เครื่องรดน้ำต้นไม้อัตโนมัติ

โดย

นางสาวสุปรียา	มะโนมั่น	รหัสนักศึกษา B5003686
นายไพสิฐ	พูลเพิ่ม	รหัสนักศึกษา B5017171

รายงานนี้เป็นส่วนหนึ่งของการศึกษาวิชา 427499 โครงงานวิศวกรรมโทรคมนาคม

หลักสูตรวิศวกรรมศาสตรบัณฑิต สาขาวิชาวิศวกรรมโทรคมนาคม หลักสูตรปรับปรุง พ.ศ. 2546

สำนักวิชาวิศวกรรมศาสตร์ มหาวิทยาลัยเทคโนโลยีสุรนารี

ประจำภาคการศึกษาที่ 3 ปีการศึกษา 2553

เครื่องรดน้ำต้นไม้อัตโนมัติ

คณะกรรมการสอบโครงงาน

(ผู้ช่วยศาสตราจารย์ เรืออากาศเอก ดร. ประโยชน์ คำสวัสดิ์)
กรรมการ/อาจารย์ที่ปรึกษาโครงงาน

(ผู้ช่วยศาสตราจารย์ ดร. รังสรรค์ ทองทา)
กรรมการ

(ผู้ช่วยศาสตราจารย์ ดร. ชาญชัย ทองโสภา)
กรรมการ

มหาวิทยาลัยเทคโนโลยีสุรนารี อนุมัติให้นับรายงานโครงงานฉบับนี้ เป็นส่วนหนึ่งของการศึกษาระดับปริญญาตรี สาขาวิชาวิศวกรรมโทรคมนาคม วิชา 427499 โครงงานวิศวกรรมโทรคมนาคม ประจำปีการศึกษา 2553

โครงงาน	เครื่องรดน้ำต้นไม้อัตโนมัติ		
ผู้ดำเนินงาน	1.นางสาวสุปรียา	มะโนมั่น	รหัสประจำตัว B5003686
	2.นายไพสิฐ	พูลเพิ่ม	รหัสประจำตัว B5017171
อาจารย์ที่ปรึกษา	ผศ.ร.อ. ดร.ประโยชน์ คำสวัสดิ์		
สาขาวิชา	วิศวกรรมโทรคมนาคม		
ภาคการศึกษา	3/2553		

บทคัดย่อ

(Abstract)

ในปัจจุบันนี้โลกกำลังเผชิญกับปัญหาภาวะโลกร้อน อันเนื่องมาจากทรัพยากรป่าไม้กำลังจะหมดไป ดังนั้นทางภาครัฐจึงได้มีการรณรงค์ให้มีการปลูกต้นไม้ และเพื่ออำนวยความสะดวกหลักการทำงานของเครื่องรดน้ำต้นไม้อัตโนมัติ คือ จะรับค่าความชื้นและอุณหภูมิผ่านตัวเซนเซอร์เข้ามาประมวลผลโดยตัวไมโครคอนโทรลเลอร์ เพื่อตัดสินใจว่าจะทำการรดน้ำต้นไม้หรือไม่ และยังสามารถทำงานได้ในโหมดของการตั้งเวลา เพื่อให้ทำการรดน้ำต้นไม้ตามเวลาที่ตั้งไว้ โดยที่การเปิด – ปิดน้ำ จะควบคุมผ่านโซลินอยด์วาล์ว จากการทดสอบการทำงานพบว่า เครื่องรดน้ำต้นไม้อัตโนมัติสามารถทำงานได้จริงตามที่ออกแบบไว้ทุกประการ

กิตติกรรมประกาศ

(Acknowledgement)

การจัดทำโครงงานเรื่อง เครื่องรดน้ำต้นไม้อัตโนมัตินี้ได้ประสบความสำเร็จด้วยดี เนื่องจากได้รับความอนุเคราะห์ในการให้คำปรึกษาในด้านต่างๆ ในระหว่างการดำเนินการจากบุคคลหลายท่านที่ได้ให้ความช่วยเหลือและให้คำปรึกษา รวมทั้งข้อเสนอแนะที่เป็นประโยชน์ในการทำโครงงานครั้งนี้ ซึ่งบุคคลเหล่านี้ประกอบด้วย

ผศ.ร.อ. ดร.ประโยชน์ คำสวัสดิ์ (อาจารย์ที่ปรึกษาที่ปรึกษาโครงงาน)

นายไพโรจน์ บุญไทย (นักศึกษาปริญญาโทสาขาวิชาแมคคาทรอนิกส์)

นายณัศรุณ บินโหรน (นักศึกษาปริญญาโทสาขาวิชาเทคโนโลยีสารสนเทศ)

ข้าพเจ้าใคร่ขอขอบพระคุณผู้ที่มีส่วนเกี่ยวข้องทุกท่านที่มีส่วนร่วมในการให้ข้อมูลและเป็นที่ปรึกษาในการทำรายงานฉบับนี้จนเสร็จสมบูรณ์ ตลอดจนให้การดูแลและให้ความเข้าใจเกี่ยวกับพื้นฐานการใช้งานโปรแกรม ซึ่งข้าพเจ้าขอขอบพระคุณเป็นอย่างสูงไว้ ณ ที่นี้

นางสาวสุปรียา มะโนมั่น

นายไพสิฐ พูลเพิ่ม

คณะผู้จัดทำ

สารบัญ

สารบัญ (ต่อ)

สารบัญ (ต่อ)

สารบัญรูป

สารบัญรูป (ต่อ)

สารบัญรูป (ต่อ)

สารบัญรูป (ต่อ)

สารบัญตาราง

บทที่ 1
บทนำ

1.1 ความเป็นมา

ในปัจจุบันนี้โลกกำลังเผชิญกับปัญหาภาวะโลกร้อน อันเนื่องมาจากทรัพยากรป่าไม้กำลังจะหมดไป ดังนั้นทางภาครัฐจึงได้มีการรณรงค์ให้มีการปลูกต้นไม้ ซึ่งมนุษย์ในโลกปัจจุบันต้องทำงานจึงไม่มีเวลาในการรดน้ำต้นไม้จึงทำให้เกิดความคิดที่จะสร้างเครื่องรดน้ำต้นไม้อัตโนมัติขึ้นมาเพื่ออำนวยความสะดวกแก่บุคคลที่รักษ์โลก และเพื่อตอบสนองความต้องการของภาครัฐที่ต้องการให้ทุกคนช่วยกันรักษาต้นไม้เพื่อลดภาวะโลกร้อน ดังนั้นโครงงานเครื่องรดน้ำอัตโนมัติจึงเหมาะกับโลกในยุคปัจจุบัน

1.2 วัตถุประสงค์

1. เพื่อตอบสนองความต้องการของส่วนบุคคลที่ต้องการความสะดวกในชีวิตประจำวัน
2. เพื่อตอบสนองความต้องการของภาครัฐ
3. เพื่อให้บุคคลทั่วไปหันมาสนใจการปลูกต้นไม้
4. เพื่อลดภาวะโลกร้อน
5. เพื่อใช้ในการเกษตร

1.3 ขอบเขตงาน

1. ศึกษาการใช้โปรแกรม WinAvR
2. ศึกษาการใช้ไมโครคอนโทรลเลอร์
3. ออกแบบวงจรต่าง ๆ
4. สร้างบอร์ดวงจรที่ควบคุมจากไมโครคอนโทรลเลอร์
5. ทดลองการใช้งานและแก้ไขสิ่งที่ผิดพลาด

1.4 ขั้นตอนการดำเนินงาน

1. ศึกษาหัวเรื่องที่ต้องการศึกษานำมาเปรียบเทียบ และทำการเลือกหัวข้อที่ต้องการศึกษาเรื่องเครื่องรดน้ำต้นไม้อัตโนมัติ
2. ศึกษาข้อมูลศึกษาข้อมูลนำมาประกอบกับความรู้ทางทฤษฎีที่เกี่ยวข้องกับหัวข้อ หาข้อมูลเพิ่มเติมเกี่ยวกับหัวข้อที่ต้องการศึกษา
3. ทำการจัดซื้ออุปกรณ์อิเล็กทรอนิกส์ที่ใช้ในการศึกษา และศึกษาอุปกรณ์อิเล็กทรอนิกส์
4. ศึกษาภาษาซีเพื่อเขียนโปรแกรมควบคุมการทำงาน
5. ตรวจสอบโปรแกรมและอุปกรณ์อิเล็กทรอนิกส์ ตลอดจนการเตรียมเอกสาร จัดทำเอกสาร และนำเสนอโครงงาน

1.5 ประโยชน์ที่คาดว่าจะได้รับ

1. ทำให้บุคคลทั่วไปหันมาสนใจการปลูกต้นไม้
2. สามารถลดภาวะโลกร้อนได้
3. สามารถนำไปใช้ในการเกษตรได้
4. สามารถทำงานเป็นทีมได้
5. สามารถวิเคราะห์งานและแก้ปัญหาอย่างเป็นระบบได้

บทที่ 2

ทฤษฎีและหลักการที่เกี่ยวข้อง

2.1 บทนำ

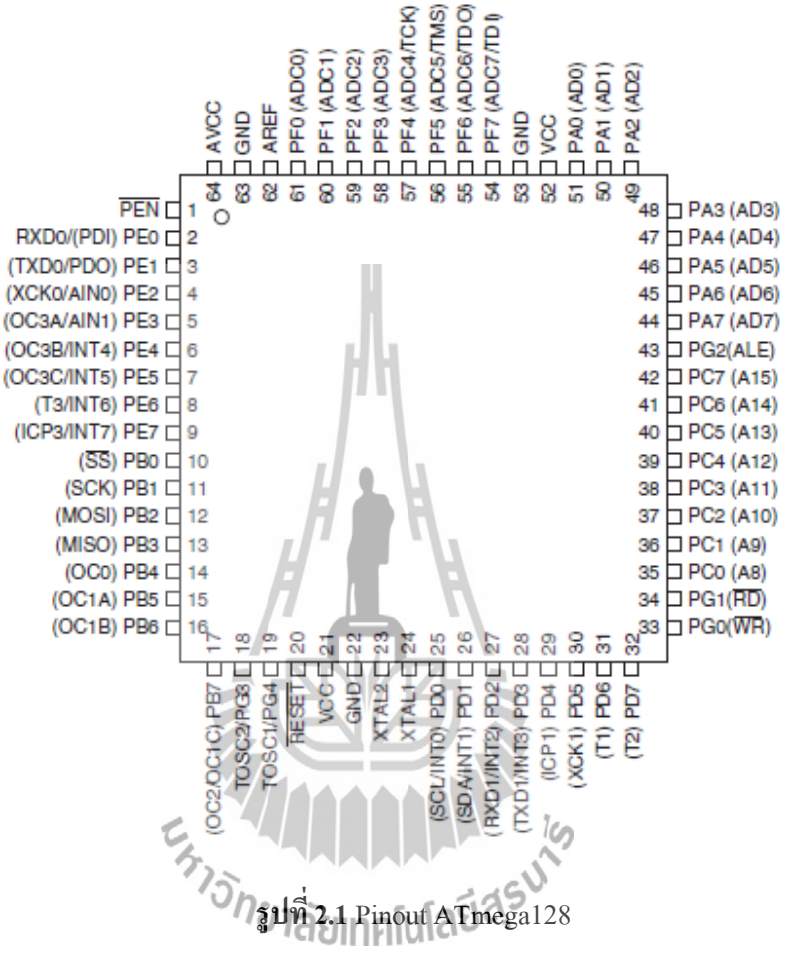
การทดสอบการทำงานของเครื่องรดน้ำต้นไม้อัตโนมัติ ได้ใช้ทฤษฎีและหลักการที่เกี่ยวข้อง คือ ไมโครคอนโทรลเลอร์ เซนเซอร์วัดความชื้นและอุณหภูมิ และโซลินอยด์ ซึ่งมีหลักการทำงานดังต่อไปนี้

2.2 หลักการทำงานของไมโครคอนโทรลเลอร์

ไมโครคอนโทรลเลอร์ (Microcontroller) มาจากคำ 2 คำ คำหนึ่งคือ ไมโคร (Micro) หมายถึงขนาดเล็ก และคำว่า คอนโทรลเลอร์ (controller) หมายถึงตัวควบคุมหรืออุปกรณ์ควบคุม ดังนั้นไมโครคอนโทรลเลอร์ จึงหมายถึงอุปกรณ์ควบคุมขนาดเล็ก แต่ในตัวอุปกรณ์ควบคุมขนาดเล็กนี้ ได้บรรจุความสามารถที่คล้ายคลึงกับระบบคอมพิวเตอร์ ที่คนโดยส่วนใหญ่คุ้นเคย กล่าวคือภายในไมโครคอนโทรลเลอร์ได้รวมเอาซีพียู หน่วยความจำ และพอร์ต ซึ่งเป็นส่วนประกอบหลักสำคัญของระบบคอมพิวเตอร์เข้าไว้ด้วยกัน โดยทำการบรรจุเข้าไว้ในตัวถังเดียวกัน

ในโครงงานนี้เป็นบอร์ดไมโครคอนโทรลเลอร์ในตระกูล AVR ของบริษัท Atmel ซึ่งบอร์ดนี้เลือกใช้ MCU เบอร์ ATmega128 ขนาด 64 Pin โดยการออกแบบโครงสร้างของบอร์ดนั้นจะเน้นเรื่องการจัดวางบอร์ดให้มีขนาดเล็กเพื่อให้ง่ายต่อการนำไปประยุกต์ใช้งาน โดยได้นำ MCU มาจัดวงจรร่วมกับอุปกรณ์พื้นฐานที่จำเป็น และจัดขาออกมาใช้งานภายนอก ซึ่งการจัดเรียงขาสัญญาณจะทำการจัดเรียงอย่างเป็นระเบียบเพื่อให้สามารถต่อใช้งานได้โดยสะดวก โดยที่ตัวบอร์ดจะใช้ไฟเลี้ยง +5V

2.3 คุณสมบัติของ MCU เบอร์ATmega128



รูปที่ 2.1 Pinout ATmega128

1. ความเร็วสัญญาณนาฬิกา Crystal 16 MHz
2. รองรับการโปรแกรมแบบ SPI และ JTAG (ต้องใช้ร่วมกับบอร์ด ET.AVR START KIT V1.0)
3. Power supply ใช้แรงดันไฟฟ้า 4.5 V - 5.5 V
4. ภายใน MCU มีหน่วยความจำโปรแกรมแบบ Flash ขนาด 64 KB หน่วยความจำข้อมูล RAM ขนาด 4 KB หน่วยความจำข้อมูลถาวรแบบ EEPROM ขนาด 2 KB สามารถลบและเขียนซ้ำได้กว่า 100,000 ครั้ง

5. จำนวน I/O สูงสุดถึง 53 I/O Pins ซึ่งขาสัญญาณ I/O จะมีการใช้งานร่วมกันของ Function อื่น ๆ อีกดังนี้

   5.1 SPI จำนวน 1 ช่อง , I2C จำนวน 1 ช่อง , 10.Bit ADC จำนวน 8 ช่อง

   5.2 Programmable Serial USARTs จำนวน 2 ช่อง

   5.3 Timers/Counters 8.Bit จำนวน 2 ช่อง , Timers/Counters 16.Bit จำนวน 2 ช่อง , 8.Bit PWM 2 ช่อง , Watchdog timer , Real time counter

6. ทนอุณหภูมิใช้งานระหว่าง -40 °C ถึง +85°C (ถ้าใช้งานที่อุณหภูมิ 85°C จะสามารถใช้งานได้ถึง 20 ปี และถ้าใช้งานที่อุณหภูมิ 25°C จะสามารถใช้งานได้ถึง 100 ปี)

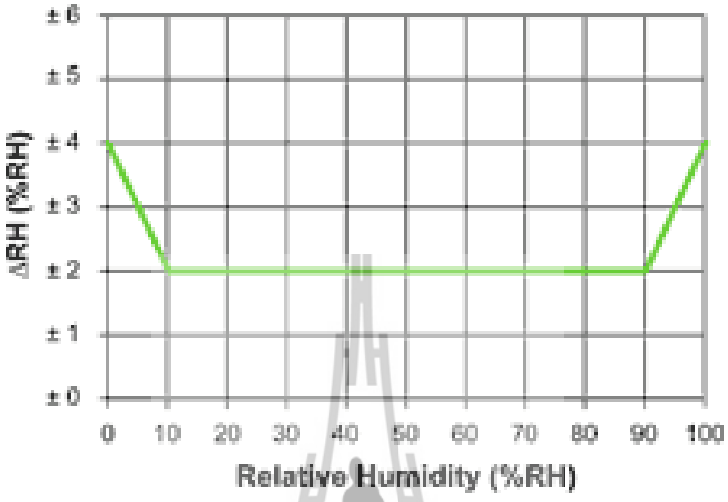
2.3 หลักการทำงานของเซนเซอร์วัดความชื้นและอุณหภูมิ

2.3.1 เซนเซอร์ SHT15

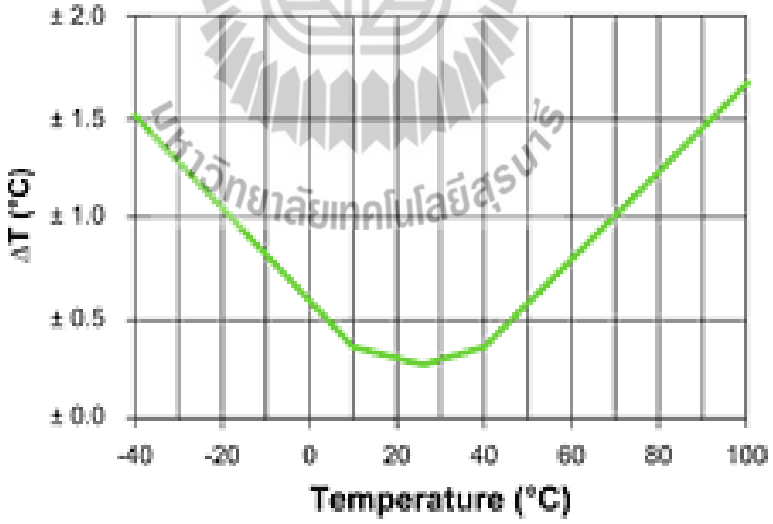
SHT15 เป็นอุณหภูมิดิจิตอลและเซนเซอร์ความชื้นให้ 14 และ 12 บิตของความแม่นยำในการวัดอุณหภูมิและความชื้นตามลำดับ

รูปที่ 2.2 เซนเซอร์ SHT15

Maximal accuracy limits for relative humdity and temperature:



รูปที่ 2.3 Relative Humidity (%RH)



รูปที่ 2.4 Temperture (°C)

2.3.2 คุณลักษณะเฉพาะของไอซี และบอร์ดวัด/ควบคุมอุณหภูมิ และความชื้นสัมพัทธ์

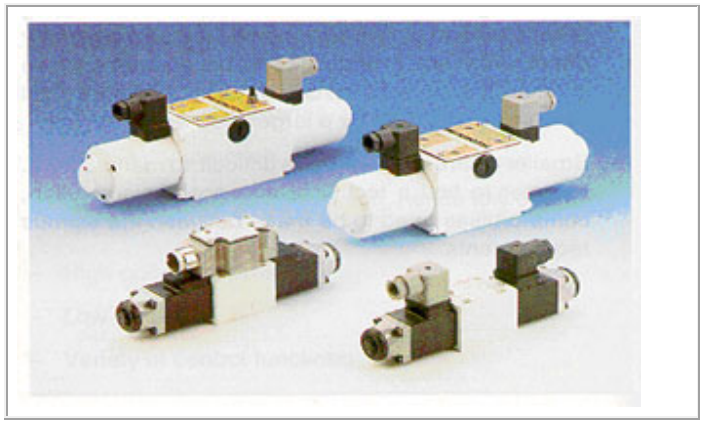
1. ตัวถังแบบ PDIP 28 ขา เป็น 8 bits AVR microcontroller เบอร์ ATmega8
2. ความต้องการไฟเลี้ยง 4.5 - 5.5V
3. เชื่อมต่อกับเซนเซอร์วัดอุณหภูมิ และความชื้นสัมพัทธ์ SHT15 จาก sensIrion ซึ่งมีคุณสมบัติเบื้องต้น ดังนี้
    - มี 2 เซนเซอร์ภายใน สำหรับวัดความชื้นสัมพัทธ์ และอุณหภูมิ
    - วัดความชื้นสัมพัทธ์ ได้ตั้งแต่ 0 - 100% RH โดยมี Absol. RH accuracy: +/- 2% RH (10...90% RH)
    - วัดอุณหภูมิ โดยมี Temp. accuracy: +/- 0.3?C @ 25?C
    - วัดอุณหภูมิได้ตั้งแต่ -40 ถึง 123.8 °C
4. มีรีเลย์จำนวน 4 ตัว แต่ละตัวสามารถตั้งให้ทำงานเพื่อควบคุมอุณหภูมิ หรือความชื้นสัมพัทธ์ได้แบบอิสระจากกัน
5. อัตราเร็วในการรับส่งข้อมูล 2,400 bps
6. ควบคุมการทำงานได้ทั้งจากปุ่มกดบนบอร์ด หรือควบคุมผ่านทางพอร์ตอนุกรมของเครื่องคอมพิวเตอร์ พร้อมจุดต่อ สำหรับการขยายระบบ
7. สามารถเชื่อมต่อบอร์ดตั้งแต่สอง หรือจำนวนมากกว่านั้น เข้าด้วยกัน เพื่อขยายจำนวนอินพุต/เอ้าต์พุต ให้ได้จำนวนตามความต้องการ โดยตั้งหมายเลขประจำตัวของบอร์ด (ID) ตั้งแต่หมายเลข 0 ถึงหมายเลข 255
8. มีแอลอีดีแสดงผลการทำงานของบอร์ดที่กำลัง active, แสดงการทำงานของรีเลย์แต่ละตัว และแสดงการส่งข้อมูล
9. มีสวิตซ์กด เพื่อควบคุมการทำงานของบอร์ด 4 ตัว
10. แสดงผลทางจอแอลซีดี 2 บรรทัด 16 ตัวอักษรต่อบรรทัด(ตัดออกได้ในกรณีที่ใช้การเชื่อมต่อ และแสดงผลกับคอมพิวเตอร์)

2.4 หลักการทำงานของโซลินอยด์

2.4.1 โซลินอยด์วาล์ว (Solenoid Valve)

รูปที่ 2.5 โซลินอยด์วาล์ว

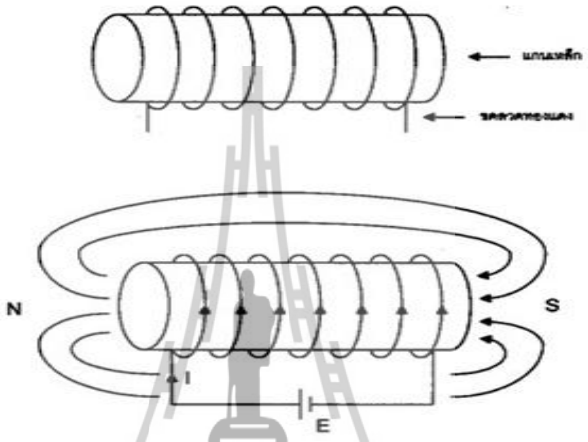
การควบคุมระบบไฮดรอลิกโดยใช้ไฟฟ้านั้น ต้องนำเมนวาล์ว (Main valve) ที่เป็นแบบที่ทำงานโดยใช้ไฟฟ้าควบคุม หรือที่เรียกว่า โซลินอยด์วาล์ว (Solenoid valve) มาใช้



รูปที่ 2.6 แสดงลักษณะของโซลินอยด์วาล์ว

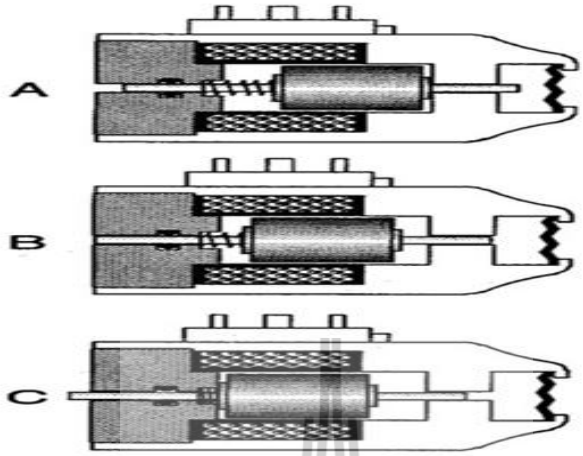
2.4.2 หลักการทำงานของโซลินอยด์วาล์ว

ขดลวดโซลินอยด์ก็คือ ขดลวดที่พันรอบแกนเหล็ก โดยมีวัตถุประสงค์ เพื่อสร้างสนามแม่เหล็ก เมื่อป้อนกระแสไฟฟ้าเข้าที่ขดลวดจะเกิดสนามแม่เหล็กขึ้นรอบขดลวด แล้วรวมตัวกันเป็นสนามแม่เหล็กที่ใหญ่ขึ้น โดยมีทิศทางวิ่งจากขั้ว N ไปขั้ว S



รูปที่ 2.7 ขดลวดแม่เหล็ก

ดังนั้น จากหลักการทำงานขดลวดโซลินอยด์ เมื่อป้อนกระแสไฟฟ้าเข้าที่ขดลวด จะทำให้เกิดสนามแม่เหล็กขึ้นรอบขดลวด ทำให้เกิด การดูดแกน Armature ซึ่งสามารถเอาชนะแรงสปริงที่ดัดแกน Armature ไว้ ทำให้แกน Armature เคลื่อนที่ไปอยู่ตรงกลางของขดลวด ดังแสดงในภาพที่ 2.8



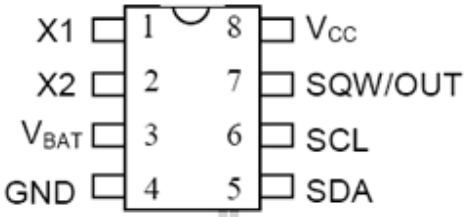
รูปที่ 2.8 สภาวะการทำงานของขดลวดโซลินอยด์
A สภาวะปกติ (ไม่ป้อนกระแสไฟฟ้าเข้าขดลวดโซลินอยด์)
B สภาวะทำงาน (ระหว่างป้อนกระแสไฟฟ้าเข้าขดลวอโซลินอยด์)
C สภาวะสุดท้าย (ป้อนกระแสไฟฟ้าเข้าขดลวดโซลินอยด์)

2.5 การใช้งาน RTC (Real Time Clock) ด้วย DS1307

ระบบฐานเวลา เป็นสิ่งสำคัญที่สามารถนำไปใช้ในอุปกรณ์อีเล็กโทรนิกส์ได้หลากหลาย ภายในไมโครคอนโทรเลอร์เองก็มีไทเมอร์เพื่อใช้ในการจับเวลา หรือนำไปใช้เป็นฐานเวลาจริงได้เช่นกัน แต่เนื่องจากไมโครคอนโทรเลอร์สามารถทำงานได้ต่อเมื่อมีไฟเลี้ยงเท่านั้น ดังนั้นการใช้ไทเมอร์ของไมโครคอนโทรลเลอร์ สร้างฐานเวลาจริงจึงไม่ เหมาะสมในบางแอปพลิเคชัน

DS1307 เป็น IC ฐานเวลาของดัลลัสเซมิคอนดักเตอร์ (Dallas semiconductor) มีบัสรับส่งข้อมูลแบบ I2C ซึ่งเป็นแบบ 2 wire สามารถสื่อสารได้ 2 ทิศทาง (bi.direction bus) ฐานเวลาของ DS1307 นั้นสามารถ เก็บข้อมูล วินาที, นาที, ชั่วโมง, วัน, วันที่, เดือน และปี ได้ ระบบเวลาสามารถทำงานโหมดรูปแบบ 24 ชั่วโมง หรือ 12 ชั่วโมง AM/PM ก็ได้ ภายมีระบบตรวจจับแหล่งจ่ายไฟ โดยถ้าแหล่งจ่ายไฟหลักถูกตัดไป DS1307 สามารถสวิตซ์ไปใช้ไฟจากแบตเตอรี่ และทำงานต่อไป โดยที่ยัง

สามารถรักษาข้อมูลไว้ได้ โครงสร้างมีขาทั้งหมด 8 ขาดังแสดงในรูปที่ 2.9 และมีรายละเอียดการทำงานของขาต่าง ๆ ดังนี้



รูปที่ 2.9 ตำแหน่งขาไอซี RTC DS1307

VCC: ใช้ต่อไฟเลี้ยง +5V

GND: ใช้ต่อกราวด์

VBAT: ใช้ต่อกับแบตเตอรี่ 3V เพื่อรักษาการทำงาน ในกรณีที่ไม่มีไฟเลี้ยงจ่าย

SDA: ขารับส่งข้อมูลด้วยระบบบัส I^2C

SCL: ขาสัญญาณนาฬิกาสำหรับการรับส่งข้อมูลด้วยระบบบัส I^2C

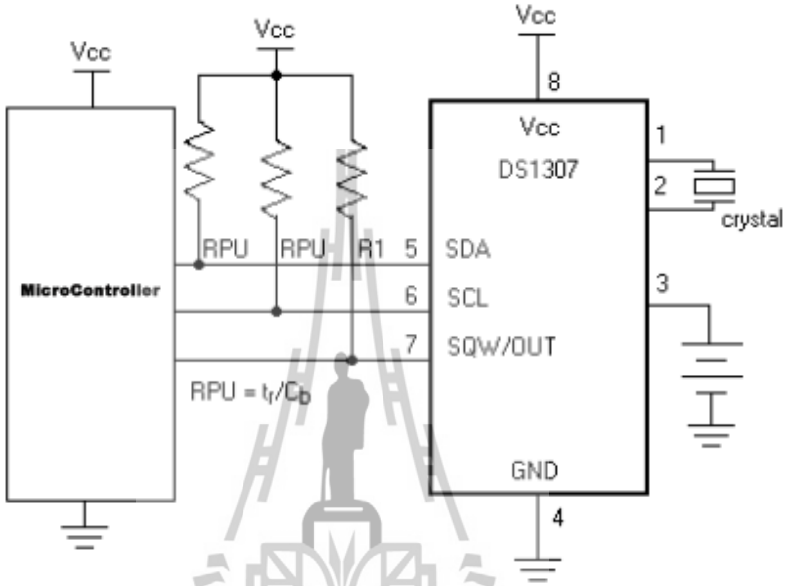
SQW/OUT: ขาเอาต์พุตสัญญาณ Square wave สามารถเลือกความถี่ได้

X1, X2: ใช้ต่อกับคริสตอลความถี่มาตรฐาน 32.768 kHz เพื่อสร้างฐานเวลาจริงให้กับ IC

ระบบบัสข้อมูลแบบ I^2C (Inter.IC Communication) ได้ถูกพัฒนาขึ้นโดยบริษัทฟิลิปส์ (Phillips) การรับส่งข้อมูลใช้สายสัญญาณเพียงแค่ 2 เส้น คือสายสัญญาณข้อมูล SDA (Serial Data line) และสายสัญญาณนาฬิกา SCL (Serial Clock line) มีการทำงานเป็นแบบ Master, Slave โดยอุปกรณ์ที่ทำหน้าที่เป็น Master (ไมโครคอนโทรลเลอร์) จะควบคุมการรับส่งข้อมูล และควบคุมสัญญาณนาฬิกาบน SCL ส่วนอุปกรณ์ Slave (DS1307) นั้นจะทำงานภายใต้การควบคุมของอุปกรณ์ Master

การต่อใช้งานร่วมกับไมโครคอนโทรลเลอร์ด้วยระบบบัส I^2C นั้นสามารถทำได้โดยต่อตัวต้านทาน Pull up ดังแสดงในรูปที่ 2.10 ในกรณีที่ต้องการต่อร่วมกับอุปกรณ์ Slave หลายตัว ก็สามารถทำได้โดยต่ออุปกรณ์ Slave ขนานกันไป การติดต่อสื่อสารระหว่างอุปกรณ์ Master กับ Slave แต่ละตัว

นั้น จะถูกแยกโดย Address ของอุปกรณ์ Slave ซึ่งจะถูกส่งจากอุปกรณ์ Master ไปยังอุปกรณ์ Slave ก่อนเริ่มการรับส่งข้อมูล



รูปที่ 2.10 การเชื่อมต่อ DS1307 เข้ากับไมโครคอนโทรลเลอร์ด้วยระบบบัสแบบ I^2C

2.5.1การรับส่งข้อมูลแบบ I^2C

การรับส่งข้อมูลแบบ I^2C นั้นมีข้อกำหนดอยู่ 2 ประการด้วยกันคือ

1. การรับส่งข้อมูลจะเริ่มขึ้นได้เมื่อบัสมีสถานะว่างเท่านั้น
2. ในช่วงที่ทำการรับส่งข้อมูลอยู่ สายสัญญาณ SDA ต้องไม่เปลี่ยนสถานะในช่วงที่ SCL มีสถานะเป็นลอจิก “1” ถ้า SDA มีการเปลี่ยนสถานะในช่วงที่ SCL เป็นลอจิก “1” จะถือว่าเป็นสัญญาณควบคุมการรับส่งข้อมูล

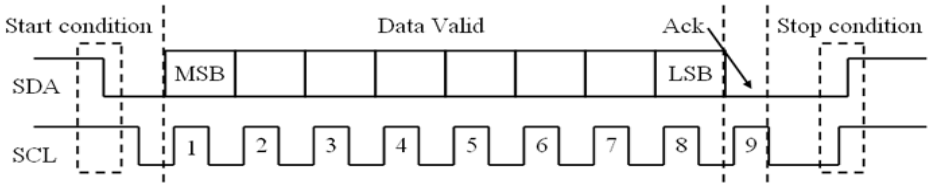
2.5.2 สถานะของการรับส่งข้อมูลแบบ I^2C

สถานะของการรับส่งข้อมูลแบบ I^2C สามารถแบ่งออกได้เป็น 5 สถานะด้วยกันดังแสดงในรูปที่ 2.11 และมีรายละเอียดดังนี้

1. สถานะว่าง (Bus not busy): สัญญาณ SDA และ SCL มีระดับสัญญาณเป็น High
2. เริ่มส่งข้อมูล (Start data transfer): มีการเปลี่ยนระดับสัญญาณของ SDA จาก High เป็น Low ในขณะที่ SCL มีระดับสัญญาณเป็น High ค้างไว้
3. หยุดส่งข้อมูล (Stop data transfer): มีการเปลี่ยนระดับสัญญาณของ SDA จาก Low เป็น High ในขณะที่ SCL มีระดับสัญญาณเป็น High ค้างไว้
4. รับส่งข้อมูล (Data valid): มีการรับส่งข้อมูลผ่านสายสัญญาณ SDA โดยข้อมูลแต่ละบิตจะถูกส่งในช่วงที่ SCL มีระดับเป็น High โดยในช่วงที่ SCL มีสถานะเป็น High อยู่นั้น SDA จะต้องไม่เกิดการเปลี่ยนระดับสัญญาณ

SDA จะเปลี่ยนระดับของสัญญาณ ในช่วงที่ SCL มีระดับสัญญาณเป็น Low เท่านั้น ตามมาตรฐานการส่งข้อมูล แบบ I^2C นี้สามารถส่งข้อมูลด้วยความถี่สัญญาณนาฬิกาสูงสุด 100 kHz ที่โหมดการทำงานธรรมดา และ 400 kHz ที่โหมดการทำงานแบบเร็ว แต่สำหรับ DS1307 สามารถทำงานได้ในโหมดธรรมดาเท่านั้น

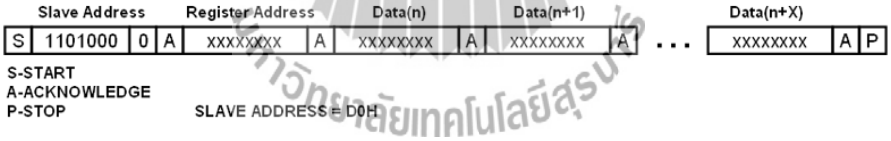
ตอบรับ (Acknowledge): เกิดขึ้นหลังจากที่มีการรับส่งข้อมูลครบแล้ว โดยอุปกรณ์ Master ต้องสร้างสัญญาณ Clock บน SCL เพิ่มอีกลูก อุปกรณ์ที่เป็นตัวรับข้อมูลจะดึงระดับสัญญาณบน SDA ให้เป็น Low เพื่อให้ตัวส่งรับรู้ว่าตัวรับได้รับข้อมูลครบแล้ว



รูปที่ 2.11 การรับส่งข้อมูลผ่านบัส I^2C

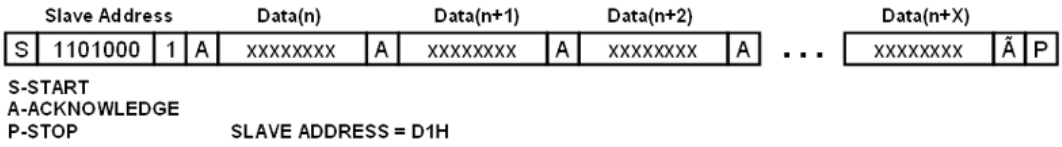
ในการรับส่งข้อมูลผ่านบัส I^2C อุปกรณ์ Master จะเป็นผู้สร้างสัญญาณ Clock บน SDA และเป็นตัวควบคุมสถานะ Start และ Stop เพื่อควบคุมการรับส่งข้อมูลทั้งหมด

การส่งข้อมูลไปยังอุปกรณ์ DS1307 ดังแสดงในรูปที่ 2.12 ไมโครคอนโทรลเลอร์ต้องสร้างสภาวะ Start ก่อน จากนั้นต้องส่ง Address ของ DS1307 ขนาด 7 บิตซึ่งมีค่าเป็น 1101000 และตามด้วยบิตระบุทิศทางของข้อมูล ในกรณีที่เป็นการเขียนข้อมูลลง DS1307 จะต้องเป็น “0” จากนั้นไมโครคอนโทรเลอร์จะต้องส่งตำแหน่ง Address ภายในรีจิสเตอร์ของ DS1307 ที่ต้องการเขียนข้อมูลลง แล้วจึงค่อยเขียนข้อมูลลง โดยในการส่งข้อมูลแต่ละไบต์จะต้องรอบิต Ack จาก DS1307 ทุกไบต์ เมื่อส่งจนครบแล้ว ถึงจะสร้างสภาวะ Stop เพื่อกลับสู่สถานะว่าง



รูปที่ 2.12 การเขียนข้อมูลอุปกรณ์ Slave ผ่านบัส I^2C

การรับข้อมูลจากอุปกรณ์ Slave ดังแสดงในรูปที่ 2.13 เริ่มแรกไมโครคอนโทรลเลอร์ต้องสร้างสภาวะ Start ก่อน จากนั้นต้องส่ง Address ของ DS1307 ขนาด 7 บิตซึ่งมีค่าเป็น 1101000 และตามด้วยบิตระบุทิศทางของข้อมูล ในกรณีที่เป็นการอ่านข้อมูลจาก DS1307 จะต้องเป็น “1” จากนั้นจึงค่อยรับข้อมูลจากอุปกรณ์ Slave ทีละไบต์ โดยตำแหน่งที่อ่านเข้ามาจะขึ้นอยู่กับตำแหน่งรีจิสเตอร์พอยท์เตอร์ ซึ่งจะเป็นตำแหน่งท้ายสุดที่ได้ทำการเขียนข้อมูลไว้ เมื่ออ่านข้อมูลครบแต่ละไบต์อุปกรณ์ Master ต้องส่ง Acknowledge บิตกลับไปให้อุปกรณ์ Slave ด้วย ในกรณีที่เป็นไบต์สุดท้าย อุปกรณ์ Master ต้องส่ง “not acknowledge” กลับไป



รูปที่ 2.13 การอ่านข้อมูลจากอุปกรณ์ Slave ผ่านบัส I^2C

ภายใน DS1307 มีรีจิสเตอร์ภายในใช้เก็บข้อมูลเวลาขนาด 7 ไบต์ 00H.06H ดังแสดงในรูปที่ 2.14 ข้อมูลค่าเวลา และวันที่จะถูกเก็บอยู่ในรูปของเลขฐาน 10 สามารถเลือกได้ว่าให้ทำงานแบบ 12 ชั่วโมง หรือ 24 ชั่วโมง โดยกำหนดที่บิตที่ 6 ที่แอดเดรส 02H โดยถ้าเป็น “1” จะเป็นการทำงานในโหมด 12 ชั่วโมง และเมื่อเลือกแบบ 12 ชั่วโมง ที่บิต 5 ในแอดเดรส 02H นั้นจะใช้แสดงค่า AM/PM โดยถ้าบิตนี้เป็น “1” จะเป็น PM ในกรณีที่แสดงแบบ 24 ชั่วโมง บิตนี้จะใช้ในการแสดงค่าของหลักสิบในของหน่วยชั่วโมงด้วย

	BIT7							BIT0	
00H	CH	10 SECONDS			SECONDS				00-59
	0	10 MINUTES			MINUTES				00-59
	0	12/24	10 HR A/P	10 HR	HOURS				01-12 00-23
	0	0	0	0	0	DAY			1-7
	0	0	10 DATE		DATE				
	0	0	0	10 MONTH	MONTH				01-12
	10 YEAR				YEAR				00-99
07H	OUT	0	0	SQWE	0	0	RS1	RS0	

รูปที่ 2.14 รีจิสเตอร์ภายในไอซีฐานเวลา DS1307

ที่แอดเดรส 07H เป็นรีจิสเตอร์ควบคุมการทำงานของ SQW/OUT โดยมีรายละเอียดดังนี้

OUT (Out control): ใช้ควบคุมเอาต์พุต

SQWE (Square Wave Enable): ใช้ควบคุมออสซิลเลเตอร์ภายใน DS1307 โดยถ้าบิตนี้เป็น “1” จะเป็นการเปิดออสซิลเลเตอร์

RS (Rate Select): ใช้ควบคุมความถี่ของ Square Wave เมื่อเปิดการทำงานของออสซิเลเตอร์ โดยสามารถปรับเปลี่ยนความถี่ได้ 4 ความถี่ด้วยกันดังแสดงในตารางที่ 2.1

ตารางที่ 2.1 การควบคุมความถี่ออสซิลเลเตอร์ด้วยการเซตบิต RS1, RS0

RS1	RS0	SQW OUTPUT FREQUENCY
0	0	1 Hz
0	1	4.096 kHz
1	0	8.192 kHz
1	1	32.768 kHz

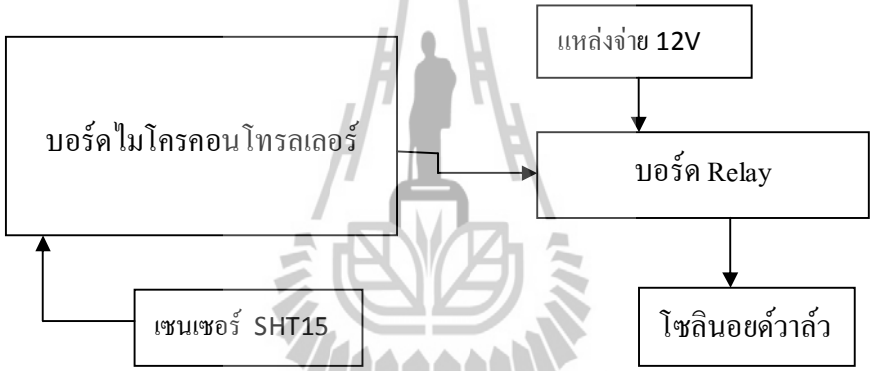
บทที่ 3

การออกแบบ

3.1 บทนำ

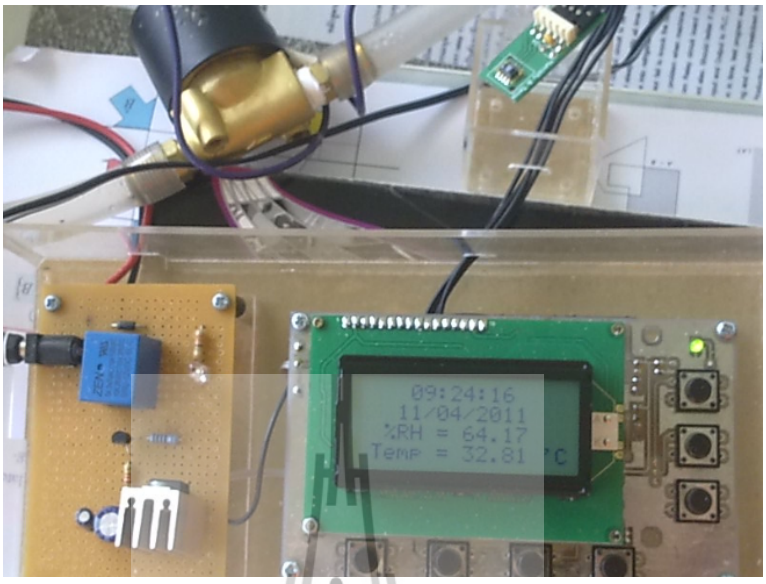
ในบทนี้จะกล่าวถึงการออกแบบเครื่องรดน้ำต้นไม้อัตโนมัติ ซึ่งแบ่งออกเป็น 2 หัวข้อ คือ การออกแบบ Hardware และ การออกแบบ Software มีการทำงานดังนี้

3.2 การออกแบบ Hardware



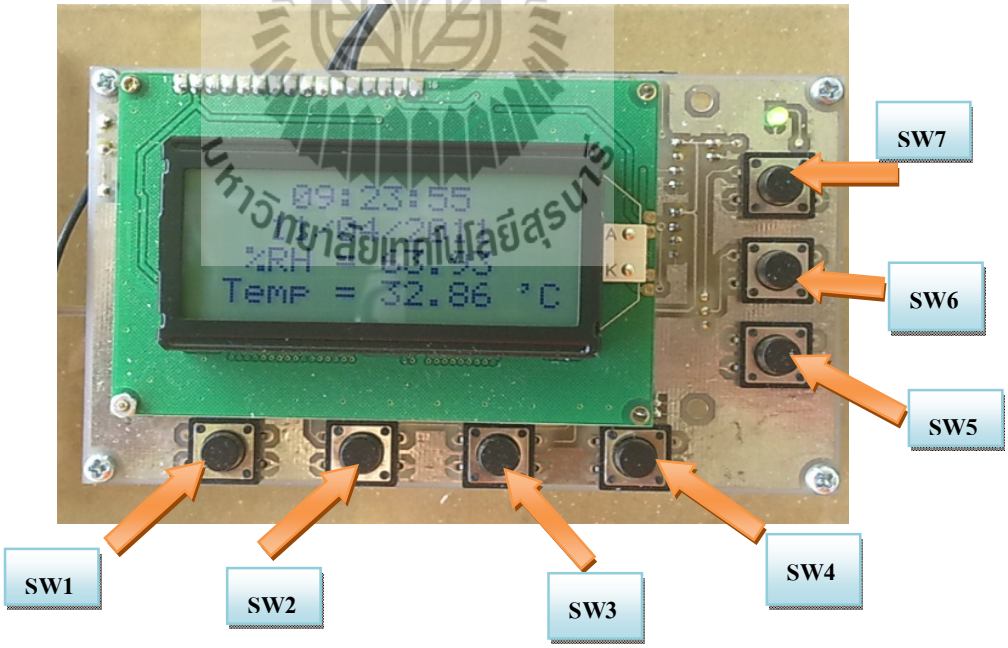
รูปที่ 3.1 ระบบเครื่องรดน้ำต้นไม้อัตโนมัติ

จากรูปที่ 3.1 เป็นแผนภาพการทำงานของเครื่องรดน้ำต้นไม้อัตโนมัติโดยมีหลักการคือสามารถตั้งเวลารดน้ำต้นไม้ได้ และ สามารถรดน้ำได้เองได้ เมื่อความชื้นที่ เซนเซอร์ SHT15 วัดได้ขณะนั้นมีค่าน้อยกว่าค่าความชื้นที่เรา set ไว้ โดยเมื่อถึงเวลาที่เราตั้งไว้หรือเมื่อค่าความชื้นที่วัดได้มีค่าน้อยกว่าค่าที่ set ไว้ บอร์ดไมโครคอนโทรลเลอร์จะทำการสั่งให้ โซลินอยด์วาล์วเปิด เพื่อรดน้ำต้นไม้ และสั่งให้ โซลินอยด์วาล์วปิด เมื่อผ่านการรดน้ำไปแล้ว 5 นาที โดยมีบอร์ดวงจร Relay ที่ใช้ในการขับ โซลินอยด์วาล์วโดยป้อนไฟ 12 V(dc) ให้กับบอร์ดวงจร Relay



รูปที่ 3.2 เครื่องรดน้ำต้นไม้อัตโนมัติ

จากรูปที่ 3.2 เป็นภาพวงจรสมบูรณ์ของเครื่องรดน้ำต้นไม้อัตโนมัติ



รูปที่ 3.3 สวิตซ์การใช้งานเครื่องรดน้ำต้นไม้อัตโนมัติ

จากรูปที่ 3.3 เป็นการแสดงสวิตซ์การใช้งานเครื่องรดน้ำต้นไม้อัตโนมัติ โดยมีการใช้งานเป็นดังนี้

สวิตซ์หมายเลข 1 ทำหน้าที่ ตั้งค่าวันเวลาจริงและทำหน้าที่เก็บข้อมูลวันและเวลาที่เราเปลี่ยนแก้ไขได้อย่างอัตโนมัติ

สวิตซ์หมายเลข 2 ทำหน้าที่ เซตค่าความชื้น เวลาในการทดลอง

สวิตซ์หมายเลข 3 ไม่มีหน้าที่ในวงจร

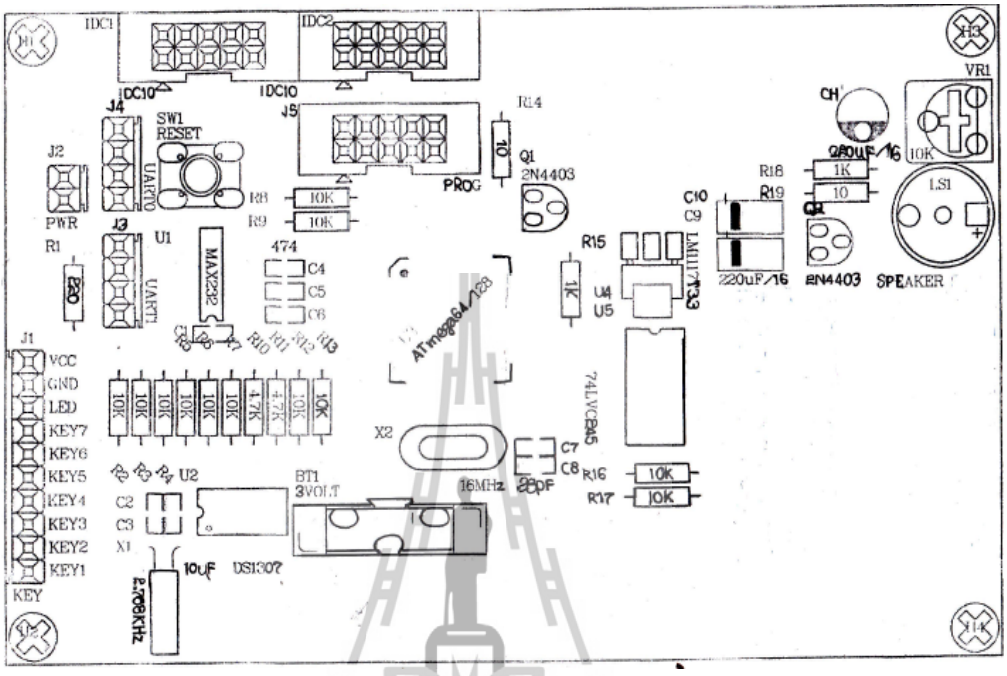
สวิตซ์หมายเลข 4 ทำหน้าที่ เลื่อนเวลา ชั่วโมง, นาที ,วินาที และสามารถตั้งค่าเวลาตามความเหมาะสม

สวิตซ์หมายเลข 5 ทำหน้าที่ เลื่อนขึ้นเวลาการปรับความชื้น เวลา และการตั้งค่าเวลาที่จะทำการทดลองและยังทำหน้าที่ในการเปิด solenoid valve ไว้ชั่วคราว ตอนที่ยังไม่ได้ทำการเซตเวลาการทดลอง

สวิตซ์หมายเลข 6 ทำหน้าที่ เลื่อนลงเวลาการปรับความชื้น เวลา และการตั้งค่าเวลาที่จะทำการทดลองและยังทำหน้าที่ในการปิด solenoid valve ไว้ชั่วคราว ตอนที่ยังไม่ได้ทำการเซตเวลาการทดลอง

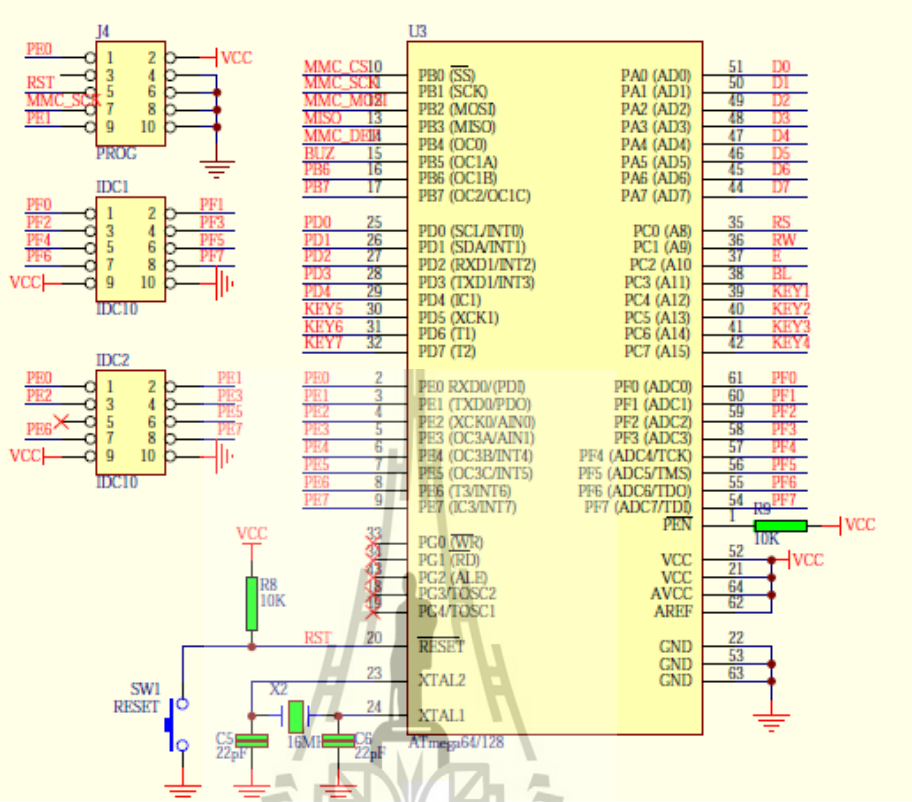
สวิตซ์หมายเลข 7 ทำหน้าที่ เคลียร์ค่าทั้งหมด ปิดสวิตซ์ solenoid valve ถาวร แล้วทำการเริ่มการตั้งโปรแกรมใหม่

3.2.1 การออกแบบไมโครคอนโทรลเลอร์



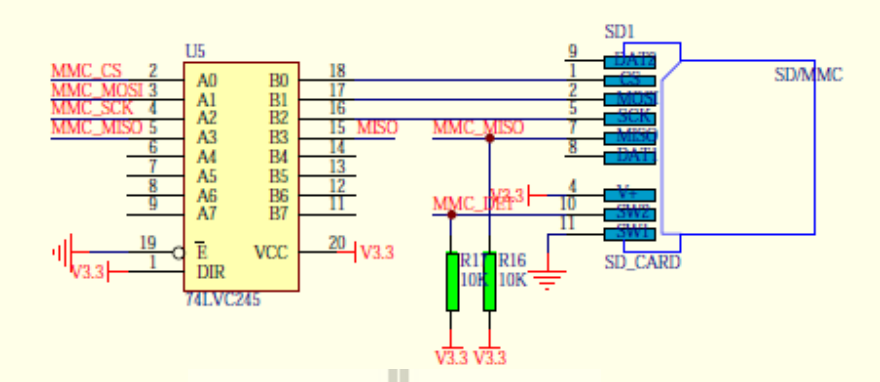
รูปที่ 3.4 วงจรบอร์ดไมโครคอนโทรลเลอร์

จากรูปที่ 3.4 เป็นลายวงจรของบอร์ดไมโครคอนโทรลเลอร์ ซึ่งประกอบด้วยอุปกรณ์อิเล็กทรอนิกส์ต่างๆ เช่น IC ATmega 128 , IC DS1307 , LM1117T33 , IC 74LVC245 , LCD , ทรานซิสเตอร์เบอร์2N4403 , MAX 232



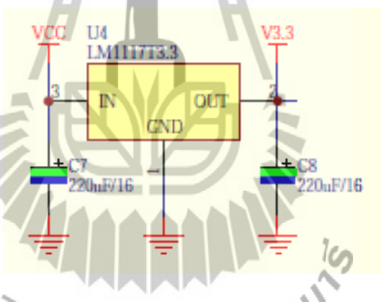
รูปที่ 3.5 วงจรการใช้งาน ATmega128

จากรูปที่ 3.5 เป็นการแสดงการต่อของ IC ATmega128 ว่าขาใดต่อ port ใด มีหน้าที่ในการใช้เขียนคำสั่งต่างๆ เพื่อส่งไปยังส่วนต่าง ๆ ซึ่งถือว่าเป็นหัวใจสำคัญของบอร์ดไมโครคอนโทรลเลอร์เลยก็ว่าได้



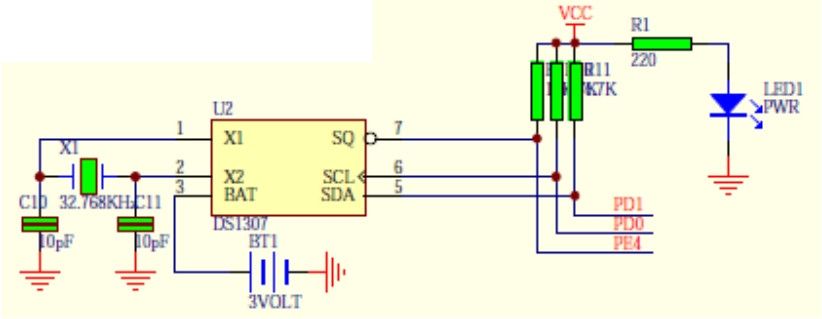
รูปที่ 3.6 วงจรการใช้งาน IC 74LVC245

จากรูปที่ 3.6 เป็นการแสดงการต่อของ IC 74LVC245 มีหน้าที่ เป็น Buffer



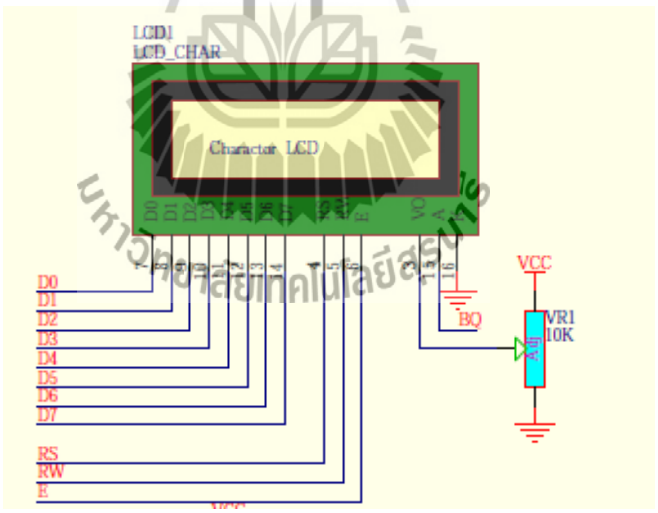
รูปที่ 3.7 วงจรการใช้งาน LM1117T3.3

จากรูปที่ 3.7 เป็นการต่อของ LM 1117T3.3 ซึ่งมีหน้าที่ ลดระดับแรงดันจาก 5 V ให้เหลือแรงดัน 3.3 V เพื่อใช้เลี้ยงวงจรและจ่ายให้กับทรานซิสเตอร์เบอร์ 2N4403 ซึ่งทำให้เกิดเสียง



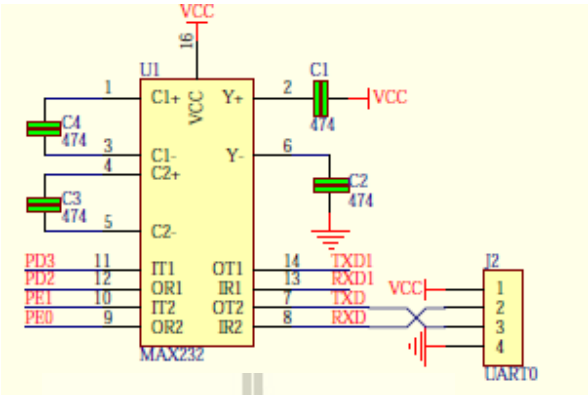
รูปที่ 3.8 วงจรการใช้งาน DS1307

จากรูปที่ 3.8 เป็นการต่อของ IC DS1307 ซึ่งมีหน้าที่เป็น Real.time clock (RTC) เพื่อแสดงวันที่ ที่เป็นปัจจุบันถึงแม้ว่าเราจะปิดเครื่องหรือเปิดเครื่องใหม่ก็ไม่ต้องตั้งเวลาอีกครั้งและมีแบตเตอรี่เป็น back up ด้วย



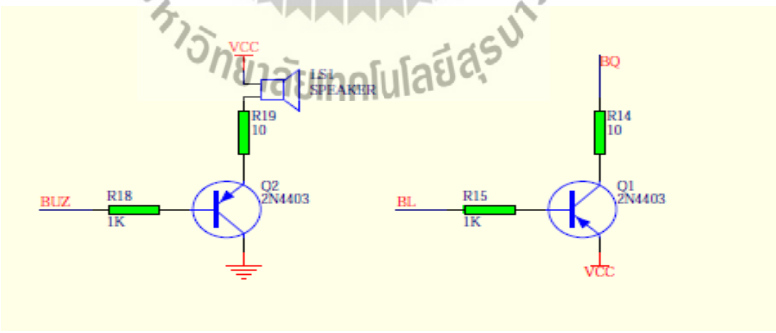
รูปที่ 3.9 วงจรการใช้งาน LCD

จากรูปที่ 3.9 เป็นการต่อของ LCD ซึ่งมีหน้าที่ แสดงผลต่างๆ บนหน้าจอแสดงผล เช่น วันที่ , เวลา , อุณหภูมิ และ ความชื้น



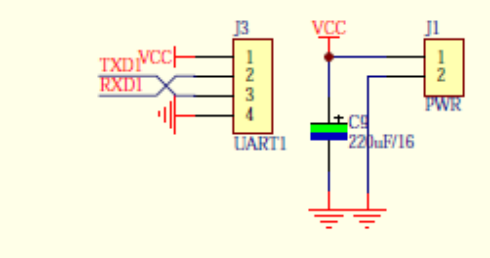
รูปที่ 3.10 วงจรการใช้งาน MAX232

จากรูปที่ 3.10 เป็นการต่อของ IC MAX 232 เป็น IC ปรับแรงดัน มีหน้าที่ในการรับส่งข้อมูลแบบอนุกรม (Serial) ระหว่างอุปกรณ์ TTL/CMOS กับ port RS-232 โดยมีช่องทางในการรับข้อมูลแบบอนุกรม 2 ช่องทาง โครงสร้างภายในของ IC MAX 232 ยังมีวงจรทวีแรงดัน (Voltage doubler) และวงจรอินเวอร์สแรงดัน (Voltage inverter) วงจรทั้งสองทำหน้าที่หลัก คือ ขยายสัญญาณก่อนที่จะส่งออกไปยัง port RS-232



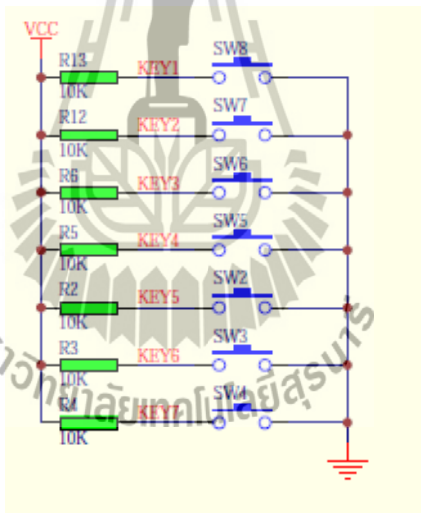
รูปที่ 3.11 วงจรการใช้งาน 2N4403

จากรูปที่ 3.11 เป็นการต่อของทรานซิสเตอร์ เบอร์ 2N4403 ซึ่งมีหน้าที่ ทำให้เกิดเสียงเตือนเมื่อถึงเวลาที่รดน้ำต้นไม้ และเมื่อรดน้ำต้นไม้เสร็จแล้ว



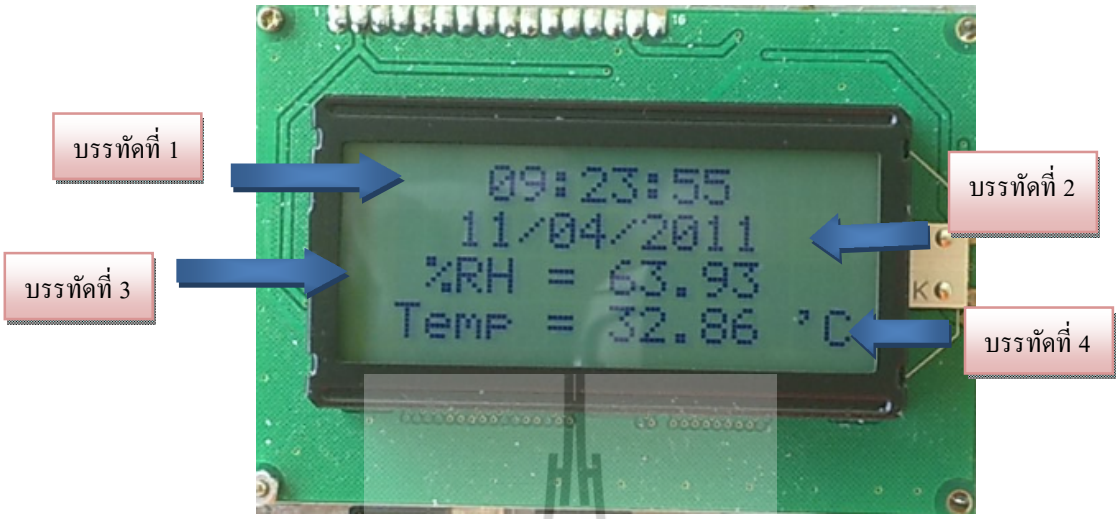
รูปที่ 3.12 วงจรการใช้งาน ไฟเลี้ยง 5 V

จากรูปที่ 3.12 เป็นการต่อ port แหล่งจ่ายให้กับไมโครคอนโทรลเลอร์ซึ่งได้มาจากการแปลงไฟ 12 V จากบอร์ดวงจร Relay เป็นไฟ 5V



รูปที่ 3.13 วงจรการใช้งาน Switch

จากรูปที่ 3.13 เป็นการออกแบบ Switch ที่จะใช้ในการกดปุ่มเพื่อตั้งค่าเวลา , วันที่ , และตั้งเวลาเพื่อรดน้ำต้นไม้



รูปที่ 3.14 การแสดงผลที่หน้าจอ LCD

จากรูปที่ 3.14 เป็นการแสดงผลที่หน้าจอ LCD เมื่อพร้อมใช้งานโดยหน้าจอ LCD จะแสดงผลดังนี้

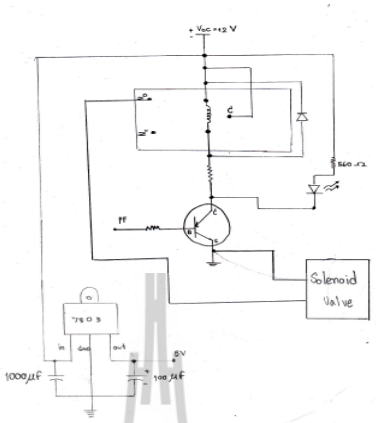
บรรทัดที่ 1 แสดง วัน:วัน/เดือน/ปี

บรรทัดที่ 2 แสดง เวลา: ชั่วโมง : นาที : วินาที

บรรทัดที่ 3 แสดง ค่าความชื้น : %RH = ?

บรรทัดที่ 4 แสดง ค่าอุณหภูมิ: Temp = ?

3.2.2 การออกแบบ Relay และ โซลินอยด์วาล์ว



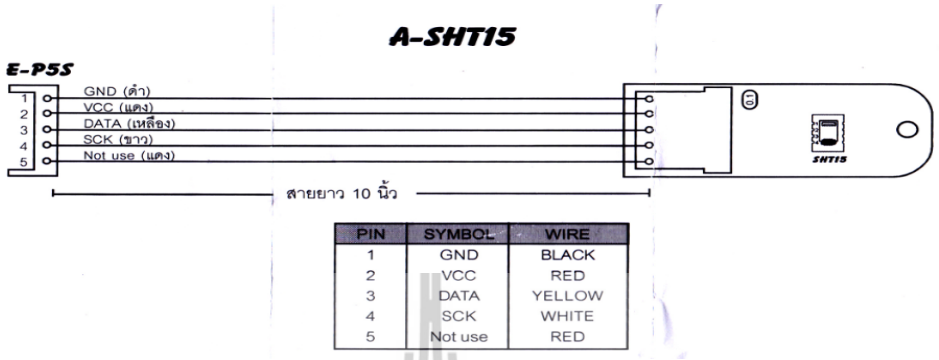
รูปที่ 3.15 วงจร Relay ขับ โซลินอยด์วาล์ว

จากรูปที่ 3.15 เป็นการออกแบบวงจร Relay เพื่อใช้ในการขับ โซลินอยด์วาล์ว โดยใช้ไฟ 12 V(dc) และมี Regulator เบอร์ 7805 เพื่อแปลงไฟ 12 V(dc) เป็นไฟ 5 V(dc) เพื่อจ่ายให้กับบอร์ดไมโครคอนโทรลเลอร์

รูปที่ 3.16 บอร์ด Relay ต่อกับ โซลินอยด์วาล์ว

จากรูปที่ 3.16 เป็นรูปวงจรของบอร์ด Relay เมื่อต่อเสร็จแล้ว ต่อเข้ากับ โซลินอยด์วาล์ว

3.2.3 การออกแบบ เซนเซอร์ SHT15



รูปที่ 3.17 วงจร เซนเซอร์ SHT15

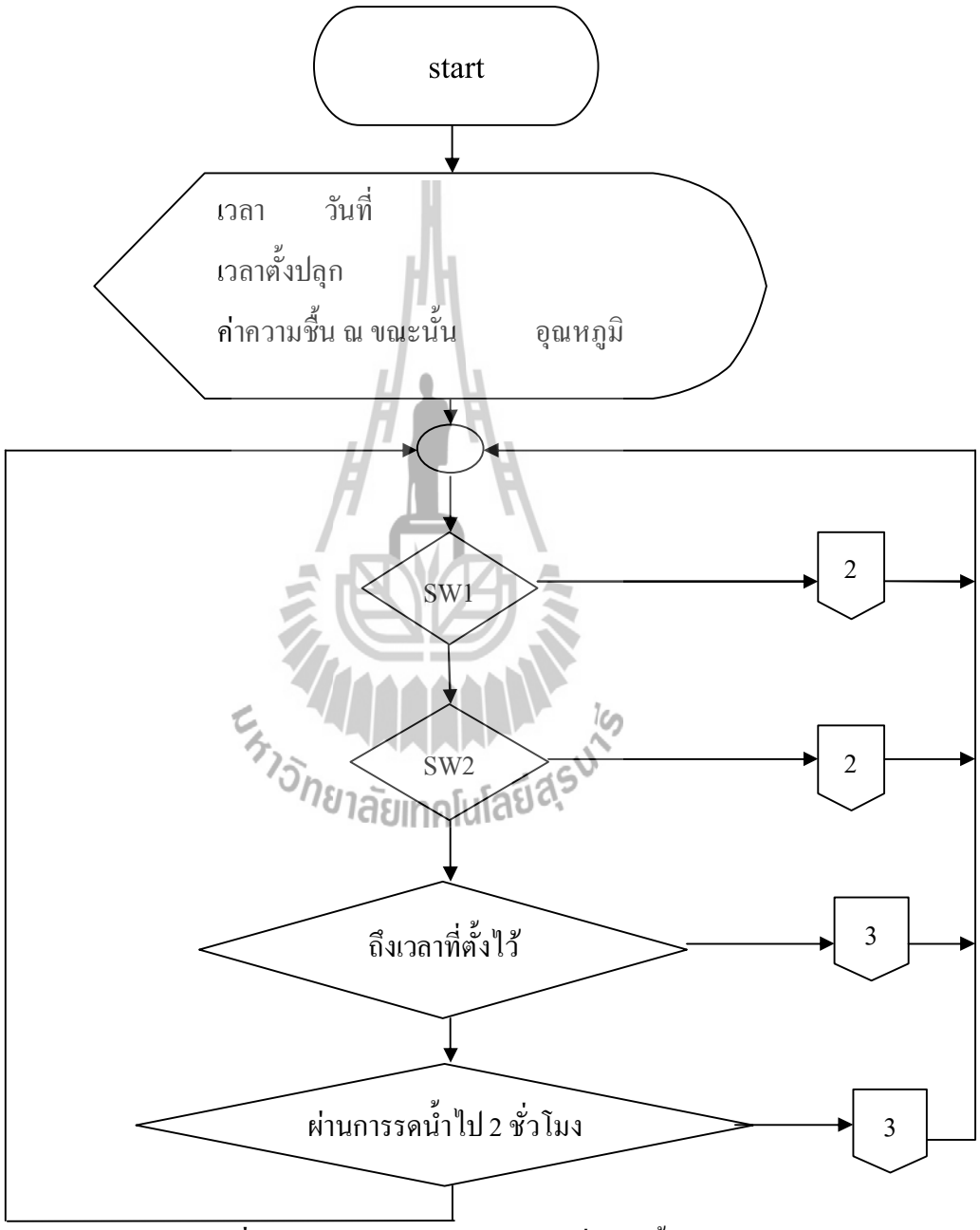
จากรูปที่ 3.17 เป็นการแสดงการต่อของ เซนเซอร์ SHT 15 และมีสายเชื่อมต่อไปยังบอร์ดไมโครคอนโทรลเลอร์ ซึ่งมีหน้าที่วัดความชื้นและอุณหภูมิ

รูปที่ 3.18 เซนเซอร์ SHT15

จากรูปที่ 3.18 เป็นรูป เซนเซอร์ SHT15 ที่ต่อเสร็จแล้ว สามารถนำไปใช้วัดความชื้น และอุณหภูมิได้เลย

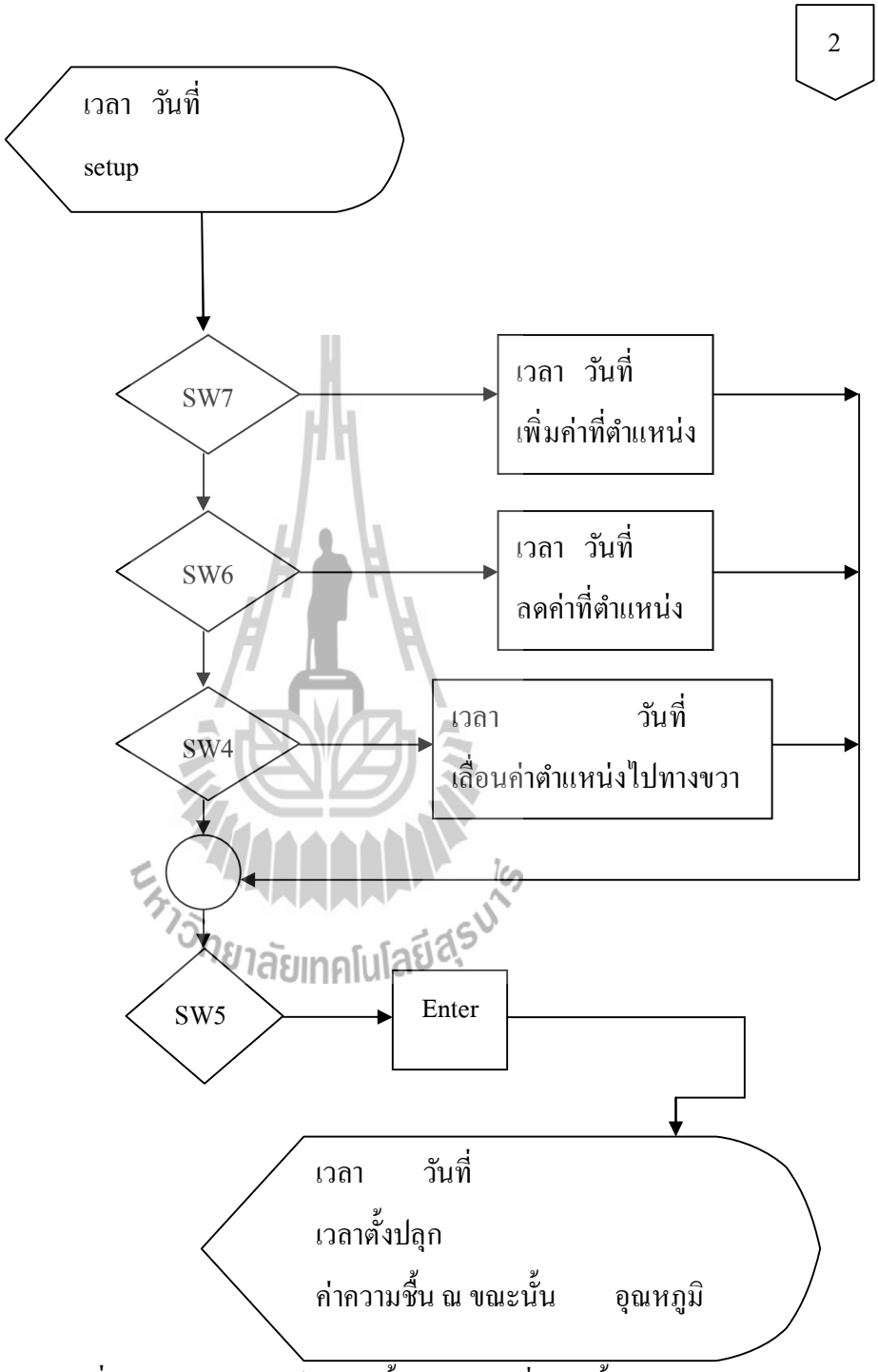
3.3 การออกแบบ Software

3.3.1 Flow chat



รูปที่ 3.19 แผนภาพการทำงานของเครื่องรดน้ำต้นไม้

จากรูปที่ 3.19 สามารถอธิบายลำดับขั้นตอนการทำงานของเครื่องรดน้ำต้นไม้อัตโนมัติ เป็นดังนี้ เมื่อเปิดเครื่องหน้าจอแสดงผลจะแสดง เวลา วันที่ เวลาที่ตั้งปลุก ค่าความชื้น และอุณหภูมิ จากนั้น ถ้าเรากดสวิตซ์หมายเลข 1 เราจะสามารถตั้งเวลาที่แท้จริง และวันที่จริงได้ สวิตซ์หมายเลข 2 จะเป็นการตั้งค่าความชื้น และเลือกว่าจะรดน้ำกี่ครั้ง เมื่อถึงเวลาที่เราตั้งไว้แล้ว เครื่องรดน้ำต้นไม้อัตโนมัติจะรดน้ำต้นไม้ แต่ถ้าเรารดน้ำต้นไม้ไปแล้วค่าความชื้นที่วัดได้ยังมีค่าน้อยกว่าค่าที่ตั้งไว้ เครื่องรดน้ำต้นไม้อัตโนมัติจะทำการรดน้ำต้นไม้อีกไปเรื่อย ๆ แต่ถ้าหากตรวจสอบแล้วพบว่า เพิ่งผ่านการรดน้ำต้นไม้ไปเมื่อ 2 ชั่วโมงที่ผ่านมา เครื่องจะไม่รดน้ำต้นไม้ เพื่อเป็นการประหยัดน้ำอีกด้วย



รูปที่ 3.20 แผนภาพฟังก์ชันการตั้งเวลาของเครื่องรดน้ำต้นไม้

จากรูปที่ 3.20 สามารถอธิบายลำดับขั้นตอนการทำงานในการตั้งเวลา วันที่ และเวลาที่ใช้ในการตั้งปลุกเตือนรดน้ำต้นไม้ ได้ดังนี้

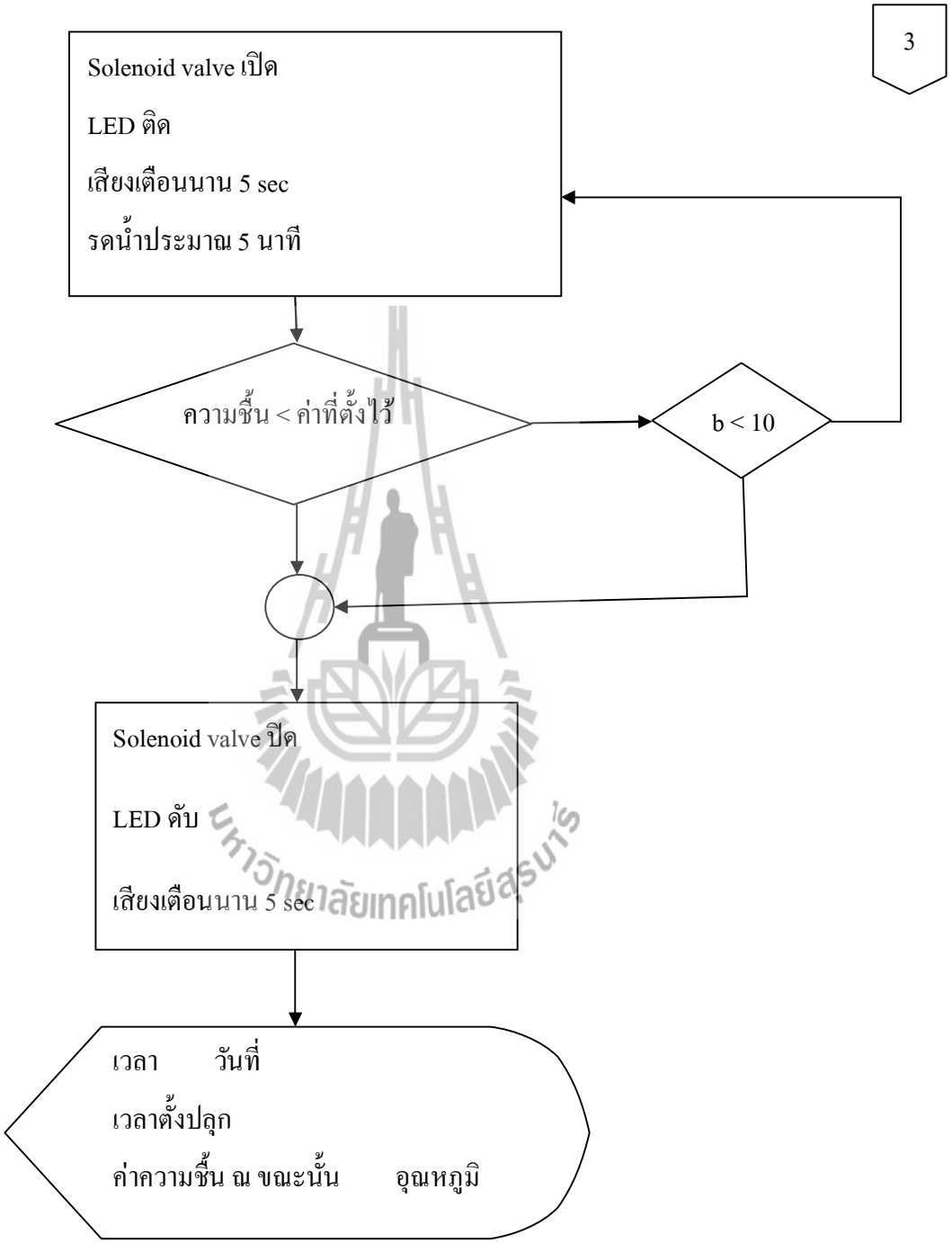
ถ้าเรากดสวิตซ์หมายเลข 7 จะเพิ่มค่าที่ตำแหน่งนั้นๆ

ถ้าเรากดสวิตซ์หมายเลข 6 จะลดค่าที่ตำแหน่งนั้นๆ

ถ้าเรากดสวิตซ์หมายเลข 4 จะเลื่อนค่าตำแหน่งไปทางขวา

ถ้าเรากดสวิตซ์หมายเลข 2 ตั้งค่าความชื้น เลือกว่าจะรดกี่ครั้ง ตั้งเวลาในการรดน้ำต้นไม้

ถ้าเรากดสวิตซ์หมายเลข 5 Enter



รูปที่ 3.21 แผนภาพฟังก์ชันการรดน้ำต้นไม้ของเครื่องรดน้ำต้นไม้

จากรูปที่ 3.21 สามารถอธิบายลำดับขั้นตอนการทำงานในการรดน้ำต้นไม้ เมื่อถึงเวลาที่ตั้งไว้หรือเมื่อค่าความชื้นที่วัดได้มีค่าน้อยกว่าค่าที่ตั้งไว้ โซลินอยด์วาล์ว จะเปิด LED จะติด มีเสียงเตือน เมื่อรดน้ำต้นไม้ และเมื่อเวลาผ่านไป 5 นาที โซลินอยด์วาล์ว จะปิด LED จะดับ มีเสียงเตือน แสดงว่าหยุดรดน้ำต้นไม้

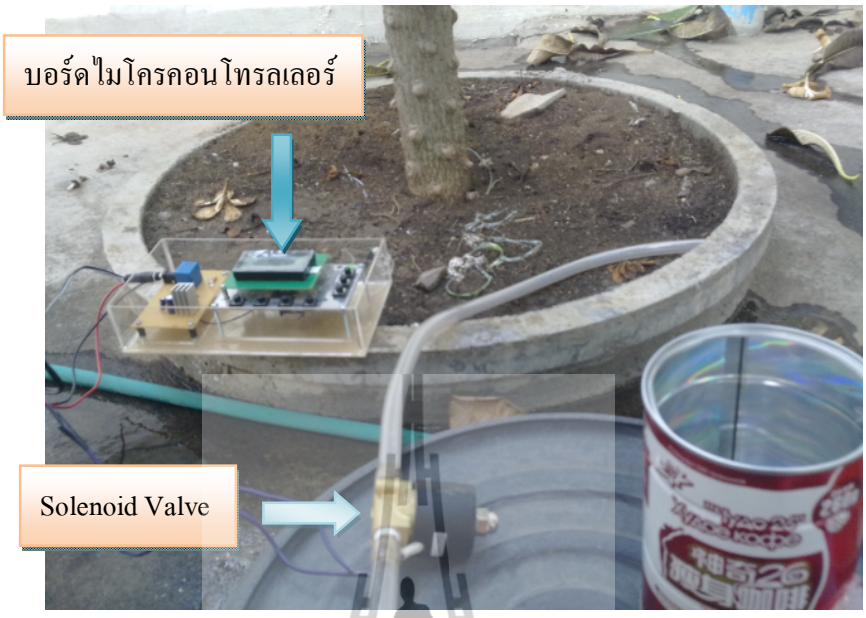
บทที่ 4

ผลการทดลอง

4.1 บทนำ

ในบทนี้เราจะกล่าวถึง การทดสอบเก็บข้อมูลค่าความชื้นเพื่อนำไปหาค่าเฉลี่ยและทดสอบการทำงานของเครื่องรดน้ำต้นไม้อัตโนมัติ เมื่อตั้งเวลาไว้และเมื่อกำหนดระดับความชื้นไว้

รูปที่ 4.1 แสดงวิธีการทดสอบวัดค่าความชื้น และอุณหภูมิ



รูปที่ 4.2 การทดลองรดน้ำต้นไม้

รูปที่ 4.3 บริเวณที่ทำการทดลอง

4.2 การวัดค่าความชื้นและอุณหภูมิ

การวัดค่าความชื้นในดินที่มีลักษณะความชื้นและอุณหภูมิแตกต่างกัน เพื่อที่จะนำมาหาค่าความชื้นเฉลี่ย ซึ่งในการทดลองแต่ละครั้งนั้นมีช่วงเวลาที่ห่างกันประมาณ 1 ชั่งโมง ภายใน 1 วัน มีผลการทดลองดังตารางต่อไปนี้

ตารางที่ 4 . 1 การวัดค่าความชื้นและอุณหภูมิ

ครั้งที่	ความชื้น (%)	อุณหภูมิ (°C)	ลักษณะของดิน
1	67.45	28.65	แห้ง
2	76.32	27.87	ชื้น
3	78.34	27.54	ชื้น
4	54.67	30.54	แห้ง
5	70.71	27.91	ชื้น
6	75.32	27.83	ชื้น
7	64.87	28.45	แห้ง
8	65.23	28.53	แห้ง
9	69.12	28.90	แห้ง
10	57.98	29.87	แห้ง
	ค่าเฉลี่ย = 68	**ค่าเฉลี่ย** = 31.25	

จากตารางที่ 4.1 พบว่า การทดลองวัดค่าความชื้นมีค่าเฉลี่ยเท่ากับ 68 %

4.3 การทดสอบการทำงานของเครื่องรดน้ำต้นไม้อัตโนมัติเมื่อทำการตั้งเวลา

ในการทดสอบการทำงานของเครื่องรดน้ำต้นไม้อัตโนมัติ เมื่อกำหนดเวลาที่แตกต่างกันจะให้ผลการทำงานดังตารางต่อไปนี้

ตารางที่ 4 . 2 ผลการทดสอบการทำงานของเครื่องรดน้ำต้นไม้อัตโนมัติ เมื่อถึงเวลาที่ตั้งไว้

วัน/เดือน/ปี	เวลา (นาฬิกา)	ผลการทำงาน	ความชื้น (%)		อุณหภูมิ (°C)	
			ก่อน	หลัง	ก่อน	หลัง
14/3/54	08.00	ทำงาน	76.78	77.05	26.64	26.25
	14.00	ทำงาน	59.83	60.72	28.73	28.34
15/3/54	09.00	ทำงาน	72.62	73.01	27.87	27.12
	15.00	ทำงาน	54.21	57.85	29.31	29.02
16/3/54	10.00	ทำงาน	69.83	70.11	27.76	27.54
	16.00	ทำงาน	55.86	57.02	26.56	26.12
17/3/54	08.00	ทำงาน	79.35	80.14	28.86	28.43
	17.00	ทำงาน	63.82	65.35	27.68	27.19
18/3/54	09.00	ทำงาน	70.79	71.85	27.95	27.65
	16.00	ทำงาน	56.02	58.13	28.97	28.68

จากตารางที่ 4.2 พบว่า การทดสอบการทำงานของเครื่องรดน้ำต้นไม้อัตโนมัติ จะเห็นว่า เครื่องรดน้ำต้นไม้อัตโนมัติทำงานตามเวลาที่ตั้งไว้ ไม่ว่าความชื้นและอุณหภูมิจะมีค่าอยู่ในระดับใด

4.4 การทดสอบการทำงานของเครื่องรดน้ำต้นไม้อัตโนมัติ เมื่อกำหนดความชื้น

ในการสอบการทำงานของเครื่องรดน้ำต้นไม้อัตโนมัติ เมื่อความชื้นในดินน้อยกว่า 68 % ซึ่งได้ทำการทดสอบในระหว่างวันที่ 9 – 10 เมษายน 2554 โดยวันที่ 9 เมษายน 2554 ได้ทำการวัดในช่วงเวลาตั้งแต่ 15.00 -20.00 น. และในวันที่ 10 เมษายน 2554 ได้ทำการวัดในช่วงเวลาตั้งแต่ 07.00 – 18.00 น. ซึ่งในแต่ละวันนั้นจะมีช่วงเวลาที่วัดห่างเท่ากัน คือ ประมาณ 30 นาที ได้ผลการทดสอบ ดังตาราง

ตารางที่ 4 .3 ผลการทดสอบการทำงานของเครื่องรดน้ำต้นไม้อัตโนมัติเมื่อกำหนดค่าความชื้น

วัน/เดือน/ปี	เวลา (นาฬิกา)	ความชื้น (%)	อุณหภูมิ (°C)	การทำงาน	ผลการทำงาน
09/04/54	15.00	60.15	36.82	ทำงาน	ทำงาน
	15.30	61.35	36.14	ทำงาน	ทำงาน
	16.00	62.72	35.89	ทำงาน	ทำงาน
	16.30	63.07	35.45	ทำงาน	ทำงาน
	17.00	65.08	35.21	ทำงาน	ทำงาน
	17.30	66.97	34.24	ทำงาน	ทำงาน
	18.00	66.85	33.82	ทำงาน	ทำงาน
	18.30	67.98	32.16	ทำงาน	ทำงาน
	19.00	69.04	31.79	ไม่ทำงาน	ไม่ทำงาน
	19.30	70.21	30.77	ไม่ทำงาน	ไม่ทำงาน
	20.00	71.84	29.64	ไม่ทำงาน	ไม่ทำงาน

ตารางที่ 4 .3 (ต่อ) ผลการทดสอบการทำงานของเครื่องรดน้ำต้นไม้อัตโนมัติเมื่อกำหนดค่าความชื้น

วัน/เดือน/ปี	เวลา (นาฬิกา)	ความชื้น (%)	อุณหภูมิ (°C)	การทำงาน	ผลการทำงาน
10/04/54	07.00	75.99	29.19	ไม่ทำงาน	ไม่ทำงาน
	07.30	74.63	29.36	ไม่ทำงาน	ไม่ทำงาน
	08.00	73.16	30.16	ไม่ทำงาน	ไม่ทำงาน
	08.30	70.01	30.88	ไม่ทำงาน	ไม่ทำงาน
	09.00	66.57	31.99	ทำงาน	ทำงาน
	09.30	63.82	32.24	ทำงาน	ทำงาน
	10.00	60.19	33.82	ทำงาน	ทำงาน
	10.30	58.25	34.96	ทำงาน	ทำงาน
	11.00	56.68	35.81	ทำงาน	ทำงาน
	11.30	55.38	36.21	ทำงาน	ทำงาน
	12.00	54.14	37.02	ทำงาน	ทำงาน
	12.30	48.59	38.16	ทำงาน	ทำงาน
	13.00	47.27	38.82	ทำงาน	ทำงาน
	13.30	46.69	39.64	ทำงาน	ทำงาน
	14.00	50.89	38.13	ทำงาน	ทำงาน
	14.30	52.57	37.91	ทำงาน	ทำงาน
	15.00	54.05	37.06	ทำงาน	ทำงาน
	15.30	57.98	36.42	ทำงาน	ทำงาน
	16.00	60.17	35.86	ทำงาน	ทำงาน
	16.30	61.33	35.01	ทำงาน	ทำงาน
	17.00	63.83	34.22	ทำงาน	ทำงาน
	17.30	65.09	33.96	ทำงาน	ทำงาน
	18.00	67.85	31.97	ทำงาน	ทำงาน

จากตารางที่ 4.3 พบว่า เครื่องรดน้ำต้นไม้อัตโนมัติ ทำงานเมื่อความชื้นในดินน้อยกว่า 68 % และพบว่า เครื่องรดน้ำต้นไม้อัตโนมัติไม่ทำงาน เมื่อความชื้นในดินมากกว่าหรือเท่ากับ 68 %

บทที่ 5

วิเคราะห์ผลการทดลอง

5.1 บทนำ

ในบทนี้เป็นการวิเคราะห์ผลการทดลองการทดสอบการทำงานของเครื่องรดน้ำต้นไม้อัตโนมัติ ซึ่งได้นำผลการทดลองจากบทที่ 4 มาวิเคราะห์ผลเป็นดังนี้

5.2 วิเคราะห์ผลการทดลอง

จากตารางที่ 4.1 การทดสอบเก็บข้อมูลค่าความชื้นทั้งหมด 10 ครั้ง จะเห็นว่าค่าความชื้นมีค่าน้อย เมื่ออุณหภูมิสูงและดินมีลักษณะแห้ง ซึ่งในทางตรงกันข้าม เมื่ออุณหภูมิต่ำและดินมีลักษณะเปียกจะทำให้ค่าความชื้นมีค่ามาก เมื่อนำมาหาค่าความชื้นเฉลี่ย พบว่ามีค่าเฉลี่ยเท่ากับ 68 %

จากผลการทดสอบการทำงานของเครื่องรดน้ำต้นไม้อัตโนมัติตารางที่ 4.2 จะเห็นว่า ในวันที่ 14 มีนาคม 2554 เวลา 08.00 น. เครื่องจะทำงาน ซึ่งวัดค่าความชื้นก่อนรดน้ำต้นไม้ได้เท่ากับ 76.78 % และหลังรดน้ำต้นไม้วัดได้เท่ากับ 77.05 % ซึ่งค่าความชื้นที่วัดได้จะมีค่ามากกว่าค่าความชื้นที่กำหนดไว้ เนื่องจากอากาศในตอนเช้ายังมีความชื้นอยู่ และในเวลา 14.00 น. เครื่องจะทำงาน ซึ่งวัดค่าความชื้นก่อนรดน้ำต้นไม้ได้เท่ากับ 59.63 % และหลังรดน้ำต้นไม้วัดได้เท่ากับ 60.72 % ซึ่งค่าความชื้นที่วัดได้จะมีค่าน้อยกว่าค่าความชื้นที่กำหนดไว้ เนื่องจากอากาศร้อนอบอ้าวในช่วงกลางวันทำให้ความชื้นในดินลดลงมาก

จากผลการทดสอบการทำงานของเครื่องรดน้ำต้นไม้อัตโนมัติตารางที่ 4.2 จะเห็นว่า ในวันที่ 15 มีนาคม 2554 เวลา 09.00 น. เครื่องจะทำงาน ซึ่งวัดค่าความชื้นก่อนรดน้ำต้นไม้ได้เท่ากับ 72.62 % และหลังรดน้ำต้นไม้วัดได้เท่ากับ 73.01 % ซึ่งค่าความชื้นที่วัดได้จะมีค่ามากกว่าค่าความชื้นที่กำหนดไว้ เนื่องจากอากาศในตอนเช้ายังมีความชื้นอยู่ และในเวลา 15.00 น. เครื่องจะทำงาน ซึ่งวัดค่าความชื้นก่อนรดน้ำต้นไม้ได้เท่ากับ 54.21 % และหลังรดน้ำต้นไม้วัดได้เท่ากับ 57.85 %ซึ่งค่าความชื้นที่วัดได้จะ

มีค่าน้อยกว่าค่าความชื้นที่กำหนดไว้ เนื่องจากอากาศร้อนอบอ้าวในช่วงกลางวันทำให้ความชื้นในดินลดลงมาก

จากผลการทดสอบการทำงานของเครื่องรดน้ำต้นไม้อัตโนมัติตารางที่ 4.2 จะเห็นว่า ในวันที่ 16 มีนาคม 2554 เวลา 10.00 น. เครื่องจะทำงาน ซึ่งวัดค่าความชื้นก่อนรดน้ำต้นไม้ได้เท่ากับ 69.83 % และหลังรดน้ำต้นไม้วัดได้เท่ากับ 70.11 % ซึ่งค่าความชื้นที่วัดได้จะมีค่ามากกว่าค่าความชื้นที่กำหนดไว้ เนื่องจากอากาศในตอนเช้ายังมีความชื้นอยู่ และในเวลา 16.00 น. เครื่องจะทำงาน ซึ่งวัดค่าความชื้นก่อนรดน้ำต้นไม้ได้เท่ากับ 55.86 % และหลังรดน้ำต้นไม้วัดได้เท่ากับ 57.02 % ซึ่งค่าความชื้นที่วัดได้จะมีค่าน้อยกว่าค่าความชื้นที่กำหนดไว้ เนื่องจากอากาศร้อนอบอ้าวในช่วงกลางวันทำให้ความชื้นในดินลดลงมาก

จากผลการทดสอบการทำงานของเครื่องรดน้ำต้นไม้อัตโนมัติตารางที่ 4.2 จะเห็นว่า ในวันที่ 17 มีนาคม 2554 เวลา 08.00 น. เครื่องจะทำงาน ซึ่งวัดค่าความชื้นก่อนรดน้ำต้นไม้ได้เท่ากับ 79.35 % และหลังรดน้ำต้นไม้วัดได้เท่ากับ 80.14 % ซึ่งค่าความชื้นที่วัดได้จะมีค่ามากกว่าค่าความชื้นที่กำหนดไว้ เนื่องจากอากาศในตอนเช้ายังมีความชื้นอยู่ และในเวลา 17.00 น. เครื่องจะทำงาน ซึ่งวัดค่าความชื้นก่อนรดน้ำต้นไม้ได้เท่ากับ 63.82 % และหลังรดน้ำต้นไม้วัดได้เท่ากับ 65.35 % ซึ่งค่าความชื้นที่วัดได้จะมีค่าน้อยกว่าค่าความชื้นที่กำหนดไว้ เนื่องจากอากาศร้อนอบอ้าวในช่วงกลางวันทำให้ความชื้นในดินลดลงมาก

จากผลการทดสอบการทำงานของเครื่องรดน้ำต้นไม้อัตโนมัติตารางที่ 4.2 จะเห็นว่า ในวันที่ 18 มีนาคม 2554 เวลา 09.00 น. เครื่องจะทำงาน ซึ่งวัดค่าความชื้นก่อนรดน้ำต้นไม้ได้เท่ากับ 70.79 % และหลังรดน้ำต้นไม้วัดได้เท่ากับ 71.85 % ซึ่งค่าความชื้นที่วัดได้จะมีค่ามากกว่าค่าความชื้นที่กำหนดไว้ เนื่องจากอากาศในตอนเช้ายังมีความชื้นอยู่ และในเวลา 16.00 น. เครื่องจะทำงาน ซึ่งวัดค่าความชื้นก่อนรดน้ำต้นไม้ได้เท่ากับ 56.02 % และหลังรดน้ำต้นไม้วัดได้เท่ากับ 58.13 % ซึ่งค่าความชื้นที่วัดได้จะมีค่าน้อยกว่าค่าความชื้นที่กำหนดไว้ เนื่องจากอากาศร้อนอบอ้าวในช่วงกลางวันทำให้ความชื้นในดินลดลงมาก

จากการทดสอบการทำงานของเครื่องรดน้ำต้นไม้อัตโนมัติทั้ง 5 วันพบว่า การทำงานของเครื่องรดน้ำต้นไม้จะทำงานตามเวลาที่ตั้งไว้

จากผลการทดสอบการทำงานของเครื่องรดน้ำต้นไม้เมื่อความชื้นในดินน้อยกว่า 68 % ในตารางที่ 4.3 จะเห็นว่าในวันที่ 09 เมษายน 2554 เวลา 15.00 – 18.30 น. เครื่องจะทำงาน เนื่องจากค่าความชื้นในดินมีค่าน้อยกว่า 68 % เวลา 19.00 – 20.00 น.เครื่องจะไม่ทำงาน เนื่องจากค่าความชื้นในดินมีค่ามากกว่ากว่า 68 % ทำให้อากาศในตอนเย็นมีความชื้นในดินเพิ่มมากขึ้น

จากผลการทดสอบการทำงานของเครื่องรดน้ำต้นไม้เมื่อความชื้นในดินน้อยกว่า 68 % ในตารางที่ 4.3 จะเห็นว่าในวันที่ 10 เมษายน 2554 เวลา 07.00 – 08.30 น. เครื่องจะไม่ทำงาน เนื่องจากค่าความชื้นในดินมีค่ามากกว่า 68 % เวลา 09.00 – 18.00 น. เครื่องจะทำงาน เนื่องจากค่าความชื้นในดินมีค่าน้อยกว่า 68 % เนื่องจากอากาศร้อนอบอ้าวในช่วงกลางวันทำให้ความชื้นในดินลดลงมาก

บทที่ 6

สรุปและข้อเสนอแนะ

6.1 บทนำ

ในบทนี้เป็นการสรุปผลการทดลองการทดสอบการทำงานของเครื่องรดน้ำต้นไม้อัตโนมัติ ซึ่งได้นำผลการทดลองจากบทที่ 4 มาสรุปเป็นดังนี้

6.2 สรุปผลการทดสอบ

โครงงานนี้ได้ทำการศึกษาเกี่ยวกับการทำงานของเครื่องรดน้ำต้นไม้อัตโนมัติ ซึ่งมีส่วนประกอบที่สำคัญ คือ ไมโครคอนโทรลเลอร์ เซนเซอร์วัดความชื้น และ โซลินอยด์วาล์ว เพื่อให้ทุกคนช่วยกันรักษาต้นไม้เพื่อลดภาวะโลกร้อนที่เกิดขึ้นในปัจจุบัน ซึ่งเราสามารถตั้งเวลาในการรดน้ำต้นไม้ในช่วงเวลาที่เราไม่อยู่บ้านได้อีกด้วย

จากการทดสอบการทำงาน พบว่า เครื่องรดน้ำต้นไม้จะทำงานตามเวลาที่ตั้งไว้และในช่วงเวลาที่ความชื้นที่วัดได้มีค่าน้อยกว่าค่าความชื้นที่กำหนดไว้ แต่ในช่วงเวลาที่ความชื้นมากกว่าที่กำหนดไว้เครื่องจะไม่ทำงาน ซึ่งจะเป็นการช่วยประหยัดน้ำและแรงงานคนได้อีกด้วย และช่วงเวลาที่มีความชื้นน้อยที่สุด คือ เวลา 12.00 – 14.00 น. เนื่องจากตอนกลางวันมีอากาศร้อนมาก ทำให้ความชื้นในดินลดลงมากด้วย และในช่วงเวลา 07.00 – 08.00 น. จะมีความชื้นมากเพราะอากาศในตอนเช้าเย็น

จากการทดสอบนี้ได้ผลสรุปว่า เครื่องรดน้ำต้นไม้อัตโนมัตินี้ มีประโยชน์ในด้านต่าง ๆ เช่น ด้านการเกษตรกรรม ซึ่งสามารถนำไปใช้กับพืชได้เกือบทุกชนิด และเป็นการรักษาต้นไม้เพื่อลดภาวะโลกร้อนได้อีกด้วย

6.3 ปัญหาและอุปสรรค

ปัญหาและอุปสรรคที่เกิดขึ้นในระหว่างทำการทดสอบสามารถสรุปได้ดังนี้

- ปัญหาเรื่องอุปกรณ์อิเล็กทรอนิกส์แต่ละตัว เมื่อใช้ระยะเวลาในการทำการทดลอง บางครั้งอุปกรณ์บางตัวเกิดความเสียหายทำให้การทดลองคาดเคลื่อน
- ปัญหาเกี่ยวกับ เซนเซอร์ วัดความชื้น มีความไวกับสภาพแวดล้อม ซึ่งเสียหายง่าย

6.4 ข้อเสนอแนะ

- เมื่อต้องการวัดความชื้นที่มีความแม่นยำมากขึ้น ควรเลือก เซนเซอร์ ที่มีความละเอียดมากกว่านี้
- พัฒนาโปรแกรมให้สามารถทำการตั้งเวลาและตั้งจำนวนครั้งในการรดน้ำต้นไม้ได้

บรรณานุกรม

1. http://www.sensirion.com/en/pdf/product_information/Datasheet-humidity-sensor-SHT1x.pdf

2. http://www.norgren.com/document_resources/KIP/Valves03.pdf

3. http://www.atmel.com/dyn/resources/prod_documents/doc2467.pdf

4. http://www.nsru.ac.th/e-learning/soil/lesson_4.php

5. บริษัทอีทีที จำกัด. คู่มือการใช้งานบอร์ด ET -AVR STAMP ATMEGA128 [on line] จาก :

http://www.ett.co.th/download/03AVR/03A08/ET-AVR%20STAMP%20ATmega64.pdf

6. บริษัทอีทีที จำกัด. คู่มือการใช้งานบอร์ด ET-RF24G V1.0 [on line] จาก :

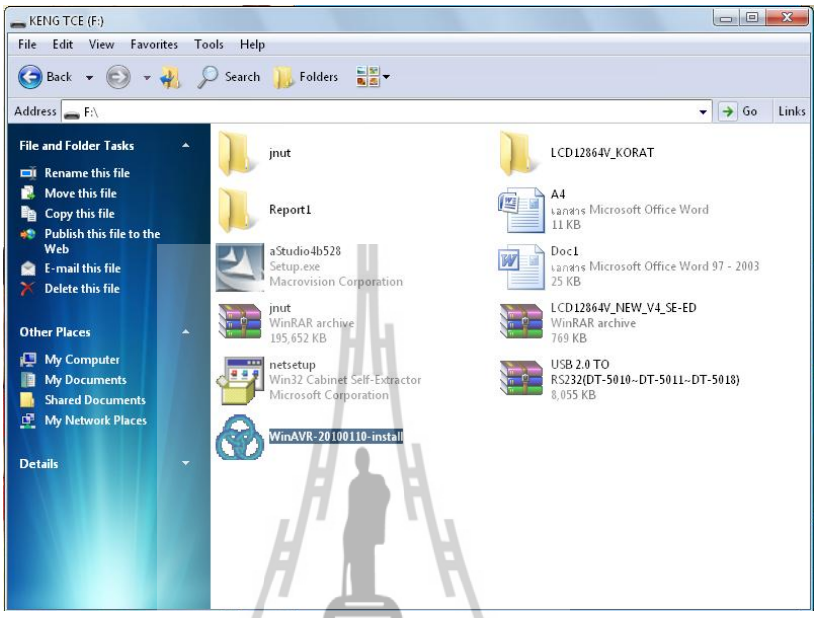
http://www.ett.co.th/download/12INTERFACE/12A25/MANUAL.rar

7. http://www.es.co.th/

ภาคผนวก ก

การติดตั้งโปรแกรม WinAVR

1. ดับเบิลคลิกที่ WinAvR.20100110.install.exe



รูปที่ ก.1 เลือกโปรแกรมที่ติดตั้ง

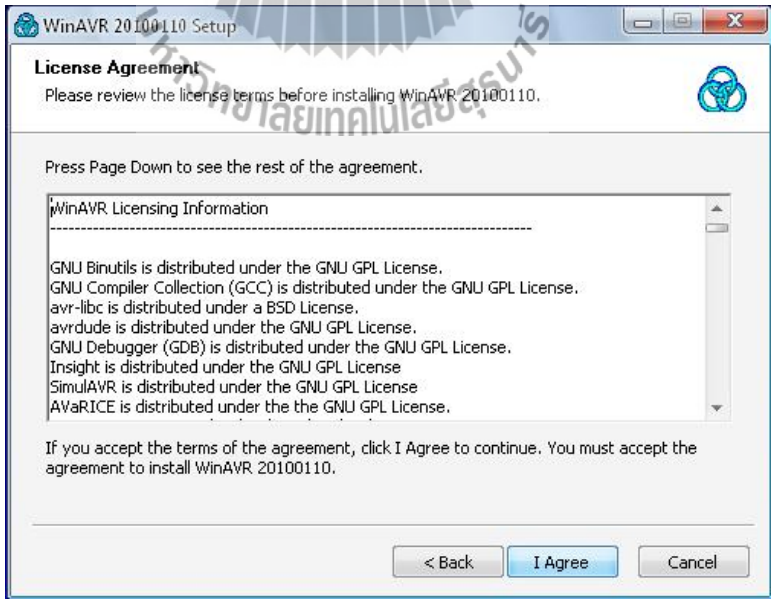
2. เลือกภาษา เสร็จแล้วคลิก OK

รูปที่ ก.2 เลือกภาษา

3. คลิก Next

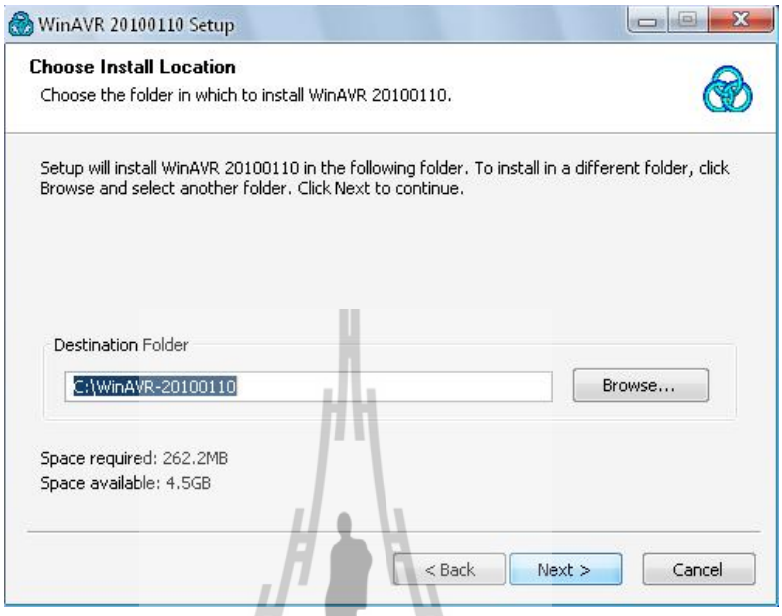
รูปที่ ก.3 หน้าต่าง Welcome to the WinAVR 20100110 setup wizard

4. คลิก I Agree



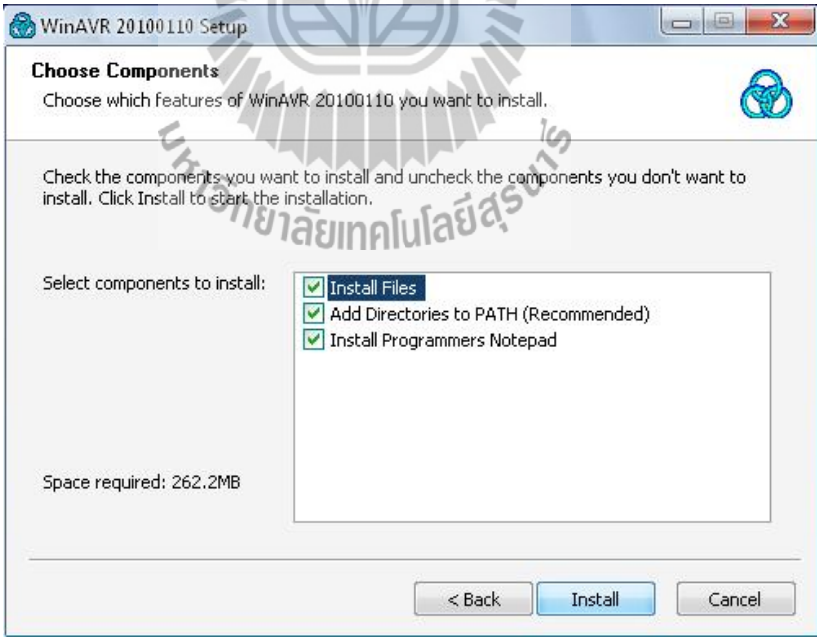
รูปที่ ก.4 หน้าต่าง License Agreement

5. เลือกที่เก็บโปรแกรม แล้วคลิก Next



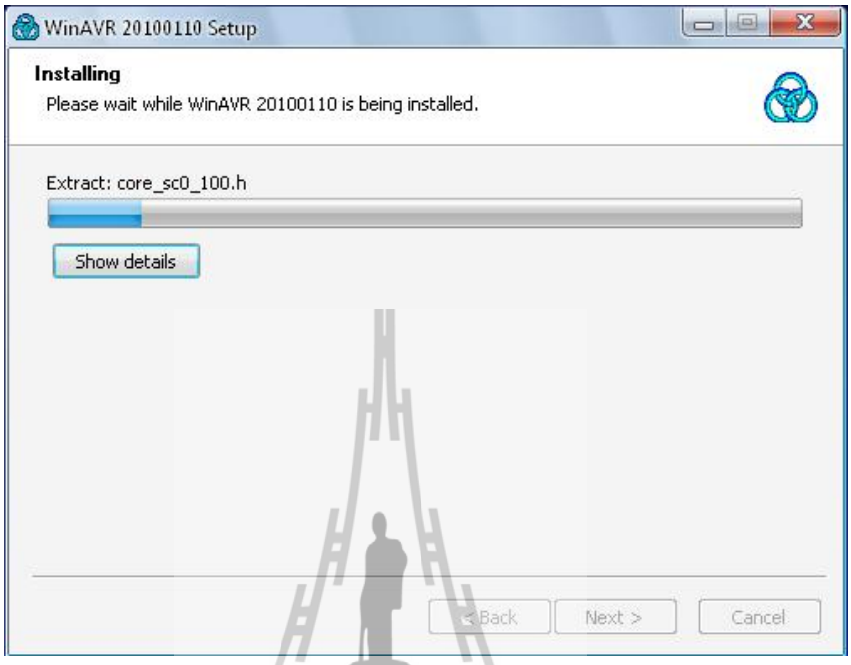
รูปที่ ก.5 หน้าต่าง Choose install location

6. คลิก Install



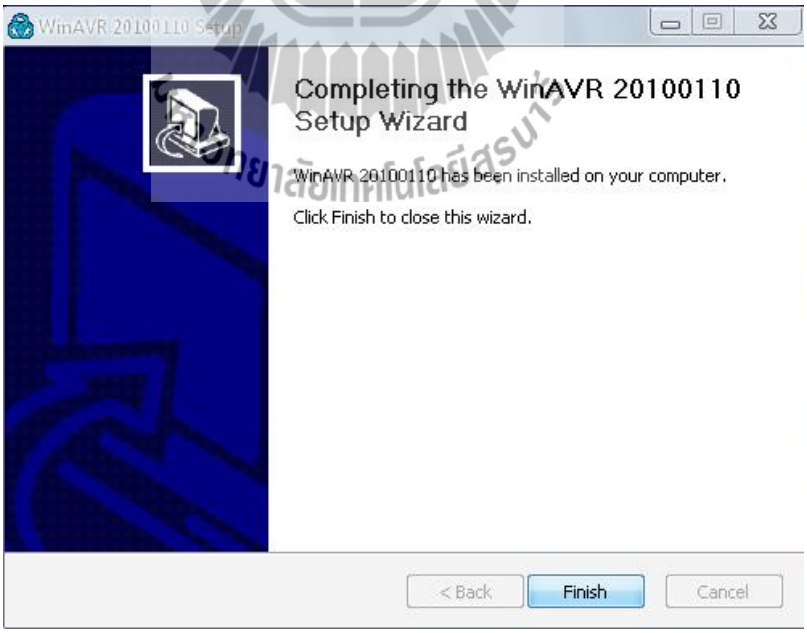
รูปที่ ก.6 หน้าต่าง Choose components

7. รอโหลดโปรแกรม



รูปที่ ก.7 หน้าต่าง Installing

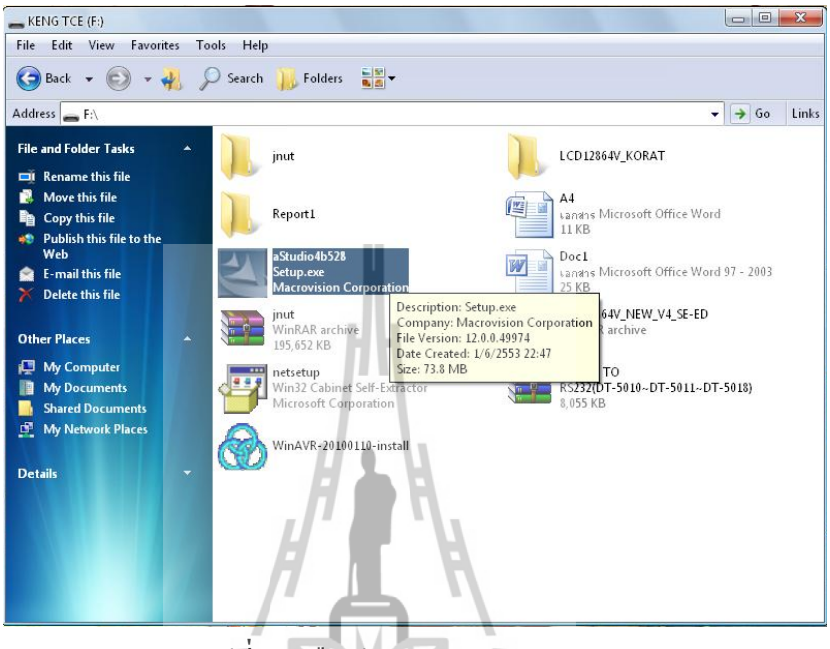
8. คลิก Finish เสร็จสิ้นการลงโปรแกรม WinAVR



รูปที่ ก.8 หน้าต่างแสดงการติดตั้งเสร็จสมบูรณ์

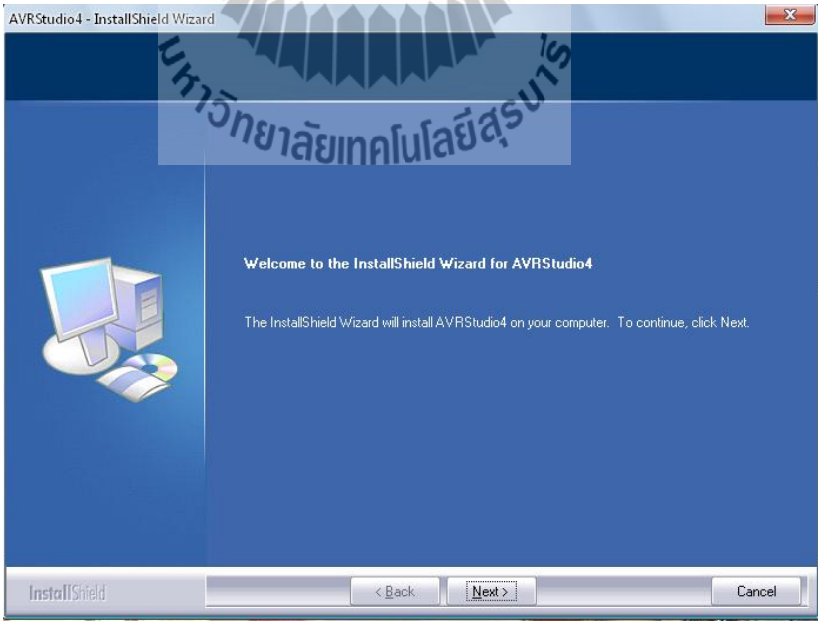
การติดตั้งโปรแกรม AVR Studio 4

1. ดับเบิลคลิกไฟล์ aStudio4b528.exe



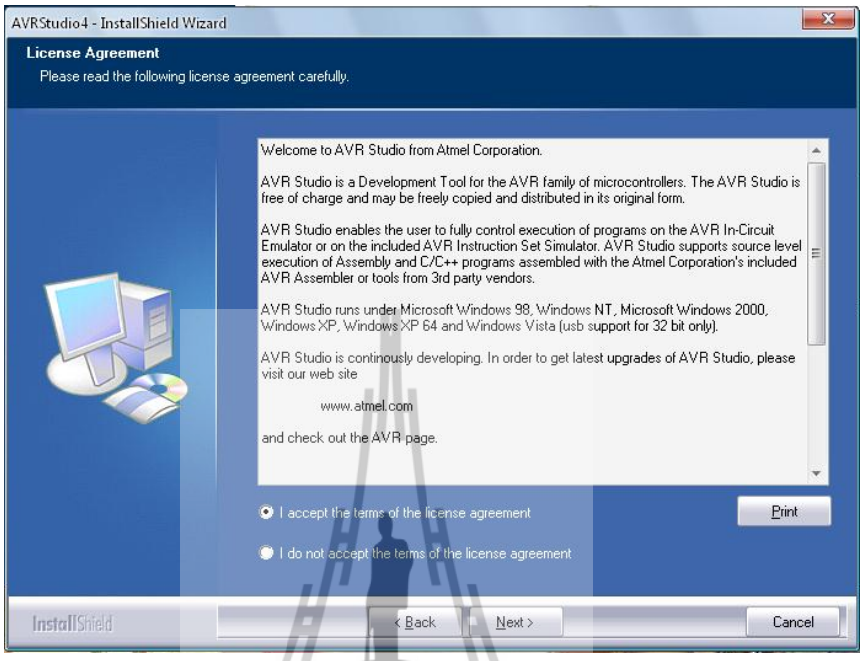
รูปที่ ก.9 เลือกโปรแกรม aStudio4b528

2. คลิก Next



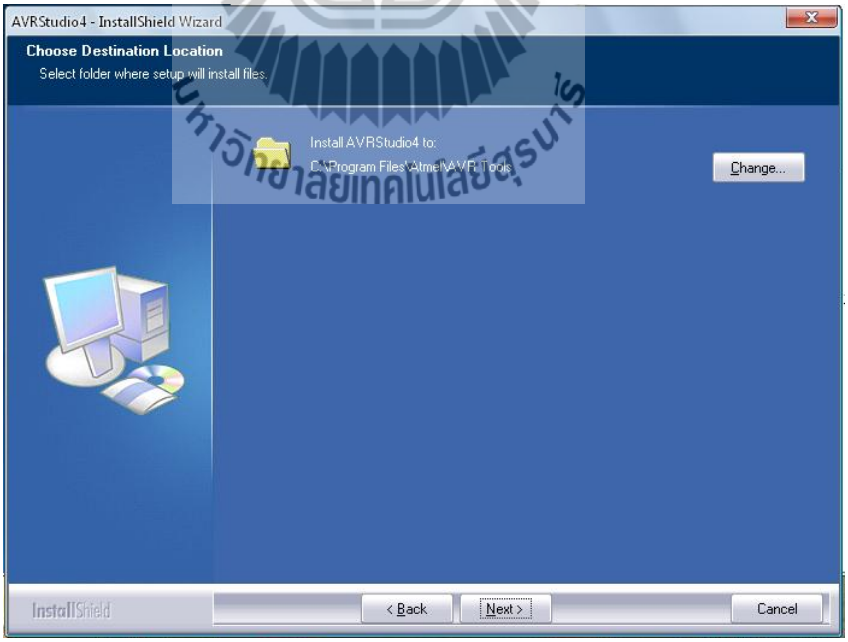
รูปที่ ก.10 หน้าต่าง Welcome to the installshield wizard for AVRStudio4

3. เลือก I accept the of the license agreement แล้วคลิก Next



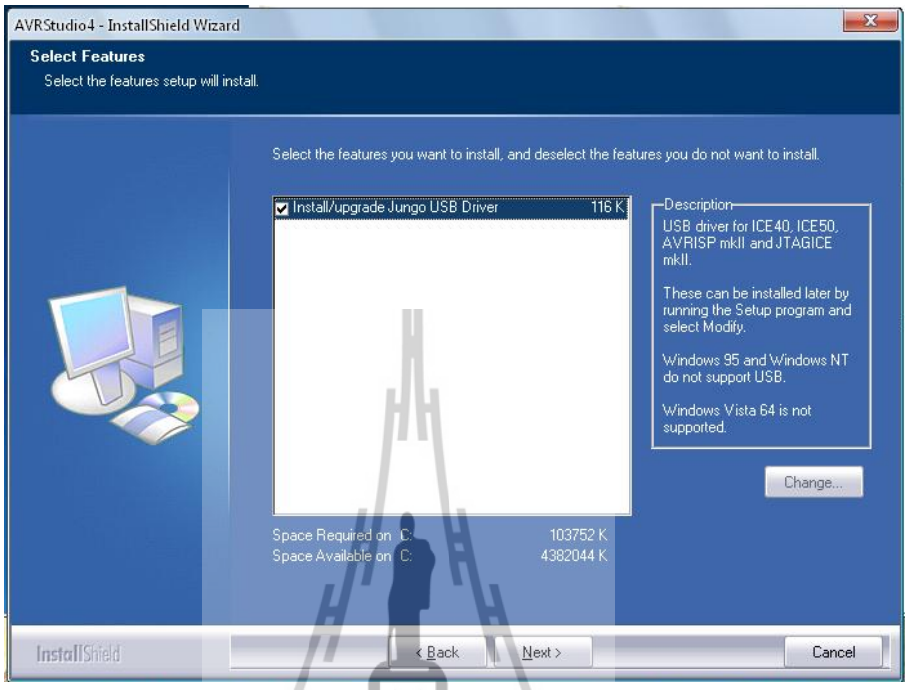
รูปที่ ก.11 หน้าต่าง License agreement

4. เลือกที่จัดเก็บโปรแกรม แล้วคลิก Next



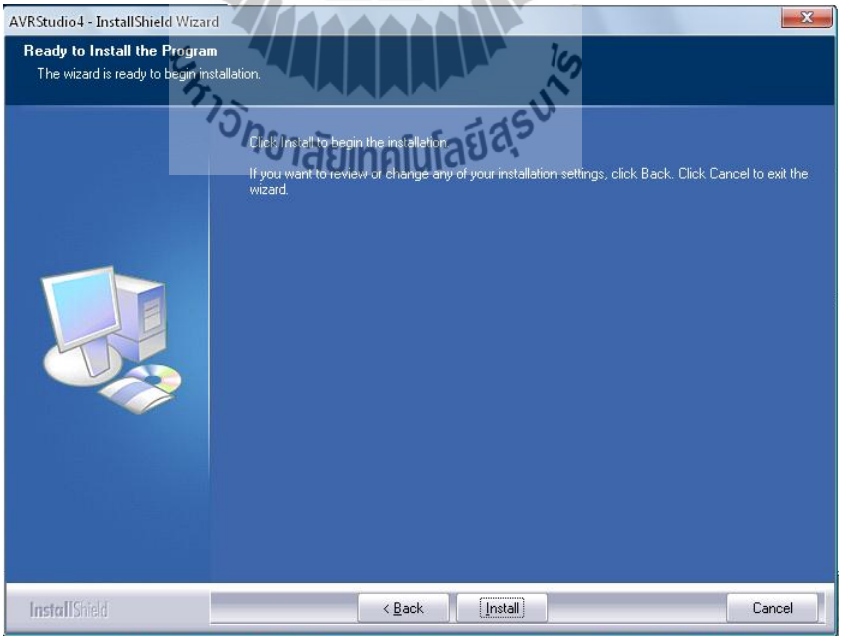
รูปที่ ก.12 หน้าต่าง Choose destination location

5. คลิก Next



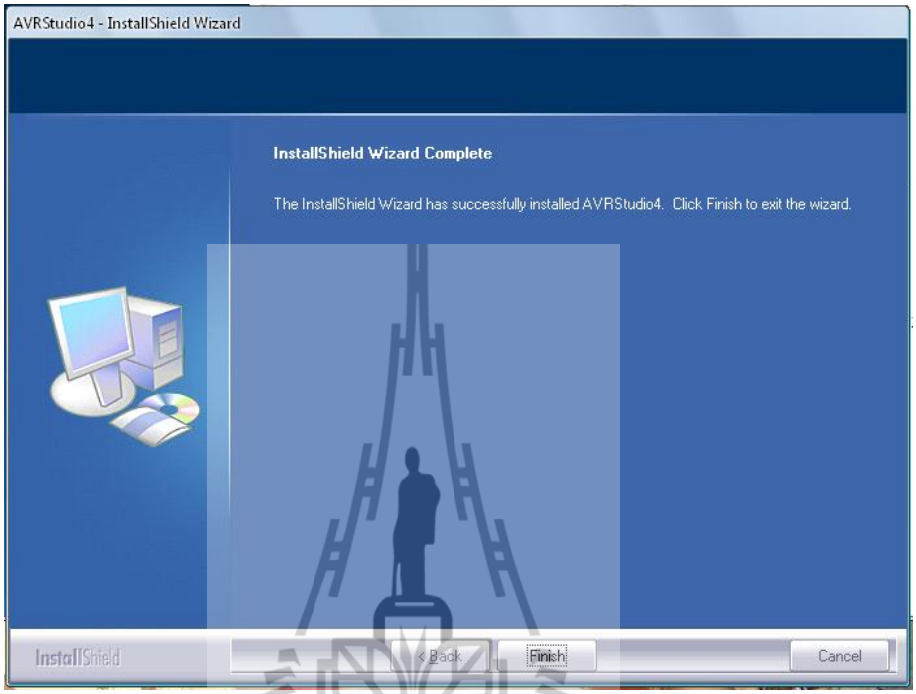
รูปที่ ก.13 หน้าต่าง Select features

6. คลิก Install



รูปที่ ก.14 หน้าต่าง Ready to install the program

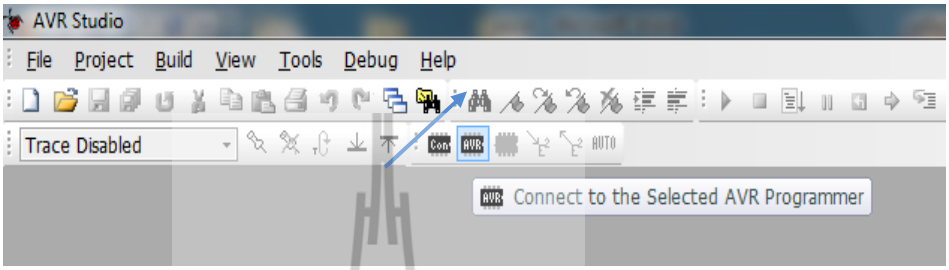
7. คลิก Finish เสร็จขั้นตอนการลงโปรแกรม AVR Studio 4



รูปที่ ก.15 หน้าต่างแสดงการเสร็จสิ้นการลงโปรแกรม AVR Studio 4

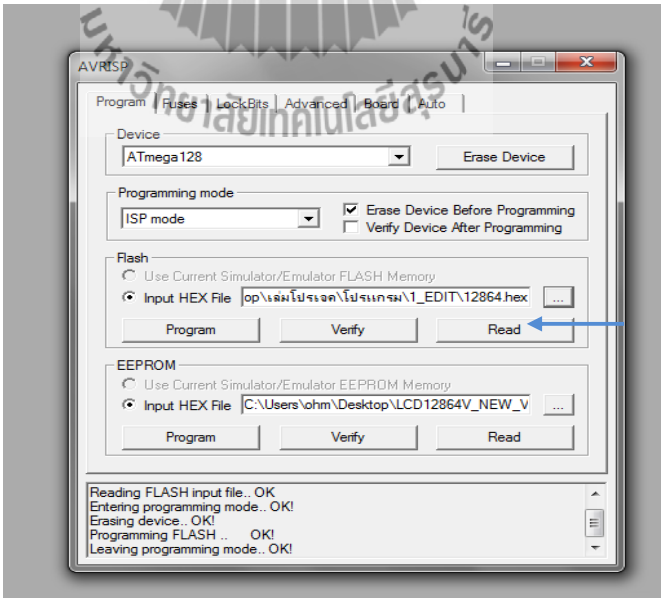
ขั้นตอนการโหลดไฟล์ HEX ลงบอร์ด

1. เปิดโปรแกรม AVR Studio 4 ขึ้นมา
2. คลิก Connect to the selected AVR programmer



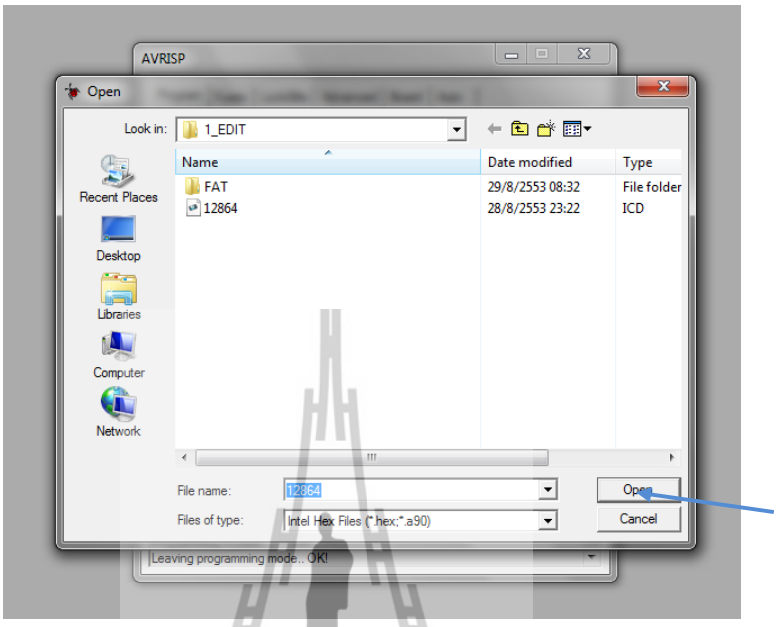
รูปที่ ก.16 การ Connect to the selected AVR programmer

3. แถว Platform เลือก STK500 or AVRISP แถว Port ให้เลือก Port ที่เราใช้ แล้วคลิก Connection
4. เลือกตามลูกศรชี้ เพื่อเปิดไฟล์ HEX



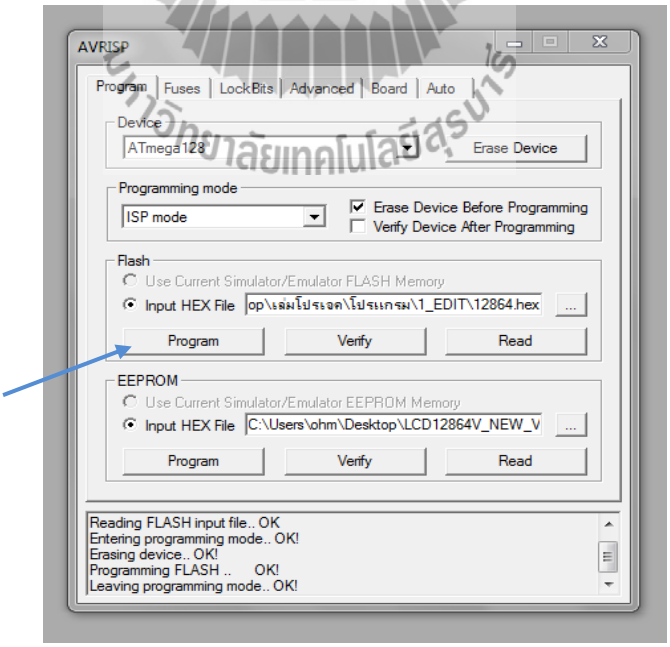
รูปที่ ก.17 วิธีการเปิดไฟล์ HEX

5. เลือกที่เก็บไฟล์ HEX



รูปที่ ก.18 แสดงการเลือกไฟล์ HEX

6. คลิก Program แล้วรอสักครู่ เสร็จการโหลดไฟล์ HEX



รูปที่ ก.19 คำสั่งในการโหลดไฟล์HEX

ภาคผนวก ข

โค้ดโปรแกรมหลัก

```
#include        <avr/io.h>

#include        <avr/interrupt.h>

#include        <stdio.h>

#include        <stdarg.h>

#include        <string.h>

#include        <stdlib.h>

#include        <math.h>

#include        "mydefs.h"

#include        "FAT/dos.h"

#endif

#include        "DS1307.h"

#include        "setup.h"

#include        "eeprom.h"

#include    "key.h"

#include        "lcd.h"

#include          "sht15.h"
```

```
char      lcdbuf[SIZEOF_LINE + 1];
uint8_t             flag,ct=0;
uint8_t  auto_ = 0;                         //mode
uint32_ttimmerOn;
double   volt,current;
ISR(INT4_vect)
{
          flag      |= DS1307_INT;          //Set Flag interrupt buffer after INT4 interrupt
}
void help (void)
{
          printf ("\r\n");
          printf ("\r\n");
}
void update (void)
{
          RTC.year  = 10;
```

```
        RTC.month = 8;
        RTC.date  = 13;
        RTC.hour = 0;
        RTC.min  = 27;
        update_RTC ();
}
int main(void)
{
        uint8_t key;
        DDRC  = 0x00;
        DDRD   = 0x00;
        DDRF   = 0x00;                          //all output
        _delay_ms (1);
        BUZ_DDR        |=  _BV(BUZZER);
        relay1_DDR     |=  _BV(relay1);
        relay1_off();
        LED_DDR  |= _BV(LED2) | _BV(LED3);
```

```
        LED_PORT &= ~_BV(LED2) | ~_BV(LED3);
        init_uart1 ();
        ser_init();                         // init uart
        Buzzer_on ();
        _delay_ms (100);
        Buzzer_off ();
        init_lcd ();                    // init lcd
#ifdef  RTC_1307
        PORT_INT4       |=  _BV(INT4);
        TWI_init ();
        EICRB  = _BV(ISC41) | (0 << ISC40);             enable_rtc ();
#endif
        Blacklight_off ();
        lcd_check_status();
        sei ();
        help ();
        beep (LONG);
```

```
        check_DS1307 (0);
#ifdef  RS232_DEBUG
        printf ("\r\nReady....");
        #endif
                _timer.flag_2=0;
                _timer.flag_1=0;
                _timer.flag_3=0;
                _timer.flag_4=0;
                _val.humid=100;
        _load_timmer();
        _delay_ms (1000);
                unsigned char i5=3,i6=0;
        while (1)
        {               if (flag & DS1307_INT)
                {
                        timmerOn =  display_rtc ();
                        flag  &= ~DS1307_INT;           //set flag RTC = 0;
```

```
                LED_PORT ^= _BV(LED3);
                ct = check_timmer ();
                if(ct){
                relay1_on();
                }
                else{relay1_off();}
                if(i5>2){
                i5=0;
                disable_rtc ();
                HumiditySHT11Func();
                enable_rtc ();
                }
                i5++;
        }
key = kbd_getc ();
        if (key != 0)
        {
```

```
#ifdef    KEY_DEBUG
                    printf ("\r\n%c",key);
#endif
                    switch (key)
                    {
                        case    CONFIG_DATE :       setup_date_time ();
                                i5=3;
                                                                    break;
                        case    CONFIG_Alarm1 :   setup_time_alarm1();
                        i5=3;                   break;
                        case    KEY_DOWN :   beep (LONG);
                                relay1_off();
                            while (kbd_getc () != 0)
                    _delay_ms (100);
                            break;
                        case    KEY_UP :            beep (LONG);
                        relay1_on();
                        while (kbd_getc () != 0);
```

```
                    _delay_ms (100);        break;
                case    KEY_show_Alarm1     : beep (LONG);
                while (kbd_getc () != 0);
            _show_timmer_alarm();  //set_alarm1();
            i5=3;               break;
                    case    KEY_ENT         : beep (LONG);
            while (kbd_getc () != 0);
            _delay_ms (100);
            if(_timer.flag_1==1){_timer.flag_1=0;alarm1.s_min=alarm1.s_min-2;}
            if(_timer.flag_2==1){_timer.flag_2=0;alarm2.s_min=alarm2.s_min-2;}
                    break;      default:
            break;
                }
            }
}while (1);
}
```

โค้ด SENSOR SHT15

```
#ifndef  sht_C_

#define  sht_C_

#include <avr/io.h>

#include <avr/pgmspace.h>

#include <inttypes.h>

#include <math.h>

#include "pgmspacehlp.h"

#include          "sht15.h"

#include          "mydefs.h"

#include          "lcd.h"

char CHAR2BCD2(signed char input)

{

   char high = 0;

   if (input < 0)

      input = ~(input) + 1;

   while (input >= 10)              // Count tens
```

```
    {
        high++;
        input -= 10;
    }
    return  (high << 4) | input;        // Add ones and return answer
}
unsigned int CHAR2BCD3(char input)
{
    int high = 0;
    while (input >= 100)                // Count hundreds
    {
        high++;
        input -= 100;
    }
    high <<= 4;
    while (input >= 10)                 // Count tens
    {
```

```
        high++;
        input -= 10;
    }
    return  (high << 4) | input;        // Add ones and return answer
}
char error;                             //global
unsigned char s_write_byte(unsigned char value)
{
    unsigned char i;
    error=0;
    MAKE_DATA_OUTPUT;                   //Make data pin an output
            _delay_us(1);
    for(i=0x80;i>0;i/=2)
    {
     if (i & value) SET_DATA;
     else CLEAR_DATA;
                SET_SCK;
```

```
        _delay_us(3);
    CLEAR_SCK;
   }
   SET_DATA;                    //Release the data line
   MAKE_DATA_INPUT;             //Make data pin an input
   SET_DATA;                    //Enable internal pull-up resistor
  _delay_us(1);
   SET_SCK;                     //Clock pulse #9 for ack
  _delay_us(3);
   error=DATA;                  //check ack (DATA will be pulled down by SHT11)
   CLEAR_SCK;
        return error;           //error=1 if no ACK from sensor
}
unsigned char s_read_byte(unsigned char ack)
{
   unsigned char i, val=0;
   MAKE_DATA_OUTPUT;            //Make data pin an output
```

```
_delay_us(1);
  SET_DATA;                    //Release the data line
  MAKE_DATA_INPUT;             //Make data pin an input
  SET_DATA;                    //Enable internal pull-up resistor
 _delay_us(1);
  for(i=0x80;i>0;i/=2)
  {
   SET_SCK;
        _delay_us(1);
   if (DATA) val=(val | i);
   CLEAR_SCK;
        _delay_us(1);
  }
  MAKE_DATA_OUTPUT;
        _delay_us(1);
  CLEAR_DATA;                  //In case 'ack==1' pull down DATA line
 _delay_us(2);
```

```
    SET_SCK;                    //Clock pulse #9 for ack
    _delay_us(3);
    CLEAR_SCK;
    _delay_us(2);
    SET_DATA;                   //Release the data line
    return val;
}
void s_transstart (void)
{
    MAKE_DATA_OUTPUT;
        _delay_us(1);
    SET_DATA;
    CLEAR_SCK;
        _delay_us(2);
    SET_SCK;;
    _delay_us(2);
    CLEAR_DATA;
```

```
    _delay_us(2);
    CLEAR_SCK;
    _delay_us(4);
    SET_SCK;
          _delay_us(2);
    SET_DATA;
          _delay_us(2);
    CLEAR_SCK;
}
void s_connectionreset (void)
{
    unsigned char i;
    MAKE_DATA_OUTPUT;
          _delay_us(1);
    SET_DATA;
    CLEAR_SCK;
    for(i=0;i<9;i++)
```

```
    {
     SET_SCK;
            _delay_us(2);
     CLEAR_SCK;
            _delay_us(2);
    }
    s_transstart();
}
unsigned char s_softreset(void)
{
    error=0;
    s_connectionreset();              //communication reset
    error = s_write_byte(RESET);      //send reset command to sensor
    return error;                     //error if no response from sensor
}
unsigned char s_read_statusreg(unsigned char *p_value, unsigned char *p_checksum)
{
```

```
    error=0;
    s_transstart();                  //transmission start
    error=s_write_byte(STATUS_REG_R);  //send command to sensor
    *p_value=s_read_byte(ACK);         //read status register (8-bit)
    *p_checksum=s_read_byte(noACK);    //read checksum (8-bit)
    return error;                    //error if no response from sensor
}
unsigned char s_write_statusreg(unsigned char *p_value)
{
    error=0;
    s_transstart();                  //transmission start
    error+=s_write_byte(STATUS_REG_W); //send command to sensor
    error+=s_write_byte(*p_value);     //send command to sensor
    return error;                    //error if no response from sensor
}
unsigned char s_measure(unsigned char *p_value, unsigned char *p_checksum, unsigned char mode)
{
```

```
unsigned int i;
error=0;
s_transstart();                  //transmission start
switch (mode)
{
  case TEMP : error += s_write_byte(MEASURE_TEMP); break;
  case HUMID: error += s_write_byte(MEASURE_HUMI); break;
  default   : break;
}
MAKE_DATA_INPUT;                 //Make data pin an input
SET_DATA;                        //Enable internal pull-up resistor
i=0;
while((DATA) && (i<65534))
{
        _delay_us(30);
 i++;
}
```

```
    if (i>65533) error+=1;             //timeout after ~2s
    *(p_value+1) = s_read_byte(ACK);   //read MSB
    *(p_value)   = s_read_byte(ACK);   //read LSB
    *p_checksum  = s_read_byte(noACK); //read checksum
    return error;
}
void calc_sht11(float *p_humidity, float *p_temperature)
{
    const float C1=-4.0;               //for 12 bit
    const float C2= 0.0405;            //for 12 bit
    const float C3=-0.0000028;         //for 12 bit
    const float T1= 0.01;              //for 14 bit at 5V
    const float T2= 0.00008;           //for 14 bit at 5V
    float rh=*p_humidity;              // rh:      humidity [Ticks] 12 bits
    float t =*p_temperature;           // t:       temperature [Ticks] 14 bits
    float rh_lin;                     // rh_lin:  humidity linear
    float rh_true;                     // rh_true: temperature compensated humidity
```

```
    float t_C;                  // t_C:     temperature degrees C
    t_C=(t*0.01) - 39.6;        // calc degrees C from ticks @ 3V=VDD
    rh_lin=C3*rh*rh + C2*rh + C1;  // calc humidity from ticks
    rh_true=(t_C-25)*(T1+T2*rh)+rh_lin;// calc temperature compensated humidity
    if(rh_true>100)rh_true=100;       // upper display bounds limit
    if(rh_true<0.1)rh_true=0.1;       // lower display bounds limit
    *p_temperature=t_C;               //return temperature
    *p_humidity=rh_true;              //return humidity
}
float calc_dewpoint(float h, float t)
{
  float logEx, dew_point;
  logEx=0.66077+7.5*t/(237.3+t)+(log10(h)-2);
  dew_point = (logEx - 0.66077)*237.3/(0.66077+7.5-logEx);
  return dew_point;
}
void HumiditySHT11Func(void)
```

```
{    uint8_t lcdbuf[SIZEOFLCD + 1];
   value humid_val,temp_val;
   unsigned char error,checksum;
   char T_TL,T_TH,HumidBCD,TempBCD,H_TL,H_TH;
   MAKE_SCK_OUTPUT;                    //make SCK line output
   s_connectionreset();
          _delay_ms(20);
   printf ("\r\n-----");
   error=0;
   error+=s_measure((unsigned char*) &humid_val.i,&checksum,HUMID);
          _delay_ms(20);
   error+=s_measure((unsigned char*) &temp_val.i,&checksum,TEMP);
   if(error != 0)
   { _delay_ms(12);
     s_connectionreset();
_delay_ms(12);
lcd_clear (3);              //for sim
```

```
          lcd_gotoxy (LINE3,2);
                    lcd_printf ("\%");
          lcd_gotoxy (LINE3,3);
         sprintf (lcdbuf,"RH = --.-- ");
         lcd_printf (lcdbuf);
           lcd_clear (4);              //for sim
         lcd_gotoxy (LINE4,3);
         sprintf (lcdbuf,"Temp = --.-- 'C");
         lcd_printf (lcdbuf);
          printf ("\r\n999");
          _delay_ms(50);
}
else
{
   humid_val.f=(float)humid_val.i;
   temp_val.f=(float)temp_val.i;
   calc_sht11(&humid_val.f,&temp_val.f);
```

```
        _val.humid=humid_val.f;
                lcd_gotoxy (LINE3,3);
               lcd_printf ("\%");
       lcd_gotoxy (LINE3,4);
      sprintf (lcdbuf,"RH = %.2f ",humid_val.f);
      lcd_printf (lcdbuf);
      lcd_gotoxy (LINE4,2);
      sprintf (lcdbuf,"Temp = %.2f 'C",temp_val.f);
      lcd_printf (lcdbuf);
humid_val.i=(unsigned int)humid_val.f;
HumidBCD = CHAR2BCD2(humid_val.i);
H_TL = (HumidBCD & 0x0F) + '0';
H_TH = ((HumidBCD >> 4) & 0x0F) + '0';
temp_val.i=(unsigned int)temp_val.f;
TempBCD = CHAR2BCD2(temp_val.i);
T_TL = (TempBCD & 0x0F) + '0';
T_TH = ((TempBCD >> 4) & 0x0F) + '0';
```

```
            printf ("\r\n%c,%c",H_TH,H_TL);
            lcd_clear (3);              //for sim
            lcd_gotoxy (LINE3,1);
                    lcd_printf ("\%");
           lcd_gotoxy (LINE3,2);
          sprintf (lcdbuf,"RH = %c%c ",H_TH,H_TL);
          lcd_printf (lcdbuf);
          lcd_gotoxy (LINE3,9);
          lcd_printf ("\%");
          lcd_clear (4);              //for sim
          lcd_gotoxy (LINE4,2);
          sprintf (lcdbuf,"Temp = %c%c 'C",T_TH,T_TL);
          lcd_printf (lcdbuf);*/
           _delay_ms(50);
    }
}
#endif
```

ภาคผนวก ค

DATASHEET SHT15

SHT1x

Humidity & Temperature Sensor

. Relative humidity and temperature sensors

. Dew point

. Fully calibrated, digital output

. Excellent long.term stability

. No external components required

. Ultra low power consumption

. Surface mountable or 4.pin fully interchangeable

. Small size

. Automatic power down

SHT1x Product Summary

The SHTxx is a single chip relative humidity and temperature multi sensor module comprising a calibrated digital output. Application of industrial CMOS processes with patented micro.machining (CMOSens® technology) ensures highest reliability and excellent long term stability. The device includes a capacitive polymer sensing element for relative humidity and a bandgap temperature sensor. Both are seamlessly coupled to a 14bit analog to digital converter and a serial interface circuit on the same chip. This results in superior signal quality, a fast response time and insensitivity to external disturbances (EMC) at a very competitive price. Each SHTxx is individually calibrated in a precision humidity chamber with a chilled mirror hygrometer as reference. The calibration coefficients are programmed into the OTP

memory. These coefficients are used internally during measurements to calibrate the signals from the sensors.
The 2.wire serial interface and internal voltage regulation allows easy and fast system integration. Its tiny size and low power consumption makes it the ultimate choice for even the most demanding applications.
The device is supplied in either a surface.mountable LCC (Leadless Chip Carrier) or as a pluggable 4.pin single.in.line type package. Customer specific packaging options may be available on request.

Applications

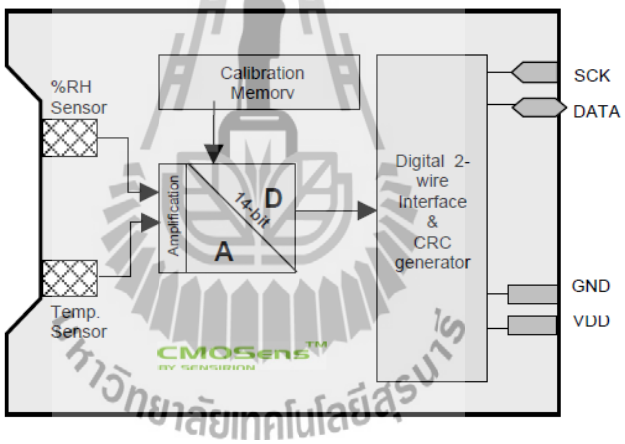
_ HVAC
_ Automotive
_ Consumer Goods
_ Weather Stations
_ Humidifiers
_ Dehumidifiers
_ Test & Measurement
_ Data Logging
_ Automation
_ White Goods
_ Medical

Ordering Information

Part Number	Humidity accuracy [%RH]	Temperature accuracy [K] @ 25 °C	Package
SHT11	±3.0	±0.4	SMD (LCC)
SHT15	±2.0	±0.3	SMD (LCC)
SHT71	±3.0	±0.4	4-pin single-in-line
SHT75	±1.8	±0.3	4-pin single-in-line

รูปที่ ค .1 Ordering Information

Block Diagram

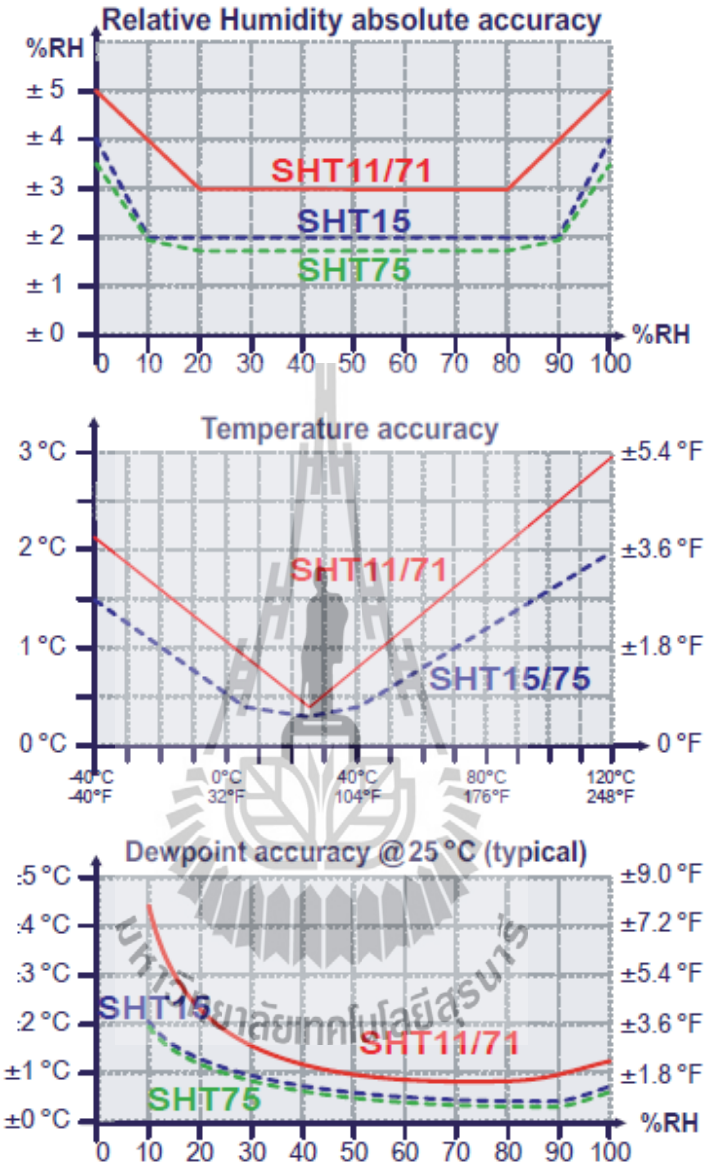


รูปที่ ค.2 Block Diagram

1 Sensor Performance Specifications

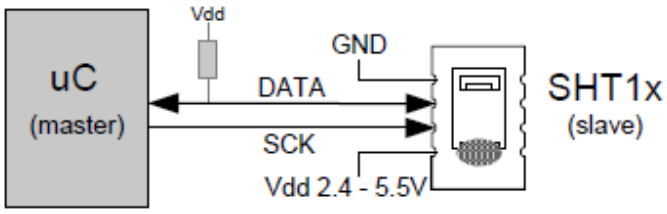
Parameter	Conditions	Min.	Typ.	Max.	Units
Humidity					
Resolution (2)		0.5	0.03	0.03	%RH
		8	12	12	bit
Repeatability			±0.1		%RH
Accuracy (1) Uncertainty	linearized	see figure 1			
Interchangeability		Fully interchangeable			
Nonlinearity	raw data		±3		%RH
	linearized		<<1		%RH
Range		0		100	%RH
Response time	1/e (63%) slowly moving air		4		s
Hysteresis			±1		%RH
Long term stability	typical		< 0.5		%RH/yr
Temperature					
Resolution (2)		0.04	0.01	0.01	°C
		0.07	0.02	0.02	°F
		12	14	14	bit
Repeatability			±0.1		°C
			±0.2		°F
Accuracy	see figure 1				
Range		-40		123.8	°C
		-40		254.9	°F
Response Time	1/e (63%)	5		30	s

รูปที่ ค.3 Table 1 Sensor Performance Specifications



รูปที่ ค.4 Rel. Humidity, Temperature and Dewpoint accuracies

2 Interface Specifications



รูปที่ ค.5 Typical application circuit

2.1 Power Pins

The SHTxx requires a voltage supply between 2.4 and 5.5 V. After powerup the device needs 11ms to reach its "sleep" state. No commands should be sent before that time. Power supply pins (VDD, GND) may be decoupled with a 100 nF capacitor.

2.2 Serial Interface (Bidirectional 2.wire)

The serial interface of the SHTxx is optimized for sensor readout and power consumption and is not compatible with I2C interfaces, see FAQ for details.

2.2.1 Serial clock input (SCK)

The SCK is used to synchronize the communication between a microcontroller and the SHTxx. Since the interface consists of fully static logic there is no minimum SCK frequency.

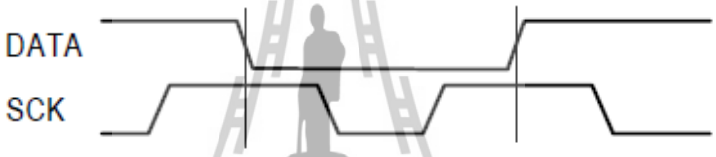
2.2.2 Serial data (DATA)

The DATA tristate pin is used to transfer data in and out of the device. DATA changes after the falling edge and is valid on the rising edge of the serial clock SCK. During transmission the DATA line must remain stable while SCK is high. To avoid signal contention the microcontroller should

only drive DATA low. An external pull.up resistor (e.g. 10 kΩ)
is required to pull the signal high. (See Figure 2) Pull.up
resistors are often included in I/O circuits of microcontrollers.
See Table 5 for detailed IO characteristics.

2.2.3 Sending a command

To initiate a transmission, a “Transmission Start” sequence
has to be issued. It consists of a lowering of the DATA line
while SCK is high, followed by a low pulse on SCK and
raising DATA again while SCK is still high.



รูปที่ ค.6 "Transmission Start" sequence

The subsequent command consists of three address bits
(only “000” is currently supported) and five command bits.
The SHTxx indicates the proper reception of a command by
pulling the DATA pin low (ACK bit) after the falling edge of
the 8th SCK clock. The DATA line is released (and goes
high) after the falling edge of the 9th SCK clock.

Command	Code
Reserved	0000x
Measure Temperature	**00011**
Measure Humidity	**00101**
Read Status Register	00111
Write Status Register	00110
Reserved	0101x-1110x
Soft reset, resets the interface, clears the status register to default values wait minimum 11 ms before next command	**11110**

รูปที่ ค.7 SHTxx list of commands

2.2.4 Measurement sequence (RH and T)

After issuing a measurement command ('00000101' for RH, '00000011' for Temperature) the controller has to wait for the measurement to complete. This takes approximately 11/55/210 ms for a 8/12/14bit measurement. The exact time varies by up to ±15% with the speed of the internal oscillator. To signal the completion of a measurement, the SHTxx pulls down the data line and enters idle mode. The controller must wait for this "data ready" signal before restarting SCK to readout the data. Measurement data is stored until readout, therefore the controller can continue with other tasks and readout as convenient.

Two bytes of measurement data and one byte of CRC checksum will then be transmitted. The uC must acknowledge each byte by pulling the DATA line low. All values are MSB first, right justified. (e.g. the 5th SCK is MSB for a 12bit value, for a 8bit result the first byte is not used). Communication terminates after the acknowledge bit of the CRC data. If CRC.8 checksum is not used the controller may terminate the communication after the measurement data LSB by keeping ack high.

The device automatically returns to sleep mode after the measurement and communication have ended.

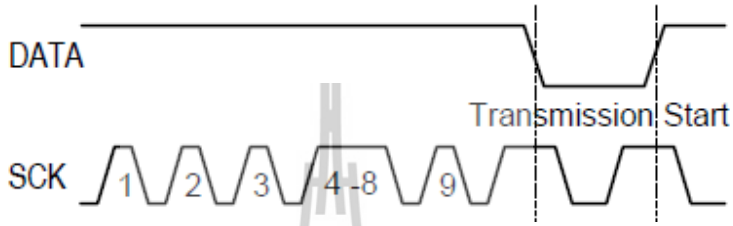
Warning: To keep self heating below 0.1 °C the SHTxx should not be active for more than 10% of the time (e.g. max. 2 measurements / second for 12bit accuracy).

2.2.5 Connection reset sequence

If communication with the device is lost the following signal

sequence will reset its serial interface:

While leaving DATA high, toggle SCK 9 or more times. This must be followed by a “Transmission Start” sequence preceding the next command. This sequence resets the interface only. The status register preserves its content.

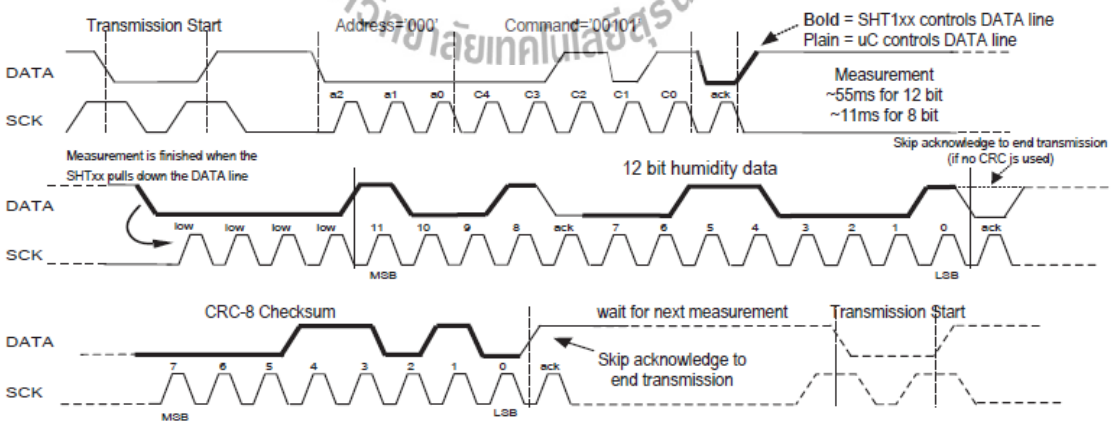


รูปที่ ค.8 Connection reset sequence

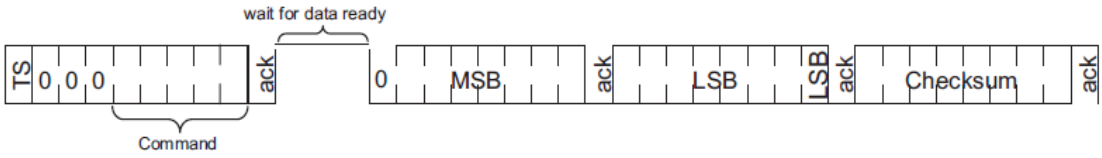
2.2.6 CRC.8 Checksum calculation

The whole digital transmission is secured by a 8 bit checksum. It ensures that any wrong data can be detected and eliminated.

Please consult application note “CRC.8 Checksum Calculation” for information on how to calculate the CRC.



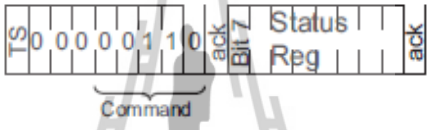
รูปที่ ค.9 Example RH measurement sequence for value “0000’1001 ’ 0011’0001”= 2353 = 75.79 %RH (without temperature compensation)



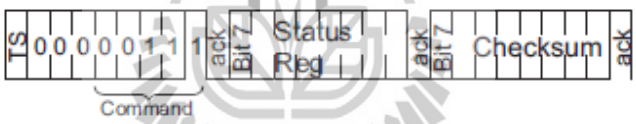
รูปที่ ค.10 Overview of Measurement Sequence (TS = Transmission Start)

2.3 Status Register

Some of the advanced functions of the SHTxx are available through the status register. The following section gives a brief overview of these features. A more detailed description is available in the application note “Status Register”



รูปที่ ค.11 Status Register Write



รูปที่ ค.12 Status Register Read

Bit	Type	Description	Default	
7		reserved	0	
6	R	End of Battery (low voltage detection) ‘0’ for Vdd > 2.47 ‘1’ for Vdd < 2.47	X	No default value, bit is only updated after a measurement
5		reserved	0	
4		reserved	0	
3		For Testing only, do not use	0	
2	R/W	Heater	0	off
1	R/W	no reload from OTP	0	reload
0	R/W	‘1’ = 8bit RH / 12bit Temperature resolution ‘0’ = 12bit RH / 14bit Temperature resolution	0	12bit RH 14bit Temp.

รูปที่ ค.13 Status Register Bits

2.3.1 Measurement resolution

The default measurement resolution of 14bit (temperature) and 12bit (humidity) can be reduced to 12 and 8bit. This is especially useful in high speed or extreme low power applications.

2.3.2 End of Battery

The “End of Battery” function detects VDD voltages below 2.47 V. Accuracy is ฑ0.05 V

2.3.3 Heater

An on chip heating element can be switched on. It will increase the temperature of the sensor by 5.15 °C (9.27 °F). Power consumption will increase by ~8 mA @ 5 V.

Applications:

By comparing temperature and humidity values before and after switching on the heater, proper functionality of both sensors can be verified.

• In high (>95 %RH) RH environments heating the sensor element will prevent condensation, improve response time and accuracy

Warning: While heated the SHTxx will show higher temperatures and a lower relative humidity than with no heating.

2.4 Electrical Characteristics(1)

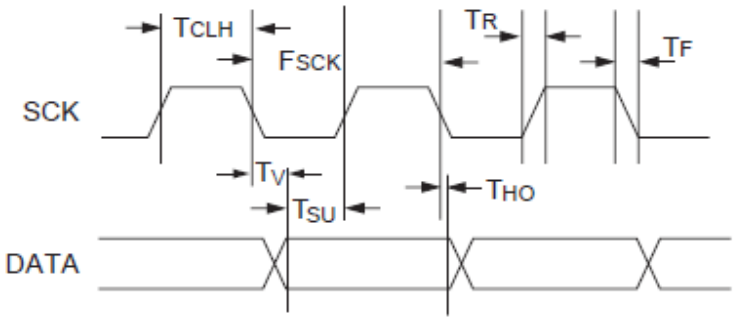
VDD=5V, Temperature = 25 °C unless otherwise noted

Parameter	Conditions	Min.	Typ.	Max.	Units
Power supply DC		2.4	5	5.5	V
Supply current	measuring		550		µA
	average	2[2]	28[3]		µA
	sleep		0.3	1	µA
Low level output voltage		0		20%	Vdd
High level output voltage		75%		100%	Vdd
Low level input voltage	Negative going	0		20%	Vdd
High level input voltage	Positive going	80%		100%	Vdd
Input current on pads				1	µA
Output peak current	on			4	mA
	Tristated (off)		10		µA

รูปที่ ค.14 SHTxx DC Characteristics

	Parameter	Conditions	Min	Typ.	Max.	Unit
F_{SCK}	SCK frequency	VDD > 4.5 V			10	MHz
		VDD < 4.5 V			1	MHz
T_{RFO}	DATA fall time	Output load 5 pF	3.5	10	20	ns
		Output load 100 pF	30	40	200	ns
T_{CLx}	SCK hi/low time		100			ns
T_V	DATA valid time			250		ns
T_{SU}	DATA set up time		100			ns
T_{HO}	DATA hold time		0	10		ns
T_R/T_F	SCK rise/fall time			200		ns

รูปที่ ค.15 SHTxx I/O Signals Characteristics



รูปที่ ค.16 Timing Diagram

3 Converting Output to Physical Values

3.1 Relative Humidity

To compensate for the non.linearity of the humidity sensor
and to obtain the full accuracy it is recommended to convert
the readout with the following formula1:

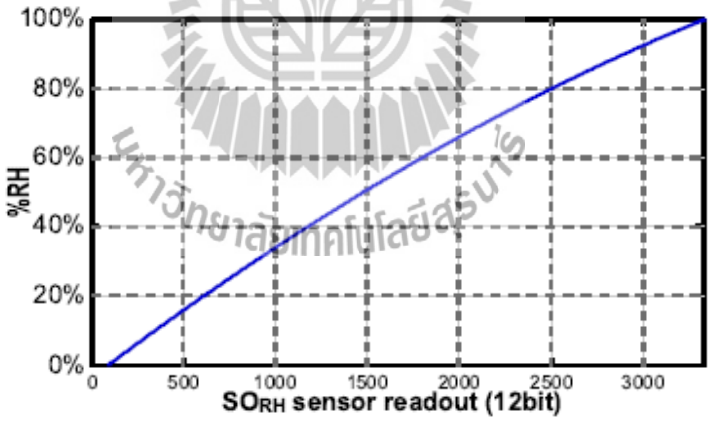
2RHlinear = c1 +c2 •SORH +c3 •SORH

SO_{RH}	c_1	c_2	c_3
12 bit	-4	0.0405	$-2.8 * 10^{-6}$
8 bit	-4	0.648	$-7.2 * 10^{-4}$

รูปที่ ค.17 Humidity conversion coefficients

For simplified, less computation intense conversion formulas
see application note “RH and Temperature Non.Linearity
Compensation”.

The humidity sensor has no significant voltage dependency.



รูปที่ ค.18 Conversion from SORH to relative humidity

3.1.1 Humidity Sensor RH/Temperature compensation

For temperatures significantly different from 25 °C (~77 °F)
the temperature coefficient of the RH sensor should be
considered:

RHtrue = (T฀C . 25)•(t1 +t2 •SORH)+RHlinear

SO_{RH}	t_1	t_2
12 bit	0.01	0.00008
8 bit	0.01	0.00128

รูปที่ ค.19 Temperature compensation coefficients

This equals ~0.12 %RH / °C @ 50 %RH

3.2 Temperature

The bandgap PTAT (Proportional To Absolute Temperature) temperature sensor is very linear by design. Use the following formula to convert from digital readout to temperature:

Temperatur e = d1 +d2 •SOT

VDD	d_1 [°C]	d_1 [°F]
5V	-40.00	-40.00
4V	-39.75	-39.50
3.5V	-39.66	-39.35
3V	-39.60	-39.28
2.5V	-39.55	-39.23

SO_T	d_2 [°C]	d_2 [°F]
14bit	0.01	0.018
12bit	0.04	0.072

รูปที่ ค.20 Temperature conversion coefficients

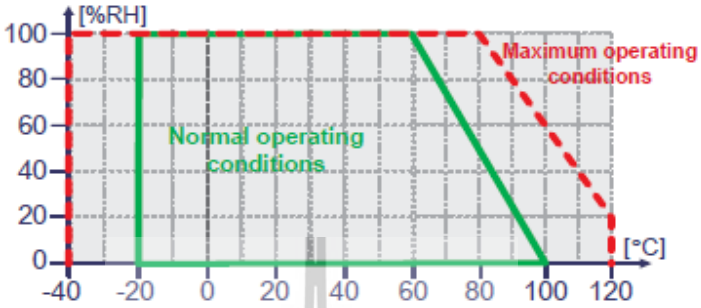
For improved accuracies in extreme temperatures with more computation intense conversion formulas see application note “RH and Temperature Non.Linearity Compensation”.

3.3 Dewpoint

Since humidity and temperature are both measured on the same monolithic chip, the SHTxx allows superb dewpoint measurements. See application note “Dewpoint calculation” for more.

4 Applications Information

4.1 Operating and Storage Conditions



รูปที่ ค.21 Recommended operating conditions

Conditions outside the recommended range may temporarily offset the RH signal up to ฑ3 %RH. After return to normal conditions it will slowly return towards calibration state by itself. See 4.3 “Reconditioning Procedure” to accelerate this process. Prolonged exposure to extreme conditions may accelerate ageing.

4.2 Exposure to Chemicals

Chemical vapors may interfere with the polymer layers used for capacitive humidity sensors. The diffusion of chemicals into the polymer may cause a shift in both offset and sensitivity. In a clean environment the contaminants will slowly outgas. The reconditioning procedure described below will accelerate this process. High levels of pollutants may cause permanent damage to the sensing polymer.

4.3 Reconditioning Procedure

The following reconditioning procedure will bring the sensor back to calibration state after exposure to extreme conditions or chemical vapors.

80.90 °C (176.194°F) at < 5 %RH for 24h (baking) followed by

20.30 °C (70.90°F) at > 74 %RH for 48h (re.hydration)

4.4 Temperature Effects

The relative humidity of a gas strongly depends on its temperature. It is therefore essential to keep humidity sensors at the same temperature as the air of which the relative humidity is to be measured.

If the SHTxx shares a PCB with electronic components that give off heat it should be mounted far away and below the heat source and the housing must remain well ventilated.

To reduce heat conduction copper layers between the SHT1x and the rest of the PCB should be minimized and a slit may be milled in between (see figure 13).

4.5 Membranes

A membrane may be used to prevent dirt from entering the housing and to protect the sensor. It will also reduce peak concentrations of chemical vapors. For optimal response times air volume behind the membrane must be kept to a minimum. For the SHT1x package Sensirion recommends the SF1 filter cap for optimal IP67 protection.

4.6 Light

The SHTxx is not light sensitive. Prolonged direct exposure to sunshine or strong UV radiation may age the housing.

4.7 Materials Used for Sealing / Mounting

Many materials absorb humidity and will act as a buffer, increasing response times and hysteresis. Materials in the vicinity of the sensor must therefore be carefully chosen.

Recommended materials are: All Metals, LCP, POM (Delrin), PTFE (Teflon), PE, PEEK, PP, PB, PPS, PSU, PVDF, PVF

For sealing and gluing (use sparingly): High filled epoxy for electronic packaging (e.g. glob top, underfill), and Silicone. Outgassing of these materials may also contaminate the SHTxx (cf. 4.2). Store well ventilated after manufacturing or bake at 50°C for 24h to outgas contaminants before packing.

4.8 Wiring Considerations and Signal Integrity

Carrying the SCK and DATA signal parallel and in close proximity (e.g. in wires) for more than 10cm may result in cross talk and loss of communication. This may be resolved by routing VDD and/or GND between the two data signals. Please see the application note "ESD, Latchup and EMC" for more information.

Power supply pins (VDD, GND) should be decoupled with a 100 nF capacitor if wires are used.

4.9 Qualifications

Extensive tests were performed in various environments. Please contact SENSIRION for detailed information.

Environment	Norm	Results(1)
Temperature Cycles	JESD22-A104-B -40 °C / 125 °C, 1000 cy	Within Specifications
HAST Pressure Cooker	JESD22-A110-B 2.3 bar 125 °C 85 %RH	Reversible shift by +2 %RH
High Temperature and Humidity	JESD22-A101-B 85 °C 85 %RH 1250h	Reversible shift by +2 %RH
Salt Atmosphere	DIN-50021ss	Within Spec.
Condensing Air	-	Within Spec.
Freezing cycles fully submerged	-20 / +90 °C, 100 cy 30min dwell time	Reversible shift by +2 %RH
Various Automotive Chemicals	DIN 72300-5	Within Specifications

รูปที่ ค.22 Qualification tests (excerpt)

4.10 ESD (Electrostatic Discharge)

ESD immunity is qualified according to MIL STD 883E, method 3015 (Human Body Model at ฑ2 kV)).

Latch.up immunity is provided at a force current of ฑ100 mA with Tamb = 80 °C according to JEDEC 17. See application note “ESD, Latchup and EMC” for more information.

5 Package Information

5.1 SHT1x (surface mountable)

Pin	Name	Comment
1	GND	Ground
2	DATA	Serial data, bidirectional
3	SCK	Serial clock, input
4	VDD	Supply 2.4 - 5.5 V
	NC	Remaining pins must be left unconnected

รูปที่ ค.23 SHT1x Pin Description

5.1.1 Package type

The SHT1x is supplied in a surface.mountable LCC (Leadless Chip Carrier) type package. The sensors housing consists of a Liquid Crystal Polymer (LCP) cap with epoxy glob top on a standard 0.8 mm FR4 substrate. The device is free of lead, Cd and Hg.

Device size is 7.42 x 4.88 x 2.5 mm (0.29 x 0.19 x 0.1 inch)

Weight 100 mg

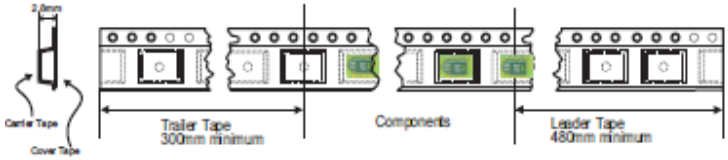
The production date is printed onto the cap in white numbers in the form wwy. e.g. ”351” = week 35, 2001.

5.1.2 Delivery Conditions

The SHT1x are shipped in 12mm tape at 100pcs or 400pcs..

Reels are individually labelled with barcode and human

readable labels. The Lot numbers allow full traceability through production, calibration and test.



รูปที่ ค.24 Tape configuration and unit orientation

5.1.3 Soldering Information

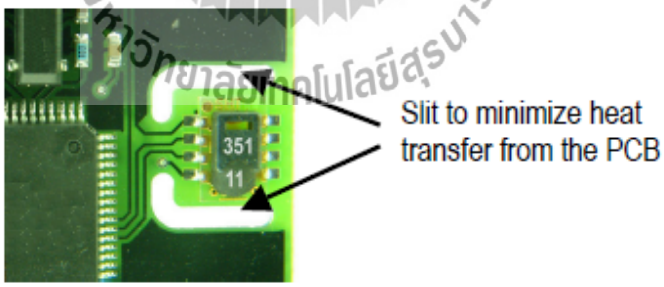
Standard reflow soldering ovens may be used. For details please see application note "soldering procedure".

For manual soldering contact time must be limited to 5 seconds at up to 350 °C.

After soldering the devices should be stored at >74 %RH for at least 24h to allow the polymer to rehydrate.

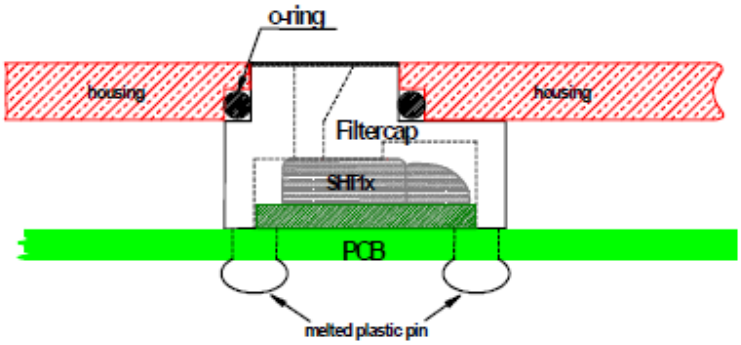
Please consult the application note “Soldering procedure” for more information.

5.1.4 Mounting Examples

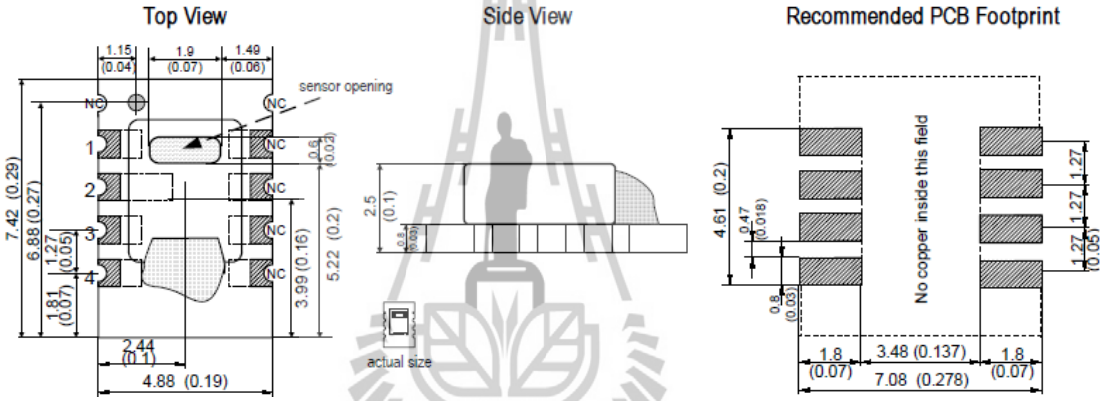


รูปที่ ค.25 SHT1x PCB Mounting example

The SF1 membrane filter cap is available for optimal IP67 protection. When mounted through a housing the interior can be protected from the environment while still allowing high quality humidity measurements (see example below).



รูปที่ ค.26 SF1 IP67 filter cap mounting example



รูปที่ ค.27 SHT1x drawing and footprint dimensions in mm (inch)

ประวัติผู้เขียน

นายไพสิฐ พูลเพิ่ม เกิดเมื่อวันที่ 2 กุมภาพันธ์ พ.ศ. 2532 ภูมิลำเนาอยู่ที่ ตำบลประตูชัย อำเภอพระนครศรีอยุธยา จังหวัดพระนครศรีอยุธยา สำเร็จการศึกษาระดับมัธยมศึกษาตอนปลายจากโรงเรียนอยุธยาวิทยาลัย อำเภอพระนครศรีอยุธยา จังหวัดพระนครศรีอยุธยา ปัจจุบันเป็นนักศึกษาชั้นปีที่ 4 สาขาวิชาวิศวกรรมโทรคมนาคม สำนักวิชาวิศวกรรมศาสตร์ มหาวิทยาลัยเทคโนโลยีสุรนารี

นางสาวสุปรียา มะโนมั่น เกิดเมื่อวันที่ 30 มกราคม พ.ศ.2532 ภูมิลำเนาอยู่ที่ ตำบลบัวใหญ่ อำเภอบัวใหญ่ จังหวัดนครราชสีมา สำเร็จการศึกษาระดับมัธยมศึกษาตอนปลายจากโรงเรียนวัดประชานิมิตร อำเภอบัวใหญ่ จังหวัดนครราชสีมา ปัจจุบันเป็นนักศึกษาชั้นปีที่ 4 สาขาวิชาวิศวกรรมโทรคมนาคม สำนักวิชาวิศวกรรมศาสตร์ มหาวิทยาลัยเทคโนโลยีสุรนารี