การพัฒนาระบบควบคุมค่าความเป็นกรด-ด่างแบบอัตโนมัติ ในอุตสาหกรรมการเลี้ยงกุ้ง

โดย

นายวรุตม์ บุญเลี่ยม
นายพรประสิทธิ์ บุญทอง
นายอาทิตย์ อยู่เย็น

สนับสนุนงบประมาณโดย
มหาวิทยาลัยเทคโนโลยีราชมงคลรัตนโกสินทร์
ประจำปีงบประมาณ 2558

Development control acidity - alkalinity automatically shrimp industry

By
Mr. WAROOT BOONLIAM
Mr. PORNPRASIT BOONTONG
Mr. ARTHIT YOOYEN

Granted by
Rajamangala University of Technology Rattanakosin
Fiscal year 2015

กิตติกรรมประกาศ

งานวิจัยเรื่องการพัฒนาระบบควบคุมค่าความเป็นกรด-ด่างแบบอัตโนมัติในอุตสาหกรรมการเลี้ยงกุ้งนี้สำเร็จลุล่วงได้ ทั้งนี้ได้รับความช่วยเหลือจาก นายเฉลิมพงษ์ อังคาระอาพันธ์ ผู้ประกอบการบ่อเลี้ยงกุ้งซึ่งเป็นที่ปรึกษาของงานวิจัย ที่ได้เสียสละเวลาให้คำปรึกษา และแนะนำวิธีต่างๆ เพื่อให้โครงการสำเร็จลุล่วงไปได้ด้วยดี

สุดท้ายนี้ขอขอบพระคุณ มหาวิทยาลัยเทคโนโลยีราชมงคลรัตนโกสินทร์ วิทยาเขตวังไกลกังวล ที่สนับสนุนงบประมาณในการทำวิจัยจนเสร็จสมบูรณ์ได้ในครั้งนี้

นายวรุตม์ บุญเลี่ยม
นายพรประสิทธิ์ บุญทอง
นายอาทิตย์ อยู่เย็น
สิงหาคม 2559

บทคัดย่อ

รหัสโครงการ : Inno 006/2558
ชื่อโครงการ : การพัฒนาระบบควบคุมค่าความเป็นกรด-ด่างแบบอัตโนมัติในอุตสาหกรรมการเลี้ยงกุ้ง
ชื่อนักวิจัย : นายวรุตม์ บุญเลี่ยม, นายพรประสิทธิ์ บุญทอง, นายอาทิตย์ อยู่เย็น

การพัฒนาระบบควบคุมค่าความเป็นกรด-ด่างแบบอัตโนมัติในอุตสาหกรรมการเลี้ยงกุ้ง เป็นการพัฒนาควบคุมค่าความเป็นกรด-ด่าง เพื่อช่วยลดการเกิดโรคหรือการตายของกุ้ง ระบบควบคุมค่าความเป็นกรด-ด่าง มีส่วนประกอบ 4 ส่วนคือ 1) ชุด Senser วัดค่าความเป็นกรด-ด่าง 2) ไมโครคอลโทรลเลอร์ (Arduino) 3) ชุดจอภาพ 4) ชุดควบคุมสารเพิ่ม ลดค่าความเป็นกรด-ด่าง โดยระบบควบคุมค่าความเป็นกรด-ด่างในบ่อเลี้ยงกุ้งนั้นจะถูกติดตั้งไว้ที่บ่อเลี้ยงกุ้งจำลองตลอดเวลา เพื่อจะทำการวัดค่าความเป็นกรด-ด่างและทำการปรับเปลี่ยนค่าความเป็นกรด-ด่างของน้ำให้คงสภาพตามปรกติ

ผลการทดลองพบว่า เมื่อนำชุดการทดลองไปทดสอบพบว่าในการทดสอบ ร้อยละ 100 สามารถทำงานได้ตามปรกติสามารถปรับสภาพได้ให้คงที่ได้ และส่วนใหญ่ชุดควบคุมระบบค่าความเป็นกรด-ด่างสามารถทำงานได้ปรกติ

E-mail Address : waroot.boo@rmutr.ac.th
ระยะเวลาโครงการ : ตุลาคม 2557 – สิงหาคม 2558

Abstract

Code of project : Inno 006/2015

Project name : Development control acidity - alkalinity automatically shrimp industry.

Researcher name : Waroot Boonliam, Pornprasit Boontong, Arthit Yooyen

Development of automatic pH control system in shrimp farming industry Is the development of control of pH. Reduce the occurrence of disease or death of shrimp. The pH control system has 4 components: 1) PH senser 2) Microcontroller (Arduino) 3) Display unit 4) Additive Control PH. The pH control system in the shrimp pond will be attached to the simulated shrimp pond all the time. In order to measure the pH and adjust the pH of the water to normal condition.

The results showed that. When testing the experiment set, it was found that in the test. 100 percent can work normally, can be adjusted to be stable. And most acid-alkali control systems can work normally.

E-mail Address : waroot.boo@rmutr.ac.th

Period of Project : October 2014 – November 2015

สารบัญ

หน้า

กิตติกรรมประกาศ ..... ก
บทคัดย่อภาษาไทย ..... ข
บทคัดย่อภาษาอังกฤษ ..... ค
สารบัญ ..... ง
สารบัญภาพ ..... ฉ
สารบัญตาราง ..... ซ
บทที่
1 บทนำ 1
1. หลักการและเหตุผลการวิจัย 2
2. วัตถุประสงค์การวิจัย 2
3. สมมติฐานการวิจัย 2
4. ขอบเขตการวิจัย 2
5. วิธีดำเนินการวิจัย 2
6. ประโยชน์ที่คาดว่าจะได้รับ 3
2 ทฤษฎีแนวคิดและงานวิจัยที่เกี่ยวข้อง 4
1. ทฤษฎีแนวคิด 4
2. งานวิจัยที่เกี่ยวข้อง 22
3 วิธีดำเนินการวิจัย 24
1. ขั้นตอนการดำเนินงาน 24
2. ผังระบบ (Flow Chart) รวม 25
3. การพัฒนาและออกแบบอุปกรณ์ 28
4. ขั้นตอนการทดลอง 33
5. สรุปผลการทดลอง 37
4 ผลการวิเคราะห์ข้อมูล 38
1. ผลการวิเคราะห์ประสิทธิภาพของระบบ 38
3. สรุปผลการวิเคราะห์ข้อมูล 39
5 สรุป อภิปราย และข้อเสนอแนะ 41
1. สรุปผลการทำวิจัย 41
2. อภิปรายผลการวิจัย 41
3. ข้อเสนอแนะ 42

สารบัญ (ต่อ)

หน้า

บรรณานุกรม 43
ภาคผนวก ก 44
ประวัติผู้วิจัย 82

สารบัญภาพ

ภาพที่		หน้า
1	ปริมาณการนำเข้ากุ้งและผลิตภัณฑ์สดแช่แข็งของสหรัฐฯ	1
2	แสดงการใช้ฟังก์ชั่น setup ()	8
3	แสดงการใช้คำสั่ง loop ()	9
4	แสดงเงื่อนไขที่อยู่ภายในวงเล็บที่ต้องใช้ตัวกระทำเปรียบเทียบต่าง ๆ	10
5	แสดงอินพุตที่อ่านได้ 500 ให้ทำอะไร ถ้ามากกว่าให้ทำอีกอย่าง	11
6	หลังคำสั่ง else สามารถตามด้วยคำสั่ง if ได้ไม่จำกัดจำนวน	11
7	แสดงคำสั่งโดยการใช้ switch-case	13
8	แสดงการใช้คำสั่ง while ซึ่งทำงานวนรอบไปเรื่อย ๆ	13
9	แสดงบอร์ด Arduino Duemilanove	14
10	วงจรเซนเซอร์วัดค่า PH	15
11	โครงสร้างภายในของ Controller PCD8544 โครงสร้างภายในของ Controller PCD8544	16
12	แสดงโครงสร้าง Address และ Data ของหน้าจอแสดงผล LCD โครงสร้างภายในของ Controller PCD8544	17
13	การเพิ่มค่าแอดเดรสทางแนวตั้ง (Vertical addressing mode(V=1) โครงสร้างภายในของ Controller PCD8544	17
14	การเพิ่มค่าแอดเดรสทางแนวนอน(Horizontal addressing (V = 0)) โครงสร้างภายในของ Controller PCD8544	18
15	การเชื่อมต่อควบคุมจอแสดงผล LCD กับบอร์ด Arduino MEGA โครงสร้างภายในของ Controller PCD8544	18
16	ET-NOKIA LCD 5110Nokia โครงสร้างภายในของ Controller PCD8544	19
17	โครงสร้างRelay	20
18	ขาสัญญาณและการเชื่อมต่อ	21
19	การต่อใช้งานกับบอร์ด Arduino	21
20	SOLENOID VALVE (โซลินอยด์) 2 หุน 220V	22
21	บล็อกไดอะแกรมการทำงานของระบบวัด PH และ อุณหภูมิ อัตโนมัติด้วยเซนเซอร์ตรวจความเปลี่ยนแปรของน้ำ	24
22	ผังระบบ (Flow Chart) รวม	25
23	วงจรรวมทั้งหมด	28
24	ติดตั้ง SD Card สำหรับการเก็บข้อมูล	29
25	ติดตั้ง Real Time Clock	29
26	ติดตั้งเซ็นเซอร์วัดค่าความชื้น AM2302	30

สารบัญภาพ (ต่อ)

ภาพที่		หน้า
27	ติดตั้งจอแสดงผลกราฟิก ET-NOKIA LCD 5110Nokia	30
28	ติดตั้ง Relay Module ตัวที่ 1	31
29	ติดตั้ง Relay Module ตัวที่ 2	31
30	ติดตั้ง SOLENOID VALVE (โซลินอยด์) 2 หุน 220V ตัวที่ 1	32
31	ติดตั้ง SOLENOID VALVE (โซลินอยด์) 2 หุน 220V ตัวที่ 2	32
32	ติดตั้งเซนเซอร์ PH	33
33	เริ่มต้นตรวจสอบการทำงานของ SOLENOID VALVE	33
34	อุณหภูมิต่ำ SOLENOID VALVE 1 ทำงาน	34
35	อุณหภูมิต่ำ SOLENOID VALVE 2 ทำงาน	34
36	อุณหภูมิปกติ SOLENOID VALVE ไม่ทำงาน	35
37	วงจรรวมการควบคุม SOLENOID	35
38	ทดลองค่าสารละลายที่มีค่า PH 4.00	36
39	ทดลองค่าสารละลายที่มีค่า PH 7.00	36

สารบัญตาราง

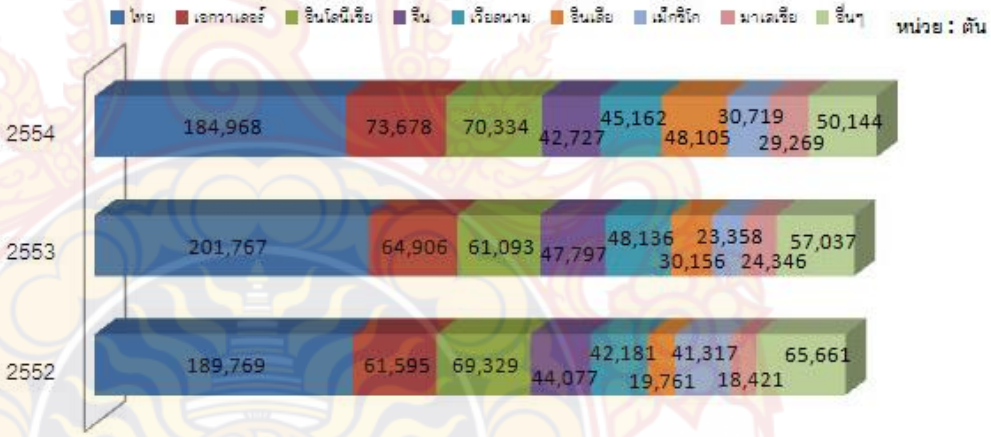
ตารางที่ **หน้า**

1 แสดงผลการทดลองการปิดเปิดน้ำของ SOLENOID VALVE โดยอ้างอิงจากเซนเซอร์วัดอุณหภูมิน้ำ 37
2 ตารางแสดงผลการทดสอบการรับค่าจากเซ็นเซอร์ต่าง ๆ 38
3 ตารางแสดงผลการทดสอบการบันทึกลงฐานข้อมูล 38
4 ตารางแสดงผลระบบควบคุมค่าความเป็นกรด-ด่างแบบอัตโนมัติทำการทดสอบควบคุมค่า PH ของน้ำให้อยู่ในช่วง 7.0-7.5 39

บทที่ 1
บทนำ

1. หลักการและเหตุผลการวิจัย

การเพาะเลี้ยงกุ้งในประเทศไทยมีมาเป็นเวลานาน ส่งผลให้อุตสาหกรรมการเพาะเลี้ยงและการแปรรูปกุ้งทะเลเลี้ยงของไทยมีพัฒนาการอย่างต่อเนื่องมาตั้งแต่ปี พ.ศ. 2528 โดยมีการเลี้ยงกุ้งกุลาดำในระยะแรก ต่อมามีการเพาะเลี้ยงกุ้งสายพันธุ์ต่างประเทศ คือ กุ้งขาวแวนนาไม จากปี พ.ศ. 2546 ถึงปัจจุบัน ประเทศไทยผลิตกุ้งทะเลเลี้ยงได้ 5.0-5.5 แสนตันต่อปี แปรรูปและส่งออกจนเป็นที่ยอมรับของตลาดทั่วโลก ทำรายได้มากกว่าปีละแสนล้านบาท



รูปที่1 ปริมาณการนำเข้ากุ้งและผลิตภัณฑ์สดแช่แข็งของสหรัฐฯ
ที่มา : NMFS of USA // ธนาคารแห่งประเทศไทย

[1]จังหวัดประจวบคีรีขันธ์มีความเหมาะสมด้านกายภาพในการเลี้ยงกุ้งทะเล ได้แก่ ความเหมาะสมของดิน คุณสมบัติของน้ำทะเล ลักษณะและความลาดชันของชายฝั่งทะเล ความสะดวกต่อการระบายน้ำ และปัจจัยทางเศรษฐกิจและสังคมที่ใช้เสริมในการพิจารณาพื้นที่ ที่สามารถนำมาพัฒนาเพื่อการเลี้ยงกุ้งทะเล ได้แก่ ถนน ไฟฟ้า แหล่งบริการข้อมูลและเทคโนโลยี โรงเพาะฟักกุ้งทะเล ราคาที่ดิน ตลาด และแหล่งเงินทุนในพื้นที่ เพื่อทำการจัดลำดับความเหมาะสมของพื้นที่ต่อการพัฒนาเพื่อการเลี้ยงกุ้งทะเลของจังหวัด ในระยะ 3 กิโลเมตร จากชายฝั่งทะเลคิดเป็นพื้นที่รวม 240,668.79 ไร่ ยกเว้นอำเภอหัวหินที่ถูกกำหนดให้เป็นพื้นที่การท่องเที่ยว โดยมีการเลี้ยงกุ้งทะเลในพื้นที่ส่วนใหญ่ของอำเภอปราณบุรีและอำเภอกุยบุรีซึ่งมีความ

เหมาะสมของปัจจัยกายภาพในระดับปานกลางแต่มีความพร้อมในปัจจัยทางเศรษฐกิจและสังคม จึงทำให้เป็นพื้นที่ที่มีการเลี้ยงกุ้งทะเลมากกว่าพื้นที่อื่น ๆ พื้นที่ที่มีความเหมาะสมมากที่สุดทั้งปัจจัยทางกายภาพและปัจจัยทางเศรษฐกิจและสังคมคือพื้นที่ในเขตอำเภอเมือง รองลงมา คือ อำเภอกุยบุรี ส่วนที่มีความเหมาะสมในระดับปานกลาง คือ อำเภอบางสะพานที่มีศักยภาพต่อการขยายตัวในอนาคต และอำเภอที่มีความเหมาะสมต่อการพัฒนาน้อยที่สุดคืออำเภอบางสะพานน้อย เนื่องจากมีความพร้อมด้านปัจจัยทางเศรษฐกิจและสังคมน้อยกว่าอำเภออื่น ๆ ในจังหวัดประจวบคีรีขันธ์

จากการที่มหาวิทยาลัยเทคโนโลยีราชมงคลรัตนโกสินทร์ วิทยาเขตวังไกลกังวลได้ทำการสำรวจอุตสาหกรรมการเลี้ยงกุ้งในบริเวณ อำเภอปราณบุรี อำเภอกุยบุรี พบปัญหาคือ การควบคุมค่าความเป็นกรด-ด่างของน้ำ ที่บางครั้งเกิดการเปลี่ยนแปลงอย่างรวดเร็ว หรือในช่วงเวลาที่ไม่สามารถสังเกตด้วยการมองได้ ทำให้ไม่สามารถปรับสมดุลของน้ำได้ทันท่วงทีที่ค่าความเป็นกรด-ด่างเปลี่ยนแปลงอย่างรวดเร็ว ส่งผลให้กุ้งเกิดการอาการน๊อกน้ำ หรืออาจตายได้หากไม่สามารถทำการแก้ปัญหาได้ทัน

2 วัตถุประสงค์การวิจัย

2.1 เพื่อพัฒนาระบบควบคุมค่าความเป็นกรด-ด่างแบบอัตโนมัติ

3. สมมติฐานการวิจัย

3.1 ระบบมีค่าประสิทธิ์ภาพของพัฒนาระบบควบคุมค่าความเป็นกรด-ด่างแบบอัตโนมัติ มีค่าไม่ต่ำกว่าร้อยละ 80

4. ขอบเขตการวิจัย

4.1 สามารถควบคุมค่าความเป็นกรด-ด่างให้อยู่ในช่วง 7.0-8.0

4.2 สามารถสั่งให้เปิดวาล์วที่ถังน้ำสารเคมีได้

4.3 ทดสอบระบบแบบจำลองในตู้กระจกเลี้ยงปลา

5. วิธีดำเนินการวิจัย

5.1 ออกแบบและพัฒนาเครื่องมือวัดค่าความเป็นกรด-ด่าง

5.2 ออกแบบและพัฒนาระบบควบคุมค่าความเป็นกรด-ด่าง

5.3 ติดตั้งระบบเครื่องมือวัดค่าความเป็นกรด-ด่าง

5.4 ทดสอบและปรับปรุงระบบ

5.5 บันทึกการทำงานของระบบควบคุมค่าความเป็นกรด-ด่าง

5.6 วิเคราะห์ข้อมูลและสรุปผลการวิจัย

6. ประโยชน์ที่คาดว่าจะได้รับ

6.1 นำมาใช้ในสถานประการขนาดเล็ก SME ด้านการเลี้ยงกุ้ง

บทที่ 2
ทฤษฎีแนวคิดและงานวิจัยที่เกี่ยวข้อง

ในการพัฒนาระบบควบคุมค่าความเป็นกรด-ด่างแบบอัตโนมัติให้บรรลุวัตถุประสงค์ของงานวิจัยครั้งนี้ จำเป็นต้องอาศัยหลักการและแนวคิดพื้นฐานทางทฤษฎีที่เกี่ยวกับการจัดทำดังต่อไปนี้

1 ทฤษฎีแนวคิด

1.1 นิยามของ pH

pH (พีเอช) เป็นค่าที่แสดงความเป็นกรดจากปฏิกิริยาของอิออนของไฮโดรเจน (H^+) สามารถทดสอบได้หลายวิธี โดยวิธีที่นิยมและง่ายสุดคือทดสอบด้วยกระดาษลิตมัสจากการเปลี่ยนสีสำหรับตัวเลขที่แสดงค่าพีเอช ถ้ามีค่าเท่ากับ 7 แสดงว่าสารนั้นเป็นกลางไม่มีฤทธิ์เป็นกรดหรือเบส ถ้ามีค่าน้อยกว่า 7 แสดงว่าเป็นกรด และถ้ามากกว่า 7 แสดงว่าเป็นเบส pH หมายถึง หน่วยวัดค่าความเป็น กรด-ด่าง ซึ่งมีช่วงตั้งแต่ 0-14 คือ ถ้าความเป็นกรดในอาหารสูงมาก ค่า pH = 0 แต่ถ้าความเป็นด่างสูงมาก ค่า pH = 14 หรือถ้าเป็นกลาง ไม่เป็นกรด ไม่เป็นด่าง ค่า pH = 7

1.1.1 กรณี pH ที่เกี่ยวข้องกับการเพาะเลี้ยงสัตว์น้ำ

pH (พี เอช) ย่อมาจาก positive potential of the hydrogen ions pH (พี เอช) หมายถึง ปริมาณความเข้มข้นของไฮโดรเจนอิออนที่มีอยู่ในน้ำค่า pH 4.0 หรือต่ำกว่า เป็นจุดอันตรายที่ทำให้สัตว์น้ำตายได้ค่า pH 4.0 - 6.0 สัตว์น้ำบางชนิดอาจไม่ตาย แต่จะทำให้สัตว์น้ำเจริญเติบโตช้าและทำให้การสืบพันธุ์หยุดชะงัก ค่า pH 6.5 - 9.0 เป็นระดับที่เหมาะสมกับการเพาะเลี้ยงสัตว์น้ำค่า pH 9.0 - 11.0 ไม่เหมาะสมกับการดำรงชีวิต หากสัตว์น้ำอาศัยอยู่เป็นเวลานาน ๆ จะทำให้ได้รับผลผลิตต่ำค่า pH 11.0 หรือมากกว่า เป็นพิษต่อสัตว์น้ำ

1.1.2 การควบคุมค่า pH

การวัดและควบคุมค่า pH แบบ on-line เป็นงานที่สำคัญประการหนึ่งในการวิเคราะห์กรรมวิธี ProMinent สามารถเสนอผลิตภัณฑ์ต่าง ๆ เพื่อใช้ในงานนี้ตั้งแต่ pH อิเลคโทรด จนถึงเครื่องทำให้ค่า pH เป็นกลาง

1.1.3 การตรวจวิเคราะห์หาค่าความเป็นกรด-ด่าง (PH)

ค่า (PH) เป็นค่าแสดงปริมาณเข้มข้นของอนุภาคไฮโครเจน (H^+) ในน้ำการวัดค่า PH มีประโยชน์ในด้านการควบคุมการทำงานและควรตรวจวิเคราะห์ทุกวัน เนื่องจากเป็นองค์ประกอบของสิ่งแวดล้อมที่สำคัญและมีผลต่อการทำงานของแบคทีเรีย ในระบบบำบัดน้ำเสียแบบบ่อผึ่งน้ำและคลองวนเวียนควรควบคุมให้ค่า PH ใกล้ 7 มากที่สุด และไม่ควรมีค่าเกิน 6.5-8.5 สำหรับค่าPH ที่เหมาะสมต่อการบำบัดน้ำเสียแบบไม่ใช้อ๊อกซิเจน เช่น ถังกรองไร้อากาศควรอยู่ในช่วง 6.6 -7.6 ถ้าค่าPH สูงหรือต่ำกว่านี้ ประสิทธิภาพของระบบบำบัดจะลดลงที่ค่าPH ต่ำกว่า 6.2 ประสิทธิภาพจะลดลงอย่างรวดเร็วเพราะสภาวะที่เป็นอันตรายต่อแบคทีเรียพวก METHANE FORMER

1.2 การวัดค่า PH ของน้ำด้วยเครื่องวัด pH

การวัดค่า pH ของสารละลาย หรือน้ำ วัดได้ทั้งวิธี colorimetric และ electronic (ใช้เครื่องวัด pH) ในที่นี้ ขอกล่าวถึงการวัดค่า pH ด้วยวิธี electronic เนื่องจากเป็นวิธีที่นิยมกันในปัจจุบัน และเป็นวิธีที่มีความถูกต้องแม่นยำสูงเครื่องวัด pH อาศัยหลักการวัดความต่างศักย์ไฟฟ้า (electrical potential) ที่เกิดขึ้นระหว่าง indicator electrode และ reference electrode ซึ่งจุ่มอยู่ในสารละลาย แล้วเปลี่ยนค่าความต่างศักย์ไฟฟ้า ให้เป็นค่า pH โดยการเทียบค่ากับสารละลายบัฟเฟอร์มาตรฐาน แต่เนื่องจากความต่างศักย์ไฟฟ้าที่เกิดขึ้นนั้น เปลี่ยนแปลงตามอุณหภูมิ ซึ่งเป็นผลให้ค่า pH เปลี่ยนแปลงตามไปด้วย เครื่องวัด pH จึงต้องมีการชดเชยค่าของการวัด อันเนื่องจากอุณหภูมิ โดยใช้หัววัดวัดอุณหภูมิของตัวอย่าง ขณะทำการวัด แล้วปรับเปลี่ยนเป็นค่า pH ณ ขณะนั้น

1.2.1 ส่วนประกอบเครื่องวัด pH

ตัวเครื่อง เป็นส่วนที่ใช้ในการวัดสัญญาณไฟฟ้า และแปลงหน่วยของความต่างศักย์จากมิลลิโวลต์ (mV) เป็นหน่วย pH โดยอ้างอิงจากสมการทางไฟฟ้าเคมี แล้วแสดงผลที่ได้ทางจอภาพ หรือเครื่องพิมพ์

หัววัดค่า pH (electrode) หัววัดค่า pH เป็นหัวใจหลักของการวัด ปัจจุบันนิยมใช้ combination electrode คือ มีทั้ง glass electrode และ reference electrode รวมอยู่ในอันเดียวกัน บริเวณปลายกระเปาะเป็นส่วนของ glass electrode ที่มีผิวเยื่อแก้วลักษณะเป็นเจลหนาประมาณ 10 มิลลิเมตร เป็นบริเวณที่เกิดการแลกเปลี่ยนอิออน ระหว่างโซเดียมอิออนในเยื่อแก้วกับไฮโดรเจนอิออนในสารละลาย

1.3 การ calibrate เทียบค่ามาตรฐานค่า pH

การ calibrate เป็นการจัดความสัมพันธ์ของกระแสสัญญาณของหัววัด (mV) กับค่ามาตรฐาน เมื่อนำความสัมพันธ์ที่ได้มาพล็อต (plott) กราฟ จะได้เป็นกราฟเส้นตรง ซึ่งค่ามาตรฐานค่าแรกที่ต้องทำการ calibrate คือ pH 7.00 กับค่าที่ต้องการควบคุมคุณภาพ โดยมีอุณหภูมิเป็นตัวแปร

1.4 สารละลายบัฟเฟอร์

สารละลายบัฟเฟอร์ใช้สำหรับการอ้างอิงในการวัดค่า pH เพื่อการวัดที่ถูกต้อง ควร calibrate ด้วยสารละลายบัฟเฟอร์ ที่มีค่า pH แตกต่างกัน 2 ค่า เช่น ใช้ pH 4 กับ 9 หรือ pH 4 กับ 7 เป็นต้น สารละลายบัฟเฟอร์สามารถเตรียมขึ้นมาเองได้ แต่โดยทั่วไป มักจัดหาจากผู้จำหน่ายสารเคมี

1.4.1 การดูแลรักษาสารละลายบัฟเฟอร์

1.4.1.1 ไม่ใช้สารละลายบัฟเฟอร์ที่เลยวันหมดอายุ ขึ้นรา และมีสิ่งปนเปื้อนที่อาจทำให้ pH ในสารละลายบัฟเฟอร์ เปลี่ยนได้

1.4.1.2 ปิดฝาภาชนะที่เก็บสารละลายบัฟเฟอร์เสมอ เพื่อป้องกันการปนเปื้อนและลดการดูดก๊าซคาร์บอนไดออกไซด์

1.4.1.3 PH ของสารละลายบัฟเฟอร์ เปลี่ยนแปลงตามอุณหภูมิ จึงต้องวัดอุณหภูมิของสารละลายบัฟเฟอร์ และคูณค่า pH ที่อุณหภูมินั้นก่อนจะปรับเครื่อง

1.4.1.4 ต้องเลือกสารละลายบัฟเฟอร์ที่มี pH ใกล้เคียงกับ pH ของน้ำตัวอย่าง เพื่อให้ได้ผลถูกต้องแม่นยำ

1.4.1.5 ห้ามเทสารละลายบัฟเฟอร์ที่ใช้แล้วกลับคืนขวด

1.5 Arduino

Arduino คือ ไมโครคอนโทรลเลอร์ ชนิดหนึ่ง ซึ่งเป็นแบบที่เรียกว่า Open Hardware กล่าวคือ Arduino อุปกรณ์ที่มีแบบส่วนประกอบเป็นมาตรฐานที่เปิดเผย หมายความว่า เราสามารถทำเองโดยใช้แบบที่มีการเปิดเผยทั่วไปก็ได้ หรือสามารถซื้อหาได้ง่าย มีราคาถูก มีซอฟต์แวร์ให้ใช้งานฟรี สามารถนำไปใช้งานทั่วไปหรือแบบธุรกิจได้โดยไม่ต้องเสียค่าลิขสิทธิ์ เป็นรูปแบบที่มีข้อมูลมากที่สุดบนอินเทอร์เน็ต การพัฒนาก็ง่าย เพราะมีตัวอย่างมากมาย และไม่ต้องเขียนโปรแกรมในรูปแบบ Low Level หมายความว่า เราสามารถใช้คำสั่งเขียนโปรแกรมได้เสมือนโปรแกรมภาษาชั้นสูงทั่วไป

1.5.1 โครงสร้างโปรแกรมของ Arduno

ในการเขียนโปรแกรมสำหรับโมดูล Freeduino จะต้องเขียนโปรแกรมโดยใช้ภาษาของ Arduino (Arduino Programming Language) ซึ่งตัวภาษาของ Arduino เองก็นำเอาโอเพ่นซอร์สโปรเจคชื่อ Wiring มาพัฒนาต่อภาษาของ Arduino

แบ่งออกเป็น ๒ ส่วนหลักคือ

- โครงสร้างภาษา (Structure) ตัวแปรและค่าคงที่
- ฟังก์ชั่น (Function)

ภาษาของ Arduino จะอ้างอิงตามภาษา C/C++ ซึ่งการเขียนโปรแกรมสำหรับ Arduino ก็คือการเขียนโปรแกรมภาษาซี โดยเรียกใช้ฟังก์ชั่นและไลบรารี่ที่ทาง Arduino ได้เตรียมไว้ให้แล้ว ซึ่งสะดวก และทำ ให้ผู้ที่ไม่มีความรู้ด้านไมโครคอนโทรลเลอร์อย่างลึกซึ่งสามารถเขียนโปรแกรมสั่งงานได้

โปรแกรมของ Arduino แบ่งได้เป็นสองส่วนคือ void setup () และ void loop () โดยฟังก์ชั่น setup () เมื่อโปรแกรมทำงานจะทำคำสั่งของฟังก์ชั่นนี้เพียงครั้งเดียว ใช้ในการกำหนดค่าเริ่มต้นของการทำงาน ส่วนฟังก์ชั่น loop () เป็นส่วนทำงาน โปรแกรมจะทำคำสั่งในฟังก์ชั่นนี้ต่อเนื่องกันตลอดเวลา โดยปกติใช้กำหนดโหมดการทำงานของขาต่าง ๆ กำหนดการสื่อสารแบบอนุกรม ฯลฯ ส่วนของ loop () เป็นโค้ดโปรแกรมที่ทำงาน เช่น อ่านค่าอินพุต ประมวลผล สั่งงานเอาต์พุต ฯลฯ โดยกำหนดค่าเริ่มต้น เช่น ตัวแปร จะต้องเขียนที่ส่วนหัวของโปรแกรม ก่อนถึงตัวฟังก์ชั่น นอกจากนั้นยังต้องคำนึงถึง ตัวพิมพ์เล็ก-ใหญ่ ของตัวแปรและชื่อฟังก์ชั่นให้ถูกต้อง

1.5.2 ส่วนของฟังก์ชั่น setup ()

ฟังก์ชั่นนี้จะเขียนที่ส่วนต้นของโปรแกรม ทำงานเมื่อโปรแกรมเริ่มต้นเพียงครั้งเดียว ใช้เพื่อกำหนดค่าของตัวแปร โหมดการทำงานของขาต่าง ๆ เริ่มต้นเรียกใช้ไลบรารี่ ฯลฯ เช่น

```
int  buttonPin  =  3;
void setup()
{
      beginSerial(9600);
      pinMode(buttonPin, INPUT);
}
void loop()
{
      if(digitalRead(buttonPin == HIGH)
      serialWrite('H');
      else
      serialWrite('L');
      delay(1000);
}
```

ภาพที่ 2 แสดงการใช้ฟังก์ชั่น setup ()

1.5.3 ส่วนของฟังก์ชั่น loop ()

หลักจากที่เขียนฟังก์ชั่น setup () ที่กำหนดค่าเริ่มต้นของโปรแกรมแล้ว ส่วนถัดมา คือ ฟังก์ชั่น loop () ซึ่งมีการทำงานตรงตามชื่อ คือ จะทำงานตามฟังก์ชั่นนี้วนต่อเนื่องตลอดเวลา ภายในฟังก์ชั่นนี้จะมีโปรแกรมของผู้ใช้ เพื่อรับค่าจาก พอร์ต ประมวลผล แล้วสั่งเอาต์พุตออกขาต่าง ๆ เพื่อควบคุมการทำงานของบอร์ด

```
int  buttonPin  =  3;
// setup initializes serial and the button pin
void setup()
{
      beginSerial(9600);
      pinMode(buttonPin, INPUT);
}

//loop checks the button pin each time
//and will send serial if it is pressed

Void loop()
{
      if(digitalRead(buttonPin == HIGH)
      serialWrite('H');
      else
      serialWrite('L');
      delay(1000);
}
```

ภาพที่ 3 แสดงการใช้คำสั่ง loop ()

1.5.4 คำสั่งควบคุมการทำงาน

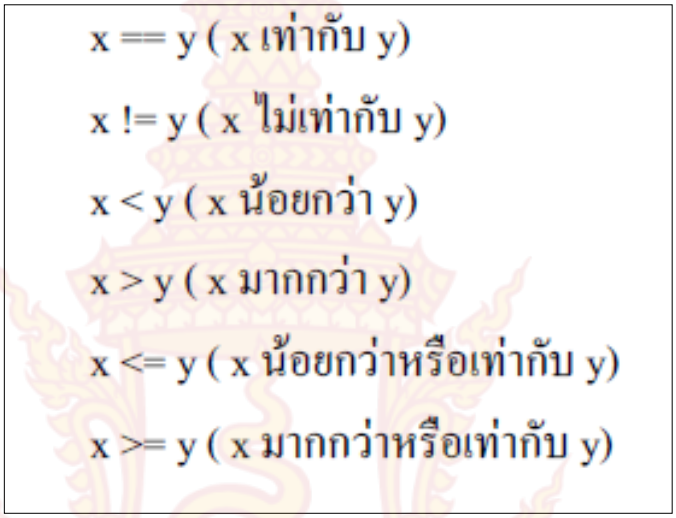
1.5.4.1 คำสั่ง if

ใช้ทดสอบเพื่อกำหนดเงื่อนไขการทำงานของโปรแกรม เช่น ถ้าอินพุตมีค่ามากกว่าค่าที่กำหนดไว้จะให้ทำอะไร โดยมีรูปแบบการเขียน ดังนี้

```
if(someVariable > 200)
{
 // do something here
 }
```

ตัวโปรแกรม จะทดสอบว่า ถ้าตัวแปร someVariable มีค่ามากกว่า 50 หรือไม่ ถ้าใช่ให้ทำอะไร ถ้าไม่ใช่ให้ข้ามการทำงานส่วนนี้ การทำงานของคำสั่งจะทดสอบเงื่อนไข ที่เขียน

ในเครื่องหมายวงเล็บ ถ้าเงื่อนไขเป็นจริง ทำตามคำสั่งที่เขียนในวงเล็บปีกกา ถ้าเงื่อนไขเป็นเท็จ ข้ามการทำงานส่วนนี้ไป



ภาพที่ 4 แสดงเงื่อนไขที่อยู่ภายในวงเล็บที่ต้องใช้ตัวกระทำเปรียบเทียบต่าง ๆ

ในการเปรียบเทียบตัวแปรให้ใช้ตัวกระทำ == (เช่น if (x == 10) ห้ามเขียนผิดเป็น = (เช่น if (x = 0)) คำสั่งที่เขียนผิดในแบบที่สองนี้ ทำให้ผลการทดสอบเป็นจริงเสมอ และเมื่อผ่านคำสั่งนี้แล้ว x มีค่าเท่ากับ 10 ทำให้การทำงานของโปรแกรมผิดเพี้ยนไป ไม่เป็นไปตามที่กำหนด เราสามารถใช้คำสั่ง if ในคำสั่งควบคุมการแยกเส้นทางของโปรแกรม โดยใช้คำสั่ง if .. else

1.5.4.2 คำสั่ง if...else

ใช้ทดสอบเพื่อกำหนดเงื่อนไขการทำงานของโปรแกรมได้มาก กว่าคำสั่ง if ธรรมดา โดยสามารถกำหนดได้ว่า ถ้าเงื่อนไขเป็นจริงให้ทำอะไร ถ้าเงื่อนไขเป็นเท็จให้ทำอะไร เช่น ถ้าค่าอินพุตแอนาลอกที่อ่านได้น้อยกว่า 500 ให้ทำอะไร ถ้าค่ามากกว่าหรือเท่ากับ 500 ให้ทำอีกอย่าง สามารถเขียนคำสั่งได้ดังนี้

```
if(pinFiveInput < 200)
{
      //  do Thing A
}
else
{
      // do Thing B
}
```

ภาพที่ 5 แสดงอินพุตที่อ่านได้ 500 ให้ทำอะไร ถ้ามากกว่าให้ทำอีกอย่าง

หลังคำสั่ง else สามารถตามด้วยคำสั่ง if สำหรับการทดสอบอื่น ๆ ทำให้รูปแบบคำสั่งกลายเป็น if....else....if เป็นการทดสอบ อื่น ๆ เมื่อเป็นจริงให้ทำตามที่ต้องการ ดังตัวอย่างต่อไปนี้

```
if(pinFiveInput < 500)
{
      //  do Thing A
}

else if (pinFiveInput <= 1000)
{
      // do Thing B
}
else if
{
      // do Thing C
}
```

ภาพที่ 6 หลังคำสั่ง else สามารถตามด้วยคำสั่ง if ได้ไม่จำกัดจำนวน

หลังคำสั่ง else สามารถตามด้วยคำสั่ง if ได้ไม่จำกัดจำนวน (สามารถใช้คำสั่ง switch case แทนคำสั่ง if...else...if สำหรับการทดสอบเงื่อนไขจำนวนมาก ๆ ได้ เมื่อใช้คำสั่ง if..else แล้วต้องกำหนดด้วยว่าถ้าทดสอบไม่ตรงกับเงื่อนไขใด ๆเลยให้ทำอะไรโดยให้กำหนดที่คำสั่ง else ตัสุดท้าย

1.5.4.3 คำสั่ง for ()

คำสั่งนี้ใช้เพื่อสั่งให้คำสั่งที่อยู่ภายในวงเล็บปีกกาหลัง for มีการทำงานซ้ำกันตามจำนวนรอบที่ต้องการ คำสั่งนี้มีประโยชน์มากสำหรับการทำงานใด ๆ ที่ต้องทำซ้ำกันและทราบจำนวนรอบของการทำซ้ำที่แน่นอน มักใช้คู่กับตัวแปรอะเรย์ ในการเก็บสะสมค่าที่อ่านได้จากขาอินพุตแอนาลอกหลาย ๆ ขาที่มีหมายเลขขาต่อเนื่องกัน

รูปแบบของคำสั่ง for () แบ่งได้ ๓ ส่วนดังนี้

```
for (initialization; codition; increment)
}
// statement (s);
}
```

เริ่มต้นด้วย initialization ใช้กำหนดค่าเริ่มต้นของตัวแปรควบคุมการวนรอบในการทำงานแต่ละรอบจะทดสอบ condition ถ้าเงื่อนไขเป็นจริงจะทำตามคำสั่งในวงเล็บปีกกาแล้วมาเพิ่มหรือลดค่าตัวแปรตามที่สั่งใน increment แล้วทดสอบเงื่อนไขอีก ทำซ้ำกว่าเงื่อนไขเป็นเท็จ

```
for(int i=1; i<=8; i++)
{
//statement using the value i;
}
```

คำสั่ง for ของภาษาซีจะยืดหยุ่นกว่าคำสั่ง for ของภาษาคอมพิวเตอร์อื่น ๆ โดยสามารถละเว้นบางส่วนหรือทั้งสามส่วนของคำสั่ง for ได้ อย่างไรก็ตามยังคงต้องมีเซมิโคลอน เราสามารถนำคำสั่งภาษาซีที่มีตัวแปรที่ไม่เกี่ยวข้องมาเขียนในส่วนของ initialization , condition และ increment ของคำสั่ง for ได้

1.5.4.4 คำสั่ง switch-case

ใช้ทดสอบเงื่อนไขเพื่อกำหนดการทำงานของโปรแกรม ถ้าตัวแปรที่ทดสอบตรงกับเงื่อนไขใดก็ให้ทำงานตามที่กำหนดไว้

พารามิเตอร์

var ตัวแปรที่ต้องการทดสอบว่าตรงกับเงื่อนไขใด

default ถ้าไม่ตรงกับเงื่อนไขใด ๆ เลยให้ทำคำสั่งต่อท้ายนี้

break เป็นส่วนสำคัญมากให้เขียนต่อท้าย case ต่างๆเมื่อพบเงื่อนไขนั้นแล้วให้ทำตามคำสั่งต่างๆแล้วให้หยุดการทำงานของคำสั่ง switch-case ถ้าลืมเขียน break เมื่อพบเงื่อนไขทำตามเงื่อนไขแล้ว โปรแกรมจะทำงานตามเงื่อนไขต่อไปเรื่อยๆ จนกว่าจะพบคำสั่ง break

```
switch (var)
{
        case 1:
                // do something when var ==1
        break;
        case 2:
                // do something when var ==2
        break;
        default:
                // if nothing else matches, do the default
}
```

ภาพที่ 7 แสดงคำสั่งโดยการใช้ switch-case

1.5.4.5 คำสั่ง while

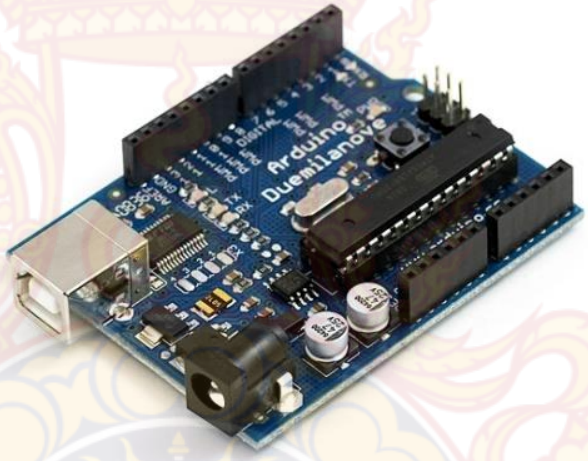
เป็นคำสั่งวนรอบ โดยจะทำคำสั่งที่เขียนไว้ในวงเล็บปีกกาอย่างต่อเนื่อง จนกว่าเงื่อนที่เขียนไว้ในวงเล็บของคำสั่ง while( ) จะเป็นเท็จ คำสั่งที่ให้ทำซ้ำจะต้องมีการเปลี่ยนแปลงค่าตัวแปรที่ใช้ทดสอบ เช่นมีการเพิ่มค่าตัวแปรหรือมีเงื่อนไข ภายนอกเช่น อ่านค่าจากเซ็นเซอร์ได้เรียบร้อยแล้วให้หยุดการอ่านค่า มิฉะนั้นเงื่อนไขในวงเล็บของ while( ) เป็นจริงตลอดเวลาทำให้คำสั่ง while ทำงานวนรอบไปเรื่อยๆไม่รู้จบ

```
var  = 0;
while (var <= 7)
{
        // do something repetitive 200 times
        var++;
}
```

ภาพที่ 8 แสดงการใช้คำสั่ง while ซึ่งทำงานวนรอบไปเรื่อย ๆ

1.5.4.6 Arduino Duemilanove

ในการทำลิฟต์ครั้งนี้กลุ่มลิฟต์ได้ใช้ บอร์ด Arduino Duemilanove ซึ่งเป็นบอร์ดไมโครคอนโทรลเลอร์ที่ใช้ชิป ATmega168 (datasheet) หรือ ATmega328 (datasheet) มีขาใช้งานที่เป็น I/O ดิจิตอล 14 ขา ซึ่งมี 6 ขาใช้งานเป็น PWM (pulse width modulation) มี 6ขารับสัญญาณอนาล็อค ใช้ oscillator 16 MHz การเชื่อมต่อแบบ USB, Power Jack, ICSP header, และ ปุ่มรีเซ็ต เป็นบอร์ดไมโครคอนโทรลเลอร์ที่รวมทุกอย่างไว้ในบอร์ดง่ายต่อการเชื่อมต่อกับคอมพิวเตอร์ ไม่ว่าจะเป็นทาง usb cable หรือ AC-to-DC adapter



ภาพที่ 9 แสดงบอร์ด Arduino Duemilanove

บนบอร์ด สามารถใช้เป็นขา input/output ได้ สามารถใช้ฟังก์ชั่น pinMode(), digitalWrite(), anddigitalRead()ได้ ซึ่งขาพวกนี้จะมีแรงดัน 5volt กระแสสูงสุด 40 mA. สามารถมีการต่อตัวต้านทางแบบ pull-up เพื่อเพิ่มกระแส ขนาด 20-50kOhms. ในบางขาอาจมีฟังก์ชั่นพิเศษ จึงควรศึกษาให้ดี ก่อนจะเพิ่ม

Serial: 0 (RX) and 1 (TX). ใช้รับ RX(recieve)และส่งTX(transmit) TTL การส่งข้อมูลทางพอร์ทอนุกรม ซึ่งขาเหล่านี้จะต้องต่อตรงกับ pin ของ FTDI USB-to-TTL Serial chip.

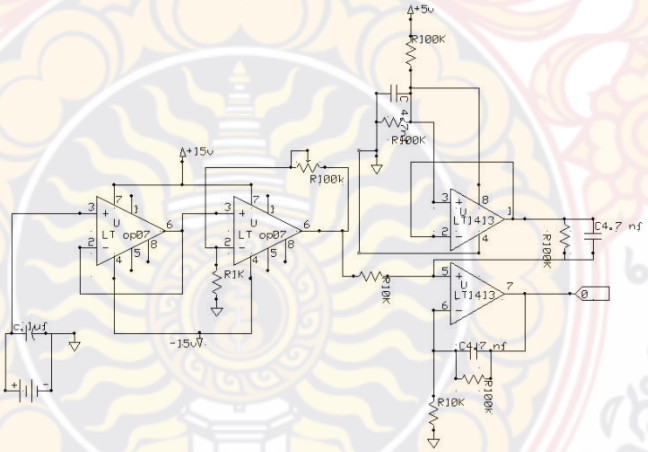
Arduino Duemilanove มีรูปแบบการติดต่อสื่อสารกับคอมพิวเตอร์ The ATmega168 and ATmega328 จะใช้การติดต่อสื่อสารทางพอร์ทอนุกรม Pin0 (RX) , Pin1 (TX) . ชิฟ FTDI FT232RL บนบอร์ด ใช้เป็นช่องทางการสื่อสารแบบอนุกรม เป็นมากกว่าUSB และ FTDI drivers (มาจาก Arduino software) เป็นซอฟแวร์ที่ติดต่อกับคอมพิวเตอร์พอร์ท

The Arduino software ประกอบด้วย การส่งข้อมูลบนหน้าจอ ที่มาจากบอร์ด Arduino ขา TX , RXบนบอร์ดมีหลอดLedติดอยู่จะมีการกระพริบ เมื่อเริ่มมีการส่งข้อมูล ด้วยชิฟ FTDI และการเชื่อมต่อกับ Usb กับคอมพิวเตอร์

Arduino Duemilanove มีการรีเซ็ต ที่ช่วยในการป้องกันพอร์ต USB ของคอมพิวเตอร์จากการช็อตหรือกระแสเกิน ถึงแม้ว่าคอมพิวเตอร์ส่วนใหญ่จะมีการป้องกันภายในตัว โดยมีฟิวส์เป็นการเพิ่มชั้นการป้องกัน ถ้ามีกระแสไหลผ่านมากกว่า 500mA เข้ามาทางพอร์ต USB ฟิวจะตัดการเชื่อมต่อโดยอัตโนมัติ จนกว่าการช็อตหรือการเกินกระแสจะถูกตัดออก ส่วนบอร์ด Arduino Duemilanove ความยาวและความกว้างสูงสุดมีขนาด 2.7 และ 2.1 inches ตามลำดับ มีการเชื่อมต่อ USB และ Power Jack ที่ยื่นออกมา มีรูสำหรับใส่น็อต (screw) จำนวนสามรู ไว้สำหรับยึดติดกับพื้นผิว หรือ กรณีอื่น

1.6 เซนเซอร์วัดค่า PH

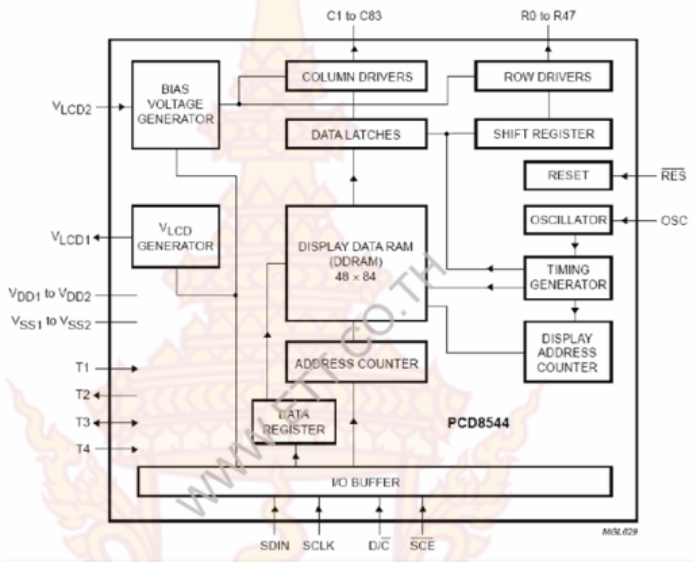
เป็นวงจรกระแสโดยใช้โป๊ปในการวัดค่าโดยใช้ Opamp เป็นตัวขยายสัญญาณและยกกระแสของแรงดันโป๊ป และจะทำการส่งข้อมูลออกทางขา Opamp และประมวลผลข้อส่งข้อมูล Data Serial Port



ภาพที่ 10 วงจรเซนเซอร์วัดค่า PH

1.7 จอแสดงผลกราฟิก ET-NOKIA LCD 5110Nokia

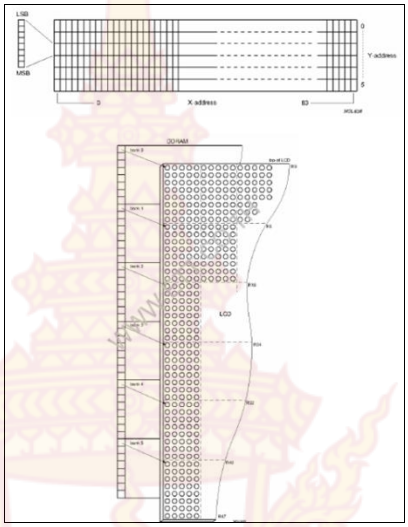
LCD 5110 เป็น LCD Graphic ขนาด 48x84 Dot ซึ่งมี Controller/Driver ภายใน คือ PCD8544 ทำหน้าที่ควบคุม การแสดงผล และ การทำงานทั้งหมด โดยภายใน Controller PCD8544 จะมีโครงสร้างไดอะแกรมดังรูป



ภาพที่ 11 โครงสร้างภายในของ Controller PCD8544 โครงสร้างภายในของ Controller PCD8544 ที่มา http://www.arduino.in.th/

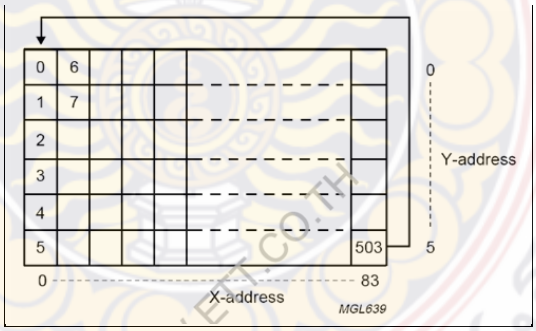
1.7.1 การจัดการแอดเดรสของ LCD (Addressing)

ลักษณะการจัดเรียงตำแหน่งแอดเดรสของหน่วยความจำที่แสดงผลบนหน้าจอ LCD (DDRAM) จะจัดแบบเมตริก (Matrix) ประกอบด้วย แถว (Y Address) จำนวน 6 แถว ประกอบด้วย Y-Address 0 ถึง Y-Address 5 และทางด้านคอลัมน์ (X Address) จำนวน 84 คอลัมน์ ประกอบด้วย X-Address 0 ถึง X-Address 83 การเข้าถึงตำแหน่งการแสดงผลบนหน้าจอ LCD จะอาศัยความสัมพันธ์ของ X-Address และ Y-Address ประกอบกัน ส่วนข้อมูล(Data) ที่จะส่งไปแสดงผล จะมีขนาด 8 บิต (1 ไบต์) ซึ่งจะถูกวางในแนวตั้ง โดยบิต MSB จะ อยู่ด้านล่าง และ LSBอยู่ด้านบน ดังรูปต่อไปนี้

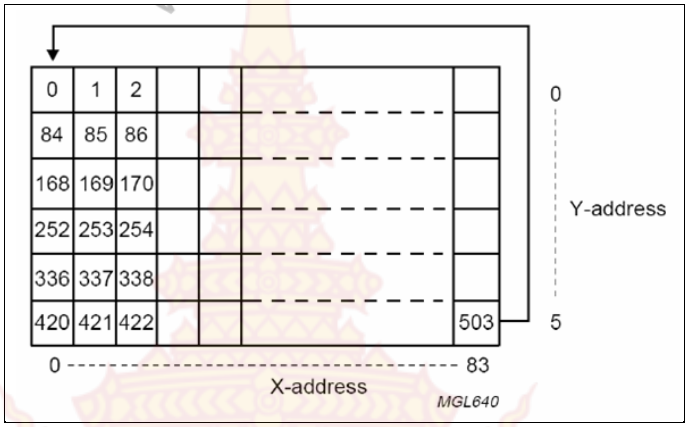


ภาพที่ 12 แสดงโครงสร้าง Address และ Data ของหน้าจอแสดงผล LCD โครงสร้างภายในของ Controller PCD8544 ที่มา http://www.arduino.in.th/

สามารถเขียนข้อมูลลงไปในตำแหน่งของหน่วยความจำแสดงผล (DDRAM) ได้อย่างต่อเนื่อง ซึ่งค่าของตำแหน่ง X-Address และ Y-Address จะเพิ่มค่าขึ้นเองโดยอัตโนมัติ ซึ่งสามารถกำหนดรูปแบบการทำงานของแอดเดรสได้ 2 รูปแบบ คือ แบบ แนวตั้ง(Vertical addressing mode(V=1)) โดยค่าของ Y-Address จะเพิ่มขึ้น 1 ค่าทุกครั้ง และ แบบ แนวนอน (Horizontal addressing (V = 0)) ค่าของ X-Address จะเพิ่มขึ้น 1 ค่าทุกครั้ง



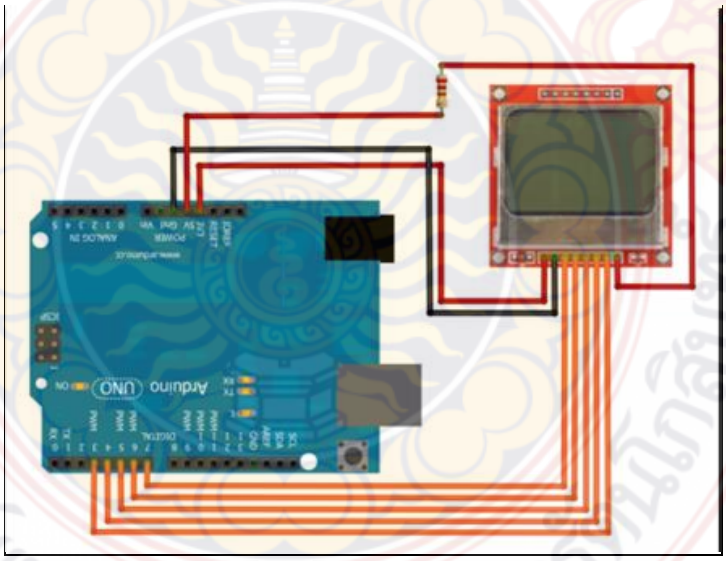
ภาพที่ 13 การเพิ่มค่าแอดเดรสทางแนวตั้ง (Vertical addressing mode(V=1) โครงสร้างภายในของ Controller PCD8544 ที่มา http://www.arduino.in.th/



ภาพที่ 14 การเพิ่มค่าแอดเดรสทางแนวนอน(Horizontal addressing (V = 0)) โครงสร้างภายในของ Controller PCD8544 ที่มา http://www.arduino.in.th/

1.4.2 การเชื่อมต่อควบคุมจอแสดงผล LCD กับบอร์ด

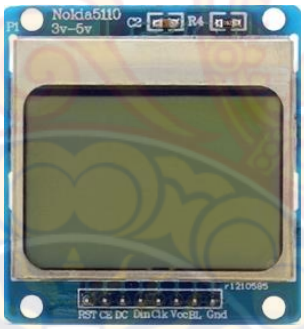
การเชื่อมต่อสัญญาณเพื่อควบคุมการทำงานของ LCD จะเป็นแบบอนุกรมโดยจะมีขาสัญญาณต่าง ๆ ในการเชื่อมต่อดังนี้



ภาพที่ 15 การเชื่อมต่อควบคุมจอแสดงผล LCD กับบอร์ด Arduino MEGA โครงสร้างภายในของ Controller PCD8544 ที่มา http://www.arduino.in.th/

1.7.3 หน้าที่ของขาสัญญาณ LCD

- VCC เป็นขาไฟเลี้ยงบวก ใช้ได้ตั้งแต่ 2.7-5 VDC
- GND ขาสัญญาณกราว์
- SCE ขาสัญญาณ CHIP ENABLE ทำหน้าที่ควบคุมการทำงานของขาสัญญาณควบคุมต่าง ๆ
- RESET สัญญาณรีเซตการทำงานของ LCD
- D/C เป็นขาสัญญาณที่ใช้กำหนดประเภทของข้อมูล ระหว่าง ข้อมูล (Data) กับ คำสั่ง (Command)
- SDIN ขาสัญญาณข้อมูล (SERIAL DATA LINE)
- SCLK ขาสัญญาณนาฬิกา (SERIAL CLOCK LINE)
- LED ขาสัญญาณควบคุมการทำงานของหลอดไฟ LED (Back Light)



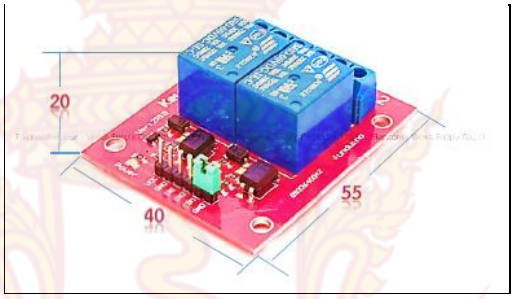
ภาพที่ 16 ET-NOKIA LCD 5110Nokia โครงสร้างภายในของ Controller PCD8544 ที่มา http://www.arduino.in.th/

1.7.4 คุณสมบัติของ LCD 5110

- หน้าจอแสดงผลความละเอียด 48 x 84 Dot
- ติดต่อสื่อสารแบบระบบบัสอนุกรม (Serial Bus Interface) ความเร็วสูงสุด 4.0 Mbits/S
- มีคอนโทรลเลอร์เบอร์ PCD8544 ภายในควบคุมการทำงาน
- มีหลอดไฟ Back-Light
- ทำงานที่แรงดัน 2.7 - 5.0 โวลท์
- กินกำลังงานต่ำ เหมาะกับฟังก์ชันการใช้งานกับพวกแบตเตอรี่
- ช่วงอุณหภูมิการทำงาน -25 ถึง +70 องศาเซลเซียส
- รองรับสัญญาณอินพุทแบบ CMOS

1.8 Relay Module

เป็นบอร์ดรีเลย์ขนาด 2 ช่อง มีเอาต์พุตคอนเน็คเตอร์ที่รีเลย์เป็น NO/COM/NC สามารถใช้กับโหลดได้ทั้งแรงดันไฟฟ้า DC และ AC โดยใช้สัญญาณในการควบคุมการทำงานด้วยสัญญาณโลจิก TTL



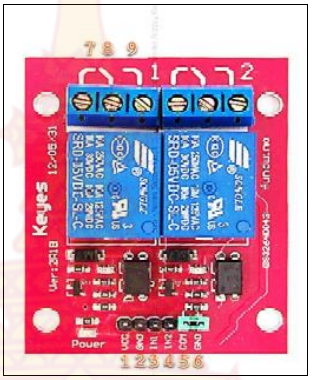
ภาพที่ 17 โครงสร้างRelay ที่มา http://www.micontechlab.com/

1.8.1 คุณสมบัติ (Features)

- รีเลย์เอาต์พุตแบบ SPDT จำนวน 2 ช่อง
- สั่งงานด้วยระดับแรงดัน TTL
- CONTACT OUTPUT ของรีเลยBรับแรงดันได$สูงสุด 250 VAC 10 A , 30 VDC 10 A
- มี LED แสดงสถานะ การทำงานของรีเลยBและแสดงสถานะของบอร์ดมีจัมพ์เปอร์สำหรับเลือกว่าจะใช้กราวด์รวมหรือแยก
- มี OPTO-ISOLATED เพื่อแยกกราวด์ส่วนของสัญญาณควบคุมกับไฟที่ขับรีเลยBออกจากกัน

1.8.2 คุณลักษณะ (Specification)

- ควบคุมไฟ DC ได้สูงสุด 30VDC 10A และ ไฟ AC สูงสุด 250VAC 10A
- ระดับสัญญาณอินพุตควบคุมแบบ TTL ทำงานด้วยสัญญาณแบบ Active High
- ขนาดรูยึดบอร์ด 3mm
- ขนาด (L x W x H): 55 x40 x 20 mm

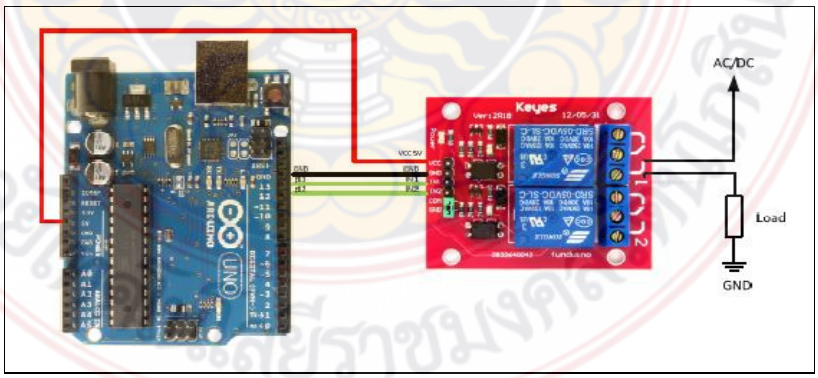


ภาพที่ 18 ขาสัญญาณและการเชื่อมต่อ ที่มา http://www.micontechlab.com/

1.8.3 ขาสัญญาณและการเชื่อมต่อ(Pin Definition and Rating)

- +VCC ขาไฟ 5VDC
- GND (ของรีเลย์)
- ขาสัญญาณอินพุต Relay 1 (IN1)
- ขาสัญญาณอินพุต Relay 2 (IN2)
- COM (คอมมอนของ OPTO)
- GNE (กราว์ของบอร์ดเป็นกราว์เดียวกันกับขาที่ 2)
- NC (Normal Close) ซึ่งหมายถึงหน้าสัมผัสแบบปกติปิด
- COM (Common) ที่จะตัดหรือต่อวงจรขา NC, NO
- NO (Normal Open) ซึ่งหมายถึงหน้าสัมผัสแบบปกติเปิด

1.8.4 การใช้งาน (Usage) การเชื่อมต่อฮาร์ดแวร์ (Hardware Installation)



ภาพที่ 19 การต่อใช้งานกับบอร์ด Arduino ที่มา http://www.micontechlab.com/

1.9 SOLENOID VALVE (โซลินอยด์) 2 หุน 220V

1.9.1 คุณสมบัติของ SOLENOID VALVE

- เป็นอุปกรณ์ทางน้ำเข้า-ออก ใช้ท่อน้ำขนาด 1/4 FIP (เกลียวใน)
- 0-120 PSI
- Orifice 2.5 mm
- Normal Close NC (Normal Close หรือ ปรกติปิด)
- เป็นวาล์วเปิด-ปิดอัตโนมัติ ที่สั่งการด้วยไฟฟ้า
- Fitting รูขนาด 2 หุน
- วาล์ว เปิด เมื่อมีการจ่ายไฟฟ้าเข้า
- ไฟฟ้า 220 VAC
- ตัวเครื่องทำด้วยวัสดุพลาสติก

ภาพที่ 20 SOLENOID VALVE (โซลินอยด์) 2 หุน 220V) ที่มา http://www.duan-daw.com/product/271/solenoid-valve-โซลินอยด์-2-หุน-24-vdc-0-3a-เกลียวใน

2. งานวิจัยที่เกี่ยวข้อง

ทัศนีย์ นลวชัยและคณะ[2] ทำการทดลองผลของปริมาณออกซิเจนที่ละลายในน้ำที่แตกต่างกัน 3 ระดับ ต่อการกินอาหารของกุ้งขาวแวนนาไม (Litopenaeus vannamei) (7กรัม) ในตู้ทดลองขนาดความจุ 100 ลิตร (ความเค็ม 25 ส่วนในพันส่วน อุณหภูมิ 28+1 OC) พบว่ากุ้งชุดการทดลองที่เลี้ยงในน้ำที่มีปริมาณออกซิเจนที่ละลายในน้ำ (DO) ต่ำกว่า 2 มิลลิกรัมต่อลิตร (พีพีเอ็ม) มีอาหารเหลือหลังจากให้อาหาร 30นาทีมากที่สุด (73.31+3.65 เปอร์เซ็นต์)ซึ่งแตกต่างอย่างมีนัยสำคัญทางสถิติ($p<0.05$) กับกุ้งชุดการทดลองที่ควบคุม DO ให้อยู่ระหว่าง 2-4 พีพีเอ็ม เท่ากับ 13.22+5.67เปอร์เซ็นต์ และชุดการทดลองที่มีค่า DO มากกว่า 4 พีพีเอ็มเท่ากับ 2.60+3.31เปอร์เซ็นต์ ตามลำดับ และเมื่อทำการทดลองผลของระดับออกซิเจน ร่วมกับการ

ควบคุมปริมาณแอมโมเนียรวมให้ได้ 3 มิลลิกรัมต่อลิตร และค่าพีเอช 7.5 และ 8.5 ต่อการกินอาหารของกุ้งขาวแวนนาไม พบว่ากุ้งชุดการทดลองที่ควบคุม DO ต่ำกว่า 2 พีพีเอ็มที่พีเอช 8.5 มีอาหารเหลือหลังจากให้อาหาร 30 นาทีมากที่สุด (73.38+1.90 เปอร์เซ็นต์) ซึ่งไม่แตกต่างทางสถิติ กับกุ้งชุดการทดลองที่ควบคุม DO ต่ำกว่า 2 พีพีเอ็ม ที่พีเอช 7.5 (73.09+1.44เปอร์เซ็นต์) แต่แตกต่างอย่างมีนัยสำคัญทางสถิติกับกุ้งในชุดการทดลองที่ควบคุม DO ให้อยู่ระหว่าง 2-4 และมากกว่า 4 พีพีเอ็ม ทั้งพีเอช 7.5 และ 8.5 การทดลองนี้แสดงให้เห็นว่าปริมาณออกซิเจนที่ละลายน้ำและพีเอช มีผลต่อการกินอาหารของกุ้งขาวแวนนาไม

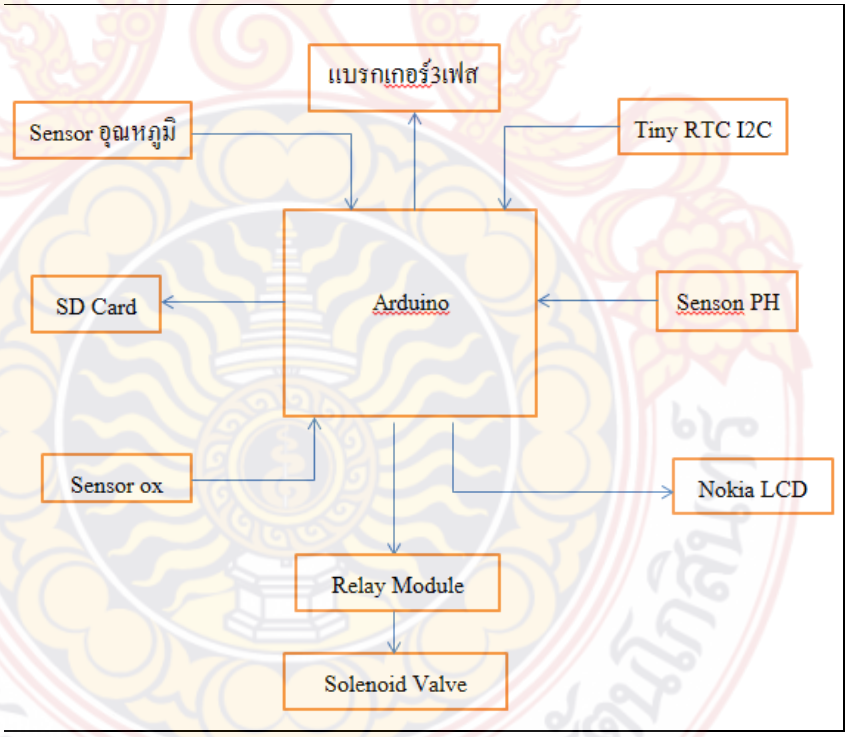
ศิริชัย เติมโชคเกษม และ สุรเชษฐ์ โทวราภา[3] ได้นำเสนอ “ตู้ปลาอัจฉริยะ” ซึ่งอาศัยการทำงานของไมโครคอนโทรลเลอร์สำหรับควบคุมส่วนประกอบของระบบต่างๆ ภายในตู้ปลา เช่น ระบบการเปลี่ยนน้ำโดยอาศัยเครื่องมือวัดค่า pH ที่สามารถตรวจสอบความเป็นกรด – เบสของน้ำจากมูลปลา ระบบกรองของเสียออกจากตู้ปลาที่สามารถหมุนเวียนน้ำระหว่างตู้ปลา กับระบบกรองของเสียแบบอัตโนมัติ ระบบให้อาหารที่ทำหน้าที่ควบคุมการดูแลให้อาหารปลาอย่างสม่ำเสมอ รวมถึงระบบการเปิด-ปิดแสงไฟภายในตู้ปลาที่สามารถทำงานได้ทั้งแบบกำหนดเวลาเปิด-ปิดอัตโนมัติหรือแบบใช้ LDR เป็นตัววัดแสงเพื่อควบคุมแสงสว่างแบบอัตโนมัติ ทั้งนี้ตู้ปลาอัจฉริยะดังกล่าว จะช่วยอำนวยความความสะดวกให้แก่ผู้ที่สนใจเลี้ยงปลาได้เป็นอย่างดี

บทที่ 3
วิธีดำเนินการวิจัย

ในการศึกษางานวิจัยครั้งนี้ ผู้จัดทำได้รวบรวมแนวคิด ทฤษฎี เอกสาร และงานวิจัยที่เกี่ยวข้อง แล้วนำข้อมูลที่รวบรวมได้ มาสรุปสาระสำคัญในประเด็นที่จะก่อให้เกิดประโยชน์ต่อการพัฒนาระบบควบคุมค่าความเป็นกรด-ด่างแบบอัตโนมัติในอุตสาหกรรมการเลี้ยงกุ้ง ซึ่งมีรายละเอียดดังนี้

1. ขั้นตอนการดำเนินงาน

1.1 บล็อกไดอะแกรมแสดงการทำงานของระบบวัด PH และ อุณหภูมิ อัตโนมัติด้วยเซนเซอร์ตรวจความเปลี่ยนแปรของน้ำ

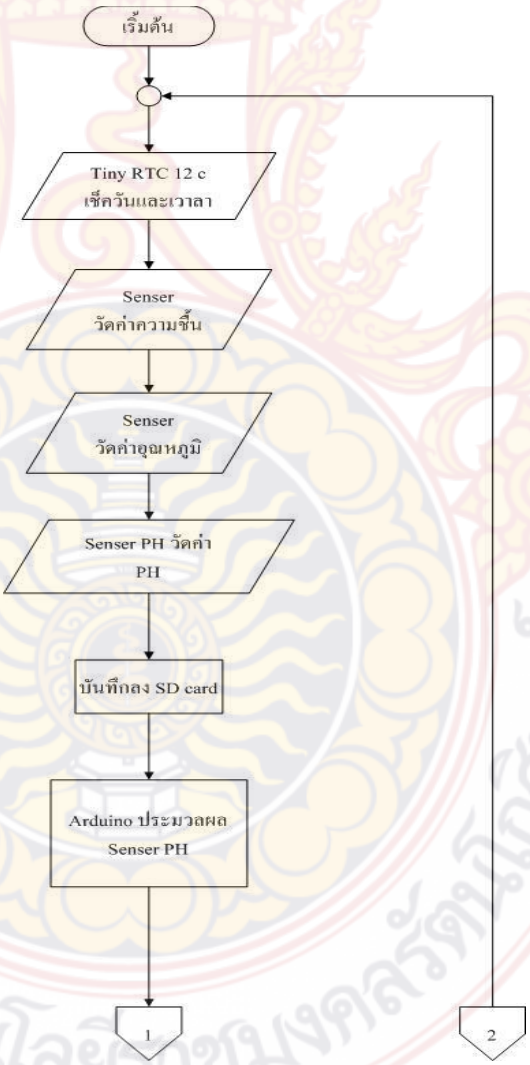


ภาพที่ 21 บล็อกไดอะแกรมการทำงานของระบบวัด PH และ อุณหภูมิ อัตโนมัติด้วยเซนเซอร์ตรวจความเปลี่ยนแปรของน้ำ

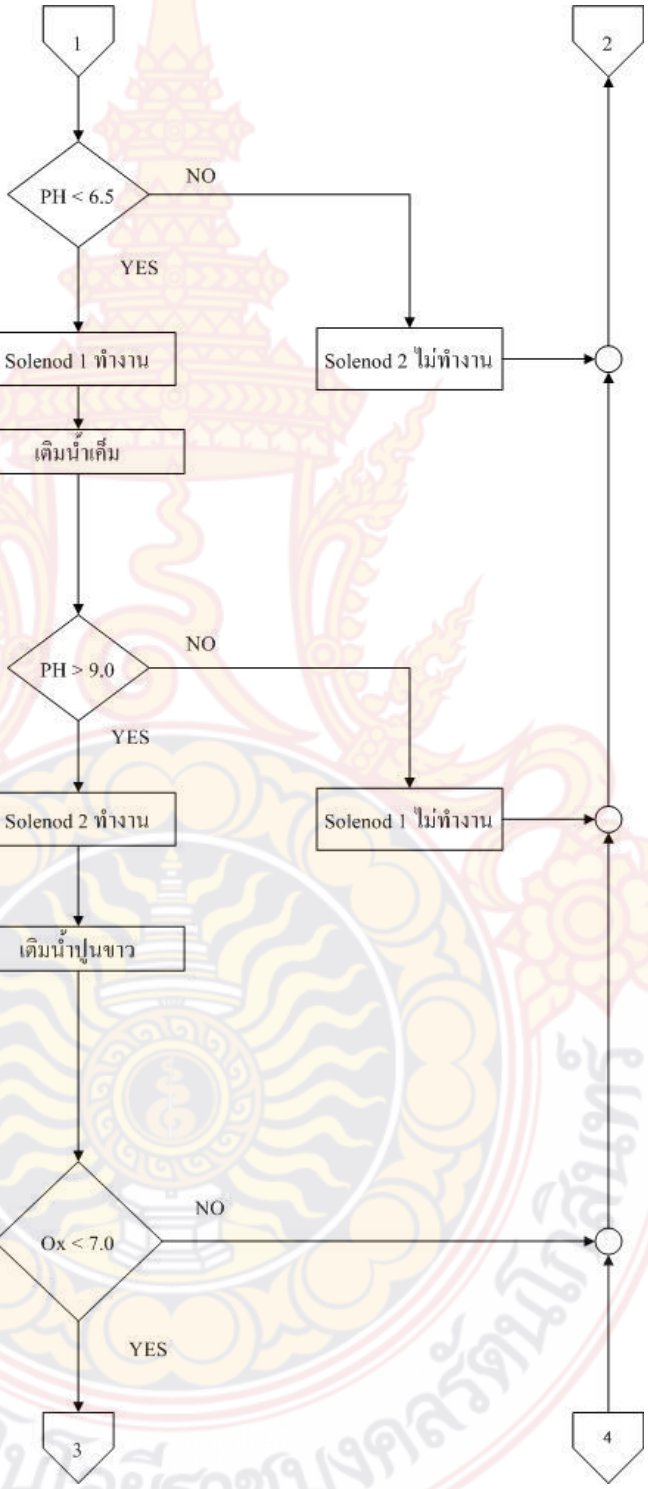
จากรูปที่ 3.1 สามารถอธิบายการทำงานได้ดังนี้การทำงานของบล็อกไดอะแกรมจะมีการทำงานตามขั้นตอนดังนี้ Tiny Rte I2c จะทำการส่งข้อมูลวันและเวลาไปที่บอร์ด Arduino แล้วส่งไป

เก็บที่ SD Card และ Senser อุณหภูมิจะทำการวัดค่าส่งไปให้บอร์ด Arduino ทำการประมวลผลและส่งไปเก็บใน SD Caerd และ Senser Oxygen และ PH จะทำการตรวจสอบค่าและส่งผ่านบอร์ด Arduino ไปเก็บไว้ใน SD Card ต่อ Arduino จะทำการประมวลผลข้อมูลของ Senser PH เพื่อทำการหาค่า PH เพื่อทำการสั่งให้ Solenoid ทำงานตามคำสั่งในโปรแกรม Arduino เมื่อเสร็จก็จะส่งข้อมูล PH กับไปเก็บใน SD Card และบอร์ด Arduino จะทำการประมวลผลข้อมูลของ Senser Oxygen เพื่อที่จะสั่งให้เบรกเกอร์ไฟ 3 เฟส ทำงานโดยจะสั่งผ่านบอร์ด Arduino และจะทำการส่งข้อมูลไปเก็บไว้ที่ SD card ต่อไป

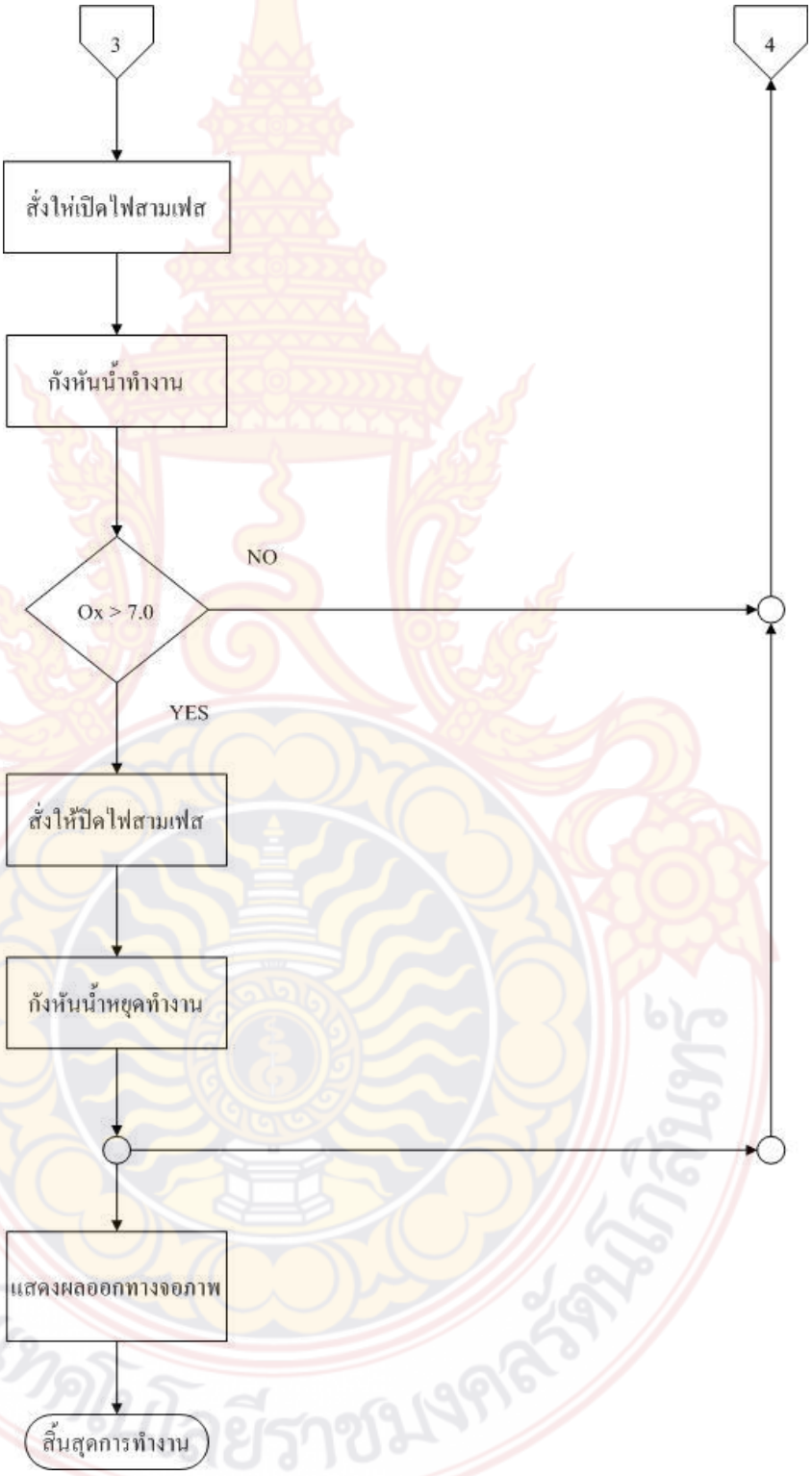
2. ผังระบบ (Flow Chart) รวม



ภาพที่ 22 ผังระบบ (Flow Chart) รวม (ต่อ)



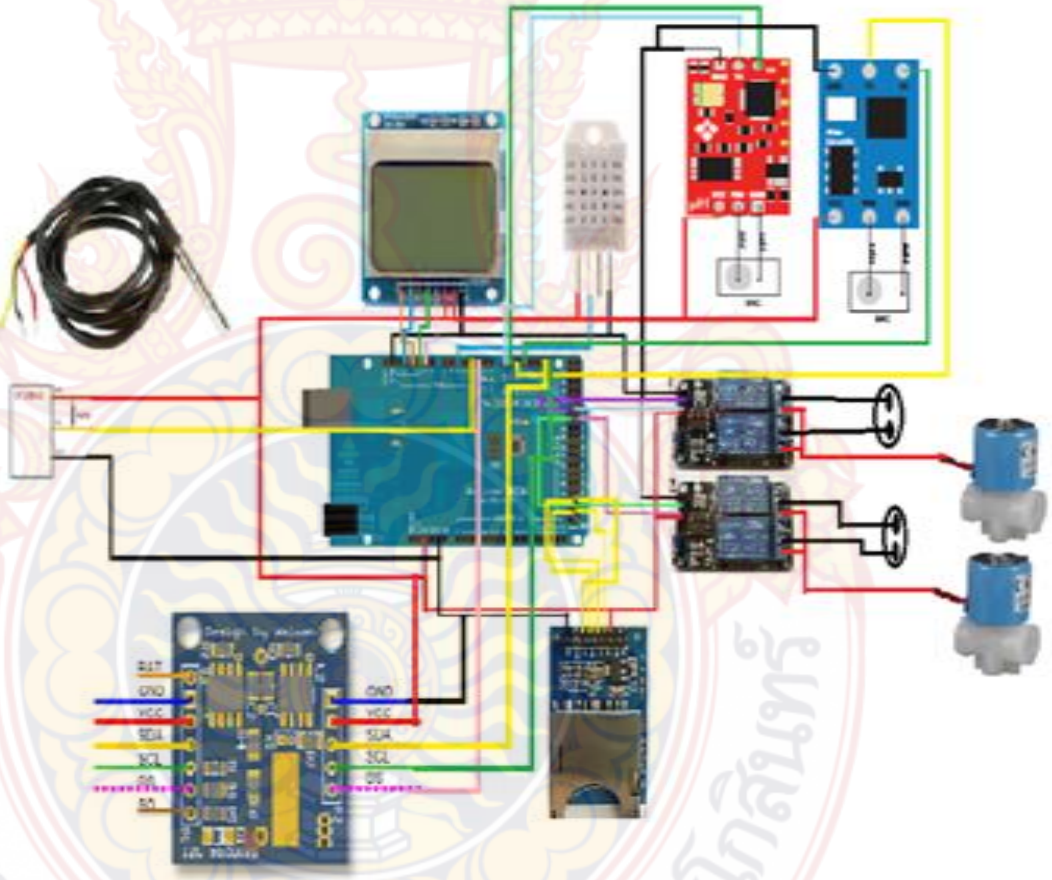
ภาพที่ 22 ผังระบบ (Flow Chart) รวม (ต่อ)



ภาพที่ 22 ผังระบบ (Flow Chart) รวม (ต่อ)

จากภาพที่ 22 สามารถอธิบายการทำงานได้ดังนี้เริ่มต้นการทำงานของวงจรรวมเริ่มต้นจากการทำงานของ Senser PH ทำการวัดค่า PH และข้อมูลจะถูกส่งมาที่บอร์ด Arduino จะทำการประมวลข้อมูลของ Senser PH เพื่อทำการค่า PH ว่ามีค่าเท่าไร ถ้าค่าของ PH มีค่าน้อยกว่า 6.5 Solenoid ตัวที่ 1 จะทำการปล่อยน้ำเค็ม แต่ Solenoid ตัวที่ 2 จะไม่ทำการปล่อยน้ำใดๆ แต่ตรวจสอบว่าค่า PH นั้นมีค่ามากกว่า 9.0 Solenoid ตัวที่ 1 จะไม่ทำการปล่อยน้ำเค็ม แต่ Solenoid ตัวที่ 2 จะทำการปล่อยน้ำปูนขาว

3. **การพัฒนาและออกแบบอุปกรณ์**



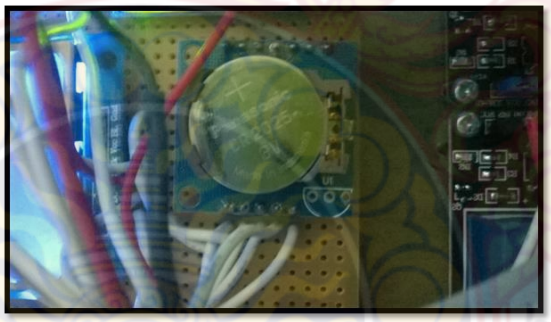
ภาพที่ 23 วงจรรวมทั้งหมด

3.1 การต่อวงจรรวมของอุปกรณ์ที่ใช้ในการทำวิจัย

3.1.1 การต่อ SD card เริ่มต้นจากพอร์ต +5 vcc เข้าที่ pin +5 vcc พอร์ต cs เข้าที่ pin d53 พอร์ต mosi เข้าที่ pin d51 พอร์ต sck เข้าที่ pin d52 พอร์ต miso เข้าที่ pin d50 พอร์ต Gnd เข้าที่ pin Gnd ของบอร์ด Arduino

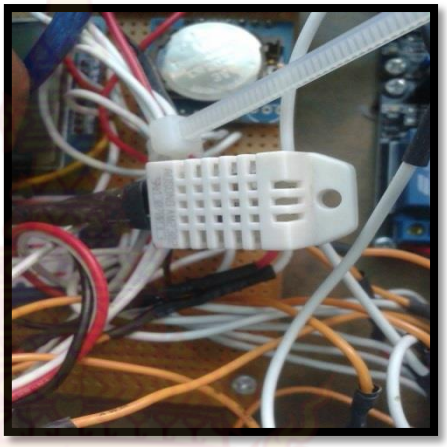
ภาพที่ 24 ติดตั้ง SD Card สำหรับการเก็บข้อมูล

3.1.2. การต่อ Tiny RTC I2C เริ้มต้นจากพอร์ต ds เข้าที่ pin 2 พอร์ต SCL เข้าที่ pin a5 พอร์ต SDL เข้าที่ pin a4 พอร์ต Vcc เข้าที่ pin +5 vcc พอร์ต Gnd เข้าที่ pin Gnd ของบอร์ด Arduino



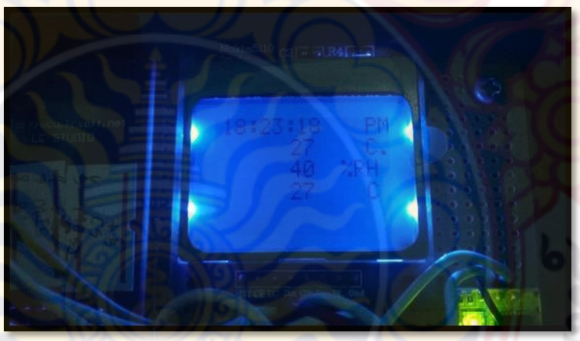
ภาพที่ 25 ติดตั้ง Real Time Clock

3.1.3 การต่อเซ็นเซอร์วัดค่าความชื้น AM2302เริ้มต้นจากพอร์ต vcc เข้าที่ pin +5vcc พอร์ต data เข้าที่ pin A5 พอร์ต Gnd เข้าที่ pin Gnd ของบอร์ด Arduino



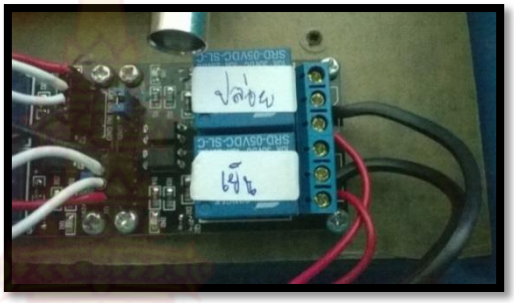
ภาพที่ 26 ติดตั้งเซ็นเซอร์วัดค่าความชื้น AM2302

3.1.4 จอภาพ Nokia 5110เริ้มต้นจากพอร์ต Gnd เข้าที่ pin Gnd พอร์ต vcc เข้าที่ pin +5 vcc พอร์ต clk เข้าที่ pin 8 พอร์ตdin เข้าที่ pin 9 พอร์ต dc เข้าที่ pin 10 พอร์ต ce เข้าที่ pin 12 พอร์ต rst เข้าที่ pin 11 ของบอร์ด Arduino



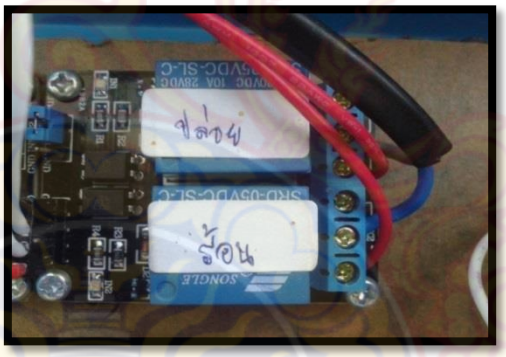
ภาพที่ 27 ติดตั้งจอแสดงผลกราฟิก ET-NOKIA LCD 5110Nokia

3.1.6 Relay Module 1 เริ่มต้นจากพอร์ต vcc เข้าที่ pin +5 vcc พอร์ต Int เข้าที่ pin 30 พอร์ต Int 2 เข้าที่ pin 31 พอร์ต Gnd เข้าที่ pin Gnd



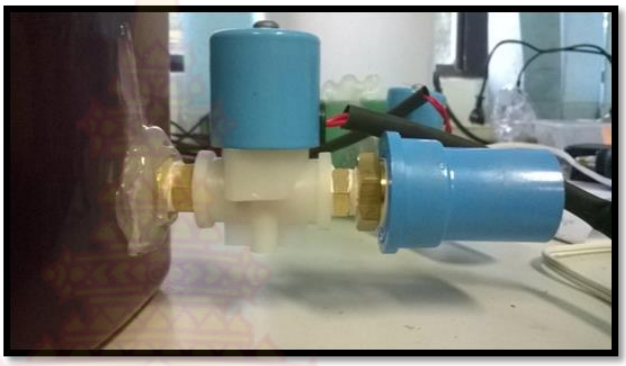
ภาพที่ 28 ติดตั้ง Relay Module ตัวที่ 1

3.1.7. Relay Module 2 เริ่มต้นจากพอร์ต vcc เข้าที่ pin +5 vcc พอร์ต Int เข้าที่ pin 32 พอร์ต Int 2 เข้าที่ pin 33 พอร์ต Gnd เข้าที่ pin Gnd



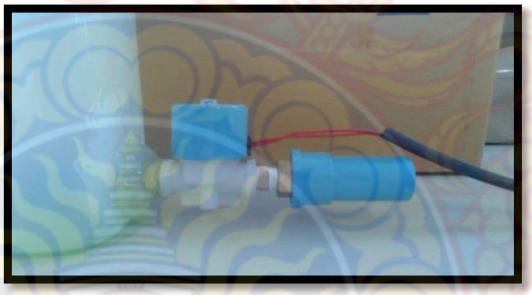
ภาพที่ 29 ติดตั้ง Relay Module ตัวที่ 2

3.1.8. Solenod 1 เริ่มต้นจาก สายเส้นที่ 1 ของsolenod 1 เข้าที่ relay 1 ชุดที่ 1 พอร์ต 1 สายเส้นที่ 2 เข้าที่ relay 1 ชุดที่ 2 พอร์ต 2 และปลั๊กไฟ 220 v สายเส้นที่ 1 เข้าที่ relay 1 ชุดที่ 1 พอร์ต 2 สายเส้นที่ 2 เข้าที่ relay 1 ชุดที่ 2 พอร์ต 2



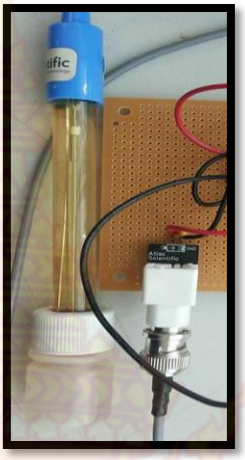
ภาพที่ 30 ติดตั้ง SOLENOID VALVE (โซลินอยด์) 2 หุน 220V ตัวที่ 1

3.1.9 Solenod เริ่มต้นจาก สายเส้นที่ 1 ของsolenod 2 เข้าที่ relay 2 ชุดที่ 1 พอร์ต 1 สายเส้นที่ 2 เข้าที่ relay 2 ชุดที่ 2 พอร์ต 2 และปลั๊กไฟ 220 v สายเส้นที่ 1 เข้าที่ relay 2 ชุดที่ 1 พอร์ต 2 สายเส้นที่ 2 เข้าที่ relay 2 ชุดที่ 2 พอร์ต 2



ภาพที่ 31 ติดตั้ง SOLENOID VALVE (โซลินอยด์) 2 หุน 220V ตัวที่ 2

3.1.10 วงจรเซนเซอร์วัดค่า PH เริ่มต้นจาก Off Amp Op07 =2 ตัว R 100K = 1ตัว R รอบ 0-100K 1ตัวR 10K =4 ตัว R 20K =2 ตัว IC เซนามิค 4.7 nf 3 ตัว IC เซนามิค 1.68 uf 1 ตัว OffAmp MCp 6022 1 ตัว หม้อแปลง +15-15v Data ขา AO Vcc 5v GND ต่อเข้า GND



ภาพที่ 32 ติดตั้งเซนเซอร์ PH

4. ขั้นตอนการทดลอง

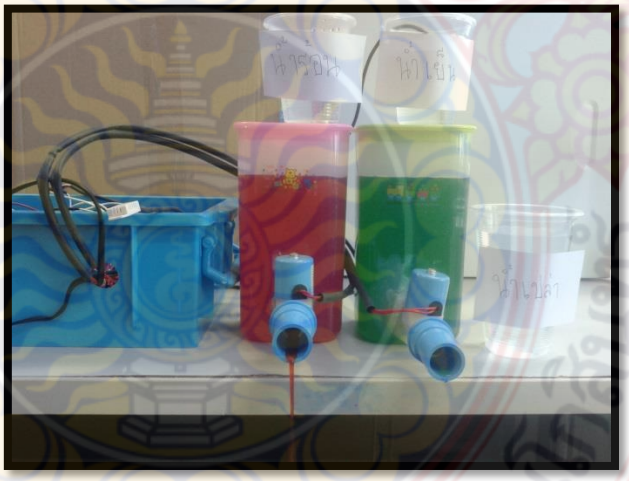
4.1 ตรวจสอบการทำงานของ SOLENOID VALVE

ภาพที่ 33 เริ่มต้นตรวจสอบการทำงานของ SOLENOID VALVE

4.2 การทดลองอุณหภูมิที่ต่ำกว่า 25 องศา

ภาพที่ 34 อุณหภูมิต่ำ SOLENOID VALVE 1 ทำงาน

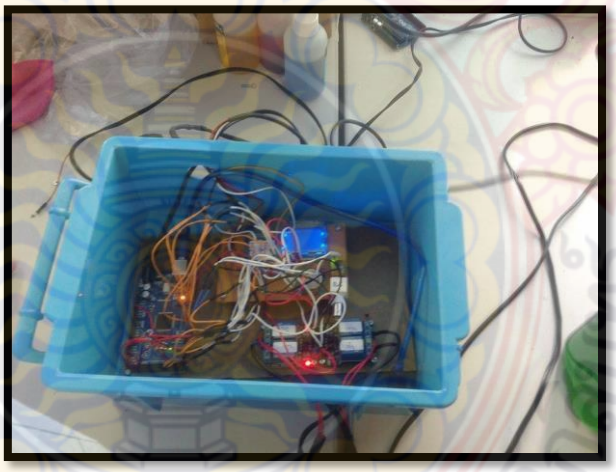
4.3 การทดลองอุณหภูมิที่สูงกว่า 30 องศา



ภาพที่ 35 อุณหภูมิสูง SOLENOID VALVE 2 ทำงาน

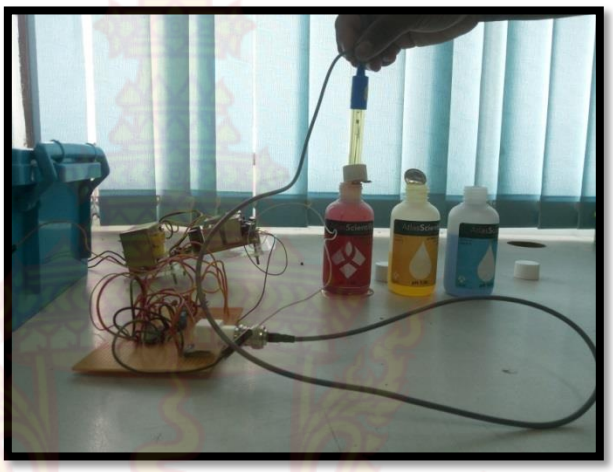
4.4 การทดลองอุณหภูมิที่ปกติระหว่าง 25-30 องศา SOLENOID VALVE จะไม่ทำงานทั้งคู่

ภาพที่ 36 อุณหภูมิปกติ SOLENOID VALVE ไม่ทำงาน



ภาพที่ 37 วงจรรวมการควบคุม SOLENOID

4.5 การทดลองค่า PH ของสารละลายที่มีค่า PH 4.00



ภาพที่ 38 ทดลองค่าสารละลายที่มีค่า PH 4.00

4.6 การทดลองค่า PH ของสารละลายสารละลายที่มีค่า PH 7.00

ภาพที่ 39 ทดลองค่าสารละลายที่มีค่า PH 7.00

4.7 การทดลองค่า PH ของสารละลายสารละลาย

ตารางที่ 1 แสดงผลการทดลองการปิดเปิดน้ำของ SOLENOID VALVE โดยอ้างอิงจากเซนเซอร์วัดอุณหภูมิน้ำ

ครั้งที่	อุณหภูมิน้ำเย็น	SOLENOID ทำงาน	อุณหภูมิน้ำร้อน	SOLENOID ทำงาน
1	24	✓	45	✓
2	20	✓	30	✓
3	18	✓	33	✓
4	16	✓	37	✓
5	20	✓	35	✓
6	10	✓	39	✓
7	19	✓	40	✓
8	23	✓	32	✓
9	15	✓	31	✓
10	17	✓	34	✓

จากตารางการทดลองจะแสดงให้เห็นว่าเมื่อ Senser วัดอุณหภูมิทำการวัดค่าถ้ามีอุณหภูมิต่ำกว่า 24 องศาจะสั่งให้ SOLENOID ทำการปล่อยน้ำสีเขียวลงไปเพื่อทำให้น้ำมีค่าปกติแต่ถ้า Senser วัดอุณหภูมิวัดค่าแล้วมีค่าสูงกว่า 30 องศา จะทำการปล่อยน้ำสีแดงลงไปเพื่อทำให้น้ำมีค่าปกติและ Senser วัดค่าอุณหภูมิแล้วมีค่าระหว่าง 24-30 องศา จะสั่งให้ SOLENOID นั้นไม่ทำงานทั้งสองตัวเพราะน้ำมีค่าปกติแล้ว

5. สรุปผลการทดลอง

ผลการทดลองจะเห็นได้ว่าผลที่ออกมาเป็นที่น่าพอใจการทำงานของ SOLENOID นั้นทำงานได้อย่างปกติไม่มีติดขัดใดๆจึงแสดงให้เห็นได้ว่า Snser วัดอุณหภูมิและ SOLENOID นั้นทำงานได้ปกติและปล่อยน้ำได้ตามอุณหภูมิที่กำหนดไว้

บทที่ 4
ผลการวิเคราะห์ข้อมูล

การวิจัยนี้มีวัตถุประสงค์ที่จะพัฒนาระบบควบคุมค่าความเป็นกรด-ด่างแบบอัตโนมัติ เพื่อศึกษาประสิทธิ์ภาพของระบบ โดยมีผลการวิจัยดังนี้

1. ผลการวิเคราะห์ประสิทธิภาพของระบบ ตามสมมุติฐานการวิจัยข้อที่ 1

สมมุติฐานการวิจัยข้อที่ 1 ระบบมีค่าประสิทธิ์ภาพของพัฒนาระบบควบคุมค่าความเป็นกรด-ด่างแบบอัตโนมัติ มีค่าไม่ต่ำกว่าร้อยละ 80 ผลการวิเคราะห์ข้อมูลดังปรากฏในตารางที่ 4.1 แสดงการรับค่าจากเซ็นเซอร์ทุก ๆ 3 วินาที จำนวน 100 รอบ โดยวัดอุณหภูมิอากาศ คิดเป็นร้อยละ 98 อุณหภูมิในน้ำ คิดเป็นร้อยละ 99 ความเร็วลม คิดเป็นร้อยละ 100 ทิศทางลม คิดเป็นร้อยละ 100 ปริมาณน้ำฝน คิดเป็นร้อยละ 95 ความเข้มของแสง คิดเป็นร้อยละ 100 และ ค่า PH ในน้ำ คิดเป็นร้อยละ 100

ตารางที่ 2 ตารางแสดงผลการทดสอบการรับค่าจากเซ็นเซอร์ต่าง ๆ

เซนเซอร์	ตรวจจับค่า (รอบ)	ไม่ตรวจจับค่า (รอบ)	เปอร์เซ็นการตรวจจับค่า
อุณหภูมิในน้ำ	100	1	99 %
อุณหภูมิอากาศ	100	2	98 %
ความเร็วลม	100	-	100 %
ทิศทางลม	100	-	100 %
ปริมาณน้ำฝน	100	5	95%
ความเข้มของแสง	100	-	100 %
ค่า PH ในน้ำ	100	-	100 %

จากนั้นนำข้อมูลที่ได้ทดสอบการบันทึกลงฐานข้อมูลจำนวน 100 ครั้ง ของทุกเซ็นเซอร์ ตามตารางที่ 4.2 สรุปผลการทดลองดังนี้ บันทึกอุณหภูมิอากาศลงฐานข้อมูลถูกต้อง คิดเป็นร้อยละ 100 บันทึกอุณหภูมิในน้ำลงฐานข้อมูลถูกต้อง คิดเป็นร้อยละ 100 บันทึกความเร็วลมลงฐานข้อมูลถูกต้อง คิดเป็นร้อยละ 100 บันทึกทิศทางลมลงฐานข้อมูลถูกต้อง คิดเป็นร้อยละ 100

บันทึกปริมาณน้ำฝนลงฐานข้อมูลถูกต้อง คิดเป็นร้อยละ 95 บันทึกความเข้มของแสงลงฐานข้อมูลถูกต้อง คิดเป็นร้อยละ 100 และ บันทึกค่า PH ในน้ำลงฐานข้อมูลถูกต้อง คิดเป็นร้อยละ 100

ตารางที่ 3 ตารางแสดงผลการทดสอบการบันทึกลงฐานข้อมูล

เซนเซอร์	รับข้อมูล	ไม่รับข้อมูล
อุณหภูมิในน้ำ	100	-
อุณหภูมิอากาศ	100	-
ความเร็วลม	100	-
ทิศทางลม	100	-
ปริมาณน้ำฝน	100	-
ความเข้มของแสง	100	-
ค่า PH ในน้ำ	100	-

นำระบบควบคุมค่าความเป็นกรด-ด่างแบบอัตโนมัติ ทำการทดสอบควบคุมค่า PH ให้อยู่ในช่วง 7.0-7.5 โดยการติดตั้งระบบควบคุมค่าความเป็นกรด-ด่างแบบอัตโนมัติ กับบ่อกุ้งจำลองเป็นตู้ปลาขนาด 48 นิ้ว มีถังสารละลาย 2 ถัง ใช้โซลีนอยด์วาล์วควบคุมการไหลของสารละลายปรับค่า PH ของน้ำในตู้ปลาเป็นค่าต่าง ๆ ค่าละ 10 ครั้ง ตามตารางที่ 4.3 สรุปผลการทดลองดังนี้ เมื่อค่า PH ของน้ำในตู้ปลาต่ำกว่า 7.0 ระบบควบคุมค่าความเป็นกรด-ด่างแบบอัตโนมัติ สั่งโซลีนอยด์วาล์วของถังสารละลายเพิ่มค่า PH ทำงาน และโซลีนอยด์วาล์วของถังสารละลายลดค่า PH ไม่ทำงาน คิดเป็นร้อยละ 100 ค่า PH ของน้ำในตู้ปลามีค่าระหว่าง 7.0-7.5 ระบบควบคุมค่าความเป็นกรด-ด่างแบบอัตโนมัติ สั่งโซลีนอยด์วาล์วของถังสารละลายเพิ่มค่า PH ไม่ทำงาน และโซลีนอยด์วาล์วของถังสารละลายลดค่า PH ไม่ทำงาน คิดเป็นร้อยละ 100 ค่า PH ของน้ำในตู้ปลาสูงกว่า 7.5 ระบบควบคุมค่าความเป็นกรด-ด่างแบบอัตโนมัติ สั่งโซลีนอยด์วาล์วของถังสารละลายเพิ่มค่า PH ไม่ทำงาน และโซลีนอยด์วาล์วของถังสารละลายลดค่า PH ทำงาน คิดเป็นร้อยละ 100 ดังนั้นระบบควบคุมค่าความเป็นกรด-ด่างแบบอัตโนมัติ ทำงานได้ถูกต้องคิดเป็นร้อยละ 100

ตารางที่ 4 ตารางแสดงผลระบบควบคุมค่าความเป็นกรด-ด่างแบบอัตโนมัติ ทำการทดสอบควบคุมค่า PH ของน้ำให้อยู่ในช่วง 7.0-7.5

ค่า PH	สารละลายลดค่า PH		สารละลายเพิ่มค่า PH	
	ทำงาน (ครั้ง)	ไม่ทำงาน (ครั้ง)	ทำงาน (ครั้ง)	ไม่ทำงาน (ครั้ง)
6.5	-	10	10	-
6.6	-	10	10	-
6.7	-	10	10	-
6.8	-	10	10	-
6.9	-	10	10	-
7.0	-	10	-	10
7.1	-	10	-	10
4.2	-	10	-	10
7.3	-	10	-	10
7.4	-	10	-	10
7.5	-	10	-	10
7.6	10	-	-	10
7.7	10	-	-	10
7.8	10	-	-	10
7.9	10	-	-	10
8.0	10	-	-	10

2. สรุปผลการวิเคราะห์ข้อมูล

ดังนั้นจึงสามารถสรุปผลการวิเคราะห์ได้ว่า ระบบควบคุมค่าความเป็นกรด-ด่างแบบอัตโนมัติ มีผลวิเคราะห์ประสิทธิ์ภาพของระบบควบคุมค่าความเป็นกรด-ด่างแบบอัตโนมัติ มีค่าไม่ต่ำกว่าร้อยละ 100 ซึ่งสอดคล้องกับสมมติฐานงานวิจัยในข้อที่ 1 ที่ได้กำหนดไว้ไม่น้อยกว่าร้อยละ 80

บทที่ 5
สรุป อภิปรายผลและข้อเสนอแนะ

การวิจัยนี้มุ้งเน้นการพัฒนาระบบควบคุมค่าความเป็นกรด-ด่างแบบอัตโนมัติ โดยนำไปจำลองทำการทดลองใช้กับตู้ปลาขนาด 42 นิ้ว จำลองบ่อเลี้ยงกุ้ง เพื่อหาข้อสรุปพัฒนาระบบควบคุมค่าความเป็นกรด-ด่างแบบอัตโนมัติ ดังนี้

1. สรุปผลการทำวิจัย

การพัฒนาระบบควบคุมค่าความเป็นกรด-ด่างแบบอัตโนมัติในอุตสาหกรรมการเลี้ยงกุ้ง เป็นการพัฒนาควบคุมค่าความเป็นกรด-ด่าง เพื่อช่วยลดการเกิดโรคหรือการตายของกุ้ง ระบบควบคุมค่าความเป็นกรด-ด่าง มีส่วนประกอบ 4 ส่วนคือ 1) ชุด Senser วัดค่าความเป็นกรด-ด่าง 2) ไมโครคอลโทรลเลอร์ (Arduino) 3) ชุดจอภาพ 4) ชุดควบคุมสารเพิ่ม ลดค่าความเป็นกรด-ด่าง โดยระบบควบคุมค่าความเป็นกรด-ด่างในบ่อเลี้ยงกุ้งนั้นจะถูกติดตั้งไว้ที่บ่อเลี้ยงกุ้งจำลองตลอดเวลา เพื่อจะทำการวัดค่าความเป็นกรด-ด่างและทำการปรับเปลี่ยนค่าความเป็นกรด-ด่างของน้ำให้คงสภาพตามปรกติ

สรุปผลวิจัยมีดังนี้

1.1 สมมุติฐานการวิจัยข้อที่ 1 ระบบมีค่าประสิทธิ์ภาพของพัฒนาระบบควบคุมค่าความเป็นกรด-ด่างแบบอัตโนมัติ มีค่าไม่ต่ำกว่าร้อยละ 100

2. อภิปรายผลการวิจัย

การวิจัยนี้เป็นการศึกษาประสิทธิภาพ พัฒนาระบบควบคุมค่าความเป็นกรด-ด่างแบบอัตโนมัติ มีค่าไม่ต่ำกว่าร้อยละ 100

เมื่อพิจารณาจากระดับประสิทธิภาพของระบบควบคุมค่าความเป็นกรด-ด่าง แบบอัตโนมัติ ที่สูงกว่าเกณฑ์ที่กำหนดไว้ในสมมุติฐานข้อที่ 1 อาจมาจากสาเหตุหลายประการ ได้แก่ กระบวนการออกแบบ การพัฒนา การทดสอบ รวมไปถึงการนำระบบควบคุมค่าความเป็นกรด-ด่างแบบอัตโนมัติ มาทดสอบในแบบจำลองโดยใช้ตู้ปลาขนาด 42 นิ้ว ซึ่งเป็นการทดสอบในแบบจำลอง ตัวอุปกรณ์ไม่ได้รับผลกระทบจากสภาพอากาศที่บ่อเลี้ยงกุ้ง จึงทำงานได้อย่างถูกต้อง หากระบบควบคุมค่าความเป็นกรด-ด่าง แบบอัตโนมัติถูกนำไปทดสอบในพื้นที่บ่อเลี้ยงกุ้งจริงอาจทำงานได้ถูกต้องร้อยละ 100 แค่ช่วงแรก ๆ ทำให้ต้องศึกษาประสิทธิภาพของระบบควบคุมค่าความเป็นกรด-ด่างแบบอัตโนมัติเป็นระยะเวลาพอสมควร

3. ข้อเสนอแนะ

การวิจัยนี้มีวัตถุประสงค์ เพื่อศึกษาประสิทธิภาพระบบควบคุมค่าความเป็นกรด-ด่างแบบอัตโนมัติ มีข้อเสนอแนะดังนี้

3.1 นำระบบควบคุมค่าความเป็นกรด-ด่างแบบอัตโนมัติไปทดลองในสถานที่จริง

บรรณานุกรม

การวัด pH ของน้ำด้วยเครื่องวัด pH [Online] from http://www.ginkosea.com/article/9-ph-Indicator accessed September 21, 2019

คมน์ ศิลปาจารย์. 2538. การจำแนกและแบ่งเขตการเลี้ยงกุ้งทะเลจังหวัดประจวบคีรีขันธ์. ในรายงานการสัมมนาวิชาการประจำปี 2538, วันที่ 18 –20 กันยายน 2538. กรมประมง, กรุงเทพฯ. น. 560 – 564.

ทัศนีย์ นลวชัย,วัชริยา ภรีวิโรจน์กุล , นิติ ชเชิดู, เกศินี หลายสุทธิสาร และ ชลอ ลิ้มสุวรรณ. ผลของระดับออกซิเจน แอมโมเนีย และ พีเอชต่อการกินอาหารของกุ้งขาวแวนนาไม วารสารวิจัยเทคโนโลยีการประมง ปี ที่ 6 ฉบับที่ 1 มกราคม – มิถุนายน 2555

พีเอช (เคมี) [Online] from https://th.wikipedia.org/wiki/พีเอช_(เคมี) accessed September 21, 2019

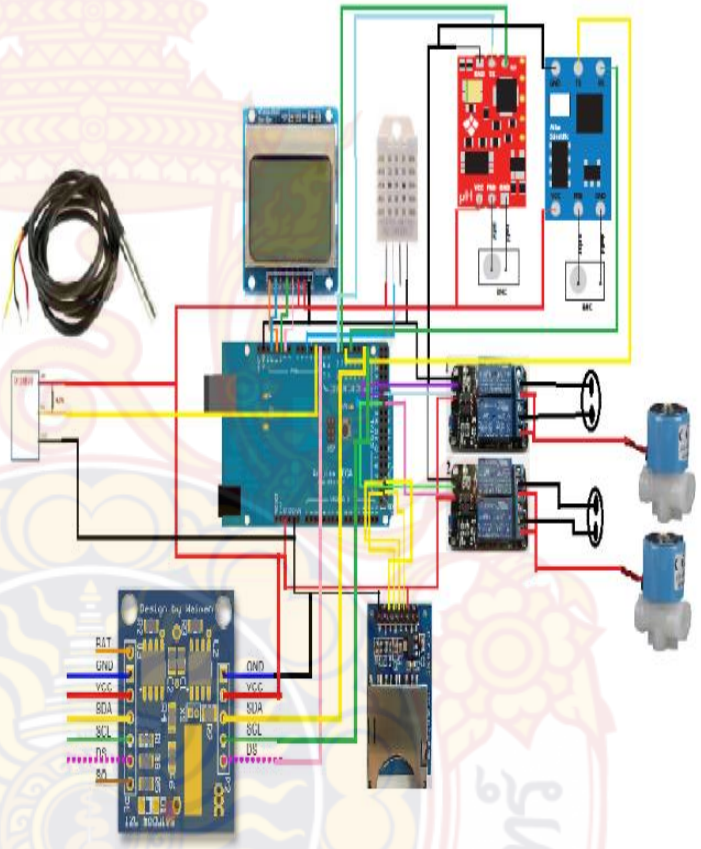
พื้นฐานไมโครคอนโทรลเลอร์ด้วย [Online] from Arduino http://arduino-thai.blogspot.com/2013/05/arduino.html accessed September 21, 2019

ศิริชัย เติมโชคเกษม และ สุรเชษฐ์ โทวราภา. ตู้ปลาอัจฉริยะ ECTI-CARD 2010, May'10, Pattaya, Thailand. ISBN: 978-974-8242-54-5

ภาคผนวก ก

วงจร โปรแกรม และตัวอย่างโปรแกรม

วงจรรวมทั้งหมดของโปรแกรม



โค๊ดรวมของวงจรทั้งหมด

```
//LCD
#include <LCD5110_Graph.h>

LCD5110 myGLCD(8,9,10,11,12);

extern unsigned char SmallFont[];

int y;
//


  #include <SD.h>

File myFile;


  String inputstring = "";
  String sensorstring = "";
String sensorstringcopy = "";   //a string
  boolean input_stringcomplete = false;
  boolean sensor_stringcomplete = false;


//one
#include <OneWire.h>

    OneWire  ds(3);  // on pin 10 (a 4.7K
  float olecelsius=0;
  //one

  //
#include <Wire.h>
#include "RTClib.h"

RTC_DS1307 rtc;
int oleminute;
```

```
int Relay1 = 30;
int Relay2 = 31;
int Relay3 = 32;
int Relay4 = 33;
// the setup routine runs
//

//am
const byte DATA_PIN = 5;
byte data[5];
float oletemp=0;
float olerh=0;


int Relay1 = 30;
int Relay2 = 31;
int Relay3 = 32;
int Relay4 = 33;
// the setup routine runs
//

//am
const byte DATA_PIN = 5;
byte data[5];
float oletemp=0;
float olerh=0;


  void setup(){      //set up the hardware
    //
// initialize the digital pin as an output.
pinMode(Relay1, OUTPUT);
pinMode(Relay2, OUTPUT);
pinMode(Relay3, OUTPUT);
pinMode(Relay4, OUTPUT);
//
     //LCD
  myGLCD.InitLCD();
  myGLCD.setFont(SmallFont);

  //
    //am
    pinMode( DATA_PIN, INPUT );
  digitalWrite( DATA_PIN, HIGH ); // enable internal pull-up

//am
```

```
    Serial.begin(9600);
    Serial3.begin(38400);
    inputstring.reserve(5);
    sensorstring.reserve(30);

  //
#ifdef AVR
  Wire.begin();
#else
  Wirel.begin(); // Shield I2C pins connect to
#endif
  rtc.begin();

  if (! rtc.isrunning()) {
    Serial.println("RTC is NOT running!");
    // following line sets the RTC to the date
    rtc.adjust(DateTime(__DATE__, __TIME__));
  }


    while (!Serial) {
    ; // wait for serial port to connect. Nee
  }
  pinMode(53, OUTPUT);
  if (!SD.begin(53)) {
    Serial.println("initialization failed!");
    return;
  }
  Serial.println("initialization done.");


}
  void serialEvent() {
            char inchar = (char)Serial.read();
            inputstring += inchar;
            if(inchar == '\r') {input_stringcomplete = true;}
            }
  void serialEvent3(){
            char inchar = (char)Serial3.read();
            sensorstring += inchar;
            if(inchar == '\r') {sensor_stringcomplete = true;}
           }
void loop()  {
```

```
 if (input_stringcomplete){
     Serial3.print(inputstring);
     inputstring = "";
     input_stringcomplete = false;
     }

if (sensor_stringcomplete){       //if a
     sensorstringcopy = "";
    sensorstringcopy=sensorstring;
     Serial.println(sensorstringcopy);
     Serial.println(sensorstring);
     sensorstring = "";
     sensor_stringcomplete = false;
}

byte itw;
 byte present = 0;
 byte type_s;
 byte datatw[12];
 byte addr[8];
 float celsius, fahrenheit;

 if ( !ds.search(addr)) {
   Serial.println("No more addresses.");
   Serial.println();
   ds.reset_search();
   //delay(250);
   return;
 }

switch (addr[0]) {
  case 0x10:
    Serial.println("  Chip = DS18S20");  // or old DS1820
    type_s = 1;
    break;
  case 0x28:
    Serial.println("  Chip = DS18B20");
    type_s = 0;
    break;
  case 0x22:
    Serial.println("  Chip = DS1822");
    type_s = 0;
    break;
  default:
    Serial.println("Device is not a DS18x20 family device.");
    return;
}
```

```
ds.reset();
ds.select(addr);
ds.write(0x44, 1);
present = ds.reset();
ds.select(addr);
ds.write(0xBE);

 for ( itw = 0; itw < 9; itw++) {
   datatw[itw] = ds.read();

 int16_t raw = (datatw[1] << 8) | datatw[0];
 if (type_s) {
   raw = raw << 3; // 9 bit resolution default
   if (datatw[7] == 0x10) {
     // "count remain" gives full 12 bit resolution
     raw = (raw & 0xFFF0) + 12 - datatw[6];
   }
 } else {
   byte cfg = (datatw[4] & 0x60);
   // at lower res, the low bits are undefined, so let's zero them
   if (cfg == 0x00) raw = raw & ~7;  // 9 bit resolution, 93.75 ms
   else if (cfg == 0x20) raw = raw & ~3; // 10 bit res, 187.5 ms
   else if (cfg == 0x40) raw = raw & ~1; // 11 bit res, 375 ms
   //// default is 12 bit resolution, 750 ms conversion time
 }

 celsius = (float)raw / 16.0;
 fahrenheit = celsius * 1.8 + 32.0;
 Serial.print("  Temperature = ");
 Serial.print(celsius);
 olecelsius=celsius;
 Serial.print(" Celsius, ");
 Serial.print(fahrenheit);
 Serial.println(" Fahrenheit");
 Serial.print("  Temperature = ");
 Serial.print(olecelsius);
 Serial.println(" Celsius, ");
```

```
{
int count = 0;
 byte it=0, j=0;
 byte new_state, state = HIGH;

 for (byte x=0; x < 5; x++) {
   data[x] = 0; // clear data buffer
 }
 pinMode( DATA_PIN, OUTPUT ); // change direction to output
 digitalWrite( DATA_PIN, LOW ); // output low (send the start bit)
 delayMicroseconds( 1000 );
 digitalWrite( DATA_PIN, HIGH ); // output high
 delayMicroseconds( 40 );
 pinMode( DATA_PIN, INPUT ); // change direction to input
 digitalWrite( DATA_PIN, HIGH ); // enable internal pull-up


 unsigned long tl, t0 = micros();
 while (1) {
    new_state = digitalRead( DATA_PIN );
    if ( state != new_state ) {
       tl = micros();
       if ( (state == HIGH) && (it > 2) ) {
          byte b = ( (tl-t0) > 40 ) ?  1 : 0;
          data[j/8] <<= 1;
          data[j/8] |= b;
          j++;
       }
       it++;
       state = new_state;
       t0 = tl;
       count = 0;
    } else {
       count++;
       if ( count > 1000 )
          break;
    }
 }
 byte check_sum = 0x00;
 for (byte x=0; x < 4; x++) {
   check_sum += data[x];
 }
```

```
  if ( check_sum != data[4] ) {
     Serial.println( "CHECKSUM error" );
  } else if(((data[0] << 8) | data[1])/10.0 != 0)
  {
     Serial.print( ((data[0] << 8) | data[1])/10.0 );
     olerh=((data[0] << 8) | data[1])/10.0;
     Serial.print( "%RH, " );
     Serial.print( ((data[2] << 8) | data[3])/10.0 );
     oletemp=((data[2] << 8) | data[3])/10.0 ;
     Serial.println( " C" );
  }
  //
  if ( olecelsius==25) // ปล่อยน้ำร้อน
 {
   digitalWrite(Relay1, HIGH); // turn the RELAY on(HIGH is the voltage level)
digitalWrite(Relay2, HIGH);

digitalWrite(Relay3, LOW); // turn the RELAY off by making the voltage LOW
digitalWrite(Relay4, LOW);
delay (10000);
//stop
digitalWrite(Relay1, HIGH); // turn the RELAY off by making the voltage LOW
digitalWrite(Relay2, HIGH);
digitalWrite(Relay3, HIGH); // turn the RELAY off by making the voltage LOW
digitalWrite(Relay4, HIGH);
delay (10000);
 }
```

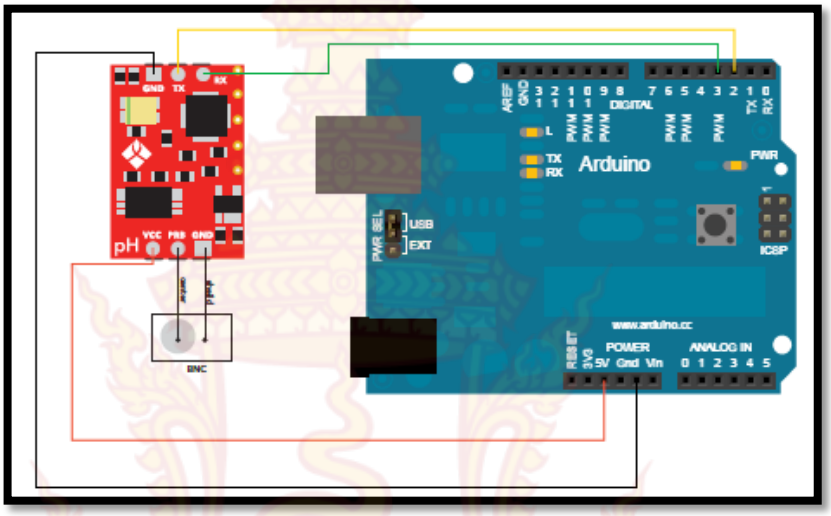
```
 if ( olecelsius==26) // ปล่อยน้ำร้อน
 {
   digitalWrite(Relay1, HIGH); // turn the RELAY on(HIGH is the voltage level)
digitalWrite(Relay2, HIGH);

digitalWrite(Relay3, LOW); // turn the RELAY off by making the voltage LOW
digitalWrite(Relay4, LOW);
delay (5000);
//stop
digitalWrite(Relay1, HIGH); // turn the RELAY off by making the voltage LOW
digitalWrite(Relay2, HIGH);
digitalWrite(Relay3, HIGH); // turn the RELAY off by making the voltage LOW
digitalWrite(Relay4, HIGH);
delay (10000);
 }

 if ( olecelsius<25) // ปล่อยน้ำร้อน
 {
   digitalWrite(Relay1, HIGH); // turn the RELAY on(HIGH is the voltage level)
digitalWrite(Relay2, HIGH);

digitalWrite(Relay3, LOW); // turn the RELAY off by making the voltage LOW
digitalWrite(Relay4, LOW);

 }

 if (  olecelsius==29)//ปล่อยน้ำเย็น
 {

digitalWrite(Relay1, LOW); // turn the RELAY off by making the voltage LOW
digitalWrite(Relay2, LOW);

digitalWrite(Relay3, HIGH); // turn the RELAY on(HIGH is the voltage level)
digitalWrite(Relay4, HIGH);
delay (5000);
//stop
digitalWrite(Relay1, HIGH); // turn the RELAY off by making the voltage LOW
digitalWrite(Relay2, HIGH);
digitalWrite(Relay3, HIGH); // turn the RELAY off by making the voltage LOW
digitalWrite(Relay4, HIGH);
delay (10000);

 }
```

```
 if (olecelsius==30)//ปล่อยน้ำเย็น
 {

digitalWrite(Relay1, LOW); // turn the RELAY off by making the voltage LOW
digitalWrite(Relay2, LOW);

digitalWrite(Relay3, HIGH); // turn the RELAY on(HIGH is the voltage level)
digitalWrite(Relay4, HIGH);
delay (10000);
//stop
digitalWrite(Relay1, HIGH); // turn the RELAY off by making the voltage LOW
digitalWrite(Relay2, HIGH);
digitalWrite(Relay3, HIGH); // turn the RELAY off by making the voltage LOW
digitalWrite(Relay4, HIGH);
delay (10000);

 }


if (  olecelsius>30)//ปล่อยน้ำเย็น
 {
digitalWrite(Relay1, LOW); // turn the RELAY off by making the voltage LOW
digitalWrite(Relay2, LOW);
digitalWrite(Relay3, HIGH); // turn the RELAY on(HIGH is the voltage level)
digitalWrite(Relay4, HIGH);
 }
 if (  olecelsius>=27&&olecelsius<=28)// อุณหภูมิปรกติ
 {
digitalWrite(Relay1, HIGH); // turn the RELAY off by making the voltage LOW
digitalWrite(Relay2, HIGH);
digitalWrite(Relay3, HIGH); // turn the RELAY off by making the voltage LOW
digitalWrite(Relay4, HIGH);

 }
```

```
    DateTime now = rtc.now();

    Serial.print(now.year(), DEC);
    Serial.print('/');
    Serial.print(now.month(), DEC);
    Serial.print('/');
    Serial.print(now.day(), DEC);
    Serial.print(' ');
    Serial.print(now.hour(), DEC);
    Serial.print(':');
    Serial.print(now.minute(), DEC);
    Serial.print(':');
    Serial.print(now.second(), DEC);
    Serial.println();

     if(oleminute!=now.minute())//ตรวจสอบเวลาใหม่นั้นที่กทุก 10 s
    {

        myFile = SD.open("test.txt", FILE_WRITE);

// if the file opened okay, write to it:
     if (myFile)
       {
            myFile.print(now.year(), DEC);
   myFile.print('/');
   myFile.print(now.month(), DEC);
   myFile.print('/');
   myFile.print(now.day(), DEC);
   myFile.print(' ');
   myFile.print(now.hour(), DEC);
   myFile.print(':');
   myFile.print(now.minute(), DEC);
   myFile.print(':');
   myFile.print(now.second(), DEC);
   myFile.println();
```

```
      myFile.print('w');
      myFile.print(olecelsius);                    //temp water
       myFile.println(' ');
        //
         myFile.print('o');
        oleminute=now.minute();
        myFile.print(sensorstringcopy);            //OX
     myFile.println();
       //am
         myFile.print('r');
      myFile.print(olerh);                         //RH
       myFile.print('a');
      myFile.print(oletemp);                       //temp air

         // am
   myFile.close();
        Serial.println("done.");
      // delay(100);
      }else
       {
 // if the file didn't open, print an error:
       Serial.println("error opening test.txt");
       }                                            //send that string to the Atlas Scientific product


   oleminute=now.minute();
}
//LCD

 myGLCD.setFont(SmallFont);
     myGLCD.printNumI(now.hour(),3, 0);
   myGLCD.print(":",16,0);
    myGLCD.printNumI(now.minute(),21, 0);
      myGLCD.print(":",34,0);
     myGLCD.printNumI(now.second(),39,0);
    myGLCD.print( "PM", RIGHT, 0);
      myGLCD.printNumI(celsius, CENTER, 9);
      myGLCD.print("C.", RIGHT,9);
      myGLCD.printNumI( ((data[0] << 8) | data[1])/10.0 ,CENTER, 18);
      myGLCD.print( "%RH ", RIGHT, 18 );
      myGLCD.printNumI( ((data[2] << 8) | data[3])/10.0,CENTER, 27 );
      myGLCD.print( "C " , RIGHT, 27);
    myGLCD.update();
 delay(1000);

}
}
```

ตัวอย่างโปรแกรมย่อย

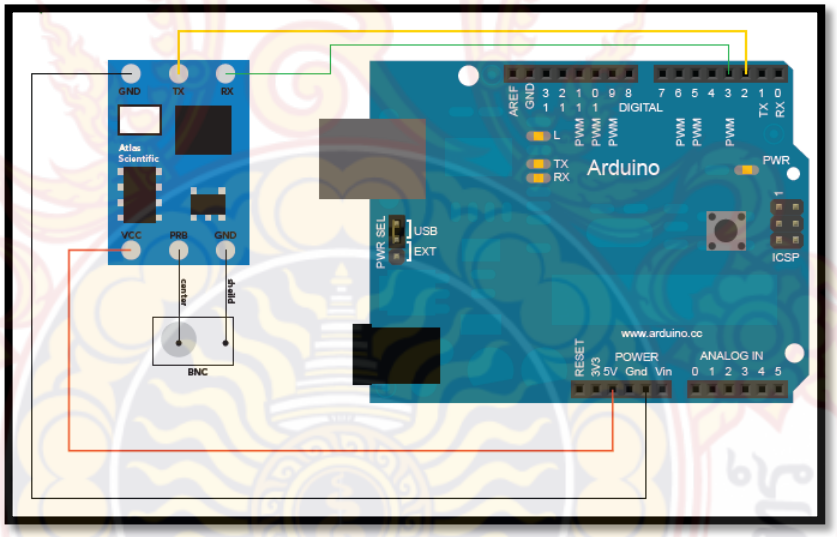


PH

```
#include <SoftwareSerial.h>
#define rx 2
#define tx 3
SoftwareSerial myserial(rx, tx);
char ph_data[20];
char computerdata[20];
byte received_from_computer=0;
byte received_from_sensor=0;
byte arduino_only=0;
byte startup=0;
float ph=0;
byte string_received=0;
void setup(){
```

```
Serial.begin(38400);
myserial.begin(38400);
}
void serialEvent(){
if(arduino_only!=1){
received_from_computer=Serial.readBytesUntil(13,computerdata,20);
computerdata[received_from_computer]=0;
myserial.print(computerdata);
myserial.print('\r');
}
}
void loop(){
if(myserial.available() > 0){
received_from_sensor=myserial.readBytesUntil(13,ph_data,20);
ph_data[received_from_sensor]=0;
string_received=1;
Serial.println(ph_data);
}
if(arduino_only==1){Arduino_Control();}
}
void Arduino_Control(){
if(startup==0){
myserial.print("e\r");
delay(50);
myserial.print("e\r");
delay(50);
startup=1;
}
```

```
delay(800);
myserial.print("R\r");
if(string_received==1){
ph=atof(ph_data);
if(ph>=7.5){Serial.println("high\r");}
if(ph<7.5){Serial.println("low\r");}
string_received=0;}
}
```

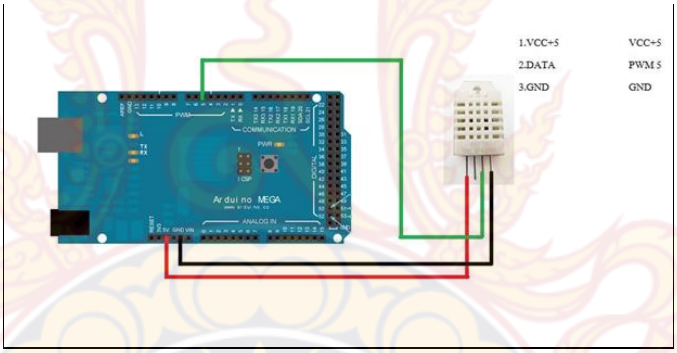


Oxygen

```
#include <SoftwareSerial.h>
#define rxpin 2
#define txpin 3
SoftwareSerial myserial(rxpin, txpin);
String inputstring = "";
```

```
String sensorstring = "";
boolean input_stringcomplete = false;
boolean sensor_stringcomplete = false;
void setup(){
 Serial.begin(38400);
 myserial.begin(38400);
 inputstring.reserve(5);
 sensorstring.reserve(30);
 }
void serialEvent() {
 char inchar = (char)Serial.read();
 inputstring += inchar;
 if(inchar == '\r') {input_stringcomplete = true;}
 }
void loop(){
 if (input_stringcomplete){
 myserial.print(inputstring);
 inputstring = "";
 input_stringcomplete = false;
 }
while (myserial.available()) {
 char inchar = (char)myserial.read();
 sensorstring += inchar;
 if (inchar == '\r') {sensor_stringcomplete = true;}
}
```

```
if (sensor_stringcomplete){
 Serial.print(sensorstring);
 sensorstring = "";
 sensor_stringcomplete = false;
 }
}
```



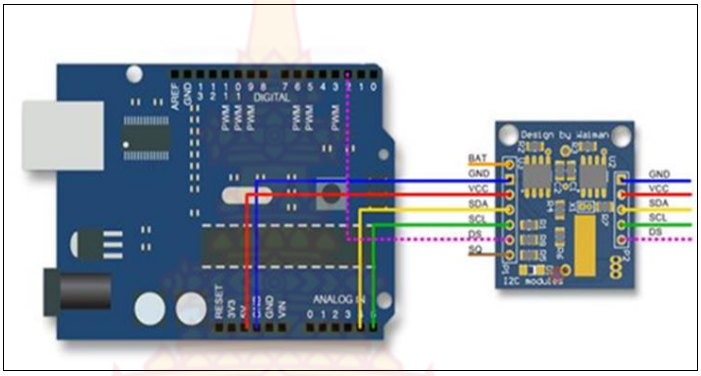
โค๊ด

เซนเซอร์วัดอุณหภูมิอากาศและความชื้น AM2302

```
/*
 AM2302 Temperature / Humidity Sensor (DHT22)
 Current Code
 Created by Derek Erb 30/01/2013
 Modified 30/01/2013
 Requirement : Arduino + AM2302 sensor connected to pin 2
 DHT Library from:
```

```
 https://github.com/adafruit/DHT-sensor-library
 v0.03 Delays and formatting
 v0.02 Delay 3 seconds between readings
*/
#include "DHT.h"          // DHT & AM2302 library
// Version number
const float fVerNum = 0.03;
// Data pin connected to AM2302
#define DHTPIN 2
#define DHTTYPE DHT22       // DHT 22  (AM2302)
DHT dht(DHTPIN, DHTTYPE);   // LED pins
//////////////////////////////////////////////////
//
// SETUP
//
void setup() {
  // Setup serial monitor
  Serial.begin(9600);
  // Wait 3 seconds
  delay(3000);
  Serial.println(F("\nAM2302 Sensor"));
  Serial.print(F("Version : "));
  Serial.println(fVerNum);
  Serial.println(F("Arduino - Derek Erb\n"));
  delay(5000);
```

```
  dht.begin();
}
void loop() {
  // Reading temperature or humidity takes about 250 milliseconds!
  // Sensor readings may also be up to 2 seconds 'old' (its a very slow sensor)
  float h = dht.readHumidity();
  float t = dht.readTemperature();
  // check if returns are valid, if they are NaN (not a number) then something went wrong!
  if (isnan(t) || isnan(h)) {
    Serial.println(F("Failed to read from DHT"));
  }
  else {
    Serial.print(F("Humidity: "));
    Serial.print(h);
    Serial.print(F(" %\t"));
    Serial.print(F("Temperature: "));
    Serial.print(t);
    Serial.println(F(" C"));
  }
  // Wait 3 seconds
  delay(3000);
}
```



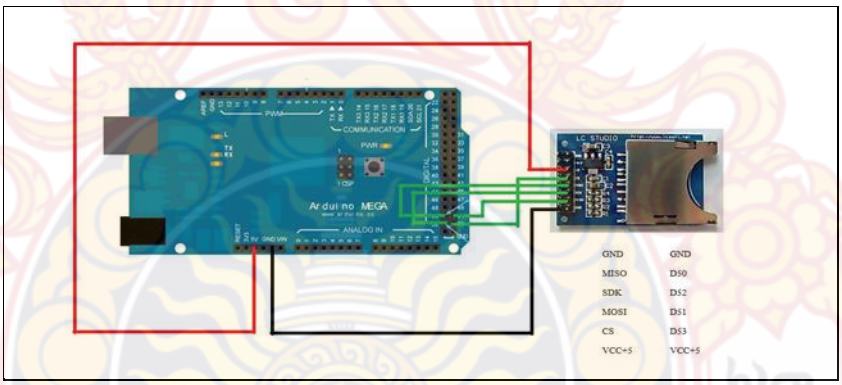
รูปที่ 2.17 การต่อใช้งาน**การใช้งาน RTC (Real Time Clock)** กับบอร์ด Arduino ที่มา http://www.arduino.in.th/

โค๊ด

Tiny RTC I2C

```
#include <Wire.h>
#include "RTClib.h"
RTC_DS1307 RTC;

void setup () {
  //Initialize the serial port, wire library and RTC module
    Serial.begin(9600);
    Wire.begin();
    RTC.begin();
    //If we remove the comment from the following line, we will set up the
module time and date with the computer one
    //RTC.adjust(DateTime(__DATE__, __TIME__));
}
void loop () {
    DateTime now = RTC.now();
    //We print the day
    Serial.print(now.day(), DEC);
    Serial.print('/');
    //We print the month
   Serial.print(now.month(), DEC);
   Serial.print('/');
   //We print the year
```

```
    Serial.print(now.year(), DEC);
    Serial.print(' ');
    //We print the hour
    Serial.print(now.hour(), DEC);
     Serial.print(':');
     //We print the minutes
     Serial.print(now.minute(), DEC);
     Serial.print(':');
     //We print the seconds
     Serial.print(now.second(), DEC);
     Serial.println();
     //We check the time and sent through the serial port every 3s
     delay(3000);
}
```



รูปที่ 2.19 การต่อใช้งาน SD Card Moduleกับบอร์ดArduin ที่มา http://www.arduino.in.th/

โค๊ด SD card

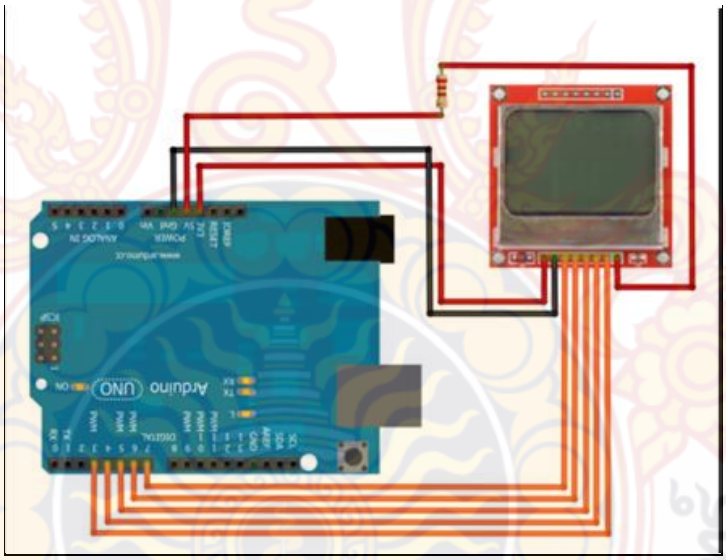
```
Sd2Card card;

SdVolume volume;

SdFile root;
```

```
const int chipSelect = 4;   // 4 for Arduino boards, might be different in other non-Arduino boards.


// we'll use the initialization code from the utility libraries
// since we're just testing if the card is working!
if (!card.init(SPI_HALF_SPEED, chipSelect)) {
        Serial.println("initialization failed. Things to check:");
        Serial.println("* is a card is inserted?");
        Serial.println("* Is your wiring correct?");
        Serial.println("* did you change the chipSelect pin to match your shield or module?");
        return;
}
else {
        Serial.println("Wiring is correct and a card is present.");
}


// print the type of card
```

```
Serial.print("\nCard type: ");
switch(card.type()) {
        case SD_CARD_TYPE_SD1:
                Serial.println("SD1");
                break;
        case SD_CARD_TYPE_SD2:
                Serial.println("SD2");
                break;
        case SD_CARD_TYPE_SDHC:
                Serial.println("SDHC");
                break;
        default:
                Serial.println("Unknown");
}
// Now we will try to open the 'volume'/'partition' - it should be FAT16 or FAT32
if (!volume.init(card)) {
        Serial.println("Could not find FAT16/FAT32 partition.\nMake sure you've formatted
the card");
```

```
        return;
}
// print the type and size of the first FAT-type volume
uint32_t volumesize;
Serial.print("\nVolume type is FAT");
Serial.println(volume.fatType(), DEC);
Serial.println();
volumesize = volume.blocksPerCluster();    // clusters are collections of blocks
volumesize *= volume.clusterCount();       // we'll have a lot of clusters
volumesize *= 512;                         // SD card blocks are always 512 bytes
Serial.print("Volume size (bytes): ");
Serial.println(volumesize);
Serial.print("Volume size (Kbytes): ");
volumesize /= 1024;
Serial.println(volumesize);
Serial.print("Volume size (Mbytes): ");
volumesize /= 1024;
Serial.println(volumesize);
```

```
Serial.println("\nFiles found on the card (name, date and size in bytes): ");

root.openRoot(volume);



// list all files in the card with date and size

root.ls(LS_R | LS_DATE | LS_SIZE);
```



รูปที่ 2.24 การเชื่อมต่อควบคุมจอแสดงผล LCD กับบอร์ด Arduino MEGA โครงสร้างภายในของ Controller PCD8544 ที่มา http://www.arduino.in.th/

โค๊ด

NOKIA 5110 LCD

```
#define PIN_SCE   7
```

```
#define PIN_RESET 6

#define PIN_DC    5

#define PIN_SDIN  4

#define PIN_SCLK  3



#define LCD_C     LOW

#define LCD_D     HIGH

#define LCD_X     84

#define LCD_Y     48

static const byte ASCII[][5] =

{

 {0x00, 0x00, 0x00, 0x00, 0x00} // 20

,{0x00, 0x00, 0x5f, 0x00, 0x00} // 21 !

,{0x00, 0x07, 0x00, 0x07, 0x00} // 22 "

,{0x14, 0x7f, 0x14, 0x7f, 0x14} // 23 #

,{0x24, 0x2a, 0x7f, 0x2a, 0x12} // 24 $

,{0x23, 0x13, 0x08, 0x64, 0x62} // 25 %

,{0x36, 0x49, 0x55, 0x22, 0x50} // 26 &
```

```
,{0x00, 0x05, 0x03, 0x00, 0x00} // 27 '
,{0x00, 0x1c, 0x22, 0x41, 0x00} // 28 (
,{0x00, 0x41, 0x22, 0x1c, 0x00} // 29 )
,{0x14, 0x08, 0x3e, 0x08, 0x14} // 2a *
,{0x08, 0x08, 0x3e, 0x08, 0x08} // 2b +
,{0x00, 0x50, 0x30, 0x00, 0x00} // 2c ,
,{0x08, 0x08, 0x08, 0x08, 0x08} // 2d -
,{0x00, 0x60, 0x60, 0x00, 0x00} // 2e .
,{0x20, 0x10, 0x08, 0x04, 0x02} // 2f /
,{0x3e, 0x51, 0x49, 0x45, 0x3e} // 30 0
,{0x00, 0x42, 0x7f, 0x40, 0x00} // 31 1
,{0x42, 0x61, 0x51, 0x49, 0x46} // 32 2
,{0x21, 0x41, 0x45, 0x4b, 0x31} // 33 3
,{0x18, 0x14, 0x12, 0x7f, 0x10} // 34 4
,{0x27, 0x45, 0x45, 0x45, 0x39} // 35 5
,{0x3c, 0x4a, 0x49, 0x49, 0x30} // 36 6
,{0x01, 0x71, 0x09, 0x05, 0x03} // 37 7
,{0x36, 0x49, 0x49, 0x49, 0x36} // 38 8
```

```
,{0x06, 0x49, 0x49, 0x29, 0x1e} // 39 9

,{0x00, 0x36, 0x36, 0x00, 0x00} // 3a :

,{0x00, 0x56, 0x36, 0x00, 0x00} // 3b ;

,{0x08, 0x14, 0x22, 0x41, 0x00} // 3c <

,{0x14, 0x14, 0x14, 0x14, 0x14} // 3d =

,{0x00, 0x41, 0x22, 0x14, 0x08} // 3e >

,{0x02, 0x01, 0x51, 0x09, 0x06} // 3f ?

,{0x32, 0x49, 0x79, 0x41, 0x3e} // 40 @

,{0x7e, 0x11, 0x11, 0x11, 0x7e} // 41 A

,{0x7f, 0x49, 0x49, 0x49, 0x36} // 42 B

,{0x3e, 0x41, 0x41, 0x41, 0x22} // 43 C

,{0x7f, 0x41, 0x41, 0x22, 0x1c} // 44 D

,{0x7f, 0x49, 0x49, 0x49, 0x41} // 45 E

,{0x7f, 0x09, 0x09, 0x09, 0x01} // 46 F

,{0x3e, 0x41, 0x49, 0x49, 0x7a} // 47 G

,{0x7f, 0x08, 0x08, 0x08, 0x7f} // 48 H

,{0x00, 0x41, 0x7f, 0x41, 0x00} // 49 I

,{0x20, 0x40, 0x41, 0x3f, 0x01} // 4a J
```

,{0x7f, 0x08, 0x14, 0x22, 0x41} // 4b K

,{0x7f, 0x40, 0x40, 0x40, 0x40} // 4c L

,{0x7f, 0x02, 0x0c, 0x02, 0x7f} // 4d M

,{0x7f, 0x04, 0x08, 0x10, 0x7f} // 4e N

,{0x3e, 0x41, 0x41, 0x41, 0x3e} // 4f O

,{0x7f, 0x09, 0x09, 0x09, 0x06} // 50 P

,{0x3e, 0x41, 0x51, 0x21, 0x5e} // 51 Q

,{0x7f, 0x09, 0x19, 0x29, 0x46} // 52 R

,{0x46, 0x49, 0x49, 0x49, 0x31} // 53 S

,{0x01, 0x01, 0x7f, 0x01, 0x01} // 54 T

,{0x3f, 0x40, 0x40, 0x40, 0x3f} // 55 U

,{0x1f, 0x20, 0x40, 0x20, 0x1f} // 56 V

,{0x3f, 0x40, 0x38, 0x40, 0x3f} // 57 W

,{0x63, 0x14, 0x08, 0x14, 0x63} // 58 X

,{0x07, 0x08, 0x70, 0x08, 0x07} // 59 Y

,{0x61, 0x51, 0x49, 0x45, 0x43} // 5a Z

,{0x00, 0x7f, 0x41, 0x41, 0x00} // 5b [

,{0x02, 0x04, 0x08, 0x10, 0x20} // 5c ¥

```
,{0x00, 0x41, 0x41, 0x7f, 0x00} // 5d ]

,{0x04, 0x02, 0x01, 0x02, 0x04} // 5e ^

,{0x40, 0x40, 0x40, 0x40, 0x40} // 5f _

,{0x00, 0x01, 0x02, 0x04, 0x00} // 60 `

,{0x20, 0x54, 0x54, 0x54, 0x78} // 61 a

,{0x7f, 0x48, 0x44, 0x44, 0x38} // 62 b

,{0x38, 0x44, 0x44, 0x44, 0x20} // 63 c

,{0x38, 0x44, 0x44, 0x48, 0x7f} // 64 d

,{0x38, 0x54, 0x54, 0x54, 0x18} // 65 e

,{0x08, 0x7e, 0x09, 0x01, 0x02} // 66 f

,{0x0c, 0x52, 0x52, 0x52, 0x3e} // 67 g

,{0x7f, 0x08, 0x04, 0x04, 0x78} // 68 h

,{0x00, 0x44, 0x7d, 0x40, 0x00} // 69 i

,{0x20, 0x40, 0x44, 0x3d, 0x00} // 6a j

,{0x7f, 0x10, 0x28, 0x44, 0x00} // 6b k

,{0x00, 0x41, 0x7f, 0x40, 0x00} // 6c l

,{0x7c, 0x04, 0x18, 0x04, 0x78} // 6d m

,{0x7c, 0x08, 0x04, 0x04, 0x78} // 6e n
```

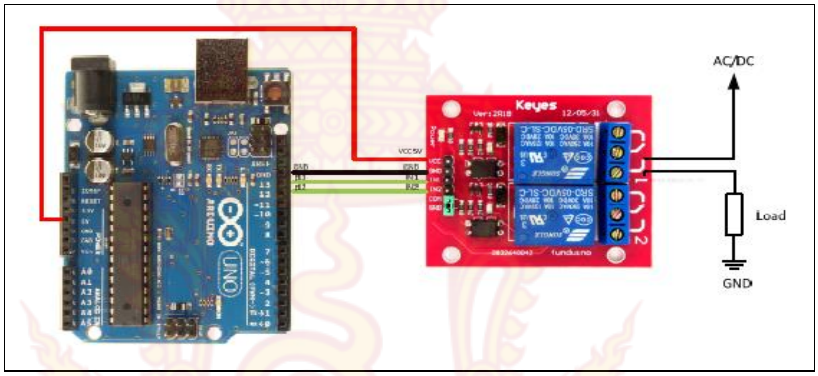
```
,{0x38, 0x44, 0x44, 0x44, 0x38} // 6f o
,{0x7c, 0x14, 0x14, 0x14, 0x08} // 70 p
,{0x08, 0x14, 0x14, 0x18, 0x7c} // 71 q
,{0x7c, 0x08, 0x04, 0x04, 0x08} // 72 r
,{0x48, 0x54, 0x54, 0x54, 0x20} // 73 s
,{0x04, 0x3f, 0x44, 0x40, 0x20} // 74 t
,{0x3c, 0x40, 0x40, 0x20, 0x7c} // 75 u
,{0x1c, 0x20, 0x40, 0x20, 0x1c} // 76 v
,{0x3c, 0x40, 0x30, 0x40, 0x3c} // 77 w
,{0x44, 0x28, 0x10, 0x28, 0x44} // 78 x
,{0x0c, 0x50, 0x50, 0x50, 0x3c} // 79 y
,{0x44, 0x64, 0x54, 0x4c, 0x44} // 7a z
,{0x00, 0x08, 0x36, 0x41, 0x00} // 7b {
,{0x00, 0x00, 0x7f, 0x00, 0x00} // 7c |
,{0x00, 0x41, 0x36, 0x08, 0x00} // 7d }
,{0x10, 0x08, 0x08, 0x10, 0x08} // 7e □
,{0x78, 0x46, 0x41, 0x46, 0x78} // 7f □
};
```

```
void LcdCharacter(char character)
{
  LcdWrite(LCD_D, 0x00);
  for (int index = 0; index < 5; index++)
  {
    LcdWrite(LCD_D, ASCII[character - 0x20][index]);
  }
  LcdWrite(LCD_D, 0x00);
}
void LcdClear(void)
{
  for (int index = 0; index < LCD_X * LCD_Y / 8; index++)
  {
    LcdWrite(LCD_D, 0x00);
  }
}

void LcdInitialise(void)
```

```
{
  pinMode(PIN_SCE, OUTPUT);
  pinMode(PIN_RESET, OUTPUT);
  pinMode(PIN_DC, OUTPUT);
  pinMode(PIN_SDIN, OUTPUT);
  pinMode(PIN_SCLK, OUTPUT);
  digitalWrite(PIN_RESET, LOW);
  digitalWrite(PIN_RESET, HIGH);
  LcdWrite(LCD_C, 0x21 );  // LCD Extended Commands.
  LcdWrite(LCD_C, 0xB1 );  // Set LCD Vop (Contrast).
  LcdWrite(LCD_C, 0x04 );  // Set Temp coefficent. //0x04
  LcdWrite(LCD_C, 0x14 );  // LCD bias mode 1:48. //0x13
  LcdWrite(LCD_C, 0x0C );  // LCD in normal mode.
  LcdWrite(LCD_C, 0x20 );
  LcdWrite(LCD_C, 0x0C );
}
void LcdString(char *characters)
{
```

```
  while (*characters)
  {
    LcdCharacter(*characters++);
  }
}
void LcdWrite(byte dc, byte data)
{
  digitalWrite(PIN_DC, dc);
  digitalWrite(PIN_SCE, LOW);
  shiftOut(PIN_SDIN, PIN_SCLK, MSBFIRST, data);
  digitalWrite(PIN_SCE, HIGH);
}
void setup(void)
{
  LcdInitialise();
  LcdClear();
  LcdString("Hello World!");
}
```

```
void loop(void)

{

}
```

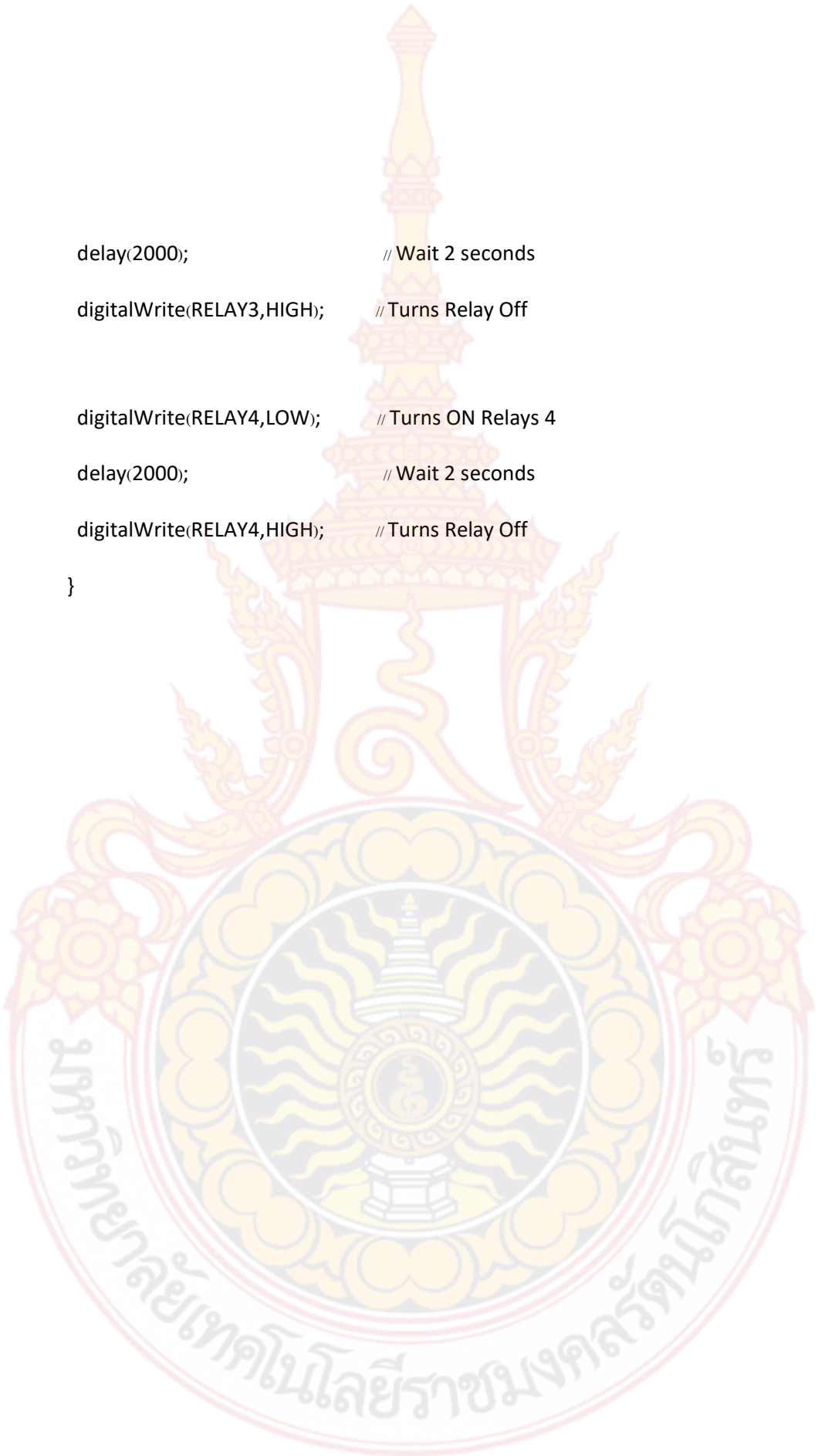


รูปที่ 2.28 การต่อใช้งานกับบอร์ด Arduino ที่มา http://www.micontechlab.com/

```
// Basic 4 Realy board connection
// Each relay is turned on for 2 seconds and then off.
// You can here them click as there state changes from off to on and on to
// off.
// You will also see the corresponding Red LED on the 4 Relay board
// light up when the relay is on.


// define names for the 4 Digital pins On the Arduino 7,8,9,10
// These data pins link to 4 Relay board pins IN1, IN2, IN3, IN4


#define RELAY1  6
```

```
#define RELAY2  7
#define RELAY3  8
#define RELAY4  9

void setup()
{
// Initialise the Arduino data pins for OUTPUT
  pinMode(RELAY1, OUTPUT);
  pinMode(RELAY2, OUTPUT);
  pinMode(RELAY3, OUTPUT);
  pinMode(RELAY4, OUTPUT);
}

 void loop()
{
  digitalWrite(RELAY1,LOW);       // Turns ON Relays 1
  delay(2000);                    // Wait 2 seconds
  digitalWrite(RELAY1,HIGH);      // Turns Relay Off

  digitalWrite(RELAY2,LOW);       // Turns ON Relays 2
  delay(2000);                    // Wait 2 seconds
  digitalWrite(RELAY2,HIGH);      // Turns Relay Off

  digitalWrite(RELAY3,LOW);       // Turns ON Relays 3
```

```
  delay(2000);                    // Wait 2 seconds
  digitalWrite(RELAY3,HIGH);      // Turns Relay Off

  digitalWrite(RELAY4,LOW);       // Turns ON Relays 4
  delay(2000);                    // Wait 2 seconds
  digitalWrite(RELAY4,HIGH);      // Turns Relay Off
}
```



ประวัติผู้วิจัย

1. ชื่อ - นามสกุล (ภาษาไทย) นายวรุตม์ บุญเลี่ยม
ชื่อ - นามสกุล (ภาษาอังกฤษ) Mr. Waroot Boonliam
2. เลขหมายบัตรประจำตัวประชาชน 3 779800011 03 2
3. ตำแหน่งปัจจุบัน พนักงานมหาวิทยาลัย (อาจารย์)
4. หน่วยงานและสถานที่อยู่ที่ติดต่อได้สะดวก พร้อมหมายเลขโทรศัพท์ที่ทำงาน โทรศัพท์มือถือ โทรสาร และไปรษณีย์อิเล็กทรอนิกส์ (e-mail)

คณะอุตสาหกรรมและเทคโนโลยี มหาวิทยาลัยเทคโนโลยีราชมงคลรัตนโกสินทร์ ถ. เพชรเกษม ต.หนองแก อ.หัวหิน จ.ประจวบคีรีขันธ์ 77110 โทรศัพท์ 0-3261-8500 ต่อ 4033 โทรสาร 0-3261-8570 E-nail k_waroot@yahoo.com ,waroot.boo@rmutr.ac.th

5. ประวัติการศึกษา

ปีที่จบ	ระดับปริญญา	ชื่อย่อปริญญา	สาขาวิชา	สถาบัน	ประเทศ
2552	ปริญญาโท	วท.ม.	เทคโนโลยีคอมพิวเตอร์เพื่อการศึกษา	มหาวิทยาลัยราชภัฏเพชรบุรี	ไทย
2543	ปริญญาตรี	วศ.บ.	วิศวกรรมคอมพิวเตอร์	มหาวิทยาลัยเทคโนโลยีมหานคร	ไทย

6. สาขาวิชาการที่มีความชำนาญพิเศษ (แตกต่างจากวุฒิการศึกษา) ระบุสาขาวิชาการ
- ไมโครคอนโทรลเลอร์

7. ประสบการณ์ที่เกี่ยวข้องกับการบริหารงานวิจัยทั้งภายในและภายนอกประเทศ โดยระบุสถานภาพในการทำการวิจัยว่าเป็นผู้อำนวยการแผนงานวิจัย หัวหน้าโครงการวิจัย หรือผู้ร่วมวิจัย ในแต่ละข้อเสนอการวิจัย

1. ชื่อ - นามสกุล (ภาษาไทย) นายพรประสิทธิ์ บุญทอง
   ชื่อ - นามสกุล (ภาษาอังกฤษ) Mr. Pornprasit Boontong
2. เลขหมายบัตรประจำตัวประชาชน 3 8102 00065 561
3. ตำแหน่งปัจจุบัน อาจารย์ 7
4. หน่วยงานและสถานที่อยู่ที่ติดต่อได้สะดวก พร้อมหมายเลขโทรศัพท์ที่ทำงาน โทรศัพท์มือถือ โทรสาร และไปรษณีย์อิเล็กทรอนิกส์ (e-mail)

   คณะอุตสาหกรรมและเทคโนโลยี มหาวิทยาลัยเทคโนโลยีราชมงคลรัตนโกสินทร์ ถ. เพชรเกษม ต.หนองแก อ.หัวหิน จ.ประจวบคีรีขันธ์ 77110 โทรศัพท์ 0-3261-8500 ต่อ 4033 โทรสาร 0-3261-8570 E-nail pornprasit@rmutr.ac.th

5. ประวัติการศึกษา

ปีที่จบ	ระดับปริญญา	ชื่อย่อปริญญา	สาขาวิชา	สถาบัน	ประเทศ
2550	ปริญญาโท	วศ.ม.	วิศวกรรมคอมพิวเตอร์	สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง	ไทย
2539	ปริญญาตรี	วศ.บ.	วิศวกรรมคอมพิวเตอร์	สถาบันเทคโนโลยีราชมงคล	ไทย

6. สาขาวิชาการที่มีความชำนาญพิเศษ (แตกต่างจากวุฒิการศึกษา) ระบุสาขาวิชาการ
   - ระบบเครือข่าย
   - ระบบฐานข้อมูล
7. ประสบการณ์ที่เกี่ยวข้องกับการบริหารงานวิจัยทั้งภายในและภายนอกประเทศ โดยระบุสถานภาพในการทำการวิจัยว่าเป็นผู้อำนวยการแผนงานวิจัย หัวหน้าโครงการวิจัย หรือผู้ร่วมวิจัยในแต่ละข้อเสนอการวิจัย

1. ชื่อ - นามสกุล (ภาษาไทย) นายอาทิตย์ อยู่เย็น
ชื่อ - นามสกุล (ภาษาอังกฤษ) Mr. Arthit Yooyen

2. เลขหมายบัตรประจำตัวประชาชน 3 770100018 38 4

3. ตำแหน่งปัจจุบัน พนักงานมหาวิทยาลัย (อาจารย์)

4. หน่วยงานและสถานที่อยู่ที่ติดต่อได้สะดวก พร้อมหมายเลขโทรศัพท์ที่ทำงาน โทรศัพท์มือถือ โทรสาร และไปรษณีย์อิเล็กทรอนิกส์ (e-mail)

คณะอุตสาหกรรมและเทคโนโลยี มหาวิทยาลัยเทคโนโลยีราชมงคลรัตนโกสินทร์ ถ. เพชรเกษม ต.หนองแก อ.หัวหิน จ.ประจวบคีรีขันธ์ 77110 โทรศัพท์ 0-3261-8500 ต่อ 4033 โทรสาร 0-3261-8570 E-nail thit19@hotmail.com, thit19@gmail.com

5. ประวัติการศึกษา

ปีที่จบ	ระดับปริญญา	ชื่อย่อปริญญา	สาขาวิชา	สถาบัน	ประเทศ
2554	ปริญญาโท	วท.ม.	เทคโนโลยีสารสนเทศ	มหาวิทยาลัยเทคโนโลยีมหานคร	ไทย
2547	ปริญญาตรี	อส.บ.	เทคโนโลยีคอมพิวเตอร์	สถาบันเทคโนโลยีราชมงคล วข.วังไกลกังวล	ไทย

6. สาขาวิชาการที่มีความชำนาญพิเศษ (แตกต่างจากวุฒิการศึกษา) ระบุสาขาวิชาการ
- ไมโครคอนโทรลเลอร์

7. ประสบการณ์ที่เกี่ยวข้องกับการบริหารงานวิจัยทั้งภายในและภายนอกประเทศ โดยระบุสถานภาพในการทำการวิจัยว่าเป็นผู้อำนวยการแผนงานวิจัย หัวหน้าโครงการวิจัย หรือผู้ร่วมวิจัย ในแต่ละข้อเสนอการวิจัย