ระบบควบคุมอุปกรณ์เครื่องใช้ไฟฟ้าภายในบ้านระยะไกล

Remote Electronic Control System

โดย

นางสาวชาริณีย์ อิสระพันธุ์ดัง

นางสาวณัฐกานต์ กิตติพรพันธ์

นายธนัตถ์ สุประสงค์สิน

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรวิศวกรรมศาสตรบัณฑิต

สาขาวิศวกรรมโทรคมนาคม

คณะวิศวกรรมศาสตร์

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ปีการศึกษา 2556

ระบบควบคุมอุปกรณ์เครื่องใช้ไฟฟ้าภายในบ้านระยะไกล

Remote Electronic Control System

โดย

นางสาวชาริณีย์ อิสระพันธุ์ดัง

นางสาวณัฐกานต์ กิตติพรพันธ์

นายธนัตถ์ สุประสงค์สิน

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต

สาขาวิชาวิศวกรรมโทรคมนาคม

คณะวิศวกรรมศาสตร์

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ปีการศึกษา 2556

ระบบควบคุมอุปกรณ์เครื่องใช้ไฟฟ้าภายในบ้านระยะไกล

Remote Electronic Control System

โดย

นางสาวชาริณีย์	อิสระพันธุ์ดัง	53010359
นางสาวณัฐกานต์	กิตติพรพันธ์	53010445
นายธนัตถ์	สุประสงค์สิน	53010669

อาจารย์ที่ปรึกษา

ผศ. ดร. สมเกียรติ ฤกษ์วรัญญู

อาจารย์ที่ปรึกษาร่วม

รศ. ดร. พรชัย ทรัพย์นิธิ

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต

สาขาวิชาวิศวกรรมโทรคมนาคม

คณะวิศวกรรมศาสตร์

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ปีการศึกษา 2556

ผ่านการตรวจรูปเล่มแล้ว

(.....)

อาจารย์ที่ปรึกษา

13 / 3 / 57

ผ่านการตรวจชิ้นงานแล้ว

(.....)

กรรมการผู้ตรวจชิ้นงาน

17 / 3 / 57

ปริญญานิพนธ์ปีการศึกษา 2556
สาขาวิชาวิศวกรรมโทรคมนาคม
คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง
เรื่อง ระบบควบคุมอุปกรณ์เครื่องใช้ไฟฟ้าภายในบ้านระยะไกล
REMOTE ELECTRONIC CONTROL SYSTEM

ผู้จัดทำ

1.	นางสาวชาริณีย์ อิสระพันธุ์ดัง	53010359
2.	นางสาวณัฐกานต์ กิตติพรพันธ์	53010445
3.	นายธนัตถ์ สุประสงค์สิน	53010669

..... อาจารย์ที่ปรึกษา
(ผศ. ดร. สมเกียรติ ฤกษ์วรัญญู)

..... อาจารย์ที่ปรึกษาร่วม
(รศ. ดร. พรชัย ทรัพย์นิธิ)

กิตติกรรมประกาศ

โครงงานนี้สำเร็จลุล่วงได้ด้วยความอนุเคราะห์จาก ผศ.ดร.สมเกียรติ ฤกษ์วรัญญู อาจารย์ที่ปรึกษาโครงงาน และ รศ.ดร.พรชัย ทรัพย์นิธิ อาจารย์ที่ปรึกษาร่วม ที่ได้ให้คำเสนอแนะ แนวคิด ตลอดจนแก้ไขปัญหาข้อบกพร่องต่างๆมาโดยตลอด จนโครงงานนี้เสร็จสมบูรณ์ ขอกราบขอบพระคุณเป็นอย่างสูง

ขอกราบขอบพระคุณ คุณพ่อ คุณแม่ และครอบครัวที่ให้คำปรึกษาในเรื่องต่างๆ รวมทั้งยังเป็นกำลังใจที่ดีเสมอมา

ขอขอบคุณ นายนิพัทธ์ กานต์กัมพล รุ่นพี่ที่ให้คำปรึกษา และช่วยแก้ปัญหาของโครงงานนี้

ขอบคุณเพื่อนๆในกลุ่ม และในห้องโปรเจคที่ให้คำปรึกษาและช่วยเหลือในด้านต่างๆ

ผู้จัดทำมีความซาบซึ้งในความกรุณาอันดียิ่งจากทุกท่านที่ได้กล่าวนามมา และขอกราบขอบพระคุณมา ณ โอกาสนี้

นางสาวชาริณีย์ อิสระพันธุ์ดัง
นางสาวณัฐกานต์ กิตติพรพันธ์
นายธนัตถ์ สุประสงค์สิน
ผู้จัดทำ

ระบบควบคุมอุปกรณ์เครื่องใช้ไฟฟ้าภายในบ้านระยะไกล
REMOTE ELECTRONIC CONTROL SYSTEM

โดย นางสาวชาริณีย์ อิสระพันุดัง 53010359
นางสาวณัฐกานต์ กิตติพรพันธ์ 53010445
นายธนัตถ์ สุประสงค์สิน 53010669

อาจารย์ที่ปรึกษา ผศ. ดร. สมเกียรติ ฤกษ์วรัญญู

บทคัดย่อ

ในปัจจุบันไม่ว่าเราจะไปที่ไหนก็จะพบเห็นผู้คนใช้โทรศัพท์มือถือแบบสมาร์ทโฟนกันมากมาย ไม่ว่าจะเป็นนักเรียน นักศึกษา หรือ คนทำงานก็ตาม ซึ่งโดยส่วนมากก็จะเลือกใช้เพราะความสะดวกสบายในการใช้บริการระบบอินเตอร์เน็ตที่สามารถจะเข้าอินเตอร์เน็ตเพื่อสืบค้นข้อมูลติดต่อเพื่อนหรือติดต่องานได้ทุกที่ ระบบอินเตอร์เน็ตจึงได้เข้ามามีบทบาทในชีวิตประจำวันของผู้คนมากขึ้นเรื่อยๆ และมีแนวโน้มว่าในอนาคตระบบอินเตอร์เน็ตจะสามารถใช้งานได้อย่างรวดเร็วและครอบคลุมมากยิ่งขึ้น

ดังนั้นจึงได้นำสมาร์ทโฟนและระบบอินเตอร์เน็ตมาใช้เพื่อควบคุมเครื่องใช้ไฟฟ้าภายในบ้านโดยส่งข้อมูลผ่านทีซีพีไอพีไปยังไมโครคอนโทรลเลอร์และให้ไมโครคอนโทรลเลอร์ควบคุมอุปกรณ์ไฟฟ้า โดยจะแบ่งการทำงานออกเป็น 3 ส่วน คือ ส่วนพัฒนาแอพลิเคชั่นบนระบบปฏิบัติการแอนดรอยด์ ส่วนการประมวลผลบนไมโครคอนโทรลเลอร์ และส่วนฮาร์ดแวร์ โดยจะสามารถควบคุมเครื่องใช้ไฟฟ้าภายในบ้านได้จากทุกที่จากสมาร์ทโฟนที่มีบริการระบบอินเตอร์เน็ต

สารบัญ

สารบัญ (ต่อ)

สารบัญ (ต่อ)

หน้า

สารบัญ (ต่อ)

สารบัญรูป

สารบัญรูป (ต่อ)

สารบัญรูป (ต่อ)

สารบัญรูป (ต่อ)

หน้า

รูปที่

สารบัญตาราง

บทที่ 1

บทนำ

1.1 ความเป็นมาและความสำคัญของปัญหา

เนื่องมาจากทุกวันนี้ สมาร์ทโฟน (Smartphone) ได้เข้ามามีความสำคัญในชีวิตประจำวันของมนุษย์เรามากขึ้น จนแทบจะเป็นอีกปัจจัยหนึ่งในการดำรงชีวิต เพราะคำว่าสมาร์ทโฟนนั้น ไม่ได้หมายความถึงมือถือที่ทำได้แค่โทรเข้า และโทรออก แต่มีแอพพลิเคชั่น (application) ที่รองรับการใช้งานได้หลากหลายรูปแบบ เช่น ถ่ายรูป ปรับแต่งภาพ รับ-ส่งอีเมล์ ซื้อ-ขายสิ่งของ พยากรณ์อากาศ เกมส์ต่างๆ แม้กระทั่งอ่านหนังสือ หรือนิตยสารออนไลน์ โดยเฉพาะอย่างยิ่งแอพพลิเคชั่นที่เกี่ยวกับการติดต่อสื่อสาร ทั้งพูดคุยธรรมดา ไปถึงการแชร์ภาพถ่าย หรือวีดีโอคอล (video call) ซึ่งแอพพลิเคชั่นต่างๆ เหล่านี้ ก็มาจากการพัฒนาเพื่อความสะดวกสบายที่มากขึ้น ในการทำการต่างๆ ผ่านสมาร์ทโฟน จากข้อนี้เอง จึงเป็นแรงบันดาลใจให้คิดแอพพลิเคชั่นเพื่อการควบคุมอุปกรณ์ไฟฟ้าภายในบ้านขึ้นมา เพื่อความสะดวกสบายในการเปิด-ปิด อุปกรณ์ไฟฟ้าต่างๆ ภายในบ้าน โดยที่เราไม่ต้องเดินไปถึงอุปกรณ์ต่างๆ เหล่านั้น เพื่อเปิด-ปิด ด้วยตนเอง เป็นการอำนวยความสะดวกทั้งบุคคลธรรมดา และรวมไปถึงผู้พิการ ที่มีความลำบากในการเข้าถึงอุปกรณ์ต่างๆ ด้วยตนเอง แอพพลิเคชั่นนี้เปรียบเสมือนรีโมต ที่มีระยะการใช้งานที่ไกลมาก เพราะทำงานผ่านระบบอินเทอร์เน็ต ทำให้สามารถสั่งงานอุปกรณ์ต่างๆ ภายในบ้านได้ แม้ผู้ใช้จะยังอยู่ที่ทำงานก็ตาม คุณสามารถสั่งปิดไฟ หรือปิดอุปกรณ์ที่คุณลืมปิดการทำงานของมัน เมื่อคุณรีบร้อนออกจากบ้าน แอพพลิเคชั่นนี้จึงช่วยในความสะดวกสบาย รวมไปถึงความปลอดภัยของบ้านเรือนผู้ใช้ได้อีกด้วย

1.2 วัตถุประสงค์

1) เพื่อศึกษาการพัฒนาแอพพลิเคชันบนโทรศัพท์มือถือ ระบบปฏิบัติการแอนดรอยด์
2) เพื่อศึกษาการควบคุมอุปกรณ์ต่างๆ โดยใช้ไมโครคอนโทรลเลอร์
3) เพื่อสร้างระบบต้นแบบ ในการควบคุมการทำงานของเครื่องใช้ไฟฟ้าในระยะไกล และประยุกต์ใช้งานแอพพลิเคชั่นบนแอนดรอยด์ เพื่อควบคุมเครื่องใช้ไฟฟ้าได้

1.3 ขอบเขตของปริญญานิพนธ์

ระบบควบคุมการเปิด-ปิดอุปกรณ์เครื่องใช้ไฟฟ้า ผ่านโทรศัพท์มือถือระบบปฏิบัติการแอนดรอยด์ (android) เป็นการพัฒนาเขียนแอพพลิเคชัน ให้สามารถสั่งงานเปิด-ปิดอุปกรณ์ไฟฟ้าผ่านทางแอพพลิเคชั่นบนแอนดรอยด์สมาร์ทโฟน โดยผ่านระบบ TCP/IP ส่งผ่านข้อมูล ไปยังส่วนฮาร์ดแวร์ (hardware) ซึ่งข้อมูลจะถูกประมวลผลและ ส่งต่อข้อมูลสถานะการเปิด-ปิดไปยังไมโครคอนโทรลเลอร์ เพื่อควบคุมเครื่องใช้ไฟฟ้า ตามคำสั่งที่ได้รับมาจากแอพพลิเคชั่นบนแอนดรอยด์

บทที่ 2

ทฤษฎีและหลักการที่เกี่ยวข้อง

2.1 ภาษาจาวา (JAVA) [1]

จาวา หรือ Java programming language คือภาษาโปรแกรมเชิงวัตถุ พัฒนาโดย เจมส์ กอสลิง และวิศวกรคนอื่นๆ ที่บริษัท ซัน ไมโครซิสเต็มส์ ภาษานี้มีจุดประสงค์เพื่อใช้แทนภาษาซีพลัสพลัส (C++) โดยรูปแบบที่เพิ่มเติมขึ้นคล้ายกับภาษาอ็อบเจกต์ทีฟซี (Objective-C) แต่เดิม ภาษานี้เรียกว่า ภาษาโอ๊ก (Oak) ซึ่งตั้งชื่อตามต้นโอ๊กใกล้ที่ทำงานของ เจมส์ กอสลิง แล้วภายหลังจึงเปลี่ยนไปใช้ชื่อ "จาวา" ซึ่งเป็นชื่อกาแฟแทน จุดเด่นของภาษาจาวา อยู่ที่ผู้เขียนโปรแกรมสามารถใช้หลักการของ Object-Oriented Programming มาพัฒนาโปรแกรมของตนด้วยจาวาได้

ภาษาจาวา เป็นภาษาสำหรับเขียนโปรแกรมที่สนับสนุนการเขียนโปรแกรมเชิงวัตถุ (OOP : Object-Oriented Programming) โปรแกรมที่เขียนขึ้นถูกสร้างภายในคลาส ดังนั้นคลาสคือที่เก็บเมทอด (Method) หรือพฤติกรรม (Behavior) ซึ่งมีสถานะ (State) และรูปพรรณ (Identity) ประจำพฤติกรรม (Behavior)

2.1.1 ข้อดีของ ภาษาจาวา

2.1.1.1 ภาษาจาวา เป็นภาษาที่สนับสนุนการเขียนโปรแกรมเชิงวัตถุแบบสมบูรณ์ ซึ่งเหมาะสำหรับพัฒนาระบบที่มีความซับซ้อน การพัฒนาโปรแกรมแบบวัตถุจะช่วยให้เราสามารถใช้คำหรือชื่อ ต่าง ๆ ที่มีอยู่ในระบบงานนั้นมาใช้ในการออกแบบโปรแกรมได้ ทำให้เข้าใจได้ง่ายขึ้น

2.1.1.2 โปรแกรมที่เขียนขึ้นโดยใช้ภาษาจาวา จะมีความสามารถทำงานได้ในระบบปฏิบัติการที่แตกต่างกัน ไม่จำเป็นต้องดัดแปลงแก้ไขโปรแกรม เช่น หากเขียนโปรแกรมบนเครื่อง Sun โปรแกรมนั้นก็สามารถถูกคอมไพล์ (compile) และรัน (run) บนเครื่องพีซีธรรมดาได้

2.1.1.3 ภาษาจาวามีการตรวจสอบข้อผิดพลาดทั้งตอนระยะเวลาในการคอมไพล์ (compile time) และระยะเวลาในการรัน (runtime) ทำให้ลดข้อผิดพลาดที่อาจเกิดขึ้นในโปรแกรม และช่วยให้ debug โปรแกรมได้ง่าย

2.1.1.4 ภาษาจาวามีความซับซ้อนน้อยกว่าภาษา C++ เมื่อเปรียบเทียบโค้ด (code) ของโปรแกรมที่เขียนขึ้นโดยภาษาจาวา กับ C++ พบว่า โปรแกรมที่เขียนโดยภาษา จาวา จะมีจำนวนโค้ดน้อยกว่าโปรแกรมที่เขียนโดยภาษา C++ ทำให้ใช้งานได้ง่ายกว่าและลดความผิดพลาดได้มากขึ้น

2.1.1.5 ภาษาจาวาถูกออกแบบมาให้มีความปลอดภัยสูงตั้งแต่แรก ทำให้โปรแกรมที่เขียนขึ้นด้วยจาวามีความปลอดภัยมากกว่าโปรแกรมที่เขียนขึ้น ด้วยภาษาอื่น เพราะ Java มี security ทั้ง low level และ high level ได้แก่ electronic signature, public andprivate key management, access control และ certificatesของ

2.1.1.6 มี IDE, application server, และ library ต่าง ๆ มากมายสำหรับจาวาที่เราสามารถใช้งานได้โดยไม่ต้องเสียค่าใช้จ่าย ทำให้เราสามารถลดค่าใช้จ่ายที่ต้องเสียไปกับการซื้อ tool และ s/w ต่าง ๆ

2.1.2 ข้อเสียของ ภาษา Java

2.1.2.1 ทำงานได้ช้ากว่า native code (โปรแกรมที่ compile ให้อยู่ในรูปของภาษาเครื่อง) หรือโปรแกรมที่เขียนขึ้นด้วยภาษาอื่น อย่างเช่น C หรือ C++ ทั้งนี้ก็เพราะว่าโปรแกรมที่เขียนขึ้นด้วยภาษาจาวาจะถูกแปลงเป็นภาษากลาง ก่อน แล้วเมื่อโปรแกรมทำงานคำสั่งของภาษากลางนี้จะถูกเปลี่ยนเป็นภาษาเครื่องอีก ทีหนึ่ง ทีล่ะคำสั่ง (หรือกลุ่มของคำสั่ง) ณ runtime ทำให้ทำงานช้ากว่า native code ซึ่งอยู่ในรูปของภาษาเครื่องแล้วตั้งแต่ compile โปรแกรมที่ต้องการความเร็วในการทำงานจึงไม่นิยมเขียนด้วยจาวา

2.1.2.2 tool ที่มีในการใช้พัฒนาโปรแกรมจาวามักไม่ค่อยเก่ง ทำให้หลายอย่างโปรแกรมเมอร์จะต้องเป็นคนทำเอง ทำให้ต้องเสียเวลาทำงานในส่วนที่ tool ทำไม่ได้ ถ้าเราดู tool ของ MS จะใช้งานได้ง่ายกว่า และพัฒนาได้เร็วกว่า (แต่เราต้องซื้อ tool ของ MS และก็ต้องรันบน platform ของ MS)

2.1.3 Statement และ Expression

Statement คือ คำสั่งง่าย ๆ ที่ถูกเขียนลงในโปรแกรมภาษาทั่วไป ทำให้เกิดผลลัพธ์บางอย่างขึ้น มีหลายชนิดเช่น การกำหนดค่า การพิมพ์ผลลัพธ์ออกไป หรือการเรียกใช้ Method หรือฟังก์ชัน

Expression คือ statement ที่ให้ผลลัพธ์ออกมาเป็นค่า โดยค่า ๆ นั้นจะถูกเก็บไว้ใช้ในตลอดกระบวนการของโปรแกรม ไม่ว่าจะเป็นการนำไปใช้ทันที หรือ เก็บเอาไว้ใช้ใน statement อื่น ๆ หรือแม้แต่ไม่ได้ถูกนำไปใช้เลยก็ได้ โดยค่าที่ได้จาก statementแบบนี้ว่า return value

บาง Expression ให้ค่าคืน (return value) ออกมาเป็นตัวเลข บ้างให้ค่าเป็นตัวอักษร บ้างเป็นค่าความจริงทางตรรกศาสตร์ หรือแม้แต่ให้ค่าออกมาเป็น Object

Java Statement ที่เห็นอยู่ส่วนใหญ่เขียนอยู่ในบรรทัดเดียวในแต่ละคำสั่งนั้น ไม่ใช่เหตุผลที่ทำให้ Java สามารถแยกความเป็น statement ใด statement หนึ่งได้ สิ่งที่ทำให้รู้ได้ก็คือ การลงท้ายแต่ละคำสั่งนั้น ๆ ด้วยเครื่องหมาย เซมิโคลอน (;) และเมื่อเป็นดังนี้แล้ว โปรแกรมเมอร์สามารถเขียนโปรแกรมโดยมีหลาย ๆ statement ในบรรทัดเดียวกันได้

การจัดการกลุ่มของ Statement เราบอก Java ได้ว่า ไม่ว่าคำสั่งใดที่อยู่ภายใต้เครื่องหมาย { และ } จะถือเป็นกลุ่มคำสั่งหนึ่ง ๆ ซึ่งมักจะใช้ตามหลังติด ๆ มากับคำสั่งประเภท Block Statement เช่น If , While หรือ For

2.1.4 การสร้างตัวแปร

ก่อนที่จะมีการนำตัวแปรไปใช้ แน่นอนว่าเราจะต้องสร้างมันเพื่อให้ Java รู้จักเสียก่อน โดยกำหนดชื่อ และชนิดของข้อมูลที่จะถูกดำเนินการกับตัวแปรนั้น ๆ ดังนี้ <ชนิดข้อมูล> <ชื่อตัวแปร>;

ถ้ามีการสร้างตัวแปรที่มีชนิดของข้อมูลเหมือนกัน เราสามารถกำหนดมันไว้ใน statement เดียวกัน และแยกชื่อของตัวแปรเหล่านั้นออกด้วยเครื่องหมายคอมมา (,)

นอกเหนือจากการสร้างตัวแปร หลาย ๆ ตัว เรายังสามารถกำหนดค่าเริ่มต้นให้มันได้เลยพร้อม ๆ กับการสร้างตัวแปรโดยคั่นการกำหนดค่าเหล่านั้นด้วยเครื่องหมายคอมมาเช่นกัน (,)

2.1.5 การตั้งชื่อตัวแปร

ชื่อของตัวแปรจาวา จะต้องขึ้นต้นด้วยตัวอักษร หรือเครื่องหมาย _ หรือ $ แต่ไม่สามารถขึ้นต้นด้วยตัวเลขหรือเครื่องหมายอื่น ๆ ได้ ซึ่งหลังจากอักษรตัวแรกแล้วสามารถเป็นตัวอักษรหรือตัวเลขก็ได้ ที่สำคัญเป็นอย่างยิ่ง การอ่านชื่อต่าง ๆ ของจาวา ไม่ว่าจะเป็นตัวแปร หรือ Method จาวาถือว่าตัวเล็ก หรือตัวใหญ่นั้นเป็นคนละตัวกัน ดังนั้นตัวแปร A และ a จึงเป็นคนละตัวแปรกัน

การตั้งชื่อให้สื่อความหมาย ก็เป็นอีกเรื่องหนึ่งที่สำคัญ (ไม่ได้กับจาวา) แต่สำคัญต่อนักเขียนโปรแกรม เพื่อความเข้าใจและการแก้ไขในภายหลัง ว่าตัวแปรนั้น ๆ ถูกสร้างมาเพื่ออะไร ดังนั้นชื่อที่ดีจึงประกอบไปด้วยคำสองสามคำ (ซึ่งรวมกันแล้วให้ความหมาย) บวกกับการใช้ตัวอักษรเล็ก หรือใหญ่เพื่อแยกคำเหล่านั้น กฎง่ายการตั้งชื่อ คือ ตัวอักษรตัวแรกของคำแรก ให้ใช้ตัวเล็กตัวอักษรตัวแรกของคำอื่น ๆ ที่ตามมาให้ใช้ตัวใหญ่ และตัวอักษรที่เหลือให้ใช้ตัวเล็ก

2.1.6 ชนิดของตัวแปร

สามารถแบ่งตัวแปรจาวาออกเป็น 3 ชนิดหลัก ๆ คือ

2.1.6.1 ชนิดของตัวแปรที่ใช้สำหรับข้อมูลพื้นฐาน (Data Types)

2.1.6.2 ชนิดของตัวแปรที่ถ่ายทอดมากจากคลาส หรือ Interface (Class Types)

2.1.6.3 ชนิดของตัวแปรที่เป็นอาร์เรย์ (Array Types)

2.1.7 การกำหนดค่าให้กับตัวแปร

เมื่อเราได้ทำการสร้างตัวแปร เรียบร้อยแล้ว เราสามารถทำการกำหนดค่าให้กับตัวแปรเหล่านั้นได้ด้วยเครื่องหมายเท่ากับ (=)

2.1.8 Comments

การตั้งชื่อตัวแปรที่ให้ความหมายเป็นที่เข้าใจแก่ผู้อ่านโปรแกรมแล้ว อีกวิธีหนึ่งที่จะทำให้โปรแกรมเข้าใจง่าย หรืออ่านง่ายก็คือ การใส่คำอธิบาย หรือ อะไรก็แล้วแต่ที่ขยายความแก่โปรแกรม โดยที่ไม่กระทบต่อการคอมไพล์โปรแกรมมี 3 รูปแบบดังนี้

2.1.8.1 // ไม่ว่าอะไรก็แล้วแต่ที่อยู่หลังเครื่องหมายสแลชสองตัว ไปจนถึงก่อนการขึ้นบรรทัดใหม่ end of line

2.1.8.2 ถ้าต้องการใส่ Comment ที่มีความยาวมากกว่าหนึ่งบรรทัด เราคงไม่ต้องใส่สแลช กันทุกบรรทัดไป ทำได้โดยใส่ Comment ที่ต้องการนั้นไว้ระหว่างเครื่องหมาย /* และ */

2.2 Application android [2]

แอนดรอยด์ (Android) กูเกิลแอนดรอยด์ (Google Android) หรือ ระบบปฏิบัติการแอนดรอยด์ (Android Operating System) เป็นชื่อเรียกชุดซอฟท์แวร์ หรือแพลตฟอร์ม (Platform) สำหรับอุปกรณ์อีเล็กทรอนิกส์ ที่มีหน่วยประมวลผลเป็นส่วนประกอบ อาทิเช่น คอมพิวเตอร์, โทรศัพท์ (Telephone), โทรศัพท์เคลื่อนที่ (Cell phone), อุปกรณ์เล่นอินเตอร์เน็ตขนาดพกพา (MID) เป็นต้น แอนดรอยด์นั้น ถือกำเนิดอย่างเป็นทางการในวันที่ 5 พฤศจิกายน 2550 โดยบริษัท กูเกิล จุดประสงค์ของแอนดรอยด์นั้น มีจุดเริ่มต้นมาจากบริษัท Android Inc. ที่ได้นำเอาระบบปฏิบัติการลีนุกซ์ (Linux) ซึ่งนิยมนำไปใช้งานกับเครื่องแม่ข่าย (Server) เป็นหลักนำมาลดทอนขนาดตัว (แต่ไม่ลดทอนความสามารถ) เพื่อให้เหมาะสมกับการนำไปติดตั้งบนอุปกรณ์พกพา ที่มีขนาดพื้นที่จัดเก็บข้อมูลที่จำกัด โดยหวังว่า แอนดรอยด์ นั้นจะเป็นหุ่นยนต์ตัวน้อย ๆ ที่คอยช่วยเหลืออำนวยความสะดวกแก่ผู้ที่พกพามัน ไปในทุกที่ ทุกเวลา

2.2.1 Application Component คือ Component หลักที่ใช้ในการสร้าง Android Application โดย Application Component แบ่งออกเป็น 4 ประเภท ได้แก่ Activity, Service, Content Provider, และ Broadcast Receiver ซึ่งแต่ละประเภทของ Application Component นี้มีเป้าหมายในการใช้งานที่แตกต่างกัน มีรูปแบบการกระตุ้นให้เกิดการทำงานที่แตกต่างกัน (กล่าวคือบาง Application Component ถูกกระตุ้นให้เกิดการทำงานโดย System และบาง Application Component ถูกกระตุ้นให้เกิดการทำงานโดย Application Component อื่น ๆ) รวมถึงมีวงจรชีวิตที่แตกต่างกันด้วย (กล่าวคือแต่ละ Application Component จะมีรูปแบบที่แตกต่างกันว่า Application Component นั้น ๆ จะถูกสร้าง (Create) เมื่อใด หรือถูกทำลาย (Destroy) เมื่อใด)

2.2.2 Activity คือ Application Component ที่ใช้ในการควบคุมการสร้าง User Interface เช่น การแสดงผลหน้าจอรายการอีเมล์, การแสดงหน้าจอแบบฟอร์มการส่งอีเมล์ เป็นต้น

รวมถึงควบคุมการมีปฏิสัมพันธ์ระหว่างผู้ใช้กับ User Interface ด้วย เช่น เมื่อผู้ใช้เลือกรายการอีเมล์ก็จะทำการตอบสนองผู้ใช้โดยการแสดงข้อมูลรายการอีเมล์ที่เลือก เป็นต้น สำหรับการสร้าง Activity นั้น ทำได้โดยการสร้าง Class และให้สืบทอดจาก Class Activity หรือสืบทอดจาก Class ใด ๆ ก็ตามที่ได้รับการสืบทอดมาจาก Class Activity โดย Activity หนึ่ง ๆ จะควบคุมการแสดงผล User Interface หนึ่ง ๆ เท่านั้น และนั่นแสดงให้เห็นว่า Application หนึ่ง ๆ จะประกอบด้วย Activity จำนวนมากที่ทำงานร่วมกันอยู่ อย่างไรก็ตามถึงแม้ว่า Activity จะทำงานร่วมกัน แต่ Activity เหล่านั้นยังคงเป็นอิสระจากกัน

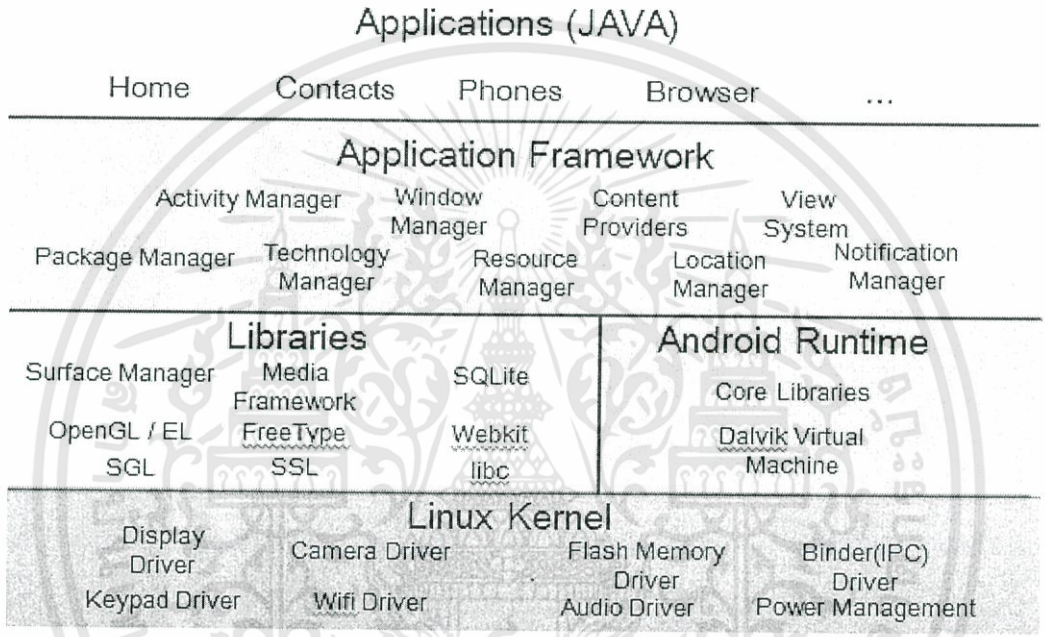
2.2.3 **Service** คือ Application Component ที่ไม่มี User Interface และจะทำการประมวลผลใน Background กล่าวคือเป็นการประมวลผลที่ดำเนินไปพร้อมกับที่ผู้ใช้สามารถไปใช้งาน Application อื่น ๆ ได้ หรือกล่าวอีกมุมหนึ่ง การประมวลผลใน Background คือการประมวลผลที่สามารถทำงานขนานกันกับการทำงานอื่น ๆ ของผู้ใช้ ทั้งนี้ก็เพื่อทำให้เกิดการทำงานใด ๆ โดยที่ผู้ใช้ไม่จำเป็นต้องอยู่ในหน้าจอนั้น ๆ ได้ ซึ่งอาจเป็นเพราะการทำงานนั้นต้องใช้ระยะเวลานาน เช่น การใช้ Service เปิดเพลง เพื่อให้ผู้ใช้สามารถไปใช้ Application อื่น ๆ ได้ แต่เพลงยังคงเล่นอยู่ หรือ การใช้ Service ดาวน์โหลดข้อมูลใด ๆ ที่มีขนาดใหญ่ เพื่อให้ผู้ใช้สามารถไปใช้ Application อื่น ๆ ได้ แต่การดาวน์โหลดยังคงดำเนินอยู่ เป็นต้น สำหรับการสร้าง Service นั้นทำได้โดยการสร้าง Class และให้สืบทอดจาก Class Service หรือสืบทอดจาก Class ใด ๆ ก็ตามที่ได้รับการสืบทอดมาจาก Class Service

2.2.4 **Content Provider** คือ Application Component ที่ทำหน้าที่ในการควบคุมข้อมูลใด ๆ ของ Application ที่ต้องการ Share ให้ Application อื่น ๆ สามารถนำข้อมูลนั้น ๆ ไปใช้งานได้ หรือกล่าวในทางกลับกันก็คือ Application ใด ๆ สามารถนำข้อมูล (รวมถึงแก้ไขข้อมูลได้ ถ้า Content Provider อนุญาติ) ของ Application อื่น ๆ มาใช้งานได้ โดยกระทำผ่าน Content Provider เช่น System ได้จัดเตรียม Content Provider ที่เป็นข้อมูลรายชื่อผู้ติดต่อ (Contact) ไว้ เพื่อให้ Application ที่ต้องการใช้ข้อมูลรายชื่อผู้ติดต่อนี้ สามารถนำข้อมูลไปใช้หรือแก้ไขข้อมูลได้ เป็นต้น สำหรับการสร้าง Content Provider นั้น ทำได้โดยการสร้าง Class และให้สืบทอดจาก Class ContentProvider หรือสืบทอดจาก Class ใด ๆ ก็ตามที่ได้รับการสืบทอดมาจาก Class ContentProvider

2.2.5 **Broadcast Receiver** คือ Application Component ที่ไม่มี User Interface โดยจะทำหน้าที่รับรู้สิ่งที่เกิดขึ้นของ System และนำมาบอกให้ผู้ใช้ได้รับรู้ เช่น เมื่อ Battery ต่ำ, เมื่อหน้าจอถูก Capture, เมื่อมีการพักหน้าจอ เป็นต้น ทั้งนี้ Application ใด ๆ สามารถนำ Broadcast Receiver มาใช้ประโยชน์ได้ เช่น เมื่อ Application ได้ Download ข้อมูลเสร็จเรียบร้อย เป็นต้น ซึ่งโดยส่วนมากแล้วการตอบสนองของ Broadcast Receiver จะกระทำผ่าน Notification เพื่อแจ้งสิ่งที่เกิดขึ้นให้ผู้ใช้ได้รับรู้ สำหรับการสร้าง Broadcast Receiver นั้น

ทำได้โดยการสร้าง Class และให้สืบทอดจาก Class BroadcastReceiver หรือสืบทอดจาก Class ใด ๆ ก็ตามที่ได้รับการสืบทอดมาจาก Class BroadcastReceiver

2.2.6 โครงสร้างของแอนดรอยด์ เป็นสิ่งสำคัญเพราะถ้านักพัฒนาโปรแกรม สามารถมองภาพโดยรวมของระบบได้ทั้งหมด จะให้สามารถเข้าใจถึงกระบวนการทำงานได้ดียิ่งขึ้น และสามารถนำไปช่วยในการออกแบบโปรแกรมที่ต้องการพัฒนา เพื่อให้เกิดประสิทธิภาพในการทำงาน ดังรูปที่ 2.1



รูปที่ 2.1 โครงสร้างของแอนดรอยด์

จากโครงสร้างของระบบปฏิบัติการแอนดรอยด์ จะสังเกตุได้ว่า มีการแบ่งออกมาเป็นส่วนๆ ที่มีความเกี่ยวเนื่องกัน โดยส่วนบนสุดจะเป็นส่วนที่ผู้ใช้งานทำการติดต่อโดยตรงซึ่งก็คือส่วนของ (Applications) จากนั้นก็จะลำดับลงมาเป็นองค์ประกอบอื่นๆตามลำดับ และสุดท้ายจะเป็นส่วนที่ติดต่อกับอุปกรณ์โดยผ่านทาง Linux Kernel โครงสร้างของแอนดรอยด์ พอที่จะอธิบายเป็นส่วนๆได้ดังนี้

2.2.6.1 Applications ส่วน Application หรือส่วนของโปรแกรมที่มีมากับระบบปฏิบัติการ หรือเป็นกลุ่มของโปรแกรมที่ผู้ใช้งานได้ทำการติดตั้งไว้ โดยผู้ใช้งานสามารถเรียกใช้โปรแกรมต่างๆได้โดยตรง ซึ่งการทำงานของแต่ละโปรแกรมจะเป็นไปตามที่ผู้พัฒนาโปรแกรมได้ออกแบบและเขียนโค้ดโปรแกรมเอาไว้

2.2.6.2 Application Framework เป็นส่วนที่มีการพัฒนาขึ้นเพื่อให้นักพัฒนาสามารถพัฒนาโปรแกรมได้สะดวก และมีประสิทธิภาพมากยิ่งขึ้น โดยนักพัฒนาไม่จำเป็นต้องพัฒนาในส่วนที่มีความยุ่งยากมากๆ เพียงแค่ทำการศึกษาถึงวิธีการเรียกใช้งาน Application Framework ในส่วนที่ต้องการใช้งาน แล้วนำมาใช้งาน ซึ่งมีหลายกลุ่มด้วยกัน ตัวอย่างเช่น

Activities Manager เป็นกลุ่มของชุดคำสั่งที่จัดการเกี่ยวกับวงจรการทำงานของหน้าต่างโปรแกรม (Activity)

1) Content Providers เป็นกลุ่มของชุดคำสั่ง ที่ใช้ในการเข้าถึงข้อมูลของโปรแกรมอื่น และสามารถแบ่งปันข้อมูลให้โปรแกรมอื่นเข้าถึงได้

2) View System เป็นกลุ่มของชุดคำสั่งที่เกี่ยวกับการจัดการโครงสร้างของหน้าจอที่แสดงผลในส่วนที่ติดต่อกับผู้ใช้งาน (User Interface)

3) Telephony Manager เป็นกลุ่มของชุดคำสั่งที่ใช้ในการเข้าถึงข้อมูลด้านโทรศัพท์ เช่นหมายเลขโทรศัพท์ เป็นต้น

4) Resource Manager เป็นกลุ่มของชุดคำสั่งในการเข้าถึงข้อมูลที่เป็น ข้อความ, รูปภาพ

5) Location Manager เป็นกลุ่มของชุดคำสั่งที่เกี่ยวกับตำแหน่งทางภูมิศาตร์ ที่ระบบปฏิบัติการได้รับค่าจากอุปกรณ์

6) Notification Manager เป็นกลุ่มของชุดคำสั่งที่จะถูกเรียกใช้เมื่อโปรแกรม ต้องการแสดงผลให้กับผู้ใช้งาน ผ่านทางแถบสถานะ (Status Bar) ของหน้าจอ

2.2.6.3 Libraries เป็นส่วนของชุดคำสั่งที่พัฒนาด้วย C/C++ โดยแบ่งชุดคำสั่งออกเป็นกลุ่มตามวัตถุประสงค์ของการใช้งาน เช่น Surface Manage จัดการเกี่ยวกับการแสดงผล, Media Framework จัดการเกี่ยวกับการการแสดงภาพและเสียง, Open GL | ES และ SGL จัดการเกี่ยวกับภาพ 3 มิติ และ 2 มิติ, SQLlite จัดการเกี่ยวกับระบบฐานข้อมูล เป็นต้น

2.2.6.4 Android Runtime จะมี Darvik Virtual Machine ที่ถูกออกแบบมา เพื่อให้ทำงานบนอุปกรณ์ที่มี หน่วยความจำ (Memmory), หน่วยประมวลผลกลาง (CPU) และพลังงาน (Battery) ที่จำกัด ซึ่งการทำงานของ Darvik Virtual Machine จะทำการแปลงไฟล์ที่ต้องการทำงาน ไปเป็นไฟล์ .DEX ก่อนการทำงาน เหตุผลก็เพื่อให้มีประสิทธิภาพเพิ่มขึ้นเมื่อใช้งานกับ หน่วยประมวลผลกลางที่มีความเร็วไม่มาก ส่วนต่อมาคือ Core Libraries ที่เป็นส่วนรวบรวมคำสั่งและชุดคำสั่งสำคัญ โดยถูกเขียนด้วยภาษาจาวา (Java Language)

2.2.6.5 Linux Kernel เป็นส่วนที่ทำหน้าที่หัวใจสำคัญ ในจัดการกับบริการหลักของระบบปฏิบัติการ เช่น เรื่องหน่วยความจำ พลังงาน ติดต่อกับอุปกรณ์ต่างๆ ความปลอดภัย เครือข่าย โดยแอนดรอยด์ได้นำเอาส่วนนี้มาจากระบบปฏิบัติการลินุกซ์ รุ่น 2.6 (Linux 26. Kernel) ซึ่งได้มีการออกแบบมาเป็นอย่างดี

2.3 เอ็กซ์เอ็มแอล (XML) [3]

เอ็กซ์เอ็มแอล ย่อมาจาก Extensible Markup Language เป็นภาษาหนึ่งที่ใช้ในการแสดงผลข้อมูล ถ้าเปรียบเทียบกับภาษาเอชทีเอ็มแอล (HTML) จะแตกต่างกันที่ภาษาเอชทีเอ็มแอล ถูกออกแบบมาเพื่อการแสดงผลอย่างเดียวเท่านั้น เช่นให้แสดงผลตัวเล็ก ตัวหนา ตัวเอียง เหมือนที่คุณเคยเห็นในเวบเพจทั่วไป แต่ภาษาเอ็กซ์เอ็มแอล นั้นถูกออกแบบมาเพื่อเก็บข้อมูล โดยทั้งข้อมูลและโครงสร้างของข้อมูลนั้นๆไว้ด้วยกัน

โครงสร้างประกอบด้วย แท็ก (Tag) เปิด และแท็กปิด เช่นเดียวกับภาษาเอชทีเอ็มแอล แต่ภาษาเอ็กซ์เอ็มแอล คุณสามารถสร้างแท็กรวมทั้งกำหนดโครงสร้างของข้อมูลได้เอง ซึ่งความสามารถตรงนี้ตัวภาษาเอชทีเอ็มแอล ทำไม่ได้เพราะภาษาเอชทีเอ็มแอล ตัวอย่างส่วนประกอบของเอ็กซ์เอ็มแอลดังนี้

2.3.1 แท็ก สำหรับใน XML แล้วแท็กมีความหมายในลักษณะเดียวกับที่ใช้ใน HTML tag คือข้อความที่อยู่ระหว่างสัญลักษณ์ "<" และ ">" มี 2 แบบคือ

2.3.2 Element คือโครงสร้างหลักของ XML ซึ่งอยู่ในรูปของแท็กจะมีลักษณะซ้อนกันเป็นชั้นๆโดย Element เริ่มต้นที่แท็กเปิดและสิ้นสุดที่แท็กปิดในแท็กดียวกัน และ Root element จะเป็น Element บนสุดของไฟล์ XML

2.3.3 Content ข้อมูลที่เก็บ

2.3.4 Attribute คือข้อมูลความหมายเพิ่มเติมเป็นค่าคงที่ ถูกเขียนอยู่ภายใน tag เปิด <....> จะมีมากกว่า 1 มี 1 อันหรือไม่มีเลยก็ได้

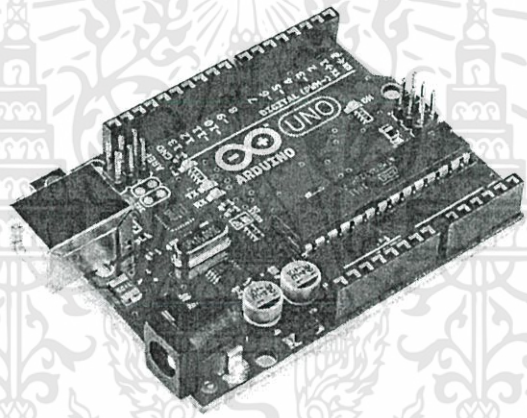
2.4 อาดูโน ไมโครคอนโทรลเลอร์ (Arduino Microcontroller) [4]

ไมโครคอนโทรลเลอร์ คือ อุปกรณ์ที่มีหน่วยประมวลผล และความจำขนาดเล็กภายในตัวเอง สามารถรับ-ส่งข้อมูลได้ทั้งแบบดิจิตอลและอนาล็อก ใช้พลังงานน้อย ทำให้เป็นที่นิยมในการใช้งานในรูปแบบฝังตัวกับอุปกรณ์ (Embedded) เช่น เครื่องใช้ไฟ้าอัจฉริยะทั้งหลาย เช่น รับสัญญาณจากสวิตช์ หรือเซนเซอร์ ควบคุมหลอดไฟ มอเตอร์ เป็นต้น

ในโครงงานนี้ ได้เลือกใช้บอร์ดไมโครคอนโทรลเลอร์อาดูโน (Arduino) ซึ่งเป็นแบบที่เรียกว่าโอเพ่นฮาร์ดแวร์ (Open Hardware) นั่นคือ อุปกรณ์ที่มีรูปแบบส่วนประกอบเป็นมาตรฐานที่เปิดเผย หมายความว่า สามารถทำขึ้นได้เอง โดยใช้รูปแบบที่มีการเปิดเผยทั่วไป และนำไปดัดแปลงต่อขยายและเพิ่มประสิทธิภาพการทำงานได้ (ภายใต้ Creative Commons License) สามารถซื้อหาได้ง่าย มีราคาถูกเมื่อเปรียบเทียบกับไมโครคอนโทรลเลอร์ชนิดอื่นๆ มีซอฟต์แวร์ (Software) ให้ใช้งานฟรี สามารถนำไปใช้งานทั่วไป หรือแบบธุรกิจได้ โดยไม่ต้องเสียค่าลิขสิทธิ์ เป็นรูปแบบที่มีข้อมูลมากที่สุดบนอินเตอร์เน็ต ใช้งานง่าย มีโปรแกรมพัฒนาที่ไม่ซับซ้อน แต่สามารถพัฒนาได้จนถึงขั้นมืออาชีพ เพราะพัฒนาโดยใช้คำสั่งเขียนโปรแกรมได้เสมือนโปรแกรมภาษาชั้นสูงทั่วไป (ภาษา C, C++ หรือ Java) และสามารถเปิดเผยซอร์ซโค้ด และนำไปพัฒนาต่อได้ เพิ่มเติม

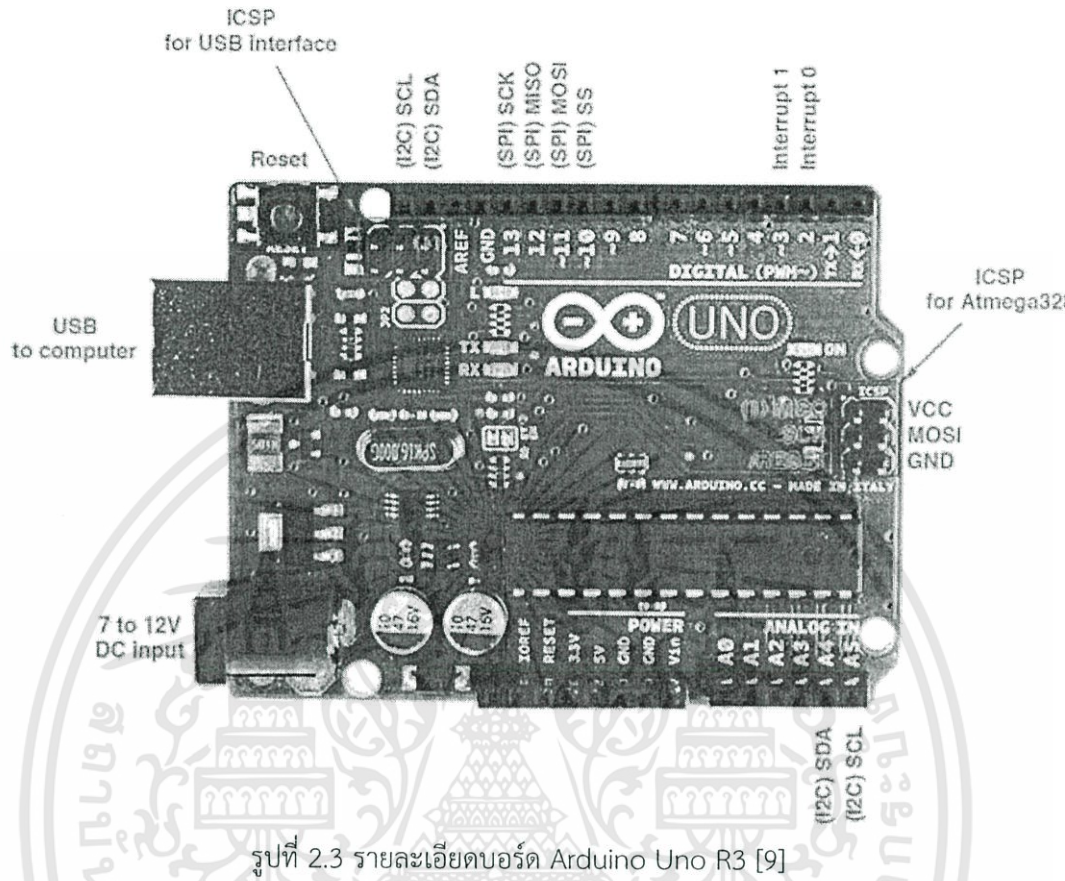
โค้ดเองได้ ทำงานได้หลายแพลตฟอร์ม (platform) นั่นคือทำงานได้ทั้งบนวินโดวส์ (Windows) แมคอินทอช (Macintosh OSX) หรือบนลีนุกซ์ (Linux) ในขณะที่บอร์ดอื่นๆ ทำงานได้เฉพาะบนวินโดวส์

บอร์ดอาดูโน พัฒนามาจากการใช้งานไมโครคอนโทรลเลอร์ในตระกูล AVR ที่กำลังได้รับความนิยมอย่างสูงทั่วโลก เพราะว่าเป็นโอเพ่นซอร์ซ (Open Source) สามารถดัดแปลงไปใช้งานได้ทั้งฮาร์ดแวร์และซอร์ฟแวร์ได้ทันที มีไลบรารี่ (Libaries) ต่างๆ ให้พร้อมเรียกใช้งานได้ทันทีมากมาย ครอบคลุมการติดต่อกับอินพุทและเอาท์พุท (I/O) ต่างๆ ได้อย่างกว้างมาก การใช้งานในการอัปโหลด (upload) โปรแกรม ใช้การเสียบสายยูเอสบี (USB) เข้ากับคอมพิวเตอร์ ก็จะใช้งานได้ทันที โดยบอร์ดอาดูโน่ ที่ใช้ในการทดลองการทำงานก่อนการทำบอร์ดเองในโครงงานนี้ ได้เลือกใช้ Arduino Uno R3 เป็นต้นแบบ แสดงดังรูปที่ 2.2



รูปที่ 2.2 บอร์ด Arduino Uno R3

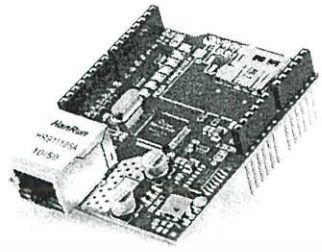
โดยบอร์ด Arduino Uno R3 เป็นบอร์ดไมโครคอนโทรลเลอร์ที่ใช้ชิป ATmega168 (datasheet) หรือ ATmega328 (datasheet) มีขาใช้งานที่เป็นดิจิทัล (digital) อินพุทและเอาท์พุท 14 ขา ขารับสัญญาณอานาล็อค (Analog) 6 ขา ใช้ออสซิเลเตอร์ (oscillator) 16 เมกะเฮิรตซ์ (MHz) การเชื่อมต่อแบบยูเอสบี พาวเวอร์แจ๊ค (Power Jack) ไอซีเอสพี เฮ้ดเดอร์ (ICSP header) และปุ่มรีเซ็ต (reset) เป็นบอร์ดไมโครคอนโทรลเลอร์ที่รวมทุกอย่างไว้ในบอร์ด ง่ายต่อการเชื่อมต่อกับคอมพิวเตอร์ ไม่ว่าจะเป็นทางสายยูเอสบี (USB cable) หรืออะแดปเตอร์แปลงไฟ (AC-to-DC adapter) ดังรูปที่ 2.3 ที่จะแสดงรายละเอียดของขาการใช้งานของบอร์ดไว้ด้วย



รูปที่ 2.3 รายละเอียดบอร์ด Arduino Uno R3 [9]

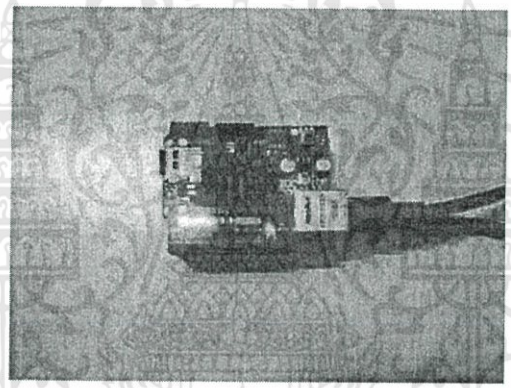
2.5 อาดูโน อีเธอร์เน็ตชิลด์ (Arduino Ethernet Shield) [5]

อาดูโนอีเธอร์เน็ตชิลด์เป็นอุปกรณ์เสริมที่ช่วยในการเชื่อมต่ออาดูโนไมโครคอนโทรลเลอร์ให้สามารถรับข้อมูลผ่านทาง TCP/IP ได้ โดยที่ Ethernet คำนี้จะหมายถึงส่วนของการสื่อสารระหว่างคอมพิวเตอร์ที่อยู่ภายใน Local Area Network (LAN) ซึ่งจะใช้เป็นส่วนพื้นฐานในการส่งผ่านข้อมูล โดยที่การสื่อสารผ่าน Ethernet จะต้องมีการระบุที่อยู่ของผู้ส่งและผู้รับ หรือ MAC Address (Media Address Control) ซึ่งอาดูโนอีเธอร์เน็ตชิลด์จะแสดงดังรูปที่ 2.4



รูปที่ 2.4 อาดูโน่อีเธอร์เน็ตชิลด์โมดูล

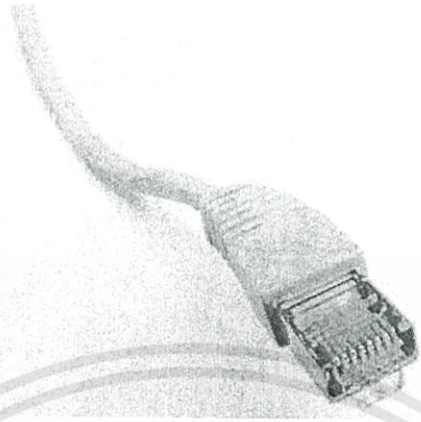
โดยที่เราสามารถนำอีเธอร์เน็ตชิลด์ไปต่อลงบนตัวอาดูโนไมโครคอนโทรลเลอร์บอร์ดโดยตรงได้เลย และรับส่งข้อมูลโดยใช้สาย RJ-45 ในการเชื่อมต่อ ดังรูปที่ 2.5



รูปที่ 2.5 การต่ออีเธอร์เน็ตชิลด์กับอาดูโนไมโครคอนโทรลเลอร์

2.5.1 สาย RJ-45

สาย RJ-45 คือ หัวต่อที่ใช้กับสายสัญญาณเชื่อมเครือข่ายแบบสายคู่ตีเกลียว (สาย UTP) ตัวผู้ มี 2 ชนิด ได้แก่ 1. หัวต่อตัวผู้ RJ-45 (หรือที่เรียกว่ RJ-45 Connecter หรือ RJ-45 Jack Plug) เป็นอุปกรณ์สำหรับใส่ที่ปลายสาย UTP มีลักษณะเป็นพลาสติกสี่เหลี่ยมคล้ายหัวต่อโทรศัพท์ มีช่องสำหรับเสียบสายที่ด้านหลัง ด้านล่างเรียบ ส่วนด้านบนมีตัวล๊อค ถ้าหันหน้าเข้าด้านหน้าของหัวต่อพิน 1 จะอยู่ทางด้านซ้ายมือ ในขณะที่พิน 8 จะอยู่ทางขวามือดังรูปที่ 2.6



รูปที่ 2.6 หัวต่อตัวผู้ RJ-45

2.5.2 หัวต่อตัวเมีย RJ-45 (หรือเรียกว่า RJ-45 Jack Face) [6]

มีลักษณะเป็นเบ้าเสียบสำหรับหัวต่อ RJ-45 ตัวผู้ เมื่อมองจากด้านที่จะนำหัวต่อตัวผู้เสียบ พิน 8 จะอยู่ทางซ้าย ส่วนพิน 1 จะอยู่ทางขวา หัวต่อตัวเมียจะมีลักษณะเป็นกล่องมีช่องสำหรับเสียบหัวต่อ ด้านในกล่องจะมีขั้วซึ่งจะเป็นส่วนที่เชื่อมกับสายนำสัญญาณ ดังรูปที่ 2.7

รูปที่ 2.7 หัวต่อตัวเมีย RJ-45

2.5.3 MAC Address (Media Address Control) [7]

MAC Address คือหมายเลขของ Network Card (LAN, Wireless LAN) ซึ่งหมายเลขจะไม่ซ้ำกัน โดยค่าหมายเลขนั้นจะถูกกำหนดมาจากโรงงานที่ผลิต Network Card โดยรูปแบบของค่า MAC Address จะอยู่ในรูปแบบของเลขฐานสิบหกขนาด 6 ไบต์ เช่น 00-1A-2A-3C-44-55

2.6 TCP/IP (Transmission Control Protocol/Internet Protocol) [8]

TCP/IP เป็นชุดของโปรโตคอลที่ถูกใช้ในการสื่อสารผ่านเครือข่ายอินเทอร์เน็ต โดยมีวัตถุประสงค์เพื่อให้สามารถใช้สื่อสารจากต้นทางข้ามเครือข่ายไปยังปลายทางได้ และสามารถหาเส้นทางที่จะส่งข้อมูลไปได้เองโดยอัตโนมัติ ถึงแม้ว่าในระหว่างทางอาจจะผ่านเครือข่ายที่มีปัญหาโปรโตคอลก็ยังคงหาเส้นทางอื่นในการส่งผ่านข้อมูลไปให้ถึงปลายทางได้

ชุดโปรโตคอลนี้ได้รับการพัฒนามาตั้งแต่ปี 1960 ซึ่งถูกใช้เป็นครั้งแรกในเครือข่าย ARPANET ซึ่งต่อมาได้ขยายการเชื่อมต่อไปทั่วโลกเป็นเครือข่ายอินเตอร์เน็ต ทำให้ TCP/IP เป็นที่ยอมรับอย่างกว้างขวางจนถึงปัจจุบัน

2.6.1 TCP (Transmission Control Protocol)

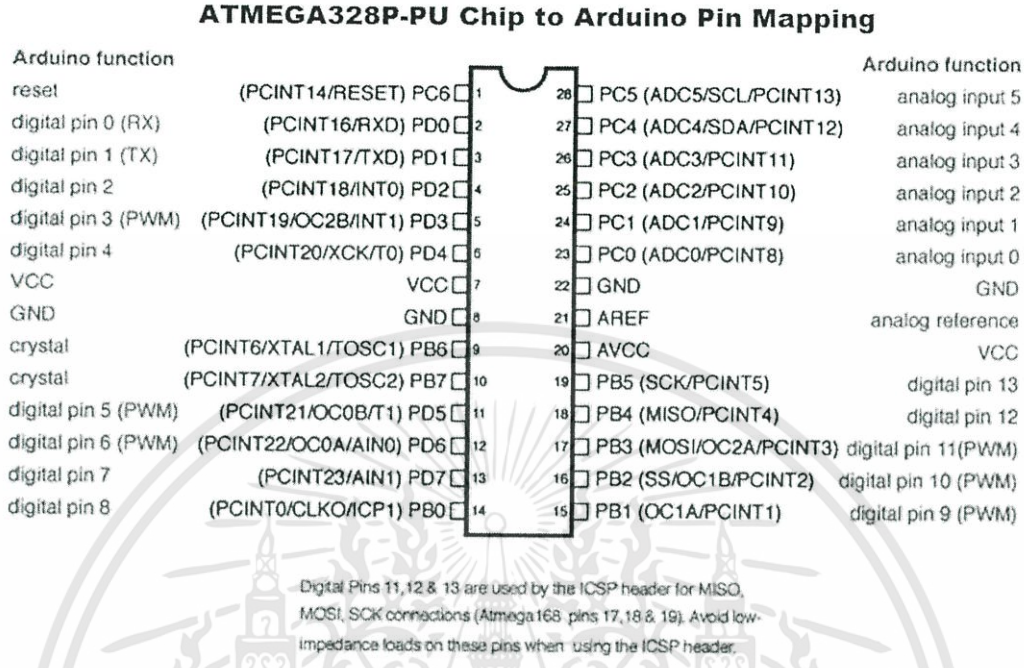
ทำหน้าที่ในการแยกข้อมูลเป็นส่วนๆหรือที่เรียกว่า Package ส่งออกไป ส่วน TCP ปลายทาง ก็จะทำการรวบรวมข้อมูลแต่ละส่วนเข้าด้วยกัน เพื่อนำไปประมวลผลต่อไป โดยระหว่างการรับส่งข้อมูลนั้นก็จะมีการตรวจสอบความถูกต้องของข้อมูลด้วย ถ้าเกิดผิดพลาด TCP ปลายทางก็จะขอไปยัง TCP ต้นทางให้ส่งข้อมูลมาใหม่

2.6.2 IP (Internet Protocol)

ทำหน้าที่ในการจัดส่งข้อมูลจากเครื่องต้นทางไปยังเครื่องปลายทาง โดยอาศัย IP Address ซึ่ง IP Address คือ ระบบการอ้างอิง การมีตัวตนอยู่ของคอมพิวเตอร์ ซึ่งอ้างอิงจากหมายเลขประจำเครื่องคอมพิวเตอร์ ซึ่งประกอบด้วยเลข 4 ชุด มีเครื่องหมายจุดขั้นระหว่างชุด เช่น IP Address 192.168.1.1 เป็นต้น

2.7 ไอซี ATmega328p-pu

ไอซีที่ใช้กับบอร์ดอาดูโน่นั้น ได้เลือกใช้ไมโครคอนโทรลเลอร์ของ Atmel เบอร์ ATmega328p-pu (ATmega48A/48PA/88A/88PA/168A/168PA/328/328P Datasheet) เป็นไมโครคอนโทรลเลอร์ ตระกูล AVR มี 28 ขา โดยหน้าที่และตำแหน่งของแต่ละขาแสดงดังรูปที่ 2.8



รูปที่ 2.8 ตำแหน่งขา และหน้าที่แต่ละขาของไอซี ATmega328p-pu [10]

2.8 รีเลย์ (Relay) [9]

รีเลย์ คือ อุปกรณ์อิเล็กทรอนิกส์ที่ทำหน้าที่ตัด-ต่อวงจรคล้ายกับสวิตช์ โดยใช้หลักการหน้าสัมผัส และการที่จะให้มันทำงาน ต้องจ่ายไฟให้มันตามที่กำหนด เพราะเมื่อจ่ายไฟให้กับตัวรีเลย์ มันจะทำให้หน้าสัมผัสติดกัน กลายเป็นวงจรปิด และตรงข้าม ทันทีที่ไม่ได้จ่ายไฟให้มัน มันก็จะกลายเป็นวงจรเปิด ไฟที่เราใช้ป้อนให้กับตัวรีเลย์ก็จะเป็นไฟที่มาจาก เพาเวอร์ฯ ของเครื่องเรา ดังนั้นทันทีที่เปิดเครื่อง ก็จะทำให้รีเลย์ทำงาน

มีหลักการทำงานคล้ายกับ ขดลวดแม่เหล็กไฟฟ้าหรือโซลินอยด์ (Solenoid) รีเลย์ใช้ในการควบคุมวงจร ไฟฟ้าได้อย่างหลากหลาย รีเลย์เป็นสวิตช์ควบคุมที่ทำงานด้วยไฟฟ้า แบ่งออกตามลักษณะการใช้งานได้เป็น 2 ประเภทคือ

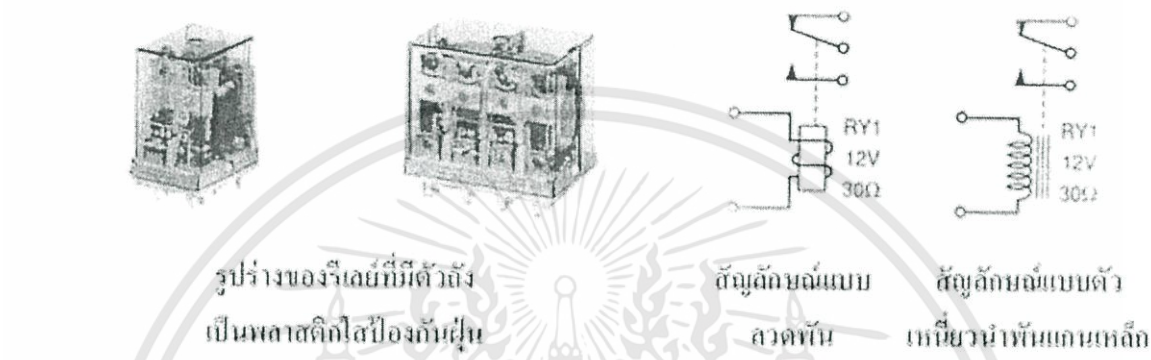
1. รีเลย์กำลัง (power relay) หรือมักเรียกกันว่า คอนแทกเตอร์ (Contactor or Magneticcontactor) ใช้ในการควบคุมไฟฟ้ากำลัง มีขนาดใหญ่กว่ารีเลย์ธรรมดา

2. รีเลย์ควบคุม (control Relay) มีขนาดเล็กกำลังไฟฟ้าต่ำ ใช้ในวงจรควบคุมทั่วไปที่มีกำลังไฟฟ้าไม่มากนัก หรือเพื่อการควบคุมรีเลย์หรือคอนแทกเตอร์ขนาดใหญ่ รีเลย์ควบคุมบางทีเรียกกันง่าย ๆ ว่า "รีเลย์"

ซึ่งในโครงงานนี้ได้ใช้รีเลย์ควบคุมในการควบคุมการเปิด-ปิดของเครื่องใช้ไฟฟ้า

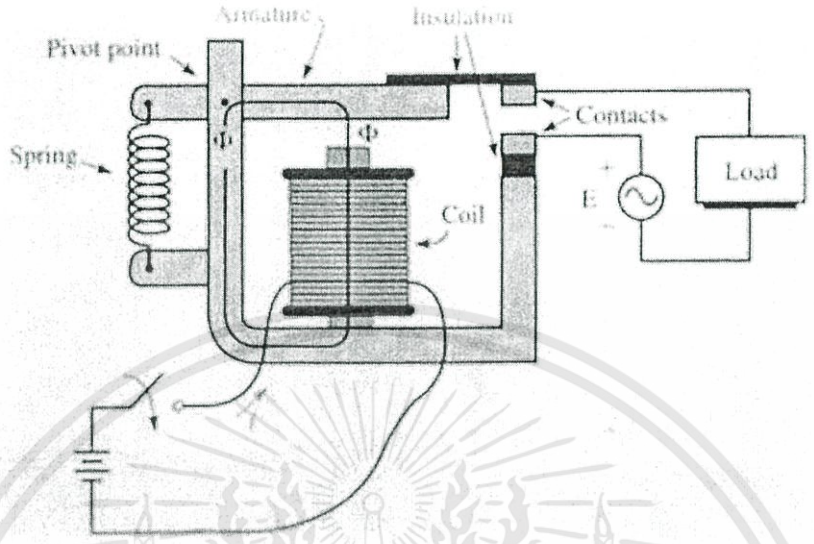
2.8.1 หลักการเบื้องต้น

จากที่กล่าวมาว่า รีเลย์เป็นอุปกรณ์ที่นิยมมาทำเป็นสวิตช์ทางด้านอิเล็กทรอนิกส์ โดยจะต้องป้อนกระแสไฟฟ้าให้ไหลผ่านขดลวดจำนวนหนึ่ง เพื่อนำไปควบคุมกำลังงานสูงๆ ที่ต่ออยู่กับหน้าสัมผัส หรือคอนแทคต์ของรีเลย์ ดังรูปที่ 2.9 ที่แสดงรูงร่างและสัญลักษณ์ของรีเลย์



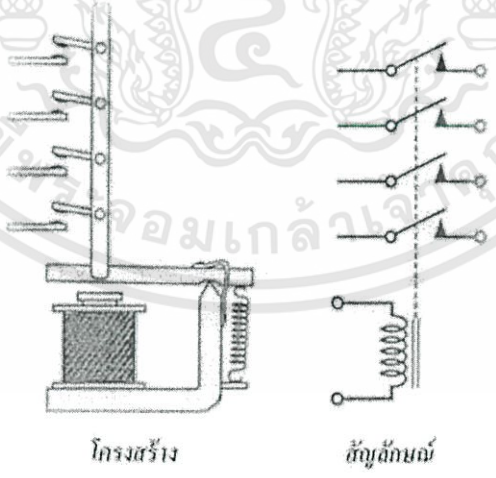
รูปที่ 2.9 รูปร่าง และสัญลักษณ์ของรีเลย์

หลักการทำงานเบื้องต้นของรีเลย์แสดงดังรูปที่ 2.10 การทำงานเริ่มจากปิดสวิตช์ เพื่อป้อนกระแสให้กับขดลวด (Coil) โดยทั่วไปจะเป็นขดลวดพันรอบแกนเหล็ก ทำให้เกิดสนามแม่เหล็กไปดูดเหล็กอ่อนที่เรียกว่าอาร์เมเจอร์ (Armature) ให้ต่ำลงมา ที่ปลายของอาร์เมเจอร์ด้านหนึ่งมักยึดติดกับสปริง (Spring) และปลายอีกด้านหนึ่งยึดติดกับหน้าสัมผัส (Contacts) การเคลื่อนที่อาร์เมเจอร์ จึงเป็นการควบคุมการเคลื่อนที่ของหน้าสัมผัส ให้แยกจาก หรือแตะกับหน้าสัมผัสอีกอันหนึ่งซึ่งยึดติดอยู่กับที่ เมื่อเปิดสวิตช์อาร์เมเจอร์ ก็จะกลับสู่ตำแหน่งเดิม เราสามารถนำหลักการนี้ไปควบคุมโหลด (Load) หรือวงจรอิเล็กทรอนิกส์ต่าง ๆ ได้ตามต้องการ



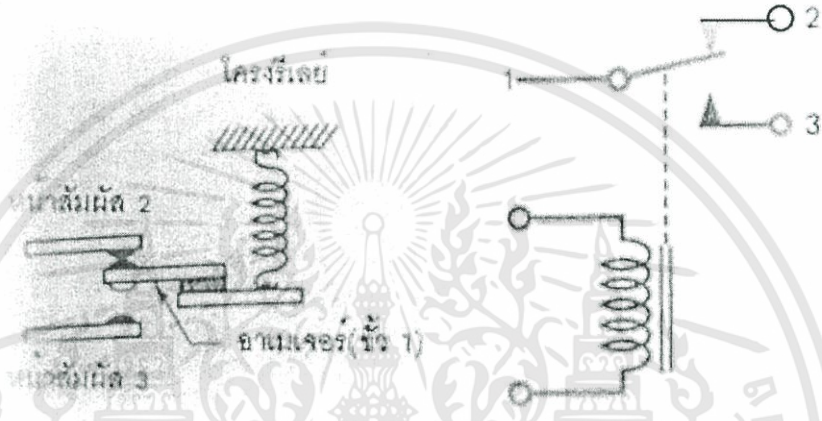
รูปที่ 2.10 หลักการทำงานเบื้องต้นของรีเลย์

รูปที่ 2.10 แสดงรีเลย์ที่มีหน้าสัมผัสเพียงชุดเดียว ปัจจุบันรีเลย์ที่มีขดลวดชุดเดียว สามารถควบคุมหน้าสัมผัสได้หลายชุด ดังรูปที่ 2.11 อาร์เมเจอร์อันเดียวถูกยึดอยู่กับหน้าสัมผัสที่เคลื่อนที่ได้ 4 ชุด ดังนั้นรีเลย์ตัวนี้ จึงสามารถควบคุมการแตะ หรือจากกันของหน้าสัมผัสได้ถึง 4 ชุด



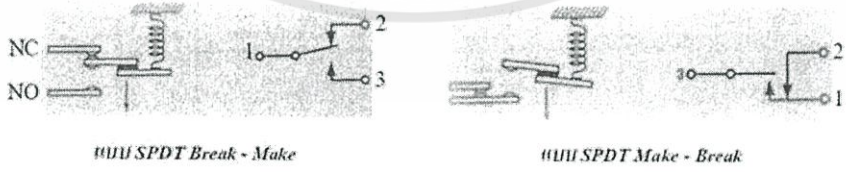
รูปที่ 2.11 โครงสร้างและสัญลักษณ์ของชุดหน้าสัมผัสแบบ 4PST

แต่ละหน้าสัมผัสที่เคลื่อนที่ได้มีชื่อเรียกว่าขั้ว (Pole) รีเลย์ในรูปที่ 2.11 มี 4 ขั้ว จึงเรียกหน้าสัมผัสแบบนี้ว่าเป็นแบบ 4PST (Four Pole Single Throw) ถ้าแต่ละขั้วที่เคลื่อนที่แล้วแยกจากหน้าสัมผัสอันหนึ่งไปแตะกับหน้าสัมผัสอีกอันหนึ่ง เหมือนกับสวิตช์โยก โดยเป็นการเลือกหน้าสัมผัส ที่ขนาบอยู่ทั้งสองด้านดังรูปที่ 2.12 หน้าสัมผัสแบบนี้มีชื่อว่า SPDT (Single Pole Double Throw)



รูปที่ 2.12 หน้าสัมผัสแบบ SPDT

ในกรณีที่ไม่มีการป้อนกระแสไฟฟ้าเข้าขดลวดของรีเลย์ สภาวะ NO (Normally Open) คือสภาวะปกติหน้าสัมผัสกับขั้วแยกจากกัน ถ้าต้องการให้สัมผัสกันจะต้องป้อนกระแสไฟฟ้าเข้าขดลวด ส่วนสภาวะ NC (Normally Closed) คือสภาวะปกติหน้าสัมผัสกับขั้วสัมผัสกัน ถ้าต้องการให้แยกกันจะต้องป้อนกระแสไฟฟ้าเข้าขดลวด นอกจากนี้ยังมีแบบแยกก่อนแล้วสัมผัส (Break-Make) หมายถึงหน้าสัมผัสระหว่าง 1 และ 2 จะแยกจากกันก่อนที่หน้าสัมผัส 1 และ 3 จะสัมผัสกัน แต่ถ้าหากตรงข้ามกันคือ หน้าสัมผัส 1 และ 2 จะสัมผัสกัน และจะไม่แยกจากกัน จนกว่าหน้าสัมผัส 1 และ 3 จะสัมผัสกัน (Make-Break) ดังรูปที่ 2.13



รูปที่ 2.13 หน้าสัมผัสแบบ SPDT แบบ Break – Make และ Make - Brea

2.8.2 การแบ่งชนิดของรีเลย์สามารถแบ่งได้ 11 แบบ คือ

ชนิดของรีเลย์แบ่งตามลักษณะของคอยล์ หรือ แบ่งตามลักษณะการใช้งาน (Application) ได้แก่รีเลย์ดังต่อไปนี้

2.8.2.1 รีเลย์กระแส (Current relay) คือ รีเลย์ที่ทำงานโดยใช้กระแสมีทั้งชนิดกระแสขาด (Under- current) และกระแสเกิน (Over current)

2.8.2.2 รีเลย์แรงดัน (Voltage relay) คือ รีเลย์ ที่ทำงานโดยใช้แรงดันมีทั้งชนิดแรงดันขาด (Under-voltage) และ แรงดันเกิน (Over voltage)

2.8.2.3 รีเลย์ช่วย (Auxiliary relay) คือ รีเลย์ที่เวลาใช้งานจะต้องประกอบเข้ากับรีเลย์ชนิดอื่น จึงจะทำงานได้

2.8.2.4 รีเลย์กำลัง (Power relay) คือ รีเลย์ที่รวมเอาคุณสมบัติของรีเลย์กระแส และรีเลย์แรงดันเข้าด้วยกัน

2.8.2.5 รีเลย์เวลา (Time relay) คือ รีเลย์ที่ทำงานโดยมีเวลาเข้ามาเกี่ยวข้องด้วย ซึ่งมีอยู่ด้วยกัน 4 แบบ คือ

1) รีเลย์กระแสเกินชนิดเวลาผกผันกับกระแส (Inverse time over current relay) คือ รีเลย์ ที่มีเวลาทำงานเป็นส่วนกลับกับกระแส

2) รีเลย์กระแสเกินชนิดทำงานทันที (Instantaneous over current relay) คือรีเลย์ที่ทำงานทันทีทันใดเมื่อมีกระแสไหลผ่านเกินกว่าที่กำหนดที่ตั้งไว้

3) รีเลย์แบบดิฟฟินิตไทม์แล็ก (Definite time lag relay) คือ รีเลย์ ที่มีเวลาการทำงานไม่ขึ้นอยู่กับความมากน้อยของกระแสหรือค่าไฟฟ้าอื่นๆ ที่ทำให้เกิดงานขึ้น

4) รีเลย์แบบอินเวอสดิฟฟินิตมินิมั่มไทม์แล็ก (Inverse definite time lag relay) คือ รีเลย์ ที่ทำงานโดยรวมเอาคุณสมบัติของเวลาผกผันกับกระแส (Inverse time) และ แบบดิฟฟินิตไทม์แล็ก (Definite time lag relay) เข้าด้วยกัน

2.8.2.6 รีเลย์กระแสต่าง (Differential relay) คือ รีเลย์ที่ทำงานโดยอาศัยผลต่างของกระแส

2.8.2.7 รีเลย์มีทิศ (Directional relay) คือรีเลย์ที่ทำงานเมื่อมีกระแสไหลผิดทิศทาง มีแบบรีเลย์กำลังมีทิศ (Directional power relay) และรีเลย์กระแสมีทิศ (Directional current relay)

2.8.2.8 รีเลย์ระยะทาง (Distance relay) คือ รีเลย์ระยะทางมีแบบต่างๆ ดังนี้

1) รีแอกแตนซ์รีเลย์ (Reactance relay)
2) อิมพีแดนซ์รีเลย์ (Impedance relay)
3) โมห์รีเลย์ (Mho relay)
4) โอห์มรีเลย์ (Ohm relay)

5) โพลาไรซ์โมห์รีเลย์ (Polaized mho relay)

6) ออฟเซทโมห์รีเลย์ (Off set mho relay)

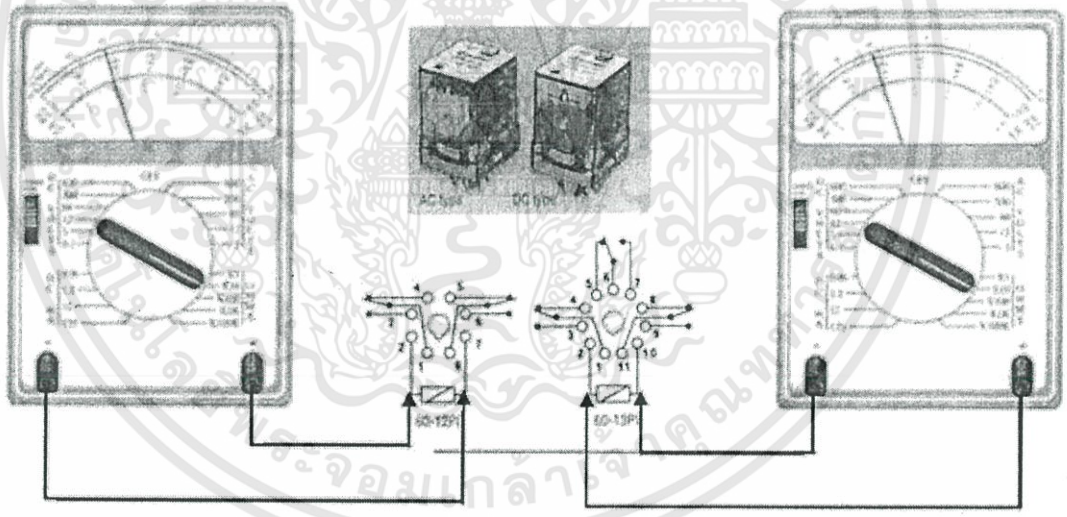
2.8.2.9 รีเลย์อุณหภูมิ (Temperature relay) คือ รีเลย์ที่ทำงานตามอุณหภูมิที่ตั้งไว้

2.8.2.10 รีเลย์ความถี่ (Frequency relay) คือ รีเลย์ที่ทำงานเมื่อความถี่ของระบบต่ำกว่าหรือมากกว่าที่ตั้งไว้

2.8.2.11 บูคโฮลซ์รีเลย์ (Buchholz's relay) คือรีเลย์ที่ทำงานด้วยก๊าซ ใช้กับหม้อแปลงที่แช่อยู่ในน้ำมันเมื่อเกิด ฟอลต์ ขึ้นภายในหม้อแปลง จะทำให้น้ำมันแตกตัวและเกิดก๊าซขึ้นภายในไปดันหน้าสัมผัส ให้รีเลย์ทำงาน

2.8.3 การตรวจสอบรีเลย์

การตรวจสอบรีเลย์ว่าอยู่ในสภาพดี หรือชำรุดนั้น สามารถกระทำได้โดยใช้มัลติมิเตอร์ตั้งย่านวัดโอห์ม แล้วใช้สายวัดทั้งสองสัมผัสที่ขั้วขดลวด (Coil) ของรีเลย์ทั้งสองขั้ว ถ้าเข็มมิเตอร์เบี่ยงเบนแสดงค่าความต้านทาน แสดงว่ารีเลย์อยู่ในสภาพที่ใช้งานได้ แต่ถ้าหากเข็มไม่ขึ้นแสดงว่าไม่สามารถใช้งานได้ มัลติมิเตอร์แสดงดังรูป 2.14



รูปที่ 2.14 การใช้มัลติมิเตอร์ตรวจสภาพรีเลย์

2.9 เซ็นเซอร์ความชื้น และอุณหภูมิ DHT11 (Humitdity and Temperature Sensor) [10]

โมดูล DHT11 ใช้ในการวัดอุณหภูมิ กับความชื้นในอากาศ โดยมีรายละเอียดคร่าวๆ ดังนี้

2.9.1 ย่านวัดความชื้น 20-90% RH โดยมีค่าความแม่นยำ +- 5% RH ความละเอียดในการวัด 1 % แสดงผลแบบ 8 บิต

2.9.2 ย่านวัดอุณหภูมิ 0 - 50 องศาเซลเซียส โดยมีค่าความแม่นยำ +- 2 องศาเซลเซียส ความละเอียดในการวัด 1 องศาเซลเซียส แสดงผลแบบ 8 บิต

2.9.3 มี PIN 4 ขา

2.9.4 กินกระแส 0.5 - 2.5 mA (ขณะทำการวัดค่า) ที่ระดับแรงดัน 3 - 5.5 VDC

2.9.5 อ่านค่าสัญญาณ (Sample Rate) ทุก 1 วินาที

รายละเอียดขา และรูปร่างลักษณะของอุปกรณ์แสดงดังรูปที่ 2.15

Pin	Name	Description
1	VDD	Power supply 3 - 5.5 V DC
2	DATA	Serial data output
3	NC	Not connected
4	GND	GND

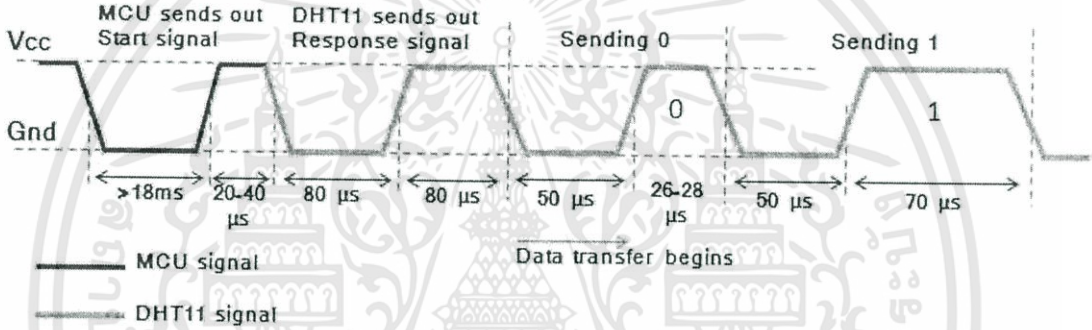
รูปที่ 2.15 รายละเอียดขาของเซ็นเซอร์ DHT11

ข้อดีของ DHT11 คือสามารถใช้งานได้กับ Arduino หลายรุ่น ทั้ง Arduino Due หรือ UNO และ Mega/Mega ADK โดยมีวิธีการส่งข้อมูลของ DHT11 กับไมโครคอนโทรลเลอร์ ด้วยวิธีการสื่อสารอนุกรมสองทางโดยใช้สายเส้นเดียว (Single-wire Two-way Serial interface) ซึ่งหมายความว่า การสื่อสารแบบนี้จะใช้สายสื่อสารเพียงเส้นเดียว และส่งข้อมูลได้ทั้งจากไมโครคอนโทรลเลอร์ไปที่ตัว DHT11 และในทางกลับกันก็ได้

โดยการส่งข้อมูลของ DHT11 จะใช้วิธี Single-wire Two-way Serial interface กับไมโครคอนโทรลเลอร์ นั่นก็คือ การสื่อสารอนุกรมสองทางโดยใช้สายเส้นเดียว หรือหมายความว่า การสื่อสารแบบนี้จะใช้สายสื่อสารเพียงเส้นเดียวและส่งข้อมูลได้ทั้งจากไมโครคอนโทรลเลอร์ ไปที่ตัว DHT11 และในทางกลับกันก็ได้

ระดับแรงดันของสัญญาณในสายส่งข้อมูล คือแรงดันระดับ "สูง" และจะมีแรงดันในระดับต่ำเมื่อมีอุปกรณ์ดึงสัญญาณลงในระดับ "ต่ำ" ดังนั้นหากวัดระดับสัญญาณได้เป็น "สูง" ตลอดเวลา ก็หมายความว่าอุปกรณ์ของเราอาจจะผิดปกติ

ในการสื่อสารโดยใช้สายเส้นเดียวนั้น จำเป็นต้องใช้โปรโตคอลที่ตกลงกันไว้ระหว่างตัวไมโครคอนโทรลเลอร์และอุปกรณ์ที่ต้องการสื่อสารด้วย อันดับแรกอาดูโนจะส่ง Start signal ที่เป็นแรงดันไฟฟ้าระดับต่ำอย่างน้อย 18 ไมโครวินาที ไปที่ DHT11 เพื่อให้เข้าใจว่าจะเริ่มสั่งข้อมูล แล้วรอไป 20-40 ไมโครวินาทีเพื่อรอการตอบกลับ ต่อไปเพื่อให้รู้ว่าอีกฝั่งพร้อมที่จะส่งข้อมูลกลับ ก็จะส่งแรงดันระดับต่ำกลับไปบ้าง การส่งแรงดันกลับไปจะนาน 80 ไมโครวินาที จากนั้นจะรออีก 80 ไมโครวินาที ก่อนที่จะส่งข้อมูลบิตแรก ดังรูปที่ 2.16



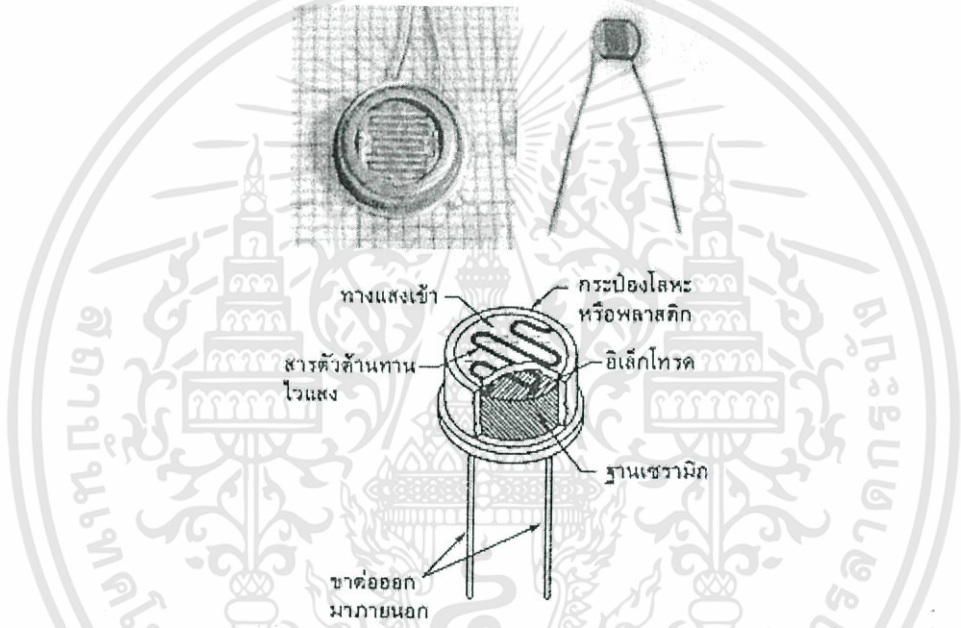
รูปที่ 2.16 บิตการส่งข้อมูลของ DHT11

คราวนี้จากรูปที่ 2.16 การจะส่งบิต "0" กับ บิต "1" จะมีความแตกต่างกัน สำหรับการส่งบิตเป็น "0" จะดึงระดับแรงดันลงต่ำนาน 50 ไมโครวินาที และปล่อยเป็นระดับ "สูง" นาน 26-28 ไมโครวินาที (ดูช่วง Sending 0) แต่ถ้าเป็นการส่งข้อมูลเป็น "1" ตัวส่งจะดึงสายสัญญาณลงระดับต่ำ 50 ไมโครวินาที และปล่อยให้เป็นระดับสูงนาน 70 ไมโครวินาที (ดูช่วง Sending 1) และส่งมาจนครบข้อมูลหนึ่งชุดในที่สุด

ในแต่ละชุดของข้อมูลที่ส่งมาจาก DH11 จะยาว 40 บิต และใช้เวลาส่งประมาณ 4 มิลลิวินาที ซึ่งใน 40 บิตที่ส่งมาจะประกอบด้วย " 8bit integral RH data + 8bit decimal RH data + 8bit integral T data + 8bit decimal T data + 8bit check sum"

2.10 แอลดีอาร์ : ความต้านทานชนิดไวต่อแสง (LDR : Light Dependent Resistor) [11]

แอลดีอาร์ คือ ความต้านทานชนิดที่ไวต่อแสง หรือกล่าวได้ว่า คือตัวต้านทานนี้สามารถเปลี่ยนสภาพทางความนำไฟฟ้าได้ เมื่อมีแสงมาตกกระทบ บางครั้งเรียกว่าโฟโต้สีซิสเตอร์ (Photo Resistor) หรือ โฟโต้คอนดัคเตอร์ (Photo Conductor) เป็นตัวต้านทานที่ทำมาจากสารกึ่งตัวนำ (Semiconductor) เอามาฉาบลงบนแผ่นเซรามิก ที่ใช้เป็นฐานรอง แล้วต่อขาจากสารที่ฉาบไว้ออกมา ดังรูปที่ 2.17



รูปที่ 2.17 โครงสร้างแอลดีอาร์

ซึ่งแสงที่ตกกระทบนั้น ต้องเป็นแสงในช่วงความยาวคลื่นประมาณ 4,000 อังสตรอม (1 อังสตรอมเท่ากับ 10-10 เมตร) ถึงประมาณ 10,000 อังสตรอมเท่านั้น ซึ่งสายตาที่คนเราเห็น ก็อยู่ที่ประมาณ 4,000 – 7,000 อังสตรอม นั่นคืออยู่ในแสงอาทิตย์ แสงจากหลอดไฟแบบไส้ และจากหลอดฟลูออเรสเซนต์ด้วย

2.11 ไดโอด (Diode)

ไดโอด ถือเป็นอิเล็กทรอนิกส์ชนิดหนึ่ง ที่จำกัดทิศทางการไหลของประจุไฟฟ้า มันจะยอมให้กระแสไฟฟ้าไหลในทิศทางเดียว และกั้นการไหลในทิศทางตรงกันข้าม ดังนั้นจึงอาจถือว่าไดโอดเป็นวาล์วตรวจสอบแบบอิเล็กทรอนิกส์อย่างหนึ่ง ซึ่งนับเป็นประโยชน์อย่างมากในวงจรอิเล็กทรอนิกส์ เช่น ใช้เป็นเรียงกระแสไฟฟ้าในวงจรภาคจ่ายไฟ เป็นต้น

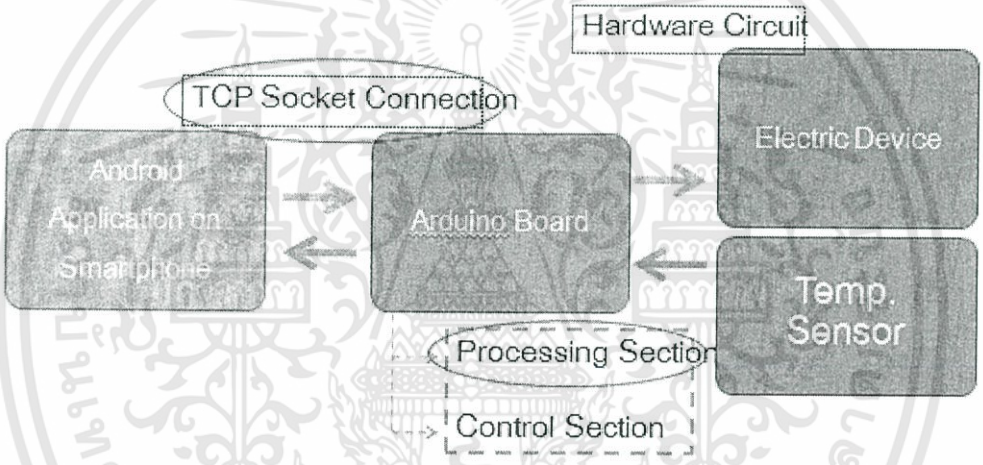
ไดโอดเป็นอุปกรณ์ที่ทำจากสารกึ่งตัวนำ p-n เช่น ซิลิกอน หรือ เจอร์เมเนียมสามารถควบคุมให้กระแสไฟฟ้าจากภายนอกไหลผ่านตัวมันได้ทิศทางเดียว ไดโอดประกอบด้วยขั้ว 2 ขั้ว คือ แอโนด (Anode; A) ซึ่งต่ออยู่กับสารกึ่งตัวนำชนิด p และ แคโทด (Cathode; K) ซึ่งต่ออยู่กับสารกึ่งตัวนำชนิด n โดยขั้วเวลาต่อสามารถสังเกตได้จากแถบสีขาวที่คาดรอบตัวไดโอดเอง ดังรูปที่ 2.18

รูปที่ 2.18 ลักษณะของตัวไดโอด

บทที่ 3

การออกแบบและการจัดทำปริญญานิพนธ์

ระบบควบคุมอุปกรณ์เครื่องใช้ไฟฟ้าภายในบ้านระยะไกลนี้ มีพื้นฐานการออกแบบและจัดทำจากความสะดวกสบายในการใช้งานเป็นหลัก โดยเฉพาะอย่างยิ่งจากแอพพลิเคชั่น (Application) บนแอนดรอยด์สมาร์ทโฟน (Android Smartphone) เพราะฉะนั้นจึงเน้นหน้าตาการใช้งานส่วนติดต่อของผู้ใช้ (User Interface) ที่เข้าใจง่าย ควบคุมสะดวก ไม่ยุ่งยาก และซับซ้อนระบบทำงานได้ผ่านทางเครือข่ายอินเทอร์เน็ต ใช้ไมโครคอนโทรลเลอร์ในการควบคุมการเปิด-ปิดของอุปกรณ์ไฟฟ้า โดยผ่านสายแลน (RJ45) โดยมีบล็อกไดอะแกรมรวมของระบบดังรูปที่ 3.1



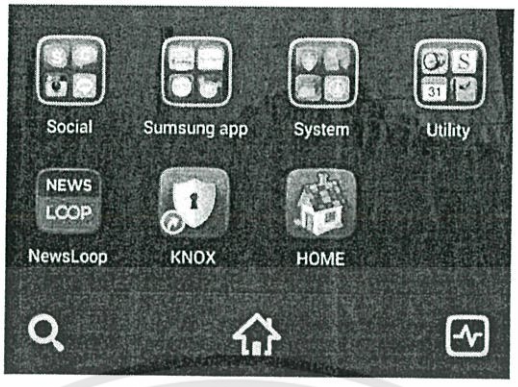
รูปที่ 3.1 บล็อกไดอะแกรมรวมของระบบ

3.1 การออกแบบ

3.1.1 ส่วนแอพพลิเคชั่นบนแอนดรอยด์สมาร์ทโฟน

3.1.1.1 การออกแบบส่วนติดต่อผู้ใช้

1) การออกแบบในส่วนไอคอน (Icon) เพื่อใช้ในการเข้าสู่แอพพลิเคชั่น ในที่นี้ได้ตั้งชื่อแอพพลิเคชั่นว่า “Home” ดังรูปที่ 3.2



รูปที่ 3.2 ไอคอนแอพพลิเคชั่น Home

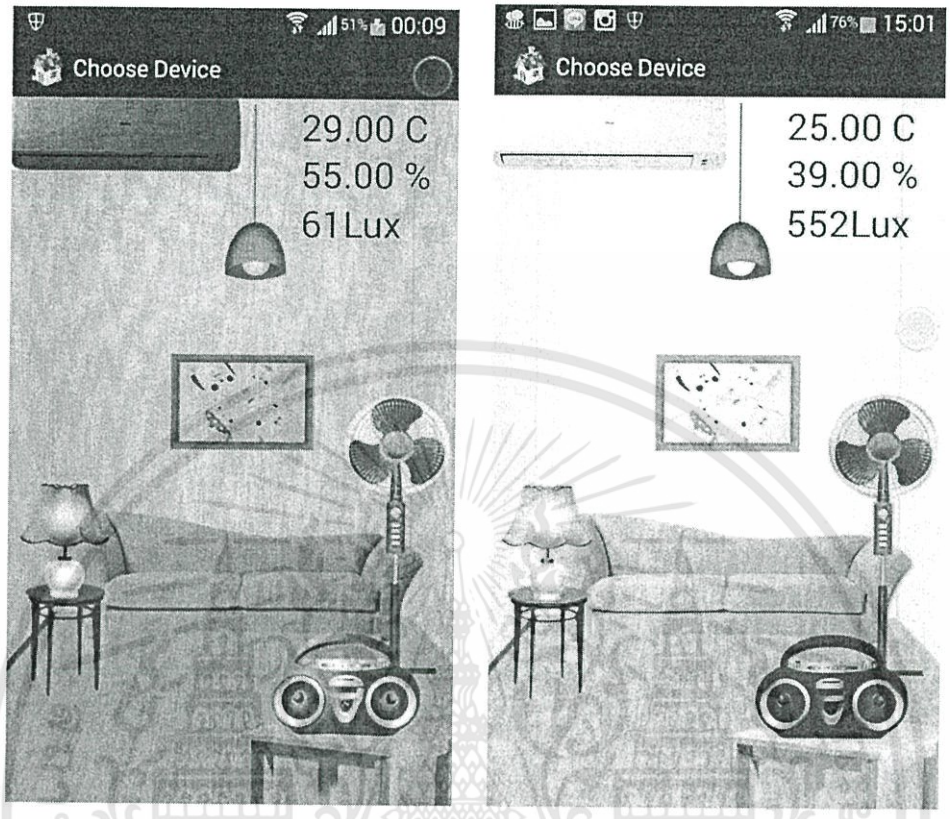
2) การออกแบบในส่วนแบบฟอร์มการเข้าสู่ระบบของแอพพลิเคชั่น Home ประกอบไปด้วยช่องสำหรับใส่ชื่อผู้ใช้ (Username) และรหัสผ่านผู้ใช้ (Password) และปุ่มสำหรับกดเข้าสู่ระบบ (Log in) ดังรูปที่ 3.3

รูปที่ 3.3 หน้าต่างการเข้าสู่ระบบ

ในกรณีที่ผู้ใช้ใส่ชื่อผู้ใช้ หรือรหัสผ่านผู้ใช้ไม่ถูกต้อง จะมีการแจ้งเตือนว่า invalid login เพื่อแจ้งเตือนให้ผู้ใช้ใส่ชื่อผู้ใช้ หรือรหัสผ่านผู้ใช้อีกครั้ง ดังรูปที่ 3.4

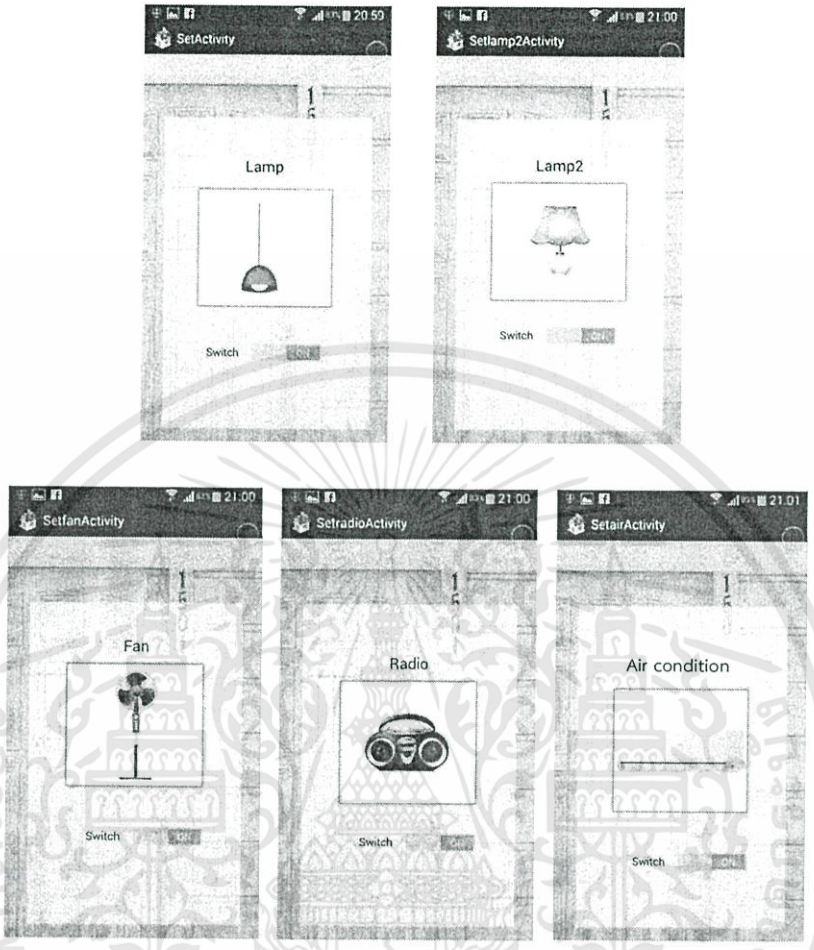
รูปที่ 3.4 หน้าต่างการเข้าสู่ระบบ กรณีที่ผู้ใช้ใส่ชื่อผู้ใช้ หรือรหัสผ่านผู้ใช้ไม่ถูกต้อง

3) การออกแบบหน้าต่างสำหรับเลือกอุปกรณ์ไฟฟ้าที่ต้องการควบคุม สามารถแสดงสถานะของอุปกรณ์ต่างๆ ว่าแต่ละอุปกรณ์อยู่ในสถานะใดๆ โดยการเปลี่ยนสีของอุปกรณ์ชิ้นนั้นๆ แสดงอุณหภูมิในขณะนั้นๆในหน่วยองศาเซลเซียส แสดงความชื้นในหน่วยเปอร์เซ็นต์ และแสดงผลความเข้มแสงผ่านการเปลี่ยนสีผ่านพื้นหลังของหน้าจอและตัวเลขตามความเข้มแสงที่เปลี่ยนแปลงไป ตัวอย่างดังรูปที่ 3.5



รูปที่ 3.5 หน้าต่างสำหรับเลือกอุปกรณ์ไฟฟ้าขณะแสดงสถานะนั้นๆ

4) การออกแบบหน้าต่างสำหรับควบคุมการเปิด-ปิดอุปกรณ์ไฟฟ้า จะประกอบไปด้วยรูปอุปกรณ์ไฟฟ้านั้นๆ และปุ่มสำหรับเปิด-ปิด การออกแบบหน้าต่างสำหรับควบคุมการเปิด-ปิด อุปกรณ์ไฟฟ้า แสดงดังรูปที่ 3.6

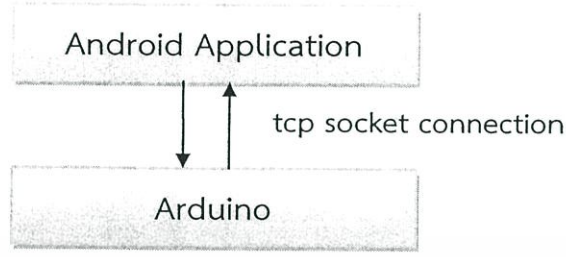


รูปที่ 3.6 หน้าต่างสำหรับควบคุมการเปิด-ปิด อุปกรณ์ไฟฟ้า

3.1.1.2 การออกแบบส่วนติดต่อกับเซิร์ฟเวอร์ภาพรวม

1) แนวคิดการทำงานของระบบ

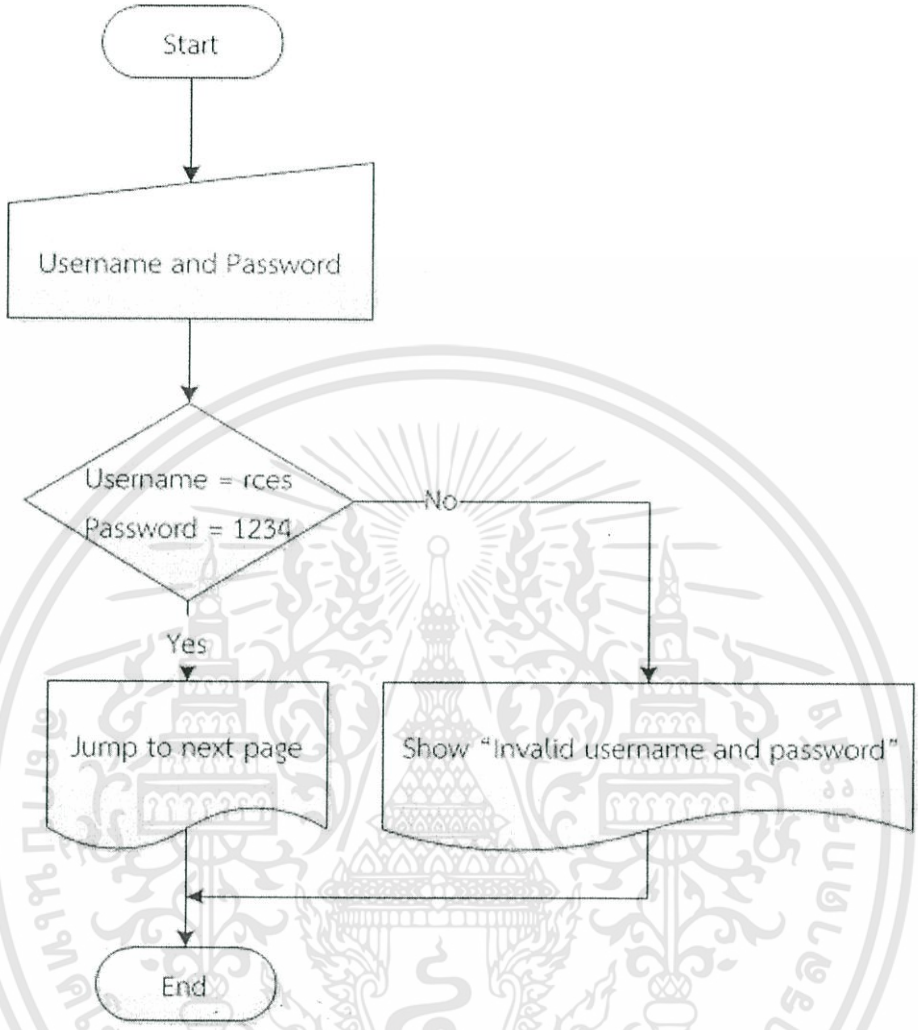
แนวคิดการทำงานของแอพพลิเคชั่นจริง จะเชื่อมต่อไปยังอาดูโน่ แสดงดังรูปที่ 3.7



รูปที่ 3.7 แนวคิดการทำงานของระบบ

2) หลักการเข้าสู่ระบบของแอพพลิเคชั่น

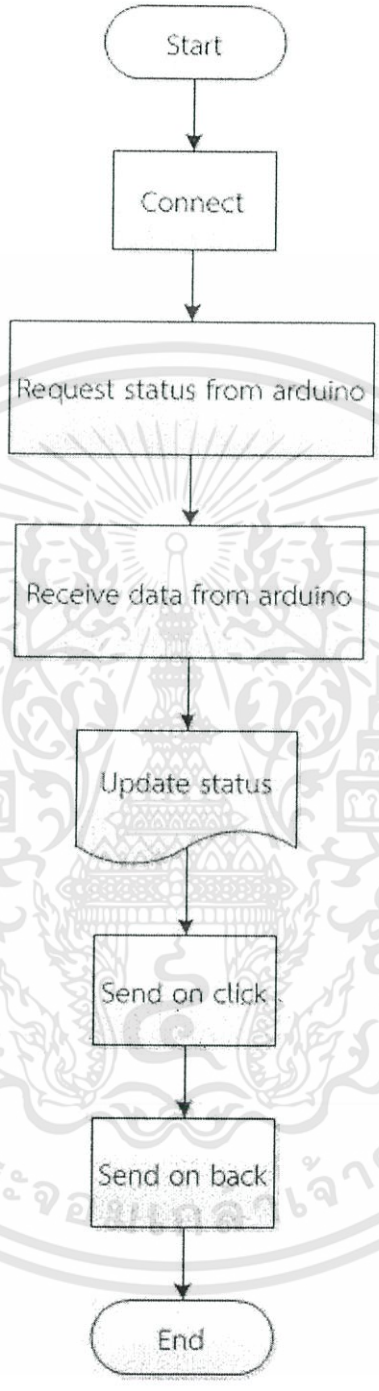
ระบบการทำงานหน้าเข้าสู่ระบบของแอพพลิเคชั่น เมื่อผู้ใช้งานใส่ชื่อผู้ใช้งานและใส่รหัสผ่านอย่างถูกต้องจะสามารถเข้าสู่ระบบเพื่อทำการควบคุมอุปกรณ์ที่ต้องการได้ มีหลักการทำงาน ดังรูปที่ 3.8



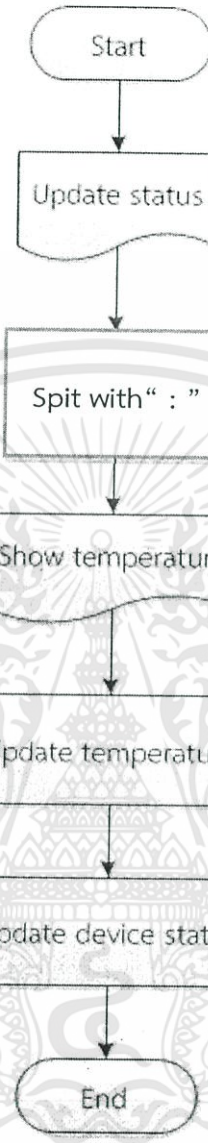
รูปที่ 3.8 หลักการทำงานการเข้าสู่ระบบของแอพพลิเคชั่น

3) หลักการหน้าต่างเลือกประเภทอุปกรณ์ที่ต้องการควบคุม

ระบบการทำงานการติดต่อระหว่างแอพพลิเคชั่นกันอาดูโน สามารถเลือกอุปกรณ์เพื่อเข้าไปควบคุมอุปกรณ์ที่ต้องการควบคุม มีหลักการดังรูปที่ 3.9 มีหลักการการปรับเปลี่ยนสถานะตามอุปกรณ์จริง เพื่อแสดงผลว่าอุปกรณ์ใดกำลังเปิดหรือปิดอยู่ และแสดงตัวเลขอุณหภูมิในขณะนั้นๆ มีหลักการดังรูปที่ 3.10



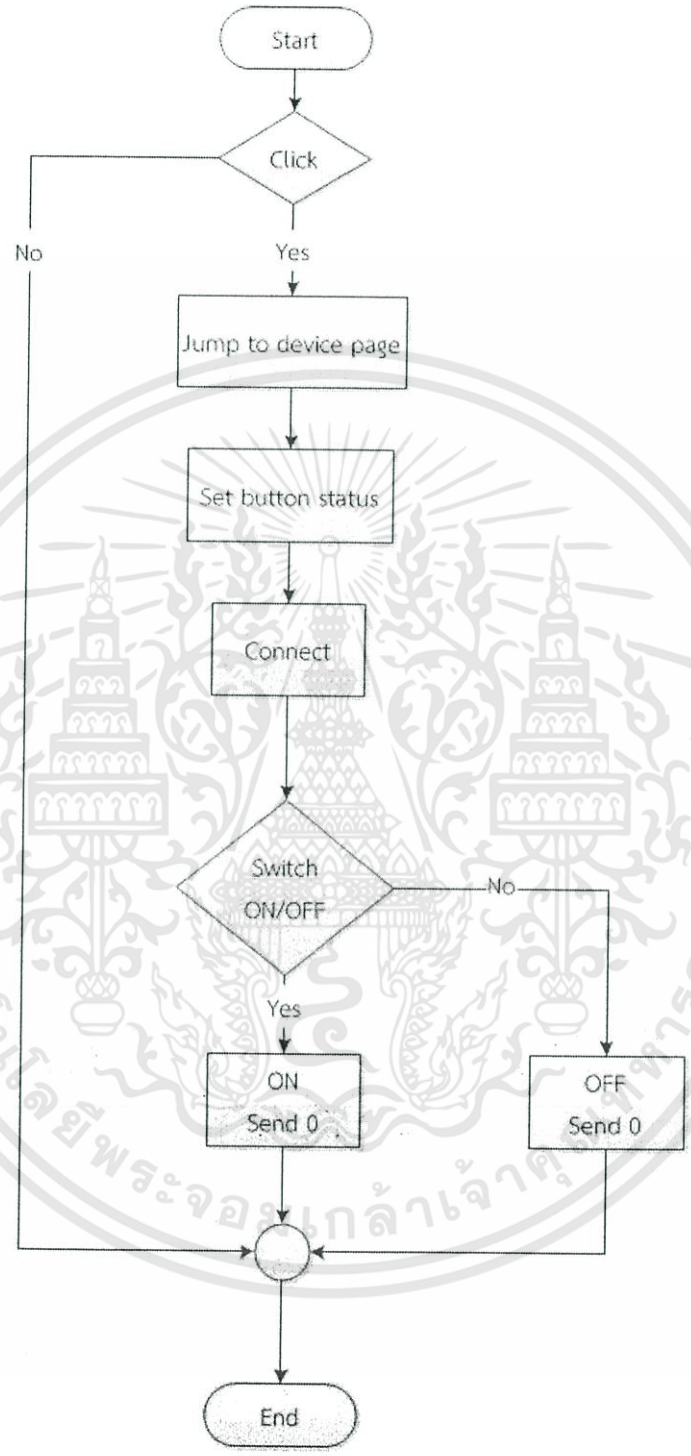
รูปที่ 3.9 หลักการทำงานหน้าต่างสำหรับเลือกประเภทอุปกรณ์ที่ต้องการควบคุม



รูปที่ 3.10 หลักการการปรับเปลี่ยนสถานะตามอุปกรณ์จริง

4) หลักการทำงานหน้าต่างสำหรับควบคุมอุปกรณ์

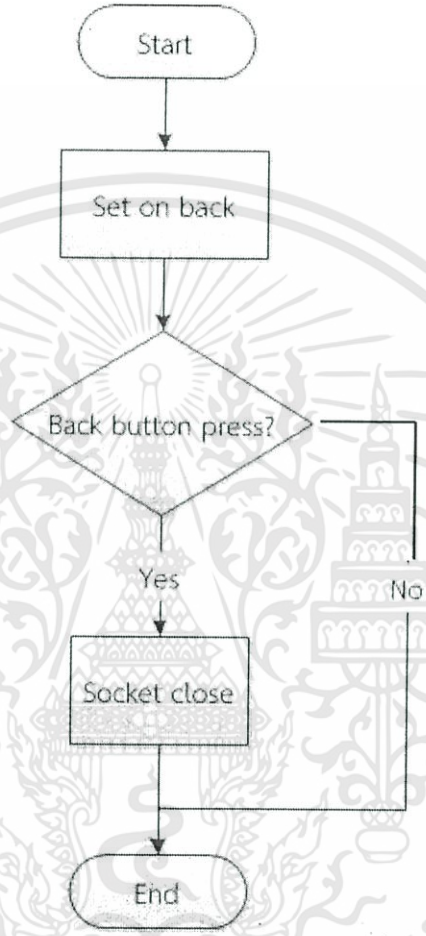
ระบบการทำงานการติดต่อระหว่างแอพพลิเคชั่นกันอาดูโน่ สามารถสั่งการเปิดหรือปิดอุปกรณ์ และแสดงสถานะของอุปกรณ์นั้นๆ มีหลักการดังรูป 3.11



รูปที่ 3.11 หลักการทำงานหน้าต่างสำหรับควบคุมอุปกรณ์

5) หลักการทำงานเมื่อกดปุ่มย้อนกลับ

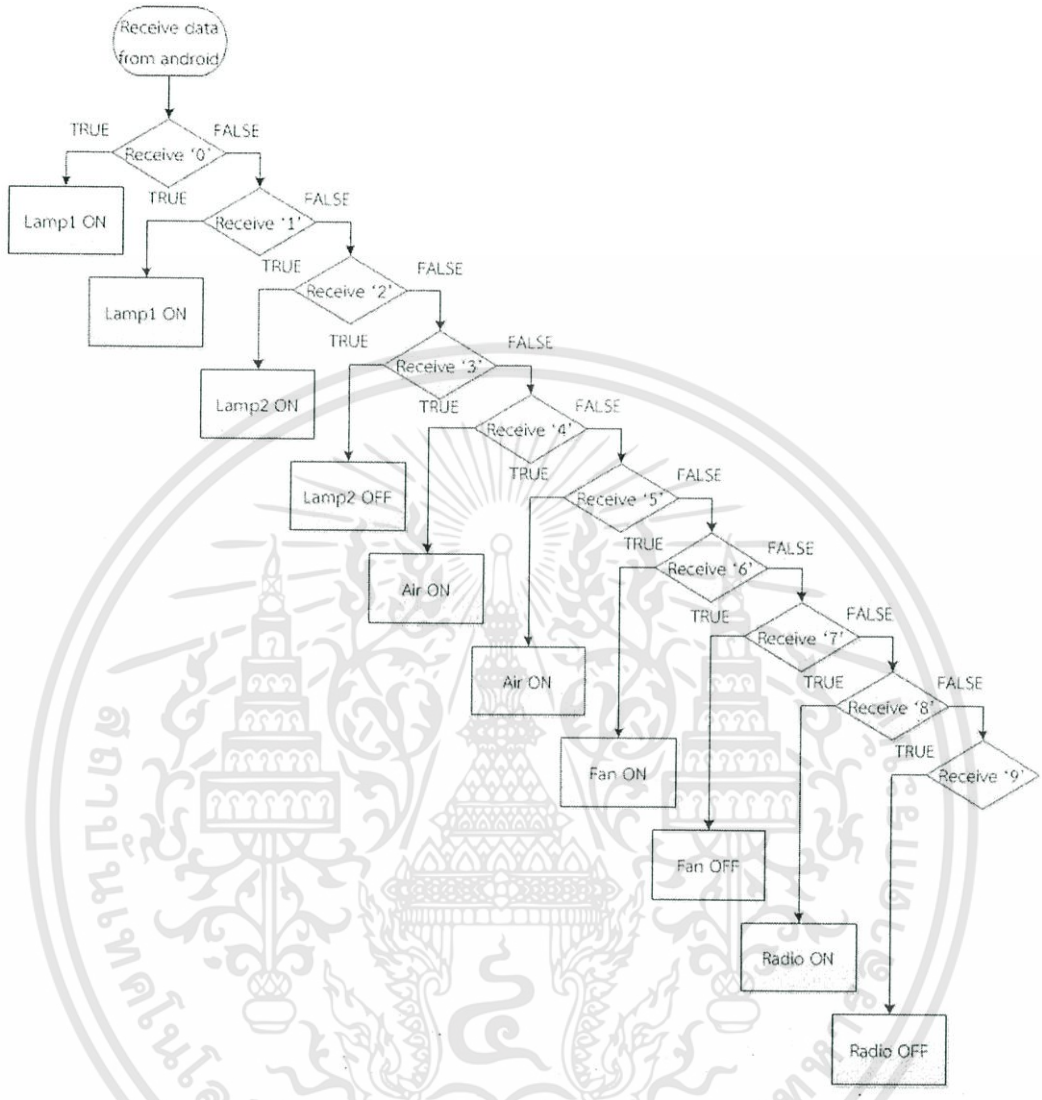
หลักการทำงานเมื่อกดปุ่มย้อนกลับ มีการใช้คำสั่งเพื่อยกเลอก การเชื่อมต่อ หลักการทำงานแสดงดังรูปที่ 3.12



รูปที่ 3.12 หลักการทำงานเมื่อกดปุ่มย้อนกลับ

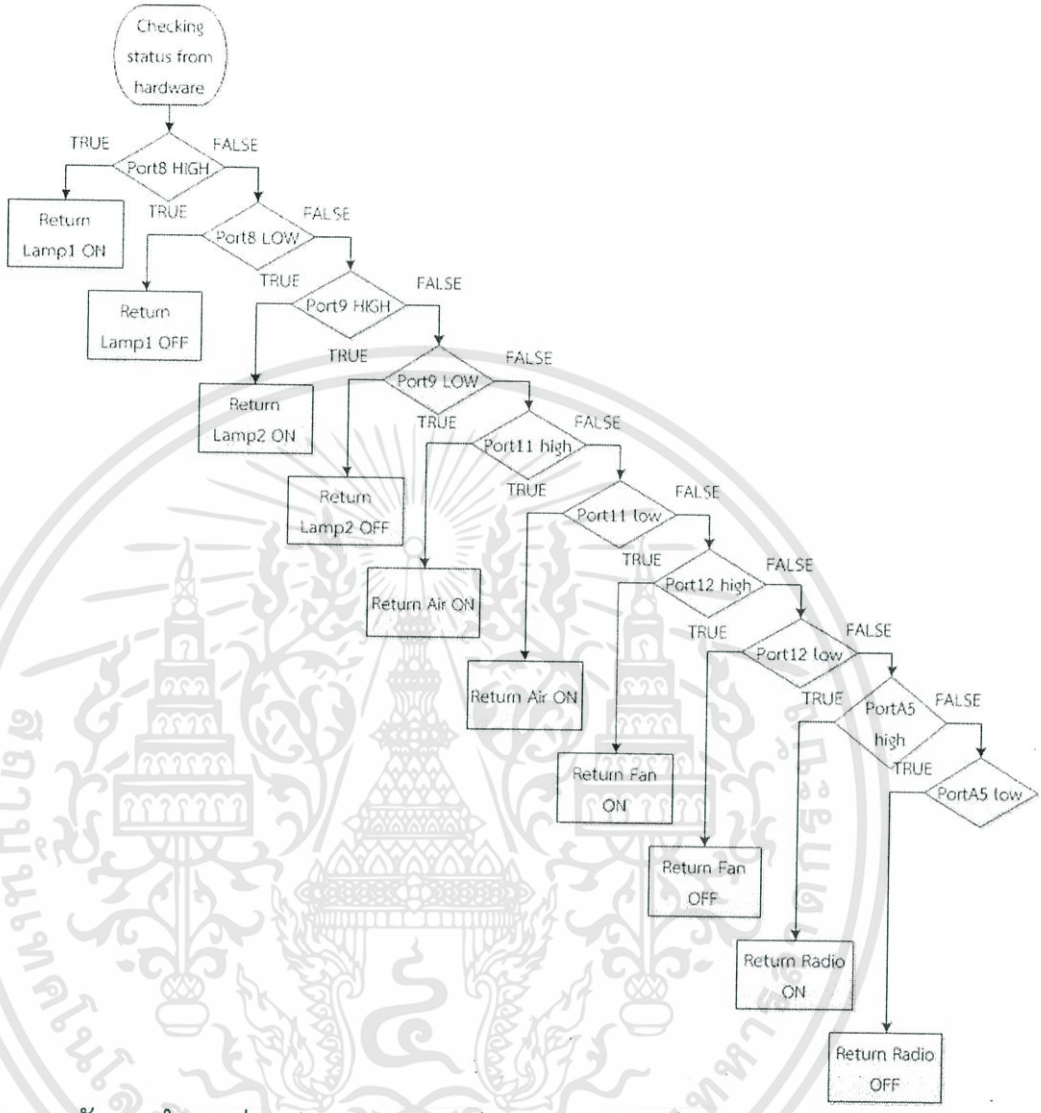
3.1.2 ส่วนประมวลผล

แบ่งการทำงานเป็น 3 ส่วนคือส่วนประมวลผลการรับค่าคำสั่งจากแอนดรอยด์ ส่วนประมวลผลสั่งงาน เปิด-ปิด อุปกรณ์ โดยมีแผนผังการทำงานดังรูปที่ 3.13



รูปที่ 3.13 แผนผังการทำงานของส่วนประมวลผล

และ ส่วนประมวลผลสถานะของอุปกรณ์เพื่อส่งค่าคืนให้แอนดรอยด์ ซึ่งมีแผนผังการทำงานดังรูปที่ 3.14



รูปที่ 3.14 แผนผังการทำงานส่วนประมวลผลการส่งสถานะอุปกรณ์ไปยังแอนดรอยด์

3.1.2.1 ส่วนประมวลผลการรับค่าจากแอนดรอยด์

เขียนโปรแกรมเพื่อรับค่าคำสั่งจากแอนดรอยด์ โดยแอนดรอยด์จะส่งตัวเลขคำสั่งมาเพื่อสั่งเปิด-ปิดอุปกรณ์ผ่านทาง TCP/IP ทำการประมวลผลคำสั่งว่าส่งคำสั่งอะไรมา

3.1.2.2 ส่วนประมวลผลสั่งงานเปิด-ปิดอุปกรณ์

เขียนโปรแกรมเพื่อสั่งการเปิด-ปิดอุปกรณ์ไฟฟ้าตามที่ได้รับคำสั่งมาจากแอนดรอยด์

3.1.2.3 ส่วนประมวลผลสถานะของอุปกรณ์และส่งคืนให้แอนดรอยด์

เขียนโปรแกรมเพื่อตรวจสอบสถานะของอุปกรณ์ผ่านทางวงจร ตรวจสอบสถานะของอุปกรณ์และส่งค่าสถานะของอุปกรณ์คืนให้กับส่วนแอนดรอยด์

3.1.3 ส่วนฮาร์ดแวร์

ส่วนฮาร์ดแวร์รวมนั้น รับข้อมูลมาจากส่วนเชื่อมต่อระบบอีกทอดหนึ่ง และคอบส่งค่ากลับให้ด้วย แต่เนื่องจากส่วนเชื่อมต่อของระบบนั้น ก็ใช้โปรแกรมอาดูโน่ และไมโครคอนโทรลเลอร์เดียวกันในการสั่งใช้งานส่วนแสดงผล เพราะฉะนั้นส่วนฮาร์ดนี้จึงรวมโค้ดโปรแกรม และตัวฮาร์ดแวร์ของงานทั้งหมดไว้ด้วยกัน

โดยต้องเขียนโปรแกรมไว้คอยรับค่าที่จะเข้ามาจากทางด้านผู้ใช้ เพื่อทำงานควบคุมเครื่องใช้ไฟฟ้าตามที่ได้รับคำสั่ง และส่งค่าอุณหภูมิจากเซ็นเซอร์ออกไปด้วยพร้อมๆ กัน โดยบอร์ดอาดูโน่เป็นตัวหลักในการควบคุม เพื่อเป็นตัวควบคุมการเปิด-ปิดของอุปกรณ์ไฟฟ้า โดยมีบล็อกไดอะแกรมการทำงานของส่วนฮาร์ดแวร์ดังรูปที่ 3.15

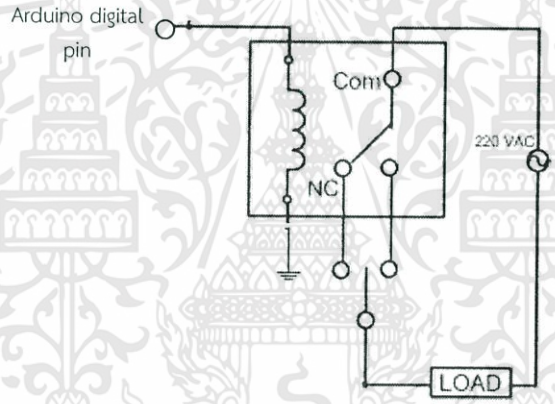
รูปที่ 3.15 บล็อกไดอะแกรมการทำงานของส่วนฮาร์ดแวร์

จากที่ได้กล่าวมาในข้างต้นถึงการทำงานในส่วนฮาร์ดแวร์ และจากบล็อกไดอะแกรมในรูปที่ 3.15 จะแบ่งการทำงานของส่วนฮาร์ดแวร์ได้ 2 ส่วนใหญ่ๆ โดยทั้งสองส่วนใช้โปรแกรมอาดูโน่ในการเขียนโค้ดคำสั่ง

3.1.3.1 ส่วนที่รับข้อมูล

1) วงจรควบคุมการเปิดปิดอุปกรณ์ไฟฟ้า

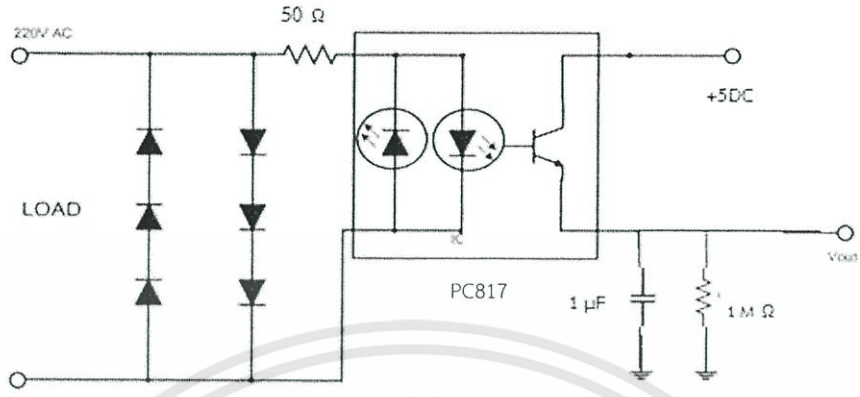
จะรับข้อมูลการเปิด-ปิด และพอร์ทของเครื่องใช้ไฟฟ้า จากนั้นทำการควบคุมให้เครื่องใช้ไฟฟ้านั้นๆ เปิด หรือ ปิดตามคำสั่ง โดยให้ไมโครคอรโทรลเลอร์และรีเลย์เป็นตัวหลักในการควบคุม โดยออกแบบให้สามารถควบคุมอุปกรณ์ไฟฟ้าได้ 5 ตัว โดยใช้รีเลย์ 5 ตัว แทนสวิตซ์ควบคุมการเปิด-ปิด ในที่นี้จะเรียกว่า 5 พอร์ท มีการออกแบบวงจร ดังรูปที่ 3.16



รูปที่ 3.16 วงจรควบคุมอุปกรณ์ไฟฟ้า

2) วงจรตรวจสอบกระแสไฟ (ตรวจสอบสถานะของอุปกรณ์ไฟฟ้า)

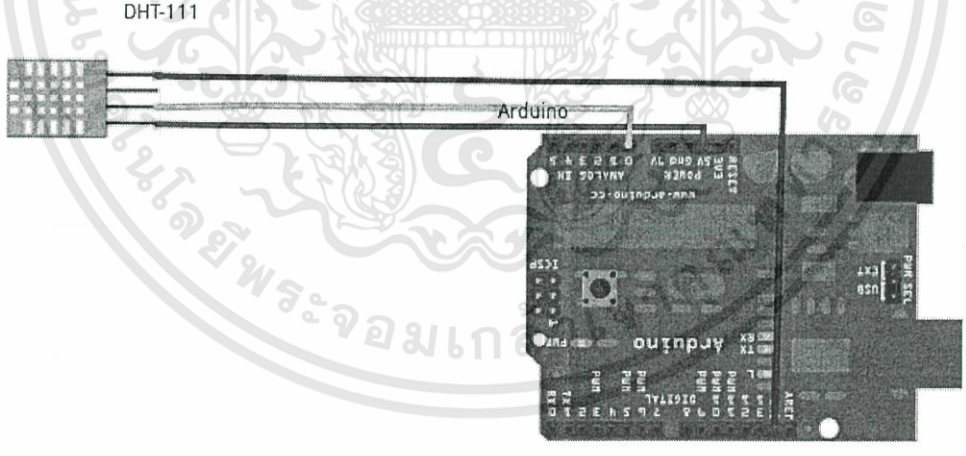
เพื่อตรวจสอบว่าอุปกรณ์นั้นๆ ได้เปิด หรือปิดอยู่จริงๆ โดยใช้การทำงานของออปโต้คัปเปลอร์เป็นตัวหลัก ออปโต้คัปเปลอร์จะทำงานเมื่อมีแรงดันไฟฟ้าผ่านไดโอด แล้วทำการส่งเอาท์พุตจากออปโต้คัปเปลอร์เข้าขาดิจิตอลอินพุทของบอร์ดอาดูโน่ เพื่อรับสถานะว่าเป็นสูง หรือต่ำ โดยมีรูปแบบวงจรตรวจสอบกระแสไฟดังรูปที่ 3.17



รูปที่ 3.17 วงจรตรวจสอบกระแสไฟ

3.1.3.2 ส่วนส่งข้อมูลออก

คราวนี้อินพุทจะเข้าทางฝั่งฮาร์ดแวร์ นั่นคือเซ็นเซอร์อุณหภูมิ DHT11 และแอลดีอาร์ เมื่อมีอินพุทเข้ามาทางตัวเซ็นเซอร์ แล้วผ่านการประมวลผลตามที่เราได้เขียนโค้ดไว้แล้วนั้น ก็จะส่งข้อมูลออกไปให้แก่แอนดรอยด์ ดังรูป 3.18



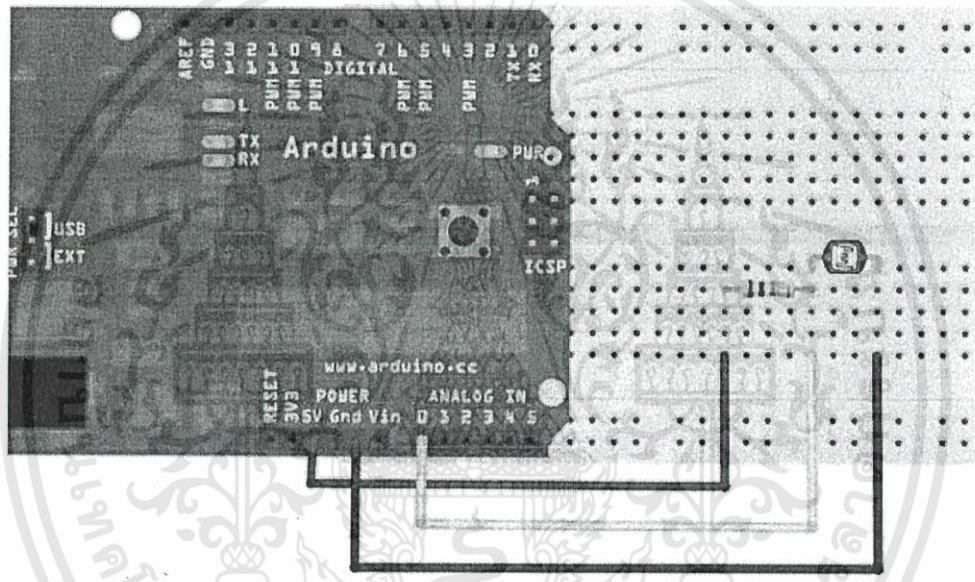
รูปที่ 3.18 การเชื่อมต่อระหว่างเซ็นเซอร์ DHT11 กับบอร์ดอาดูโน่

แสดงรายละเอียดขาของการเชื่อมต่อระหว่างเซ็นเซอร์ DHT11 กับบอร์ดอาดูโน่ ได้ดังตารางที่ 1

ตารางที่ 3.1 การเชื่อมต่อระหว่างเซ็นเซอร์ DHT11 กับบอร์ดอาดูโน่

DHT11	Arduino
Pin 1	Vcc
Pin 2	Analog0
Pin 4	Gnd

และการเชื่อมต่อระหว่างแอลดีอาร์ กับอาดูโน่ ดังรูปที่ 3.19

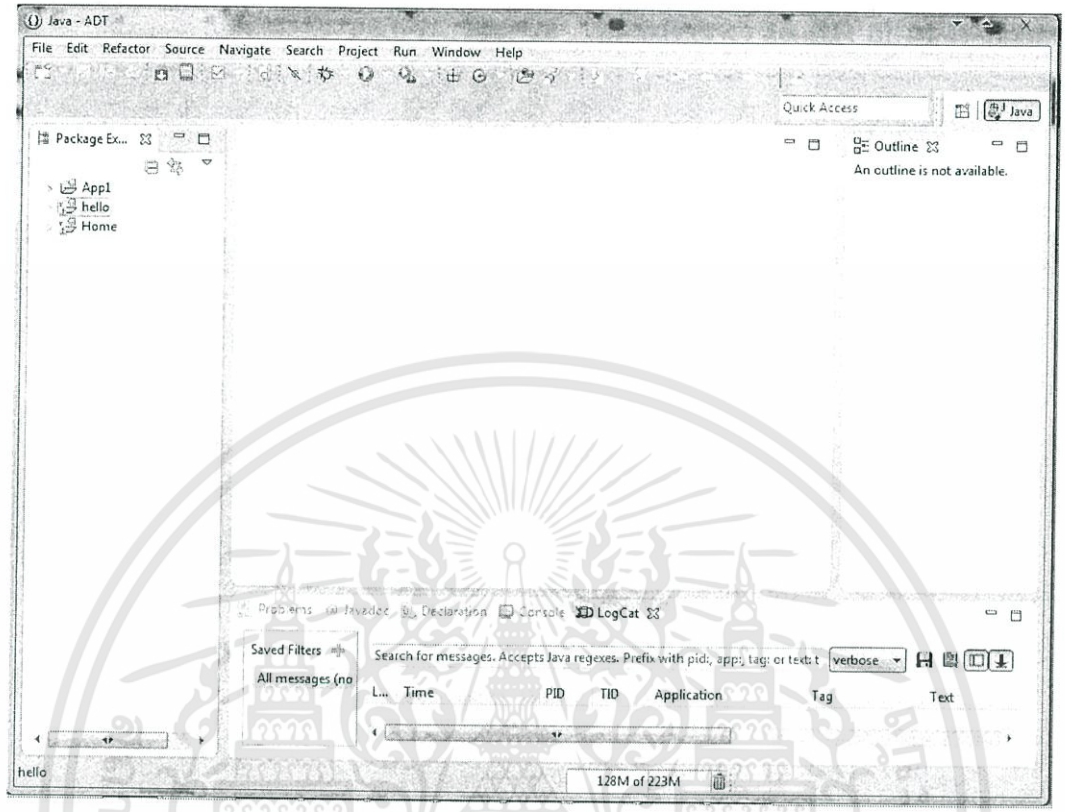


รูปที่ 3.19 การเชื่อมต่อระหว่างแอลดีอาร์ กับบอร์ดอาดูโน่

3.2 เครื่องมือที่ใช้ในการทดลอง

3.2.1 โปรแกรม Eclipse

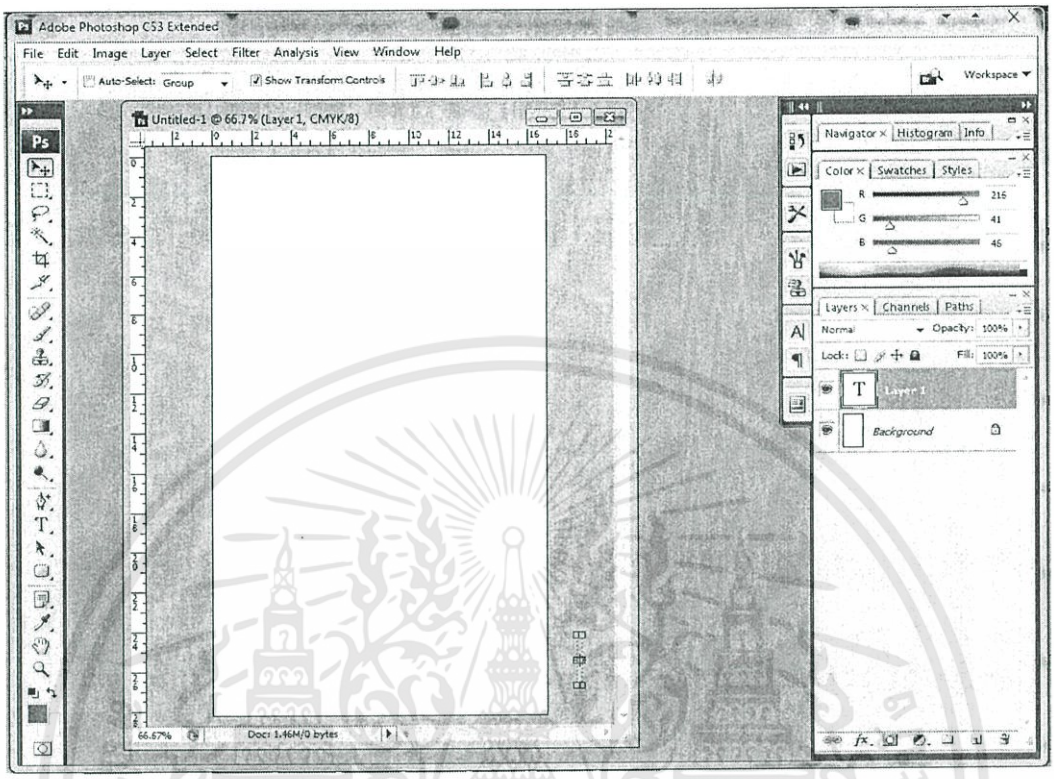
Eclipse คือโปรแกรมที่ใช้สำหรับพัฒนาภาษา Java ซึ่งโปรแกรม Eclipse เป็นโปรแกรมหนึ่งที่ใช้ในการพัฒนา Application Server ได้อย่างมีประสิทธิภาพ และเนื่องจาก Eclipse เป็นซอฟต์แวร์ Open Source ที่พัฒนาขึ้นเพื่อใช้โดยนักพัฒนาเอง ทำให้ความก้าวหน้าในการพัฒนาของ Eclipse เป็นไปอย่างต่อเนื่องและรวดเร็ว ในที่นี้ใช้สำหรับพัฒนาแอนดรอยด์แอพพลิเคชั่น หน้าต่างโปรแกรมแสดงดังรูปที่ 3.20



รูปที่ 3.20 หน้าต่างโปรแกรม Eclipse

3.2.2 โปรแกรม Adobe Photoshop

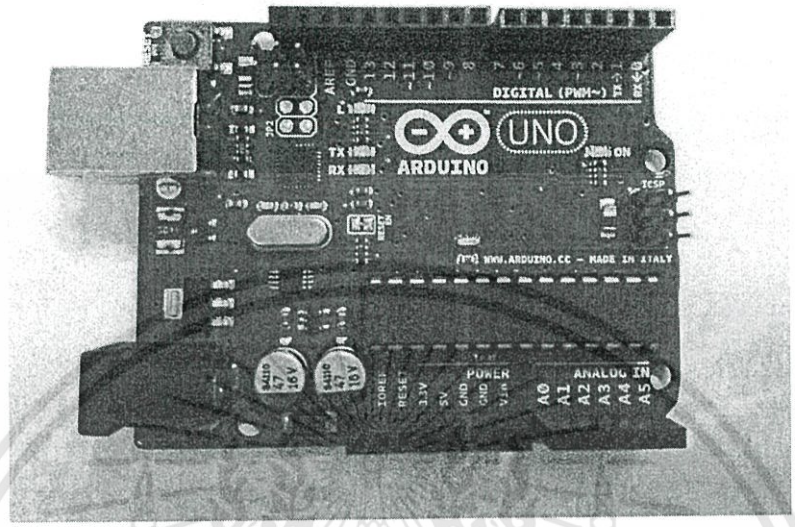
ใช้สำหรับออกแบบส่วนติดต่อผู้ใช้งานหน้าต่างๆ โดยโปรแกรม Photoshop เป็นโปรแกรมที่มีความสามารถในการออกแบบกราฟิก เพื่อนำไปใช้ร่วมกับงานในด้านต่าง ๆ เช่น งานกราฟิกที่เกี่ยวกับสื่อสิ่งพิมพ์ทุกประเภท งานกราฟิกบนเว็บไซต์และการตกแต่งภาพถ่ายจากกล้องดิจิตอล ซึ่งอาจกล่าวได้ว่าเป็นโปรแกรมที่มีผู้นิยมนำมาใช้ในการออกแบบและตกแต่งภาพถ่ายกันมากที่สุด หน้าต่างโปรแกรมแสดงดังรูปที่ 3.21



รูปที่ 3.21 หน้าต่างโปรแกรม Photoshop

3.2.3 บอร์ดอาร์ดูโน และไมโครคอนโทรลเลอร์ ATmega328p-pu

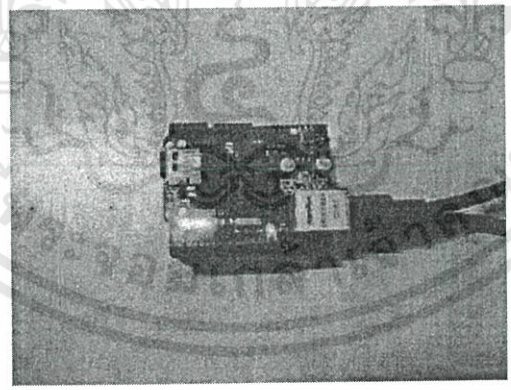
ใช้ควบคุมการทำงานของระบบ โดยใช้บอร์ด Arduino Uno R3 และใช้ไมโครคอนโทรลเลอร์ AVR เบอร์ ATmega328p-pu แสดงในรูปที่ 3.22



รูปที่ 3.22 บอร์ดอาร์ดูโนที่มีไมโครคอนโทรลเลอร์เบอร์ ATmega328p-pu

3.2.4 อาร์ดูโนอีเธอร์เน็ตชิลด์ (Arduino Ethernet Shield)

อุปกรณ์เสริมของอาร์ดูโนที่ใช้เพื่อเชื่อมต่อผ่าน TCP/IP ทำให้แอนดรอยด์สามารถเชื่อมต่อกับไมโครคอนโทรลเลอร์อาร์ดูโนได้ ดังรูปที่ 3.23



รูปที่ 3.23 อาร์ดูโนอีเธอร์เน็ตชิลด์

3.2.5 เราเตอร์ (Router)

ทำหน้าที่หาเส้นทางและส่ง package ข้อมูลระหว่างเครือข่ายคอมพิวเตอร์ ไปยังเครือข่ายปลายทางที่ต้องการ ทำการจ่าย IP Address ให้กับอุปกรณ์ที่ต่อกับเราเตอร์ตัว Router จะมีช่องที่ใช้เสียบต่อสายสัญญาณเรียกว่า Port LAN ซึ่งโดยทั่วไปมักมี 4 Ports หรือมากกว่า ใน เราเตอร์ 1 ตัว ดังรูปที่ 3.24

รูปที่ 3.24 เราเตอร์

3.2.6 สาย RJ-45

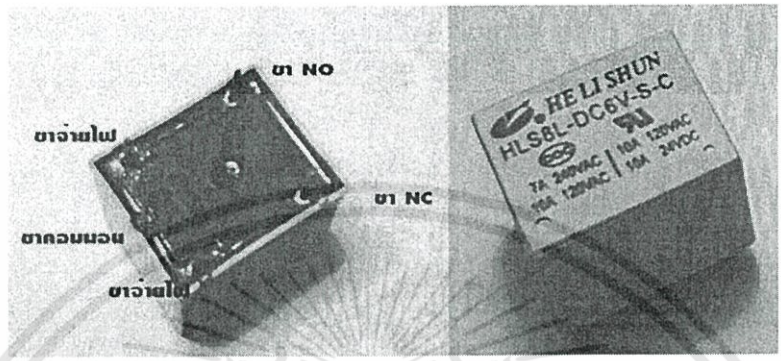
ทำหน้าที่เชื่อมต่อไมโครคอนโทรลเลอร์กับเราเตอร์ เพื่อส่งข้อมูลผ่าน TCP/IP โดยสาย RJ-45 แสดงดังรูปที่ 3.25

รูปที่ 3.25 สาย RJ-45

3.2.7 รีเลย์

ในโครงงานได้ใช้รีเลย์แบบ 5 ขา ประกอบไปด้วยขาจ่ายแรงดันใช้งาน 2 ขา ขา C หรือคอมมอน (Common) เป็นขาระหว่าง NO และ NC ขา NO (Normally opened) ซึ่งปกติขานี้จะเปิดเอาไว้ จะทำงานก็ต่อเมื่อป้อนแรงดันให้รีเลย์ ส่วนขาสุดท้าย NC (Normally closed) โดย

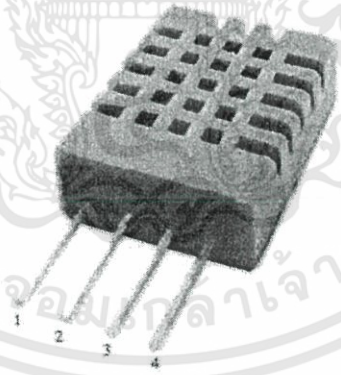
ปกติขานี้จะต่อกับขา C ในกรณีที่ไม่ได้จ่ายแรงดัน หน้าสัมผัสของ C และ NC จะต่อถึงกัน ดังรูปที่ 3.26



รูปที่ 3.26 รีเลย์ 5 ขา

3.2.8 เซ็นเซอร์ความชื้น และอุณหภูมิ DHT11

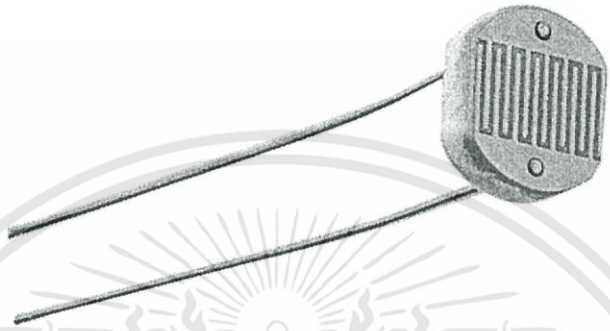
ใช้สำหรับวัดอุณหภูมิภายในห้องนั้นๆ ณ ที่เซนเซอร์อยู่ โดยตัวเซนเซอร์เราใช้งานเพียง 3 ขา จากทั้งหมด 4 ขา ดังที่กล่าวไปในส่วนของการออกแบบ ดังรูปที่ 3.27



รูปที่ 3.27 เซ็นเซอร์ความชื้น และอุณหภูมิ DHT11

3.2.9 แอลดีอาร์

ใช้สำหรับวัดแสง เพื่อส่งข้อมูลไปยังส่วนหน้าจอผู้ใช้งาน แสดงความสว่างของที่นั้นแสดงบนหน้าจอมือถือ ดังรูปที่ 3.28



รูปที่ 3.28 แอลดีอาร์

3.2.10 โปรแกรม Arduino IDE

ใช้สำหรับเขียนโปรแกรมสั่งงานไมโครคอนโทรลเลอร์ โดยเมื่อเขียนโค้ดเสร็จแล้วโปรแกรมจะสามารถอัพโหลด (upload) ลงไมโครคอนโทรลเลอร์ผ่านยูเอสบีพอร์ตได้เลย หน้าต่างโปรแกรมแสดงดังรูปที่ 3.29 และไอคอนโปรแกรมแสดงดังรูปที่ 3.30

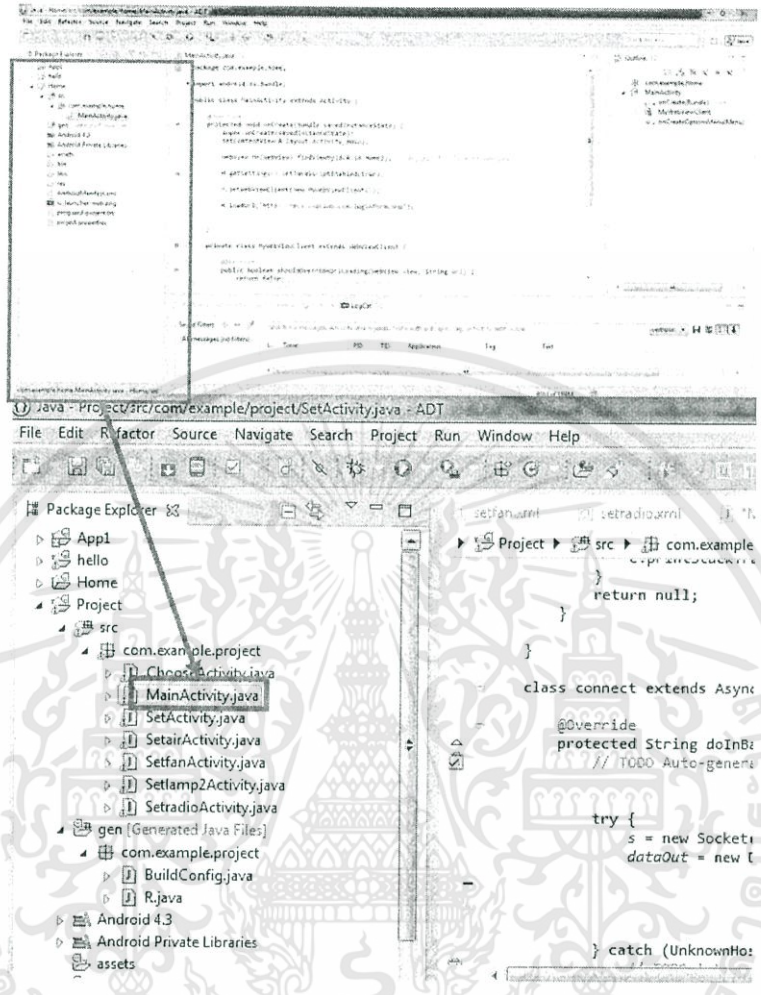
รูปที่ 3.29 หน้าต่างโปรแกรม arduino IDE

รูปที่ 3.30 ไอคอนโปรแกรม arduino IDE

3.3 การจัดเก็บผลการทดลอง

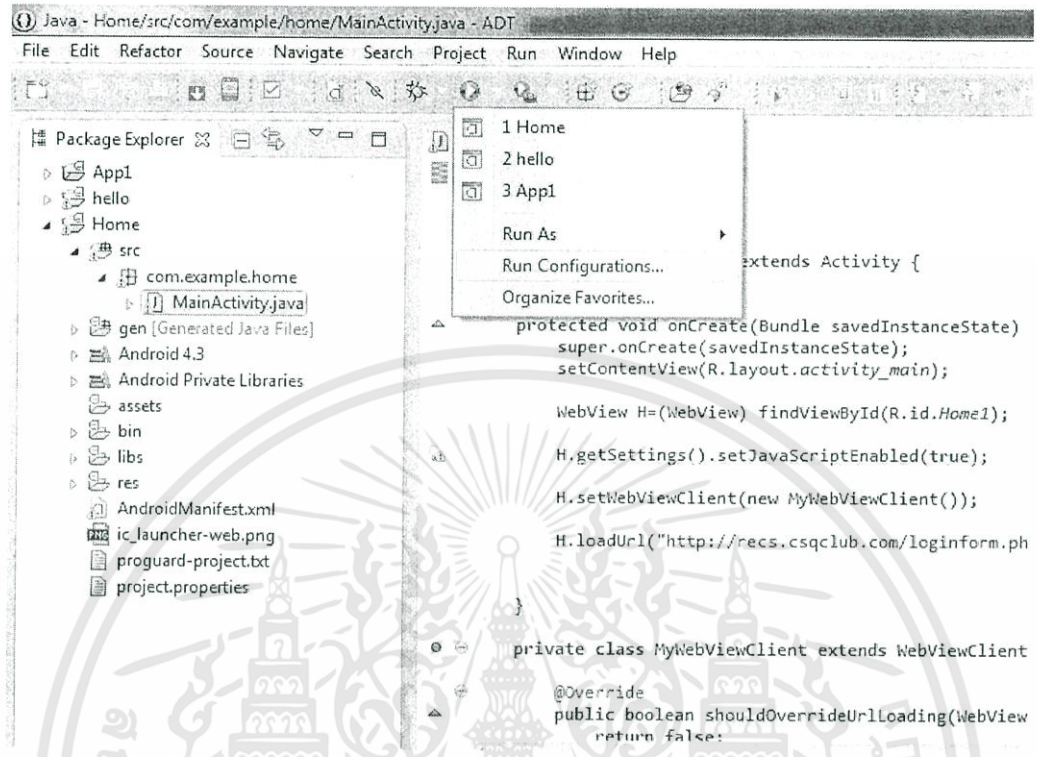
3.3.1 ส่วนแอพพลิเคชั่นบนแอนดรอยด์สมาร์ทโฟน

3.3.1.1 ตัวอย่างเปิดโปรแกรม Eclipse > src > com.example.home >MainActivity.java จะพบกับโค้ดโปรแกรมชื่อ MainActivity.java ซึ่งเป็นโค้ดโปรแกรมหน้าการเข้าสู่ระบบ และสามารถเปิดหน้าต่างอื่นได้ในชื่อถัดๆไป โดยหน้าต่างโปรแกรมแสดงดังรูปที่ 3.31



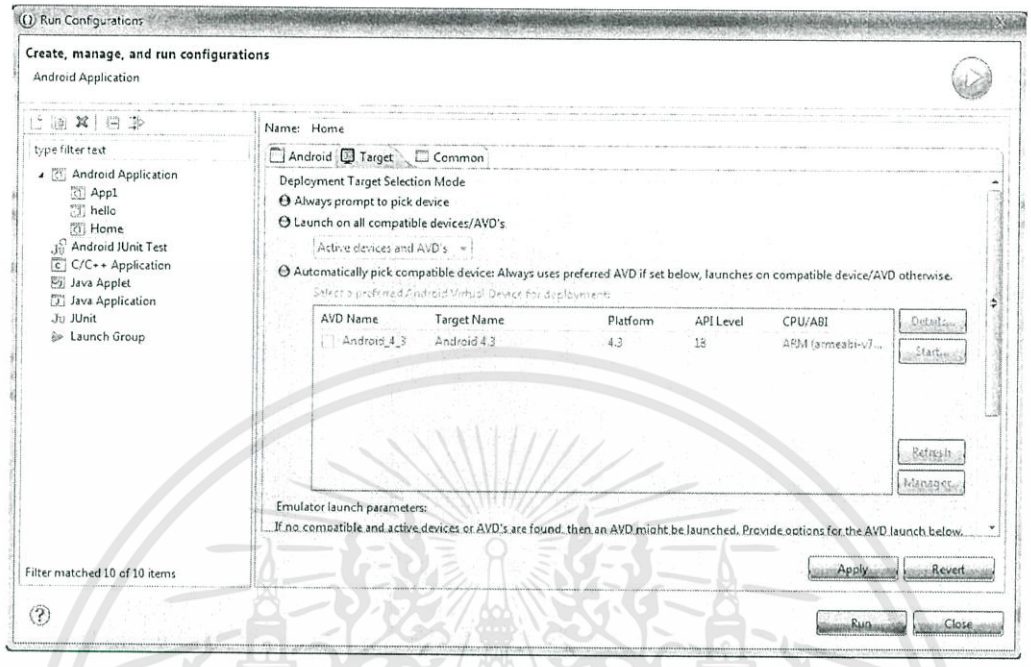
รูปที่ 3.31 หน้าต่างโปรแกรม

3.3.1.2 เข้าที่ Run Configuration เพื่อตั้งระบบให้เป็นแบบ manual เพื่อให้สามารถเลือกใช้งานได้กับอุปกรณ์ต้องการ ดังรูปที่ 3.32

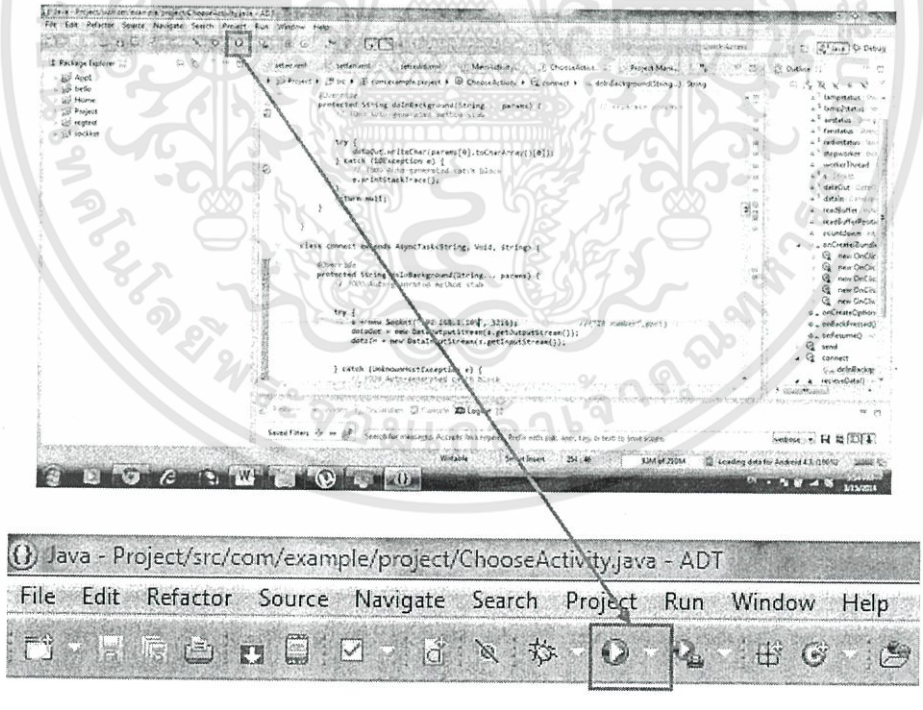


รูปที่ 3.32 หน้าต่างโปรแกรม

3.3.1.3 กดที่ Target > Always prompt to pick device > Apply เพื่อให้สามารเลือกอุปกรณ์ที่ต้องการ หน้าต่างโปรแกรมดังรูปที่ 3.33 และหลังจากนั้นสามารถกดรันได้ที่ปุ่มดังรูป 3.34 เพื่อเปิดแอพพลิเคชั่น

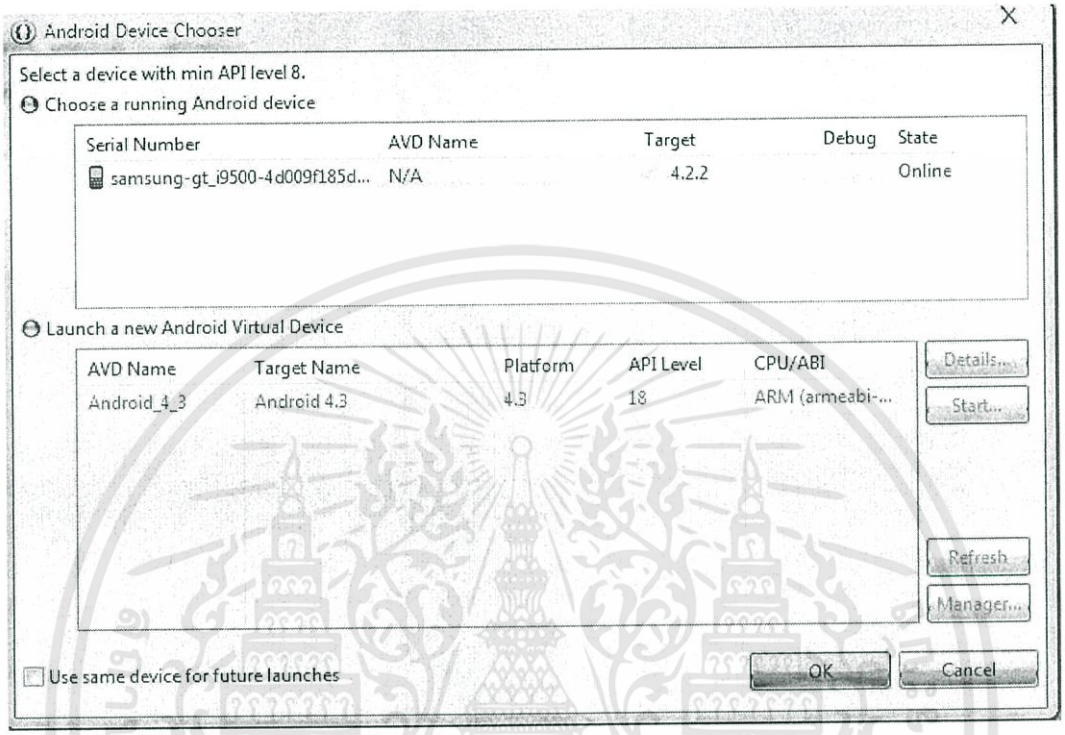


รูปที่ 3.33 หน้าต่างโปรแกรม



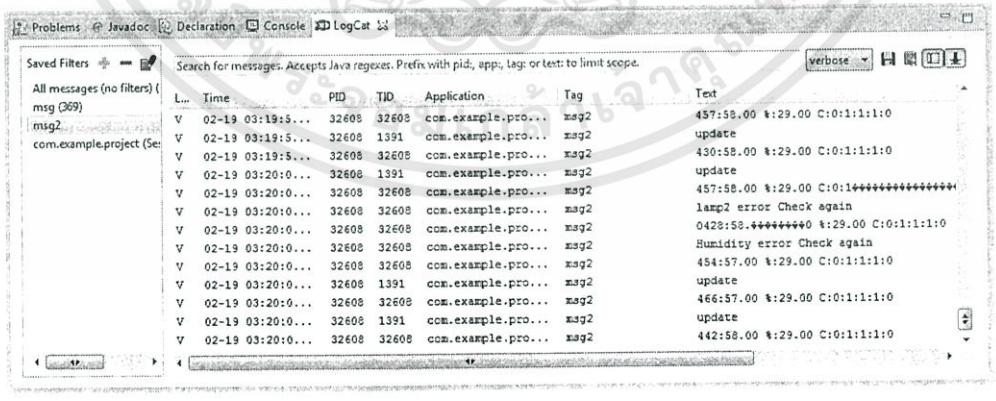
รูปที่ 3.34 ปุ่มสำหรับรันโปรแกรม

3.3.1.4 เลือกอุปกรณ์ที่ต้องการ โดยคลิกที่อุปกรณ์ที่ต้องการ ดังรูปที่ 3.35



รูปที่ 3.35 หน้าต่างโปรแกรม

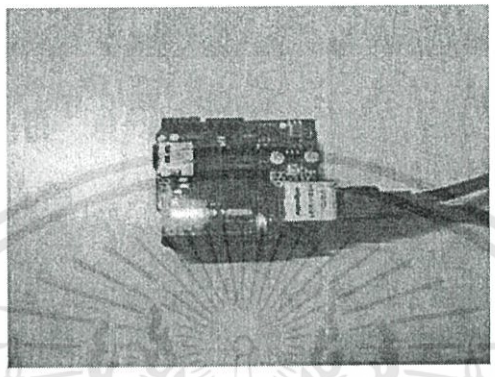
3.3.1.5 เมื่อแอพพลิเคชั่นทำงานจะแสดงผลสถานะต่างๆออกทางหน้าต่างสถานะ ดังรูปที่ 3.36



รูปที่ 3.36 หน้าต่างสถานะ

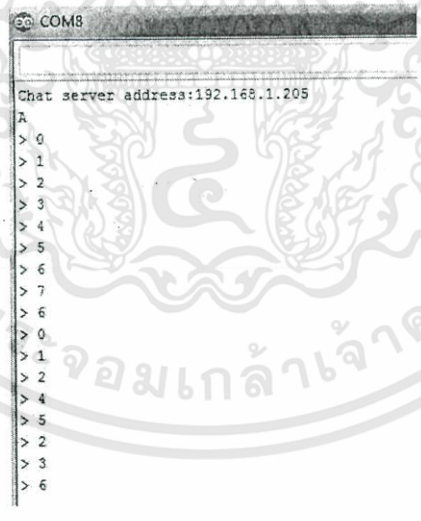
3.3.2 ส่วนประมวล

ทำการเขียนโค้ดโปรแกรมในโปรแกรม Arduino IDE ทดลองให้ส่วนแอนดรอยด์ส่งคำสั่งมาที่ไมโครคอนโทรลเลอร์ที่ต่อสาย RJ-45 ผ่านทางอีเธอร์เน็ตชิลด์ ดังรูปที่ 3.37



รูปที่ 3.37 ไมโครคอนโทรลเลอร์ที่ต่อสาย RJ-45 ผ่านอีเธอร์เน็ตชิลด์

เปิดหน้าจอซีเรียลมอนิเตอร์เพื่อดูค่าตัวเลขคำสั่งที่แอนดรอยด์ส่งมา ดังรูปที่ 3.38



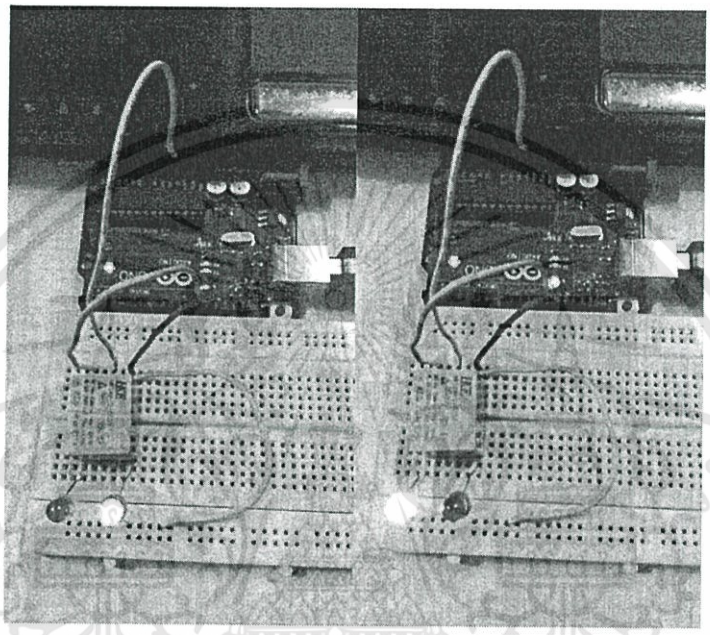
รูปที่ 3.38 หน้าจอซีเรียลมอนิเตอร์เมื่อมีการรับค่าจากแอนดรอยด์

จะเห็นว่าได้ไมโครคอนโทรลเลอร์ได้รับคำสั่งการ เปิด-ปิด อุปกรณ์ไฟฟ้าจากแอนดรอยด์มาเป็นคำสั่งตัวเลข

3.3.3 ส่วนฮาร์ดแวร์

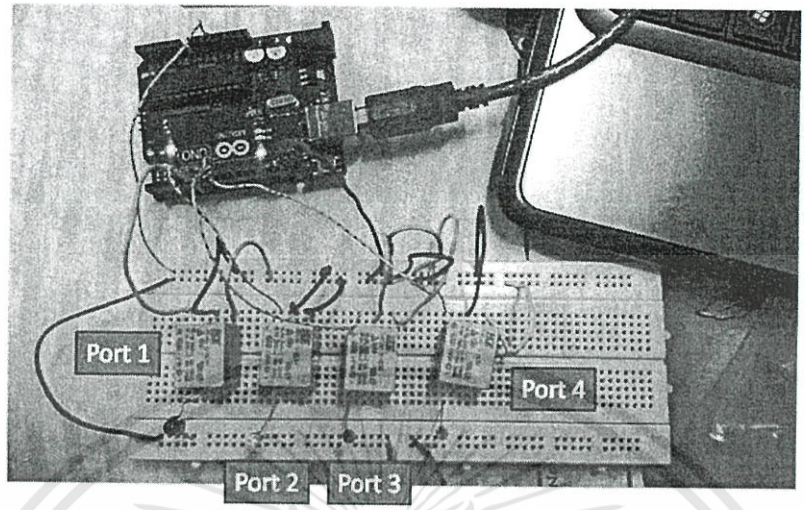
3.3.3.1 ส่วนควบคุมอุปกรณ์เครื่องใช้ไฟฟ้า

1) ทดลองต่อรีเลย์กับบอร์ดอาร์ดูโน่ และเขียนโปรแกรมอัพโหลดลงไมโครคอนโทรลเลอร์เพื่อทดลองการทำงานเปิด-ปิด หลอดแอลอีดี ดังรูปที่ 3.39



รูปที่ 3.39 การต่อบอร์ดอาร์ดูโน่ กับรีเลย์ และหลอดแอลอีดี

2) ต่อไปทดลองต่อรีเลย์ให้เพิ่มเป็น 4 พอร์ท เพื่อรองรับเครื่องใช้ไฟฟ้า 4 ตัว (ใช้หลอดแอลอีดีในการทดลองแทนเครื่องใช้ไฟฟ้า) โดยให้ด้านซ้ายสุดเป็นพอร์ท 1 แล้วเรียงลำดับเป็นพอร์ทที่ 2, 3 และ 4 อยู่ด้านขวาสุด ดังรูปที่ 3.40

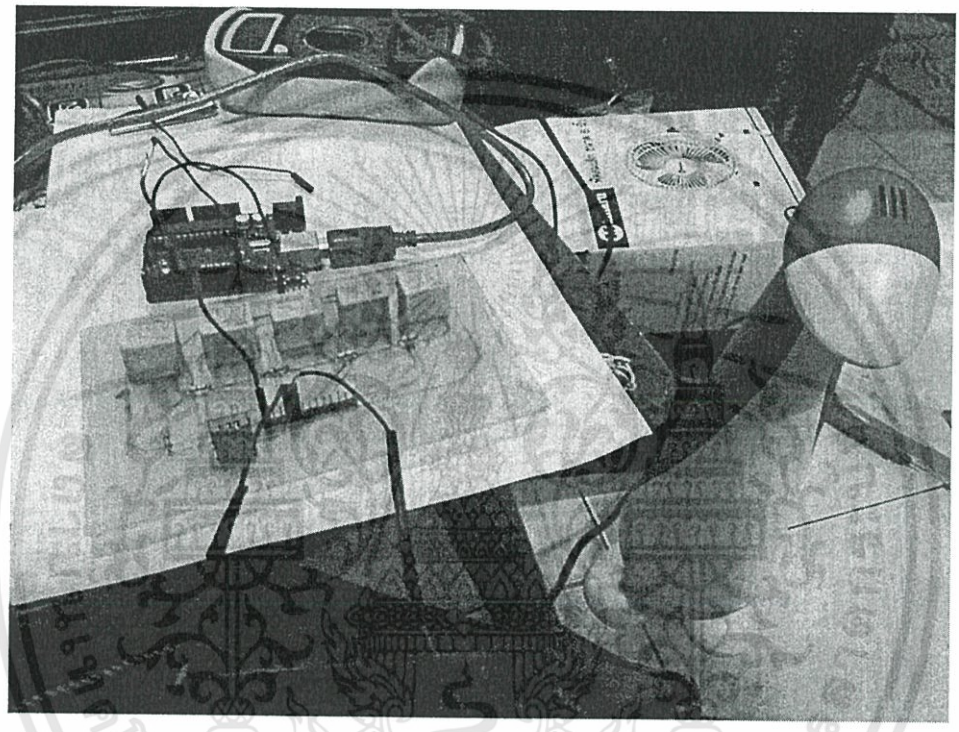


รูปที่ 3.40 วงจรฮาร์ดแวร์จำลองที่มีครบ 4 พอร์ท

จากนั้น ทดลองพิมพ์ตัวแปร จำลองตัวแปรที่จะต้องได้รับมาจากซีเรียลพอร์ท เพื่อทดลองควบคุมการเปิด-ปิด ของหลอดแอลอีดี โดยใช้ออพชั่น (option) ซีเรียลมอนิเตอร์ (Serial Monitor) ของโปรแกรมอาดูโน่ โดยทดลองส่งข้อมูล a, b, c, d, e และ f (ตัวเลขที่แสดงบนหน้าจอซีเรียลมอนิเตอร์ เป็นภาษาแอสกี้ (ASCII) ของ a, b, c, d, e และ f) ดังรูปที่ 3.41 จากนั้นสังเกตุผลที่เกิดขึ้นกับหลอดแอลอีดี

รูปที่ 3.41 หน้าต่างหน้าจอซีเรียลมอนิเตอร์

3) จากนั้นทำการทดสอบกับเครื่องใช้ไฟฟ้าจริง โดยทดสอบกับโคมไฟ เนื่องจากสายไฟของเครื่องใช้ไฟฟ้าจะมี 2 เส้น เราทำการแยกสายออกมาหนึ่งเส้น แล้วผ่าแยกตรงกลาง เพื่อนำไปคร่อมกับตัวรีเลย์ เพื่อเป็นโหลดของรีเลย์ และให้รีเลย์เป็นตัวควบคุมการจ่ายไฟให้ครบวงจรแก่เครื่องใช้ไฟฟ้านั้นๆ นั่นเอง จากนั้นจึงนำไปทดลองกับวงจร เพื่อทดสอบการทำงาน ดังรูปที่ 3.42 (จากรูปทดสอบพอร์ท 1)

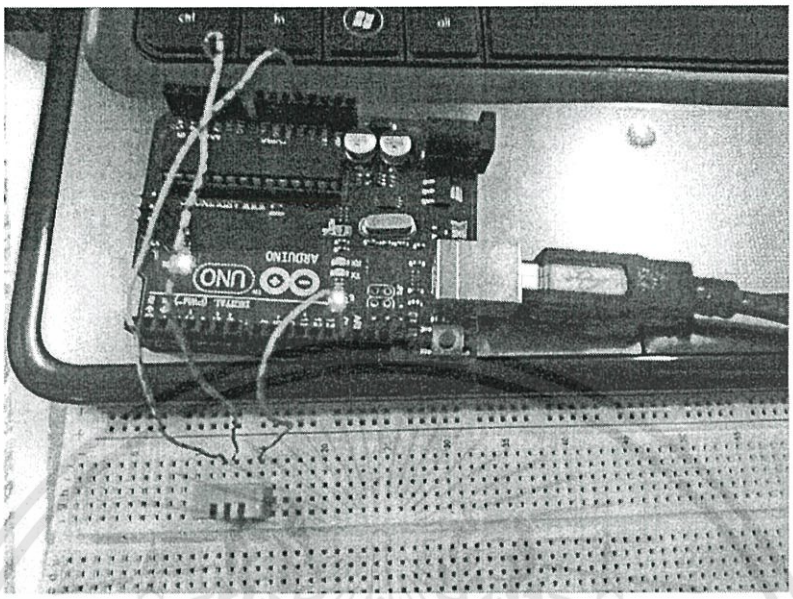


รูปที่ 3.42 ทดสอบการทำงานกับเครื่องใช้ไฟฟ้าจริง

3.3.3.2 ส่วนเซ็นเซอร์อุณหภูมิ

ทำการดาวโหลดไลบรารี่ของ dht มาก่อน เพื่อนำมาใช้ในการทำงาน และเขียนโปรแกรมให้ค่าที่ได้จากเซ็นเซอร์ แสดงอุณหภูมิขึ้นทางหน้าจอซีเรียลมอนิเตอร์

จากนั้นทำการอัพโหลดโค้ดโปรแกรมเข้าในบอร์ดอาร์ดูโน่ และดูผลการทำงานจากหน้าจอซีเรียลมอนิเตอร์ โดยทดลองวางที่อุณหภูมิห้องปกติ และทดสอบกับอุณหภูมิที่แตกต่างเพื่อทดลองการทำงานของเซ็นเซอร์ และโค้ดโปรแกรม ดังรูปที่ 3.43



รูปที่ 3.43 ทดสอบการทำงานของเซ็นเซอร์ และโค้ดโปรแกรม

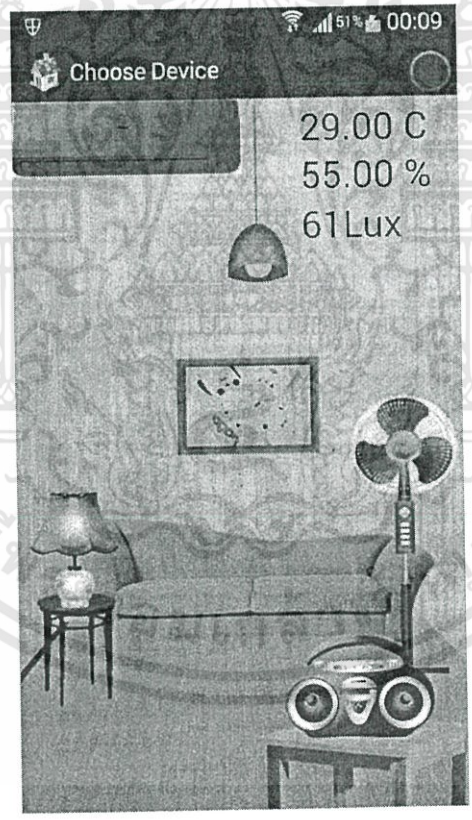
บทที่ 4

ผลการทดลอง

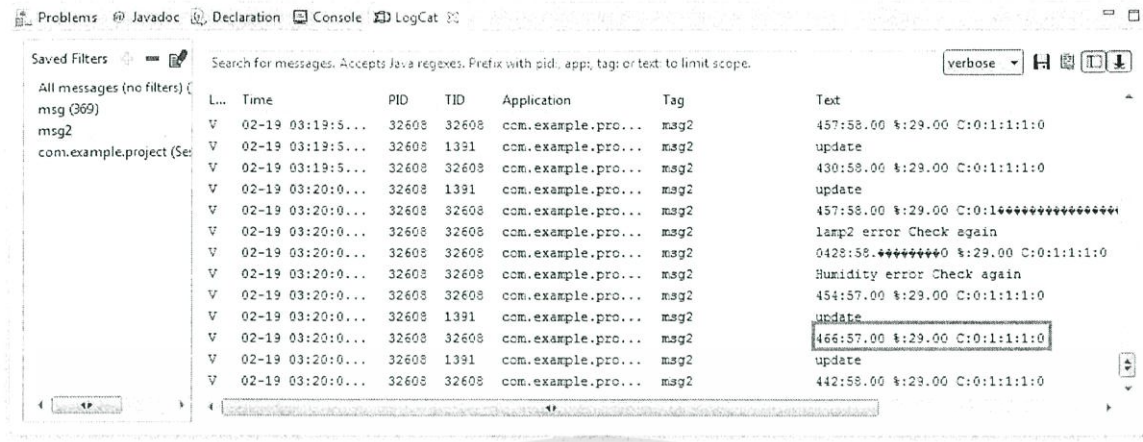
ผลการทดลองจะแบ่งออกเป็นสี่ส่วนใหญ่ นั่นคือ ส่วนแอพพลิเคชั่นบนแอนดรอยด์สมาร์ทโฟน ส่วนประมวลผลที่รับค่าจากแอนดรอยด์ ส่วนฮาร์ดแวร์ และผลโดยรวมของระบบ

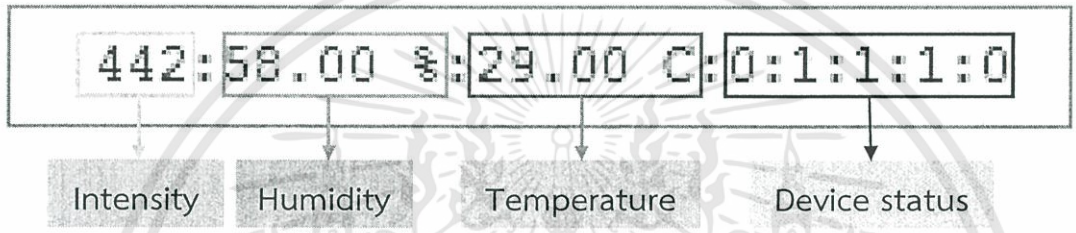
4.1 ส่วนแอพพลิเคชั่นบนแอนดรอยด์สมาร์ทโฟนที่รับค่าจากอาร์ดูโน

เมื่อผู้ใช้เข้ามาในหน้าเลือกอุปกรณ์ที่ต้องการ มีการรับข้อมูลแสดงผลว่าอุปกรณ์ชิ้นใดเปิดหรือปิดอยู่ ซึ่งเมื่อรับมาเป็น 0 จะหมายความว่าอุปกรณ์ชนิดนั้นปิดอยู่ และหากรับมาเป็น 1 จะหมายความว่าอุปกรณ์ชนิดนั้นเปิดอยู่ นอกจากนั้นจะแสดงระดับอุณหภูมิในหน่วยองศาเซลเซียส ความชื้นและความสว่างภายในห้องด้วย จากตัวอย่างดังรูปที่ 4.1 คือกรณีที่ไฟแขวนเปิด โคมไฟปิด พัดลมเปิด วิทยุปิด เครื่องปรับอากาศปิด และแสดงข้อมูลที่รับจากอาดูโนทั้งหมดผ่านทางหน้าจอ Logcat เพื่อแสดงผลการรับข้อมูลได้จริงในรูปของแพ็คเกจข้อมูลที่มีการเรียกข้อมูลทุกๆ 10 วินาที เพื่อแสดงผลได้ถูกต้องตัวอย่างแพ็คเกจข้อมูลดังรูปที่ 4.2



รูปที่ 4.1 หน้าเลือกอุปกรณ์ที่ต้องการควบคุม



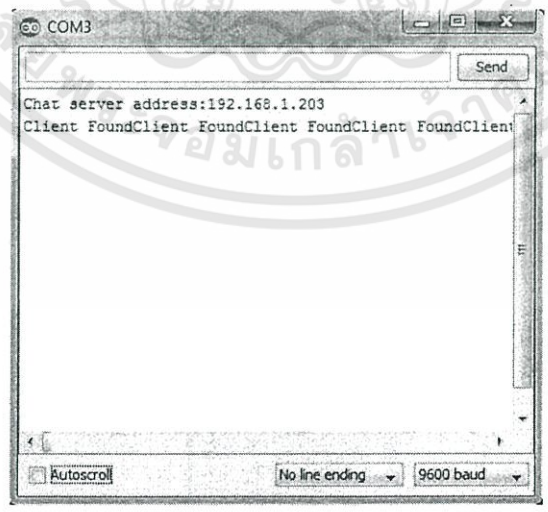


รูปที่ 4.2 หน้าจอ Logcat

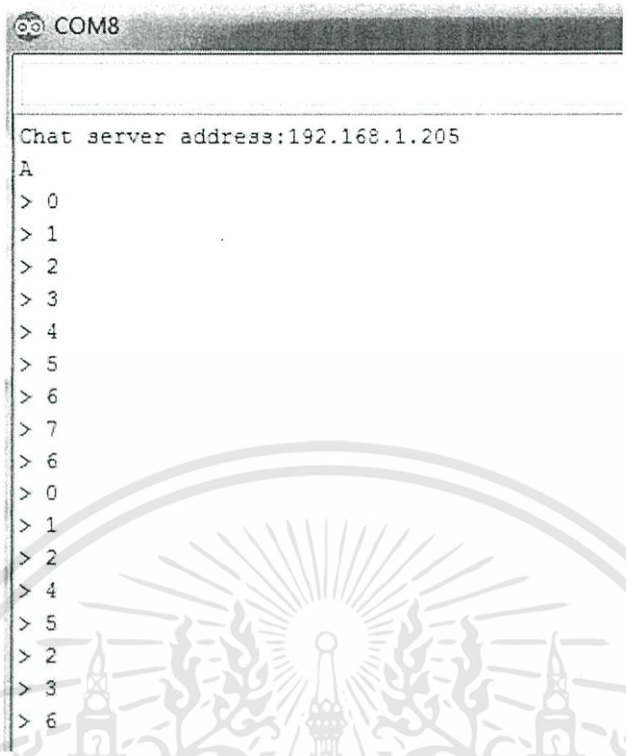
4.2 ส่วนประมวลผลที่รับค่าจากแอนดรอยด์

4.2.1 ส่วนประมวลผลรับค่าจากแอนดรอยด์

ในส่วนของการประมวลผลบนไมโครคอนโทรลเลอร์ได้ทำการเขียนโปรแกรมเชื่อมต่อกับแอนดรอยด์ และรับค่าตัวเลขคำสั่งจากแอนดรอยด์ แสดงผลผ่านจอซีเรียลมอนิเตอร์จะได้ผลดังรูปที่ 4.3 และรูปที่ 4.4



รูปที่ 4.3 หน้าจอซีเรียลมอนิเตอร์แสดงผลเมื่อมีผู้ใช้ล็อคอินเข้าระบบ

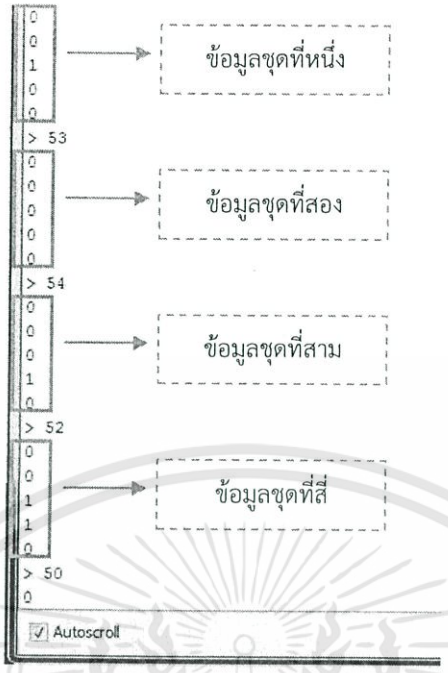


รูปที่ 4.4 หน้าจอซีเรียลมอนิเตอร์แสดงผลรับค่าตัวเลขคำสั่งมาจากแอนดรอยด์

จะเห็นได้ว่าไมโครคอนโทรลเลอร์มีการรับค่าคำสั่งที่ส่งมาจากแอนดรอยด์ได้และนำไปประมวลผลเพื่อสั่งงาน เปิด-ปิด อุปกรณ์ไฟฟ้าในส่วนฮาร์ดแวร์

4.2.2 ส่วนประมวลผลการตรวจสอบสถานะอุปกรณ์

ในส่วนนี้ได้ทำการเขียนโปรแกรมตรวจสอบสถานะของอุปกรณ์ โดยใช้วงจรตรวจสอบสถานะ จะได้ค่า ‘0’ ซึ่งหมายถึงอุปกรณ์ปิดอยู่ กับ ‘1’ ซึ่งหมายถึงอุปกรณ์เปิดอยู่ และส่งค่ากลับไปให้แอนดรอยด์ ดังรูปที่ 4.5



รูปที่ 4.5 หน้าจอซีเรียลมอนิเตอร์แสดงผลค่าสถานะที่ตรวจสอบมา

จากรูปที่ 4.5 จะแสดงค่าที่ได้รับมาจากวงจรตรวจสอบกระแสไฟ เป็นการตรวจสอบสถานะของเครื่องใช้ไฟฟ้าว่ามีการเปิด หรือปิดอยู่ โดยเรียงจาก Device1 ถึง Device5 ตามลำดับ โดยค่า 0 หมายถึงเครื่องใช้ไฟฟ้าอยู่ในสถานะปิด และค่า 1 หมายถึงเครื่องใช้ไฟฟ้าอยู่ในสถานะเปิดทำงาน

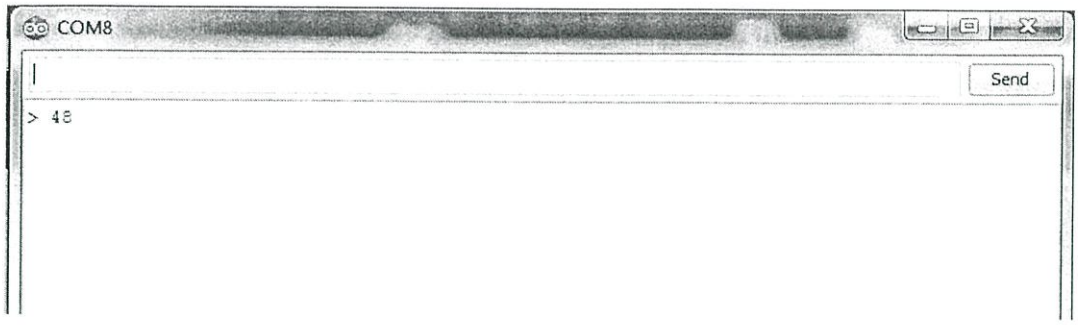
ข้อมูลชุดที่หนึ่ง หมายถึง Device1-off, Device2-off, Device3-on, Device4-off, Device5-off
ข้อมูลชุดที่สอง หมายถึง Device1-off, Device2-off, Device3-off, Device4-off, Device5-off
ข้อมูลชุดที่สาม หมายถึง Device1-off, Device2-off, Device3-off, Device4-on, Device5-off
ข้อมูลชุดที่สี่ หมายถึง Device1-off, Device2-off, Device3-on, Device4-on, Device5-off

4.3 ส่วนฮาร์ดแวร์

ส่วนฮาร์ดแวร์เป็นส่วนที่ต้องรับข้อมูลเพื่อมาควบคุมอุปกรณ์เครื่องใช้ไฟฟ้า และส่งค่าอุณหภูมิที่วัดได้จากเซ็นเซอร์ไปแสดงผลทางหน้าจอ

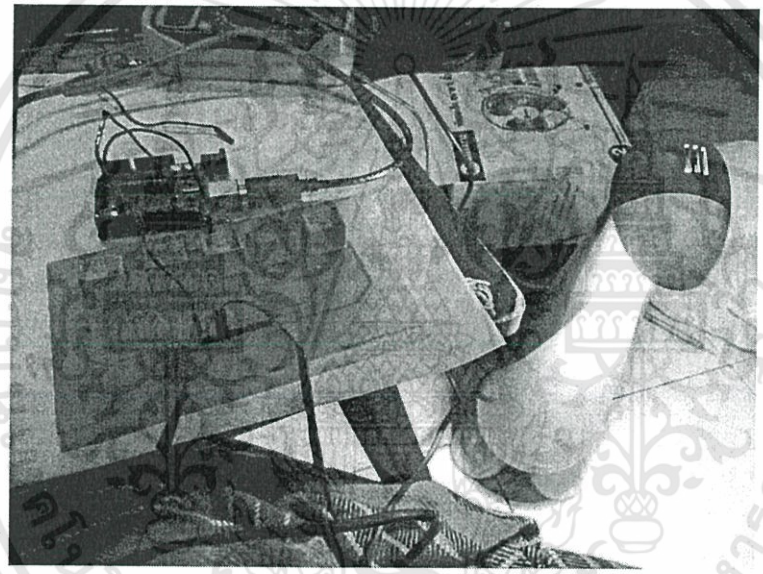
4.3.1 ส่วนควบคุมอุปกรณ์ด้วยไมโครคอนโทรลเลอร์

จากการทดลองพิมพ์ค่าสมมติ ลงในหน้าต่างซีเรียลมอนิเตอร์ของโปรแกรมอาร์ดูโน ที่ได้ทำการเขียนโค้ดโปรแกรมควบคุมรีเลย์ไว้แล้วเพื่อทดลองเฉพาะส่วนควบคุมอุปกรณ์ ได้ผลตัวอย่างดังรูปที่ 4.6และ 4.7



รูปที่ 4.6 ทดลองส่งค่า 0 เพื่อเปิดพอร์ท 1

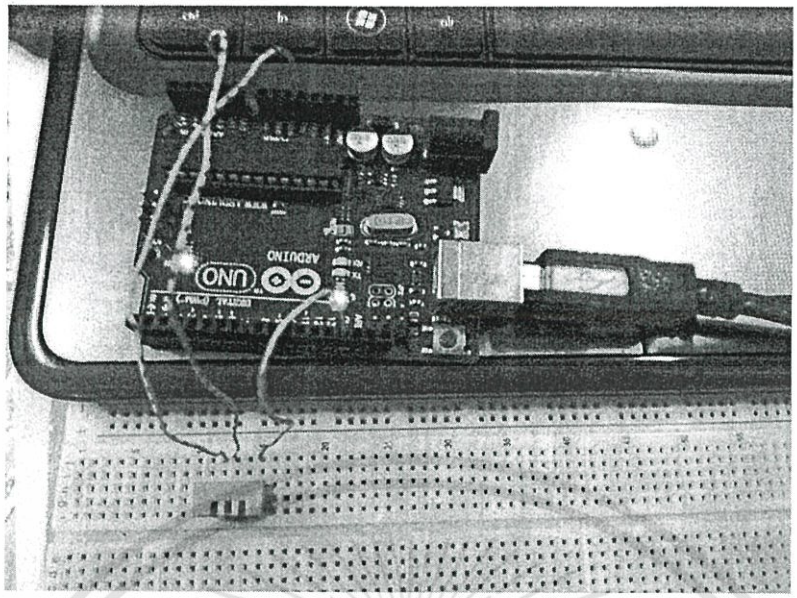
ได้ผลเป็นโคมไฟที่พอร์ท 1 ติด ถูกต้องตามคำสั่ง และผลการทำงานของเครื่องใช้ไฟฟ้า ดังรูปที่ 4.7



รูปที่ 4.7 ผลเครื่องใช้ไฟฟ้าที่เปิดขึ้นตามคำสั่ง

4.3.2 ส่วนเซ็นเซอร์อุณหภูมิและความชื้น

จากการทดสอบการทำงานของเซ็นเซอร์อุณหภูมิและความชื้น สามารถดูผลการทำงานของเซ็นเซอร์ได้จากหน้าจอซีเรียลมอนิเตอร์ของโปรแกรมอาร์ดูโน่ โดยขั้นแรก ทดลองการทำงานของเซ็นเซอร์ในสภาวะอุณหภูมิห้องปกติดังรูป 4.8

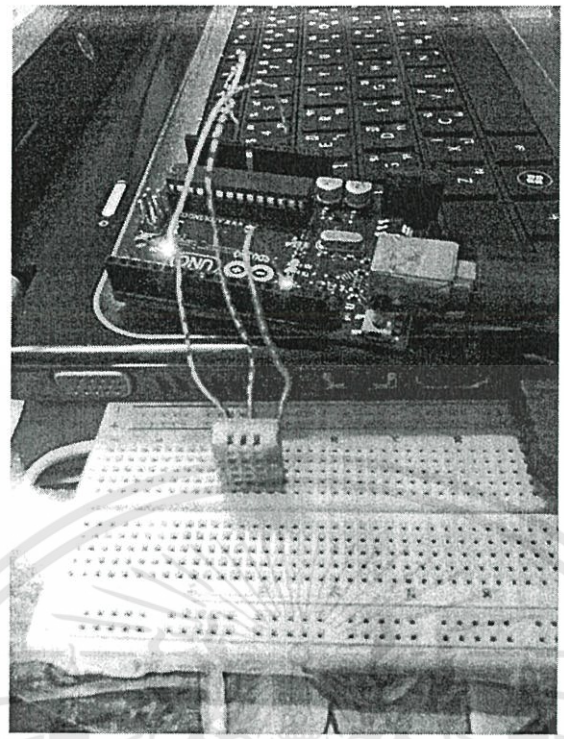


รูปที่ 4.8 ทดสอบการทำงานของเซ็นเซอร์ในสภาวะอุณหภูมิห้องปกติ

ได้ผลการทดลองในหน้าจอซีเรียลมอนิเตอร์ ดังรูปที่ 4.9

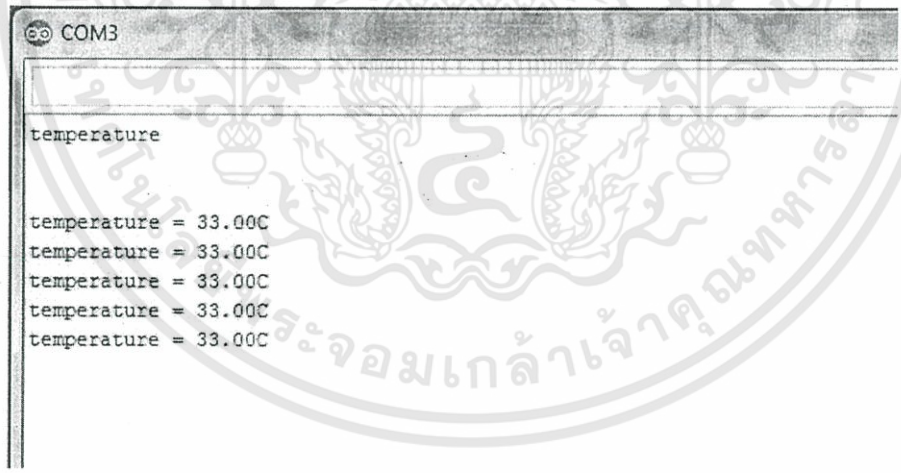
รูปที่ 4.9 หน้าจอซีเรียลมอนิเตอร์แสดงผลในสภาวะอุณหภูมิห้องปกติ

จากนั้นทดสอบการทำงานของเซ็นเซอร์ในสภาวะที่อุณหภูมิสูงกว่าปกติ โดยวางเซ็นเซอร์ที่ช่องระบายความร้อนของโน้ตบุค เพื่อทดสอบการทำงาน และผลที่หน้าจอซีเรียลมอนิเตอร์ดังรูปที่ 4.10



รูปที่ 4.10 ทดสอบการทำงานของเซ็นเซอร์ในสภาวะที่อุณหภูมิสูงกว่าปกติ

ได้ผลบนหน้าจอซีเรียลมอนิเตอร์ ดังรูปที่ 4.11

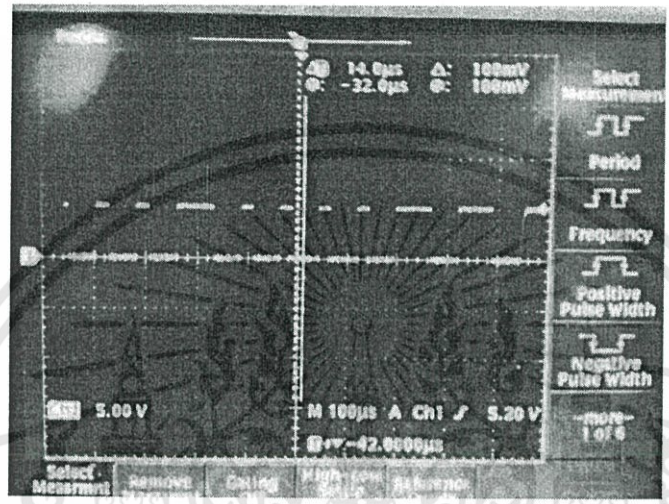


รูปที่ 4.11 หน้าจอซีเรียลมอนิเตอร์แสดงผลในสภาวะอุณหภูมิสูงกว่าปกติ

จากผลการทดลองได้ผลถูกต้องตามที่ควรจะเป็น แต่มีดีเลย์จากการทำงานของตัวเซ็นเซอร์เอง เช่น เมื่อทดลองย้ายจากสภาวะอุณหภูมิห้องปกติ ไปที่สภาวะอุณหภูมิสูงกว่าปกติ (ช่องระบายความร้อนของโน้ตบุค) ต้องรอประมาณหนึ่งนาที เพื่อให้ค่าอุณหภูมิขยับสูงขึ้นจากค่าเดิม ณ ที่สภาวะปกติ หรือเมื่อทดสอบทิ้งเซ็นเซอร์ไว้ในสภาวะอุณหภูมิสูงเป็นระยะเวลานาน เซ็นเซอร์จะมีการสะสมความร้อนไว้มากขึ้น ค่า

อุณหภูมิก็จะสามารถมีค่าสูงขึ้นต่อไปเรื่อยๆ จากเดิมด้วย (สังเกตจากรูปที่ 4.11 เป็นการวางเซ็นเซอร์ไว้ในสภาวะอุณหภูมิสูงในระยะเวลาสั้นๆ แต่ถ้าวางเซ็นเซอร์ไว้ ณ สภาวะนี้นานขึ้น ค่าอุณหภูมิสามารถขึ้นไปถึง 35 องศาเซลเซียส) หรือเมื่อย้ายตัวเซ็นเซอร์สลับตำแหน่งกลับมาที่เดิม ณ อุณหภูมิปกติ จะต้องรอช่วงเวลาหนึ่งจึงจะแสดงผลให้เห็นว่าอุณหภูมิมีระดับลดลง

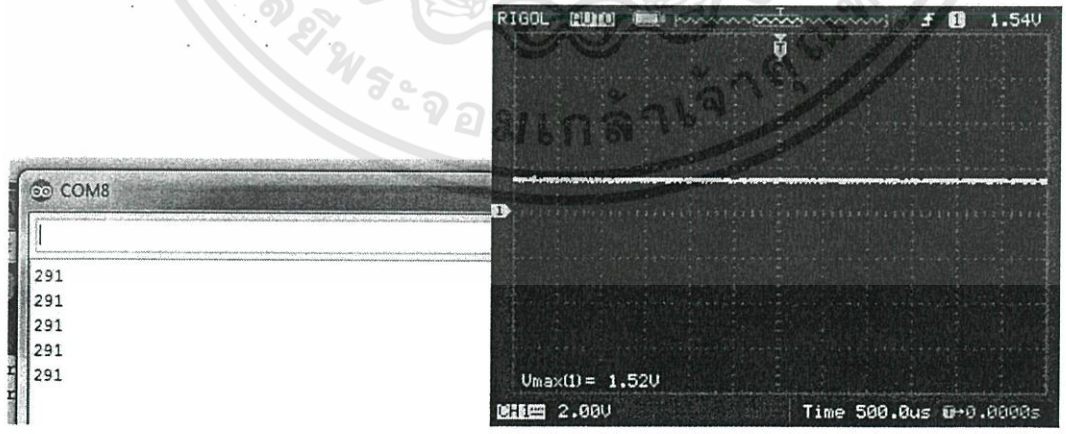
เมื่อทดลองจับสัญญาณที่ส่งออกมาจากขาข้อมูล (pin2) ของตัว DHT11 จะพบว่ามีการส่งข้อมูลออกมาเป็นบิตดิจิตอล 0 และ 1 โดยบิต 0 จะมีช่วงสัญญาณยกระดับสูงที่สั้นกว่าบิต 1 ดังรูปที่ 4.12



รูปที่ 4.12 สัญญาณเอาท์พุทจาก DHT11

4.3.3ส่วนแอลดีอาร์

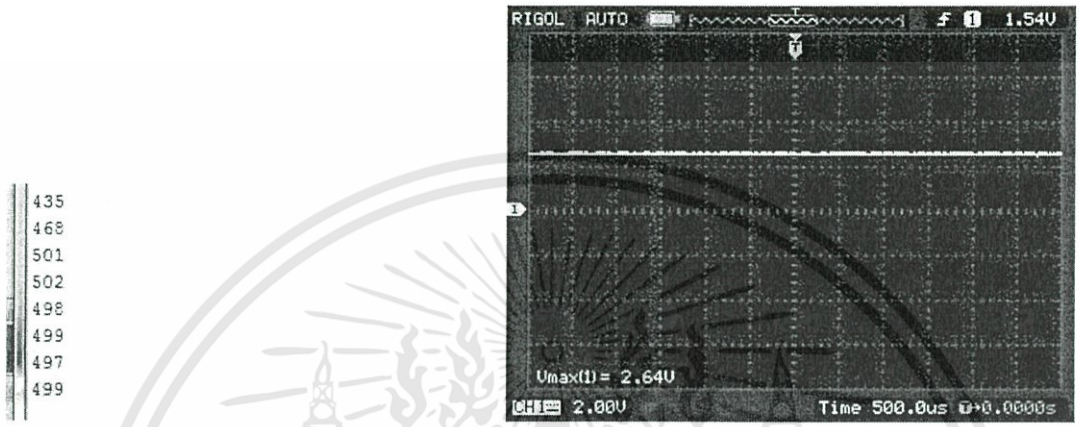
ผลการทำงานของแอลดีอาร์ สามารถวัดได้จากหน้าจอซีเรียลมอนิเตอร์ และหน้าจอสโคป โดยขั้นแรกทดลองวางที่แสงไฟในห้องที่สว่างมาก ดังรูปที่ 4.13



รูปที่ 4.13 ผลการทำงานของแอลดีอาร์เมื่อวางที่แสงไฟในห้องที่สว่างมาก

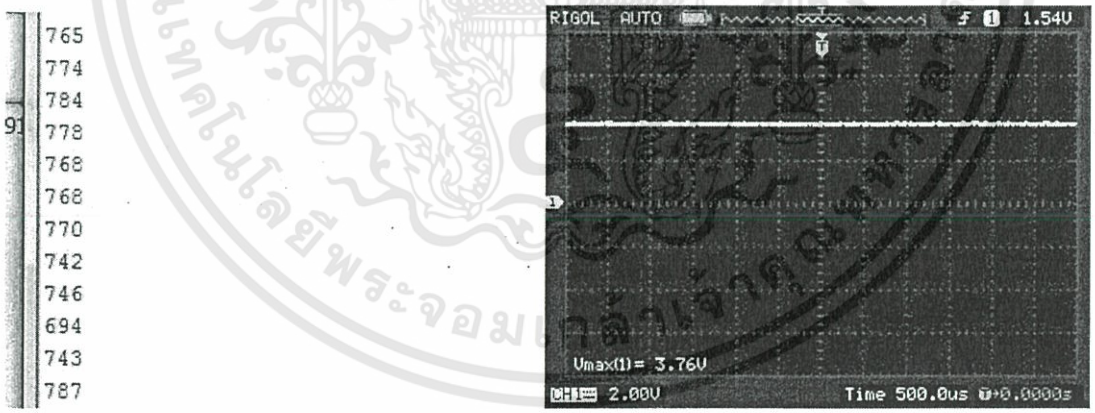
จากรูปที่ 4.13 จะเห็นได้ว่า ค่าของแอลดีอาร์เป็น 291 และมีแรงดันไฟตกคร่อมตัวแอลดีอาร์ 1.52 โวลต์

จากนั้น ทดลองหรี่ไฟลงมา แล้วทำการเก็บผลจากหน้าจอซีเรียลมอนิเตอร์ และหน้าจอสโคป ได้ผลลัพธ์ดังรูปที่ 4.14 นั่นคือมีค่าแอลดีอาร์มากขึ้น และโวลต์ตกคร่อมมากขึ้น



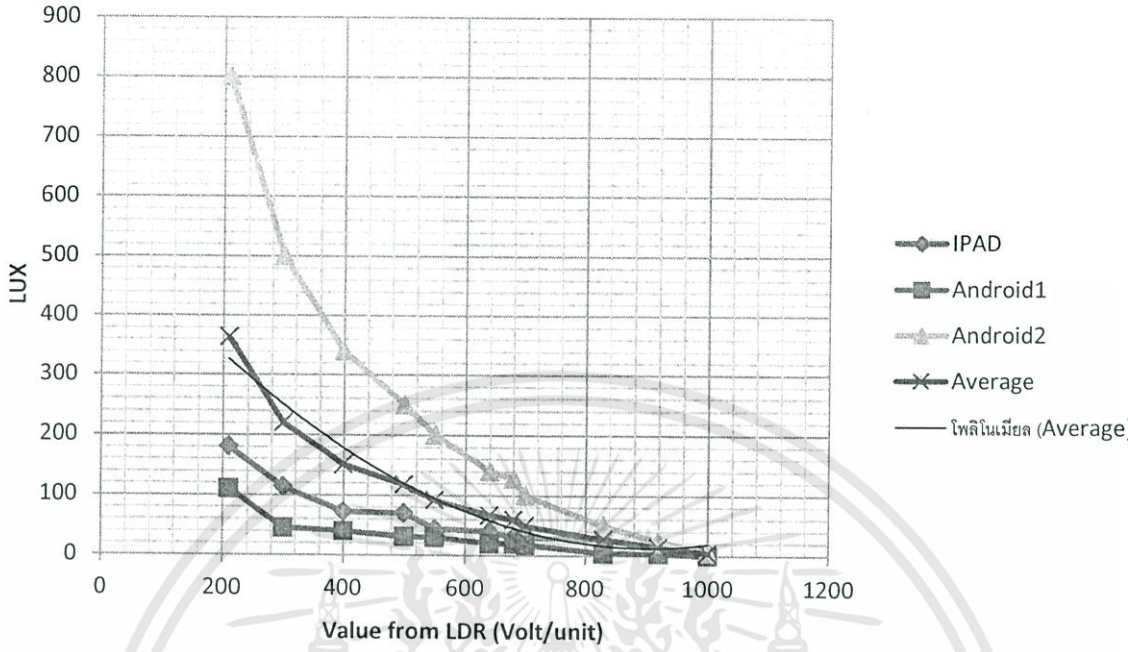
รูปที่ 4.14 ผลการทำงานของแอลดีอาร์เมื่อทดลองหรี่ไฟลง

สุดท้ายทดลองทำการปิดไฟในห้องเหลือเพียงไฟหรี่เล็กๆ ได้ผลการทำงานดังรูปที่ 4.15



รูปที่ 4.15 ผลการทำงานของแอลดีอาร์เมื่อทดลองปิดไฟ

จะเห็นว่าค่าของแอลดีอาร์ และโวลต์ที่ตกคร่อมจะเปลี่ยนแปลงตามระดับความสว่างของแสง จากนั้นได้ทดลองใช้แอพพลิเคชั่น จากอุปกรณ์สามอย่าง ประกอบไปด้วยจากไอแพด (IPAD) แอนดรอยด์เครื่องที่หนึ่ง และแอนดรอยด์เครื่องที่สอง วัดค่าความเข้มแสงเป็นหน่วยลักซ์ (LUX) ซึ่งค่าความสัมพันธ์ระหว่างค่าแรงดันต่อยูนิตที่วัดได้จากขาอนาลอคของอาร์ดูโน่บนหน้าจอซีเรียลมอนิเตอร์ (volt/unit) กับความเข้มแสงในหน่วยของลักซ์ได้ดังรูปที่ 4.16



รูปที่ 4.16 กราฟความสัมพันธ์ระหว่างค่าแรงดันต่อยูนิตจากแอลดีอาร์และความเข้มแสงในหน่วยลักซ์

จากนั้นหาค่าเฉลี่ยของทั้งสาม ได้สมการค่าเฉลี่ยเป็น

$$LUX = 0.0007R^2 - 1.1799R + 545.76 \quad (4.1)$$

นำสมการนี้ ไปใช้หาค่าความเข้มแสงต่อไป โดยค่า R หมายถึงค่าที่ขา Analog ของบอร์ดอาร์ดูโนอ่านได้นั่นเอง

4.4 ระบบรวม

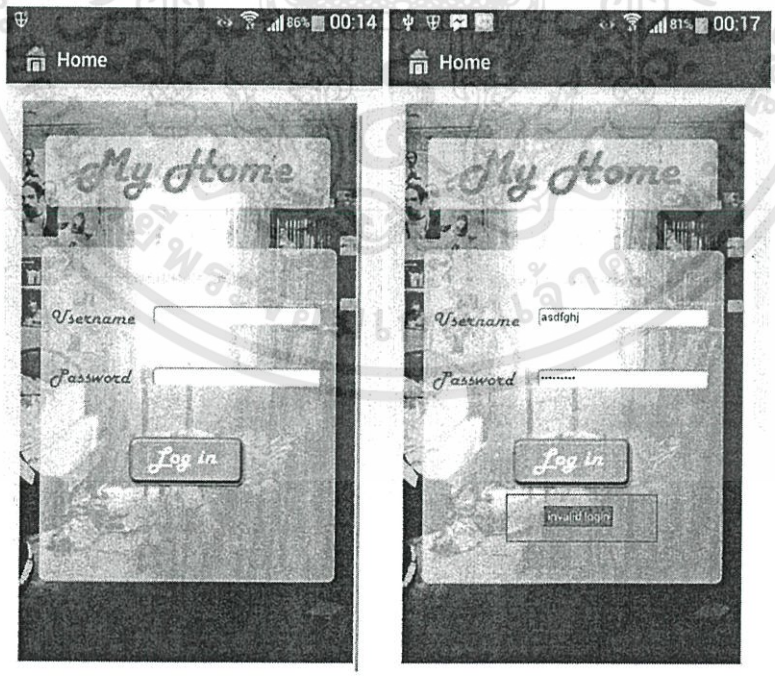
การทำงานของระบบโดยรวมทั้งสามส่วน ได้แก่ ส่วนแอพพลิเคชั่นบนแอนดรอยด์สมาร์ทโฟน ส่วนเชื่อมต่อ และส่วนฮาร์ดแวร์ เชื่อมต่อให้นำมาทำงานได้ร่วมกัน

4.4.1 การสั่งงานในแอพพลิเคชั่นบนแอนดรอยด์

เริ่มจากการล็อคอิน (Log In) เข้าสู่แอพพลิเคชั่นชื่อว่า Home บนแอนดรอยด์ดังรูปที่ 4.17

รูปที่ 4.17 ไอคอนแอพพลิเคชั่นบนแอนดรอยด์

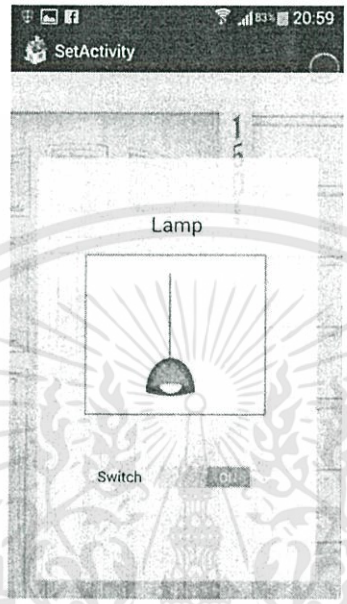
จากนั้นสามารถล็อคอิน โดยต้องกรอกชื่อผู้ใช้ (username) และรหัสผ่าน (password) เพื่อเชื่อมต่อกับฐานข้อมูลได้ และสามารถแจ้งเตือนผู้ใช้ เมื่อใส่ชื่อผู้ใช้และรหัสผ่านผิดพลาดโดยแจ้งเตือนคำว่า “invalid login” (ในกรอบสี่เหลี่ยมรูปที่ 4.18) ที่หน้าแรกของแอพพลิเคชั่นแสดงดังรูปที่ 4.18



รูปที่ 4.18 หน้าลงชื่อผู้ใช้ และเมื่อมีการใส่ชื่อผู้ใช้ และรหัสผ่านผิดพลาด

เมื่อใส่ชื่อผู้ใช้ และรหัสผ่านถูกต้อง จะมีการเข้าสู่หน้าแอพพลิเคชั่นในส่วนเลือกอุปกรณ์ไฟฟ้าที่ต้องการควบคุมต่อไป ดังรูปที่ 4.1 ซึ่งจากรูปจะเห็นได้ว่า ไม่มีเครื่องใช้ไฟฟ้าใดใดที่เปิดอยู่ และมีระดับอุณหภูมิเป็น 29 องศาเซลเซียส

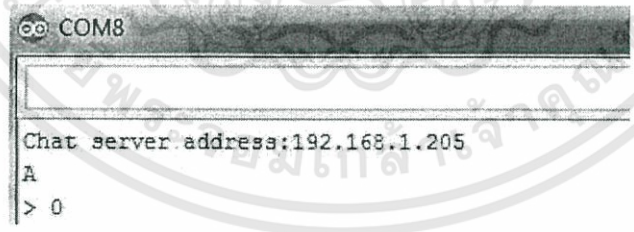
จากนั้นทดสอบสั่งเปิดไฟเพดาน (lamp1) ดังรูปที่ 4.19



รูปที่ 4.19 หน้าแอพพลิเคชั่นเมื่อกดเปิดไฟเพดาน

4.4.2 ส่วนเชื่อมต่อ

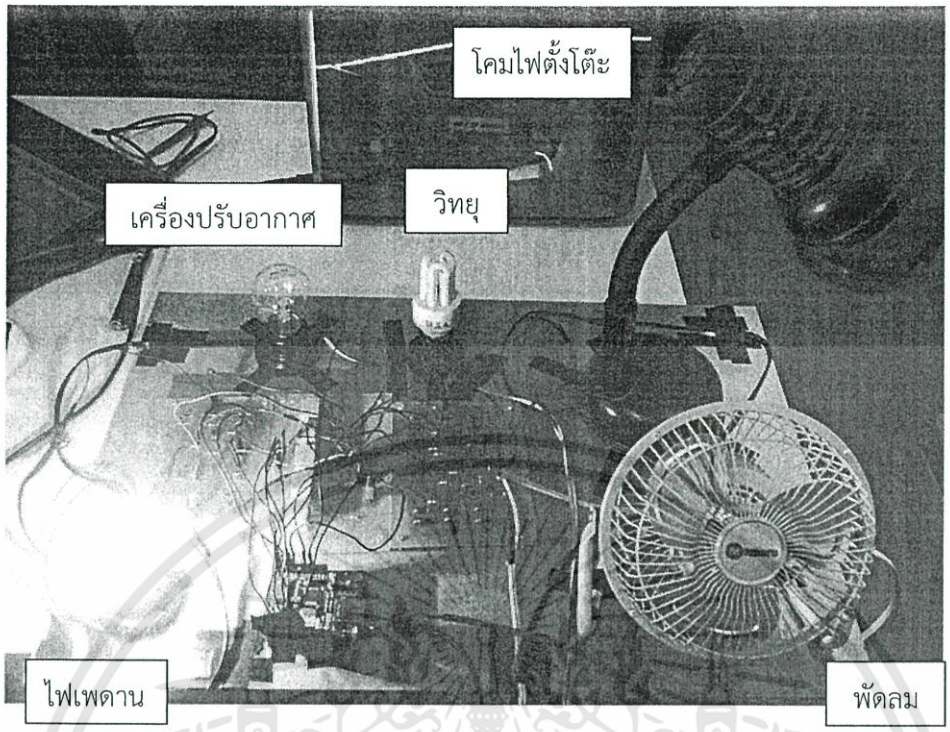
เมื่อมีการสั่งเปิดไฟเพดาน หรือหมายความว่า เป็นการสั่งให้เครื่องใช้ไฟฟ้าพอร์ท 1 เปิด (0) หน้าจอซีเรียลมอนิเตอร์จะรับค่านั้นมา และแสดงผลให้เห็นถึงค่าที่ได้รับมาด้วย ดังรูปที่ 4.20



รูปที่ 4.20 หน้าจอซีเรียลมอนิเตอร์ของระบบเมื่อสั่งให้เครื่องใช้ไฟฟ้าพอร์ท 1 เปิด

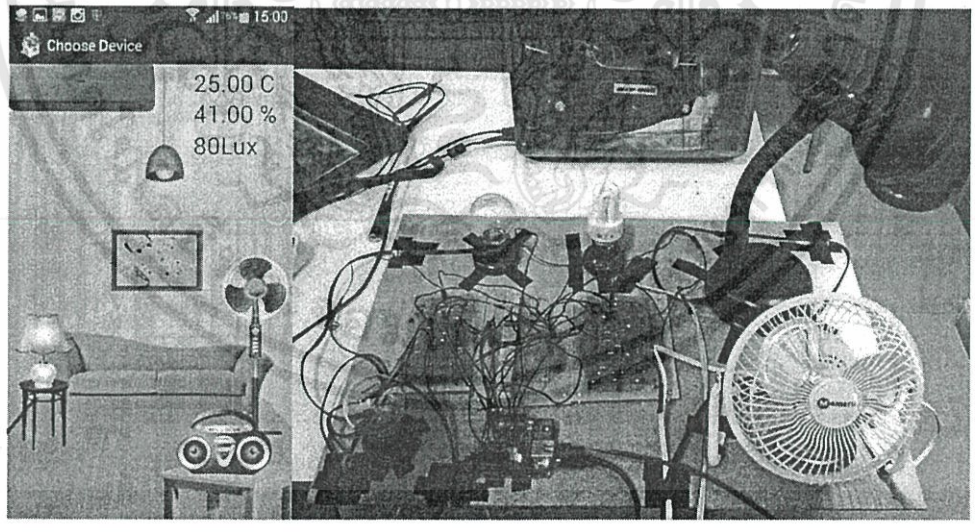
4.4.3 รับข้อมูล เพื่อมาควบคุมอุปกรณ์

เมื่อข้อมูลได้เข้ามาสู่อาร์ดูโนแล้ว จึงเข้าเงื่อนไขที่ตั้งไว้ตามการเขียนโค้ดโปรแกรม เพื่อไปควบคุมรีเลย์ เพื่อการเปิด-ปิด เครื่องใช้ไฟฟ้าต่อไป จากที่การสั่งงานให้เปิดไฟเพดานนั้น จะได้ผลการทำงานดังรูปที่ 4.21

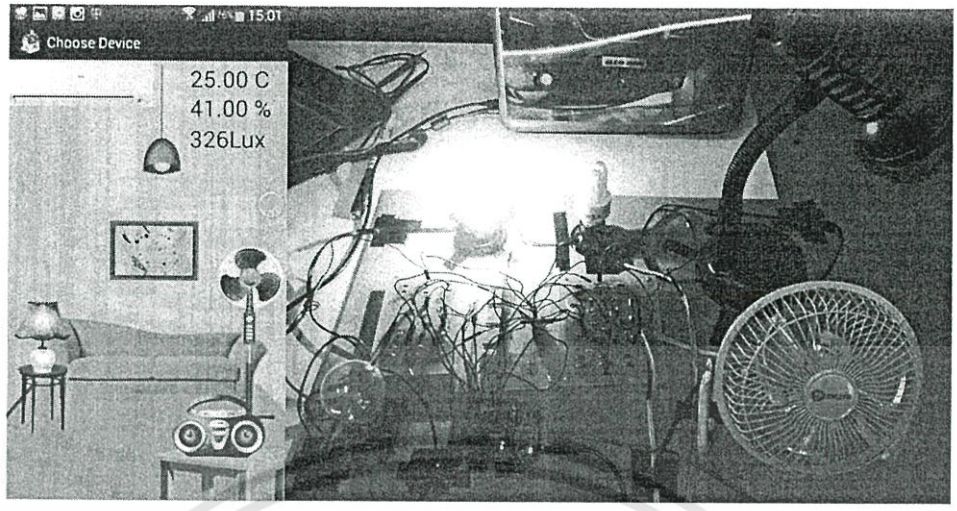


รูปที่ 4.21 ผลการทำงานของอุปกรณ์ไฟฟ้าเมื่อมีการสั่งให้โคมไฟเพดานเปิด

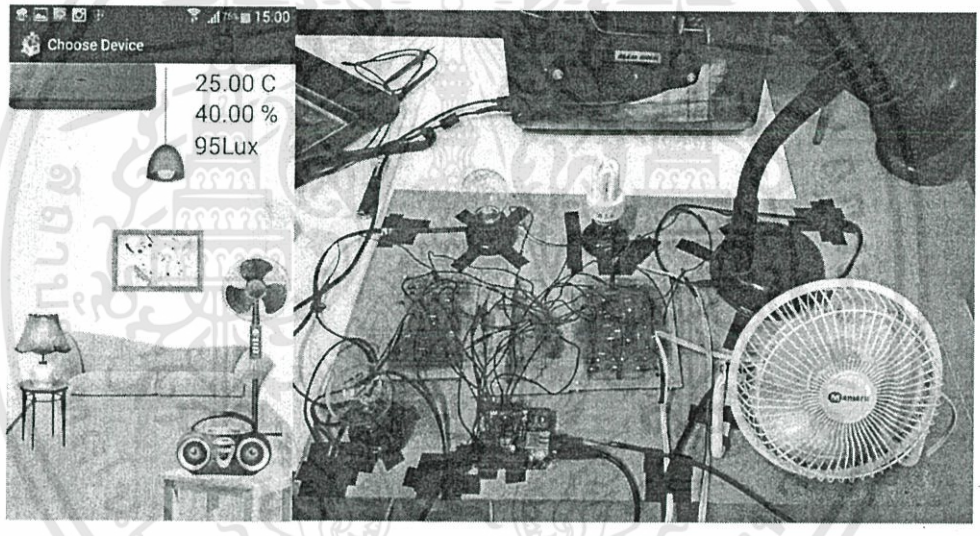
หลังจากนั้น ทดลองสั่งให้เปิด และปิดเครื่องใช้ไฟฟ้าอื่นๆ จนครบทุกพอร์ท จะได้ผลการทำงานดังรูปที่ 4.22 – 4.24



รูปที่ 4.22 ผลการทำงานของอุปกรณ์ไฟฟ้าเมื่อมีการสั่งให้โคมไฟตั้งโต๊ะเปิด

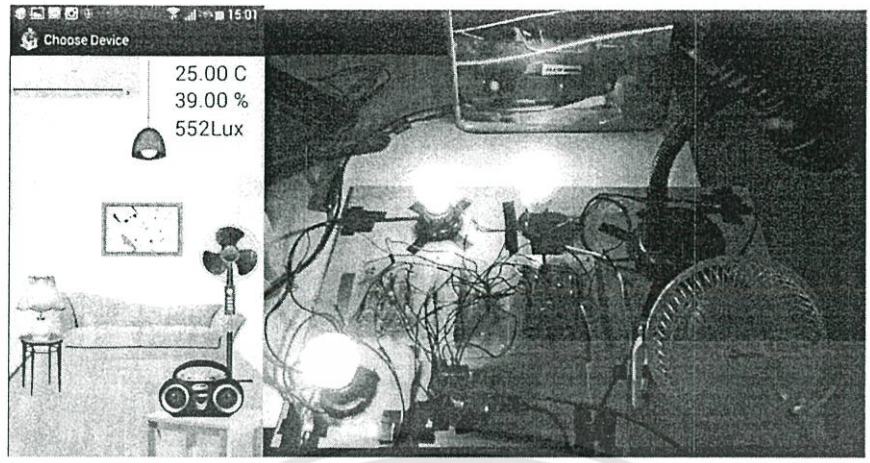


รูปที่ 4.23 ผลการทำงานของอุปกรณ์ไฟฟ้าเมื่อมีการสั่งให้เครื่องปรับอากาศเปิด



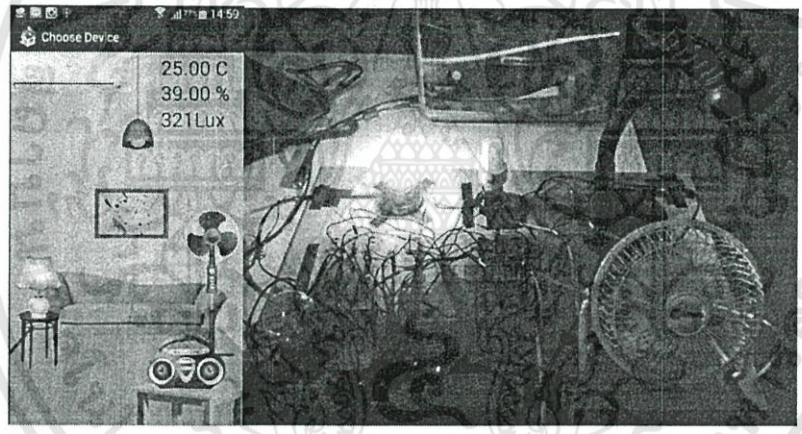
รูปที่ 4.24 ผลการทำงานของอุปกรณ์ไฟฟ้าเมื่อมีการสั่งให้พัดลมเปิด

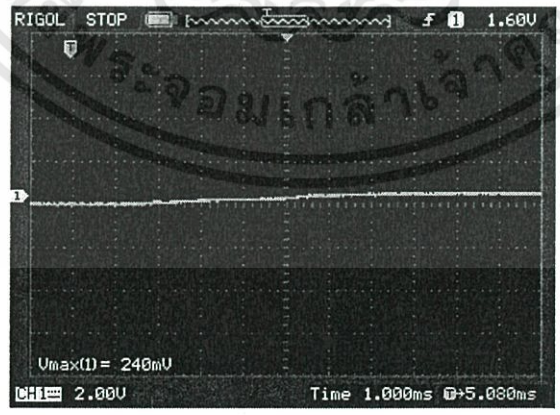
จากนั้นได้ทดลอง สั่งงานเปิดเครื่องใช้ไฟฟ้าทุกตัวทั้งหมดพร้อมกัน ได้ผลการทำงานดังรูปที่ 4.25



รูปที่ 4.25 ผลการทำงานของอุปกรณ์ไฟฟ้าเมื่อมีการสั่งเปิดเครื่องใช้ไฟฟ้าทุกตัวพร้อมกัน

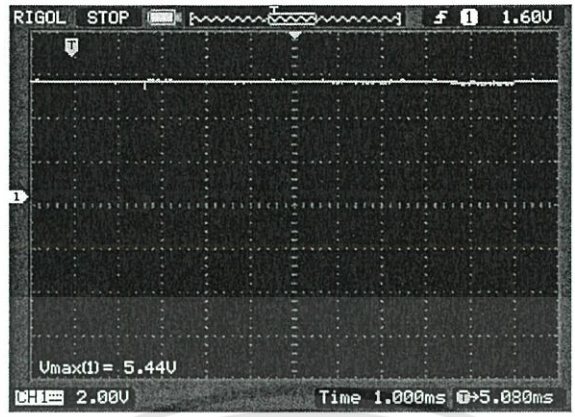
จากนั้นทดลองปิดไฟเพดาน และวิทยุ แล้วทำการวัดแรงดันที่ขาที่ 2, 3, 5, 6 และ 7 ของอาร์ดูโนจากหน้าจอสโคป ได้ผลสัญญาณดังรูปที่ 4.26



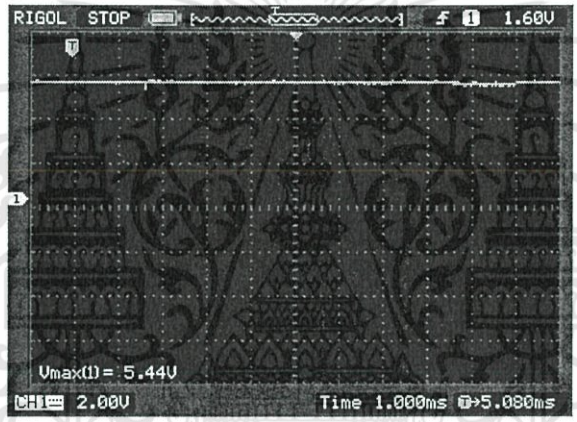


Device1 (Lamp1) = off

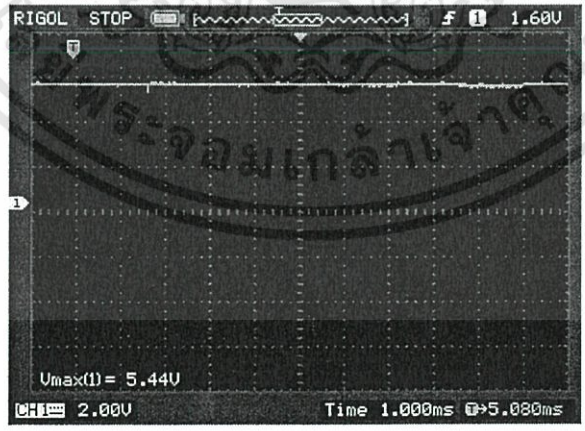
รูปที่ 4.26 ผลการทำงานเมื่อมีการสั่งเปิดเครื่องใช้ไฟฟ้าจำนวน 3 เครื่องพร้อมกัน



Device2 (Lamp2) = on

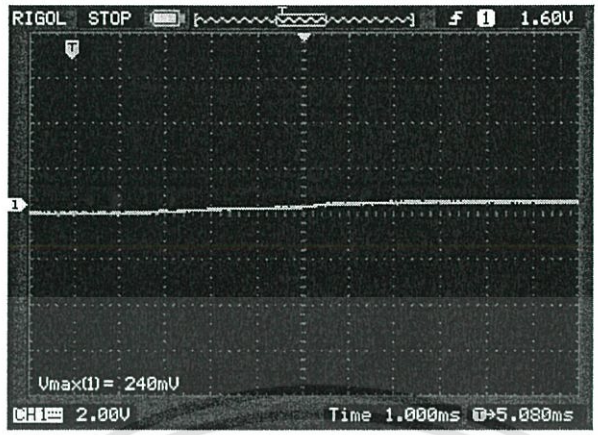


Device3 (Air Conditioner) = on



Device4 (Fan) = on

รูปที่ 4.26 ผลการทำงานเมื่อมีการสั่งเปิดเครื่องใช้ไฟฟ้าจำนวน 3 เครื่องพร้อมกัน (ต่อ)



Device5 (Radio) = off

รูปที่ 4.26 ผลการทำงานเมื่อมีการสั่งเปิดเครื่องใช้ไฟฟ้าจำนวน 3 เครื่องพร้อมกัน (ต่อ)

จากผลที่ได้ในรูปที่ 4.26 จะเห็นได้ว่าเมื่ออุปกรณ์ใดเปิดขาอาร์ดูโนก็จะทำการจ่ายไฟในระดับสูง (5 โวลต์ หรือจากสโคปได้ 5.44 โวลต์ เนื่องจากจ่ายไฟให้บอร์ดอาร์ดูโนเกิน 5 โวลต์เล็กน้อย) เพื่อให้รีเลย์ทำงานแล้วไปจ่ายไฟให้อุปกรณ์ไฟฟ้า ส่วนขาใดที่สั่งงานปิด อาร์ดูโนก็จะไม่จ่ายไฟให้ (หรือจากสโคปคือ 240 มิลลิโวลต์ เกิดความคลาดเคลื่อนเล็กน้อยจากไฟที่จ่ายออกจากขาอาร์ดูโนเอง) ทำให้ไม่มีกระแสไฟไหลเข้ารีเลย์ รีเลย์ก็จะไม่ทำงาน โดยระบบทั้งหมดมีความสัมพันธ์ดังตารางที่ 4.1

ตารางที่ 4.1 ความสัมพันธ์ของตัวเลข ตัวแปร พอร์ทและชนิดของเครื่องใช้ไฟฟ้าในแต่ละส่วน

ชนิดของเครื่องใช้ไฟฟ้า	ตัวแปรในส่วนของการรับส่ง (เปิด -ปิด)	พอร์ทของวงจร	หมายเหตุ
ไฟติดเพดาน	0 – 1	1	แทนด้วยหลอดไฟ bulb ด้านซ้ายล่าง
โคมไฟตั้งโต๊ะ	2 – 3	2	-
เครื่องปรับอากาศ	4 – 5	3	แทนด้วยหลอดไฟ bulb ด้านซ้ายบน
พัดลม	6 - 7	4	-
วิทยุ	8 - 9	5	แทนด้วยหลอดฟลูออเรสเซนส์

บทที่ 5

สรุปผลและข้อเสนอแนะ

5.1 สรุปผล

จากการทดสอบการทำงานของระบบควบคุมอุปกรณ์ไฟฟ้าภายในบ้านระยะไกล สามารถควบคุมอุปกรณ์ไฟฟ้าได้โดยใช้บอร์ดอาร์ดูโน รุ่น UNO R3 ที่มีไมโครคอนโทรลเลอร์เอวีอาร์ ATmega328p-pu เป็นอุปกรณ์ควบคุมการทำงานของอุปกรณ์ไฟฟ้า สามารถให้ผู้ใช้งานสั่งการควบคุมผ่านแอพพลิเคชั่นบนแอนดรอยด์ และแสดงอุณหภูมิ ความชื้น และความสว่างของห้องในขณะนั้นๆ ให้แก่ผู้ใช้งานบนหน้าจอในแอพพลิเคชั่นได้

ในส่วนของวงจรที่ใช้ในการควบคุมการเปิด-ปิดอุปกรณ์ไฟฟ้า ทำงานโดยการรับอินพุทจากบอร์ดอาร์ดูโน เพื่อควบคุมรีเลย์ให้ทำการเปลี่ยนสถานะอุปกรณ์ไฟฟ้า และมีการตรวจสอบสถานะการเปิด-ปิดของอุปกรณ์ไฟฟ้าจากวงจรตรวจสอบกระแสไฟ โดยหน้าจอผู้ใช้งานบนแอพพลิเคชั่นจะมีการเรียกค่าใหม่เรื่อยๆ เพื่อแสดงถึงสถานะของเครื่องใช้ไฟฟ้าปัจจุบัน เช่นเดียวกับค่าอุณหภูมิ ค่าความชื้นสัมพัทธ์ และความสว่างของห้องจากตัวเซนเซอร์ DHT11 และแอลดีอาร์ ซึ่งหน้าจอควบคุมกลางนี้จะทำหน้าที่เป็นศูนย์กลางในการควบคุมการเปิด-ปิดเครื่องใช้ไฟฟ้าทั้ง 5 ชนิด ได้แก่ ไฟเพดาน โคมไฟตั้งโต๊ะ พัดลม วิทยุ และ เครื่องปรับอากาศ

ในส่วนของทางแอพพลิเคชั่นบนแอนดรอยด์นั้น สามารถเข้าถึงแอพพลิเคชั่นได้จากไอคอนรูปบ้าน โดยมีชื่อแอพพลิเคชั่นว่าโฮม (Home) จากนั้นเมื่อกดเข้าแอพพลิเคชั่นแล้ว จะมีการให้กรอกชื่อผู้ใช้ และรหัสผ่าน เพื่อทำการยืนยันตัวตน และเป็นการรักษาความปลอดภัยให้แก่ระบบ จากนั้นจึงเข้าสู่หน้าควบคุมกลาง เมื่อผู้ใช้ต้องการเปิด-ปิดเครื่องใช้ไฟฟ้าใด ก็สามารถกดเลือกรูปเครื่องใช้ไฟฟ้านั้นเพื่อเข้าไปสั่งการเปิด-ปิดได้ สำหรับการเชื่อมต่อแอนดรอยด์แอพพลิเคชั่นกับบอร์ดอาร์ดูโนนั้นใช้อาร์ดูโนอีเธอร์เน็ตชิลด์เป็นอุปกรณ์เสริมเชื่อมต่อผ่านทาง TCP Socket โดยใช้สาย RJ- 45 ในการเชื่อมต่อ

5.2 ข้อเสนอแนะ

5.2.1 สามารถประยุกต์ใช้กับอุปกรณ์ไฟฟ้าได้หลากหลายชนิดมากยิ่งขึ้น

5.2.2 สามารถเข้าสู่ระบบโดยการลงชื่อผู้ใช้งาน ได้มากกว่า 1 ชื่อผู้ใช้งาน

5.2.3 สามารถพัฒนาอุปกรณ์และวงจรให้สามารถเคลื่อนย้าย และทนต่อสภาพแวดล้อมได้สะดวกยิ่งขึ้น เช่น บรรจุอุปกรณ์ และวงจรลงในบรรจุภัณฑ์ที่เหมาะสม

5.2.4 สามารถนำอุปกรณ์เพิ่มความปลอดภัยทางไฟฟ้ามาประยุกต์ใช้กับวงจรที่สามารถตัดไฟเมื่อมีการลัดวงจรเกิดขึ้น เช่น เบรกเกอร์ (Breaker)

บรรณานุกรม

[1] Wannika. "บทที่ 4 แนะนำหลักการเขียนโปรแกรมเบื้องต้น." http://nwannika.tripod.com/java/Chapter4.htm . (สืบค้นวันที่ : 10 มิถุนายน 2556)

[2] ดวงพร เพ็รช์แบน. "ระบบปฏิบัติการแอนดรอยด์ คืออะไร." http://potinimi.blogspot.com/2013/02/1_16.html. (สืบค้นวันที่ : 10 มิถุนายน 2556)

[3] จักรกฤษณ์ แร่ทอง. "รู้จัก XML เบื้องต้น(มากๆ)" http://www.moe.go.th/moe/th/news/detail.php?NewsID=8952&Key=itnews (สืบค้นวันที่ : 10 มิถุนายน 2556)

[4] "พื้นฐานไมโครคอนโทรลเลอร์ด้วย Arduino ." http://www.thainetbeans.com/arduino/start/start.php . (สืบค้นวันที่ : 10 มิถุนายน 2556)

"บอร์ด arduino."www.ee.buu.ac.th/.../บอร์ด%20Arduino%20Duemilanove.docx (สืบค้นวันที่ : 10 มิถุนายน 2556)

[5] "RJ-45." http://fourcro.blogspot.com/2008/12/rj-45-utp-2-1.html. (สืบค้นวันที่ : 10 ตุลาคม 2556)

[6] "MAC Address คืออะไร." http://ict.scphc.ac.th/?p=512. (สืบค้นวันที่ : 10 ตุลาคม 2556)

[7] ชวลิต ทินกรสูติบุตร และทีมงาน ThaiCERT . "ความรู้พื้นฐานเกี่ยวกับ โปรโตคอล TCP/IP." http://www.tnetsecurity.com/content_basic/tcp_ip_knowledge.php (สืบค้นวันที่ : 10 ตุลาคม 2556)

[8] "ATmega168/328-Arduino Pin Mapping." http://arduino.cc/en/Hacking/PinMapping168. (สืบค้นวันที่ : 10 มิถุนายน 2556)

[9] "รีเลย์." http://th.wikipedia.org/wiki/%E0%B8%A3%E0%B8%B5%E0%B9%80%E0%B8%A5%E0%B8%A2%E0%B9%8C (สืบค้นวันที่ : 12 มิถุนายน 2556)

"หน่วยที่ 9 รีเลย์." http://kpp.ac.th/elearning/elearning3/book-09.html. (สืบค้นวันที่ : 12 มิถุนายน 2556)

[10] Mountain "A" . “การใช้งาน DHT11 Humitdity and Temperature Sensor กับบอร์ด Arduino.” http://www.arduitronics.com/article/%E0%B8%81%E0%B8%B2E0%B8%A3%E0%B9%83%E0%B8%8A%E0%B9%89%E0%B8%87%E0%B8%B2%E0%B8%99-dht11-humitdity-and-temperature-sensor-%E0%B8%81%E0%B8%B1%E0%B8%9A%E0%B8%9A%E0%B8%AD%E0%B8%A3%E0%B9%8C%E0%B8%94-arduino. (สืบค้นวันที่ : 4 กรกฎาคม 2556)

[11] “LDR : Light Dependent Resister.” http://www.mwit.ac.th/~ponchai/CAI_electronics/image/LDR.HTM (สืบค้นวันที่ : 10 ตุลาคม 2556)

ภาคผนวก ก
โค้ดส่วนประมวลผลหน้าต่างการเข้าสู่ระบบ (JAVA)

```
package com.example.project;
import android.R.string;
import android.os.Bundle;
import android.app.Activity;
import android.content.Intent;
import android.text.Editable;
import android.util.Log;
import android.view.Menu;
import android.view.View;
import android.view.View.OnClickListener;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Switch;
import android.widget.TextView;

public class MainActivity extends Activity {
 static Button button2;
    @Override
    protected void onCreate(Bundle savedInstanceState)
    {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.login);

        final EditText txtUser = (EditText)findViewById(R.id.editText1);

        final EditText txtPass = (EditText)findViewById(R.id.editText2);

       Button button = (Button) findViewById(R.id.button1);

       button.setOnClickListener(new OnClickListener()

       {


            @Override
            public void onClick(View v)
            {
                  // TODO Auto-generated method stub

            String gtUser = txtUser.getText().toString();


            String gtPass =  txtPass.getText().toString();

                  String strUser = "rces";
                  String strPass = "1234";

                  if (strUser.equals(gtUser) && strPass.equals(gtPass) )
                  {

                        Intent i = new Intent(getApplicationContext(),
ChooseActivity.class) ; // next to choose page
                        startActivity(i) ;
                  }

                  else
                  {
                        TextView invalid_text =
(TextView)findViewById(R.id.invalid);
                        invalid_text.setText("invalid username or
password");
                  }
            }
      });
    }
```

```
    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        // Inflate the menu; this adds items to the action bar if it is
present.
        getMenuInflater().inflate(R.menu.main, menu);
        return true;
    }
}
```

ภาคผนวก ข
โค้ดส่วนแสดงผลหน้าต่างการเข้าสู่ระบบ

```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="@drawable/bg"
    android:paddingBottom="@dimen/activity_vertical_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    tools:context=".login" >

    <FrameLayout
        android:id="@+id/frameLayout2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginTop="144dp" >
    </FrameLayout>

    <EditText
        android:id="@+id/editText1"
        android:layout_height="30dp"
        android:layout_width="175dp"
        android:layout_marginTop="190dp"
        android:layout_marginLeft="150dp"
        android:background="#FFFFFF"
        android:ems="10" >

        <requestFocus />
    </EditText>

    <EditText
        android:id="@+id/editText2"
         android:layout_height="30dp"
        android:layout_width="175dp"
        android:layout_marginTop="250dp"
        android:layout_marginLeft="150dp"
        android:background="#FFFFFF"
        android:ems="10"
        android:inputType="textPassword" />

    <Button
        android:id="@+id/button1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignLeft="@+id/editText2"
        android:layout_below="@+id/editText2"
        android:layout_marginTop="17dp"
        android:background="#CDB38B"
        android:text="Log in"

        />



  <TextView
        android:id="@+id/invalid"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignRight="@+id/editText2"
        android:layout_below="@+id/button1"
        android:layout_marginRight="50dp"
        android:layout_marginTop="26dp"
        android:background="#FF0000" />

</RelativeLayout>
```

ภาคผนวก ค
โค้ดส่วนประมวลผลหน้าต่างสำหรับเลือกอุปกรณ์ไฟฟ้า (JAVA)

```
package com.example.project;

import java.io.DataInputStream;
import java.io.DataOutputStream;
import java.io.IOException;
import java.net.Socket;
import java.net.UnknownHostException;
import java.util.regex.Matcher;
import java.util.regex.Pattern;

import com.example.project.SetActivity.connect;
import com.example.project.SetActivity.send;

import android.os.AsyncTask;
import android.os.Bundle;
import android.os.Handler;
import android.app.Activity;
import android.content.Intent;
import android.util.Log;
import android.view.Menu;
import android.view.View;
import android.view.View.OnClickListener;
import android.widget.Button;
import android.widget.ImageButton;
import android.widget.LinearLayout;
import android.widget.RelativeLayout;
import android.widget.TextView;

public class ChooseActivity extends Activity {


	static String lampstatus;
	static String lamp2status;
	static String airstatus;
	static String fanstatus;
	static String radiostatus;


	volatile boolean stopworker;
	Thread workerThread;

	static Socket s;
	static DataOutputStream dataOut;
	static DataInputStream dataIn;
	byte[] readBuffer;
	int readBufferPosition;
	int countdown = 10;

	@Override
	protected void onCreate(Bundle savedInstanceState)
	{
		super.onCreate(savedInstanceState);
		setContentView(R.layout.choose);


			new connect().execute();

			try {
				Thread.sleep(2000);
	//delay(ms)
			} catch (InterruptedException e1) {
				// TODO Auto-generated catch block
				e1.printStackTrace();
			}
```

```
                    //new send().execute("A");                                    //
send "A" to arduino
                    beginupdate();
                    recieveData ();

                    ImageButton button = (ImageButton)
findViewById(R.id.imageButton4);  //Lamp


                    button.setOnClickListener(new OnClickListener() {

                    @Override
                    public void onClick(View v) {

                              try {
                                    Thread.sleep(500);
                              } catch (InterruptedException e) {
                                    // TODO Auto-generated catch block
                                    e.printStackTrace();
                              }

                      Intent i = new Intent(getApplicationContext(),
SetActivity.class) ;

                     i.putExtra("lampstatus",lampstatus);                         //
send parameter i to next page

                     startActivity(i) ;


               }

         });


                    ImageButton button2 = (ImageButton)
findViewById(R.id.imageButton3); //lamp2


                    button2.setOnClickListener(new OnClickListener() {

                    //     @Override
                         public void onClick(View v) {
                               try {
                                    Thread.sleep(500);

                               } catch (InterruptedException e) {
                                    // TODO Auto-generated catch block
                                    e.printStackTrace();
                               }
                            Intent i2 = new Intent(getApplicationContext(),
Setlamp2Activity.class) ;

                               i2.putExtra("lamp2status",lamp2status);
                               startActivity(i2) ;
                     }

                });


                   ImageButton button3 = (ImageButton)
findViewById(R.id.imageButton2); //Radio
```

```
                button3.setOnClickListener(new OnClickListener() {

                //      @Override
                        public void onClick(View v) {
                                try {
                                        Thread.sleep(500);

                                } catch (InterruptedException e) {
                                        // TODO Auto-generated catch block
                                        e.printStackTrace();
                                }

                                Intent i3 = new
Intent(getApplicationContext(), SetradioActivity.class) ;

                                i3.putExtra("radiostatus",radiostatus);
                                startActivity(i3) ;
                        }

                });


                ImageButton button4 = (ImageButton)
findViewById(R.id.imageButton1); //Fan


                button4.setOnClickListener(new OnClickListener() {

                //      @Override
                        public void onClick(View v) {

                                try {
                                        Thread.sleep(500);

                                } catch (InterruptedException e) {
                                        // TODO Auto-generated catch block
                                        e.printStackTrace();
                                }

                                Intent i4 = new Intent(getApplicationContext(),
SetfanActivity.class) ;
                                i4.putExtra("fanstatus",fanstatus);
                                startActivity(i4) ;
                        }

                });


                ImageButton button5 = (ImageButton)
findViewById(R.id.imageButton5); //Air


                button5.setOnClickListener(new OnClickListener() {

                //      @Override
                        public void onClick(View v) {

                                try {
                                        Thread.sleep(500);

                                } catch (InterruptedException e) {
                                        // TODO Auto-generated catch block
                                        e.printStackTrace();
                                }
```

```
                        Intent i5 = new Intent(getApplicationContext(),
SetairActivity.class) ;
                        i5.putExtra("airstatus",airstatus);
                        startActivity(i5) ;
                    }

                });

        }

        @Override
        public boolean onCreateOptionsMenu(Menu menu) {
                // Inflate the menu; this adds items to the action bar if it is
present.
                getMenuInflater().inflate(R.menu.choose, menu);
                return true;
        }

        @Override
        public void onBackPressed (){
                try {

                        s.close();                      //Socket has been close

                } catch (IOException e) {
                        // TODO Auto-generated catch block
                        e.printStackTrace();
                }
                super.onBackPressed();
        }

        @Override
        public void onResume (){

                Log.v("msg", "yeh");

                new send().execute("A");                        // send
"A" to arduino

                recieveData ();

                super.onResume();


        }

        class send extends AsyncTask<String, Void, String> {

                @Override
                protected String doInBackground(String... params) {
                // separate process
                        // TODO Auto-generated method stub


                        try {
                                dataOut.writeChar(params[0].toCharArray()[0]);

                        } catch (IOException e) {
                                // TODO Auto-generated catch block
                                e.printStackTrace();
                        }
                        return null;
                }
```

```
	}

	class connect extends AsyncTask<String, Void, String> {

		@Override
		protected String doInBackground(String... params) {
			// TODO Auto-generated method stub


			try {
				s = new Socket("192.168.1.205", 3216);
		//("IP number",port)
				dataOut = new
DataOutputStream(s.getOutputStream());
				dataIn = new DataInputStream(s.getInputStream());


			} catch (UnknownHostException e) {
				// TODO Auto-generated catch block
				e.printStackTrace();
			} catch (IOException e) {
				// TODO Auto-generated catch block
				e.printStackTrace();
			}
			return null;
		}

	}

	void recieveData()
	{
		final Handler handler = new Handler();


		stopworker = false;
		readBufferPosition = 0;
	    readBuffer = new byte[1024];
	    final byte delimiter = 35;
		workerThread = new Thread(new Runnable(){

			@Override
			public void run() {
				// TODO Auto-generated method stub

				// Thread begin

				while(!Thread.currentThread().isInterrupted() &&
!stopworker){

					try{

						int bytesAvailable =
dataIn.available();

						if(bytesAvailable > 0)

						{
							byte[] packetBytes = new
byte[bytesAvailable];

							dataIn.read(packetBytes);
							for(int
i=0;i<bytesAvailable;i++)
{
```

```
                                byte b = packetBytes[i];

                                if(b == delimiter)
                                {
                                      byte[] encodedBytes = new
byte[readBufferPosition];
                                      System.arraycopy(readBuffer, 0,
encodedBytes, 0, encodedBytes.length);
                                      final String data = new
String(encodedBytes, "US-ASCII");

                                      readBufferPosition = 0;
        //prepare for read new data
                                    handler.post(new Runnable()
                                    {
                                        public void run()
                                        {
                                            Log.v("msg2", data);
                                            update(data);

                                        }
                                    });
                                }
                                else
                                {
                                    readBuffer[readBufferPosition++] = b;
// read packetBytes(data in) continuous
                                }
                            }

                                }

                            }

                            catch(IOException error){

                                stopworker = true;

                            }
                        }

                    }

        });

        workerThread.start();
    }

    public void beginupdate(){

        Thread thread1 = new Thread(new Runnable(){

            @Override
            public void run() {

                while(true)   {
                    if(countdown > 0){
                        countdown--;
                    }else{
                        new send().execute("A");
```

```
                                        Log.v("msg2", "update");
                                        countdown = 10;
                                }
                                try {
                                        Thread.sleep(1000);
                                } catch (InterruptedException e) {
                                        // TODO Auto-generated catch block
                                        e.printStackTrace();
                                }
                        }

                }
        });
        thread1.start();
    }

    void request(String msg){
        try {
                Thread.sleep(500);
        } catch (InterruptedException e1) {
                // TODO Auto-generated catch block
                e1.printStackTrace();
        }
        Log.v("msg2", msg);
        new send().execute("A");
        countdown = 5;
    }

    void update(String data){

        String[] datasplit = data.split(":");
        //Check array length
        if(datasplit.length != 8){
                request("Split error Check again");
        }else{

        String temp = datasplit[2];
        String lamp = datasplit[3];
        String lamp2 = datasplit[4];
        String fan = datasplit[6];
        String radio = datasplit[7];
        String air = datasplit[5];
        String humidity = datasplit[1];
        String intensity = datasplit[0];
        int inten = 0;
        try{
                inten = Integer.parseInt(intensity);
                TextView temp_text = (TextView)findViewById(R.id.temp);
                TextView humidity_text =
(TextView)findViewById(R.id.humidity);

                //Validate temp
                Pattern r = Pattern.compile("^[0-9]+.[0-9]+ C$");
                Matcher m = r.matcher(temp);

                //Validate humidity
                Pattern r2 = Pattern.compile("^[0-9]+.[0-9]+ %$");
                Matcher m2 = r2.matcher(humidity);

                if(!m.find()){
                        request("Temp error Check again");
                }
                else if(!m2.find()){
```

```
                        request("Humidity error Check again");
                }
                else{
                        temp_text.setText(temp);
                        humidity_text.setText(humidity);

                        RelativeLayout
layout=(RelativeLayout)findViewById(R.id.bg);


                        if (inten >= 200 && inten <= 360){

        layout.setBackground(getResources().getDrawable(R.drawable.bg2));

                        }

                        if (inten >= 361 && inten <= 520){

        layout.setBackground(getResources().getDrawable(R.drawable.bg21));

                        }

                        if (inten >= 521 && inten <= 680){

        layout.setBackground(getResources().getDrawable(R.drawable.bg22));

                        }

                        if (inten >= 681 && inten <= 840){

        layout.setBackground(getResources().getDrawable(R.drawable.bg23));

                        }

                        if (inten >= 841 && inten <= 1000){

        layout.setBackground(getResources().getDrawable(R.drawable.bg24));

                        }


                        //Check Lamp
                        ImageButton lamp_pic =
(ImageButton)findViewById(R.id.imageButton4);
                        if(lamp.equals("0")){

        lamp_pic.setImageResource(R.drawable.lamp_b);
                                lampstatus = "0";

                                Log.v("msg", "Lamp off");

                        }else if(lamp.equals("1")){
                                lamp_pic.setImageResource(R.drawable.lamp);

                                lampstatus = "1";

                                Log.v("msg", "Lamp on");
                        }else{
                                request("lamp error Check again");
                        }
```

```
                        //Check Lamp2
                        ImageButton lamp2_pic =
(ImageButton)findViewById(R.id.imageButton3);
                        if(lamp2.equals("0")){

      lamp2_pic.setImageResource(R.drawable.lamp2_b);
                              lamp2status = "0";
                              Log.v("msg", "Lamp2 off");

                        }else if(lamp2.equals("1")){

      lamp2_pic.setImageResource(R.drawable.lamp2);
                              lamp2status = "1";
                              Log.v("msg", "Lamp2 on");
                        }else{
                              request("lamp2 error Check again");
                        }

                        //Check fan
                        ImageButton fan_pic =
(ImageButton)findViewById(R.id.imageButton1);
                        if(fan.equals("0")){
                              fan_pic.setImageResource(R.drawable.fan_b);
                              fanstatus = "0";
                              Log.v("msg", "Fan off");
                        }else if(fan.equals("1")){
                              fan_pic.setImageResource(R.drawable.fan);
                              fanstatus = "1";
                              Log.v("msg", "Fan on");

                        }else{
                              request("fan error Check again");
                        }

                        //Check radio
                        ImageButton radio_pic =
(ImageButton)findViewById(R.id.imageButton2);
                        if(radio.equals("0")){

      radio_pic.setImageResource(R.drawable.radio_b);
                              radiostatus = "0";
                              Log.v("msg", "Radio off");
                        }else if(radio.equals("1")){

      radio_pic.setImageResource(R.drawable.radio);
                              radiostatus = "1";
                              Log.v("msg", "Radio on");

                        }else{
                              request("Radio error Check again");
                        }

                        //Check air
                        ImageButton air_pic =
(ImageButton)findViewById(R.id.imageButton5);
                        if(air.equals("0")){
                              air_pic.setImageResource(R.drawable.air2_b);
                              airstatus = "0";
                              Log.v("msg", "Air off");
                        }else if(air.equals("1")){
                              air_pic.setImageResource(R.drawable.air2);
                              airstatus = "1";
                              Log.v("msg", "Air on");
```

```
                        }else{
                                request("Air error Check again");
                        }
                }
        }catch(Exception e){
                request("Intensity error Check again");
        }


    }

    }
}
```

ภาคผนวก ง
โค้ดส่วนแสดงผลหน้าต่างสำหรับเลือกอุปกรณ์ไฟฟ้า (XML)

```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/bg"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="@drawable/bg2"
    android:padding="0dp"
    android:paddingBottom="@dimen/activity_vertical_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    tools:context=".MainActivity" >

    <FrameLayout
        android:id="@+id/frameLayout2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginTop="144dp" >
    </FrameLayout>

    <ImageButton
        android:id="@+id/imageButton3"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentBottom="true"
        android:layout_alignParentRight="true"
        android:layout_marginBottom="160dp"
        android:layout_marginRight="275dp"
        android:src="@drawable/lamp2_b"
         android:background="@drawable/backgroundstate" />


    <ImageButton
        android:id="@+id/imageButton4"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentBottom="true"
        android:layout_alignParentRight="true"
        android:layout_marginBottom="420dp"
        android:layout_marginRight="130dp"
        android:src="@drawable/lamp_b"
         android:background="@drawable/backgroundstate" />


    <ImageButton
        android:id="@+id/imageButton1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentBottom="true"
        android:layout_alignParentRight="true"
        android:layout_marginBottom="105dp"
        android:layout_marginRight="0dp"
        android:src="@drawable/fan_b"
        android:background="@drawable/backgroundstate" />

    <ImageButton
        android:id="@+id/imageButton5"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentLeft="true"
        android:layout_alignParentTop="true"
        android:src="@drawable/air2_b"
         android:background="@drawable/backgroundstate" />


    <ImageButton
```

```
        android:id="@+id/imageButton2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentBottom="true"
        android:layout_alignParentRight="true"
        android:layout_marginBottom="50dp"
        android:layout_marginRight="20dp"
        android:src="@drawable/radio_b"
         android:background="@drawable/backgroundstate" />

    <TextView
        android:id="@+id/temp"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignBottom="@+id/imageButton5"
        android:layout_alignParentRight="true"
        android:layout_marginBottom="17dp"
        android:layout_marginRight="24dp"
        android:text="Loading..."
        android:textSize="40dp" />

     <TextView
        android:id="@+id/humidity"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignLeft="@+id/temp"
        android:layout_below="@+id/imageButton5"
        android:text="Loading..."
        android:textSize="40dp" />

</RelativeLayout>
```

ภาคผนวก จ
โค้ดส่วนประมวลผลหน้าต่างสำหรับควบคุมอุปกรณ์ไฟฟ้าโดยตัวอย่างหลอดไฟ (JAVA)

```
package com.example.project;

import java.io.DataOutputStream;
import java.io.IOException;
import java.io.OutputStream;
import java.io.PrintWriter;
import java.net.Socket;
import java.net.UnknownHostException;

import com.example.project.Setlamp2Activity.connect;
import com.example.project.Setlamp2Activity.send;

import android.os.AsyncTask;
import android.os.Bundle;
import android.app.Activity;
import android.content.Intent;
import android.util.Log;
import android.view.Menu;
import android.view.View;
import android.widget.Switch;

public class SetActivity extends Activity
{
static Socket s;
static DataOutputStream dataOut;

@Override
protected void onCreate(Bundle savedInstanceState)
{
      super.onCreate(savedInstanceState);
      setContentView(R.layout.set);

      Intent i = getIntent();
                  // Receive data from previous page

      int lampstatus = Integer.parseInt(i.getStringExtra("lampstatus"));  //
Send string convert to Integer


      new connect().execute();


      final Switch switchlamp = (Switch)findViewById(R.id.switch1);

      if ( lampstatus == 1){
            switchlamp.toggle();      //change status button
      }


       switchlamp.setOnClickListener(new View.OnClickListener()
       {

            @Override
            public void onClick(View v)
            {
                  // TODO Auto-generated method stub

                  if (switchlamp.isChecked() == true )     //Lamp on
                              {

                                    new send().execute("0");   //Send
type and status of device to arduino.
                              }
```

```
                                        else
//Lamp off
                                        {

                                                new send().execute("1");
                                        }

                }
        });

}

@Override
public void onBackPressed (){
        try {

                s.close();                      //Socket close

        } catch (IOException e) {
                // TODO Auto-generated catch block
                e.printStackTrace();
        }
        super.onBackPressed();
}


@Override
public boolean onCreateOptionsMenu(Menu menu)
{
        // Inflate the menu; this adds items to the action bar if it is
present.
        getMenuInflater().inflate(R.menu.set, menu);
        return true;
}

class send extends AsyncTask<String, Void, String> {

        @Override
        protected String doInBackground(String... params) {
                // TODO Auto-generated method stub


                try {
                        dataOut.writeChar(params[0].toCharArray()[0]);
                } catch (IOException e) {
                        // TODO Auto-generated catch block
                        e.printStackTrace();
                }
                return null;
        }

}

class connect extends AsyncTask<String, Void, String> {

        @Override
        protected String doInBackground(String... params) {
                // TODO Auto-generated method stub


                try {
                        s = new Socket("192.168.1.205", 3216);
                        dataOut = new DataOutputStream(s.getOutputStream());
```

```
            } catch (UnknownHostException e) {
                  // TODO Auto-generated catch block
                  e.printStackTrace();
            } catch (IOException e) {
                  // TODO Auto-generated catch block
                  e.printStackTrace();
            }
            return null;
      }

}
}
```

ภาคผนวก ฉ
โค้ดส่วนแสดงผลหน้าต่างสำหรับควบคุมอุปกรณ์ไฟฟ้าโดยตัวอย่างหลอดไฟ (XML)

```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="@drawable/bg3"
    android:padding="0dp"
    android:paddingBottom="@dimen/activity_vertical_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    tools:context=".set" >

    <FrameLayout
        android:id="@+id/frameLayout2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginTop="144dp" >
    </FrameLayout>

    <ImageView
        android:id="@+id/imageView1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_centerHorizontal="true"
        android:layout_centerVertical="true"
        android:src="@drawable/lamp11" />

    <TextView
        android:id="@+id/temp"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_above="@+id/imageView1"
        android:layout_centerHorizontal="true"
        android:layout_marginBottom="18dp"
        android:text="Lamp"
        android:textAppearance="?android:attr/textAppearanceLarge" />

    <Switch
        android:id="@+id/switch1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_below="@+id/imageView1"
        android:layout_centerHorizontal="true"
        android:layout_marginTop="52dp"
        android:text="Switch" />

</RelativeLayout>
```

ภาคผนวก ช
โค้ดโปรแกรมส่วนประมวลผลบนอาร์ดูโนและสั่งการเปิด-ปิดอุปกรณ์ไฟฟ้า

```
#include <SPI.h>
#include <Ethernet.h>
#include <dht.h>
#define dht_dpin A0

dht DHT;
int ldr = 1;
int ldr_value = 0;
int incomingByte = 0;

byte mac[] = {
 0xDE, 0xAD, 0xBE, 0xEF, 0xFE, 0xED };
IPAddress ip(192,168,1, 105);
IPAddress gateway(192,168,1, 1);
IPAddress subnet(255, 255, 255, 0);

EthernetServer server(3216);
boolean alreadyConnected = false;
   char lamp1='0';
   char lamp2='0';
   char air='0';
   char fan='0';
   char radio='0';

   int lamp1_i, lamp2_i, fan_i, radio_i, air_i;
   int lamp1_cmd=0, lamp2_cmd=0, fan_cmd=0, radio_cmd=0, air_cmd=0;

void setup()
{
 Serial.begin(9600);
  while (!Serial)
 {
  ;
 }

 pinMode(2, OUTPUT);
 pinMode(3, OUTPUT);
 pinMode(5, OUTPUT);
```

```
  pinMode(6, OUTPUT);
  pinMode(7, OUTPUT);
  pinMode(8, INPUT);
  pinMode(9, INPUT);
  pinMode(A3, INPUT);
  pinMode(A4, INPUT);
  pinMode(A5, INPUT);
  delay(1000);

  Ethernet.begin(mac, ip, gateway, subnet);
  Serial.print("IP Address:");
  Serial.println(Ethernet.localIP());
}

void loop()
{
  EthernetClient client = server.available();
  if (client)
  {
    char incomingByte=client.read();

    //Port1 lamp1
    if (incomingByte == '0')
    {
      digitalWrite(2, HIGH);
      Serial.print ("> ");
      Serial.println (incomingByte);
      lamp1_cmd = 1;
    }
    else if (incomingByte == '1')
    {
      digitalWrite(2, LOW);
      Serial.print ("> ");
      Serial.println (incomingByte);
      lamp1_cmd = 0;
    }

    //Port2 lamp2
    else if (incomingByte == '2')
```

```
{
  digitalWrite(3, HIGH);
  Serial.print ("> ");
  Serial.println (incomingByte);
  lamp2_cmd = 1;
}
else if (incomingByte == '3')
{
  digitalWrite(3, LOW);
  Serial.print ("> ");
  Serial.println (incomingByte);
  lamp2_cmd = 0;
}

//Port3 air
else if (incomingByte == '4')
{
  digitalWrite(5, HIGH);
  Serial.print ("> ");
  Serial.println (incomingByte);
  air_cmd = 1;
}
else if (incomingByte == '5')
{
  digitalWrite(5, LOW);
  Serial.print ("> ");
  Serial.println (incomingByte);
  air_cmd = 0;
}

//Port4 fan
else if (incomingByte == '6')
{
  digitalWrite(6, HIGH);
  Serial.print ("> ");
  Serial.println (incomingByte);
  fan_cmd = 1;
}
else if (incomingByte == '7')
```

```
{
 digitalWrite(6, LOW);
 Serial.print ("> ");
 Serial.println (incomingByte);
 fan_cmd = 0;
}

//Port5 radio
else if (incomingByte == '8')
{
 digitalWrite(7, HIGH);
 Serial.print ("> ");
 Serial.println (incomingByte);
 radio_cmd = 1;
}
else if (incomingByte == '9')
{
 digitalWrite(7, LOW);
 Serial.print ("> ");
 Serial.println (incomingByte);
 radio_cmd = 0;
}
else if (incomingByte == 'A')
{
 Serial.println(incomingByte);
 DHT.read11(dht_dpin);
 ldr_value = analogRead(ldr);

 client.print(ldr_value);
 client.print(":");
 client.print(DHT.humidity);
 client.print(" ");
 client.print("%");
 client.print(":");
 client.print(DHT.temperature);
 client.print(" ");
 client.print("C");
 client.print(":");
 client.print(lamp1);
```

```
    client.print(":");
    client.print(lamp2);
    client.print(":");
    client.print(air);
    client.print(":");
    client.print(fan);
    client.print(":");
    client.print(radio);
    client.print("#");
  }
}

int attemp = 3, err_accpt = 5;

//-----------------------------------Lamp1
int cnt = 0, temp = 0, err = 0;
while(true){
  lamp1_i = digitalRead(8);
  if(cnt == 0){ cnt++; temp = lamp1_i; }
  else{
    if(lamp1_i == temp){
      cnt++;
    }else{
      cnt = 0;
      err++;
    }
    temp = lamp1_i;
  }

  if(cnt > attemp){
    if(lamp1_cmd != lamp1_i) Serial.println("Lamp1 Error!, Please check hardware connection.");
    if(lamp1_i == 1) lamp1 = '1' ; else if(lamp1_i == 0) lamp1 = '0';
    cnt = 0; temp = 0; err = 0;
    break;
  }

  if(err > err_accpt){
    Serial.println("Error! data very confuse, Please check hardware connection!");
  }
```

```
}

//-------------------------------------Lamp2
cnt = 0, temp = 0, err = 0;
while(true){
  lamp2_i = digitalRead(9);
  if(cnt == 0){ cnt++; temp = lamp2_i; }
  else{
    if(lamp2_i == temp){
      cnt++;
    }else{
      cnt = 0;
      err++;
    }
    temp = lamp2_i;
  }

  if(cnt > attemp){
    if(lamp2_cmd != lamp2_i) Serial.println("Lamp1 Error!, Please check hardware connection.");
    if(lamp2_i == 1) lamp2 = '1' ; else if(lamp2_i == 0) lamp2 = '0';
    cnt = 0; temp = 0; err = 0;
    break;
  }

  if(err > err_accpt){
   Serial.println("Error! data very confuse, Please check hardware connection!");
  }
}

//-------------------------------------Air
cnt = 0, temp = 0, err = 0;
while(true){
  air_i = digitalRead(A3);
  if(cnt == 0){ cnt++; temp = air_i; }
  else{
    if(air_i == temp){
      cnt++;
    }else{
      cnt = 0;
```

```
        err++;
      }
      temp = air_i;
    }

    if(cnt > attemp){
      if(air_cmd != air_i) Serial.println("Lamp1 Error!, Please check hardware connection.");
      if(air_i == 1) air = '1' ; else if(air_i == 0) air = '0';
      cnt = 0; temp = 0; err = 0;
      break;
    }

    if(err > err_accpt){
      Serial.println("Error! data very confuse, Please check hardware connection!");
    }
}

//-----------------------------------Fan
cnt = 0, temp = 0, err = 0;
while(true){
    fan_i = digitalRead(A4);
    if(cnt == 0){ cnt++; temp = fan_i; }
    else{
      if(fan_i == temp){
        cnt++;
      }else{
        cnt = 0;
        err++;
      }
      temp = fan_i;
    }

    if(cnt > attemp){
      if(fan_cmd != fan_i) Serial.println("Lamp1 Error!, Please check hardware connection.");
      if(fan_i == 1) fan = '1' ; else if(fan_i == 0) fan = '0';
      cnt = 0; temp = 0; err = 0;
      break;
    }
```

```
    if(err > err_accpt){
      Serial.println("Error! data very confuse, Please check hardware connection!");
    }
  }

  //-------------------------------------Radio
  cnt = 0, temp = 0, err = 0;
  while(true){
    radio_i = digitalRead(A5);
    if(cnt == 0){ cnt++; temp = radio_i; }
    else{
      if(radio_i == temp){
        cnt++;
      }else{
        cnt = 0;
        err++;
      }
      temp = radio_i;
    }

    if(cnt > attemp){
      if(radio_cmd != radio_i) Serial.println("Lamp1 Error!, Please check hardware connection.");
      if(radio_i == 1) radio = '1' ; else if(radio_i == 0) radio = '0';
      cnt = 0; temp = 0; err = 0;
      break;
    }

    if(err > err_accpt){
      Serial.println("Error! data very confuse, Please check hardware connection!");
    }
  }

  Serial.print(lamp1-48);
  Serial.print(":");
  Serial.print(lamp2-48);
  Serial.print(":");
  Serial.print(air-48);
  Serial.print(":");
  Serial.print(fan-48);
```

```
  Serial.print(":");
  Serial.println(radio-48);
}
```