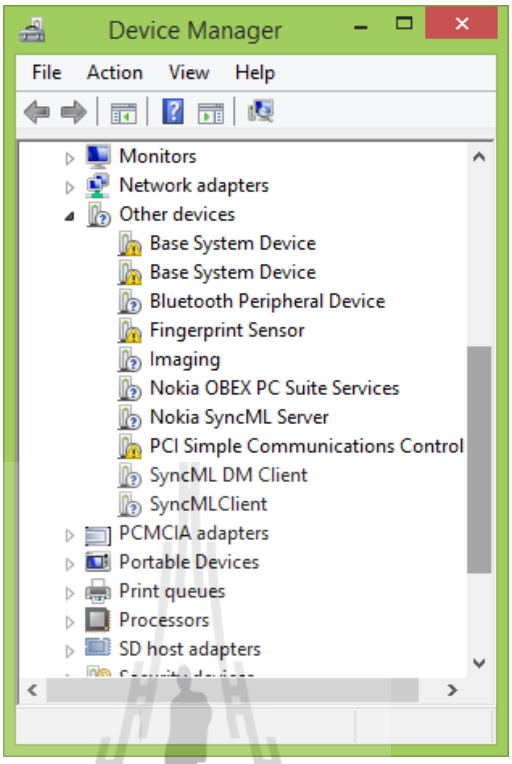
ไมโครคอนโทรลเลอร์	ATmega328
Operating Voltage	5V
Input Voltage (recommended)	7-12V
Input Voltage (limits)	6-20V
Digital I/O Pins	14
Analog Input Pins	6
DC Current per I/O Pin	40 mA
DC Current for 3.3V Pin	50 mA
Flash Memory 32 KB (ATmega328) of which 0.5 KB used by bootloader	
SRAM	2 KB (ATmega328)
EEPROM	1 KB (ATmega328)
Clock Speed	16 MHz



รูปที่ 3.13 โครงสร้างของบอร์ด Arduino UNO R3

การใช้งานบอร์ด Arduino UNO R3

1.ก่อนการติดตั้งเข้าไปดูใน Device Manager ว่ามีรายการอะไรอยู่บ้าง

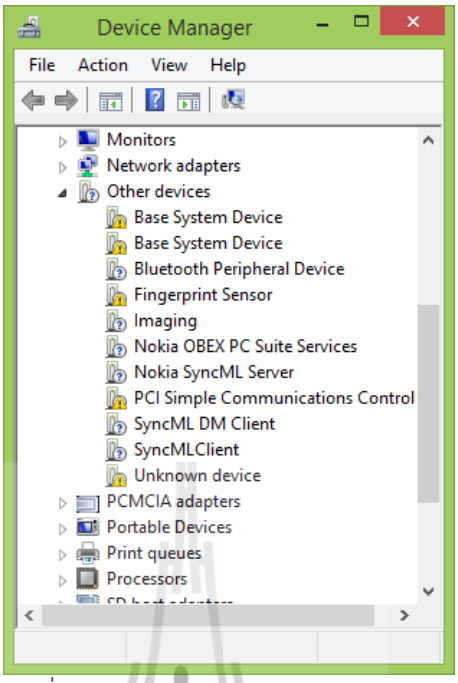


รูปที่ 3.14 การใช้งานบอร์ด Arduino UNO R3

2. เสียบสาย USB (ซึ่งต่ออยู่กับบอร์ด Arduino) เข้าเครื่องคอมพิวเตอร์

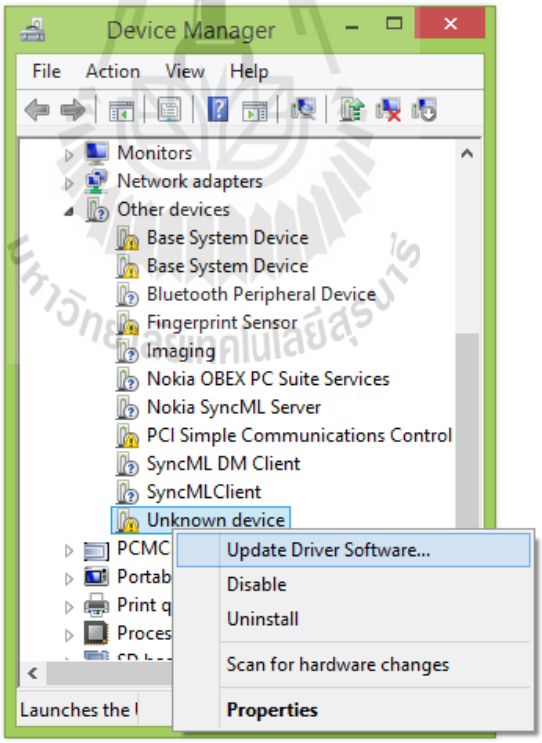
รูปที่ 3.15 เสียบสาย USB (ซึ่งต่ออยู่กับบอร์ด Arduino)

จะมีรายการปรากฏขึ้นเพิ่มเติม (ซึ่งตอนที่ไม่ได้เสียบ USB ไม่มีรายการนี้) ดังรูป (Unknown Device)



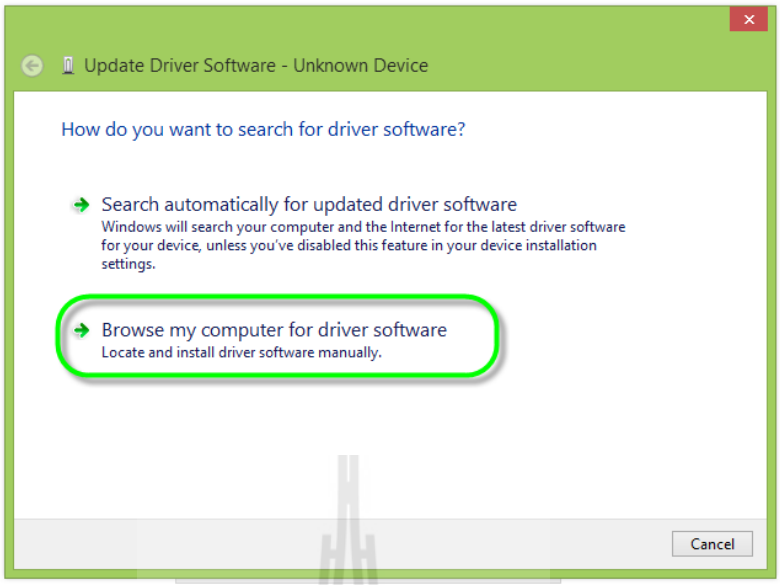
รูปที่ 3.16 ดู driver Arduino UNO R3

3. คลิกขวาที่รายการดังกล่าว เลือก Update Driver Software...



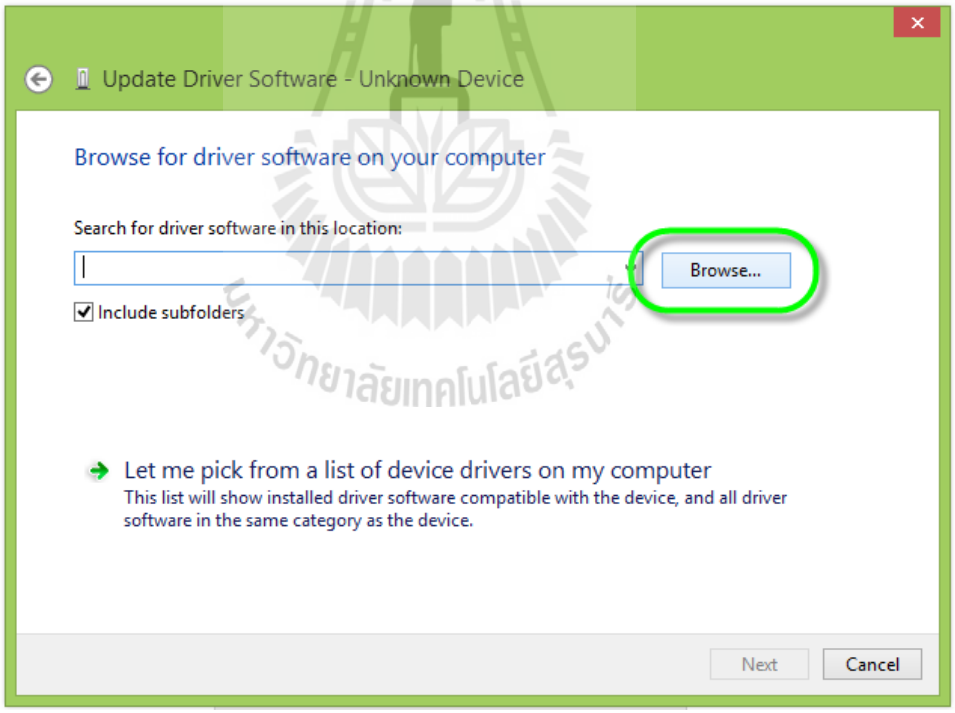
รูปที่ 3.17 Update Driver

4. คลิก Browse my computer for driver software



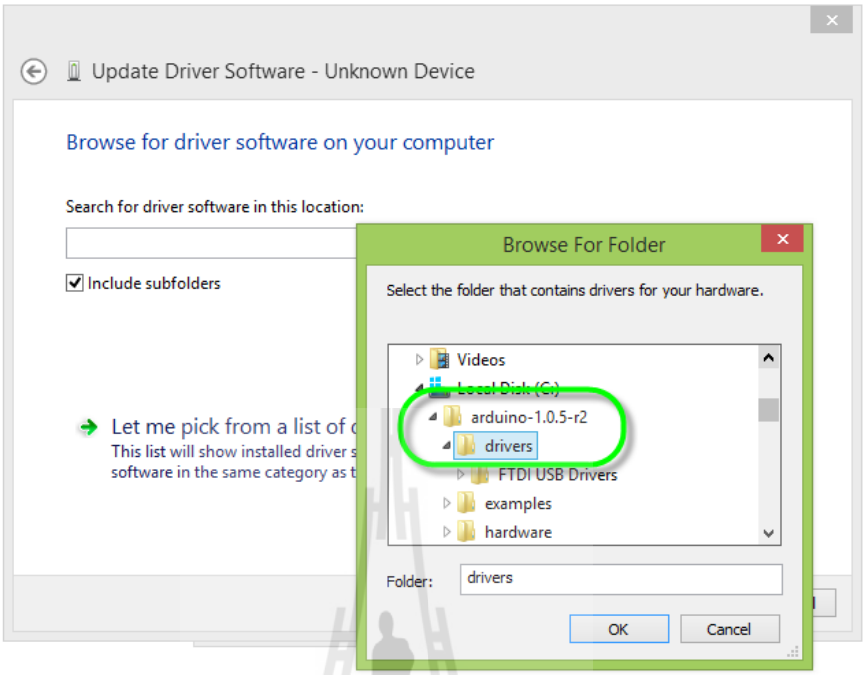
รูปที่ 3.18 update driver software

5. คลิก Browse เพื่อหาตำแหน่งเก็บไฟล์ Driver

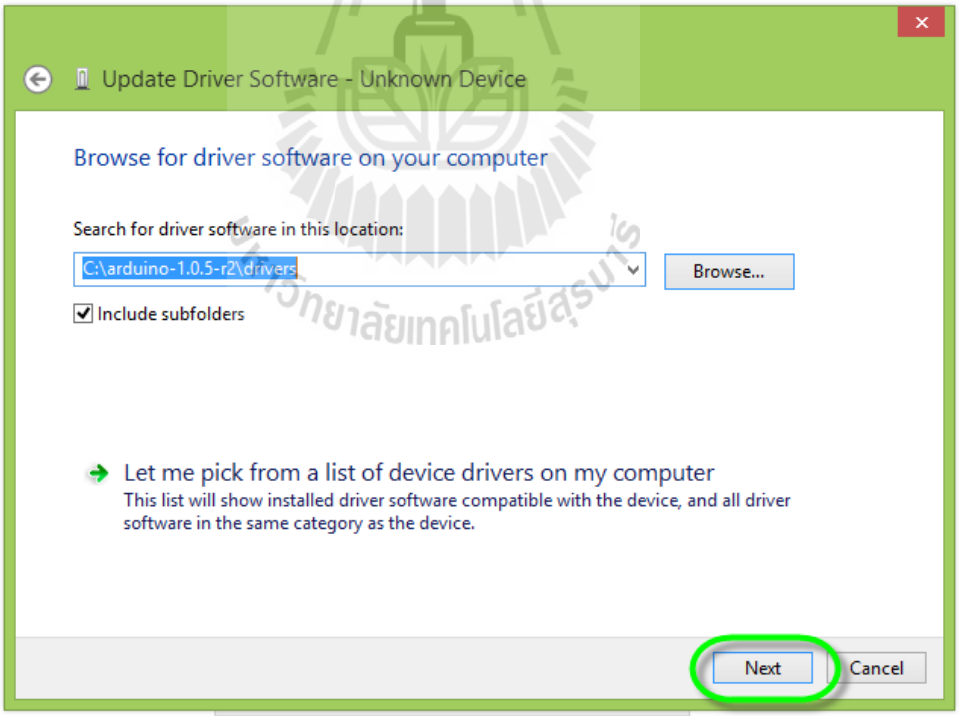


รูปที่ 3.19 Browse เพื่อหาตำแหน่งเก็บไฟล์ Driver

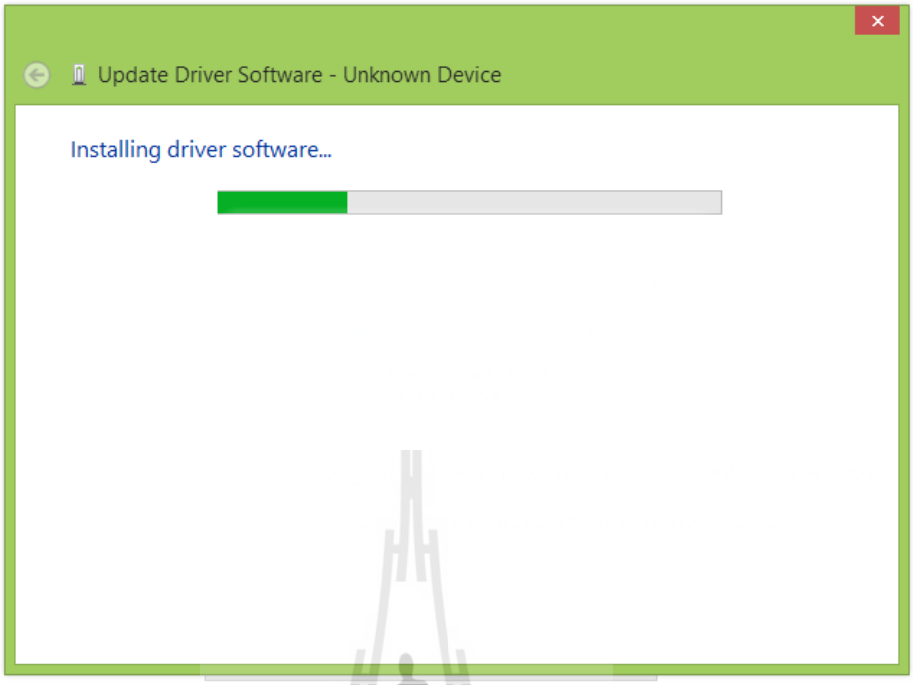
6. ตำแหน่ง Driver จะอยู่ในโฟลเดอร์โปรแกรม Arduino ดังรูป จากนั้นคลิก Next



รูปที่ 3.20 ตำแหน่ง Driver

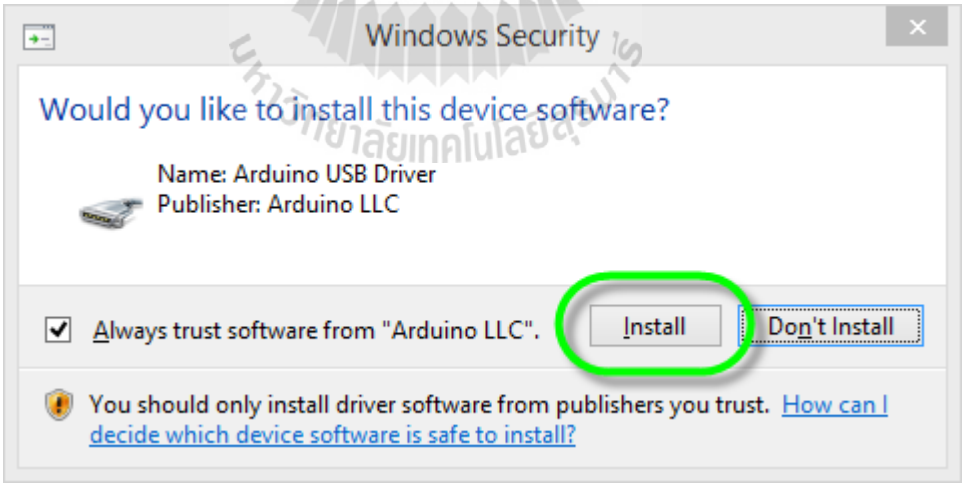


รูปที่ 3.21 next install program



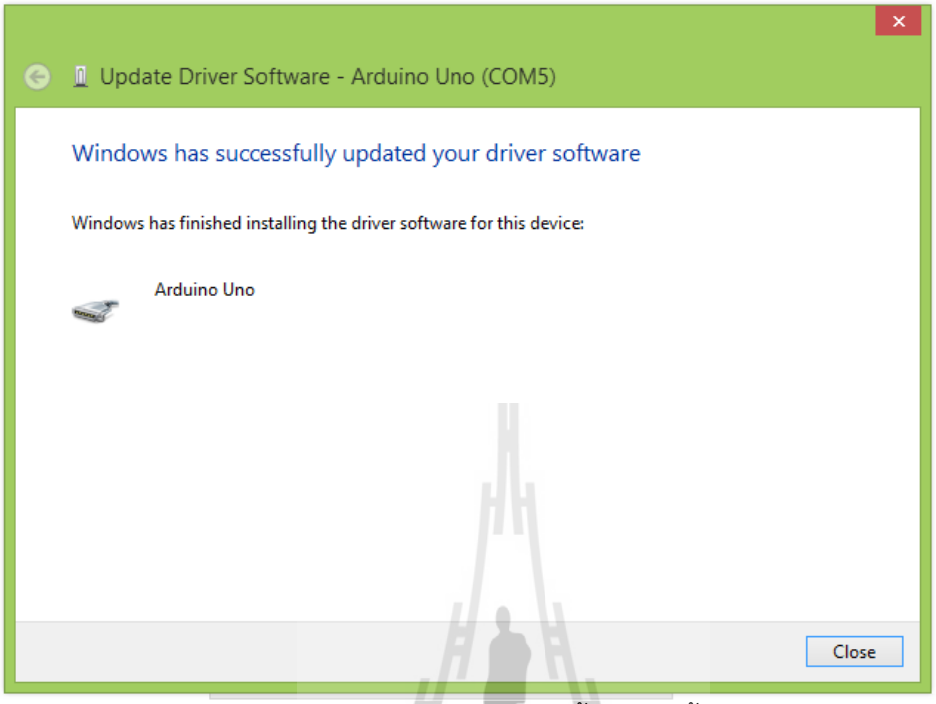
รูปที่ 3.22 รอลงโปรแกรม

7. คลิก Install เพื่อติดตั้ง Driver ลง Window



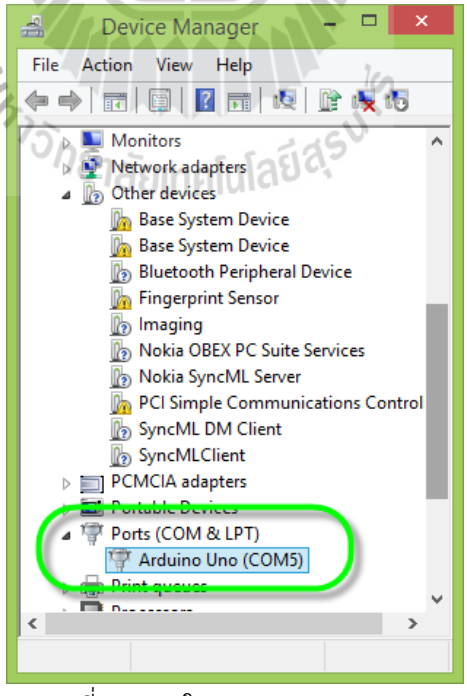
รูปที่ 3.23 Install เพื่อติดตั้ง Driver ลง Window

8. เสร็จสิ้นการติดตั้ง



รูปที่ 3.24 เสร็จสิ้นการติดตั้ง

9. เมื่อไปดูใน Device Manager จะเห็นรายการ Arduino UNO พร้อมหมายเลข Com พอร์ตที่ใช้ในการเชื่อมต่อ

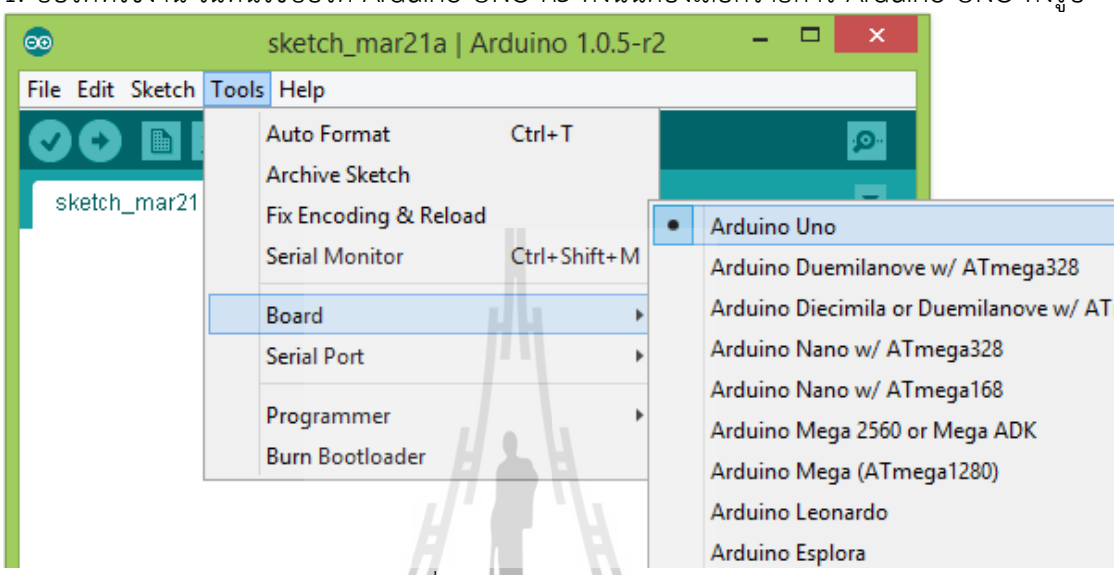


รูปที่ 3.25 ดูใน Device Manager

เริ่มต้นการทดลอง

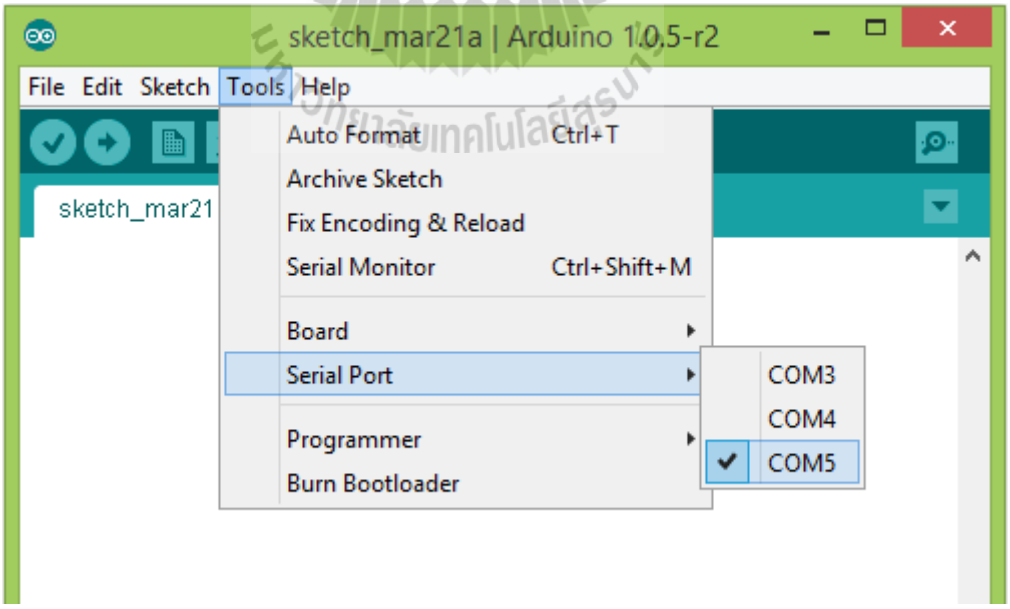
ก่อนที่จะเริ่มต้นการทดลองทุกครั้ง (หลังจากเปิดเครื่องคอมพิวเตอร์ใหม่) จำเป็นต้องมีการตั้งค่าบอร์ดที่ใช้งานโดยมีการตั้งค่า 3 รายการคือ

1. บอร์ดที่ใช้งาน ในที่นี้ใช้บอร์ด Arduino UNO R3 ดังนั้นต้องเลือกรายการ Arduino UNO ดังรูป



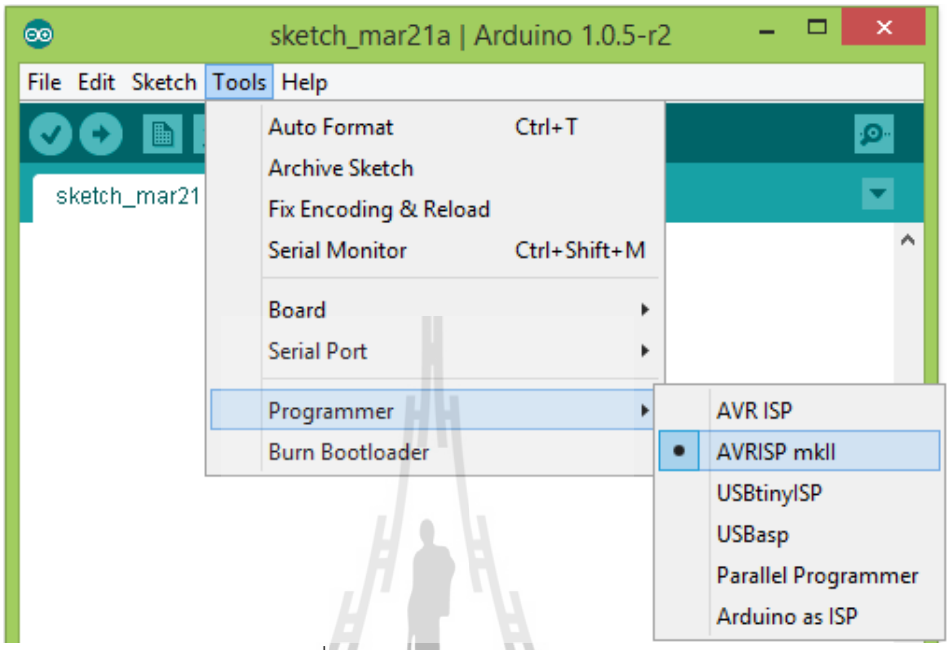
รูปที่ 3.26 เลือกชนิด Arduino

2. COM พอร์ตที่บอร์ดทำการเชื่อมต่อ โดยดูได้จาก Device Manager ดังที่ได้กล่าวมาแล้วข้างต้น



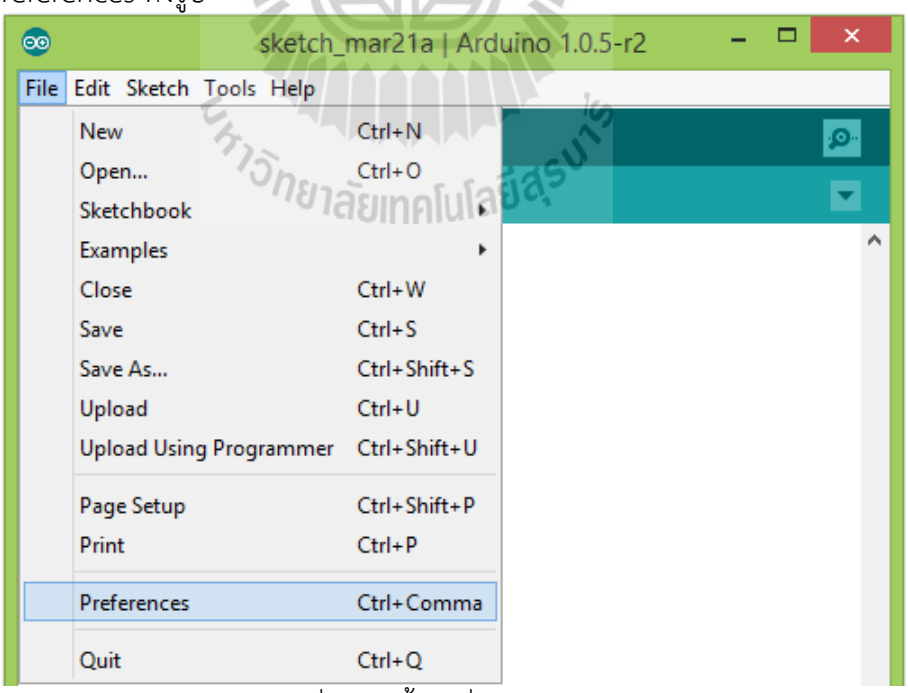
รูปที่ 3.27 เลือกport

3. เลือกชนิดเครื่องโปรแกรม เนื่องจากบูตโหลดเดอร์ของบอร์ดได้จำลองตัวเองเป็น AVRISP mkII ดังนั้นเครื่องโปรแกรมจึงต้องเลือกรายการดังรูป

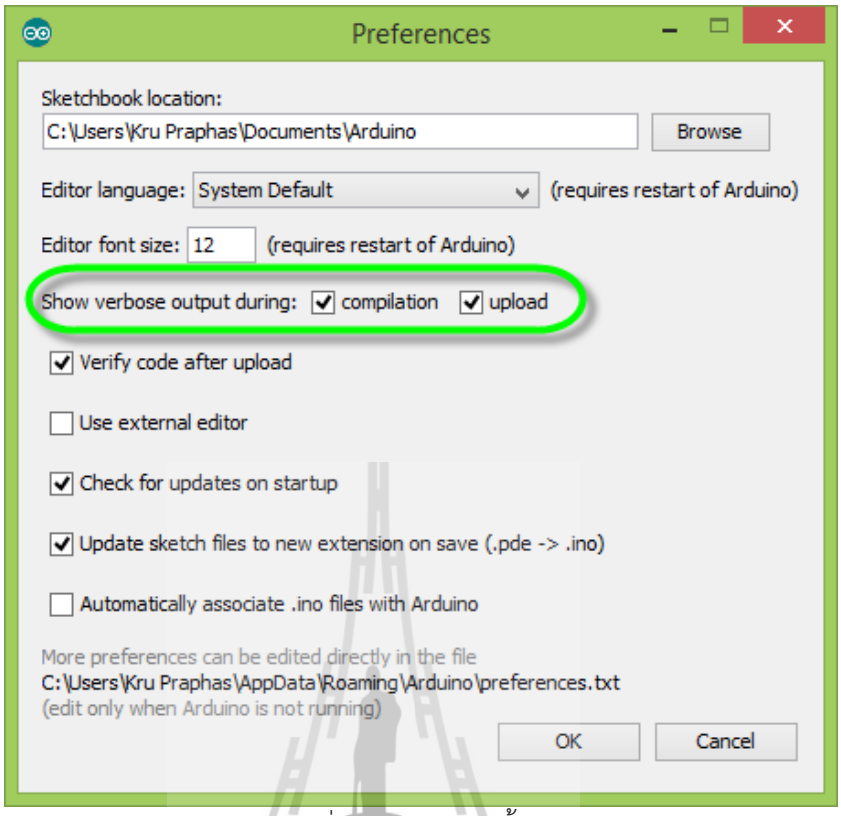


รูปที่ 3.28 เลือกชนิด Programmer

สำหรับการตั้งค่าให้แสดงตำแหน่งของไฟล์ภาษาเครื่อง เพื่อจะนำไปใช้ในการจำลองการทำงานจะต้องตั้งค่าที่ Preferences ดังรูป

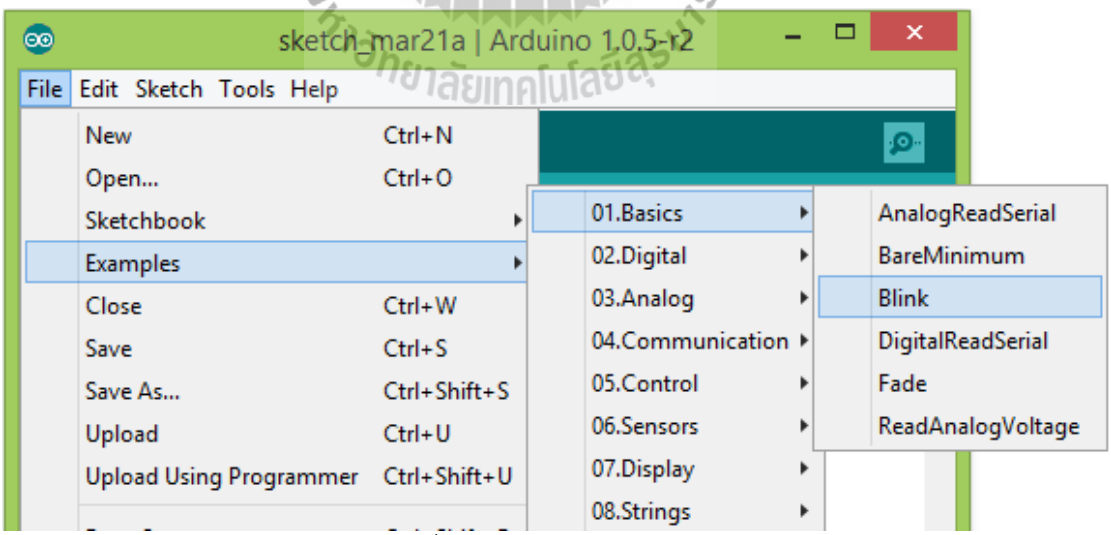


รูปที่ 3.29 ตั้งค่าที่ Preferences



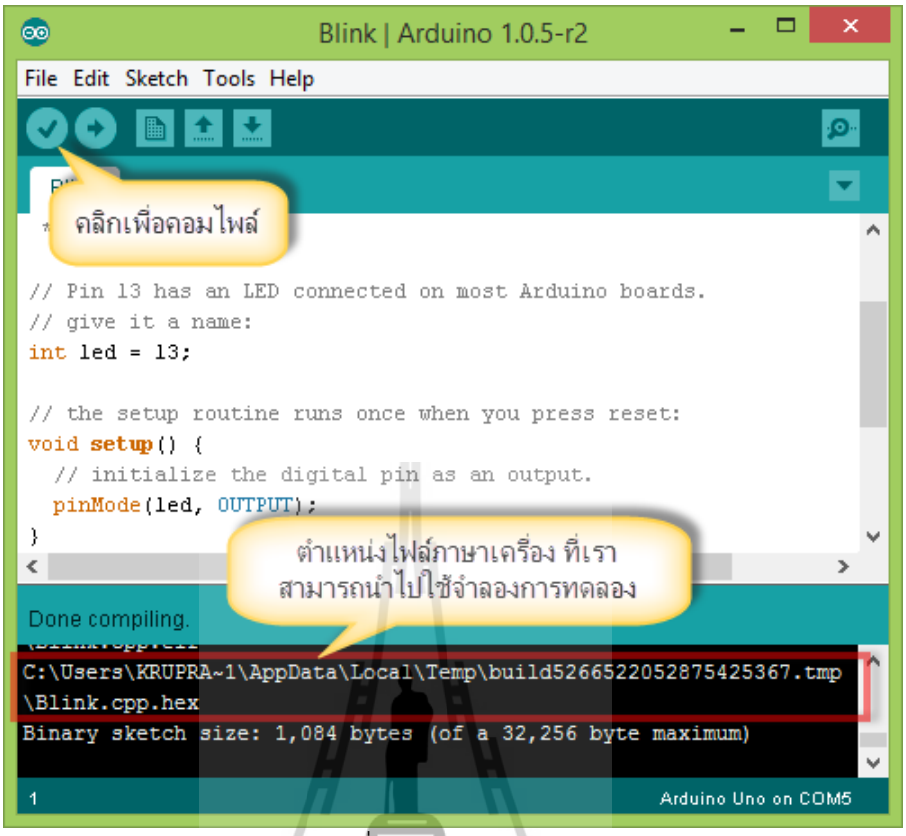
รูปที่ 3.30 หน้าต่างตั้งค่า

เริ่มทดลองจากตัวอย่างไฟกระพริบ Blink โดยเรียกไฟล์ตัวอย่างดังรูป



รูปที่ 3.31 ทดลองจากตัวอย่าง

ทดลองแปลงไฟล์โดยกดที่ไอคอนเครื่องหมายถูกดังรูป แต่หากต้องการอัพโหลดลงบอร์ดจริงด้วยให้กดที่ปุ่มหัวลูกศร (ปุ่มถัดไป)

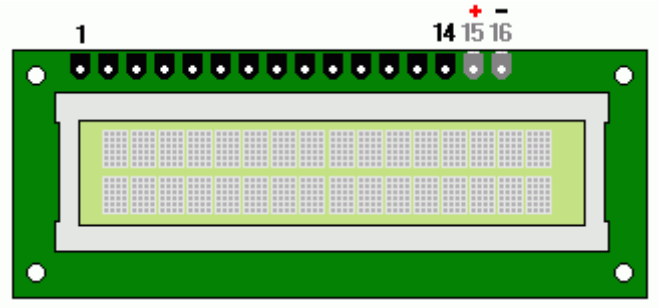


รูปที่ 3.32 แสดงสถานะ

3.3.3 จอแสดงผล LCD

รูปที่ 3.33 จอแสดงผล LCD

ลักษณะและตำแหน่งของขา LCD



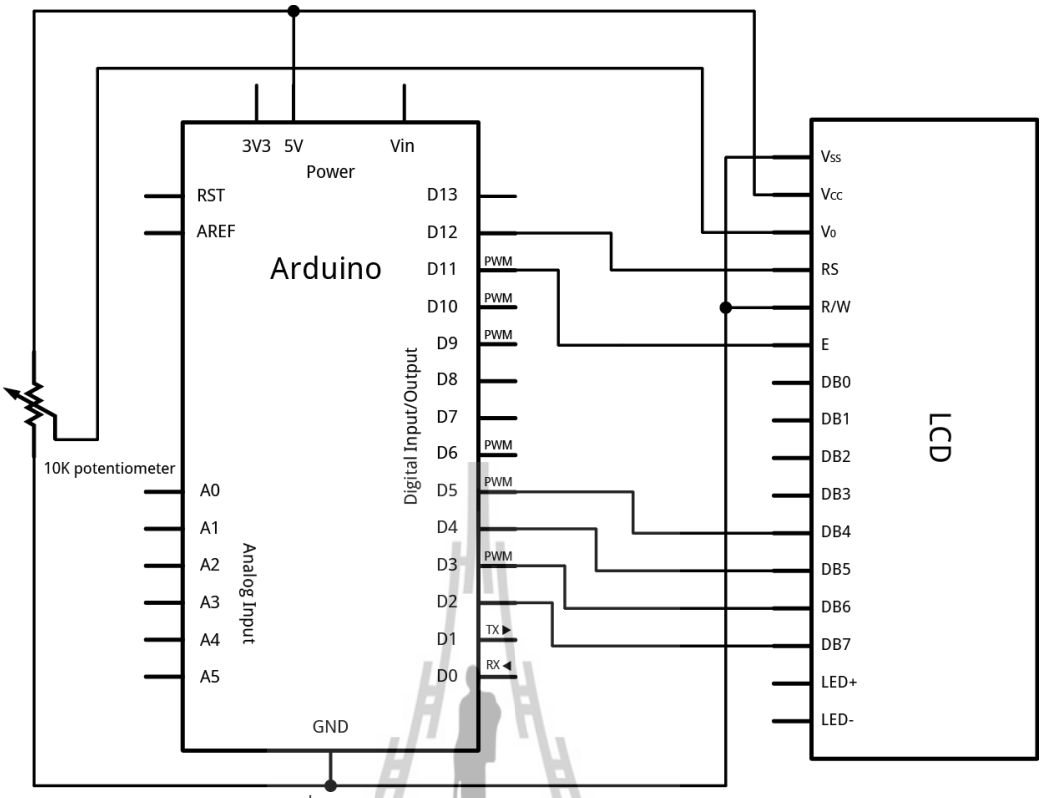
รูปที่ 3.34 ตำแหน่งขาของ LCD

ตำแหน่งของขาและหน้าที่การใช้งานของ LCD โมดูล

ตารางที่ 3.1 ตำแหน่งของขาและหน้าที่การใช้งานของ LCD

Pin No.	Symbol	Description	Level	Function	
1	VSS	Ground	-	0V	Ground
2	VDD	Power Supply	-	+5V	ต่อกับแรงดันไฟเลี้ยง +5V
3	VO	LCD Contr	-	-	ต่อกับแรงดันเพื่อปรับความเข้มของการแสดงผล
4	RS	Register Select	H/L	RS = 0 หมายถึงต้องการติดต่อกับรีจิสเตอร์คำสั่ง (Instruction Register) RS = 1 หมายถึงต้องการติดต่อกับรีจิสเตอร์ข้อมูล (Data Register)	
5	R/W	Read/Write	H/L	R/W = 0 หมายถึงต้องการเขียนข้อมูลไปยัง LCD โมดูล R/W = 1 หมายถึงต้องการอ่านข้อมูลจาก LCD โมดูล	
6	E	Enable	H, H->L	Enable Signal	
7 - 14	DB0-DB7	Data Bus	H/L	Data Bus Line	
15	A	Back Light A	-	Back Light +5V (สำหรับรุ่นที่มี Back Light)	
16	K	Back Light K	-	Back Light 0V (สำหรับรุ่นที่มี Back Light)	

การต่อ LCD เข้ากับบอร์ด Arduino

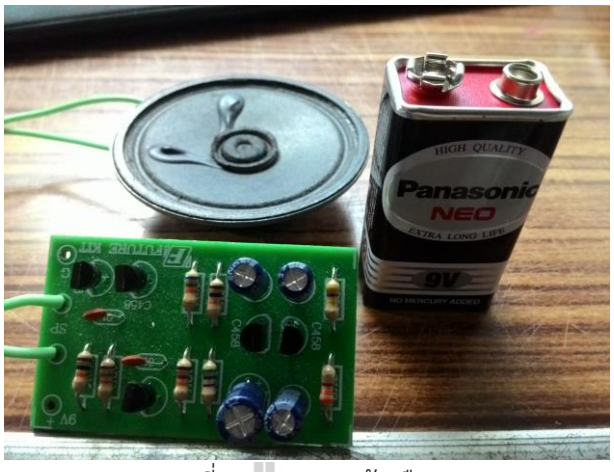


รูปที่ 3.35 การต่อ LCD เข้ากับบอร์ด Arduino

สามารถต่อบอร์ด Aduino กับ LCD ได้ดังนี้

- LCD RS ต่อกับ Digital 2
- LCD Enable ต่อกับ Digital 4
- LCD D4 ต่อกับ Digital 5
- LCD D5 ต่อกับ Digital 15
- LCD D6 ต่อกับ Digital 16
- LCD D7 ต่อกับ Digital 17
- LCD Gnd และ LCD R/W ต่อเข้า GND
- LCD Vcc ต่อเข้า 5V
- LCD VO ต่อกับ Digital 3

3.3.4 วงจรเสียงแจ้งเตือน



รูปที่ 3.36 วงจรแจ้งเตือน

วงจรเสียงไซเรน 6 เสียงชุดนี้เป็นวงจรกำเนิดเสียงชนิดหนึ่งซึ่งใช้หลักการทางดิจิตอลเมโมรี่เป็นสัญญาณทางดิจิตอลและโปรแกรมเข้าไปในตัวไอซี ซึ่งขบวนการทั้งหมดนี้จะทำมาจากทางโรงงานผู้ผลิตไอซี เราจะมาเปลี่ยนแปลงโปรแกรมเหล่านี้ไม่ได้ วงจรนี้เหมาะสำหรับนำไปใช้ติดกับรถของเล่นขนาดเล็ก ข้อมูลทางด้านเทคนิค: - ใช้แหล่งจ่ายไฟขนาด 9 โวลท์ดีซี - กินกระแสสูงสุดประมาณ 105 มิลลิแอมป์ - ขนาดแผ่นวงจรพิมพ์ : 1.13 x 1.00 นิ้ว

3.3.5 โซล่าเซลล์

รูปที่ 3.37 โซล่าเซลล์

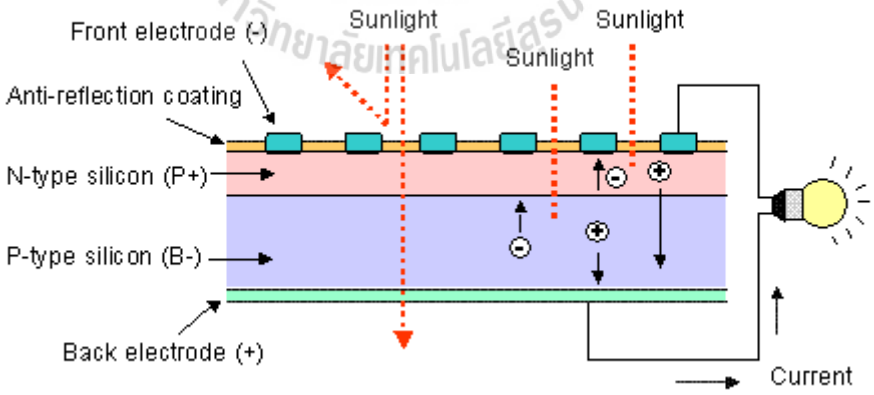
หลักการทำงานของโซล่าเซลล์

โซล่าเซลล์เป็นสิ่งประดิษฐ์กรรมทาง electronic ที่สร้างขึ้นเพื่อเป็นอุปกรณ์สำหรับเปลี่ยนพลังงานแสงอาทิตย์ให้เป็นพลังงาน ไฟฟ้า โดยการนำสารกึ่งตัวนำ เช่น ซิลิกอน ซึ่งมีราคาถูกที่สุดและมีมากที่สุดบนพื้นโลกมาผ่านกระบวนการทางวิทยาศาสตร์ เพื่อผลิตให้เป็นแผ่นบางบริสุทธิ์ และทันทีที่แสงตกกระทบบนแผ่นเซลล์ รังสีของแสงที่มีอนุภาคของพลังงานประกอบที่เรียกว่า โฟตอน (Proton) จะถ่ายเทพลังงานให้กับอิเล็กตรอน (Electron) ในสารกึ่งตัวนำจนมีพลังงานมากพอที่จะ

กระโดดออกมาจากแรงดึงดูดของอะตอม (atom) และเคลื่อนที่ได้อย่างอิสระ ดังนั้นเมื่ออิเล็กตรอนเคลื่อนที่ครบวงจรจะทำให้เกิดไฟฟ้ากระแสตรงขึ้นวัสดุสำคัญที่ใช้ทำโซล่าเซลล์ ได้แก่ สารซิลิคอน (Si) ซึ่งเป็นสารชนิดเดียวกับที่ใช้ทำชิพในคอมพิวเตอร์ และเครื่องอิเล็กทรอนิกส์ ซิลิคอนเป็นสารซึ่งไม่เป็นพิษ มีการนำมาผลิตโซล่าเซลล์ใช้กันอย่างแพร่หลายเพราะมีราคาถูก คงทน และเชื่อถือได้ นอกจากนี้ยังมีวัสดุชนิดอื่นที่สามารถนำมาผลิตโซล่าเซลล์ได้ เช่น แกลเลียมอาเซไนด์ CIS และ แคดเมียมเทลเลอไรด์ แต่ยังมีราคาสูง และบางชนิดยังไม่มีการพิสูจน์เรื่องอายุการใช้งานว่าสามารถใช้งานได้นาน

ข้อเสียของ Si : การทำให้บริสุทธิ์และอยู่ในรูปสารที่พร้อมจะทำเซลล์ฯ มีราคาแพง และ แตกหักง่ายในขบวนการผลิต

การทำงานของโซล่าเซลล์ เป็นขบวนการเปลี่ยนพลังงานแสงเป็นกระแสไฟฟ้าได้โดยตรง โดยเมื่อแสงซึ่งเป็นคลื่นแม่เหล็กไฟฟ้าและมีพลังงานกระทบกับสารกึ่งตัวนำ จะเกิดการถ่ายทอดพลังงานระหว่างกัน พลังงานจากแสงจะทำให้เกิดการเคลื่อนที่ของกระแสไฟฟ้า (อิเลคตรอน) ขึ้นในสารกึ่งตัวนำ จึงสามารถต่อกระแสไฟฟ้าดังกล่าวไปใช้งานได้ n– type ซิลิคอน ซึ่งอยู่ด้านหน้าของเซลล์ คือ สารกึ่งตัวนำที่ได้การโดปปิ้งด้วยสารฟอสฟอรัส มีคุณสมบัติเป็นตัวให้อิเล็กตรอนเมื่อรับพลังงานจากแสงอาทิตย์ p – type ซิลิคอน คือสารกึ่งตัวนำที่ได้การโดปปิ้งด้วยสารโบรอน ทำให้โครงสร้างของอะตอมสูญเสียอิเล็กตรอน (โฮล) เมื่อรับพลังงาน จากแสงอาทิตย์จะทำหน้าที่เป็นตัวรับอิเล็กตรอน เมื่อนำซิลิคอนทั้ง 2 ชนิด มาประกบต่อกันด้วย p – n junction จึงทำให้เกิดเป็น ” โซล่าเซลล์ ” ในสภาวะที่ยังไม่มีแสงแดด n – type ซิลิคอนซึ่งอยู่ด้านหน้าของเซลล์ ส่วนประกอบส่วนใหญ่พร้อมจะให้อิเล็กตรอน แต่ก็ยังมีโฮลปะปนอยู่บ้างเล็กน้อย ด้านหน้าของ n – type จะมีแถบโลหะเรียกว่า Front Electrode ทำหน้าที่เป็นตัวรับอิเล็กตรอน ส่วน p – type ซิลิคอนซึ่งอยู่ด้านหลังของเซลล์ โครงสร้างส่วนใหญ่เป็นโฮล แต่ยังคงมีอิเล็กตรอนปะปนบ้างเล็กน้อย ด้านหลังของ p – type ซิลิคอน จะมีแถบโลหะเรียกว่า Back Electrode ทำหน้าที่เป็นตัวรวบรวมโฮล



รูปที่ 3.38 แสดง type และ การถ่ายเทพลังงาน

เมื่อมีแสงอาทิตย์ตกกระทบ แสงอาทิตย์จะถ่ายเทพลังงานให้กับอิเล็กตรอนและโฮล ทำให้เกิดการเคลื่อนไหว เมื่อพลังสูงพอทั้งอิเล็กตรอนและโฮลจะวิ่งเข้าหาเพื่อจับคู่กัน อิเล็กตรอนจะวิ่งไปยังชั้น n – type และโฮลจะวิ่งไปยังชั้น p type

เมื่อมีแสงอาทิตย์ตกกระทบ แสงอาทิตย์จะถ่ายเทพลังงานให้กับอิเล็กตรอนและโฮล ทำให้เกิดการเคลื่อนไหว เมื่อพลังสูงพอทั้งอิเล็กตรอนและโฮลจะวิ่งเข้าหาเพื่อจับคู่กัน อิเล็กตรอนจะวิ่งไปยังชั้น n – type และโฮลจะวิ่งไปยังชั้น p type อิเล็กตรอนวิ่งไปรวมกันที่ Front Electrode และโฮลวิ่งไปรวมกันที่ Back Electrode เมื่อมีการต่อวงจรไฟฟ้าจาก Front Electrode และ Back Elec trode ให้ครบวงจร ก็จะเกิดกระแสไฟฟ้าขึ้น เนื่องจากทั้งอิเล็กตรอนและโฮลจะวิ่งเพื่อจับคู่กันตัวแปรที่สำคัญที่มีส่วนทำให้โซล่าเซลล์มีประสิทธิภาพการทำงานในแต่ละ พื้นที่ต่างกัน และมีความสำคัญในการพิจารณานำไปใช้ในแต่ละพื้นที่ ตลอดจนการนำไปคำนวณระบบหรือคำนวณจำนวนแผงแสงอาทิตย์ที่ต้องใช้ในแต่ละ พื้นที่ คือความเข้มของแสง และอุณหภูมิ

กระแสไฟ (Current) จะเป็นสัดส่วนโดยตรงกับความเข้มของแสง หมายความว่าเมื่อความเข้มของแสงสูง กระแสที่ได้จากโซล่าเซลล์ก็จะสูงขึ้น ในขณะที่แรงดันไฟฟ้าหรือโวลต์แทบจะไม่แปรไปตามความเข้มของแสงมากนัก ความเข้มของแสงที่ใช้วัดเป็นมาตรฐานคือ ความเข้มของแสงที่วัดบนพื้นโลกในสภาพอากาศปลอดโปร่ง ปราศจากเมฆหมอกและวัดที่ระดับน้ำทะเลในสภาพที่แสงอาทิตย์ตั้งฉากกับพื้นโลก ซึ่งความเข้ม ของแสงจะมีค่าเท่ากับ 100 mW ต่อ ตร.ซม. หรือ 1,000 W ต่อ ตร.เมตร ซึ่งมีค่าเท่ากับ AM 1.5 (Air Mass 1.5) และถ้าแสงอาทิตย์ทำมุม 60 องศากับพื้นโลกความเข้มของแสง จะมีค่าเท่ากับประมาณ 75 mW ต่อ ตร.ซม. หรือ 750 W ต่อ ตร.เมตร ซึ่งมีค่าเท่ากับ AM2 กรณีของแผงโซล่าเซลล์นั้นจะใช้ค่า AM 1.5 เป็นมาตรฐานในการวัดประสิทธิภาพของแผง

กระแสไฟ (Current) จะไม่แปรตามอุณหภูมิที่เปลี่ยนแปลงไป ในขณะที่แรงดันไฟฟ้า (โวลท์) จะลดลงเมื่ออุณหภูมิสูงขึ้น ซึ่งโดยเฉลี่ยแล้วทุกๆ 1 องศาที่เพิ่มขึ้น จะทำให้แรงดันไฟฟ้าลดลง 0.5% และในกรณีของแผงโซล่าเซลล์มาตรฐานที่ใช้กำหนดประสิทธิภาพของแผงแสง อาทิตย์คือ ณ อุณหภูมิ 25 องศา C เช่น กำหนดไว้ว่าแผงแสงอาทิตย์มีแรงดันไฟฟ้าที่วงจรเปิด (Open Circuit Voltage หรือ V oc) ที่ 21 V ณ อุณหภูมิ 25 องศา C ก็จะหมายความว่า แรงดันไฟฟ้าที่จะได้จากแผงแสงอาทิตย์เมื่อยังไม่ได้ต่อกับอุปกรณ์ไฟฟ้า ณ อุณหภูมิ 25 องศา C จะเท่ากับ 21 V ถ้าอุณหภูมิสูงกว่า 25 องศา C เช่น อุณหภูมิ 30 องศา C จะทำให้แรงดันไฟฟ้าของแผงแสงอาทิตย์ลดลง 2.5% (0.5% x 5 องศา C) นั่นคือ แรงดันของแผงแสงอาทิตย์ที่ V oc จะลดลง 0.525 V (21 V x 2.5%) เหลือเพียง 20.475 V (21V ? 0.525V) สรุปได้ว่า เมื่ออุณหภูมิสูงขึ้น แรงดันไฟฟ้าก็จะลดลง ซึ่งมีผลทำให้กำลังไฟฟ้าสูงสุดของแผงแสงอาทิตย์ลดลงด้วย

3.3.6 Xbee

รูปที่ 3.39 Xbee Pro

Xbee เป็นอุปกรณ์ที่มี ไมโครคอนโทรลเลอร์ และ RF IC อยู่ภายใน (อ้างอิงได้จาก Schematic) ทำหน้าที่เป็น อุปกรณ์ transceiver (อุปกรณ์รับ-ส่งสัญญาณ) แบบ Half Duplex ย่านความถี่ 2.4 GHz มีการจัดการโดยใช้พลังงานต่ำ ใช้งานง่าย มี interface ที่ใช้รับและส่งข้อมูลกับ Xbee เป็น UART (TTL) ซึ่งสำหรับทางด้านไมโครคอนโทรลเลอร์ สามารถนำขาที่ใช้ติดต่อสื่อสาร UART ของ Xbee ต่อเข้ากับ UART ของ ไมโครคอนโทรลเลอร์ ได้โดยตรง

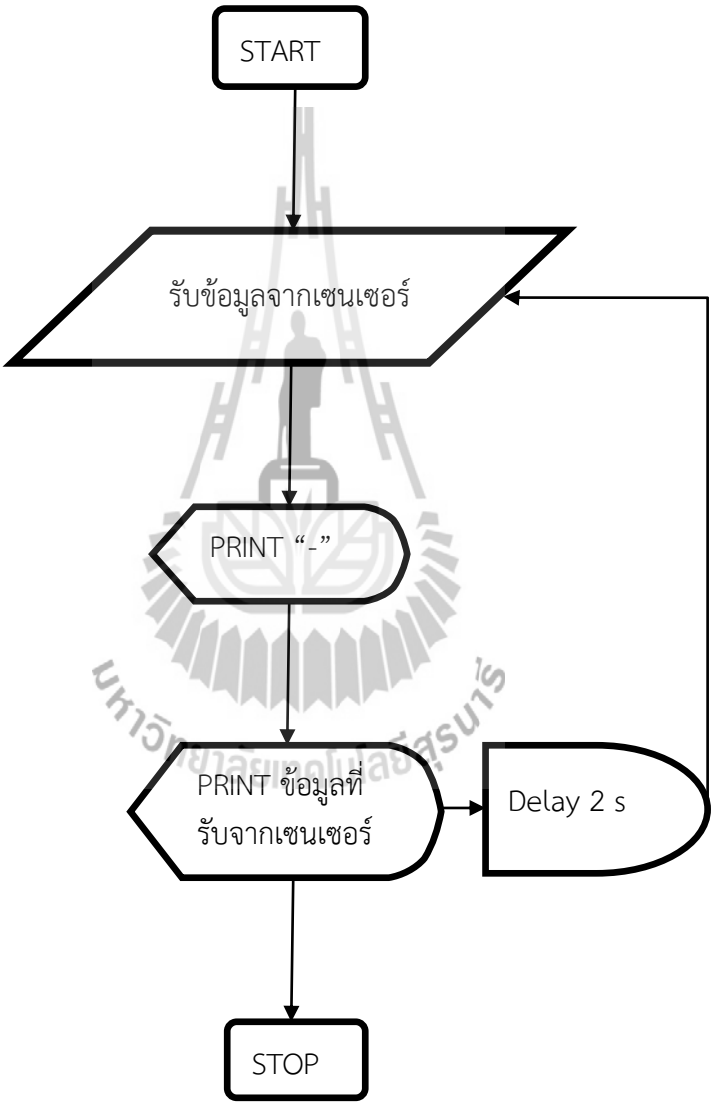
Xbee สามารถใช้งานตามมาตรฐาน Zigbee ได้ โดยที่ไม่ต้องเขียนโปรแกรมสร้าง เครือข่าย Zigbee เลย เพราะว่าทางผู้ผลิตได้จัดทำ firmware ที่จะโหลดเข้าไปในตัว Xbee ให้สามารถ set parameter ผ่าน software interface (X-CTU หรือโปรแกรมที่เขียนขึ้นเอง) , ผ่านทาง At command (เหมือนกับการควบคุม GSM Module) โดยใช้ Hyper terminal หรือ ผ่านทางการรับส่งข้อมูลด้วยไมโครคอนโทรลเลอร์ ได้อย่างง่ายดาย โดยเมื่อ set Xbee ให้ทำงานเป็นอุปกรณ์ในเครือข่าย Zigbee แล้ว และจะเรียก Xbee แต่ละตัวว่าเป็น Node

Firmware ที่ใช้กับ Xbee จะใช้โหลดผ่านโปรแกรม X-CTU ทั้ง นี้ Xbee แต่ละรุ่น จะสามารถ Setting Function การใช้งานได้มากมาย ทำให้ Firmware ที่จะต้องโหลดเข้าไปนั้น มีมากมายหลายแบบ และต้องเลือกให้เหมาะสมกับการใช้งาน

3.4 การเขียนโปรแกรมของระบบตรวจวัดอุณหภูมิอัตโนมัติโดยผ่านเครือข่าย xbee โดยมีแหล่งจ่ายเป็นโซล่าเซลล์

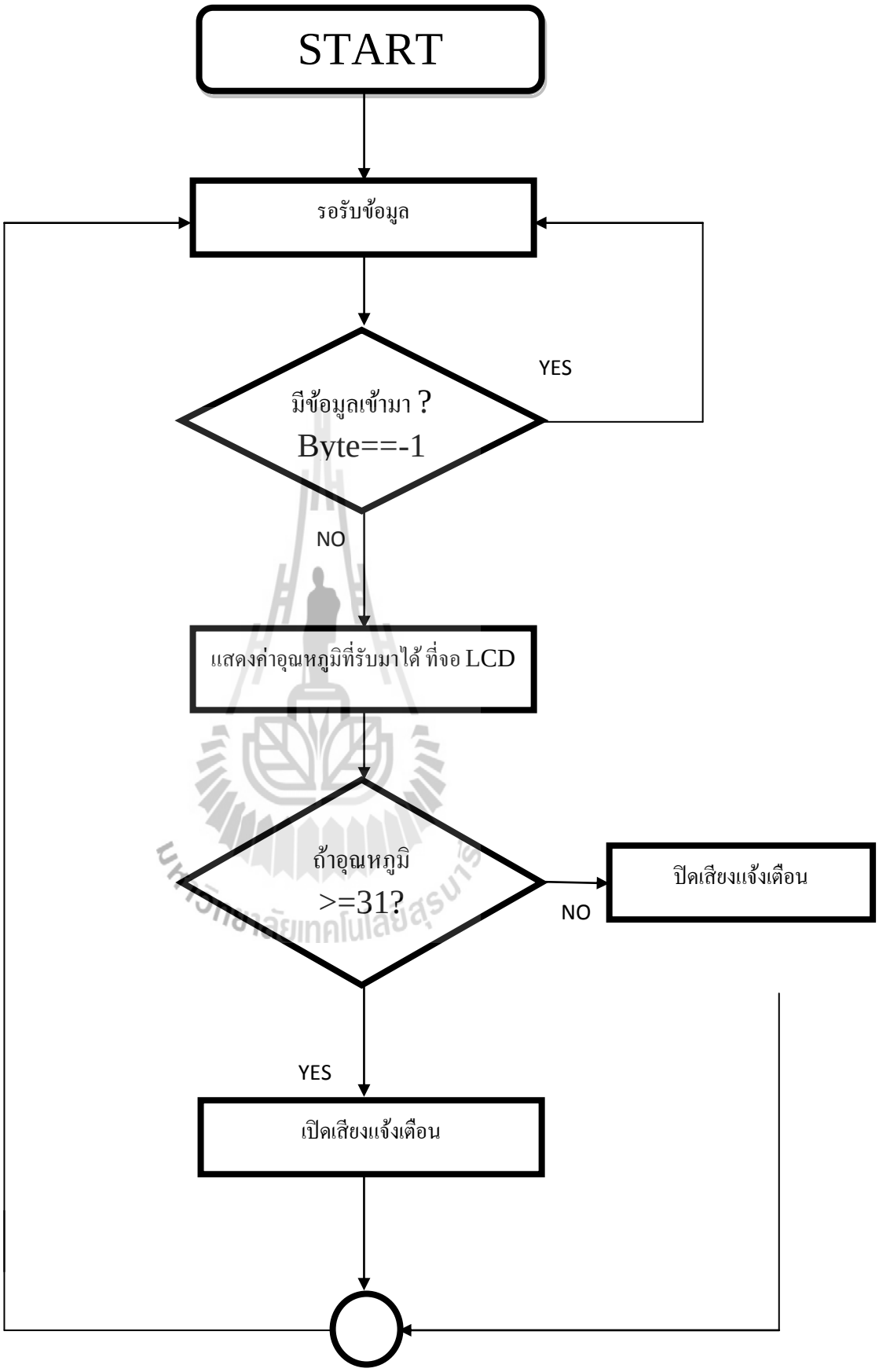
3.4.1 แผนผังการทำงานของโปรแกรม (Flow chart)

ภาคส่ง



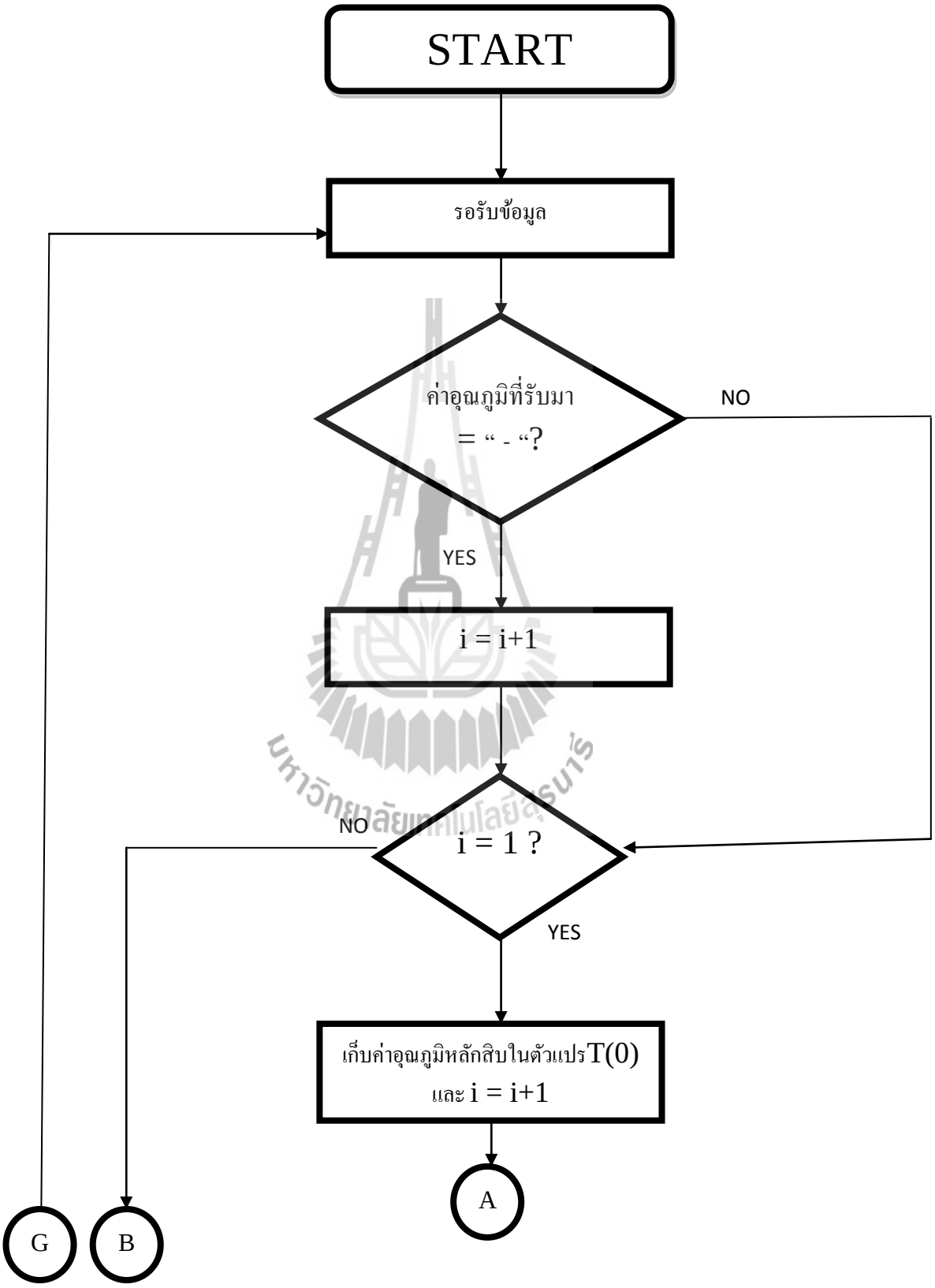
รูปที่ 3.40 แผนผังการทำงานของภาคส่ง

ภาครับ

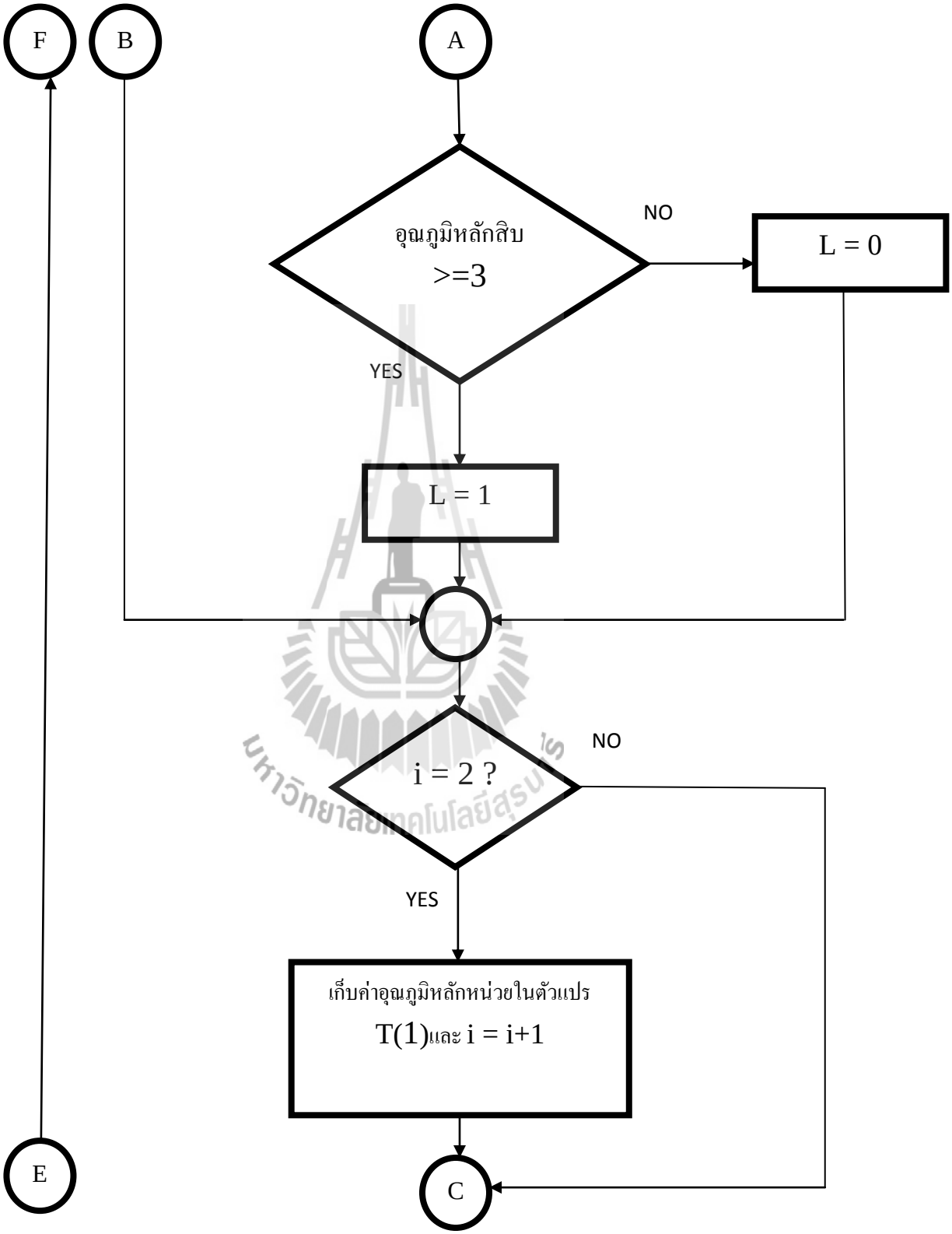


รูปที่ 3.41 แผนผังการทำงานของภาครับ

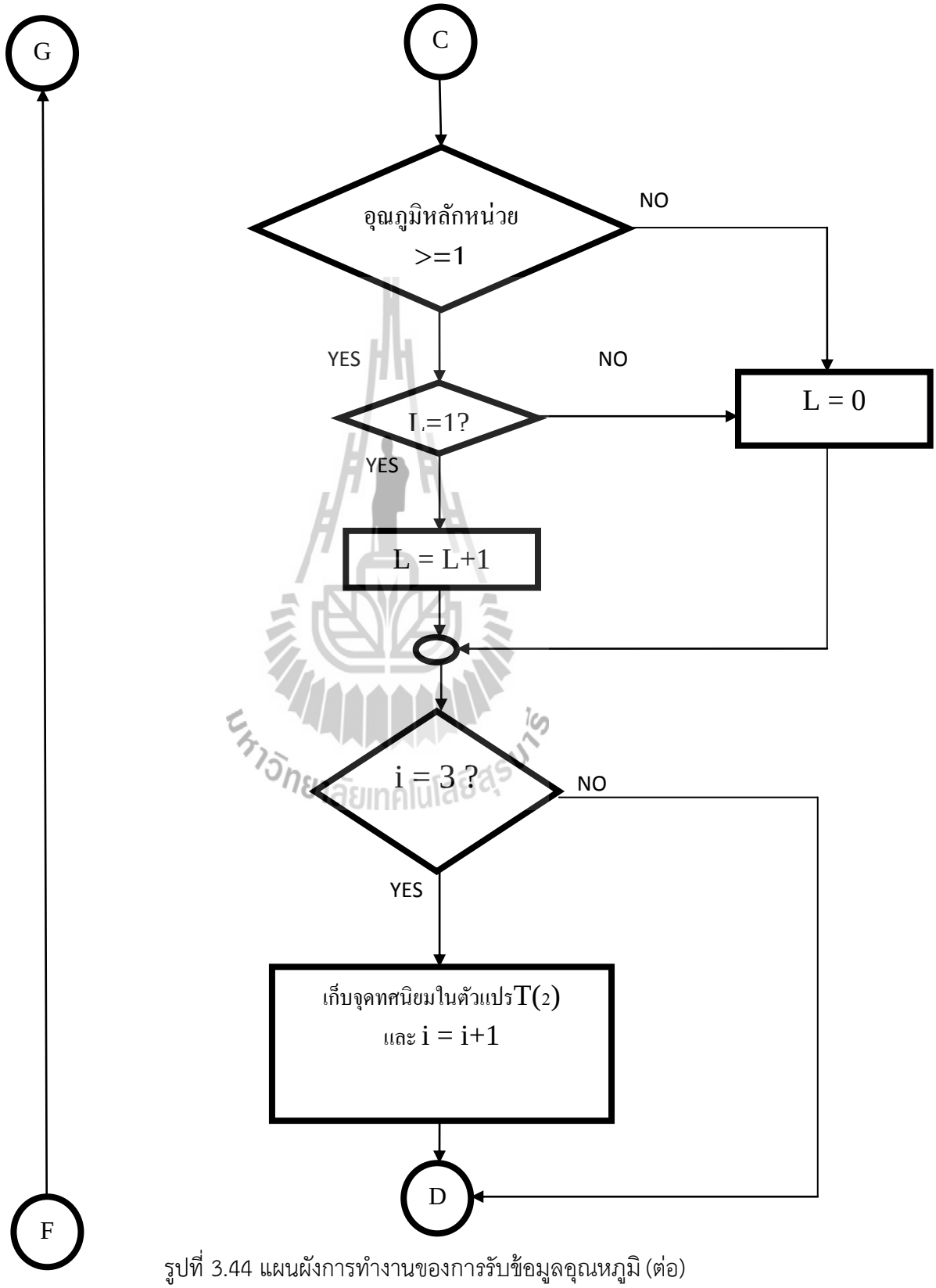
การรับข้อมูลอุณหภูมิ



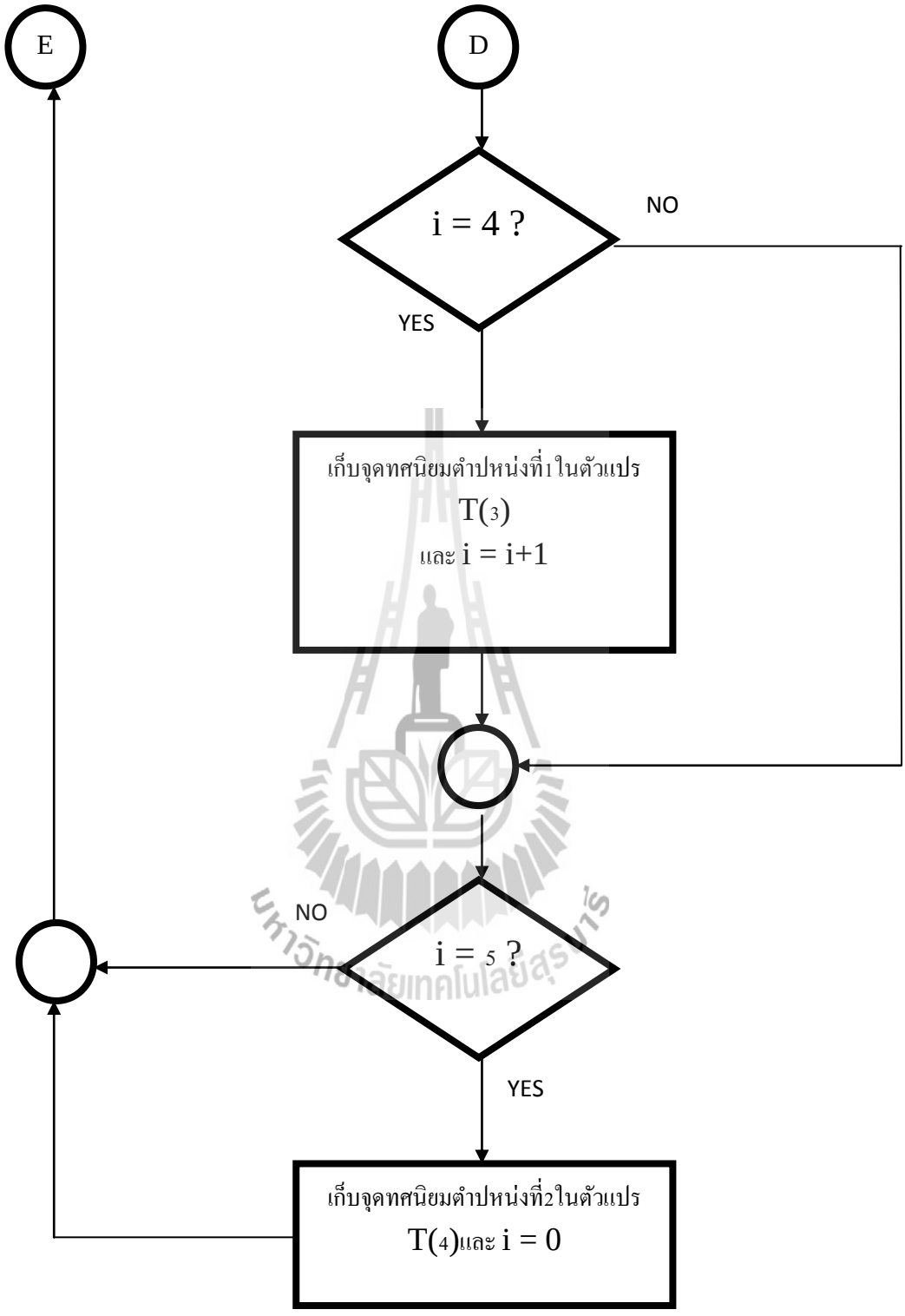
รูปที่ 3.42 แผนผังการทำงานของการรับข้อมูลอุณหภูมิ



รูปที่ 3.43 แผนผังการทำงานของการรับข้อมูลอุณหภูมิ (ต่อ)



รูปที่ 3.44 แผนผังการทำงานของการรับข้อมูลอุณหภูมิ (ต่อ)



รูปที่ 3.45 แผนผังการทำงานของการรับข้อมูลอุณหภูมิ (ต่อ)

3.4.2 Code program และคำอธิบาย

ภาคส่ง

```
#include <OneWire.h>                    //ในการเขียนโปรแกรมสำหรับ Arduino เพื่อ
                                        เชื่อมต่อกับไอซี DS18B20 สามารถใช้ไลบรารี่
                                        "OneWire.h" ของ Arduino ซึ่งทำให้สะดวกใน
                                        การเขียนโปรแกรม เช่น เพื่อการค้นหาหลายเลข
                                        (64-bit code) ของอุปกรณ์ที่ต่ออยู่กับบัส 1-
                                        Wire การสั่งให้ไอซีตามหมายเลขอุปกรณ์อ่าน
                                        และแปลงค่าอุณหภูมิ
#include <DallasTemperature.h>
#define ONE_WIRE_BUS 2                  //รับข้อมูล จากเซ็นเซอร์ เข้ามาที่ pin
                                        D2

OneWire oneWire(ONE_WIRE_BUS);
DallasTemperature sensors(&oneWire);

void setup(void)
{
  Serial.begin(9600);                   //เซ็ต Buadrate เริ่มต้น = 9600
  sensors.begin();                      //สั่งให้ เซ็นเซอร์ (DS18B20) เริ่ม
                                        ทำงาน
}

void loop(void)
{
  sensors.requestTemperatures();        //รับค่าจากเซ็นเซอร์ (DS18B20)
  Serial.print("-");                    //สั่ง เครื่องหมาย -
  Serial.println(sensors.getTempCByIndex(0));  //ส่งค่าที่ได้จากเซ็นเซอร์ (DS18B20)
delay(2000);                            //หน่วงเวลา 2วินาที
}
```

ภาครับ

```
#include <LiquidCrystal.h>
LiquidCrystal lcd(12, 11, 5, 4, 3, 2);
char T[5];
int i=0;
int l=0;
void setup() {                          //ฟังก์ชั่น ที่มีหน้าที่ set หน้าที่ของแต่ล่ะ pin
Serial.begin(9600);                     //กำหนด Baud rate โดยตัวเลขที่อยู่ในวงเล็บ
                                        คือค่าความเร็วในการรับ-ส่งข้อมูล (Baud rate)
lcd.begin(16, 2);                       //ชนิดของ LCD แบบ 16 ตัวอักษร 2 บรรทัด
lcd.print("Temperature");               //สั่ง พิมข้อความ Temperature บน LCD
pinMode(8,OUTPUT);                      //output ออกที่ pin8
}
void loop() {
char byte=Serial.read();                //ประกาศตัวแปร byte = ค่าที่อ่านได้จาก
                                        DS18B20 (เซ็นต์เซอร์อุณหภูมิ) ซึ่งการส่ง
                                        ข้อมูลของ DS18B20 จะเรียงลำดับตามนี้ 1.ส่ง"-
                                        " 2.ส่งหลักสิบ 3.ส่งหลักหน่วย 4.ส่งจุดทศนิยม
                                        5.ทศนิยมตำแหน่งที่ 16.ทศนิยมตำแหน่งที่2

if(byte== -1){                          //ถ้ามีมีการส่งข้อมูลค่า xbee ที่ส่งมาจะเป็น -1
  delay(1);
}
else{

if(byte=='-'){i=i+1;}                   //1.รับค่า อันดับแรกได้ - เข้ามา จึงมา เริ่มที่ตรง
                                        นี้ จากนั้น ค่า i จะเป็น 1

if(i==1){                               //2.ถ้า i=1 จะทำการเก็บตัวแปรอะเรย์ ข้อมุลที่
                                        รับได้ (รับหลักสิบ) ไปที่ T(0) จากนั้นเพิ่มให้ i=2
    T[i-1]=byte;
    i=i+1;
    if(byte>='3'){                      //2.1แต่ถ้า ค่าที่รับได้มีค่า >=3 จะทำให้ l=1
                                        ถ้าไม่ใช่ l=0
      l=1;
    }
    else{
```

```
      l=0;
    }
  }
    if(i==2){                          //3. ถ้า i=2  จะทำการเก็บตัวแปรอะเรย์ ข้อมุล
                                       ที่รับได้(รับหลักหน่วย)ไปที่ T(1)  จากนั้นเพิ่มให้
                                       i=3
    T[i-1]=byte;
    i=i+1;
    if(byte>='1'){                     //3.1ถ้า ค่าที่รับมาได้ มีค่า>=1 จะไปดูว่า l มีค่า
                                       =1 หรือไม่ ถ้าใช่ จะทำให้ l=2 ถ้าไม่ใช่จะทำให้
                                       l=0
      if(l==1){
        l=2;
      }
      else{l=0;}
    }
  }
if(i==3){                              //4. ถ้า i=3  จะทำการเก็บตัวแปรอะเรย์ ข้อมุล
                                       ที่รับ (รับ จุดทศนิยม) ได้ไปที่ T(2)  จากนั้นเพิ่ม
                                       ให้ i=4
    T[i-1]=byte;
    i=i+1;
  }
if(i==4){                              //5. ถ้า i=4 จะทำการเก็บตัวแปรอะเรย์ ข้อมุลที่รับ
                                       (ทศนิยมตำแหน่งที่1) ได้ไปที่ T(3) จากนั้นเพิ่มให้
                                       i=5
    T[i-1]=byte;
    i=i+1;
  }
    if(i==5){                          //6.ถ้า i=5 จะทำการเก็บตัวแปรอะเรย์ ข้อมุลที่รับ
                                       (ทศนิยมตำแหน่งที่2) ได้ไปที่ T(4) จากนั้นเพิ่มให้
                                       i=0
    T[i-1]=byte;
    i=0;
    lcd.setCursor(0, 1);               //7.set ตำแหน่งเริ่มต้น lcd ที่คอลัมน์ 1 แถว 2
                                       จากนั้นแสดง อะเรย์T() ทั้งหมดที่เก็บค่าไว้
```

```
    lcd.print(T);
  }
}
if(l==2){                          //8.ถ้า l=2 ให้ส่ง output ออกที่ pin8 แต่ถ้า ไม่ใช่
                                   ก็ไม่ส่ง ซึ่งในส่วนนี้หมายถึงว่า
digitalWrite(8,HIGH);              ที่ l=2 ก็คือ อุณภูมิของเราเกิน 31 แล้ว จึงส่ง
                                   output ออกที่ pin8 เพื่อให้วงจร Alarm แจ้งเตือน
}
else{
digitalWrite(8,LOW);
}

}
```

3.4.3 การเขียนโปรแกรม Arduino IDE

บอร์ดArduino นั้นต้องเขียนด้วยโปรแกรม Arduino IDE (Integrated Development Environment) ซึ่งตัวโปรแกรมตัวนี้นั้นถูกพัฒนาต่อมาจากโปรแกรม open source อย่าง Processing และ Wiring และด้วยความที่ทั้งตัวโปรแกรม Processing และ Wiring นั้นถูกพัฒนาขึ้นมาด้วยภาษา JAVA จึงทำให้สามารถใช้งานได้ทั้งบนระบบปฏิบัติการ Windows, Linux และ Mac OS ได้ ตัวโปรแกรม Arduino IDE ถูกพัฒนาขึ้นมาเพื่อให้ใช้งานได้ง่ายสะดวกและลดความซับ ซ้อนของการเขียนโปรแกรมในรูปแบบของภาษา C/C++ ให้ดูเรียบง่าย ไม่ซับซ้อน มีคำสั่งและไลบรารี่เตรียมไว้ให้แล้ว

- **ชนิดของตัวแปรในการเขียนโปรแกรม**

Char มีขนาด 8 บิต ใช้แทนข้อมูล -128 ถึง 127
Int มีขนาด 16 บิต ใช้แทนข้อมูลจำนวนเต็ม ในช่วง -32,768 ถึง +32,767
Long มีขนาด 32 บิต ใช้แทนข้อมูลจำนวนเต็ม ในช่วง -2,147,483,648 ถึง 2,147,483,647
Float มีขนาด 32 บิต ใช้แทนข้อมูลจำนวนจริงในช่วง 10 ยกกำลัง -38 ถึง 10 ยกกำลัง 38
Double มีขนาด 64 บิต ใช้แทนข้อมูลจำนวนจริงในช่วง 10 ยกกำลัง -308 ถึง 10 ยกกำลัง 38

- **โครงสร้างการเขียนโปแกรม**

#include<ประกาศLibrary >
ประกาศตัวแปรต่างๆ
void setup() {

กำหนดใช้งานportต่างๆ โดยใน Arduino เรียก port ต่างๆเป็นหมายเลข

```
}
    void loop()
{
    การกำหนดในการใช้ฟังก์ชันต่างๆ
}
```

- **MENU bar ของ Arduino IDE จะประกอบด้วย**

1. Menu File โดยทำหน้าที่บริหารจัดการข้อมูลที่เรียกว่า Sketch File (ไฟล์ที่ทำการเขียนจะเรียกว่า sketch files)

* New : สร้าง Sketch File ใหม่
* Sketchbook :
1. Open : เรียก Sketchbook ที่บันทึกไว้ก่อนหน้านี้
2. Example :เปิด Sketchbook ตัวอย่างที่มาจาก Library
* Save : บันทึก Sketch File ณ.ปัจจุบัน
* Save as : บันทึก Sketch File ในที่ อื่นหรือชื่อใหม่
* Upload to I/O board : การ burn สู่ MCU(AVR) หลังจากที่เขียนโปรแกรมเสร็จ
* Preference : ตั้งค่าต่าง ๆ ของArduino IDE
2. Edit Menu ทำหน้าที่แก้ไข wording

* Undo : ยกเลิกคำสั่งหรือการพิมพ์ครั้งล่าสุด
* Redo : ทำซ้ำคำสั่งหรือการพิมพ์ครั้งล่าสุด
* Cut : ตัดข้อความไว้ใน clipboard
* Copy : คัดลอกข้อความจาก clipboard
* Paste : แปะข้อความจาก clipboard
* Select all : เลือก (แรเงา) ข้อความทั้งหมด
* Find : ค้นหาข้อความ
* Find Next : ค้นหาข้อความถัดไป

3. Sketch Menu เป็นเมนูที่ใช้คำสั่ง Compile ,เพิ่มLibrary
* Verify/Compile : Compile ภาษา c/c++ เป็นภาษา Machine
* Stop : หยุดการ Compile
* Import Library : เลือก/เพิ่ม Library โดยแทรกในบรรทัดแรก
* Show Sketch Folders : ทำการแสดงที่เก็บโปรแกรม
* Add File.. : คัดลอก File เลือก/เพิ่มมาบันทึกรวมใน Folder ปัจจุบัน
4. Tools Menu :ใช้จัด code ,เลือกเบอร์ ไมโครคอนโทรลเลอร์ ,เลือก port serial

* Auto Format : จัดวาง code ให้เป็นระเบียบสวยงาม ให้อ่านได้ง่ายโดยแยกสี code ที่เป็นคำสั่งและตัวแปร

* copy to Forum : คัดลอก code ลง clipboard

* Archive Sketch : สั่งให้บีบอัด Folder ปัจจุบันเป็นไฟล์บีบอัด มีประโยชน์ในการทำ CVS

* Board : เลือก Hardware Device ที่ใช้งานกับโปรแกรม เช่น Arduino Nano เป็นต้น

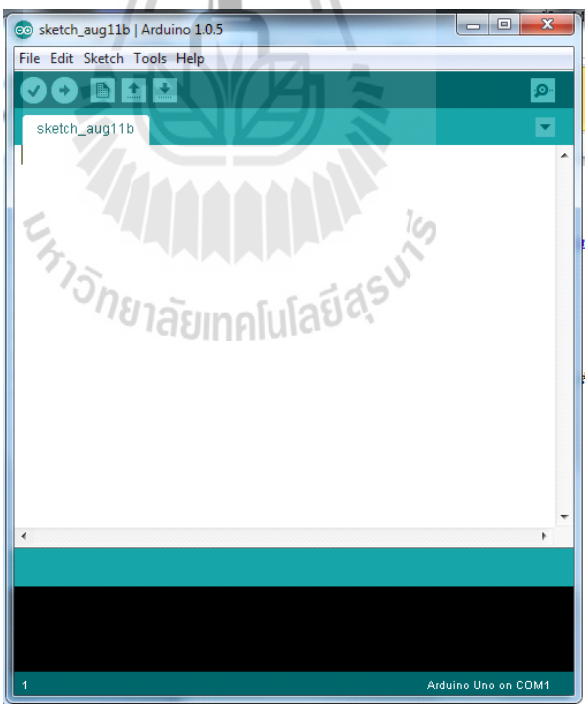
* Serial Ports : เลือก port เพื่อ communication ระหว่าง Hardware Device กับ Arduino IDE

* Burn Bootloader : เพื่อทำการ bootloadder ให้ Hardware Device โดยต้องมีเครื่อง burn

3.4.4 การโหลดโปรแกรมลงบอร์ด

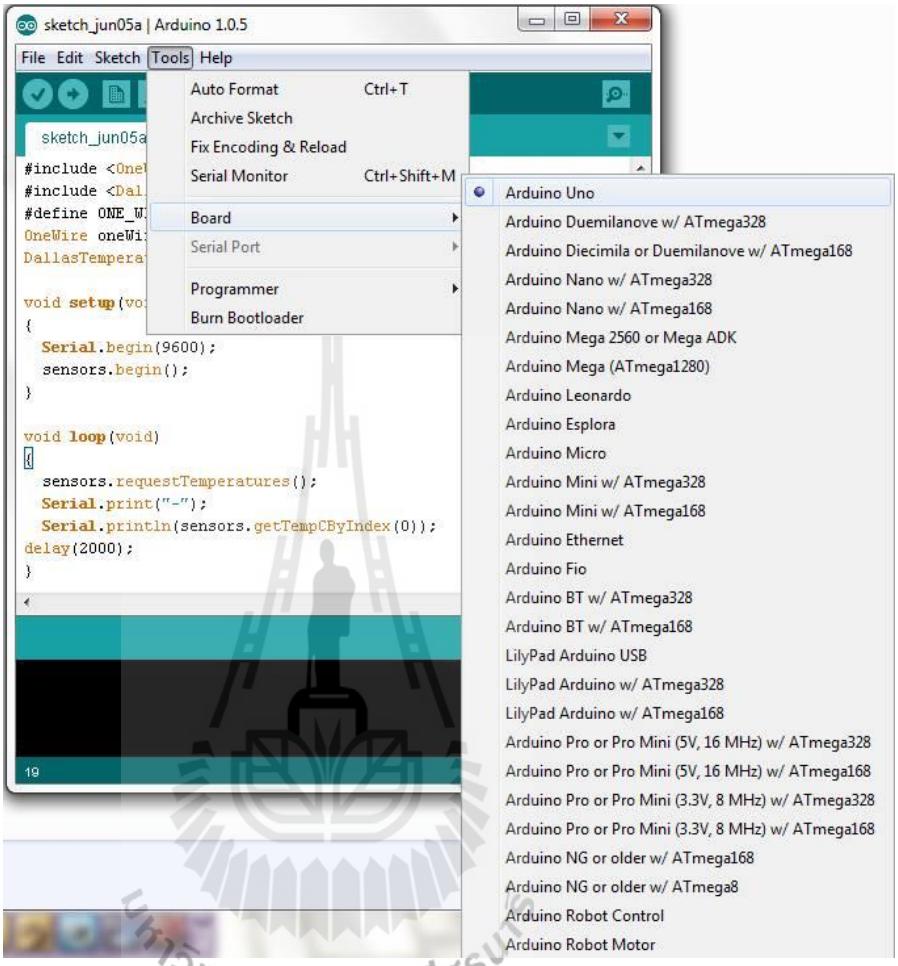
1. เปิดโปรแกรม Arduino IDE ที่ได้โหลดมาจาก http://Arduino.cc/en/Main/Software

2. เมื่อเปิดโปรแกรมแล้วจะพบกับหน้าต่างของ IDE ดังรูป



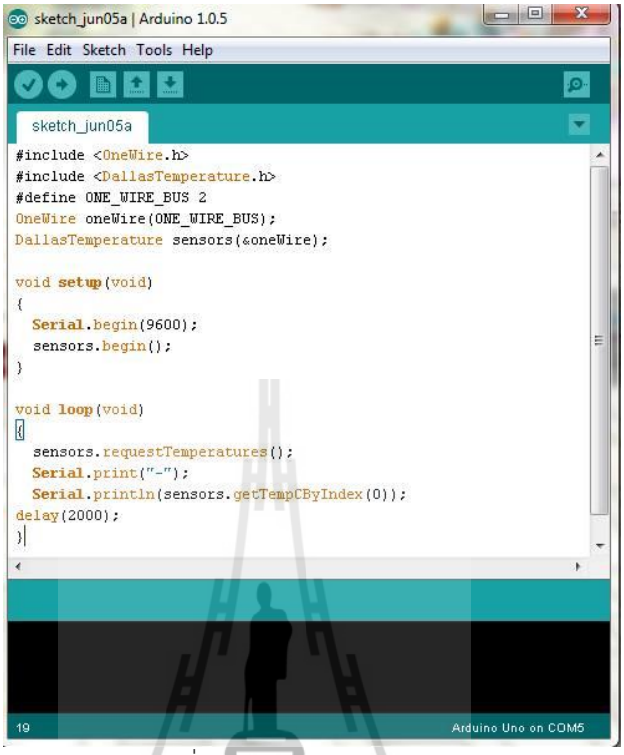
รูปที่ 3.46 หน้าต่างโปรแกรม

3. ไปที่ Tools->Board แล้วเลือกให้ตรงกับบอร์ดที่ใช้งาน สำหรับ Arduino UNO R3 ให้เลือกบอร์ด Arduino UNO



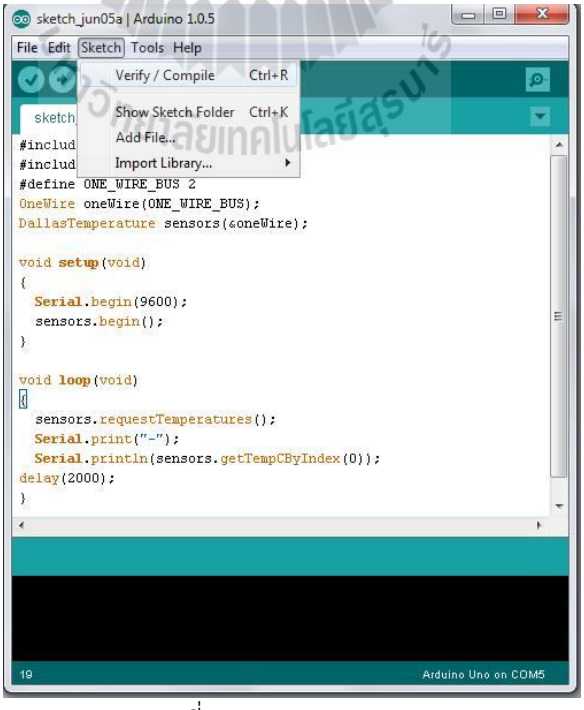
รูปที่ 3.47 เลือกบอร์ด

4. เขียนโปรแกรม



รูปที่ 3.48 การเขียนโปรแกรม

5. จากนั้นคอมไฟล์โปรแกรมโดยไปที่ Sketch->Verify / Compile

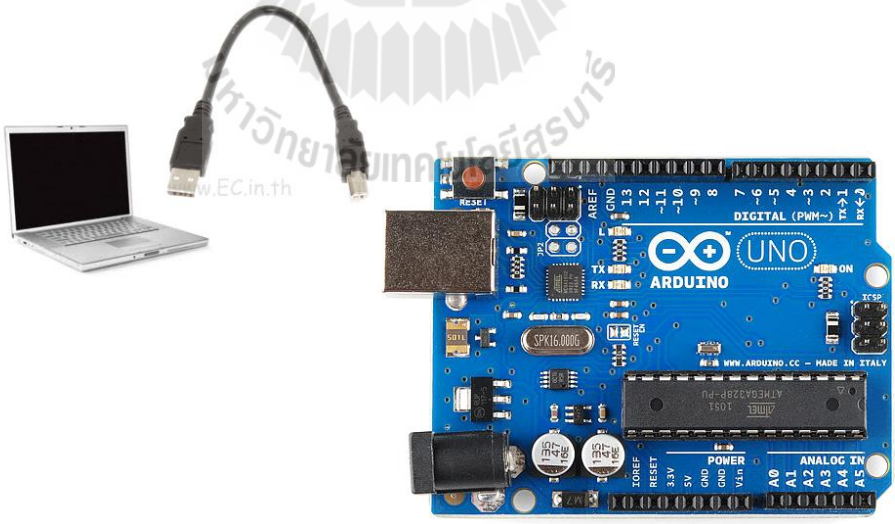


รูปที่ 3.49 การ compile

6. เมื่อคอมไพล์เรียบร้อยจะมีข้อความปรากฏดังรูป

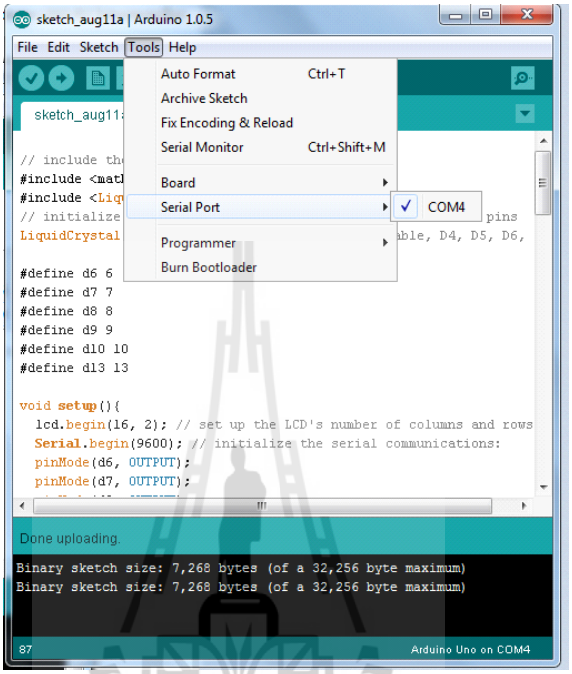
รูปที่ 3.50 การ compile เสร็จเรียบร้อย

7. ต่อบอร์ด Arduino UNO R3 เข้ากับคอมพิวเตอร์ผ่านทางพอร์ต USB



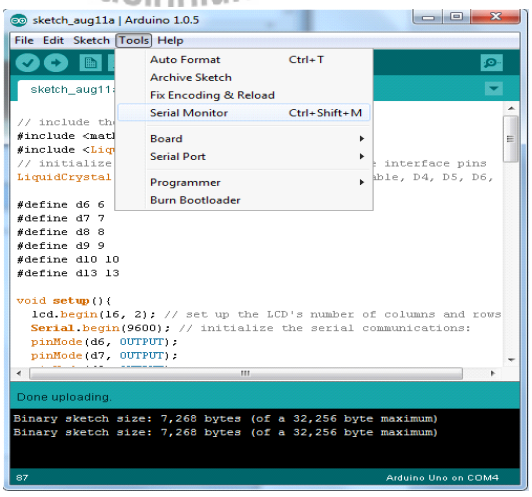
รูปที่ 3.51 การต่อ Arduino กับ คอมพิวเตอร์

8. จากนั้นให้ไปที่ Tools->Serial Port และเลือกให้ตรงกับบอร์ด Arduino UNO ที่ใช้งาน (สำหรับบอร์ด Arduino UNO R3 โปรแกรมจะเลือกให้อัตโนมัติ)



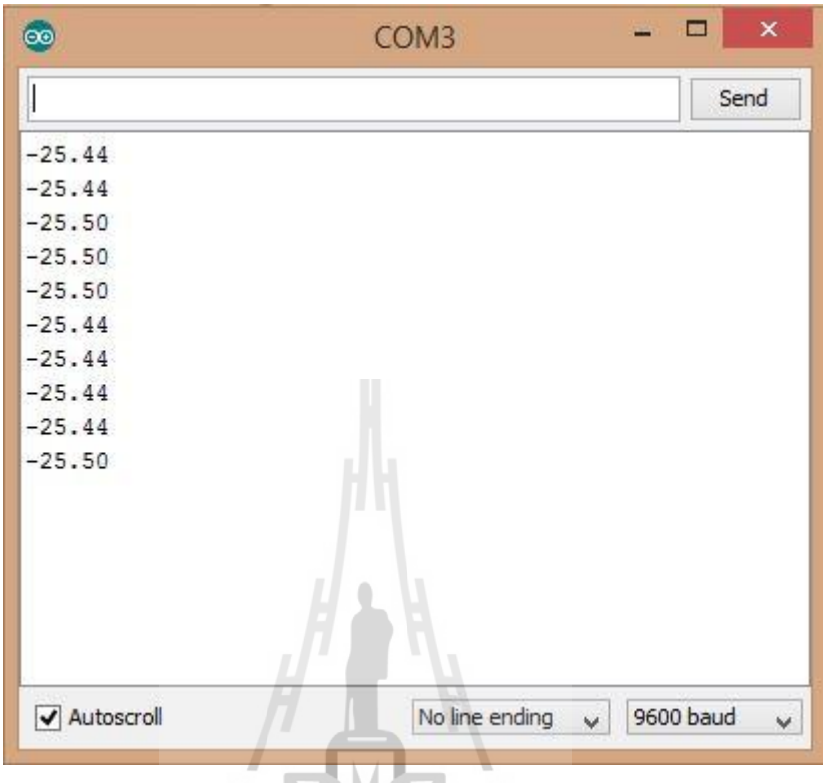
รูปที่ 3.52 การเลือก serialport

9. โหลดโปรแกรมเข้าบอร์ด Arduino UNO โดยไปที่ File->Upload
10. จากนั้นเปิด Serial Monitor ของ Arduino IDE โดยไปที่ Tools->Serial Monitor



รูปที่ 3.53 วิธีการแสดงผลที่ serial monitor

11. เมื่อเปิด Serial Monitor จะได้ข้อความดังรูป



รูปที่ 3.54 แสดงผลที่ serial monitor

บทที่ 4
การทดสอบระบบตรวจวัดอุณหภูมิอัตโนมัติโดยผ่านเครือข่าย xbee

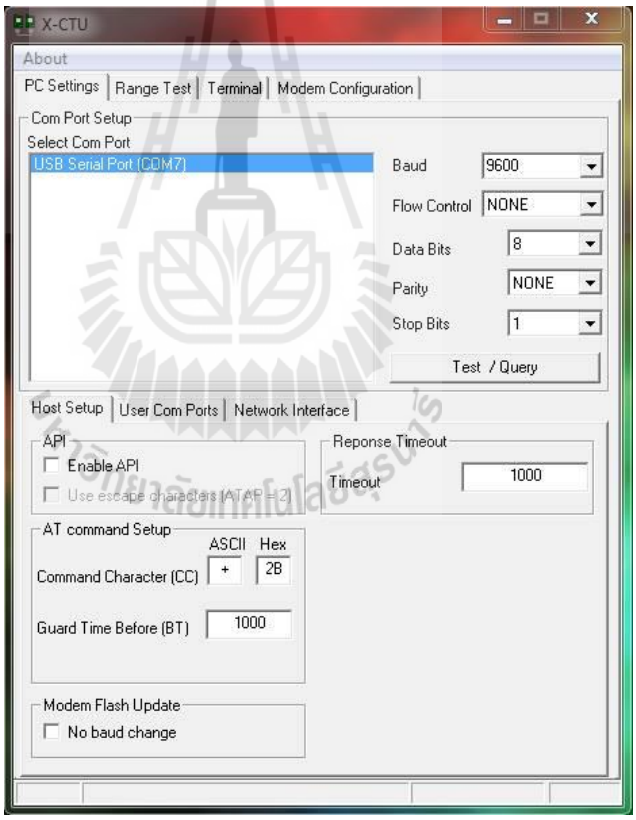
4.1 บทนำ

จากการศึกษาและทำความเข้าใจเกี่ยวกับทฤษฎีพื้นฐานในบทที่ 2 และ 3 นั้น ทำให้สามารถเชื่อมต่อวงจรโครงสร้างของระบบเสร็จสมบูรณ์พร้อมที่จะนำไปทดสอบการใช้งานจริงเพื่อให้บรรลุวัตถุประสงค์ของโครงงาน

4.2 การเชื่อมต่อวงจรของระบบในการทดลอง

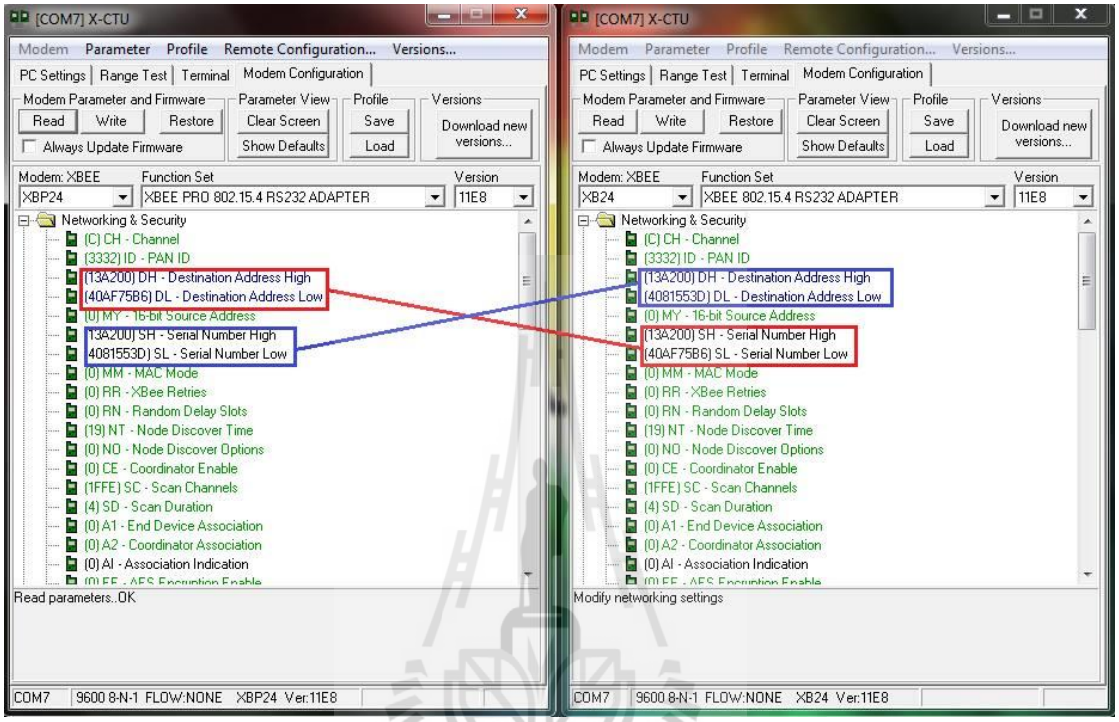
4.2.1 เชื่อมต่อ xbee เข้ากับคอมพิวเตอร์

เมื่อทำการเชื่อมต่อวงจรเสร็จเรียบร้อยแล้ว จะสามารถกำหนดค่าพารามิเตอร์ต่างๆได้เช่น Baudrate, Destination Address High, Destination Address Low เป็นต้น



รูปที่ 4.1 การปรับ Baudrate

ลำดับต่อมา จะเป็นการเซ็ตพารามิเตอร์ที่ทำให้ Xbee ทั้งภาคส่งภาครับ สามารถทำการเชื่อมต่อกันได้ โดยการเซ็ต DH DL ของภาคส่งให้ตรงกับ SH SL ของภาครับ และ เซ็ต DH DLของภาครับให้ตรงกับ SH SL ของภาคส่ง (รูปที่4.2) ซึ่งค่า SH SL ของ Xbee แต่ละตัวจะมีค่าต่างกัน สามารถดูได้ด้านหลังตัว Xbee (รูปที่4.3)

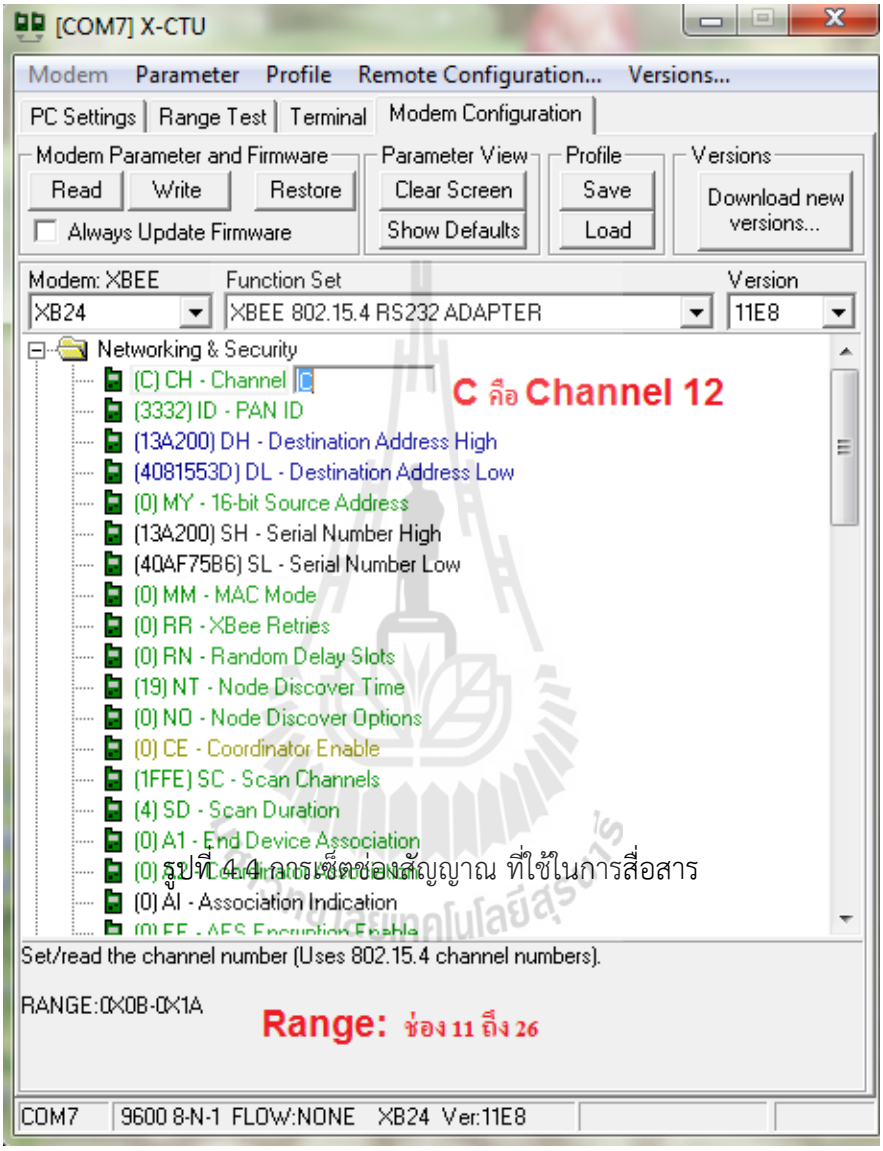


รูปที่ 4.2 เป็นการเซ็ต Destination Address

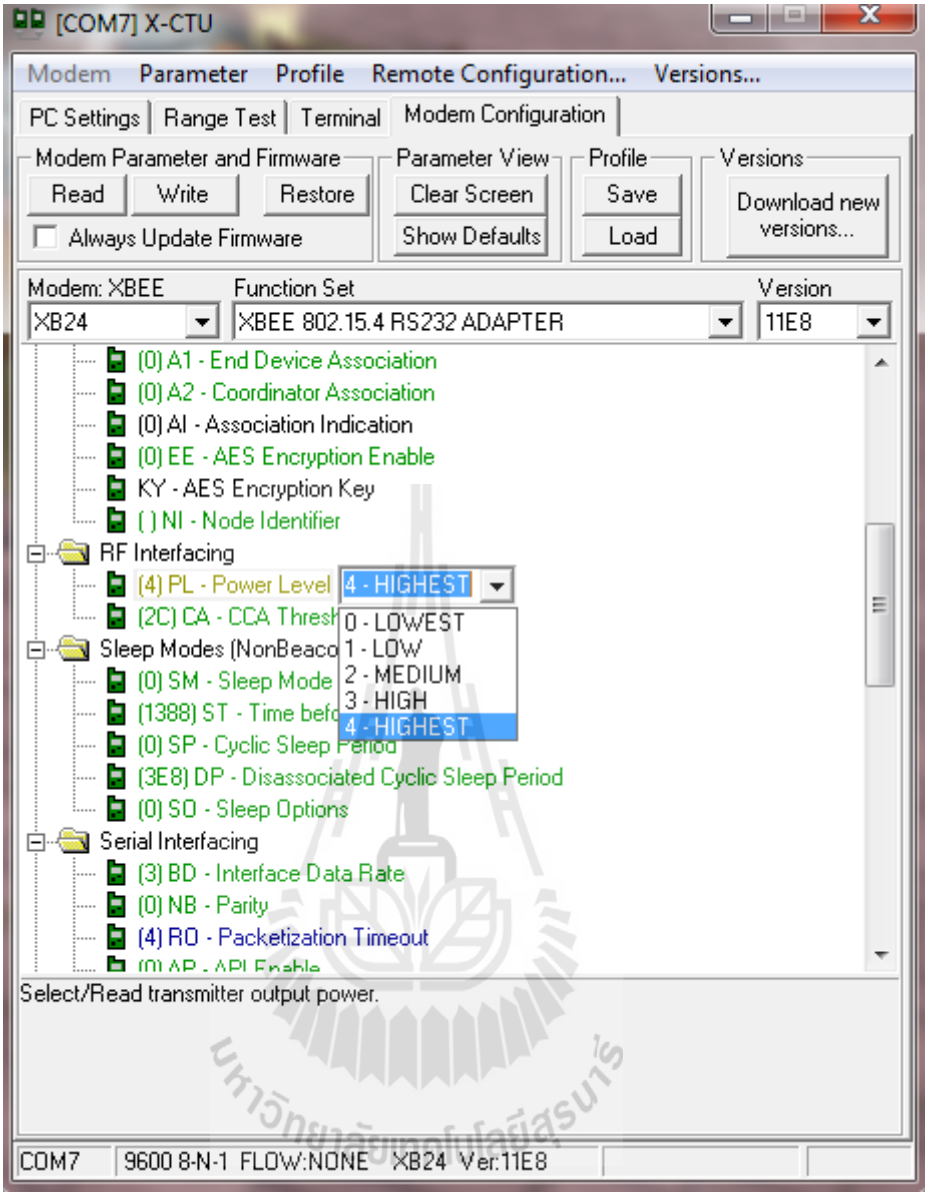
รูปที่ 4.3 แสดงค่า SL SH ของ Xbee

ค่าSH SL ของ Xbee ทั้งภาครับและภาคส่ง

ลำดับต่อมาจะเป็นการเซ็ตพารามิเตอร์ต่างๆ ที่จำเป็นต้องกำหนดเพื่อใช้ในการส่งข้อมูล เช่น การเซ็ตช่องสัญญาณ (รูปที่ 4.4) และ การเซ็ตกำลังงานที่ใช้ในการส่งสัญญาณ (รูปที่ 4.5)



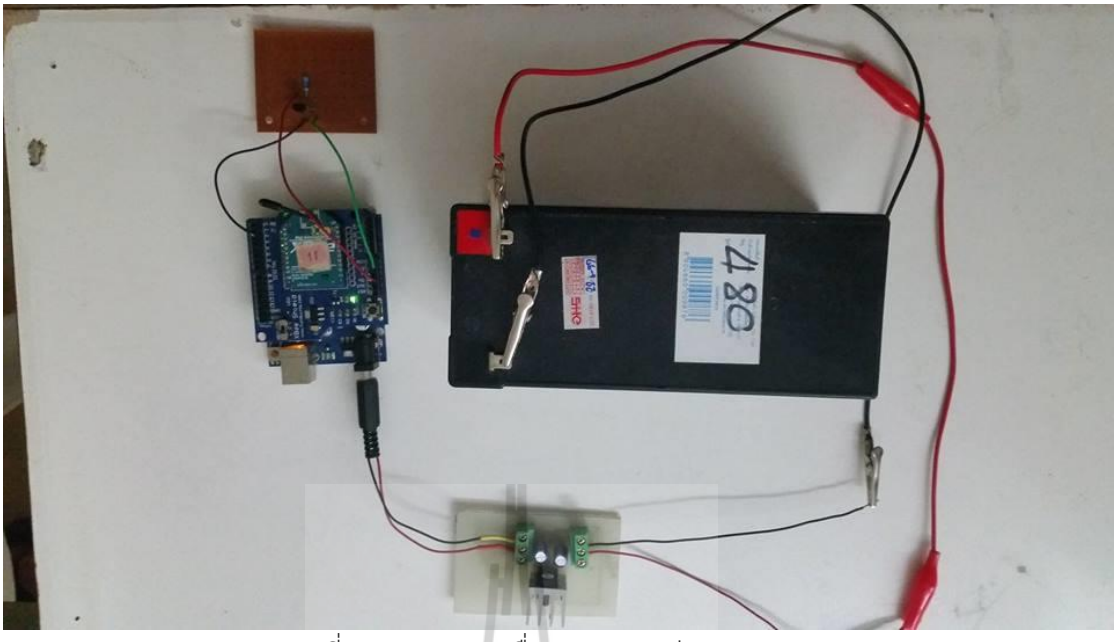
รูปที่ 4.4 การเซ็ตช่องสัญญาณ ที่ใช้ในการสื่อสาร



รูปที่ 4.5 การเซ็ตกำลังงานในการส่งข้อมูล (Power Level)

4.2.2 เชื่อมต่อวงจรภาคส่ง

ทำการเชื่อมต่อบอร์ด Arduino กับเซนเซอร์วัดอุณหภูมิ DS18B20 และเชื่อมต่อบอร์ด Arduino กับบอร์ด Xbee Shield ที่ต่อกับตัว Xbee เรียบร้อยแล้ว โดยใช้แบตเตอรี่ 12 v เป็นแหล่งจ่ายไฟให้กับบอร์ด Arduino และ Xbee ดังรูปที่ 4.6



รูปที่ 4.6 แสดงการเชื่อมวงจรภาคส่ง

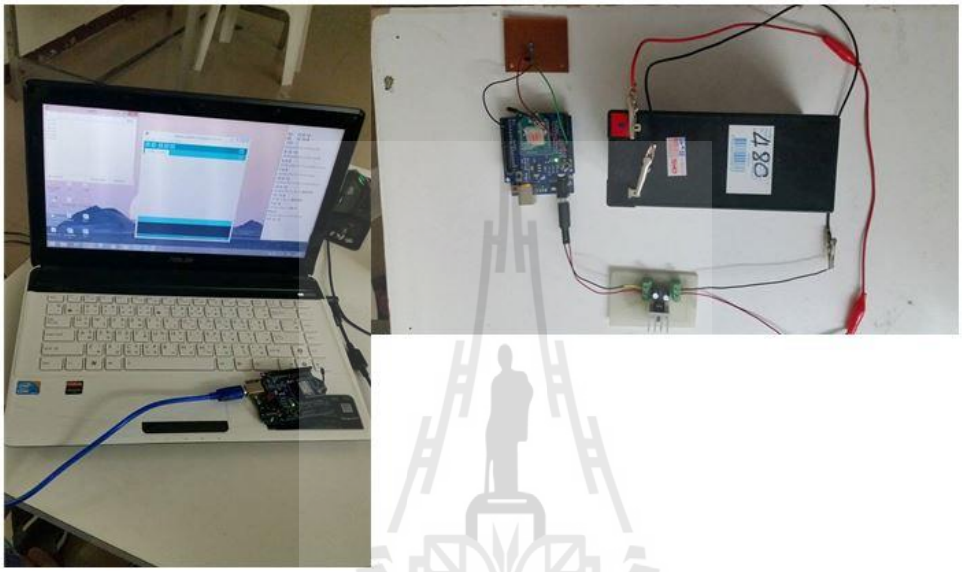
4.2.3 เชื่อมต่อวงจรภาครับ

ทำการเชื่อต่อ xbee กับ Arduino เข้ากับคอมพิวเตอร์เพื่อแสดงผลค่าอุณหภูมิที่รับได้ผ่านโปรแกรม Arduino ดังรูป4.7

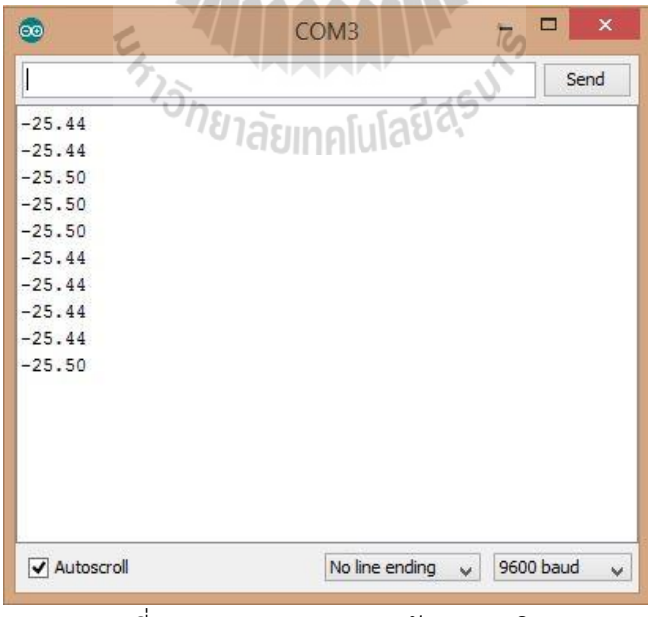
รูปที่ 4.7 เชื่อมต่อภาครับกับคอมพิวเตอร์

4.2.4 เชื่อมต่อวงจรภาครับกับภาคส่ง

ทำการเชื่อมต่อวงจรภาครับกับภาคส่ง ดังรูปที่ 4.8 การทำงานของวงจรคือเซนเซอร์ของภาคส่งทำการอ่านอุณหภูมิแล้วส่งค่าอุณหภูมิที่ได้ไปยัง บอร์ด Arduino ภาคส่ง จากนั้นบอร์ด Arduino จะทำการส่งค่าอุณหภูมิที่ได้ไปที่ Xbee แล้ว Xbee ของภาคส่งทำการส่งค่าอุณหภูมิไปยัง Xbee ของภาครับแบบไร้สาย (wireless) จากนั้น Xbee ของภาครับจะทำการส่งค่าอุณหภูมิที่ได้นี้ไปแสดงผลที่โปรแกรมArduino ทางหน้าจอคอมพิวเตอร์ ผลอุณหภูมิที่เซนเซอร์ตรวจจับได้แสดงดังรูปที่ 4.9 โดยแสดงค่าอุหณภูมิทุกๆ 3 วินาที



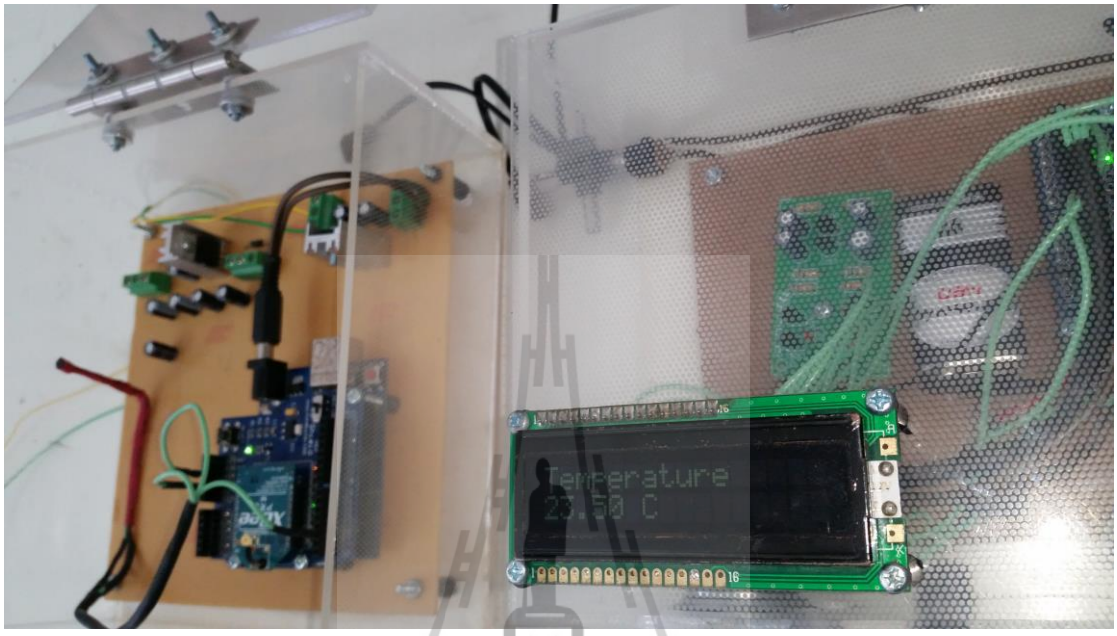
รูปที่ 4.8 แสดงรูปการเชื่อมวงจรภาครับกับภาคส่ง



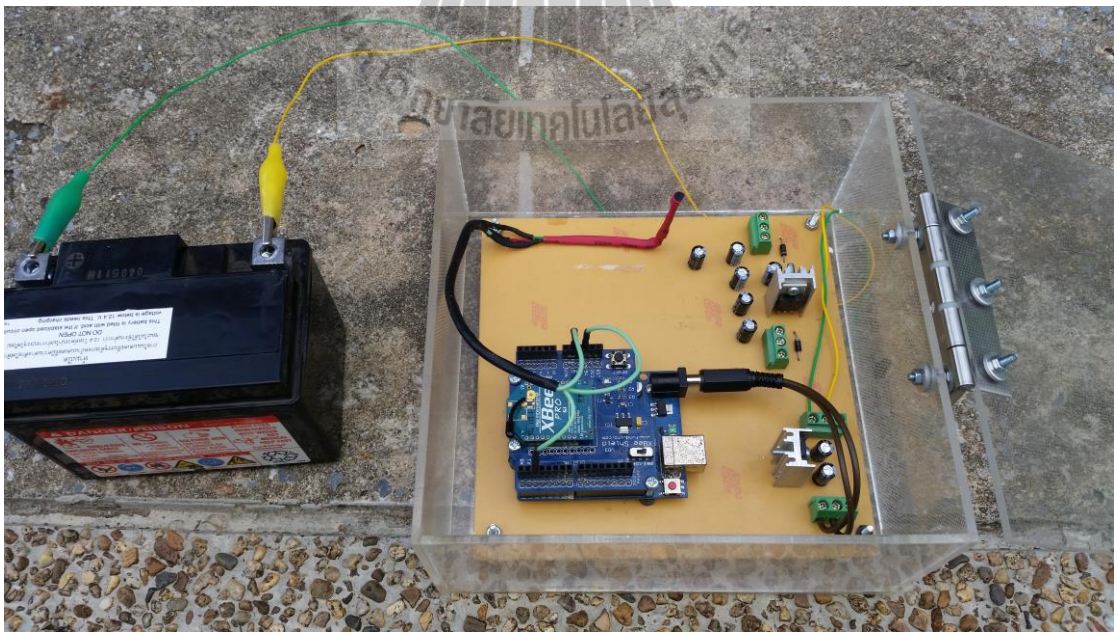
รูปที่ 4.9 แสดงผลการตรวจจับอุณหภูมิ

4.3 ผลการทดสอบโครงสร้างของระบบตรวจวัดอุณหภูมิอัตโนมัติโดยผ่านเครือข่าย xbeeโดยมีแหล่งจ่ายเป็นโซล่าเซลล์

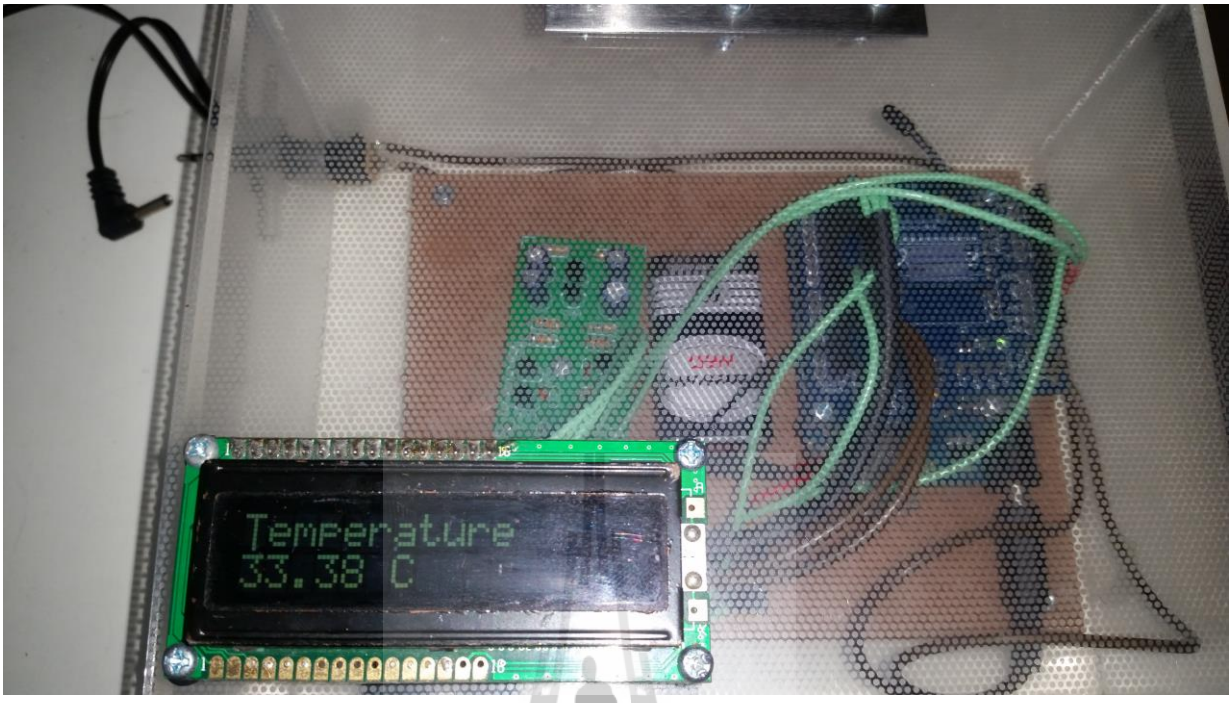
4.3.1การทดสอบการทำงานของวงจรแจ้งเตือนระบบตรวจวัดอุณหภูมิ



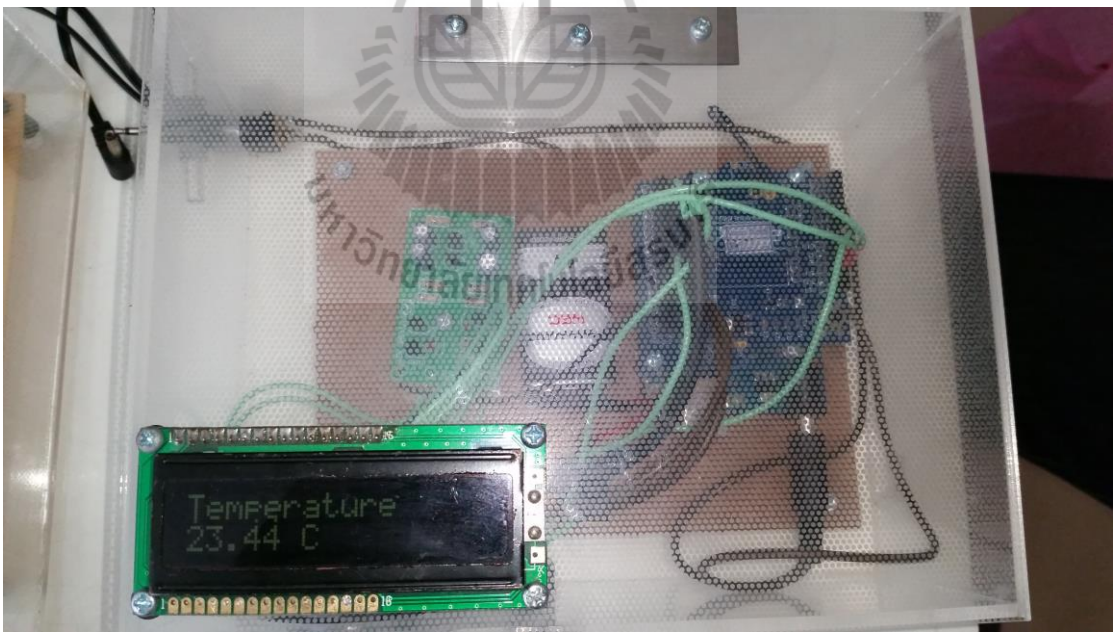
รูปที่ 4.10 ทดสอบวัดภาคส่งที่อุณหภูมิห้อง



รูปที่ 4.11 ทดสอบวัดภาคส่งที่อุณหภูมิภายนอก



รูปที่ 4.12 ภาครับรับอุณหภูมิจากภายนอกและมีการแจ้งเตือน



รูปที่ 4.13 ภาคส่งกลับมาวัดในอุณหภูมิห้องเหมือนเดิม

หลังจากการทดสอบการทำงานของวงจรแจ้งเตือนระบบตรวจวัดอุณหภูมิเสร็จสมบูรณ์แล้ว ต่อไปจะเป็นการทดสอบการใช้งานจริงในพื้นที่ต่างๆ โดยบริเวณที่เลือกใช้ในการในการทดสอบจะมี 2 ลักษณะ คือ 1.กรณี non line-of-sight (รอบๆอาคารและภายในอาคาร)

2.กรณี line-of-sight (ภายนอกอาคาร)

4.3.2 การทดสอบกรณีที่ 1 non line-of-sight

ภาครับจุด R ตั้งอยู่ภายในอาคาร แล้วภาคส่งอยู่บริเวณรอบๆอาคารเรียนรวม 2

รูปที่ 4.14 แสดงตำแหน่งทดสอบในบริเวณอาคารเรียนรวมสอง

รูปที่ 4.15 ภาคส่งอยู่ที่หน้าห้องศูนย์สหกิจศึกษา

รูปที่ 4.16 ภาคส่งอยู่หน้าห้อง Labcom 5

รูปที่ 4.17 ภาครับวางไว้อยู่ที่หน้าห้อง 600

ผลการทดสอบกรณี non line-of-sight

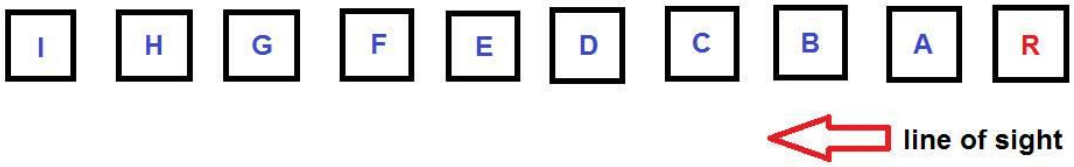
ตารางที่ 4.3.1 การทดสอบภายในอาคารและรอบๆอาคารเสมือนเป็น non line-of-sight

จุดทดสอบ	ระยะทาง (m)จาก อุปกรณ์ตัวส่ง	จำนวน ครั้งที่ส่ง	จำนวนครั้งที่รับได้				คิดเป็น %
			1	2	3	เฉลี่ย	
A หน้าศูนย์ สหกิจศึกษา	50	10	9	10	10	10	100
B ทางเดิน รวมหนึ่ง	80	10	6	9	9	8	80
C ป้ายรถเมล์เรียนรวมสอง	75	10	7	9	9	8.33	83.33
D ที่จอดรถรวมสอง	60	10	10	8	10	9.33	93.33
E ชั้นสองรวมสองจุดตรงกัน	5	10	10	9	10	9.66	96.66
F หน้าห้อง B5205	20	10	10	10	9	9.66	96.66
G หน้าห้อง B5209	40	10	7	9	10	8.66	86.66
H หน้าห้อง lab 5	40	10	9	6	6	7	70

นำข้อมูลจากตารางมาพล็อตกราฟได้ดังนี้

รูปที่ 4.18 กราฟแสดงความสัมพันธ์ระหว่างสถานที่กับการส่งข้อมูล

4.3.3 การทดสอบกรณีที่ 2 line-of-sight



รูปที่ 4.19 แสดงตำแหน่งทดสอบบริเวณรอบๆอาคารเรียนรวมสอง

หมายเหตุ: R คือที่เป็นอุปกรณ์ภาครับ และ A-I คือจุดที่เป็นอุปกรณ์ภาคส่ง

ทำการทดสอบในแต่ละตำแหน่งตามบริเวณที่ได้กำหนดไว้ สำหรับวิธีการส่งข้อมูลจากวงจรภาคส่งนั้นจะกำหนดการส่งข้อมูลเป็นครั้งละ 30 วินาที ซึ่งภายใน 30 วินาทีจะส่งข้อมูลทุกๆประมาณ 3 วินาที เนื่องจากเซ็ตโปรแกรมที่ภาคส่งใช้การหน่วงเวลาเป็น 3 วินาที โดยวงจรภาคส่งเป็นตัวส่งข้อมูลและวงจรภาครับเป็นตัวรับข้อมูลพร้อมทั้งแสดงผลทางโปรแกรม Arduino แล้วบันทึกข้อมูลจำนวนครั้งที่รับได้ในแต่ละตำแหน่ง

รูปที่ 4.20 การวัดในสถานที่จริง

ผลการทดสอบกรณี line-of-sight

ตารางที่ 4.3.2 การทดสอบภายนอกอาคารเสมือนเป็น line-of-sight

จุดทดสอบ	ระยะทาง(m)จากอุปกรณ์ตัวส่ง	จำนวนครั้งที่ส่ง	จำนวนครั้งที่รับได้				คิดเป็น %
			1	2	3	เฉลี่ย	
A	10	10	10	10	10	10	100
B	20	10	10	10	10	10	100
C	30	10	10	10	10	10	100
D	40	10	10	9	9	9.33	93.33
E	50	10	10	10	8	9.33	93.33
F	60	10	10	10	10	10	100
G	70	10	10	10	7	9	90
H	80	10	10	9	10	9.66	96.66
I	90	10	10	10	8	9.33	93.33

นำข้อมูลจากตารางมาพล็อตกราฟได้ดังนี้

รูปที่ 4.21 กราฟแสดงความสัมพันธ์

4.3.4 ทดสอบการทำงานของเซนเซอร์วัดอุณหภูมิของกรณี non line-of-sight

จะแบ่งเป็น 2 กรณีคือ

กรณีที่ 1 ทดสอบช่วงเวลากลางวัน ในเวลา 15.30 น.

กรณีที่ 2 ทดสอบช่วงเวลากลางคืน ในเวลา 19.00 น.

ตารางที่ 4.3.3 ค่าอุณหภูมิที่ตรวจจับได้ในเวลากลางวันที่เวลา 15.30 น. ในหน่วย (องศาเซลเซียส)

	จุดทดสอบ							
	A	B	C	D	E	F	G	H
ระยะทาง (เมตร)	50	80	75	60	5	20	40	40
ครั้งที่1	32.44	32.81	34.31	33.31	33.00	32.38	32.50	30.81
ครั้งที่2	32.44	32.88	34.63	33.31	32.94	32.19	31.44	30.81
ครั้งที่3	32.50	32.88	34.69	33.19	32.94	32.19	31.44	30.75
ครั้งที่4	32.50	32.94	34.69	33.19	32.94	32.19	31.44	30.75
ครั้งที่5	32.56	33.00	34.75	32.45	32.88	32.19	31.44	30.69
ครั้งที่6	32.63	33.25	34.75	32.31	32.69	32.13	31.50	30.75
ครั้งที่7	32.69	33.44	34.69	32.31	32.69	32.13	31.44	30.69
ครั้งที่8	32.69	33.50	34.75	32.25	32.63	32.13	31.37	30.62
ครั้งที่9	32.75	33.50	34.75	32.06	32.63	32.19	31.37	30.69
ครั้งที่10	32.81	33.44	34.75	32.19	32.56	32.19	31.25	30.69
เฉลี่ย	32.60	33.06	34.68	32.66	32.79	32.19	31.52	30.73
เฉลี่ยรวม	32.52							

หมายเหตุ

จุด A หน้าศูนย์ สหกิจศึกษา

จุด B ทางเดิน รวมหนึ่ง

จุด C ป้ายรถเมล์เรียนรวมสอง

จุด D ที่จอดรถรวมสอง

จุด E ชั้นสองรวมสองจุดตรงกัน

จุด F หน้าห้อง B5205

จุด G หน้าห้อง B5209

จุด H หน้าห้อง lab 5

ตารางที่ 4.3.4 ค่าอุณหภูมิที่ตรวจจับได้ในเวลากลางคืนที่เวลา 19.00 น.ในหน่วย (องศาเซลเซียส)

	จุดทดสอบ							
	A	B	C	D	E	F	G	H
ระยะทาง (เมตร)	50	80	75	60	5	20	40	40
ครั้งที่1	26.00	26.31	26.00	25.75	27.12	27.94	29.19	28.44
ครั้งที่2	26.00	26.31	25.87	25.75	27.12	27.94	29.19	28.44
ครั้งที่3	26.00	26.31	25.87	25.75	27.19	27.94	29.19	28.37
ครั้งที่4	26.00	26.50	25.87	25.69	27.19	28.00	29.19	28.31
ครั้งที่5	25.94	26.62	25.75	25.69	27.19	28.12	29.31	28.31
ครั้งที่6	25.94	26.62	25.69	25.75	27.19	28.12	29.37	28.06
ครั้งที่7	25.94	26.65	25.69	25.81	27.25	28.25	29.37	28.06
ครั้งที่8	25.94	26.65	25.69	25.87	27.31	28.27	29.44	28.06
ครั้งที่9	25.94	26.72	25.62	25.87	27.37	28.27	29.45	27.97
ครั้งที่10	25.94	26.69	25.62	25.94	27.37	28.30	29.45	27.81
เฉลี่ย	25.96	26.54	25.77	25.79	27.23	28.12	29.25	28.18
เฉลี่ยรวม	27.11							

หมายเหตุ

จุด A หน้าศูนย์ สหกิจศึกษา

จุด B ทางเดิน รวมหนึ่ง

จุด C ป้ายรถเมล์เรียนรวมสอง

จุด D ที่จอดรถรวมสอง

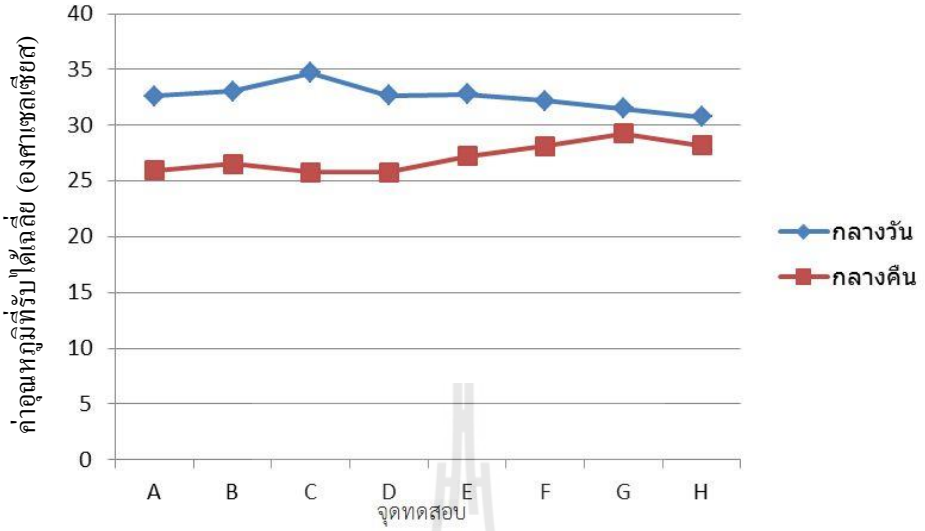
จุด E ชั้นสองรวมสองจุดตรงกัน

จุด F หน้าห้อง B5205

จุด G หน้าห้อง B5209

จุด H หน้าห้อง lab 5

นำข้อมูลจากตารางมาพล๊อตกราฟได้ดังนี้



รูปที่ 4.22 กราฟแสดงความสัมพันธ์ระหว่างอุณหภูมิเวลากลางวันกับเวลากลางคืน

4.3.5 ทดสอบการทำงานของเซนเซอร์วัดอุณหภูมิของกรณี line-of-sight

จะแบ่งเป็น 2 กรณีคือ

กรณีที่ 1 ทดสอบช่วงเวลากลางวัน ในเวลา 14.40 น.

กรณีที่ 2 ทดสอบช่วงเวลากลางคืน ในเวลา 20.00 น.

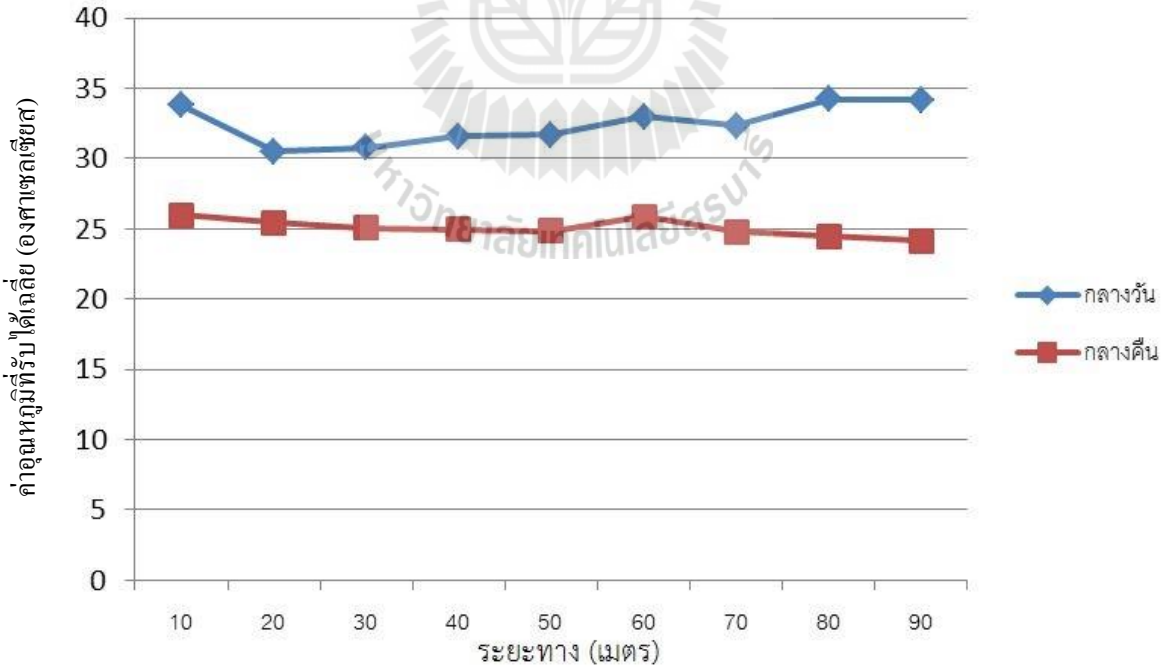
ตารางที่ 4.3.5 ค่าอุณหภูมิที่ตรวจจับได้ในเวลากลางวันที่เวลา 14.40 น.ในหน่วย (องศาเซลเซียส)

	ระยะทาง (เมตร)								
	10	20	30	40	50	60	70	80	90
ครั้งที่ 1	33.38	30.37	30.62	31.37	31.69	32.94	32.00	34.25	34.13
ครั้งที่ 2	33.44	30.44	30.69	31.44	31.69	32.88	32.06	34.25	34.25
ครั้งที่ 3	33.56	30.44	30.69	31.50	31.69	33.06	32.19	34.25	34.44
ครั้งที่ 4	33.63	30.50	30.69	31.56	31.69	32.94	32.25	34.25	34.13
ครั้งที่ 5	33.94	30.50	30.69	31.56	31.62	33.00	32.31	34.31	34.19
ครั้งที่ 6	34.00	30.56	30.69	31.62	31.62	33.06	32.38	34.31	34.32
ครั้งที่ 7	34.00	30.56	30.75	31.62	31.69	33.06	32.50	34.31	34.31
ครั้งที่ 8	34.06	30.56	30.87	31.62	31.75	33.00	32.56	34.19	34.75
ครั้งที่ 9	34.00	30.62	30.94	31.69	31.75	33.00	32.63	34.13	33.63
ครั้งที่ 10	34.00	30.62	31.00	31.75	31.81	33.00	32.63	33.94	33.40
เฉลี่ย	33.80	30.52	30.76	31.57	31.7	32.99	32.35	34.22	34.16
เฉลี่ยรวม	32.45								

ตารางที่ 4.3.6 ค่าอุณหภูมิที่ตรวจจับได้ในเวลากลางคืนที่เวลา 20.00 น.ในหน่วย (องศาเซลเซียส)

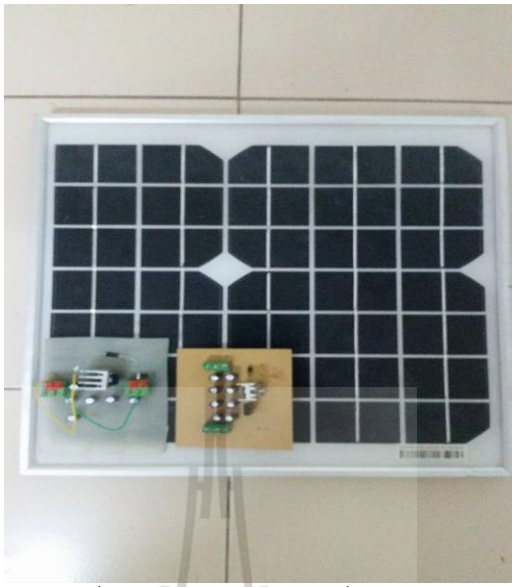
	ระยะทาง (เมตร)								
	10	20	30	40	50	60	70	80	90
ครั้งที่ 1	26.12	25.50	25.19	25.00	24.75	26.25	24.87	24.50	24.19
ครั้งที่ 2	26.06	25.50	25.19	25.00	24.69	26.25	24.87	24.44	24.19
ครั้งที่ 3	26.06	25.50	25.12	25.00	24.69	26.12	24.81	24.44	24.19
ครั้งที่ 4	26.06	25.50	25.06	24.94	24.69	26.00	24.81	24.44	24.12
ครั้งที่ 5	26.00	25.44	25.06	24.94	24.69	25.87	24.75	24.44	24.12
ครั้งที่ 6	26.00	25.44	25.06	24.94	24.75	25.87	24.75	24.44	24.12
ครั้งที่ 7	25.94	25.37	25.06	24.87	24.69	25.72	24.75	24.44	24.12
ครั้งที่ 8	25.87	25.37	25.06	24.87	25.44	25.87	24.69	24.44	24.12
ครั้งที่ 9	25.81	25.31	25.06	24.87	25.06	25.44	24.69	24.44	24.12
ครั้งที่ 10	25.69	25.25	25.06	24.87	25.12	25.37	24.69	24.44	24.12
เฉลี่ย	25.96	25.42	25.09	24.93	24.86	25.88	24.77	24.45	24.14
เฉลี่ยรวม	24.61								

นำข้อมูลจากตารางมาพล็อตกราฟได้ดังนี้



รูปที่ 4.23 กราฟแสดงความสัมพันธ์ระหว่างอุณหภูมิเวลากลางวันกับเวลากลางคืน

4.3.6 ทดสอบการทำงานของโซล่าเซลล์



รูปที่ 4.24 แผงโซล่าเซลล์ที่ใช้ทดสอบ

การทดสอบในการชาร์จจากแผงโซล่าเซลล์แบตเตอร์รี่ เราจะทำการทดลองแบ่งเป็น 2 กรณี คือ 1.ทดสอบชาร์จขณะที่มีความเข้มแสงน้อยหรือแสงแดดน้อย

2.ทดสอบการชาร์จขณะที่ความเข้าแสงมากหรือแสงแดดมาก

ผลการทดสอบมีดังนี้

ตารางที่ 4.3.7 ทดสอบการใช้งานโซล่าเซลล์ขณะมีแสงแดดน้อยหรือความเข้มแสงน้อย

บริเวณที่มีแสงน้อย	ครั้งที่ 1	ครั้งที่ 2	ค่าเฉลี่ย
อุณหภูมิเริ่มต้น	26.94 C	27.35 C	27.15 C
ระยะเวลาการชาร์จ	5 ชั่วโมง	5 ชั่วโมง	5 ชั่วโมง
ระยะเวลาการใช้งาน	2 ชั่วโมง	2 ชั่วโมงครึ่ง	2.25 ชั่วโมง

ตารางที่ 4.3.8 ทดสอบการใช้งานโซล่าเซลล์ขณะมีแสงแดดมากหรือความเข้มแสงมาก

บริเวณที่มีแสงมาก	ครั้งที่ 1	ครั้งที่ 2	ค่าเฉลี่ย
อุณหภูมิเริ่มต้น	39.67 C	39.13 C	39.40 C
ระยะเวลาการชาร์จ	5 ชั่วโมง	5 ชั่วโมง	5 ชั่วโมง
ระยะเวลาการใช้งาน	14 ชั่วโมง	15 ชั่วโมง	14 ชั่วโมงครึ่ง

4.4 วิเคราะห์ผลการทดลอง

4.4.1 วิเคราะห์การทดสอบโครงสร้างระบบการวัดอุณหภูมิโดยผ่านเครือข่ายไร้สาย ZigBee

จากการทดสอบภายในอาคาร (อาคารเรียนรวม2) ผลจากการทดสอบตามจุดต่างๆของอาคาร จะเห็นได้ว่าในชั้นเดียวกันกับบริเวณที่รับและบริเวณที่ทางตรงกับจุดรับจะส่งได้ดีกว่า บริเวณที่มีสิ่งกีดขวางเยอะ และในชั้นที่ 2 บริเวณที่ใกล้กับจุดส่งก็มีค่าที่รับได้ดีกว่า บริเวณที่มีสิ่งกีดขางเยอะเช่นกัน แต่อย่างไรก็ตาม จากผลการทดสอบข้อมูลที่ได้รับยังมีจำนวนมาก โดยเฉลี่ยประมาณ 85 % ของข้อมูลที่ส่ง

จากการทดสอบภายนอกอาคาร (รอบๆอาคารเรียนรวม2) ผลจากการทดสอบตามจุดต่างๆ จะเห็นได้ว่า จุด B กับจุด C เป็นจุด 2 จุดที่อยู่ไกลที่สุดจากจุดภาครับ อาจจะเป็นเพราะมีสิ่งกีดขวางเป็นอาคารกับสิ่งแวดล้อมภายนอก เช่น ต้นไม้ ทำให้ส่งได้ลดลง จากผลการทดสอบข้อมูลที่ได้รับยังมีจำนวนมาก โดยเฉลี่ยประมาณ 80 % ของข้อมูลที่ส่ง และจุด A กับจุด D อยู่ใกล้กับภาครับมากกว่าอีก 2 จุดทำให้ส่งข้อมูลได้เกือบ 100 %

จากการทดสอบภายนอกอาคาร (ทางเดินเรียนรวม2) เสมือนเป็น line-of-sight ผลจากการทดสอบตามจุดต่างๆ จะเห็นได้ว่าจุดทดสอบจะเป็นลักษณะเส้นตรงและระยะห่างมากขึ้นเรื่อยๆ ผลที่ได้คือในระยะใกล้ๆกับอุปกรณ์ทดสอบ ข้อมูลที่ได้รับมีค่าเท่ากับจำนวนข้อมูลที่ส่ง และมีค่าเท่ากัน 3 จุดคือ A,B และ C และจุดอื่นๆที่ห่างออกไปก็จะเห็นได้ว่าได้รับสัญญาณน้อยลงเรื่อยๆตามลำดับซึ่งจากการทดสอบได้ระยะทางถึงประมาณ 90 เมตรยังสามารถรับข้อมูลได้และคาดว่ายังรับข้อมมูลได้อีก ซึ่งจะเห็นได้ว่าค่าเฉลี่ยในการส่งแปรผกผันกับระยะทาง

4.4.2 วิเคราะห์การทำงานของเซนเซอร์วัดอุณหภูมิ

จากการทดลองได้ทำการวัดอุณหภูมิในเวลากลางวันในบริเวณอาคารเรียนรวม 2 เริ่มต้นวัดในเวลา 14.40 น. โดยค่าอุณหภูมิเฉลี่ยที่ได้จากการทดลองเท่ากับ 32.52 องศาเซลเซียส และจากการทดลองที่ทำการวัดค่าอุณหภูมิในเวลากลางคืนได้เริ่มต้นทำการทดลองที่เวลา 19.00 น. ได้อุณหภูมิเฉลี่ยเท่ากับ 27.11 องศาเซลเซียส

จากการทดลองได้ทำการวัดอุณหภูมิในเวลากลางคืนในบริเวณสนามหญ้า เริ่มต้นวัดในเวลา 15.30 น. โดยค่าอุณหภูมิเฉลี่ยที่ได้จากการทดลองเท่ากับ 32.45 องศาเซลเซียส และจากการทดลองที่ทำการวัดค่าอุณหภูมิในเวลากลางคืนได้เริ่มต้นทำการทดลองที่เวลา 20.00 น. ได้อุณหภูมิเฉลี่ยเท่ากับ 24.61 องศาเซลเซียส

4.4.3 วิเคราะห์การทำงานของโซล่าเซลล์

จากผลการทดสอบการชาร์จในบริเวณที่ความเข้มแสงน้อยที่อุณหภูมิเริ่มต้นที่ 27.15 องศาเซลเซียส โดยใช้เวลาในการชาร์จ 5 ชั่งโมง และเวลาในการใช้งาน 2.25 ชั่วโมง กับบริเวณที่มีความเข้มของแสงมากที่อุณหภูมิเริ่มต้นในเวลา 39.4 องศาเซลเซียส โดยใช้เวลาในการชาร์จ 5 ชั่งโมง และเวลาในการใช้งาน 14 ชั่วโมงครึ่ง

4.5 สรุปผลการทดลอง

4.5.1สรุปผลการทดสอบระบบการส่งและรับอุณหภูมิโดยผ่านเครือข่ายไร้สาย ZigBee

จากการทดสอบระบบการส่งและรับอุณหภูมิโดยผ่านเครือข่ายไร้สาย ZigBee เสร็จสมบูรณ์แล้วซึ่งการทดสอบในที่นี้มีการทดสอบอยู่ 3 พื้นที่คือ ภายในอาคาร ภายนอกอาคาร และภายนอกอาคารเสมือนเป็น line-of-sight พบว่าระบบการวัดอุณหภูมิโดยผ่านเครือข่ายไร้สายที่ใช้อุปกรณ์ Xbee เป็นตัวส่งและรับข้อมูลระหว่างกัน เมื่อทำการทดสอบระยะทางเพิ่มมากขึ้นทำให้การส่งและรับข้อมูลระหว่าง Xbee น้อยลง ทั้งนี้ยังรวมถึงสิ่งแวดล้อมที่มีสิ่งกีดขวางและสัญญาณรบกวนที่เกิดจากการใช้ช่องสัญญาณเดียวกันกับ Xbee เนื่องจาก Xbee ใช้ช่องสัญญาณในย่าน 2.4 GHz ซึ่งเป็นย่านเดียวกันกับ wireless ของตัวอาคารของการทดสอบ แต่จากที่ทำการตรวจสอบพบว่า wireless หรือ SUT-Wifi มีการใช้ ช่องความที่ 1 6 และ 11 ซึ่งต่างตาก ช่องสัญญาณของ Xbee จึงได้รับผลกระทบสัญญาณบ้าง แต่ไม่ค่อยมีผลเท่าไร แต่ส่วนที่ผลจริงๆ ก็อย่างการ การเกิด Multipath เนื่องจากการทดสอบภายในอาคาร จะเกิดการเลี้ยวเบนของคลื่น การสะท้อนของคลื่น เนื่องจาก ผนังของตัวอาคารนั่นเอง ซึ่งสังเกตได้จากการผลการทดลองทำให้การทดสอบภายในอาคารมีประสิทธิภาพน้อยลง จากการทดสอบทั้ง 3 พื้นที่ จะเห็นได้ว่า การทดสอบภายนอกอาคารเสมือนเป็น line-of-sight จะมีประสิทธิภาพการส่งได้ดีที่สุด เพราะไม่มีสิ่งกีดขวางและก็สัญญาณรบกวนภายในอาคาร

4.5.2 สรุปการทำงานของเซนเซอร์วัดอุณหภูมิ

จากตารางที่ 4.3.3 ตารางที่ 4.3.4 ตารางที่ 4.3.5 และตารางที่ 4.3.6 จะเห็นว่าค่าอุณหภูมิในตารางดังกล่าว มีค่าที่แตกต่างกันก็ออกไป ทั้งในเวลากลางวันและเวลากลางคืน ก็แสดงว่า ระบบวัดอุณหภูมิของเรานั้นทำงานได้จริง สามารถวัดอุณหภูมิได้จริง

4.5.3 สรุปผลการทดลองการทำงานของโซล่าเซลล์

จากผลการทดสอบการชาร์จในบริเวณที่ความเข้มแสงน้อยกับบริเวณที่มีความเข้มของแสงมากในอุณหภูมิ 39.4 จะเห็นได้ว่าบริเวณที่มีความเข้มแสงน้อยในอุณหภูมิ 27.15 เมื่อชาร์จในเวลาที่มีเท่าๆกันกับบริเวณที่มีความเข้มแสงมาก จะพบว่าเมื่อชาร์จในบริเวณที่มีความเข้มแสงมาก ระยะการใช้งานก็จะมาก ถ้าความเข้มแสงน้อย ระยะการใช้งานก็จะน้อยลงไปด้วย จะเห็นได้ว่าการเปรียบเทียบ 2 อย่างนี้แปรผันตรงกัน

บทที่ 5
สรุปผลการทดลองและข้อเสนอแนะ

5.1 บทนำ

เนื้อหาในบทนี้เป็นการกล่าวถึงบทสรุปของโครงงานระบบตรวจวัดอุณหภูมิอัตโนมัติโดยผ่านเครือข่าย xbeeวัดอุณหภูมิโดยมีแหล่งจ่ายเป็นโซล่าเซลล์ซึ่งประกอบด้วยข้อสรุปของโครงงาน ปัญหาที่พบขณะดำเนินงาน วิธีแก้ปัญหา ตลอดจนข้อเสนอแนะและวิธีพัฒนาโครงงานต่อไป

5.2 บทสรุปผลของโครงงาน

ระบบตรวจวัดอุณหภูมิอัตโนมัติโดยผ่านเครือข่าย xbee วัดอุณหภูมิโดยมีแหล่งจ่ายเป็นโซล่าเซลล์ที่สร้างขึ้นในโครงงานนี้ มีส่วนประกอบหลักๆ ดังนี้ 1.) บอร์ดไมโครคอนโทรเลอร์ Aduino UNO R3 2.) เซ็นเซอร์วัดอุณหภูมิ Ds18b20 3.) จอ LCD 16x2 4.) วงจรแจ้งเตือน (alarm) มีองค์ประกอบคือ รีเลย์ กับ ลำโพง โดยมีหลักการทำงานคือ เมื่อทำการวัดอุณหภูมิจากตัวเซนเซอร์ที่ภาคส่งจะส่งข้อมูลอุณหภูมิไปโดยใช้ zigbee โดยที่แหล่งจ่ายของภาคส่งคือ โซลล่าเซลล์ ในการส่งข้อมูลไร้สาย และที่ภาครับจะได้ข้อมูลที่รับได้มาแสดงผลบนจอ LCD โดยมีเงื่อนไขอยู่ว่าถ้าอุณหภูมิที่รับได้เกินจากที่กำหนดไว้ บอร์ด arduino UNO R3 จะทำการสั่งให้วงจรแจ้งเตือนดัง

การทำงานโซล่าเซลล์ในบริเวณที่ความเข้มแสงน้อยกับบริเวณที่มีความเข้มของแสงมาก จะเห็นได้ว่าบริเวณที่มีความเข้มแสงน้อยเมื่อชาร์จในเวลาที่มีเท่าๆกันกับบริเวณที่มีความเข้มแสงมาก จะพบว่าเมื่อชาร์จในบริเวณที่มีความเข้มแสงมาก ระยะการใช้งานก็จะมาก ถ้าความเข้มแสงน้อย ระยะการใช้งานก็จะน้อยลงไปด้วย

5.3 ปัญหาและอุปสรรค

ปัญหาและอุปสรรค	วิธีแก้ไข
1. ระยะทางในการวัด Xbee มีข้อจำกัด	1. จากการศึกษาข้อมูลพบว่าระยะทางที่สามรถวัดได้สามารถวัดได้ถึงประมาณ 90 เมตร ก็วัดตามสเปคของ Xbee
2.การชาร์จเตอร์รี่ไม่เข้า	2. ทำให้ศักย์จองโซล่าเซลล์สูงกว่าแบตเตอร์รี่จึงทำให้ชาร์จเข้า
3.สภาพอากาศในการชาร์จแบตเตอร์รี่จากแผงโซล่าเซลล์	3.รอสภาพอากาศในวันที่แสงแดดแรง มันจะทำให้มีความเข้มแสงในการชาร์จได้สูง
4. แบตเตอร์รี่ที่ใช้ในการส่งเข้าบอร์ด ardiono ไม่พอจากตอนแรกใช้แบตเตอร์รี่ 3.7 v 1000mAh 4 ก้อน	4. เพิ่มขนาดความจุของแบตเตอร์รี่เป็น 7.2 Ah และขนาดของแรงดันเป็น 12 V

5. กำลังของแผงโซลล่าเซลล์ที่ใช้ในการทดลองตอนแรกมีค่าน้อย	5.เปลี่ยนแผงโซล่าเซลล์ให้มีขนาดกำลังงานให้มีค่าสูงขึ้นจากตอนแรก จาก 1w เป็น 10w เพื่อใช้ในการชาร์จแบตเตอร์รี่ที่เร็วขึ้นและมีขนาดใหญ่ขึ้น

5.4 ข้อเสนอแนะ

1. ถ้าต้องการให้ได้ค่ากำลังงานในการชาร์จเร็วขึ้น ก็เพิ่มขนาดกำลังงานของแผงโซล่าเซลล์
2. สามารถเพิ่ม Function ให้กับเซนเซอร์ได้มากขึ้นอีก หรือเปลี่ยนเป็นเซนเซอร์ที่มีคุณภาพยิ่งขึ้นแต่ราคาก็สูงขึ้นด้วย
3. สามารถเพิ่ม Function ในการแจ้งเตือนได้อีก เช่น เพิ่มไฟ LED หรือวงจรเสียงที่มีเสียงดังขึ้น
4. สามารถเขียนโปรแกรมให้จอ LCD Freeze ค่าสูงสุดไว้เพื่อสะดวกต่อการอ่านค่า
5. สามารถเปลี่ยนตัวส่งและตัวรับข้อมูลไร้สาย (zigbee) ให้มีสายกาอาศที่มีกำลังงานส่งได้

บรรณานุกรม

[1] โซล่าเซลล์ [ออนไลน์] เข้าถึงได้จาก: https://sites.google.com/site/thethousandsunnyproject/thvsdi-thi-keiywkhxng/solar-cell

โซล่าเซลล์ [ออนไลน์] เข้าถึงได้จาก: https://sites.google.com/site/netceo2010/10

[2] Xbee [ออนไลน์] เข้าถึงได้จาก: http://www.thaieasyelec.com/Review-Product-Article/step-by-step-to-use-xbee-from-digi.html และ http://wara.com/article-736.html

[3] เซนเซอร์ DS18B20 [ออนไลน์] เข้าถึงได้จาก: http://www.hobbyist.co.nz/?q=temperature_sensor_ds18b20

[4] Code เซนเซอร์ DS18B20 [ออนไลน์] เข้าถึงได้จาก: http://www.ee.kmutnb.ac.th/eerobot/esl/learning/index.php?article=ds18b20-temperature-sensor

[5] Code Arduino [ออนไลน์] เข้าถึงได้จาก: http://www.arduino.cc/

ประวัติผู้เขียน

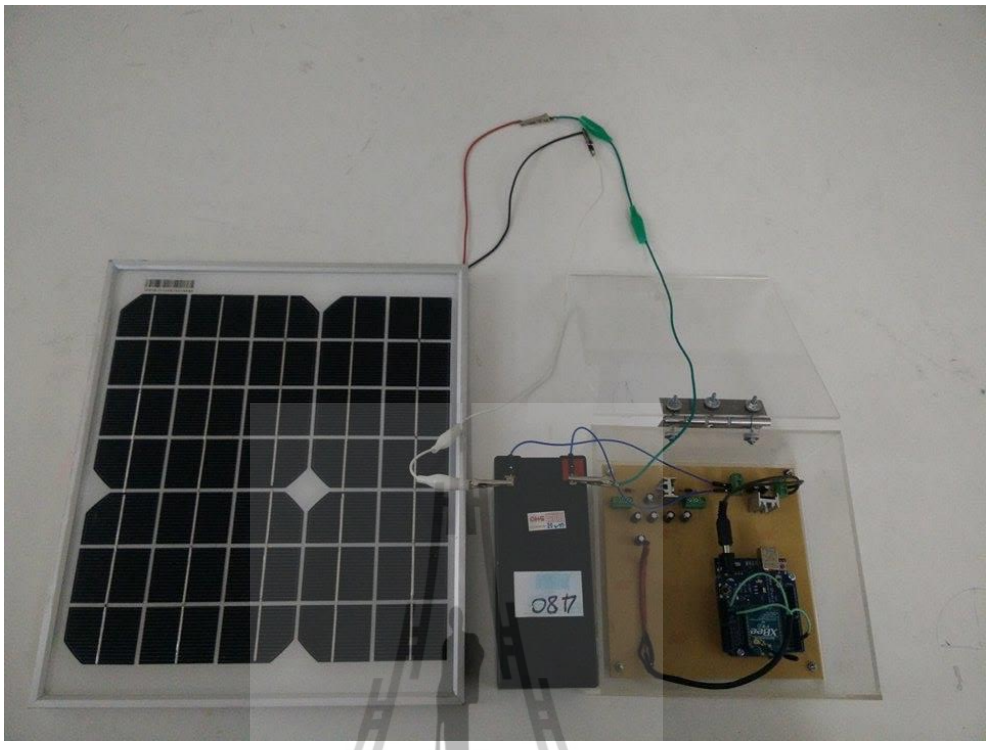
นาย สุภนัย กระจายศรี เกิดเมื่อวันที่ 8 เมษายน พ.ศ.2536 ภูมิลำเนาอยู่ที่ 46/7 ซ.พิมพา ต.เขาสามยอด อ.เมืองลพบุรี จังหวัดลพบุรี สำเร็จการศึกษาระดับมัธยมตอนปลายจากโรงเรียนพระนารายณ์ อ.เมือง จังหวัดลพบุรี ปัจจุบันเป็นนักศึกษาชั้นปีที่ 4 สาขาวิศวกรรมโทรคมนาคม สำนักวิชาวิศวกรรมศาสตร์ มหาวิทยาลัยเทคโนโลยีสุรนารี จังหวัดนครราชสีมา

นาย ศณัฐชะพล อัตตวนิช เกิดเมื่อวันที่ 5 สิงหาคม พ.ศ.2536 ภูมิลำเนาอยู่ที่ 2/237 หมู่ 13 ต.คลองหนึ่ง อ.คลองหลวง จังหวัดปทุมธานี สำเร็จการศึกษาระดับมัธยมตอนปลายจากโรงเรียนธรรมศาสตร์คลองหลวงวิทยาคม อ.คลองหลวง จังหวัดปทุมธานี ปัจจุบันเป็นนักศึกษาชั้นปีที่ 4 สาขาวิศวกรรมโทรคมนาคม สำนักวิชาวิศวกรรมศาสตร์ มหาวิทยาลัยเทคโนโลยีสุรนารี จังหวัดนครราชสีมา

นายกฤษฎา กลีบจำปา เกิดเมื่อวันที่ 20 มกราคม พ.ศ.2536 ภูมิลำเนาอยู่ที่ 63 หมู่ 3 ต.พิกุลทอง อ.ท่าช้าง จังหวัดสิงห์บุรี สำเร็จการศึกษาระดับมัธยมตอนปลายจากโรงเรียนพิบูลวิทยาลัยลพบุรี อ.เมือง จังหวัดลพบุรี ปัจจุบันเป็นนักศึกษาชั้นปีที่ 4 สาขาวิศวกรรมโทรคมนาคม สำนักวิชาวิศวกรรมศาสตร์ มหาวิทยาลัยเทคโนโลยีสุรนารี จังหวัดนครราชสีมา

ภาคผนวก

ภาคผนวก ก



รูปอุปกรณ์ภาคส่ง

รูปอุปกรณ์ภาครับ

ภาคผนวก ข

เซนเซอร์อุณหภูมิ Dsb1820

maxim integrated™

DS18B20
Programmable Resolution 1-Wire Digital Thermometer

DESCRIPTION

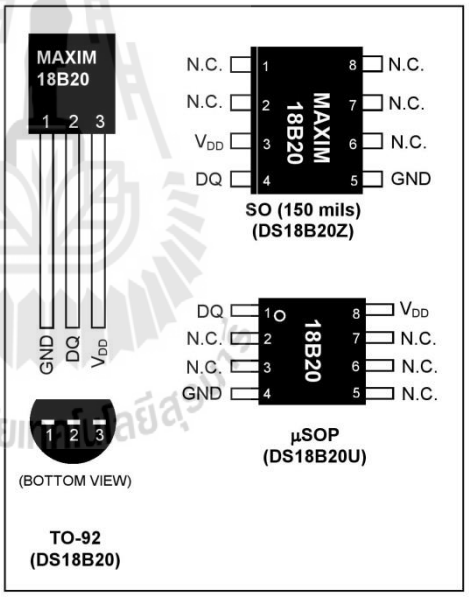
The DS18B20 digital thermometer provides 9-bit to 12-bit Celsius temperature measurements and has an alarm function with nonvolatile user-programmable upper and lower trigger points. The DS18B20 communicates over a 1-Wire bus that by definition requires only one data line (and ground) for communication with a central microprocessor. It has an operating temperature range of -55°C to +125°C and is accurate to ±0.5°C over the range of -10°C to +85°C. In addition, the DS18B20 can derive power directly from the data line ("parasite power"), eliminating the need for an external power supply.

Each DS18B20 has a unique 64-bit serial code, which allows multiple DS18B20s to function on the same 1-Wire bus. Thus, it is simple to use one microprocessor to control many DS18B20s distributed over a large area. Applications that can benefit from this feature include HVAC environmental controls, temperature monitoring systems inside buildings, equipment, or machinery, and process monitoring and control systems.

FEATURES

- Unique 1-Wire® Interface Requires Only One Port Pin for Communication
- Each Device has a Unique 64-Bit Serial Code Stored in an On-Board ROM
- Multidrop Capability Simplifies Distributed Temperature-Sensing Applications
- Requires No External Components
- Can Be Powered from Data Line; Power Supply Range is 3.0V to 5.5V
- Measures Temperatures from -55°C to +125°C (-67°F to +257°F)
- ±0.5°C Accuracy from -10°C to +85°C
- Thermometer Resolution is User Selectable from 9 to 12 Bits
- Converts Temperature to 12-Bit Digital Word in 750ms (Max)
- User-Definable Nonvolatile (NV) Alarm Settings
- Alarm Search Command Identifies and Addresses Devices Whose Temperature is Outside Programmed Limits (Temperature Alarm Condition)
- Available in 8-Pin SO (150 mils), 8-Pin µSOP, and 3-Pin TO-92 Packages
- Software Compatible with the DS1822
- Applications Include Thermostatic Controls, Industrial Systems, Consumer Products, Thermometers, or Any Thermally Sensitive System

PIN CONFIGURATIONS



1-Wire is a registered trademark of Maxim Integrated Products, Inc.

For pricing, delivery, and ordering information, please contact Maxim Direct at 1-888-629-4642, or visit Maxim's website at www.maximintegrated.com.

REV: 042208

ORDERING INFORMATION

PART	TEMP RANGE	PIN-PACKAGE	TOP MARK
DS18B20	-55°C to +125°C	3 TO-92	18B20
DS18B20+	-55°C to +125°C	3 TO-92	18B20
DS18B20/T&R	-55°C to +125°C	3 TO-92 (2000 Piece)	18B20
DS18B20+T&R	-55°C to +125°C	3 TO-92 (2000 Piece)	18B20
DS18B20-SL/T&R	-55°C to +125°C	3 TO-92 (2000 Piece)*	18B20
DS18B20-SL+T&R	-55°C to +125°C	3 TO-92 (2000 Piece)*	18B20
DS18B20U	-55°C to +125°C	8 µSOP	18B20
DS18B20U+	-55°C to +125°C	8 µSOP	18B20
DS18B20U/T&R	-55°C to +125°C	8 µSOP (3000 Piece)	18B20
DS18B20U+T&R	-55°C to +125°C	8 µSOP (3000 Piece)	18B20
DS18B20Z	-55°C to +125°C	8 SO	DS18B20
DS18B20Z+	-55°C to +125°C	8 SO	DS18B20
DS18B20Z/T&R	-55°C to +125°C	8 SO (2500 Piece)	DS18B20
DS18B20Z+T&R	-55°C to +125°C	8 SO (2500 Piece)	DS18B20

+Denotes a lead-free package. A "+" will appear on the top mark of lead-free packages.
T&R = Tape and reel.
**TO-92 packages in tape and reel can be ordered with straight or formed leads. Choose "SL" for straight leads. Bulk TO-92 orders are straight leads only.*

PIN DESCRIPTION

PIN			NAME	FUNCTION
SO	µSOP	TO-92		
1, 2, 6, 7, 8	2, 3, 5, 6, 7	—	N.C.	No Connection
3	8	3	V_{DD}	Optional V_{DD}. V_{DD} must be grounded for operation in parasite power mode.
4	1	2	DQ	Data Input/Output. Open-drain 1-Wire interface pin. Also provides power to the device when used in parasite power mode (see the *Powering the DS18B20* section.)
5	4	1	GND	Ground

OVERVIEW

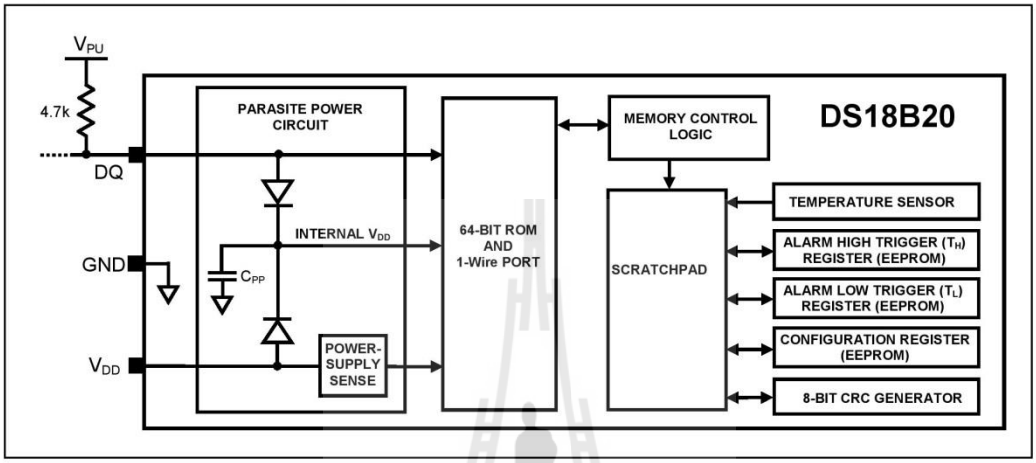
Figure 1 shows a block diagram of the DS18B20, and pin descriptions are given in the *Pin Description* table. The 64-bit ROM stores the device's unique serial code. The scratchpad memory contains the 2-byte temperature register that stores the digital output from the temperature sensor. In addition, the scratchpad provides access to the 1-byte upper and lower alarm trigger registers (T_H and T_L) and the 1-byte configuration register. The configuration register allows the user to set the resolution of the temperature-to-digital conversion to 9, 10, 11, or 12 bits. The T_H, T_L, and configuration registers are nonvolatile (EEPROM), so they will retain data when the device is powered down.

The DS18B20 uses Maxim's exclusive 1-Wire bus protocol that implements bus communication using one control signal. The control line requires a weak pullup resistor since all devices are linked to the bus via a 3-state or open-drain port (the DQ pin in the case of the DS18B20). In this bus system, the microprocessor (the master device) identifies and addresses devices on the bus using each device's unique 64-bit code. Because each device has a unique code, the number of devices that can be addressed on one

bus is virtually unlimited. The 1-Wire bus protocol, including detailed explanations of the commands and "time slots," is covered in the *1-Wire Bus System* section.

Another feature of the DS18B20 is the ability to operate without an external power supply. Power is instead supplied through the 1-Wire pullup resistor via the DQ pin when the bus is high. The high bus signal also charges an internal capacitor (C_{PP}), which then supplies power to the device when the bus is low. This method of deriving power from the 1-Wire bus is referred to as "parasite power." As an alternative, the DS18B20 may also be powered by an external supply on V_{DD}.

Figure 1. DS18B20 Block Diagram



OPERATION—MEASURING TEMPERATURE

The core functionality of the DS18B20 is its direct-to-digital temperature sensor. The resolution of the temperature sensor is user-configurable to 9, 10, 11, or 12 bits, corresponding to increments of 0.5°C, 0.25°C, 0.125°C, and 0.0625°C, respectively. The default resolution at power-up is 12-bit. The DS18B20 powers up in a low-power idle state. To initiate a temperature measurement and A-to-D conversion, the master must issue a Convert T [44h] command. Following the conversion, the resulting thermal data is stored in the 2-byte temperature register in the scratchpad memory and the DS18B20 returns to its idle state. If the DS18B20 is powered by an external supply, the master can issue "read time slots" (see the *1-Wire Bus System* section) after the Convert T command and the DS18B20 will respond by transmitting 0 while the temperature conversion is in progress and 1 when the conversion is done. If the DS18B20 is powered with parasite power, this notification technique cannot be used since the bus must be pulled high by a strong pullup during the entire temperature conversion. The bus requirements for parasite power are explained in detail in the *Powering the DS18B20* section.

The DS18B20 output temperature data is calibrated in degrees Celsius; for Fahrenheit applications, a lookup table or conversion routine must be used. The temperature data is stored as a 16-bit sign-extended two's complement number in the temperature register (see Figure 2). The sign bits (S) indicate if the temperature is positive or negative: for positive numbers S = 0 and for negative numbers S = 1. If the DS18B20 is configured for 12-bit resolution, all bits in the temperature register will contain valid data. For 11-bit resolution, bit 0 is undefined. For 10-bit resolution, bits 1 and 0 are undefined, and for 9-bit resolution bits 2, 1, and 0 are undefined. Table 1 gives examples of digital output data and the corresponding temperature reading for 12-bit resolution conversions.

Figure 2. Temperature Register Format

	BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
LS BYTE	2^3	2^2	2^1	2^0	2^{-1}	2^{-2}	2^{-3}	2^{-4}
	BIT 15	**BIT 14**	**BIT 13**	**BIT 12**	**BIT 11**	**BIT 10**	**BIT 9**	**BIT 8**
MS BYTE	S	S	S	S	S	2^6	2^5	2^4

S = SIGN

Table 1. Temperature/Data Relationship

TEMPERATURE (°C)	DIGITAL OUTPUT (BINARY)	DIGITAL OUTPUT (HEX)
+125	0000 0111 1101 0000	07D0h
+85*	0000 0101 0101 0000	0550h
+25.0625	0000 0001 1001 0001	0191h
+10.125	0000 0000 1010 0010	00A2h
+0.5	0000 0000 0000 1000	0008h
0	0000 0000 0000 0000	0000h
-0.5	1111 1111 1111 1000	FFF8h
-10.125	1111 1111 0101 1110	FF5Eh
-25.0625	1111 1110 0110 1111	FE6Fh
-55	1111 1100 1001 0000	FC90h

**The power-on reset value of the temperature register is +85°C.*

OPERATION—ALARM SIGNALING

After the DS18B20 performs a temperature conversion, the temperature value is compared to the user-defined two's complement alarm trigger values stored in the 1-byte T_H and T_L registers (see Figure 3). The sign bit (S) indicates if the value is positive or negative: for positive numbers S = 0 and for negative numbers S = 1. The T_H and T_L registers are nonvolatile (EEPROM) so they will retain data when the device is powered down. T_H and T_L can be accessed through bytes 2 and 3 of the scratchpad as explained in the *Memory* section.

Figure 3. T_H and T_L Register Format

BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
S	2^6	2^5	2^4	2^3	2^2	2^1	2^0

Only bits 11 through 4 of the temperature register are used in the T_H and T_L comparison since T_H and T_L are 8-bit registers. If the measured temperature is lower than or equal to T_L or higher than or equal to T_H, an alarm condition exists and an alarm flag is set inside the DS18B20. This flag is updated after every temperature measurement; therefore, if the alarm condition goes away, the flag will be turned off after the next temperature conversion.

The master device can check the alarm flag status of all DS18B20s on the bus by issuing an Alarm Search [ECh] command. Any DS18B20s with a set alarm flag will respond to the command, so the master can determine exactly which DS18B20s have experienced an alarm condition. If an alarm condition exists and the T_H or T_L settings have changed, another temperature conversion should be done to validate the alarm condition.

POWERING THE DS18B20

The DS18B20 can be powered by an external supply on the V_{DD} pin, or it can operate in "parasite power" mode, which allows the DS18B20 to function without a local external supply. Parasite power is very useful for applications that require remote temperature sensing or that are very space constrained. Figure 1 shows the DS18B20's parasite-power control circuitry, which "steals" power from the 1-Wire bus via the DQ pin when the bus is high. The stolen charge powers the DS18B20 while the bus is high, and some of the charge is stored on the parasite power capacitor (C_{PP}) to provide power when the bus is low. When the DS18B20 is used in parasite power mode, the V_{DD} pin must be connected to ground.

In parasite power mode, the 1-Wire bus and C_{PP} can provide sufficient current to the DS18B20 for most operations as long as the specified timing and voltage requirements are met (see the *DC Electrical Characteristics* and *AC Electrical Characteristics*). However, when the DS18B20 is performing temperature conversions or copying data from the scratchpad memory to EEPROM, the operating current can be as high as 1.5mA. This current can cause an unacceptable voltage drop across the weak 1-Wire pullup resistor and is more current than can be supplied by C_{PP}. To assure that the DS18B20 has sufficient supply current, it is necessary to provide a strong pullup on the 1-Wire bus whenever temperature conversions are taking place or data is being copied from the scratchpad to EEPROM. This can be accomplished by using a MOSFET to pull the bus directly to the rail as shown in Figure 4. The 1-Wire bus must be switched to the strong pullup within 10µs (max) after a Convert T [44h] or Copy Scratchpad [48h] command is issued, and the bus must be held high by the pullup for the duration of the conversion (t_{CONV}) or data transfer (t_{WR} = 10ms). No other activity can take place on the 1-Wire bus while the pullup is enabled.

The DS18B20 can also be powered by the conventional method of connecting an external power supply to the V_{DD} pin, as shown in Figure 5. The advantage of this method is that the MOSFET pullup is not required, and the 1-Wire bus is free to carry other traffic during the temperature conversion time.

The use of parasite power is not recommended for temperatures above +100°C since the DS18B20 may not be able to sustain communications due to the higher leakage currents that can exist at these temperatures. For applications in which such temperatures are likely, it is strongly recommended that the DS18B20 be powered by an external power supply.

In some situations the bus master may not know whether the DS18B20s on the bus are parasite powered or powered by external supplies. The master needs this information to determine if the strong bus pullup should be used during temperature conversions. To get this information, the master can issue a Skip ROM [CCh] command followed by a Read Power Supply [B4h] command followed by a "read time slot". During the read time slot, parasite powered DS18B20s will pull the bus low, and externally powered DS18B20s will let the bus remain high. If the bus is pulled low, the master knows that it must supply the strong pullup on the 1-Wire bus during temperature conversions.

Figure 4. Supplying the Parasite-Powered DS18B20 During Temperature Conversions

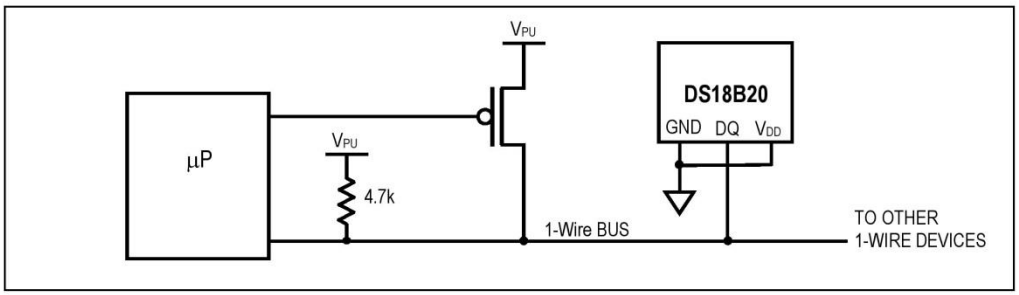
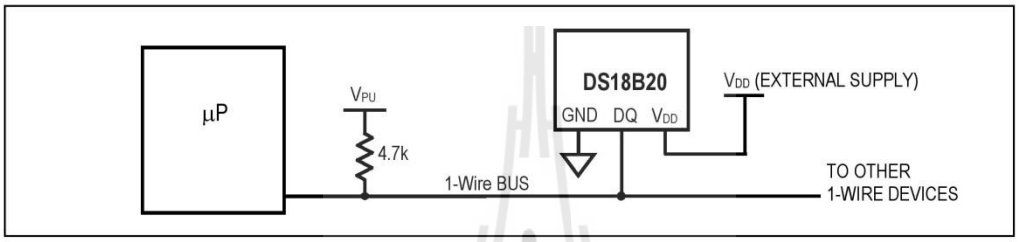


Figure 5. Powering the DS18B20 with an External Supply



64-BIT LASERED ROM CODE

Each DS18B20 contains a unique 64–bit code (see Figure 6) stored in ROM. The least significant 8 bits of the ROM code contain the DS18B20's 1-Wire family code: 28h. The next 48 bits contain a unique serial number. The most significant 8 bits contain a cyclic redundancy check (CRC) byte that is calculated from the first 56 bits of the ROM code. A detailed explanation of the CRC bits is provided in the *CRC Generation* section. The 64-bit ROM code and associated ROM function control logic allow the DS18B20 to operate as a 1-Wire device using the protocol detailed in the *1-Wire Bus System* section.

Figure 6. 64-Bit Lasered ROM Code

8-BIT CRC		48-BIT SERIAL NUMBER		8-BIT FAMILY CODE (28h)	
MSB	LSB	MSB	LSB	MSB	LSB

MEMORY

The DS18B20's memory is organized as shown in Figure 7. The memory consists of an SRAM scratchpad with nonvolatile EEPROM storage for the high and low alarm trigger registers (T_H and T_L) and configuration register. Note that if the DS18B20 alarm function is not used, the T_H and T_L registers can serve as general-purpose memory. All memory commands are described in detail in the *DS18B20 Function Commands* section.

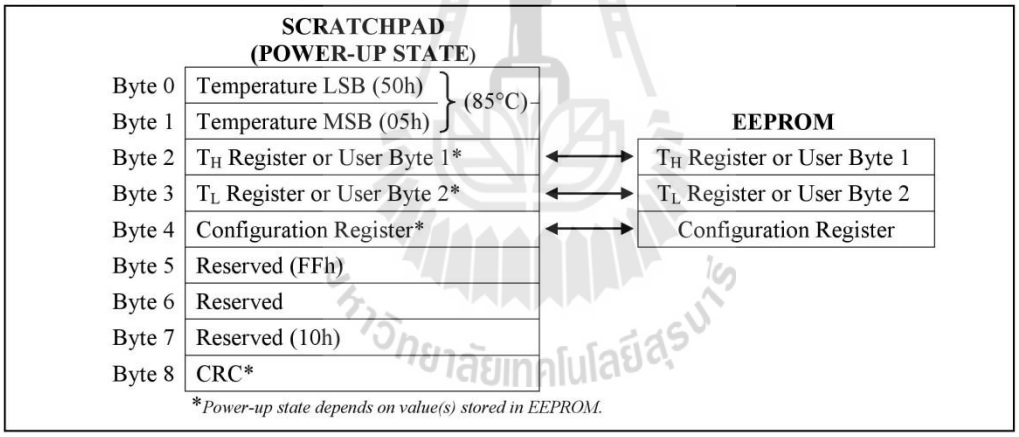
Byte 0 and byte 1 of the scratchpad contain the LSB and the MSB of the temperature register, respectively. These bytes are read-only. Bytes 2 and 3 provide access to T_H and T_L registers. Byte 4 contains the configuration register data, which is explained in detail in the *Configuration Register* section. Bytes 5, 6, and 7 are reserved for internal use by the device and cannot be overwritten.

Byte 8 of the scratchpad is read-only and contains the CRC code for bytes 0 through 7 of the scratchpad. The DS18B20 generates this CRC using the method described in the *CRC Generation* section.

Data is written to bytes 2, 3, and 4 of the scratchpad using the Write Scratchpad [4Eh] command; the data must be transmitted to the DS18B20 starting with the least significant bit of byte 2. To verify data integrity, the scratchpad can be read (using the Read Scratchpad [BEh] command) after the data is written. When reading the scratchpad, data is transferred over the 1-Wire bus starting with the least significant bit of byte 0. To transfer the T_H, T_L and configuration data from the scratchpad to EEPROM, the master must issue the Copy Scratchpad [48h] command.

Data in the EEPROM registers is retained when the device is powered down; at power-up the EEPROM data is reloaded into the corresponding scratchpad locations. Data can also be reloaded from EEPROM to the scratchpad at any time using the Recall E^2 [B8h] command. The master can issue read time slots following the Recall E^2 command and the DS18B20 will indicate the status of the recall by transmitting 0 while the recall is in progress and 1 when the recall is done.

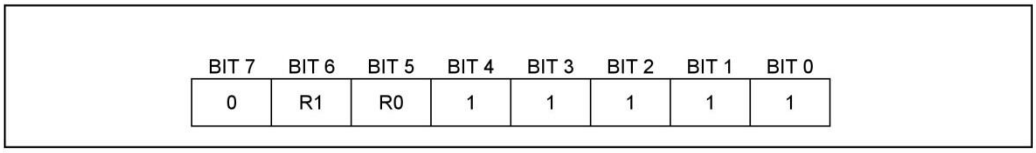
Figure 7. DS18B20 Memory Map



CONFIGURATION REGISTER

Byte 4 of the scratchpad memory contains the configuration register, which is organized as illustrated in Figure 8. The user can set the conversion resolution of the DS18B20 using the R0 and R1 bits in this register as shown in Table 2. The power-up default of these bits is R0 = 1 and R1 = 1 (12-bit resolution). Note that there is a direct tradeoff between resolution and conversion time. Bit 7 and bits 0 to 4 in the configuration register are reserved for internal use by the device and cannot be overwritten.

Figure 8. Configuration Register



BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
0	R1	R0	1	1	1	1	1

Table 2. Thermometer Resolution Configuration

R1	R0	RESOLUTION (BITS)	MAX CONVERSION TIME	
0	0	9	93.75ms	(t_{CONV}/8)
0	1	10	187.5ms	(t_{CONV}/4)
1	0	11	375ms	(t_{CONV}/2)
1	1	12	750ms	(t_{CONV})

CRC GENERATION

CRC bytes are provided as part of the DS18B20's 64-bit ROM code and in the 9th byte of the scratchpad memory. The ROM code CRC is calculated from the first 56 bits of the ROM code and is contained in the most significant byte of the ROM. The scratchpad CRC is calculated from the data stored in the scratchpad, and therefore it changes when the data in the scratchpad changes. The CRCs provide the bus master with a method of data validation when data is read from the DS18B20. To verify that data has been read correctly, the bus master must re-calculate the CRC from the received data and then compare this value to either the ROM code CRC (for ROM reads) or to the scratchpad CRC (for scratchpad reads). If the calculated CRC matches the read CRC, the data has been received error free. The comparison of CRC values and the decision to continue with an operation are determined entirely by the bus master. There is no circuitry inside the DS18B20 that prevents a command sequence from proceeding if the DS18B20 CRC (ROM or scratchpad) does not match the value generated by the bus master.

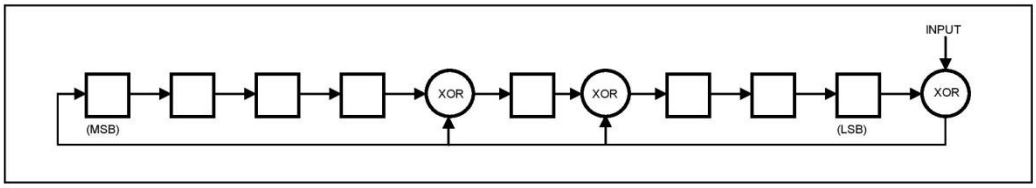
The equivalent polynomial function of the CRC (ROM or scratchpad) is:

$$CRC = X^8 + X^5 + X^4 + 1$$

The bus master can re-calculate the CRC and compare it to the CRC values from the DS18B20 using the polynomial generator shown in Figure 9. This circuit consists of a shift register and XOR gates, and the shift register bits are initialized to 0. Starting with the least significant bit of the ROM code or the least significant bit of byte 0 in the scratchpad, one bit at a time should shifted into the shift register. After shifting in the 56th bit from the ROM or the most significant bit of byte 7 from the scratchpad, the polynomial generator will contain the re-calculated CRC. Next, the 8-bit ROM code or scratchpad CRC from the DS18B20 must be shifted into the circuit. At this point, if the re-calculated CRC was correct, the shift register will contain all 0s. Additional information about the Maxim 1-Wire cyclic redundancy check

is available in *Application Note 27: Understanding and Using Cyclic Redundancy Checks with Maxim iButton Products*.

Figure 9. CRC Generator



1-WIRE BUS SYSTEM

The 1-Wire bus system uses a single bus master to control one or more slave devices. The DS18B20 is always a slave. When there is only one slave on the bus, the system is referred to as a "single-drop" system; the system is "multidrop" if there are multiple slaves on the bus.

All data and commands are transmitted least significant bit first over the 1-Wire bus.

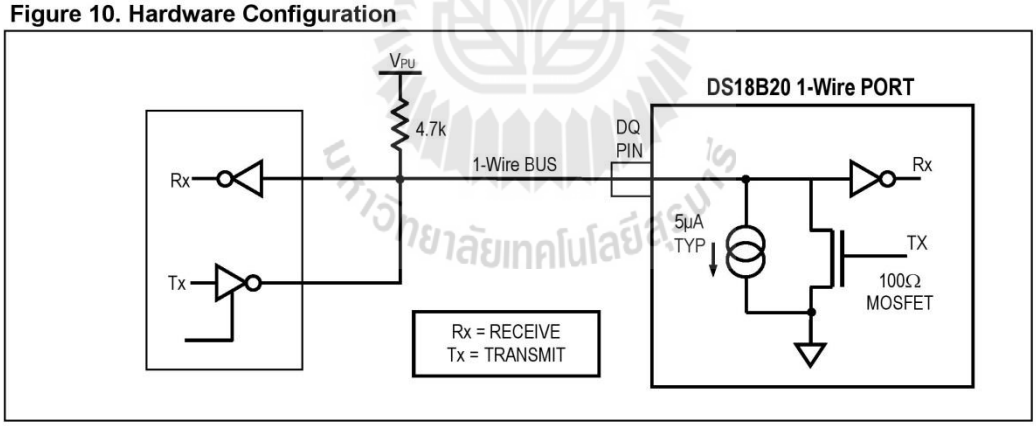
The following discussion of the 1-Wire bus system is broken down into three topics: hardware configuration, transaction sequence, and 1-Wire signaling (signal types and timing).

HARDWARE CONFIGURATION

The 1-Wire bus has by definition only a single data line. Each device (master or slave) interfaces to the data line via an open-drain or 3-state port. This allows each device to "release" the data line when the device is not transmitting data so the bus is available for use by another device. The 1-Wire port of the DS18B20 (the DQ pin) is open drain with an internal circuit equivalent to that shown in Figure 10.

The 1-Wire bus requires an external pullup resistor of approximately 5kΩ; thus, the idle state for the 1-Wire bus is high. If for any reason a transaction needs to be suspended, the bus MUST be left in the idle state if the transaction is to resume. Infinite recovery time can occur between bits so long as the 1-Wire bus is in the inactive (high) state during the recovery period. If the bus is held low for more than 480µs, all components on the bus will be reset.

Figure 10. Hardware Configuration



TRANSACTION SEQUENCE

The transaction sequence for accessing the DS18B20 is as follows:

Step 1. Initialization

Step 2. ROM Command (followed by any required data exchange)

Step 3. DS18B20 Function Command (followed by any required data exchange)

It is very important to follow this sequence every time the DS18B20 is accessed, as the DS18B20 will not respond if any steps in the sequence are missing or out of order. Exceptions to this rule are the Search ROM [F0h] and Alarm Search [ECh] commands. After issuing either of these ROM commands, the master must return to Step 1 in the sequence.

INITIALIZATION

All transactions on the 1-Wire bus begin with an initialization sequence. The initialization sequence consists of a reset pulse transmitted by the bus master followed by presence pulse(s) transmitted by the slave(s). The presence pulse lets the bus master know that slave devices (such as the DS18B20) are on the bus and are ready to operate. Timing for the reset and presence pulses is detailed in the *1-Wire Signaling* section.

ROM COMMANDS

After the bus master has detected a presence pulse, it can issue a ROM command. These commands operate on the unique 64-bit ROM codes of each slave device and allow the master to single out a specific device if many are present on the 1-Wire bus. These commands also allow the master to determine how many and what types of devices are present on the bus or if any device has experienced an alarm condition. There are five ROM commands, and each command is 8 bits long. The master device must issue an appropriate ROM command before issuing a DS18B20 function command. A flowchart for operation of the ROM commands is shown in Figure 11.

SEARCH ROM [F0h]

When a system is initially powered up, the master must identify the ROM codes of all slave devices on the bus, which allows the master to determine the number of slaves and their device types. The master learns the ROM codes through a process of elimination that requires the master to perform a Search ROM cycle (i.e., Search ROM command followed by data exchange) as many times as necessary to identify all of the slave devices. If there is only one slave on the bus, the simpler Read ROM command (see below) can be used in place of the Search ROM process. For a detailed explanation of the Search ROM procedure, refer to the *iButton® Book of Standards* at www.maxim-ic.com/ibuttonbook. After every Search ROM cycle, the bus master must return to Step 1 (Initialization) in the transaction sequence.

READ ROM [33h]

This command can only be used when there is one slave on the bus. It allows the bus master to read the slave's 64-bit ROM code without using the Search ROM procedure. If this command is used when there is more than one slave present on the bus, a data collision will occur when all the slaves attempt to respond at the same time.

MATCH ROM [55h]

The match ROM command followed by a 64-bit ROM code sequence allows the bus master to address a specific slave device on a multidrop or single-drop bus. Only the slave that exactly matches the 64-bit ROM code sequence will respond to the function command issued by the master; all other slaves on the bus will wait for a reset pulse.

SKIP ROM [CCh]

The master can use this command to address all devices on the bus simultaneously without sending out any ROM code information. For example, the master can make all DS18B20s on the bus perform simultaneous temperature conversions by issuing a Skip ROM command followed by a Convert T [44h] command.

Note that the Read Scratchpad [BEh] command can follow the Skip ROM command only if there is a single slave device on the bus. In this case, time is saved by allowing the master to read from the slave without sending the device's 64-bit ROM code. A Skip ROM command followed by a Read Scratchpad command will cause a data collision on the bus if there is more than one slave since multiple devices will attempt to transmit data simultaneously.

ALARM SEARCH [ECh]

The operation of this command is identical to the operation of the Search ROM command except that only slaves with a set alarm flag will respond. This command allows the master device to determine if any DS18B20s experienced an alarm condition during the most recent temperature conversion. After every Alarm Search cycle (i.e., Alarm Search command followed by data exchange), the bus master must return to Step 1 (Initialization) in the transaction sequence. See the *Operation—Alarm Signaling* section for an explanation of alarm flag operation.

DS18B20 FUNCTION COMMANDS

After the bus master has used a ROM command to address the DS18B20 with which it wishes to communicate, the master can issue one of the DS18B20 function commands. These commands allow the master to write to and read from the DS18B20's scratchpad memory, initiate temperature conversions and determine the power supply mode. The DS18B20 function commands, which are described below, are summarized in Table 3 and illustrated by the flowchart in Figure 12.

CONVERT T [44h]

This command initiates a single temperature conversion. Following the conversion, the resulting thermal data is stored in the 2-byte temperature register in the scratchpad memory and the DS18B20 returns to its low-power idle state. If the device is being used in parasite power mode, within 10µs (max) after this command is issued the master must enable a strong pullup on the 1-Wire bus for the duration of the conversion (t_{CONV}) as described in the *Powering the DS18B20* section. If the DS18B20 is powered by an external supply, the master can issue read time slots after the Convert T command and the DS18B20 will respond by transmitting a 0 while the temperature conversion is in progress and a 1 when the conversion is done. In parasite power mode this notification technique cannot be used since the bus is pulled high by the strong pullup during the conversion.

WRITE SCRATCHPAD [4Eh]

This command allows the master to write 3 bytes of data to the DS18B20's scratchpad. The first data byte is written into the T_H register (byte 2 of the scratchpad), the second byte is written into the T_L register (byte 3), and the third byte is written into the configuration register (byte 4). Data must be transmitted least significant bit first. All three bytes MUST be written before the master issues a reset, or the data may be corrupted.

READ SCRATCHPAD [BEh]

This command allows the master to read the contents of the scratchpad. The data transfer starts with the least significant bit of byte 0 and continues through the scratchpad until the 9th byte (byte 8 – CRC) is read. The master may issue a reset to terminate reading at any time if only part of the scratchpad data is needed.

COPY SCRATCHPAD [48h]

This command copies the contents of the scratchpad T_H, T_L and configuration registers (bytes 2, 3 and 4) to EEPROM. If the device is being used in parasite power mode, within 10µs (max) after this command is issued the master must enable a strong pullup on the 1-Wire bus for at least 10ms as described in the *Powering the DS18B20* section.

RECALL E^2 [B8h]

This command recalls the alarm trigger values (T_H and T_L) and configuration data from EEPROM and places the data in bytes 2, 3, and 4, respectively, in the scratchpad memory. The master device can issue read time slots following the Recall E^2 command and the DS18B20 will indicate the status of the recall by transmitting 0 while the recall is in progress and 1 when the recall is done. The recall operation happens automatically at power-up, so valid data is available in the scratchpad as soon as power is applied to the device.

READ POWER SUPPLY [B4h]

The master device issues this command followed by a read time slot to determine if any DS18B20s on the bus are using parasite power. During the read time slot, parasite powered DS18B20s will pull the bus low, and externally powered DS18B20s will let the bus remain high. See the *Powering the DS18B20* section for usage information for this command.

Table 3. DS18B20 Function Command Set

COMMAND	DESCRIPTION	PROTOCOL	1-Wire BUS ACTIVITYAFTER COMMAND IS ISSUED	NOTES
TEMPERATURE CONVERSION COMMANDS				
Convert T	Initiates temperature conversion.	44h	DS18B20 transmits conversion status to master (not applicable for parasite-powered DS18B20s).	1
MEMORY COMMANDS				
Read Scratchpad	Reads the entire scratchpad including the CRC byte.	BEh	DS18B20 transmits up to 9 data bytes to master.	2
Write Scratchpad	Writes data into scratchpad bytes 2, 3, and 4 (T_H, T_L, and configuration registers).	4Eh	Master transmits 3 data bytes to DS18B20.	3
Copy Scratchpad	Copies T_H, T_L, and configuration register data from the scratchpad to EEPROM.	48h	None	1
Recall E^2	Recalls T_H, T_L, and configuration register data from EEPROM to the scratchpad.	B8h	DS18B20 transmits recall status to master.	
Read Power Supply	Signals DS18B20 power supply mode to the master.	B4h	DS18B20 transmits supply status to master.	

Note 1: For parasite-powered DS18B20s, the master must enable a strong pullup on the 1-Wire bus during temperature conversions and copies from the scratchpad to EEPROM. No other bus activity may take place during this time.
Note 2: The master can interrupt the transmission of data at any time by issuing a reset.
Note 3: All three bytes must be written before a reset is issued.

Figure 11. ROM Commands Flowchart

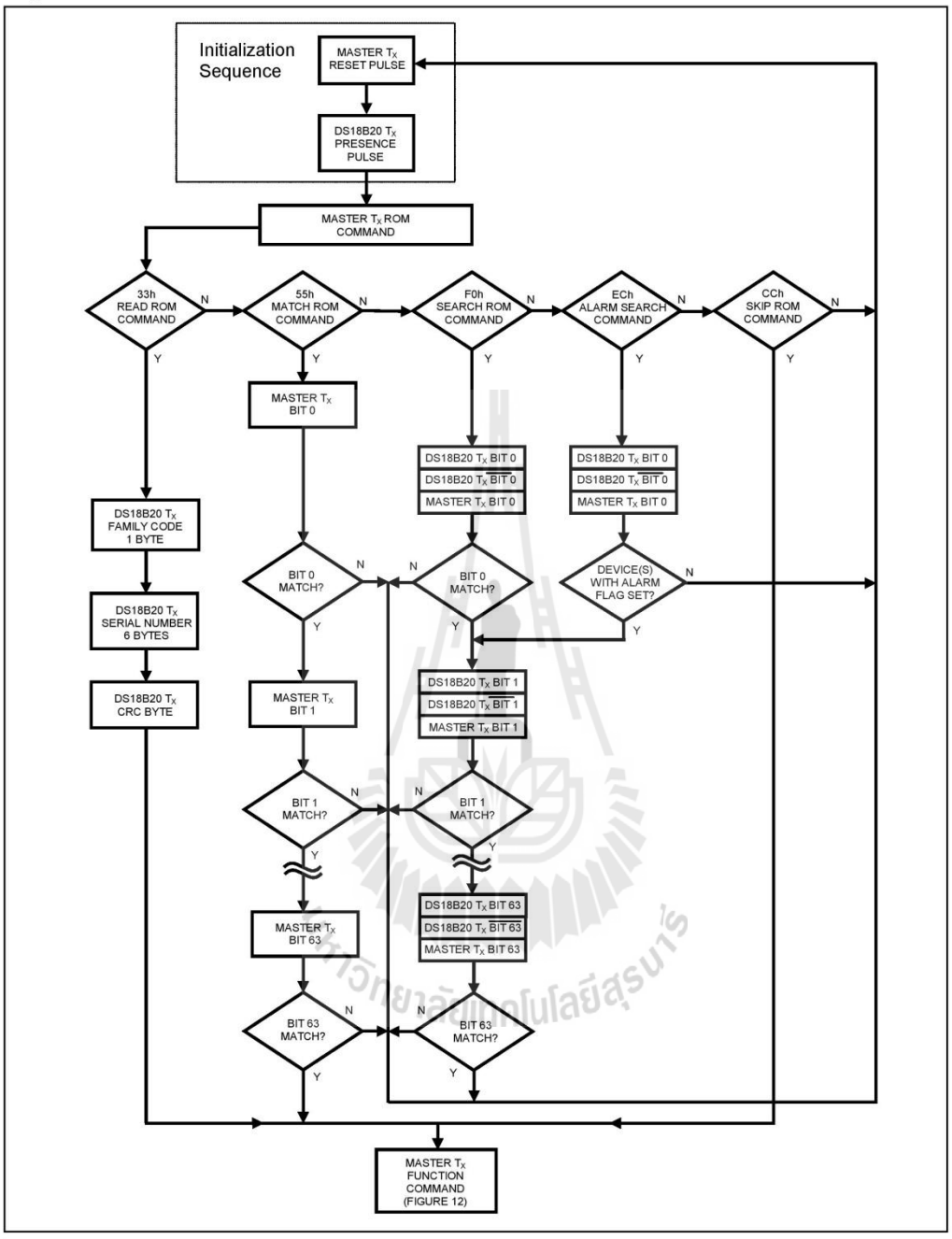
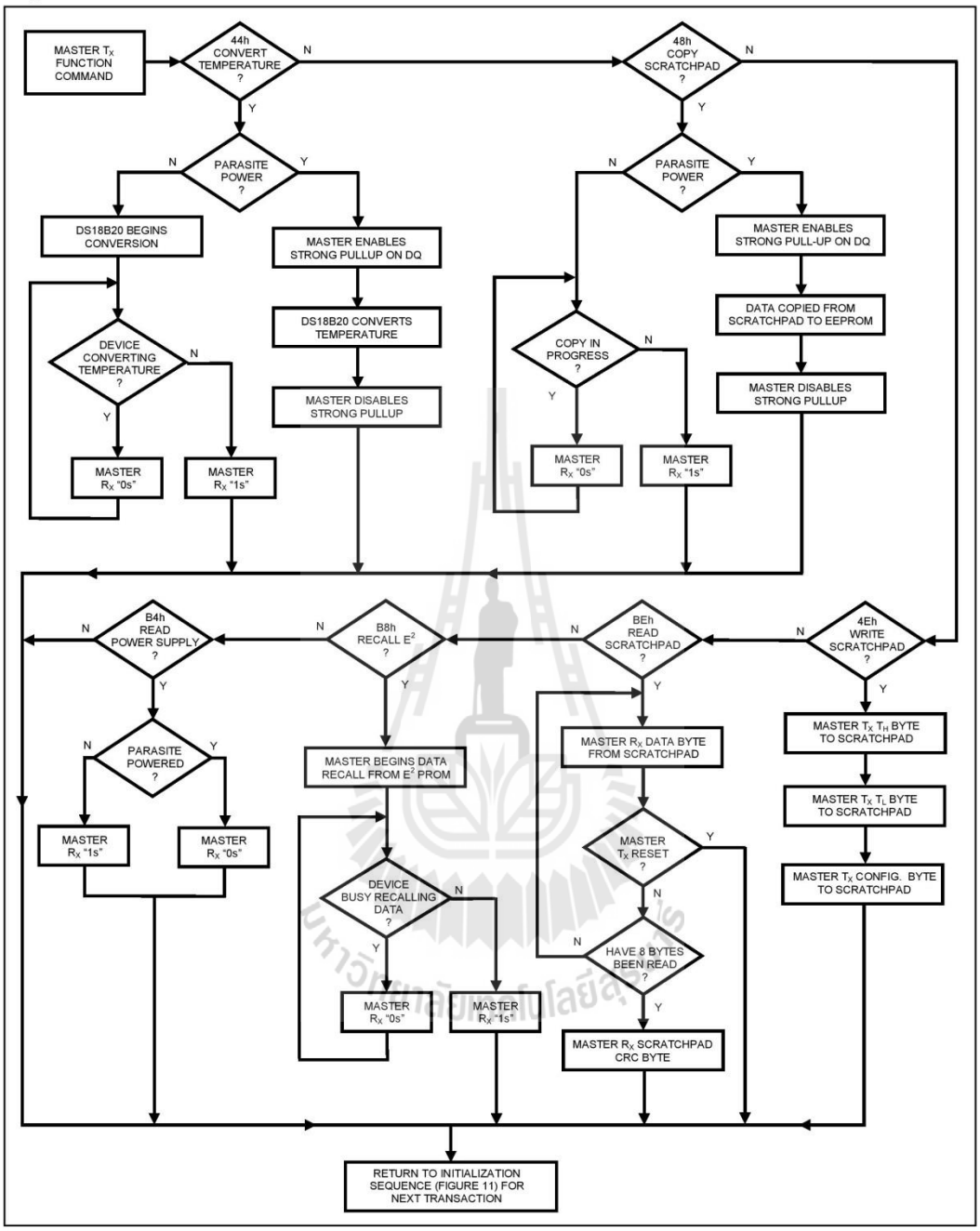


Figure 12. DS18B20 Function Commands Flowchart



1-WIRE SIGNALING

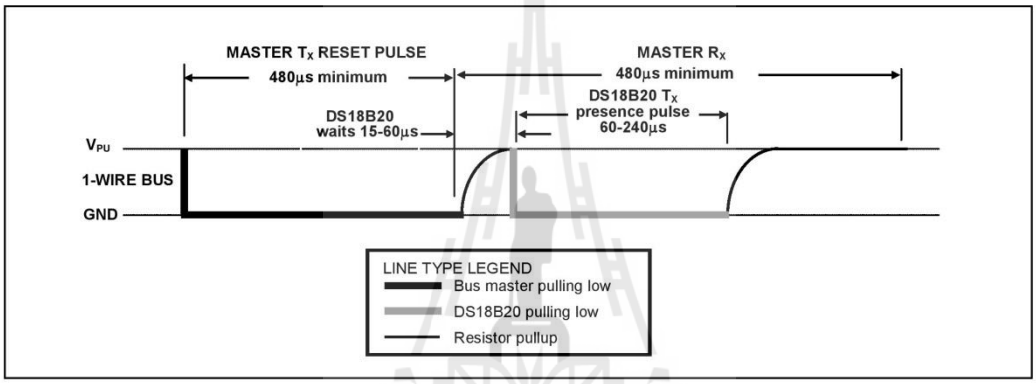
The DS18B20 uses a strict 1-Wire communication protocol to ensure data integrity. Several signal types are defined by this protocol: reset pulse, presence pulse, write 0, write 1, read 0, and read 1. The bus master initiates all these signals, with the exception of the presence pulse.

INITIALIZATION PROCEDURE—RESET AND PRESENCE PULSES

All communication with the DS18B20 begins with an initialization sequence that consists of a reset pulse from the master followed by a presence pulse from the DS18B20. This is illustrated in Figure 13. When the DS18B20 sends the presence pulse in response to the reset, it is indicating to the master that it is on the bus and ready to operate.

During the initialization sequence the bus master transmits (T_X) the reset pulse by pulling the 1-Wire bus low for a minimum of 480µs. The bus master then releases the bus and goes into receive mode (R_X). When the bus is released, the 5kΩ pullup resistor pulls the 1-Wire bus high. When the DS18B20 detects this rising edge, it waits 15µs to 60µs and then transmits a presence pulse by pulling the 1-Wire bus low for 60µs to 240µs.

Figure 13. Initialization Timing



READ/WRITE TIME SLOTS

The bus master writes data to the DS18B20 during write time slots and reads data from the DS18B20 during read time slots. One bit of data is transmitted over the 1-Wire bus per time slot.

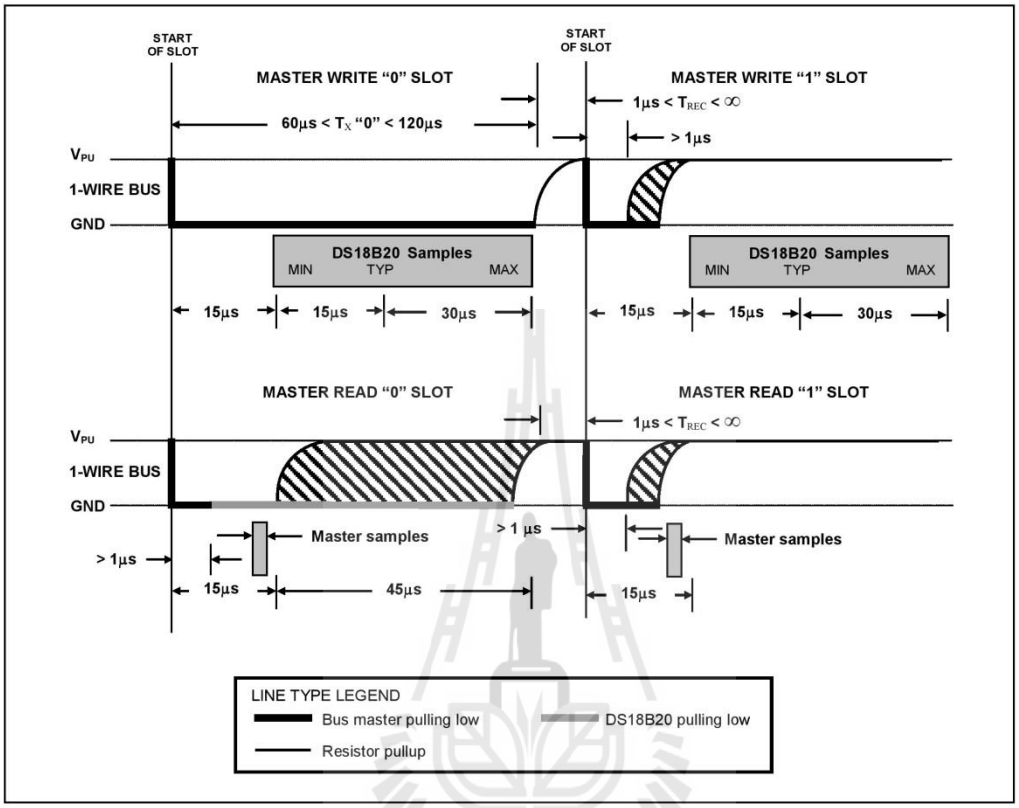
WRITE TIME SLOTS

There are two types of write time slots: "Write 1" time slots and "Write 0" time slots. The bus master uses a Write 1 time slot to write a logic 1 to the DS18B20 and a Write 0 time slot to write a logic 0 to the DS18B20. All write time slots must be a minimum of 60µs in duration with a minimum of a 1µs recovery time between individual write slots. Both types of write time slots are initiated by the master pulling the 1-Wire bus low (see Figure 14).

To generate a Write 1 time slot, after pulling the 1-Wire bus low, the bus master must release the 1-Wire bus within 15µs. When the bus is released, the 5kΩ pullup resistor will pull the bus high. To generate a Write 0 time slot, after pulling the 1-Wire bus low, the bus master must continue to hold the bus low for the duration of the time slot (at least 60µs).

The DS18B20 samples the 1-Wire bus during a window that lasts from 15µs to 60µs after the master initiates the write time slot. If the bus is high during the sampling window, a 1 is written to the DS18B20. If the line is low, a 0 is written to the DS18B20.

Figure 14. Read/Write Time Slot Timing Diagram



READ TIME SLOTS

The DS18B20 can only transmit data to the master when the master issues read time slots. Therefore, the master must generate read time slots immediately after issuing a Read Scratchpad [BEh] or Read Power Supply [B4h] command, so that the DS18B20 can provide the requested data. In addition, the master can generate read time slots after issuing Convert T [44h] or Recall E^2 [B8h] commands to find out the status of the operation as explained in the *DS18B20 Function Commands* section.

All read time slots must be a minimum of 60µs in duration with a minimum of a 1µs recovery time between slots. A read time slot is initiated by the master device pulling the 1-Wire bus low for a minimum of 1µs and then releasing the bus (see Figure 14). After the master initiates the read time slot, the DS18B20 will begin transmitting a 1 or 0 on bus. The DS18B20 transmits a 1 by leaving the bus high and transmits a 0 by pulling the bus low. When transmitting a 0, the DS18B20 will release the bus by the end of the time slot, and the bus will be pulled back to its high idle state by the pullup resister. Output

data from the DS18B20 is valid for 15µs after the falling edge that initiated the read time slot. Therefore, the master must release the bus and then sample the bus state within 15µs from the start of the slot.

Figure 15 illustrates that the sum of T_{INIT}, T_{RC}, and T_{SAMPLE} must be less than 15µs for a read time slot. Figure 16 shows that system timing margin is maximized by keeping T_{INIT} and T_{RC} as short as possible and by locating the master sample time during read time slots towards the end of the 15µs period.

Figure 15. Detailed Master Read 1 Timing

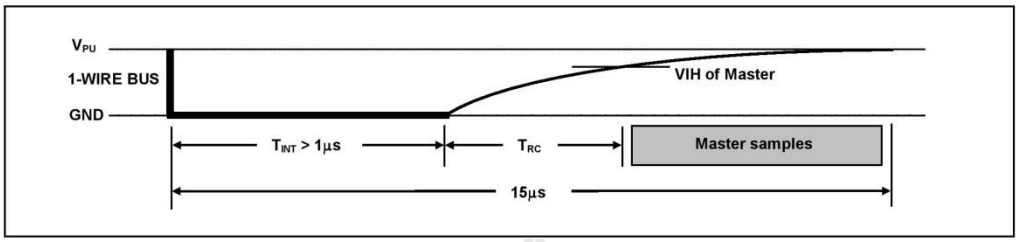
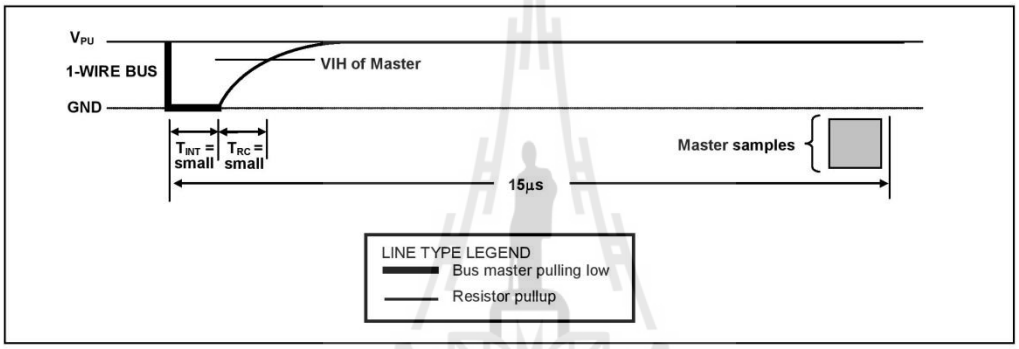


Figure 16. Recommended Master Read 1 Timing



RELATED APPLICATION NOTES

The following application notes can be applied to the DS18B20 and are available on our website at www.maxim-ic.com.

Application Note 27: Understanding and Using Cyclic Redundancy Checks with Maxim iButton Products
Application Note 122: Using Dallas' 1-Wire ICs in 1-Cell Li-Ion Battery Packs with Low-Side N-Channel Safety FETs Master
Application Note 126: 1-Wire Communication Through Software
Application Note 162: Interfacing the DS18x20/DS1822 1-Wire Temperature Sensor in a Microcontroller Environment
Application Note 208: Curve Fitting the Error of a Bandgap-Based Digital Temperature Sensor
Application Note 2420: 1-Wire Communication with a Microchip PICmicro Microcontroller
Application Note 3754: Single-Wire Serial Bus Carries Isolated Power and Data

Sample 1-Wire subroutines that can be used in conjunction with *Application Note 74: Reading and Writing iButtons via Serial Interfaces* can be downloaded from the Maxim website.

DS18B20 OPERATION EXAMPLE 1

In this example there are multiple DS18B20s on the bus and they are using parasite power. The bus master initiates a temperature conversion in a specific DS18B20 and then reads its scratchpad and recalculates the CRC to verify the data.

MASTER MODE	DATA (LSB FIRST)	COMMENTS
Tx	Reset	Master issues reset pulse.
Rx	Presence	DS18B20s respond with presence pulse.
Tx	55h	Master issues Match ROM command.
Tx	64-bit ROM code	Master sends DS18B20 ROM code.
Tx	44h	Master issues Convert T command.
Tx	DQ line held high by strong pullup	Master applies strong pullup to DQ for the duration of the conversion (t_{CONV}).
Tx	Reset	Master issues reset pulse.
Rx	Presence	DS18B20s respond with presence pulse.
Tx	55h	Master issues Match ROM command.
Tx	64-bit ROM code	Master sends DS18B20 ROM code.
Tx	BEh	Master issues Read Scratchpad command.
Rx	9 data bytes	Master reads entire scratchpad including CRC. The master then recalculates the CRC of the first eight data bytes from the scratchpad and compares the calculated CRC with the read CRC (byte 9). If they match, the master continues; if not, the read operation is repeated.

DS18B20 OPERATION EXAMPLE 2

In this example there is only one DS18B20 on the bus and it is using parasite power. The master writes to the T_H, T_L, and configuration registers in the DS18B20 scratchpad and then reads the scratchpad and recalculates the CRC to verify the data. The master then copies the scratchpad contents to EEPROM.

MASTER MODE	DATA (LSB FIRST)	COMMENTS
Tx	Reset	Master issues reset pulse.
Rx	Presence	DS18B20 responds with presence pulse.
Tx	CCh	Master issues Skip ROM command.
Tx	4Eh	Master issues Write Scratchpad command.
Tx	3 data bytes	Master sends three data bytes to scratchpad (T_H, T_L, and config).
Tx	Reset	Master issues reset pulse.
Rx	Presence	DS18B20 responds with presence pulse.
Tx	CCh	Master issues Skip ROM command.
Tx	BEh	Master issues Read Scratchpad command.
Rx	9 data bytes	Master reads entire scratchpad including CRC. The master then recalculates the CRC of the first eight data bytes from the scratchpad and compares the calculated CRC with the read CRC (byte 9). If they match, the master continues; if not, the read operation is repeated.
Tx	Reset	Master issues reset pulse.
Rx	Presence	DS18B20 responds with presence pulse.
Tx	CCh	Master issues Skip ROM command.
Tx	48h	Master issues Copy Scratchpad command.
Tx	DQ line held high by strong pullup	Master applies strong pullup to DQ for at least 10ms while copy operation is in progress.

ABSOLUTE MAXIMUM RATINGS

Voltage Range on Any Pin Relative to Ground .....-0.5V to +6.0V
Operating Temperature Range .....-55°C to +125°C
Storage Temperature Range.....-55°C to +125°C
Solder Temperature .....Refer to the IPC/JEDEC J-STD-020 Specification.

These are stress ratings only and functional operation of the device at these or any other conditions above those indicated in the operation sections of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods of time may affect reliability.

DC ELECTRICAL CHARACTERISTICS (-55°C to +125°C; V_{DD}=3.0V to 5.5V)

PARAMETER	SYMBOL	CONDITIONS	MIN	TYP	MAX	UNITS	NOTES
Supply Voltage	V_{DD}	Local Power	+3.0		+5.5	V	1
Pullup Supply Voltage	V_{PU}	Parasite Power	+3.0		+5.5	V	1,2
		Local Power	+3.0		V_{DD}		
Thermometer Error	t_{ERR}	-10°C to +85°C			±0.5	°C	3
		-55°C to +125°C			±2		
Input Logic-Low	V_{IL}		-0.3		+0.8	V	1,4,5
Input Logic-High	V_{IH}	Local Power	+2.2		The lower of 5.5 or V_{DD} + 0.3	V	1, 6
		Parasite Power	+3.0				
Sink Current	I_L	$V_{I/O}$ = 0.4V	4.0			mA	1
Standby Current	I_{DDS}			750	1000	nA	7,8
Active Current	I_{DD}	V_{DD} = 5V		1	1.5	mA	9
DQ Input Current	I_{DQ}			5		µA	10
Drift				±0.2		°C	11

NOTES:

1) All voltages are referenced to ground.
2) The Pullup Supply Voltage specification assumes that the pullup device is ideal, and therefore the high level of the pullup is equal to V_{PU}. In order to meet the V_{IH} spec of the DS18B20, the actual supply rail for the strong pullup transistor must include margin for the voltage drop across the transistor when it is turned on; thus: $V_{PU\_ACTUAL} = V_{PU\_IDEAL} + V_{TRANSISTOR}$.
3) See typical performance curve in Figure 17.
4) Logic-low voltages are specified at a sink current of 4mA.
5) To guarantee a presence pulse under low voltage parasite power conditions, V_{ILMAX} may have to be reduced to as low as 0.5V.
6) Logic-high voltages are specified at a source current of 1mA.
7) Standby current specified up to +70°C. Standby current typically is 3µA at +125°C.
8) To minimize I_{DDS}, DQ should be within the following ranges: GND ≤ DQ ≤ GND + 0.3V or V_{DD} − 0.3V ≤ DQ ≤ V_{DD}.
9) Active current refers to supply current during active temperature conversions or EEPROM writes.
10) DQ line is high ("high-Z" state).
11) Drift data is based on a 1000-hour stress test at +125°C with V_{DD} = 5.5V.

AC ELECTRICAL CHARACTERISTICS—NV MEMORY

(-55°C to +100°C; V_{DD} = 3.0V to 5.5V)

PARAMETER	SYMBOL	CONDITIONS	MIN	TYP	MAX	UNITS
NV Write Cycle Time	t_{WR}			2	10	ms
EEPROM Writes	N_{EEWR}	-55°C to +55°C	50k			writes
EEPROM Data Retention	t_{EEDR}	-55°C to +55°C	10			years

AC ELECTRICAL CHARACTERISTICS (-55°C to +125°C; V_{DD} = 3.0V to 5.5V)

PARAMETER	SYMBOL	CONDITIONS	MIN	TYP	MAX	UNITS	NOTES
Temperature Conversion Time	t_{CONV}	9-bit resolution			93.75	ms	1
		10-bit resolution			187.5		
		11-bit resolution			375		
		12-bit resolution			750		
Time to Strong Pullup On	t_{SPON}	Start Convert T Command Issued			10	µs	
Time Slot	t_{SLOT}		60		120	µs	1
Recovery Time	t_{REC}		1			µs	1
Write 0 Low Time	t_{LOW0}		60		120	µs	1
Write 1 Low Time	t_{LOW1}		1		15	µs	1
Read Data Valid	t_{RDV}				15	µs	1
Reset Time High	t_{RSTH}		480			µs	1
Reset Time Low	t_{RSTL}		480			µs	1,2
Presence-Detect High	t_{PDHIGH}		15		60	µs	1
Presence-Detect Low	t_{PDLOW}		60		240	µs	1
Capacitance	$C_{IN/OUT}$				25	pF	

NOTES:

1) See the timing diagrams in Figure 18.
2) Under parasite power, if t_{RSTL} > 960µs, a power-on reset may occur.

Figure 17. Typical Performance Curve

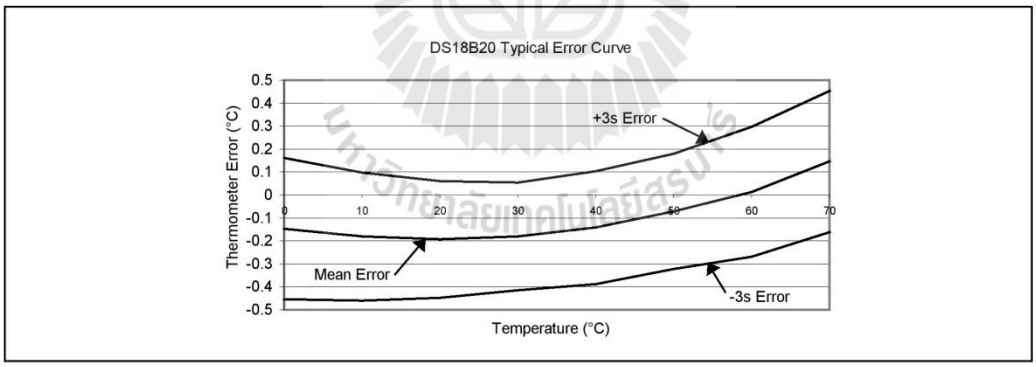
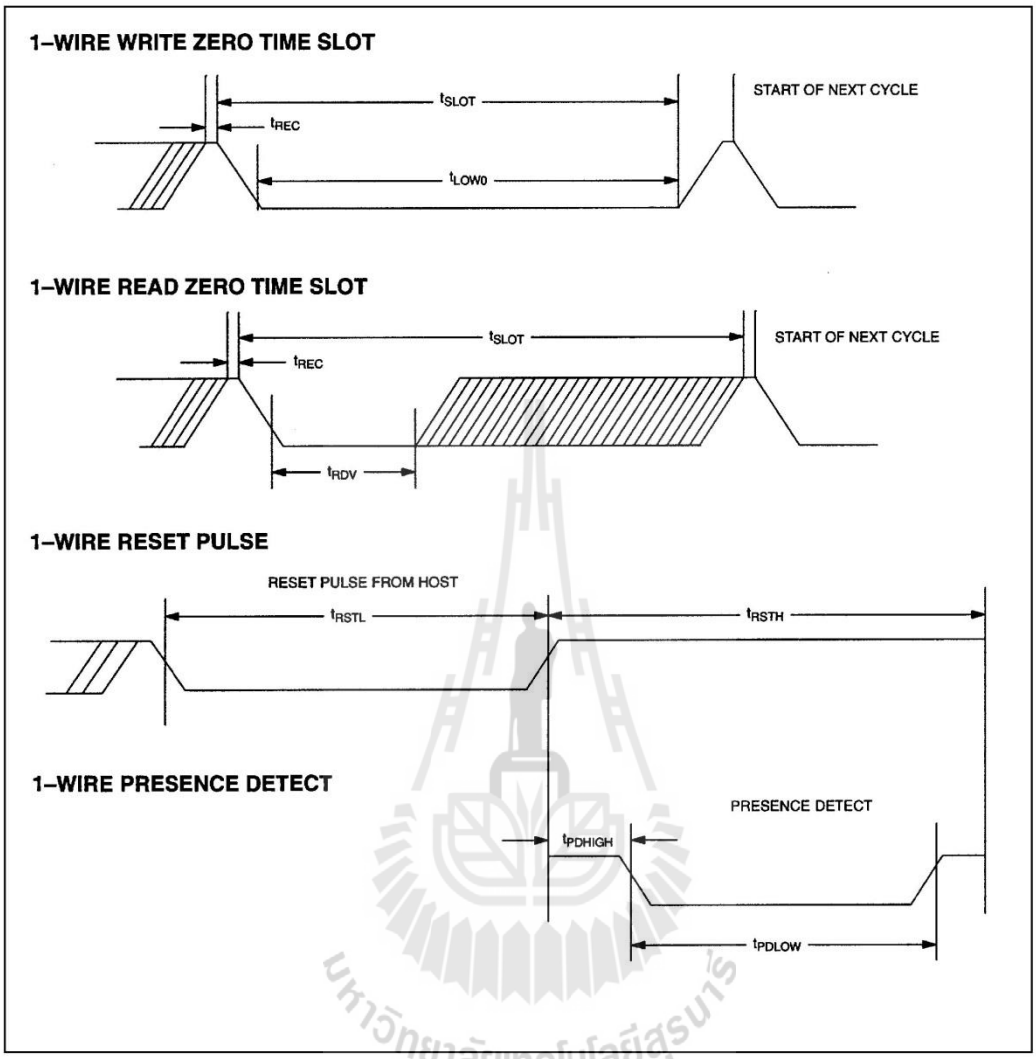


Figure 18. Timing Diagrams



REVISION HISTORY

REVISION DATE	DESCRIPTION	PAGES CHANGED
030107	In the *Absolute Maximum Ratings* section, removed the reflow oven temperature value of +220°C. Reference to JEDEC specification for reflow remains.	19
101207	In the *Operation—Alarm Signaling* section, added "or equal to" in the desciption for a TH alarm condition	5
	In the *Memory* section, removed incorrect text describing memory.	7
	In the *Configuration Register* section, removed incorrect text describing configuration register.	8
042208	In the *Ordering Information* table, added TO-92 straight-lead packages and included a note that the TO-92 package in tape and reel can be ordered with either formed or straight leads.	2