การจำลองและออกแบบระบบหล่อเย็นภายในโรงเรือนเลี้ยงไก่แบบปิด

SIMULATION AND DESIGN OF EVAPORATIVE COOLING SYSTEM FOR POULTRY CLOSED HOUSE

โดย

นายพงศธร รุ่งเรือง

นายพิพัฒน์ เจริญพจน์

นายศุภวิชญ์ จิตรอักษร

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต

ภาควิชาวิศวกรรมโทรคมนาคม

คณะวิศวกรรมศาสตร์

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ปีการศึกษา 2561

การจำลองและออกแบบระบบหล่อเย็นภายในโรงเรือนเลี้ยงไก่แบบปิด

SIMULATION AND DESIGN OF EVAPORATIVE COOLING SYSTEM FOR POULTRY CLOSED HOUSE

โดย

นายพงศธร	รุ่งเรือง	58010811
นายพิพัฒน์	เจริญพจน์	58010888
นายศุภวิชญ์	จิตรอักษร	58011251

อาจารย์ที่ปรึกษา

รศ.ดร.มนตรี คำเงิน

ผศ.ดร.สมเกียรติ ฤกษ์วรัญญู

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต

ภาควิชาวิศวกรรมโทรคมนาคม

คณะวิศวกรรมศาสตร์

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ปีการศึกษา 2561

ผ่านการตรวจรูปเล่มแล้ว

(.....)

อาจารย์ที่ปรึกษา

23/05/62

ผ่านการตรวจชิ้นงานแล้ว

(.....)

กรรมการผู้ตรวจชิ้นงาน

23/5/62

ปริญญานิพนธ์ปีการศึกษา 2561

ภาควิชาวิศวกรรมโทรคมนาคม

คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

เรื่อง การจำลองและออกแบบระบบหล่อเย็นภายในโรงเรือนเลี้ยงไก่แบบปิด
SIMULATION AND DESIGN OF EVAPORATIVE COOLING SYSTEM
FOR POULTRY CLOSED HOUSE

ผู้จัดทำ

1. นายพงศธร รุ่งเรือง 58010811
2. นายพิพัฒน์ เจริญพจน์ 58010888
3. นายศุภวิชญ์ จิตรอักษร 58011251

..... อาจารย์ที่ปรึกษา
(รศ.ดร.มนตรี คำเงิน)

..... อาจารย์ที่ปรึกษาร่วม
(ผศ.ดร.สมเกียรติ ฤกษ์วรัญญู)

กิตติกรรมประกาศ

ปริญญานิพนธ์นี้สำเร็จไปได้ด้วยดี ด้วยความกรุณาของ รศ.ดร.มนตรี คำเงิน ผศ.ดร.สมเกียรติ ฤกษ์วรัญญู เป็นอาจารย์ที่ปรึกษาปริญญานิพนธ์ ซึ่งท่านได้ให้คำแนะนำ ช่วยเหลือและสนับสนุนสิ่งต่างๆ ในระหว่างการดำเนินงาน ทางคณะผู้จัดทำขอขอบพระคุณเป็นอย่างสูง

ขอขอบพระคุณ บิดา มารดา เพื่อน ตลอดจนผู้ที่เกี่ยวข้องทุกท่าน ที่ได้ให้กำลังใจและคอยให้การสนับสนุนช่วยเหลือในการทำปริญญานิพนธ์ให้สำเร็จลุล่วง

ท้ายที่สุดนี้ ทางคณะผู้จัดทำหวังเป็นอย่างยิ่งว่าปริญญานิพนธ์นี้จะเป็นประโยชน์กับผู้ที่สนใจไม่มากก็น้อย

นายพงศธร รุ่งเรือง
นายพิพัฒน์ เจริญพจน์
นายศุภวิชญ์ จิตรอักษร
ผู้จัดทำ

การจำลองและออกแบบระบบหล่อเย็นภายในโรงเรือนเลี้ยงไก่แบบปิด
SIMULATION AND DESIGN OF EVAPORATIVE COOLING SYSTEM FOR POULTRY CLOSED HOUSE

โดย นายพงศธร รุ่งเรือง 58010811
นายพิพัฒน์ เจริญพจน์ 58010888
นายศุภวิชญ์ จิตรอักษร 58011251

อาจารย์ที่ปรึกษา รศ.ดร.มนตรี คำเงิน
อาจารย์ที่ปรึกษาร่วม ผศ.ดร.สมเกียรติ ฤกษ์วรัญญู

บทคัดย่อ

ปริญญานิพนธ์นี้จะทำการออกแบบระบบควบคุมอุณหภูมิและความชื้นสัมพัทธ์ที่อยู่ในโรงเรือนเลี้ยงไก่แบบปิดผ่านหน้าจอมอนิเตอร์ โดยโรงเรือนนั้นจะทำการส่งข้อมูลจากเซ็นเซอร์เข้าไปยังหน้าจอแสดงผลเพื่อแสดงค่าอุณหภูมิและความชื้นสัมพัทธ์ในโรงเรือนและข้อมูลนั้นจะถูกส่งไปเก็บไว้ในฐานข้อมูล ซึ่งระบบนี้ใช้โมดูล XBee ในการรับส่งข้อมูล นอกจากนี้หน้าจอมอนิเตอร์ยังสามารถควบคุมพัดลมดูดอากาศ แผงความเย็น และไฟส่องสว่างภายในโรงเรือนเพื่อที่จะควบคุมอุณหภูมิและความชื้นสัมพัทธ์ในโรงเรือนนั้นไม่ให้เกินค่าที่กำหนด เพื่อให้สามารถรักษาสภาพอากาศภายในโรงเรือนให้คงที่ให้เหมาะสมสำหรับการเลี้ยงไก่

ABSTRACT

This project designs the system to control temperature and relative humidity in a poultry closed house through the monitor screen. The poultry closed house sends information from the sensor to the monitor screen for show the temperature and relative humidity in the house and that information was send to the database. This system uses the XBee module to transmit data. In addition, the monitor screen can control fan, cooling pad and light in the poultry closed house for requirement of the temperature and relative humidity. The specified values in order to maintain the weather that suitable for raising chickens can be obtained.

สารบัญ

		หน้า
กิตติกรรมประกาศ		I
บทคัดย่อ		II
สารบัญ		IV
สารบัญรูป		V
สารบัญตาราง		VI
บทที่ 1	**บทนำ**	**1**
	1.1 ความเป็นมาและความสำคัญของปัญหา	1
	1.2 วัตถุประสงค์	2
	1.3 ขอบเขตของปริญญานิพนธ์	2
บทที่ 2	**ทฤษฎีและหลักการที่เกี่ยวข้อง**	**3**
	2.1 Microsoft Visual Studio	3
	2.2 Microsoft Access	4
	2.3 ภาษา SQL	5
	2.4 เทคโนโลยี Zigbee	6
	2.5 Arduino Uno R3	17
	2.6 เซ็นเซอร์วัดอุณหภูมิและความชื้นสัมพัทธ์ DHT22 (AM2302)	19
	2.7 อุณหภูมิและความชื้นสัมพัทธ์	20
	2.8 Relay	21
	2.9 Switching Power Supply	23
	2.10 โรงเรือนเลี้ยงไก่แบบปิด	25
บทที่ 3	**การออกแบบและการจัดทำปริญญานิพนธ์**	**32**
	3.1 การออกแบบ	32

สารบัญ (ต่อ)

		หน้า
	3.2 เครื่องมือที่ใช้ในการทดลอง	54
	3.3 การจัดเก็บผลการทดลอง	55
บทที่ 4	**ผลการทดลอง**	**56**
	4.1 ผลการทดลองการวัดค่าอุณหภูมิ	56
	4.2 ผลการทดลองการวัดระยะทางในการรับส่งข้อมูลของเครือข่าย XBee	58
	4.3 ผลการทดลองระบบควบคุมอุณหภูมิและความชื้นสัมพัทธ์ในแบบจำลองโรงเรือนเลี้ยงไก่	61
	4.4 ผลการทดลองในการเก็บค่าในฐานข้อมูล	71
	4.5 การทำงานของหน้าจอแสดงผล	72
บทที่ 5	**สรุปผลและข้อเสนอแนะ**	**76**
	5.1 สรุปผล	76
	5.2 ข้อเสนอแนะ	76
บรรณานุกรม		**78**
ภาคผนวก ก	โค้ด XBee ฝั่ง Router	79
ภาคผนวก ข	โค้ด XBee ฝั่ง Coordinator	88

สารบัญรูป

รูปที่		หน้า
2.1	หน้าจอโปรแกรม Visual Basic	4
2.2	โมดูล Zigbee PRO S2	6
2.3	ย่านความถี่ใช้งานตามมาตรฐาน Zigbee	7
2.4	Zigbee Frame Work	8
2.5	Topology แบบ Star	9
2.6	Topology แบบ Tree	10
2.7	Topology แบบ Cluster Tree	11
2.8	Topology แบบ Mesh	12
2.9	แสดงการส่งข้อมูลของ XBee โหมด AT	13
2.10	การตั้งค่าแอดเดรสปลายทางให้ XBee สามารถสื่อสารกันได้ในโหมด AT	13
2.11	แสดงการส่งข้อมูลของ XBee โหมด API	14
2.12	โครงสร้างของเฟรมที่ใช้ภายใต้ XBee โหมด API	14
2.13	API Frame สำหรับส่งข้อมูลด้วยแอดเดรส 64-bit	15
2.14	API Frame ที่จะได้รับ เมื่อต้นทางส่งข้อมูลมาให้ด้วยแอดเดรส 64-bit	16
2.15	แต่ละกลุ่มของชุดข้อมูล API mode	16
2.16	Layout & Pinout Arduino Uno R3	17
2.17	เซ็นเซอร์ DHT22	19
2.18	สัญลักษณ์ในวงจรไฟฟ้าของ Relay	22
2.19	Relay 8 Channel	23
2.20	แผนผัง Switching Power Supply	24
2.21	Switching Power Supply	25
2.22	ลักษณะของอากาศที่เข้าไปในโรงเรือนโดยผ่านแผ่นรังผึ้ง	28
2.23	การหมุนเวียนของอากาศในโรงเรือน	29
3.1	บล็อกไดอะแกรมของระบบ	32
3.2	ส่วนด้านหน้าของแบบจำลอง	33

สารบัญรูป (ต่อ)

หน้า

รูปที่

3.3	ส่วนด้านหลังของแบบจำลอง	34
3.4	ส่วนด้านข้างและด้านนอกของแบบจำลอง	34
3.5	ส่วนด้านในของแบบจำลอง	35
3.6	แบบจำลองโรงเรือนเลี้ยงไก่แบบปิด	35
3.7	แผนผังการการควบคุมเปิดปิดพัดลมและหลอดไฟด้วยตัวเอง	36
3.8	แผนผังการควบคุมเปิดปิดพัดลมและปั้มน้ำแบบอัตโนมัติ	38
3.9	รายละเอียดการเชื่อมต่อระหว่าง Arduino Uno R3 กับ DHT22	39
3.10	หน้าแรกของหน้าจอแสดงผล	40
3.11	หน้าจอส่วนของการเลือกพอร์ตเชื่อมต่อกับ Arduino	40
3.12	หน้าจอแสดงผลเมื่อทำการใส่ค่าอุณหภูมิที่ต้องการให้เกิดขึ้นในแบบจำลอง	41
3.13	หน้าจอแสดงผลในโหมดเลือกระยะเวลาการเลี้ยงไก่	41
3.14	หน้าจอแสดงผลในโหมดตั้งค่าอุณหภูมิ	42
3.15	หน้าต่างให้ผู้ใช้งานตั้งค่าอุณหภูมิในแต่ละวัน	42
3.16	ปุ่ม Light แสดงสถานะหลอดไฟเปิด	43
3.17	ปุ่ม Light แสดงสถานะหลอดไฟปิด	43
3.18	ปุ่ม Fan แสดงสถานะพัดลมเปิดและแสดงสถานะพัดลมกับปั้มน้ำในระบบหล่อเย็น	44
3.19	หน้าต่างของปุ่ม Temperature	44
3.20	หน้าต่างของปุ่ม Humidity	45
3.21	หน้าต่างของปุ่ม Database	45
3.22	แผนผังการทำงานของหน้าจอแสดงผล	46
3.23	ตารางฐานข้อมูล Microsoft Access ที่ใช้ในการเก็บค่าอุณหภูมิและความชื้นสัมพัทธ์ ณ วันที่และเวลาต่างๆ	47
3.24	การ Set Parameter ให้ติดต่อกันแบบ Point to Point	48
3.25	การ Set Parameter PAN ID	49

สารบัญรูป (ต่อ)

หน้า

รูปที่

3.26	การ Set Parameter SH SL DH DL ของฝั่ง Router	49
3.27	การ Set Parameter SH SL DH DL ของฝั่ง Coordinator	49
3.28	การ Set Parameter API Enable ของฝั่ง Router	50
3.29	การ Set Parameter API Enable ของฝั่ง Coordinator	50
3.30	แผนผังการทำงานของ XBee ฝั่งที่เชื่อมต่อกับหน้าจอแสดงผล	51
3.31	แผนผังการทำงานของ XBee ฝั่ง Router	53
4.1	กราฟแสดงการเปรียบเทียบค่าอุณหภูมิที่อ่านจากเทอร์โมมิเตอร์กับค่าอุณหภูมิที่อ่านจากเซ็นเซอร์ DHT22	57
4.2	ระยะทางจากถนนหน้าตึกวิชาภาคโทรคมนาคมถึงสุดถนนอาคารเรียนรวม 12 ชั้น	58
4.3	แบบจำลองโรงเรือนเลี้ยงไก่ที่ตั้งอยู่ริมถนนหน้าตึกภาควิชาโทรคมนาคม	59
4.4	ทดลองส่งข้อมูลที่สุดถนนหน้าอาคารเรียนรวม 12 ชั้น	59
4.5	กราฟแสดงความสัมพันธ์ระหว่างจำนวนครั้งที่ส่งและรับกับระยะห่าง	61
4.6	แบบจำลองโรงเรือนเลี้ยงไก่ที่ตั้งไว้ในช่วงเวลากลางวัน	62
4.7	หน้าจอแสดงผลที่แสดงค่าอุณหภูมิภายในแบบจำลองโรงเรือนเลี้ยงไก่และค่าเกณฑ์อุณหภูมิที่เรากำหนดเองในเวลากลางวัน	63
4.8	ค่าอุณหภูมิและความชื้นสัมพัทธ์ภายในโรงเรือนเลี้ยงไก่ ณ เวลา 16.50 น.	64
4.9	ค่าอุณหภูมิและความชื้นสัมพัทธ์ภายในโรงเรือนเลี้ยงไก่ ณ เวลา 16.51 น.	64
4.10	ค่าอุณหภูมิและความชื้นสัมพัทธ์ภายในโรงเรือนเลี้ยงไก่ ณ เวลา 16.53 น.	64
4.11	ค่าอุณหภูมิและความชื้นสัมพัทธ์ภายในโรงเรือนเลี้ยงไก่ ณ เวลา 16.54 น.	65
4.12	ค่าอุณหภูมิและความชื้นสัมพัทธ์ภายในโรงเรือนเลี้ยงไก่ ณ เวลา 16.56 น.	65
4.13	ค่าอุณหภูมิและความชื้นสัมพัทธ์ภายในโรงเรือนเลี้ยงไก่ ณ เวลา 17.03 น.	65
4.14	การปรับค่าเกณฑ์อุณหภูมิใหม่	66
4.15	ค่าอุณหภูมิและความชื้นสัมพัทธ์ภายในแบบจำลองโรงเรือนเลี้ยงไก่ ณ เวลา 17.34 น.	66

สารบัญรูป (ต่อ)

รูปที่		หน้า
4.16	กราฟความสัมพันธ์ระหว่างอุณหภูมิกับเวลาที่แสดงบนหน้าจอแสดงผล ช่วงเวลากลางวัน	67
4.17	กราฟความสัมพันธ์ระหว่างความชื้นสัมพัทธ์กับเวลาที่แสดงบนหน้าจอแสดงผล ช่วงเวลากลางวัน	67
4.18	แบบจำลองโรงเรือนเลี้ยงไก่ที่ตั้งไว้ในช่วงเวลากลางคืน	68
4.19	หน้าจอแสดงผลที่แสดงค่าอุณหภูมิภายในแบบจำลองโรงเรือนเลี้ยงไก่และค่าเกณฑ์อุณหภูมิที่เรากำหนดเองในเวลากลางคืน	68
4.20	ค่าอุณหภูมิและความชื้นสัมพัทธ์ภายในโรงเรือนเลี้ยงไก่ ณ เวลา 19.46 น.	69
4.21	ค่าอุณหภูมิและความชื้นสัมพัทธ์ภายในโรงเรือนเลี้ยงไก่ ณ เวลา 20.00 น.	69
4.22	ค่าอุณหภูมิและความชื้นสัมพัทธ์ภายในโรงเรือนเลี้ยงไก่ ณ เวลา 20.44 น.	69
4.23	กราฟความสัมพันธ์ระหว่างอุณหภูมิกับเวลาที่แสดงบนหน้าจอแสดงผล ช่วงเวลากลางวัน	70
4.24	กราฟความสัมพันธ์ระหว่างความชื้นสัมพัทธ์กับเวลาที่แสดงบนหน้าจอแสดงผล ช่วงเวลากลางคืน	71
4.25	ค่าอุณหภูมิและความชื้นสัมพัทธ์ ณ วันที่และเวลาต่างๆ	72
4.26	หน้าจอที่ใช้กดสั่งให้เปิดปิดไฟ	73
4.27	(ก) เมื่อสั่งเปิดไฟแล้วไฟติด (ข) เมื่อสั่งปิดไฟแล้วไฟดับ	73
4.28	หน้าจอที่ใช้กดสั่งให้เปิดปิดพัดลมและแสดงสถานะของระบบหล่อเย็น	74
4.29	(ก) เมื่อสั่งปิดพัดลมดับ (ข) เมื่อสั่งเปิดพัดลมติด	74
4.30	ค่าอุณหภูมิภายในแบบจำลองโรงเรือนเลี้ยงไก่แบบปิด	75
4.31	ค่าความชื้นสัมพัทธ์ภายในแบบจำลองโรงเรือนเลี้ยงไก่แบบปิด	75

สารบัญตาราง

ตารางที่		หน้า
2.1	คุณสมบัติต่าง ๆ ของบอร์ด Arduino Uno R3	18
2.2	หน้าที่ของแต่ละขาของ DHT22	19
4.1	การเปรียบเทียบอุณหภูมิระหว่างเซ็นเซอร์ DHT22 กับเทอร์โมมิเตอร์ ในตู้เย็น	56
4.2	การทดลองในกรณีที่เป็นแบบ Line-Of-Sight	60

บทที่ 1

บทนำ

1.1 ความเป็นมาและความสำคัญของปัญหา

เนื่องจากประเทศไทยเป็นประเทศที่อยู่ในเขตร้อนมีอากาศและอุณหภูมิสูง การเลี้ยงไก่ส่วนมากจะเป็นการเลี้ยงไก่ในแบบโรงเรือนระบบเปิด ซึ่งจะทำให้ภายในโรงเรือนไม่สามารถควบคุมอุณหภูมิได้ สภาพอากาศภายในโรงเรือนจะขึ้นอยู่กับอากาศภายนอก ถ้าหากอากาศเปลี่ยนแปลงบ่อย ๆ หรืออากาศร้อนจนเกินไปก็จะทำให้ไก่ตายได้ง่าย และถ้าหากไก่ติดเชื้อไข้หวัดนกก็ไม่สามารถควบคุมการแพร่ระบาดของโรคได้ จึงมีการใช้โรงเรือนระบบปิดในการเลี้ยงไก่ ถือว่าเป็นสิ่งที่จำเป็นและมีความสำคัญมาก โดยจะมีการใช้ระบบเข้ามาช่วยในการดูแลเลี้ยงไก่

ในปัจจุบันได้มีการนำไมโครคอนโทรลเลอร์ Arduino ซึ่งเป็นบอร์ดที่ง่ายต่อการพัฒนา มีรูปแบบคำสั่งที่ไม่ซับซ้อน และมีราคาไม่แพง มาประยุกต์ใช้งานมากมาย เช่น การเขียนโปรแกรมใช้งานฟังก์ชันอินพุตเอาต์พุตดิจิตอล การเขียนโปรแกรมควบคุมอุปกรณ์ต่างๆ และยังสามารถนำมาเชื่อมต่อกับตัวเซ็นเซอร์ได้หลากหลายชนิดเพื่อตรวจจับหรือวัดค่าต่างๆที่เราต้องการ มีการใช้โปรแกรม Visual Studio ซึ่งเป็นโปรแกรมที่ทำงานในระบบปฏิบัติการวินโดวส์ รับรองภาษาในการออกแบบได้หลายภาษา โปรแกรมนี้สามารถนำมาสร้างแอพพลิเคชั่นได้หลายแบบ เช่น โปรแกรมด้านธุรกิจ โปรแกรมควบคุมการทำงานของระบบต่างๆ และยังสามารถเชื่อมต่อกับกับฐานข้อมูลได้อีกด้วย มีการใช้โปรแกรมฐานข้อมูล ซึ่งใช้ในการเก็บข้อมูลต่างๆ เช่น ค่าอุณหภูมิ ความชื้น สถานะต่างๆของระบบ เพื่อนำไปใช้ประมวลต่อไป นอกจากนี้ยังมีการใช้โมดูลสื่อสารไร้สายต่างๆ เช่น XBee RoLa ESP8266 ในการรับส่งข้อมูลจากต้นทางไปยังปลายทาง ซึ่งโมดูลเหล่านี้นั้นจะมีขนาดเล็ก สามารถรับส่งข้อมูลในระยะไกลได้อย่างรวดเร็วและแม่นยำ

จากปัญหาและข้อดีของเทคโนโลยีสมัยใหม่ที่ได้กล่าวมาข้างต้น จึงทำให้ผู้จัดทำได้จำลองและออกแบบระบบหล่อเย็นในโรงเรือนเลี้ยงไก่แบบปิด โดยได้นำ Arduino มาเชื่อมต่อกับเซ็นเซอร์วัดอุณหภูมิและความชื้นสัมพัทธ์ ซึ่งตัวเซ็นเซอร์จะวัดค่าอุณหภูมิในโรงเรือนว่าเหมาะสมหรือไม่ ถ้าไม่เหมาะสม Arduino ก็จะสั่งการให้ระบบหล่อเย็นในโรงเรือนทำงาน เพื่อเป็นการควบคุมอุณหภูมิและความชื้นสัมพัทธ์ให้มีความเหมาะสมต่อการเลี้ยงไก่ตลอดเวลาและลดปัญหาการเจ็บป่วยหรือติดเชื้อของไก่ที่เกิดจากสภาพอากาศที่มีความแปรปรวน นอกจากนี้ได้ใช้โปรแกรม Visual Studio ในการสร้างหน้าจอแสดงผล เพื่อให้ผู้ใช้งานได้รับรู้ข้อมูลต่างที่อยู่ในโรงเรือนได้แก่ อุณหภูมิ ความชื้น

สัมพัทธ์ สถานะการทำงานของอุปกรณ์ต่างๆ และใช้ในการสั่งเปิดปิด ไฟ พัดลม ในโรงเรือนได้อีกด้วย ซึ่งข้อมูลที่ถูกแสดงในหน้าจอแสดงผลจะถูกส่งไปเก็บไว้ในโปรแกรมฐานข้อมูล Microsoft Access โดยหน้าจอแสดงผลนี้จะถูกติดตั้งอยู่ในบ้านหรือศูนย์ควบคุมของผู้ดูแลโรงเรือน และในส่วนของการรับส่งข้อมูลระหว่างหน้าจอแสดงผลกับโรงเรือนเลี้ยงไก่จะใช้โมดูล XBee เพื่อให้การสื่อสารระหว่างหน้าจอแสดงผลกับโรงเรือนเลี้ยงไก่นั้นสามารถรับส่งข้อมูลในระยะไกลได้อย่างรวดเร็วและแม่นยำ และเพื่อไม่ให้ผู้ดูแลต้องเสียเวลาในการเดินทางไปมาระหว่างศูนย์ควบคุมกับโรงเรือนเลี้ยงไก่

1.2 วัตถุประสงค์

1. เพื่อศึกษาการเขียนโปรแกรมไมโครคอนโทรลเลอร์
2. เพื่อศึกษาการทำงานระบบเซนเซอร์ อุณหภูมิ ความชื้น การเปิดและปิดอุปกรณ์ไฟฟ้า
3. เพื่อศึกษารายละเอียดการส่งข้อมูล และการสื่อสารระหว่างกันโดยใช้ XBee
4. เพื่อสร้างระบบที่สามารถควบคุม อุณหภูมิ ความชื้น การเปิดและปิดอุปกรณ์ไฟฟ้า ให้ใช้งานได้

1.3 ขอบเขตของปริญญานิพนธ์

1. สามารถควบควบคุมอุณหภูมิและความชื้น ให้มีสภาพอากาศคงที่เหมาะสมสำหรับการเลี้ยงไก่ได้จริง

2. สามารถส่งข้อมูลรายละเอียด การสื่อสารหากันได้อย่างถูกต้องแม่นยำโดยใช้โมดูล XBee

3. สามารถแสดงข้อมูลรายละเอียดและระบบให้สามารถทำงานได้โดยอัตโนมัติหรือควบคุมผ่านหน้าจอมอนิเตอร์ได้

บทที่ 2

ทฤษฎีและหลักการที่เกี่ยวข้อง

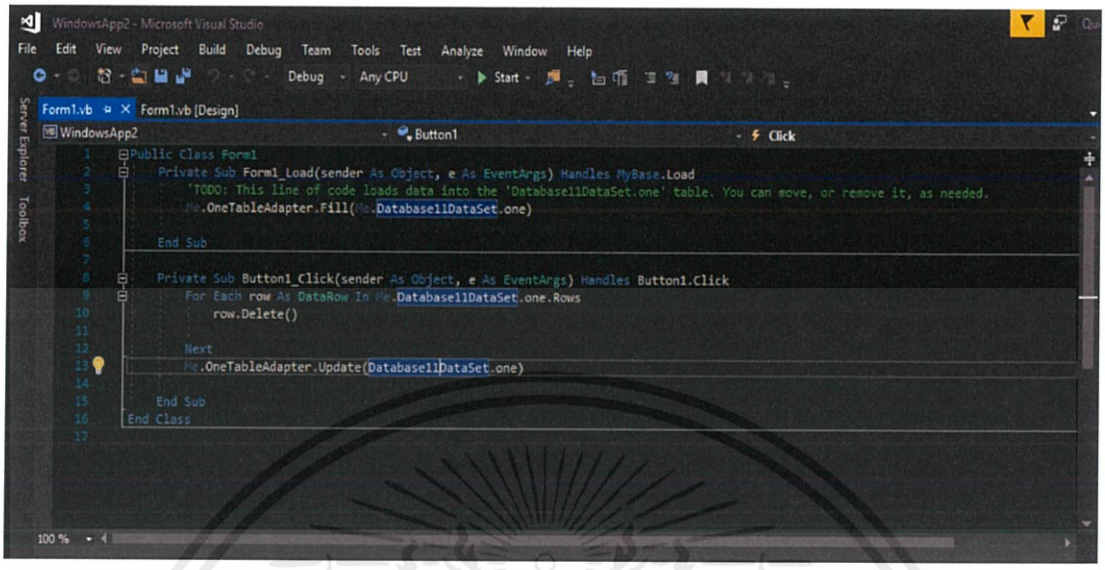
2.1 Microsoft Visual Studio

Microsoft Visual Studio คือ Integrated Development Environment (IDE) เป็นโปรแกรมประยุกต์ซอฟแวร์ ที่ใช้สำหรับพัฒนาโปรแกรมและยังอำนวยความสะดวกให้แก่ผู้เขียนโปรแกรมคอมพิวเตอร์ในการพัฒนาซอฟต์แวร์ประเภทต่าง ๆ พัฒนาขึ้นโดยบริษัทไมโครซอฟท์ Microsoft Visual Studio จะมีภาษาให้เลือกพัฒนาโปรแกรมแล้วแต่ความถนัดของผู้พัฒนาโปรแกรมหนึ่งในนั้นก็คือ Visual Basic

Visual Basic เป็นภาษาโปรแกรมที่สามารถสร้างโปรแกรมประยุกต์ได้ทั้งในแบบ Web-Base Application และ Stand-Alone Enterprise Applications โดยที่ Web-Base Application จะทำงานบน Web Server จะให้ส่งโปรแกรมเมื่อมีร้องขอจากผู้ใช้งานมาให้ที่เครื่องคอมพิวเตอร์ของผู้ใช้งานโดยโปรแกรมประเภทนี้จะต้องมีเบราว์เซอร์เป็นตัวช่วยในการแสดงผล ส่วน Stand - Alone Enterprise Applications จะเป็นการพัฒนาโปรแกรมที่สามารถทำงานบนระบบปฏิบัติการในเครื่องคอมพิวเตอร์ได้เลย เรียกอีกอย่างหนึ่งว่า Windows-Based โปรแกรมประเภท Windows-Based เป็นที่นิยมใช้ในงานธุรกิจขององค์กรต่าง ๆ ซึ่งหน้าจอเริ่มต้นของโปรแกรม Visual Studio จะแสดงดังรูปที่ 2.1

2.1.1 ภาษา Visual Basic

ภาษาวิชวลเบสิก (Visual Basic) หรือ VB เป็นหนึ่งในภาษาโปรแกรมที่นิยมใช้พัฒนาโปรแกรมที่ใช้ในด้านธุรกิจ วิชวลเบสิกพัฒนามาจากภาษาเบสิก และยังได้พัฒนาต่อเป็นภาษา VB.NET อีกด้วย วิชวลเบสิกสนับสนุน Rapid Application Development (RAD) ทั้งด้านการพัฒนาโปรแกรมประยุกต์แบบ Graphical User Interface (GUI) การเข้าถึงฐานข้อมูลโดยใช้การเชื่อมต่อแบบ DAO RDO หรือ ADO และการสร้าง ActiveX Control จุดเด่นอีกอย่างหนึ่งของวิชวลเบสิกคือนักเขียนโปรแกรมสามารถนำโปรแกรมประยุกต์หลายๆโปรแกรมมารวมกันในโปรแกรมเดียว และยังสามารถประยุกต์ใช้คอนโทรลและคอมโพเนนต์ที่วิชวลเบสิกเตรียมไว้ให้แล้วได้อีกด้วย



รูปที่ 2.1 หน้าจอโปรแกรม Visual Basic

2.2 Microsoft Access

Microsoft Access คือ โปรแกรมที่ใช้เพื่อพัฒนาระบบฐานข้อมูล มีตารางเก็บข้อมูลและสร้างแบบสอบถามได้ง่าย มีวัตถุคอนโทลให้เรียกใช้ในรายงานและฟอร์ม สร้างมาโครและโมดูลด้วยภาษาเบสิก เพื่อประมวลผลตามหลักภาษาโครงสร้าง สามารถใช้โปรแกรมนี้เป็นเพียงระบบฐานข้อมูลให้โปรแกรมจากภายนอกเรียกใช้ก็ได้

ไมโครซอฟท์แอคเซส (Microsoft Access) ต่างกับ วิชวลเบสิก (Visual Basic) หรือ วิชวลเบสิกดอทเน็ต (Visual Basic.Net) เพราะ วิชวลเบสิกไม่มีส่วนเก็บข้อมูลในตนเอง แต่สามารถพัฒนาโปรแกรมได้หลากหลาย เช่น พัฒนาโปรแกรมควบคุมอุปกรณ์ โปรแกรมประยุกต์ทางวิทยาศาสตร์ เกมส์ หรือเชื่อมต่อกับระบบฐานข้อมูลภายนอกเป็นภาษาที่เหมาะกับการพัฒนาโปรแกรมประยุกต์ (Application) ส่วนไมโครซอฟท์แอคเซสเหมาะสำหรับนักพัฒนาระบบฐานข้อมูลที่ไม่ต้องการโปรแกรมที่ซับซ้อน ความสามารถของโปรแกรมที่สำคัญคือสร้างตาราง แบบสอบถาม ฟอร์ม หรือรายงานในแฟ้มเดียวกันได้ ด้วยคุณสมบัติพื้นฐานและวิซซาร์ดจึงอำนวยให้พัฒนาโปรแกรมให้แล้วเสร็จได้ในเวลาอันสั้น มีเครื่องมือที่อำนวยความสะดวกในการพัฒนาระบบฐานข้อมูลอย่างครบถ้วน

2.3 ภาษา SQL

ภาษา SQL (Structured Query Language) หรือ ภาษาสอบถามเชิงโครงสร้าง เป็นภาษาทางด้านฐานข้อมูลที่ได้รับความนิยม ในการใช้งานกับฐานข้อมูลเชิงสัมพันธ์มากที่สุดในปัจจุบัน เริ่มต้นพัฒนาครั้งแรกโดยบริษัท IBM ซึ่งถือเป็นต้นแบบของภาษา SQL ในซอฟต์แวร์ที่เกี่ยวกับฐานข้อมูลต่าง ๆ แต่เนื่องจากความแตกต่าง ของการใช้ งานซอฟต์แวร์ของแต่ละบริษัท ดังนั้นในปี ค.ศ. 1986 American National Standards Institute (ANSI) จึงได้กำหนดมาตรฐานของคำสั่งภาษา SQL ขึ้น เพื่อให้ผลิตภัณฑ์ทางด้านฐานข้อมูลของ แต่ละบริษัทมีมาตรฐานเดียวกัน อย่างไรก็ตามถึงแม้ว่าจะมีการกำหนดมาตรฐานของ SQL แล้วก็ตาม แต่ภาษา SQL ของแต่ละผลิตภัณฑ์ก็ยังมีความแตกต่างกันบ้างเล็กน้อย

วัตถุประสงค์ของ SQL

1) สร้างฐานข้อมูลและโครงสร้างรีเลชั่น

2) สนับสนุนงานด้านการจัดการฐานข้อมูลพื้นฐาน เช่น การเพิ่ม การปรับปรุง การลบข้อมูลจากรีเลชั่น

3) สนับสนุนการค้นหา สืบถาม หรือคิวรี่ข้อมูลและการแปลงข้อมูลให้อยู่ในรูปสารสนเทศ

ประเภทของคำสั่งภาษา SQL

1) ภาษานิยามข้อมูล (Data Definition Language : DDL) เป็นกลุ่มคำสั่งที่ใช้ในการสร้างฐานข้อมูล การกำหนดโครงสร้างข้อมูลว่ามีคอลัมน์หรือแอตตริบิวต์ใด ชนิดข้อมูลเป็นประเภทใด รวมทั้งการจัดการด้านการเพิ่ม แก้ไข ลบ แอตตริบิวต์ต่าง ๆ ในรีเลชั่นและการสร้างดัชนี

2) ภาษาการจัดการข้อมูล (Data Manipulation Language : DML) เป็นกลุ่มคำสั่งที่ถือเป็นแกนสำคัญของภาษา SQL โดยกลุ่มคำสั่งเหล่านี้จะใช้ในการ Update เพิ่ม ปรับปรุงและการ Query ข้อมูลในฐานข้อมูล ซึ่งอาจเป็นชุดคำสั่งในลักษณะ Interactive SQL หรือ Embedded SQL ก็ได้

3) ภาษาควบคุมข้อมูล (Data Control Language : DCL) ซึ่งเป็นกลุ่มคำสั่งที่จะช่วยให้ผู้บริหารฐานข้อมูล (DBA) สามารถควบคุมฐานข้อมูลเพื่อกำหนดสิทธิการอนุญาต (Grant) หรือการยกเลิกการเข้าใช้ (Revoke) ฐานข้อมูล ซึ่งเป็นกระบวนการป้องกันความปลอดภัยในฐานข้อมูล รวมทั้งการจัดการทรานแซกชั่น (Transaction Management) แต่ละ DBMS จะมีการกำหนดชนิดข้อมูลซึ่งประกอบไปด้วยตัวแปรต่าง ๆ เช่น Numeric String Date Time เป็นต้น

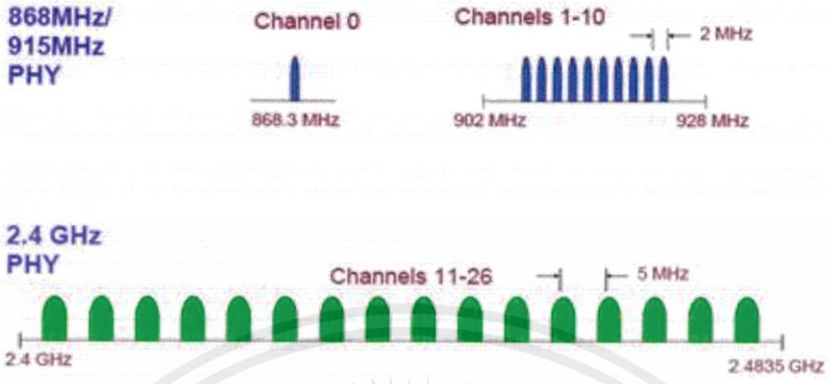
2.4 เทคโนโลยี Zigbee

Zigbee จะเป็นการสื่อสารไร้สายประเภทหนึ่งมีการกำหนดมาตรฐานโดย Zigbee Alliance โดยจะอ้างอิงตามมาตรฐานการทำงานของระบบเครือข่ายไร้สาย 802.15.4 ซึ่งข้อดีของโมดูล ZIgbee เองจะใช้พลังงานน้อย ขนาดอุปกรณ์มีขนาดเล็ก ซึ่งในปัจจุบันนิยมนำเอาโมดูล Zigbee มาใช้ในระบบ Wireless Sensor Network เนื่องจากใช้พลังงานน้อยจึงส่งผลเอื้อประโยชน์ให้สามารถนำไป วางไว้ตามจุดต่าง ๆ ในพื้นที่ทำงานได้อย่างสะดวก ซึ่งระบบนี้จะสามารถทำงานในร่ม กลางแจ้ง ทนแดด ทนฝน และอยู่ได้ด้วยแบตเตอรี่ก้อนเล็ก (เช่นถ่าน AA 2 ก้อน) นานเป็นเดือนเป็นปี เหมาะสมใช้งานกับพวกมอนิเตอร์ต่าง ๆ ซึ่งโมดูล Zigbee จะแสดงดังรูปที่ 2.2

รูปที่ 2.2 โมดูล Zigbee PRO S2C [3]

Zigbee จะมีความถี่หลัก ๆ ที่ใช้งานกันอยู่ 3 ย่านความถี่ ได้แก่ (1.) ย่านความถี่ 868 MHz มี 1 ช่องสัญญาณ อัตราการส่งข้อมูลจะอยู่ที่ 20 kbps (2.) ย่านความถี่ 915 MHz มี 10 ช่องสัญญาณ อัตราการส่งข้อมูลจะอยู่ที่ 40 kbps (3.) ย่านความถี่ 2.4 GHz มี 16 ช่องสัญญาณ อัตราการส่งข้อมูลจะอยู่ที่ 250 kbps

ในการใช้งาน Zigbee นั้นจะเห็นได้ว่ามีคลื่นความถี่ 3 ระดับนั้นเราไม่สามารถนำมาใช้งานร่วมกันได้ ในการเลือก Zigbee มาใช้งานนั้น ภายในระบบต้องเป็นย่านความถี่เดียวกันทั้งหมดโดยส่วนใหญ่แล้วผู้ใช้งานมักจะเลือกย่านความถี่ที่ 2.4 GHz มาใช้งาน โดยย่านความถี่ที่ใช้งานในตามมาตรฐาน Zigbee จะแสดงดังรูปที่ 2.3



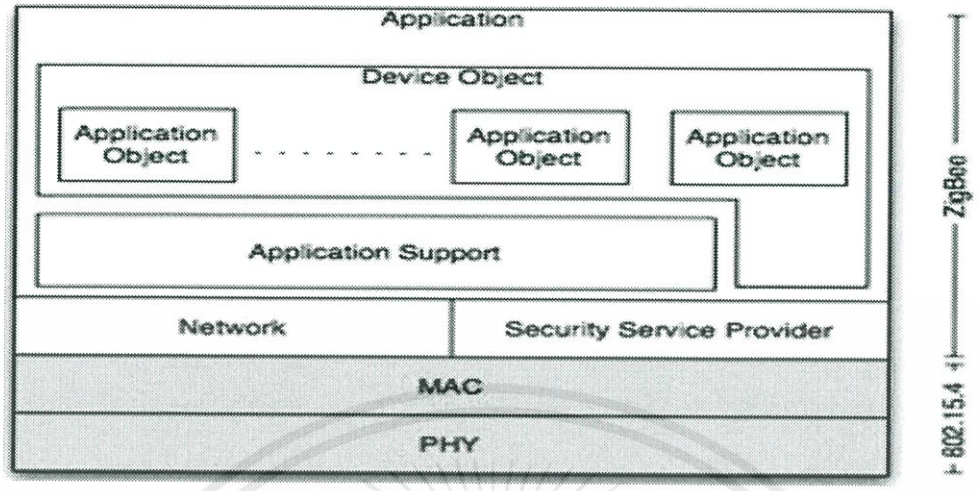
รูปที่ 2.3 ย่านความถี่ใช้งานตามมาตรฐาน Zigbee [4]

Zigbee จะใช้มาตรฐานการทำงานของระบบเครือข่ายไร้สาย 802.15.4 ซึ่งเป็นมาตรฐานการกำหนดการสื่อสารไร้สายแบบ WPAN (Wireless Personal Area Network) ซึ่งระดับชั้นที่ 1 จะเป็น Physical Layer ระดับชั้นที่ 2 จะเป็น Mac Address ส่วนด้านบนขึ้นไปจะเป็นส่วนที่ Zigbee จัดไว้ให้ ได้แก่ Security Service Provider กับ Network Zigbee ถูกออกแบบมาเฉพาะในส่วนของ Application Layer Application Support Layer และ Network Layer เท่านั้นซึ่งแต่ละเลเยอร์จะมีหน้าที่ดังต่อไปนี้ และ Zigbee Frame Work จะถูกแสดงดังรูปที่ 2.4

1) Application Layer เป็นชั้นที่มีส่วนของ Endpoint อยู่ เรียกว่า Application framework โดยมี Zigbee Device Object (ZDO) ทำหน้าที่ในการจัดการในการเข้าถึงและใช้งาน Application Layer

2) Application Support Layer ทำหน้าที่ในการสร้างเฟรมของ Application Layer และทำหน้าที่ในการรับส่งข้อมูล รวมถึงการจัดการด้านต่าง ๆที่เกี่ยวข้องกับ Application Layer

3) Network Layer ทำหน้าที่ในการ Routing ข้อมูลต่าง ๆ จากต้นทางไปยังปลายทางที่อาจอยู่ภายในเครือข่ายเดียวกัน หรือต่างเครือข่ายกัน



รูปที่ 2.4 Zigbee Frame Work [4]

จากที่กล่าวมา Zigbee จะสามารถสร้างเป็นเครือข่ายได้เพราะอิงมาตรฐานตาม IEEE 802.15.4 และมีการจัดการในแบบของ Zigbee ในเลเยอร์ถัดไป ทั้งนี้ IEEE 802.15.4 แบ่งชนิดอุปกรณ์ในเครือข่ายออกเป็น 2 ประเภท คือ FFD (Full Function Device) ซึ่งหมายถึงอุปกรณ์ที่สามารถทำงานได้ทุกอย่างในเครือข่าย และ RFD (Reduce Function Device) ซึ่งหมายถึงอุปกรณ์ที่ถูกลดความสามารถการทำงานในเครือข่าย

Zigbee ได้แบ่งตามลักษณะการทำงาน 3 แบบ คือ

1) Coordinator มีหน้าที่สร้างการสื่อสารเชื่อมโยงเครือข่ายระหว่าง End Device กับ Router หรือ กับ Coordinator ด้วยกัน หรือ Coordinator กับ Router กำหนดแอดเดรสให้กับอุปกรณ์ที่อยู่ในวงเครือข่าย ไม่ให้ซ้ำกัน ดูแลจัดการเรื่องการ Routing เส้นทาง ซึ่งเทียบได้กับ FFD

2) End Device เป็นอุปกรณ์ปลายทางสุด ซึ่งจะใช้รับสัญญาณจากเซ็นเซอร์ที่ปลายทาง โดยที่ใช้พลังงานต่ำในการทำงาน เทียบได้กับ RFD หรือ FFD บางกรณี ขึ้นอยู่กับเซ็นเซอร์ที่ใช้

3) Router มีหน้าที่รับส่งข้อมูล ในเส้นทางต่าง ๆ ของเครือข่าย ซึ่งเทียบได้กับ FFD

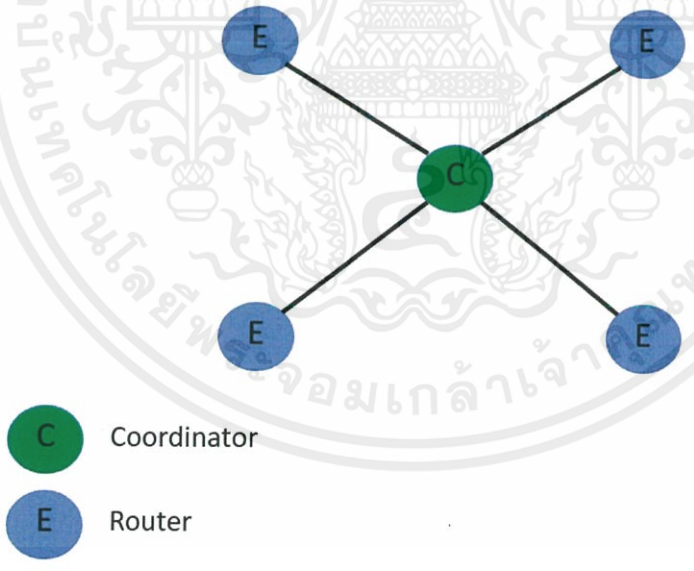
XBee เป็นการพัฒนาชุดการสื่อสาร โดยอ้างอิงจากมาตรฐานของ Zigbee เพื่อสร้างระบบเครือข่ายโดยจะประกอบด้วยอุปกรณ์ที่มี Microcontroller, RF และ IC อยู่ภายใน ทำหน้าที่เป็น

อุปกรณ์รับ-ส่งสัญญาณแบบ Half Duplex ย่านความถี่ 2.4 GHz มีการจัดการโดยใช้พลังงานต่ำ ใช้งานง่าย มี Interface ที่ใช้รับและส่งข้อมูลเป็น UART, TTL ในการใช้งานนั้นเราสามารถสร้างระบบเครือข่ายโดยการตั้งค่าผ่านทางโปรแกรม X-CTU ได้เลย

ในการสร้างโครงข่ายไร้สายของ XBee นั้น จะต้องประกอบด้วยโหนดจำนวนอย่างน้อยที่สุด 2 ชนิด คือ Coordinator และโหนดลูกข่าย ชนิดใดชนิดหนึ่ง (Router/End device) จึงจะสามารถสื่อสารและทำงานในรูปแบบของ PAN (Personal Area Network) ได้ โดย ZigBee สามารถแบ่งรูปแบบ เครือข่ายได้เป็น 3 รูปแบบ ดังนี้

1) Star Topology

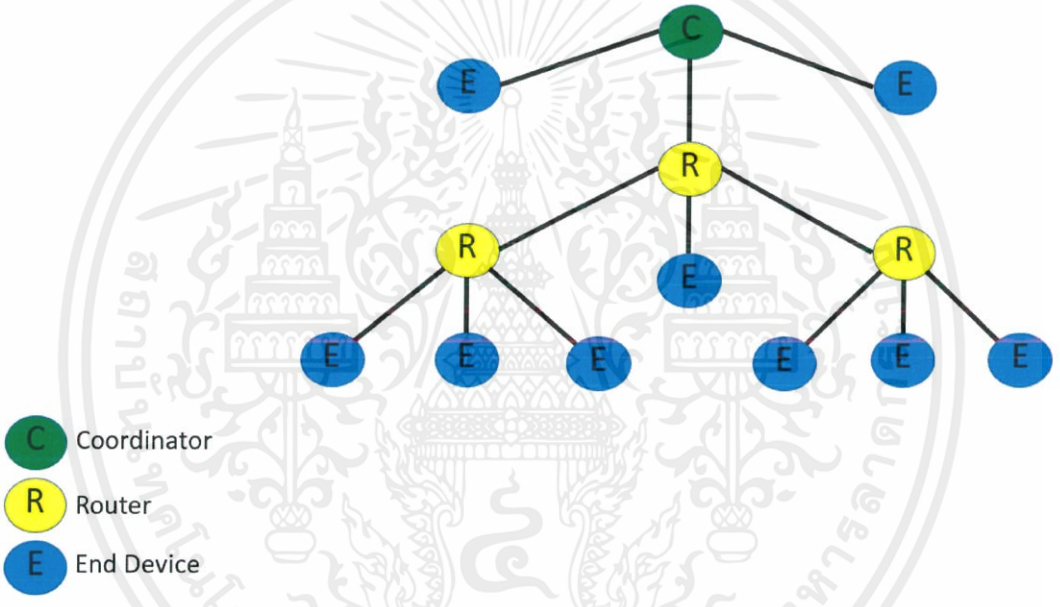
รูปแบบการเชื่อมต่อเครือข่ายแบบ Star ดูจะเป็น Topology ที่ง่ายที่สุด ตัวเครือข่ายจะประกอบไปด้วย ตัว Hub ที่ทำหน้าที่เป็นศูนย์กลางของเครือข่าย เชื่อมต่อกับลูกข่ายหลายๆตัว ในกรณีของโมดูล XBee ตัว Coordinator จะมารับหน้าที่ในการเป็น Hub ของเครือข่าย ส่วนลูกข่าย อาจจะเป็น Router หรือ End Device ก็ได้แล้วแต่ความเหมาะสมของงาน โดย Topology แบบ Star จะแสดงดังรูปที่ 2.5



รูปที่ 2.5 Topology แบบ Star

2) Tree Topology

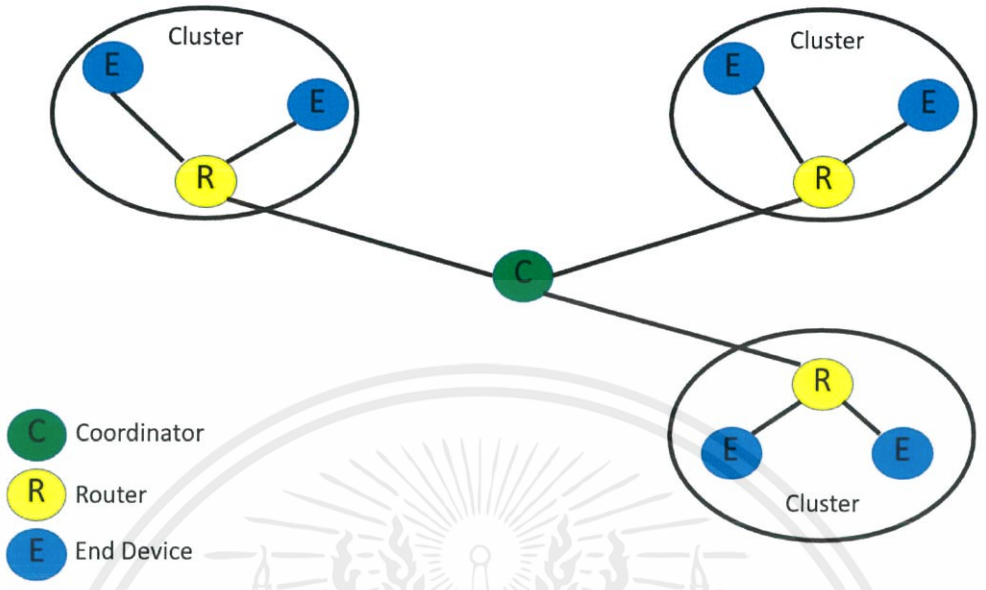
เป็นรูปแบบการเชื่อมต่อของเครือข่ายในลักษณะของแก่นรากที่แตกกิ่งก้านสาขาออกไป โดยที่แก่นรากของเครือข่าย หรือ Root นั้นอาจจะประกอบไปด้วย Coordinator, Router และ End Device ก็ได้ แต่กิ่งก้านที่แตกออกไปจากตัวราก นั้นจะเรียกว่า Children ซึ่งเฉพาะโหนด Coordinator กับ Router เท่านั้นที่จะสามารถมี Children เพิ่มเติมได้ อีกทั้ง Children แต่ละตัวจะสามารถสื่อสารได้กับ Parent ของใครของมันเท่านั้น โดย Topology แบบ Tree จะแสดงดังรูปที่ 2.6



รูปที่ 2.6 Topology แบบ Tree

2.1) Cluster Tree Topology

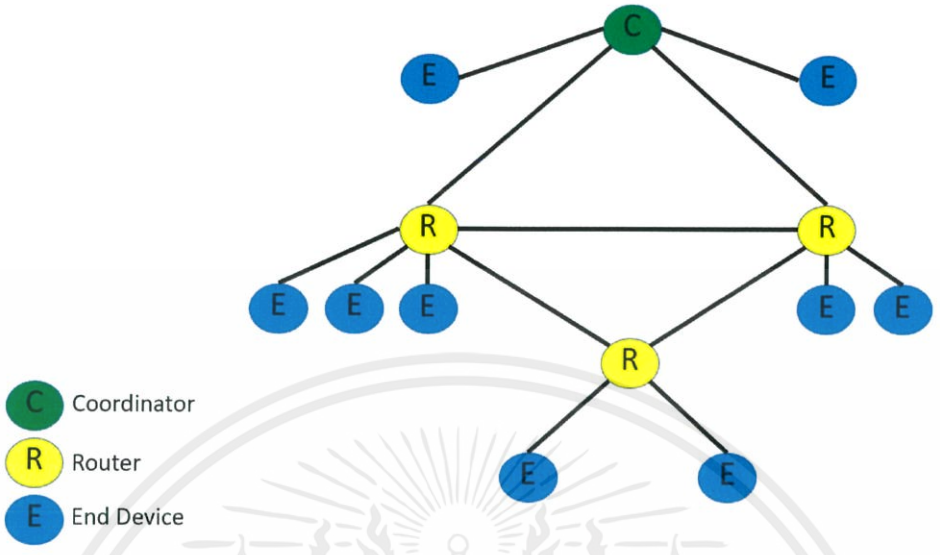
เป็นรูปแบบการเชื่อมโยงเครือข่ายที่ต่อยอดมาจาก Tree Topology ลักษณะแนวคิดคล้ายกัน เพียงแต่ว่าโหนด Parent ที่มี Children หลายๆตัวเชื่อมต่ออยู่ จะถูกจัดเป็นกลุ่มก้อน เรียกว่า Cluster ซึ่งแต่ละกลุ่ม ก็จะมี Cluster ID ประจำของใครของมัน ทำให้ง่ายต่อการระบุตัวตนยิ่งขึ้น โดย Topology แบบ Cluster Tree จะแสดงดังรูปที่ 2.7



รูปที่ 2.7 Topology แบบ Cluster Tree

3) Mesh Topology

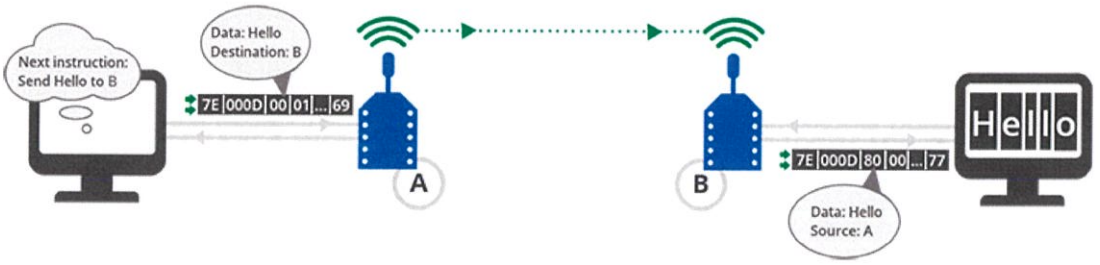
เป็นรูปแบบการเชื่อมโยงเครือข่ายที่มีประสิทธิภาพ และนำไปใช้งานอย่างแพร่หลาย ในเครือข่ายจะประกอบไปด้วยตัว Coordinator, Router และ End Device หลาย ๆ ตัว การรับส่งแต่ละ Packet ข้อมูลนั้นจะกระทำในลักษณะของ Multihop คือข้อมูลจะ Hop ไปเรื่อย ๆ จนกว่าจะถึงปลายทางที่ต้องการ โดย Topology แบบ Mesh จะแสดงดังรูปที่ 2.8



รูปที่ 2.8 Topology แบบ Mesh

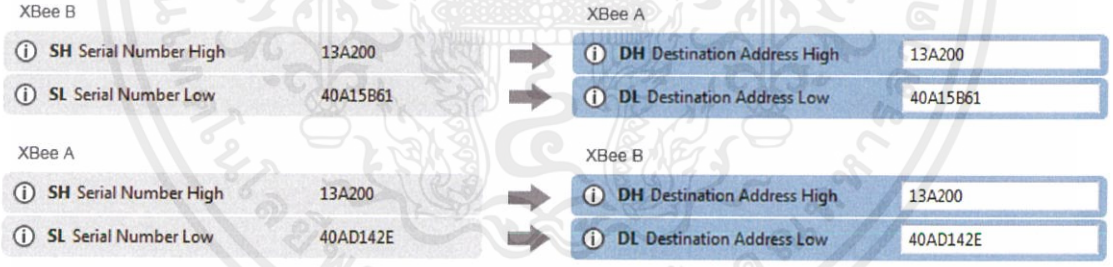
โหมดการทำงานของ XBee มีอยู่ 2 แบบ คือ Application Transparent Mode (AT Mode) และ Application Programming Interface (API Mode)

1) Application Transparent Mode (AT Mode) เป็นโหมดการทำงานที่ตัวโมดูลจะทำการส่งผ่านข้อมูลที่ได้รับไปยังที่อยู่ปลายทาง จะมีการติดต่อกันและรับ-ส่งข้อมูลกันตลอดเวลา ตัวอย่างเช่น จากรูปที่ 2.8 ถ้าหากตัวไมโครคอนโทรลเลอร์ ส่งข้อความผ่าน Serial Interface ไปยังโมดูล XBee โมดูล A ว่า "Hello" ตัวโมดูล A ก็จะรับข้อความนั้น ๆ และทำการส่งต่อไปยังโมดูล B ที่อยู่ระยะห่างออกไป ฝั่งโมดูล B เมื่อได้รับข้อความ ก็จะทำการส่งข้อความนั้น ๆ ผ่านออกไปทาง Serial Interface ไปยังอุปกรณ์ต่อพวงที่ต่ออยู่กับตัวมันเอง พูดอีกอย่างหนึ่งคือ โมดูล XBee ทำหน้าที่เป็นเหมือนสายส่งข้อมูล Serial Interface นั่นเอง



รูปที่ 2.9 แสดงการส่งข้อมูลของ XBee โหมด AT [5]

ซึ่งลักษณะการทำงานเช่นนี้ ทำให้ตัว XBee ที่จะส่งข้อมูลจำเป็นต้องรู้แอดเดรส (Address) ของผู้รับด้วย ดังนั้นเมื่อใช้งานในโหมดนี้ จะต้องทำการตั้งค่าแอดเดรส XBee ที่ต้องการให้สื่อสารกันเอาไว้ โดยลักษณะของแอดเดรสปลายทางที่ XBee เก็บนั้นจะมีขนาด 64-bit โดยจะต้องทำการโปรแกรมเอดเดรสนี้ในพารามิเตอร์สองตัว คือ Destination Address High (DH) และ Destination Address Low (DL) ซึ่งถ้าต้องการให้อุปกรณ์ A และ B สื่อสารกันได้ ก็จะต้องตั้งค่าแอดเดรสปลายทางของ A ให้ตรงกับ MAC Address ของ B และตั้งแอดเดรสปลายทางของ B ให้ตรงกับ MAC Address ของ A เช่นกัน ดังตัวอย่างในรูปที่ 2.10



รูปที่ 2.10 การตั้งค่าแอดเดรสปลายทางให้ XBee สามารถสื่อสารกันได้ในโหมด AT [5]

ค่าเริ่มต้นการทำงานของ XBee นั้น ถูกกำหนดมาให้ทำงานใน Transparent Mode แต่การที่ XBee ทำงานที่โหมดนี้ก็ทำให้มีข้อเสียบางอย่าง เช่น การเขียนหรืออ่านค่าคอนฟิก (Configuration Data) ของ XBee จะต้องทำการเข้าสู่คอมมานโหมด (Command Mode) ซึ่งเมื่อเข้าสู่คอมมานโหมดแล้ว ข้อมูลทุกอย่างที่ผ่านขาอินพุท (Serial Input) เข้ามานั้นจะถือว่าเป็นคำสั่งสำหรับตั้งค่าอุปกรณ์ทั้งหมด หรือถ้าเกิดว่าอยากส่งข้อมูลไปให้กับ XBee ตัวอื่น ก็จะต้องทำการตั้ง

ค่าปลายทางใหม่ และการทำงานใน Transparent Mode นี้ทำให้อุปกรณ์ตัวรับ ไม่สามารถทราบได้ว่าผู้ส่งมีแอดเดรสอะไร ถ้าเกิดต้องการทราบที่อยู่ผู้ส่งด้วย ผู้ใช้จำเป็นต้องสร้างโพรโทคอลใหม่ขึ้นมาเอง

2) Application Programming Interface (API Mode) เป็นโหมดการสื่อสารที่ข้อมูลที่รับส่งกันจะถูกจัดแจงให้อยู่ในรูปของ Packet ข้อมูลที่มีการกำหนดค่าต่าง ๆ วิธีนี้จะช่วยทำให้ผู้ใช้งานสามารถรับส่งข้อมูลกันระหว่างโมดูลได้ในรูปแบบที่ซับซ้อนมากยิ่งขึ้น โดยจากรูปที่ 2.11 เมื่อฝั่งส่ง A ต้องการส่งข้อมูลไปยังฝั่งรับ B ฝั่งส่งจะทำการแปลงข้อมูลที่ต้องการส่งให้อยู่ในรูปของ Packet ข้อมูล ซึ่งจะประกอบไปด้วย ที่อยู่ผู้รับ ที่อยู่ผู้ส่ง ประเภทของข้อมูลที่ต้องการนำส่ง ข้อความที่ต้องการนำส่ง บิทตรวจสอบความถูกต้อง เป็นต้น เมื่อ Packet ข้อมูลถูกประกอบขึ้นเรียบร้อย ตัวโมดูล A จะทำการส่งข้อมูลไปยังโมดูล B เมื่อโมดูล B ได้รับ ก็จะส่งข้อมูลไปยังอุปกรณ์ต่อพ่วง ซึ่งอาจจะเป็นคอมพิวเตอร์ หรือบอร์ดไมโครคอนโทรลเลอร์ใด ๆ จากนั้นก็จะเข้าสู่กระบวนการถอด Packet ข้อมูลออก นำข้อความจริง ๆ ที่ต้องการไปใช้งาน

รูปที่ 2.11 แสดงการส่งข้อมูลของ XBee โหมด API [5]

Start Delimiter	Length		Frame Data							Checksum
			Frame type	Data						
1	2	3	4	5	6	7	8	9	...	n	n + 1
0x7E	MSB	LSB	API frame type	Frame-type-specific data						Single byte

MSB: most-significant byte, **LSB**: least-significant byte

รูปที่ 2.12 โครงสร้างของเฟรมที่ใช้ภายใต้ XBee โหมด API [6]

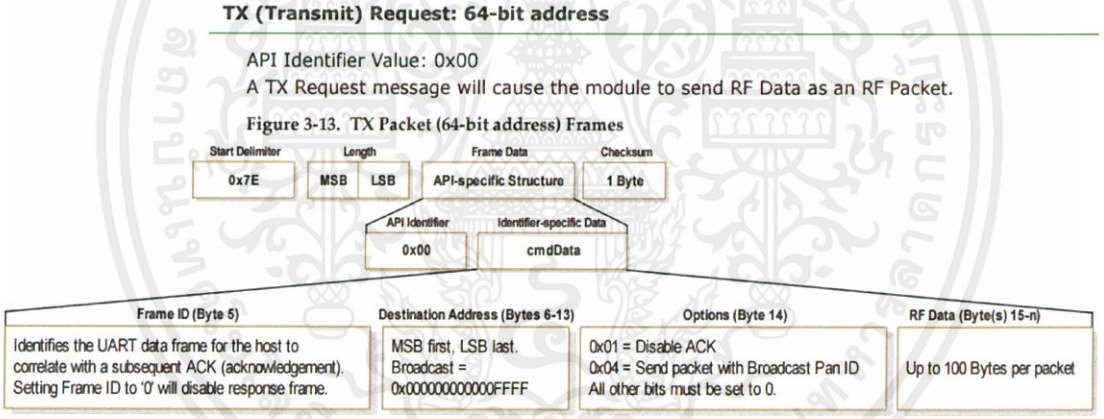
จากรูปที่ 2.12 API ใน 1 ชุดคำสั่งจะแบ่งออกเป็น 4 กลุ่มด้วยกันคือ

1) Start Delimiter เป็นส่วนเริ่มต้นของ API จะใช้ 0X7E เป็นตัวคั่นเพื่อบอกให้รู้ว่านี่คือจุดเริ่มต้นของ API มีขนาด 1 Byte

2) Length คือ จำนวน Byte ของ Frame Data มีขนาด 2 Byte

3) Frame Data จะแบ่งออกเป็นสองส่วน คือ ส่วน Byte แรก จะเป็น API Frame Type หรือ API Identifier ซึ่งจะมีค่าเฉพาะแล้วแต่การทำงาน และส่วนของเดต้าภายในหรือ Frame-Type-Specific Data ซึ่งมีค่าและความหมายแตกต่างกันตาม API Identifier ดังตัวอย่างในรูปที่ 2.13 และ 2.14

4) Checksum เป็นตัวที่เอาไว้ตรวจความถูกต้องของ Frame Data ที่ได้รับมาว่ามีค่าถูกหรือไม่ โดยค่าของ Checksum สามารถคำนวณได้โดยนำข้อมูลใน Frame Data มาบวกกันและนำ 0xFF – LSB (ของผลบวก) ค่าที่ได้จากผลลบก็คือค่า Checksum นั่นเอง



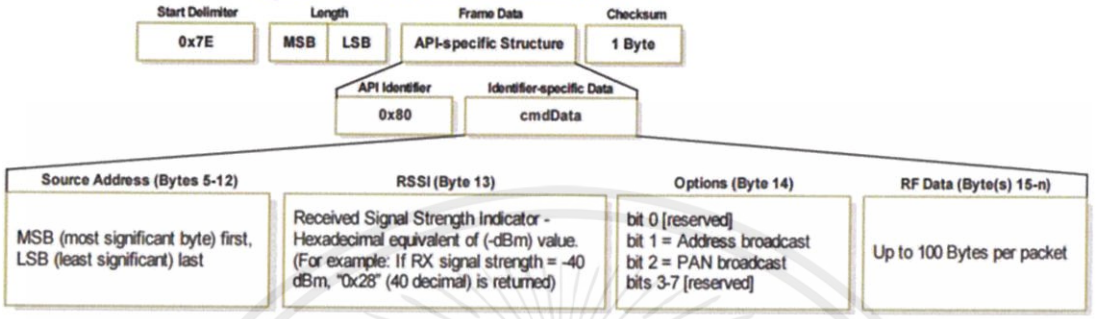
รูปที่ 2.13 API Frame สำหรับส่งข้อมูลด้วยแอดเดรส 64-bit [6]

RX (Receive) Packet: 64-bit Address

API Identifier Value: 0x80

When the module receives an RF packet, it is sent out the UART using this message type.

Figure 3-16. RX Packet (64-bit address) Frames



รูปที่ 2.14 API Frame ที่จะได้รับ เมื่อต้นทางส่งข้อมูลมาให้ด้วยแอดเดรส 64-bit [6]

ตัวอย่างการหา Length และ Checksum จาก Protocol API ที่ใช้ส่งข้อมูล "7E 00 16 10 01 00 13 A2 00 40 3D 8A 28 FF FE 00 00 54 78 44 61 74 61 30 41 56" สามารถแยกออกตามกลุ่ม 4 กลุ่มได้ดังนี้ โดยแต่ละกลุ่มของชุดข้อมูล API Mode จะแสดงดังรูปที่ 2.15

รูปที่ 2.15 แต่ละกลุ่มของชุดข้อมูล API mode [6]

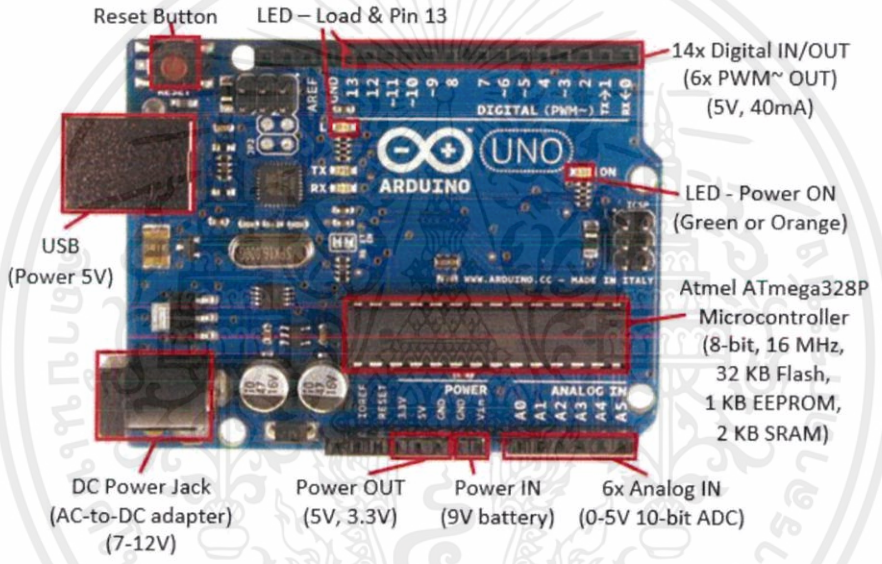
1) Length = จำนวน Byte ที่อยู่ในกลุ่มของ Frame Data ซึ่งจากตัวอย่างสามารถนับได้ 22 Byte (เมื่อแปลง 22 เป็น Hex จะได้เท่ากับ 0x16 เนื่องจาก Length มีขนาด 2 Byte ที่ เหลือจึงเป็น 00)

2) Checksum = 0xFF – (ค่าของแต่ละ Byte ในFrame Data บวกกัน) ยกตัวอย่างเช่น 0x6A9 =10+01+ 00+ 13+ A2+ 00+ 40+ 3D+ 8A+28 +FF+ FE+ 00+ 00+ 54+ 78+ 44+ 61+ 74+ 61+ 30+ 41 นำ 0xFF - 0x06A9 = FFFFFA56

3) Checksum = 0x56

2.5 Arduino Uno R3

เป็นบอร์ด Arduino ที่ได้รับความนิยมมากที่สุด เนื่องจากราคาไม่แพง ส่วนใหญ่โปรเจคและ Library ต่าง ๆ ที่พัฒนาขึ้นมา Support จะอ้างอิงกับบอร์ดนี้เป็นหลัก และข้อดีอีกอย่างคือกรณีที่ MCU เสีย ผู้ใช้งานสามารถซื้อมาเปลี่ยนเองได้ง่าย ส่วนที่เป็น Software คือภาษา Arduino เป็นภาษาสำหรับเขียนโปรแกรมควบคุมไมโครคอนโทรลเลอร์ มีไวยากรณ์แบบเดียวกับ C/C++ และมี Arduino IDE เป็นเครื่องมือสำหรับเขียนโปรแกรมด้วยภาษา Arduino ประมวลผลโปรแกรม (Compile) และอัพโหลดโปรแกรมลงบอร์ด (Upload)



รูปที่ 2.16 Layout & Pinout Arduino Uno R3 [7]

ส่วนประกอบและตำแหน่งต่าง ๆ ของบอร์ด Arduino Uno R3 จะแสดงดังรูปที่ 2.16 และคุณสมบัติต่าง ๆ ของบอร์ด Arduino Uno R3 จะแสดงดังตารางที่ 2.1 ซึ่งแต่ละตำแหน่งนั้นจะมีหน้าที่ดังต่อไปนี้

1) USBPort : ใช้สำหรับต่อกับ Computer เพื่ออับโหลดโปรแกรมเข้า MCU และจ่ายไฟให้กับบอร์ด

2) Reset Button : เป็นปุ่ม Reset ใช้กดเมื่อต้องการให้ MCU เริ่มการทำงานใหม่

3) Digital IN/OUT Port : Digital I/O ตั้งแต่ขา D0 ถึง D13 นอกจากนี้ บาง Pin จะทำหน้าที่อื่นๆ เพิ่มเติมด้วย เช่น Pin0,1 เป็นขา Tx,Rx Serial, Pin3,5,6,9,10 และ 11 เป็นขา PWM

4) MCU : Atmega328 เป็น MCU ที่ใช้บนบอร์ด Arduino

5) Analog IN Port : นอกจากจะเป็น Digital I/O แล้ว ยังเปลี่ยนเป็น ช่องรับสัญญาณอนาล็อก ตั้งแต่ขา A0-A5

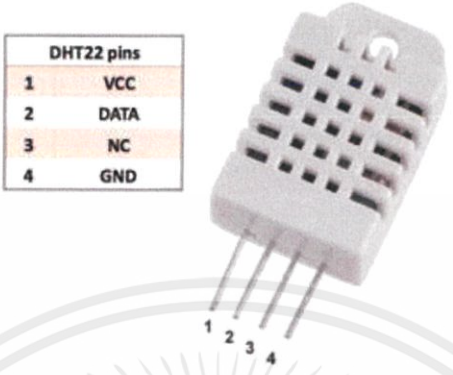
6) Power Port : ไฟเลี้ยงของบอร์ดเมื่อต้องการจ่ายไฟให้กับวงจรภายนอก ประกอบด้วยขาไฟเลี้ยง +3.3 V, +5V, GND, Vin

7) Power Jack : รับไฟจาก Adapter โดยที่แรงดันอยู่ระหว่าง 7-12 V

ตารางที่ 2.1 คุณสมบัติต่าง ๆ ของบอร์ด Arduino Uno R3

ไมโครคอนโทรลเลอร์	ATmega328
แหล่งจ่ายไฟ	5 V
แรงดันไฟฟ้าเข้า	7-12 V
ขา Digital output/ Digital input	14 ขา
ขา Analog input	6 ขา
กระแสไฟฟ้า DC ต่อขา I/O	40 mA
กระแสไฟฟ้าออก DC สำหรับขา 3.3 V	50 mA
Flash Memory	32 KB (ATmega328)
SRAM	2 KB (ATmega328)
EEPROM	1 KB (ATmega328)
Clock Speed	16 MHz

2.6 เซ็นเซอร์วัดอุณหภูมิและความชื้นสัมพัทธ์ DHT22 (AM2302)



รูปที่ 2.17 เซ็นเซอร์ DHT22 [8]

อุปกรณ์เซ็นเซอร์สำหรับวัดอุณหภูมิและความชื้นสัมพัทธ์ DHT22 เป็นอุปกรณ์ที่สามารถนำมาประยุกต์ใช้งานทางด้านระบบสมองกลฝังตัวได้หลากหลาย เช่น การวัดและควบคุมอุณหภูมิและความชื้น ระบบบันทึกข้อมูลเกี่ยวกับอุณหภูมิและความชื้นในห้อง เป็นต้น DHT22 เป็นเซ็นเซอร์สำหรับวัดอุณหภูมิและความชื้นที่มีความแม่นยำสูงในการวัด สามารถวัดได้ในย่านอุณหภูมิ ตั้งแต่ -40 องศาเซลเซียส ถึง +80 องศาเซลเซียส ความแม่นยำน้อยกว่า +-0.5เซลเซียส และวัดความชื้นสัมพัทธ์ได้ในย่าน 0-100%RH ความแม่นยำ +-2-5%RH สามารถวัดได้ละเอียดในระดับทศนิยม 1 ตำแหน่ง (0.1) ใช้งานได้นานและทนทาน เหมาะสำหรับนำไปใช้ในงานวัดที่ต้องการความแม่นยำสูง คนความร้อนเย็นวัดในย่านติดลบได้ ซึ่งเซ็นเซอร์ DHT22 มีอยู่ 4 ขา แต่ขาที่ต้องต่อใช้งานนั้นมีอยู่แค่ 3 ขาเท่านั้น รายละเอียดแต่ละขาจะแสดงดังตารางที่ 2.2 และเซ็นเซอร์ DHT22 ดังแสดงรูปที่ 2.17

ตารางที่ 2.2 หน้าที่ของแต่ละขาของ DHT22

ขา	ชื่อ	หน้าที่
1	VDD	ต่อกับไฟเลี้ยง (3.3V-5V)
2	SDA	ส่งข้อมูล
3	NC	-
4	GND	กราวด์

2.7 อุณหภูมิและความชื้นสัมพัทธ์

2.7.1 อุณหภูมิ

อุณหภูมิ (Temperature) คือค่าตัวเลขที่มีความสัมพันธ์กับระดับพลังงานจลน์ภายในอะตอมในระบบองศาสัมบูรณ์ (Absolute Temperature) ระดับพลังงานที่อุณหภูมิ 0 เคลวิน (-273°C) อะตอมไม่มีพลังงานอยู่เลย ดังนั้นอนุภาคทุกอย่างภายในอะตอมหยุดนิ่ง แม้กระทั่งอิเล็กตรอนก็ไม่โคจรรอบนิวเคลียส แต่เมื่ออะตอมได้รับพลังงานจนมีระดับอุณหภูมิสูงขึ้น อิเล็กตรอนก็จะเคลื่อนที่รอบนิวเคลียสและยกระดับชั้นวงโคจรสูงขึ้น ถ้าหากอะตอมได้รับพลังงานจนมีระดับอุณหภูมิสูงขึ้นไปอีก อิเล็กตรอนอาจจะยกตัวหลุดจากวงโคจรกลายเป็นประจุ (Ion) อย่างไรก็ตามพื้นผิวโลกและชั้นบรรยากาศที่เราอยู่อาศัยมีอุณหภูมิประมาณ 139 - 331 เคลวิน (-89°C ถึง 58 °C) ที่ระดับพลังงานขนาดนี้ อะตอมจะไม่อยู่อย่างโดดเดี่ยวแต่จะเกาะตัวกันเป็นโมเลกุล การเคลื่อนที่ของโมเลกุลทำให้เกิดรูปแบบของพลังงานจลน์ซึ่งเรียกว่า “ความร้อน” (Heat) ในปัจจุบันสเกลอุณหภูมิที่นิยมใช้มี 3 ระบบ ดังนี้

1) องศาฟาเรนไฮต์

ปี ค.ศ.1714 กาเบรียล ฟาเรนไฮต์ (Gabrial Fahrenheit) นักฟิสิกส์ชาวเยอรมันได้ประดิษฐ์เทอร์มอมิเตอร์ซึ่งบรรจุปรอทไว้ในหลอดแก้ว เขาพยายามทำให้ปรอทลดต่ำสุด (0°F) โดยใช้น้ำแข็งและเกลือผสมน้ำ เขาพิจารณาจุดหลอมละลายของน้ำแข็งเท่ากับ 32°F และจุดเดือดของน้ำเท่ากับ 212°F ปัจจุบันสเกลฟาเรนไฮต์เป็นที่นิยมแต่ในประเทศสหรัฐอเมริกา

2) องศาเซลเซียส

ปี ค.ศ.1742 แอนเดอส์ เซลเซียส (Anders Celsius) นักดาราศาสตร์ชาวสวีเดน ได้ออกแบบสเกลเทอร์มอมิเตอร์ให้อ่านได้ง่ายขึ้น โดยมีจุดหลอมละลายของน้ำแข็งเท่ากับ 0°C และจุดเดือดของน้ำเท่ากับ 100°C สเกลเซลเซียสจึงได้รับความนิยมใช้กันทั่วโลก อย่างไรก็ตามทั้งสเกลฟาเรนไฮต์และเซลเซียสอ้างอิงอยู่กับจุดเยือกแข็งและจุดเดือดของน้ำ ซึ่งเป็นสิ่งที่ใช้ทั่วไปในชีวิตประจำวัน

3) องศาสัมบูรณ์

ในคริสศตวรรษที่ 19 ลอร์ด เคลวิน (Lord Kelvin) นักฟิสิกส์ชาวอังกฤษ ผู้ค้นพบความสัมพันธ์ระหว่างความร้อนและอุณหภูมิว่า ณ อุณหภูมิ -273°C อะตอมไม่มีพลังงาน และไม่มีอุณหภูมิใดต่ำไปกว่านี้ เขาจึงกำหนดให้ 0 K = -273°C (ไม่ต้องใช้เครื่องหมาย ° กำกับหน้าอักษร K) เนื่องจากนักวิทยาศาสตร์ศึกษาความสัมพันธ์และการถ่ายเทพลังงานของสสาร

ดังนั้นในวงการวิทยาศาสตร์จึงนิยมใช้สเกลองศาสัมบูรณ์ มากกว่าองศาฟาเรนไฮต์และองศาเซลเซียส

ความสัมพันธ์ของสเกลอุณหภูมิทั้ง 3 ระบบมีดังนี้

1) อุณหภูมิในหน่วย Kelvin : $T(K) = T(°C) + 273.15$

2) อุณหภูมิในหน่วย Fahrenheit : $T(°F) = 1.8T(°C) + 32$

3) อุณหภูมิในหน่วย Celcius : $T(°C) = (T(°F) - 32) / 1.8$

2.7.2 ความชื้นสัมพัทธ์

ความชื้น (Humidity) หมายถึง จำนวนไอน้ำที่มีอยู่ในอากาศ ความชื้นของอากาศมีการเปลี่ยนแปลงอยู่ตลอดเวลา จะมากหรือน้อยขึ้นอยู่กับความดันและอุณหภูมิ ความชื้นสัมพัทธ์ (Relative Humidity) หมายถึง "อัตราส่วนของ ปริมาณไอน้ำที่มีอยู่จริงในอากาศ ต่อ ปริมาณไอน้ำที่จะทำให้อากาศอิ่มตัว ณ อุณหภูมิเดียวกัน" หรือ "อัตราส่วนของความดันไอน้ำที่มีอยู่จริง ต่อความดันไอน้ำอิ่มตัว" ค่าความชื้นสัมพัทธ์แสดงในรูปของร้อยละ (%)

2.8 Relay

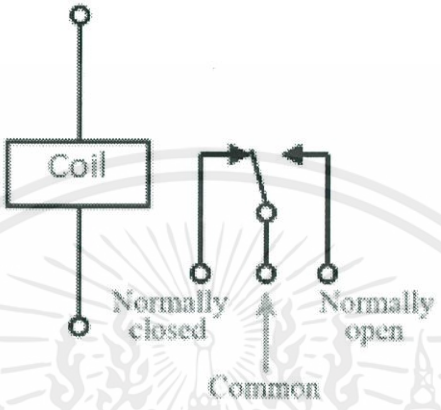
Relay เป็นอุปกรณ์ไฟฟ้าชนิดหนึ่ง ซึ่งทำหน้าที่ตัดต่อวงจรแบบเดียวกับสวิตช์ โดยควบคุมการทำงานด้วยไฟฟ้า Relay มีหลายประเภท ตั้งแต่ Relay ขนาดเล็กที่ใช้ในงานอิเล็กทรอนิกส์ทั่วไป จนถึง Relay ขนาดใหญ่ที่ใช้ในงานไฟฟ้าแรงสูง โดยมีรูปร่างหน้าตาแตกต่างกันออกไป แต่มีหลักการทำงานที่คล้ายคลึงกัน สำหรับการนำ Relay ไปใช้งาน จะใช้ในการตัดต่อวงจร ทั้งนี้ Relay ยังสามารถเลือกใช้งานได้หลากหลายรูปแบบ

ภายใน Relay จะประกอบไปด้วยขดลวดและหน้าสัมผัส ซึ่งสัญลักษณ์ในวงจรไฟฟ้าของ Relay จะเป็นไปตามรูปที่ 2.18

1) หน้าสัมผัส NC (Normally Close) เป็นหน้าสัมผัสปกติปิด โดยในสภาวะปกติหน้าสัมผัสนี้จะต่อเข้ากับขา COM (Common) และจะลอยหรือไม่สัมผัสกันเมื่อมีกระแสไฟฟ้าไหลผ่านขดลวด

2) หน้าสัมผัส NO (Normally Open) เป็นหน้าสัมผัสปกติเปิด โดยในสภาวะปกติจะลอยอยู่ ไม่ถูกต่อกับขา COM (Common) แต่จะเชื่อมต่อกันเมื่อมีกระแสไฟไหลผ่านขดลวด

3) ขา COM (Common) เป็นขาที่ถูกใช้งานร่วมกันระหว่าง NC และ NO ขึ้นอยู่กับว่าขณะนั้นมีกระแสไฟฟ้าไหลผ่านขดลวดหรือไม่ หน้าสัมผัสใน Relay 1 ตัวอาจมีมากกว่า 1 ชุด ขึ้นอยู่กับผู้ผลิตและลักษณะของงานที่ถูกนำไปใช้



รูปที่ 2.18 สัญลักษณ์ในวงจรไฟฟ้าของ Relay [9]

ซึ่งโครงงานนี้ได้ใช้ Relay 8 ช่อง (Relay Module 8 Channels) ควบคุมในการเปิด-ปิดของอุปกรณ์ต่าง ๆ

2.8.1 Relay Module 8 Channels

บอร์ด Relay ขนาด 8 ช่อง มีเอาต์พุตคอนเน็คเตอร์ที่รีเลย์เป็น NO/COM/NC สามารถใช้กับโหลดได้ทั้งแรงดันไฟฟ้า DC และ AC โดยใช้สัญญาณในการควบคุมการทำงานด้วยสัญญาณโลจิก TTL (5V) มีคุณสมบัติดังนี้

1) รีเลย์เอาต์พุตแบบ SPDT จำนวน 8 ช่อง

2) ควบคุมไฟ DC ได้สูงสุด 24VDC/7A

3) ควบคุมไฟ AC ได้สูงสุด 220VAC/7A และ 110VAC/10A

4) ระดับสัญญาณอินพุตควบคุมแบบ TTL ทำงานด้วยสัญญาณแบบ Active High

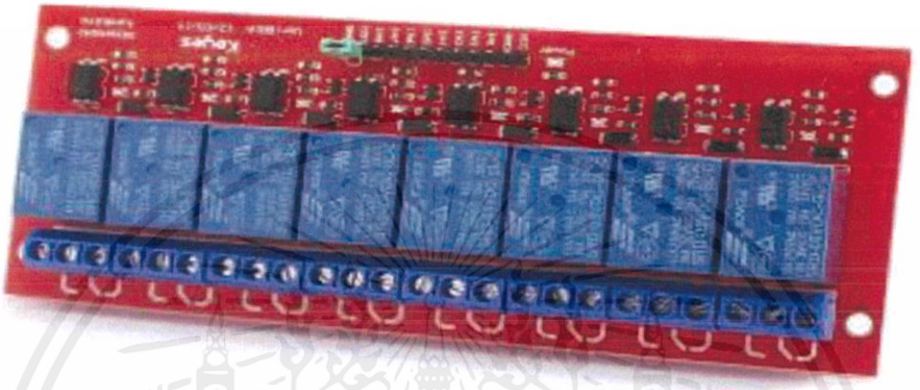
5) มี OPTO-ISOLATED เพื่อแยกกราวด์ส่วนของสัญญาณควบคุมกับไฟที่ขับรีเลย์ออกจากกัน

6) มีจัมพ์เปอร์สำหรับเลือกว่าจะใช้กราวด์ร่วมหรือแยก

7) มี LED แสดงสถานะ การทำงานของรีเลย์และแสดงสถานะของบอร์ด

8) ขนาดรูยึดบอร์ด 3mm

9) ขนาด (L x W x H) : 135 x 55 x 20 mm



รูปที่ 2.19 Relay 8 Channel [10]

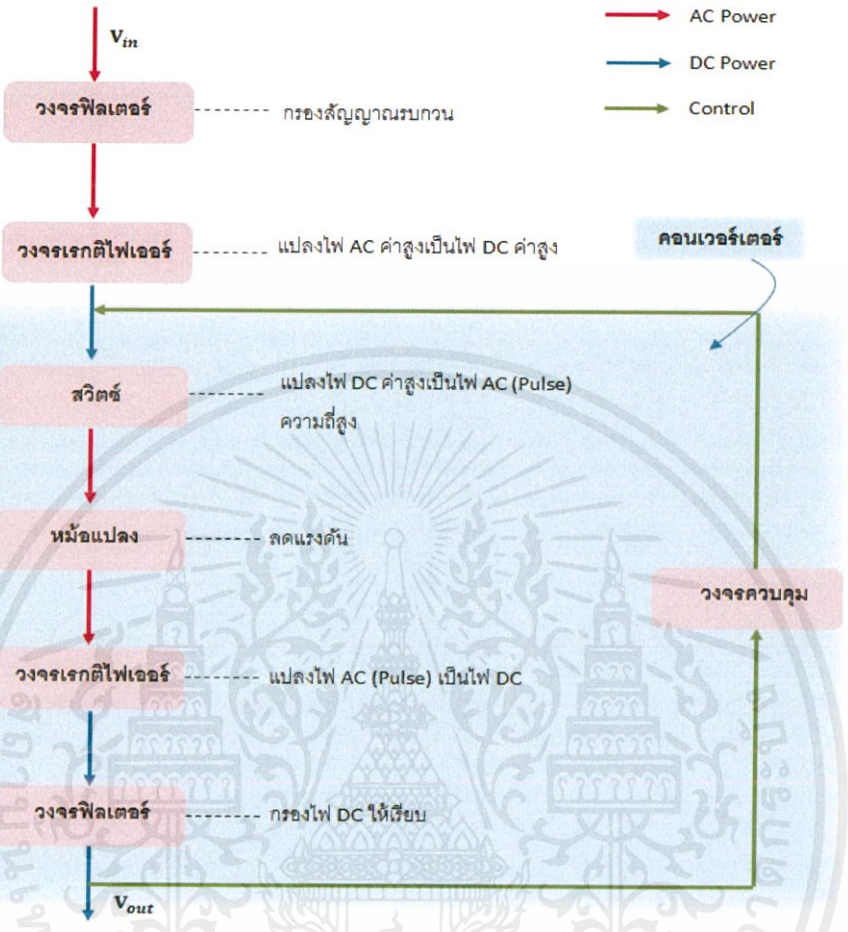
2.9 Switching Power Supply

Switching Power Supply นั้นถูกสร้างขึ้นมาเพื่อใช้ในงานอิเล็กทรอนิกส์เป็นแหล่งจ่ายไฟให้กับอุปกรณ์ต่าง ๆ และสามารถเปลี่ยนแรงดันไฟจากไฟสลับโวลต์สูงให้เป็นแรงดันไฟตรงโวลต์ต่ำได้ ซึ่งองค์ประกอบพื้นฐานนั้นโดยทั่วไปจะคล้ายกันและสิ่งที่สำคัญที่สุดขององค์ประกอบนี้ คือ คอนเวอร์เตอร์ โดย Switching Power Supply จะประกอบด้วย 3 ส่วนใหญ่ๆ คือ

1) วงจรฟิลเตอร์และเรกติไฟเออร์ ทำหน้าที่แปลงแรงดันไฟสลับเป็นไฟตรง

2) คอนเวอร์เตอร์ ทำหน้าที่แปลงไฟตรงเป็นไฟสลับความถี่สูง และแปลงกลับเป็นไฟตรงโวลต์ต่ำ

3) วงจรควบคุม ทำหน้าที่ควบคุมการทำงานของคอนเวอร์เตอร์ เพื่อให้ได้แรงดันเอาต์พุตตามต้องการ



รูปที่ 2.20 แผนผัง Switching Power Supply [11]

การคงค่าแรงดันจะทำโดยการป้อนค่าแรงดันที่ Output กลับมายังวงจรควบคุม เพื่อควบคุมให้การนำกระแสมากขึ้นหรือน้อยลงตามการเปลี่ยนแปลงของแรงดันที่ Output ซึ่งจะมีผลทำให้แรงดัน Output คงที่ได้

รูปที่ 2.21 Switching Power Supply [11]

2.10 โรงเรือนเลี้ยงไก่แบบปิด

เนื่องจากประเทศไทยเป็นประเทศที่อยู่ในเขตร้อนมีอุณหภูมิของอากาศค่อนข้างสูง ส่วนใหญ่ผู้เลี้ยงสัตว์มักสร้างโรงเรือนเป็นโรงเรือนเปิด ทั้งนี้เพื่อต้องการให้อากาศภายในโรงเรือนมีการหมุนเวียนและระบายอากาศเป็นการลดความร้อนภายในโรงเรือนได้ดี โรงเรือนเปิดไม่สามารถควบคุมอุณหภูมิได้อุณหภูมิของโรงเรือนจะผันแปรไปตามสภาพของอากาศภายนอกโรงเรือน ช่วงหน้าร้อนอากาศ จะร้อนมาก สัตว์เลี้ยงบางชนิด เช่น ไก่เนื้อ อาจทนอากาศร้อนไม่ไหว เพื่อหลีกเลี่ยงจากอากาศร้อนและต้องการควบคุมอุณหภูมิของโรงเรือนจึงได้มีการคิดค้นโรงเรือนระบบปิดขึ้นโดยใช้หลักการระบายความร้อนด้วยน้ำและใช้พัดลมเป็นตัวถ่ายเทอากาศ โดยมีแผ่นรังผึ้ง (Cooling Pad) ที่ปล่อยน้ำไหลผ่านจนเปียกชุ่ม เมื่อเดินพัดลมซึ่งอยู่ในแนวตรงกันข้ามกับแผ่นรังผึ้ง อากาศภายนอกจะถูกดูดผ่านแผ่นรังผึ้งเข้าภายในโรงเรือน ภายในโรงเรือนจะเย็นสบายโดยใช้หลักการระเหยของน้ำ นอกจากนี้โรงเรือนระบบปิดยังสามารถป้องกันโรคได้อย่างดีโดยเฉพาะโรคไข้หวัดนก ซึ่งการติดตั้งและระบบการทำงานของโรงเรือนเลี้ยงแบบปิด มีดังนี้

2.10.1 หลักการทำงานของระบบหล่อเย็นในโรงเรือนเลี้ยงไก่แบบปิด

ระบบหล่อเย็นในโรงเรือนเลี้ยงไก่แบบปิดนี้มีหลักการทำงาน ซึ่งไม่ยุ่งยากสลับซับซ้อนมากนัก ถ้าหากเข้าใจระบบการทำงานแล้วผู้เลี้ยงสัตว์ก็สามารถที่จะติดตั้งระบบหล่อเย็นได้ที่โรงเรือนของตนเอง โดยได้สรุปหลักการปฏิบัติเกี่ยวกับระบบหล่อเย็นไว้ดังนี้

1) ขนาดของโรงเรือน โรงเรือนมีขนาดมาตรฐานคือ กว้าง 12 เมตร และยาว 120 เมตร

2) หลังคา หลังคาเป็นแบบจั่วชั้นเดียว หลังคาจั่วสูงจากพื้น 4 เมตร โครงสร้างทั้งหมดทำด้วยเหล็กฉาก ยกเว้นแบบซึ่งใช้ไม้เนื้อแข็ง วัสดุที่นำมาใช้คลุมหลังคา โรงเรือน ทำด้วยแผ่นสังกะสีฉาบด้วยกาลวาไนส์ (Galvanized) ภายใต้หลังคามุงด้วยฉนวนใยแก้ว (Micro – Fiber) กันความร้อน ใต้ฉนวนกันความร้อนบุด้วยแผ่นพลาสติกไวนิล (Vinyl) เพื่อป้องกันการแผ่รังสีความร้อนจากหลังคาไม่ให้ลงมาในโรงเรือนได้ ถัดลงมาจากแผ่นกันความร้อนยังมีแผ่นไม้อัดที่ติดตั้งใต้เพดานขวางตามความยาวของโรงเรียน เรียกว่า แผ่นชิงลม (Spoiler) คิดเป็นระยะทุก 12 เมตร เพื่อดักลมด้านบนให้พัดผ่านด้านล่างอย่างสม่ำเสมอและทั่วถึง

3) ผนังโรงเรือน ผนังด้านหน้าและท้ายโรงเรือนปิดทึบ ส่วนผนังด้านข้างทั้ง 2 ข้าง ก่ออิฐสูงประมาณ 60 ซม. เปิดช่องลมและปิดด้วยผ้าม่านพลาสติกขนาด 1.20 เมตร และมีตาข่ายอย่างดีล้อมรอบผนังด้านข้าง เปิดประตูหน้า – หลัง และด้านกลางของโรงเรือนด้วย

4) แผ่นรังผึ้ง แผ่นรังผึ้งเป็นส่วนสำคัญที่ปรับให้อุณหภูมิในโรงเรือนลดลง ซึ่งทำด้วยกระดาษสังเคราะห์พิเศษมีความทนทาน มีความหนา 2 ขนาด คือ ขนาดหนา 10 เซนติเมตร และ 15 เซนติเมตร ความสูงของแผ่นรังผึ้ง 180 เซนติเมตร ความยาวประมาณ 15 เมตร และ 21.6 เมตร ต่อโรงเรือน การติดแผ่นรังผึ้งจะติดด้านเดียวหรือ 2 ด้านก็ได้ แต่การติด 2 ด้านนั้น การไหลเวียนของอากาศจะทั่วถึงและสม่ำเสมอดีกว่าติดด้านเดียวและไม่ต้องติดพัดลมเสริมภายในอีก

5) พัดลม พัดลมที่ใช้จะติดตั้งอยู่ในโรงเรือนด้านหลัง (ด้านท้าย) ตรงข้ามแผ่นรังผึ้ง มีขนาดเส้นผ่าศูนย์กลาง 48 นิ้ว

6) ระบบควบคุมอุณหภูมิในโรงเรือน การควบคุมอุณหภูมิภายในโรงเรือนนั้นใช้พัดลมและแผ่นรังผึ้ง โดยมีตัวควบคุมอุณหภูมิ (Thermostats) อยู่ ถ้าโรงเรือนมีพัดลม 10 เครื่อง จะมีตัวควบคุมอุณหภูมิอยู่ 11 ตัว เพราะอีก 1 ตัวนั้นสำหรับควบคุมอุณหภูมิ การปิดเปิด น้ำของเครื่องปั๊มน้ำในการปล่อยให้น้ำไหลผ่านแผ่นรังผึ้ง โดยในสภาพที่อุณหภูมิทั่วไปพัดลมจะเปิดทำงาน 1 เครื่อง อยู่ตลอดเวลาและพัดลมที่เหลืออีกจะทำงานเมื่ออุณหภูมิสูงกว่าที่เครื่องควบคุมอุณหภูมิดังต่อไปนี้

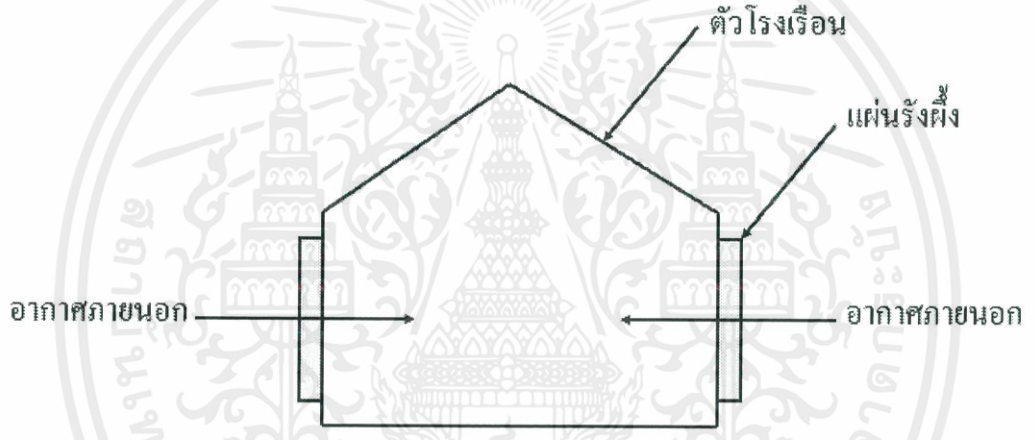
สูงกว่า 60 องศาฟาเรนไฮต์ พัดลมเครื่องที่ 2 จะทำงาน
สูงกว่า 72 องศาฟาเรนไฮต์ พัดลมเครื่องที่ 3 จะทำงาน
สูงกว่า 74 องศาฟาเรนไฮต์ พัดลมเครื่องที่ 4 จะทำงาน
สูงกว่า 76 องศาฟาเรนไฮต์ พัดลมเครื่องที่ 5 จะทำงาน
สูงกว่า 78 องศาฟาเรนไฮต์ พัดลมเครื่องที่ 6 จะทำงาน
สูงกว่า 80 องศาฟาเรนไฮต์ พัดลมเครื่องที่ 7 จะทำงาน
สูงกว่า 82 องศาฟาเรนไฮต์ พัดลมเครื่องที่ 8 จะทำงาน

ในกรณีที่โรงเรือนมีพัดลม 10 เครื่อง จะตั้งตัวควบคุมพัดลมที่อุณหภูมิช่วงระหว่าง 60 – 72 (องศาฟาเรนไฮต์) อีก 2 เครื่อง เมื่ออากาศเปลี่ยนแปลงไป ระบบอัตโนมัติที่ติดตั้งไว้จะทำงานเพื่อปรับสภาพอากาศและอุณหภูมิในโรงเรือนให้คงที่ตลอดเวลา และพัดลมจะเป็นตัวดูอากาศผ่านรังผึ้งซึ่งมีความเย็นเข้าไปแทนที่อากาศร้อนภายในซึ่งจะถูกดูดออกไปอีกทางหนึ่ง เมื่ออากาศเย็นเข้าไปแทนที่จะทำให้อุณหภูมิภายในลดลงได้จากปกติถึง 7 องศาเซสเซียส หรือมากกว่านั้น แต่ถ้าช่วงไหนอากาศเย็นสบายอยู่แล้ว พัดลมดูดอากาศบางตัวจะหยุดทำงานไปโดยอัตโนมัติ และม่านอะลูมิเนียมที่หลังพัดลม ก็จะเปิดเพื่อป้องกันอากาศเข้าออกโรงเรือน และเมื่ออุณหภูมิเริ่มสูงขึ้นม่านอะลูมิเนียมก็จะเปิดพัดลม ก็จะทำงานอีกครั้ง ในสภาวะที่อากาศภายนอกโรงเรือนเย็นอาจจะไม่จำเป็นต้องใช้น้ำช่วยปรับอากาศเลยก็ได้ เพียงแค่ใช้พัดลมระบายอากาศอย่างเดียวก็พอ เนื่องจากอากาศภายในเย็นพอเพียง

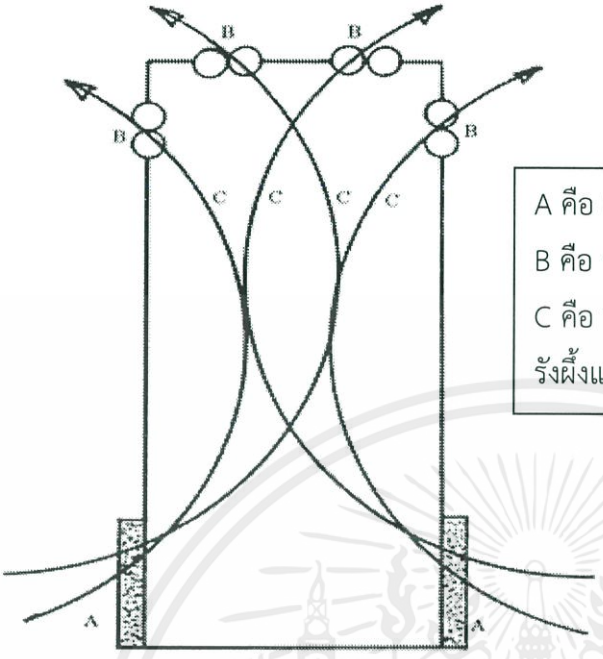
7) ระบบการไหลเวียนของน้ำในแผ่นรังผึ้ง การไหลเวียนของน้ำในแผ่นรังผึ้งนี้มีความสำคัญต่ออายุการใช้งานของแผ่นรังผึ้ง น้ำต้องสะอาดและไม่ทำลายแผ่นรังผึ้ง บริเวณที่น้ำไหลไปไม่ทั่วถึงจะเริ่มอุดตัน แนะนำให้ความเร็วของน้ำไหล 6 ลิตร/นาที/พื้นที่แผ่นรังผึ้ง 1 ตารางเมตร (ความหนา 10 เซนติเมตร) และ 9 ลิตร/นาที/พื้นที่แผ่นรังผึ้ง 1 ตารางเมตร (ความหนา 15 เซนติเมตร) การทำงานของน้ำจะมาจากเครื่องปั๊มน้ำขนาด 0.75 แรงม้า 1 เครื่อง ปั๊มจากบ่อเก็บน้ำด้านล่างข้าง ๆ แผ่นรังผึ้งมักทำเป็นบ่อซีเมนต์ขนาดกว้างประมาณ 3 เมตร ยาว 6 เมตร สูง 1.5 เมตร เมื่อสูบน้ำขึ้นมาปล่อยใส่แผ่นรังผึ้งให้น้ำไหลผ่านลงมา น้ำที่ไหลผ่านจะไหลไปรวมกันที่รางรวมน้ำข้างล่างและไหลลงบ่อเก็บน้ำเดิมอีกเป็นวงจรหมุนเวียนไป แผ่นรังผึ้งมีหน้าที่ทำให้เกิดพื้นที่ผิวของการระเหยของน้ำหรือเพิ่มการระเหยและเมื่ออากาศพัดผ่านก็จะหอบเอาความเย็น ความชื้นเข้าไปในโรงเรือนด้วยโดยอากาศที่ร้อนเมื่อพัดผ่านจะกลายเป็นอากาศเย็นทันที

8) ปัญหาการอุดตันของแผ่นรังผึ้ง อายุการใช้งานของแผ่นรังผึ้งขึ้นอยู่กับคุณภาพ

ของน้ำที่ใช้ ปกติน้ำจะมีปริมาณของแร่ธาตุต่าง ๆ แตกต่างกันตามแหล่งที่มาและมีแต่น้ำสะอาดและบริสุทธิ์เท่านั้นที่สามารถผ่านแผ่นรังผึ้งและระเหยเข้าไปในโรงเรือนได้ ส่วนแร่ธาตุต่าง ๆ จะต้องตกค้างอยู่ที่แผ่นรังผึ้ง ทำให้แผ่นรังผึ้งอุดตันเมื่อใช้ไปนาน ๆ โดยเฉพาะแร่ธาตุพวกแคลเซี่ยม (Calcium) ในส่วนของแผ่นรังผึ้งหรือที่เรียกว่า Cooling Pad สามารถทำความสะอาดได้อย่างง่ายด้วยการผสมน้ำยาฆ่าเชื้อเข้าไปในน้ำ ที่ปล่อยลงมาจากท่อพีวีซีเพื่อให้สัมผัสกับแผ่นแพดและพ่นฆ่าเชื้อโรคให้ทั่วอีกครั้งก็ใช้ได้แล้วไม่จำเป็นต้องถูด้วยแปรงหรือทำความสะอาดละเอียดนัก เนื่องจากแผ่นรังผึ้งนี้ทำด้วยกระดาษสังเคราะห์ที่ค่อนข้างจะบอบบาง อาจจะฉีกขาดได้และถ้าหากน้ำที่ใช้ในฟาร์มไม่สะอาดพอจะมีหินปูนมาเกาะตามแผ่นรังผึ้งมาก จึงต้องทำความสะอาดด้วยกรดไฮโดรคลอริก



รูปที่ 2.22 ลักษณะของอากาศที่เข้าไปในโรงเรือนโดยผ่านแผ่นรังผึ้ง



A คือ แผ่นรังผึ้งทำความเย็น
B คือ พัดลมสำหรับระบายอากาศ
C คือ อากาศที่เข้ามาในโรงเรือน โดยผ่านแผ่นรังผึ้งและระบายออกนอกโรงเรือนโดยพัดลม

รูปที่ 2.23 การหมุนเวียนของอากาศในโรงเรือน [12]

9) น้ำ เป็นปัจจัยสำคัญต่อระบบนี้ นอกจากตัวพัดลมสำหรับระบายอากาศแล้ว น้ำจะขาดเสียไม่ได้ หากไม่มีน้ำระบบนี้ก็ไม่เกิดขึ้น และน้ำที่นำมาใช้ต้องเป็นน้ำที่ไม่มีตะกอนต่าง ๆ หรือมีพวกธาตุเหล็กมากเกินไป ถ้าน้ำมีตะกรันหรือธาตุเหล็กมาก ๆ จะต้องนำมากรองก่อนที่จะนำมาผ่านรังผึ้งเพราะถ้าตะกอนไปจับแผ่นรังผึ้ง จะทำให้แผ่นรังผึ้งตัน

10) อุณหภูมิ ภายในโรงเรือนจะขึ้นอยู่กับความชื้นสัมพัทธ์ภายในโรงเรือน ตามปกติ อุณหภูมิที่วัดได้ภายในโรงเรือนจะต่ำกว่าอุณหภูมิภายนอก 3-5 องศาเซลเซียส

2.10.2 ข้อดีและข้อเสียของโรงเรือนเลี้ยงไก่แบบปิด

ทั้งโรงเรือนเปิดและโรงเรือนระบบปิดก็มีข้อดีข้อเสียต่างกันออกไป ผู้เลี้ยงสัตว์จะต้องตัดสินใจว่าควรจะเลือกใช้โรงเรือนระบบใด แต่ในภาพรวมแล้วโรงเรือนระบบปิดจะช่วยแก้ปัญหาเรื่องอากาศร้อน และป้องกันโรคได้ดีกว่าโรงเรือนเปิด ข้อดีและข้อเสียของโรงเรือนระบบปิดมีดังนี้

ข้อดีของระบบทำความเย็นด้วยแผ่นรังผึ้งในโรงเรือนเลี้ยงไก่แบบปิดมีดังนี้

1) ลดความเครียดที่เกิดจากความร้อนและทำให้ไก่สุขภาพดีขึ้น

2) ในพ่อแม่พันธุ์ไก่กระทงจะให้ผลผลิตสูงขึ้น

3) ลดอัตราการตาย โดยเฉพาะอย่างยิ่งในช่วงอากาศร้อนจัด

4) ใช้พัดลมน้อยกว่าเมื่อเทียบกับโรงเรียนแบบเปิด และเป็นการประหยัด ค่ากระแสไฟฟ้า

5) สามารถใช้ร่วมกับระบบทึบแสงเพื่อเลี้ยงไก่พ่อ–แม่พันธุ์ได้อย่างมีประสิทธิภาพมากกว่าโรงเรือนแบบเปิด

6) การหมุนเวียนอากาศภายในโรงเรือนสม่ำเสมอมาก อากาศบริสุทธิ์จากภายนอกจะผ่านแผ่นรังผึ้งเข้ามาภายในโรงเรือนและระบายเอาอากาศเสียออกไปภายนอกโรงเรือนโดยพัดลมใช้เวลาสั้น ๆ เท่านั้น เป็นการลดปัญหาระดับแอมโมเนียในโรงเรือนได้

7) อัตราการเจริญเติบโตดีกว่าและประสิทธิภาพการเปลี่ยนอาหารเป็นเนื้อดีในไก่กระทง

8) ลดการใช้ยาปฏิชีวนะ

9) สามารถเลี้ยงไก่ได้มากขึ้นกว่าโรงเรือนแบบเปิด เมื่อเทียบกับพื้นที่เท่ากัน

10) สามารถควบคุมอุณหภูมิ ความชื้น การระบายอากาศและแสงสว่างในโรงเรือนได้

ข้อเสียของระบบทำความเย็นด้วยแผ่นรังผึ้งในโรงเรือนเลี้ยงไก่แบบปิดมีดังนี้

1) การลงทุนในระยะเริ่มต้นสูงและมีค่าใช้จ่ายที่ต้องตามมาอีก ได้แก่ ค่าไฟ ค่าน้ำ และค่าสึกหลอของอุปกรณ์ (ขึ้นอยู่กับประสิทธิภาพในการดูแลของแต่ละฟาร์ม)

2) เนื่องจากระบบควบคุมอุณหภูมิของโรงเรือนขึ้นอยู่กับปัจจัยต่าง ๆ หลายชนิด เช่น ชนิดและขนาดของแผ่นให้ความเย็น ระดับความชื้นภายนอกและภายในโรงเรือน พื้นที่และความหนาแน่นของการเลี้ยง จำนวนพัดลมและการวางผังตำแหน่งของพัดลม เพื่อให้เกิดการหมุนเวียนอากาศในระดับความเร็วลมที่เหมาะสมและทั่วถึงทั้งโรงเรือน การไม่เข้าใจในระบบและการปฏิบัติที่ไม่ถูกต้องจะนำไปสู่ความเสียหายที่มากกว่าการเลี้ยงในโรงเรือนระบบเปิด

3) การเลี้ยงสัตว์ที่หนาแน่นเกินขอบเขตความสามารถในการจัดการเลี้ยงดู สภาพของโรงเรือนและจำนวนอุปกรณ์ ก่อให้เกิดปัญหาสุขภาพและการให้ผลผลิตที่ต่ำกว่า มาตรฐานได้

4) การพิจารณาถึงขนาดของโรงเรือนในระบบปิดของฟาร์มนั้นต้องคำนึงถึงความสามารถในการจัดการแบบเข้าหมดออกหมด (All-In All-Out) ของฟาร์มได้ โรงเรือนที่มีขนาดใหญ่เกินไปไม่สามารถที่จะย้ายสัตว์ออกได้หมดภายในระยะเวลาหนึ่ง และทำให้ต้องมีการนำสัตว์ รุ่นต่อมาทยอยเข้าไปในโรงเรือน ในขณะที่สัตว์ชุดก่อนยังมีการเลี้ยงอยู่ในโรงเรือนที่จะส่งผลต่อสุขภาพของสัตว์ใน

รุ่นใหม่อย่างแน่นอน

5) โรงเรือนในระบบปิดถูกออกแบบให้ช่วยในการเพิ่มผลผลิตของสัตว์ ส่วนการป้องกันโรคหรือการติดเชื้อของสัตว์ควรเน้นที่การป้องกันฟาร์มในระบบปิดมากกว่า

ข้อควรระมัดระวังในการใช้ระบบหล่อเย็นในโรงเรือนเลี้ยงไก่แบบปิด

1) ต้องมีเครื่องกำเนิดไฟฟ้าขนาดใหญ่ไว้สำรองทุกฟาร์มและระบบสัญญาณเตือนต่าง ๆ ในกรณีไฟฟ้าดับ ถ้าไม่มีเครื่องกำเนิดไฟฟ้าสำรองไก่อาจตายอย่างรวดเร็วถ้าไฟฟ้าดับเป็นเวลานาน

2) ต้องตรวจสอบเป็นประจำและทำความสะอาดพัดลม สายพาน ระบบอากาศเข้า การระบายอากาศเสียและทำงานเต็มประสิทธิภาพ

3) ในพื้นที่ที่น้ำมีแคลเซี่ยมมากจะต้องล้างและทำความสะอาดแผ่นรังผึ้งและแท็งค์เก็บน้ำต้องป้องกันการเกาะตัวของแคลเซี่ยมบนแผ่นรังผึ้ง

4) ใช้ยาฆ่าแมลง เพื่อควบคุมแมลงในแผ่นฉนวนใต้หลังคา

5) ต้องควบคุมพวกตะไคร่น้ำ เพื่อรักษาประสิทธิภาพการทำงานของแผ่นรังผึ้ง

6) มีต้นทุนการสร้างที่สูงกว่าระบบเปิด

7) มีต้นทุนค่าไฟฟ้าที่สูงกว่า

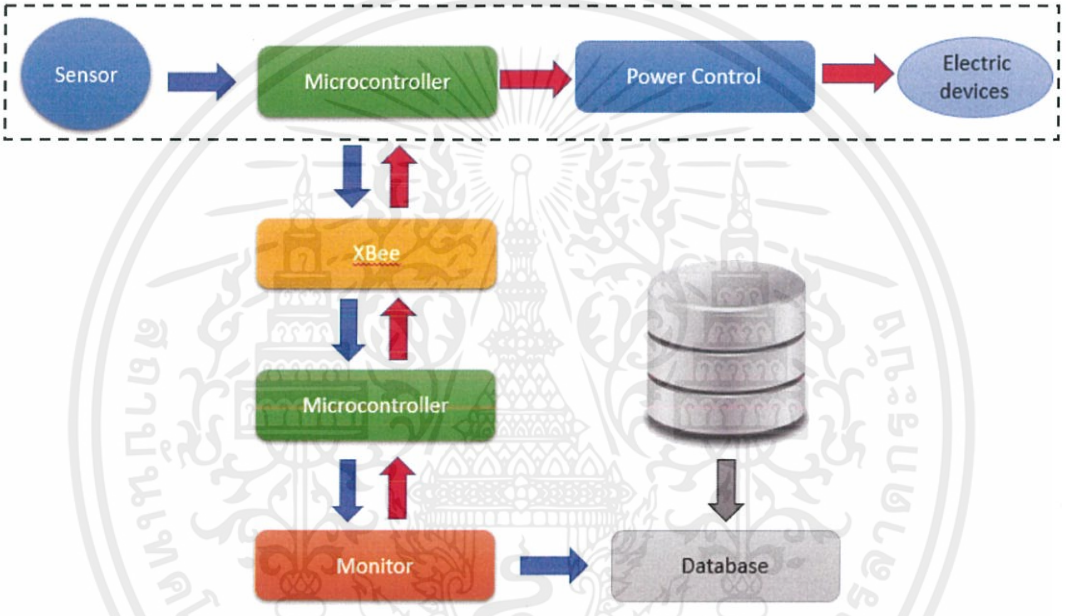
8) ไม่เหมาะที่จะนำไปใช้ในพื้นที่ที่มีความชื้นสัมพัทธ์ในอากาศค่อนข้างสูง เนื่องจากแผ่นรังผึ้งจะระบายน้ำได้น้อย และไม่สามารถลดอุณหภูมิภายในโรงเรือนได้

บทที่ 3

การออกแบบและการจัดทำปริญญานิพนธ์

3.1 การออกแบบ

การออกแบบและจัดทำปริญญานิพนธ์ จะเริ่มด้วยการวางแผนจัดทำโครงงานโดยได้ออกแบบการทำงานของระบบดังรูป 3.1



รูปที่ 3.1 บล็อกไดอะแกรมของระบบ

โดยการทำงานของระบบจะแบ่งการออกแบบเป็นสามส่วน คือ ส่วนไมโครคอนโทรลเลอร์ที่ติดตั้งอยู่ในแบบจำลองโรงเรือนเลี้ยงไก่แบบปิด (กรอบเส้นประ คือ แบบจำลองโรงเรือนเลี้ยงไก่แบบปิด) ส่วนหน้าจอแสดงผลและส่วนของฐานข้อมูล จากรูปที่ 3.1 สามารถอธิบายการทำงานของระบบเริ่มจากฝั่งของหน้าจอแสดงผลจะส่งค่าอุณหภูมิที่เหมาะสมสำหรับโรงเรือนไปยังไมโครคอนโทรลเลอร์ที่ติดตั้งอยู่ในแบบจำลอง ซึ่งไมโครคอนโทรลเลอร์ที่อยู่ในแบบจำลองนั้นจะมีเซ็นเซอร์สำหรับวัดอุณหภูมิและความชื้นสัมพัทธ์และมีวงจรรีเลย์ต่ออยู่เพื่อควบคุมอุปกรณ์ไฟฟ้า การส่งและรับข้อมูลระหว่างหน้าจอแสดงผลกับไมโครคอนโทรลเลอร์ที่อยู่ในแบบจำลองจะใช้โมดูล XBee โดยฝั่งไมโครคอนโทรลเลอร์ที่อยู่ในแบบจำลองจะรับค่าอุณหภูมิที่เหมาะสมสำหรับโรงเรือน

กับรับคำสั่งในการเปิดปิด พัดลมและหลอดไฟ ส่วนฝั่งหน้าจอแสดงผลจะรับค่าอุณหภูมิและความชื้นสัมพัทธ์ที่วัดได้จากเซ็นเซอร์มาแสดงในหน้าจอและนำค่าเหล่านี้ไปเก็บไว้ในฐานข้อมูล

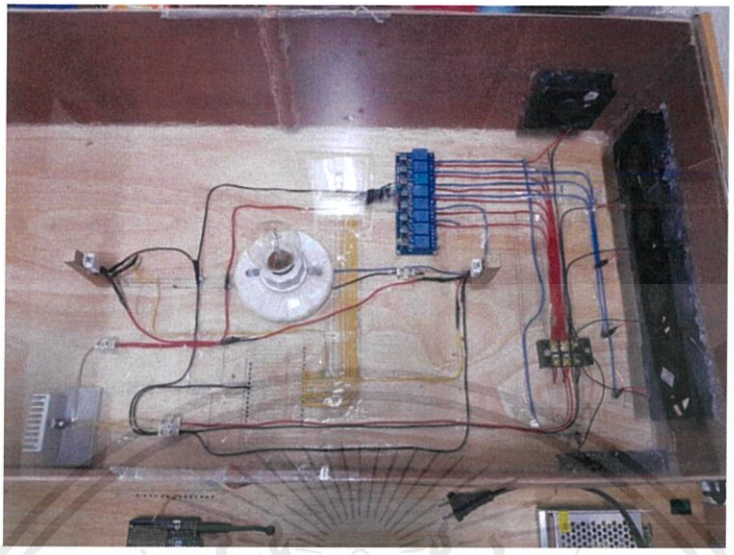
3.1.1 การออกแบบแบบจำลองโรงเรือนเลี้ยงไก่แบบปิด

แบบจำลองนี้ได้จำลองระบบหล่อเย็น (Evaporative Cooling System) มาจากโรงเรือนเลี้ยงไก่จริงโดยใช้ไม้ในการสร้าง โดยจะมีกระดาษรังผึ้งขนาด 35 ซม. X 18 ซม. ติดไว้ด้านหน้าของแบบจำลอง ส่วนด้านหลังและด้านข้างของแบบจำลองจะมีพัดลมจำนวน 6 ด้านหลังจะมีพัดลมอยู่ 4 ตัว ด้านข้างจะมีพัดลมอยู่ 2 ตัว ส่วนภายในแบบจำลองจะทำการติดตั้งเซ็นเซอร์วัดอุณหภูมิและความชื้นสัมพัทธ์ DHT22 จำนวน 2 ตัว ตัวแรกจะติดตั้งที่ด้านหน้าของแบบจำลอง ตัวที่สองจะติดตั้งที่ด้านหลังของแบบจำลอง ในส่วนตรงกลางของแบบจำลองจะมีหลอดไฟอยู่ 1 ดวง และในส่วนด้านนอกของแบบจำลองจะมีตัวปั้มน้ำอยู่ 1 ตัวเพื่อใช้ในระบบหล่อเย็น และจะมีตัว Arduino UNO R3 ที่เชื่อมต่อกับโมดูล XBee เพื่อใช้ควบคุมการทำงานของระบบหล่อเย็นและรับส่งค่าระหว่างแบบจำลองกับหน้าจอแสดงผล และจะมีตัว Switching Power Supply ทำหน้าที่แปลงความต่างศักย์ไฟฟ้าจาก 220 โวลต์(V) ไปเป็น 12 โวลต์(V) ซึ่งแบบจำลองนี้จะแสดงได้ดังรูปที่ 3.2, 3.3, 3.4, 3.5, และ 3.6 ตามลำดับ

รูปที่ 3.2 ส่วนด้านหน้าของแบบจำลอง

รูปที่ 3.3 ส่วนด้านหลังของแบบจำลอง

รูปที่ 3.4 ส่วนด้านข้างและด้านนอกของแบบจำลอง



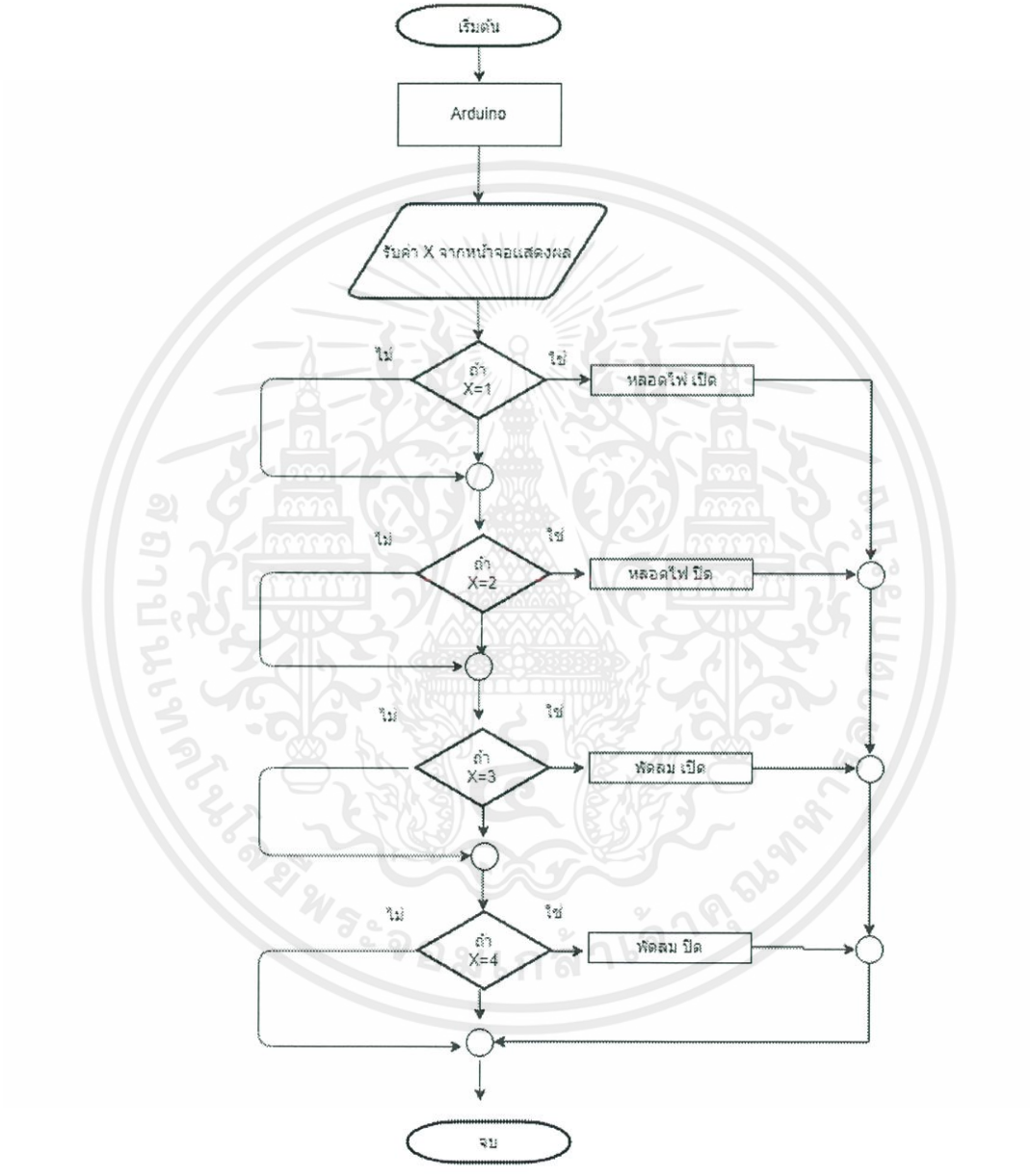
รูปที่ 3.5 ส่วนด้านในของแบบจำลอง

รูปที่ 3.6 แบบจำลองโรงเรือนเลี้ยงไก่แบบปิด

3.1.2 การออกแบบการทำงานของระบบหล่อเย็นในแบบจำลองโรงเรือนเลี้ยงไก่แบบปิด

การออกแบบการทำงานของระบบหล่อเย็นในแบบจำลองโรงเรือนเลี้ยงไก่แบบปิดนั้นจะใช้บอร์ด Arduino Uno R3 และ เซ็นเซอร์วัดอุณหภูมิและความชื้นสัมพัทธ์ DHT22 ในการควบคุมการเปิดปิด อุปกรณ์ต่างๆ โดยระบบหล่อเย็นจะแบ่งเป็น 2 ส่วน คือ ส่วนของการควบคุม

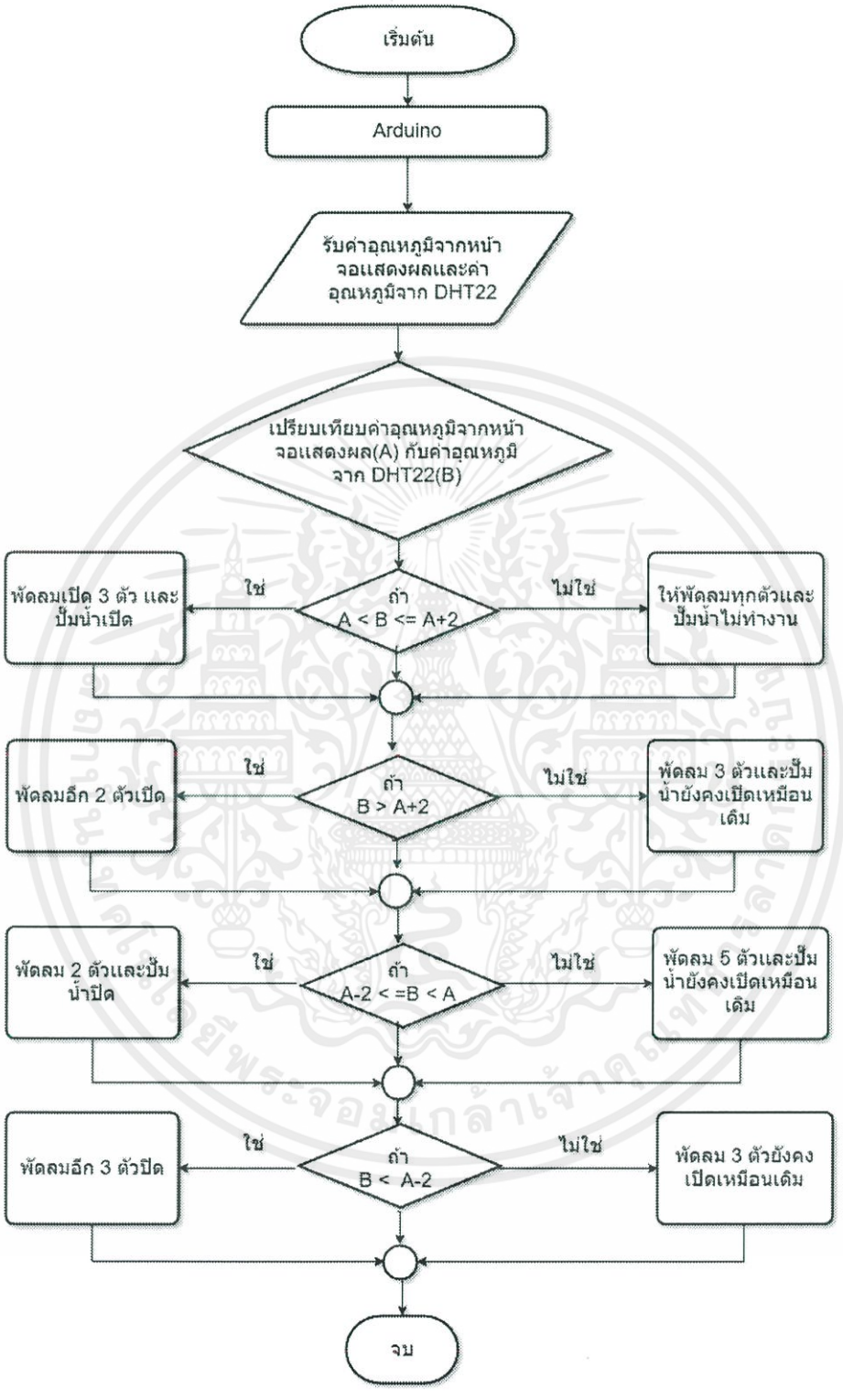
เปิด-ปิดพัดลมและหลอดไฟด้วยตัวเอง กับส่วนของการควบคุมเปิดปิดพัดลมและปั้มน้ำแบบอัตโนมัติ ซึ่งในส่วนของโดยจะแสดงการควบคุมเปิดปิดพัดลมและหลอดไฟด้วยตัวเอง จะแสดงแผนผังการทำงานในรูปที่ 3.7



รูปที่ 3.7 แผนผังการการควบคุมเปิดปิดพัดลมและหลอดไฟด้วยตัวเอง

จากรูปที่ 3.7 ส่วนการควบคุมเปิดปิดพัดลมและหลอดไฟด้วยตัวเองจะเริ่มจาก Arduino รับคำสั่งจากหน้าจอแสดงผลในการเปิด-ปิดพัดลม 1 ตัวและหลอดไฟได้ตามต้องการ ถ้ารับค่ามาเท่ากับ 1 2 3 4 ระบบก็จะส่งให้หลอดไฟเปิด หลอดไฟปิด พัดลมเปิด พัดลมปิด ตามลำดับ ซึ่งในระบบหล่อเย็นนี้จะสั่งการให้พัดลม 1 ตัวเปิดทำงานตลอดเวลาเพื่อรักษาอุณหภูมิในแบบจำลองโรงเรือนให้คงที่

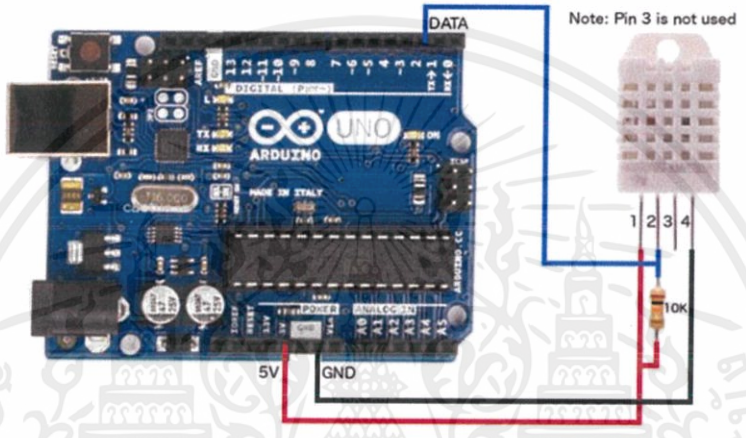
ในส่วนของการควบคุมเปิดปิดพัดลมและปั้มน้ำแบบอัตโนมัติจะเริ่มจาก Arduino รับค่าอุณหภูมิที่กำหนดเองจากหน้าจอแสดงผลและรับค่าอุณหภูมิที่วัดได้จากเซ็นเซอร์ DHT22 เมื่อรับค่าอุณหภูมิทั้งสองเสร็จก็จะนำมาเปรียบเทียบกัน โดยระบบหล่อเย็นในแบบจำลองนี้จะแบ่งลำดับการทำงานเป็น 4 ระดับ ระดับที่ 1 ถ้าค่าอุณหภูมิที่วัดได้จากเซ็นเซอร์มากกว่าค่าอุณหภูมิที่รับมาจากหน้าจอแสดงผล 1 องศาเซลเซียส ระบบจะสั่งการให้พัดลมที่อยู่ด้านหลังของแบบจำลองโรงเรือนทำงาน 3 ตัว และตัวปั้มน้ำก็จะทำงาน แต่ถ้าอุณหภูมิทั้งสองไม่มีค่าตรงกับเกณฑ์ดังกล่าว อุปกรณ์ไฟฟ้าในแบบจำลองจะไม่ทำงาน ส่วนในระดับที่ 2 ถ้าค่าอุณหภูมิที่วัดได้จากเซ็นเซอร์มากกว่าค่าอุณหภูมิที่รับมาจากหน้าจอแสดงผล 3 องศาเซลเซียส ระบบจะสั่งการให้พัดลมที่อยู่ทางด้านข้างของแบบจำลองโรงเรือนทำงานเพิ่มขึ้นอีก 2 ตัว แต่ถ้าอุณหภูมิทั้งสองไม่มีค่าตรงกับเกณฑ์ดังกล่าวพัดลมก็จะทำงานแค่ 3 ตัวเท่าเดิมและปั้มน้ำก็จะยังคงทำงานอยู่ เมื่อระบบสามารถควบคุมอุณหภูมิในแบบจำลองโรงเรือนให้มีค่าตามเกณฑ์ที่กำหนดไว้ได้แล้วหรือมีค่าอุณหภูมิในแบบจำลองโรงเรือนเท่ากับหรือน้อยกว่าค่าอุณหภูมิที่วัดได้จากเซ็นเซอร์ ระบบก็จะเข้าสู่ลำดับการทำงานในระดับที่ 3 คือถ้าค่าอุณหภูมิที่วัดได้จากเซ็นเซอร์มีค่าน้อยกว่าอุณหภูมิที่รับจากหน้าจอแสดงผล 1 องศาเซลเซียส ระบบจะสั่งการให้พัดลมที่อยู่ทางด้านข้างของแบบจำลองโรงเรือน 2 ตัวหยุดทำงาน และปั้มน้ำก็จะหยุดทำงานด้วย แต่ถ้าอุณหภูมิทั้งสองไม่มีค่าตรงกับเกณฑ์ดังกล่าวพัดลมทั้ง 5 ตัว และปั้มน้ำก็จะยังคงทำงานเหมือนเดิม ส่วนในระดับที่ 4 ถ้าค่าอุณหภูมิที่วัดได้จากเซ็นเซอร์มีค่าน้อยกว่าอุณหภูมิที่รับจากหน้าจอแสดงผล 3 องศาเซลเซียส ระบบจะสั่งการให้พัดลมที่อยู่ด้านหลังของแบบจำลองโรงเรือน 2 ตัวหยุดทำงาน แต่ถ้าแต่ถ้าอุณหภูมิทั้งสองไม่มีค่าตรงกับเกณฑ์ดังกล่าวพัดลมทั้ง 3 ตัวจะยังคงทำงานเหมือนเดิม ซึ่งจะแสดงดังรูปที่ 3.8



รูปที่ 3.8 แผนผังการควบคุมเปิดปิดพัดลมและปั๊มน้ำแบบอัตโนมัติ

3.1.2.1 ส่วนของเซ็นเซอร์วัดอุณหภูมิและความชื้นสัมพัทธ์

เมื่อมีอินพุทเข้ามาที่เซนเซอร์วัดอุณหภูมิและความชื้น แล้วผ่านการประมวลผลตามโปรแกรมที่เราได้เขียนไว้ ก็จะส่งข้อมูลไปทำการควบคุมการเปิด-ปิดพัดลมทั้ง 5 ตัว และปั้มน้ำ โดยรายละเอียดการเชื่อมต่อระหว่าง Arduino Uno R3 กับ เซนเซอร์วัดอุณหภูมิและความชื้น DHT22 จะแสดงดังรูปที่ 3.9



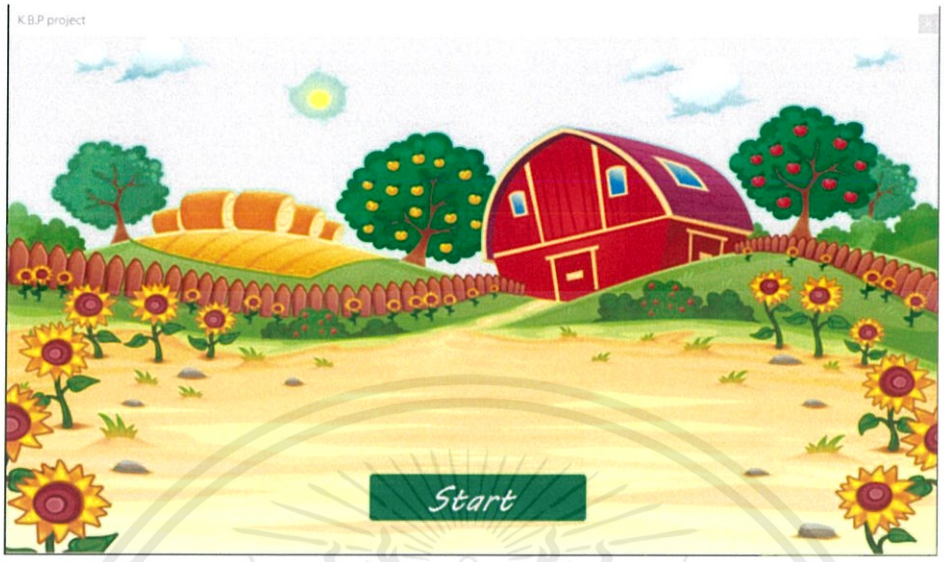
รูปที่ 3.9 รายละเอียดการเชื่อมต่อระหว่าง Arduino Uno R3 กับ DHT22

3.1.3 การออกแบบหน้าจอแสดงผล

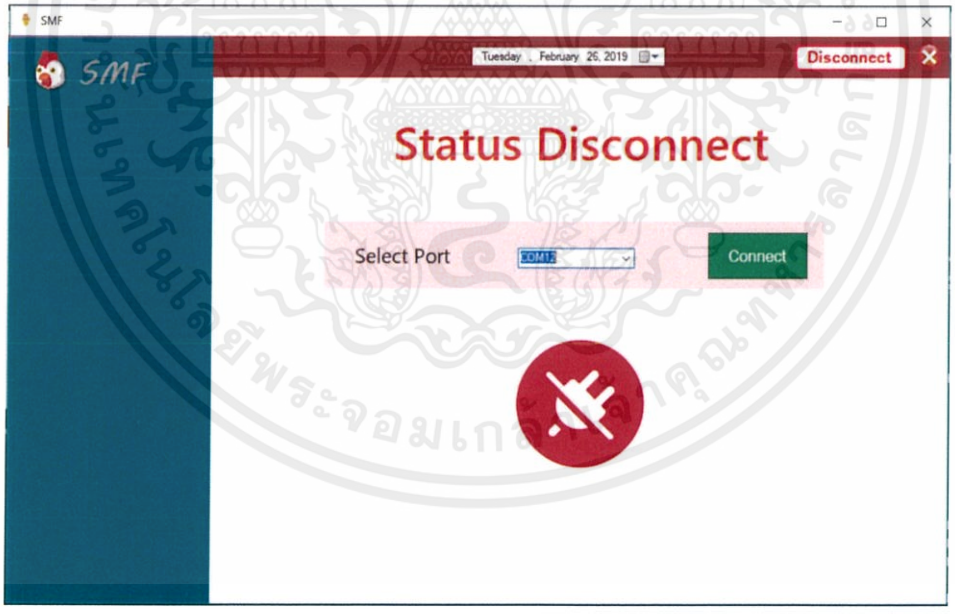
การออกแบบหน้าจอแสดงผลได้ใช้โปรแกรม Visual Studio 2017 ในการสร้าง ซึ่งจะแบ่งออกเป็น 2 ส่วน ได้แก่ ส่วนของหน้าแรก กับ ส่วนของหน้าต่างแสดงผลค่าข้อมูล

3.1.3.1 การออกแบบส่วนของหน้าแรก

โดยในหน้าแรกของหน้าจอแสดงผลนั้นจะให้ผู้ใช้งานทำการกดปุ่ม start ก่อนแล้วทำการเลือกพอร์ตของ Arduino ที่จะทำการเชื่อมต่อถ้าไม่ทำการเชื่อมต่อพอร์ตจะไม่สามารถใช้งานหน้าจอแสดงผลได้ ซึ่งจะแสดงได้ดังรูปที่ 3.10 และ 3.11



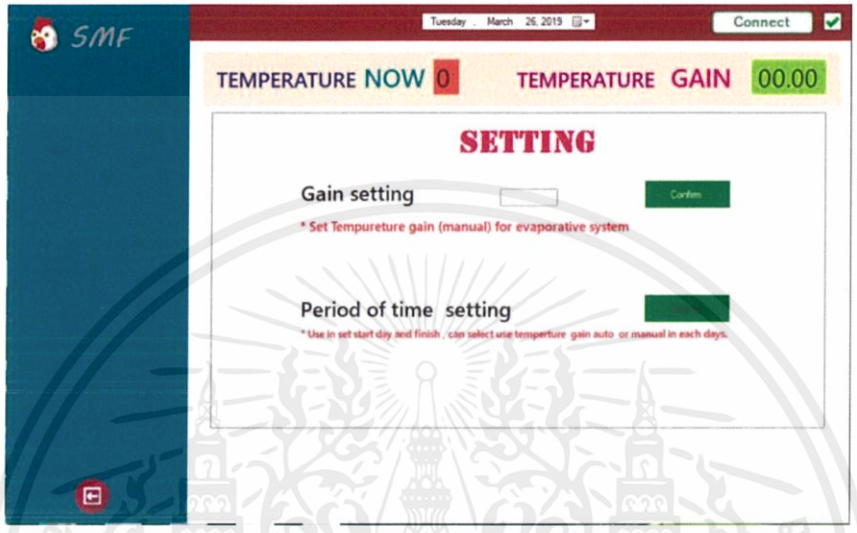
รูปที่ 3.10 หน้าแรกของหน้าจอแสดงผล



รูปที่ 3.11 หน้าจอส่วนของการเลือกพอร์ตเชื่อมต่อกับ Arduino

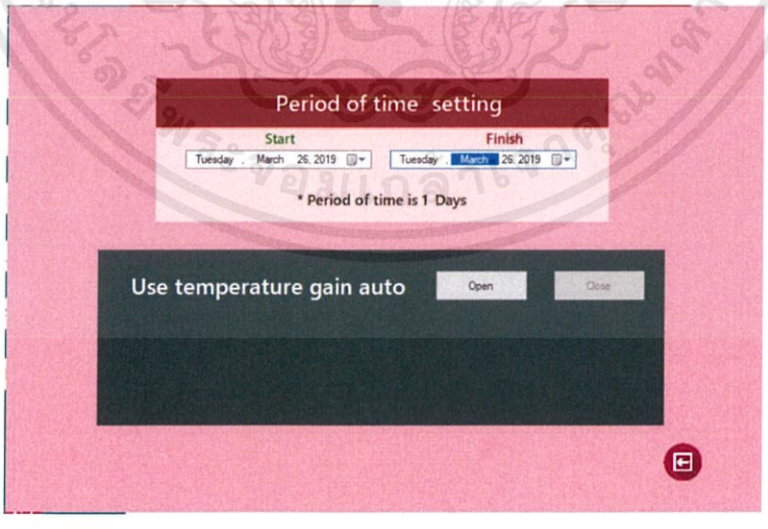
และเมื่อเราทำการเชื่อมต่อพอร์ตของ Arduino เรียบร้อยแล้วในหน้าจอแสดงผลก็จะเข้าสู่หน้าต่าง setting ในหน้าต่างนี้จะมีโหมดให้ผู้ใช้งานเลือกว่าจะใส่ค่าค่าอุณหภูมิที่

ต้องการให้เกิดขึ้นในแบบจำลองโรงเรือนเลี้ยงไก่ไปเลย หรือจะเลือกระยะเวลาในการเลี้ยงไก่ก่อน ถ้าผู้ใช้งานเลือกโหมดใส่ค่าอุณหภูมิที่ต้องการ เมื่อใส่ค่าเสร็จแล้วหน้าจอแสดงผลก็จะแสดงในส่วนของหน้าต่างแสดงผลค่าข้อมูล ซึ่งจะแสดงได้ดังรูปที่ 3.12

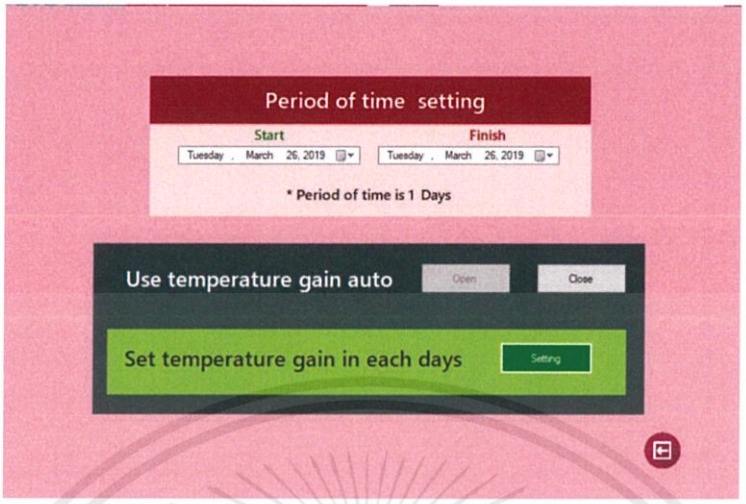


รูปที่ 3.12 หน้าจอแสดงผลเมื่อทำการใส่ค่าอุณหภูมิที่ต้องการให้เกิดขึ้นในแบบจำลอง

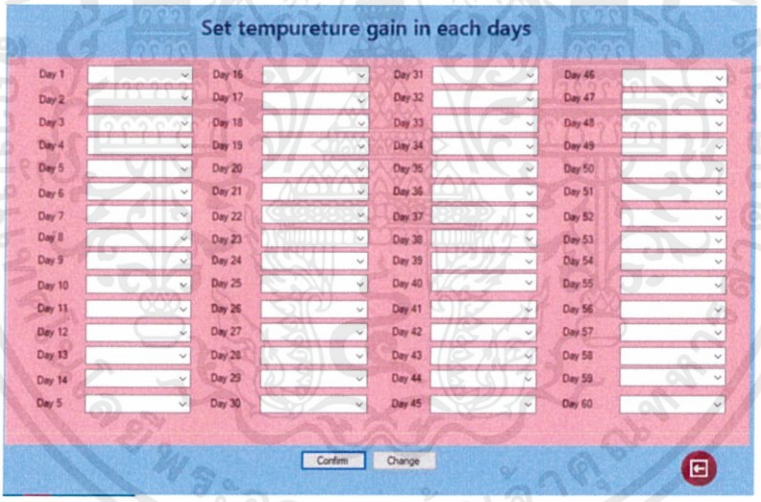
แต่ถ้าผู้ใช้งานเลือกโหมดระยะเวลาในการเลี้ยงไก่ หน้าจอแสดงผลก็จะให้เลือกวันที่เริ่มเลี้ยงไก่และวันที่สิ้นสุดวันเลี้ยงไก่ หลังจากนั้นก็จะให้ผู้ใช้งานเลือกตั้งค่าอุณหภูมิตามจำนวนวันที่เราต้องการเลี้ยงไก่ ซึ่งจะแสดงได้ดังรูปที่ 3.13 3.14 และ 3.15



รูปที่ 3.13 หน้าจอแสดงผลในโหมดเลือกระยะเวลาการเลี้ยงไก่



รูปที่ 3.14 หน้าจอแสดงผลในโหมดตั้งค่าอุณหภูมิ

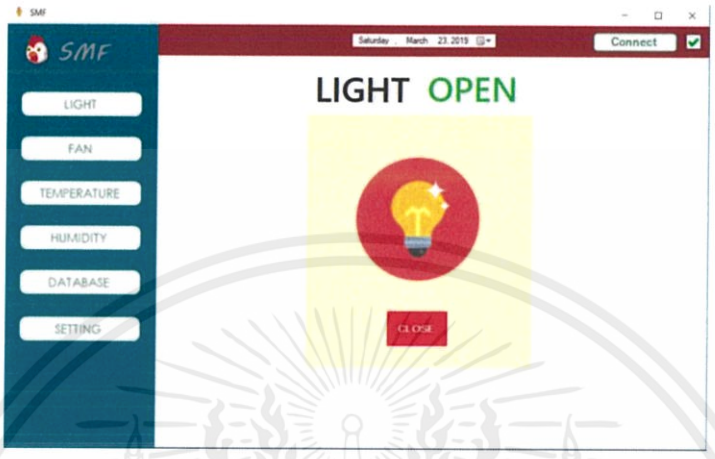


รูปที่ 3.15 หน้าต่างให้ผู้ใช้งานตั้งค่าอุณหภูมิในแต่ละวัน

3.1.3.2 การออกแบบในส่วนของหน้าต่างแสดงผลค่าข้อมูล

ส่วนของหน้าต่างแสดงผลค่าข้อมูลจะมีปุ่มให้ผู้ใช้งานเลือกกดดูได้ 6 ปุ่ม ได้แก่ Light Fan Temperature Humidity Database และ Setting โดยในส่วนของปุ่ม Light นั้นจะมีปุ่มให้กดเปิดและปิดการทำงานของหลอดไฟ โดยถ้าจะต้องการเปิดการทำงานของหลอดไฟ

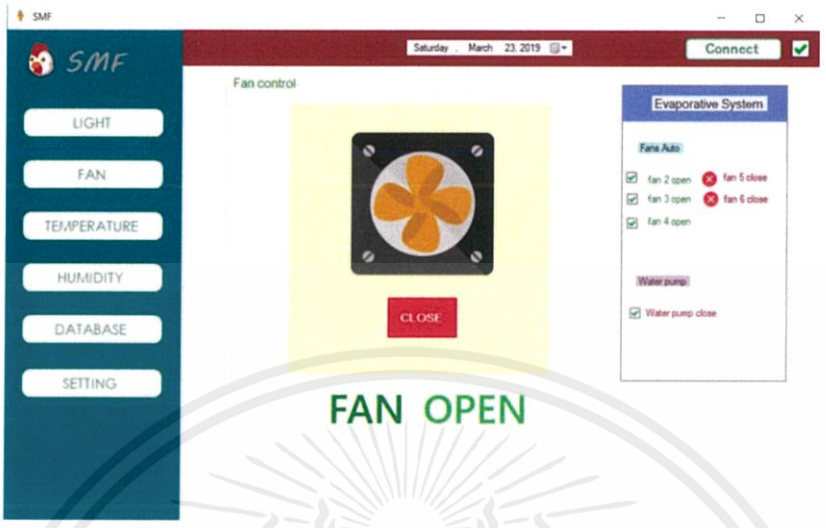
ให้ผู้ใช้งานกดปุ่ม OPEN และถ้าผู้ใช้งานต้องการจะปิดการทำงานของหลอดไฟให้กดปุ่ม CLOSE ซึ่งจะแสดงได้ดังรูปที่ 3.16 3.17 ตามลำดับ



รูปที่ 3.16 ปุ่ม Light แสดงสถานะหลอดไฟเปิด

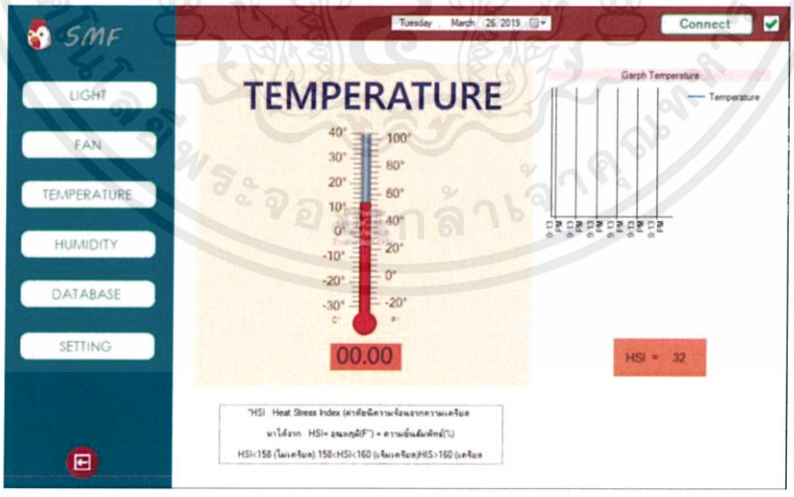
รูปที่ 3.17 ปุ่ม Light แสดงสถานะหลอดไฟปิด

ในส่วนของปุ่ม Fan นั้นจะมีปุ่มให้กดเปิดและปิดการทำงานของพัดลม โดยถ้าจะต้องการเปิดการทำงานของพัดลมให้ผู้ใช้งานกดปุ่ม OPEN แต่ถ้าผู้ใช้งานต้องการจะปิดการทำงานขอพัดลมให้กดปุ่ม CLOSE และปุ่มนี้จะแสดงสถานะการทำงานของพัดลมและปั้มน้ำที่ใช้ในระบบหล่อเย็นอีกด้วย ซึ่งจะแสดงได้ดังรูปที่ 3.18

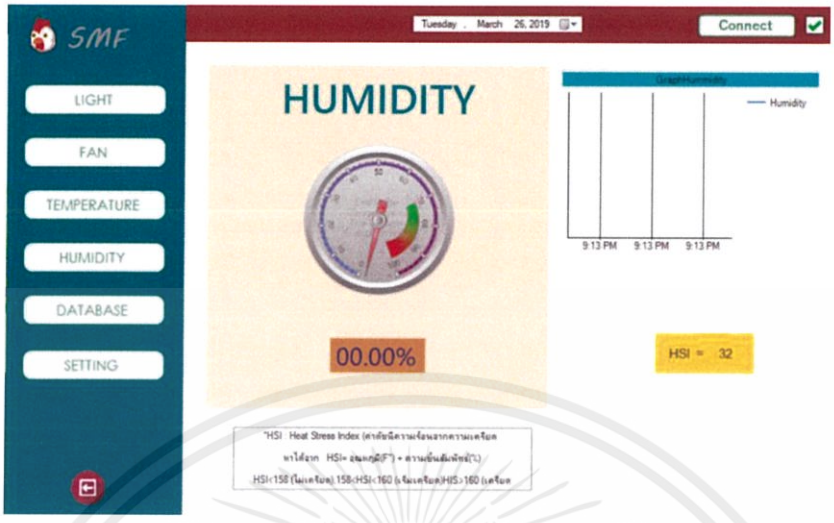


รูปที่ 3.18 ปุ่ม Fan แสดงสถานะพัดลมเปิดและแสดงสถานะพัดลมกับปั้มน้ำในระบบหล่อเย็น

ในส่วนของปุ่ม Temperature และ ปุ่ม Humidity จะแสดงค่าอุณหภูมิและความชื้นสัมพัทธ์ที่รับมาจากแบบจำลองโรงเรือนเลี้ยงไก่ และจะแสดงกราฟอุณหภูมิและความชื้นสัมพัทธ์เทียบกับเวลา และจะแสดงค่าดัชนีความเครียดเนื่องจากความความร้อน ซึ่งจะแสดงได้ดังรูปที่ 3.19 3.20 ตามลำดับ

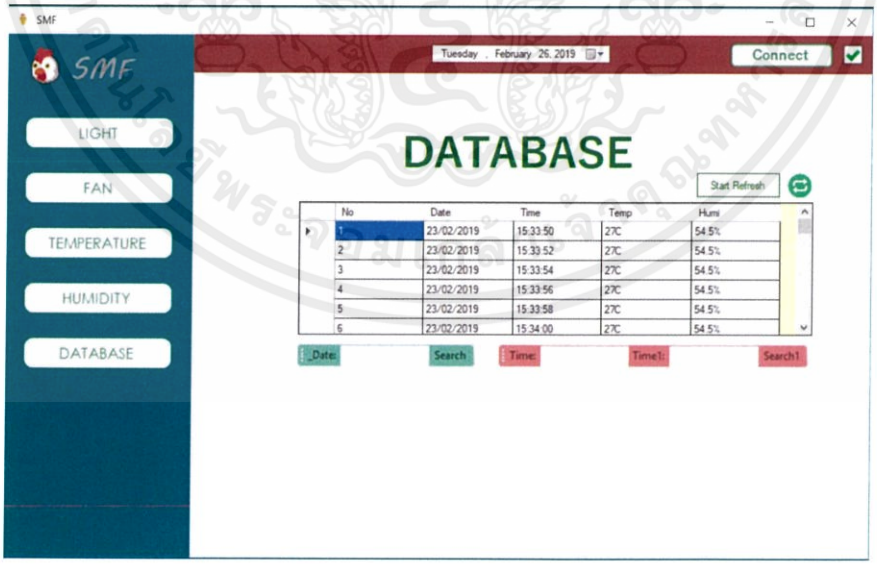


รูปที่ 3.19 หน้าต่างของปุ่ม Temperature



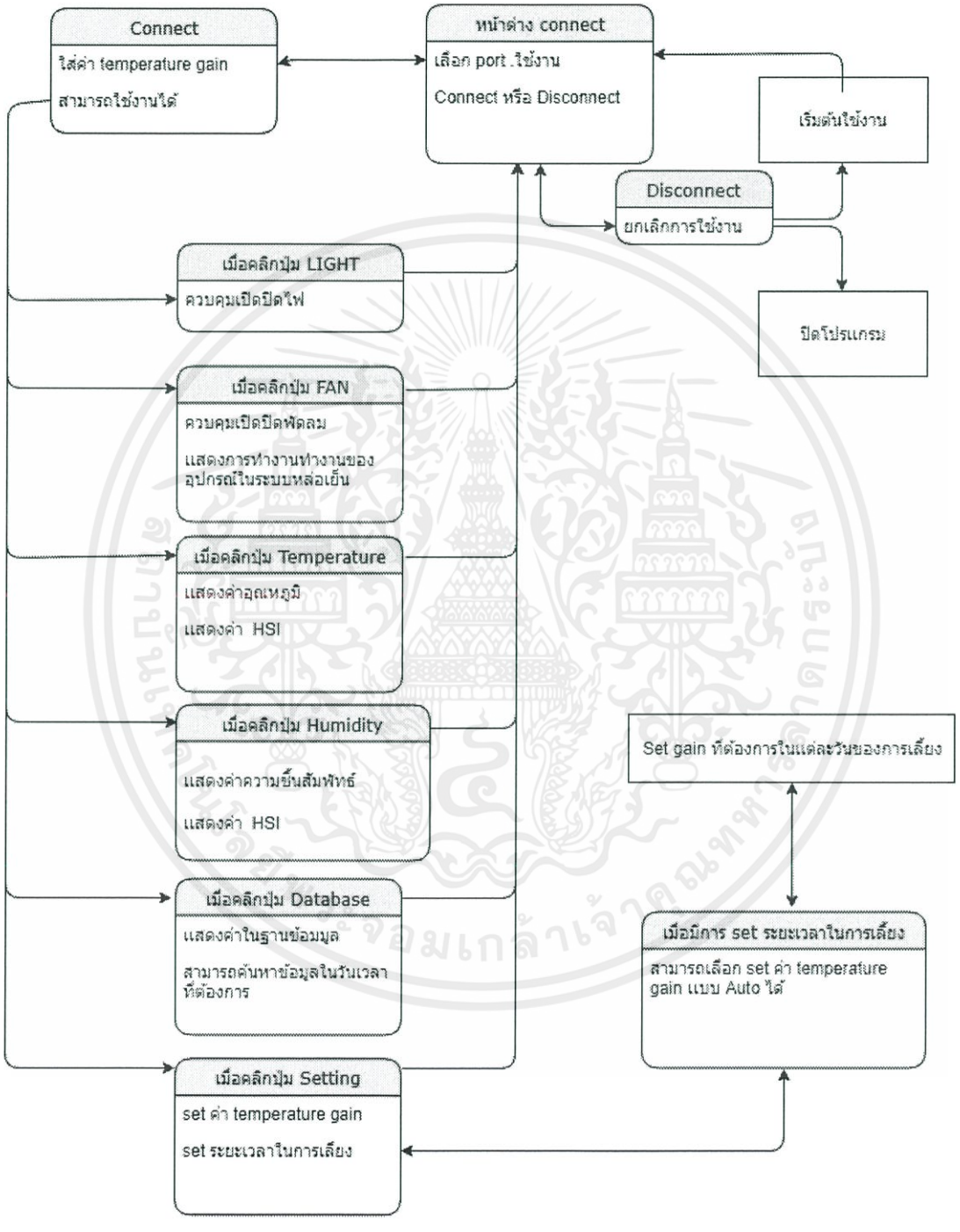
รูปที่ 3.20 หน้าต่างของปุ่ม Humidity

ในส่วนของปุ่ม Database จะแสดงตารางค่าอุณหภูมิและความชื้น ณ วันที่และเวลาต่างๆ มีปุ่มค้นหาอุณหภูมิและความชื้น ณ วันที่และช่วงเวลาต่างๆในตารางฐานข้อมูล และมีปุ่มรีเฟรชและหยุดรีเฟรชค่าในตารางฐานข้อมูล โดยตารางฐานข้อมูลที่ได้แสดงนั้นเป็นการดึงตารางฐานข้อมูลจากโปรแกรม Microsoft Access มาแสดง ซึ่งจะแสดงได้ดังรูปที่ 3.21



รูปที่ 3.21 หน้าต่างของปุ่ม Database

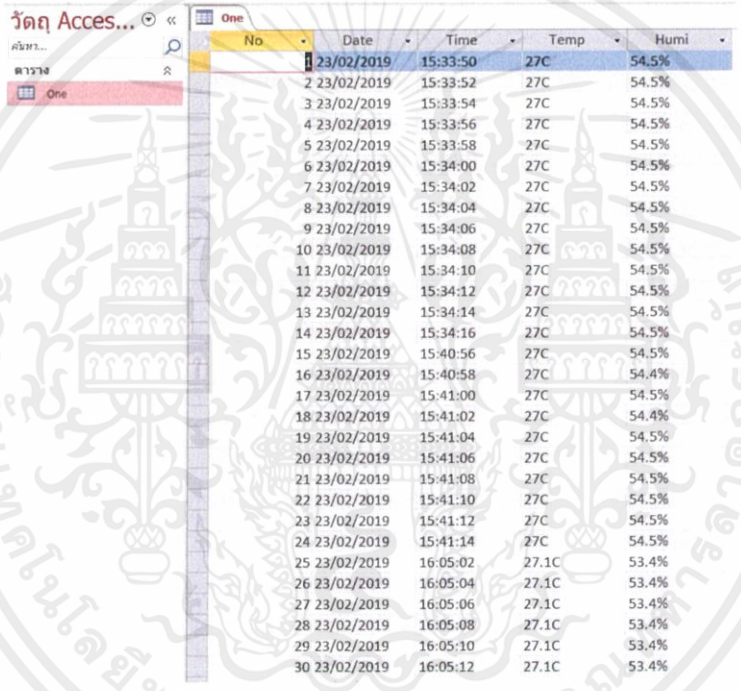
ซึ่งการทำงานทั้งหมดของหน้าจอแสดงผล สามารถสรุปเป็นแผนผังการทำงานได้ดังรูปที่ 3.22



รูปที่ 3.22 แผนผังการทำงานของหน้าจอแสดงผล

3.1.4 การออกแบบตารางฐานข้อมูล Microsoft Access

ได้ทำการออกแบบสร้างฐานข้อมูลเพื่อเก็บค่าอุณหภูมิและความชื้นที่ได้จากเซ็นเซอร์ DHT22 โดยดึงค่าอุณหภูมิ ความชื้น วันที่ และเวลา ที่แสดงในหน้าจอแสดงผลเอาไปเก็บไว้ในฐานข้อมูล ซึ่งฐานข้อมูลที่ได้ทำการสร้างนั้น คือ ฐานข้อมูล Microsoft Access ซึ่งฐานข้อมูลนี้เป็นฐานข้อมูลแบบออฟไลน์ และสามารถนำตารางฐานข้อมูลไปแสดงในโปรแกรม Visual Basic ได้ ฐานข้อมูลนี้เราได้ทำการสร้างฟิลด์เพื่อใช้ในการเก็บข้อมูลต่างๆจำนวน 5 ฟิลด์ ได้แก่ ลำดับ วันที่ เวลา อุณหภูมิ ความชื้น โดยตารางฐานข้อมูลจะแสดงได้ดังรูปที่ 3.23



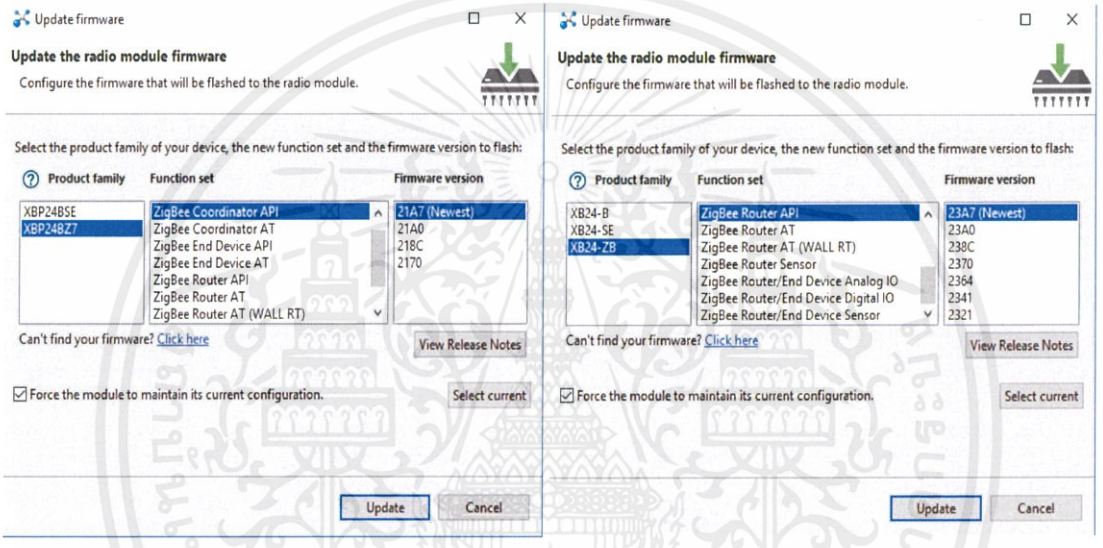
No	Date	Time	Temp	Humi
1	23/02/2019	15:33:50	27C	54.5%
2	23/02/2019	15:33:52	27C	54.5%
3	23/02/2019	15:33:54	27C	54.5%
4	23/02/2019	15:33:56	27C	54.5%
5	23/02/2019	15:33:58	27C	54.5%
6	23/02/2019	15:34:00	27C	54.5%
7	23/02/2019	15:34:02	27C	54.5%
8	23/02/2019	15:34:04	27C	54.5%
9	23/02/2019	15:34:06	27C	54.5%
10	23/02/2019	15:34:08	27C	54.5%
11	23/02/2019	15:34:10	27C	54.5%
12	23/02/2019	15:34:12	27C	54.5%
13	23/02/2019	15:34:14	27C	54.5%
14	23/02/2019	15:34:16	27C	54.5%
15	23/02/2019	15:40:56	27C	54.5%
16	23/02/2019	15:40:58	27C	54.4%
17	23/02/2019	15:41:00	27C	54.5%
18	23/02/2019	15:41:02	27C	54.4%
19	23/02/2019	15:41:04	27C	54.5%
20	23/02/2019	15:41:06	27C	54.5%
21	23/02/2019	15:41:08	27C	54.5%
22	23/02/2019	15:41:10	27C	54.5%
23	23/02/2019	15:41:12	27C	54.5%
24	23/02/2019	15:41:14	27C	54.5%
25	23/02/2019	16:05:02	27.1C	53.4%
26	23/02/2019	16:05:04	27.1C	53.4%
27	23/02/2019	16:05:06	27.1C	53.4%
28	23/02/2019	16:05:08	27.1C	53.4%
29	23/02/2019	16:05:10	27.1C	53.4%
30	23/02/2019	16:05:12	27.1C	53.4%

รูปที่ 3.23 ตารางฐานข้อมูล Microsoft Access ที่ใช้ในการเก็บค่าอุณหภูมิและความชื้นสัมพัทธ์ ณ วันที่และเวลาต่างๆ

3.1.5 การออกแบบการรับส่งข้อมูลผ่านเครือข่าย XBee

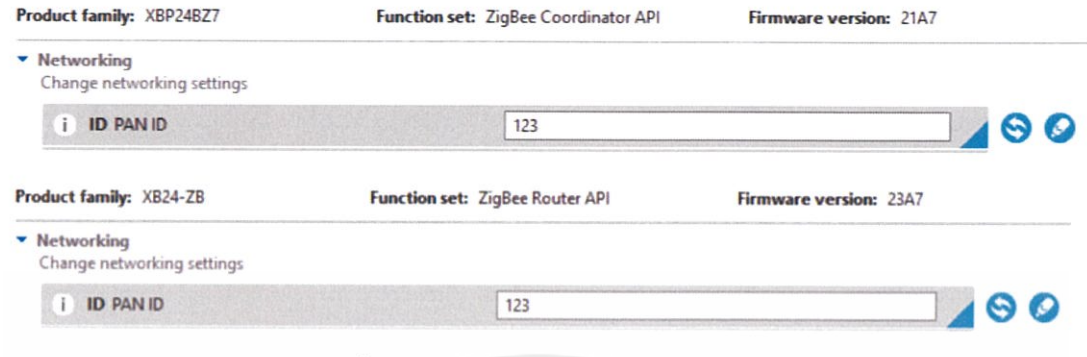
การรับส่งข้อมูลผ่านเครือข่าย XBee นั้นทางผู้จัดทำได้ทำการรับ-ส่งข้อมูลในโหมด API โดยจะแบ่งออกเป็น 2 ฝั่ง ได้แก่ ฝั่งที่เชื่อมต่อกับหน้าจอแสดงผล (ฝั่ง Router) กับ ฝั่งที่เชื่อมต่อกับอุปกรณ์ภายในแบบจำลองโรงเรือนเลี้ยงไก่ (ฝั่ง Coordinator) ซึ่งการออกแบบนั้นจะเริ่มต้นจากการนำ XBee ทั้งสองตัวมาตั้งค่าในโปรแกรม X-CTU ก่อนโดยมีขั้นตอนดังนี้

1. ทำการ Set Parameter ให้ติดต่อกันแบบ Point to Point เมื่อทำการต่อ XBee Series2 และ Mini XBee USB Dongle V2 เข้ากับ PC ทั้ง 2 ชุดเรียบร้อยแล้วให้เปิดโปรแกรม X-CTU ขึ้นมา 2 ชุดเช่นกัน แล้วทำการเลือก COM Port (UART) ในแต่ละชุดให้ถูกต้อง หลังจากให้เลือก Tab “Update” แล้วทำการ Set Parameter โดยให้ฝั่งหนึ่งเป็น Zigbee Router API และอีกฝั่งหนึ่งเป็น ZIgbee Coordinator API ซึ่งจะแสดงดังรูปที่ 3.24

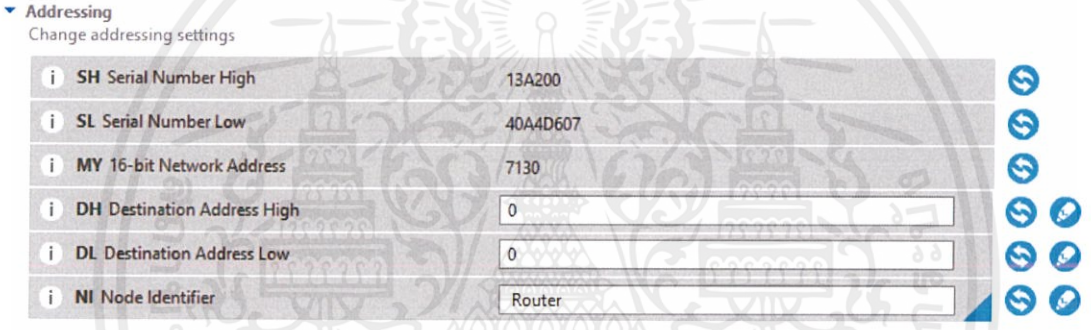


รูปที่ 3.24 การ Set Parameter ให้ติดต่อกันแบบ Point to Point

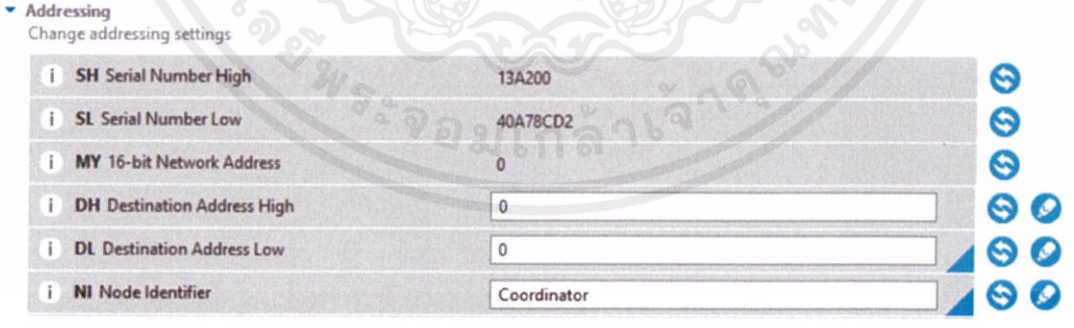
2. ให้ทำการ Set Parameter ดังนี้ Parameter PAN ID ฝั่ง Router และ Coordinator ต้องมีค่าตรงกัน ส่วน Parameter SH กับ SL นั้นแต่ละฝั่งต้องใส่ค่าของตัวที่ต้องการจะรับ-ส่งข้อมูลด้วย โดยค่า SH กับ SL ของแต่ละตัวนั้นสามารถดูได้ที่ด้านหลังของตัว XBee มันจะมีสติกเกอร์ ติดหมายเลข SH กับ SL ไว้แล้ว ส่วน Parameter DH กับ DL นั้นแต่ละฝั่งต้องใส่ค่าให้ตรงกัน ส่วน Parameter API Enable ให้ทั้งสองฝั่งใส่เลข 2 ซึ่งจะแสดงได้ดังรูปที่ 3.25 3.26 3.27 3.28 และ 3.29 ตามลำดับ



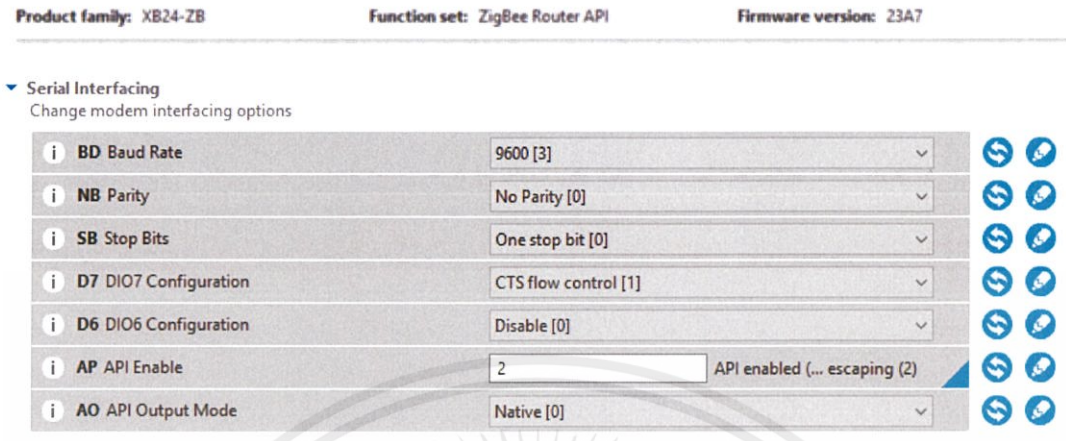
รูปที่ 3.25 การ Set Parameter PAN ID



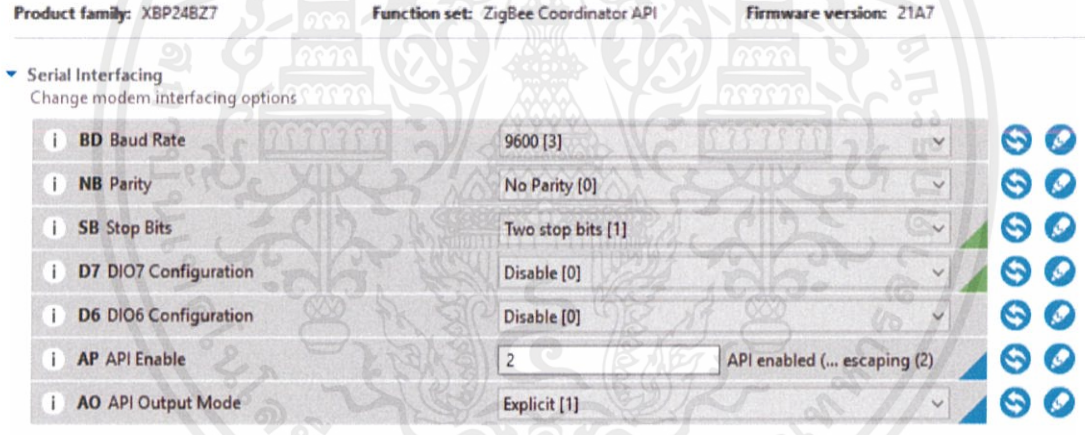
รูปที่ 3.26 การ Set Parameter SH SL DH DL ของฝั่ง Router



รูปที่ 3.27 การ Set Parameter SH SL DH DL ของฝั่ง Coordinator



รูปที่ 3.28 การ Set Parameter API Enable ของฝั่ง Router

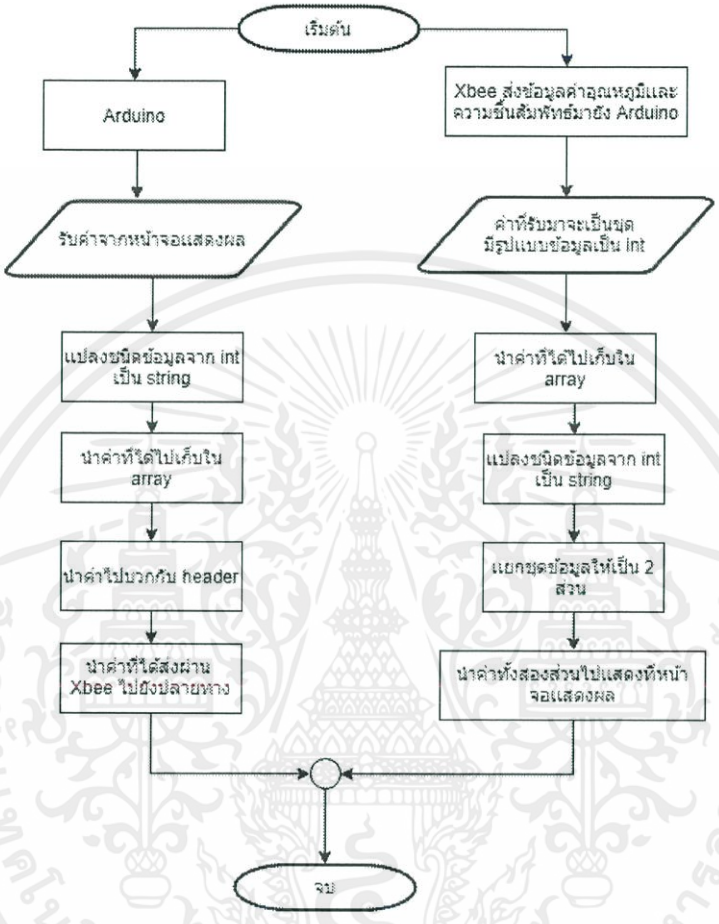


รูปที่ 3.29 การ Set Parameter API Enable ของฝั่ง Coordinator

3.1.5.1 การออกแบบการรับส่งข้อมูลของ XBee ฝั่งที่เชื่อมต่อกับหน้าจอแสดงผล

XBee ฝั่งที่เชื่อมต่อกับหน้าจอแสดงผลนั้นหรือ ฝั่ง Coordinator จะมีระบบการทำงานอยู่ 2 ส่วน คือ ส่วนของการรับข้อมูล และ ส่วนของการส่งข้อมูล โดยส่วนของการส่งข้อมูลนั้น XBee จะทำการส่งค่าอุณหภูมิที่เราต้องการให้กับแบบจำลองโรงเรือนเลี้ยงไก่ กับ ส่งคำสั่งในการควบคุมพัดลมและหลอดไฟในแบบจำลองโรงเรือนเลี้ยงไก่ และในส่วนของการรับข้อมูล

นั้น XBee จะรับค่าอุณหภูมิและความชื้นสัมพัทธ์ที่วัดได้จากเซ็นเซอร์ DHT22 ในแบบจำลองโรงเรือนเลี้ยงไก่ ซึ่งจะแสดงแผนผังการทำงานดังรูปที่ 3.30



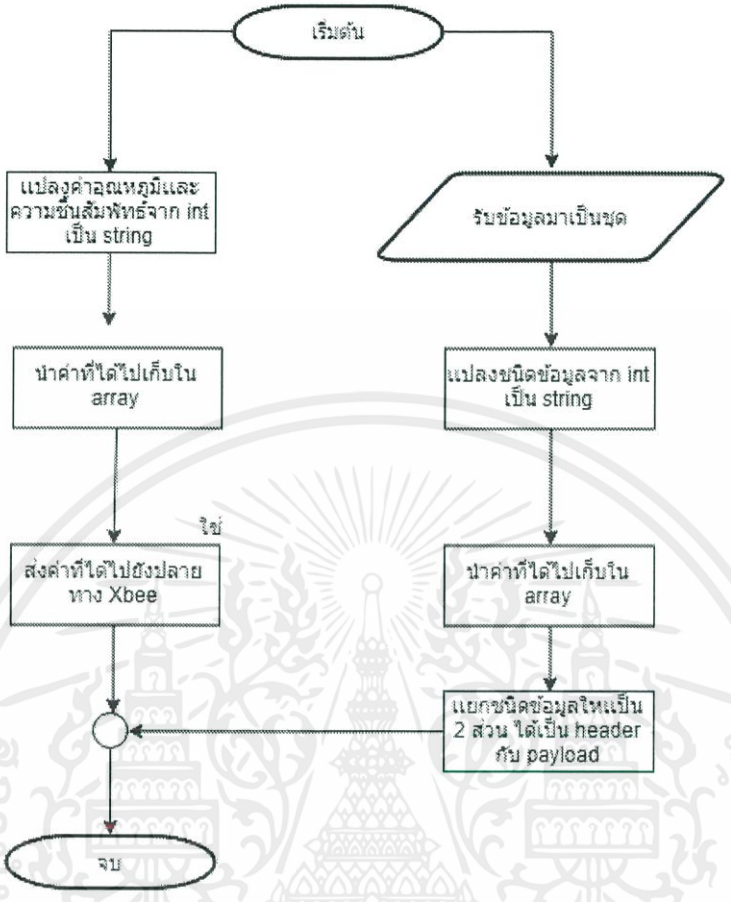
รูปที่ 3.30 แผนผังการทำงานของ XBee ฝั่งที่เชื่อมต่อกับหน้าจอแสดงผล

จากแผนผังการทำงานในรูปที่ 3.30 ในส่วนของการส่งข้อมูลนั้นชนิดของค่าอุณหภูมิที่ได้รับจากคีย์บอร์ดนั้นจะเป็นรูปแบบ Int จากนั้นก็ทำการแปลงรูปแบบข้อความจาก Int ให้กลายเป็น String แล้วนำไปเก็บใน Array ในรูปแบบ Char หลังจากนั้นก็ทำการส่งข้อมูลไปยังปลายทาง ซึ่งข้อมูลที่ทำการส่งไปนั้นจะเพิ่มตัว Header เข้าไปด้วยเพื่อแยกแยะชนิดของคำสั่งที่ส่งไปยังปลายทาง โดยถ้าตัว Header มีค่าเท่ากับ 9 ข้อมูลที่ส่งไปนั้นจะเป็นค่าอุณหภูมิที่เราต้องการให้กับแบบจำลองโรงเรือนเลี้ยงไก่ แต่ถ้าตัว Header มีค่าเท่ากับ 1 2 3 4 ข้อมูลที่ส่งไปนั้นจะเป็นคำสั่งให้ระบบเปิดหลอดไฟ ปิดหลอดไฟ เปิดพัดลม ปิดพัดลม ตามลำดับ และในส่วนของการรับข้อมูล XBee จะรับค่าอุณหภูมิและความชื้นสัมพัทธ์จากเซ็นเซ็อร์ DHT22 โดยข้อมูลที่

รับมาจะเป็นชุด คือ ค่าอุณหภูมิและความชื้นสัมพัทธ์ที่รับมาจะติดกันซึ่งชนิดของข้อมูลจะเป็นรูปแบบ Int จากนั้นจะทำการสร้าง Array เพื่อรองรับค่าที่รับเข้ามาซึ่งสามารถเก็บค่าที่เข้ามาได้ 12 ตัวอักษร หลังจากนั้นก็ทำการแปลงชนิดข้อมูลจาก Int ให้เป็น String แล้วทำการแยกชุดของข้อมูล จากนั้นก็ทำการแปลงข้อมูลจาก String เป็น Int อีกทีหนึ่งเพื่อแสดงข้อมูลอุณหภูมิและความชื้นสัมพัทธ์ที่รับเข้ามา

3.1.5.2 การออกแบบการรับส่งข้อมูลของ XBee ฝั่งที่เชื่อมต่อกับกับอุปกรณ์ภายในแบบจำลองโรงเรือนเลี้ยงไก่

XBee ของฝั่งที่เชื่อมต่อกับกับอุปกรณ์ภายในแบบจำลองโรงเรือนเลี้ยงไก่ หรือ ฝั่ง Router จะมีระบบการทำงานอยู่ 2 ส่วน คือ ส่วนของการรับข้อมูล และ ส่วนของการส่งข้อมูล โดยส่วนของการส่งข้อมูลนั้น XBee ทำการส่งค่าอุณหภูมิและความชื้นสัมพัทธ์ที่วัดได้จากเซ็นเซอร์ DHT22 และในส่วนของการรับข้อมูลนั้น XBee จะทำการรับข้อมูลค่าอุณหภูมิที่เราต้องการให้กับแบบจำลองโรงเรือนเลี้ยงไก่จากหน้าจอแสดงผล กับ รับคำสั่งควบคุมการเปิดปิดหลอดไฟและพัดลมจากหน้าจอแสดงผล ซึ่งจะแสดงแผนผังการทำงานดังรูปที่ 3.31



รูปที่ 3.31 แผนผังการทำงานของ XBee ฝั่ง Router

จากแผนผังการทำงานในรูปที่ 3.31 ในส่วนของการรับข้อมูลคำสั่งในการควบคุมพัดลมและหลอดไฟนั้น นั้นข้อมูลที่รับมาจะมีแค่ตัว Header ถ้าตัว Header มีค่าเท่ากับ 1 2 3 4 ระบบก็จะสั่งให้ทำการเปิดหลอดไฟ ปิดหลอดไฟ เปิดพัดลม ปิดพัดลม ตามลำดับ ในส่วนของการรับข้อมูลค่าอุณหภูมิที่เราต้องการให้กับแบบจำลองโรงเรือนเลี้ยงไก่ จะรับข้อมูลมาเป็นชุด คือ ข้อมูลนั้นจะมีตัว Header กับ ค่าอุณหภูมิที่เรากำหนดเอง จากนั้นจะทำการสร้าง Array เพื่อรองรับค่าที่รับเข้ามาซึ่งสามารถเก็บค่าที่เข้ามาได้ 5 ตัวอักษร หลังจากนั้นก็ทำการแปลงชนิดข้อมูลจาก Int ให้เป็น String แล้วทำการแยกชุดข้อมูล Header กับค่าอุณหภูมิออกจากกัน จากนั้นก็ทำการแปลงข้อมูลทั้งสองจาก String เป็น Int อีกทีหนึ่งเพื่อแสดงค่า Header กับ อุณหภูมิที่รับเข้ามา โดยค่า Header จะเป็นเลข 9 ตลอด และในส่วนของการส่งค่าอุณหภูมิและความชื้นสัมพัทธ์จากเซ็นเซอร์ DHT22 ค่าที่ต้องส่งไปตอนแรกจะเป็นชนิดข้อมูลแบบ Int จากนั้นก็ทำการแปลงรูปแบบ

ข้อความจาก Int ให้กลายเป็น String แล้วนำไปเก็บใน Array ในรูปแบบ Char หลังจากนั้นก็ทำการส่งข้อมูลไปยังปลายทาง

3.2 เครื่องมือที่ใช้ในการทดลอง

3.2.1 อุปกรณ์ด้านฮาร์ดแวร์

1) เครื่องคอมพิวเตอร์	1	เครื่อง
2) Arduino Uno R3	2	ตัว
3) Relay 8 Channel	1	ตัว
4) เซ็นเซอร์วัดอุณหภูมิและความชื้น DHT22	2	ตัว
5) ปั้มน้ำ 220 V	1	ตัว
6) พัดลม 12 V	6	ตัว
7) โมดูล XBee	2	ตัว
8) XBee Shield	2	ตัว
9) หลอดไฟ 220 V	1	ดวง
10) Power Supply Switching	1	เครื่อง

3.2.2 อุปกรณ์ด้านซอฟแวร์

1) Arduino IDE
2) X-CTU
3) Microsoft Visual Basic
4) Microsoft Access

3.3 การจัดเก็บผลการทดลอง

การจัดเก็บผลการทดลองแบ่งออกเป็น 5 ส่วน คือ

3.3.1 เก็บผลการทดลองการวัดค่าอุณหภูมิ

ใช้เซ็นเซอร์วัดอุณหภูมิและความชื้น DHT22 ที่ต่ออยู่กับไมโครคอนโทรลเลอร์วัดค่าอุณหภูมิภายในตู้เย็นและวัดค่าอุณหภูมิจริงจากเทอร์โมมิเตอร์และเปรียบเทียบผล

3.3.2 การทดสอบการวัดระยะทางในการรับ-ส่งข้อมูลผ่านเครือข่าย XBee

ทำการทดลองวัดระยะทางในการรับส่งข้อมูลของเครือข่าย XBee ในกรณี Line-Of- Sight (บริเวณที่โล่งไม่มีสิ่งกีดขวางกั้น) ว่าสามารถรับส่งข้อมูลได้อย่างครบถ้วนมากเท่าไหร่

3.3.3 ทดสอบระบบควบคุมอุณหภูมิและความชื้นสัมพัทธ์ในแบบจำลองโรงเรือนเลี้ยงไก่แบบปิด

ทำการทดลองโดยถ้าอุณหภูมิในแบบจำลองโรงเรือนเลี้ยงไก่มากกว่าอุณหภูมิที่เราได้กำหนดให้กับโรงเรือนพัดลมกับปั๊มน้ำจะทำงานกี่ตัว และทำทดลองระบบสามารถควบคุมอุณหภูมิให้มีค่าคงที่ตามเกณฑ์ที่กำหนดไว้ได้นานกี่นาที

3.3.4 ทดสอบการแสดงผลที่หน้าจอแสดงผล

ทำการทดสอบควบคุมเปิดปิดพัดลม หลอดไฟ ผ่านหน้าจอแสดงผลว่าสามารถใช้งานได้จริงหรือไม่ ทำการทดสอบการปรับค่าอุณหภูมิที่เราต้องการให้กับแบบจำลองโรงเรือนเลี้ยงไก่ว่าสามารถปรับได้จริงหรือไม่ และทำการทดสอบว่าหน้าจอแสดงผลนั้นสามารถแสดงค่าอุณหภูมิและความชื้นสัมพัทธ์ที่วัดได้จากเซ็นเซอร์ DHT22 ได้หรือไม่ และทำการทดสอบว่าหน้าจอแสดงผลนั้นสามารถแสดงตารางฐานข้อมูลที่ดึงมาจาก Microsoft Access สามารถเรียกดูค่าอุณหภูมิและความชื้นสัมพัทธ์ย้อนหลังจากตารางฐานข้อมูลได้หรือไม่

3.3.5 ทดสอบการส่งข้อมูลไปยังฐานข้อมูล

ทำการทดสอบว่าสามารส่งค่าอุณหภูมิและความชื้นสัมพัทธ์จากหน้าจอแสดงผลไปยังตารางฐานข้อมูล Microsoft Access ได้หรือไม่

บทที่ 4

ผลการทดลอง

4.1 ผลการทดลองการวัดค่าอุณหภูมิ

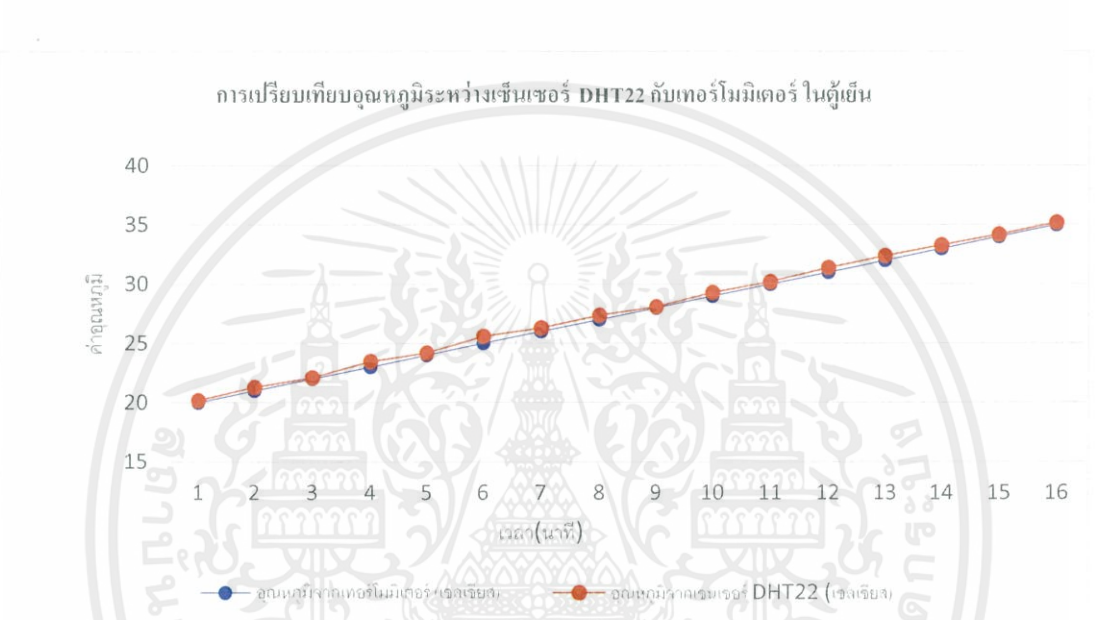
การทดลองได้ทำการเปิดตู้เย็นให้มีการทำงานทิ้งไว้เพื่อให้ภายในตู้เย็นมีค่าอุณหภูมิต่ำ หลังจากนั้นได้นำเซ็นเซอร์วัดอุณหภูมิและความชื้น DHT22 กับ เทอร์โมมิเตอร์ เอาเข้าไปไว้ในตู้เย็น เมื่อเริ่มการทดลองจะเปิดประตูตู้เย็นออกเพื่อให้อุณหภูมิภายในตู้เย็นสูงขึ้น แล้วเริ่มอ่านค่าอุณหภูมิจากเทอร์โมมิเตอร์เทียบกับค่าอุณหภูมิที่วัดได้จากเซ็นเซอร์ ซึ่งผลการทดลองเปรียบเทียบระหว่างเซ็นเซอร์ DHT22 กับเทอร์โมมิเตอร์ ณ อุณหภูมิต่าง ๆ จะแสดงดังตารางที่ 4.1 และรูปที่ 4.1

ตารางที่ 4.1 การเปรียบเทียบอุณหภูมิระหว่างเซ็นเซอร์ DHT22 กับเทอร์โมมิเตอร์ ในตู้เย็น

เวลา (นาที)	อุณหภูมิจากเทอร์โมมิเตอร์ (เซลเซียส)	อุณหภูมิจากเซนเซอร์ DHT22 (เซลเซียส)
1	20	20.2
2	21	21.3
3	22	22.1
4	23	23.5
5	24	24.2
6	25	25.6
7	26	26.3
8	27	27.4
9	28	28.1
10	29	29.3
11	30	30.2
12	31	31.4
13	32	32.4
14	33	33.3

15	34	34.2
16	35	35.2

ตารางที่ 4.1 การเปรียบเทียบอุณหภูมิระหว่างเซ็นเซอร์DHT22 กับ เทอร์โมมิเตอร์ ในตู้เย็น (ต่อ)



รูปที่ 4.1 กราฟแสดงการเปรียบเทียบค่าอุณหภูมิที่อ่านจากเทอร์โมมิเตอร์ กับค่าอุณหภูมิที่อ่านจากเซ็นเซอร์ DHT22

โดยสมการคำนวณค่าเปอร์เซ็นต์ความผิดพลาดเป็นดังนี้

$$\%Error = \left| \frac{X_{mea} - X_t}{X_t} \right| \times 100 \qquad (4.1)$$

โดย X_t คือ ค่าจริงหรือค่าจากเทอร์โมมิเตอร์ (True Value) และ X_{mea} คือ ค่าที่ได้จากการวัดหรือค่าจากเซ็นเซอร์ DHT22 (Measure Value)

และจากตารางที่ 4.1 ค่าเปอร์เซ็นต์ความผิดพลาดที่คำนวณออกมานั้นมีความคลาดเคลื่อนเท่ากับ 0.012% ซึ่งไม่เกิน 1% ซึ่งสามารถสรุปได้ว่าเซ็นเซอร์ DHT22 นั้นสามารถใช้งานได้อย่างมีประสิทธิภาพ

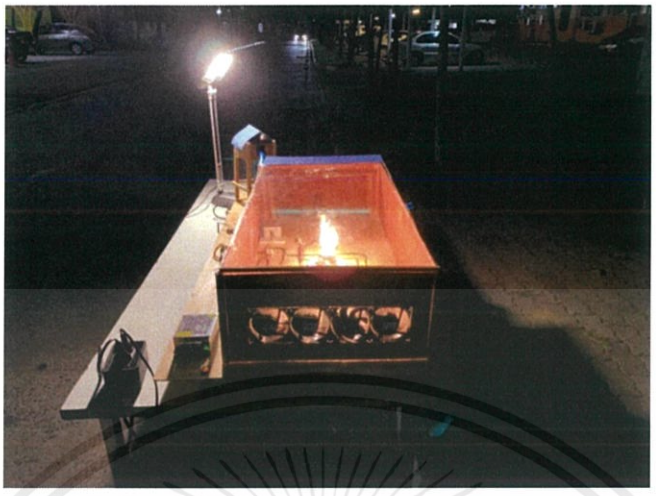
4.2 ผลการทดลองการวัดระยะทางในการรับส่งข้อมูลของเครือข่าย XBee

การทดลองนี้จะเป็นการทดลองการรับส่งข้อมูลของเครือข่าย XBee ในพื้นที่โล่งหรือบริเวณพื้นที่ที่ไม่มีสิ่งกีดขวาง (Line-Of-Sight) ว่ามีความเสถียรมากน้อยแค่ไหน

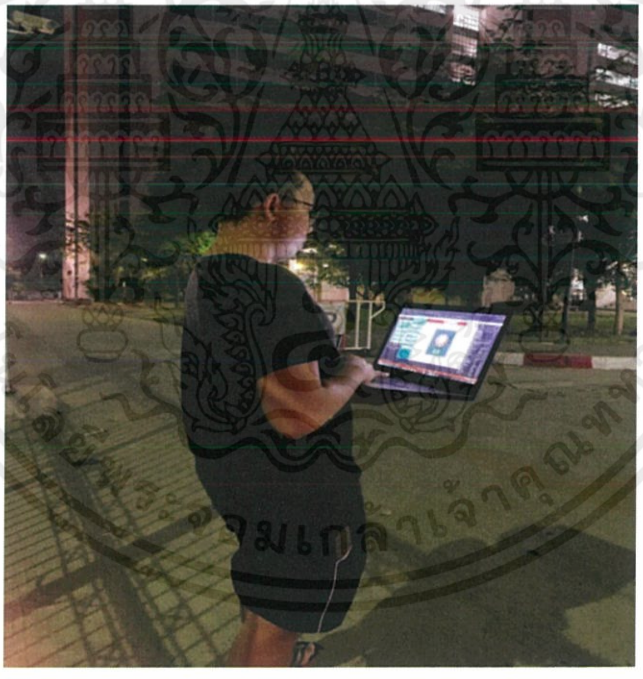
4.2.1 การทดลองในกรณีที่เป็นแบบ Line-Of-Sight

เป็นการทดลองการวัดระยะทางในการรับส่งข้อมูลของเครือข่าย XBee โดยจะทำการตั้งแบบจำลองโรงเรือนเลี้ยงไก่ไว้ตรงริมขอบถนนหน้าตึกภาควิชาโทรคมนาคม ซึ่งจะมี XBee ที่ตั้งค่าเป็น Router ที่ติดตั้งอยู่กับแบบจำลองด้วย และ XBee อีกหนึ่งตัวที่ตั้งค่าเป็น Coordinator จะต่อกับโน้ตบุ๊ค โดยจะทำการวัดจากถนนหน้าตึกภาควิชาโทรคมนาคม (จุดสีฟ้า) ถึงสุดถนนของอาคารเรียนรวม 12 ชั้น (จุดสีแดง) เป็นระยะทางรวม 500 เมตรแสดงในรูปที่ 4.2 ซึ่งจะทำการส่งข้อมูลทุก ๆ ระยะ 50 เมตร จาก XBee ที่เป็น Coordinator ไปยังตัวที่เป็น Router เพื่อไปควบคุมไฟและพัดลมที่ติดตั้งอยู่แบบจำลองให้สามารถทำงานได้ โดยจะแสดงได้ดังรูปที่ 4.2 4.3 และ 4.4 ตามลำดับ

รูปที่ 4.2 ระยะทางจากถนนหน้าตึกภาควิชาโทรคมนาคมถึงสุดถนนอาคารเรียนรวม 12 ชั้น



รูปที่ 4.3 แบบจำลองโรงเรือนเลี้ยงไก่ที่ตั้งอยู่ริมถนนหน้าตึกภาควิชาโทรคมนาคม

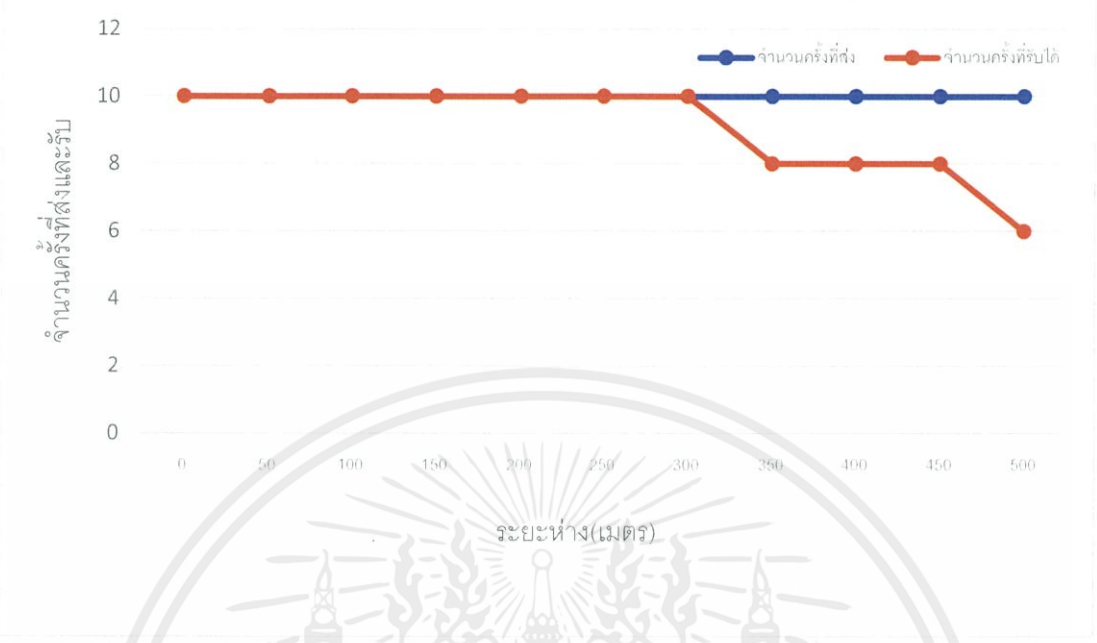


รูปที่ 4.4 ทดลองส่งข้อมูลที่สุดถนนหน้าอาคารเรียนรวม 12 ชั้น

ตารางที่ 4.2 การทดลองในกรณีที่เป็นแบบ Line-Of-Sight

ระยะทาง (เมตร) จากตัวส่งไปตัวรับ	จำนวนครั้งที่ส่ง	จำนวนครั้งที่รับได้	คิดเป็นเปอร์เซ็นต์
0	10	10	100
50	10	10	100
100	10	10	100
150	10	10	100
200	10	10	100
250	10	10	100
300	10	10	100
350	10	8	80
400	10	8	80
450	10	8	80
500	10	6	60

จากตารางที่ 4.2 การทดลองในกรณีที่เป็นแบบ Line-Of-Sight ผลจากการทดลองตามจุดต่าง ๆ จะเห็นได้ว่าจุดทดลองจะเป็นลักษณะเส้นตรงและระยะห่างมากขึ้นเรื่อย ๆ ผลที่ได้ คือ ในระยะใกล้ ๆ กับอุปกรณ์ทดลอง ข้อมูลที่ได้รับมีค่าเท่ากับจำนวนข้อมูลที่ส่ง และจุดอื่น ๆ ที่ห่างออกไปก็จะเห็นได้ว่า ได้รับสัญญาณน้อยลงเรื่อย ๆ ตามลำดับ ซึ่งจากการทดลองได้ระยะทางถึงประมาณ 500 เมตร ยังสามารถรับข้อมูลได้และคาดว่ายังรับข้อมูลได้อีกด้วย นำข้อมูลจากตารางที่ 4.2 นำไปพล็อตกราฟ โดยจะแสดงดังรูปที่ 4.5



รูปที่ 4.5 กราฟแสดงความสัมพันธ์ระหว่างจำนวนครั้งที่ส่งและรับกับระยะห่าง

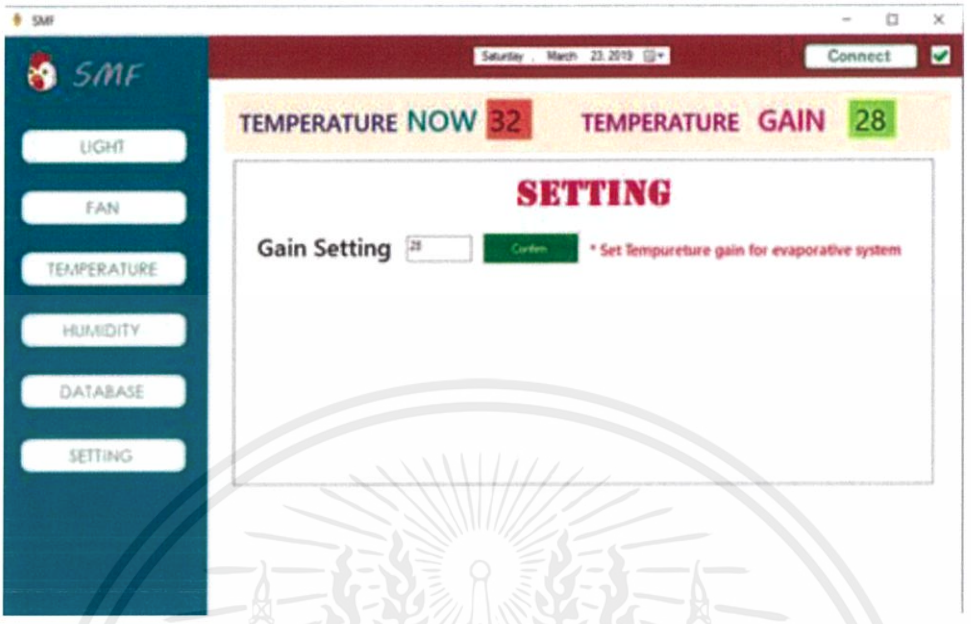
4.3 ผลการทดลองระบบควบคุมอุณหภูมิและความชื้นสัมพัทธ์ในแบบจำลองโรงเรือนเลี้ยงไก่

ได้ทำการออกแบบแบบจำลองโรงเรือนเลี้ยงไก่แบบปิด ซึ่งภายในโรงเรือนจะมีอุปกรณ์ต่าง ๆ ติดตั้งอยู่ จะเป็นการทดลองการควบคุมอุณหภูมิและความชื้นสัมพัทธ์ภายในโรงเรือนแบบอัตโนมัติ ซึ่งหลักการทำงานของระบบจะขึ้นกับค่าของอุณหภูมิ โดยเราจะมีหน้าจอแสดงผลที่สามารถตั้งค่าของอุณหภูมิตามที่เราต้องการจะให้ระบบทำงานได้ เมื่อค่าอุณหภูมิที่รับได้จากเซ็นเซอร์ DHT22 มีค่าสูงกว่าค่าอุณภูมิที่เรากำหนดระบบก็จะเริ่มทำงาน เราสามารถแบ่งการทดลองออกเป็น 2 ช่วง คือ ช่วงกลางวันและช่วงกลางคืน เนื่องจากในแต่ละช่วงเวลามีค่าอุณหภูมิที่แตกต่างกัน

4.3.1 การทดลองในช่วงเวลากลางวัน

ขั้นตอนแรกจะนำแบบจำลองโรงเรือนออกมาตั้งไว้ในช่วงเวลากลางวันเพื่อที่จะทำการทดลอง ซึ่งจะแสดงดังรูปที่ 4.6 โดยจะมีหน้าจอแสดงผลเป็นตัวตั้งค่ากำหนดเกณฑ์อุณหภูมิที่เราต้องการ เมื่อเราเปิดโปรแกรมหน้าจอแสดงผลขึ้นมา XBee ที่ติดตั้งอยู่กับแบบจำลองโรงเรือนเลี้ยงไก่ จะทำการส่งค่าอุณหภูมิและความชื้นสัมพัทธ์ที่อยู่ภายในแบบจำโรงเรือนเลี้ยงไก่ไปยัง XBee ที่ติดตั้งกับหน้าจอแสดงผล แล้วเราก็จะทำการกำหนดค่าเกณฑ์อุณหภูมิที่เราต้องการเพื่อให้ระบบทำงานได้อย่างถูกต้อง ซึ่งค่าอุณหภูมิบนหน้าจอแสดงผลฝั่งซ้ายจะเป็นค่าของอุณภูมิภายในแบบจำลองโรงเรือนเลี้ยงไก่และค่าอุณหภูมิฝั่งขวาจะเป็นค่าเกณฑ์อุณหภูมิที่เรากำหนดเอง ซึ่งในการทดลองได้กำหนดค่าเกณฑ์อุณหภูมิที่ 28 องศาเซลเซียส โดยจะแสดงดังรูปที่ 4.7

รูปที่ 4.6 แบบจำลองโรงเรือนเลี้ยงไก่ที่ตั้งไว้ในช่วงเวลากลางวัน



รูปที่ 4.7 หน้าจอแสดงผลที่แสดงค่าอุณหภูมิภายในแบบจำลองโรงเรือนเลี้ยงไก่และค่าเกณฑ์อุณหภูมิที่เรากำหนดเองในเวลากลางวัน

การเริ่มทำงานของระบบขึ้นอยู่กับค่าเกณฑ์อุณหภูมิที่เรากำหนด คือ ถ้าเรากำหนดเกณฑ์ไว้ที่ค่าใดค่าหนึ่งแล้ว เมื่ออุณหภูมิภายในแบบจำลองโรงเรือนเลี้ยงไก่มีค่าสูงกว่าเกณฑ์อุณหภูมิที่เรากำหนดอยู่ 1 องศาเซลเซียส ระบบจะเริ่มทำงานโดยมีพัดลมตัวที่ 2 3 และ 4 กับปั๊มน้ำที่ใช้ในการทำระบบหล่อเย็นจะเริ่มทำงาน และเมื่ออุณหภูมิภายในแบบจำลองโรงเรือนเลี้ยงไก่มีค่าสูงกว่าเกณฑ์อุณหภูมิที่เรากำหนดอยู่ 3 องศาเซลเซียส จะทำให้พัดลมตัวที่ 5 และ 6 เริ่มทำงาน ส่วนการหยุดทำงานก็ต่อเมื่ออุณหภูมิภายในแบบจำลองโรงเรือนเลี้ยงไก่มีค่าต่ำกว่าเกณฑ์อุณหภูมิที่เรากำหนดอยู่ 1 องศาองศาเซลเซียส จะทำให้พัดลมตัวที่ 5 และ 6 กับปั๊มน้ำหยุดทำงาน และเมื่ออุณหภูมิภายในแบบจำลองโรงเรือนเลี้ยงไก่มีค่าต่ำกว่าเกณฑ์อุณหภูมิที่เรากำหนดอยู่ 3 องศาเซลเซียส จะทำให้พัดลมตัวที่ 2 3 และ 4 จะหยุดทำงาน ซึ่งค่าอุณหภูมิและความชื้นสัมพัทธ์ภายในแบบจำลองโรงเรือนเลี้ยงไก่ที่เปลี่ยงแปลงในช่วงของการทดลองนั้นจะแสดงดังรูปที่ 4.8 4.9 4.10 4.11 4.12 และ 4.13

One

No	Date	Time	Temp	Humi
15406	23/03/2019	16:50:01	34C	63%
15407	23/03/2019	16:50:03	34C	63%
15408	23/03/2019	16:50:05	34C	63%
15409	23/03/2019	16:50:07	34C	65%
15410	23/03/2019	16:50:09	34C	65%

รูปที่ 4.8 ค่าอุณหภูมิและความชื้นสัมพัทธ์ภายในแบบจำลองโรงเรือนเลี้ยงไก่ ณ เวลา 16.50 น.

One

No	Date	Time	Temp	Humi
15436	23/03/2019	16:51:01	33C	63%
15437	23/03/2019	16:51:03	33C	63%
15438	23/03/2019	16:51:05	33C	63%
15439	23/03/2019	16:51:07	33C	64%
15440	23/03/2019	16:51:09	33C	64%

รูปที่ 4.9 ค่าอุณหภูมิและความชื้นสัมพัทธ์ภายในแบบจำลองโรงเรือนเลี้ยงไก่ ณ เวลา 16.51 น.

One

No	Date	Time	Temp	Humi
15493	23/03/2019	16:52:55	32C	73%
15494	23/03/2019	16:52:57	32C	73%
15495	23/03/2019	16:52:59	32C	73%
15496	23/03/2019	16:53:01	32C	73%
15497	23/03/2019	16:53:03	32C	73%
15498	23/03/2019	16:53:05	32C	73%

รูปที่ 4.10 ค่าอุณหภูมิและความชื้นสัมพัทธ์ภายในแบบจำลองโรงเรือนเลี้ยงไก่ ณ เวลา 16.53 น.

One

No	Date	Time	Temp	Humi
15525	23/03/2019	16:53:59	31C	78%
15526	23/03/2019	16:54:01	31C	78%
15527	23/03/2019	16:54:03	31C	78%
15528	23/03/2019	16:54:05	31C	78%
15529	23/03/2019	16:54:07	31C	78%

รูปที่ 4.11 ค่าอุณหภูมิและความชื้นสัมพัทธ์ภายในแบบจำลองโรงเรือนเลี้ยงไก่ ณ เวลา 16.54 น.

One

No	Date	Time	Temp	Humi
15587	23/03/2019	16:56:03	30C	81%
15588	23/03/2019	16:56:05	30C	81%
15589	23/03/2019	16:56:07	30C	81%
15590	23/03/2019	16:56:09	30C	81%
15591	23/03/2019	16:56:11	30C	81%

รูปที่ 4.12 ค่าอุณหภูมิและความชื้นสัมพัทธ์ภายในแบบจำลองโรงเรือนเลี้ยงไก่ ณ เวลา 16.56 น.

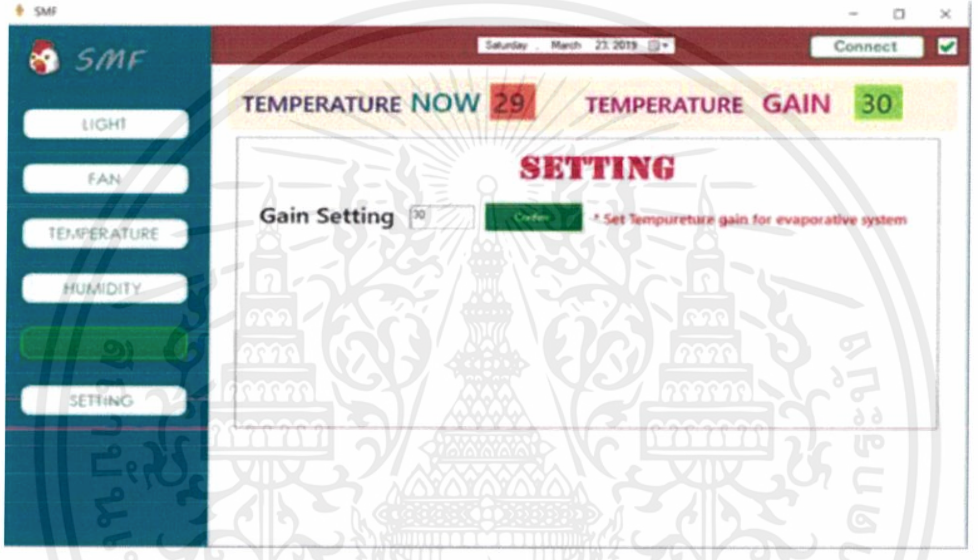
One

No	Date	Time	Temp	Humi
15810	23/03/2019	17:03:29	29C	85%
15811	23/03/2019	17:03:31	29C	85%
15812	23/03/2019	17:03:33	29C	85%
15813	23/03/2019	17:03:35	29C	85%
15814	23/03/2019	17:03:37	29C	85%

รูปที่ 4.13 ค่าอุณหภูมิและความชื้นสัมพัทธ์ภายในแบบจำลองโรงเรือนเลี้ยงไก่ ณ เวลา 17.03 น.

จากผลการทดลองในช่วงเวลากลางวันจะเห็นได้ว่า จากรูปที่ 4.8 ถึง 4.13 ใช้เวลาเพียง 13 นาทีทำให้ให้อุณหภูมิจาก 34 องศาเซลเซียส ลดลงมาเหลือเพียง 29 องศาเซลเซียส ซึ่งลดลงมาถึง 5 องศาเซลเซียส และจะเห็นว่า เมื่ออุณหภูมิลดต่ำลงค่าของความชื้นสัมพัทธ์จะสูงขึ้น หลังจากนั้น

เราได้ทำการผลปรับค่าเกณฑ์อุณหภูมิใหม่ที่ 30 องศาเซลเซียส เพื่อให้ระบบหล่อเย็นหยุดทำงาน โดยแสดงในรูปที่ 4.14 เพื่อทดลองดูว่าที่อุณหภูมิ 29 องศาเซลเซียส จะสามารถรักษาความคงที่ของอุณหภูมินี้ได้นานเท่าไหร่ ผลปรากฏว่าคงรักษาอุณหภูมิได้นานถึง 31 นาที ซึ่งแสดงดังรูปที่ 4.15 และจากผลการทดลองทั้งหมดนั้นสามารถแสดงในรูปแบบของกราฟได้ 2 รูปแบบ ได้แก่ กราฟความสัมพันธ์ระหว่างอุณหภูมิกับเวลา และ กราฟความสัมพันธ์ระหว่างความชื้นสัมพัทธ์กับเวลา โดยจะแสดงดังรูปที่ 4.16 4.17 ตามลำดับ

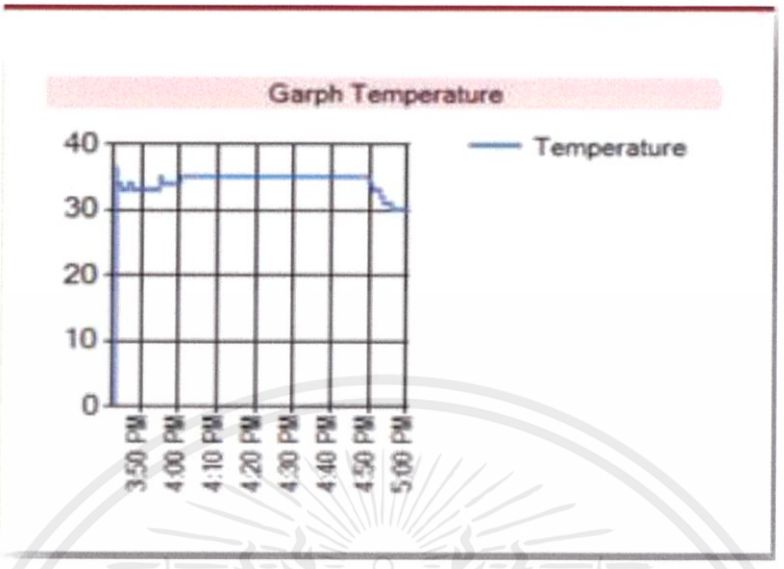


รูปที่ 4.14 การปรับค่าเกณฑ์อุณหภูมิใหม่

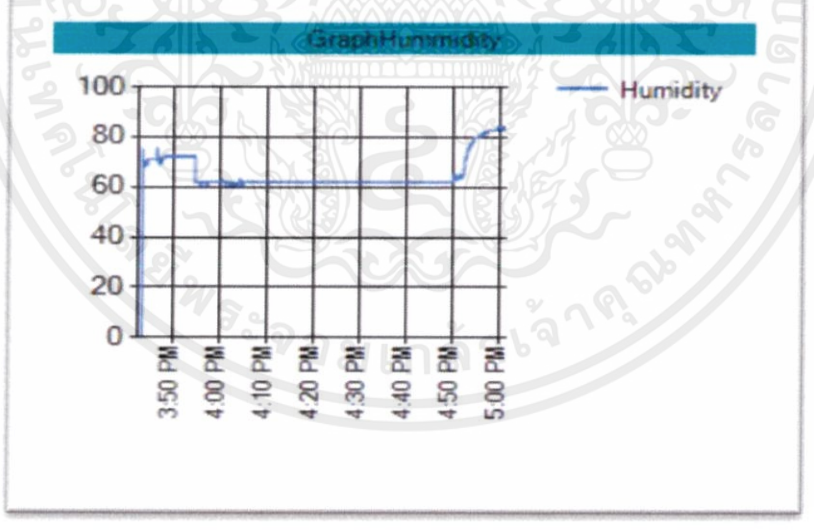
One

No	Date	Time	Temp	Humi
16743	23/03/2019	17:34:35	29C	83%
16744	23/03/2019	17:34:37	29C	83%
16745	23/03/2019	17:34:39	29C	83%
16746	23/03/2019	17:34:41	30C	83%
16747	23/03/2019	17:34:43	30C	83%
16748	23/03/2019	17:34:45	30C	83%
16749	23/03/2019	17:34:47	30C	83%
16750	23/03/2019	17:34:49	30C	83%
16751	23/03/2019	17:34:51	30C	83%
16752	23/03/2019	17:34:53	30C	83%

รูปที่ 4.15 ค่าอุณหภูมิและความชื้นสัมพัทธ์ภายในแบบจำลองโรงเรือนเลี้ยงไก่ ณ เวลา 17.34 น.



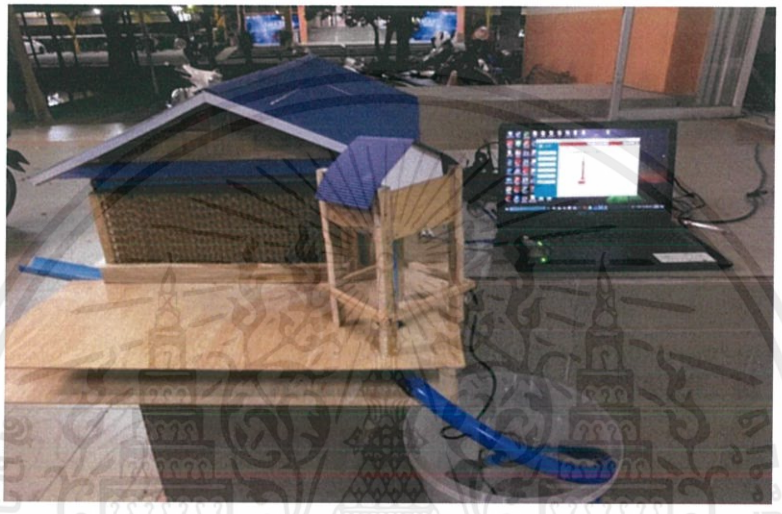
รูปที่ 4.16 กราฟความสัมพันธ์ระหว่างอุณหภูมิกับเวลาที่แสดงบนหน้าจอแสดงผล ช่วงเวลากลางวัน



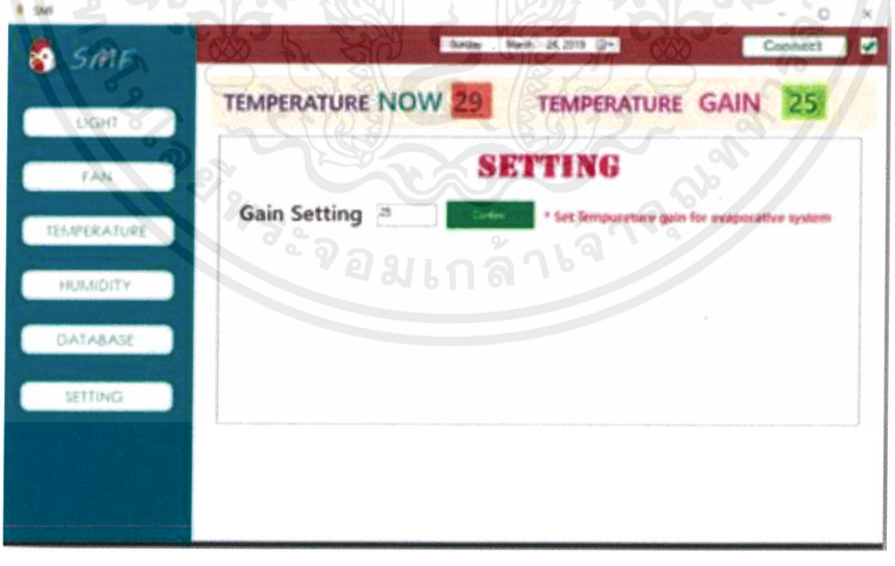
รูปที่ 4.17 กราฟความสัมพันธ์ระหว่างความชื้นสัมพัทธ์กับเวลาที่แสดงบนหน้าจอแสดงผล ช่วงเวลากลางวัน

4.3.2 การทดลองในช่วงเวลากลางคืน

เราจะนำแบบจำลองโรงเรือนเลี้ยงไก่มาทดลองในช่วงเวลากลางคืน โดยจะเริ่มจากการกำหนดค่าเกณฑ์อุณหภูมิที่ 25 องศาเซลเซียส เพื่อทดสอบว่าอุณหภูมิในแบบจำลองจะสามารถลดลงได้กี่องศาเซลเซียสเมื่อเทียบกับอุณหภูมิภายนอก ซึ่งจะแสดงได้ดังรูปที่ 4.18 และ 4.19



รูปที่ 4.18 แบบจำลองโรงเรือนเลี้ยงไก่ที่ตั้งไว้ในช่วงเวลากลางคืน



รูปที่ 4.19 หน้าจอแสดงผลที่แสดงค่าอุณหภูมิภายในแบบจำลองโรงเรือนเลี้ยงไก่และค่าเกณฑ์อุณหภูมิที่เรากำหนดเองในเวลากลางคืน

ซึ่งได้ผลการทดลองดังรูปต่อไปนี้

One

No	Date	Time	Temp	Humi
19520	24/03/2019	19:46:12	29C	89%
19521	24/03/2019	19:46:14	29C	89%
19522	24/03/2019	19:46:16	28C	89%
19523	24/03/2019	19:46:18	28C	89%
19524	24/03/2019	19:46:20	28C	89%

รูปที่ 4.20 ค่าอุณหภูมิและความชื้นสัมพัทธ์ภายในแบบจำลองโรงเรือนเลี้ยงไก่ ณ เวลา 19.46 น.

One

No	Date	Time	Temp	Humi
19949	24/03/2019	20:00:30	28C	90%
19950	24/03/2019	20:00:32	28C	90%
19951	24/03/2019	20:00:34	27C	90%
19952	24/03/2019	20:00:36	27C	90%
19953	24/03/2019	20:00:38	27C	90%

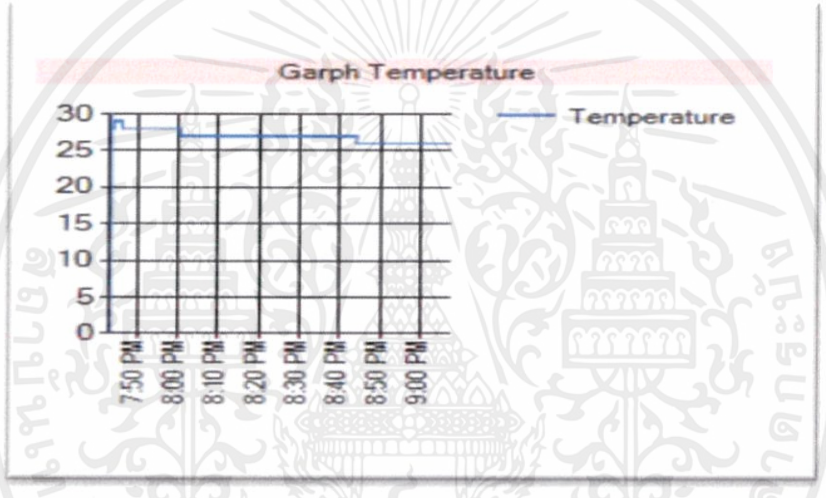
รูปที่ 4.21 ค่าอุณหภูมิและความชื้นสัมพัทธ์ภายในแบบจำลองโรงเรือนเลี้ยงไก่ ณ เวลา 20.00 น.

One

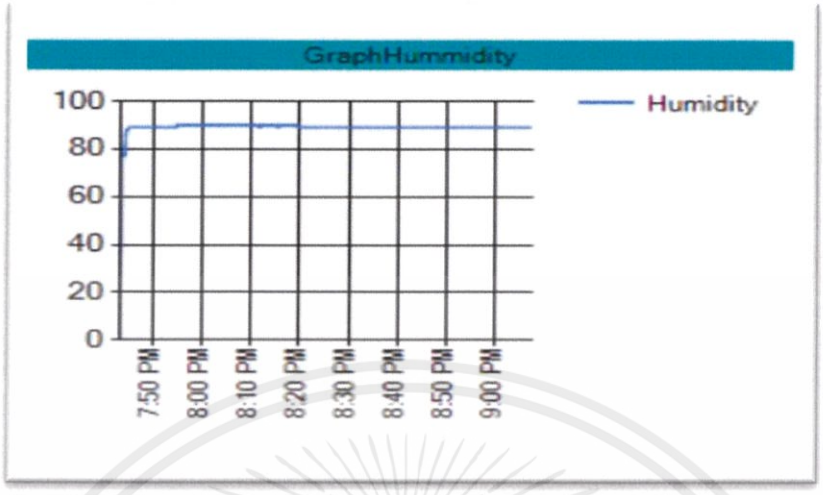
No	Date	Time	Temp	Humi
21252	24/03/2019	20:43:56	27C	89%
21253	24/03/2019	20:43:58	27C	89%
21254	24/03/2019	20:44:00	27C	89%
21255	24/03/2019	20:44:02	26C	89%
21256	24/03/2019	20:44:04	26C	89%

รูปที่ 4.22 ค่าอุณหภูมิและความชื้นสัมพัทธ์ภายในแบบจำลองโรงเรือนเลี้ยงไก่ ณ เวลา 20.44 น.

ผลการทดลองในช่วงเวลากลางคืนจะเห็นได้ว่า จากรูปที่ 4.20 – 4.22 เริ่มต้นจากอุณหภูมิภายในแบบจำลองโรงเรือนเลี้ยงไก่ 29 องศาเซลเซียส ลดลงเหลือ 26 องศาเซลเซียส ซึ่งลดลงมา 3 องศาเซลเซียส โดยใช้เวลาประมาณ 1 ชั่วโมง จะเห็นว่าช่วงเวลากลางคืนมีอุณหภูมิต่ำกว่าอุณหภูมิช่วงเวลากลางวันและมีความชื้นสัมพัทธ์สูงกว่าช่วงเวลากลางวัน และจากผลการทดลองทั้งหมดนั้นสามารถแสดงในรูปแบบของกราฟได้ 2 รูปแบบ ได้แก่ กราฟความสัมพันธ์ระหว่างอุณหภูมิกับเวลา และ กราฟความสัมพันธ์ระหว่างความชื้นสัมพัทธ์กับเวลา โดยจะแสดงดังรูปที่ 4.23 และ 4.24 ตามลำดับ



รูปที่ 4.23 กราฟความสัมพันธ์ระหว่างอุณหภูมิกับเวลาที่แสดงบนหน้าจอแสดงผล ช่วงเวลากลางคืน



รูปที่ 4.24 กราฟความสัมพันธ์ระหว่างความชื้นสัมพัทธ์กับเวลาที่แสดงบนหน้าจอแสดงผล ช่วงเวลากลางคืน

จากผลการทดลองที่ 4.3.1 และ 4.3.2 จะเห็นได้ว่าค่าอุณหภูมิเริ่มต้นมีค่าแตกต่าง ถึงแม้การทดลองจะไม่สามารถลดค่าอุณหภูมิทั้ง 2 ช่วงเวลาให้เท่ากันได้ แต่ทั้ง 2 ช่วงเวลาก็สามารถลดอุณหภูมิลงให้ถึงค่าที่เหมาะสมต่อการเลี้ยงไก่ได้ เพราะว่าขึ้นอยู่กับสภาพของอุณหภูมิภายนอกด้วย

4.4 ผลการทดลองในการเก็บค่าในฐานข้อมูล

ค่าของข้อมูลอุณหภูมิและความชื้นสัมพัทธ์ภายในแบบจำลองโรงเรือนเลี้ยงไก่แบบปิด และข้อมูลของวันและเวลา จะถูกส่งมาเก็บไว้ในฐานข้อมูล และสามารถเปิดดูย้อนหลังได้ตามต้องการ โดยจะแสดงดังรูปที่ 4.25

Day4

No	Date	Time	Temperature	Humidity	Click to Add
1378	11/03/2562	13:49:44	27 C	44.6 %	
1379	11/03/2562	13:49:45	27 C	44.6 %	
1380	11/03/2562	13:49:46	27 C	44.6 %	
1381	11/03/2562	13:49:47	27 C	44.6 %	
1382	11/03/2562	13:49:48	27 C	44.6 %	
1383	11/03/2562	13:49:49	27 C	44.6 %	
1384	11/03/2562	13:49:50	27 C	44.6 %	
1385	11/03/2562	13:49:51	27 C	44.6 %	
1386	11/03/2562	13:49:52	27 C	44.7 %	
1387	11/03/2562	13:49:53	27 C	44.7 %	
1388	11/03/2562	13:49:54	27 C	44.7 %	
1389	11/03/2562	13:49:55	27 C	44.7 %	
1390	11/03/2562	13:49:56	27 C	44.7 %	
1391	11/03/2562	13:49:57	27 C	44.7 %	
1392	11/03/2562	13:49:58	27 C	44.7 %	
1393	11/03/2562	13:49:59	27 C	44.7 %	
1394	11/03/2562	13:50:00	27 C	44.7 %	
1395	11/03/2562	13:50:01	27 C	44.7 %	
1396	11/03/2562	13:50:02	27 C	44.7 %	
1397	11/03/2562	13:50:03	27 C	44.7 %	
1398	11/03/2562	13:50:04	27.1 C	45 %	
1399	11/03/2562	13:50:05	27.1 C	45 %	

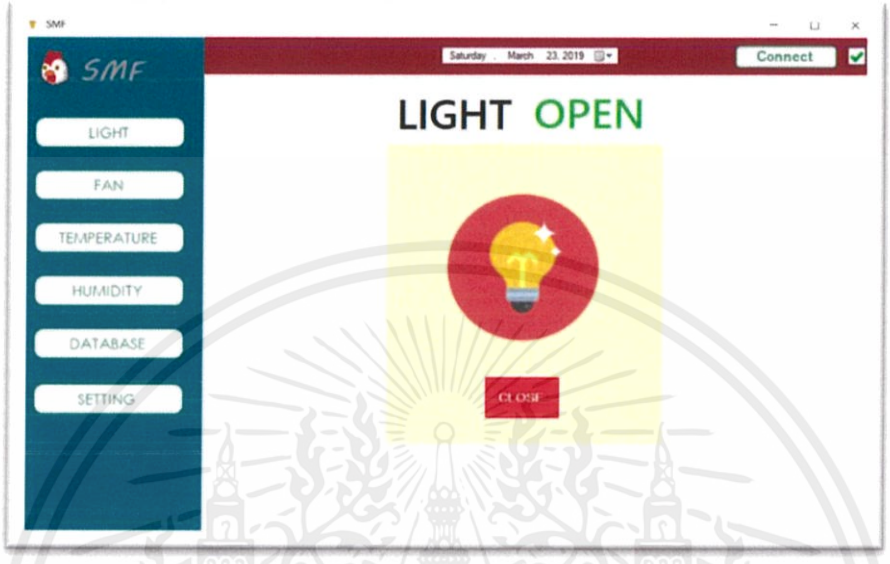
Record: 1 of 4328 No Filter Search

รูปที่ 4.25 ค่าอุณหภูมิและความชื้นสัมพัทธ์ ณ วันที่และเวลาต่างๆ

4.5 การทำงานของหน้าจอแสดงผล

เป็นส่วนของการกำหนดค่าเกณฑ์อุณหภูมิที่เราต้องการ เมื่อค่าอุณหภูมิและความชื้นสัมพัทธ์ภายในแบบจำลองโรงเรือนเลี้ยงไก่แบบปิดที่ถูกส่งข้อมูลจาก XBee ที่เป็นตัว Router มายัง XBee ที่เป็นตัว Coordinator ได้เข้าสู่หน้าจอแสดงผล และหน้าจอแสดงผลนี้สามารถสั่งเปิดปิด ไฟ พัดลม และแสดงค่าอุณหภูมิและความชื้นสัมพัทธ์ภายในแบบจำลองโรงเรือนเลี้ยงไก่แบบปิดได้ โดยจะแสดงได้ดังรูปที่ 4.26 4.27 4.28 4.29 4.30 และ 4.31 ตามลำดับ

ตัวเลือก LIGHT ใช้ในการสั่งเปิดปิดไฟ

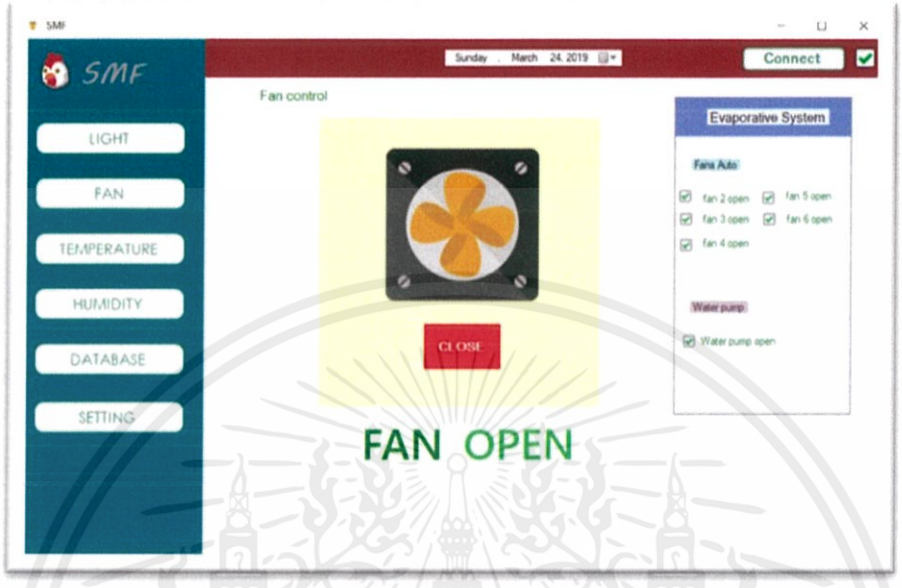


รูปที่ 4.26 หน้าจอที่ใช้กดสั่งให้เปิดปิดไฟ

(ก) (ข)

รูปที่ 4.27 (ก) เมื่อสั่งเปิดไฟแล้วไฟติด (ข) เมื่อสั่งปิดไฟแล้วไฟดับ

ตัวเลือก FAN ใช้ในการสั่งเปิดปิดพัดลม และแสดงการทำงานของระบบหล่อเย็น

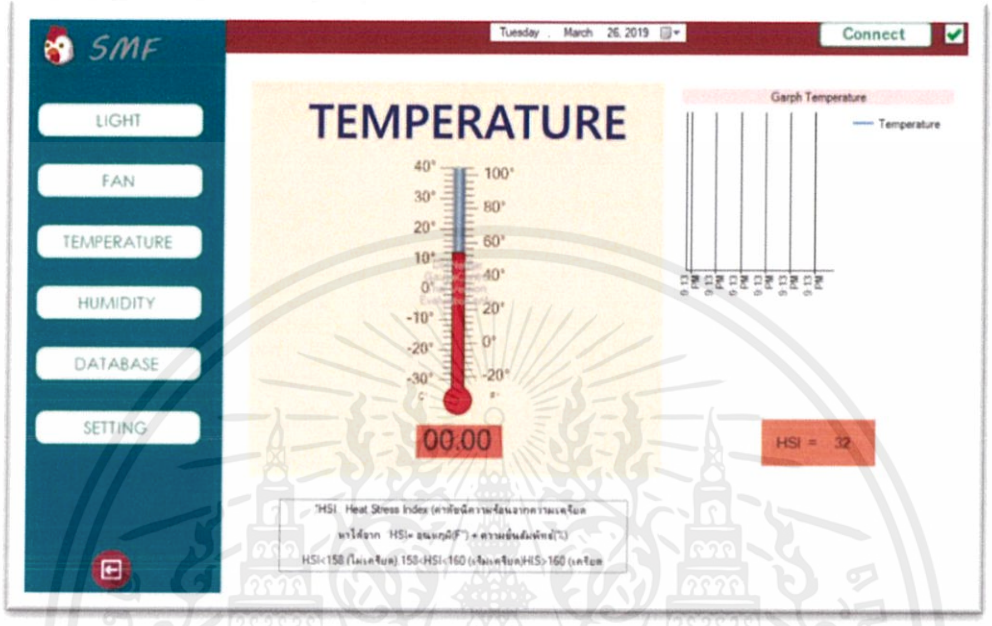


รูปที่ 4.28 หน้าจอที่ใช้กดสั่งให้เปิดปิดพัดลมและแสดงการทำงานของระบบหล่อเย็น

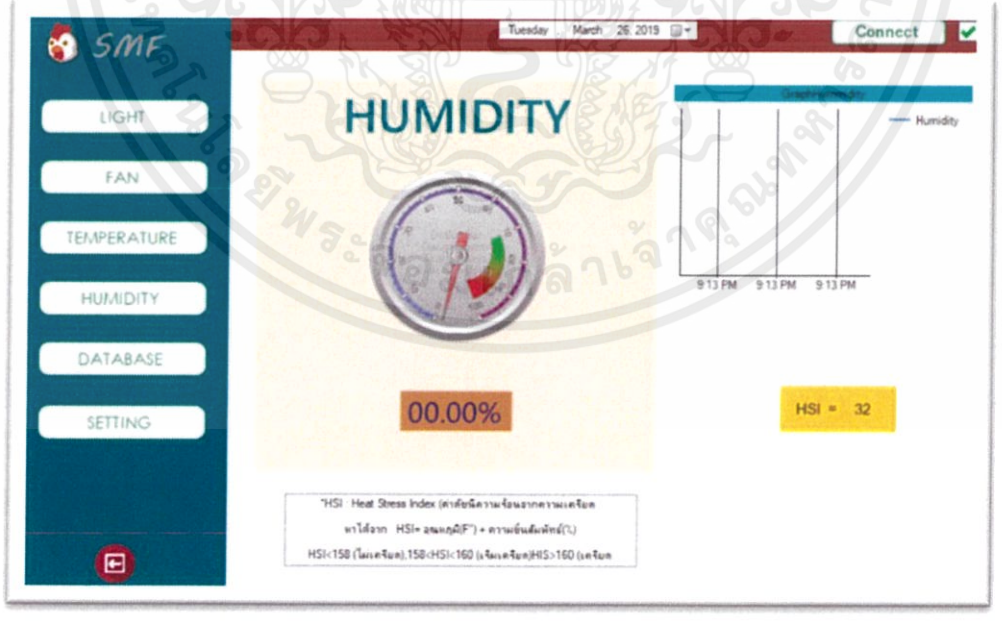
(ก) (ข)

รูปที่ 4.29 (ก) เมื่อสั่งปิดพัดลมดับ (ข) เมื่อสั่งเปิดพัดลมติด

ตัวเลือก TEMPERATURE และ HUMIDITY เป็นหน้าจอที่แสดงค่าอุณหภูมิและความชื้นสัมพัทธ์ภายในแบบจำลองโรงเรือนเลี้ยงไก่ และแสดงค่าดัชนีความร้อนจากความเครียด (HSI)



รูปที่ 4.30 ค่าอุณหภูมิภายในแบบจำลองโรงเรือนเลี้ยงไก่แบบปิด



รูปที่ 4.31 ค่าความชื้นสัมพัทธ์ภายในแบบจำลองโรงเรือนเลี้ยงไก่แบบปิด

บทที่ 5
สรุปผลและข้อเสนอแนะ

5.1 สรุปผล

จากการทดสอบการทำงานของระบบ พบว่าระบบสามารถใช้ XBee ในการสื่อสารรับส่งข้อมูลระหว่างอุปกรณ์ต้นทางกับอุปกรณ์ปลายทางที่อยู่ในระยะไกลได้ โดยอุปกรณ์ต้นทางจะติดตั้งอยู่ที่แบบจำลองโรงเรือนเลี้ยงไก่เพื่อทำการวัดค่าอุณหภูมิและความชื้นสัมพัทธ์ได้โดยใช้เซ็นเซอร์ DHT22 ทำงานร่วมกับไมโครคอนโทรเลอร์ แล้วส่งค่าข้อมูลผ่าน XBee ไปยังอุปกรณ์ปลายทาง โดยอุปกรณ์ปลายทางนั้นจะใช้เป็นคอมพิวเตอร์คอยรับค่าข้อมูลจากอุปกรณ์ต้นทางโดยใช้โปรแกรม Visual Studio ในการสร้างแอปพลิเคชั่นในการแสดงผลค่าข้อมูล เก็บข้อมูลเข้าฐานข้อมูล และควบคุมสั่งการกลับไปยังอุปกรณ์ต้นทาง โดยสามารถใช้ควบคุมเปิดปิดไฟ พัดลม และส่งค่าอุณหภูมิที่เราต้องการให้กับโรงเรือนเพื่อใช้ในการควบคุมการทำงานของระบบหล่อเย็น ซึ่งระบบหล่อเย็นจะทำงานอัตโนมัติเพื่อปรับค่าอุณหภูมิภายในโรงเรือนให้เป็นไปตามเกณฑ์ที่ต้องการได้ โดยจะมีการควบคุมการเปิดปิดพัดลม 5 ตัว และ ปั๊มน้ำเพื่อให้สามารถคงค่าอุณหภูมิให้เป็นไปตามเกณฑ์ที่เรากำหนดไว้ ทั้งนี้ระยะห่างในการติดตั้งอุปกรณ์ต้นทางและปลายทาง โดยใช้ XBee เป็นตัวรับส่งข้อมูลจากการทดลองนั่นสามารถได้ไกลถึง 500 เมตร ในพื้นที่โล่ง หรือ Line-Of-Sight หากมีการเพิ่มจำนวน XBee จะทำให้สามารถเพิ่มระยะห่างในการรับส่งข้อมูลได้มากยิ่งขึ้น

5.2 ข้อเสนอแนะ

ปริญญานิพนธ์นี้สามารถนำไปใช้ได้จริงแต่มีข้อจำกัดอยู่บ้างบางอย่างดังนี้

5.2.1 ในการใช้งานจริงการวัดค่าอุณหภูมิและความชื้นในโรงเรือนนั้นมีปัจจัยที่ต้องคำนึงถึงหลายอย่าง ไม่ว่าจะเป็นปัจจัยด้านสภาพแวดล้อม ปัจจัยด้านสภาพอากาศ และ ปัจจัยด้านอื่นๆ เนื่องด้วยปัจจัยเหล่านี้อาจทำให้การวัดค่าอุณหภูมิและค่าความชื้นอาจไม่แม่นยำเท่าที่ควรในบางครั้ง

5.2.2 พัฒนาระบบภายในโรงเรือนจำลองให้มีระบบเสมือนฟาร์มจริงมากขึ้น เช่น มีการจำลองการให้อาหาร มีควบคุมค่าแก๊สภายในโรงเรือน สามารถทดลองให้ใช้กับสัตว์จริงๆได้ เป็นต้น เพื่อที่จะทำให้ได้ผลการทดลองที่ใกล้เคียงกับสภาพความเป็นจริงมากยิ่งขึ้น

5.2.3 หากสามารถสร้างแอปพลิเคชั่นให้สามารถรองรับการใช้ที่หลากหลายและมีประสิทธิภาพมากยิ่งขึ้น เช่น มีการแสดงค่าข้อมูลผ่านมือถือ หรือสามารถเข้าดูข้อมูลได้จากที่ห่างไกลได้จะเป็นการอำนวยความสะดวกให้กับผู้ใช้งานมากยิ่งขึ้น

บรรณานุกรม

[1] Robert Faludi. (2012). Building Wireless Sensor Networks. New York : O'Reilly Media, Inc

[2] เอกชัย มะการ. (2552). เรียนรู้ เข้าใจ ใช้งาน ไมโครคอนโทรเลอร์ตระกูล AVR ด้วย Arduino. กรุงเทพฯ : บริษัท อีทีที จำกัด

[3] https://www.makerlab-electronics.com/product/xbee-pro-s2c-wire-antenna/

[4] https: / / www. thaieasyelec. com/ article- wiki/ basic- electronics/ what- is-zigbee.html

[5] http://www.arduino.codemobiles.com/article/15/พื้นฐานการสื่อสารด้วยโมดูล-xbee-part3-communication

[6] https://thaieasyelec.com/article-wiki/embedded-electronics-application/xbee-api-mode-tutorial-and-lab.html

[7] http://itbakery.net/2018/03/21/เข้าใจ-ขา-pinout-ของ-arduino/

[8] https://leeselectronic.com/en/product/7375.html

[9] https://www.electronicsnotes.com/articles/electronic_components/electrical-electronic-relay/what-is-a-relay-basics.php

[10] https://www.modmypi.com/raspberry-pi/relays-and-home-automation-1032/relay-boards-1033/8-channel-12v-relay-module

[11] https://mall.factomart.com/principle-of-switching-power-supply/

[12] http://www.natres.psu.ac.th/animal/doc/515-497-1-2560

ภาคผนวก ก

โค้ด XBee ฝั่ง Router

```
#include <XBee.h>
#include <Printers.h>
#include <AltSoftSerial.h>
#include <SoftwareSerial.h>

XBeeWithCallbacks xbee;
SoftwareSerial XBee(2,3);
AltSoftSerial SoftSerial;
#define DebugSerial Serial
#define XBeeSerial SoftSerial

#include <DHT.h>
#define DHTPIN1 10
#define DHTPIN2 11
#define DHTTYPE DHT22
DHT dht1(DHTPIN1, DHTTYPE);
DHT dht2(DHTPIN2, DHTTYPE);


struct PayloadStruct
{
  uint8_t  Index;
  float fTemperature;
  float fHumidity;
} __attribute__((packed));

PayloadStruct
  payload1;
```

```
int header_0;
float humi_0;
float temp_0;
int Num;
float t;
float h;
float t2;
float h2;
float t1;
float h1;

int ledpin = 5;
int fan2_pin = 7;
int fan1_pin = 6;
int fan34_pin = 8;
int fan56_pin = 9;
int pump_pin = 4;
int length_rx;

float temp_vb;
float temp_vb1;

void setup() {
  DebugSerial.begin(9600);
  DebugSerial.println(F("Starting..."));
  XBee.begin(9600);
  xbee.begin(XBee);
  dht1.begin();
  dht2.begin();
```

```
  // Setup callbacks
sendPacket();
pinMode(ledpin,OUTPUT);
pinMode(fan1_pin,OUTPUT);
pinMode(fan2_pin,OUTPUT);
pinMode(fan34_pin,OUTPUT);
pinMode(fan56_pin,OUTPUT);
pinMode(pump_pin,OUTPUT);
digitalWrite(ledpin,HIGH);
digitalWrite(fan1_pin,HIGH);
digitalWrite(fan2_pin,HIGH);
digitalWrite(fan34_pin,HIGH);
digitalWrite(fan56_pin,HIGH);
digitalWrite(pump_pin,HIGH);
}
void processRxPacket(ZBRxResponse& rx, uintptr_t) {
  DebugSerial.print(F("Received packet from "));
  printHex(DebugSerial, rx.getRemoteAddress64());
  DebugSerial.println();
  DebugSerial.print(F("Payload: "));
  DebugSerial.write(rx.getData(), rx.getDataLength());
  DebugSerial.println();
  length_rx = rx.getDataLength();
  char pay_rx[5];
  pay_rx[0] = rx.getData(0);
  pay_rx[1] = rx.getData(1);
  pay_rx[2] = rx.getData(2);
  pay_rx[3] = rx.getData(3);
```

```
  String mystr(pay_rx);
  String header = mystr.substring(0,1);
  String  temp = mystr.substring(1,3);
  header_0 = header.toInt();
  DebugSerial.println(header);
  DebugSerial.print("Header Code = ");
  DebugSerial.println(header_0);

   if(header_0 == 9 ){
  temp_vb = temp.toInt();
  DebugSerial.print("Temperature Gain = ");
  DebugSerial.println(temp_vb);
  }
}
void sendPacket() {
    // Prepare the Zigbee Transmit Request API packet
    ZBTxRequest txRequest;
    XBeeAddress64 address = XBeeAddress64(0x13a200, 0x40a78cd2);
    txRequest.setAddress64(0x000000000000FFFF);
    h1 = dht1.readHumidity();
    t1 = dht1.readTemperature();
    h2 = dht2.readHumidity();
    t2 = dht2.readTemperature();
    t = (t2+t1)/2;
    h = (h2+h1)/2;
    payload1.Index = 1;
    payload1.fTemperature = t;
    payload1.fHumidity = h;
```

```
    char payload[12];
    ((payload1.fTemperature)+String(payload1.fHumidity)).toCharArray(payload,12);
    DebugSerial.println(payload);
    txRequest.setPayload(payload, sizeof(payload));
    xbee.send(txRequest);
}
void sendPacket_ledon(){
    // Prepare the Zigbee Transmit Request API packet
    ZBTxRequest txRequest;
    XBeeAddress64 address = XBeeAddress64(0x13a200, 0x40a78cd2);
    txRequest.setAddress64(0x000000000000FFFF);
    char payload[5];
    ("8"+String(Num)).toCharArray(payload,4);
    DebugSerial.println(payload);
    txRequest.setPayload(payload, sizeof(payload));
    xbee.send(txRequest);
}
unsigned long last_tx_time = 0;
void loop() {
  // Check the serial port to see if there is a new packet available
  xbee.loop();
   if (millis() - last_tx_time > 10000) {
    last_tx_time = millis();
    sendPacket();
    sendPacket_ledon();
  }
  xbee.onZBExplicitRxResponse(processRxPacket);
  xbee.onZBRxResponse(processRxPacket);
  temp_0 = t;
```

```
if(temp_vb !=0)
{
if(temp_0 > temp_vb+1 && temp_0 <= temp_vb +2)
{
 digitalWrite(fan2_pin,LOW);
 digitalWrite(fan34_pin,LOW);
 digitalWrite(fan56_pin,HIGH);
 digitalWrite(pump_pin,LOW);
 Serial.println("3 FAN  and PUMP is ON ");
 Num = 5;
 delay(100);
}
 else if(temp_0 > temp_vb+2 )
{
 digitalWrite(fan2_pin,LOW);
 digitalWrite(fan34_pin,LOW);
 digitalWrite(fan56_pin,LOW);
 digitalWrite(pump_pin,LOW);
 Serial.println(" ALL Devices is ON ");
 Num = 6;
 delay(100);
}
 if(temp_vb >= temp_0 && temp_0 > temp_vb-2)
 {
  digitalWrite(fan56_pin,HIGH);
  digitalWrite(pump_pin,HIGH);
  Serial.println(" 2 FAN and PUMP is OFF");
  Num = 7;
  delay(100);
```

```
 }
 else if(temp_0 <= temp_vb-2)
 {
  digitalWrite(fan56_pin,HIGH);
  digitalWrite(fan34_pin,HIGH);
  digitalWrite(fan2_pin,HIGH);
  digitalWrite(pump_pin,HIGH);
  Serial.println("ALL Devices is OFF");
  Num = 8;
  delay(100);
 }
if(header_0 == 1)
{
 digitalWrite(ledpin,LOW);
 Serial.println("LED IS ON");
 Num = 1;
 delay(1000);
}
else if(header_0 == 2)
{
 digitalWrite(ledpin,HIGH);
Serial.println("LED IS OFF");
Num = 2;
delay(1000);
}
else if(header_0 == 3)
{
digitalWrite(fan1_pin,LOW);
Serial.println("FAN 1 is ON");
```

```
    Num = 3;
    delay(1000);
    }
    else if(header_0 == 4)
    {
     digitalWrite(fan1_pin,HIGH);
    Serial.println("FAN 1 IS OFF");
    Num = 4;
    delay(1000);
    }
    else if(header_0 == 5){
      fan34_pin = 8;
      fan56_pin = 9;
    }
    else if(header_0 == 6){
      fan34_pin = 9;
      fan56_pin = 8;
    }
     }
     else
     {
       Serial.println("Wait Gain");
     }
    delay(1000);
}
```

ภาคผนวก ข
โค้ด XBee ฝั่ง Coordinator

```
#include <XBee.h>
#include <Printers.h>
#include <AltSoftSerial.h>
#include <SoftwareSerial.h>

int temp_vb;
int humi_0;
int temp_0;
int head_control_int;

#include "binary.h"
XBeeWithCallbacks xbee;
SoftwareSerial XBee(2,3);
AltSoftSerial SoftSerial;
#define DebugSerial Serial
#define XBeeSerial SoftSerial

unsigned long last_tx_time = 0;
int read_input;

void setup() {
  //Setup debug serial output
  DebugSerial.begin(9600);
  DebugSerial.println(F("Starting..."));
  // Setup XBee serial communication
  XBee.begin(9600);
  xbee.begin(XBee);
  delay(1);
  sendPacket5();
```

```
}
void processRxPacket(ZBRxResponse& rx, uintptr_t) {
  DebugSerial.print(F("Received packet from "));
  printHex(DebugSerial, rx.getRemoteAddress64());
  DebugSerial.println();
  DebugSerial.print(F("Payload: "));
  DebugSerial.write(rx.getData(), rx.getDataLength());
  DebugSerial.println();

  int length_rx = rx.getDataLength();
  if (length_rx >= 11){
  char pay_rx[12];

  pay_rx[0] = rx.getData(0);
  pay_rx[1] = rx.getData(1);
  pay_rx[2] = rx.getData(2);
  pay_rx[3] = rx.getData(3);
  pay_rx[4] = rx.getData(4);
  pay_rx[5] = rx.getData(5);
  pay_rx[6] = rx.getData(6);
  pay_rx[7] = rx.getData(7);
  pay_rx[8] = rx.getData(8);
  pay_rx[9] = rx.getData(9);
  pay_rx[10] = rx.getData(10);
  pay_rx[11] = rx.getData(11);
  String mystr(pay_rx);

  String temp = mystr.substring(0,5);
  String humi = mystr.substring(5,11);
```

```
temp_0 = temp.toInt();
Serial.print("Temperature is = ");
Serial.println(temp_0);
humi_0 = humi.toInt();
Serial.print("Humidity is = ");
Serial.println(humi_0);
}else if(length_rx <= 5 ){
char pay_rx[5];
pay_rx[0] = rx.getData(0);
pay_rx[1] = rx.getData(1);
pay_rx[2] = rx.getData(2);
pay_rx[3] = rx.getData(3);
pay_rx[4] = rx.getData(4);
String mystr(pay_rx);
String head_control = mystr.substring(0,2);
head_control_int = head_control.toInt();
Serial.print("Header is = ");
Serial.println(head_control_int);
if (head_control_int == 81){
  Serial.println(" LIGHT is Open");
  }
  else if(head_control_int == 82){
  Serial.println(" LIGHT is Close");
  }
  else if(head_control_int == 83){
  Serial.println(" Fan1 is ON");
  }
  else if(head_control_int == 84){
  Serial.println("  Fan1 is Close");
```

```
    }
    else if(head_control_int == 85){
    Serial.println("3 FAN and PUMP is ON");
    }
    else if(head_control_int == 86){
    Serial.println("ALL Devices is ON");
    }
    else if(head_control_int == 87){
    Serial.println("2 FAN and PUMP is OFF");
    }
    else if(head_control_int == 88){
    Serial.println("ALL Devices is OFF");
    }
  }
}
void sendPacket() {
    // Prepare the Zigbee Transmit Request API packet
    ZBTxRequest txRequest;
    XBeeAddress64 address = XBeeAddress64(0x13a200, 0x40a4d607);
    txRequest.setAddress64(0x000000000000FFFF);
    char payload[5];
    ("9"+String(temp_vb)).toCharArray(payload,4);
    DebugSerial.println(payload);
    txRequest.setPayload(payload, sizeof(payload));
    xbee.send(txRequest);
    DebugSerial.println();
    }
void sendPacket1() {
    // Prepare the Zigbee Transmit Request API packet
```

```
    ZBTxRequest txRequest;
    XBeeAddress64 address = XBeeAddress64(0x13a200, 0x40a4d607);
    txRequest.setAddress64(0x000000000000FFFF);
    char payload[5];
    ("1"+String("")).toCharArray(payload,4);
    DebugSerial.println(payload);
    txRequest.setPayload(payload, sizeof(payload));
    xbee.send(txRequest);
}
void sendPacket2() {
    // Prepare the Zigbee Transmit Request API packet
    ZBTxRequest txRequest;
    XBeeAddress64 address = XBeeAddress64(0x13a200, 0x40a4d607);
    txRequest.setAddress64(0x000000000000FFFF);
    char payload[5];
    ("2"+String("")).toCharArray(payload,4);
    DebugSerial.println(payload);
    txRequest.setPayload(payload, sizeof(payload));
    xbee.send(txRequest);
}
void sendPacket3() {
    // Prepare the Zigbee Transmit Request API packet
    ZBTxRequest txRequest;
    XBeeAddress64 address = XBeeAddress64(0x13a200, 0x40a4d607);
    txRequest.setAddress64(0x000000000000FFFF);
    char payload[5];
    ("3"+String("")).toCharArray(payload,4);
    DebugSerial.println(payload);
    txRequest.setPayload(payload, sizeof(payload));
```

```
    xbee.send(txRequest);
}
void sendPacket4() {
    // Prepare the Zigbee Transmit Request API packet
    ZBTxRequest txRequest;
    XBeeAddress64 address = XBeeAddress64(0x13a200, 0x40a4d607);
    txRequest.setAddress64(0x000000000000FFFF);
    char payload[5];
    ("4"+String("")).toCharArray(payload,4);
    DebugSerial.println(payload);
    txRequest.setPayload(payload, sizeof(payload));
    xbee.send(txRequest);
}
void sendPacket5() {
    // Prepare the Zigbee Transmit Request API packet
    ZBTxRequest txRequest;
    XBeeAddress64 address = XBeeAddress64(0x13a200, 0x40a4d607);
    txRequest.setAddress64(0x000000000000FFFF);
    char payload[5];
    ("5"+String("")).toCharArray(payload,4);
    DebugSerial.println(payload);
    txRequest.setPayload(payload, sizeof(payload));
    xbee.send(txRequest);
}
void sendPacket6() {
    // Prepare the Zigbee Transmit Request API packet
    ZBTxRequest txRequest;
    XBeeAddress64 address = XBeeAddress64(0x13a200, 0x40a4d607);
    txRequest.setAddress64(0x000000000000FFFF);
```

```
    char payload[5];
    ("6"+String("")).toCharArray(payload,4);
    DebugSerial.println(payload);
    txRequest.setPayload(payload, sizeof(payload));
    xbee.send(txRequest);
}
void loop() {
  // Check the serial port to see if there is a new packet available
  xbee.loop();
  // Send a packet every 10 seconds
 /* if (millis() - last_tx_time > 10000) {
    last_tx_time = millis();
    sendPacket();
  }*/
  xbee.onZBExplicitRxResponse(processRxPacket);
  xbee.onZBRxResponse(processRxPacket);
  read_input = Serial.parseInt();
  if(Serial.available()>0)
  {
      if(read_input == 1)
      {
        Serial.println("Send 1");
        sendPacket1();
        delay(1000);
       }
      else if(read_input == 2)
      {
        Serial.println("Send 2");
        sendPacket2();
```

```
  delay(1000);
}
else if(read_input == 3)
{
  Serial.println("Send 3");
  sendPacket3();
  delay(1000);
}
else if(read_input == 4)
{
  Serial.println("Send 4");
  sendPacket4();
  delay(1000);
}
else if(read_input == 5)
{
  Serial.println("Send 5");
  sendPacket5();
  delay(1000);
}
 else if(read_input == 6)
{
  Serial.println("Send 6");
  sendPacket6();
  delay(1000);
}
else
{
temp_vb = read_input;
```

```
        Serial.println("Send Temperature Gain");
        sendPacket();
        delay(1000);
        }
    }
}
```