การพัฒนาต้นแบบระบบจัดการพลังงานในบ้านตามมาตรฐาน IEEE1888 และ ECHONET Lite

นายมนต์ชัย กายาสมบูรณ์

วิทยานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรมหาบัณฑิต
สาขาวิชาวิศวกรรมไฟฟ้า ภาควิชาวิศวกรรมไฟฟ้า
คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย
ปีการศึกษา 2560

Prototype Development of Home Energy Management System Complying with the IEEE1888 and the ECHONET Lite Standards

Mr. Monchai Kayasomboon

A Thesis Submitted in Partial Fulfillment of the Requirements
for the Degree of Master of Engineering Program in Electrical Engineering
Department of Electrical Engineering
Faculty of Engineering
Chulalongkorn University
Academic Year 2017

หัวข้อวิทยานิพนธ์	การพัฒนาต้นแบบระบบจัดการพลังงานในบ้านตามมาตรฐาน IEEE1888 และ ECHONET Lite
โดย	นายมนต์ชัย กายาสมบูรณ์
สาขาวิชา	วิศวกรรมไฟฟ้า
อาจารย์ที่ปรึกษาวิทยานิพนธ์หลัก	ผู้ช่วยศาสตราจารย์ ดร. วันเฉลิม โปรา

คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย อนุมัติให้นับวิทยานิพนธ์ฉบับนี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญามหาบัณฑิต

.....คณบดีคณะวิศวกรรมศาสตร์

(รองศาสตราจารย์ ดร. สุพจน์ เตชวรสินสกุล)

คณะกรรมการสอบวิทยานิพนธ์

.....ประธานกรรมการ

(ผู้ช่วยศาสตราจารย์ ดร.สุรีย์ พุ่มรินทร์)

.....อาจารย์ที่ปรึกษาวิทยานิพนธ์หลัก

(ผู้ช่วยศาสตราจารย์ ดร. วันเฉลิม โปรา)

.....กรรมการภายนอกมหาวิทยาลัย

(รองศาสตราจารย์ ดร.เอกชัย ลีลารัศมี)

มนต์ชัย กายาสมบูรณ์ : การพัฒนาต้นแบบระบบจัดการพลังงานในบ้านตามมาตรฐาน IEEE1888 และ ECHONET Lite (Prototype Development of Home Energy Management System Complying with the IEEE1888 and the ECHONET Lite Standards) อ.ที่ปรึกษาวิทยานิพนธ์หลัก: ผศ. ดร. วันเฉลิม โปรา, 51 หน้า.

มาตรฐาน ECHONET Lite และ มาตรฐาน IEEE1888 ถูกรวมเข้าด้วยกันเพื่อประโยชน์ในการบริหารจัดการพลังงาน ซึ่งแนวคิดเกี่ยวกับ Registry ได้ถูกนำมาประยุกต์รวมในงานวิจัยนี้ด้วย นั่นคืออุปกรณ์ที่ไม่ได้ทำการลงทะเบียนไว้กับ Registry จะไม่สามารถสื่อสารกับอุปกร์อื่นๆในระบบได้ ดังนั้น Registry สามารถเพิ่มประสิทธิภาพในด้านความปลอดภัยของระบบได้ Registry จะถูกสร้างขึ้นด้วยโปรแกรม Processing โดยมีหน้าที่เก็บข้อมูลของอุปกรณ์ภายในระบบ ส่วน Gateway ที่ทำงานอยู่บน Raspberry Pi 1 B+ จะมีหน้าที่แปลงข้อมูลระหว่าง ECHONET Lite และ IEEE1888 และโปรแกรม MongoDB ถูกใช้เป็นเครื่องมือสร้างฐานข้อมูลสำหรับ Storage เพื่อใช้ในการเก็บข้อมูลของอุปกรณ์ ECHONET Lite ซึ่งผลจากการทดลองเมื่อมีการใส่ Registry เข้าไปในระบบ จะทำให้อุปกรณ์ที่มีอยู่ในฐานข้อมูลของ Registry เท่านั้นที่จะสามารถรับ-ส่งข้อมูลถึงกันได้

ภาควิชา วิศวกรรมไฟฟ้า ลายมือชื่อนิสิต .....

สาขาวิชา วิศวกรรมไฟฟ้า ลายมือชื่อ อ.ที่ปรึกษาหลัก .....

ปีการศึกษา 2560

5770272921 : MAJOR ELECTRICAL ENGINEERING
KEYWORDS: ECHONET LITE / IEEE1888 / REGISTRY

MONCHAI KAYASOMBOON: Prototype Development of Home Energy Management System Complying with the IEEE1888 and the ECHONET Lite Standards. ADVISOR: ASST. PROF. WANCHALERM PORA, Ph.D., 51 pp.

The ECHONET Lite standard is combined with the IEEE1888 standard for the energy management purpose. Registry concept of IEEE1888 is, for the first time, implemented in this paper. Devices which have not been registered cannot communicate with others. Therefore, the registry improves system security. The registry, implemented using the Processing language, is used to keep other devices' information. Gateway, developed on Raspberry Pi 1 B+ using the Python language, is used to translate data from ECHONET Lite to IEEE1888 and vice versa. MongoDB is employed for creating a database for the Storage, which is used to store data from ECHONET Lite devices. The testing results show that only the registered ECHONET devices are allowed to store data in the database of the IEEE1888 Storage.

Department: Electrical Engineering Student's Signature .....
Field of Study: Electrical Engineering Advisor's Signature .....
Academic Year: 2017

กิตติกรรมประกาศ

วิทยานิพนธ์ฉบับนี้สำเร็จได้ด้วยดี เนื่องมาจากได้รับความช่วยเหลือและกรุณาอย่างสูงจาก ผู้ช่วย ศาสตราจารย์ ดร. วันเฉลิม โปรา อาจารย์ที่ปรึกษางานวิจัย ที่ได้ให้คำปรึกษาเป็นอย่างดี ตลอดจนถึงการแนะนำ ปรับปรุง และแก้ไขข้อบกพร่องต่างๆในงานวิจัย ด้วยความเอาใจใส่ ผู้วิจัยตระหนักถึงความตั้งใจและความทุ่มเทของอาจารย์ จึงขอกราบขอบพระคุณเป็นอย่างสูงไว้ ณ ที่นี้

ผู้เขียนขอกราบขอบพระคุณ ผู้ช่วย ศาสตราจารย์ สุรีย์ พุ่มรินทร์ ประธานกรรมการวิทยานิพนธ์ และ รองศาสตราจารย์ ดร. เอกชัย ลีลารัศมี กรรมการวิทยานิพนธ์ ที่ได้กรุณาชี้แนะแนวทางและให้คำแนะนำ ตลอดจนข้อสังเกตต่างๆ ทำให้ผู้เขียนสามารถพัฒนาแนวคิด และเห็นถึงปัญหาต่างๆได้อย่างรอบคอบยิ่งขึ้น

ขอขอบคุณความช่วยเหลือเป็นอย่างดีจาก เพื่อน พี่ และน้อง ภายในแลป ESID

สุดท้ายนี้ ผู้วิจัยขอขอบพระคุณ คุณแม่ วัชรี ฉัตรกอบกูล และ คุณพ่อ มณเฑียร กายาสมบูรณ์ ที่ได้เปิดโอกาสให้ได้รับการศึกษาเล่าเรียน ตลอดจนช่วยเหลือและให้กำลังใจอย่างดีเสมอมา

สารบัญ

หน้า

หน้า

สารบัญรูป

บทที่ 1
บทนำ

1.1 ความเป็นมาและความสำคัญ

ในปัจจุบันมาตรฐานการสื่อสารมีความหลากหลายและมีความซับซ้อนในการใช้งานที่แตกต่างกันไป เช่น Mod-Bus [1], Bluetooth [2], ZigBee [3], LonWorks [4], BACnet [5]เป็นต้น

ในการออกแบบระบบการจัดการพลังงานภายในบ้าน (HEMS) หรือ ระบบการจัดการพลังงานภายในอาคาร (BEMS) อุปกรณ์ภายในระบบอาจมีการสื่อสารได้มากกว่า 1 แบบ เช่น Smart Meter อาจใช้การสื่อสารแบบ Mod Bus หรือ Power Line Carrier(PLC), เครื่องปรับอากาศที่ใช้รีโมต Infrared, Smart Plug ที่ใช้ Wi-Fi และอื่นๆอีกมากมาย ซึ่งจะเห็นได้ว่าหากอุปกรณ์ภายในนั้นมีการสื่อสารหลายแบบ การที่จะออกแบบตัวควบคุมที่ทำหน้าที่รับข้อมูลจาก Sensor หรือ อุปกรณ์ต่างๆ เพื่อนำมาประมวลผลนั้นจะทำได้ยาก เพราะระบบมีความหลากหลายนั่นเอง

มาตรฐาน IEEE1888 [6] จึงได้รับการพัฒนาขึ้นเพื่อที่จะแก้ไขปัญหาในการแลกเปลี่ยนข้อมูล ที่เกิดจากความหลากหลายของการสื่อสารของแต่ละอุปกณ์ โดยทำหน้าที่เสมือนตัวกลางที่จะเปลี่ยนข้อมูลจากการสื่อสารรูปแบบหนึ่งไปยังอีกรูปแบบหนึ่ง ทำให้อุปกรณ์ที่มีระบบสื่อสารไม่เหมือนกันสามารถแลกเปลี่ยนข้อมูลกันได้ เช่น อุปกรณ์ที่ใช้ Serial Bus จะสามารถรับข้อมูลจากอุปกรณ์ที่ใช้ ZigBee ได้ เป็นต้น ซึ่งจะเป็นประโยชน์อย่างมากในการออกแบบระบบจัดการภายในอาคาร เพราะสามารถแก้ไขปัญหาที่กล่าวไว้ข้างต้นได้เป็นอย่างดี

ระบบการสื่อสารแบบ ECHONET Lite [7, 8] กำลังถูกพัฒนาขึ้นภายในประเทศญี่ปุ่น ซึ่งเป็นระบบสื่อสารมาตรฐานที่จะนำไปใช้กับอุปกรณ์ไฟฟ้าต่างๆที่ใช้ภายในอาคารหรือบ้านเรือน แต่เนื่องจากมาตรฐาน ECHONET Lite เป็นการใช้งานได้ดีในระบบขนาดเล็กถึงขนาดกลาง แต่หากระบบต้องการมีการควบคุมที่ไกลขึ้น สามารถเข้าถึงข้อมูลจากที่ห่างไกล ECHONET Lite จะทำได้ไม่ดีนัก จึงจะนำมาตรฐาน IEEE1888 ซึ่งเป็นมาตรฐานที่ใช้ในการสื่อสาร แลกเปลี่ยนข้อมูลระยะไกลซึ่งเป็นระบบขนาดใหญ่ มาเชื่อมต่อกับมาตรฐาน ECHONET Lite จะทำให้สามารถใช้ข้อดีของทั้ง 2 มาตรฐานได้อย่างเต็มที่

หากต้องการให้ทั้ง 2 มาตรฐานเชื่อมต่อกันได้นั้นจำเป็นจะต้องมีอุปกรณ์ที่เรียกว่า Gateway เพื่อเป็นตัวกลางในการเปลี่ยนแปลงข้อมูลจาก ECHONET Lite เป็น IEEE1888 หรือจาก IEEE1888 เป็น ECHONET Lite นั้นเอง

1.2 ทบทวนวรรณกรรม

1.2.1 งานวิจัยที่เกี่ยวข้องเรื่อง “Data transmission of IEEE1888 communication for wide-area real-time smart grid applications” [9] เป็นงานวิจัยเกี่ยวกับปรับปรุงเวลาหน่วงของการส่งข้อมูลแบบ soft-real-time โดยใช้มาตรฐาน IEEE1888 เป็นมาตรฐานในการรับส่งข้อมูลแบบ WAN ของ smart grid และจำลองการทำงานด้วยโปรแกรม NS2 มีข้อเสียคือระบบไม่มีการจัดการด้านความปลอดภัยที่เหมาะสม

1.2.2 งานวิจัยที่เกี่ยวข้องเรื่อง “Implementation of smart meter working as IEEE1888-6LoWPAN gateway for the building energy management systems” [10] เป็นงานวิจัยเกี่ยวกับการนำมาตรฐาน IEEE1888 และ 6LoWPAN ไปใช้ใน smart meter เพื่อที่จะให้ smart meter สามารถทำหน้าที่เป็น gateway ไปในตัวด้วย ซึ่งเป็นข้อดีเนื่องจากลดอุปกรณ์ในระบบ และยังใช้พลังงานในการทำงานที่น้อยอีกด้วย มีข้อเสียคือระบบมีเพียง Gateway, Storage และ Application ไม่มีการจัดการด้านความปลอดภัย ทำได้เพียงอ่าน เขียน และ แสดงข้อมูลเท่านั้น

1.2.3 งานวิจัยที่เกี่ยวข้องเรื่อง “Development of Real-Time Interworking between IEEE1888 and ECHONET Lite Standards for Building Energy Management System” [11] เป็นงานวิจัยเกี่ยวกับการนำเอามาตรฐาน ECHONET Lite มาเชื่อมต่อกับมาตรฐาน IEEE1888 เพื่อเก็บข้อมูลแบบ real-time เพื่อนำไปใช้ในการจัดการพลังงานภายในอาคาร แต่ยังคงมีข้อเสียคือไม่มีการรวม Registry ซึ่งมีหน้าที่เกี่ยวกับความปลอดภัยของข้อมูลภายในระบบ

1.3 วัตถุประสงค์ของงานวิจัย

1.3.1 จัดทำอุปกรณ์(Registry, Storage, Application) ที่ทำงานบนมาตรฐาน IEEE1888 และเชื่อมต่อกับเครือข่ายด้วย Wi-Fi หรือ LAN

1.3.2 จัดทำอุปกรณ์ไฟฟ้า เช่น หลอดไฟ ที่สามารถทำงานบนมาตรฐาน ECHONET Lite และเชื่อมต่อกับเครือข่ายด้วย Wi-Fi

1.3.3ออกแบบต้นแบบ Gateway ที่ใช้สำหรับแปลงข้อมูลระหว่าง IEEE1888 และ ECHONET Lite

1.4 ขอบเขตของงานวิจัย

1.4.1 สร้างอุปกรณ์ไฟฟ้า เช่น หลอดไฟ และ พัดลม เป็นต้น ที่ใช้มาตรฐาน ECHONET Lite ในการสื่อสารอย่างละ 2 ตัว

1.4.2 สร้าง Registry, Storage และ Application เบื้องต้นในการใช้งานมาตรฐาน IEEE1888 1 ชุด

1.4.3 สร้าง Gateway ที่ใช้เป็นตัวกลางระหว่าง IEEE1888 และ ECHONET Lite

1.5 ประโยชน์ที่ได้รับ

1.5.1 อุปกรณ์ที่เป็นมาตรฐาน ECHONET Lite สามารถส่งข้อมูลไปยังอุปกรณ์ที่เป็นมาตรฐาน IEEE1888 ได้

1.5.2 ระบบมีความปลอดภัยมากขึ้นจากการมี Registry ในมาตรฐาน IEEE1888

1.5.3 ผู้ใช้สามารถนำข้อมูลไปคำนวณหาค่าต่างๆที่เกี่ยวกับระบบจัดการพลังงาน

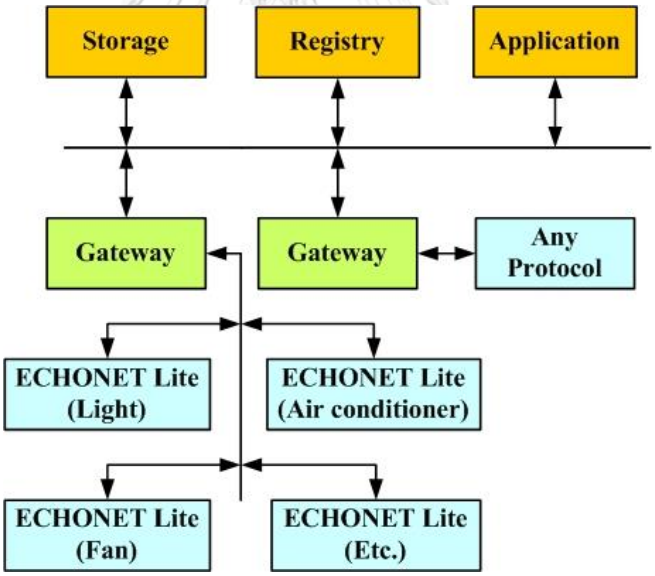
บทที่ 2
ทฤษฎีที่เกี่ยวข้อง

2.1 มาตรฐาน IEEE1888

มาตรฐาน IEEE1888 ถูกใช้งานด้วยจุดประสงค์เพื่อทำให้อุปกรณ์ที่มีการสื่อสารหลายแบบในเครือข่ายอุปกรณ์ระหว่างอาคารต่างๆ สามารถที่จะทำงานร่วมกันได้

2.1.1 สถาปัตยกรรมของระบบ

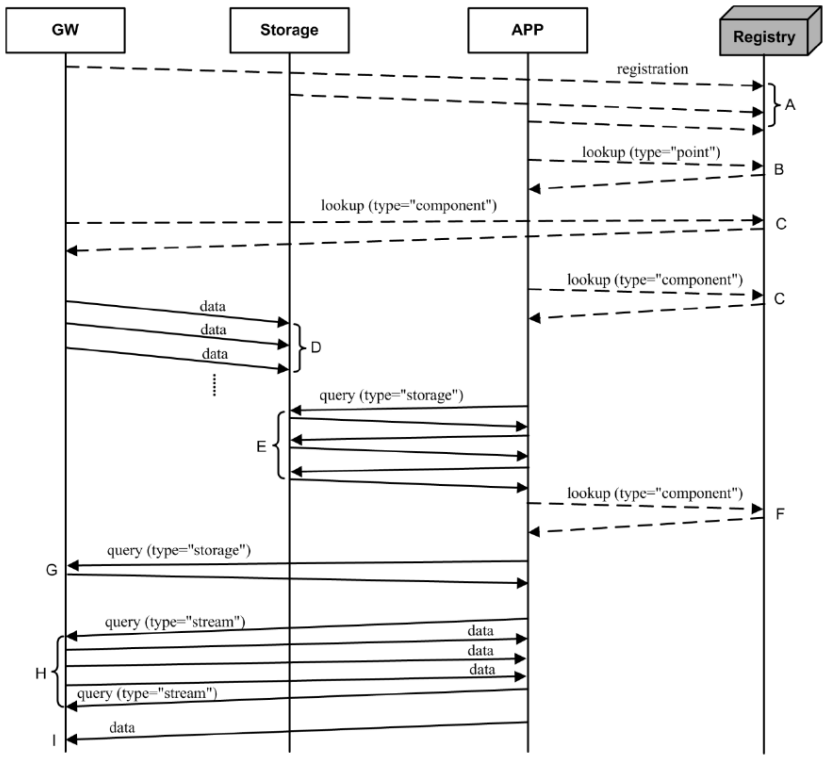
มาตรฐาน IEEE1888 มีองค์ประกอบที่สำคัญ 4 ส่วน คือ เกตเวย์ (Gateway), ฐานเก็บข้อมูล (Storage) และ แอพพลิเคชั่น (Application) จะถูกเรียกโดยรวมว่า อุปกรณ์ (Component) และส่วนที่ 4 ถูกเรียกว่า รีจิสทรี (Registry) ดังรูปที่ 1 ภาพรวมสถาปัตยกรรมของมาตรฐาน IEEE1888รูปที่ 1 โดยแต่ละส่วนก็จะมีหน้าที่การทำงานที่แตกต่างกัน



รูปที่ 1 ภาพรวมสถาปัตยกรรมของมาตรฐาน IEEE1888

โครงข่ายอุปกรณ์ (Field Bus Network) คือ โครงข่ายของอุปกรณ์ที่สื่อสารในมาตรฐานต่างๆ เช่น Bluetooth, ZigBee, Modbus เป็นต้น จะเห็นได้ว่าภายในระบบหนึ่งๆสามารถที่จะมีมาตรฐานการสื่อสารมากกว่าหรือเท่ากับ 1 รูปแบบเสมอ หากต้องการให้มาตรฐานใดๆสามารถส่งข้อมูลเป็น IEEE1888 จะต้องใช้เกตเวย์เพื่อแปลงรูปแบบข้อมูลของแต่ละการสื่อสารหรือโพรโตคอล ให้กลายเป็น

รูปแบบข้อมูลของ IEEE1888 โดยข้อมูลที่ส่งผ่านออกมาจะถูกนำไปเก็บไว้ในฐานเก็บข้อมูล ในรูปแบบของข้อมูลอนุกรมเวลา ซึ่งสามารถถูกนำไปคำนวณหรือวิเคราะห์เพื่อหาค่าที่เกี่ยวข้องกับการใช้พลังงาน หรือนำไปสร้างกราฟแสดงการใช้งานได้จากการสร้างแอพพลิเคชั่น และอุปกรณ์อย่างสุดท้าย คือ Registry ที่มีหน้าที่ในการจัดเก็บรายชื่อของอุปกรณ์ทั้งหมดที่อยู่ภายในระบบ หากอุปกรณ์ใดที่ไม่ได้อยู่ในรายชื่อของรีจิสทรี อุปกรณ์นั้นๆจะไม่สามารถสื่อสารกับอุปกรณ์ตัวอื่นๆภายในระบบได้



รูปที่ 2 การทำงานของ IEEE1888 ระหว่าง component-to-component และ component-to-Registry

จากรูปที่ 2 เป็นการแสดงลำดับการทำงานของมาตรฐาน IEEE1888 โดยแต่ละลำดับมีรายละเอียดดังต่อไปนี้

ลำดับ *A*: เป็นการขอลงทะเบียนของอุปกรณ์ต่างๆได้แก่ เกตเวย์, ฐานเก็บข้อมูล และ แอพพลิเคชั่น

ลำดับ *B*: แอพพลิเคชั่น สืบค้น Point ID ที่ต้องการจาก รีจิสทรี

ลำดับ *C*: เกตเวย์ และ แอพพลิเคชั่น สืบค้น Point ID ของ ฐานข้อมูล จาก รีจิสทรี

ลำดับ *D*: การส่งข้อมูลที่เกตเวย์ได้รับมาจากตัวตรวจรู้หรืออุปกรณ์อื่นๆ ไปเก็บยังฐานเก็บข้อมูล
ลำดับ *E*: การอ่านข้อมูลของ Point ID ที่ ฐานเก็บข้อมูล โดย แอพพลิเคชั่น
ลำดับ *F*: แอพพลิเคชั่น สืบค้น Pont ID ของ เกตเวย์ จาก รีจิสทรี
ลำดับ *G*: แอพพลิเคชั่น อ่านข้อมูลของ Point ID ที่ต้องการจาก เกตเวย์
ลำดับ *H*: แอพพลิเคชั่น อ่านข้อมูลของ Point ID ที่ต้องการจาก เกตเวย์ ในรูปของข้อมูลต่อเนื่อง

จะเห็นได้ว่า หากต้องการติดต่อกับอุปกรณ์ใดๆในระบบ IEEE1888 จะต้องทำการสืบค้นไปยังรีจิสทรีก่อนในครั้งแรกเพื่อตรวจสอบว่ามีอุปกรณ์นั้นๆอยู่ในระบบหรือไม่ หากไม่มีการสืบค้นจะไม่สามารถรู้ได้ว่าอุปกรณ์ที่ต้องการจะสื่อสารด้วยมี Point ID อะไร ซึ่งจะทำให้ไม่สามารถรับส่งข้อมูลได้

2.1.2 โพรโตคอลการสื่อสาร

2.1.2.1 อุปกรณ์ กับ อุปกรณ์ (component-to-component)

การสื่อสารแบบอุปกรณ์ กับ อุปกรณ์ คือการสื่อสารกันระหว่างอุปกรณ์ 2 ตัว เช่น เกตเวย์ กับ ฐานเก็บข้อมูล หรือ แอพพลิเคชั่น กับ เกตเวย์ เป็นต้น โดยการสื่อสารระหว่างอุปกรณ์ กับ อุปกรณ์ สามารถแบ่งออกได้เป็น 3 รูปแบบคือ

FETCH – การอ่านข้อมูลจากอุปกรณ์ที่ต้องการ โดยอุปกรณ์ต้นทางจะต้องส่งคำขอข้อมูลที่ต้องการอ่าน เช่น ข้อมูลอุณหภูมิในช่วงเวลาหนึ่ง เป็นต้น ไปยังอุปกรณ์ปลายทางที่มีข้อมูลที่ต้องการ อุปกรณ์ปลายทางก็จะส่งข้อมูลที่เกี่ยวข้องกลับไปให้อุปกรณ์ต้นทาง

WRITE - จะมีลักษณะการทำงานคล้ายคลึงกับ FETCH แต่เปลี่ยนจากการอ่านข้อมูลเป็นการเขียนข้อมูลลงไปในอุปกรณ์ที่ต้องการแทน

TRAP - การอ่านชุดของข้อมูลจากอุปกรณ์ที่ต้องการ ในเหตุการณ์ หรือเงื่อนไขที่ถูกกำหนดโดยอุปกรณ์ต้นทาง หรือผู้ใช้งาน

2.1.2.2 อุปกรณ์ กับ รีจิสทรี (component-to-registry)

การสื่อสารแบบ อุปกรณ์ กับ รีจิสทรี เป็นการสื่อสารกันระหว่าง รีจิสทรี และ อุปกรณ์ต่างๆ ในระบบ โดยแบ่งออกเป็น 2 รูปแบบคือ

REGISTRATION – ใช้ในการขอลงทะเบียน Point ID หรือ PointSet

LOOKUP – ใช้ในการสืบค้น Point ID หรือ PointSet

ทั้งนี้ คำขอลงทะเบียน และ คำขอสืบค้น นั้นมีลักษณะคำตอบที่คล้ายคลึงกัน คำขอลงทะเบียน (REGISTRATION) ด้วย Point ID หากสำเร็จรีจิสทรีจะตอบกลับด้วยข้อความ OK หากไม่สำเร็จจะตอบกลับด้วยข้อความ ERROR แล้วตามด้วยชนิดของความผิดพลาด เช่นเดียวกันกับคำขอสืบค้น (LOOKUP) หากพบ รีจิสทรีจะตอบด้วยข้อความ OK และหากไม่พบก็จะตอบด้วยข้อความ ERROR เช่นกัน

2.1.3 โครงสร้างข้อมูล

2.1.3.1 Point, URI-Base Identification และ PointSet

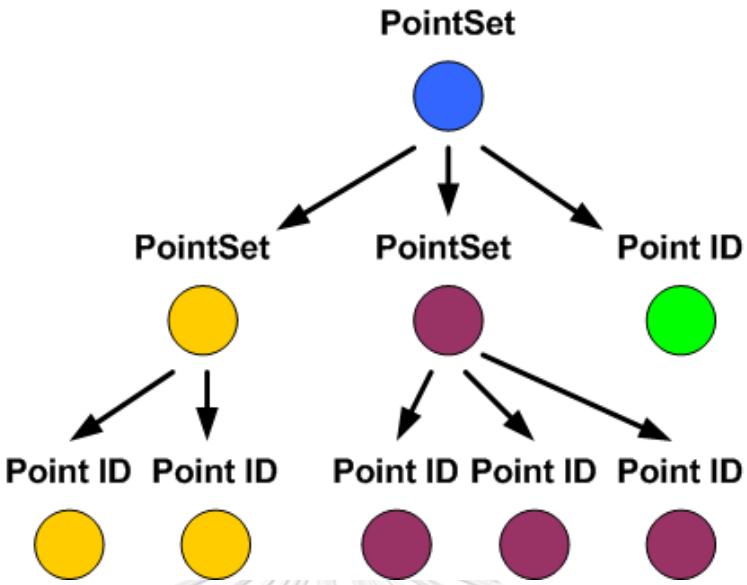
ความหมายของ Point คือ ชื่อลำดับข้อมูลของอุปกรณ์ หรือ ตัวตรวจรู้ต่างๆ เช่น อุณหภูมิ ความเข้มแสง ความชื้น แรงดันไฟฟ้า กระแสไฟฟ้า หรือ สถานะ เป็นต้น

อุปกรณ์ หรือ ตัวตรวจรู้แต่ละตัวจะมีการระบุชื่อที่แตกต่างกัน เพื่อใช้ในการระบุว่าข้อมูลที่ต้องการอ่านหรือเขียนนั้นเป็นอุปกรณ์ตัวใด โดยในมาตรฐาน IEEE1888 จะเรียกว่า Point ID ซึ่ง Point ID นี้จะมีหลักการตั้งชื่อแบบ URI-Base Identification โดยมีหลักการตั้งชื่อดังนี้

Point ID = http://<Gateway Host Name>/<Any format to identify the Point in Gateway>

ในบางอุปกรณ์อาจมีข้อมูลได้หลายอย่าง เช่น เครื่องปรับอากาศ มีข้อมูลของ อุณหภูมิ ความชื้น ความแรงลม เป็นต้น กล่าวได้ว่าอุปกรณ์ตัวนี้มีหลาย Point ID จึงเป็นที่มาของ PointSet

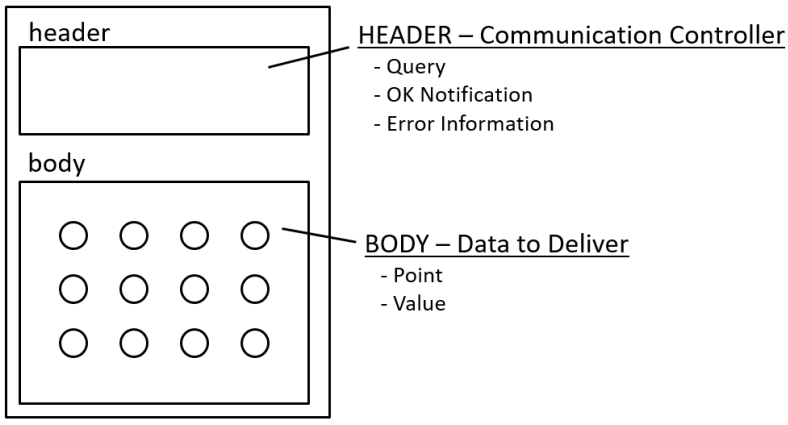
PointSet คือ กลุ่มของ Point ID ภายในอุปกรณ์หนึ่งๆ หรือภายในพื้นที่ที่สนใจ ซึ่งภายใน PointSet สามารถมี PointSet เป็นส่วนประกอบได้เช่นกัน ดังแสดงในรูปที่ 3



รูปที่ 3 ลักษณะของ PointSet และPoint ID

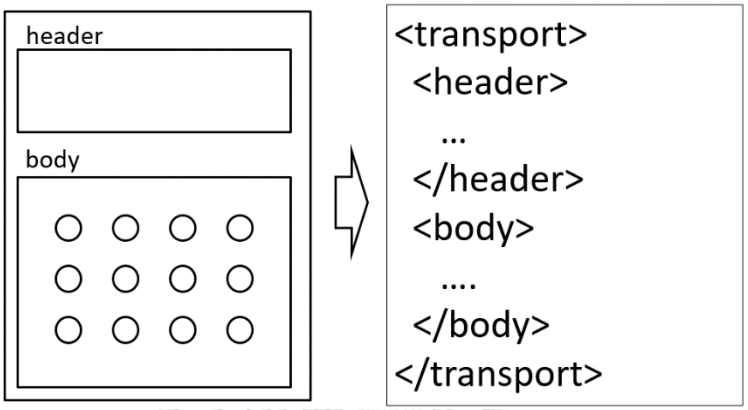
2.1.3.2 IEEE1888 Frame Format

ลักษณะของการส่งและรับข้อมูลระหว่างอุปกรณ์ต่างๆในของมาตรฐาน IEEE1888 จะใช้ภาษา eXtensible Markup Language (XML) [12] ในการสื่อสาร ซึ่งข้อความที่ส่งจาก Requester จะแบ่งออกเป็น 2 ส่วนคือ Header และ Body ดังรูปที่ 4

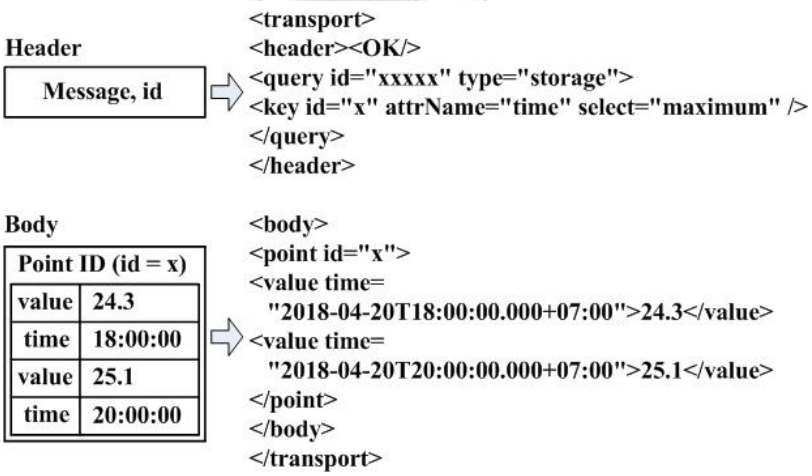


รูปที่ 4 รูปแบบข้อความการสื่อสารของมาตรฐาน IEEE1888

ในส่วนของ Header จะเก็บข้อมูลเกี่ยวกับการแจ้งเตือนการตอบกลับ รายละเอียดความผิดพลาด และ ข้อมูลที่ต้องการสืบค้น และในส่วนของ Body จะเก็บข้อมูลของ Point และค่าของ Point ที่ต้องการอ่านหรือเขียน โดย Header และ Body จะมีรูปแบบใน XML ดังรูปที่ 5 โดยมีตัวอย่างการใช้งานดังรูปที่ 6



รูปที่ 5 รูปแบบข้อความการสื่อสารของมาตรฐาน IEEE1888 แบบ XML



รูปที่ 6 ตัวอย่างการส่งคำขออ่านข้อมูลจาก Point ID = x

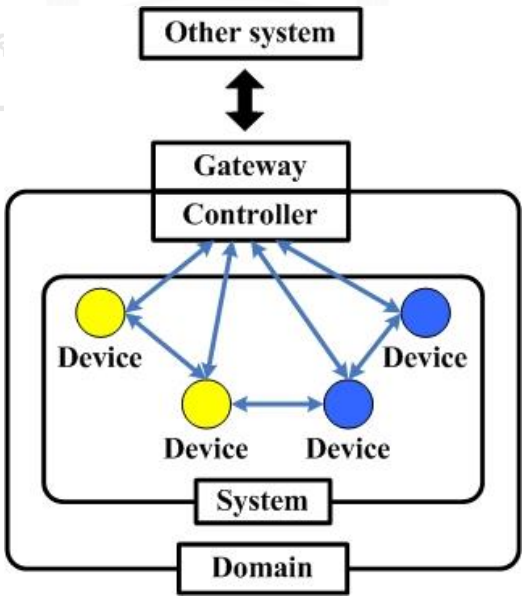
2.2 มาตรฐาน ECHONET Lite

Energy Conversation and Homecare Network Lite หรือ ECHONET Lite เป็นมาตรฐานที่ถูกออกแบบโดย ECHONET Consortium มีจุดประสงค์เพื่อใช้ในการควบคุมอุปกรณ์ไฟฟ้า หรือ ตัวตรวจรู้ต่างๆ ภายในอาคารที่พักอาศัย เช่น เครื่องปรับอากาศ หลอดไฟส่องสว่าง กลอนประตูอิเล็คทรอนิกส์ ตัวตรวจจับควัน เครื่องวัดอุณหภูมิ เป็นต้น

ECHONET Lite นั้นสามารถเลือกใช้การสื่อสารได้ทั้งมีสายและไร้สาย แต่โดยทั่วไปแล้วจะใช้การสื่อสารแบบไร้สาย เช่น Wi-Fi หรือ Bluetooth เป็นต้น ซึ่งจะทำให้การติดตั้งทำได้ง่ายและประหยัดพลังงาน อีกทั้งยังง่ายต่อการพัฒนาเนื่องจากมีรูปแบบของข้อมูลที่ไม่ซับซ้อน และมีเครื่องมือออกแบบให้เลือกใช้ได้หลากหลาย

2.2.1 สถาปัตยกรรมของระบบ

ECHONET Lite ประกอบไปด้วยอุปกรณ์ต่างๆ ซึ่งถูกเรียกว่า Node แต่ละ Node สามารถส่งหรือรับข้อมูลจากกันและกันได้ เมื่อ Node หลายๆตัว อยู่ในพื้นที่ใกล้ๆกัน เช่น หลอดไฟ หรือเครื่องปรับอากาศที่อยู่ภายในห้องเดียวกัน จะสามารถเรียกรวมกันว่า System ซึ่งเมื่อมี System มารวมกันหลายๆชุด จะถูกเรียกว่า Domain ซึ่งสามารถส่งหรือรับข้อมูลจากระบบภายนอกได้ผ่านทางเกตเวย์ ดังที่แสดงในรูปที่ 7

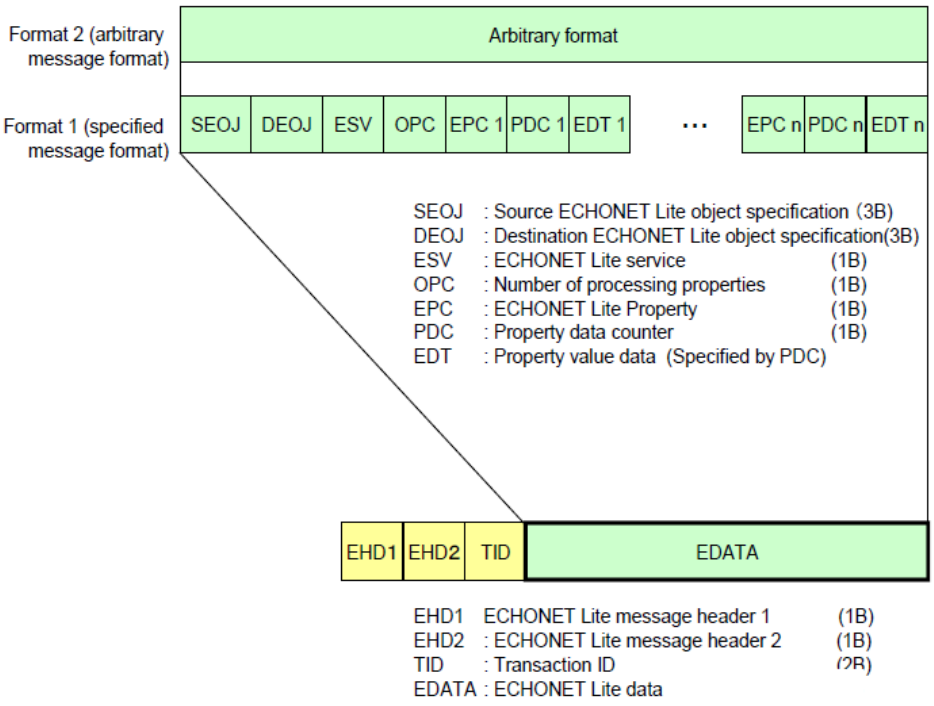


รูปที่ 7 ภาพรวมของมาตรฐาน ECHONET Lite

อุปกรณ์ที่สำคัญอีกอย่างหนึ่งคือ ตัวควบคุม (Controller) มีหน้าที่รับส่งข้อมูลจากเกตเวย์ เพื่อใช้ในการควบคุม หรือ ตรวจสอบค่าตัวแปรต่างๆของอุปกรณ์ภายในระบบ โดยค่าที่สามารถอ่าน หรือสั่งงาน จะเป็นค่า ณ ปัจจุบันเท่านั้น ไม่สามารถอ่านค่าย้อนหลังได้ หรือก็คือ ECHONET Lite ไม่มีอุปกรณ์ที่สามารถจัดเก็บข้อมูลต่างๆภายในระบบ ซึ่งเป็นข้อเสียของ ECHONET Lite นั่นเอง

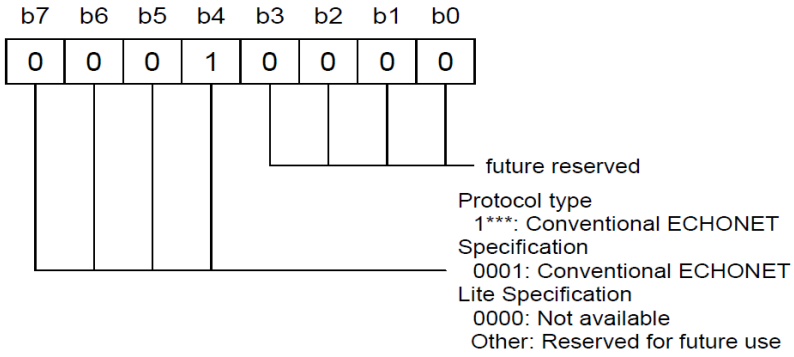
2.2.2 โครงสร้างข้อมูล

การสื่อสารของ ECHONET Lite สามารถทำได้หลายรูปแบบดังที่กล่าวไว้ข้างต้น โดยมี ECHONET Lite Frame Format แสดงดังรูปที่ 8 โดยแต่ละส่วนมีรายละเอียดดังต่อไปนี้



รูปที่ 8 รูปแบบเฟรมข้อมูลของ ECHONET Lite

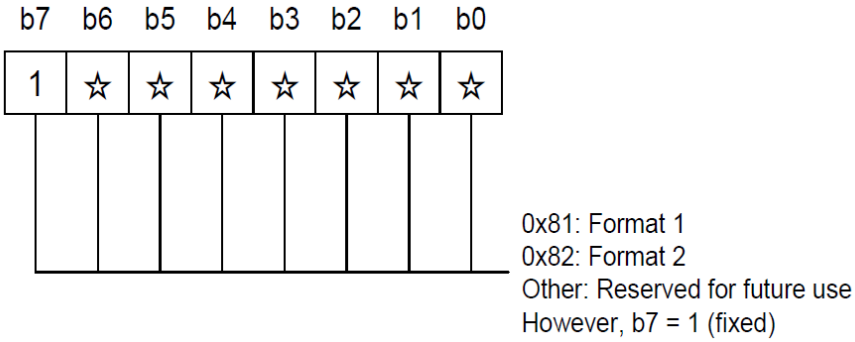
EHD1 - ตัวแปร ECHONET Lite Header 1 ถูกใช้เพื่อบ่งบอกถึงมาตรฐานที่ถูกใช้งานมีรายละเอียดดังรูปที่ 9



รูปที่ 9 รายละเอียดของ EHD1

ค่าของตัวแปร EHD1 ในบิตที่ b7:b4 จะถูกกำหนดให้เท่ากับ 0b0001 ซึ่งบ่งบอกถึงมาตรฐาน ECHONET Lite และบิตที่ b3:b0 ถูกกำหนดให้เป็น 0b0000 เนื่องจากปัจจุบันยังไม่มีการใช้งาน

EHD2 - ECHONET Lite Header 2 ใช้เพื่อกำหนดรูปแบบของข้อมูล (EDATA) ซึ่งมีอยู่ 2 รูปแบบ คือ Specified message format และ Arbitrary format ดังรูปที่ 10

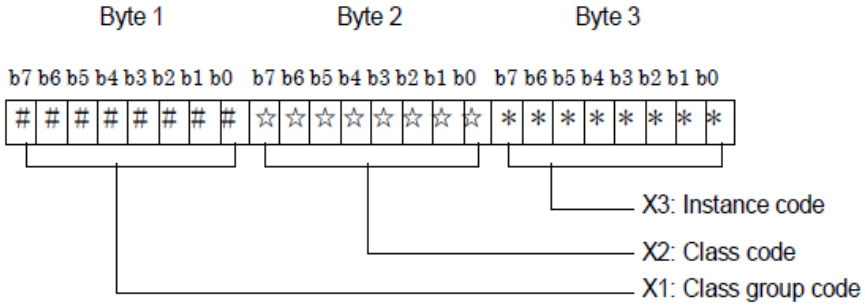


รูปที่ 10 รายละเอียดของ EHD2

TID - Transaction ID คือ ค่าตัวแปรที่ใช้ในการระบุผู้รับและผู้ส่ง เมื่อผู้รับได้รับข้อความจะต้องจดจำค่า TID และทำการส่งข้อความตอบรับที่มี TID เดียวกันเพื่อยืนยันกับผู้ส่งว่าได้รับข้อความแล้ว

EDATA - ECHONET Lite data เป็นส่วนที่ประกอบไปด้วยข้อมูลต่างๆที่อุปกรณ์แต่ละตัวต้องการสื่อสารกันมีรายละเอียดดังต่อไปนี้

SEOJ และ DEOJ - ECHONET Lite Source Object และ ECHONET Lite Destination Object คือ ชนิดของอุปกรณ์ตัวรับ และ ชนิดของอุปกรณ์ตัวส่ง ประกอบไปด้วยคุณลักษณะ 3 อย่าง คือ Class group, Code class และ Instance code ดังรูปที่ 11



รูปที่ 11 รายละเอียดของ SEOJ และ DEOJ

Class group code มีค่าที่ถูกกำหนดไว้ดังรูปที่ 12 โดยแต่ละกลุ่ม จะมี Class code แยกย่อยอีกทีหนึ่งด้วย ยกตัวอย่างดังรูปที่ 13 และ รูปที่ 14

GROUP CODE	GROUP NAME	REMARKS
0x00	Sensor-related device class group	
0x01	Air conditioner-related device class group	
0x02	Housing/facility-related device class group	
0x03	Cooking/housework-related device class group	
0x04	Health-related device class group	
0x05	Management/control-related device class group	
0x06	AV-related device class group	
0x07-0x0D	Reserved for future use	
0x0E	Profile class group	
0x0F	User definition class group	
0x10-0xFF	Reserved for future use	

รูปที่ 12 รายชื่อ Class group code

CLASS CODE	CLASS NAME	DETAILED SPECS.	REMARKS
0x00-0xFC	Reserved for future use		
0xFD	Switch		
0xFE	Portable terminal		
0xFF	Controller		

รูปที่ 13 รายชื่อของ Class code (Class group code = 0x05)

CLASS CODE	CLASS NAME	DETAILED SPECS.	REMARKS
0x00-0xEF	Reserved for future use		
0xF0	Node profile		
0xF1-0xFF	Reserved for future use		

รูปที่ 14 รายชื่อของ Class code (Class group code = 0x0E)

ESV - ECHONET Lite Service คือ ตัวแปรที่ใช้เพื่อบ่งบอกว่าข้อมูลที่จะสื่อสารนี้เป็นการอ่าน, เขียน, ต้องการข้อความตอบกลับ หรือ ไม่ต้องการข้อความตอบกลับ โดยมีรายละเอียดดังรูปที่ 15 และ รูปที่ 16

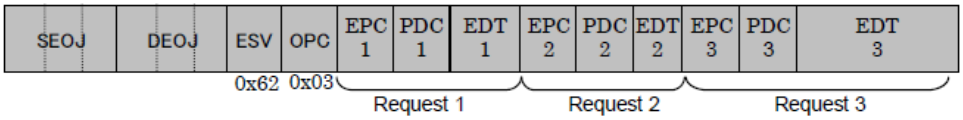
Service Code (ESV)	ECHONET Lite Service Content	Symbol	Remarks
0x60	Property value write request (no response required)	SetI	Broadcast possible
0x61	Property value write request (response required)	SetC	
0x62	Property value read request	Get	Broadcast possible
0x63	Property value notification request	INF_REQ	Broadcast possible
0x64-0x 6D	Reserved for future use		
0x6E	Property value write & read request	SetGet	Broadcast possible
0x6F	Reserved for future use		

รูปที่ 15 รายชื่อของ service code request

Service Code (ESV)	ECHONET Lite Service Content	Symbol	Remarks
0x71	Property value Property value write response	Set_Res	ESV=0x61 response; Individual response
0x72	Property value read response	Get_Res	ESV=0x62 response; Individual response
0x73	Property value notification	INF	*1 : Both individual notification and broadcast notification
0x74	Property value notification (response required)	INFC	Individual notification
0x75-0x79	Reserved for future use		
0x7A	Property value notification response	INFC_Res	ESV=0x74 response; Individual response
0x7B-0x7D	Reserved for future use		
0x7E	Property value write & read response	SetGet_Res	ESV=0x6E response; Individual response
0x7F	Reserved for future use		

รูปที่ 16 รายชื่อของ service code response

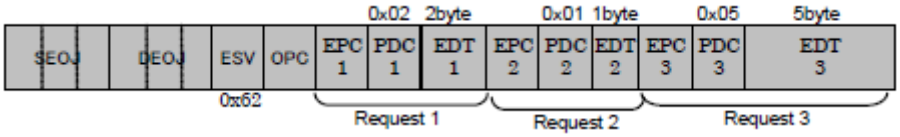
OPC - Processing Target Property Counters คือ ตัวแปรที่มีเพื่อบ่งบอกว่าข้อมูลที่ต้องการอ่านหรือเขียนนั้นมีจำนวนกี่ชุด เช่น ต้องการอ่านข้อมูลจำนวน 3 ชุด ดังรูปที่ 17



รูปที่ 17 Processing Target Property Counters (OPC) = 3 Request

EPC - ECHONET Lite Property คือ ชนิดของข้อมูลที่ถูกเก็บอยู่ภายในอุปกรณ์ ซึ่งแต่ละอุปกรณ์จะมีข้อมูลบางชนิดที่เหมือนกัน และบางชนิดที่แตกต่างกัน แล้วแต่ประเภทของอุปกรณ์ เช่น หลอดไฟ และ เครื่องปรับอากาศ มีชนิดข้อมูลที่เรียกว่า สถานะ เปิดปิด เหมือนกัน แต่ หลอดไฟจะไม่มีข้อมูลที่เกี่ยวกับอุณหภูมิแบบเครื่องปรับอากาศ ในทางกลับกันเครื่องปรับอากาศก็ไม่มีข้อมูลเกี่ยวกับความเข้มแสงเช่นกัน ซึ่งค่า EPC ของอุปกรณ์แต่ละชนิดสามารถดูรายละอียดได้จาก [13]

PDC - Property Data Counter คือตัวแปรที่ใช้กำหนดว่า เฟรมข้อมูลที่ส่งหรือรับมานั้นมีค่า property value กี่ชนิด ดังแสดงในรูปที่ 18 โดยค่า OPC = 0x03, PDC1 = 0x02, PDC2 = 0x01 และ PDC3 = 0x05



รูปที่ 18 รายละเอียดของ Property Data Counter

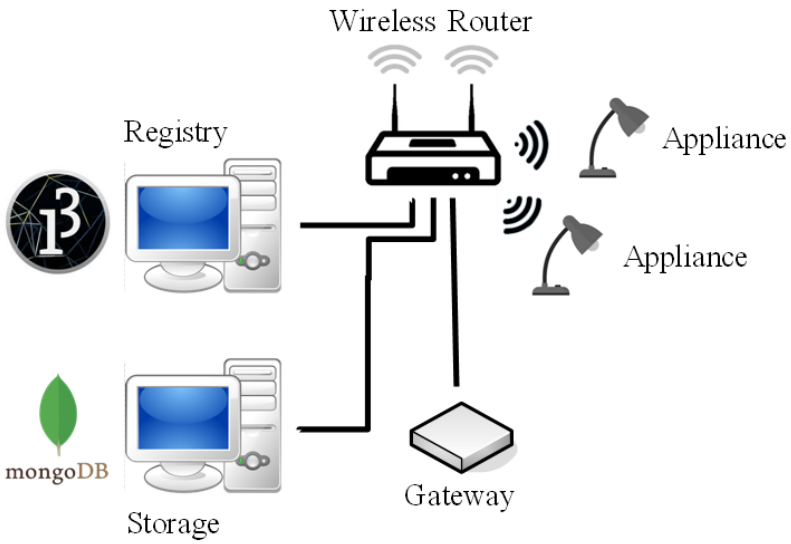
EDT - ECHONET Property Value Data คือค่าของชนิดข้อมูล (EPC) เช่น ค่าอุณหภูมิ ค่าความชื้น ค่ากระแส สถานะการทำงาน เป็นต้น

บทที่ 3
การออกแบบและพัฒนา

3.1 โครงสร้างระบบ

ภายในระบบที่ถูกออกแบบถูกแบ่งออกเป็น 2 ส่วน คือ ส่วนของ ECHONET Lite ที่เป็นอุปกรณ์ไฟฟ้า และ ส่วนของ IEEE1888 ที่เป็นส่วนของ ฐานเก็บข้อมูล และ รีจิสทรี

ด้วยการนำข้อดี และ ข้อเสียของทั้งสองมาตรฐานมารวมกัน ทำให้สามารถแก้ไขข้อเสียของแต่ละมาตรฐานได้เป็นอย่างดี กล่าวคือมาตรฐาน ECHONET Lite นั้นมีข้อดีในการจัดการกับระบบภายในอาคารขนาดเล็ก มีการออกแบบที่รองรับอุปกรณ์ไฟฟ้าต่างๆ แต่มีข้อเสียที่ไม่สามารถเก็บข้อมูลย้อนหลังได้ ในจุดนี้จึงนำข้อดีของมาตรฐาน IEEE1888 มาใช้เพื่อแก้ไข เนื่องจากมาตรฐาน IEEE1888 นั้นถูกออกแบบมาเพื่อใช้ในการเก็บข้อมูลของระบบในอาคารขนาดใหญ่ และสามารถเก็บข้อมูลย้อนหลังได้ในรูปแบบอนุกรมเวลา โดยรูปแบบของระบบที่ถูกออกแบบจะเป็นไปดังรูปที่ 19

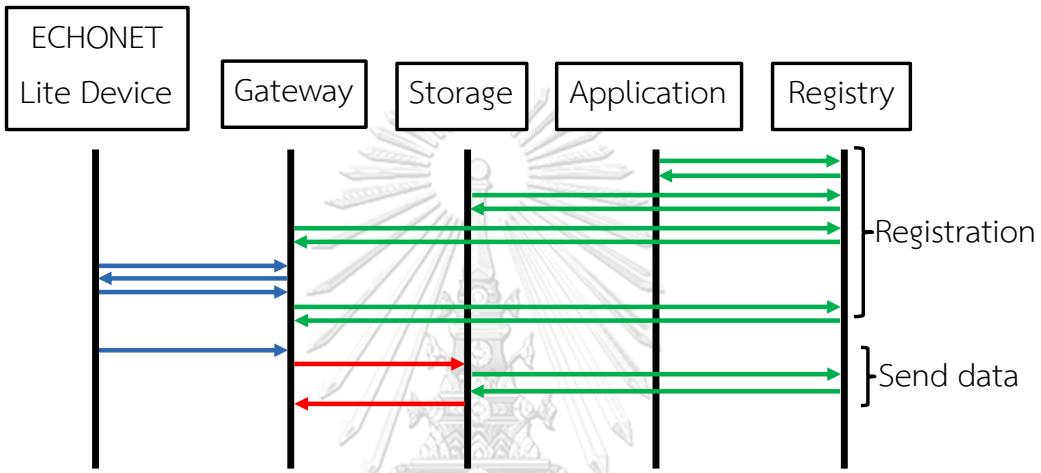


รูปที่ 19 ภาพรวมของระบบที่ถูกออกแบบ

รายละเอียดของระบบที่ถูกออกแบบมีดังต่อไปนี้ ฐานเก็บข้อมูล และ รีจิสทรี จะทำงานอยู่บนคอมพิวเตอร์ โดยฐานเก็บข้อมูลจะใช้พอร์ต 18880 ในการสื่อสาร ส่วนรีจิสทรีจะใช้พอร์ต 18888 ใน

การสื่อสาร ซึ่งคอมพิวเตอร์จะเชื่อมต่อแบบ LAN กับ Wireless Router ส่วนเกตเวย์จะทำงานอยู่บนระบบปฏิบัติการ Raspbian บนบอร์ด Raspberry Pi B+ โดยเชื่อมต่อกับ Wireless Router ด้วย LAN และสุดท้ายคืออุปกรณ์ ECHONET Lite จะทำงานบนบอร์ด Node MCU V3 เชื่อมต่อกับ Wireless Router แบบไร้สายด้วยพอร์ต 3610

ขั้นตอนในการเริ่มทำงานของอุปกรณ์ต่างๆภายในระบบนั้นมีรายละเอียดดังที่แสดงในรูปที่ 20



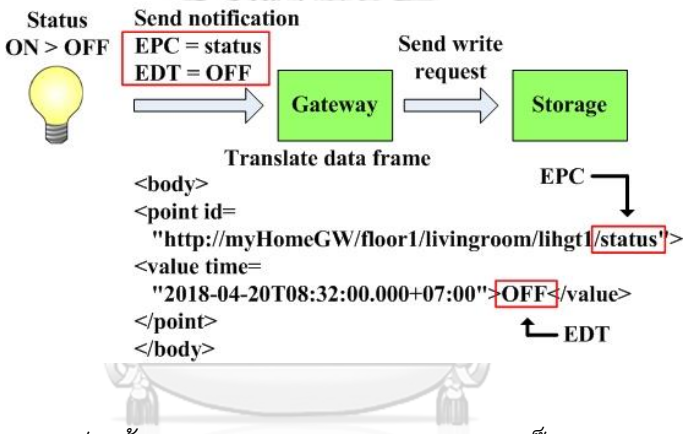
รูปที่ 20 ลักษณะของการลงทะเบียนอุปกรณ์บนระบบที่ออกแบบ

ขั้นตอนที่ 1 – เกตเวย์, ฐานข้อมูล และ แอพพลิเคชั่น จะต้องทำการส่งคำขอลงทะเบียนพร้อมกับ Point ID ของตัวเองไปให้กับรีจิสทรี เพื่อให้รีจิสทรีทำการตรวจสอบว่า Point ID ที่ส่งคำขอลงทะเบียนมานั้นได้ถูกกำหนดเอาไว้ในฐานข้อมูลของรีจิสทรีหรือไม่ หากมีอยู่ในฐานข้อมูลรีจิสทรีจะทำการบันทึกว่า Point ID นี้ได้ถูกลงทะเบียนและส่งคำตอบว่าลงทะเบียนสำเร็จกลับไปยังอุปกรณ์ที่ขอลงทะเบียน แต่หาก Point ID นั้นไม่มีในฐานข้อมูล รีจิสเทรีจะส่งคำตอบกลับว่าการลงทะเบียนล้มเหลวเนื่องจากไม่มี Point ID ที่ตรงกันในฐานข้อมูล

ขั้นตอนที่ 2 - เมื่ออุปกรณ์ ECHONET Lite ถูกเปิดใช้งาน จะมีการส่งข้อความแจ้ง (Notification message) ไปที่เกตเวย์ จากนั้นเกตเวย์จะตรวจสอบว่าอุปกรณ์ที่ส่งข้อความแจ้งเข้ามานั้นถูกลงทะเบียนแล้วหรือไม่ หากยังไม่ได้ลงทะเบียน เกตเวย์จะส่งคำขอ IDnumber จากอุปกรณ์ ECHONET Lite เมื่อได้ IDnumber แล้ว เกตเวย์ก็จะส่งคำขอลงทะเบียนพร้อมกับ IDnumber ไปยังรีจิสทรี เพื่อทำการตรวจสอบว่ามี Point ID หรือ PointSet ใดในฐานข้อมูล ที่เกี่ยวข้องกับ

IDnumber นั้นหรือไม่ หากมี Point ID หรือ PointSet ที่เกี่ยวข้องกัน รีจิสทรีจะทำการส่งข้อความตอบกลับว่าลงทะเบียนสำเร็จพร้อมทั้ง Point ID หรือ PointSet ที่เกี่ยวข้องกลับไปยังเกตเวย์ โดยเกตเวย์จะต้องเก็บบันทึก Point ID หรือ PointSet ที่ได้มาเหล่านั้นเอาไว้เพื่อใช้ในการแปลงข้อมูลจาก ECHONET Lite Frame Format เป็น IEEE1888 Frame Format

การแปลงข้อมูลจาก ECHONET Lite Frame Format ให้เป็น IEEE1888 Frame Format เป็นหน้าที่ของเกตเวย์ที่จะต้องรู้ว่าอุปกรณ์ ECHONET Lite นั้นส่งข้อมูลอะไรมา เช่น อุปกรณ์ ECHONET Lite ที่เป็นหลอดไฟ ส่งข้อมูลสถานะการทำงานมายังเกตเวย์ ว่าสถานะการทำงานได้เปลี่ยนไปจากเปิด เป็น ปิด เกตเวย์จะทำการตรวจสอบว่า หลอดไฟที่ส่งข้อมูลสถานะมานั้นมี Point ID ทีเกี่ยวข้องบันทึกไว้หรือไม่ หากมีการบันทึกไว้ เกตเวย์จะนำข้อมูลสถานะที่ได้รับไปแปลงให้เป็น IEEE1888 Frame Format เพื่อที่จะส่งต่อไปให้กับฐานเก็บข้อมูลต่อไปดังรูปที่ 21



รูปที่ 21 การแปลงข้อมูลจาก ECHONET Lite Frame เป็น IEEE1888 Frame

3.2 ระบบซอฟต์แวร์

โปรแกรมที่ถูกใช้ในการจำลองและควบคุมอุปกรณ์ภายในระบบที่ออกแบบมีอยู่ 4 โปรแกรม คือ

1. โปรแกรม Processing 3.3.6 – เป็นโปรแกรมที่มีเครื่องมือจัดการเกี่ยวการสร้าง User Interface และ เครื่องมือจัดการเกี่ยวกับเครือข่าย ใช้ลักษณะการเขียนโปรแกรมแบบภาษา C ซึ่งโปรแกรม Processing จะถูกใช้ในการสร้างและทดสอบตัวจำลองรีจิสทรี

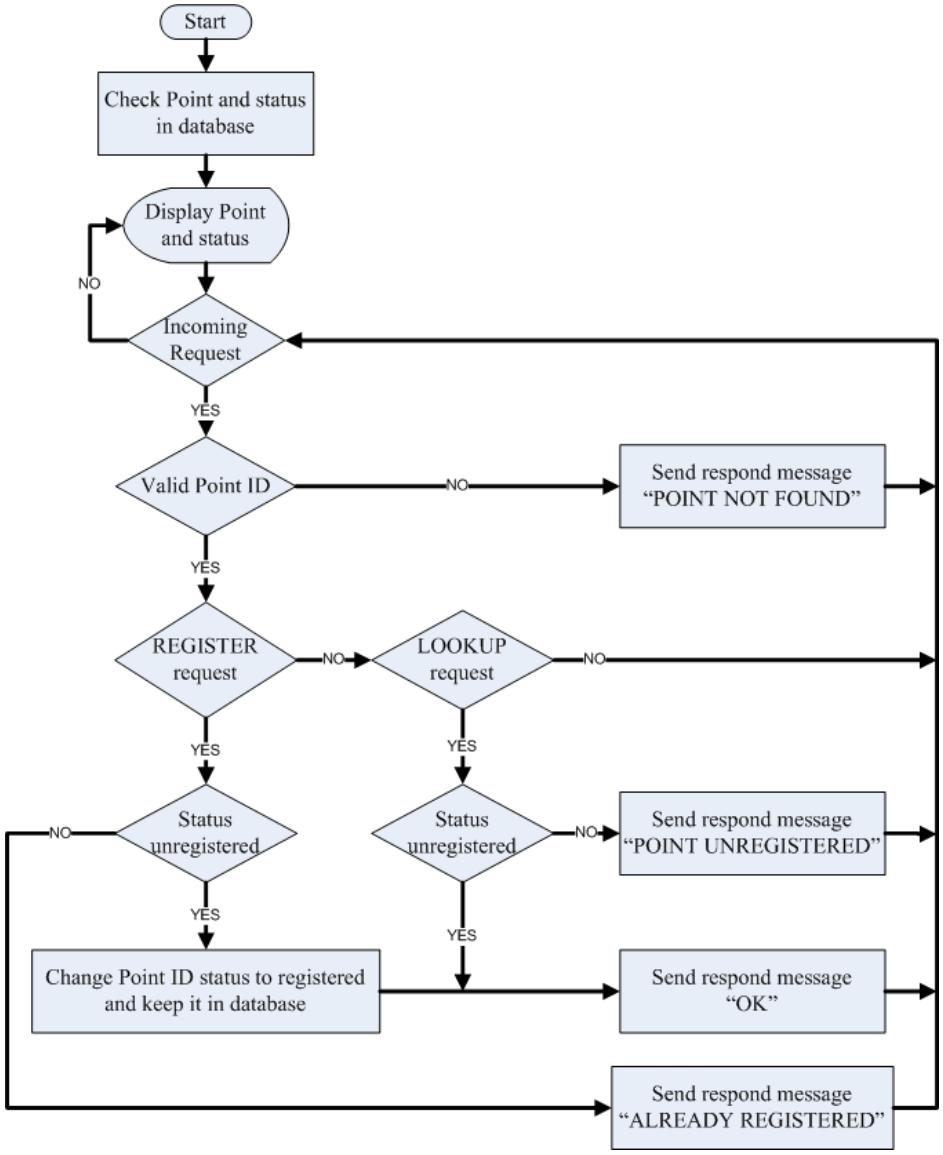
2. โปรแกรม Arduino IDE 1.8.5 – เป็นโปรแกรมที่ใช้ร่วมกันกับบอร์ด NodeMCU V3 ใช้สำหรับควบคุมอุปกรณ์ ECHONET Lite ที่เป็นฮาร์ดแวร์สำหรับทดสอบการทำงานร่วมกันกับอุปกรณ์อื่นๆในระบบ

3. โปรแกรม Python 3.6 – เป็นโปรแกรมที่ถูกใช้เพื่อสร้างตัวจำลองของฐานเก็บข้อมูล และ เกตเวย์ เนื่องจากมีเครื่องมือจัดการเกี่ยวกับระบบฐานข้อมูล โดยใช้ภาษา Python ในการเขียนโปรแกรม

4. โปรแกรม MongoDB – เป็นโปรแกรมสำหรับสร้างฐานข้อมูลเพื่อใช้เก็บข้อมูลต่างๆจากอุปกรณ์ หรือตัวตรวจรู้ในระบบ โดยทำงานร่วมกันกับโปรแกรม Python 3.6 และมีโปรแกรม MongoDB Compass ที่จะแสดงข้อมูลในฐานข้อมูลออกมาเป็นลักษณะที่เข้าใจได้ง่าย

3.2.1 แผนภูมิการทำงานของรีจิสทรี (Registry)

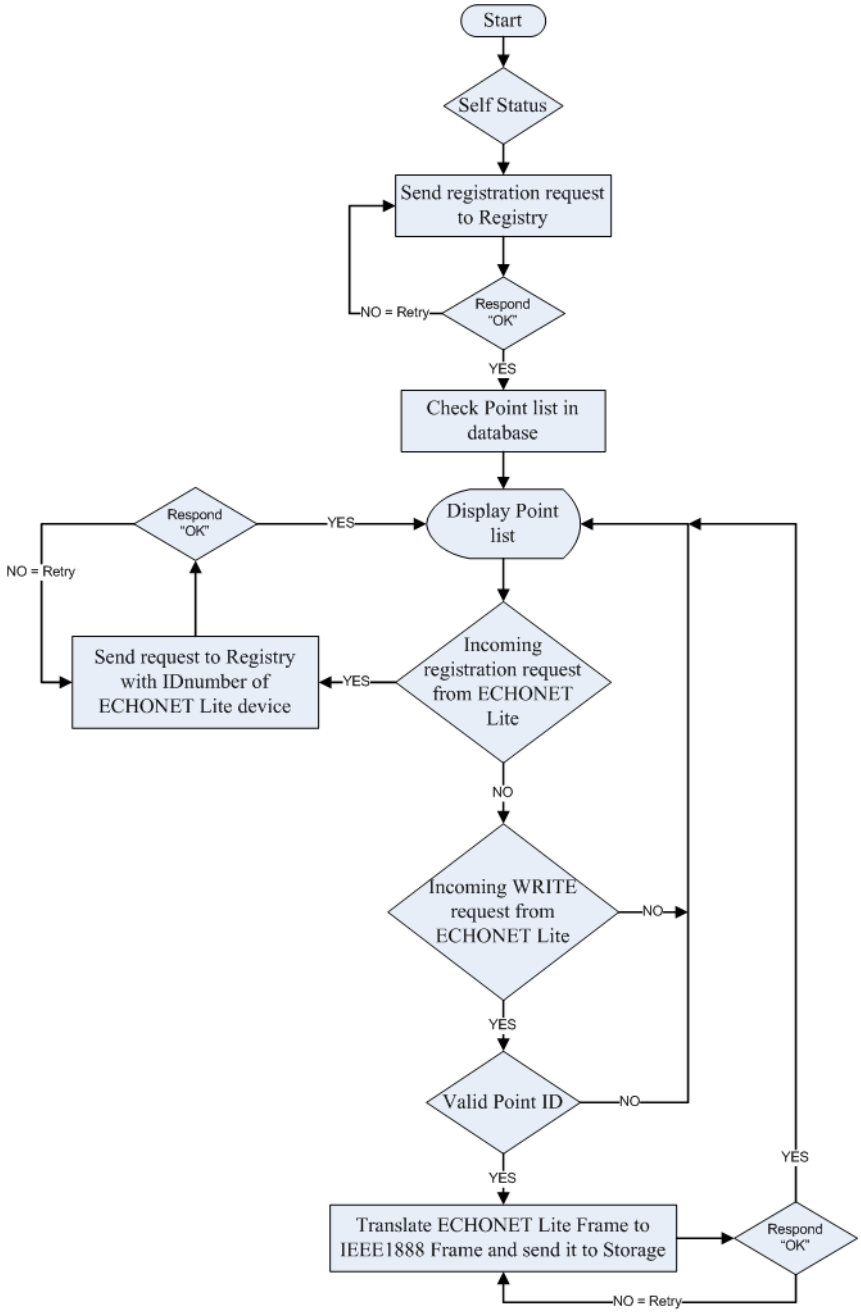
การทำงานของตัวจำลองรีจิสทรีเป็นไปดังรูปที่ 22 คือ รีจิสทรีจะรอให้มีคำขอลงทะเบียนหรือคำขอสืบค้นถูกส่งเข้ามา จากนั้นรีจิสทรีจะทำการตรวจสอบว่า Point ID ที่ถูกส่งเข้ามานั้นมีอยู่ในฐานข้อมูลหรือไม่ แล้วจึงส่งข้อความตอบกลับว่าคำขอที่ส่งมานั้นสำเร็จหรือหรือล้มเหลว



รูปที่ 22 แผนภูมิการทำงานของรีจิสทรี (Registry)

3.2.2 แผนภูมิการทำงานของเกตเวย์ (Gateway)

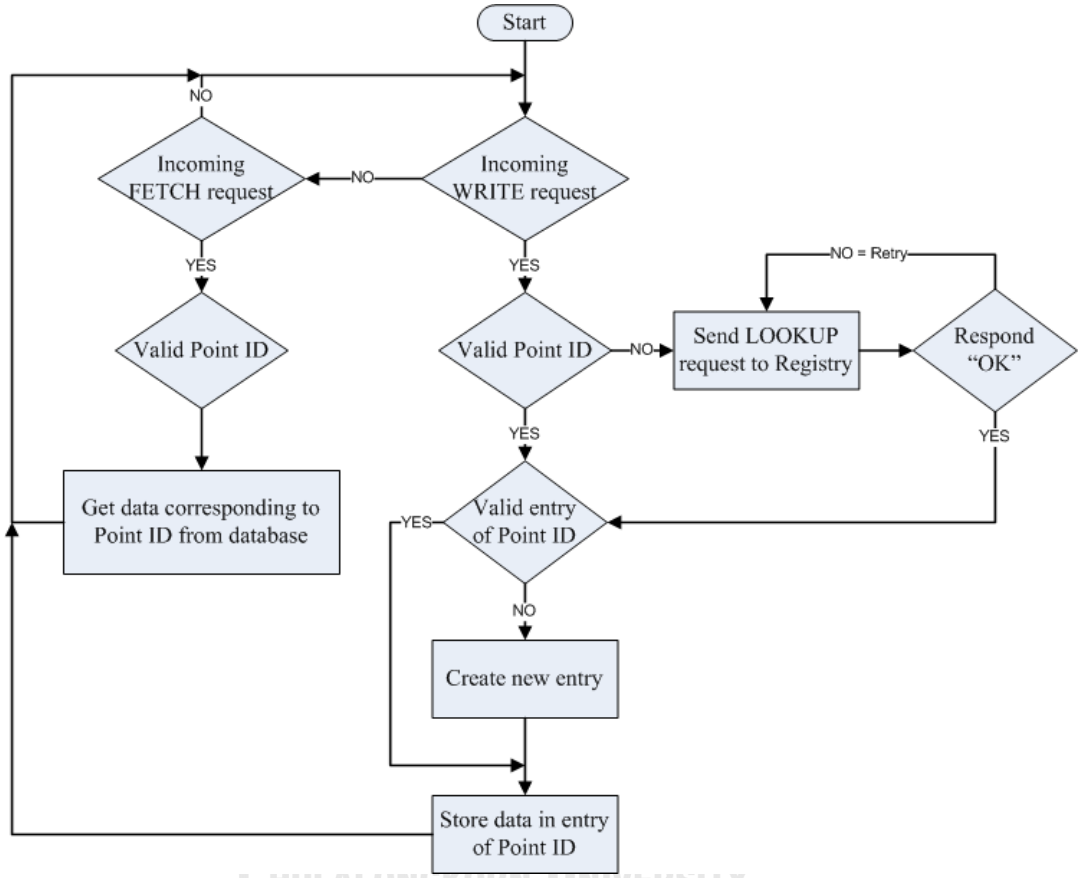
การทำงานของโปรแกรมจำลองเกตเวย์เป็นไปตามรูปที่ 23 ตัวจำลองเกตเวย์มีหน้าที่อยู่ 2 อย่าง คือ ส่งคำขอลงทะเบียนของอุปกรณ์ ECHONET Lite ไปยังรีจิสทรี และ ส่งข้อมูลที่เปลี่ยนแปลงไปของอุปกรณ์ ECHONET Lite ไปเก็บยังฐานเก็บข้อมูล



รูปที่ 23 แผนภูมิการทำงานของเกตเวย์ (Gateway)

3.2.3 แผนภูมิการทำงานของฐานเก็บข้อมูล (Storage)

การทำงานของตัวจำลองฐานเก็บข้อมูลเป็นไปดังรูปที่ 24 โดยตัวจำลองฐานเก็บข้อมูลจะรอรับคำขอ WRITE ข้อมูลจากเกตเวย์หรือแอพพลิเคชั่น และ คำขอ FETCH ข้อมูลที่มีอยู่ในฐานเก็บข้อมูลจากแอพพลิเคชั่นหรือโปรแกรมอื่นๆ



รูปที่ 24 แผนภูมิการทำงานของฐานเก็บข้อมูล (Storage)

3.2.4 แผนภูมิการทำงานของอุปกรณ์ ECHONET Lite

โปรแกรมที่จะถูกนำไปใช้กับอุปกรณ์ ECHONET Lite ถูกเขียนขึ้นด้วย Arduino IDE เพื่อนำไปใช้งานกับบอร์ดควบคุม NodeMCU V3 ที่จะกล่าวถึงในหัวข้อที่ 3.3.1 โดยภายในโปรแกรมมีตัวแปรที่สำคัญดังต่อไปนี้

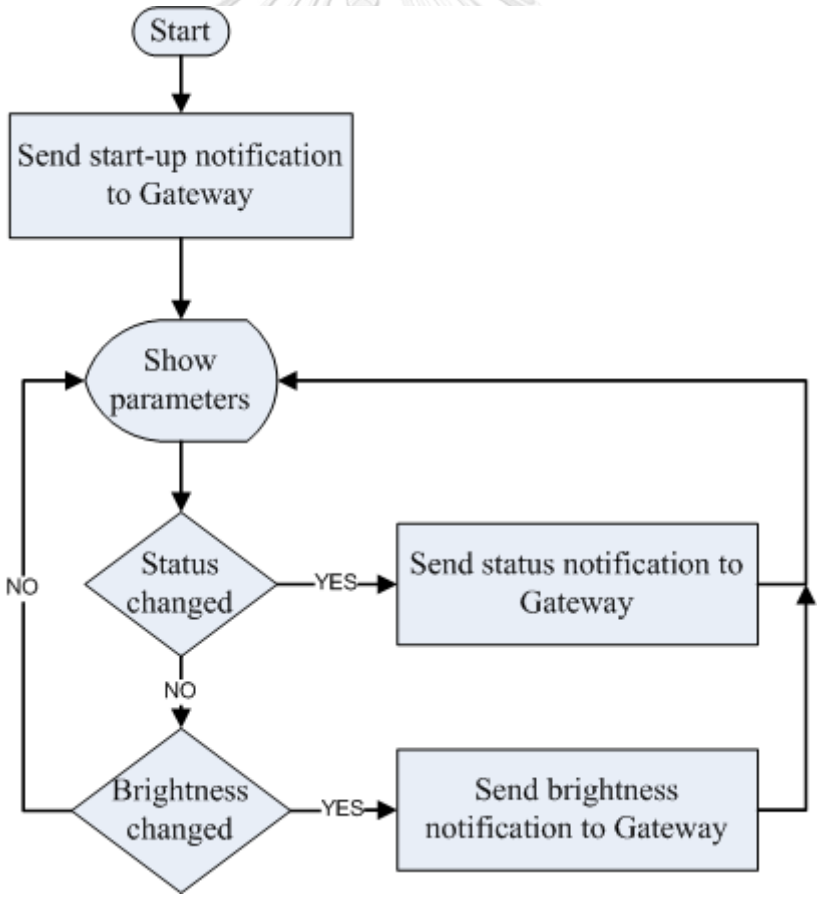
IDnumber – เก็บค่าระบุชื่ออุปกรณ์ ซึ่งแต่ละอุปกรณ์ในระบบจะไม่ซ้ำกัน

Location – เก็บค่าระบุที่อยู่ในการติดตั้งอุปกรณ์

Status – เก็บค่าสถานะการทำงานของอุปกรณ์

Brightness – เก็บค่าความสว่างของอุปกรณ์

โดยโปรแกรมที่ถูกออกแบบนี้มีการทำงานดังแผนภูมิในรูปที่ 25



รูปที่ 25 แผนภูมิการทำงานอุปกรณ์ ECHONET Lite

3.3 ระบบฮาร์ดแวร์

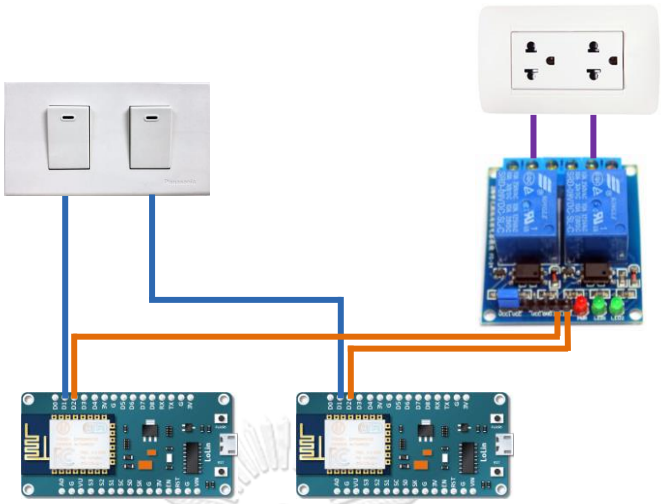
3.3.1 อุปกรณ์ ECHONET Lite

อุปกรณ์ ECHONET Lite ใช้บอร์ด NodeMCU V3 ในการทำงาน ซึ่งมีคุณลักษณะดังต่อไปนี้

- Supply: 4 – 12 V
- Based on ESP8266
- CH340 USB to serial interface
- 802.11 b/g/n Wi-Fi standards
- GPIO 12 channels
- UART 2 channels
- ADC 1 channel

รูปที่ 26 บอร์ด NodeMCU V3

บอร์ด NodeMCU V3 มีลักษณะดังรูปที่ 26 โดยการสร้างอุปกรณ์ ECHONET Lite ในงานวิจัยนี้ จะใช้อุปกรณ์เพิ่มเติมคือ สวิตช์, รีเลย์ และ เต้ารับ โดย NodeMCU V3 แต่ละตัวจะมีอินพุตคือ สวิตช์ 1 ตัว และมีเอาท์พุตเป็นรีเลย์ 1 ตัว ดังรูปที่ 27



รูปที่ 27 วงจรของอุปกรณ์ ECHONET Lite

เมื่ออุปกรณ์ ECHONET Lite เริ่มทำงาน NodeMCU V3 ที่เป็นบอร์ดควบคุมจะทำการอ่านค่าสวิตช์ และส่งข้อความแจ้งไปให้เกตเวย์ ในรูปแบบ ECHONET Lite Frame Format เพื่อให้เกตเวย์รับรู้ว่ามีอุปกรณ์ตัวใหม่ถูกเปิดใช้งานและทำการตรวจสอบว่าอุปกรณ์นี้ได้ถูกลงทะเบียนกับรีจิสทรีแล้วหรือไม่ จากนั้นหากสวิตช์มีการเปลี่ยนแปลง (เปิด-ปิด) NodeMCU V3 จะบันทึกค่าสถานะของสวิตช์และควบคุมรีเลย์จาก ปกติเปิด (Normally Open) ให้เป็น ปกติปิด (Normally Close) หรือ ปกติปิด เป็น ปกติเปิด ตามสถานะของสวิตช์ที่อ่านได้ ซึ่งหน้าสัมผัสของรีเลย์จะถูกต่อเข้ากับสายไลน์ของเต้ารับ เมื่อนำอุปกรณ์ไฟฟ้า เช่น หลอดไฟ LED มาต่อกับเต้ารับ เมื่อหน้าสัมผัสเป็นปกติปิด จะส่งผลให้อุปกรณ์ไฟฟ้าทำงาน

ถึงแม้ว่าอุปกรณ์ ECHONET Lite จะทำการลงทะเบียนกับรีจิสทรีไม่สำเร็จ ตัวอุปกรณ์ก็จะยังสามารถทำงานได้โดยปกติ เพียงแต่ข้อมูลสถานะจะไม่สามารถถูกส่งไปเก็บเอาไว้ในฐานข้อมูล

3.3.2 Raspberry Pi B+ และ เกตเวย์ (Gateway)

บอร์ด Raspberry Pi B+ มีลักษณะดังรูปที่ 28 มีการทำงานบนระบบปฏิบัติการที่มีพื้นฐานจาก Linux เช่น NOOBS หรือ Raspbian ซึ่งในงานวิจัยนี้จะใช้ระบบปฏิบัติการ Raspbian ในการทำงาน โดยบอร์ด Raspberry Pi B+ มีคุณลักษณะดังต่อไปนี้

- Broadcom BCM2835 SoC, 700 MHz Low Power ARM1176JZFS Applications Processor
- Dual Core VideoCore IV® Multimedia Co-Processor
- 512MB SDRAM
- 10/100 BaseT Ethernet socket
- Extended 40-pin GPIO header
- Full-size HDMI
- 4 USB 2.0 ports
- Micro SD port for loading your operating system and storing data
- Micro USB socket 5V, 2A

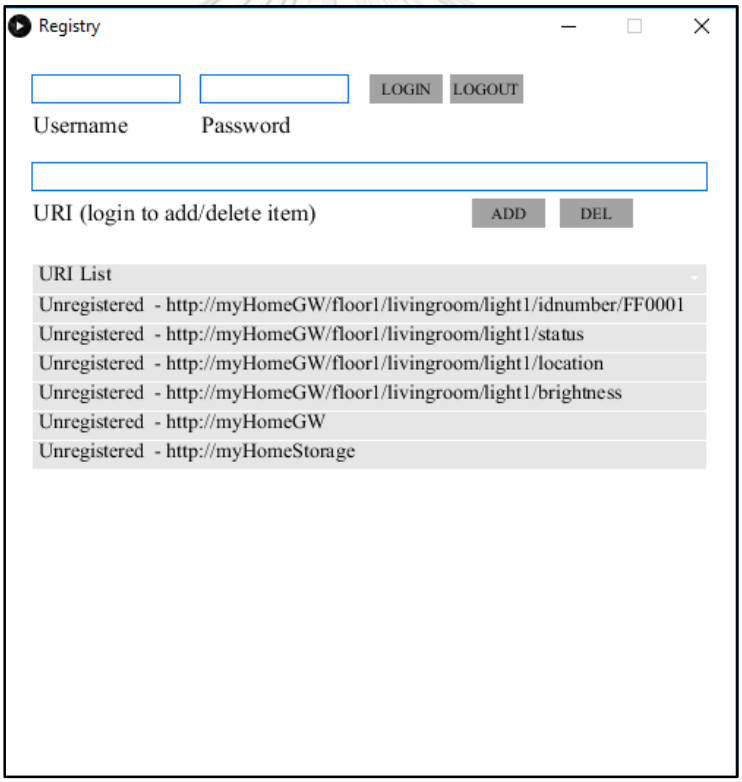
รูปที่ 28 บอร์ด Raspberry Pi B+

การทำงานของเกตเวย์บนบอร์ด Raspberry Pi B+ คือ เมื่อทำการเปิดใช้งาน โปรแกรมเกตเวย์จะส่งคำขอลงทะเบียนและ Point ID ของเกตเวย์ไปยังรีจิสทรี หากการลงทะเบียนสำเร็จ เกตเวย์จะรอรับข้อความจากอุปกรณ์ ECHONET Lite หรือ แอพพลิเคชั่นอื่นๆ หากข้อความที่ได้รับนั้นมาจากอุปกรณ์ หรือ แอพพลิเคชั่นที่ไม่มีอยู่ในรายชื่อ Point ID หรือ Point Set ที่ถูกเก็บไว้ เกตเวย์จะทำการส่งคำขอสืบค้นไปยังรีจิสทรี เพื่อตรวจสอบว่า Point ID หรือ Point Set ที่ได้รับข้อมูลมานั้นมีอยู่ในฐานข้อมูลหรือไม่ หากไม่มีอยู่ในฐานข้อมูลเกตเวย์จะไม่สนใจข้อความที่ได้รับทันที แต่หากมีอยู่ในฐานข้อมูลเกตเวย์จะส่งข้อมูลต่อไปยังฐานเก็บข้อมูล หรือส่งข้อมูลที่แอพพลิเคชั่นต้องการ

บทที่ 4
การทดลองและผลการทดสอบ

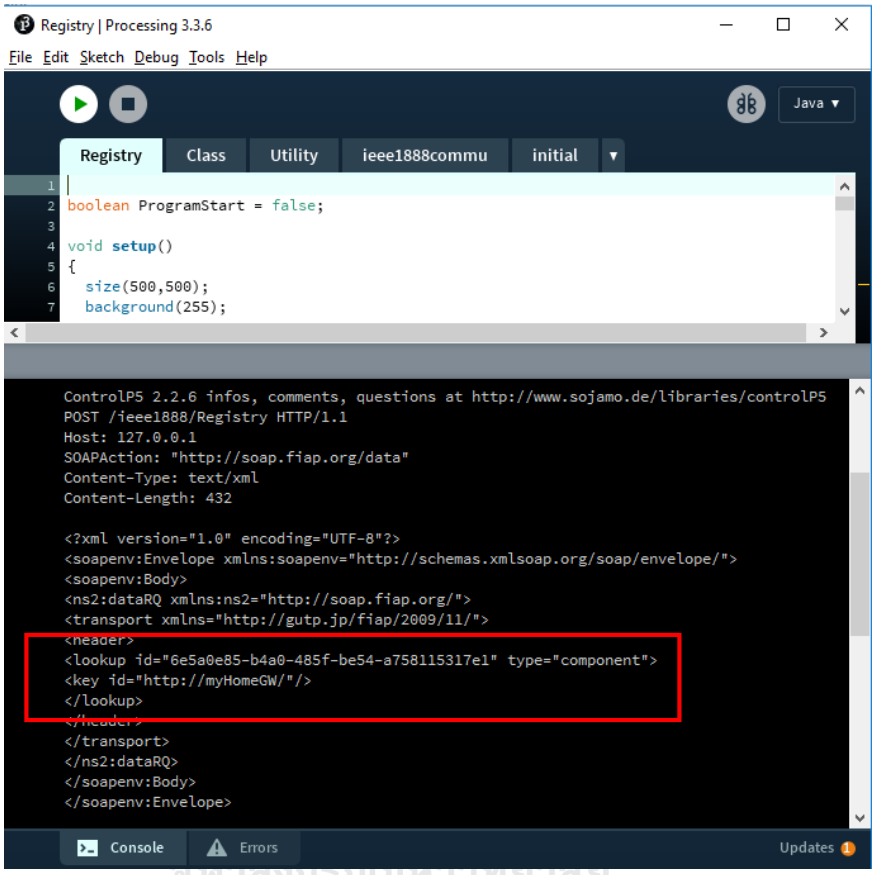
4.1 การทดสอบการทำงานของรีจิสทรี (Registry)

โปรแกรมรีจิสทรีที่ถูกเขียนด้วยโปรแกรม Processing เป็นดังรูปที่ 29 โดยรีจิสทรีจะประกอบด้วย 3 ส่วนหลักที่สำคัญคือ ส่วน Login, ส่วน Add-Delete และ ส่วน URI List โดยในส่วน Add-Delete นั้นจะต้องทำการ Login ด้วย Username และ Password ที่ถูกกำหนดเอาไว้จึงจะสามารถใช้งานได้ ผู้ใช้สามารถกรอกชื่อ Point ID หรือ Point Set ที่ต้องการในช่องว่างและกดปุ่ม ADD เพื่อเพิ่มเข้าไปใน URI List โดย Point ID และ Point Set จะถูกเก็บเป็นไฟล์ text ชื่อ uri_list.txt



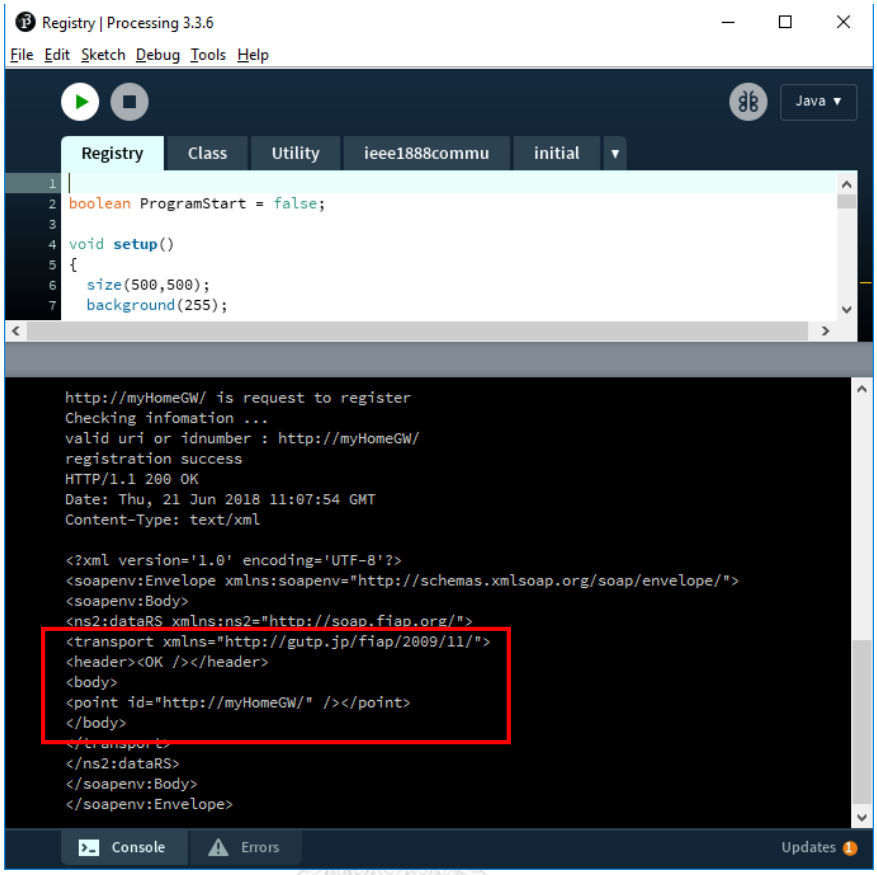
รูปที่ 29 โปรแกรมรีจิสทรีในมาตรฐาน IEEE1888

เมื่อรีจิสทรีได้รับคำขอลงทะเบียนจากอุปกรณ์ เช่น เกตเวย์ ดังรูปที่ 30 รีจิสทรีจะทำการตรวจสอบภายในไฟล์ uri_list.txt เพื่อตรวจสอบว่า Point ID ของอุปกรณ์ที่ส่งคำขอเข้ามานั้นมีอยู่หรือไม่



รูปที่ 30 รีจิสทรี ได้รับคำขอลงทะเบียนจาก “http://myHomeGW”

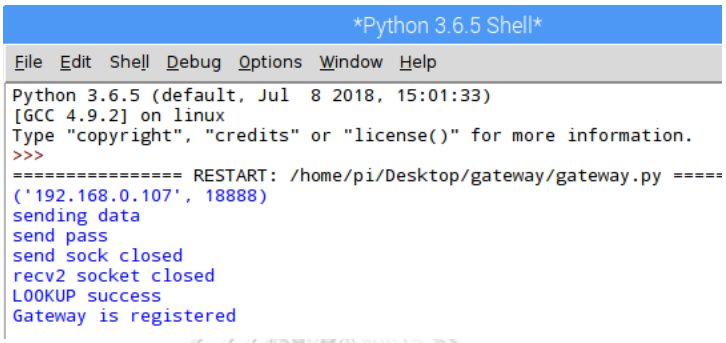
หากตรวจสอบแล้วพบว่ามีอยู่ รีจิสทรีจะทำการส่งคำตอบกลับไปยังอุปกรณ์ที่ส่งคำขอมาดังรูปที่ 31 และจะทำการบันทึกว่า Point ID นี้ได้ทำการลงทะเบียนแล้ว หากมีอุปกรณ์อื่นๆที่มี Point ID เดียวกันส่งคำขอลงทะเบียนมาหลังจากนี้ รีจิสทรีจะไม่ยอมรับให้ลงทะเบียนสำหรับอุปกรณ์นั้น



รูปที่ 31 รีจิสทรี ส่งคำตอบกลับไปที่เกตเวย์ชื่อ http://myHomeGW

4.2 การทดสอบการทำงานของเกตเวย์ (Gateway)

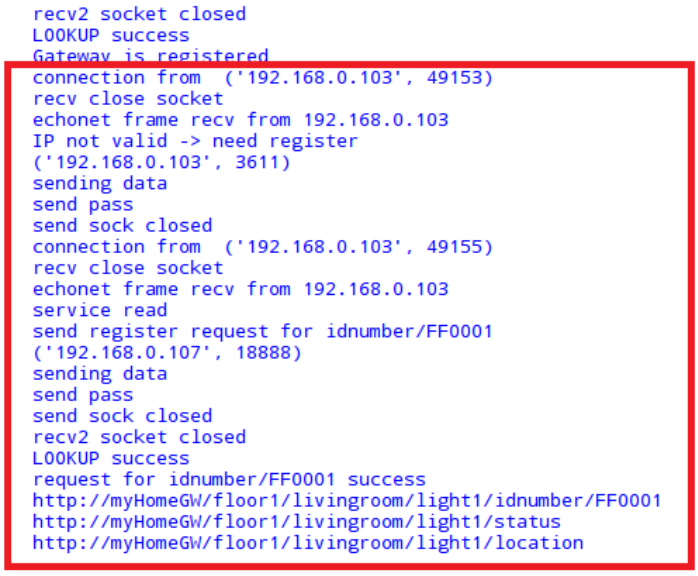
โปรแกรมเกตเวย์ถูกเขียนโดยโปรแกรม Python 3.6 และทำงานบนบอร์ด Raspberry Pi B+ เมื่อเกตเวย์เริ่มทำงาน เกตเวย์จะส่งคำขอลงทะเบียนไปยังรีจิสทรีเพื่อขอลงทะเบียนเข้ามาในระบบ เมื่อรีจิสทรีได้รับคำขอลงทะเบียนแล้วจะทำการตรวจสอบและส่งข้อความตอบกลับว่าคำขอลงทะเบียนของเกตเวย์นั้นสำเร็จหรือไม่ โดยในรูปที่ 32 แสดงถึงการลงทะเบียนสำเร็จของเกตเวย์



รูปที่ 32 เกตเวย์ทำการลงทะเบียนกับรีจิสทรีสำเร็จ

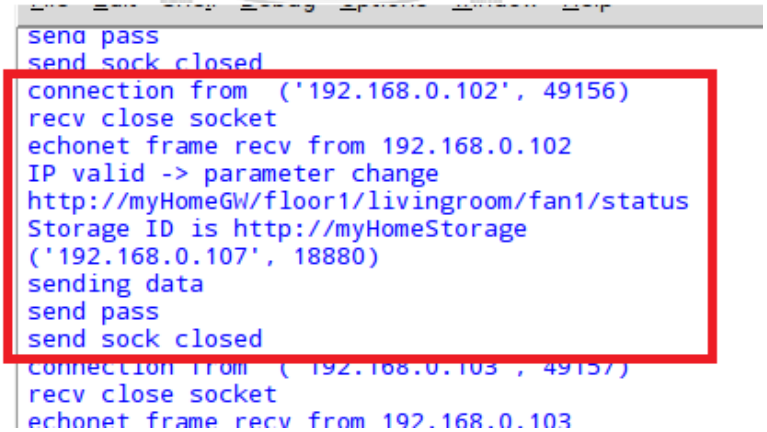
หากเกตเวย์ไม่สามารถลงทะเบียนได้สำเร็จ เกตเวย์ก็จะไม่สามารถทำหน้าที่ในการแปลงข้อมูลจากอุปกรณ์ ECHONET Lite ให้เป็นข้อมูลของ IEEE1888 ได้และในทางกลับกันด้วย แต่โดยปกติแล้วการลงทะเบียนจะสำเร็จและเกตเวย์สามารถส่งผ่านข้อมูลไปยังฐานข้อมูลได้

หลังจากทำการลงทะเบียนแล้ว หากมีอุปกรณ์ ECHONET Lite ถูกเชื่อมต่อเข้ามายังเกตเวย์ เกตเวย์จะทำการตรวจสอบว่ามี IDnumber ของอุปกรณ์นั้นๆเก็บอยู่หรือไม่ หากยังไม่มีแสดงว่าอุปกรณ์นั้นยังไม่ได้ทำการลงทะเบียนกับรีจิสทรี ดังนั้นเกตเวย์จะทำการส่งคำขอลงทะเบียนพร้อมกับ IDnumber ของอุปกรณ์ ECHONET Lite ไปยังรีจิสทรี และหากตรวจสอบแล้วว่า IDnumber ที่รีจิสทรีได้รับมีอยู่ในฐานข้อมูล รีจิสทรีจะทำการส่งข้อความตอบกลับพร้อมกับ Point ID ต่างๆที่เกี่ยวข้องกับ IDnumber นั้นๆดังรูปที่ 33 จากนั้นเกตเวย์จะบันทึกค่า Point ID หรือ Point Set, IDnumber และ IP Address ของอุปกรณ์ เพื่อนำมาใช้ในการแปลงข้อมูลในภายหลัง



รูปที่ 33 เกตเวย์ส่งคำขอลงทะบียนอุปกรณ์ ECHONET Lite ไปยังรีจิสทรี

จากนั้นหากเกตเวย์ได้รับข้อความแจ้งจากอุปกรณ์ ECHONET Lite ว่าค่าพารามิเตอร์มีการเปลี่ยนแปลง เกตเวย์จะตรวจสอบหมายเลข IP และชนิดของค่าพารามิเตอร์ที่เปลี่ยนแปลงไป จากนั้นจึงส่งต่อข้อมูลไปยังฐานเก็บข้อมูลต่อไปดังรูปที่ 34

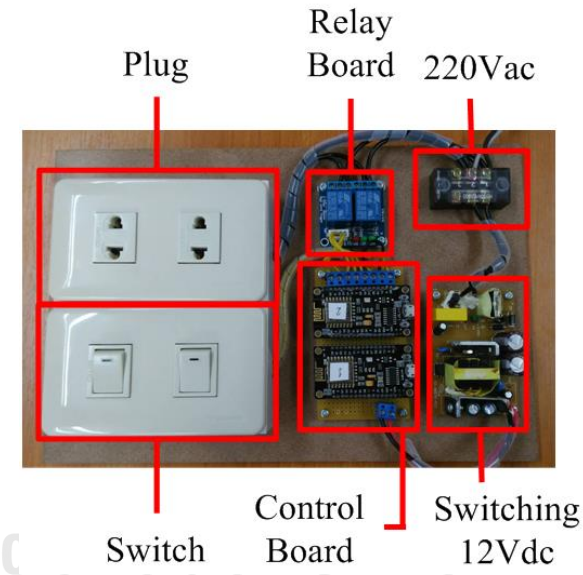


รูปที่ 34 เกตเวย์ส่งค่าพารามิเตอร์ไปยังฐานเก็บข้อมูล

4.3 การทดสอบการทำงานของอุปกรณ์ ECHONET Lite

จากวงจรที่ได้ออกแบบไว้ในหัวข้อ 3.3.1 สามารถสร้างอุปกรณ์ ECHONET Lite โดยมีรายละเอียดดังรูปที่ 35 และใช้หลอดไฟ LED จำนวน 2 ดวง ดังรูปที่ 36 เป็นอุปกรณ์ไฟฟ้าในการทดสอบ โดยเมื่อทำการจ่ายไฟ 220 Vac ให้กับอุปกรณ์ ECHONET Lite บอร์ดควบคุมแต่ละตัวจะมีการรับส่งข้อความเพื่อลงทะเบียนกับเกตเวย์ จากนั้นเมื่อเปิดสวิตช์บอร์ดควบคุมมีการส่งข้อความแจ้งสถานะเปลี่ยนแปลงไปยังเกตเวย์ และเกตเวย์ทำการแปลงข้อความและส่งต่อไปยังฐานเก็บข้อมูล

อุปกรณ์ ECHONET Lite ชุดนี้มีข้อจำกัดคือ ต้องให้บอร์ดควบคุมตัวใดตัวหนึ่งทำงานก่อนเป็นเวลาประมาณ 15 วินาที เนื่องจากการลงทะเบียนต้องใช้เวลาในการแลกเปลี่ยนข้อมูลระหว่างบอร์ดควบคุม, เกตเวย์ และ รีจิสทรี หากบอร์ดควบคุมทำงานพร้อมกันจะทำให้การลงทะเบียนไม่สำเร็จ



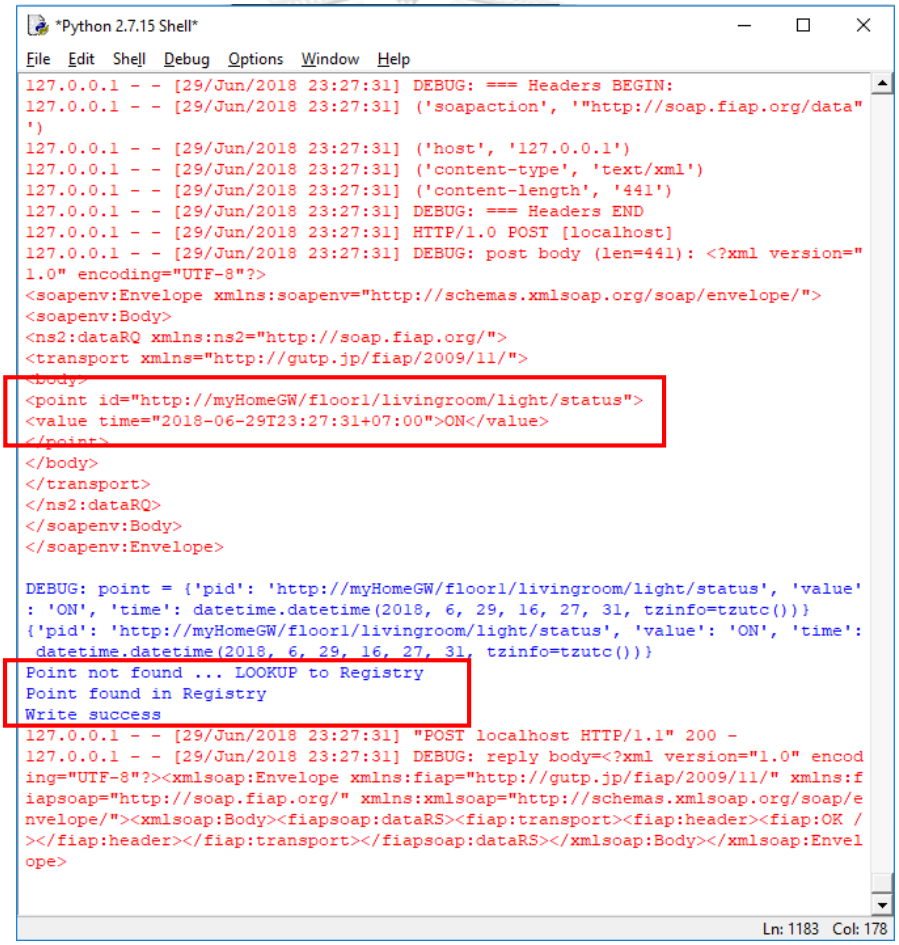
รูปที่ 35 อุปกรณ์ ECHONET Lite

รูปที่ 36 โหลดหลอดไฟ LED

4.4 การทดสอบการทำงานของฐานเก็บข้อมูล (Storage)

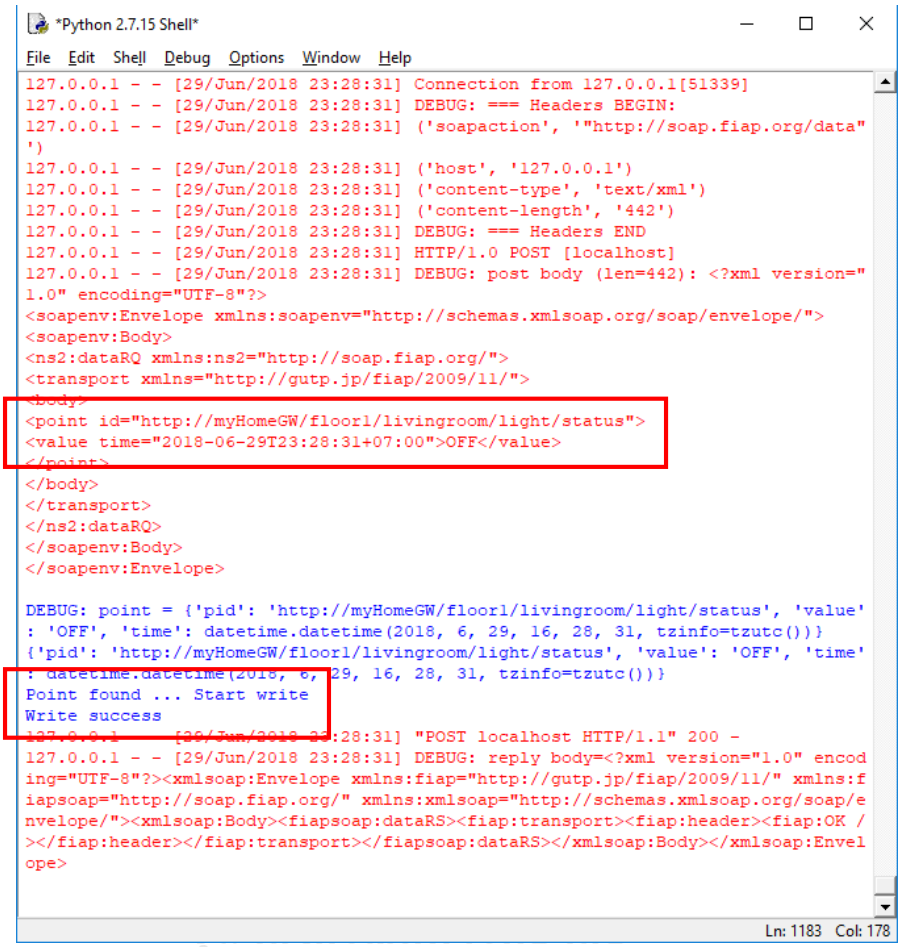
ฐานเก็บข้อมูลถูกเขียนด้วยโปรแกรม Python 3.6 และเชื่อมต่อกับฐานข้อมูล MongoDB เพื่อใช้ในการเก็บข้อมูลที่ถูกส่งมาจากอุปกรณ์อื่นๆ โดยได้มีการนำโปรแกรมจาก [14] มาใช้และเพิ่มเติมส่วนของการทำงานเกี่ยวกับรีจิสทรีลงไปในโปรแกรม

จากรูปที่ 37 เมื่อฐานเก็บข้อมูลได้รับคำขอในการ WRITE ก็จะทำการตรวจสอบว่า Point ID ที่จะทำการ WRITE นั้น เคยมีข้อมูลอยู่ในฐานเก็บข้อมูลแล้วหรือไม่ หากยังไม่มี ฐานเก็บข้อมูลจะทำการส่งคำขอสืบค้นไปยังรีจิสทรี เพื่อตรวจสอบว่ามี Point ID อยู่ในฐานข้อมูลหรือไม่ เมื่อตรวจสอบแล้วว่ามีอยู่ในฐานข้อมูลของรีจิสทรี ฐานเก็บข้อมูลก็จะทำการสร้างพื้นที่จัดเก็บข้อมูลของ Point ID นั้นขึ้นมาใหม่ภายในฐานข้อมูล MongoDB



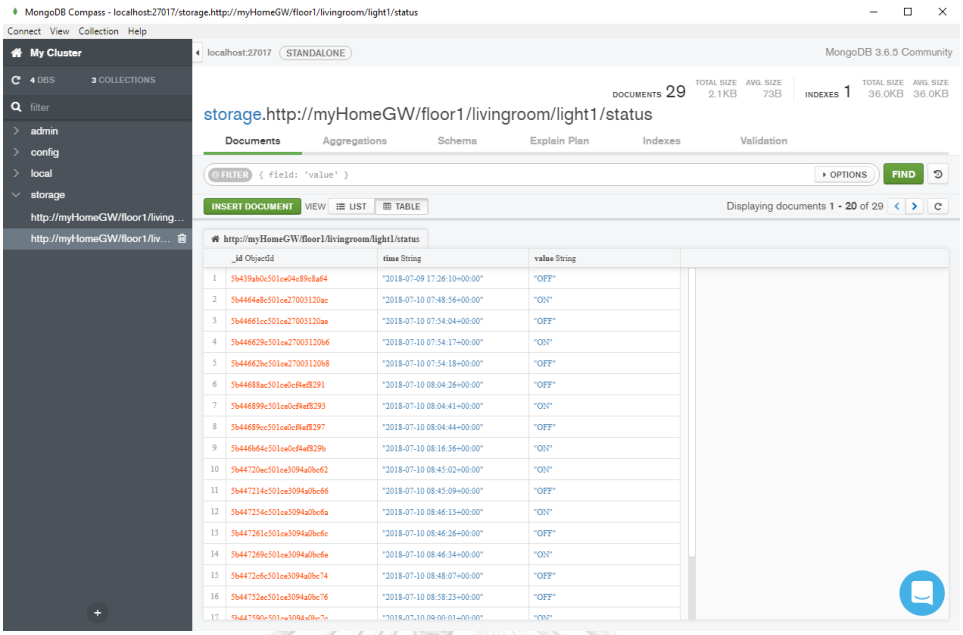
รูปที่ 37 ฐานเก็บข้อมูลตรวจสอบไม่พบ Point ID ในฐานข้อมูล

ถ้าหากว่า Point ID ที่ต้องการ WRITE ข้อมูลนั้นมีอยู่ในฐานเก็บข้อมูลอยู่แล้วนั้น ฐานเก็บข้อมูลก็จะทำการ WRITE ข้อมูลต่อจากข้อมูลเดิมในทันทีดังรูปที่ 38

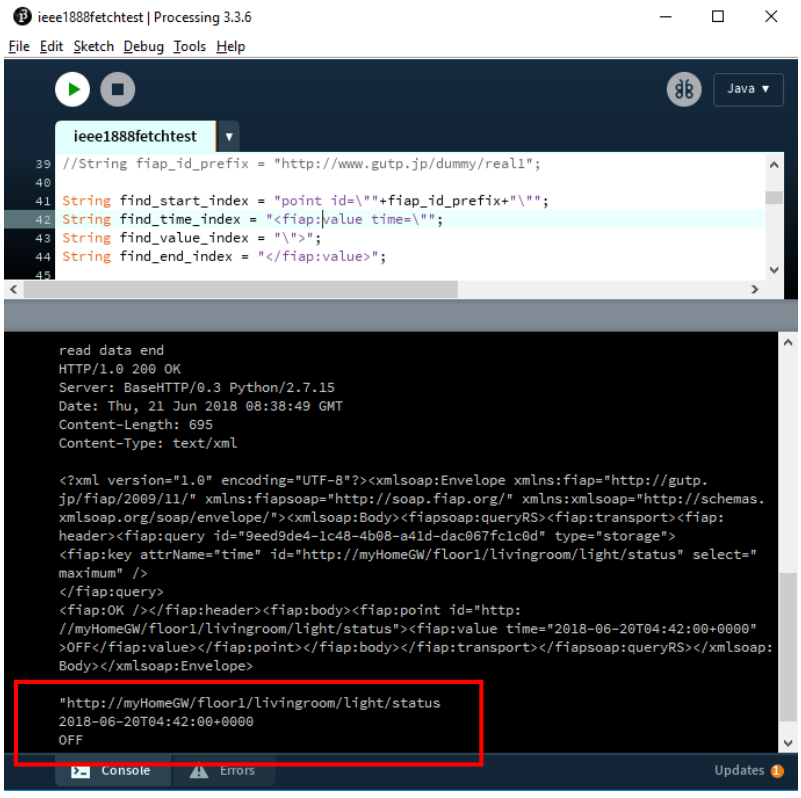


รูปที่ 38 ฐานเก็บข้อมูลตรวจสอบพบ Point ID ในฐานข้อมูล

ผู้ใช้สามารถเรียกดูข้อมูลได้จากโปรแกรม MongoDB Compass ดังรูปที่ 39 หรือสร้างโปรแกรมหรือ แอพพลิเคชั่นเพื่อ FETCH ข้อมูลจากฐานเก็บข้อมูลได้อีกด้วย เช่น ใช้โปรแกรม ieee1888fetchtest ที่ถูกเขียนด้วยโปรแกรม Processing ทำการ FETCH ข้อมูลครั้งสุดท้ายของ Point ID = http://myHomeGW/floor1/livingroom/light/status ดังรูปที่ 40



รูปที่ 39 แสดงข้อมูลที่ถูกเก็บใน MongoDB ด้วย MongoDB Compass



รูปที่ 40 ตัวอย่างการ FETCH ข้อมูลจาก Storage ด้วย Processing

บทที่ 5
สรุปผลงานวิจัย

5.1 สรุป

จากการให้ รีจิสทรี (Registry), เกตเวย์ (Gateway), ฐานเก็บข้อมูล (Storage) และ อุปกรณ์ ECGONET Lite ทำงานร่วมกันเป็นระบบตามที่ได้ออกแบบไว้ ผลที่ได้คือ ฐานเก็บข้อมูล, เกตเวย์ และ อุปกรณ์ ECHONET Lite ที่มีการลงทะเบียนกับรีจิสทรีเท่านั้นที่จะสามารถสื่อสารกับภายในระบบได้ ซึ่งเป็นไปตามจุดประสงค์ที่ได้ออกแบบเอาไว้ และส่งผลให้ระบบที่ทำการทดสอบมีความปลอดภัยในด้านข้อมูลมากขึ้น เพราะไม่มีผู้อื่นหรืออุปกรณ์อื่นสามารถเขียนข้อมูลลงในฐานข้อมูลได้นั่นเอง

5.2 ข้อเสนอแนะ

5.2.1 เพิ่มแอพพลิเคชั่นบนคอมพิวเตอร์หรือมือถือ เพื่อใช้อ่านหรือดูข้อมูลจากอุปกรณ์ต่างๆ

5.2.2 ออกแบบแผ่นวงจรสำหรับอุปกรณ์ ECHONET Lite

5.2.3 เพิ่มอุปกรณ์หรือตัวตรวจรู้ในมาตรฐาน ECHONET Lite ให้หลากหลายมากขึ้น

5.2.4 ปรับปรุง GUI ของโปรแกรมรีจิสทรี ให้มีการใช้งานที่หลากหลายขึ้น

รายการอ้างอิง

1. Olivier Hersent, David Boswarthick and Omar Elloumi, *Modbus*, in *The Internet of Things:Key Applications and Protocols*. 2012, Wiley Telecom. p. 376.
2. Jaap Haartsen, et al., *Bluetooth: vision, goals, and architecture.* SIGMOBILE Mob. Comput. Commun. Rev., 1998. **2**(4): p. 38-45.
3. ZigBee Alliance, *ZigBee and Wireless Radio Frequency Coexistence.* White paper, 2007.
4. Echelon, *Introduction to the LonWorks System*, in *Echelon Corporation*. 2003.
5. ANSI/ASHRAE Standard 135-2008, *BACnet: A Data Communication Protocol for Building Automation and Control Networks*, in *American Society of Heating, Refrigeration, and Air-Conditioning Engineers Inc. (ASHRAE)*. 2008.
6. *IEEE Standard for Ubiquitous Green Community Control Network Protocol.* IEEE Std 1888-2014 (Revision of IEEE Std 1888-2011), 2014: p. 1-71.
7. T. Murakami, H. Sugimura and M. Isshiki, *Application of ECHONET Lite which is open standard into energy management system*, in *2016 IEEE International Conference on Consumer Electronics (ICCE)*. 2016. p. 455-458.
8. S. Matsumoto, *Echonet: A Home Network Standard.* IEEE Pervasive Computing, 2010. **9**(3): p. 88-92.
9. C. Ninagawa, H. Yoshida, S. Kondo and H. Otake, *Data transmission of IEEE1888 communication for wide-area real-time smart grid applications*, in *International Renewable and Sustainable Energy Conference (IRSEC)*. 2013. p. 509-514.
10. D. H. Le and W. Pora, *Implementation of smart meter working as IEEE1888-6LoWPAN gateway for the building energy management systems*, in *International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology (ECTI-CON)*. 2014. p. 1-5.
11. C. Sangumpai and C. Aswakul, *Development of Real-Time Interworking between IEEE1888 and ECHONET Lite Standards for Building Energy Management System.* Engineering Journal, 2017. **21**: p. 1-10.

12. H. A. Elmadany, M. Alfonse and M. Aref. *XML summarization: A survey*. in *IEEE Seventh International Conference on Intelligent Computing and Information Systems (ICICIS)*. 2015.
13. *APPENDIX, Detailed Requirements for ECHONET Device Objects*. Available from: https://echonet.jp/wp/wp-content/uploads/pdf/General/Standard/Echonet/Version_2_11_en/spec_v211e_Appendix.pdf.
14. Tanupoo. *Fiapy*. Available from: https://github.com/tanupoo/fiapy.

ภาคผนวก

ประวัติผู้เขียนวิทยานิพนธ์

นาย มนต์ชัย กายาสมบูรณ์ เกิดวันที่ 11 มีนาคม พ.ศ. 2532 ที่จังหวัด กรุงเทพมหานครฯ มารดาชื่อ นาง วัชรี ฉัตรกอบกูล บิดาชื่อ นาย มณเฑียร กายาสมบูณ์

สำเร็จการศึกษาระดับปริญญาวิศวกรรมศาสตรบัณฑิต สาขา วิศวกรรมไฟฟ้า คณะวิศวกรรมศาสตร์ มหาวิทยาลัยพระจอมเกล้าพระนครเหนือ ในปี พ.ศ. 2554

จากนั้นได้เข้าทำงานใน บริษัท ใจอินเวนเตอร์ จำกัด ในปีเดียวกัน และเข้าศึกษาต่อในหลักสูตรวิศกรรมมหาบัณฑิต ภาค วิศวกรรมไฟฟ้า คณะ วิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย ในปี พ.ศ. 2557